TP1 RAPPORT INF5081

Pour le 15 mars 2024

AMADOU SARA BAH
[BAHA09019703]
Tamrat Beede Mikael
[TAMB89080102]

Introduction

Contexte général du projet :

Ce projet qui est un travail pratique est basé sur l'apprentissage automatique. L'objectif est de comprendre comment les algorithmes de machine learning peuvent être appliqués pour analyser des ensembles de données spécifiques et résoudre des problèmes de classification. Pour être plus précis, le projet se concentre sur l'utilisation de données collectées à partir de Twitter, une plateforme de médias sociaux où divers types d'utilisateurs interagissent. Parmi ces utilisateurs, certains sont identifiés comme des pollueurs de contenu (Content Polluters), qui diffusent du contenu indésirable ou abusif(spam..), tandis que d'autres sont des utilisateurs légitimes engagés dans une utilisation normale de la plateforme. On veut trouver moyen de pouvoir les différencier.

Objectif du travail :

L'objectif de ce travail est de développer un ensemble de modèles de classification capables de distinguer efficacement entre les utilisateurs légitimes et les pollueurs de contenu sur Twitter. Pour atteindre cet objectif, plusieurs étapes sont nécessaires :

1. **La préparation et le nettoyage des données**, qui impliquent la suppression des doublons, le traitement des valeurs manquantes, et la normalisation des données pour garantir une analyse cohérente.
2. **Le calcul des attributs spécifiques** qui caractérisent les comportements des utilisateurs sur Twitter, tels que la fréquence des tweets, etc. Ces attributs ont tous d'abord étais calculer à la mains (trouver la bonne formule) comme elle sont essentiels pour permettre aux algorithmes de classifier correctement les utilisateurs.
3. **L'entraînement et l'évaluation de différents modèles** de machine learning, tels que l'arbre de décision, la forêt aléatoire (Random Forest) et la classification bayésienne naïve, pour déterminer lequel offre la meilleure performance en termes de précision de classification, en utilisant des métriques telles que le taux de vrais positifs, le taux de faux positifs, la mesure F et l'aire sous la courbe ROC (AUC).
4. **L'analyse comparative des modèles**, on ne se concentre pas seulement à l'évaluation des performances entre ces modèles mais à l'impact qu'a la sélection d'attributs sur les performances des modèles. Cette étape vise à identifier les attributs les plus pertinent et à évaluer comment leur utilisation influence la capacité des modèles à distinguer entre les différents types d'utilisateurs.

Calcul des attributs :

Pour les calculs comme on l'a dit précédemment nous avons d'abord trouver les formules dont on va employer pour retrouver chaque attribut dont on va utiliser. C'est attribut bien que donné par l'instruction non pas étais choisis aléatoirement, les attributs choisis ont été déterminés en fonction de leur pertinence à différencier les utilisateurs légitimes et pollueurs. Par exemple la longueur des noms d'utilisateurs (LengthOfScreenName) chez les pollueurs à tendance à être des noms d'écran plus longs et génériques qui nous aidera à différencier entre les utilisateurs. Mais une fois tous attributs trouver et fait, nous avons commencé le code. Pour le code, on ne pouvait pas juste prendre les données et appliquer la formule dessus. Il fallait quand prépare, corrige et range ces données, pour faire cela nous avons passé par plusieurs étapes.

Étapes effectuées : On a commencé par une phase de nettoyage et de prétraitement des données brutes collectées. Cette phase avait pour objectif de rendre les données cohérentes, exploitables et prêtes pour l'analyse. Nous avons procédé à la:

- **Suppression des doublons** : Pour assurer l'unicité de chaque instance dans notre dataset, nous avons éliminé les entrées répétées.
- **Remplacement des valeurs manquantes** : remplacé par la médiane par exemple.
- **Conversion des dates** : Les champs temporels tels que CreatedAt ont été convertis en format datetime standard de Python pour faciliter les manipulations et les calculs temporels ultérieurs.
- **Normalisation des données** : Afin d'homogénéiser les échelles des différents attributs numériques et de réduire les biais potentiels, nous avons appliqué une normalisation z-score aux colonnes sélectionnées.
    - Pour, la normalisation z-score ajuste les chiffres pour que chaque caractéristique, soit évaluée de manière juste, indépendant de sa taille(nombre de tweet...). Permettant ainsi à tous les utilisateurs d'être comparés équitablement. En bref, cette normalisation nous aide à ce concentré sur les caractéristiques des utilisateurs sans être confus(biaiser) par la taille des chiffres.

Prétraitement : Le prétraitement a également impliqué l'identification et le traitement des valeurs manquantes qui pouvaient affecter la qualité de nos analyses. Pour les attributs numériques, la stratégie adoptée a consisté à remplacer les valeurs manquantes par la médiane comme mentionné précédemment.

Méthode utilisée pour traiter les valeurs manquantes : La décision de traiter les valeurs manquantes par remplacement ou suppression a été fait en analysant les données. Le choix de la médiane pour les données numériques et dut au fait de sa moindre sensibilité aux valeurs extrêmes par rapport à la moyenne, ce qui en fait une méthode de remplacement préférable dans nos cas d'usage.

Une fois le nettoyage des données et la normalisation de certaine de ces données fait, nous avons procéder à la création du fichier .csv, pour l'analyse garce a notre code qui encapsule toutes les données dans un fichiers .csv en les classant en 2 catégorie (1 pour les utilisateurs polluant et 0 pour légitimes) pour les futures analyses de l'étape qui suit.

3. Entrainement des modèles et analyse comparative :

3.1 Utilisation de tout l'ensemble d'attributs

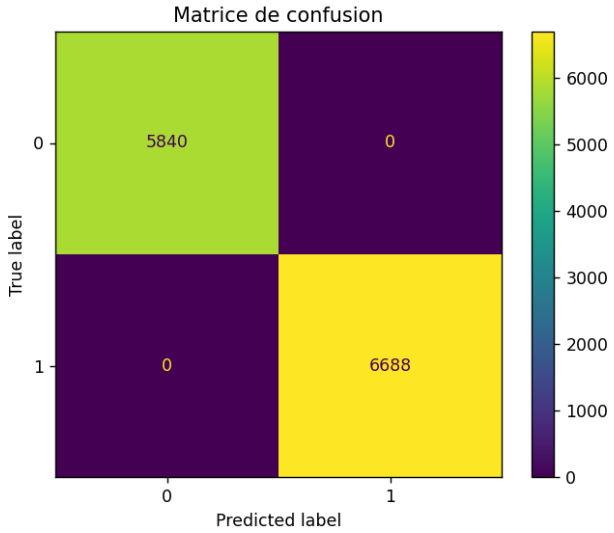
Maintenant que nous avons notre fichier .csv qui contient tous les données pertinent(attributs important...) des utilisateurs polluant et légitimes séparés en catégorie, nous pouvons utiliser ceci pour l'entrainement des algorithmes. Nous utilisons notre classe "comparaison_full_features" pour séparer cet ensemble de données, avec **70%** des données utilisé comme apprentissage pour nos algorithme et **30%** des données utilisés pour les tests. Cette fonction est exécutée par "run_comparaison_full_features" qui utilise les définitions dans cette classe pour pouvoir exécuter l'apprentissage.

Une fois cela fait “run_comparaison_full_features” va ensuite utiliser cet apprentissage pour exécuter les tests. Il fait les calculs et affiches les résultats, rechercher, mais il se parvient d’une autre classe nommer “model_visualizer” pour aussi générer la matrice de confusion et la courbe ROC de chaque algorithme utiliser pour les tests, nous obtenons les résultats suivants :

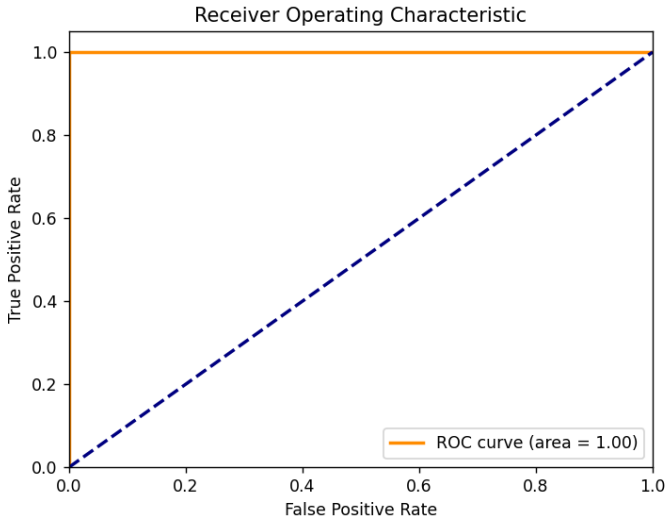
- Analyse avec l’arbres de décisions :
  - Accuracy: 1.0
  - Taux de VP (Vrais Positifs): 1.0
  - Taux de FP (Faux Positifs): 0.0
  - F-measure: 1.0
  - AUC: 1.0
  - Rapport de Clasification :

	precision	recall	f1-score	support
0	1.00	1.00	1.00	5840
1	1.00	1.00	1.00	6688
accuracy			1.00	12528
Macro avg	1.00	1.00	1.00	12528
Weighted avg	1.00	1.00	1.00	12528

  - Matrice de confusion :



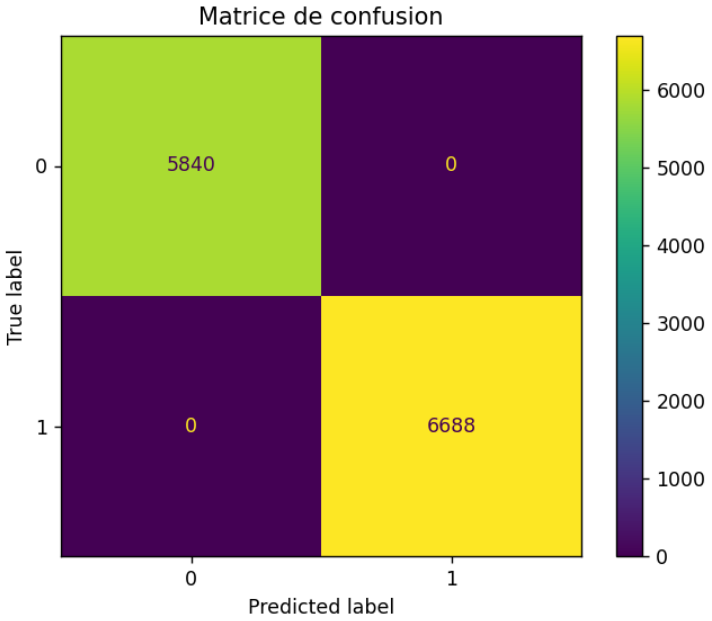
  - Courbes ROC :



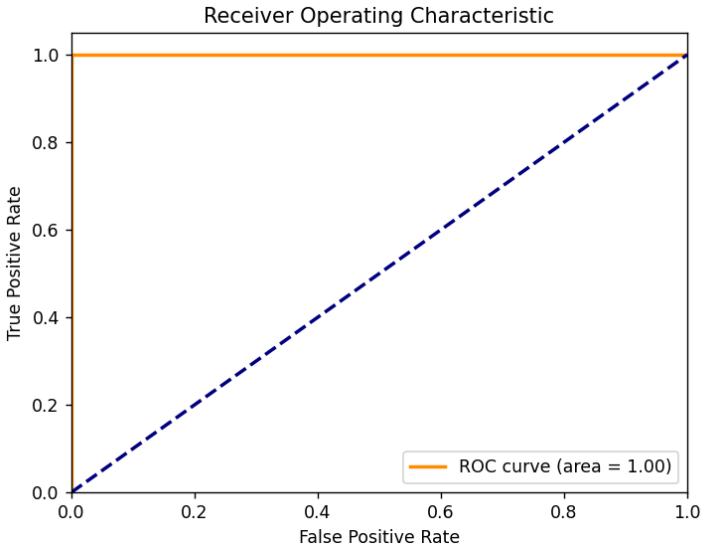
- Analyse avec la forêt aléatoire :
  - Accuracy: 1.0
  - Taux de VP (Vrais Positifs): 1.0
  - Taux de FP (Faux Positifs): 0.0
  - F-measure: 1.0
  - AUC: 1.0
  - Rapport de Clasification :

	precision	recall	f1-score	support
0	1.00	1.00	1.00	5840
1	1.00	1.00	1.00	6688
accuracy			1.00	12528
Macro avg	1.00	1.00	1.00	12528
Weighted avg	1.00	1.00	1.00	12528

  - Matrice de confusion :



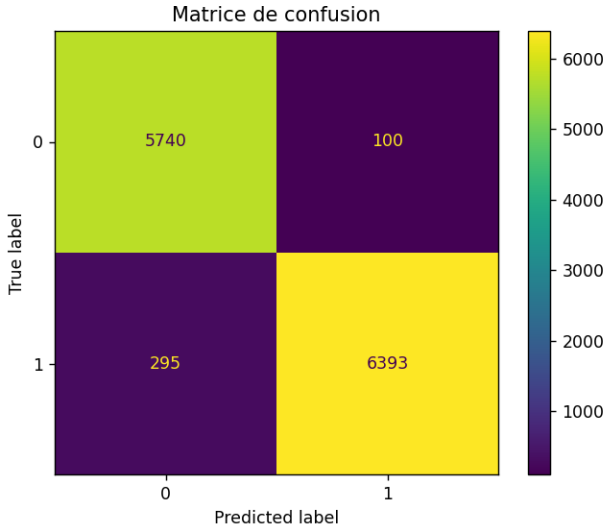
  - Courbes ROC :



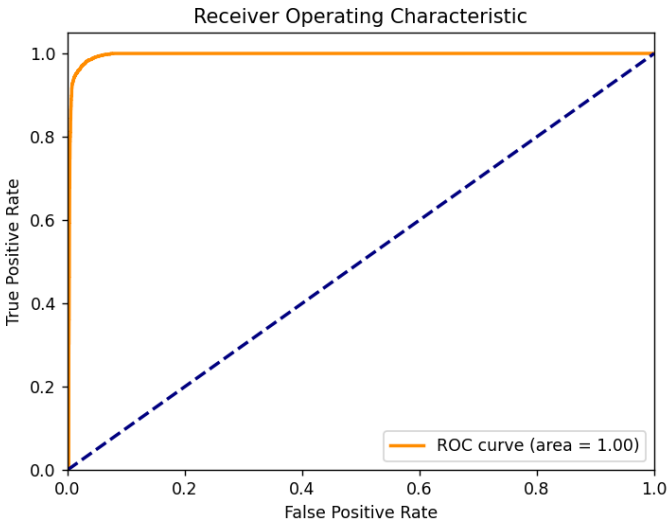
- Analyse avec la classification bayésienne naïve :
  - Accuracy: 0.9685
  - Taux de VP (Vrais Positifs): 0.9559
  - Taux de FP (Faux Positifs): 0.0171
  - F-measure: 0.9700
  - AUC: 0.9952
  - Rapport de Clasification :

	precision	recall	f1-score	support
0	0.95	0.98	0.97	5840
1	0.98	0.96	0.97	6688
accuracy			0.97	12528
Macro avg	0.97	0.97	0.97	12528
Weighted avg	0.97	0.97	1.00	12528

  - Matrice de confusion :



  - Courbes ROC :



D'après les résultats dont on a obtenus ne pouvons voir les informations suivantes sur les algorithmes de l'arbres de décisions et la forêt aléatoire s'appliquent :

- Les résultats sont complètement identiques avec un accuracy, Taux de VP, F-mesure et AUC égal à 1 et un Taux de FP égal à 0

- **Point fort :**Oon a obtenu des scores parfaits pour tous les critères (TP Rate, FP Rate, F-measure, AUC). Cela suggère que le modèle a parfaitement classé les instances dans le jeu de données utilisé.
- **Point faible** : Ces résultats bien qu'il semble idéal est parfaite, qui veut dire qu'ils ont parfaitement séparés les utilisateurs légitimes et polluant, n'est pas vraiment aussi bien que l'on suppose. Comme un résultat aussi parfait est irréaliste et indique du overfitting, signifiant que l'algorithme pourrait ne pas bien fonctionner sur des données nouvelles.

Pour l'analyse avec la classification bayésienne naïve, nous pouvons voir ceci :

- **Point fort** :Les résultats montrent une performance très élevée, même si un peu inférieure à celle des deux autres modèles, ceci pourrait indiquer que ce modèle est moins touché par le surajustement et pourrait mieux généraliser.
- **Point faible** : Le taux de FP (Faux Positifs) est plus élevé comparé aux deux autres modèles, ce qui indique une chance de classer incorrectement des non-pollueurs comme pollueurs, ce qui pourrait entraîner des erreurs de classification réel.

3.2 Sélection d'attributs

- **Affichage des 7 meilleurs attributs.**

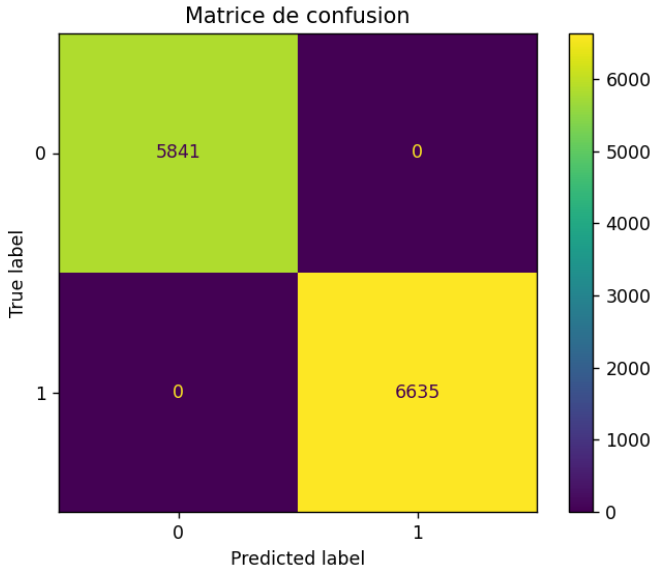
  Pour l'affichage des 7 meilleures attributs, nous executons «selection_information_gain.py », et on obtient ceci :

  - CreatedAt
  - std_followings
  - following_followers_ratio
  - NumberOfTweets
  - NumberOfFollowers
  - NumberOfFollowings
  - CollectedAt

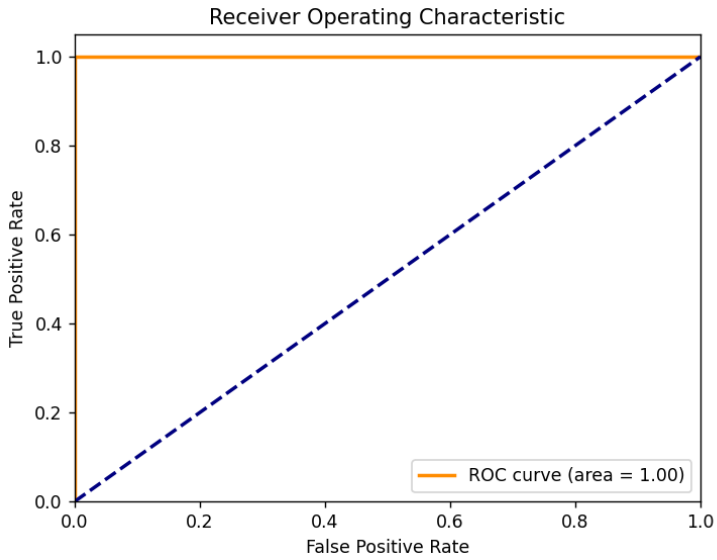
- **Affichage des résultats tel qu'indiquer dans l'énoncé. :**
  - Analyse avec l'arbres de décisions :
    - Accuracy: 1.0
    - Taux de VP (Vrais Positifs): 1.0
    - Taux de FP (Faux Positifs): 0.0
    - F-measure: 1.0
    - AUC: 1.0
    - Rapport de Clasification :

	precision	recall	f1-score	support
0	1.00	1.00	1.00	5841
1	1.00	1.00	1.00	6635
accuracy			1.00	12476
Macro avg	1.00	1.00	1.00	12476
Weighted avg	1.00	1.00	1.00	12476

- Matrice de confusion :



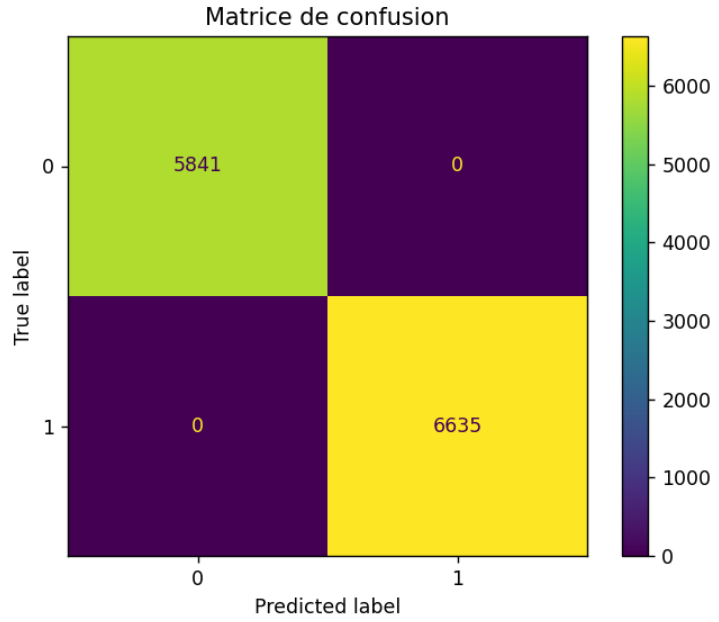
- Courbes ROC :



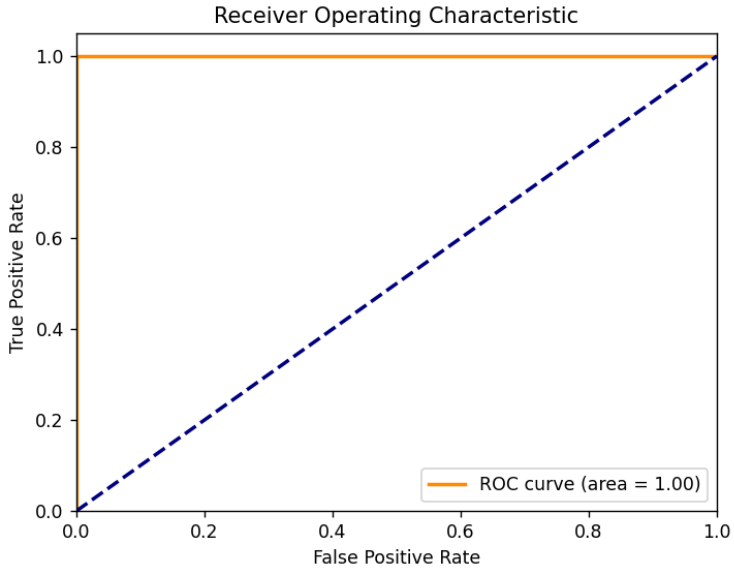
- Analyse avec la forêt aléatoire :
  - Accuracy: 1.0
  - Taux de VP (Vrais Positifs): 1.0
  - Taux de FP (Faux Positifs): 0.0
  - F-measure: 1.0
  - AUC: 1.0
  - Rapport de Classification :

	precision	recall	f1-score	support
0	1.00	1.00	1.00	5841
1	1.00	1.00	1.00	6635
accuracy			1.00	12476
Macro avg	1.00	1.00	1.00	12476
Weighted avg	1.00	1.00	1.00	12476

- Matrice de confusion :



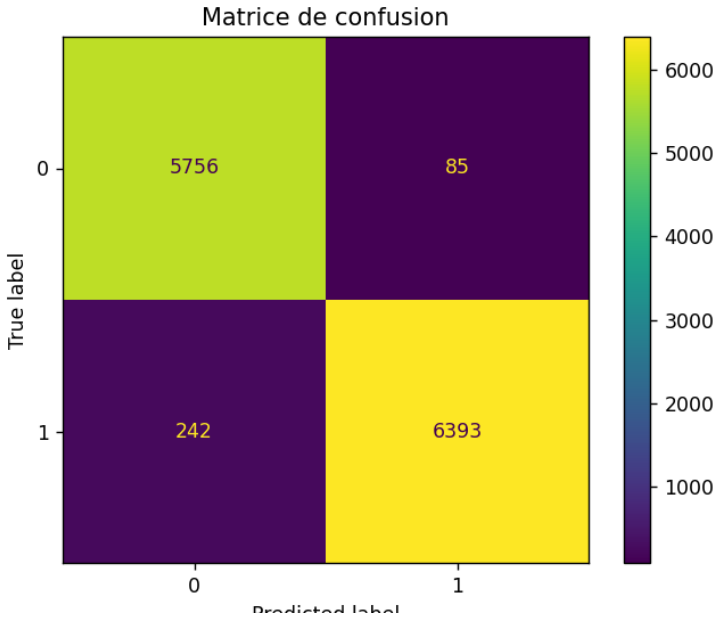
- Courbes ROC :



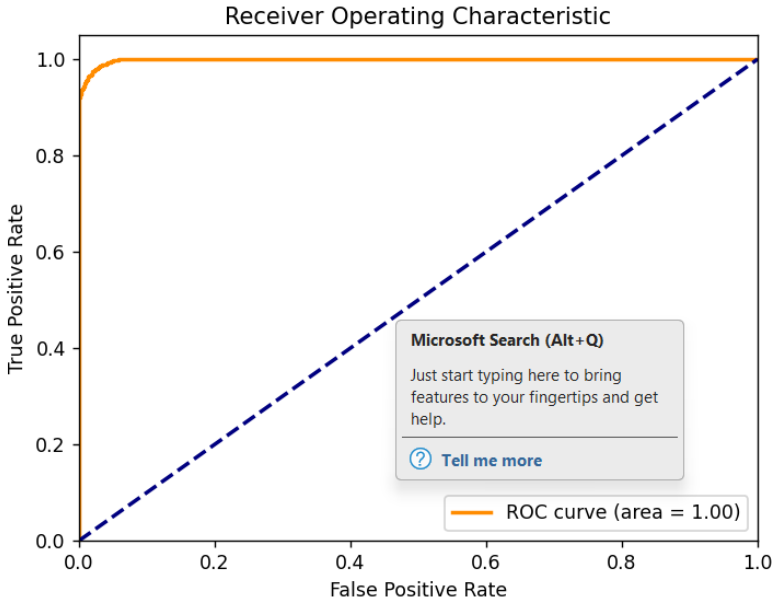
- Analyse avec la classification bayésienne naïve :
  - Accuracy: 0.9685
  - Taux de VP (Vrais Positifs): 0.9559
  - Taux de FP (Faux Positifs): 0.0171
  - F-measure: 0.9700
  - AUC: 0.9952
  - Rapport de Classification :

	precision	recall	f1-score	support
0	0.96	0.99	0.97	5841
1	0.99	0.96	0.98	6635
accuracy			0.97	12476
Macro avg	0.97	0.97	0.97	12476
Weighted avg	0.97	0.97	1.00	12476

- Matrice de confusion :



- Courbes ROC :



• Analyse des résultats (comparaison avec l'expérimentation qui implique l'utilisation de tout l'ensemble d'attributs) :

- En faisant une comparaison entre l'analyse comparative en utilisant tous les *features* et analyse comparative avec sélection d'attributs, nous pouvons voir que la différence est vraiment minime entre les 2 méthodes. Les métriques de performance telles que l'accuracy, le taux de vrais positifs, et l'AUC restent à des niveaux élevés, ce qui indique que les attributs les plus informatifs ont été conservés. Même si on peut voir une différence la taille du support.
  L'analyse comparative montre également la pertinence des attributs sélectionnés. La méthode du gain d'information a probablement écarté des attributs bruités ou redondants qui n'apportent pas de valeur

significative à la prédiction. Cela suggère que l'approche adoptée pour la sélection d'attributs est efficace et pourrait être utilisée pour identifier les caractéristiques les plus saillantes des données en vue d'autres analyses. Mais vue comme la différence des résultats obtenus restes minimes, le choix d'avoir des attribue spécifiques ou de tous ensemble d'attribut n'aurais pas un impact important sur nos résultats.

Conclusion

• Conclure votre rapport en discutant, d'une façon générale, les problèmes rencontrés ainsi que les démarches possibles qui peuvent considérés pour améliorer vos résultats.

Pour conclure ce rapport sur l'analyse comparative des algorithmes de machine learning, on peut premièrement, on a constaté que les performances parfaites des modèles d'arbres de décision et de forêt aléatoire pourraient suggérer du overfitting. Cette situation interpelle sur la nécessité de méthodes de validation plus solides, comme la validation avec des tests sur de nouveaux ensembles de données, pour vérifier la capacité des modèles à généraliser.
De plus, la classification bayésienne naïve, bien qu'un petits peu moins performante selon certaines métriques après la sélection d'attributs, fournit une perspective importante sur l'équilibre entre la performance et la généralisation. Une légère réduction dans les mesures de performance pourrait indiquer un modèle moins sujet au overfitting, ce qui est une qualité désirable.
On peut améliorer le code selon plusieurs démarches, nous pouvons considérer :

1. **L'expérimentation avec des hyperparamètres** : Un réglage fin des hyperparamètres des algorithmes pourrait permettre d'optimiser davantage les performances.
2. **La validation avec d'autres ensembles de données externes** : Tester les modèles sur des jeux de données extérieurs à ceux utilisés pour l'entraînement pendant test initial permettrait de mieux évaluer les capacités à généraliser.