Note Méthodologique

OpenClassrooms - Parcours Data Scientist

Project 7 : Implémentez un modèle de scoring

Fanjamalala Rajaonalison

Décembre 2021

I. Contexte

L'entreprise **"Prêt à dépenser"**, qui propose des crédits à la consommation pour des personnes ayant peu ou pas du tout d'historique de prêt, souhaite mettre en œuvre un outil de **"scoring crédit"** pour calculer la probabilité qu'un client rembourse son crédit, puis classifie la demande en crédit accordé ou refusé.

Le développement d'un algorithme de classification a, pour cela, été réalisé en s'appuyant sur des sources de données variées (données comportementales, données provenant d'autres institutions financières, etc.).

Comme tout projet de Machine Learning, ce projet de classification s'est construit en plusieurs étapes successives (*fig.1*).

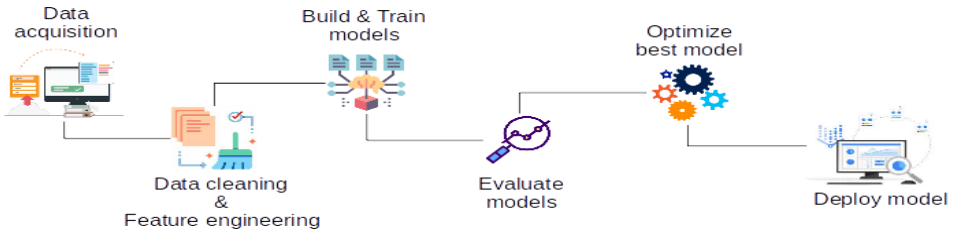


Fig.1 : Démarche d'un projet de Machine Learning

Chacune de ces étapes est présentée dans la partie méthodologie du présent document.

II. Méthodologie

1. Acquisition des données

Les données ont été récupérées sur Home Credit Default Risk et sont constituées de dix tables contenant des informations relatives à différents clients (données financières, comportementales, etc.) et des notes descriptives. Cette première étape consiste donc à récupérer les diverses données nécessaires et effectuer une brève exploration.

2. Data Cleaning & Feature engineering

Le jeu de données, constitué des dix tables, contient des données incomplètes, non formalisées. Cette étape consiste pour chaque tableau utile à standardiser les données, traiter les valeurs manquantes puis assembler les tableaux former une seule grande base.

Pour cela, on s'est basé sur LightGBM with simple features contenant :

- une fonction pour chaque table, qui supprime les données aberrantes, encode les features catégorielles, créée des features spécifiques à l'étude bancaire (crédit, taux, etc.) en appliquant les fonctions min, max, moyenne ou somme de certaines tables groupées.
- une fonction concaténant les tables.

Ajoutées à cela une fonction remplaçant les valeurs manquantes par la médiane pour chaque feature est réalisée.

Observant un trop grand nombre de features (N = 556), une étape de sélection de features a été réalisée en se basant sur Feature selection qui fait appel à plusieurs méthodes (corrélation de Pearson, Chi2, RandomForest, LightGBM) afin de sélectionner les cent plus importants features en comparant les résultats de toutes ces méthodes.

3. Construction des modèles de classification des clients

Un modèle de prédiction peut généralement être représenté par une fonction qui prend en entrée des données et renvoie en sortie une décision, décision binaire dans notre cas (avec ou sans difficulté de paiement).

Pour construire ce type de modèle, et plus particulièrement dans le cas d'apprentissage supervisé où nous souhaitons expliquer une variable de sortie, l'échantillon est subdivisé en 2 parties :

- L'échantillon d'apprentissage : correspond à l'échantillon principal pour entraîner et ajuster les modèles. Il représente, ici, 75% des données.

- L'échantillon de test : correspondant à l'échantillon utilisé uniquement pour évaluer le modèle optimal (résultat de la validation croisée). Le modèle sélectionné est donc indépendant de cet échantillon test. Il représente 25% des données.

Notre projet de classification des clients étant binaire, il faut s'assurer que chaque échantillon contient une proportion raisonnable des classes 0 et 1. L'Analyse exploratoire des données a cependant identifié un fort déséquilibre entre la classe 0 et la classe 1 (fig.2).

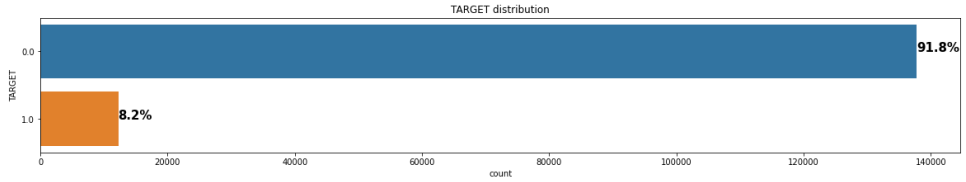


Fig. 2: Distribution des classes 0 et 1 montrant un déséquilibre

Si habituellement, un simple train_test_split() de Scikit-Learn aurait permis d'obtenir une répartition aléatoire des individus en base d'apprentissage et de test, l'utilisation de SMOTE (Synthetic Minority Oversampling Technique) est ici nécessaire pour corriger ce problème de déséquilibre de classe. En effet, SMOTE permet de créer des données synthétiques à partir des données existantes et rééquilibrer les classes (*Tab.1*).

Tab.1 : Description des échantillons d'apprentissage avant et après SMOTE

	Avant SMOTE	**Après SMOTE**
Train size	X : (104997, 99) Y : (104997,)	X : (192652, 99) Y : (192652,)
Pourcentage de 1 dans Y train	8.26	50.0

Une fois les données séparées en échantillon train et test, plusieurs modèles de classification ont été testés : un algorithme statistique classique (LogisticRegression), un algorithme forêt aléatoire (RandomForestClassifier), et trois algorithmes boosting (LGBMClassifier, XGBClassifier, CatBoostClassifier).

4. Evaluation des modèles

L'objectif de ce projet étant de permettre à l'entreprise d'accorder ou non la demande de crédit des clients en s'appuyant sur leur probabilité de remboursement grâce à cet outil. Il est donc important que l'outil minimise les opportunités manquées (cas où le crédit est refusé à un client sans difficulté de paiement) et évite les pertes financières (cas où le crédit est accordé à un client avec un difficulté de paiement).

En utilisant les terminologies relatives à la matrice de confusion, qui est tableau croisé entre les valeurs réelles et les prédictions (*tab.2*), l'idée est de minimiser les Faux Positifs et éviter les Faux Négatifs.

Tab.2 : Matrice de confusion

	Clients prédits avec des difficultés de paiement	**Clients prédits sans difficultés de paiement**
Clients ayant réellement des difficultés de paiement	Vrais positifs (VP)	Faux négatifs (FN)
Clients sans difficulté de paiement	Faux positifs (FP)	Vrais négatifs (VN)

La performance de chaque modèle de classification est donc évaluée à partir de plusieurs métriques tenant en compte ce critère (minimiser FP et éviter au maximum FN).

- Métriques

L'indicateur le plus simple est *l'**accuracy*** qui indique le pourcentage de bonnes prédictions (VP et VN). Pour compléter l'*accuracy*, on calcule également le ***recall*** qui se concentre uniquement sur les clients ayant réellement des difficultés de paiement et donne une indication sur la part de FN. Un troisième indicateur qui vient compléter l'*accuracy* et le *recall* est la ***precision***. Il se concentre uniquement sur les clients pour lesquels le modèle a prédit des difficultés de paiement et donne une indication sur les FP.

$$\text{Accuracy} = \frac{VP + VN}{VP + VN + FN + FP} \qquad \text{Recall} = \frac{VP}{VP + FN} \qquad \text{Precision} = \frac{VP}{VP + FP}$$

L'accuracy, le recall et la *precision* sont à utiliser ensemble pour donner une vision complète de la performance. On cherche à avoir des valeurs le plus proche possible de 100% pour les trois indicateurs. Pour notre problématique métier, le *recall* est plus important que la *precision* car il est plus important de limiter un risque de perte financière (FN) plutôt qu'un risque de perte de client potentiel (FP).

Le ***F-score*** combine la *precision* et le *rappel*, permettant d'obtenir une mesure unique plus facilement interprétable, fondé sur la moyenne harmonique des deux indicateurs. Cette moyenne harmonique *F-score* peut être pondérée afin de donner plus d'importance à un indicateur, c'est le ***F-beta***. On cherche à avoir des valeurs le plus proche possible de 100% pour ces deux indicateurs.

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \qquad F_\beta = (1 + \beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}}$$

Le choix de β détermine le poids que l'on souhaite donner à chacune des deux mesures : si $\beta < 1$ la *précision* est privilégiée, si $\beta > 1$ c'est le *recall* qui est privilégié. Afin de porter l'importance sur *recall* et répondre à notre problématique métier, on choisit de définir **$\beta = 3$** (hypothèse qui pourra être ajustée avec un interlocuteur métier).

- Fonction coût-métier

L'objet de cette fonction est de calculer un autre indicateur ***Bank score*** répondant au mieux à la problématique métier qui est ici d'éviter au maximum les FN et si possible minimiser les FP.

Comme les autres métriques, ce Bank score s'appuye sur la matrice de confusion. Il se différencie par le fait qu'il rajoute une pondération aux différentes valeurs selon leur degré d'importance ($P_{VP} = P_{VN} = 1$, $P_{FP} = -1$ et $P_{VN} = -10$) et qu'une normalisation est effectuée pour obtenir le score.

$$model = (VN * P_{VN}) + (FP * P_{FP}) + (FN * P_{FN}) + (VP * P_{VP})$$

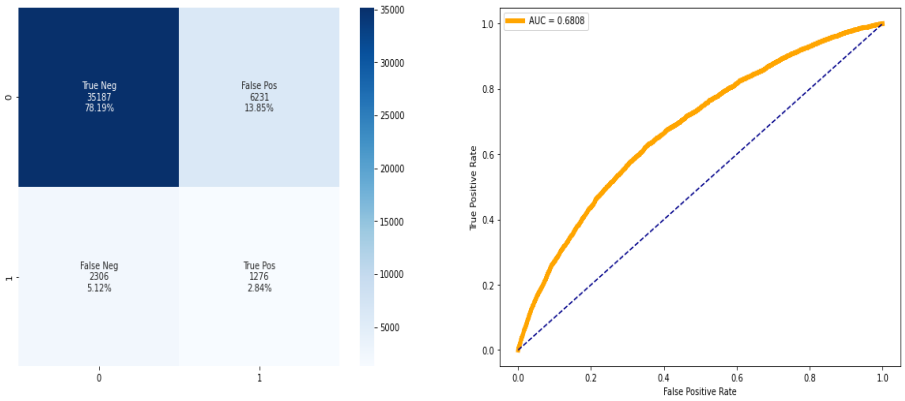
$$positif\ model = (VN + FP) * P_{VN} + (FN + VP) * P_{VP}$$

$$negatif\ model = (VN + FP) * P_{FP} + (FN + VP) * P_{FN}$$

$$Bank\ score = \frac{model - negatif\ model}{positif\ model - negatif\ model}$$

On cherche à avoir des valeurs le plus proche possible de 100% pour cet indicateur.

En se basant sur ces divers indicateurs, et plus particulièrement sur *Bank score*, le meilleur modèle est le RandomForestClassifier (*Fig.3*).



Accuracy	Precision	Recall	F-score	Fbeta	Bank score
0.81	0.17	0.36	0.23	0.31	0.69

Fig.3 : Résultats du meilleur modèle (RandomForestClassifier) : Matrice de confusion, AUC score, Métriques

5. Optimisation du meilleur modèle

L'objectif est de trouver la meilleure combinaison d'hyperparamètres pour le meilleur modèle (ici RandomForestClassifier) qui maximiserait le Bank score.

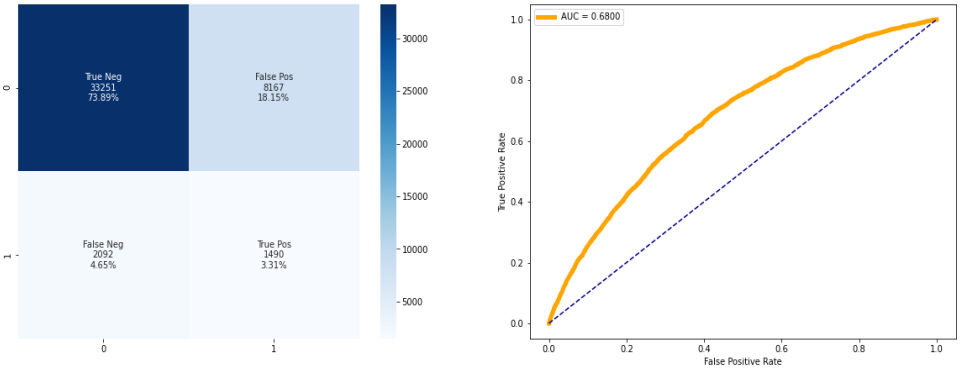
Pour cela, la méthode effectuée est GridSearchCV constitué d'une Cross Validation qui va permettre d'entraîner, ajuster et valider le modèle successivement sur k-échantillons issus d'une découpe aléatoire du dataset ; combiné avec un Grid Search qui va permettre de tester une série de paramètres et de comparer le Bank score pour en déduire le meilleur paramétrage.

Pour ce projet :

- k (pour la Cross Validation) est défini à : 5

- les paramètres testés sont : n_estimators (nombre d'arbres), max_depth (profondeur maximale d'un arbre), random_state (générateur de nombres aléatoires) et max_samples (fraction d'observations à sélectionner pour chaque arbre)
- le score à maximiser est : *Bank score*.

Les résultats (*Fig.4*) obtenus suite à l'optimisation du meilleur modèle sont retrouvés ci-dessous :



Accuracy	Precision	Recall	F-score	Fbeta	Bank score
0.77	0.15	0.42	0.23	0.36	0.68

Fig.4 : Résultats du modèle RandomForestClassifier optimal : Matrice de confusion, AUC score, Métriques

III. Interprétabilité du modèle de classification

Le modèle étant destiné à des équipes opérationnelles devant être en mesure d'expliquer les décisions de l'algorithme à des clients réels, le modèle est accompagné d'une partie interprétabilité pour fournir du sens en termes compréhensibles par toute personne.

Pour réaliser cette étape, deux approches ont été réalisées :

- ***feature_importances_*** qui permet de représenter l'importance relative de chaque feature,
- librairie ***Shap*** qui peut s'appliquer à n'importe quel modèle de machine learning et permet de comprendre l'importance de chaque feature en comparant la prédiction du modèle avec et sans ce feature.

Fig. 5 : Interprétabilité de la Classification (Feature importance & Shap)

IV. Limites et améliorations

La principale limite de ce projet vient du fait que la modélisation effectuée se base sur une hypothèse forte mais non confirmée par les équipes métier : la définition d'un Bank score s'appuyant sur des pondérations hypothétiques des prédictions selon si elle est vrai ou fausse. Le premier axe d'amélioration serait donc de définir plus précisément ce Bank score en collaboration avec les équipes métier.

Une autre piste d'amélioration serait d'étoffer l'interprétabilité du modèle en considérant les variables issues du one hot encoding comme un seul et même feature (en d'autres termes revenir au feature initial).

Par ailleurs, la partie de traitement préalable du jeu de données a été abordée de façon assez succint en réutilisant un notebook issu de Kaggle qui se base principalement sur une table du jeu de données. Des améliorations peuvent probablement être apportées à la modélisation en considérant d'autres features plus pertinents en travaillant conjointement avec les équipes métier.