02

Filtro para recomendação

Transcrição

Como eu falei, estamos utilizando uma variação do algoritmo k-NN, cuja ideia é fazer uma classificação ou uma regressão baseada nos vizinhos mais próximos. Dados os vizinhos, por exemplo vizinho = 1, pegamos o(s) valor(es) desse usuário mais próximo. Mas se pegarmos os 20 vizinhos mais próximos, de alguma maneira tiraremos uma média das informações que temos sobre eles.

No nosso caso, as informações que temos são as notas que os usuários/vizinhos deram para cada um dos filmes, e a partir dessas notas tiramos uma média. A ideia agora é testarmos o algoritmo que criamos.

Para isso, vamos criar um novo usuário. Para que nosso algoritmo funcione, as notas desse usuário precisarão estar dentro do dataset notas. Portanto, peguei uma amostra dos filmes e atribuí algumas notas aleatórias a eles. Vamos dar uma olhada nesses filmes fazendo filmes.loc[[]], passando a lista de Ids desses filmes entre dois colchetes.

Temos na lista alguns filmes como Deadpool, Sociedade dos Poetas Mortos, Planeta dos Macacos, A Outra História Americana, A Vida é Bela, Toy Story, Sabrina, Jumanji e dois episódios de Star Wars.

Agora, vamos definir uma função novo_usuario(dados). Esses dados precisam ser no formato de uma lista de listas contendo [filme, nota]. Passaremos então para a função novo_usuario() a lista dos filmes e suas respectivas notas, respeitando a formatação adequada.

No corpo da função, faremos pd.DataFrame(dados, columns=["filmeId", "nota"]) para criarmos um dataframe de duas colunas. Repare que não estamos colocando o momento em que a nota foi dada, afinal não estamos utilizando essa coluna para nenhum tipo de análise. Atribuiremos esse dataframe a uma variável notas_do_usuario_novo.

Gostaríamos de criar uma coluna nova que seja o Id desse usuário. Vamos conferir o maior usuário do dataset notas com notas['usuarioId'].max(), obtendo o resultado 8620. Utilizaremos então notas['usuarioId'].max()+1 para criarmos o Id do novo_usuario.

Faremos então notas_do_usuario_novo['usuarioId'] = novo_usuario. Então, retornaremos a concatenação do novo dataset com o dataset notas, escrevendo pd.cnocat([notas, notas_do_usuario_novo]). Feito isso, poderemos chamar novo_usuario() passando os filmes e notas que criamos anteriormente, obtendo como retorno o dataset notas com os dados antigos e o usuário que acabamos de adicionar. Vamos sobrescrever o dataset notas com esse novo conjunto, e chamaremos notas.tail() para conferirmos os últimos cinco elementos e verificarmos se tudo funcionou.

Agora que temos o usuário 8621, vamos chamar sugere_para(8621).head() para recebermos as primeiras cinco recomendações. Dentre os resultados, um dos filmes eu realmente gostei (Moonrise Kingdom), mas os outros quatro eu nunca ouvi falar. Temos aqui um detalhe que já comentamos nesse curso: estamos deixando passar nas nossas recomendações alguns filmes que possuem poucos votos e que, às vezes, podem ser muito de nicho. Repare que a maioria dos filmes na lista realmente têm pouquíssimos votos.

Poderíamos ter um algoritmo que funcionasse somente para descobertas de novos filmes, e nesse recomendar filmes com poucas avaliações seria interessante. Mas não é o caso no nosso sistema, e, portanto, vamos utilizar somente as notas de filmes com mais de 50 avaliações para evitarmos essa situação.

Dado que estamos utilizando o dataset notas em diversos momentos de operação do nosso algoritmo, a maneira mais simples de fazermos essa alteração é alterando o próprio dataset. Em um momento anterior do curso, nós pegamos os

filmes_com_mais_de_50_votos, que inclusive já possui os próprios filmes como índice.

Para isso, faremos notas.set_index("filmeId").loc[filmes_com_mais_de_50_votos].index. Nesse processo, estamos setando o índice de notas como sendo o filmeId, e então localizando somente os filmes_com_mais_de_50_votos por meio do índice deles. Então, sobrescreveremos a variável notas com esse novo dataset. Nesse ponto, devemos tomar cuidado ao executar trechos de código que criamos no passado, pois os dados mudaram e os resultados podem ser outros!

Ainda teremos que tomar outro cuidado: nós forçamos que o filmeId fosse utilizado como índice, e as operações com a coluna filmeId não funcionarão mais. Portanto, vamos sobrescrever novamente a variável notas com notas.rest_index(), resetando o índice.

Agora, se executarmos novamente nosso algoritmo de sugestões, não teremos mais aqueles filmes que declaramos como filmes de nicho (ou que não eram populares o suficiente para a nossa recomendação).

No próximo vídeo, refinaremos ainda mais um ponto do nosso algoritmo!