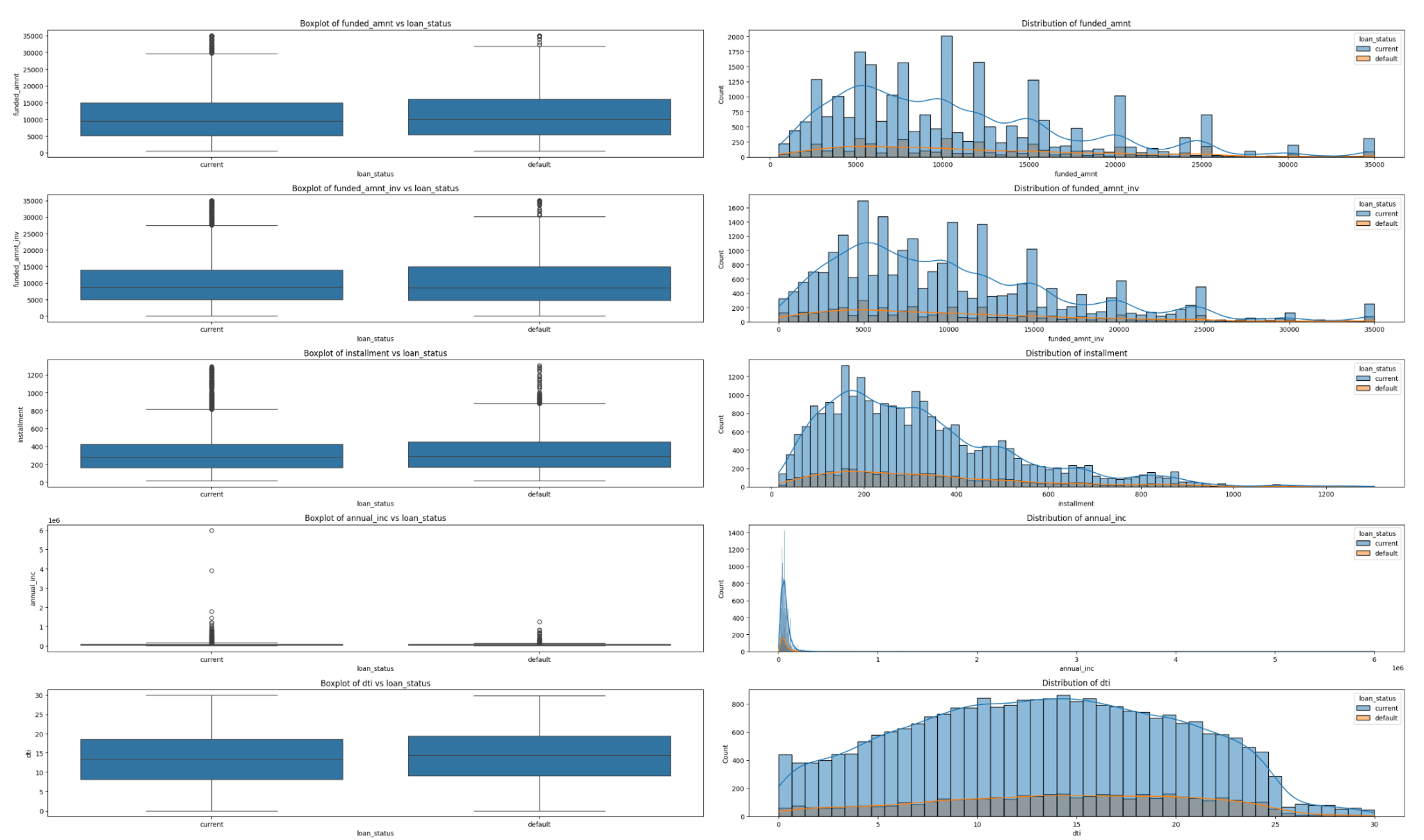
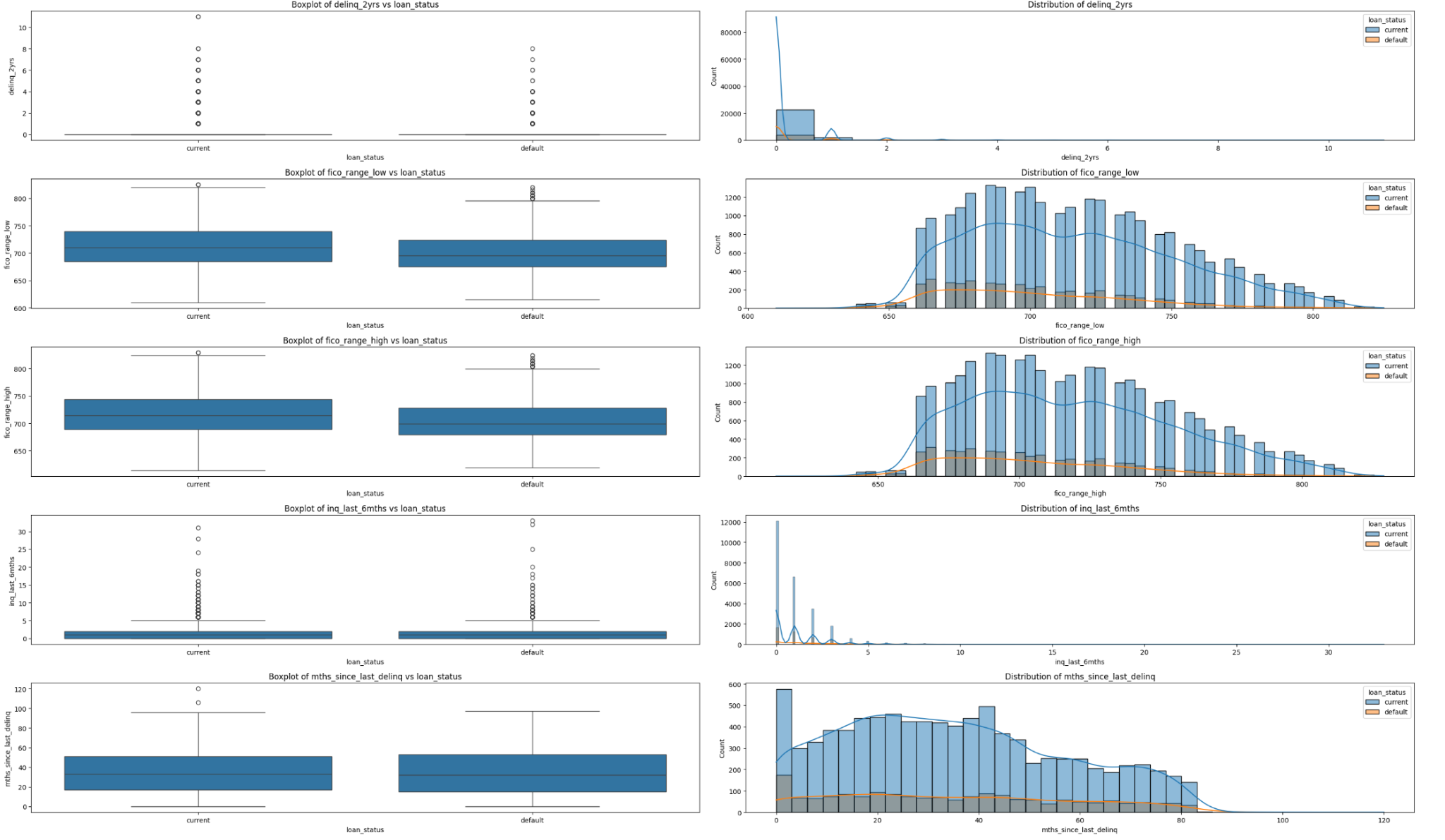
# Model Report

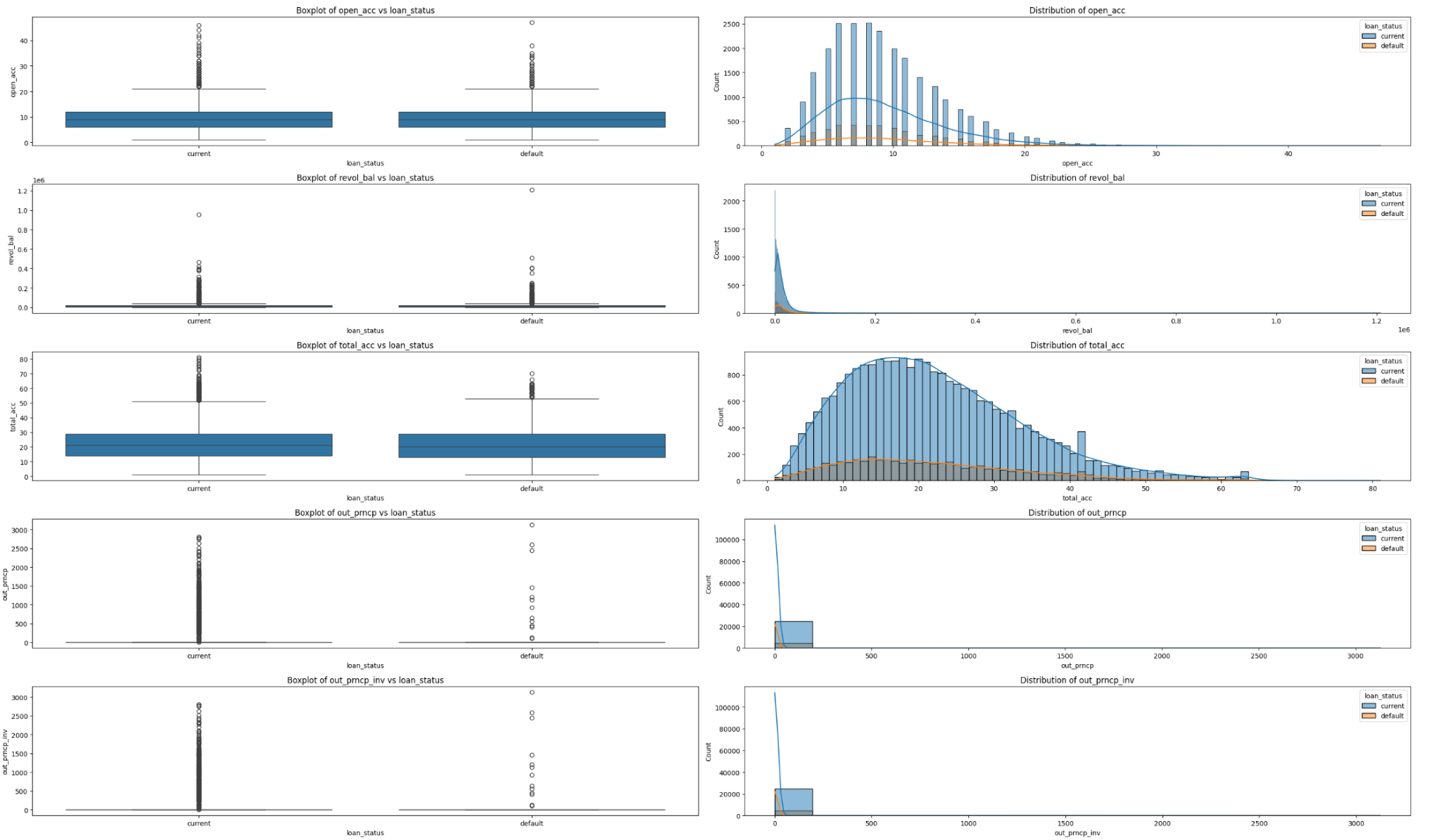
## I. Data Exploration and Preprocessing

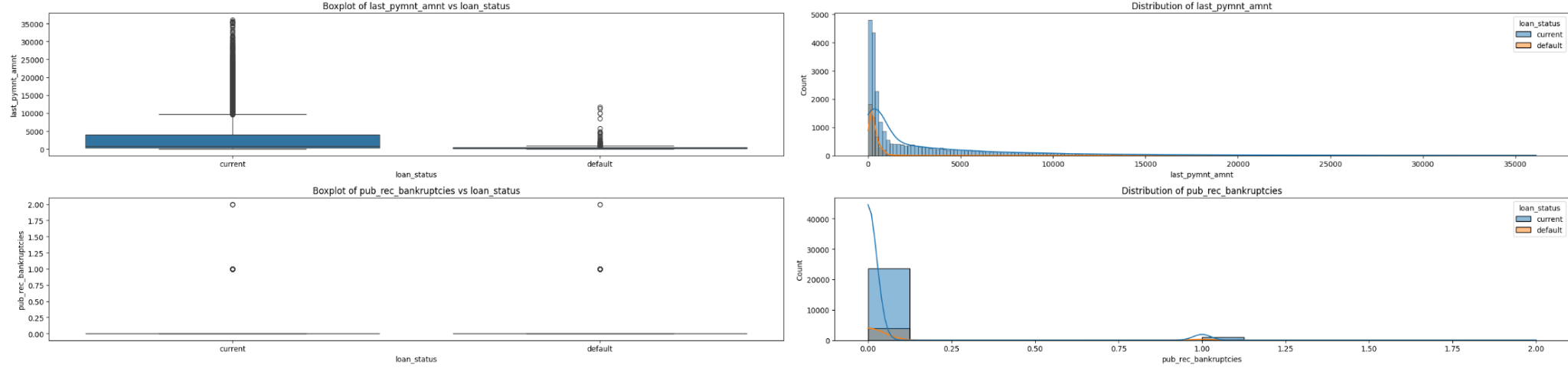
**Exploratory Data Analysis (EDA) & FEATURE SCREENING**: Conduct an initial analysis to understand the data's characteristics, including distribution of the target variable, missing values, and potential outliers. PUT SOME CHARTS AND TABLES in this section!

1. Numerical Features







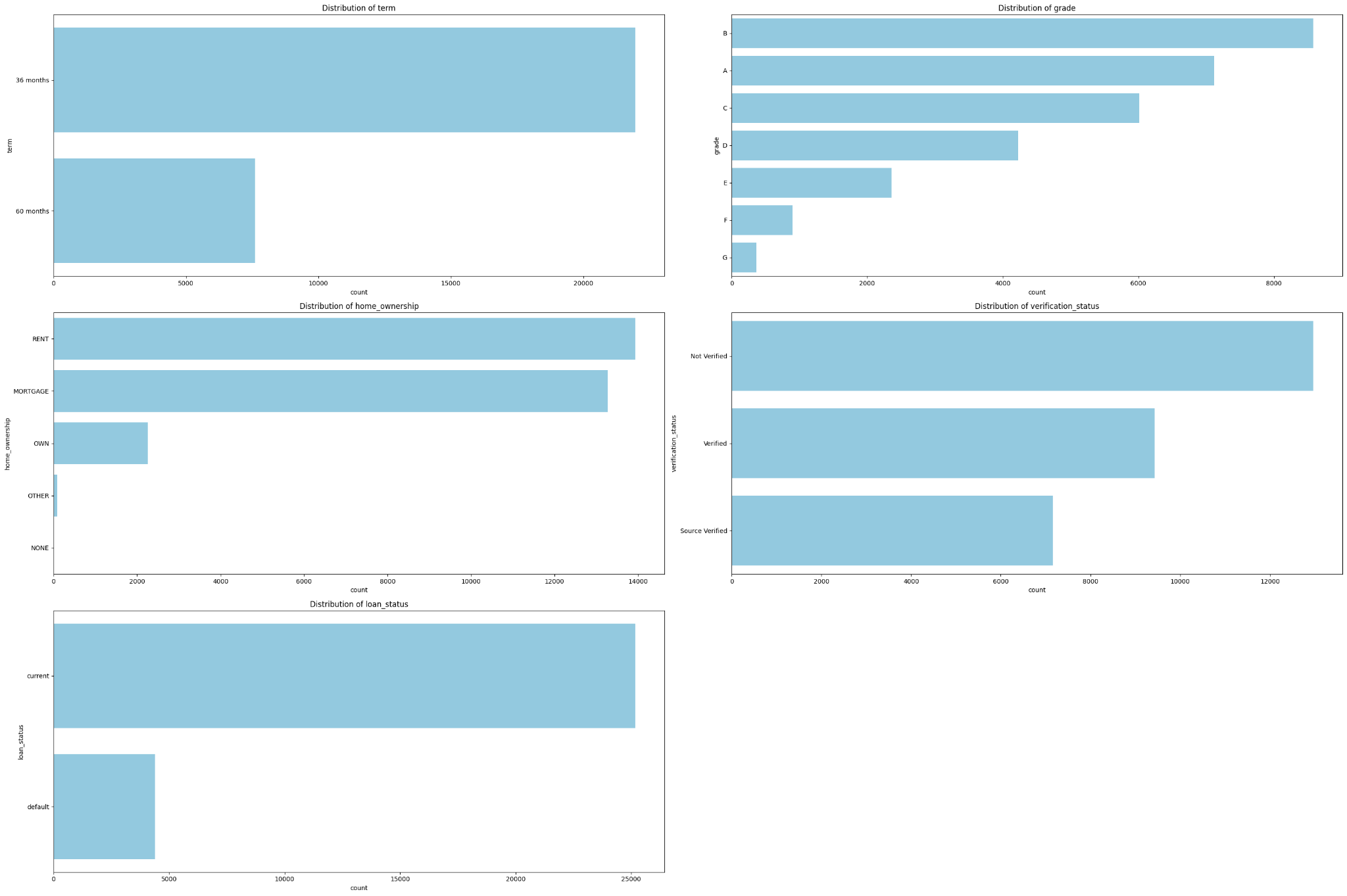


While most numerical features are heavily skewed, there are noticeable differences between current and default transactions. Specifically:

* Last Payment Amount: Default transactions typically have smaller last payment amounts.
* Funded Amount: The median funded amount for default loans is slightly higher than for current transactions.

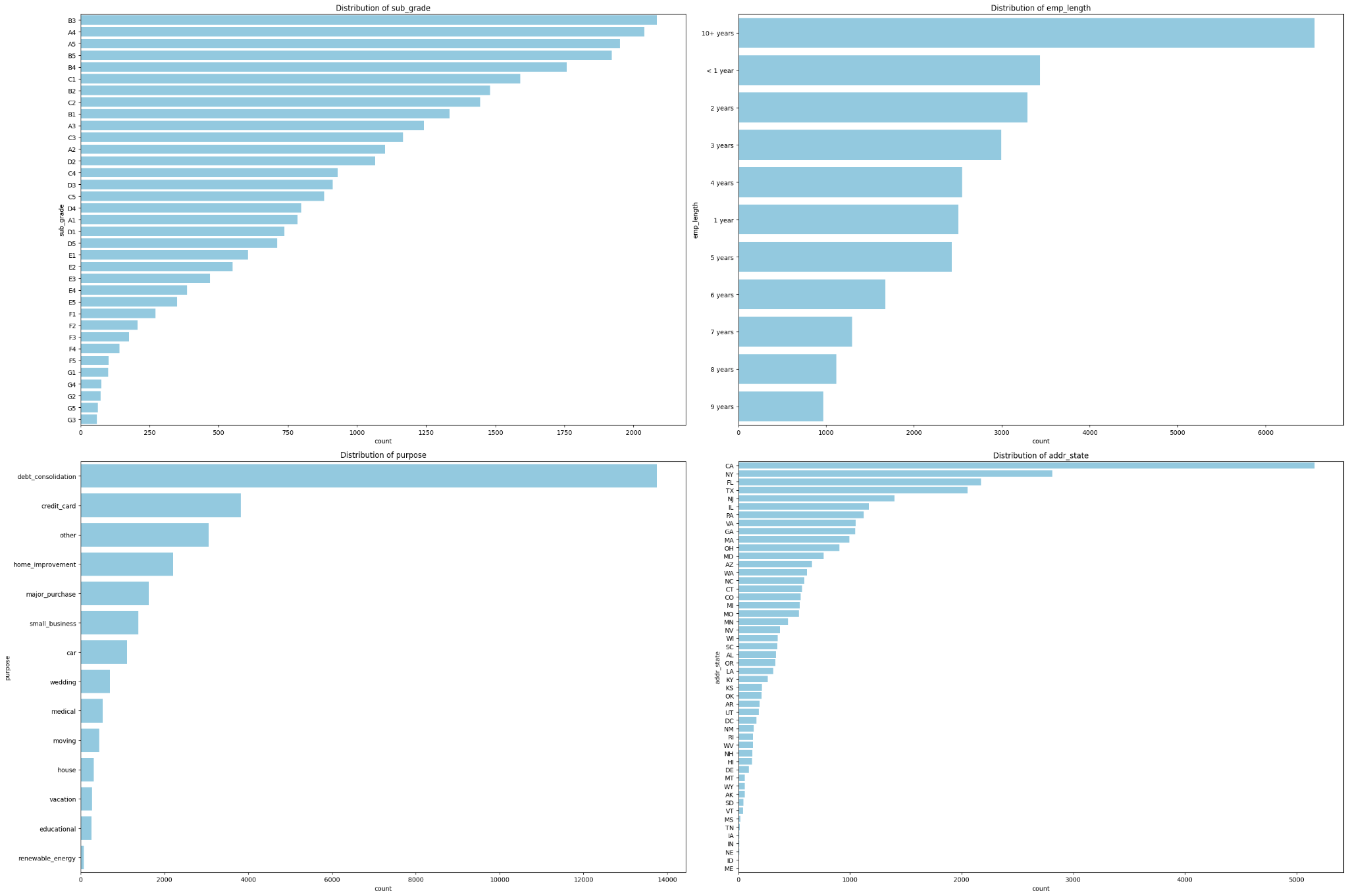
1. Categorical Features

*Low Cardinality (Less than 10 unique values)*



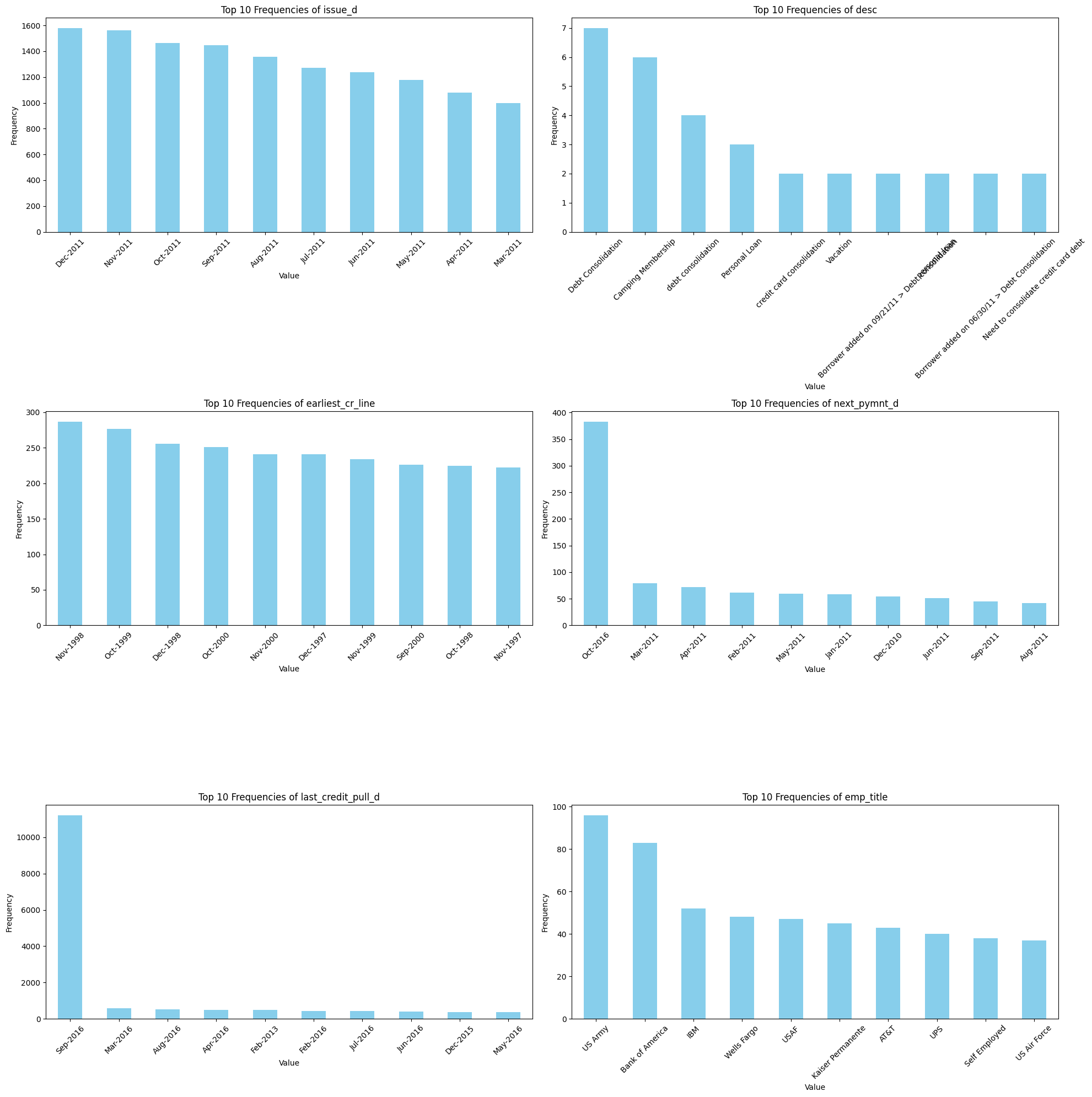
* The most common loan term is 36 months, and the most frequent loan grades are B, A, C, and D.
* Most borrowers are either renting a house or paying a mortgage.
* The target variable has a moderately low percentage of defaults.

*Medium Cardinality (Less than 50 unique values)*

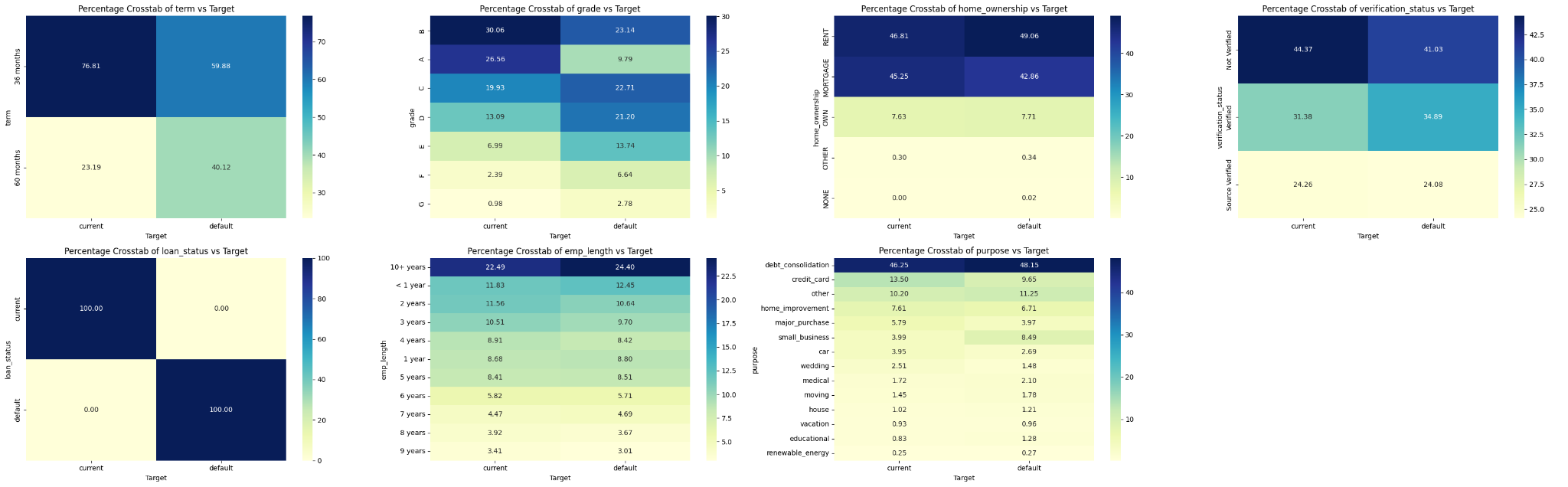


* A significant portion of our borrowers have been employed for more than a decade.
* Over half of our loans are used for debt consolidation purposes.
* Approximately 40% of our total loan volume originates from California, New York, Florida, and Texas.

*Large Cardinality (More than 50 unique values): Due to the high level of this category, we only show the most frequent level in each feature.*



* From the last credit pull and date and the next payment date features, we can assume that the current date for our dataset is September 2016.



* Default loans are commonly associated with longer terms (60 months vs 36 months) compared to non-default loans, with 40% of default loans having a longer term, in contrast to 23% of non-default loans.
* Among default loans, approximately 25% belong to grades G, F, and E, while only 10% of current loans are categorized within those grades.

**Data Preprocessing**: Address missing values, encode categorical variables, and standardize numerical features to prepare the data for modeling. PUT A TABLE of DATA TRANSFORMATIONS in this section.

Missing Values: Our approach to missing values is tailored to the specific feature

* Simple Replacement: For columns with less than 10% missing data, we replace missing values with the median (for numerical data) or the mode (for categorical data).
* Feature Engineering: When missing values are more significant, we use feature engineering to create a new category representing the missing data. This preserves information without harming the analysis.
* Removing Useless Features: Columns with extremely high levels of missing data or that provide no valuable information are removed entirely.

Encoding Categorical Features: We use a two-pronged approach to handle categorical features

* Feature Engineering: For certain categorical features, we apply feature engineering techniques to reduce dimensionality. This makes the data more manageable for our models.
* One-Hot Encoding: Remaining categorical features are one-hot encoded within the model's pipeline. This converts them into a numerical format that machine learning algorithms can understand.

Outliers: While our data shows some skewness, we're employing robust models like XGBoost and Random Forest, which are less sensitive to outliers. Therefore, extensive outlier treatment isn't currently a top priority.

Standardization: To potentially improve algorithm performance, we standardize most numerical features within our data pipeline. This ensures features are on a similar scale, enhancing model training.

Feature Engineering:

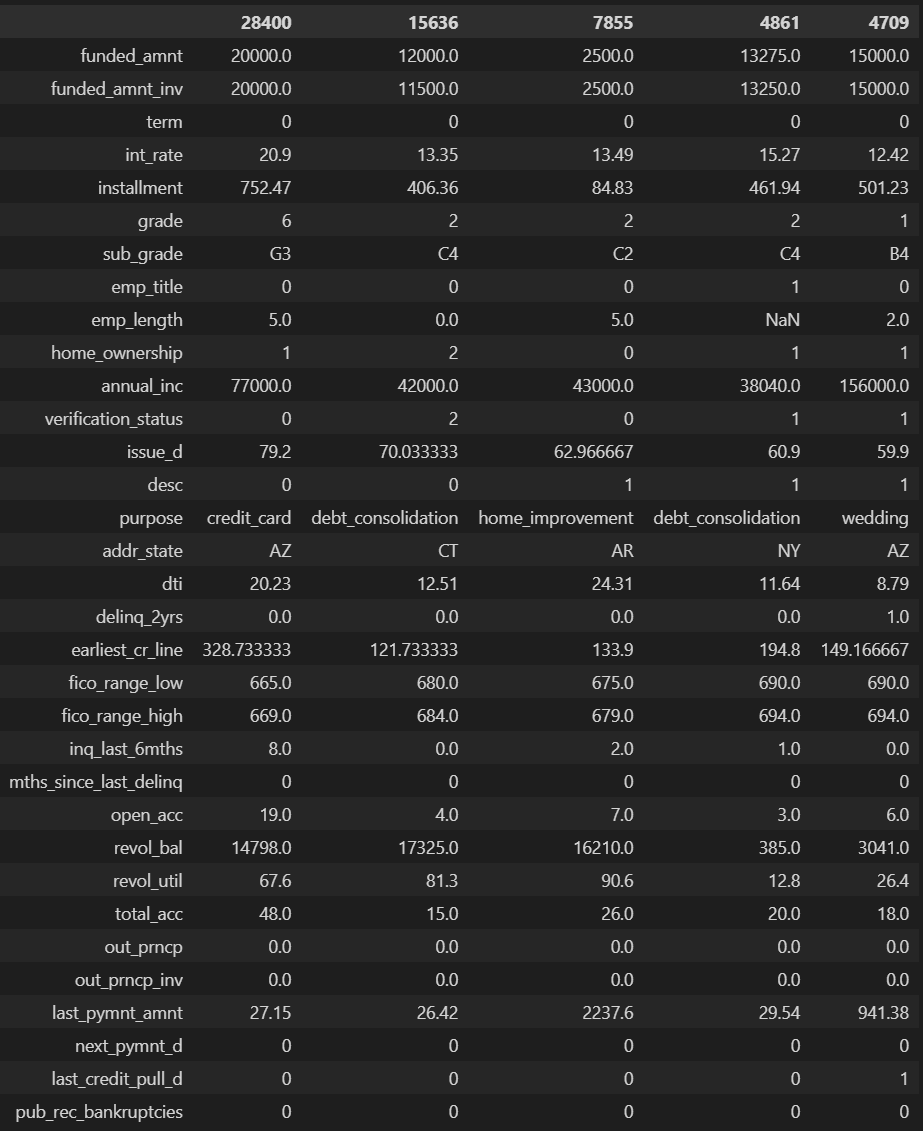
*Categorical Encoding*

* Label Encoding: Features with ordinal categories (grades, subgrades) are mapped to numerical values (e.g., A:0, B:1) preserving the order.
* One-Hot Encoding: Features with nominal categories (e.g., verification\_status, addr\_state) are split into multiple binary columns to represent each unique value.

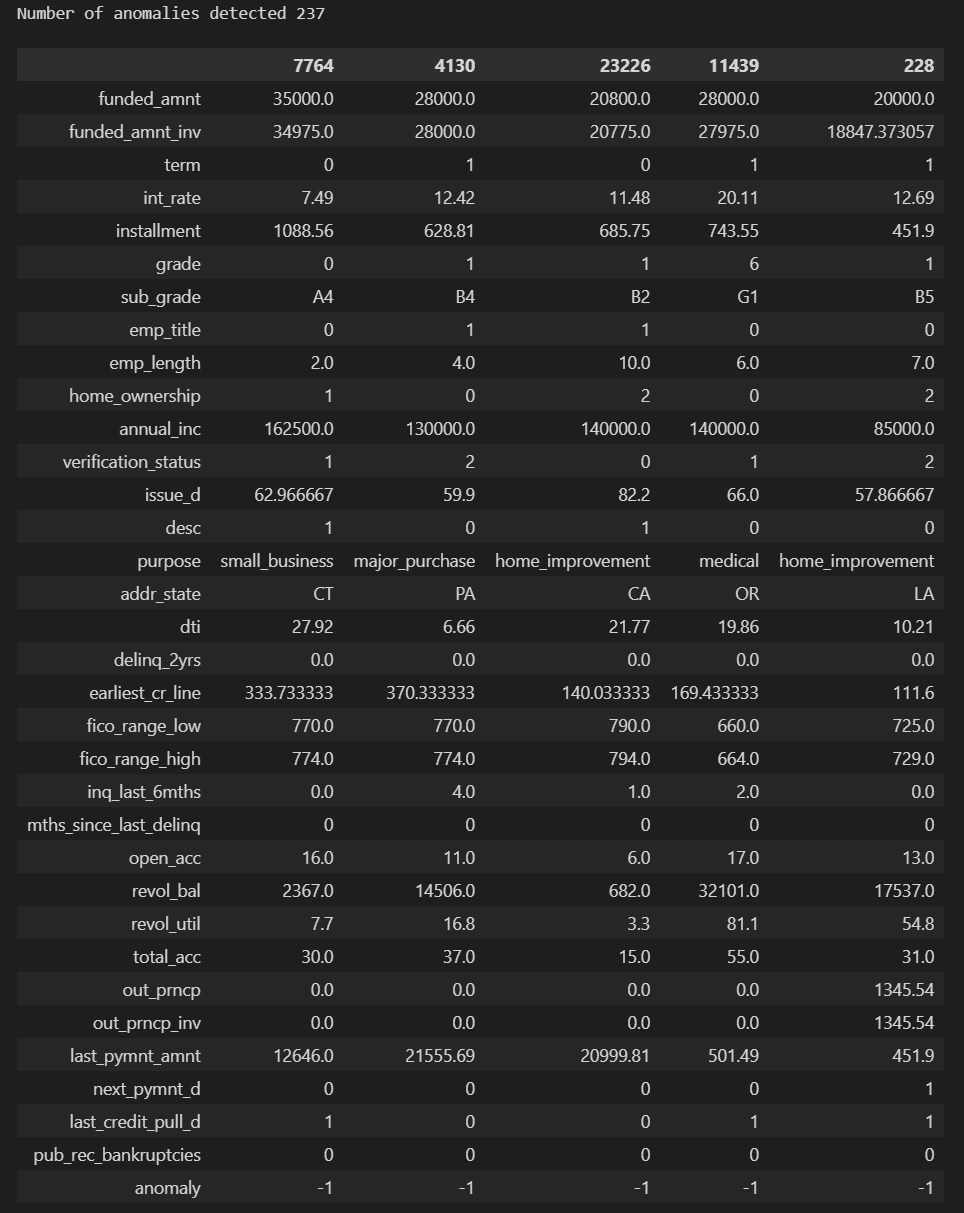
*Numerical Transformations*

* Type Conversion: Strings are converted to appropriate number types (e.g., int\_rate to float).
* Time Calculations: Date features are transformed into the number of months from a reference point, making them more meaningful for modeling.
* Missing Value Handling: The 'desc' feature is binarized, indicating whether a description is present or missing.

*Dataset post transformation*



**Anomaly Detection**



We've noticed some potential anomalies in these records:

* Relatively high income
* The first 3 borrowers have last payment amounts that are notably higher than their funded amounts.
* The first borrower has a significantly high funded amount and a high debt-to-income ratio (DTI).

## II. Model Development

Model Training: Develop models using Logistic Regression, Random Forest, and GBM/XGBoost, Neural Network and an AugoGluon or SKLEARN stacking ensemble model on the training data.

* We've constructed several models and aim to identify the most robust one based on its ROC-AUC performance on the validation dataset. A higher ROC-AUC indicates a better ability of the model to distinguish between default and non-default transactions.
* The models we've developed include:
* Logistic Regression
* Base and Fine-tuned Random Forest
* Base and Fine-tuned XGBoost
* Base and Fine-tuned Neural Network
* Stacking Ensemble (composed of Logistic Regression, Gradient Boosting Machine (GBM), XGBoost, Random Forest, with logistic regression as the final estimator.

**Parameter Tuning**: Optimize model parameters to enhance performance.

To efficiently fine tune our models, I employed a two-step hyperparameter tuning approach for **Random Forest** and **XGBoost** models. ROC-AUC is used to select the best performing model.

Step 1: Random Search

* I began with a random search, exploring various combinations of hyperparameters within a defined search space. This will give us a good starting point for defining the search space for our grid search.

Step 2: Grid Search

* The best hyperparameter values identified in the random search were used as a starting point for a focused grid search.
* The grid search systematically explored values near the previously identified optimal hyperparameters, helping us refine our model configuration for peak performance.

For fine-tuning **Neural Network**, since I built the model with TensorFlow, its fine-tuning process is slightly different:

Model Creation:

* I define a function to create a Keras model with customizable parameters like the number of hidden layers, neurons, activation function, and dropout rate.

Early Stopping:

* Early stopping is set up to prevent overfitting during training by monitoring validation loss.

Hyperparameter Grid:

* A grid of hyperparameters is defined, including the number of hidden layers, neurons per layer, dropout rates, and epochs.

Manual Grid Search:

* Grid search is performed by iterating through each hyperparameter combination. For each combination, a model is trained, and its performance is evaluated using the recall metric.
* The best performing model's parameters and recall score are recorded.

Result:

* Finally, the best parameters and the corresponding best recall score are printed.

## III. Model Evaluation

1. Model Comparison: Compare the models based on their performance and feature importance scores to identify the most effective model on both the train and test sets. What metric should you choose to pick the best performing model and why?

The metrics we have chosen to pick the best performing model is ROC – AUC. F-1 score is also another factor that we pay close attention to.

F-score takes the harmonic mean of precision and recall, giving a balanced view of how well the model performed. A perfect F-score of 1 indicates the model found all the relevant cases and none of the irrelevant ones.

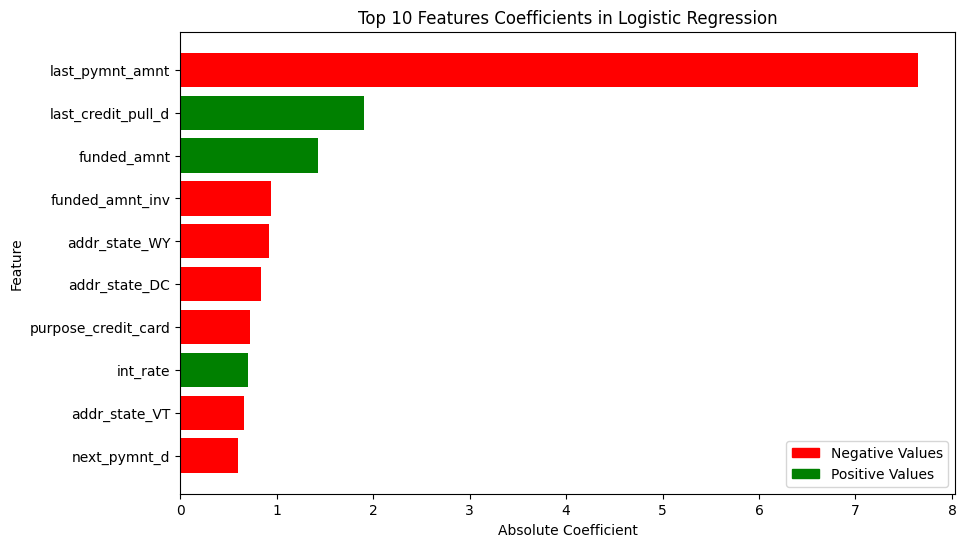
Based on the metrics table below, we can see that the Tuned XGBoost has the highest ROC-AUC and F1-score. Most metrics are calculated using the threshold of 0.5.

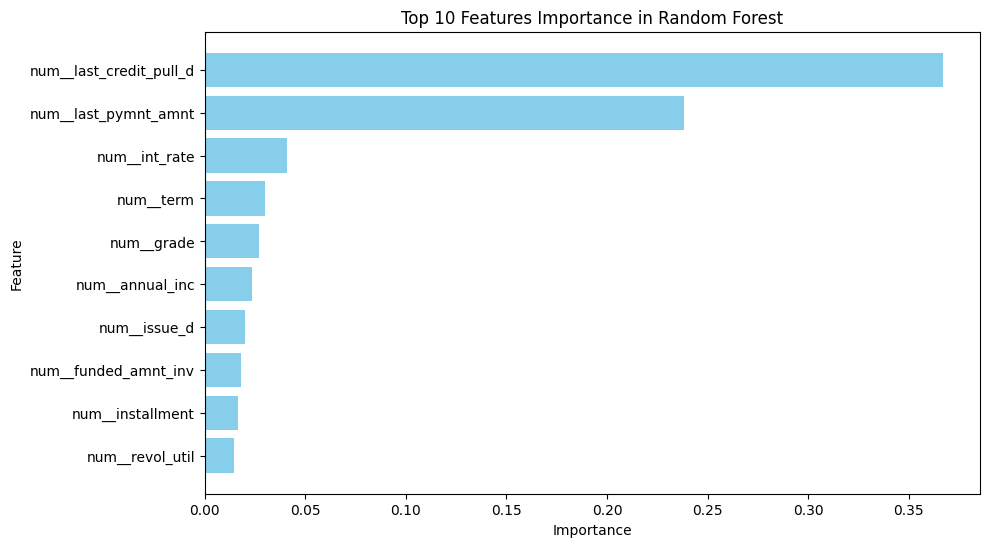
A table of numbers with black text

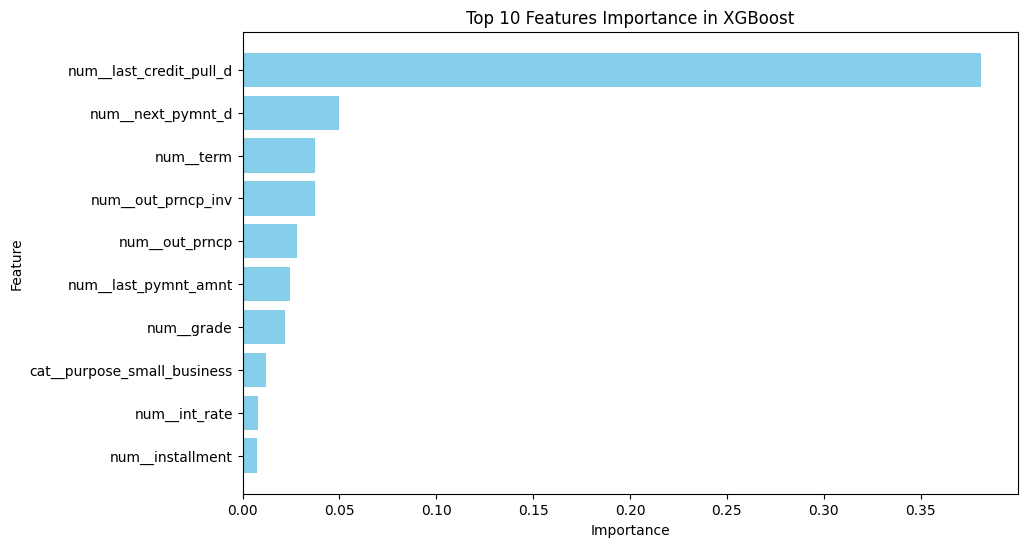
Description automatically generated

### Global Explanation

1. Feature Importance: compare the feature importance from one model to another, why would they be different?

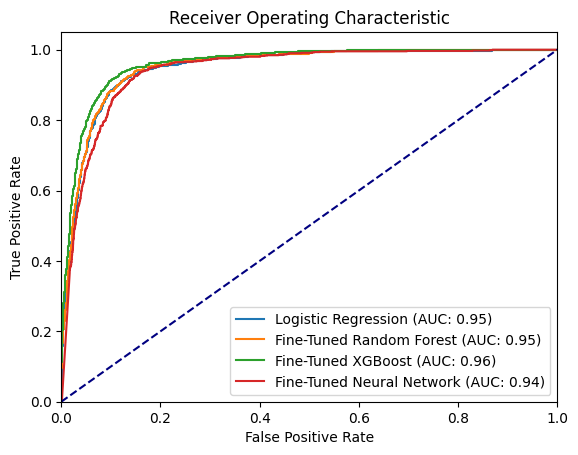






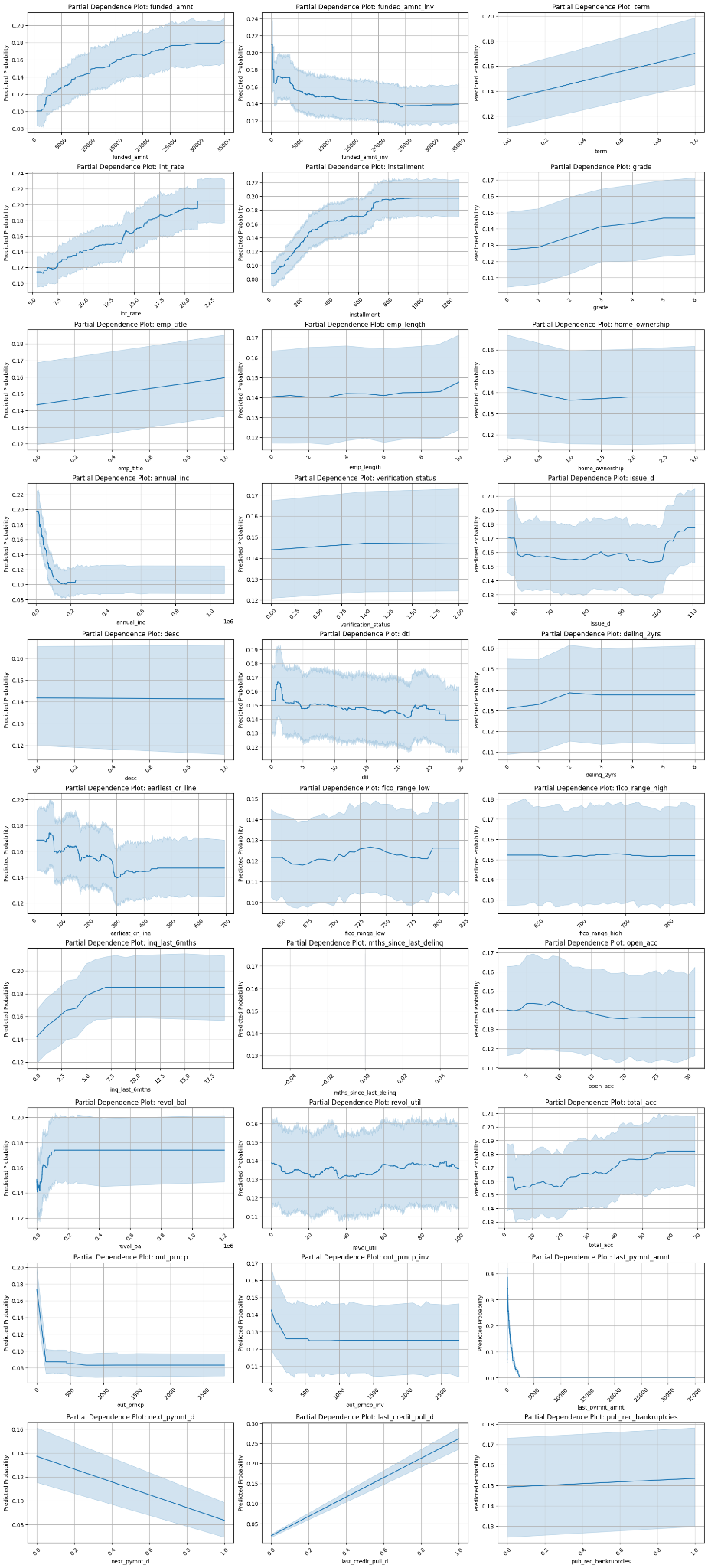
* Though the most important features often overlap between these algorithms, subsequent rankings can vary significantly between tree-based models and logistic regression. This is due to several factors:
* Model Focus: Logistic regression emphasizes linear relationships between features and the outcome. In contrast, Random Forest and XGBoost excel at capturing complex, non-linear interactions, leading to different feature prioritization.
* Importance Calculations within Tree-based Models: While both Random Forest and XGBoost rely on decision trees, there are subtle differences in how they measure feature importance:
* Random Forest: Prioritizes features that consistently reduce impurity across trees, even if the improvements are incremental.
* XGBoost: Emphasizes features that drive strong, immediate performance gains within the sequential tree-building process. This may favor features particularly useful early in the process.

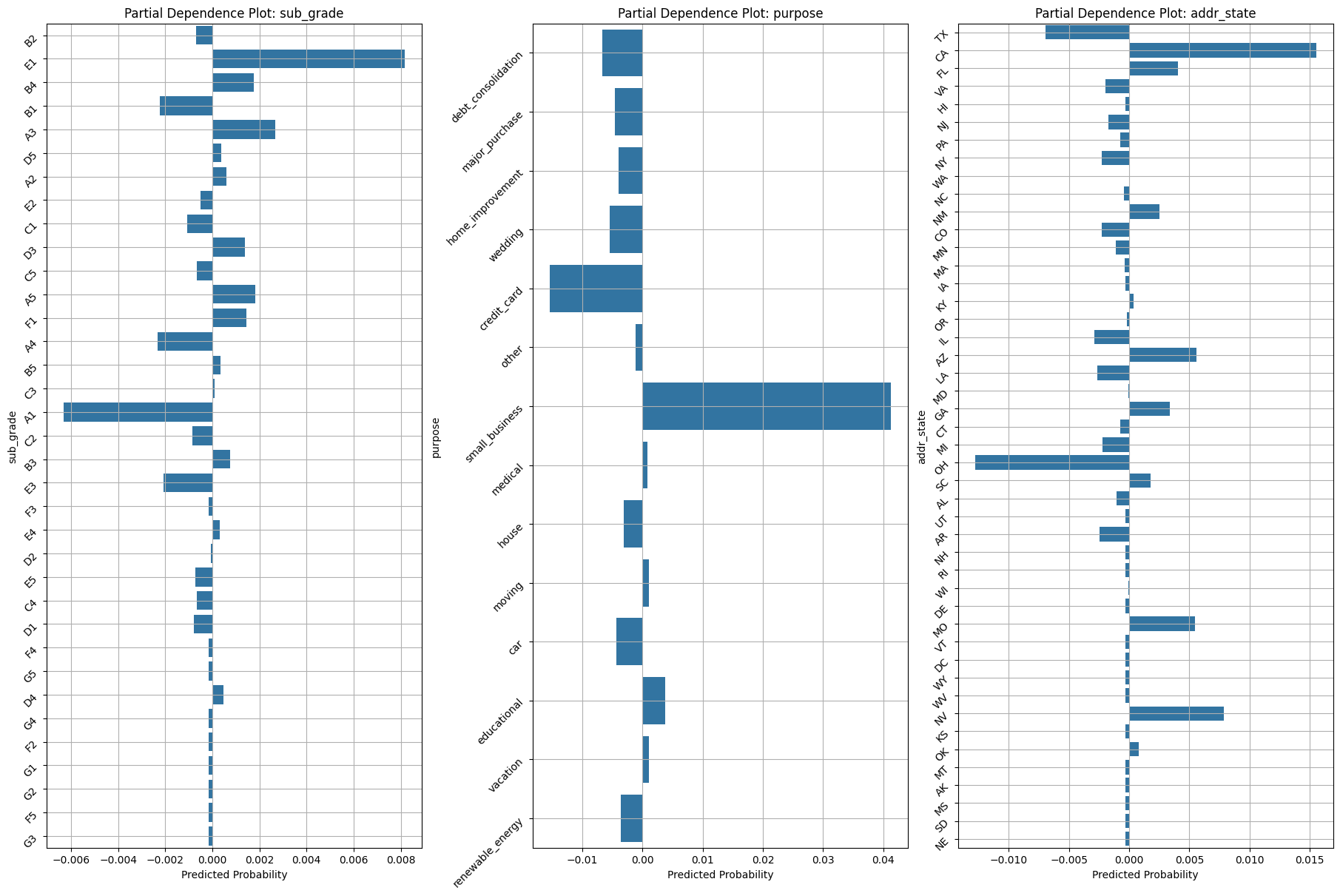
1. Roc Curve & PR Curves on TEST set. you need to plot all of your models with a ROC curve and a PR curve, explain what they mean and how to interpret them.



* The tuned XGBoost’s ROC-AUC is found to be the best. This score on the test set indicates exceptional performance in differentiating between default and non-default transactions, even when faced with previously unseen data.
* A high ROC-AUC score measures a model's ability to discriminate between classes (defaults vs. non-defaults in this case), with a score of 1 representing perfect differentiation, and 0.5 signifying a random guess.
* Other models also perform well with ROC-AUC scores of 0.95 and 0.94.

1. Partial Dependance Plot: Pick your best performing model and generate partial dependance plots explaining the AVERAGE impact to predictions in a way that a business person can understand.





Key factors findings from the partial dependence plot

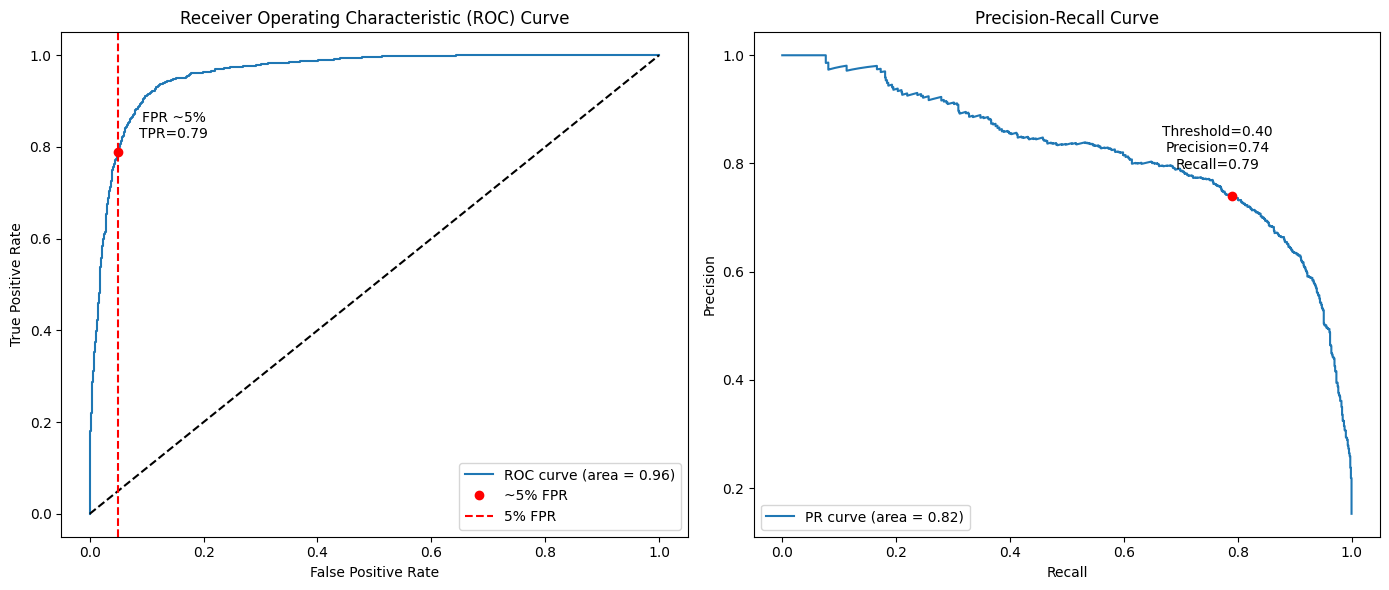
The dependency plot reveals several factors associated with an **increased probability of default**:

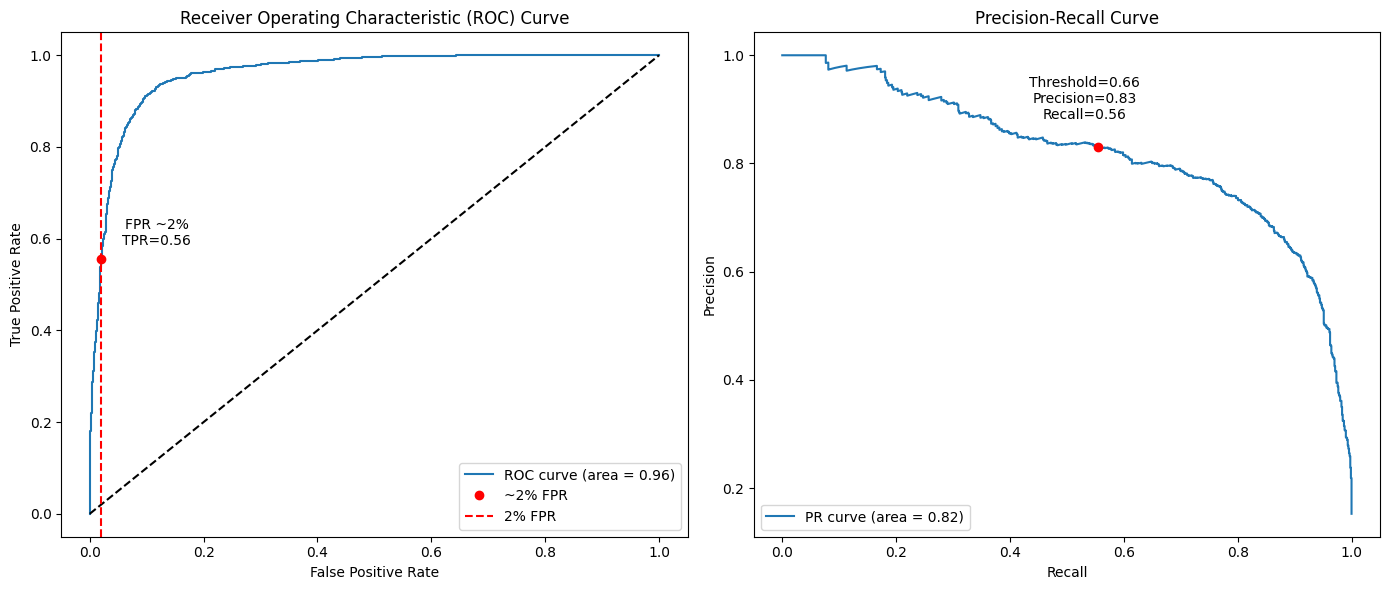
* Interest Rate: When the interest rate is higher, borrowers are more likely to default because they have to pay back more money over time, making it harder to keep up with payments.
* Funded Amount and Term: If the loan amount is larger and the term (duration) of the loan is longer, borrowers may struggle to make payments consistently, increasing the risk of default.
* Monthly Payment Amount: A higher monthly payment puts more strain on the borrower's finances, making it more likely for them to default if they're unable to keep up with payments.
* Total Number of Credit Lines: Having a larger number of existing credit lines indicates that the borrower may already have a significant amount of debt, which increases the likelihood of defaulting on additional loans.
* Small Business: For small businesses, there may be greater uncertainty and variability in cash flow, making it more challenging to meet loan obligations consistently.

Here are some factors that are associated with the **decreased probability of default**:

* Scheduled Next Payment Date: If customers have already scheduled their next payment, it suggests they are actively managing their finances and are less likely to default in the near future.
* Previous Large Payment Amount: Making large payments in the past indicates financial stability and responsibility, reducing the likelihood of defaulting on future payments.
* Lower Interest Rate: A lower interest rate means borrowers have to pay less in interest over time, making it easier for them to manage payments and decreasing the likelihood of default.

1. Operational Strategy at 2% and at 5% FPR: Propose a strategy to achieve and maintain a 2% and also 5% false positive rate, detailing its implications on recall and precision. What does this mean for the business in plain language?





* To maintain false positive rates of 2% and 5%, we need to devise a strategy that carefully balances the trade-offs between identifying actual loan defaults and minimizing incorrect predictions.

At a 5% False Positive Rate:

* Correctly Identified Loan Defaults (True Positive Rate): 79%
* Precision (Accuracy of Identified Defaults): 74%
* Model Threshold (Decision Point): 0.4
* This means that out of all the cases the model identifies as indicating a default (including both correctly identified defaults and incorrectly identified ones), 74% are indeed true loan defaults. However, it also means that around 21% of actual loan defaults are missed by the model (false negatives). With a threshold set relatively low at 0.4, more cases are classified as potential defaults, leading to increased true positives but also more false positives.

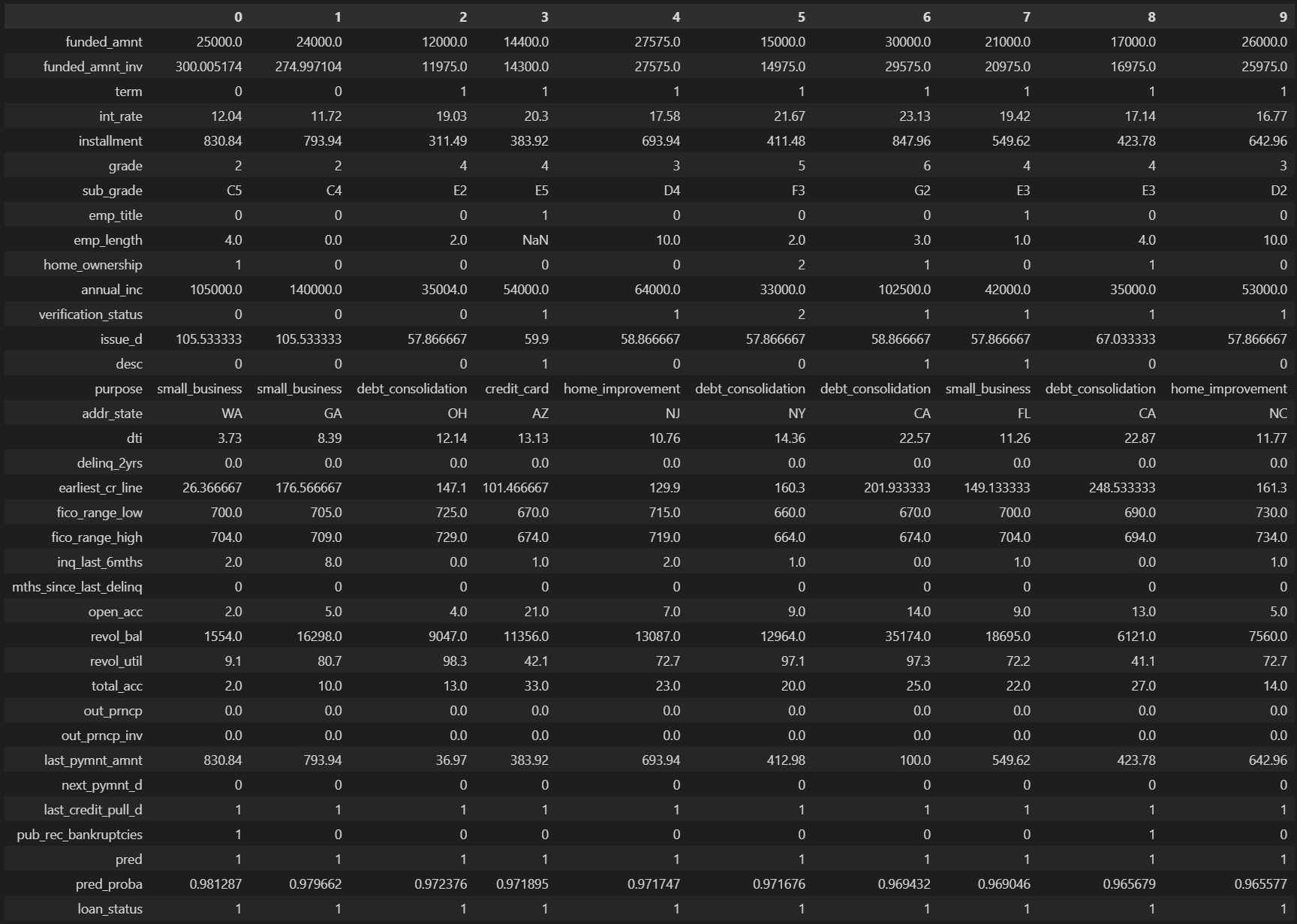
At a 2% False Positive Rate:

* True Positive Rate drops to 56%
* Precision improves to 83%
* Model Threshold increases to 0.66
* Here, our goal is to reduce the false positive rate to 2%, meaning only 2% of the cases identified as indicating a default are not defaults. However, achieving this lower false positive rate results in a lower true positive rate of 56%. On the positive side, precision improves significantly to 83%, indicating that a higher percentage of cases identified as defaults are indeed true defaults. To achieve this lower false positive rate, we adjust the threshold for classifying cases as defaults to 0.66, making the model more conservative in labeling cases as indicating a default.

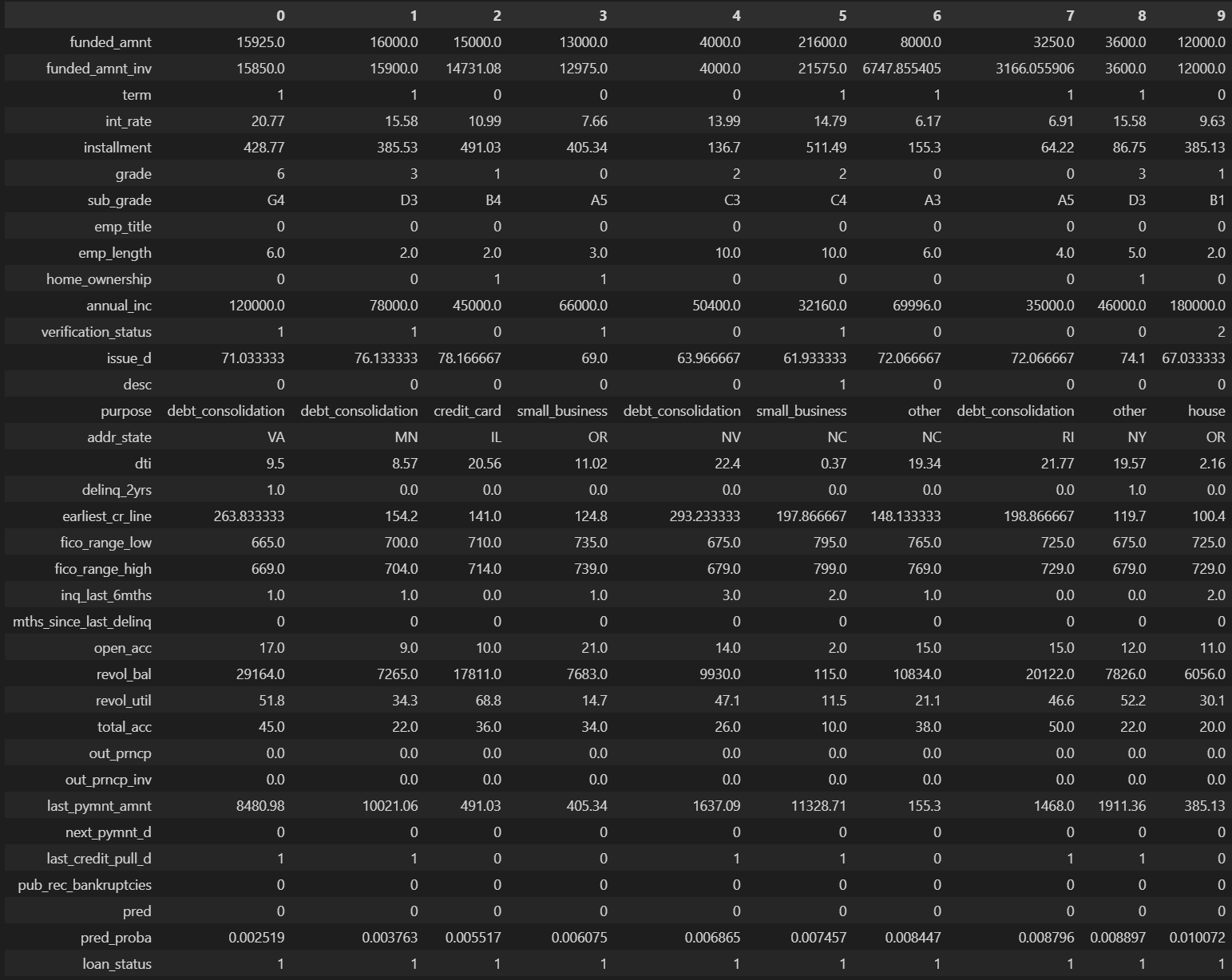
### Local Explanation

Using your best performing model, make predictions on your test set and identify the TOP 10 best True Positives, your TOP 10 False Positives and your Top 10 False Negatives. Eyeball the data or use plots or trees or some other method to explain why you think you gave a prediction a particular score. Is there a pattern or common theme that you notice for wrong predictions?

*Top 10 True Positive*

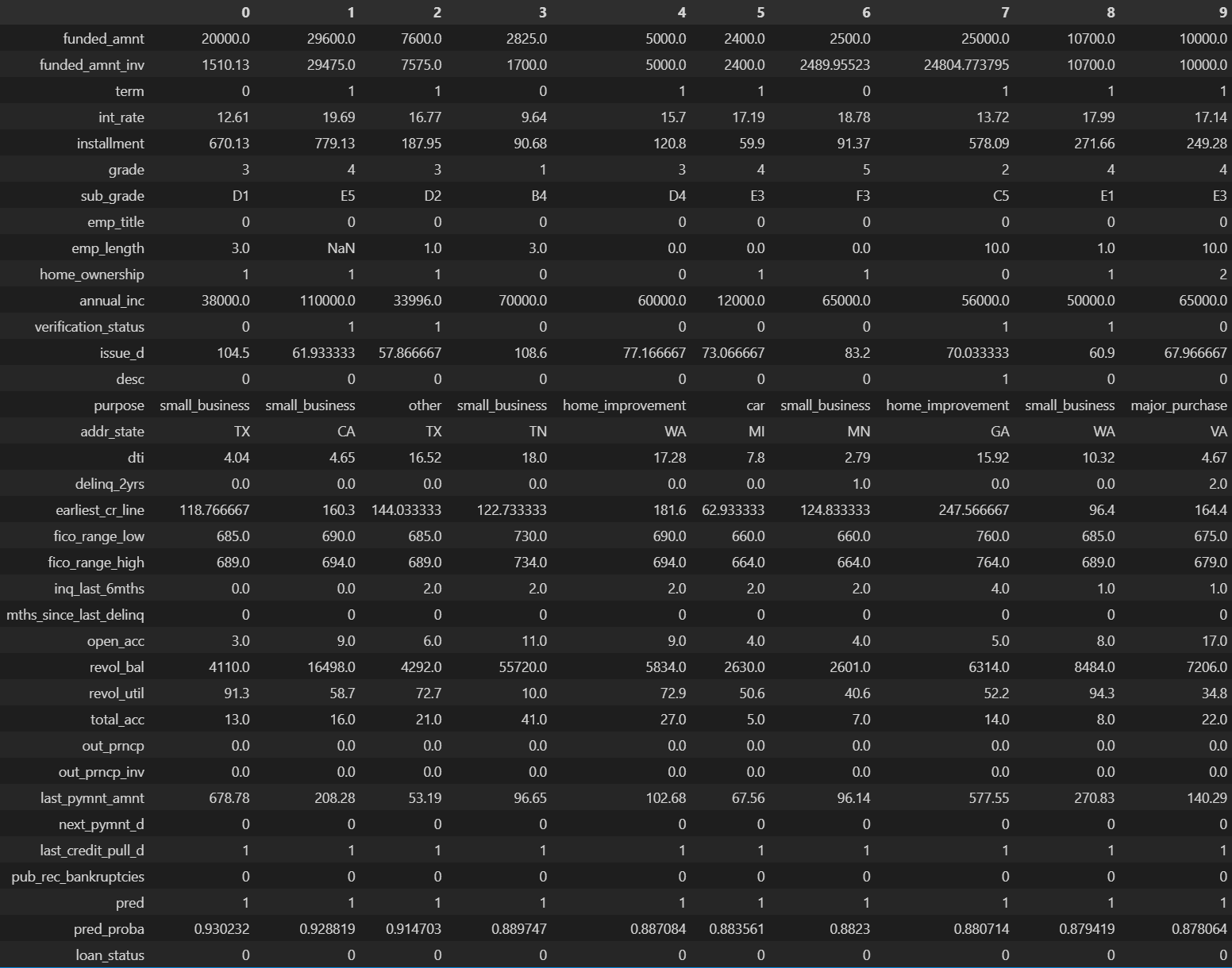


*Top 10 False Negative*



* Our analysis of the table and local breakdown plot shows that our model misclassified default loans primarily due to high last payment amounts. Since high last payments typically reduce default risk, the model was misled. This pattern is clear within the top 10 false negatives, where unusually high last payments are common.

*Top 10 False Positive*



* Feature importance analysis reveals 'last\_credit\_pull\_d' as the most influential factor across our logistic regression, random forest, and XGBoost models. This feature tells us if the borrower made any credit transactions within the current month. And our dependence plot shows that a value of '1' in this feature correlates with a higher predicted default risk. This is supported by the table, where all top 10 observations have a '1' in this column.
* Additionally, some of these borrowers have relatively low last payment amounts, a pattern also identified in our dependence plot as contributing to higher default probability.
* These factors led the model to misclassify these borrowers as default, due to their heightened risk.