05

Total de votos, nota média e possíveis dificuldades de heurísticas simples

Transcrição

Vamos dar uma olhada mais a fundo nas recomendações que fizemos com essa heurística? Ao invés de recomendarmos 5 filmes, vamos recomendar 10. Nessa lista, realmente temos filmes que são "famosos", como Um Sonho de Liberdade e Pulp Fiction.

Agora, analisaremos as notas desses filmes. Com notas.groupby("filmeId").mean(), vamos agrupar as notas pelo filmeId e tirar as médias delas. Teremos, então, que o filmeId 1 teve a nota média 3.88, o filmeId 2 teve a nota 3.14, e assim por diante. Não estamos interessados nas médias de usuarioId ou de momento, portanto filtraremos apenas as notas médias com notas.groupby("filmeId").mean()["nota"].

Atribuiremos essa série a uma variável notas_medias. Como utilizamos o groupby() do Pandas, elas já estão indexadas pelo índice. Portanto, podemos colocá-las no dataframe dos filmes com `filmes["nota_media"] = notas _medias` . Em seguida, pediremos novamente os 10 filmes com o maior total de votos.

O primeiro filme da lista tem nota média 4.41, o segundo 4.08, o terceiro 4.17, e assim por diante. Repare que Jurassic Park tem uma nota média 3.65, ou seja, ele é muito popular em quantidade de votos, mas não tem uma nota tão alta. Inclusive, o filme seguinte, A Lista de Schindler, o próximo filme na lista, foi visto por menos pessoas, mas foi melhor avaliado.

Ou seja, ordenar os filmes pelos mais populares em votos não necessariamente irá implicar naqueles que as pessoas mais gostaram. Portanto, temos um problema na nossa heurística, que é a própria definição de popular. Seriam os filmes mais vistos? Mais próximos da nota 5? Só poderíamos criar a nossa heurística a partir de uma definição formal de popular, que inicialmente assumimos como sendo aqueles filmes com a maior quantidade de votos, mas que às vezes pode não fazer sentido.

Uma segunda heurística/abordagem que podemos utilizar para nos trazer resultados satisfatórios é ordenar os filmes pela nota média. Afinal, se muita gente gostou do filme, a nota média é alta; se quase ninguém gostou, a nota média é baixa. Com filmes.sort_values("nota_media", ascending = False).head(10), faremos essa ordenação e mostraremos os 10 primeiros da lista.

Porém, dos 10 resultados, a maioria é bem desconhecido. Eu, por exemplo, só assisti ao último. Repare que todos eles têm média 5... mas também têm um total de votos 1. Complicado, não? Afinal, não dá para definirmos os melhores filmes como sendo aqueles que têm a maior nota se eles só possuem uma avaliação cada um.

Portanto, o problema de recomendação é mais complexo do que uma simples ordenação, mesmo que tenhamos dados coletados de maneira colaborativa.

Uma alternativa seria, por exemplo, criar mais variações de recomendação, como um sistema que recomenda os filmes mais vistos que o usuário ainda não viu, e outro que recomenda os filmes com maior média de notas. Porém, essa heurística que criamos possui diversos problemas. Por exemplo, imagine o caso dos filmes de nicho: são poucas pessoas avaliando, e quem avalia já tem uma disposição a gostar desses filmes. Portanto, a nota ficará alta, e acabaremos recomendo filmes de nichos muito específicos para o nosso usuário.

Para tentarmos resolver esse problema, criaremos uma query() que filtra os nossos filmando, excluindo aqueles cujo total de votos seja menor que 10. Em seguida, executaremos a mesma ordenação que fizemos anteriormente, mostrando so 10 de maior média.

Dessa vez, só temos filmes cuja nota média não é 5 (afinal, não foram avaliados somente por uma pessoa). Um dos filmes no nosso top de popularidade, Um Sonho de Liberdade, aparece nessa lista. Porém, também aparecem alguns filmes mais de nicho, como alguns que têm apenas 10 votos. Portanto, vamos filtrar novamente, dessa vez excluindo os filmes com menos de 50 votos, e atribuiremos o retorno a uma variável filmes_com_mais_de_50_votos.

Dessa vez, temos mais filmes famosos e interessantes nessa lista. Novamente, aqueles que não são tão conhecidos têm um número de votos razoavelmente baixo, o que é um problema que teríamos que lidar. Quando estamos atacando uma heurística, temos que, de alguma maneira, balancear o nosso filtro para excluir elementos que são muito específicos de modo a criar um sistema de recomendação para alguém sobre quem não sabemos nada a respeito.

Essa segunda heurística, que chamaremos de "nota média e filtrando votos", e a primeira, que chamaremos de "heurística de total de votos", trazem resultados interessantes. A de total de votos realmente está ligada à popularidade, ainda que seja possível que alguma comunidade na internet faça uma brincadeira em que todo mundo vote em algum filme para que ele apareça nos mais votados, deturpando os resultados. Porém, existem artigos científicos dizendo que esse tipo de recomendação acaba sendo razoavelmente bom. Ainda assim, percebemos que também é interessante levar as notas em consideração.

No fim, talvez você queira colocar o sistema de recomendações no ar e medir o resultado com seus clientes, verificando se ele está cumprindo o objeto que os usuários da sua plataforma estão buscando. Se as pessoas estiverem satisfeitas, você mantém; se não estiverem, você troca.

Implementamos essas duas heurísticas sem sabermos nada dos nossos usuários. Porém, muitas vezes temos informações sobre eles. Eu sei, por exemplo, os filmes que eu assisti. Se eu te passo essa informação, você poderá fazer recomendações um pouco melhores. Afinal, se eu disser que não gostei de Star Wars, mas gostei de Deadpool, você poderá me recomendar um filme mais relacionado a super-heróis e menos relacionado ao espaço.

Ou seja, com mais informações sobre o usuário é possível refinar as recomendações, e é isso que faremos daqui a pouco.