Home Loan Application Prediction

Aarya Parekh

Data Science Institute, Brown University

Github: https://github.com/aaryaparekh/data1030_project/tree/main

Introduction

The motivation behind this project is to build a model that can accurately predict if a home loan application for a given applicant should be approved or denied. Such a model can be used by banks to speed up the loan approval process.

The dataset was collected by Dream Home Financing as part of a public competition and later published on Kaggle. The currency in the dataset was obfuscated before publication. Our target variable is a yes or no depending on if the loan application is approved or denied. Therefore, we approach this problem as a binary classification problem.

Exploratory Data Analysis

A description of the dataset is as follows:

Feature Name	**Data Type**	**Rows Missing**	**Description**
Loan_ID	Continuous	None	Unique row identifier for dataset.
Gender	Categorical	13	Gender of applicant, male, female.
Married	Categorical	3	Marital status.
Dependents	Ordinal	15	Number of dependents.
Education	Ordinal	None	Highest education level.
Self_Employed	Categorical	32	If the applicant is self-employed or not.
ApplicantIncome	Continuous	None	Income of applicant, in unknown currency.
CoapplicantIncome	Continuous	None	Income of co applicant, in unknown currency.
LoanAmount	Continuous	22	Amount requested as loan, in unknown currency.

Feature Name	Data Type	Rows Missing	Description
Loan_Amount_Term	Ordinal	14	Term of loan, in months.
Credit_History	Categorical	50	If the applicant has credit history or not.
Property_Area	Categorical	None	Location of property, urban, rural, or semi-urban.
Loan_Status	Categorical	None	If the applicant's loan applicant was approved or denied.

There are 614 rows in the dataset, of which 134 unique rows have one or more columns missing.

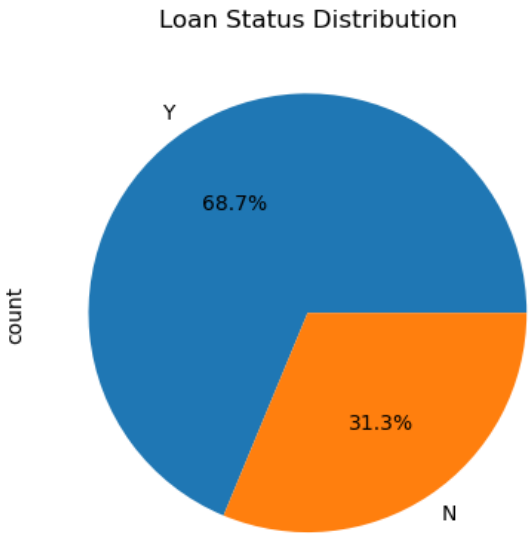

Figure 1: The distribution of the target variable, Loan_Status, shows the dataset is not egregiously imbalanced. 68.7% of the loans are approved while 31.3% are rejected.

The relatively balanced nature of the dataset indicates that we don't have to use a splitting strategy that balances the classes in each split. Additionally, the small amount of data (614 rows) means KFoldSplit strategy may be effective.

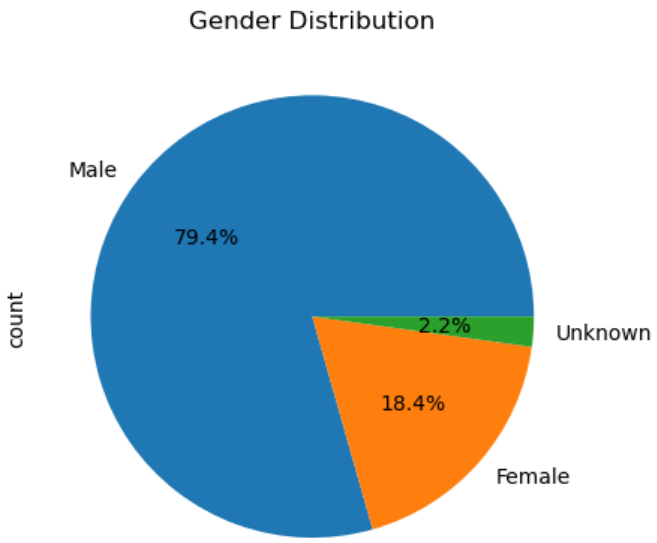

Figure 2: The gender feature is heavily skewed towards male applicants, with 79.4% of applications being submitted by males while 18.4% of applications being submitted by females.

The unequal representation of female applicants raises the question of biases that can be introduced in our models based on gender. This concern motivates our next figure.

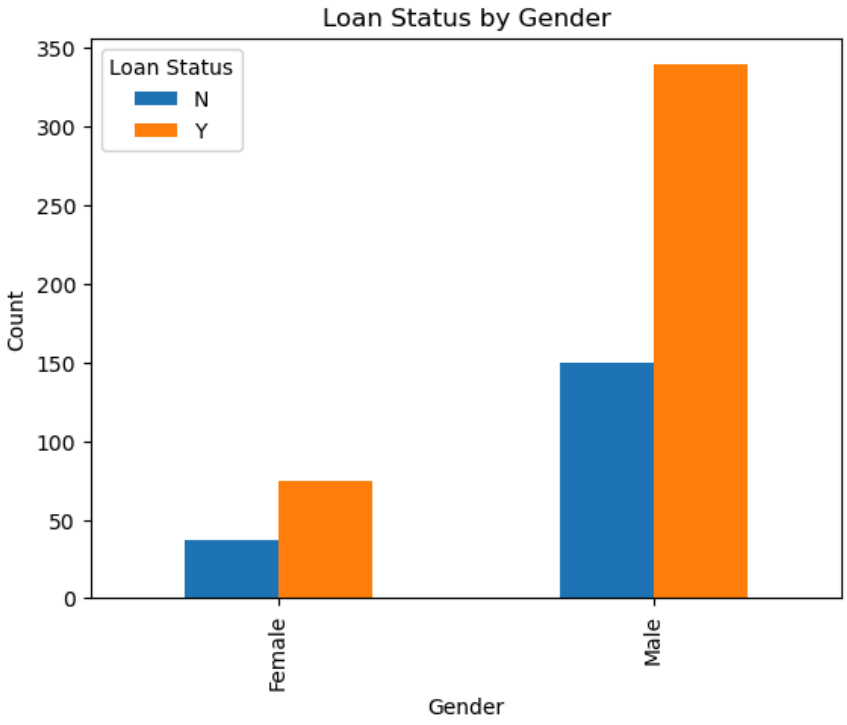

Figure 3: The graph shows the acceptance versus denial ratio stratified by gender. We can see that female applicants have a slightly higher rate of denial than males. This figure gives a quick (and limited) glimpse into biases that may be present in our data.

The above two figures raise the concern of gender based biases in our model. We will come back to this concern when interpreting our model in a later section of this report.

Finally, we look at the influence of income and loan amount, two features I believed would be most influential in determining an applicant's loan approval. Initially, I believed these two features alone would allow us to see two distinct clusters of rejected and approved applications.

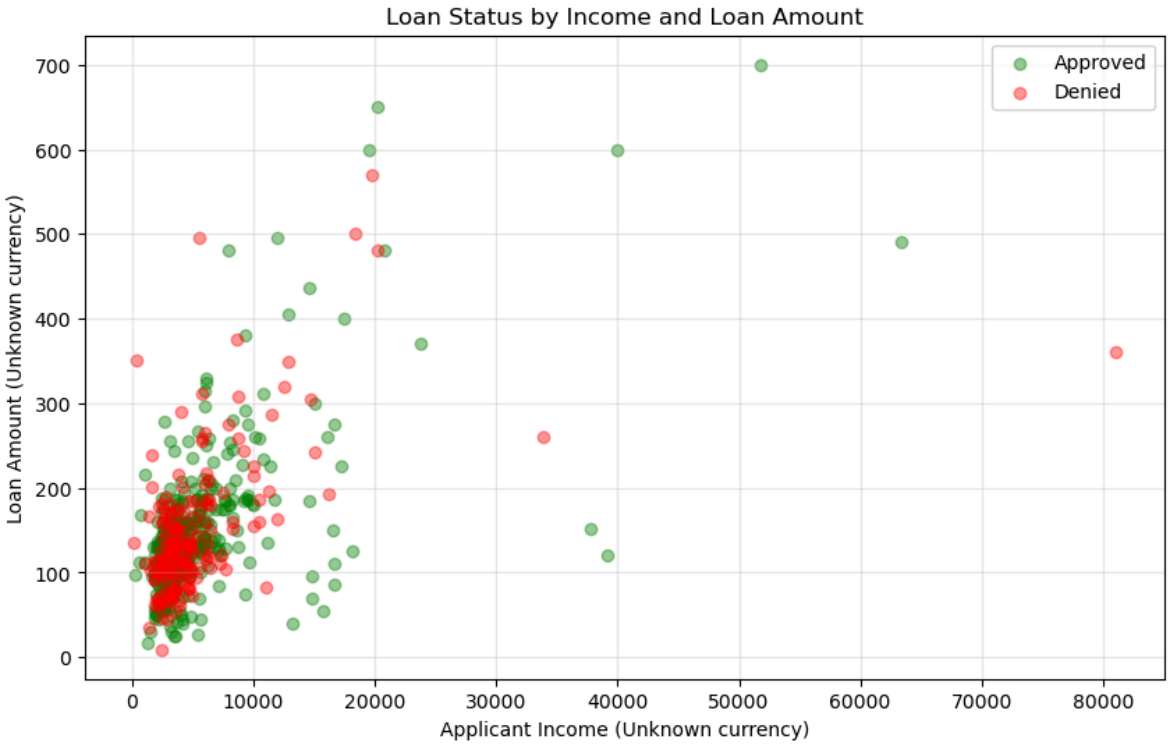

Figure 4: Applicant income vs loan amount, and the color of the point indicates if the application was approved or rejected. We can see the two clusters overlapping, making it hard to create an easy decision boundary based on these three features alone.

Contrary to my original hypothesis, loan amount and applicant income are not sufficient to determine the status of the loan application. The two clusters cannot be easily separated, and it appears that an applicant with a high income and relatively low loan amount also got rejected (right most point), making it clear that we need to use more features when training our models.

Methods

Now that we are finished exploring our data, we can develop a strategy to deal with the missing values. All features with missing values, except loan amount, are categorical or ordinal. Therefore, we add a new "missing" category to categorical and ordinal features with missing values. We drop the rows with a missing loan amount as this comprises only 22 rows, which represents a small portion of the dataset.

Next, we choose a splitting strategy. The data is IID, as all applications contain unique applicants. Additionally, no time series data is present. The data contains 592 points after dropping rows with missing loan amounts, and does not contain an egregious imbalance in the target feature, as we saw in our exploratory data analysis. Therefore, we use a basic splitting strategy of 60% training, 20% validation, and 20% holdout.

We need to encode our categorical and ordinal features, and scale our continuous features. We apply a One Hot Encoder to all categorical features, an Ordinal Encoder for ordinal features and a standard scaler for continuous data. We use SKLearn's Pipeline to do this easily.

To minimize variance due to random states in our models, we perform the end-to-end training and evaluation, from splitting and preprocessing to training and validation, on five different integers that we use as our random state. We pick our test set before the five random states get involved so it remains constant for all models and random states.

We train four models: Logistic regression, random forest classifier, support vector classifier (SVC), and XGBClassifier. Each model is trained using GridSearchCV which uses KFold to automatically tune hyperparameters, a good choice considering our dataset is small. We pick the best model by average each model's performance on the validation set across all random states. We also calculate the baseline score by predicting the dominant class for each datapoint, in our case we predict that the loan is approved for all points. Finally, we calculate the score for the baseline and best model on the test to get model performance on data the model has never seen before.

A company giving out home loans will want to minimize their risk. Giving a home loan to an individual who is not capable of paying it back leads to a loss for the company. Therefore, we want to minimize false positives. To accomplish this, we choose precision as our evaluation metric. Precision measures the fraction of all correct positive predictions out of all positive predictions by the model. This allows us to minimize false positives in our model, therefore minimizing the company's risk profile and losses.

In summary, the training script workflow is as follows:

1. Choose a test set (20% of data).

2. For each random state:
    a. Split data into training (60%) and validation (20%) .
    b. Apply preprocessing pipeline to encode and scale features.
    c. Calculate baseline score (predict all data points as loan approved).
    d. Train logistic regression, random forest classifier, SVC, and XGBClassifier.
3. Average validation scores for each model across all random states to pick the best model.
4. Calculate test scores for baseline and best model.

GridSearchCV's hyperparameter ranges were calculated by trying each model with a specific hyperparameter value manually, then graphing the validation and test scores to make sure the range of hyperparameter values included underfitting and overfitting.

Summary of hyperparameters trained for each model and their respective values for the best models of each type:

Logistic regression:

- C: 10
- Class_weight: Balanced
- Max_iter: 3000
- Penalty: L2
- Solver: Saga

Random Forest Classifier:

- Max_depth: 20

SVC:

- C: 0.1
- Gamma: Scale
- Kernel: Linear

XGBClassifier:

- Max_depth: 6

Results

Summary of averaged results across all random states for each model:

Model	**Accuracy**	**Precision**	**Recall**	**F1**

Baseline	0.65	0.686	1.00	0.79
Logistic Regression	0.73	0.796	0.83	0.82
Random Forest Classifier	0.77	0.787	0.94	0.84
SVC	0.83	0.782	0.99	0.89
XGBClassifier	0.81	0.781	0.88	0.87

Since the metric we are optimizing for is precision, logistic regression performs the best.

We now look at global feature importance to gain insight into which features influence the model's outputs.

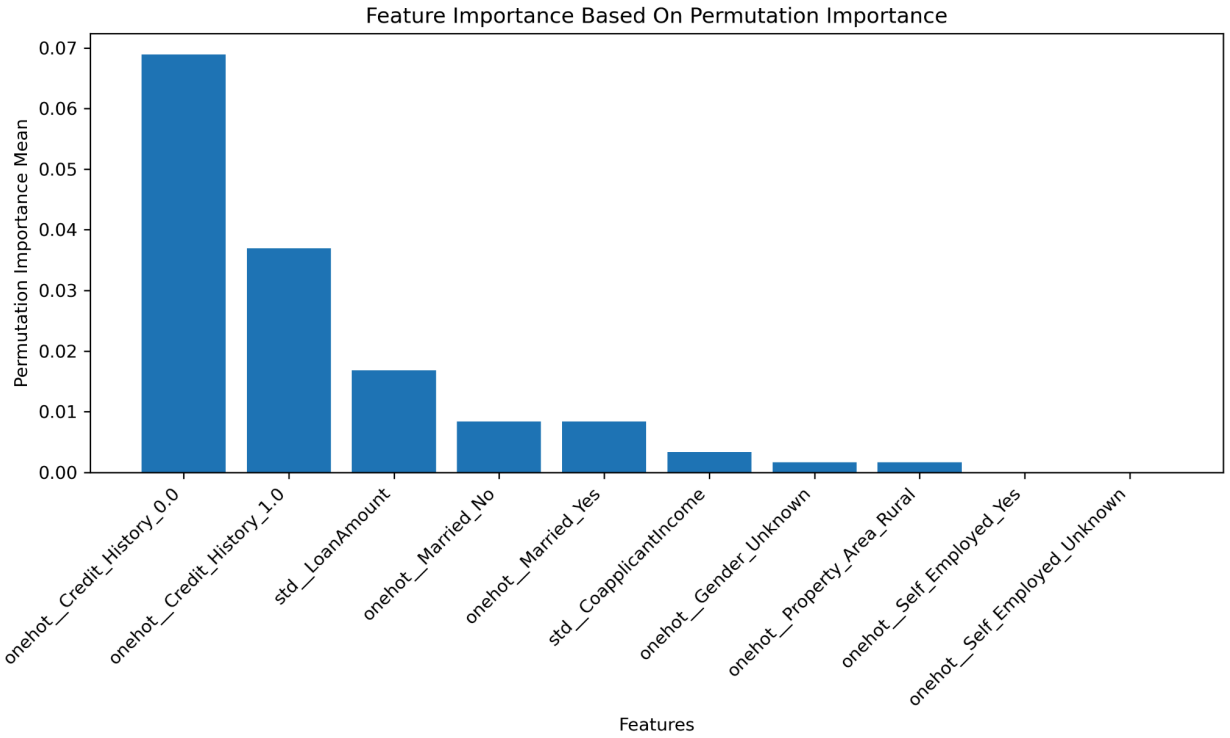

Figure 5: This graph shows the importance of each feature based on permutation importance. We can see that credit history is the most important feature, followed by loan amount and marital status.

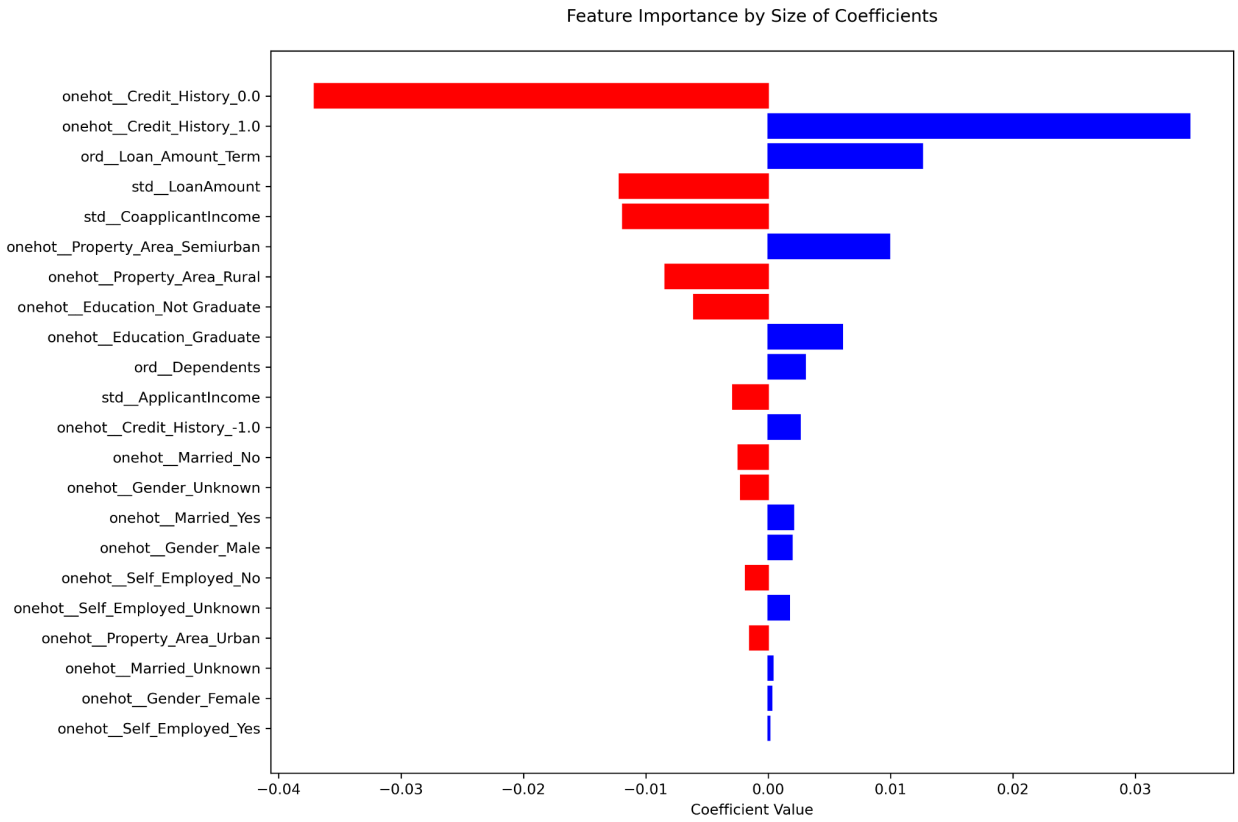

Figure 6: This graph shows feature importance based on size of the coefficients in the weight matrix. We can see that credit history is the most important feature, followed by loan amount term, loan amount, and co-applicant income.

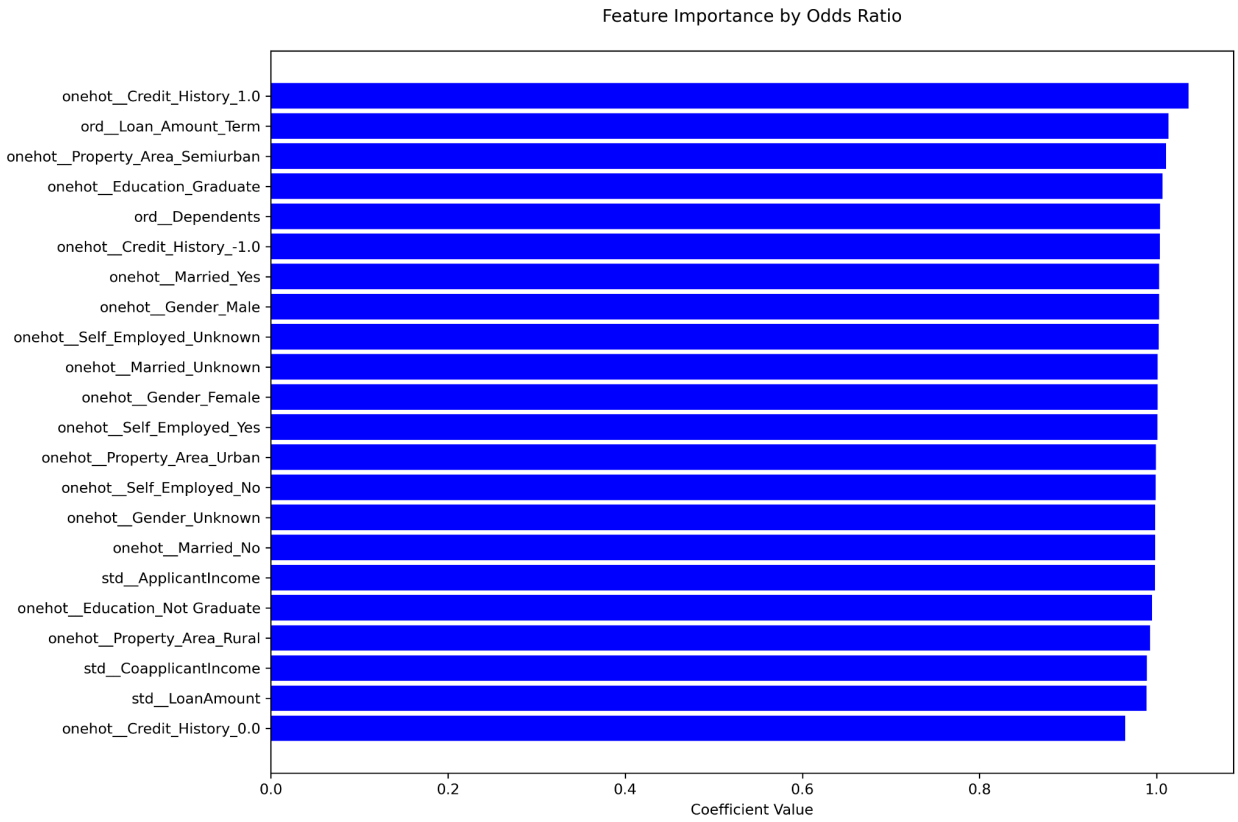

Figure 7: This graph shows feature importance based on odds ratio. An odds ratio represents the odds that an outcome will occur given a particular condition, compared to the odds of the outcome occurring without that condition. Here we see that credit history being present, loan amount term, and property area being suburban are the most important features to have a loan application be approved. The lack of credit history hurts an applicant's chances the most.

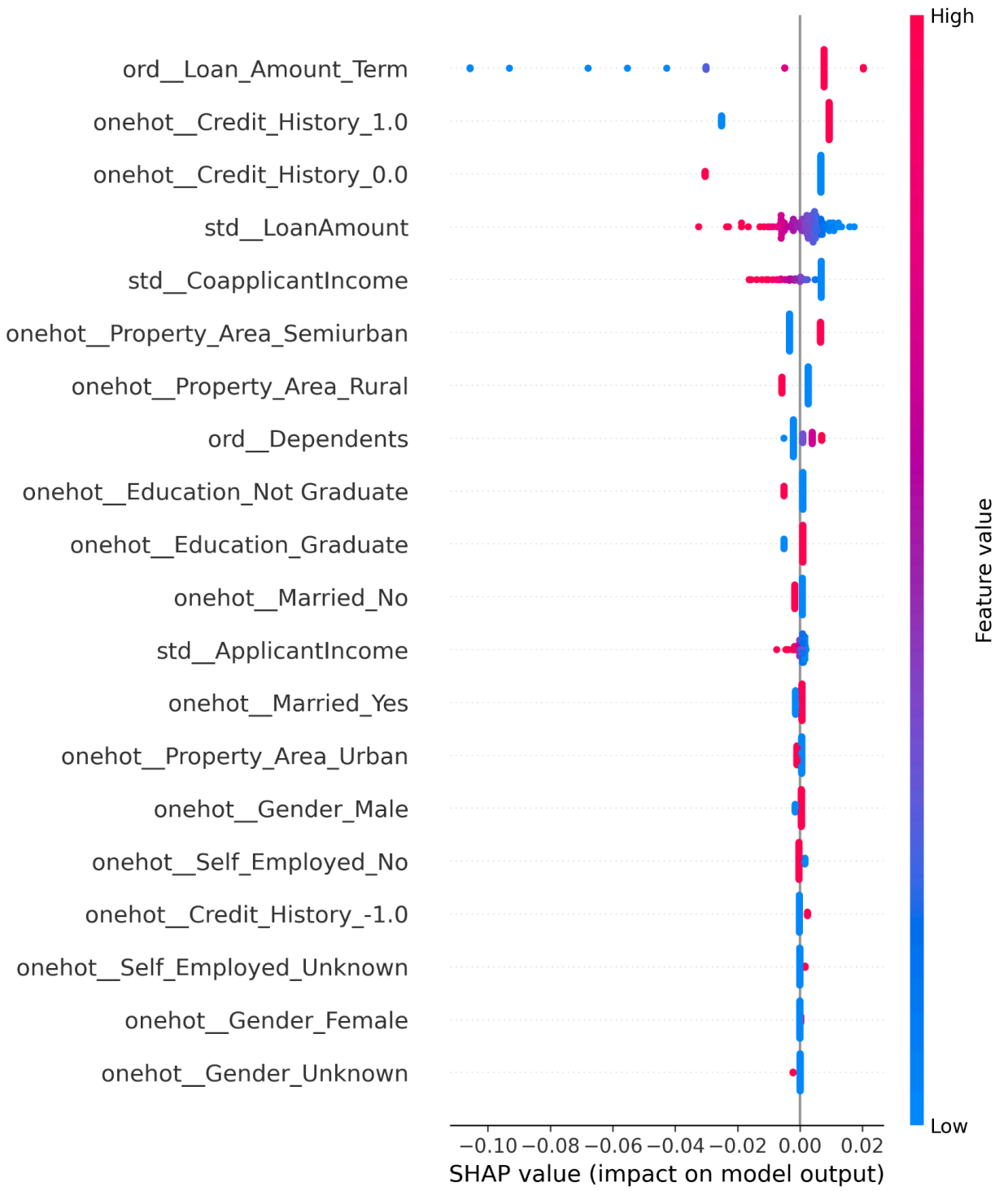

Figure 8: This figure shows the local feature importance as determined by shape values. We see that loan amount term, credit history, and loan amount are very important factors and have a high impact on the model.

The above global feature importance gives insight into the model's behavior. All graphs show that credit history is the most influential factor in an applicant's loan application. The odds ratio gives us more insight into this by showing that the lack of a credit history (denoted by a 0) hurt the applicants chances of approval more than any other feature. The SHAP values also show that the lack of a credit history hurts an applicant's chances of approval more than the presence of a credit history helps. Both, global importance and local importance, corroborate the need for a credit history.

We can also see that loan amount term, loan amount, and applicant income are all important factors when determining an applicant's loan status based on global and local importance. In our EDA, we concluded that loan amount and applicant income were not sufficient to determine a loan approval or rejection, but we now know that they are important factors nevertheless.

Our EDA also made us weary about the lack of female applicants in our data relative to male applicants. This was a cause for concern. However, the global feature importance metrics show that gender is not a very influential factor in our model. The local importance SHAP values also say the same thing. Thus our worries are laid to rest.

Outlook

The biggest weakness with this project is the relatively small dataset. Collecting more data, as well as increasing the number of features, will greatly improve our model's performance. Factors like age of applicant, number of houses, total net worth, history of loans, credit history, and more can give more insight.

Additionally, I would like to train more versions of the models by trying more hyperparameters, like min_child_weight, colsample_bytree, and subsample in XGBClassifier. I ultimately had to refrain from tuning these hyperparameters due to limitations with my computer.

Work Cited

Dataset: https://www.kaggle.com/datasets/rishikeshkonapure/home-loan-approval