



Université Mohammed
Premier Faculté des Sciences
d'Oujda Département
d'Informatique



Mastère Spécialisé En Ingénierie
Informatique

*Implémentation de l'arbre de décision pour les
Chaines de caractères*



Présenté par :

OUFKIR Ibrahim
BOUAROUR Ayoub

Encadré par :

Pr. MAZROUI Azzedine

Année universitaire : 2023-2024

TABLE DES MATIERES

Table des matières	i
Liste des figures	iii
Remerciements	iv
introduction générale.....	1
Algorithme de l'arbre de décision	4
1. Introduction.....	4
2. Définition des Arbres de Décision.....	4
3. Intérêt des Arbres de Décision	5
3.1. Polyvalence.....	5
3.2. Interprétation aisée	6
3.3. Traitement de divers types de données.....	6
3.4. Facilité de modification et de mise à jour.....	6
3.5. Anticipation des conséquences des décisions.....	6
4. Problématique et Limitations	6
4.1. Surajustement (<i>overfitting</i>).....	7
4.2. Sensibilité aux données bruitées	7
4.3. Complexité des modèles	7
5. Algorithme générale	7
6. Construire un Arbre de décision.....	9
6.1. Entropie	9
6.2. Gain d'informations	12
6.3. Indice de Gini	12
7. Exemple pratique.....	12
7.1. Calcul de l'entropie pour l'ensemble de données complet T.....	13
7.2. Calcul de l'entropie conditionnelle pour chaque attribut.....	14
7.3. Calcul du gain d'information pour chaque attribut	15
7.4. Choisir l'attribut avec le gain d'information le plus élevé	16
7.5. Répéter les étapes de manière récursive.....	17
7.6. Arbre de décision	18

8.	Conclusion	19
	Implémentation et résultats expérimentaux	21
1.	Introduction.....	21
2.	Implémentation de l'Arbre de Décision et de la Forêt Aléatoire 21	
	2.1. Arbre de décision	22
	2.2. Algorithme ID3.....	22
	2.3. CART	23
	2.4. Forêt Aléatoire (<i>Random Forest</i>).....	24
	2.5. Évaluation des Performances	25
3.	Démonstration des interfaces de l'application	26
	3.1. Page d'accueil.....	26
	3.2. Importation des Données	27
	3.3. Entraînement des Modèles.....	27
	3.4. Évaluation.....	30
	3.5. Visualisation d'Arbre de Décision.....	31
4.	Comparaison des Résultats des Algorithmes d'Arbres de Décision.....	32
	4.1. Jeux de caractères.....	33
	4.2. Méthodologie.....	33
	4.3. Résultats.....	33
	4.4. Analyse des Résultats.....	34
5.	Conclusion	36
	conclusion général.....	37
	Webographie	38

LISTE DES FIGURES

Figure 1 : Exemple d'un arbre de décision	5
Figure 2 : Entropie - Cas de deux classes	11
Figure 3 : jeux de données	13
Figure 4 Calcul de l'Entropie et du Gain d'Information pour chaque Attribut	16
Figure 5 : Division basée sur l'Attribut "Temps" - Cas Ensoleillé.....	17
Figure 6 : Division basée sur l'Attribut "Temps" - Cas Nuageux	18
Figure 7 : Arbre de décision obtenue	19
Figure 8 : Interface accueil.....	26
Figure 9 : Interface d'importation des données.....	27
Figure 10 : Interface d'Évaluation.....	31
Figure 11 : Interface de Visualisation d'Arbre de Décision.....	32
Figure 12 : Tableau de comparaison de « contact-lenses.arff ».....	34
Figure 13 : Tableau de comparaison de « breast-cancer.arff ».....	34

REMERCIEMENTS

Au début de ce rapport, nous souhaitons exprimer notre profonde gratitude à Allah, notre Seigneur et notre guide, pour Sa miséricorde et Sa guidance tout au long de notre parcours académique.

Nous tenons également à adresser nos plus sincères remerciements à nos parents, dont le soutien inconditionnel et les sacrifices ont été une source d'inspiration et de motivation tout au long de notre formation.

Un immense merci à notre professeur Mr. MAZZROUI Azzedine pour son excellent travail, son expertise et son dévouement exceptionnel. Ses précieux enseignements et son encadrement attentif ont grandement contribué à notre développement académique et professionnel.

Enfin, nous tenons à remercier tous nos enseignants du master d'ingénierie informatique pour leur contribution précieuse à notre formation et pour leur dévouement à transmettre leur savoir et leur expérience.

INTRODUCTION GENERALE

Dans le cadre de notre formation de Master en Ingénierie Informatique, nous avons eu l'occasion d'exécuter un projet majeur axé sur l'intelligence artificielle. Ce projet nous a permis de mettre en pratique nos compétences théoriques et techniques, tout en nourrissant notre esprit de recherche et de créativité. Concrètement, notre mission était d'implémenter un algorithme d'arbre de décision en Java, en partant de zéro, et de développer une interface en Java pour permettre des démonstrations interactives. Travaillant en étroite collaboration en équipe et sous la supervision attentive de notre professeur encadrant, nous avons pu mener ce projet à bien.

Ce rapport commence par une définition et une explication approfondie de l'algorithme de l'arbre de décision, suivi par une présentation de la problématique que nous avons explorée. Nous approfondirons ensuite les concepts mathématiques essentiels pour comprendre pleinement cet algorithme. Une section sera dédiée à la description détaillée de notre implémentation en Java, où nous expliquerons les différentes classes, leurs attributs et méthodes. Nous inclurons également une démonstration pratique de notre application via l'interface utilisateur. Pour évaluer la performance de notre modèle, nous fournirons des statistiques sur sa précision et discuterons de ses avantages et inconvénients.

En conclusion, nous synthétiserons les travaux réalisés et présenterons notre bibliographie, offrant une vision claire et structurée de notre projet, tout en soulignant les défis rencontrés et les solutions apportées. Ce rapport vise à fournir une compréhension exhaustive de notre démarche et de nos résultats,

en mettant en lumière les aspects théoriques et pratiques de notre travail sur l'intelligence artificielle.

Chapitre 1

ALGORITHME DE L'ARBRE DE DECISION

1. Introduction

La classification par les arbres de décision est une méthode d'apprentissage supervisé largement utilisée en intelligence artificielle et en statistique. Elle repose sur la décomposition d'un problème complexe en une série de décisions plus simples, organisées sous la forme d'un arbre. Chaque nœud de cet arbre représente un test sur un attribut, chaque branche correspond à un résultat possible de ce test, et chaque feuille représente une classe de décision ou une valeur prédite.

Cette technique est appréciée pour sa simplicité et sa facilité d'interprétation, rendant les modèles accessibles même pour ceux qui ne sont pas experts en data science. Les arbres de décision sont capables de gérer à la fois des données qualitatives et quantitatives et peuvent modéliser des relations non linéaires entre les variables.

2. Définition des Arbres de Décision

Un **arbre de décision** (decision tree) est un moyen de classer visuellement et logiquement les informations et de prendre des décisions. Cette méthode permet grâce à un simple système de calcul de trouver quel est le choix le plus viable.¹

Les arbres de décision ont une structure hiérarchique et sont composés de **nœuds** et de **feuilles** (aussi appelées nœuds terminaux) reliés par des branches. Dans leur représentation graphique, la racine est placée en haut et les feuilles en bas. Les nœuds internes sont appelés des **nœuds de décision**. Ils peuvent contenir une ou plusieurs règles (aussi appelées tests ou conditions).

¹ “Arbre de Décision : Le Guide Complet (+3 Exemples Utiles et Concrets).”

Les valeurs que peut prendre une variable dans un arbre de décision sont appelées **instances** ou **attributs**. Les nœuds terminaux contiennent **la classe** aussi appelée **classe à prédire** ou **variable cible**. Après sa construction, un arbre de d´décision peut être traduit sous la forme d'un ensemble de règles de décision.

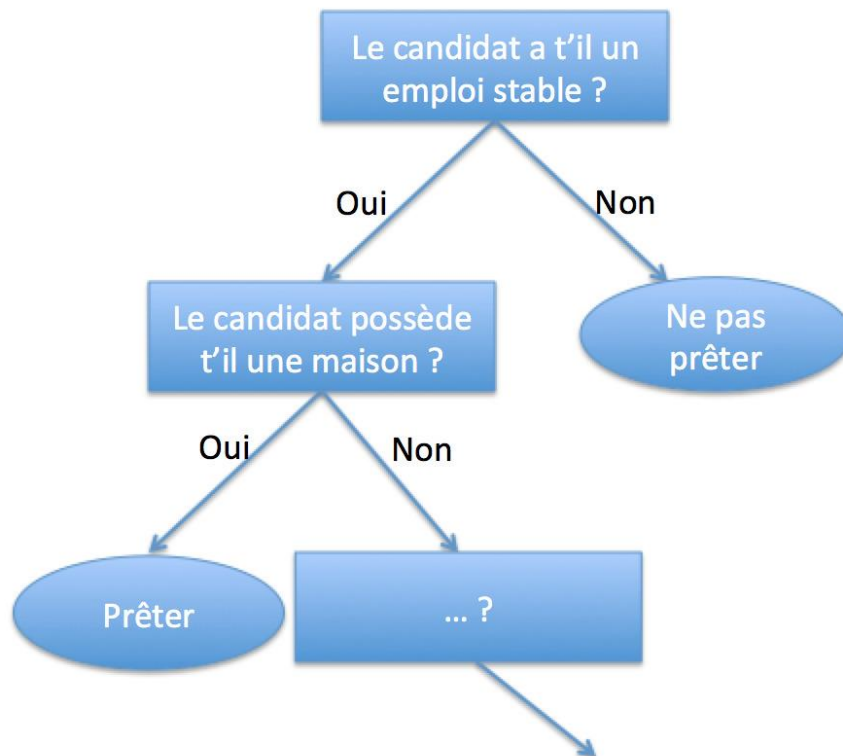


Figure 1 : Exemple d'un arbre de décision

3. Intérêt des Arbres de Décision

3.1. Polyvalence

Les arbres de décision offrent une polyvalence remarquable, pouvant être utilisés à diverses échelles, que ce soit pour des décisions simples du quotidien ou pour des analyses complexes de grands ensembles de données. Leur flexibilité les rend accessibles à tous les niveaux, des individus aux grandes entreprises.

3.2. Interprétation aisée

L'un des atouts majeurs des arbres de décision réside dans leur facilité d'interprétation. Malgré la complexité des décisions qu'ils représentent, leur format graphique simple les rend facilement compréhensibles pour tous les membres d'une équipe, favorisant ainsi la collaboration et la prise de décision informée.

3.3. Traitement de divers types de données

Les arbres de décision sont capables de manipuler efficacement une variété de données, qu'elles soient numériques ou qualitatives. Cette capacité en fait des outils précieux dans de nombreux domaines, de l'apprentissage automatique à la gestion des affaires, où la diversité des données est la norme.

3.4. Facilité de modification et de mise à jour

La structure dynamique des arbres de décision permet des modifications et des mises à jour aisées, permettant ainsi de rester adapté aux évolutions et aux changements. Cette caractéristique est essentielle pour les équipes qui doivent régulièrement ajuster leurs stratégies en fonction des nouvelles informations ou des évolutions du contexte.

3.5. Anticipation des conséquences des décisions

Les arbres de décision offrent la possibilité d'explorer en détail les différentes conséquences et résultats potentiels de diverses décisions. En envisageant tous les scénarios possibles, les utilisateurs peuvent évaluer de manière approfondie les options et choisir la meilleure voie à suivre, contribuant ainsi à des décisions plus éclairées et réfléchies.

4. Problématique et Limitations

Les arbres de décision, bien qu'ils présentent de nombreux avantages, ne sont pas exempts de certaines problématiques et limitations importantes qui peuvent affecter leur performance et leur fiabilité dans certains contextes. Parmi ces problématiques, nous pouvons citer :

4.1. Surajustement (*overfitting*)

L'une des principales préoccupations lors de l'utilisation d'arbres de décision est le risque de surajustement aux données d'entraînement. Le surajustement se produit lorsque le modèle **s'adapte trop précisément** aux données d'entraînement, capturant ainsi le **bruit** et les **variations aléatoires** au lieu de modéliser la véritable relation sous-jacente entre les variables. Cela peut entraîner une **performance médiocre** du modèle lorsqu'il est confronté à de **nouvelles données**, car il **ne généralise pas** bien au-delà de l'ensemble d'entraînement initial.

4.2. Sensibilité aux données bruitées

Les arbres de décision sont sensibles aux **données bruitées** ou **aberrantes** présentes dans l'ensemble d'entraînement. Ces valeurs aberrantes peuvent entraîner des **décisions erronées** ou des divisions inappropriées dans l'arbre, ce qui nuit à la performance globale du modèle. Par conséquent, il est crucial de **nettoyer** et de **prétraiter** soigneusement les données avant de construire un arbre de décision pour éviter que ces valeurs aberrantes n'affectent négativement les résultats.

4.3. Complexité des modèles

La construction d'arbres de décision complexes avec une **profondeur** importante ou un grand **nombre de branches** peut entraîner des modèles difficiles à interpréter et à expliquer. Une **complexité excessive** peut rendre l'arbre de décision **trop spécifique** aux données d'entraînement, augmentant ainsi le risque de **surajustement**. De plus, des arbres trop **complexes** peuvent devenir **peu pratiques** à utiliser dans des applications réelles, nécessitant des temps de **calcul** plus longs et des **ressources** informatiques plus importantes.

5. Algorithme générale

La création d'un arbre de décision est une technique essentielle en apprentissage **supervisé**, permettant de résoudre des problèmes de classification et de régression en **décomposant** les **décisions complexes** en étapes **simples**. Le

processus repose sur la **sélection** optimale des **caractéristiques**, la **division** des données en **sous-ensembles** pertinents et la **construction récursive** de l'arbre jusqu'à ce que toutes les données soient correctement **classifiées**.

Initialiser l'arbre courant à l'arbre vide ; la racine est le nœud courant

Répéter

Décider si le nœud courant est terminal (une feuille).

Si le nœud est terminal **Alors**

Affecter une classe au nœud courant

Sinon

Sélectionner un test et créer autant de nouveaux fils qu'il y a de possibilités de réponses

Fin Si

Passer au nœud suivant

Jusqu'à l'obtention de l'arbre de décision

Il existe plusieurs algorithmes automatiques pour construire des arbres de décision, chacun ayant ses propres caractéristiques et domaines d'application. Voici quelques-uns des plus couramment utilisés :

ID3 (Itérative Dichotomiser 3) : Développé en **1986** par **Ross Quinlan**, ID3 est un algorithme pionnier dans la création des arbres de décision. Il se base sur le concept de **gain d'information** pour sélectionner la caractéristique qui divise le mieux les données à chaque étape. Cependant, ID3 est **limité** à l'utilisation de caractéristiques **nominales**, ce qui **restreint** son application dans des situations où les caractéristiques **continues** sont présentes. L'algorithme est principalement utilisé pour des tâches de **classification**.

C4.5 : Également développé par **Ross Quinlan**, C4.5 est une **extension améliorée** de l'**ID3**. Cet algorithme surmonte les **limitations** de l'**ID3** en permettant l'utilisation de caractéristiques à la fois **nominales** et **continues**. De plus, C4.5 introduit des mécanismes pour gérer les **données manquantes** et **prunier** les arbres (réduction de la taille de l'arbre) afin d'éviter le **surajustement** (*overfitting*). Il est largement utilisé pour les tâches de **classification** et est considéré comme l'un des algorithmes les plus robustes et efficaces pour la création d'arbres de décision.

C5.0 : Cette version **commerciale** de **C4.5**, développée également par **Ross Quinlan**, offre des améliorations significatives en termes de **performance** et d'efficacité. C5.0 utilise moins de **mémoire** et exécute plus **rapidement** que C4.5, tout en produisant des modèles plus compacts et plus précis. En plus de ces améliorations techniques, C5.0 offre des options supplémentaires pour le traitement des données, comme la manipulation de grandes bases de données et l'implémentation de techniques de **boosting** pour améliorer la précision du modèle.

CART (*Classification and Regression Trees*) : Développé par **Leo Breiman, Jerome Friedman, Charles J. Stone** et **Richard A. Olshen**, CART est un algorithme qui, contrairement à C4.5, utilise différentes métriques pour la construction des arbres, telles que **l'indice de Gini** pour la **classification** et la **variance** pour la **régression**. Ce qui distingue CART est sa capacité à traiter à la fois les problèmes de **classification** et de **régression**, ce qui le rend **très polyvalent**. De plus, CART produit des **arbres binaires** (chaque nœud interne a exactement deux enfants), ce qui simplifie souvent l'interprétation du modèle.

6. Construire un Arbre de décision

Puisque notre projet est spécifiquement axé sur les chaînes de caractères, nous avons choisi l'algorithme ID3 pour notre implémentation. Cet algorithme est particulièrement adapté à la manipulation de données catégorielles, ce qui en fait un choix approprié pour notre cas d'utilisation.

6.1. Entropie

Soit une distribution de probabilité $P = (p_1, p_2, \dots, p_n)$ L'entropie de la distribution P est la quantité d'information qu'elle peut apporter. Elle est donnée par l'équation :

$$E(P) = - \sum_{i=1}^n p_i \log p_i \quad (1)$$

Soit un ensemble de données T caractérisé par n classes (C_1, C_2, \dots, C_n) selon la variable cible. La quantité d'informations nécessaire pour identifier la classe d'un individu de l'ensemble T correspond à l'entropie $E(P)$, où P est la distribution de probabilité de la partition (C_1, C_2, \dots, C_n) :

$$P = \left(\frac{|C_1|}{|T|}, \frac{|C_2|}{|T|}, \dots, \frac{|C_n|}{|T|} \right) \quad (2)$$

$|C_i|$ désigne le cardinal de la classe i , c'est-à-dire nombre d'éléments de la classe i . L'entropie de T est déduite à partir de l'équation (1), qui donne :

$$E(T) = - \sum_{i=1}^n \frac{|C_i|}{|T|} \log \frac{|C_i|}{|T|} \quad (3)$$

Supposons que l'ensemble des données T est, en plus, partitionné ainsi (T_1, T_2, \dots, T_m). m correspond au nombre de valeurs que peut prendre l'attribut X considéré. Dans ce cas, l'information nécessaire pour identifier la classe d'un individu de T_i devient :

$$E(X, T) = - \sum_{j=1}^m \frac{|T_j|}{|T|} E(T_j) \quad (4)$$

Cas de deux classes

Soit S un ensemble de données d'apprentissage caractérisé par deux classes. P_+ est la proportion d'échantillons positifs et P_- est la proportion d'échantillons négatifs. D'après l'équation 1, l'entropie de l'ensemble S est égale à :

$$E(S) = - p_+ \log p_+ - p_- \log p_- \quad (5)$$

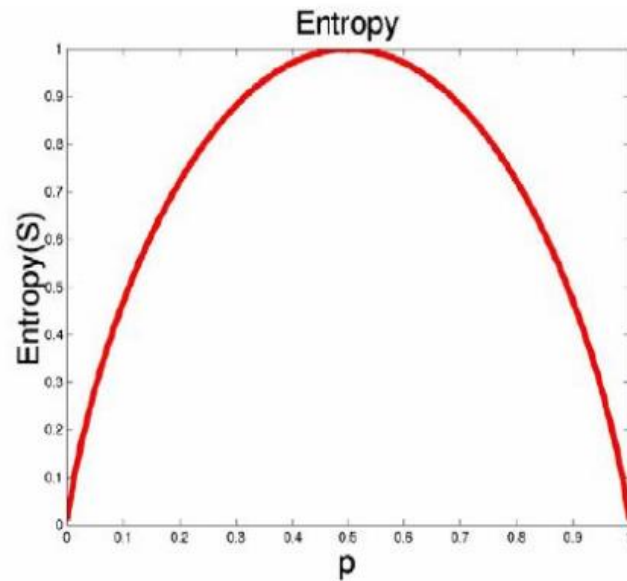


Figure 2 : Entropie - Cas de deux classes

La figure 2 représente la variation de l'entropie dans le cas de deux classes. Sur l'axe des abscisses, nous avons la proportion d'échantillons positifs p_+ et sur l'axe des ordonnées, nous avons l'entropie. Cette fonction d'entropie peut être interprétée selon :

- $p_+ + p_- = 1$ (Puisque la somme des proportions est égale à l'unité).
- $0 \leq E(S) \leq 1$.
- Si $p_+ = 0$ ou $p_- = 0$, alors $E(X) = 0$. En effet, $p_+ = 0$ veut dire que tous les échantillons sont positifs. Dans ce cas, la quantité d'information est nulle.
- Si $p_+ = p_- = 0.5$, alors $E(X) = 1$: ainsi, s'il y a autant de positifs que de négatifs, l'entropie est maximale (Figure 2).

6.2. Gain d'informations

Soit un ensemble de données T , le gain d'informations de T par rapport à une partition T_j donnée est la variation d'entropie causée par la partition de T selon T_j .

$$Gain(X, T) = E(T) - E(X, T) = E(T) - \sum_{j=1}^m \frac{|T_j|}{|T|} E(T_j) \quad (6)$$

6.3. Indice de Gini

L'indice de Gini a été introduit par Breiman en 2001. Cet indice mesure l'impureté, qui est un concept très utile dans la construction des arbres de décision : La qualité d'un nœud et son pouvoir discriminant peuvent être évalués par son impureté. L'indice Gini est donné par la relation suivante :

$$Gini(T) = 1 - \sum_{j=1}^m \left(\frac{|T_j|}{|T|} \right)^2 \quad (7)$$

7. Exemple pratique

Pour mettre en pratique l'algorithme ID3, nous allons utiliser le jeu de données suivant, qui contient des informations météorologiques et une décision de jouer au tennis ("Play"). En suivant les étapes de l'algorithme ID3, nous allons construire un arbre de décision. Voici le jeu de données :

N°	Temps	Température	Humidité	Vent	Jouer
1	Ensoleillé	Chaude	Haute	Non	Non
2	Ensoleillé	Chaude	Haute	Oui	Non
3	Nuageux	Chaude	Haute	Non	Oui
4	Pluvieux	Douce	Haute	Non	Oui

5	Pluvieux	Fraîche	Normale	Non	Oui
6	Pluvieux	Fraîche	Normale	Oui	Non
7	Nuageux	Fraîche	Normale	Oui	Oui
8	Ensoleillé	Douce	Haute	Non	Non
9	Ensoleillé	Fraîche	Normale	Non	Oui
10	Pluvieux	Douce	Normale	Non	Oui
11	Ensoleillé	Douce	Normale	Oui	Oui
12	Nuageux	Douce	Haute	Oui	Oui
13	Nuageux	Chaude	Normale	Non	Oui
14	Pluvieux	Douce	Haute	Oui	Non

Figure 3 : jeux de données

7.1. Calcul de l'entropie pour l'ensemble de données complet T

Pour notre jeu de données, nous avons 14 instances au total :

- 9 instances où Jouer = Oui
- 5 instances où Jouer = Non

$$\text{Donc } P_{oui} = \frac{9}{14} \text{ et } P_{non} = \frac{5}{14}.$$

Puisque nous avons deux classes (Oui, Non), et selon l'équation (5), nous calculons l'entropie initiale de l'ensemble des données :

$$E(T) = - \left(\frac{9}{14} \log_2 \left(\frac{9}{14} \right) + \frac{5}{14} \log_2 \left(\frac{5}{14} \right) \right)$$

$$E(T) \approx 0.940$$

7.2. Calcul de l'entropie conditionnelle pour chaque attribut

Calcul de l'entropie pour l'attribut "Temps"

L'attribut "Temps" peut prendre les valeurs : Ensoleillé, Nuageux, Pluvieux.

- Ensoleillé (5 instances) : 2 Oui, 3 Non

$$E(\text{Ensoleillé}) = - \left(\frac{2}{5} \log_2 \left(\frac{2}{5} \right) + \frac{3}{5} \log_2 \left(\frac{3}{5} \right) \right) \approx 0.971$$

- Nuageux (4 instances) : 4 Oui, 0 Non

$$E(\text{Nuageux}) = - \left(\frac{4}{4} \log_2 \left(\frac{4}{4} \right) \right) = 0$$

- Pluvieux (5 instances) : 3 Oui, 2 Non

$$E(\text{Pluvieux}) = - \left(\frac{3}{5} \log_2 \left(\frac{3}{5} \right) + \frac{2}{5} \log_2 \left(\frac{2}{5} \right) \right) \approx 0.971$$

Selon l'équation (4), l'entropie conditionnelle pour "Temps" est :

$$E(\text{Temps}, T) = \frac{5}{14} \times 0.971 + \frac{4}{14} \times 0 + \frac{5}{14} \times 0.971 \approx 0.693$$

Calcul de l'entropie pour l'attribut "Température"

L'attribut "Température" peut prendre les valeurs : Chaude, Douce, Fraîche.

- Chaude (4 instances) : 2 Oui, 2 Non

$$E(\text{Chaude}) = - \left(\frac{2}{4} \log_2 \left(\frac{2}{4} \right) + \frac{2}{4} \log_2 \left(\frac{2}{4} \right) \right) = 1$$

- Douce (6 instances) : 4 Oui, 2 Non

$$E(\text{Douce}) = - \left(\frac{4}{6} \log_2 \left(\frac{4}{6} \right) + \frac{2}{6} \log_2 \left(\frac{2}{6} \right) \right) \approx 0.9179$$

- Fraîche (4 instances) : 3 Oui, 1 Non

$$E(\text{Fraîche}) = - \left(\frac{3}{4} \log_2 \left(\frac{3}{4} \right) + \frac{1}{4} \log_2 \left(\frac{1}{4} \right) \right) \approx 0.811$$

Selon l'équation (4), l'entropie conditionnelle pour "Température" est :

$$E(\text{Température}, T) = \frac{4}{14} \times 1 + \frac{6}{14} \times 0.9179 + \frac{4}{14} \times 0.811 \approx 0.9108$$

Calcul de l'entropie pour l'attribut "Humidité"

L'attribut "Humidité" peut prendre les valeurs : Haute, Normale.

- Haute (7 instances) : 3 Oui, 4 Non

$$E(\text{Haute}) = - \left(\frac{3}{7} \log_2 \left(\frac{3}{7} \right) + \frac{4}{7} \log_2 \left(\frac{4}{7} \right) \right) \approx 0.983$$

- Normale (7 instances) : 6 Oui, 1 Non

$$E(\text{Normale}) = - \left(\frac{6}{7} \log_2 \left(\frac{6}{7} \right) + \frac{1}{7} \log_2 \left(\frac{1}{7} \right) \right) \approx 0.591$$

Selon l'équation (4), l'entropie conditionnelle pour "Humidité" est :

$$E(\text{Humidité}, T) = \frac{7}{14} \times 0.983 + \frac{7}{14} \times 0.591 \approx 0.787$$

Calcul de l'entropie pour l'attribut "Vent"

L'attribut "Vent" peut prendre les valeurs : Non, Oui.

- Non (8 instances) : 6 Oui, 2 Non

$$E(\text{Non}) = - \left(\frac{6}{8} \log_2 \left(\frac{6}{8} \right) + \frac{2}{8} \log_2 \left(\frac{2}{8} \right) \right) \approx 0.811$$

- Oui (6 instances) : 3 Oui, 3 Non

$$E(\text{Oui}) = - \left(\frac{3}{6} \log_2 \left(\frac{3}{6} \right) + \frac{3}{6} \log_2 \left(\frac{3}{6} \right) \right) = 1$$

Selon l'équation (4), l'entropie conditionnelle pour "Vent" est :

$$E(\text{Vent}, T) = \frac{8}{14} \times 0.811 + \frac{6}{14} \times 1 \approx 0.892$$

7.3. Calcul du gain d'information pour chaque attribut

Selon l'équation (6) :

$$\text{Gain}(X, T) = E(T) - E(X, T)$$

– Gain (Temps) :

$$\text{Gain(Temps, } T) = 0.940 - 0.693$$

$$\text{Gain(Temps, } T) \approx 0.247$$

– Gain (Température) :

$$\text{Gain(Température, } T) = 0.940 - 0.9108$$

$$\text{Gain(Température, } T) \approx 0.0292$$

– Gain (Humidité) :

$$\text{Gain(Humidité, } T) = 0.940 - 0.787$$

$$\text{Gain(Humidité, } T) \approx 0.153$$

– Gain (Vent) :

$$\text{Gain(Vent, } T) = 0.940 - 0.892$$

$$\text{Gain(Vent, } T) \approx 0.048$$

7.4. Choisir l'attribut avec le gain d'information le plus élevé

Attribut	Entropie conditionnelle	Gain d'information
Temps	0.693	$0.940 - 0.693 = 0.247$
Température	0.9108	$0.940 - 0.9108 = 0.0292$
Humidité	0.787	$0.940 - 0.787 = 0.153$
Vent	0.892	$0.940 - 0.892 = 0.048$

Figure 4 Calcul de l'Entropie et du Gain d'Information pour chaque Attribut

L'attribut "Temps" a le gain d'information le plus élevé (0.247). Nous choisissons donc "Temps" comme le premier nœud de notre arbre de décision.

7.5. Répéter les étapes de manière récursive

Nous divisons le data-set en trois sous-ensembles en fonction des valeurs de l'attribut "Temps" (Ensoleillé, Nuageux, Pluvieux) et appliquons les étapes 1 à 4 à chaque sous-ensemble jusqu'à ce que nous atteignons des nœuds feuilles ou que tous les attributs soient épuisés.

Pour simplifier les résultats obtenus à chaque itération, nous allons présenter les calculs d'entropie et de gain d'information dans des tableaux séparés pour chaque étape de l'algorithme. Chaque tableau montrera les résultats intermédiaires :

– Cas 1 : **Ensoleillé**

Attribut	Entropie conditionnelle	Gain d'information
Température	0.4	$0.971 - 0.4 = 0.571$
Humidité	0	$0.971 - 0 = 0.971$
Vent	0.9508	$0.971 - 0.9508 = 0.0202$

Figure 5 : Division basée sur l'Attribut "Temps" - Cas Ensoleillé

Pour "Ensoleillé", l'attribut "Humidité" est choisi avec un gain d'information de 0.971.

– Cas 2 : **Nuageux**

Pour "Nuageux", il n'y a pas besoin de diviser plus car toutes les instances sont "Oui".

– Cas 3 : **Pluvieux**

Attribut	Entropie conditionnelle	Gain d'information
Température	0.95098	$0.971 - 0.95098 = 0.02002$
Humidité	0.9502	$0.971 - 0.9502 = 0.0208$
Vent	0	$0.971 - 0 = 0.971$

Figure 6 : Division basée sur l'Attribut
"Temps" - Cas Nuageux

Pour " Nuageux ", l'attribut "Vent" est choisi avec un gain d'information de 0.971.

Pour construire les sous-arbres, nous répétons le processus en sélectionnant la meilleure caractéristique pour diviser les données, puis en continuant à diviser les sous-ensembles de données jusqu'à ce que les conditions d'arrêt soient atteintes. Cela implique de calculer l'entropie et l'information de gain pour chaque caractéristique, puis de diviser les données en conséquence, jusqu'à ce que nous obtenions des sous-ensembles homogènes ou que d'autres critères définis soient satisfaits. Ce processus récursif nous permet de construire un arbre de décision complet en tenant compte des relations entre les données et en respectant les conditions d'arrêt définies.

7.6. Arbre de décision

Après un certain nombre d'itérations, nous obtenons l'arbre de décision suivant :

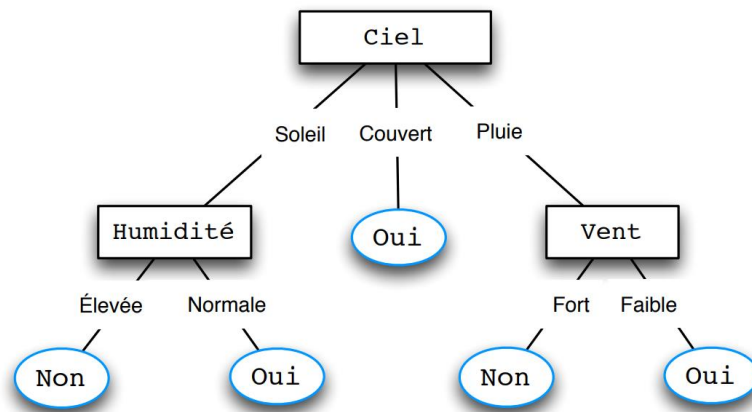


Figure 7 : Arbre de décision obtenue

8. Conclusion

Dans ce chapitre, nous avons d'abord présenté l'algorithme de l'arbre de décision, en mettant en avant ses avantages et ses limites. Nous avons ensuite approfondi notre compréhension en fournissant des explications détaillées sur le fonctionnement de cet algorithme, en précisant ses fondements mathématiques. Pour illustrer concrètement ces concepts, nous avons terminé par un exemple pratique de construction d'un arbre de décision en utilisant l'algorithme ID3. Cette démarche a permis de démontrer non seulement l'efficacité de l'algorithme, mais aussi les étapes méthodiques nécessaires pour sa mise en œuvre. En résumé, ce chapitre a offert une vue d'ensemble complète, allant de la théorie à la pratique, pour une compréhension approfondie de l'algorithme de l'arbre de décision. Cette exploration nous a équipé des connaissances nécessaires pour appliquer cet algorithme dans divers contextes et comprendre ses implications dans le domaine de l'apprentissage automatique.

Chapitre 2

IMPLEMENTATION ET RESULTATS EXPERIMENTAUX

1. Introduction

Dans ce chapitre, nous plongeons dans les méandres de notre implémentation personnalisée de l'algorithme de l'arbre de décision, réalisée à l'aide du langage de programmation Java. Nous détaillons les étapes de construction de cet arbre, notamment la sélection des meilleurs attributs, la division des données, et la création des feuilles de l'arbre.

Ensuite, nous confrontons les performances de notre implémentation aux résultats produits par le logiciel **Weka**, une référence dans le domaine de l'apprentissage automatique. Nous analysons en profondeur les différences entre les taux de classification correcte et incorrecte, les matrices de confusion, ainsi que les mesures de précision, de rappel et de F-mesure, sur deux ensembles de données distincts.

Enfin, nous clôturons ce chapitre en examinant de près les différentes facettes de notre application desktop dédiée à la classification par arbre de décision. Nous mettons en lumière les fonctionnalités offertes aux utilisateurs, telles que l'importation des données, l'entraînement des modèles, l'évaluation des performances, et la visualisation graphique des arbres de décision générés.

2. Implémentation de l'Arbre de Décision et de la Forêt

Aléatoire

Dans notre projet, nous avons développé une implémentation complète des algorithmes d'arbre de décision et de forêt aléatoire en Java. Ces implémentations combinent les principes de l'algorithme ID3, offrant ainsi une classification efficace.

2.1. Arbre de décision

Notre implémentation de l'arbre de décision repose sur plusieurs classes représentant les différents composants de l'arbre : les nœuds, les attributs, les branches et les instances.

- **Nœuds et Attributs** : Les nœuds représentent les attributs ou les feuilles. Les nœuds intermédiaires divisent les données selon les valeurs des attributs, et les feuilles représentent les décisions finales.
- **Branches** : Chaque nœud possède des branches correspondant aux différentes valeurs possibles de son attribut. Les branches mènent à des sous-nœuds ou des feuilles.
- **Instances** : Les instances sont les enregistrements de données à classer, contenant des valeurs pour chaque attribut et une classe cible.

2.2. Algorithme ID3

Dans notre implémentation de l'algorithme ID3, nous avons développé une méthode pour construire des arbres de décision à partir de données d'entraînement. Voici une explication détaillée de ce que nous avons dans notre implémentation de ID3 :

- **Sélection du Meilleur Attribut** : À chaque étape de la construction de l'arbre, notre implémentation sélectionne l'attribut qui maximise le gain d'information. Pour ce faire, nous avons développé des méthodes pour calculer l'entropie et le gain d'information, nécessaires à cette sélection.
- **Division des Données** : Une fois l'attribut sélectionné, les données sont divisées en sous-ensembles en fonction des valeurs de cet attribut. Nous avons mis en œuvre des mécanismes pour diviser les données en sous-groupes homogènes, conformément à l'algorithme ID3.

- **Répétition du Processus** : Le processus de sélection du meilleur attribut et de division des données se répète de manière itérative pour chaque sous-ensemble ainsi créé. Cela se fait de manière récursive jusqu'à ce que toutes les instances d'un sous-ensemble appartiennent à la même classe ou que tous les attributs aient été utilisés pour diviser les données.
- **Création des Feuilles de l'Arbre** : Lorsque le processus de construction de l'arbre est terminé, les feuilles de l'arbre représentent les décisions finales. Chaque feuille correspond à une classe de sortie, et les instances qui parviennent à cette feuille sont classées en fonction de cette classe.

Les données sont traitées pour calculer les mesures nécessaires à la construction de l'arbre :

- **Entropie** : Mesure de l'incertitude dans les données, calculée pour chaque attribut.
- **Gain d'Information** : Réduction d'entropie après division des données par un attribut.
- **Classe Majoritaire** : Déterminée pour les feuilles de l'arbre pour décider de la classification finale.

2.3. CART

Pour notre implémentation de l'algorithme CART, nous avons suivi un processus légèrement différent en utilisant l'indice de Gini pour déterminer les meilleures divisions :

- **Initialisation et Entraînement de l'Arbre** : Nous avons commencé par initialiser l'arbre et les classes, puis nous avons formé l'arbre en utilisant un ensemble de données d'entraînement.

- **Recherche de la Meilleure Division** : À chaque nœud, nous avons cherché la meilleure division possible en évaluant chaque attribut et ses valeurs potentielles. La meilleure division a été déterminée par la minimisation de l'indice de Gini, qui mesure l'impureté des sous-ensembles de données résultants.
- **Construction de l'Arbre** : Une fois la meilleure division trouvée, nous avons divisé les données en fonction de cette division et avons assigné les sous-ensembles résultants à des branches menant à des nœuds enfants. Ce processus a été répété de manière récursive pour chaque nœud enfant.
- **Création des Nœuds Feuilles** : Si aucune division valable n'a été trouvée, le nœud a été transformé en feuille et la classe majoritaire parmi les instances restantes a été assignée à ce nœud.

2.4. Forêt Aléatoire (*Random Forest*)

Nous avons également implémenté une forêt aléatoire, qui est une collection d'arbres de décision. Chaque arbre est construit en utilisant un sous-ensemble aléatoire des attributs et des données.

Voici comment se construit une forêt aléatoire :

- **Sous-ensembles d'Attributs** : Pour chaque arbre, un sous-ensemble d'attributs est sélectionné aléatoirement.
- **Échantillonnage Bootstrap** : Pour chaque arbre, un échantillon Bootstrap des instances est généré, ce qui signifie que certaines instances peuvent apparaître plusieurs fois dans l'échantillon, tandis que d'autres peuvent être absentes.

- **Construction des Arbres** : Chaque arbre est construit en utilisant l'algorithme ID3 amélioré avec le sous-ensemble d'attributs et l'échantillon Bootstrap.

Pour évaluer une instance, chaque arbre de la forêt fournit une prédiction. La classe majoritaire parmi ces prédictions est choisie comme la classification finale.

- **Entraînement** : Chaque arbre est entraîné sur un échantillon Bootstrap des données.
- **Prédiction** : Pour chaque instance, les prédictions des arbres sont collectées et la classe majoritaire est déterminée.
- **Matrice de Confusion** : Une matrice de confusion est générée pour évaluer la précision de la forêt aléatoire, en comparant les classes prédites aux classes réelles des instances de test.

2.5. Évaluation des Performances

Une fois l'arbre construit, les performances du modèle sont évaluées en utilisant des techniques de validation :

- **Training Data-set** : Utilisation des mêmes données pour l'entraînement et la validation.
- **Cross-Validation** : Division des données en k sous-ensembles et utilisation de chaque sous-ensemble à tour de rôle comme ensemble de test.
- **Percentage Split** : Division des données en un pourcentage pour l'entraînement et un autre pour le test.

Pour mesurer la précision, nous générons une matrice de confusion à partir des données de test, comparant les classes prédites aux classes réelles.

3. Démonstration des interfaces de l'application

Dans cette partie, nous allons démontrer les différentes interfaces de notre application de classification, développée en Java en utilisant le Framework **JavaFx**. Ces interfaces permettent aux utilisateurs de charger des données, d'entraîner des modèles de classification, d'évaluer les performances des modèles et de visualiser les arbres de décision générés. Nous illustrerons chaque interface avec des captures d'écran et fournirons des explications détaillées sur leurs fonctionnalités et leur utilisation

3.1. Page d'accueil

Sur la page d'accueil principale, nous avons une barre latérale permettant de naviguer entre les différentes sections de l'application. Sur le côté droit, différentes scènes seront affichées en fonction de l'interaction de l'utilisateur.

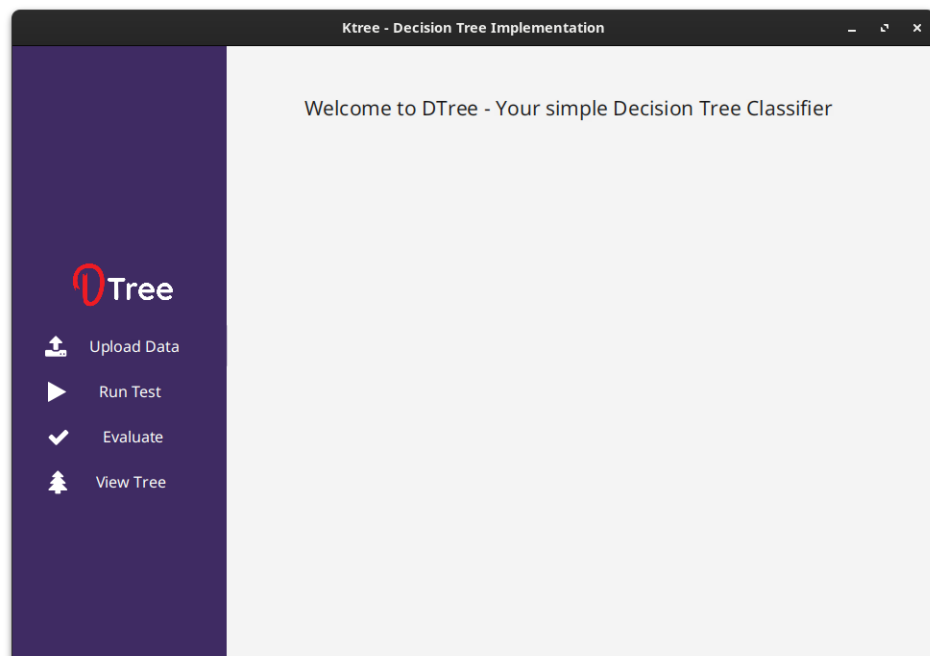
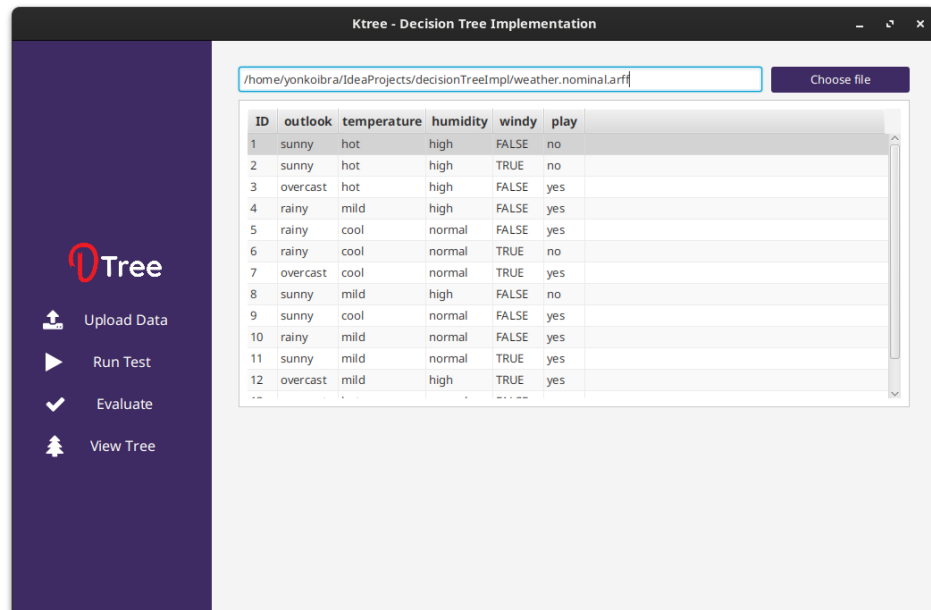


Figure 8 : Interface accueil

3.2. Importation des Données

L'interface d'importation des données permet aux utilisateurs de charger un fichier au format **".arff"**. Une fois le fichier chargé, les instances de données sont affichées sous forme de tableau. Cette visualisation permet aux utilisateurs de vérifier les données avant de procéder à l'entraînement du modèle.



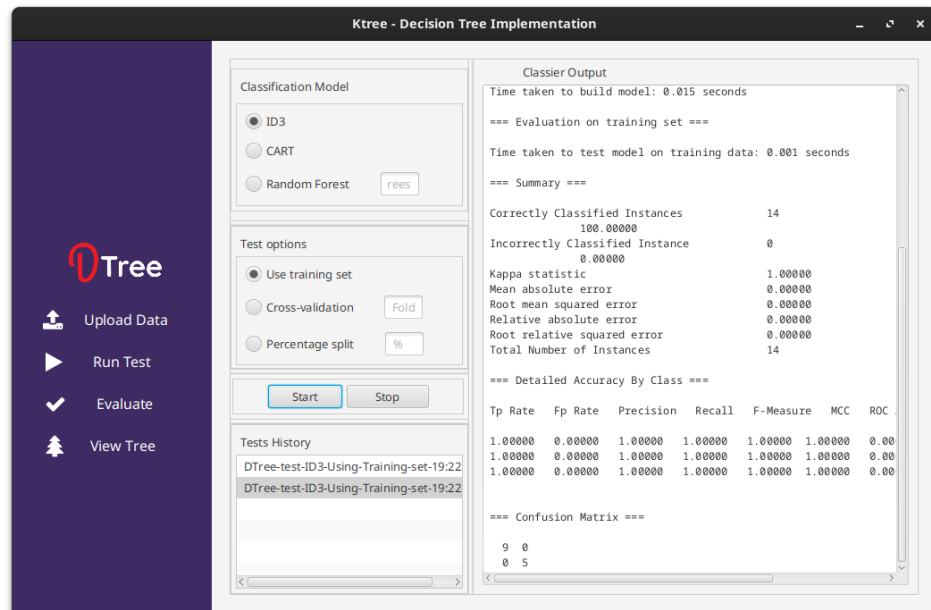
The screenshot shows a web application window titled "Ktree - Decision Tree Implementation". On the left is a dark purple sidebar with the "DTree" logo and four menu items: "Upload Data", "Run Test", "Evaluate", and "View Tree". The main area has a file input field containing the path "/home/yonkoibra/IdeaProjects/decisionTreeImpl/weather.nominal.arff" and a "Choose file" button. Below this is a table displaying 12 data instances.

ID	outlook	temperature	humidity	windy	play
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes

Figure 9 : Interface d'importation des données

3.3. Entraînement des Modèles

L'interface d'entraînement des modèles de notre application est conçue pour permettre à l'utilisateur de sélectionner et d'entraîner différents modèles de classification, tels que l'ID3, CART, et les forêts aléatoires.



Voici une description détaillée des composants et des fonctionnalités de cette interface :

– Modèle de Classification

Cette section permet à l'utilisateur de choisir le type d'algorithme de classification à utiliser pour l'entraînement.

Voici les options disponibles :

- **ID3** : Sélectionne l'algorithme ID3 pour l'entraînement du modèle.
- **CART** : Sélectionne l'algorithme CART pour l'entraînement du modèle.
- **Random Forest** : Sélectionne l'algorithme de forêt aléatoire pour l'entraînement du modèle.

– Options de Test

Cette section permet de choisir la méthode de validation pour évaluer la performance du modèle entraîné.

Voici les options disponibles :

- **Use training set** : Utilise l'ensemble de données d'entraînement pour évaluer le modèle.
- **Cross validation** : Utilise la validation croisée avec la possibilité de définir le nombre de **Folds**.
- **Percentage split** : Permet de spécifier le pourcentage de données à utiliser pour l'entraînement et le test.

– Boutons de Contrôle

Ces boutons permettent de démarrer ou d'arrêter le processus d'entraînement du modèle.

Voici les options disponibles :

- **Start** : Démarre l'entraînement du modèle sélectionné avec les options de test spécifiées.
- **Stop** : Arrête le processus d'entraînement en cours.

– Historique des Tests

Cette section affiche l'historique des tests effectués, incluant le type de modèle et les options de test utilisées.

– Sortie du Classificateur

Cette section présente les résultats détaillés de l'entraînement et de l'évaluation du modèle.

– **Informations Affichées**

- **Informations générales** : Détails sur le schéma, les instances, les attributs, et le mode de test utilisé.
- **Temps d'Entraînement** : Durée nécessaire pour entraîner le modèle.
- **Évaluation sur le Jeu de Données** : Résultats de l'évaluation du modèle sur le jeu de données d'entraînement ou de test.
- **Statistiques Résumées** : Inclut des métriques telles que le nombre d'instances correctement et incorrectement classifiées, le coefficient Kappa, et les erreurs absolues et relatives.
- **Précision Détaillée par Classe** : Inclut les taux de vrais positifs, faux positifs, précision, rappel, F-mesure, MCC, ROC Area, PRC Area par classe.
- **Matrice de Confusion** : Montre la matrice de confusion, fournissant une vue détaillée des prédictions correctes et incorrectes pour chaque classe.

3.4. Évaluation

Cette interface permet aux utilisateurs de saisir une nouvelle instance en utilisant les attributs existants du jeu de données. Les valeurs possibles pour chaque attribut sont automatiquement affichées dans des boîtes des choix, permettant à l'utilisateur de les sélectionner facilement. Après avoir effectué cette sélection, ils peuvent cliquer sur le bouton "Évaluer" pour obtenir la prédiction de la classe à laquelle appartient cette nouvelle instance. Les résultats sont ensuite affichés dans un tableau.

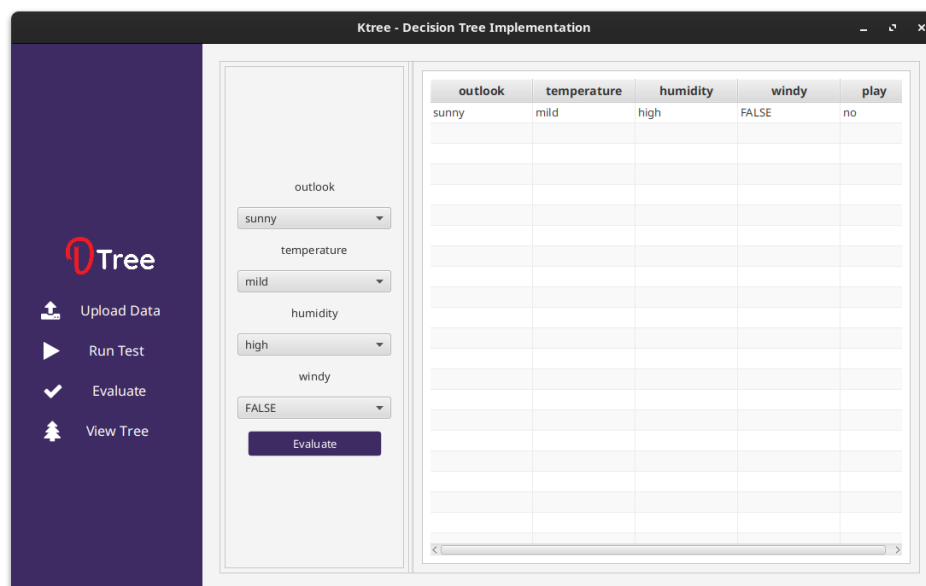


Figure 10 : Interface d'Évaluation

3.5. Visualisation d'Arbre de Décision

Cette interface offre une visualisation graphique des arbres de décision générés par les modèles. Elle permet aux utilisateurs de voir la structure de l'arbre, les attributs utilisés à chaque nœud, ainsi que les décisions prises à chaque feuille. Cela aide à mieux comprendre comment le modèle arrive à ses prédictions.

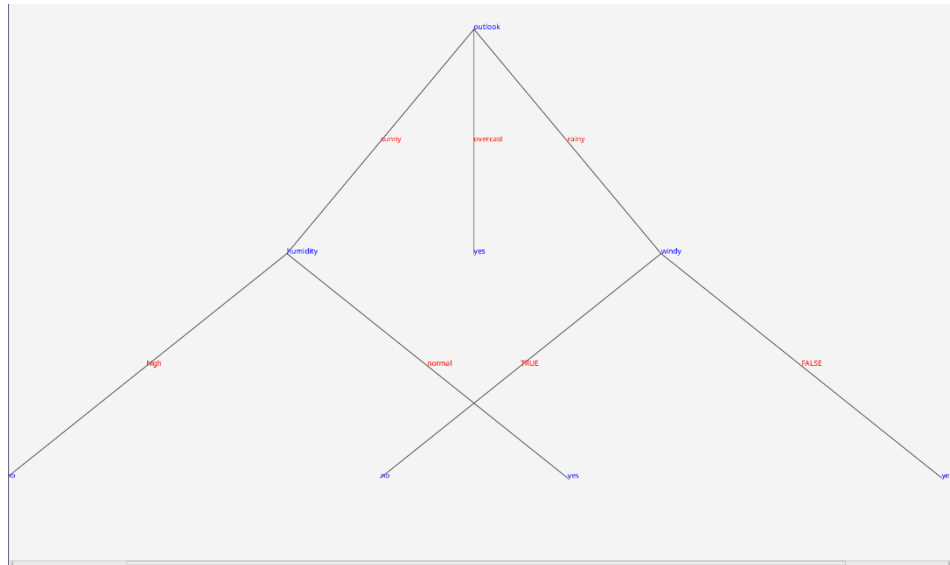


Figure 11 : Interface de Visualisation d'Arbre de Décision

4. Comparaison des Résultats des Algorithmes d'Arbres de Décision

Cette section présente une comparaison des performances de trois algorithmes de classification implémentés en interne, à savoir Id3, CART et Forêt aléatoire, par rapport aux versions correspondantes des algorithmes fournis par Weka : Id3, J48 (C4.5), et Random Forest. La comparaison a été réalisée sur deux jeux de données distincts, « [contact-lenses.arff](#) » et « [breast-cancer.arff](#) », afin d'évaluer la précision, le rappel et la F-mesure de chaque algorithme dans des contextes différents

Dans cette partie, nous allons effectuer des comparaisons avec le logiciel Weka sur le même jeu de données en utilisant une segmentation de « 80% pour l'entraînement et 20% pour les tests » ainsi qu'une validation croisée avec 10 échantillons. Nous utiliserons deux algorithmes déjà présents dans Weka : le premier est l'ID3, et l'autre est le modèle J48, qui est basé sur l'algorithme de création C4.5, une extension de l'ID3. Ensuite, nous comparerons les résultats avec l'algorithme ID3 que nous avons codé.

4.1. Jeux de caractères

- **Contact-lenses.arff** : Ce jeu de données contient 24 instances et 5 attributs.
- **Breast-cancer.arff** : Ce jeu de données contient un nombre significativement plus élevé d'instances et d'attributs, ce qui permet d'évaluer la robustesse des algorithmes sur des données plus complexes.

4.2. Méthodologie

Pour chaque algorithme, nous avons évalué les performances en utilisant le même jeu de données, en mesurant les taux de classification correcte et incorrecte, les matrices de confusion, la précision moyenne, le rappel moyen, et la mesure F moyenne.

4.3. Résultats

Résultats sur « contact-lenses.arff » :

Algorithme	Instances correctement classifiées (%)	Instances incorrectement classifiées (%)	Matrice de confusion	Précision (Moy.)	Rappel (Moy.)	F-Mesure (Moy.)
Notre Id3	100	0	$\begin{bmatrix} 5 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 15 \end{bmatrix}$	1	1	1
Notre CART	100	0	$\begin{bmatrix} 5 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 15 \end{bmatrix}$	1	1	1
Notre Random Forest	100	0	$\begin{bmatrix} 5 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 15 \end{bmatrix}$	1	1	1
Id3 de Weka	100	0	$\begin{bmatrix} 5 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 15 \end{bmatrix}$	1	1	1
J48 de Weka	91.6667	8.3333	$\begin{bmatrix} 5 & 0 & 0 \\ 0 & 3 & 1 \\ 1 & 0 & 15 \end{bmatrix}$	0.833	1	0.909

Random Forest de Weka	100	0	$\begin{bmatrix} 5 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 15 \end{bmatrix}$	1	1	1
-----------------------	-----	---	--	---	---	---

Figure 12 : Tableau de comparaison de « contact-lenses.arff »

Résultats sur « breast-cancer.arff » :

Algorithme	Instances correctement classifiées (%)	Instances incorrectement classifiées (%)	Matrice de confusion	Précision (Moy.)	Rappel (Moy.)	F-Mesure (Moy.)
Notre Id3	97.47292	2.52708	$\begin{bmatrix} 194 & 2 \\ 5 & 76 \end{bmatrix}$	0.97462	0.96403	0.96913
Notre CART	97.47292	2.52708	$\begin{bmatrix} 195 & 1 \\ 6 & 75 \end{bmatrix}$	0.97850	0.96041	0.96889
Notre Random Forest	75.45126	24.54874	$\begin{bmatrix} 194 & 2 \\ 66 & 15 \end{bmatrix}$	0.81425	0.58749	0.57850
J48 de Weka	75.8741	24.1259	$\begin{bmatrix} 194 & 7 \\ 62 & 23 \end{bmatrix}$	0.760	0.759	0.716
Random Forest de Weka	97.9021	2.0979	$\begin{bmatrix} 198 & 3 \\ 3 & 82 \end{bmatrix}$	0.979	0.979	0.979

Figure 13 : Tableau de comparaison de « breast-cancer.arff »

4.4. Analyse des Résultats

– Notre Id3 par rapport à J48 de Weka

→ Sur le jeu de données « contact-lenses.arff »

Notre implémentation d'Id3 atteint une précision parfaite avec 100% d'instances correctement classifiées, tandis que la version de Weka atteint seulement 70.8333%.

→ Sur le jeu de données « breast-cancer.arff »

Notre Id3 a également des performances très élevées (97.47292%) comparées à J48 de Weka (75%).

- **Notre CART par rapport à Id3**

- Sur « contact-lenses.arff »

Notre implémentation de CART atteint 100% de précision, ce qui montre une performance parfaite.

- Sur « breast-cancer.arff »

Notre CART a des performances très élevées (97.47292%), très similaires à celles de notre Id3

- **Notre Forêt aléatoire par rapport à Random Forest de Weka**

- Sur « contact-lenses.arff »

Les deux implémentations de Random Forest montrent des résultats parfaits (100%).

- Sur « breast-cancer.arff »

Notre Forêt aléatoire montre des performances nettement inférieures (75.45126%) comparées à celles de Weka (97.9021%).

- **J48 (C4.5) de Weka**

- Sur « contact-lenses.arff »

J48 de Weka atteint une précision élevée de 91.6667%, mais est surpassé par nos implémentations de Id3, CART et Forêt aléatoire.

- Sur « breast-cancer.arff »

J48 atteint 75.8741%, ce qui est supérieur à l'Id3 de Weka mais inférieur à notre Id3 et CART ainsi qu'au Random Forest de Weka.

5. Conclusion

Ce chapitre a été une exploration approfondie de notre implémentation de l'algorithme de l'arbre de décision, comparée aux résultats générés par le logiciel Weka, ainsi qu'une présentation détaillée de notre application desktop dédiée à la classification par arbre de décision.

Nous avons pu constater que notre implémentation personnalisée de l'algorithme a produit des performances compétitives, voire supérieures dans certains cas, par rapport à celles de Weka. Nos résultats ont démontré des taux de classification correcte élevés et des métriques de précision, de rappel et de F-mesure satisfaisantes sur les ensembles de données testés.

Cependant, des variations ont été observées selon les jeux de données et les algorithmes spécifiques. Notamment, notre Forêt aléatoire a présenté des performances inférieures sur un ensemble de données plus complexe par rapport à Weka, indiquant la nécessité d'optimisations supplémentaires.

En outre, notre application desktop offre une interface conviviale pour les utilisateurs, facilitant l'importation des données, l'entraînement des modèles, l'évaluation des performances et la visualisation graphique des arbres de décision générés.

En conclusion, ce chapitre a mis en lumière les avantages et les défis associés à notre implémentation de l'algorithme de l'arbre de décision, ainsi que les fonctionnalités de notre application desktop. Il ouvre la voie à des recherches futures visant à améliorer et à optimiser notre approche, dans le but d'atteindre une robustesse comparable à celle des outils de référence comme Weka.

CONCLUSION GENERAL

Dans cette étude, nous avons d'abord présenté l'algorithme de l'arbre de décision en examinant ses fondements théoriques et en illustrant son fonctionnement avec un exemple pratique utilisant l'algorithme ID3. Cette exploration nous a permis de comprendre en profondeur le processus de création d'un arbre de décision.

Ensuite, nous avons abordé l'implémentation de l'algorithme dans une application réelle. Nous avons détaillé les étapes de développement, en mettant l'accent sur les interfaces utilisateur et les fonctionnalités de l'application. Nous avons comparé les résultats obtenus avec ceux générés par le logiciel Weka, ce qui nous a permis d'évaluer la robustesse et l'efficacité de notre solution par rapport à une référence bien établie.

En somme, cette recherche nous a offert une perspective enrichissante sur l'algorithme de l'arbre de décision, de sa théorie à sa mise en pratique. Nous avons pu démontrer la puissance de cet outil pour la classification et la prédiction, tout en reconnaissant les possibilités d'amélioration et d'optimisation. Les futures recherches pourraient se concentrer sur l'exploration d'autres algorithmes d'arbres de décision et sur l'ajustement des paramètres pour améliorer les performances. Cette étude constitue donc une base solide pour des développements ultérieurs dans le domaine de la modélisation prédictive.

WEBOGRAPHIE

“Arbre de Décision : Le Guide Complet (+3 Exemples Utiles et Concrets).”

Accessed June 8, 2024. <https://everlaab.com/arbre-de-decision/>.

Discuss Career & Computing - OpenGenus. “Using ID3 Algorithm to Build a Decision Tree to Predict the Weather - Software Engineering,” June 23, 2019.

<https://discourse.opengenus.org/t/using-id3-algorithm-to-build-a-decision-tree-to-predict-the-weather/3343>.

Discuss Career & Computing - OpenGenus. “Using ID3 Algorithm to Build a Decision Tree to Predict the Weather - Software Engineering,” June 23, 2019.

<https://discourse.opengenus.org/t/using-id3-algorithm-to-build-a-decision-tree-to-predict-the-weather/3343>.

