AnyFin Hiring Task Report

Data Science Case - Probability of Default Prediction

Name: NGUYEN Thanh-Long

Email: nguyenthanhlong1990@gmail.com

Linkedin: https://www.linkedin.com/in/longtng/

Problem Statement

The purpose of this task is to build a predictive model that assigns default probabilities to loan applications to classify customers going to **default (target=1)** or **non-default (target=0)**. The noticeable problem is the data is highly imbalanced as defaulted customers contributed only 6% in total.

Data Processing and Model Building Stages:

- Data Exploration and General Pre-processing:
    - Remove Duplicates - 2,256 duplicated data points were dropped.
    - Missing values - `max_dpd_120d`, `min_dpd_120d`, `loans_created_120d`, `has_paid_120d`, `payments_120d`, `payment_vacation_120d`, `has_payment_plan` were dropped due to it percentages of missing values or undetermined distribution of values.
    - Check high correlation features and create new crossing-feature to remove collinearity:
        - `day` vs. `week_of_month`: drop day, consider week_of_month as the binning of day.
        - `credit_used` vs. `mortgage_house` vs. `mortgage_apartment`: as (`mortgage_house` + `mortgage_apartment`) < `credit_used`, drop all three, then create new `credit_used_left` = `credit_used` - (`mortgage_house` + `mortgage_apartment`).
        - `income_gross` vs. `salary_surplus`: create new feature `sal_sur_inc_gross` = `salary_surplus`/`income_gross`.
        - `creditors_count` vs. `credit_count`: create new feature `credit_creditors` = `credit_count`/`creditors_count`.

- Split Data and Create Dummy Variables:
    - Data set is split into 80% train/ 20% test while the distribution of `target` remains unchanged in both train/ test sets.
    - Fine-classing/ Coarse classing for creating Dummy Variables on both train and test set:
        - Turn the features into Dummy categorical variables - via fine-classing/ coarse classing methods (binning into different small categories).
        - Binning creates dummy variables and drops the one with the lowest WoE value to avoid the Dummy trap.

- The Logistic model was trained with following strategy:

    - Train a baseline model with class_weight = {0:6, 1:94}, which is revert of the dataset classes distribution. The class_weight hyperparameter is used to penalty on the Loss Function due to the imbalanced data
    - The Logistic model's hyper-parameters optimization via `GridSearchCV`:
        - Hyperparameters tuning on `C` and `class_weight`.
        - `RepeatedStratifiedKFold` Cross-Validation was used to avoid overfitting/ data leakage.
        - The training task will focus on maximizing the `roc_auc_score` while fine-tuning the hyperparameters.
        - Calculate the p_values of each coefficient output from the best performance model and only keep the statistically significant ones (p_values < 0.05). Note that, only the features, which got all dummies variables are statistically non-significant, will be removed.
        - The `address_count` feature has been removed as its all dummies variables are statistically non-significant. It is also correct when the Information Value of `address_count` is also lowest within all features at 0.05.
        - Refit the old model with final selected feature for the final model output.

- Model Evaluation:

    - Calculate the best threshold of Area Under the Receiver Operating Characteristic Curve (AUROC) from the final model.
    - Calculate the GINI coefficient.

Further Discussions/ Suggestions:

- The feature definitions are not provided in the instructions, if provided, the feature engineering would be more focused and result in a better performance model.
- The task was conducted in the classic PD model fashion which concentrated on explainability and only used Logistic Regression. Further methods could be considered such as tree-based, ensemble models such as voting, boosting, bagging or stacking - refer to my Kaggle notebook about this issue [Bank Marketing: Ensemble Learning Pipeline]
- Other model tuning techniques could be considered, such as Random search, Bayesian, etc.
- The roc_auc_score was focused as there is no information about the required evaluation metrics, such as minimizing the False Negative/ focus on Positive Class, etc. The notebook could extend in this fashion by using other metrics such as Precision-Recall AUC, Fbeta, etc.