Projet 12

DETECTEZ DES FAUX BILLETS AVEC PYTHON

Daniela MENGUI

CONTEXTE

ORGANISATION NATIONALE DE LUTTE CONTRE LE FAUX-MONNAYAGE (ONCFM)

ORGANISATION PUBLIQUE

EN CHARGE DE LA MISE EN PLACE DE METHODES D'IDENTIFICATION DES FAUX BILLETS

MISSION EN DATA ANALYSE POUR DEVELOPPER UN ALGORITHME DE DETECTION DE FAUX BILLETS A PARTIR DES DIMENSIONS DES BILLETS

DETECTION DE CONTREFACON OU FRAUDE : CAS TYPIQUE DE PROJET EN MACHINE LEARNING

METHODOLOGIE

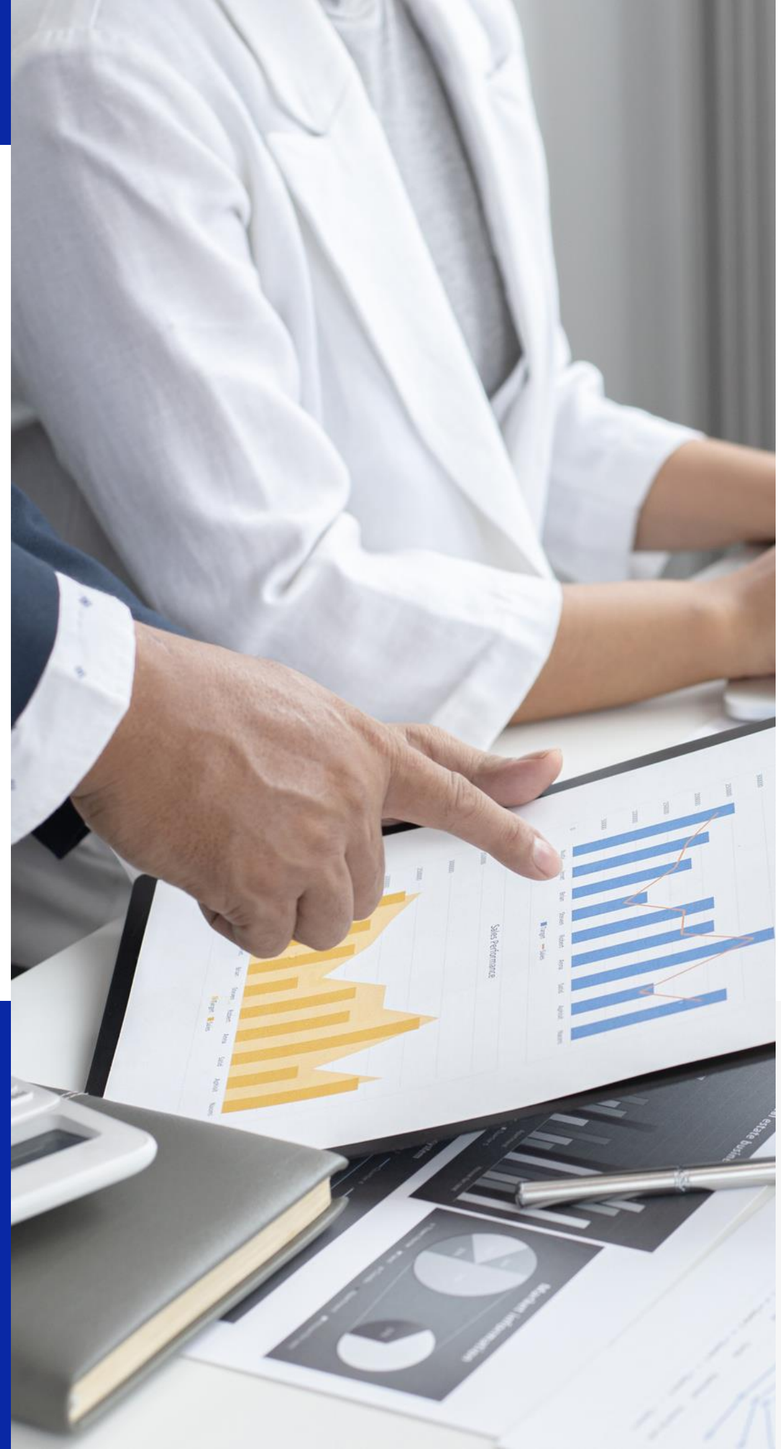
ETAPES DE PROJET EN MACHINE LEARNING

LOGICIEL : JUPYTER NOTEBOOK

DONNEES AU FORMAT CSV

POINTS DE VIGILANCE :

VEILLER A LA BONNE QUALITE DES DONNEES + DES MODELES POUR OBTENIR DES PREDICTIONS LES PLUS PERFORMANTES



SOMMAIRE

1- COLLECTE DES DONNEES

EXTRACTION DES DONNEES A PARTIR DE LEURS SOURCES

1.2 - Chargements des fichiers

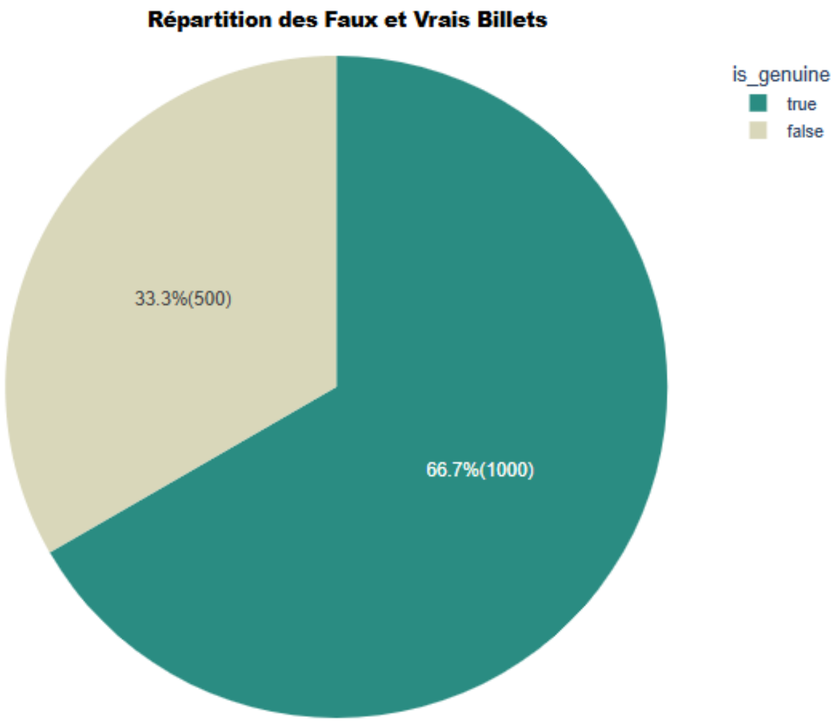
```
:  #Importation du fichier Billet
   df_billets = open('billets.csv')
   data1 = pd.read_csv(df_billets, encoding='utf-8', delimiter=';')
   display(data1)
```

	is_genuine	diagonal	height_left	height_right	margin_low	margin_up	length
0	True	171.81	104.86	104.95	4.52	2.89	112.83
1	True	171.46	103.36	103.66	3.77	2.99	113.09
2	True	172.69	104.48	103.50	4.40	2.94	113.16
3	True	171.36	103.91	103.94	3.62	3.01	113.51
4	True	171.73	104.28	103.46	4.04	3.48	112.54
...	...	...	...	...	...	...	...
1495	False	171.75	104.38	104.17	4.42	3.09	111.28
1496	False	172.19	104.63	104.44	5.27	3.37	110.97
1497	False	171.80	104.01	104.12	5.51	3.36	111.95
1498	False	172.06	104.28	104.06	5.17	3.46	112.25
1499	False	171.47	104.15	103.82	4.63	3.37	112.07

1500 rows × 7 columns

2- ANALYSE EXPLORATOIRE DES DONNEES

OBSERVATION ET COMPREHENSION DU JEU DE DONNEES



1500 LIGNES
7 COLONNES

- 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).

2-1 PRE TRAITEMENT

PREPARATION DES DONNEES AVANT DE LES FOURNIR A LA MACHINE POUR SON APPRENTISSAGE

Inférence	Remplacement des Valeurs manquantes Margin low
Encodage	Conversion des données qualitatives en quantitatives 0 = Vrais Billets / 1 = Faux Billets
Normalisation	Mise à la même échelle des données

2-2 TRAITEMENT DES VALEURS MANQUANTES

PAR LA REGRESSION LINEAIRE

MODELE DE PREDICTION LE + SIMPLE DU MACHINE LEARNING

VERIFICATION DES HYPOTHESES DU TEST :

HOMOSCEDASTICITE
NORMALITE DES RESIDUS
INDEPENDANCE DES RESIDUS

```
:
# Prédiction des valeurs manquantes grâce au modèle
df_billets_na['margin_low'] = reg.predict(
    df_billets_na.drop(columns=['is_genuine', 'margin_low']))

display(df_billets_na)
```

	is_genuine	diagonal	height_left	height_right	margin_low	margin_up	length
72	True	171.94	103.89	103.45	4.323133	3.25	112.79
99	True	171.93	104.07	104.18	4.393907	3.14	113.08
151	True	172.07	103.80	104.38	4.416845	3.02	112.93
197	True	171.45	103.66	103.80	4.337374	3.62	113.27
241	True	171.83	104.14	104.06	4.634614	3.02	112.36
251	True	171.80	103.26	102.82	3.815222	2.95	113.22
284	True	171.92	103.83	103.76	4.190862	3.23	113.29
334	True	171.85	103.70	103.96	4.133982	3.00	113.36

2-3 DATASET TRANSFORME

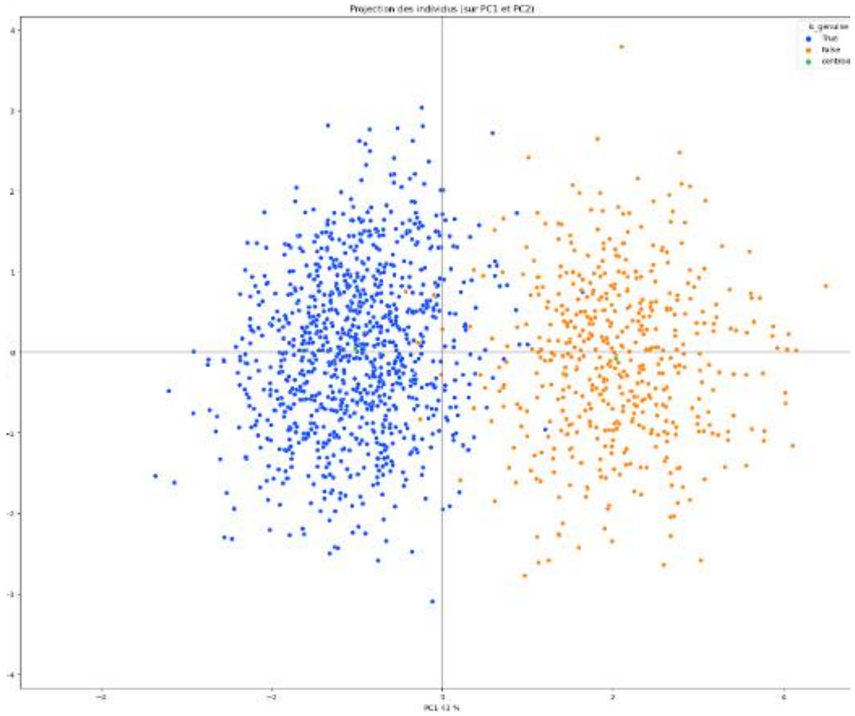
	is_false	diagonal	height_left	height_right	margin_low	margin_up	length
0	0	171.94	103.89	103.45	4.323133	3.25	112.79
1	0	171.93	104.07	104.18	4.393907	3.14	113.08
2	0	172.07	103.80	104.38	4.416845	3.02	112.93
3	0	171.45	103.66	103.80	4.337374	3.62	113.27
4	0	171.83	104.14	104.06	4.634614	3.02	112.36

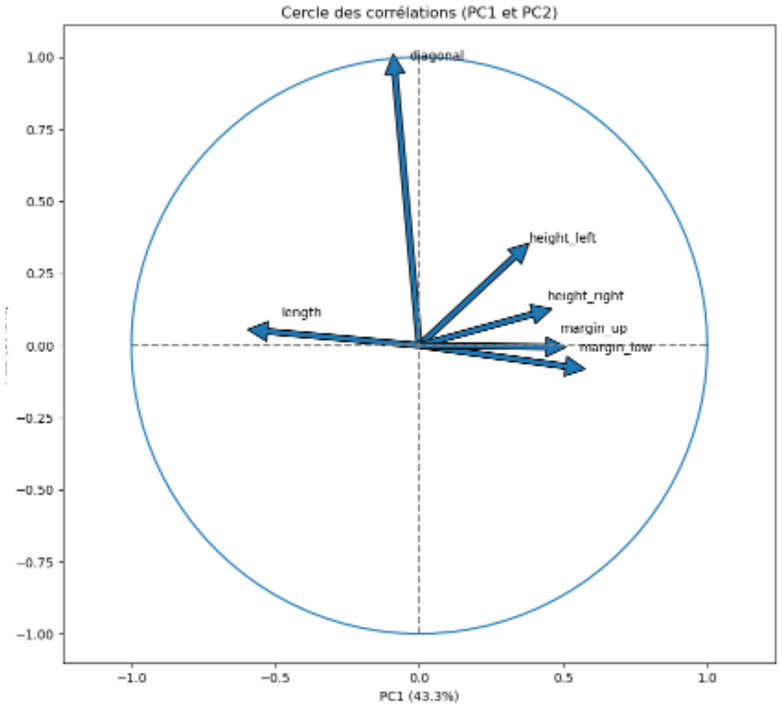
- **Aucune valeur manquante**

2-4 PROJECTION DES DONNEES

PAR ANALYSE EN COMPOSANTES PRINCIPALES (ACP)

TECHNIQUE POUR FACILITER LA REPRESENTATION GRAPHIQUE D'UN DATASET EN CREANT DE NOUVELLES COLONNES QUI SYNTHETISENT LES 6 AUTRES





3- SELECTION DES VARIABLES PERTINENTES

REPERAGE DES VARIABLES QUI VONT LE + INFLUER DANS NOTRE MODELE

MODELE DE MACHINE LEARNING

Y : is_false

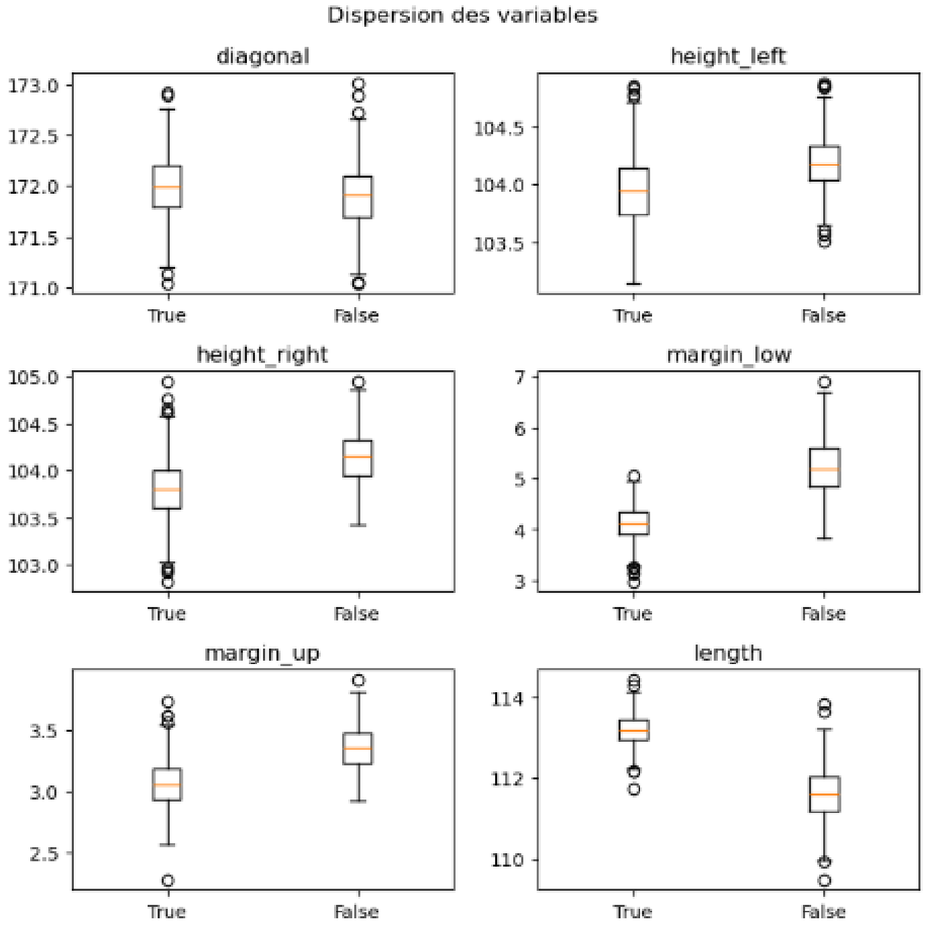
Variable cible que l'on cherche à prédire

X : 6 autres variables

Variables qui viennent influencer Y

3- SELECTION DES VARIABLES PERTINENTES

DIAGRAMME EN BOITE

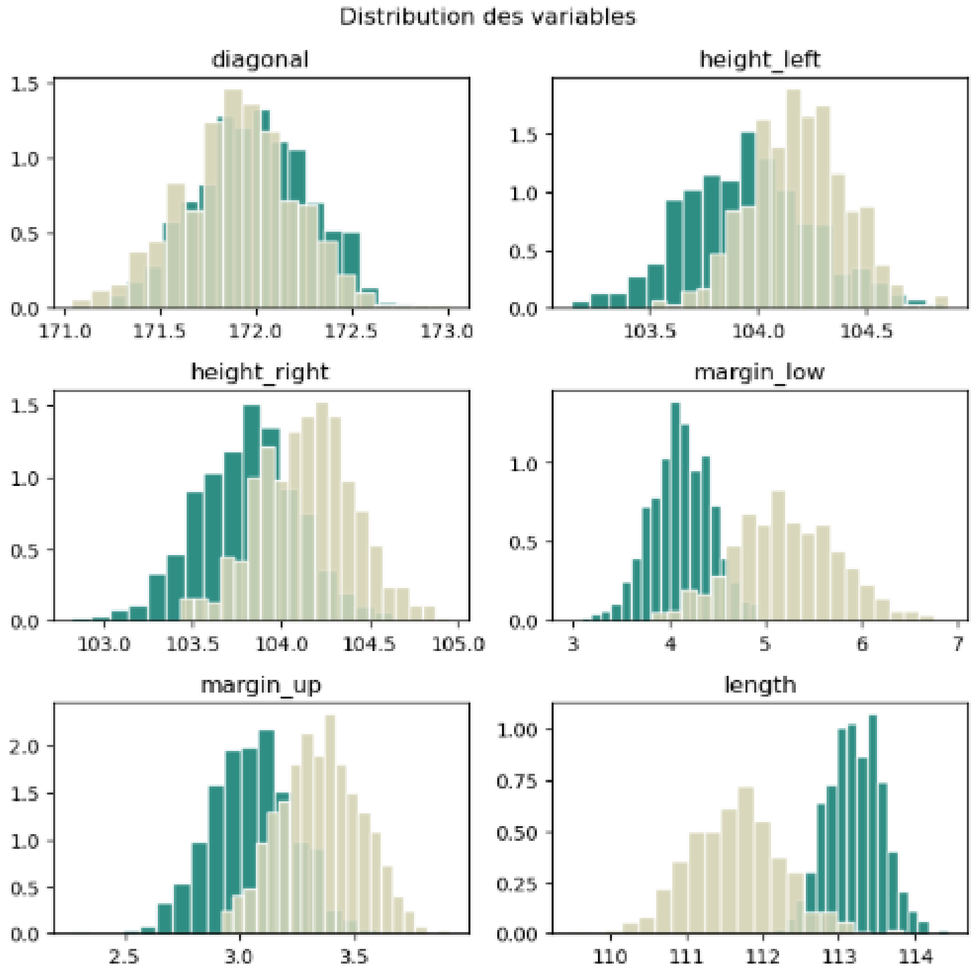


Mise en évidence de la répartition des données par variable + les outliers

Ce sont les variables margin_low (1.06 de différence) et lenght (1.575 de différence) qui enregistrent les plus grands écarts en termes de médiane entre les vrais/faux billets

3- SELECTION DES VARIABLES PERTINENTES

HISTOGRAMME



Les distributions des Faux Billets en comparaison des Vrais billets sont plus décalées sur :

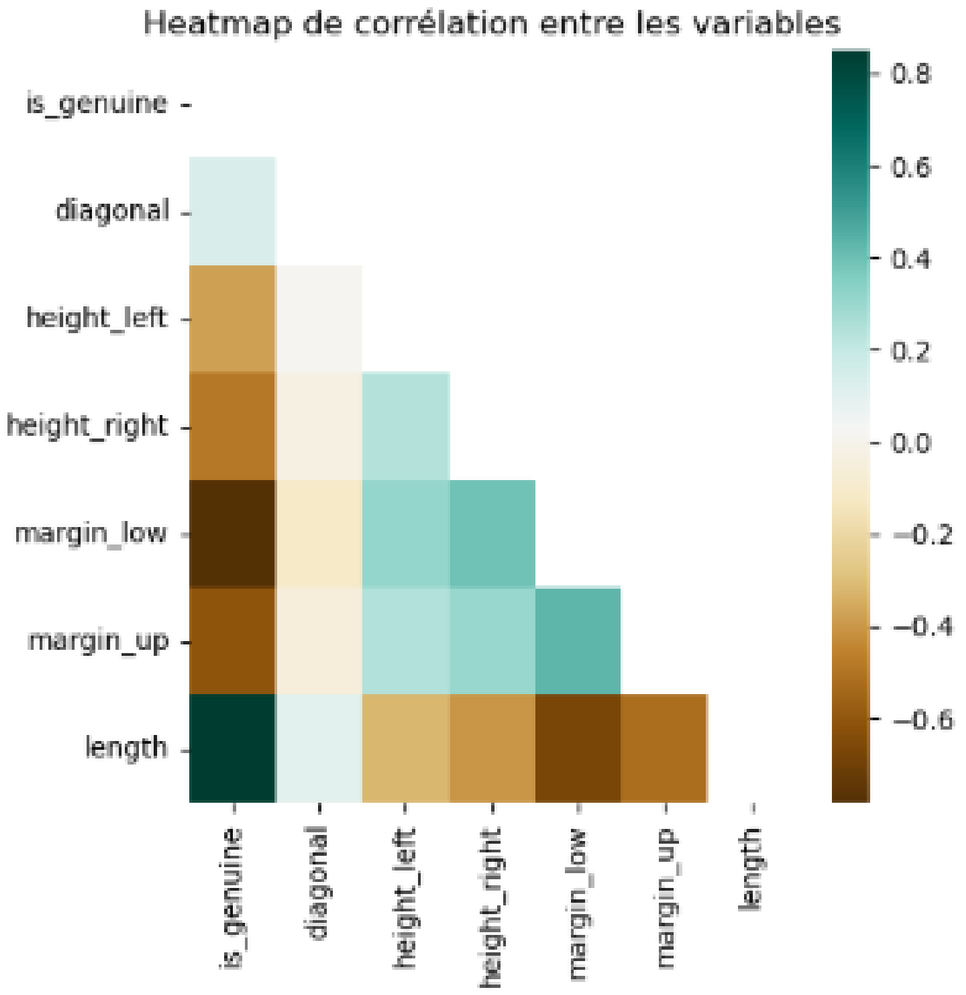
- **length**
- **margin low**
- **margin up**

et le sont moins sur :

- **diagonal**

3- SELECTION DES VARIABLES PERTINENTES

HEATMAP DE CORRELATION



La variable cible is_genuine est correlée très fortement avec :

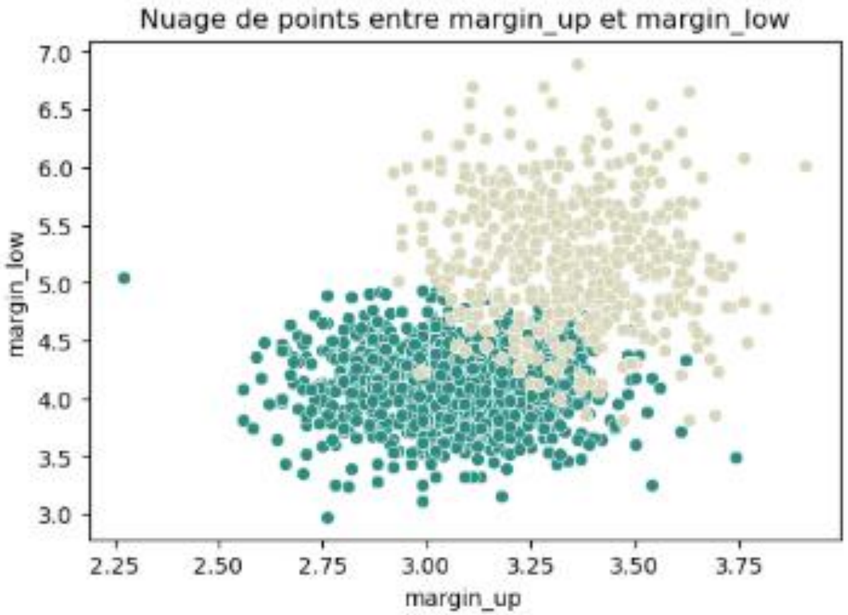
- **length(0.85)**
- **margin low (0.78)**
- **margin up (0.61)**

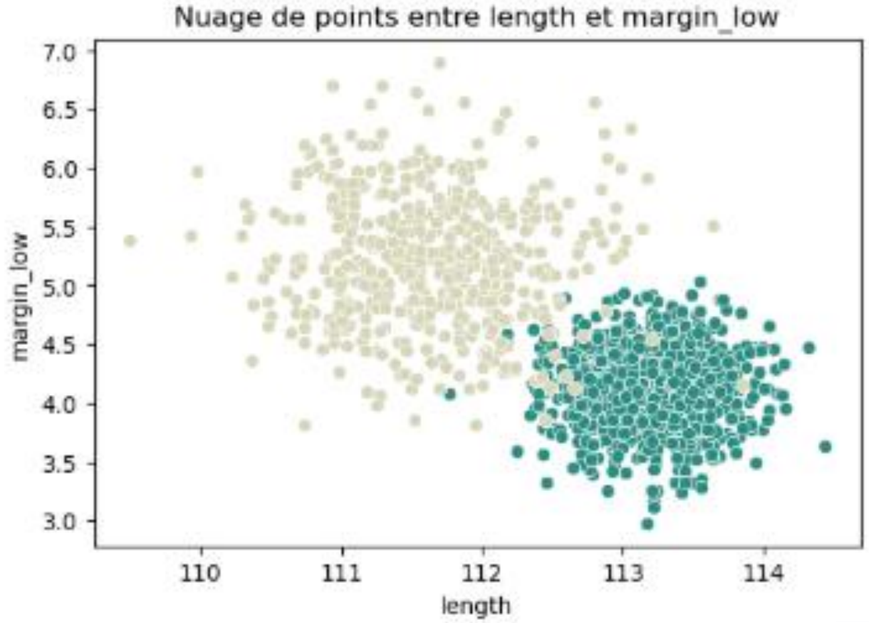
et à l'inverse très faiblement avec diagonal (0.13)

On observe également que Margin low est fortement corrélée à length (-0.67)

3- SELECTION DES VARIABLES PERTINENTES

NUAGE DE POINTS





- **Plus le margin up& margin low augmentent, et plus les billets sont faux**
- **Plus le lenght augmente et plus les billets sont vrais**

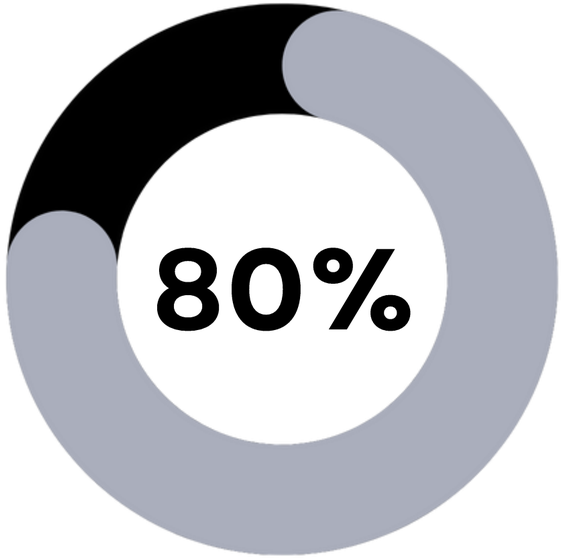
4- ENTRAINEMENT DES MODELES

L’objectif de cette partie est d’entrainer le jeu de données sur 4 Modèles, et ensuite de choisir le meilleur grâce à des scores de Performance

Division du Dataset (split)

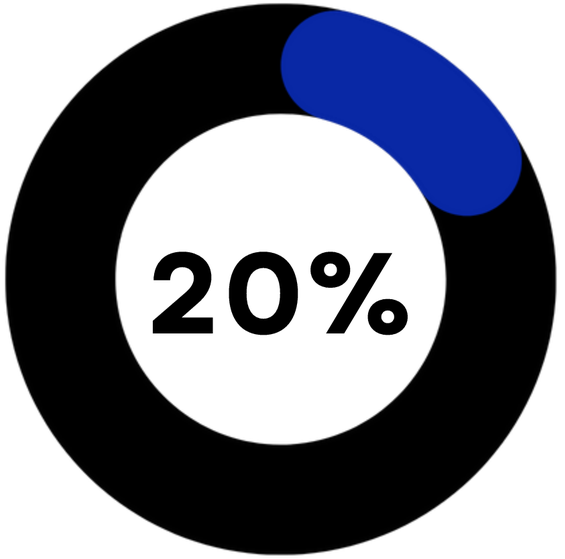
Partie Entrainement

1200 Lignes



Partie Test

300 Lignes



X TRAIN

Y TRAIN

X TEST

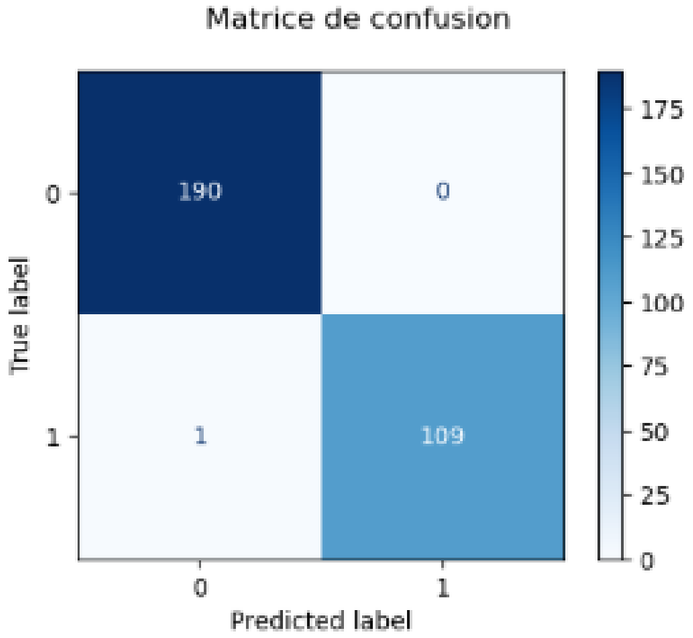
Y TEST

REGRESSION LOGISTIQUE

1/ Définition : La régression logistique est un modèle statistique permettant de prédire la probabilité qu'un événement arrive (valeur de 1) ou non (valeur de 0)

```
# Régression Logistique
clf_log_reg = LogisticRegression().fit(X_train, y_train)

# Prédiction sur les données test
y_pred = clf_log_reg.predict(X_test)

# Affichage des coefficients de la régression
for column, coef in zip(X.columns, clf_log_reg.coef_[0]):
    print(f'{column} : \t{coef:>9.6f}')
```

2/ Evaluation du Modèle

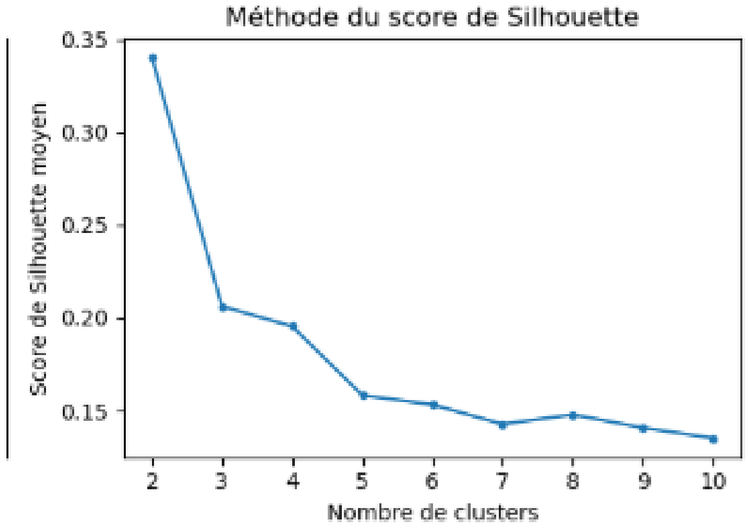


La matrice de confusion dit ceci :

- Les vrais positifs : pour 109 billets, nous avons correctement prédit qu'ils sont faux.
- Les vrais négatifs : pour 190 billets, nous avons correctement prédit qu'ils sont authentiques.
- Les faux positifs : absent.
- Les faux négatifs : pour 1 billet, nous avons prédit à tort qu'il est authentique.

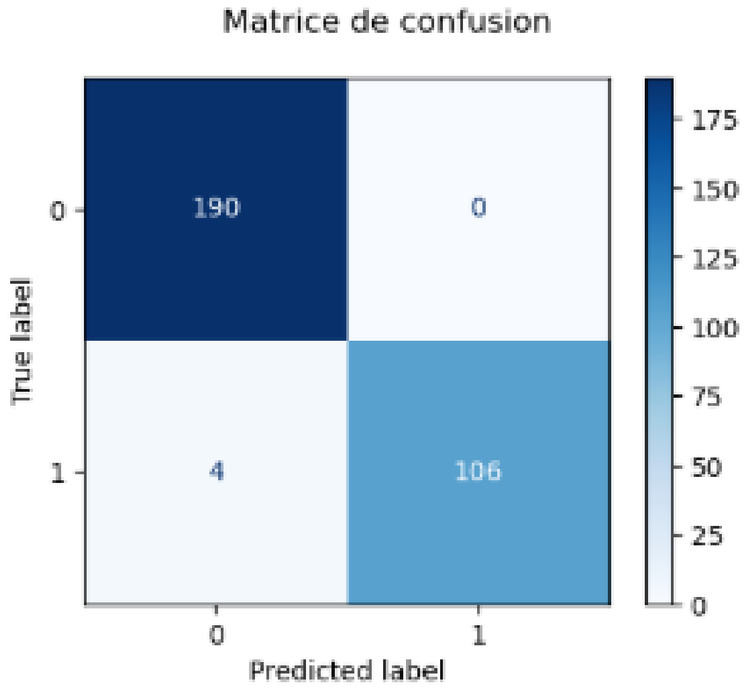
K MEANS

1/ Définition : Le clustering k-means est un algorithme d'apprentissage non supervisé utilisé dans le partitionnement de données, qui regroupe les points de données non étiquetés en groupes, ou clusters




```
# Le clustering est réalisé avec 2 clusters
clf_kmeans = KMeans(n_clusters=2, init='k-means++', n_init='auto',
                    random_state=42)
clf_kmeans.fit(X_train_scaled)
```

2/ Evaluation du Modèle

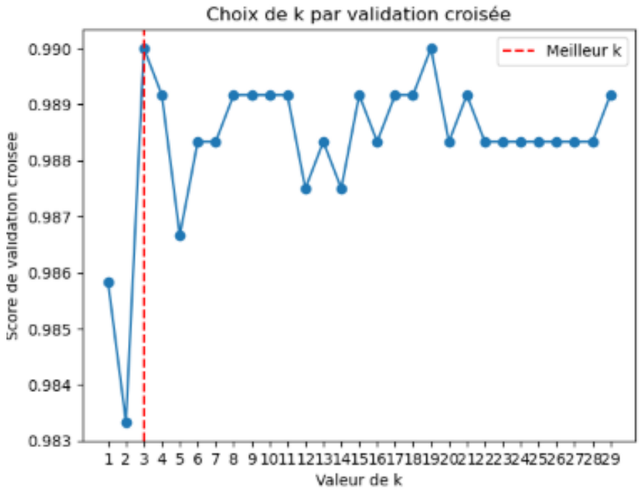


La matrice de confusion dit ceci :

- Les vrais positifs : pour 106 billets, nous avons correctement prédit qu'ils sont faux.
- Les vrais négatifs : pour 190 billets, nous avons correctement prédit qu'ils sont authentiques.
- Les faux positifs : absent.
- Les faux négatifs : pour 4 billets, nous avons prédit à tort qu'il sont authentiques.

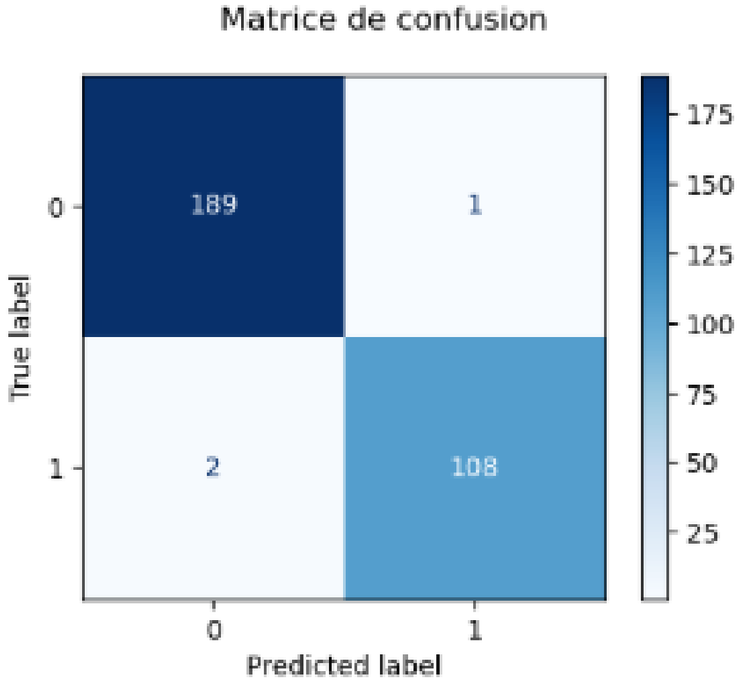
KNN

1/ Définition : type d'apprentissage supervisé qui peut être utilisé pour la classification et la régression. KNN fonctionne en trouvant les 'K' échantillons les plus proches dans l'ensemble de données d'apprentissage d'un nouvel échantillon non classé. .




```
# Recherche de grille avec validation croisée
grid_search = GridSearchCV(knn, param_grid, cv=5)
grid_search.fit(X_train, y_train)

# Extraction des scores de validation croisée et des valeurs de k correspondantes
cv_scores = grid_search.cv_results_['mean_test_score']
best_k = grid_search.best_params_['n_neighbors']
print(best_k)
print(grid_search.best_score_)
```


2/ Evaluation du Modèle



La matrice de confusion dit ceci :

- Les vrais positifs : pour 108 billets, nous avons correctement prédit qu'ils sont faux.
- Les vrais négatifs : pour 189 billets, nous avons correctement prédit qu'ils sont authentiques.
- Les faux positifs : pour 1 billet, nous avons prédit à tort qu'il est faux.
- Les faux négatifs : pour 2 billets, nous avons prédit à tort qu'ils sont authentiques.

RANDOM FOREST

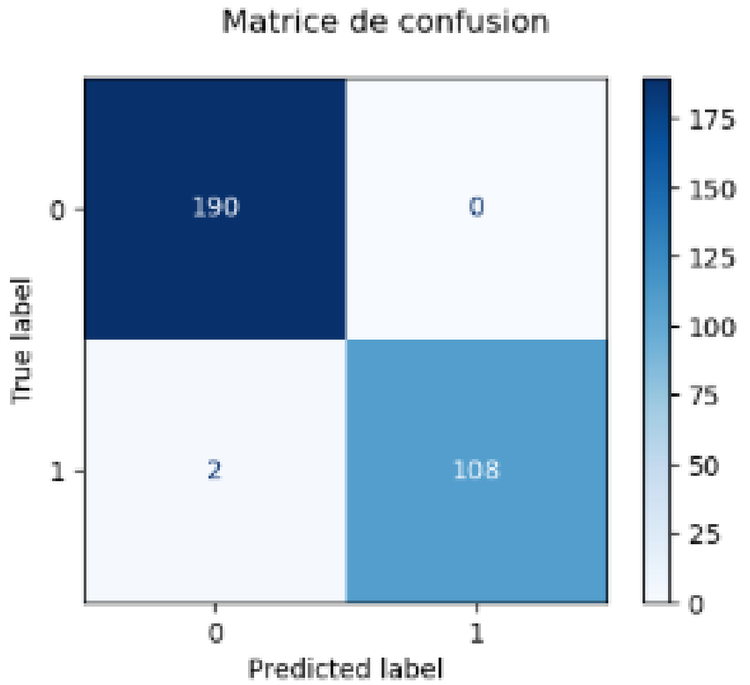
1/ Définition : La forêt d'arbres décisionnels est un algorithme de machine learning couramment utilisé, qui combine les résultats de plusieurs arbres de decision pour obtenir un résultat unique.

```python
# Les hyperparamètres à tester à l'aide de GridSearch
param_grid = {
    'n_estimators': [50,100,150,200,250,300,350,400],
    'max_depth': [1,2,3,4,6,7,8,9,10]
}

# Création d' un modèle de forêt aléatoire
rf_model = RandomForestClassifier(oob_score=True, random_state=42)

# GridSearch pour trouver les meilleurs paramètres
grid_search = GridSearchCV(rf_model, param_grid, cv=5)
grid_search.fit(X_train, y_train)

# Les meilleurs paramètres trouvés
print("Meilleurs paramètres :", grid_search.best_params_)
```

```
Meilleurs paramètres : {'max_depth': 6, 'n_estimators': 50}
```

```python
# Prédire les classes sur les ensembles d'entraînement et de test
y_train_pred = random_forest.predict(X_train)
y_test_pred = random_forest.predict(X_test)
```

2/ Evaluation du Modèle



La matrice de confusion dit ceci :

- Les vrais positifs : pour 108 billets, nous avons correctement prédit qu'ils sont faux.
- Les vrais négatifs : pour 190 billets, nous avons correctement prédit qu'ils sont authentiques.
- Les faux positifs : aucun.
- Les faux négatifs : pour 2 billets, nous avons prédit à tort qu'ils sont authentiques.

5- CHOIX DU MODELE

Modèle de Rgéression logistique :

- **Meilleurs résultats sur la totalité des métriques de performance**
- **Meilleur pourcentage de billets bien classés (accuracy)**
- **Meilleure capacité à ne pas se tromper lorsqu'elle identifie de faux billets (precision)**
- **La plus forte capacité à repérer les faux billets(recall).**

RECAPITULATIF

	accuracy	precision	recall	f1-score
logistic regression	0.996667	1.000000	0.990909	0.995434
k-means	0.986667	1.000000	0.963636	0.981481
knn + gridsearch cv	0.990000	0.990826	0.981818	0.986301
random forest + gridsearch cv	0.993333	1.000000	0.981818	0.990826

5- ALGORITHME DE DETECTION AUTOMATIQUE DE FAUX BILLETS

VOIR NOTEBOOK

THANK YOU