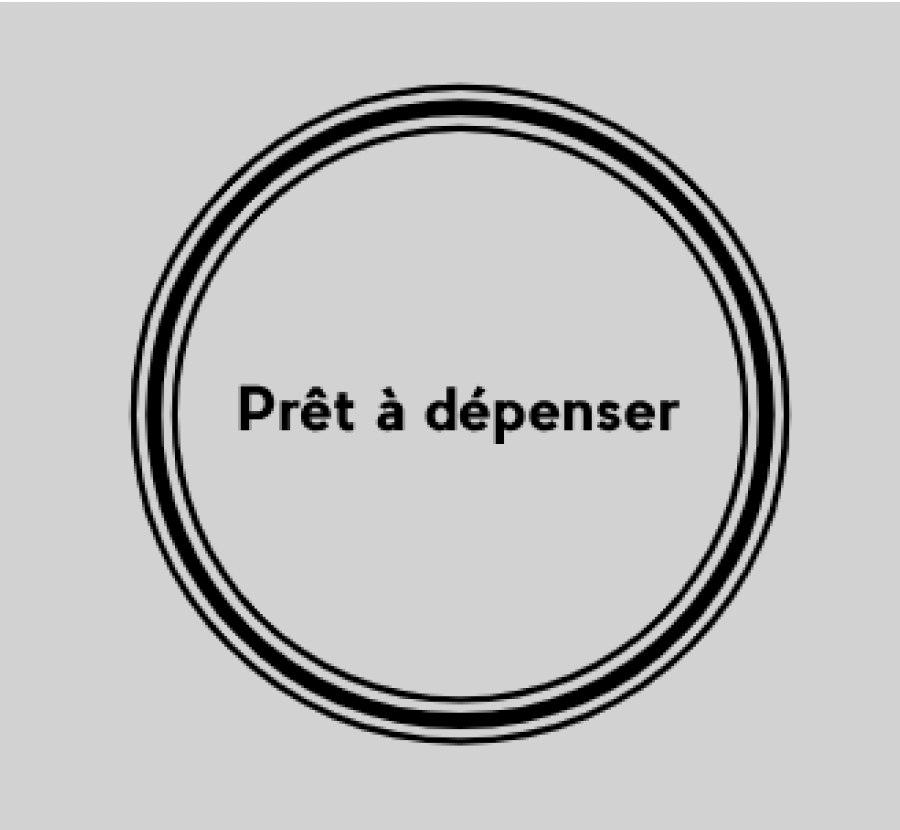
Note Méthodologie

Projet 7 – Data Scientist

*Implémentez un modèle scoring*

Lu Jing

Sommaire

* Contexte du projet
* Traitement des données
* Méthodologie d’entraînement du modèle
* Fonction du coût métier / Algorithme d’optimisation/ Métrique d’évaluation
* Interprétabilité globale et locale du modèle
* Limites et amélioration possibles
* API
* Dashboard

Contexte du projet

En tant qu’un Data Scientist dans une société financière « Prêt à dépenser »

* Mettre en œuvre un outil de “scoring crédit” pour calculer la probabilité en développant un algorithme de classification afin de classifier une demande d’un client ayant peu ou sans crédit historique, puis évaluer sa demande soit accordée ou refusée
* Mise en place d’industrialisation d’un API via ***Flask*** en appliquant le modèle entraîné en afin de donner une prédiction sur la probabilité de solvabilité d’un client de façon automatique
* Création d’un Dashboard interactif via ***Streamlit*** afin de disposer un maximum de transparence de la décision de la société et offrir un service illustratif par le service client en face des clients

Traitement des données

Un kernel Kaggle est mise à disposition pour faciliter la préparation des données / l’analyse exploratoire/ le feature engineering nécessaires à l’élaboration du modèle de scoring.

Après avoir analysé, la base des données contient 307.000 clients et 121 variables, ils ont composé par à la fois des informations générales (revenue /âge /genre etc…) et des informations spécifiques (nombre de prêts précédant / durée /solde bancaire etc…).

Afin d’entraîner un modèle de Machine Learning, une approche de pre-processing a été effectué en amont suivants les étapes suivantes :

**Sélection des données** : garder uniquement des données avec min 50% de complétion

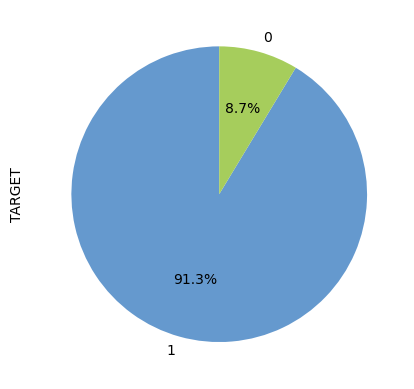
**Séparation des données** : séparation aux sous ensemble entrainement et de test (20% en test ; 80% en apprentissage)

**Imputation des données** :

* les variables numériques :remplacer les valeurs manquante par la médiane
* les variables catégorielles :remplacer les valeurs manquante par le plus fréquent

**Normalisation des données** : standardiser les données numériques

Une fois les données sont traitées, nous observons que nous confrontés à un problème de classification binaire d’une base de données avec des classes déséquilibrées (« 0 » ---🡪8.7 % des individus ne seront pas capables de rembourser ses crédits contre « 1 » ---🡪91.3 % des individus seront capables de rembourser ses crédits). ***Nous serons amenés à appliquer des différentes approches lors de l’étape de modélisation pour ré équilibrer les classes afin de réduire l’impact.***



Méthodologie d’entraînement

Afin d’entraîner le modèle, un modèle naïf, une baseline « DummyClassifier » a été créé en amont Nous l’utiliserons pour mettre en perspective les résultats obtenus avec les modèles suivants : Random Forest Classifier /Light GBM Classifier /Extreme Gradient Boosting / Logistic Regression. En même temps, nous appliquons GridSearchCV afin d’optimiser hyperparamètres les plus optimaux.

Comme évoqué précédemment un problème de déséquilibre des classes existe, nous allons donc entraîner de nouveau les modèles mentionnés ci-dessous en combinant les techniques suivantes afin d’obtenir un résultat plus optimal :

GridSearchCV

1ème approche, nous entraînons chacun de ces modèles à l’aide de GridSearchCV, qui va nous permettre de trouver les hyperparamètres qui donnent les meilleurs résultats. De plus, GridSearchCV réalise une cross validation.

GridSearchCV + SMOTE (sur échantillonnage aléatoire de la classe minoritaire)

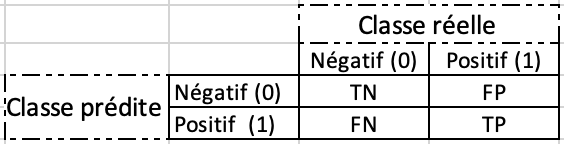
2ème approche, nous avons ensuite réalisé un sur-échantillonnage de notre classe minoritaire jusqu’à n’avoir que 2 fois plus d’individus majoritaires. Nous avons ensuite entrainé chacun des modèles présentés précédemment à l’aide d’une grille de recherche d’hyperparamètres (GridSearchCV) sur notre jeu de données sur-échantillonné.

GridSearchCV + SMOTE + RandomUnderSampler (sous échantillonnage aléatoire de la classe majoritaire)

3ème approche, nous avons sur-échantillonné notre classe minoritaire puis sous-échantillonné notre classe majoritaire puis nous avons encore réentraîné nos modèles avec GridSearchCV.

Fonction du coût métier :

Afin définir l’algorithme optimal, chacun des quatre modèles a été entraîné selon les différentes méthodes pour pailler au déséquilibre. Les classes prédites sont comparées aux classes réelles et il est possible de créer une matrice de confusion pour se représenter les erreurs commises par le modèle :



Dans le tableau ci-dessous, la classe 0 correspond aux clients solvables qui sont négatifs au refus de l’accord de prêt contrairement aux clients no solvable de la classe 1 qui sont positifs au refus de l’accord de prêt.

Accorder un crédit à un client ne pouvant pas le rembourser (FP) ----> perte

Accorder un crédit à un client pouvant rembourser (TP) ----> gain

Ne pas accorder un crédit à un client ne pouvant pas rembourser (TN) ----> neutre

Ne pas accorder un crédit à un client pouvant rembourser (FN) ----> perte

Une entreprise de finance cherche à gagner de l’argent et à « maximiser » ses gains. Il est donc possible de créer pour chaque modèle un score gain total en pondérant chaque cas par un coefficient qui pénalisera négativement les prédictions néfastes à la banque (FN et FP) et un qui valorisera les prédictions bénéfiques pour la banque (TN).

Cette fonction de coût a été implémentée afin de pénaliser l’impact des erreurs sur la décision d’octroi de crédit.

Notre meilleur modèle pour le moment (LGBM) présente la matrice de confusion ci-contre (qui représente un coût métier de 41 750).

Nous cherchons ensuite à minimiser le coût métier de notre modèle. Pour cela, nous utiliserons deux approches différentes.

Une image contenant graphique

Description générée automatiquement

Une image contenant graphique

Description générée automatiquementUne image contenant graphique

Description générée automatiquementNous gardons le meilleur modèle en appliquant une approche « choix d’un seuil de probabilité » et nous cherchons le seuil de probabilité qui minimise le coût métier et nous avons trouvé un seuil de 0.92 avec un coût métier de 25 703.

Métrique d’évaluation

Nous avons calculé des différents métriques d’évaluation afin de sélectionner le modèle le plus pertinent.

Interprétabilité globale et locale du modèle

Afin d’interpréter le modèle choisi, la méthode SHAP a été utilisée afin de comprendre comment chacune des variables du client a influencer sur la décision finale.

Interprétabilité locale permet de connaître quelles variables ont influencé la prédiction d’un client en particulier. Par example, dans le cas d’un client donné, nous voyons que les variables xxxxx ont influencé positivement sa demande de crédit que xxxxxxxx

API

Une API a été créé via ***Flask***, en suite déployé par ***Heroku*** afin de réaliser la mise en production d’une prédiction de solvabilité d’un client donné de façon automatique.

Dashboard

Afin d’avoir une meilleure interpretabilité des prédictions faites par le modèle entraîné, et d’avoir une meilleure amélioration de la connaisance client des chargés de client client, un dashboard interactif a été construit sous ***Streamlit*** en suite déployé par ***Heroku*** afin de réaliser la mise en disposition au publique.

Limites et améliorations possible

Effectuer un traitement feature enginnerring plus poussée afin d’améliorer la performance de notre modèle