# Note méthodologique projet 7

## Rappels des objectifs et de la problématique

En tant que Data Scientist au sein d'une société financière, nommée **"Prêt à dépenser** »,qui propose des crédits à la consommation pour des personnes ayant peu ou pas du tout d'historique de prêt, l’objectif de se projet est de **mettre en œuvre un outil de “scoring crédit” pour calculer la probabilité**qu’un client rembourse son crédit.

L’objectif est donc de développer un **algorithme de classification** en s’appuyant sur des sources de données variées (données comportementales, données provenant d'autres institutions financières, etc. pour 307 511 clients).

## La méthodologie d'entraînement du modèle

Pour entrainer notre modèle, nous sommes partis des données disponibles et après EDA et feature engineering, avons récupérer une base de 307 511 clients et 81 features.

Pour pouvoir utiliser un modèle sur nos données, celle-ci doivent passer par un processus de pre-processing que l’on appliquera à l’aide d’un pipeline.

Avant de pouvoir réaliser le pre-processing, nous procédons à la **séparation des données** en données d’entrainement (70% des données disponibles sur lesquelles ont entrainera le modèle) et données de test (qui ne permettront d’évaluer le modèle) pour ne fiter notre pipeline que sur les données d’entrainement, évitant ainsi la fuite de données.

Ainsi préparé, nous appliqueront le pipeline qui engendrera les modifications suivantes :

**Pour les données catégorielles :** imputation des données avec les valeurs les plus fréquentes et encodage des données avec get\_dummies.

**Pour les données numériques :** imputation des données avec les valeurs médianes et normalisation des données avec Robustscaler.

De plus, l’EDA a mis en évidence un autre problème que nous allons devoir traiter pour procéder à la modélisation : notre problème de classification binaire (0 : les clients voient leur crédit accordé, 1 : le crédit est refusé) possède une classe (la classe 1) qui est sous-représenté.

Avec près de 92% des clients qui voient leur crédit accordé contre 8% pour lesquels il est refusé, un classifier « naïf » de type dummy pourrait nous donner des résultats satisfaisants (92% d’accuracy) mais ne détecterait pas les clients dont on devrait refuser le crédit.

Pour produire un modèle satisfaisant il est donc nécessaire de rééquilibrer nos classes.

SMOTE, une méthode d’over-sampling, a permis de générer de nouvelles données dans notre classe sous-représentée à partir des observations existantes, nous permettant ainsi de régler ce problème.

Dans l’optique d’éviter la fuite de données, cette méthode n’a été appliqué que sur nos données d’entrainement.

Avec les données ainsi retravailler, 4 modèles ont été entrainés pour pouvoir évaluer les résultats de chacun :

* Régression logistique
* Random Forest
* Xgboost

Chacun de ces modèles ont été évalué sur différents métrics :

* Roc\_Auc
* Accuracy
* Recall
* F1 score
* Durée de traitement

Ces métrics nous serviront de base pour évaluer nos modèles et nous permettront ainsi de retenir celui ou ceux qui seront les plus pertinent pour effectuer une recherche sur grille avec une cross validation.

## La fonction coût métier, l'algorithme d'optimisation et la métrique d'évaluation

### Fonction coût métier

Les fonctions de coût pour les algorithmes entrainés sont les suivantes :

|  |  |
| --- | --- |
| Algorithmes | Fonction de coût |
| Régression logistique | Fonction sigmoïde |
| Random Forest | Minimisation du coeff. GINI |
| Xgboost Classifier | Régression logistique pour classification binaire |

### Algorithme d’optimisation

Les deux modèles les plus performants ont été retenu et optimisé à l’aide d’une recherche sur grille et d’une cross validation (4 fold). La meilleure combinaison d’hyperparamètre a été retenu pour chacun des modèles qui ont été à nouveau évalué sur chacun des métrics cités précédemment.

Le modèle plus performant a alors été retenu. La régression logistique propose globalement le meilleur score sur l’ensemble des métrics évalués et est donc l’option privilégiée.

### Métrique d’évaluation

Ayant à faire un problème de classification binaire avec des classes déséquilibrés (8% de clients sont en défaut), nous considérons la matrice de confusion suivante :

|  |  |  |
| --- | --- | --- |
|  | Prédiction sans défaut | Prédiction en défaut |
| Client sans défaut | Vrai positif | Faux négatif |
| Client en défaut | Faux positif | Vrai négatif |

La logique, du point de vue bancaire, nous pousse à penser qu’un client en défaut auquel on accorde un crédit est une forte source de perte. Notre objectif ici est donc de minimiser au mieux le nombre de faux positif tout en maximisant le nombre de vrai positif.

Nous avons donc un objectif double :

* Maximiser le recall

Une image contenant texte

Description générée automatiquement

* Maximiser la précision

Une image contenant texte

Description générée automatiquement

Un client en défaut nous paraissant plus couteux que l’acquisition d’un nouveau client, nous préférerons dans la mesure du possible maximiser le recall, source des pertes les plus importantes par rapport au gain potentiel que nous rapporterai un nouveau client.

Pour maximiser au mieux ces deux métrics, nous avons développé un score custom en attribuant un coefficient à chacun de nos éléments.

Faux négatif = -0.5 🡺 pour chaque client à qui on refuse un crédit et qui rembourserai, on perd les intérêts

Faux positif = -2 🡺 pour chaque client qui ne rembourse pas, on perd potentiellement le capital investi

Vrai positif= 0.5 🡺 pour chaque client à qui on accorde un crédit et qui rembourse, on gagne les intérêts

Vrai négatif = 0 🡺 ni gain ni perte dans ce cas de figure

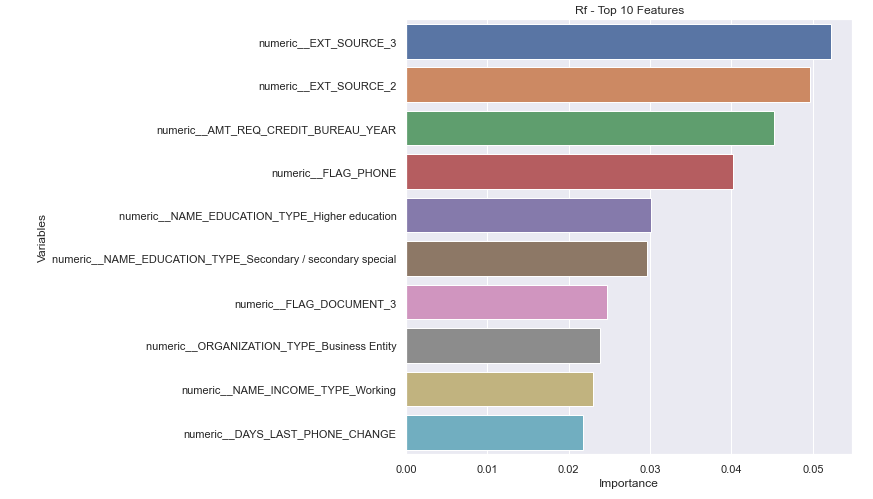
Nous pouvons alors proposer la métric d’évaluation suivante qui nous permet de valoriser les éléments les plus important dans notre cas de figure :

tp\*tp\_value + tn\*tn\_value + fp\*fp\_value + fn\*fn\_value

## L’interprétabilité globale et locale du modèle

Pour offrir une compréhension du modèle optimale aux clients et aux équipes amenées à y recourir, le modèle est expliqué à un niveau global et à un niveau local.

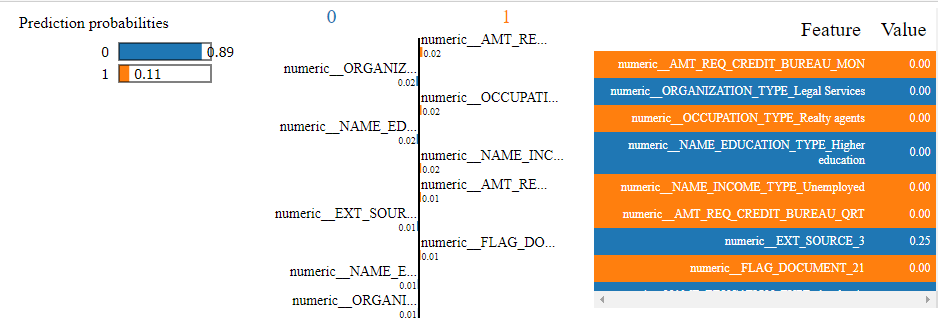
A un niveau global, l’importance des features nous permet de comprendre les bases sur lesquelles le modèle s’appuie pour classifier les clients. Ainsi, lorsque l’on considère l’ensemble des inputs (données d’entrainement), il nous est possible d’expliqué quels features ont le plus pesés dans la balance.



Exemple issue de la modélisation random forest

*EXT\_source 2 et 3 sont les features qui ont globalement le plus d’impact sur notre modèle.*

A un niveau local, nous avons utiliser LIME (Local Interpretable Model-agnostic Explanations) qui nous permet de déterminer pour un individu les features qui ont conduits à sa classification.



Exemple issue de la modélisation random forest

*Ici notre client a vu son crédit accordé et ce notamment grâce à EXT\_Source 3 qui est évalué positivement dans son cas.*

Ces éléments nous permettent ainsi de comprendre globalement comment le modèle fonctionne et plus particulièrement de déterminer pour un client ce qui a conduit à l’acceptation ou au refus de son crédit.

## Les limites et les améliorations possibles

Il y a plusieurs limitation / axes d’améliorations à envisagé sur ce projet. Nous en retiendrons trois principales qui nous permettrait d’obtenir de meilleurs résultats.

1. **Le feature engineering**

Le feature engineering n’a pas été poussé à son paroxysme et n’a pas forcément pris en compte l’ensemble des données à disposition. Notre objectif dans ce projet étant concentré sur le déploiement d’une API et d’un dashboard, cette étape n’a en effet pas été développé au maximum bien qu’elle nous permettrait probablement de tirer de meilleure performance de nos modèles. Il est également à noter que nous traitons ici un set de données assez volumineux et que l’ajout de nouvelles features aurait également ralenti le temps de traitement.

Une piste d’optimisation consisterait donc à approfondir l’exploration et l’analyse des données pour proposer un feature engineering de meilleure qualité.

1. **Le métric d’évaluation**

Dans ce projet, nous avons proposé une approche de métric custom pour évaluer la pertinence de nos modèles. Ce métric a été calculé à l’aide du bon sens mais ne permet pas de prendre en considération la situation réelle de l’entreprise. Pour améliorer cette approche, une analyse des gains et des couts de l’entreprise par rapport aux prédictions de notre modèle est essentielle. C’est en effet une analyse de ce type qui nous permettra de calibrer les coefficients qui servent de base à cet métric et donc de considéré le gain optimal.

Cette piste d’optimisation est de loin la plus importante car c’est elle qui nous permet d’évaluer nos modèles.