OPENCLASSROOMS

Projet 4

Construisez un modèle de scoring

Développerez vos compétences en apprentissage supervisé

David Scanu | Janvier 2025

Parcours **AI Engineer**

⭐ Étapes d'un projet de data science

1. Compréhension de la problématique métier
2. Analyse exploratoire des données (EDA)
3. Nettoyage et prétraitement des données
4. Feature Engineering et sélection
5. Entraînement des modèles
6. Comparaison des performances des modèles
7. Interprétabilité du modèle
8. Conclusion

💶 Compréhension de la problématique métier

Contexte du projet

Pour accorder un crédit à la consommation, l'entreprise "Prêt à dépenser" souhaite mettre en œuvre un outil de "**scoring crédit**" qui calcule la probabilité qu'un client le rembourse ou non, puis classifie la demande : **crédit accordé ou refusé**.

Objectif du projet

- **Développer un algorithme de classification** pour aider à décider si un prêt peut être accordé à un client.
- Utilisateurs : **chargés de relation client**.
- Le modèle de prédiction doit être **facilement interprétable**.

Notre rôle

- **Data Scientist** au sein d'une société financière, nommée "Prêt à dépenser".

🔎 Description du jeu de données

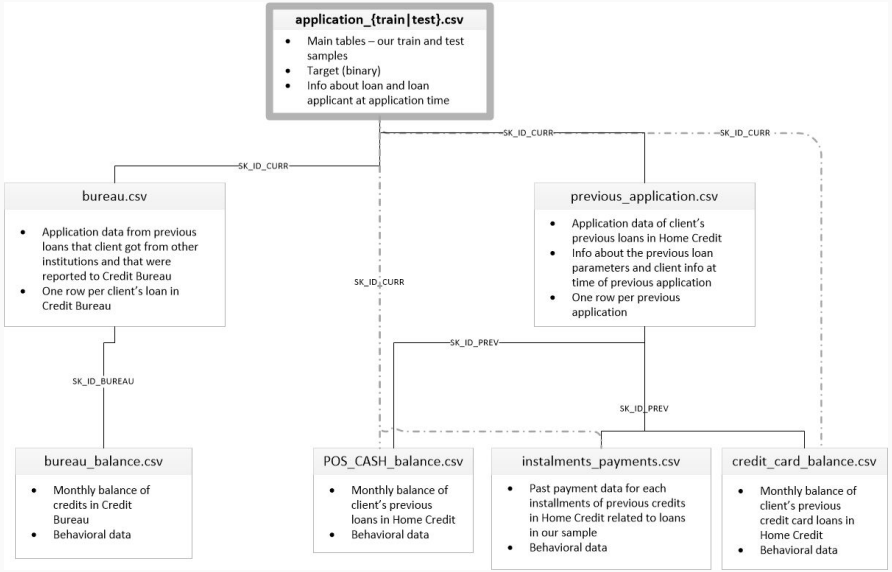
Données fournies par **Home Credit**, un service dédié à l'octroi de lignes de crédit (prêts) à la population non bancarisée.

Il existe 7 sources de données différentes :

- **application_train (lignes : 307 511, variables : 122)**
- application_test
- bureau
- bureau_balance
- previous_application
- POS_CASH_BALANCE
- credit_card_balance
- installments_payment

Deux fichiers additionnels :

- HomeCredit_columns_description : Description des colonnes de la demande de crédit
- sample_submission : Exemple de fichier à soumettre pour la compétition Kaggle.



👀 Compréhension des variables

Essentiel pour :

- Feature engineering
- Prétraitement

Repose sur trois piliers principaux :

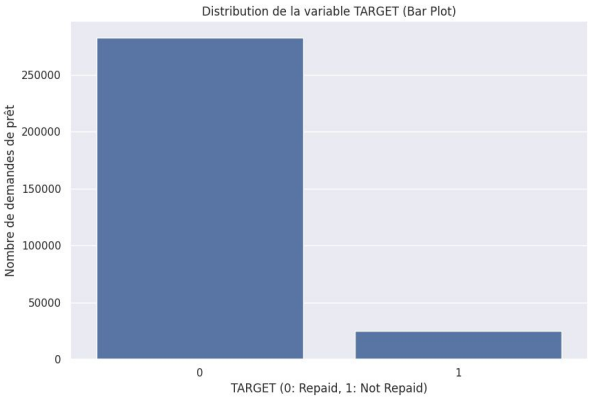
1. Connaissance métier
2. Description des colonnes
3. Observation des données

Types de variables identifiées :

- 🔴 Variables catégorielles
- 🟠 Variables binaires
- 🟡 Variables quantitatives
- 🟢 Variables quantitatives normalisées
- 🔵 Variables quantitatives ordinales ("ratings")
- 🟣 Variables cycliques

🎯 La cible : Défaut de paiement `**TARGET**` (binaire)

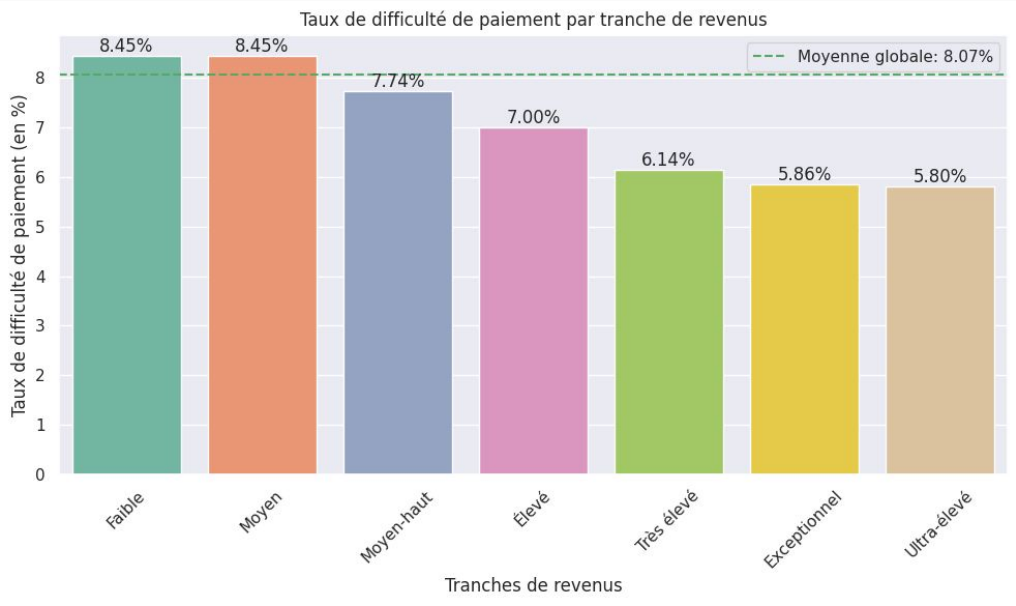
- **1** : le client a eu un **retard de paiement** de plus de X jours sur au moins une des Y premières échéances du prêt dans notre échantillon
- **0** : tous les autres cas



📊 Analyse exploratoire des données (EDA)

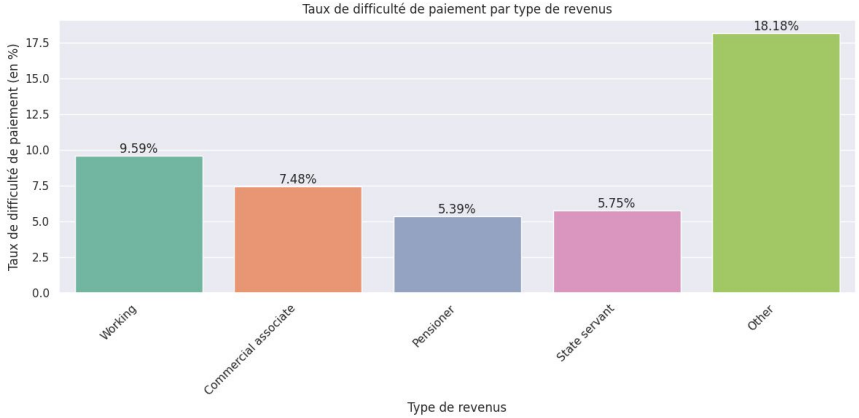
Taux de difficulté de paiement par tranches de revenu

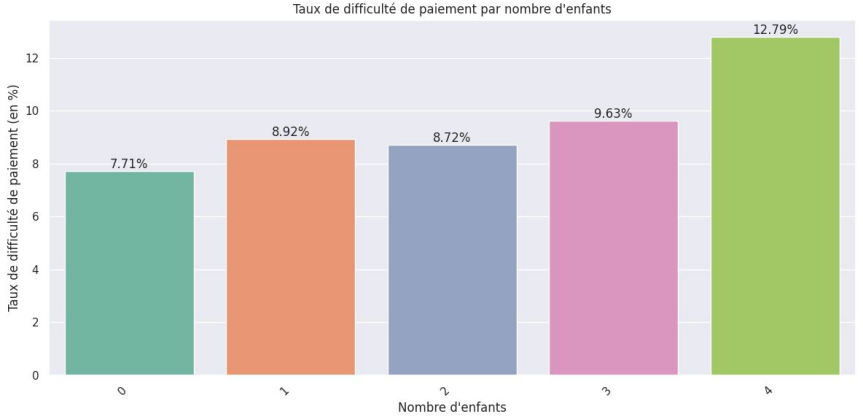
- Création de **tranches de revenu** à partir **AMT_INCOME_TOTAL**
  - Discrétisation d'une variable quantitative
- Calcul d'un **taux de difficulté de paiement**
  - *Nb de clients "défaut de paiement" (1) / Nb clients totaux*

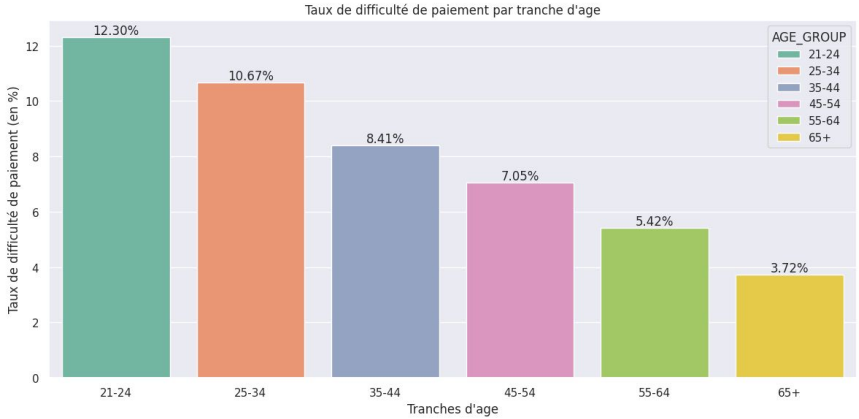


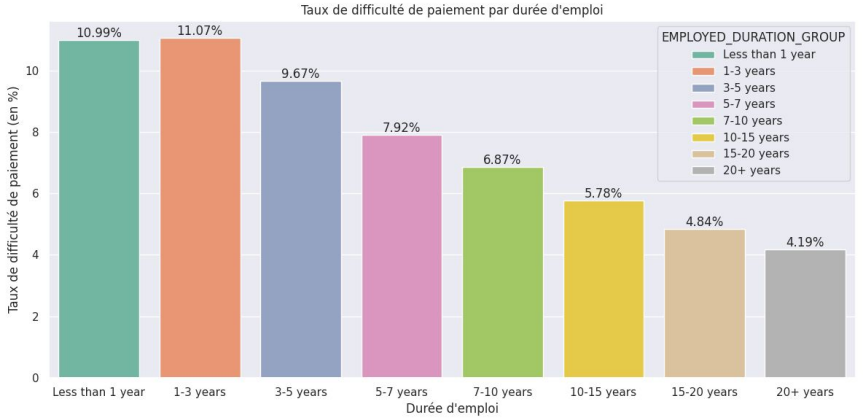
Analyse exploratoire des données (EDA)

OPENCLASSROOMS

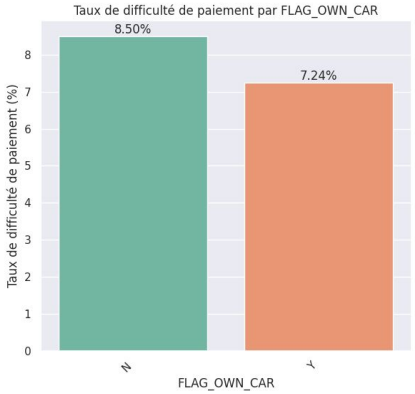


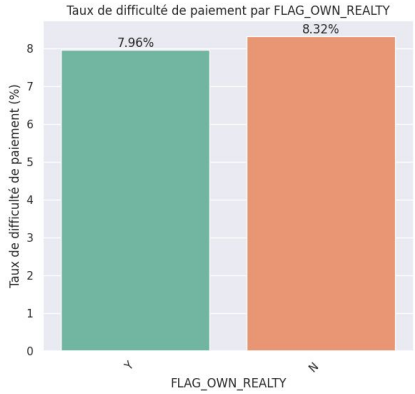


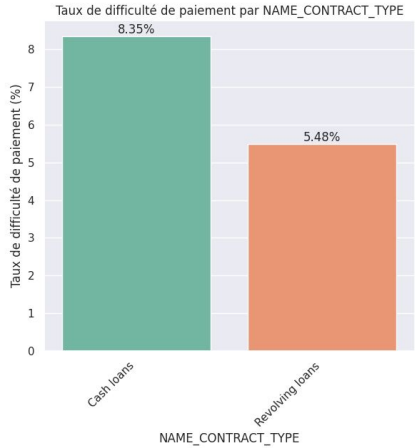


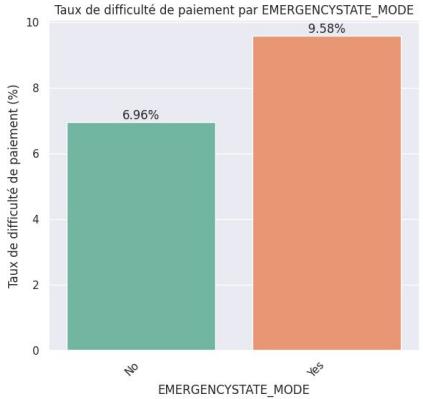


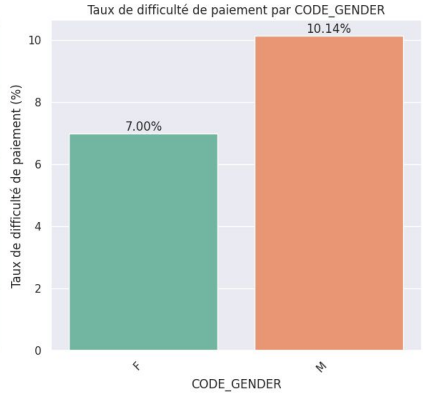
Analyse exploratoire des données (EDA)











Matrice de corrélation pour les variables quantitatives

OPENCLASSROOMS

Observations générales

Corrélations élevées entre certaines variables :

- **AMT_CREDIT**, **AMT_ANNUITY** et **AMT_GOODS_PRICE**
- **CNT_CHILDREN** et **CNT_FAM_MEMBERS**

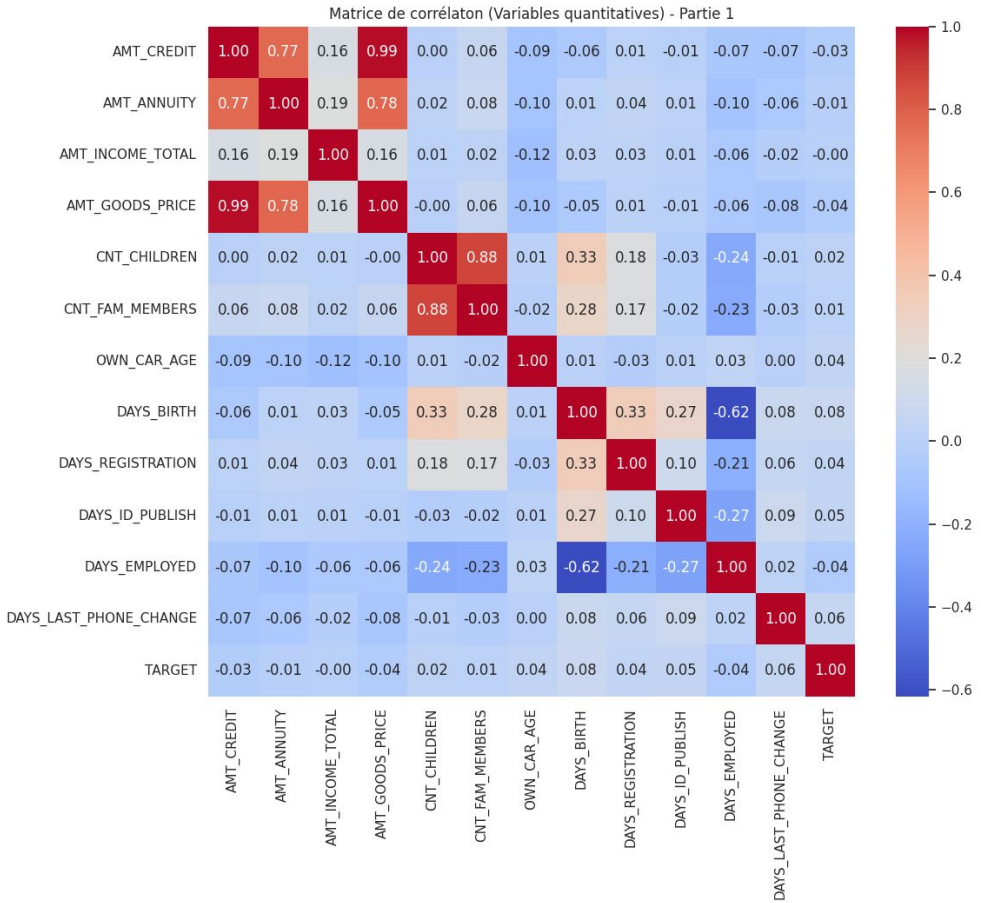
Feature engineering recommandé

Supprimer les colonnes :

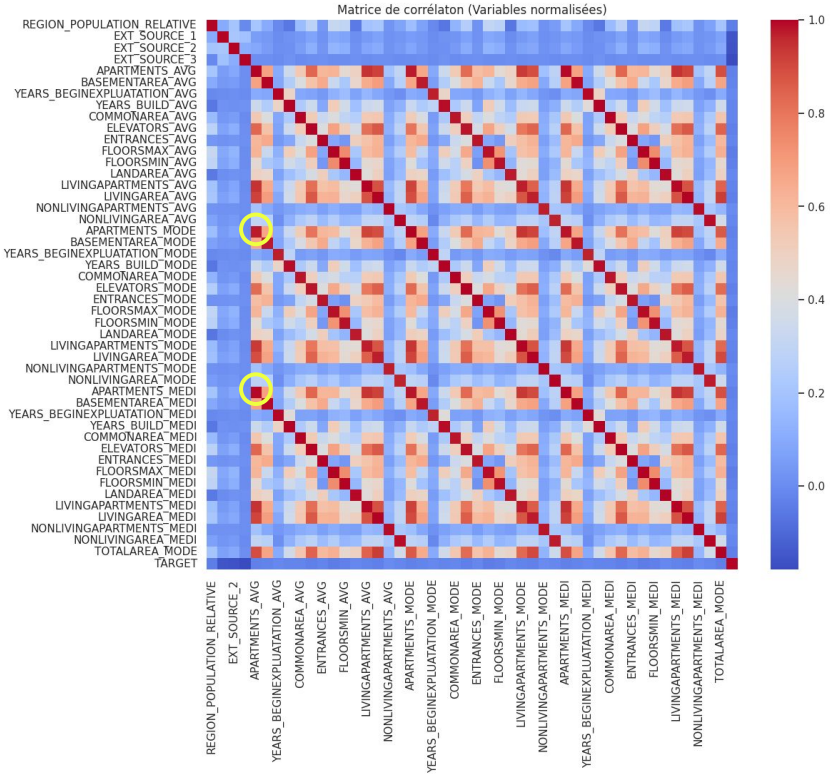
- **AMT_GOODS_PRICE**
- **CNT_FAM_MEMBERS**

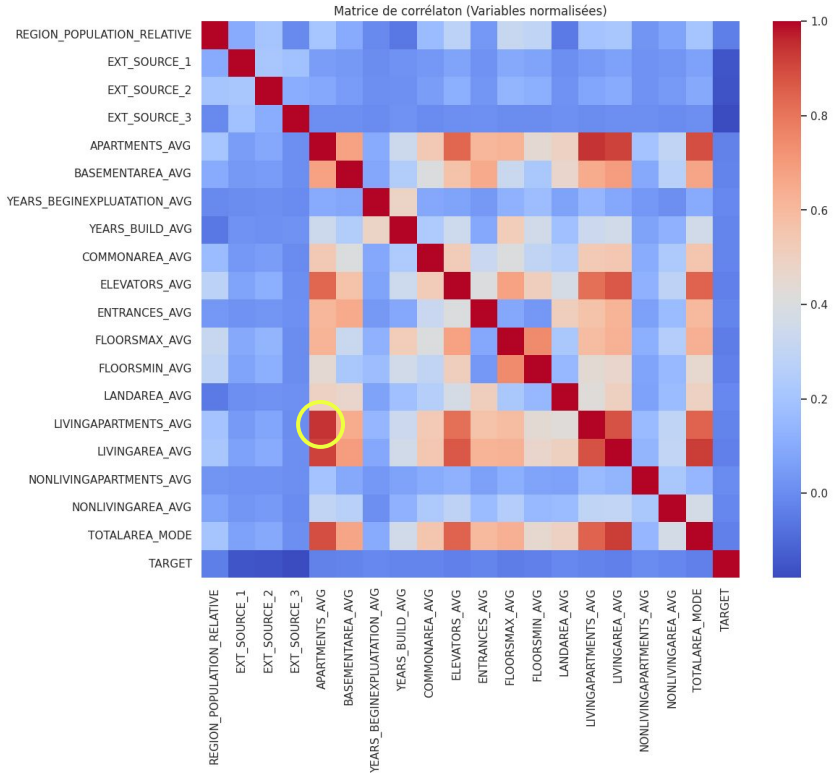
Objectif

- Éviter des problèmes de colinéarité



Matrice de corrélation pour les variables quantitatives normalisées





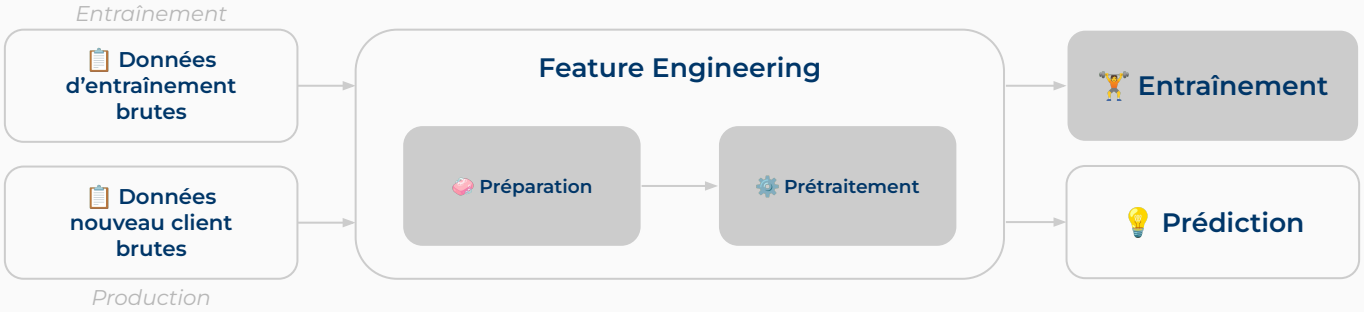
Feature engineering recommandé

- Conservation des versions `**AVG**` des variables (plus intuitives)
- Suppression des versions `**MODE**` et `**MEDI**`

Feature engineering recommandé

- Supprimer les variables fortement corrélées :
  `**LIVINGAPARTMENTS_AVG**`, `**LIVINGAREA_AVG**`

⚙️ Feature Engineering



Préparation

- ✔️ **Suppression colonnes inutiles**
- ✔️ **Création des tranches d'heure** pour `HOUR_APPR_PROCESS_START`
- ✔️ **Transformation `NAME_INCOME_TYPE`**
- ✔️ **Transformation `CNT_CHILDREN`**
- ✔️ **Création de ratios**
- ✔️ **Remplacement des valeurs manquantes** par une modalité qui existe déjà
- ✔️ **Conversion des colonnes contenant les valeurs "Y" et "N"**
- ✔️ **Remplissage des valeurs NaN** pour l'age de la voiture `OWN_CAR_AGE`

Preprocessing

Création d'un **pipeline de prétraitement** utilisant `Scikit-learn`

- **Imputation** des valeurs manquantes
- **Encodage** des variables catégorielles (encodage one-hot)
- **Standardisation** des variables numériques (robuste aux outliers)

Pré-processing particulier pour :

1. 🔴 Colonnes catégoriques
2. 🟠 Colonnes binaires
3. 🟡 Colonnes quantitatives
4. 🟢 Colonnes normalisées
5. 🔵 Colonnes de notation

⚖️ Gestion du déséquilibre de classe

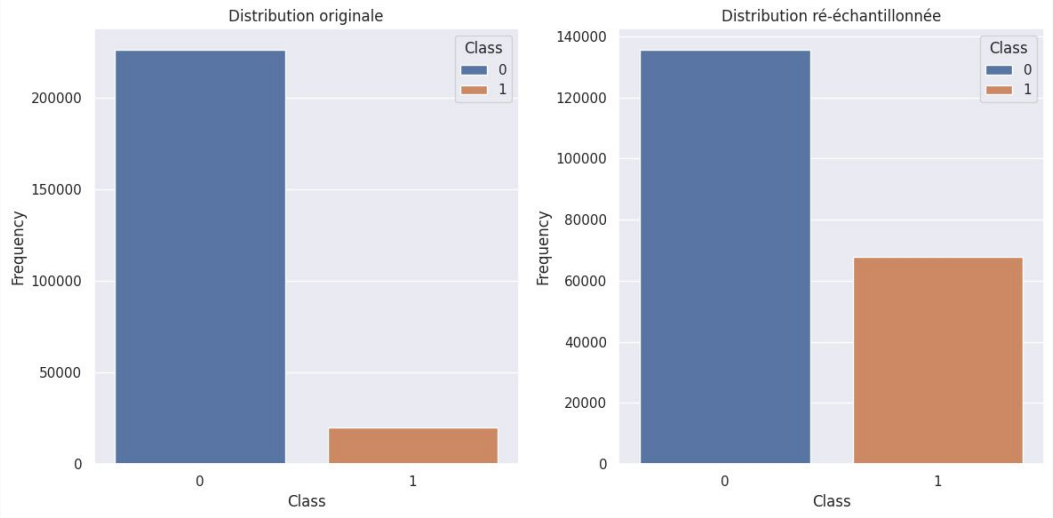
La variables cible `Target` contient :

- **91,9 % de classe 0**
- **8,1 % de classe 1** (Défaut de paiement)

Risque : le modèle pourrait être biaisé en faveur de la classe majoritaire (0) s'il n'est pas correctement géré. En effet, un modèle non ajusté pourrait simplement prédire systématiquement la classe 0 pour maximiser son taux de précision globale, mais cela serait inutile pour détecter les cas critiques de défaut de paiement (classe 1).

Il est donc essentiel d'appliquer des techniques adaptées :

- **Sur-échantillonnage** de la classe minoritaire avec `SMOTE`
- **Sous-échantillonnage** de la classe majoritaire avec `RandomUnderSampler`



📊 Métriques d'évaluation

✅ ROC AUC Score

- **Aire sous la courbe ROC**
- **Relation entre le taux de vrais positifs et le taux de faux positifs** à différents seuils de détection
- Aide à choisir un **seuil de détection optimal**

✅ Recall (Rappel)

- **Proportion des instances positives correctement identifiées** parmi tous les cas positifs réels (inclus les faux négatifs)
- Formule : **TP / (TP + FN)**
- Interprétation :
  - Important dans les cas où **détecter les positifs (classe 1) est critique** (par exemple, défaut de paiement)
  - Très efficace pour **mesurer les faux négatifs** (pour éviter un mauvais client détecté comme bon client)
  - **Un recall faible indique que le modèle manque beaucoup de vrais positifs** (donc produit trop de faux négatifs)

✅ Courbe Precision-Recall

✅ Matrice de confusion

- Tableau de comparaison des prédictions aux valeurs réelles
- Vrais positifs, faux positifs, vrais négatifs et faux négatifs

✅ Score métier

Minimisation du coût d'erreur de prédiction des FN et FP

- Faux négatif (FN) : Mauvais client prédit bon client, donc crédit accordé et perte en capital
- Faux positif (FP) : Bon client prédit mauvais client, donc refus crédit et manque à gagner en marge

Formule : 10 * FN + FP

✅ Optimisation du seuil de détection

- Seuil par défaut : 0,5
- Evaluation du modèle pour plusieurs seuils
- Un seuil bas favorise le rappel (détection des vrais positifs)

🚂 Entraînement de modèles de classification binaire

Objectifs

- **Identifier le modèle offrant les meilleures performances**.
- **Déterminer le seuil de détection optimal** pour classer les clients.

Approche Méthodologique

- **Comparer une gamme de modèles**, des plus simples aux plus avancés.
- **Utiliser les pipelines de scikit-learn** pour faciliter l'optimisation et l'évaluation des modèles.

Entraînement et évaluation

- **Cross-Validation** : méthode d'évaluation lors de l'entraînement du modèle.
- **GridSearchCV** : Optimisation des hyperparamètres.
- **ROC AUC** : métrique par laquelle les performances du modèle doivent être évaluées.
- **Synthèse des résultats**.

🧪 Comparaison des performances des modèles

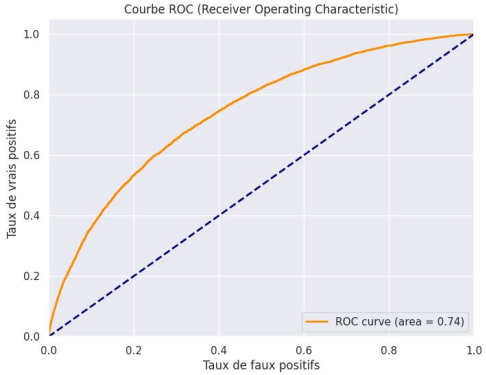
Model (best)	ROC AUC	Recall	Precision	F1	Faux-négatifs	Faux-positifs
Regression logistic*	0.74	0.41	0.22	0.28	2902	7128
Random Forest	0.72	0.03	0.33	0.06	4767	356
Gradient Boosting Classifier	0.76	0.09	0.40	0.15	4486	693

**Meilleurs hyperparamètres : { 'C': 100, 'penalty': 'l1', 'solver': 'liblinear' }*

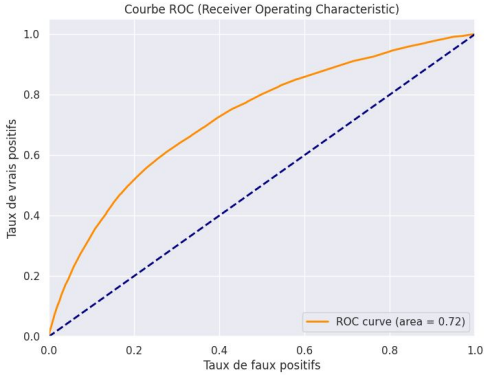
Courbes ROC et Matrices de confusion

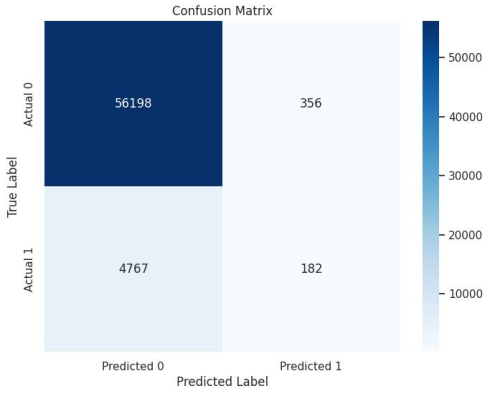
OPENCLASSROOMS

Régression logistique (best)

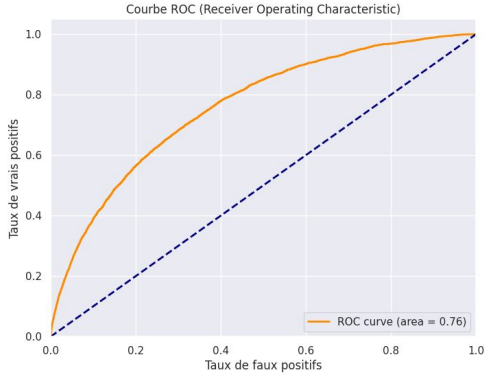


Random Forest (best)





Gradient Boosting Classifier (best)



📊 Régression logistique (best)

Courbe Precision-Recall

- Utile lorsque **les classes sont déséquilibrées**
- **Compromis entre précision et rappel** pour différents seuils.

Métriques

- ROC AUC Score : 0.74
- Recall : 0.41
- Precision : 0.22
- F1 Score : 0.29
- F2 Score : 0.35

📊 Seuil de détection en fonction du coût métier

Coût métier

Objectif

- Minimiser le coût métier.
- Déterminer le seuil à partir duquel nous décidons d'octroyer un crédit.

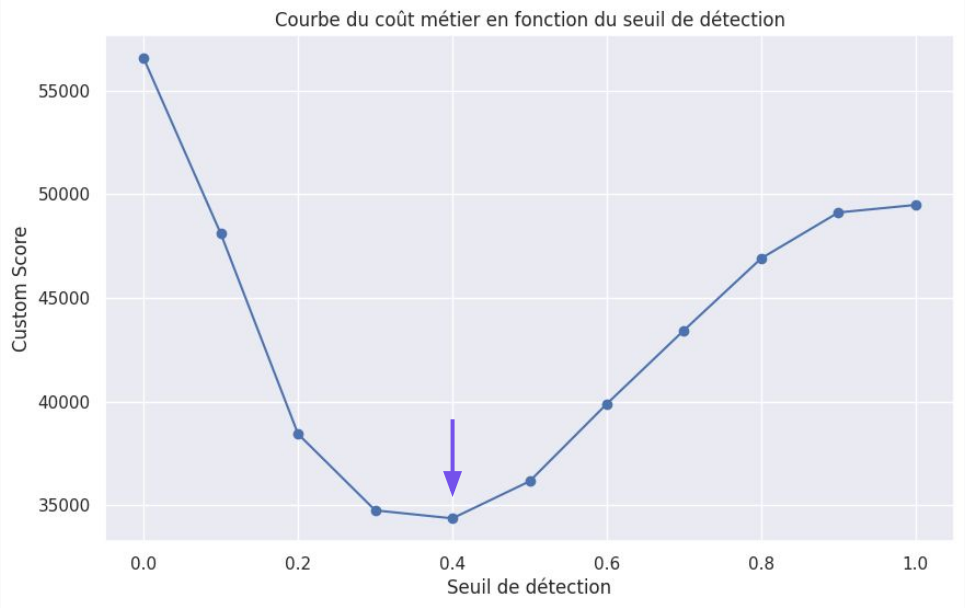
Coût métier = 10 FN + FP

- **Faux négatif (FN)** : Mauvais client prédit bon client : donc crédit accordé et perte en capital
- **Faux positif (FP)** : Bon client prédit mauvais : donc refus crédit et manque à gagner en marge

*L'objectif est de **minimiser les faux négatifs**.*

***Privilégier un rappel (recall) élevé**, car cette métrique mesure la capacité du modèle à identifier correctement toutes les instances positives (classe 1).*

*Cela peut nécessiter un seuil de détection plus bas, ce qui **augmente la détection des vrais positifs** au détriment potentiel de la précision.*



🔬 Interprétabilité du modèle

Objectif

Dans les processus d'approbation de prêts, **les banques doivent expliquer au client les raisons qui ont motivé la décision**, notamment en cas de rejet.

Méthodes de mesure d'importance des variables

- Coefficients de la régression logistique
- SHAP

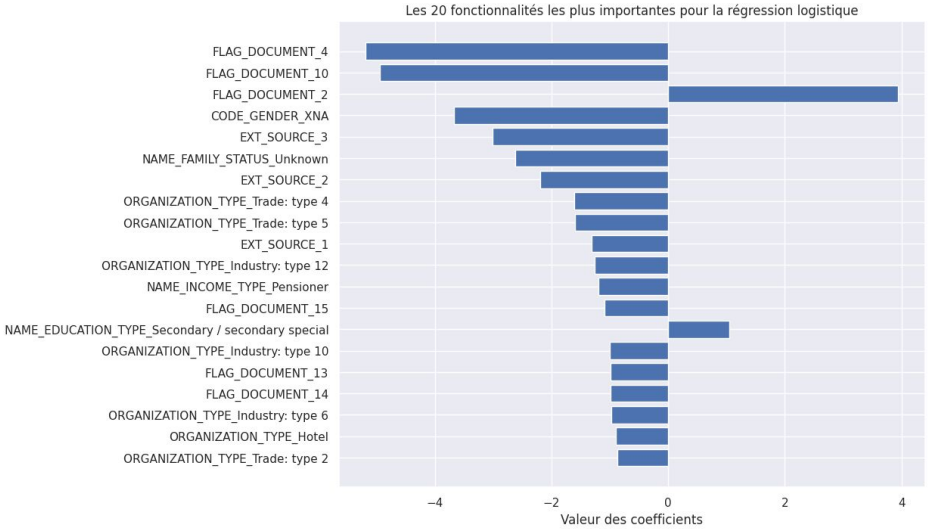
Scope

- **Analyse globale**
  - Permet d'identifier les caractéristiques les plus importantes sur l'ensemble du jeu de données.
- **Analyse locale** pour un client donné
  - Se concentre sur une prédiction individuelle
  - Utile pour justifier ou expliquer une décision particulière (comme un refus de prêt)

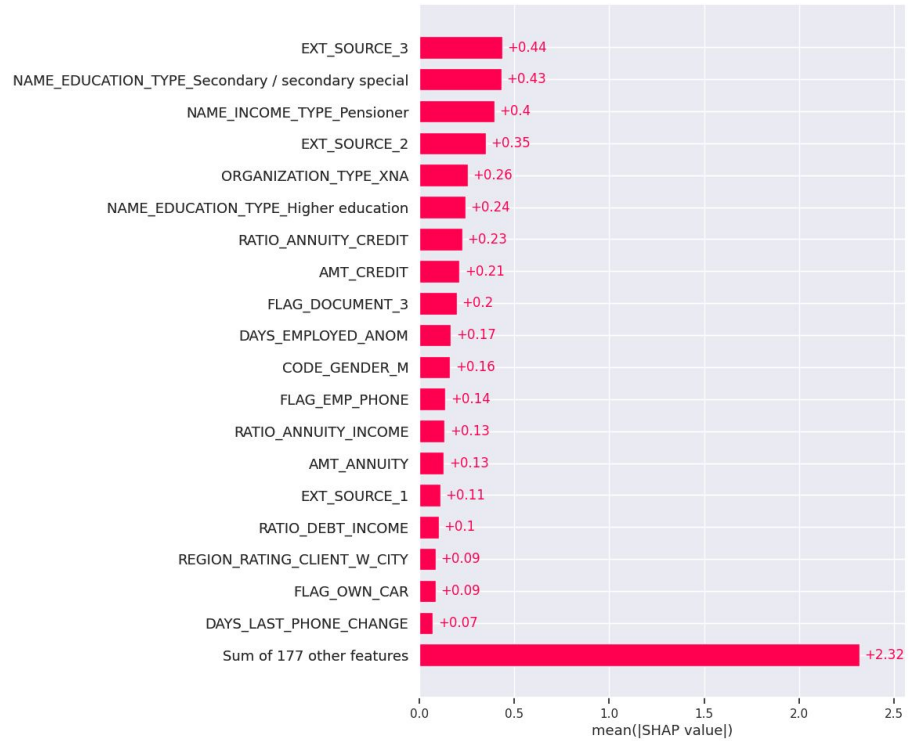
Interprétabilité : Analyse globale

OPENCLASSROOMS

Coefficients de la régression logistique



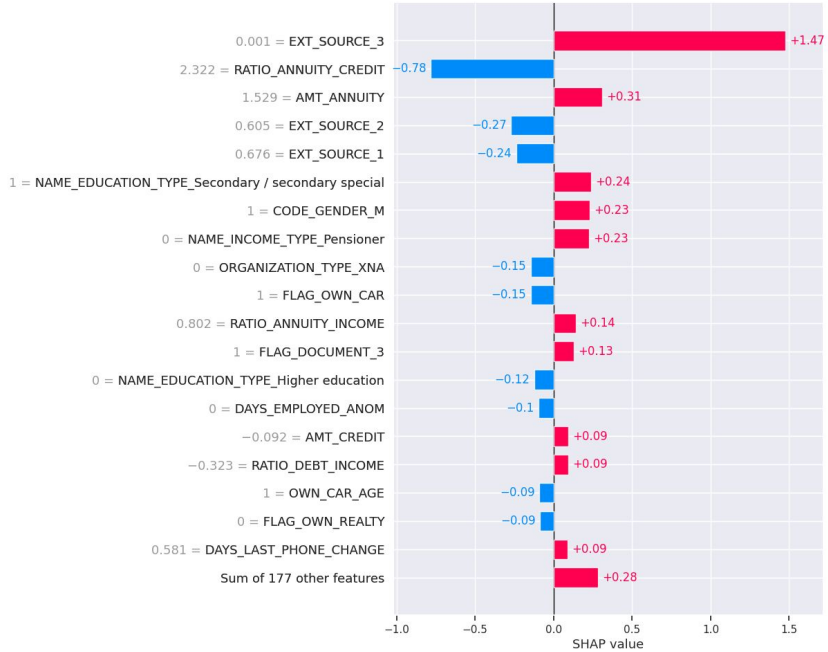
Importance globale des variables avec SHAP



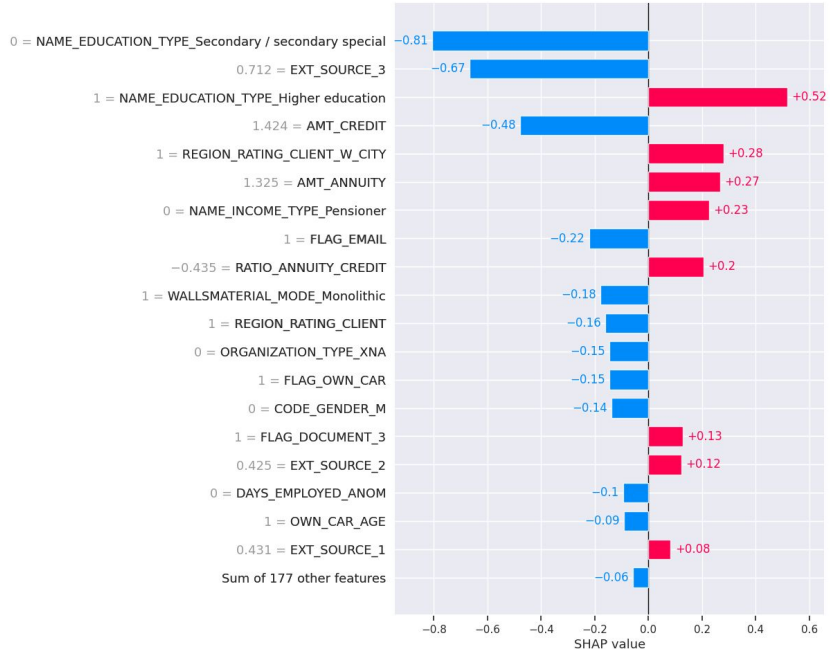
Interprétabilité : Analyse locale pour un client donné

OPENCLASSROOMS

Exemple pour un “bon client” (TARGET=0)



Exemple pour un “mauvais client” (TARGET=1)



Recommandation : *mise en place d’un pipeline d'applicabilité pour chaque prédiction. (en particulier en cas de refus de prêt).*

🎯 Conclusion

Le projet visait à **développer un modèle de prédiction pour l'octroi de prêts** permettant de déterminer si un client remboursera ou non son crédit.

Modèle retenu

- **La régression logistique**
- **Meilleur compromis entre performance et interprétabilité**

Le modèle permet de :

- Réaliser des prédictions
- Identifier les variables clés influençant chaque décision

🔍 **Recommandations :**

Intégrer **un pipeline d'explicabilité pour expliquer chaque décision**, notamment **en cas de refus de prêt**, afin de mieux accompagner les chargés de relation client et les utilisateurs finaux.

Merci 🙏

Avez-vous des questions ?

📦 Livrables

- Dépôt GitHub

🎓 Etudiant

- David Scanu