ONCFM

DÉTECTION DE FAUX BILLETS

Projet 10

PARCOURS DATA ANALYST_V2

ADELINE LE RAY

ORDRE DU JOUR

CONTEXTE

ANALYSE DES DONNÉES

TRAITEMENT DES VALEURS MANQUANTES

CONSTRUCTION DE L'ALGORITHME DE CLASSIFICATION

MODÈLE FINAL

ONCFM

Organisation nationale de lutte contre le faux-monnayage

RETEX ONCFM

Différences de dimensions observées entre les vrais et faux billets

DONNEES

Caractéristiques géométriques des billets relevées par une machine

MISSION

Construire un algorithme capable de définir automatiquement si le billet est vrai ou faux à partir des données géométriques

ANALYSE DES DONNÉES

L'objectif de l'analyse descriptive et exploratoire est d'en apprendre plus sur les données, de les comprendre mais aussi de répondre à la question :

Qu'est-ce qu'un faux billet ?

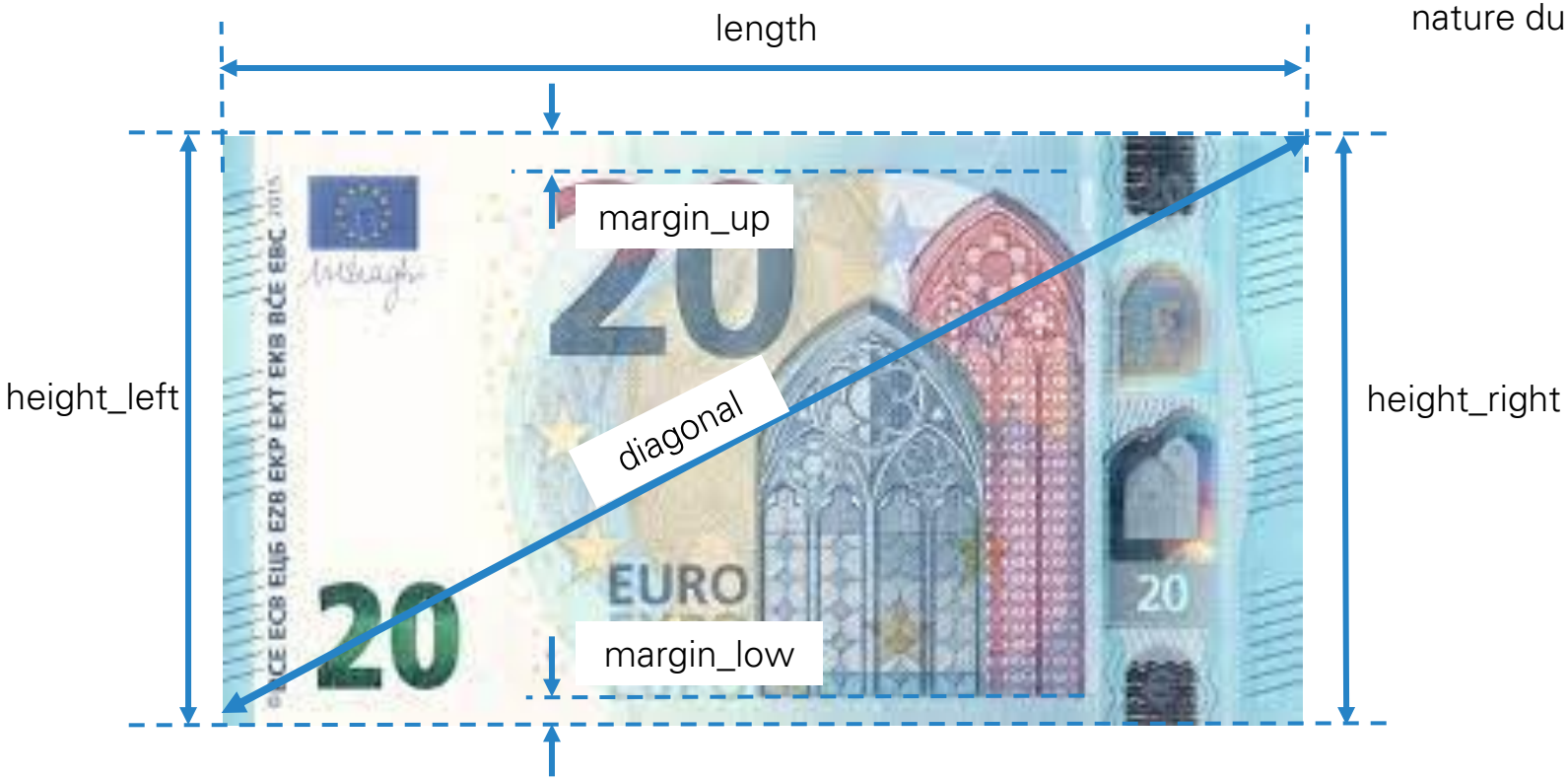
DONNÉES DE PARAMÉTRISATION

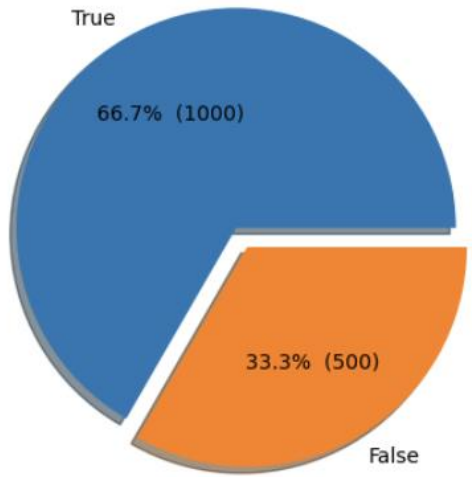
1500 lignes

	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

1 qualitative binaire : nature du billet

6 quantitatives : dimensions géométriques en mm
37 valeurs manquantes ('margin_low')



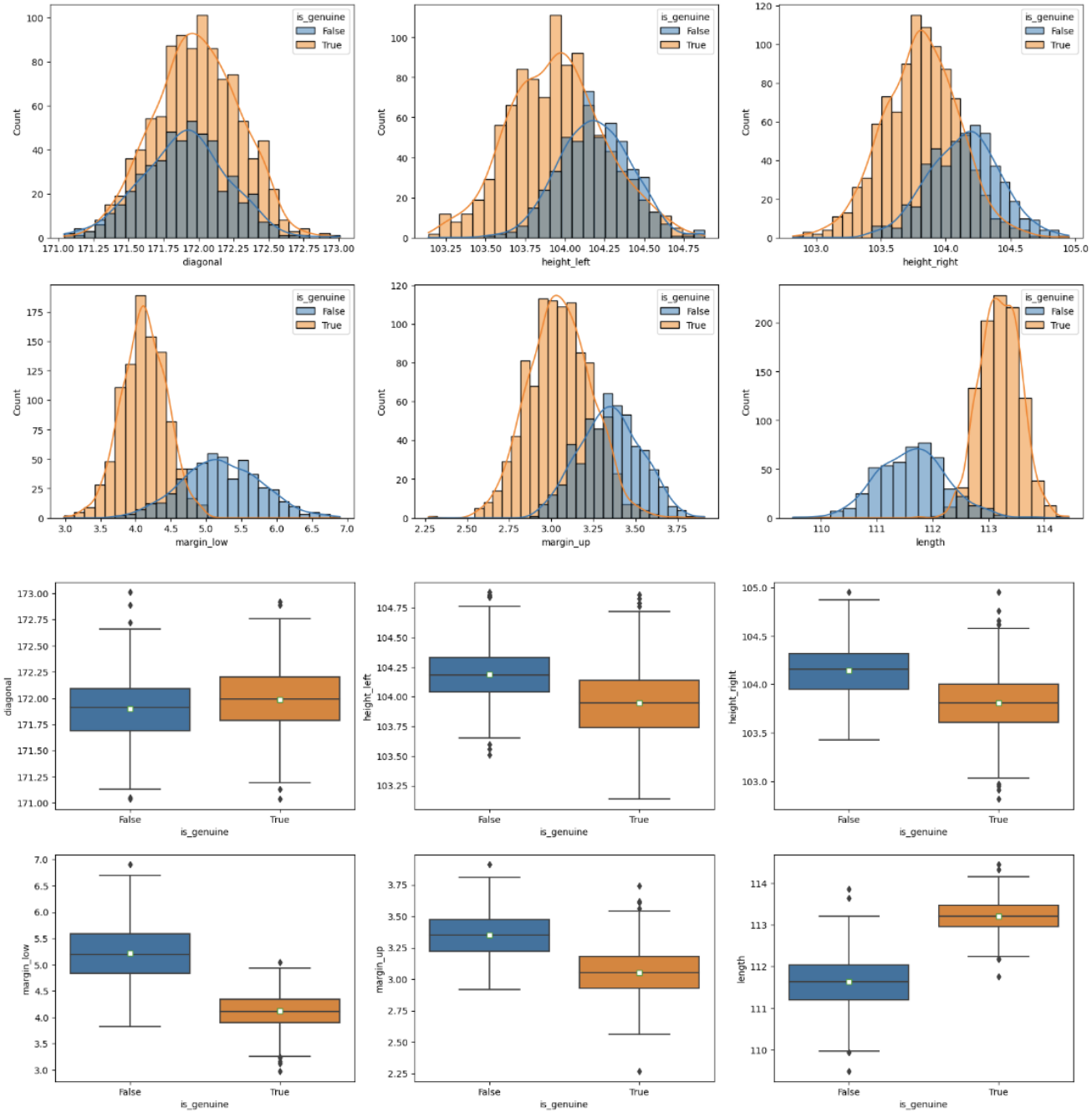


VARIABLES LES + DISCRIMINANTES : LENGTH, MARGIN_LOW, MARGIN_UP

	VRAI BILLET	FAUX BILLET
diagonal	=	=
height_left	-	+
height_right	-	+
margin_low	- - -	+ + +
margin_up	- -	+ +
length	+ + +	- - -

= similaire, + plus grand, - plus petit

Faux billet *Vrai billet*

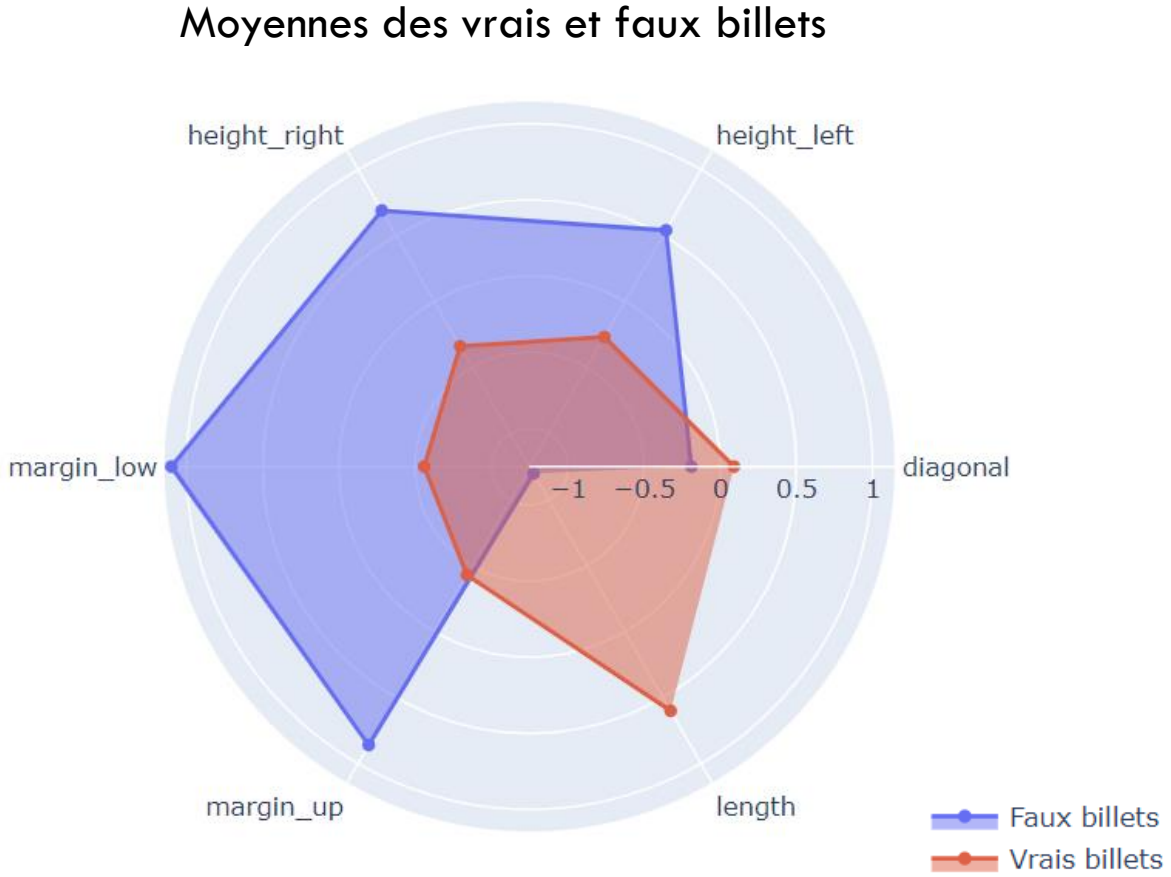


UN LIEN EXISTE BIEN ENTRE NATURE DU BILLET ET DIMENSIONS GÉOMÉTRIQUES

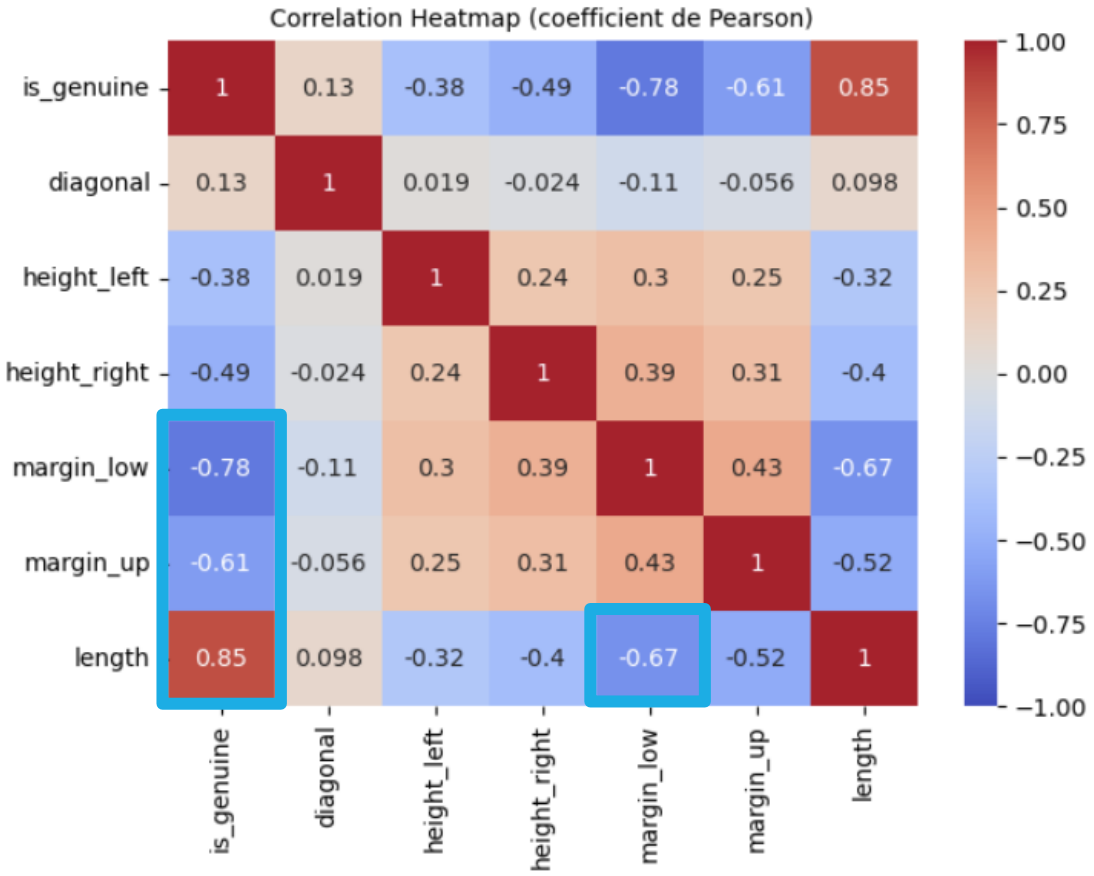
- Test statistique : t-test de Student ou de Welch
- Hypothèse H0 : les moyennes des vrais et faux billets sont égales
- H0 acceptée p-value > 0,05

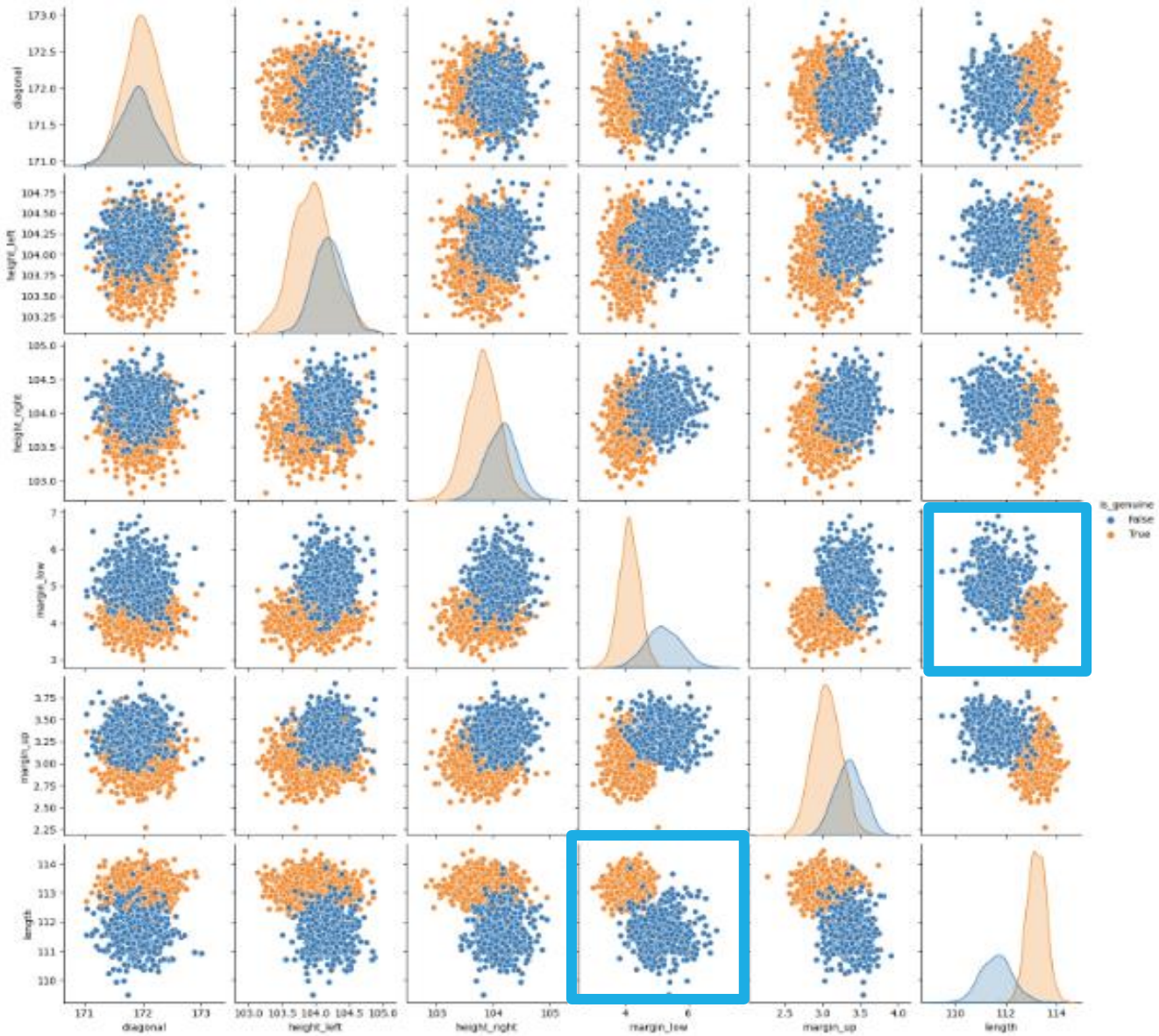
	H0
diagonal	×
height_left	×
height_right	×
margin_low	×
margin_up	×
length	×

✓ : H0 acceptée, × : H0 rejetée

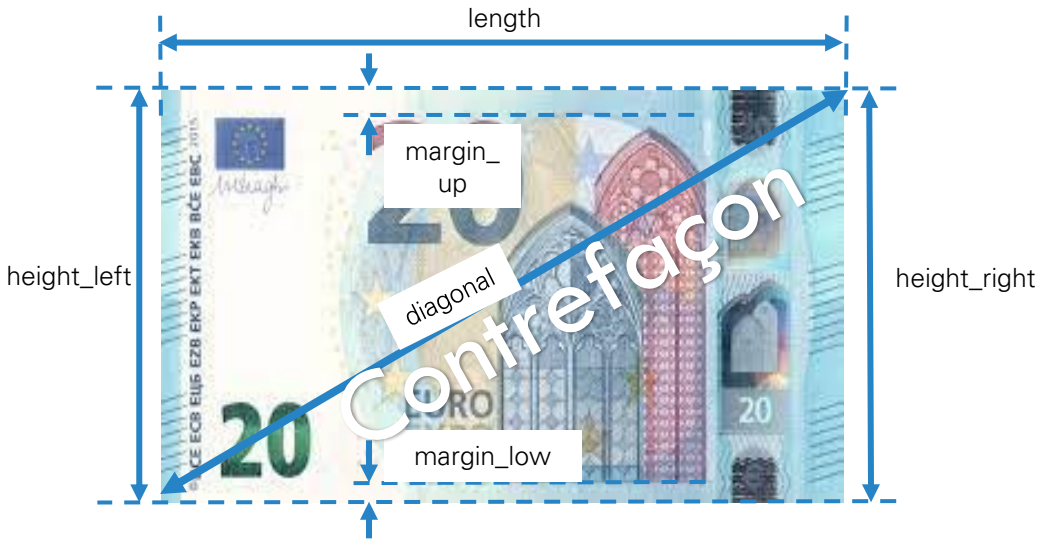


FORTES CORRÉLATIONS





QU'EST-CE QU'UN FAUX BILLET?



Par rapport à un vrai billet

Faux billet

diagonal	Low	High
height_left	Low	High
height_right	Low	High
margin_low	Low	High
margin_up	Low	High
length	Low	High

TRAITEMENT DES VALEURS MANQUANTES

Valeurs manquantes :

- Variable : ‘margin_low’
- Nombre : 37 soit 2,5% du dataset

COMMENT TRAITER LES VALEURS MANQUANTES ?

- Imputation ou suppression?

	+	-
Imputation	Pas de perte d'informations	Risque d'erreurs d'approximation
Suppression	Pas de risque d'erreur d'approximation	Perte d'informations

⇒ choix : remplacement des valeurs manquantes
/!\ Vigilance sur l'impact de l'imputation

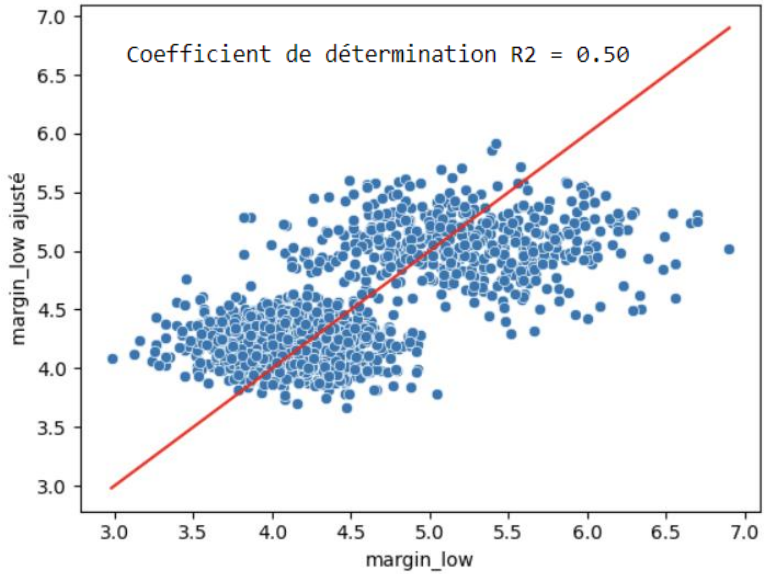
- Solutions envisagées pour l'imputation
  - Remplacement par la moyenne
  - Régression linéaire simple et multiple

RÉGRESSION LINÉAIRE MULTIPLE

- Variable à expliquer : ‘margin_low’
- Variables explicatives: ‘diagonal’, ‘height_left’, ‘height_right’, ‘margin_up’, ‘length’
- Non retenue : ‘is_genuine’, ce n’est pas une variable géométrique et si le modèle doit resservir dans le futur, cette variable ne sera pas disponible.

Echantillonnage stratifié	Entraînement du modèle	Evaluation du modèle	Analyses des résultats
▪ 80% training set / 20% testing set ▪ Stratification : nature du billet	▪ Toutes les variables sont significatives ▪ R2 : 0,47	▪ R2 = 0,5 ▪ Le modèle explique 50% de la variation totale.	▪ Colinéarité ▪ Analyse des résidus ▪ Atypicité et influence

margin_low ajusté en fonction de margin_low



ANALYSES DES RÉSULTATS

Analyses des résidus :

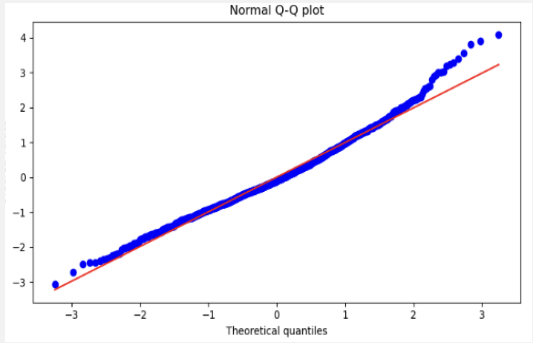
- Les résidus sont indépendants ✓
- La distribution des résidus suit une loi normale ✓
- La variance des résidus n'est pas homogène : hétéroscédasticité ✗

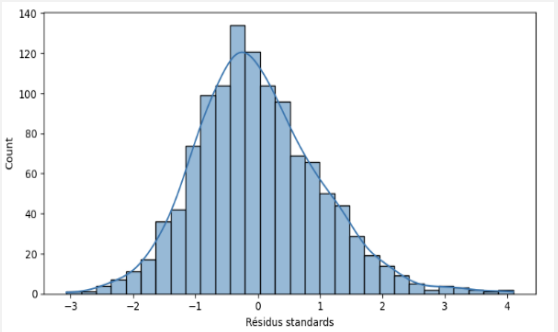
Variables explicatives non colinéaires ✓

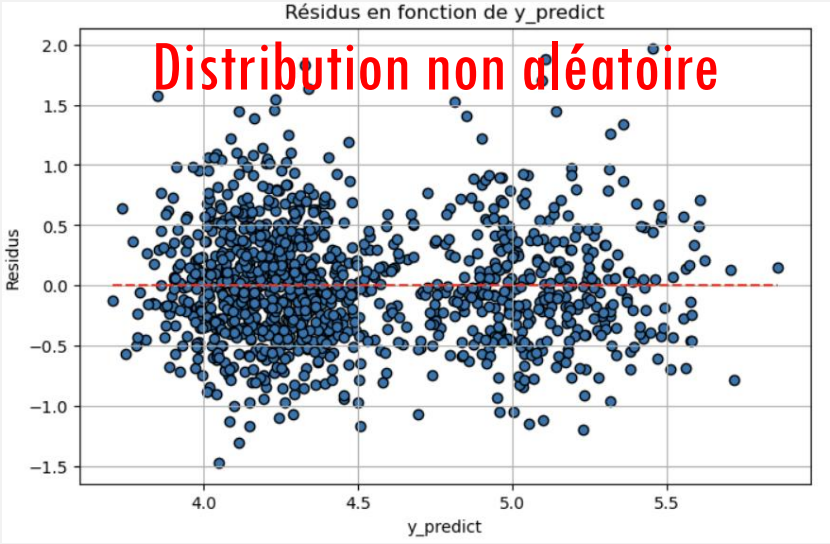
Atypicité et influence : *test réalisé sans les observations atypiques et influentes : pas d'amélioration du modèle => observations conservées*

=> Le modèle sera malgré tout utilisé pour le remplacement des valeurs manquantes

Nb Données >> 30 ➡ Normalité ✓







PISTES EXPLORÉES POUR LA CONSTRUCTION DU MODÈLE

- Algorithmes testés :
  - Régression logistique
  - Méthode du k-means
  - Decision tree (*)
  - Random Forest (*)
  - k-NN (*)

- Influence de l'imputation des valeurs manquantes également testée (*)

(*) : voir annexes

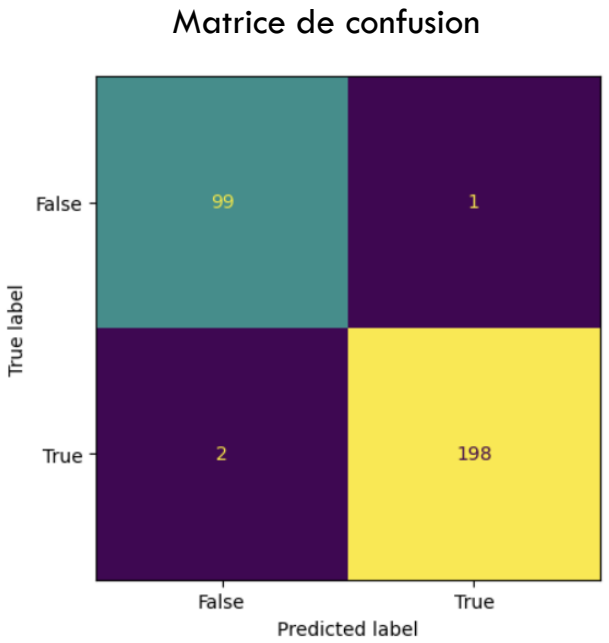
COMMENT CONSTRUIRE LE MODÈLE ?

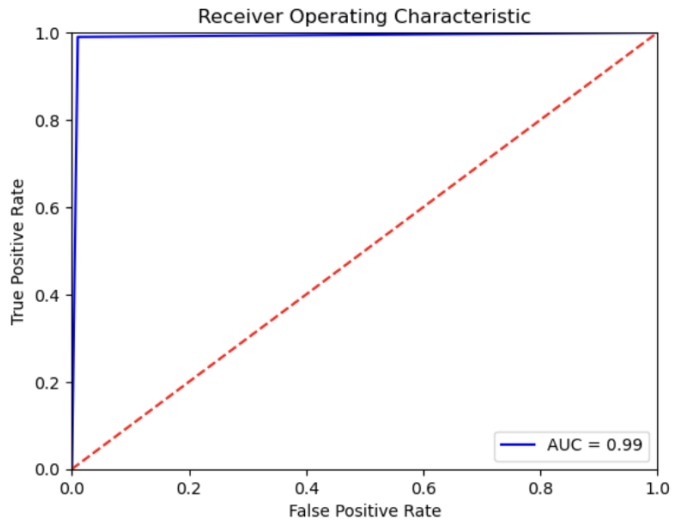
Echantillonnage stratifié	Standardisation des données	Sélection du modèle	Entraînement du modèle	**Evaluation du modèle**
▪ 80% training set / 20% testing set ▪ Stratification sur la nature du billet	Les données sont standardisées pour avoir une échelle commune.	Sélection des hyperparamètres (solver pour regression logisitique, k nombre de clusters pour k-means)	▪ Détermination des variables significatives ▪ Entraînement du modèle sur le training set	▪ Matrice de confusion ▪ Performances : accuracy, precision sensibilité, spécificité, courbe ROC et AUC

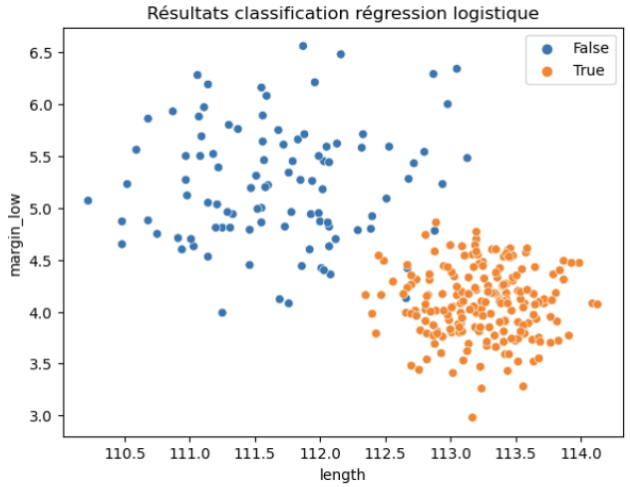
RÉGRESSION LOGISTIQUE BINOMIALE

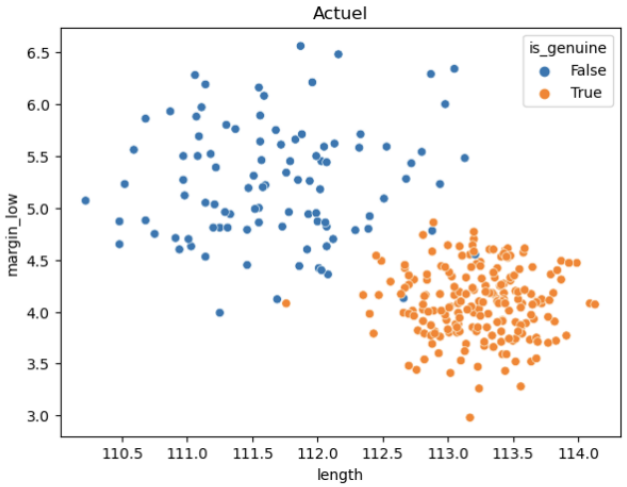
- Algorithme d'apprentissage supervisé utilisé pour la classification binaire
- Application dans notre cas :
  - Variable cible 'is_genuine' qualitative avec 2 modalités, vrai /faux
- Variables significatives :
  - Height_right, margin_low, margin_up, length
- Evaluation des performances :

Accuracy	Précision	Sensibilité	Spécificité	f1_score	R2 score	AUC
0.99	0.995	0.99	0.99	0.992	0.955	0.99



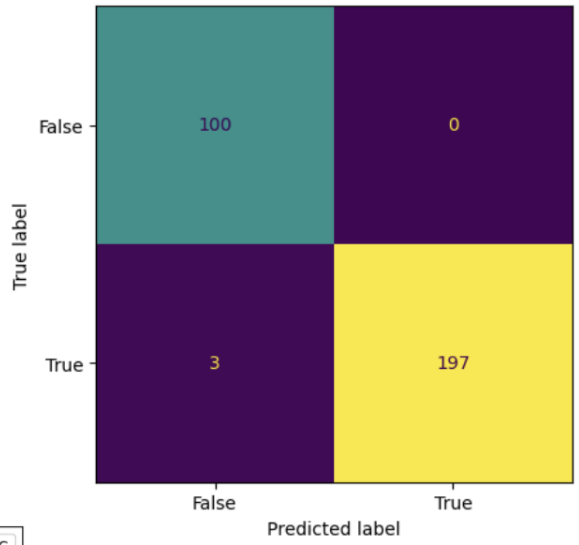




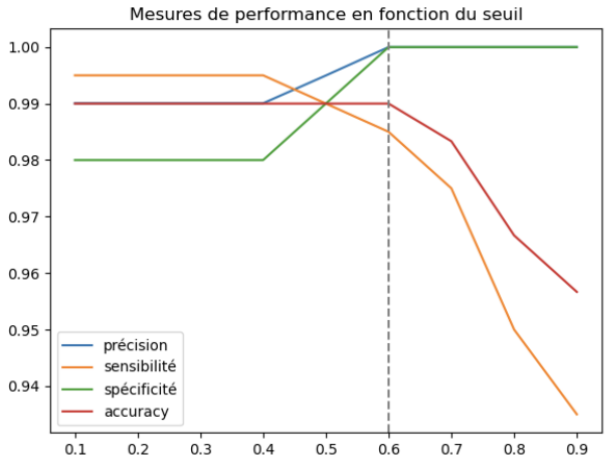


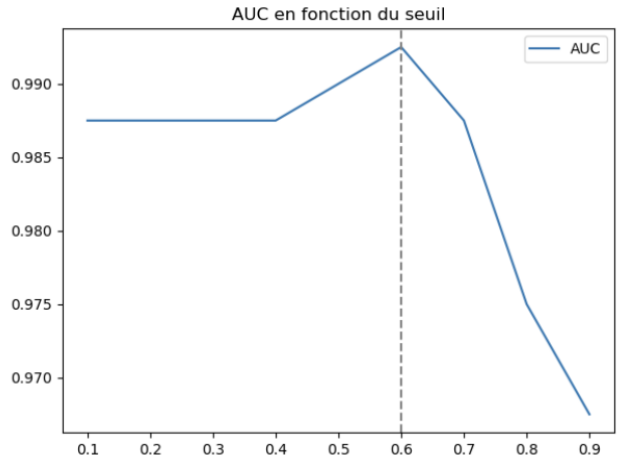
AMÉLIORATION DU MODÈLE EN CHANGEANT LE SEUIL

	thresholds	accuracy	precision	sensibilite	specificite	f1-score	AUC
0	0.1	0.990000	0.990050	0.995	0.98	0.992519	0.9875
1	0.2	0.990000	0.990050	0.995	0.98	0.992519	0.9875
2	0.3	0.990000	0.990050	0.995	0.98	0.992519	0.9875
3	0.4	0.990000	0.990050	0.995	0.98	0.992519	0.9875
4	0.5	0.990000	0.994975	0.990	0.99	0.992481	0.9900
5	0.6	0.990000	1.000000	0.985	1.00	0.992443	0.9925
6	0.7	0.983333	1.000000	0.975	1.00	0.987342	0.9875
7	0.8	0.966667	1.000000	0.950	1.00	0.974359	0.9750
8	0.9	0.956667	1.000000	0.935	1.00	0.966408	0.9675



Tous les faux billets sont prédis faux !

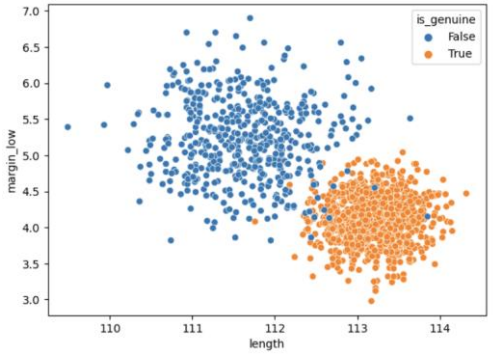




Accuracy	Précision	Sensibilité	Spécificité	f1_score	R2 score	AUC
0.99	1.0	0.985	1.0	0.992	0.955	0.9925

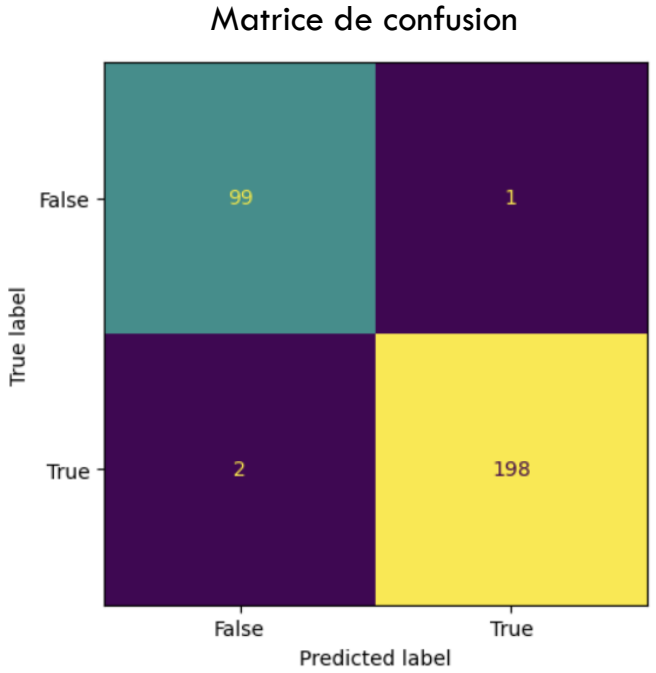
MÉTHODE DES K-MEANS

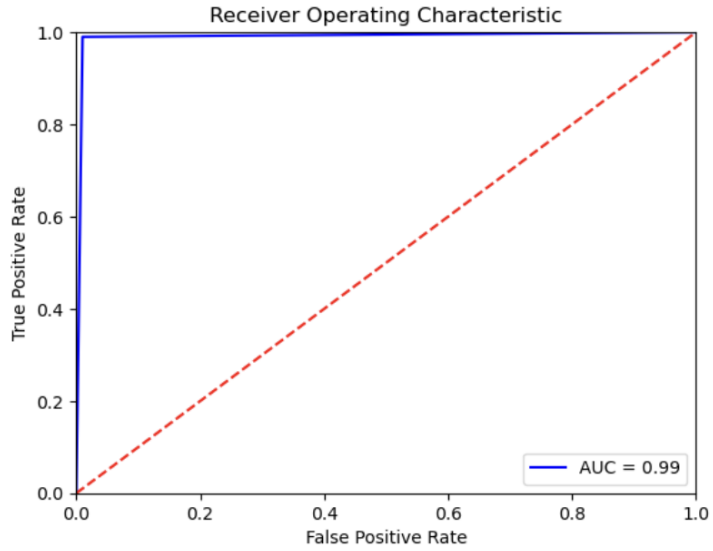
- Algorithme d'apprentissage non supervisé utilisé pour le clustering
- Application dans notre cas :
  - Nous observons bien 2 clusters distincts

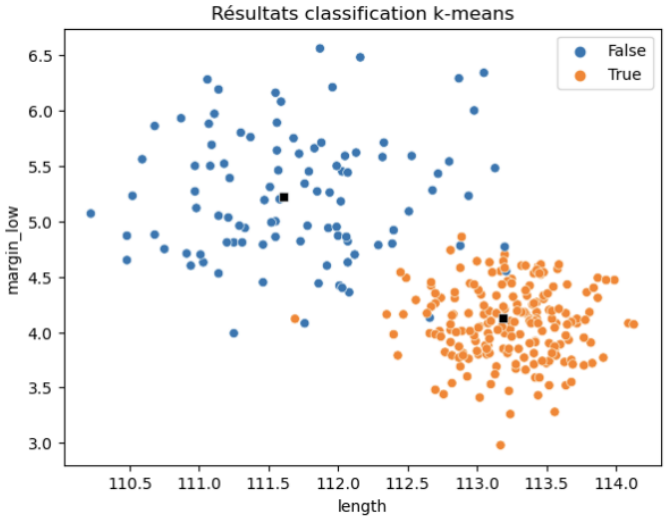


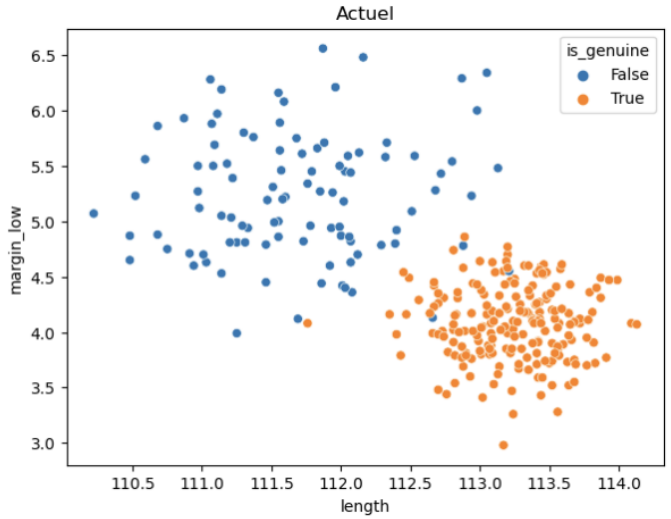
  - Nombre de clusters : k=2 *(méthode du coude, coefficient de silhouette)*
  - Utilisation des centroïdes pour prédire la classe
- Evaluation des performances :

Accuracy	Précision	Sensibilité	Spécificité	f1_score	R2 score	AUC
0.99	0.995	0.99	0.99	0.992	0.955	0.99









QUEL MODÈLE CHOISIR ?

- Que cherchons nous à optimiser ?
  - La proportion de prédictions correctes parmi les billets qui ont été prédits comme vrai => précision
- Quels sont les critères sur le modèle en lui même?
  - Facile à implémenter et à expliquer

	Modèle	Imputation NaN	Accuracy	Précision	Sensibilité	Spécificité	f1_score	R2 score	AUC
0	Dummy Classifier	Régression linéaire multiple	0.667	0.667	1.000	0.00	0.800	-0.500	0.5000
1	Régression logistique	Régression linéaire multiple	0.990	0.995	0.990	0.99	0.992	0.955	0.9900
2	Régression logistique - seuil = 0.6	Régression linéaire multiple	0.990	1.000	0.985	1.00	0.992	0.955	0.9925
3	k-means	Régression linéaire mulitiple	0.990	0.995	0.990	0.99	0.992	0.955	0.9900
4	Random Forest	Régression linéaire multiple	0.990	0.990	0.995	0.98	0.993	0.955	0.9875

➡ Modèle retenu : Régression logistique avec seuil à 0,6

MODÈLE FINAL

```python
def detection_fx_billets(model, nom_fichier):
    """
    Fonction permettant la détection de faux billets à partir d'un algorithme de classification déjà entraîné
    """

    # Importation des données
    df = pd.read_csv(nom_fichier)

    # Sélection des données significatives de la régression logistique
    X = df[['height_right','margin_low','margin_up','length']]

    # Standardisation des données
    scaler = StandardScaler()
    X_scaled = scaler.fit_transform(X.values)

    # Prédictions et probabilités
    pred = model.predict(X_scaled)
    predict_proba = model.predict_proba(X_scaled)[:,1]
    threshold_optim = 0.6 # Définition du seuil
    y_pred = (predict_proba >= threshold_optim) # Classification en fonction du seuil

    # Affichage des résultats
    df_pred = df.copy()
    df_pred['prediction'] = y_pred
    df_pred['probabilité vrai %'] = np.round(predict_proba*100,2)

    return df_pred
```

	diagonal	height_left	height_right	margin_low	margin_up	length	id	prediction	probabilité vrai %
0	171.76	104.01	103.54	5.21	3.30	111.42	A_1	False	0.23
1	171.87	104.17	104.13	6.00	3.31	112.09	A_2	False	0.09
2	172.00	104.58	104.29	4.99	3.39	111.57	A_3	False	0.07
3	172.49	104.55	104.34	4.44	3.03	113.20	A_4	True	100.00
4	171.65	103.63	103.56	3.77	3.16	113.33	A_5	True	100.00

Next step : Test du modèle en direct !

ANNEXES

- Autres algorithmes testés
- Imputation des valeurs manquantes

RÉSULTATS DES MODÈLES TESTÉS

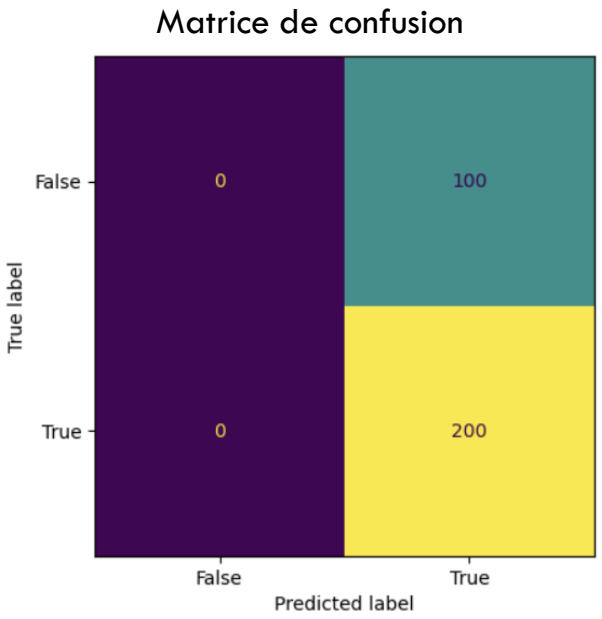
	Modèle	Imputation NaN	Accuracy	Précision	Sensibilité	Spécificité	f1_score	R2 score	AUC
0	Dummy Classifier	Régression linéaire multiple	0.667	0.667	1.000	0.00	0.800	-0.500	0.5000
1	Régression logistique	Régression linéaire multiple	0.990	0.995	0.990	0.99	0.992	0.955	0.9900
2	Régression logistique - seuil = 0.6	Régression linéaire multiple	0.990	1.000	0.985	1.00	0.992	0.955	0.9925
3	k-means	Régression linéaire mulitiple	0.990	0.995	0.990	0.99	0.992	0.955	0.9900
4	Decision Tree	Régression linéaire multiple	0.977	0.985	0.980	0.97	0.982	0.895	0.9750
5	Random Forest	Régression linéaire multiple	0.990	0.990	0.995	0.98	0.993	0.955	0.9875
6	kNN	Régression linéaire multiple	0.987	0.990	0.990	0.98	0.990	0.940	0.9850

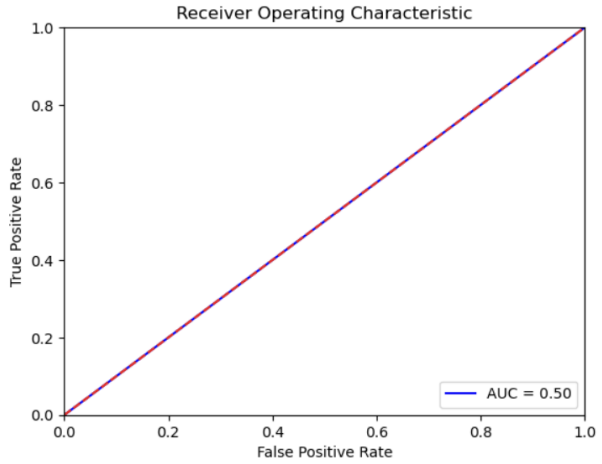
➡ Régression logistique avec seuil à 0,6 : meilleur des modèles testés

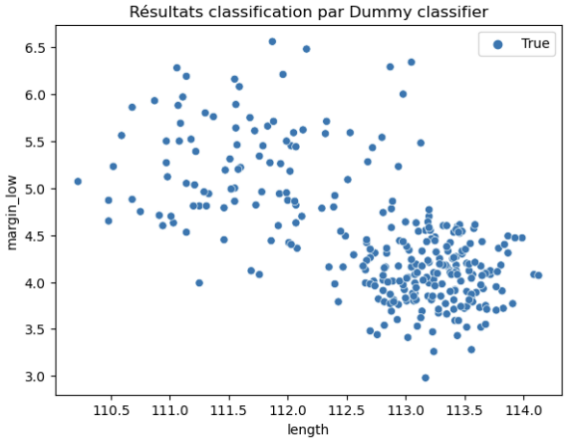
DUMMY CLASSIFIER

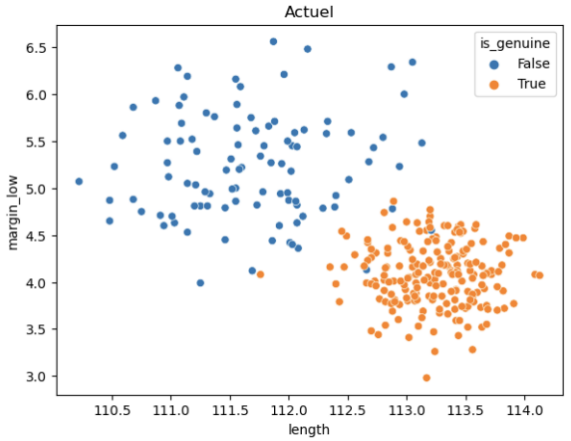
- Algorithme naïf pour servir de point de comparaison aux autres méthodes
- Méthode de classification : Most frequent
- Evaluation des performances :

Accuracy	Précision	Sensibilité	Spécificité	f1_score	R2 score	AUC
0.667	0.667	1.0	0.0	0.8	-0.5	0.5



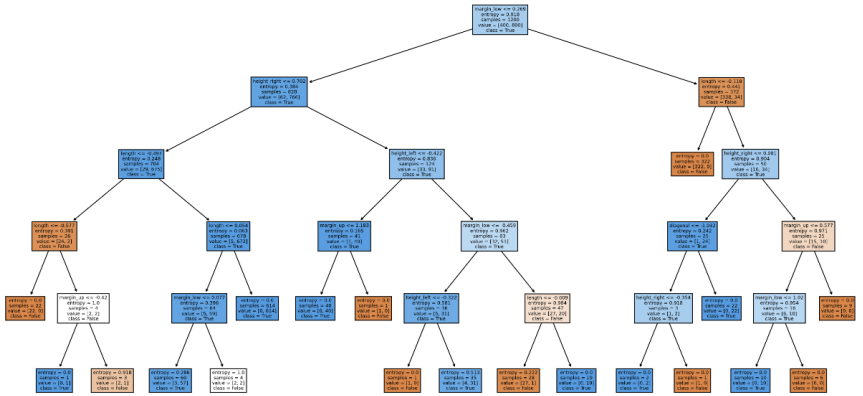






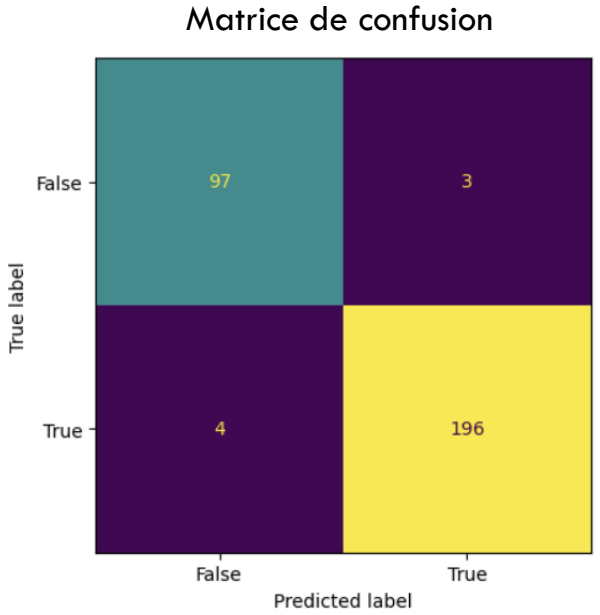
DECISION TREE

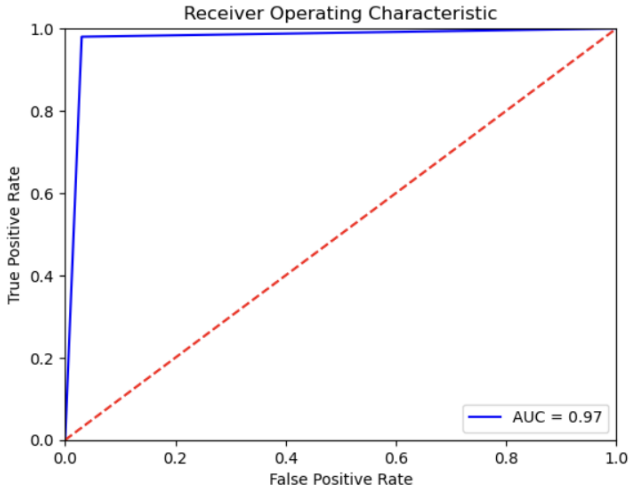
- Algorithme d'apprentissage supervisé
- Chaque variable du dataset est testée pour discriminer les données en définissant des règles logiques (un noeud qui aboutit à des branches). La décision est donnée au bout des "branches" et est appelée "feuille".

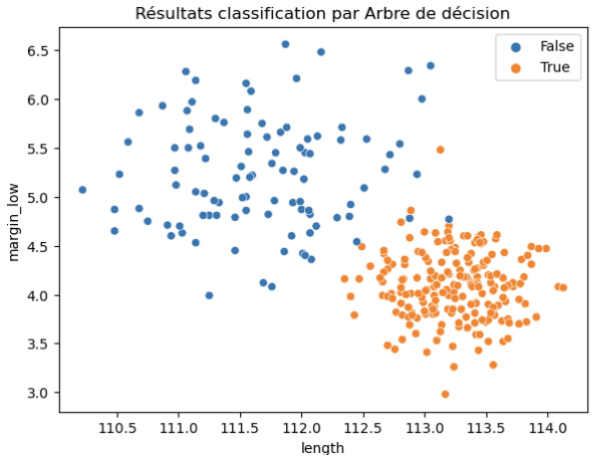


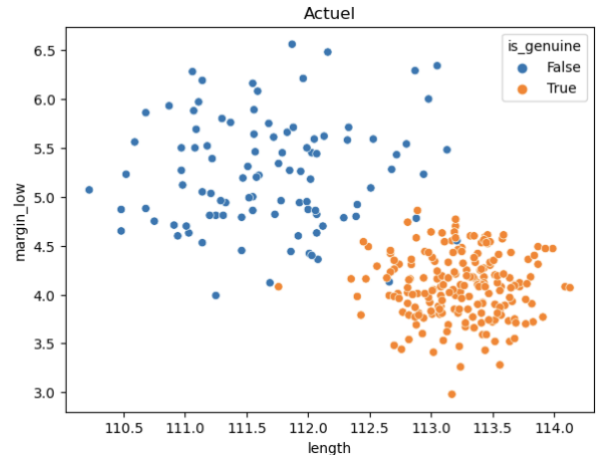
- Evaluation des performances :

Accuracy	Précision	Sensibilité	Spécificité	f1_score	R2 score	AUC
0.977	0.985	0.98	0.97	0.982	0.895	0.975







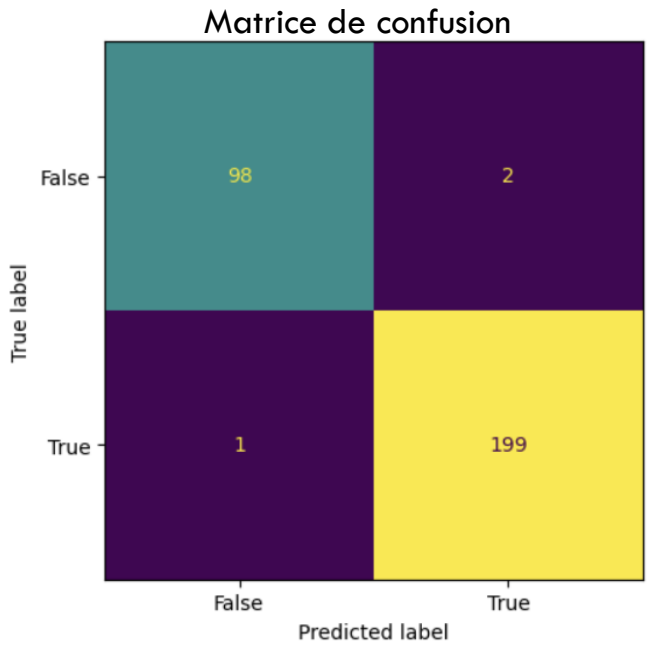


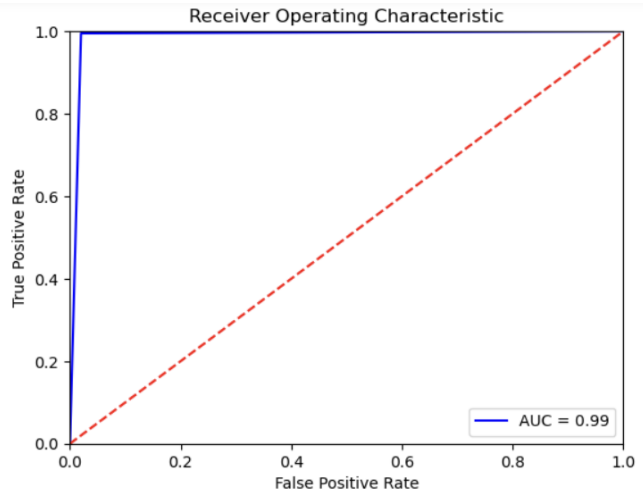
RANDOM FOREST

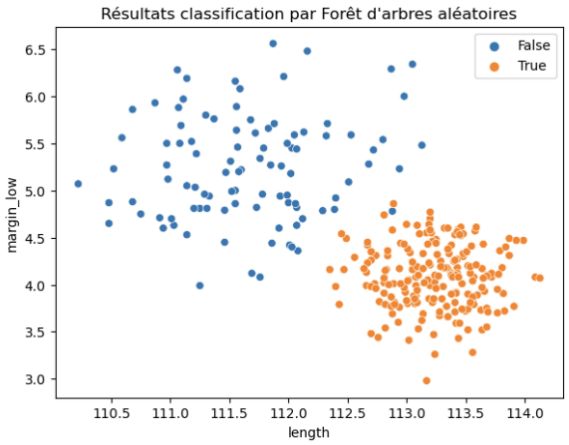
- Algorithme d'apprentissage supervisé, méthode d'ensemble
- Forêt d'arbres de décision : résultats de plusieurs arbres de décision combinés pour obtenir le résultat final.
- Evaluation des performances :

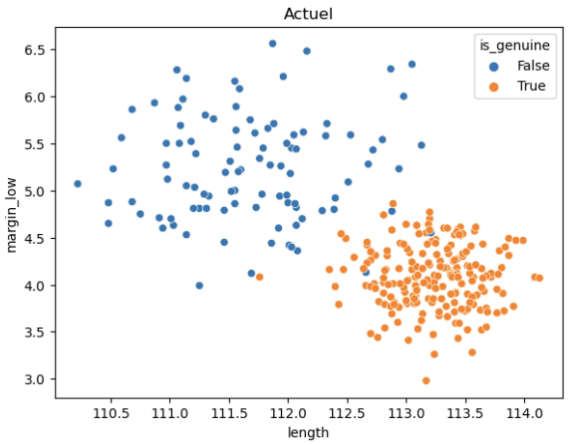
Accuracy	Précision	Sensibilité	Spécificité	f1_score	R2 score	AUC
0.99	0.99	0.995	0.98	0.993	0.955	0.9875

➡ Meilleur résultat qu'avec le Decision Tree Classifier





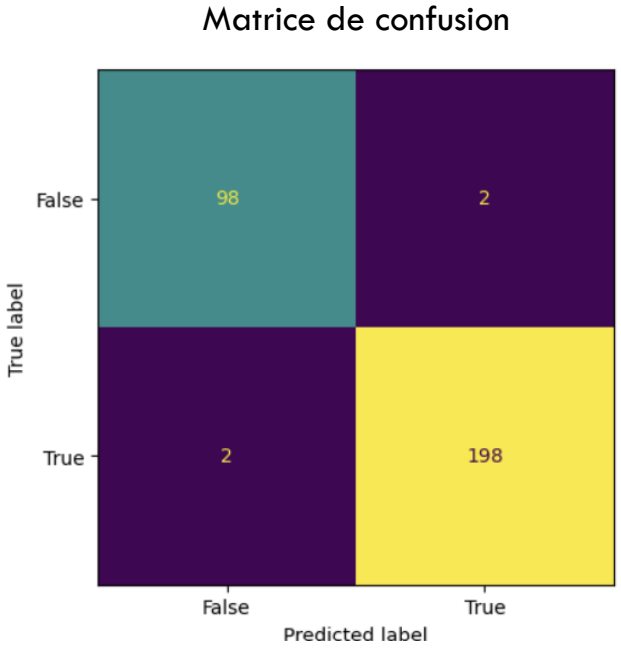


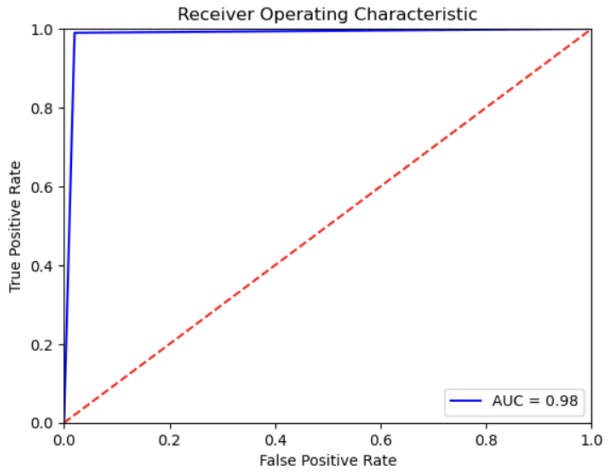


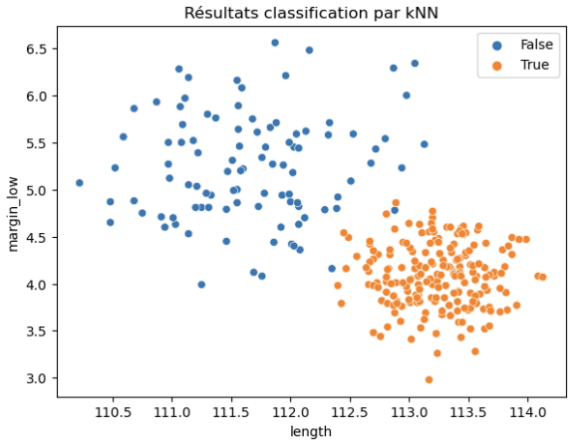
K-NN

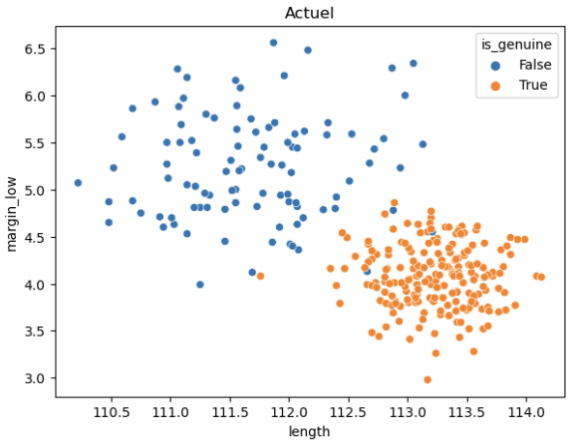
- Algorithme d'apprentissage supervisé
- Le k-NN considère les k voisins les plus proches du point à classifier et lui attribue la classe représentée par la majorité des points voisins.
  - k = 3 plus proches voisins
  - Les données d'entraînement sont conservées pour réalisées les prédictions -> /!\ capacité de mémoire
- Evaluation des performances :

Accuracy	Précision	Sensibilité	Spécificité	f1_score	R2 score	AUC
0.987	0.99	0.99	0.98	0.99	0.94	0.985









IMPUTATION DES VALEURS MANQUANTES

	Modèle	Imputation NaN	Accuracy	Précision	Sensibilité	Spécificité	f1_score	R2 score	AUC
0	Régression logistique	Sans les valeurs NaN	0.986	0.985	0.995	0.970	0.990	0.939	0.982271
1	Régression logistique	Moyenne	0.990	0.995	0.990	0.990	0.992	0.955	0.990000
2	Régression logistique	Régression linéaire multiple	0.990	0.995	0.990	0.990	0.992	0.955	0.990000
3	k-means	Sans valeurs NaN	0.980	0.975	0.995	0.949	0.985	0.908	0.972170
4	k-means	Moyenne	0.990	0.995	0.990	0.990	0.992	0.955	0.990000
5	k-means	Régression linéaire mulitiple	0.990	0.995	0.990	0.990	0.992	0.955	0.990000

- Imputation par rapport à suppression des valeurs manquantes : meilleurs résultats avec les données manquantes imputées (= algorithmes entraînés avec plus de données).
- Méthodes d'imputation moyenne ou régression linéaire multiple : résultats identiques, un remplacement des valeurs manquantes par la moyenne, méthode plus simple, aurait été suffisant.