PROJET N°7 :

IMPLÉMENTEZ UN MODÈLE DE SCORING

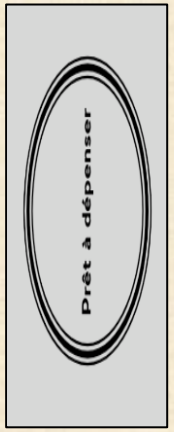
Soutenance du P7: le 22/11/2022

Version notebook : **6.3.0**
Version Python : **3.8.8**
Version Pandas : **1.2.4**
Version Seaborn : **0.11.1**
Version Matplotlib: **3.3.4**

Plan

- Contexte et présentation des Data-Set
- Analyse exploratoire
- Traitement et nettoyage des Data-Sets « Feature engineer »
- Modélisation
- Modèles retenu
- Présentation du Dashboard
- Conclusion

Contexte et présentation des Data-Set



Contexte:

- Une société financière, nommée "**Prêt à dépenser**", qui 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 probabilité 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 (données comportementales, données provenant d'autres institutions financières.

Mission:

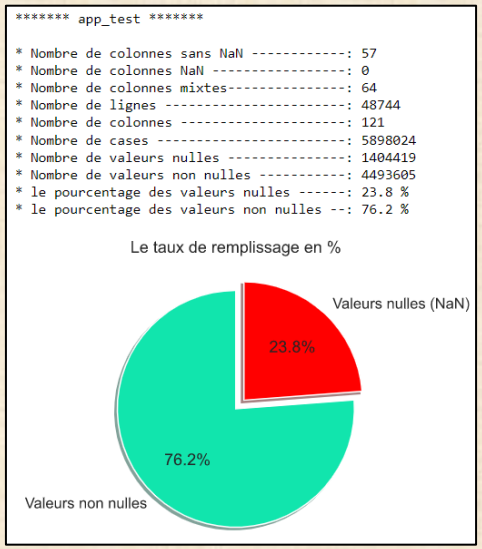
- Construire un **modèle de scoring** qui donnera une prédiction sur la probabilité de faillite d'un client de façon automatique.
- Construire un **Dashboard** interactif à destination des gestionnaires de la relation client permettant d'interpréter les prédictions faites par le modèle, et d'améliorer la connaissance client des chargés de relation client.

Contexte et présentation des Data-Set

Présentation des tables de données:

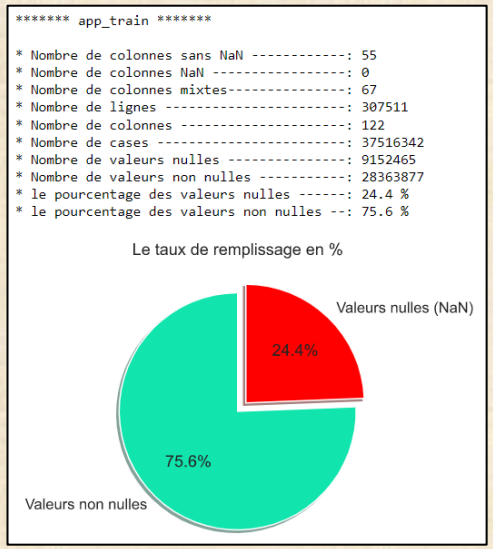
app_test

```
******* app_test *******

* Nombre de colonnes sans NaN ------------: 57
* Nombre de colonnes NaN -----------------: 0
* Nombre de colonnes mixtes---------------: 64
* Nombre de lignes -----------------------: 48744
* Nombre de colonnes ---------------------: 121
* Nombre de cases ------------------------: 5898024
* Nombre de valeurs nulles ---------------: 1404419
* Nombre de valeurs non nulles -----------: 4493605
* le pourcentage des valeurs nulles ------: 23.8 %
* le pourcentage des valeurs non nulles --: 76.2 %
```



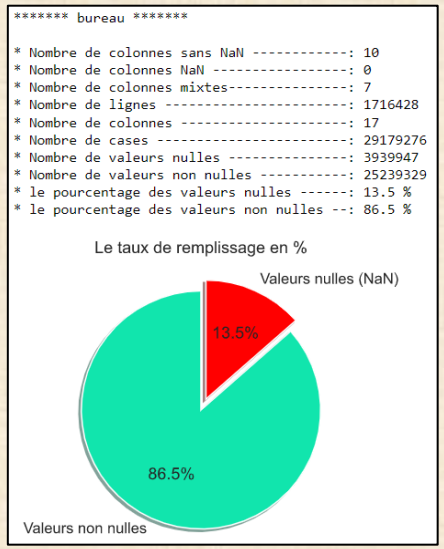
app_train

```
******* app_train *******

* Nombre de colonnes sans NaN ------------: 55
* Nombre de colonnes NaN -----------------: 0
* Nombre de colonnes mixtes---------------: 67
* Nombre de lignes -----------------------: 307511
* Nombre de colonnes ---------------------: 122
* Nombre de cases ------------------------: 37516342
* Nombre de valeurs nulles ---------------: 9152465
* Nombre de valeurs non nulles -----------: 28363877
* le pourcentage des valeurs nulles ------: 24.4 %
* le pourcentage des valeurs non nulles --: 75.6 %
```



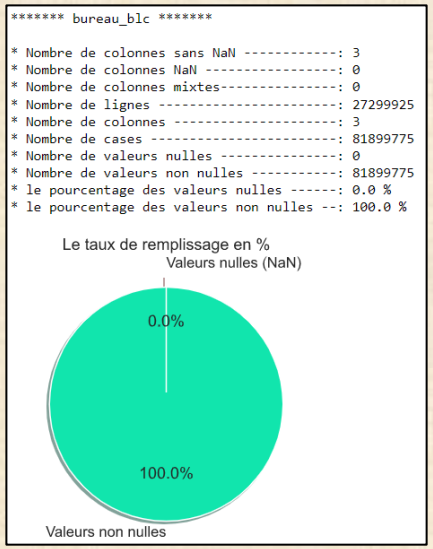
bureau

```
******* bureau *******

* Nombre de colonnes sans NaN ------------: 10
* Nombre de colonnes NaN -----------------: 0
* Nombre de colonnes mixtes---------------: 7
* Nombre de lignes -----------------------: 1716428
* Nombre de colonnes ---------------------: 17
* Nombre de cases ------------------------: 29179276
* Nombre de valeurs nulles ---------------: 3939947
* Nombre de valeurs non nulles -----------: 25239329
* le pourcentage des valeurs nulles ------: 13.5 %
* le pourcentage des valeurs non nulles --: 86.5 %
```



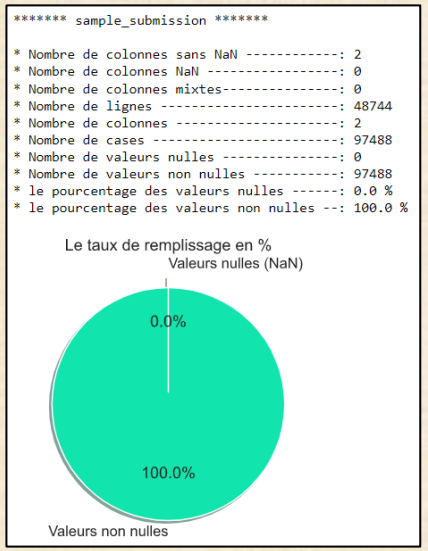
bureau_blc

```
******* bureau_blc *******

* Nombre de colonnes sans NaN ------------: 3
* Nombre de colonnes NaN -----------------: 0
* Nombre de colonnes mixtes---------------: 0
* Nombre de lignes -----------------------: 27299925
* Nombre de colonnes ---------------------: 3
* Nombre de cases ------------------------: 81899775
* Nombre de valeurs nulles ---------------: 0
* Nombre de valeurs non nulles -----------: 81899775
* le pourcentage des valeurs nulles ------: 0.0 %
* le pourcentage des valeurs non nulles --: 100.0 %
```



sample_submission

```
******* sample_submission *******

* Nombre de colonnes sans NaN ------------: 2
* Nombre de colonnes NaN -----------------: 0
* Nombre de colonnes mixtes---------------: 0
* Nombre de lignes -----------------------: 48744
* Nombre de colonnes ---------------------: 2
* Nombre de cases ------------------------: 97488
* Nombre de valeurs nulles ---------------: 0
* Nombre de valeurs non nulles -----------: 97488
* le pourcentage des valeurs nulles ------: 0.0 %
* le pourcentage des valeurs non nulles --: 100.0 %
```

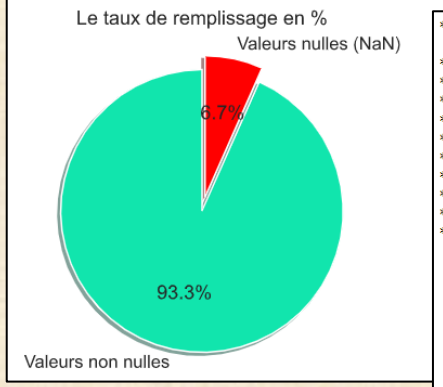


Contexte et présentation des Data-Set

Présentation des tables de données:

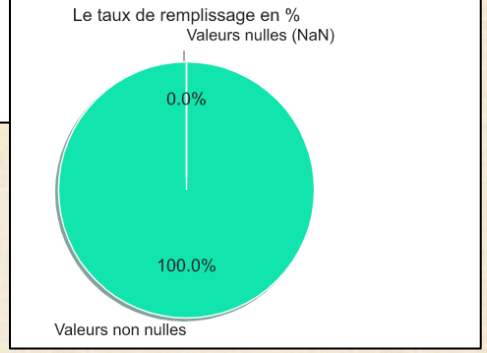
credit_card_blc

```
******* credit_card_blc *******

* Nombre de colonnes sans NaN ------------: 14
* Nombre de colonnes NaN -----------------: 0
* Nombre de colonnes mixtes---------------: 9
* Nombre de lignes -----------------------: 3840312
* Nombre de colonnes ---------------------: 23
* Nombre de cases ------------------------: 88327176
* Nombre de valeurs nulles ---------------: 5877356
* Nombre de valeurs non nulles -----------: 82449820
* le pourcentage des valeurs nulles ------: 6.7 %
* le pourcentage des valeurs non nulles --: 93.3 %
```



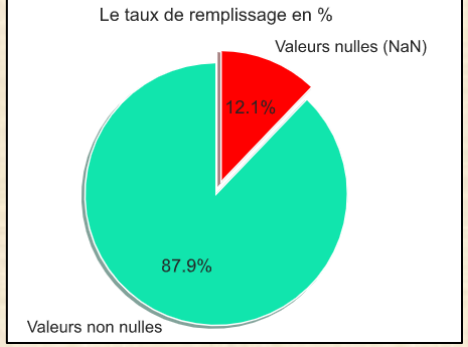
inst_payments

```
******* inst_payments *******

* Nombre de colonnes sans NaN ------------: 6
* Nombre de colonnes NaN -----------------: 0
* Nombre de colonnes mixtes---------------: 2
* Nombre de lignes -----------------------: 13605401
* Nombre de colonnes ---------------------: 8
* Nombre de cases ------------------------: 108843208
* Nombre de valeurs nulles ---------------: 5810
* Nombre de valeurs non nulles -----------: 108837398
* le pourcentage des valeurs nulles ------: 0.0 %
* le pourcentage des valeurs non nulles --: 100.0 %
```



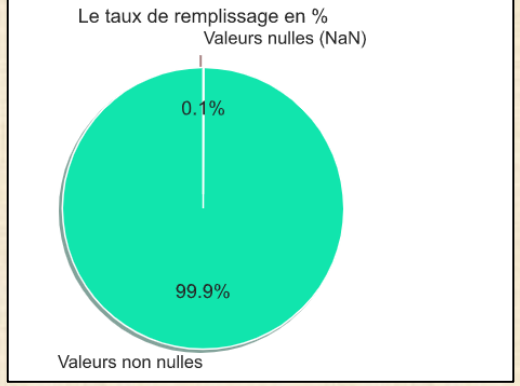
HomeCredit_clmns_desc

```
******* HomeCredit_clmns_desc *******

* Nombre de colonnes sans NaN ------------: 4
* Nombre de colonnes NaN -----------------: 0
* Nombre de colonnes mixtes---------------: 1
* Nombre de lignes -----------------------: 219
* Nombre de colonnes ---------------------: 5
* Nombre de cases ------------------------: 1095
* Nombre de valeurs nulles ---------------: 133
* Nombre de valeurs non nulles -----------: 962
* le pourcentage des valeurs nulles ------: 12.1 %
* le pourcentage des valeurs non nulles --: 87.9 %
```



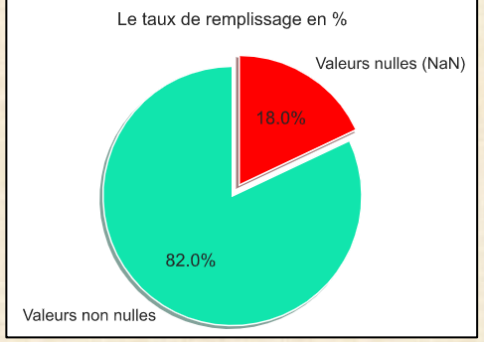
POS_CASH_balance

```
******* POS_CASH_balance *******

* Nombre de colonnes sans NaN ------------: 6
* Nombre de colonnes NaN -----------------: 0
* Nombre de colonnes mixtes---------------: 2
* Nombre de lignes -----------------------: 10001358
* Nombre de colonnes ---------------------: 8
* Nombre de cases ------------------------: 80010864
* Nombre de valeurs nulles ---------------: 52158
* Nombre de valeurs non nulles -----------: 79958706
* le pourcentage des valeurs nulles ------: 0.1 %
* le pourcentage des valeurs non nulles --: 99.9 %
```



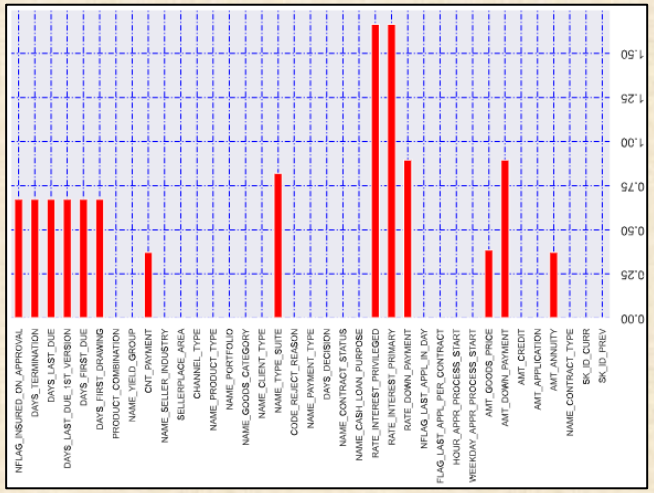
previous_app

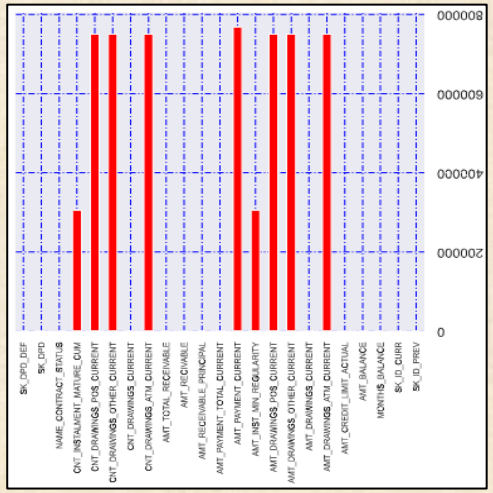
```
******* previous_app *******

* Nombre de colonnes sans NaN ------------: 21
* Nombre de colonnes NaN -----------------: 0
* Nombre de colonnes mixtes---------------: 16
* Nombre de lignes -----------------------: 1670214
* Nombre de colonnes ---------------------: 37
* Nombre de cases ------------------------: 61797918
* Nombre de valeurs nulles ---------------: 11109336
* Nombre de valeurs non nulles -----------: 50688582
* le pourcentage des valeurs nulles ------: 18.0 %
* le pourcentage des valeurs non nulles --: 82.0 %
```

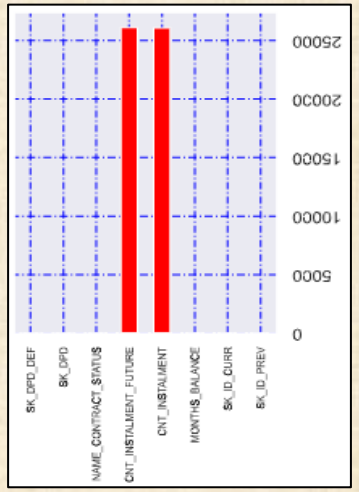


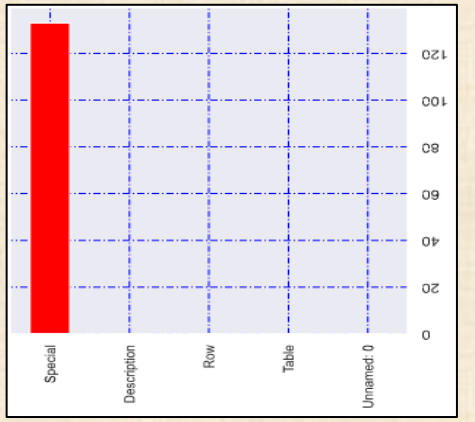
Analyse exploratoire

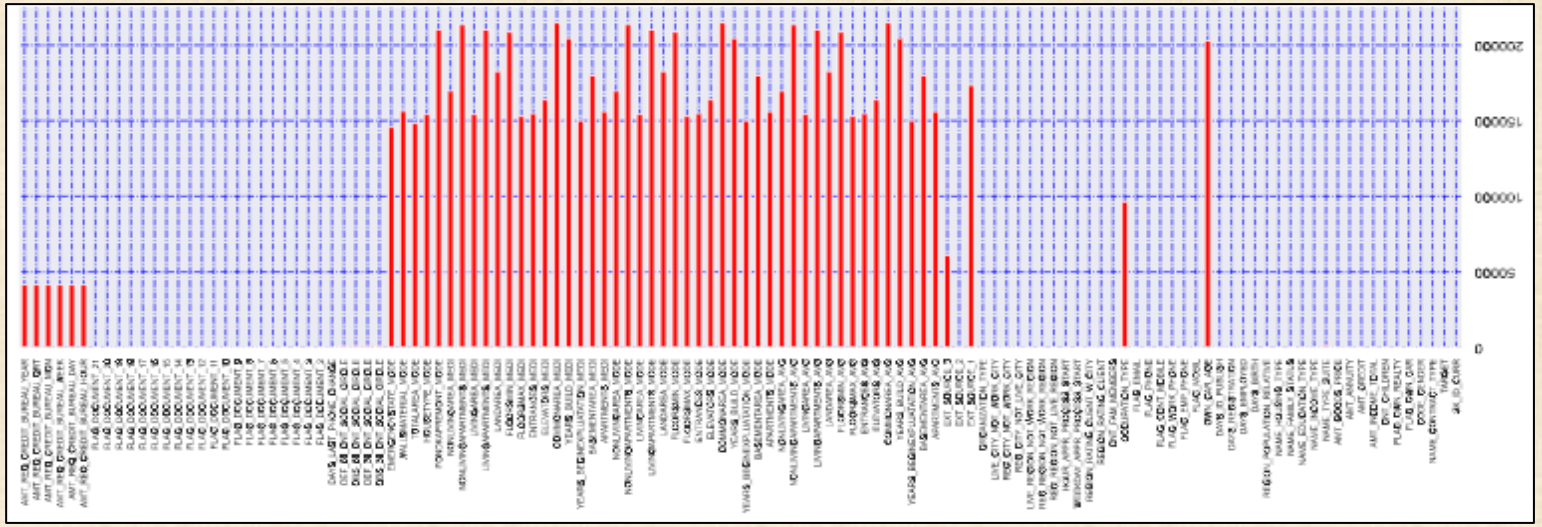
Distribution des valeurs NaN par Features

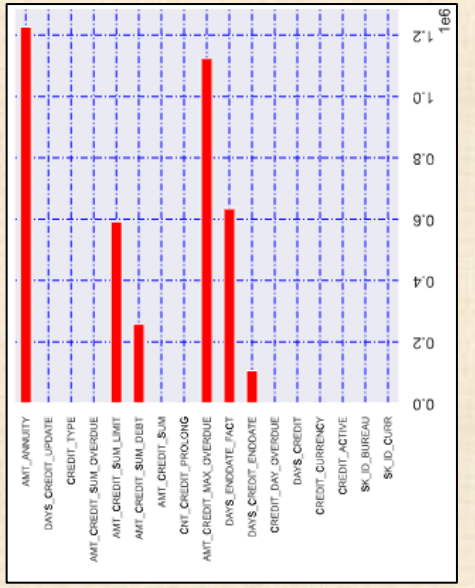










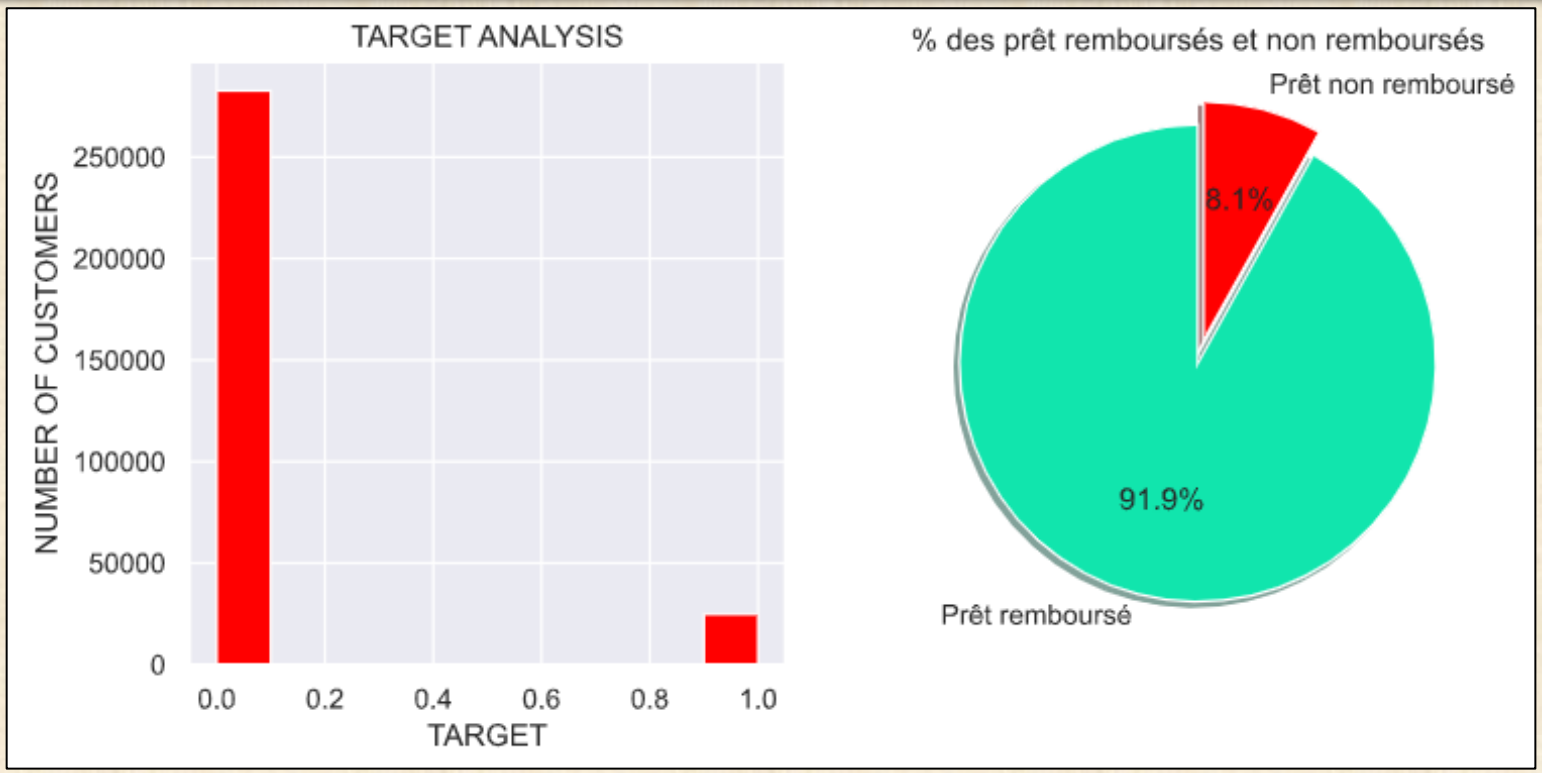


Comme constaté le taux de valeurs absentes est assez important, donc avant de procéder la construction des modèles nous devrons compléter ces valeurs manquantes par imputation et une deuxième solution est de supprimer les colonnes qui contiennent beaucoup de valeurs NaN, mais il est impossible de savoir à l'avance si ces colonnes seront utiles à notre modèle. Par conséquent, nous conserverons toutes les colonnes pour le moment.

Analyse exploratoire

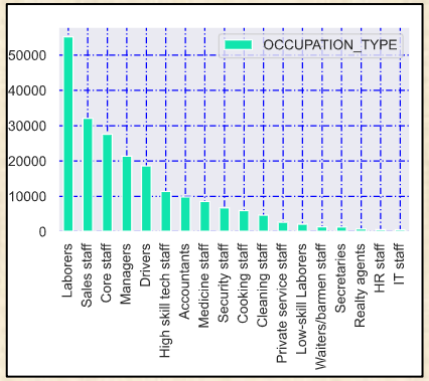
Distribution des classes « Target »

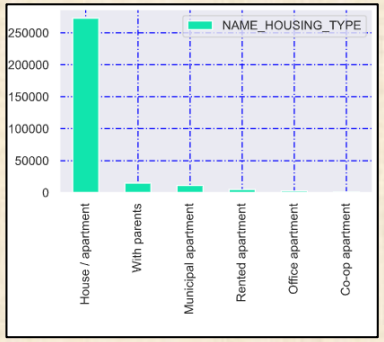
Comme présenté en dessous dans le pie. Il y a beaucoup plus de prêts remboursés à temps que de prêts non remboursés. Lorsque nous utiliserons des modèles d'apprentissage automatique plus sophistiqués, nous pourrons pondérer les classes en fonction de leur représentation dans les données afin de refléter ce déséquilibre. La technique utilisée **SMOTE** oversampling

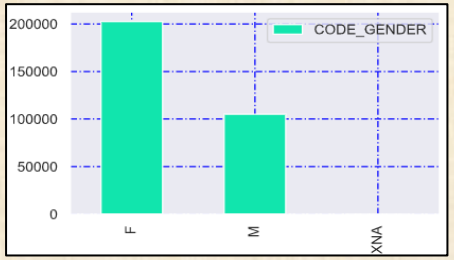


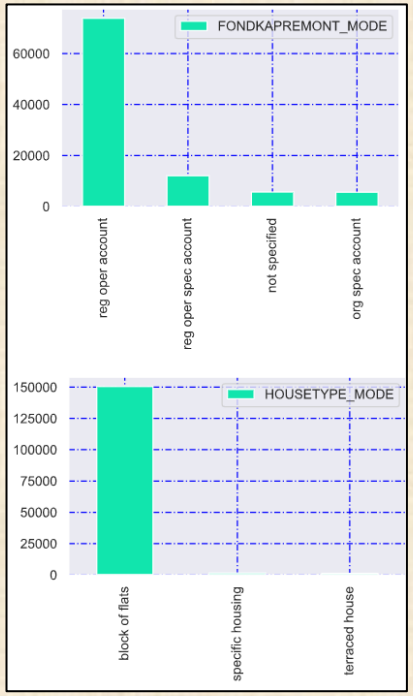
Analyse exploratoire

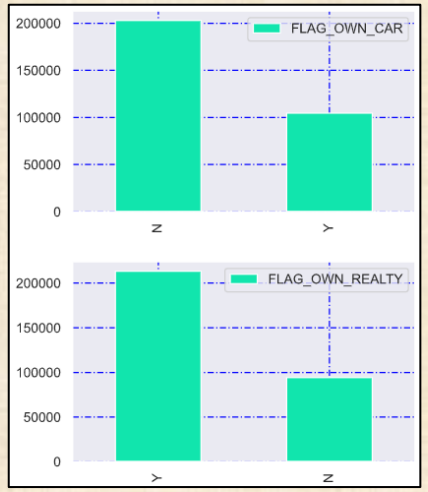
Distribution des variables catégorielles pour app_train

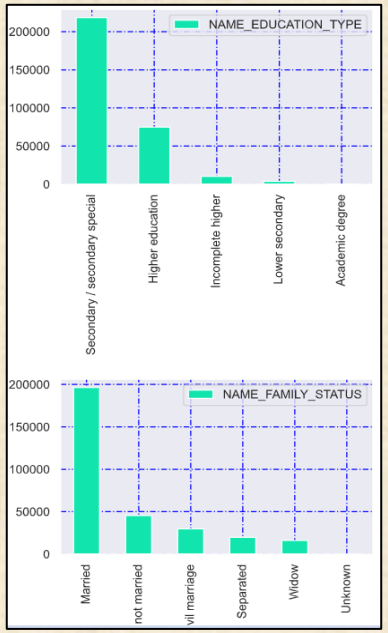


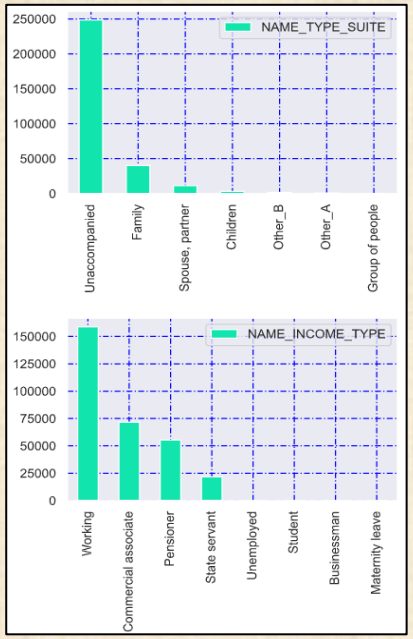


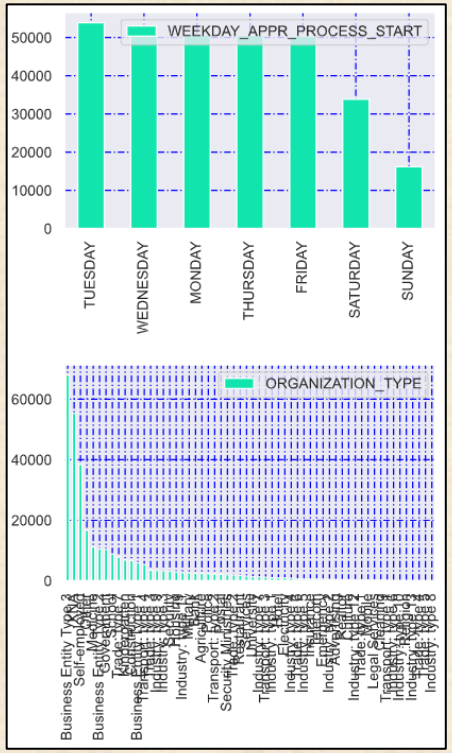


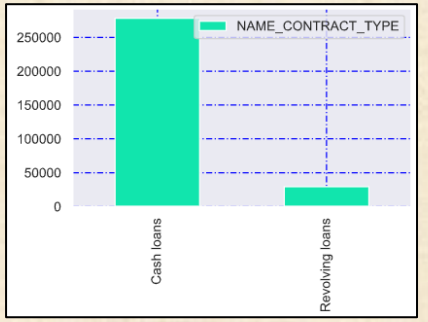


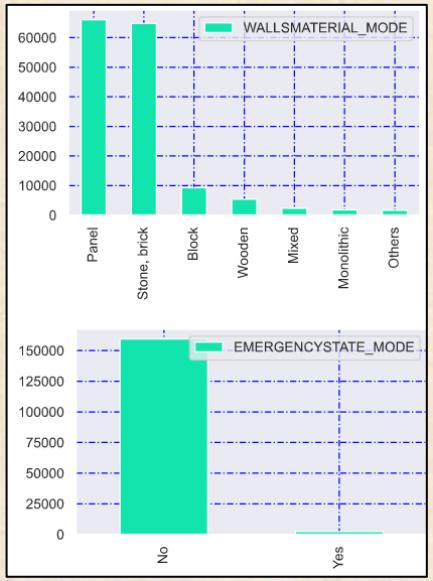






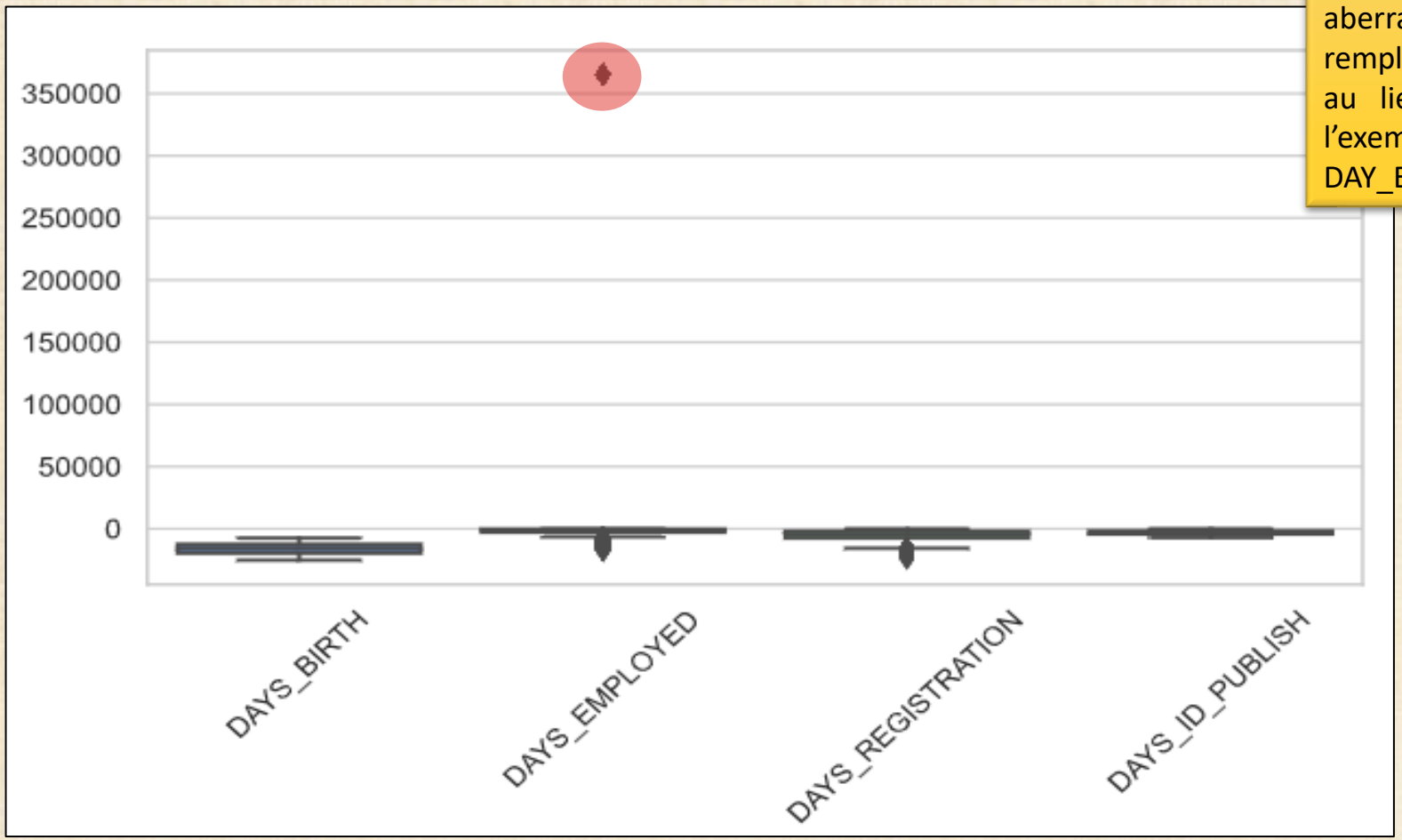






Analyse exploratoire

Valeurs aberrantes:

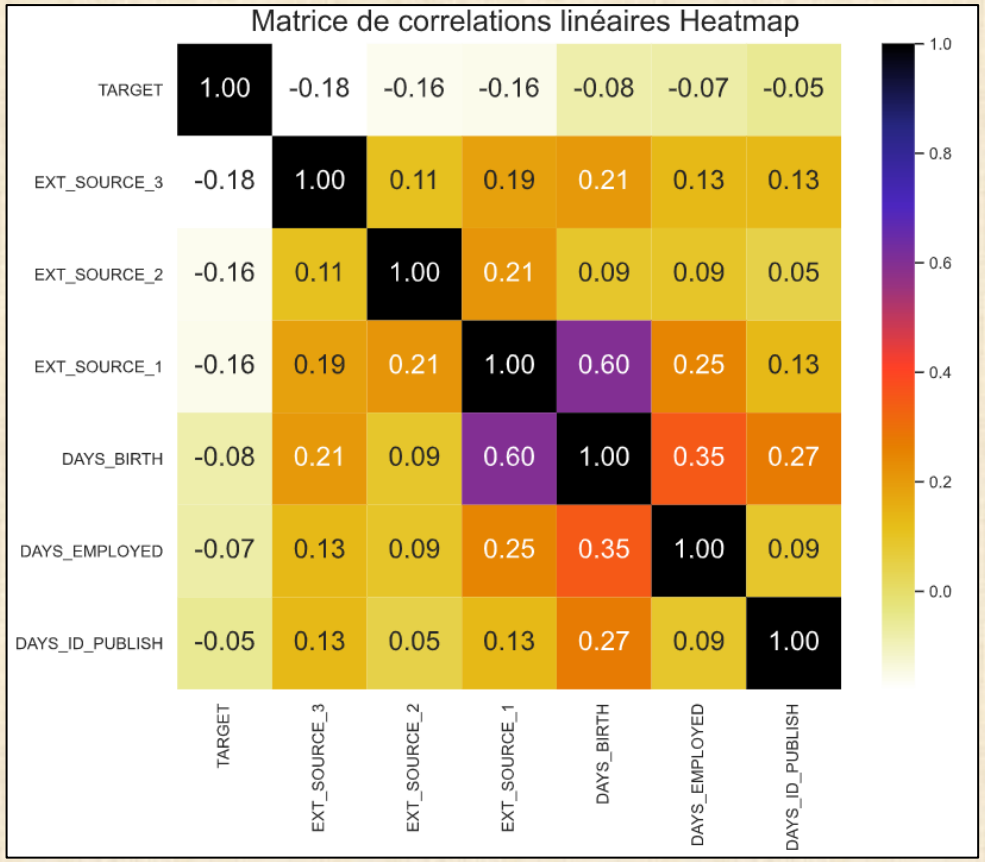


Concernant les valeurs aberrantes, elles seront remplacer par des valeurs NaN au lieu de les supprimer, ici l'exemple de la variable DAY_EMPLOYED

Analyse exploratoire

Analyse de corrélation: Corrélation avec la 'Target'

Les trois caractéristiques 'EXT_SOURCE' ont des corrélations négatives avec la 'Target', ce qui indique que plus la valeur de 'EXT_SOURCE' augmente le client est plus susceptible de rembourser le prêt. Nous pouvons également voir que 'DAYS_BIRTH' est positivement corrélé avec 'EXT_SOURCE_1', ce qui indique que l'un des facteurs de ce score est peut-être l'âge du client.




```
(EXT_SOURCE_3                  -0.178919
 EXT_SOURCE_2                  -0.160472
 EXT_SOURCE_1                  -0.155317
 DAYS_BIRTH                    -0.078239
 DAYS_EMPLOYED                 -0.074958
 DAYS_ID_PUBLISH               -0.051457
 FLOORSMAX_AVG                 -0.044003
 FLOORSMAX_MEDI                -0.043768
 FLOORSMAX_MODE                -0.043226
 DAYS_REGISTRATION             -0.041975
 AMT_GOODS_PRICE               -0.039645
 REGION_POPULATION_RELATIVE    -0.037227
 ELEVATORS_AVG                 -0.034199
 ELEVATORS_MEDI                -0.033863
 FLOORSMIN_AVG                 -0.033614
 FLOORSMIN_MEDI                -0.033394
 LIVINGAREA_AVG                -0.032997
 LIVINGAREA_MEDI               -0.032739
 FLOORSMIN_MODE                -0.032698
 TOTALAREA_MODE                -0.032596
 Name: TARGET, dtype: float64,
 REG_REGION_NOT_LIVE_REGION     0.005576
 REG_REGION_NOT_WORK_REGION     0.006942
 OBS_60_CNT_SOCIAL_CIRCLE       0.009022
 OBS_30_CNT_SOCIAL_CIRCLE       0.009131
 CNT_FAM_MEMBERS                0.009308
 CNT_CHILDREN                   0.019187
 AMT_REQ_CREDIT_BUREAU_YEAR     0.019930
 FLAG_WORK_PHONE                0.028524
 DEF_60_CNT_SOCIAL_CIRCLE       0.031276
 DEF_30_CNT_SOCIAL_CIRCLE       0.032248
 LIVE_CITY_NOT_WORK_CITY        0.032518
 OWN_CAR_AGE                    0.037612
 FLAG_DOCUMENT_3                0.044346
 REG_CITY_NOT_LIVE_CITY         0.044395
 FLAG_EMP_PHONE                 0.045982
 REG_CITY_NOT_WORK_CITY         0.050994
 DAYS_LAST_PHONE_CHANGE         0.055218
 REGION_RATING_CLIENT           0.058899
 REGION_RATING_CLIENT_W_CITY    0.060893
 TARGET                         1.000000
 Name: TARGET, dtype: float64)
```

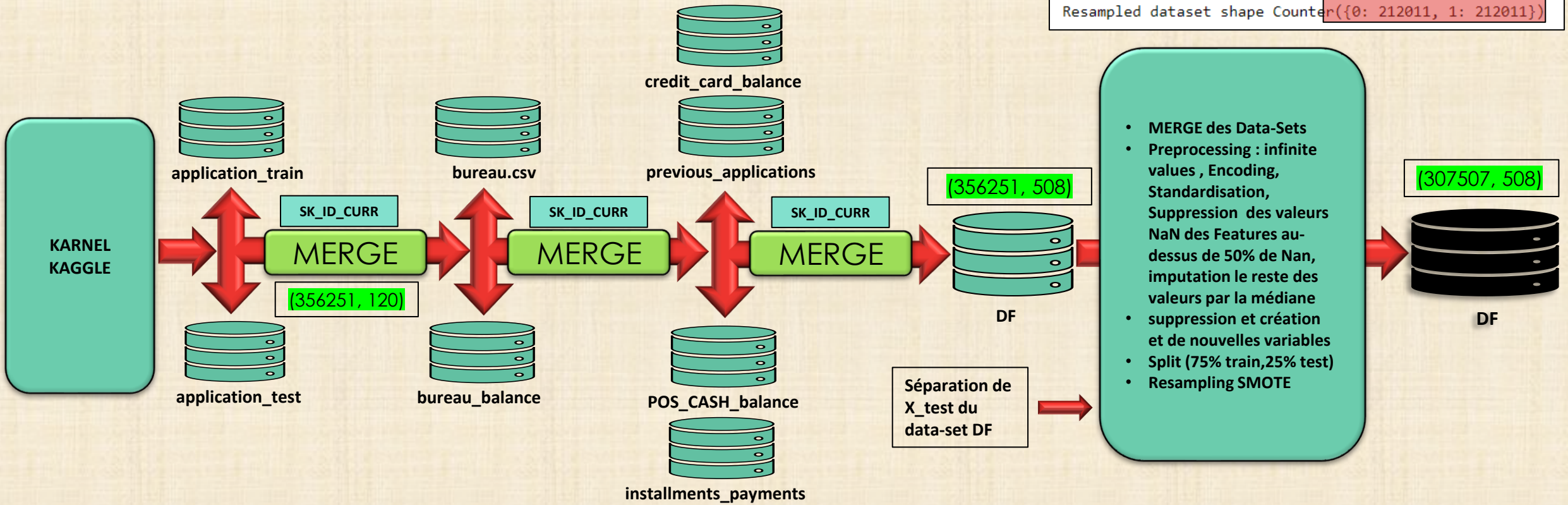
La première constatation est que la corrélation n'est pas très forte (en fait, elles sont toutes considérées comme très faibles), mais ces variables seront toujours utiles pour un modèle d'apprentissage automatique permettant de prédire si un demandeur remboursera ou non un prêt à temps.

Traitement et nettoyage des Data-Sets « Feature engineer »

Les différentes étapes de préparation de données:

```
# define smote strategy
sm = SMOTE(random_state=42)
X_train, y_train = resampling (X_train, y_train, sm)
```

```
Original dataset shape Counter({0: 212011, 1: 18619})
Resampled dataset shape Counter({0: 212011, 1: 212011})
```

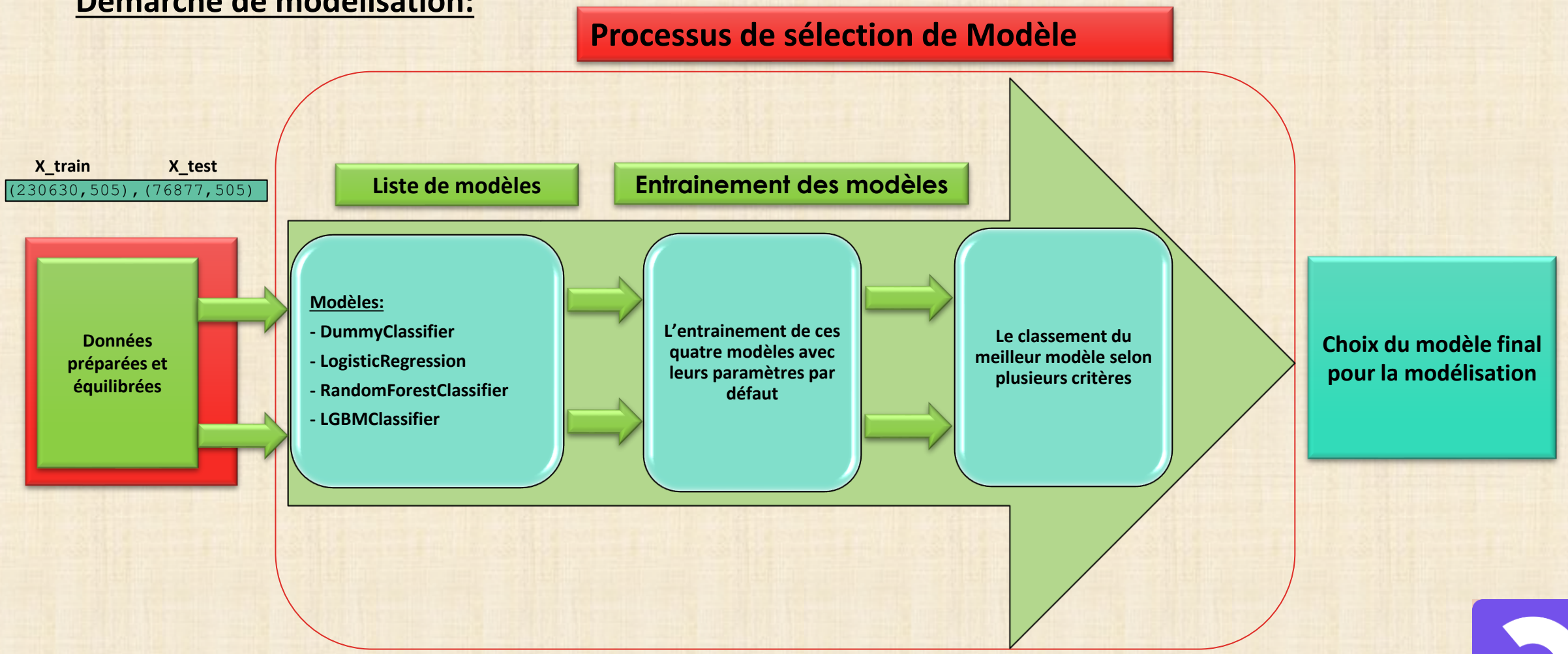


X_train X_test

```
((230630, 505), (76877, 505))
```

Modélisation

Démarche de modélisation:



Modélisation

Résultats de modélisation:

```
==================================================
Model : LGBMClassifier()
==================================================
AUC : 0.7754
============
              precision    recall  f1-score   support

           0       0.92      1.00      0.96     70671
           1       0.61      0.03      0.06      6206
```

```
==================================================
Model : RandomForestClassifier()
==================================================
AUC : 0.7175
============
              precision    recall  f1-score   support

           0       0.92      1.00      0.96     70671
           1       1.00      0.00      0.00      6206
```

```
==================================================
Model : LogisticRegression(max_iter=1000)
==================================================
AUC : 0.5884
============
              precision    recall  f1-score   support

           0       0.92      1.00      0.96     70671
           1       0.08      0.00      0.00      6206
```

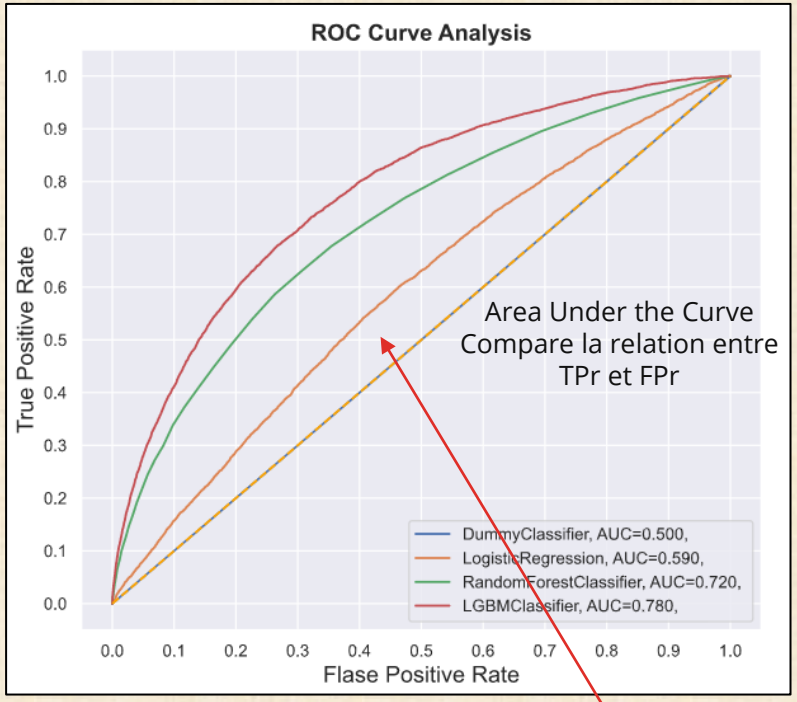
```
==================================================
Model : DummyClassifier()
==================================================
AUC : 0.5000
============
              precision    recall  f1-score   support

           0       0.92      1.00      0.96     70671
           1       1.00      0.00      0.00      6206
```

$$\text{Precision} = \frac{tp}{tp + fp}$$

$$\text{Recall} = \frac{tp}{tp + fn}$$

$$\text{Fscore} = \frac{precision.recall}{Beta^2.precision + recall}$$

$$= (1 + Beta^2).\frac{tp}{(1 + Beta^2).tp + Beta^2.fn + fp}$$



	models	fpr	tpr	fprOpt	tprOpt	Accuracy	Precision	F_beta_Score	Recall	Time	AUC
3	LGBMClassifier	[0.0, 0.0, 0.0, 1.4150075702905011e-05, 1.4150...	[0.0, 0.00016113438607798906, 0.00048340315823...	0.2695	0.6819	0.92	0.60	0.04	0.03	20.24	0.78
2	RandomForestClassifier	[0.0, 0.0, 0.0, 2.8300151405810022e-05, 4.2450...	[0.0, 0.00016113438607798906, 0.00048340315823...	0.3533	0.6761	0.92	0.60	0.00	0.00	294.75	0.72
1	LogisticRegression	[0.0, 1.4150075702905011e-05, 0.00014150075702...	[0.0, 0.0, 0.0, 0.00016113438607798906, 0.0001...	0.47	0.5838	0.92	0.13	0.00	0.00	13.39	0.57
0	DummyClassifier	[0.0, 1.0]	[0.0, 1.0]	0.0	0.0	0.92	0.00	0.00	0.00	0.28	0.50

Modèles retenu

On donne plus d'importance au recall donc aux FN

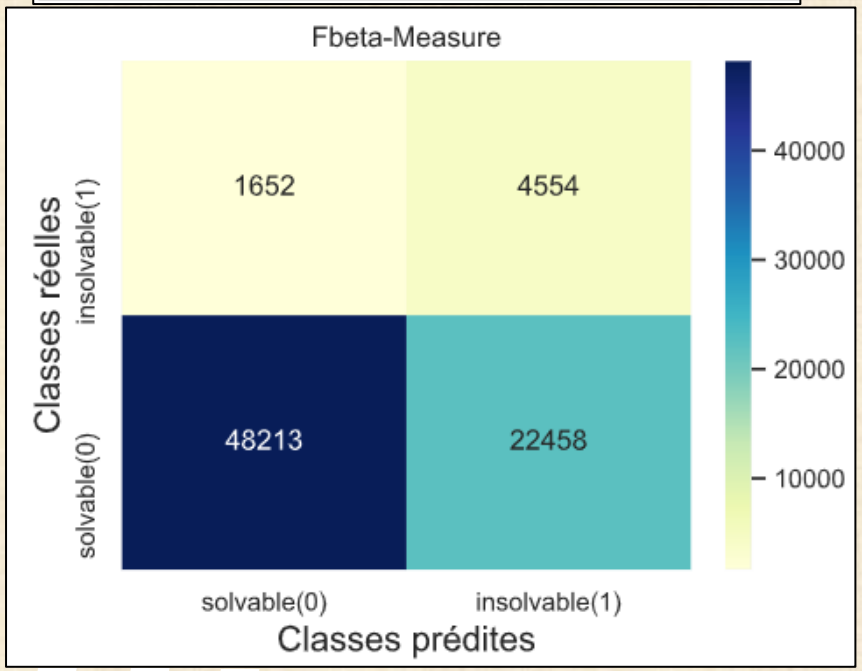
Entrainement LGBMClassifier:

- Entrainement du modèle avec **Grid Search CV**

[**LGBMClassifier**(learning_rate=0.11, max_depth=10,
min_child_weight=40, n_estimators=300, num_leaves=35, reg_alpha=1, reg_lambda=1.4,
scale_pos_weight=20, subsample=0.7),
{'n_estimators': 300, 'reg_alpha': 1, 'reg_lambda': 1.4, 'subsample': 0.7}]

```
AUC : 0.7749
============
              precision    recall  f1-score   support

           0       0.97      0.65      0.78     70671
           1       0.16      0.76      0.26      6206

    accuracy                           0.66     76877
   macro avg       0.56      0.71      0.52     76877
weighted avg       0.90      0.66      0.74     76877
```

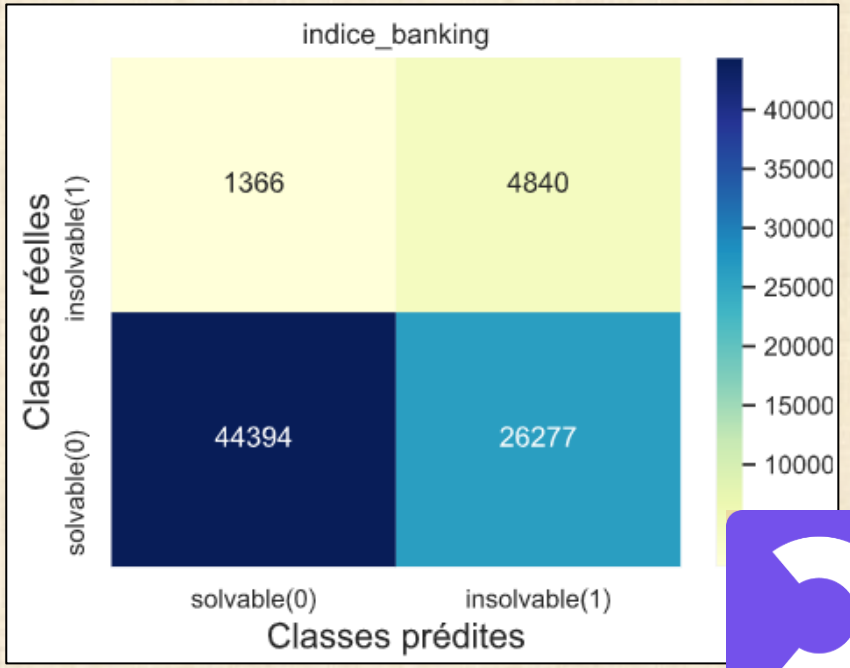
```
y_proba_0 = Scoring_credit.predict_proba(X_test)[:, 1]
y_pred_0 = (y_proba_0 > 0.53)
y_pred_0 = np.array(y_pred_0 > 0) * 1
```



La différence entre la métrique spécifique métier et la métrique technique Fbeta-measure c'est qu'on a moins de **FN**, 1366 contre 1652, au niveau des **TN** et **FP** la métrique technique est un peu mieux, sinon sur les **TP** la métrique spécifique est un peu mieux

Sur la métrique spécifique des clients ne se sont pas vus accordés de prêts alors qu'ils étaient solvables **FP**, mais à l'inverse moins de clients ont été prédits solvables alors qu'ils ne l'étaient pas **FN**. Ce que l'on a essayé par la construction d'une métrique métier est donc vérifié.

```
y_proba_1 = Scoring_credit.predict_proba(X_test)[:, 1]
y_pred_1 = (y_proba_1 > 0.4828)
y_pred_1 = np.array(y_pred_1 > 0) * 1
```



Modèles retenu

Métrique et fonction coût métier « le seuil de solvabilité optimisé »:

Au globale trois **métriques** a été testées pour définir le seuil de solvabilité en plus de cela la fonction coût métier

- **G-MEAN: 0.53**

Sensitivity = True Positive Rate
Specificity = 1 – False Positive Rate

} **G-Mean = sqrt (Sensitivity * Specificity)**

```
============================================================
thresholdOpt=0.53 gmean= 0.71 fprOpt = 0.31 tprOpt = 0.73
============================================================
```

- **Youden's J statistic : 0.5343**

Youden's J statistic : youdenJ = TPr – FPr

```
============================================================
Best Threshold: 0.5343 with Youden J statistic: 0.7089
FPR: 0.3129, TPR: 0.7314
============================================================
```

!!!! F-measure: permet d'équilibrer la précision et le rappel

- **Fbeta-Measure: 0.5343**

F2-score = (5 * precision * recall) / (4 *precision + recall)

```
============================================
Best Threshold = 0.534333, F-Score = 0.441
============================================
```

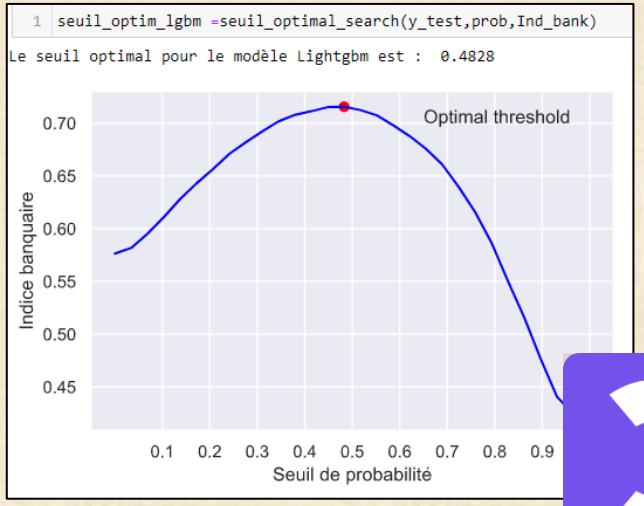
F2-Measure (beta=2.0) : Moins de poids de précision, plus de poids de rappel (plus d'importance au rappel autrement dit aux faux négatifs)

- **Métrique bancaire seuil personnalisé: 0.4828**

J = TP*TP_value + TN*TN_value + FP*FP_Value + FN*FN_value

Les coefficients qui ont été pris:

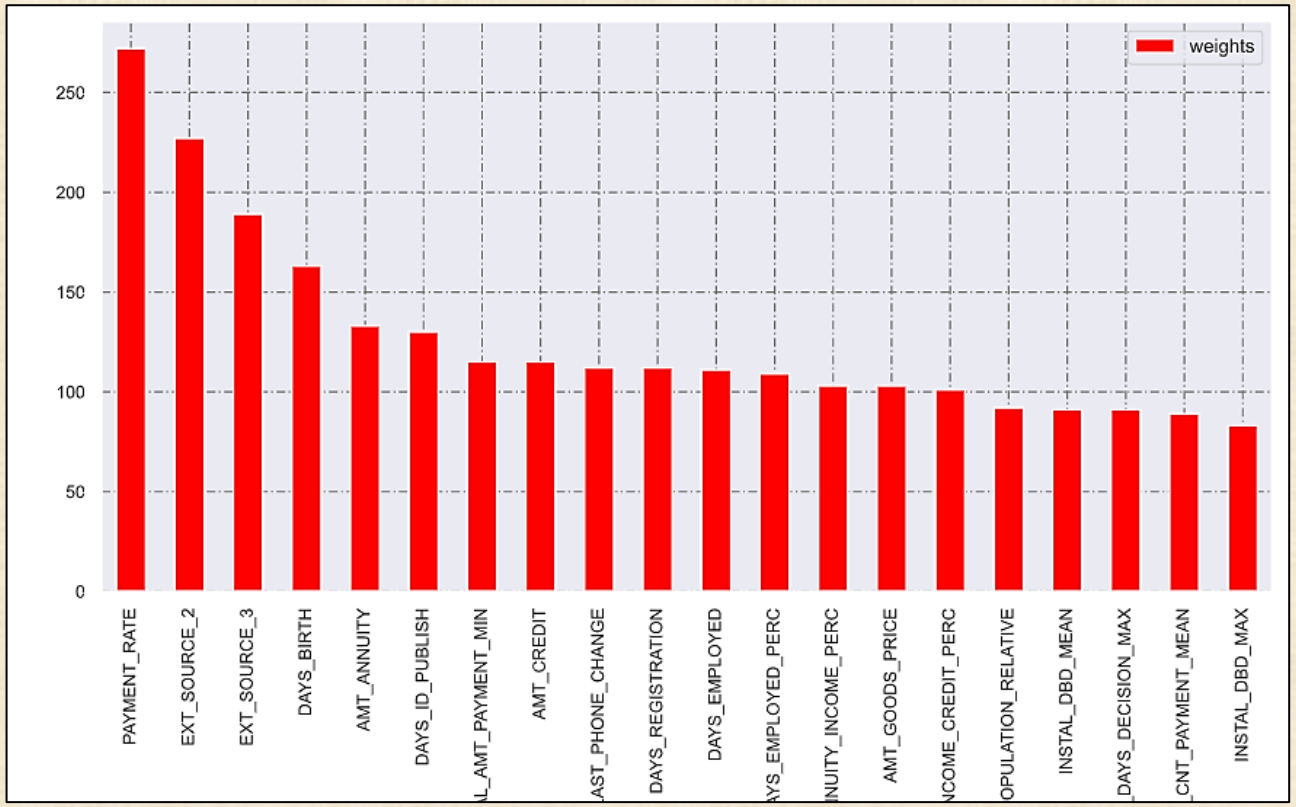
- **TP_value : 1**
- **FN_value : -30**
- **TN_value : 1**
- **FP_value : -1**



Modèles retenu

Features importance global:

les principales Features qui contribuent à l'élaboration du modèle



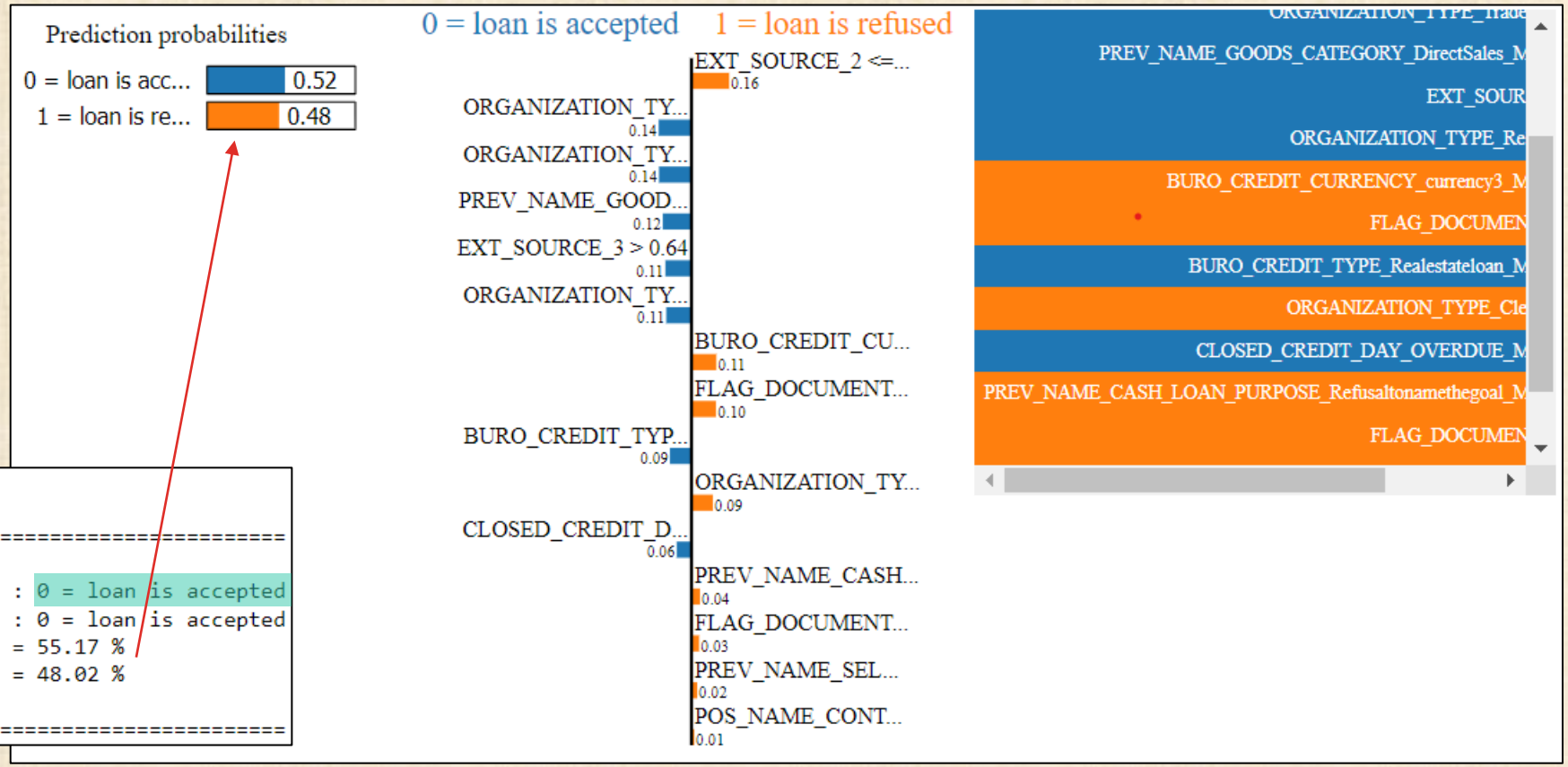
	features	weights
178	PAYMENT_RATE	272
26	EXT_SOURCE_2	227
27	EXT_SOURCE_3	189
6	DAYS_BIRTH	163
3	AMT_ANNUITY	133
9	DAYS_ID_PUBLISH	130
497	INSTAL_AMT_PAYMENT_MIN	115
2	AMT_CREDIT	115
42	DAYS_LAST_PHONE_CHANGE	112
8	DAYS_REGISTRATION	112
7	DAYS_EMPLOYED	111
174	DAYS_EMPLOYED_PERC	109
177	ANNUITY_INCOME_PERC	103
4	AMT_GOODS_PRICE	103
175	INCOME_CREDIT_PERC	101
5	REGION_POPULATION_RELATIVE	92
484	INSTAL_DBD_MEAN	91
457	APPROVED_DAYS_DECISION_MAX	91
283	PREV_CNT_PAYMENT_MEAN	89
483	INSTAL_DBD_MAX	83

Modèles retenu

Local Explanations LIME: Importance des Features pour un client spécifique

Exemple d'un client « customer ID = 123326 »

Local Interpretable Model-agnostic Explanations: est un modèle local qui cherche à expliquer la prédiction d'un individu par analyse de son voisinage.



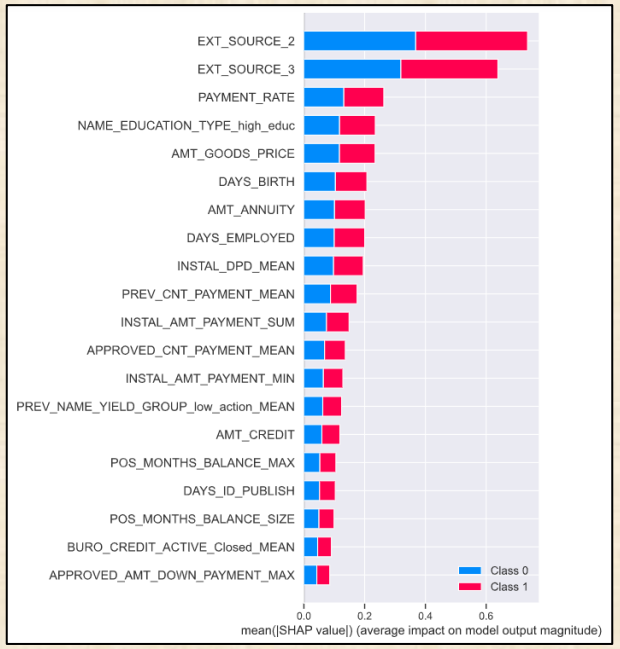

```
==========================
customer ID =  123326
==================================================

Predicted classe.............. : 0 = loan is accepted
Real classe................... : 0 = loan is accepted
Threshold model............... = 55.17 %
Loan probability ............. = 48.02 %

==================================================
```

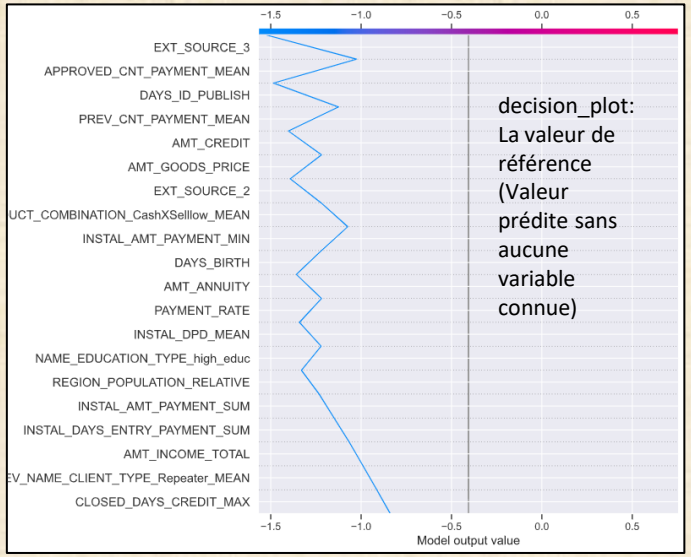
Modèles retenu

SHAP analysis :

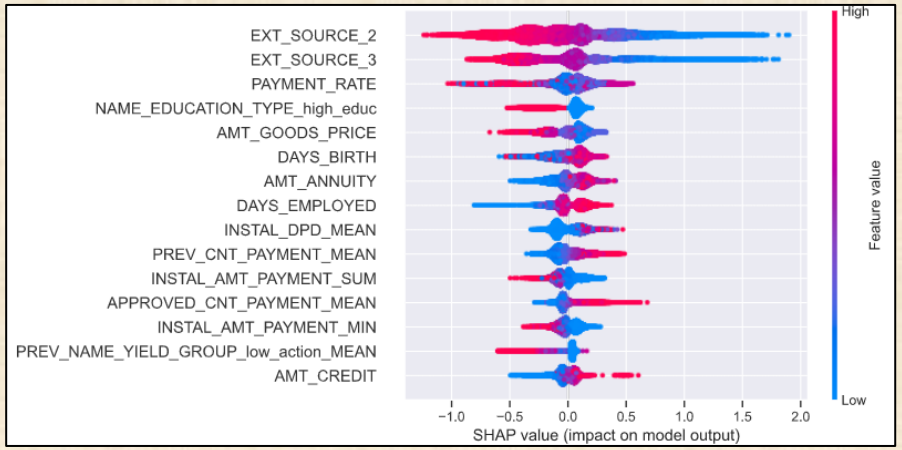
Le **SHAP** signifie SHapley Additive exPlanations et utilise l'approche de la théorie des jeux pour expliquer les prédictions du modèle.

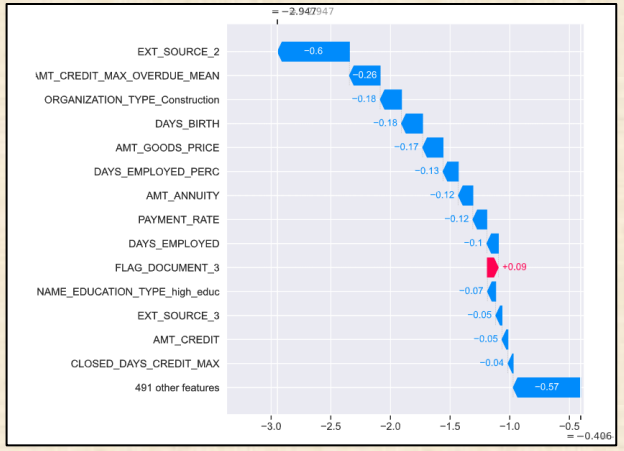
Les valeurs de Shapley calculent l'importance d'une variable en comparant ce qu'un modèle prédit avec et sans cette variable. Cependant, étant donné que l'ordre dans lequel un modèle voit les variables peut affecter ses prédictions, cela se fait dans tous les ordres possibles, afin que les fonctionnalités soient comparées équitablement. Cette approche est inspirée de la théorie des jeux.





decision_plot: La valeur de référence (Valeur prédite sans aucune variable connue)

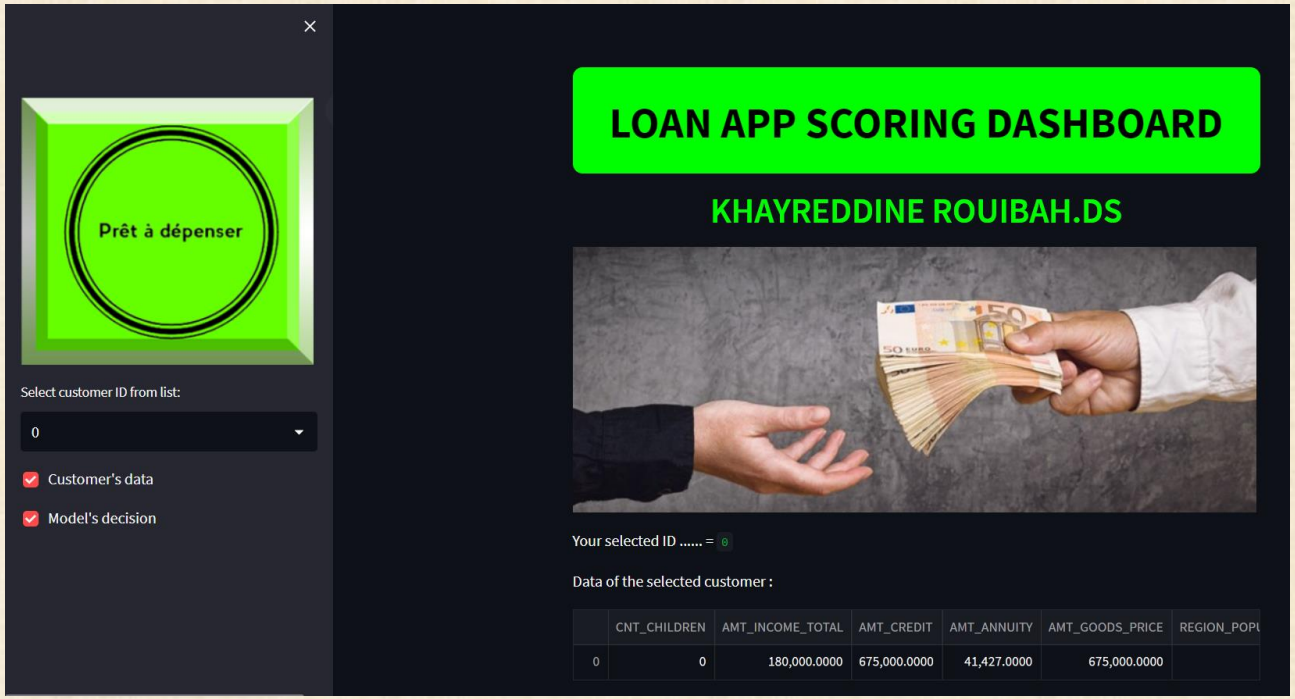




Présentation du Dashboard

Dashboard « App » + API FLASK:

L'application a été développée en backend et en frontend, Le backend **API FLASK** à été déployé sur la plateforme **Heroku** et le dashboard est développé avec **Streamlit** (Framework open-source Python), hébergée sous **GitHub** et déployée sur https://share.streamlit.io/



Cette App permet de:

- Visualiser le score et l'interprétation de ce score pour chaque client de façon intelligible
- Visualiser des informations descriptives relatives à un client (via un système de filtre).
- Comparer les informations descriptives relatives à un client à l'ensemble des clients ou à un groupe de clients similaire

Conclusion

Model

- Travailler plus sur la partie Feature Engineering
- Choix de technique pour équilibrer les classes
- Des métriques d'évaluations basées sur des hypotheses métier confirmées

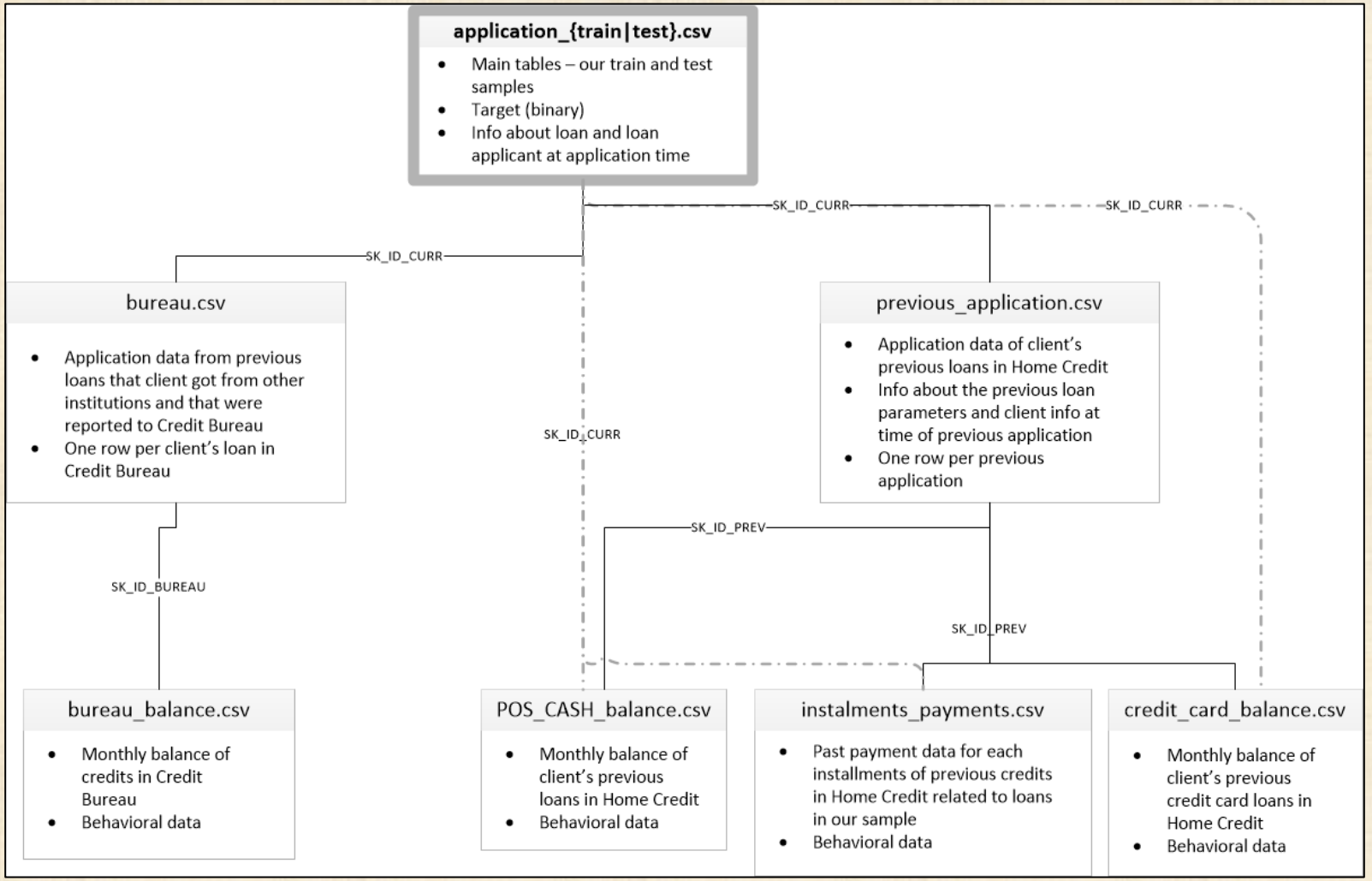
Dashboard app + API Flask:

- Création et éployement sur share.streamlit
- Déployement sur le cloud Heroku

URL de l'App:
https://bit.ly/dashboard3hkDxiN

Dépôt Github:
https://github.com/KH-spec/loan-dash-strmlt ➔ Dashboard
https://github.com/KH-spec/app_api_flask ➔ API FLASK

Annexe


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
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