AUGMENTATION DE DONNEES TABULAIRES POUR RESOUDRE UN DESEQUILIBRE DES CLASSES

MOTS CLES : DEEP LEARNING - AUGMENTATION DE DONNEES - CLASSIFICATION AUTOENCODEURS VARIATIONNELS - DESEQUILIBRE DES CLASSES

Dans une tâche de classification traitée par apprentissage statistique, le déséquilibre de classes est un problème récurrent, notamment en assurance. Les consultants de Galea experts en data science ont régulièrement l'habitude de faire face à ce type de problématique. Irchad MAMODE VALJEE, actuaire sénior, membre du DataLab Galea et titulaire du Certificat d'Études Spécialisées *Intelligence artificielle* de Télécom Paris, vous propose une méthode d'augmentation des données via l'utilisation du *Deep Learning* permettant de pallier le manque de données d'une classe minoritaire.

LE DÉSÉQUILIBRE DE CLASSE EN APPRENTISSAGE STATISTIQUE

Dans un problème d'apprentissage supervisé pour une tâche de classification, l'on parle de déséquilibre des classes ou « *Class Skew* » lorsque la répartition entre les classes n'est pas équilibrée. Il est commun de rencontrer cette problématique en assurance dans le cadre de prédiction de résiliations, de fraude ou encore de consommation sur certains risques. Dès lors que ce déséquilibre est trop prononcé, les algorithmes, au regard des métriques d'apprentissage, peinent inévitablement à produire des résultats satisfaisants.

L'intuition naturelle pour résoudre cette problématique est d'ajouter plus de données à la classe minoritaire pour générer un équilibre entre les classes. Cependant, il peut être difficile d'obtenir plus de données selon la nature du problème traité. Pour cette raison, diverses alternatives ont été proposées pour résoudre le problème du déséquilibre des classes dans le cas de données tabulaires. Elles reposent sur deux grands axes d'idées : le rééchantillonnage (sur-échantillonnage ou sous-échantillonnage) et la génération de données.

Le rééchantillonnage est la technique la plus simple pour résoudre le problème d'asymétrie de classe dans les données tabulaires. Cependant, cette méthode n'est pas toujours adaptée à la problématique selon la nature des données. En effet, le sous-échantillonnage consiste à supprimer de manière aléatoire des données de la classe majoritaire, ce qui peut réduire la variété des données tandis que le sur-échantillonnage permet d'échantillonner autant de fois que nécessaire la classe minoritaire, ce qui peut au contraire créer de la redondance. L'objectif de cette note est donc de développer la seconde alternative au déséquilibre des classes : la génération de données.

QUELQUES MÉTHODES USUELLES DE GÉNÉRATION DE DONNÉES

L'approche purement statistique résout le problème de génération de données en ajustant une distribution de probabilité sur les données réelles afin de tirer des échantillons via cette distribution selon les besoins de données. Cette première approche est rapidement limitée par la connaissance a priori de la distribution sous-jacente de données complexes en grande dimension.

Une seconde approche basée sur des techniques statistiques et, notamment de Machine Learning intitulée SMOTE (*Synthetic Minority Over-sampling Technique*) permet de sur-échantiller des données issues des classes minoritaires en créant des données synthétiques qui ne sont pas des copies de données existantes. L'idée générale de SMOTE est la génération de données basé sur l'algorithme des « k » plus proches voisins. Bien que SMOTE soit une technique permettant de générer des données tabulaires, un tel algorithme présente certaines

limites. Les données générées sont linéairement dépendantes et l'hypothèse forte qui suppose qu'une donnée générée à partir des voisins d'un point d'une certaine classe reste dans la même classe peut être remise en cause au niveau de la frontière entre deux classes. De nombreuses alternatives basées sur SMOTE ont été proposées afin de combler les limites de la technique originale telles que Borderline SMOTE ou ADASYN.

Une approche plus récente consiste à utiliser des algorithmes génératifs afin de résoudre le déséquilibre des classes. Il existe deux grandes familles de modèles génératifs, les GANs (*Generative Adversarial Networks*) et les VAEs (*Variational AutoEncoders*). Ces algorithmes sont aujourd'hui dominants grâce à une qualité supérieure des données générées.

LES AUTOENCODEURS COMME MODÈLE GÉNÉRATIF

ARCHITECTURE D'UN AUTOENCODEUR « VANILLE »

L'architecture d'un autoencodeur dit « vanille » considéré comme la version de base des modèles d'autoencodeurs, est composée des éléments suivants :

- L'encodeur E : un premier réseau de neurones permettant de condenser l'information disponible en entrée par l'extraction de *features* qui caractérisent du mieux possible l'information initiale. Le vecteur qui résulte de l'encodeur est de dimension bien inférieure au vecteur initial. Les données sont ainsi présentes sous une nouvelle forme dans un espace appelé espace latent.
- Le décodeur D : un second réseau de neurones, chargé de « reconstruire » l'information de départ à partir du vecteur condensé.

L'architecture d'un autoencodeur vanille est schématisée dans la figure 1. L'encodeur produit à partir d'une entré x une représentation compacte $z = E(x)$ dans l'espace latent de dimension réduite. z est ensuite utilisée comme variable d'entrée par le décodeur pour générer une sortie $x' = D(z)$

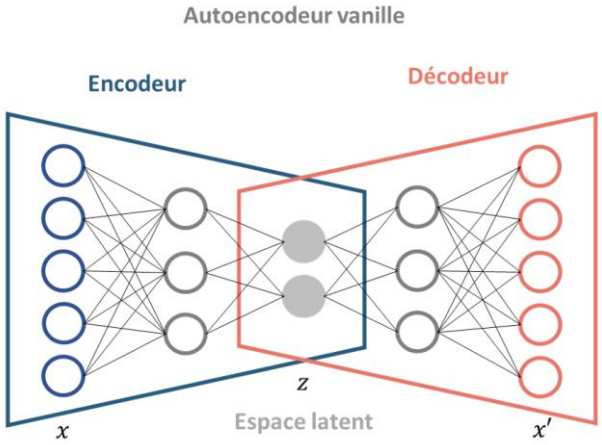

Figure 1 : Description de l'architecture d'un autoencodeur vanille

L'objectif de l'autoencodeur est d'obtenir une sortie x' la plus proche possible de l'entrée x. L'optimisation du modèle est donc la recherche des paramètres des réseaux de neurones E et D minimisant l'écart entre x et x' d'où le choix de la fonction de coût de reconstruction suivante :

$$L_{rec} = \|x - x'\|^2$$

En minimisant les erreurs de reconstruction, l'autoencodeur apprend ainsi une bonne représentation des données.

ILLUSTRATION DE L'UTILITÉ D'UN ENCODAGE DANS UN ESPACE LATENT

L'objectif de cette section est de créer un modèle d'autoencodeur simple afin de développer une intuition sur l'importance de l'espace latent dans l'architecture du modèle. L'idée est d'analyser de manière détaillée l'encodage des données dans cet espace compressé à travers un exemple pratique. Le jeu de données Kaggle utilisé pour cette application contient les transactions

effectuées par carte de crédit par des titulaires de cartes européennes[1].

La détection de fraudes est un sujet courant dans plusieurs domaines et notamment dans le domaine assurantiel. La méthode traditionnelle de détection de fraude est d'appliquer les règles de gestion pour signaler les transactions frauduleuses. Cette approche devient inefficace quand les fraudeurs réussissent à contourner les règles de gestion. Les modèles de *Machine Learning / Deep Learning* peuvent être appliqués sur les données historiques, lorsqu'elles sont disponibles en quantité suffisante, afin d'apprendre à reconnaitre les données frauduleuses parmi un ensemble de données.

D'un point de vue *machine learning,* la détection de fraude est une tâche de classification binaire classique. La difficulté majeure réside dans le déséquilibre du fait que la grande majorité des transactions sont non frauduleuses.

La base de données contient 492 transactions de fraudes sur 284 807 transactions au total. L'ensemble de données est ainsi très largement déséquilibré puisque la classe positive (fraudes) ne représente que 0.172 % des transactions. Le plus grand défi est donc la résolution du déséquilibre des classes.

Pour visualiser la distribution des données, la technique de représentation T-SNE est choisie. T-SNE (*T-Distributed Stochastic Neighbor Embedding*) est une technique de projection des données dans un espace de dimension plus faible en n'affichant que les composantes avec un maximum d'informations

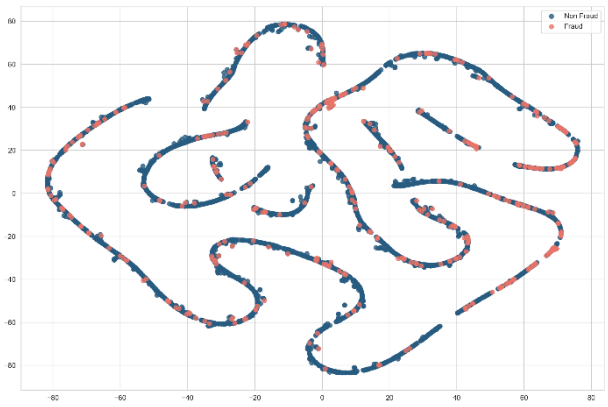

Figure 2 : Visualisation des données via la technique T-SNE

La figure 2 illustre le fort "chevauchement" des ensembles des 2 classes de données dans une forme de spirale. Il apparait donc complexe de séparer les données frauduleuses à l'aide d'une frontière de décision simple. Ce constat révèle que la représentation propre des données frauduleuses est difficile à capter. De manière générale, il est à noter que même avec un niveau de déséquilibre acceptable, le degré de chevauchement des données impacte fortement la capacité à distinguer chaque classe.

Pour cette première application, l'entrainement du modèle se fait uniquement sur les données non frauduleuses avec l'intuition selon laquelle une classe différente de celle apprise par l'encodeur se différenciera dans l'espace latent.

Après avoir entrainé un modèle d'autoencodeur sur les données non frauduleuses, l'objectif est de l'appliquer sur l'ensemble des données afin de comprendre ce qu'il se passe dans les couches d'encodage et plus spécifiquement les différences d'encodage entre les données normales et les données frauduleuses dans l'espace latent. Pour cela, un autre modèle séquentiel contenant uniquement les couches d'encodage est créé.

La visualisation des représentations codées dans la figure 3 permet de démontrer qu'au niveau de l'espace latent, les transactions frauduleuses et non frauduleuses sont distinguables et quasiment linéairement séparables. La

[1] https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud

condensation des données aux informations les plus importantes au sein de l'espace latent permet donc de réduire de manière très nette le degré de chevauchement des deux classes et justifie ainsi l'utilisation d'une telle architecture.

A ce stade, il convient de rappeler que les autoencodeurs sont des modèles génératifs dont l'objectif est de produire des échantillons réalistes de données : comment ce type d'architectures peut permettre de produire des exemples aléatoires de données ?

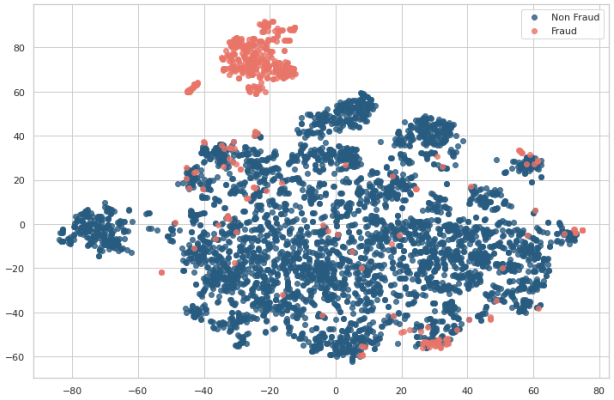

Figure 3 : Visualisation des données dans l'espace latent

ARCHITECTURE D'UN AUTOENCODEUR VARIATIONNEL

Générer des données à partir de l'espace latent afin de les décoder par la suite semble complexe à partir d'un autoencodeur simple qui tire les points de manière uniforme dans l'espace lattent. Pour cette tâche, il est nécessaire de faire appel aux autoencodeurs dit « variationnels » qui permettent d'imposer à l'encodeur de faire atterrir les données d'entrée encodées selon une structure précise (gaussienne par exemple). La modélisation de données synthétique consistera alors à :

// Échantillonner dans l'espace latent

// Décoder pour produire des données aléatoires

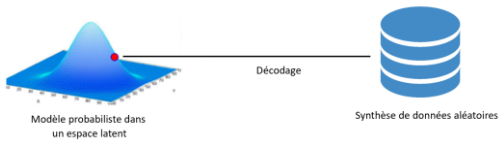

Figure 4 : Apport d'un autoencodeur variationnel

Dans un autoencodeur variationnel, dont l'architecture est représentée dans la figure 5, l'espace latent z n'est plus décrit par un vecteur fixe. Au contraire, chaque dimension d de z est représentée par une distribution de probabilité.

Pour ce faire, une loi a priori est choisie pour structurer l'espace latent, les couches en sortie de l'encodeur permettent ensuite l'apprentissage des paramètres de la loi choisie en amont. Dans la majorité des cas et dans un souci de simplicité, la loi gaussienne est retenue. Ainsi, un VAE dont l'espace latent est de structure normale sera constituée de 2 couches supplémentaires en sortie de l'encodeur pour l'apprentissage des paramètres μ et σ associés à chaque entrée x. Un échantillonnage aléatoire d'une distribution normale centrée réduite $\varepsilon \backsim \mathcal{N}(0, I)$ est ensuite généré puis mis à l'échelle en multipliant par la variance σ (le poids) de la distribution de l'espace latent et décalé via le vecteur moyenne μ (le biais). La représentation compacte z est déduite par l'équation suivante où $\odot$ est le produit matriciel de Hadamard.

$$z = \mu + \sigma \odot \varepsilon$$

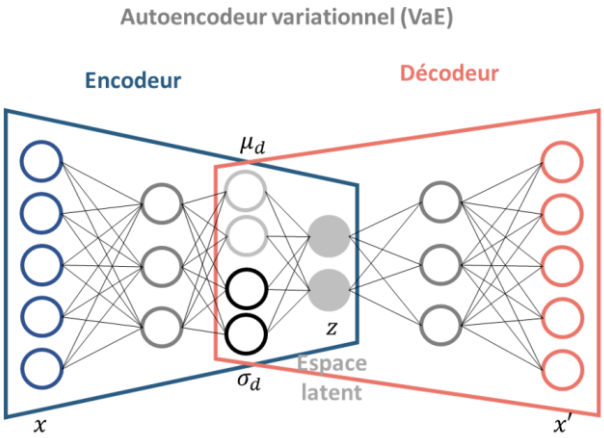

Figure 5 : Description de l'architecture d'un autoencodeur variationnel

Chaque dimension d de z est donc représentée par une distribution normale $\mathcal{N}(\mu_d, \sigma_d^2)$.

Le choix de la fonction de perte suivante reflète un objectif double de l'autoencodeur variationnel :

$$L = L_{rec} + L_{kl}$$

Avec :

$$L_{rec} = \|x - x'\|^2$$

$$L_{kl} = \frac{1}{2}\sum_d (\mu_d^2 + \sigma_d^2 - \log(\sigma_d^2) - 1)$$

- Minimiser les erreurs de la reconstruction (la sortie x' devrait être identique à l'entrée x) via la composante L_{rec}
- Imposer à l'espace latent d'avoir une distribution normale via la minimisation de la composante L_{kl} dans la fonction de coût qui est le terme de divergence de Kullback-Leibler souvent utilisée pour la comparaison entre deux distributions.

Pour finir, afin de sur-échantillonner le dataset en données « minoritaires », il suffit de générer des éléments aléatoires issus de la distribution normale de l'espace latent et dont les paramètres ont été appris auparavant via un modèle VAE entrainé sur la classe minoritaire.

APPLICATION : MISE EN PLACE D'UN VAE POUR LA GÉNÉRATION DE DONNÉES FRAUDULEUSES

Pour illustrer la difficulté des algorithmes de Machine Learning à gérer les données déséquilibrées, le modèle d'apprentissage supervisé choisi pour cette étude est le *Random Forest Classifier*. Le modèle apprend des transactions préalablement étiquetées en « frauduleux » ou « normal ». Les paramètres du modèle sont ajustés de façon à minimiser l'écart entre les labels prédits et la vérité terrain.

L'objectif est de quantifier le pouvoir du modèle à détecter les données frauduleuses dans une base de test avec une base d'apprentissage fortement déséquilibrée.

Random Forest
entrainé sans augmentation de données

	0 (non frauduleux)	1 (frauduleux)
train	*199 011*	*353*
test	*85 304*	*139*

Après l'entrainement du modèle sur la base de *train*, la prédiction des classes est faite sur les données de *test*.

La précision et le rappel (*recall*) sont les deux principales métriques pour évaluer la performance des modèles de classification binaire. La précision quantifie le taux de prédictions correctes parmi les prédictions positives et le rappel le taux d'individus positifs détectés par le modèle.

Dans le cadre de l'étude de la détection de fraudes, le *recall* est l'indicateur le plus pertinent à analyser, l'objectif étant de détecter le maximum de données réellement frauduleuses.

En se plaçant à un seuil de classification fixé à 50%, la matrice de confusion suivante est alors obtenue :

		Classe estimée	
		0	1
Classe réelle	0	*85 298*	*6*
	1	*107*	*32*

Sur 139 données de fraudes dans la base de test, uniquement 32 ont été détectées par le modèle, soit un score rappel de 23%, et donc 77% de faux négatifs. La correction du déséquilibre des classes semble donc indispensable pour améliorer la fiabilité des prédictions.

Pour ce faire, un autoencodeur variationnel est entraîné sur les données frauduleuses de la base d'entrainement puis utilisé pour générer aléatoirement 190 000 échantillons de données frauduleuses afin de rétablir un équilibre dans le jeu de données d'entrainement (la base de test, quant à elle, n'est pas modifiée).

Ces données « augmentées » sont ensuite utilisées pour l'entrainement d'un nouveau modèle *Random Forest* (avec les mêmes paramètres).

Random Forest (iso hyperparamètres) entrainé avec augmentation de données

	0 (non frauduleux)	1 (frauduleux)
train	*199 011*	***190 035***
test	*85 304*	*139*

Il est ainsi possible d'obtenir la matrice de confusion associé au même seuil de classification que précédemment (50%) :

		Classe estimée	
		0	1
Classe réelle	0	*85 239*	*65*
	1	*11*	***128***

Sur 139 données de fraudes dans la base de test, 128 ont été détectées par le modèle, soit un recall de 92% contre 23% pour le modèle entrainé sur les données déséquilibrées.

Le taux de faux négatifs a également largement diminué et passe de 77% à 8%, cependant cette méthode entraine une augmentation du taux de faux positifs (47% contre 4%) et donc une perte de précision. Ces conclusions sont logiques car en augmentant artificiellement les données de fraudes, la probabilité d'avoir de faux positifs augmente tandis que la probabilité d'avoir de faux négatifs diminue. Au regard du contexte et de la problématique de l'étude, obtenir plus de faux positifs est bien moins couteux (coût d'un contrôle uniquement) qu'une absence de détection de vrais positifs. Cette réflexion montre l'importance du contexte quant au calibrage du compromis entre précision et rappel pour un modèle.

Le score de rappel étant l'indicateur d'intérêt dans le cadre de cette étude, il est présenté dans la figure 6 les courbes de rappel pour l'ensemble des seuils de classification pour les 2 classifieurs présentés précédemment. Il apparait que, quel que soit le seuil de classification choisi, le rappel est bien meilleur avec la technique d'augmentation de données.

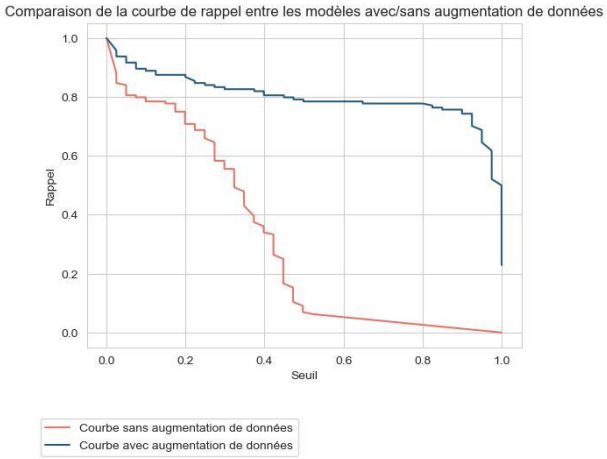

Figure 6 : comparaison des courbes de rappel des classifieurs entrainés avec les données brutes (en orange) et augmentées (en bleu)

Par ailleurs, un autre axe de comparaison consiste à confronter les scores de recall des modèles entrainés avec et sans augmentation de données calculés à nombre de prédictions positives équivalent. Cette approche permet de quantifier l'amélioration de score de rappel pour un nombre de prédictions de fraudes donné.

Ainsi, en considérant un seuil de classification fixé à 50%, le modèle entrainé sur des données augmentées permet d'établir 193 prédictions positives, pour un score de rappel de 92% (cf. ci-dessus). En considérant désormais le modèle sans augmentation de données et le seuil de classification permettant la prédiction de 193 classes fraudes (soit 17%, dont la matrice de confusion est présentée ci-dessous), il apparait un score de rappel de 56% (vs 23% avec un seuil de classification de 50%). L'augmentation de données a ainsi engendré une amélioration du rappel de 36% sur une liste de 193 fraudes potentielles qui aurait été fournie aux équipes en charge de la problématique.

		Classe estimée	
		0	1
Classe réelle	0	*85 189*	*115*
	1	*61*	*78*

CONCLUSION

Pour conclure, la motivation de cette étude peut se résumer en une phrase : « *l'ingrédient le plus important pour qu'un algorithme de Machine Learning puisse fonctionner de manière optimale est la compréhension des données* ». Comme illustré dans le cas pratique de la fraude, une mauvaise répartition des classes dans les données peut affecter sérieusement les performances des modèles ayant de fortes difficultés à bien modéliser les données déséquilibrées.

Cette étude propose une méthode d'augmentation de données en utilisant les VAEs afin de résoudre un déséquilibre des classes dans les données et de permettre une amélioration marquée du score d'intérêt. Cependant, il est important de rappeler qu'aucune approche ne fonctionne sur tout type de données déséquilibrées. Une bonne compréhension de la base de données et de la problématique en jeu sont indispensables pour orienter le choix de modélisation à mettre en œuvre.

Les consultants de GALEA sont à votre disposition pour plus de précisions sur le contenu de cette note et pour vous accompagner dans la mise en place ou la revue de vos modèles de Machine Learning, et notamment pour l'amélioration de l'apprentissage en présence de données déséquilibrées.

https://www.galea-associes.eu/

//galea