Implémentez un modèle de scoring

Data Science | Projet 7

Fırat Yasar

28/01/2022

Sommaire

Présentation

Présentation de la problématique

L'objectif :

Notre objectif est d'aider à la société financière **Prêt à dépenser** de minimiser les risques sur la decision des chargé de relation client d'accorder ou refuser un prêt à la consommation.

La mission:

Mettre en œuvre **un modèle de "scoring crédit"** de la probabilité qu'un client rembourse son crédit, puis classifie la demande en crédit accordé ou refusé.

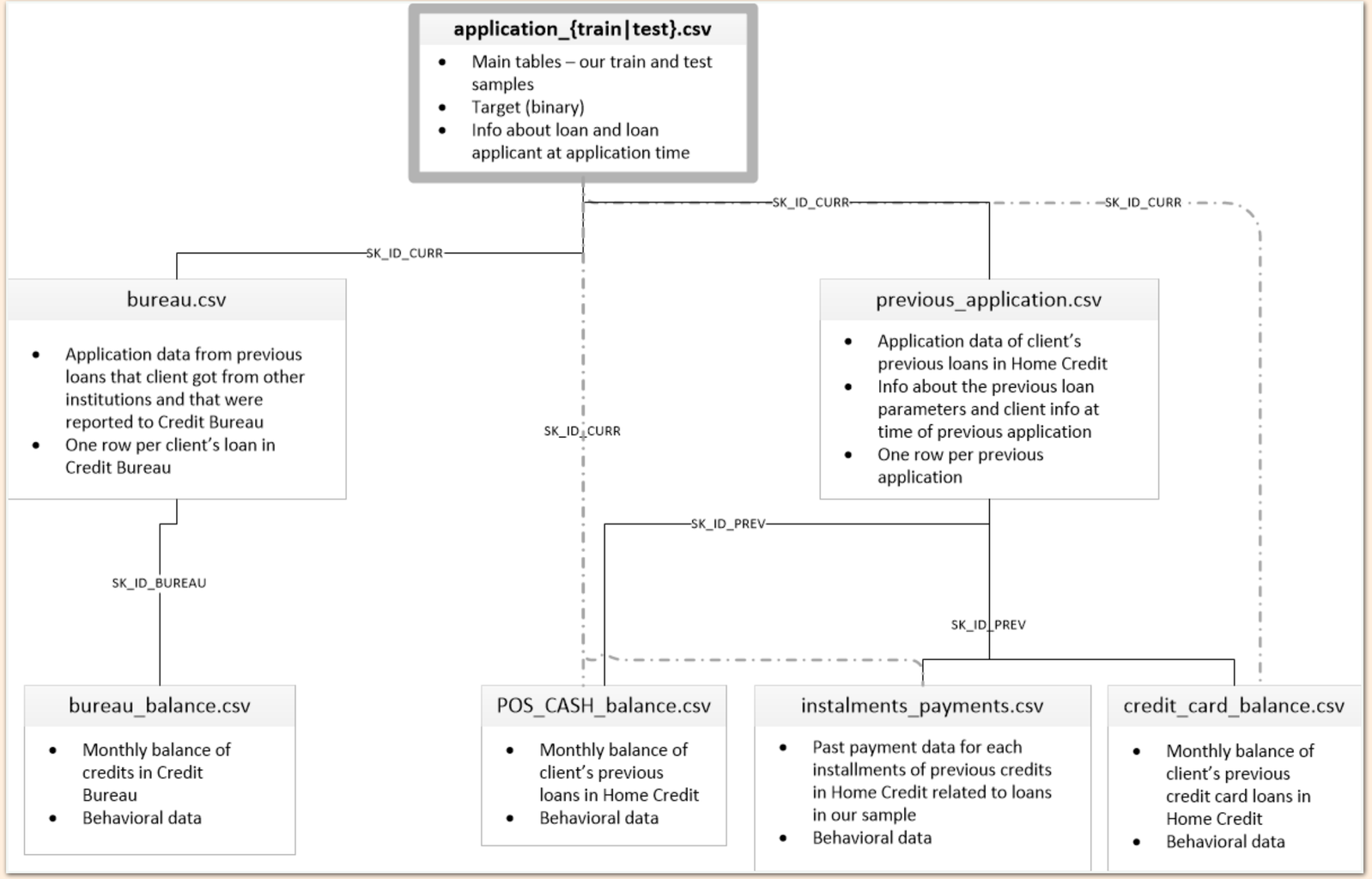
Construire **un dashboard interactif** à destination des gestionnaires de la relation client permettant d'interpréter les prédictions faites par le modèle, et d'améliorer la connaissance client des chargés de relation client.

La base de données :

Pour cela, nous avons à notre disposition la base de données :

https://www.kaggle.com/c/home-credit-default-risk/data

Découverte du jeu de données



application_train
307 511 lignes 122 colonnes

application_test
48 744 lignes 121 colonnes

Pas de TARGET

TARGET = 0 (succès)

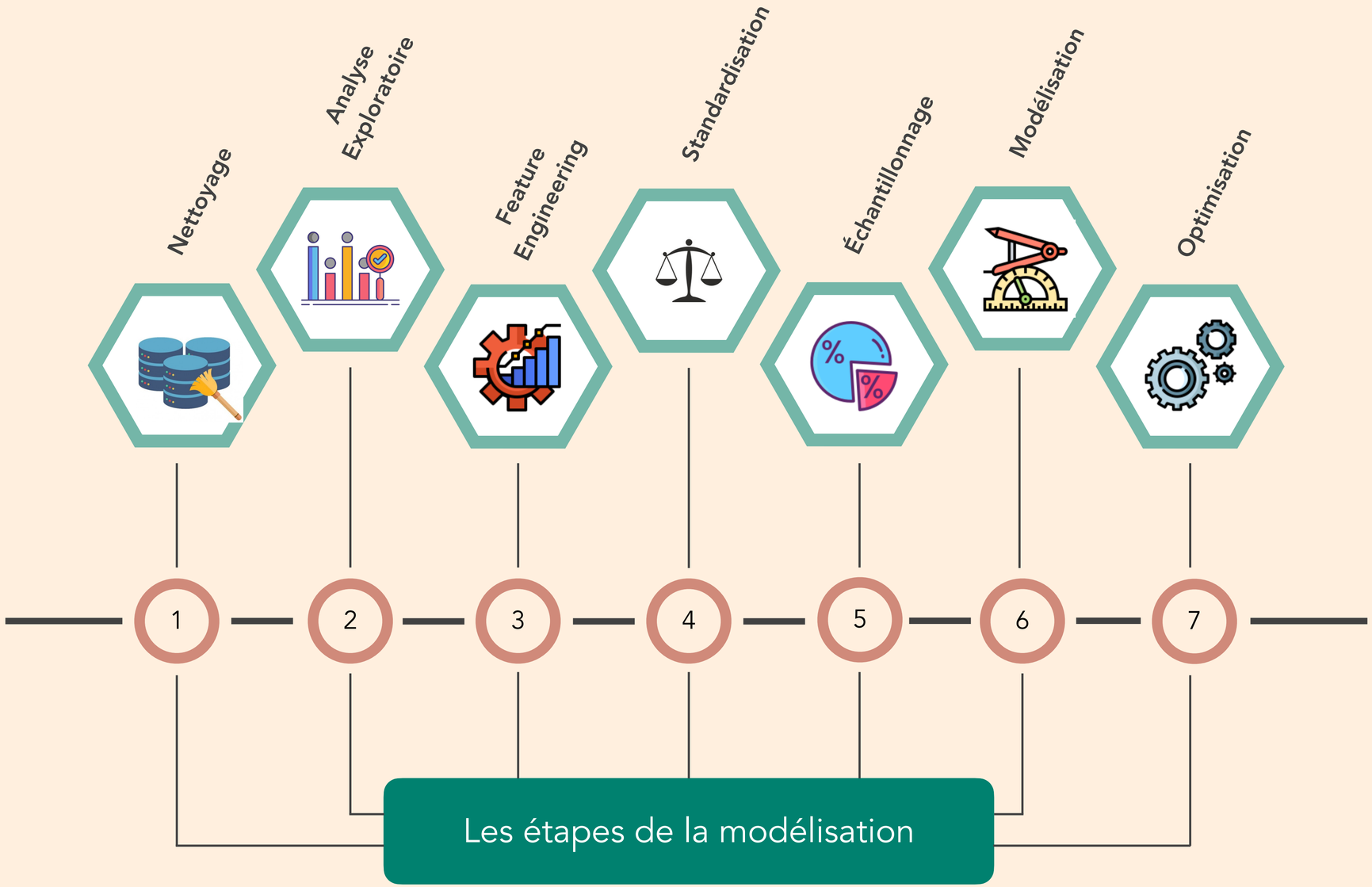
TARGET = 1 (faillite)

une ligne = un crédit accordé

Les étapes de la modélisation

Supervisée

Classification


Nettoyage
Analyse Exploratoire
Feature Engineering
Standardisation
Échantillonnage
Modélisation
Optimisation
1
2
3
4
5
6
7
Les étapes de la modélisation

Nettoyage des données

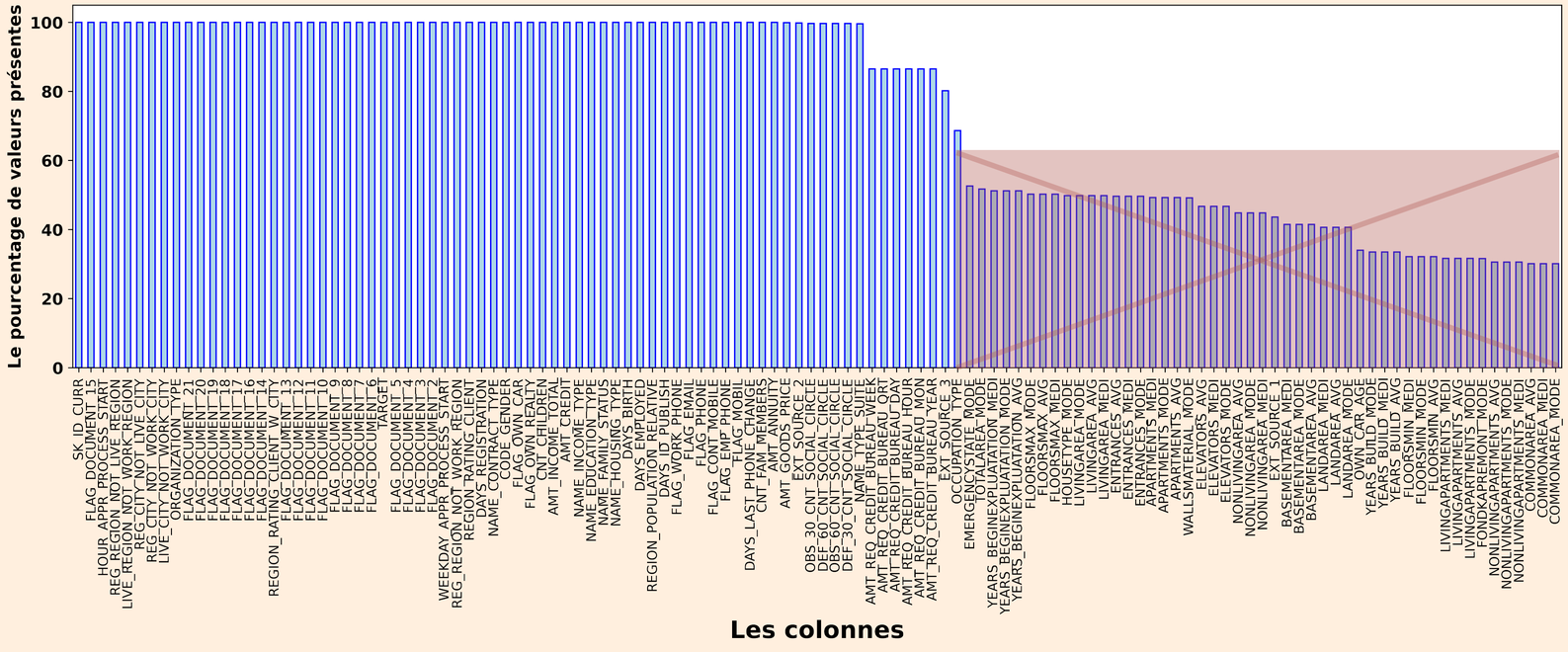
- **Les valeurs manquantes**

  Nous avons supprimé des variables qui comporte plus de 60 % de valeurs manquantes.

- **Les valeurs aberrante**

  18 % des valeurs de la variable “DAYS_EMPLOYED” sont fixés à 365 243 jours, soit 1000 ans de jours de travail. Nous avons remplacé cette valeurs impossibles par la médiane de la colonne.

Le pourcentage de remplissage par colonne



Features engineering

- **Création de deux groupes des features**

Polynomial features : Ce groupe des variables compte les combinaisons polynomiales de trois features *EXT_SOURCE_{1,2,3}*.

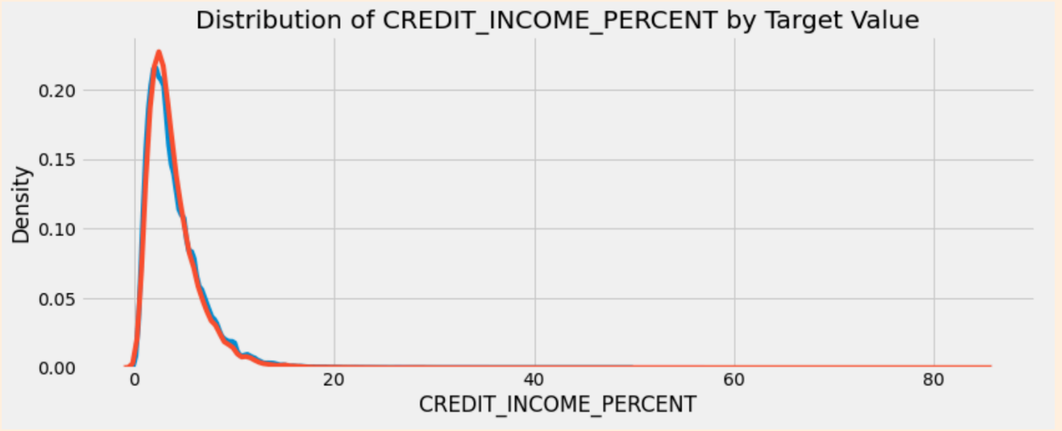
→ 35 nouveau features

```
EXT_SOURCE_2 EXT_SOURCE_3                 -0.193939
EXT_SOURCE_1 EXT_SOURCE_2 EXT_SOURCE_3    -0.189605
EXT_SOURCE_2 EXT_SOURCE_3 DAYS_BIRTH      -0.181283
EXT_SOURCE_2^2 EXT_SOURCE_3               -0.176428
EXT_SOURCE_2 EXT_SOURCE_3^2               -0.172282
EXT_SOURCE_1 EXT_SOURCE_2                 -0.166625
EXT_SOURCE_1 EXT_SOURCE_3                 -0.164065
EXT_SOURCE_2                              -0.160295
EXT_SOURCE_2 DAYS_BIRTH                   -0.156873
EXT_SOURCE_1 EXT_SOURCE_2^2               -0.156867
Name: TARGET, dtype: float64
DAYS_BIRTH      -0.078239
DAYS_BIRTH^2    -0.076672
DAYS_BIRTH^3    -0.074273
TARGET           1.000000
```

Domain knowledge features : Ces variables ont été construites s'appliquant plus au domaine du secteur bancaire.

- *"CREDIT_INCOME_PERCENT"*
- *"ANNUITY_INCOME_PERCENT"*
- *"CREDIT_TERM"*
- *"DAYS_EMPLOYED_PERCENT"*

→ 4 nouveau features



Credits : https://www.kaggle.com/willkoehrsen/start-here-a-gentle-introduction

Preprocessing

Encodage des variables catégorielles

Label Encoder pour les variables à 2 catégories.

One-hot Encoder pour les variables à plus de 2 catégories.

Standardisation

Nous avons remplacé les outliers par des valeurs nulles, ensuite les imputés par la médiane de la colonne :

SimpleImputer (strategy = 'median')

Nous avons remplacé les outliers par des valeurs nulles, ensuite les imputés par la médiane de la colonne :

MinMaxScaler(feature_range = (0, 1))

Échantillonnage

Split du jeu des données en deux parties :

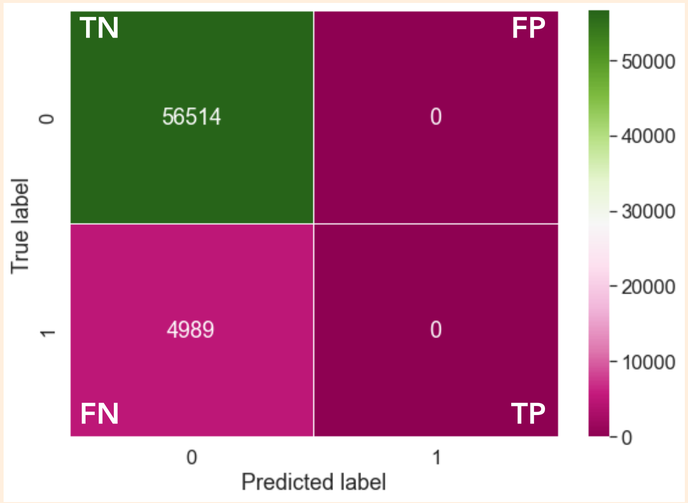
- données d'entrainement
- données de test

train_test_split(X, y, test_size=0.2, random_state=4)

Modélisation : Baseline

Baseline

Utilisation d'une régression logistique comme baseline

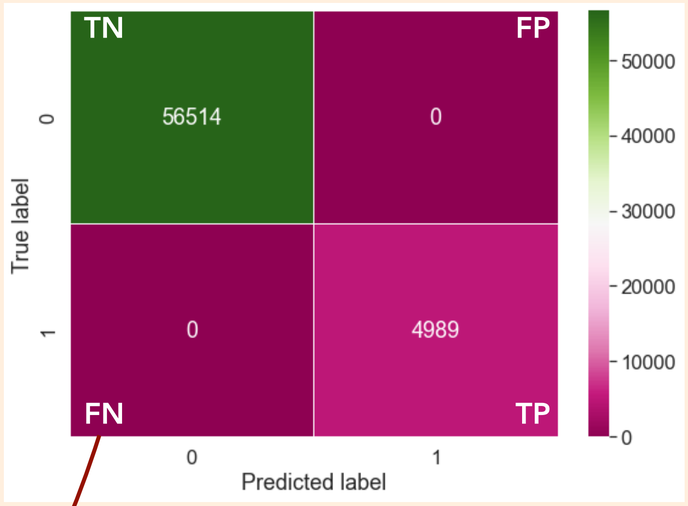


	modele	accuracy_score	precision_score	recall_score	f1_score
0	Régression Logistique	0.92	0.0	0.0	0.0

- TP = client solvable, identifié correctement
- TN = client non-solvable, identifié correctement
- FP = client solvable, identifié en tant que non-solvable
- FN = client non-solvable, identifié en tant que solvable

Le scénario idéal

Pas de FN et pas de FP

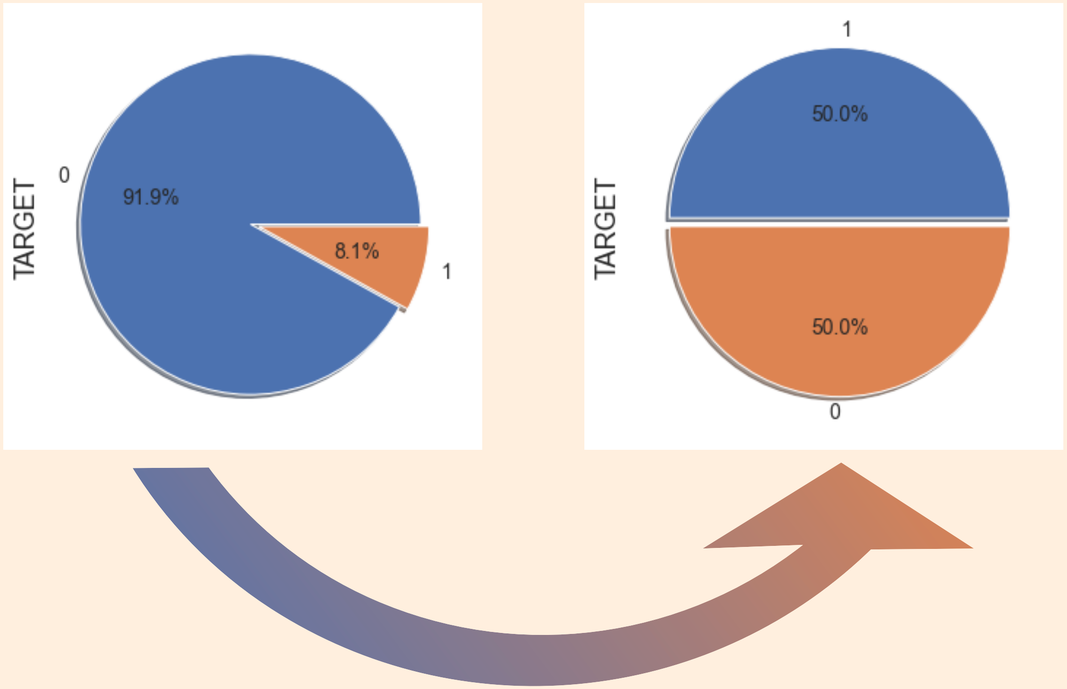


L'essentiel est de limiter les faux négatives pour éviter la perte de rentabilité

Resampling

Le problème du déséquilibre

Les deux modalités de la variable cible *"TARGET"* ne sont pas représentées de façon égale dans l'échantillon. Cela peut fausser l'utilisation du model d'apprentissage.

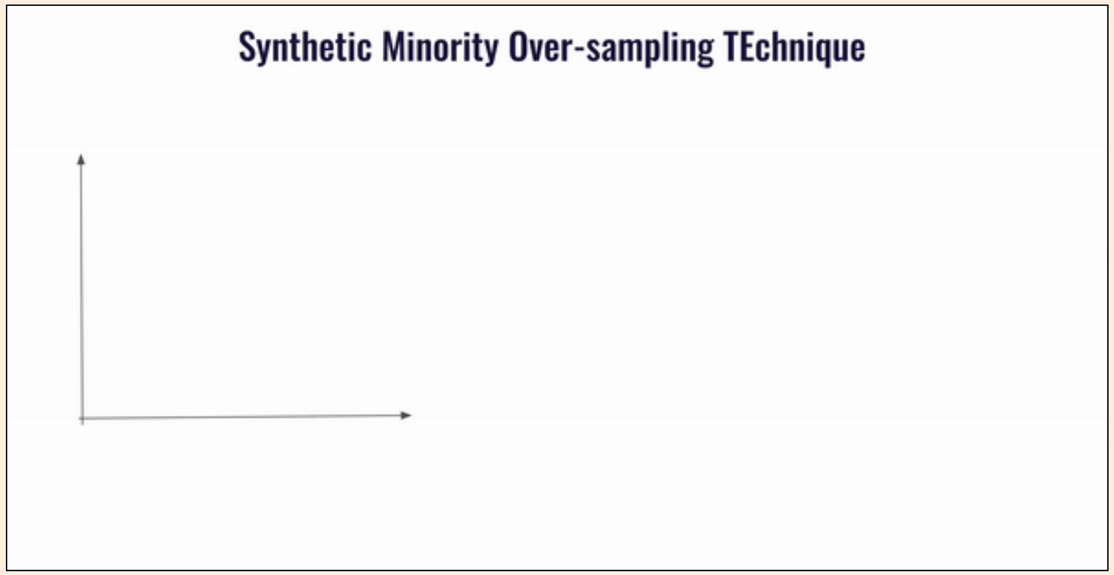


La solution : SMOTE

Ré-échantillonner les données pour se rapprocher d'une situation d'équilibre. Nous avons testé deux méthode de ré-échantillonnage :

- SMOTE (**S**ynthetic **M**inority **O**versampling **TE**chnique)
- **RandomUnderSampler** de la librarie *imblearn*

	modele	accuracy_score	precision_score	recall_score	f1_score
0	Rég. Logistique (SMOTE)	0.69	0.15	0.62	0.24
1	Rég. Logistique (RandomUnderSampler)	0.61	0.13	0.65	0.21
2	Régression Logistique	0.92	0.00	0.00	0.00



Evaluation des modèles de classification

- **Performance**

Matrice de confusion : comparer la classe prédite avec la classe réelle.
Accuracy : la proportion de points (positif ou négatif) correctement prédits.
Précision : la proportion de prédictions correctes parmi les points que l'on a prédits positifs.
Rappel (Recall): la proportion de positifs que l'on a correctement identifiés.
F1 : moyenne harmonique du rappel et de la précision.

meilleur score = plus élevé

Le courbe ROC AUC et le courbe Precision-Recall

- **Interprétabilité**

Le modèle est plus performant avec l'ensemble de features polynomiales

Sélection de la meilleure hypothèse

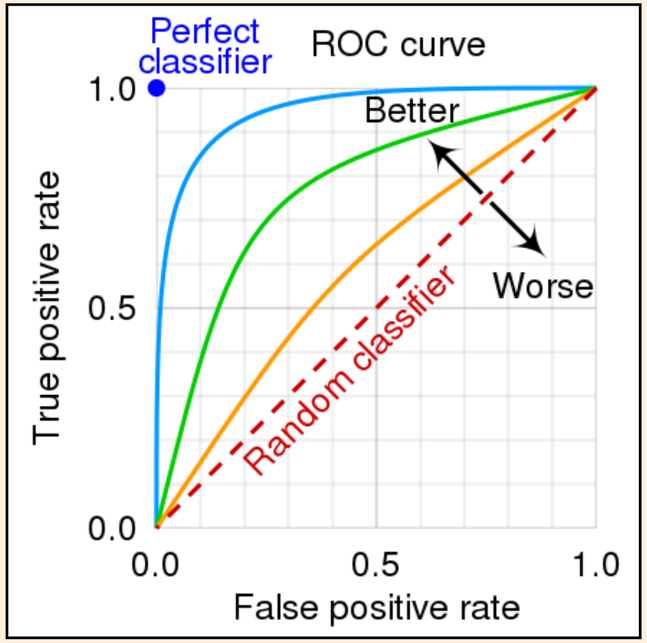
	modele	accuracy_score	precision_score	recall_score	f1_score
0	Rég. Logistique (POLY_SMOTE)	0.69	0.16	0.66	0.25
1	Rég. Logistique (SMOTE)	0.69	0.15	0.62	0.24
2	Rég. Logistique (DOMAIN_SMOTE)	0.69	0.15	0.62	0.24
3	Rég. Logistique (RandomUnderSampler)	0.61	0.13	0.65	0.21
4	Régression Logistique	0.92	0.00	0.00	0.00

$$Accuracy = \frac{TN + TP}{TN + FP + TP + FN}$$

$$Précision = \frac{TP}{TP + FP}$$

$$Rappel = \frac{TP}{TP + FN}$$

$$F_{score} = 2 \times \frac{Précision \times Rappel}{Précision + Rappel}$$



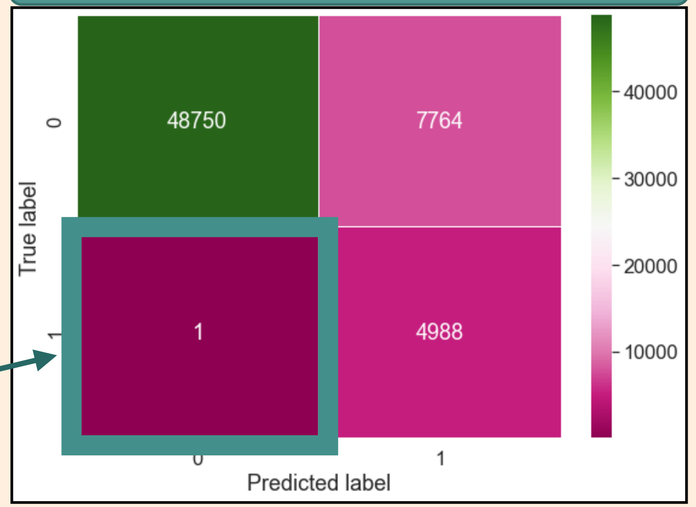
Credits : https://openclassrooms.com/fr/courses/4297211-evaluez-les-performances-dun-modele-de-machine-learning/4308256-evaluez-un-algorithme-de-classification-qui-retourne-des-valeurs-binaires

Modélisation : Analyse des résultats I

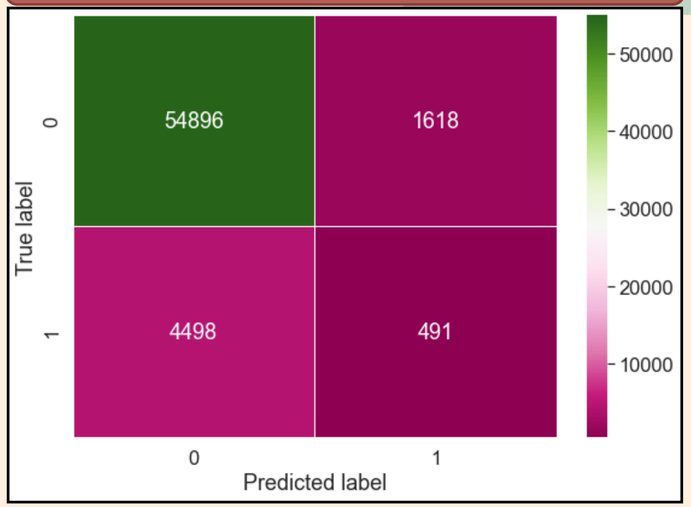
- **Matrice de confusion**

Les performances de KNN et XGBoost sont bons en termes des faux négatives.

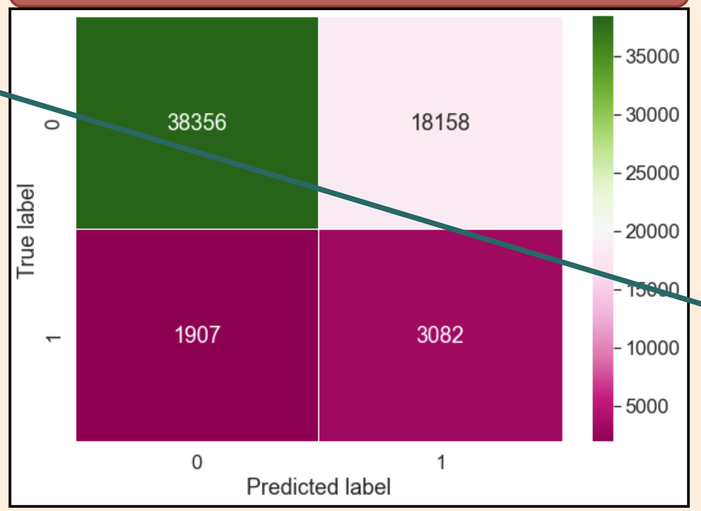
KNN (K plus proches voisins)



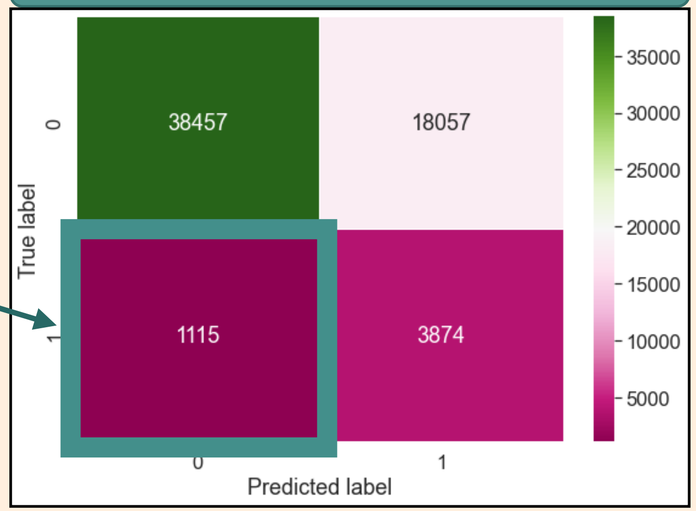
Gradient Boosting Classifieur



Random Forest Classifieur



XGB (eXtreme Gradient Boosting trees)



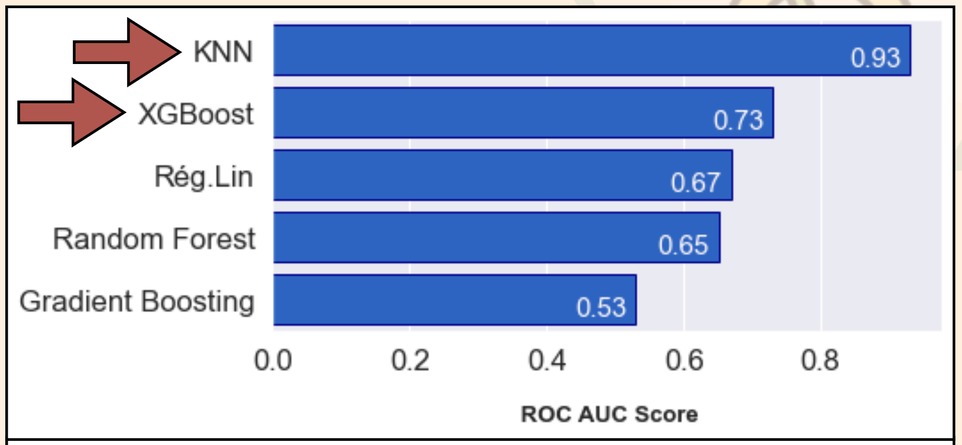
Modélisation : Analyse des résultats II

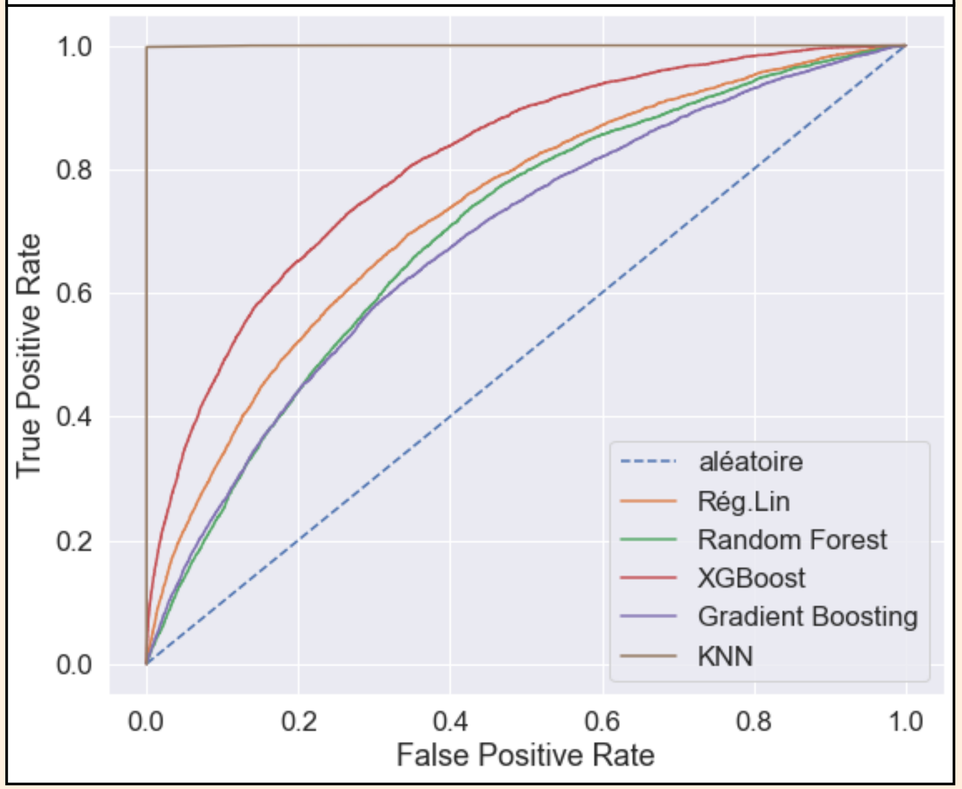
- **Performance**

KNN : les résultats sont bons mais ce modèle ne permet pas d'inspecter l'importance des variables

XGBoost : les résultats sont généralement bons mais *précision* et *F1* reste faible.

	modele	accuracy_score	precision_score	recall_score	f1_score
0	KNN	0.87	0.39	1.00	0.56
1	XGBoost	0.69	0.18	0.78	0.29
2	Random Forest	0.67	0.15	0.62	0.24
3	Gradient Boosting	0.90	0.23	0.10	0.14



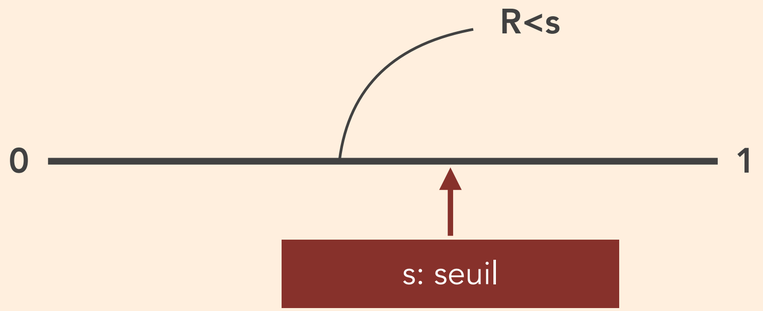


Optimisation

Le seuil de décision

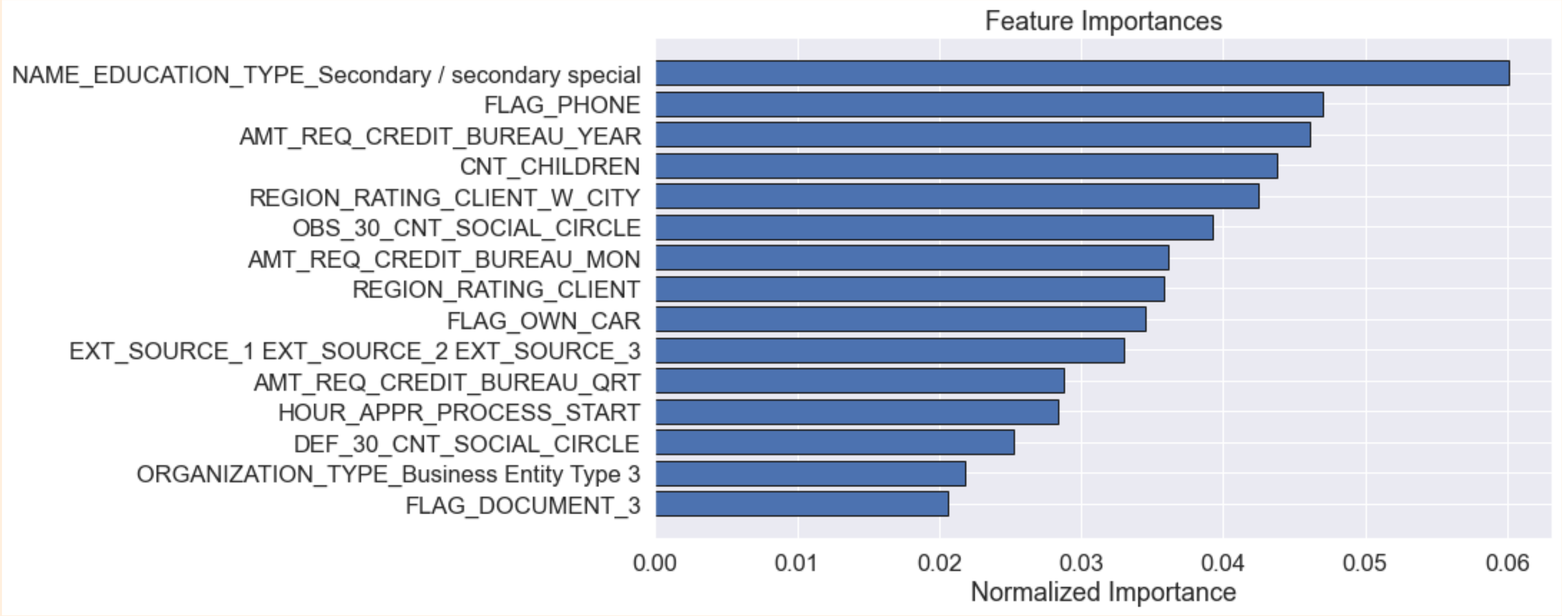
Il s'agit d'un nombre réel **s** que nous fixons qui détermine qu'une valeur appartient à la classe 0 ou à la classe 1.

Utilisation de la méthode ***predict_proba*** afin de prédire la probabilité de faillite d'un crédit.



L'importance des variables

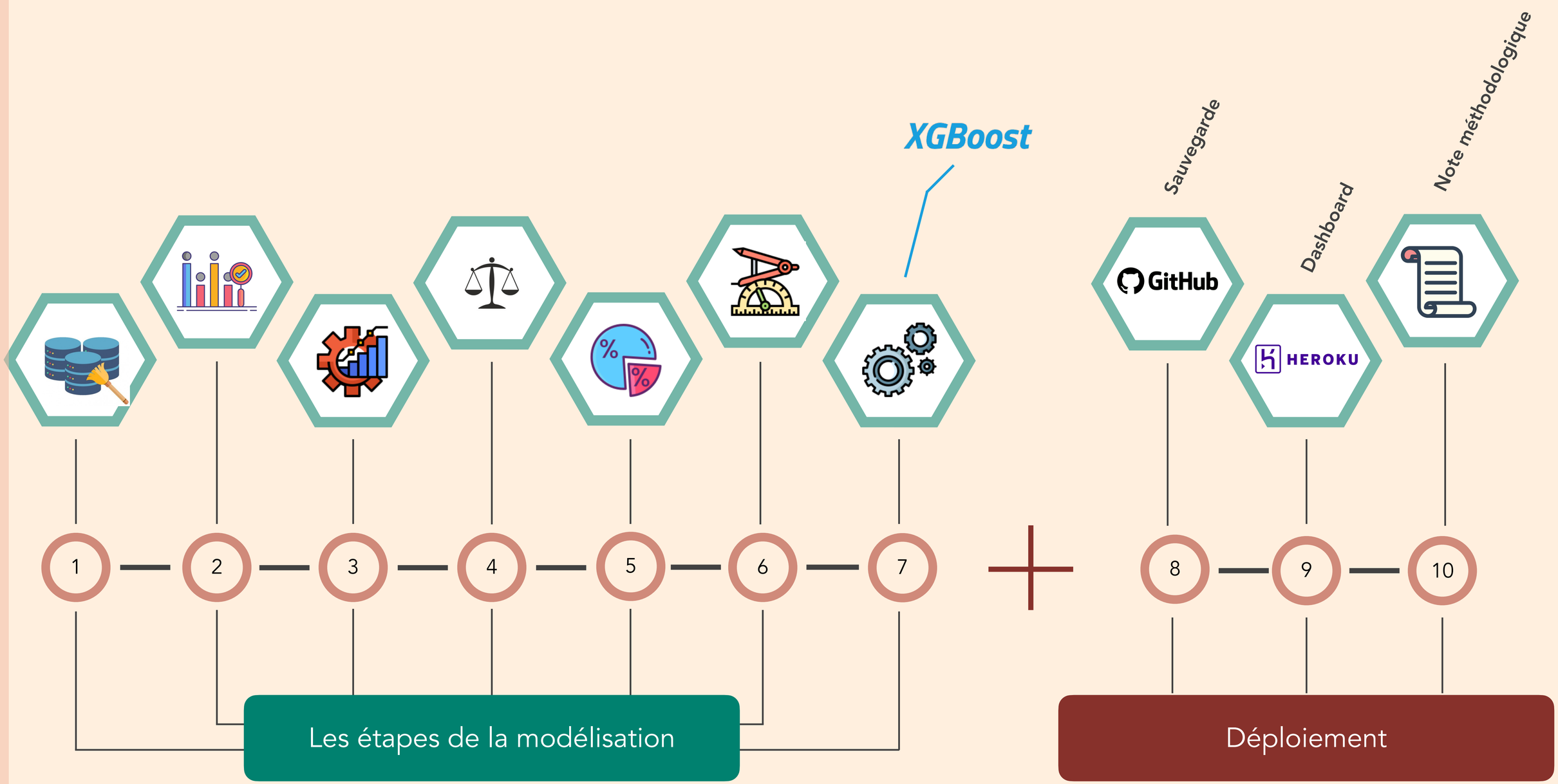
Le modèle XGBoost permet d'observer l'importance des variables pour les prédictions



Le modèle retenu

XGBoost

Déploiement sur le cloud


XGBoost
Sauvegarde
Dashboard
Note méthodologique
GitHub
HEROKU
1
2
3
4
5
6
7
8
9
10
Les étapes de la modélisation
Déploiement

Présentation du Dashboard

- **Création du fichier *P7_Dashboard.py* via Streamlit**

  Le fichier python contenant le code de l'interface à la disposition des clients.

  Local URL: http://localhost:8501

Affichage du dashboard en locale
*
Créer le fichier
`P7_Dashboard.py`

Ouvrir le terminal et écrire
`cd /Users/fyasar/Desktop/P7`

Ensuite,
`streamlit run P7_Dashboard.py`

- **Le lien du dépôt GitHub:**

  Le dépôt distant permet de stocker les différentes versions du code afin de garder un historique hébergé sur le réseau

  https://github.com/yasarigno/Projet_7

- **Le lien du dashboard Heroku :**

  Le dashboard interactif est en ligne

  https://projet-7-credit.herokuapp.com

Alternatives

Conclusion

Résumé

- Nous avons construit un modèle de classification binaire à partir d'un notebook de départ téléchargé sur Kaggle.
  https://www.kaggle.com/willkoehrsen/start-here-a-gentle-introduction

- Afin de supprimer de biais dû au déséquilibre entre les classes de la variable cible "TARGET" nous avons utilisé la méthode SMOTE.
- Nous avons obtenu le score **Recall** assez élevé; cependant la **Precision** et **F1** ne sont généralement pas très bons.

Axes d'amélioration

- Le feature engineering ne tient compte que d'un seul dataset sur les 8 disponibles. Dans ce projet, nous n'avons utilisé que les features crées dans le Kernel de départ. Afin d'améliorer les performances, il faut approfondir cette étape en étudiant le secteur financière.

- L'utilisation de réseaux neuronaux pourrait fournir les meilleurs résultats.

Merci de votre attention