Université de Pau et des Pays de l'Adour

Rapport Machine Learning : Données bancaires

Date de rendu : 09 Mars 2026

ACKER Théo, SINYEUE Noël

Sous la direction d'Olivier PERON

Table des matières

1 Introduction

Le jeu de données "Banque 12" comporte les informations de 5000 individus, clients de la banque. Les individus sont associés à plusieurs caractéristiques ; Parmi elles, des caractéristiques financières, tel que la possession ou non d'un compte épargne logement, et des caractéristiques sociodémographique, tel que l'âge ou encore le nombre d'enfants de l'individu.
On peut diviser ces individus en deux catégories ; Ceux qui se sont vus accordés un prêt et ceux qui ne se sont pas vu accordés de prêt (40 % contre 60 %). Ils sont représentés par la variable *Prêt*, qui sera notre variable cible.

Notre objectif lors de ce rapport est de trouver le modèle le plus fiable et robuste pour, à partir des caractéristiques fournies, determiner si oui ou non l'individu recevera son prêt. On en tirera également des informations sur les caractéristiques qui ont été le plus efficace dans la discrimination des deux groupes.

Le jeu étant désequilibré (la distribution des modalités de *Prêt*, notre variable cible, étant inégale), il faudra prendre des précautions lors de l'initialisation de nos modèles. On veuillera à ce que l'apprentissage se fasse de manière stable entre les deux catégories : les individus ayant reçu le prêt et les autres. Certains désequilibres ne nécessitant pas de rééquilibrage, les modèles seront testés avec et sans pondération pour s'assurer que l'application des poids ait bien eu un effet mélioratif dans les classifications obtenues.

Finalement, nous appliquerons les modèles les plus robustes dans leur classification sur 40 individus, fraîchement demandeur d'un prêt.

2 Statistiques descriptives

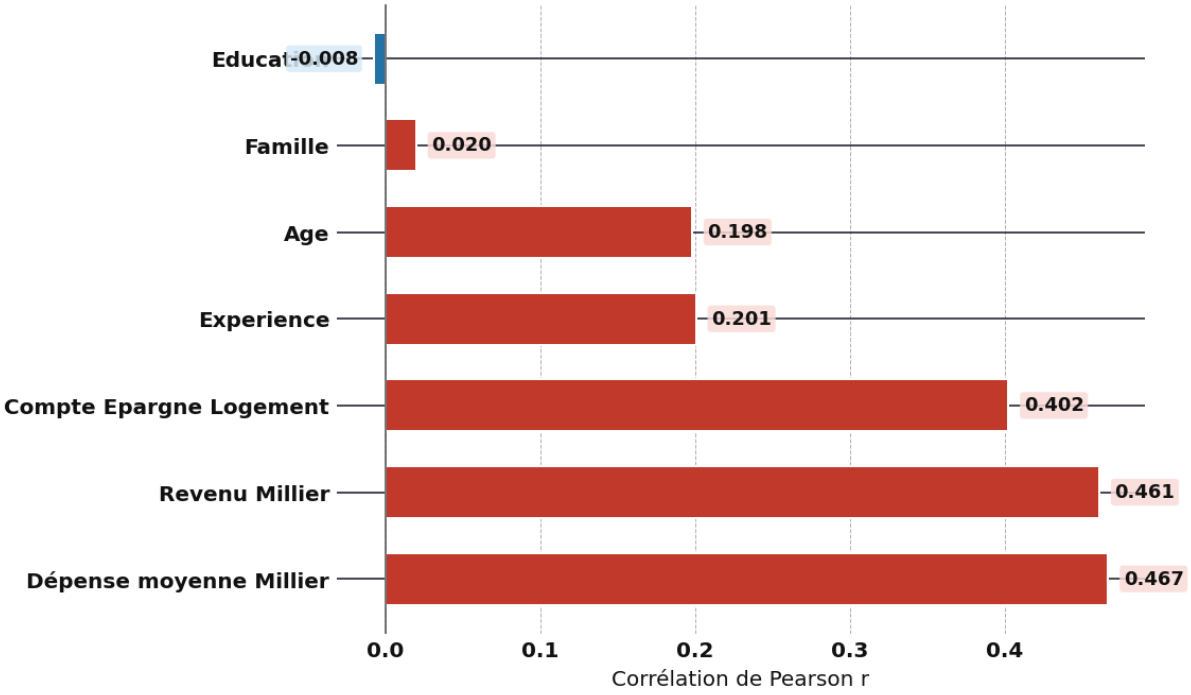

FIGURE 1 – Coefficients de corrélation des variables avec 'Prêt Personnel'

Cette figure représente la correlation linéaire (mesurée par le coefficient de Pearson) entre la variable dépendante et les variables explicatives. À première vue, les variables qui seraient le plus à même de contribuer à la puissance explicative du modèle seraient celles du revenu et des dépenses moyennes, les deux étant les correlations les plus fortes du modèle. On retrouve d'autres variables avec un lien linéaire négligeable comme celle de l'éducation ou de la famille. Il serait toutefois une erreur d'enlever ces variables des prédictions et ce pour deux raisons majeures :
- Le jeu de données comporte un faible nombre de variables
- L'avantage du machine learning est que nous pouvons aller au delà des liens linéaires entre les variables ; des liens non-linéaires peuvent donc améliorer notre capacité prédictive.
Dans le pire des scénarios, elles ajouteront un bruit ou auront une contribution marginale voire nulle, mais elles n'auront pas d'impact significativement négatif sur les prédictions des modèles.

3 Modèles et méthodologie

Les modèles testés ont été évalué sur leur capacité à détecter des individus ayant obtenu un prêt. Ainsi, les indicateurs que nous avons utilisé pour comparer les modèles sont les suivants :
- **La sensibilité**, qui mesure le pourcentage de vrais positifs d'un modèle (le nombre d'individus ayant reçu un prêt et étant correctement classé par le modèle sur le nombre total d'individus qui ont reçu un prêt dans l'échantillon).
- **La précision**, qui mesure le pourcentage d'individus classés comme ayant reçu un prêt

par le modèle et qui ont en réellement reçu un.
- **Le F1-score**, qui contient à la fois une mesure de la précision et de la sensibilité.
- **Le score AUC** (Area under curve) qui est un indicateur de la puissance prédictive générale. Cet indicateur sera moins considéré que les précédents car basé à la fois sur les vrais positifs et sur les vrais négatifs, là où seulement les vrais positifs nous intéressent ici.
Chacun des indicateurs fonctionne de la même manière, toutes choses égales par ailleurs, plus son score s'approche de 1, plus le modèle est performant.

Note: F1-score

Le F1-score fait l'arbitrage entre la précision et la sensibilité : ce score permet de sanctionner les modèles qui auraient des valeurs faibles pour l'un ou l'autre de ces indicateurs. En effet, une forte précision peut amener à de mauvaises prévisions sur les vrais positifs si elle est couplée à une faible sensibilité et inversément.

3.1 Les modèles de regression logistiques

3.1.1 Les paramètres du modèle LOGIT

Le premier des éléments qui a été testé sur le modèle est le seuil de probabilités à partir duquel on classe les individus dans la classe positive (classe des individus ayant reçu un prêt). Le seuil a été optimisé pour maximiser le F1-score (seuil : 0.44). Un modèle aura pour notation **"+ seuil F1"** lorsque que l'on utilisera ce seuil optimisé.

La régularisation est liée à l'estimation du modèle LOGIT ; elle est nécessaire pour prévenir de problèmes de multicolinéarité et de surrapprentissage qui pourrraient avoir un impact sur les performances classificatives du modèle. Deux méthodes de regularisation par pénalisation ont été testés : Lasso et Ridge.
Un modèle aura pour notation **"+régul"** si il a subit cette régularisation.

3.1.2 Résultats

Modèle \ Indicateurs	Précision	Sensibilité	F1-score
LOGIT pondéré	0.681	0.785	0.729
LOGIT pondéré + seuil F1	0.651	0.833	0.731
LOGIT pondéré + seuil F1 + régul	0.645	0.815	0.720

TABLE 1 – Tableau de comparaison des modèles LOGIT

On obtient des résultats similaires au niveau du F1-score entre le modèle pondéré simple et le modèle pondéré avec seuil optimisé. À F1-score équivalents, on préférera le modèle avec une meilleur sensibilité car ce modèle détectera plus de vrais positifs.

En optimisant les hyperparamètres pour la régularisation (comparés sur la base de la maximisation du F1-score), le groupe de paramètres qui l'emporte est systématiquement celuit qui mène à la régularisation la moins forte. On se retrouve donc avec un modèle avec une régularisation marginale. Tous les indicateurs du modèle LOGIT régulé tendent donc, toutes choses égales par ailleurs, vers les indicateurs des modèles non régulés, sans pour autant les atteindre ; La régularisation a créé un biais supplémentaire qui réduit les performances du modèle. On en conclut donc que le modèle n'est pas exposé à des problèmes de multicolinéarité ou de surrapprentissage trop important, aucune régularisation ne sera appliquée au modèle LOGIT.

On retiendra ainsi parmi ces trois modèles le modèle **"LOGIT pondéré + seuil F1"**.

3.2 Les arbres de décisions

L'ensemble des paramètres testés et utilisés pour chaque modèle se retrouve en **5. Annexes**.

3.2.1 Modèle de base

Dans cette première étape, nous comparerons le modèle d'arbre de décision de base (avec le critère de Gini et d'entropie) avec le modèle LOGIT retenu en 3.1.2.

Les critères de Gini et d'entropie sont deux critères de divisions ; Ils permettent, à chaque noeuds de l'arbre, de déterminer la variable qui expliquera le mieux les différences de modalités de la variable dépendante. Pour chaque variable associée à un noeud, les critères de Gini ou d'entropie permettent de trouver le seuil réduisant au maximum l'imprécision du modèle.

Modèle \ Indicateurs	Précision	Sensibilité	F1-score	AUC
Arbre de décision pondéré (Gini)	0.646	0.833	0.728	0.842
Arbre de décision pondéré (Entropie)	0.663	0.818	0.732	0.848
LOGIT pondéré + seuil F1	0.651	0.833	0.731	0.844

TABLE 2 – Arbres : Tableau de comparaison des modèles de base

L'arbre de de décision avec division par critère d'entroprie s'avère le plus adapté des deux, il tire de meilleurs résultats notamment grace à sa bonne précision (qui compense sa plus faible sensibilité) et sa capacité prédictive générale plus élevée. Ce modèle obtient même un F1-score plus élevé que le modèle LOGIT retenu.

3.2.2 Modèles d'ensemble

Dans cette partie, nous traiterons de tous les modèles qui se basent sur un ensemble d'arbres de décision pour leur prédiction, nommément :
- Le **Bagging**, qui tire plusieurs échantillons du jeu de données, fait des prédictions pour chacun d'entre eux et fait la moyenne des résultats obtenus.

- Le **Random Forest** qui utilise le même procédé que le Bagging en sélectionnant les variables testées de manière aléatoire.
- Le **Gradient Boosting** qui fonctionne séquentiellement, en appliquant plus d'importance aux individus qui ont été mal classés.
- L'**AdaBoost** qui fonctionne séquentiellement, en appliquant plus d'importance aux individus qui ont été mal classés et plus d'importances aux modèles qui ont été plus efficaces.

Modèle \ Indicateurs	Précision	Sensibilité	F1-score	AUC
Bagging Arbre pondéré (Gini)	0.669	0.797	0.727	0.833
Random Forest	0.749	0.680	0.713	0.862
AdaBoost	0.727	0.692	0.709	0.847
Gradient Boosting	0.713	0.708	0.709	0.830

TABLE 3 – Arbres : Tableau de comparaison des modèles d'ensemble

On retiendra parmi ces quatre modèles le modèle Bagging ; Malgré une force prédictive générale plus faible (dû à sa moins bonne capacité à trouver les vrais négatifs (la plus faible exactitude des quatre)), le modèle affiche un meilleur F1-score, qui se rapproche des scores des modèles arbre et logit précédemment retenus.

Remarque : Le Bagging a également été testé avec le critère de division entropie et les résultats étaient très proches du modèle Bagging (Gini) mais moins pertinents.

Note: Exactitude

L'exactitude mesure le taux de bonnes réponses d'un modèle. Si un modèle A a un F1-score moins élevé qu'un modèle B mais un meilleur taux d'exactitude, alors, il est très probable que le modèle A ait un meilleur taux de vrais négatifs.

Les modèles d'ensemble ont eu plus de mal à prédire les vrais positifs pour deux raisons principales : le déséquilibre initial du jeu de donnée et son faible nombre de variables.

Nous allons maintenant tester un modèle dérivé du Radom Forest classique, qui permet de mieux s'adapter aux modèles désequilibrés : Le **Balanced Random Forest Classifier**

Note: Balanced Random Forest Classifier

Ce modèle permet à ce que les individus issus de classes minoritaires soient mieux classés. Pour cela, il attribue pour chaque échantillons, un même nombre d'individus provenant de la classe minoritaire et majoritaire. Ainsi, chaque arbre du processus des Random Forest comportera autant d'individus avec et sans prêt.

Grâce à la correction des effets du déséquilibre, on obtient une bien meilleur sensibilité qui se répercute dans le F1-score et également dans le pouvoir de classification général du modèle. Cela en fait le modèle d'ensemble le plus adapté à notre mission.

Modèle \ Indicateurs	Précision	Sensibilité	F1-score	AUC
Random Forest	0.749	0.680	0.713	0.862
Balanced Random Forest Classifier	0.708	0.818	0.759	0.88

TABLE 4 – L'apport du modèle "Balanced Random Forest Classifier"

3.3 Choix des modèles

On retient donc les trois modèles suivants :
- Le modèle LOGIT pondéré avec le seuil optimisé pour le F1-score.
- Le modèle d'arbre avec pondération et critère d'entropie.
- Le modèle Balanced Random Forest Classifier.

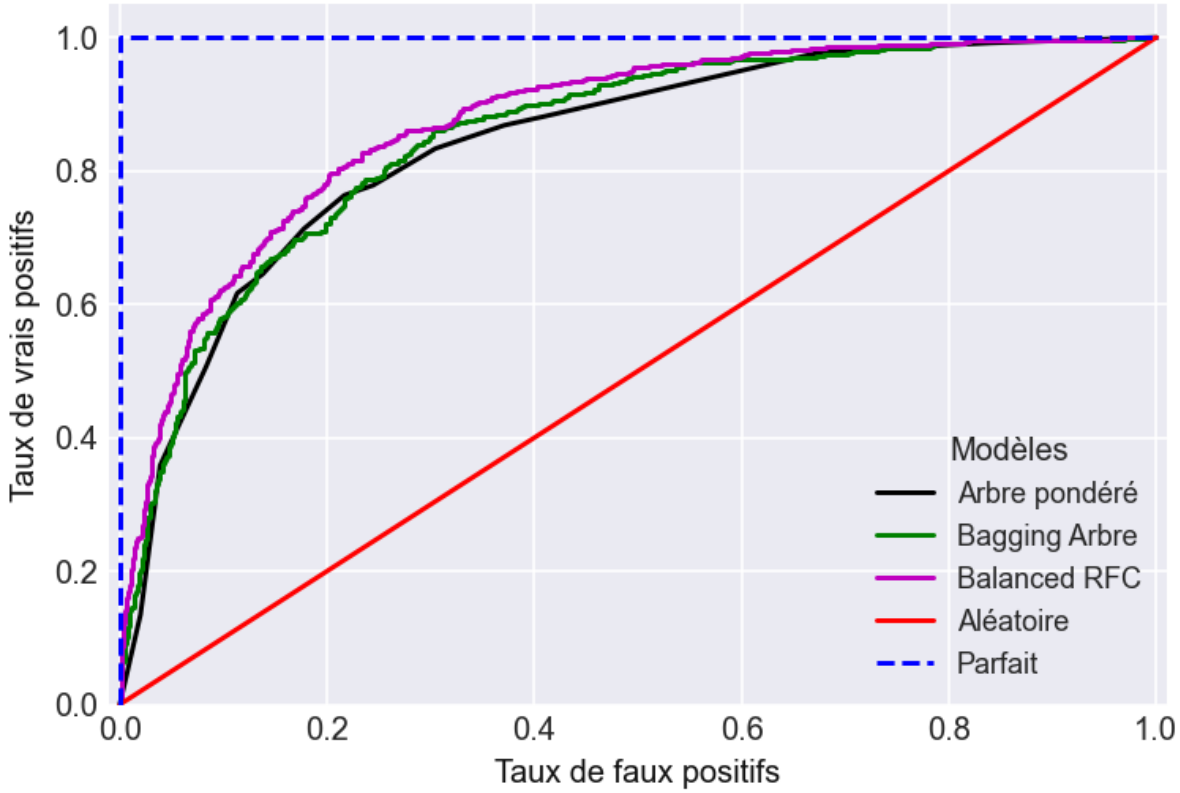

FIGURE 2 – Courbe ROC des trois modèles retenus

Comme attendu, le modèle Balanced Random Forest Classifier (Balanced RFC) domine les débats : il est celui qui affiche la meilleur performace prédictive générale.

Note: Score AUC et courbe ROC

Le score AUC est défini à partir de la courbe ROC ; Plus la courbe tend vers la courbe "parfaite"(celle telle que toutes les prédictions sont correctes), plus le score AUC est fort et donc plus les performances classificative du modèles sont fortes, elles aussi.

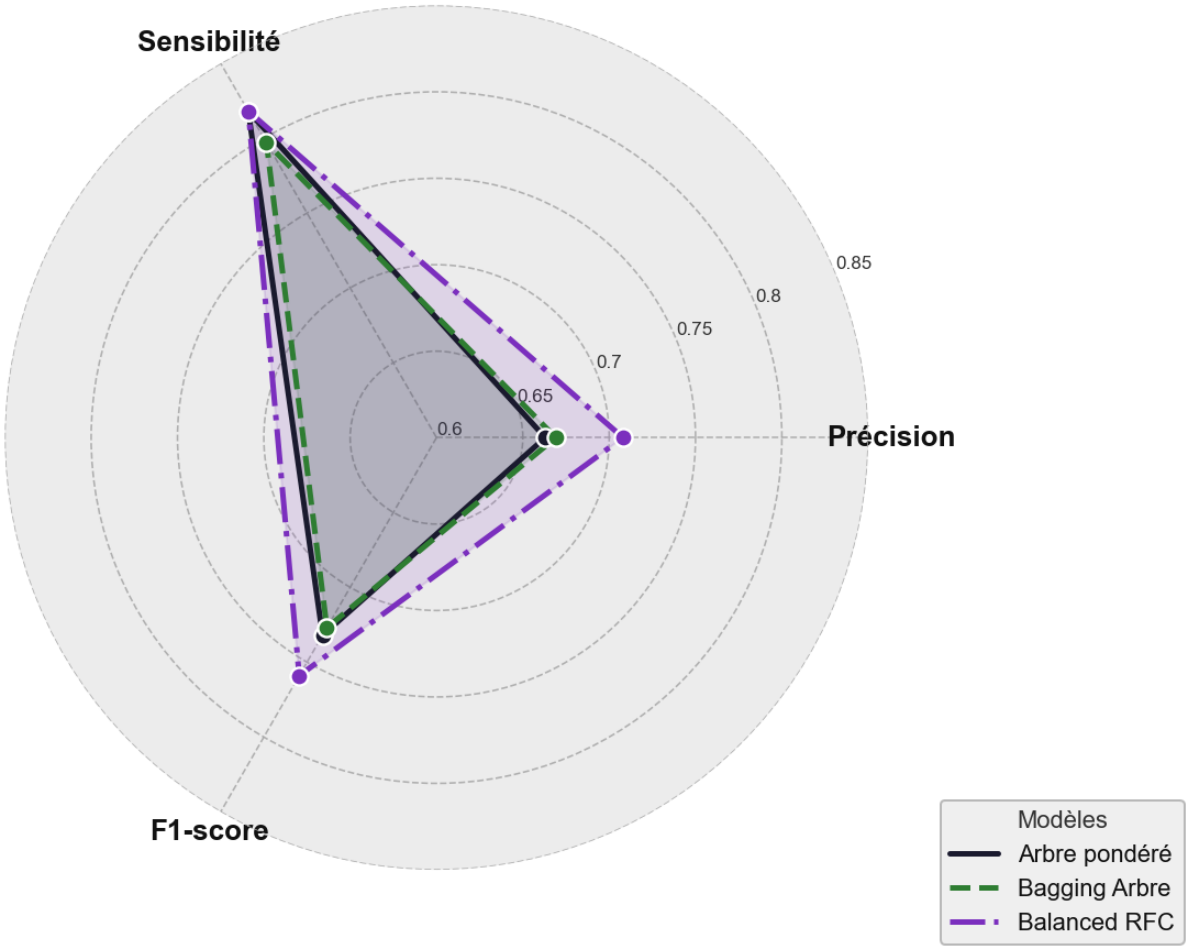

FIGURE 3 – Graphique radar des trois modèles retenus

Le modèle Balanced Random Forest Classifier est le plus performant dans la détection de classe positive pour ce jeu de données. Il domine nos deux incateurs les plus pertinents à savoir : Le f1-score et la sensibilité. Le constat est le même que celui pour la performance de classifications globale (pour les classes positives et négatives). Les estimations seront donc faites avec ce modèle.

4 Classification et importance

4.1 Les prédictions

N°	Classe Prédite	Confiance (%)
1	0	94.4
2	0	89.1
3	0	93.6
4	0	90.7
5	0	93.7
6	0	92.9
7	0	83.7
8	0	87.9
9	0	89.1
10	0	63.7
11	0	72.9
12	0	93.9
13	0	78.0
14	0	80.9
15	0	68.0
16	0	88.2
17	0	80.2
18	0	84.8
19	0	51.5
20	0	89.1

N°	Classe Prédite	Confiance (%)
21	0	87.5
22	0	83.9
23	0	94.5
24	0	89.0
25	0	77.8
26	0	89.9
27	0	83.8
28	0	73.5
29	0	85.5
30	0	60.6
31	0	85.8
32	0	92.3
33	0	85.3
34	0	95.3
35	0	93.8
36	0	82.5
37	0	73.5
38	0	84.1
39	1	58.9
40	0	86.9

TABLE 5 – Prédictions des 40 observations avec le modèle "Balanced RFC"

La confiance moyenne pour ces résultats est de 83,3%. Le modèle prédit que 39 clients verront leur demande de crédit refusée et un seul verra sa demande de crédit acceptée (le client n°39).

4.2 L'importance des variables

Variable	Arbre.pondéré	Bagging.Arbre	Balanced.RFC
Compte Epargne Logement	46.46	31.35	27.06
Revenu Millier	23.29	20.18	26.83
Dépense moyenne Millier	20.20	30.98	32.43

TABLE 6 – Importance des variables les plus contributives (en pourcentage)

Les trois variables qui avaient les trois coefficients de correlation les plus élevés contribuent, toutes les trois, le plus à la performance des modèles retenus. Toutefois, la variable *Compte Epargne Logement* se distingue par de meilleurs apports en moyenne : elle joue un rôle clé dans l'efficacité de la classification.

5 Annexes

5.1 Détails sur l'optimisation des modèles

```
1- Modèle Arbre de décision pondéré (Entropie):
max_depth=4; min_samples_split=100; min_sample_leaf=50; random_state=42

2- Modèle Random Forest :
estimator__criterion : 'entropy'
estimator__max_depth :5
estimator__min_samples_leaf: 15
estimator__min_samples_split: 20
n_estimators: 100

=> Modalités des hyperparamètres testés:
'estimator__criterion': ['gini', 'entropy']
'estimator__min_samples_split': [20, 30, 50, 100]
'estimator__min_samples_leaf': [5, 10, 15, 30]
'estimator__max_depth': [2,3, 4, 5]
'n_estimators': [50, 100, 150]

3- Modèle ADABoost (optimisé):
estimator__criterion: 'gini'
estimator__max_depth: 5
estimator__min_samples_leaf: 5
estimator__min_samples_split: 50
learning_rate: 0.5
n_estimators: 150

4- Gradient Boosting :
learning_rate: 0.01
max_depth: 4
n_estimators: 1000
subsample: 0.7

=> Modalités des hyperparamètres testés:
'n_estimators': [200, 400,1000]
'learning_rate': [0.01, 0.05, 0.1]
'max_depth': [4,8,10]
'subsample': [0.7, 0.85,1]


5- Modèle LOGIT: Optimisation de la régularisation
{'C': 100, 'penalty': 'l2', 'solver': 'lbfgs'}

=> Modalités des hyperparamètres testés:
param_grid = [
    {"penalty": ["l1"],
```

```
     "solver": ["liblinear"],
     "C": [0.001, 0.01, 0.1, 1, 10, 25, 50],}
    {"penalty": ["l2"],
     "solver": ["lbfgs"],
     "C": [0.001, 0.01, 0.1, 1, 10, 25, 50, 100] }]

6- Modèle Balanced RANDOM FOREST CLASSIFIER:
max_depth: 5,
max_features: 'sqrt'
min_samples_leaf: 2
min_samples_split: 2
n_estimators: 500

=> Modalités des hyperparamètres testés:
"n_estimators": [100, 200, 500]
"max_depth": [5, 10, None]
"max_features": ["sqrt", "log2"]
"min_samples_split": [2, 5, 10]
"min_samples_leaf": [1, 2, 4]
```

5.2 Code

L'intégralité du code se trouve en .py dans le dossier compressé.