Prédicteur d'achat
Projet n°3 du Parcours Data Analyst

Butscher Lucas 02-2019

Table des matières

I. Intro

L'objectif du projet que j'ai choisi est de réaliser un prédicteur d'achat en fonction de différentes informations collectées lors des visites sur le site web Google Merchandise Store. C'est un projet choisi sur la plateforme Kaggle. J'ai décidé pour ce sujet libre d'utiliser des données prises sur Kaggle puisqu'un Data Analyst aura au cours de sa carrière souvent besoin de travailler à l'aide de cette plateforme. J'ai donc à nouveau développé des compétences en me familiarisant avec elle.

La base de données que j'utilise au cours de ce projet a été publiée par Google dans le cadre d'une compétition Kaggle. J'en ai donc tiré les données ainsi que la problématique que j'ai adapté à mon objectif personnel.

C'est pour cela que je cherche à déterminer les variables expliquant au mieux le comportement d'achat de ce site.

II. Nettoyage

II.A. Un problème de taille

La base de données sur laquelle nous allons travailler est contenue dans le fichier "train_v2.csv". Cette base contient plusieurs millions de ligne et treize colonnes. Ainsi vint ma plus grande problématique : les performances de mon ordinateur. Celui-ci est trop âgé et trop peu performant pour effectuer des actions sur une base de données de cette ampleur. J'ai donc dû faire un choix. Plusieurs possibilités me sont donc apparues : soit changer de projet soit réduire la taille de mon DataFrame.

Ayant déjà passé beaucoup de temps à choisir ce sujet, je décide donc de ne pas l'abandonner au profit d'un autre qui m'intéresserait moins et qui aurait peut-être présenté les mêmes problèmes.

Je devais donc choisir une méthode pour réduire la taille de mes données. L'idée de retirer des colonnes a très vite été écartée aux vues de ce que je souhaitais réaliser : déterminer les variables expliquant au mieux le comportement d'achat des visiteurs du site.
Je devais alors conserver toutes mes colonnes mais choisir les individus que je garderai. Un filtre aléatoire de mes données basé sur l'index était alors une bonne possibilité. Cependant le caractère "aléatoire" de cette possibilité me parut trop incertain. J'ai donc décidé de ne conserver qu'un seul mois. En effet, les données du DataFrame sont étalés sur 24 mois : de janvier 2017 à décembre 2018.

Ne prendre qu'un seul mois me permettait donc de diviser la taille de mes données par 24 (environ). Je décide alors de sélectionner un mois n'ayant pas d'évènement particulier qui pourrait venir perturber les comportements d'achat des clients. J'exclue donc la période de fêtes de fin d'année ainsi que les soldes et les vacances d'été. Plusieurs mois restent donc éligibles, je choisi arbitrairement mon mois de naissance, mars. Je sélectionne l'année 2018 pour la proximité dans le temps.

En effet, comme le dit Gregg Thaler : "*Traitez les données comme du poisson, pas du vin... elles empirent avec l'âge, ne deviennent pas meilleures*"[1]. Prendre les données les plus récentes est donc certainement la meilleure solution.
Mon DataFrame est à présent de taille raisonnable pour mon vieil ordinateur.

II.B. Traitement des colonnes JSON

Les différentes variables de mon DataFrame sont les suivantes :

-**channelGroupig** ; **customDimensions** ; **date** ; **device** ; **fullVisitorId** ; **geoNetwork** ; **socialEngagementType** ; **totals** ; **trafficSource** ; **visitId** ; **visitNumber** ; **visitStartTime**

Cinq colonnes parmi celles-ci sont dans un format qui complique l'analyse de données. Voici par exemple la première ligne de ma colonne **totals** :

```
print(df_train.totals.loc[0])
```

```
{"visits": "1", "hits": "1", "pageviews": "1", "bounces": "1"}
```

C'est un format JSON. En réalité les cinq colonnes dans ce format contiennent chacune plusieurs variables. Ce sont possiblement des éléments permettant d'expliquer le comportement d'achat. Je vais donc convertir les colonnes de ce format en un DataFrame regroupant les différentes variables. J'y appliquerai mon nettoyage d'outliers, de doublons, de valeurs manquantes et de vraisemblance puis j'ajouterai mes 5 nouveaux DataFrames à celui d'origine via la commande "merge". J'ai utilisé les fonctions "json.loads()" ainsi que "json.json_normalize()" dans le but de répartir mes informations dans de nouveaux DataFrames. Voici ci-dessous la fonction utilisée :

```
def string_to_dict(dict_string):
    return json.loads(dict_string)
```

```
def create_new_dataset(df,json_col):
    new = df.copy()
    colonnes = new.columns
    for col in json_col:
        print(col)
        new[col] = new[col].apply(string_to_dict)
        new = pd.concat([new,(pd.io.json.json_normalize(new[col]))], axis=1

    for i in colonnes:
        new = new.drop(columns=i)

    return new
```

II.C. Création de variables

Il est important de retirer les différentes erreurs qui pourraient fausser nos interprétations quant à l'importance d'une variable pour notre objectif. Ainsi, une fois toutes mes données extraites, les différents nettoyages usuels ont été apportés, sans faire un nettoyage plus poussé. Celui-ci sera approfondi lors de la phase exploratoire. Il m'est alors possible de créer certaines variables à l'aide

[1] "Contact data ages like fish not wine... it gets worse as i gets older, not better", Gregg Thaler, Pricipal, Business Development at Marketo, an Adobe Company

de celles à ma disposition dans le but de faciliter les prochaines étapes de mon travail. Ainsi j'ai créé les variables **jour** et **heure** qui représentent le jour de la semaine où la visite à eu lieu (lundi, mardi, etc.) ainsi que le nombre correspondant à l'heure de la visite (en exceptant les minutes et les secondes). Celles-ci auront peut-être un impact sur le comportement d'achat.

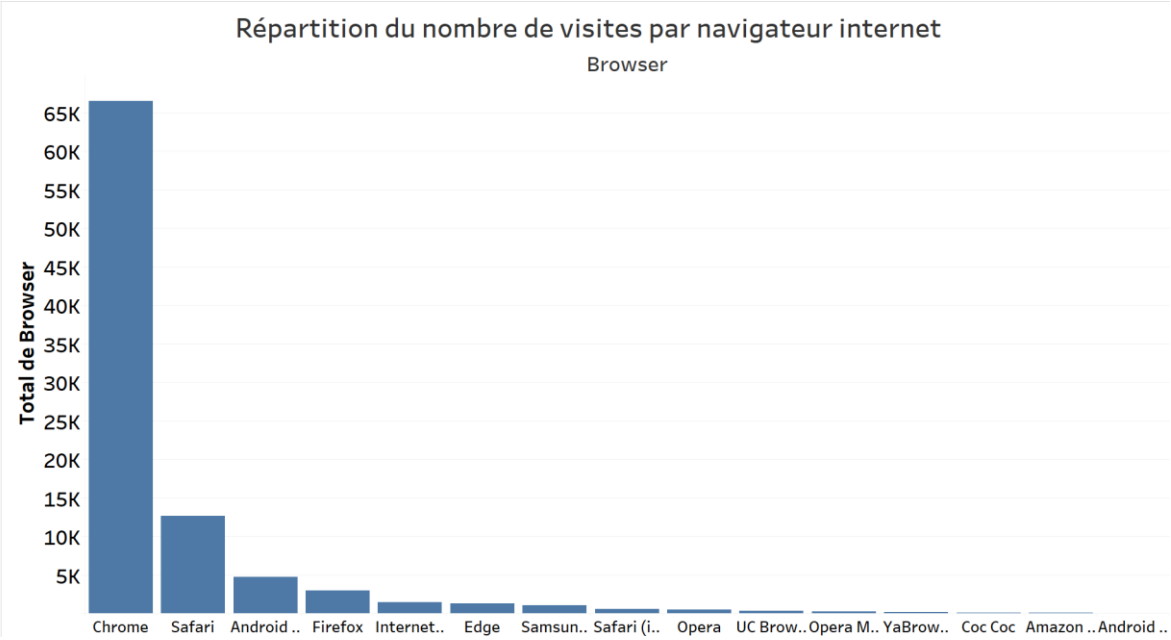
J'ai aussi étudié l'impact des minutes mais celui-ci ne semblait pas réellement corrélé à l'achat ou non. Par la suite j'ai calculé le total des visites sur le site web par utilisateur sur le mois ainsi que le total des pages vues par utilisateur sur le mois en utilisant l'id de la visite ainsi que l'id du client pour ma fonction "groupby()".

III. Analyse exploratoire

Je vais vous présenter l'analyse exploratoire de trois variables : **Browser**, **pageviews** et **OS**. Cette étape est primordiale. Elle permet de prendre mieux connaissance des données que nous analysons. Ben Shneiderman disait : " La visualisation nous donne des réponses à des questions que nous n'avions même pas."[2] Ainsi, cette étape va nous permettre à la fois de compléter le nettoyage de nos données et de percevoir ou non des possibles corrélations entre des variables et l'achat. Ainsi, cela se présente comme une pré-séléction des facteurs expliquant au mieux le comportement d'achat.

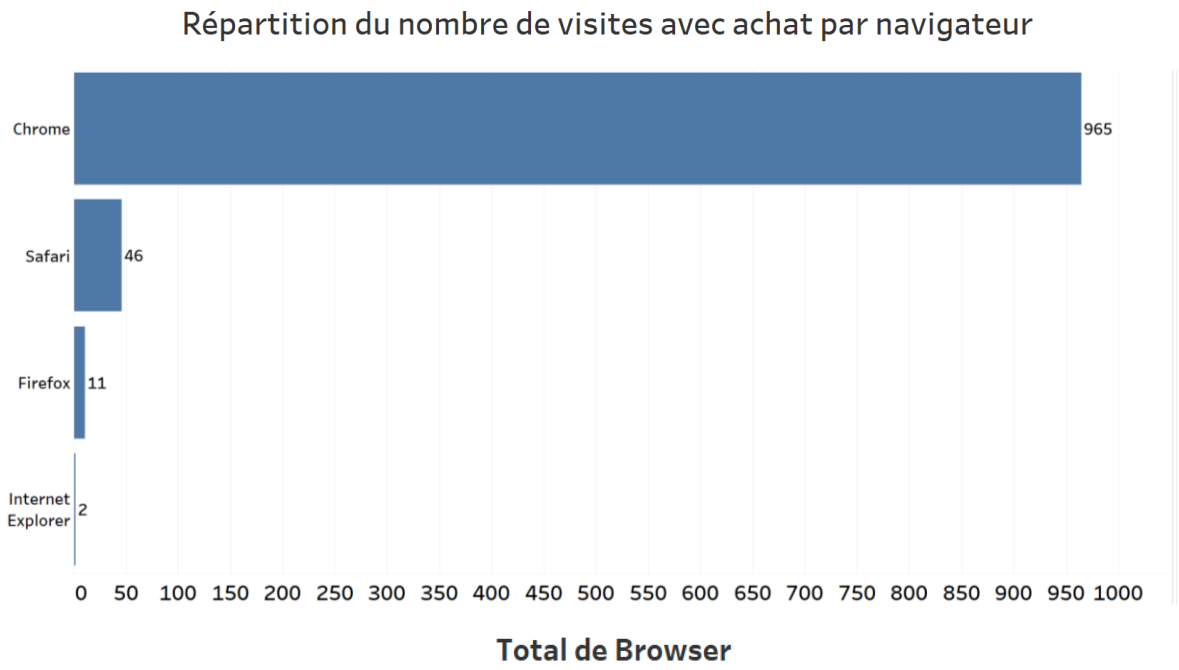
III.A. Navigateur internet

La variable **Browser** représente le navigateur utilisé par l'usager lors de sa visite sur le site. Sur le graphique ci-dessous est représentée la répartition du nombre de visites sur le site par navigateur internet utilisé par l'usager.



[2] "Visualization gives you answers to questions you didn't know you had" - Ben Shneiderman, Scientist and professor at Human Computer Interaction Lab , University of Maryland

Le graphique suivant représente les mêmes informations mais uniquement pour les visites ayant abouti à un achat.



Nous remarquons que des achats se concluent uniquement sur 4 navigateurs. Or les données en comptent plus de trente différents. Le navigateur Chrome est le plus utilisé. Il représente 78% des visites sur le site en général et 94% lors d'un achat. Cependant les navigateurs les moins utilisés peuvent-être source d'erreur. En effet, le nombre de visites effectués sur ceux-ci n'est statistiquement pas assez conséquent pour être représentatif. Mon échantillon étant assez grand, je prends donc la décision de supprimer de mes données les sites ayant été utilisés moins de mille fois sur le mois de mars.

Il me reste alors 6 navigateurs différents :

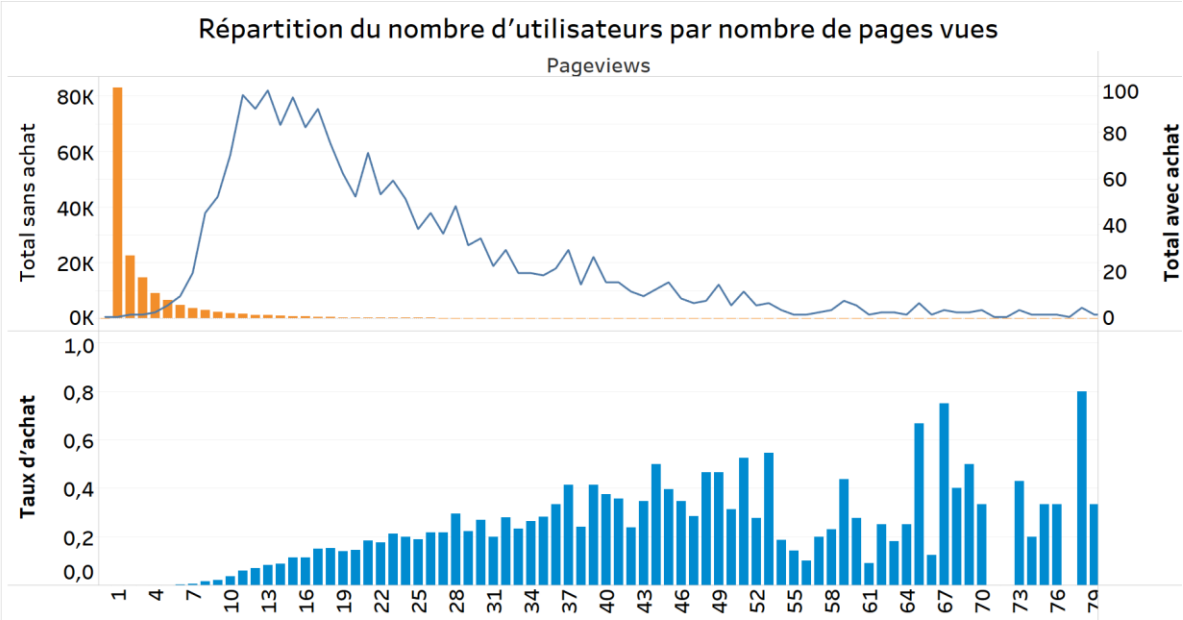
Chrome ; **Safari ; Android ; Firefox ; Internet Explorer**[3] **; Edge**

Le navigateur Android est le troisième plus utilisé sur ce site. Cependant aucun achat n'a été effectué avec ce navigateur. Etant uniquement un navigateur mobile ou tablette, je suppose donc qu'il existe une corrélation entre l'achat et le type de matériel utilisé (**device**). Aussi, Chrome est le navigateur ayant le taux de conversion le plus élevé (1,45% contre respectivement 0.37, 0.36 et 0.14% pour Firefox, Safari et IE). Je suppose alors une corrélation entre le navigateur internet utilisé et l'achat.

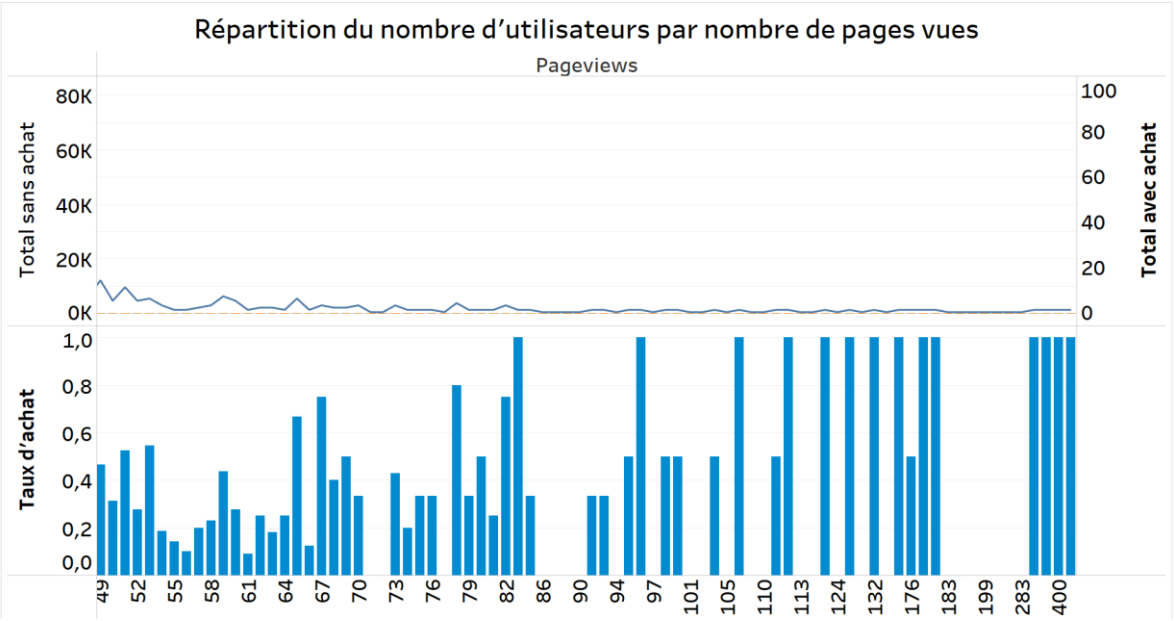
III.B. Pages vues

La variable **pageviews** compte le nombre de pages vues sur le site lors d'une seule session. Les graphiques suivants représentent, au-dessus, la répartition du nombre d'utilisateurs par nombre de pages vues sans achat (en orange) et lors d'achat (en bleu) ainsi que, en dessous, le taux de conversion nombre d'utilisateur par nombre de pages vues -> achat.

[3] Abrégé IE par la suite

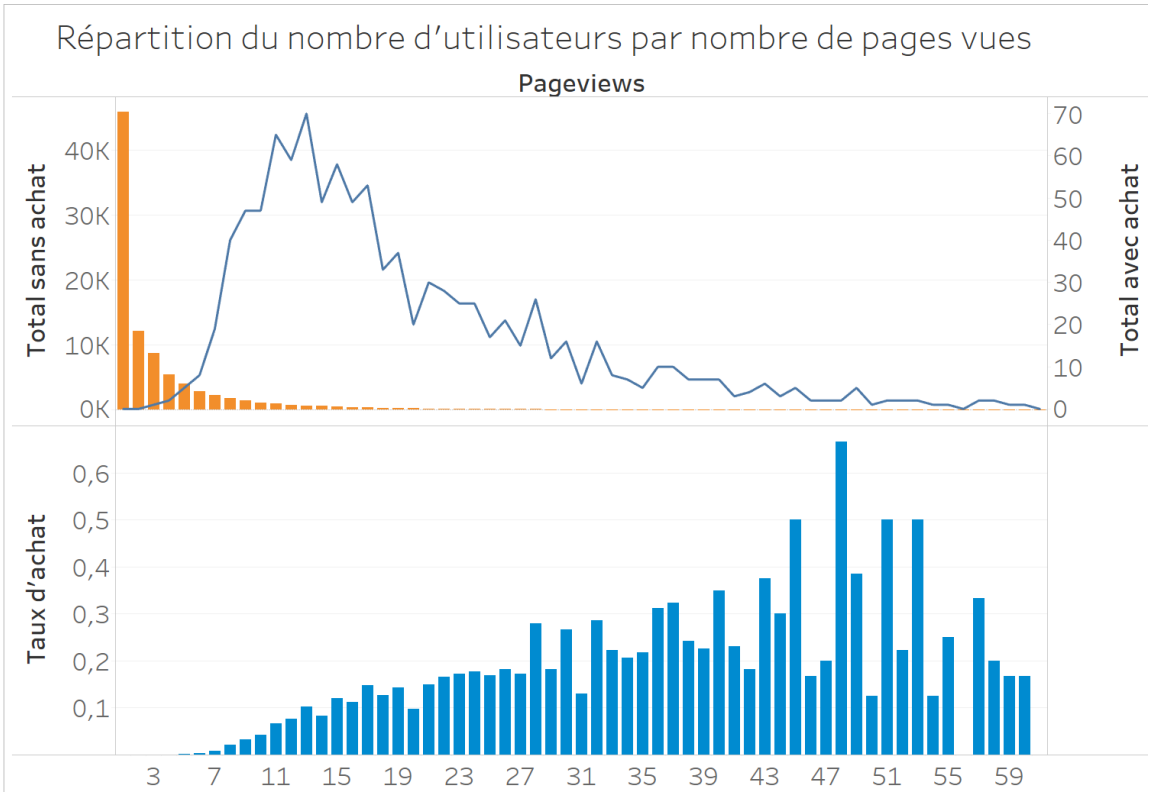


Le graphique ci-dessus ne représente pas tous les nombres de pages vues. Il s'arrête à 80 pour l'axe des abscisses par soucis de lisibilité mais s'étend jusqu'à 400.
Voici ci-dessous la deuxième partie du graphique :



On remarque qu'au-delà de 60 pages vues les effectifs ne sont plus assez importants pour être représentatifs de la réalité. Un nombre de pages vues supérieur peut être considéré comme un outlier. Je décide donc pour limiter les erreurs de supprimer de mes données les individus ayant vues plus de 60 pages sur le site lors du mois de mars.

Voici ci-dessous le graphique post-nettoyage :

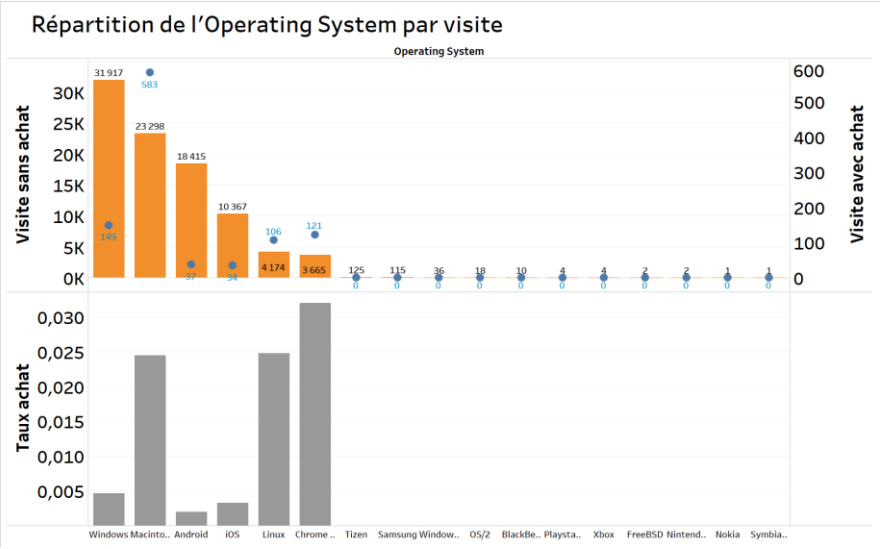


Nous remarquons que la courbe des pages vues sans achat ne fait que décroitre. La courbe des pages vues avec achat, quant à elle, croît jusqu'à 13 pages vues où elle atteint son maximum puis décroit. Je suppose alors une forte corrélation entre le nombre de pages vues et l'achat dû au fait que les deux courbes ne semblent pas suivre pas la même distribution. De plus le taux de conversion visites -> achat ne semble pas être linéaire, ce qui étaye mon hypothèse.

III.C. Système d'exploitation

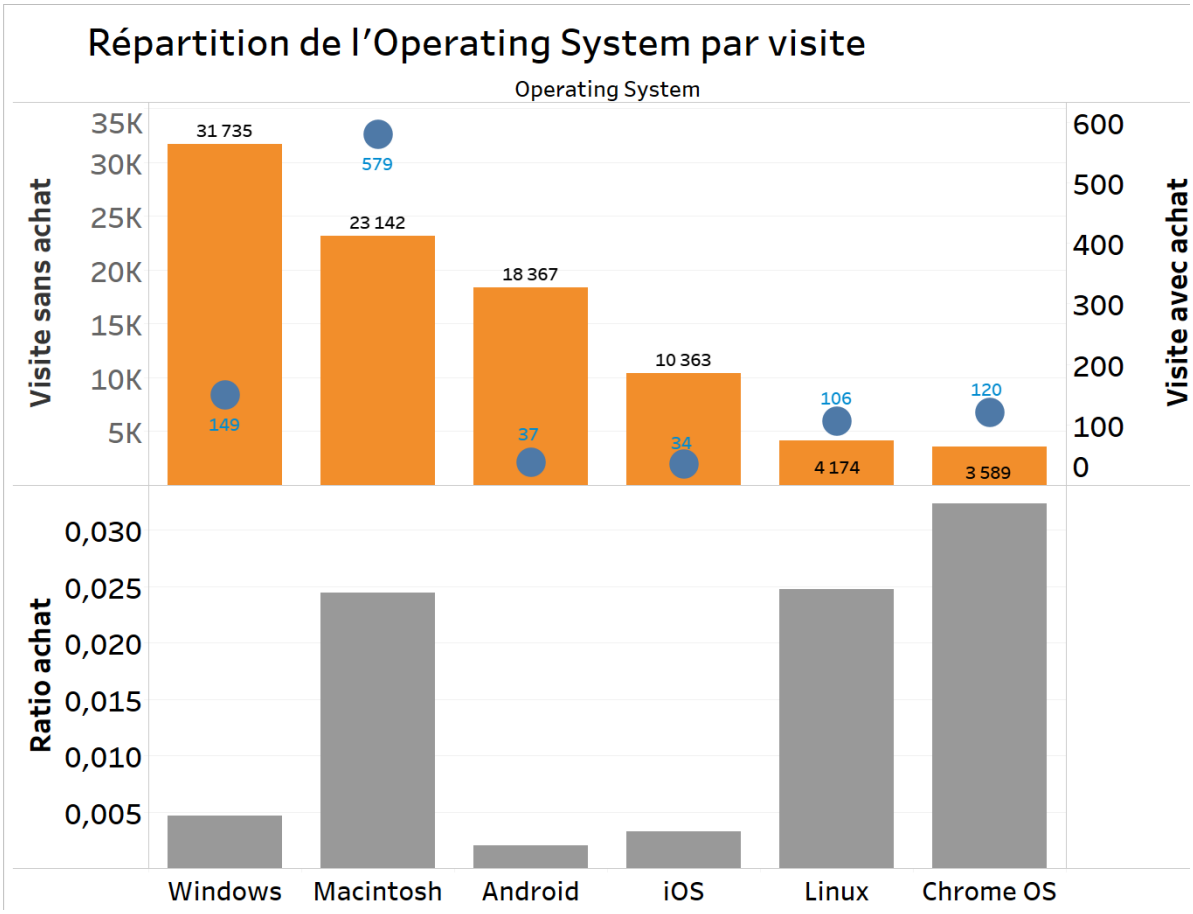
La variable **OS** désigne le système d'exploitation utilisé par le visiteur. Le graphique suivant représente, au-dessus, le nombre d'utilisateurs par système d'exploitation utilisé alors de visites n'aboutissant pas à un achat en orange et lors de visites aboutissant à un achat en bleu.

Ci-dessous est représenté le taux de conversion visite-> achat par OS.



Nous remarquons que six systèmes d'exploitation sont représentés par un grand nombre d'individus. Cependant, sept autres ne comptent que quelques individus. De plus, toutes les visites effectuées à l'aide de ces OS minoritaires se concluent par un achat. Cela n'est pas représentatif de la réalité et peut potentiellement suggérer des erreurs lors de la récolte de données. De plus ces données pourraient corrompre le futur modèle prédictif. J'ai donc décidé de les retirer.

Voici ci-dessous le graphique post-nettoyage :



Nous remarquons que le nombre d'achat effectués par **OS** n'est pas directement proportionnel au nombre de visites. En effet pour 31274 visites sur Windows seul 149 se concluent par un achat. Or pour 3589 visites sur Chrome OS 120 se concluent par un achat. Le nombre de visites infructueuses diminue de presque 90% alors que le nombre de visites fructueuses diminue seulement de 20%. Nous supposons alors une corrélation entre le navigateur et l'achat. De plus nous remarquons que les deux systèmes d'exploitation ayant le taux de conversion le plus bas sont des **OS** d'appareils mobiles. Cela renforce notre théorie de corrélation entre l'achat et le type d'appareil.

Nous avons observé l'analyse exploratoire de trois variables : **browser**, **pageviews** et **OS**. Toutes trois semblent avoir un rapport avec le comportement d'achat des usagers du site Google Merchandise Store. Voyons à présent les autres variables qui semblent être corrélées.

IV. Analyse de corrélation

Suite à l'analyse exploratoire, différentes hypothèses de corrélation ont été mises en évidence. Celles-ci suggèrent une relation entre la variable et le comportement d'achat. Cependant elles ne font que le suggérer et non pas l'affirmer. Il est essentiel de confirmer ou infirmer ces théories.

Arthur Conan Doyle disait : "C'est une erreur capitale de théoriser avant que l'on ait des données. Insensiblement on commence à tordre les faits pour suivre des théories au lieu de tordre les théories pour suivre les faits"[4].

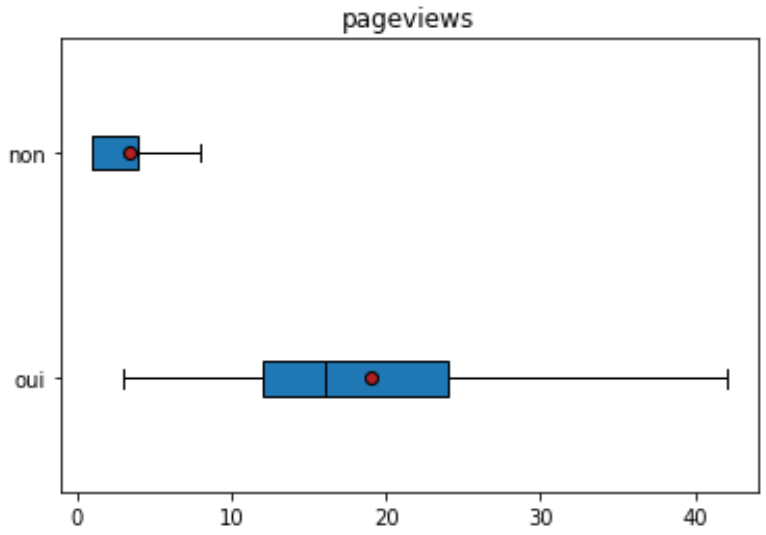
Dans notre cas il ne faut pas théoriser à partir d'intuitions plutôt que de rechercher les faits. J'ai donc effectué différents tests statistiques pour confirmer ou infirmer mes théories.

IV.A. ANOVA

Les études de corrélation suivantes sont toujours dans le but d'expliquer le comportement d'achat. Une des deux variables sera donc toujours qualitative. J'ai donc effectué des analyses de variance[5] pour toutes mes variables quantitatives.

[4] *It is a capital mistake to theorize before one has data. Insensibly one begins to twist facts to suit theories, instead of theories to suit facts. Sir Arthur Conan Doyle, Author of Sherlock Holmes stories*
[5] Analyses of Variance ou ANOVA

Voici ci-dessous par exemple l'analyse de variance de la variable vue précédemment **pageviews**



Le boxplot ci-dessus découle des graphiques montrés précédemment. Attention cependant, les valeurs avant le premier quartile et celles après le troisième quartile n'ont pas été représentées par soucis de clarté. Vous pouvez retrouver ces valeurs sur les graphiques étudiés plus tôt.

```
#Hypothèse H0 : les deux variables sont indépendantes

X = "achat"
Y= "pageviews"
calc_anova(X,Y,df_acp)

#P-value < seuil de 5%  on rejette donc l'hypothèse. Les deux variables sont donc corrélées
```

```
{'SCE': 243075.52,
 'SCT': 1831424.741,
 'SCR': 1588349.221,
 'eta_squared': 0.133,
 'Valeur_F': 10988.638012520572,
 'P-valeur': 0.0}
```

Notre hypothèse H0 suppose que les deux variables, **achat** et **pageviews**, sont indépendantes. Le test ANOVA nous donne une P-Value arrondie à 0.0. C'est inférieur au seuil de 5%, on rejette donc l'indépendance et pouvons conclure que les deux variables sont corrélées. De plus, le test nous donne un état carré de 0.133. Cette variable explique donc 13,3% du comportement d'achat des visiteurs. Cette valeur est très grande compte tenu de la complexité de ce que l'on cherche à expliquer.

La même procédure a été effectuée pour les variables suivantes :

TimeOnSite ; nbVisites ; totalPageviews ; visitNumber ; hits ; sessionQualityDim ; heure

Seule la variable heure n'est pas considérée comme corrélée à l'achat. Voici donc un tableau résumant les états carrés des différentes variables.

Variable	État carré
SessionQualityDim	0.362
totalPageviews	0.142
hits	0.134
pageviews	0.133
timeOnSite	0.064
nbVisites	0.017
visitNumber	0.004

La variable **sessionQualityDim** est celle ayant l'état carré le plus élevé. Je suppose qu'elle sera indispensable pour la création du modèle prédictif. **TotalPageviews**, **hits** et **pageviews** ont aussi un état carré important. Cependant je suspecte une forte corrélation entre-elles et suppose n'en utiliserai qu'une.

IV.B. Chi-2

Les études de corrélation suivantes auront toujours le même but, c'est-à-dire expliquer le comportement d'achat. Cette fois-ci nous étudierons les variables qualitatives. Le test du Chi-2 s'impose donc.

Voici ci-dessous un exemple de test de Chi-2 avec la variable étudiée précédemment :

```
#Hypothèse H0 : On suppose les variables achat et OS d'être indépendantes.

sous_echantillon=df_acp.copy()

X = "achat"
Y = "OS"

c = sous_echantillon[[X,Y]].pivot_table(index=X,columns=Y,aggfunc=len)
cont = c.copy()

obs=cont
chi2, p, dof, expected = st.chi2_contingency(obs)
print (chi2)
print (p)

# P value = 0%  < 5%  on rejette donc l'hypothèse. Les deux variables sont corrélées
```

```
838.3230894493942
5.914843711815437e-179
```

Notre hypothèse H0 suppose que les deux variables, **achat** et **OS**, sont indépendantes. Le test Chi-2 nous donne une P-Value arrondie à 5.91e-179. C'est très inférieur au seuil de 5%, on rejette donc l'indépendance et pouvons conclure que les deux variables sont corrélées.

Ce test a été effectué pour toutes les variables qualitatives, voici ci-dessous un tableau récapitulant les résultats.

Variable	P-Value
newVisits	1.08e-254
isTrueDirect	1.24e-218
channelGrouping	9.21e-182
OS	5.91e-179
isMobile	1.45e-45
jour	1.67e-42
Chrome	1.08e-31

Toutes ces variables ont une P-Value inférieur au seuil de 5%, elles sont donc toutes corrélées au comportement d'achat. Les variables **newVisits** et **isTrueDirect** semblent être les plus explicatives. Cependant je suppose qu'elles soient corrélées négativement entre-elles. En effet, la variable **isTrueDirect** renvoie oui si le visiteur a eu un accès direct au site, c'est-à-dire sans passer par un moteur de recherche. C'est le cas lorsque l'on tape directement l'adresse dans la barre URL du navigateur ou lorsque le site a déjà été visité par le passé et est en mémoire dans le navigateur. Or la variable **newVisits** renvoie oui si l'utilisateur vient pour la première fois sur le site. Les deux variables semblent donc avoir un lien logique entre-elles. Une seule pourra être gardée dans le modèle si la corrélation est avérée.
Nous étudierons cela si nécessaire suite à notre sélection de variables suivante.

V. Modèle prédictif

V.A. ACP

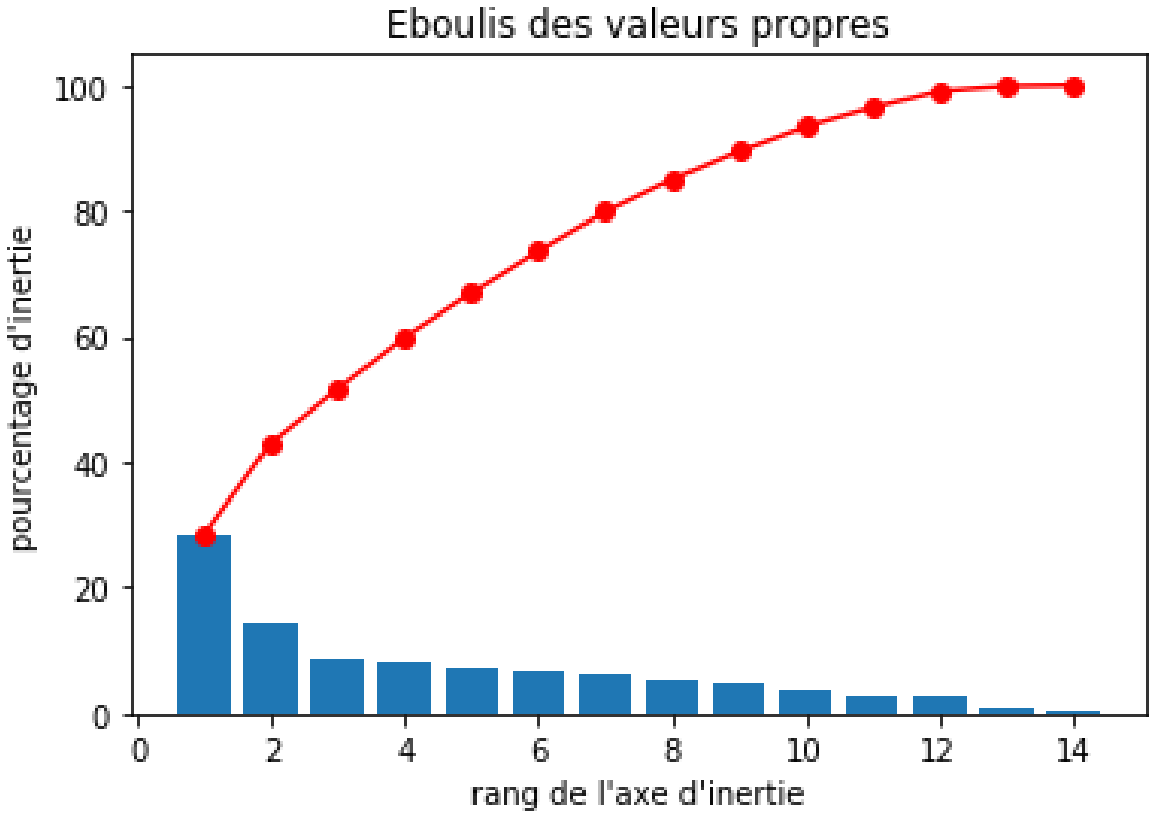
Nous avons détecté 14 variables étant plus ou moins corrélées à l'achat. Cependant, dans le but de prédire l'achat ou non nous allons utiliser une régression logistique par la suite. Or une régression logistique avec 14 paramètres sera très peu robuste et ne rendra pas compte de la réalité. Il faut donc réduire ce nombre. C'est pour cela qu'une ACP a été réalisée : réduire efficacement le nombre de dimensions étudiées.

Voici ci-dessous le code utilisé :

```
# choix du nombre de composantes à calculer
n_comp = 14

# import de l'échantillon
X = df_acp[["totalPageviews","hits","newVisits","timeOnSite","pageviews","sessionQualityDim"
std_scale = preprocessing.StandardScaler().fit(X)
X_scaled = std_scale.transform(X)

# Calcul des composantes principales
pca = decomposition.PCA(n_components= n_comp)
pca.fit(X_scaled)

# Eboulis des valeurs propres
display_scree_plot(pca)

# Cercle des corrélations
pcs = pca.components_
display_circles(pcs, n_comp, pca, [(0,1)],labels = np.array(df_acp[["totalPageviews","hits",'
display_circles(pcs, n_comp, pca, [(2,3)],labels = np.array(df_acp[["totalPageviews","hits",'

# Projection des individus
X_projected = pca.transform(X_scaled)
display_factorial_planes(X_projected, n_comp, pca, [(0,1)],illustrative_var=df_acp.achat)
display_factorial_planes(X_projected, n_comp, pca, [(2,3)],illustrative_var=df_acp.achat)
```

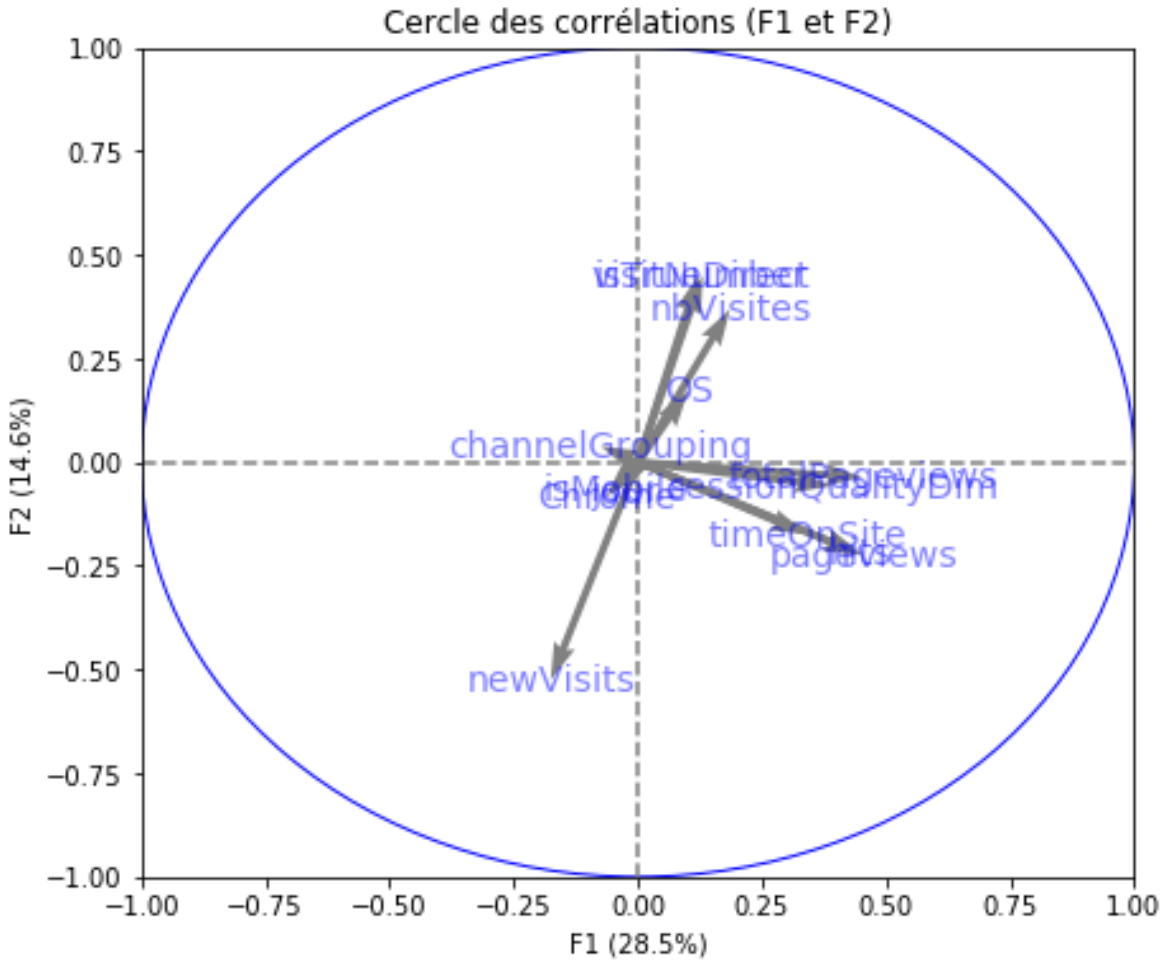
On remarque que la valeur 14 a été attribuée à la variable **n_comp**. En effet, celle-ci définit le nombre de composants utilisés lors de mon ACP, c'est à dire toutes les variables vues précédemment. Aussi la variable illustrative utilisée est **achat** étant donné que nous cherchons toujours à étudier le comportement d'achat.

Voici l'éboulis des valeurs propres :



On remarque qu'en utilisant une seule variable, le l'inertie est d'environ 30%. Par la suite l'augmentation du pourcentage d'inertie est élevée jusqu'à environ 5 composants où l'inertie atteint 70% environ.

Voici le cercle des corrélations sur le premier plan factoriel :



Celui-ci représente les variables qui représente le mieux les axes F1 et F2. Différentes variables semblent alors intéressantes à intégrer dans la régression logistique :

-**newVisits** ; **isTrueDirect** ; **nbVisites** ; **totalPageviews** ; **sessionQualityDim** ; **pageviews** ; **hits**

V.B. Corrélation entre les variables du modèle

J'ai choisi de présélectionner la variable la plus explicative de ce modèle : **sessionQualityDim**. En effet, se séparer de cette variable nous ferai perdre la plus grande partie de l'explication du comportement d'achat. J'ai donc fait des tests de corrélation entre cette variable et toutes les autres de la liste précédente.

```
#Hypothèse H0 : les deux variables sont indépendantes

X = "newVisits"
Y = "sessionQualityDim"
calc_anova(X,Y,df_acp)

#P-value < seuil de 5%  on rejette donc l'hypothèse. Les deux variables sont donc corrélées
```

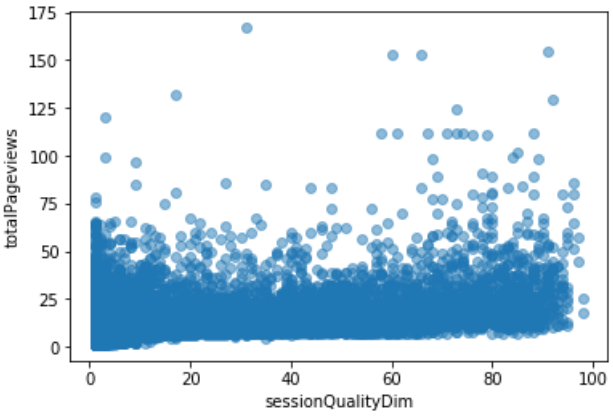
```
{'SCE': 462727.168,
 'SCT': 10763030.043,
 'SCR': 10300302.875,
 'eta_squared': 0.04299227695044328,
 'Valeur_F': 3225.5627650347856,
 'P-valeur': 0.0}
```

```
#Hypothèse H0 : Les deux variables sont indépendantes

X = "isTrueDirect"
Y = "sessionQualityDim"
calc_anova(X,Y,df_acp)

#P-value < seuil de 5%  on rejette donc l'hypothèse. Les deux variables sont donc corrélée
```

```
{'SCE': 338133.501,
 'SCT': 10763030.043,
 'SCR': 10424896.543,
 'eta_squared': 0.03141619965627126,
 'Valeur_F': 2328.8790817275267,
 'P-valeur': 0.0}
```

Les variables **newVisites** et **isTrueDirect** sont corrélés à **sessionQualityDim**. Elles sont donc exclues du model. Les quatre autres variables sont corrélées à **sessionQualityDim**. J'ai donc décidé d'inclure la plus explicative des quatre.

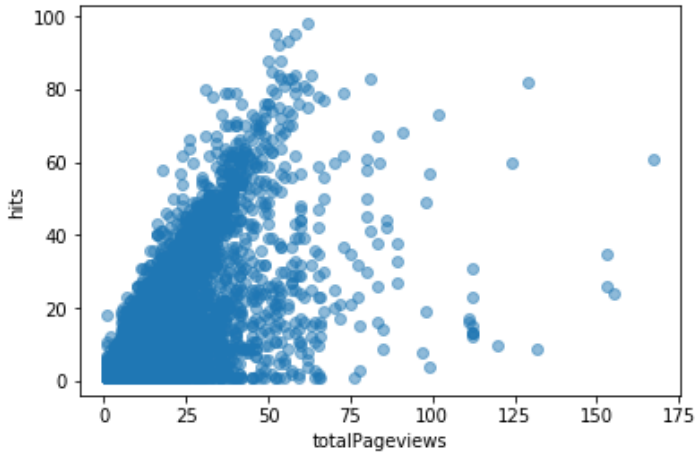
```
# TESTS  A GARDER A PARTIR D ICI

#Hypothèse H0 : On suppose les variables sessionQualityDim et totalPageviews d'être indépendante

plt.plot(df_acp.sessionQualityDim,df_acp.totalPageviews,'o',alpha=0.5)
plt.xlabel("sessionQualityDim")
plt.ylabel("totalPageviews")
plt.show()

print(st.pearsonr(df_acp.totalPageviews,-df_acp.sessionQualityDim))
print(st.spearmanr(df_acp.totalPageviews,-df_acp.sessionQualityDim))

# Le coefficient de Spearman ne permet pas d'affirmer la corrélation entre les deux variables.
```




```
(-0.6134948013844154, 0.0)
SpearmanrResult(correlation=-0.6478484935039228, pvalue=0.0)
```

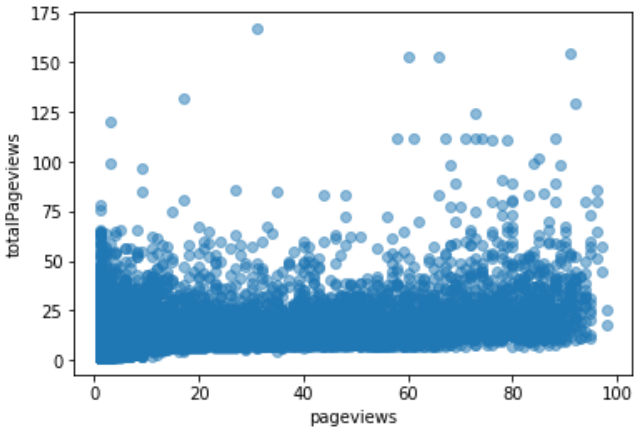
Je vais alors effectuer des tests de corrélation entre la variable **totalPageviews** et les trois restantes.

```
#Hypothèse H0 : On suppose les variables hits et totalPageviews d'être indépendantes

plt.plot(df_acp.totalPageviews,df_acp.hits,'o',alpha=0.5)
plt.xlabel("totalPageviews")
plt.ylabel("hits")
plt.show()

print(st.pearsonr(df_acp.totalPageviews,-df_acp.hits))
print(st.spearmanr(df_acp.totalPageviews,-df_acp.hits))

# Le coefficient de Spearman est proche de -1. Les deux variables sont donc corrélée
```



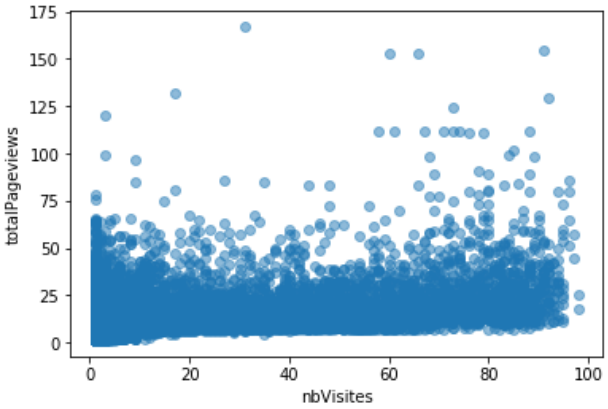

```
(-0.8139728550096743, 0.0)
SpearmanrResult(correlation=-0.915440021157226, pvalue=0.0)
```

```
#Hypothèse H0 : On suppose les variables pageviews et totalPageviews d'être indépendan

plt.plot(df_acp.sessionQualityDim,df_acp.totalPageviews,'o',alpha=0.5)
plt.xlabel("pageviews")
plt.ylabel("totalPageviews")
plt.show()

print(st.pearsonr(df_acp.totalPageviews,-df_acp.pageviews))
print(st.spearmanr(df_acp.totalPageviews,-df_acp.pageviews))

# Le coefficient de Spearman est proche de -1. Les deux variables sont donc corrélées
```




```
(-0.8300311666270838, 0.0)
SpearmanrResult(correlation=-0.9234387852154625, pvalue=0.0)
```

```
#Hypothèse H0 : On suppose les variables nbVisites et totalPageviews d'être indépendantes.

plt.plot(df_acp.sessionQualityDim,df_acp.totalPageviews,'o',alpha=0.5)
plt.xlabel("nbVisites")
plt.ylabel("totalPageviews")
plt.show()

print(st.pearsonr(df_acp.totalPageviews,-df_acp.nbVisites))
print(st.spearmanr(df_acp.totalPageviews,-df_acp.nbVisites))

# Le coefficient de Spearman ne permet pas d'affirmer la corrélation entre les deux variables.
```




```
(-0.46355502651063857, 0.0)
SpearmanrResult(correlation=-0.39377920038506625, pvalue=0.0)
```

Les variables **hits** et **pageviews** sont corrélées à **totalPageviews**. La variable **nbVisites** quant à elle sera incluse dans le modèle de régression logistique avec les variables **sessionQualityDim** et **totalPageviews**.

V.C. Régression logistique

```
X = df_acp[["sessionQualityDim","totalPageviews","nbVisites"]].values
y = df_acp["achat"].values
i=0
p=0

from random import shuffle
nb_erreur = 0

# découpe du jeu d'exemples en 80 % pour l'entraînement du modèle, le reste pour son test
p = 0.8 # 80%
listeDesIndices = [i for i in range (len (X))]
shuffle (listeDesIndices)

# la liste des indices des exemples utilisés pour l'entraînement
indices_entrainement = listeDesIndices [0:int(p*len(X))]

# la liste des indices des exemples utilisés pour le test
indices_test = listeDesIndices[int(p*len(X)):len(X)]

# on entraîne le modèle
lr.fit(X[indices_entrainement], y[indices_entrainement])

# on prédit la classe des exemples du jeu de test
classe_predite = lr.predict(X[indices_test])
classe_predite_entrainement = lr.predict(X[indices_entrainement])
lr_predict=pd.DataFrame(lr.predict_proba(X[indices_test]))
```

J’ai donc effectué une régression logistique dans le but de prédire l’achat ou non. Pour ce faire j’ai découpé mon jeu de données en deux parties :

-80% utilisés pour le jeu d’entrainement

-20% utilisés pour le jeu de test

J’ai donc entrainé l’algorithme avec le premier jeu et effectué les prédictions sur le deuxième.

```
# on calcule le nombre d'erreurs de prédiction
for i in range (len(indices_test)):
    if y[indices_test [i]] != classe_predite [i]:
        nb_erreur += 1

# et on calcule le taux d'erreur demandé
taux_erreur = nb_erreur / len (indices_test)
print("\n Le taux d'erreur est donc de : ",taux_erreur)

matrice_confusion = [[0, 0], [0, 0]]
for i in range (len (indices_test)):
    matrice_confusion [y[indices_test [i]]] [classe_predite [i]] += 1
print("\n Voici le pourcentage de non achat prédit justement :", (int(matrice_confusion[0
print("\n Voici le pourcentage d'achat prédit justement :", (int(matrice_confusion[1][1])
results_entrainement={"sessionQualityDim":pd.DataFrame(X[indices_entrainement])[0].tolist
df_results=pd.DataFrame(results_entrainement)
ax=sns.scatterplot(x="sessionQualityDim", y="totalPageviews", hue="achat",data=df_acp)
ax.set_title("Projection des 80% d'entrainement")
plt.show()

results_test={"sessionQualityDim":pd.DataFrame(X[indices_test])[0].tolist(),"totalPagevie
df_results=pd.DataFrame(results_test)
ax=sns.scatterplot(x="sessionQualityDim", y="totalPageviews", hue="achat",data=df_acp)
ax.set_title("Projection des 20% de test")
plt.show()
```

```
Le taux d'erreur est donc de :  0.011071652391894714

Voici le pourcentage de non achat prédit justement : 0.9954821403360158

Voici le pourcentage d'achat prédit justement : 0.5128205128205128
```

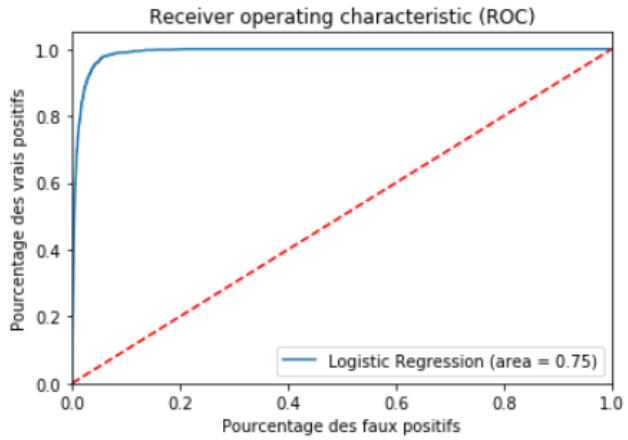
Je décide alors de calculer les pourcentages d’erreur afin de voir la solidité de mon test. On remarque que le taux d’erreur global est de 1,1%. Ce taux est plutôt bas. Cependant ce chiffre n’est pas suffisant pour conclure que la régression est très bonne ou non.

Je décide alors de regarder les résultats plus précisément. Je vois alors que dans 99,5% des cas, le fait qu’il n’y a pas d’achat est correctement prédit. Cependant, la prédiction d’achat est quant à elle correcte à seulement 51,2%. C’est-à-dire que lorsque l’algorithme prédit qu’il y a un achat, c’est faux quasiment une fois sur deux. Cependant, lorsque l’algorithme prédit qu’il n’y a pas d’achat, la régression donne la bonne réponse 199 fois sur 200. Mais pourquoi un taux de prédiction d’achat si bas ?

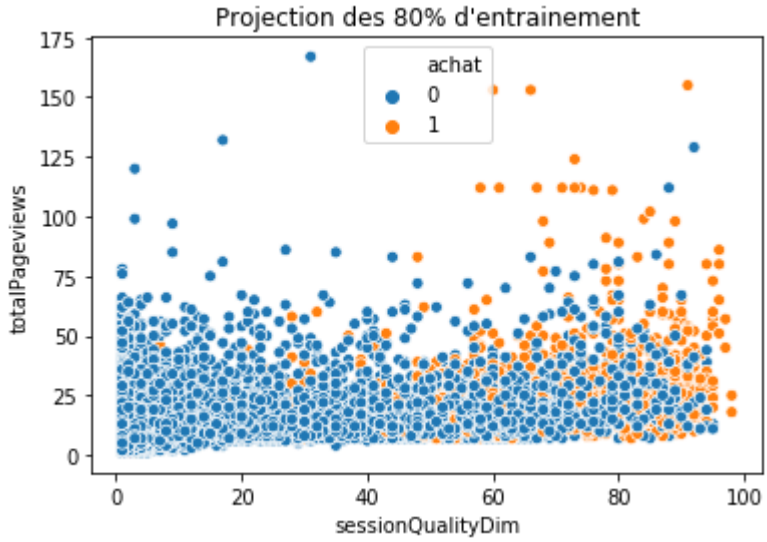
J’ai alors étudié la courbe ROC de la régression logistique. Celle-ci a une aire de 0.75. Ce n’est pas un résultat réellement satisfaisant. Mais il n’est pas médiocre non plus.

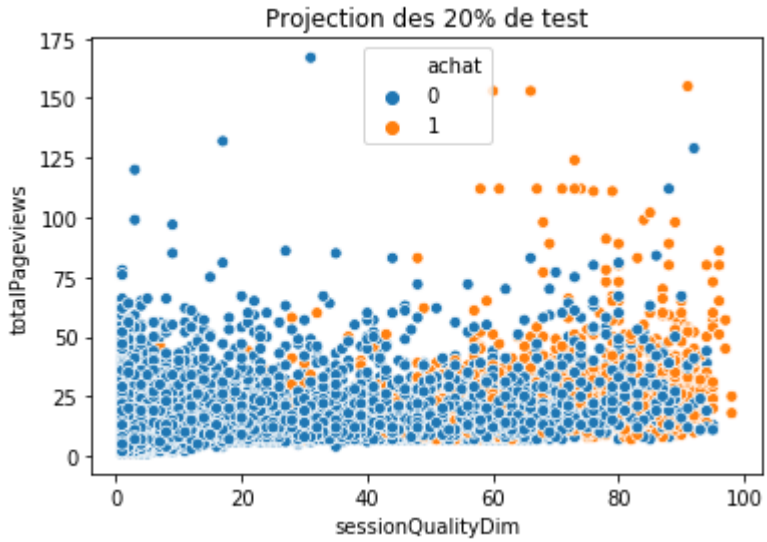
```
logit_roc_auc = roc_auc_score(y, lr.predict(X))
print(logit_roc_auc)
fpr, tpr, thresholds = roc_curve(y, lr.predict_proba(X)[:,1])
plt.figure()
plt.plot(fpr, tpr, label='Logistic Regression (area = %0.2f)' % logit_roc_auc
plt.plot([0, 1], [0, 1],'r--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('Pourcentage des faux positifs')
plt.ylabel('Pourcentage des vrais positifs')
plt.title('Receiver operating characteristic (ROC)')
plt.legend(loc="lower right")
plt.savefig('Log_ROC')
plt.show()
```

```
0.7454668011536134
```



Ci-dessous les projections des différents individus en fonction des deux variables les plus représentatives : **sessionQualityDim** et **totalPageviews**.





On remarque que les individus aboutissants à un achat sont regroupés vers le côté haut gauche du graphique. Cependant, beaucoup de non achat sont confondus avec les achats. Il est donc compréhensible que notre algorithme ne puisse les différencier. Certains individus qui n'achètent pas ont un comportement identique à ceux effectuant un achat.

Notre modèle n'est donc pas complet. En effet, nous étudions le comportement d'humains. Or étudier les humains est très complexe. Nous ne disposions pas de suffisamment de variables et de facteurs au cours de ce projet pour décrire avec exactitude le comportement d'humains.

VI. Conclusion

L'objectif du projet était de détecter au mieux les variables expliquant le comportement d'achat ainsi que de créer le modèle prédictif lié. Cet objectif n'a cependant pas été totalement rempli. Le modèle développé n'explique que partiellement notre objectif. Toutes les variables qui y sont corrélées ne sont peut-être pas utilisées. La restriction matérielle est certainement aussi un frein à l'algorithme étant donné qu'il nous prive de toute analyse ou modélisation de série temporelle. Peut-être que la plupart des individus détectés comme ayant fait un achat le feront dans les mois suivants ? De plus, un comportement lié à l'humain n'est pas résumable en une vingtaine de variable. L'humain dans son individualité est certainement l'une des choses les plus difficiles à prévoir.

Cependant, il est certain qu'avec les données en ma possession, un meilleur modèle est réalisable. Ronald Coase a dit : « Si vous torturez les données assez longtemps elles se confesseront »[6]. La formation Data Analyst m'a donné les armes pour traiter, corriger, interroger, explorer, et réaliser des modèles prédictifs. Cependant c'est ce dernier point que je souhaite approfondir. Les modèles neuronaux me semblent passionnants. Les concepts liés, d'une incroyable ingéniosité. Et les modèles prédictifs qui en découlent, d'une puissance et d'une précision effrayante. La formation Data Scientist m'aidera certainement à développer les compétences que je recherche.

[6] « If you torture the data long enough, i twill confess », Ronald Coase

Lien des images utilisées :

- https://www.kaggle.com/arunsankar/kaggle-logo

- https://sospc.name/10-soucis-pannes-frequents-ordinateur/