0.1 Knn

0.1.1 Definition

Decision Tree est un algorithme d'apprentissage supervisé utilisé pour la classification et la régression. Il fonctionne en divisant récursivement l'espace des caractéristiques en sous-ensembles basés sur des conditions de seuil sur les caractéristiques, formant ainsi une structure arborescente où chaque nœud interne représente une condition de décision et chaque feuille représente une prédiction de classe ou de valeur.

Les critères couramment utilisés pour mesurer la qualité des divisions incluent l'indice de Gini, le gain d'information (entropie) et la réduction de la variance.

Les paramètres clés de l'algorithme Decision Tree incluent :

- **Max Depth :** La profondeur maximale de l'arbre, généralement utilisé pour contrôler la complexité de l'arbre et éviter le surapprentissage.
- **Min Samples Split :** Le nombre minimum d'échantillons requis pour diviser un nœud, généralement fixé à la valeur minimale de 2.
- **Min Samples Leaf :** Le nombre minimum d'échantillons requis pour être à une feuille, généralement fixé à la valeur minimale de 1.
- **Criterion :** La fonction utilisée pour mesurer la qualité des divisions (par exemple, Gini, Entropie).

0.1.2 Implemenation from scratch

Voici notre implémentation de l'algorithme Decision Tree en Python.

`Implemenation Decision Tree from scratch`

0.2 Normalization et Encodage de données

Decision Tree n'est pas basé sur la distance entre les points de données, donc la normalisation des caractéristiques n'est pas nécessaire dans ce cas. Cependant, l'encodage des caractéristiques catégorielles est important pour permettre à l'algorithme de traiter correctement ces données. Les techniques que nous avons utilisées sont :

- Pour l'encodage des caractéristiques categorilaes, nous avons utilisé **Ordinal Encoding** pour ne pas introduire beaucoup de nouvelle caractéristiques ainsi que pour preserver un order entre les valeurs pour permettre à l'algorithme de mieux séparer les données.

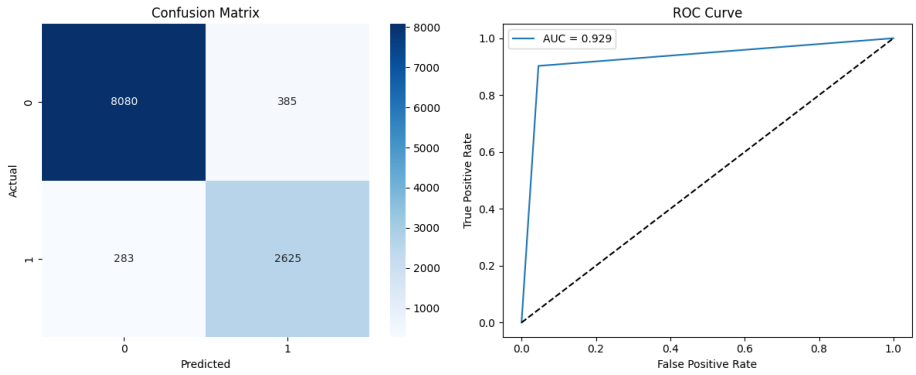

Figure 1: Decision Tree avec données originales

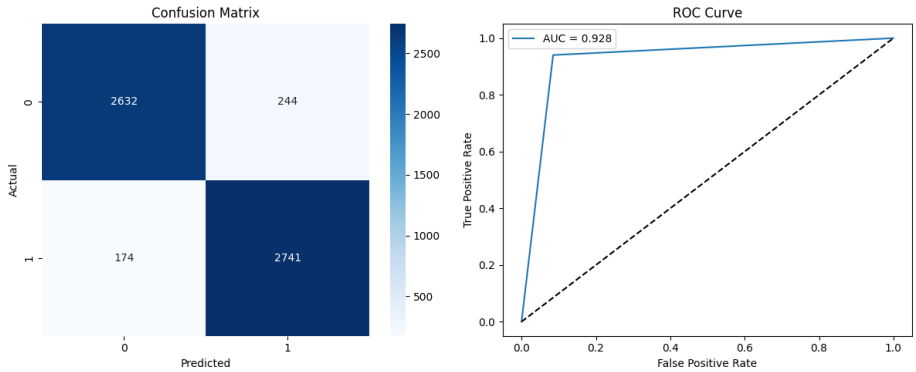

Figure 2: Decision Tree avec données random under sampling

0.3 Balancement de données

L'algorithme de Decision Tree est sensible à la distribution des classes dans les données comme KNN, nous avons donc fait la meme exploration de différentes techniques de balancement pour équilibrer les classes, nous avons utilisé un model avec **criterion='gini'** et le reste des paramètres pas defaut, et le pour la comparaison des resultats nous avons utilisé le métrique **ROC AUC**. Les techniques de balancement que nous avons testées sont :

- **Original Data :**
- **Random Under Sampling :**
- **Ranodom Over Sampling :**
- **Smote Over Sampling :**
- **Tomek Under Sampling :**

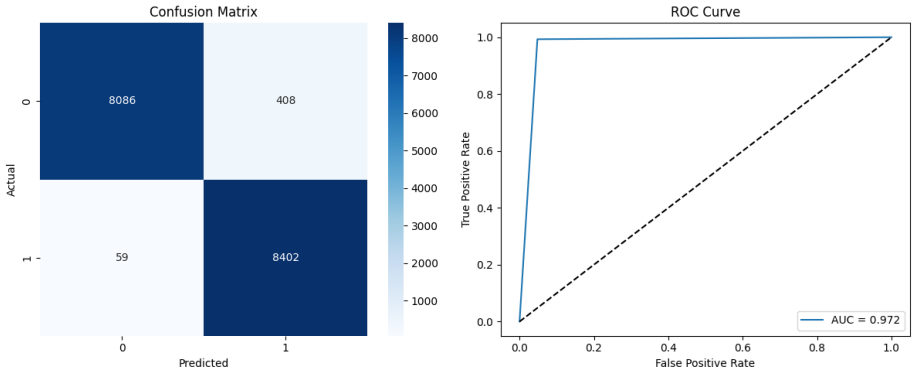

Figure 3: Decision Tree avec données random over sampling

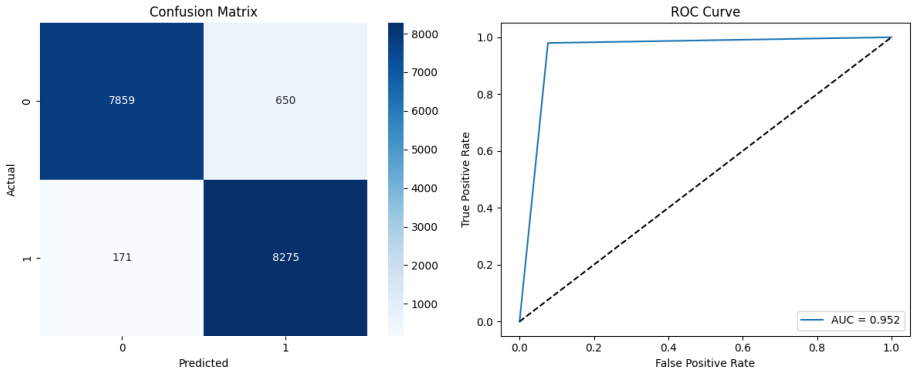

Figure 4: KNN avec données smote over sampling

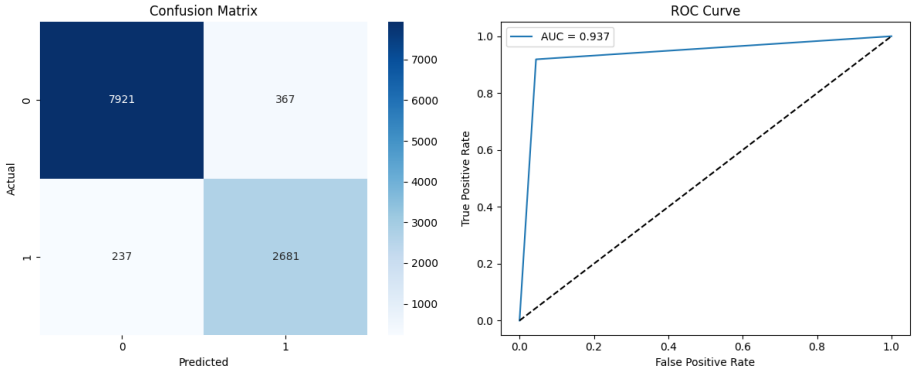

Figure 5: Decision Tree avec données tomek under sampling

À la fin la meilleure technique de balancement pour Decision Tree est **Random Over Sampling** avec les resultats suivants :

- Best sampling method: Random Over-Sampling with ROC AUC: 0.9725
- New dataset sizes: Train=67819, Test=16955, Full=84774
- Class distribution: class 0: 33893, class 1: 33926

0.4 Réglage des paramètres

Pour le réglage des paramètres de l'algorithme Decision Tree, nous avons utilisé la validation croisée (k-folds avec $k = 5$) avec une recherche d'optimisation bayésienne de 50 itérations, la plage des paramètres testés est la suivante :

- **max_depth :** [1, 50] la profondeur maximale de l'arbre
- **min_samples_split :** [2, 50] le nombre minimum d'échantillons requis pour diviser un nœud interne
- **min_samples_leaf :** [1, 50] le nombre minimum d'échantillons requis pour être à une feuille
- **max_features :** [None, "sqrt", "log2"] le nombre de caractéristiques à considérer lors de la recherche de la meilleure division, None signifie que toutes les caractéristiques sont utilisées
- **criterion :** ["gini", "entropy"] la fonction pour mesurer la qualité d'une division

Les meilleurs paramètres trouvés sont sont illustrés dans Table **??**. On remarque que tous les paramètres sont à leurs valeurs minimales sauf max depth qui est limité à 39 pour éviter le surapprentissage.

0.5 Réduction de dimensionnalité

Decision Tree peut être affecté par la malédiction de la dimensionnalité comm KNN, nous avons donc appliqué une réduction de dimensionnalité en utilisant l'analyse en composantes principales (PCA) pour réduire le nombre de caractéristiques tout en conservant 90% de la variance des données. Après l'application de PCA, le nombre de caractéristiques est passé de 38 à 1. Nous avons ensuite répété le processus de réglage des paramètres avec les mêmes plages de valeurs. Les meilleurs paramètres trouvés après la réduction de dimensionnalité sont exactement les mêmes que ceux trouvés avant la réduction de dimensionnalité et sont illustrés dans Table **??**.

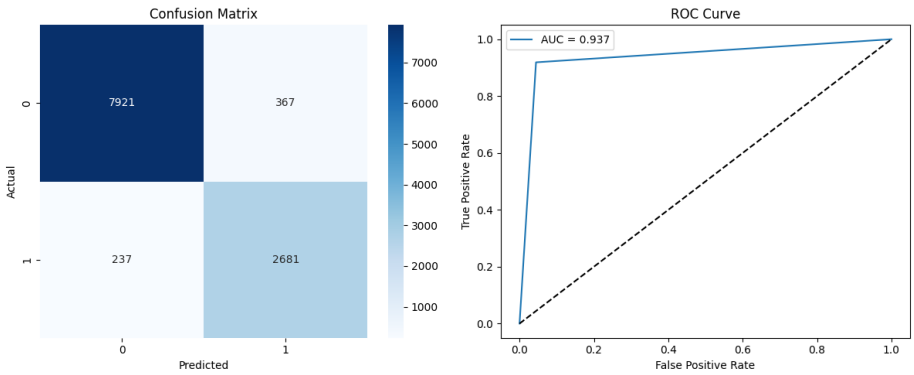

Figure 6: KNN - Final Results with no PCA

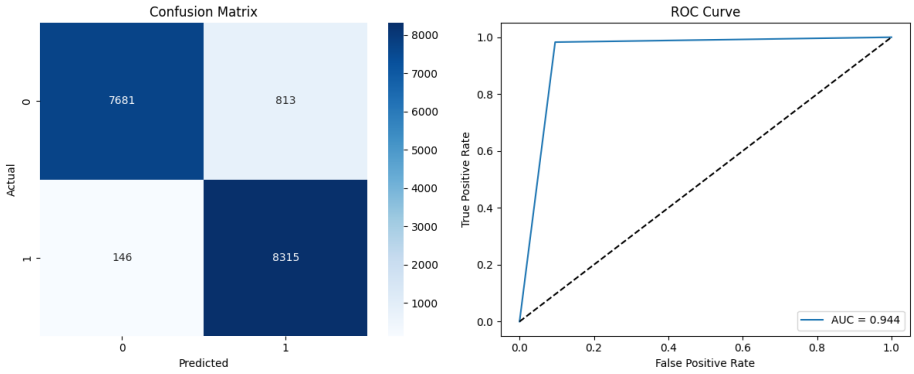

Figure 7: KNN - Final Results with PCA

Configuration	Criterion	Max Depth	Min Samples Split	Min Samples Le
Sans réduction de dimensionnalité	gini	39	2	1
Avec réduction de dimensionnalité	gini	50	2	1

Table 1: Comparaison des performances avec et sans réduction de dimensionnalité

Configuration	Accuracy	Precision	Recall	F1 Score
Sans réduction de dimensionnalité	0.9706	0.9528	0.9902	0.9711
Avec réduction de dimensionnalité	0.9694	0.9509	0.9898	0.9700

Table 2: Comparaison des performances avec et sans réduction de dimensionnalité

0.6 Comparaison de resultats

Les resultats finaux obtenus avec KNN après le réglage des paramètres et l'utilisation de la technique de balancement Random Over Sampling sont les suivants :

Nous observons que la réduction de dimensionnalité n'a pas eu un impact significatif sur les performances de l'algorithme KNN dans ce cas, mais elle a permis de réduire significativement le temps de calcul et les ressources nécessaires pour l'entraînement et la prédiction du modèle en réduisant tous les caractéristiques du dataset en une seule composante.