Note méthodologique

Parcours Data Science

OPENCLASSROOMS

Projet n°7

Implémentez un modèle de scoring

Kevin

2022

Sommaire

1. Description Projet

1.1. Cahier des charges

Prêt à dépenser est une organisation financière qui propose des crédits à la consommation pour des clients ayant peu ou pas d'historique de prêt.

Le but de ce projet est de concevoir un modèle de scoring permettant de probabiliser le défaut de paiement d'un client, afin de déterminer si un conseiller doit accepter ou non sa demande de prêt.

1.2. Jeu de données

Dataset

8 fichiers CSV de forme variable. Les deux dataset principaux ont plus de 120 variables et 300.000 lignes.

Les 6 autres ont moins de variables (quinzaine) mais varie d'1,5 à 6m de lignes

Variable TARGET

0 -> Rembourse son prêt / 1 : Ne rembourse pas son prêt

Observations

Jeu de données très déséquilibré (91% du dataset rembourse son prêt)

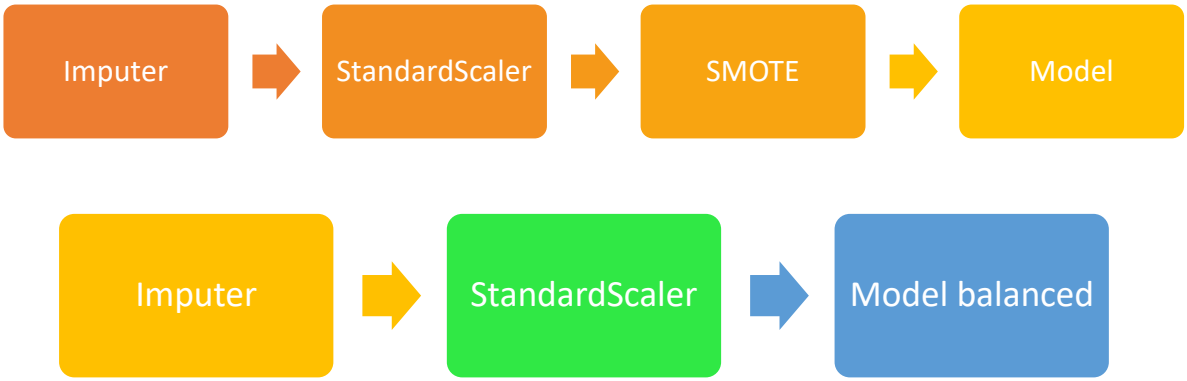
2. Entrainement du modèle

2.1. Selection des données

- Feature engineering avec le Kernel OC + Distinction des features importances
- Analyse exploratoire
- Selection des 40 features les plus importantes (représente 1/3 des features importances)

2.2. Transformation des données

- Test avec plusieurs pipelines :



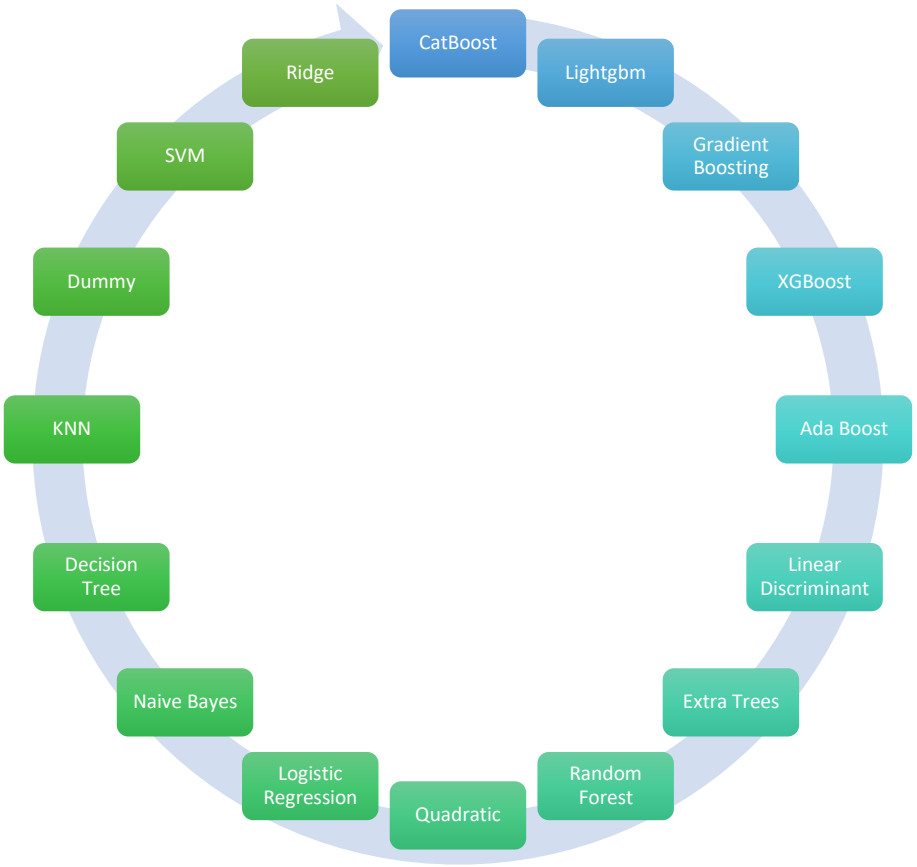
Imputer : Test avec fillna / SimpleImputer (median) / IterativeImputer

Le premier pipeline utilise SMOTE afin de rééquilibrer la variable TARGET, qui est composé à 91% de personnes pouvant rembourser son prêt.

Le deuxième pipeline se base sur un hyperparamètre de balance de poids des classes.

2.3. Selection du modèle

- Test de modèles en se basant sur une métrique personnalisée :

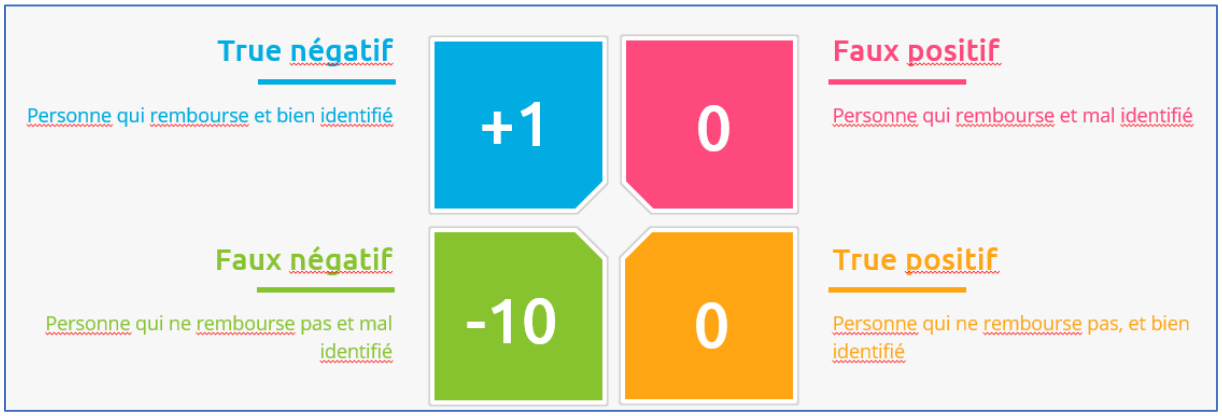


Metrique personnalisée

Pour la fonction métier, le but est de :

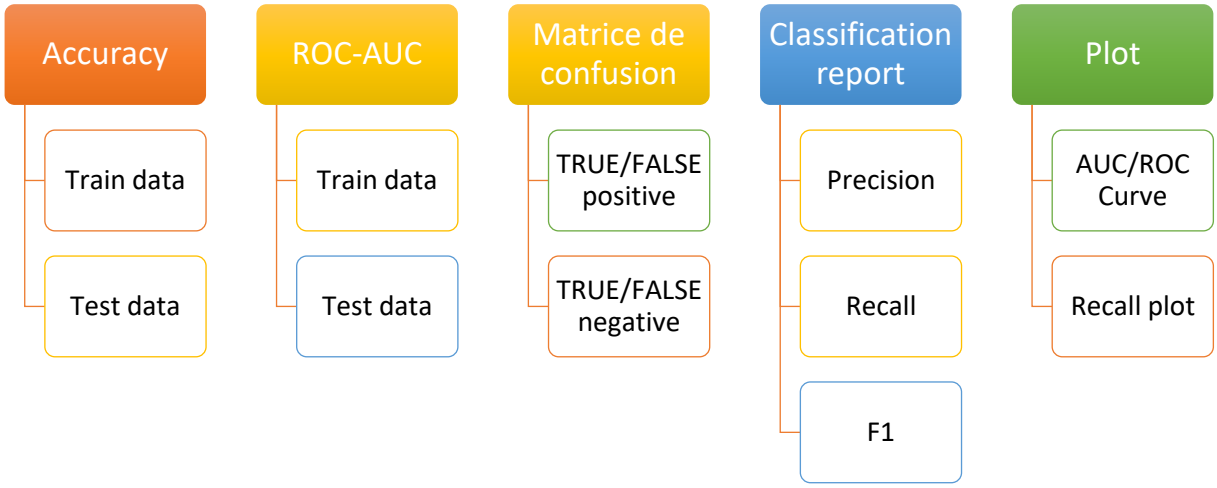
- Minimiser prioritairement les faux négatifs, afin d'éviter d'accorder des prêts à des personnes qui ne pourraient pas rembourser, et qui feraient perdre de l'argent.
- Minimiser les faux positifs, pour éviter de refuser des prêts à des personnes qui auraient pu rembourser

Pour cela, la métrique personnalisée consiste à attribuer un poids à chaque résultat. Le total donne un score, où un FN pénalise lourdement le score, tandis qu'un FN augmente le score .



2.4. Evaluation de l'entrainement

- Selection du modèle le plus performant
- Split de la data avec une partie entrainement et une partie test



3. Optimisation

3.1. Optimisation

On optimise les hyperparamètres de l'algorithme avec GridSearchCV ou RandomizedSearchCV, en se basant toujours sur la métrique personnalisée et la même procédure d'évaluation.

4. Visualisation

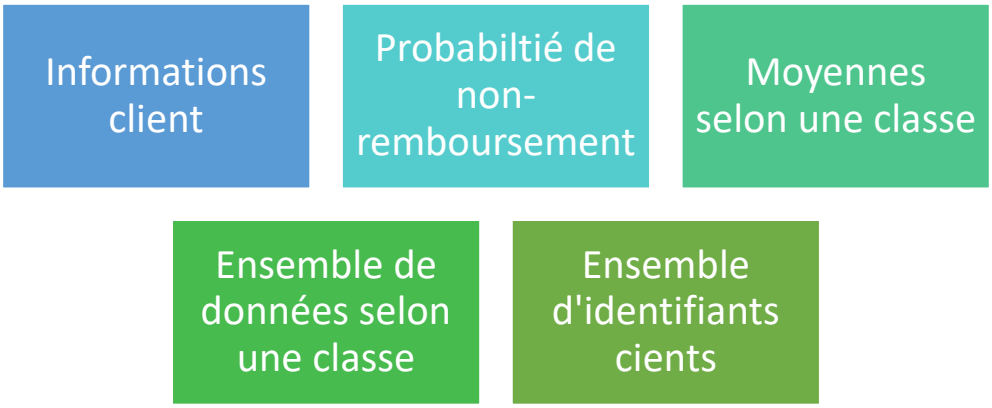
4.1. Explication

Shap ou Feature importance ?

5. Rendu final

5.1. API

Utilisation de FastAPI avec plusieurs requêtes possibles, à partir de l'identifiant client ou non :



5.2. Dashboard

Utilisation de Streamlit avec 4 modules qui font appel à l'API :

Infos clients

- Les informations principales du client et son scoring

Comparaison

- Comparaison avec des clients qui remboursent / ne remboursent pas

Shap

- Explication des résultats de l'algorithme

Data brute

- Voir la data "brute" du client

5.3. Livrables

API : https://kevin-oc-api.herokuapp.com/docs

Dashboard : https://kevin-oc.herokuapp.com/

6. Limites et améliorations

Poids à attribuer

- Etudier avec le client les "poids" qu'il souhaite pour l'optimisation de l'algorithme avec la métrique personnalisée. Ma façon de faire a été totalement arbitraire, et ne satisfera peut-être pas sa demande.

Dataset complexe

- Beaucoup de manipulation ont dû être effectués : Equilbrage des classes, nouvelles variables, sélection de variables, création d'une métrique adaptée à la problématique métier....

Pas d'avis extérieur

- Difficile à faire sans l'avis d'expert du milieu

Explication des variables

- Toutes les variables ne sont pas correctement expliquées dans le fichier de description fourni