**Parcours Data Scientist**

Projet 07 - Implémentez un modèle de scoring

Etudiant : Romain Dufief

**Note méthodologique**

1. **Méthodologie d’entraînement du modèle**

* Datasets utilisés : application\_train.csv
* Nombre de variables numériques : 51
* Nombre de variables catégorielles : 9
* Nombre de lignes : 307 511

Paramétrage des variables :

Au préalable, avant entraînement du modèle :

* Encodage des variables catégorielles : **LabelEncoder** (si le nombre de catégories est <=2), **OneHotEncoder** pour les autres. *-- (sklearn.preprocessing)*

Puis dans le cadre d’un modèle en pipeline :

* Imputation par la médiane : SimpleImputer(strategy=’median’) -- *(sklearn.impute)*
* Standardisation des variables numériques : StandardScaler*--**(sklearn.preprocessing)*
* Stabilité des paramètres : GridSearchCV *-- (sklearn.model\_selection)*

Entraînement du modèle :

Modèles testés :

* RandomForestClassifier *-- (sklearn.ensemble)*
* LogisticRegression *-- (sklearn.linear\_model)*
* LinearSVC *-- (sklearn.svm)*
* LGBMClassifier *-- (lightgbm.lgb)*

Modèle choisi :

LGBMClassifier(n\_estimators=100,

objective = 'binary',

class\_weight = 'balanced',

learning\_rate = 0.05,

reg\_alpha = 0.1,

reg\_lambda = 0.01,

subsample = 0.8,

n\_jobs = -1,

random\_state = 50)

1. **Détails de la modélisation**
   1. Fonction Coût

‘binary’ : classification à 2 classes, paramètre par défaut (minimisation de l’entropie croisée par échantillonnage).

* 1. Algorithme d’optimisation

Des paramètres proches du défaut ont été utilisés ici. Nous avons fait varier les paramètres de régularisation, sans effet manifeste, nous avons donc fini par conserver ceux initialement proposés.

Nous avons considéré que l’objectif principal de ce projet n’était pas de proposer un modèle de classification hyper-performant, mais plutôt de creuser l’aspect ‘mise en production’ et visualisation du modèle. En conséquence nous avons fait le choix d’utiliser un modèle peu optimisé pour avancer.

* 1. Métrique d’évaluation

Plusieurs métriques ont été utilisées : ROC-AUC, Recall, Accuracy, Precision.

Si la GridSearchCV utilisait ROC-AUC comme méthode de scoring, toutes ces métriques ont été utilisées pour l’observation des performances de nos différents tests de modèle.

Par ailleurs une métrique ‘custom’ a été créée pour ce modèle. En effet, l’objectif métier étant la minimisation du risque de défaut, nous avons considéré qu’il était préférable de minimiser le nombre de faux négatifs, quitte à avoir un modèle imprécis (cf notebook). Nous avons donc attribué 10 points à chaque faux négatif, 1 point à chaque faux positif, et 0 aux classifications correctes.

def custom\_asymmetric\_eval(y\_true, y\_pred):

residual = (y\_true - y\_pred).astype("float")

loss = np.where(residual == 1, 10,(np.where(residual == -1, 1, 0)))

return np.mean(loss)

En conséquence notre métrique personnalisée nous a conduit à abaisser le seuil de classification à 0.45, après observation de la variation des différents éléments en fonction du seuil.

1. Interprétabilité du modèle

|  |  |
| --- | --- |
| Feature du modèle | **Importance** |
| EXT\_SOURCE\_3 | 2405 |
| AMT\_CREDIT | 2376 |
| EXT\_SOURCE\_2 | 2279 |
| DAYS\_BIRTH | 2275 |
| DAYS\_ID\_PUBLISH | 1965 |
| DAYS\_REGISTRATION | 1955 |
| DAYS\_LAST\_PHONE\_CHANGE | 1953 |
| AMT\_INCOME\_TOTAL | 1263 |
| LANDAREA\_AVG | 1048 |

Le modèle repose donc fortement sur le scoring déjà utilisé par l’organisme de crédit, et retenu dans les variables EXT\_SOURCE. Par ailleurs on remarque que l’âge du client et le montant du crédit sont très importants dans le fonctionnement du modèle.

1. Limites & améliorations possibles

Les limites à ce modèle est le choix nécessaire d’un risque au détriment d’un autre. Le modèle n’a, en conséquence, qu’une précision plutôt faible.

Plusieurs pistes d’améliorations ont été identifiées :

* Feature Engineering complémentaire, avec plusieurs pistes :
  + utilisation des datasets de complément proposés dans le jeu de données initial
  + création de variables supplémentaires
  + optimisation des existantes (suppression, réduction de dimension, augmentation à l’aide des méthodes poiynomiales par exemple)
* Ajout de compléments de comparaison sur le dashboard
* Construction d’une API qui héberge le modèle et score les clients en temps réel, sur la base d’informations fournies par le chargé de clientèle.