Rapport Apprentissage

Membres de l'équipe :

DE CUNIAC Axel
DE FERRIERES DE SAUVEBOEUF Remi
MACHER Vadim

Notre projet a été codé entièrement sur google colab : une plateforme de code collaborative. Vous pouvez trouvez le lien du projet ici :
https://colab.research.google.com/drive/1VnxFZScBdBxv86FyZ7x3a5USeFRLY5Bb
Cela vous permettra de l'exécuter si vous le souhaitez. De plus comme nous avons codé sur colab il sera plus aisé de comprendre le code sur colab. Cependant vous trouverez aussi notre projet en format python dans le rendu.

I. Exploration du dataset

La première étape de notre projet a été d'explorer et d'analyser le jeu de données. Nous avons remarqué qu'il était nécessaire de le normaliser pour extraire les informations des JSON contenues dans les colonnes. Pour cela nous avons utilisé le code mis à disposition en R sur Brightspace. Le reste du projet a cependant été conçu en python.

a. Valeurs uniques

Tout d'abord nous avons cherché à savoir ce que contenaient les colonnes de notre jeu de données normalisés. Nous avons donc affiché le nombre de valeurs uniques par colonnes (cf. tableau ci-joint). Nous nous sommes rendus compte qu'un certain nombre de colonne contenait qu'une valeur unique. Or les modèles de Machine Learning ont besoin de variance au sein d'une colonne pour qu'elle soit exploitable, nous avons donc fait le choix de supprimer toutes ces colonnes qui ne contenaient qu'une seule valeur unique.

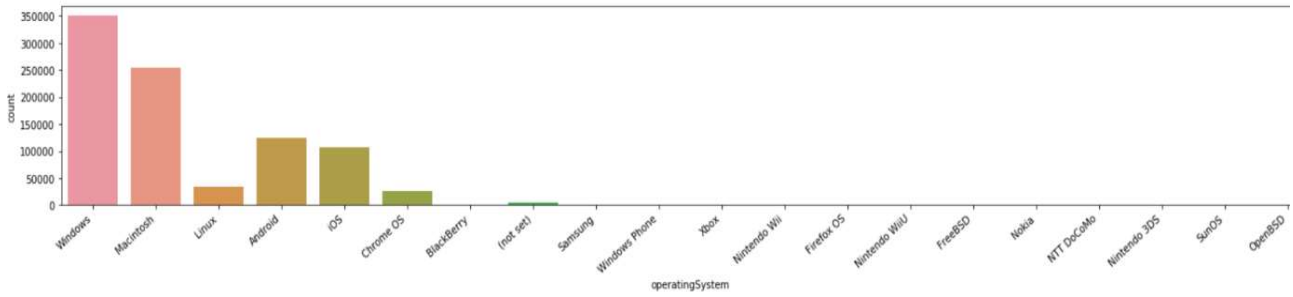
sessionId	902755
socialEngagementType	1
visitId	886303
visitNumber	384
visitStartTime	887159
browser	54
browserVersion	1
browserSize	1
operatingSystem	20
operatingSystemVersion	1
isMobile	2
mobileDeviceBranding	1
mobileDeviceModel	1
mobileInputSelector	1
mobileDeviceInfo	1
mobileDeviceMarketingName	1
flashVersion	1
language	1
screenColors	1
screenResolution	1

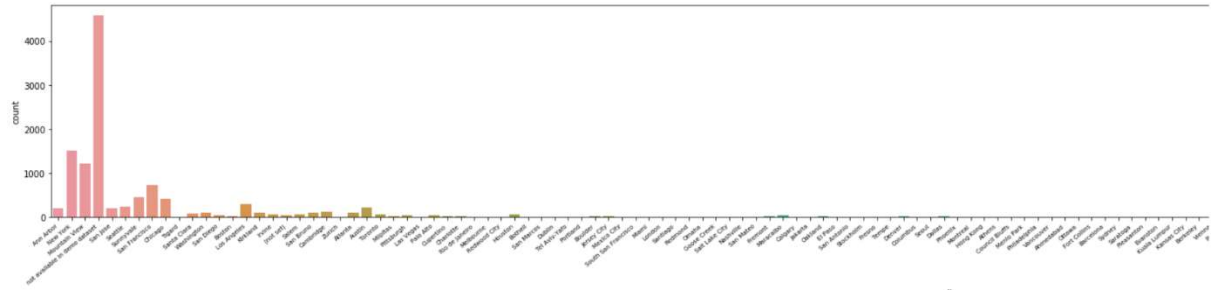
b. Valeurs non explicatives

Ensuite nous avons analyser les colonnes qui nous restaient. Ces dernières sont :

```
['channelGrouping', 'date', 'fullVisitorId', 'visitNumber',
 'visitStartTime', 'browser', 'operatingSystem', 'isMobile',
 'deviceCategory', 'continent', 'subContinent', 'country', 'region',
 'metro', 'city', 'networkDomain', 'hits', 'pageviews', 'bounces',
 'newVisits', 'transactionRevenue', 'campaign', 'source', 'medium',
 'keyword', 'isTrueDirect', 'referralPath', 'adwordsClickInfo.page',
 'adwordsClickInfo.slot', 'adwordsClickInfo.gclId',
 'adwordsClickInfo.adNetworkType', 'adwordsClickInfo.isVideoAd',
```

On affiche les différents graphiques pour se représenter la répartition des valeurs dans le jeu de données.



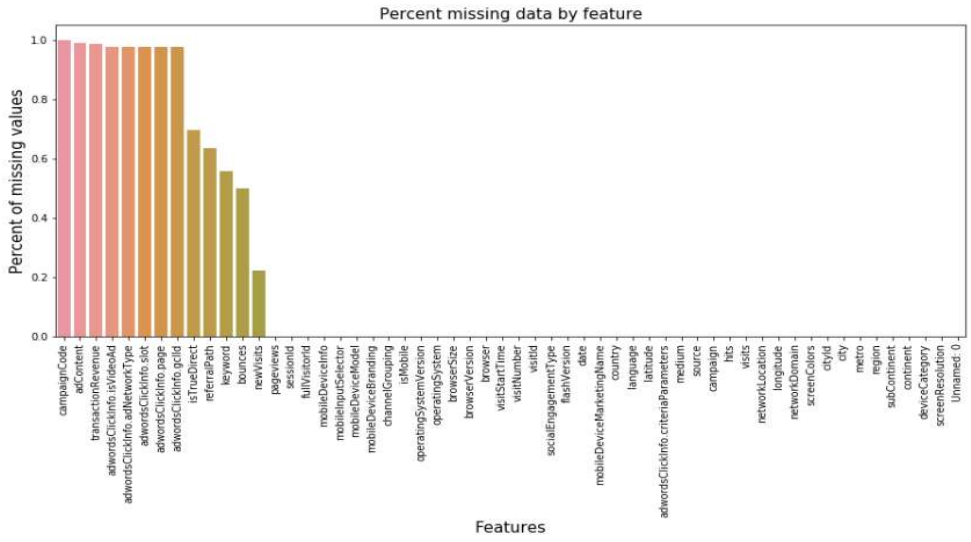


c. Gestion des NAN

Pour connaitre la répartition des NaN dans notre jeu de donnée, nous avons déterminé le nombre de NaN par colonne. Ensuite, nous avons calculer le rapport entre le nombre de NaN et le nombre de valeur par colonne.

Comme on peut le voir sur le graphique représentant le pourcentage de valeurs manquantes par colonnes, peu de colonnes contiennent un taux important de valeurs manquantes.

On remplit ainsi les colonnes de NaN par des 0 pour les variables quantitatives, de string ayant la mention “unknown” pour les variables catégorielles.



II. Features Engineering

a. Gestion des dates

Pour la gestion des dates, on s'intéresse d'une part à savoir si les joueurs ont joué pendant la semaine ou en week-end. On peut de ce fait savoir si les joueurs jouent plus pendant la semaine ou alors s'ils préfèrent jouer pendant le weekend. Cette information étant plus pertinente qu'une date.

D'autres part, on s'intéresse au mois de jeu des joueurs. On peut ainsi s'apercevoir que les joueurs jouent de manière équitable selon les différents mois.

b. Gestion de l'heure de la session

Pour l'heure de la session, on a découpé la journée en différentes catégories : "morning", "afternoon" et "night" afin de connaitre les habitudes de jeu des joueurs et de pouvoir tirer un sens nouveau à la valeur heure de session.

c. Gestion des données géographiques avec PIB

Nous avons décidé d'ajouter une colonne PIB qui correspond au PIB du pays du joueur en question. Etant donné le nombre de pays différents dans le jeu de données, le PIB permet d'ajouter un autre sens à la valeur pays.

d. Gestion des données catégorielles

Nous avons constaté que plusieurs colonnes contenaient des données catégorielles en majorité de type "objet". Pour éviter ce type et le remplacer par une variable numérique nous avons créé des colonnes "dummies" pour chaque colonne catégorielle. Cela consiste à crée des nouvelles colonnes booléen (0 et 1) pour chaque valeur différente de la colonne de base.

e. Création d'un premier modèle par session

Tout d'abord nous avons créé un modèle par session. Cela permettait de faire une prédiction pour chaque session du joueur et donc de faire une prédiction par ligne du jeu de données. Avec ce type de modèles nous parvenions à des R2 d'environ 10%. De plus comme nous l'avons précédemment ce type de modèle nécessité de faire beaucoup de « dummies » sur les colonnes ce qui nous amenait à des jeux d'entrainements de tailles très importantes.

f. Agrégation des données par joueur

En explorant les données, on s'est aperçu que certains joueurs jouaient plusieurs fois. On a ainsi regroupé les données pour chaque joueur unique avec la colonne fullVisitorId. En effet on se disait qu'il pouvait être intéressant de suivre le parcours d'un joueur sur plusieurs sessions plutôt que sur une seule car cela permettrait potentiellement de mieux capter son comportement. Ci-joint, voici la méthode utilisée pour chaque feature. On prend ainsi la somme des transactionRevenue pour chaque joueur.

Les process précédents ont tous été testé mais certain n'ont pas été retenu (notamment la gestion de donnée catégorielle) car le score

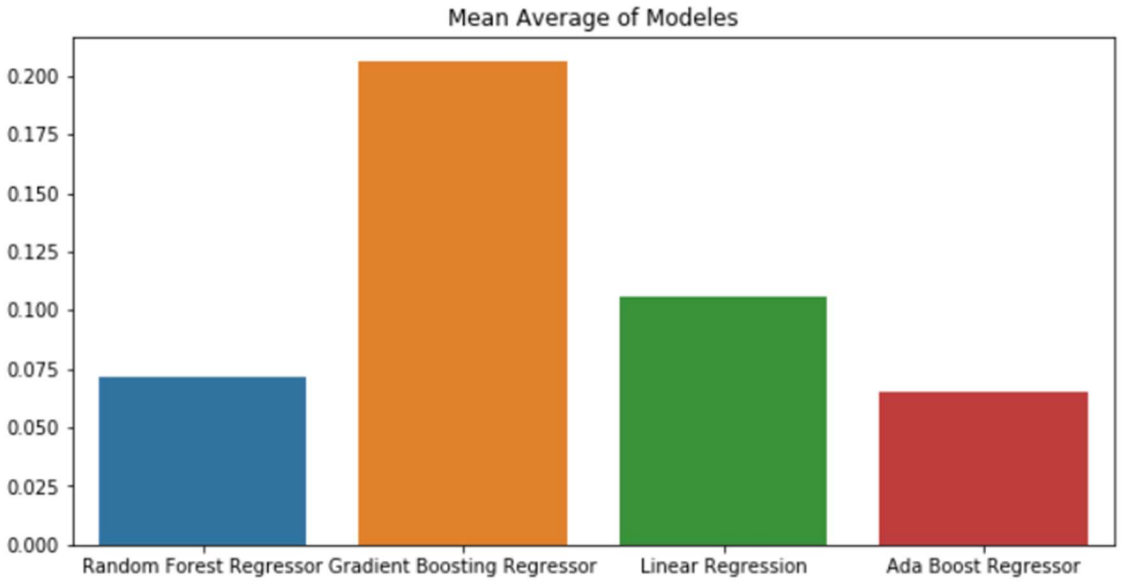
```
aggregate_data = data.groupby("fullVisitorId").agg({
    "date" : "count",
    "visitStartTime" : "count",
    "hits" : "sum",
    "pageviews" : "sum",
    "bounces" :"sum",
    "newVisits" : "count",
    "transactionRevenue" : "sum",
    "isMobile" : "max",
    "deviceCategory" : "count",
    "continent" : 'count',
    "subContinent" : 'count',
    'country' : 'count',
    'region' : 'count',
    'city' : 'count',
    'networkDomain' : 'count',
    'isTrueDirect' : "max",
    "operatingSystem" : 'count',
    "adwordsClickInfo.page" :"sum",
    'adwordsClickInfo.slot' : 'count',
    'adwordsClickInfo.gclId' :'count',
    'adwordsClickInfo.adNetworkType' : 'count',
    "adwordsClickInfo.isVideoAd" : "max",
    "adContent" : 'count',
    "keyword" : "count",
    "source" : 'count',
    "medium" : "count",
    "campaign" : "count"
```

III. Les modèles de Machine Learning

a. Choix du modèle

Pour les modèles, nous sommes partis sur une régression. On a ainsi testé 4 différents algorithmes de Machine Learning :

- Linear Regression
- Random Forest Regressor
- Gradient Boosting Regressor
- Ada Boost Regressor

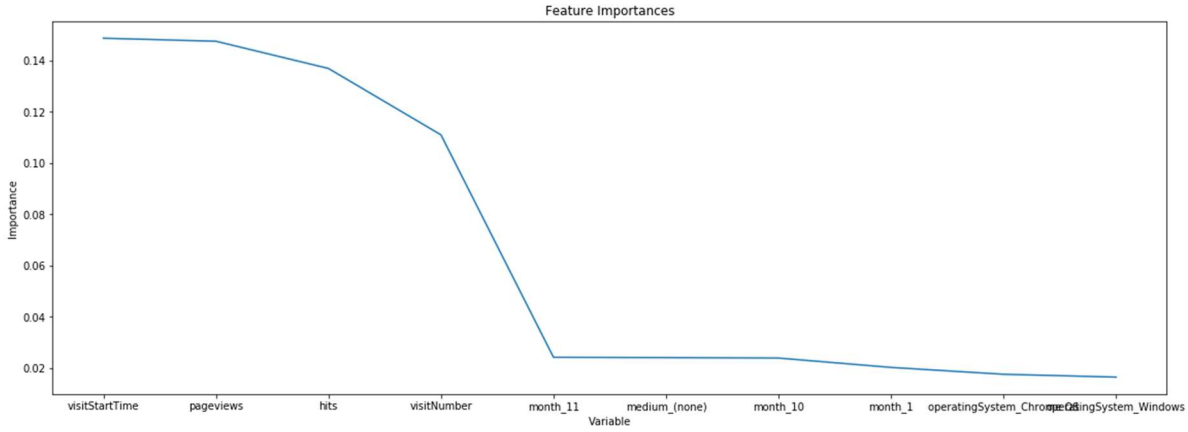


Comme on peut le voir sur le graphique ci-dessus qui représente le R2 moyen de chaque modèle le meilleur modèle pour notre cas semble être le Gradient Boosting Regressor.

b. Dépouillement du modèle

Pour le dépouillement des modèles, nous avons principalement analysé la feature importance issue de notre Gradient Boosting Regressor. On remarque certaines features sont très importantes pour le

modèle : VisitStartTime, pageviews, hits sont chacune utilisés dans 15% des cas environ. Ensuite d'autres semblent aussi être porteuses d'informations comme VisitNumber par exemple.



Afin d'optimiser notre modèle de Gradient Boosting Regressor nous avons essayé d'entrainer le modèle en faisant varier deux paramètres principales : le learning rate (taux d'apprentissage réduit la contribution de chaque arbre du learning rate) et le n_estimators (Le nombre d'étapes de renforcement à effectuer).

En faisant varier ces deux paramètres nous avons obtenues les meilleurs résultats pour learning_rate =0.01 (vs 00.1 par défaut) et le n_estimators=100 (100 aussi par défaut). Nous arrivons finalement à un score de R2 d'environ 22%.

En termes d'améliorations futures nous pensions possiblement utiliser le NLP pour traiter certaines colonnes comme le keyword ou extraire des informations des liens par exemple.