Projet 7 : Implémentez un modèle de scoring

Présenté par :
Bourama FANE
Etudiant Data Scientist

Dirigé par :
Babou M'BAYE
Mentor chez OpenClassrooms

11 Décembre 2023

Sommaire

Plan de la présentation

Problématique

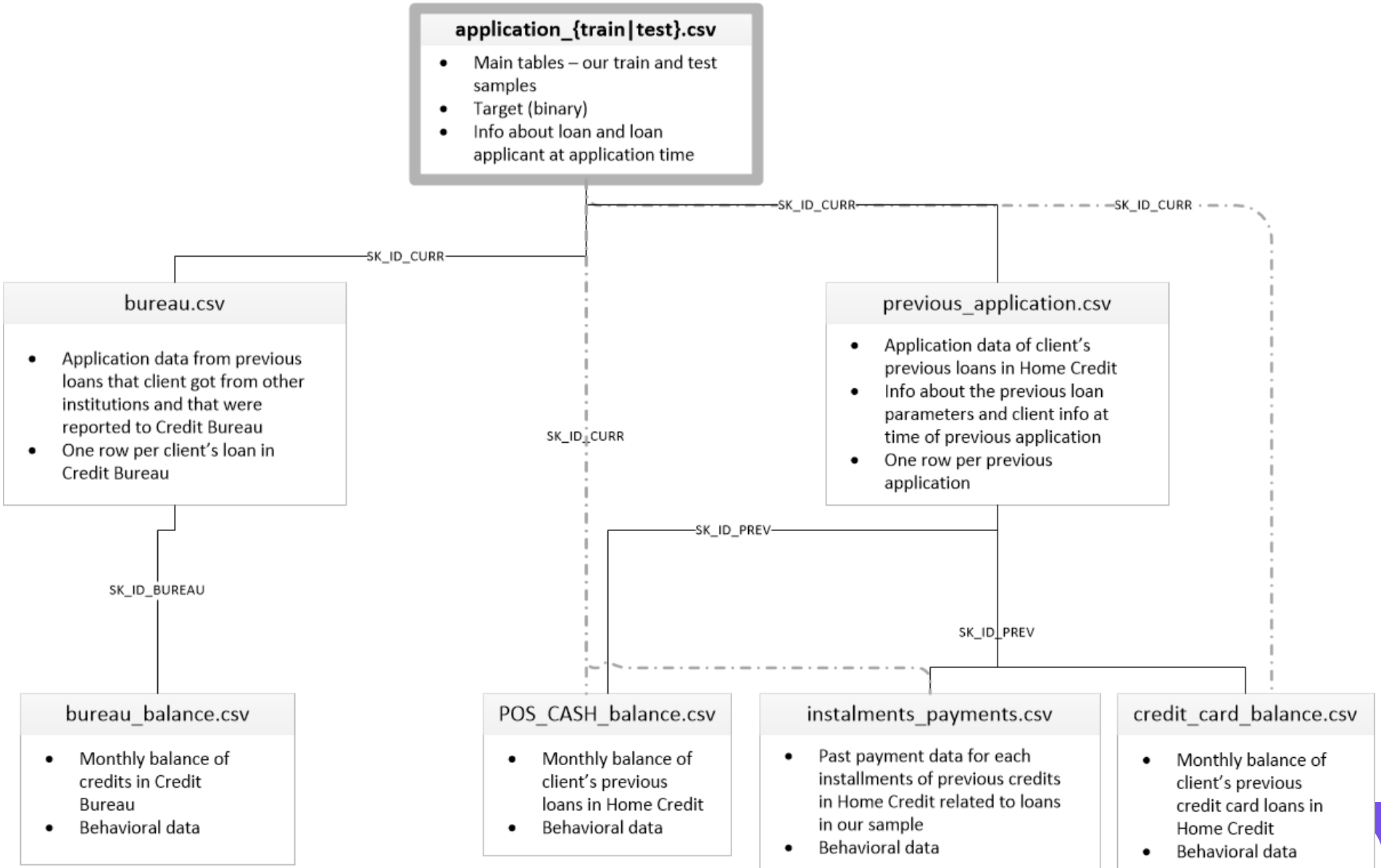
La société financière, nommée **"Prêt à dépenser"**, propose des crédits à la consommation pour des personnes ayant peu ou pas du tout d'historique de prêt. L'entreprise souhaite mettre en œuvre un outil de **scoring crédit** pour calculer la qu'**un client rembourse son crédit**, puis classifie la demande en crédit **accordé** ou **refusé**. Elle souhaite donc développer un algorithme de classification en s'appuyant sur des sources de données variées.

- ✓ identité,
- ✓ données comportementales,
- ✓ données provenant d'autres institutions bancaires,
- ✓ etc.

Objectifs

- ☞ Analyse d'un jeu de données :
  - Nettoyage du jeu de données
  - Recherche du modèle optimal
  - Mise en place d'une métrique adaptée pour la banque
- ☞ API
- ☞ Dashboard interactif
  - Visualisation score et interprétation
  - Informations du client
  - Interprétation prédiction modèle

Données

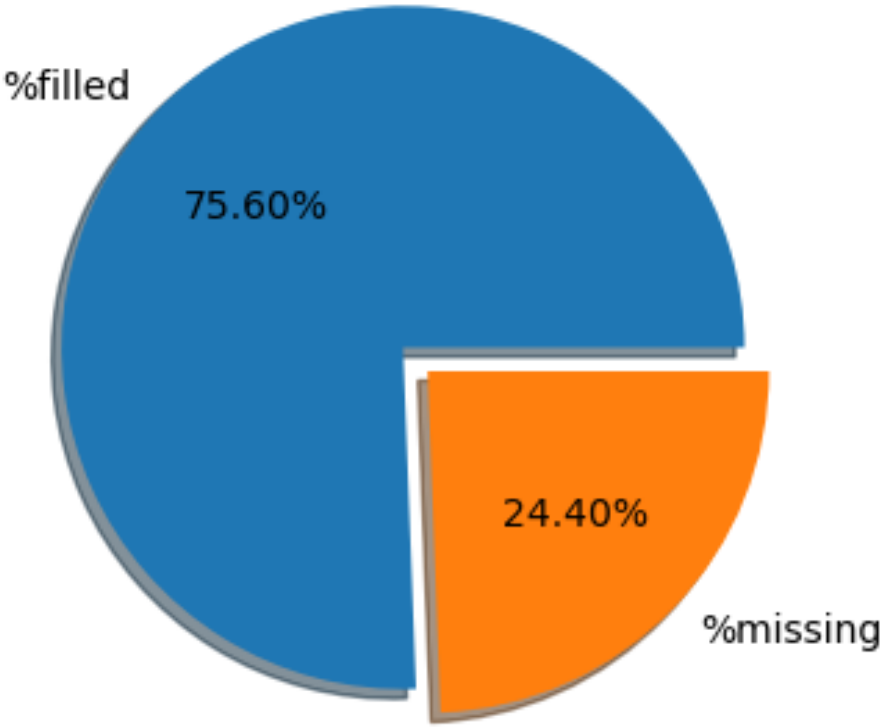

application_{train|test}.csv
Main tables – our train and test samples
Target (binary)
Info about loan and loan applicant at application time
SK_ID_CURR
SK_ID_CURR
SK_ID_CURR
SK_ID_CURR
bureau.csv
Application data from previous loans that client got from other institutions and that were reported to Credit Bureau
One row per client's loan in Credit Bureau
previous_application.csv
Application data of client's previous loans in Home Credit
Info about the previous loan parameters and client info at time of previous application
One row per previous application
SK_ID_BUREAU
SK_ID_PREV
SK_ID_PREV
bureau_balance.csv
Monthly balance of credits in Credit Bureau
Behavioral data
POS_CASH_balance.csv
Monthly balance of client's previous loans in Home Credit
Behavioral data
instalments_payments.csv
Past payment data for each installments of previous credits in Home Credit related to loans in our sample
Behavioral data
credit_card_balance.csv
Monthly balance of client's previous credit card loans in Home Credit
Behavioral data

Caractéristiques des fichiers

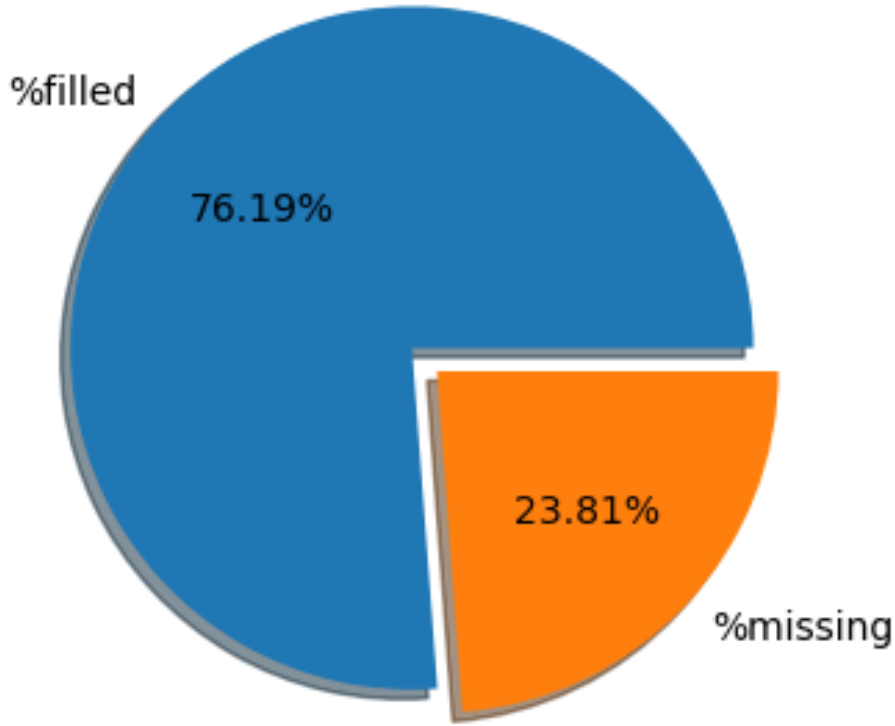
	nb_lignes	nb_cols	%missings	%doublons	object_dtype	float_dtype	int_dtype	bool_dtype	Mo_Memory
application_test.csv	48744	121	23.81	0.0	16	65	40	0	44.998
application_train.csv	307511	122	24.40	0.0	16	65	41	0	286.227
bureau.csv	1716428	17	13.50	0.0	3	8	6	0	222.620
bureau_balance.csv	27299925	3	0.00	0.0	1	0	2	0	624.846
credit_card_balance.csv	3840312	23	6.65	0.0	1	15	7	0	673.883
HomeCredit_columns_description.csv	219	5	12.15	0.0	4	0	1	0	0.008
installments_payments.csv	13605401	8	0.01	0.0	0	5	3	0	830.408
POS_CASH_balance.csv	10001358	8	0.07	0.0	1	2	5	0	610.435
previous_application.csv	1670214	37	17.98	0.0	16	15	6	0	471.481
sample_submission.csv	48744	2	0.00	0.0	0	1	1	0	0.744

Taux de remplissage

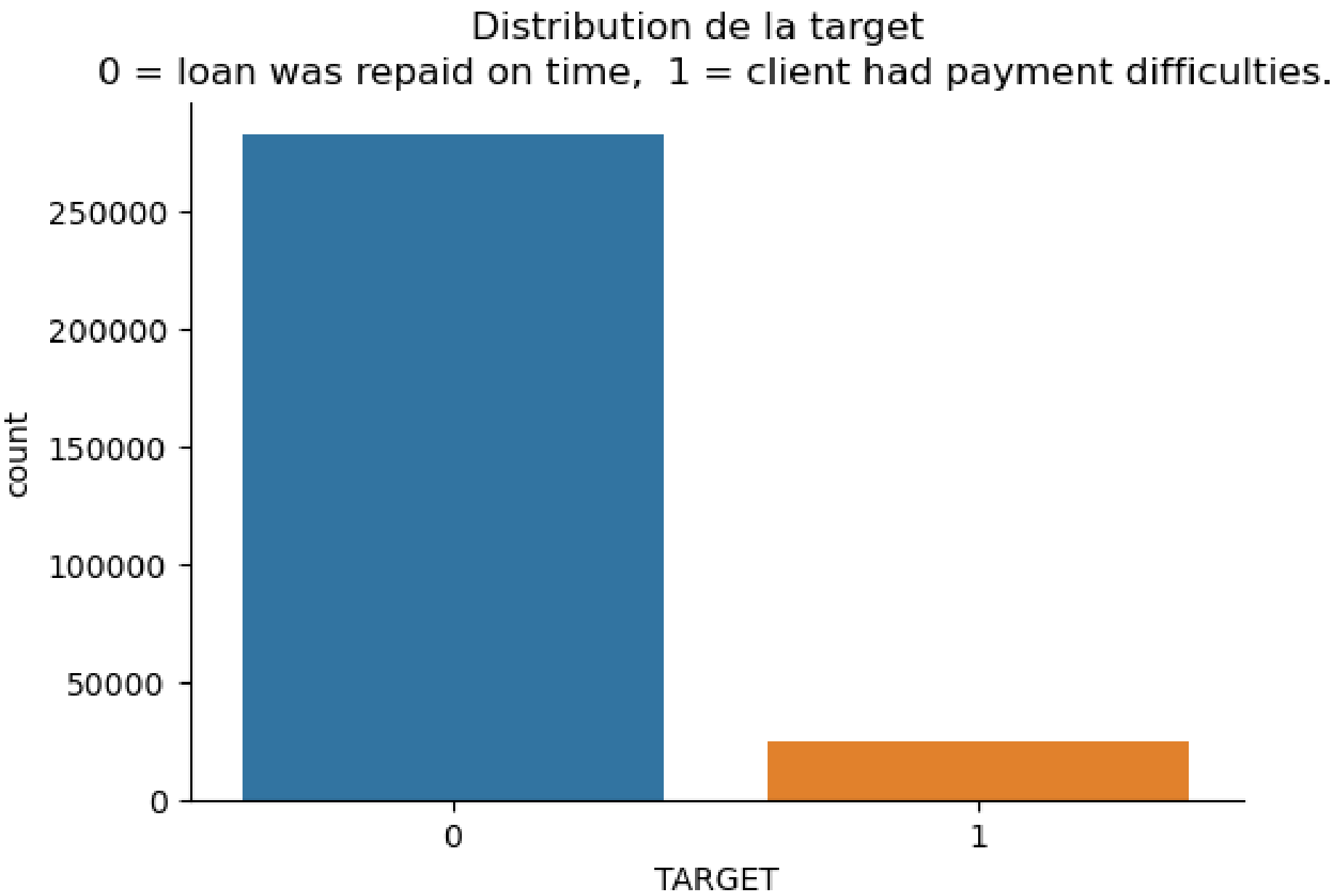
Taux de completion (application_train)



Taux de completion (application_test)



Analyse de la target



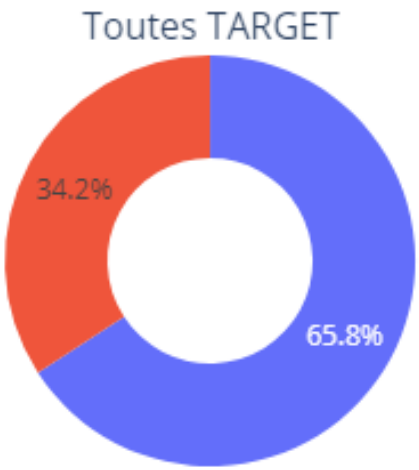
Plan de la présentation

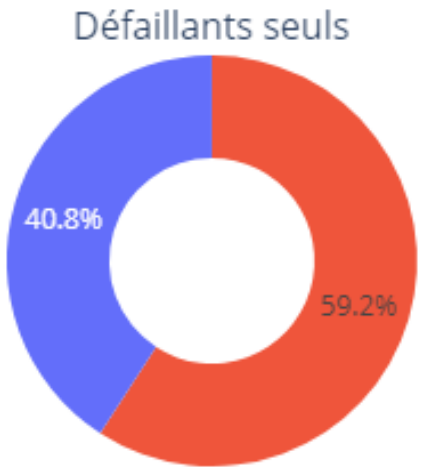
Distribution de la target : NAME_CONTRACT_TYPE

Répartition de la variable NAME_CONTRACT_TYPE

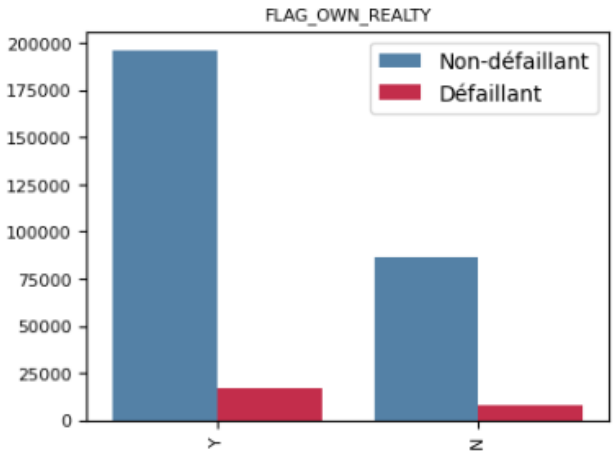
Distribution de la target : CODE_GENDER

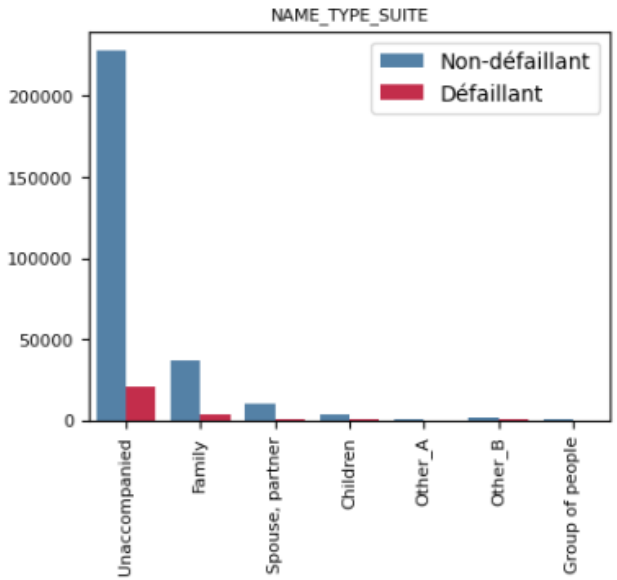
Répartition de la variable CODE_GENDER

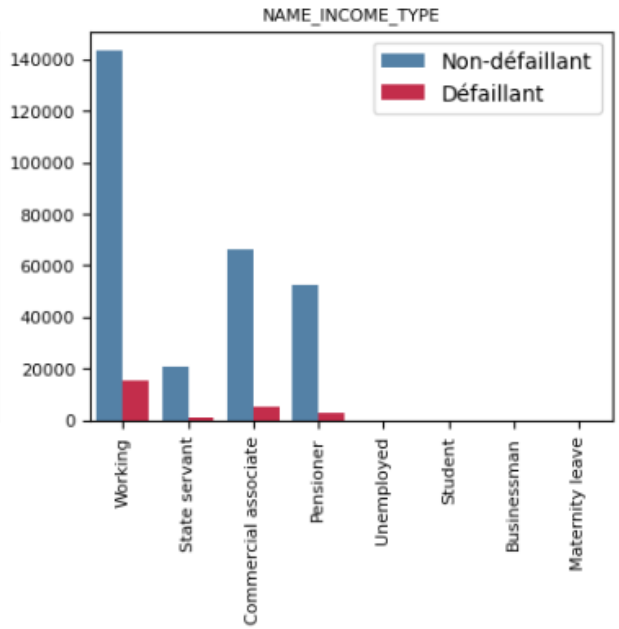




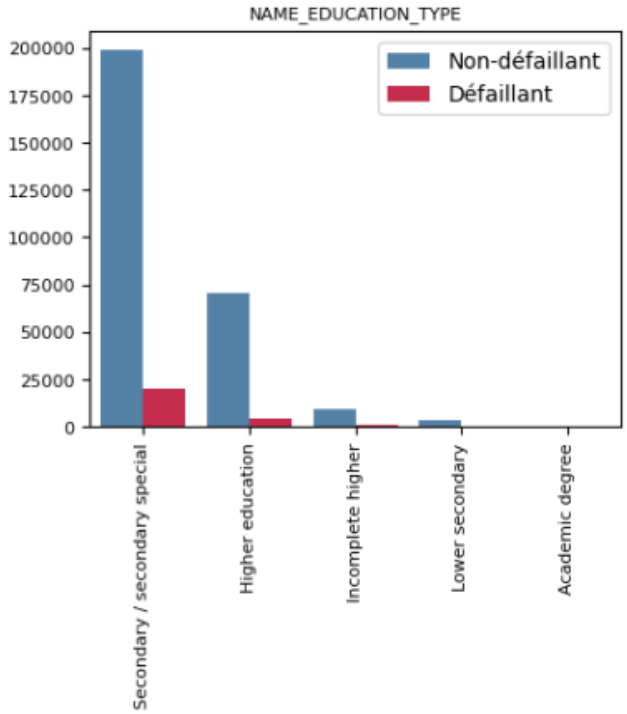
Distribution de la target : Autres variables

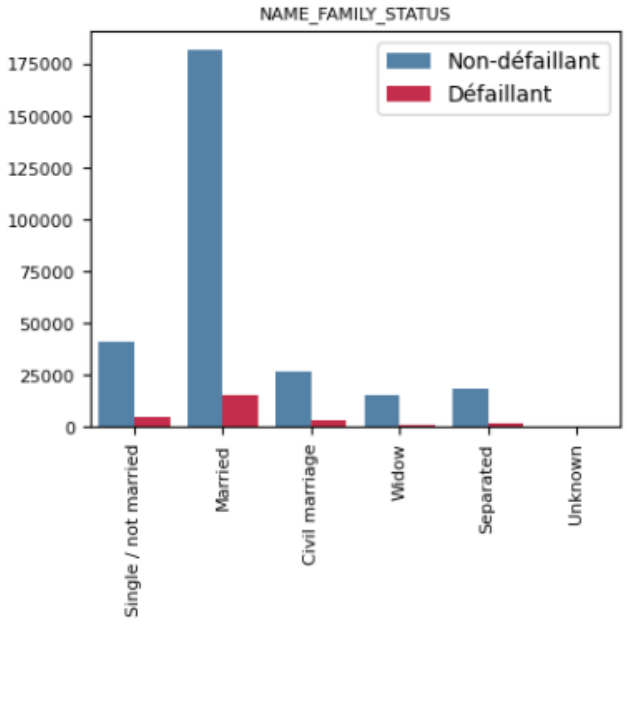


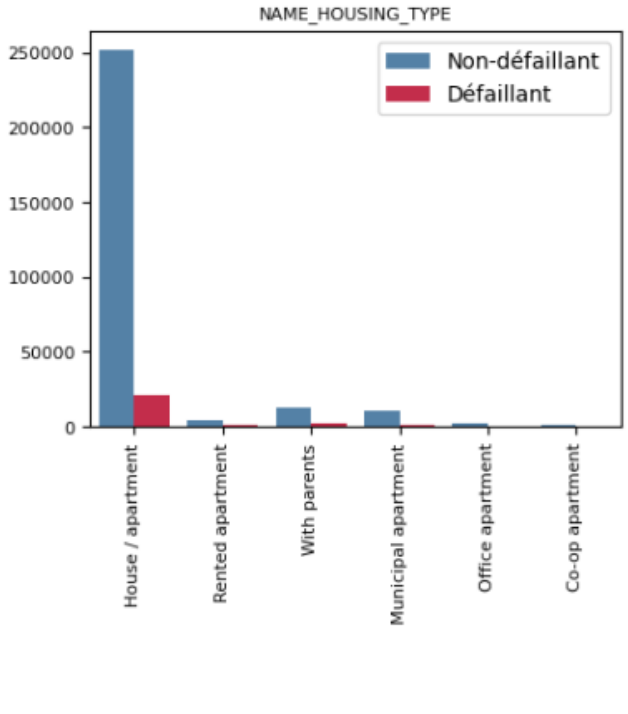


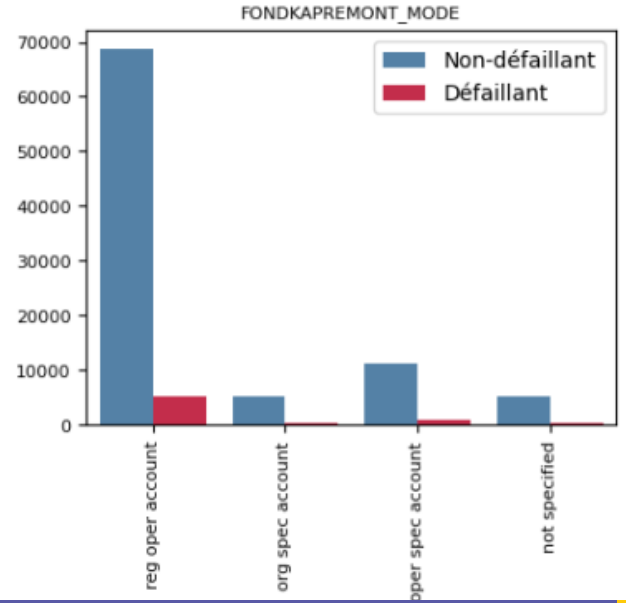


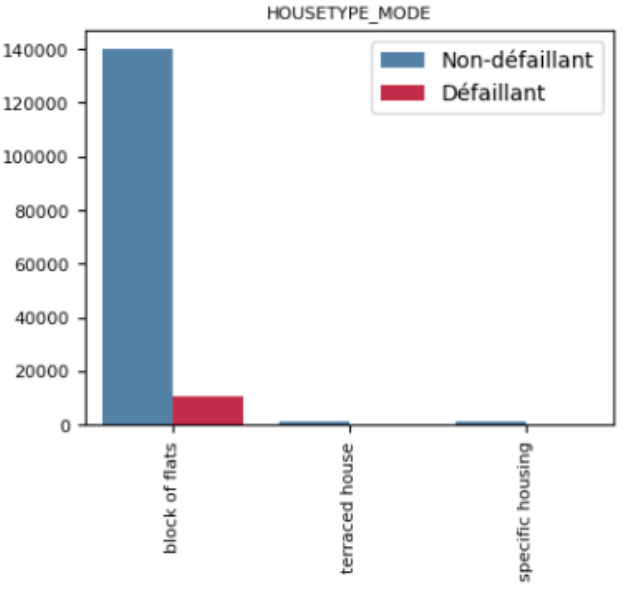
Distribution de la target : Autres variables

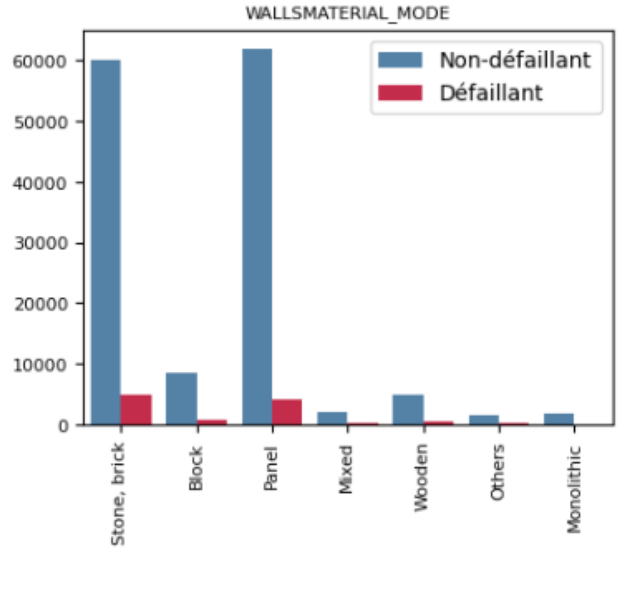












Plan de la présentation

Utilisation du Kernel Kaggle

- ☞ Jointure des tables selon la clé primaire ;
- ☞ Imputation des valeurs manquantes/aberrantes ;
- ☞ Features engineering (création de nouvelles variables) ;
- ☞ Encodage des variables catégorielles ;
- ☞ Agreggation des données par **client** ;

Autres traitements

- ☞ Suppression des colonnes avec plus de **40% de missings** ;
- ☞ Supression des colonnes constantes ;
- ☞ conversion des ages (nombre de jours) en **années** ;
- ☞ changement des valeurs négatives en valeurs positives ;
- ☞ Imputation avec la **méthode interpolate** ;

Selection de features : 6 méthodes

- ☞ Pearson Correlation ;
- ☞ SelectKBest ;
- ☞ RFE (**Recursive Feature Elimination**) ;
- ☞ Logistics Regression L1 ;
- ☞ Random Forest ;
- ☞ LightGBM ;
- ☞ Comparaison des **100 'best' features** selectionnés

Plan de la présentation

Algorithmes de classification

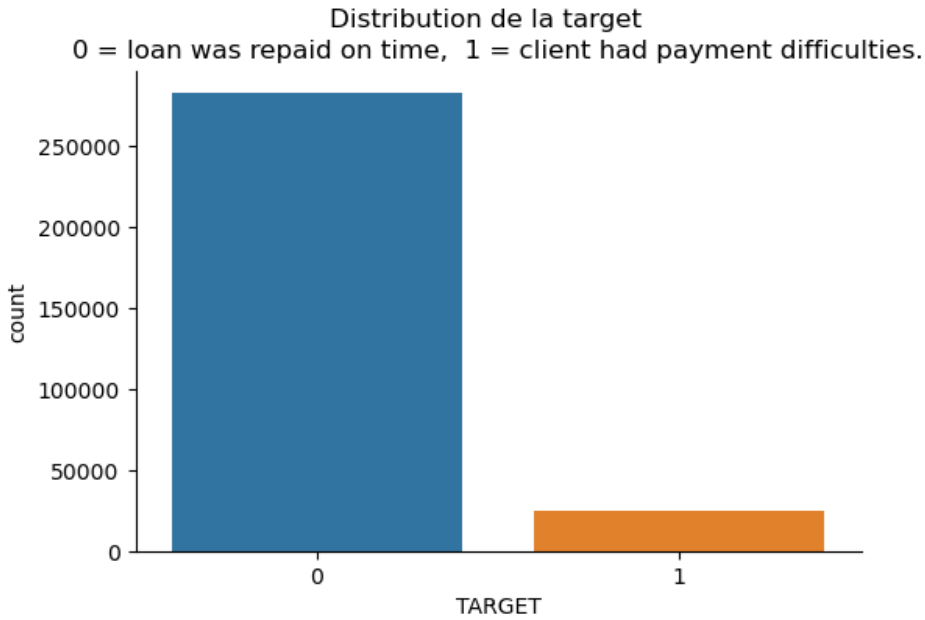
Les Algorithmes

- ☞ Dummy Classifier
- ☞ Logistic Regression
- ☞ SVC
- ☞ Decision Tree
- ☞ Random Forest
- ☞ XG Boost
- ☞ Light GBM

Désequilibre entre classes

Techniques d'équilibrage

- ☞ class_weight
- ☞ SMOTE
- ☞ Tomek Links
- ☞ RandomUnderSampler
- ☞ RandomOverSampler



Les méthodes **RandomUnderSampler** et **class_weight** fournissent les meilleurs scores. Nous optons pour l'approche d'undersampling en utilisant RandomUnderSampler pour équilibrer nos données.

Chaque modèle est entraîné en utilisant ces techniques.

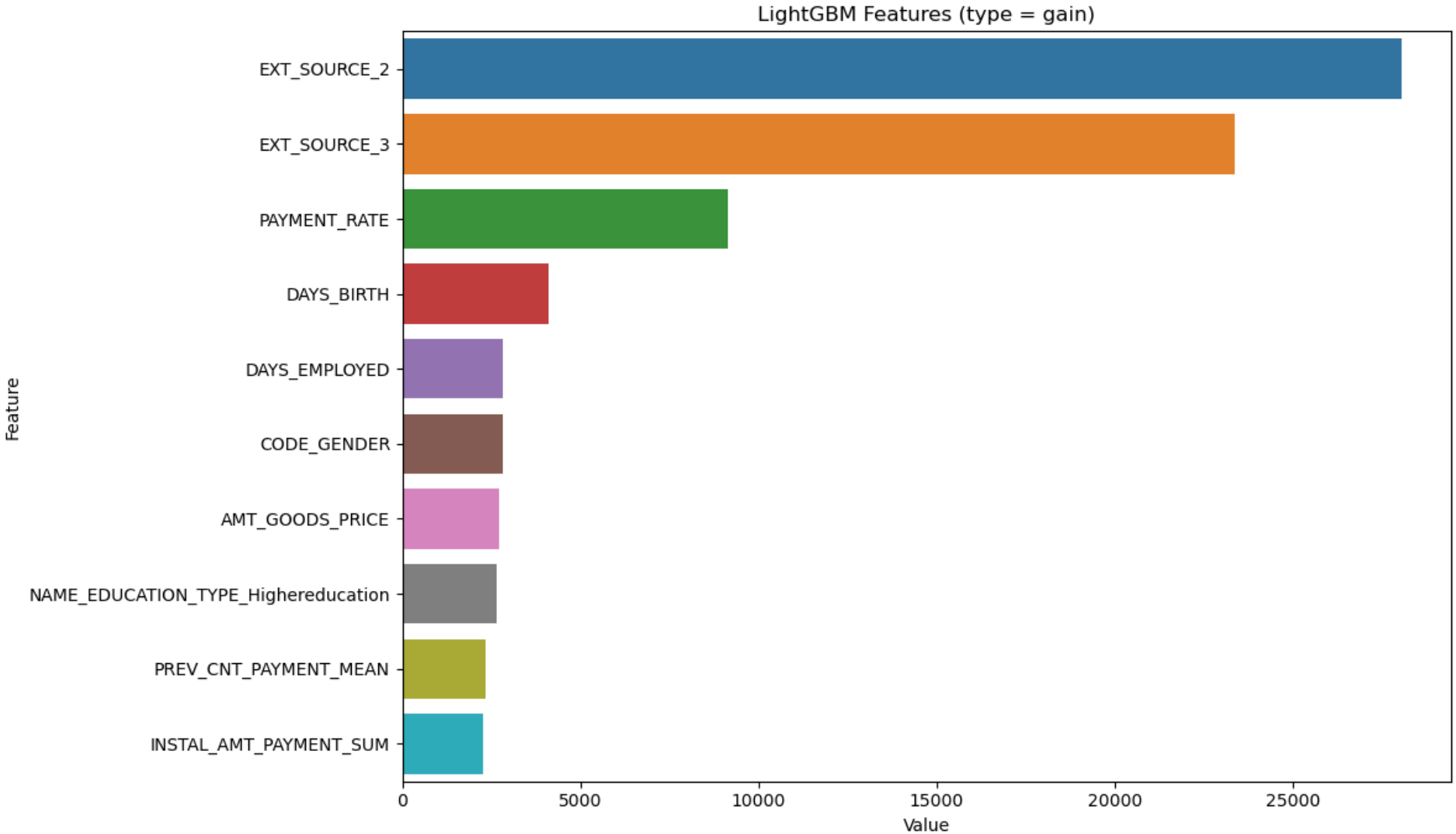
Modèle final et métriques

```
model_performance = pd.concat(models_perf, axis=0)
model_performance.sort_values(by        = ['Recall class 1','AUC', 'score Gain', 'FN'],
                              ascending = [False,            False,  False,        True],
                              inplace   = True, )

model_performance = (model_performance.loc[model_performance.Modele.str.contains('Model', case=False),:]
                     .drop_duplicates())
model_performance.reset_index(drop=True, inplace=True)
model_performance
```

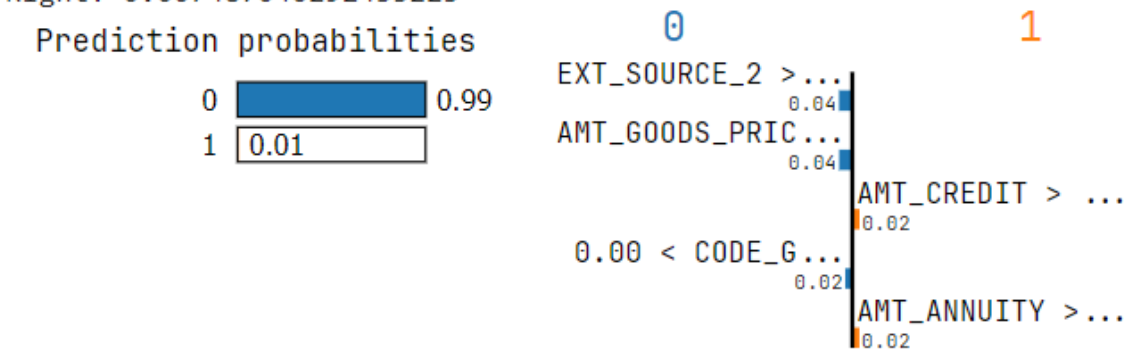
	Modele	Accuracy	AUC	Recall class 1	F1	fbeta	TP	Precision	FN	score Gain	train_time	predict_time
0	Model_LGBMClassifier	0.699815	0.765181	0.696073	0.272405	0.429114	3456	0.169337	1509	0.698336	6.136364	1.406532
1	Model_LGBMClassifier	0.699815	0.765181	0.696073	0.272405	0.429114	3456	0.169337	1509	0.698336	6.233220	1.624582
2	Model_SVC	0.688986	0.752176	0.689829	0.263685	0.418981	3425	0.162994	1540	0.689319	2533.136927	300.204419
3	Model_LogisticRegression	0.691717	0.751976	0.681974	0.263174	0.416718	3386	0.163047	1579	0.687868	7.234787	1.064942
4	Model_XGBClassifier	0.687929	0.745665	0.676737	0.259329	0.411684	3360	0.160397	1605	0.683507	10.105386	1.399076
5	Model_RandomForestClassifier	0.691376	0.740535	0.667472	0.258815	0.409095	3314	0.160531	1651	0.681932	98.769776	3.561240
6	Model_DecisionTreeClassifier	0.586290	0.588409	0.590937	0.187404	0.317484	2934	0.111360	2031	0.588126	14.551581	0.949493

Interpretabilité globale :Features importances



Interpretabilité locale : LIME & Shap.force

Le client 100009 est selectionné
Intercept 0.11387343997572051
Prediction_local [0.04884055]
Right: 0.007487046292433223



Feature	Value
EXT_SOURCE_2	0.72
AMT_GOODS_PRICE	1395000.00
AMT_CREDIT	1560726.00
CODE_GENDER	1.00
AMT_ANNUITY	41301.00

Le client 100009 est selectionné

LightGBM binary classifier with TreeExplainer shap values output has changed to a list of ndarray

Github & API & Dashboard

Github

https://github.com/bouramayaya/OC-Projet-7

API

http://54.172.177.114:8000/

http://54.172.177.114:8000/docs

Dashboard

http://54.172.177.114:8080/

Plan de la présentation

Conclusion

- ☞ Utilisation du **kernel Kaggle** fourni dans les ressources ;
- ☞ Selection de **100 Variables (de façon arbitraire)** ;
- ☞ lightGBM a été le modele final retenu ;
- ☞ Un score AUC autour de **0.77**.

MERCI POUR VOTRE AIMABLE ATTENTION