02

Calculando a distância entre um e todos os usuários

Transcrição

Já conseguimos calcular a distância entre dois usuários do nosso dataset, algo que começamos a fazer manualmente, entendendo as distâncias em uma ou duas dimensões, e depois implementando bibliotecas que fazem isso automaticamente.

Agora, queremos que o sistema identifique, dentre todos os usuários, quais são mais parecidos de um específico. Anteriormente, quando trabalhamos com João, Maria e Joaquina, fizemos exatamente isso, comparando as distâncias entre eles. Agora, se escolhermos o usuario1 como base, teremos que compará-lo com todos os outros usuários que existem no nosso banco de dados.

Com len(notas[usuarioId].unique()), verificaremos quantos usuários existem no dataset, que são 1240. Portanto, teremos que comparar o usuario1 com 1239 outros usuários. Armazenaremos essa informação em uma variável quantidade_de_usuarios e imprimiremos com print("Temos %d usuarios" % quantidade_de_usuarios).

Para calcularmos as distâncias entre todos eles, precisaremos criar um for. Repare que, quando calculamos a distância entre dois usuários, somente nos é devolvido um número - por exemplo, 3.04. Como queremos que esse retorno seja mais claro, vamos armazenar a distancia_de_vetores() em uma variável distancia e retornaremos uma lista com [usuario+id1, usuarioid2, distancia].

Começaremos criando um for para cada usuario_id dentro de notas['usuarioId'].unique(). Para testarmos essa iteração, vamos imprimir usuario_Id. Essa é uma operação simples que nos ajuda a ter certeza de que o código está saindo da forma que gostaríamos, o que é uma boa prática.

Vamos criar uma variável voce_id representando o usuário 1. Queremos, então, ao invés de imprimir os usuários, calcular a distancia_de_usuarios() entre voce_id e usuario_id. O retorno dessa função será armazenado em uma variável informacoes. Por enquanto, vamos somente imprimir os resultados.

Novamente teremos 1240 números, mas dessa vez contendo o voce_id, o usuario_id que está sendo comparado e a distância. Vamos criar uma variável distancias, que começará como um conjunto vazio, e com distancias.append(informacoes) armazenaremos toda a lista retornada na nossa iteração. Podemos, então, imprimir os cinco primeiro valores com distancias[:5] (já que estamos trabalhando com uma lista). Com isso, veremos que nosso código parece estar funcionando corretamente.

Vamos refinar o nosso código, refatorando-o e colocando-o em uma função. Primeiro, vamos definir essa operação em uma função distancia_de_todos(voce_id), e definiremos distancias como nosso retorno. Então, poderemos fazer distancia_de_todos(1). Com isso, calcularemos as distâncias entre todos os usuários e o usuario1.

Essa é uma maneira tradicional (também chamada de "imperativa") de fazermos esse for. Uma delas, que você já deve ter visto em outros cursos da Alura, é colocando o for entre colchetes para criar uma lista. À esquerda, definiremos o que iremos fazer com essa lista, que no nosso caso é chamar a função distancia_de_usuarios(voce_id, usuario_id).

Fora da lista, definiremos uma variável todos_os_usuarios recebendo notas['usuarioId'].unique(). Faremos, então, for usuario_id in todos_os_usuarios. Dessa forma, para cada usuario_id dentro de todos_os_usuarios, chamaremos a função distancia_de_usuarios e criaremos uma lista. isso será armazenado em uma variável distancias, que continuará sendo o retorno da função.

Em algumas situações, essa forma de criarmos o for é mais vantajosa. Isso é algo que discutimos em outros cursos de linguagem Python. Se testarmos a função, teremos exatamente os mesmos valoress, o que significa que ela está funcionando corretamente.

Para melhorarmos a compreensão dos nossos dados, ao invés de simplesmente retornarmos as distancias, vamos criar um dataframe com distancias = pd.DataFrame(distancias, columns = ["voce", "outra_pessoa", "distancia"]. Agora que estamos trabalhando com um dataframe do Pandas, podemos chamar distancia_de_todos(1).head() para exibir os cinco primeiros elementos desse dataframe.

Com isso, fomos capazes de criar uma função que calcula a distância entre um usuário e todos os outros usuários do nosso dataframe. Nosso próximo passo será procurar os usuários mais próximos.