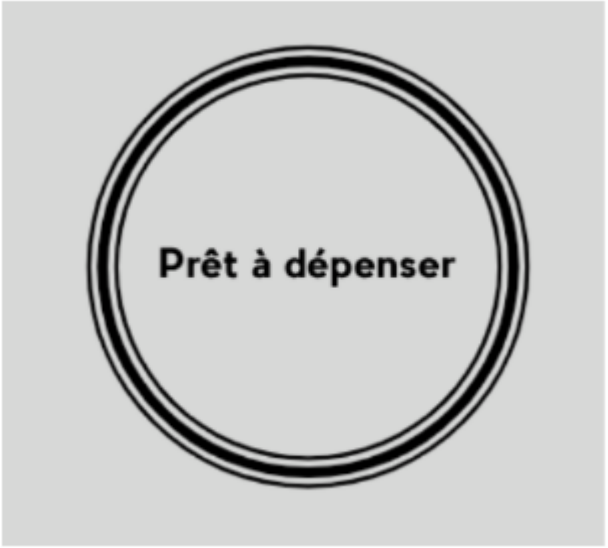
Note méthodologique

Projet 7 : Implémentez un modèle de scoring



Yihan ZHANG

Mai 2024

https://github.com/yihanzh/projectscoring/tree/main

Contexte

Nous sommes des Data Scientists au sein de la société financière "Prêt à dépenser", qui propose des crédits à la consommation pour des personnes ayant peu ou pas 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 classifier 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 des institutions financières, etc.).

De plus, les chargés de relation client ont remonté le fait que les clients sont de plus en plus demandeurs de transparence vis-à-vis des décisions d'octroi de crédit. Cette demande de transparence des clients va tout à fait dans le sens des valeurs que l'entreprise veut incarner. Prêt à dépenser décide donc de développer un tableau de bord interactif pour que les chargés de relation client puissent à la fois expliquer de façon la plus transparente possible les décisions d'octroi de crédit, mais également permettre à leurs clients de disposer de leurs informations personnelles et de les explorer facilement.

Mission

- Construire un modèle de scoring qui donnera une prédiction sur la probabilité de faillite d'un client de façon automatique.
- Construire un tableau de bord 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.
- Mettre en production le modèle de scoring de prédiction à l'aide d'une API, ainsi que le tableau de bord interactif qui appelle l'API pour les prédictions.

Cette note méthodologique décrit la méthodologie d'entraînement du modèle, le traitement du déséquilibre des classes, la fonction coût métier, l'algorithme d'optimisation et la métrique d'évaluation, un tableau de synthèse des résultats, l'interprétabilité globale et locale du modèle, les limites et les améliorations possibles, ainsi que l'analyse du Data Drift.

1. Préparation des Données

1.1 Feature Engineering:

- Création de Nouvelles Caractéristiques: Application de fonctions statistiques (min, max, mean, sum, variance) sur les tables groupées pour générer de nouvelles caractéristiques pertinentes.
- Encodage des Variables Catégorielles: Utilisation de l'encodage one-hot pour les variables catégorielles afin de les transformer en caractéristiques numériques.

1.2 Séparation des Données:

- Division en données d'Entraînement et de Test: Séparation des données en données d'entraînement (80%) et de test (20%) en utilisant `train_test_split` de `sklearn`.
- Validation croisée: Utilisation de la validation croisée stratifiée (`StratifiedKFold`) pour s'assurer que chaque fold a une proportion similaire de classes.

1.3 Imputation:

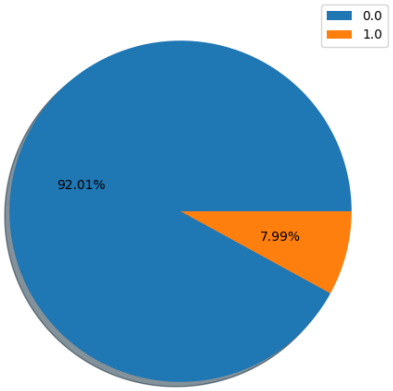
- Imputation des Valeurs Manquantes: Utilisation de `KNNImputer` et `SimpleImputer` pour combler les valeurs manquantes selon la nature des données.

2. Traitement du Déséquilibre des Classes

Le déséquilibre des classes est un problème courant dans les jeux de données de classification où une classe est beaucoup plus représentée que l'autre. Cela peut entraîner un biais du modèle vers la classe majoritaire, rendant la détection de la classe minoritaire moins efficace. Pour traiter ce problème, plusieurs techniques peuvent être utilisées.

2.1 Inspection des Données:

- Distribution des Classes: Analyse de la répartition des classes pour comprendre l'étendue du déséquilibre. Selon le fichier de description des colonnes, la cible (TARGET) du modèle est définie comme suit :
  - 1 : “client with payment difficulties: he/she had late payment more than X days on at least one of the first Y installments of the loan in our sample”.
  - 0 : all other cases
- Visualisation:



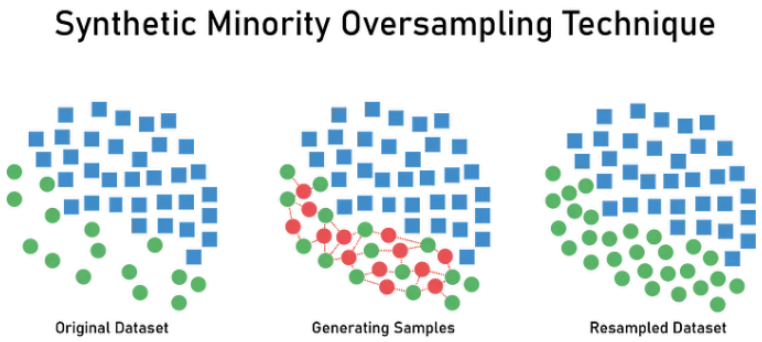
- La distribution de la TARGET dans les données d'entraînement présente un fort déséquilibre de classes : environ 92% pour la classe 0 et environ 8% pour la classe 1.

2.2 SMOTE

SMOTE (Synthetic Minority Oversampling Technique): Génération d'échantillons synthétiques pour la classe minoritaire en combinant les caractéristiques de ses voisins les plus proches.

Avantages: Augmente le nombre d'exemples de la classe minoritaire sans duplication, aidant à créer un ensemble de données plus équilibré.

Inconvénients: Peut introduire du bruit si les nouveaux échantillons synthétiques ne sont pas représentatifs.



3. La fonction coût métier, l'algorithme d'optimisation et la métrique d'évaluation

3.1 La fonction coût métier

Afin d'entraîner les modèles et choisir le meilleur modèle, nous définissons une fonction de coût métier à optimiser. L'optimisation consiste à trouver les meilleurs hyperparamètres de nos modèles afin de minimiser le coût métier.

		Vraie valeur	
		- TARGET=0	+ TARGET=1
Valeur prédite	- TARGET=0	Client prédit sans difficulté de remboursement, et réellement sans difficulté de remboursement (TN)	⚠⚠⚠ Octroi de crédit à un client qui aura des difficultés de remboursement (FN) ⇒ Pertes en capital, frais.
	+ TARGET=1	Refus de crédit à un client qui n'aurait pas eu de difficulté de remboursement (FP) ⇒ Manque à gagner.	Client prédit avec difficultés de remboursement, et réellement avec difficultés de remboursement (TP)

Objectifs :

- Maximiser la capacité à détecter tous les clients avec difficultés de paiement, cad maximiser le rappel : TP / (TP+FN).
- Maximiser la capacité à ne pas détecter comme "mauvais" un "bon" client, cad maximiser la précision : TP / (TP+FP).

Définition d'une fonction de coût à optimiser:

- Hypothèse : le coût d'un FN est 10 fois supérieur au coût d'un FP
- score = 1*fp + 10*fn

3.2 Les métriques d'évaluation

Précision (Accuracy)

La précision est le ratio du nombre de prédictions correctes (toutes classes confondues) sur le nombre total de prédictions. C'est une mesure simple de l'efficacité globale du modèle.

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$

AUC - Area Under the ROC Curve

L'AUC mesure la capacité d'un modèle à distinguer entre les classes positives et négatives. Elle est calculée comme l'aire sous la courbe ROC (Receiver Operating Characteristic), qui trace le taux de vrais positifs (Recall) contre le taux de faux positifs (1 - Specificity) pour différents seuils de classification.

Le temps d'exécution

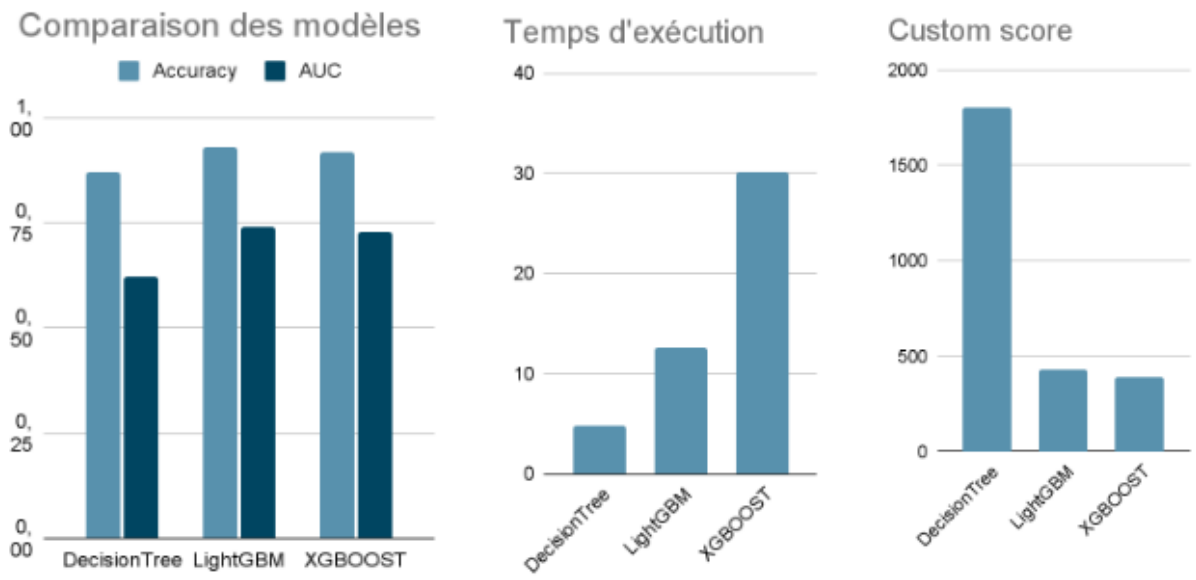
Le coût métier

4. Entraînement des Modèles

- Choix des Algorithmes: Entraînement de plusieurs modèles, y compris `LGBMClassifier`, `RandomForestClassifier`, et `XGBClassifier`.
- Normalisation: Application de techniques de normalisation (StandardScaler) pour s'assurer que les caractéristiques sont sur une échelle similaire.
- Optimisation des Hyperparamètres: Utilisation de `GridSearchCV` pour trouver les meilleurs hyperparamètres en utilisant le score métier.
- Suivi des Expériences: Utilisation de `mlflow` pour suivre les différents essais, les paramètres et les résultats.
- Les hyperparamètres choisis à optimiser

Modèle	**Hyperparamètre**	**Valeurs Testées**
Decision Tree	`model__min_samples_split`	[10, 100]
LightGBM	`model__n_estimators`	[200, 500, 1000]
	`model__learning_rate`	[0.02, 0.1, 0.5]
	`model__max_depth`	[3, 5, 8]
XGBOOST	`model__learning_rate`	[0.5]
	`model__max_depth`	[3, 5, 8]
	`model__n_estimators`	[200, 500, 1000]

5. synthèse des résultats



Le modèle choisi est le LightGBM.

6. Interprétabilité globale et locale du modèle

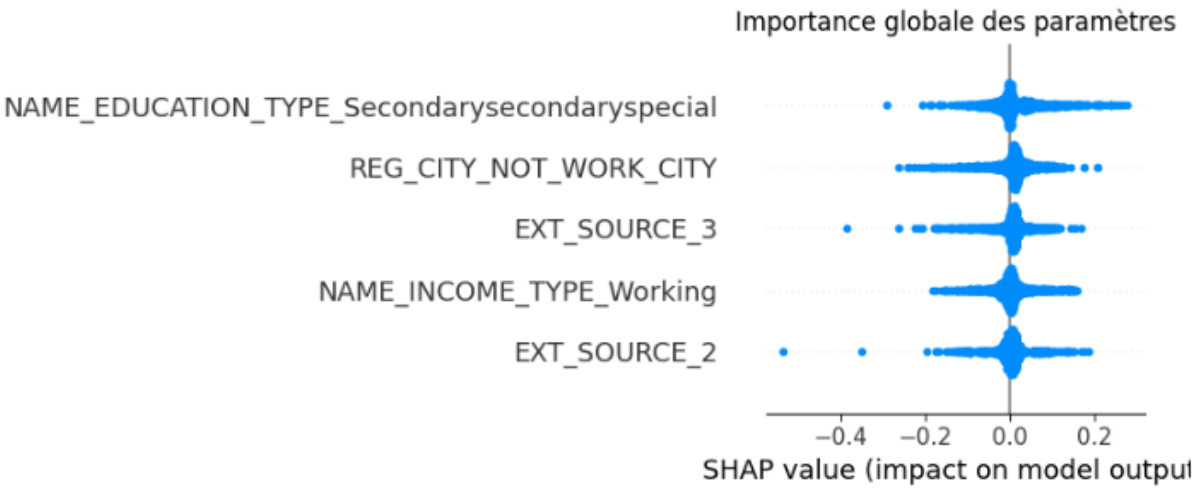
SHAP (SHapley Additive exPlanations) est une méthode d'interprétation des modèles de machine learning basée sur les valeurs de Shapley, un concept issu de la théorie des jeux coopératifs. Les valeurs de Shapley fournissent une manière équitable et cohérente de distribuer les gains (ou coûts) parmi les joueurs (ou features) en fonction de leur contribution à un résultat donné.

L'objectif de SHAP est d'expliquer la prédiction d'un modèle en décomposant la contribution de chaque feature de manière additive, de sorte que la somme des contributions de toutes les features égale la différence entre la prédiction du modèle et la prédiction moyenne.

6.1 Interprétabilité globale

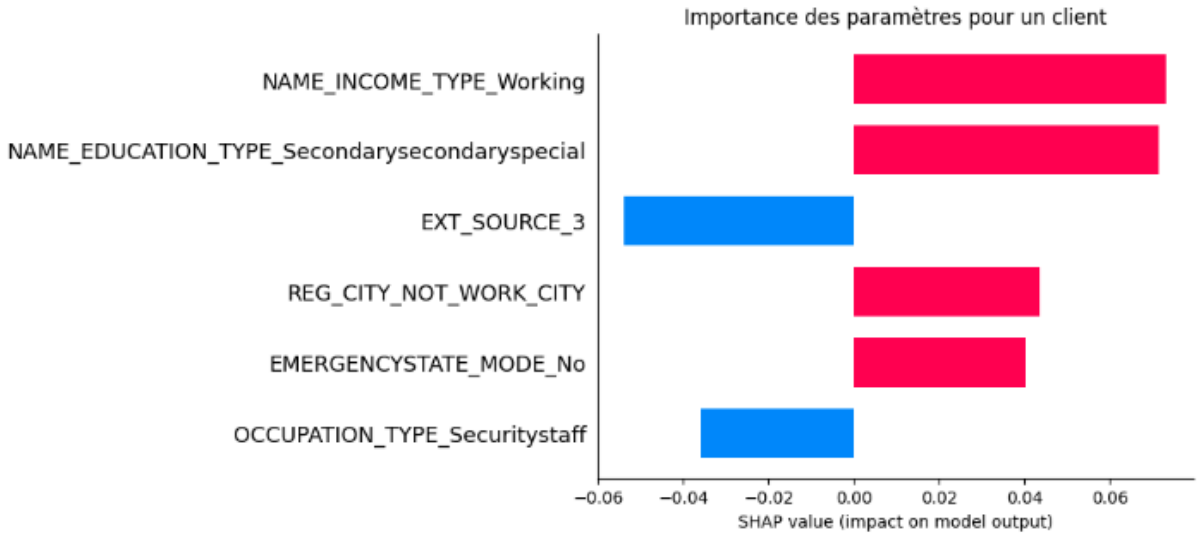
L'interprétabilité globale d'un modèle fait référence à la compréhension de la manière dont les features (caractéristiques) influencent les prédictions du modèle à un niveau agrégé. Contrairement à l'interprétabilité locale, qui se concentre sur des prédictions spécifiques, l'interprétabilité globale donne une vue d'ensemble des relations entre les features et les prédictions sur l'ensemble du jeu de données.

La méthode `shap.summary_plot` est une fonctionnalité de la bibliothèque SHAP (SHapley Additive exPlanations) qui permet de visualiser les contributions des features à la prédiction d'un modèle de manière globale. Ce plot combine plusieurs aspects d'interprétabilité pour offrir une vue d'ensemble des importances des features et de leur impact sur les prédictions.



6.2 Interprétabilité locale

L'interprétabilité locale d'un modèle fait référence à la compréhension des raisons spécifiques pour lesquelles le modèle a fait une prédiction particulière pour une instance donnée. Contrairement à l'interprétabilité globale, qui se concentre sur le comportement général du modèle sur l'ensemble des données, l'interprétabilité locale se concentre sur des prédictions individuelles.
Interpréter feature importance avec la méthode `shap.bar_plot`

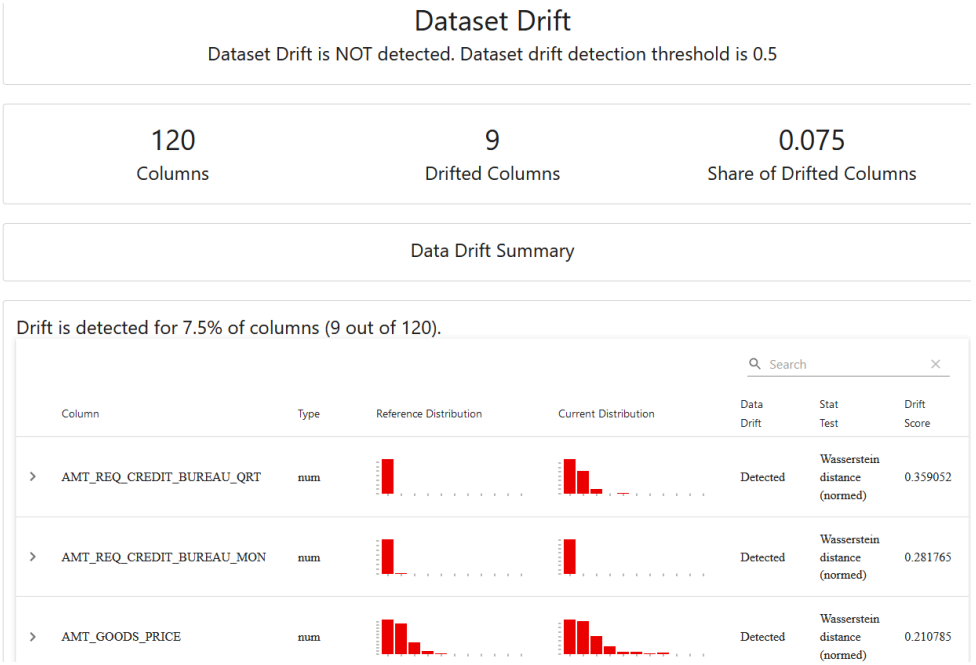


7. L’analyse du Data Drift

Evidently AI est un outil open-source pour la surveillance et l'analyse des performances des modèles de machine learning en production. Il se concentre sur la détection et l'analyse du drift des données (Data Drift), du drift du modèle, des biais, et d'autres problèmes potentiels qui peuvent survenir après le déploiement des modèles.
Le Data Drift se produit lorsque la distribution statistique des données d'entrée change au fil du temps, ce qui peut affecter la performance du modèle.

Mise en oeuvre du rapport Data Drift

- reference_data : Données de référence(modélisation) ⇒ “application_train.csv”
- current_data : Données de production (nouveaux clients) => “application_test.csv”



Interprétation du rapport

9 features sur 120 présentent du data drift, soit une proportion de data drift de 7.5%. Le seuil de détection par défaut est fixé à 50% (data drift détecté si au moins 50% des features dérivent). Le rapport nous indique donc que le data drift n’est pas détecté sur nos données.

8. Les limites et les améliorations possibles

La préparation des données et le feature engineering ont été réalisés en se basant sur un notebook Kaggle. On pourrait améliorer la modélisation en développant de nouvelles features pertinentes par rapport à la problématique métier, en collaboration avec les équipes métier, en s'appuyant sur leur expérience d’octroi de crédit à un client donné.

Le seuil de classification utilisé pour déterminer si une instance est classée comme positive ou négative a été laissé par défaut à 0.5. Il serait pertinent d'optimiser ce seuil pour maximiser la précision et le rappel.

Amélioration du modèle en approfondissant le “fine tuning” du modèle : les modèles mis en œuvre possèdent de nombreux hyperparamètres. Il est possible d’améliorer les performances du modèles en agrandissant l’espace de la recherche des meilleurs hyperparamètres avec Hyperopt.

De plus, nous pourrions utiliser l'importance des features pour optimiser notre modèle de prédiction : en sélectionnant les features en fonction de leur score d'importance, il serait possible de simplifier le modèle, d’améliorer sa rapidité et éventuellement ses performances.