# Note méthodologique

MD

2022-08-02

## I. Méthodologie d'entrainement du modèle

Plusieurs fichiers de données sont disponibles afin de réaliser ce projet. Les fichiers **application_XXXX.csv** contiennent déjà beaucoup d'informations *a priori* pertinentes pour constuire le modèle de classification. Nous pouvons commencer par exploiter cette première source d'information avant d'exploiter les autres fichiers. (A noter, basé sur l'exemple Kaggle, le meilleur score obtenu pour le ROC AUC - après *features engineering* et *hyperparameters tunning* est d'environ 0.78 avec lightGBM).

### I. a. Feature engineering

Différentes opérations ont été réalisées sur le jeu de données (cf. jypter notebook pour les détails) de manière à en réduire la dimensionalité :

- Les features avec >40% de valeurs manquantes ont été supprimées (exceptée la variable EXT_SOURCE_1 qui semble être importante d'après les résultats Kaggle).
- La majorité des variables 'flag' ne semble pas utile *a priori* pour réaliser la classification (par ex. la présence de tel document, ou de l'existence d'un numéro de téléphone) et ont donc été écartées.
- D'autres informations relatives aux enregistrements dans la base de données ont également été écartés (par ex. heure à laquelle le client à fait sa demande).

Puis de manière à construire un jeu de données adapté aux analyses :

- Les valeurs abérrantes ont été remplacées par des valeurs manquantes
- Les variables catégorielles ont été encodées (one-hot encoding)
- De nouvelles variables ont été crées (cf. jupyter notebook)

Le jeu de données final contient 141 variables pour env. 307k individus et a été standardisé (centré-réduit).

### I. b. Modèles et entrainement

Le choix a été fait de comparer deux modèles de classification : (i) régression logistique (avec régularisation *l1* ou *l2* - Lasso/Ridge), (ii) classifieur basé sur les ensembles d'apprenants faibles (XGBoost).

Alors que XGBoost peut utiliser les données manquantes pour la classification, cela n'est pas possible avec la régression logistique. Ainsi, le choix a été fait de remplacer les valeurs manquantes par la moyenne de

la variable considérée. En effet, supprimer les individus présentant au moins une valeur manquante aurait aboutit une réduction de près de 70% de la taille du jeu de données.

Il convient de noter que le jeu de données est assez déséquilibré (8% de positifs). Afin de considérer ce potentiel problème, l'entrainement des modèles a été réalisé sur plusieurs ensembles de données (avec ou sans ré-équilibrage des classes - cf. notebook pour plus de détails).

Une approche par recherche sur grille et validation croisée a été mise en place afin de sélectionner les hyperparamètres des modèles de classification (30% des données conservées pour la validation, et séparation des données d'entrainemen en 5-folds). Les hyperparamètres explorés sont détaillés pour chaque modèle dans le notebook. Finalement, deux métriques d'évaluation ont été utilisées : (i) l'aire sous la courbe ROC (AUC-ROC), (ii) le F-Score (cf. partie suivante pour davantage de détails).

# II. Coût métier, optimisation et métrique d'évaluation

Le métrique utilisée pour l'évaluation des modèles est l'aire sour la courbe ROC (AUC-ROC) puisque nous sommes dans le cas d'une classification binaire. La courbe ROC représente les valeurs du (i) taux de vrais positifs (i.e. proportion des positifs effectivement détéctés) en ordonnée, et (ii) taux de faux positifs (i.e. proportion des négatifs détéctés positifs) en abcisse, en fonction du seuil choisi pour différencier les deux catégories. Plus l'aire sous courbe tend vers un, plus le classifieur est exact (à noter, une valeur proche de 0.5 correspond à un classifieur aléatoire).

Il convient néanmoins de noter que cette métrique peut ne pas être parfaitement adéquate aux attentes métiers. En effet, l'objectif principal du modèle construit ici est d'obtenir une estimation du défaut de paiement d'un crédit de manière à l'éviter. Ainsi, maximiser le taux de vrais positifs est plus important - dans une certaine mesure, que le risque de ne pas attribuer un crédit qui aurait été remboursé (i.e. taux de faux positifs).

La métrique F-score est ici toute indiquée puisqu'elle permet justement de choisir l'importance relative du taux de vrais positifs vs. taux de faux positifs via le paramètre $\beta$. Pour les modèles construits ici, la valeur de $\beta$ a été fixée à 2 - ce qui signifie que le taux de vrais positifs est considéré comme deux fois plus important que le taux de faux positifs. Cette métrique permet donc pour un modèle donné de choisir un seuil de séparation pertinent de manière à minimiser le taux de vrais positifs. Elle permet également, via l'étude de cette valeur en fonction du seuil choisi, de sélectionner un modèle et ses hyperparamètres. Cette démarche a été utilisée dans le notebook d'analyse lié à ce projet, et l'utilisation de cette métrique est également permise dans le dashboard de manière à ce que l'utilisateur puisse choisir un seuil de classification maximisant, au niveau voulu, le taux de vrais positifs.

# III. Interprétabilité globale et locale du modèle

Le notebook détaillé est disponible sur GitHub (https://github.com/mxdub/deploy/blob/main/Projet7_notebook.ipynb, sections Global/Local features importance) - peut-être nécessaire pour visualiser les représentations graphiques.

Il est possible d'interpréter le modèle, ou plus exactement l'importance des différentes features pour la classification via différentes approches. Une première manière est de s'intéresser au vecteur de *features importances* produit par XGBoost. Ce dernier décrit l'importance relative des différentes features pour la classification (au niveau global), néanmoins, il ne permet pas de comprendre comment les valeurs des features influencent la sortie du modèle (cad est-ce qu'une valeur importante de telle variable augmente ou au contraire réduit la probabilité d'être attribué à telle classe). Avec cette première approche, il est possible de montrer que les variables les plus importantes sont : les sources d'informations externes, le niveau d'étude, l'origine des revenus, le sexe, ainsi que le type de contrat.

Une seconde approches a donc été utilisée, la méthode SHAP (pour SHapley Additive exPlanations), basée sur l'idée de perturbations, qui permet d'expliquer les prédictions individuelles (i.e. locales). L'ensemble des explications locales peuvent être représentées sous forme de distribution pour produire un explication globale au modèle (présenté dans le notebook). Cette méthode permet également d'identifier les features les plus importantes : sources d'info. externes, le sexe, le taux de remboursement, etc. qui sont donc assez similaires à celles observées avec le vecteur de *feature importances*. Néanmoins, elle permet d'affiner l'interprétation en indiquant le sens dans lequelle les valeurs des features influencent la sortie du modèle (par ex. on peut ici montrer que plus un client a un bon score pour les sources externes d'information, moins il a de chance de faire défaut pour un crédit, ou encore, que les femmes ont moins de chances que les hommes de faire défaut).

Etant avant tout utilisée pour interpréter les résultats localement (cad pour un individu), ces valeurs sont également représentées pour chaque client dans le dashboard, et permettent donc aux chargés de relation client ainsi qu'aux clients de comprendre pourquoi il peuvent obtenir ou non un crédit selon le modèle.

Pour finir, on peut noter qu'une valeur de Shapley est obtenue pour chaque catégorie d'une feature catégorielle - rendant l'interprétation possiblement délicate. Il est néanmoins possible d'additionner les valeurs de chaque catégorie de manière à obtenir une valeur de Shapley pour la feature catégorielle. Cette démarche a été adopté dans le notebook, et facilite l'interprétation des résultats.

# IV. Limites & améliorations possibles

Une première limite que l'on peut avancer est qu'il n'existe pas de valeur optimale permettant d'atteindre les 100% de classifications exactes, ainsi il existe toujours des faux négatifs ainsi que des faux positifs, les deux étant sources de pertes. Dans le même ordre d'idée, lorsqu'une importance forte est placée sur le taux de vrais positifs, le taux de faux positifs peut devenir assez élevés (par ex. de l'ordre de 60% avec un $\beta = 5$), ce qui signifie que plus d'une personne sur deux qui aurait probablement remboursé son crédit n'obtiendra pas ce dernier - il sagit ici encore d'une perte importante du point de vue financier.

Il pourrait également être intéressant de produire une métrique considérant les sommes engagées (et notamment les bénéfices / pertes effectives), en effet, l'utilisation du F-Score pose l'hypothèse (implicite) que les valeurs des crédits sont égales quelque soit la classe considérée (ou du moins, distribuées de manière similaire dans les deux classes), or rien de permet d'étayer cette hypothèse *a priori* et les gains pourraient être maximisés en levant cette hypothèse.

D'un point de vue méthode, et notamment du dashboard, il aurait pu être pertinent de permettre de regrouper les différentes valeurs d'une variable catégorielle pour les scores d'importances locales (tel que réalisé dans le notebook d'analyse).

Il serait également intéressant d'utiliser l'ensemble des sources d'informations (i.e. tous les fichiers fournis) afin d'évaluer leur pertinence pour améliorer notre classifieur. A noter que les temps de calcul nécessaires pour le choix du modèle de classification et l'optimisation des hyper-paramètres seraient alors bien supérieurs.