Projet 7 : Implémentez un modèle de scoring

Jessy PATRICE
Parcours Data scientist

INTRODUCTION

L'entreprise « Prêt à dépenser » souhaite mettre en œuvre un outil de "scoring crédit" pour calculer la probabilité qu'un client rembourse son crédit, puis classifier la demande en crédit accordé ou refusé. Elle souhaite donc développer un algorithme de classification en s'appuyant sur des sources de données variées (données comportementales, données provenant d'autres institutions financières, etc.).

De plus, les chargés de relation client ont fait remonter le fait que les clients sont de plus en plus demandeurs de transparence vis-à-vis des décisions d'octroi de crédit. Cette demande de transparence des clients va tout à fait dans le sens des valeurs que l'entreprise veut incarner.

Prêt à dépenser décide donc de développer un dashboard interactif pour que les chargés de relation client puissent à la fois expliquer de façon la plus transparente possible les décisions d'octroi de crédit, mais également permettre à leurs clients de disposer de leurs informations personnelles et de les explorer facilement.

La Mission

1. Construire un modèle de scoring qui donnera une prédiction sur la probabilité de faillite d'un client de façon automatique.
2. Construire un dashboard interactif à destination des gestionnaires de la relation client permettant d'interpréter les prédictions faites par le modèle, et d'améliorer la connaissance client des chargés de relation client.
3. Mettre en production le modèle de scoring de prédiction à l'aide d'une API, ainsi que le dashboard interactif qui appelle l'API pour les prédictions.

LA METHODOLOGIE D'ENTRAINEMENT DU MODELE

1-Présentation du jeu de données

« Prêt à dépenser » est un service dédié à la fourniture de prêts à la population non bancarisée. Leur objectif consiste à prédire si un client remboursera ou non un prêt ou s'il aura des difficultés. L'entreprise a mis à disposition 7 fichiers CSV contenant des données spécifiques :

application_train/application_test : les principales données de train et de test donnant des informations sur chaque demande de prêt. Chaque prêt a sa propre ligne et est identifié par la caractéristique SK_ID_CURR. Les données de train sont accompagnées de la caractéristique TARGET indiquant 0 : le prêt a été remboursé ou 1 : le prêt n'a pas été remboursé.

bureau : données concernant les crédits antérieurs du client auprès d'autres institutions financières. Chaque crédit précédent a sa propre ligne dans bureau, mais un prêt existant dans application_(train|test) peut avoir plusieurs crédits précédents.

bureau_balance : données mensuelles concernant les crédits précédents dans bureau. Chaque ligne correspond à un mois de crédit précédent, et un crédit précédent unique peut avoir plusieurs lignes, une pour chaque mois de la durée du crédit.

previous_application : demandes précédentes de prêts des clients qui ont des prêts dans les données de application_(train|test). Chaque prêt actuel dans les données de 'application_(train|test)' peut avoir plusieurs prêts précédents. Chaque demande précédente a une ligne et est identifiée par la caractéristique SK_ID_PREV.

POS_CASH_BALANCE : données mensuelles sur les prêts au point de vente ou en espèces que les clients ont eu avec la banque. Chaque ligne correspond à un mois d'un prêt au point de vente ou d'un prêt en espèces précédent, et un seul prêt précédent peut avoir plusieurs lignes.

credit_card_balance : données mensuelles sur les cartes de crédit que les clients ont eu avec l'entreprise. Chaque ligne correspond à un mois de solde de carte de crédit, et une seule carte de crédit peut avoir plusieurs lignes.

installments_payment : historique des paiements pour les prêts précédents. Il y a une ligne pour chaque paiement effectué et une ligne pour chaque paiement manqué.

La base de données nommée « application_train », nous a servi à entraîner notre modèle.

2-Modélisation

Après avoir exploré les données, nous avons effectué le preprocessing avec l'aide d'un pipeline puis nous avons choisi d'entrainer plusieurs algorithmes de classification : le dummy classifier, la regression logistique, le ridgeCV et le LightGBM.

Avant de passer à l'entrainement des modèles, une étape indispensable a été de séparer notre jeu d'entraînement en deux : un sous-échantillon d'entraînement regroupant 70% des données et un autre avec 30% des données pour la validation. Chaque sous-échantillon contient le même mélange d'exemples par classe, c'est-à-dire environ 92% de classe 0 et 8% de classe 1.

Cependant, les clients en difficulté de paiement étant largement sous-représentés (8.1%) dans les données d'entraînement, nous ne pouvions passer directement à la phase d'entrainement des algorithmes. En effet,

les méthodes de machine learning classiques ne sont pas toujours adaptées pour la classification sur des données déséquilibrées. Elles donnent souvent de mauvais résultats et, pire encore, elles peuvent induire en erreur avec des scores trop optimistes. C'est pourquoi il a fallu se montrer vigilant dans l'élaboration du modèle, concernant ce déséquilibre. Pour ce faire, nous avons utilisé l'approche d'échantillonnage de SMOTE (Synthetic Minority Oversampling Technic ou suréchantillonnage minoritaire synthétique). Le principe est que plutôt réduire la taille de la classe majoritaire (0), on cherche à augmenter celle de la classe minoritaire (1). Le résultat du rééquilibrage sur les données d'entrainement a été le suivant :

La classe 0 : 28367

La classe 1 : 28367

Ainsi, pour l'entrainement des modèles, nous avons utilisé une fonction qui entraine le modèle sur l'échantillon d'entrainement Nous avons généré les courbes d'apprentissage relatif à chaque modèle et avons évalué les modèles candidats à l'aide d'une validation croisée stratifiée répétée k-fold.

A l'issue de la phase de d'entrainement (avec le *recall* comme scoring d'optimisation) puis de la phase de prédiction sur l'échantillon de test, nous avons obtenu les rapports de classification suivants pour chacun des modèles :

```
ridge
              precision    recall  f1-score   support

           0       0.95      0.63      0.76     12140
           1       0.14      0.64      0.22      1079

    accuracy                           0.63     13219
   macro avg       0.54      0.64      0.49     13219
weighted avg       0.89      0.63      0.72     13219

--------------------------------------------------
logistic
              precision    recall  f1-score   support

           0       0.95      0.64      0.76     12140
           1       0.14      0.65      0.23      1079

    accuracy                           0.64     13219
   macro avg       0.55      0.64      0.49     13219
weighted avg       0.89      0.64      0.72     13219

--------------------------------------------------
dummy
              precision    recall  f1-score   support

           0       0.92      1.00      0.96     12140
           1       0.00      0.00      0.00      1079

    accuracy                           0.92     13219
   macro avg       0.46      0.50      0.48     13219
weighted avg       0.84      0.92      0.88     13219

--------------------------------------------------
lgbm
              precision    recall  f1-score   support

           0       0.92      1.00      0.96     12140
           1       0.32      0.02      0.04      1079

    accuracy                           0.92     13219
   macro avg       0.62      0.51      0.50     13219
weighted avg       0.87      0.92      0.88     13219

--------------------------------------------------
```

Figure 1: Rapport de classification des algorithmes évalués

Comme l'**accuracy** ne distingue pas le type des erreurs commises, elle est souvent complétée par deux indicateurs : le recall et la précision.

Le recall c'est la proportion de la classe positive détectée (compris entre 0 et 1). Un fort recall indique donc que presque tous les cas de la classe positive ont été détectés, par exemple qu'une très grande partie des patients réellement malades ont été classés comme tels.

La précision indique la proportion des vrais positifs dans l'ensemble des positifs détectés (aussi comprise entre 0 et 1). Elle permet d'analyser, dans les cas qui sont sortis positifs par le modèle, quelle proportion l'est réellement.

Selon les problématiques métiers, il peut être intéressant de ne pas choisir un modèle selon son accuracy mais selon son rappel et/ou sa précision.

Enfin, le but du **F1-score** est de donner un unique indicateur qui prenne en compte le rappel et la précision, sans tomber dans les pièges de l'accuracy. Il se définit par la moyenne harmonique de la précision et du rappel. Sa valeur varie de 0 à 1. Un score de 1 indique une précision et un rappel de 100 %.

Le Dummy classifier a servi de baseline afin de comparer les résultats des algorithmes entraînés. Ce classificateur dit naïf effectue des prédictions qui ignorent les entités d'entrée.

Pour établir le scoring nous avons décidé d'optimiser la régression logistique, le RidgeCv et le LightGBM et à partir des meilleurs résultats obtenus, nous avons sélectionné le LightGbm qui est un algorithme basé sur les arbres de décision et dont la particularité est de développer l'arbre verticalement, feuilles par feuilles au lieu de niveau par niveau. Il a pour avantage de converger plus rapidement. Les meilleurs hyperparamètres qui ont servi à l'optimiser avec le RandomizerSearchcv sont :

'subsample': 1, 'num_leaves': 80, 'n_jobs': -1, 'n_estimators': 300, 'max_depth': 12, 'learning_rate': 0.1, 'colsample_bytree': 0.5

3-Choix de la fonction de coût

Afin de discriminer correctement les faux négatifs nous avons une fonction cout métier, qui pondère chaque faux négatif à -10, chaque vrai négatif à 1, chaque faux positif à -1 et chaque vrai positif à 0.

Dans notre problème les faux négatifs sont plus dommageables que les faux positifs. Les faux négatifs sont des cas où des mauvais clients sont classés comme étant des bons clients et se voient accorder un prêt alors que les faux positifs sont des cas où des bons clients sont classés comme étant des mauvais clients et la société de crédit ne lui accorde pas de prêt. Les faux négatifs lui coûtent donc plus cher à la banque que les faux positifs.

Nous nous sommes intéressé à la courbe ROC AUC qui est un graphique représentant les performances d'un modèle de classification pour tous les seuils de classification. Cette courbe trace le taux de vrais positifs en fonction du taux de faux positifs. L'aire sous la courbe (AUC) s'explique par son nom ! Il s'agit simplement de l'aire sous la courbe ROC. (Il s'agit de l'intégrale de la courbe). Cette métrique est comprise entre 0 et 1, un meilleur modèle obtenant un score plus élevé. Un modèle qui se contente de deviner au hasard aura une AUC ROC de 0,5.

En mesurant un classificateur selon l'AUC ROC, nous ne générons pas des prédictions 0 ou 1, mais plutôt une probabilité entre 0 et 1. Les mesures comme le ROC AUC ou le score F1 sont utilisées pour refléter plus précisément les performances d'un classificateur.

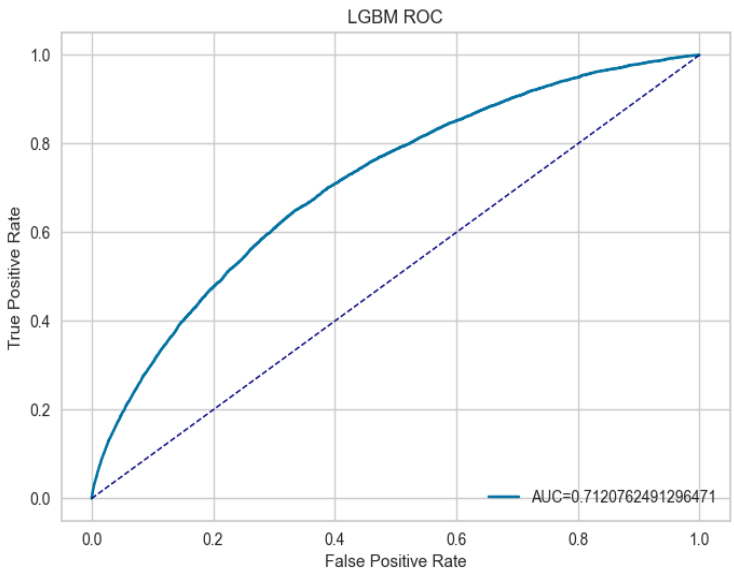


Figure 2: Courbe ROC du LGBM

A travers la courbe ROC nous pouvons voir les performances du modèle. La ligne en pointillée représente la courbe ROC d'un classificateur purement aléatoire. Un bon classificateur s'en écarte autant que possible (vers le coin supérieur gauche).

Nous pouvons voir qu'il existe un certain nombre de points ou de seuils proches du coin supérieur gauche du graphique. Le score AUC est égal à 0.6789.

Afin de minimiser le coût d'erreur de prédiction des Faux Négatifs et des Faux Positifs, nous avons également optimisé du seuil qui détermine, à partir d'une probabilité, la classe 0 ou 1. Si la différence entre la probabilité de la classe 1 et de la classe 0 est supérieur ou égale à ce seuil, le client est classé comme étant à risque. Ci-dessous, nous pouvons jeter un œil aux deux matrices de confusion du LGBM, la première sans le seuil de probabilité et la seconde tenant compte du seuil de probabilité pour la prédiction :

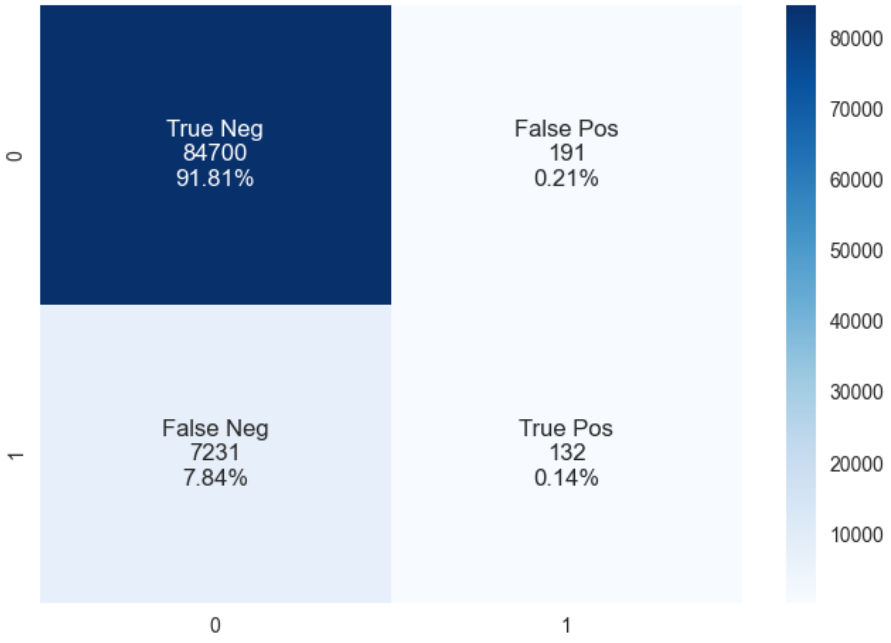


Figure 3: Confusion matrix du LGBM

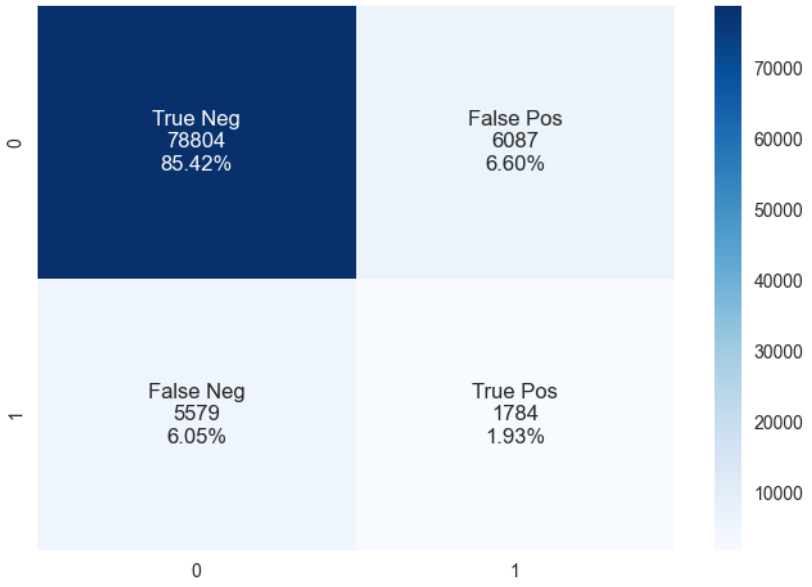
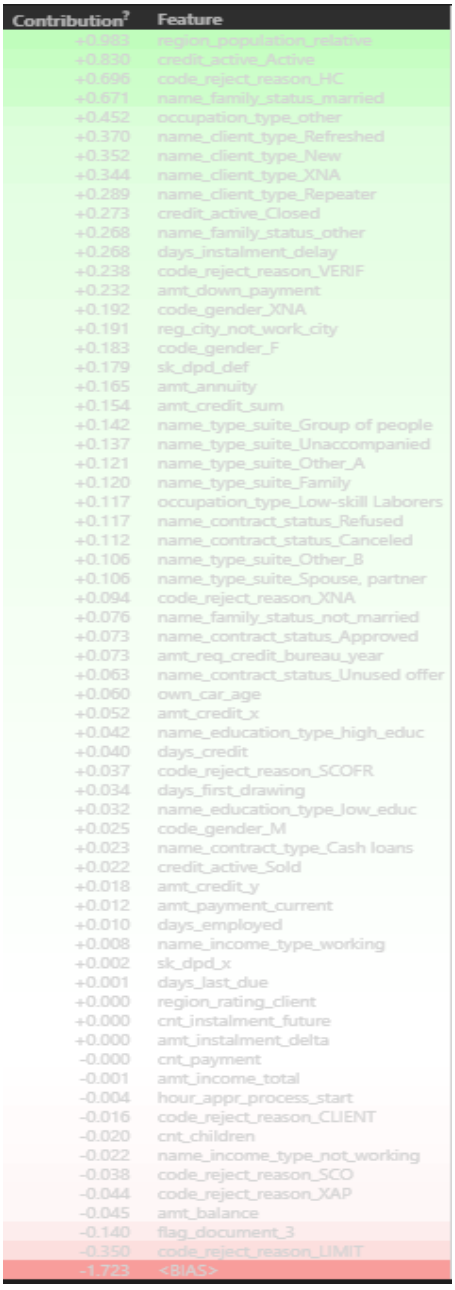


Figure 4: Confusion matrix du LGBM avec le seuil optimal

Le seuil optimal a amélioré la classification des clients et on observe une réduction des faux négatifs par rapport au rapport de classification précédent.

FEATURE IMPORTANCE LOCALE ET GLOBALE



Contribution	Feature
+0.983	region_population_relative
+0.830	credit_active_Active
+0.696	code_reject_reason_HC
+0.671	name_family_status_married
+0.452	occupation_type_other
+0.370	name_client_type_Refreshed
+0.352	name_client_type_New
+0.344	name_client_type_XNA
+0.289	name_client_type_Repeater
+0.273	credit_active_Closed
+0.268	name_family_status_other
+0.268	days_instalment_delay
+0.238	code_reject_reason_VERIF
+0.232	amt_down_payment
+0.192	code_gender_XNA
+0.191	reg_city_not_work_city
+0.183	code_gender_F
+0.179	sk_dpd_def
+0.165	amt_annuity
+0.154	amt_credit_sum
+0.142	name_type_suite_Group of people
+0.137	name_type_suite_Unaccompanied
+0.121	name_type_suite_Other_A
+0.120	name_type_suite_Family
+0.117	occupation_type_Low-skill Laborers
+0.117	name_contract_status_Refused
+0.112	name_contract_status_Canceled
+0.106	name_type_suite_Other_B
+0.106	name_type_suite_Spouse, partner
+0.094	code_reject_reason_XNA
+0.076	name_family_status_not_married
+0.073	name_contract_status_Approved
+0.073	amt_req_credit_bureau_year
+0.063	name_contract_status_Unused offer
+0.060	own_car_age
+0.052	amt_credit_x
+0.042	name_education_type_high_educ
+0.040	days_credit
+0.037	code_reject_reason_SCOFR
+0.034	days_first_drawing
+0.032	name_education_type_low_educ
+0.025	code_gender_M
+0.023	name_contract_type_Cash loans
+0.022	credit_active_Sold
+0.018	amt_credit_y
+0.012	amt_payment_current
+0.010	days_employed
+0.008	name_income_type_working
+0.002	sk_dpd_x
+0.001	days_last_due
+0.000	region_rating_client
+0.000	cnt_instalment_future
+0.000	amt_instalment_delta
-0.000	cnt_payment
-0.001	amt_income_total
-0.004	hour_appr_process_start
-0.016	code_reject_reason_CLIENT
-0.020	cnt_children
-0.022	name_income_type_not_working
-0.038	code_reject_reason_SCO
-0.044	code_reject_reason_XAP
-0.045	amt_balance
-0.140	flag_document_3
-0.350	code_reject_reason_LIMIT
-1.723	<BIAS>

Un chargé de clientèle doit pouvoir utiliser le modèle via l'application mise à disposition, en face à face avec son client et vulgariser la décision envisagée dans l'étude de son dossier. L'importance des variables mesure l'impact global de chaque descripteur dans le modèle. Elle peut être estimée en modélisation (sur l'échantillon d'apprentissage) ou en prédiction (sur l'échantillon test). Pour réaliser ce module, la première perspective envisagée était d'utiliser l'importance des features issues des différents modèles utilisés mais cette approche n'est pas optimale car les variables en sortie de modèle sont difficiles à interpréter lorsqu'elles sont issues de One Hot Encoding. Eli5 fournit un moyen de calculer l'importance des caractéristiques pour tout estimateur en mesurant comment le score diminue lorsqu'une caractéristique n'est pas disponible ; la méthode est également connue sous le nom de « permutation importance.

Ci-contre, voici un exemple de graphique généré avec le module Eli5 pour afficher l'importance locale des variables pour la prédiction de prêt relative au client 100200. Les coefficients de chaque variable contribuant à la prédiction sont indiqués dans la colonne « contribution ».

Les fonctionnalités positives apparaissent comme les fonctionnalités les plus importantes alors que les fonctionnalités avec des importances négatives perturbent le modèle. Quant aux caractéristiques proches de zéro elles contiennent peu ou pas de données utiles.

Ce graphique répond à la question : pourquoi la demande de prêt d'un client X a-t-elle été approuvée ou rejetée ?

Par ailleurs, le graphique de l'importance globale prend la même forme que le graphique de l'importance locale, avec une différence au niveau des colonnes nommées « Weight » et « Feature ».

En tenant compte du fait que le problème traite deux catégories, qu'il est donc binaire, une seule colonne d'attributs est renvoyée à savoir « contribution » et « weight ». Ainsi les poids positifs pour une classe deviennent les négatifs pour l'autre classe.

LES LIMITES ET LES AMELIORATIONS POSSIBLES

Dans un premier temps, mes faibles connaissances du domaine bancaire ont été une première limite pour la réalisation de ce projet que cela concerne le coût métier ou le seuil optimal.

Deuxièmement, dans la vraie vie il serait nécessaire d'échanger avec les équipes métiers afin de déterminer quelles variables sont les plus pertinentes au vu de leur expérience dans leur choix d'accorder un prêt ou non à un client donné. Cela réduirait le nombre de variables à afficher sur les graphiques de feature importance.

D'autre part, pour améliorer le dashboard, il serait bien de proposer une page afin de tester des valeurs aléatoires qui permettrait au client de comprendre sur quelles variables il aurait pu jouer pour obtenir son prêt comme par exemple ajuster le montant du prêt demandé. Aussi, réduire le nombre de variables à montrer au client rendrait plus rapidement lisible le graphique.

Enfin, l'utilisation d'autres algorithmes de boosting donneraient peut-être de meilleurs résultats.