INF7370

Apprentissage automatique

ADIL LABIAD

LABA06058003

ISSA BABAN CHAWAI, ABDOULAYE

ISSA23038902

Devoir 1

Université de Montréal à Québec

Hiver 2024

Contents

1 Introduction

Dans cette étude, nous avons pour objectif de comparer les performances de différents algorithmes de classification, notamment la Classification bayésienne naïve, l'Arbre de décision, la Forêt d'arbres décisionnels, Bagging, et l'Adaboost, appliqués à nos données. Notre but est d'analyser ces méthodes afin d'en déduire des conclusions pertinentes.

Pour réaliser cette comparaison, nous utiliserons le langage de programmation Python, accompagné de bibliothèques ci-dessous:

- `scikit-learn==1.3.2`: Cette bibliothèque sera employée pour mettre en œuvre l'apprentissage automatique.
- `Scikit-plot==0.3.7`: Utilisée pour la visualisation des données, elle nous permettra de modéliser et d'expérimenter avec nos résultats à différents stades de l'apprentissage automatique.
- `Numpy==1.24.4`: Cette bibliothèque est essentielle pour la manipulation de tableaux multidimensionnels et l'exécution d'opérations mathématiques sur ces tableaux.
- `Pandas==1.4.0`: Elle facilite la manipulation et l'analyse de données.
- `Matplotlib==3.7.5`: Nous l'utiliserons pour la création de graphiques, afin de visualiser les données de manière efficace.

L'étude vise à développer des modèles capables de classifier les restaurants en deux catégories distinctes : ceux fermés définitivement et ceux encore en activité. Pour ce faire, nous exploitons les données fournies par Yelp, une plateforme qui propose une sélection de restaurants adaptée aux préférences des clients (par exemple, le type de cuisine, les services offerts). Yelp permet également à ses utilisateurs de partager leurs expériences en attribuant des évaluations et commentaires sur les établissements. Notre base de données inclut diverses informations telles que la localisation géographique des restaurants, leurs horaires d'ouverture et de fermeture, ainsi que leur statut actuel (ouvert ou fermé), entre autres.

Pour atteindre notre objectif, nous avons établi une série d'étapes méthodiques, à savoir :

- Ingénierie des attributs: Nous avons calculé plusieurs attributs (tels que la moyenne des étoiles, le nombre de restaurants par zone, le nombre d'avis positifs) en exploitant les données disponibles (avis, catégories, checkin, conseils, horaires, utilisateurs, services).

- Prétraitement des données: Cette étape cruciale consiste à nettoyer et transformer les données brutes de Yelp afin de les rendre adaptées à l'analyse.
- Entraînement des modèles et analyse comparative: Nous appliquerons différents algorithmes de classification aux restaurants en utilisant les attributs préalablement calculés. Nous évaluerons et comparerons la performance de ces modèles à l'aide de plusieurs métriques, telles que la F-mesure, l'AUC (aire sous la courbe ROC), le taux de vrais positifs et le taux de faux positifs.
- Nous commencerons par utiliser l'ensemble des attributs identifiés puis, en se basant sur le gain d'information, nous sélectionnerons les dix meilleurs attributs pour appliquer les cinq algorithmes de classification. Nous comparerons leur performance en utilisant les métriques mentionnées précédemment, présenterons les résultats dans des tableaux, afficherons les matrices de confusion et fournirons une analyse critique des résultats.

Le projet sera organisé en trois fichiers principaux : features.py pour le calcul des attributs, processing.py pour le pré-traitement des données, et learning.py pour l'entraînement des différents modèles.

2 Calcul des attributs

Dans cette section, nous détaillons le calcul des attributs effectué pour chaque restaurant, en utilisant les données issues de notre fichier `restaurants.csv` ainsi que d'autres sources de données complémentaires. Voici un résumé des attributs calculés et des méthodologies employées :

- Attributs de base : Les attributs `moyenne_etoiles`, `ville`, `zone`, et `restaurant_id` sont directement récupérés à partir du fichier `restaurants.csv`.
- `nb_restaurants_zone` : Nous réalisons un groupement par `zone` et comptons le nombre de restaurants dans chaque `zone`.
- `zone_categories_intersection` : Nous effectuons une jointure entre la table restaurants et categories, suivie d'un groupement par `zone` et `categorie` pour compter le nombre total de catégories partagées par restaurant dans chaque `zone`.
- `ville_categories_intersection` : Nous effectuons une jointure entre la table restaurants et categories, suivie d'un groupement par `ville` et `categorie` pour compter le nombre total de catégories partagées par restaurant dans chaque `ville`.

- `nb_restaurant_meme_annee` : Après avoir filtré les données pour obtenir l'année de la première publication d'un avis comme année d'ouverture du restaurant, nous comptons le nombre de restaurants ouverts chaque année et associons chaque restaurant au nombre de restaurants ouverts la même année.
- `ecart_type_etoiles` et `tendance_etoiles` : Nous calculons l'écart type des moyennes d'étoiles annuelles pour chaque restaurant et déterminons la tendance des étoiles en comparant la moyenne des étoiles de la première à la dernière année.
- `nb_avis` : nous regroupons les avis en fonction de l'identifiant unique de chaque restaurant `restaurant_id` puis nous procédons au décompte du nombre total d'avis émis pour chaque établissement.
- `nb_avis_favorables` : Le total des avis positifs est calculé pour chaque restaurant.
- `nb_avis_defavorables` : Le total des avis négatifs est calculé pour chaque restaurant.
- `ratio_avis_favorables` : Après avoir calculé le nombre total d'avis pour chaque restaurant ainsi que le nombre d'avis positifs (calculé précédemment), le ratio d'avis favorables par rapport au nombre total d'avis est calculé pour chaque établissement.
- `ratio_avis_defavorables` : Après avoir calculé le nombre total d'avis négatifs pour chaque restaurant, le ratio d'avis défavorables par rapport au nombre total d'avis est calculé pour chaque établissement.
- `nb_avis_favorables_mention` : Nous filtrons d'abord les avis pour chaque restaurant, retenant ceux ayant un nombre d'étoiles supérieur ou égal à 3 (considérés comme favorables). Ensuite, parmi ces avis favorables, nous comptons ceux qui reçoivent une mention "useful", "funny", ou "cool", constituant ainsi notre critère de mention.
- `nb_avis_defavorables_mention` : Nous filtrons les avis pour chaque restaurant, retenant ceux ayant un nombre d'étoiles inférieur à 3 (considérés comme défavorables). Ensuite, parmi ces avis défavorables, nous comptons ceux qui reçoivent une mention "useful", "funny", ou "cool", constituant ainsi notre critère de mention.
- `nb_avis_favorables_elites` : Nous filtrons d'abord les avis favorables (étoiles $>= 3$) et ceux rédigés par des utilisateurs élites. Nous sélectionnons ensuite ces avis favorables émanant d'utilisateurs élites pour calculer enfin le nombre total de ces avis par restaurant.
- `nb_avis_defavorables_elites` : Nous filtrons d'abord les avis défavorables (étoiles < 3) et ceux rédigés par des utilisateurs élites. Nous sélectionnons ensuite ces avis défavorables émanant d'utilisateurs élites pour calculer enfin le nombre total de ces avis par restaurant.
- `nb_conseils` : Le total des conseils émis est calculé pour chaque restaurant.

- `nb_conseils_compliment` : Nous filtrons les conseils qui ont reçu au moins un compliment. Ensuite, le total de ces conseils complimentés est calculé pour chaque restaurant.

- `nb_conseils_elites` : Nous filtrons d'abord les utilisateurs élites. Ensuite, nous associons leurs conseils à travers une jointure et regroupons ces données par l'identifiant unique du restaurant "restaurant_id". Le décompte des conseils issus d'utilisateurs élites par restaurant est alors effectué.

- `nb_checkin` : Nous regroupons les entrées du fichier `check-in.csv` en fonction de l'identifiant du restaurant "restaurant_id" et procédons ensuite à un décompte des check-in pour chaque restaurant.

- `moyenne_checkin` : Nous calculons d'abord le nombre total de visites pour chaque restaurant sur une base annuelle. Ensuite, la moyenne de ces visites par an est déterminée pour chaque établissement.

- `ecart_type_checkin` : Nous commençons par déterminer le total annuel des visites pour chaque restaurant. Sur cette base, nous calculons ensuite l'écart type des visites annuelles pour chaque établissement.

- `chaine` : Nous regroupons d'abord les restaurants selon leur nom et comptabilisons les occurrences de chaque nom. Ensuite, nous attribuons la valeur 1 à l'attribut chaine pour les restaurants faisant partie d'une chaîne (identifiés par plus d'une occurrence de leur nom) et 0 pour ceux qui n'en font pas partie.

- `nb_heures_ouverture_semaine` : Nous mettons en place une fonction qui transforme les plages horaires en un total d'heures. Cette fonction prend en compte les situations où l'heure de fermeture dépasse minuit, traitant ces cas comme une ouverture de 12 heures supplémentaires pour s'adapter au format 24 heures. La conversion est réalisée pour chaque jour, permettant ainsi de calculer le total des heures d'ouverture sur l'ensemble de la semaine pour chaque restaurant.

- `ouvert_samedi` : Nous examinons la valeur attribuée à la colonne "samedi", résultant du calcul des heures d'ouverture hebdomadaires effectué par la fonction précédemment définie. Si cette valeur est supérieure ou égale à 1, cela indique que le restaurant est ouvert le samedi (valeur attribuée : 1) ; sinon, cela signifie qu'il est fermé ce jour-là (valeur attribuée : 0).

- `ouvert_dimanche` : nous examinons la valeur de la colonne "dimanche", résultant du calcul des heures d'ouverture hebdomadaires réalisé par la fonction mentionnée précédemment. Si cette valeur est supérieure ou égale à 1, cela indique que le restaurant est ouvert le dimanche (valeur attribuée : 1) ; dans le cas contraire, le restaurant est considéré comme fermé ce jour-là (valeur attribuée : 0).

- `ouvert_lundi` : nous examinons la valeur de la colonne "Lundi", résultant du calcul des heures d'ouverture hebdomadaire réalisé par la fonction mentionnée précédemment. Si cette valeur est supérieure ou égale à 1, cela signifie que le restaurant est ouvert le lundi (valeur attribuée : 1) ; dans le cas contraire, le restaurant est considéré comme fermé ce jour-là (valeur attribuée : 0).
- `ouvert_vendredi` : nous examinons la valeur de la colonne "vendredi", résultant du calcul des heures d'ouverture hebdomadaire réalisé par la fonction mentionnée précédemment. Si cette valeur est supérieure ou égale à 1, cela signifie que le restaurant est ouvert le vendredi (valeur attribuée : 1) ; dans le cas contraire, le restaurant est considéré comme fermé ce jour-là (valeur attribuée : 0).
- `emporter`, `livraison`, `bon_pour_groupes`, `bon_pour_enfants`, `reservation`, `prix` et `terrasse` : Nous extrayons les informations spécifiées dans la colonne correspondante du fichier de données `services.csv`.

3 Traitement du dataset

Dans cette section, plusieurs stratégies de pré-traitement des données ont été appliquées pour gérer les valeurs manquantes, comme indiqué ci-après :

- Pour les colonnes avec une distribution normale, telles que `moyenne_checkin`, `ratio_avis_favorables` et `ratio_avis_defavorables`, les valeurs manquantes ont été remplacées par la moyenne.
- Pour les valeurs manquantes catégorielles, la stratégie adoptée consiste à remplacer ces valeurs par le mode. Cela s'applique, par exemple, à la colonne `prix`, permettant ainsi de conserver la tendance générale de la catégorie sans altérer la distribution des données.
- Les colonnes caractérisées par le nombre d'occurrences, telles que `nb_avis_favorables`, `nb_avis_defavorables`, `nb_avis_favorables_mention`, `nb_avis_favorables_elites`, etc., sont remplacées par zéro.

Il est également important de noter deux décisions spécifiques prises dans le traitement :

- Les valeurs manquantes des colonnes `ecart_type_etoiles` et `ecart_type_checkin` ont été systématiquement remplacées par zéro, en tenant compte du fait que certains restaurants peuvent ne disposer de données sur les étoiles que pour une seule année, rendant ainsi l'écart type inapplicable ou trompeur.

- Les lignes présentant des valeurs nulles dans la colonne `zone` ont été supprimées. Cette action est justifiée par le besoin d'éviter l'introduction de biais dans l'analyse en excluant les données incomplètes, surtout lorsque la localisation géographique est cruciale pour l'analyse.

Colonne	zéro	moyenne	mode
ecart_type_etoiles	X	-	-
nb_avis_favorables	X	-	-
nb_avis_defavorables	X	-	-
ratio_avis_favorables	-	X	-
ratio_avis_defavorables	-	X	-
nb_avis_favorables_mention	X	-	-
nb_avis_defavorables_mention	X	-	-
nb_avis_favorables_elites	X	-	-
nb_avis_defavorables_elites	X	-	-
nb_conseils	X	-	-
nb_conseils_compliment	X	-	-
nb_conseils_elites	X	-	-
nb_checkin	X	-	-
moyenne_checkin	-	X	-
ecart_type_checkin	X	-	-
chaine	X	-	-
terrasse	X	-	-
prix	-	-	X

Table 1: Tableau récapitulatif du traitement effectué sur chaque colonne.

4 Entraînement des modèles et analyse comparative

4.1 Utilisation de tout l'ensemble d'attributs

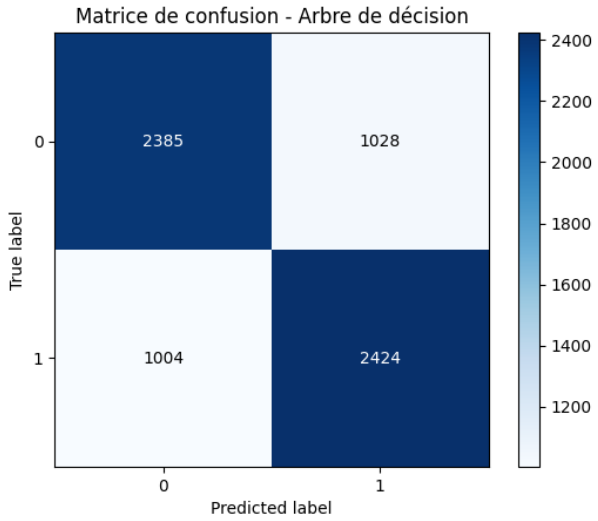
Dans cette section, nous analysons les performances de différents modèles de classification pour prédire la fermeture définitive des restaurants. L'objectif principal de cette analyse est d'évaluer et de comparer les performances de ces modèles en utilisant l'ensemble des 37 attributs disponibles dans notre jeu de données.

Nous avons entraîné plusieurs modèles de classification sur nos données, y compris l'arbre de décision, la forêt aléatoire, le Naive Bayes, le Bagging, et AdaBoost en utilisant les paramètres (hyperparamètres) par défaut. Avant d'entraîner les modèles, nous avons d'abord effectué deux étapes de pré-traitement des données. Tout d'abord, nous avons normalisé les données en utilisant le Z-score, également appelé StandardScaler dans Scikit-learn, afin de centrer et de mettre à l'échelle les caractéristiques. Ensuite, nous avons divisé nos données en ensembles d'entraînement et de test. Dans notre configuration, nous avons choisi de réserver 20% de nos données pour le test, tandis que les 80% restants ont été utilisés pour l'entraînement des modèles. Cette division nous permet d'évaluer les performances des modèles sur des données indépendantes.

Nous présentons ensuite les résultats détaillés de chaque modèle, ainsi que des analyses approfondies des performances de prédiction. Nous évaluons également les forces et les faiblesses de chaque modèle en nous basant sur les critères suivants :

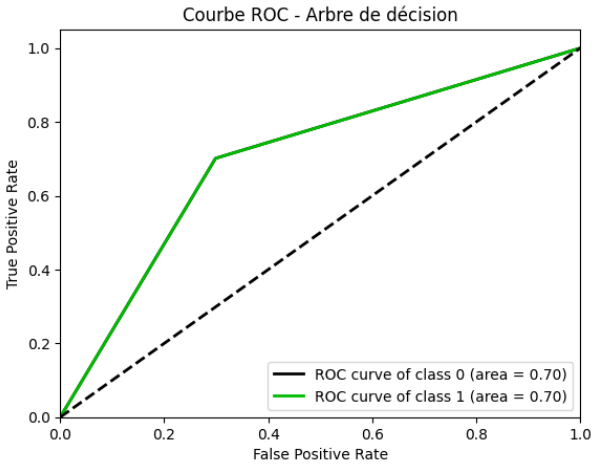
- Le taux de vrais positifs (TP Rate) de la classe Restaurants fermés définitivement.
- Le taux de faux positifs (FP Rate) de la classe Restaurants fermés définitivement.
- F-measure de la classe Restaurants fermés définitivement.
- La surface sous la courbe ROC (AUC).

4.1.1 Analyses détaillées



(a) Matrice de confusion

La matrice de confusion indique que le modèle classe correctement 2385 restaurants ouverts et 2424 fermés. Toutefois, il classe à tort 1028 restaurants ouverts comme fermés et 1004 fermés comme ouverts.



(b) La courbe ROC

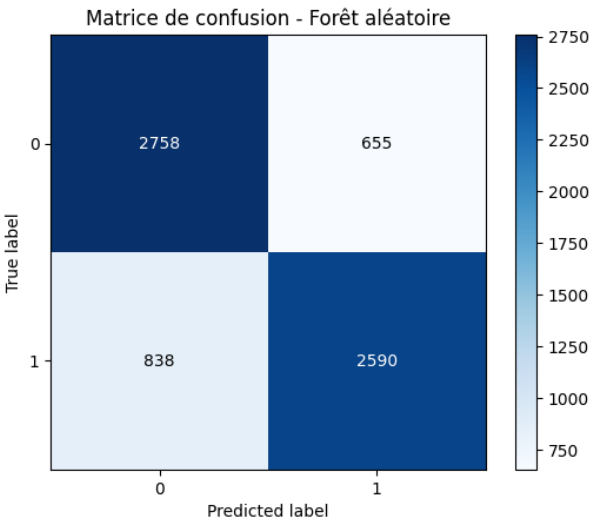
La courbe ROC avec une AUC de 0.70 illustre une capacité acceptable du modèle à séparer les classes de restaurants ouverts et fermés. Cette valeur suggère que le modèle a une probabilité de 70% de classer correctement un cas aléatoire, ce qui est supérieur à un choix au hasard.

Figure 1: Évaluation du modèle (Arbre de décision)

La table 2 résume les performances du modèle dans la classification des restaurants ouverts et fermés. Avec un taux de vrais positifs de 0.71, le modèle a correctement identifié la majorité des restaurants fermés. Cependant, son taux de faux positifs de 0.30 indique qu'il a également classé à tort un pourcentage notable de restaurants ouverts comme fermés. Pourtant, la F-mesure globale de 0.71 suggère une performance équilibrée dans la prédiction des deux classes, démontrant ainsi une capacité satisfaisante du modèle à distinguer entre les restaurants ouverts et fermés.

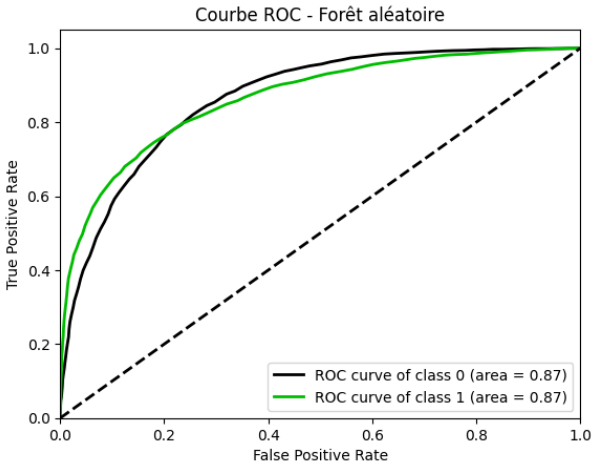
Modèle	TP Rate	FP Rate	F-mesure
Arbre de décision	0.71	0.30	0.71

Table 2: Performance du modèle (Arbre de décision)



(a) Matrice de confusion

La matrice de confusion indique que le modèle classe correctement 2758 restaurants ouverts et 2590 fermés. Toutefois, il classe à tort 655 restaurants ouverts comme fermés et 838 fermés comme ouverts.



(b) La courbe ROC

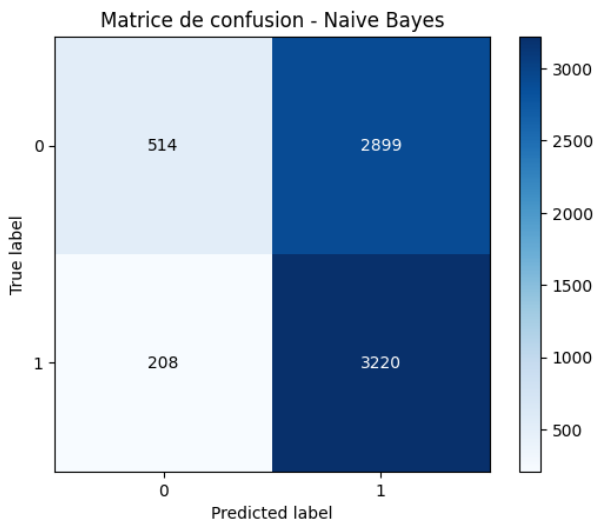
La courbe ROC avec une AUC de 0.87 illustre une meilleure capacité du modèle à séparer les classes de restaurants ouverts et fermés. Cette valeur suggère que le modèle a une probabilité de 87% de classer correctement un cas aléatoire, ce qui est supérieur à un choix au hasard.

Figure 2: Évaluation du modèle (Forêt aléatoire)

La table 3 résume les performances du modèle dans la classification des restaurants ouverts et fermés. Avec un taux de vrais positifs de 0.76, le modèle a correctement identifié une grande partie des restaurants fermés. Cependant, son faible taux de faux positifs de 0.19 indique qu'il a classé à tort un pourcentage relativement faible de restaurants ouverts comme fermés. La F-mesure globale de 0.78 suggère une performance solide dans la prédiction des deux classes, démontrant ainsi une capacité satisfaisante du modèle à distinguer entre les restaurants ouverts et fermés.

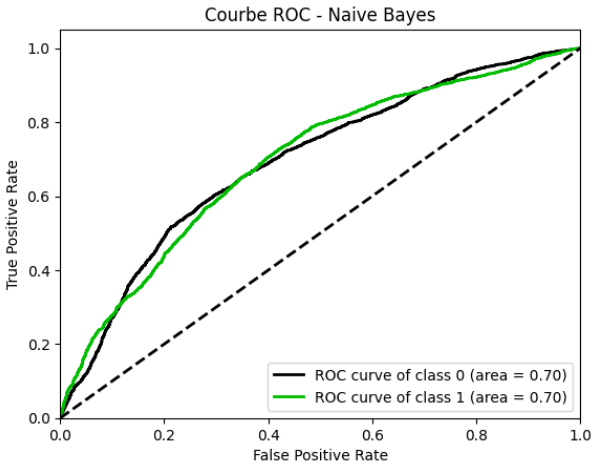
Modèle	TP Rate	FP Rate	F-mesure
Forêt aléatoire	0.76	0.19	0.78

Table 3: Performance du modèle (Forêt aléatoire)



(a) Matrice de confusion

La matrice de confusion indique que le modèle classe correctement 514 restaurants ouverts et 3220 fermés. Toutefois, il classe à tort 2899 restaurants ouverts comme fermés et 208 fermés comme ouverts.



(b) La courbe ROC

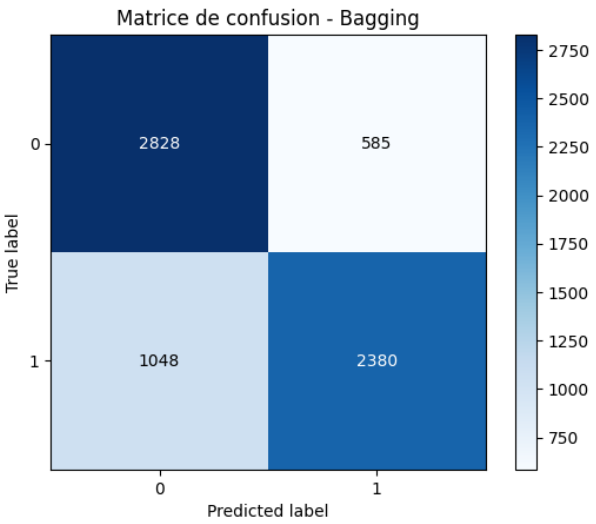
La courbe ROC avec une AUC de 0.70 illustre une capacité acceptable du modèle à séparer les classes de restaurants ouverts et fermés. Cette valeur suggère que le modèle a une probabilité de 70% de classer correctement un cas aléatoire, ce qui est supérieur à un choix au hasard.

Figure 3: Évaluation du modèle (Naive Bayes)

La table 4 résume les performances du modèle dans la classification des restaurants ouverts et fermés. Avec un taux de vrais positifs de 0.94, le modèle a correctement identifié la majorité des restaurants fermés. Cependant, son taux de faux positifs élevé de 0.85 indique qu'il a également classé à tort un pourcentage important de restaurants ouverts comme fermés. La F-mesure globale de 0.67 suggère une performance déséquilibrée dans la prédiction des deux classes, démontrant ainsi une capacité limitée du modèle à distinguer entre les restaurants ouverts et fermés.

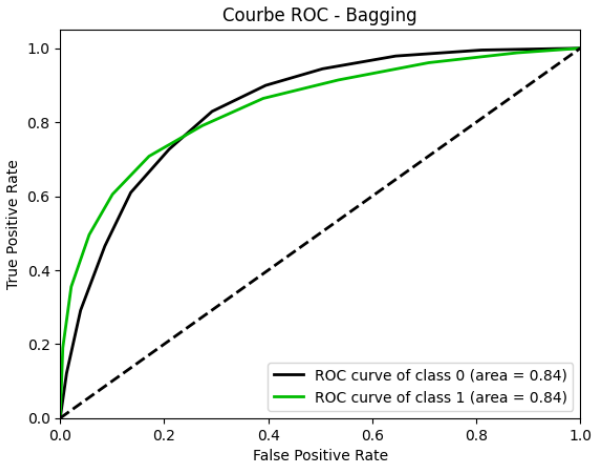
Modèle	TP Rate	FP Rate	F-mesure
Naive Bayes	0.94	0.85	0.67

Table 4: Performance du modèle (Naive Bayes)



(a) Matrice de confusion

La matrice de confusion indique que le modèle classe correctement 2828 restaurants ouverts et 2380 fermés. Toutefois, il classe à tort 585 restaurants ouverts comme fermés et 1048 fermés comme ouverts.



(b) La courbe ROC

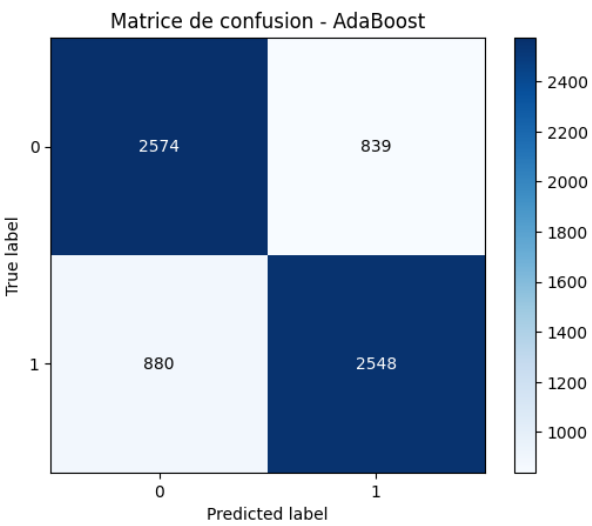
La courbe ROC avec une AUC de 0.84 illustre une bonne capacité du modèle à séparer les classes de restaurants ouverts et fermés. Cette valeur suggère que le modèle a une probabilité de 84% de classer correctement un cas aléatoire, ce qui est supérieur à un choix au hasard.

Figure 4: Évaluation du modèle (Bagging)

La table 5 résume les performances du modèle dans la classification des restaurants ouverts et fermés. Avec un taux de vrais positifs de 0.71, le modèle a correctement identifié une proportion significative des restaurants fermés. Cependant, son faible taux de faux positifs de 0.17 indique qu'il a classé à tort un pourcentage relativement faible de restaurants ouverts comme fermés. La F-mesure globale de 0.75 suggère une performance solide dans la prédiction des deux classes, démontrant ainsi une capacité efficace du modèle à distinguer entre les restaurants ouverts et fermé.

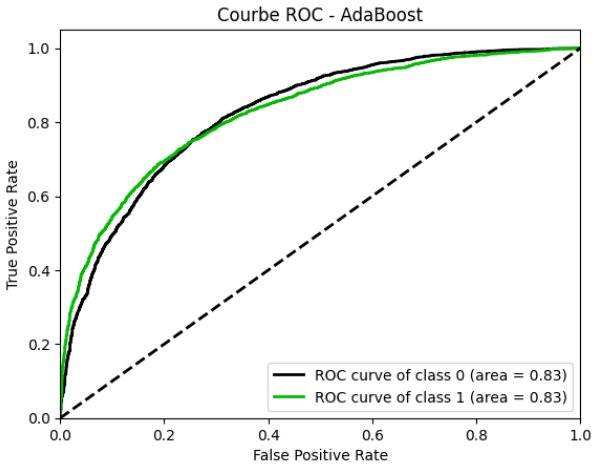
Modèle	TP Rate	FP Rate	F-mesure
Bagging	0.71	0.17	0.75

Table 5: Performance du modèle (Bagging)



(a) Matrice de confusion

La matrice de confusion indique que le modèle classe correctement 2574 restaurants ouverts et 2548 fermés. Toutefois, il classe à tort 839 restaurants ouverts comme fermés et 880 fermés comme ouverts.



(b) La courbe ROC

La courbe ROC avec une AUC de 0.83 illustre une bonne capacité du modèle à séparer les classes de restaurants ouverts et fermés. Cette valeur suggère que le modèle a une probabilité de 83% de classer correctement un cas aléatoire, ce qui est supérieur à un choix au hasard.

Figure 5: Évaluation du modèle (AdaBoost)

La table 6 résume les performances du modèle dans la classification des restaurants ouverts et fermés. Avec un taux de vrais positifs de 0.74, le modèle a correctement identifié une proportion importante des restaurants fermés. Cependant, son taux de faux positifs de 0.25 indique qu'il a classé à tort un pourcentage relativement modéré de restaurants ouverts comme fermés. La F-mesure globale de 0.75 suggère une performance solide dans la prédiction des deux classes, démontrant ainsi une capacité efficace du modèle à distinguer entre les restaurants ouverts et fermés.

Modèle	TP Rate	FP Rate	F-mesure
AdaBoost	0.74	0.25	0.75

Table 6: Performance du modèle (AdaBoost)

4.1.2 Synthèse de l'analyse

L'analyse détaillée des performances de différents algorithmes fournit un aperçu précieux de leurs capacités et limites dans le contexte de la classification des restaurants ouverts et fermés. En utilisant les métriques de performance et les résultats des matrices de confusion spécifiques à chaque modèle, nous pouvons dresser un tableau comparatif 7 des forces et des faiblesses inhérentes à chaque approche.

Modèle	TP Rate	FP Rate	F-mesure	AUC
Arbre de décision	0.70	0.30	0.70	0.70
Forêt aléatoire	0.76	0.19	0.78	0.87
Naive Bayes	0.94	0.85	0.67	0.70
Bagging	0.71	0.17	0.75	0.84
AdaBoost	0.74	0.25	0.75	0.83

Table 7: Performance des modèles avec tous les attributs

- **Arbre de décision**: Ce modèle présente un équilibre avec un taux de vrais positifs et une F-mesure de 0.70, indiquant une performance modérée. Sa principale faiblesse est un taux de faux positifs de 0.30, illustrant une tendance à classifier à tort un nombre significatif de négatifs comme positifs, ce qui peut limiter son utilité dans des contextes nécessitant une grande spécificité. L'AUC de 0.70 reflète sa capacité juste à distinguer les classes, mais suggère aussi une marge d'amélioration en termes de précision et de rappel. De plus, sa vulnérabilité au sur-apprentissage, particulièrement avec une profondeur élevée, peut compromettre sa capacité de généralisation

- **Forêt aléatoire**: Ce modèle excelle avec un taux de vrais positifs de 0.76 et une F-mesure de 0.78, montrant une forte capacité à identifier correctement les cas positifs tout en maintenant un bon équilibre entre précision et rappel. Son AUC élevé à 0.87 indique une excellente aptitude à distinguer entre les classes. Néanmoins, la Forêt aléatoire peut être exigeante en termes de ressources de calcul, en particulier pour des ensembles de données volumineux, ce qui peut poser problème dans des environnements avec des ressources limitées ou nécessitant des temps de réponse rapides.

- **Naive Bayes**: Ce modèle excelle dans la sensibilité (TP Rate de 0.94) mais son FP Rate extrêmement élevé à 0.85 et une F-mesure de 0.67 indiquent un manque de spécificité, résultant en une classification excessive des instances négatives comme positives. Ce déséquilibre le rend moins adapté pour des applications où les faux positifs ont des conséquences graves. Malgré son AUC de 0.70, la simplicité du modèle et la supposition d'indépendance peuvent réduire sa précision dans des cas de données complexes et interdépendantes.

- **Bagging**: Ce modèle montre une bonne performance globale avec un équilibre entre sensibilité et spécificité, reflété par son AUC de 0.84. Toutefois, comme pour la Forêt aléatoire, la complexité et le coût computationnel peuvent être des inconvénients, particulièrement dans le traitement de grands ensembles de données ou pour des applications nécessitant une réponse rapide.
- **AdaBoost**: Ce modèle offre une bonne balance entre sensibilité et spécificité avec une F-mesure de 0.75 et un AUC de 0.83. Néanmoins, sa sensibilité aux données bruitées et aux valeurs aberrantes peut représenter un point faible, pouvant conduire à un surajustement dans certaines situations. De plus, le processus d'ajustement séquentiel des poids peut augmenter la complexité de calcul, rendant le modèle moins pratique pour des applications à grande échelle.

L'examen de la performance de ces algorithmes pour classer les restaurants ouverts et fermés met en évidence des aspects critiques de leur utilisation pratique. La forêt aléatoire se distingue par sa robustesse et sa capacité à bien généraliser, ce qui en fait un choix préférentiel lorsque la précision et la capacité de généralisation sont primordiales. Cependant, elle peut être exigeante en ressources, ce qui peut poser problème lorsque les ressources sont limitées.

L'arbre de décision, tout en étant plus simple et facile à interpréter, peut souffrir de surajustement. Ce risque peut être mitigé par une attention particulière à l'élagage et à la sélection de caractéristiques, ainsi qu'à l'ajustement de la profondeur de l'arbre. Le Naive Bayes, malgré sa tendance à mal classer les cas négatifs dans ce contexte spécifique, peut être avantageux dans des situations où une haute sensibilité est requise, bien qu'au détriment de la spécificité.

Le Bagging et AdaBoost offrent un bon compromis en termes de performance et de risque de surajustement, avec une capacité notable à améliorer la classification par rapport à un modèle unique. Cependant, leur efficacité peut varier en fonction de la nature des données et de la présence de bruit. Il est important de noter que AdaBoost peut être particulièrement sensible aux données bruitées et aux valeurs aberrantes, ce qui nécessite une évaluation attentive.

Le choix du modèle le plus adapté dépend de l'équilibre entre la nécessité de bien classer les cas (efficacité), la capacité à s'adapter à de nouvelles données (généralisation) et les ressources disponibles. Une compréhension approfondie de chaque modèle et de ses caractéristiques, accompagnée de l'utilisation de techniques d'évaluation comme la validation croisée, est essentielle pour choisir l'approche la plus appropriée à un problème donné.

4.2 Sélection d'attributs

L'application de la fonction mutual_info_classif de scikit-learn à travers SelectKBest, en spécifiant k=10, a permis d'identifier les 10 caractéristiques les plus pertinents. Cette démarche, basée sur

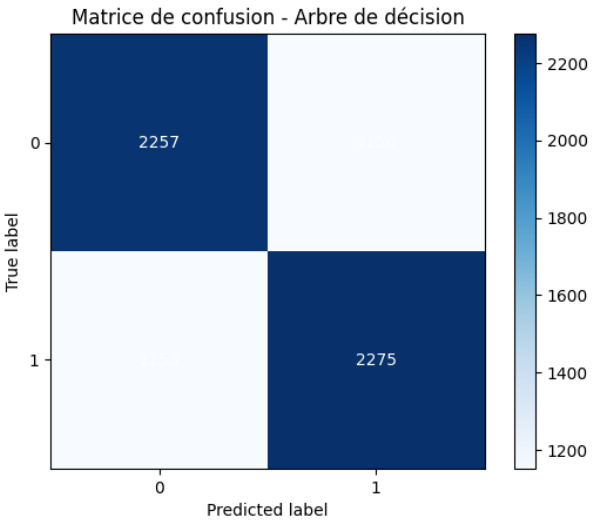
le Gain d'information, a facilité la sélection des attributs les plus pertinents pour une analyse plus ciblée, comme détaillé dans le Tableau 8.

Cette sélection se concentre sur les attributs offrant le plus grand potentiel d'impact sur la classification des restaurants, permettant d'affiner les modèles prédictifs précédemment élaborés avec l'ensemble des caractéristiques. En se focalisant sur ces attributs spécifiques, l'analyse vise à comparer l'efficacité des modèles de classification dans un cadre plus restreint, offrant une perspective enrichie sur la pertinence des données sélectionnées pour la prédiction.

Indice	Feature	Score
22	moyenne_checkin	0.061555
23	ecart_type_checkin	0.053150
25	nb_heures_ouverture_semaine	0.038147
6	nb_restaurant_meme_annee	0.036121
9	nb_total_avis	0.028671
12	ratio_avis_favorables	0.028561
27	ouvert_dimanche	0.028548
7	ecart_type_etoiles	0.028284
29	ouvert_vendredi	0.028277
26	ouvert_samedi	0.025990

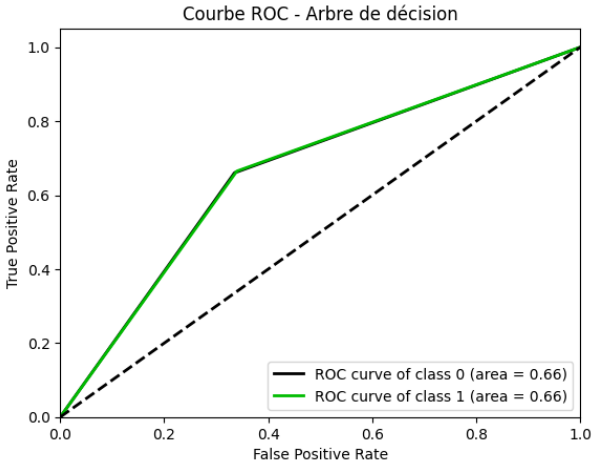
Table 8: 10 meilleurs attributs

4.2.1 Analyses détaillées



(a) Matrice de confusion

La matrice de confusion indique que le modèle classe correctement 2257 restaurants ouverts et 2275 fermés. Toutefois, il classe à tort 1156 restaurants ouverts comme fermés et 1153 fermés comme ouverts.



(b) La courbe ROC

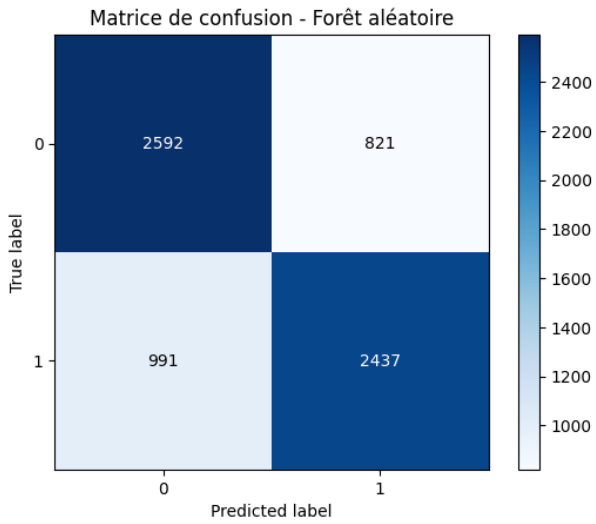
La courbe ROC avec une AUC de 0.66 illustre une capacité acceptable du modèle à séparer les classes de restaurants ouverts et fermés. Cette valeur suggère que le modèle a une probabilité de 66% de classer correctement un cas aléatoire, ce qui est supérieur à un choix au hasard.

Figure 6: Évaluation du modèle réduit (Arbre de décision)

La table 9 résume les performances du modèle dans la classification des restaurants ouverts et fermés. Avec un taux de vrais positifs de 0.66, le modèle a correctement identifié une proportion significative des restaurants fermés. Cependant, son taux de faux positifs de 0.34 indique qu'il a également classé à tort un pourcentage notable de restaurants ouverts comme fermés. Pourtant, la F-mesure globale de 0.66 suggère une performance équilibrée dans la prédiction des deux classes, démontrant ainsi une capacité réduite du modèle à distinguer entre les restaurants ouverts et fermés.

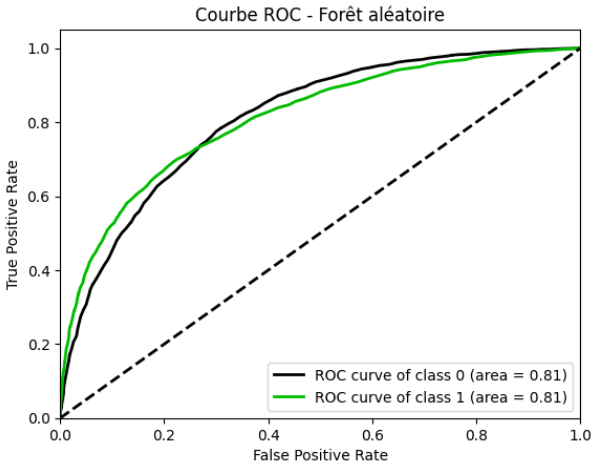
Modèle	TP Rate	FP Rate	F-mesure
AdaBoost	0.66	0.34	0.66

Table 9: Performance du modèle réduit (Arbre de décision))



(a) Matrice de confusion

La matrice de confusion indique que le modèle classe correctement 2592 restaurants ouverts et 2437 fermés. Toutefois, il classe à tort 821 restaurants ouverts comme fermés et 991 fermés comme ouverts.



(b) La courbe ROC

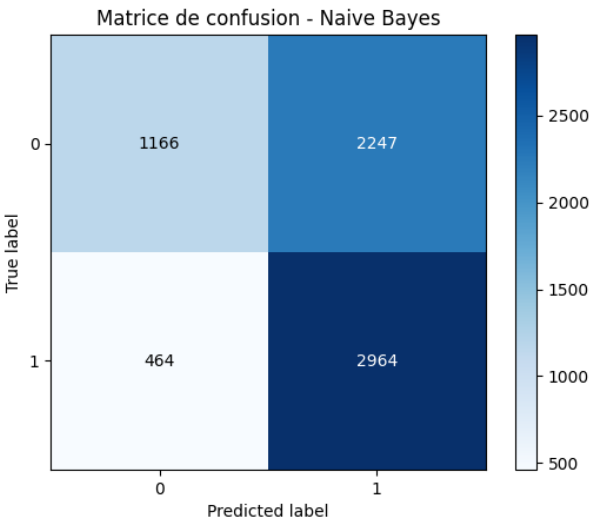
La courbe ROC avec une AUC de 0.81 illustre une meilleure capacité du modèle à séparer les classes de restaurants ouverts et fermés. Cette valeur suggère que le modèle a une probabilité de 81% de classer correctement un cas aléatoire, ce qui est supérieur à un choix au hasard.

Figure 7: Évaluation du modèle réduit (Forêt aléatoire)

La table 10 résume les performances du modèle dans la classification des restaurants ouverts et fermés. Avec un taux de vrais positifs de 0.72, le modèle a correctement identifié une grande partie des restaurants fermés. Cependant, son faible taux de faux positifs de 0.24 indique qu'il a classé à tort un pourcentage relativement faible de restaurants ouverts comme fermés. La F-mesure globale de 0.73 suggère une performance solide dans la prédiction des deux classes, démontrant ainsi une capacité satisfaisante du modèle à distinguer entre les restaurants ouverts et fermés.

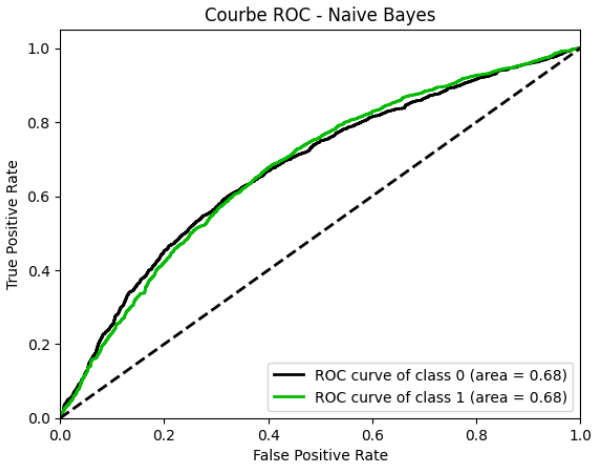
Modèle	TP Rate	FP Rate	F-mesure
Forêt aléatoire	0.72	0.24	0.73

Table 10: Performance du modèle réduit (Forêt aléatoire)



(a) Matrice de confusion

La matrice de confusion indique que le modèle classe correctement 1166 restaurants ouverts et 2964 fermés. Toutefois, il classe à tort 2247 restaurants ouverts comme fermés et 464 fermés comme ouverts.



(b) La courbe ROC

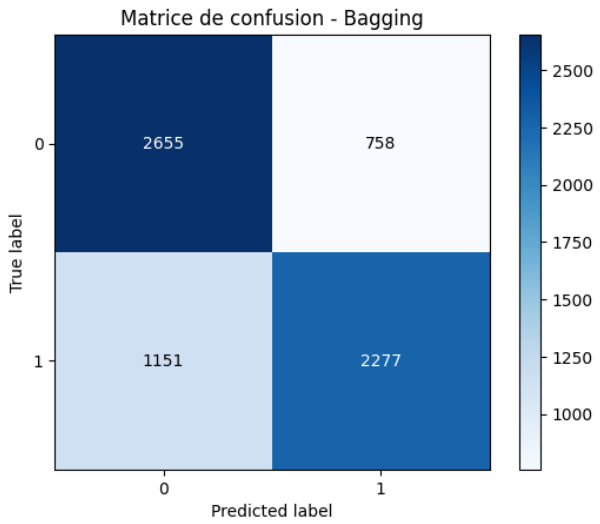
La courbe ROC avec une AUC de 0.68 illustre une capacité acceptable du modèle à séparer les classes de restaurants ouverts et fermés. Cette valeur suggère que le modèle a une probabilité de 68% de classer correctement un cas aléatoire, ce qui est supérieur à un choix au hasard.

Figure 8: Évaluation du modèle réduit (Naive Bayes)

La table 11 résume les performances du modèle dans la classification des restaurants ouverts et fermés. Avec un taux de vrais positifs de 0.94, le modèle a correctement identifié la majorité des restaurants fermés. Cependant, son taux de faux positifs élevé de 0.85 indique qu'il a classé à tort un pourcentage élevé de restaurants ouverts comme fermés. La F-mesure globale de 0.67 suggère une performance acceptable dans la prédiction des deux classes, démontrant ainsi une capacité limitée du modèle à distinguer entre les restaurants ouverts et fermés.

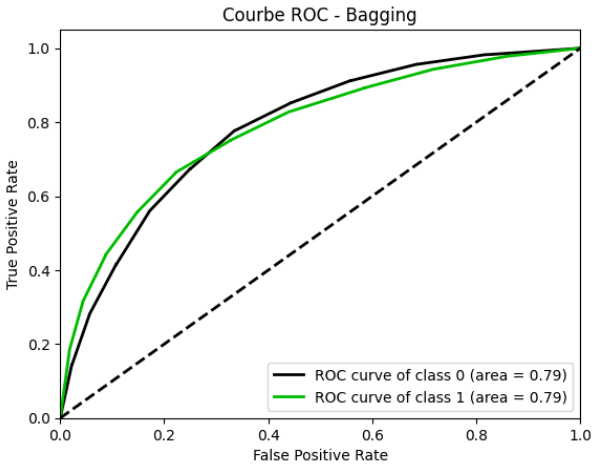
Modèle	TP Rate	FP Rate	F-mesure
Forêt aléatoire	0.94	0.85	0.67

Table 11: Performance du modèle réduit (Naive Bayes)



(a) Matrice de confusion

La matrice de confusion indique que le modèle classe correctement 2655 restaurants ouverts et 2277 fermés. Toutefois, il classe à tort 758 restaurants ouverts comme fermés et 1151 fermés comme ouverts.



(b) La courbe ROC

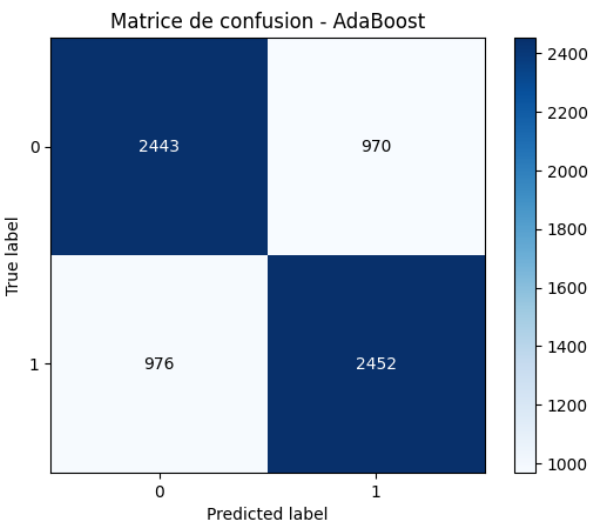
La courbe ROC avec une AUC de 0.79 illustre une bonne capacité du modèle à séparer les classes de restaurants ouverts et fermés. Cette valeur suggère que le modèle a une probabilité de 79% de classer correctement un cas aléatoire, ce qui est supérieur à un choix au hasard.

Figure 9: Évaluation du modèle réduit (Bagging)

La table 12 résume les performances du modèle dans la classification des restaurants ouverts et fermés. Avec un taux de vrais positifs de 0.66, le modèle a correctement identifié une proportion significative des restaurants fermés. Cependant, son taux relativement faible de faux positifs de 0.22 indique qu'il a classé à tort un pourcentage relativement faible de restaurants ouverts comme fermés. La F-mesure globale de 0.70 suggère une performance solide dans la prédiction des deux classes, démontrant ainsi une capacité efficace du modèle à distinguer entre les restaurants ouverts et fermé.

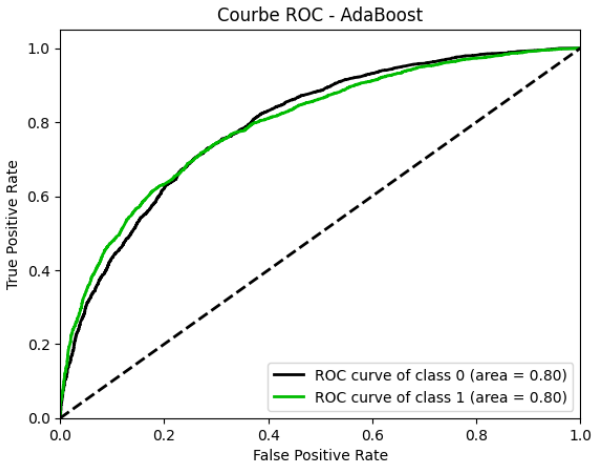
Modèle	TP Rate	FP Rate	F-mesure
Bagging	0.66	0.22	0.70

Table 12: Performance du modèle réduit (Bagging)



(a) Matrice de confusion

La matrice de confusion indique que le modèle classe correctement 2443 restaurants ouverts et 2452 fermés. Toutefois, il classe à tort 970 restaurants ouverts comme fermés et 976 fermés comme ouverts.



(b) La courbe ROC

La courbe ROC avec une AUC de 0.80 illustre une bonne capacité du modèle à séparer les classes de restaurants ouverts et fermés. Cette valeur suggère que le modèle a une probabilité de 80% de classer correctement un cas aléatoire, ce qui est supérieur à un choix au hasard.

Figure 10: Évaluation du modèle réduit (AdaBoost)

La table 13 résume les performances du modèle dans la classification des restaurants ouverts et fermés. Avec un taux de vrais positifs de 0.73, le modèle a correctement identifié une proportion importante des restaurants fermés. Cependant, son taux de faux positifs de 0.28 indique qu'il a classé à tort un pourcentage relativement élevé de restaurants ouverts comme fermés. La F-mesure globale de 0.72 suggère une performance solide dans la prédiction des deux classes, démontrant ainsi une capacité efficace du modèle à distinguer entre les restaurants ouverts et fermés.

Modèle	TP Rate	FP Rate	F-mesure
AdaBoost	0.73	0.28	0.72

Table 13: Performance du modèle réduit (AdaBoost)

4.2.2 Synthèse de l'analyse

Modèle	TP Rate	FP Rate	F-mesure	AUC
Arbre de décision	0.66	0.33	0.67	0.67
Forêt aléatoire	0.71	0.24	0.73	0.81
Naive Bayes	0.89	0.72	0.68	0.68
Bagging	0.66	0.22	0.70	0.78
AdaBoost	0.72	0.28	0.72	0.79

Table 14: Performance des modèles après sélection d'attributs

La comparaison des performances des modèles avant et après la sélection des 10 meilleurs attributs via le Gain d'information révèle plusieurs points clés sur l'efficacité de la réduction de dimensionnalité et son impact sur différents algorithmes.

La performance de l'Arbre de décision montre une légère baisse après la sélection des attributs, indiquant que certains attributs éliminés portaient des informations pertinentes pour la classification. La Forêt aléatoire et AdaBoost montrent une résilience notable, avec des performances relativement stables, suggérant une capacité à maintenir une bonne généralisation même avec moins d'attributs. Le Naive Bayes voit son taux de vrais positifs diminuer légèrement, ce qui pourrait refléter la perte d'attributs contribuant à sa sensibilité élevée dans le modèle complet.

L'observation la plus notable est la variation des taux de faux positifs (FP Rate) et des mesures d'efficacité comme la F-mesure et l'AUC. Pour la plupart des modèles, ces indicateurs restent comparables ou connaissent une légère variation, ce qui suggère que la sélection d'attributs a réussi à condenser l'information nécessaire à la classification sans sacrifier significativement la performance.

La Forêt aléatoire et AdaBoost se distinguent par leur capacité à bien performer malgré la réduction de la dimensionnalité, ce qui les rend particulièrement adaptés pour les contextes où la rapidité de calcul et la simplicité du modèle sont cruciales. Le Naive Bayes, malgré une baisse de performance, reste un choix intéressant pour des situations où la haute sensibilité est prioritaire, bien que cela puisse venir au détriment de la précision due à une augmentation des faux positifs.

En résumé, la sélection des 10 meilleurs attributs montre qu'il est possible de simplifier les modèles sans compromettre de manière significative leur capacité de prédiction. Cette approche peut offrir un avantage considérable dans des environnements à ressources limitées ou lorsque l'interprétabilité est une priorité. Toutefois, il est crucial d'évaluer l'impact de la réduction de dimensionnalité au cas par cas, car l'importance des attributs peut varier selon le modèle et le contexte d'application spécifique.

5 Conclusion

Dans ce rapport, nous avons exploré la performance de différents modèles, notamment l'Arbre de décision, la Forêt aléatoire, le Naive Bayes, le Bagging et AdaBoost, à travers deux scénarios distincts : l'utilisation de l'ensemble complet des attributs et la sélection des 10 meilleurs attributs via le Gain d'information de scikit-learn. Cette étude visait à comprendre l'impact de la réduction de dimensionnalité sur la capacité prédictive de ces modèles dans le contexte de la classification des restaurants ouverts et fermés.

Les résultats ont révélé des variations notables dans la performance des modèles suite à la réduction des attributs, mettant en lumière la complexité inhérente à la sélection d'attributs et son influence sur la précision des prédictions. Il a été observé que, bien que certains modèles, comme l'Arbre de décision, aient subi une légère baisse de performance, d'autres, tels que la Forêt aléatoire et AdaBoost, ont maintenu une robustesse relative, démontrant ainsi leur capacité à extraire efficacement les informations pertinentes même d'un nombre réduit d'attributs.

La conclusion principale de cette étude est que la sélection des 10 meilleurs attributs constitue une stratégie efficace pour optimiser les ressources computationnelles sans sacrifier de manière significative l'efficacité des modèles. Cette approche est particulièrement bénéfique dans des contextes où les contraintes de temps ou de ressources limitent l'utilisation de modèles plus complexes. De plus, la simplification des modèles par la réduction de dimensionnalité peut également améliorer l'interprétabilité.

Il est essentiel de souligner l'importance d'une évaluation soignée et d'une validation croisée lors de l'application de techniques de sélection d'attributs pour s'assurer que les attributs conservés contiennent les informations les plus pertinentes pour la tâche de prédiction. Les modèles basés sur des ensembles, tels que la Forêt aléatoire et AdaBoost, se sont révélés particulièrement résilients à la réduction de dimensionnalité, offrant un équilibre optimal entre performance et complexité du modèle.

En fin de compte, cette étude souligne que, malgré les défis posés par la sélection d'attributs et la réduction de dimensionnalité, des stratégies judicieuses peuvent conduire à des modèles efficaces et efficients. Ces modèles peuvent offrir des insights précieux et soutenir la prise de décisions stratégiques dans divers domaines d'application, allant de la gestion de restaurants à d'autres contextes industriels et commerciaux. La clé du succès réside dans la compréhension approfondie des caractéristiques de chaque modèle et dans l'ajustement minutieux des paramètres pour adapter chaque algorithme au contexte spécifique de son application.