Aula 07 - Profs Thiago Cavalcanti e Erick Muzart

Banco do Brasil (Escriturário - Agente de Tecnologia) Banco de Dados - 2023 (Pós-Edital)

Autor:
Thiago Rodrigues Cavalcanti,
Erick Muzart Fonseca dos Santos,
Diego Carvalho

24 de Janeiro de 2023

Índice

#ATENÇÃO

Avisos Importantes

O curso abrange todos os níveis de conhecimento...

Esse curso foi desenvolvido para ser acessível a **alunos com diversos níveis de conhecimento diferentes.** Temos alunos mais avançados que têm conhecimento prévio ou têm facilidade com o assunto. Por outro lado, temos alunos iniciantes, que nunca tiveram contato com a matéria ou até mesmo que têm trauma dessa disciplina. A ideia aqui é tentar atingir ambos os públicos - iniciantes e avançados - da melhor maneira possível..

Por que estou enfatizando isso?

O **material completo** é composto de muitas histórias, exemplos, metáforas, piadas, memes, questões, desafios, esquemas, diagramas, imagens, entre outros. Já o **material simplificado** possui exatamente o mesmo núcleo do material completo, mas ele é menor e bem mais objetivo. *Professor, eu devo estudar por qual material?* Se você quiser se aprofundar nos assuntos ou tem dificuldade com a matéria, necessitando de um material mais passo-a-passo, utilize o material completo. Se você não quer se aprofundar nos assuntos ou tem facilidade com a matéria, necessitando de um material mais direto ao ponto, utilize o material simplificado.

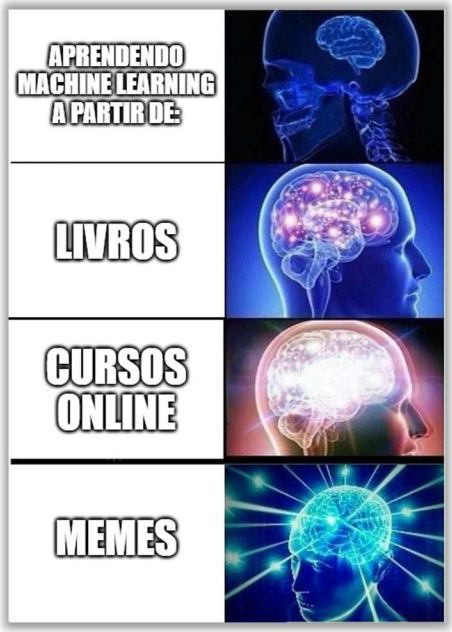
Por fim...

O curso contém diversas questões espalhadas em meio à teoria. Essas questões possuem um comentário mais simplificado porque **têm o único objetivo de apresentar ao aluno como bancas de concurso cobram o assunto previamente administrado**. A imensa maioria das questões para que o aluno avalie seus conhecimentos sobre a matéria estão dispostas ao final da aula na lista de exercícios e **possuem comentários bem mais completos, abrangentes e direcionados.**

APRESENTAÇÃO

Seus lindos, a aula de hoje é sobre Machine Learning! Eu não vou mentir para vocês: essa é uma das aulas mais difíceis que eu já fiz nada vida. E olha que eu já dou aula há quase 10 anos e já escrevi mais de 12.000 páginas sobre os mais diversos assuntos relacionados à tecnologia da informação. Aqui veremos o estado da arte da tecnologia atual, que se mistura bastante com matemática e estatística, mas juro que tentei facilitar ao máximo a vida de vocês. *Legal?* Fechou...

PROFESSOR DIEGO CARVALHO - WWW.INSTAGRAM.COM/PROFESSORDIEGOCARVALHO



Galera, todos os tópicos da aula possuem Faixas de Relevância, que indicam se o assunto cai muito ou pouco em prova. *Diego, se cai pouco para que colocar em aula?* Cair pouco não significa que não cairá justamente na sua prova! A ideia aqui é: se você está com pouco tempo e precisa ver somente aquilo que cai mais, você pode filtrar pelas relevâncias média, alta e altíssima; se você tem tempo sobrando e quer ver tudo, vejam também as relevâncias baixas e baixíssimas. *Fechado?*

RELEVÂNCIA EM PROVA: BAIXÍSSIMA

RELEVÂNCIA EM PROVA: BAIXA

RELEVÂNCIA EM PROVA: MÉDIA

RELEVÂNCIA EM PROVA: ALTA

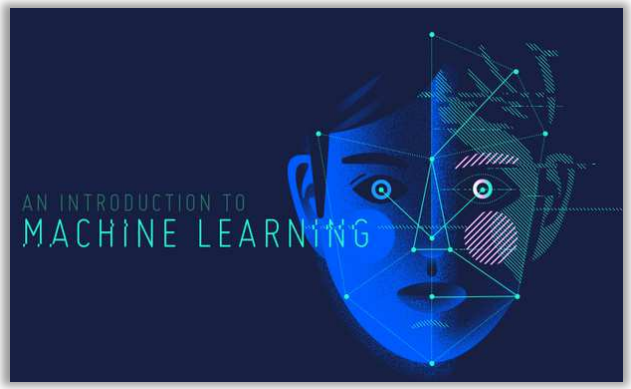
RELEVÂNCIA EM PROVA: ALTÍSSIMA

Além disso, essas faixas não são por banca – é baseado tanto na quantidade de vezes que caiu em prova independentemente da banca e também em minhas avaliações sobre cada assunto...

MACHINE LEARNING

Contextualização Inicial

RELEVÂNCIA EM PROVA: ALTÍSSIMA



Vamos começar falando sobre **Inteligência Artificial (IA)**! Nós – humanos – decidimos denominar a nossa própria espécie como *Homo Sapiens* (Homem Sábio). *Vocês sabem o porquê?* Porque nós consideramos que a nossa inteligência é o que nos diferencia de outros animais. Aliás, a inteligência sempre foi motivo de grande curiosidade em nossa espécie. Durante milhares de anos, nós procuramos entender como funcionava o pensamento e a inteligência humana.

De fato, esse é um tópico muito intrigante! *Vocês já pararam para pensar como um mero punhado de matéria orgânica pode perceber, compreender, prever e manipular um mundo muito maior e mais complicado que ela própria?* Pois é! Após a segunda guerra mundial, esse estudo se intensificou com o surgimento do campo de inteligência artificial, que buscava não apenas compreender, mas também construir coisas inteligentes.

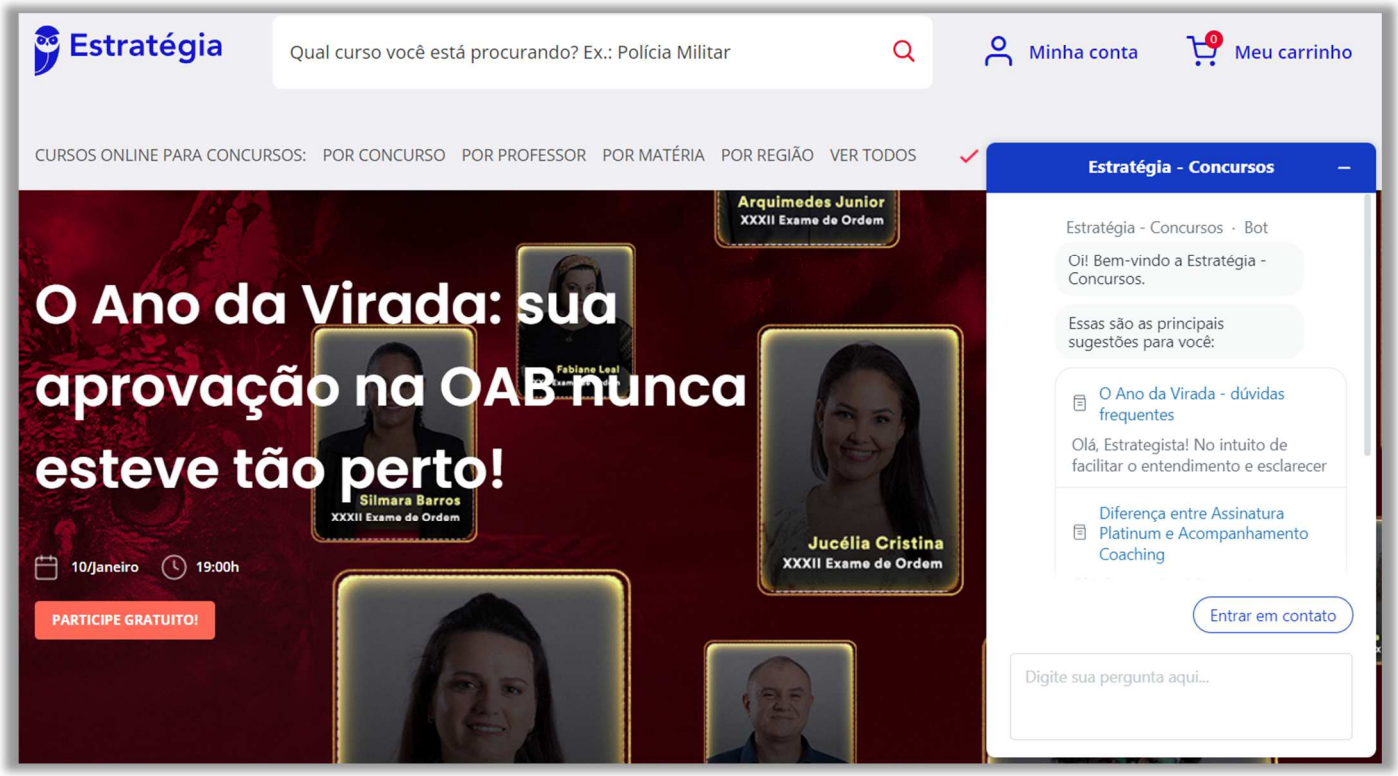
Seu estudo começou após o fim da 2ª guerra mundial por meio do genial Alan Turing (sim, aquele encenado no filmaço The Imitation Game), que estava fascinado com a ideia de que um computador poderia eventualmente se tornar inteligente. *Professor, um computador não é inteligente?* Não, longe disso! Albert Einstein dizia que computadores são incrivelmente rápidos, precisos e **burros**! Quando era criança, eu achava que bastava falar para o computador fazer algo que ele faria...

Ora, não é bem assim! Ele não faz absolutamente nada se alguém não o programar detalhadamente sobre o que ele deve que fazer. Cada coisinha que você vê em um computador/smartphone, alguém teve que criar para o computador tão somente processar. *Por que, Diego?* Porque computadores são ótimos em processar dados de maneira absurdamente rápida, mas tudo que ele faz, alguém teve que programá-lo para fazer!

Alan Turing não parava de se perguntar como as gerações futuras iriam saber se um computador havia adquirido inteligência ou não. Ele, então, propôs um teste projetado para fornecer uma definição satisfatória sobre a inteligência de um computador. Ele dizia que um computador passaria em um teste de inteligência se um interrogador humano fizesse um conjunto de perguntas por escrito e as passasse para um computador e para um humano responder.

Ambos responderiam as perguntas também por escrito e repassariam de volta para o interrogador. Se ele não conseguisse distinguir se as respostas escritas vinham de uma pessoa ou de um computador, poderíamos dizer que o computador adquiriu inteligência. Trazendo para os dias

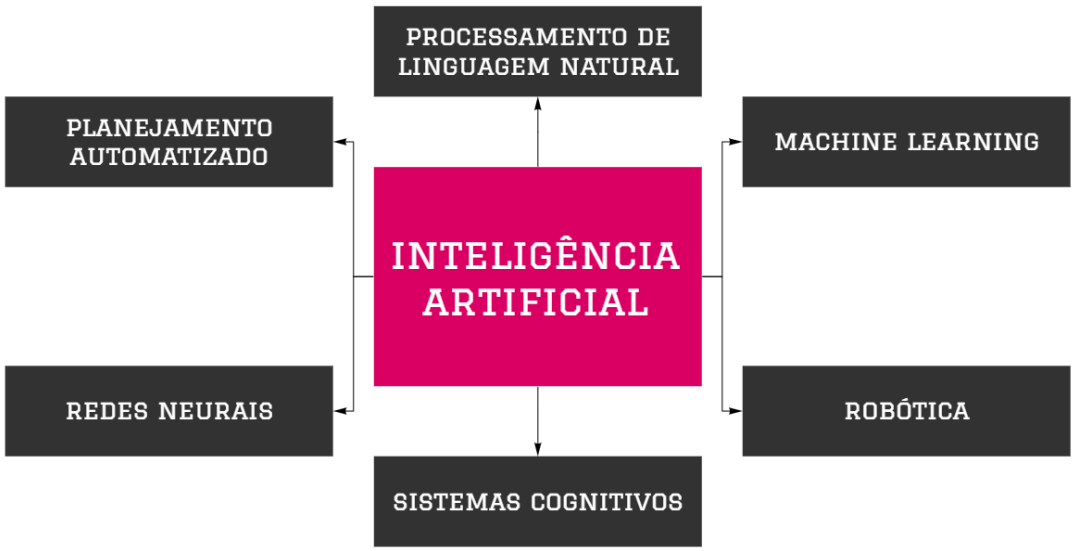
atuais, acho que todos vocês já tiveram que tratar com chatbots (robôs de chat). Aliás, o próprio Estratégia Concursos tem o seu...



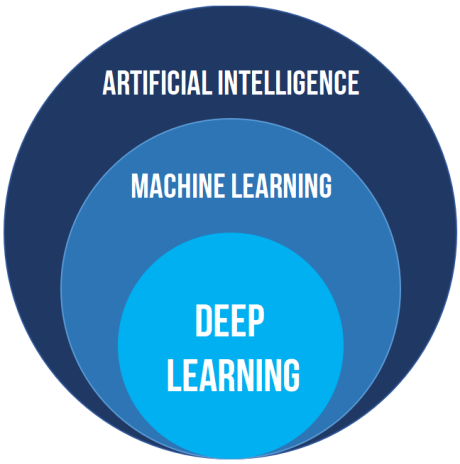
Se esse teste ocorresse nos dias atuais, poderíamos dizer que finalmente chegamos à era dos computadores inteligentes quando tentássemos resolver algum problema ou tirar alguma dúvida via chatbot e não conseguíssemos mais diferenciar se estamos falando com um robô ou com um humano. *Legal, né?* Ainda estamos bem longe disso visto que os chatbots podem ser bem irritantes e, por vezes, não resolverem nossos problemas!

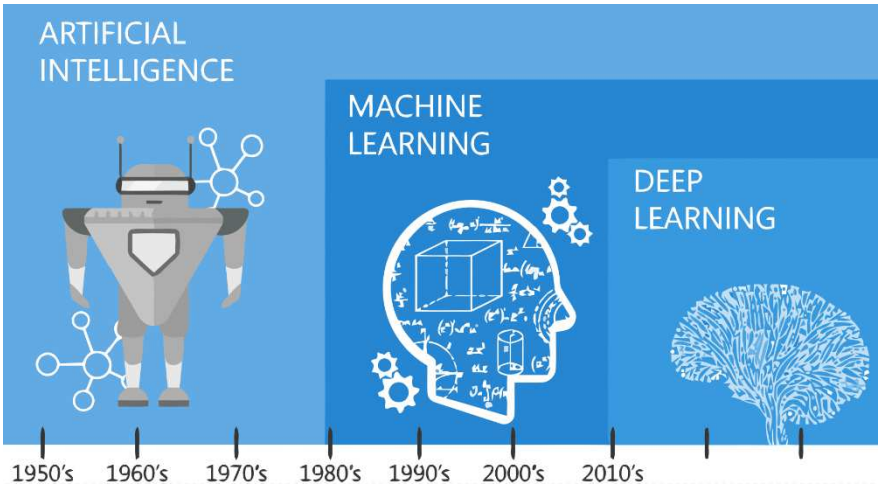
Galera, o termo "inteligência artificial" foi utilizado pela primeira vez em 1956 por um pesquisador americano chamado John McCarthy! De acordo com ele, a inteligência artificial se refere à ciência e à engenharia de construir máquinas inteligentes. *Simples, não?* Outra definição um pouco mais extensa afirma que a inteligência artificial é o campo da ciência da computação que lida com a simulação de comportamento inteligente em computadores. Na imagem ao lado, podemos ver diversas aplicações práticas da inteligência artificial.

Já a imagem abaixo apresenta diversos subcampos da inteligência artificial, tais como: processamento de linguagem natural, redes neurais e aprendizado de máquina.



Agora chegamos no tema principal dessa aula: Machine Learning. Note que se trata de apenas um dos diversos subcampos da inteligência artificial. Vamos ver um pouquinho do histórico...





A inteligência artificial surgiu em meados da década de 1950; o aprendizado de máquina começa a ser estudado no início da década e 1980; e a última década foi marcada pelo aprendizado profundo. *E como eles se diferenciam?* Ora, a inteligência artificial trata de programas que podem sentir, raciocinar, agir e se adaptar a fim de imitar a inteligência humana por meio de diversas técnicas – sendo o aprendizado de máquina uma delas.

Já o aprendizado de máquina trata de algoritmos cujo desempenho melhoram à medida que eles são expostos a mais dados no decorrer do tempo. Por fim, o aprendizado profundo é um subconjunto do aprendizado de máquina em que um software é capaz de treinar a si mesmo para

executar diversas tarefas por meio da utilização de redes neurais para aprender algo a partir de uma quantidade massiva de dados. O foco dessa aula é especificamente Aprendizado de Máquina!

O Aprendizado de Máquina (*Machine Learning*) é a ciência e a arte de programar computadores para que eles possam aprender com os dados. Uma definição um pouco mais formal diria que se trata do campo de estudo que dá aos computadores a capacidade de aprender sem ser explicitamente programado. Eu gosto mais dessa segunda definição porque ela traz uma comparação entre algoritmo de programação tradicional e algoritmo de aprendizado de máquina.

O que é a programação tradicional? Trata-se do processo manual de escrever um conjunto de regras em uma sequência de passos – também chamado de algoritmo – utilizando uma linguagem de programação para que o computador as execute sobre um conjunto de dados de entrada a fim de produzir um conjunto de resultados de saída. Imaginem um software que recebe um conjunto de nomes de alunos e suas respectivas notas em uma prova e retorna um ranking de notas.

Os dados de entrada são nome/nota dos alunos; as regras são escritas por um programador capaz de entender o problema e decompô-lo em passos que um computador entenda utilizando uma linguagem de programação; e o resultado será o ranking de notas. *Simples, não?* Ocorre que, para alguns casos, essa abordagem apresenta diversos problemas. Imaginem um software que recebe uma imagem de entrada qualquer e identifica se há um gato nela ou não. É bem complexo...

Humanos conseguem fazer isso em frações de segundos, mas uma máquina teria bem mais dificuldades. Utilizando a programação tradicional, teríamos diversos gargalos: em primeiro lugar, seria necessário ter um ou mais programadores; em segundo lugar, não basta que ele entenda de programação, ele também deve entender do problema e sugerir uma maneira de resolvê-lo. Ora, criar um conjunto de passos para identificar um gato em uma imagem não é uma tarefa simples...

Uma forma de tentar resolver esse problema é por meio do aprendizado de máquina. Ocorre que ele funciona de uma maneira praticamente inversa à programação tradicional. Nós continuamos entrando com dados, mas – em vez de um programador criar manualmente as regras – são inseridos exemplos de resultados passados. Já a saída de um algoritmo de aprendizado de máquina são justamente as regras. Note que as entidades mudaram de lugar...

No caso do software identificador de gatos, nós continuaríamos inserindo imagens de entrada quaisquer, mas também seriam inseridos diversos exemplos de resultados (isto é, imagens que efetivamente contêm gatos). A saída do algoritmo de aprendizado de máquina seria capaz, por si só, de realizar um mapeamento estatístico entre os dados de entrada e os exemplos de resultados esperados a fim descobrir se há ou não um gato em uma imagem. Em vez de um programador dizer quais são as regras, quem diz é o algoritmo! Diz se não é genial...

Em outras palavras, o algoritmo de aprendizado de máquina consegue extrair regras de identificação de gatos por meio de padrões estatísticos comuns entre os dados de entrada e os resultados esperados. *Professor, isso significa que o algoritmo de aprendizado de máquina retornará um código-fonte?* Não, pessoal... são regras estatísticas na forma de um modelo matemático – composto de diversas funções e parâmetros – capaz de identificar padrões a partir de exemplos.

Vocês notaram que as áreas de tecnologia de informação estatística começaram a despencar em concurso público recentemente? Pois é, esse é um dos grandes motivos – existe um relacionamento íntimo entre essas áreas no contexto de aprendizado de máquina. Voltando: quanto mais exemplos de resultados você oferece, mais o algoritmo é treinado, mais regras são aprendidas e mais ajustado se torna o modelo. Essa etapa do processo de aprendizado de máquina é chamada de **Treinamento**.

Trata-se de uma etapa custosa porque idealmente nós temos que inserir quantidades massivas de exemplos de resultados para que o modelo fique o mais ajustado possível. Após essa fase, nós temos a etapa de **Inferência**, que é bem menos custosa. Ela ocorre quando utilizamos uma

programação bem próxima à programação tradicional com regras aprendidas na etapa anterior e novos dados para gerar inferir resultados.

O cientista de dados será responsável pela etapa de treinamento a fim de gerar regras aprendidas e outros softwares as utilizam como entrada e com novos dados para processar e gerar novos resultados. *Vocês se lembram do exemplo do gato?* Após o cientista de dados treinar o algoritmos e chegar a um conjunto de regras (modelo), outros softwares podem utilizar esse modelo na programação clássica para inferir se uma imagem possui ou não um gato.

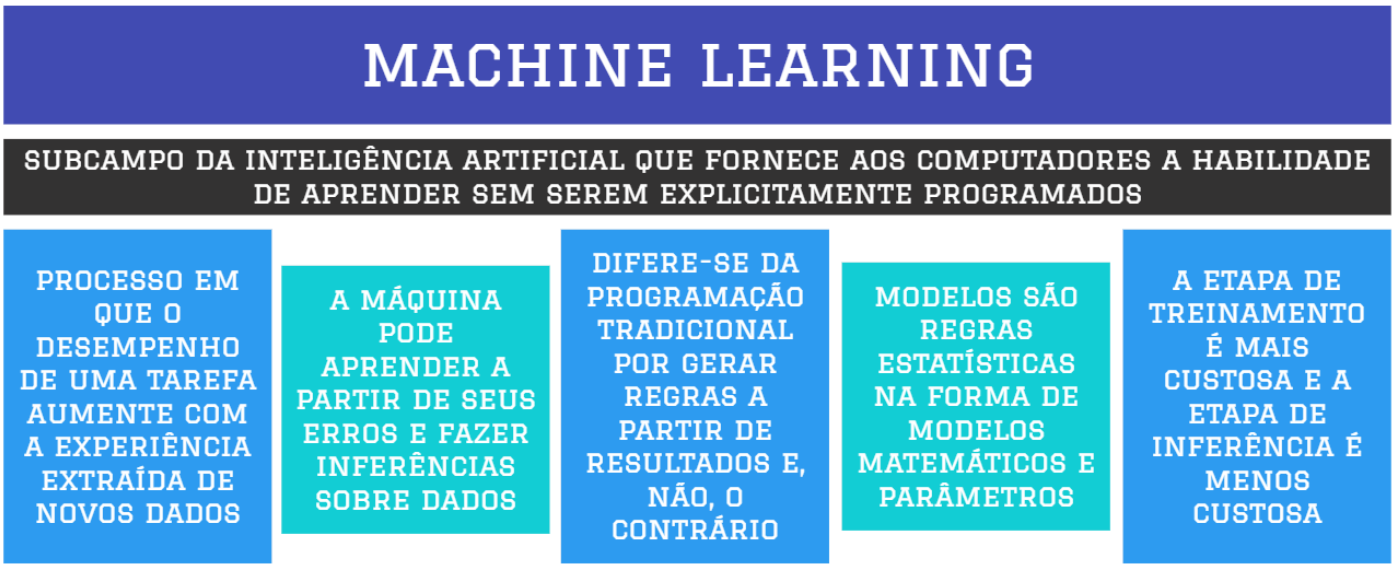
É claaaaaro que nem tudo é perfeito! *Lembra da estatística?* Pois é, a inferência é probabilística e, não, determinística. Isso significa que ela identificará uma alta ou uma baixa probabilidade de ter um gato em uma imagem – ela não vai cravar que existe ou não um gato em uma imagem. *Querem um exemplo?* As quatro imagens a seguir possuem um gato! Ora, se nós temos dificuldades, imaginem uma máquina...




Veja que a máquina não funciona tão diferente dos humanos: nós só sabemos o que é um gato porque já vimos vários exemplos de gatos e de não-gatos. Dessa forma, nosso cérebro consegue abstrair e generalizar o que seria um gato. Agora se fizéssemos esse experimento quando nós fôssemos bem pequenos, nós provavelmente não saberíamos diferenciar porque ainda não fomos bem treinados com vários exemplos e não-exemplos. *Querem ver uma prova?*

Tentem identificar qualquer objeto na imagem ao lago. *Alguém conseguiu identificar qualquer coisa?* Não! *Por que, professor?* Porque essa imagem foi criada especificamente para confundir nosso cérebro. Vejam que ele fica doidinho tentando identificar qualquer coisa, mas ele não teve um treinamento anterior com objetos semelhantes para fazer a comparação. Nós só conseguimos identificar algo porque fazemos uma comparação mental. A máquina começa desse jeito também, mas depois de ser treinada com exemplos e não-exemplos, ela consegue identificar novos objetos. Experimentem mostrar uma fita cacete para uma criança bem pequena hoje em dia. Ela provavelmente nunca viu uma, logo não conseguirá identificar...

Vocês viram a grande utilidade do aprendizado de máquina? Em algumas situações, é extremamente difícil criar regras. Se fôssemos criar um algoritmo utilizando apenas a programação tradicional para identificar gatos em uma imagem, provavelmente teríamos que contratar zoólogos/biólogos – além dos programadores. O aprendizado de máquina supre esse gargalo, não sendo necessária a atuação de zoólogos/biológicos – a máquina aprenderá apenas por meio de dados de treinamento.



Por fim, vamos falar um pouquinho sobre o vocabulário utilizado no contexto de aprendizado de máquina que veremos intensamente na aula a partir das próximas páginas. É importante para que

vocês tenham uma base de conhecimento sobre o assunto, mas infelizmente (ou não) as questões de prova costumam intercambiar esses termos. Logo, eu coloquei mais para vocês terem uma noção, mas não sejam rigorosos na hora da prova. *Fechado?*

TERMO	DESCRIÇÃO
TAREFA	Definição genérica daquilo que se deseja produzir como resultado do modelo preditivo. Ex: classificar um documento em três possíveis categorias ou prever o valor de determinada medida.
TÉCNICA	Conjunto de procedimentos que permite melhorar resultados preditivos. Ex: regularização é uma técnica para prevenir o *overfitting*; *hold-out* é uma técnica de separação de dados para medir o desempenho em generalização de um modelo.
ALGORITMO	Fórmula no sentido lato, que permite relacionar as variáveis independentes para prever a variável dependente. Quando aplicamos (ou treinamos) um algoritmo, temos um modelo treinado. Exemplo: Regressão Linear ou Árvores de Decisão.
MODELO (TREINADO)	Objeto computacional que efetivamente transforma uma observação (variáveis independentes) em uma previsão utilizando um algoritmo específico, instanciado e treinado.

Tipos de Aprendizado

RELEVÂNCIA EM PROVA: ALTÍSSIMA

As aplicações de aprendizagem de máquina abrangem desde jogos passando pela detecção de fraudes até a análise estatísticas da bolsa de valores. É utilizada para construir sistemas de recomendação da Netflix e Spotify que sugerem músicas e/ou vídeos aos usuários com base no seu histórico de acesso, seus favoritos e outros dados; ou sistemas que encontram todos os artigos de notícias similares em um determinado dia.

Ele também pode ser utilizado para categorizar páginas web automaticamente conforme o gênero (esportes, economia, política, bem-estar, etc) ou marcar mensagens de e-mail como spam. Os usos da aprendizagem de máquina são inúmeros e surgem novos todos os dias, mas eles não funcionam todos da mesma maneira. Existem diferentes abordagens de algoritmos de aprendizado de máquina que podem ser classificadas conforme veremos a seguir:

Aprendizado Supervisionado

APRENDIZADO SUPERVISIONADO

Trata-se da abordagem que busca encontrar um conjunto de regras ou funções (também chamadas de modelo) a partir dos dados de treinamento que possam ser utilizadas para prever um rótulo ou valor que caracterize um novo exemplo, com base nos valores de seus atributos de entrada. O termo supervisionado vem da presença de um supervisor externo, que conhece a saída (rótulo) desejada para cada exemplo. Com isso, ele pode avaliar a capacidade do algoritmo de predizer o valor de saída para novos exemplos.

Trata-se de um conjunto de técnicas de aprendizado para treinar um modelo com dados rotulados manualmente, isto é, um especialista/supervisor externo diz qual é a saída esperada para cada dado histórico utilizado no treinamento. São as técnicas mais comumente empregadas no contexto de aprendizado de máquina e geralmente são utilizadas para identificar padrões específicos, prever resultados dado um conjunto de amostras de treinamento.

Em outras palavras, a saída desejada para cada exemplo de entrada já é conhecida no aprendizado supervisionado, isto é, os dados de saída são previamente rotulados. Essa abordagem – também chamada de tarefa de previsão – é bastante semelhante à aprendizagem humana sob a supervisão de um professor. O professor fornece bons exemplos para o aluno memorize, e o aluno então deriva as regras gerais desses exemplos específicos. Vamos ver alguns exemplos...

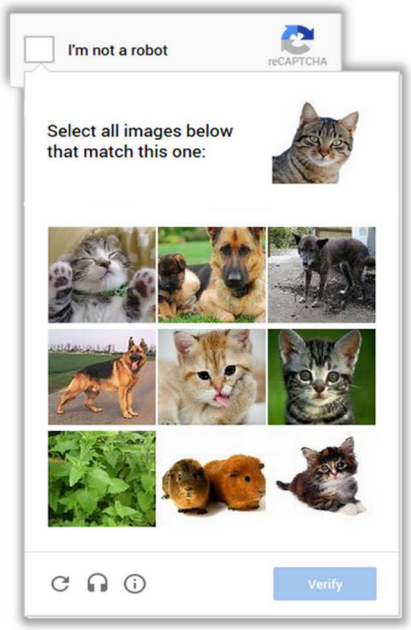
Suponha que você tenha um sobrinho que acabou de fazer dois anos e está aprendendo a falar. Você quer ensiná-lo o que é um cachorro e o que é um gato. *Então, o que você faz?* Você pode mostrar a ele diversos vídeos de cães e depois idealmente mostrar cães e gatos pessoalmente para que ele possa ver melhor. Você, então, pode dizer para ele algumas coisas que sejam semelhantes e diferentes entre cães de gatos.

Por exemplo: ambos possuem quatro patas e uma cauda; cachorros podem ser grandes ou pequenos, gatos geralmente são pequenos; cachorros possuem focinho/boca longos, enquanto gatos têm focinho/bocas pequenos; cachorros latem e gatos miam; cachorros sempre têm pupilas arredondadas, já gatos podem ficar com a pupila arredondada ou em formato de fenda; cachorros têm diferentes formatos de orelha, enquanto quase todos os gatos possuem o mesmo formato.

Finalizada a explicação, agora você leva seu sobrinho de volta para casa e mostra a ele fotos de diferentes cães e gatos. Se ele for capaz de diferenciar o cão do gato, significa que ele aprendeu corretamente. *Então, o que aconteceu aqui?* Você estava lá para guiá-lo até o objetivo de diferenciar entre um cão e um gato; você ensinou a ele todas as diferenças e semelhanças que existem entre um cão e um gato; e você então a testou para ver se ele era capaz de aprender.

Se ele não conseguisse identificar o cachorro e o gato, você teria que analisar os erros e oferecer mais exemplos até que ele finalmente conseguisse identificar cachorros e gatos com maior precisão. Agora note que você atuou como um supervisor/professor e seu sobrinho atuou como o algoritmo que precisava aprender. Você já sabia o que era um cachorro e o que era um gato, e o orientou em seu aprendizado. Por essa razão, a técnica se chama aprendizado supervisionado...

Nesse caso, o processo de aprendizagem de um algoritmo em relação ao conjunto de dados de entrada é acompanhado por um supervisor ou professor acompanhando o processo de aprendizado. O professor sabe quais são as respostas certas, o algoritmo faz as predições sobre o conjunto de dados de entrada e ele vai corrigindo a saída à medida que suas respostas não são de acordo o resultado esperado. Outro exemplo...

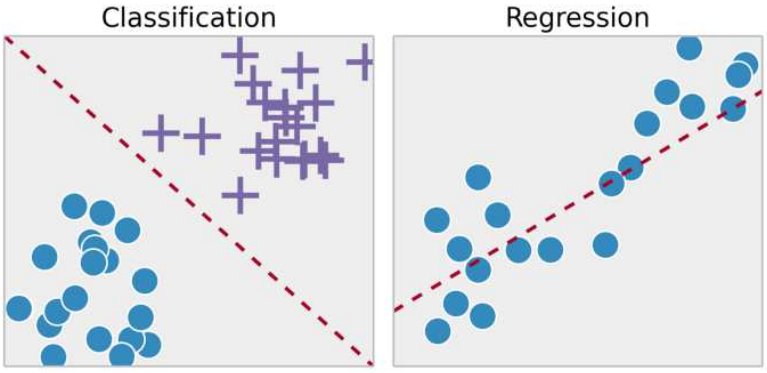


Sabe quando vamos acessar uma página web e aparece uma caixinha falando para nós selecionarmos imagens com algum objeto específico para indicar que não somos um robô? Pois é, o nome daquilo é reCAPTCHA! A verdade é que o objetivo principal desse teste não é verificar se somos robôs, visto que existem maneiras mais simples para isso. O objetivo principal é treinar algoritmos de identificação de imagens. Nesse caso, nós somos os supervisores do algoritmo!

Toda vez que vocês ouvirem falar em aprendizado supervisionado, vocês devem saber que: (1) trata-se da abordagem mais comum de aprendizado de máquina; (2) existe um supervisor ou professor responsável por treinar o algoritmo; e (3) o supervisor conhece de antemão os rótulos[1]. Essa última parte é a mais cobrada em prova, portanto vamos enfatizá-la: o supervisor já conhece de antemão os rótulos.

Logo, o supervisor decide qual é o resultado esperado em cada contexto. Ele pode pegar um conjunto de fotos de pessoas e classificá-las com o rótulo que ele quiser (Ex: alto ou baixo; masculino ou feminino; bonito ou feio; criança ou adulto; loiro, moreno ou ruivo; maior que um determinado valor ou menor que um determinado valor). Quem escolhe o rótulo no aprendizado supervisionado é o supervisor e ele o utiliza para treinar o algoritmo.

Em suma: o aprendizado supervisionado é uma abordagem de aprendizado de máquina em que um supervisor já conhece de antemão o resultado (rótulo/classe) e pode guiar o aprendizado mapeando as entradas em saídas por meio do ajuste de parâmetros em um modelo capaz de prever rótulos desconhecidos. Por fim, é interessante saber que os problemas de aprendizado supervisionado geralmente tratam de uma variável quantitativa ou qualitativa. *Como é, Diego?*



Se os rótulos se referem a um conjunto infinito de valores numéricos contínuos (Ex: > R$100,00 ou < R$100,00), a tarefa se chama regressão. Já se os rótulos se referem a um conjunto finito e não

[1] Rótulo também pode ser chamado de categoria, classe, sinal, variável alvo ou, em inglês, de *target*, *label* ou *tag*.

ordenado de valores categóricos (Ex: Alto ou Baixo), a tarefa se chama classificação. Os modelos supervisionados mais conhecidos são: árvores de decisão, regressão linear, regressão logística, redes neurais, K-Nearest Neighbors (KNN), Support Vector Machines (SVM), etc.

Aprendizado Não-Supervisionado

APRENDIZADO NÃO SUPERVISIONADO

Trata-se da abordagem em que o algoritmo busca encontrar um padrão subjacente nos dados sem a utilização de um supervisor externo para atribuir rótulos ou categorias pré-definidas para as amostras de treinamento. Os algoritmos são formulados de forma que possam encontrar padrões autonomamente com o intuito de explorar dados desconhecidos e encontrar estruturas interessantes ou ocultas nos dados que não eram visíveis anteriormente para os cientistas de dados.

Trata-se de um conjunto de técnicas para treinar um modelo em que não se sabe a saída esperada para cada dado usado no treinamento. Diferentemente do aprendizado supervisionado, você não utiliza rótulos/categorias para as amostras de treinamento. Os algoritmos são formulados de tal forma que podem encontrar estruturas e padrões adequados nos dados por conta própria. Em outras palavras, não existe um supervisor ou professor para rotular os dados...

Sendo assim, o algoritmo identifica as semelhanças nos dados apresentados e reage com base na presença ou ausência dessas tais semelhanças. Essa abordagem – também chamada de tarefa de descrição – permite que o algoritmo aprenda a categorizar dados autonomamente. A ideia aqui não é prever nada e, sim, organizar os dados de alguma forma ou descrever sua estrutura. *Professor, como o algoritmo vai gerar as próprias categorias? E se eu não gostar das categorias escolhidas?*

Excelente pergunta! O processo de aprendizado não supervisionado é mais complexo porque não existe um supervisor para treinar o algoritmo e nem existem categorias pré-definidas. Logo, existe realmente uma chance de o algoritmo gerar categorias completamente diferentes do que você esperava. Existem dois grandes sub-grupos de aprendizado não-supervisionado: Agrupamentos (*Clustering*) e Regras de Associação (*Association Rules*). Vamos ver ambas...

De forma resumida, podemos dizer que as regras de **associação** são um tipo de aprendizado não-supervisionado que permite estabelecer regras capazes de verificar como determinados elementos em um conjunto estão intimamente associados, isto é, se a presença de um elemento implica a presença de outro dentro em uma mesma transação. Os principais modelos são: Apriori e FP-Growth.

Um exemplo clássico utilizado na literatura trata do supermercado. Vá agora até um mercado e fique observando o que as pessoas estão comprando. Tem cidadão que pegou pão e leite; em seguida, entra outro cidadão e pega pão e ovos; posteriormente, entra outro cidadão e compra ovos, leite e açúcar; depois entra mais um cidadão e compra pão, leite e açúcar. *Como as regras de associação se aplicam a esse exemplo?*

As regras de associação permitem verificar se existe um padrão na compra de determinados produtos. Como podemos ver no exemplo, pão, leite, ovos e açúcar são elementos que parecem ter uma associação mais forte do que pão, fígado e uma furadeira. *Alguém discorda?* Pois é, as regras de associação são um tipo de algoritmo de aprendizado não-supervisionado que permite extrair esses padrões para a tomada de decisão. *E o agrupamento?*

De forma resumida, podemos dizer que o **agrupamento** é um tipo de aprendizado não-supervisionado em que se busca encontrar padrões em um conjunto de dados e agrupá-los em subconjuntos que – ao comparar dados de um mesmo grupo – sejam o máximo possível homogêneos/semelhantes e – ao comparar dados de grupos diferentes – sejam o máximo possível heterogêneos/diferentes. Os principais modelos são: k-Means e Agrupamento Hierárquico.

Veremos agora um exemplo clássico de aprendizado não supervisionado por meio de agrupamento. A imagem ao lado contém um conjunto de fotos de diversos cachorros e gatos. Vocês sabem disso porque nós batemos o olho e rapidamente conseguimos identificar que não há outro tipo de animal na imagem. No entanto, a máquina não sabe nada disso – nunca se esqueçam que computadores são burros! Para ela, cada imagem dessas é um conjunto de pixels representados por vetores de números binários. Em suma: podemos afirmar que essa imagem é simplesmente um conjunto de diversos zeros e uns que indicam a cor dos pixels que compõem a imagem.

Se eu rodar um algoritmo de aprendizado não-supervisionado nesse conjunto de imagens e pedir para ele dividi-lo em duas categorias, eu aposto que vocês intuitivamente pensarão que o algoritmo retornará duas categorias: uma categoria de cachorros e outra categoria de gatos. *Acertei?* Pois é, mas você errou! Nós – humanos – sabemos que só há cachorros e gatos na imagem, mas afirmo novamente que a máquina enxerga apenas zeros e uns.

Lembrem-se que o aprendizado não-supervisionado permite avaliar um conjunto de dados de entrada e – sem interferência externa de um supervisor – sugerir categorias. O algoritmo pode, por exemplo, dividir as imagens nas categorias claras e escuras, com fundo verde ou sem fundo verde, maiores ou menores, entre outros. *Vocês se lembram que eu falei que existe a possibilidade de eu não gostar das categorias escolhidas?*

Pois é... o algoritmo não está nem aí para o que eu quero – ele vai buscar padrões nas imagens, a fim de dividi-las em dois grupos de tal modo que as imagens dentro de cada grupo sejam o máximo possível homogêneas e as imagens de grupos diferentes sejam o máximo possível heterogêneas. *E se eu quiser definir de antemão os subgrupos que eu quero dividir as imagens?* Aí você deve usar um algoritmo de aprendizado supervisionado e, não, um algoritmo não-supervisionado.

É preciso entender que cada abordagem é adequada para um tipo de problema e tem suas vantagens e suas desvantagens. O aprendizado não-supervisionado é bem mais barato que o aprendizado supervisionado. Vamos pensar na Amazon! *Vocês imaginaram a quantidade absurda de produtos que tem à venda lá?* Eu fiz uma pesquisa rápida em sua página web e vi que existem 20 categorias principais de produtos.

Só que dentro de cada categoria existem diversas subcategorias que possuem dentro delas várias outras subcategorias. Se tudo isso fosse feito utilizando aprendizado supervisionado, custaria uma fortuna e demoraria anos para concluir. A Amazon teria que contratar uma equipe gigantesca de especialistas em diversas áreas, entregar para eles uma lista com uns 500.000 produtos à venda e pedir para eles criarem rótulos/categorias de modo que contemplasse todos esses produtos.

Uma abordagem mais interessante seria utilizar o aprendizado não-supervisionado, porque ele é muito mais barato e permite lidar com quantidades massivas de dados. *Pode conter erros?* Pode! *Pode gerar categorias indesejadas?* Também pode! Por outro lado, isso é mais barato de resolver. *Professor, esses algoritmos só funcionam com imagem?* Não, eu utilizo exemplos com imagens porque são os mais fáceis de entender, mas ele se aplica a basicamente qualquer formato de dados.

Para finalizar, podemos afirmar que se o aprendizado de máquina fosse representado como uma criança aprendendo a andar de bicicleta, o aprendizado supervisionado seria o pai correndo atrás da bicicleta, segurando-a na vertical e corrigindo a trajetória da criança; e o aprendizado não-supervisionado seria o pai entregar a bicicleta na mão da criança, dar um tapinha nas costas e dizer: "Se vira, garoto!". *Entendido?*

Aprendizado Semi-Supervisionado

APRENDIZADO SEMISUPERVISIONADO

Trata-se da abordagem que abrange técnicas de aprendizado de máquina que usam um conjunto de dados parcialmente rotulado. Este conjunto de dados inclui alguns dados com rótulos e alguns dados sem rótulos. As técnicas de aprendizado semi-supervisionado são usadas para combinar dados rotulados e não rotulados para melhorar a precisão do modelo.

Trata-se de um meio termo entre o aprendizado supervisionado e o não-supervisionado. Nesse caso, utilizamos dados rotulados e não-rotulados para o treinamento. Em geral, utiliza-se uma pequena quantidade de dados rotulados e uma grande quantidade de dados não-rotulados, visto que dados não rotulados são mais baratos e são obtidos com menos esforço. Ela pode ser aplicada para o agrupamento, regras de associação, classificação ou regressão. É isso, aqui não tem segredo!

Aprendizado Por Reforço

APRENDIZADO POR REFORÇO

Trata-se da abordagem que se baseia na recompensa e na punição para ensinar a máquina a realizar tarefas específicas. Ele permite que a máquina aprenda a partir de suas experiências, reforçando ou punindo ações específicas de acordo com o resultado. Utiliza técnicas de reforço positivo, em que a máquina é recompensada por boas ações, e reforço negativo, em que é punida por ações ruins.

Trata-se de um conjunto de técnicas que utilizam tentativa e erro para descobrir decisões ótimas de como interagir com ambiente ou com outros agentes. Ele tem como meta reforçar ou recompensar uma ação considerada positiva e punir uma ação considerada negativa. Um exemplo são os famosos robôs aspiradores! Esses robôs percorre os cômodos de uma casa, identifica obstáculos e armazena quais rotas funcionam melhor para limpar a casa.

Ele literalmente constrói um mapa da casa e refina/atualiza esse mapa a cada nova limpeza. Se ele tenta um determinado percurso e encontra um obstáculo, ele pune essa ação considerada negativa não fazendo mais esse percurso; se ele tenta um determinado percurso e não encontra um obstáculo, ele reforça essa ação considerada positiva fazendo novamente esse percurso da próxima vez. É claro que no começo ele erra bastante, mas depois ele acerta...

Eu já tive um robozinho desses e era uma mão na roda, mas o meu era bem antigo e não fazia mapeamento. Logo, ele batia na primeira vez, não aprendia nada e batia novamente em todas as outras. Vendi ele porque não estava sendo mais útil e porque meu cachorro (Chico) ficava o caçando incessantemente, mas os mais novos são extremamente modernos. Recomendo bastante – e aceito de presente ;)

Em suma: algoritmos de aprendizado por reforço baseiam-se em reforço positivo/negativo para otimização de resultado. No caso dos robôs aspiradores, eles punem a passagem por trechos pouco promissores e recompensam a passagem por trechos mais promissores. É uma técnica bem menos empregada e que eu particularmente nunca vi cair em prova, mas não custa nada mencionar em aula porque sempre pode cair.

Pré-Processamento de Dados

PRÉ-PROCESSAMENTO DE DADOS

Trata-se do processo de preparação de dados para análise e modelagem adicionais. Envolve a transformação de dados brutos em um formato mais adequado para algoritmos de aprendizado de máquina por meio de tarefas como limpeza, normalização e organização dos dados para que possam ser analisados mais facilmente e usados para fazer previsões. É também uma etapa essencial em qualquer projeto de aprendizado de máquina, pois garante que os dados estejam em um formato adequado para o processo de modelagem.

Ao construir uma casa nova, antes de pensar em aspectos estéticos ou nos móveis planejados, é necessário construir uma base sólida sobre a qual serão construídas as paredes. Além disso, quanto mais difícil for o terreno em que você construirá a casa, mais tempo e esforço podem ser necessários. Se você deixar de criar uma base robusta, nada construído sobre ela resistirá ao tempo e à natureza por muito tempo.

O mesmo problema ocorre no aprendizado de máquina. Não importa o nível de sofisticação do algoritmo de aprendizado, se você não preparar bem sua base – ou seja, seus dados – seu algoritmo não durará muito quando testado em situações reais de dados. Mesmo os modelos mais sofisticados e avançados atingem um limite e têm desempenho inferior quando os dados não estão devidamente preparados.

Tem um ditado no meio de tecnologia da informação que diz: "*Garbage In, Garbage Out*". Em outras palavras, se entrar dados ruins para serem processados por algoritmos de aprendizado de máquina, dados ruins sairão das predições dos algoritmos. Não há mágica! *E o que seria um dado ruim, Diego?* São dados ausentes, anômalos, redundantes, não padronizados, entre outros. O tratamento dos dados brutos é útil para facilitar a compreensão, visualização e identificação de padrões.

O tempo gasto no tratamento de dados pode levar cerca de 80% do tempo total que você dedica a um projeto de aprendizado de máquina. Vamos ver os principais problemas e soluções:

Valores Ausentes

Um valor ausente costuma ser representado por um código de ausência, que pode ser um valor específico, um espaço em branco ou um símbolo (por exemplo, "?"). Um valor ausente caracteriza um valor ignorado ou que não foi observado, e, nesse sentido, a substituição de valores ausentes, também conhecida como imputação, tem como objetivo estimar os valores ausentes com base nas informações disponíveis no conjunto de dados.

A imputação de valores ausentes assume que essa ausência de valor implica a perda de informação relevante de algum atributo. Consequentemente, o valor a ser imputado não deve somar nem subtrair informação à base, ou seja, ele não deve enviesar a base. Associado a isso está o fato de

que muitos algoritmos de mineração não conseguem trabalhar com os dados na ausência de valores e, portanto, a imputação é necessária para a análise.

Além disso, o tratamento incorreto ou a eliminação de objetos com valores ausentes pode promover erros das ferramentas de análise. Para resolver o problema de valores ausentes, temos algumas opções: ignorar os objetos que possuem um ou mais valores ausentes; imputar manualmente os valores ausentes; usar uma constante global para imputar o valor ausente; utilizar a média ou moda de um atributo para imputar o valor ausente; entre outros.

Dados Inconsistentes

No contexto de aprendizado de máquina, a consistência de um dado está relacionada à sua discrepância em relação a outros dados ou a um atributo, e tal consistência influencia na validade, na utilidade e na integridade da aplicação de mineração de dados. Um problema fundamental é que diferentes atributos podem ser representados pelo mesmo nome em diferentes sistemas, e o mesmo atributo pode ter diferentes nomes em diferentes sistemas.

Sempre gosto de lembrar do exemplo de bases de dados que utilizam nomes "Caixa Econômica Federal", "Caixa Econômica", "Caixa", "CEF" para representar o mesmo atributo. Outros problemas típicos de inconsistência ocorrem quando o valor apresentado na tabela não corresponde aos valores possíveis de um atributo ou o valor está fora do domínio correspondente; problemas de capitalização de textos, caracteres especiais, padronização de formatos (Ex: Data, CPF), etc.

Uma das formas de resolver esse problema é por meio de uma análise manual auxiliada por rotinas de verificação que percorrem a base de dados em busca de inconsistências.

Redução de Dimensionalidade

É intuitivo pensar que, quanto maior a quantidade de objetos e atributos, mais informações estão disponíveis para o algoritmo de mineração de dados. Entretanto, o aumento do número de objetos e da dimensão do espaço (número de atributos/variáveis na base) pode fazer com que os dados disponíveis se tornem esparsos e as medidas matemáticas usadas na análise tornem-se numericamente instáveis.

Além disso, uma quantidade muito grande de objetos e atributos pode tornar o processamento dos algoritmos de aprendizado de máquina muito complexo, assim como os modelos gerados. O ideal é utilizar técnicas de redução de dimensionalidade para reduzir a quantidade de atributos que descrevem os objetos. Não vamos nos aprofundar nesse tema agora porque teremos um tópico dedicado a esse tema mais à frente.

Normalização Numérica

A normalização é uma técnica geralmente aplicada como parte da preparação de dados para o aprendizado de máquina. O objetivo da normalização é mudar os valores das colunas numéricas no conjunto de dados para usar uma escala comum, sem distorcer as diferenças nos intervalos de valores nem perder informações. A normalização também é necessária para alguns algoritmos para modelar os dados corretamente.

Por exemplo: vamos supor que o conjunto de dados de entrada contenha uma coluna com valores variando de 0 a 1 e outra coluna com valores variando de 10.000 a 100.000. A grande diferença na escala dos números pode causar problemas ao tentar combinar os valores durante a modelagem. A normalização criando novos valores que mantêm a distribuição geral e as proporções nos dados de origem, mantendo os valores em uma escala aplicada em todas as colunas numéricas do modelo.

E como resolve isso? Você pode mudar todos os valores para uma escala de 0 a 1. Por exemplo: se você tiver um conjunto de dados de idades, poderá normalizá-lo para que todas as idades fiquem entre 0 e 1, subtraindo a idade mínima de cada valor e dividindo pela diferença entre as idades máxima e mínima. Vamos considerar o seguinte conjunto de dados de idade: [12, 17, 21, 29, 54, 89]. Agora utilizamos a seguinte fórmula:

$$x_i' = \frac{x_i - x_{min}}{x_{max} - x_{min}}$$

,em que x é a idade e i ∈ {1, ..., n}. Logo, vamos normalizar os valores do nosso conjunto original de dados para a escala entre 0 e 1:

$$\text{IDADE}_1\text{'} = \frac{IDADE1 - IDADEmin}{IDADEmáx - IDADEmin} = \frac{12-12}{89-12} = \frac{12-12}{89-12} = \frac{0}{77} = 0{,}000$$

$$\text{IDADE}_2\text{'} = \frac{IDADE2 - IDADEmin}{IDADEmáx - IDADEmin} = \frac{17-12}{89-12} = \frac{17-12}{89-12} = \frac{5}{77} = 0{,}064$$

$$\text{IDADE}_3\text{'} = \frac{IDADE3 - IDADEmin}{IDADEmáx - IDADEmin} = \frac{21-12}{89-12} = \frac{21-12}{89-12} = \frac{9}{77} = 0{,}116$$

$$\text{IDADE}_4\text{'} = \frac{IDADE4 - IDADEmin}{IDADEmáx - IDADEmin} = \frac{29-12}{89-12} = \frac{29-12}{89-12} = \frac{17}{77} = 0{,}220$$

$$\text{IDADE}_5\text{'} = \frac{IDADE5 - IDADEmin}{IDADEmáx - IDADEmin} = \frac{54-12}{89-12} = \frac{54-12}{89-12} = \frac{42}{77} = 0{,}545$$

$$\text{IDADE}_6\text{'} = \frac{IDADE6 - IDADEmin}{IDADEmáx - IDADEmin} = \frac{89-12}{89-12} = \frac{89-12}{89-12} = \frac{77}{77} = 1{,}000$$

Pronto! Agora o conjunto de dados original foi normalizado em um novo conjunto de dados contemplando valores que variam entre 0 e 1: [0, 0,064, 0,116, 0,220, 0,545, 1]. Esse tipo de normalização se chama Normalização Max-Min, mas existem outros tipos. Por fim não confundam

a normalização numérica de dados com a normalização de bases de dados relacionais, que é um conceito totalmente diferente visto dentro do contexto de bancos de dados.

Discretização

Alguns algoritmos de mineração operam apenas com atributos categóricos e, portanto, não podem ser aplicados a dados numéricos. Nesses casos, atributos numéricos podem ser discretizados, dividindo o domínio do atributo em intervalos e ampliando a quantidade de métodos de análise disponíveis para aplicação. Além disso, a discretização reduz a quantidade de valores de um dado atributo contínuo, facilitando, em muitos casos, o processo de mineração.

A maneira mais óbvia de discretizar um certo atributo é dividindo seu domínio em um número predeterminado de intervalos iguais, o que normalmente é feito no momento da coleta dos dados. *Vamos ver um exemplo?* Imagine que tenhamos um conjunto de dados numéricos que representam o peso de um grupo de pessoas. Nesse caso, podemos dividi-los em três faixas: 50 a 75 kg, 76 a 100 kg e 101 a 150 kgs.

Anomalias (Outliers)

Também chamado de valores ruidosos, referem-se a modificações dos valores originais e que, portanto, consistem em erros de medidas ou em valores consideravelmente diferentes da maioria dos outros valores do conjunto de dados. Casos típicos percebidos quando se conhece o domínio esperado para os valores em um atributo ou a distribuição esperada para os valores de um atributo, mas, no conjunto de dados, aparecem alguns valores fora desse domínio ou dessa distribuição.

Alguns exemplos são: valores que naturalmente deveriam ser positivos, mas, no conjunto de dados, aparecem como negativos, como seria o caso de observar um valor negativo para a quantidade de vendas de um restaurante em um período de um mês; ou ainda, observar que o valor de vendas desse mesmo restaurante ultrapassa, com uma grande margem, o valor total de vendas de todos os anos anteriores.

No primeiro caso, diz-se que o valor está errado por não fazer sentido no contexto dos dados; no segundo caso, tem-se um exemplo de *outlier*, que pode representar um valor errado ou uma mudança não esperada, porém real, do comportamento dos valores de um atributo, e, nesse caso, há a necessidade de identificar qual é a explicação que se adéqua à situação. Para resolver esse problema, pode-se fazer inspeções/correções manuais ou identificação/limpeza automáticas.

Dados Categóricos

Dados categóricos são comuns em problemas aprendizado de máquina e, na maioria das vezes, podem ser mais desafiadores de extrair informações do que dados numéricos. Esses dados deverão ser transformados em dados numéricos para que possam ser incluídos na etapa de treinamento.

Assim, uma melhor representação dos dados categóricos afeta diretamente a performance do treinamento como também introduz melhores formas de explicar sua contribuição para a predição.

A maioria dos algoritmos de aprendizado de máquina trabalha com variáveis numéricas, mas e quando temos dados categóricos? Aí temos que fazer uma conversão! Lembrando que dados categóricos podem ser ordenados (Ex: baixo, médio, alto) ou não ordenados (Ex: vermelho, azul, verde). No primeiro caso, pode-se representar por meio de uma codificação chamada *Ordinal Encoding*, em que cada valor varia de 1 até n classes (Ex: baixo = 1, médio = 2 e alto = 3).

Professor, essa atribuição de valores não pode dar problema? Pode, sim! Nesse exemplo, meu modelo pode entender que "alto" tem três vezes o peso de "baixo" – o que pode não ser verdadeiro em meu modelo de negócio. O nome desse problema é ponderação arbitrária, dado que estamos dando pesos arbitrários às classes. No caso de dados categóricos não ordenados, é mais complicado ainda e nem a solução anterior funciona.

Para resolver esse tipo de problema, o ideal é utilizar uma codificação chamada *One-Hot Enconding* ou *Dummy Encoding*. Vamos ver um exemplo rapidamente: imagine uma classificação de dados nas categorias Vermelho, Verde e Azul. Note que não existe uma ordem natural dessas categorias e atribuir valores aleatórios poderia recair no problema da ponderação arbitrária. Podemos utilizar, portanto, variáveis *dummy*. *O que seria isso, Diego?*

Variáveis *dummy* são variáveis que assumem os valores binários (0 ou 1) para representar a ausência ou presença de um determinado atributo, característica, categoria, evento ou critério.

Agora vamos criar uma tabelinha com uma coluna para cada categoria (vermelho, verde e azul). Em seguida, para cada observação do nosso conjunto de dados, vamos atribuir o valor 1 (um) – quando correspondente à categoria – e 0 (zero) – quando não correspondente à categoria. Logo, a primeira observação é "vermelho", logo inserimos 1 (um) na coluna "vermelho" e 0 (zero) nas outras colunas; e fazemos assim para cada uma das observações.

OBVERVAÇÕES		VERMELHO	VERDE	AZUL
VERMELHO		1	0	0
VERDE		0	1	0
AZUL		0	0	1
VERDE	ONE-HOT ENCODING	0	1	0
AZUL		0	0	1
VERMELHO		1	0	0
VERMELHO		1	0	0

Note que essa operação é bidirecional: é possível sair do valor original para o *one-hot enconding* quanto do *one-hot encoding* para o valor original. Dessa forma, não temos uma perda nem um acréscimo de informação! Existem variações dessa técnica para a codificação de dados categóricos. A vantagem dessas técnicas é permitir que o algoritmo de aprendizado de máquina entenda melhor as relações entre os dados e faça previsões mais precisas sem perda/acréscimo de dados.

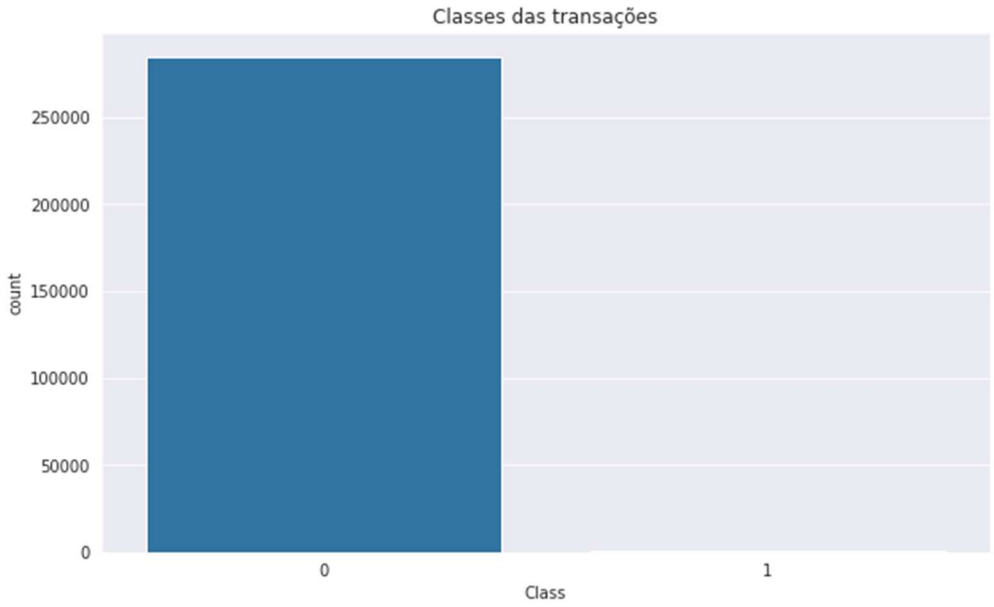
Para finalizar, note que o *one-hot encoding* tem uma redundância: se eu tenho *n* categorias, eu não preciso de *n* colunas na tabela para representá-las. No caso das categorias de cores, se uma observação não é vermelha nem verde, ela é necessariamente azul; se não é vermelha nem azul, ela é necessariamente verde; e se não é verde nem azul, ela é necessariamente vermelha. Logo, para representar essas categorias, bastava n-1 colunas na tabela. Vejamos:

Note que, com apenas duas colunas, foi possível representar a mesma informação da tabela anterior. É bem simples: tudo que não for vermelho ou verde, será azul!

Classes Desbalanceadas

Um problema muito comum de classificação ocorre quando existe uma desproporção nos dados, isto é, temos muito mais dados de uma classe do que de outra (pode ocorrer também para problemas com mais de duas classes). Então, quando temos um desequilíbrio de classe, o classificador de aprendizado de máquina tende a ser mais tendencioso em relação à classe majoritária, causando má classificação da classe minoritária. *Como assim, Diego?*

Imagine os dados sobre análise de fraudes em transações de cartões de crédito. *Sabe quando aparece no seu celular uma notificação sobre uma compra que você sabe que você não fez?* É aquela correria para ligar no banco e bloquear o cartão. Pois é, tem sido cada vez mais comum! Quando analisamos um conjunto de dados sobre transações de cartões de crédito, vemos basicamente duas classes: transações normais e transações fraudulentas. *E onde entra o desbalanceamento?*



Ora, eu já tive cartões bloqueados em duas oportunidades por conta de transações fraudulentas, mas é a exceção da exceção da exceção: a regra é que a imensa maioria das transações sejam normais. Vejam na imagem anterior um exemplo de um conjunto de dados de transações de uma operadora de cartão de crédito: a Classe o representa as transações normais e a Classe 1 representa as transações fraudulentas.

Note que nem é possível ver nada azul da Classe 1 porque é tão pouco que é quase irrelevante, mas esse gráfico representa 284.315 transações normais e 492 transações fraudulentas. Em outras palavras, temos 99,82% de transações normais e 0,17% de transações fraudulentas. *E por que isso é um problema, Diego?* Porque esse desequilíbrio pode levar a modelos tendenciosos que não são representativos da população como um todo.

Isso pode ocasionar previsões imprecisas e baixo desempenho do modelo. Além disso, pode levar a classificadores excessivamente sensíveis à classe majoritária, ignorando a classe minoritária. Galera, imagine que eu faça a pior modelagem possível de um modelo de classificação! Por pior que ele seja, se ele simplesmente "chutar" sempre que uma transação foi normal, ele acertará na maioria das vezes porque raramente uma transação é fraudulenta.

Nesse caso, a acurácia do nosso modelo será próxima de 100%, mesmo ele tendo sido pessimamente modelado. A máquina não aprendeu nada – ela apenas foi guiada a minimizar o erro no conjunto de dados de treinamento (que tem ampla maioria de transações normais) e ignorou os padrões de transações fraudulentas. O nome disso é Paradoxo da Acurácia, em que parâmetros de um algoritmo não diferenciam a classe minoritária das demais categorias.

Ora, mas o principal objetivo de um modelo de aprendizado de máquina que trata de transações de cartão de crédito é justamente identificar padrões de transações fraudulentas a fim de impedi-las. Bacana! *E como resolve isso?* Bem, uma alternativa é conseguir mais dados de treinamento –

preferencialmente da classe minoritária. Outra alternativa é alterar a métrica de desempenho: em vez de usar a acurácia.

A acurácia é uma métrica muito sensível ao acerto médio, logo não tem um bom desempenho na medição da qualidade de modelos quando o conjunto de dados tem uma desproporção muito grande entre as classes. Para resolver esse problema, podemos utilizar outras métricas de desempenho, tais como F-Score ou Curva ROC. Outra alternativa é fazer uma reamostragem e, para isso, temos duas opções...

Podemos fazer um *undersampling* majoritário (ou subamostragem), que consiste em eliminar aleatoriamente dados da classe majoritária até que ambas as classes tenham a mesma quantidade de dados. Inversamente, podemos fazer um *oversampling* minoritário (ou superamostragem), que consiste em criar novos dados baseados nos dados da classe minoritária de forma aleatória até que ambas as classes tenham a mesma quantidade de dados.

Trata-se de uma forma de garantir que os dados da classe minoritária apareçam várias vezes – é como se eu replicasse aleatoriamente dados da classe minoritária para eliminar a desproporção. É claro que essas soluções também possuem problemas: a subamostragem perde informações úteis, o que pode fazer o modelo gerar *overfitting*; já a superamostragem pode induzir o modelo a encontrar padrões de dados que não refletem a realidade da classe minoritária.

Existem outras técnicas, tais como: utilização de amostras sintéticas, atribuição de pesos diferentes às classes, utilização de algoritmos específicos para detecção de anomalias, avaliação de algoritmos menos sensíveis ao desbalanceamento (como as árvores de decisão). Enfim... existem diversas soluções e essas soluções podem ser combinadas de diversas formas para minimizar o problema do desbalanceamento (também chamado de amostras enviesadas).

Esse foi basicamente o tema de uma das questões da prova discursiva do TCU! Ele mostrava um caso em que um classificador de peças boas ou defeituosas atingiu altíssima precisão. Ocorre que o modelo basicamente que todas as peças eram boas e nenhuma peça era defeituosa – caso típico de desbalanceamento de classes. Como o modelo tinha o objetivo principal justamente de indicar peças defeituosas, ele fez um péssimo serviço!

Pedia-se também para dar exemplos de soluções, tais como *oversampling*, *undersampling*, atribuição de pesos diferentes às classes, entre outras – tudo que acabamos de estudar!

Técnicas e Tarefas

Antes de adentrarmos nesse tópico, é importante entender que existe um sombreamento entre aprendizado de máquina e mineração de dados. Logo, algumas tarefas de mineração de dados são também tarefas de aprendizado de máquina. No entanto, aqui nós vamos detalhar um pouco mais dentro do contexto de aprendizado de máquina. Vamos começar falando sobre a técnica de classificação de dados.

Classificação

RELEVÂNCIA EM PROVA: MÉDIA

CLASSIFICAÇÃO

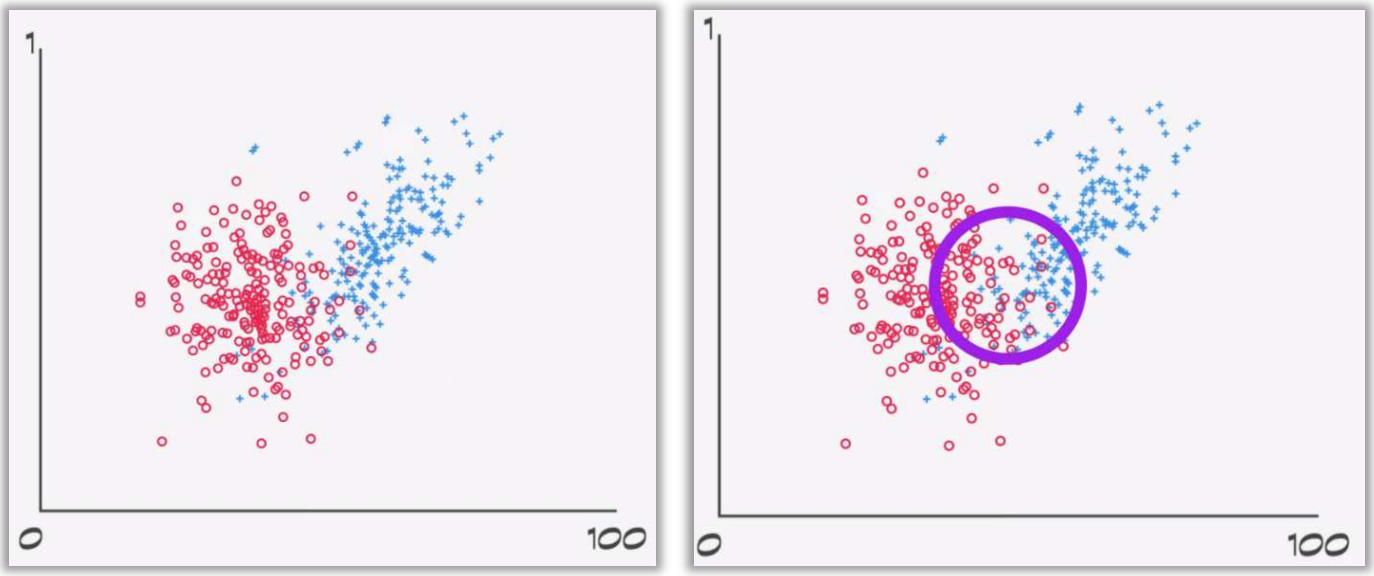
Trata-se de uma técnica de aprendizado de máquina que permite aos usuários classificar os dados em grupos para facilitar a análise. É usado para prever o comportamento futuro de um dado ou para descobrir padrões em um conjunto de dados.

Nós já sabemos que se trata de uma técnica de aprendizado supervisionado para distribuir um conjunto de dados de entrada em categorias ou classes pré-definidas de saída. Por exemplo: nós poderíamos utilizar um algoritmo de classificação binária para decidir de uma mariposa é da espécie Imperatriz (imagem da esquerda) ou Luna (imagem da direita). Note que eu já disse de antemão quais são os rótulos ou categorias – dado que é um algoritmo de aprendizado supervisionado.

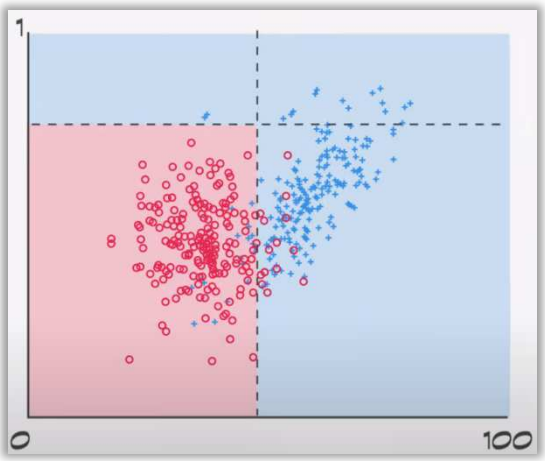
E como um algoritmo poderia decidir algo dessa natureza? Para tal, o supervisor pode escolher um conjunto de *features* (também chamados de características ou qualidades). Essas *features* são basicamente valores que caracterizam de forma útil as coisas que desejamos classificar. Para o nosso exemplo, vamos utilizar duas *features*: envergadura e massa. Para treinar nosso algoritmo de classificação para fazer boas previsões, vamos precisar de dados de treinamento.

E como conseguimos? Nós podemos enviar um entomologista a uma floresta para capturar diversas mariposas de ambas as espécies, examiná-las e registrar os valores das duas *features* que nós escolhemos em uma tabela. A tabela final resultante contendo todas as mariposas catalogadas com seu rótulo de espécie, envergadura e massa é também chamado de dados rotulados. Vejamos como seria uma possível tabela de dados rotulados...

Como temos apenas duas *features*, é fácil visualizar esses dados em um gráfico de dispersão. Na imagem da esquerda, eu plotei os dados de 100 mariposas imperatriz em vermelho e 100 mariposas luna em azul. No eixo horizontal, temos a envergadura em milímetros; no eixo vertical, temos a massa em gramas. Observe na imagem da direita que é possível ver dois agrupamentos, mas no meio (círculo roxo) existe uma sobreposição.



Essa sobreposição indica que não é tão óbvio separar dois grupos. É aqui que entram os algoritmos de aprendizado de máquina: eles são capazes de encontrar uma divisão ideal entre os dois grupos!



Vamos explicar isso melhor: imagine que eu trace uma linha reta vertical na altura dos 45 milímetros de envergadura e afirme que tudo que estiver à esquerda provavelmente é uma mariposa imperatriz e tudo à direita uma mariposa luna. Além disso, eu posso traçar uma linha horizontal na altura dos 0,75 gramas de massa e afirmar que tudo que estiver abaixo desse valor provavelmente é uma mariposa imperatriz. Com isso, vejam na imagem ao lado que se forma um quadrante em que sua parte inferior esquerda representa prováveis mariposas imperatriz e o restante representa as mariposas luna.

Essas linhas que dividem o espaço de decisão são chamadas de limites de decisão porque auxiliam a indicar qual será o classificador sugerido. Agora notem que interessante: o entomologista capturou 200 mariposas e criou a tabela de dados rotulados. Logo, nós podemos comparar os dados rotulados com os dados resultantes do gráfico de dispersão para verificar se as linhas sugeridas fizeram uma divisão satisfatória ou não para identificar as espécies de mariposa.

Esse é uma das formas de avaliar se o processo de classificação foi satisfatório ou não! Se olharmos atentamente, podemos verificar que 86 mariposas imperatriz (vermelho) terminaram de forma correta dentro da região de decisão (em vermelho), mas 14 delas acabaram de forma incorreta no território da mariposa luna (em azul). Por outro lado, 82 mariposas luna (azul) foram classificadas corretamente (em azul), com 18 caindo para o lado errado (em vermelho).

Nós podemos representar esses valores por meio de uma matriz de confusão! *É o que, Diego?* A matriz de confusão é uma tabela utilizada para avaliar a qualidade de um modelo que mostra as frequências de classificação para cada classificador/rótulo do modelo. Trata-se geralmente de uma tabela com duas linhas e duas colunas que exibe a quantidade de erros e acertos de classificação de uma amostra de dados. *Professor, quem está confuso sou eu agora...*

Vejam só: as linhas que nós traçamos no gráfico de dispersão acabou ocasionando alguns erros. Das 100 mariposas imperatriz, nós acertamos 86 e erramos 14; das 100 mariposas luna, nós acertamos 82 e erramos 18. Nós podemos colocar esses dados em uma matriz para visualizar mais facilmente seus acertos e seus erros a fim de avaliar com maior clareza o desempenho do meu modelo de classificação (aliás, ela também costuma ser chamada de matriz de erro).

Na página seguinte, temos a nossa matriz de confusão! Note que ela possui dois eixos: horizontal, que indica o valor previsto ou esperado; e vertical, que indica o valor real. *Como posso interpretar essa matriz?* Vejamos: nós tivemos 86 mariposas das 200 coletadas pelo entomólogo em que seu rótulo real na tabela de dados rotulados era mariposa imperatriz e o meu modelo de classificação classificou corretamente como mariposa imperatriz.

Da mesma forma, nós tivemos 14 mariposas das 200 coletadas pelo entomólogo em que seu rótulo real na tabela de dados rotulados era mariposa imperatriz, mas o meu modelo de classificação

classificou incorretamente como mariposa luna. Tivemos 18 mariposas em que seu rótulo real era mariposa luna, mas meu modelo classificou como mariposa imperatriz; e tivermos 82 mariposas luna corretamente classificadas pelo modelo como mariposa luna.

Observe que não há como desenharmos linhas que nos forneçam 100% de acurácia! Se reduzirmos o valor limite de decisão da envergadura da mariposa (eixo horizontal), classificaremos erroneamente mais mariposas imperatriz como mariposas luna; por outro lado, se o aumentarmos, classificaremos incorretamente mais mariposas luna. E o mesmo tipo de problema ocorre se alterarmos os limites de massa (eixo vertical).

O trabalho dos algoritmos de aprendizado de máquina é tentar maximizar as classificações corretas enquanto minimiza seus erros. *E como podemos medir efetivamente como foi o desempenho do nosso modelo?* Nós podemos fazê-lo utilizando a métrica de acurácia, isto é, a divisão do número de acertos pelo total de predições. Em nosso caso, tivemos 82+86 = 168 acertos em uma amostra de 200 mariposas. Dessa forma, a nossa acurácia foi de 168/200 = 84%.

Outras configurações de linhas poderiam ser testadas para tentar aumentar a acurácia e melhorar o nosso modelo de classificação. Nós podemos generalizar a matriz de confusão da seguinte forma:

Um bom exemplo para os dias atuais são os exames de coronavírus. Podemos afirmar que um exame pode resultar em positivo ou negativo, logo se trata de uma classificação binária porque temos apenas dois rótulos (Classe 1 = Positivo e Classe 2 = Negativo). Logo, os resultados possíveis são Verdadeiro-Positivo (VP), Falso-Positivo (FP), Verdadeiro-Negativo (VN) ou Falso-Negativo (FN). *Professor, por que você repetiu duas vezes a imagem abaixo?*

Observe que há uma pequena diferença nos quadrantes vermelhos: a nomenclatura utilizada pela área de estatística para Falso-Positivo é Erro Tipo I e de Falso-Negativo é Erro Tipo II. Apenas isso! Pessoal, uma maneira de medir o desempenho de um processo de classificação é por meio da acurácia, mas existem outras formas de medir. Nós vamos falar rapidamente sobre a Acurácia, Sensibilidade, Especificidade, Precisão e F1-Score...

MEDIÇÃO	DESCRIÇÃO	FÓRMULA
ACURÁCIA	Trata-se da métrica mais simples que permite mensurar o percentual de acertos, isto é, a quantidade de previsões corretas dentro do total de previsões possíveis. Responde à pergunta: dentre todas as previsões realizadas, quantas o modelo acertou?	$\frac{VP + VN}{VP + FP + VN + FN}$
SENSIBILIDADE	Trata-se da métrica que permite avaliar a capacidade do classificador de detectar com sucesso resultados positivos (também chamado de revocação ou recall). Responde à pergunta: *dentre os valores realmente positivos, quantos o modelo acertou (previu corretamente como positivo)?*	$\frac{VP}{VP + FN}$
ESPECIFICIDADE	Trata-se da métrica que permite avaliar a capacidade do classificador de detectar com sucesso resultados negativos. Responde à pergunta: dentre os valores realmente negativos, quantos o modelo acertou (previu corretamente como negativo)?	$\frac{FP}{FP + VN}$
PRECISÃO	Trata-se da métrica que permite mensurar a proporção de previsões positivas corretas sobre a soma de todos os valores positivos. Responde à pergunta: dentre os valores previstos como positivos, quantos o modelo acertou (previu corretamente como positivo)?	$\frac{VP}{VP + FP}$

F1-SCORE	Trata-se da média harmônica calculada com base na precisão e na sensibilidade, logo é uma medida derivada dessas outras medidas. Essa medida tenta condensar em uma única medida um pouco da precisão e um pouco da sensibilidade.	$2 * \frac{\text{PRECISÃO} * \text{RECALL}}{\text{PRECISÃO} + \text{RECALL}}$

Professor, qual é a necessidade desse tanto de medidas diferentes? Dependendo do contexto, o desempenho pode ser medido de maneira diferente para refletir melhor a efetividade da medição. Em alguns casos, ter falsos-negativos não é tão relevante; em outros casos, ter falsos-positivos pode ser muito relevante. Por isso existem tantas formas diferentes de medir o desempenho. Vamos ver isso melhor...

A precisão pode ser utilizada em situações em que falsos-positivos são mais prejudiciais que os falsos-negativos. Por exemplo: ao classificar ações da bolsa de valores como boas ou ruins, um falso-positivo pode fazer uma pessoa investir em uma ação ruim e ter prejuízos; já um falso-negativo pode fazer uma pessoa não investir em uma ação boa e deixar de ter lucros, mas ela não terá prejuízos, logo é menos prejudicial.

Já o recall pode ser utilizado em situações em que falsos-negativos são mais prejudiciais que os falsos-positivos. Por exemplo: ao classificar uma pessoa com vacinado ou não-vacinado, um falso-positivo pode fazer uma pessoa saudável não pegar um avião com outras pessoas; já um falso-negativo pode fazer uma pessoa infectada pegar um avião com outras pessoas e infectá-las com seu vírus. *Entenderam a importância de medidas diferentes?*

Agrupamento

RELEVÂNCIA EM PROVA: MÉDIA

AGRUPAMENTO

Trata-se de uma técnica de aprendizado de máquina utilizada para identificar grupos em dados. É usado para descobrir padrões e relações entre variáveis. Por exemplo, pode ser usado para agrupar pessoas com base em características como idade, gênero ou localização. O agrupamento pode ser usado para encontrar grupos com características semelhantes, classificar objetos e prever comportamentos futuros.

Vamos falar agora sobre o agrupamento (*clustering*). De acordo com Leandro Castro e Daniel Ferrari em Introdução à Mineração de Dados, uma das habilidades mais básicas dos organismos vivos é a capacidade de agrupar objetos similares para produzir uma taxonomia, isto é, organizar coisas similares em categorias (também chamadas de grupos ou clusters). A análise de agrupamento busca analisar dados multivariados a fim de descobrir grupos homogêneos de objetos.

A análise de grupos pode ser definida como a organização de um conjunto de objetos (normalmente representados por vetores de características, ou seja, pontos em um espaço multidimensional) em grupos baseada na similaridade entre eles. Dito de outra forma, agrupar objetos é o processo de

particionar um conjunto de dados em subconjuntos (grupos) de forma que os objetos em cada grupo compartilhem características comuns.

Idealmente, objetos pertencentes ao mesmo grupo são mais similares entre si do que a objetos pertencentes a grupos distintos. Dessa forma, um grupo pode ser definido em função da coesão interna (homogeneidade) e do isolamento externo (separação) de seus objetos. *Diego, qual é a diferença do agrupamento para a classificação?* Esse é um ponto chave que as bancas adoram cobrar em prova: a classificação é um algoritmo supervisionado e o agrupamento é não supervisionado.

Isso significa que a base de dados de entrada é previamente rotulada, isto é, cada objeto da base possui a classe correspondente à qual pertence definida *a priori*. O algoritmo, então, busca identificar a classe à qual pertence um novo objeto ainda não apresentado e com rótulo de classe desconhecido. Em regra, objetos rotulados são apresentados ao algoritmo de classificação para que ele seja treinado, isto é, para que um novo modelo seja criado a fim de classificar novos objetos.

No agrupamento, os dados não são previamente rotulados. Aliás, os rótulos dos objetos são obtidos apenas a partir do algoritmo de agrupamento e não são utilizados durante o processo de treinamento do algoritmo. Como se trata de um algoritmo de aprendizado não supervisionado, não existe um supervisor que guie o aprendizado a fim de minimizar os erros de previsão. A ideia aqui é que observações que tenham valores de variáveis mais parecidos estejam dentro do mesmo grupo.

Não só isso: observações que estejam em grupos diferentes devem ter os valores de variáveis mais diferentes possíveis. Em suma: o agrupamento (ou *clustering*) é um algoritmo não supervisionado – aquele que não possui um rótulo/target que possa orientar o treinamento de um modelo preditivo – em que cada observação é atribuída a um grupo formado por outras observações mais similares entre si e mais diferentes das observações dos demais grupos.

Por que o clustering é importante? Porque ele é muito útil em diversas áreas: no comércio, ele pode ser utilizado para determinar grupos de clientes que têm padrões de compra semelhantes; na medicina, pode ser importante para determinar grupos de pacientes que mostram reações semelhantes aos medicamentos receitados. Enfim, há diversas e diversas possibilidades de utilização dessa técnica. Agora partimos para as regras de associação...

Regras de Associação

RELEVÂNCIA EM PROVA: MÉDIA

REGRAS DE ASSOCIAÇÃO

Trata-se de uma técnica de aprendizado de máquina utilizada para descobrir relações entre variáveis em conjunto de dados. Essas regras descrevem padrões que ocorrem com frequência e podem ser usadas para prever comportamentos futuros. Geralmente, elas são usadas para identificar padrões de compra, comportamento de consumidor, padrões de fraudes, etc.

Regras de Associação são algoritmos não supervisionados utilizados para descobrir relações interessantes entre variáveis de um grande conjunto de dados. Essas regras nos permitem identificar frequentemente padrões recorrentes em dados, que podem ser utilizados para realizar previsões sobre eventos futuros e tomar melhores decisões de negócio. O exemplo clássico de regras de associação é o do carrinho de supermercado.

Toda vez que você faz compras, cada item das suas compras é armazenado no banco de dados do supermercado – ele guarda quais produtos você comprou, valores, quantidades, etc. Os itens são sendo armazenados por meio de transações do banco de dados, logo são utilizados bancos de dados transacionais, ou seja, aqueles especializados em armazenar informações sobre transações (inserções, leituras, atualizações e exclusões) efetuadas pelo usuário.

Nesse contexto, nós temos a Análise de Cesta de Mercado! A cesta de mercado corresponde aos conjuntos de itens que um consumidor compra em um supermercado durante uma visita. Exemplo:

TRANSAÇÃO	HORA	ITENS
1	20:45	Leite, Pão, Biscoito, Suco
2	12:09	Leite, Pão
3	08:58	Leite, Ovos, Pão
4	15:15	Pão, Biscoito, Café

FORMATO DE REGRAS DE ASSOCIAÇÃO: {ANTECEDENTE → CONSEQUENTE}

A regra de associação segue o formato X → Y, em que X = {x_1, x_2, ..., x_n} e Y = {y_1, y_2, ..., y_m} são conjuntos de itens, com x_i e y_i sendo itens distintos para todo *i* e todo *j*. Essa associação indica que, se um cliente compra X, ele provavelmente também comprará Y. Na tabela acima, é possível notar que há uma possível associação entre leite e pão. E isso reflete a realidade: quando você vai ao mercado e compra cerveja, gelo e carne, é provável que você também comprará carvão.

EXEMPLO DE REGRA DE ASSOCIAÇÃO: {CERVEJA, GELO, CARNE → CARVÃO}

E isso funciona em diversos contextos: quando você compra uma raquete de tênis, já já aparecem banners de bolinhas de tênis; quando você assiste um filme de terror na Netflix e dá uma boa nota, da próxima vez ela já te recomenda diversos filmes de terror que pessoas que assistiram o mesmo que você também gostaram; quando você procura sobre o edital de um concurso que acabou de sair, em dois tempos aparece o banner do Estratégia Concursos (aliás, obrigado pela confiança).

Isso ocorre porque bases de dados gigantescas possuem uma infinidade de dados que podem ser minerados em busca de associações interessantes. *Vocês entendem agora o porquê do massivo investimento em inteligência artificial, aprendizado de máquina, mineração de dados, entre outros?* Se uma loja consegue analisar os produtos que você comprou recentemente e, baseado nisso, consegue fazer uma boa previsão de outro produto para você comprar, ela tem ouro nas mãos!

O Diego comprou um curso para estudar para um concurso, vou oferecer para ele uma mentoria. O Diego comprou um celular novo, vou oferecer para ele uma capinha bacana. *Sacaram a ideia?* Diferentemente do agrupamento, que busca relações de similaridade entre objetos, as regras de associação buscam relações entre os atributos dos objetos, ou seja, os itens que compõem a base. O objetivo é encontrar regras fortes de acordo com alguma medida do grau de interesse da regra.

Como uma grande quantidade de regras de associação pode ser derivada a partir de uma base de dados, mesmo que pequena, normalmente se objetiva a derivação de regras que suportem um grande número de transações e que possuam uma confiança razoável para as transações às quais elas são aplicáveis. Esses requisitos estão associados a dois conceitos centrais em mineração de regras de associação:

MEDIDAS DE INTERESSE	DESCRIÇÃO
SUPORTE/ PREVALÊNCIA	Trata-se da frequência com que um conjunto de itens específicos ocorrem no banco de dados, isto é, o percentual de transações que contém todos os itens do conjunto. Em termos matemáticos, a medida de suporte para uma regra X → Y é a frequência em que o conjunto de itens aparece nas transações do banco de dados. Um suporte alto nos leva a crer que os itens do conjunto X e Y costumam ser comprados juntos, pois ocorrem com alta frequência no banco
CONFIANÇA/ FORÇA	Trata-se da probabilidade de que exista uma relação entre itens. Em termos matemáticos, a medida de confiança para uma regra X → Y é a força com que essa regra funciona. Ela é calculada pela frequência dos itens Y serem comprados dado que os itens X foram comprados.

	Uma confiança alta nos leva a crer que exista uma alta probabilidade de que se X for comprado, Y também será.

Na tabela a seguir, temos um exemplo de dez transações contidas no banco de dados de um determinado supermercado. Note que a Compra 1 continha cerveja, gelo, carne, carvão e outros; a Compra 2 continha apenas cerveja, carne e carvão; a Compra 4 continha apenas outros; e assim por diante. A medida de suporte da regra de associação é calculada por meio da razão da quantidade de ocorrências simultâneas dos itens da esquerda pela quantidade total de transações.

REGRA DE ASSOCIAÇÃO: {CERVEJA, GELO, CARNE → CARVÃO}

COMPRA	CERVEJA	GELO	CARNE	CARVÃO	OUTROS
1	X	X	X	X	X
2	X		X	X	
3	X	X	X	X	
4					X
5			X	X	
6	X	X	X	X	
7	X	X	X		
8		X			X
9	X				

Dada a regra de associação acima, vamos primeiro calcular a quantidade de ocorrências simultâneas dos itens da esquerda: {Cerveja, Gelo, Carne}. Eles aparecem simultaneamente quatro vezes: Compra 1, Compra 3, Compra 6 e Compra 7. Como temos 10 transações, a medida de suporte será 4/10 = 40%. Já a medida de confiança da regra de associação é calculada de uma maneira um pouquinho diferente...

Nós dividimos a quantidade de vezes que os itens da esquerda apareceram simultaneamente (nós já calculamos acima e sabemos que é 4). Dadas essas quatro transações, finalmente calculamos em quantas vezes o item da direita (Carvão) ocorreu e dividimos pelo valor anterior. Ora, vamos olhar novamente para as quatro transações: *em quantas delas Carvão também apareceu?* Em 3 transações: Compra 1, Compra 3 e Compra 6). Logo, a medida de confiança da regra de associação é ¾ = 75%.

Note que houve uma transação em que foram comprados cerveja, gelo e carne, mas não foi comprado carvão (Compra 7). *Talvez ele já tivesse o carvão em casa né?* 😊

Modelos Lineares

RELEVÂNCIA EM PROVA: MÉDIA

MODELOS LINEARES

Trata-se de um conjunto de técnicas de aprendizado de máquina usado para prever a resposta de uma variável dependente (variável que você está tentando prever) com base em um ou mais variáveis independentes (variáveis que você usa para prever a variável dependente). Estes modelos têm como base a hipótese de que a relação entre as variáveis é linear. Os modelos lineares são largamente utilizados na análise de regressão, pois permitem a previsão de resultados futuros com base em dados passados.

Uma das principais aplicações de aprendizado de máquina é a previsão, isto é, quando queremos prever algum atributo tendo somente alguns dados de entrada. Determinamos como fazer essa previsão com base em exemplos históricos de dados de entrada e saída, isto é, baseado em comportamentos observados no passado, conseguimos fazer inferências sobre o futuro. Para tal, nós utilizamos modelos preditivos. *O que seria isso, Diego?*

Um modelo preditivo é aquele responsável por relacionar dados de entrada (também chamados de variáveis independentes) com o resultado esperado (também chamados de variável dependente ou variável alvo contínua[2]). Diferentes modelos geram formas matematicamente muito diferentes de construir a relação entre as variáveis de entrada e de saída, tornando-os assim capazes de captar padrões estatísticos também diferentes.

Em regra, é preciso realizar experimentos computacionais, avaliando o desempenho de modelos de tipos diferentes para descobrir qual é o mais adequado a uma tarefa e um conjunto específico de dados. *Por que, Diego?* Porque cada tipo de modelo tem suas características, seus pontos fortes e fracos e sua lógica de funcionamento. Não é preciso reimplementar um algoritmo do zero para entender suas propriedades fundamentais e utilizá-lo adequadamente.

Em suma: modelos preditivos são basicamente uma função matemática que, quando aplicada a uma massa de dados, é capaz de identificar padrões e oferecer uma previsão do que pode ocorrer. Existem vários tipos de modelos de predição, dentre eles se destacam os modelos lineares. Esses modelos compreendem uma ampla família de modelos derivados da estatística, embora apenas dois deles (regressão linear e a regressão logística) sejam frequentemente utilizados.

Estatísticos, econometristas e cientistas de muitas disciplinas há muito usam modelos lineares para confirmar suas teorias por meio de validação de dados e para obter previsões práticas. Os modelos lineares são fáceis de entender, rápidos de criar e moleza de implementar do zero. Se você os dominar, você realmente tem o equivalente a um canivete suíço para aprendizado de máquina que não pode fazer tudo perfeitamente, mas pode atendê-lo prontamente e com excelentes resultados.

Os dois modelos lineares mais conhecidos são a regressão linear e a regressão logística. Vamos ver como funciona cada uma delas, mas antes vamos entender o que é a regressão...

---

[2] Variáveis contínuas são variáveis numéricas que têm um número infinito de valores entre dois valores quaisquer (Ex: uma régua possui valores contínuos, dado que 3 cm é diferente de 3,01 cm, que é diferente de 3,01354 cm, que é diferente de 3,01355, e assim por diante).

Regressão é uma técnica para investigar a relação entre variáveis ou *features* independentes e uma variável ou resultado dependente. É usado como um método de modelagem preditiva em aprendizado de máquina, no qual um algoritmo é usado para prever resultados contínuos. Resolver problemas de regressão é uma das aplicações mais comuns para modelos de aprendizado de máquina, especialmente em aprendizado de máquina supervisionado.

Os algoritmos são treinados para entender a relação entre variáveis independentes e um resultado ou variável dependente. O modelo pode então ser aproveitado para prever o resultado de dados de entrada novos e não vistos, ou para preencher uma lacuna nos dados ausentes. A análise de regressão é parte integrante de qualquer modelo preditivo, portanto é um método comum encontrado na análise preditiva baseada em aprendizado de máquina.

Apesar de haver diversas diferenças entre classificação e regressão, é possível converter uma na outra e vice-versa, modificando a representação da variável-alvo. Por exemplo: se eu quiser transformar uma regressão em uma classificação, eu posso definir um valor limite para determinar a qual classe uma observação pertence. *Como assim, Diego?* Imagine que eu queira classificar um conjunto de mulheres em baixas, regulares e altas.

Eu posso definir um intervalo de 1,60 cm a 175 cm de forma que aquelas que tenham altura menor ou igual a 1,60 cm serão consideradas baixas; aquelas que tenham altura entre 1,61 e 1,74 serão consideradas regulares; e aquela que tenham altura maior ou igual a 1,75 serão consideradas altas. É isso, basta dividir o conjunto de possíveis valores numéricos em intervalos de forma que cada intervalo se torne uma classe.

Analogamente, se eu quiser transformar uma classificação em uma regressão, eu posso associar um valor numérico para cada classe. É claro que as classes têm que ser ordenáveis (Ex: baixo, regular ou alta), caso contrário não é possível fazer essa associação (Ex: Masculino e Feminino). A imensa maioria dos algoritmos que veremos em aula têm uma versão tanto para classificação quanto para regressão. *Bacana?* Vamos ver agora um pouquinho sobre regressão linear...

Regressão Linear

REGRESSÃO LINEAR

Trata-se da ferramenta estatística que nos ajuda a quantificar a relação entre uma variável específica e um resultado que nos interessa enquanto controlamos outros fatores. Em outras palavras, podemos isolar o efeito de uma variável enquanto mantemos os efeitos das outras variáveis constantes.

A regressão linear é a ferramenta estatística que nos ajuda a quantificar a relação entre uma variável específica e um resultado que nos interessa enquanto controlamos outros fatores. Em outras palavras, podemos isolar o efeito de uma variável enquanto mantemos os efeitos das outras

variáveis constantes. A imensa maioria dos estudos que você lê nos jornais é baseada em análise de regressão. Vamos ver alguns exemplos...

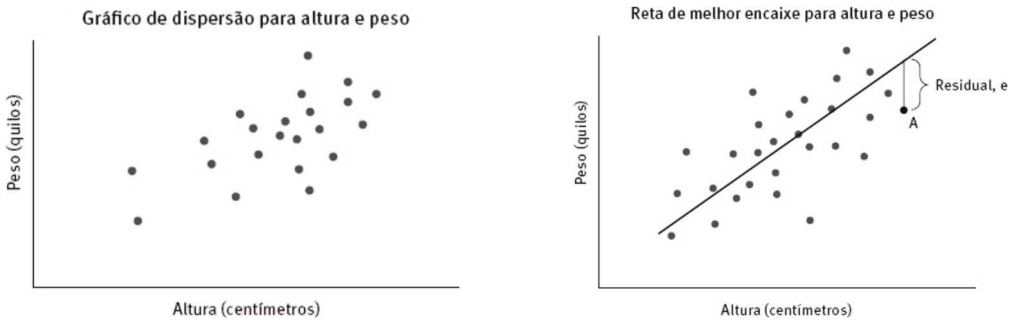
Há um estudo que afirma que, crianças que passam muito tempo em creches são mais propensas a apresentar problemas comportamentais quando vão para escola do que crianças que passam esse tempo em casa com seus pais. *Ora, por acaso os pesquisadores designaram aleatoriamente milhares de crianças pequenas a creches ou a ficarem em casa com um dos pais para fazer a pesquisa?* É evidente que não!

Diferentes famílias tomam decisões distintas em relação a como cuidar dos filhos porque são diferentes. Alguns lares têm os dois pais presentes; outros não têm. Alguns têm os dois pais trabalhando; outros não têm. Alguns lares são mais ricos e mais cultos que outros. Nós sabemos que todas essas coisas afetam a decisão de como cuidar dos filhos e afetam o desempenho futuro das crianças dessas famílias na escola.

Quando feita adequadamente, a regressão linear pode nos ajudar a estimar os efeitos das creches separadamente de outras coisas que afetam crianças pequenas como renda familiar, estrutura familiar, educação parental, e assim por diante. Hoje em dia, de posse dos dados adequados e com acesso a um computador, uma criança de seis anos pode usar um programa de estatística básica para gerar resultados de regressão. Aliás, o próprio MS-Excel possui essa funcionalidade...

O computador tornou possível realizar a parte mecânica da análise de regressão quase sem nenhum esforço. Até porque existe uma ideia básica subjacente a absolutamente todas as formas de análise de regressão – desde as relações estatísticas mais simples até os modelos supercomplexos criados por ganhadores do Prêmio Nobel de Economia! Em essência, a regressão linear busca encontrar o "melhor encaixe" para uma relação linear entre duas variáveis.

Um exemplo simples é a relação entre altura e peso. *Como assim, Diego?* Ora, pessoas mais altas tendem a pesar mais – embora obviamente esse não seja sempre o caso. Vejamos um exemplo:



Se lhe fosse pedido que descrevesse o padrão, você poderia dizer algo mais ou menos do tipo: "*O peso parece aumentar com a altura*". É um bom entendimento, mas a regressão linear nos dá a possibilidade de ir além e "encaixar uma reta" que melhor descreva uma relação linear entre as duas variáveis (Peso e Altura). *Muitas retas possíveis são amplamente consistentes com os dados de altura e peso, mas como sabemos qual é a melhor reta para esses dados?*

É aqui que entra em cena o aprendizado de máquina! A ideia do algoritmo é oferecer vários dados para que ele encontre a equação que melhor descreve e se ajusta aos dados, isto é, que minimize a variância dos erros em uma predição. A Regressão Linear utiliza tipicamente uma metodologia chamada de Mínimos Quadrados Ordinários (MQO). O ponto-chave dessa metodologia reside na parte dos "*Mínimos Quadrados*" do nome...

O MQO encaixa a reta que minimiza a soma dos residuais elevados ao quadrado. Calma, não é tão complicado quanto parece! Cada observação nos nossos dados de altura e peso tem um residual, que é a distância vertical a partir da reta de regressão, exceto para aquelas observações que se situam diretamente em cima da reta, para as quais o residual vale zero. Note que, quanto maior a soma geral dos residuais, pior é o encaixe da reta.

Por que o nome do método contém as palavras "mínimo quadrados"? Porque a fórmula pega o quadrado de cada residual antes de somar todos e isso aumenta o peso dado àquelas observações mais distantes da reta de regressão – chamadas de extremos ou outliers. Dessa forma, os mínimos quadrados ordinários "encaixam" a reta que minimiza a soma dos residuais ao quadrado conforme é ilustrado na imagem da página anterior.

Se os detalhes técnicos lhe deram dor de cabeça, você estará desculpado se entender apenas o ponto principal, que é o seguinte: os mínimos quadrados ordinários nos dão a melhor descrição de uma relação linear entre duas variáveis. O resultado não é somente uma reta, mas uma equação que descreve essa reta. Essa equação é conhecida como equação de regressão linear simples[3] e assume a seguinte forma apresenta a seguir:

$$y = a + bx \text{ ou}$$
$$y = \alpha + ßx$$

onde **y** é o peso em quilos; **a** é o intercepto, isto é, ponto em que a reta intercepta o eixo y (valor de y quando x = 0); **b** é a inclinação da reta; e **x** é a altura em centímetros. A inclinação da reta que encaixamos descreve a "melhor" relação linear entre altura e peso para essa amostra, conforme definida pelos mínimos quadrados ordinários. *A reta de regressão é perfeita?* É claro que não! Ela com certeza não descreve perfeitamente toda observação nos dados.

Por outro lado, ela é a melhor descrição que podemos obter para o que é claramente uma relação significativa entre altura e peso. Poderíamos escrever a nossa equação de regressão dessa forma:

$$peso = a + (altura)*x$$

Regressão Logística

[3] Existe também a Regressão Linear Múltipla, quando há mais de uma variável independente (Ex: **y** = **a** + **bx** + **cy** + **dz**).

REGRESSÃO LOGÍSTICA

Trata-se da ferramenta estatística que tem como objetivo produzir, a partir de um conjunto de observações, um modelo que permita a predição de valores tomados por uma variável categórica, frequentemente binária, a partir de uma série de variáveis explicativas contínuas e/ou binárias.

A ferramenta de regressão linear é eficaz, mas possui suas limitações! Por exemplo: ela permite trabalhar apenas com dados quantitativos (contínuos) e, não, com dados qualitativos (categóricos). Quando essa é a sua necessidade, você deve recorrer à regressão logística, que é um algoritmo de aprendizagem de máquina supervisionado utilizado para classificação (apesar de ter a palavra *regressão* em seu nome – cuidado com a pegadinha!).

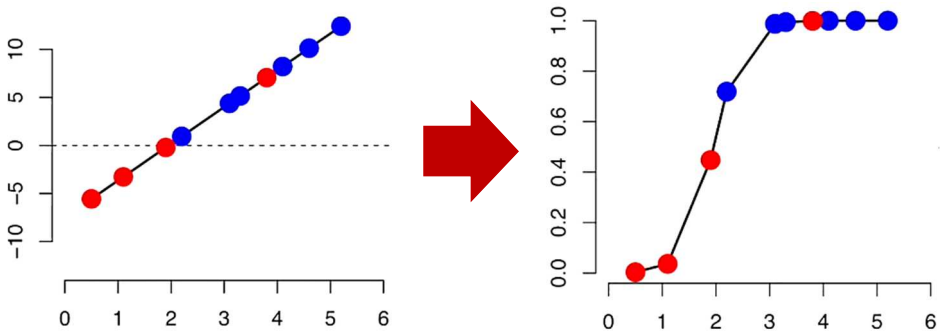
Em geral, a utilização da regressão logística se dá com categorias binárias, isto é, aquelas que podem assumir somente dois valores (Ex: grande ou pequeno, alto ou baixo, sim ou não, lucro ou prejuízo, válido ou inválido, entre outros). Vamos imaginar que queiramos definir se um determinado paciente está ou não infectado com coronavírus, logo nossas categorias são: infectado ou não-infectado.

Para tal, nós vamos reunir diversas informações contidas em um exame de sangue, tais como contagem de anticorpos, contagem de plaquetas, contagem de leucócitos, entre outros – essas informações são também chamadas de variáveis independentes. Em seguida, será aplicado um coeficiente ou peso a cada uma dessas variáveis que comporão uma função de regressão linear múltipla e retornará um determinado valor como resposta (variável dependente).

Ocorre que a regressão logística é um tipo de algoritmo de classificação, logo precisamos transformar esse valor real retornado pela regressão linear em uma das categorias pré-definidas por um supervisor. Para tal, nós temos que utilizar um modelo logístico (chamado *logit*) para fazer um mapeamento desse valor dentro de um intervalo entre [0, 1], que pode ser interpretado como a probabilidade de ser da categoria que nos interessa.

A função de ativação que recebe como entrada um número real [-∞, +∞] retornado por uma função de regressão linear e sempre retorna um número entre [0,1] é chamada de Função Sigmóide[4].

[4] Trata-se de uma função estritamente crescente que varia [-∞,+∞] horizontalmente e [0,1] verticalmente. Por conta de seu formato, pode ser chamada de Função em S.



Galera, todas as informações de cada pessoa entram no processo de treinamento do modelo de aprendizado de máquina. Em outras palavras, como eu tenho um conjunto de casos em que eu já sei se a pessoa estava infectada ou não, eu posso usá-lo para treinar meu modelo de modo que eu possa ir ajustando até encontrar um valor razoável. *Como assim, Diego?* Vamos supor que eu tenha em mãos aqui os exames de várias pessoas com coronavírus.

Eu posso ir ajustando os coeficientes das variáveis independentes da minha regressão linear para refletir um valor coerente de probabilidade após aplicar a regressão logística. Vejamos:

```
resultado = a + b*(qtd_anticorpos) + c*(qtd_leucócitos) + d*(qtd_plaquetas) ...
```

Eu coloco os valores de quantidade de anticorpos, leucócitos e plaquetas de uma pessoa que eu sei que está com coronavírus, o modelo ajusta os coeficientes (a, b, c, d) e retorna um resultado. Em seguida, após aplicar a função sigmóide nesse resultado, eu espero que ela retorne uma alta probabilidade de essa pessoa estar com coronavírus (Ex: 0,9), dado que a pessoa realmente está infectada com coronavírus, logo esse seria um resultado coerente.

Se ele retornar um valor como 0,2 (isto é, 20% de probabilidade de essa pessoa estar infectada com coronavírus), significa que o modelo ainda não está bacana porque sabemos que a pessoa efetivamente está infectada com coronavírus. É claro que – quanto mais dados de treinamento – mais o modelo de aprendizado de máquina ajusta os coeficientes e maiores as chances de ele retornar uma probabilidade coerente com a realidade.

Redes Neurais Feed-Forward

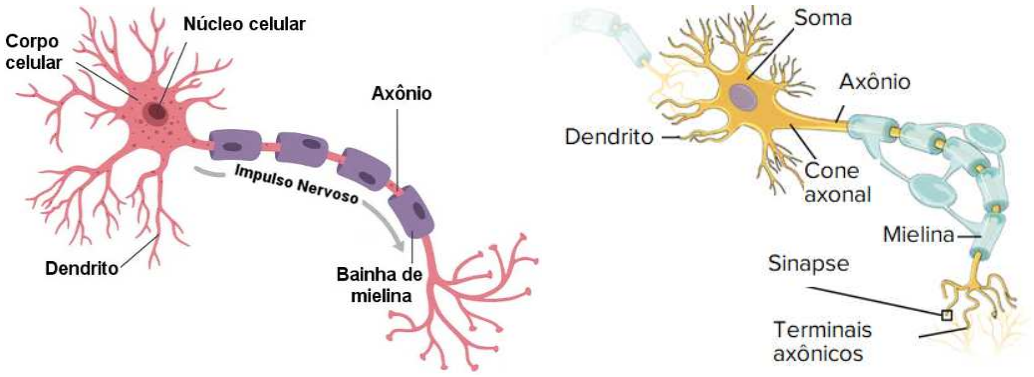
RELEVÂNCIA EM PROVA: MÉDIA

REDES NEURAIS FEED-FORWARD

Trata-se de um tipo de rede neural artificial em que os dados fluem em apenas uma direção, da entrada para a saída. É o tipo mais comum de redes neurais artificiais e é usado para tarefas de aprendizado supervisionado. Em uma rede feed-forward, os neurônios são organizados em camadas e o sinal se propaga de uma camada para outras. Cada neurônio recebe entradas dos neurônios da camada anterior, realiza uma soma ponderada das entradas e passa o resultado para a próxima camada.

Pássaros nos inspiraram a voar, raízes de *arctium* inspiraram a criação do velcro e a natureza inspirou muitas outras invenções. Parece lógico, então, olhar para a arquitetura do cérebro em busca de inspiração sobre como construir uma máquina inteligente. Esta é a ideia que inspirou as Redes Neurais Artificiais. Claro que, embora aviões tenham sido inspirados por pássaros, eles não precisam bater as asas. Assim, essa tecnologia foi ficando cada vez mais diferente dos neurônios.

A redes neurais artificiais são versáteis, poderosas e escaláveis, tornando-as ideais para lidar com tarefas de aprendizado de máquina altamente complexas, como classificar bilhões de imagens (Ex: Google Imagens), alimentar serviços de reconhecimento de voz (Ex: Siri da Apple), recomendar os melhores vídeos para assistir (Ex: YouTube Recommendations). No entanto, antes de discutirmos os neurônios artificiais, vamos dar uma olhada rápida em um neurônio biológico.



Trata-se de uma célula de aparência incomum encontrada principalmente em córtex cerebrais animais, composta basicamente por um corpo celular, que contém o núcleo e a maioria dos componentes complexos da célula; e muitas extensões ramificadas chamadas dendritos, além de uma extensão muito longa chamada de axônio. Os neurônios biológicos recebem impulsos elétricos curtos – chamados sinais – de outros neurônios por meio dessas sinapses.

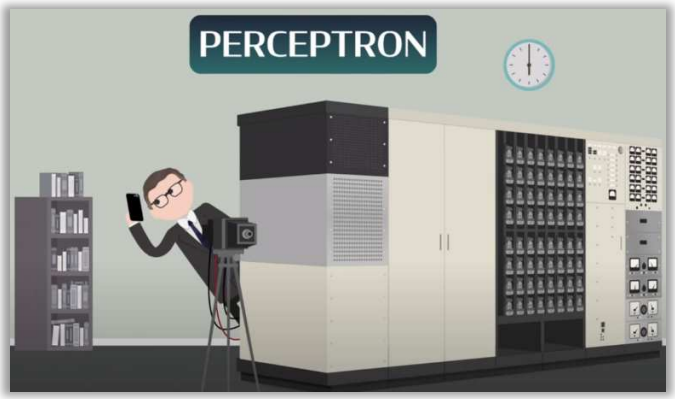
Quando um neurônio recebe um número suficiente de sinais de outros neurônios em alguns milissegundos, ele mesmo dispara seus próprios sinais. Observe que neurônios biológicos individuais parecem se comportar de uma maneira bastante simples, mas eles são organizados em

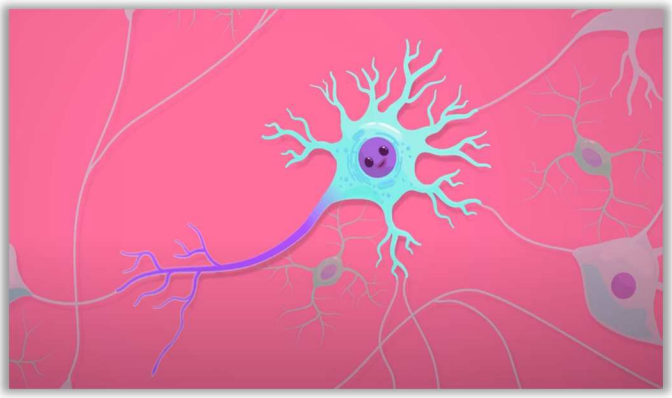
uma vasta rede de bilhões de neurônios – cada neurônio normalmente conectado a milhares de outros neurônios.

Cálculos altamente complexos podem ser realizados por uma vasta rede de neurônios bastante simples, da mesma forma que um formigueiro complexo pode emergir dos esforços combinados de formigas simples. Dois pesquisadores – Warren McCulloch e Walter Pitts – propuseram um modelo muito simples de neurônio biológico, que mais tarde ficou conhecido como neurônio artificial: ele tem uma ou mais entradas binárias (liga/desliga) e uma saída também binária.

O neurônio artificial funciona de maneira muito simples: ele ativa sua saída quando mais de um certo número de suas entradas estão ativas. Os pesquisadores mostraram que mesmo com um modelo tão simplificado é possível construir uma rede de neurônios artificiais que computa qualquer proposição lógica que você deseja. Pois bem! Em 1958, esses pesquisadores mostraram suas descobertas em uma conferência de inteligência artificial na Universidade de Dartmouth.

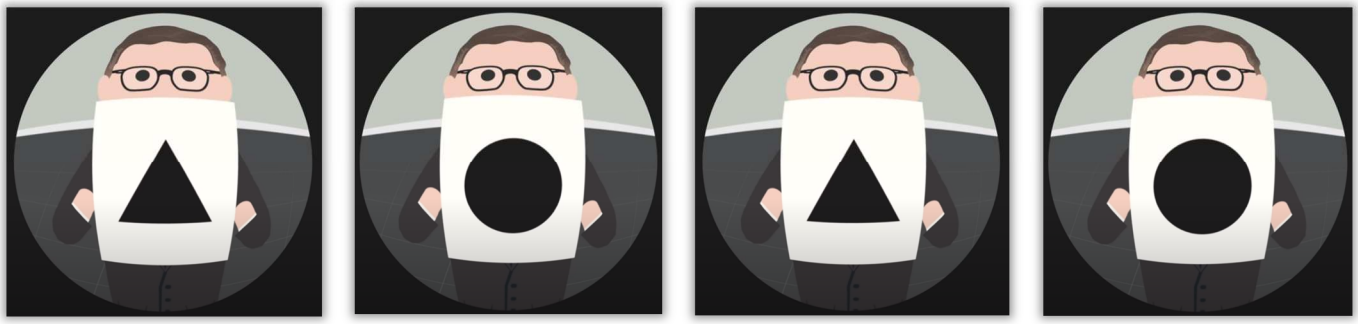
Como esse assunto é muito empolgante (*Vai me dizer que não é?* Não é possível!), um psicólogo chamado Frank Rosenblatt que assistiu a conferência e saiu de lá inspirado e determinado a criar um neurônio artificial. Seu objetivo era ensinar um computador gigantesco de inteligência artificial chamado Perceptron a se comportar como um neurônio ao classificar imagens como triângulos ou não-triângulos com sua supervisão/treinamento.





O psicólogo sabia já naquela época que neurônios são células que processam e transmitem mensagens utilizando sinais elétricos e químicos. Eles recebem um ou mais sinais de entrada de outras células, processam os sinais recebidos e depois emitem seu próprio sinal. Os neurônios, então, formam uma colossal rede de interconexão que é capaz de processar informações extremamente complexas.

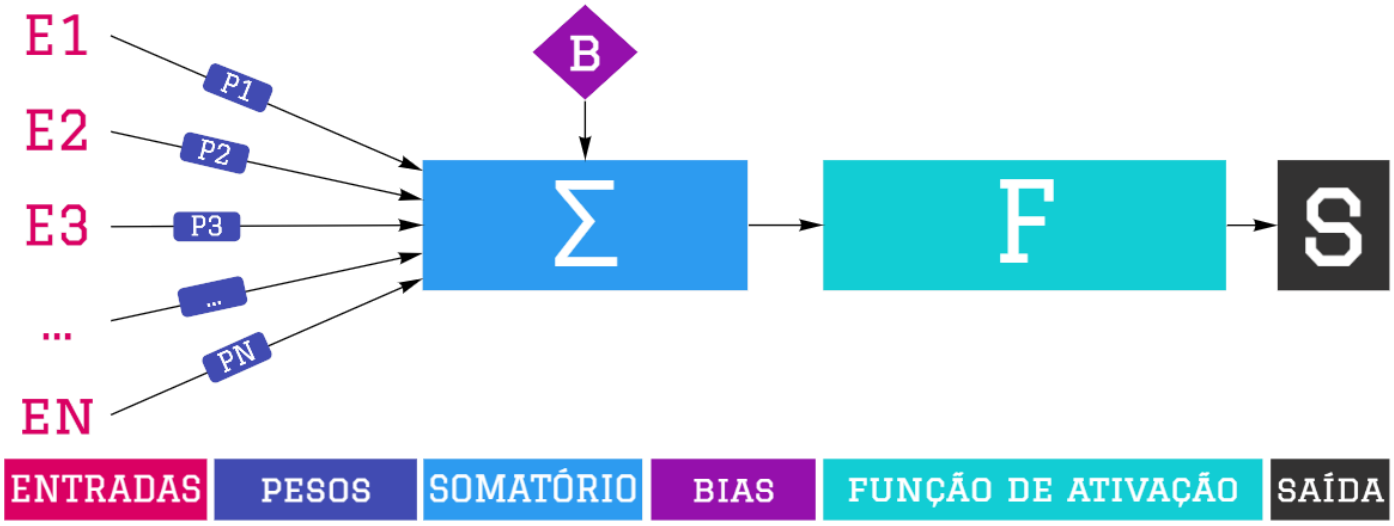
Rosenblatt conectou o Perceptron a uma câmera fotográfica de inacreditáveis 400 pixels (eu disse pixels e, não, megapixels). Essa era a tecnologia de ponta da época – cerca de um bilhão de vezes menos poderosa do que a câmera traseira de qualquer smartphone hoje em dia. A ideia era mostrar para a câmera a imagem de um triângulo ou não-triângulo (Ex: círculo). Dependendo se a câmera viu tinta ou não em cada pixel, ela enviaria um sinal elétrico diferente para o Perceptron.



O Perceptron somava os sinais elétricos referentes aos pixels que correspondiam com a forma de um triângulo – essa soma era chamada de carga elétrica total. Se a carga fosse acima de um certo limite estipulado pelo psicólogo, a máquina a enviaria para acender uma lâmpada – equivalente a um neurônio artificial falando: "Sim, isso é um triângulo!"; mas se a carga fosse abaixo do limite, ela não seria suficiente para acender a lâmpada – seria o equivalente a dizer: "Não é um triângulo!".

No início, o Perceptron estava basicamente fazendo suposições aleatórias, até que Rosenblatt começou a treinar o algoritmo. Quando o Perceptron acertava, ele apertava um botão para confirmar e fim; mas quando o Perceptron errava, ele apertava um botão para indicar o erro, o que disparava uma cadeia de eventos e cálculos para reajustar o limite de carga elétrica suficiente para acender a lâmpada de forma que as chances de a máquina acertar nas próximas vezes aumentava.

Vocês pegaram a ideia? O perceptron é a unidade básica de uma Rede Neural Artificial (RNA), sendo equivalente a um neurônio biológico. Vejamos abaixo como seria sua estrutura básica:



Note que temos um conjunto de n entradas (e_1, e_2, e_3, ..., e_n) que são multiplicadas por pesos específicos associados a cada entrada (p_1, p_2, p_3, ..., p_n). Cada um desses pesos é livremente ajustável de forma independente dos demais. Em seguida, nós realizamos a soma de cada entrada multiplicada por seu respectivo peso associado. E o próximo passo é somar tudo isso com uma entrada especial denominada viés ou *bias* (b).

Em outras palavras, podemos afirmar que o peso de uma entrada representa seu grau de força ou influência no resultado – quanto maior o peso de uma determinada entrada, maior será a influência no resultado; e o viés (bias) é um valor que pode ser ajustado para aumentar ou diminuir a força do sinal ao adicionar um valor positivo/negativo com o intuito de regular o formato/curvatura da função e ajustá-la ao propósito desejado (veremos em detalhes mais à frente).

Ora, isso lembra alguma coisa para vocês? Estamos multiplicando valores por um coeficiente, somando-os e adicionando uma constante! Bem, isso parece uma regressão linear múltipla:

$$y = a + bx + cy + dz \ldots$$

No nosso caso, seria algo mais parecido como:

$$s = b + p_1e_1 + p_2e_2 + p_3e_3 \ldots + p_ne_n$$

A lógica básica de um perceptron até essa parte inicial é realmente parecida com uma regressão linear, mas ainda não acabamos de ver a sua estrutura. Note que a etapa seguinte passa o resultado da regressão linear por uma função de ativação, responsável por fazer uma transformação não-linear do resultado da etapa anterior e definir se um nó será ativado ou não. Explicando um pouco melhor sem entrar nos detalhes matemáticos...

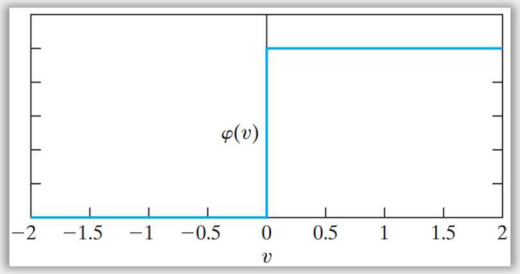
Funções lineares (Ex: regressão linear) são aquelas que podem ser representadas em um plano cartesiano como uma reta. O objetivo da função de ativação é transformar o formato de reta em um formato não linear, tornando a função mais flexível/adaptável para tarefas complexas – além de permitir que pequenas alterações na entrada de dados não causem grandes alterações na saída de dados. Existem duas grandes classes de funções de ativação: funções de limite e funções sigmóides.

Os perceptrons originais utilizavam funções de limite. Nesse caso, se o resultado da função de ativação fosse maior ou igual a um determinado limite (em inglês, *threshold*), então essa função retornaria `1` (nó ativado); se o resultado da função de ativação fosse menor que um determinado limite, então essa função retornaria `0` (nó não-ativado). Existem outras variações, mas funções desse tipo sempre têm formato de um degrau (em inglês, *step*).

É importante destacar que o significado dos valores `1` e `0` dependerá do contexto de utilização, podendo ser ligado ou desligado, ativo ou inativo, masculino ou feminino, triângulo ou não-triângulo, entre outros. Ocorre que a maioria das aplicações necessita de um resultado probabilístico (contínuo) e, não, de um resultado binário (discreto) – e a função limite não é capaz de retornar esse tipo de resultado.

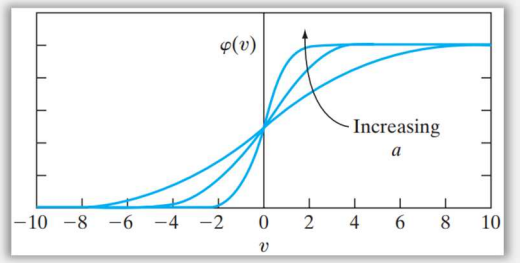
FUNÇÃO DE ATIVAÇÃO	FUNÇÕES DE LIMITE
DESCRIÇÃO DA FUNÇÃO	Essa função compara um valor com um determinado limite (nesse caso, o limite é o) e

	decide se um neurônio será ativado (1) ou não será ativado (0).
REPRESENTAÇÃO DA FÓRMULA	$f(x) = \begin{cases} 0 & for\ x < 0 \\ 1 & for\ x \geqslant 0 \end{cases}$



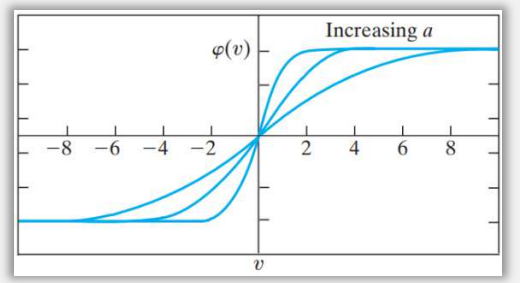
Por conta disso, perceptrons modernos utilizam funções sigmóide. Nesse caso, podemos ver o resultado do perceptron como uma espécie de regressão logística (estudada no tópico anterior). Trata-se de uma função estritamente crescente que recebe um valor real qualquer que varia horizontalmente entre [-∞,+∞] e varia verticalmente [0,1]. A função sigmóide mais famosa é a função logística apresentada a seguir (frequentemente são tratadas como sinônimos):

FUNÇÃO DE ATIVAÇÃO	FUNÇÕES LOGÍSTICAS
DESCRIÇÃO DA FUNÇÃO	Essa função recebe um valor real qualquer como entrada [-∞,+∞] e retorna um valor de saída entre 0 e 1.
REPRESENTAÇÃO DA FÓRMULA	$f(x) = \frac{1}{1+e^{-x}}$



Há também o caso em que desejamos que o resultado assuma valores entre [-1,1]. Para tal, podemos utilizar um outro tipo de função sigmóide chamada de tangente hiperbólica:

FUNÇÃO DE ATIVAÇÃO	FUNÇÃO TANGENTE HIPERBÓLICA
DESCRIÇÃO DA FUNÇÃO	Essa função recebe um valor real qualquer como entrada [-∞,+∞] e retorna um valor de saída entre -1 e 1.
REPRESENTAÇÃO DA FUNÇÃO	$f(x) = \frac{(e^{x} - e^{-x})}{(e^{x} + e^{-x})}$



Essa transformação não-linear realizada pelas funções de ativação permite tornar o perceptron mais capaz de aprender e executar tarefas complexas. Pronto! Agora nós passamos por todo o funcionamento do perceptron original e moderno, mas ainda não sabemos como essa unidade de redes neurais vai aprender alguma coisa! A ideia é atribuir um valor aleatório aos pesos e comparar o valor de saída do perceptron com o valor esperado (que nós conhecemos de antemão).

Quanto mais iterações houver desse processo de aprendizado, mais os pesos serão ajustados, mais treinado estará o algoritmo e menor será o erro médio das suas previsões. Vamos ver um exemplo

para consolidar o entendimento: suponha que queiramos saber o gênero de determinada pessoa (masculino ou feminino). Considerem também que nós possuímos uma tabela contendo em cada linha dados de vários atributos de milhares de pessoas.

Dentre os atributos dessa tabela, temos: altura, peso, envergadura, cor do cabelo, classe social e escolaridade – e nós já sabemos de antemão o gênero de cada uma das pessoas. Cada entrada do nosso perceptron será correspondente aos dados de uma determinada pessoa, logo ele terá seis entradas – uma para cada atributo. O algoritmo começa atribuindo um peso aleatório para cada uma dessas entradas.

Em seguida, ele faz a multiplicação dos valores de entrada pelos seus respectivos pesos aleatórios, soma tudo (inclusive o viés), passa por uma função de ativação de não linearidade e retorna um valor de probabilidade de pessoa ser do gênero masculino ou feminino. Como nós sabemos o resultado esperado (isto é, gênero), nós podemos treinar o algoritmo corrigindo ou ratificando o resultado obtido pelo perceptron para cada uma das pessoas da nossa tabela.

A cada erro, o perceptron poderá fazer um ajuste nos pesos a fim de reduzir falhas e fornecer uma previsão mais acurada. É provável que, após algumas iterações, o algoritmo perceba que alguns atributos têm mais influência para decidir sobre o gênero biológico de uma pessoa do que outros. Intuitivamente, podemos dizer uma pessoa de 2,00m de altura tem maior probabilidade de ser do gênero masculino do que ser do gênero feminino.

Da mesma forma, podemos dizer intuitivamente que uma pessoa com cabelo vermelho tem maior probabilidade de ser do gênero feminino do que ser do gênero masculino. Por outro lado, podemos dizer também que a classe social não é um atributo que influencia bastante na probabilidade de uma pessoa ser do gênero masculino ou feminino. Dessa forma, o algoritmo vai aprendendo, vai se ajustando e vai dando pesos mais altos aos atributos efetivamente mais importantes.

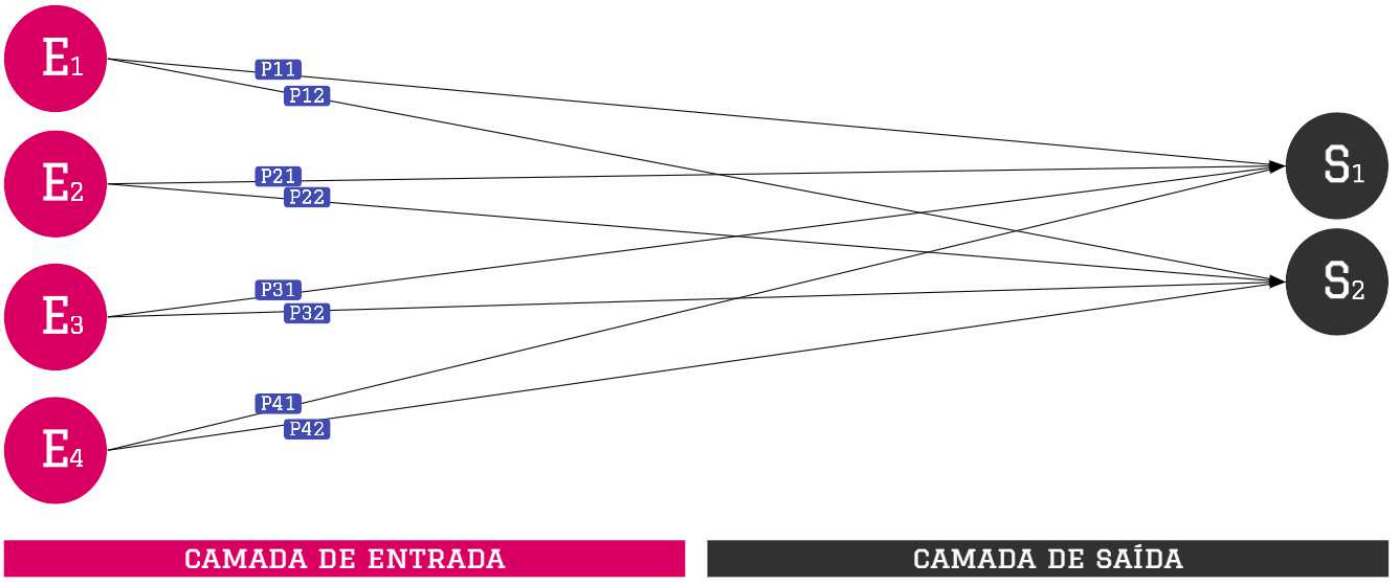
Dito isso, é importante saber que o perceptron é a unidade básica de uma rede neural artificial, assim como o neurônio é a unidade básica de uma rede neural biológica. Vejamos uma comparação:

NEURÔNIO BIOLÓGICO	NEURÔNIO ARTIFICIAL
Célula (Núcleo)	Nó (Perceptron)
Dendritos	Entradas
Sinapses	Pesos ou Conexões
Axônio	Saídas

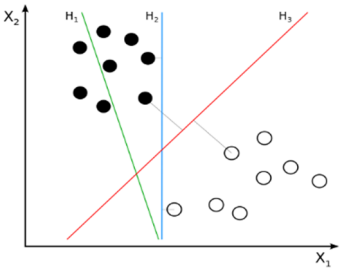
Agora que entendemos os fundamentos básicos dos perceptrons, podemos evoluir um pouco mais em nosso estudo! Na prática, os perceptrons não são utilizados isoladamente – eles são combinados com diversos outros perceptrons. Existem diversas formas de organização de perceptrons, sendo a organização por camadas a mais comum. Aquelas que serão objeto de estudo nesse tópico serão: Single Layer Perceptrons (SLP) ou Multilayer Perceptrons (MLP).

Single Layer Perceptrons (SLP)

A imagem seguinte apresenta uma rede neural do tipo Single Layer Perceptron (SLP). Trata-se de uma rede neural que organiza os perceptrons em uma única camada de processamento. *Professor, não seriam duas camadas como mostra a imagem?* Não! Apesar de possuir duas camadas, apenas os perceptrons (que agora vamos chamar de nós) da camada de saída realizam processamentos – os nós da camada de entrada apenas transferem os valores diretamente para a camada de saída.



Esse tipo de rede neural é raramente utilizado justamente por utilizar perceptrons originais, isto é, aqueles cuja função de ativação é uma função de limite (em formato de degrau). Esse tipo de redes neurais é mais adequado para resolver problemas de classificação, porém apenas aqueles em que as classes são linearmente separáveis. *Como é, Diego?* Isso significa que ele só lida com situações em que é possível traçar ao menos uma linha reta separando classes em um plano coordenado[5].



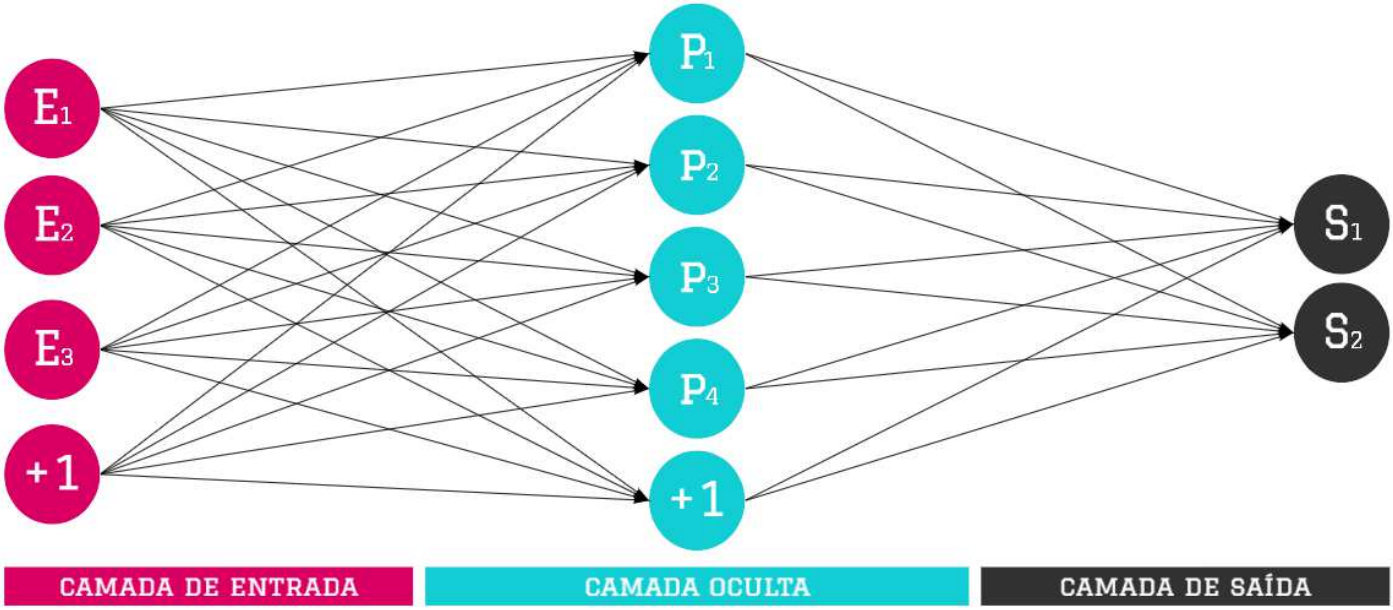
Note pela imagem anterior que as retas H2 e H3 permitem separar esse conjunto de dados de em bolas sólidas e bolas vazias; já a reta H1 não separa corretamente. O SLP é capaz de aprender apenas padrões linearmente separáveis. Essas limitações importantes fazem esse tipo de rede neural ser raramente utilizada, portanto nem vamos nos aprofundar. Na verdade, eu passei essa noção mais para vocês entenderem melhor o próximo tipo de rede neural: Multilayer Perceptrons (MLP).

Multilayer Perceptrons (MLP)

Trata-se de uma arquitetura de redes neurais artificiais que utiliza múltiplas camadas de perceptrons. Nesse caso, temos duas camadas que realizam processamentos: camada oculta e

[5] Seria o equivalente a um hiperplano dentro de um espaço tridimensional.

camada de saída. Da mesma forma da arquitetura anterior, os nós da camada de entrada apenas transferem os valores (com seus respectivos pesos) para os nós da camada oculta ou camada escondida – eles são os únicos perceptrons que não executam uma função de ativação.



Já em contraste com a arquitetura anterior, aqui são utilizados perceptrons modernos, isto é, aqueles cuja função de ativação é uma função sigmóide. Aos dados de entrada são aplicados pesos, em seguida eles são sendo processados pela camada oculta até chegarem a um resultado de saída. Um exemplo resultado de saída poderia ser a probabilidade de uma pessoa ser do gênero masculino ou feminino, de forma que aquela que tivesse maior probabilidade seria a previsão do algoritmo.

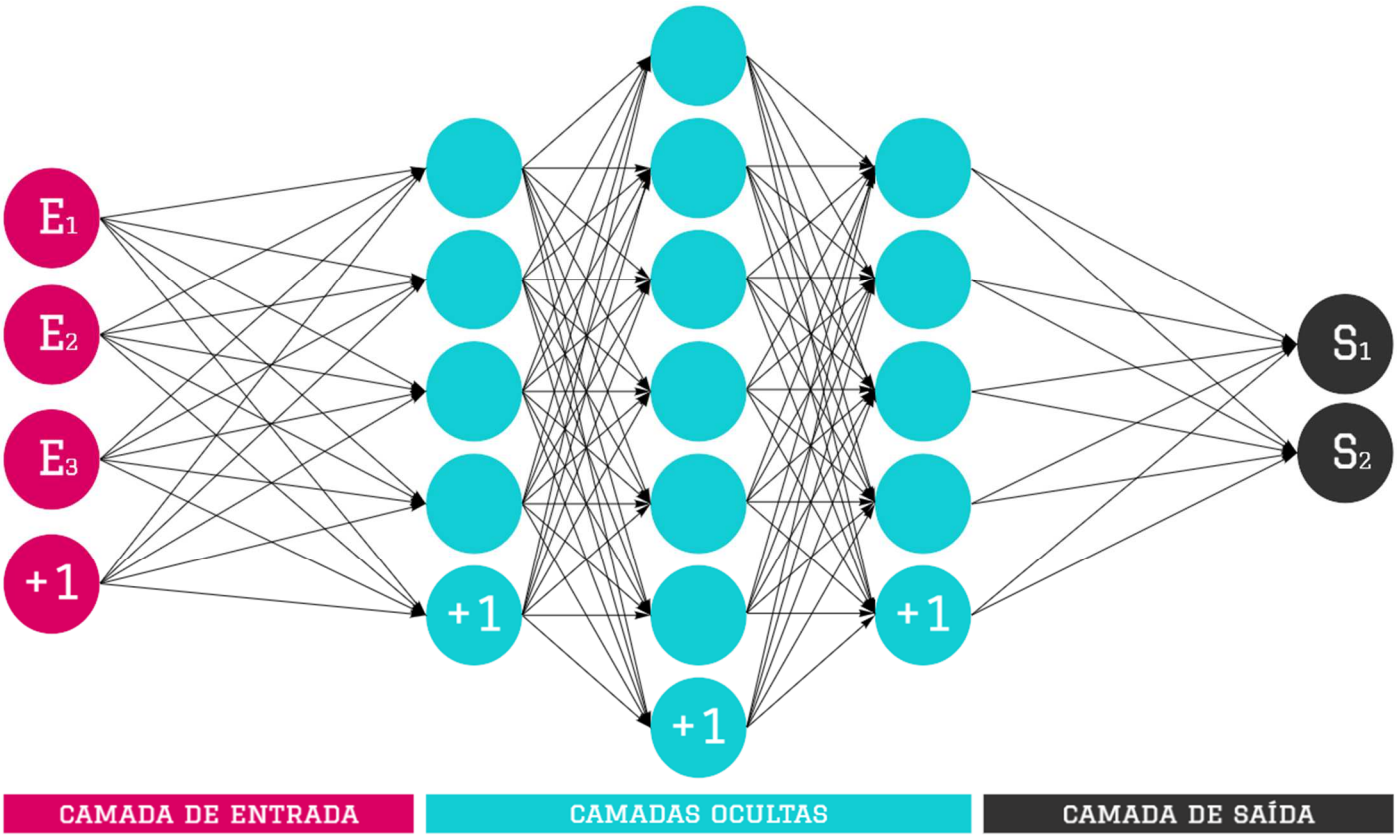
Em MLP, a lógica de conexão é: todos os elementos de uma camada se conectam a todos os elementos da camada seguinte e apenas a eles! Em outras palavras, notem que elementos da mesma camada não se conectam entre si. Até existem diversas outras arquiteturas de redes neurais (Ex: Convolutivas (CNN), Recorrentes (RNN), Long Short-Term Memory (LSTM)), mas o nosso foco aqui é em redes neurais de multicamadas. Há outro detalhe: são redes neurais feed-forward.

Isso significa que os sinais de informação de uma camada somente alimentam a próxima camada – jamais alimentam a camada anterior ou a camada atual. Logo, os sinais não formam ciclos (em contraste com redes neurais recorrentes, por exemplo). Em redes neurais feed-forward, os sinais avançam sempre para frente, fluindo da camada atual para a próxima de forma sequencial. Por falar em camadas, é possível que a rede neural tenha várias camadas ocultas[6].

Atualmente existem redes neurais com mais de mil camadas escondidas, além de trilhões de perceptrons. *Agora vocês se lembram que nós falamos de Deep Learning no início da aula?* Nós dissemos que era uma aplicação de aprendizado de máquina! Pois é, sendo mais específico, o

[6] Esse termo se refere ao fato de que essa parte da rede neural não é vista diretamente nem pelas nem pelas entradas nem pelas saídas. É como se fosse uma caixa-preta em que entregamos um valor, ela faz uma série de processamentos e retorna um resultado.

aprendizado profundo é uma rede neural artificial em que múltiplas camadas escondidas (ao menos três[7]) de processamento são utilizadas para extrair progressivamente características de dados.



A imagem anterior apresenta um exemplo de aprendizado profundo. A ideia é que, após um grande número de iterações (repetições), os pesos estejam relativamente ajustados e a rede convirja para um ponto de estabilidade, isto é, há pouco ou nenhum ajuste dos pesos e os erros são bem mais raros. Quando chegamos nesse ponto, podemos utilizar essa rede neural para generalizar um problema para futuros novos dados de maneira assertiva.

Esse fenômeno em que erros de previsão entre saídas obtidas e saídas esperadas de uma rede neural artificial são quantificados por meio de uma função de custo/perda (Ex: Erro Quadrático Médio – EQM) e retornam para a rede em forma de ajuste dos pesos e vieses é chamado de Retropropagação (ou *Backpropagation*). Sendo um pouco mais detalhista, esse algoritmo consiste basicamente em duas etapas principais:

1. Etapa de Propagação: as entradas fluem através das camadas ocultas da rede neural e previsões são obtidas na camada de saída;

2. Etapa de Retropropagação: calcula-se o gradiente da função de custo/perda na camada de saída e ele é utilizado para atualizar os pesos (inclusive o viés) recursivamente.

[7] Não há um consenso sobre a quantidade mínima de camadas ocultas para se considerar uma rede neural como rede profunda.

BACKPROPAGATION

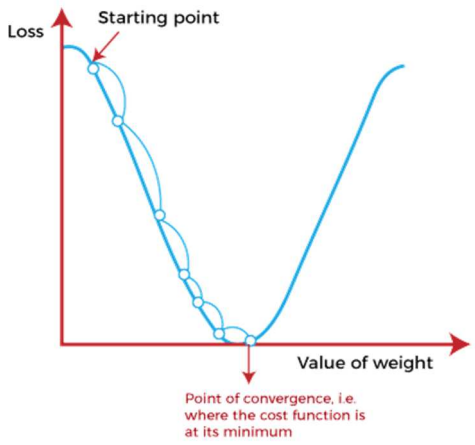
Trata-se de um algoritmo para treinamento de redes neurais artificiais que utiliza gradiente descendente para ajustar pesos na rede a fim de reduzir o custo total e minimizar os erros. O custo é calculado comparando a saída da rede com a saída desejada e calculando a diferença entre elas. A retropropagação usa o erro para calcular o gradiente da função de custo em relação aos pesos e desvios na rede e ajusta os pesos de acordo. Este processo é repetido até que o custo atinja um nível aceitável.

O *backpropagation* é indiscutivelmente o algoritmo mais importante na história das redes neurais – sem esse algoritmo, seria quase impossível treinar redes de aprendizagem profunda da forma que vemos hoje. *E o que é gradiente, Diego?* Gradiente é um vetor de derivadas parciais (primeira derivada) de uma função de saída em relação a valores de entrada. *Falei grego, né?* Vou explicar rapidamente porque só vi isso cair até hoje em um concurso...

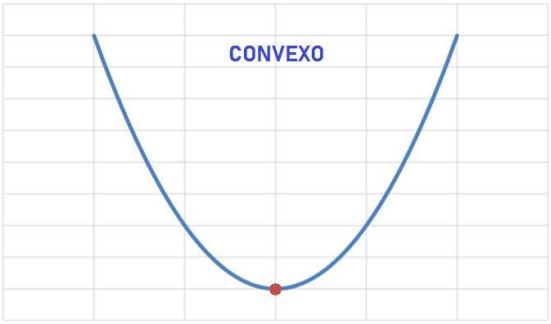
No aprendizado de máquina, frequentemente nós queremos otimizar alguma coisa. Para tal, nós vamos alterando valores de uma função (às vezes aleatoriamente, às vezes de forma estruturada) para retornar a melhor saída possível – em geral, buscamos os coeficientes que minimizam o erro. Por exemplo: queremos treinar um algoritmo para, dada uma imagem, reconhecer se é um gato ou um cachorro. *O que fazemos?*

Vamos alterando os coeficientes de uma determinada função até chegar ao menor erro possível. Falando metaforicamente: imaginem que uma pessoa foi sequestrada e liberada no topo de uma montanha. Ela está sozinha e deseja voltar para casa. Para tal, ela precisa descer até a base da montanha, mas tem muita névoa – ao ponto de ela enxergar bem só 1 metro à frente dela. *Qual seria a melhor estratégia para essa pessoa?*

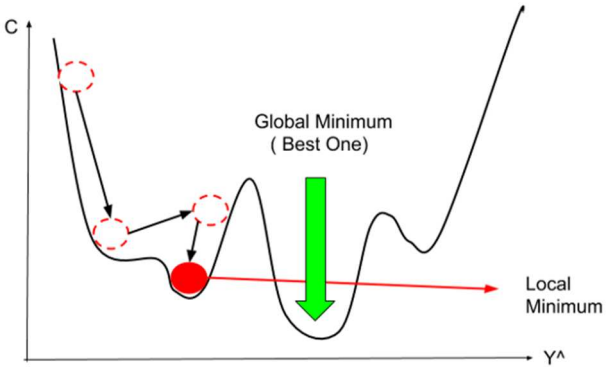
Ela poderia dar um passo na direção sul e voltar, um passo na direção norte e voltar, um passo na direção leste e voltar e um passo na direção oeste e voltar. Pronto! Ela compara qual passo chega em uma altitude menor e desce nessa direção. Se ela fizer isso repetidamente, a cada passo ela descerá para uma altitude mais baixa. Voltando à realidade, essa montanha pode ser representada como na imagem seguinte:



Notem que, para cada ponto, é calculado o gradiente e vamos descendo até chegar ao ponto mais baixo, por isso é chamado de gradiente descendente. Ocorre que, assim como as montanhas, as funções podem ser convexas ou não convexas. Uma função é dita convexa somente quando uma linha traçada entre quaisquer dois pontos está sempre acima da curva; o que não ocorre em funções não convexas. Vejam as diferenças:



Note que na curva que representa a função convexa, temos apenas um ponto mínimo; e na curva que representa a função não convexa, temos mais de um ponto mínimo. Vejam outro exemplo:



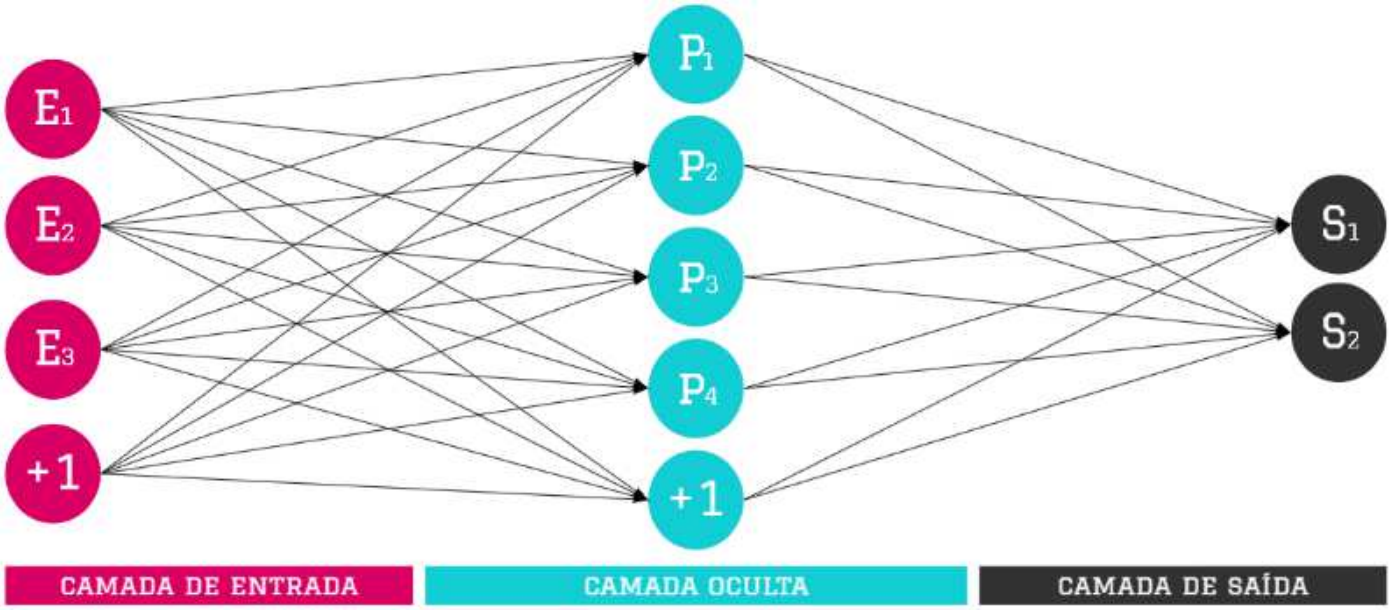
Notem que, nas funções não convexas, temos um mínimo global, mas podemos ter vários mínimos locais. Para finalizar esse assunto, é importante saber que existem dois tipos de gradiente descendente: em lote (*batch*) e estocástico. O gradiente descendente em lote é adequado para funções convexas, contínuas e diferenciáveis, e utiliza – a cada iteração – todos os dados da base de treinamento para otimizar os pesos dos nós de uma rede neural.

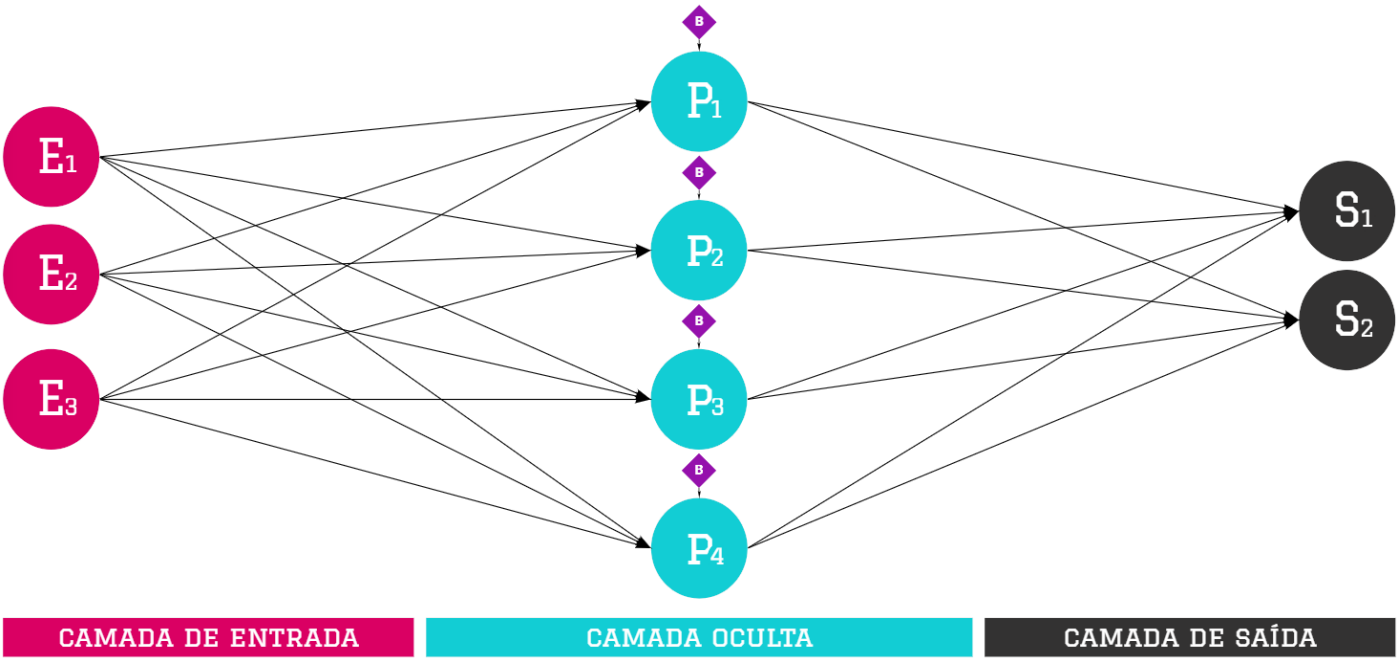
Já o gradiente descendente estocástico é adequado para funções não convexas, e escolhe uma sub-amostra aleatória dos dados da base de treinamento para otimizar parâmetros. *Diego, se a questão não especificar de qual gradiente descendente ela está falando, qual deles eu considero como padrão?* O padrão é pensar no gradiente descendente em lote. Existe uma matemática pesada envolvendo todo esse assunto, mas evidentemente não entraremos nesse nível de detalhe.

Agora vamos falar de um ponto que eu acho que vocês não notaram: há duas maneiras de representar o viés em uma arquitetura de multicamadas de perceptrons. É possível representá-lo como um valor externo adicionado à soma ponderada de cada perceptron ou como um nó sempre

ativo que recebe uma entrada de valor fixo = 1 e é multiplicado por um valor de peso variável. Ambos são matematicamente idênticos! *E por que existe o viés?*

Considere um exemplo simples: você tem um perceptron com dois nós de entrada e_1 e e_2 e um nó de saída s. As entradas e_1 e e_2 são recursos binários e, em nosso exemplo, têm o valor $e_1 = e_2 = 0$. Ora, se os valores de entrada são 0, então os pesos não terão nenhuma relevância porque qualquer valor multiplicado por 0 é 0. Se não existisse o viés, o nó de saída retornaria 0. Ele é útil, portanto, para aumentar a flexibilidade do modelo para se adequar aos dados.





Note que, no segundo caso, o nó especial de viés é inserido na camada de entrada e nas camadas ocultas, mas não faz sentido incluí-lo na camada de saída. Além disso, não está representado na imagem, mas cada segmento de reta que liga representa um valor da camada anterior multiplicado

pelo seu respectivo peso. Já caiu um nome alternativo para o viés em outras provas chamado Termo de Interceptação (ou *Intercept Term*).

É análogo ao intercepto em um modelo de regressão linear e tem exatamente a mesma função – representa o valor no qual uma função cruza o eixo y. Como se trata de um valor constante independente de outras variáveis, mesmo que as entradas sejam `0`, ainda teremos um valor de saída. Bem, para finalizar é importante dizer que nesse tópico falamos apenas da aplicação de redes neurais no aprendizado supervisionado.

No entanto, é importante destacar que há aplicações de redes neurais artificiais também em aprendizado não-supervisionado – como são bem mais raros, não entramos em detalhes aqui nessa aula. Por fim, a mensagem que eu gostaria de deixar para vocês é que a razão pela qual o aprendizado de máquina – especialmente o aprendizado profundo – tem um papel dominante atualmente se dá também pela quantidade gigantesca de dados que temos disponíveis hoje em dia.

A junção do avanço tecnológico colossal em relação aos processadores de computadores de grande porte com um conjunto de dados de treinamento virtualmente infinito em alguns casos permite o treinamento de redes neurais com um número surreal de perceptrons. E essas redes conseguem identificar padrões estatísticos cada vez mais sutis que permitem fazer coisas inimagináveis alguns anos atrás. Eu sei que foi um tópico pesado, mas isso é o estado da arte em tecnologia ;)

Fontes de Erro em Modelos Preditivos

RELEVÂNCIA EM PROVA: BAIXÍSSIMA

Um modelo preditivo busca acertar o máximo de previsões sobre novos dados (dados desconhecidos – aqueles que não fizeram parte dos dados do conjunto de treinamento do algoritmo de aprendizado de máquina). No entanto, erros acontecem! Por melhor que seja um modelo preditivo, ele possuirá uma taxa de erros. *E o que seria um erro em um modelo preditivo?* Trata-se de uma diferença entre o valor previsto e o valor real obtido. Legal...

E por que erros acontecem em um modelo preditivo? Atualmente existem tecnologias de hardware e software modernas capazes de realizar processamentos extremamente complexos que executam cálculos matemáticos absurdamente avançados e, ainda assim, nossos modelos preditivos ainda erram previsões. *Por que isso ocorre?* Vamos novamente imaginar o cenário em que um algoritmo busca prever o gênero de uma pessoa baseada em um conjunto de características.

Suponha que uma tabela contenha 100 milhões de linhas, cada uma representando uma pessoa. Nas colunas, há dados sobre diversas variáveis disponíveis sobre essas pessoas (Ex: altura, peso, idade, cor do cabelo, envergadura, classe social, nacionalidade, quantidade de plaquetas, taxa de testosterona, taxa de estrogênio, percentual de gordura, etc). Um bom algoritmo vai aprender com essas 100 milhões de entradas e ajustará os pesos de cada característica.

Ele provavelmente dará um peso baixíssimo para o atributo de classe social, uma vez que a taxa de homens e mulheres por classe social é praticamente idêntica, logo esse atributo influenciará pouco na previsão de gênero de uma pessoa. Por outro lado, o percentual de gordura terá um peso maior, porque geralmente homens tem menor percentual de gordura que mulheres. É provável que esse algoritmo tenha uma taxa de acerto em suas previsões de, por exemplo, 99,00%.

Isso significa que, a cada 100 observações, esse modelo erra uma única previsão! Apenas uma, mas erra! *E por quê?* Porque é possível existir um homem com altura e peso medianos, baixa taxa de testosterona, alta taxa de estrogênio, entre outros atributos. Dito de outra forma, um algoritmo preditivo geralmente não é capaz de eliminar todas as possíveis zonas de interseção entre diversas categorias de classificação. *E quais são as fontes de erro de um modelo preditivo?*

Quando queremos prever o resultado de um fenômeno, idealmente precisamos modelá-lo. Logo, os dados observados a partir de um fenômeno qualquer pode ser modelado como:

Note que os dados observados podem ser generalizados como uma função das variáveis disponíveis + uma função de variáveis indisponíveis + variações. O dado observado ao final é a previsão se uma pessoa é homem ou mulher; as variáveis disponíveis são aquelas que efetivamente conhecemos e estão presentes em nossa tabela, tal como peso, altura, etc; as variáveis desconhecidas são aquelas que influenciam no resultado, mas nós não as conhecemos. *Como assim, Diego?*

Uma variável desconhecida poderia ser a taxa de hemoglobina no sangue, que é um grande indicativo de gênero – dado que homens possuem uma taxa maior que mulheres. Já as variações podem ser divididas em aleatoriedades e erros: a primeira indica um possível fenômeno que não obedece a nenhum padrão estatístico; e a segunda pode ocorrer de diversas formas tais como erros de ruídos, erros de medida, erros de aproximação, erros de registro, entre outros.

Erros de ruídos ocorrem, por exemplo, quando o rádio começa a chiar – o que significa que ele está captando sinais que não deveriam estar sendo captados; erros de medida ocorrem quando, por exemplo, há um desgaste no instrumento de medida, o que faz com que os resultados obtidos sejam diferentes; erros de aproximação ocorrem quando fazemos, por exemplo, arredondamentos nas medidas; erros de registros ocorrem quando, por exemplo, anotamos errado as medidas.

Note, portanto, que o resultado obtido (homem ou mulher) depende de funções que eu realmente consigo modelar, mas também de funções de variáveis desconhecidas (que não podem ser modeladas), além de padrões aleatórios que – por definição – não podem ser modelados individualmente e um conjunto de possíveis erros sobre os quais nem sempre podemos modelar de forma satisfatória.

Vocês se lembram que uma ideia fundamental do aprendizado de máquina é minimizar o erro? Pois é! Conforme podemos notar pelo diagrama apresentado, nada podemos fazer quanto às variáveis desconhecidas ou aleatoriedades, mas podemos tentar minimizar erros de ruído, medida, registro, aproximação, entre outros. *Dito isso, por que modelos erram?* Em regra, porque ele não é capaz de modelar de forma satisfatória todos os elementos que compõem o resultado obtido de um dado.

Não é possível modelar a função de variáveis desconhecidas assim como a aleatoriedade de um dado individual, justamente por ser aleatória[8]; é possível modelar alguns erros, mas nem sempre de forma satisfatória; é difícil separar no resultado obtido qual é a contribuição da função de variáveis disponíveis do restante dos elementos; é possível que o modelo encontre padrões que não existem por conta dos outros elementos; é possível haver *overfitting* ou *underffiting* (veremos adiante).

Por fim, um bom modelo encontra padrões que podem ser generalizados por uma função que reduz o erro de generalização obtido para entradas desconhecidas. Lembrando que a generalização é a capacidade de aprender com os dados disponíveis as regras gerais que você pode aplicar a todos os

[8] Apesar de complexo, é até possível modelar a aleatoriedade de um conjunto de dados, mas não de dados individuais.

outros dados. Dados fora da amostra de treinamento, portanto, tornam-se essenciais para descobrir se é possível aprender com os dados e até que ponto. *Fechou?* :)

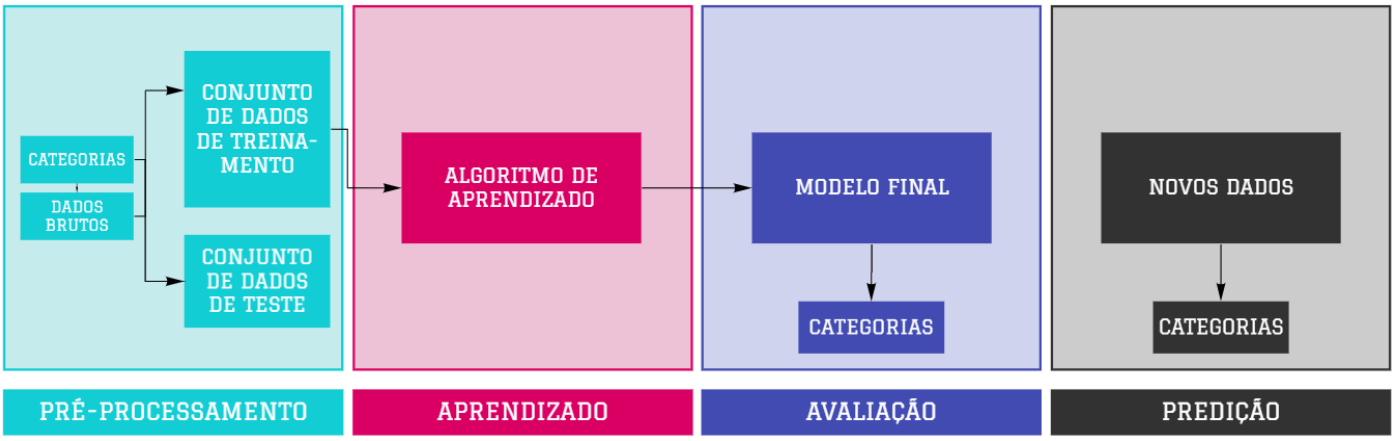
Validação e Avaliação de Modelos Preditivos

RELEVÂNCIA EM PROVA: BAIXA

VALIDAÇÃO/AVALIAÇÃO DE MODELOS PREDITIVOS

Trata-se do processo de avaliar o desempenho de um modelo preditivo em um conjunto de dados separado que não foi usado para treinamento, a fim de estimar o desempenho do modelo em dados não vistos. É uma etapa essencial no desenvolvimento de um modelo preditivo e é usado para avaliar a precisão e a confiabilidade do modelo.

Vamos passar rapidamente pelo fluxo de processos típico de aprendizado de máquina para modelos preditivos. Para tal, vejam o diagrama a seguir:



O pré-processamento é a primeira etapa na construção de um modelo de aprendizado de máquina. Em um contexto de aprendizado supervisionado, são recebidos dados brutos (conhecidos como variáveis independentes) e as possíveis categorias que o modelo tentará prever (conhecidas como variáveis dependentes). É realizada a limpeza e formatação dos dados brutos, além da extração e seleção de características, isto é, os atributos serão avaliados.

Essa etapa também é responsável por remover possíveis características redundantes, excessivas, desnecessárias ou altamente correlacionadas – também chamado de redução de dimensionalidade (que veremos mais à frente). Em seguida, separamos os dados de forma aleatória em duas partes: conjunto de dados de treinamento (ou indução) e conjunto de dados de teste. Em geral, divide-se de forma que o primeiro seja maior que o segundo – mas isso não é uma regra!

Em poucas palavras, podemos dizer que os dados de treinamento são utilizados para construir um modelo, enquanto os dados de teste são utilizados para validar o modelo construído. *Professor, o que seria esse tal modelo?* Um modelo é basicamente uma equação matemática utilizada para definir valores de saída a partir de valores de entrada. A ideia do algoritmo de aprendizado de máquina é ir ajustando essa equação até que ela consiga fazer previsões satisfatórias! Voltando...

Nós podemos afirmar que os dados de treinamento são aqueles utilizados como fonte para identificação de um padrão estatístico consolidado que possa ser representado por um modelo de aprendizado de máquina, isto é, são os dados que vamos utilizar para treinar a máquina. Já os dados de teste são aqueles que não foram utilizados para treinamento e serão utilizados apenas para a avaliação do desempenho do modelo, isto é, são dados de teste que a máquina nunca teve contato.

Uma vez que os conjuntos de dados estejam prontos, o segundo passo é selecionar um algoritmo para executar sua tarefa desejada. Na etapa de aprendizagem, é possível testar alguns algoritmos e ver como eles se comportam. Há uma grande variedade de métricas que podem ser usadas para medir o desempenho de um modelo de aprendizado de máquina (Ex: Acurácia – proporção de previsões corretas sobre o total de previsões possíveis).

Um bom modelo é aquele que foi bem treinado ao ponto de conseguir ser generalizado para dados desconhecidos. Dito isso, se o modelo criado a partir dos dados de treinamento for utilizado nos dados de teste e resultar em uma boa acurácia, podemos dizer que ele encontrou um padrão geral nos dados e faz uma boa generalização; se tiver uma baixa acurácia, podemos dizer que ele identificou um padrão muito específico dos dados de treino, mas não faz uma boa generalização.

Ao final do treinamento, teremos um modelo final criado com base nos padrões estatísticos extraídos do conjunto de dados de treinamento. A etapa seguinte busca realizar previsões de rótulos (categorias) baseado nos dados de teste, que são desconhecidos pelo algoritmo. Como nós sabemos qual é o resultado esperado de categoria, podemos avaliar se o modelo criado acertou a categoria e descobrir se ele permite fazer uma boa generalização dos dados.

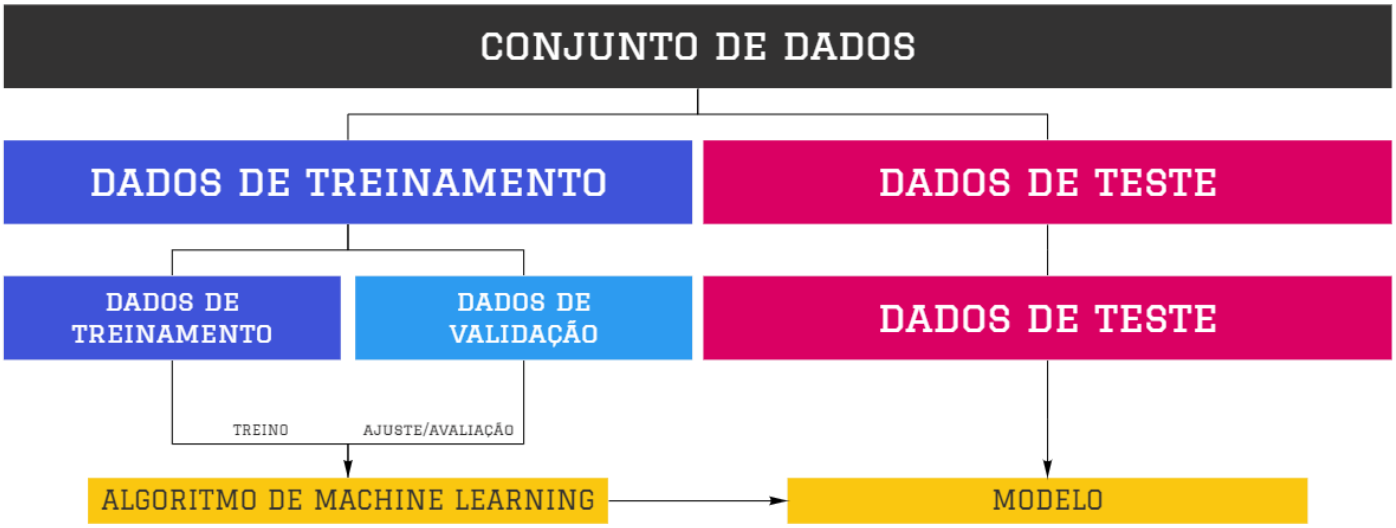
Professor, por que fazemos a avaliação baseado no conjunto de dados de teste e, não, no conjunto de dados de treinamento? Ora, porque o modelo foi criado com base nos dados de treinamento, logo ele certamente teria um bom desempenho com os dados de treinamento. Nós somente podemos avaliar se o modelo criado tem realmente um bom desempenho se fizermos testes com dados que ele nunca teve contato, como os dados de teste.

Agora vejam só: eu falei para vocês que o conjunto de dados seria dividido de forma aleatória em duas partes. Ocorre que essa divisão não é tão simples assim! Em primeiro lugar, é importante que essa divisão seja aleatória, isto é, não escolher individualmente os dados que farão parte de um conjunto ou de outro. Em segundo lugar, é importante que não existam dados repetidos nos dois conjuntos – lembrem-se que os dados de teste devem ser inéditos e desconhecidos para o modelo.

Agora imagine que eu passei um bom tempo treinando meu algoritmo para criar um modelo e, ao utilizar os dados de teste, descubro que o modelo está com uma baixa acurácia. Poxa, agora nós

gastamos todos os nossos dados de treinamento, gastamos todos os nossos dados de teste e só descobrimos ao final que o modelo não estava adequado. *O que fazer, Diego?* É possível usar uma estratégia que avalia o desempenho do modelo enquanto ocorre o processo de treinamento.

Essa estratégia divide os dados em três conjuntos: treinamento, validação e teste. Por meio dela, é possível avaliar o desempenho de vários modelos distintos ou até mesmo avaliar o desempenho dos modelos com a otimização de hiperparâmetros diferentes (veremos mais à frente). A ideia é fazer diversos ajustes de modelagem ainda em tempo de treinamento que permitam verificar qual modelo (e com que parâmetros) generaliza melhor os dados.



O final do processo continua o mesmo: o modelo final é executado com os dados de teste (que permanecem desconhecidos por parte do modelo) e seu desempenho é calculado por meio da métrica de acurácia. Ocorre que temos um problema: nem sempre temos dados suficientes para treinar e testar o modelo. Há casos em que temos uma infinidade de dados, logo é até irrelevante como será a divisão dos dados para cada um dos conjuntos.

Em outros casos, a quantidade de dados é pequena! *E aí, como dividimos entre os conjuntos?* Se colocarmos muitos dados para o conjunto de treinamento, teremos poucos dados para o conjunto de testes – isso significa que teremos provavelmente um modelo que faz uma generalização dos dados de boa qualidade, mas não teremos como saber porque nossa estimativa de desempenho tem pouca confiabilidade (baixa precisão preditiva). Temos também o caso inverso...

Se colocarmos muitos dados para o conjunto de testes, teremos poucos dados para o conjunto de treinamento, logo a estimativa de desempenho do modelo será confiável (alta precisão preditiva), mas ele fará uma generalização de má qualidade. É como um cobertor curto: se cobrir as pernas, a cabeça fica de fora; e se cobrir a cabeça, as pernas ficam de fora. Dito tudo isso, agora precisamos falar sobre validação cruzada.

Validação Cruzada

VALIDAÇÃO CRUZADA

Trata-se de uma técnica usada para avaliar modelos de aprendizado de máquina. É um processo de dividir um conjunto de dados em várias partes, treinando o modelo em uma parte e testando-o em outra parte. Esse processo é repetido várias vezes, com cada parte usada como um conjunto de teste uma vez. Isso permite que o modelo seja testado em vários conjuntos de dados e fornece uma medida mais precisa de seu desempenho.

A validação cruzada é uma técnica para avaliar a capacidade de generalização de um modelo, a partir de um conjunto de dados – é empregada quando se deseja realizar predições. Busca-se estimar o quão preciso é um modelo na prática, ou seja, o seu desempenho em um novo conjunto de dados. Idealmente, o modelo deve ser avaliado em amostras que não foram usadas para construir ou ajustar o modelo, de modo que forneçam um senso imparcial de eficácia do modelo.

A Validação Cruzada pode ocorrer de forma exaustiva ou não-exaustiva. Vejamos os detalhes de cada uma dessas formas a seguir:

- **Forma Exaustiva:** essa técnica basicamente envolve testar o modelo de todas as formas possíveis – dividem-se os dados originais definidos em conjuntos de treinamento e teste (Ex: Leave-P-Out – LpO e Leave-One-Out – LoO).

Para provas de concurso, a forma não-exaustiva é mais cobrada, portanto – nesse momento – basta saber que o LpO é uma abordagem que deixa de fora *p* observações ao fazer a validação cruzada, isto é, havendo *n* observações, haverá *n-p* amostras para treinamento e *p* observações são usadas pela validação. Já o LoO é um caso particular do LpO em que $p = 1$. Nesse caso, apenas uma observação é retirada do conjunto de treinamento ao fazer a validação cruzada.

- **Forma Não-Exaustiva:** essa técnica também divide os dados originais definidos em conjuntos de treinamento e teste, mas isso não ocorre analisando todas as permutações e combinações possíveis (Ex: Holdout e K-Fold).

O Método Holdout consiste basicamente em dividir o conjunto de dados em subconjuntos mutuamente exclusivos (disjuntos): um para treinamento e outro para testes. O subconjunto de treinamento pode ser subdividido também para validação (otimização de parâmetros). O problema desse método é que não é certo que o conjunto de dados de validação seja representativo da amostra, além de ter pouca eficiência quando se tem uma amostra poucos dados.

Um regime de hold-out muito comum é o 50/30/20. *Como assim, Diego?* É simplesmente a divisão do conjunto de dados em 50% para treinamento, 30% para avaliação e 20% para testes.

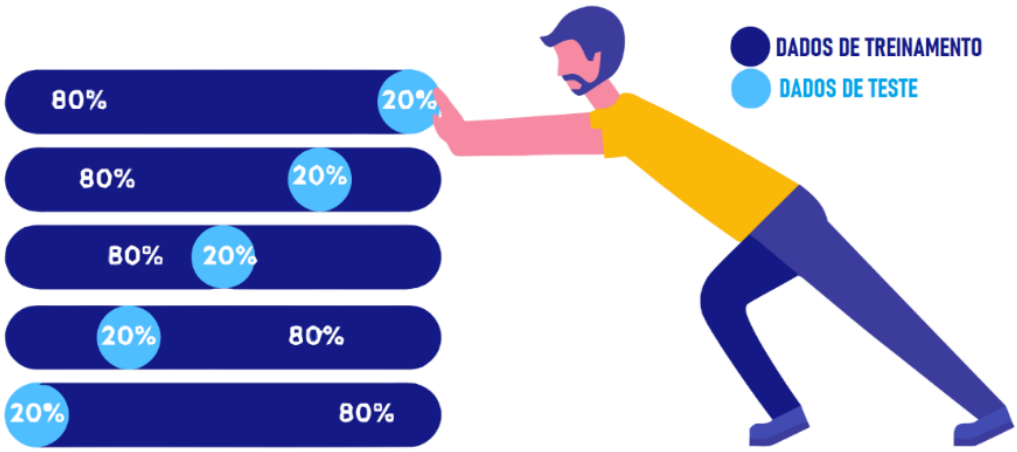
MÉTODO HOLDOUT

Trata-se de um método de validação de modelos preditivos usada para avaliar o desempenho de um modelo de aprendizado de máquina. Isso é feito dividindo os dados em um conjunto de treinamento, que é usado para treinar o modelo, e um conjunto de teste, que é usado para avaliar o modelo (o subconjunto de treinamento pode ser subdividido em um conjunto para validação, isto é, otimização/ajuste de parâmetros). O método busca fazer uma estimativa imparcial do desempenho do modelo.

Uma solução para esse problema é utilizar o Método *K-Fold*. A ideia aqui é simples: *sabe quando mostramos a nossa primeira estratégia, que dividia o conjunto de dados em dados de treinamento e de teste?* Pois é, a validação cruzada fará exatamente a mesma coisa, mas diversas vezes. No exemplo abaixo, nós fizemos cinco vezes essa divisão entre dados de treino e dados de teste. No entanto, observe que cada uma das partições ou divisões é diferente uma da outra.

MÉTODO K-FOLD

Trata-se de um método de validação de modelos preditivos usado para avaliar o desempenho de um modelo de aprendizado de máquina - além de avaliar a sua robustez. Ele envolve a divisão de um conjunto de dados em k subconjuntos de tamanhos iguais. Um subconjunto é usado para testar o modelo, enquanto os outros subconjuntos k-1 são usados para treinar o modelo. O modelo é então testado k vezes, cada vez usando um subconjunto de teste diferente. A média dos resultados é usada para medir a precisão/desempenho do modelo.



Pensem que esses dados estão dispostos em uma tabela com diversas linhas. A primeira partição pegou as 80% primeiras linhas da tabela para treino e o restante para teste; a segunda pegou as 60% primeiras linhas para treino, as 20% seguintes para teste e o restante também para treino; a terceira pegou as 40% primeiras linhas para treino, as 20% seguintes para teste e o restante para treino; e assim por diante...

Dessa forma, conseguimos fazer cinco modelos diferentes e independentes de forma que cada modelo será treinado e avaliado com um conjunto (particionamento) diferente dos dados. Essa

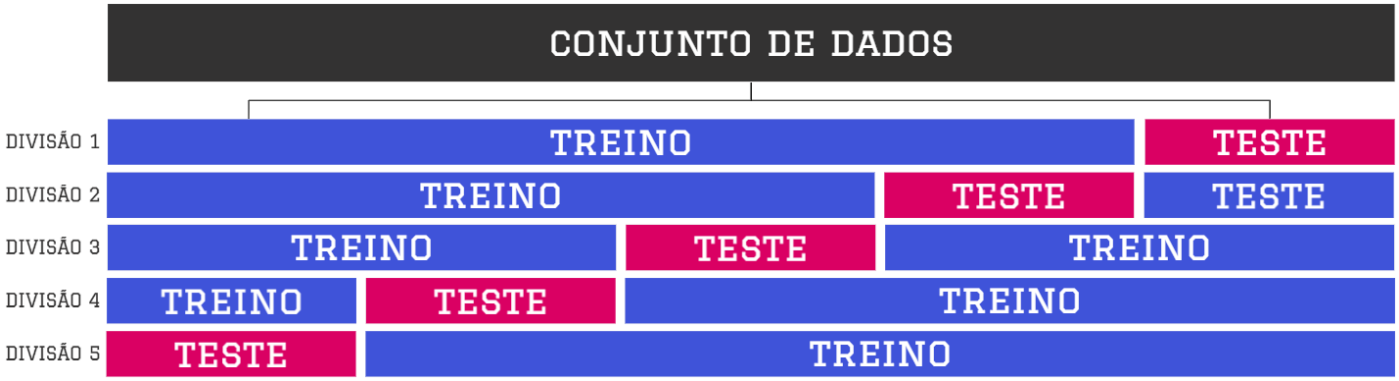
ideia foi genial! Logo, temos agora cinco modelos e cinco métricas de desempenho diferentes. Pronto! Agora podemos fazer uma avaliação do meu modelo preditivo como uma média dos cinco desempenhos diferentes.

Nós não precisamos mais decidir se vamos atribuir mais dados para o treinamento ou mais dados para o teste. Basta fazer algumas partições, pegar a média e teremos uma avaliação mais satisfatória do desempenho do meu modelo preditivo. Não existe uma partição ideal entre treino e teste, mas é comum a divisão em 80% Treino e 20% Teste. Bem, a única desvantagem é que há um custo computacional em fazer esses testes diversas vezes.

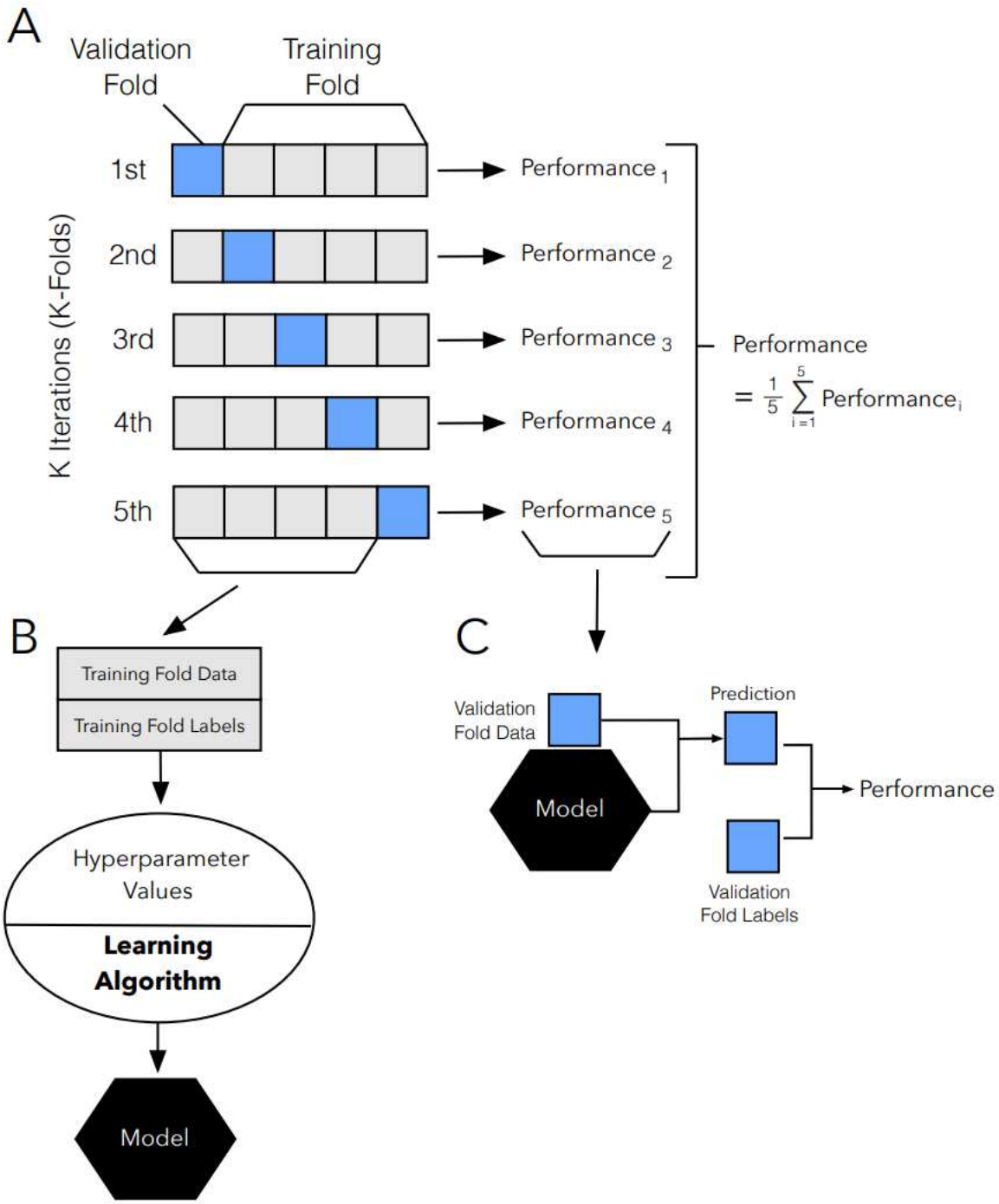
Em suma: o método de validação cruzada denominado k-fold consiste em dividir o conjunto total de dados em k subconjuntos mutuamente exclusivos do mesmo tamanho e, a partir daí, um subconjunto é utilizado para teste e os k-1 restantes são utilizados para treinamento do modelo de predição. Este processo é realizado k vezes sendo que, ao final das k iterações, calcula-se a acurácia sobre os erros encontrados.

O resultado desse método é uma medida mais confiável sobre a capacidade de um modelo preditivo de generalizar os dados a partir de um conjunto de dados de treinamento. O esquema a seguir representa bem como funciona a validação cruzada: a Parte A mostra as diversas iterações de divisões em dados de treino e dados de teste; a Parte B mostra o aprendizado de máquina pelos dados de treinamento; e a Parte C mostra a avaliação do desempenho pelos dados de teste.

Note que a performance é calculada como 1/5 do somatório das cinco performances (= média das performances). Em outras palavras, trata-se de da média aritmética das performances. Note que a diferença fundamental entre o Método Holdout e o Método K-Fold é que o segundo faz diversas divisões diferentes dos dados para calcular o desempenho. Logo, cada subconjunto será utilizado para teste em algum momento...



Por fim, também é importante destacar que o Método K-Fold é geralmente mais adequado quando temos um conjunto de dados de tamanho limitado/pequeno e também pode ser utilizado para identificar *overfitting* (veremos adiante). Já o Holdout é mais adequado quando o conjunto de dados tem um tamanho maior e também pode ser utilizado para fornecer uma estimativa de quão bem um modelo generaliza para dados não vistos. Vejamos um esquema final do Método K-Fold:



Curva ROC

CURVA ROC (RECEIVER OPERATING CHARACTERISTIC)

Trata-se de uma representação gráfica do desempenho de um sistema de classificação binária à medida que seu limite de discriminação é variado. É um gráfico da taxa de verdadeiros-positivos em relação à taxa de falsos-positivos para um determinado limite de probabilidade. Ele exibe a capacidade de um modelo de distinguir entre uma classe positiva e uma classe negativa.

Vamos pensar no contexto da pandemia de coronavírus! Imagine que no início da pandemia, um médico italiano era responsável por avaliar se um paciente estava infectado com o vírus ou não. Ainda não havia testes rápidos e a demanda por exames era absurdamente alta, então ele tinha que realizar o diagnóstico apenas por meio de alguns sintomas (Ex: uma pontuação positiva ou negativa para indicadores como febre, falta de ar, tosse, espirros, dores no corpo, perda de paladar, etc).

Ocorre que, em uma situação emergencial como a que vivemos, se tivermos um falso-negativo (ou seja, uma pessoa estava infectada com o vírus, mas não foi diagnosticada), a consequência é gravíssima porque ela pode retornar para o seu convívio social e infectar diversas outras pessoas. Nesse sentido, podemos afirmar que o sucesso do médico italiano pode ser medido de duas formas: pela taxa de verdadeiros-positivos e pela taxa de falsos-positivos.

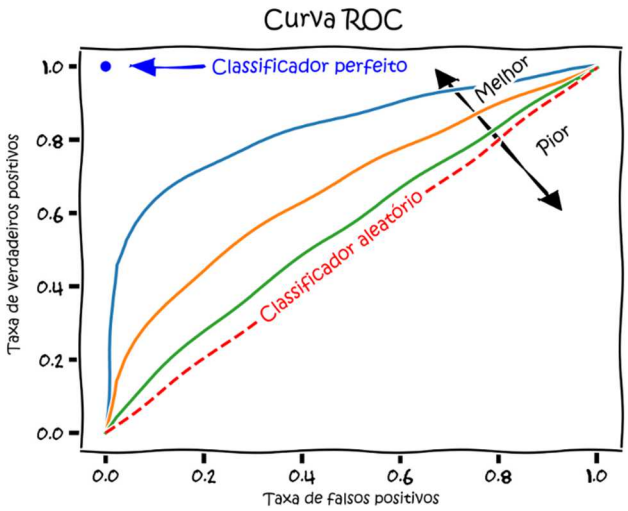
Os verdadeiros-positivos seriam os pacientes diagnosticados com o vírus e que realmente estavam infectados pelo vírus; já os falsos-positivos seriam os pacientes diagnosticados com os vírus e que, na verdade, estavam apenas com uma gripe ou outro vírus qualquer. Dito isso, o médico italiano poderia criar um modelo que prevê se um paciente com um conjunto de sintomas está ou não infectado com o coronavírus.

Em seguida, ele pode testar o modelo criado com um conjunto de dados de treino em que já se conheça de antemão se o paciente está ou não infectado. Ao fazer isso, o modelo devolverá um valor para cada paciente que indicará a probabilidade de esse paciente estar infectado com coronavírus (Ex: Paciente 1 tem 12% de probabilidade de estar infectado, enquanto o Objeto 2 tem 89% de probabilidade de não estar infectado).

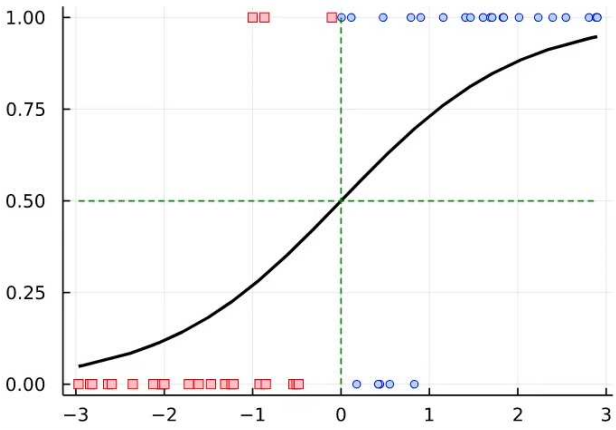
A partir daí, o médico italiano poderá definir um limiar (ou ponto de corte): por exemplo, se o modelo que ele treinou retornar uma probabilidade maior ou igual a 50%, ele vai considerar que o paciente está infectado; já se retornar uma probabilidade menor ou igual a 50%, ele vai considerar que o paciente não está infectado. Só que a escolha desse é importantíssima, porque estamos falando de contaminação em massa e potenciais vítimas fatais. *Como assim, Diego?*

Galera, se o limiar for 50%, pode ser que ocorram muitos falsos-negativo, isto é, o médico achou que o paciente não estava infectado, mas infelizmente ele estava. *Resultado?* Ele mandou o paciente para casa, que acabou infectando diversas outras pessoas e ocasionando mortes. Uma alternativa seria reduzir o limiar, maximizando a cobertura dos falsos-negativos e aumentando os falsos-positivos. *Professor, o risco de um falso-positivo não é pior?*

Não! Um falso-positivo indica que o médico diagnosticou o paciente como infectado, mas ele não estava – o impacto disso é muito menor, dado que o paciente apenas ficará isolado em observação. *Bacana, mas e qual é o limiar ideal? 20%? 40%? 25%? 35%? 1%?* É possível criar uma matriz de confusão para cada um desses limiares para tentar descobrir o limiar ótimo, mas há uma ferramenta melhor: Curva ROC!



O que ela faz, Diego? Ela basicamente fornece uma maneira simples de resumir matrizes de confusão de acordo com cada valor possível para o limiar. Vejamos alguns gráficos...

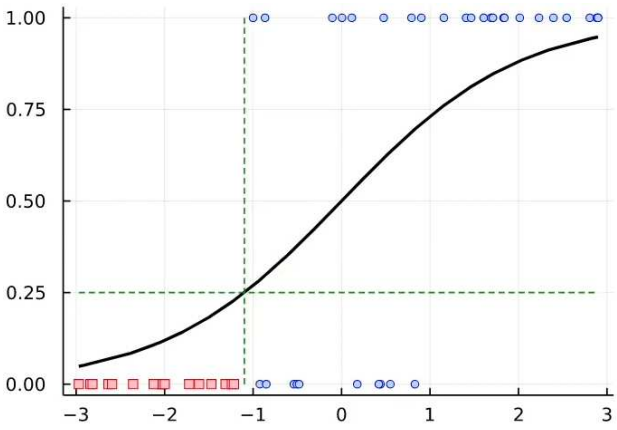


O gráfico acima exibe a curva de uma regressão logística com uma função sigmoide em que o eixo X representa, por exemplo, a pontuação para um conjunto de sintomas de um paciente e o eixo Y representa, por exemplo, a probabilidade de esse paciente estar infectado ou não com o coronavírus. Note que as bolinhas azuis acima da curva representam os verdadeiros-positivos (Ex: o médico previu que o paciente estava infectado com o coronavírus e ele realmente estava).

As bolinhas azuis abaixo da curva representam os falsos-positivo (Ex: o médico previu que o paciente estava infectado, mas ele não estava). Já os quadradinhos vermelhos acima da curva representam os falsos-negativos (Ex: o médico previu que o paciente não estava infectado, mas ele estava). Por fim, os quadradinhos vermelhos abaixo da curva representam os verdadeiros-negativos (Ex: o médico previu que o paciente não estava infectado e ele realmente não estava).

Percebam pelo gráfico que o limiar está em 50% (0,5), logo qualquer valor igual ou maior que 0,5 será previsto como infectado e qualquer valor menor que 0,5 será previsto como não infectado. *Considerando que estamos tratando de uma questão de vida ou morte, esse limiar foi bem escolhido?* Bem, vejam que tivemos três ocasiões em que o modelo previu que o paciente não estava infectado, mas que infelizmente ele estava.

Note também que tivemos cinco ocasiões (não dá para ver muito bem, mas tem uma bolinha azul colada na outra) em que o modelo previu que o paciente estava infectado, mas que ele não estava, no entanto, esse caso é menos grave nesse contexto. Dito isso, podemos concluir que é mais importante minimizar os falsos-negativos, mesmo que ocorra um aumento substancial de falsos-positivos. Para tal, podemos ajustar o limiar conforme o gráfico seguinte:



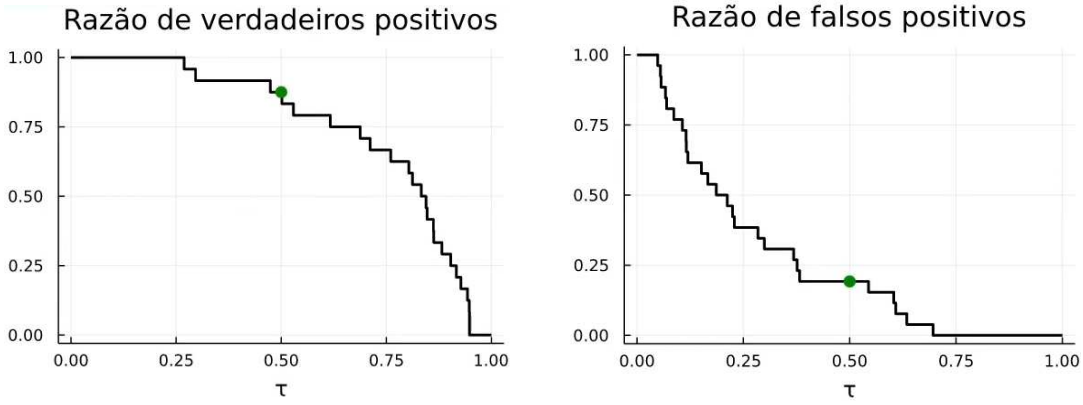
Percebam que agora o limiar agora está em 0,25 e que não temos mais nenhum falso-negativo, mesmo que tenham aumentado os falsos-positivos (alarme falso) e que tenham reduzido os verdadeiros-negativos. No gráfico anterior, o modelo acerta em todos os casos positivos, apesar de errar mais em casos negativos. Bem, nós vimos em tópicos anterior os conceitos de sensibilidade (razão de verdadeiros positivos) e especificidade (razão de falsos negativos).

Vamos relembrar rapidamente a diferença entre sensibilidade e especificidade. Quando uma mulher está com sua menstruação atrasada e decide por um teste de farmácia, o que estamos buscando é a presença de GCH (Gonadotrofina Coriônica Humana), e simplesmente sua presença. Em outras palavras, os testes de farmácia estão calibrados para acusar qualquer traço de GCH na urina da paciente.

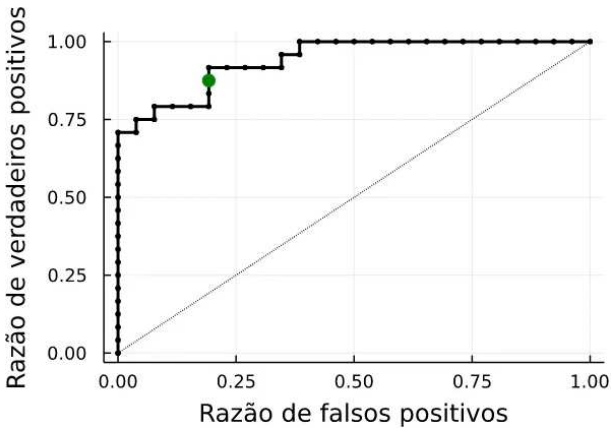
Nesse momento, não importa se iremos quantificar ou não, se queremos saber semanas etc., queremos saber se tem ou se não tem GCH na urina, para que depois façamos uma análise mais criteriosa a partir desse primeiro exame. Assim, o teste de farmácia para gravidez é um teste com alta sensibilidade. Indo na contramão da sensibilidade, temos a especificidade. Logo, imaginemos uma paciente que acusou positivo para GCH na urina.

Precisamos de um exame que seja mais específico para identificar essa substância no sangue da paciente, que não acuse os falsos-positivos e seja determinante para o diagnóstico. Nessa hora, não basta saber se tem ou se não tem GCH, queremos saber definitivamente se a paciente está ou não grávida. Nessa hora, entra o exame de sangue, que é mais específico para o objetivo: identificar uma possível gravidez.

Baseado nesses conceitos e no primeiro gráfico que vimos (com limiar em 0,5), podemos criar mais dois gráficos: Sensibilidade x Limiar e (1 – Especificidade) x Limiar.

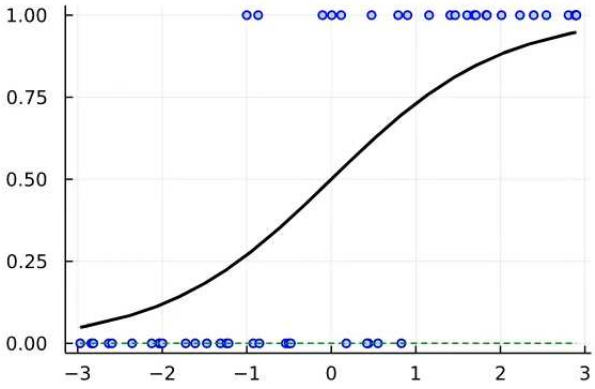


Não tem nada complexo nesses gráficos: apenas fizemos uma matriz de confusão para cada valor de limiar (0 a 1) e plotamos a **Sensibilidade** (razão de verdadeiros positivos) e **1 – Especificidade** (razão de falsos positivos) em função de cada limiar. Já a Curva ROC é a curva formada em um gráfico de sensibilidade por 1 – especificidade (conforme apresentado no gráfico a seguir). Ela permite avaliar a variação de sensibilidade/especificidade à medida que se modifica o limiar.

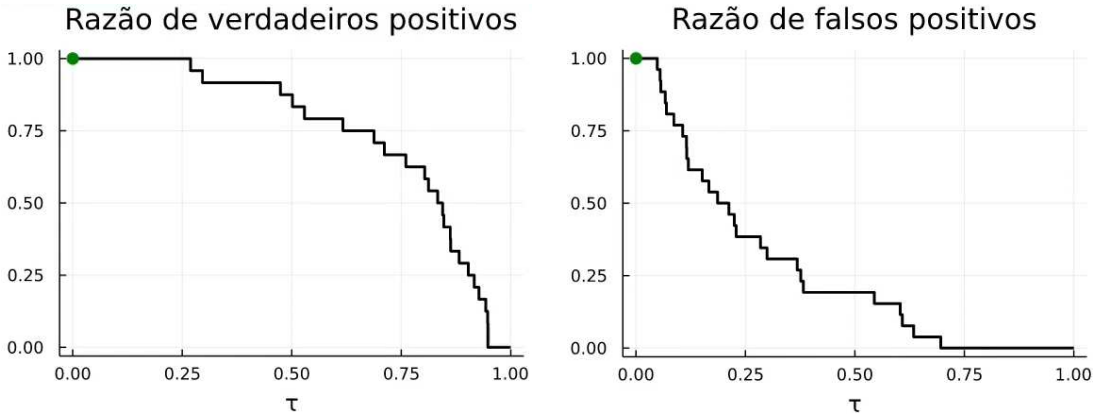


À medida que se escolhe um limiar com maior sensibilidade, necessariamente, se penitencia a classificação diagnóstica por sua menor especificidade e vice-versa. Em outras palavras, quanto mais a curva se aproxima do canto superior esquerdo, melhor é a qualidade do teste quanto à capacidade para discriminar os grupos. E a linha de referência diagonal apresentada no gráfico representa uma região de completa aleatoriedade do teste em que sensibilidade = especificidade.

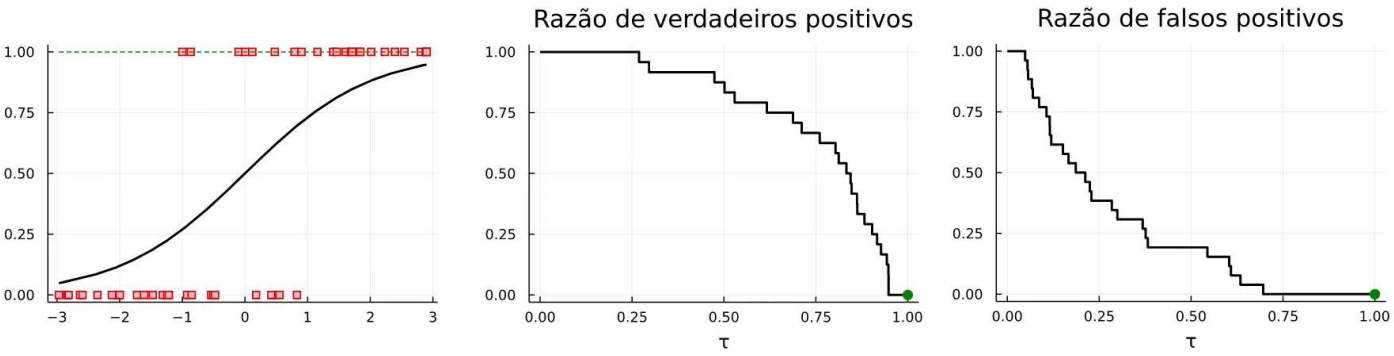
Agora se vocês entenderam mesmo, vão assimilar rapidinho a interpretação de alguns gráficos! *O que acontece se o médico definir que o limiar é 0,00 (0%)?* Vejamos:



Note que com um limiar zerado, diagnosticamos todo mundo com coronavírus. Logo, o modelo acerta todos os casos em que o paciente realmente estava infectado – tanto os verdadeiros-positivos quanto os falsos-positivos (representado nos gráficos seguintes). Por outro lado, o modelo erra todos os casos em que o paciente não estava infectado – tanto os verdadeiros negativos quanto os falsos-negativos.

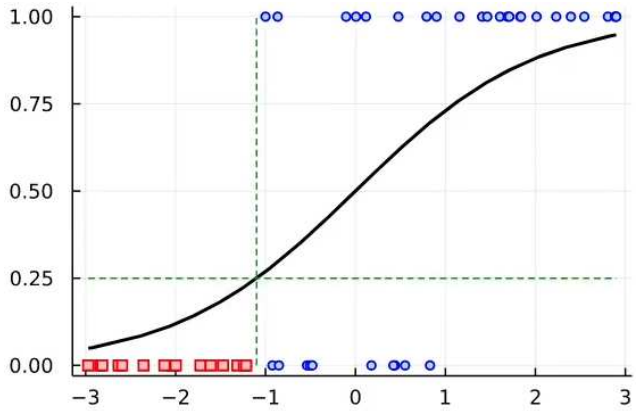


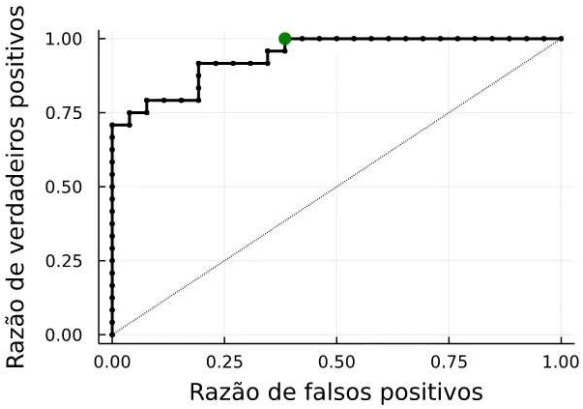
Se o médico decidir que o limiar será 1,00 (100%), ocorre o exato oposto – conforme podemos ver nos gráficos seguintes:



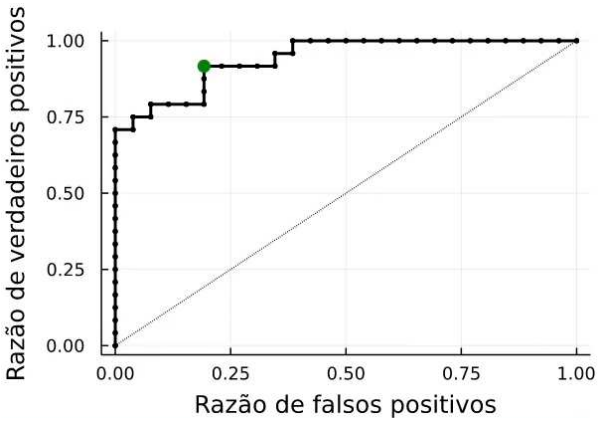
Legal, mas uma maneira melhor de visualizar esses dados é por meio da Curva ROC, que resume os dois gráficos de razão em um só. *Por que essa curva é importante?* Porque ela permite verificar rapidamente qual é o valor de limiar que maximiza os verdadeiros-positivos ao mesmo tempo que

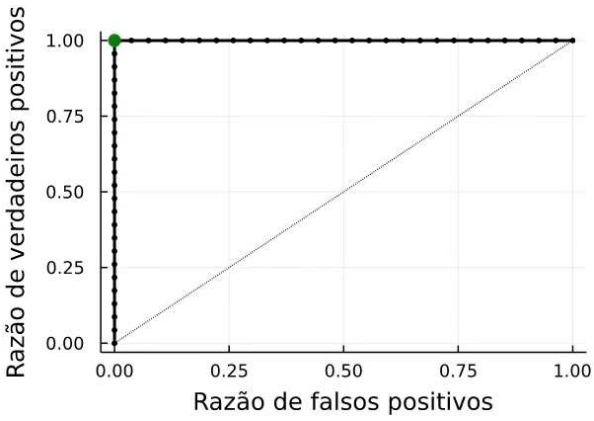
minimiza os falsos-negativos. Vejam no gráfico à esquerda que – com a escolha do limiar em 0,25 – não temos nenhum falso-negativo, apesar de termos mais falsos-positivos.





Já a curva ROC nos mostra que, a partir do ponto verde, acertamos todos os verdadeiros-positivos e uma boa parte dos falsos-positivos. *E qual é o limiar ideal?* É aquele ponto que está mais longe do classificador aleatório representado pela linha diagonal. No gráfico à esquerda, temos um limiar ótimo para o contexto, dado que o ponto verde é o ponto da curva que está mais distante da linha diagonal; já no gráfico à direita, temos um modelo de acurácia perfeita (o que não é muito realista).





Por fim, é importante falar sobre AUC (*Area Under the Curve*). Trata-se do termo usado para descrever a área sob a curva ROC, sendo usada no aprendizado de máquina para medir a capacidade de um modelo de discriminar duas classes e também para medir o desempenho de uma variedade de outros algoritmos. É bem simples: se a área sob a curva de um algoritmo for maior que a área sob a curva de outro algoritmo, significa que ele possui um desempenho melhor.

Avaliação de Regressão

Quando lidamos com modelos de classificação, as métricas fazem comparações se a classes foram corretamente previstas ou não. Ao utilizarmos a regressão, isto fica inviável, pois estamos lidando com valores numéricos potencialmente infinitos. Logo, a principal abordagem das métricas de regressão baseia-se na diferença entre o valor real e o previsto, no qual e representa o desvio, y representa o valor real e $\hat{y}$ é o valor preditos.

$$e = y - \hat{y}$$

Nos próximos parágrafos, vamos analisar as principais métricas de desempenho de uma regressão (sim, existem diversas formas de calcular o desempenho):

Erro Médio Absoluto (EMA)

Trata-se da medida da média dos erros em um conjunto de previsões, sem considerar sua direção. É calculado como a média das diferenças absolutas entre os valores previstos e os valores reais. O MAE (*Mean Absolute Error*) é uma medida da precisão do modelo e ajuda a identificar o viés e a variância. Quanto menor seu valor, mais preciso é o modelo. Ele é frequentemente usado para comparar o desempenho entre modelos diferentes.

$$MAE\,(y, \hat{y}) = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i|$$

Note que é utilizado o operador módulo | | para capturar apenas a diferença positiva entre valores reais e valores preditos.

Erro Quadrático Médio (EQM)

Trata-se de uma medida da média dos quadrados dos erros em um conjunto de previsões. É calculado como a média das diferenças quadradas entre os valores previstos e os valores reais. O MSE (*Mean Squared Error*) não utiliza o modulo para capturar a diferença positiva entre valores reais e preditos e, sim, o quadrado (por isso se chama erro **quadrático** médio). Dessa forma, ele penaliza diferenças fora do normal (*outliers*). *Como assim, Diego?*

$$MSE\,(y, \hat{y}) = \frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

Imagine que, no contexto do nosso problema, o valor real de uma observação seja 5 e o valor predito pelo modelo de regressão seja 35. Dito isso, temos que:

- **MAE (*Mean Absolute Error*):** `|5 - 35| = |-30| = +30;`
- **MSE (*Mean Squared Error*):** `(5 - 35)² = (-30)² = +900.`

Viram como essa métrica penaliza muito mais valores com desvio alto do que a métrica anterior? Pois é! Quanto maior é o valor de MSE, significa que o modelo não performou bem em relação as previsões. Ocorre que essa métrica tem um problema grave: por haver a elevação ao quadrado, a unidade de medida fica distorcida, isto é, se a unidade de medida for em metros (m), o resultado será em m². Isso dificulta a interpretação se o erro foi grande ou pequeno!

Raiz do Erro Quadrático Médio (REQM)

Trata-se de uma medida da raiz da média dos quadrados dos erros em um conjunto de previsões. diferentes modelos. O RMSE (*Root Mean Squared Error*) é basicamente o mesmo cálculo do MSE, porém aplicada a raiz quadrática para lidar com o problema de interpretabilidade da diferença entre unidades dado que a unidade do desvio fica com a mesma escala da unidade original. Por outro lado, ela não penaliza com tanta veemência os valores outliers.

$$RMSE\,(y, \hat{y}) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Coeficiente de Determinação (R^2)

Também chamado de Coeficiente de Determinação, trata-se da proporção da variabilidade dos dados que é explicado pelo modelo. Em outras palavras, é a medida de quão bem um modelo de regressão se ajusta os dados. É calculado como a proporção da variância na variável dependente que é explicada pelo modelo. Quanto maior o valor de R^2, melhor o modelo se ajusta aos dados – os valores de R^2 variam de 0 a 1, com 1 indicando um ajuste perfeito.

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

Por exemplo: um R^2 = 0,8234 significa que o modelo linear explica 82,34% da variância da variável dependente a partir do regressores (variáveis independentes) incluídas naquele modelo.

- R^2 = 0% indica que o modelo não explica nada da variabilidade dos dados de resposta ao redor de sua média.
- R^2 = 100% indica que o modelo explica toda a variabilidade dos dados de resposta ao redor de sua média.

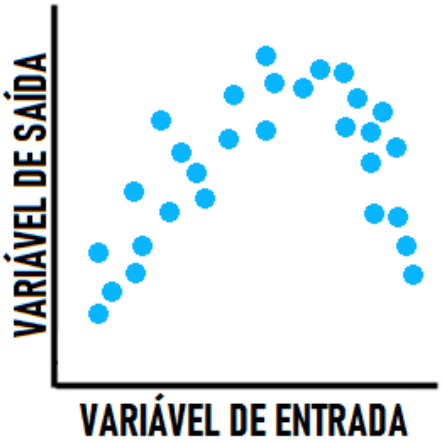
Underfitting e Overfitting

RELEVÂNCIA EM PROVA: ALTA

UNDERFITTING E OVERFITTING

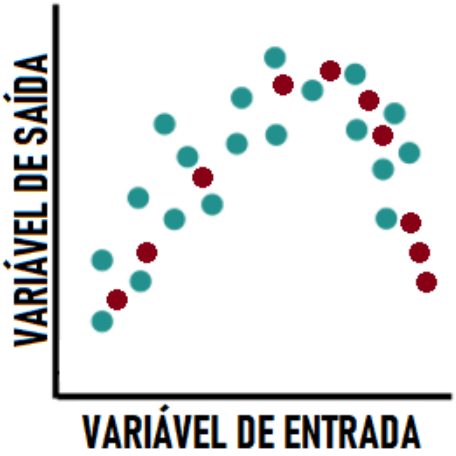
Underfitting ocorre quando um modelo de aprendizado de máquina não captura adequadamente o padrão subjacente dos dados. Trata-se de uma falha na captura da variação nos dados, resultando em uma representação imprecisa desses. Já o Overfitting ocorre quando um modelo de aprendizado de máquina captura muita variação nos dados, resultando em um modelo que não generaliza bem e é excessivamente sensível a pequenas alterações nos dados.

Vamos fazer alguns experimentos? Eu vou precisar bastante da atenção de vocês agora! Vejam esse conjunto de pontos de dados azuis apresentado a seguir em um plano cartesiano: no eixo das abscissas, nós temos variáveis de entradas; e no eixo das ordenadas, nós temos as variáveis de saída. Dado um valor de variável de entrada, temos um valor de variável de saída respectivo. *E o que isso tem a ver com aprendizado de máquina, professor?*



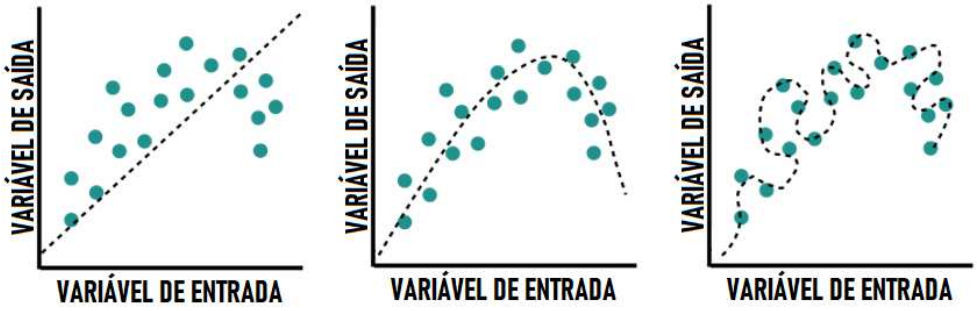
A ideia por trás de um algoritmo de aprendizado de máquina é criar um modelo – representado no plano cartesiano como uma curva – que melhor se ajuste aos dados. Essa curva tem que ser de tal forma que generalize os dados de treino e minimize os erros para dados novos, isto é, se eu inserir um novo dado de entrada, eu gostaria que o meu modelo fizesse uma previsão de dado de saída de forma que tivesse o menor erro (distância) possível em relação à curva.

Agora vocês se lembram que nós vimos a diferença entre dados de treinamento e dados de teste? Pois é, isso será muito útil agora! Primeiro, nós precisamos treinar o nosso algoritmo de forma que ele gere a curva desejada; depois nós precisamos testar se a curva se ajusta aos dados de teste; e só depois ele estará pronto para ser efetivamente usado com novos dados. Dito isso, agora é o momento de dividir esses pontos de dados em pontos de treinamento e pontos de dados de teste:



Pronto! Os pontos verdes são aqueles que serão utilizados para treinar nosso modelo e os pontos vermelhos serão aqueles que serão utilizados para testar o nosso modelo. Agora nós já podemos começar a treinar o nosso modelo, mas faremos isso – evidentemente – utilizando apenas os pontos de dados de teste. Logo, temos que excluir os pontos vermelhos, porque eles não podem ser utilizados para o treinamento do algoritmo. *Lembram?* Vamos deixá-los quietinhos...

Legal! *Agora como nós vamos fazer o treinamento do modelo?* É bastante simples: nós podemos utilizar diferentes modelos de aprendizado de máquina e verificar quais deles geram uma curva que melhor se ajusta aos dados. Podemos ver na imagem seguinte três exemplos de modelos e suas respectivas curvas. O primeiro modelo apresenta uma linha reta; o segundo modelo apresenta uma espécie de parábola; e o terceiro modelo apresenta uma linha bastante sinuosa.

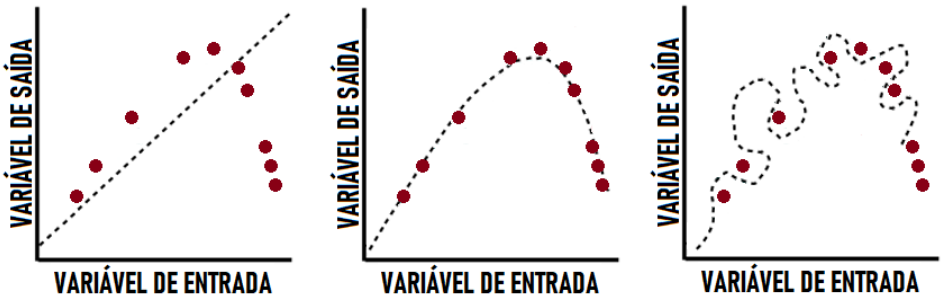


Note que o primeiro modelo apresenta uma linha bem distante da maioria pontos – não está muito ajustada; já o segundo modelo apresenta uma curva um pouco mais próxima aos pontos; por fim, o terceiro modelo apresenta uma curva praticamente perfeita – ele está impecavelmente ajustado aos pontos de dados. Pergunto: qual é o melhor modelo? *Pô, professor... eu sei que a aula é difícil, mas eu não sou tão bobo assim – é óbvio que o terceiro modelo é o melhor!*

RESPOSTA: ERRADA

Coooooooooooooomooooooooo aaaaaaaaaaaaassiiiiiiim, Diego? Galera, os pontos de dados verdes representam nossos dados de... **treino** e quem define qual é o melhor modelo são os dados de... **teste**. *Se liguem porque essa pegadinha pode cair na sua prova!* Dessa forma, a maneira mais interessante de verificar qual é o modelo é plotar as mesmas três curvas, porém com os dados de teste (pontos vermelhos). Vamos relembrar como eles eram...

Opa... agora basta plotar as curvas que descobrimos durante o nosso treinamento, porém aplicadas aos dados de teste. Vejam só como é que fica:



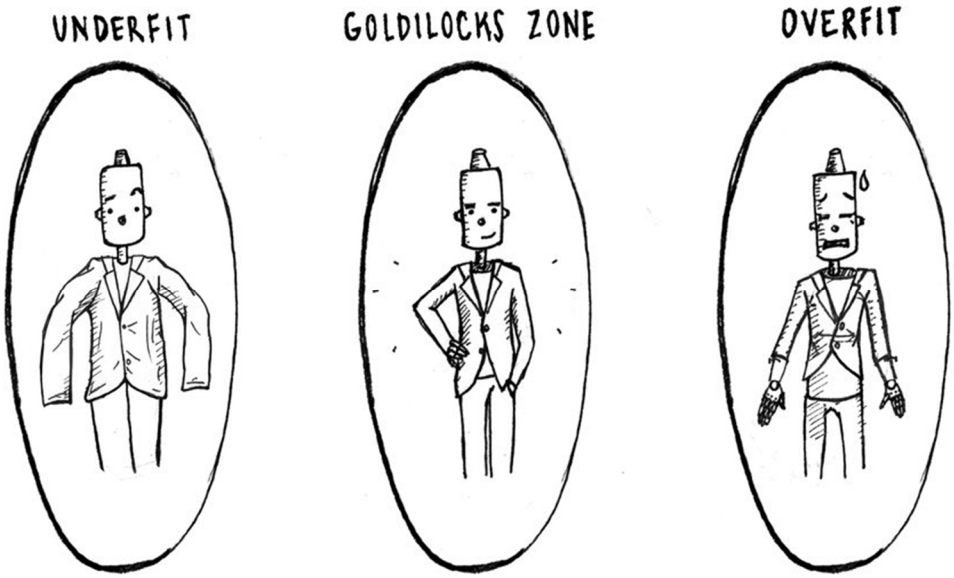
E agora, mudaram de opinião? Note que o modelo representado pela segunda curva é o que melhor generaliza os dados. Na primeira curva, ocorre o que chamamos de Underfitting (Subajuste): a reta não se ajusta bem nem aos dados de treino nem aos dados de teste, logo podemos afirmar que o modelo possui falta de parâmetros, desempenho ruim, pouca expressividade e excessiva simplicidade para modelar a real complexidade do problema para novos dados.

Na terceira curva, ocorre o que chamamos de Overfitting (Sobreajuste): a curva se ajusta muito bem aos dados de treino, mas não se ajusta tão bem aos dados de teste, logo podemos afirmar que o modelo possui parâmetros demais, desempenho ruim, muita expressividade e excessiva complexidade para modelar a real complexidade do problema. O modelo apenas memoriza os dados de treino, logo não é capaz de generalizar para novos dados.

Vocês se lembram do diagrama apresentado acima? Pois é! Dados observados são o resultado de um conjunto de variáveis disponíveis + um conjunto de variáveis indisponíveis + aleatoriedade + erros. Se a nossa curva está acertando perfeitamente todos os pontos do conjunto de dados de treino, significa que não há nenhuma variável indisponível, nenhuma aleatoriedade e nenhum erro. Isso até pode ocorrer, mas somente em casos extremamente raros e excessivamente simples.

Na maioria dos casos, existem variáveis indisponíveis + aleatoriedade + erros. Se o modelo tiver um excesso de complexidade, ele não generalizará bem para novos dados. A essência do *overfiting* é extrair parte da variação residual para a modelagem da curva como se essa variação representasse a estrutura do modelo (e, não, apenas um ruído). Bem, como eu gosto de fazer comparações, vamos ver algumas agora para você **nunca mais esquecer**!

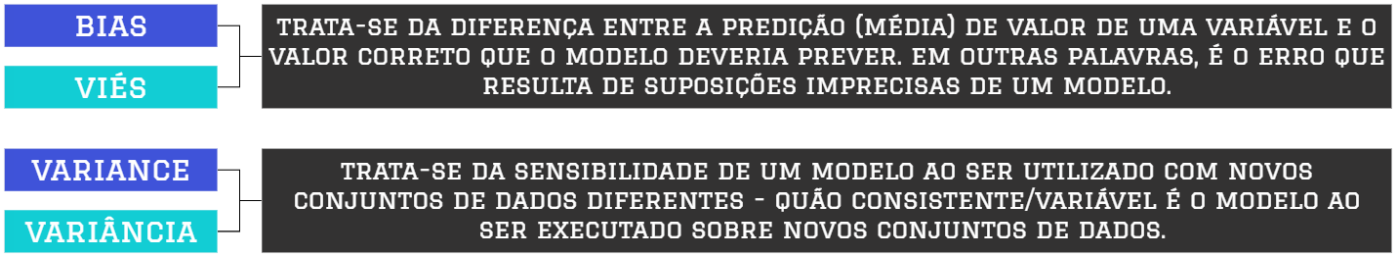
Imagine que você vá ao shopping comprar uma camisa social. Você não quer errar, então pede ao vendedor que traga todos os tamanhos possíveis: P, M, G!



Você veste a camisa G e nota que ela fica completamente folgada, isto é, ela não se ajusta bem ao seu corpo – há sobras para todo lado (*underfitting*); depois você veste a camisa P e parece que ela foi ajustada perfeitamente para você – não há uma sobra sequer (*overfitting*); por fim, você veste a camisa M e fica um modelo intermediário – nem muito ajustada nem muito folgada. *Por que seria ruim comprar a camisa P?*

Porque se você tirar férias e engordar uns 3 kg, a camisa não entra mais; ou se você estiver estudando muito para o seu concurso e emagrecer uns 3 kg, ela fica muito folgada. Logo, o modelo intermediário é que o melhor se ajusta a eventuais novos dados (kg a mais ou a menos). A seguir, há dois exemplos humorísticos clássicos em qualquer aula sobre esse assunto. Se você não os entender, chame no fórum de dúvidas que a gente esclarece ;)

Viés e Variância



Antes de descobrir como resolver esses tipos de problema, vamos discutir dois conceitos importantes: viés e variância. A palavra "viés" é comumente utilizada em nossa língua com o significado de viciado ou fortemente inclinado a respeito de algo. Por exemplo: se um chefe pergunta para um funcionário qual é a opinião dele sobre seu próprio desempenho no trabalho, essa opinião poderá estar enviesada, isto é, conter um vício, uma inclinação, uma parcialidade.

Em outras palavras, a pessoa acha que está tendo um excelente desempenho, mas a realidade é que o desempenho dela está longe do ideal. Aqui é bem parecido: o modelo acha que está fazendo uma boa previsão, mas a realidade é que sua previsão está longe do ideal. Logo, podemos dizer que o viés é o erro que resulta de suposições imprecisas de um modelo. Em termos técnicos, trata-se da diferença entre a predição do valor de uma variável e o valor correto que o modelo deveria prever.

Dessa forma, o viés trata da incapacidade de um modelo de capturar o verdadeiro relacionamento entre variáveis. Dito isso, um modelo de aprendizado de máquina com alto viés é aquele que erra bastante as previsões de valores (isto é, possui baixa acurácia); já um modelo de aprendizado de máquina com baixo viés é aquele que acerta bastante as previsões de valores (isto é, possui alta acurácia). *Entendido?*

Já a variância trata de quanto as previsões de um modelo variam ao serem usados diferentes dados de treino – trata da sensibilidade de um modelo ao ser utilizado com conjuntos de dados de treino diferentes. Dito isso, um modelo com alta variância é aquele cujas previsões sobre diferentes conjuntos de dados variam bastante (baixa generalização); e um modelo com baixa variância é aquele cujas previsões sobre diferentes conjuntos de dados são consistentes (alta generalização).

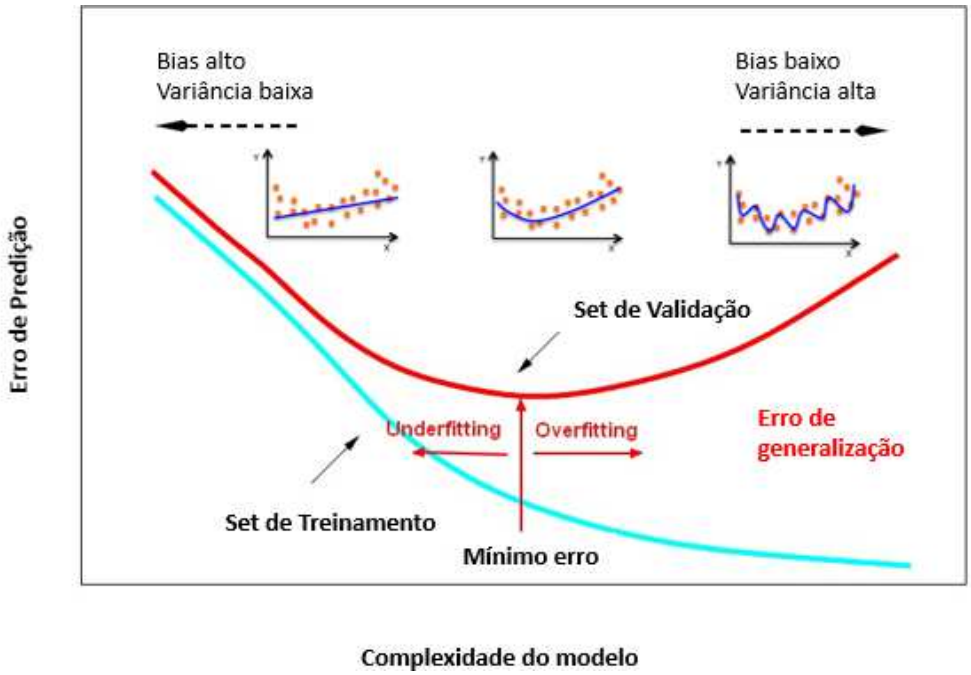
E qual é o modelo ideal? **O modelo ideal é aquele com baixo viés e baixa variância**. Dito isso, vamos tentar interpretar a imagem seguinte:

COMBINAÇÕES	DESCRIÇÃO	REPRESENTAÇÃO
BAIXO VIÉS E BAIXA VARIÂNCIA	Trata-se de um modelo que possui ótima precisão em suas previsões com dados de treino e que varia muito pouco quando aplicado a novos dados. Note que os pontos estão no centro do alvo e não estão espalhados.	
BAIXO VIÉS E ALTA VARIÂNCIA	Trata-se de um modelo que possui boa precisão em suas previsões com dados de treino (*overfitting*), mas que varia bastante quando aplicado a novos dados. Note que os pontos estão próximos ao centro do alvo, porém estão um pouco espalhados.	
ALTO VIÉS E BAIXA VARIÂNCIA	Trata-se de um modelo que possui péssima precisão em suas previsões com dados de treino (*underfitting*), mas que varia pouco quando aplicado a novos dados. Note que os dados estão longe do centro do alvo, porém não estão espalhados.	
ALTO VIÉS E ALTA VARIÂNCIA	Trata-se de um modelo que possui péssima precisão em suas previsões com dados de treino e que varia bastante quando aplicado a novos dados. Note que os pontos estão longe do centro do alvo e também estão bastante espalhados.	

Para termos o modelo mais próximo do ideal, devemos fazer algumas escolhas: se aumentarmos a variância, reduziremos o viés; se reduzirmos a variância, aumentaremos o viés. Logo, a melhor estratégia é encontrar o equilíbrio entre esses dois erros que melhor atenda ao modelo de aprendizado de máquina em treinamento. É importante destacar que modelos lineares geralmente tem alto viés e baixa variância e modelos não lineares geralmente são o contrário.

Dito isso, para resolver problemas de *overfitting*, podemos tomar algumas atitudes: utilizar mais dados de treinamento (com maior variedade); utilizar validação cruzada com vários modelos para detectar o melhor desempenho; realizar a combinação de múltiplos modelos diferentes (também chamado de *ensemble*); limitar a complexidade do modelo por meio de técnicas de regularização; adicionar ruído aleatório nos dados de treinamento para reduzir o ajuste.

Já para resolver problemas de *underfitting*, podemos aumentar a complexidade do modelo; aumentar o tempo de treino; selecionar variáveis; reduzir a regularização.



Para finalizar esse assunto, vamos ver um gráfico muito famoso: Complexidade do Modelo x Erros de Predição. A complexidade de um modelo de aprendizado de máquina é definida em função da quantidade de parâmetros ou variáveis: quanto mais parâmetros, mais complexo; quanto menos parâmetros, menos complexo. Já a quantidade de erros de predição nos indica quão bom é um modelo: quanto menos erros de predição, melhor; quanto mais erros, pior.

Agora observem que temos duas curvas: (1) azul – referente ao conjunto de dados de treinamento; e (2) vermelho – referente ao conjunto de dados de validação. Vamos analisá-las separadamente: a curva azul nos mostra que, ao aplicar o modelo a um conjunto de dados de treinamento, a quantidade de erros diminui à medida que a complexidade do modelo aumenta. Logo, quanto mais complexo é um modelo, menos erros teremos – por isso, trata-se de uma curva descendente.

Já a curva vermelho nos mostra que, ao aplicar o modelo a um conjunto de dados de validação, a quantidade de erros diminui, chega em um vale, e depois aumenta novamente – percebam que a curva tem formato de uma parábola invertida. Esse ponto em que a curva chega em um vale é chamado de ponto de mínimo erro, porque é quando o modelo que foi treinado com dados de treinamento apresenta menos erros de predição quando aplicada aos dados de validação.

Note que, à esquerda, temos uma região com curvas mais próximas, o que nos indica um modelo pouco complexo e apresenta alta taxa de erros de predição tanto nos dados de treinamento quanto nos dados de validação. Já à direita, temos uma região com curvas mais distantes, o que nos indica um modelo complexo e apresenta baixa taxa de erros de predição nos dados de treinamento e alta taxa de erros de predição nos dados de validação.

Ora, quando um modelo vai mal em suas predições quando aplicado a um conjunto de dados de treinamento e também a um conjunto de dados de validação, ocorre underfitting (região à esquerda); quando um modelo vai bem em suas predições quando aplicado a um conjunto de dados treinamento e vai mal quando aplicado a um conjunto de dados de validação, ocorre overfitting (região à direita). Nesse último caso, dizemos que o modelo generaliza mal os dados.

Ensemble

ENSEMBLE

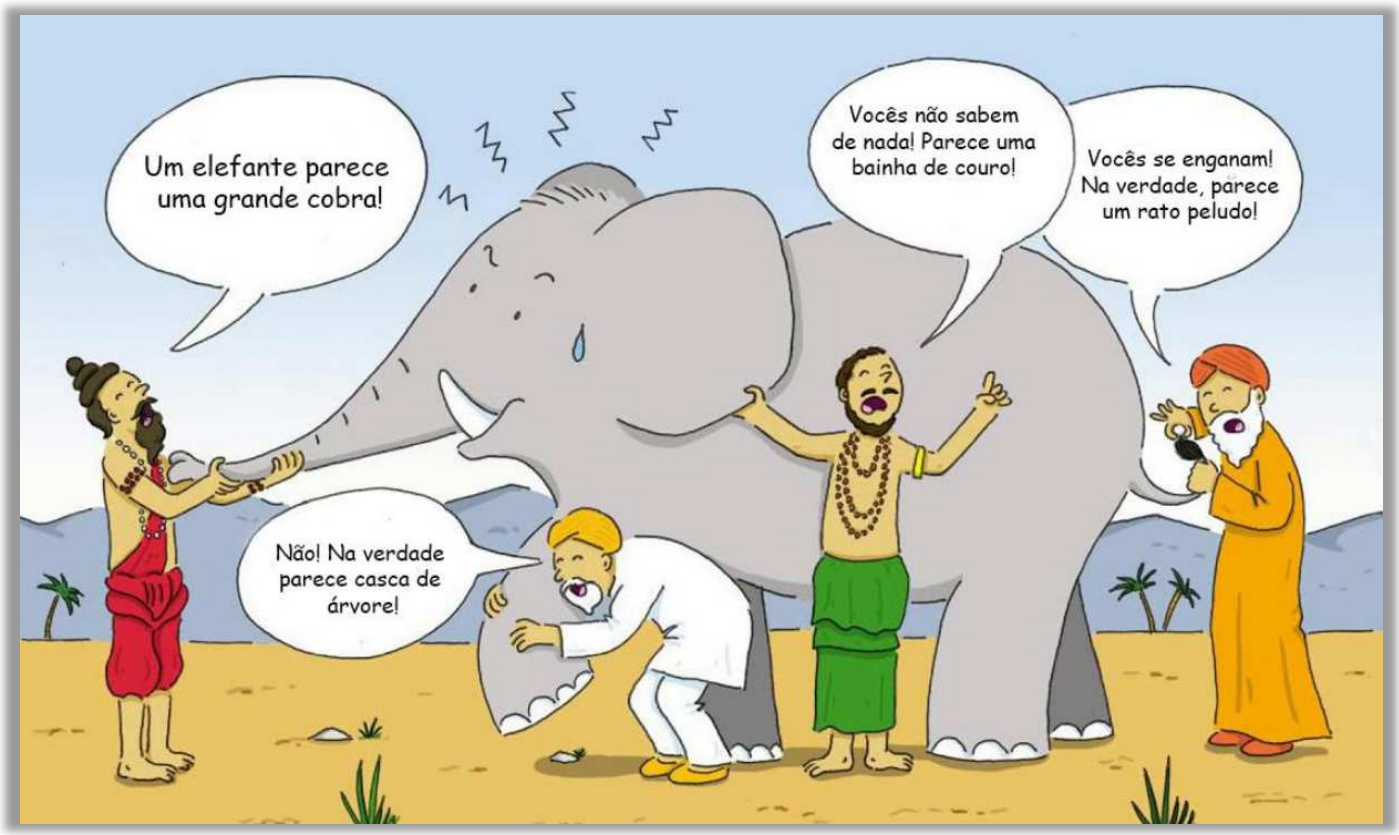
Trata-se de um tipo de técnica de aprendizado de máquina em que vários modelos são usados juntos para fazer previsões mais precisas do que qualquer modelo individual. Os modelos de Ensemble combinam as previsões de vários modelos para produzir melhor desempenho preditivo do que poderia ser obtido de qualquer um dos modelos individuais sozinhos. Essa técnica pode ser usada para problemas de regressão e classificação.

Nós já vimos diversos modelos de aprendizado de máquina na aula até agora. Cada um tem a sua particularidade, então pergunto: *por que não combinar esses modelos?* Ora, alguém já teve essa ideia: o nome disso é **Ensemble**! É normalmente utilizado quando os modelos individuais têm algum viés ou quando os dados são muito complexos para um único modelo capturar com precisão todos os padrões. São uma poderosa ferramenta para aumenta a precisão e reduzir o *overfitting*.

Vamos entender isso melhor! No aprendizado de máquina, não importa se estamos diante de um problema de classificação ou regressão, a escolha do modelo a ser utilizado é extremamente importante para termos alguma chance de obter bons resultados. Essa escolha pode depender de muitas variáveis diferentes do problema, tais como: quantidade de dados, dimensionalidade do espaço, tipo de dado, entre outros.

A imagem seguinte representa a *Fábula dos Homens Cegos e o Elefante*. Trata-se da história de um grupo de homens cegos que nunca viram um elefante antes e que imaginam como ele é ao tocá-lo. Cada um sente uma – e apenas uma – parte diferente do corpo do elefante (Ex: tromba, pata, orelha,

rabo). Eles, então, descrevem o elefante com base em sua experiência limitada e suas descrições do elefante são diferentes umas das outras (Ex: cobra, árvore, bainha de couro, rato peludo).

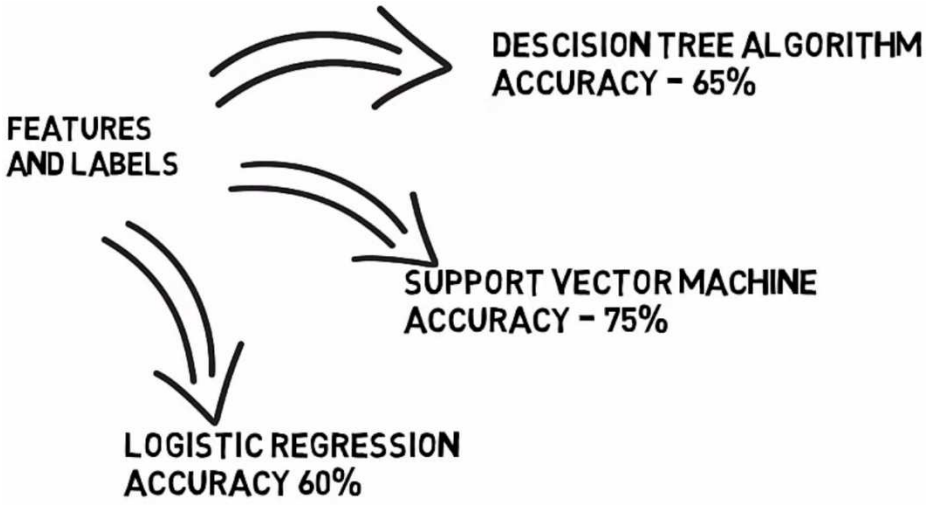


A história dos métodos de aprendizado de máquina é relativamente parecida. Quando temos um conjunto de dados complexos, faz-se necessário reunir os esforços de vários métodos (diferentes ou não) para se obter a melhor previsão possível. Falando de forma mais técnica, idealmente é desejado chegar a um modelo de aprendizado de máquina que tenha baixo viés e baixa variância. No entanto, no mundo real isso é muito difícil de alcançar com algoritmos isolados. *Por quê?*

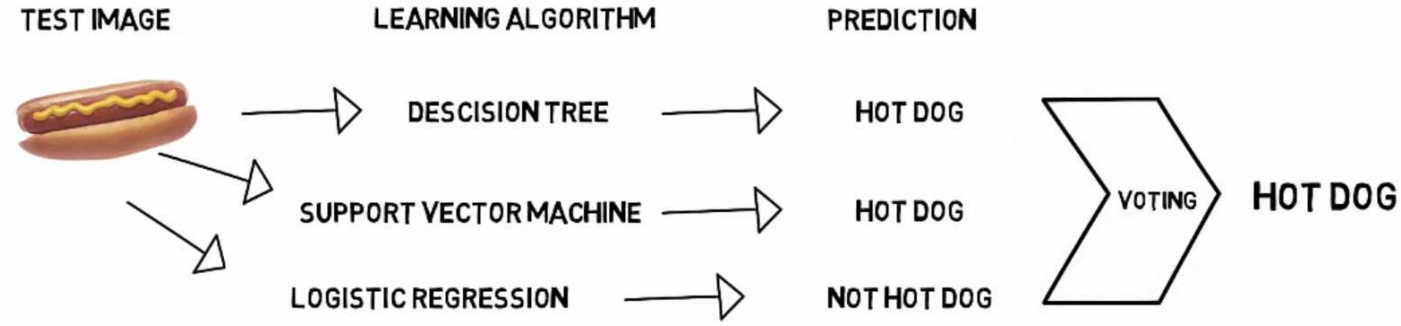
Porque sabemos que existe um trade-off: se aumentarmos a complexidade de um modelo, ele fica mais preciso (menor viés), mas fica mais sensível a pequenas flutuações (maior variância); se reduzirmos a complexidade de um modelo, ele fica menos sensível a pequenas flutuações (menor variância), mas fica menos preciso (maior viés). No entanto, se nós combinarmos diversos métodos de aprendizado de máquina, podemos chegar a algo mais razoável.

No contexto do *ensemble*, nós chamamos de *weak learner* (ou modelos de base) os modelos de aprendizado de máquina que podem ser utilizados como blocos de construção para projetar modelos mais complexos, combinando vários deles. Conforme vimos, na maioria das vezes, esses modelos básicos não funcionam tão bem sozinhos, seja porque apresentam um viés alto, seja porque têm muita variância para serem suficientemente robustos.

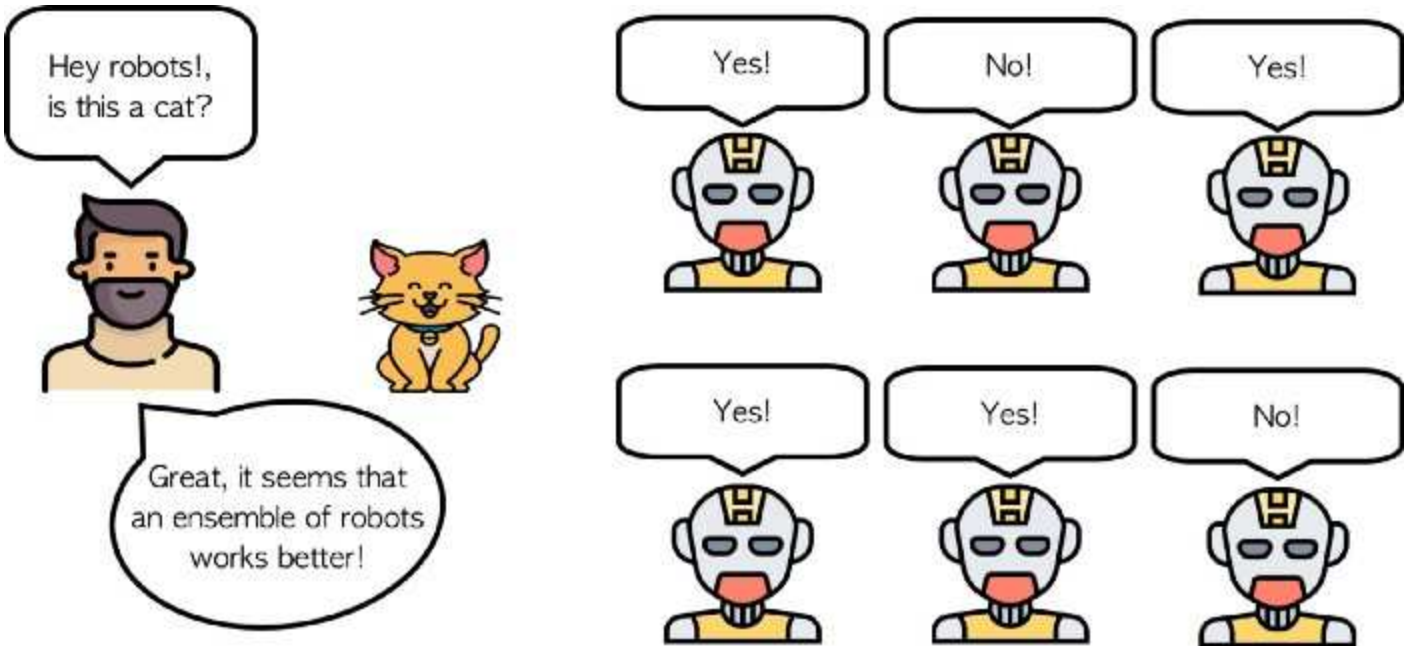
Assim, a ideia por trás do *ensemble* é tentar reduzir o viés e/ou variância desses *weak learners*, combinando vários deles para criar um *Strong Learner* (ou *Ensemble*) que alcance melhores desempenhos. *Vamos ver um exemplo?* Imagine que eu queira treinar um modelo de aprendizado de máquina para classificar imagens como imagens de cachorro-quente ou não imagens de cachorro-quente. Para tal, nós vamos combinar três algoritmos diferentes...



Em nossos treinamentos, conseguimos 65% de acurácia para Árvores de Decisão, 75% de acurácia para Support Vector Machine (SVM) e 60% de acurácia para regressão logística, logo o algoritmo que acertou mais vezes a imagem de um cachorro-quente foi o SVM. Ocorre que essa foi apenas a fase de treinamento! Para verificar se modelo de aprendizado é realmente bom, nós precisamos verificar com novos dados. Logo, testamos os três algoritmos com uma nova imagem:



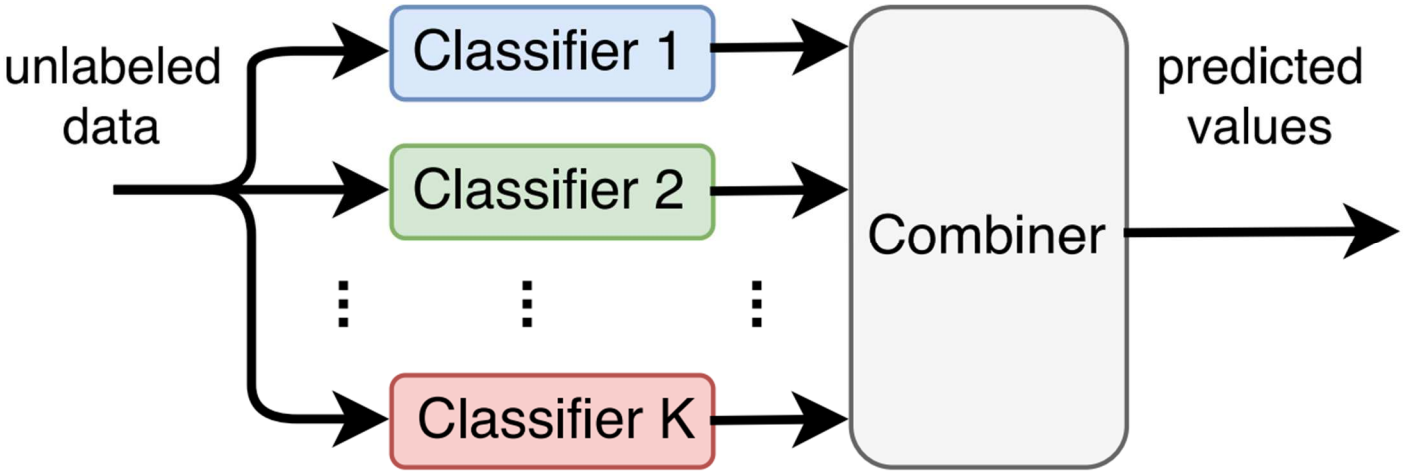
Note que o primeiro algoritmo classificou a imagem como sendo de um cachorro-quente e o segundo também, já o terceiro classificou como não sendo de um cachorro-quente. Como se trata de um *ensemble*, precisamos combinar as previsões para decidir qual será a decisão final do agregado de algoritmos. Para tal, fazemos uma votação: como o resultado foi 2x1 para cachorro-quente, é assim que a imagem será classificada.



Na imagem acima também tivemos uma votação em que cada robô representa uma instância de um algoritmo de aprendizado de máquina e o placar foi: 4x2 para imagem de um gato!

Professor, como seria se o ensemble fosse utilizado para regressão em vez de classificação? Nesse caso, o valor de saída seriam valores numéricos e, não, categóricos. Assim, a predição seria dada pela média aritmética simples ou ponderada dos valores de saída dos três algoritmos. O *ensemble* funciona, portanto, sob a forma de um comitê, utilizando-se de resultados de vários modelos preditivos aplicados sobre a mesma base de dados para atingir melhores resultados.

É importante destacar que é possível realizar diferentes tipos de agregação: nós podemos mudar os algoritmos de aprendizado utilizados; nós podemos aumentar/diminuir a quantidade de algoritmos utilizados; nós podemos mudar os hiperparâmetros utilizados nos algoritmos. Esse último caso é interessante: é possível fazer um *ensemble* com diversas instâncias do algoritmo de aprendizado de máquina, mas utilizando hiperparâmetros diferentes, por exemplo.



Quando utilizamos instâncias do mesmo algoritmo, dizemos que se trata de um *ensemble* homogêneo; caso contrário, dizemos que se trata de um *ensemble* heterogêneo. Outro ponto importante: a escolha dos algoritmos deve ser coerente com a forma como os agregamos. *Como assim, Diego?* Se escolhermos algoritmos com baixo viés, mas alta variância, ele deve estar com outro algoritmo que tende a reduzir a variância; e o contrário também.

Dito isso, fica a pergunta: *como combinar esses modelos diferentes de agregação de algoritmos de aprendizado de máquina?* Existem três maneiras (também chamadas de meta-algoritmos):

Bagging

ENSEMBLE: BAGGING

Trata-se de uma técnica de aprendizado de máquina de ensemble utilizado para minimizar a variância de um modelo. Para tal, ele cria vários modelos que são treinados em diferentes subconjuntos dos mesmos dados e, em seguida, combina suas previsões. A ideia é que cada modelo individual tenha pontos fortes diferentes e, ao combiná-los, a precisão geral do modelo será aprimorada.

O *Bagging* (*Bootstrap Aggregating*) cria classificadores para o *ensemble* a partir de uma redistribuição do conjunto de dados de treinamento. Esse conjunto de dados de treinamento é gerado selecionando-se aleatoriamente os exemplos da base de aprendizagem com reposição. Dessa forma, o algoritmo provê a diversidade, lançando-se mão do conceito de redistribuição aleatória dos dados. Vamos entender isso melhor...

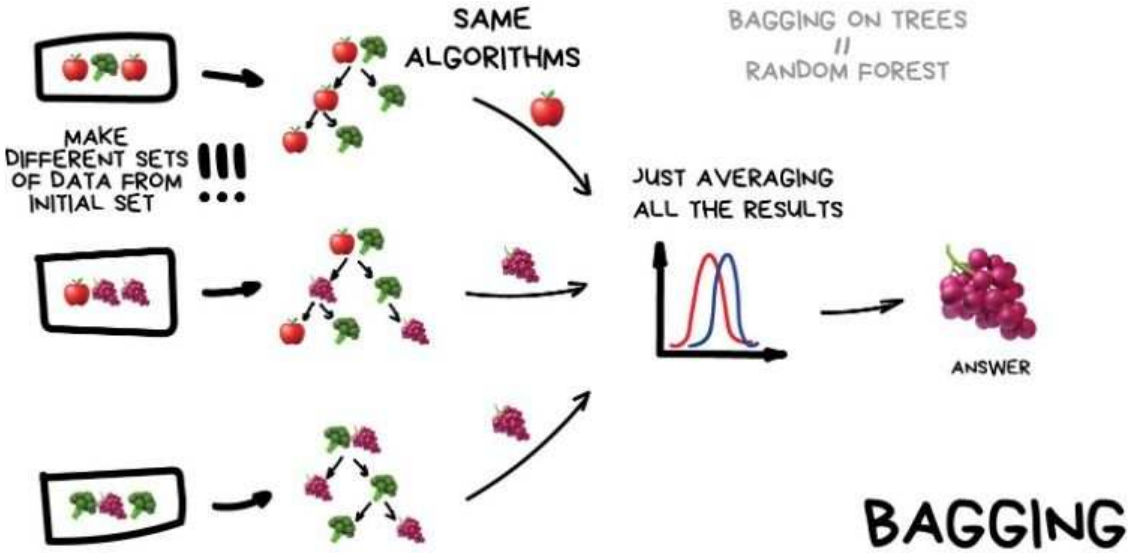
Você tem que entender dois pontos: primeiro, esse método utiliza um conjunto de algoritmos de aprendizado de máquina homogêneo (ou seja, são diversas instâncias do mesmo algoritmo); segundo, esse método utiliza uma técnica chamada *Bootstrapping*. Isso significa que são extraídos vários subconjuntos do conjunto de dados de treinamento por meio de uma amostragem aleatória, sendo que os pontos de dados podem aparecer mais de uma vez no subconjunto (podem se repetir).

O *Bootstrap* é utilizado para criar muitos conjuntos de treinamento diferentes que podem ser usados para treinar vários modelos. Vamos ver um exemplo:

Note que o conjunto de dados de treinamento original contém doze pontos de dados: {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12}. Em seguida, foram gerados diversos subconjuntos desse conjunto de dados de treinamento: {1, 10, 3, 8, 3}, {4, 2, 11, 9, 3}, ..., {5, 12, 1, 5 ,6}. O nome disso é amostragem com substituição e significa que, ao criar subconjuntos de dados, cada ponto de dados é escolhido aleatoriamente e pode ser incluído mais de uma vez no mesmo subconjunto.

No contexto de *Bagging*, essas diferentes amostras são utilizadas em diversas instâncias diferentes de algoritmos de aprendizado de máquina e geram um resultado numérico (regressão) ou categórico (classificação). Pense em um conjunto de dados de treinamento formado por fotos de diversos vegetais. Nós podemos utilizar o *bootstrapping* para gerar amostras com substituição: {maçã, brócolis, maçã}, {maçã, uva, uva} e {brócolis, uva, brócolis}.

Em seguida, nós passamos essas amostras para diferentes instâncias do mesmo algoritmo de aprendizado de máquina (Ex: Árvores de Decisão) de forma que ela possa classificar uma foto nova e, por votação ou média, decidir se a imagem é de uma maçã, uma uva ou um brócolis. É mais ou menos isso que é representado na imagem seguinte. Observação: as árvores aleatórias (*random forests*) são basicamente a utilização de *bagging* em árvores de decisão.



Em síntese: *bagging* é um método que utiliza diversas instâncias do mesmo algoritmo de aprendizado de máquina com o conjunto de dados de treinamento extraído por meio de amostragem por substituição do conjunto de dados originais a fim de minimizar a variância sem aumentar o viés combinando os resultados dos modelos de base em uma saída (output) que representa a média dos valores (regressão) ou maioria dos votos (classificação).

Por meio da combinação estatística de cada algoritmo, é possível aprimorar a estabilidade e acurácia, além de evitar o *overfitting*. É importante observar que o número de subconjuntos, bem como o número de itens por subconjunto e o tipo de algoritmo serão determinados pela natureza

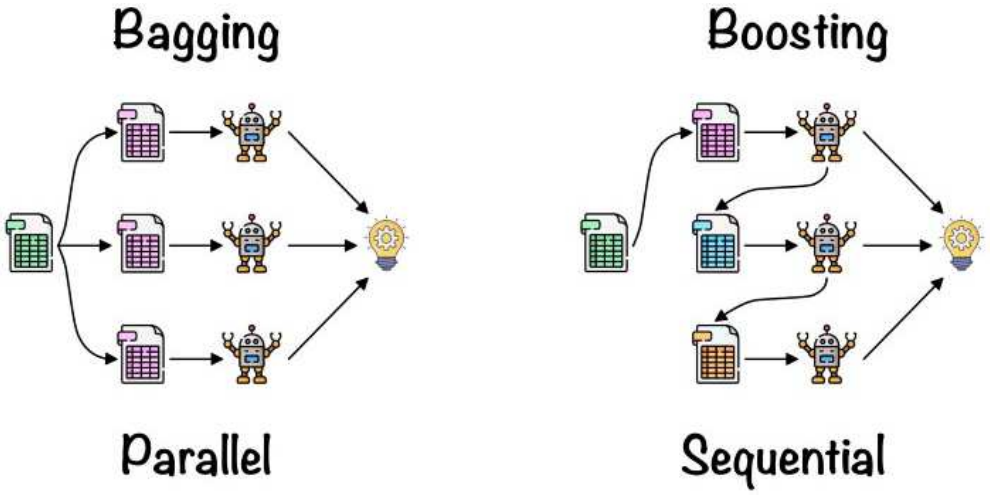
do seu problema de aprendizado de máquina. Para problemas de classificação, em geral são necessários mais subconjuntos em comparação com problemas de regressão.

Boosting

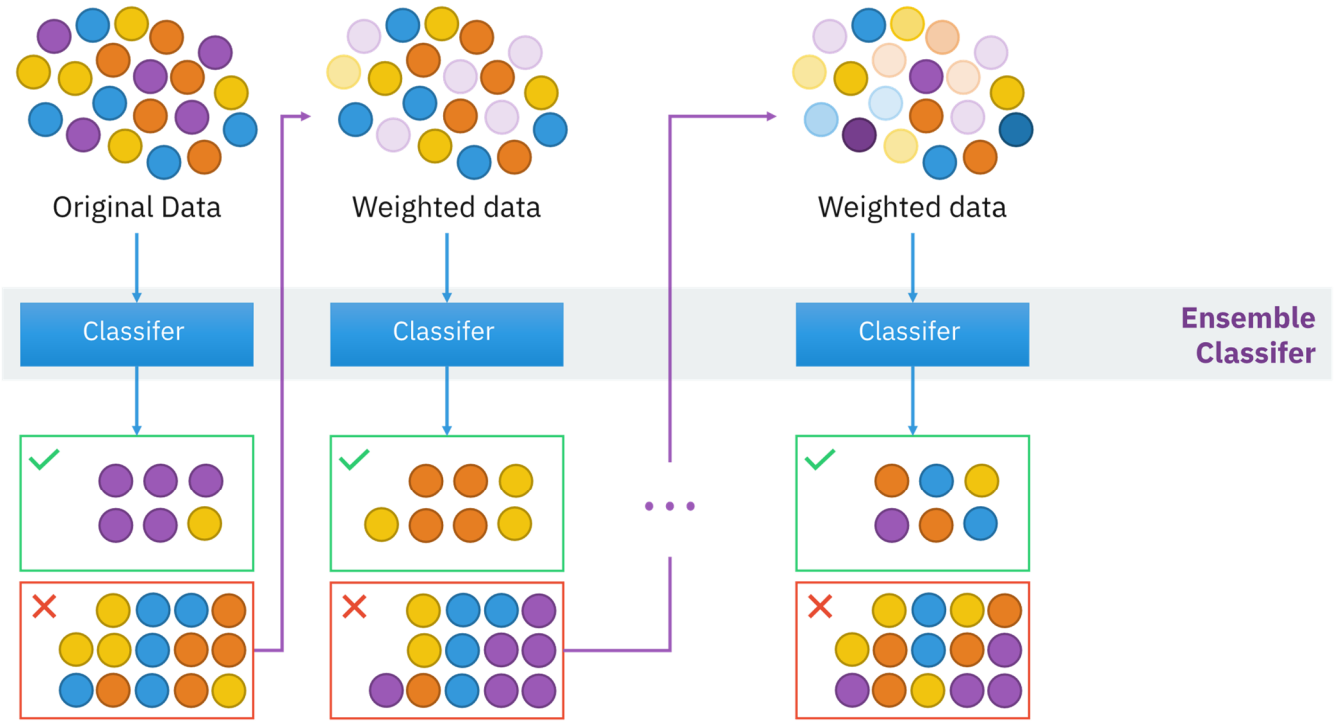
ENSEMBLE: BOOSTING

Trata-se de uma técnica de aprendizado de máquina de ensemble que combina vários weak learners para criar um strong learner. Funciona treinando cada weak learner em uma sequência, com cada aluno subsequente focando nos erros cometidos pelo aluno anterior. A saída dos weak learners é combinada para produzir um único strong learner. Boosting é freqüentemente usado para melhorar a precisão de modelos que são propensos a underfitting.

O *Boosting* também busca combinar algoritmos de aprendizado de máquina para gerar um *ensemble*, no entanto ele difere do *Bagging* na forma como os algoritmos são treinados. No *Bagging*, os algoritmos eram treinados independentemente uns dos outros, enquanto no *Boosting* os algoritmos são treinados sequencialmente, de forma que cada algoritmo subsequente se foque nos erros cometidos pelo algoritmo anterior.



Além disso, o *Bagging* é utilizado para reduzir a variância sem aumentar o viés e é geralmente mais adequado para resolver problemas de *overfitting*, enquanto o *Boosting* é utilizado para reduzir o viés sem aumentar a variância e é geralmente mais adequado para resolver problemas de *underfitting*. Para entender isso melhor, vamos dar uma olhada na imagem seguinte em temos um conjunto de dados original com pesos idênticos.



Esse conjunto de dados passa por um modelo de base que faz previsões - acertando algumas e errando outras classificações. Note pela imagem que ele acertou cinco bolinhas roxas e uma amarela; mas errou diversas outras. O algoritmo de *boosting* avalia as previsões e atribui pesos maiores às classificações incorretas do primeiro algoritmo – além de atribuir um peso ao modelo de base em si de acordo com seu desempenho. *Por que isso, Diego?*

Para que modelos que produzam previsões excelentes tenham uma grande influência sobre a decisão final. Os mesmos dados do conjunto inicial são passados para o próximo modelo de base fazer a mesma rotina, mas agora ponderados (com peso). Note que, no segundo passo, os dados classificados corretamente pelo modelo de base anterior estão mais claros (menor peso) e os dados classificados incorretamente pelo modelo de base anterior estão mais escuros (maior peso).

O segundo modelo de base recebe os dados que foram classificados pelo modelo anterior e também faz diversas previsões – acertando algumas e errando outras classificações. Essa rotina é repetida diversas vezes e, ao final de vários ciclos, o método de *boosting* combina essas regras fracas de previsão em uma única regra de previsão poderosa. Em geral, o *boosting* é utilizado com árvores de decisão: os algoritmos de *boosting* mais famosos são *AdaBoosting* e *Gradient Boosting*.

Vamos entender melhor essa questão da ponderação. Os pesos basicamente refletem a importância ou influência de um ponto de dado no modelo. *Como é isso, Diego?* Imaginem que um algoritmo de base está tentando detectar se há a presença de um cachorro em uma determinada imagem. Se a imagem for do rosto de um cachorro, de frente e bem iluminada, vocês hão de concordar comigo que é mais fácil de prever.

Já se a imagem for de um cachorro distante, em uma posição diferente e a imagem estiver com a qualidade meio ruim, é claro que é mais difícil de prever. Dito isso, os pontos de dados (imagens de cachorros) que foram classificadas erroneamente por um modelo de base devem ter maior influência (peso) para o próximo modelo de base. Além disso, o modelo de base que conseguir classificar corretamente uma imagem com peso maior também deve receber um peso maior.

Em síntese: *boosting* combina vários *weak learners* (modelos de base) de forma independente e sequencial, de modo que cada um compense a fraqueza do algoritmo anterior – além de utilizar uma amostragem de dados por substituição com ponderação (atribuição de pesos aos dados incorretamente previstos e aos modelos em si). Além disso, ele é fácil de entender e fácil de interpretar, aprendem com seus erros e não requerem pré-processamento de dados.

Stacking

ENSEMBLE: STACKING

Trata-se de uma técnica de aprendizado de máquina de ensemble que difere do bagging e boosting por utilizar um conjunto heterogêneo de weak learners e também por utilizar a saída dos weak learners como entrada em um meta-modelo com o objetivo de aprender o mapeamento entre as saídas e as classes corretas.

A ideia aqui é um pouquinho diferente: primeiro, nós dividimos nosso conjunto de dados em três partes: dados de treinamento, dados de validação e dados de teste – os dados de treinamento são usados para construir o modelo; os dados de validação são usados para avaliar o modelo e ajustar seus parâmetros; e os dados de teste são usados para medir o desempenho do modelo após ele ter sido treinado e otimizado (eles fornecem uma medida imparcial do desempenho para dados novos).

A

X0	x1	x2	xn	y
0.17	0.25	0.93	0.79	1
0.35	0.61	0.93	0.57	0
0.44	0.59	0.56	0.46	0
0.37	0.43	0.74	0.28	1
0.96	0.07	0.57	0.01	1

B

X0	x1	x2	xn	y
0.89	0.72	0.50	0.66	0
0.58	0.71	0.92	0.27	1
0.10	0.35	0.27	0.37	0
0.47	0.68	0.30	0.98	0
0.39	0.53	0.59	0.18	1

C

X0	x1	x2	xn	y
0.29	0.77	0.05	0.09	?
0.38	0.66	0.42	0.91	?
0.72	0.66	0.92	0.11	?
0.70	0.37	0.91	0.17	?
0.59	0.98	0.93	0.65	?

Vejam a imagem anterior: ela é composta de três tabelas, que representam os conjuntos de dados A, B e C – sendo A o conjunto de dados de treinamento, B o conjunto de dados de validação e C o conjunto de dados de teste. Cada tabela tem ***n*** variáveis de entrada ou *features* (x_0, x_1, x_2, ... x_n) e uma variável-alvo (y). Interpretando a tabela A, podemos ver que os valores da primeira linha (0.17, 0.25, 0,93, 0.79), por exemplo, geram um resultado que foi classificado como y = 1.

Professor, por que tem uma interrogação na variável alvo da tabela C? Porque essa é a tabela dos dados de teste, isto é, a tabela utilizada para medir o desempenho – o resultado da variável-alvo é justamente o que queremos prever. Bacana! Então agora nós vamos utilizar três algoritmos de aprendizado de máquina diferentes para treinar nosso modelo sobre o mesmo conjunto de dados de treinamento (Ex: KNN será o algoritmo 0, SVM será o algoritmo 1 e RNA será o algoritmo 2).

Após treinar o algoritmo por meio dos três algoritmos (chamados de modelos de base) utilizando os dados da Tabela A, vamos aplicá-los aos dados das tabelas B e C – o que resultará nas predições apresentadas nas tabelas B1 e C1 a seguir. Notem que as tabelas B1 e C1 contêm três colunas (pred0, pred1 e pred2) e a mesma variável-alvo. Cada modelo foi treinado e, quando aplicados os valores das *features* de B e C, foi retornado um valor de predição. Vamos interpretar as tabelas...

B1

pred0	pred1	pred2	y
0.24	0.72	0.70	0
0.95	0.25	0.22	1
0.64	0.80	0.96	0
0.89	0.58	0.52	0
0.11	0.20	0.93	1

C1

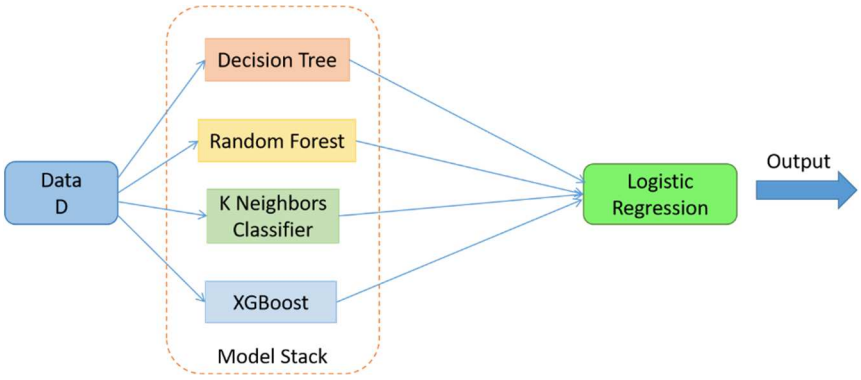
pred0	pred1	pred2	y	Preds3
0.50	0.50	0.39	?	0.45
0.62	0.59	0.46	?	0.23
0.22	0.31	0.54	?	0.99
0.90	0.47	0.09	?	0.34
0.20	0.09	0.61	?	0.05

O algoritmo 0 (Ex: KNN) foi treinado com os dados da Tabela A e depois aplicado aos dados da Tabela B, gerando um output (Ex: 0.24), e também aos dados da Tabela C, gerando outro output (Ex: 0.50). É claro que isso foi feito para cada linha das tabelas, então – como tínhamos cinco linhas nas Tabelas B e C – foram geradas cinco linhas também nas tabelas B1 e C1. Legal, agora vem o pulo do gato: o metamodelo (também chamado de modelo de nível 1).

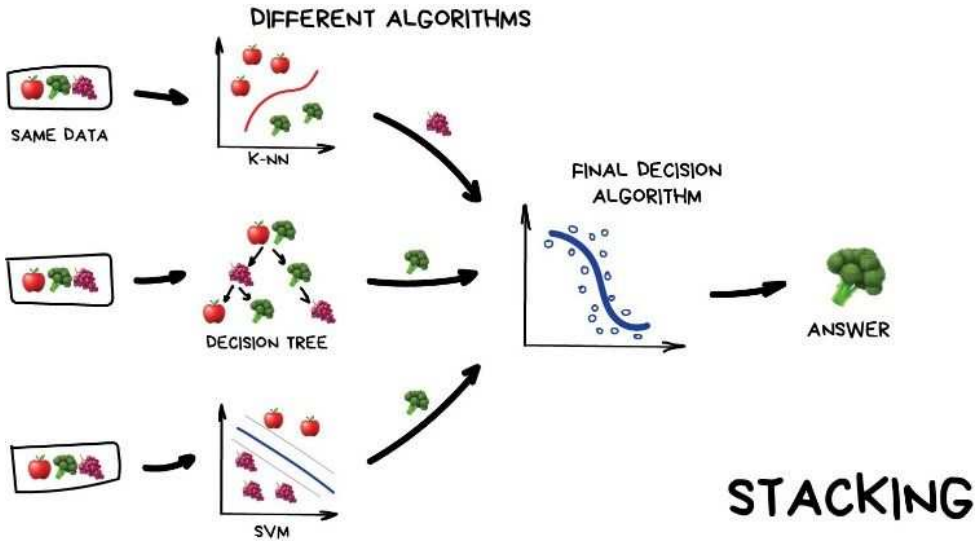
O metamodelo é mais um algoritmo de aprendizado de máquina construído sobre um conjunto de outros modelos de base (também chamados aqui de modelos de nível 0). Esse metamodelo busca a combinação ótima dos modelos de base, recebendo como entrada (*input*) a previsão dos modelos da etapa anterior com o intuito de devolver uma saída (*output*) com a previsão final – o que ajuda a reduzir o viés e melhorar a previsão.

Ao final, as predições de B1 se tornam entrada para treinar o Algoritmo 3 e fazemos previsões para C1 a fim de verificar a acurácia.

O nome *stacking* é porque nós vamos "empilhando" algoritmos tipicamente heterogêneos de aprendizado de máquina para fazer previsões. O esquema seguinte representa de forma simplificada tudo que vimos: um mesmo conjunto de dados de treinamento é passado para diferentes modelos de aprendizado de máquina que geram predições; essas predições servem de entrada para um metamodelo fazer a predição final.



O esquema a seguir exibe de forma mais lúdica: temos os mesmos dados de entrada, mas diferentes modelos de base fazem predições que se tornam entradas para o metamodelo.



Técnicas de Regularização

RELEVÂNCIA EM PROVA: BAIXA

REGULARIZAÇÃO

Trata-se do controle fino do nível de complexidade de um dado modelo, limitando seu grau de liberdade (flexibilidade) para se ajustar aos dados de treinamento, visando evitar sobreajustamento (overfitting). Em outras palavras, essa técnica reduz a variância de um modelo, tornando-o mais generalizável.

Nós vimos anteriormente que – para qualquer problema de aprendizado de máquina – é possível afirmar que um ponto de dado observado é o resultado de uma função de variáveis disponíveis + uma função de variáveis indisponíveis + variações. Como – por óbvio – não temos disponíveis as variáveis indisponíveis, podemos simplificar para: um determinado ponto de dado qualquer é o resultado de uma função de variáveis disponíveis + variações.

Por exemplo: se quiséssemos modelar o preço de um apartamento em um bairro de uma cidade grande, poderíamos dizer que isso dependerá da área total do apartamento, número de quartos, quantidade de vagas de garagem, entre outros. Se disponibilizássemos esses dados de milhares de apartamentos de um bairro para treinamento, o modelo de aprendizado de máquina tentaria encontrar um padrão a partir dessas características na forma de uma equação.

Ocorre que os apartamentos não obedecem diretamente a esse padrão. *Como assim, Diego?* Ora, essa fórmula é muito simples: nós informamos a área total, o número de quartos e a quantidade de banheiros, e ela me retorna uma previsão de valor de um apartamento come essas características. Ocorre que existem diversos casos de apartamentos com exatamente a mesma área total, mesmo número de quartos e mesma quantidade de banheiros, e que possuem preços diferentes.

Vocês nunca viram apartamentos em um mesmo prédio com as mesmas característica, mas que possuem valores bem distintos? Pois é, essa variação de preço entre apartamentos com as mesmas características modeladas é parte das variações – principalmente por conta de ruído. Você pode me dizer: *professor, não basta adicionar mais algumas características para ter uma previsão mais certeira do preço de um apartamento?* Boa ideia, nós podemos tentar!

Vamos adicionar mais alguns parâmetros, tais como: valor do condomínio, proximidade do centro da cidade e valor do IPTU. Podemos disponibilizar essas informações para o treinamento do modelo de aprendizado de máquina de forma que ele nos forneça uma equação mais complexa e precisa para previsão do preço de um apartamento nesse bairro. Ocorre que ainda assim existirão apartamentos com essas características exatamente iguais e preços diferentes.

Professor, vou apelar agora: quero inserir mais características! Opa... você é quem manda! Vamos adicionar agora a quantidade de guarda-roupas, lâmpadas e torneiras para o nosso modelo fazer a melhor previsão possível. Bacana! Agora nós temos um modelo tão preciso e tão complexo que ele

é capaz de prever o preço de qualquer apartamento em nossa lista de dados de apartamentos. *Esse é o mundo perfeito, correto?* Não, vocês caíram em um erro já conhecido chamado:

OVERFITTING

À medida que vamos adicionando mais variáveis, tendemos a fazer previsões cada vez melhores sobre os dados de treinamento. No entanto, além de um certo ponto, a adição de mais variáveis deixa de ajudar na modelagem e começa a atrapalhar. O modelo começa a encontrar padrões onde existem variações (aleatoriedades, ruídos, etc), dado que não existe lei que obrigue uma pessoa a seguir uma equação para definir o preço de seu apartamento – ela escolhe o preço que quiser!

Observe que o nosso modelo está tão complexo que ele está até mais complexo que o próprio fenômeno que ele está tentando modelar. Nós sabemos que o valor de um apartamento está intimamente ligado a algumas dessas características, mas não todas. *Onde já se viu calcular o valor de um apartamento pela quantidade de torneiras?* Nós só adicionamos essas variáveis para se ajustar aos nossos dados de treinamento, mas esquecemos que os preços de imóveis possuem variações.

Ao adicionar cada vez mais variáveis, o modelo começa a procurar padrões onde não existe! Ele vai tentar encontrar padrões onde, na verdade, existe apenas uma variação/ruído. *E há como encontrar padrão em variações aleatórias, professor?* Pessoal, há como encontrar padrão em basicamente qualquer coisa! Existe um fenômeno cognitivo chamado Apofenia, que é a percepção de padrões ou conexões em dados aleatórios.



Sabe aquelas pessoas que veem rostos na fumaça do acidente do 11 de Setembro, rostos nas crateras da lua ou a imagem de Nossa Senhora no vidro de uma janela no interior de São Paulo? Ou aquelas pessoas que "descobrem" que todo vencedor da mega-sena da Virada de anos pares fez uma aposta única em que o segundo número era primo? Pois é, não existe nenhum padrão nessas situações! É tudo aleatório, mas – se forçarmos a barra – nós conseguimos encontrar padrões na aleatoriedade.

Se nós – humanos – conseguimos encontrar padrões em coisas aleatórias, imagine uma máquina com altíssimo poder computacional. No entanto, é bom sempre salientar que se trata de um padrão espúrio, ilegítimo, mentiroso! *E como podemos ter certeza disso?* A avaliação sobre o desempenho de um modelo vem por meio da sua execução sobre dados de teste. Como ele modelou padrões espúrios, ele não conseguirá fazer boas previsões sobre novos dados. *Como é, Diego?*

Voltando ao nosso exemplo lúdico, a pessoa que achou aquele padrão espúrio na mega-sena da virada não vai vencer no próximo sorteio porque esse padrão encontrado é falso! Esse é o caso típico de *overfitting*, em que um modelo possui ótima precisão em suas previsões com dados de treinamento, mas varia bastante quando aplicado a novos dados. *E onde entra a regularização nessa história?* A regularização é o ajuste fino da complexidade do modelo.

A regularização pode ser aplicada a modelos de árvore de decisão, modelos de regressão, modelos de redes neurais, etc. Vamos ver como a regularização se aplica a esses modelos...

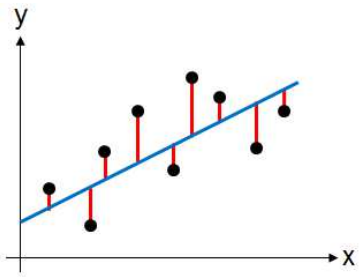
Aplicada a Regressões

A técnica de regularização para modelos de aprendizado de máquina que utilizam regressão linear busca efetivamente regularizar, normalizar ou suavizar modelos excessivamente complexos ou que dão muito destaque para uma característica específica. *Como assim, Diego?* A ideia aqui é manter as características, mas impor uma restrição à magnitude dos coeficientes. Quando as restrições são aplicadas aos parâmetros, o modelo produz uma função mais suave e menos propensa a *overfitting*.

Para tal, são introduzidos parâmetros de regularização, conhecidos como fatores de regularização ou termos de penalização, que controlam a magnitude dos pesos dos parâmetros, comprimem seus valores e garantem que o modelo não esteja se adaptando aos dados de treinamento. É como se ele inserisse uma penalidade à reta de melhor ajuste aos dados de treino a fim de reduzir a variância nos dados de teste e restringir a influência das variáveis sobre o resultado previsto.

$$y_i = b + \underbrace{w_1 x_{i1} + \cdots + w_p x_{ip}}_{\mathbf{w}\cdot\mathbf{x}_i}.$$

Acima temos a fórmula de uma regressão linear de múltiplas variáveis. Para essa equação, nós temos que: x é o valor de cada variável de entrada; y é o valor real de saída previsto; p é o número total de variáveis; b é o coeficiente do modelo; w é o peso ou coeficiente de cada variável. *E como sabemos que uma função está bem ou mal ajustada aos dados de teste?* Por meio do cálculo do erro: quanto menor o erro, melhor o ajuste!



$$\text{RSS} = \sum_{i=1}^{n} [y_i - (\mathbf{w} \cdot \mathbf{x}_i + b)]^2$$

O erro é a diferença entre o valor obtido e o valor previsto, logo a distância em vermelho entre os pontos pretos e a reta azul é chamada de erro de previsão. Cada ponto de dados em um conjunto cria tais erros e, para calculá-lo, utilizamos uma função erro (também chamada de função custo). Essa função geralmente é representada como a Residual Squared Sum (RSS), que é a soma dos quadrados dos erros e sabemos que o erro é a diferença entre valor obtido e previsto.

É isso que a fórmula anterior tenta representar: um modelo preditivo de aprendizado de máquina tenta ajustar os dados de forma que minimize a função de custo, isto é, reduza a soma dos quadrados dos erros ou resíduos. Existem dois tipos básicos: Lasso (L1) e Ridge (L2). A diferença entre elas está do termo de penalização ou regularização utilizado no coeficiente, mas ambas ajudam a reduzir o excesso de adaptação aos dados de treinamento.

Lasso (L1)

REGRESSÃO LASSO (L1)

Trata-se de uma técnica de regularização chamada Least Absolute Shrinkage and Selection Operator (LASSO) para reduzir a complexidade de um modelo preditivo. Essa técnica reduz certos coeficientes a zero, eliminando assim as variáveis menos importantes do modelo e ajudando a reduzir o overfitting. O resultado é um modelo esparso que é mais fácil de interpretar e melhor na generalização para novos dados.

Além de diminuir a variância do modelo, essa regularização tem uma outra importante aplicação em aprendizado de máquina: quando há múltiplas variáveis altamente correlacionadas (ou seja, que se comportam da mesma maneira) essa regularização seleciona apenas uma dessas variáveis e zera os coeficientes das outras. Desse modo, esse modelo realiza uma seleção de variáveis de forma automática, gerando vários coeficientes com peso zero.

$$\text{RSS}_{\text{lasso}} = \sum_{i=1}^{n} [y_i - (\mathbf{w} \cdot \mathbf{x}_i + b)]^2 + \alpha \sum_{j=1}^{p} |w_j|$$

$\alpha \sum_{j=1}^{p} |w_j|$: regularização ℓ_1

Vejam que o início da fórmula é idêntico à fórmula de RSS, mas ao final temos uma soma (representada como Regularização L1) responsável por penalizar a função objetivo[9]. No caso do Lasso (L1), essa regularização se dá pela soma (Σ) dos pesos (*w*) em valores absolutos (|*w*|). Isso leva a ter diversos atributos com peso zero, isto é, modelos mais simples, com menos atributos – apenas aqueles que são realmente fundamentais para reduzir significativamente o erro de previsão.

Ridge (L2)

REGRESSÃO RIDGE (L2)

Trata-se de uma técnica de regularização que adiciona um termo de penalidade à função de custo para reduzir a complexidade do modelo. O termo de penalidade é um parâmetro de regularização que reduz as estimativas do coeficiente para zero. Essa técnica é útil para prevenir o overfitting e melhorar a precisão do modelo. Ela também pode ser usada para identificar variáveis importantes em um conjunto de dados.

Nesse caso, a penalização consiste nos quadrados dos coeficientes, ao invés de seus módulos. *Qual será o efeito dessa regularização nos coeficientes de duas variáveis altamente correlacionadas?* Poderíamos ter duas variáveis com coeficientes parecidos, ou uma com coeficiente alto, e outra com coeficiente zero. Como a penalização é desproporcionalmente maior para coeficientes maiores, essa regularização faz com que variáveis correlacionadas tenham coeficientes parecidos.

$$\mathrm{RSS}_{\mathrm{ridge}} = \sum_{i=1}^{n} \left[y_i - (\mathbf{w} \cdot \mathbf{x}_i + b) \right]^2 + \alpha \sum_{j=1}^{p} w_j^2$$

regularização ℓ_2

Mais uma vez, o início da fórmula também é idêntico à fórmula de RSS, mas ao final temos uma soma (representada pela Regularização L2) responsável por penalizar a função objetivo. No caso da Ridge (L2), essa regularização se dá pela soma (Σ) do quadrado dos pesos (w^2), mas não em valores absolutos (percebam que não há o sinal de módulo). Entre a regularização L1 e L2, a segunda é bem mais comum de ser utilizada.

OBS: EU NÃO ACREDITO QUE SEJA NECESSÁRIO MEMORIZAR ESSAS FÓRMULAS (OREMOS!)

Vamos dar uma resumida: a regularização no contexto de regressões funciona adicionando um termo de penalidade à função objetivo (função de custo) do modelo. Este termo de penalidade é em função da magnitude dos coeficientes (pesos) do modelo e aumenta à medida que a magnitude dos coeficientes aumenta – o que encoraja o modelo a manter os coeficientes pequenos. Isso ajuda a reduzir o *overfitting* e melhorar a generalização do modelo.

[9] A função objetivo é utilizada para medir o desempenho de um modelo de aprendizado de máquina. Trata-se de um valor numérico que representa o quão bem o modelo prevê resultados corretos. Em geral, a função objetivo é uma função de custo/perda, como o Erro Quadrático Médio (EQM) ou o Erro Absoluto Médio (EAM). O objetivo é minimizar a função objetivo para obter o melhor desempenho possível do modelo.

Há duas formas mais comuns: Lasso (L1) e Ridge (L2). A primeira calcula a soma dos pesos absolutos e a segunda calcula a soma do quadrado dos pesos. É isso que vocês precisam saber...

Aplicada a Árvores de Decisão

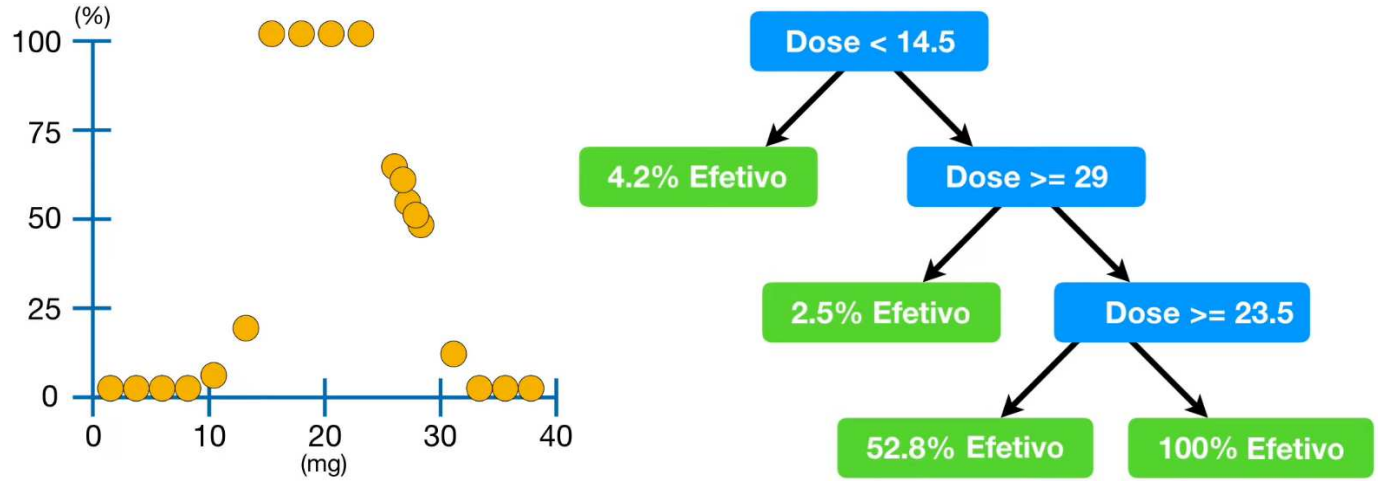
A regularização em árvores de decisão funciona introduzindo uma penalidade para a complexidade de uma árvore. *Como, Diego?* Bem, isso pode ser feito limitando a profundidade a árvore, a quantidade de *features*, a quantidade de observações por nó ou por folha; ganho mínimo de informação; entre outros. Quanto mais complexa for uma árvore, maior será a penalidade, o que encoraja o modelo a simplificar a árvore e evitar o *overfitting*. Para tal, utiliza-se a técnica de poda...

Pruning (Poda)

PRUNING

Trata-se de uma técnica de reegularização utilizada para reduzir a complexidade de um modelo removendo parâmetros ou conexões desnecessárias em uma rede. Isso ajuda a melhorar a precisão do modelo preditivo, reduzindo o overfitting, melhorando a velocidade de treinamento e reduzindo os requisitos de memória.

Vamos imaginar uma árvore de decisão de regressão que modele a efetividade de um remédio de acordo com a sua dosagem de tal modo que, quando a dose do remédio é muito baixa ou muito alta, ele não era efetivo. Lembrando que uma árvore de decisão de regressão é aquela utilizada para prever valores contínuos (em contraste com as árvores de decisão de classificação, que buscam prever categorias). A imagem seguinte representa o gráfico (dose x efetividade) e a árvore...



Vamos interpretar o que a árvore de decisão quer nos dizer: se a dose < 14,5 mg, o remédio é apenas 4,2% efetivo; se a dose >= 14,5 mg e <= 29 mg, o remédio é 2,5% efetivo; por outro lado se a dose < 29 mg e >= 23,5 mg, o remédio é 100% efetivo. Logo, de acordo com os dados de treinamento, a

dose ideal é entre 23,5 mg e 29 mg. Agora imagine que nós aumentemos ainda mais a profundidade da árvore com novos nós.

Como assim, Diego? Imagine um novo nível que verifica a melhor dose entre 24 mg e 28mg; e depois outro nível que verifica a melhor dose entre 25 mg e 27 mg; e depois outro nível que verifica a melhor dose entre 25,5 mg e 26,5 mg; e por aí vai! Ora, quando mais aumentamos a profundidade da árvore (e seus nós), mais a árvore de ajusta aos dados de treinamento e menos ela se torna capaz de generalizar novos dados de teste.

Nesse caso, teremos uma árvore de decisão excessivamente complexa que modela padrões espúrios (ruídos) em vez de modelar tendências do sinal subjacente. Em outras palavras, a nossa árvore de decisão realiza um sobreajustamento sobre os dados de treinamento – também conhecido como *overfitting*. Uma maneira de simplificar a árvore e evitar esse sobreajuste é por meio da técnica de regularização de árvores de decisão chamada *pruning* (ou poda).

Essa técnica de regularização busca reduzir o tamanho da árvore ao remover nós desnecessários ou redundantes. Isso ajuda a evitar o *overfitting* e melhora a generalização do modelo. O processo de poda começa calculando o custo de uma árvore antes e depois de um nó ser removido. Se o custo após a remoção for menor do que o custo anterior, o nó será removido; e o processo de remoção continua até que nenhum outro nó possa ser removido.

A poda também pode ser usada para controlar a complexidade da árvore. Ao definir um limiar, é possível controlar o tamanho da árvore e evitar que ela se torne excessivamente complexa.

Aplicada a Redes Neurais

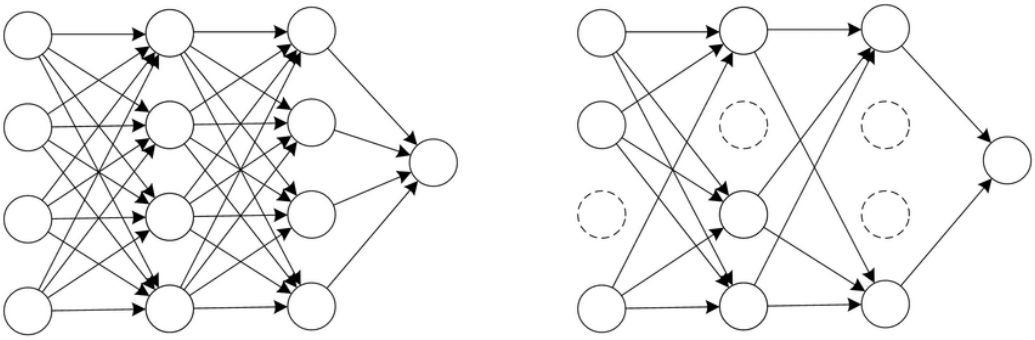
Agora vamos ver como a regularização pode ser aplicada a redes neurais, mas já tenho uma observação a fazer: os perceptrons de redes neurais fazem uma regressão linear/logística, logo as técnicas de regularização aplicadas a regressões também podem ser aplicadas a redes neurais. Por outro lado, aqui veremos outras técnicas de regularização, tais como: *Dropout*, *Early Stopping* e *Data Augmentation*.

Dropout

DROPOUT

Trata-se de uma técnica de regularização usada em redes neurais para evitar o overfitting. Ele funciona desconectando aleatoriamente uma fração das unidades de entrada durante o treinamento, o que força o modelo a aprender com menos parâmetros e reduz a sua complexidade. A fração de unidades de entrada a serem descartadas é normalmente definida como uma porcentagem entre 20% e 50%.

Trata-se de uma técnica de regularização utilizada em redes neurais que desativa/desconecta aleatoriamente neurônios da rede durante cada sessão de treinamento do modelo – isso ajuda a evitar o *overfitting* e melhorar a generalização. Essa técnica é utilizada somente durante o treinamento da rede neural e, não, durante o teste. Os neurônios desativados não recebem mais entradas, não produzem mais saídas e também não são ajustados.



(a) Standard Neural Network

(b) Network after Dropout

Nesse contexto, temos o conceito de taxa de *dropout*, que é basicamente a proporção de neurônios que são desativados aleatoriamente. Uma taxa de *dropout* mais alta significa que mais neurônios são desativados, e uma taxa de *dropout* mais baixa significa que menos neurônios são desativados. Em geral, recomenda-se uma taxa de *dropout* de 20% a 50%. Vejam na imagem anterior que tínhamos 12 neurônios e 4 deles descartados, logo temos 4/12 = 33% de taxa de *dropout.*

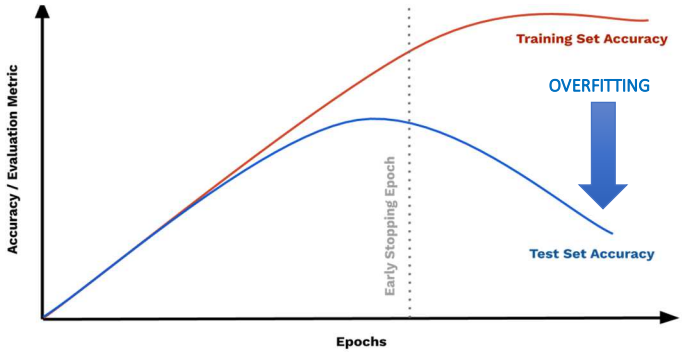
Ao desativar alguns neurônios, a rede neural não poderá contar com ativações específicas durante o período de treinamento do modelo. Dessa forma, ela será forçada a descentralizar em múltiplos neurônios algum padrão específico, gerando representações diferentes, distribuídas e redundantes. Por fim, note que – em contraste com as regularizações *lasso* e *ridge* – a técnica de *dropout* não depende de penalizações da função objetivo (perda ou custo) para evitar o *overfitting*.

Early Stopping

EARLY STOPPING

Trata-se de uma técnica regularização usada em redes neurais para evitar o overfitting dos dados de treinamento. Ele funciona monitorando o desempenho do modelo em um conjunto de dados de validação durante o treinamento e interrompendo o processo de treinamento quando o desempenho no conjunto de dados de validação não melhora por um determinado número de épocas de treinamento.

Também chamada de *Parada Precoce* ou *Parada Antecipada*, essa técnica envolve monitorar a precisão da validação do modelo durante o treinamento e interromper o treinamento quando a precisão parar de melhorar. A parada precoce é especialmente útil ao treinar modelos com grandes conjuntos de dados, uma vez que pode reduzir o tempo necessário para o treinamento. Para entendê-la melhor, vamos analisar o seguinte gráfico...



Temos um gráfico de Acurácia x Época! *O que seria uma época, professor?* Uma época é basicamente uma passagem completa pelo conjunto de dados. Nesse sentido, note que a curva vermelha representa o conjunto de dados de treinamento e a curva azul representa o conjunto de dados de teste. À medida que se aumenta o número de épocas, a acurácia também aumenta tanto nos dados de treinamento quanto nos dados de teste.

No entanto, percebam que em determinado momento, a acurácia dos dados de treinamento permanece aumentando até estabilizar enquanto a acurácia dos dados de teste começa a cair vertiginosamente. *O que isso nos indica?* Isso indica que houve um sobreajuste (*overfitting*), isto é, o modelo tem um excelente desempenho (alta acurácia) com dados de treinamento, mas um péssimo desempenho (baixa acurácia) com dados de teste.

Em outras palavras, quando redes neurais são treinadas múltiplas vezes, elas tendem a detectar padrões cada vez mais sutis, modelando o ruído em vez de modelar tendências subjacentes (*quantas zilhões de vezes já vimos isso?*). Então, é importante interromper o treinamento antes de a rede neural começar a se ajustar excessivamente aos dados de treinamento. Note no gráfico que a linha tracejada cinza nos mostra o momento em que o treinamento começa a gerar *overfitting*.

A ideia por trás da parada precoce é verificar periodicamente o desempenho dos dados de teste, isto é, se temos uma redução da acurácia nos dados de teste, é hora de interromper o treinamento.

Data Augmentation

DATA AUGMENTATION

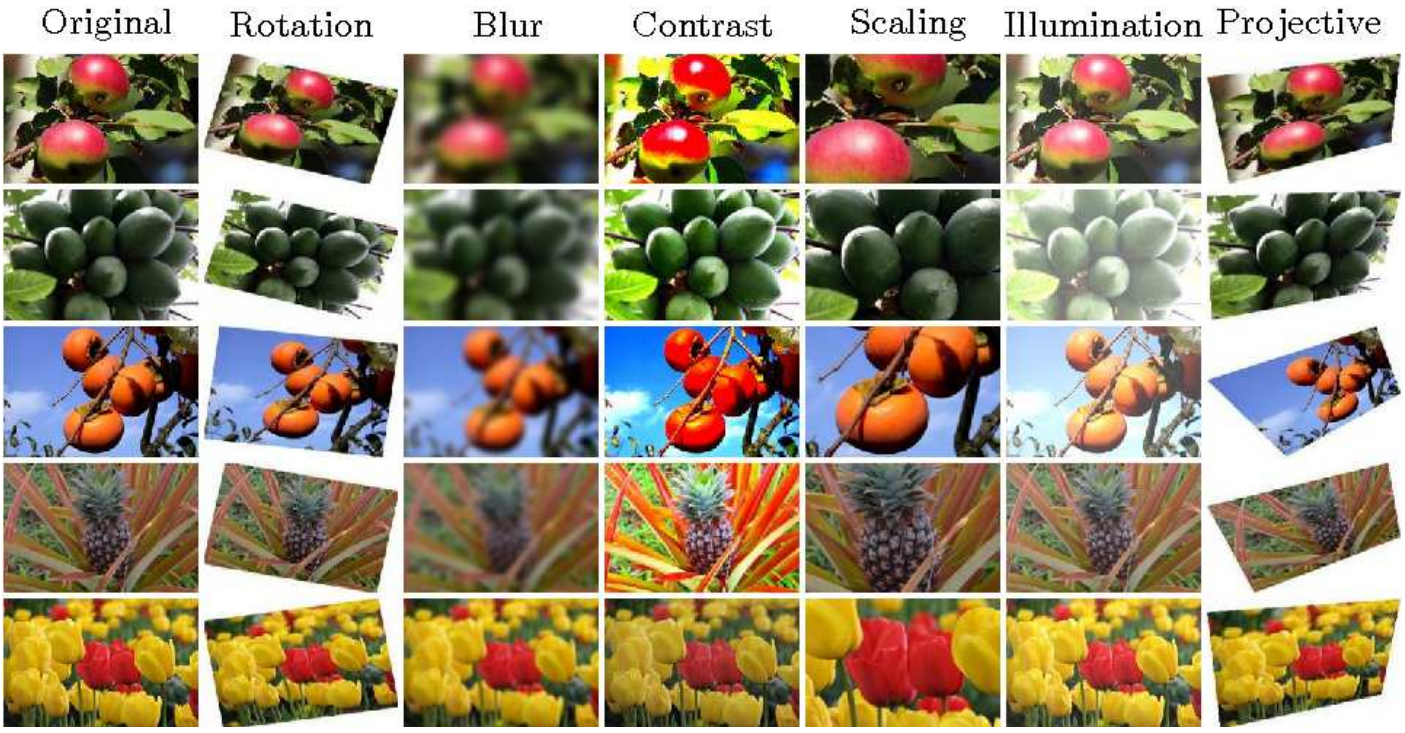
Trata-se de uma técnica de regularização usada para aumentar artificialmente o tamanho de um conjunto de dados de treinamento, manipulando e adicionando transformações aleatórias aos dados existentes. Isso ajuda a melhorar a precisão e a robustez dos modelos de aprendizado de máquina, introduzindo novas variações nos dados de treinamento. As técnicas de aumento de dados incluem corte, inversão, rotação, adição de ruído, deslocamento, etc.

Também chamado de Aumento/Expansão de Dados, essa técnica envolve a geração artificial de dados de treinamento adicionais aplicando transformações aleatórias aos dados de treinamento existentes. Isso ajuda a melhorar a generalização e reduzir o risco de *overfitting*. O Data Augmentation é útil para melhorar o desempenho e os resultados dos modelos de aprendizado de máquina, formando exemplos novos e diferentes para treinar conjuntos de dados.

Se o conjunto de dados em um modelo de aprendizado de máquina for rico e suficiente, o modelo terá um desempenho melhor e mais preciso. Além disso, essa técnica faz com que os modelos sejam mais robustos ao criar variações de dados que o modelo poderá ver no mundo real. Nós sabemos que coletar e rotular dados pode ser um processo exaustivo e caro. Essa técnica permite que as empresas reduzam esses custos operacionais.

O aumento de dados pode ser aplicado a outros tipos de dados, mas é mais comum com dados de imagens. Então pensem comigo: temos que treinar um algoritmo de aprendizado de máquina responsável por classificar a imagem de um animal como sendo de um cachorro ou de um gato. É possível buscar na internet e coletar fotos desses animais, mas é um processo custoso e o modelo será melhor, quanto mais bem treinado estiver.

Então, em vez de buscar mais imagens, nós podemos pegar as a imagens originais e gerar artificialmente mais dados de treinamento aplicando pequenas transformações que geram variações úteis para o treinamento, tais como cortar a imagem, invertê-la, rotacioná-la, redimensioná-la, aumentar o contraste, reduzir o brilho, colocar em preto & branco, dar zoom-in, dar zoom-out, entre outros.



Percebam que uma única imagem do conjunto de dados originais pode ser alvo de diversas transformações diferentes, aumentando a quantidade de dados de treinamento e permitindo que

o modelo generalize melhor os dados. Isso ajuda a reduzir a variância do modelo, pois o modelo será exposto a uma variedade maior de pontos de dados, o que o ajudará a generalizar melhor o modelo de aprendizado de máquina.

Controle de Complexidade

Pode-se controlar a complexidade do modelo por meio da redução do tamanho total da rede, redução do número de camadas ou redução da quantidade de camadas, entre outros.

Otimização de Hiperparâmetros

RELEVÂNCIA EM PROVA: BAIXA

OTIMIZAÇÃO DE HIPERPARÂMETROS

Trata-se do processo de selecionar o melhor conjunto de hiperparâmetros para um determinado algoritmo de aprendizado de máquina, a fim de maximizar seu desempenho em um determinado conjunto de dados. Isso geralmente é feito por meio de um processo de tentativa e erro, usando métodos como pesquisa em grade, pesquisa aleatória ou otimização bayesiana.

No aprendizado de máquina, um hiperparâmetro é um parâmetro cujo valor é definido antes do início do processo de aprendizado. Por outro lado, os valores de outros parâmetros são derivados por meio de treinamento. Na verdade, um hiperparâmetro é uma espécie de característica ou restrição inserida em um algoritmo de aprendizado de máquina para que ele possa realizar o treinamento. Vejamos definições mais completas:

PARÂMETROS	TRATA-SE DE REPRESENTAÇÕES INTERNAS DO MODELO AJUSTADAS AUTOMATICAMENTE PELO PROCESSO DE APRENDIZAGEM OU TREINAMENTO, SINTETIZADOS A PARTIR DE PADRÕES ESTATÍSTICOS DOS DADOS, TAIS COMO OS PESOS DE UMA REGRESSÃO OU DE UMA REDE NEURAL. EM OUTRAS PALAVRAS, TRATA-SE DE UMA VARIÁVEL DE CONFIGURAÇÃO INTERNA A UM MODELO E CUJOS VALORES PODEM SER ESTIMADOS A PARTIR DOS DADOS.
HIPER PARÂMETROS	TRATA-SE DAS CARACTERÍSTICAS DO MODELO OU DO SEU PROCESSO DE TREINAMENTO QUE REFLETEM UMA OPÇÃO DO CIENTISTA, QUE NÃO SÃO EXTRAÍDOS AUTOMATICAMENTE DOS DADOS, TAIS COMO A INTENSIDADE DE REGULARIZAÇÃO DE UM MODELO, A PROFUNDIDADE MÁXIMA DE ÁRVORES DE DECISÃO, ETC. EM OUTRAS PALAVRAS, TRATA-SE DE UMA VARIÁVEL DE CONFIGURAÇÃO EXTERNA A UM MODELO E CUJOS VALORES NÃO PODEM SER ESTIMADOS A PARTIR DOS DADOS.

Veja que os valores dos parâmetros são extraídos do próprio conjunto de dados a partir do aprendizado de máquina, logo estão sob controle do algoritmo e, não, do cientista de dados. Quando os valores estão sob controle do cientista de dados, chamamos de hiperparâmetros! Não há como saber com exatidão qual é o melhor valor para um hiperparâmetro de um problema específico porque isso dependerá de testes de tentativa e erro.

Nós vimos como funciona a regularização no tópico anterior, então agora vamos ver um exemplo de regressão linear com regularização. Ela poderia ser modelada com a seguinte equação:

$$y = a + bx \text{ ou}$$

Nesse contexto, **y** é o que desejamos prever (Ex: nota em uma prova); **x** é uma variável (Ex: horas de estudo); e **a** e **b** são parâmetros normais cujo valor o modelo vai aprender para minimizar os erros de predição – são também chamados de coeficientes ou pesos da equação. Ocorre que estamos falando de uma regressão linear com regularização, logo temos alguns parâmetros adicionais que definem pesos para evitar o famoso *overfitting*.

Vamos relembrar a regularização L1? Vejam que ela possui um parâmetro α antes do somatório responsável por penalizar variáveis com coeficientes altos.

$$\text{RSS}_{\text{lasso}} = \sum_{i=1}^{n} \left[y_i - (\mathbf{w} \cdot \mathbf{x}_i + b) \right]^2 + \alpha \sum_{j=1}^{p} |w_j|$$

regularização ℓ_1

Esse parâmetro α é, na verdade, um hiperparâmetro. *Por que, Diego?* Porque ele é configurado manualmente antes do treinamento para penalizar muito ou pouco os coeficientes da regressão linear a fim de melhorar a performance do modelo. Essa escolha de hiperparâmetros é também chamada de otimização (*tunning*) e trata da realização de experimentos com valores diferentes de hiperparâmetros com o intuito de descobrir quais deles geram os modelos mais eficientes.

Diferentes escolhas de hiperparâmetros levam a modelos treinados distintos com níveis de desempenho potencialmente diferentes. É importante entender também que, quando há mais de um hiperparâmetro em um modelo, eles podem interagir afetando o desempenho do modelo de uma maneira bastante complexa e imprevisível. Nesses casos, temos que testar também as combinações de valores de hiperparâmetros que geram melhores resultados.

Vejam a complexidade: o cientista de dados precisa definir quais hiperparâmetros ele utilizará; em seguida, ele precisa definir quais valores de hiperparâmetros ele testará para cada hiperparâmetro; por fim, ele precisa fazer experimentos com as diversas combinações de hiperparâmetros e seus valores para chegar ao modelo com melhor desempenho possível, isto é, o modelo mais otimizado. Vamos ver uma pequena metáfora para entender isso melhor...

Vocês já ouviram falar em carro tunado? Pois é, o *tuning* é o mesmo que otimizar, isto é, tirar o melhor desempenho de algo. Um carro vem de fábrica com sua configuração original, mas o dono pode modificar alguns componentes para melhorar o seu desempenho. É possível adicionar um chip de potência, trocar o filtro de combustível, trocar as velas de ignição, trocar válvulas do motor, trocar o escapamento, trocar a turbina, entre outros.

Da mesma forma que é possível fazer a otimização (*tuning*) de um carro, também é possível para um modelo de aprendizado de máquina – ambos utilizando hiperparâmetros. Galera, é claro que isso é apenas uma metáfora e toda metáfora tem a sua limitação. A minha ideia aqui foi apenas mostrar para vocês que um hiperparâmetro é como um parâmetro configurado externamente que permite otimizar algo. Bem, existem diversos tipos de otimização de hiperparâmetros:

Grid Search

GRID SEARCH

Trata-se de uma técnica usada para otimizar hiperparâmetros de um determinado modelo, a fim de obter o melhor desempenho possível. É uma pesquisa exaustiva sobre um conjunto de hiperparâmetros especificados manualmente pelo usuário. Funciona explorando sistematicamente cada combinação dos parâmetros de uma grade especificada e avaliando o desempenho do modelo para cada combinação. A combinação com a pontuação mais alta é então selecionada como o modelo de melhor desempenho.

Trata-se de um método eficaz para ajustar hiperparâmetros, sendo utilizado para melhorar o desempenho de generalização de um modelo. Ele testa exaustivamente todas as combinações possíveis dos valores de hiperparâmetros de interesse até encontrar os melhores. É fácil de implementar, mas é mais adequado quando temos poucas combinações – quando há muitas combinações, ele se torna computacionalmente inviável, demorando um tempo excessivo.

Em outras palavras, esse tipo de otimização testará por força bruta todas as combinações possíveis de valores de hiperparâmetros. Vamos voltar à nossa metáfora: imagine que cada uma das características que podem ser modificadas em um carro representa um hiperparâmetro do nosso modelo e que eu queira tunar o meu carro com o intuito de aumentar o desempenho e atingir o maior torque possível (torque é a força que um motor consegue gerar).

Eu levo meu carro a um especialista que será o responsável por testar todo e qualquer tipo de combinação de componentes para descobrir aquela que mais aumenta o desempenho do meu carro. Ocorre que sabemos que existem dezenas de marcas, tipos, formas, opções de chips de potência, turbinas, escapamentos, pneus, filtros, velas, entre outros. O coitado do especialista teria que testar absolutamente todas as combinações de cada um desses valores até descobrir a melhor.

Isso é inviável em termos de esforço e tempo! Além disso, há um segundo problema: o especialista é obrigado a testar combinações de valores que claramente (e intuitivamente) não aumentarão o desempenho do carro. Tudo isso é válido para o Grid Search: é útil quando temos que testar poucas combinações de valores para poucos hiperparâmetros, mas é inviável computacionalmente quando temos que testar muitas combinações de valores para muitos hiperparâmetros.

Random Search

RANDOM SEARCH

Trata-se de uma técnica usada para otimizar hiperparâmetros que envolve a amostragem aleatória de uma gama de valores de hiperparâmetros possíveis e, em seguida, a seleção do melhor conjunto com base nos resultados. É um dos métodos mais simples e eficientes de ajustar um modelo e é frequentemente usado como linha de base para comparação com técnicas de otimização mais avançadas.

Uma alternativa para resolver esse tipo de problema é por meio de uma pesquisa aleatória. Em vez de definir previamente quais serão os valores escolhidos para cada hiperparâmetro, nós definimos uma quantidade de testes e deixamos o algoritmo escolher aleatoriamente quais valores serão testados para cada hiperparâmetro. Após isso, ele executa a quantidade predefinida de testes com combinações aleatórias dos valores de hiperparâmetros.

Trata-se de uma alternativa mais barata porque não precisa testar exaustivamente todas as combinações possíveis de valores de hiperparâmetros, portanto aumenta bastante a velocidade e reduz bastante o tempo. Esse método é bastante útil quando temos uma quantidade extraordinária de hiperparâmetros, visto que sua aleatoriedade permite descobrir combinações que não teríamos imaginado intuitivamente.

Seria como se o especialista escolhesse aleatoriamente alguns componentes para avaliar possíveis trocas que melhorassem o desempenho do meu carro. Um problema com essa abordagem é que ela não garante que encontrará a melhor combinação de hiperparâmetros. Além disso, é possível que o algoritmo acaba explorando muito uma região do espaço de busca e explorando pouco outras. Por outro lado, há um maior controle do custo computacional.

Bayesian Search

BAYESIAN SEARCH

Trata-se de uma técnica usada para otimizar hiperparâmetros que usa a inferência bayesiana para construir um modelo probabilístico de um espaço de pesquisa. Ele usa esse modelo para otimizar o processo de busca, levando em consideração a incerteza do problema e orientando a busca para melhores soluções. É particularmente útil para problemas onde o espaço de busca é grande e o custo de avaliação de uma única solução é alto.

Ao contrário dos tipos vistos anteriormente, a otimização bayesiana utiliza desempenhos de hiperparâmetros anteriores para orientar quais valores de hiperparâmetros serão testados posteriormente. Em outras palavras, ela tenta estimar a probabilidade de desempenho de combinações em função de resultados já avaliados. Após cada avaliação, o algoritmo detecta quais valores de hiperparâmetro são mais interessantes de explorar e quais não são.

Após um número definido de iterações, o algoritmo retorna o valor ótimo. Voltando à nossa metáfora, seria como se o especialista testasse uma combinação:

```
  Chip de Potência = 600cv
+ Turbina = 32mm
+ Escapamento Downpipe = 3,5””
_____________________________
= Torque de 35kgfm
```

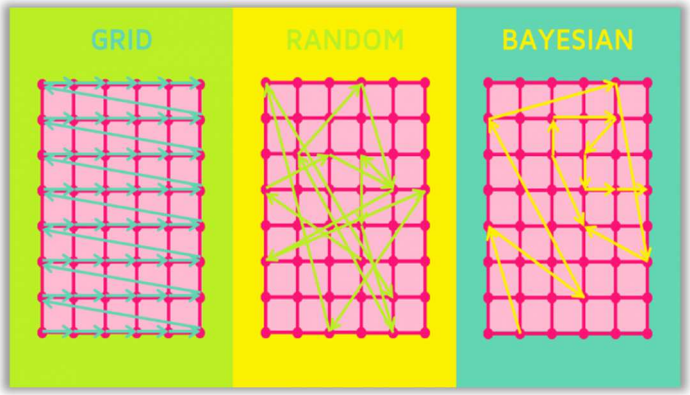
Poxa, ele viu que deu um desempenho bacana! Então agora ele vai testar uma combinação com valores próximos a esses

```
  Chip de Potência = 400cv
+ Turbina = 42mm
+ Escapamento Downpipe = 2,5””
_____________________________
= Torque de 45kgfm
```

Note que ele levou em consideração o teste anterior para reduzir alguns hiperparâmetros e aumentar outros; e, com isso, chegou a um desempenho melhor. Galera, é evidente que isso é apenas uma grande simplificação e eu não entendo quase nada de carro. O que eu quero mostrar para vocês é que um especialista externo realizou configurações no carro para permitir otimizar seu desempenho baseado em avaliações anteriores.

Ele não fez essas configurações de forma exaustiva, testando todas as possibilidades, como no Grid Search; ele também não fez testando valores aleatórios, como no Random Search. Ele se utilizou de valores de testes anteriores para realizar novos testes. A busca bayesiana é uma forma mais inteligente de testar hiperparâmetros a fim de fazer uma sintonia fina que ajuste da melhor forma possível um algoritmo de aprendizado de máquina.

A grande vantagem é que essa abordagem perde pouco tempo buscando valores de hiperparâmetros onde há pouca probabilidade de encontrá-los e se foca em realizar buscas em áreas onde há maior probabilidade. Por fim, é importante mencionar que já existem softwares capazes de buscar automaticamente quais são os melhores valores de hiperparâmetros dentro de um modelo. *Fechado?*



Há campeonatos em que os vencedores são aqueles que conseguem otimizar modelos. Imagine que uma pessoa consiga melhorar em 1% a precisão de recomendações da Netflix. Parece pouco, mas é bastante! O maior vencedor desses campeonatos no mundo todo é brasileiro Giba Titericz! Quem tiver curiosidade e quiser saber um pouquinho mais sobre ele e sobre o mundo de ciência de dados em geral não pode perder essa entrevista com o pessoal do **Let's Data**:

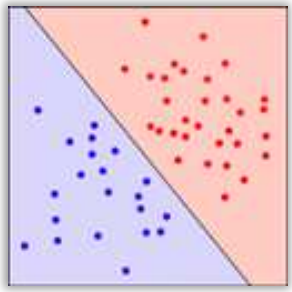
HTTPS://WWW.YOUTUBE.COM/WATCH?V=ULSOHUA7C98

Separabilidade de Dados

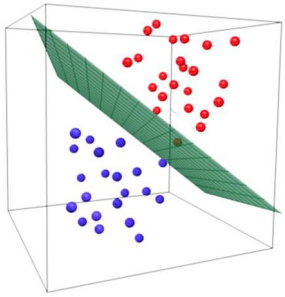
RELEVÂNCIA EM PROVA: MÉDIA

SEPARABILIDADE DE DADOS

Trata-se da capacidade de separar dados em grupos distintos e separados. É uma medida de quão bem os dados podem ser divididos em clusters, classes ou categorias distintas e uma ferramenta útil para agrupamento e algoritmos de classificação, pois a separabilidade de dados pode ajudar a identificar e classificar pontos de dados em grupos específicos.

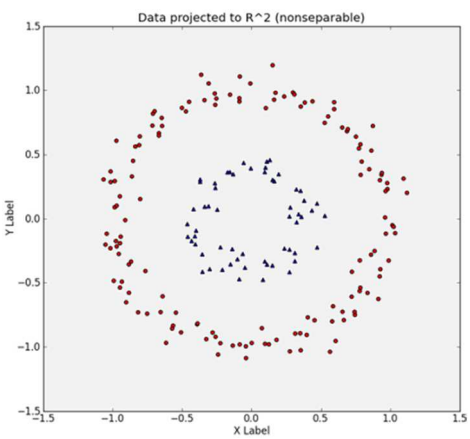


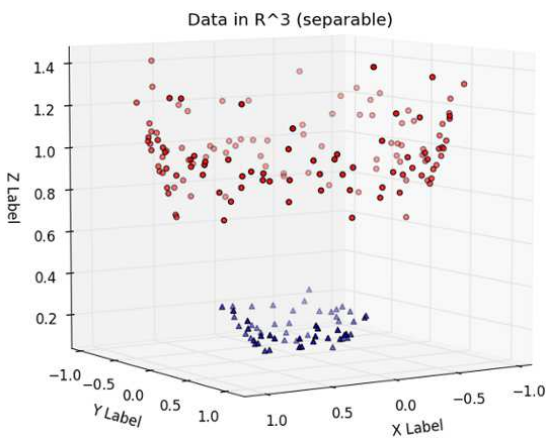
Nós já falamos um pouquinho sobre esse tema em outros tópicos, mas agora vamos detalhar um pouco mais. Em primeiro lugar, vamos assumir que estamos tratando de separabilidade **linear** de dados. Bem, nós sabemos que um modelo de aprendizado de máquina pode encontrar uma equação capaz de generalizar um problema a fim de realizar classificações. Essa equação recebe um ou mais valores (variáveis independentes) e retorna outro valor (variável dependente).



A separabilidade linear é basicamente apenas uma propriedade que existe entre dois ou mais conjuntos de pontos. Pensem em dois conjuntos de pontos, sendo um conjunto de cor azul e outro de cor vermelha. Esses dois conjuntos são linearmente separáveis se existir pelo menos uma linha no plano com todos os pontos azuis de um lado da linha e todos os pontos vermelhos do outro lado, isto é, se existe ao menos uma linha reta que separa esses dois conjuntos de pontos.

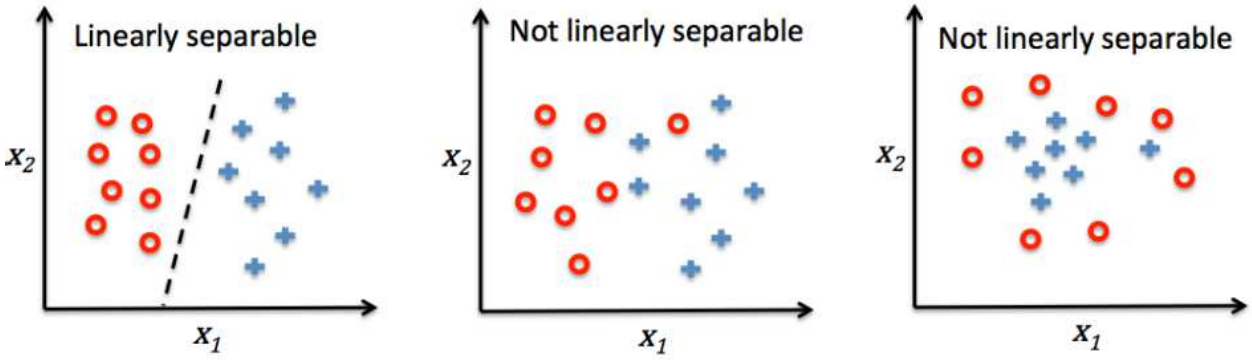
Isso é mais facilmente visualizado em duas dimensões, mas também vale para três ou mais. A separabilidade de dados em duas dimensões se dá por meio de uma linha; já em três dimensões se dá por meio de um hiperplano conforme imagem acima. Aliás, é possível existir um cenário em que há separabilidade de dados em três dimensões, mas não há separabilidade de dados em duas dimensões conforme podemos ver na imagem a seguir. *Como é, Diego?*





Note que é possível traçar um hiperplano na imagem acima da direita separando os pontos azuis dos pontos vermelhos, mas não é possível traçar uma linha reta na imagem da esquerda que consiga separar os pontos azuis dos pontos vermelhos. *E o que isso tem a ver com o aprendizado de máquina?* Os pontos azuis e vermelho são classes, isto é, são categorias resultantes de um modelo de aprendizado de máquina.

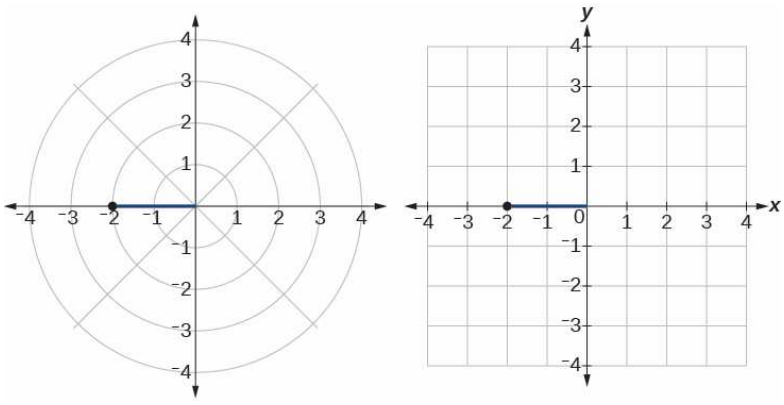
Imagine que eu queira descobrir se uma pessoa é homem ou mulher baseado em seu peso e altura. Eu posso plotar esses dados em um plano cartesiano, em que o peso seria representado no eixo das abscissas e a altura seria representada no eixo das ordenadas. Poderíamos colocar pontos azuis para representar homens e pontos vermelhos para representar mulheres. A ideia por trás da separabilidade linear é ter ao menos uma linha capaz de separar essas duas categorias.



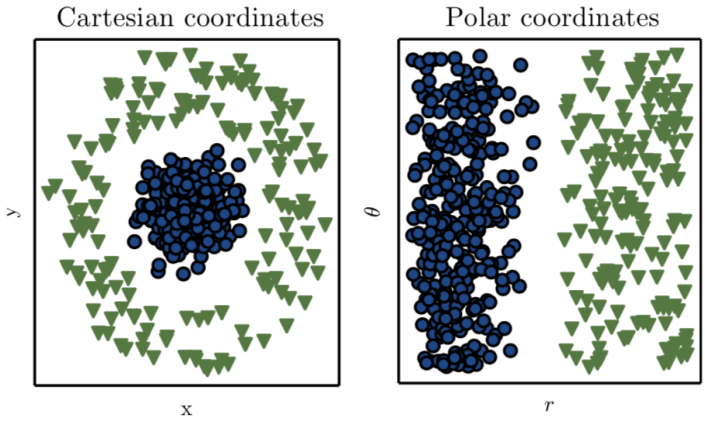
Esse problema provavelmente não permitirá que uma linha divida os dados porque há muitas pessoas com alturas medianas e pesos medianos que podem ser homem ou mulher. Por exemplo: se eu digo que uma pessoa pesa 50 quilos, é mais provável ser uma mulher; se eu digo que uma pessoa tem 2,00m de altura, é mais provável que seja um homem; mas se eu digo que uma pessoa tem 65kg e 1,67m, pode ser tanto um homem quanto uma mulher com probabilidades próximas.

Logo, o gráfico fica bastante misturado, sendo impossível traçar uma linha reta que separe os pontos de cada categoria. *O que fazer, Diego?* É possível modificar explicitamente a representação dos dados originais para um novo formato de representação em que as classes sejam mais facilmente separáveis. *Falou grego, professor!* Galera, eu disse para vocês que a separabilidade linear é basicamente apenas uma propriedade que existe entre dois ou mais conjuntos de pontos.

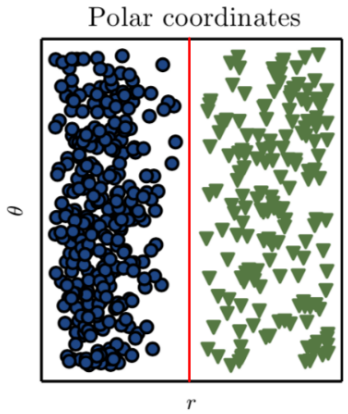
Em nenhum momento eu disse que essa propriedade só podia ser representada em um plano cartesiano. Apesar de ser disparado o mais utilizado para representação de dados em um plano bidimensional, existem outras alternativas: coordenadas polares, coordenadas cilíndricas, coordenadas esféricas, coordenadas elípticas, coordenadas parabólicas, coordenadas hiperbólicas, coordenadas parabólicas cilíndricas, entre outros. A representação dos dados é muito importante!



Querem um exemplo? Números podem ser representados por algarismos arábicos ou romanos, mas é muito mais fácil fazer conta com o primeiro do que com o segundo (Ex: tentem multiplicar III x LDI). *Querem outro exemplo?* Na escola, nós aprendemos sobre coordenadas polares, que é um sistema de coordenadas bidimensionais em que cada ponto no plano é determinado por uma distância e um ângulo em relação a um ponto fixo de referência.



Não vamos entrar em detalhes matemáticos aqui porque não há necessidade, o que importa é que vocês saibam que é possível representar um conjunto de pontos de outras maneiras. Na imagem acima, temos exatamente os mesmos dados representados em coordenadas cartesianas e em coordenadas polares. Note que é impossível traçar uma linha reta que divida as duas categorias em coordenadas cartesianas, mas é possível fazê-lo em coordenadas polares.

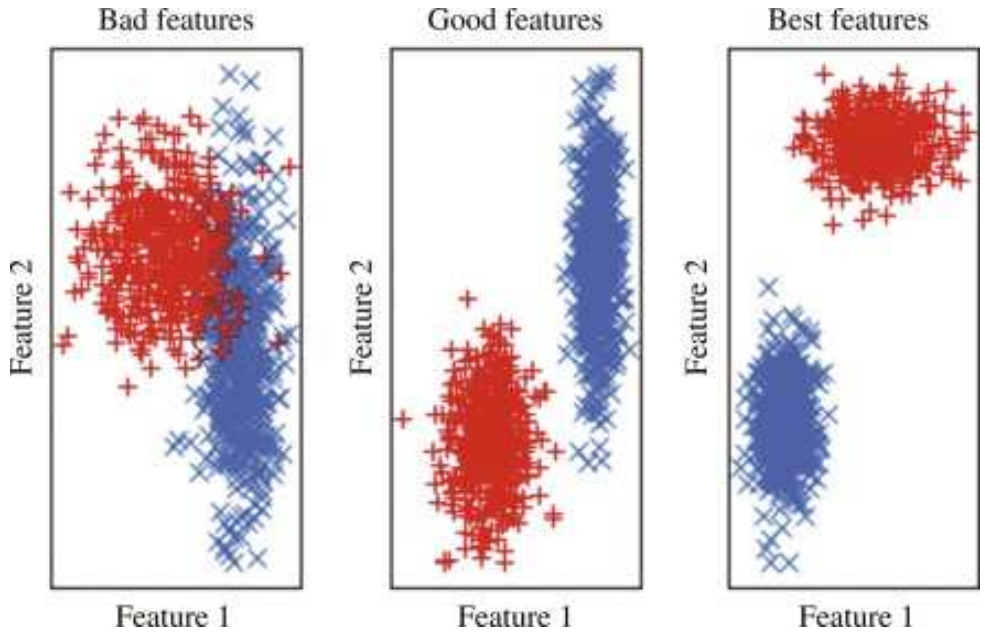


Neste simples exemplo, nós mesmos encontramos e escolhemos uma transformação (coordenadas cartesianas para polares) para obter uma melhor representação dos dados. No entanto, existem algoritmos de aprendizado de máquina capazes de fazer isso automaticamente, isto é, algoritmos capazes de buscar de forma autônoma outras representações dos dados que satisfazem o objetivo de uma tarefa de predição qualquer – essa abordagem se chama aprendizado de representação.

Outra maneira de aumentar a separabilidade para facilitar a classificação é por meio da seleção de *features* (também chamados de características, atributos ou variáveis). A ideia é selecionar *features* que permitam uma melhor separabilidade dos dados. Vamos ver um exemplo: suponha que o IBGE vá até a casa de milhares de pessoas e faça um questionário com perguntas sobre diversos assuntos, tais como: idade, renda, religião, etnia, tempo de deslocamento para o trabalho, entre outros.

Cada um desses atributos poderá ser utilizado para prever a classe social de uma pessoa. O atributo idade é ruim, porque há pessoas de todas as idades em todas as classes sociais; já o atributo tempo de deslocamento para o trabalho é bom, porque pessoas de classes sociais mais altas levam menos tempo para se deslocar até o trabalho enquanto pessoas de classes sociais mais baixas levam mais tempo para se deslocar até o trabalho.

Por fim, a renda seria um ótimo atributo dado que pessoas com renda mais baixa tendem a ser de classes sociais mais baixas enquanto pessoas com renda social mais alta tendem a ser de classes sociais mais altas (a não ser que a pessoa ganhe muito e gaste mais ainda). No exemplo abaixo, temos uma imagem que mostra a representação da escolha de atributos ruins, bons e ótimos. Vejam que quanto melhor o atributo escolhido, maior a separabilidade linear dos dados.



Redução de Dimensionalidade

RELEVÂNCIA EM PROVA: ALTA

REDUÇÃO DE DIMENSIONALIDADE

Trata-se do processo de redução do número de features em um conjunto de dados, mantendo as informações mais importantes. É usado para reduzir a complexidade de um conjunto de dados enquanto ainda preserva as características essenciais dos dados; além de ser usado no aprendizado de máquina para reduzir o overfitting, reduzir o tempo de computação e melhorar a precisão dos modelos.

Na matemática, uma reta é um exemplo de espaço unidimensional. *Por que, Diego?* Porque é um conjunto de pontos localizados em uma única dimensão – essa dimensão poderia ser o comprimento da reta. Já um plano é um exemplo de espaço bidimensional. *Por que, Diego?* Porque é um conjunto de pontos localizados em duas dimensões – essas dimensões poderiam ser o comprimento e a largura do plano.

Por fim, um cubo é um exemplo de espaço tridimensional. *Por que, Diego?* Porque é um conjunto de pontos localizados em três dimensões – essas dimensões poderiam ser o comprimento, a largura e a profundidade do cubo. Logo, podemos concluir que dimensões são características (também chamadas de *features* ou variáveis) de um conjunto de pontos. Note que, se escolhermos qualquer ponto em um cubo, teremos um valor para cada uma de suas dimensões.

Quando tratamos de mais de três dimensões, é difícil imaginar a forma de representação. No entanto, podemos abstrair como uma tabela, em que cada linha representa uma observação, cada coluna representa uma dimensão e cada célula representa o valor de cada dimensão para cada observação. Quando temos mais de três dimensões, dizemos que se trata de uma representação n-dimensional, em que *n* é a quantidade de dimensões.

Esse entendimento é semelhante no contexto de aprendizado de máquina. A dimensionalidade é o conjunto de características ou variáveis associadas a cada uma das observações em um conjunto de dados. Vamos imaginar que desejamos prever o valor de um imóvel com base em diversas de suas características: área total, tipo, bairro, número de quartos, número de banheiros, número de vagas de garagem, entre outros.

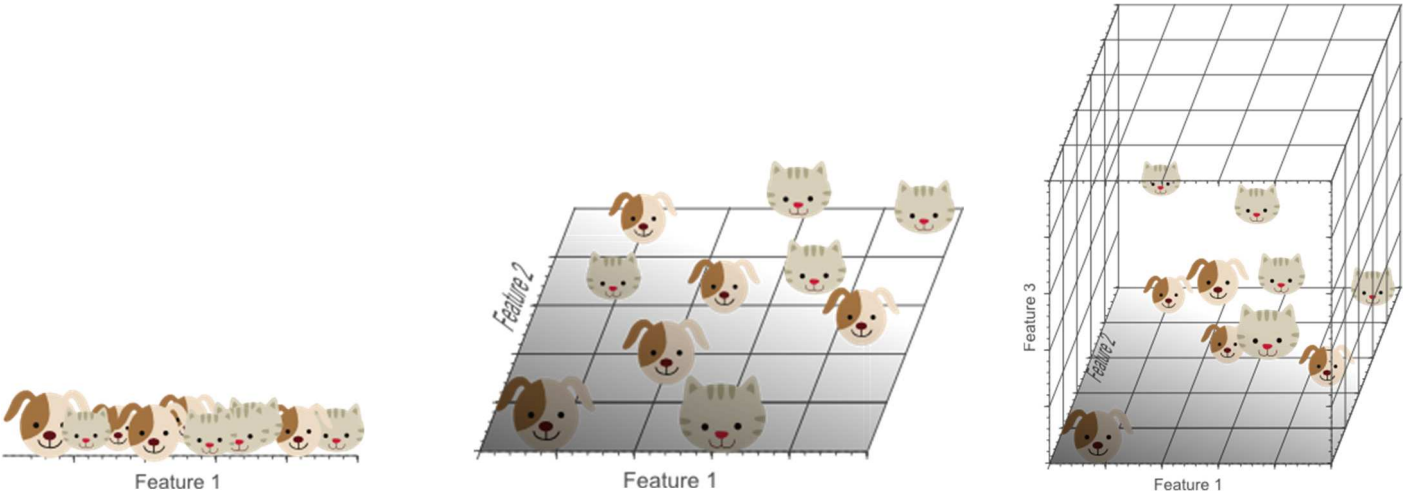
Cada uma dessas características são variáveis que podem ser utilizadas para prever o valor de um imóvel. Nesse caso, teremos uma tabela em que cada linha representa uma observação (ou ponto) e cada coluna representa o valor de uma variável (ou dimensão). Galera, é muito comum a crença de que – quanto maior o número de variáveis – melhor será a previsão, mas nem sempre essa crença é uma realidade.

Nós já vimos que, se continuarmos aumentando o número de variáveis durante o processo de treinamento de um modelo de aprendizado de máquina, após um certo ponto, o desempenho

tende a diminuir. Ocasionalmente, a introdução de muitas variáveis pode resultar em *overfitting*. Ademais, é preciso um espaço muito maior para armazenar dados com um grande número de dimensões, além de ser mais difícil de analisar e visualizar.

Aliás, o processo de visualização é conhecido como interpretação geométrica de um conjunto de dados. Para entender isso melhor, vamos imaginar que uma barata percorra uma linha reta de 100m e você deve tentar encontrá-la. *É difícil?* Não, rapidinho você anda esses 100m e encontra a barata. Se adicionarmos mais uma dimensão, agora temos uma área plana de 100m² – próximo ao tamanho de um campo de futebol. Opa, agora ficou bem mais complexo de encontrar a barata!

Vamos piorar um pouco mais adicionando mais uma dimensão, contemplando um espaço cúbico de 100m³ – como se fosse um galpão, sendo que a barata pode estar no chão, nas paredes ou no teto. Agora complicou de vez: ficou muito mais difícil encontrar a barata. Quanto mais dimensões existirem, maior será o volume espacial, mais esparsos ficam os dados. Em altas dimensões, todos os conjuntos de dados são esparsos, isto é, são poucos dados em um espaço muito grande.



Imagine que eu queira prever a expectativa de vida de um cão ou gato baseado em 10 amostras de dados de treinamento. Se eu escolher uma característica (ex: peso), poderei representá-la por meio de uma linha, logo teremos os pontos de dados juntinhos como mostra a primeira imagem; se eu escolher duas características (ex: peso e altura), poderei representá-las por meio de um plano, mas veja que os pontos de dados ficam mais esparso como mostra a segunda imagem.

Por fim, se eu escolher três características (ex: peso, altura e nível de atividade), poderei representá-las por meio de um cubo, mas veja que agora os dados ficaram bem mais esparsos ainda como mostra a terceira imagem. Note que, quanto mais aumentamos a quantidade de dimensões, mais esparsos ficam os dados de gatos e cães dado que a amostra não foi modificada. Logo, como os dados estão mais esparsos, torna-se mais difícil para o algoritmo encontrar padrões.

Durante o treinamento de um modelo, uma parte dos dados disponíveis são utilizados para treinamento e outra parte para teste. Uma maneira eficaz de construir um modelo que generaliza bem os dados é capturar diferentes combinações possíveis dos valores das variáveis. Ocorre que,

quanto mais dimensões nós temos, mais combinações devem ser analisadas e mais dados de treinamento são necessários; caso contrário, o modelo não generalizará bem.

O nome do conjunto de fenômenos que surgem quando analisamos e organizamos dados em espaços de alta dimensionalidade é Maldição da Dimensionalidade (Curse of Dimensionality). *O que fazer para lidar com esses problemas, professor?* Podemos realizar a redução de dimensionalidades, isto é, utilizar um conjunto de técnicas que reduzem o número de variáveis de entrada em um conjunto de dados. A redução de dimensionalidade possui algumas vantagens:

VANTAGENS DA REDUÇÃO DE DIMENSIONALIDADE
Simplificação dos modelos de aprendizado de máquina: é mais fácil ajustar um modelo que tenha duas variáveis de entrada do que um modelo que tenha 80 variáveis de entrada.
Redução do *overfitting*: é muito mais difícil ocorrer sobreajuste em um modelo de aprendizado de máquina com menos variáveis do que com muitas variáveis.
Simplificação da representação gráfica: visualizar dados representados em mais de três dimensões é inviável para seres humanos.
Redução do custo computacional: com menos variáveis, é necessário utilizar menos recursos computacionais para realizar o treinamento de um modelo de aprendizado de máquina.
Redução do tempo de treinamento: como há menos variáveis para ajustar, leva menos tempo para treinar o modelo de aprendizado de máquina.
Aumentar a performance: como há variáveis com nenhuma correlação, sua eliminação ajuda a melhorar o desempenho do modelo de aprendizado de máquina.

Tipos de Métodos

Existem dois tipos de métodos de redução de dimensionalidade: seleção de variáveis e fatorização de matrizes. Vamos conhecê-los melhor...

Seleção de Variáveis/Atributos

SELEÇÃO DE VARIÁVEIS

Trata-se do processo de selecionar um subconjunto de variáveis relevantes para uso na construção do modelo, isto é, aquelas que têm o relacionamento mais forte com a variável dependente e também que são mais úteis para a previsão. Busca-se selecionar aquelas que possuem maior poder preditivo, descartando as demais. Essa técnica é importante porque pode ajudar a reduzir a complexidade dos modelos e melhorar a precisão das previsões.

A seleção de variáveis tem como objetivo descobrir um subconjunto de variáveis relevantes para uma tarefa. Ela é importante, entre outras coisas, por tornar o processo de aprendizagem mais eficiente. A ideia aqui é descartar variáveis irrelevantes, que sejam pouco relacionadas com a variável que desejamos prever. *Vocês se lembram do exemplo do apartamento?* Ora, existe pouca (ou nenhuma) correlação entre a quantidade de torneiras e o valor de um apartamento.

Em outras palavras, os dados indicam que essa variável não tem nenhuma relação com o valor do apartamento. Logo, ela é irrelevante e poderia ser tranquilamente eliminada do conjunto de variáveis utilizadas no aprendizado de máquina, reduzindo a dimensionalidade do nosso modelo. Há também casos em que existe correlação, mas não existe causalidade, isto é, os dados indicam que apartamentos anunciados em quartas-feiras possuem um valor maior.

Ora, pode até existir essa correlação, mas não existe causalidade! Em outras palavras, os valores mais altos não são devidos ao dia da semana em que o apartamento foi avaliado – trata-se apenas de uma coincidência. Por fim, há também as variáveis redundantes, isto é, duas ou mais variáveis com semânticas semelhantes cujos dados são parecidos. Isso prejudica o desempenho do algoritmo tanto em relação ao custo computacional quanto em relação à taxa de acerto.

Informações redundantes podem confundir o algoritmo, ao invés de auxiliá-lo na busca de um modelo ajustado para o problema. Por exemplo: número de quartos e número de dormitórios são basicamente duas variáveis redundantes, dado que basicamente qualquer quarto pode ser um dormitório. Esse tipo de técnica parece fácil, mas imaginem um cenário em que tenhamos 250 variáveis – não é tão intuitivo selecioná-las. Para tal, temos algumas técnicas...

Método Filter

MÉTODO FILTER

Trata-se de uma técnica de seleção de variáveis que usa várias métricas estatísticas para avaliar a importância dos recursos e selecionar os recursos mais relevantes para a construção de um modelo preditivo. Ela ajuda a reduzir a dimensionalidade dos dados para melhorar a precisão do modelo. Em geral, é usado em combinação com outras técnicas.

Ele método conduz uma análise inicial supervisionada das variáveis a fim de selecionar o melhor subconjunto de variáveis que sejam relevantes para a predição. Para avaliar quais variáveis são relevantes, é necessário utilizar alguma métrica de correlação, tais como variância, qui-quadrado ou coeficiente de correlação. Basicamente, essa avaliação analisará a correlação entre uma ou mais variáveis em relação ao valor que desejamos prever.

Em seguida, podemos reduzir a dimensionalidade do nosso modelo ao filtrar, descartar ou eliminar aquelas variáveis menos relevantes, isto é, que possuem pouca correlação com a variável alvo. Após isso, basta executar o algoritmo de aprendizado de máquina. Esse método pode ser utilizado com diferentes tipos de modelos, tais como classificação, regressão, entre outros. Ele é bastante simples, rápido e robusto.

Por outro lado, eles tendem a selecionar variáveis redundantes dado que não consideram as relações entre variáveis. Em outras palavras, eles consideram apenas as relações entre variável independente e variável alvo, mas não consideram as relações entre as próprias variáveis independentes. *Vocês se lembram do exemplo do quarto e dormitório?* Ambos têm alta correlação com o valor do apartamento e ambos seriam preservados – apesar de serem redundantes.

Idealmente, apenas uma delas deveria ser preservada e a outra eliminada. Apesar disso, atualmente já existem algumas técnicas para acabar com esses problemas.

Método Wrapper

MÉTODO WRAPPER

Trata-se de uma técnica de seleção de variáveis que avalia o conjunto de features escolhido otimizando uma métrica de desempenho. Ele funciona selecionando subconjuntos de features e, em seguida, avaliando o desempenho de um modelo treinado nesses recursos. O objetivo é encontrar o conjunto de features que produz o modelo de melhor desempenho.

Esse método busca empacotar (*wrap*) um problema de seleção de variáveis em uma caixa preta. Dentro dessa caixa preta, são treinados modelos de aprendizado de máquina. Em outras palavras, esse método executará diversos testes de treinamento de modelos com as variáveis a fim de encontrar o melhor subconjunto de variáveis. Na prática, ele inicialmente realiza o treinamento do modelo considerando todas as variáveis e analisa o desempenho.

Em seguida, ele descarta alguma variável, realiza o treinamento novamente e verifica se o desempenho melhorou. Ele realiza esse procedimento iterativamente até alcançar um conjunto de variáveis com um desempenho preditivo melhor que o conjunto de variáveis original. Logo, esse método considera as relações entre as variáveis independentes, sendo capaz de descartar variáveis redundantes. Por outro lado, esse método consome muito tempo e pode resultar em *overfitting*.

Método Embedded

MÉTODO EMBEDDED

Trata-se de uma técnica de seleção de variáveis que combina as qualidades do método filter e do método wrapper para treinar um classificador em um subconjunto de features e, em seguida, adicionar recursos adicionais ao modelo conforme necessário. O algoritmo usará a melhor combinação de features para gerar as previsões mais precisas.

Esse método busca combinar as vantagens dos métodos anteriores. Logo, ele tenta selecionar variáveis e executar o treinamento de aprendizado de máquina simultaneamente.

Fatoração de Matrizes

Decompõe a matriz dos dados originais em produtos de matrizes mais simples, com propriedades que permitem identificar as dimensões mais relevantes da variabilidade dos dados enquanto combinações lineares dos dados originais. *É o que, Diego?* Lá vou eu te mostrar um exemplo para

ficar fácil de entender. Vamos tentar descobrir como a Netflix faz recomendações de filmes para os seus usuários. Venham comigo...

FATORAÇÃO DE MATRIZES

Trata-se de uma técnica utilizada para fatorar a matriz original em outras matrizes menores a fim de encontrar o melhor subconjunto de dados com menor dimensionalidade que seja capaz de representar a matriz original. Ela pode ser usada para recomendar itens aos usuários, agrupar itens semelhantes e reduzir a dimensionalidade de um conjunto de dados.

Vamos pensar em quatro usuários: Antônio, Bernadete, Clemente e Doralice! Digamos que nossos usuários assistiram cinco filmes e atribuíram uma nota de classificação de 1 a 5 estrelas para cada um deles. Nós vamos registrar esses valores de classificação em uma tabela, onde as linhas representam os usuários, as colunas representam os filmes e as células representas as notas que cada usuário atribuiu para cada filme.

TABELA 1	FILME 1	FILME 2	FILME 3	FILME 4	FILME 5
ANTÔNIO	3	3	3	3	3
BERNADETE	3	3	3	3	3
CLEMENTE	3	3	3	3	3
DORALICE	3	3	3	3	3

Agora eu gostaria de convidá-los para uma reflexão: *a tabela anterior parece refletir a tabela real de classificações de filmes da NetFlix?* Ora, todos os usuários deram nota 3 para todos os filmes – isso não me parece muito realista. Seria o equivalente a dizer que todos os usuários têm exatamente as mesmos preferências de filmes e que todos os filmes teriam a mesma nota independentemente de quem avaliou. Não me parece nada com a realidade...

TABELA 3	FILME 1	FILME 2	FILME 3	FILME 4	FILME 5
ANTÔNIO	1	3	2	5	4
BERNADETE	2	1	1	1	5
CLEMENTE	3	2	3	1	5
DORALICE	2	4	1	5	2

Já a tabela acima parece ter quase todos os números aleatórios. Ora, isso também não me parece realista – seria o equivalente a dizer que todos os usuários têm preferências de filmes completamente diferentes e que todos os filmes são diferentes em termos de classificação. Em geral, pessoas podem ter gostos diferentes, mas é raro ter notas tão díspares uma das outras em termos de classificação.

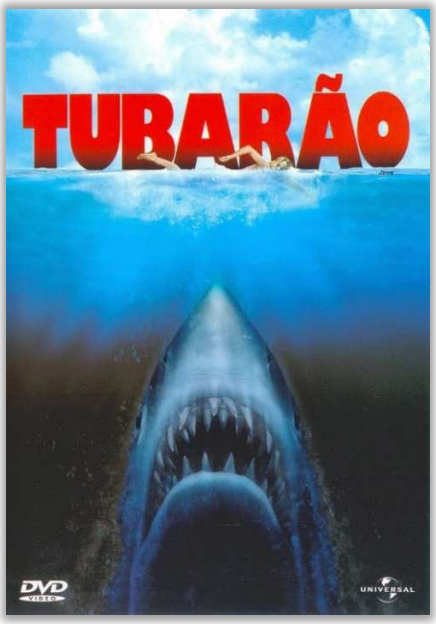
TABELA 2	FILME 1	FILME 2	FILME 3	FILME 4	FILME 5

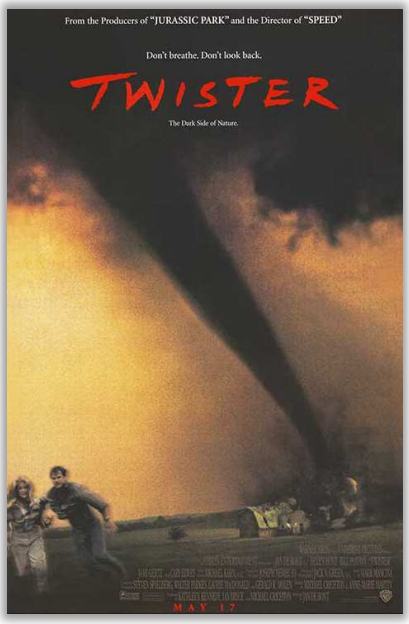
ANTÔNIO	3	1	1	3	1
BERNADETE	1	2	4	1	3
CLEMENTE	3	1	1	3	1
DORALICE	4	3	5	4	4

Agora a tabela apresentada acima contém algumas particularidades interessantes. Em princípio, ela parece aleatória como a tabela anterior, mas – se observarmos bem – podemos notar que ela possui algumas dependências bastante peculiares. Em primeiro lugar, note que a primeira e a terceira linhas são exatamente iguais – isso significa que Antônio e Clemente possuem gostos de filme bastante similares. Logo, para NetFlix, eles são tratados como semelhantes.

Note também que a primeira e quarta colunas são exatamente iguais – isso significa que o primeiro e o quarto filmes são bastante similares, ao menos em termos de classificação. Logo, para NetFlix, eles também são tratados como semelhantes. Há ainda algumas dependências ocultas que são mais sofisticadas. Por exemplo: ao analisar as três últimas linhas, podemos observar que as notas da quarta linha são exatamente a soma da segunda e terceira linhas.

E o que isso tem a ver, Diego? Uma possível interpretação que poderíamos especular seria a de que Bernadete gosta de filmes de comédia, Clemente gosta de filmes de romance e Doralice gosta de filmes de comédia romântica. Agora vejam a segunda, terceira e quinta colunas: observe que as notas da quinta coluna são exatamente a média entre as notas da segunda e terceira colunas. *Qual seria uma possível interpretação?*





Poderíamos especular que o segundo filme seja Tubarão – aquele filme clássico que trata de ataques de tubarões em uma praia americana; o terceiro filme seja Twister – aquele filme clássico que trata de um tornado que destrói uma cidade inteira; e o quinto filme seja Sharknado – aquele

filme em que um tornado espalha tubarões nas águas de uma cidade alagada. Se você gostou de filmes de ataques de tubarões e filmes de tornados, gostará de filmes que misturam os dois.

Galera, é claro que tudo isso é especulação – nem sempre nós vamos conseguir interpretar o significado de um determinado padrão. O ponto aqui foi apenas demonstrar duas coisas: (1) não só é possível como também é comum armazenar dados sobre variáveis de um modelo em formato de matrizes; (2) é possível encontrar dependências e relacionamentos entre linhas e colunas de uma matriz. *E qual é a grande vantagem disso?*

A vantagem é que matrizes são estruturas que podem ser manipuladas por meio de técnicas de álgebra linear para terem sua dimensionalidade reduzida. *E como fazemos para descobrir todas essas dependências em uma matriz de alta dimensionalidade?* Resposta: por meio de métodos de fatoração ou decomposição de matrizes. Quando crianças, nós aprendemos nas aulas de matemática que um número grande pode ser fatorado ou decomposto em dois ou mais números menores.

Por exemplo: 512 é um número grande, mas eu posso decompô-lo por meio de números menos, tais como 256 x 2 ou 128 x 4 ou 64 x 8. A fatorização de matrizes busca fazer um procedimento semelhante, isto é, representar uma matriz de alta dimensionalidade como o produto entre matrizes de baixa dimensionalidade. Existe uma matemática extraordinariamente complexa por trás desses métodos, logo não vamos detalhar.

A ideia aqui é apenas mostrar que essa técnica permite encontrar subconjuntos de dados que melhor representam a matriz original. No contexto do aprendizado de máquina, é como se a máquina tentasse fatorar a matriz original iterativamente até encontrar o melhor subconjunto de dados com menor dimensionalidade que seja capaz de representar a matriz original. Uma outra comparação interessante é entre população e amostra.

Um laboratório não precisa retirar todo seu sangue para inferir com algum grau de confiança se você está com alguma doença – basta selecionar uma amostra representativa da população total.

Sistemas de Recomendação

SISTEMAS DE RECOMENDAÇÃO

Trata-se de um algoritmo que faz previsões sobre os interesses de um usuário. Esses sistemas usam dados históricos do usuário e dados de itens específicos para fazer recomendações aos usuários sobre itens nos quais eles podem estar interessados. O objetivo de um sistema de recomendação é fornecer recomendações personalizadas e relevantes aos usuários.

Bem, a fatoração de matrizes é a principal base para criação de sistemas de recomendação. O objetivo é encontrar padrões ocultos em uma matriz de dados para prever o que os usuários vão gostar ou não. Ela faz isso descompondo a matriz em submatrizes menores, cada uma

representando um grupo de fatores ocultos que influenciam a escolha do usuário. Então, usa-se a informação contida nessas submatrizes para prever quais produtos o usuário pode gostar.

Existem três modelos de sistemas de recomendação: (1) baseado em filtragem colaborativa; (2) baseado em conteúdo; e (3) híbrido. No primeiro modelo, os algoritmos usam aprendizado de máquina para prever seus gostos com base em usuários que têm perfis similares. Por exemplo: suponha que João ouve com frequência em um serviço de *streaming* de músicas as bandas *Pink Floyd*, *Metallica* e *Blind Guardian*.

Suponha também que Alice assina a mesma plataforma de *streaming* e escuta com frequência *Pink Floyd*, *Metallica* e *Symphony X*, o serviço poderá recomendar *Blind Guardian* para Ana. Em sistemas mais complexos, como o de recomendações de filmes e livros, mais pontos além de uma interação genérica devem ser levados em consideração — aqui podem entrar a avaliação que o usuário dá para o conteúdo ou tempo médio de consumo, entre outros.

No segundo modelo, o sistema se baseia em características de um conteúdo, sem depender necessariamente de uma interação de outro usuário. Nesse caso, as indicações são baseadas em atributos dos itens recomendados, e o mecanismo cria uma espécie de perfil genérico de usuário que deve se interessar por temas semelhantes. Por exemplo: Alice assistiu todos os episódios da série médica Grey's Anatomy na NetFlix.

A plataforma pode identificar as características dessa série (Ex: trata-se de uma série de drama, passada nos EUA que fala sobre o dia a dia de médicos) e pode recomendar outra série que possua atributos similares (Ex: House). Por fim, o terceiro modelo é o mais utilizado e se fundamenta tanto nas características de similaridade de perfis quanto nas características de similaridade de conteúdo. *Sabe quem tem o melhor sistema de recomendação do mercado atualmente?* TikTok!

Eu não uso essa rede social, mas ela é um caso de sucesso do algoritmo. Assim como outras big techs, o TikTok usa inteligência artificial para sugerir seu conteúdo a outros usuários. Sua particularidade é que a recomendação está sempre mostrando algo novo em uma velocidade absurda (na aba For You) — mas sem perder a relevância. Para isso, o TikTok usa dados referentes às suas interações, hashtags, áudios, localização e legenda de cada vídeo.

O algortimo de recomendação do TikTok deu tão certo que virou um produto, e agora pode ser comercializado para outras empresas por meio de uma unidade de negócios chamada BytePlus. Outro caso de sucesso são os reels do Instagram! Eu que o diga! Começo a ver vídeos de cachorro e passo uma hora descendo aquele troço e não para de surgir (ótimos) vídeos novos de cachorro. Vejamos agora os principais métodos...

Principais Métodos

Os principais métodos de redução de dimensionalidade são: PCA (*Principal Component Analysis*), t-SNE e MDS (Multi-Dimensional Scaling), mas aqui vamos nos ater apenas ao primeiro...

PCA

PCA (PRINCIPAL COMPONENT ANALYSIS)

Trata-se de um método estatístico usado para reduzir a dimensionalidade dos dados por meio da transformação de um grande conjunto de variáveis em um conjunto menor de variáveis, chamado de Componentes Principais, enquanto retém o máximo possível da variação dos dados. O PCA é amplamente para interpretar as características mais importantes de um conjunto de dados.

A Análise de Componentes Principais busca encontrar os componentes que explicam a maior variação nos dados. Ele faz isso transformando os dados originais em um novo conjunto de variáveis chamado *componentes principais*. Cada componente principal é uma combinação linear de todas as variáveis originais, independentes entre si, e buscam capturar o máximo de informação em termos de variação nos dados. Para entender isso melhor, vamos ver um exemplo...

Vamos supor que você more em Maragogi/AL e, de repente, ganhou na mega-sena, largou a vida de concurseiro e decidiu se mudar para alguma cobertura na frente da praia. Só que você está milionário, então – apesar de você amar Maragogi – agora você tem a possibilidade de morar em qualquer praia do planeta. Você decide, portanto, pesquisar cidades parecidas com Maragogi em outros lugares do mundo.

Logo, você começa a coletar dados de centenas de outras cidades praianas, tais como: número de habitantes, temperatura média, nível de trânsito, densidade populacional, área de espaço verde, altitude, nível de segurança, nível médio de escolaridade, renda *per capita*, IDH, proximidade de aeroportos internacionais, entre outros. Você, então, decide desenhar um gráfico unidimensional com apenas a população de cada cidade. *Qual será o resultado disso?*

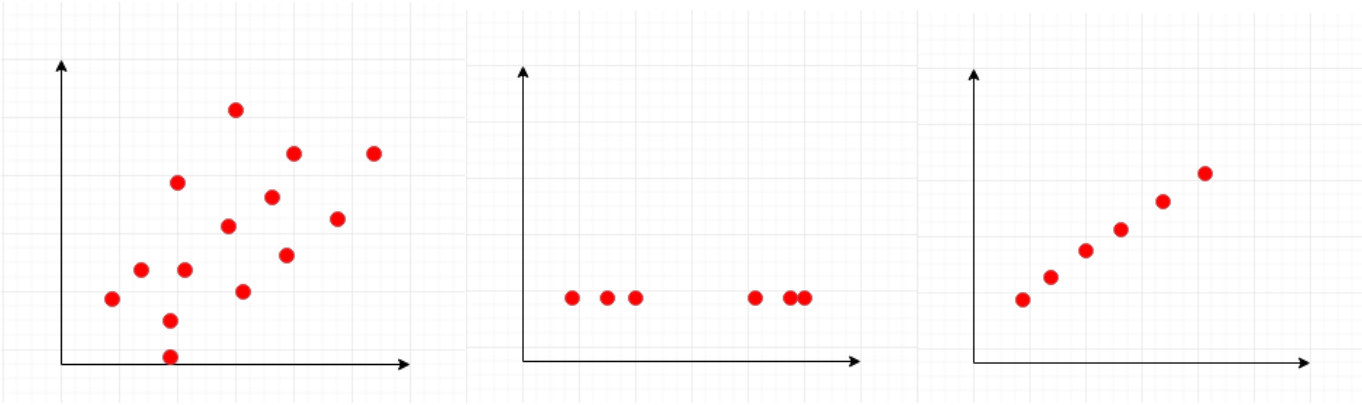
Acho que é intuitivo pensar que cidades grandes ficarão próximas de um lado do gráfico e cidades pequenas também ficarão próximas, mas do outro lado do gráfico (e Maragogi provavelmente ficaria próximo das cidades pequenas). Você não fica satisfeito e adiciona mais uma dimensão, desenhando um gráfico bidimensional de População x Temperatura Média. Agora você tem cidades grandes quentes, cidades grandes frias, cidades pequenas quentes e cidades pequenas frias.

É claro que há cidades que ficam no meio termo dessas características, mas isso não é relevante no momento. Ocorre que estar próximo a um aeroporto internacional é muito importante para você, dado que agora você poderá viajar sempre que quiser, então você adiciona mais uma dimensão, desenhando um gráfico tridimensional de População x Temperatura Média x Proximidade de Aeroportos Internacionais. E assim você prossegue adicionando mais e mais e mais dimensões!

No entanto, você notará que algumas dessas variáveis são bastante correlacionadas. *Como assim, Diego?* Ora, cidades praianas com maior nível de escolaridade geralmente tem maior renda *per capita*, maior IDH e maior nível de segurança; cidades com maior quantidade de habitantes

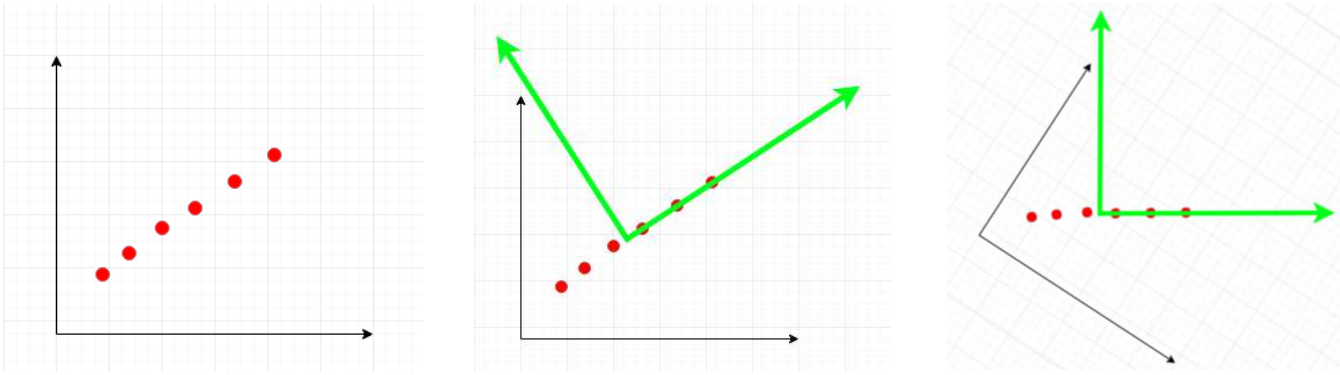
geralmente também têm maior densidade populacional; cidades com maior densidade populacional geralmente tem menos área verde e mais trânsito; e por aí vai...

É nesse momento que nosso método chega para brilhar! O PCA aplicará uma transformação linear sobre as variáveis originais e criará um novo sistema de coordenadas bidimensional de modo que Eixo X será o *primeiro componente principal* e o Eixo Y será o *segundo componente principal*. Eu sei, eu sei que você viajou agora, mas vou tentar desenhar um pouco e abstrair a parte matemática porque isso envolve diversos conceitos de álgebra linear. Vamos lá...



Observem os três gráficos apresentados e respondam: *o primeiro gráfico tem quantas dimensões?* Duas, professor! Exato, trata-se de gráfico 2D (bidimensional). *E o segundo gráfico?* Podemos dizer que ele tem apenas uma dimensão (unidimensional) porque, para qualquer valor de *x*, o y tem o mesmo valor. *Legal, e o terceiro gráfico?* Olha a pegadinha: eu posso dizer que ele tem apenas uma dimensão. *O que, Diego?*

Galera, se eu rotacionar os eixos um pouquinho, os pontos vão se alinhar exatamente como no segundo gráfico. Vejam só...



De forma bastante abstrata, isso é o que chamamos de transformação linear. *Diego, se você fizer isso, você não estará mexendo nos dados do gráfico?* Galera, a correlação entre as variáveis originais se perderá, mas não haverá perda de informação porque a rotação dos eixos não altera a distância relativa que um ponto tem do outro – que é o importante para nós nesse momento. Aqui estamos preocupados com a variância, isto é, queremos enfatizar a variação dos dados.

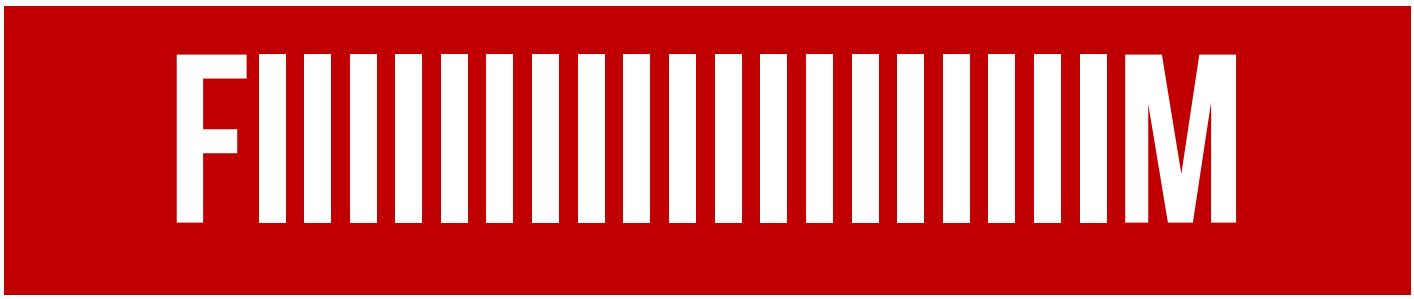
Como modificamos os eixos, as variáveis originais já não fazem mais sentido no novo gráfico. No gráfico gerado, o Eixo X será o *primeiro componente principal* e o Eixo Y será o *segundo componente principal*. É bom enfatizar também que – da mesma forma que as variáveis originais eram correlacionadas – os novos componentes principais são completamente não correlacionados (também chamado de ortogonais) – que era tudo que nós queríamos.

Nós queríamos isso, Diego? Sim! O que nos interessa é ver a variação de dados! Logo, é comum configurar um limiar de variância. Por exemplo: queremos um novo gráfico que represente 90% da variância dos dados (perda de 10%). O PCA fará seus cálculos e retornará um gráfico que contemple essa variância e com uma determinada quantidade de componentes principais (que poderá ser igual ou, geralmente, menor que a quantidade de variáveis originais).

Em nosso exemplo, tínhamos um caso muito simples. De toda forma, eliminamos uma variável/dimensão (2D → 1D), não perdemos informações relevantes e representamos em uma única dimensão a variabilidade dos dados (variância). No entanto, no mundo real os sistemas são bem mais complexos e com bem mais variáveis. Logo, a redução de dimensionalidade possibilita o desenvolvimento de modelos mais simples, o que permite melhor análise.

O PCA converte as correlações (diretas ou inversas) entre variáveis de forma que variáveis altamente correlacionadas fiquem agrupadas e representem padrões fortes de conjuntos de dados grandes e complexos. O primeiro componente principal é mais importante (representa maior variância dos dados) que o segundo componente principal, e assim por diante. Isto é, há uma sequência de componentes principais ortogonais entre si e em ordem decrescente de importância.

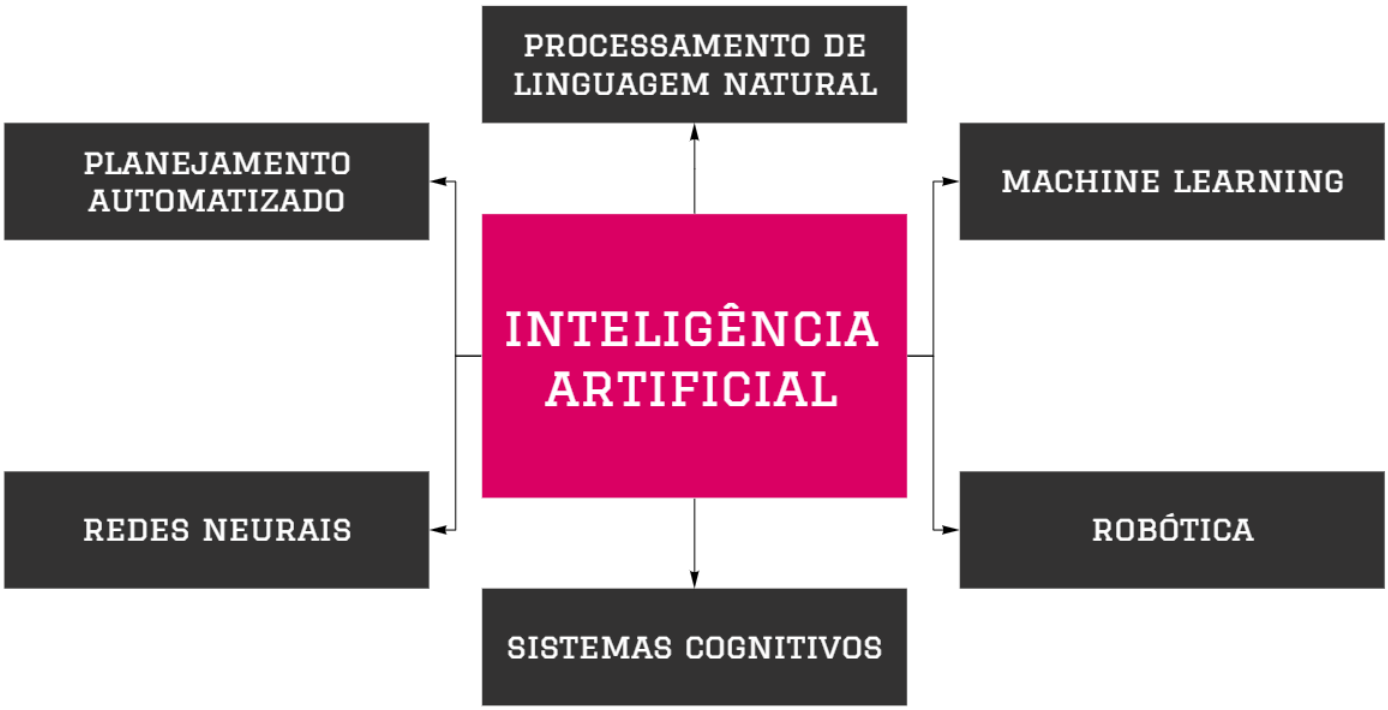
Notem que os componentes principais buscam capturar a essência dos dados, isto é, aqueles dados que representam a maior variação no conjunto. Ele aplica uma transformação em um conjunto de dados com variáveis possivelmente correlacionadas, para convertê-las em um conjunto de dados de variáveis linearmente não correlacionadas. Era isso que eu tinha para dizer sobre a análise de componentes principais. Dito isso, acho que chegamos ao...

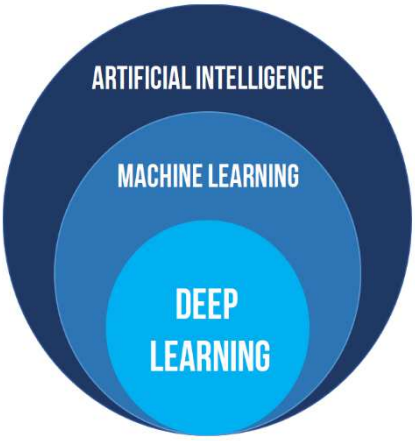


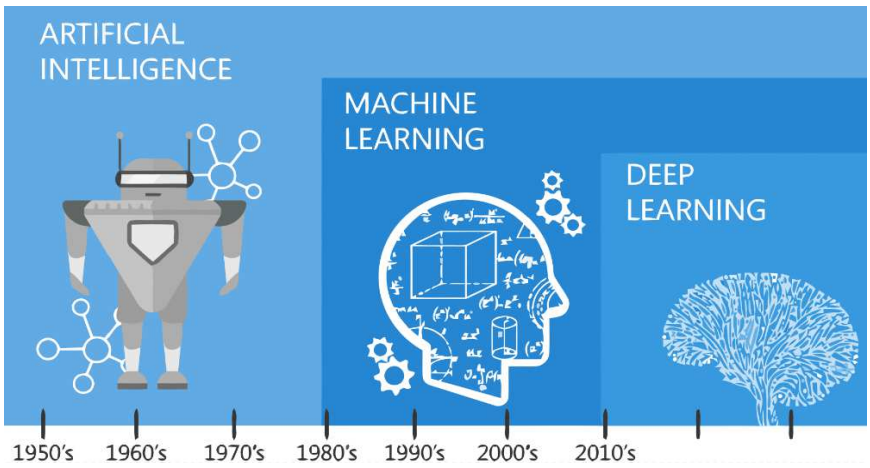
EU NÃO SEI QUEM SOFREU MAIS NESSA AULA: EU, QUE TIVE QUE ESCREVER; OU VOCÊS, QUE TÊM QUE ESTUDAR!

RESUMO

INTELIGENCIA ARTIFICIAL







MACHINE LEARNING

MACHINE LEARNING

SUBCAMPO DA INTELIGÊNCIA ARTIFICIAL QUE FORNECE AOS COMPUTADORES A HABILIDADE DE APRENDER SEM SEREM EXPLICITAMENTE PROGRAMADOS

- PROCESSO EM QUE O DESEMPENHO DE UMA TAREFA AUMENTE COM A EXPERIÊNCIA EXTRAÍDA DE NOVOS DADOS
- A MÁQUINA PODE APRENDER A PARTIR DE SEUS ERROS E FAZER INFERÊNCIAS SOBRE DADOS
- DIFERE-SE DA PROGRAMAÇÃO TRADICIONAL POR GERAR REGRAS A PARTIR DE RESULTADOS E, NÃO, O CONTRÁRIO
- MODELOS SÃO REGRAS ESTATÍSTICAS NA FORMA DE MODELOS MATEMÁTICOS E PARÂMETROS
- A ETAPA DE TREINAMENTO É MAIS CUSTOSA E A ETAPA DE INFERÊNCIA É MENOS CUSTOSA

TERMO	DESCRIÇÃO
TAREFA	Definição genérica daquilo que se deseja produzir como resultado do modelo preditivo. Ex: classificar um documento em três possíveis categorias ou prever o valor de determinada medida.
TÉCNICA	Conjunto de procedimentos que permite melhorar resultados preditivos. Ex: regularização é uma técnica para prevenir o *overfitting*; *hold-out* é uma técnica de separação de dados para medir o desempenho em generalização de um modelo.
ALGORITMO	Fórmula no sentido lato, que permite relacionar as variáveis independentes para prever a variável dependente. Quando aplicamos (ou treinamos) um algoritmo, temos um modelo treinado. Exemplo: Regressão Linear ou Árvores de Decisão.
MODELO (TREINADO)	Objeto computacional que efetivamente transforma uma observação (variáveis independentes) em uma previsão utilizando um algoritmo específico, instanciado e treinado.

TIPOS DE APRENDIZADO

APRENDIZADO SUPERVISIONADO

Trata-se da abordagem que busca encontrar um conjunto de regras ou funções (também chamadas de modelo) a partir dos dados de treinamento que possam ser utilizadas para prever um rótulo ou valor que caracterize um novo exemplo, com base nos valores de seus atributos de entrada. O termo supervisionado vem da presença de um supervisor externo, que conhece a saída (rótulo) desejada para cada exemplo. Com isso, ele pode avaliar a capacidade do algoritmo de predizer o valor de saída para novos exemplos.

APRENDIZADO NÃO SUPERVISIONADO

Trata-se da abordagem em que o algoritmo busca encontrar um padrão subjacente nos dados sem a utilização de um supervisor externo para atribuir rótulos ou categorias pré-definidas para as amostras de treinamento. Os algoritmos são formulados de forma que possam encontrar padrões autonomamente com o intuito de explorar dados desconhecidos e encontrar estruturas interessantes ou ocultas nos dados que não eram visíveis anteriormente para os cientistas de dados.

APRENDIZADO SEMISUPERVISIONADO

Trata-se da abordagem que abrange técnicas de aprendizado de máquina que usam um conjunto de dados parcialmente rotulado. Este conjunto de dados inclui alguns dados com rótulos e alguns dados sem rótulos. As técnicas de aprendizado semi-supervisionado são usadas para combinar dados rotulados e não rotulados para melhorar a precisão do modelo.

APRENDIZADO POR REFORÇO

Trata-se da abordagem que se baseia na recompensa e na punição para ensinar a máquina a realizar tarefas específicas. Ele permite que a máquina aprenda a partir de suas experiências, reforçando ou punindo ações específicas de acordo com o resultado. Utiliza técnicas de reforço positivo, em que a máquina é recompensada por boas ações, e reforço negativo, em que é punida por ações ruins.

TÉCNICAS E TAREFAS

CLASSIFICAÇÃO

Trata-se de uma técnica de aprendizado de máquina que permite aos usuários classificar os dados em grupos para facilitar a análise. É usado para prever o comportamento futuro de um dado ou para descobrir padrões em um conjunto de dados.

AGRUPAMENTO

Trata-se de uma técnica de aprendizado de máquina utilizada para identificar grupos em dados. É usado para descobrir padrões e relações entre variáveis. Por exemplo, pode ser usado para agrupar pessoas com base em características como idade, gênero ou localização. O agrupamento pode ser usado para encontrar grupos com características semelhantes, classificar objetos e prever comportamentos futuros.

REGRAS DE ASSOCIAÇÃO

Trata-se de uma técnica de aprendizado de máquina utilizada para descobrir relações entre variáveis em conjunto de dados. Essas regras descrevem padrões que ocorrem com frequência e podem ser usadas para prever comportamentos futuros. Geralmente, elas são usadas para identificar padrões de compra, comportamento de consumidor, padrões de fraudes, etc.

MODELOS LINEARES

Trata-se de um conjunto de técnicas de aprendizado de máquina usado para prever a resposta de uma variável dependente (variável que você está tentando prever) com base em um ou mais variáveis independentes (variáveis que você usa para prever a variável dependente). Estes modelos têm como base a hipótese de que a relação entre as variáveis é linear. Os modelos lineares são largamente utilizados na análise de regressão, pois permitem a previsão de resultados futuros com base em dados passados.

TIPOS DE REGRESSÃO

REGRESSÃO LINEAR

Trata-se da ferramenta estatística que nos ajuda a quantificar a relação entre uma variável específica e um resultado que nos interessa enquanto controlamos outros fatores. Em outras palavras, podemos isolar o efeito de uma variável enquanto mantemos os efeitos das outras variáveis constantes.

REGRESSÃO LOGÍSTICA

Trata-se da ferramenta estatística que tem como objetivo produzir, a partir de um conjunto de observações, um modelo que permita a predição de valores tomados por uma variável categórica, frequentemente binária, a partir de uma série de variáveis explicativas contínuas e/ou binárias.

MATRIZ DE CONFUSÃO

TIPOS DE MEDIÇÃO

MEDIÇÃO	DESCRIÇÃO	FÓRMULA
ACURÁCIA	Trata-se da métrica mais simples que permite mensurar o percentual de acertos, isto é, a quantidade de previsões corretas dentro do total de previsões possíveis. Responde à pergunta: dentre todas as previsões realizadas, quantas o modelo acertou?	$\frac{VP + VN}{VP + FP + VN + FN}$
SENSIBILIDADE	Trata-se da métrica que permite avaliar a capacidade do classificador de detectar com sucesso resultados positivos (também chamado de revocação ou recall). Responde à pergunta: *dentre os valores realmente positivos, quantos o modelo acertou (previu corretamente como positivo)?*	$\frac{VP}{VP + FN}$
ESPECIFICIDADE	Trata-se da métrica que permite avaliar a capacidade do classificador de detectar com sucesso resultados negativos. Responde à pergunta: dentre os valores realmente negativos, quantos o modelo acertou (previu corretamente como negativo)?	$\frac{FP}{FP + VN}$
PRECISÃO	Trata-se da métrica que permite mensurar a proporção de previsões positivas corretas sobre a soma de todos os valores positivos. Responde à pergunta: dentre os valores previstos como positivos, quantos o modelo acertou (previu corretamente como positivo)?	$\frac{VP}{VP + FP}$
F1-SCORE	Trata-se da média harmônica calculada com base na precisão e na sensibilidade, logo é uma medida derivada dessas outras medidas. Essa medida tenta condensar em uma única medida um pouco da precisão e um pouco da sensibilidade.	$2 * \frac{PRECISÃO * RECALL}{PRECISÃO + RECALL}$

PRÉ-PROCESSAMENTO DE DADOS

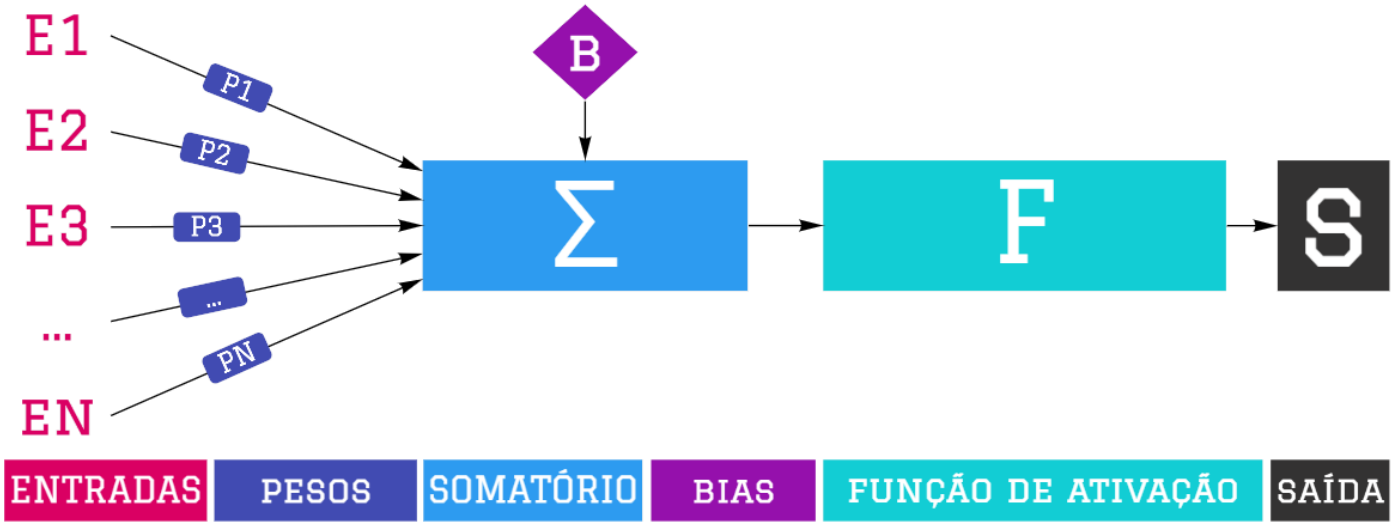
PRÉ-PROCESSAMENTO DE DADOS

Trata-se do processo de preparação de dados para análise e modelagem adicionais. Envolve a transformação de dados brutos em um formato mais adequado para algoritmos de aprendizado de máquina por meio de tarefas como limpeza, normalização e organização dos dados para que possam ser analisados mais facilmente e usados para fazer previsões. É também uma etapa essencial em qualquer projeto de aprendizado de máquina, pois garante que os dados estejam em um formato adequado para o processo de modelagem.

REDES NEURAIS FEED-FORWARD

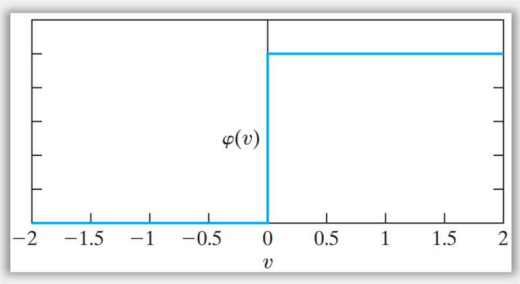
REDES NEURAIS FEED-FORWARD

Trata-se de um tipo de rede neural artificial em que os dados fluem em apenas uma direção, da entrada para a saída. É o tipo mais comum de redes neurais artificiais e é usado para tarefas de aprendizado supervisionado. Em uma rede feed-forward, os neurônios são organizados em camadas e o sinal se propaga de uma camada para outras. Cada neurônio recebe entradas dos neurônios da camada anterior, realiza uma soma ponderada das entradas e passa o resultado para a próxima camada.

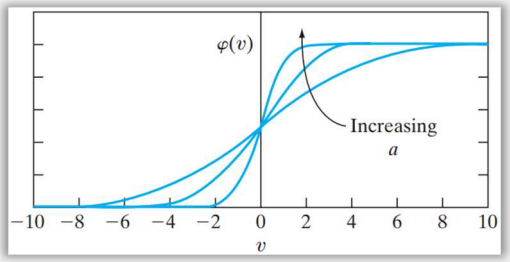


TIPOS DE FUNÇÕES DE ATIVAÇÃO

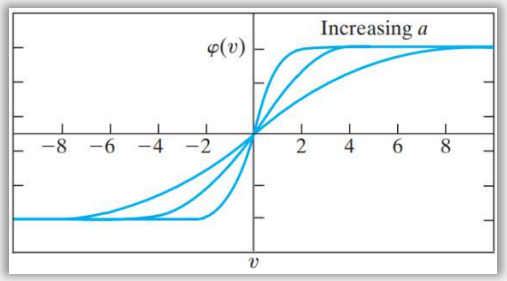
FUNÇÃO DE ATIVAÇÃO	FUNÇÕES DE LIMITE
DESCRIÇÃO DA FUNÇÃO	Essa função compara um valor com um determinado limite (nesse caso, o limite é 0) e decide se um neurônio será ativado (1) ou não será ativado (0).
REPRESENTAÇÃO DA FÓRMULA	$f(x) = \begin{cases} 0 & for\ x < 0 \\ 1 & for\ x \geqslant 0 \end{cases}$



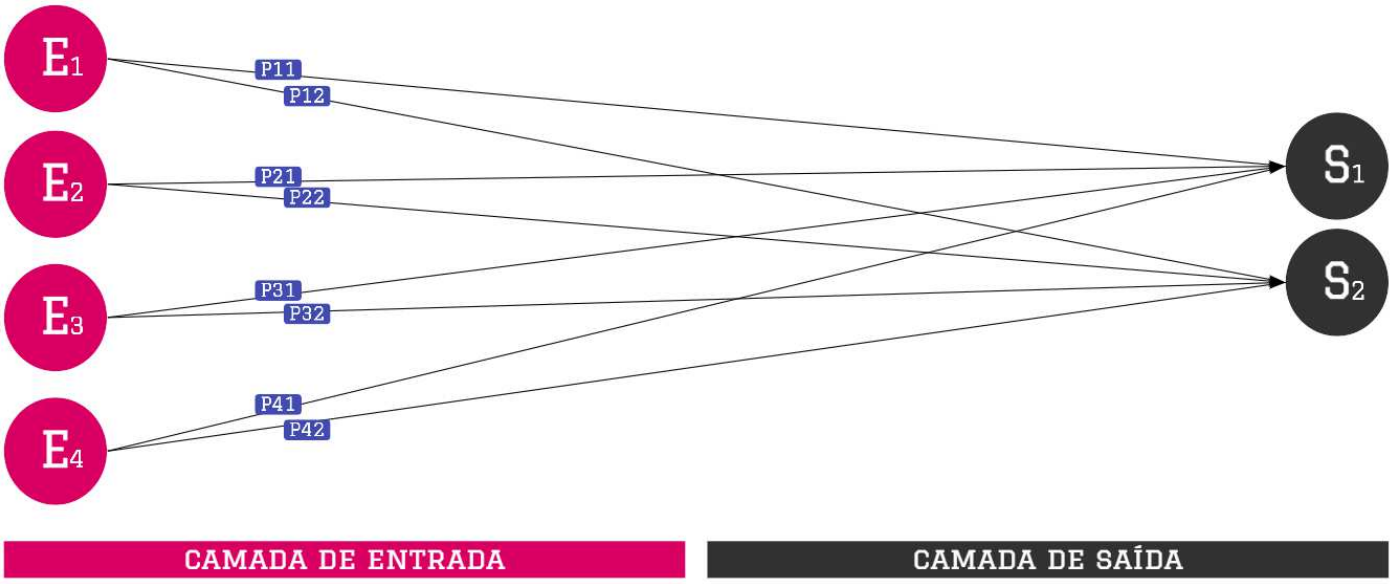
FUNÇÃO DE ATIVAÇÃO	FUNÇÕES LOGÍSTICAS
DESCRIÇÃO DA FUNÇÃO	Essa função recebe um valor real qualquer como entrada [-∞,+∞] e retorna um valor de saída entre 0 e 1.
REPRESENTAÇÃO DA FÓRMULA	$f(x) = \frac{1}{1+e^{-x}}$



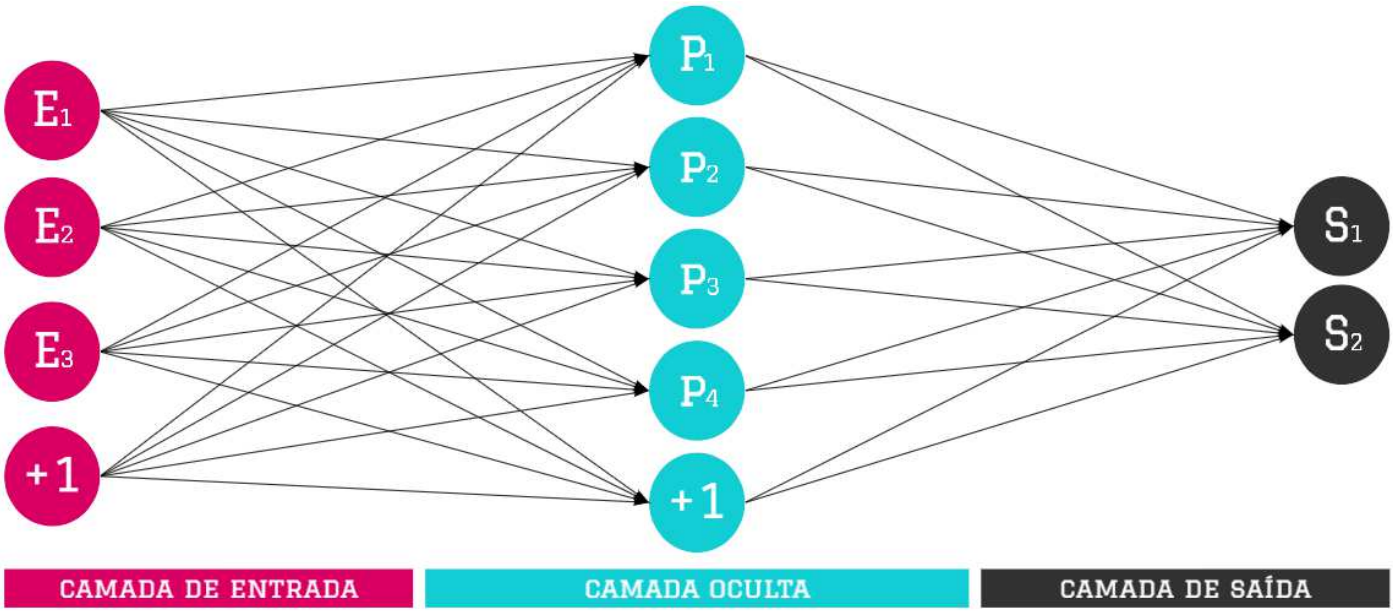
FUNÇÃO DE ATIVAÇÃO	FUNÇÃO TANGENTE HIPERBÓLICA
DESCRIÇÃO DA FUNÇÃO	Essa função recebe um valor real qualquer como entrada [-∞,+∞] e retorna um valor de saída entre -1 e 1.
REPRESENTAÇÃO DA FUNÇÃO	$f(x) = \frac{(e^{x} - e^{-x})}{(e^{x} + e^{-x})}$



SINGLE LAYER PERCEPTRON (SLP)

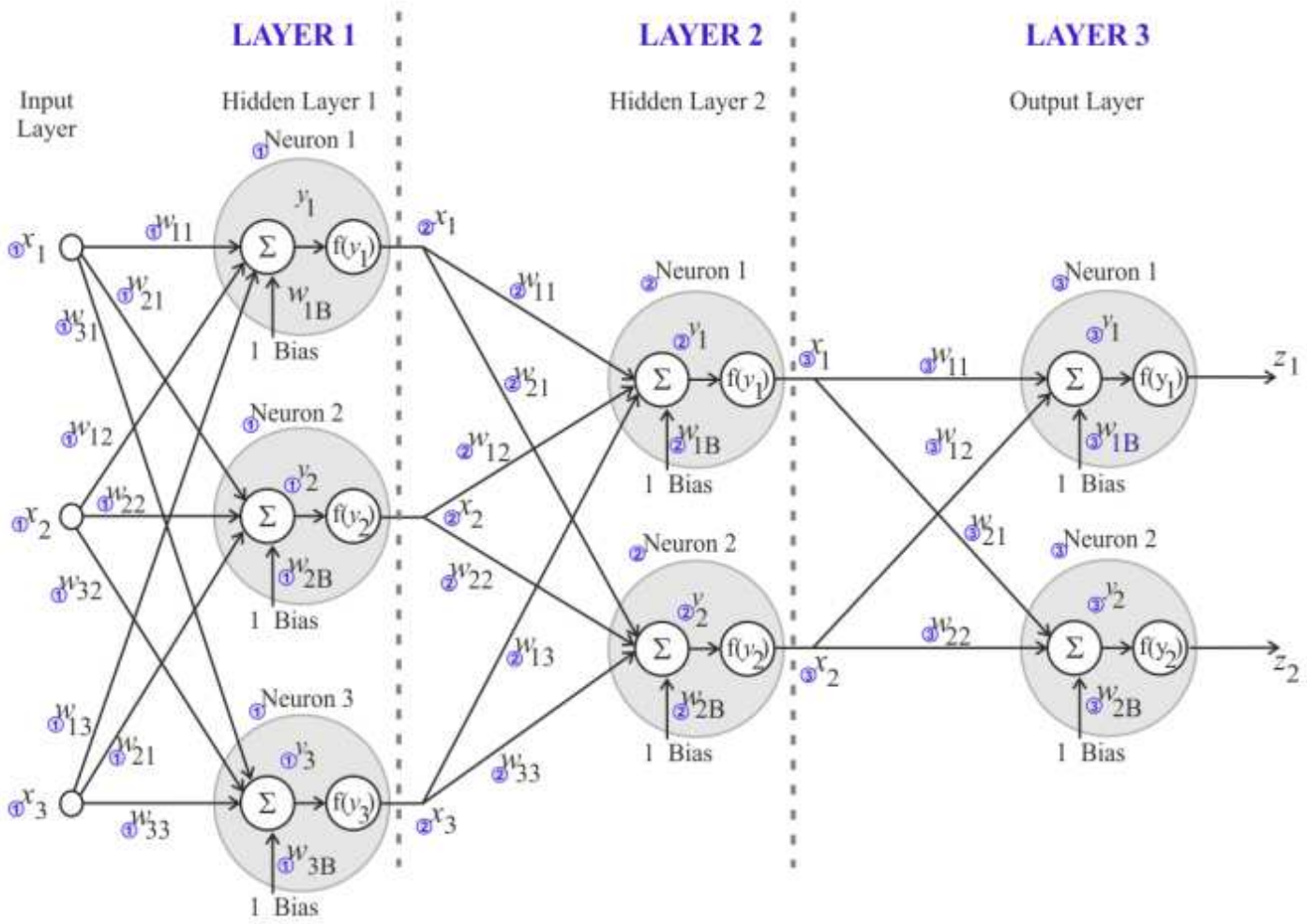


MULTIPLE LAYER PERCEPTRON (MLP)



BACKPROPAGATION

Trata-se de um algoritmo para treinamento de redes neurais artificiais que utiliza gradiente descendente para ajustar pesos na rede a fim de reduzir o custo total e minimizar os erros. O custo é calculado comparando a saída da rede com a saída desejada e calculando a diferença entre elas. A retropropagação usa o erro para calcular o gradiente da função de custo em relação aos pesos e desvios na rede e ajusta os pesos de acordo. Este processo é repetido até que o custo atinja um nível aceitável.

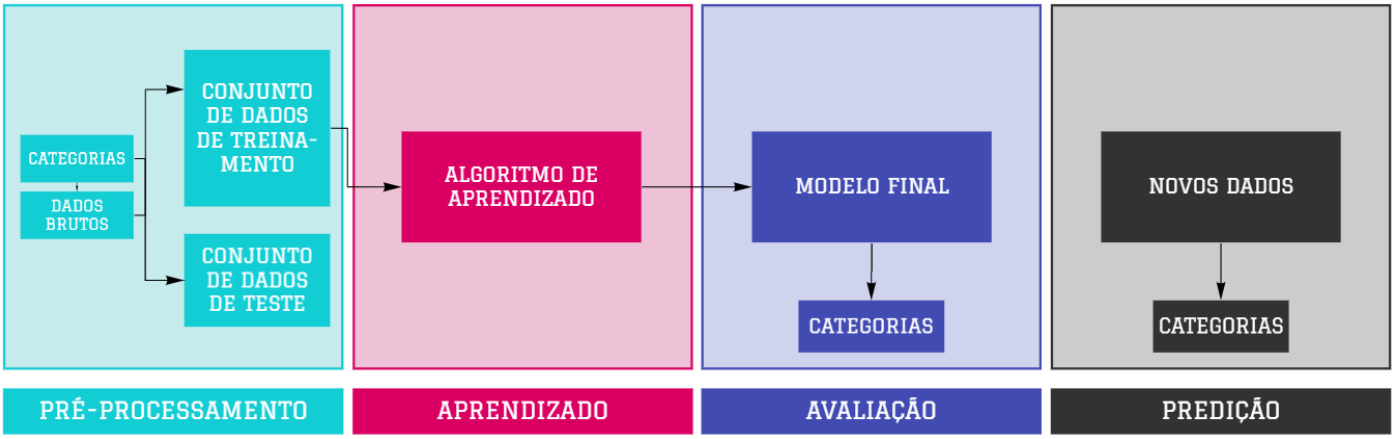


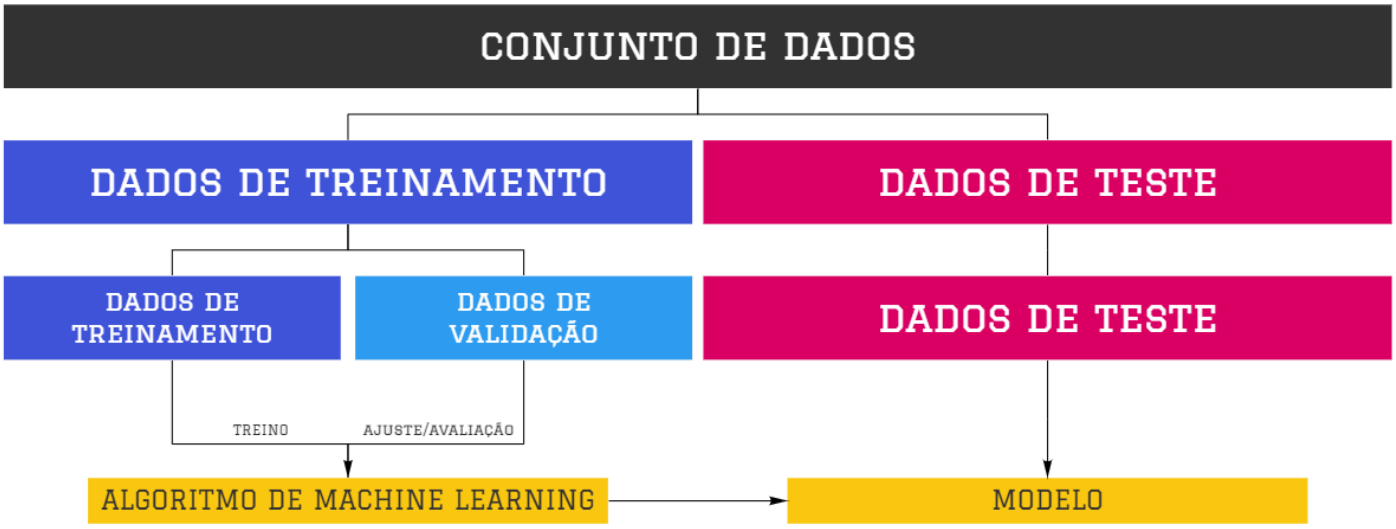
FONTES DE ERROS EM MODELOS PREDITIVOS

VALIDAÇÃO/AVALIAÇÃO DE MODELOS PREDITIVOS

VALIDAÇÃO/AVALIAÇÃO DE MODELOS PREDITIVOS

Trata-se do processo de avaliar o desempenho de um modelo preditivo em um conjunto de dados separado que não foi usado para treinamento, a fim de estimar o desempenho do modelo em dados não vistos. É uma etapa essencial no desenvolvimento de um modelo preditivo e é usado para avaliar a precisão e a confiabilidade do modelo.





VALIDAÇÃO CRUZADA

VALIDAÇÃO CRUZADA

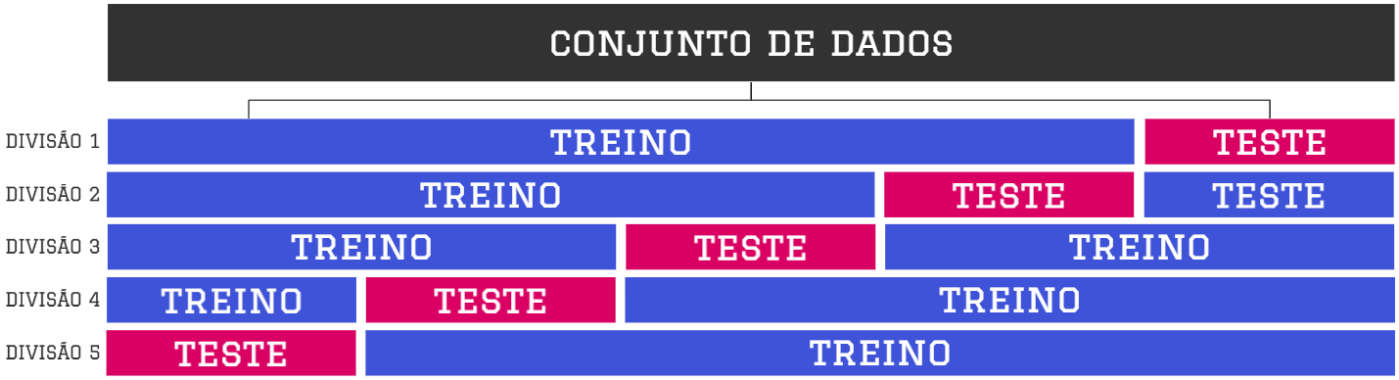
Trata-se de uma técnica usada para avaliar modelos de aprendizado de máquina. É um processo de dividir um conjunto de dados em várias partes, treinando o modelo em uma parte e testando-o em outra parte. Esse processo é repetido várias vezes, com cada parte usada como um conjunto de teste uma vez. Isso permite que o modelo seja testado em vários conjuntos de dados e fornece uma medida mais precisa de seu desempenho.

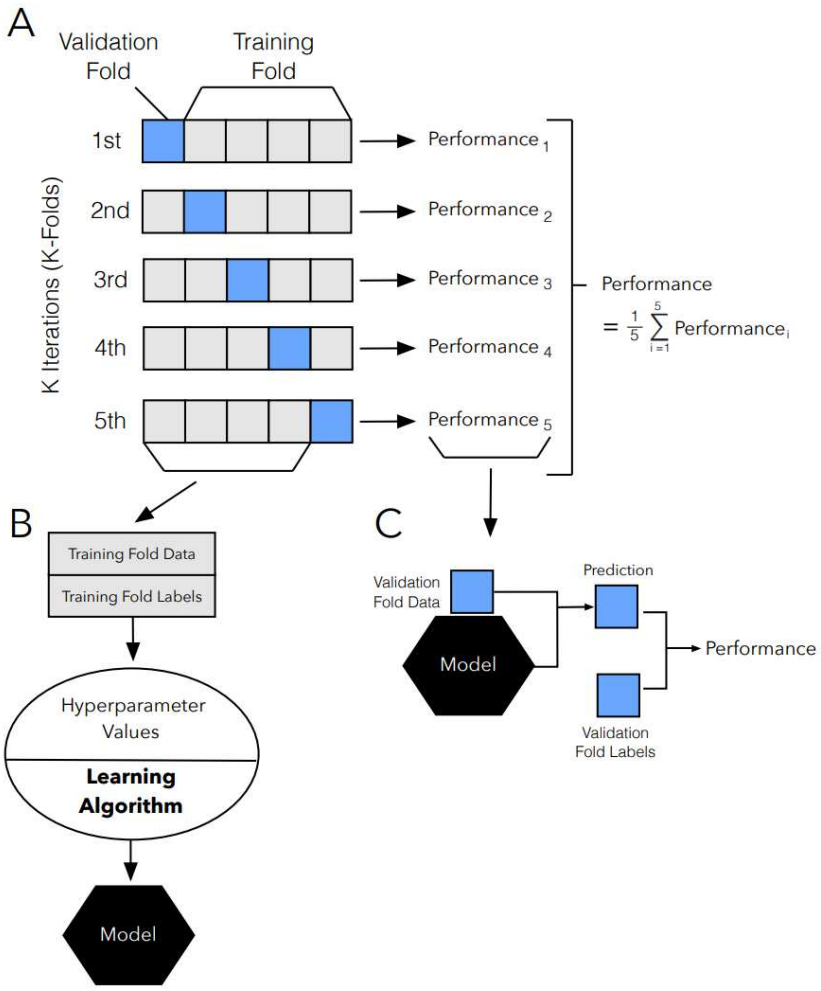
MÉTODO HOLDOUT

Trata-se de um método de validação de modelos preditivos usada para avaliar o desempenho de um modelo de aprendizado de máquina. Isso é feito dividindo os dados em um conjunto de treinamento, que é usado para treinar o modelo, e um conjunto de teste, que é usado para avaliar o modelo (o subconjunto de treinamento pode ser subdividido em um conjunto para validação, isto é, otimização/ajuste de parâmetros). O método busca fazer uma estimativa imparcial do desempenho do modelo.

MÉTODO K-FOLD

Trata-se de um método de validação de modelos preditivos usado para avaliar o desempenho de um modelo de aprendizado de máquina - além de avaliar a sua robustez. Ele envolve a divisão de um conjunto de dados em k subconjuntos de tamanhos iguais. Um subconjunto é usado para testar o modelo, enquanto os outros subconjuntos k-1 são usados para treinar o modelo. O modelo é então testado k vezes, cada vez usando um subconjunto de teste diferente. A média dos resultados é usada para medir a precisão/desempenho do modelo.





CURVA ROC

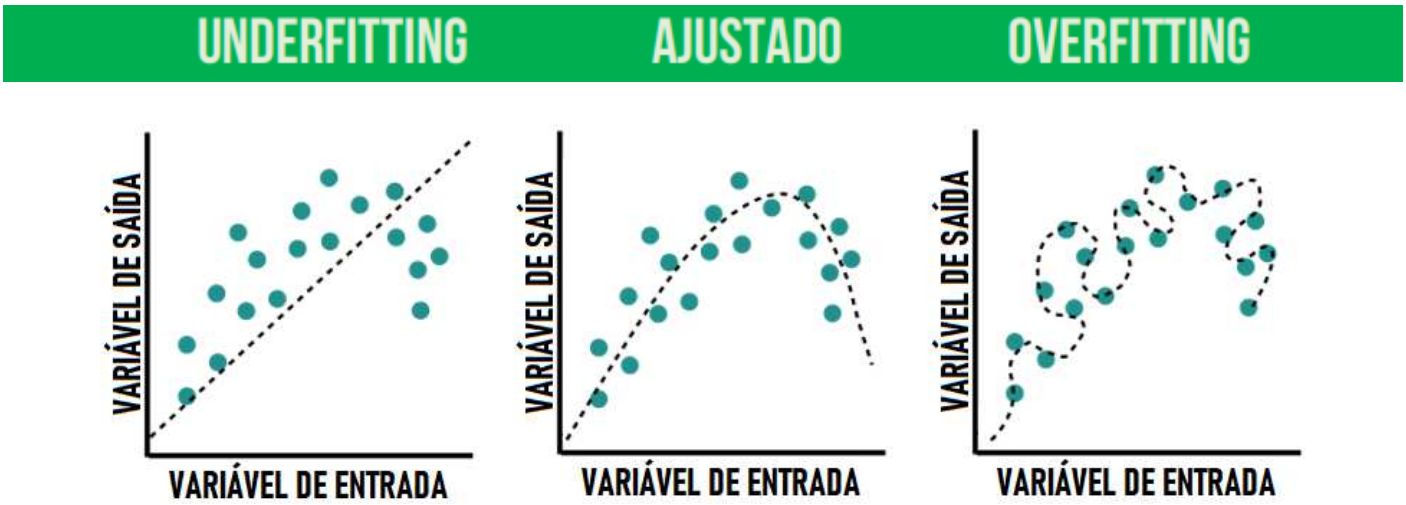
CURVA ROC (RECEIVER OPERATING CHARACTERISTIC)

Trata-se de uma representação gráfica do desempenho de um sistema de classificação binária à medida que seu limite de discriminação é variado. É um gráfico da taxa de verdadeiros-positivos em relação à taxa de falsos-positivos para um determinado limite de probabilidade. Ele exibe a capacidade de um modelo de distinguir entre uma classe positiva e uma classe negativa.

UNDERFITTING E OVERFITTING

UNDERFITTING E OVERFITTING

Underfitting ocorre quando um modelo de aprendizado de máquina não captura adequadamente o padrão subjacente dos dados. Trata-se de uma falha na captura da variação nos dados, resultando em uma representação imprecisa desses. Já o Overfitting ocorre quando um modelo de aprendizado de máquina captura muita variação nos dados, resultando em um modelo que não generaliza bem e é excessivamente sensível a pequenas alterações nos dados.



VIÉS E VARIÂNCIA

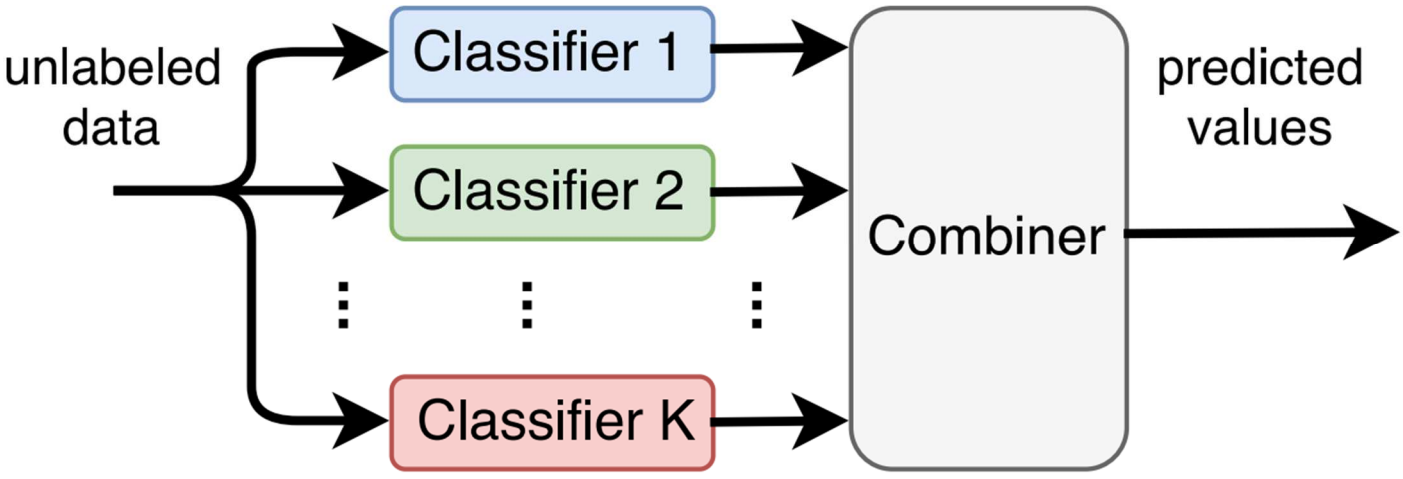
BIAS / VIÉS — TRATA-SE DA DIFERENÇA ENTRE A PREDIÇÃO (MÉDIA) DE VALOR DE UMA VARIÁVEL E O VALOR CORRETO QUE O MODELO DEVERIA PREVER. EM OUTRAS PALAVRAS, É O ERRO QUE RESULTA DE SUPOSIÇÕES IMPRECISAS DE UM MODELO.

VARIANCE / VARIÂNCIA — TRATA-SE DA SENSIBILIDADE DE UM MODELO AO SER UTILIZADO COM NOVOS CONJUNTOS DE DADOS DIFERENTES - QUÃO CONSISTENTE/VARIÁVEL É O MODELO AO SER EXECUTADO SOBRE NOVOS CONJUNTOS DE DADOS.

COMBINAÇÕES	DESCRIÇÃO	REPRESENTAÇÃO
BAIXO VIÉS E BAIXA VARIÂNCIA	Trata-se de um modelo que possui ótima precisão em suas previsões com dados de treino e que varia muito pouco quando aplicado a novos dados. Note que os pontos estão no centro do alvo e não estão espalhados.	
BAIXO VIÉS E ALTA VARIÂNCIA	Trata-se de um modelo que possui boa precisão em suas previsões com dados de treino (*overfitting*), mas que varia bastante quando aplicado a novos dados. Note que os pontos estão próximos ao centro do alvo, porém estão um pouco espalhados.	
ALTO VIÉS E BAIXA VARIÂNCIA	Trata-se de um modelo que possui péssima precisão em suas previsões com dados de treino (*underfitting*), mas que varia pouco quando aplicado a novos dados. Note que os dados estão longe do centro do alvo, porém não estão espalhados.	
ALTO VIÉS E ALTA VARIÂNCIA	Trata-se de um modelo que possui péssima precisão em suas previsões com dados de treino e que varia bastante quando aplicado a novos dados. Note que os pontos estão longe do centro do alvo e também estão bastante espalhados.	

ENSEMBLE

Trata-se de um tipo de técnica de aprendizado de máquina em que vários modelos são usados juntos para fazer previsões mais precisas do que qualquer modelo individual. Os modelos de Ensemble combinam as previsões de vários modelos para produzir melhor desempenho preditivo do que poderia ser obtido de qualquer um dos modelos individuais sozinhos. Essa técnica pode ser usada para problemas de regressão e classificação.



ENSEMBLE: BAGGING

Trata-se de uma técnica de aprendizado de máquina de ensemble utilizado para minimizar a variância de um modelo. Para tal, ele cria vários modelos que são treinados em diferentes subconjuntos dos mesmos dados e, em seguida, combina suas previsões. A ideia é que cada modelo individual tenha pontos fortes diferentes e, ao combiná-los, a precisão geral do modelo será aprimorada.

ENSEMBLE: BOOSTING

Trata-se de uma técnica de aprendizado de máquina de ensemble que combina vários weak learners para criar um strong learner. Funciona treinando cada weak learner em uma sequência, com cada aluno subsequente focando nos erros cometidos pelo aluno anterior. A saída dos weak learners é combinada para produzir um único strong learner. Boosting é freqüentemente usado para melhorar a precisão de modelos que são propensos a underfitting.

ENSEMBLE: STACKING

Trata-se de uma técnica de aprendizado de máquina de ensemble que difere do bagging e boosting por utilizar um conjunto heterogêneo de weak learners e também por utilizar a saída dos weak learners como entrada em um meta-modelo com o objetivo de aprender o mapeamento entre as saídas e as classes corretas.

TÉCNICAS DE REGULARIZAÇÃO

REGULARIZAÇÃO

Trata-se do controle fino do nível de complexidade de um dado modelo, limitando seu grau de liberdade (flexibilidade) para se ajustar aos dados de treinamento, visando evitar sobreajustamento (overfitting). Em outras palavras, essa técnica reduz a variância de um modelo, tornando-o mais generalizável.

REGRESSÃO LASSO (L1)

Trata-se de uma técnica de regularização chamada Least Absolute Shrinkage and Selection Operator (LASSO) para reduzir a complexidade de um modelo preditivo. Essa técnica reduz certos coeficientes a zero, eliminando assim as variáveis menos importantes do modelo e ajudando a reduzir o overfitting. O resultado é um modelo esparso que é mais fácil de interpretar e melhor na generalização para novos dados.

REGRESSÃO RIDGE (L2)

Trata-se de uma técnica de regularização que adiciona um termo de penalidade à função de custo para reduzir a complexidade do modelo. O termo de penalidade é um parâmetro de regularização que reduz as estimativas do coeficiente para zero. Essa técnica é útil para prevenir o overfitting e melhorar a precisão do modelo. Ela também pode ser usada para identificar variáveis importantes em um conjunto de dados.

PRUNING

Trata-se de uma técnica de reegularização utilizada para reduzir a complexidade de um modelo removendo parâmetros ou conexões desnecessárias em uma rede. Isso ajuda a melhorar a precisão do modelo preditivo, reduzindo o overfitting, melhorando a velocidade de treinamento e reduzindo os requisitos de memória.

DROPOUT

Trata-se de uma técnica de regularização usada em redes neurais para evitar o overfitting. Ele funciona desconectando aleatoriamente uma fração das unidades de entrada durante o treinamento, o que força o modelo a aprender com menos parâmetros e reduz a sua complexidade. A fração de unidades de entrada a serem descartadas é normalmente definida como uma porcentagem entre 20% e 50%.

EARLY STOPPING

Trata-se de uma técnica regularização usada em redes neurais para evitar o overfitting dos dados de treinamento. Ele funciona monitorando o desempenho do modelo em um conjunto de dados de validação durante o treinamento e interrompendo o processo de treinamento quando o desempenho no conjunto de dados de validação não melhora por um determinado número de épocas de treinamento.

DATA AUGMENTATION

Trata-se de uma técnica de regularização usada para aumentar artificialmente o tamanho de um conjunto de dados de treinamento, manipulando e adicionando transformações aleatórias aos dados existentes. Isso ajuda a melhorar a precisão e a robustez dos modelos de aprendizado de máquina, introduzindo novas variações nos dados de treinamento. As técnicas de aumento de dados incluem corte, inversão, rotação, adição de ruído, deslocamento, etc.

OTIMIZAÇÃO DE HIPERPARÂMETROS

OTIMIZAÇÃO DE HIPERPARÂMETROS

Trata-se do processo de selecionar o melhor conjunto de hiperparâmetros para um determinado algoritmo de aprendizado de máquina, a fim de maximizar seu desempenho em um determinado conjunto de dados. Isso geralmente é feito por meio de um processo de tentativa e erro, usando métodos como pesquisa em grade, pesquisa aleatória ou otimização bayesiana.

PARÂMETROS	TRATA-SE DE REPRESENTAÇÕES INTERNAS DO MODELO AJUSTADAS AUTOMATICAMENTE PELO PROCESSO DE APRENDIZAGEM OU TREINAMENTO, SINTETIZADOS A PARTIR DE PADRÕES ESTATÍSTICOS DOS DADOS, TAIS COMO OS PESOS DE UMA REGRESSÃO OU DE UMA REDE NEURAL. EM OUTRAS PALAVRAS, TRATA-SE DE UMA VARIÁVEL DE CONFIGURAÇÃO INTERNA A UM MODELO E CUJOS VALORES PODEM SER ESTIMADOS A PARTIR DOS DADOS.
HIPER PARÂMETROS	TRATA-SE DAS CARACTERÍSTICAS DO MODELO OU DO SEU PROCESSO DE TREINAMENTO QUE REFLETEM UMA OPÇÃO DO CIENTISTA, QUE NÃO SÃO EXTRAÍDOS AUTOMATICAMENTE DOS DADOS, TAIS COMO A INTENSIDADE DE REGULARIZAÇÃO DE UM MODELO, A PROFUNDIDADE MÁXIMA DE ÁRVORES DE DECISÃO, ETC. EM OUTRAS PALAVRAS, TRATA-SE DE UMA VARIÁVEL DE CONFIGURAÇÃO EXTERNA A UM MODELO E CUJOS VALORES NÃO PODEM SER ESTIMADOS A PARTIR DOS DADOS.

GRID SEARCH

Trata-se de uma técnica usada para otimizar hiperparâmetros de um determinado modelo, a fim de obter o melhor desempenho possível. É uma pesquisa exaustiva sobre um conjunto de hiperparâmetros especificados manualmente pelo usuário. Funciona explorando sistematicamente cada combinação dos parâmetros de uma grade especificada e avaliando o desempenho do modelo para cada combinação. A combinação com a pontuação mais alta é então selecionada como o modelo de melhor desempenho.

RANDOM SEARCH

Trata-se de uma técnica usada para otimizar hiperparâmetros que envolve a amostragem aleatória de uma gama de valores de hiperparâmetros possíveis e, em seguida, a seleção do melhor conjunto com base nos resultados. É um dos métodos mais simples e eficientes de ajustar um modelo e é frequentemente usado como linha de base para comparação com técnicas de otimização mais avançadas.

BAYESIAN SEARCH

Trata-se de uma técnica usada para otimizar hiperparâmetros que usa a inferência bayesiana para construir um modelo probabilístico de um espaço de pesquisa. Ele usa esse modelo para otimizar o processo de busca, levando em consideração a incerteza do problema e orientando a busca para melhores soluções. É particularmente útil para problemas onde o espaço de busca é grande e o custo de avaliação de uma única solução é alto.

SEPARABILIDADE DE DADOS

SEPARABILIDADE DE DADOS

Trata-se da capacidade de separar dados em grupos distintos e separados. É uma medida de quão bem os dados podem ser divididos em clusters, classes ou categorias distintas e uma ferramenta útil para agrupamento e algoritmos de classificação, pois a separabilidade de dados pode ajudar a identificar e classificar pontos de dados em grupos específicos.

REDUÇÃO DE DIMENSIONALIDADE

Trata-se do processo de redução do número de features em um conjunto de dados, mantendo as informações mais importantes. É usado para reduzir a complexidade de um conjunto de dados enquanto ainda preserva as características essenciais dos dados; além de ser usado no aprendizado de máquina para reduzir o overfitting, reduzir o tempo de computação e melhorar a precisão dos modelos.

VANTAGENS DA REDUÇÃO DE DIMENSIONALIDADE

Simplificação dos modelos de aprendizado de máquina: é mais fácil ajustar um modelo que tenha duas variáveis de entrada do que um modelo que tenha 80 variáveis de entrada.
Redução do *overfitting*: é muito mais difícil ocorrer sobreajuste em um modelo de aprendizado de máquina com menos variáveis do que com muitas variáveis.
Simplificação da representação gráfica: visualizar dados representados em mais de três dimensões é inviável para seres humanos.
Redução do custo computacional: com menos variáveis, é necessário utilizar menos recursos computacionais para realizar o treinamento de um modelo de aprendizado de máquina.
Redução do tempo de treinamento: como há menos variáveis para ajustar, leva menos tempo para treinar o modelo de aprendizado de máquina.
Aumentar a performance: como há variáveis com nenhuma correlação, sua eliminação ajuda a melhorar o desempenho do modelo de aprendizado de máquina.

SELEÇÃO DE VARIÁVEIS

Trata-se do processo de selecionar um subconjunto de variáveis relevantes para uso na construção do modelo, isto é, aquelas que têm o relacionamento mais forte com a variável dependente e também que são mais úteis para a previsão. Busca-se selecionar aquelas que possuem maior poder preditivo, descartando as demais. Essa técnica é importante porque pode ajudar a reduzir a complexidade dos modelos e melhorar a precisão das previsões.

MÉTODO FILTER

Trata-se de uma técnica de seleção de variáveis que usa várias métricas estatísticas para avaliar a importância dos recursos e selecionar os recursos mais relevantes para a construção de um modelo preditivo. Ela ajuda a reduzir a dimensionalidade dos dados para melhorar a precisão do modelo. Em geral, é usado em combinação com outras técnicas.

MÉTODO WRAPPER

Trata-se de uma técnica de seleção de variáveis que avalia o conjunto de features escolhido otimizando uma métrica de desempenho. Ele funciona selecionando subconjuntos de features e, em seguida, avaliando o desempenho de um modelo treinado nesses recursos. O objetivo é encontrar o conjunto de features que produz o modelo de melhor desempenho.

MÉTODO EMBEDDED

Trata-se de uma técnica de seleção de variáveis que combina as qualidades do método filter e do método wrapper para treinar um classificador em um subconjunto de features e, em seguida, adicionar recursos adicionais ao modelo conforme necessário. O algoritmo usará a melhor combinação de features para gerar as previsões mais precisas.

FATORAÇÃO DE MATRIZES

Trata-se de uma técnica utilizada para fatorar a matriz original em outras matrizes menores a fim de encontrar o melhor subconjunto de dados com menor dimensionalidade que seja capaz de representar a matriz original. Ela pode ser usada para recomendar itens aos usuários, agrupar itens semelhantes e reduzir a dimensionalidade de um conjunto de dados.

SISTEMAS DE RECOMENDAÇÃO

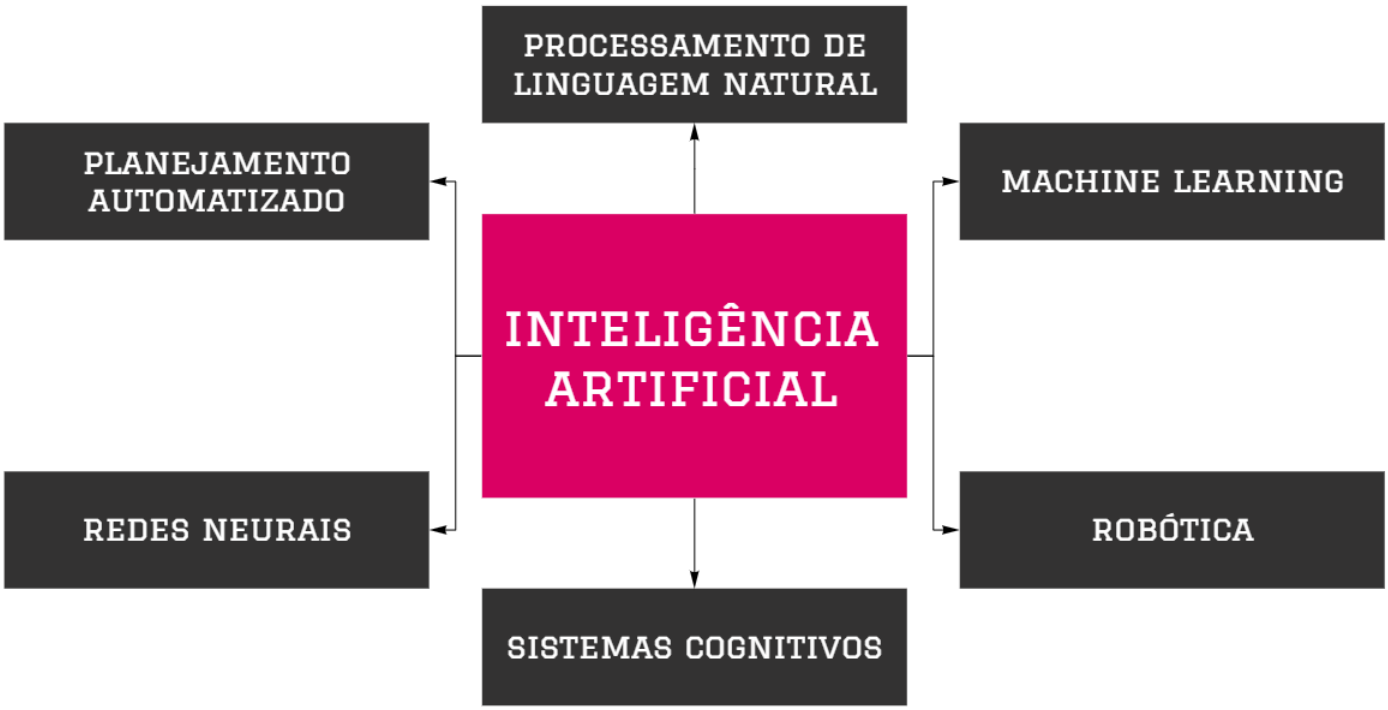
Trata-se de um algoritmo que faz previsões sobre os interesses de um usuário. Esses sistemas usam dados históricos do usuário e dados de itens específicos para fazer recomendações aos usuários sobre itens nos quais eles podem estar interessados. O objetivo de um sistema de recomendação é fornecer recomendações personalizadas e relevantes aos usuários.

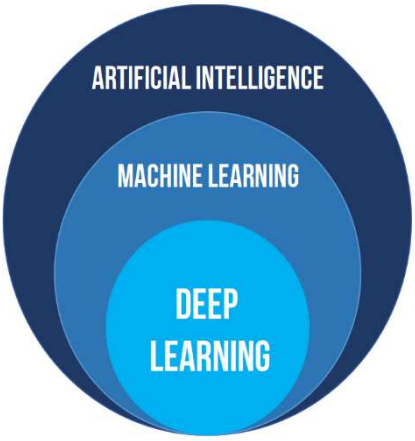
PCA (PRINCIPAL COMPONENT ANALYSIS)

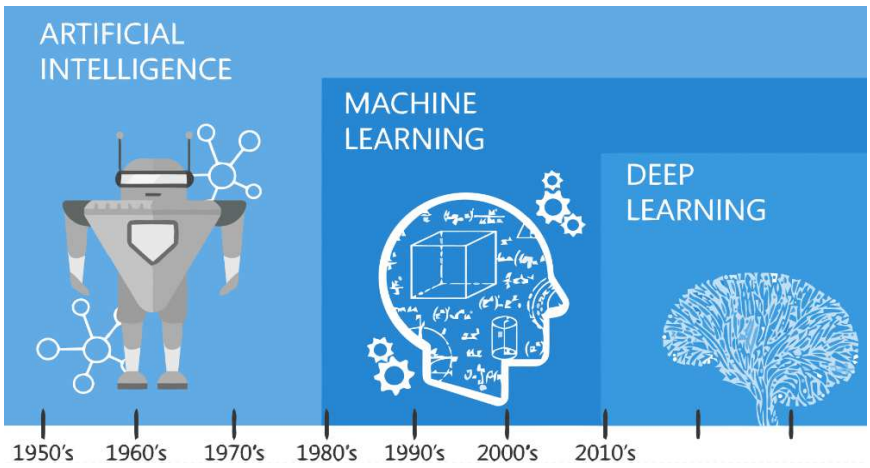
Trata-se de um método estatístico usado para reduzir a dimensionalidade dos dados por meio da transformação de um grande conjunto de variáveis em um conjunto menor de variáveis, chamado de Componentes Principais, enquanto retém o máximo possível da variação dos dados. O PCA é amplamente para interpretar as características mais importantes de um conjunto de dados.

RESUMO

INTELIGENCIA ARTIFICIAL







MACHINE LEARNING

MACHINE LEARNING

SUBCAMPO DA INTELIGÊNCIA ARTIFICIAL QUE FORNECE AOS COMPUTADORES A HABILIDADE DE APRENDER SEM SEREM EXPLICITAMENTE PROGRAMADOS

- PROCESSO EM QUE O DESEMPENHO DE UMA TAREFA AUMENTE COM A EXPERIÊNCIA EXTRAÍDA DE NOVOS DADOS
- A MÁQUINA PODE APRENDER A PARTIR DE SEUS ERROS E FAZER INFERÊNCIAS SOBRE DADOS
- DIFERE-SE DA PROGRAMAÇÃO TRADICIONAL POR GERAR REGRAS A PARTIR DE RESULTADOS E, NÃO, O CONTRÁRIO
- MODELOS SÃO REGRAS ESTATÍSTICAS NA FORMA DE MODELOS MATEMÁTICOS E PARÂMETROS
- A ETAPA DE TREINAMENTO É MAIS CUSTOSA E A ETAPA DE INFERÊNCIA É MENOS CUSTOSA

TERMO	DESCRIÇÃO
TAREFA	Definição genérica daquilo que se deseja produzir como resultado do modelo preditivo. Ex: classificar um documento em três possíveis categorias ou prever o valor de determinada medida.
TÉCNICA	Conjunto de procedimentos que permite melhorar resultados preditivos. Ex: regularização é uma técnica para prevenir o *overfitting*; *hold-out* é uma técnica de separação de dados para medir o desempenho em generalização de um modelo.
ALGORITMO	Fórmula no sentido lato, que permite relacionar as variáveis independentes para prever a variável dependente. Quando aplicamos (ou treinamos) um algoritmo, temos um modelo treinado. Exemplo: Regressão Linear ou Árvores de Decisão.
MODELO (TREINADO)	Objeto computacional que efetivamente transforma uma observação (variáveis independentes) em uma previsão utilizando um algoritmo específico, instanciado e treinado.

TIPOS DE APRENDIZADO

APRENDIZADO SUPERVISIONADO

Trata-se da abordagem que busca encontrar um conjunto de regras ou funções (também chamadas de modelo) a partir dos dados de treinamento que possam ser utilizadas para prever um rótulo ou valor que caracterize um novo exemplo, com base nos valores de seus atributos de entrada. O termo supervisionado vem da presença de um supervisor externo, que conhece a saída (rótulo) desejada para cada exemplo. Com isso, ele pode avaliar a capacidade do algoritmo de predizer o valor de saída para novos exemplos.

APRENDIZADO NÃO SUPERVISIONADO

Trata-se da abordagem em que o algoritmo busca encontrar um padrão subjacente nos dados sem a utilização de um supervisor externo para atribuir rótulos ou categorias pré-definidas para as amostras de treinamento. Os algoritmos são formulados de forma que possam encontrar padrões autonomamente com o intuito de explorar dados desconhecidos e encontrar estruturas interessantes ou ocultas nos dados que não eram visíveis anteriormente para os cientistas de dados.

APRENDIZADO SEMISUPERVISIONADO

Trata-se da abordagem que abrange técnicas de aprendizado de máquina que usam um conjunto de dados parcialmente rotulado. Este conjunto de dados inclui alguns dados com rótulos e alguns dados sem rótulos. As técnicas de aprendizado semi-supervisionado são usadas para combinar dados rotulados e não rotulados para melhorar a precisão do modelo.

APRENDIZADO POR REFORÇO

Trata-se da abordagem que se baseia na recompensa e na punição para ensinar a máquina a realizar tarefas específicas. Ele permite que a máquina aprenda a partir de suas experiências, reforçando ou punindo ações específicas de acordo com o resultado. Utiliza técnicas de reforço positivo, em que a máquina é recompensada por boas ações, e reforço negativo, em que é punida por ações ruins.

TÉCNICAS E TAREFAS

CLASSIFICAÇÃO

Trata-se de uma técnica de aprendizado de máquina que permite aos usuários classificar os dados em grupos para facilitar a análise. É usado para prever o comportamento futuro de um dado ou para descobrir padrões em um conjunto de dados.

AGRUPAMENTO

Trata-se de uma técnica de aprendizado de máquina utilizada para identificar grupos em dados. É usado para descobrir padrões e relações entre variáveis. Por exemplo, pode ser usado para agrupar pessoas com base em características como idade, gênero ou localização. O agrupamento pode ser usado para encontrar grupos com características semelhantes, classificar objetos e prever comportamentos futuros.

REGRAS DE ASSOCIAÇÃO

Trata-se de uma técnica de aprendizado de máquina utilizada para descobrir relações entre variáveis em conjunto de dados. Essas regras descrevem padrões que ocorrem com frequência e podem ser usadas para prever comportamentos futuros. Geralmente, elas são usadas para identificar padrões de compra, comportamento de consumidor, padrões de fraudes, etc.

MODELOS LINEARES

Trata-se de um conjunto de técnicas de aprendizado de máquina usado para prever a resposta de uma variável dependente (variável que você está tentando prever) com base em um ou mais variáveis independentes (variáveis que você usa para prever a variável dependente). Estes modelos têm como base a hipótese de que a relação entre as variáveis é linear. Os modelos lineares são largamente utilizados na análise de regressão, pois permitem a previsão de resultados futuros com base em dados passados.

TIPOS DE REGRESSÃO

REGRESSÃO LINEAR

Trata-se da ferramenta estatística que nos ajuda a quantificar a relação entre uma variável específica e um resultado que nos interessa enquanto controlamos outros fatores. Em outras palavras, podemos isolar o efeito de uma variável enquanto mantemos os efeitos das outras variáveis constantes.

REGRESSÃO LOGÍSTICA

Trata-se da ferramenta estatística que tem como objetivo produzir, a partir de um conjunto de observações, um modelo que permita a predição de valores tomados por uma variável categórica, frequentemente binária, a partir de uma série de variáveis explicativas contínuas e/ou binárias.

MATRIZ DE CONFUSÃO

TIPOS DE MEDIÇÃO

MEDIÇÃO	DESCRIÇÃO	FÓRMULA
ACURÁCIA	Trata-se da métrica mais simples que permite mensurar o percentual de acertos, isto é, a quantidade de previsões corretas dentro do total de previsões possíveis. Responde à pergunta: dentre todas as previsões realizadas, quantas o modelo acertou?	$\frac{VP + VN}{VP + FP + VN + FN}$
SENSIBILIDADE	Trata-se da métrica que permite avaliar a capacidade do classificador de detectar com sucesso resultados positivos (também chamado de revocação ou recall). Responde à pergunta: *dentre os valores realmente positivos, quantos o modelo acertou (previu corretamente como positivo)?*	$\frac{VP}{VP + FN}$
ESPECIFICIDADE	Trata-se da métrica que permite avaliar a capacidade do classificador de detectar com sucesso resultados negativos. Responde à pergunta: dentre os valores realmente negativos, quantos o modelo acertou (previu corretamente como negativo)?	$\frac{FP}{FP + VN}$
PRECISÃO	Trata-se da métrica que permite mensurar a proporção de previsões positivas corretas sobre a soma de todos os valores positivos. Responde à pergunta: dentre os valores previstos como positivos, quantos o modelo acertou (previu corretamente como positivo)?	$\frac{VP}{VP + FP}$
F1-SCORE	Trata-se da média harmônica calculada com base na precisão e na sensibilidade, logo é uma medida derivada dessas outras medidas. Essa medida tenta condensar em uma única medida um pouco da precisão e um pouco da sensibilidade.	$2 * \frac{PRECISÃO * RECALL}{PRECISÃO + RECALL}$

PRÉ-PROCESSAMENTO DE DADOS

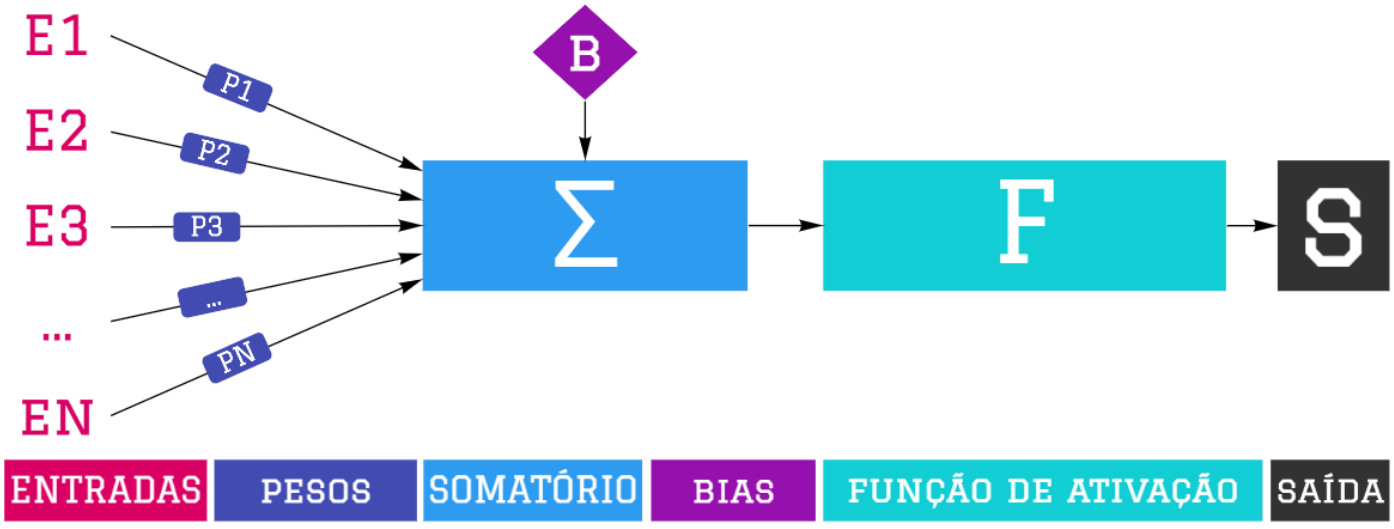
PRÉ-PROCESSAMENTO DE DADOS

Trata-se do processo de preparação de dados para análise e modelagem adicionais. Envolve a transformação de dados brutos em um formato mais adequado para algoritmos de aprendizado de máquina por meio de tarefas como limpeza, normalização e organização dos dados para que possam ser analisados mais facilmente e usados para fazer previsões. É também uma etapa essencial em qualquer projeto de aprendizado de máquina, pois garante que os dados estejam em um formato adequado para o processo de modelagem.

REDES NEURAIS FEED-FORWARD

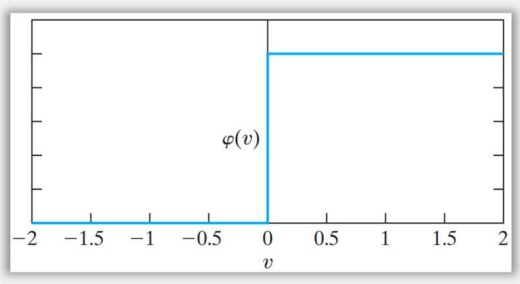
REDES NEURAIS FEED-FORWARD

Trata-se de um tipo de rede neural artificial em que os dados fluem em apenas uma direção, da entrada para a saída. É o tipo mais comum de redes neurais artificiais e é usado para tarefas de aprendizado supervisionado. Em uma rede feed-forward, os neurônios são organizados em camadas e o sinal se propaga de uma camada para outras. Cada neurônio recebe entradas dos neurônios da camada anterior, realiza uma soma ponderada das entradas e passa o resultado para a próxima camada.

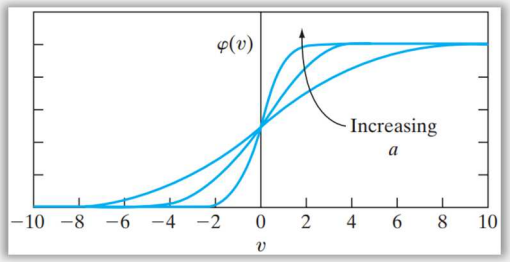


TIPOS DE FUNÇÕES DE ATIVAÇÃO

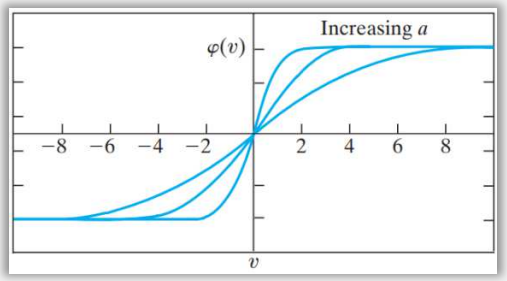
FUNÇÃO DE ATIVAÇÃO	FUNÇÕES DE LIMITE
DESCRIÇÃO DA FUNÇÃO	Essa função compara um valor com um determinado limite (nesse caso, o limite é 0) e decide se um neurônio será ativado (1) ou não será ativado (0).
REPRESENTAÇÃO DA FÓRMULA	$f(x) = \begin{cases} 0 & for\ x < 0 \\ 1 & for\ x \geqslant 0 \end{cases}$



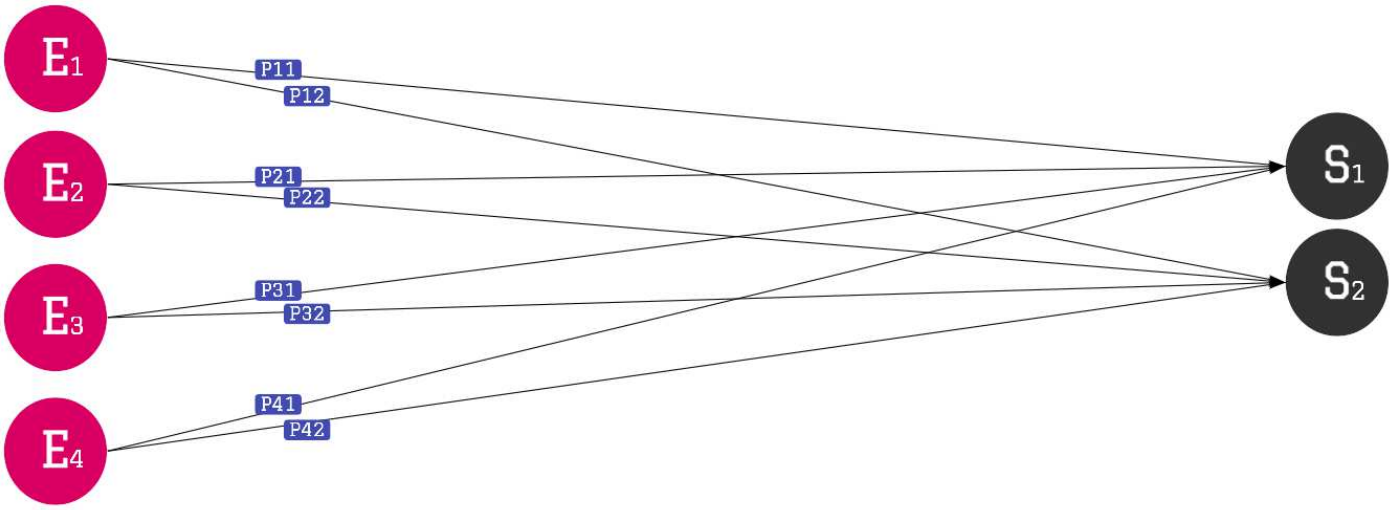
FUNÇÃO DE ATIVAÇÃO	FUNÇÕES LOGÍSTICAS
DESCRIÇÃO DA FUNÇÃO	Essa função recebe um valor real qualquer como entrada [-∞,+∞] e retorna um valor de saída entre 0 e 1.
REPRESENTAÇÃO DA FÓRMULA	$f(x) = \frac{1}{1+e^{-x}}$



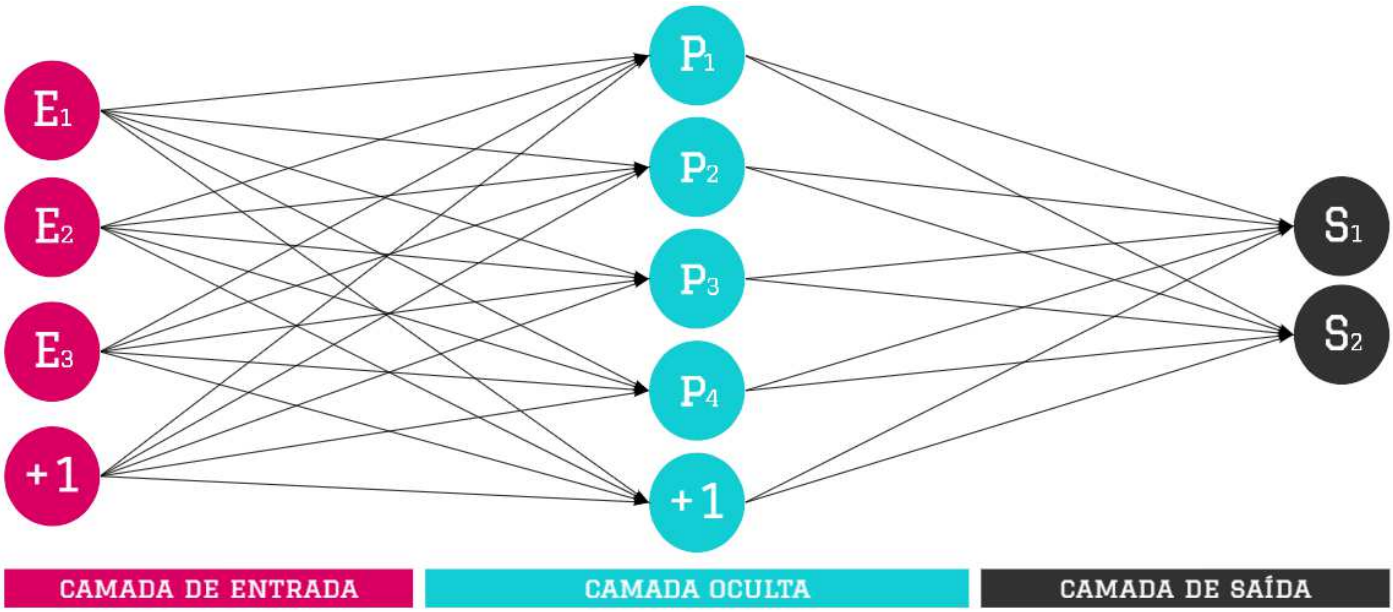
FUNÇÃO DE ATIVAÇÃO	FUNÇÃO TANGENTE HIPERBÓLICA
DESCRIÇÃO DA FUNÇÃO	Essa função recebe um valor real qualquer como entrada [-∞,+∞] e retorna um valor de saída entre -1 e 1.
REPRESENTAÇÃO DA FUNÇÃO	$f(x) = \frac{(e^{x} - e^{-x})}{(e^{x} + e^{-x})}$



SINGLE LAYER PERCEPTRON (SLP)

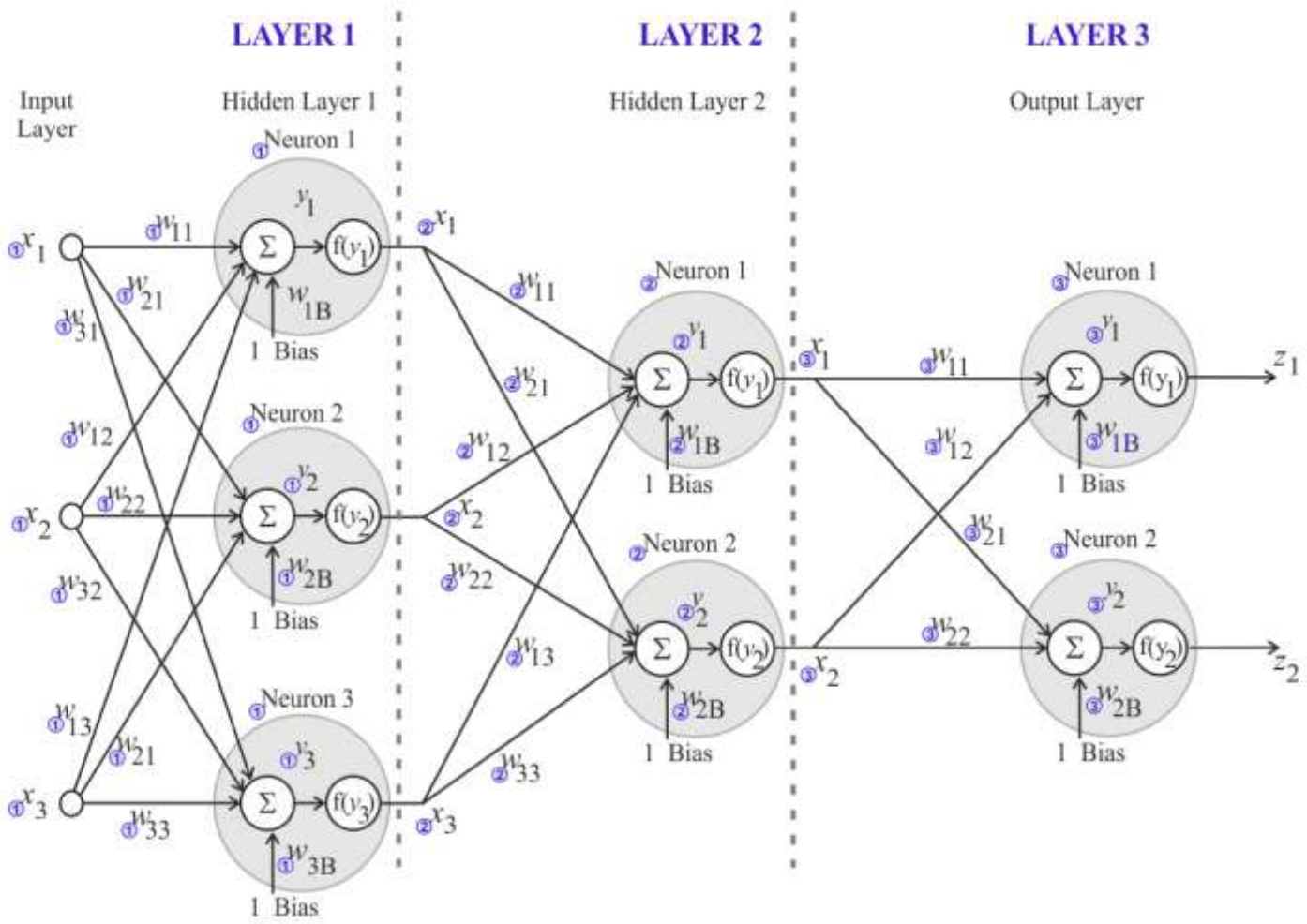


MULTIPLE LAYER PERCEPTRON (MLP)



BACKPROPAGATION

Trata-se de um algoritmo para treinamento de redes neurais artificiais que utiliza gradiente descendente para ajustar pesos na rede a fim de reduzir o custo total e minimizar os erros. O custo é calculado comparando a saída da rede com a saída desejada e calculando a diferença entre elas. A retropropagação usa o erro para calcular o gradiente da função de custo em relação aos pesos e desvios na rede e ajusta os pesos de acordo. Este processo é repetido até que o custo atinja um nível aceitável.

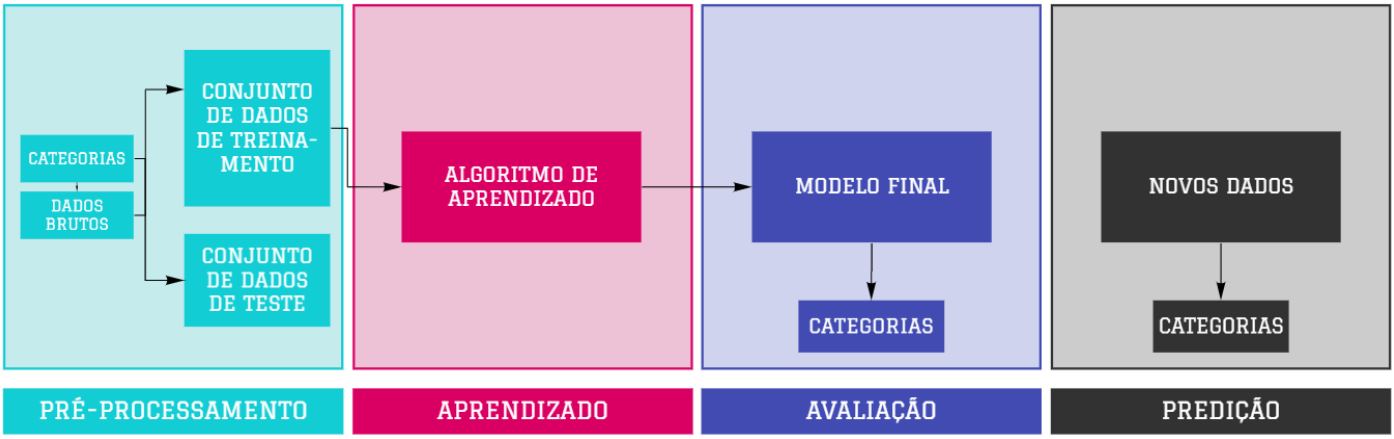


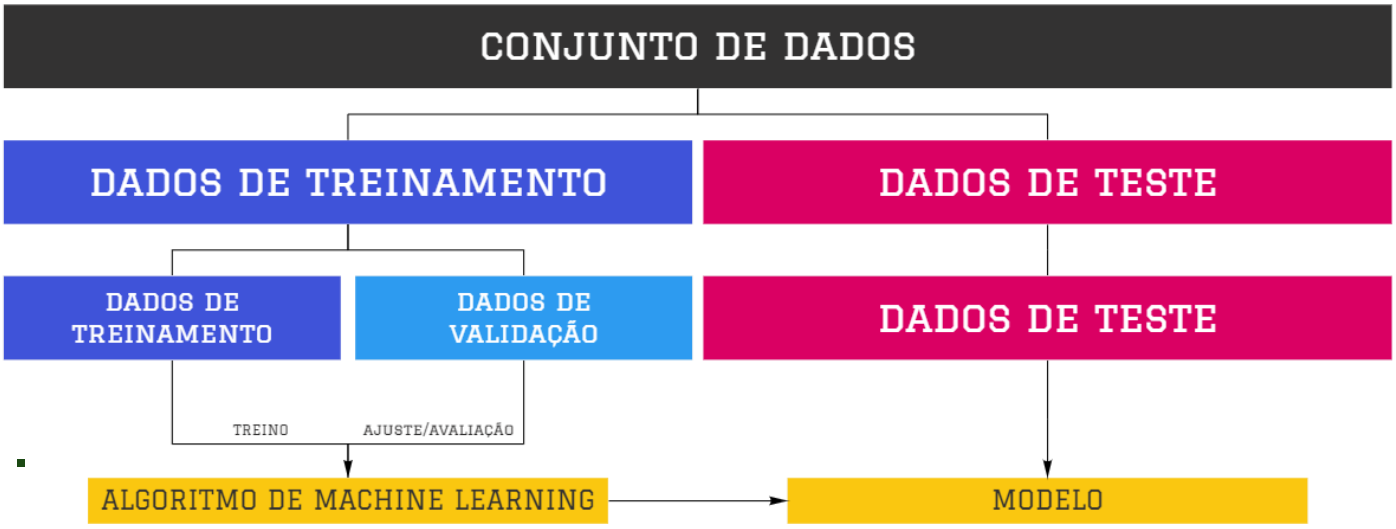
FONTES DE ERROS EM MODELOS PREDITIVOS

VALIDAÇÃO/AVALIAÇÃO DE MODELOS PREDITIVOS

VALIDAÇÃO/AVALIAÇÃO DE MODELOS PREDITIVOS

Trata-se do processo de avaliar o desempenho de um modelo preditivo em um conjunto de dados separado que não foi usado para treinamento, a fim de estimar o desempenho do modelo em dados não vistos. É uma etapa essencial no desenvolvimento de um modelo preditivo e é usado para avaliar a precisão e a confiabilidade do modelo.





VALIDAÇÃO CRUZADA

VALIDAÇÃO CRUZADA

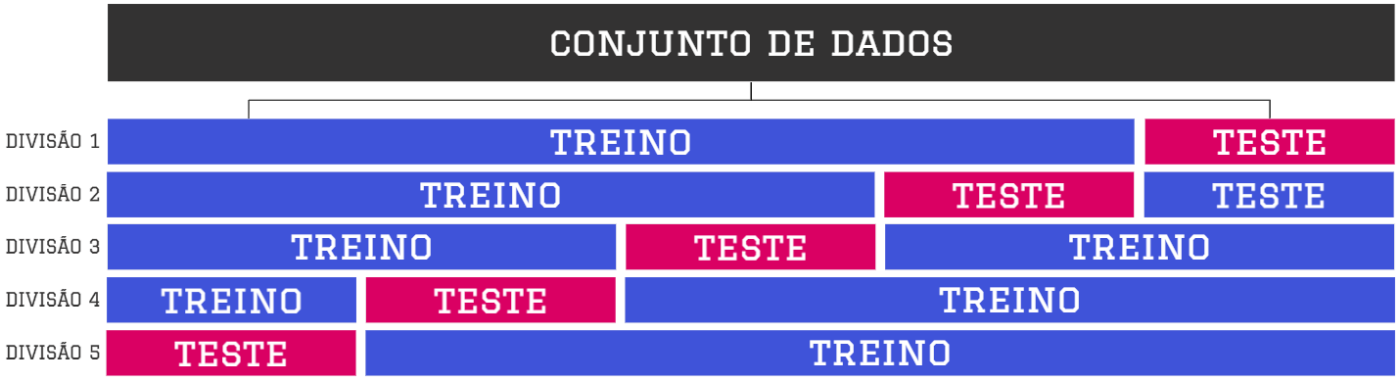
Trata-se de uma técnica usada para avaliar modelos de aprendizado de máquina. É um processo de dividir um conjunto de dados em várias partes, treinando o modelo em uma parte e testando-o em outra parte. Esse processo é repetido várias vezes, com cada parte usada como um conjunto de teste uma vez. Isso permite que o modelo seja testado em vários conjuntos de dados e fornece uma medida mais precisa de seu desempenho.

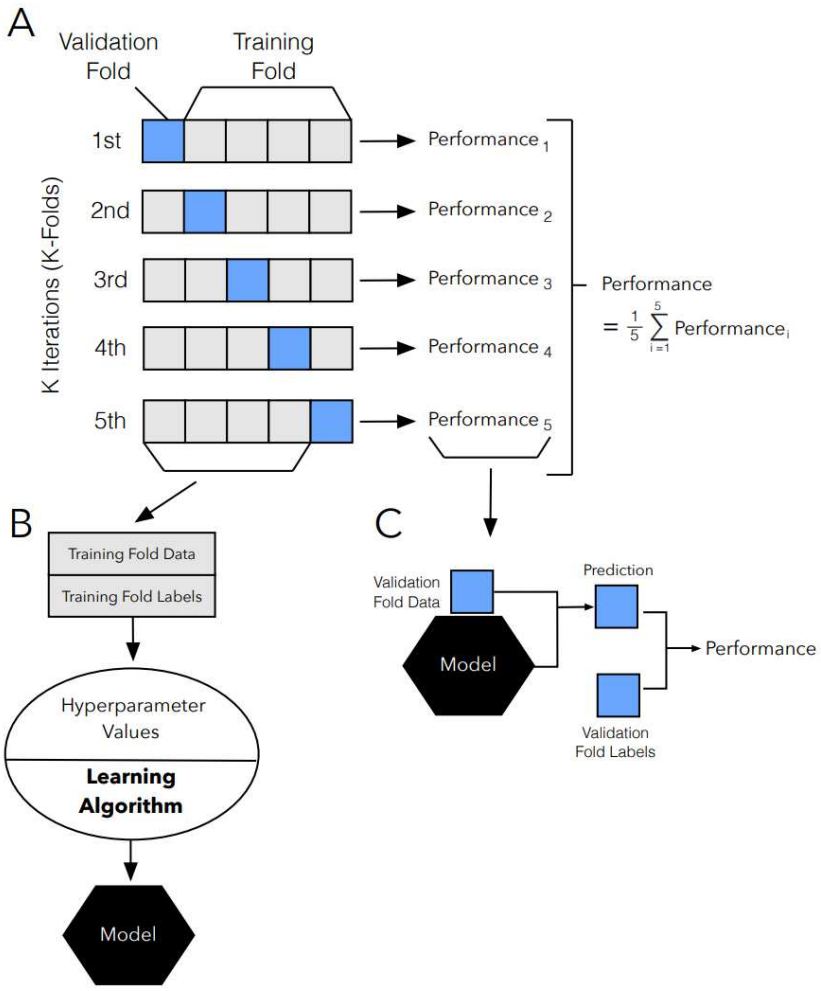
MÉTODO HOLDOUT

Trata-se de um método de validação de modelos preditivos usada para avaliar o desempenho de um modelo de aprendizado de máquina. Isso é feito dividindo os dados em um conjunto de treinamento, que é usado para treinar o modelo, e um conjunto de teste, que é usado para avaliar o modelo (o subconjunto de treinamento pode ser subdividido em um conjunto para validação, isto é, otimização/ajuste de parâmetros). O método busca fazer uma estimativa imparcial do desempenho do modelo.

MÉTODO K-FOLD

Trata-se de um método de validação de modelos preditivos usado para avaliar o desempenho de um modelo de aprendizado de máquina - além de avaliar a sua robustez. Ele envolve a divisão de um conjunto de dados em k subconjuntos de tamanhos iguais. Um subconjunto é usado para testar o modelo, enquanto os outros subconjuntos k-1 são usados para treinar o modelo. O modelo é então testado k vezes, cada vez usando um subconjunto de teste diferente. A média dos resultados é usada para medir a precisão/desempenho do modelo.





CURVA ROC

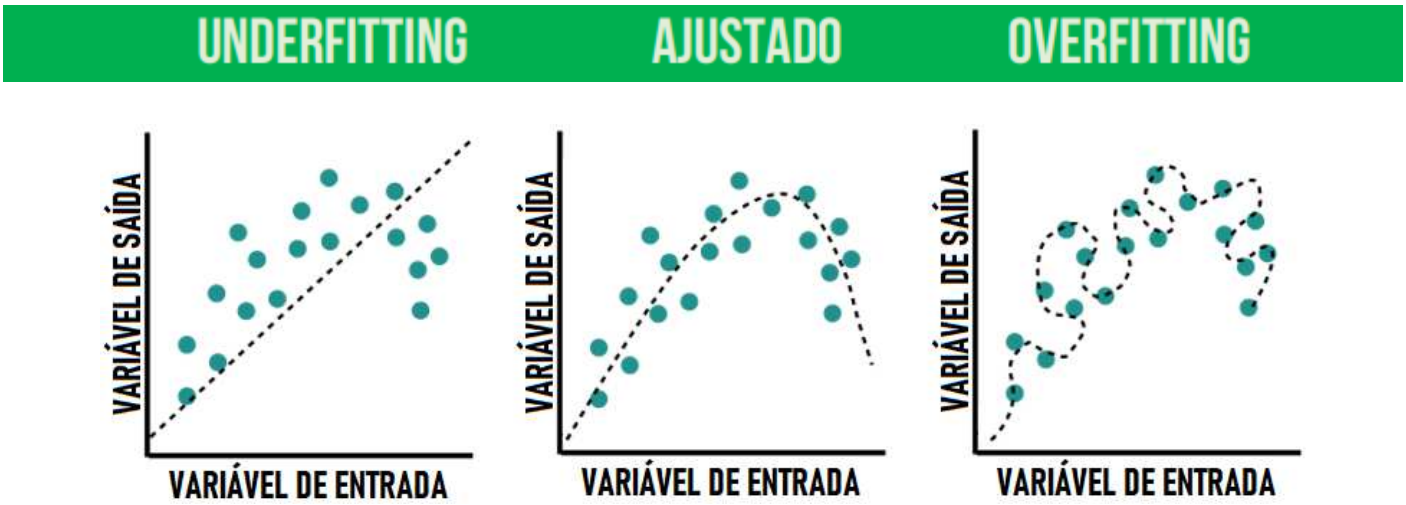
CURVA ROC (RECEIVER OPERATING CHARACTERISTIC)

Trata-se de uma representação gráfica do desempenho de um sistema de classificação binária à medida que seu limite de discriminação é variado. É um gráfico da taxa de verdadeiros-positivos em relação à taxa de falsos-positivos para um determinado limite de probabilidade. Ele exibe a capacidade de um modelo de distinguir entre uma classe positiva e uma classe negativa.

UNDERFITTING E OVERFITTING

UNDERFITTING E OVERFITTING

Underfitting ocorre quando um modelo de aprendizado de máquina não captura adequadamente o padrão subjacente dos dados. Trata-se de uma falha na captura da variação nos dados, resultando em uma representação imprecisa desses. Já o Overfitting ocorre quando um modelo de aprendizado de máquina captura muita variação nos dados, resultando em um modelo que não generaliza bem e é excessivamente sensível a pequenas alterações nos dados.



VIÉS E VARIÂNCIA

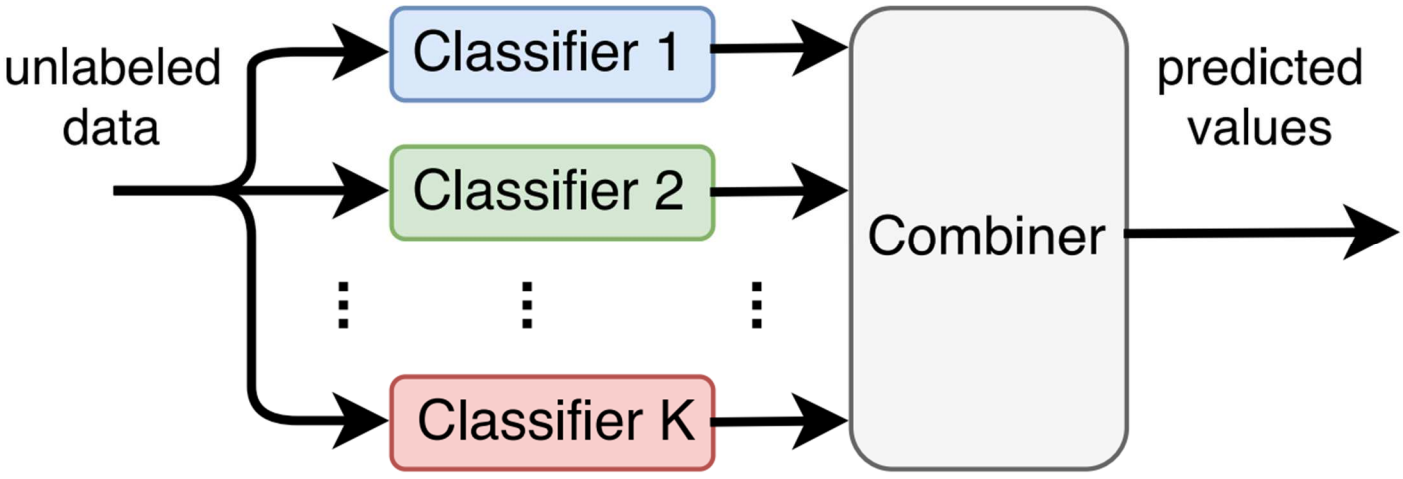
BIAS / VIÉS — TRATA-SE DA DIFERENÇA ENTRE A PREDIÇÃO (MÉDIA) DE VALOR DE UMA VARIÁVEL E O VALOR CORRETO QUE O MODELO DEVERIA PREVER. EM OUTRAS PALAVRAS, É O ERRO QUE RESULTA DE SUPOSIÇÕES IMPRECISAS DE UM MODELO.

VARIANCE / VARIÂNCIA — TRATA-SE DA SENSIBILIDADE DE UM MODELO AO SER UTILIZADO COM NOVOS CONJUNTOS DE DADOS DIFERENTES - QUÃO CONSISTENTE/VARIÁVEL É O MODELO AO SER EXECUTADO SOBRE NOVOS CONJUNTOS DE DADOS.

COMBINAÇÕES	DESCRIÇÃO	REPRESENTAÇÃO
BAIXO VIÉS E BAIXA VARIÂNCIA	Trata-se de um modelo que possui ótima precisão em suas previsões com dados de treino e que varia muito pouco quando aplicado a novos dados. Note que os pontos estão no centro do alvo e não estão espalhados.	
BAIXO VIÉS E ALTA VARIÂNCIA	Trata-se de um modelo que possui boa precisão em suas previsões com dados de treino (*overfitting*), mas que varia bastante quando aplicado a novos dados. Note que os pontos estão próximos ao centro do alvo, porém estão um pouco espalhados.	
ALTO VIÉS E BAIXA VARIÂNCIA	Trata-se de um modelo que possui péssima precisão em suas previsões com dados de treino (*underfitting*), mas que varia pouco quando aplicado a novos dados. Note que os dados estão longe do centro do alvo, porém não estão espalhados.	
ALTO VIÉS E ALTA VARIÂNCIA	Trata-se de um modelo que possui péssima precisão em suas previsões com dados de treino e que varia bastante quando aplicado a novos dados. Note que os pontos estão longe do centro do alvo e também estão bastante espalhados.	

ENSEMBLE

Trata-se de um tipo de técnica de aprendizado de máquina em que vários modelos são usados juntos para fazer previsões mais precisas do que qualquer modelo individual. Os modelos de Ensemble combinam as previsões de vários modelos para produzir melhor desempenho preditivo do que poderia ser obtido de qualquer um dos modelos individuais sozinhos. Essa técnica pode ser usada para problemas de regressão e classificação.



ENSEMBLE: BAGGING

Trata-se de uma técnica de aprendizado de máquina de ensemble utilizado para minimizar a variância de um modelo. Para tal, ele cria vários modelos que são treinados em diferentes subconjuntos dos mesmos dados e, em seguida, combina suas previsões. A ideia é que cada modelo individual tenha pontos fortes diferentes e, ao combiná-los, a precisão geral do modelo será aprimorada.

ENSEMBLE: BOOSTING

Trata-se de uma técnica de aprendizado de máquina de ensemble que combina vários weak learners para criar um strong learner. Funciona treinando cada weak learner em uma sequência, com cada aluno subsequente focando nos erros cometidos pelo aluno anterior. A saída dos weak learners é combinada para produzir um único strong learner. Boosting é freqüentemente usado para melhorar a precisão de modelos que são propensos a underfitting.

ENSEMBLE: STACKING

Trata-se de uma técnica de aprendizado de máquina de ensemble que difere do bagging e boosting por utilizar um conjunto heterogêneo de weak learners e também por utilizar a saída dos weak learners como entrada em um meta-modelo com o objetivo de aprender o mapeamento entre as saídas e as classes corretas.

TÉCNICAS DE REGULARIZAÇÃO

REGULARIZAÇÃO

Trata-se do controle fino do nível de complexidade de um dado modelo, limitando seu grau de liberdade (flexibilidade) para se ajustar aos dados de treinamento, visando evitar sobreajustamento (overfitting). Em outras palavras, essa técnica reduz a variância de um modelo, tornando-o mais generalizável.

REGRESSÃO LASSO (L1)

Trata-se de uma técnica de regularização chamada Least Absolute Shrinkage and Selection Operator (LASSO) para reduzir a complexidade de um modelo preditivo. Essa técnica reduz certos coeficientes a zero, eliminando assim as variáveis menos importantes do modelo e ajudando a reduzir o overfitting. O resultado é um modelo esparso que é mais fácil de interpretar e melhor na generalização para novos dados.

REGRESSÃO RIDGE (L2)

Trata-se de uma técnica de regularização que adiciona um termo de penalidade à função de custo para reduzir a complexidade do modelo. O termo de penalidade é um parâmetro de regularização que reduz as estimativas do coeficiente para zero. Essa técnica é útil para prevenir o overfitting e melhorar a precisão do modelo. Ela também pode ser usada para identificar variáveis importantes em um conjunto de dados.

PRUNING

Trata-se de uma técnica de reegularização utilizada para reduzir a complexidade de um modelo removendo parâmetros ou conexões desnecessárias em uma rede. Isso ajuda a melhorar a precisão do modelo preditivo, reduzindo o overfitting, melhorando a velocidade de treinamento e reduzindo os requisitos de memória.

DROPOUT

Trata-se de uma técnica de regularização usada em redes neurais para evitar o overfitting. Ele funciona desconectando aleatoriamente uma fração das unidades de entrada durante o treinamento, o que força o modelo a aprender com menos parâmetros e reduz a sua complexidade. A fração de unidades de entrada a serem descartadas é normalmente definida como uma porcentagem entre 20% e 50%.

EARLY STOPPING

Trata-se de uma técnica regularização usada em redes neurais para evitar o overfitting dos dados de treinamento. Ele funciona monitorando o desempenho do modelo em um conjunto de dados de validação durante o treinamento e interrompendo o processo de treinamento quando o desempenho no conjunto de dados de validação não melhora por um determinado número de épocas de treinamento.

DATA AUGMENTATION

Trata-se de uma técnica de regularização usada para aumentar artificialmente o tamanho de um conjunto de dados de treinamento, manipulando e adicionando transformações aleatórias aos dados existentes. Isso ajuda a melhorar a precisão e a robustez dos modelos de aprendizado de máquina, introduzindo novas variações nos dados de treinamento. As técnicas de aumento de dados incluem corte, inversão, rotação, adição de ruído, deslocamento, etc.

OTIMIZAÇÃO DE HIPERPARÂMETROS

OTIMIZAÇÃO DE HIPERPARÂMETROS

Trata-se do processo de selecionar o melhor conjunto de hiperparâmetros para um determinado algoritmo de aprendizado de máquina, a fim de maximizar seu desempenho em um determinado conjunto de dados. Isso geralmente é feito por meio de um processo de tentativa e erro, usando métodos como pesquisa em grade, pesquisa aleatória ou otimização bayesiana.

PARÂMETROS	TRATA-SE DE REPRESENTAÇÕES INTERNAS DO MODELO AJUSTADAS AUTOMATICAMENTE PELO PROCESSO DE APRENDIZAGEM OU TREINAMENTO, SINTETIZADOS A PARTIR DE PADRÕES ESTATÍSTICOS DOS DADOS, TAIS COMO OS PESOS DE UMA REGRESSÃO OU DE UMA REDE NEURAL. EM OUTRAS PALAVRAS, TRATA-SE DE UMA VARIÁVEL DE CONFIGURAÇÃO INTERNA A UM MODELO E CUJOS VALORES PODEM SER ESTIMADOS A PARTIR DOS DADOS.
HIPER PARÂMETROS	TRATA-SE DAS CARACTERÍSTICAS DO MODELO OU DO SEU PROCESSO DE TREINAMENTO QUE REFLETEM UMA OPÇÃO DO CIENTISTA, QUE NÃO SÃO EXTRAÍDOS AUTOMATICAMENTE DOS DADOS, TAIS COMO A INTENSIDADE DE REGULARIZAÇÃO DE UM MODELO, A PROFUNDIDADE MÁXIMA DE ÁRVORES DE DECISÃO, ETC. EM OUTRAS PALAVRAS, TRATA-SE DE UMA VARIÁVEL DE CONFIGURAÇÃO EXTERNA A UM MODELO E CUJOS VALORES NÃO PODEM SER ESTIMADOS A PARTIR DOS DADOS.

GRID SEARCH

Trata-se de uma técnica usada para otimizar hiperparâmetros de um determinado modelo, a fim de obter o melhor desempenho possível. É uma pesquisa exaustiva sobre um conjunto de hiperparâmetros especificados manualmente pelo usuário. Funciona explorando sistematicamente cada combinação dos parâmetros de uma grade especificada e avaliando o desempenho do modelo para cada combinação. A combinação com a pontuação mais alta é então selecionada como o modelo de melhor desempenho.

RANDOM SEARCH

Trata-se de uma técnica usada para otimizar hiperparâmetros que envolve a amostragem aleatória de uma gama de valores de hiperparâmetros possíveis e, em seguida, a seleção do melhor conjunto com base nos resultados. É um dos métodos mais simples e eficientes de ajustar um modelo e é frequentemente usado como linha de base para comparação com técnicas de otimização mais avançadas.

BAYESIAN SEARCH

Trata-se de uma técnica usada para otimizar hiperparâmetros que usa a inferência bayesiana para construir um modelo probabilístico de um espaço de pesquisa. Ele usa esse modelo para otimizar o processo de busca, levando em consideração a incerteza do problema e orientando a busca para melhores soluções. É particularmente útil para problemas onde o espaço de busca é grande e o custo de avaliação de uma única solução é alto.

SEPARABILIDADE DE DADOS

SEPARABILIDADE DE DADOS

Trata-se da capacidade de separar dados em grupos distintos e separados. É uma medida de quão bem os dados podem ser divididos em clusters, classes ou categorias distintas e uma ferramenta útil para agrupamento e algoritmos de classificação, pois a separabilidade de dados pode ajudar a identificar e classificar pontos de dados em grupos específicos.

REDUÇÃO DE DIMENSIONALIDADE

Trata-se do processo de redução do número de features em um conjunto de dados, mantendo as informações mais importantes. É usado para reduzir a complexidade de um conjunto de dados enquanto ainda preserva as características essenciais dos dados; além de ser usado no aprendizado de máquina para reduzir o overfitting, reduzir o tempo de computação e melhorar a precisão dos modelos.

VANTAGENS DA REDUÇÃO DE DIMENSIONALIDADE

Simplificação dos modelos de aprendizado de máquina: é mais fácil ajustar um modelo que tenha duas variáveis de entrada do que um modelo que tenha 80 variáveis de entrada.
Redução do *overfitting*: é muito mais difícil ocorrer sobreajuste em um modelo de aprendizado de máquina com menos variáveis do que com muitas variáveis.
Simplificação da representação gráfica: visualizar dados representados em mais de três dimensões é inviável para seres humanos.
Redução do custo computacional: com menos variáveis, é necessário utilizar menos recursos computacionais para realizar o treinamento de um modelo de aprendizado de máquina.
Redução do tempo de treinamento: como há menos variáveis para ajustar, leva menos tempo para treinar o modelo de aprendizado de máquina.
Aumentar a performance: como há variáveis com nenhuma correlação, sua eliminação ajuda a melhorar o desempenho do modelo de aprendizado de máquina.

SELEÇÃO DE VARIÁVEIS

Trata-se do processo de selecionar um subconjunto de variáveis relevantes para uso na construção do modelo, isto é, aquelas que têm o relacionamento mais forte com a variável dependente e também que são mais úteis para a previsão. Busca-se selecionar aquelas que possuem maior poder preditivo, descartando as demais. Essa técnica é importante porque pode ajudar a reduzir a complexidade dos modelos e melhorar a precisão das previsões.

MÉTODO FILTER

Trata-se de uma técnica de seleção de variáveis que usa várias métricas estatísticas para avaliar a importância dos recursos e selecionar os recursos mais relevantes para a construção de um modelo preditivo. Ela ajuda a reduzir a dimensionalidade dos dados para melhorar a precisão do modelo. Em geral, é usado em combinação com outras técnicas.

MÉTODO WRAPPER

Trata-se de uma técnica de seleção de variáveis que avalia o conjunto de features escolhido otimizando uma métrica de desempenho. Ele funciona selecionando subconjuntos de features e, em seguida, avaliando o desempenho de um modelo treinado nesses recursos. O objetivo é encontrar o conjunto de features que produz o modelo de melhor desempenho.

MÉTODO EMBEDDED

Trata-se de uma técnica de seleção de variáveis que combina as qualidades do método filter e do método wrapper para treinar um classificador em um subconjunto de features e, em seguida, adicionar recursos adicionais ao modelo conforme necessário. O algoritmo usará a melhor combinação de features para gerar as previsões mais precisas.

FATORAÇÃO DE MATRIZES

Trata-se de uma técnica utilizada para fatorar a matriz original em outras matrizes menores a fim de encontrar o melhor subconjunto de dados com menor dimensionalidade que seja capaz de representar a matriz original. Ela pode ser usada para recomendar itens aos usuários, agrupar itens semelhantes e reduzir a dimensionalidade de um conjunto de dados.

SISTEMAS DE RECOMENDAÇÃO

Trata-se de um algoritmo que faz previsões sobre os interesses de um usuário. Esses sistemas usam dados históricos do usuário e dados de itens específicos para fazer recomendações aos usuários sobre itens nos quais eles podem estar interessados. O objetivo de um sistema de recomendação é fornecer recomendações personalizadas e relevantes aos usuários.

PCA (PRINCIPAL COMPONENT ANALYSIS)

Trata-se de um método estatístico usado para reduzir a dimensionalidade dos dados por meio da transformação de um grande conjunto de variáveis em um conjunto menor de variáveis, chamado de Componentes Principais, enquanto retém o máximo possível da variação dos dados. O PCA é amplamente para interpretar as características mais importantes de um conjunto de dados.

QUESTÕES COMENTADAS – CESPE

1. **(CESPE / Petrobrás – 2022)** A tabela abaixo apresenta parte de um conjunto de dados referentes a uma variável categórica chamada opinião que possui quatro categorias de resposta: muito satisfeito, satisfeito, insatisfeito e muito insatisfeito.

observação	opinião
1	satisfeito
2	muito satisfeito
3	insatisfeito
4	insatisfeito
5	muito insatisfeito
6	satisfeito

Com base nessas informações, julgue o próximo item.

Para lidar numericamente com os dados categóricos, uma codificação binária proporciona uma conversão de cada categoria de resposta para uma sequência de dígitos binários, em que cada dígito binário representa uma variável numérica que assume valores 0 ou 1. Na situação em tela, uma possível codificação binária é exemplificada na tabela abaixo.

observação	D1	D2	D3
1	1	0	0
2	0	1	0
3	0	0	1
4	0	0	1
5	0	0	0
6	1	0	0

Comentários:

Perfeito! Vejam que temos o seguinte mapeamento das observações/opiniões:

- D1 = Satisfeito: 1 e 6
- D2 = Muito Satisfeito: 2
- D3 = Insatisfeito:3 e 4
- = Muito Insatisfeito: 5

Note que, apesar de termos quatro categorias, não precisamos de quatro colunas para representa-las porque tudo que não for classificada como "Satisfeito", "Muito Satisfeito" e "Insatisfeito, é obrigatoriamente "Muito Insatisfeito".

Essa é uma forma de representar numericamente os dados categóricos por meio de uma codificação binária que converte cada categoria como 0 ou 1.

Gabarito: Correto

2. **(CESPE / Petrobrás – 2022)** Em um processo em que se utiliza a ciência de dados, o número de variáveis necessárias para a realização da investigação de um fenômeno é direta e simplesmente igual ao número de variáveis utilizadas para mensurar as respectivas características desejadas; entretanto, é diferente o procedimento para determinar o número de variáveis explicativas, cujos dados estejam em escalas qualitativas.

   Considerando esse aspecto dos modelos de regressão, julgue o item a seguir.

   Para evitar um erro de ponderação arbitrária, deve-se recorrer ao artifício de uso de variáveis dummy, o que permitirá a estratificação da amostra da maneira que for definido um determinado critério, evento ou atributo, para então serem inseridas no modelo em análise; isso permitirá o estudo da relação entre o comportamento de determinada variável explicativa qualitativa e o fenômeno em questão, representado pela variável dependente.

Comentários:

Vamos por partes porque essa questão é complexa! "*O número de variáveis necessárias para a realização da investigação de um fenômeno é direta e simplesmente igual ao número de variáveis utilizadas para mensurar as respectivas características desejadas"?* Sim, as variáveis em ciência de dados representam as características ou atributos que desejamos mensurar – são as chamadas variáveis independentes. Elas são utilizadas para calcular a variável dependente!

"*... entretanto, é diferente o procedimento para determinar o número de variáveis explicativas, cujos dados estejam em escalas qualitativas"?* Sim, variáveis explicativas que estejam em escalas qualitativas são as variáveis categóricas, logo temos um procedimento diferente para esse tipo de variável dado que elas não são numéricas.

"*Para evitar um erro de ponderação arbitrária, deve-se recorrer ao artifício de uso de variáveis dummy*"? Sim, conforme vimos em aula, se atribuirmos um valor (ordenado ou não) às categorias de um problema de aprendizado de máquina, podemos cair no erro de ponderação arbitrária, em que os pesos atribuídos geram informações falsas sobre os atributos (Ex: mulher = 1, homem =2. *Ora, homem tem o dobro do peso?* Não)! Para resolver, podemos utilizar variáveis *dummy*, que são variáveis que assumem os valores 0 ou 1 para representar a ausência ou presença de um determinado atributo, característica, categoria, evento ou critério.

"*o que permitirá a estratificação da amostra da maneira que for definido um determinado critério, evento ou atributo, para então serem inseridas no modelo em análise*"? Sim, para cada atributo (Ex: Cor), vamos estratificá-lo em vários outros atributos (Ex: vermelho, verde e azul). Essa estratificação que transforma uma variável (categórica) em várias variáveis (numéricas) agora pode finalmente ser inserida no modelo.

"Isso permitirá o estudo da relação entre o comportamento de determinada variável explicativa qualitativa e o fenômeno em questão, representado pela variável dependente"? Sim, ao utilizar variáveis *dummy*, podemos transformar dados categóricos em dados numéricos e finalmente utilizá-las para explicar um fenômeno (variável dependente).

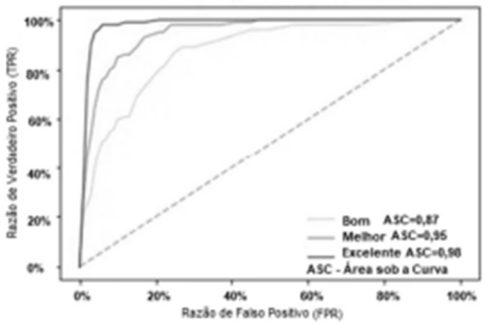
Gabarito: Correto

---

3. **(CESPE / Petrobrás – 2022)** As métricas de avaliação de desempenho de um modelo de aprendizado de máquina, que é um componente integrante de qualquer projeto de ciência de dados, destinam-se a estimar a precisão da generalização de um modelo sobre os dados futuros (não vistos ou fora da amostra). Dentre as métricas mais conhecidas, estão a matriz de confusão, precisão, recall, pontuação, especificidade e a curva de características operacionais do receptor (ROC).

   Acerca das características específicas dessas métricas, julgue o próximo item.

   As curvas ROC a seguir mostram a taxa de especificidade (verdadeiros positivos) versus a taxa de sensibilidade (falsos positivos) do modelo adotado; a linha tracejada é a linha de base da métrica de avaliação e define uma adivinhação aleatória.



Comentários:

Opa! Elas realmente mostram a taxa de especificidade versus a taxa de sensibilidade, no entanto a especificidade trata dos falsos-positivos e a sensibilidade trata dos verdadeiros-positivos.

Gabarito: Errado

---

4. **(CESPE / Petrobrás – 2022)** O algoritmo de backpropagation consiste das fases de propagação e de retro propagação: na primeira, as entradas são passadas através da rede e as previsões de saída são obtidas; na segunda, se calcula o termo de correção dos pesos e, por conseguinte, a atualização dos pesos.

Comentários:

Perfeito! Conforme vimos em aula, temos:

1. Etapa de Propagação: as entradas fluem através das camadas ocultas da rede neural e previsões são obtidas na camada de saída;

2. Etapa de Retropropagação: calcula-se o gradiente da função de custo/perda na camada de saída e ele é utilizado para atualizar os pesos (inclusive o viés) recursivamente.

Gabarito: Correto

---

5. **(CESPE / ANP – 2022)** A ação de realizar agrupamento hierárquico tem como premissa básica encontrar elementos em um conjunto de dados que impliquem a presença de outros elementos na mesma transação, com um grau de certeza definido pelos índices de fator de suporte e o fator de confiança, que pode ser realizado, por exemplo, por meio do algoritmo a priori.

Comentários:

A ação de ~~realizar agrupamento hierárquico~~ **construir regras de associação** tem como premissa básica encontrar elementos em um conjunto de dados que impliquem a presença de outros elementos na mesma transação, com um grau de certeza definido pelos índices de fator de suporte e o fator de confiança, que pode ser realizado, por exemplo, por meio do algoritmo a priori.

Gabarito: Errado

---

6. **(CESPE / ANP – 2022)** O algoritmo *random forest* é um algoritmo de aprendizado de máquina supervisionado em que se agrupam os resultados de várias árvores de decisão de cada nó para se obter uma conclusão própria e aumentar a precisão do modelo, não sendo o referido algoritmo adequado para grandes conjuntos de dados.

Comentários:

O algoritmo Random Forest é um algoritmo de aprendizado de máquina? Sim! *Ele é supervisionado?* Sim, trata-se de um algoritmo de classificação. *Ele agrupa os resultados de várias árvores de decisão de cada nó para se obter uma conclusão própria e aumentar a precisão do modelo?* Sim, por meio da utilização de diversas árvores aleatórias. *Não é adequado para grandes conjuntos de dados?* Opa, ele é ótimo para lidar com grandes conjuntos de dados.

Gabarito: Errado

---

7. **(CESPE / ANP – 2022)** A técnica de redução de dimensionalidade (PCA) permite transformar dados que inicialmente pertencem a um espaço de dimensão n em um espaço de dimensão m, em que m < n, sendo utilizada, por exemplo, para reduzir a dimensionalidade de certo conjunto de dados através do descarte de características não úteis e que ainda permita realizar o reconhecimento de padrões.

Comentários:

Perfeito! A redução de dimensionalidade busca justamente reduzir as dimensões eliminando características que não serão úteis para o reconhecimento de padrões. Por meio da seleção de atributos, é possível eliminar atributos irrelevantes ou redundantes.

Gabarito: Correto

---

8. **(CESPE / ANP – 2022)** As aplicações em inteligência artificial são definidas como uma subárea da área de aprendizagem de máquina (*machine learning*).

Comentários:

Opa! É o inverso: as aplicações de aprendizagem de máquina (*machine learning*) são definidas como uma subárea da área de inteligência artificial.

Gabarito: Errado

---

9. **(CESPE / ANP – 2022)** Em se tratando de modelos de regressão linear, indica-se a utilização dos seguintes métodos não paramétricos para a estimação dos resultados: Mínimos Quadrados (MQ) e de Support Vector Machines (SVM).

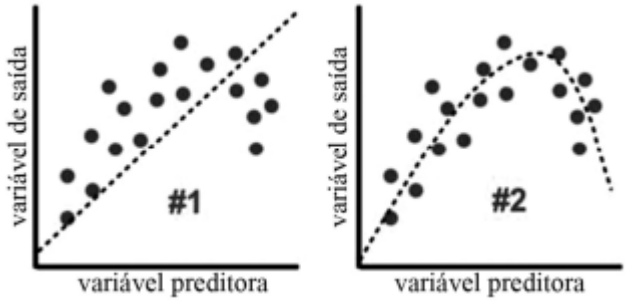
Comentários:

O Método dos Mínimos Quadrados é paramétrico e o SVM, no contexto de regressão linear, também é considerado paramétrico.

Gabarito: Errado

---

10. **(CESPE / ANP – 2022)** Considerando-se, nos gráficos a seguir, que o resultado #2 corresponda ao melhor desempenho do algoritmo, é correto afirmar que o resultado #1 indica que houve *underfitting*.



Comentários:

Vejam o resultado #2: a curva acompanha os dados oferecendo o melhor desempenho do algoritmo. Já no resultado #1, a reta não se ajusta bem aos dados, porque considera-se que houve um *underfitting* (subajuste), isto é, o modelo é excessivamente simples para modelar a real complexidade do problema para novos dados.

Gabarito: Correto

Determinado parâmetro *β* será estimado recursivamente com a ajuda de um método de otimização matemática com base em uma função objetivo *g(β)*. Para essa estimação, a base de dados de treinamento consistirá de *n* observações.

11. (CESPE / SERPRO – 2021) Em cada iteração na estimação do parâmetro β, o método do gradiente descendente requer n observações da base de treinamento, ao passo que o método do gradiente descendente estocástico utiliza uma observação selecionada aleatoriamente dessa base de treinamento.

Comentários:

Perfeito! O método do gradiente descendente (em lote) requer toda a base de treinamento, isto é, n observações. Ao passo que o gradiente descendente estocástico utiliza uma observação aleatória.

Gabarito: Correto

---

12. (CESPE / SERPRO – 2021) Entre as condições ideais relativas à função objetivo g(β) para a aplicação do método do gradiente descendente incluem-se convexidade, continuidade e diferenciabilidade.

Comentários:

Perfeito! O método do gradiente descendente (em lote) precisa que a função de custo/perda seja convexa, contínua e diferenciável. Uma função contínua é aquela que não possui pontos de descontinuidade; e uma função é diferenciável quando é possível calcular as derivadas (conceito matemática que mede a taxa de variação de uma função em um ponto) de seus pontos. Logo, o gradiente descendente é mais adequado quando a função de custo/perda tem um único ponto mínimo, não possui saltos de descontinuidades e pode-se calcular as derivadas de seus pontos.

Gabarito: Correto

---

13. (CESPE / SERPRO – 2021) O método do gradiente descendente é equivalente ao método de Newton-Raphson, no qual o incremento, para a estimação do parâmetro β, depende da primeira e da segunda derivada da função objetivo g(β).

Comentários:

Não é necessário saber o que é o método de Newton-Raphson para responder à questão. Basta saber que para a estimação de um parâmetro, o gradiente descendente depende apenas da primeira derivada da função objetivo (nome genérico de funções matemáticas usadas para modelar problemas de optimização).

Gabarito: Errado

---

14. (CESPE / SERPRO – 2021) O gradiente descendente em lote é um método probabilístico de otimização no qual, para cada iteração, encontram-se L × n observações geradas mediante amostragem (com reposição) da base de dados de treinamento (em que L representa o número de lotes, com L > 1).

Comentários:

Opa... quem usa amostragem da base de dados de treinamento é o gradiente descendente estocástico.

Gabarito: Errado

---

15. (CESPE / SEFAZ-AL - 2021) A regressão logística é um modelo de regressão no qual a relação entre as variáveis independentes e a variável dependente é representada por uma função degrau, a qual, por sua vez, pode ser representada por uma *spline*.

Comentários:

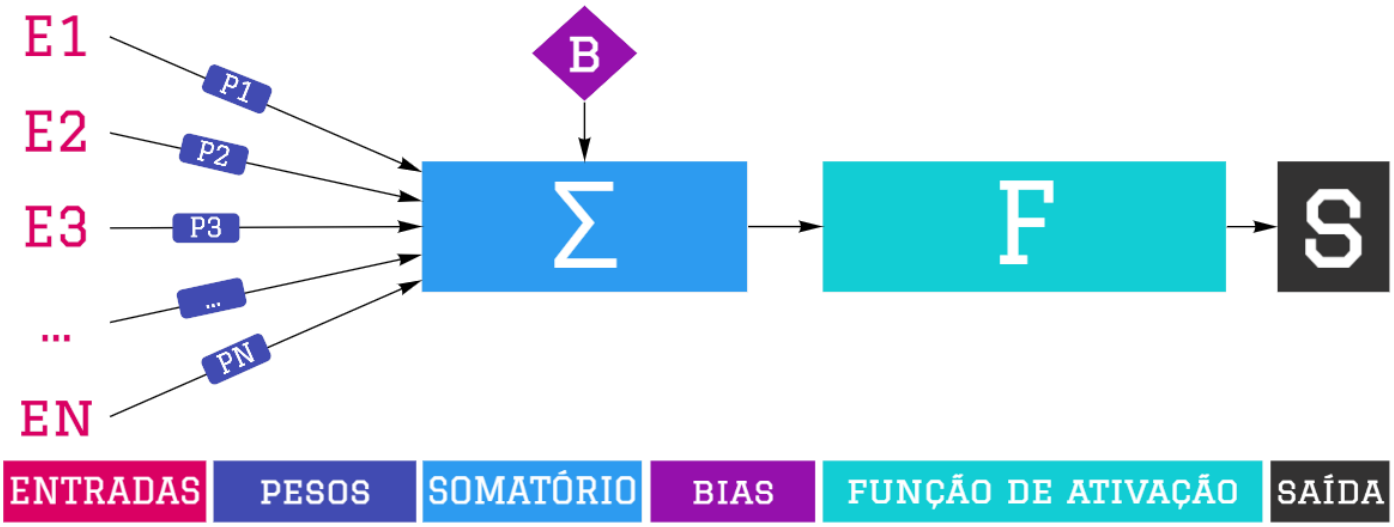
A regressão logística se caracteriza por ser estritamente crescente e ser representada por uma função em S. Quem se caracteriza por não ser estritamente crescente e ser representada por uma função degrau (ou *step*) é a função limite. Por fim, *spline* é uma interpolação contínua que arredonda uma função – não faz sentido no contexto da questão.

Gabarito: Errado

---

16. (CESPE / SEFAZ-CE – 2021) Cada unidade de uma rede neural artificial possui um valor e um peso, no seu nível mais básico, para indicar sua importância relativa.

Comentários:



Note que cada perceptron (unidade de uma rede neural artificial) possui vários valores de entrada com seus respectivos pesos para indicar sua importância relativa, isto é, qual grau de influência de cada entrada. A questão foi considerada como correta, mas uma redação mais adequada enfatizaria que se trata de vários valores e pesos.

Gabarito: Correto

---

17. (CESPE / SEFAZ-CE – 2021) Redes neurais do tipo LSTM (long short-term memory) mantêm o nível de precisão independentemente do tamanho do modelo utilizado.

Comentários:

Na verdade, independentemente do tipo de rede neural, o tamanho importa! Em geral, quanto mais unidades, mais complexa e mais capaz de modelar problemas complexos. Da mesma forma, quanto menos unidades, menos complexa e menos capaz de modelar problemas complexos. Logo, o tamanho do tamanho interfere no nível de precisão.

Gabarito: Errado

---

18. (CESPE / TJ-AM – 2019) A técnica *machine learning* pode ser utilizada para apoiar um processo de *data mining*.

Comentários:

Perfeito! Data Mining é o processo de coleta, análise e extração de informações de conjuntos de dados para descobrir padrões e correlações. É bastante usado para descobrir tendências e fazer previsões. Já Machine Learning é o processo de utilizar algoritmos para analisar dados e fazer previsões com base nos padrões descobertos. É bastante usado para identificar correlações e relacionamentos entre diferentes variáveis. Notem que há uma imensa correlação entre ambos,

sendo que em diversos momentos esses conceitos se confundem. Logo, o aprendizado de máquina evidentemente pode ser utilizado para apoiar um processo de mineração de dados.

Gabarito: Correto

---

19. (CESPE / TCE-MG – 2018) Em machine learning, a categoria de aprendizagem por reforço identifica as tarefas em que:

a) um software interage com um ambiente dinâmico, como, por exemplo, veículos autônomos.
b) as etiquetas de classificação não sejam fornecidas ao algoritmo, de modo a deixá-lo livre para entender as entradas recebidas.
c) o aprendizado pode ser um objetivo em si mesmo ou um meio para se atingir um fim.
d) o objetivo seja aprender um conjunto de regras generalistas para converter as entradas em saídas predefinidas.
e) são apresentados ao computador exemplos de entradas e saídas desejadas, fornecidas por um orientador.

Comentários:

(a) Correto! Veículos autônomos seguem uma rota de trânsito em um ambiente dinâmico até o seu destino final. O carro precisa escolher o melhor caminho, evitando colisões e infrações de trânsito, sendo penalizado quando não cumpre seu papel corretamente e recompensado quando o faz. Essa é uma utilização típica de aprendizado por reforço; (b) Errado, esse seria o comportamento típico de uma tarefa não supervisionada; (c) Errado, esse seria o comportamento típico do aprendizado não supervisionado, em que o aprendizado pode ser um objetivo em si mesmo (descobrir novos padrões nos dados) ou um meio para atingir um fim; (d) Errado, esse seria o comportamento típico do aprendizado supervisionado, em que se busca aprender uma regra geral que mapeia entradas de dados em saídas de dados; (e) Errado, esse seria o comportamento típico do aprendizado supervisionado, em que são apresentadas ao computador exemplos de entradas de dados e suas respectivas saídas desejadas por um supervisor/professor/orientador.

Gabarito: Letra A

---

20. (CESPE / TRE/PE – 2017 – Item A) Em uma curva ROC, o ponto que aperfeiçoa a sensibilidade em função da especificidade é aquele que se encontra mais próximo do canto superior esquerdo do gráfico.

Comentários:

Perfeito! Idealmente, o ponto que aperfeiçoa a sensibilidade em função da especificidade é aquele mais distante do classificador aleatório, próximo do canto superior esquerdo.

Gabarito: Correto

---

21. **(CESPE / INPI – 2013)** Em um banco de dados, foram armazenadas informações relativas a diversas pesquisas realizadas por pesquisadores de institutos renomados. Entre as variáveis constantes desse banco destacam-se: nome, gênero e titulação do pesquisador; valor financiado da pesquisa; instituto ao qual o pesquisador pertence; número de componentes da equipe; e número de artigos publicados pelo pesquisador.

Com base nessas informações, julgue os itens a seguir.

Se, na análise de componentes principais, fossem utilizadas 5 variáveis quantitativas, então, a técnica geraria, no máximo, 3 componentes, se essas correspondessem a, pelo menos, 95% da variância explicada.

Comentários:

A técnica pode gerar uma quantidade de componentes principais menor ou igual a quantidade de variáveis. Logo, poderia gerar quatro ou cinco!

Gabarito: Errado

---

22. **(CESPE / MPE-PI – 2012)** Em datamining, o uso de holdout induz o modelo de classificação a partir do conjunto de treinamento, e seu desempenho é avaliado no conjunto de teste. Quanto menor o conjunto de treinamento, maior a variância do modelo; no entanto, se o conjunto de treinamento for grande demais, a precisão estimada calculada a partir do conjunto menor é menos confiável.

Comentários:

Vamos lá! *O que é hold-out?* É uma técnica para avaliar modelos preditivos que divide o conjunto de dados em duas partes: dados de treinamento e dados de teste. *Dito isso, ele induz o modelo de classificação a partir do conjunto de treinamento?* Eu honestamente não faço ideia do que o examinador quis dizer com o verbo "*induzir*". *O desempenho do modelo de classificação é avaliado no conjunto de teste?* Sim, os dados de teste são utilizados para avaliar a performance do modelo.

Agora pensem comigo: eu tenho um conjunto de dados e vou dividi-lo em dados de treinamento e dados de avaliação. Dito isso, temos que:

- Quanto maior for o conjunto de dados de treinamento, mais dados meu modelo terá disponível para treinar, mais precisa será sua predição nos dados de treinamento, logo menor será a variância dos dados (menos erros). Além disso, quanto maior for o conjunto de dados de treinamento, menor será o conjunto de dados de teste, logo teremos menos dados novos para avaliar o modelo, portanto poderemos confiar menos na precisão alcançada durante o treinamento;

- Quanto menor for o conjunto de dados de treinamento, menos dados meu modelo terá disponível para treinar, menos precisa será sua predição nos dados de treinamento, logo maior será a variância dos dados (mais erros); além disso, quanto menor for o conjunto de dados de treinamento, maior será o conjunto de dados de teste, logo teremos mais dados novos para avaliar o modelo, portanto poderemos confiar mais na precisão alcançada durante o treinamento.

É sempre útil usar a analogia com estudo por questão. Quanto mais questões de concurso sobre um determinado tema você tem disponível para treinar, mais precisas ficarão suas respostas, logo menor será a variância (menos erros).

Gabarito: Correto

---

23. (CESPE / CORREIOS – 2011) Para criar um ranking das universidades brasileiras, um pesquisador dispõe das seguintes variáveis: X_1 = número de professores doutores; X_2 = quantidade de pesquisas publicadas em periódicos nacionais; X_3 = quantidade de pesquisas publicadas em periódicos internacionais; X_4 = área total do campus; X_5 = quantidade de cursos de pós-graduação.

Considerando essas informações e os conceitos de análise multivariada, julgue os itens seguintes.

Se o pesquisador utilizar a técnica de análise de componentes principais, serão geradas 5 componentes.

Comentários:

O PCA gera uma quantidade de componentes principais menor ou igual a quantidade de variáveis. A banca deu o gabarito como correto, mas eu acho que caberia recurso porque esse tipo de redação o torna ambíguo. O aluno fica sem saber o que marcar...

Gabarito: Correto

---

24. (CESPE / EMBRAPA – 2006) Em modelos de classificação, ocorre overfitting quando o número de erros cometidos no grupo de dados usado para treinar (ajustar) o modelo é muito pequeno e o número de erros de generalização é grande.

Comentários:

Perfeito! *Overfitting* é a situação em que há poucos erros no treino, mas há muitos erros no teste (erro de generalização).

Gabarito: Correto

QUESTÕES COMENTADAS – FCC

25. (FCC / TRT-RS – 2022) O Gráfico ROC de uma Análise ROC:

I. é bidimensional, onde o eixos Y e X do gráfico representam as medidas TVP (Taxa de Verdadeiros Positivos) e TFP (Taxa de Falsos Positivos), respectivamente.

II. tem sete regiões importantes que representam: Céu ROC, Inferno ROC, Quase Nunca Positivo, Quase Sempre Positivo, Quase Nunca Negativo, Quase Sempre Negativo e Variáveis Fora da Curva.

III. tem uma linha diagonal que representa Classificadores Aleatórios.

Está correto o que se afirma APENAS em:

a) I.
b) I e II.
c) I e III.
d) II e III.
e) III.

Comentários:

(I) Correto, realmente temos a razão de verdadeiros-positivos pela razão de falsos-positivos; (II) Errado, isso não é relevante para vocês, mas temos quatro regiões importantes: Céu ROC, Inferno ROC, Quase Sempre Positivo e Quase Sempre Negativo; (III) Correto, a linha diagonal realmente representa um classificador aleatório.

Gabarito: Letra C

QUESTÕES COMENTADAS – FGV

26. (FGV / SEFAZ-AM – 2022) Certo conjunto de dados contém 10000 observações, em que cada observação possui 10 variáveis. A análise de componentes principais (PCA) sobre estes dados apontou que a primeira componente principal é dada pelo vetor w. A esse respeito, assinale a afirmativa correta.

a) A variância dos dados é a mesma nas direções dadas por w ou pelas demais componentes principais.

b) A variância dos dados é a menor na direção de w e aumenta na direção das demais componentes principais.

c) A variância ao longo da direção dada por w é igual a 1.

d) A variância ao longo da direção dada por w é menor do que 1.

e) A variância dos dados é máxima na direção dada por w.

Comentários:

Uma breve explicação: um vetor é um conjunto de números que representam uma combinação linear de pontos no espaço multidimensional, que é frequentemente usado para representar os principais componentes de um conjunto de dados. Não entramos nesse nível de detalhe na aula, mas não é relevante no momento. Dito isso, é importante saber que a variância dos dados é máxima na direção dada por w. *Por que, Diego?*

Porque o enunciado da questão disse que a primeira componente principal (1CP) é dada pelo vetor w. Como ela é a primeira, ela representa a variância máxima! Lembrem-se:

Variância(1CP) > Variância (2CP) > ... > Variância (nCP),

em que ***n*** é a quantidade de componentes principais representadas pela Análise de Componentes Principais.

Gabarito: Letra E

---

27. (FGV / TRT-MA – 2022) Com relação aos conceitos de aprendizado de máquina, assinale V para a afirmativa verdadeira e F para a falsa.

I. Os três principais paradigmas de aprendizado de máquina são os de aprendizado supervisionado, não supervisionado e por inteligência profunda.

II. Os algoritmos de classificação e clusterização estão correlacionados com paradigma de aprendizado supervisionado.
III. Os algoritmos de Support Vector Machines e Random Forest são paradigmas do aprendizado de inteligência profunda.

As afirmativas são, respectivamente,

a) V, V e V.
b) V, V e F.
c) V, F e V.
d) F, V e V.
e) F, F e F.

Comentários:

(I) Errado. Trata-se do Aprendizado Supervisionado, Aprendizado Não-Supervisionado e Aprendizado por Reforço; (II) Errado. Clusterização utiliza o paradigma de Aprendizado Não-Supervisionado; (III) Errado. Ambos utilizam Aprendizado Supervisionado.

Gabarito: Letra E

---

28. (FGV / TJDFT – 2022) Após alguns resultados insatisfatórios usando funções de ativação linear em um projeto de rede neural artificial, um cientista de dados resolve tentar outras funções e recebe algumas sugestões de um colega.

Dadas as alternativas abaixo, cada uma representando uma sugestão de função recebida, aquela que apresenta uma função apropriada ao uso como ativação em uma rede neural é:

a) $sen(\sqrt{x^2})$

b) $|x|$

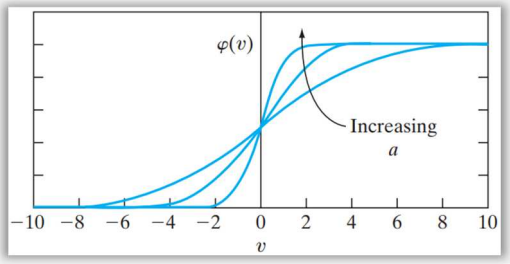
c) $\frac{1}{e^{-x}+1}$

d) $\frac{e^x}{e^{-x}-1}$

e) $\frac{1}{1-\log x}$

Comentários:

FUNÇÃO DE ATIVAÇÃO	FUNÇÕES LOGÍSTICAS
DESCRIÇÃO DA FUNÇÃO	Essa função recebe um valor real qualquer como entrada [-∞,+∞] e retorna um valor de saída entre 0 e 1.
REPRESENTAÇÃO DA FÓRMULA	$f(x) = \frac{1}{1+e^{-x}}$



Aqui infelizmente o candidato teria que se lembrar da fórmula da função logística:

$$f(x) = \frac{1}{1+e^{-x}}$$

Gabarito: Letra C

---

29. (FGV / TJDFT – 2022) Durante o processo de treinamento e validação de uma rede neural, foi observado o fenômeno de underfitting do modelo, necessitando de ajustes ao procedimento. A arquitetura utilizada foi a Multilayer Perceptron (MLP) e o conjunto de dados foi separado em regime de holdout (50%, 30% e 20% para treinamento, validação e teste, respectivamente). Dois fatores que podem ter condicionado o fenômeno observado são:

São elas:

a) iterações insuficientes; amostragem dos dados;
b) excesso de parâmetros; excesso de iterações;
c) insuficiência de parâmetros; excesso de camadas;
d) excesso de iterações; entrada não normalizada;
e) insuficiência de camadas; saída normalizada.

Comentários:

Underfitting ocorre quando um modelo não é complexo o suficiente para capturar com precisão as relações entre os recursos de um conjunto de dados e uma variável de destino. Logo, vamos analisar as alternativas da questão:

(a) Correto. Treinamento com poucas iterações e uma amostragem pouco representativa são causas típicas de underfitting; (b) Errado. Excesso de parâmetros e excesso de iterações são causas típicas de overfitting; (c) Errado. Excesso de camadas no MLP é uma causa típica de overfitting; (d) Errado. Excesso de iterações é uma causa típica de overfitting*; (e) Errado. Saída normalizada é causa típica de overfitting.

* A normalização das entradas é uma das técnicas mais básicas e eficientes no aprendizado de máquina, e isso não fica de fora quando estamos lidando com redes neurais. Quando temos dados com muita variância e que talvez não sejam muito consistentes, é extremamente recomendado a aplicação de uma técnica de normalização para as entradas. Entrada de dados normalizada é aquela em que os dados foram ajustados para que alguns deles não dominem a amostra – é como uma remoção de anomalias.

Gabarito: Letra A

---

30. (FGV / TJDFT – 2022) Considerando a seguinte matriz de confusão obtida de um experimento de classificação:

previsto / real	gato	rato	cachorro
gato	10	2	3
rato	5	14	1
cachorro	1	2	12

Os valores corretos das métricas de precisão e recall (revocação/sensibilidade), para a classe rato, são, respectivamente:

a) 0,62 e 0,67;
b) 0,64 e 0,77;
c) 0,67 e 0,62;
d) 0,78 e 0,7;
e) 0,8 e 0,85.

Comentários:

RECALL	Trata-se da métrica que permite avaliar a capacidade do classificador de detectar com sucesso resultados positivos (também conhecida como sensibilidade/revocação). Responde à pergunta: dentre os valores realmente positivos, quantos o modelo acertou (previu corretamente como positivo)?	$\frac{\mathbf{VP}}{\mathbf{VP + FN}}$
PRECISÃO	Trata-se da métrica que permite mensurar a proporção de previsões positivas corretas sobre a soma de todos os valores positivos. Responde à pergunta: dentre os valores previstos como positivos, quantos o modelo acertou (previu corretamente como positivo)?	$\frac{VP}{VP + FP}$

A precisão depende do Verdadeiros Positivos (VP) e Falsos Positivos (FP). VP ocorre quando o modelo prevê que será um rato e realmente é um rato, logo 14; FP ocorre quando o modelo prevê que será um rato, mas não é um rato, logo 2 (gato) + 2 (cachorro). Note que o algoritmo previu em

duas oportunidades que seria um rato, mas era um gato; e em outras duas oportunidades que seria um rato, mas era um cachorro. Dito isso, temo que:

$$\text{Precisão} = VP/(VP+FP) = 14/(14+(2+2)) = 14/(14+4) = 14/18 = 0{,}7777777 = 0{,}78;$$

O recall depende do Verdadeiros Positivos (VP) e Falsos Negativos (FN). VP ocorre quando o modelo prevê que será um rato e realmente é um rato, logo 14; FN ocorre quando o modelo não prevê que será um rato, mas é um rato, logo 5 (gato) + 1 (cachorro). Note que o algoritmo previu em cinco oportunidades que seria um gato, mas era um rato; e em outra oportunidade que seria um cachorro, mas era um rato. Dito isso, temo que:

$$\text{Recall} = VP/(VP+FN) = 14/(14 + (5+1)) = 14/(14+6) = 14/20 = 0{,}7.$$

Gabarito: Letra D

---

31. (FGV / Prefeitura de Niterói – 2018) No contexto das redes neurais, é comum o uso da função *sigmoid* no papel de função de ativação. Assinale a definição correta dessa função na referida aplicação.

a) $f(z) = \frac{1}{1-e^{z}}$

b) $f(z) = e^{-z}$

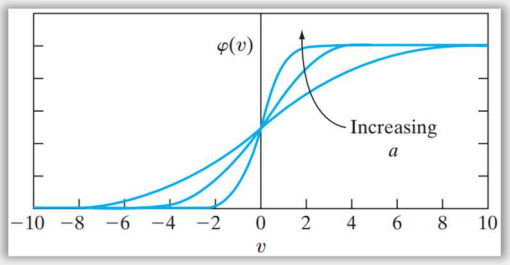
c) $f(z) = \frac{1}{z+e^{-z}}$

d) $f(z) = \frac{z}{1+e^{z}}$

e) $f(z) = \frac{1}{1+e^{-z}}$

Comentários:

FUNÇÃO DE ATIVAÇÃO	FUNÇÕES LOGÍSTICAS
DESCRIÇÃO DA FUNÇÃO	Essa função recebe um valor real qualquer como entrada [-∞,+∞] e retorna um valor de saída entre 0 e 1.
REPRESENTAÇÃO DA FÓRMULA	$f(x) = \frac{1}{1+e^{-x}}$

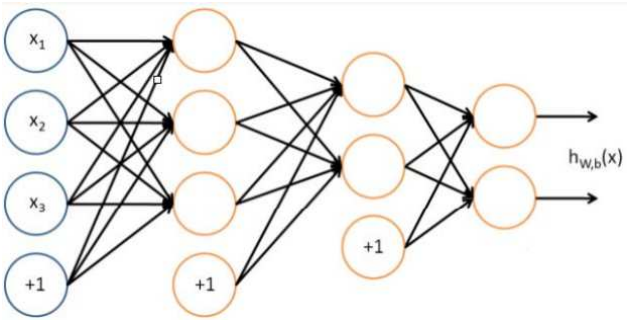


Puro decoreba! Tinha que memorizar a fórmula...

Gabarito: Letra E

---

32. (FGV / Prefeitura de Niterói – 2018) Analise a rede neural exibida a seguir.



Sobre essa rede, analise as afirmativas a seguir.

I. Não possui camadas intermediárias (hidden layers).
II. Admite três sinais de entrada (input units) além do intercept term.
III. É apropriada para aplicações de deep learning.

Está correto o que se afirma em:

a) I, apenas.
b) II, apenas.
c) III, apenas.
d) I e II, apenas.
e) I, II e III.

Comentários:

(I) Errado, possuem duas camadas intermediárias; (II) Correto, admite três sinais de entrada (x_1, x_2, x_3), além do intercepto ou viés (+1); (III) Errado, a questão considerou que – por ter apenas duas camadas ocultas – não é apropriada para aplicações de Deep Learning. Não há nenhuma definição concreta, mas em geral se considera ao menos três camadas intermediárias para ser considerado como aprendizado profundo.

Gabarito: Letra B

---

33. (FGV / Fiocruz – 2010) Assinale a alternativa que indique o problema mais apropriado para aplicação da regressão logística.

a) Para obter o risco relativo de se desenvolver a diabetes tipo 2, em um período de 10 anos, associado com o peso do indivíduo e outros fatores de risco.

b) Para descrever o tamanho esperado de crianças com menos de um ano, de acordo com sua idade em meses.

c) Para predizer o tempo de sobrevivência de pacientes de câncer de pulmão, de acordo com características clínicas do paciente.

d) Para descrever a distribuição de pesos de indivíduos do sexo feminino em uma certa comunidade.

e) Para predizer o número de casos de uma doença em diferentes municípios de acordo com algumas variáveis populacionais e epidemiológicas.

Comentários:

Uma diferença fundamental entre uma regressão linear e uma regressão contínua é que a primeira retorna um valor contínuo e o segundo retorna um valor categórico. Dito isso, vamos analisar:

(a) Correto. A questão não deixou claro o domínio do risco relativo, mas podemos inferir que se trata de uma categoria, tal como baixo ou alto; (b) Errado, tamanho é um valor numérico contínuo, logo se trata de uma regressão linear; (c) Errado, tempo é um valor numérico contínuo, logo se trata de uma regressão linear; (d) Errado, distribuição de pesos é um valor numérico contínuo, logo se trata de uma regressão linear; (e) Errado, número de casos é um valor contínuo, logo se trata de uma regressão linear.

Sendo honesto, eu acredito que caiba recurso! Risco também poderia ser um valor numérico contínuo. Apesar de esse ser o item menos errado, a questão não foi clara!

Gabarito: Letra A

QUESTÕES COMENTADAS – DIVERSAS BANCAS

34. (CESGRANRIO / BB – 2021) Ao tentar resolver um problema de aprendizado de máquina que separava um evento entre duas classes, um desenvolvedor encontrou uma acurácia de exatamente 90%. Analisando a matriz de confusão, o desenvolvedor constatou que os verdadeiros positivos eram 14169, que os verdadeiros negativos eram 15360, os falsos positivos eram 1501, e os falsos negativos eram:

a) 1778
b) 1779
c) 1780
d) 1781
e) 1782

Comentários:

Vamos montar uma matriz de confusão com os dados do enunciado:

		Valor Previsto	
		Negativo	Positivo
Valor Real	Negativo	15360	1501
	Positivo	?	14169

Agora vamos relembrar o que é acurácia:

ACURÁCIA	Trata-se da métrica mais simples que permite mensurar o percentual de acertos, isto é, a quantidade de previsões corretas dentro do total de previsões possíveis. Responde à pergunta: dentre todas as previsões realizadas, quantas o modelo acertou?	$\frac{\mathbf{VP}+\mathbf{VN}}{\mathbf{VP}+\mathbf{FP}+\mathbf{VN}+\mathbf{FN}}$

Logo, temos que:

$$\text{ACURÁCIA} = \frac{\mathbf{VP}+\mathbf{VN}}{\mathbf{VP}+\mathbf{FP}+\mathbf{VN}+\mathbf{FN}} \rightarrow 0{,}9 = \frac{14169+15360}{14169+1505+15360+\text{FN}} \rightarrow 0{,}9 = \frac{29529}{31030+\mathbf{FN}} \rightarrow \text{FN} = 1780$$

Gabarito: Letra C

---

35. (AOCP / MJSP – 2020) Um cientista de dados necessita estimar a precisão preditiva de um classificador medindo essa precisão para uma amostra de dados ainda não utilizada. Quais são as três estratégias principais, comumente usadas para isso, que o cientista de dados pode utilizar?

a) Divisão de dados em conjunto de treino e conjunto de teste, validação cruzada k-fold e validação cruzada N-fold.

b) Erro padrão, divisão de dados em conjunto de treino e conjunto de teste, re-execução de testes.

c) Conjunto de dados permanentes, conjunto de teste e re-execução de testes.

d) Troca de valores mais frequentes, validação cruzada e conjunto de treino e teste.

e) Conjunto de dados ausente, conjunto de treino e conjunto de teste.

Comentários:

As três estratégias poderiam ser a divisão de dados em conjunto de treino e conjunto de teste; validação cruzada K-fold; e validação cruzada N-fold. Não falamos em aula do N-fold, mas ele é basicamente N repetições do k-fold. Os outros itens não fazem qualquer sentido!

Gabarito: Letra A

---

36. **(FUNDATEC / GHC-RS – 2020 – Letra B)** A curva ROC (Receiver Operator Characteristic) é uma forma de expressar graficamente a relação entre a sensibilidade e a especificidade. Para construí-la, deve-se plotar a taxa de verdadeiros positivos (sensibilidade) contra a taxa de verdadeiros negativos (especificidade).

Comentários:

Perfeito! A Curva ROC realmente expressa a relação entre sensibilidade e especificidade por meio da taxa de verdadeiros-positivos e verdadeiros-negativos.

Gabarito: Correto

---

37. **(FUNDATEC / Prefeitura de Porto Mauá/RS – 2019)** A curva ROC (Característica Operatória do Receptor) é uma técnica gráfica para quantificar a acurácia de um teste diagnóstico e ilustrar o contrabalanço entre:

a) A prevalência e a incidência.
b) A sensibilidade e a especificidade.
c) O valor preditivo positivo e a incidência.
d) O valor preditivo negativo e a prevalência.
e) A variável preditora e a variável antecedente.

Comentários:

Conforme vimos em aula, ela ilustra o contrabalanço entre Sensibilidade e Especificidade.

Gabarito: Letra B

---

38. (FUNDEP / CODEMIG – 2018) O objetivo principal da Análise de Componentes Principais é:

a) obtenção de um pequeno número de combinações lineares, de um conjunto de variáveis, que retenham o máximo possível da informação contida nas variáveis originais;

b) calcular, para cada indivíduo da amostra, a probabilidade associada a um determinado fenômeno;

c) testar, por meio de sub-amostras, a distribuição de probabilidades do conjunto de dados estudados;

d) explicar a correlação ou covariância, entre um conjunto de variáveis, em termos de um número limitado de variáveis não-observáveis;

e) identificar se há multicolinearidade nos resíduos, após a estimação de uma curva de regressão.

Comentários:

O objetivo principal da Análise de Componentes Principais – como uma das técnicas de Fatoração de Matrizes – é obtenção de um pequeno número de combinações lineares, de um conjunto de variáveis, que retenham o máximo possível da informação contida nas variáveis originais.

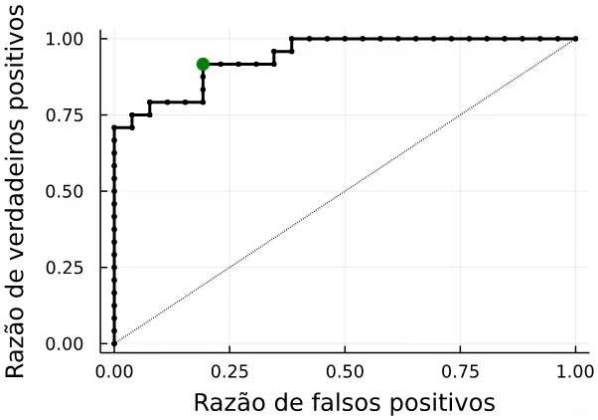
Gabarito: Letra A

---

39. (FUNDATEC / Prefeitura de Vacaria/RS – 2016 – Item III) A Curva ROC é construída levando-se em consideração a taxa de verdadeiro-positivos contra a taxa de falso-positivos. Os valores nos eixos representam medidas de probabilidade, variando de 0 a 1.

Comentários:

Perfeito! Conforme apresenta a imagem seguinte, a curva realmente é baseada na razão de verdadeiros-positivos pela razão de falsos-positivos, que variam de 0 a 1.



Gabarito: Correto

---

40. (ESAF / ANAC – 2016) A redução da dimensionalidade de uma base de dados altamente correlacionados é objetivo da Análise:

a) de Componentes Principais.
b) de Campos de Prioridades.
c) de Componentes de Regressão.
d) Dimensional de Covariância.
e) Interativa de Componentes.

Comentários:

A redução da dimensionalidade é basicamente a redução da quantidade de variáveis de um modelo de alta dimensionalidade. Para tal, existem diversas técnicas como a Análise de Componentes Principais (do inglês, *Principal Component Analysis* – PCA).

Gabarito: Letra A

---

41. (ESAF / Receita Federal – 2014) Em Datamining, redução da dimensionalidade é:

a) A expressão de um conjunto de dados por um conjunto menor de características do que em sua forma original.

b) A redução dos espaços de variação dos dados em relação a seus espaços originais.

c) A supressão de características consideradas de menor prioridade pelo gestor.

d) A expressão de um conjunto de dados por um conjunto de características de dimensionalidade conhecida.

e) A expressão de um conjunto de características por um outro conjunto de características de dimensionalidade invariante em relação à sua forma original.

Comentários:

(a) Correto; (b) Errado, é a redução dos espaços dos dados e, não, dos espaços de variação dos dados; (c) Errado, a supressão se dá em função do processo de modelagem e, não, prioridade do gestor; (d) Errado, é a expressão de um conjunto de dados por um conjunto de características de dimensionalidade menor; (e) Errado, é a expressão de um conjunto de características por um outro conjunto de características de dimensionalidade menor em relação à sua forma original.

Gabarito: Letra A

---

42. (COSEAC / DATAPREV – 2009) O objetivo principal da Análise de Componentes Principais é:

a) obtenção de um pequeno número de combinações lineares, de um conjunto de variáveis, que retenham o máximo possível da informação contida nas variáveis originais;

b) calcular, para cada indivíduo da amostra, a probabilidade associada a um determinado fenômeno;

c) testar, por meio de sub-amostras, a distribuição de probabilidades do conjunto de dados estudados;

d) explicar a correlação ou covariância, entre um conjunto de variáveis, em termos de um número limitado de variáveis não-observáveis;

e) identificar se há multicolinearidade nos resíduos, após a estimação de uma curva de regressão.

Comentários:

O PCA busca reduzir a dimensionalidade de um conjunto de variáveis de modo que elas representem o máximo de informação (variância) possível. Nenhum dos outros itens faz qualquer sentido!

Gabarito: Letra A

LISTA DE QUESTÕES – CESPE

1. **(CESPE / Petrobrás – 2022)** A tabela abaixo apresenta parte de um conjunto de dados referentes a uma variável categórica chamada opinião que possui quatro categorias de resposta: muito satisfeito, satisfeito, insatisfeito e muito insatisfeito.

observação	opinião
1	satisfeito
2	muito satisfeito
3	insatisfeito
4	insatisfeito
5	muito insatisfeito
6	satisfeito

Com base nessas informações, julgue o próximo item.

Para lidar numericamente com os dados categóricos, uma codificação binária proporciona uma conversão de cada categoria de resposta para uma sequência de dígitos binários, em que cada dígito binário representa uma variável numérica que assume valores o ou 1. Na situação em tela, uma possível codificação binária é exemplificada na tabela abaixo.

observação	D1	D2	D3
1	1	0	0
2	0	1	0
3	0	0	1
4	0	0	1
5	0	0	0
6	1	0	0

2. **(CESPE / Petrobrás – 2022)** Em um processo em que se utiliza a ciência de dados, o número de variáveis necessárias para a realização da investigação de um fenômeno é direta e simplesmente igual ao número de variáveis utilizadas para mensurar as respectivas características desejadas; entretanto, é diferente o procedimento para determinar o número de variáveis explicativas, cujos dados estejam em escalas qualitativas.

Considerando esse aspecto dos modelos de regressão, julgue o item a seguir.

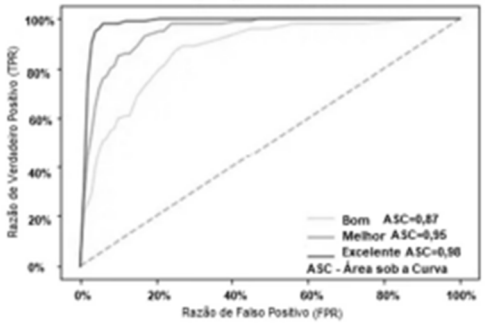
Para evitar um erro de ponderação arbitrária, deve-se recorrer ao artifício de uso de variáveis dummy, o que permitirá a estratificação da amostra da maneira que for definido um determinado critério, evento ou atributo, para então serem inseridas no modelo em análise; isso permitirá o estudo da relação entre o comportamento de determinada variável explicativa qualitativa e o fenômeno em questão, representado pela variável dependente.

3. **(CESPE / Petrobrás – 2022)** As métricas de avaliação de desempenho de um modelo de aprendizado de máquina, que é um componente integrante de qualquer projeto de ciência de

dados, destinam-se a estimar a precisão da generalização de um modelo sobre os dados futuros (não vistos ou fora da amostra). Dentre as métricas mais conhecidas, estão a matriz de confusão, precisão, recall, pontuação, especificidade e a curva de características operacionais do receptor (ROC).

Acerca das características específicas dessas métricas, julgue o próximo item.

As curvas ROC a seguir mostram a taxa de especificidade (verdadeiros positivos) versus a taxa de sensibilidade (falsos positivos) do modelo adotado; a linha tracejada é a linha de base da métrica de avaliação e define uma adivinhação aleatória.



4. **(CESPE / Petrobrás – 2022)** O algoritmo de backpropagation consiste das fases de propagação e de retro propagação: na primeira, as entradas são passadas através da rede e as previsões de saída são obtidas; na segunda, se calcula o termo de correção dos pesos e, por conseguinte, a atualização dos pesos.

5. **(CESPE / ANP – 2022)** A ação de realizar agrupamento hierárquico tem como premissa básica encontrar elementos em um conjunto de dados que impliquem a presença de outros elementos na mesma transação, com um grau de certeza definido pelos índices de fator de suporte e o fator de confiança, que pode ser realizado, por exemplo, por meio do algoritmo a priori.

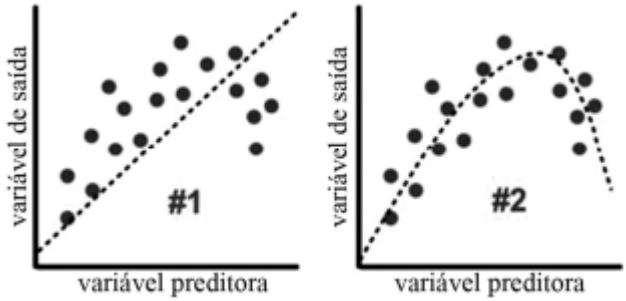
6. **(CESPE / ANP – 2022)** O algoritmo *random forest* é um algoritmo de aprendizado de máquina supervisionado em que se agrupam os resultados de várias árvores de decisão de cada nó para se obter uma conclusão própria e aumentar a precisão do modelo, não sendo o referido algoritmo adequado para grandes conjuntos de dados.

7. **(CESPE / ANP – 2022)** A técnica de redução de dimensionalidade (PCA) permite transformar dados que inicialmente pertencem a um espaço de dimensão n em um espaço de dimensão m, em que m < n, sendo utilizada, por exemplo, para reduzir a dimensionalidade de certo conjunto de dados através do descarte de características não úteis e que ainda permita realizar o reconhecimento de padrões.

8. **(CESPE / ANP – 2022)** As aplicações em inteligência artificial são definidas como uma subárea da área de aprendizagem de máquina (*machine learning*).

9. **(CESPE / ANP – 2022)** Em se tratando de modelos de regressão linear, indica-se a utilização dos seguintes métodos não paramétricos para a estimação dos resultados: Mínimos Quadrados (MQ) e de Support Vector Machines (SVM).

10. **(CESPE / ANP – 2022)** Considerando-se, nos gráficos a seguir, que o resultado #2 corresponda ao melhor desempenho do algoritmo, é correto afirmar que o resultado #1 indica que houve *underfitting*.



Determinado parâmetro β será estimado recursivamente com a ajuda de um método de otimização matemática com base em uma função objetivo $g(\beta)$. Para essa estimação, a base de dados de treinamento consistirá de n observações.

11. **(CESPE / SERPRO – 2021)** Em cada iteração na estimação do parâmetro β, o método do gradiente descendente requer n observações da base de treinamento, ao passo que o método do gradiente descendente estocástico utiliza uma observação selecionada aleatoriamente dessa base de treinamento.

12. **(CESPE / SERPRO – 2021)** Entre as condições ideais relativas à função objetivo g(β) para a aplicação do método do gradiente descendente incluem-se convexidade, continuidade e diferenciabilidade.

13. **(CESPE / SERPRO – 2021)** O método do gradiente descendente é equivalente ao método de Newton-Raphson, no qual o incremento, para a estimação do parâmetro β, depende da primeira e da segunda derivada da função objetivo g(β).

14. **(CESPE / SERPRO – 2021)** O gradiente descendente em lote é um método probabilístico de otimização no qual, para cada iteração, encontram-se L × n observações geradas mediante amostragem (com reposição) da base de dados de treinamento (em que L representa o número de lotes, com L > 1).

15. **(CESPE / SEFAZ-AL - 2021)** A regressão logística é um modelo de regressão no qual a relação entre as variáveis independentes e a variável dependente é representada por uma função degrau, a qual, por sua vez, pode ser representada por uma *spline*.

16. **(CESPE / SEFAZ-CE – 2021)** Cada unidade de uma rede neural artificial possui um valor e um peso, no seu nível mais básico, para indicar sua importância relativa.

17. **(CESPE / SEFAZ-CE – 2021)** Redes neurais do tipo LSTM (long short-term memory) mantêm o nível de precisão independentemente do tamanho do modelo utilizado.

18. **(CESPE / TJ-AM – 2019)** A técnica *machine learning* pode ser utilizada para apoiar um processo de *data mining*.

19. **(CESPE / TCE-MG – 2018)** Em machine learning, a categoria de aprendizagem por reforço identifica as tarefas em que:

a) um software interage com um ambiente dinâmico, como, por exemplo, veículos autônomos.
b) as etiquetas de classificação não sejam fornecidas ao algoritmo, de modo a deixá-lo livre para entender as entradas recebidas.
c) o aprendizado pode ser um objetivo em si mesmo ou um meio para se atingir um fim.
d) o objetivo seja aprender um conjunto de regras generalistas para converter as entradas em saídas predefinidas.
e) são apresentados ao computador exemplos de entradas e saídas desejadas, fornecidas por um orientador.

20. **(CESPE / TRE/PE – 2017 – Item A)** Em uma curva ROC, o ponto que aperfeiçoa a sensibilidade em função da especificidade é aquele que se encontra mais próximo do canto superior esquerdo do gráfico.

21. **(CESPE / INPI – 2013)** Em um banco de dados, foram armazenadas informações relativas a diversas pesquisas realizadas por pesquisadores de institutos renomados. Entre as variáveis constantes desse banco destacam-se: nome, gênero e titulação do pesquisador; valor financiado da pesquisa; instituto ao qual o pesquisador pertence; número de componentes da equipe; e número de artigos publicados pelo pesquisador.

Com base nessas informações, julgue os itens a seguir.

Se, na análise de componentes principais, fossem utilizadas 5 variáveis quantitativas, então, a técnica geraria, no máximo, 3 componentes, se essas correspondessem a, pelo menos, 95% da variância explicada.

22. **(CESPE / MPE-PI – 2012)** Em datamining, o uso de holdout induz o modelo de classificação a partir do conjunto de treinamento, e seu desempenho é avaliado no conjunto de teste. Quanto menor o conjunto de treinamento, maior a variância do modelo; no entanto, se o conjunto de treinamento for grande demais, a precisão estimada calculada a partir do conjunto menor é menos confiável.

23. **(CESPE / CORREIOS – 2011)** Para criar um ranking das universidades brasileiras, um pesquisador dispõe das seguintes variáveis: X1 = número de professores doutores; X2 = quantidade de pesquisas publicadas em periódicos nacionais; X3 = quantidade de pesquisas publicadas em periódicos internacionais; X4 = área total do campus; X5 = quantidade de cursos de pós-graduação.

Considerando essas informações e os conceitos de análise multivariada, julgue os itens seguintes.

Se o pesquisador utilizar a técnica de análise de componentes principais, serão geradas 5 componentes.

24. **(CESPE / EMBRAPA – 2006)** Em modelos de classificação, ocorre overfitting quando o número de erros cometidos no grupo de dados usado para treinar (ajustar) o modelo é muito pequeno e o número de erros de generalização é grande.

25. **(FCC / TRT-RS – 2022)** O Gráfico ROC de uma Análise ROC:

I. é bidimensional, onde o eixos Y e X do gráfico representam as medidas TVP (Taxa de Verdadeiros Positivos) e TFP (Taxa de Falsos Positivos), respectivamente.

II. tem sete regiões importantes que representam: Céu ROC, Inferno ROC, Quase Nunca Positivo, Quase Sempre Positivo, Quase Nunca Negativo, Quase Sempre Negativo e Variáveis Fora da Curva.

III. tem uma linha diagonal que representa Classificadores Aleatórios.

Está correto o que se afirma APENAS em:

a) I.
b) I e II.
c) I e III.
d) II e III.
e) III.

26. **(FGV / SEFAZ-AM – 2022)** Certo conjunto de dados contém 10000 observações, em que cada observação possui 10 variáveis. A análise de componentes principais (PCA) sobre estes dados apontou que a primeira componente principal é dada pelo vetor w. A esse respeito, assinale a afirmativa correta.

a) A variância dos dados é a mesma nas direções dadas por w ou pelas demais componentes principais.

b) A variância dos dados é a menor na direção de w e aumenta na direção das demais componentes principais.

c) A variância ao longo da direção dada por w é igual a 1.

d) A variância ao longo da direção dada por w é menor do que 1.

e) A variância dos dados é máxima na direção dada por w.

27. **(FGV / TRT-MA – 2022)** Com relação aos conceitos de aprendizado de máquina, assinale V para a afirmativa verdadeira e F para a falsa.

I. Os três principais paradigmas de aprendizado de máquina são os de aprendizado supervisionado, não supervisionado e por inteligência profunda.
II. Os algoritmos de classificação e clusterização estão correlacionados com paradigma de aprendizado supervisionado.
III. Os algoritmos de Support Vector Machines e Random Forest são paradigmas do aprendizado de inteligência profunda.

As afirmativas são, respectivamente,

a) V, V e V.
b) V, V e F.
c) V, F e V.
d) F, V e V.
e) F, F e F.

28. **(FGV / TJDFT – 2022)** Após alguns resultados insatisfatórios usando funções de ativação linear em um projeto de rede neural artificial, um cientista de dados resolve tentar outras funções e recebe algumas sugestões de um colega.

Dadas as alternativas abaixo, cada uma representando uma sugestão de função recebida, aquela que apresenta uma função apropriada ao uso como ativação em uma rede neural é:

a) $sen(\sqrt{x^2})$

b) $|x|$

c) $\frac{1}{e^{-x}+1}$

d) $\frac{e^x}{e^{-x}-1}$

e) $\frac{1}{1-\log x}$

29. **(FGV / TJDFT – 2022)** Durante o processo de treinamento e validação de uma rede neural, foi observado o fenômeno de underfitting do modelo, necessitando de ajustes ao procedimento. A arquitetura utilizada foi a Multilayer Perceptron (MLP) e o conjunto de dados foi separado em regime de holdout (50%, 30% e 20% para treinamento, validação e teste, respectivamente). Dois fatores que podem ter condicionado o fenômeno observado são:

São elas:

a) iterações insuficientes; amostragem dos dados;
b) excesso de parâmetros; excesso de iterações;
c) insuficiência de parâmetros; excesso de camadas;
d) excesso de iterações; entrada não normalizada;
e) insuficiência de camadas; saída normalizada.

30. **(FGV / TJDFT – 2022)** Considerando a seguinte matriz de confusão obtida de um experimento de classificação:

previsto / real	**gato**	**rato**	**cachorro**
gato	10	2	3
rato	5	14	1
cachorro	1	2	12

Os valores corretos das métricas de precisão e recall (revocação/sensibilidade), para a classe rato, são, respectivamente:

a) 0,62 e 0,67;
b) 0,64 e 0,77;
c) 0,67 e 0,62;
d) 0,78 e 0,7;
e) 0,8 e 0,85.

31. **(FGV / Prefeitura de Niterói – 2018)** No contexto das redes neurais, é comum o uso da função *sigmoid* no papel de função de ativação. Assinale a definição correta dessa função na referida aplicação.

a) $f(z) = \frac{1}{1 - e^z}$

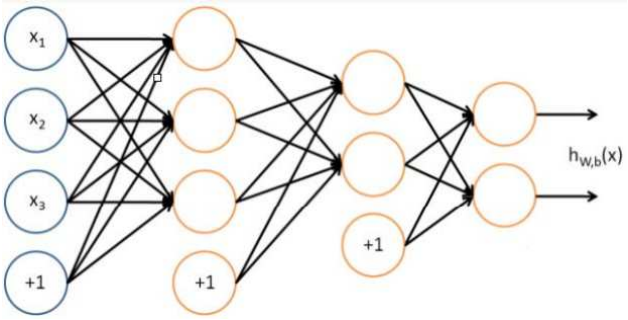
b) $f(z) = e^{-z}$

c) $f(z) = \frac{1}{z + e^{-z}}$

d) $f(z) = \frac{z}{1 + e^z}$

e) $f(z) = \frac{1}{1 + e^{-z}}$

32. (FGV / Prefeitura de Niterói – 2018) Analise a rede neural exibida a seguir.



Sobre essa rede, analise as afirmativas a seguir.

I. Não possui camadas intermediárias (hidden layers).
II. Admite três sinais de entrada (input units) além do intercept term.
III. É apropriada para aplicações de deep learning.

Está correto o que se afirma em:

a) I, apenas.
b) II, apenas.
c) III, apenas.
d) I e II, apenas.
e) I, II e III.

33. (FGV / Fiocruz – 2010) Assinale a alternativa que indique o problema mais apropriado para aplicação da regressão logística.

a) Para obter o risco relativo de se desenvolver a diabetes tipo 2, em um período de 10 anos, associado com o peso do indivíduo e outros fatores de risco.

b) Para descrever o tamanho esperado de crianças com menos de um ano, de acordo com sua idade em meses.

c) Para predizer o tempo de sobrevivência de pacientes de câncer de pulmão, de acordo com características clínicas do paciente.

d) Para descrever a distribuição de pesos de indivíduos do sexo feminino em uma certa comunidade.

e) Para predizer o número de casos de uma doença em diferentes municípios de acordo com algumas variáveis populacionais e epidemiológicas.

34. **(CESGRANRIO / BB – 2021)** Ao tentar resolver um problema de aprendizado de máquina que separava um evento entre duas classes, um desenvolvedor encontrou uma acurácia de exatamente 90%. Analisando a matriz de confusão, o desenvolvedor constatou que os verdadeiros positivos eram 14169, que os verdadeiros negativos eram 15360, os falsos positivos eram 1501, e os falsos negativos eram:

a) 1778
b) 1779
c) 1780
d) 1781
e) 1782

35. **(AOCP / MJSP – 2020)** Um cientista de dados necessita estimar a precisão preditiva de um classificador medindo essa precisão para uma amostra de dados ainda não utilizada. Quais são as três estratégias principais, comumente usadas para isso, que o cientista de dados pode utilizar?

a) Divisão de dados em conjunto de treino e conjunto de teste, validação cruzada k-fold e validação cruzada N-fold.

b) Erro padrão, divisão de dados em conjunto de treino e conjunto de teste, re-execução de testes.

c) Conjunto de dados permanentes, conjunto de teste e re-execução de testes.

d) Troca de valores mais frequentes, validação cruzada e conjunto de treino e teste.

e) Conjunto de dados ausente, conjunto de treino e conjunto de teste.

36. **(FUNDATEC / GHC-RS – 2020 – Letra B)** A curva ROC (Receiver Operator Characteristic) é uma forma de expressar graficamente a relação entre a sensibilidade e a especificidade. Para

construí-la, deve-se plotar a taxa de verdadeiros positivos (sensibilidade) contra a taxa de verdadeiros negativos (especificidade).

37. **(FUNDATEC / Prefeitura de Porto Mauá/RS – 2019)** A curva ROC (Característica Operatória do Receptor) é uma técnica gráfica para quantificar a acurácia de um teste diagnóstico e ilustrar o contrabalanço entre:

a) A prevalência e a incidência.
b) A sensibilidade e a especificidade.
c) O valor preditivo positivo e a incidência.
d) O valor preditivo negativo e a prevalência.
e) A variável preditora e a variável antecedente.

38. **(FUNDEP / CODEMIG – 2018)** O objetivo principal da Análise de Componentes Principais é:

a) obtenção de um pequeno número de combinações lineares, de um conjunto de variáveis, que retenham o máximo possível da informação contida nas variáveis originais;

b) calcular, para cada indivíduo da amostra, a probabilidade associada a um determinado fenômeno;

c) testar, por meio de sub-amostras, a distribuição de probabilidades do conjunto de dados estudados;

d) explicar a correlação ou covariância, entre um conjunto de variáveis, em termos de um número limitado de variáveis não-observáveis;

e) identificar se há multicolinearidade nos resíduos, após a estimação de uma curva de regressão.

39. **(FUNDATEC / Prefeitura de Vacaria/RS – 2016 – Item III)** A Curva ROC é construída levando-se em consideração a taxa de verdadeiro-positivos contra a taxa de falso-positivos. Os valores nos eixos representam medidas de probabilidade, variando de 0 a 1.

40. **(ESAF / ANAC – 2016)** A redução da dimensionalidade de uma base de dados altamente correlacionados é objetivo da Análise:

a) de Componentes Principais.
b) de Campos de Prioridades.
c) de Componentes de Regressão.
d) Dimensional de Covariância.
e) Interativa de Componentes.

41. **(ESAF / Receita Federal – 2014)** Em Datamining, redução da dimensionalidade é:

a) A expressão de um conjunto de dados por um conjunto menor de características do que em sua forma original.

b) A redução dos espaços de variação dos dados em relação a seus espaços originais.

c) A supressão de características consideradas de menor prioridade pelo gestor.

d) A expressão de um conjunto de dados por um conjunto de características de dimensionalidade conhecida.

e) A expressão de um conjunto de características por um outro conjunto de características de dimensionalidade invariante em relação à sua forma original.

42. (COSEAC / DATAPREV – 2009) O objetivo principal da Análise de Componentes Principais é:

a) obtenção de um pequeno número de combinações lineares, de um conjunto de variáveis, que retenham o máximo possível da informação contida nas variáveis originais;

b) calcular, para cada indivíduo da amostra, a probabilidade associada a um determinado fenômeno;

c) testar, por meio de sub-amostras, a distribuição de probabilidades do conjunto de dados estudados;

d) explicar a correlação ou covariância, entre um conjunto de variáveis, em termos de um número limitado de variáveis não-observáveis;

e) identificar se há multicolinearidade nos resíduos, após a estimação de uma curva de regressão.

GABARITO

1. CORRETO
2. CORRETO
3. ERRADO
4. CORRETO
5. ERRADO
6. ERRADO
7. CORRETO
8. ERRADO
9. ERRADO
10. CORRETO
11. CORRETO
12. CORRETO
13. ERRADO
14. ERRADO
15. ERRADO
16. CORRETO
17. ERRADO
18. CORRETO
19. LETRA A
20. CORRETO
21. ERRADO
22. CORRETO
23. CORRETO
24. CORRETO
25. LETRA C
26. LETRA E
27. LETRA E
28. LETRA C
29. LETRA A
30. LETRA D
31. LETRA E
32. LETRA B
33. LETRA A
34. LETRA C
35. LETRA A
36. CORRETO
37. LETRA B
38. LETRA A
39. CORRETO
40. LETRA A
41. LETRA A
42. LETRA A

ESSA LEI TODO MUNDO CONHECE:
PIRATARIA É CRIME.

Mas é sempre bom revisar o porquê e como você pode ser prejudicado com essa prática.

1. 

Professor investe seu tempo para elaborar os cursos e o site os coloca à venda.

2. 

Pirata divulga ilicitamente (grupos de rateio), utilizando-se do anonimato, nomes falsos ou laranjas (geralmente o pirata se anuncia como formador de "grupos solidários" de rateio que não visam lucro).

3. 

Pirata cria alunos fake praticando falsidade ideológica, comprando cursos do site em nome de pessoas aleatórias (usando nome, CPF, endereço e telefone de terceiros sem autorização).

4. 

Pirata compra, muitas vezes, clonando cartões de crédito (por vezes o sistema anti-fraude não consegue identificar o golpe a tempo).

5. 

Pirata fere os Termos de Uso, adultera as aulas e retira a identificação dos arquivos PDF (justamente porque a atividade é ilegal e ele não quer que seus fakes sejam identificados).

6. 

Pirata revende as aulas protegidas por direitos autorais, praticando concorrência desleal e em flagrante desrespeito à Lei de Direitos Autorais (Lei 9.610/98).

7. 

Concurseiro(a) desinformado participa de rateio, achando que nada disso está acontecendo e esperando se tornar servidor público para exigir o cumprimento das leis.

8. 

O professor que elaborou o curso não ganha nada, o site não recebe nada, e a pessoa que praticou todos os ilícitos anteriores (pirata) fica com o lucro.

Deixando de lado esse mar de sujeira, aproveitamos para agradecer a todos que adquirem os cursos honestamente e permitem que o site continue existindo.