Kmeans recommendation

- Recomendação usando o algoritmo Kmeans.

User-based

- A premissa é a mesma de todos os sistemas de recomendação, quer se recomendar um item i para um usuário u que faça sentido para ele.
- Em casos em que só se tem a disposição as avaliações de usuários para determinados itens, o melhor a se fazer é buscar outros usuários similares a u, e a partir daí fazer recomendações.
- A utilização do Kmeans facilitará a busca pelos usuários similares, sendo consideravelmente mais otimizado que a criação da matriz de similaridade de todos os usuários.
- O algoritmo usado, em alto nível, pode ser visto da seguinte maneira,

```
Alg Kmeans_ub_recommendation(data, user_id, n):
        - guardar a linha do usuario
        - tirar essa linha da tabela
        - treinar o modelo kmeans com o restante da tabela e guardar os
cluster
        - ver o cluster do usuario
        - calcular a media dos itens
        - retornar o n itens com maior media
```

- Com mais detalhes pode ser visto a seguir (baseado fortemente no python),

```
Alg Kmeans_ub_recommendation(data,user_id, number_recommendations):
        // o algortimo foi feito pensando que os dados estão organizados com
as linhas
        // representando os usuários e as colunas sendo os itens, alem da
existencia
        // de uma coluna chamada "user_avg_rating"

        user_data <- data[user_id] // dados do usuario estao na linha user_id
        alt_data <- data.drop(user_id)
        model <- Kmeans()
```

```
        model <- model.fit(alt_data)
        clusters <- model.predict(alt_data) //salvar os clusters dos demais
usuarios
        alt_data['cluster'] <- clusters // criacao de uma nova coluna
'clusters'
        user_cluster <- model.predict(user_data)
        user_data.append(user_cluster)
        alt_data <- alt_data.insert(user_data, user_id)
        alt_data['cluster' == user_cluster]  // restringir o dataset para
conter
                                                // apenas os usuarios do
cluster igual ao do usuario u

        items_means <- empty_list()

        // a funcao desse laco eh calcular os valores de media os items que
ainda nao
        // foram avaliados pelo usuario u, tambem sao descontados da media
aqueles
        // usuarios que nao avaliaram o item, para evitar que itens mais
avaliados
        // tenham vantagem

        for item in alt_data: //
                if alt_data[user_id, item] = 0:
                        temp_data <- alt_data[item != 0]
                        mean <- temp_data[item].mean()
                        items_means.append([mean, item])
        items_means.sort(reverse)

        return items_means[0:number_recommendations]
```

Item-based

- Assim como dito anteriormente, a premissa segue sendo a mesma.
- Nesse caso, quando se tem, além dos ratings, características sobre os itens é possível fazer recomendações baseadas em tais características.

- O pensado aqui foi, pegar itens bem avaliados de um usuário, e buscar itens que sejam similares a ele.
- O algoritmo em alto nível pode ser visto a seguir,

```
// em tese pegar do outro dataset os filmes que foram bem avaliadaos pelo
usuario

Alg Kmeans_ib_recommendation(data, watched_films, liked_films)
// aqui data eh uma tabela com o nome dos filmes, data de lancamento e o
genero
// data nao possui informacao alguma sobre o usuario
        - retirar do dataset aqueles itens que ja foram avaliados pelo usuario
        - criar modelo e formar os clusters
        - buscar os clusters dos filmes assistidos
        - do cluster deles, escolher filmes bem avaliados entre
usuarios(exportar da
         coluna avg_movie_rating)
        - retornar aqueles que mais aparecerem.
```

Um sistema mais ou menos completo

```
Alg: Sistema de recomendação(rating_data, items_characteristcs, User_u):

// um sistema de recomendacao pode ser hibrido com isso vou tentar retornar X
itens totais, sendo y baseados em usuarios similares a ele, z baseados em seus
itens avaliados e w baseados em popularidade

// para pegar os baseados em usuarios similares a ele pode-se usar um
algoritmo de user-based kmeans
        user_based_items <- Kmeans_user_based_recommendation()

// para pegar os itens baseados nos que o usuario gosta, pode-se usar um
algoritmo de item-based kmeans
        item_based_items <- Kmeans_item_based_recommendation()

// para pegar os itens mais populares, pode-se pegar aqueles itens mais
avaliados e que não estejam entre os itens ja avaliados pelo usuario U, nem
entre os ja recomendados anteriormente
```

```
        popularity_based_items <- popularity_based_items()

// por fim juntá-los todos em uma lista de recomendação e enviar ao usuario
        print('usuarios como voce tambem gostaram de {user_based_items},
                  baseado no que voce curtiu recentemente: {item_based_items}
                  em alta: {popularity_based_items}')
```