P10 - Détectez des faux billets avec Python 💶

SOMMAIRE

1- Contexte du projet

ONCFM (Organisation nationale de lutte contre le faux-monnayage), est une entreprise qui à pour objectif de mettre en place des solutions d'identifications de faux billets à partir des dimensions des billets.

C'est dans ce cade que nous avons été sollicité afin de mettre en place un algorithme permettant de faire une catégorisation automatique entre les vrais et faux billets.

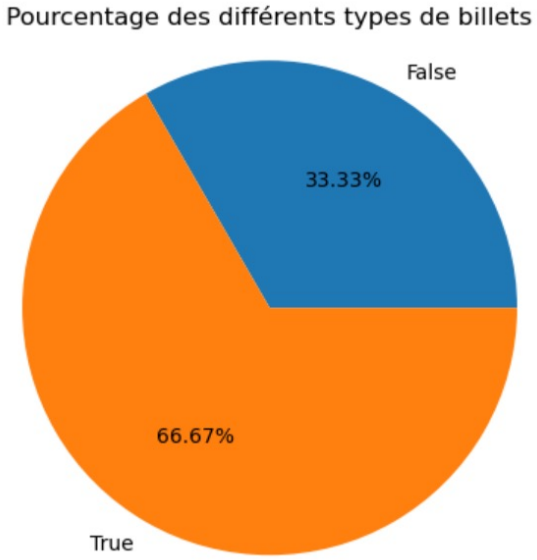
2- Préparation et descriptions des données

Les informations concernant les caractéristiques des billets sont les suivantes:

- **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 inferieur du billet et l'image de celui-ci (en mm)
- **diagonal** : la diagonale du billet (en mm)

Ainsi qu'une variable booléenne « **is_genuine** » qui précise le type du billet.

2- Préparation et descriptions des données



- 1500 lignes et 7 colonnes
- 37 valeurs manquantes, soit 2.46% des données (margin_low)
- Pas de valeur négative
- Pas de doublon
- On observe des outliers dans notre dataset
- 1000 (66,67%) vrais billets contre 500 (33,33) faux

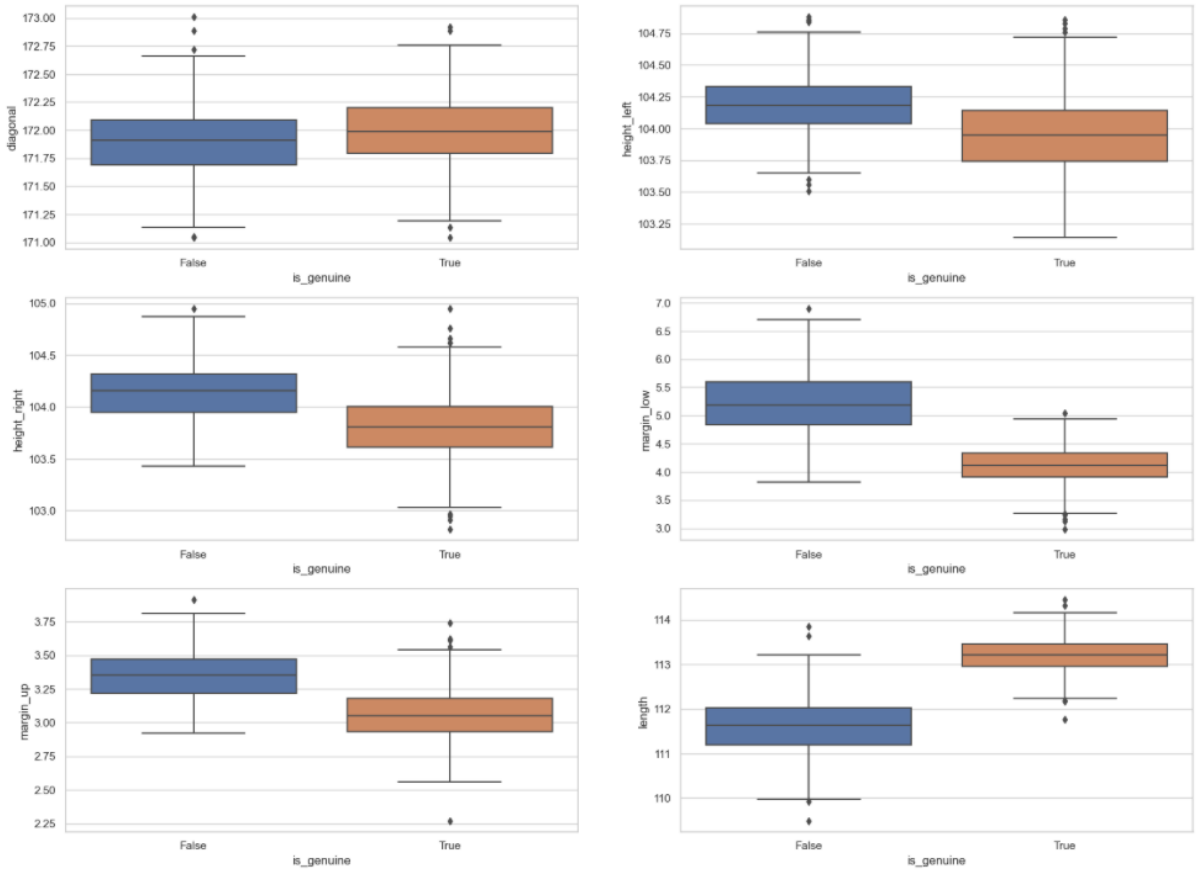
```
# Moyenne des caractéristiques selon le type des billets

data_billets.groupby(["is_genuine"]).mean()
```

is_genuine	diagonal	height_left	height_right	margin_low	margin_up	length
False	171.901	104.190	104.144	5.216	3.350	111.631
True	171.987	103.949	103.809	4.116	3.052	113.202

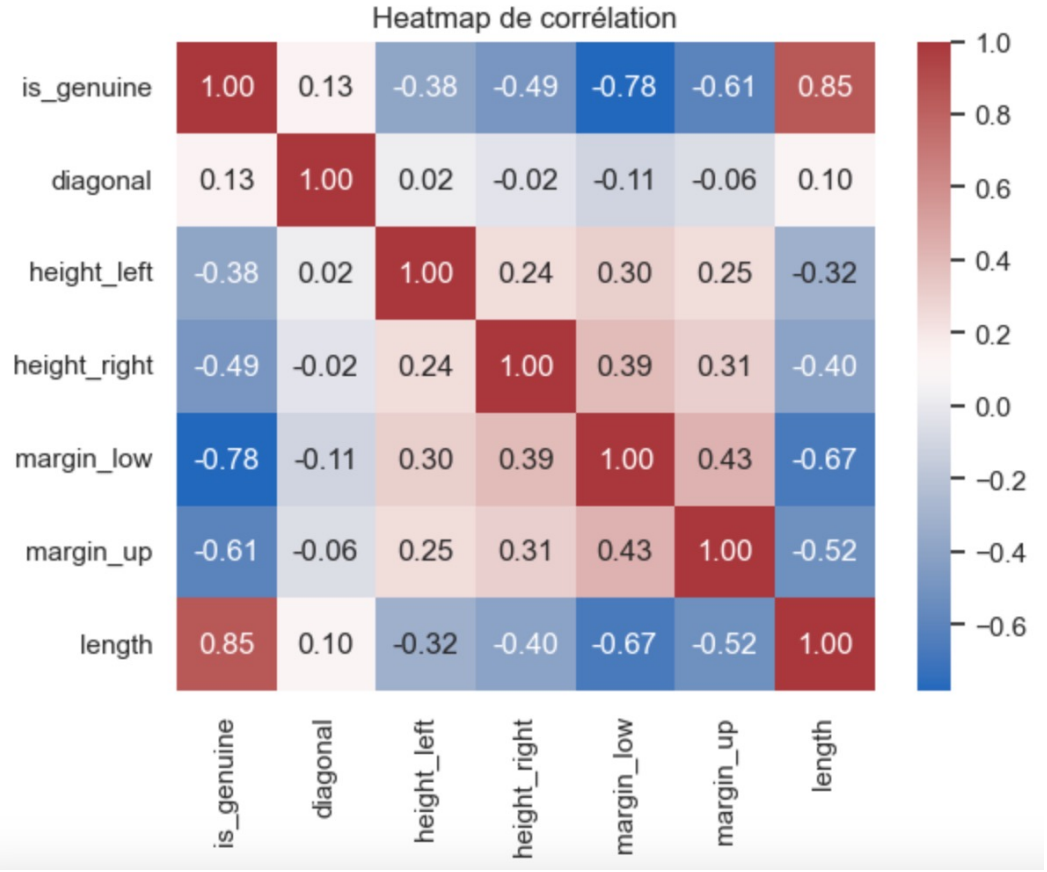
Les vrais billets ont une diagonale et une longueur supérieur a ceux des faux.

2- Préparation et descriptions des données



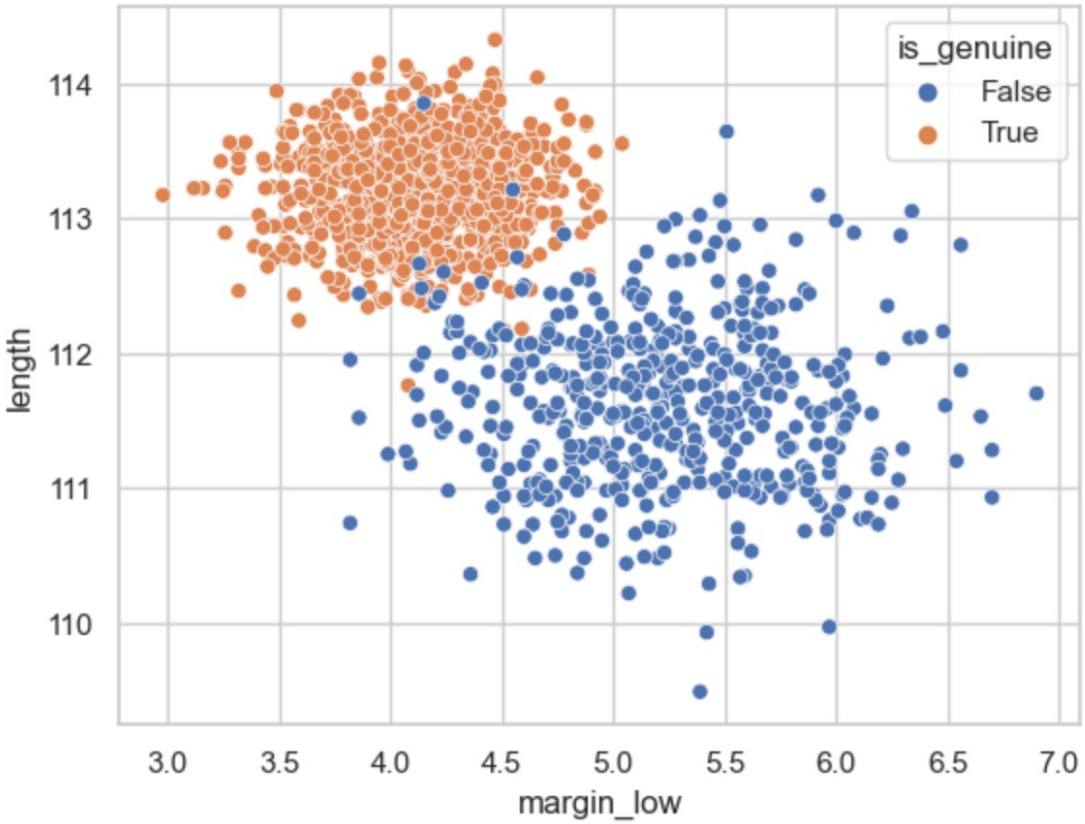
Les écarts les plus importants sont observés au niveau de ‘margin_low’ et ‘lenght’

2- Préparation et descriptions des données



De fortes corrélations sont observées entre les types de billets et les variables ‘margin_low ’ et ‘lenght’.

2- Préparation et descriptions des données



De nos précédentes observations, on remarque mieux la distinction entre faux et vrais billets.

3- Traitement des valeurs manquantes

Afin de remplacer les valeurs manquantes de ‘margin_low’, nous avons opté pour la régression linaire selon les étapes suivantes.

- La régression linaire simple
- La régression linaire multiple

Avec la régression linéaire simple, le coefficient de détermination (R^2) était de 43,3% de la variation de 'length' pour expliquer 'margin_low', contre environ 47,8% avec la régression linéaire multiple pour l'ensemble des autres variables.

Régression linéaire simple Coéfficients: [0. -0.50467955] / Intercept: 61.34769617889449
Régression linéaire multiple Coéfficients: [0. -0.13875154 0.17902523 0.24536231 0.35386137 -0.39168795] / Intercept: 27.23845194868485

Avec la régression linéaire simple, l'Erreur quadratique (MSE) est légèrement plus élevée (0.49) par rapport à la régression multiple (0.47).

La régression multiple est donc bien meilleure que la régression simple.

- Détection des valeurs aberrantes et influentes

53 outliers et 275 valeurs influentes ont été observées, de ces derniers 16 ont été identifiées comme aberrantes et influentes.

Après suppression des données atypiques et influentes, nos données n’ont pas vraiment évoluées. Ainsi notre coefficient de détermination est passée à 48%, l'erreur quadratique est lui resté à 0.47

- **Vérification des hypothèses (avant la mise en application de la régression)**

* **La normalité** : Il s'agit ici de vérifier que les erreurs résiduelles sont distribuées normalement. **NON VERIFIEE**
* **Homoscédasticité** : On va vérifier que la variance des erreurs résiduelles est constante à tous les niveaux de la variable prédite. **NON VERIFIEE**
* **Multicolinéarité** : Il s'agit de regarder la relation entre les variables prédictives (ou indépendantes) dans notre modèle de régression. **VERIFIEE**

Ainsi la régression linaire multiple a été utilisée pour remplacer les valeurs manquantes

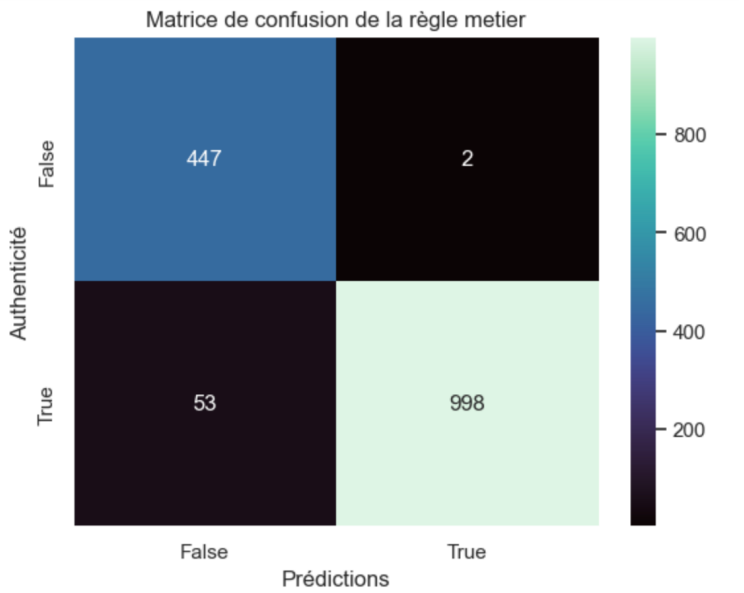
4- Classification par règle métier

Processus de mise en place de la classification avec notre règle métier
Règle métier: **Règle métier simple: Si ‘lenght’ >= 112 et ‘margin_low’ <= 5 alors = Vrai billet**

- Application de la règle métier sur notre dataframe avec la création d’une nouvelle colonne
- Remplacement des valeurs ‘NaN’ par ‘False’
- Séparation des données en deux, ceux contenant l’étiquette de départ et ceux prédite.

4- Classification par règle métier

Calcul des performances de notre modèle



Nous avons 1051 vrais billets dont 998 vrai positifs et 53 faux négatif.

Et 449 faux billets, dont 447 vrai négatif et 2 faux positif.

Précision globale de notre règle métier: 0.9633333333333334

Spécificité de notre règle métier: 0.998

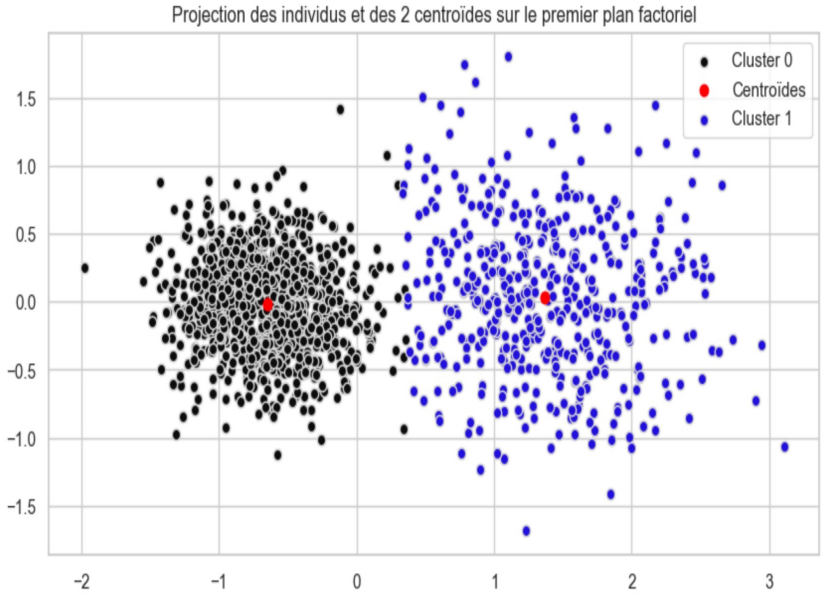
Précision de notre règle métier: 0.9495718363463368

Score F1 de notre règle métier: 0.9731838127742564

5- Classification par Kmeans

L'algorithme de Kmeans nous a permis de partitionner nos données en sous groupes par un apprentissage non supervisé.

Nombre de composantes retenues après standardisation des données : 4 (plus de 80% de la variance expliquée)
Nombre de dimension retenue après le critère de Kaiser: 2
Nombre de clusters optimal après la méthode du coude et de la silhouette 2



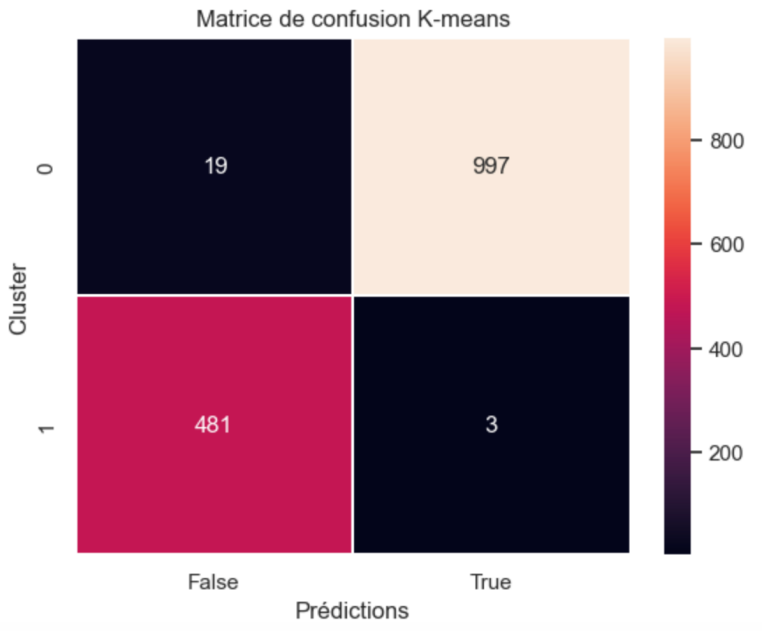
Cluster	diagonal	height_left	height_right	margin_low	margin_up	length	NB billets
0	171.987	103.952	103.813	4.124	3.058	113.196	1016
1	171.898	104.193	104.145	5.238	3.347	111.592	484

Le cluster 0 à les mêmes caractéristiques que les billets ayant 'is_genuine' = **True**

Le cluster 1 à les mêmes caractéristiques que les billets ayant 'is_genuine' = **False**

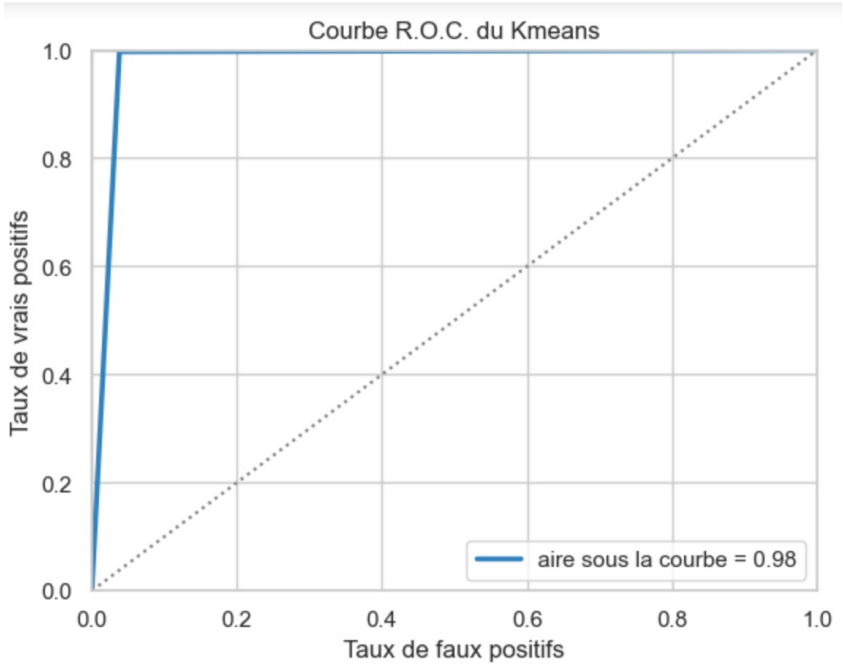
5- Classification par Kmeans

Calcul des performances du kmeans



1016 vrais billets, nous avons 997 vrais positif et 19 faux négatif.

484 faux billets, nous avons 481 vrai négatif et 3 faux positif.



Précision du Kmeans: 0.9853333333333333
Spécificité du Kmeans: 0.9812992125984252
Sensibilité du Kmeans: 0.997
Score F1 du Kmeans: 0.9890873015873016

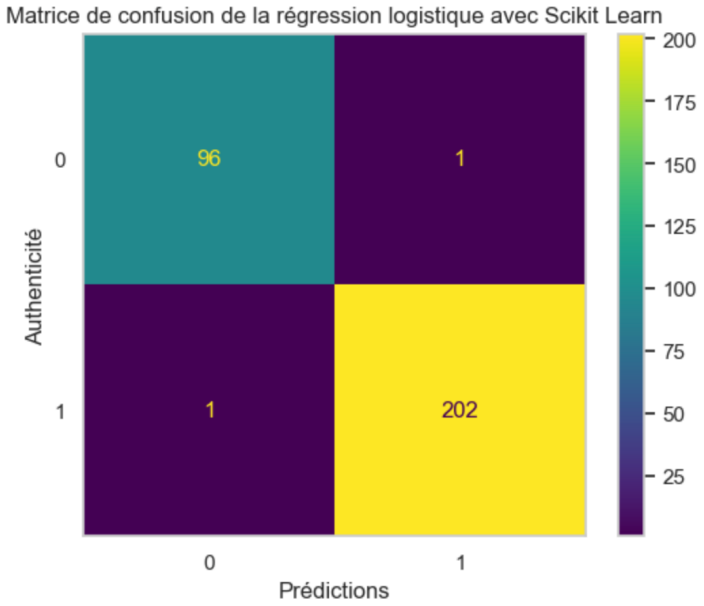
6- Classification par régression logistique

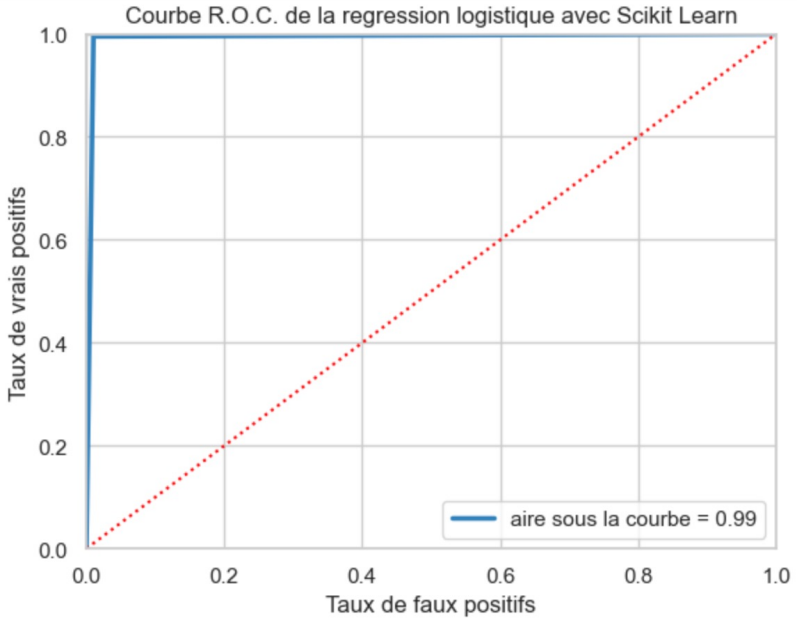
Processus de mise en place de la régression logistique avec Scikit Lean

- Définition des variables X et y
- Découpage de notre jeu de données en un jeu d'entrainement et un jeu de test (80% pour l'entrainement)
- Elimination des valeurs non-significatives avec la librairie RFECV
- Les variables significatives retenues sont 'height_right', 'margin_low','margin_up', 'length'
- Entrainement de notre modèle sur les données de train
- Prédiction des données sur nos données de test

6- Classification par régression logistique

Calcul des performances de notre modèle





Nous avons 203 vrais billets, avec 202 vrai positifs et 1 faux négatif.

Et 97 faux billets, avec 96 vrai négatif et 1 faux positif.

Précision global de la régression : 0.9933333333333333
Spécificité de la régression : 0.9950738916256158
Précision de la régression logistique : 0.9950738916256158
Score F1 de la régression logistique : 0.9950738916256159

7- Choix du modèle final

Score F1 de notre règle métier**:** **0.9731838127742564**

Score F1 du Kmeans : **0.9890873015873016**

Score F1 de la régression logistique : **0.9950738916256159**

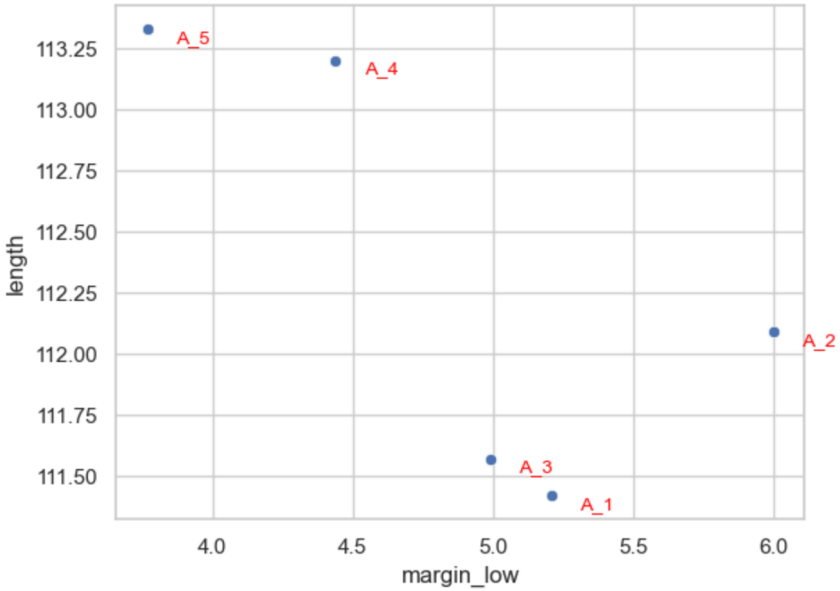
La régression logistique est la solution la plus optimale pour la prédiction

8- Application du modèle à des données inconnues

Aperçu du data set

	diagonal	height_left	height_right	margin_low	margin_up	length	id
0	171.760	104.010	103.540	5.210	3.300	111.420	A_1
1	171.870	104.170	104.130	6.000	3.310	112.090	A_2
2	172.000	104.580	104.290	4.990	3.390	111.570	A_3
3	172.490	104.550	104.340	4.440	3.030	113.200	A_4
4	171.650	103.630	103.560	3.770	3.160	113.330	A_5

Projection de nos données



8- Application du modèle à des données inconnues

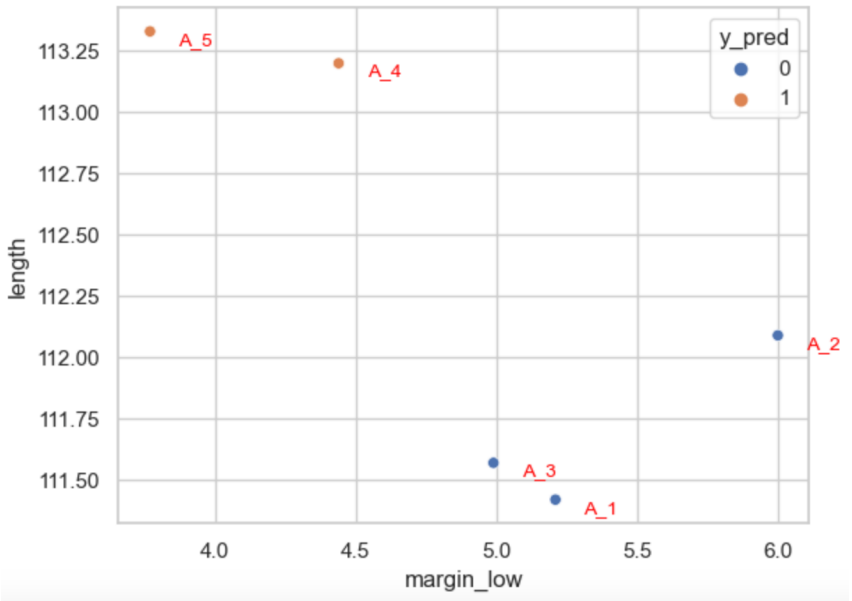
Résultat de la prédiction

	id	y_pred	proba_pred
0	A_1	0	0.008
1	A_2	0	0.002
2	A_3	0	0.003
3	A_4	1	0.947
4	A_5	1	0.999

Identification des billets:

Le billet A_1 est un faux billet
Le billet A_2 est un faux billet
Le billet A_3 est un faux billet
Le billet A_4 **est un billet authentique**
Le billet A_5 **est un billet authentique**

Projection de nos données prédites



9- Conclusion

La mise en place de la régression linaire multiple nous à permit de remplacer les données manquantes, mais nous aurions pu combler les valeurs en question en prenant la moyenne des autres données ou la valeur médiane, nous aurions pu également soit utiliser les valeurs précédentes ou les suivantes.

La régression logistique avec les librairies Scikit Lean et Stat model ont retournés les mêmes résultats.

La performance de la régression logistique était très bonne (Score F1 de 99,5*%), la méthode du Kmeans aurait pu également être utilisée (Score F1 de 98,90%).

Les deux méthodes ont données satisfaction car nous avons pu atteindre les résultats attendus.