**Note Méthodologique – Implémentez un modèle de scoring**

Sommaire :

1. Contexte
2. Méthodologie d'entraînement du modèle
3. Fonction coût métier, l'algorithme d'optimisation et la métrique d'évaluation
4. Interprétabilité globale et locale du modèle
5. Limites et améliorations possibles
6. *Contexte*

La **société financière « Prêt à dépenser »** propose des **crédits à la consommation** pour des personnes ayant peu (voire pas du tout) d’historique de prêt. Dans ce contexte, elle est confrontée à **deux problématiques principales**:

* **Anticiper la possibilité qu’un nouveau client**, peu connu jusque-là, **n’ait pas la capacité de rembourser** le prêt demandé
* Permettre aux chargés de relation client d’**expliquer de manière transparente** les décisions d’octroi de crédit, et, par voie de conséquence, leur permettre de **mieux connaitre leurs clients**

Face à ces deux problématiques « métiers », l’entreprise souhaite **développer un outil de scoring crédit** pour **calculer la probabilité qu’un client rembourse son crédit**. En langage de *Data Scientist*, il s’agit donc de développer une solution dans laquelle on mettra en œuvre un **algorithme de classification supervisée**.

1. *Méthodologie d'entraînement du modèle*
   1. *Feature engineering*

La société nous a fourni une **dizaine de jeux de données**, tous assez différents et avec un grand nombre de variables, qui ne seront pas toutes utiles dans le cadre de notre travail. C’est pourquoi nous avons mis en place **plusieurs prétraitements & traitements** :

* **Suppression des colonnes** avec un **fort taux de valeurs manquantes** & imputation de la médiane dans les valeurs manquantes restantes
* **Vérification de la « qualité » de nos données**, avec gestion des colonnes problématiques (qui contiennent par exemple des valeurs absurdes dans le cadre de notre étude)
* **Obtention de nouvelles variables à partir d’agrégats** (numériques et catégoriels)
* Création de **nouvelles variables pertinentes** du point de vue « métier »

Ce travail de *feature engineering* a été réalisé sur la base de plusieurs **notebooks Jupyter issus de Kaggle** afin de fluidifier le traitement de jeux de données complexes. A la suite de ce premier travail, le jeu de données produit pour prédire le score client **contient près de 400K clients et une trentaine de variables** ( (dont 25 variables « prédictrices », une variable « ID » et une variable « cible » qui contient la classe à prédire).

* 1. *Résolution du déséquilibre entre les classes*

Telle qu’elle est initialement construite, la base de données qui va nous servir à l’entrainement de notre modèle est déséquilibrée. En effet, elle contient **92% de clients qui remboursent effectivement leur prêt** (valeur « cible » = 0, c’est la « classe majoritaire »), ce qui signifie que seulement **8% ne le remboursent pas** (valeur « cible » = 1, c’est la « classe minoritaire »). Puisque notre objectif reste bien de cibler en priorité les 8% de clients qui ne rembourseraient potentiellement pas leur prêt, il s’agit de traiter cette problématique de surreprésentation du groupe de clients qui remboursent leur prêt.

Pour résoudre ce problème, **plusieurs solutions s’offrent à nous** :

* **Changer la structure** au global **de l’algorithme**
* **Réduire le nombre d’individus** dans la classe majoritaire
* **Collecter davantage de données** sur la classe minoritaire
* **Dupliquer** des individus sous-représentés
* **Pondérer les observations dans le training** *(traité directement dans les hyperparamètres de nos algorithmes)*
* Choisir une **métrique de performance adaptée** *(expliqué ci-dessous)*
* Créer des **individus « synthétiques »** *(technique de sur-échantillonnage de minorité synthétique, appelée SMOTE)*

Dans ce contexte, nous avons décidé de mettre en œuvre **les trois dernières solutions** pour résoudre notre problème de déséquilibre.

* 1. *Entrainement du modèle*

Une fois la **problématique de déséquilibre résolue**, nous avons **testé différents modèles de classification** avec recherche d’hyperparamètres et cross validation :

* **Régression logistique**
* **SVM**
* **Forêt Aléatoire**

Le choix du meilleur modèle s’est ensuite basé sur le modèle avec le meilleur score sur le jeu de test.

1. *Fonction coût métier, l'algorithme d'optimisation et la métrique d'évaluation*
   1. *Fonction coût métier & métrique d’évaluation*

Rappelons en préambule que **notre client est une banque** et qu’elle cherche donc à :

* Avant tout, **repérer les clients qui ne rembourseraient pas leur prêt alors qu’on avait pourtant prédit qu’ils seraient en mesure de le rembourser**

*🡪 On cherche à minimiser la part de faux négatifs et donc à maximiser le pourcentage de vrais positifs (****maximiser le recall****)*

* Ensuite, **repérer les clients détectés comme ne pouvant pas rembourser leur prêt alors qu’ils auraient été en mesure de le faire**

*🡪 On cherche à minimiser la part de faux positifs (****maximiser la précision****)*

**Afin de répondre à notre problématique métier, le recall reste plus important que la precision**, car on préférera limiter la perte financière plutôt que limiter la perte potentielle de clients. On cherche donc un compromis entre recall et précision, et on cherchera pour ce faire à **maximiser le F Beta Score** (avec **Beta** le **coefficient d’importance relative du recall par rapport à la precision**).

On cherche donc à **attribuer un « poids » au Beta** ; pour ce faire, il faut estimer d’un point de vue métier sa valeur, via :

* **Coût moyen du défaut de paiement**
* **Coût d’opportunité d’un client potentiel accidentellement écarté**

**Pour y parvenir, nos hypothèses métiers sont les suivantes** (elles seront à ajuster en fonction du dialogue avec les métiers justement et parce que les consignes du projet en tant que telles ne nous permettent pas de les fixer directement) :

* Chaque défaut entraine la dépense d’un cinquième (20%) du montant du crédit en pertes/frais de recouvrement (coefficient recall = 20% x montant moyen du crédit des personnes en défaut de paiement)
* Un client a 12% de chance de souscrire au crédit quand il en fait la demande à un conseiller ; le coût d'opportunité pour un client potentiel accidentellement écarté est de 12% également (coefficient précision = 12% x montant moyen du crédit des personnes sans défaut de paiement)

On obtient donc : **Beta = coefficient recall / coefficient précision = 2,7**

* 1. *Algorithme d’optimisation*

La meilleure combinaison d’hyperparamètres a été retenue pour chaque algorithme. Le modèle ayant le meilleur score en cross validation sur le jeu de training a été retenu : **il s’agit du modèle de Forêt Aléatoire.**

1. *Interprétabilité globale et locale du modèle*

A faire

1. *Limites et améliorations possibles*

Nous avons identifié **plusieurs points d’amélioration possibles** :

*Améliorations - Modélisation*

* Nous avons conçu les notebooks comme des outils de travail propre au data scientist de cette mission : **les variables d’environnement**, qui appellent à date les items dans ses fichiers personnels, **pourraient être retravaillés afin de permettre à n’importe qui d’utiliser les notebooks**
* Nous avons modélisé nos données grâce à un **échantillon sur le training set** ; avec une machine plus puissante, il pourrait être intéressant de **faire tourner notre modèle sur l’ensemble des données**
* Nous pourrions **rajouter un algorithme de clustering** pour réaliser la séparation des clients en différents clusters/groupes (et non pas via la prédiction, qui est bien une utilisation métier mais pas de data science)

*Améliorations - Dashboard*

* **Rajout du coût métier dans le dashboard**, afin de permettre à un banquier de savoir exactement ce qu’il perdrait (ou gagnerait) à accepter/refuser le dossier d’un client *(pour rappel : ce coût métier a été présenté plus haut dans cette note méthodologique puis dans la présentation à l’examinateur)*
* **Insertion des valeurs de Shapley dans le dashboard**, non présentées car problème de version *(pour rappel : ces valeurs ont bien été présentées dans le notebook correspondant à la modélisation)*
* Allégement du dashboard, qui fonctionne à date pour l’utilisation relativement simple qu’on veut en faire ici (utilisation par quelques utilisateurs seulement), mais difficile à mettre en œuvre dans le contexte global d’une entreprise (ou trouver un autre logiciel)