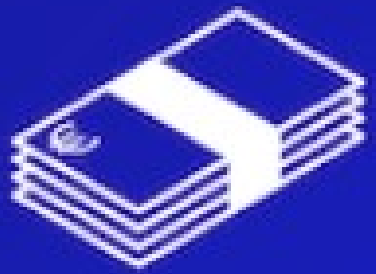
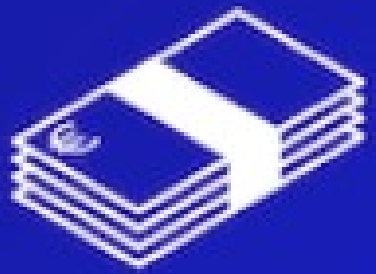


ONCFM

Projet 10 : Détectez des faux billets avec Python

Déployer l'application : https://dashboard.heroku.com/apps/oncfm/deploy/github



ONCFM

Projet 10 : Détectez des faux billets avec Python

Introduction

L’ONCFM a pour objectif de mettre en place des méthodes d’identification des contrefaçons des billets

- **Problématique :** mettre en place une modélisation qui serait capable d’identifier automatiquement les vrais des faux billets uniquement à partir simplement de certaines dimensions du billet et des éléments qui le compose
- **Cahier des charges :**
  - **une analyse descriptive des données,** notamment la répartition des dimensions des billets, le nombre de vrais / faux billets,
  - Nous avons à notre disposition six données géométriques pour chaque billet.
  - L’algorithme doit être capable de prendre en entrée un fichier contenant les dimensions de plusieurs billets, et de déterminer le type de chacun d’entre eux, à partir des seules dimensions.
  - Nous aimerions pouvoir mettre en concurrence deux méthodes de prédiction :
    - **Une régression logistique classique ;**
    - **Un k-means**, duquel seront utilisés les centroïdes pour réaliser la prédiction.
  - Pour une évaluation optimale des modèles, nous souhaitons avoir une analyse des nombres de faux positifs et faux négatifs via une matrice de confusion

SOMMAIRE

- Analyse descriptive
  - Imputation des données manquantes par Régression linéaire
    - Modèle de régression linéaire multiple
    - bilan
  - Description du jeu de donnée
    - Analyse univariée
    - Analyse bivariée
- Algorithmes
  - Comparaisons entre les algorithmes
  - Démonstration et test du modèle finale

Analyse descriptive

Les données :

Un dataset de travail avec 6 variables et une colonne étiquette

- **length** : la longueur du billet (en mm) ;
- **height_left** : la hauteur du billet (mesurée sur le côté gauche, en mm) ;
- **height_right** : la hauteur du billet (mesurée sur le côté droit, en mm) ;
- **margin_up** : la marge entre le bord supérieur du billet et l'image de celui-ci (en mm) ;
- **margin_low** : la marge entre le bord inférieur du billet et l'image de celui-ci (en mm) ;
- **diagonal** : la diagonale du billet (en mm).

Analyse descriptive

```
Entrée [4]:
1 # On cherche les valeurs manquantes
2 billets_df.isnull().sum()

Out[4]:
is_genuine      0
diagonal        0
height_left     0
height_right    0
margin_low     37
margin_up       0
length          0
dtype: int64
```

Première étape :
traiter les valeurs manquantes

Imputation des données manquantes par Régression linéaire

On applique la fonction pour trouver le model optimal par algorithme "backward"

```
from functions_RL import *
columns = ['is_genuine','margin_low','diagonal','height_left','height_right','margin_up','length']
reg_backward = backward_selected(df_valide[columns], 'margin_low')
```

```
_____________________________________
margin_low ~ height_left + diagonal + height_right + length + is_genuine + margin_up + 1
remove length (p-value : 0.868 )

_____________________________________
margin_low ~ height_left + diagonal + height_right + is_genuine + margin_up + 1
remove diagonal (p-value : 0.719 )

_____________________________________
margin_low ~ height_left + height_right + is_genuine + margin_up + 1
remove height_right (p-value : 0.496 )

_____________________________________
margin_low ~ height_left + is_genuine + margin_up + 1
remove height_left (p-value : 0.454 )

_____________________________________
margin_low ~ is_genuine + margin_up + 1
is the final model !
```

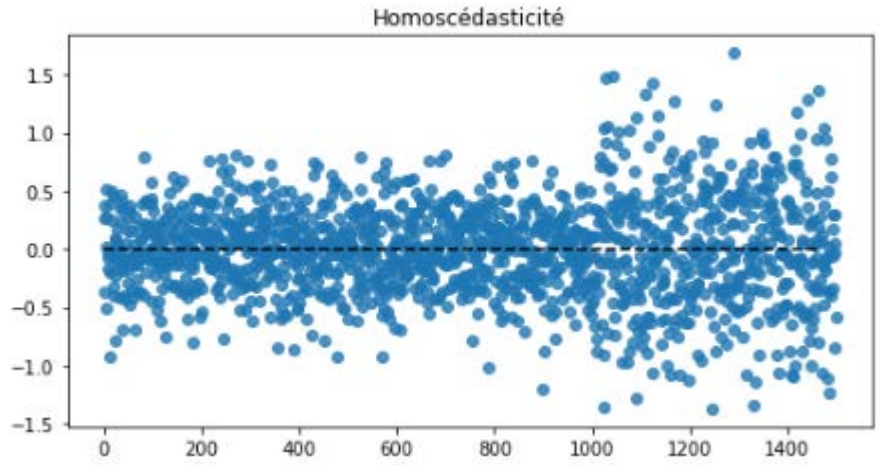
Le meilleur modele pour la régression Linéaire est : 'margin_low ~ is_genuine + margin_up + 1'

- R^2 : 0.617
- AIC: 1555.
- BIC: 1571.

ONCFM

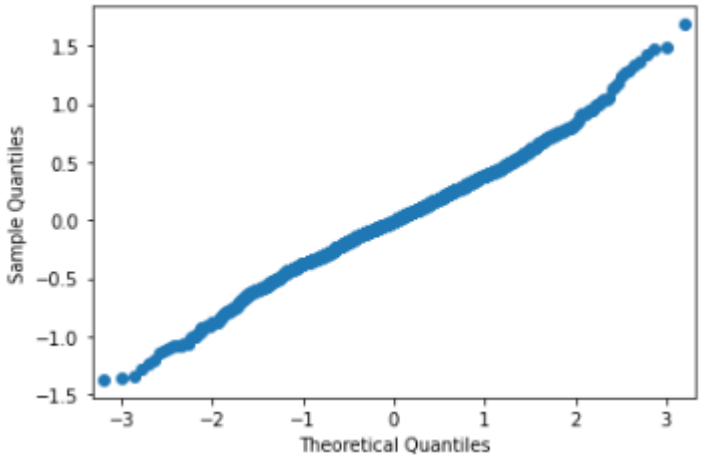
Modèle de régression linéaire multiple

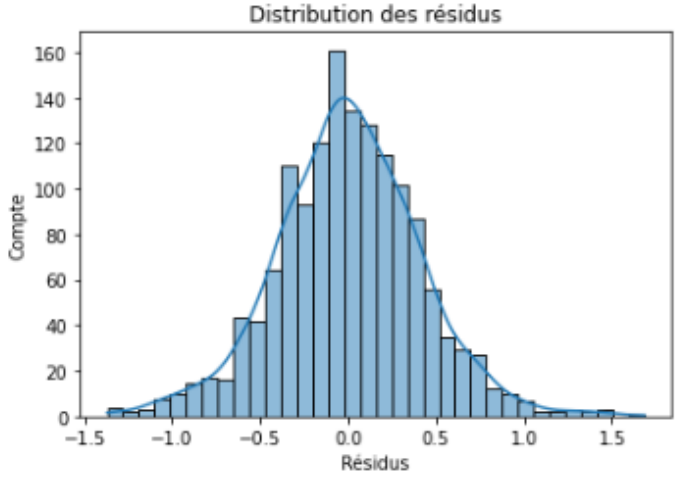
- Homoscédasticité :
  - **Le test de White :**
    - La p-value est ici très inférieure au seuil : p-value': 4.769905016347682e-35, L'hypothèse d'homoscédasticité de notre régression linéaire validée



- La normalité des résidus

Le qqplot et la répartition des résidus montre
Que les résidus suivent une loi normale (gaussienne)

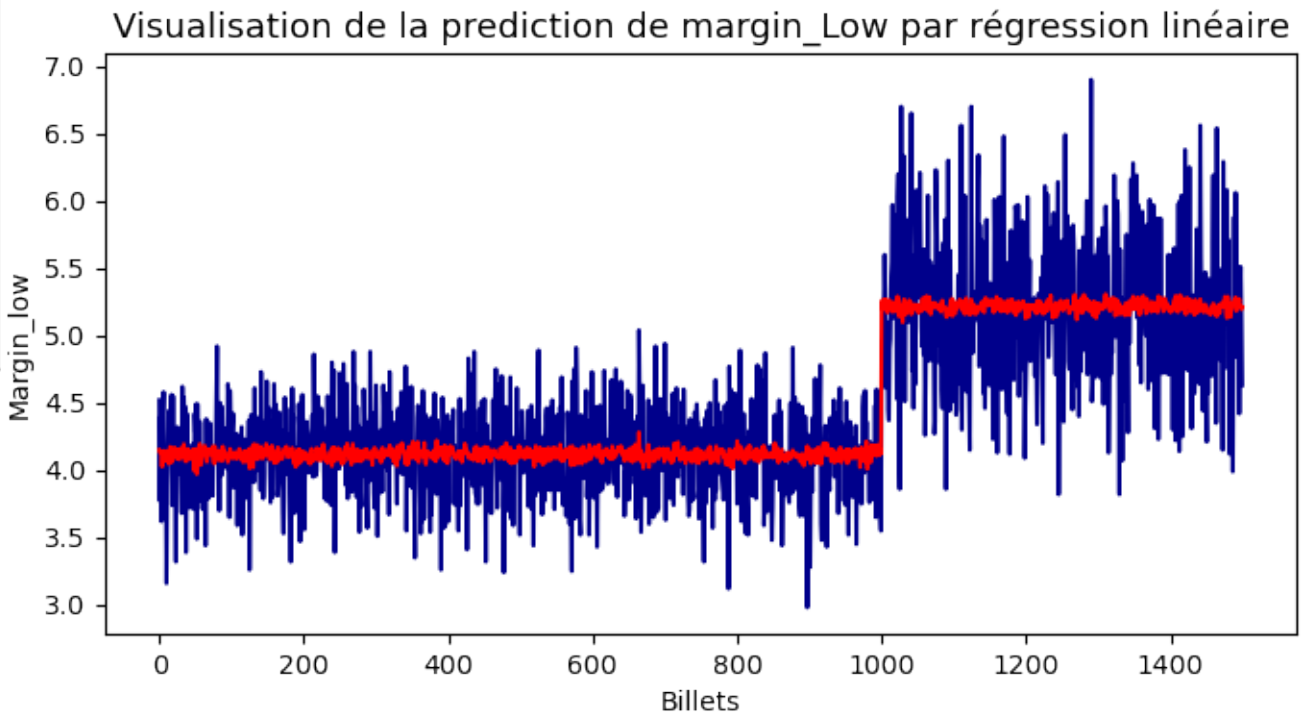




ONCFM

Bilan

```
Comparaison avec les premières valeurs de margin_low
              AVANT : APRES :
       mean : 4.486 : 4.483 : proche
       std  : 0.664 : 0.660 : proche
       min  : 2.980 : 2.980 : égalité
       25%  : 4,015 : 4.027 : proche
       50%  : 4.310 : 4.310 : **égalité**
       75%  : 4.870 : 4.870 : **égalité**
       max  : 6.900 : 6.900 : **égalité**
```



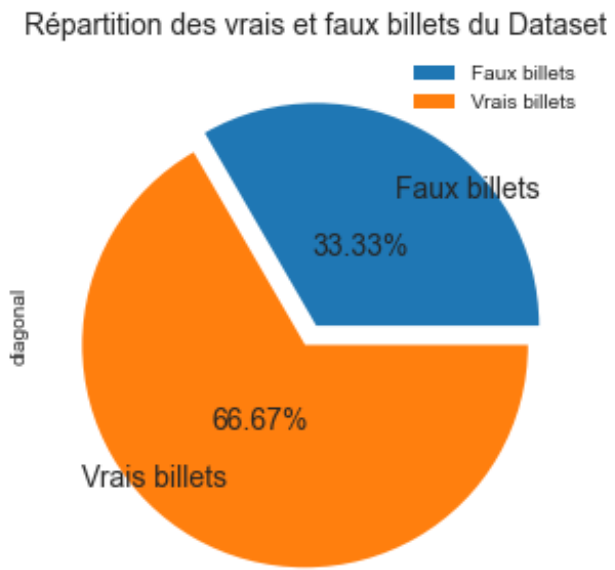
Les hypothèses du modèle de régression linéaire sont confirmées par notre analyse:

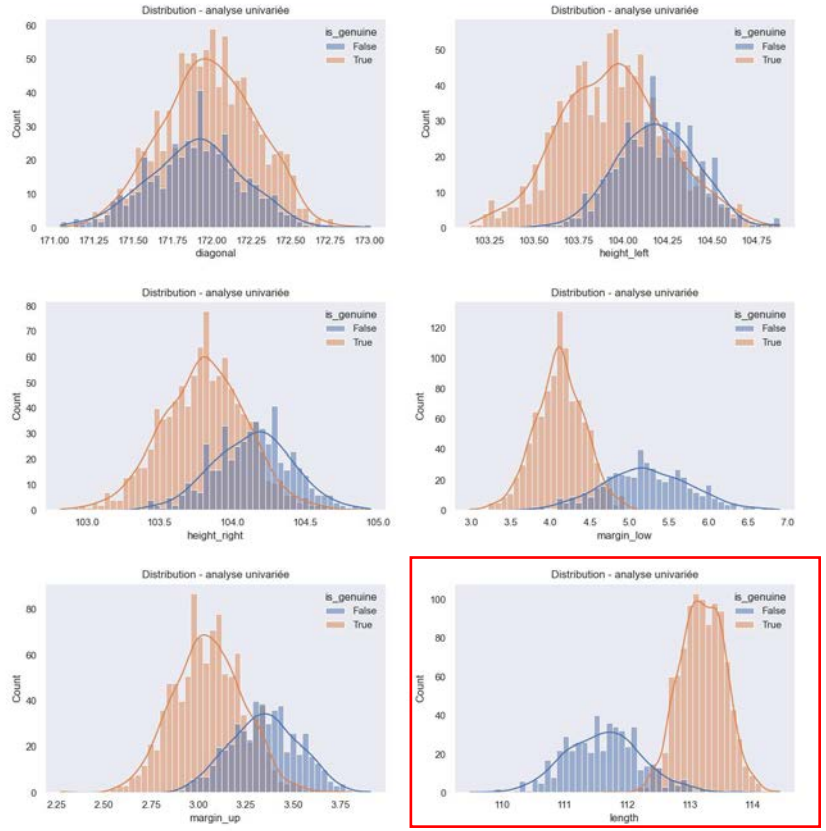
- **Linéarité** : La relation entre les variables dépendantes et indépendantes est linéaire.
- **Homoscédasticité** : la variance constante des erreurs est maintenue.
- **Normalité multivariée** : les résidus sont distribués normalement.
- **Manque de multicolinéarité** : il y a pas de multicolinéarité dans les données.

ONCFM

Description du jeu de donnée - complet
Comprendre les données

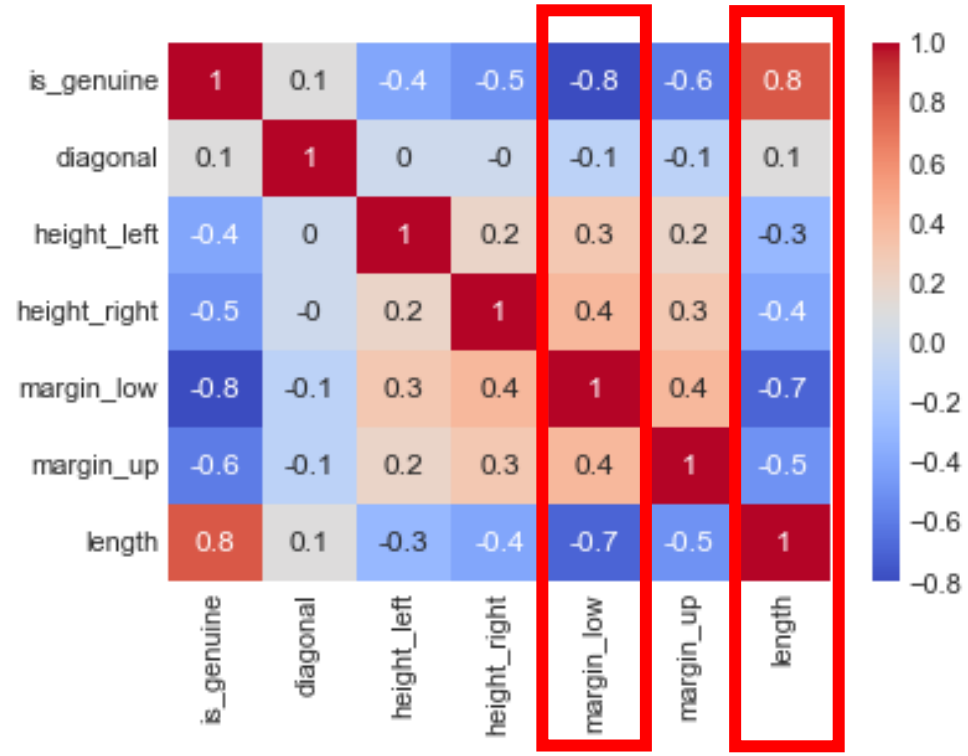
- Le dataset complet contient : 1500 billets
  - 1000 billets vrais
  - 500 faux

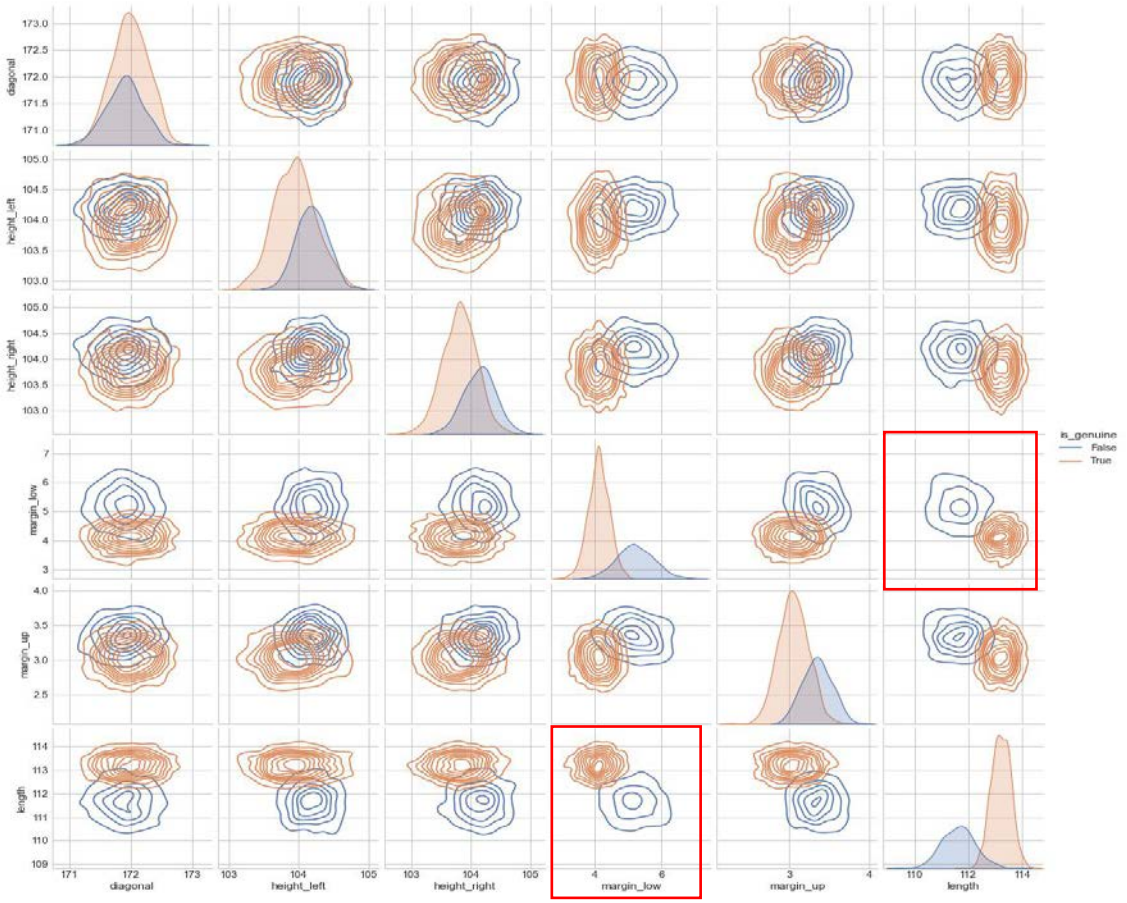




ONCFM

Corrélation et répartition





Margin_low et length sont les variables les plus corrélée aux autres mais ne suffisent pas à séparer totalement les vrais des faux billets

ONCFM

« les différentes pistes explorées pour la construction de l'algorithme, ainsi que le modèle final retenu ».

Les notebooks pour…

… **visualiser** les clusters :

- CAH
- ACP TSNE

… **explorer** les algorithmes :

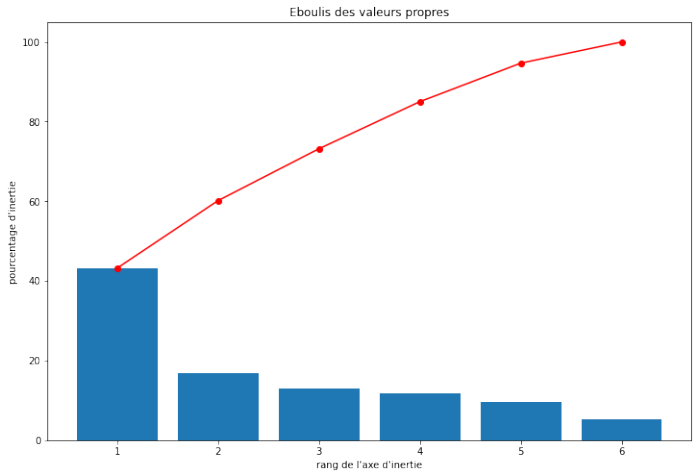
- Kmeans
- Régression logistique et SVC
- Random Forest

… **répondre** aux demandes :

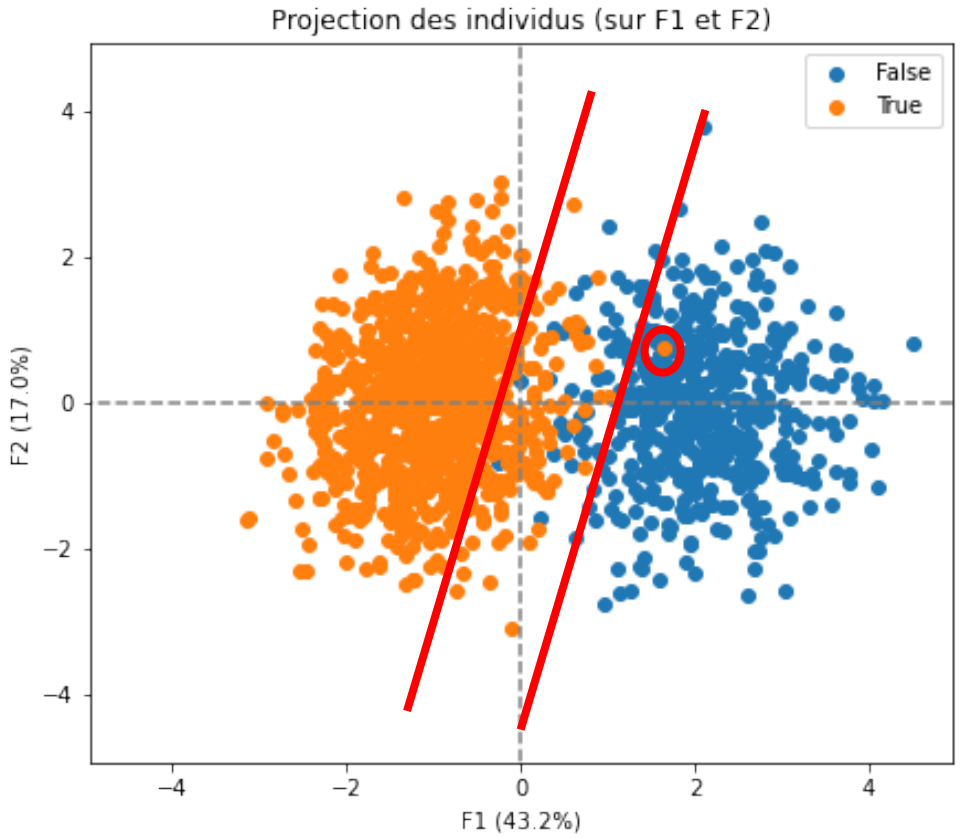
- Livrable

Visualiser les clusters

ACP



Les deux premières composantes de l'ACP expliquent **60%** de la variance.



ONCFM

Explorer les algorithmes :

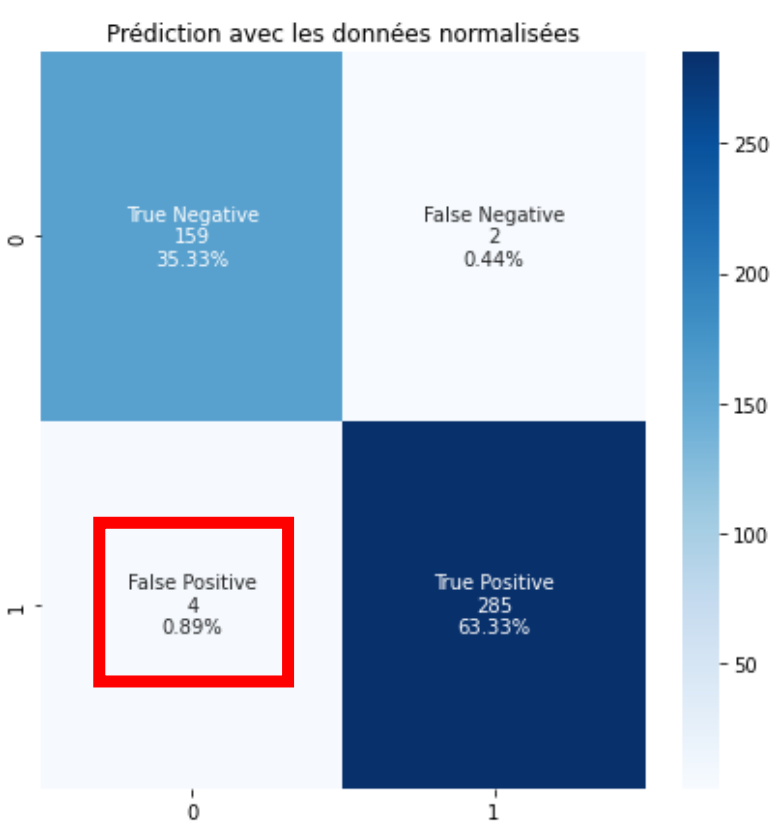
- Kmeans
- Régression logistique et SVC
- Random Forest

- Le modèle conforme aux demandes de Marie
  - Livrable

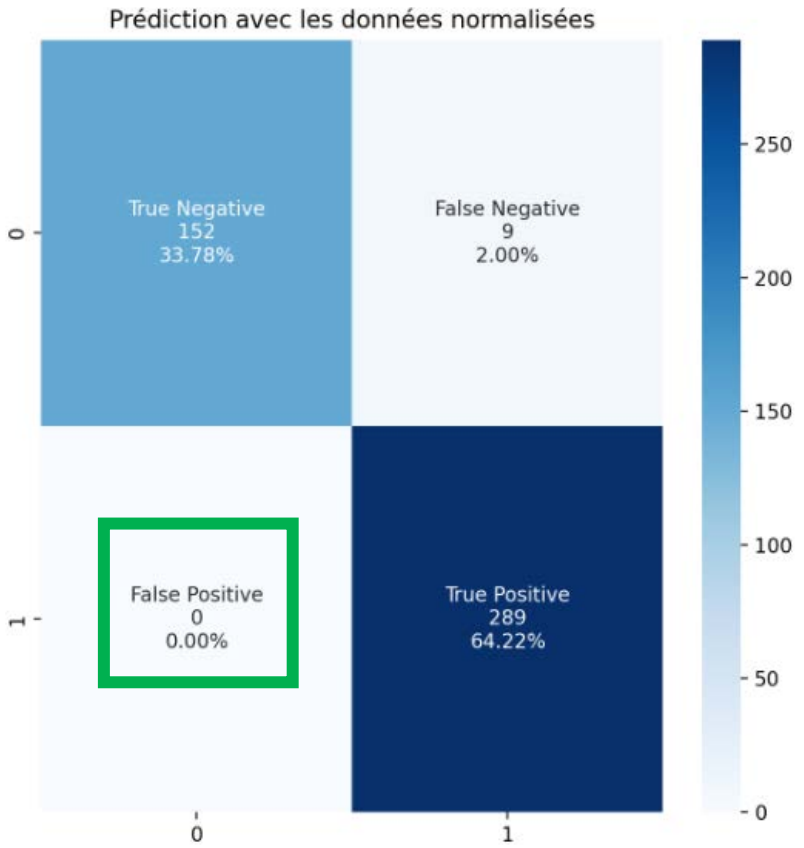
Régression logistique

Régression logistique

Comparaison entre Kmeans et Régression logistique



KMEANS



Régression logistique

La régression logistique est plus Performante pour identifier les La validité des billets.

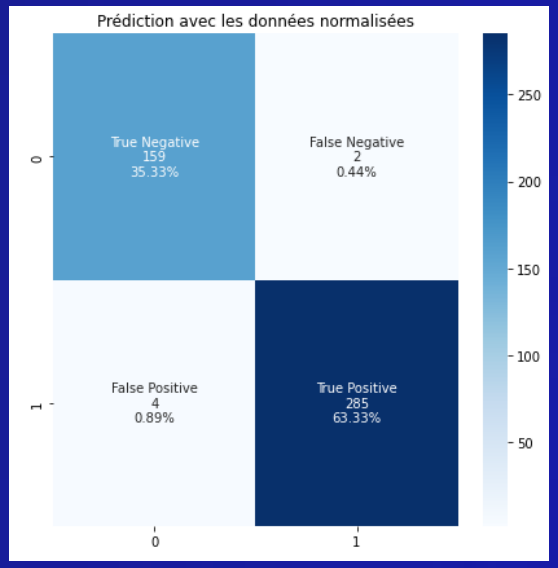
- **Le Kmeans** :
  - **Precision : 99,3 %**
  - **Recall : 98,62 %**
  - **Accuracy : 98,67 %**
  - **F1 score : 98,96 %**

- **Regression logistique**
  - **Precision** : **97,0** %
  - **Recall** : **100 %**
  - **Accuracy** : **98,0** %
  - **F1 score** : **98,5 %**

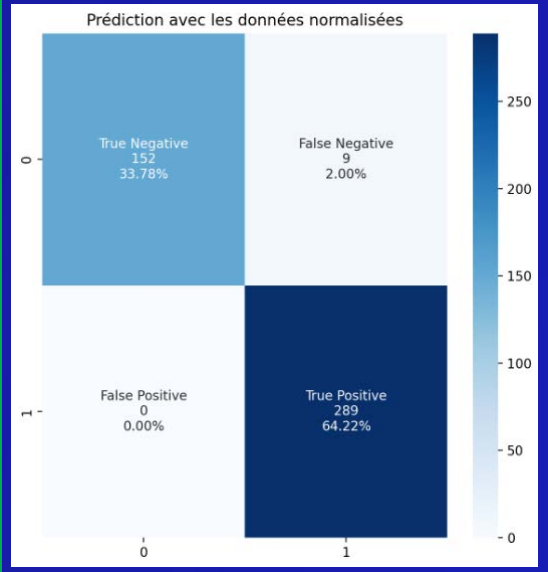
ONCFM

Conclusion

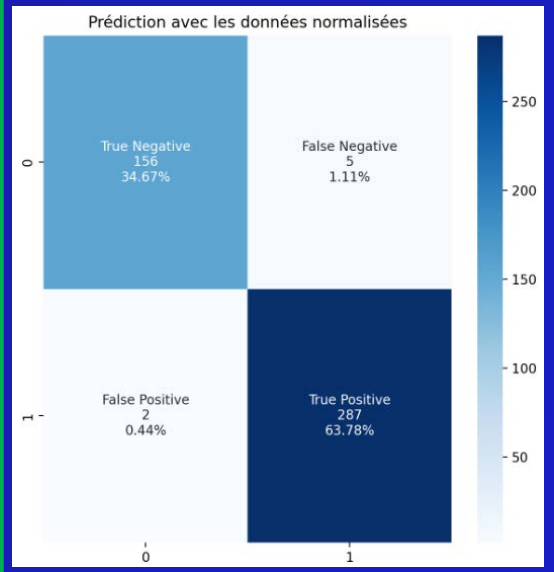
Demandés



KMEANS

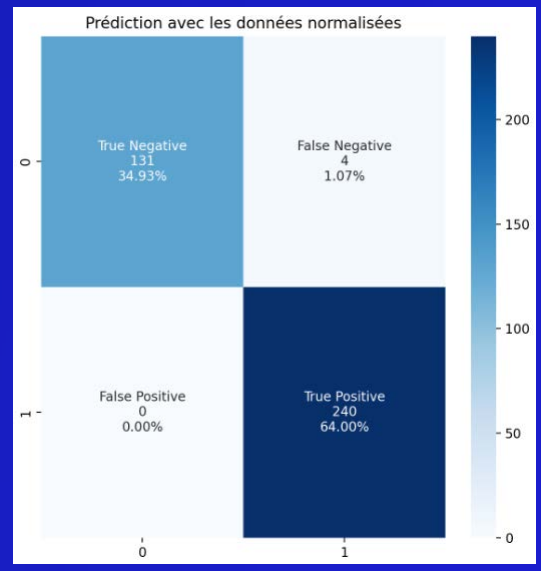


Régression logistique



SVC

Recommandation



Random forest

La régression logistique est la plus pertinente parmi les demandes de Marie

Parmi tous les algorithmes testés : Le meilleur est celui du Random forest :

- Il ne laisse passer aucun faux billets du set (Faux Positif)
- Il minimise les erreurs de prédiction de billet identifié comme faux alors qu'ils sont vrais (Faux Positif)

Démonstrations et modèles

- Création d’une app python avec streamlit
- Déploiement de l’application avec heroku
  - https://oncfm.herokuapp.com/