Casamian Marius/(cm205773)

Introduction à l'Intelligence Artificielle

Licence 2 — Portail Sciences et Technologies

Travaux dirigés No 8 : Apprentissage supervisé—courbe ROC

Objectif du TP :

- **Utilisation du jeu de donnée wine de Sickit-Learn**
- **Arbre de décision vu dans le dernier TP**
- **La courbe ROC**

Réalisation du TP :

Après avoir importé les librairies et dépendances nécessaires à la réalisation du TP, on charge un jeu de données de vins qu'on stocke dans la variable **wine**.
On exploite ensuite les données en deux **DataFrame** (X,y) ;
Dans notre cas, le jeu de données de vins comporte **trois classes** représentées par des entiers 0, 1 et 2.

La fonction **label_binarize()** convertit la variable cible **y** d'une colonne de classe 0,1 ou 2 en une matrice binaire.

Chaque colonne correspond à une classe et chaque ligne indique la présence ou l'absence de cette classe pour l'échantillon qui correspond.

Cette partie est importante car la matrice binaire est ensuite exploitée dans l'analyse de la courbe ROC.
Exemple : Si l'échantillon avait une étiquette 1, la représentation binaire de la matrice serait [0,1,0] après l'appelle de la fonction **label_binarize()**. On évalue donc trois classificateurs binaires.

Ensuite comme nous l'avons vu dans plusieurs TP on construit un jeu d'entrainement (70%) et de test (30%) en premier, puis, 50% et 50% dans un deuxième temps.
En résumé cela créer un modèle à partir de données connues(entrainement) afin de de le reproduire sur des données inconnues (test).

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.30, random_state = 0)
```

On construit le modèle avec un arbre de décision comme suit :

```
model_decisiontree = DecisionTreeClassifier(criterion = "gini", random_state = 100, max_depth = 3, min_samples_leaf = 5)
```

Et on l'utilise pour prédire :

```
y_score = model_decisiontree.predict(X_test)
```

```
[0 0 1]
 [0 1 0]
 [1 0 0]
 [0 1 0]
 [0 1 0]
 [1 0 0]
 [0 0 1]
 [0 1 0]
 …
 …
 [0 0 1]
 [1 0 0]
 [0 0 1]
 [0 0 1]
 [1 0 0]
 [0 0 1]]
```

Ce résultat partiel La variable **y_score** contient les prédictions du modèle d'arbre de décision sur l'ensemble de test. Chaque ligne de cette matrice correspond à la prédiction pour un échantillon particulier dans l'ensemble de test, et chaque colonne correspond à une classe spécifique. Les valeurs binaires indiquent si le modèle prédit la présence (1) ou l'absence (0) de chaque classe.

Exemple d'analyse pour la première ligne de **y_score** :

- La première colonne (indice 0) est à 1, indiquant que le modèle prédit la classe 0 pour le premier échantillon.

- Les colonnes suivantes (indices 1 et 2) sont à 0, indiquant que le modèle prédit l'absence des classes 1 et 2 pour le premier échantillon.

En résumé, pour chaque échantillon, la classe prédite est celle pour laquelle la valeur est 1, et les autres classes sont prédites comme étant absentes (0).

Cette partie du code effectue une analyse de la courbe pour évaluer les performances du modèle d'arbre de décision sur un problème de classification.

//Explication du programme python :

1.Initialisation des dictionnaires :

tfp (faux positifs), **tvp** (vrais positifs), et **roc_auc** (aire sous la courbe) sont initialisés en tant que dictionnaires pour stocker les résultats pour chaque classe.

2. Boucle sur les classes :

Une boucle for parcourt chaque classe.

À chaque itération, la fonction **roc_curve** est utilisée pour calculer les taux de faux positifs (**tfp**), les taux de vrais positifs (**tvp**), et les seuils de décision pour la classe actuelle.

3. Calcul de l'aire sous la courbe (AUC) :

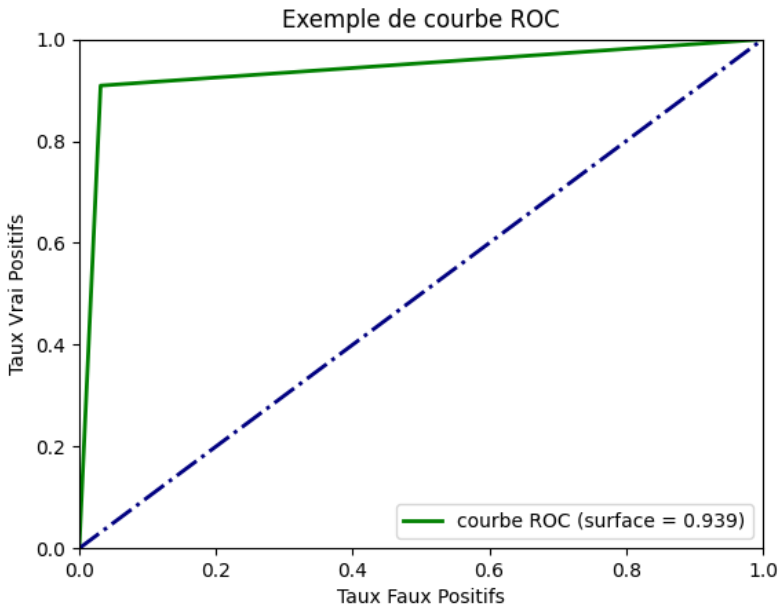
Pour chaque classe, l'aire sous la courbe ROC est calculé à l'aide de la fonction **auc** à partir des taux de faux positifs et des taux de vrais positifs.

4. Tracé de la courbe ROC pour une classe spécifique :

La courbe ROC pour une classe spécifique (ici, la classe avec l'indice 1) est tracée à l'aide de plt.plot. Les taux de faux positifs (tfp[1]) sont représentés sur l'axe des x, les taux de vrais positifs (tvp[1]) sur l'axe des y. La couleur est définie en vert.

Une ligne en pointillés (navy) est tracée pour représenter la ligne de référence aléatoire.

La légende est ajoutée pour indiquer l'aire sous la courbe ROC pour la classe spécifique.



La courbe ROC permet une représentation visuelle de la performance d'un modèle de classification.

La ligne en pointillés diagonale de couleur navy représente le modèle aléatoire, où le taux de faux positifs est égal au taux de vrais positifs. Tout modèle performant devrait être au-dessus de cette ligne.

Chaque courbe ROC représente la performance du modèle pour une classe spécifique. Plus la courbe est éloignée de la ligne aléatoire vers le coin supérieur gauche, meilleure est la performance.

L'aire sous la courbe ROC (AUC) est un indicateur quantitatif de la performance globale du modèle. Une AUC proche de 1 indique une excellente capacité discriminante, tandis qu'une AUC de 0.5 suggère une performance aléatoire.

On peut comparer les performances entre les différentes classes en examinant les AUC correspondantes. Cela peut révéler si le modèle est plus performant pour certaines classes que pour d'autres, mais en particulier dans ce cas le modèle est performant comme le montre le ROC = 0.939.

Conclusion :

Nous avons vu les objectifs du TP et découvert la courbe ROC en pratique qui confirme un modèle performant de classification.