Auto Loan Decisions Model

Mohammad Rafiqul Islam

January 22, 2025

1 Introduction

The **Auto Loan Decisions Model** project focuses on enhancing the application decision process for auto loans, a type of secured credit that allows individuals to finance vehicle purchases with the vehicle itself serving as collateral. Leveraging data on borrower creditworthiness and demographics, we aim to build predictive models using machine learning techniques, including *Logistic Regression*, *Decision Tree Classifier*, and *Random Forest Classifier*, to accurately classify applicants into approved or rejected categories.

2 Understand the data and Exploratory Data Analysis

We are given Auto Loan account data that contains around 21,000 records in the training data and 5,400 records in the test data. Both of the datasets contain 43 columns including the target variable 'bad_flag' that contains either 0 or 1, indicating Poor/Bad Credit quality and Good Credit quality, respectively. So our goal is to make a binary classification model that successfully predict the class given other predictors.

2.1 Data Information

Both training and test set contain 34 columns with float64 type data, 6 columns with int64 type data, and 3 columns of object type data, totaling 43 columns. Both sets have many missing values.

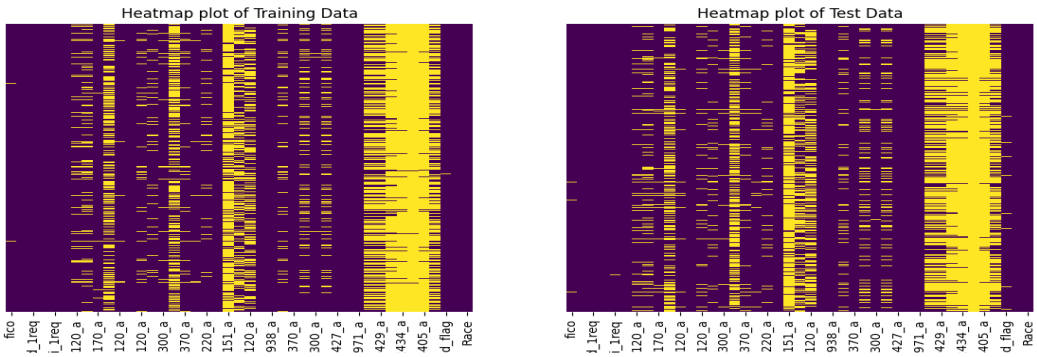

Figure 1: Heatmap plot of the missing data in both train and test data

From figure (1) we see that a few columns missing a vast amount of data. We need to handle those features and other predictors that has moderately missing data.

2.2 Exploratory Data Analysis (EDA)

2.2.1 Target Variable

So, the dependent variable or target is significantly imbalanced in class distribution with around 95.51% labeled as 'Poor Credit Quality' being the majority class and around 4.49% labeled as 'Good Credit Quality' being the minority class. Also, out of 21,606 observations, there are 258 that are missing or nan. Therefore, we have the following things consider:

1. **Before Modeling:** We need to handle the missing observations. We just can't impute the missing values in the target variable as it may introduce bias in the modeling process.
2. **While Modeling:** Since the data is imbalanced, we may need to use resampling techniques to build good predictive models. Things to consider include *undersampling the majority class* or *oversampling the minority class*. Two modeling approaches will be considered: (i) Without Resampling, and (ii) With Resampling.

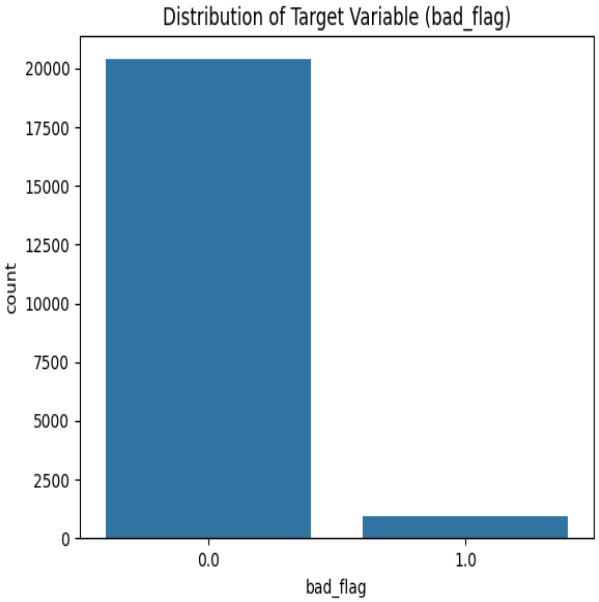

Figure 2: Distribution of the target variable

3. **After Modeling:** For the evaluation of our model, we will use 'AUC-ROC', 'F1-Score', and the 'Classification Report' as the evaluation metrics instead of accuracy.

2.2.2 Analysis of the predictors

Investigation of the missing values: Out of 42 predictors, 33 predictors contain missing values where 9 of those contains more than 50% missing values. These 9 columns are the same in both training and test data. So, we further investigate these features to see how they correlate to the target variable. We calculate each of these 9 features' correlations with the target variable and found they are all negatively correlated with values close to zero[1]. Therefore, we drop these columns while analyzing and building our models.

Next, to impute the missing values we need to see what type of data in those missing columns. We see that 'p12_all6250_a','p12_aua6200_a','p12_all2427_a', 'p12_alm6200_a', and 'p12_all6971_a' are frequency typed columns[2]. For example, 'p12_alm6200_a' is the feature that contains the worst ever status on a trade including non-medical collections and indeterminate and it takes values either 1,30,60,90, or 400. So columns like this one can be imputed by their mode.

[1]Please see table 7 in appendix A for details
[2]Please see table 8 in appendix A for details

For the continuous features such as 'fico', 'amtfinanced_1req', and other continuous features we can use median to impute the missing values. The rationale for this median technique is to avoid the effect of outliers in these features.

2.2.3 Data Visualization

First we see the categorical features and their interactions with the target variable.

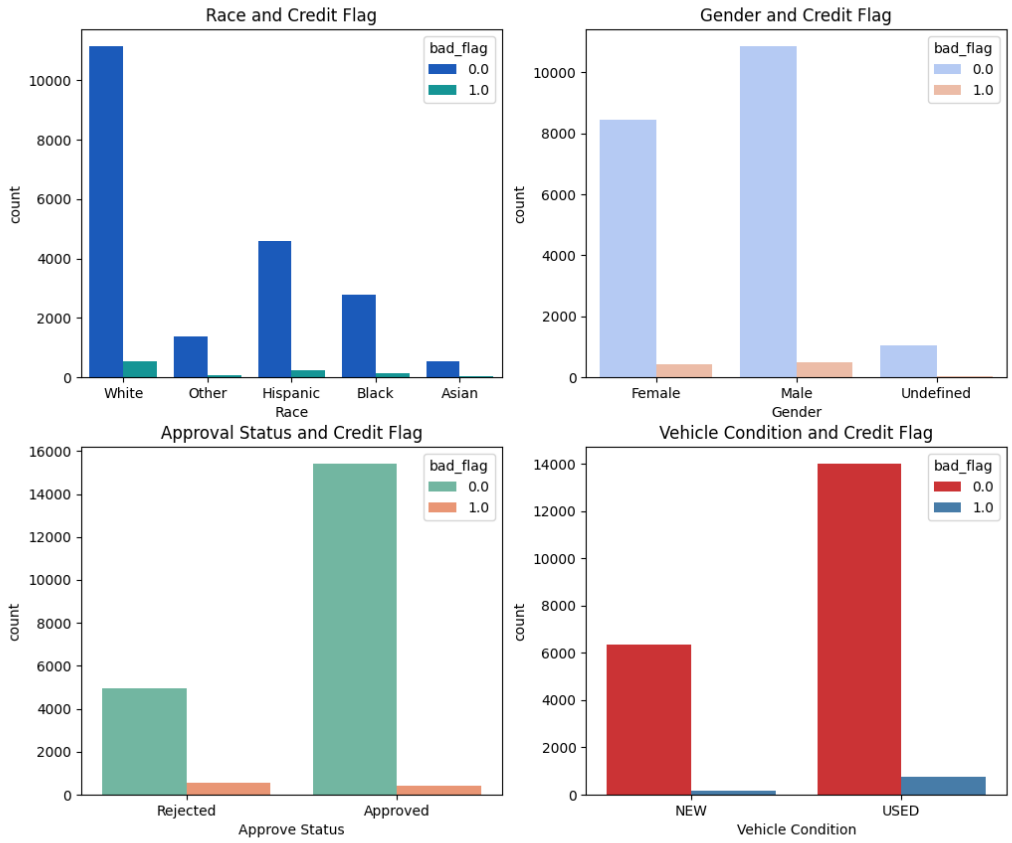

Figure 3: Bi-variate analysis of the target and categorical features

Insights: These plots highlight trends in the credit quality 'bad_flag' across different demographic and loan application attributes:
Race: White and Hispanic applicants dominate in volume, and good credit is more prevalent across all races.
Gender: Slightly more Male applicants with good credit.
Approval Status: Approved applications generally have good credit, but some rejected ones also have good credit, indicating that factors other than 'bad_flag' influence approval.
Vehicle Condition: More applicants for used vehicles, with a higher number of good credit applicants.

Then we see the bi-variate analysis of those frequency based features.

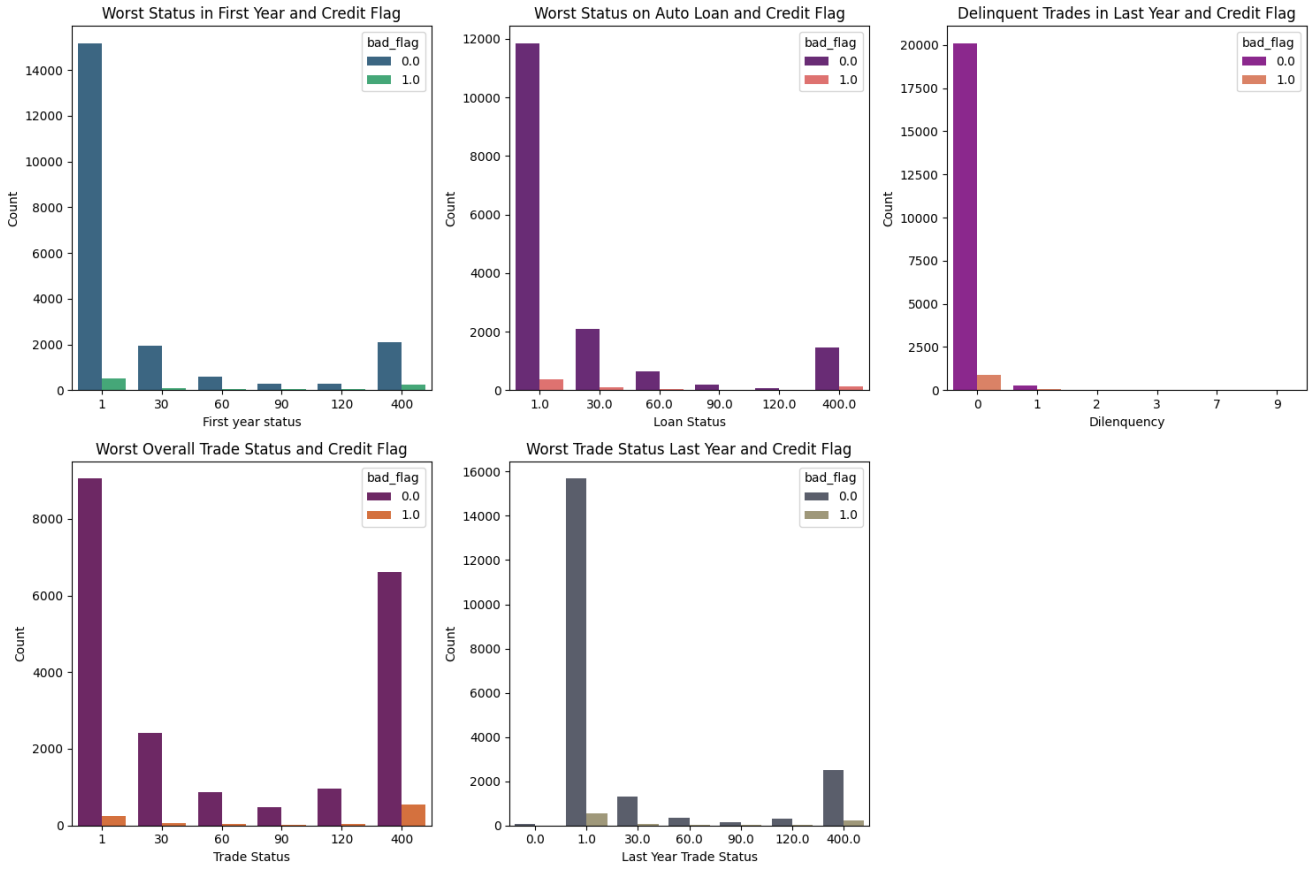

Figure 4: Bi-variate analysis of the frequency based features with the target variable

Insights: Strong Predictors of Poor Credit:
1. Higher worst statuses (400, 120, etc.) and an increasing number of delinquent trades are highly indicative of poor credit (bad_flag=1).
2. Good Credit Majority: Most applicants fall into the best status categories for all variables and are labeled as good credit (bad_flag=0).
3. Severity of Status: Variables that represent "worst status" (e.g., 'p12_all6250_a', 'p12_alm6200_a', 'p12_aua6200_a') have a clear gradient: as the status worsens, the likelihood of poor credit increases.

Similarly, we perform more visualization of other continuous features with the target variable[3]. The take away from those visualizations are:
1. Credit Utilization as a Risk Factor:
• Across various balance-to-credit (BTC) ratios, higher credit utilization rates (whether revolving, bankcard, auto loan, or single account) are associated with a higher likelihood of poor credit outcomes.
• This indicates that applicants who are closer to their credit limits may be at greater risk of financial stress or default.
2. Debt Ratios and Credit Risk:
• Higher PTI and LTV ratios correlate with increased credit risk, highlighting that applicants with higher debt burdens relative to their income and assets may be at greater risk.

[3]Please see the figure 13 for details

3. FICO Score as a Predictor:

• FICO score remains an essential indicator, with lower scores showing a strong association with poor credit flags, reinforcing its role as a primary predictor of creditworthiness.

3 Model

From the EDA, we have seen that our data is an imbalanced dataset. So, we may want to apply some resampling techniques. However, we want to explore the model performances and predictability without resampling first. Then we will also build our model on resampled data. We implement three machine learning algorithms namely *Logistic Regression, Decision Tree Classifier, and Random Forest Classifier* for our modeling purpose.

3.1 Overview of Models

Logistic Regression: A linear model that predicts probabilities using the logistic function. It is interpretable and efficient for binary classification.

Decision Tree Classifier: A rule-based model that splits data into branches, creating interpretable but potentially overfitted rules.

Random Forest Classifier: An ensemble method that combines decision trees to improve robustness and accuracy through bagging and feature randomness.

3.2 Implementation

We use 'scikit-learn' library for our modeling and evaluations. We import 'LogisticRegression', 'DecisionTreeClassifier', and 'RandomForestClassifier' models from 'sklearn'. Then we import 'StratifiedKFold' to preserve the percentage of samples in each class. We also import 'GridSearchCV' to find the optimal parameters for the tree-based models. We have the given exclusive test data. However, for better generalization we divide the given training data into X_train, X_test, y_train, and y_test using 'train_test_split'. In this split, we stratify the split by the y variable.

3.2.1 Modeling Without Resampling

Logistic Regression: We use the defaults for the parameters in the LogisticRegression except for max_iter which is set to 1000, and the class_weight='balanced' as our target class is not balanced.

Decision Tree and Random Forest: We set a set of parameters for the tree-based models to be used by grid search for fine tuning.

Parameter	Decision Tree (DT)	Random Forest (RF)
`max_depth`	[5, 10, 20]	[5, 10, 20]
`min_samples_split`	[2, 5, 10]	[2, 5, 10]
`min_samples_leaf`	[1, 5, 10]	[1, 5, 10]
`class_weight`	['balanced']	['balanced']
`n_estimators`	Not Applicable	[100, 200]

Table 1: Parameter Grids for Decision Tree and Random Forest Models

With grid search we obtain Logistic Regression mean ROC AUC score of 0.7978 with standard deviation 0.0123, Decision Tree with mean ROC AUC score of 0.7539 with standard deviation 0.0240, and Random Forest mean ROC AUC score of 0.8078 with standard deviation 0.0093. The optimal parameters are as follows:

```
DecisionTreeClassifier(
    random_state=42, max_depth=5,
    min_samples_split=10,
    min_samples_leaf=1,
    class_weight='balanced'
)
```

```
RandomForestClassifier(
    random_state=42, max_depth=20,
    min_samples_split=2,
    min_samples_leaf=10,
    n_estimators=200,
    class_weight='balanced'
)
```

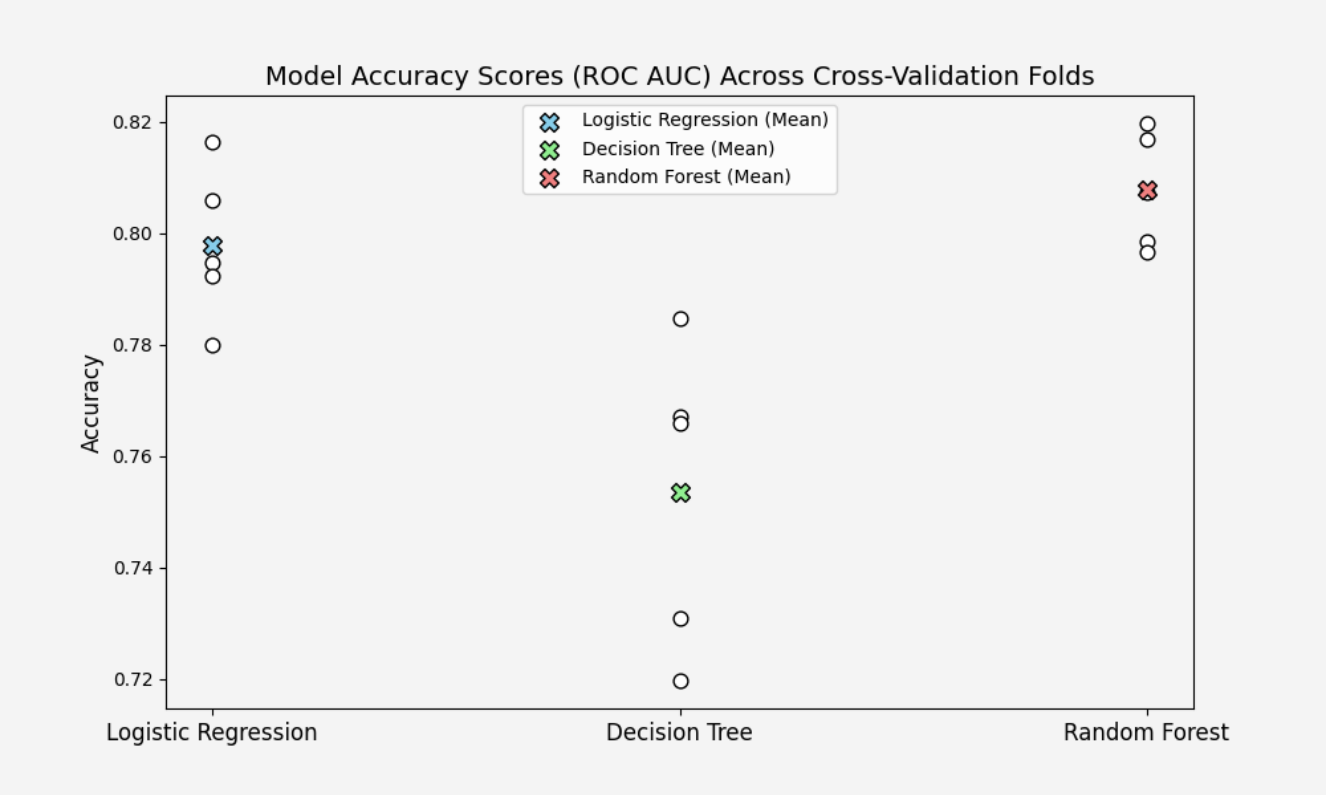

Figure 5: Model Accuracy Scores (ROC AUC) Acrross Cross-Validation Folds

Logistic Regression and Random Forest get us close mean ROC-AUC score but Logistic regression has higher standard deviation. Therefore, based on these results we choose our final model to be ***RandomForestClassifer***.

3.2.2 Model Performance

Training Data: Since for better generalization we split our training data to an additional train-test split, we obtain the following performance.

Training Data			Test Data		
Metric	**Class 0.0**	**Class 1.0**	**Metric**	**Class 0.0**	**Class 1.0**
Precision	0.998628	0.711281	Precision	0.962999	0.216931
Recall	0.981486	0.971279	Recall	0.963708	0.213542
F1-Score	0.989983	0.821192	F1-Score	0.963353	0.215223
Support	16312	766	Support	4078	192
Accuracy		0.981028	Accuracy		0.929977
Macro Avg		0.854954	Macro Avg		0.589965
Weighted Avg		0.985739	Weighted Avg		0.929452

Table 2: Classification Reports for Training and Test Data

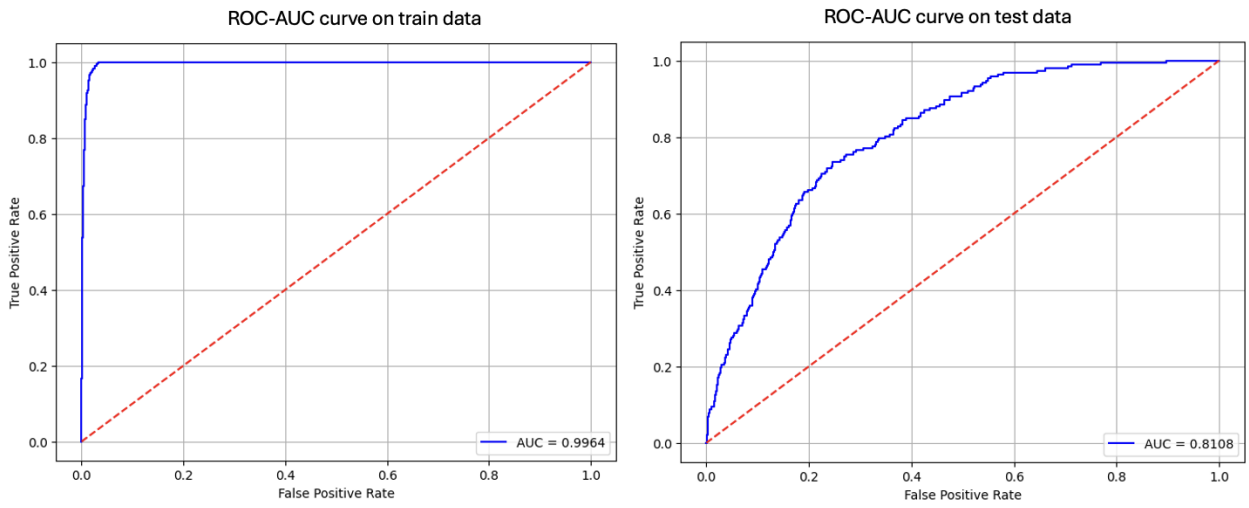

Figure 6: ROC-AUC curve for both training and test data from the original training data

Unseen Test Data: For the given exclusive test data, we obtain the following results

Metric	**Class 0.0**	**Class 1.0**	**Macro Avg**	**Weighted Avg**
Precision	0.961553	0.268750	0.615152	0.930133
Recall	0.977032	0.177686	0.577359	0.940780
F1-Score	0.969231	0.213930	0.591581	0.934976
Support	5094	242	5336	5336
Accuracy	0.94078			

Table 3: Classification Report for Unseen Test Data

The model correctly predicts 4977 as class 0 that were actually labeled as class 0 giving us a precision of 96.16% with an F1-Score of 0.97 indicates a good predictive model. Even though accuracy score is not a proper metric for this specific data, we can see that the accuracy score is also very good (94.08%) on this exclusive test data.

	Predicted 0	**Predicted 1**
Actual 0	4977	117
Actual 1	199	43

Table 4: Confusion Matrix for Unseen Test Data

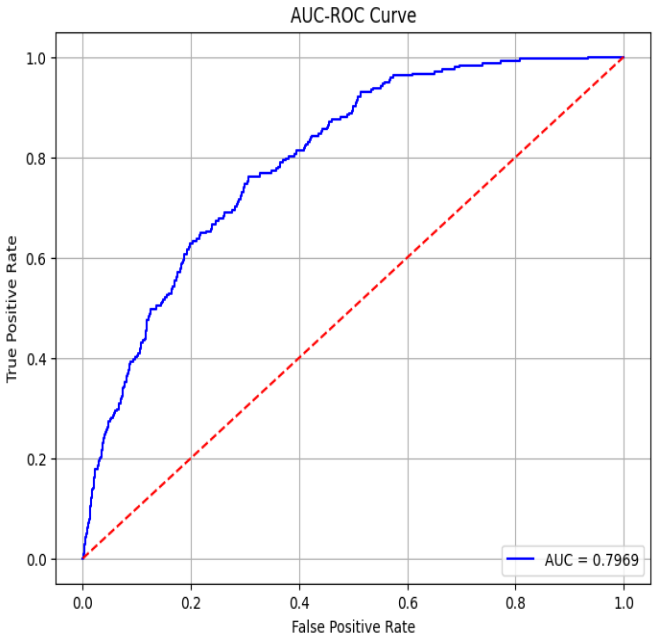

Figure 7: ROC-AUC curve on actual test data

3.2.3 Modeling With Resampling

From our exploratory data analysis, we have seen the imbalance class distribution in the target variable. So we apply the **Synthetic Minority Oversampling Technique (SMOTE)** resampling. After resampling, we apply the same modeling techniques with grid search for the optimized parameters for the tree-based models. We obtain the optimal model with resampling as follows

```
DecisionTreeClassifier(
    random_state=42, max_depth=20,
    min_samples_split=10,
    min_samples_leaf=2,
    class_weight='balanced'
)
```

```
RandomForestClassifier(
    random_state=42, max_depth=20,
    min_samples_split=2,
    min_samples_leaf=1,
    n_estimators=200,
    class_weight='balanced'
)
```

However, this resampling technique does not work very well for this dataset. It introduces overfitting during the training but does not perform well in the test data. The performance details are shown in the appendix.

4 Final Model and its Predictability

Based on all our analysis in the previous sections we proceed to build our final model with the **Random Forest Classifier**. To enhance code efficiency and maintainability, we implemented Object-Oriented

Programming (OOP) practices. Two classes were designed: one to encapsulate the model's functionality and another to explain its predictions using the LIME framework. We use **Local Interpretable Model-agnostic Explanations (LIME)** model to explain our model.

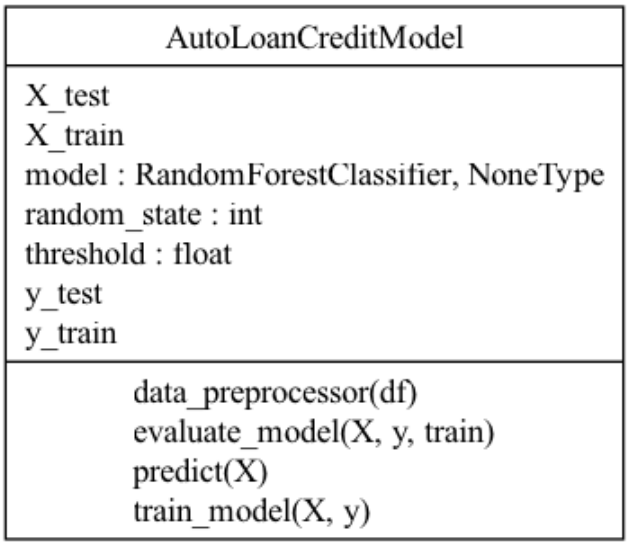

Figure 8: Model

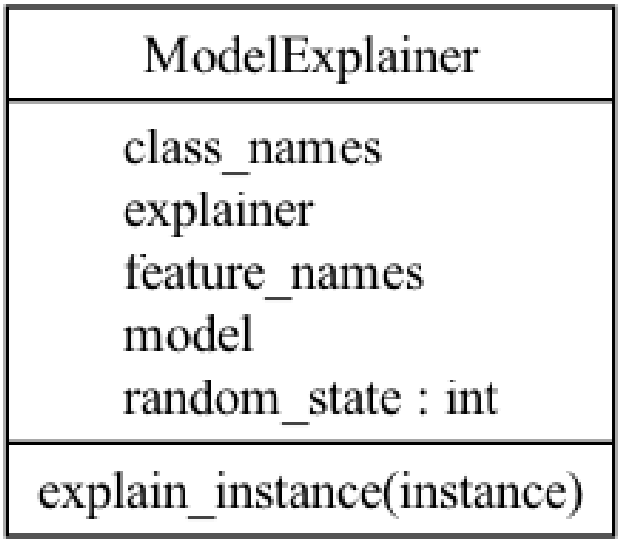

Figure 9: Model Explainer

4.1 Explain a Decision

Suppose our model successfully predicted an application with the probability of 0.10 to be class 0. It rejected the application and the customer asks for an explanation. Because they believes that it should get approved. Now we analyze the following result

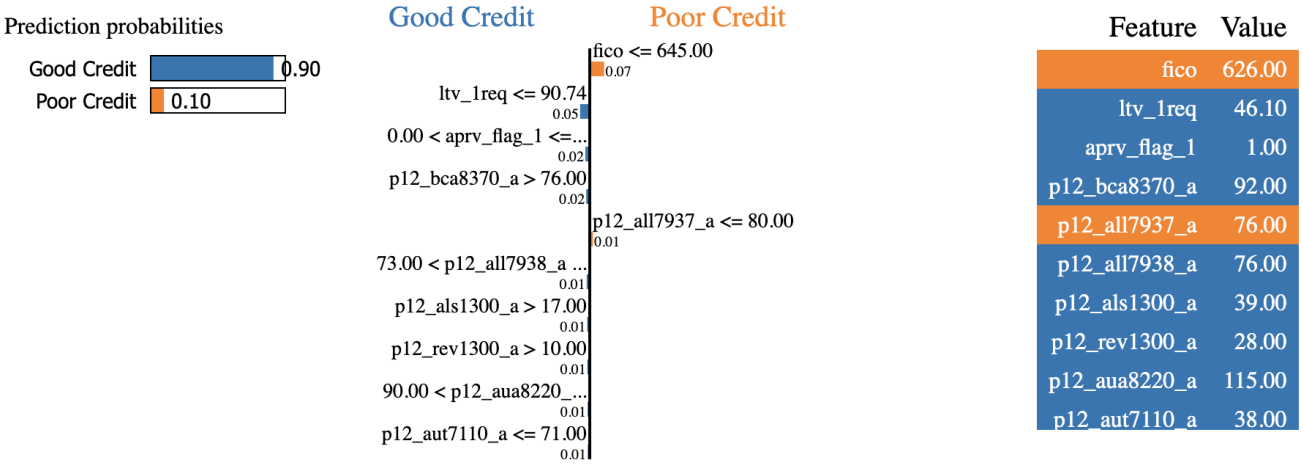

Figure 10: Model Explanation

The customer's fico score was 625 which is very low. Also, the percentage of trades reported in the last 12 months is less than 80. These factors triggered poor credit quality.

4.2 Gender Equality

Next we answer if our model maintained gender equality in approval rate. In the test data 2873 customers identified as Male, 2268 customers identified as Female, and 259 customers are unidentified. Our model provides gender neutral approval rates.

Gender	Approval Rate
Female	0.022046
Male	0.026801
Undefined	0.050193

Table 5: Approval Rates by Gender

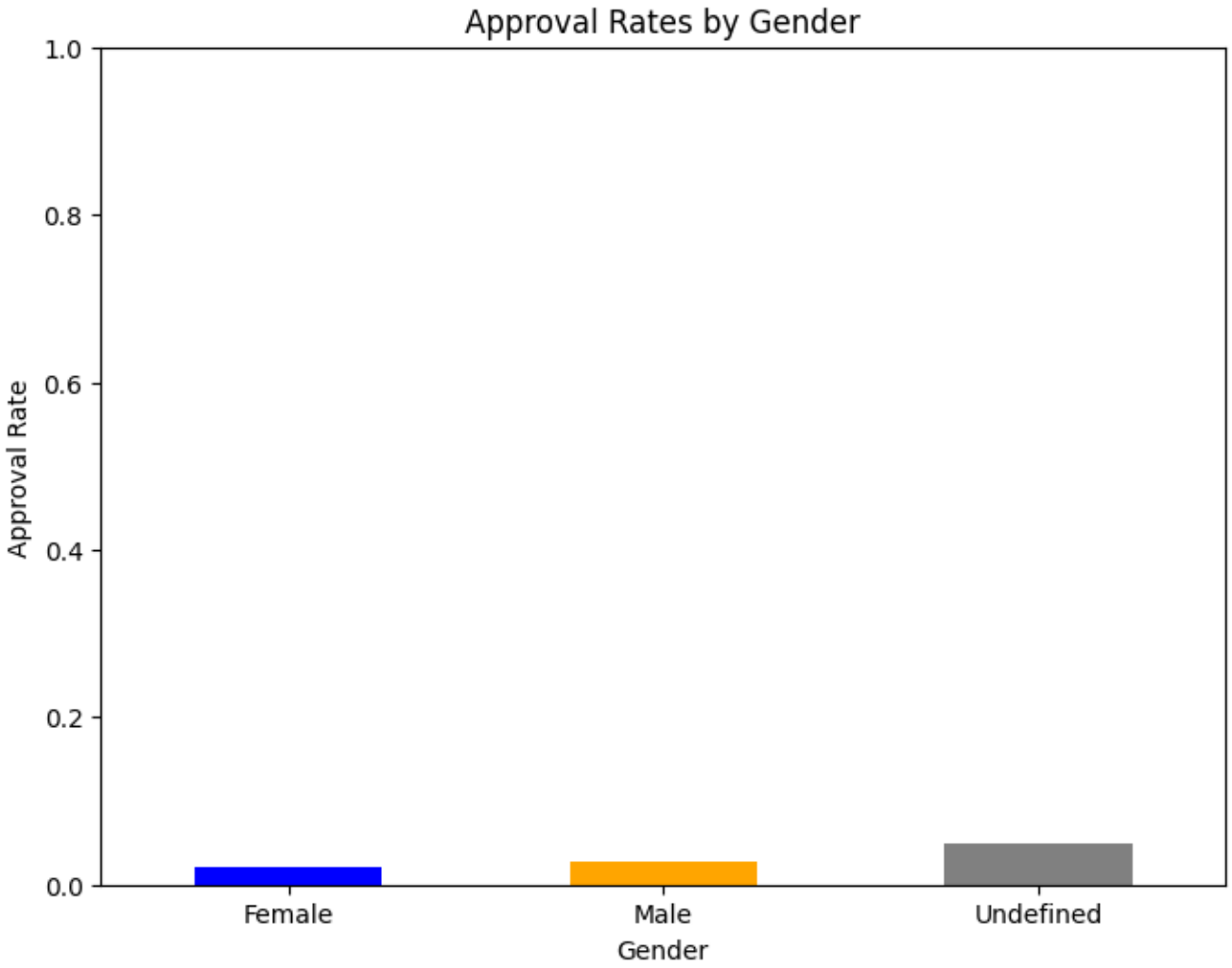

Figure 11: Model Approval rate by Gender

4.3 Racial Discrimination

Next, we answer if our model maintains racial equality. In the test data we have White 2932, Hispanic 1226, Black 730, Other 373, and Unknown 139. Our model approved customers irrespective of their racial background. Here is the approval rate for racial background.

Race	Approval Rate
Black	0.023288
Hispanic	0.026917
Other	0.029491
Unknown	0.021583
White	0.025921

Table 6: Approval Rates by Race

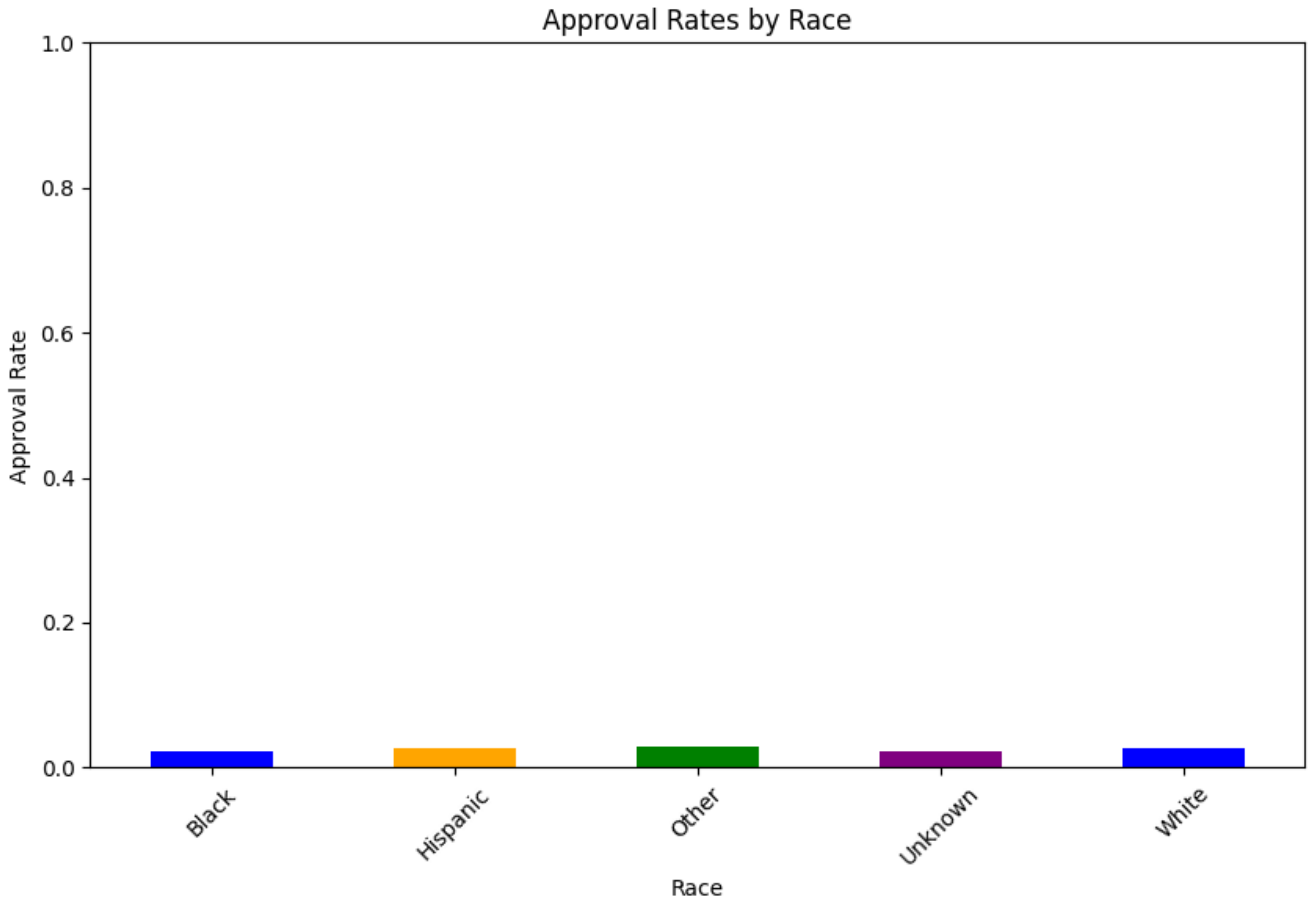

Figure 12: Approval rates across different race

5 Conclusion

The Random Forest Classifier demonstrated strong predictive performance with high accuracy and interpretability, effectively identifying applicants with poor credit quality. Despite its success, challenges such as class imbalance and overfitting with resampling were noted. Overall, the model provides a reliable foundation for auto loan decisioning, with opportunities for further enhancement through advanced techniques.

6 Further Investigation

While our model demonstrates strong performance, there are areas for potential improvement:

1. **Data Enhancements**: Refine imputation strategies for missing data and explore feature engineering to uncover hidden patterns in the dataset.
2. **Advanced Models**: Evaluate gradient boosting methods like XGBoost or LightGBM, which often perform well on structured data.
3. **Resampling Techniques**: Experiment with alternative methods like ADASYN to address class imbalance without overfitting.
4. **Explainability**: Extend the use of LIME or introduce SHAP to enhance interpretability for key stakeholders.

7 References

1. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.
2. Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1135–1144, 2016.
3. Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, 16, pp. 321–357, 2002.

A Exploratory Data Analysis (EDA) Details

	Missing in Training Data			Missing in Test Data	
Feature	**Values**	**Proportion (%)**	**Correlation**	**Values**	**Proportion (%)**
p12_all8150_a	12399	57.387	-0.036	3127	57.907
p12_aua8151_a	17814	82.449	-0.024	4471	82.796
clntr9437_a	12654	58.567	-0.117	3128	57.926
clact9429_a	12654	58.567	-0.117	3128	57.926
clall5010_a	19216	88.938	-0.066	4793	88.759
clall2434_a	20887	96.672	-0.044	5200	96.296
cloil0214_a	21528	99.639	-0.038	5382	99.667
cltra4405_a	20819	96.357	-0.036	5197	96.241
clact9428_a	12654	58.567	-0.117	3128	57.926

Table 7: Missing Values and Correlations with the Target Variable (`bad_flag`)

Feature	**Values**	**Feature**	**Values**	**Feature**	**Values**
fico	447	aprv_flag	2	amtfinanced_1req	16961
collateral_dlrinput_newused_1req	2	pti_1req	2124	ltv_1req	7877
p12_reh7120_a	170	p12_bcx7110_a	136	p12_all7170_a	102
p12_aut7110_a	110	p12_all7120_a	286	p12_all7937_a	97
p12_bcc8120_a	331	p12_iln7410_a	95	p12_rev1300_a	63
p12_bca8370_a	352	p12_all7517_a	87	p12_iln8220_a	379
p12_all6250_a	6	p12_rtr7110_a	132	p12_pil8120_a	255
p12_aua0300_a	37	p12_all7938_a	97	p12_bcc3456_a	19
p12_all8370_a	248	p12_aua8220_a	238	p12_als1300_a	83
p12_aua6200_a	7	p12_all2427_a	6	p12_alm6200_a	6
p12_all6971_a	8	bad_flag	3	Gender	3
Race	5				

Table 8: Number of unique values for each feature

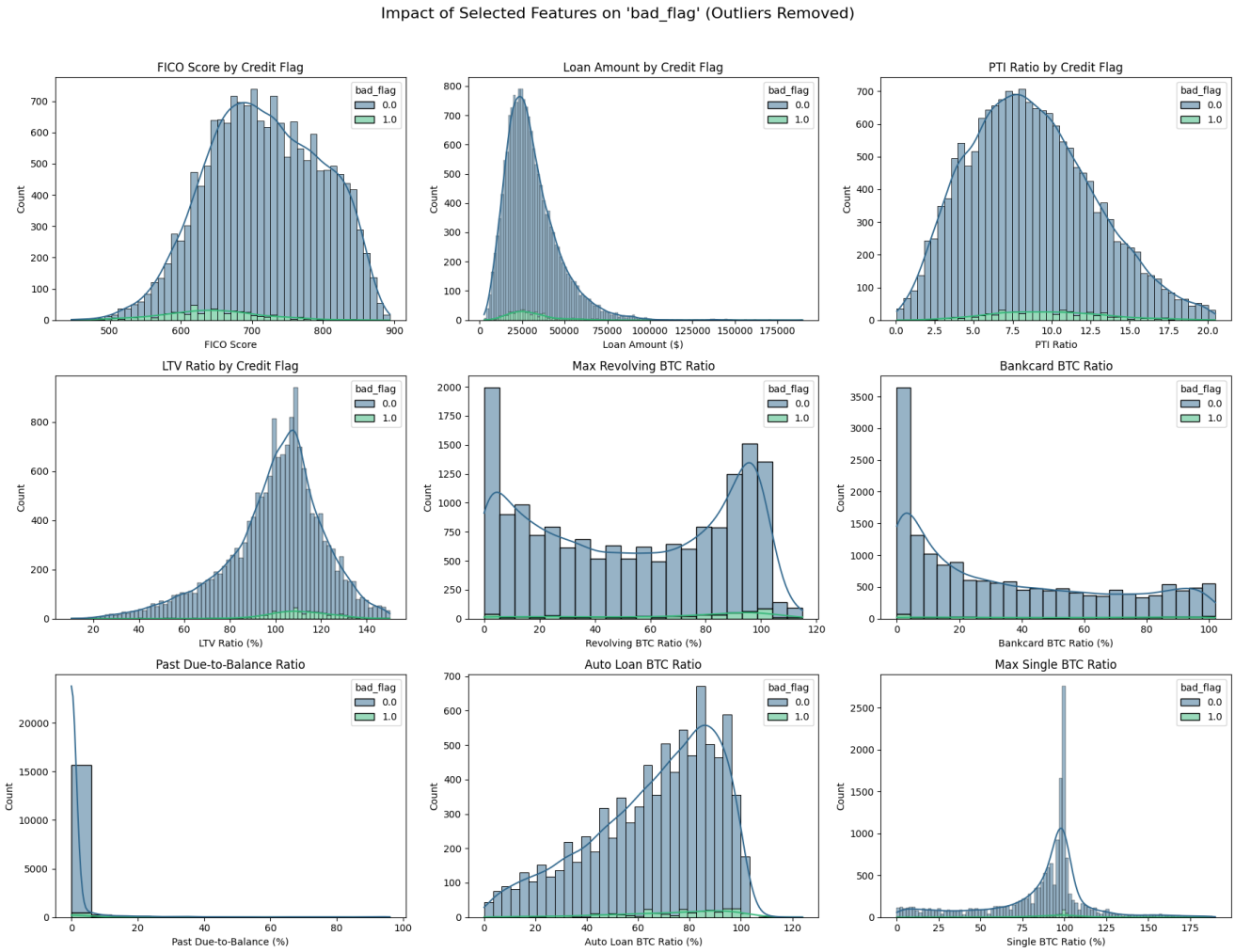

Figure 13: Bi-variate analysis of other continuous features

B Details of the resampling modeling

The following table contains the classification report from the resampling models. Note that this test data is actually the validation data from the original train data.

Training Data			**Test Data**		
Metric	**Class 0.0**	**Class 1.0**	**Metric**	**Class 0.0**	**Class 1.0**
Precision	1.000000	0.998286	Precision	0.964208	0.977369
Recall	0.998283	1.000000	Recall	0.977685	0.963708
F1-Score	0.999141	0.999142	F1-Score	0.970900	0.970490
Support	16312	16312	Support	4078	4078
Accuracy		0.999142	Accuracy		0.970696
Macro Avg		0.999143	Macro Avg		0.970788
Weighted Avg		0.999143	Weighted Avg		0.970788

Table 9: Classification Reports for Training and Test Data (with resampling)

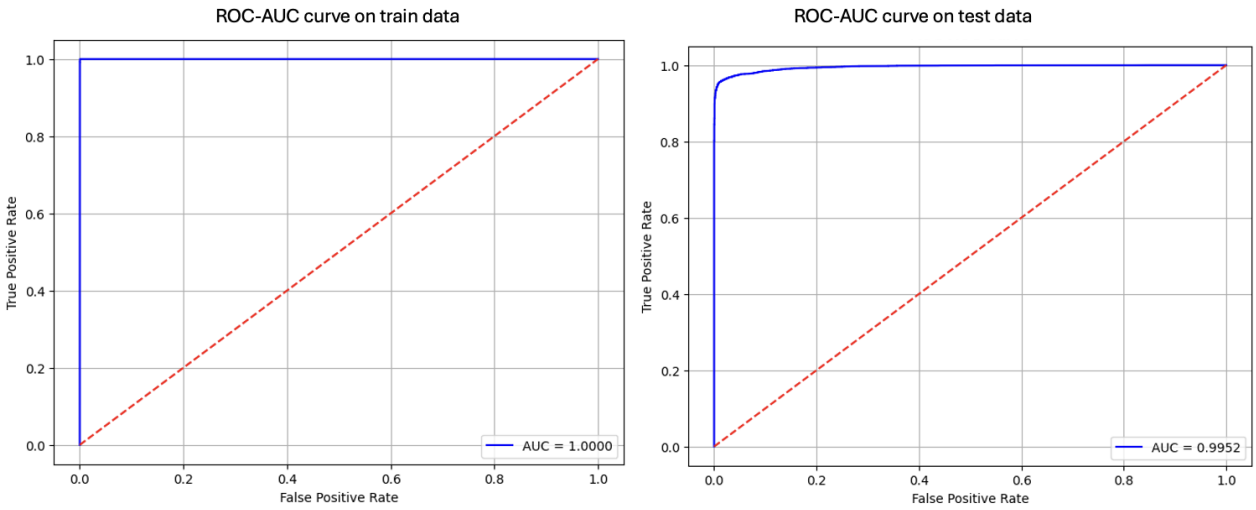

Figure 14: ROC-AUC curve for the train and test(validation) data from the resampling models

Table 9 and figure 15 clearly indicate overfitting. As a result, we get poor performance on the actual test data which is shown in table 10 and

Metric	**Class 0.0**	**Class 1.0**	**Macro Avg**	**Weighted Avg**
Precision	0.957676	0.159420	0.558548	0.921473
Recall	0.977228	0.090909	0.534069	0.937031
F1-Score	0.967353	0.115789	0.541571	0.928733
Support	5094	242	5336	5336
Accuracy	0.937031			

Table 10: Classification Report for the actual test data

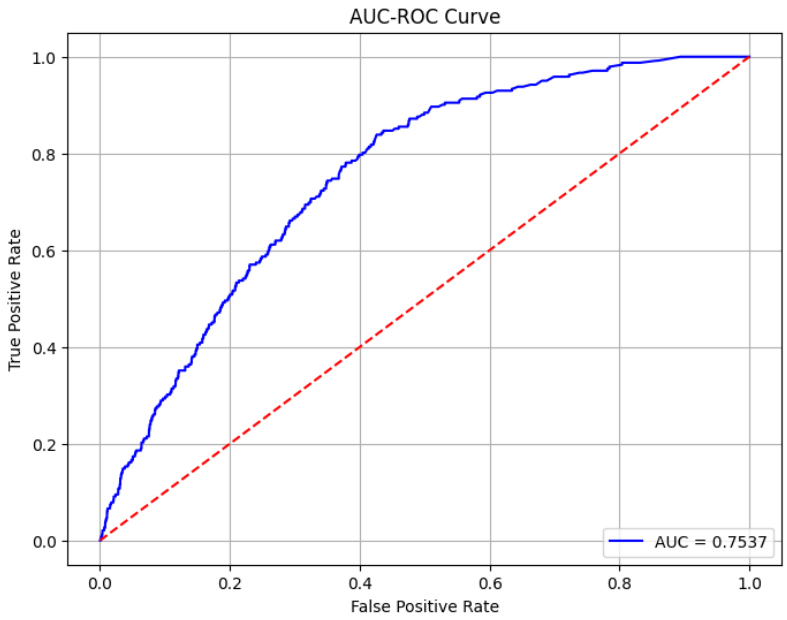

Figure 15: ROC-AUC curve for the actual test data