Projet : Segmentation de client et recommandation de films

Rebecca Leygonie Nemanja Kostadinovic

Problématique

Nous travaillons sur les données d'un centre commercial. Nous souhaitons catégoriser les clients en différents groupes partageant des caractéristiques communes. Nous souhaitons ensuite, pour un nouveau client, l'attribuer à un des groupes trouvés au-préalable.
De plus, le centre commercial souhaite ouvrir un cinéma, c'est pourquoi nous nous intéressons à la recommandation de films basée sur l'historique des avis des clients.

Données

Données clients : (https://www.kaggle.com/vjchoudhary7/customer-segmentation-tutorial-in-python)

Composé des colonnes suivantes: (CustomID, Gender, Age, Annual Income (k$),Spending Score (1-100))

Données des avis sur les films : (https://www.kaggle.com/rounakbanik/the-movies-dataset?select=ratings.csv

Composé des colonnes suivantes: (userId, movieId, rating, timestamp)

Données de description des films: (https://www.kaggle.com/rounakbanik/the-movies-dataset?select=movies_metadata.csv)

Composé des colonnes suivantes :
(adult,belongs_to_collection,budget,genres,homepage,id,imdb_id,original_language,original_title,overview,popularity,poster_path,production_companies,production_countries,release_date,revenue,runtime,spoken_languages,status,tagline,title,video,vote_average,vote_count)

Organisation du répertoire

input_data : les données clients et des films pour l'apprentissage de nos modèles.

src : contenant les sources python

main.ipynb : le fichier principal contenant la comparaison des modèles et le système final.

utils : répertoire des modèles utilisés dans le main.

annexe : répertoire contenant les analyses exploratoires de nos données ainsi que les fonctions utiles à la comparaison des modèles.

présentation : rapport et présentation des résultats de notre projet

Readme

Système final

Pour la partie segmentation client :

Algorithme de clustering, *Kmeans*, utilisé pour la segmentation de clients en plusieurs groupes (5 groupes).

Algorithme de classification, *RandomForest* pour classer un nouveau client dans un des groupes trouvés par l'étape précédente.

L'utilisation de ces deux algorithmes constitue notre approche hybride.

Pour la partie recommandation de films :

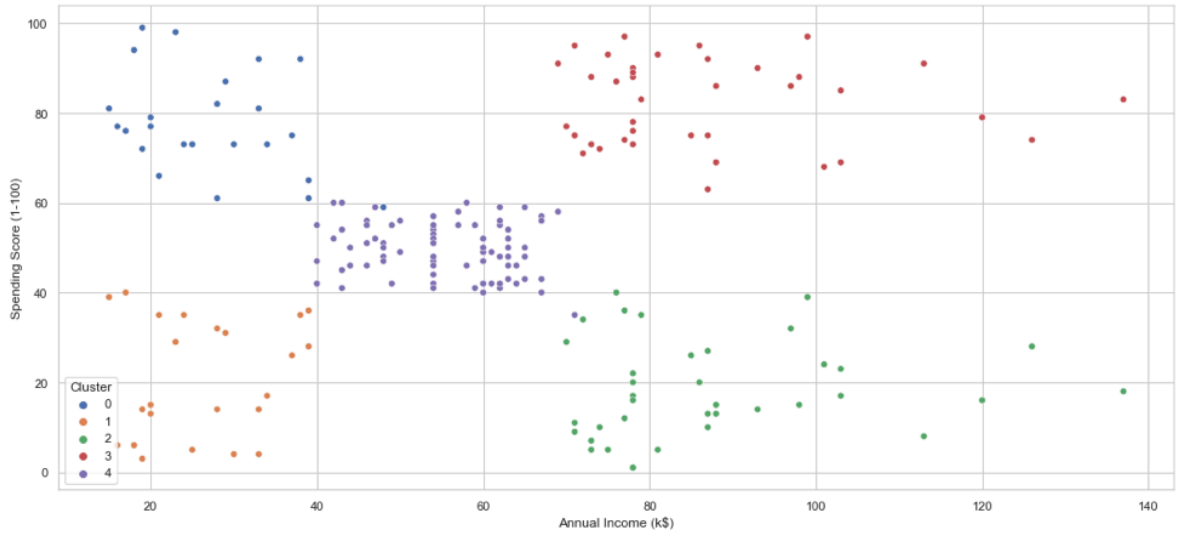
Jointure entre les clients et les notes de films (création de notre table pour traiter le sujet de la recommandation).

Utilisation d'un l'algorithme de filtrage collaboratif pour recommander des films en fonction des clients.

Résultats

Pour la segmentation de clients avec Kmeans (k=5):

```
Silhouette =  0.6276007113904088
```



Pour la classification avec RandomForest :

```
Accuracy = 100% (car le jeu de données ne contient pas assez
d'exemple ?)
```

```
+-------------------+-------+--------------------+--------------------+----------+
|           features|Cluster|       rawPrediction|         probability|prediction|
+-------------------+-------+--------------------+--------------------+----------+
|[18.0,33.0,1.0,92.0]|      0|[17.9473684210526...|[0.89736842105263...|       0.0|
|[19.0,64.0,1.0,46.0]|      4|[1.24466016467657...|[0.06223300823382...|       4.0|
| [19.0,81.0,1.0,5.0]|      2|[0.0,0.0357142857...|[0.0,0.0017857142...|       2.0|
| [20.0,16.0,0.0,6.0]|      1|[0.0,17.881868131...|[0.0,0.8940934065...|       1.0|
| [20.0,73.0,1.0,5.0]|      2|[0.0,0.0357142857...|[0.0,0.0017857142...|       2.0|
|[21.0,33.0,0.0,81.0]|      0|[18.9473684210526...|[0.94736842105263...|       0.0|
|[22.0,20.0,1.0,79.0]|      0|[19.9473684210526...|[0.99736842105263...|       0.0|
|[23.0,70.0,0.0,29.0]|      2|[0.0,5.8818681318...|[0.0,0.2940934065...|       2.0|
|[24.0,38.0,1.0,92.0]|      0|[16.9473684210526...|[0.84736842105263...|       0.0|
|[24.0,39.0,0.0,65.0]|      0|[16.9473684210526...|[0.84736842105263...|       0.0|
|[25.0,24.0,1.0,73.0]|      0|[18.9473684210526...|[0.94736842105263...|       0.0|
|[26.0,62.0,1.0,55.0]|      4|[0.24466016467657...|[0.01223300823382...|       4.0|
|[27.0,60.0,0.0,50.0]|      4|[0.24466016467657...|[0.01223300823382...|       4.0|
|[27.0,67.0,1.0,56.0]|      4|[0.24466016467657...|[0.01223300823382...|       4.0|
|[28.0,101.0,1.0,6...|      3|[0.0,0.0,0.0,20.0...|[0.0,0.0,0.0,1.0,...|       3.0|
|[30.0,137.0,1.0,8...|      3|[0.0,0.0,0.0,20.0...|[0.0,0.0,0.0,1.0,...|       3.0|
|[31.0,39.0,0.0,61.0]|      0|[10.5869561219610...|[0.52934780609805...|       0.0|
|[31.0,72.0,0.0,71.0]|      3|[0.83333333333333...|[0.04166666666666...|       3.0|
|[31.0,81.0,0.0,93.0]|      3|[0.0,0.0,0.0,20.0...|[0.0,0.0,0.0,1.0,...|       3.0|
|[32.0,76.0,0.0,87.0]|      3|[0.0,0.0,0.0,20.0...|[0.0,0.0,0.0,1.0,...|       3.0|
+-------------------+-------+--------------------+--------------------+----------+
```

Pour la recommandation de films avec ALS :

```
+-------+--------------------------------------------------+--------------------------------------------------
------------------------------+
|movieId|recommendations                                   |title
|
+-------+--------------------------------------------------+--------------------------------------------------
------------------------------+
|148    |[[83, 2.7878337], [169, 2.7877886], [168, 2.787404]] |The Secret Life of Words
|
|471    |[[83, 3.4891572], [169, 3.4891005], [168, 3.4886198]]|Bandyta
|
|496    |[[83, 3.1753914], [169, 3.17534], [168, 3.1749024]]  |Borat: Cultural Learnings of America for Make Benefi
t Glorious Nation of Kazakhstan|
|833    |[[83, 2.6024714], [169, 2.6024292], [168, 2.6020706]]|Umberto D.
|
|1088   |[[83, 3.0998192], [169, 3.0997689], [168, 3.0993419]]|Whale Rider
|
|1580   |[[83, 3.4191318], [169, 3.4190764], [168, 3.4186056]]|Released
|
|1591   |[[83, 2.5174723], [169, 2.5174315], [168, 2.5170846]]|Nowhere in Africa
|
|1645   |[[83, 3.3626971], [169, 3.3626425], [168, 3.362179]] |A Time to Kill
|
|1829   |[[83, 2.9150498], [169, 2.9150026], [168, 2.9146008]]|The Story of Adele H
|
|1959   |[[83, 3.4865506], [169, 3.4864936], [168, 3.4860134]]|Swept from the Sea
|
|2122   |[[83, 2.5333629], [169, 2.5333216], [168, 2.5329726]]|The Whole Ten Yards
|
|2142   |[[83, 2.924748], [169, 2.9247005], [168, 2.9242976]] |Cop Land
|
|3175   |[[83, 3.4257376], [169, 3.425682], [168, 3.4252102]] |Barry Lyndon
|
|4519   |[[83, 3.1578095], [169, 3.1577582], [168, 3.1573234]]|Conversations with My Gardener
|
|4935   |[[83, 3.2363183], [169, 3.236266], [168, 3.23582]]   |Howl's Moving Castle
|
|5156   |[[83, 2.7750869], [169, 2.7750423], [168, 2.7746596]]|Bicycle Thieves
|
|6466   |[[83, 3.4009705], [169, 3.4009151], [168, 3.4004467]]|Freddy vs. Jason
|
|6620   |[[83, 3.637123], [169, 3.637064], [168, 3.6365633]]  |[{'iso_3166_1': 'US', 'name': 'United States of Amer
ica'}]                                                |
|7340   |[[83, 2.8415718], [169, 2.8415256], [168, 2.841134]] |If you've got a taste for terror... take Carrie to t
he prom.                                              |
|8592   |[[83, 3.1870253], [169, 3.1869733], [168, 3.186535]] |Dick Tracy
|
+-------+--------------------------------------------------+--------------------------------------------------
------------------------------+
```

D'après la figure ci-dessus pour l'utilisateur (3) nous pouvons recommander le film The Secret Life of Words ou Bandyta.

Le modèle ALS :

Les données utilisées sont explicites car ce sont des données pour lesquelles nous avons une sorte de notation. Les notes vont de 1 à 5 sur l'ensemble de données. Nous savons ici à quel point un utilisateur aime ou n'aime pas un film par la note qu'il a proposé.

L'alternance des moindres carrés (ALS) est un modèle que nous utiliserons pour ajuster nos données et trouver des similitudes.

L'ALS est un processus d'optimisation itératif où nous essayons à chaque itération d'arriver de plus en plus près d'une représentation factorisée de nos données d'origine.
Nous avons notre matrice originale R de taille (u x i) avec nos utilisateurs, les *Id* de films et une note. Nous voulons ensuite trouver un moyen de la transformer en une matrice avec les utilisateurs et les caractéristiques cachées de taille (u x f) et une matrice avec les éléments et les caractéristiques cachées de taille (f x i). Dans U et V, nous avons des pondérations pour la façon dont chaque utilisateur/film se rapporte à chaque caractéristique. Nous calculons U et V de manière à ce que leur produit se rapproche le plus possible de R : $R \approx U \times V$.

En attribuant aléatoirement les valeurs de U et V et en utilisant les moindres carrés de manière itérative, nous pouvons déterminer les poids qui donnent la meilleure approximation de R.
L'approche des moindres carrés dans ses formes de base signifie qu'il faut ajuster une ligne aux données, mesurer la somme des carrés des distances de tous les points à la ligne et essayer d'obtenir un ajustement optimal en minimisant cette valeur.

Avec l'approche des moindres carrés alternés, nous utilisons la même idée mais nous alternons itérativement entre l'optimisation de U et la fixation de V et vice versa. Nous faisons cela pour chaque itération afin d'arriver plus près de $R = U \times V$.

Observation: Nous obtenons un algorithme capable de prédire les films qu'un client va aimer en fonction de ses précédentes notes et les notes de chaque film donné par les autres utilisateurs.

Expérimentations

Afin de trouver le meilleur système pour notre cas d'usage. Nous avons testé différents modèles pour chaque partie (en fonction des disponibilités sur SparkML).

Pour la partie segmentation de clients, nous avons comparer (par le biais de la mesure *silhouette*) des modèles Kmeans et BisectingKmeans pour k variant de 2 à 10.
Le meilleur Kmeans était pour k=5 avec silhouette = 0.62
Le meilleur BisectingKmeans était pour k=9 avec silhouette = 0.5

Nous avons donc utilisé Kmeans avec k=5 pour clusteriser nos clients.

Pour la classification, nous avons comparé RandomForest et Décision Tree.
L'accuracy de RandomForest = 100% tandis que celle de Decison Tree = 95%
Nous expliquons le résultat parfait de RandomForest par le fait que le jeu de données ne contient pas assez d'exemple.

Pour un nouveau client donné, il se verra donc attribuer à un des 5 groupes par le modèle RandomForest.

Enfin, pour la recommandation de films, nous utilisons un modèle de filtrage collaboratif.