

## Exemplo básico

### Transcrição

Agora que sabemos como funcionam os algoritmos de recomendação, vamos utilizar um exemplo básico com o Mahout e um pouco de código Java para depois utilizarmos os dados reais em um exemplo mais complexo.

O exemplo que vamos trabalhar agora é para saber qual usuário deu qual nota para qual curso. Uma ilustração disso seria: O Guilherme (*usuário*) deu nota 1.0 para o Curso X. Como utilizaremos apenas números para descobrir tais informações, teremos a seguinte abstração: o usuário 1, deu nota 2.0 para o item 3.

O formato em que esses dados serão apresentados ao algoritmo é em CSV (*comma-separated values*), ou seja, valores separados por vírgula. Sendo assim, teremos:

```
1,10,1.0
1,11,2.0
1,12,5.0
1,13,5.0
.
.
.
4,15,3.0
4,16,1.0
4,17,4.0
4,18,1.0
```

No primeiro dígito temos o id do usuário, no segundo o item avaliado e por último a nota. Note que estamos limitando o número de itens e de usuários. Para esse exemplo, temos 9 itens que vão do 10 até o 18 e apenas 4 usuários. No código Java nós montamos o modelo de análise utilizando o seguinte trecho de código:

```
DataModel model = new FileDataModel(new File(dados.csv));
```

Neste trecho estamos basicamente montando o modelo de análise, algo semelhante a tabela que vimos anteriormente. Com 9 itens e 4 usuários chegamos a uma matriz de 36 espaços. Algo possível de se trabalhar em memória, mas, caso tivéssemos 300 usuários com 10.000 itens isso seria inviável! Por isso, teríamos que utilizar outra estrutura de dados, algo já resolvido pela biblioteca.

A próxima etapa é escolher o algoritmo de similaridade que será utilizado. Neste caso utilizamos o *Person Correlation Similarity* que é o que já estudamos na aula anterior:

```
DataModel model = new FileDataModel(new File(dados.csv));

UserSimilarity similarity = new PersonCorrelationSimilarity(model)
```

Precisamos definir também o que será considerado próximo, seria o valor 1, 10, 15? O que será a proximidade? Neste caso, utilizaremos a classe *ThresholdUserNeighborhood* e definiremos que o espaço para ser considerado próximo será de 0.1:

```
DataModel model = new FileDataModel(new File(dados.csv));

UserSimilarity similarity = new PersonCorrelationSimilarity(model)

UserNeighborhood neighborhood = new ThresholdUserNeighborhood(0.1, similarity, model);
```

Por último precisamos criar nosso recomendador de fato. Este será, como vimos, baseado nos usuários. Dessa forma teremos:

```
DataModel model = new FileDataModel(new File(dados.csv));

UserSimilarity similarity = new PersonCorrelationSimilarity(model)

UserNeighborhood neighborhood = new ThresholdUserNeighborhood(0.1, similarity, model);

UserBasedRecommender recommender = new GenericUserBasedRecommender(model, neighborhood, similarity);
```

Agora sim podemos pedir para o recomendador fazer uma recomendação de fato. Neste exemplo, pediremos para que sejam recomendados até 3 cursos para o usuário 2 e os imprimiremos no console.

```
List recommendations = recommender.recommend(2, 3);
for ( RecommendedItem recommendation : recommendations ){
    System.out.println(recommendation);
}
```

Lembre-se que são até três recomendações, no exemplo anterior o Guilherme só podia receber uma recomendação, o João até 3 e assim por diante.