OC – Parcours «  Data Scientist »

Pauline blivet

OpenClassrooms – Parcours «  Data Scientist »

Projet 7 – ancienne édition

Pauline BLIVET

Mémoire méthodologique

Implémentez un modèle de scoring

Table des matières

Table des matières

[I. Contexte 2](#_Toc94433371)

[1) Présentation du sujet 2](#_Toc94433372)

[2) Présentation des données 2](#_Toc94433373)

[II. Exploration des données 3](#_Toc94433374)

[1) Analyse des données manquantes 3](#_Toc94433375)

[2) Analyse statistique descriptive 3](#_Toc94433376)

[3) Analyse bivariée 6](#_Toc94433377)

[Traitement des données 9](#_Toc94433378)

[1) Feature Engineering 9](#_Toc94433379)

[2) Données déséquilibrées 9](#_Toc94433380)

[a) SMOTE 10](#_Toc94433381)

[b) Pondération 11](#_Toc94433382)

[III. Méthodologie d’entraînement du modèle 11](#_Toc94433383)

[1) Séparation train/test 11](#_Toc94433384)

[2) Cross-validation 12](#_Toc94433385)

[IV. Algorithme d’optimisation et métrique d’évaluation 13](#_Toc94433386)

[1) Métriques d’évaluation 13](#_Toc94433387)

[2) Optimisation 15](#_Toc94433388)

[V. Interprétabilité du modèle 17](#_Toc94433389)

[1) Importance des variables 17](#_Toc94433390)

[2) Matrice de confusion 19](#_Toc94433391)

[VI. Limites et améliorations 20](#_Toc94433392)

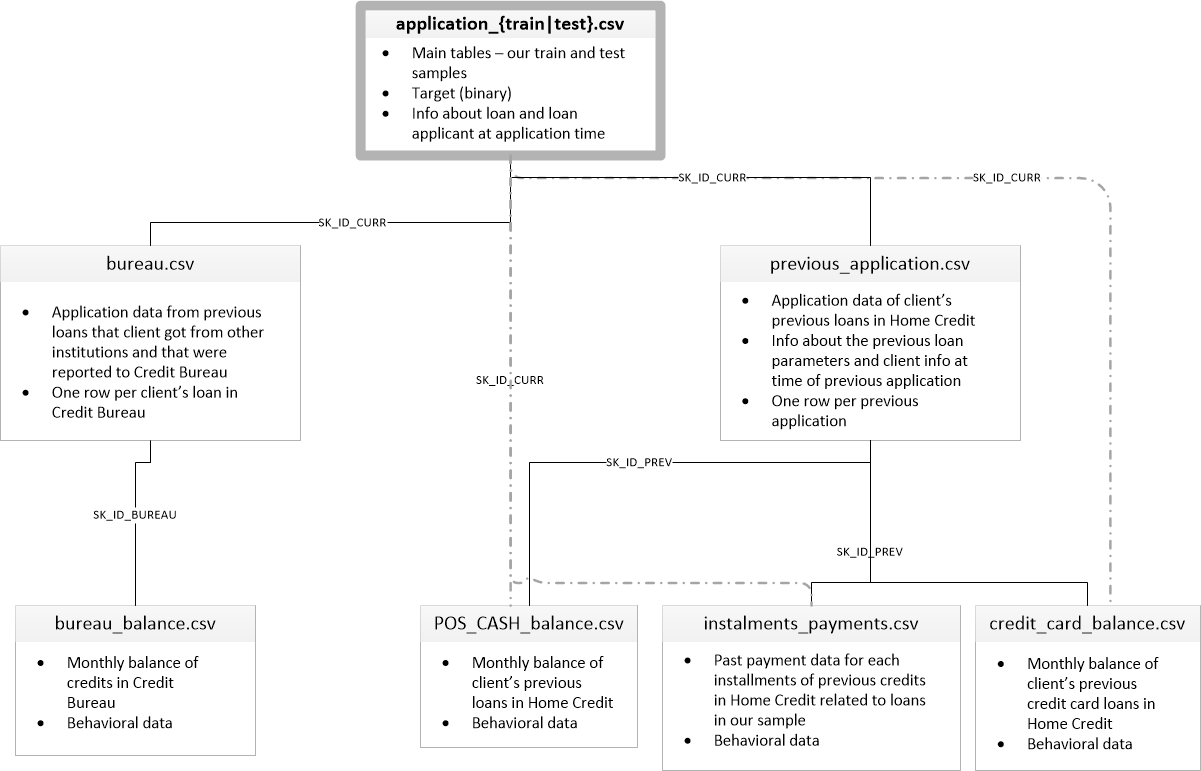
# Contexte

## Présentation du sujet

Ce mémoire constitue l’un des livrables du projet « Implémentez un modèle de scoring » du parcours Data Scientist d’Openclassrooms.

Il présente le processus de modélisation et d’interprétabilité du modèle mis en place dans le cadre du projet.

Le projet est issu d’une compétition Kaggle « Home Credit Default Risk » qui présente des sources de données variées sur la demande de prêt actuelle et les demandes précédentes, dont le schéma de données est ci-dessous :



Le projet consiste à développer pour la société « Prêt à Dépenser », une société de crédit de consommation, un modèle de scoring de la probabilité de défaut de paiement d’un client avec pas ou peu d’historique de prêt. Les données utilisées pour ce projet sont une base de données de 307 000 clients comportant 122 features (âge, sexe, emploi, logement, revenus, informations relatives au crédit, notation externe, etc.).

## Présentation des données

Les sources de données étant diverses, nous nous sommes essentiellement appuyés sur le jeu de données « application\_train » qui est le jeu d’entraînement et la table principale. Cette table représente les données statistiques de tous les demandeurs, chaque ligne représentant un prêt.

Le jeu d’entraînement inclut la variable cible « TARGET » que nous cherchons à prédire, celle-ci sera supprimée lors de l’entraînement du modèle.

Parmi les 122 colonnes présentent dans le jeu de données, 65 sont décimales, 41 sont entières et 16 sont catégorielles.

# Exploration des données

L’exploration des données nous permet de mieux comprendre les données sous-jacentes et savoir lesquelles utiliser.

L’analyse de données exploratoire permet de connaître la distribution des variables, trouver les anomalies, valeurs manquantes et de trouver des relations entre les données. Cela nous permettra de mieux choisir le ou les modèles à utiliser.

## Analyse des données manquantes

Dans la table « application\_train.csv », 67 colonnes sur 122 présentent des valeurs manquantes dont 17 en contiennent au moins 66%.

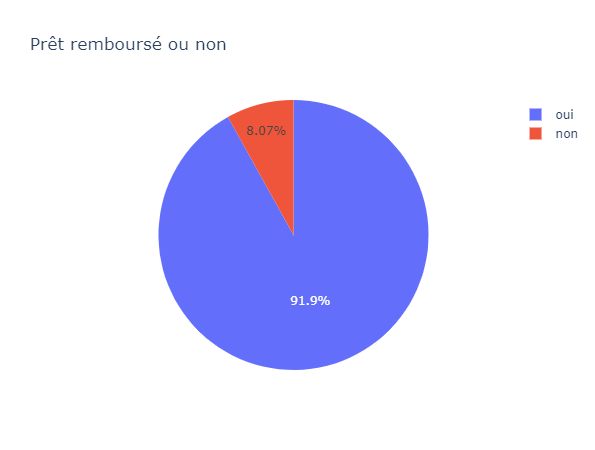
|  |  |  |
| --- | --- | --- |
|  | **Missing Values** | **% of Total Values** |
| COMMONAREA\_MEDI | 214865 | 69.9 |
| COMMONAREA\_AVG | 214865 | 69.9 |
| COMMONAREA\_MODE | 214865 | 69.9 |
| NONLIVINGAPARTMENTS\_MEDI | 213514 | 69.4 |
| NONLIVINGAPARTMENTS\_MODE | 213514 | 69.4 |
| NONLIVINGAPARTMENTS\_AVG | 213514 | 69.4 |

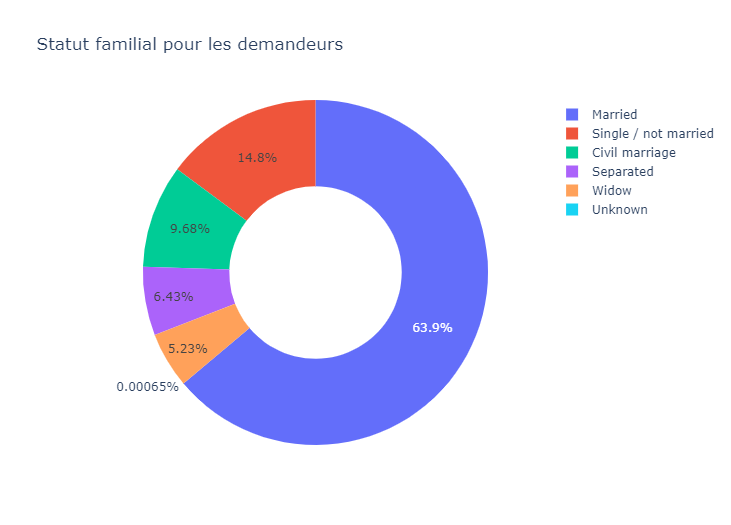
Selon le modèle utilisé, s’il gère l’imputation de données ou non, nous verrons à posteriori comment gérer les valeurs manquantes.

## Analyse statistique descriptive

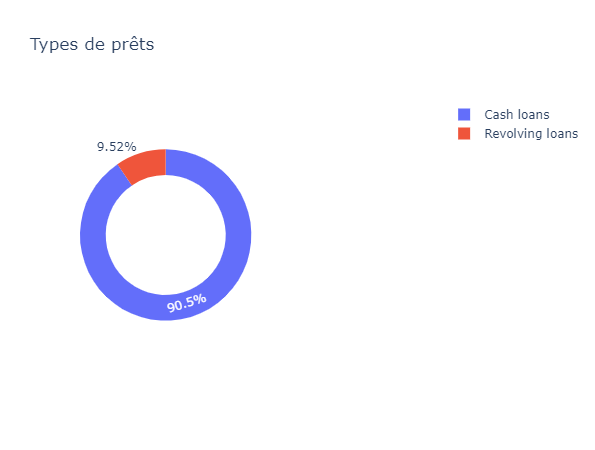
Tout d’abord, nous allons analyser la distribution de la variable cible, à savoir TARGET.

La variable cible est binaire, 1 étant les personnes qui présentent un risque de défaut et 0 les personnes qui arriveront à rembourser le prêt. On remarque à travers cette distribution que celle-ci est disparate puisqu’il y a que 8% des personnes qui présentent un risque de défaut. Les données sont alors très déséquilibrées.

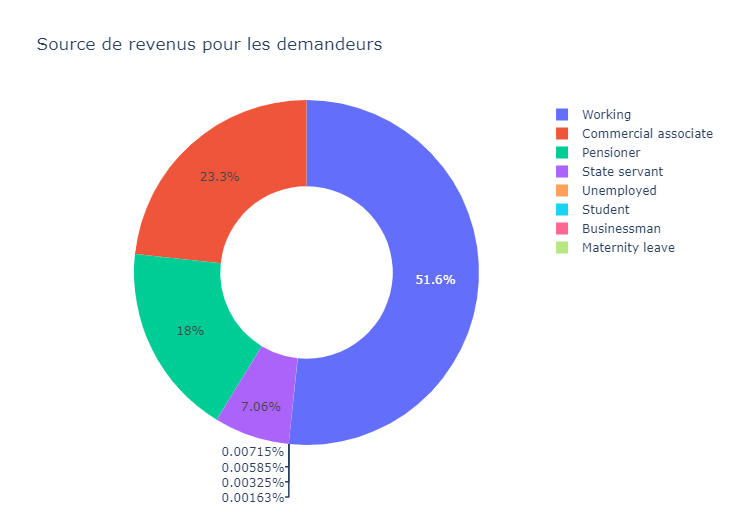


Les demandeurs peuvent avoir différents statuts familiaux, la grande majorité à hauteur de 64% est marié, près de 15% sont célibataires ou non mariés.

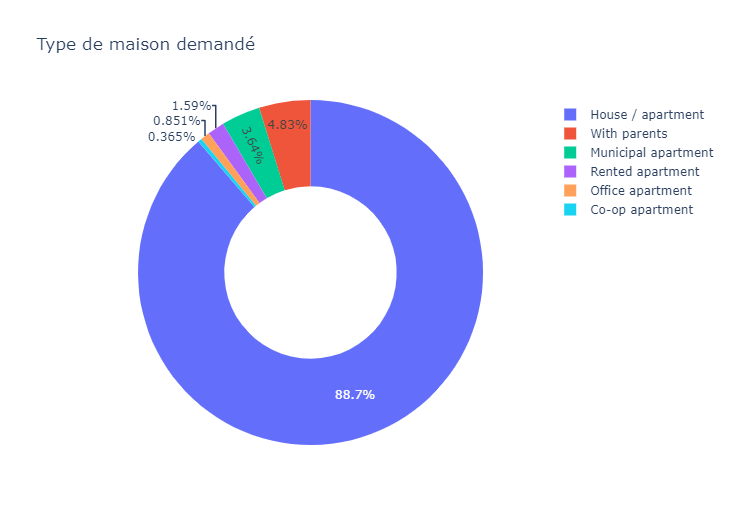
Les demandeurs peuvent contracter deux types de prêts, à savoir les prêts en espèces qu’on pourrait dire comptant et les prêts renouvelables. Les prêts ou crédits renouvelables sont des crédits à la consommation qui présentent une grande liberté d’utilisation et de remboursement, le taux d’intérêt est généralement plus élevé. Les prêts demandés sont majoritairement des prêts en espèces.



Les demandeurs peuvent avoir diverses sources de revenus, 51,6% d’entre eux travaillent, 23.3% sont des commerciaux et 18% sont des retraités.



Dans le cadre de la demande de prêt pour un bien immobilier, le plus demandé est pour une maison ou un appartement, à hauteur de 88.7%, le deuxième étant un bien avec les parents à 4,83% et le troisième correspond à un appartement municipal (3,64%).



Les prêts demandés dans notre jeu de données, sont faits par différents types de sociétés, le plus courant étant : « Business Entity Type 3 », le deuxième étant self-employed à savoir l’entrepreunariat et le troisième étant médecin.

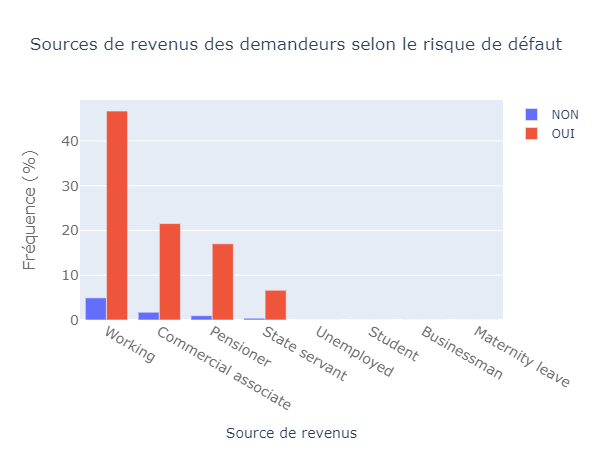


Afin de mieux comprendre l’impact des différentes variables sur la variable cible, nous allons procéder à l’analyse bivariée.

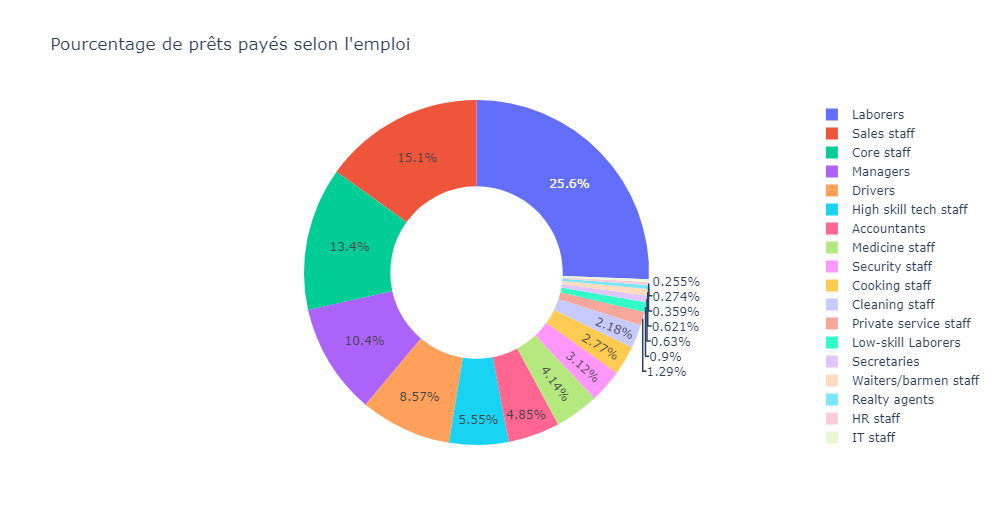
## Analyse bivariée

Tout d’abord, nous étudierons l’analyse bivariée qualitative.

Les sources de revenus du demandeur selon si le prêt a été remboursé ou non, nous montre que les sources de revenus de ceux qui ont remboursé leurs prêts sont majoritairement des travailleurs (45%) et des commerciaux (21%). Néanmoins, 5% des travailleurs n’ont pas remboursé leurs prêts.

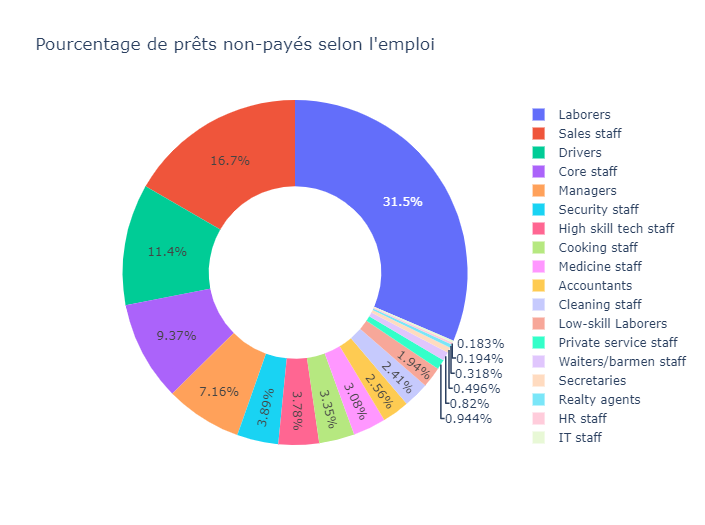


Afin de mieux comprendre, si les sources de revenus, en l’occurrence l’emploi du demandeur a un impact sur la propension à rembourser le prêt, nous allons dans un premier temps regarder l’emploi des demandeurs qui ont remboursé puis celui de ceux qui n’ont pas remboursé.



Les cinq emplois qui ont davantage rembourser leurs prêts sont les ouvriers (25,6%), le personnel de vente (15,1%), le personnel de base (13,4%), les managers (10,4%) et les conducteurs (8,57%).

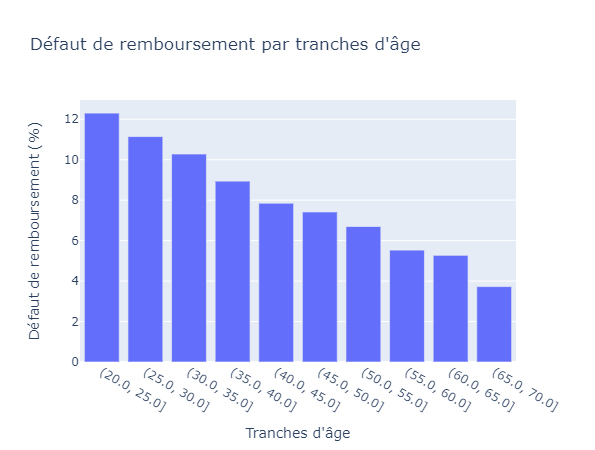
Nous allons désormais regarder la distribution des différents emplois parmi les demandeurs qui n’ont pas remboursé leurs prêts.



Les ouvriers sont majoritaires dans la catégorie de mauvais payeur, à hauteur de 31,5%. Ils sont d’ailleurs plus majoritaires que dans la catégorie de « bon payeur ».

Le personnel de vente représente une part de 16,7% des mauvais payeurs, ce qui est légèrement supérieur à la part qu’ils avaient précédemment (15,1%). La part des conducteurs dans les « mauvais-payeurs » est légèrement plus importante (presque 3 points de pourcentage) que dans les bons payeurs. Le personnel de base et les managers ont une part moins importante dans les mauvais payeurs que dans les bons payeurs.

La dernière variable catégorielle étudiée est les tranches d’âge.



La distribution de l’échec de remboursement en fonction des tranches d’âges, nous montre que l’âge a un effet notable dans cet échec. En effet, il apparaît que plus on est jeune plus le demandeur aura des difficultés à rembourser le prêt. Nous pourrions vérifier cela avec des corrélations, lors de l’analyse bivariée quantitative.

L’analyse bivariée quantitative a été faite à partir de corrélations de Pearson.

Les variables catégorielles qui ont aux plus deux labels ont été label-encodées et les autres ont été dichotomisées.

Les quinze premières corrélations positives sont :

Les variables DAYS\_BIRTH et DAYS\_EMPLOYED sont corrélés positivement avec la variable cible (ces variables sont négatives en nombre de jours au moment de la demande de prêt). Ainsi, nous les transformerons en valeur absolue.

Une personne jeune aura une propension à ne pas rembourser plus importante qu’une personne plus âgée. Une personne qui travaille depuis peu aura une propension à ne pas rembourser plus importante qu’une personne qui travaille depuis plus longtemps.

|  |  |
| --- | --- |
| **Variables** | **Corrélations** |
| OCCUPATION\_TYPE\_Laborers | 0,04 |
| FLAG\_DOCUMENT\_3 | 0,04 |
| REG\_CITY\_NOT\_LIVE\_CITY | 0,04 |
| FLAG\_EMP\_PHONE | 0,05 |
| NAME\_EDUCATION\_TYPE\_Secondary | 0,05 |
| REG\_CITY\_NOT\_WORK\_CITY | 0,05 |
| DAYS\_ID\_PUBLISH | 0,05 |
| CODE\_GENDER\_M | 0,05 |
| DAYS\_LAST\_PHONE\_CHANGE | 0,06 |
| NAME\_INCOME\_TYPE\_Working | 0,06 |
| REGION\_RATING\_CLIENT | 0,06 |
| REGION\_RATING\_CLIENT\_W\_CITY | 0,06 |
| DAYS\_EMPLOYED | 0,07 |
| DAYS\_BIRTH | 0,08 |
| TARGET | 1,00 |

Les quinze premières corrélations négatives sont :

|  |  |
| --- | --- |
| **Variables** | **Corrélations** |
| EXT\_SOURCE\_3 | -0,18 |
| EXT\_SOURCE\_2 | -0,16 |
| EXT\_SOURCE\_1 | -0,16 |
| NAME\_EDUCATION\_TYPE\_Higher education | -0,056593 |
| CODE\_GENDER\_F | -0,05 |
| NAME\_INCOME\_TYPE\_Pensioner | -0,05 |
| DAYS\_EMPLOYED\_ANOM | -0,05 |
| ORGANIZATION\_TYPE\_XNA | -0,05 |
| FLOORSMAX\_AVG | -0,04 |
| FLOORSMAX\_MEDI | -0,04 |
| FLOORSMAX\_MODE | -0,04 |
| EMERGENCYSTATE\_MODE\_No | -0,04 |
| HOUSETYPE\_MODE\_block\_of\_flats | -0,040594 |
| AMT\_GOODS\_PRICE | -0,04 |
| REGION\_POPULATION\_RELATIVE | -0,04 |

Les sources extérieures présentent les corrélations les plus négatives. Selon la documentation, ces variables représentent un score normalisé provenant de source(s) extérieure(s), on ne sait pas exactement ce dont il s’agit.

L’éducation élevée du demandeur est la quatrième corrélation la plus négative, soit une personne ayant fait des études importantes aura une propension plus faible à ne pas rembourser le prêt par rapport à une personne qui n’a pas ce niveau d’éducation.

Une femme aura aussi une propension plus faible à ne pas rembourser le prêt par rapport à un autre genre (inconnu ou homme).

Après avoir traité les variables catégorielles, nous procéderons au traitement des données, en commençant par le feature engineering.

# Traitement des données

Le traitement des données peut améliorer le modèle en rajoutant des variables polynomiales ou/et des variables issues du domaine bancaire. Nous devons aussi gérer le problème des données déséquilibrées sur la variable cible.

## Feature Engineering

Les variables polynomiales permettent de rajouter des variables à une puissance plus élevée de variables existantes ainsi que la création de termes d’interaction.

Nous créerons plusieurs variables issues des variables les plus corrélées à savoir les sources extérieures (EXT\_SOURCE\_1, EXT\_SOURCE\_2, EXT\_SOURCE\_3) et l’âge (DAYS\_BIRTH). Nous irons jusqu’au degré 3, ce qui constitue 31 nouvelles variables dont la constante 1.

Les variables issues du domaine bancaire sont :

* **CREDIT\_INCOME\_PERCENT**: Pourcentage du montant du crédit par rapport au revenu du client
* **ANNUITY\_INCOME\_PERCENT**: Pourcentage d’un prêt par annuité par rapport au revenu du client
* **CREDIT\_TERM**: Durée du paiement en mois (si l’annuité est un montant mensuel)
* **DAYS\_EMPLOYED\_PERCENT**: Pourcentage du nombre de jours employés par rapport à l’âge du client.

Nous verrons en intégrant ces variables à notre modèle si elles seront utiles.

## Données déséquilibrées

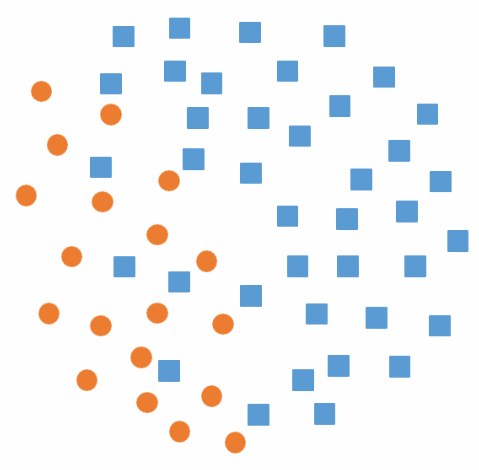
Afin de traiter le problème des données déséquilibrées, nous avons deux méthodes, SMOTE et la pondération issue du modèle. Le problème des données déséquilibrées entraîne un biais étant donné que l’algorithme d’apprentissage aura peu de données de la classe minoritaire et sera moins robuste.

De plus, les prédictions peuvent être faussées et dans notre cas, prédire un score majoritairement bas et que le client ne risque pas de faire de défaut.

### SMOTE

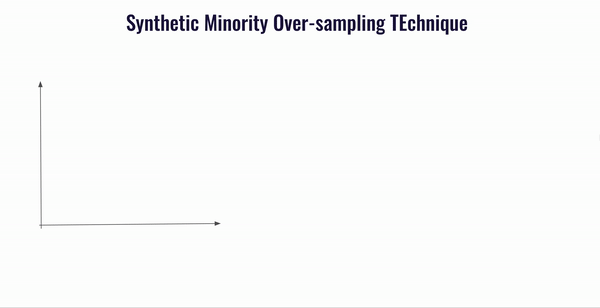
SMOTE est une technique de sur-échantillonnage, elle va donc créer des échantillons synthétiques de la classe minoritaire, la taille de classe majoritaire augmentera, contrairement aux méthodes de sous-échantillonnage qui diminueront la taille de la classe majoritaire.

SMOTE (Synthehic Minority OverSampling Technique) fonctionne en sélectionnant d’abord un point de la classe minoritaire, noté A et cherche ses k plus proches voisins faisant partie de la classe minoritaire. Un point synthétique est alors créé en choisissant un des k plus proches voisins, noté B au hasard. Les points B et A se connectent en formant un segment dans l’espace de la variable. Les points synthétiques sont générés comme une combinaison convexe des deux points choisis, A et B.



SMOTE a cinq étapes principales :

1. Choix d’un vecteur caractéristique de notre classe minoritaire que nous appellerons A ;
2. Sélection des k-voisins les plus proches (k=5 par défaut) et choix de l’un d’eux au hasard que l’on appellera B ;
3. Calcul de la différence pour chaque valeur caractéristique (feature value) i, A[i]-B[i] et multiplication de celle-ci par un nombre aléatoire entre [0,1];
4. Ajout du résultat précédent à la valeur de la caractéristique i du vecteur A afin d’obtenir un nouveau point (une nouvelle donnée) dans l’espace des caractéristiques ;
5. Répétition de ces opérations pour chaque point de données de la classe minoritaire ;



### Pondération

La pondération permet que toutes les classes aient une importance équivalente, sans regarder combien d’individus sont présents dans chaque classe, sur l’ensemble d’entraînement. Cela permet d’éviter que les modèles prédisent la classe la plus fréquente, juste parce qu’elle est plus commune.

La pondération permet de spécifier au modèle que de classer un point de la classe minoritaire est plus important que de classer un point de la classe majoritaire. Ainsi, une erreur de classification d’un point de la classe minoritaire est plus pénalisant par rapport à une erreur de classification sur la classe majoritaire.

# Méthodologie d’entraînement du modèle

## Séparation train/test

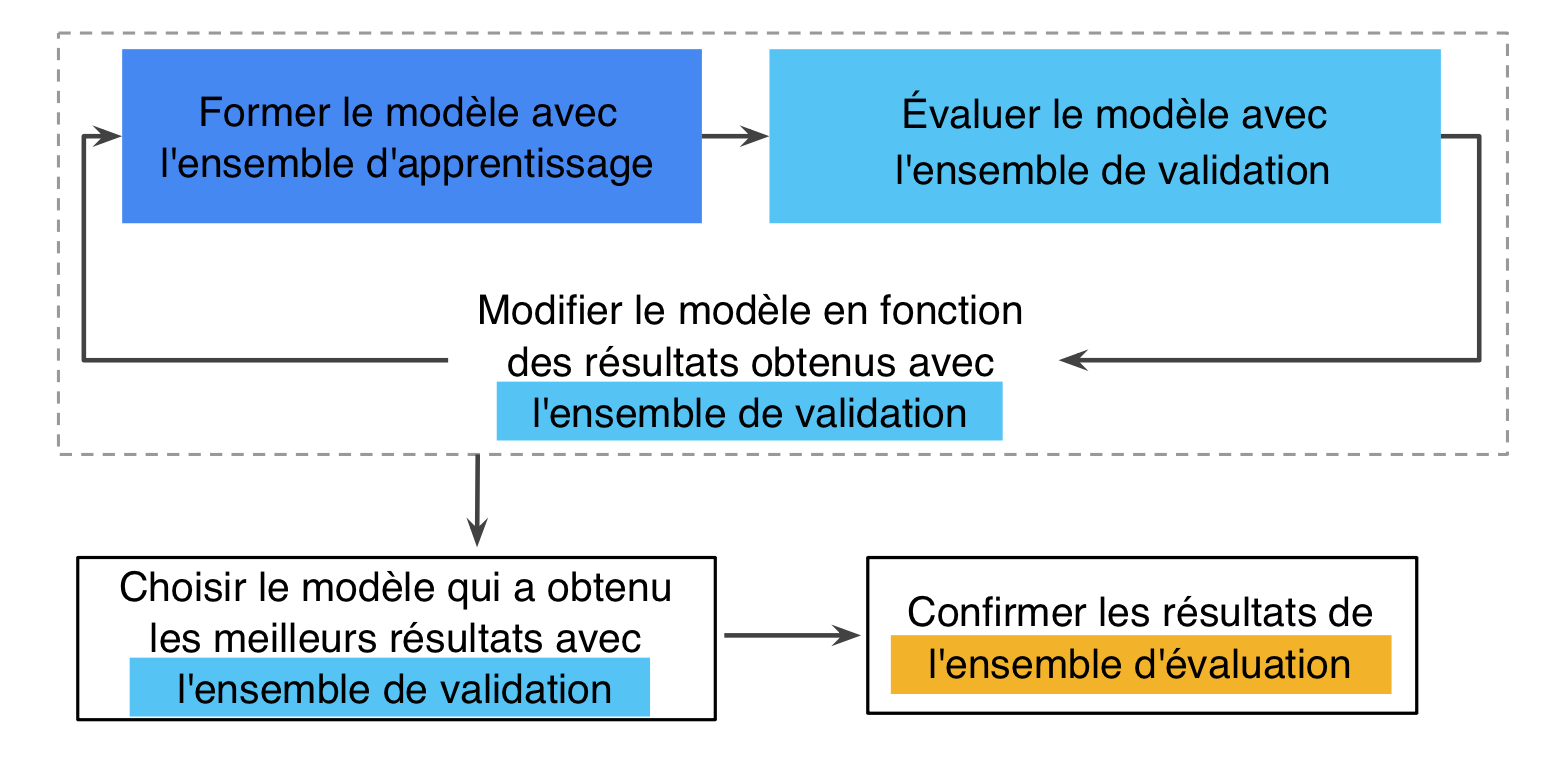
Tout d’abord, nous avons séparé le jeu de données « application\_train » en deux parties train/test afin d’éviter la fuite de données et qui servira pour l’évaluation du modèle.

L’ensemble de test constitue 20% du jeu de données initial.

Afin d’éviter le sur-apprentissage, tout en convergeant vers le meilleur modèle possible, il est préférable d’utiliser non pas deux jeux de données (apprentissage et évaluation) mais trois, c’est-à-dire apprentissage, validation et évaluation.

L’ensemble de validation sert à évaluer les résultats de l’ensemble d’apprentissage et l’ensemble d’évaluation sert à vérifier l’évaluation après que le modèle ait passé l’étape de validation.

La figure ci-dessous permet de mieux comprendre ce processus :



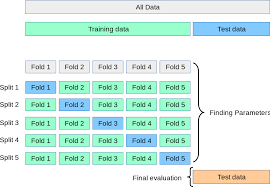
## Cross-validation

Pour évaluer le modèle avec l’ensemble de validation et trouver les meilleurs hyperparamètres, nous allons procéder à une cross-validation en 5 nœuds. Le jeu de données d’apprentissage a été divisé en deux, l’ensemble d’entraînement contenant 80% des données et l’ensemble de validation contenant les 20% restants.

En faisant de la cross-validation, nous nous assurons que toutes les observations du jeu de données original puissent apparaître dans l’ensemble d’entraînement, mais aussi dans l’ensemble de validation.

On effectue une séparation aléatoire de l’ensemble d’apprentissage en K folds, en l’occurrence 5, c’est-à-dire le nombre de groupes dans lequel l’échantillon sera réparti. L’ajustement du modèle est effectué avec les folds K-1, le modèle est validé sur le K fold restant. Tous les scores et les erreurs sont notées à chaque K fold, on répète le processus jusqu’à que tous les K-folds aient servi dans l’ensemble d’apprentissage. On peut désormais calculer la moyenne des scores enregistrés, soit la métrique de performance du modèle.

Le schéma ci-dessous issu de la documentation de scikit-learn permet de mieux visualiser cette méthode :



# Algorithme d’optimisation et métrique d’évaluation

## Métriques d’évaluation

Dans cette analyse, nous avons principalement pris en compte deux métriques, le F1-score et le FBeta score.

Nous nous sommes aussi appuyés sur les matrices de confusion, représentées comme suit :

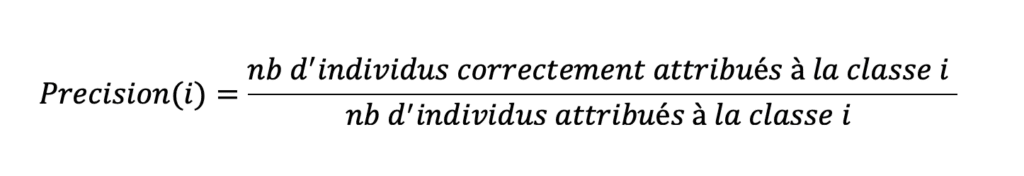
|  |  |  |  |
| --- | --- | --- | --- |
|  |  | **Classe prédite** | |
|  |  | 0 (positif) | 1 (négatif) |
| **Classe actuelle** | 0 (positif) | Vrai positif (TP) | Faux positif (FP) |
| 1 (negatif) | Faux négatif (FN) | Vrai négatif (TN) |

La matrice de confusion permet de connaître les prédictions correctes et incorrectes et nous donnera un indice sur le type d’erreurs commises.

Les quatre catégories du tableau se définissent par :

* **True Positive (TP)** : la prédiction concorde avec la classe actuelle (pour la classe A). La prédiction et la valeur réelle sont positives. Dans notre cas, des personnes n’ayant aucun risque de défaut (classe : 0) sont prédites comme telles
* **True Negative (TN) :** La prédiction et la valeur réelle sont négatives (classe B). Dans notre cas, des personnes ayant un risque de défaut sont prédites comme telles.
* **False Positive (FP) :** La prédiction ne concorde pas avec la classe réelle. La prédiction indique la classe A alors que l’individu est de la B. Il s’agirait qu’une personne ayant un risque de défaut soit prédite comme n’ayant pas de risque de défaut.
* **False Negative (FN) :** La prédiction est négative alors que la classe réelle est positive. Dans notre cas, des personnes n’ayant aucun risque de défaut seraient prédites comme ayant un risque.

La précision correspond au nombre de documents correctement attribués à la classe i par rapport au nombre total de documents prédits comme attribués à la classe i (total predicted positive). Dans notre exemple, il s’agit du nombre de fois où un client n'a vraiment pas remboursé son prêt (c’était vraiment lui) par rapport au nombre de fois où il a été prédit (même s'il l'avait remboursé).

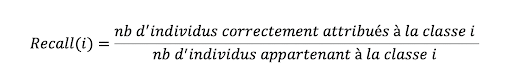


**Precision = TP / (TP+FP)**

La précision permet de mesurer le coût des faux positifs, c’est-à-dire ceux détectés par erreur. Si l’on cherche à limiter les faux positifs, c’est cet indicateur que l’on va chercher à minimiser.

Rappel (=Recall)

Le rappel correspond au nombre de documents correctement attribués à la classe i par rapport au nombre total de documents appartenant à la classe i (total true positive). Le rappel correspond au nombre de fois où un client a été prédit comme "mauvais payeur" par rapport au nombre de fois où il aurait dû être prédit comme tel.

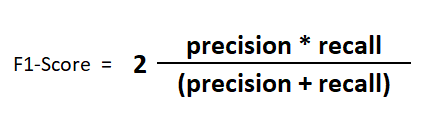


**Recall = tp/(tp+fn)**

Ce calcul permet d’estimer combien de documents réellement positifs nous avons réussi à capturer et combien sont passés au travers des mailles du filet. Maximiser le rappel implique qu’on minimisera les faux négatifs.

Le recall est aussi appelé sensibilité (sensitivity).

Le F1-Score combine subtilement la précision et le rappel. Il est intéressant et plus intéressant que l’accuracy car le nombre de vrais négatifs (tn) n’est pas pris en compte. Et dans les situations de données déséquilibrées, nous avons une majorité de vrais négatifs qui faussent complètement notre perception de la performance de l’algorithme. Un grand nombre de vrais négatifs (tn) laissera le F1-Score de marbre.



Le F1-score (F-mesure) est en réalité la moyenne harmonique de la précision et du rappel, chacun a le même poids. Ainsi, un F-score élevé implique une précision et un rappel élevés.

Quant au FBeta-score, il s’agit de la F-mesure généralisé à laquelle on a rajouté une constante . Par défaut, celle-ci vaut 1, ce qui équivaut au F1-score. Une valeur plus petite de , comme 0.5, donne plus de poids à la précision et moins au rappel. Une valeur plus grande de , comme 2 donnera moins de poids à la précision et plus au rappel.

**Fbeta = ((1 + beta^2) \* Precision \* Recall) / (beta^2 \* Precision + Recall)**

Ainsi, afin de minimiser l’erreur de faux négatifs, les personnes qui feront défaut mais pas prédites en tant que telles, nous utiliserons le F2-score pour évaluer nos modèles. En effet, nous pensons qu’une personne qui fera défaut mais pas prédite en tant que telle, coûtera plus cher à la société « prêt à dépenser » qu’une personne prédite en défaut mais qui remboursera.

## Optimisation

Pour choisir entre différents modèles, trois au total, à savoir :

1. le modèle naïf par la régression logistique
2. RandomForest
3. LightGBM

Pour chaque modèle, on a optimisé les hyper-paramètres afin de maximiser le f1-score d’un côté et le F2-score de l’autre.

Chaque modèle a été évalué avec et sans SMOTE (sans SMOTE impliquant avec pondération), tous les modèles utilisés ont une option de pondération propre.

Les modèles ont aussi été évalués avec et sans Feature engineering, c’est-à-dire avec ou sans les variables polynomiales et les variables de domaines. Par contre, les variables polynomiales n’ont été utilisées que pour la régression logistique étant donné qu’il s’agit du seul modèle linéaire.

Notre modèle de référence est le modèle naïf, en l'occurrence la régression logistique, avec les scores F1 et F2 associés pour les hyper-paramètres choisis.

Notre deuxième modèle est la forêt aléatoire qui a été optimisé de la même manière que le modèle naïf, en faisant varier les hyper-paramètres et en calculant les scores F1 et F2 associés.

Le troisième modèle, LightGBM a été optimisé avec optuna qui est un système d’optimisation d’hyper-paramètres.

Nous avons ainsi obtenu le tableau de résultats suivant :

RF signifie RandomForestClassifier et FE signifie Feature Engineering

|  |  |  |  |
| --- | --- | --- | --- |
| **model** | **Logistique** | **RF** | **LGBM** |
| **smote** | False | False | False |
| **FE** | False | False | True |
| **C** | 1000 | None | NaN |
| **subsample** | None | None | 1 |
| **max\_depth** | None | 5 | 5 |
| **num\_leaves** | None | None | 663 |
| **min\_child\_samples** | None | None | 144 |
| **min\_child\_weight** | None | None | 1 |
| **reg\_lambda** | None | None | 0,00 |
| **F1** | 0,2573 | 0,2418 | 0,2696 |
| **F2** | 0,4091 | 0,3917 | 0,4216 |

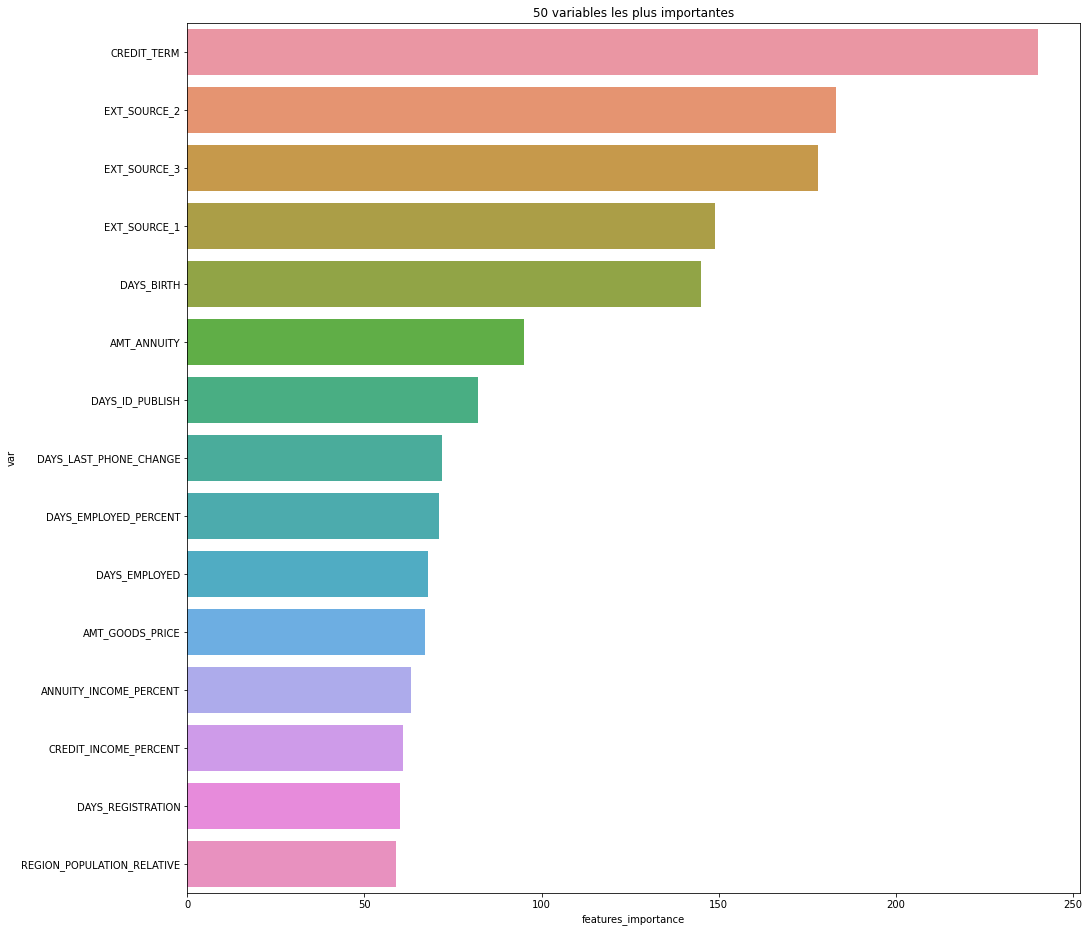
Le meilleur modèle est LightGBM avec pondération et intégrant les variables de domaine.

On remarque que SMOTE pénalise généralement le modèle et que notre modèle naïf n’était pas si mauvais.

# Interprétabilité du modèle

## Importance des variables

Les 50 variables les plus importantes sont :



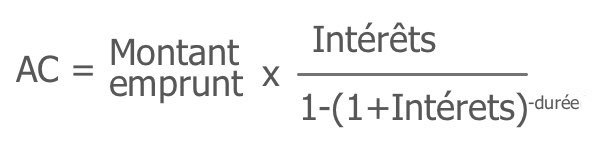
Dans les cinq premières variables les plus pertinentes selon notre modèle, on retrouve CREDIT\_TERM, une variable issue du domaine bancaire, qui représente la durée du paiement en mois.

On retrouve ensuite les trois variables bancaires issues d’une source extérieure

La quatrième variable est la variable d’âge qui influe sur le défaut de paiement (positivement ou négativement) et la cinquième variable est AMT\_ANNUITY, qui représente l’annuité du prêt, c’est-à-dire le paiement annuel d’une partie du capital emprunté et des intérêts.

Étant donné que les intérêts d’un emprunt sont calculés à l'année, une annuité représente ce qu'il faut donner à notre prêteur une année.

La formule utilisée pour calculer l'annuité constante est la suivante:



## Matrice de confusion

La matrice de confusion sur l’ensemble de test se présente comme suit :

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | **Classe prédite** | |
|  |  | 0 (positif) | 1 (négatif) |
| **Classe actuelle** | 0 (positif) | 29628 | 26291 |
| 1 (negatif) | 1538 | 3416 |

Le nombre de faux négatifs est égale à 1538 (FN = 1538).

Le nombre de vrais négatifs est égal à 3416, VP = 3416.

Le taux de faux négatifs est donc de : FN/(TN+FN) = 31%

Ainsi, dans la classe 1, on retrouve une erreur de prédiction de 31%. On a un risque qu’une personne prédite comme “sans risque” fasse défaut à hauteur de 31%.

Le taux de faux positifs est de : FP/(TP+FP) = 48%

Dans la classe 0, l’erreur de classification est plus importante, à hauteur de 48%. Ainsi, une personne prédite comme à risque aura 47% de chances de ne pas faire défaut.

Le nombre de vrais positifs est égal à TP = 29628

Ainsi le recall vaut : TP/(TP+FN) = 0.95

La précision vaut : TP/(TP+FP) = 0.52

Le F2-score est alors égal à :

# Limites et améliorations

Dans les limites, on pourrait dire que l’optimisation des algorithmes pourraient être améliorés pour la régression logistique et la forêt aléatoire, en faisant une grille de paramètres plus importante.

Dans les variables proposés, 55% présentaient des valeurs manquantes, ce qui demande beaucoup d’imputation et peut représenter un biais.

Dans les améliorations, on aurait pu essayer d’optimiser le nombre de variables, en essayant d’en supprimer et voir si l’algorithme s’améliore ou utiliser RFE.

On aurait aussi pu calculer des proches voisins au client et lui présenter des clients similaires et en quoi ils diffèrent, notamment si leurs prédictions étaient différentes.

Dans le dashboard notamment, on aurait pu faire varier le score à partir duquel le prêt du client est accepté et on aurait aussi pu faire varier les variables importantes en relançant la prédiction pour comprendre comment cela influait et que le client, si possible, prenne des dispositions.