

Stage D Report: Model Selection and Setup

This report presents the progress achieved during the POC (Proof of Concept) stage, starting with a dataset of 71 features and 326,403 instances (data preparation process is detailed in reports B and C). As elucidated in the preceding report (step C), our focus centers around the development of a classification model for predicting loan returns above 2% and a regression model for yield prediction. The report encompasses extensive findings and technical information, including the attached code file (stepD.py). We have successfully concluded training, testing, and initial performance evaluation, with a keen emphasis on vigilant overfitting monitoring. Looking ahead, we are dedicated to further refining and evaluating the models over the next four weeks, as we strive to identify and select the best-performing model.

Documenting Completed Work Assignments-

1. **Skewness Handling Approaches:** In our analysis, we encountered a skewed distribution issue in both the numerical yield and the classification of loans with over 2% yield. Appendix 1 shows that 78.8% of the data belongs to the over 2% yield group. The yearly yield variable demonstrated a significantly negative skewness (-40.14), indicating a highly skewed distribution with a long left tail and the Shapiro-Wilk test ($p < 0.05$) revealed a departure from normality (refer to appendix B). To mitigate bias and improve model performance, we applied specific techniques for classification and regression. For classification, we employed oversampling, class weighting, and threshold adjustment to balance class distribution. We also utilized algorithms like AdaBoost, and XGBoost, known for their robustness to unbalanced data. For regression, we employed transformations, stratified sampling, and model-specific techniques such as decision trees or random forests. These strategies effectively addressed the skewed distribution problem and enhanced model accuracy. Detailed information on the specific methods used for each model will be provided later.
2. **Training and Testing Ratio:** To ensure effective model training and evaluation, we divided the dataset into separate training and testing sets. We explored two split options: 80:20 and 70:30, with most data assigned for training. After evaluating the model's performance, we determined that the 70:30 split yielded better results (refer to appendix C). To maintain class distribution, we employed stratified sampling during the split. This approach safeguards against overfitting and ensures reliable model evaluation.
3. **Data Normalization:** During our process, we encountered challenges related to non-normal distributions and outliers in our features, which created complications when applying standard normalization techniques. To overcome these issues, we conducted comparisons and tests on the features using different types of normalization. We found that strong normalization, known for its effectiveness in handling non-normal and outlier distributions, yielded the best results. Further details about our findings will be discussed later.
4. **Model approach:** Our approach encompasses two key strategies to address the research question on the relevance of P2P lending as an asset. Firstly, we build a classifier to identify loans with a yield over 2%, which aligns with the current annual realized returns of GreatYields. This choice allows us to focus on potentially lucrative investments and select high-yield opportunities to potentially boost client returns. We also construct a regression model that predicts the expected realized return, offering insights into the profitability of P2P lending. By estimating potential returns, Walter can evaluate the attractiveness of P2P lending as an opportunity. By combining models, we gain insights into P2P lending's relevance and profitability, supporting informed investment decisions for Walter.

5. **Modeling process in general:** To optimize and monitor overfitting of the model, the following steps were undertaken:

- i. Cross-validation: Use k-fold cross-validation to assess model performance on different data subsets.
- ii. Monitor training set performance.
- iii. Hyperparameter tuning: Fine-tune hyperparameters to control model complexity and mitigate overfitting
- iv. Regularization: Apply L1 and L2 regularization to prevent overfitting and balance model complexity.
- v. Utilize algorithms for mitigating overfitting: Consider using algorithms like XGBoost and AdaBoost known for their ability to reduce overfitting and improve generalization.

6. **Classification Models:**

- a. **Evaluation metrics:** We considered multiple performance metrics to comprehensively assess our models. Considering the data imbalance, we acknowledged that precision and recall alone were inadequate measures, prompting us to consider the accuracy rate as an evaluation metric, despite the business significance of precision. The false positive rate (FPR) was of particular importance in terms of business impact, as investing in unprofitable loans can be highly detrimental. To address this, we utilized the ROC curve to maximize the area under the curve, considering both FPR and the true positive rate. Furthermore, we incorporated the false negative rate (FNR) to ensure valuable investment opportunities were not missed, using the F1 score as a combined metric for FPR and FNR evaluation.

We employed three types of classification models for our analysis:

- i. Logistic Regression: Logistic Regression: chosen for its simplicity, interpretability, and robustness in capturing the relationship between input variables and class probabilities. Forward selection identified the most relevant variables, while the class_weight argument handled the imbalanced distribution. We optimized the model by testing various hyperparameter combinations and normalization techniques, prioritizing high F1 scores. Reliable model assessment was performed using k-fold cross-validation (k=5). Appendix D provides details on the model and methodology, and Appendix E showcases the model results.
 - ii. XGBoost: selected for its excellent performance in handling complex datasets and its ability to fine-tune hyperparameters for optimal model performance. To tackle the imbalanced distribution, we used an oversampling technique and applied k-fold cross-validation (k=5). We tested different hyperparameter combinations and normalization techniques, emphasizing high F1 scores. More details on the model and methodology can be found in Appendix F, and Appendix G showcases the model results.
 - iii. AdaBoost: chosen for its sequential boosting approach, which improves weak classifiers and handles challenging instances, particularly in scenarios with class imbalance or overlapping decision boundaries. To address the imbalanced distribution, we used an oversampling technique and conducted k-fold cross-validation (k=5) for robust evaluation. We tested various hyperparameter combinations and normalization techniques, aiming for high F1 scores. For more information on the model and methodology, see Appendix H, and Appendix I presents the model results.
- b. **Models Evaluation**: In Appendix J, it is evident that logistic regression achieved the highest scores across all indices, with an AUC area of approximately 0.6 on the test set. Although this model is not perfect, it can still provide valuable predictions regarding the probability of a loan yielding above 2%. In the next step, we will further fine-tune the hyperparameters to improve the models and we will compare these results with the existing Grade model for a comprehensive analysis.

7. Regression Models:

- a. Evaluation metrics:** We use Mean Squared Error (MSE) and R-squared (R^2) as evaluation metrics for our regression model. MSE helps measure the accuracy of our model in predicting realized returns, allowing us to make informed investment decisions. R^2 quantifies the proportion of variability in returns explained by the model, providing insights into the model's ability to capture factors influencing P2P lending profitability. In the next step, we will incorporate additional evaluation metrics for a more comprehensive assessment.
- b. We employed two types of Regression models for our analysis:**
- Linear Regression:** Chosen for its simplicity and interpretability, linear regression captures the linear relationship between input variables and the target variable. We employed forward selection to select the relevant features and experimented with various normalization methods. To ensure reliable estimation, we performed 10-fold cross-validation and evaluated the models using mean squared error (MSE) and R-squared measures. We selected the model with the highest R-squared and lowest MSE. Further refinement will involve fine-tuning the model hyperparameter. For more information and evaluation on the model, see Appendix K.
 - Polynomial Regression:** Used to capture non-linear relationships between input variables and the target variable, polynomial regression also employed forward selection and different normalization methods. Similarly, we performed 10-fold cross-validation for robust estimation and fine-tuned the model by determining the optimal degree of the polynomial. Further refinement will involve fine-tuning the model hyperparameter. For more information and evaluation on the model, see Appendix L.

In-depth evaluation and comparison of the models in the next step.

- 8. Financial Potential Analysis:** To assess the financial potential given a specific budget, two-line plots were generated. The first line plot displayed the cumulative loan amount against the expected yield predicted by the model, providing insights into potential returns at different budget levels. The second line plot compared the cumulative loan amount with the real yield, allowing for an evaluation of how accurately the models captured the actual returns. These visualizations offered a concise understanding of the financial outcomes based on budget and the performance of the models in predicting yields. The line plots can be found in Appendix M.
- 9. Potential Pitfalls:** Throughout our analysis, we encountered numerous potential pitfalls that warrant careful attention. These pitfalls encompass aspects such as model evaluation, multicollinearity, overfitting, skewness handling, and other crucial factors. To provide a comprehensive understanding of these pitfalls and their implications, we have detailed them extensively in Appendix N, ensuring transparency and guiding future research endeavors.

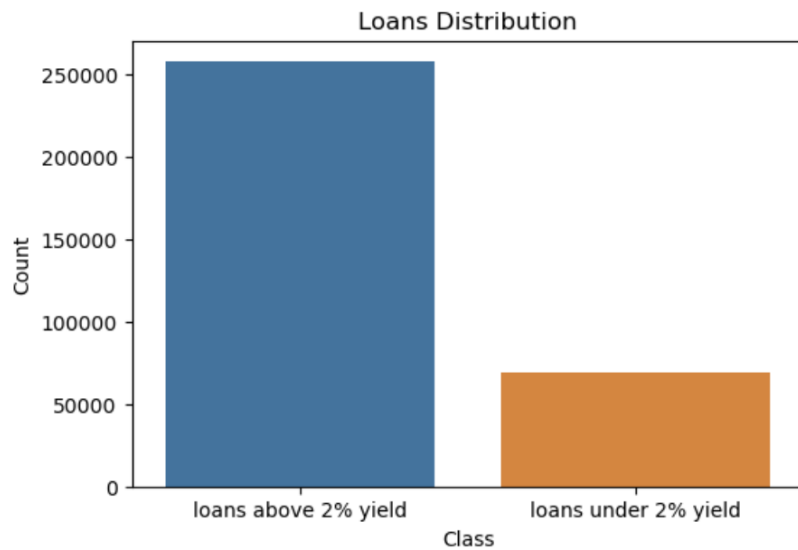
In the next step, our focus will be on conducting a thorough investigation into the models' hyperparameters, performing extensive evaluations and comparisons among the models and with the existing model (grade), analyzing the impact of addressing the skewed distribution problem on the models, conducting a comprehensive Financial Potential Analysis, and ultimately selecting the most suitable model.

Best regards,

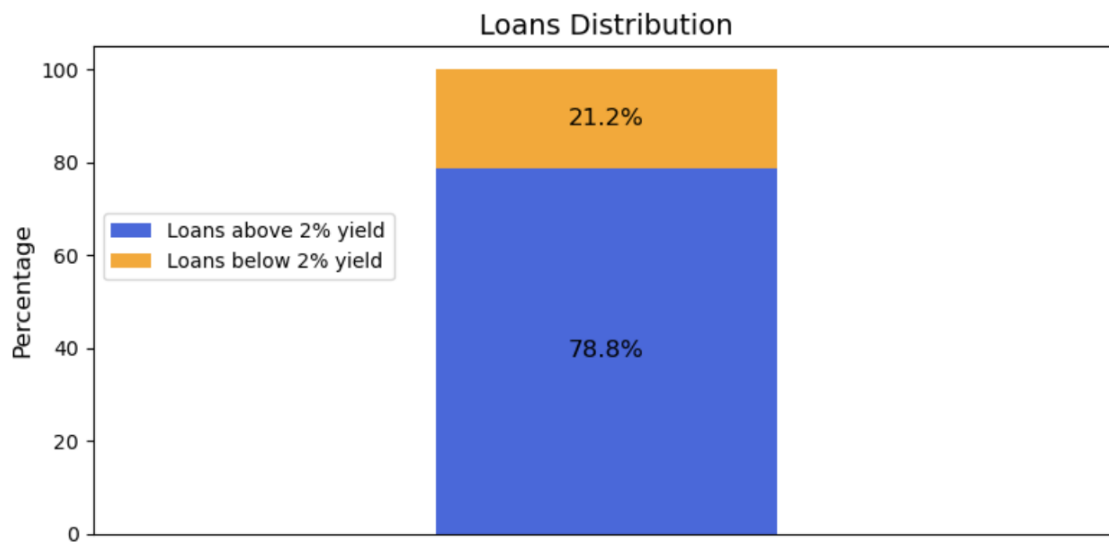
The DataDriven Portfolio Solutions Management Team

Appendices

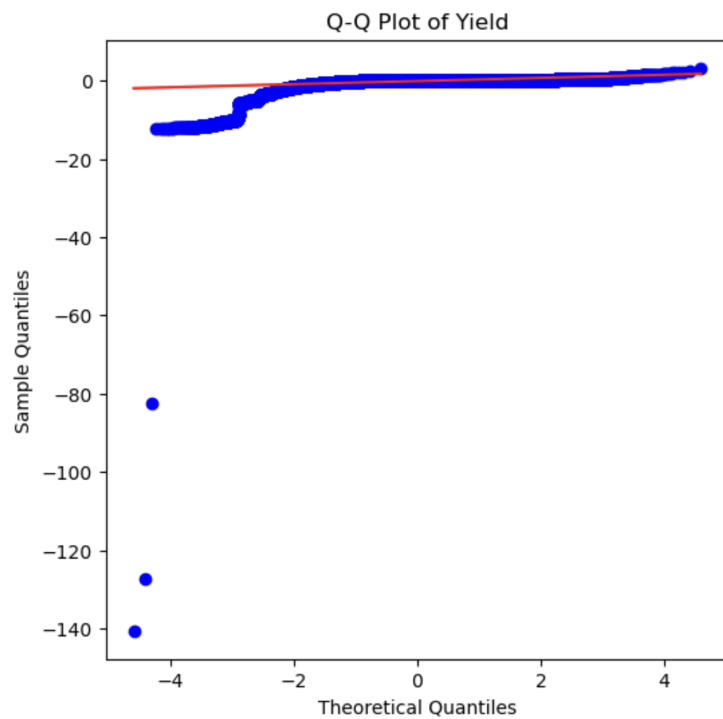
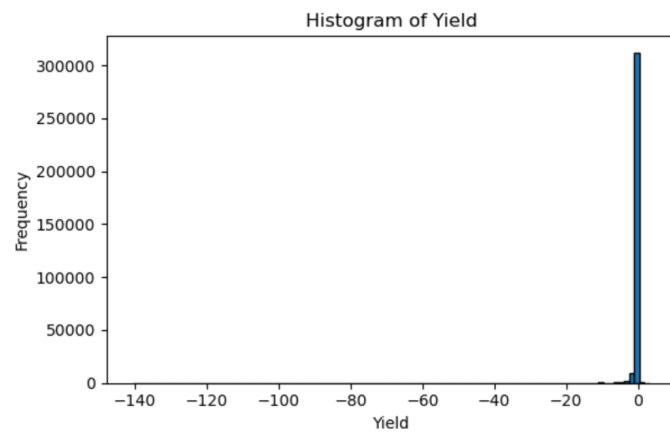
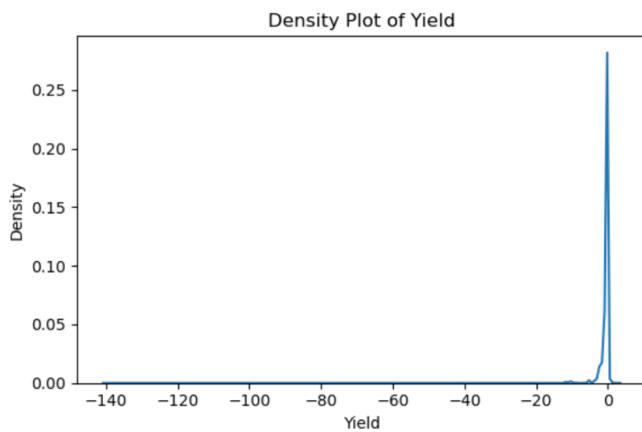
Appendix A-Distribution of the binary target variable:



We have 257230 loans above 2% yield (21.2%) 69173 loans below 2% yield (78.8%)



Appendix B-Target continuous variable distribution:



```
# Calculate skewness
skewness = stats.skew(data['yield_yearly_1'])
print("Skewness:", skewness)

# Shapiro-Wilk test for normality
_, p_value = stats.shapiro(data['yield_yearly_1'])
print("Shapiro-Wilk p-value:", p_value)
```

Skewness: -40.13921158421023
Shapiro-Wilk p-value: 0.0

Appendix C-Comparison between 70:30 and 80:20:

Logistic Regression Performance

Comparison

		LG by 80:20	LG by 70:30
0	F1	0.716291	0.721337
1	recall	0.735301	0.739657
2	precision	0.703312	0.709107
3	accuracy	0.735301	0.739657

XG-Boost Performance Comparison

		XG-Boost by 80:20	XG-Boost by 70:30
0	F1	0.610538	0.612292
1	recall	0.610863	0.612698
2	precision	0.611239	0.613158
3	accuracy	0.610863	0.612698

AdaBoost Performance Comparison

		AdaBoost by 80:20	AdaBoost by 70:30
0	F1	0.545727	0.604397
1	recall	0.565866	0.604608
2	precision	0.580227	0.604841
3	accuracy	0.565866	0.604608

	Model	MSE	R ²
0	PR Degree-2 (70:30)	0.007032	0.030277
1	PR Degree-3 (70:30)	0.007140	0.015245
2	PR Degree-2 (80:20)	0.007042	0.030021
3	PR Degree-3 (80:20)	0.007182	0.011088

Linear regression

	Normalization	Split	MSE	R ² Score
0	RobustScaler	80:20	0.007101	0.023337
1	RobustScaler	70:30	0.007098	0.023385
2	min-max	70:30	0.007203	0.212900
3	min-max	80:20	0.007303	0.202900
4	standard	70:30	0.007156	0.232400
5	standard	80:20	0.007178	0.231000

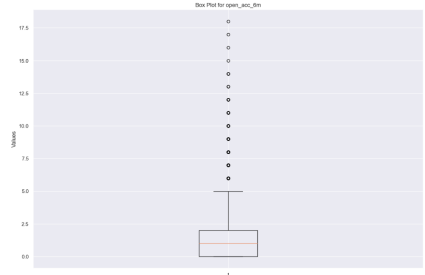
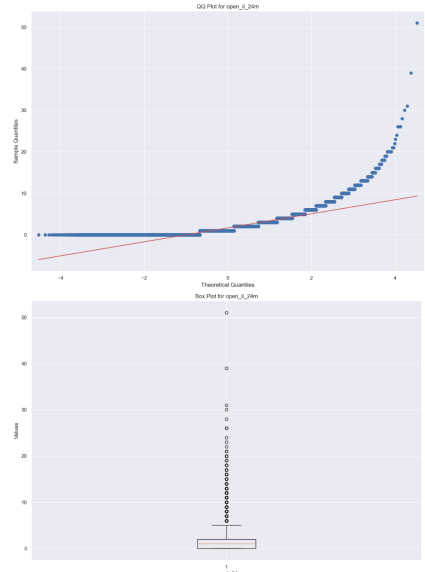
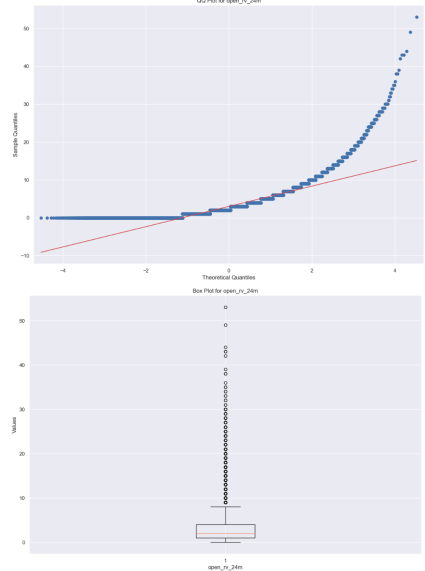
Appendix D: Logistic Regression Model and Methodology-

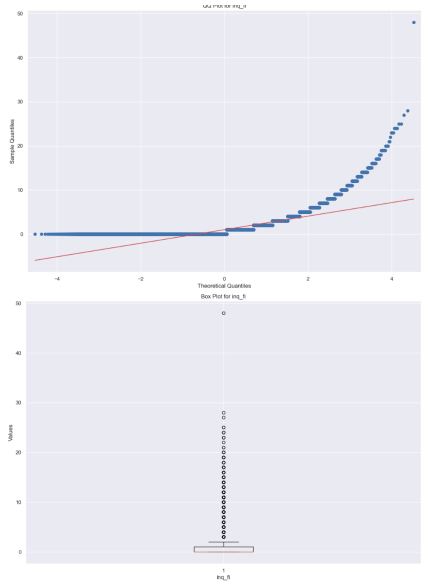
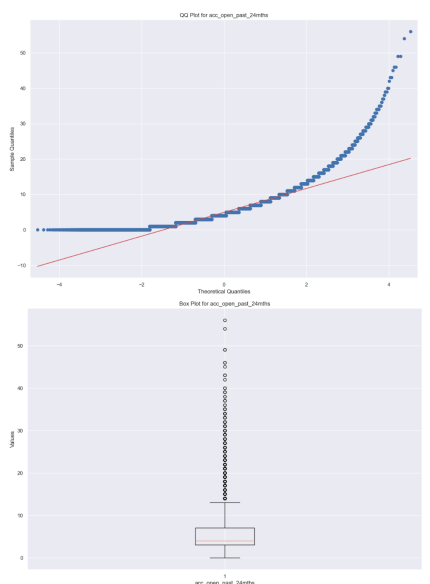
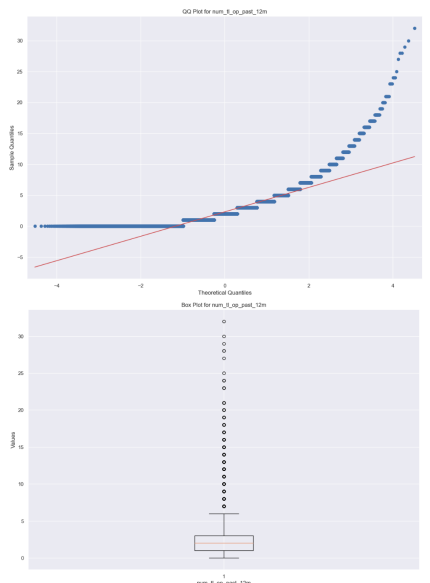
To develop an effective logistic regression model, we implemented a series of steps and made specific choices to ensure its accuracy and reliability. The following is an overview of the model and methodology employed:

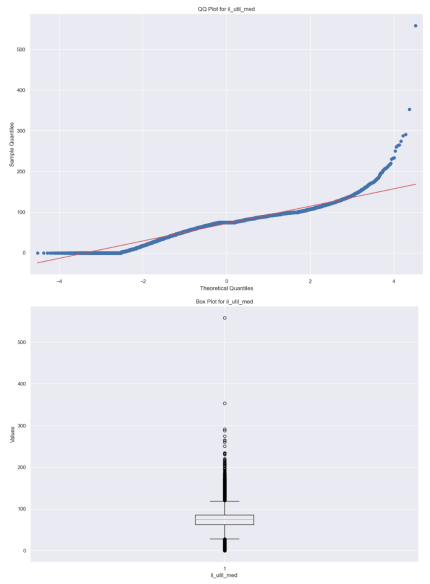
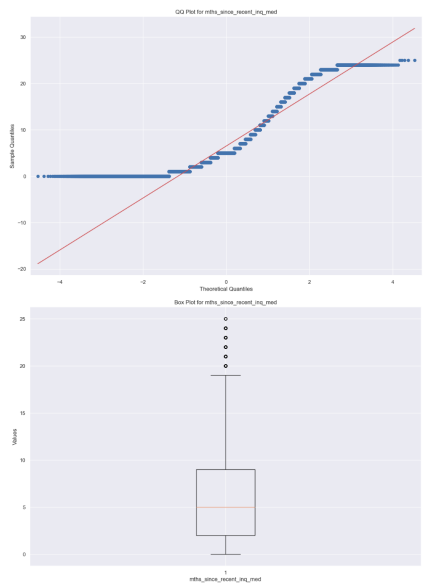
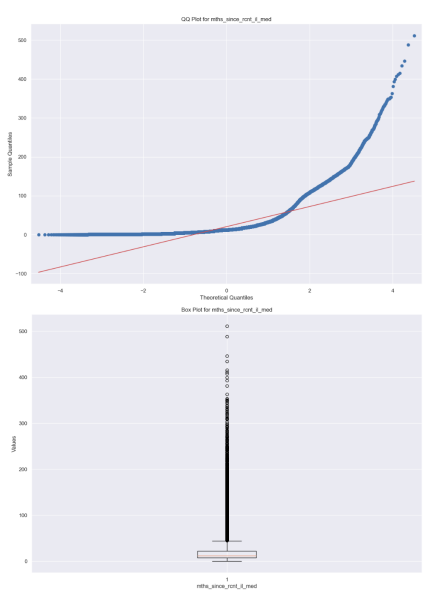
Feature Preselection: Feature preselection is crucial to identify the most relevant predictor variables for our logistic regression model. We chose to use forward selection, a method that sequentially adds variables based on their predictive power. This approach helps us avoid the curse of dimensionality and strike a balance between model complexity and performance. Choice of k_features- To determine the optimal number of features, we selected three values for k_features: 10, 15, and 20. By considering a minimum of 10 features, we aimed to avoid an overly simplistic model. Additionally, capping it at 20 features prevents excessive complexity. This range allows us to explore a sufficient number of predictors without compromising model performance.

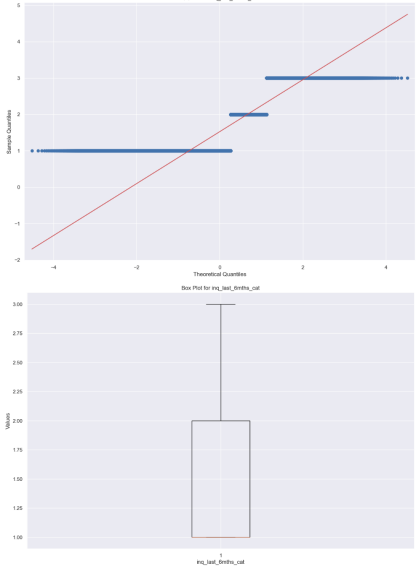
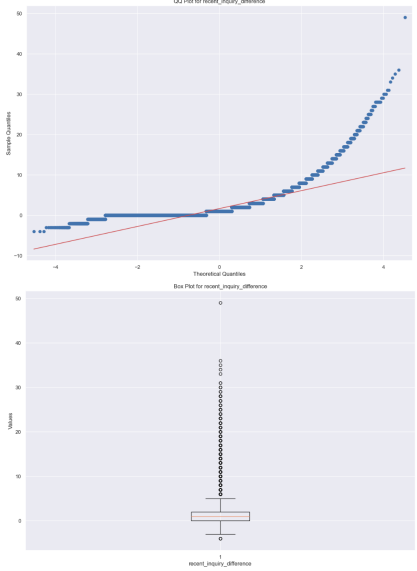
List of the best model features and normalization type:

<u>Features Name:</u>	<u>Normalization type:</u>	<u>Graph:</u>
dti	Robust normalization- Considering the presence of outliers and non-normal data distribution, conventional normalization methods like minimum-maximum scale and standard normalization are not suitable. Instead, Robust normalization is a robust and versatile approach that handles outliers effectively and doesn't rely on the assumption of a normal distribution.	
open_acc_6m	Robust normalization- Considering the presence of outliers and non-normal data distribution, conventional normalization methods like minimum-maximum scale and standard normalization are not suitable. Instead, Robust normalization is a robust and versatile approach that handles outliers effectively and doesn't rely on the assumption of a	

	normal distribution.	 <p>Box Plot for open_1m_6m</p> <p>This box plot shows the distribution of values for the variable 'open_1m_6m'. The y-axis is labeled 'Values' and ranges from 0.0 to 17.5. The plot shows a very narrow box with a median near 0.0 and a long upper whisker extending to approximately 5.0. Numerous outliers are plotted as individual points above the whisker, reaching up to 17.5.</p>
open_il_24m	Robust normalization- Considering the presence of outliers and non-normal data distribution, conventional normalization methods like minimum-maximum scale and standard normalization are not suitable. Instead, Robust normalization is a robust and versatile approach that handles outliers effectively and doesn't rely on the assumption of a normal distribution.	 <p>QQ Plot for open_il_24m</p> <p>Box Plot for open_il_24m</p> <p>The top plot is a Q-Q Plot for 'open_il_24m'. The x-axis is 'Theoretical Quantiles' ranging from -4 to 4, and the y-axis is 'Sample Quantiles' ranging from 0 to 50. The data points (blue dots) follow a red diagonal line for most of the range but curve sharply upwards for theoretical quantiles greater than 2, indicating a heavy right tail. The bottom plot is a box plot for the same variable, showing a median near 0, a box extending from approximately -1 to 1, and a long upper whisker with many outliers plotted as points above it, reaching up to 50.</p>
open_rv_24m	Robust normalization- Considering the presence of outliers and non-normal data distribution, conventional normalization methods like minimum-maximum scale and standard normalization are not suitable. Instead, Robust normalization is a robust and versatile approach that handles outliers effectively and doesn't rely on the assumption of a normal distribution.	 <p>QQ Plot for open_rv_24m</p> <p>Box Plot for open_rv_24m</p> <p>The top plot is a Q-Q Plot for 'open_rv_24m'. The x-axis is 'Theoretical Quantiles' ranging from -4 to 4, and the y-axis is 'Sample Quantiles' ranging from -10 to 50. The data points (blue dots) follow a red diagonal line for most of the range but curve sharply upwards for theoretical quantiles greater than 2, indicating a heavy right tail. The bottom plot is a box plot for the same variable, showing a median near 0, a box extending from approximately -1 to 1, and a long upper whisker with many outliers plotted as points above it, reaching up to 50.</p>

inq_fi	<p>Robust normalization- Considering the presence of outliers and non-normal data distribution, conventional normalization methods like minimum-maximum scale and standard normalization are not suitable. Instead, Robust normalization is a robust and versatile approach that handles outliers effectively and doesn't rely on the assumption of a normal distribution.</p>	 <p>The figure displays two diagnostic plots for the variable 'inq_fi'. The top plot is a Q-Q Plot where the sample quantiles (blue dots) follow the theoretical quantiles (red line) closely, indicating a normal distribution. The bottom plot is a box plot showing the distribution of 'inq_fi' with a median around 1, a mean slightly above 1, and a range from approximately 0 to 4.</p>
acc_open_past_24mths	<p>Robust normalization- Considering the presence of outliers and non-normal data distribution, conventional normalization methods like minimum-maximum scale and standard normalization are not suitable. Instead, Robust normalization is a robust and versatile approach that handles outliers effectively and doesn't rely on the assumption of a normal distribution.</p>	 <p>The figure displays two diagnostic plots for the variable 'acc_open_past_24mths'. The top plot is a Q-Q Plot where the sample quantiles (blue dots) follow the theoretical quantiles (red line) closely, indicating a normal distribution. The bottom plot is a box plot showing the distribution of 'acc_open_past_24mths' with a median around 1, a mean slightly above 1, and a range from approximately 0 to 4.</p>
num_tl_op_past_12m	<p>Robust normalization- Considering the presence of outliers and non-normal data distribution, conventional normalization methods like minimum-maximum scale and standard normalization are not suitable. Instead, Robust normalization is a robust and versatile approach that handles outliers effectively and doesn't rely on the assumption of a normal distribution.</p>	 <p>The figure displays two diagnostic plots for the variable 'num_tl_op_past_12m'. The top plot is a Q-Q Plot where the sample quantiles (blue dots) follow the theoretical quantiles (red line) closely, indicating a normal distribution. The bottom plot is a box plot showing the distribution of 'num_tl_op_past_12m' with a median around 1, a mean slightly above 1, and a range from approximately 0 to 4.</p>

<p>il_util_med</p>	<p>Robust normalization- Considering the presence of outliers and non-normal data distribution, conventional normalization methods like minimum-maximum scale and standard normalization are not suitable. Instead, Robust normalization is a robust and versatile approach that handles outliers effectively and doesn't rely on the assumption of a normal distribution.</p>	 <p>The figure displays two diagnostic plots for the variable 'il_util_med'. The top plot is a Q-Q Plot titled 'Q-Q Plot for il_util_med', showing Sample Quantiles on the y-axis (ranging from 0 to 500) against Theoretical Quantiles on the x-axis (ranging from -4 to 4). The data points follow a red diagonal line, indicating a normal distribution. The bottom plot is a Box Plot titled 'Box Plot for il_util_med', showing the distribution of values on the y-axis (ranging from 0 to 500) against the variable 'il_util_med' on the x-axis. The box plot shows a median around 100, with whiskers extending from approximately 0 to 400, and several outliers plotted as individual points above the upper whisker.</p>
<p>mths_since_recent_inq_med</p>	<p>Robust normalization- Considering the presence of outliers and non-normal data distribution, conventional normalization methods like minimum-maximum scale and standard normalization are not suitable. Instead, Robust normalization is a robust and versatile approach that handles outliers effectively and doesn't rely on the assumption of a normal distribution.</p>	 <p>The figure displays two diagnostic plots for the variable 'mths_since_recent_inq_med'. The top plot is a Q-Q Plot titled 'Q-Q Plot for mths_since_recent_inq_med', showing Sample Quantiles on the y-axis (ranging from -20 to 30) against Theoretical Quantiles on the x-axis (ranging from -4 to 4). The data points follow a red diagonal line, indicating a normal distribution. The bottom plot is a Box Plot titled 'Box Plot for mths_since_recent_inq_med', showing the distribution of values on the y-axis (ranging from 0 to 25) against the variable 'mths_since_recent_inq_med' on the x-axis. The box plot shows a median around 5, with whiskers extending from approximately 0 to 20, and several outliers plotted as individual points above the upper whisker.</p>
<p>mths_since_rcnt_il_med</p>	<p>Robust normalization- Considering the presence of outliers and non-normal data distribution, conventional normalization methods like minimum-maximum scale and standard normalization are not suitable. Instead, Robust normalization is a robust and versatile approach that handles outliers effectively and doesn't rely on the assumption of a normal distribution.</p>	 <p>The figure displays two diagnostic plots for the variable 'mths_since_rcnt_il_med'. The top plot is a Q-Q Plot titled 'Q-Q Plot for mths_since_rcnt_il_med', showing Sample Quantiles on the y-axis (ranging from -100 to 500) against Theoretical Quantiles on the x-axis (ranging from -4 to 4). The data points follow a red diagonal line, indicating a normal distribution. The bottom plot is a Box Plot titled 'Box Plot for mths_since_rcnt_il_med', showing the distribution of values on the y-axis (ranging from 0 to 500) against the variable 'mths_since_rcnt_il_med' on the x-axis. The box plot shows a median around 10, with whiskers extending from approximately 0 to 400, and several outliers plotted as individual points above the upper whisker.</p>

Is_MORTGAGE	Feature binary- there is no need for normalization.	
Is_RENT	Feature binary- there is no need for normalization.	
inq_last_6mths_cat	Robust normalization- Considering the presence of outliers and non-normal data distribution, conventional normalization methods like minimum-maximum scale and standard normalization are not suitable. Instead, Robust normalization is a robust and versatile approach that handles outliers effectively and doesn't rely on the assumption of a normal distribution.	 <p>The figure consists of two plots for the variable 'inq_last_6mths_cat'. The top plot is a Q-Q Plot showing Sample Quantiles on the y-axis (ranging from -2 to 5) against Theoretical Quantiles on the x-axis (ranging from -4 to 4). The data points (blue dots) deviate significantly from the red diagonal line, indicating a non-normal distribution. The bottom plot is a Box Plot showing the distribution of 'inq_last_6mths_cat' with a y-axis ranging from 100 to 300. The box plot shows a median around 150, with a long upper whisker and several outliers above 200.</p>
recent_inquiry_difference	Robust normalization- Considering the presence of outliers and non-normal data distribution, conventional normalization methods like minimum-maximum scale and standard normalization are not suitable. Instead, Robust normalization is a robust and versatile approach that handles outliers effectively and doesn't rely on the assumption of a normal distribution.	 <p>The figure consists of two plots for the variable 'recent_inquiry_difference'. The top plot is a Q-Q Plot showing Sample Quantiles on the y-axis (ranging from -10 to 50) against Theoretical Quantiles on the x-axis (ranging from -4 to 4). The data points (blue dots) follow the red diagonal line closely for most of the range but curve upwards at the high end, indicating a right-skewed distribution. The bottom plot is a Box Plot showing the distribution of 'recent_inquiry_difference' with a y-axis ranging from 0 to 50. The box plot shows a median around 5, with a long upper whisker and many outliers extending up to 50.</p>

Hyperparameter Tuning: Hyperparameter tuning is essential to optimize the logistic regression model, We provided an explanation of the significance and rationale for this step in section 5b of the report. We focused on three key hyperparameters: maximum number of iterations (n_iter), tolerance (eps), and penalty types (penalty). We tested various combinations of these hyperparameters. Due to the limitation of computational power, we chose only these parameters and limited the number of combinations.

- n_iter: We considered three values, 1000, 2000, and 5000, to strike a balance between convergence and computational efficiency. Higher values allow for more iterations, potentially improving convergence and accuracy.
- eps: We explored a range of values, including 0.01, 0.001, 0.0001, 0.00001, and 0.000001, to control the convergence threshold. A lower epsilon value indicates higher precision in achieving convergence.
- penalty: We assessed two penalty types, 'l1' and 'l2', to evaluate the impact of regularization on the model's performance. The choice of penalty type helps control overfitting and influences the selection of relevant features.

Skewness Handling and Threshold Adjustment: In addressing the imbalanced distribution of classes, we used the 'class_weight' argument to handle skewness in the data. Additionally, we evaluated the model's performance at multiple classification thresholds: 0.40, 0.50, and 0.60. Adjusting the threshold allows us to strike a balance between false positives and false negatives, considering the business impact and the trade-off between investment opportunities and risks.

The results of the best logistic regression model, including the chosen hyperparameter combinations, selected features, evaluated thresholds, and their corresponding F1 scores, can be found in Appendix E. These results provide valuable insights into the model's performance and guide decision-making in achieving accurate predictions.

Evaluation Metric: To assess the model's performance, we employed k-fold cross-validation with k=5. This technique ensures reliable and robust evaluation by randomly dividing the data into training and validation sets, maintaining a 70:30 ratio. The F1 score, which considers the weighted average across the classes, was chosen as the evaluation metric. It provides a comprehensive assessment of both precision and recall. We provided an explanation of the significance and rationale for our evaluation metric in section 5a and 3 of the report.

Best Model Selection: Based on the averaged F1 scores, we identified the best-performing AdaBoost models.

The results of the best Logistic Regression models, including the selected hyperparameter combinations and their corresponding F1 scores, can be found in Appendix E, these results provide insights into the performance of the AdaBoost models.

Appendix E- Logistic Regression Model Evaluation Results and Best Hyperparameters:

Best model parameters:

n_iter	eps	penalty	selected_features_list	threshold
5,000	1e-06	l2	[0, 1, 2, 4, 6, 7, 11, 12, 13, 14, 16, 17, 18, 19]	0.4

Best model evaluation:

+	-----	+	-----	+
	Metric		Value	
+	=====	+	=====	+
	F1		0.721943	
+	-----	+	-----	+
	recall		0.740397	
+	-----	+	-----	+
	precision		0.709648	
+	-----	+	-----	+
	accuracy		0.740397	
+	-----	+	-----	+

Appendix F: XGBoost Model and Methodology

The XGBoost (Extreme Gradient Boosting) algorithm was chosen for its superior performance in handling complex datasets and its ability to finely tune hyperparameters for optimal model performance. This appendix provides an overview of the XGBoost model and the methodology employed.

Hyperparameter Selection: To optimize the XGBoost model, an extensive hyperparameter search was conducted. Three key hyperparameters were focused on: maximum depth (max_depth), learning rate (eta), and the number of estimators (n_estimators). Various combinations of these hyperparameters were tested to identify the best-performing configuration. Due to computational limitations, a limited number of parameter combinations were considered. The ranges explored for each hyperparameter were as follows:

- max_depth: [1, 2, 3]
- eta (learning rate): [0.1, 0.3, 0.4]
- n_estimators: [30, 50, 80]

A list of combinations of max_depth, eta, and n_estimators was generated using the itertools library. From this list, 10 combinations were randomly selected for evaluation, considering the computational constraints (due to computational limitations). These selected combinations were used to train and evaluate multiple XGBoost models.

Skewness Handling, Oversampling: To address the issue of class imbalance in the dataset, the RandomOverSampler technique was employed. This technique oversamples the minority class to achieve a balanced distribution. This step was crucial to prevent the model from being biased toward the majority class. The oversampling process was applied before each fold of cross-validation to ensure fair evaluation.

Feature selection: XGBoost determines feature importance by assessing their impact on reducing model loss during training. These scores guide feature selection and offer insights into their relative significance. After training, the feature importance scores provide insights into the relative significance of each feature in the model. However, feature importance alone does not establish relevance or causality.

The feature names and importance scores:

Feature Name	Importance
fico_mean	0.204
Is_MORTGAGE	0.166
acc_open_past_24mths	0.142
inq_last_6mths_cat	0.069
mort_acc	0.062

dti	0.060
ls_RENT	0.050
mths_since_recent_inq_med	0.045
avg_cur_bal	0.041
all_util	0.026
num_tl_op_past_12m	0.025
inq_fi	0.016
il_util_med	0.016
mths_since_rcnt_il_med	0.013
recent_inquiry_difference	0.012
open_acc_6m	0.012
mths_since_recent_inq	0.012
open_rv_24m	0.011
il_util	0.011
open_il_24m	0.006
mths_since_rcnt_il_mean	0.000

The scores indicate the relative significance of each feature in making accurate predictions. Features like "fico_mean," "ls_MORTGAGE," and "acc_open_past_24mths" have the highest importance, while others like "mths_since_rcnt_il_mean" and "open_il_24m" have relatively lower importance in the model.

Feature Scaling: Three normalization techniques were implemented to address feature scaling variations and enhance model performance: Min-Max scaling (scaler_minmax), standard scaling (scaler_std), and robust scaling. These techniques ensured consistent feature values during model training. Cross-validation was performed individually for each fold to prevent data leakage. **The robust scaling technique exhibited the best performance and was consequently applied to the model.**

Evaluation Metric: To assess the model's performance, we employed k-fold cross-validation with k=5. This technique ensures reliable and robust evaluation by randomly dividing the data into training and validation sets, maintaining a 70:30 ratio. The F1 score, which considers the weighted average across the classes, was chosen as the evaluation metric. It provides a comprehensive assessment of both precision and recall. We provided an explanation of the significance and rationale for our evaluation metric in section 5a and 3 of the report.

Best Model Selection: Based on the averaged F1 scores, we identified the best-performing AdaBoost models.

The results of the best XG-Boost models, including the selected hyperparameter combinations and their corresponding F1 scores, can be found in Appendix G, these results provide insights into the performance of the AdaBoost models.

Appendix G-XGBoost Model Evaluation Results and Best Hyperparameters:

Best model parameters:

max_depth	eta	n_estimators	selected_features_list
3	0.4	50	[20, 16, 7, 18, 9, 0, 17, 13, 8, 5, 11, 6, 12, 14, 19, 1, 10, 4, 3, 2, 15]

Best model evaluation:

Metric	Value
F1	0.612419
Recall	0.612795
Precision	0.613225
Accuracy	0.612795

Appendix H: AdaBoost Model and Methodology

The AdaBoost (Adaptive Boosting) algorithm was chosen for its sequential boosting approach, which enhances weak classifiers and effectively handles challenging instances, particularly in scenarios with class imbalance or overlapping decision boundaries. This appendix provides an overview of the AdaBoost model, and the methodology employed.

Hyperparameter Selection: To optimize the AdaBoost model, we conducted an extensive hyperparameter search. We focused on three key hyperparameters: maximum depth (`max_depth`), learning rate (`learning_rate`), and the number of estimators (`n_estimators`). By testing various combinations of these hyperparameters, we aimed to identify the best-performing configuration. Due to the limitation of computational power, we chose only these parameters and limited the number of combinations. The ranges considered for each hyperparameter were as follows:

- `max_depth`: [1, 2, 3] We tested `max_depth` values of 1, 2, and 3 to control the complexity of decision trees in AdaBoost, balancing model intricacy and generalization.
- `learning_rate`: [0.1, 0.5, 0.6] We explored learning rates of 0.1, 0.5, and 0.6 to adjust the influence of each base classifier in AdaBoost, aiming to find the right balance between classifier contribution and overall model performance.
- `n_estimators`: [30, 50, 80] We considered 30, 50, and 80 as values for the number of estimators in AdaBoost, balancing model complexity and computational efficiency.

We generated a list of combinations of `max_depth`, `learning_rate`, and `n_estimators` using the `itertools` library. From this list, we randomly selected 10 combinations for evaluation (due to the limitation of computational power). These selected combinations were used to train and evaluate multiple AdaBoost models.

Skewness Handling, Oversampling: To address class imbalance in the dataset, we employed the `RandomOverSampler` technique, which oversamples the minority class to achieve a balanced distribution. This step was crucial to prevent the model from being biased toward the majority class. The oversampling process was applied before each fold of cross-validation to ensure fair evaluation.

Feature selection: AdaBoost determines feature importance by assessing their impact on improving model performance during training. These scores guide feature selection and offer insights into their relative significance. After training, the feature importance scores provide insights into the relative significance of each feature in the model. However, feature importance alone does not establish relevance or causality.

The feature names and importance scores for AdaBoost are as follows:

Feature Name	Importance
fico_mean	0.167

dti	0.133
avg_cur_bal	0.133
all_util	0.100
acc_open_past_24mths	0.100
mort_acc	0.067
inq_last_6mths_cat	0.067
inq_fi	0.033
il_util_med	0.033
mths_since_recent_inq_med	0.033
mths_since_rcnt_il_mean	0.033
ls_MORTGAGE	0.033
ls_RENT	0.033
recent_inquiry_difference	0.033
open_acc_6m	0.015
open_il_24m	0.015
il_util	0.015
open_rv_24m	0.015
mths_since_recent_inq	0.015
num_tl_op_past_12m	0.001
mths_since_rcnt_il_med	0.000

The AdaBoost model highlights important features such as 'fico_mean', 'dti', and 'avg_cur_bal', which significantly impact its performance. However, feature importance alone does not establish relevance or causality.

Feature Scaling: Three normalization techniques were implemented to address feature scaling variations and enhance model performance: Min-Max scaling (scaler_minmax), standard scaling (scaler_std), and robust scaling. These techniques ensured consistent feature values during model training. Cross-validation was performed individually for each fold to prevent data leakage. **The robust scaling technique exhibited the best performance and was consequently applied to the model.**

Evaluation Metric: To robustly assess the performance of the AdaBoost models, we implemented 5-fold cross-validation. This technique randomly divided the data into training and validation sets, maintaining a 70:30 ratio, for each fold. This ensured representative data distribution and reduced the risk of overfitting. The evaluation metric used was the F1 score, which provides a comprehensive evaluation of both precision and recall. The F1 scores were calculated for each model on the validation set of each fold and then averaged across the five folds, providing a reliable and unbiased assessment of the models' performance. The F1 scores were weighted averages across the classes, providing an overall measure of the models' effectiveness in capturing both precision and recall. We provided an explanation of the significance and rationale for our evaluation metric in section 5a and 3 of the report.

Best Model Selection: Based on the averaged F1 scores, we identified the best-performing AdaBoost models.

The results of the best AdaBoost models, including the selected hyperparameter combinations and their corresponding F1 scores, can be found in Appendix I, these results provide insights into the performance of the AdaBoost models.

Appendix I- AdaBoost Model Evaluation Results and Best Hyperparameters:

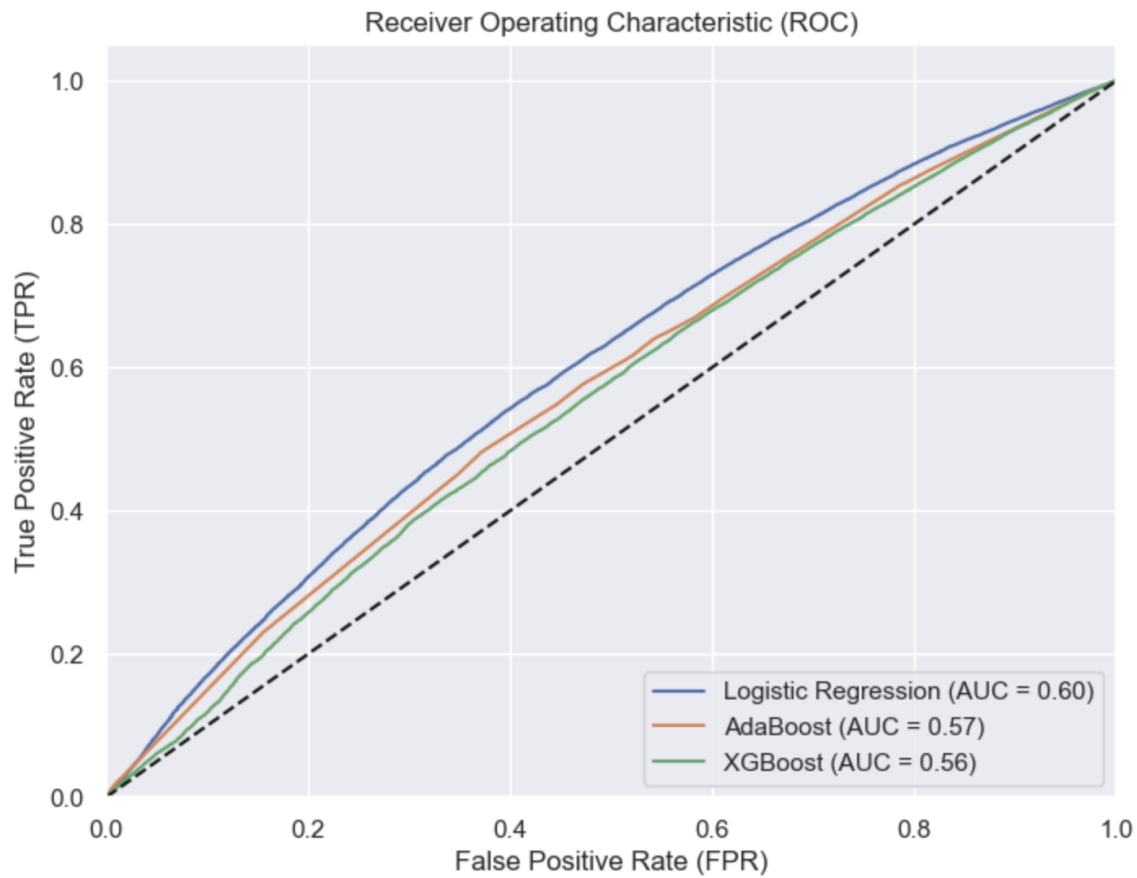
Best model parameters:

max_depth	learning_rate	n_estimators	selected_features_list
5,000	0.6	30	[20, 0, 8, 5, 7, 9, 18, 6, 12, 13, 15, 16, 17, 19, 1, 2, 10, 11, 14]

Best model evaluation:

Metric	Value
F1	0.604113
Recall	0.6043
Precision	0.60448
Accuracy	0.6043

Appendix J- Evaluation of the classification models:



	<u>Logistic Regression</u>	<u>AdaBoost</u>	<u>XGBoost</u>
<u>F1</u>	0.722	0.604	0.612
<u>Recall</u>	0.714	0.604	0.6124
<u>Precision</u>	0.710	0.6045	0.6128
<u>Accuracy</u>	0.741	0.604	0.612

Appendix K: Linear Regression Model Methodology-

Feature Preselection: Feature preselection is crucial to identify the most relevant predictor variables for our linear regression model. We chose to use forward selection, a method that sequentially adds variables based on their predictive power. This approach helps us avoid the curse of dimensionality and strike a balance between model complexity and performance.

We chose to start with the best 30 features that give the lowest mse score.

Skewness Handling : RobustScaler is a useful data normalization technique in situations where data contains outliers, is skewed, or when preserving relative relationships between data points is important. It helps improve the robustness and reliability of regression models, particularly in scenarios where other scaling methods may be influenced by extreme values or assumptions about data distribution.

The model is first optimized on the training set, and then its parameters are tested on the testing set to verify the model is fitting (using cross validation which we tested its impact on the model by 10 folds). This step also helps us to detect overfitting. We examined two options 80:20 and 70:30 (majority for training and the rest for testing) to see the impact on model performance and found out that using 70% of the data to train the model and 30% to test it yield better performance.

Feature Scaling: Three normalization techniques were implemented to address feature scaling variations and enhance model performance: Min-Max scaling (scaler_minmax), standard scaling (scaler_std), and robust scaling. These techniques ensured consistent feature values during model training. Cross-validation was performed individually for each fold to prevent data leakage. **The robust scaling technique exhibited the best performance and was consequently applied to the model.**

Evaluation Metric: In evaluating our regression model, we utilize two key metrics: Mean Squared Error (MSE) and R-squared (R^2). MSE provides a measure of the average squared difference between predicted and actual values, allowing us to assess the accuracy of our predictions. Lower MSE values indicate better predictive performance. On the other hand, R^2 represents the proportion of variance in the dependent variable that can be explained by the independent variables in the model. Higher R^2 values indicate a better fit of the model to the data. By considering both MSE and R^2 , we obtain insights into the accuracy and goodness of fit of our regression model, enabling us to make informed decisions in the business context. Furthermore, we investigated the cumulative loan amount to the weighted yield.

A summary table of the models according to the distribution ratio and the normalization method:

<u>MSE</u>	<u>R2_Score</u>	<u>Normalization</u>	<u>split</u>	<u>Learning curve</u>	<u>Cumulative Loan Amount vs Yield</u>
0.007101	0.023337	RobustScaler	80:20		
0.007098	0.023385	RobustScaler	70:30		
0.007203	0.2129	min-max	70:30		
0.007303	0.2029	min-max	80:20		
0.007156	0.2324	standard	70:30		
0.007178	0.2310	standard	80:20		

	Normalization	Split	MSE	R ² Score
0	RobustScaler	80:20	0.007101	0.023337
1	RobustScaler	70:30	0.007098	0.023385
2	min-max	70:30	0.007203	0.212900
3	min-max	80:20	0.007303	0.202900
4	standard	70:30	0.007156	0.232400
5	standard	80:20	0.007178	0.231000

Appendix L: Polynomial Regression Model and Methodology -

Feature Preselection: Feature preselection is crucial to identify the most relevant predictor variables for our linear regression model. We chose to use forward selection, a method that sequentially adds variables based on their predictive power. This approach helps us avoid the curse of dimensionality and strike a balance between model complexity and performance.

Skewness Handling :Polynomial regression is effective for handling skewness in data due to its ability to capture non-linear relationships. Skewness represents the asymmetry in data distribution, and linear regression may not adequately capture the non-linear patterns present in skewed data. By incorporating higher-order polynomial terms, polynomial regression can flexibly fit the data and adjust to the non-linear patterns, allowing for better approximation and improved model performance. This makes polynomial regression a suitable choice for handling skewness and providing more accurate predictions by capturing the non-linear relationships in the data.

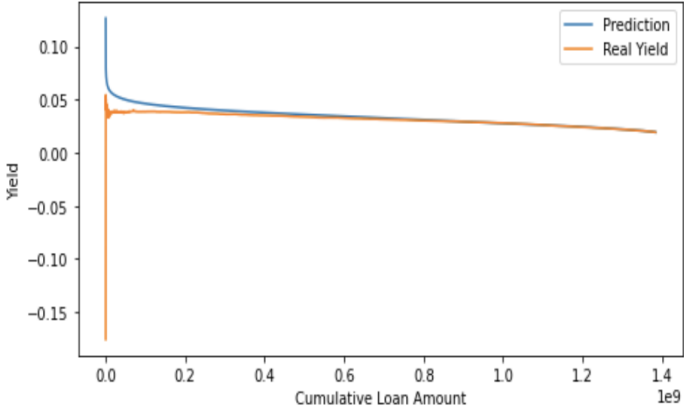
The model is first optimized on the training set, and then its parameters are tested on the testing set to verify the model is fitting (using cross validation which we tested its impact on the model by 5-10 folds). This step also helps us to detect overfitting. We examined two options 80:20 and 70:30 (majority for training and the rest for testing) to see the impact on model performance and found out that using 70% of the data to train the model and 30% to test it yield better performance.

Also we examined the performance of the model based on several degrees.

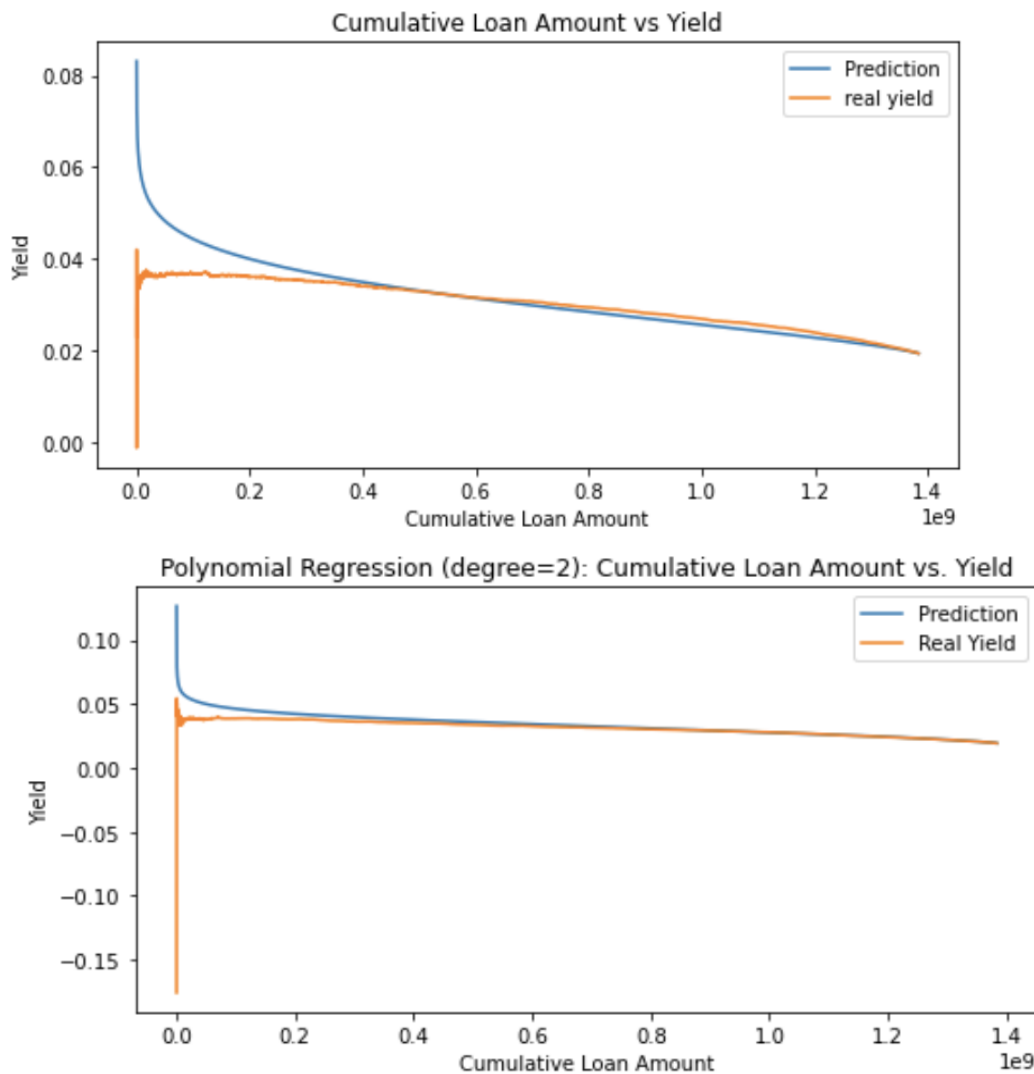
Evaluation Metric: In evaluating our regression model, we utilize two key metrics: Mean Squared Error (MSE) and R-squared (R²). MSE provides a measure of the average squared difference between predicted and actual values, allowing us to assess the accuracy of our predictions. Lower MSE values indicate better predictive performance. On the other hand, R² represents the proportion of variance in the dependent variable that can be explained by the independent variables in the model. Higher R² values indicate a better fit of the model to the data. By considering both MSE and R², we obtain insights into the accuracy and goodness of fit of our regression model, enabling us to make informed decisions in the business context.

Furthermore we investigated the cumulative loan amount to the weighted yield.

Best model parameters:

<u>Degree</u>	<u>split</u>	<u>MSE</u>	<u>R2</u>	<u>Cumulative Loan Amount vs Yield</u>
2	70:30	0.00703248	0.030021364	<p>Polynomial Regression (degree=2): Cumulative Loan Amount vs. Yield</p> 

Appendix M - Financial Potential Analysis:



The two-line plots presented in the appendix showcase the relationship between the Cumulative Loan Amount and the predicted yield and real yield, respectively. In the first graph, which represents the predictions of the linear regression model, it can be observed that higher investment amounts are associated with lower yields, although still higher than the 2% threshold. However, the model's predictions for small loan amounts are less accurate, resulting in a higher degree of error compared to the second graph.

The second graph depicts the Polynomial Regression model's predictions of yield as a function of the Cumulative Loan Amount. This model demonstrates a similar trend, indicating that increasing the investment amount leads to diminishing yields, albeit still exceeding the 2% threshold. Notably, the Polynomial Regression model outperforms the linear regression model by providing more accurate predictions across a wider range of loan amounts, resulting in reduced prediction errors.

These visualizations, showcasing the relationship between investment amount and yield for both models, offer valuable insights into the financial outcomes of different budget levels. Additionally, they highlight the superior predictive performance of the Polynomial Regression model in capturing the nuances of the data and generating more reliable predictions of yield.

Appendix N - Potential Pitfalls:

- Failure to account for changing market conditions and external factors: Ignoring the influence of evolving market conditions and external factors that can impact the model's performance and accuracy.
- Over-reliance on a 2% yield threshold, missing true potential and associated risks: Placing excessive emphasis on a specific yield threshold without considering the potential benefits and risks associated with different yield levels.
- Insufficient consideration of factors beyond yield (e.g., risk, default rates): Failing to consider other important factors, such as risk and default rates, that can significantly impact the performance and accuracy of the model.
- Failure to address potential overfitting issues: Not taking necessary steps to prevent the model from fitting noise in the training data and lacking generalization capability for unseen data.
- Inadequate assessment of alternative skewness handling methods: Not sufficiently evaluating and comparing different techniques for handling skewed data distributions.
- Failure to balance skewed distribution while maintaining interpretability: Neglecting the need to balance skewed data distributions while still ensuring the interpretability of the model's predictions and insights.
- Neglecting potential drawbacks of oversampling techniques: Failing to consider and address the potential drawbacks and limitations associated with oversampling techniques used to handle imbalanced data.
- Limited evaluation of models using relevant metrics and benchmarks: Inadequate assessment of model performance using appropriate evaluation metrics and comparison with industry benchmarks.
- Accounting for non-linear relationships for improved accuracy: Considering non-linear relationships between variables to improve the accuracy and predictive power of the model.
- Lack of comparison with existing models or approaches: Not comparing the performance and effectiveness of the developed models with existing models or approaches in the field.
- Overfitting: complex models that may overfit the training data and fail to generalize well to unseen data.
- Underfitting: overly simplistic models that fail to capture the complexity of the data, resulting in poor performance on both the training data and unseen data.
- Computational power limitations restricted model testing: Limitations in computational resources hindered thorough testing and exploration of the models' capabilities.