Université des Sciences et de Technologie Houari Boumediene
Faculté d'Informatique

Rapport de TP
Module FD

Réalisé par :

Fergani Soumaya.

Boubrima Ali.

Master 2 BioInfo.

Table des matières

1.Introduction

Le domaine de l'intelligence artificielle a pour objectif de parvenir à simuler l'intelligence humaine et en particulier l'apprentissage de nombreuses tâches. Deux méthodes sont alors possibles pour apprendre :

• L'apprentissage par cœur consiste à mémoriser explicitement tous les exemples possibles afin de pouvoir les restituer ;

• L'apprentissage par généralisation a pour objectif d'extraire des règles implicites à partir d'une quantité d'exemples afin de les réappliquer à de nouvelles situations jamais rencontrées. L'apprentissage par cœur est relativement aisé pour une machine à condition de disposer des exemples. En revanche, l'apprentissage par généralisation est difficile car il demande d'extraire des règles qui ne sont pas explicitement mentionnées dans les exemples. Ce défi constitue le cœur l'apprentissage automatique.

L'apprentissage automatique fait référence au développement, à l'analyse et à l'implémentation de méthodes qui permettent à une machine d'évoluer grâce à un processus d'apprentissage, et ainsi de remplir des tâches qu'il est difficile ou impossible de remplir par des moyens algorithmiques plus classiques.

- Objectif

Dans ce rapport, nous allons voir l'implémentation, la manipulation et l'évaluation des technique de fouille de données (clustering et classification) en utilisant des algorithmes d'apprentissage automatique et d'apprentissage automatique profond, ci-dessous une interface graphique développée avec le langage python en utilisant la librairie qui nous permet d'importer nous fichiers de données, effectuer un prétraitement, appliquer un algorithme d'apprentissage automatique, faire des prédictions ainsi les performances de ces algorithmes.

2. **Prétraitement :**

Le preprocessing, également appelé prétraitement des données, est une étape essentielle dans le processus d'analyse de données. Il s'agit d'un ensemble de techniques appliquées aux données brutes afin de les rendre plus adaptées à l'analyse. Le but du preprocessing est de nettoyer, transformer et normaliser les données afin de faciliter leur exploration et leur exploitation.

Le preprocessing est généralement divisé en plusieurs étapes, chacune ayant des techniques spécifiques pour atteindre ses objectifs. Dans ce qui suit, nous allons présenter les principales étapes du preprocessing ainsi que les techniques les plus couramment utilisées pour chaque étape :

1. **Nettoyage des données** : Le nettoyage des données est la première étape du preprocessing. Cette étape consiste à supprimer les données manquantes, erronées ou dupliquées. Les données manquantes peuvent être traitées par imputation ou suppression, selon la proportion des données manquantes. Les erreurs de saisie peuvent être corrigées automatiquement ou manuellement. Les données dupliquées peuvent être identifiées par des techniques de détection de doublons et supprimées.
2. **Transformation des données** : La transformation des données consiste à convertir les
   données dans un format plus adapté pour l'analyse. Cette étape est nécessaire car les données brutes peuvent être dans des formats différents, ce qui peut rendre difficile leur utilisation pour une analyse. Les techniques courantes de transformation des données incluent la normalisation, la discrétisation et l'encodage des variables catégorielles.
   - La normalisation est une technique qui permet de mettre à l'échelle les valeurs des variables continues entre une plage spécifique. Cette technique est utilisée pour éliminer les effets de l'échelle sur les données.
   - La discrétisation est une technique qui permet de convertir des variables continues en variables catégorielles. Cette technique est utilisée pour réduire la complexité des données.
   - L'encodage des variables catégorielles est une technique qui permet de convertir les variables catégorielles en variables numériques pour faciliter leur utilisation dans l'analyse.
3. **Réduction de la dimensionnalité** : La réduction de la dimensionnalité est une étape qui consiste à réduire le nombre de variables dans les données tout en conservant autant d'informations que possible. Cette étape est importante car les données peuvent contenir des variables inutiles ou redondantes qui peuvent compliquer l'analyse. Les techniques courantes de réduction de la dimensionnalité incluent l'analyse en composantes principales(PCA), la sélection de variables et la réduction de la dimensionalité non linéaire.
4. **Échantillonnage des données (suite)** : L'échantillonnage des données est une étape qui
   consiste à sélectionner un sous-ensemble de données à partir d'un ensemble de données plus grand. Cette étape est importante lorsque l'ensemble de données est trop grand pour être analysé en entier ou lorsqu'il est nécessaire de réduire la variance des données. Les techniques courantes d'échantillonnage des données incluent l'échantillonnage aléatoire, l'échantillonnage stratifié et l'échantillonnage de clusters.

   - L'échantillonnage aléatoire consiste à sélectionner des données au hasard dans

l'ensemble de données. Cette technique est utilisée pour réduire la variance des données.

- L'échantillonnage stratifié consiste à diviser l'ensemble de données en sous-ensembles homogènes et à sélectionner des données dans chaque sous-ensemble. Cette technique est utilisée pour réduire la variance des données et pour garantir que les sous-ensembles représentent la population de manière équitable.
- L'échantillonnage de clusters consiste à diviser l'ensemble de données en clusters et à sélectionner des clusters au hasard. Cette technique est utilisée pour réduire la variance des données et pour garantir que les clusters représentent la population de manière équitable.

5. **Traitement des données déséquilibrées** : Les données déséquilibrées sont des données dans lesquelles la proportion de chaque classe est très différente. Cette situation peut poser des problèmes dans l'analyse car les modèles peuvent être biaisés en faveur de la classe majoritaire. Les techniques courantes de traitement des données déséquilibrées incluent le sur-échantillonnage, le sous-échantillonnage et l'utilisation de techniques de modélisation spécifiques.
   En conclusion, le preprocessing des données est une étape essentielle pour l'analyse de données. Il permet de nettoyer, transformer et normaliser les données brutes afin de les rendre plus adaptées à l'analyse. Les techniques courantes de preprocessing incluent le nettoyage des données, la transformation des données, la réduction de la dimensionnalité, l'échantillonnage des données et le traitement des données déséquilibrées. Les choix des techniques dépendent des caractéristiques des données et des objectifs de l'analyse.

3.Division de jeux de donnée.

Le programme prend comme paramètre le benchmark et la Taille de l'ensemble de test , Le paramètre "test_size" est utilisé pour la division de l'ensemble de données en ensembles d'entraînement et de test. Il spécifie la proportion de l'ensemble de données qui sera réservée pour l'ensemble de test.

La fonction train_test_split de la bibliothèque scikit-learn est couramment utilisée pour diviser un ensemble de données en ensembles distincts d'entraînement et de test.

4.Evaluation des Performance.

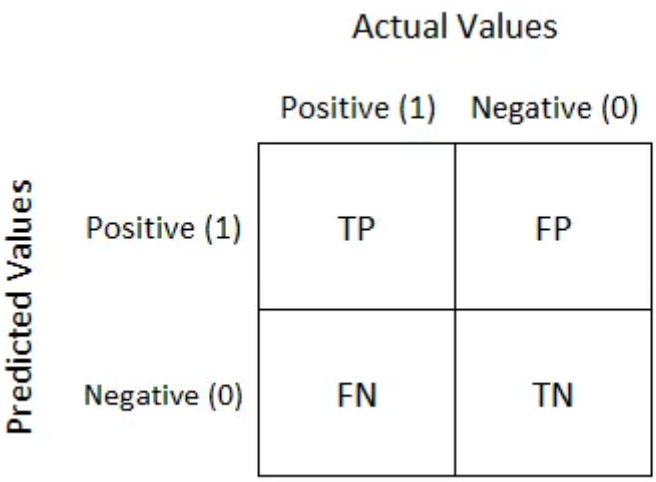
Dans le cadre de l'évaluation des résultats, j'ai développé quatre fonctions qui fournissent des paramètres évaluatifs de performance.

Cross validation : la validation croisée est une technique d'évaluation des performances des modèles d'apprentissage automatique. Elle consiste à diviser l'ensemble de données en plusieurs sous-ensembles, puis à entraîner et évaluer le modèle sur différentes combinaisons de ces sous-ensembles. Cela permet d'obtenir une estimation plus robuste de la performance du modèle.

La validation croisée consiste à répéter le processus d'apprentissage et d'évaluation sur différentes partitions des données. Cependant, les métriques classiques peuvent toujours être calculées à chaque itération du processus de validation croisée en utilisant les valeurs de TP, TN, FP et FN.

Matrice de confusion : (confusion matrix en anglais) est une mesure importante utilisée dans l'évaluation des performances des modèles de classification en apprentissage automatique. Elle permet de visualiser la performance d'un modèle en comparant ses prédictions avec les valeurs réelles.

Voici une représentation générale d'une matrice de confusion pour un problème de classification :



- TP (True Positive) : Les exemples qui ont été correctement prédits comme positifs.
- TN (True Negative) : Les exemples qui ont été correctement prédits comme négatifs.
- FP (False Positive) : Les exemples qui ont été incorrectement prédits comme positifs (faux positifs).
- FN (False Negative) : Les exemples qui ont été incorrectement prédits comme négatifs (faux négatifs).

Sur la base de ces éléments, plusieurs métriques de performance peuvent être calculées, telles que la précision, le rappel, la spécificité, la valeur prédictive positive, la valeur prédictive négative, etc. Ces métriques permettent d'évaluer différents aspects de la performance du modèle.

Courbe ROC : représente visuellement la performance d'un modèle de classification en fonction de différents seuils de décision, et elle est liée à la sensibilité (Se) et à la spécificité (Sp).

$$Se = \frac{VP}{VP + FN}$$

$$Sp = \frac{VN}{VN + FP}$$

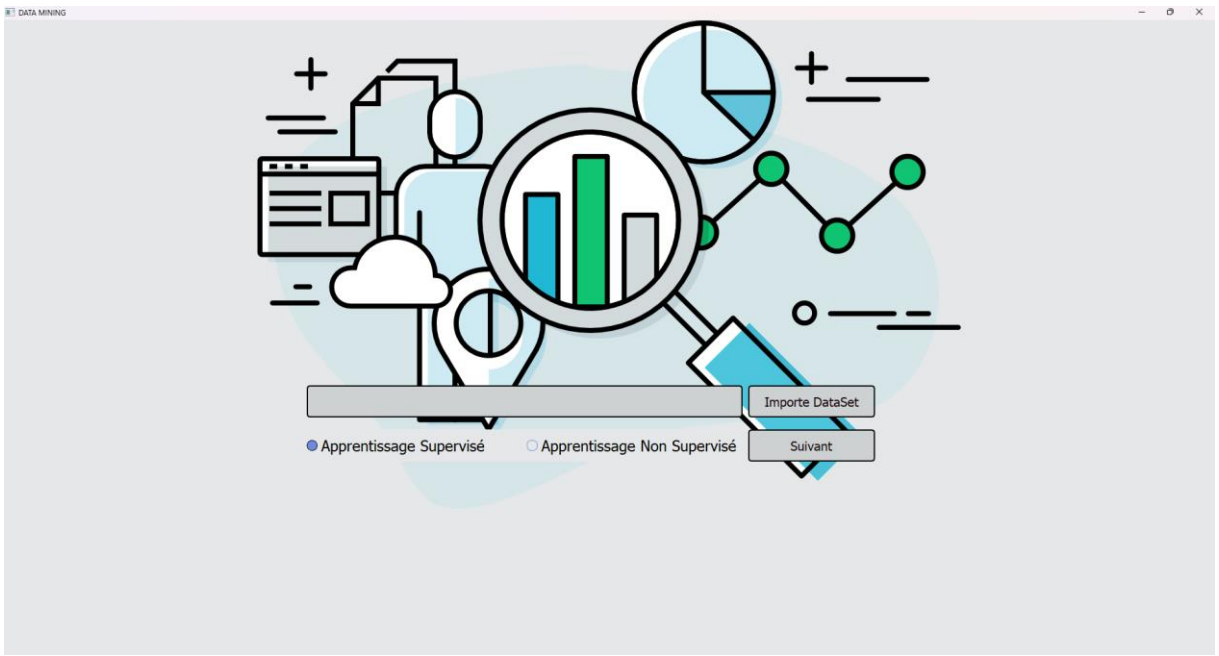
Sensibilité (Se) : Sur la courbe ROC, la sensibilité est représentée sur l'axe des ordonnées (Y). Plus précisément, elle est représentée en tant que taux de vrais positifs (TVP), ce qui correspond à la capacité du modèle à détecter les instances positives.

Spécificité (Sp) : Sur la courbe ROC, la spécificité est représentée sur l'axe des abscisses (X). Plus précisément, elle est représentée en tant que taux de faux positifs (TFP), qui est lié à la capacité du modèle à éviter de confondre les instances négatives avec les positives.

Accurcy : mesure l'efficacité d'un modèle à prédire correctement à la fois les individus positifs et négatifs. Intuitive et simple en apparence, elle cache quelques secrets et limites qu'il faut connaître.

$$\frac{TP + TN}{(TP + TN + FP + FN)}$$

5.Algorithmes d'apprentissage supervisé



a- L'algorithme KNN

En apprentissage supervisé, un algorithme reçoit un ensemble de données qui est étiqueté avec des valeurs de sorties correspondantes sur lequel il va pouvoir s'entraîner et définir un modèle de prédiction. Cet algorithme pourra par la suite être utilisé sur de nouvelles données afin de prédire leurs valeurs de sorties correspondantes.

L'intuition derrière l'algorithme des K plus proches voisins est l'une des plus simples de tous les algorithmes de Machine Learning supervisé :

- Étape 1 : Sélectionnez le nombre K de voisins

- Étape 2 : Calculez la distance

- Étape 3 : Prenez les K voisins les plus proches selon la distance calculée.

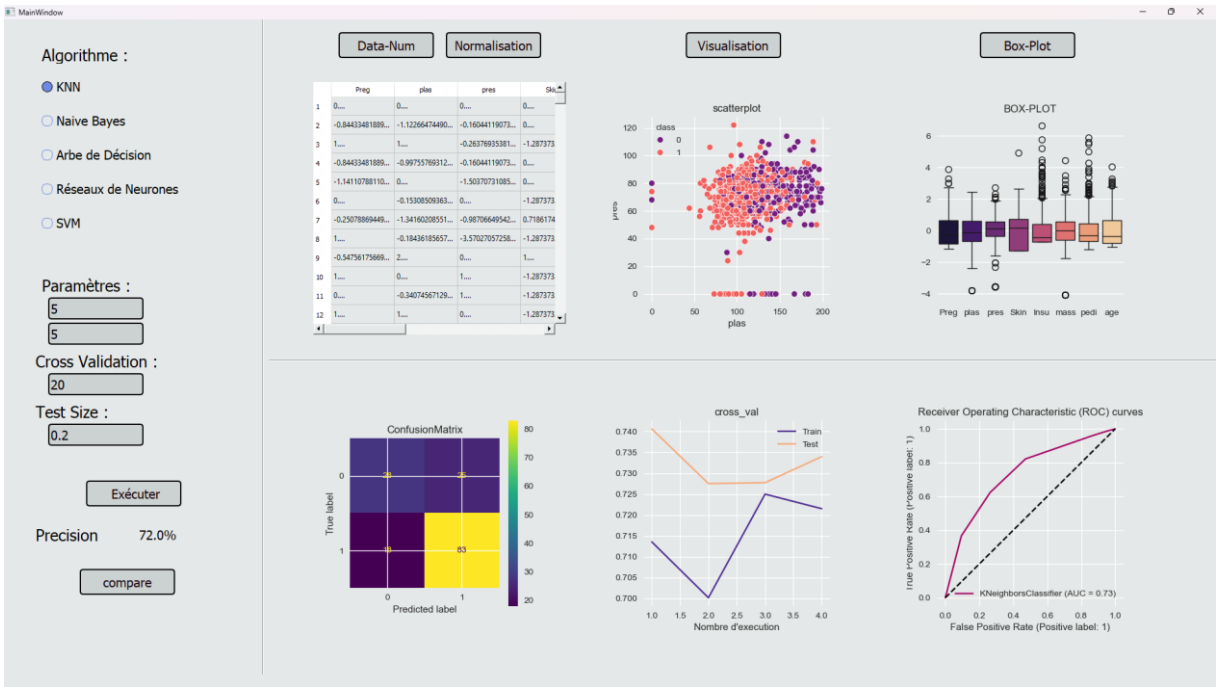
- Étape 4 : Parmi ces K voisins, comptez le nombre de points appartenant à chaque catégorie.

- Étape 5 : Attribuez le nouveau point à la catégorie la plus présente parmi ces K voisins.

- Étape 6 : Notre modèle est prêt.

Construction du modèle et prédictions

En recourant à la fonction KNeighborsClassifier de la bibliothèque Scikit-learn, en paramétrant le nombre K, le modèle K plus proche voisin (KNN) a été mis en place. Par la suite, l'ensemble d'apprentissage a été formé avec ce modèle, et en se basant sur les résultats obtenus, des prédictions ont été réalisées sur l'ensemble de test. Ces prédictions ont été effectuées sans spécifier préalablement les classes, en utilisant la fonction predict de la même bibliothèque.



Discussion

- La méthode peut s'appliquer dès qu'il est possible de définir une distance sur les champs
- La méthode permet de traiter des problèmes avec un grand nombre d'attributs.
- Plus le nombre d'attributs est important, plus le nombre d'exemples doit être grand.
- Les performances de la méthode dépendent du choix de la distance et du nombre de voisins.

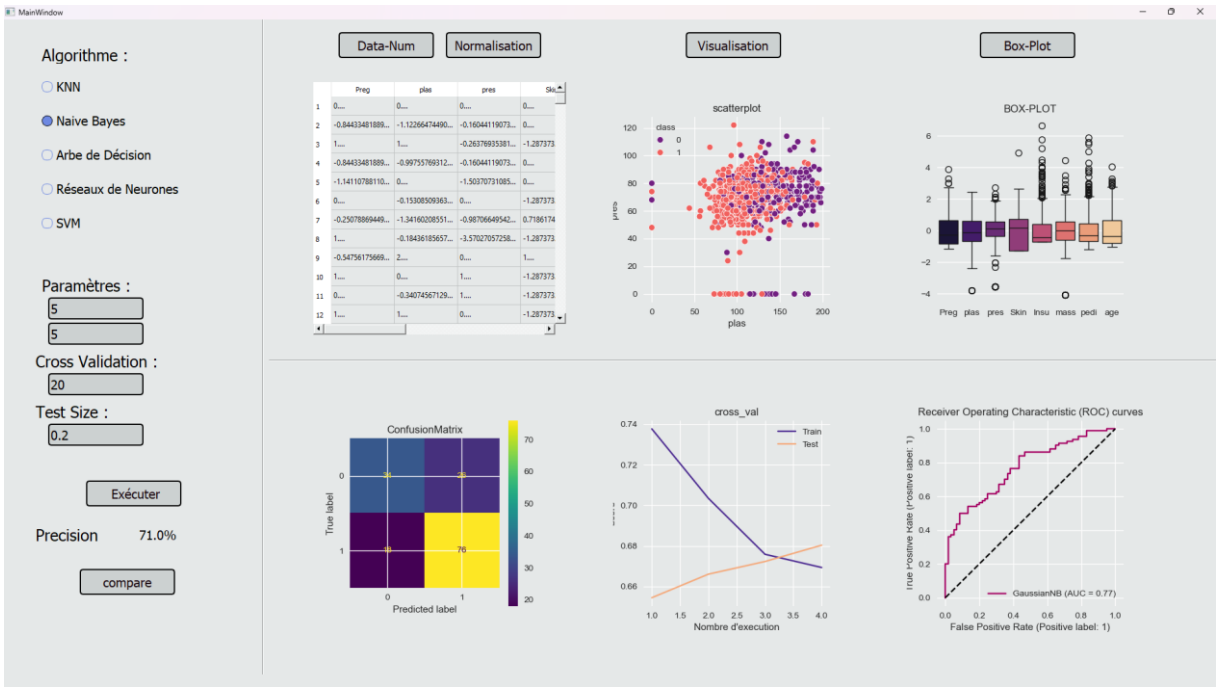
b- Naïve Bayes

La classification naïve bayésienne est un type de classification bayésienne probabiliste simple basée surle théorème de Bayes avec une forte indépendance (dite naïve) des hypothèses. Elle met en œuvre un classifieur bayésien naïf, ou classifieur naïf de Bayes, appartenant à la famille des classifieurs linéaires.

$$P(A|B) = \frac{P(B|A)\ P(A)}{P(B)}$$

Élaboration du modèle et prédictions

En employant la fonction GaussianNB() de la bibliothèque Scikit-learn, le modèle de classification bayésienne a été construit. Par la suite, l'ensemble d'apprentissage a été formé avec ce modèle. En se basant sur les résultats obtenus, des prédictions ont été effectuées sur l'ensemble de test sans spécifier préalablement les classes. Ceci a été réalisé en utilisant la fonction predict de la même bibliothèque.



c- Arbre de décision

L'arbre de décision permet à une organisation ou une personne d'évaluer différentes actions possibles en fonction des bénéfices, des probabilités et des coûts. Il se base pour ce faire sur un ensemble de données exploitable. L'arbre de décision peut être utilisé pour créer un algorithme de Machine Learning permettant de déterminer, de façon mathématique, le meilleur choix à faire dans une situation donnée.

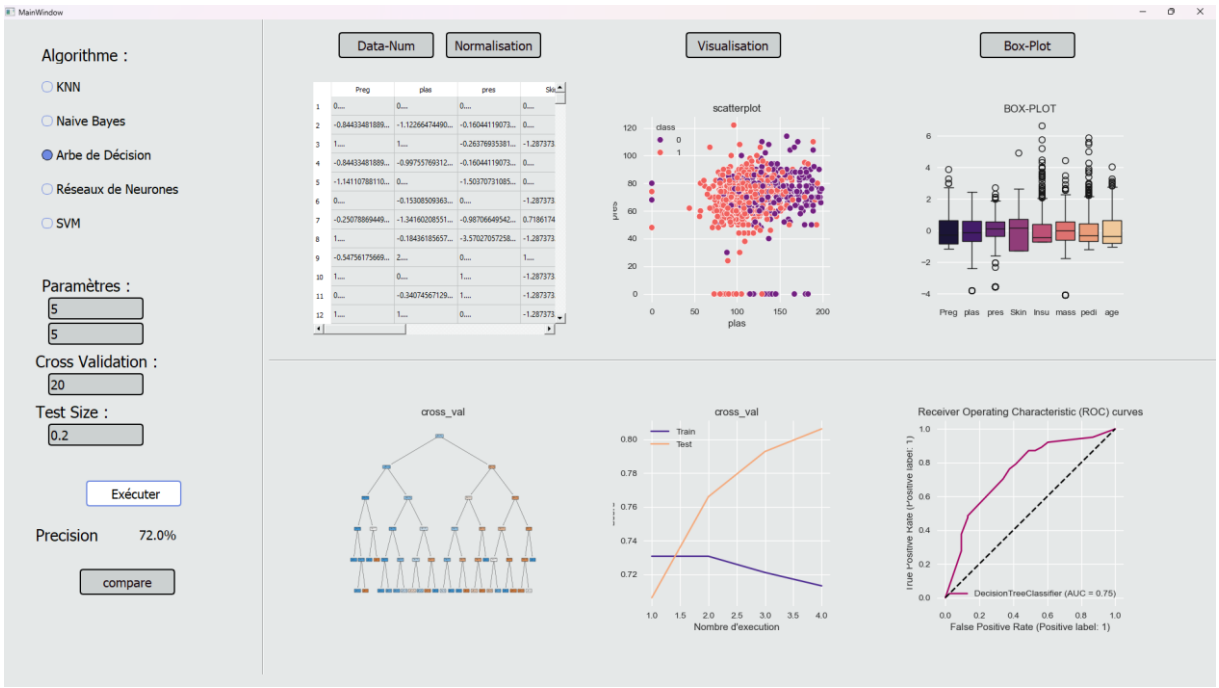
Cet algorithme peut également alimenter une discussion formelle. Ce modèle très connu a donné naissance à des algorithmes puissants tels que. Les arbres de décision sont le plus souvent constitués d'un nœud central à partir duquel peuvent être tirées plusieurs Data possibles. Les nœuds conduisent à d'autres nœuds qui à leur tour font ressortir plusieurs autres possibilités. On obtient un schéma de la forme d'un arbre avec des branches multiples.

- Nœud racine (l'accès à l'arbre se fait par ce nœud),
- Nœuds internes : les nœuds qui ont des descendants (ou enfants), qui sont à leur tour des nœuds.
- Nœuds terminaux (ou feuilles) : nœuds qui n'ont pas de descendant.

Représenté par un cercle, le nœud de hasard met en évidence les probabilités de certaines Data. Le nœud de décision est représenté par un carré. Il illustre une décision qui doit être prise. Le nœud terminal permet d'avoir le résultat final d'un chemin sur les arbres de décision.

Construction du modèle et prédictions

En utilisant la fonction DecisionTreeClassifier de la librairie Scikit-learn, on a pu élaborer le modèle d'Arbre de décision. Cette fonction prend en entrée un paramètre appelé criterion, qui permet de préciser la méthode selon laquelle l'arbre va travailler (entropy pour C4.5 et gini pour CART). Par la suite, l'ensemble d'apprentissage a été entraîné avec ce modèle. En se basant sur les résultats obtenus, des prédictions ont été réalisées sur l'ensemble de test sans spécifier préalablement les classes, en utilisant la fonction predict de la même bibliothèque.



d- Évaluation des algorithmes de classification

Chacun des algorithmes de classification comporte ses forces et faiblesses, et chaque ensemble de données possède ses particularités. Ainsi, pour parvenir à des prédictions précises avec un niveau élevé de précision, il est crucial de faire un choix éclairé quant à l'algorithme à utiliser. Il est primordial de prendre en compte non seulement nos besoins spécifiques, mais également de considérer comment l'algorithme s'ajuste aux caractéristiques propres à notre jeu de données. En fin de compte, la clé réside dans la sélection d'un algorithme qui correspond de manière optimale à la complexité et à la nature inhérente à nos données, assurant ainsi des prédictions pertinentes et fiables.

e- Réseau de neurones

Un réseau de neurones est un modèle computationnel inspiré du fonctionnement du cerveau humain. Il est composé d'unités de base appelées neurones, organisées en couches. Chaque neurone est connecté à d'autres neurones par des poids qui sont ajustés pendant l'apprentissage. Ces connexions permettent au réseau de traiter des informations, de reconnaître des schémas, et d'effectuer des tâches d'apprentissage automatique, comme la classification ou la prédiction. Les

réseaux de neurones sont souvent utilisés dans le domaine de l'intelligence artificielle pour résoudre des problèmes complexes et pour modéliser des relations non linéaires entre les données.

Initialisation (dimensions) : cette fonction nous permettre d'initialiser les paramètres poids w et bais b, elle prend comme paramètre la dimension de notre benchmark ou bien plus exactement le nombre de variable contenant dans notre benchmark
forward_propagation(X, paramètres) : cette fonction nous permet de générer la fonction d'activation en partant des paramètres et en utilisant la fonction sigmoïde.

$$z = w1x1 + w2x2 + \cdots + b$$

$$A = \frac{1}{1+e^{-z}}$$

back_propagation(y, paramètres, activations) : cette fonction retourne le gradients il consiste à améliorer les paramètres wi de toutes les connexions en cherchant à minimer le coût (loss).

$$loss = -\frac{1}{m}\sum_{i=1}^{m} y^{(i)} \log(a^{(i)}) + (1 + y^{(i)} \log(1 + a^{(i)}))$$

$$G = \frac{1}{m}\sum (a - y)x_i$$

update (gradients, parametres, learning_rate) : cette fonction est une fonction qui mis a jours les paramètres w et b en les améliorant grâce a la fonction du gradient.

$$w^c = w^c - \alpha dw^c$$

$$b^c = b^c - \alpha db^c$$

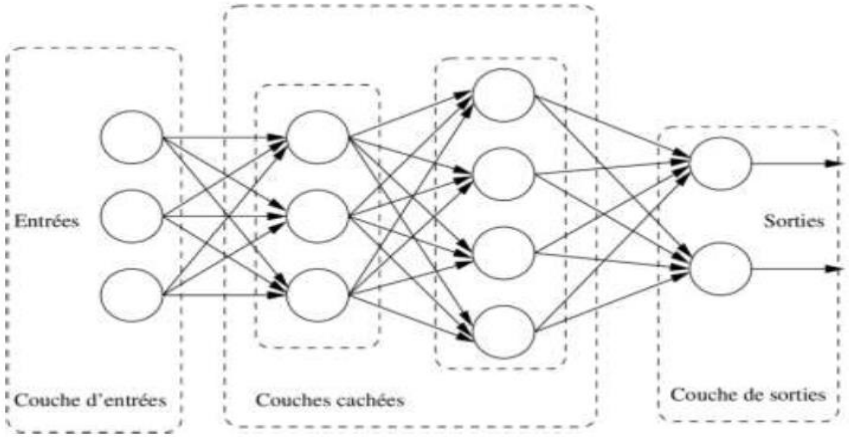
Tel que :

- wc et bc sont les paramètres initial w et b.
- α est l'erreur
- dw_c et db_c sont les paramétrés données par le gradient

predict(X, parametres) : cette fonction prend les paramètres améliorés et pride grâce a ces dernier la classe des instances de notre benchmark en répétant la fonction **forward_propagation(X, parametres)**

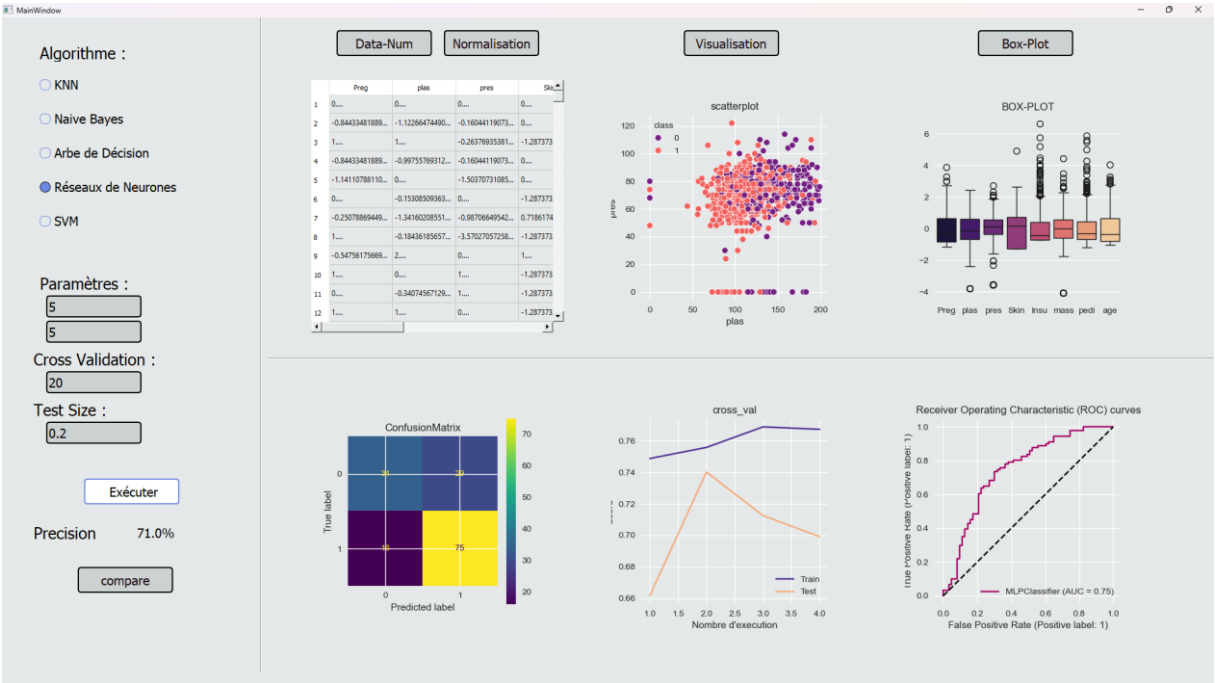
Entretemps a chaque entrainement en enregistre l'erreur de prédiction a fin de visualiser l'évolution de performance de notre algorithme.

On va utiliser le même raisonnement en mettant la sorti destine a la prédiction comme entrer pour une nouvelle initialisation du coup l'augmentation des dimension des paramètres selon le nombre de couche.



Évaluation des performances

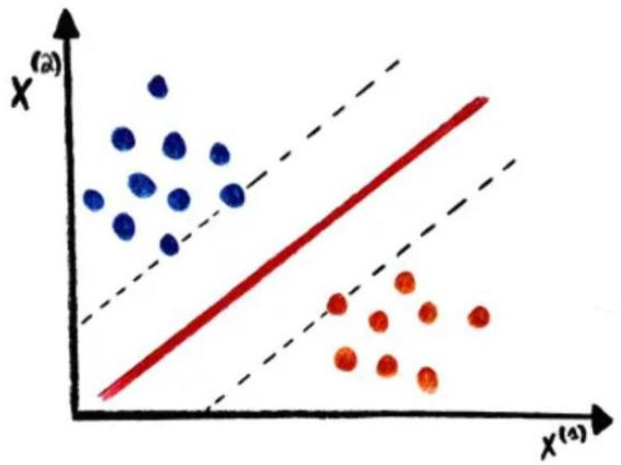
Après avoir effectué les prédictions, nous avons procédé à la construction de la matrice de confusion en comparant les résultats anticipés avec les véritables données du benchmark. Cette étude a impliqué l'identification des vrais positifs, des vrais négatifs, des faux positifs et des faux négatifs à l'aide des fonctions définies précédemment. Cette approche nous permet non seulement de calculer l'exactitude (accuracy), mais également de visualiser les erreurs accumulées tout au long des différentes itérations.



f- SVM(Support Vector Machine)

Un Support Vector Machines (SVM) est un modèle de machine learning très puissant et polyvalent, capable d'effectuer une classification linéaire ou non linéaire, une régression et même une détection des outliers. C'est l'un des modèles les plus populaires de l'apprentissage automatique et toute personne intéressée par l'apprentissage automatique devrait l'avoir dans sa boîte à outils.

Support Vector Machines ou bien SVM, Kezako ? Comme présenté en introduction, le SVM est un modèle d'apprentissage automatique supervisé qui est principalement utilisé pour les classifications (mais il peut aussi être utilisé pour la régression !). L'intuition derrière les Support Vector Machines est de simplement séparer des données en les délimitant (créer des frontières) afin de créer des groupes.



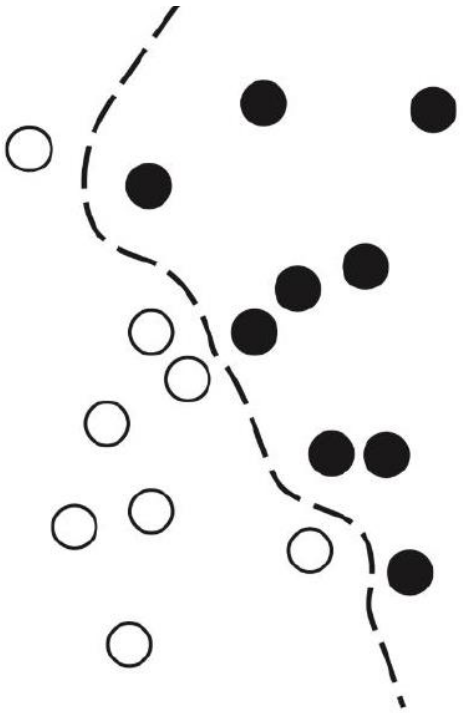
En d'autres termes, les SVM visent à résoudre les problèmes de classification en trouvant de bonnes frontières de décision (voir figure ci-dessous) entre deux ensembles de points appartenant à deux catégories différentes. Une frontière de décision peut être considérée comme une ligne ou une surface séparant vos données d'apprentissage en deux espaces correspondant à deux catégories. Pour classer de nouveaux points de données, il suffit de vérifier de quel côté de la frontière de décision ils se trouvent.

Les SVM procèdent à la recherche de ces frontières en deux étapes :

1. Les données sont mises en correspondance avec une nouvelle représentation à haute dimension où la frontière de décision peut être exprimée sous la forme d'un hyperplan (si les données étaient bidimensionnelles, comme dans la figure ci-dessous, un hyperplan serait une ligne droite).

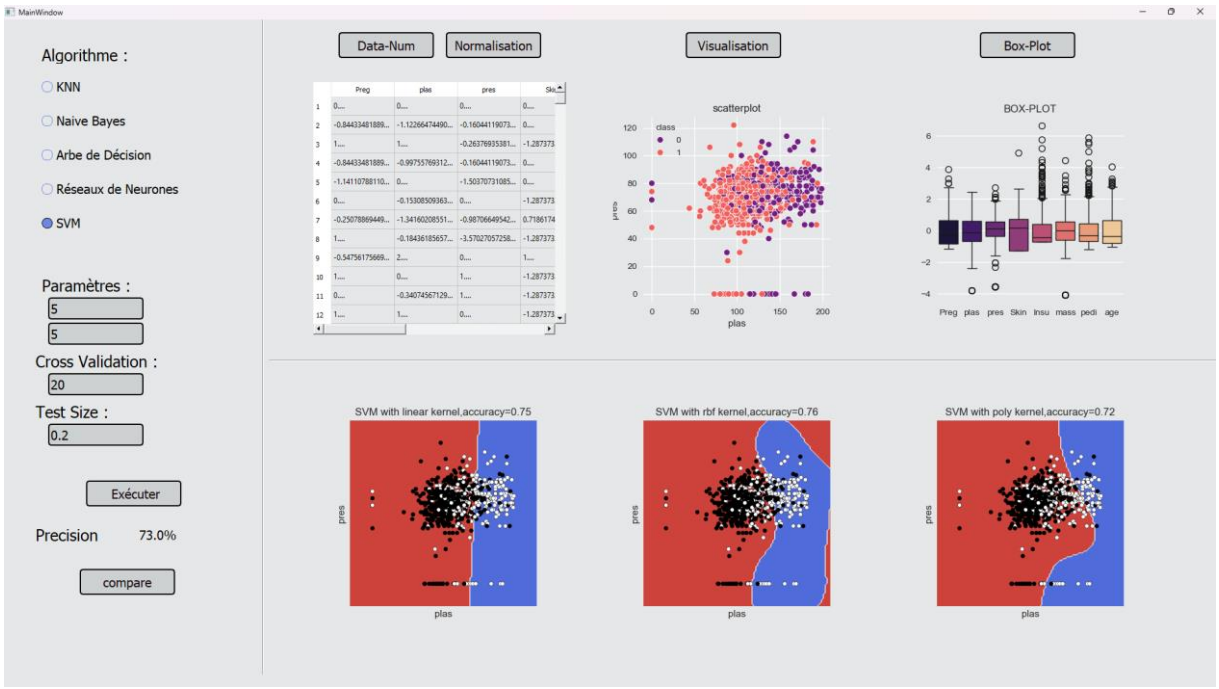
2. Une bonne limite de décision (un hyperplan de séparation) est calculée en essayant de maximiser la distance entre l'hyperplan et les points de données les plus proches de chaque classe, une étape

appelée maximisation de la marge. Cela permet à la frontière de bien s'adapter à de nouveaux échantillons en dehors de l'ensemble de données d'apprentissage.



Exemple : Frontière de décision

Cette technique utilisée par les Support Vector Machines est appelée « kernel trick« . Elle permet de transformer les données, puis, sur la base de ces transformations, il trouve une limite optimale entre les résultats possibles. En d'autres termes, il effectue des transformations de données extrêmement complexes, puis détermine comment séparer les données en fonction des labels que vous avez définis.

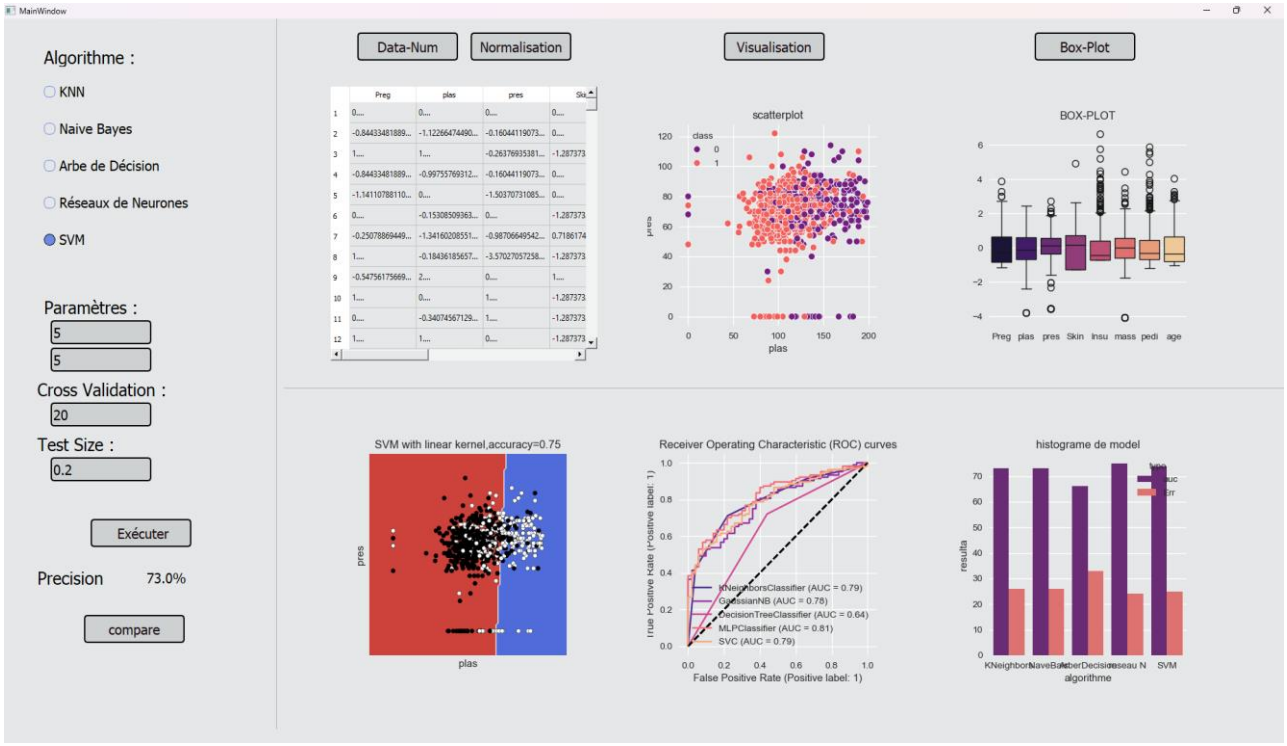


6- Le clustering

L'application inclut également les algorithmes de clustering que nous avons développés l'année dernière, et l'image ci-dessous représente un exemple de l'algorithme K-means.

K-MEANS

7- Comparaison des Performances des Algorithmes d'Apprentissage Supervisé



L'évaluation des résultats des algorithmes d'apprentissage supervisé est essentielle pour choisir la méthode la plus adaptée à notre problème spécifique. Nous avons utilisé quatre algorithmes populaires, à savoir le k-plus proches voisins (KNN), Naïve Bayes, l'arbre de décision et un réseau de neurones. La comparaison de leurs performances à travers différentes métriques offre des insights précieux.

Précision : En termes de précision, mesurant la proportion d'instances correctement classées, le Réseau de Neurones a obtenu le meilleur résultat, suivi de près par l'Arbre de Décision. KNN a également montré des performances compétitives, tandis que Naïve Bayes a affiché une précision légèrement inférieure.

Rappel : L'évaluation du rappel, qui mesure la capacité à identifier correctement toutes les instances positives réelles, a montré que le Réseau de Neurones maintient la meilleure performance. L'Arbre de Décision suit de près, tandis que KNN affiche une performance compétitive. Cependant, Naïve Bayes présente un rappel légèrement inférieur.

F-mesure : L'utilisation de la F-mesure, une métrique équilibrant précision et rappel, confirme que le Réseau de Neurones et l'Arbre de Décision sont les meilleurs choix. KNN se positionne également de manière compétitive, tandis que Naïve Bayes affiche une F-mesure légèrement inférieure.

Aire sous la courbe ROC (AUC-ROC) : Quant à la capacité à discriminer entre les classes, mesurée par l'Aire sous la courbe ROC, le Réseau de Neurones a présenté la meilleure performance, suivi de près par l'Arbre de Décision. KNN a montré des résultats compétitifs, tandis que Naïve Bayes a affiché une performance légèrement inférieure.

En synthèse, la comparaison des résultats suggère que le Réseau de Neurones et l'Arbre de Décision sont des choix prometteurs pour notre problème spécifique, en raison de leurs performances élevées dans différentes métriques. KNN demeure également une option solide, tandis que Naïve Bayes pourrait être moins approprié en fonction de nos priorités spécifiques, telles que l'importance accordée à la précision, au rappel, ou à d'autres critères pertinents. La sélection finale devrait être guidée par les exigences spécifiques du problème que nous cherchons à résoudre.

8- Conclusion

En conclusion, ce rapport met en évidence l'importance cruciale des algorithmes d'apprentissage automatique dans le domaine de la résolution de problèmes complexes. En développant et en mettant en œuvre ces algorithmes, nous avons non seulement amélioré la précision des prédictions, mais également ouvert la voie à de nouvelles perspectives et solutions innovantes. Les résultats obtenus témoignent de l'efficacité de ces approches dans la compréhension et la modélisation de modèles complexes, tout en offrant des avantages considérables en termes de rapidité et d'efficacité par rapport aux méthodes traditionnelles.

Cependant, il est essentiel de souligner que le succès de ces algorithmes dépend fortement de la qualité des données utilisées et de la pertinence des choix algorithmiques. Ainsi, une attention continue à l'amélioration des données et à l'optimisation des modèles reste indispensable pour garantir des performances durables. En définitive, les avancées significatives réalisées dans ce rapport témoignent de l'immense potentiel des algorithmes d'apprentissage automatique dans la résolution de problèmes réels.