RAPPORT DU PROJET IA MASTER 1 BIG DATA ET MANAGEMENT

« Comprendre le marché de l'industries des taxis avant d'investir »

Projet réalisé par :

Poroni KOINZI

(Data Scientist at IA School)

https://www.linkedin.com/in/poroni-koinzi/

https://github.com/PoroniF

Elvis SOUNNA

(NLP training at Data Glacier)

https://www.linkedin.com/in/elvis-sounna

https://github.com/elvissounna

Projet encadré par :

Pôle Académique IA de l'école de l'Intelligence Artificielle

SOMMAIRE

I. INTRODUCTION

En raison de la croissance remarquable de l'industrie des taxis au cours des dernières années et de plusieurs acteurs clés du marché, d'énormes investissements sont prévus et, conformément à sa stratégie Go-to-Market (plan d'action d'entreprise, visant à procéder avec succès au lancement d'un produit ou service. Sa démarche consiste à mobiliser les ressources internes et externes d'une entreprise pour augmenter le taux de réussite de l'opération), ils souhaitent comprendre le marché avant de prendre décision finale. Les investisseurs souhaitent utiliser vos informations afin d'identifier la bonne entreprise pour réaliser leur investissement.

Base de données 4 ensembles de données individuelles ont été fournies. La période d'enregistrement va du 31/01/2016 au 31/12/2018. Vous trouverez ci-dessous la liste des ensembles de données fournies pour l'analyse :

Cab_Data.csv - ce fichier contient les détails de la transaction pour 2 compagnies de taxi

Customer_ID.csv - il s'agit d'une table de mappage qui contient un identifiant unique reliant les détails démographiques du client

Transaction_ID.csv - il s'agit d'une table de mappage qui contient le mappage de la transaction au client et le mode de paiement

City.csv - ce fichier contient la liste des villes américaines, leur population et le nombre d'utilisateurs de taxi

Pour ce faire, nous devons au cours de notre étude répondre à une série de questions :

- Y a-t-il une saisonnalité ?
- La demande de l'industrie du taxi a-t-elle tendance à augmenter avec le temps ?
- Le pourcentage de trajets rentables change selon la ville.
- Comment la demande varie selon l'âge ?
- Tarifs fidélité.

- Le mode de paiement fluctue-t-il d'une année à l'autre ? En fonction de l'âge ? Citywise?

II. RÉCUPÉRATION DES DONNÉES PAR SCRAPING

Le scraping est le processus automatisé de récupération (ou de grattage) des données d'un site Web ou d'une plateforme. La première étape pour un data scientist est de récolter les données.

```
from bs4 import BeautifulSoup
from urllib.request import urlopen
import mechanicalsoup
from urllib.request import urlretrieve

ctx = ssl.create_default_context()
ctx.check_hostname = False
ctx.verify_mode = ssl.CERT_NONE
url = "https://archive.ics.uci.edu/ml/datasets.php"
html = urllib.request.urlopen(url, context=ctx.read())
soup = BeautifulSoup(html, 'html.parser')
html = soup.prettify('UTF-8')
```

```
url_down= "https://archive.ics.uci.edu/ml/machine-learning-databases/00192/"
urlretrieve(url_down, "Cab_data.csv")
url_down= "https://archive.ics.uci.edu/ml/machine-learning-databases/00196/"
urlretrieve(url_down, "City.csv")
url_down= "https://archive.ics.uci.edu/ml/machine-learning-databases/00206/"
urlretrieve(url_down, "Customer.csv")
url_down= "https://archive.ics.uci.edu/ml/machine-learning-databases/00314/"
urlretrieve(url_down, "Transaction.csv")
```

Après la récolte des données, passons au traitement et analyse.

III. IMPORTATION ET VISUALISATION DES DONNÉES

La première tâche consiste à importer les bibliothèques qui nous serviront durant notre projet. Pour ce faire, il est strictement nécessaire d'avoir des compétences en data science et analyse afin de pouvoir anticiper les étapes à travers lesquelles nous effectuerons le traitement de nos données.

Les librairies sont en fait importées dans l'ordre croissant de l'exécution des tâches.

```python
# I imported following python libraries to utilize in EDA process.
import os
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
import plotly.express as px
import plotly.graph_objs as go
import plotly
import plotly.graph_objects as go
import datetime
from datetime import datetime, timedelta
#os.chdir("/content/drive/MyDrive/DataSets/DataSets")
os.chdir("./DataSets_Week_2-main")
```

Une fois nos librairies importées, nous procédons à la visualisation de chaque fichier csv (comma-separated values) afin de prendre note premièrement de la taille de tout un chacun, mais aussi des informations primaires importantes et capitales telles que le type de valeurs. Pour cela, nous avons procédé comme le montrent les images suivantes :

```python
missing_value = 0
for column in cab.columns :
    missing_value += cab[column].isna().sum() + cab[column].isnull().sum()
print('Notre Dataset a {} valeurs , {} variables et {} valeurs manquantes'.format(cab.shape[0] ,cab.shape[1] ,missing_value))
print("\nFeature's datatypes\n\n{}".format(cab.dtypes))
cab.head(10)
```

```
Notre Dataset a 359392 valeurs , 7 variables et 0 valeurs manquantes

Feature's datatypes

Transaction ID      int64
Date of Travel      int64
Company            object
City               object
KM Travelled      float64
Price Charged     float64
Cost of Trip      float64
dtype: object
```

	Transaction ID	Date of Travel	Company	City	KM Travelled	Price Charged	Cost of Trip
0	10000011	42377	Pink Cab	ATLANTA GA	30.45	370.95	313.635
1	10000012	42375	Pink Cab	ATLANTA GA	28.62	358.52	334.854
2	10000013	42371	Pink Cab	ATLANTA GA	9.04	125.20	97.632

On note bien que le premier dataset n'a aucune valeur manquante et à partir de la boucle on peut obtenir quelques informations basiques.
Nous procédons donc ainsi pour les 3 autres datasets restants.

```
city = pd.read_csv("City.csv")
print('Cab dataset has {} entries , {} features'.format(city.shape[0] ,city.shape[1]
city.head()
```

```
Cab dataset has 20 entries , 3 features
```

	City	Population	Users
0	NEW YORK NY	8,405,837	302,149
1	CHICAGO IL	1,955,130	164,468
2	LOS ANGELES CA	1,595,037	144,132

```
customer = pd.read_csv("Customer_ID.csv")
print('Cab dataset has {} entries , {} features'.format(customer.shape[0] ,customer.
customer.head()
#print("\n \n{}".format(customer.dtypes))
```

```
Cab dataset has 49171 entries , 4 features
```

	Customer ID	Gender	Age	Income (USD/Month)
0	29290	Male	28	10813
1	27703	Male	27	9237
2	28712	Male	53	11242

```
transaction = pd.read_csv("Transaction_ID.csv")
print('Cab dataset has {} entries , {} features'.format(transaction.shape[0] ,transa
transaction.head()
```

```
Cab dataset has 440098 entries , 3 features
```

	Transaction ID	Customer ID	Payment_Mode
0	10000011	29290	Card
1	10000012	27703	Card
2	10000013	28712	Cash

IV. ANALYSE ET NETTOYAGE DES DONNÉES

Le volet Analyse et nettoyage des données occupe un très grand pourcentage dans la répartition du temps de travail d'un Data Scientist. Cela consiste en quelque sorte à vérifier la présence des valeurs manquantes et se décider sur leur devenir (les supprimer ou les remplacer par la médiane par exemple en fonction du type de travail qu'il y aura à faire), renommer les variables s'il le faut, affecter à chaque variable le type qui lui sera opportun pour l'analyse statistique.

En nous rappelant des observations faites plus haut, nous avons noté l'absence des valeurs manquantes, ce qui est déjà positif ; mais nous avons été intrigués par la variable "Date of

Travel" qui était en format <int64>. Alors nous commencerons par modifier ce format en <datetime>.

```
def to_date_format(n):
    date_str =(datetime(1899,12,30) + timedelta(n-1)).strftime("%d-%m-%Y")
    date_date =  datetime.strptime(date_str, "%d-%m-%Y")
    return date_date
```

```
cab['Date of Travel']=cab['Date of Travel'].apply(lambda x:to_date_format(x))
```

```
cab.head()
```

	Transaction ID	Date of Travel	Company	City	KM Travelled	Price Charged	Cost of Trip
0	10000011	08-01-2016	Pink Cab	ATLANTA GA	30.45	370.95	313.635
1	10000012	06-01-2016	Pink Cab	ATLANTA GA	28.62	358.52	334.854

```
# revisualisons le type de la variable "Date of Travel"
print(cab['Date of Travel'].dtypes)
```

```
datetime64[ns]
```

En observant aussi le dataset City.csv, on a remarqué la présence des virgules entre les chiffres. Donc on va procéder à la modification des variables concernées.

```
# On remarque la présence des virgules entre les valeurs des variables Population et
# Pour cela nous allons les transormer en float afin de ne pas avoir d'ennuie lors d
city['Population'] = [x.replace(',','') for x in city['Population']]
city['Users'] = [x.replace(',','') for x in city['Users']]
#city['Population'] = city['Population'].astype(float)
#city['Users'] = city['Users'].astype(float)
city.head(10)
```

	City	Population	Users
0	NEW YORK NY	8405837	302149
1	CHICAGO IL	1955130	164468
2	LOS ANGELES CA	1595037	144132

Nous pouvons donc procéder à la fusion des 4 datasets afin d'en avoir un seul compact. Pour ce faire, la fonction <merge> est très efficace et nous a accompagnés dans cette tâche.

```
principal= cab.merge(transaction, on="Transaction ID").merge(city, on= "City").merge

print("Le Dataframe principal  {} enrégistrements, {} colonnes, {} valeurs manquante

principal.dtypes
```

```
Le Dataframe principal  359392 enrégistrements, 14 colonnes, 0 valeurs manquantes
Transaction ID          int64
Date of Travel         object
Company                object
City                   object
KM Travelled          float64
Price Charged         float64
Cost of Trip          float64
Customer ID             int64
Payment_Mode           object
Population             object
Users                  object
Gender                 object
Age                     int64
Income (USD/Month)      int64
dtype: object
```

Une dernière étape de nettoyage est donc nécessaire afin de s'assurer que toutes les variables de notre grand dataset soient écrites de façon plus simple (sans les espaces) et ensuite que les types de toutes les variables soient conformes à l'étape suivante qui sera l'étude statistique.
On commence donc par remplacer les espaces dans les noms des variables par des underscores,

```
for column in principal.columns:
    if ' ' in column:
        principal = principal.rename(columns={column:column.replace(' ','_')})
```

Ensuite, convertir les types de variables.

```
principal['Date_of_Travel'] = pd.to_datetime(principal['Date_of_Travel'])

for column in ["Company", "City" , "Payment_Mode" , "Gender" ] :

        principal[column] = principal[column].astype('category')


for column in ["Population", "Users" ] :

        principal[column] = principal[column].astype('int64')
```

```
principal.head(10)
```

	Transaction_ID	Date_of_Travel	Company	City	KM_Travelled	Price_Charged	Cost_of_Tri
0	10136329	2017-01-01	Yellow Cab	WASHINGTON DC	23.28	522.73	284.947
1	10253016	2017-01-11	Yellow Cab	WASHINGTON DC	44.80	813.70	607.488

```
principal.dtypes
```

```
Transaction_ID                int64
Date_of_Travel       datetime64[ns]
Company                    category
City                       category
KM_Travelled                float64
Price_Charged               float64
Cost_of_Trip                float64
Customer_ID                   int64
Payment_Mode               category
Population                    int64
Users                         int64
Gender                     category
Age                           int64
Income_(USD/Month)            int64
dtype: object
```

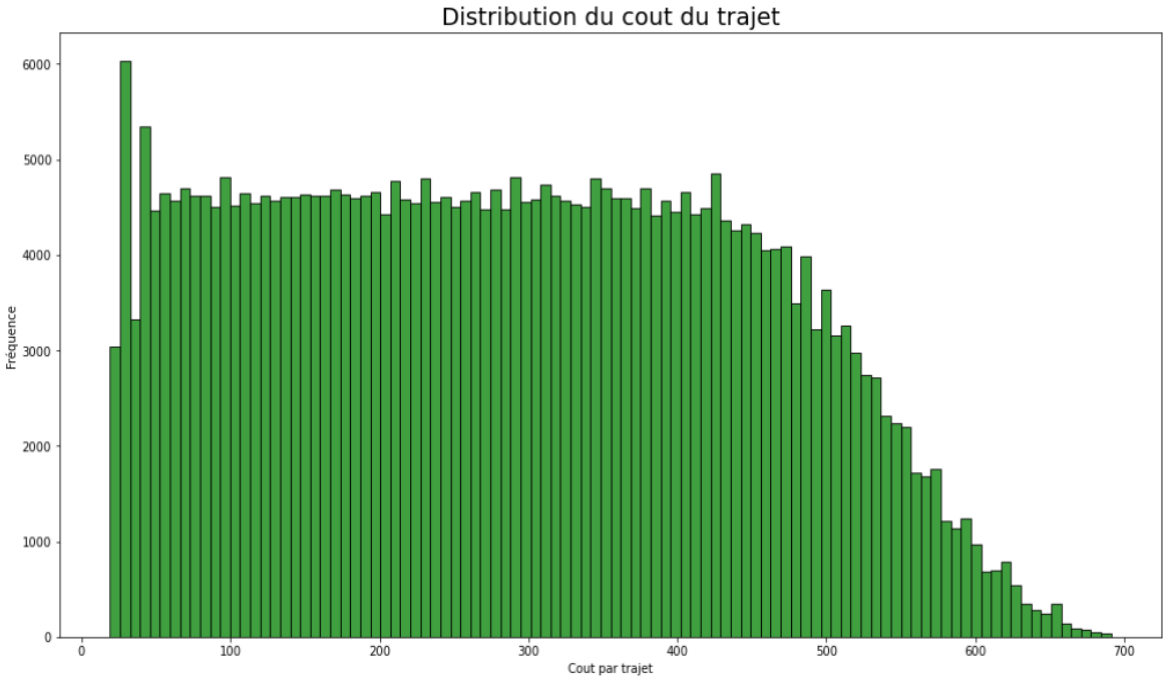
Une fois tout fait, on peut passer à l'étape suivante qui consistera aux analyses statistiques et les visualisations.

V. ETUDES ET ANALYSES STATISTIQUES

a. VISUALISATION STATISTIQUE GENERALE

Nous nous concentrerons plus sur les différentes thématiques : Distribution, Valeurs aberrantes, Corrélation, Covariance…
Commençons par une visualisation graphique de la variation globale du coût du trajet.

```
plt.figure(figsize=(17,9))
sns.histplot(data= principal,x="Cost_of_Trip",bins=100 , color= 'green')
plt.title('Distribution du cout du trajet', fontsize=20)
plt.ylabel('Fréquence')
plt.xlabel('Cout par trajet')
```



En observant ce graphique, on ne peut ressortir que les trajets à coût très élevés (supérieur à 600) sont de très faible fréquence.
Afin de mener à bon port notre étude, nous devons aller plus en profondeur. Nous commençons donc par regrouper le dataset en fonction du type d'entreprise ("Company").

```
groups = principal.groupby(principal.Company)
pink = groups.get_group("Pink Cab")
yellow = groups.get_group("Yellow Cab")
```

On a ainsi le dataset pink d'un côté, et yellow de l'autre côté. En effectuant simultanément les visualisations primaires de chacun des deux datasets, on note une similitude générale, c'est-à-dire, les valeurs des variables qui sont communes aux deux types d'entreprises sont égales. Mais en étant un peu plus curieux, on peut noter déjà des différences, par exemple en observant les déviations standard, il est très facile de se rendre compte que les valeurs sont différentes. C'est en fait en se basant sur ces différences que nous saurons répondre à la problématique principale et ainsi orienter dans le bon sens les investissements.

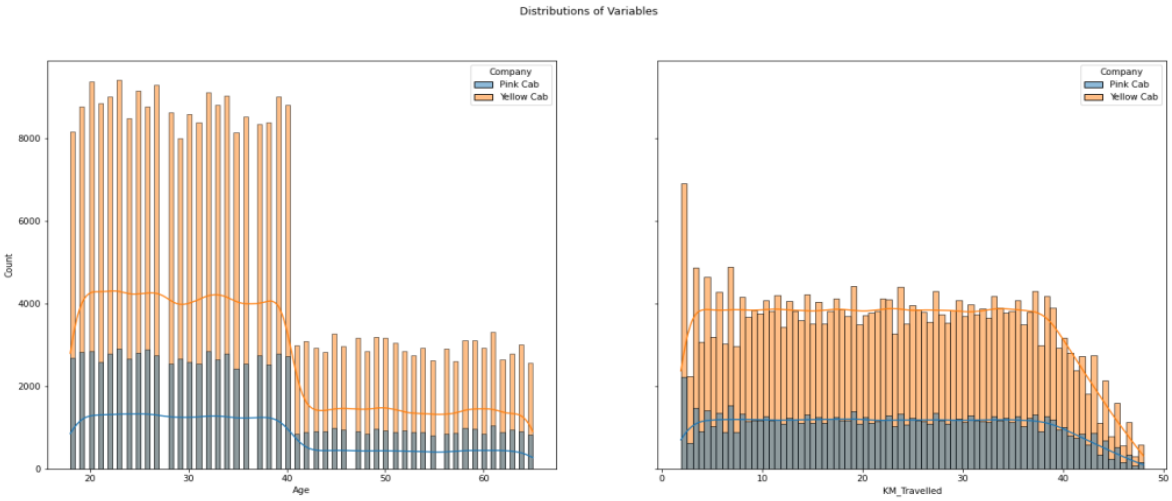
```
pink.describe()
```

	Transaction_ID	KM_Travelled	Price_Charged	Cost_of_Trip	Customer_ID	Population	
count	8.471100e+04	84711.000000	84711.000000	84711.000000	84711.000000	8.471100e+04	847
mean	1.022394e+07	22.559917	310.800856	248.148682	18422.581577	2.350642e+06	1255
std	1.261782e+05	12.231092	181.995661	135.403345	18084.830799	2.734890e+06	945
min	1.000001e+07	1.900000	15.600000	19.000000	1.000000	2.489680e+05	36
25%	1.011014e+07	12.000000	159.970000	131.868000	5317.500000	8.148850e+05	272

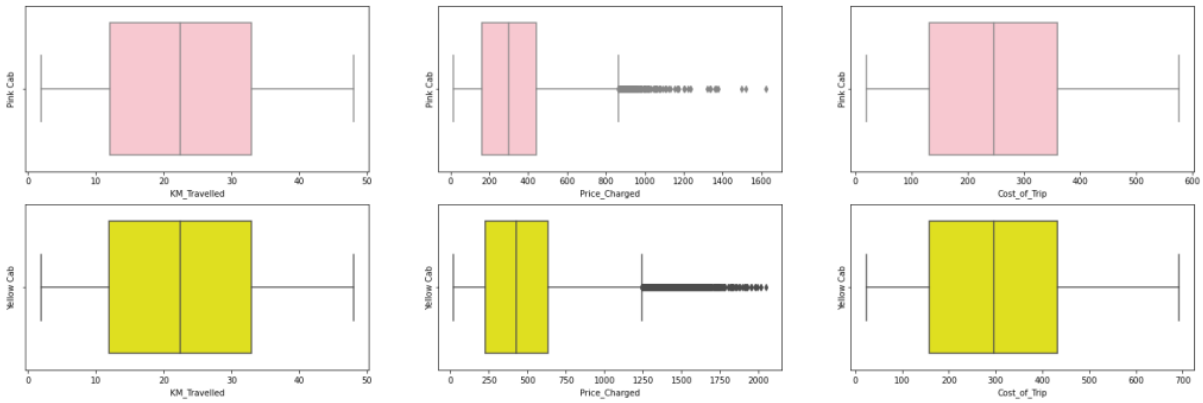
```
yellow.describe()
```

	Transaction_ID	KM_Travelled	Price_Charged	Cost_of_Trip	Customer_ID	Population	
count	2.746810e+05	274681.000000	274681.000000	274681.000000	274681.000000	2.746810e+05	2
mean	1.021978e+07	22.569517	458.181990	297.922004	19428.831732	3.373228e+06	1
std	1.269829e+05	12.234298	288.386166	162.548986	21830.791423	3.439014e+06	1
min	1.000038e+07	1.900000	20.730000	22.800000	1.000000	2.489680e+05	
25%	1.011084e+07	11.990000	226.680000	158.400000	2403.000000	6.712380e+05	

Une méthodologie performante de visualisation et d'estimation des densités de probabilité des variables aléatoires est le KDE (Kernel Density Estimation). Nous allons l'utiliser ici afin d'apprécier les différences de variations de densité de chaque entreprise.



En observant ces deux graphiques représentatifs de la variation de densité de l'âge et des distances pour chacune des compagnies, on se rend compte que la yellow a une distribution beaucoup plus statique et moins oscillante que Pink, ce qui traduit donc un certain équilibre. L'analyse des boxplots relève des informations pertinentes.

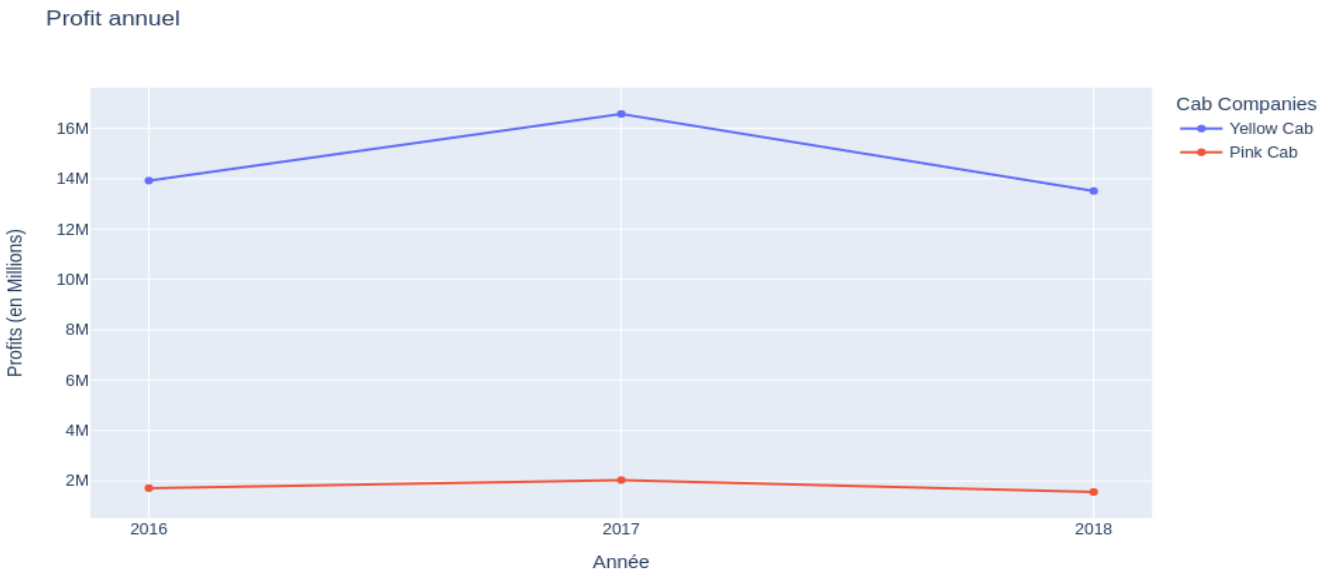


- La distribution des kilomètres parcourus est identique pour les deux entreprises
- On note la présence des outliers dans la variable "Price_Charged", on ne tiendra pas compte de cela car ce n'est pas une variable déterminante pour notre analyse.

- Au niveau de “Cost_of_Trip”, on se rend compte que Yellow a des coûts unitaires allant au-delà de 400$ tandis que Pink a un maximum d’environ 360$, les deux ayant donc une médiane assez équilibrée

b. PROFIT ANNUEL ET MENSUEL

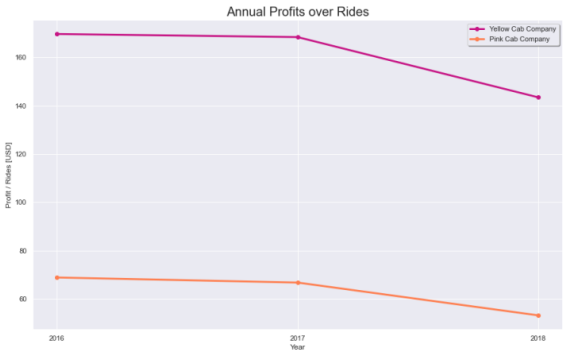
Le graphique suivant nous permet de visualiser la variation du profit annuel des deux entreprises.

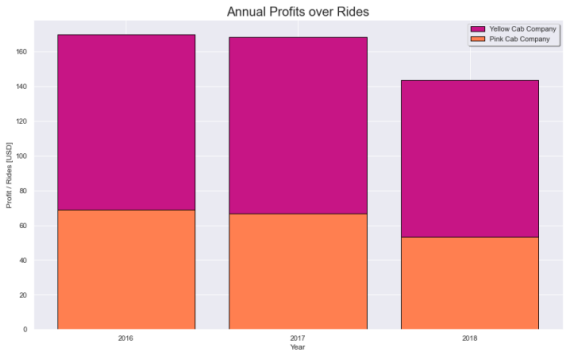


En l’observant, il nous vient presque à l’idée de se dire que les deux ont une variation très similaire, mais Yellow, qui est tout en dessus, subit une forte décroissance de 2017 à 2018. Mais nous ne pouvons pas tirer cette conclusion sans avoir calculé le coefficient directeur de chacune des droites afin de déterminer celui qui a une chute plus brutale.
Si on décide de voir la variation mensuelle du bénéfice net, on obtiendra un graphique tel que :

On remarque que Yellow a des fluctuations importantes ce qui peut se traduire par un marché très mouvementé et donc la demande pourrait être aussi importante.
Allons maintenant vérifier l’efficacité de chacune des entreprises. Le profit par trajet est un indicateur qui mesure l'efficacité de l'entreprise en termes de coûts opérationnels.

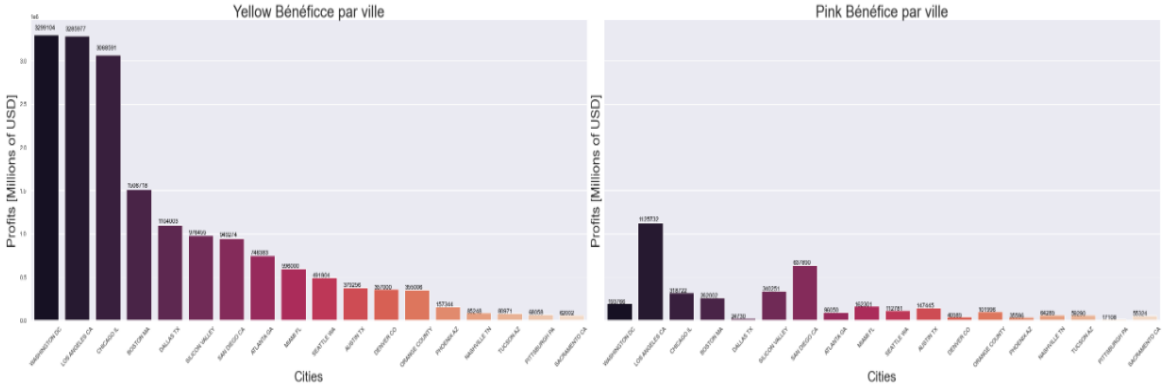
Bénéfice par trajet = (Bénéfices totaux sur une certaine période de temps) / (Nº de trajets sur cette période de temps). Graphiquement on a obtenu un résultat suivant:





On remarque donc que les deux sociétés présentent le même comportement, leur efficacité tend à diminuer avec le temps.

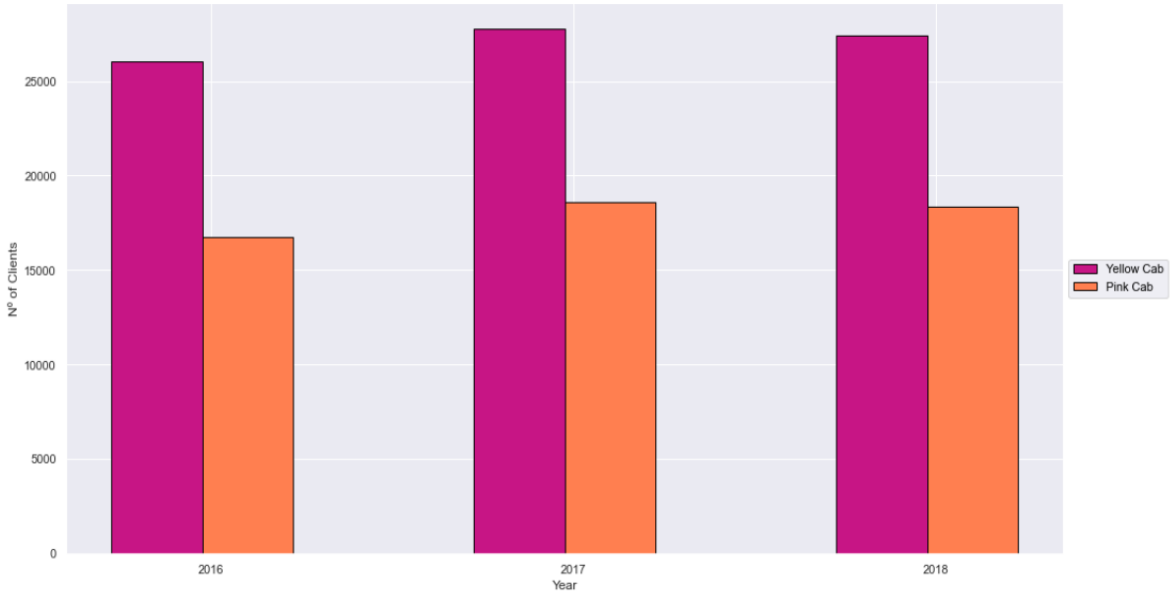
Un autre indicateur de mesure serait le bénéfice net réalisé par chacune des entreprises dans chaque ville. Le graphique ci-dessous illustre les résultats obtenus.



On remarque que dans les mêmes villes, Yellow a un bénéfice beaucoup plus élevé respectivement à son concurrent direct.

Allons maintenant regarder la fidélisation des clients de chaque entreprise par année. Pour ce faire, on peut utiliser un code qui regroupe l'ID de chaque client et additionne son nombre de voyages sur une année. En faisant ainsi on aura une idée de la compagnie la plus attractive et donc à qui les clients font le plus recours.

Clients ayant plus de 10 voyages par ans/ entreprise



VI. CONCLUSION

Nous avons évalué les deux compagnies de taxis sur les points suivants :

1. Analyse des bénéfices
Bénéfices : des bénéfices plus élevés au fil du temps et moins de fluctuations mensuelles
Bénéfices sur les trajets : meilleur taux de profit / trajet dans le temps. Yellow Cab a 2,5 bénéfices supplémentaires par rapport au trajet
Bénéfices Citywise : Yellow Cab a une plus grande part de marché dans chaque ville.
Manèges rentables : en supposant que sur un taux de 80 %, l'entreprise fonctionne bien. Yellow Cab a une haute performance parmi toutes les villes. En revanche,
Pink Cab n'est pas performant dans 8 villes.

2. Analyse de la demande
Demande : Yellow Cab a plus que triplé la demande de Pink Cab Company

3. Analyse des clients
Taux de fidélité : En supposant que les deux classes, Yellow Cab a un taux de fidélité élevé et un taux de fidélité moyen plus élevé.
Répartition des modes de paiement : les deux sociétés présentent la même répartition des modes de paiement dans le temps, par ville et par âge.

Donc sur la base des points ci-dessus, nous recommandons formellement **Yellow** pour un investissement qui serait rentable.