Rapport TDM4 : Introduction aux sciences des données et à l'intelligence artificielle

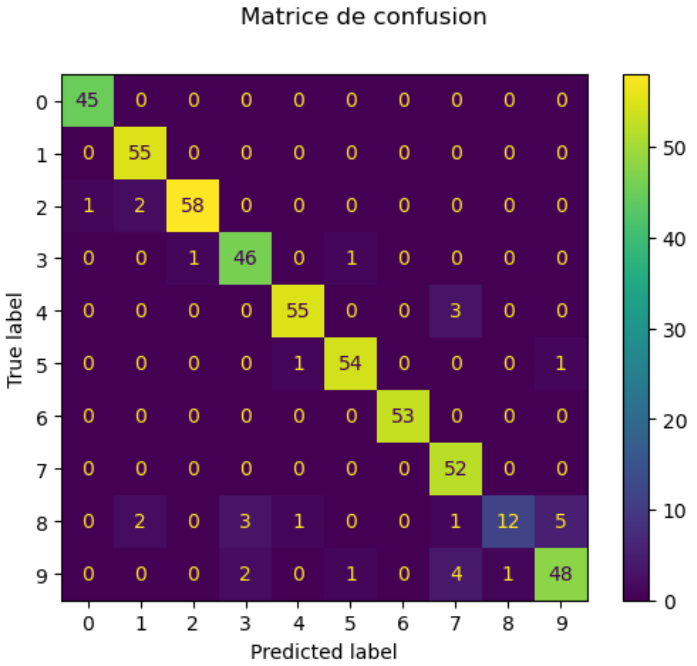
Fanani Selma et Amsaf Rim (Groupe 4)

October 4, 2024

Effets du déséquilibre des données sur la précision des prédictions

Pour ce TDM, nous avons à nouveau utilisé le jeu de données digits afin d'étudier l'impact de la réduction des exemples d'une classe sur les performances d'un modèle de classification. Nous avons créé un nouveau jeu de données en supprimant 60 % des exemples appartenant à la classe 8, réduisant ainsi leur nombre. Le code effectue cette suppression en identifiant les exemples de la classe 8 et en retirant une partie d'entre eux, laissant uniquement 40 % des exemples initiaux. Le nouveau jeu de données modifié, constitué de **Xunb** (qui contient l'ensemble des caractéristiques des exemples restants) et de **yunb** (qui contient les labels correspondants), est identique au jeu original en termes de structure, à l'exception du nombre réduit d'exemples pour la classe 8. Ce jeu sera utilisé pour évaluer comment cette réduction affecte la capacité du modèle à prédire correctement les différentes classes.

Afin d'évaluer l'impact de cette réduction sur la performance du modèle, nous avons entraîné et testé un classifieur K-Nearest Neighbors sur un hold-out de 30 % du jeu de données modifié (Xunb, yunb). En divisant les données avec un **train_test_split** (70 % pour l'entraînement et 30 % pour le test), nous avons pu mesurer la qualité des prédictions à travers la matrice de confusion. Cette matrice permet de visualiser les erreurs de classification entre les différentes classes et d'analyser la manière dont le modèle traite spécifiquement la classe 8, désormais sous-représentée.



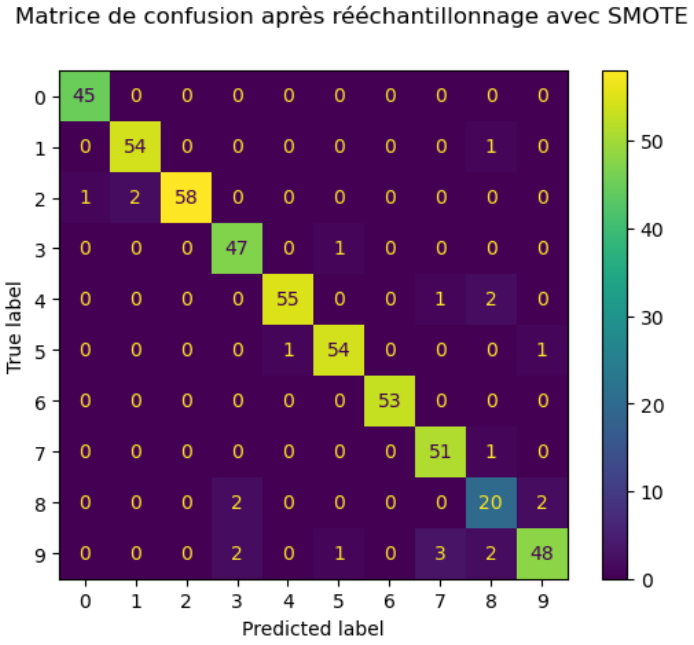
Nous remarquons que, sur les 24 données appartenant à la classe 8, seulement 12 ont été prédites correctement. Cela illustre bien l'impact du phénomène de déséquilibrage des données sur l'efficacité

du modèle. En observant le modèle ayant appris sur les données d'origine, nous notons que le nombre d'erreurs de prédiction pour la classe 8 est bien plus élevé pour le modèle déséquilibré.

Supposons maintenant que nous disposons uniquement des données `Xunb`, avec une réduction de 60% des données pour la classe 8. Plusieurs solutions existent pour améliorer les résultats. Nous pourrions, par exemple, ajouter des données artificielles à la classe 8, afin de permettre à notre modèle d'apprendre plus efficacement à prédire cette classe. Une autre approche consisterait à réduire la quantité de données des classes majoritaires pour rééquilibrer le jeu de données. Il existe également un paramètre, `class_weight='balanced'`, disponible dans certains algorithmes d'apprentissage de la bibliothèque `sklearn`, qui permet d'attribuer un poids plus élevé aux classes minoritaires, afin de compenser le déséquilibre.

Nous avons décidé d'opter pour la première méthode. La raison de ce choix est qu'avec le jeu de données d'origine, les résultats pour la classe 8 sont relativement satisfaisants, nous pensons donc que l'ajout de données artificielles pourrait améliorer d'avantage les performances de notre modèle. Pour cela, nous utilisons la technique `SMOTE` de la bibliothèque `imblearn`. Cette méthode génère des données artificielles à partir des données existantes. Plus précisément, `SMOTE` (Synthetic Minority Over-sampling Technique) crée de nouvelles instances synthétiques pour les classes minoritaires en interpolant entre les exemples réels, ce qui permet de rééquilibrer le jeu de données et d'améliorer la capacité du modèle à prédire les classes sous-représentées.

Nous obtenons alors la matrice de confusion suivante :



Nous remarquons que le même modèle (en utilisant toujours `random_state=42`) fournit de meilleurs résultats de prédiction pour les éléments de la classe 8. En effet, il ne commet que 4 erreurs sur 20 données, contre 12 erreurs pour le modèle précédent. Cela montre que l'utilisation de données artificielles avec `SMOTE` permet d'améliorer significativement la capacité du modèle à prédire les instances de la classe minoritaire.

References

[1] Imbalanced-learn Documentation. *Scikit-learn User Guide*. https://imbalanced-learn.org/stable/

[2] Vous retrouverez le code directement en suivant ce lien. https://github.com/SelmaFanani/TDM4