Prepayment Risk

Y. Fan, T. Fan, X. Liu & J. Gruber

Baruch College MFE

August 26, 2024

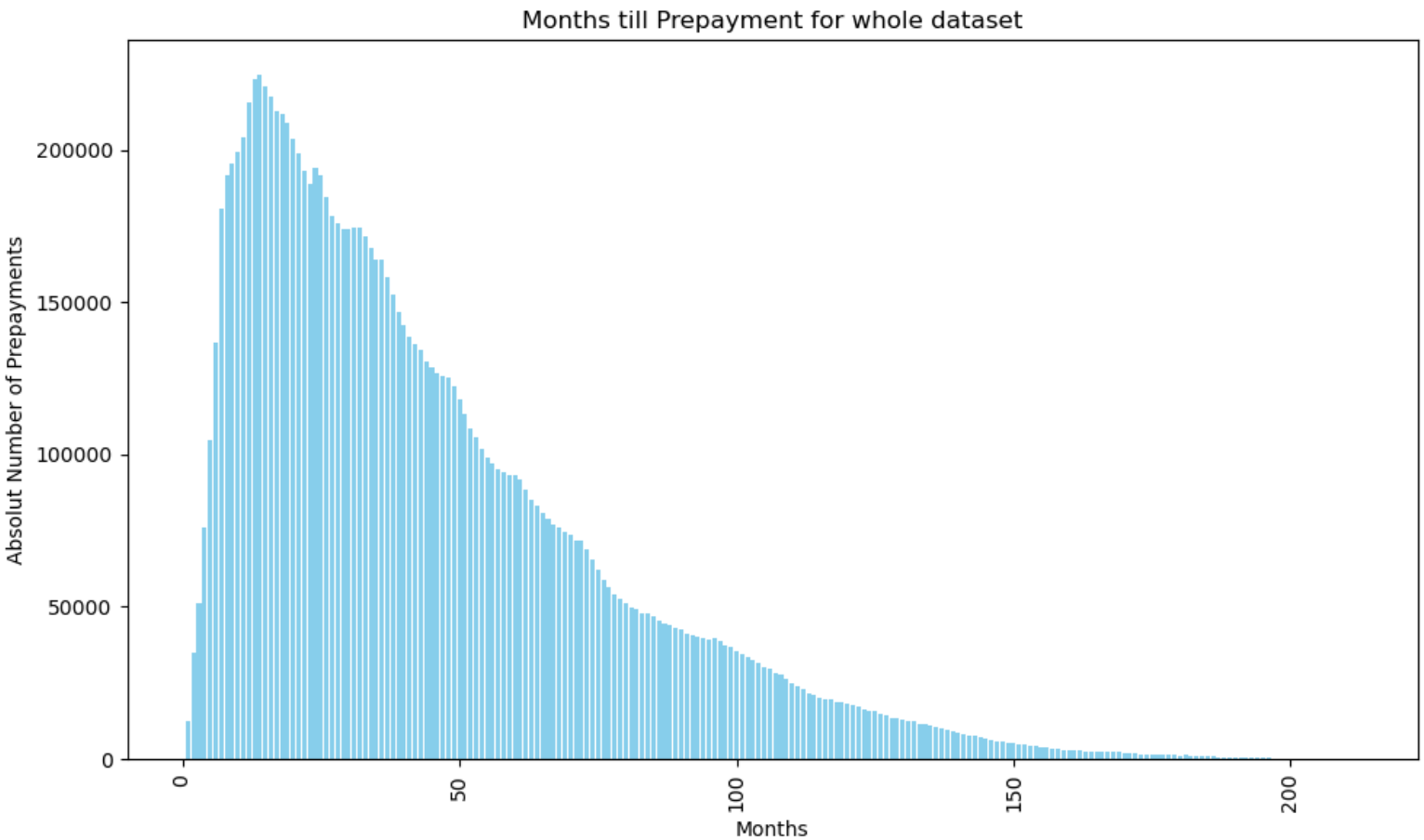
Overview

EDA of the historical times data

An explanatory data analysis was conducted first to look into the data. We tried to answer the following questions:

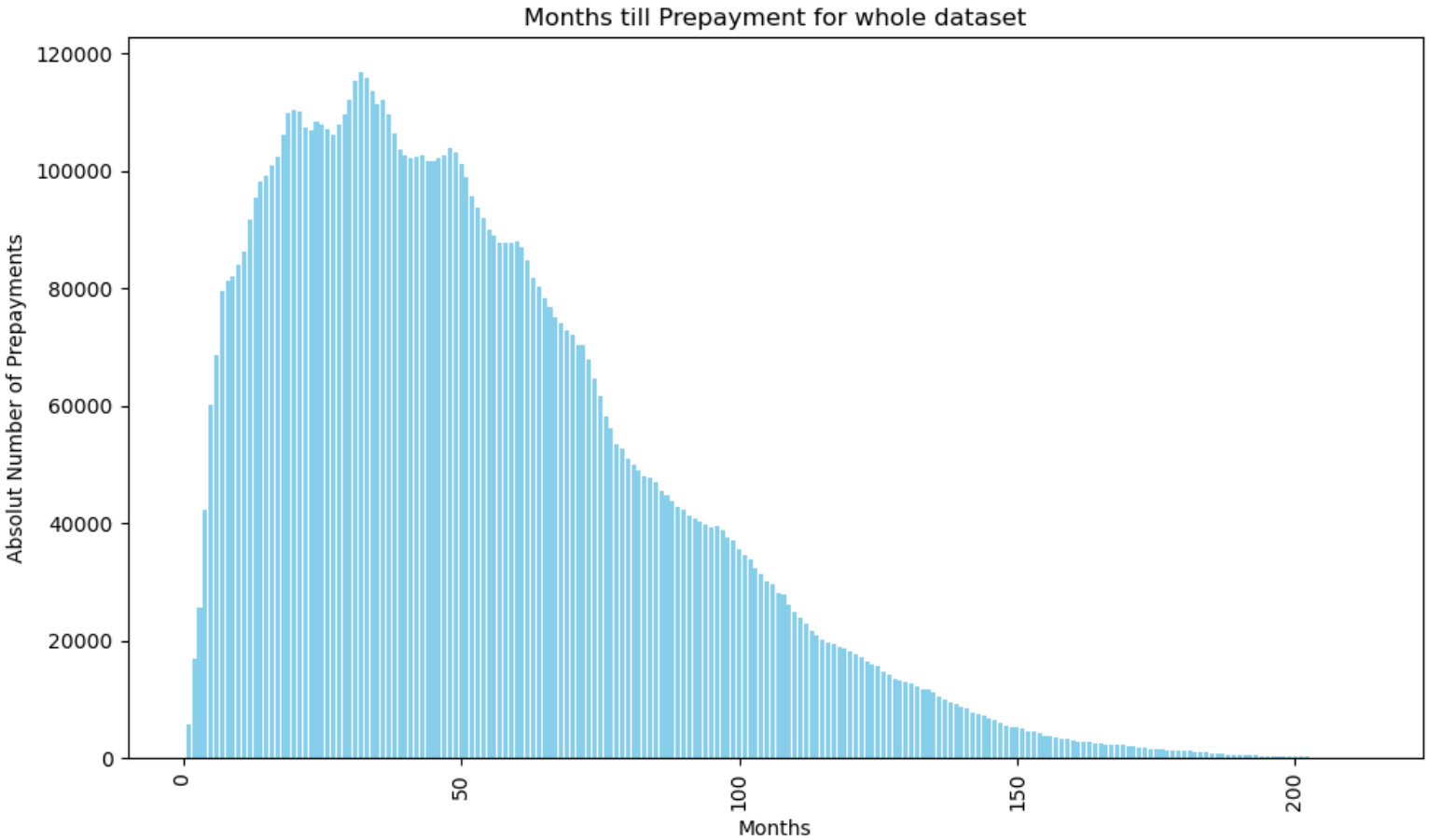
- When does prepayment happen?
- Finding a reasonable cutoff number for training and test splits
- How many loans are getting prepaid in total

After how many months does prepayment occur



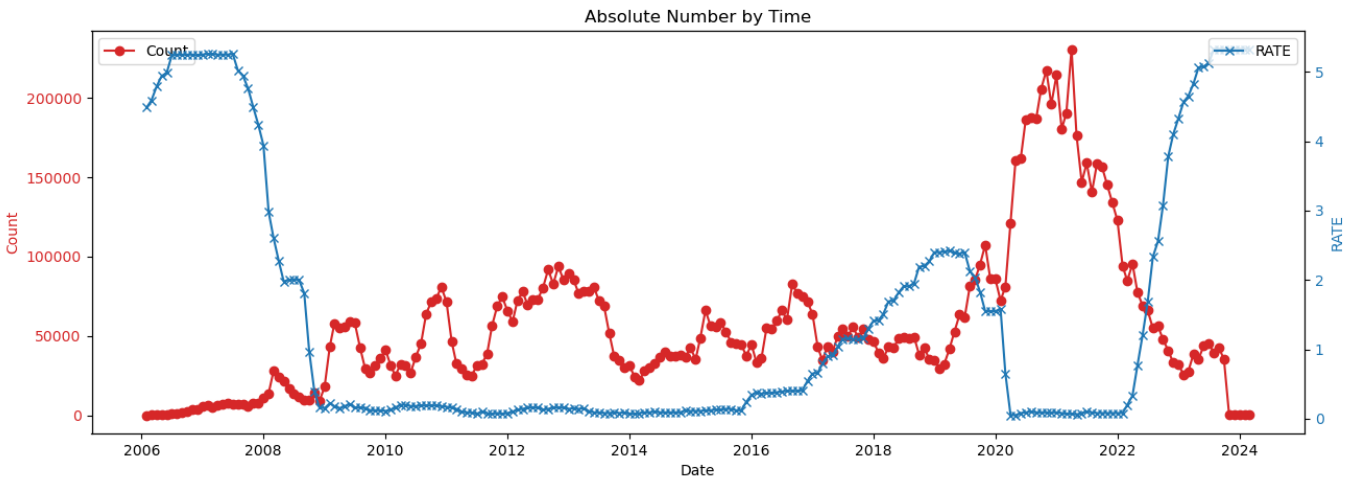
We can see in the graph that most prepayments occur after 7 years. Hence we can use this as a cuttoff period. We validate this by looking at the distribution if we exclude the prepayments from the last 7 years.

Prepayment vs Months - including cutoff



Absolute number of Prepayment vs Interest Rate Changes

Figure: Interest Rates and Absolute Number of Prepayments



What is the task that we are trying to solve

While working on the project, we realized that there are two tasks we could solve:

- Problem 1: Given a loan profile at time 0, will there ever be a prepayment? Hence Survival Analysis:
- Problem 2: Given a loan at time t, what is the probability that it will be prepaid at time t+1.

Problem 1: Analysis

Although this is the easier problem, we realised that there is a strong sensitivity to interest rates in this problem(hence a exegenous covariate). The python packages we found were not able to capture both dynamics, hence different starting dates combined with different exegenous covariates(interest rates). Though this might be an interesting question to answer, we decided to go for Problem 2, as it might be more applicable for industry related purposes. TODO: validate statement with regards packages not being capable.

Problem 1: Analysis

We have decided to first address problem one, which requires a smaller amount of data, to validate our data processing and methodology, before proceeding to problem two. For each loan in problem 1, we use its Origination data and the Performance data of the first month to model and predict its prepayment within 4 years. We have also incorporated fundamental data (e.g. mortgage rate unemployment rate, and House Price Index) for prediction.

Problem 2: Analysis

Although this is the easier problem, we realised that there is a strong sensitivity to interest rates in this problem(hence a exegenous covariate). The python packages we found were not able to capture both dynamics, hence different starting dates combined with different exegenous covariates(interest rates). Though this might be an interesting question to answer, we decided to go for Problem 2, as it might be more applicable for industry related purposes.

Data Description

Loan Classification: The dataset differentiates between a Standard Dataset for CRT-eligible mortgages and a Non-Standard Dataset for loans with specific characteristics that exclude them from CRT transactions.

Loan Features at Origination: It encompasses various loan attributes such as loan-to-value (LTV) ratio, debt-to-income (DTI) ratio, credit scores, mortgage insurance percentage, property type, and location.

Loan Performance Data: The dataset provides monthly loan performance data, including loan balance, delinquency status, and information up to loan termination events like foreclosure or REO disposition.

Problem 1: Cleaning the Data

Handling Missing Data: Samples missing crucial variables are excluded. The key variables include FICO score, CLTV and LTV. These features are universally available and missing are likely to be the result of reporting errors. We also drop the samples which has missing features that rarely miss in the whole dataset since we can hardly find information from it. For other features with missing data, indicator variables are used to denote missing data. We use median or mode (depending on the features) imputations to fill the missing data and use a boolean variable to indicate whether the feature is missing. This approach helps us to identify the implication of missing. For example, DTI ratio is often missing in subprime loans.

Problem 1: Cleaning the Data

Making Categorical Variables Dummy: For tree and neural network models, we simply encode the categorical features as numerical numbers and feed them into the models. For models like logistic regression, Categorical variables are treated as dummy variables in our data. To avoid perfect multicollinearity, we remove one dummy variable for each categorical variable and make it base class.

Building Targets: For static prediction problem, our target is whether a loan will prepay in a certain period of time (4 years). We get the information of whether a loan is prepaid and the time before prepayment from the time evolution data of the loans.

Add Macro and Regional Features: We added quarterly mortgage rate and unemployment rate as macro features for each loan. We also added housing price index on Zip code and on states as regional variables to denote the regions.

Problem 1: Modeling Approach

Our purpose is to predict the target (whether a loan will prepay in 4 years) by utilizing features we mentioned before combined with machine learning models.

Boosting Tree - LightGBM: Boosting tree methods are well known to be good at tabular data. Since we are also facing categorical features and missing values here, using tree models can steps of constructing one hot encoding and be more robust to any poterntial outliers or noises in the data.

Neural Network: As presented in the paper "Deep Learning for Mortgage Risk", Neural Network's extraordinary capability in dealing with non-linearity and interaction effects in the data can be useful when we are trying to model loan prepayment, which is full of those. We choose specific architecture of Neural Network according to the paper:

"5 hidden layers, 200 units in the first hidden layer, 140 units in each subsequent one. RELU activation."

Problem 1: Flow Chart

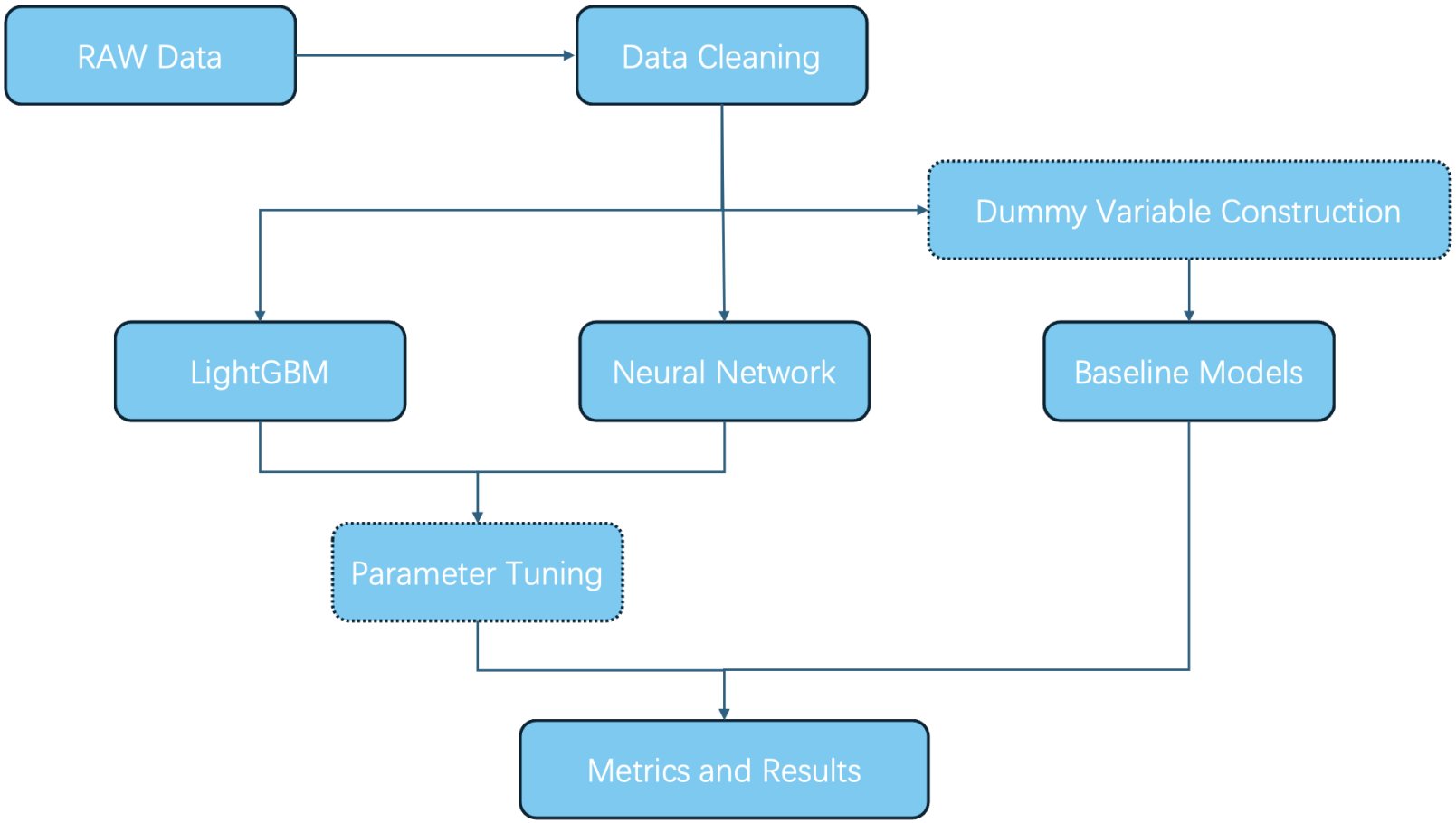


Figure: Flow Chart

ML model Results

Metrics:

1. Accuracy
2. ROC Curve
3. Precision-Recall
4. Other Indicators

We report the results of our prediction task, including metrics accuracy, ROC, Precision-Recall, but also feature importance in the tree model and train/test loss curve in the neural network model.

Results: Accuracy

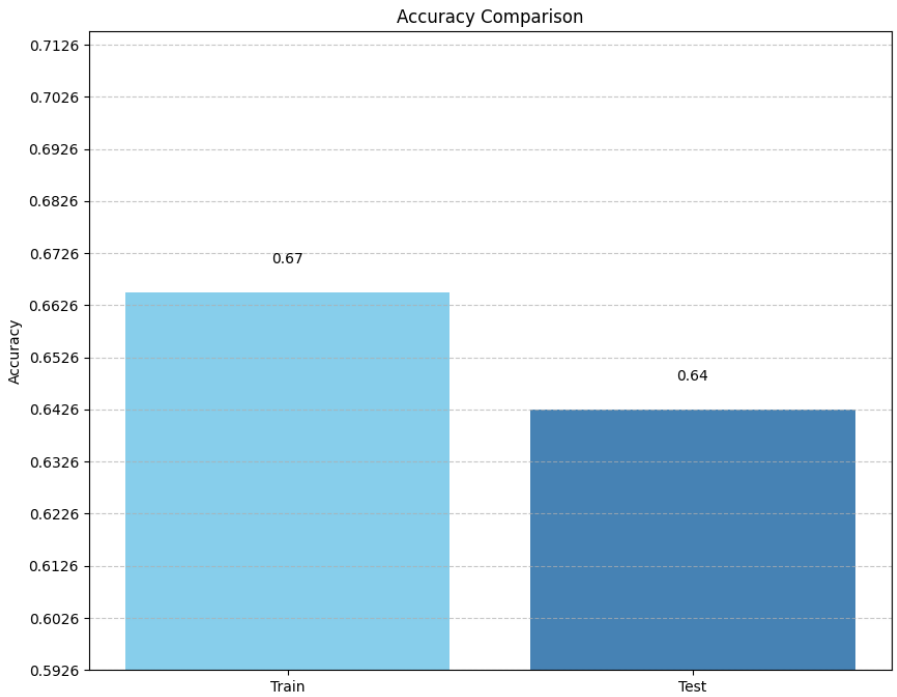


Figure: LightGBM

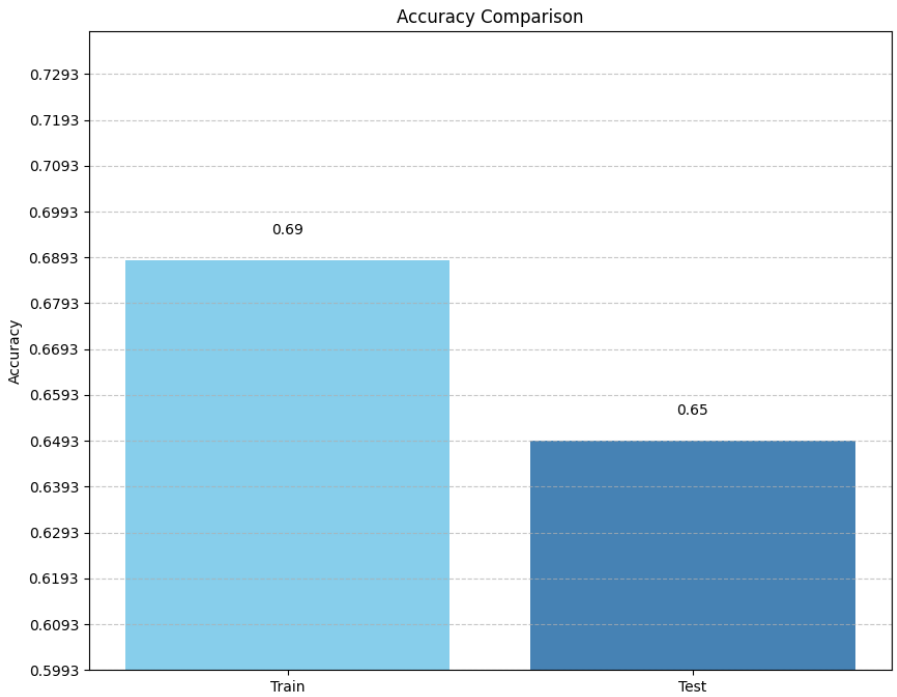


Figure: Neural Network

Results: ROC Curve

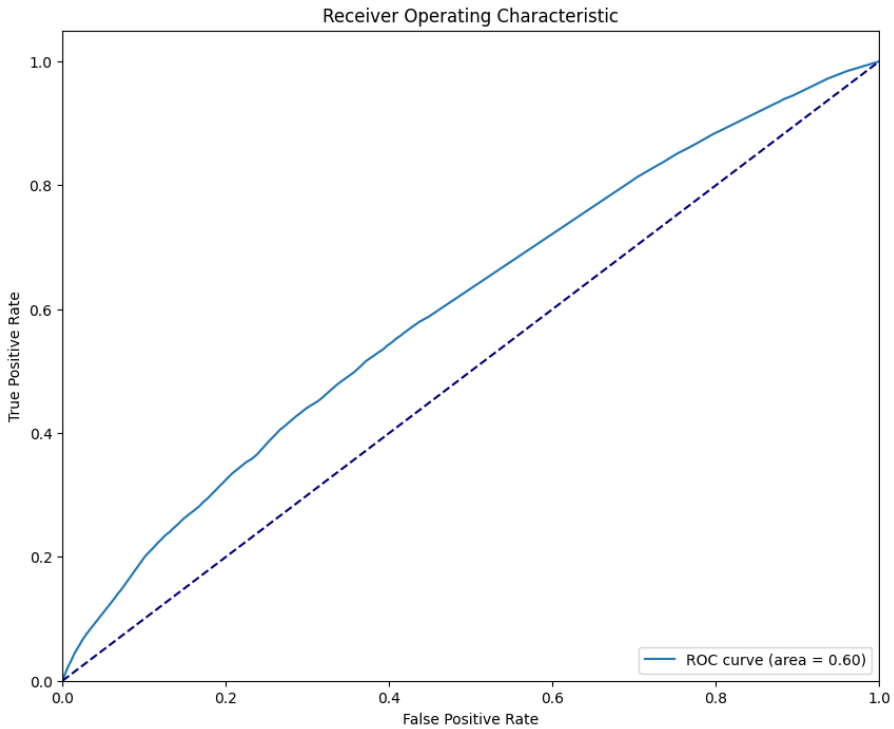


Figure: LightGBM

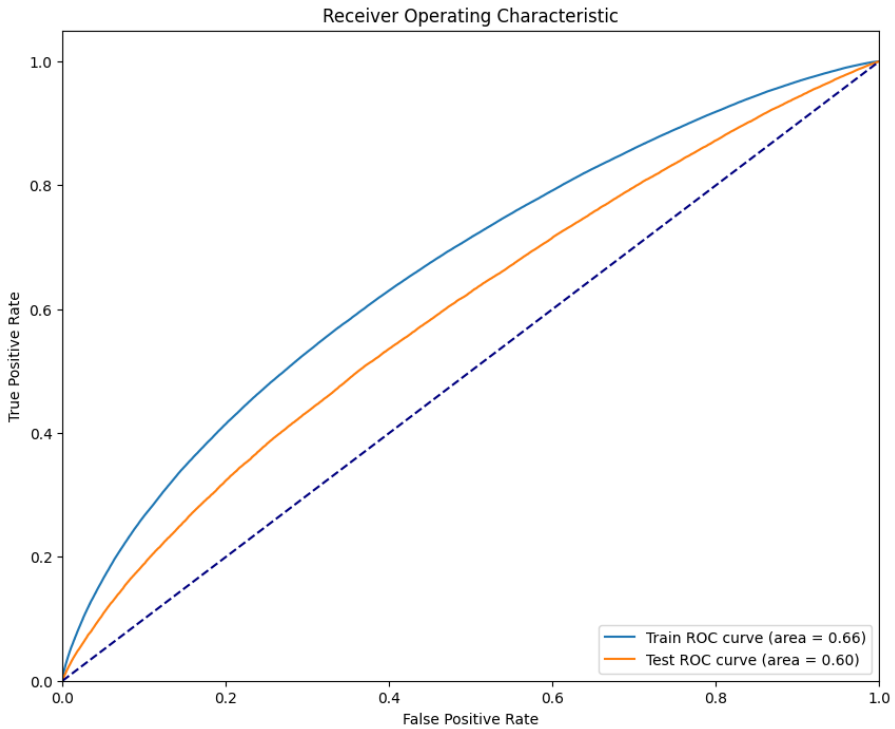


Figure: Neural Network

Results: Precision-Recall

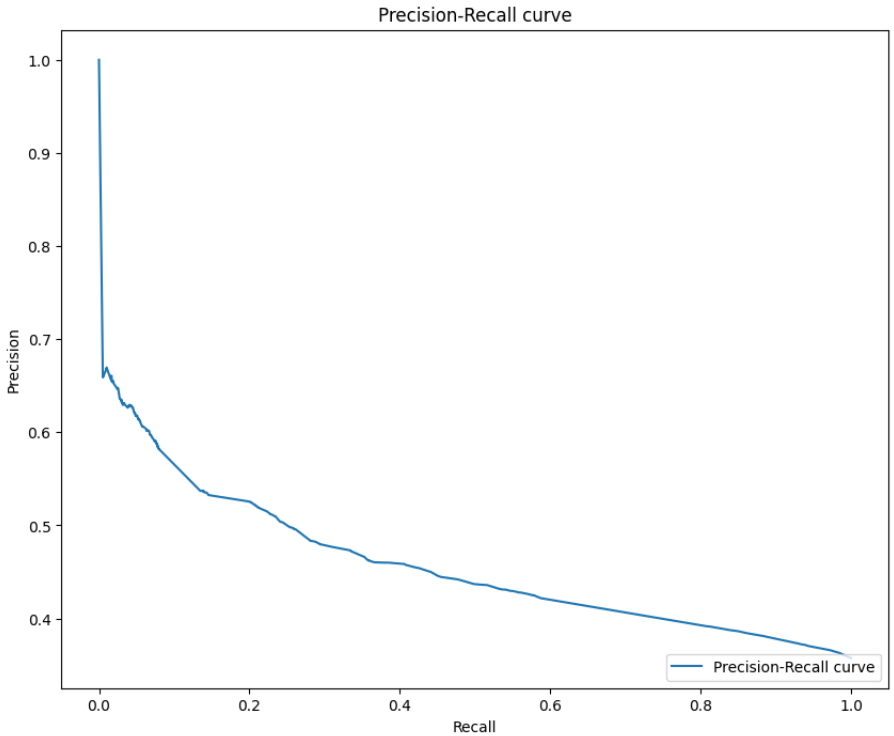


Figure: LightGBM

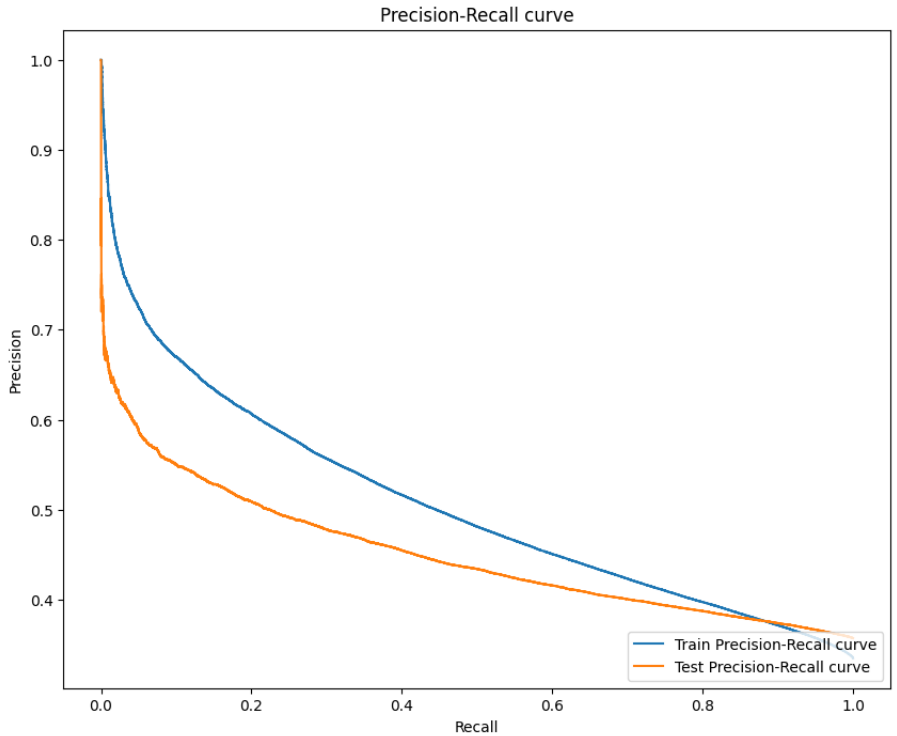


Figure: Neural Network

Results: Other Indicators

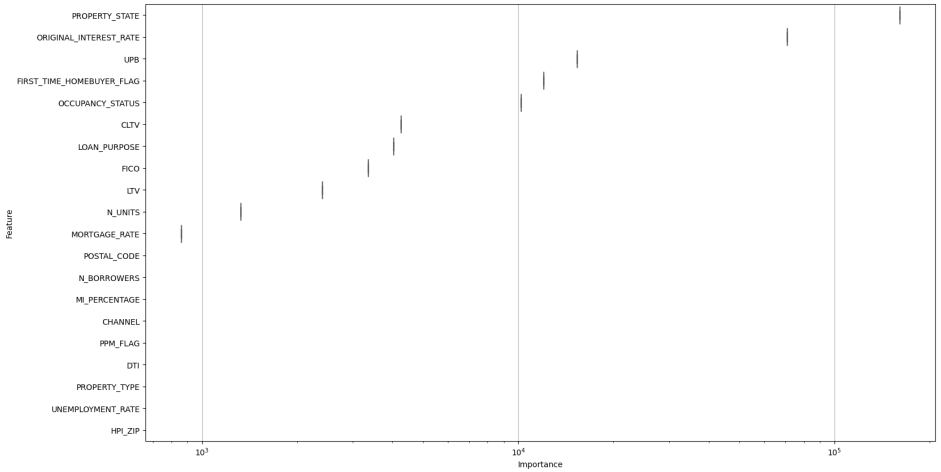


Figure: LightGBM Feature Importance, in split count

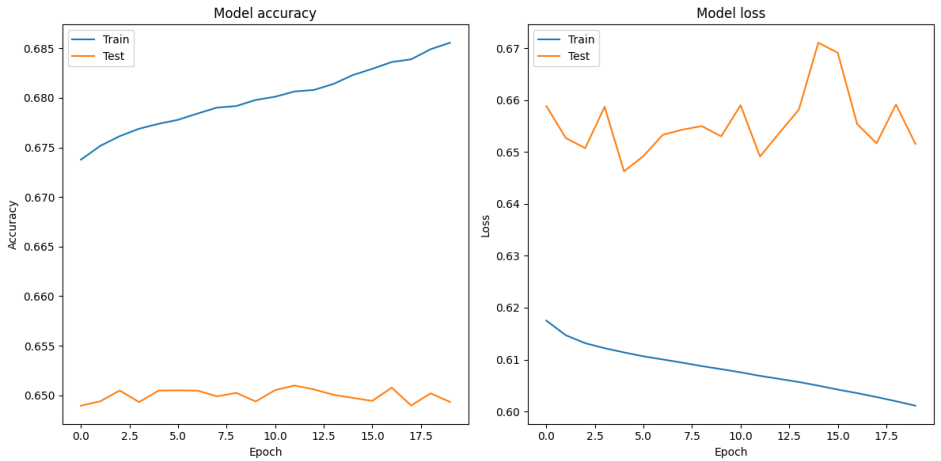


Figure: Neural Network loss/accuracy curve

Conclusion on Task 1

As we have demonstrated earlier, using the ML model to predict 4-year prepayment shows good results in terms of accuracy, ROC curve, and other indicators. However, it can also be seen that there is a noticeable overfitting phenomenon in both models. This could potentially be improved by increasing the amount of data used during training. Nevertheless, both models have shown significant prepayment prediction capabilities.

Thoughts on Task 2

Whilst this project seemed more reasonable to do for a practical perspective we faced several issues that we want to point out here We will also talk about the following technical points:

- polars vs pandas
- storing parquet vs csv vs txt

Data Preperation

1. Sort the historical_time_data by month into seperate folders
2. Do feature engineering on the historical_data files
3. Merge the histrical time data with the feature engineering dataset
4. train a classification algorithm on month t data and test on month t + 1 for performance

Polars vs Pandas

Feature	Pandas	Polars
Memory Usage	Higher, because it is unoptimized.	Optimized with a columnar memory layout only one type per column
Performance	Good for smaller datasets	Utilised parallelisation, faster for bigger dataset
API Design	Mature API	Newer API, cleaner design
Ease of Use	Very well-documented	Less documentation relative to Pandas

Table: Comparative Overview of Polars and Pandas

Comparison of Parquet, CSV, and TXT Formats

We utilized 3 different storage methods for the project: Parquet, CSV & TXT.

Feature	**Parquet**	**CSV**	**TXT**
Storage Type	Columnar	Row-based	Text-based
File Size	Small	Medium	Large
Read/Write Speed	Fast	Moderate	Slow
Schema Evolution	Supported	Not supported	Not supported
Compression	High	Low	Low
Type Safety	Yes	No	No
Suitability	Big Data	Simplicity	Plain text handling

Table: Comparative Analysis of Data Formats

Parquet is integrated with both integrated with pandas and polars.

Issues we couldn't resolve

Unfortunately, we couldn't find a way to complete the 3rd step of our initial roadmap due to memory errors we got from the code. Our preperared data could not be processed in time, despite the knowledge we gathered from before. We want to propose the following methodoligies that could help:

- More compute power: Using a bigger computer with more RAM or a cloud solution might be a possible solution for the problem
- Utilizing a database system: Instead of working with various parquet and csv files, we could have setup a database system, which is optimized for tasks like this.