30/10/2020

Projet Machine Learning: Analyse d'un dataset

Encadrer par M. Benjamin DALLARD

Andriamananjaka Stephane Rakotozanany

Table des matières

INTRODUCTION

Notre projet en Machine Learning consiste à analyser un dataset. Ces données ont été obtenues sur le site Kaggle. Ces données regroupent les informations sur tous les jeux vidéo sortis jusqu'en 2016.

On y retrouve les informations suivantes sur le jeu vidéo :

- Le nom
- La plateforme
- L'année de sortie
- Le genre
- Le studio qui l'a publié
- Le nombre de vente en million en Amérique du nord
- Le nombre de vente en million en Europe
- Le nombre de vente en million au Japon
- Le nombre de ventes en million dans les autres pays du monde
- Le nombre de ventes en million dans le monde
- La note du critique
- Le nombre de critiques
- La note de l'utilisateur
- Le nombre d'utilisateur ayant donnée une note
- Le développeur
- La classification (PEGI)

Notre but dans ce projet est d'étudié si on peut prévoir la note de l'utilisateur en fonction des autres données du tableau.

1. Le dataset

Comme vu dans l'introduction on retrouve plusieurs informations sur toutes les sorties de jeux vidéo dans le monde. Au total nous avons 16179 jeux et 16 colonnes d'information.

```
dataGamesSales.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16719 entries, 0 to 16718
Data columns (total 16 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Name             16717 non-null  object
 1   Platform         16719 non-null  object
 2   Year_of_Release  16450 non-null  float64
 3   Genre            16717 non-null  object
 4   Publisher        16665 non-null  object
 5   NA_Sales         16719 non-null  float64
 6   EU_Sales         16719 non-null  float64
 7   JP_Sales         16719 non-null  float64
 8   Other_Sales      16719 non-null  float64
 9   Global_Sales     16719 non-null  float64
 10  Critic_Score     8137 non-null   float64
 11  Critic_Count     8137 non-null   float64
 12  User_Score       10015 non-null  object
 13  User_Count       7590 non-null   float64
 14  Developer        10096 non-null  object
 15  Rating           9950 non-null   object
dtypes: float64(9), object(7)
memory usage: 2.0+ MB
```

Néanmoins ce jeu de donnée n'est pas tout à fait prêt à être étudier.

I. Nettoyage du dataset

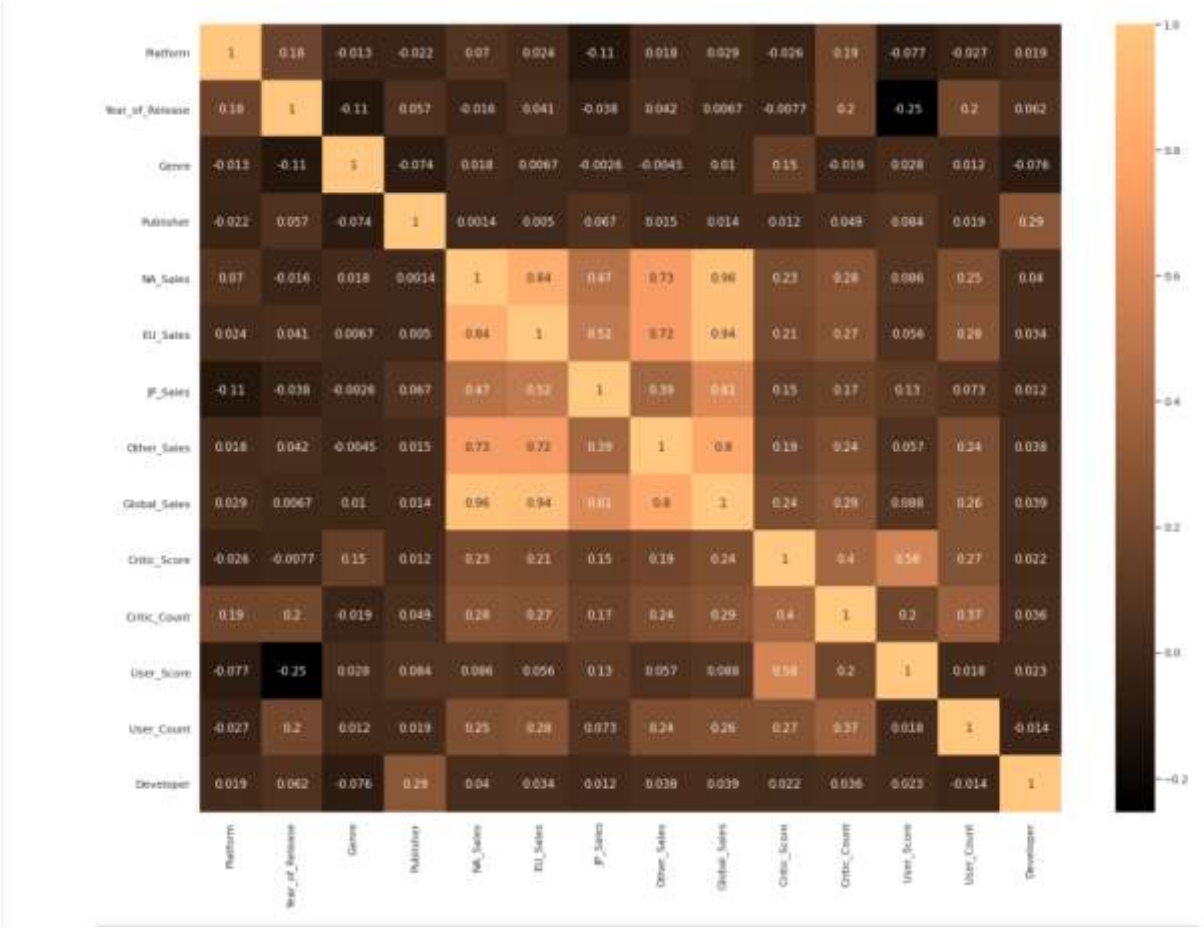
Pour pouvoir commencer à l'étudier nous avons d'abord commencé par l'étape de « cleaning » du dataset.

La première étape est de retirer toute les valeurs NaN (Not a Number) du tableau grâce à la fonction « *pandas.dropna()* ». Ce qui nous ramène ainsi à un tableau sans NaN de 6825 lignes.

II. Encoding du dataset

Dans cette étape nous allons transformer toutes les valeurs qualitatives du tableau en valeur quantitative grâce à la fonction du package *sklearn* : « *labelencoder.fit_transform()* ». Cette focntion

Cette dernière nous permet de dresser la matrice de corrélation afin de déterminer l'influence d'une valeur à une autre. On obtient alors :



Ici nous cherchons à étudier la note de l'utilisateur et on peut voir déjà que cette dernière est très corrélée à l'éditeur et aux ventes. On peut en déduire plusieurs théories à partir de ces informations comme certains éditeurs possèdent la confiance des joueurs.

III. Data Modeling

Le principal but de notre projet ici est de notre projet est de étudier le dataset afin de prédire les jeux qui sont aimés par les joueurs en fonction des données du tableau. Pour ce faire nous allons entrainer un certain nombre de lignes, sur lesquelles on va appliquer des algorithmes de classification fourni par le package « *sklearn* ». Ce qui nous donnera un score en pourcentage sur la précision donc les algorithmes ont pu déterminer les jeux bien notés.

J'ai donc choisis trois algorithme vu en cours afin d'étudier mon dataset :

- Random Forest Classifier
- KNeighbors Classifier
- Support Vector Classification

A ces derniers ont peux ajouter des paramètres afin d'augmenter leur précision.

Pour le RFC nous avons utilisé :

- « *n_estimator* » : nombre d'arbres dans la foret
- « *max_features* » : nombre de feature à considérer à chaque split
- « *max_depth* » : la profondeur de l'arbre

On a alors obtenu :

```
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, random_state = 0)
randomforest = rf(n_estimators = 35, max_features = 10, max_depth = 35)
randomforest.fit(X_train, Y_train)
Y_pred = randomforest.predict(X_test)
prob = randomforest.predict(X_test)

print(accuracy_score(Y_test, Y_pred)*100 , '%')
```

```
4.862331575864089 %
```

Pour le KNN nous avons utilisé :

- « *n_neighbors* » : nombre de voisins
- « algorithm » : l'algorithm à utiliser pour calculer le plus proche voisins
- « *weights* » : la fonction weight utilisée pour la prédiction

On a alors obtenu :

```
#Hyperparametters input
knn = KNN(n_neighbors = 35, weights = 'distance',algorithm  = 'auto')
knn.fit(X_train, Y_train)
Y_pred = knn.predict(X_test)

print(accuracy_score(Y_test, Y_pred)*100, '%')
```

```
3.9835969537199762 %
```

Pour le SVC nous avons utilisé :

- « C » : paramètre de régularisation
- « kernel » : le noyau utilisé par l'algorithme
- « max_iter » : Limite stricte des itérations dans le solveur

On a alors obtenu :

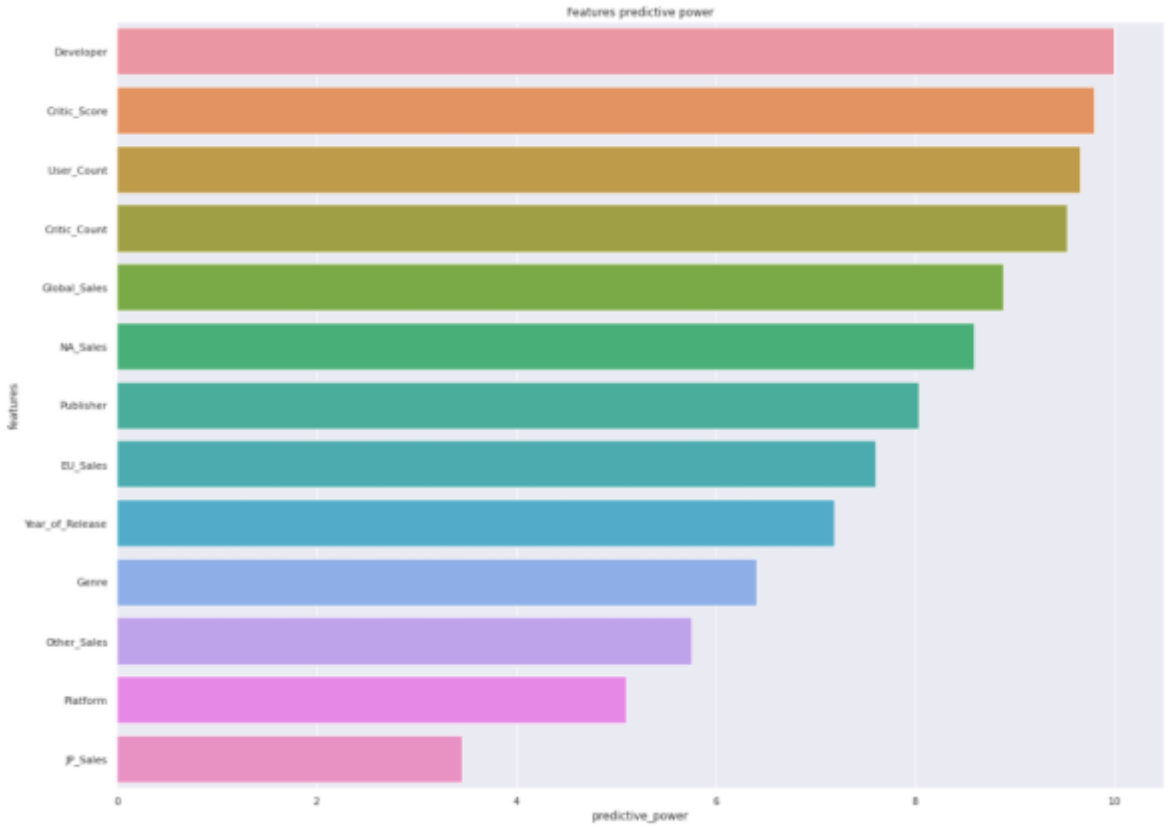
```
#Hyperparametters input
svc = SVC(C=1.0 , kernel = 'poly', max_iter=-1)
svc.fit(X_train, Y_train)
Y_pred = svc.predict(X_test)

print(accuracy_score(Y_test, Y_pred)*100 , '%')
```

```
3.8078500292911546 %
```

IV. Importance des features

Dans cette partie nous allons regarder laquelle des features qui a le plus influencer la décision de l'algorithme RFC.

Pour cela on va entrainer le modèle afin de faire un test d'importance sur les features. On obtient les résultats de ce test grâce à la fonction « *rf.feature_importances_* ». Et on l'observe les résulats à travers le graphique :



On observe que le développeur du jeu influe beaucoup sur le choix de la note par l'algorithme ce qui est presque le cas dans la réalité.

Conclusion

Pour conclure, nous avons pu faire des analyses sur une table de données assez volumineuses. Cependant cela reste une table très réduite par rapport au flux de données que l'on peut trouver dans une analyse portant dans des domaines beaucoup plus grand et important.

Le choix de l'algorithme est aussi important dans le traitement de données et dans la prédiction d'une valeur. Malgré les faibles résultats obtenus à la fin de chaque algorithme, on a pu quand même déduire une influence élevée du développeur sur la prédiction de l'algorithme avec l'influence du score de la critique pas très loin derrière. Un supposition proche de la réalité.S

Bibliographie

- Kaggle :
- Cours de Machine Learning
- Jupyter Notebooks