**Rapport méthodologique**

***Sommaire :***

1. Présentation du projet
2. Modélisation
3. Sélection des modèles
4. Utilisation des métriques
5. Utilisation de la fonction coût-métier
6. Rééquilibrage de nos données
7. Analyse des résultats obtenus
8. Interprétabilité globale et locale du modèle
9. Les limites et les améliorations possibles
10. Analyse du datadrift

III. Conclusion

1. Présentation du projet

Nous sommes sollicités par l’entreprise ‘Prêt à dépenser’ dans le cadre d’un projet de scoring. Cela doit permettre de calculer la probabilité de défaut pour la clientèle de l’entreprise.

Afin de répondre à cette problématique, l’entreprise nous a fourni une base de données sur leur clientèle. Ce fichier contient de nombreuses informations essentielles pour ce projet tels que le montant des crédits mais aussi les revenus annuels.

D’autres fichiers sont fournis par l’entreprise afin de pouvoir avoir une meilleure connaissance des données (revenus par agence par exemple). Toutes ces données seront utilisées afin de pouvoir analyser et effectuer un travail sur l’ensemble des fichiers. Cette étape sera fondamentale afin de préparer nos données pour une phase de modélisation et ainsi répondre à la demande de l’entité.

Les étapes de l’analyse exploratoire auront permis de procéder à cette préparation des données à l’aide de nombreuses techniques utilisées dans la Data Science (encodage, standardisation, transformation des données, création et sélection de variables, agrégation, …) .

1. Modélisation

*1. Sélection des modèles*

La finalisation de notre fichier va nous permettre de procéder à l’étape de modélisation. Le projet consiste à effectuer une classification entre les clients avec risque de paiement et les clients sans risque, nous sommes donc sur un projet de classification.

La première étape de la modélisation sera de procéder à une sélection des différents modèles que nous allons utiliser. Cette étape sera effectuée avec une analyse des modèles de classification les plus utilisées sur le site Kaggle mais également sur Scikit Learn dans le but de bien déterminer nos modèles et de procéder à une sélection pertinente. Nous avons pris la décision de sélectionner 5 modèles (2 modèles standards et 3 modèles ensemblistes) :

* Régression Logistique,
* Dummy Classifier,
* Random Forest,
* XGBoost,
* LIGHTGBM

Afin de pouvoir analyser l’ensemble de nos variables et de conserver l’ensemble de nos tests, nous allons intégrer l’utilisation du MLFLOW. Cet outil va nous permettre d’analyser l’ensemble des résultats obtenus mais également d’avoir une meilleure connaissance des différentes métriques de nos différents modèles.

L’utilisation ne provoque aucune modification dans notre projet mais va nous permettre de parfaire notre problématique.

MLflow est une plateforme open-source conçue pour faciliter le cycle de vie de développement, de gestion et de déploiement des modèles d'apprentissage automatique (machine learning). Elle fournit un ensemble d'outils permettant aux scientifiques des données et aux ingénieurs de suivre, de reproduire, de gérer et de partager leurs expériences et résultats en matière de ML.

L'objectif principal de MLflow est de rendre le processus de développement de modèles plus transparent, reproductible et collaboratif. Il offre une architecture modulaire comprenant quatre composants principaux :

* MLflow Tracking : C'est le composant central de MLflow. Il permet de suivre et d'enregistrer les paramètres, les métriques et les artefacts associés à chaque expérience de modèle. Il vous permet de garder une trace des différentes versions de modèles, des hyperparamètres utilisés, des performances obtenues et des données d'entrée/sortie associées. Cela facilite la comparaison des modèles et l'itération pour améliorer les performances.
* MLflow Projects : Ce composant fournit une structure pour organiser et empaqueter du code, des données et des paramètres en un format reproductible. Il vous permet de définir des dépendances, d'exécuter des scripts dans différents environnements (par exemple, localement ou dans un cluster distribué) et de suivre les résultats de chaque exécution.
* MLflow Models : Ce composant permet de gérer et de déployer des modèles MLflow. Il offre des outils pour enregistrer des modèles entraînés dans différents formats standardisés, tels que TensorFlow, PyTorch ou ONNX. Il facilite également le déploiement des modèles en tant que services Web ou dans des environnements cloud.
* MLflow Registry : Il s'agit d'un registre centralisé où vous pouvez enregistrer, partager et gérer les modèles enregistrés avec MLflow. Il offre un contrôle d'accès, un suivi des versions et des métadonnées détaillées pour chaque modèle. Cela favorise la collaboration entre les membres de l'équipe travaillant sur des projets ML communs.

L'utilisation de MLflow présente plusieurs avantages. Tout d'abord, il permet d'améliorer la reproductibilité des expériences et des modèles en enregistrant les paramètres et les résultats de manière systématique. Ensuite, il facilite la comparaison et l'itération des modèles en gardant une trace des performances obtenues avec différentes configurations. De plus, il favorise la collaboration en fournissant un moyen simple de partager des expériences et des modèles avec d'autres membres de l'équipe. Enfin, MLflow facilite le déploiement des modèles en offrant des outils pour les empaqueter et les déployer dans divers environnements.

*2. Sélection des métriques*

Afin de pouvoir analyser nos résultats, il est nécessaire de pouvoir analyser nos données à l’aide des métriques. Dans un problème de classification, certaines métriques sont essentielles à notre analyse, à savoir :

1. Accuracy

Il s'agit d'une mesure de performance qui permet de mesurer la proportion d'observations correctement classées par le modèle. Elle est calculée comme le nombre d'observations correctement classées divisé par le nombre total d'observations.

1. Recall

Le recall est une mesure qui indique le pourcentage de positifs correctement prédits par notre modèle. En d'autres termes, il correspond au nombre de vrais positifs divisé par l'ensemble des positifs (vrais positifs + faux négatifs).

Cela peut être exprimé mathématiquement comme suit :

Une image contenant texte

Description générée automatiquement

1. Precision

La précision est une mesure similaire au recall, mais il est important de bien comprendre la différence entre les deux. La précision permet de déterminer le nombre de prédictions positives correctement effectuées. En d'autres termes, elle correspond au nombre de vrais positifs divisé par l'ensemble des positifs prédits (vrais positifs + faux positifs).

Cela peut être exprimé mathématiquement comme suit :

Une image contenant texte

Description générée automatiquement

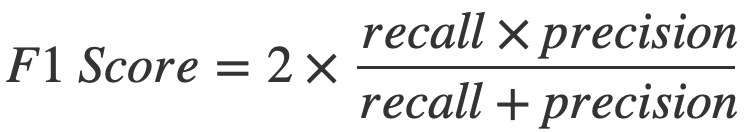
1. F1 Score

Le F1 Score permet d'effectuer une évaluation plus complète de la performance de notre modèle. Il se calcule en utilisant la formule suivante :

Le calcul du F1 Score est important car en statistique, le calcul sur les pourcentages n'est pas exactement le même que sur les nombres entiers. Le F1 Score est en fait la moyenne harmonique, un autre type de moyenne qui est idéal pour calculer la moyenne de taux ou de pourcentage (dans ce cas, la précision et le recall). Cela en fait l'une des métriques les plus utilisées par les Data Scientists.

En résumé, plus le F1 Score est élevé, meilleure est la performance de votre modèle.

Il se calcule ainsi :



1. F2 Score

Le F2 score est une métrique d'évaluation de la performance d'un modèle de classification, qui accorde plus de poids à la précision (precision) qu'au rappel (recall). Cette métrique est souvent utilisée lorsque l'accent est mis sur la minimisation des faux positifs.

La formule du F2 score est la suivante :

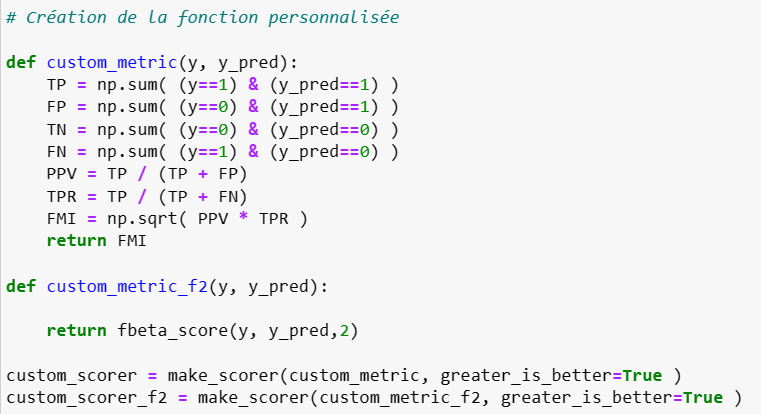
F2 = (1 + β^2) \* (precision \* recall) / (β^2 \* precision + recall)

En utilisant le F2 score, on favorise donc les modèles qui ont à la fois une bonne précision et un rappel élevé pour les exemples positifs, tout en donnant une importance légèrement plus élevée à la précision.

*3. Fonction coût métier*

Ces 4 métriques sont les plus utilisées dans une problématique de classification. Nous allons prendre la décision d’utiliser d’autres métriques mais également de mettre en place une métrique afin de rééquilibrer la différence entre le nombre de clients sans risque et avec défaut.

Pour cela, nous allons procéder à la création d’une fonction spécifique.



Cette fonction custom\_metric calcule une métrique personnalisée appelée "Fowlkes-Mallows index" (FMI) qui est utilisée pour évaluer la performance d'un modèle de classification binaire. Les variables y et y\_pred représentent respectivement les vraies étiquettes et les prédictions du modèle. Les lignes suivantes calculent les vrais positifs (TP), les faux positifs (FP), les vrais négatifs (TN) et les faux négatifs (FN) en utilisant des opérations booléennes pour comparer les étiquettes réelles et les prédictions. Ensuite, la précision positive (PPV) est calculée en divisant TP par la somme de TP et FP, et le taux de vrais positifs (TPR) est calculé en divisant TP par la somme de TP et FN. Enfin, la racine carrée du produit de PPV et TPR est retournée comme mesure de performance.

Cette fonction custom\_metric\_f2 calcule une autre métrique personnalisée appelée "F2-Measure" en utilisant la fonction fbeta\_score avec un paramètre beta de 2.0. La métrique F2-Measure accorde plus d'importance au rappel (recall) qu'à la précision, ce qui signifie qu'elle se concentre davantage sur la réduction des faux négatifs que sur les faux positifs.

Ces lignes utilisent la fonction make\_scorer pour créer des objets custom\_scorer et custom\_scorer\_f2, qui sont des métriques personnalisées pouvant être utilisées dans des fonctions d'évaluation du modèle, telles que la validation croisée. L'argument greater\_is\_better=True indique que des valeurs plus élevées de ces métriques sont meilleures.

*4. Rééquilibrage de nos données*

Après avoir procédé à une première phase de modélisation sur nos 5 modèles et ceci avec l’utilisation d’une normalisation et d’une standardisation, nous pouvons tout de suite constater les plus résultats obtenus mais également une difficulté de traitement du déséquilibre des données.

Dans le but de rééquilibrer nos données, nous allons utiliser 3 techniques, à savoir :

a. SMOTE

SMOTE (Synthetic Minority Over-sampling Technique) est une technique de sur-échantillonnage qui consiste à créer des échantillons synthétiques pour les classes minoritaires en interpolant les points de données de la classe minoritaire. Elle est souvent utilisée pour résoudre les problèmes de déséquilibre de classe dans les jeux de données en augmentant le nombre d'échantillons de la classe minoritaire.

b. RandomOverSample

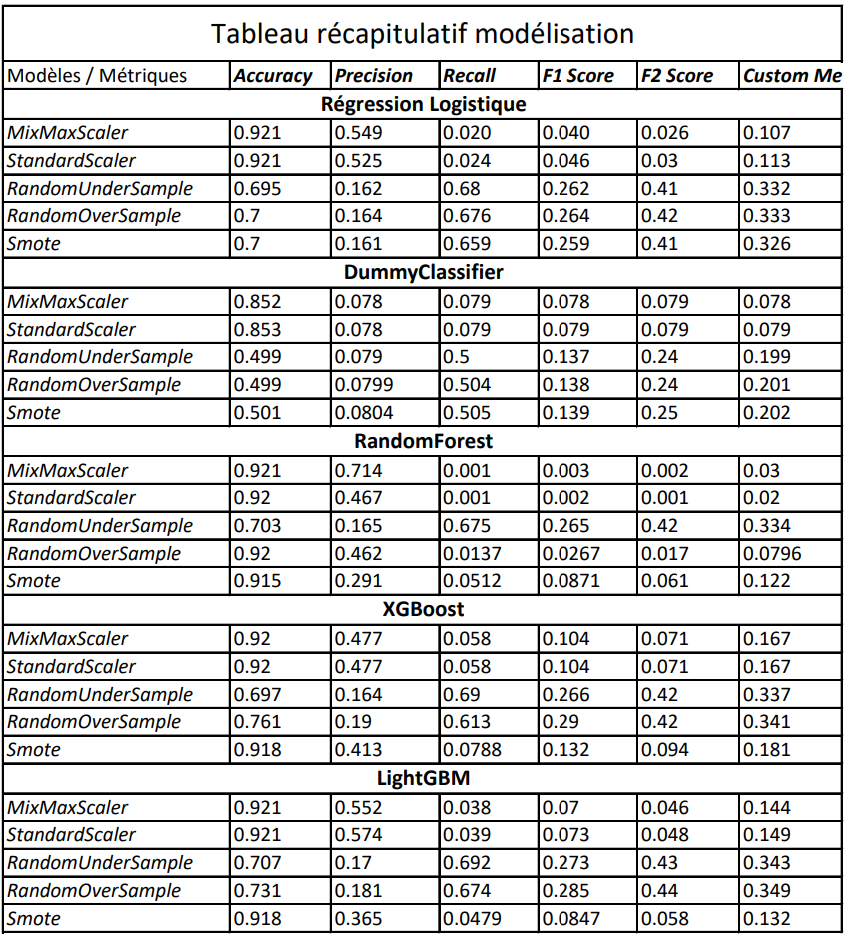
RANDOMOVERSAMPLE est une technique de suréchantillonnage qui consiste à dupliquer aléatoirement des échantillons de la classe minoritaire afin d'atteindre un nombre égal ou proche du nombre d'échantillons de la classe majoritaire. Cette méthode est simple et rapide à mettre en œuvre, mais elle comporte le risque de surapprentissage si elle est utilisée de manière excessive.

c. RandomUnderSample

D'autre part, RANDOMUNDERSAMPLE est une technique de sous-échantillonnage qui consiste à supprimer aléatoirement des échantillons de la classe majoritaire afin d'obtenir un nombre égal ou proche du nombre d'échantillons de la classe minoritaire. Cette méthode peut être efficace pour réduire le temps d'apprentissage et les coûts de traitement, mais elle peut également entraîner une perte d'informations importantes et une diminution de la précision du modèle.

Il convient de souligner que ces techniques sont souvent utilisées pour résoudre des problèmes de déséquilibre de classes dans les ensembles de données. Toutefois, il est crucial de prendre en compte les avantages et les inconvénients de chaque technique, ainsi que leur impact sur les performances du modèle, avant de choisir la méthode la plus appropriée pour un ensemble de données spécifique. Une analyse approfondie du jeu de données et une évaluation comparative des différentes approches de gestion du déséquilibre de classe sont essentielles pour prendre une décision éclairée et obtenir les meilleurs résultats possibles.

1. *Analyse des résultats obtenus*



Après avoir analysé les résultats obtenus, nous avons constaté que le StandardScaler et le MinMaxScaler produisent des résultats similaires. Nous avons donc choisi d'utiliser le StandardScaler pour le traitement du déséquilibre des données.

Parmi les différents modèles testés, la régression logistique, le XGBoost et le LightGBM ont présenté les meilleurs résultats. En revanche, le RandomForest et le Dummy classifier ont obtenu des résultats médiocres, à la fois en termes de scores et de temps d'entraînement, comme nous avons pu le constater dans MLFlow, notamment pour le RandomForest.

Le tableau présenté reflète les résultats les plus récents. Cependant, sur l'ensemble des tests effectués, le LightGBM en utilisant la technique du RandomUnderSample avec un StandardScaler a souvent obtenu les meilleurs résultats. Il est donc essentiel d'analyser attentivement les résultats obtenus, et nous avons décidé de retenir ce modèle.

Ainsi, nous concluons que le modèle à privilégier est le LightGBM avec la technique du RandomUnderSample et l'utilisation du StandardScaler. Cette combinaison a démontré une performance supérieure dans de nombreux tests effectués, et il est donc recommandé de l'adopter pour nos besoins spécifiques.

*6. Interprétabilité globale et locale du modèle*

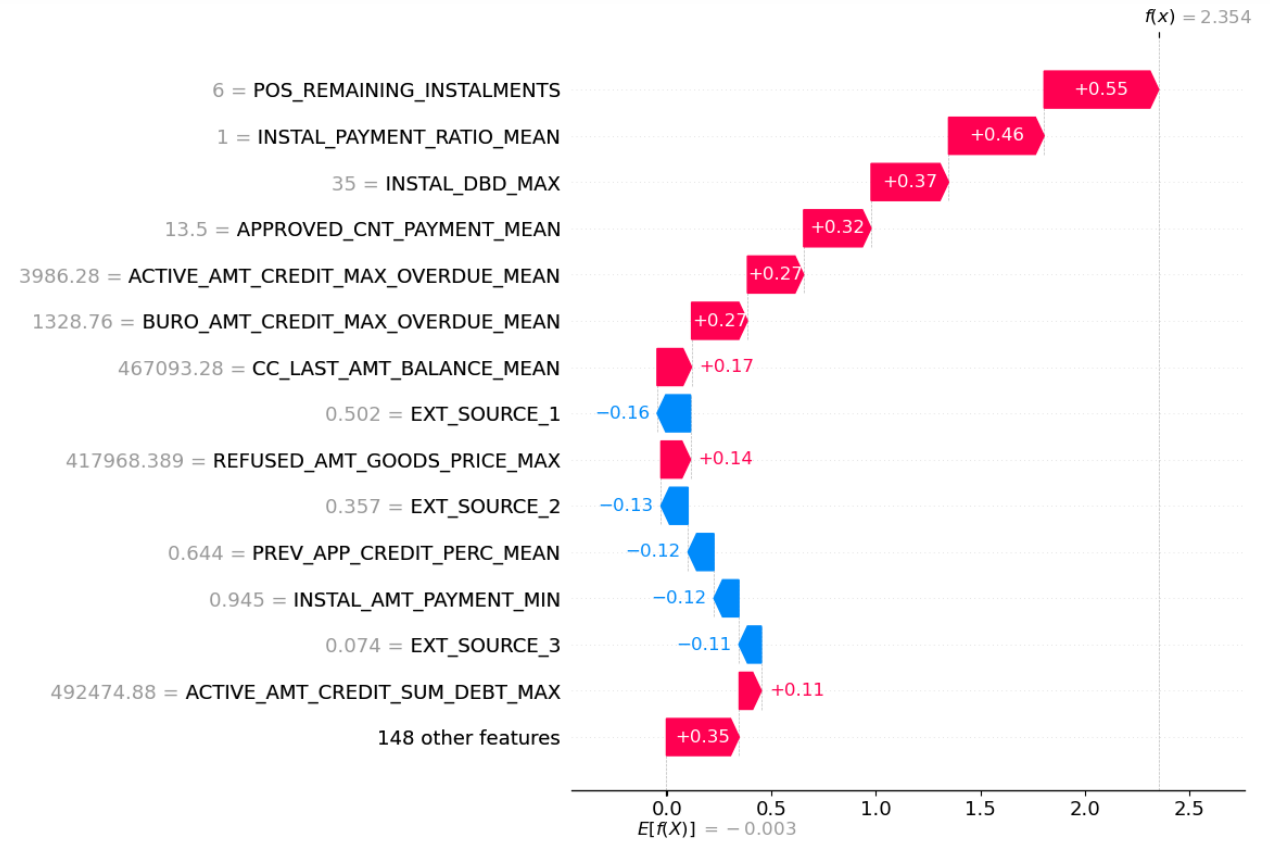
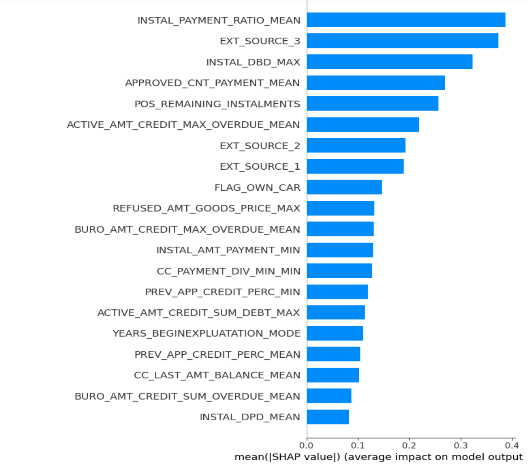
L'interprétabilité des modèles est essentielle en apprentissage automatique pour comprendre comment les décisions sont prises et quelles variables ont le plus d'influence. On peut l'analyser au niveau local (pour les prédictions individuelles) et global (sur l'ensemble des variables).

Au niveau local, on cherche à expliquer les prédictions individuelles du modèle en identifiant les caractéristiques spécifiques qui y contribuent le plus. Une approche courante est le "feature importance", qui quantifie l'influence de chaque caractéristique sur une prédiction donnée. En observant les valeurs d'importance des caractéristiques pour différentes prédictions, on peut déterminer les variables les plus pertinentes pour les décisions du modèle.

Il est cependant important de noter que l'interprétabilité peut varier en fonction de la valeur de la variable cible. Par exemple, certaines caractéristiques peuvent être plus importantes lorsque la variable cible est 0, tandis que d'autres peuvent l'être lorsque la variable cible est 1. Comprendre ces variations d'interprétabilité permet d'avoir une compréhension fine des prédictions du modèle et des facteurs qui les influencent.

L'interprétabilité peut également être examinée à l'échelle globale, en analysant l'importance des caractéristiques sur l'ensemble du jeu de données. Cette analyse globale peut révéler des différences significatives par rapport à l'interprétabilité locale. Certaines variables peuvent avoir une importance plus élevée à l'échelle globale, ce qui signifie qu'elles ont un impact plus fort sur les prédictions du modèle pour l'ensemble du jeu de données. Ces différences d'importance entre l'interprétabilité locale et globale peuvent résulter de relations complexes entre les caractéristiques et la variable cible, qui ne peuvent être pleinement capturées par une analyse locale.

En conclusion, l'interprétabilité des modèles est essentielle en apprentissage automatique. L'analyse de l'importance des caractéristiques, à la fois au niveau local et global, permet de comprendre quelles variables influencent le plus les prédictions du modèle. Toutefois, il faut prendre en compte les variations d'interprétabilité en fonction de la variable cible. Une analyse globale peut révéler des différences par rapport à l'analyse locale, mettant en évidence la complexité des relations entre les caractéristiques et la variable cible. En tenant compte de ces nuances, nous pouvons améliorer notre compréhension des modèles et prendre des décisions plus éclairées basées sur leurs prédictions.



Interprétabilité locale Interprétabilité globale

1. *Les limites et les améliorations possibles*

Lors de la modélisation, nous avons rencontré certaines limites. Nos modèles ont montré des scores prédictifs relativement faibles, malgré nos efforts d'optimisation. Ces performances peuvent être influencées par la qualité des données et les caractéristiques du problème. De plus, le déséquilibre des données, avec certaines classes sous-représentées, a également été un défi.

Pour améliorer notre approche, nous devons accorder une attention particulière à l'optimisation des modèles. Bien que nous ayons utilisé des techniques telles que la validation croisée et GridSearch CV, nous devons explorer davantage l'espace des hyperparamètres pour trouver les meilleures combinaisons. L'obtention de nouvelles données, telles que le patrimoine du client ou des cotations spécifiques, pourrait également améliorer la précision de nos modèles.

La validation croisée (cross-validation) est une méthode qui évalue les performances d'un modèle en le testant sur plusieurs sous-ensembles distincts des données. En utilisant une stratégie de "k-fold", les données sont divisées en k sous-ensembles égaux. Chaque sous-ensemble est utilisé une fois comme ensemble de test, tandis que les k-1 autres sont utilisés pour l'entraînement. Cette approche fournit une estimation plus fiable des performances en tenant compte de différentes combinaisons d'entraînement et de test.

Le GridSearch CV est une technique d'optimisation des hyperparamètres. Nous spécifions une grille de valeurs pour les hyperparamètres du modèle, puis évaluons les performances pour chaque combinaison possible. En utilisant la validation croisée, nous identifions la combinaison qui maximise les performances du modèle.

Pour la Régression Logistique, nous avons optimisé des paramètres tel que la régularisation C. Pour XGBoost et LightGBM, nous avons optimisé des hyperparamètres tels que le taux d'apprentissage, la profondeur de l'arbre, le nombre d'estimateurs, etc., en utilisant la même approche.

En conclusion, pour améliorer nos modèles, nous avons utilisé des techniques d'optimisation telles que la validation croisée et GridSearch CV. Cependant, l'amélioration des performances prédictives reste un défi complexe. D'autres axes d'amélioration, tels que l'obtention de nouvelles données et l'utilisation de techniques d'ensemble, peuvent être explorés.

Parmi les données qui auraient pu permettre une amélioration de notre modélisation, nous pouvons citer une meilleure connaissance clientèle comme son épargne ou son patrimoine (immobilier, placements, …), également une cotation client afin de mieux connaître son fonctionnement de compte et une meilleure connaissance de sa gestion.

1. *L’analyse du Datadrift*

Le data drift, ou dérive des données, est un phénomène courant dans l'apprentissage automatique où la distribution des données utilisées pour entraîner un modèle évolue au fil du temps. Cela peut entraîner une dégradation des performances du modèle, car il est optimisé pour des données passées qui ne reflètent plus la réalité actuelle. Pour détecter et gérer le data drift, différentes techniques et bibliothèques peuvent être utilisées, telles que la librairie Python "Evidently".

Cette dernière est une librairie populaire qui permet de mesurer et de surveiller le data drift dans les modèles d'apprentissage automatique. Elle fonctionne en comparant la distribution des nouvelles données d'entrée avec celle des données utilisées lors de l'entraînement du modèle. Elle peut identifier les divergences significatives entre les deux distributions et générer des alertes lorsqu'un data drift est détecté.

Un score de 0% de data drift signifie que la distribution des nouvelles données est parfaitement similaire à celle des données d'entraînement. Cela indique que le modèle n'est pas affecté par le data drift et que ses performances restent constantes dans le temps. Un score de 0% signifie que le modèle peut généraliser de manière fiable sur les nouvelles données, ce qui est essentiel pour obtenir des prédictions précises et fiables.

Avoir un data drift de 0% présente plusieurs avantages importants. Tout d'abord, cela garantit que le modèle reste performant et fiable dans des conditions changeantes. Il permet de maintenir des performances constantes dans le temps, sans nécessiter de re-entraînement fréquent du modèle. Cela permet également d'économiser des ressources, car il n'est pas nécessaire de répéter l'ensemble du processus de modélisation chaque fois que le data drift se produit.

De plus, un data drift de 0% permet de prendre des décisions basées sur des prédictions précises et actuelles. Cela est particulièrement important dans les domaines où la rapidité et l'exactitude des prédictions sont essentielles, comme la finance, la santé ou la logistique. En évitant les biais et les erreurs introduits par le data drift, les décisions prises à partir des prédictions du modèle peuvent être plus fiables et mieux alignées avec la réalité actuelle.

En résumé, le data drift se réfère à la dérive des données utilisées pour entraîner un modèle d'apprentissage automatique. Il peut être surveillé à l'aide de bibliothèques telles que DriftML. Un score de 0% de data drift indique une absence de divergence entre les nouvelles données et les données d'entraînement, assurant ainsi des performances stables et fiables du modèle. Cela offre des avantages tels que des prédictions précises, une meilleure adaptation aux conditions changeantes et des décisions plus fiables dans divers domaines d'application.

1. Conclusion

En conclusion, nous avons réussi à réaliser une implémentation de scoring pour notre modèle, ce qui constitue une étape importante dans le développement de notre solution. Cependant, il est essentiel de noter que les premiers résultats obtenus ne sont pas encore parfaits et qu'il existe encore des possibilités d'amélioration.

Nous avons mis en œuvre un ensemble de procédures pour parvenir à ces résultats, notamment en effectuant un travail approfondi sur l'ingénierie des caractéristiques (Feature engineering) afin d'optimiser les données d'entrée de notre modèle. De plus, nous avons utilisé des techniques de modélisation avancées pour créer un modèle prédictif performant.

Nous avons également intégré l'utilisation de MLFLOW, une plateforme de gestion du cycle de vie des modèles, qui nous a permis de suivre et d'enregistrer les différentes versions de notre modèle, ainsi que les paramètres utilisés lors de l'entraînement. Cela nous offre la possibilité de reproduire les résultats, de partager notre travail avec d'autres collaborateurs et d'itérer rapidement sur l'amélioration de notre modèle.

Enfin, nous avons développé un tableau de bord (Dashboard) pour visualiser les résultats de notre modèle de scoring de manière claire et intuitive. Cela facilite l'interprétation des prédictions et permet une utilisation pratique de notre solution.

Dans l'ensemble, bien que des améliorations soient nécessaires, nous avons réussi à mettre en place un processus complet et robuste, allant de l'ingénierie des caractéristiques à la modélisation, en passant par la gestion du cycle de vie des modèles et la visualisation des résultats. Cela constitue une base solide sur laquelle nous pouvons continuer à travailler pour améliorer davantage notre modèle de scoring et le rendre encore plus performant dans son utilisation future.