PSL UNIVERSITÉ PARIS

MIDO
MATHÉMATIQUES ET INFORMATIQUE
DE LA DÉCISION ET DES ORGANISATIONS

Dauphine UNIVERSITÉ PARIS

Projet Apprentissage Statistique
Sujet 4 : Ensemble de données déséquilibrées (Imbalanced data)

Par : Aminata Ndiaye

Table des matières

1 Introduction

Dans un ensemble de données déséquilibrées, l'estimateur bayésien classique donne toujours un modèle simplifié dit "naïf". Le seuil de précision dit *accuracy* est très élevé bien que le modèle ne détecte pas les données minoritaires. L'objectif de ce projet sera donc de présenter les méthodes permettant d'équilibrer ces données et d'avoir donc une prédiction plus représentative.
Dans le cadre de l'étude d'ensembles de données déséquilibrées, les différentes méthodes abordées seront illustrées par des données de fraudes bancaires.
La base de données qui sera étudiée est Credit Card Fraud Detection (CCFD) qui contient des transactions faites par des détenteurs de cartes bancaires européens en 2013. Sur les 285 296 transactions enregistrées on compte 492 fraudes, ce qui en fait une base de donnée très deséquilibrée. La CCFD compte 30 features. Les features $V1, \ldots, V28$ sont obtenues par ACP à partir des features originaux et les features "Times" et "Amount" qui contiennent respectivement le temps entre chaque transaction de la base de donnée et la première transaction enregistrée et la quantité de cette transaction.(Anonyme (s. d.))

2 Métriques

Les métriques que nous allons présenter dans cette section mesureront l'efficacité du modèle de prédiction. Nous présenterons la matrice de confusion, le F1-score, la courbe ROC et AU ROC.
Pour illustrer ces grandeurs, l'approche adoptée est d'entraîner un modèle de régression logistique (avec comme solver sag) sur les données de la CCFD où le label 1 signifiera que l'interaction est frauduleuse et le label 0 que l'interaction est non frauduleuse. Nous en déduirons ensuite la matrice de confusion, le F-score, la courbe ROC et la AU ROC associés. (Rocca (2019))

2.1 Matrice de confusion et F1-Score

— **Matrice de confusion :** La matrice de confusion donne une vision globale de la performance d'un modèle. La matrice se restreint à deux classes 0 et 1, et donne les "faux positifs" et "vrais positifs" prédits par le modèle.
De ces quantifications, l'*accuracy* du modèle peut être extraite, ainsi que la *precision* et le *recall* par classes.

	étiquette prédite (0)	étiquette prédite (1)
étiquette réelle (0)	Corrects ("vrais positifs pour la classe 0)	Faux ("faux positifs" pour la classe 1)
étiquette réelle (1)	Faux ("faux positifs pour la classe 0)	Corrects ("vrais positifs" pour la classe 1)

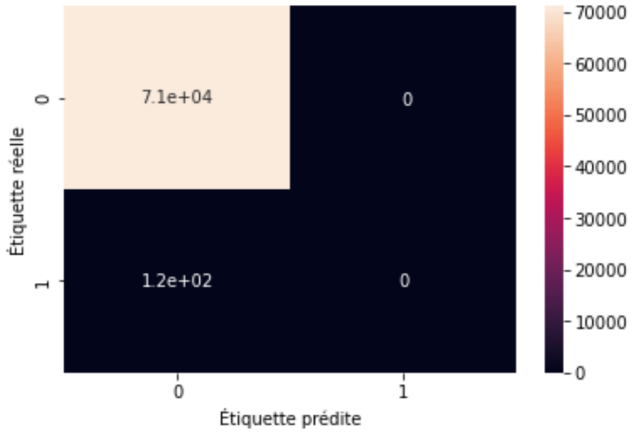
La précision globale du modèle, *accuracy*, est donnée par : $accuracy =$ 
Par classe :
— Classe 0 : $precision_0 =$  et $recall_0 =$ 
— Classe 1 : $precision_1 =$  et $recall_1 =$ 
Théoriquement, la précision (*precision*) et le rappel (*recall*) de la classe i sont définis pour tout $i \in \{0, 1\}$ par :

$$\text{précision}_i = \frac{\text{nombre d'éléments correctement attribués à la classe i}}{\text{nombre total d'éléments attribués à la classe i}}$$

$$\text{rappel}_i = \frac{\text{nombre d'éléments correctement attribués à la classe i}}{\text{nombre d'éléments appartenant à la classe i}}$$

La précision est aussi appelée *valeur prédictive positive* et le rappel *sensibilité*.

La matrice de confusion obtenue avec nos données est :



Toutes les données sont prédites comme étant d'étiquette 0 avec environ 71000 qui sont vraiment de classe 0 et environ 120 qui ne le sont pas. Il y a ici le caractère "naïf" de notre modèle qui prédit toutes les données comme appartenant à la classe majoritaire de l'échantillon d'entraînement.

— **F1-score :** Le F1-score varie entre 0 et 1 et évalue la performance du modèle et résume la précision et le rappel. Il s'agit de la moyenne harmonique entre la précision et le rappel, autrement dit :

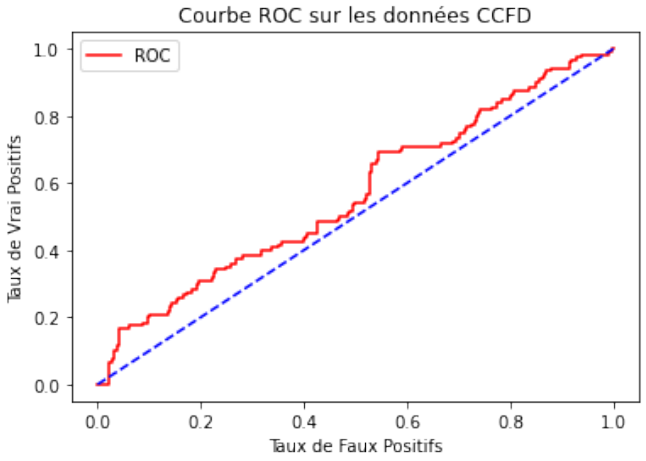
$$F1_{score} = \frac{2 \times precision \times recall}{precision + recall} = \frac{2}{\frac{1}{precision} + \frac{1}{recall}}$$

Il compare les prédictions positives correctes aux erreurs faites par le modèle. Par exemple :
si $F1_{score} = 50\%$, pour une prédiction positive correcte, le modèle fait deux erreurs.
Le $F1_{score}$ obtenue est 0.0.

2.2 Courbe ROC et AU ROC

Une autre métrique utilisée est la courbe ROC qui signifie *Receiver Operating Characteristic* définie pour une classe donnée. Pour un point donné x, le modèle nous donnera la probabilité que x appartienne à la classe C_i pour $i \in \{0, 1\}$. Cette probabilité est notée $P(x \in C_i)$. Selon cette probabilité, on dit que $x \in C$ si et seulement si $P(x \in C_i) \geq T$, pour un seuil $T \in [0, 1]$ donné.
Si $T = 1$, un point x est étiqueté comme appartenant à C_i si le modèle est sûr à 100%. Au contraire, si $T = 0$, tous les points sont étiquetés comme appartenant à C_i (la précision est nulle). Chaque valeur du seuil T génère un point (faux positif, vrai positif), et ensuite la courbe ROC décrira l'ensemble de ces points générés par T lorsqu'il varie entre 0 et 1. Cette courbe commence à $(0, 0)$ et termine à $(1, 1)$ et est strictement croissante (dans notre cas elle est croissante par morceaux car en données discrètes). Un bon modèle est un modèle dont la courbe ROC augmente rapidementde 0 à 1, ce qui veut dire qu'il suffit de peu diminuer le seuil de précision T pour que le *recall* soit haut). En abscisse, il y a le taux de faux positifs et en ordonnée le taux de vrai positifs (le *recall* de la classe étudiée).

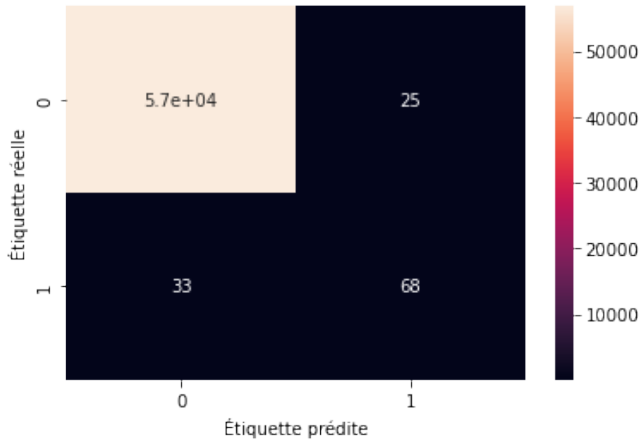
Dans le cas des données de la CCFD, nous obtenons la courbe ROC suivante :



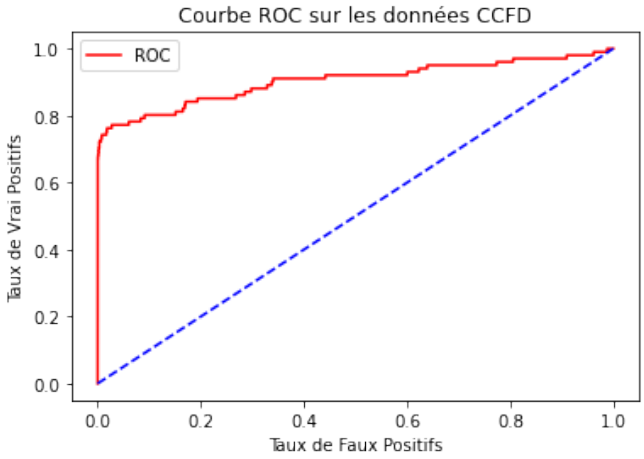
La courbe ROC est relativement proche de la première bissectrice donc il est nécessaire de diminuer le seuil pour améliorer la qualité du prédicteur.
Le AU ROC correspond simplement à l'air sous la courbe ROC, qui résume cette courbe. Il tend vers 1 pour le meilleur cas et vers 0.5 au pire des cas. Dans notre cas, nous avons un AU ROC de 0.565 plutôt proche de 0.5 ce qui corrobore le fait que le modèle n'est pas très performant.

Le modèle de prédiction n'est donc pas suffisemment efficace, et des méthodes pour rééquilibrer l'échantillon seront donc proposées dans la prochaine section.

En changeant le solver de la régression logistique en lbfgs voici la matrice de confusion que nous trouvons :



Et voici la courbe ROC associée avec un AU ROC de 0.9032.



Cette fois-ci, le modèle ne prédit pas tous les élément comme appartenant à la classe 0 et le AU ROC est supérieure au modèle de régression logistique avec solver sag donc le modèle de régression logistique avec solver lbfgs est meilleure. Nous verrons par la suite si les méthodes de re-sampling permettent d'améliorer le AU ROC.

2.3 Analyse mathématique du déséquilibre

Expliquons ce qui se passe dans les modèles de classification binaire lorsque les données sont très déséquilibrées. Soit $\mathcal{X}$ l'ensemble des features. Considérons l'éstimateur bayésien du problème de classification à 2 features où $y \in \mathcal{Y} = \{0, 1\}$:

$$f^*(x) = \begin{cases} 0 & \text{si } P(y = 0|X = x) \geq 0.5 \\ 1 & \text{sinon.} \end{cases}$$

Supposons que la classe 0 soit plus représentée que la classe 1 dans le sens où pour tout $x \in \mathcal{X}$:

$$P(X = x, Y = 0) \geq P(X = x, Y = 1) \tag{1}$$

Ceci équivaut à ce que pour tout $x \in \mathcal{X}$:

$$P(X = x|Y = 0)P(Y = 0) \geq P(X = x|Y = 1)P(Y = 1) \tag{2}$$

Ce qui implique :

$$\begin{aligned} P(Y = 0|X = x) &= \frac{P(Y = 0, X = x)}{P(X = x)} \\ &= \frac{P(X = x|Y = 0)P(Y = 0)}{P(X = x)} \\ &\geq \frac{P(X = x|Y = 1)P(Y = 1)}{P(X = x)} \\ &= P(Y = 1|X = x) \end{aligned}$$

En d'autre termes, on a que $P(y = 0|X = x) \geq 0.5$ pour tout $x \in \mathcal{X}$ donc le prédicteur bayésien f^* prédit toujours 0.

Le but des méthodes de ré-échantillonnage que nous aborderons par la suite est donc de faire en sorte que l'inégalité (1) (ou de manière équivalente l'inégalité (2)) ne soit plus vraie pour tout x de sorte à ce que le prédicteur ne renvoie pas toujours la même valeur et puisse renvoyer le label minoritaire. Nous allons faire ceci en ré-équilibrant l'échantillon d'entraînement.

3 Méthodes de ré-échantillonnage

Les méthodes présentées ci-dessous vont permettre de rééquilibrer les données pour obtenir une meilleure prédiction. Les méthodes d'*oversampling*, d'*undersampling* et de *bagging* seront étudiées. (Badr (2019), Rouvière (2006))

Nous allons mettre en œuvre la méthode ré-échantillonnage qui consiste essentiellement à rééquilibrer nos données afin d'obtenir un ensemble de données plus équilibré et ainsi éviter que notre modèle fasse du sur-apprentissage. Tout d'abord, la première étape consiste à déterminer à quel point notre jeu de données est déséquilibré, c'est-à-dire combien il y a de transactions frauduleuses et non-frauduleuses. Ensuite nous devons faire en sorte d'équilibrer les transactions frauduleuses jusqu'à un certain niveau par rapport aux transactions non frauduleuses. Pour chacune des méthodes que nous allons utiliser ce niveau de rééquilibrage des données peut être optimisé pour avoir le meilleur résultat possible. Nous avons décidé de choisir un seuil fixe de 0.5, ainsi il y aura autant de données de transactions frauduleuses que de données de transactions non frauduleuses.

Pré-traitement des données : Avant d'appliquer nos méthodes, nous devons diviser nos données en un ensemble d'entrainement et de test, et également séparer les variables explicatives de notre variable d'intêret (label). Nous appliquerons nos méthodes de resampling à notre ensemble d'entrainement uniquement.

Comment comparer nos différents méthodes de ré-échantillonnage ?

Grâce à nos différentes méthodes de ré-échantillonnage,nous disposerons de différents jeux de données rééquilibrés . Nous entrainerons 3 modèles avec nos nouveaux jeux de données :

— Le modèle de régression logistique
— Le modèle des K-plus proches voisins
— Le modèle de classsification naïve bayésienne

Nous comparerons pour chaque modèle l'efficacité de nos méthodes de ré-échantillonnage en comparant les métriques que nous avons définies dans la section précédente :le score de précision (Accuracy), le score AU ROC et le F1-Score

3.1 Méthode sans Resampling

Nous réalisons les régressions sans ré-échantionner le jeu de données. Voici les matrices de confusion que nous obtenons sans resampling.

Régression logistique

	étiquette prédicte(0)	étiquette prédite(1)
étiquette réelle (0)	56834	27
étiquette réelle (1)	36	65

K-plus proches voisins (K=3 optimisé)

	étiquette prédicte(0)	étiquette prédite(1)
étiquette réelle (0)	56859	2
étiquette réelle (1)	93	8

Classification Naive bayésienne

	étiquette prédicte(0)	étiquette prédite(1)
étiquette réelle (0)	56479	382
étiquette réelle (1)	39	62

3.2 Méthode Oversampling

Dans cette partie, nous allons distinguer deux méthodes de sur-échantillonnage. Dans un premier temps nous parlerons de la méthode Random Oversampling qui est une une méthode de resampling qui utilise les variables de notre échantillon et l'algorithme SMOTE qui sur-échantillonne à partir de données synthétiques.

3.2.1 Random Oversampling(ROS)

Cette technique consiste à tirer aléatoirement et de façon uniforme des points de la catégorie minoritaire, puis à les dupliquer. L'inconvénient majeur de cette approche est qu'elle peut conduire à un risque de sur-apprentissage du modèle, puisqu'elle génère des répliques identiques des éléments de la classe minoritaire, leur donnant artificiellement trop de poids.
Voici les matrices de confusion que nous obtenons avec random oversampling.

Régression logistique

	étiquette prédicte(0)	étiquette prédite(1)
étiquette réelle (0)	54936	1925
étiquette réelle (1)	5	96

K-plus proches voisins(K optimal=1)

	étiquette prédicte(0)	étiquette prédite(1)
étiquette réelle (0)	56861	0
étiquette réelle (1)	0	101

Classification Naive bayésienne

	étiquette prédicte(0)	étiquette prédite(1)
étiquette réelle (0)	56359	502
étiquette réelle (1)	31	70

3.2.2 Algorithme SMOTE(Synthetic Minority Over-sampling Technique)

La technique SMOTE est une méthode de sur-échantillonnage qui consiste à générer des données "synthétiques".

Pour créer un individu synthétique, les étapes définies dans l'algorithme du SMOTE sont les suivantes :

— Sélectionner aléatoirement une observation minoritaire "initiale".
— Identifier ses k plus proches voisins parmi les observations minoritaires (où k est un paramètre défini par l'utilisateur).
— Choisir aléatoirement l'un des k plus proches voisins.
— Générer aléatoirement un coefficient .
— Créer un nouvel individu entre l'observation initiale et le plus proche voisin choisi, selon la valeur du coefficient .

Ces étapes sont répétées jusqu'à ce que le nombre d'individus générés atteigne une valeur définie par l'utilisateur. Les paramètres du SMOTE sont les suivants :

— k, le nombre de plus proches voisins (nearest neighbors) candidats pour la création d'un nouvel individu.
Ce paramètre a un effet sur la distribution des individus synthétiques dans l'espace, et donc sur la performance du modèle qui sera entraîné après le SMOTE. Sa valeur optimale dépend de la structure des données. Nous l'optimisons pour chaque modèle de régression.
— α, le taux d'observations minoritaires à atteindre.(nous le fixons à 0.5)

Une des limites de SMOTE est qu'elle génère les points synthétiques sans considérer les éventuels éléments de la classe majoritaire à proximité, ce qui peut occasionner des chevauchements de catégories et une sur-généralisation du classifieur.

Voici les matrices de confusion que nous obtenons avec l'algorithme SMOTE.

Régression logistique(ksmote=13 optimisé)

	étiquette prédicte(0)	étiquette prédite(1)
étiquette réelle (0)	55742	1119
étiquette réelle (1)	11	90

K-plus proches voisins(K=1 optimisé, ksmote=1 optimisé)

	étiquette prédicte(0)	étiquette prédite(1)
étiquette réelle (0)	56578	283
étiquette réelle (1)	72	29

Classification Naive bayésienne(ksmote=13 optimisé)

	étiquette prédicte(0)	étiquette prédite(1)
étiquette réelle (0)	56461	400
étiquette réelle (1)	27	74

3.3 Méthode Random Undersampling (RUS)

Cette méthode consiste à réduire le nombre d'éléments de la classe majoritaire. Nous allons retirer des éléments de la classe majoritaire uniformément jusqu'à arriver à l'équilibre souhaité (nous choisissons 50%).

Le principal problème du Random Undersampling est le risque que notre modèle de classification ne soit pas aussi précis que nous le souhaiterions. En effet on peut remarquer qu'il y a une grande perte d'informations lors de notre ré-équilibrage et cela peut impacter la précision de notre modèle (492 transactions non frauduleuses parmi 284 315 transactions non frauduleuses).

Voici les matrices de confusion que nous obtenons avec random undersampling.

Régression logistique

	étiquette prédicte(0)	étiquette prédite(1)
étiquette réelle (0)	54327	2534
étiquette réelle (1)	7	94

K-plus proches voisins(K optimal=9)

	étiquette prédicte(0)	étiquette prédite(1)
étiquette réelle (0)	39470	17391
étiquette réelle (1)	43	58

Classification Naive bayésienne

	étiquette prédicte(0)	étiquette prédite(1)
étiquette réelle (0)	55991	870
étiquette réelle (1)	32	69

3.4 Résultats

3.4.1 Régression logistique

Après le resampling de notre ensemble d'entrainement avec nos différentes méthodes, et après entrainement du modèle de régression logistique avec les jeux de données ré-échantionnés nous obtenons les résultats suivants. Pour chaque métrique, nous avons mis en jaune le score de la méthode la plus performante.

	Accuracy	AU ROC	F1-score
Sans resampling	0.998894	0.926118	0.673575
Random Undersampling	0.955391	0.97328	0.06889
Random Oversampling	0.966118	0.983799	0.090481
SMOTE	0.980162	0.968528	0.137405

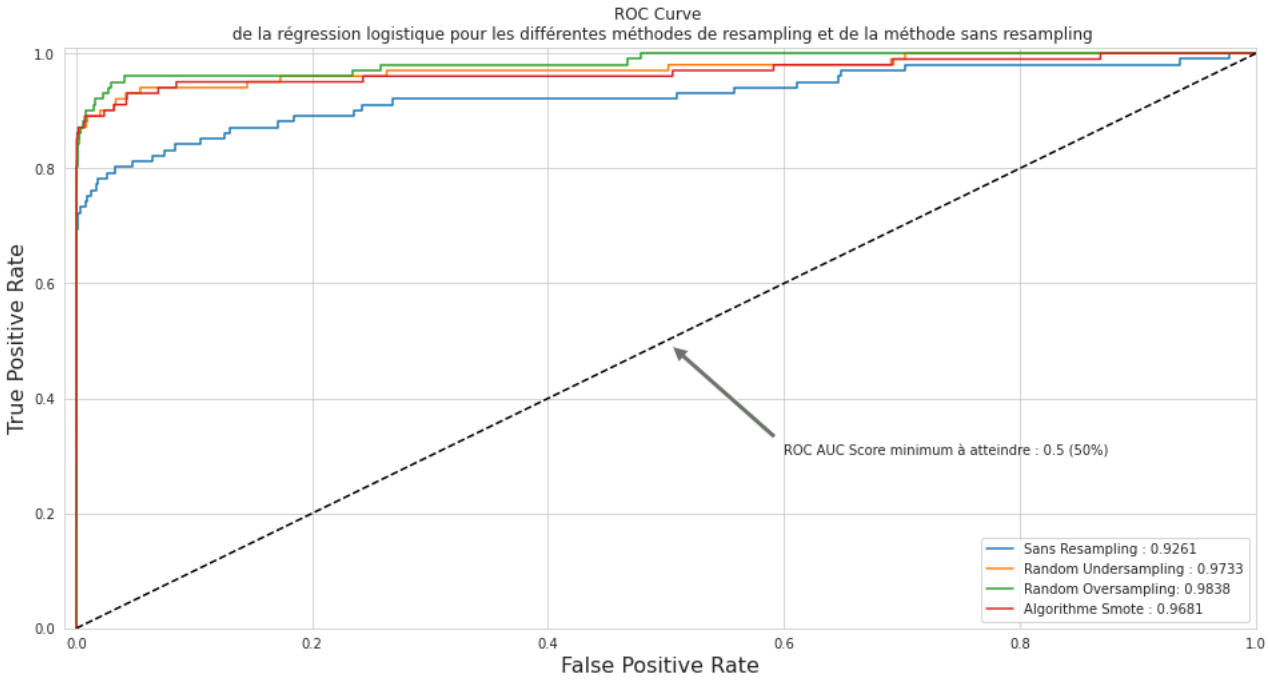


FIGURE 1 – ROC des différentes méthodes avec régression logistique

Nous pouvons constater :

— La régression sans resampling a la meilleure accuracy, cette dernière étant la métrique la plus naïve, ce résultat n'est pas étonnant.

— Toutes les méthodes de resampling ont une meilleure AU ROC que la méthode sans resampling,

— La meilleure méthode de resampling semble être le Random Oversampling pour l'AU ROC et le SMOTE pour le F1-score

— Les méthodes d'oversampling sont meilleures que la méthode d'undersampling pour les 3 métriques que nous avons choisies.

3.4.2 K-plus proches voisins

Par la classification K-NN, nous obtenons les résultats suivants :

	Accuracy	AU ROC	F1-score
Sans resampling	0.998332	0.568569	0.144144
Random Undersampling	0.693936	0.667061	0.00661
Random Oversampling	1.0	1.0	1.0
SMOTE	0.993768	0.641076	0.140436

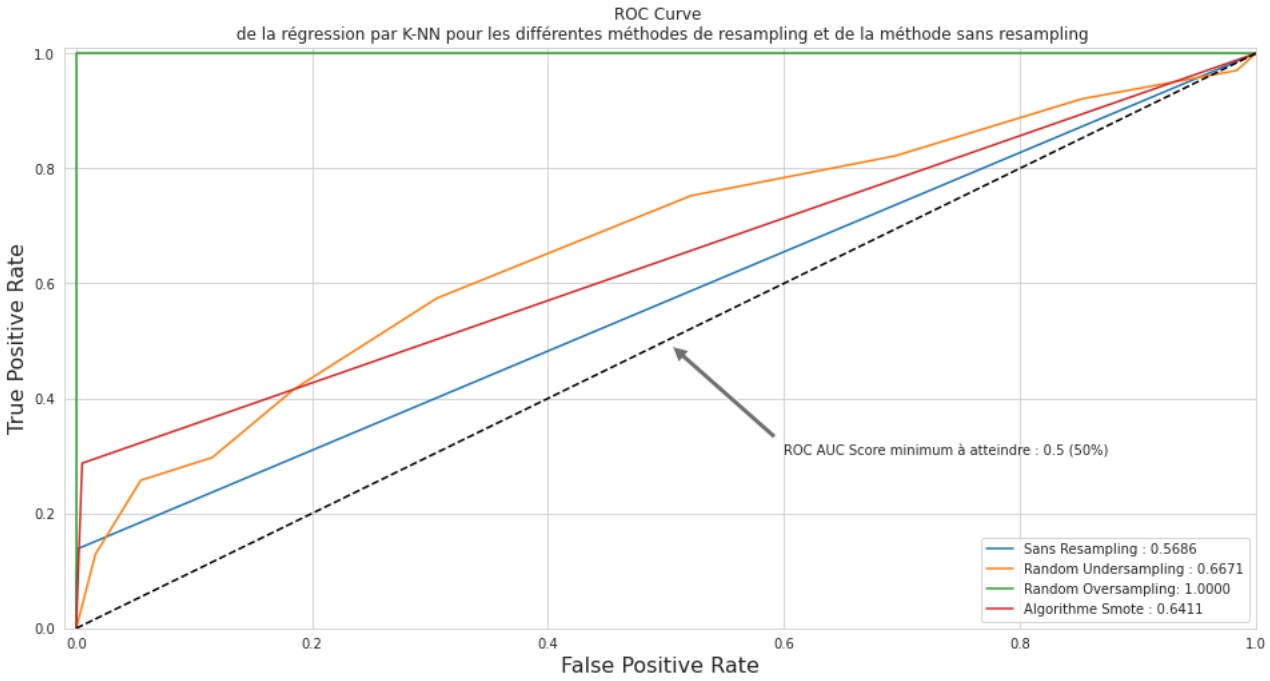


FIGURE 2 – ROC des différentes méthodes avec K-plus proches voisins

Nous pouvons constater :

— Les meilleures résultats sont données par la méthode d'oversampling avec notamment un AU ROC égal à 1 qui est très largement supérieures aux valeurs des autres méthodes

— L'écart est encore plus important en observant le F1-Score avec 100% contre seulement 14% pour la méthode sans resampling et le SMOTE et un résultat proche de 0 pour l'undersampling.

— Les résultats concernant l'Accuracy sont assez proche pour chaque méthode sauf pour la méthode d'undersampling, ce qui n'est pas étonnant car on peut observer sur notre matrice de confusion que le nombre de faux positifs est très élevé par rapport au nombre de vrais positifs comparé aux autres méthodes (39470 contre 43), impactant ainsi le taux de bonnes prédictions total.

3.4.3 Classsification naive bayésienne

Par la classification Naîve bayésienne, nous obtenons les résultats suivants :

	Accuracy	AU ROC	F1-score
Sans resampling	0.992609	0.978063	0.227523
Random Undersampling	0.984165	0.977702	0.132692
Random Oversampling	0.990643	0.977928	0.208024
SMOTE	0.992504	0.975953	0.257391

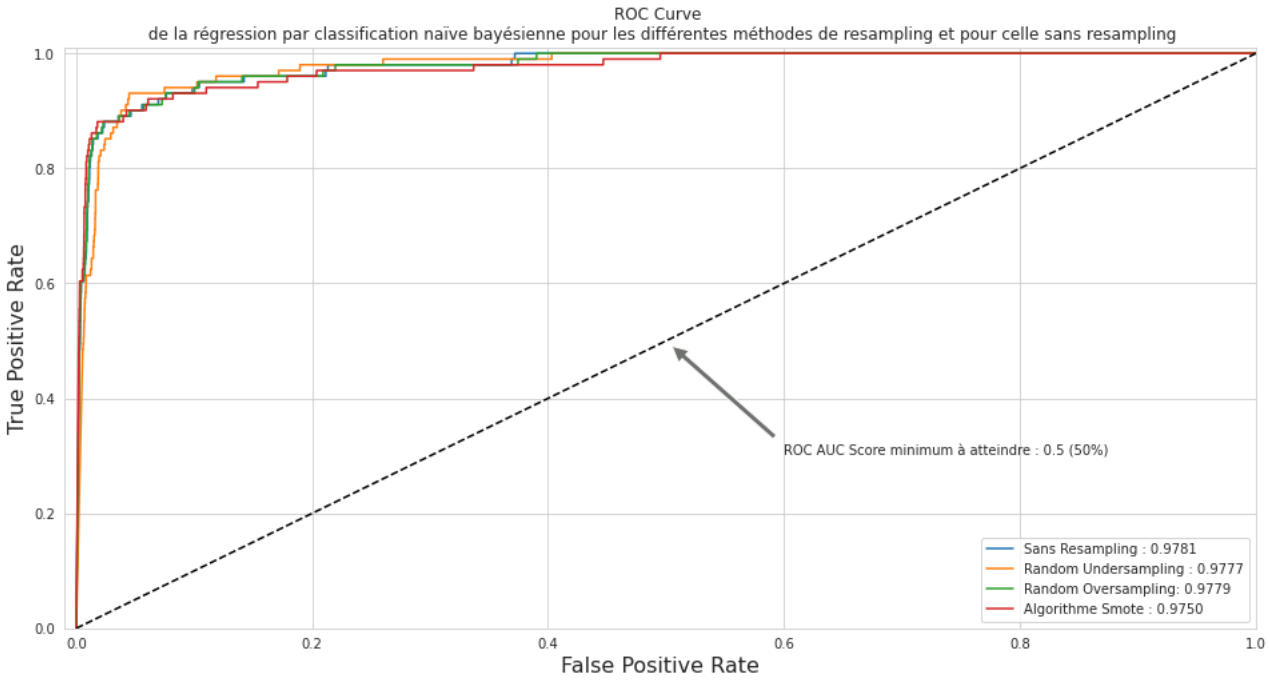


FIGURE 3 – ROC des différentes méthodes avec la classification naïve bayésienne

Nous pouvons constater :

— Les résultats sont très proches pour chaque méthode concernant l'Accuracy et l'AU ROC avec des valeurs respectivement très proches de 100% et 98%

— La différence entre ces méthodes se remarque surtout sur le F1-score, la meilleure valeur est obtenue par la méthode SMOTE avec une valeur proche de 25%. Cependant on peut ajouter que les F1-score obtenues sont assez faibles.

4 Une méthode d'assemblage : le *Bagging*

Le *bagging* est une méthode d'assemblage de données qui permet de réduire la variance du modèle et limite son sur-apprentissage. Cette méthode réalise un échantillonnage des données et entraîne l'algorithme de façon séparée sur chacun de ces échantillons. Les résultats des modèles obtenus sont alors assemblés. Ceci permettra une meilleure prédiction finale, car elle prendra en considération l'ensemble des modèles entraînés. Cette méthode s'utilise en régression comme en classification.
Un des atouts de cette méthode est de pouvoir réduire la variance. En effet même si les modèles ne sont pas entraînés sur le même jeu de donnée, les échantillons de bootstrap partagent des tuples en commun, ce qui produit du biais. Ce biais produit fait réduire la variance.
Le principe du bootstrap est de créer de nouveaux échantillons par tirage au hasard dans l'ancien échantillon, avec remise. L'algorithme est entraîné sur ces sous-ensembles de données. Les estimateurs ainsi obtenus sont moyennés (lorsque les données sont quantitatives, cas d'un arbre de régression) ou utilisés pour un "vote" à la majorité (pour des données qualitatives, cas d'un arbre de classification). C'est la combinaison de ces multiples estimateurs indépendants qui permet de réduire la variance. Toutefois, chaque estimateur est entrainé avec moins de données.
En pratique, la méthode de bagging devrait donner d'excellents résultats.

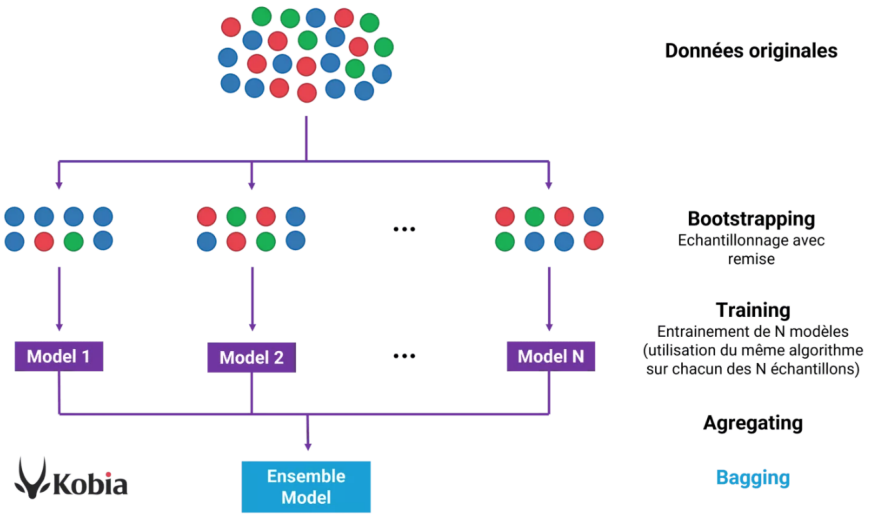


FIGURE 4 – Illustration de la méthode de bagging

5 Conclusion

En conclusion, nous avons donc défini et interprété les différentes métriques telles que l'AU ROC, le F1-Score et l'Accuracy, puis nous avons utilisé des méthodes de ré-échantillonage afin de régler le déséquilibre de nos données. Nous avons ensuite étudié 3 modèles prédictifs : la régression logistique, K-NN et la classification naïve bayésienne.

Pour chaque modèle, nous avons explicité la matrice de confusion et comparer les différentes métriques obtenues par chaque méthode de ré-échantillonage de données.

Pour chaque modèle de classification, il s'avère que les méthode de random Oversampling et Smote sont au dessus par rapport à la méthode de random Undersampling. Nous avons essayé de terminer notre étude de nos données par une méthode d'assemblage qu'est le bagging mais nous n'avons pas réussi à obtenir de résultats dû à des difficultés sur le code.

Références

Anonyme (s. d.). Credit Card Fraud Detection. url : https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud.

Badr, W. (2019). How to fix an Unbalanced Dataset. url : https://www.kdnuggets.com/2019/05/fix-unbalanced-dataset.html.

Rocca, B. (2019). Handling imbalanced datasets in machine learning. url : https://towardsdatascience.com/handling-imbalanced-datasets-in-machine-learning-7a0e84220f28.

Rouvière, L. (2006). Données déséquilibrées. url : https://lrouviere.github.io/TUTO_ML/dondes.html.