

## Ausência de dados e taxa de erro

### Transcrição

Já vimos que se pedirmos um número acima das possibilidades de recomendações, o algoritmo retorna apenas a quantidade possível, no caso anterior o usuário 2 poderia receber somente 3 recomendações, porque os outros itens já haviam sido avaliados.

Mas e se pedíssemos para recomendar um item para o usuário 6? Teremos um erro, isso ocorre porque o Mahout identifica que o usuário 6 não existe no nosso conjunto de dados. Ter poucos dados é bem diferente de dados que não existem. Para mais testes, no mesmo arquivo `cvs`, em qualquer posição, vamos inserir alguns dados do usuário 6. Primeiro apenas uma linha:

```
6,18,5.0
```

Se pedirmos para o recomendador fazer recomendações para o usuário 6 que agora está presente, ele não conseguirá porque não há dados suficientes para analisar uma possível recomendação. Vamos inserir mais alguns dados sobre o usuário 6.

```
6,17,1.0
```

Com as duas informações o algoritmo já consegue recomendar algo. Neste caso, pedimos 4 recomendações e como resultado obtemos:

```
RecommendedItem[item:12, value:4.75]  
RecommendedItem[item:16, value:4.6666665]  
RecommendedItem[item:13, value:4.5]  
RecommendedItem[item:15, value:4.1666665]
```

Considerando estes cenários, observamos que precisamos ter cuidado nestes dois pontos: (1) não pedir recomendações para usuários que não existem (*ou mesmo itens*) e (2) cuidado com a ausência de dados para um determinado usuário. Nestes dois casos não será possível fazer recomendações, veremos sobre isso mais adiante.

Sobre o recomendador em si, o que você acha dele até o momento? Acha que está bom? Veja que o primeiro item recomendado para o usuário 6 teve nota 4.75, mas isso é bom? Lembra das recomendações que pedimos para o usuário 2? Elas eram sobre os itens restantes para o usuário 2 não ter mais recomendações. **Como a gente testa se a recomendação é adequada?**

Faremos um pequeno teste, primeiro, removemos o usuário 6, porque o inventamos. Logo em seguida, observe que em uma das linhas do `csv`, temos que o usuário 2 avaliou o item 15 com nota 5.

```
2,15,5.0
```

Será que ao remover essa linha e executar o recomendador novamente, ele recomendará o item 15 com nota 5 ou próxima disso? Vamos pedir ao recomendador então para recomendar 4 itens para o usuário 2. Ao fazermos isso, o

resultado é:

```
RecommendedItem[item:12, value:4.8328104]
RecommendedItem[item:13, value:4.6656213]
RecommendedItem[item:14, value:4.331242]
RecommendedItem[item:15, value:3.7972403]
```

De 3.79 para 5 existe uma diferença considerável, foi um chute ruim. O que chama a atenção é: dependendo de qual linha é removida, nós temos um chute mais próximo ou mais distante da realidade na recomendação.

Para mais testes, vamos remover 10% da amostra de dados de três usuários, serão eles o 1, 3 e 4 que avaliaram os itens 15, 13 e 17 todos com nota 4.

```
1,15,4.0
3,13,4.0
4,17,4.0
```

No código, vamos replicar as linhas de recomendação para avaliar os resultados.

```
System.out.println("usuário 1");
List<RecommendedItem> recommendations = recommender.recommend(1, 4);
for (RecommendedItem recommendation : recommendations) {
    System.out.println(recommendation);
}
System.out.println("usuário 3");
recommendations = recommender.recommend(3, 4);
for (RecommendedItem recommendation : recommendations) {
    System.out.println(recommendation);
}
System.out.println("usuário 4");
recommendations = recommender.recommend(4, 4);
for (RecommendedItem recommendation : recommendations) {
    System.out.println(recommendation);
}
```

Colocamos algumas mensagens para indicar quais recomendações são para quais usuário.

Os resultados são:

```
Usuário 1
RecommendedItem[item:15, value:4.5403237]

Usuário 3
RecommendedItem[item:10, value:1.0]

Usuário 4
```

Avaliando bem, vemos que para o usuário 1, a diferença na avaliação foi de aproximadamente 0.54. Visto que a nota real era 4.0 e o recomendador chutou nota 4.54. Para o usuário 3, o item recomendado foi outro totalmente diferente do que aquele que removemos da amostra, para o usuário 4, não houve nenhuma recomendação.

Para estes erros extremos, vamos supor que a nota a ser chutada deveria ser algo mediano como 2.5 e, assim, podemos considerar que o erro foi em torno de 1.5

Estamos considerando as notas de 0 a 5.

Tirando a média dos erros temos:

$$0.54 + 1.5 + 1.5 / 3 = 1.18$$

Lembre-se que em casos de números negativos e positivos nesses erros, o comum é considerarmos para a média apenas números absolutos, ou seja, sem sinal.

Note que, 1.8 é a taxa de erro na recomendação e esse é um valor interessante de se observar. O que ele indica? Indica que, dentro de uma possível recomendação, onde o recomendador indique um item, a sua avaliação estará com uma variância de 1.18 para mais ou para menos.

Ou seja, caso o recomendador indique que você avaliaria um item com nota 3, a taxa de erro mostrará que a avaliação real pode se alterar entre 1.82 e 4.18, o que no caso das notas em dados atuais significa um intervalo muito grande para uma taxa de erro. Inclusive, maior que a média!

Claro que isso varia de cenário para cenário. Caso as notas possíveis fossem de 0 a 10, uma variação de 1.18 pode não ser tão significativa assim, de 0 a 100 menos ainda. Mas, se essa taxa de erro medisse a probabilidade de um foguete explodir em seu pouso o valor de 1.18 talvez fosse uma variação com peso maior.

Nosso próximo passo é fazer esse teste em código, remover uma amostra dos dados, realizar os testes e verificar qual é a taxa de erro nas recomendações. Vamos lá!