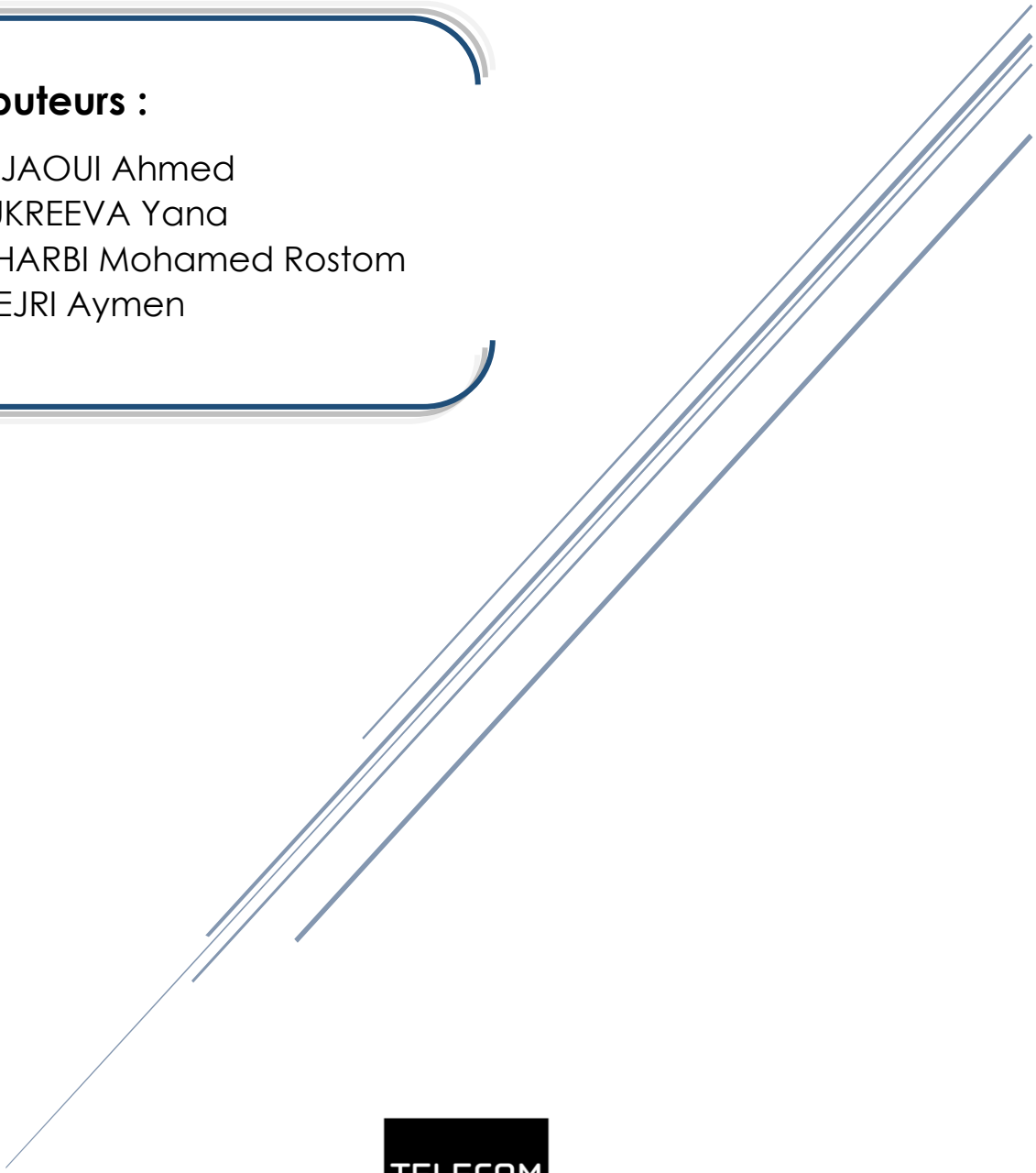
RAPPORT DE BEARING POINT'S CHALLENGE

Contributeurs :

- BEJAOUI Ahmed
- BUKREEVA Yana
- GHARBI Mohamed Rostom
- MEJRI Aymen



TELECOM ParisTech

Une école de l'IMT

Telecom ParisTech

Table des matières

I. Introduction

L'une des principales préoccupations des magasins dans le monde, c'est de prédire si un produit sera retourné par l'acheteur ou non.

Une entreprise de commerce électronique vend des chaussures et a un taux de retour élevé de ses produits, plus de 20%. Ce grand nombre de retours et d'échanges a un impact négatif sur leur marge. Pour remédier à ce problème, l'entreprise veut mieux comprendre ce phénomène et disposer d'outils pour quantifier la probabilité de retour d'un produit donné.

Pour cela, nous avons à notre disposition leur base de données de commandes réalisées entre octobre 2011 et octobre 2015.

L'objectif de ce projet est de :

Identifier les conditions qui favorisent le retour du produit (par exemple, quel type de produit est généralement retourné, quel client est le plus intéressé par le retour d'un produit, quel type de commande ou contexte d'achat conduit le plus souvent à des retours ?)

Construire un modèle de prévision de retour pour chaque produit à partir d'un panier.

Pour cela, nous avons travaillé avec les différents donner qu'on a, et nous avons essayé plusieurs modèles afin de prédire le taux de retour des produits.

Les différents modèles avaient des temps d'exécutions différents et des scores de performances différents également.

Dans ce rapport, nous détaillons notre démarche de travail, les différentes variables explicatives et nos critères de choix de ces dernières, ainsi que les différents modèles que nous avons utilisés et leurs différents résultats.

II. Chargement et exploration des données

2.1 Chargement des bibliothèques requises :

Ci-dessous sont les différentes bibliothéques dont nous avons eu besoin pour faire notre prédiction.

```
from __future__ import division
import os
from sklearn.model_selection import train_test_split
import numpy as np
import pandas as pd
import random
from sklearn.metrics import roc_auc_score
from sklearn.linear_model import LogisticRegression
from sklearn import model_selection
from catboost import CatBoostClassifier
from sklearn.preprocessing import Imputer
```

2.2 Présentation des différentes tables et variables explicatives :

2.2.1 Training data :

Il aura N = 1067290 lignes de commandes dans le training dataset. Pour chaque commande, l'ensemble des traning set signale si la commande a été renvoyée (ReturnQuantityBin) et la quantité renvoyée (ReturnQuantity). La colonne à cibler (ReturnQuantityBin) qui est une colonne binaire (y = 1, si retourné et y = 0 sinon).

2.2.2 Test data :

Le training dataset contient N = 800468 lignes de commandes. Tout le reste est similaire aux données d'entraînement.

2.2.3 Descirption des tables :

Les différentes tables à utiliser qui sont mis à notre disposition sont les tables suivantes:

a. **Order :** qui correspond à la table des commandes du commerçant pour une période donnée. Chaque ligne représente un produit commandé qui est identifié à travers un identifiant qui est le OrderNumber. L'ordre du produit dans le panier est référencé à travers LineItem et sa quantité est indiquée dans la colonne "Quantity".

Nom des colonnes	le nombre de valeurs non nulles	Type
OrderNumber	1067290	int64
VariantId	1067290	int64
LineItem	1067290	int64
CustomerId	1067290	int64
OrderStatusLabel	1067290	object
OrderTypelabel	1067290	object
SeasonLabel	1067290	object
PayementModeLabel	1067290	object
CustomerTypeLabel	1067290	object
IsoCode	1067290	object
DeviceTypeLabel	1067290	object
PricingTypeLabel	1067290	object
TotalLineItems	1067290	int64
Quantity	1067290	int64
UnitPMPEUR	1067290	object
OrderCreationDate	1067290	object
OrderShipDate	1067290	object
OrderNumCustomer	1067290	int64
IsOnSale	1059475	float64
BillingPostalCode	1067238	object

b. **Product :** cette table représente l'ensemble des différents produits. Chaque ligne correspond à une paire de chaussures définie par le modèle, la couleur, et la pointure.

Nom des colonnes	le nombre de valeurs non nulles	Type
VariantId	533520	float64
GenderLabel	533520	object
MarketTargetLabel	533520	object
SeasonLabel	533520	object
SeasonalityLabel	533520	object
BrandId	533520	float64
UniverseLabel	533520	object
TypeBrand	533520	object
ProductId	533520	float64
ProductType	532418	object
SupplierColor	533520	object
ProductColorId	533520	float64
MinSize	533520	float64
MaxSize	533520	float64
CalfTurn	63981	float64
UpperHeight	120019	float64
HeelHeight	511895	float64
PurchasePriceHT	533520	float64
IsNewCollection	533520	float64
SubtypeLabel	459215	object
UpperMaterialLabel	67933	object
LiningMaterialLabel	7709	object
OutSoleMaterialLabel	7321	object
RemovableSole	510413	object
SizeAdviceDescription	497219	object

c. **Customer :** c'est la table des clients. Chaque ligne indique un ensemble d'information sur un client à savoir sa date de naissance, son pays de résidence, son genre, et la date de sa première commande.

Nom des colonnes	le nombre de valeurs non nulles	Type
CustomerId	780238	float64
CountryISOCode	780238	object
BirthDate	780238	object
Gender	780238	object
FirstOrderDate	780238	object

d. **Returns :** ne contient pas de variables explicatives mais plutôt les observations. Chaque ligne correspond à un produit retourné, identifié par l'OrderNumber et le LineNumber.

2.2.4 La jointure des tables :

Afin de prendre en considération le maximum des variables explicatives sur le modèle nous avons décidé de faire la jointure entre les trois tables. La première jointure que nous avons faite c'est entre la table Order et la table Customers en utilisant CustomerId comme clé primaire.

La seconde jointure est faite avec la table Products en utilisant comme clé primaire VariantId

Afin de préserver le même nombre de ligne que la table Order nous avons utilisé la jointure à gauche qui implique que l'on sélectionne toutes les lignes respectant le critère de jointure, puis on ajoute toutes les lignes de la table Order qui ont été rejetées car elles ne respectaient pas le critère de jointure.

2.2.5 Les différents types de colonnes :

Nous pouvons constater que dans la majorité des cas les données sont soit des données catégoriques (prenons 2 ou plusieurs valeurs fini) soit numériques, soit sous formes de texte.

2.2.5.1 Type catégorique :

Nous constatons qu'il existe deux types de variables explicatifs catégoriques.

a. Classe binaire :

Les différentes variables sont : ***OrderTypelabel, SeasonLabel_x, CustomerTypeLabel.***

b. Multi classe :

Les différentes variables sont: ***LineItem, PayementModeLabel, IsoCodeDevice, TypeLabelPricing, TypeLabel, TotalLineItems, Quantity, IsOnSale, CountryISOCode, Gender, GenderLabel, MarketTargetLabel, SeasonLabel_y, SeasonalityLabel, UniverseLabel, TypeBrand, ProductType, MinSize, MaxSize, CalfTurn, UpperHeight, HeelHeight, IsNewCollection, SubtypeLabel, UpperMaterialLabel, LiningMaterialLabel, OutSoleMaterialLabel, RemovableSole, SizeAdviceDescription***.

2.2.5.2 Type numérique :

Les colonnes qui prennent des valeurs numériques sont soit des valeurs continues soit des valeurs discrètes. Les colonnes qui prennent des valeurs discrètes sont les colonnes qui contiennent les identifiants *(***"BrandId", "CustomerId", "ProductColorId", "ProductId ", "VariantId"***)* et les colonnes **"OrderNumber"** *et* **"OrderNumCustomer».**

Les colonnes qui prennent des valeurs continues sont : "***UnitPMPEUR***" et ***"PurchasePriceHT".***

2.2.5.3 Type texte :

Ce sont les colonnes : ***SupplierColor*** qui indique la couleur de l'article, ***BillingPostalCode*** qui contient le code postal de facturation.

2.2.5.4 Variables temporelles :

Ce sont les colonnes qui contiennent les différentes dates. Ce sont les colonnes "***BirthDate***", ***"FirstOrderDate", "OrderCreationDate***" et "***OrderShipDate".***

III. Sélection des variables explicatives :

3.1 Elimination des variables nulles

Après la jointure des 3 tables X_train, Products et customers, nous avons remarqué qu'ils av aient des variables explicatives ayant des valeurs manquantes :

Variables explicatives	nombre de valeurs manquantes
IsOnSale	7815
BillingPostalCode	52
CountryISOCode	276877
Gender	276877
FirstOrderDate	276877
GenderLabel	377620
MarketTargetLabel	377620
SeasonLabel_y	377620
SeasonalityLabel	377620
BrandId	377620
UniverseLabel	377620
TypeBrand	377620
ProductId	377620
ProductType	380096
SupplierColor	377620
ProductColorId	377620
MinSize	377620
MaxSize	377620
CalfTurn	912679
UpperHeight	862172
HeelHeight	420752
PurchasePriceHT	377620
IsNewCollection	377620
SubtypeLabel	500525
UpperMaterialLabel	985176
LiningMaterialLabel	**1054836**
OutSoleMaterialLabel	**1051884**
RemovableSole	437089
SizeAdviceDescription	450431
Birth_Years	276879
Birth_Month	276879
Birth_Day	276879

Ceci peut être expliqué par le fait que la majorité des produits vendus ne sont pas répertoriés dans la tables products ou par le fait que certains clients n'ont pas effectué de commandes pendant la période couverte par la table Order. Et par suite, lorsqu'on fait la jointure des différentes tables cela va entrainer plusieurs valeurs manquantes. Comme on peut le voir dans le tableau ci-dessus, les features **« SizeAdviceDescription, SubtypeLabel, RemovableSole, HeelHeight, UpperHeight, LiningMaterialLabel, OutSoleMaterialLabel, UpperMaterialLabel et CalfTurn »** contiennent plus que 450 000 de valeurs manquantes. C'est pour cela qu'on les a éliminés parce qu'ils constituent du bruit et n'améliorent pas notre modèle.

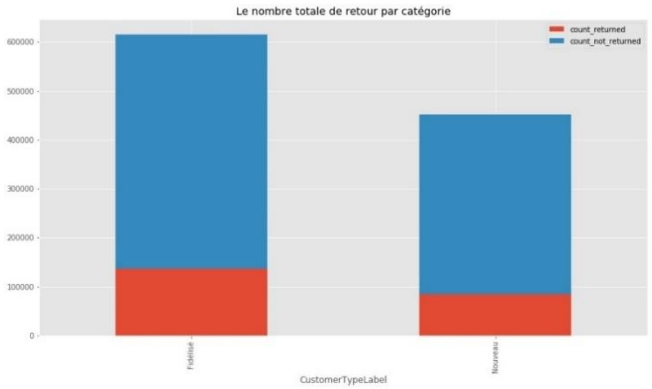
3.2 Description des variables explicatives

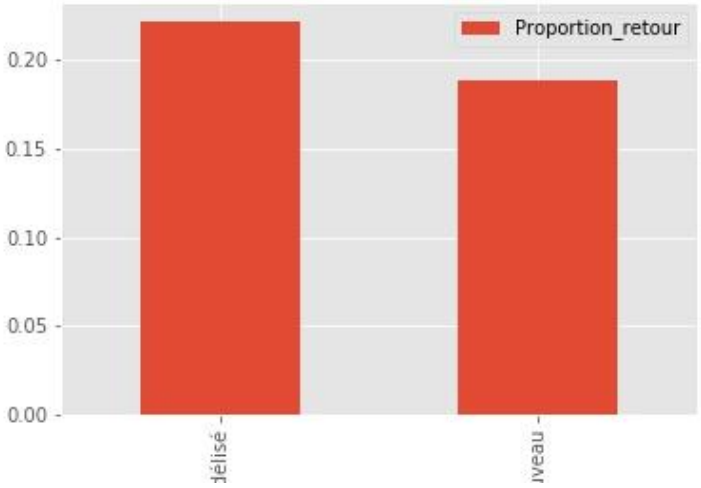
Variable catégorique :

Dans cette partie nous nous intéresserons à l'interprétation de quelques variables explicatives qui sont les plus pertinentes. Les autres variables s'interprètent de la même manière.

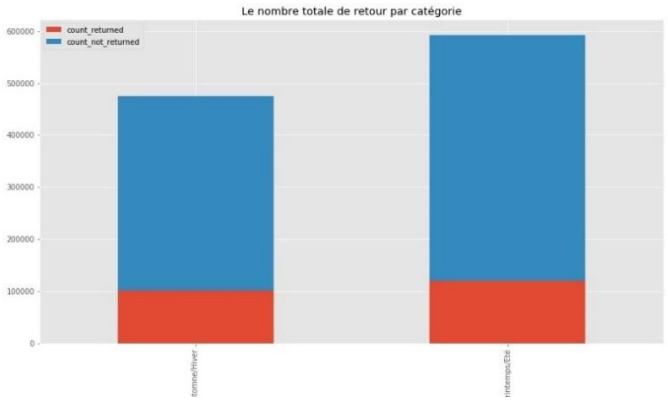
Classe binaire :

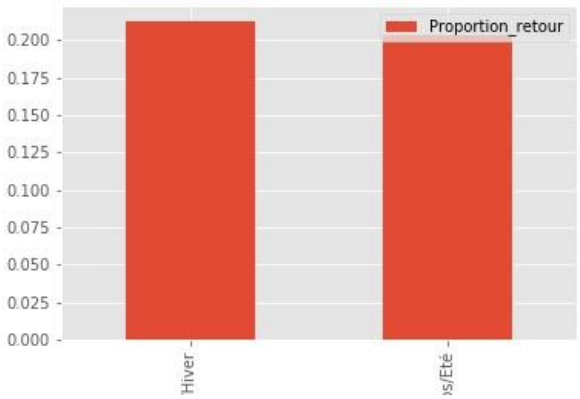
CustomerTypeLabel: nous pouvons constater à partir du graphe que le nombre de retour pour les clients fidélisé est plus important que pour les nouveaux clients. Ceci peut être expliqué par le fait que les clients fidélisés bénéficient d'un service qui leur facilite retour de produit.





SeasonLabel_x : nous pouvons constater à partir des graphes ci-dessous que le taux de ret our dans les saison automne/hiver ou Printemps/été est presque le même. Mais le nombre d'article retourné est assez important dans les deux saisons (approximativement 20% des a rticles ont été rendus) Nous constatons que cette variable est importante pour le modèle mais a moins d'influence que la variable CustomerTypeLabel.

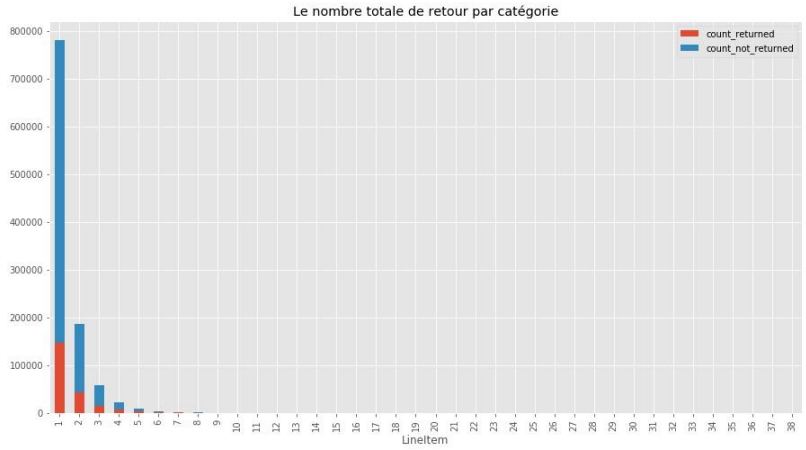


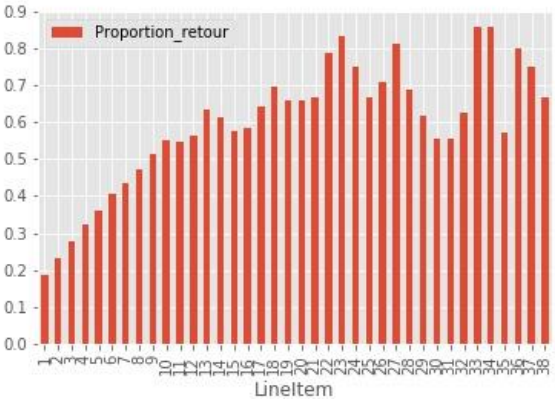


Multi classe :

LineItem : Nous pouvons constater à partir du graphe que plus l'article commandé se trou ve dans la cinquième position ou plus, plus il a de chance d'être rendu. Nous pouvons mê me dire que si la position de l'article dépasse 30, le produit est quasi-systématiquement ren du (taux>80%)

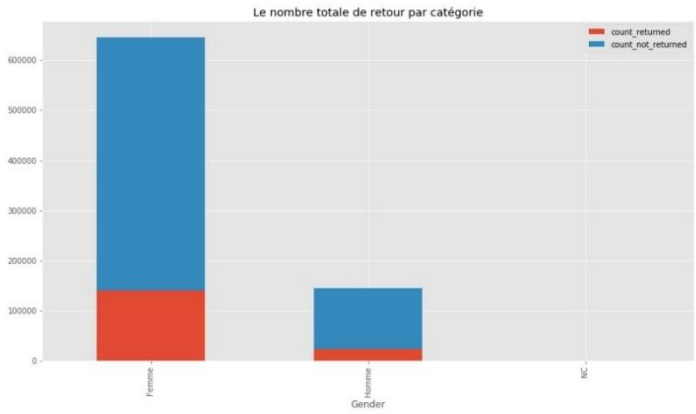
Par ailleurs Nous pouvons constater que la majorité des articles commandés se trouve dan s la première position c'est-à-dire que les clients ont tendance à ne commander qu'un se ul article ou deux lors de leur achat en ligne

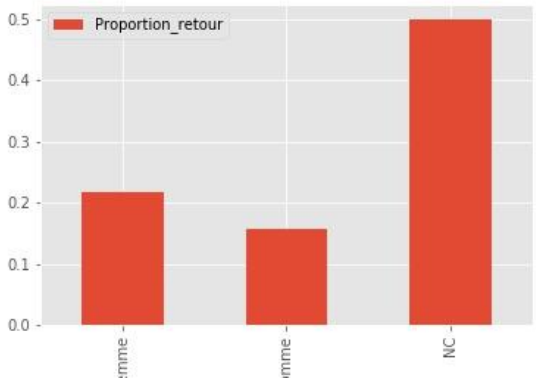




Gender :

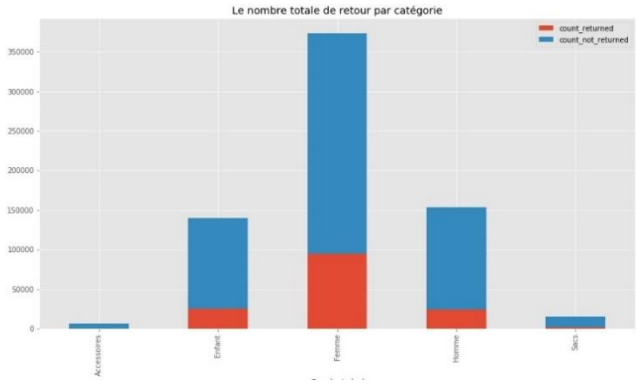
Les graphes ci-dessous montrent que la majorité des articles sont achetés par des femmes (81% des commandes sont effectuées par des femmes). Le taux de retour des articles est p lus important chez les femmes que chez les hommes.

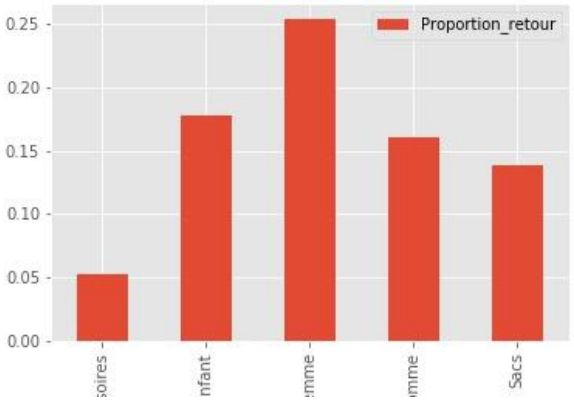




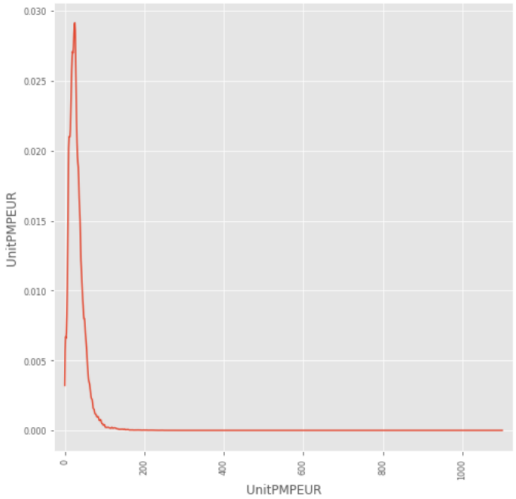
GenderLabel :

Nous pouvons voir ici que le nombre d'articles commandés dédiés aux femmes est le plus important. De plus le nombre d'articles rendu est plus important pour la catégorie « femm e » chose qui est tout à fait prévisible puisque on trouve des résultats similaires à travers la variable Gender.





Variable numérique :

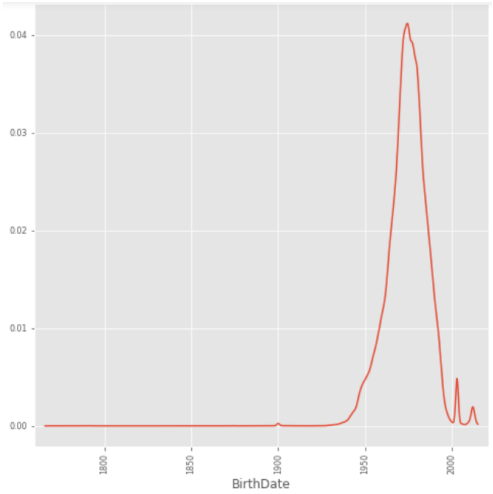


Le prix des différents articles d'étale entre 1100€ et 0€. Nous pouvons ici voir que la majorit é des articles ont un prix qui est concentré sur l'intervalle [0,100€]. Ainsi les différents articles dont le prix dépasse 200€ n'influe pas sur le modèle et peut même constituer un bruit.

Variable temporelle :

Nous nous intéresserons dans notre analyse à la variable « Birthdate » qui donne une idée s ur l'age des personnes qui font les achats en ligne

Birth date :



Nous pouvons voir à partir de ce graphe que la majorité des clients ont un age entre 20 et 40 ans.

3.2 Transformation des variables :

3.2.1 Transformation des variables catégoriques :

Nous avons remarqué que certains algorithmes de Machine Learning ne pouvaient pas être appliqués en présence des variables catégoriques. C'est pour cela que nous avons utilisé la technique de **« One Hot Encoding »** qui permet de transformer chaque catégorie de ces variables en des variables binaires indépendantes.

Par exemple, nous avons remplacé la variable SeasonLabel_x en deux variables explicatives **« SeasonLabel_x_Automne/Hiver »** et **« SeasonLabel_x_Printemps/Eté ».**

3.2.2 Transformation des variables numériques :

Nous avons décidé de transformer la variable **« UnitPmpeur»** en une variable catégorique. En fait, nous avons défini plusieurs intervalles de prix pour mieux classifier les produits : Chaque intervalle va constituer une catégorie du produit (produit de bas gamme, produit de moyenne gamme, produit de luxe, etc)

3.2.3 Transformation des variables textes :

Pour ces types de variables, nous les avons enlevés vu qu'ils prennent plusieurs valeurs et que leur influence est négligeable.
En plus, ça crée un problème de mémoire.

3.2.4 Transformation des variables temporelles :

Nous avons essayé de traiter les variables temporelles. En fait, nous avons extrait uniquement à partir du feature **« Birth »** l'année de naissance des clients parce que leurs âges pouvaient influencer leurs comportements. En fait, nous avons constaté que les jeunes ont tendance à retourner les produits achetés plus que les personnes âgées. Concernant le feature **« OrderCreationDate »**, nous avons jugé pertinent d'extraire l'année et le mois de la vente des produits car ils étaient importants pour notre modèle.

3.3 Remplissage des valeurs NAN :

Concernant les valeurs manquantes, nous avons essayé différentes techniques. Au début, on a essayé de les remplacer par la valeur zéro mais ceci avait une mauvaise répercussion sur notre modèle. Ensuite, nous avons remarqué qu'il y avait plusieurs features ayant beaucoup « d'outliers » comme la variable « MaxSize » et « UnitPMpmeur ». C'est pour cela que nous avons décidé d'utiliser la valeur médiane pour le remplissage des valeurs NAN et ceci nous a permis d'améliorer le score de notre modèle.

IV. Prédiction

Pour la prédiction, nous avons essayé plusieurs modèles de prédiction. Ci-dessous, nous allons parler des différents modèles de prédictions qu'on a suivi et ce qui nous a poussé à basculer vers un autre model.
La sélection des différents modèles a été faite sur un échantillon de la data pour minimiser le risque de memory error.

4.1 Logistic regression :

Au début, vu que c'était notre première expérience et que nous n'avons pas encore étudier les méthodes de bagging. Nous avons opté pour la régression logistique.
Le score que nous avons obtenu avec la logstic regression est : 0.500746014262
Et le temps d'exécution est : 0.9679994583129883 secondes.

4.2 Decision tree :

Après, nous avons essayer d'utiliser le model de decesion tree. Ceci nous a permis d'améliorer un peu le score, cependant, pas d'une manière importante.
D'autre part, nous avons constater que l'augmentation du max depth nous fait un over fitting, ce qui fait que lorsque nous avons testé le modèle avec nos données, nous avons obtenu un bon score, cependant, lorsque nous l'avons essayé avec toutes les données, nous avons obtenu un très mauvais score à cause de l'over fitting.
Le meilleur score que nous avons obtenu avec ce modèle est : 0.619490970452 avec une profondeur maximale d'arbre de 29.
Pour le temps d'exécution, ça nous a pris exactement 2.8774871826171875 secondes.

4.3 Random Forest :

Après l'utilisation de decision tree, nous avons utilisé le Random Forest,
ce modèle surmonte plusieurs problèmes par rapport au decision tree, notamment :

- Réduction du overfitting : En faisant la moyenne de plusieurs arbres, il y a un risque significativement plus faible de overfitting.

- Moins de variance : En utilisant plusieurs arbres, vous réduisez le risque de trébucher sur un classificateur qui ne fonctionne pas bien en raison de la relation entre le train set et test set.

Par conséquent, nous avons obtenu un meilleur score avec ce modèle là que les 2 premiers que nous avons utilisés.
Le score que nous avons obtenu avec ce modèle est de 0.642485536204 avec une profondeur maximale de 29.
Le temps d'exécution est de 2.288196563720703 secondes.

4.5 Réseau de neurones :

Nous avons opté pour ce modèle de réseau de neurones après la séance du TP que nous avons fait.
L'utilisation de ce modèle nous a permis d'avoir un meilleur score par rapport aux modèles précédents.
Cependant, le temps d'exécution de ce modèle été très important, quelquefois nous avons attendu plus que 20 minutes pour avoir un résultat.

4.7 Xgboost :

Comme nous avons déjà un modèle qui utilise Bagging, nous avons voulu d'utiliser un modèle qui est boostant. Parmi les algorithmes de boost possibles, nous avons choisis le Xgboost.
Cependant, le temps d'exécution pour avoir un résultat été très important, parce qu'en fait, pour utiliser ce modèle, il faut utiliser un pc puissant. Des fois, pour avoir un résultat, nous avons dû attendre plus qu'une heure.

4.8 Catboost :

Finalement, nous avons opté pour Catboost, pour plusieurs raisons :

- Catboost est insensible au bruit et au overfitting.
- Le score obtenu avec Catboost était meilleur que tous les autres modèles que nous avons utilisés.
- Le temps d'exécution été moins important d'une manière significative que les autres modèles, ce qui nous a permis de modifier les paramètres d'une façon plus effective et rapide.
- Supporte les variables catégoriques numériques ainsi que les variables explicatives catégoriques.
- Superior quality when compared with other GBDT libraries such as gradient boosting or Xgboost.

Pour la documentation sur ce modèle, nous avons utilisé ce lien :
https://tech.yandex.com/catboost/doc/dg/concepts/about-docpage/

V. Conclusion

Pour conclure, ce challenge été pour nous une première expérience qui nous a vraiment permis de manipuler la data et de mettre en œuvre tout ce que nous avons étudié de théorique.

Les différents manipulations et transformations que nous avons fait avec les features étaient également une première pour nous, chose que nous n'avons pas fait au cours.

Ceci a été également une chance pour nous de découvrir plus de modèle que ceux que nous avons fais au cours, notamment le Xgboost et le Catboost.

Le meilleur score que nous avons obtenu est de **0.704061897109** avec le modèle Catboost.