


UNIVERSITÉ DE STRASBOURG
INTELLIGENCE ARTIFICIELLE
Licence 3 - UFR Mathématique - Informatique

Concevoir et analyser et comparer des modèles d'apprentissage automatique

Instructions

- Ce projet est à réaliser en binôme.
- Vous devez remettre votre projet au plus tard le **3 mai 2024** via le formulaire Moodle prévu à cet effet. Les retards seront pénalisés de 2 points si rendu le 4 mai et 3 points si rendu le 5 mai. Les travaux rendus après le 5 mai recevront un zéro d'office.
- Un seul rendu par binôme est attendu. Votre rendu doit impérativement comporter :
 - Les sources de votre projet (fichiers `.py` dans une archive `zip`, ou notebook `.ipynb` propre et bien organisé) ;
 - Un compte-rendu au format `pdf` contenant les réponses aux questions posées dans le sujet, identifiées par le symbole . Toutes les réponses doivent être justifiées (méthodologie, concepts sous-jacents, ...). *Les notebooks jupyter ne sont pas acceptés comme compte-rendu.*
 - Le formulaire d'auto-évaluation complété (en annexe du sujet).
- Le projet comporte quatre parties. Les parties 3 et 4 sont indépendantes des parties 1 et 2 : même si vous ne parvenez pas à obtenir de résultats avec vos modèles, vous pouvez faire les parties traitant de l'analyse des modèles car des prédictions et les données de test correspondantes vous sont fournies pour faire les parties 3 et 4.

Compétences évaluées :

- Préparation des données (10%)
- Mise en œuvre des modèles “arbre de décision” et “réseaux de neurones artificiel” (50%) ;
- Le calcul de différentes métriques d'évaluation et leur interprétation (30%) ;
- Compréhension générale et ouverture (10%).

Pour faciliter notre évaluation de l'implémentation de vos modèles, vous devrez inclure dans vos programmes des tests pour les fonctionnalités suivantes :

- Arbre de décision • calcul de l'entropie d'une partition
- Arbre de décision • calcul du gain d'une partition
- Arbre de décision • détermination d'un meilleur partitionnement
- Réseau de neurones • passe avant avec une instance
- Réseau de neurones • rétropropagation et mise à jour après une passe avant

Les fonctionnalités plus larges (construction de l'arbre, mises en place d'époques d'entraînement, *etc.* seront mises à l'épreuve en générant des prédictions avec les différents modèles donc pas de test spécifique à faire ici : soit vous avez pu générer des prédictions, soit ça n'est pas le cas.

1 Préparation des données

Les données que vous utiliserez dans ce travail sont des données synthétiques générées automatiquement avec Sklearn. Le fichier `synthetique.csv` est à télécharger sur Moodle. Vous mettrez de côté 20% des données de côté pour tester vos modèles. Un aperçu de la distribution d'un échantillon de 100 instances selon deux attributs est visible sur la figure 1.

✎ Questions

1. Combien d'attributs comportent ces données ?
2. En combien de classes différentes les instances sont-elles catégorisées ?
3. Combien d'instances chaque classe compte-t-elle ?
4. Les données sont-elles linéairement séparables ?
5. Aurez-vous besoin, pour l'un des types de modèle ou les deux, d'utiliser un encodage en *one-hot* ? de normaliser les données ?
6. Rappelez l'intérêt de séparer les données en un jeu d'entraînement et un jeu de test.

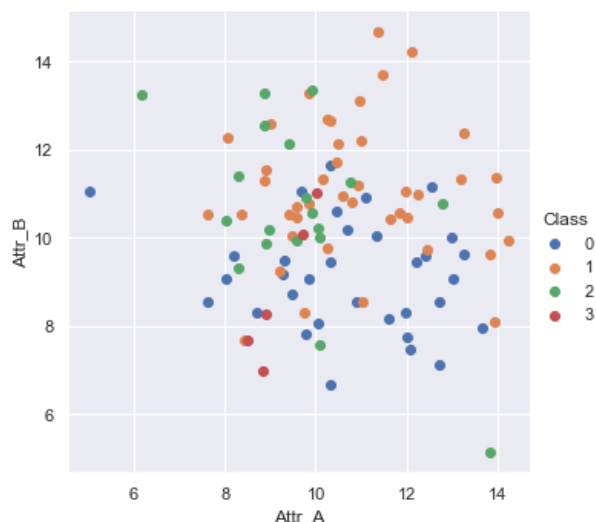


FIGURE 1 – Distribution d'un échantillon de 100 instances selon les attributs A et B

2 Mise en œuvre des modèles

Vous devez réaliser l'apprentissage sur les données avec deux types de modèles différents : arbres de décision et réseaux de neurones artificiels.

2.1 Arbre de décision

- Pour discrétiser les attributs, vous devez cette fois faire un split en utilisant des **quartiles** (cf. section 3.2 du TP sur les arbres de décisions). Attention, nous sommes toujours dans le cadre d'un arbre de décision binaire !

Vous utiliserez la fonction `quantile` de Pandas^a. Pour mesurer le gain d'information, il faudra comparer successivement les partitions $]attribute\ min\ value; quantile_n[$ et $[quantile_n; attribute\ max\ value[$ afin de déterminer quel quartile n constitue la meilleure valeur de split.

✎ À quoi correspondent les $quantile_n$? Comment les avez vous calculé ?

- Entraînez plusieurs modèles avec des profondeurs maximales différentes (entre 3 et 8). Testez les performances de vos modèles avec votre jeu de test ; vous garderez les prédictions de ce que vous estimerez être vos deux meilleurs modèles pour réaliser l'analyse des modèles (partie 3).

a. <https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.quantile.html>

2.2 Réseaux de neurones artificiels

- Vous devrez mettre de côté 15% de votre jeu d'entraînement (Fig. 2) pour constituer un jeu de *validation* afin de mettre en œuvre l'*early stopping* qui arrêtera automatiquement l'apprentissage de votre modèle. Vous utiliserez une patience de 4.
 - L'apprentissage devra se réaliser par mini-batch de 4 instances.
 - Entraînez plusieurs modèles avec les architectures suivantes :
 - Avec une activation `tanh` : (10,8,6), (10,8,4) et (6,4)
 - Avec une activation `relu` : (10,8,6), (10,8,4) et (6,4)
- Testez vos modèles avec le jeu de test. Vous garderez les prédictions de ce que vous estimerez être vos deux meilleurs modèles pour chaque version (`tanh` et `relu`) afin de réaliser l'analyse des modèles (partie 3).



FIGURE 2 – Découpage des données pour la mise en œuvre de l'*early stopping* : prélèvement de 15% des données d'entraînement pour constituer un jeu de *validation*.

3 Analyse des modèles

Si vous n'avez pas obtenu de résultats avec vos modèles, vous avez à disposition les données de prédictions (et le `y_test` correspondant) dans l'archive `predictions.zip`. Indiquez clairement dans votre compte-rendu si vous avez utilisé les prédictions fournies ou les vôtres.

Votre objectif dans cette partie est de calculer (sans utiliser de bibliothèque extérieure à votre programme) différentes métriques qui vous permettront de mieux comprendre les comportements des modèles. Pour chaque jeu de prédictions conservé lors de la partie précédente, vous calculerez :

- l'exactitude
- la précision
- le rappel
- le F1-score

On parle bien ici d'évaluer ces métriques pour chaque classe possible (autrement dit, on ne s'intéresse pas au fait, par exemple, qu'un modèle fait globalement $x\%$ de prédictions justes, mais à sa capacité à prédire correctement une classe k au regard de l'ensemble des prédictions). Dit autrement, il s'agit en fait toujours d'une analyse binaire, en considérant une classe c_i versus toutes les autres.

✎ Pour chaque modèle, présentez ces métriques sous forme d'un tableau de synthèse tel qu'illustré dans le tableau 1 ci-dessous.

✎ Calculez les matrices de confusion pour chaque modèle (ici encore, sans utiliser de bibliothèque extérieure à votre programme) et reportez les résultats dans des tableaux tels que celui illustré ci-dessous (table 2).

TABLE 1 – Classification report pour un modèle m

	Modèle m			
Classes	c_1	c_2	c_3	c_4
Accuracy				
Precision				
Recall				
F1-score				

TABLE 2 – Matrice de confusion pour un modèle m

		Modèle m			
		0	1	2	3
True label	0				
	1				
	2				
	3				
	Predicted label				

4 Le meilleur modèle

✎ Comparez l'ensemble de vos différents modèles : quel est, selon vous, le meilleur modèle ? Choisiriez-vous un des modèles préférentiellement aux autres et si oui, pourquoi ? (pour vous aidez, vous pouvez imaginer ce que pourraient représenter les données utilisées : diagnostics médicaux, anomalies dans un système critique, *etc.*)

✎ Sur ces données, les différences de performances observées entre les arbres de décision et les réseaux de neurones ne sont pas très grandes. Partant de là, choisiriez-vous préférentiellement l'un ou l'autre type de modèle si vous deviez justifier les décisions du modèle ? (si votre modèle est utilisé pour faire du diagnostic médical, on peut aisément imaginer que les patients auront à cœur de comprendre une décision prise par votre modèle).

Vos réponses doivent être étayées !

A Formulaire d'auto-évaluation

Membres du binôme :

Nom 1	Prénom 1
Nom 2	Prénom 2

Les données :

Vous avez effectué, en amont de l'entraînement, les traitements nécessaires selon le type de modèle utilisé

Avancement du développement des modèles :

L'arbre de décision est totalement opérationnel

Si non, cochez les fonctionnalités opérationnelles :

- Calcul de l'entropie d'une partition de données
- Calcul du gain d'une partition
- Détermination de la partition offrant le meilleur gain
- Construction de l'arbre (apprentissage du modèle)
- Utilisation de l'arbre pour faire des prédictions sur le jeu de test

Le réseau de neurones est totalement opérationnel

Si non, cochez les fonctionnalités opérationnelles :

- Propagation avant
- Rétropropagation
- Époques d'entraînement
- Early stopping
- Utilisation du réseau pour faire des prédictions sur le jeu de test

Calcul des métriques d'évaluation :

L'ensemble des métriques d'évaluation demandées est calculé

Si non, cochez les métriques calculées :

- Exactitude
- Précision
- Rappel
- F1-score

Les matrices de confusion sont calculées

Expression libre