Leveraging Machine Learning to Predict Loan Defaults

Phase 4: Final Project HCDR (Implementing Deep Learning)

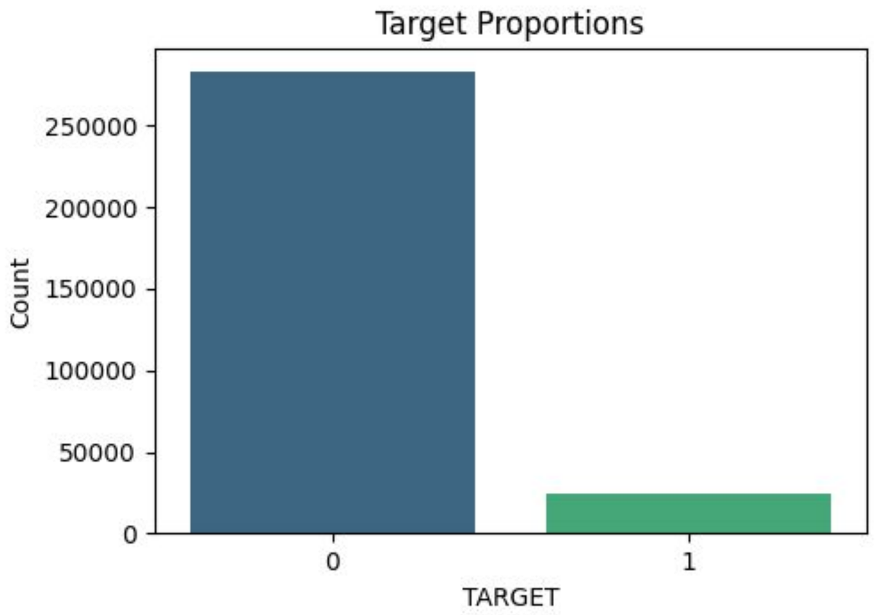
Group 4

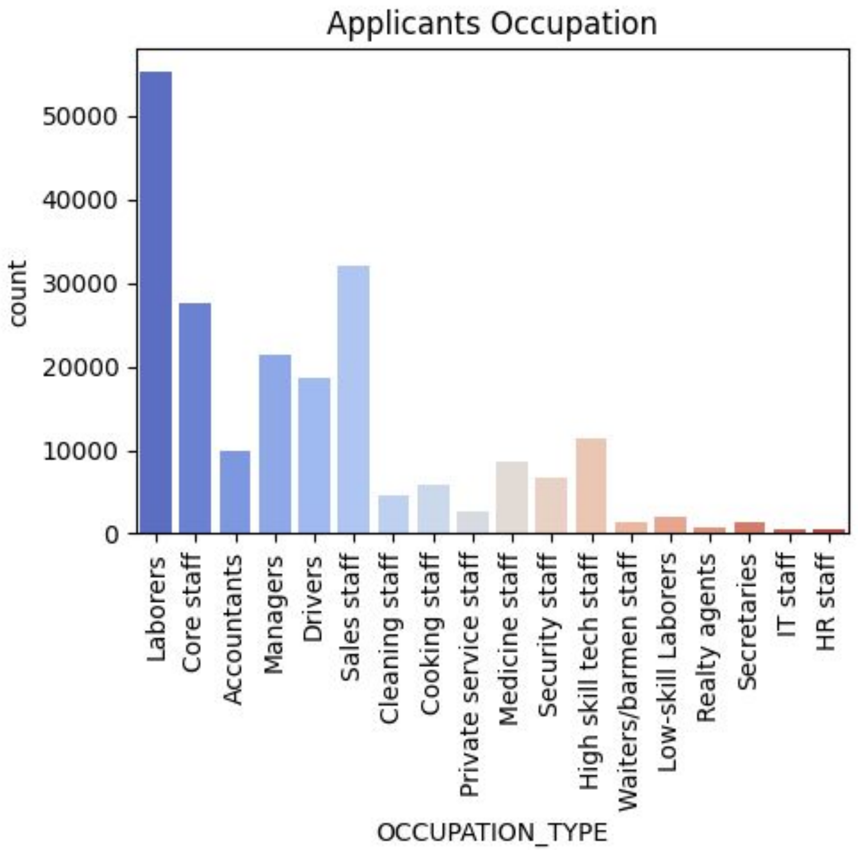
Maria P. Aroca, Cassie Cagwin, Lexi Colwell, Nasheed Jafri

Project Overview

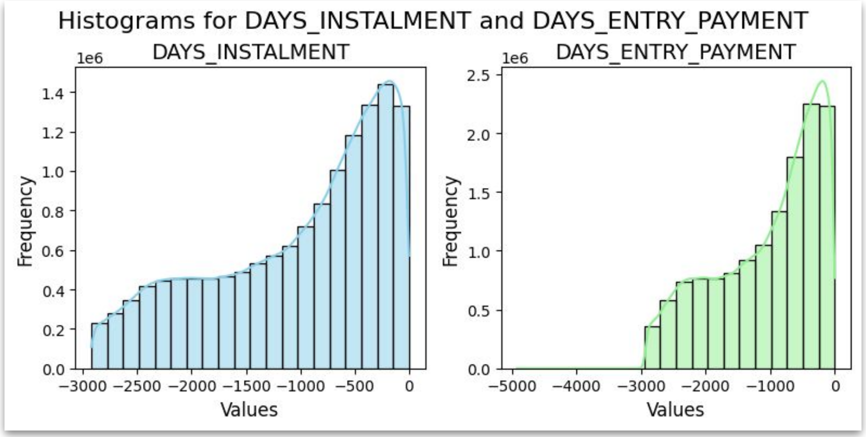
- **Purpose:** The project aims to help lending institutions improve loan approval processes by predicting an applicant's ability to repay loans using a combination of Telco and Transactional data.
- **Problem:** Many loan applications are rejected due to insufficient credit history, forcing applicants to turn to predatory lenders. This project seeks to reduce unjust rejections while minimizing the risk of defaults.
- **Goal:** Build a predictive system to identify potential defaulters using applicant data.
- **Data**: The dataset from the Home Credit Default Risk project comprises a rich and diverse collection of data sources:
    - **Application Data:** Demographic and financial details for current loan applications
    - **Credit Bureau Data:** Records of clients' past credits and monthly balances (other sources)
    - **Credit Card Balances:** Records of monthly balances for previous Home Credit credit cards
    - **POS & Cash Loans:** Monthly snapshots of repayment histories for POS and cash loans
    - **Installment Payments:** Records of repayment histories, including missed installments
    - **Previous Applications:** Records of past loan applications with Home Credit

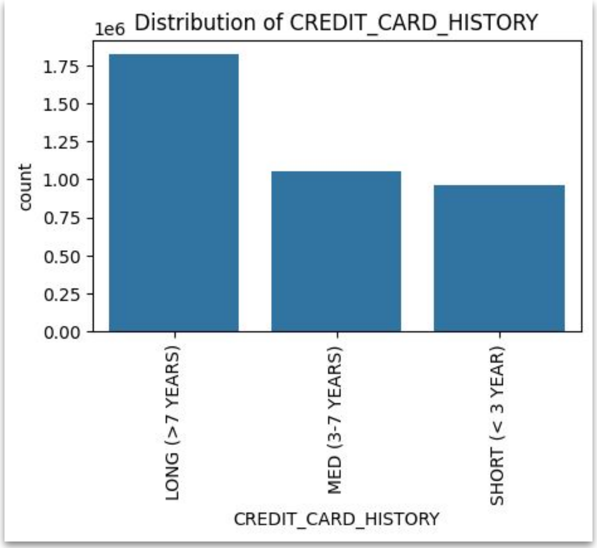
Key insights from EDA

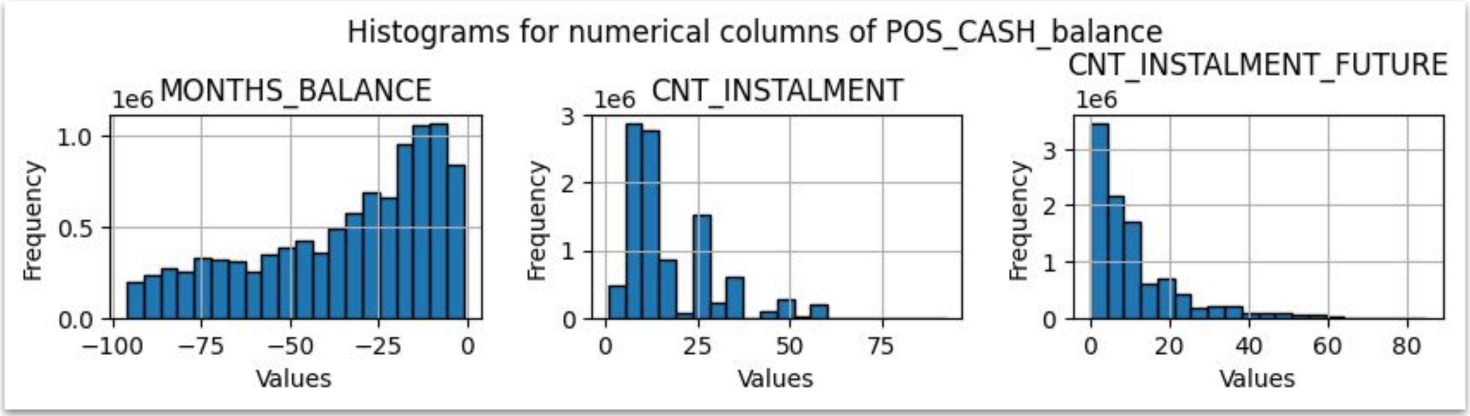




Key insights from EDA

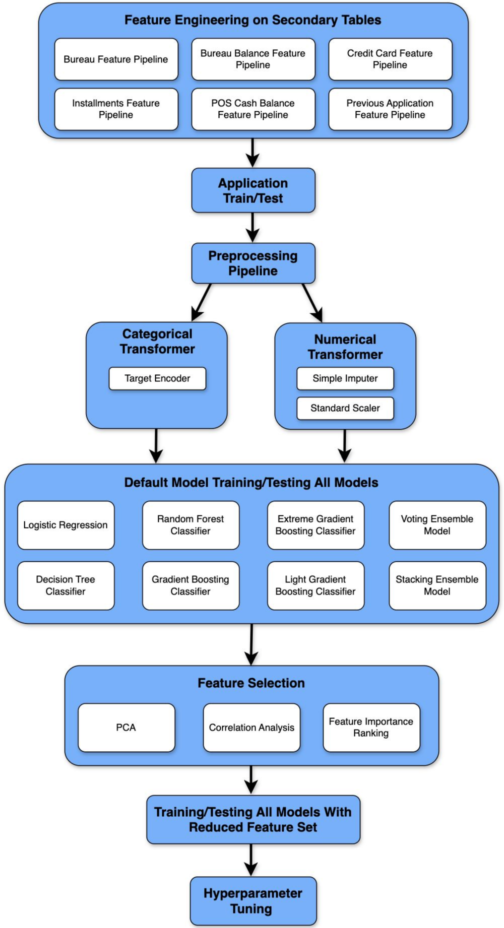




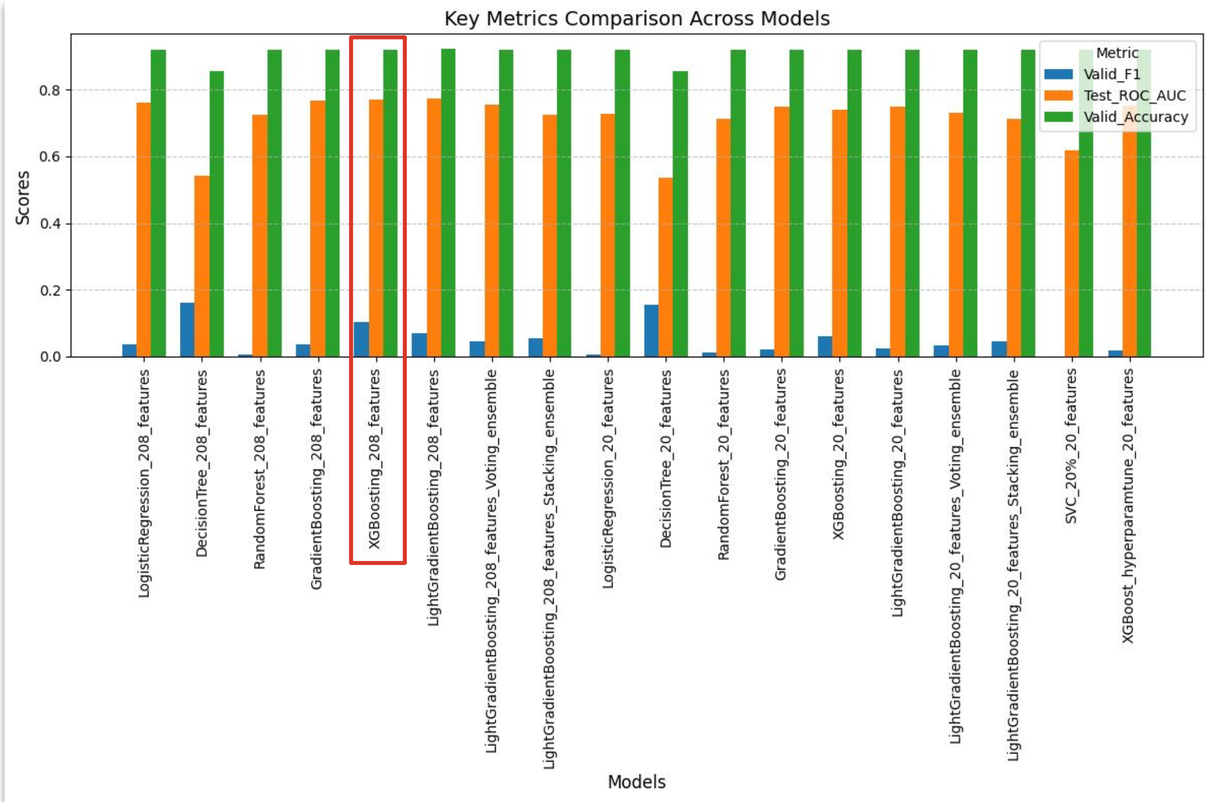


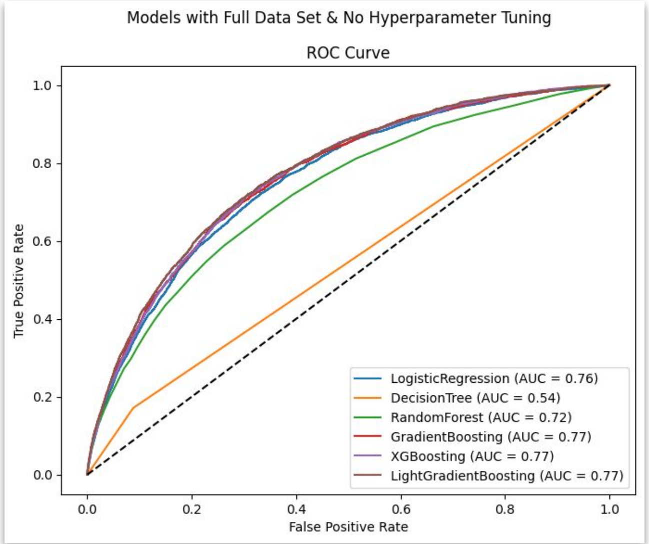
Overview of Modeling Pipeline

- Engineered Features Based on EDA and Business Understanding. Created single dataset.
- Transformed categorical and numerical features
- Used PCA, feature importance from decision trees and random forests and correlation analysis to reduce data to **20 features**
- Tested complete and reduced feature set with different models:
  - Logistic Regression
  - Random Forest
  - Decision Trees
  - Gradient Boosting
  - Extreme Gradient Boosting
  - Light Gradient Boosting
  - Stacking Ensemble Model
  - Voting Ensemble Model
- Hyperparameter tuning on best performing model

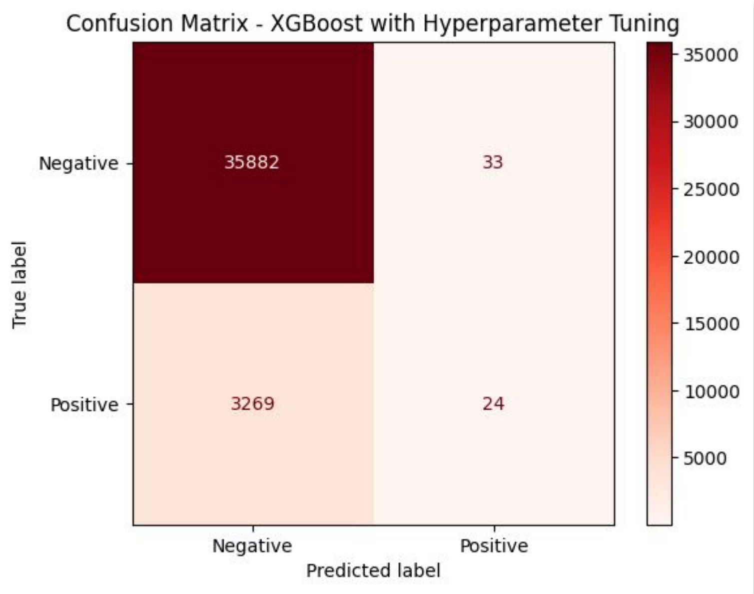


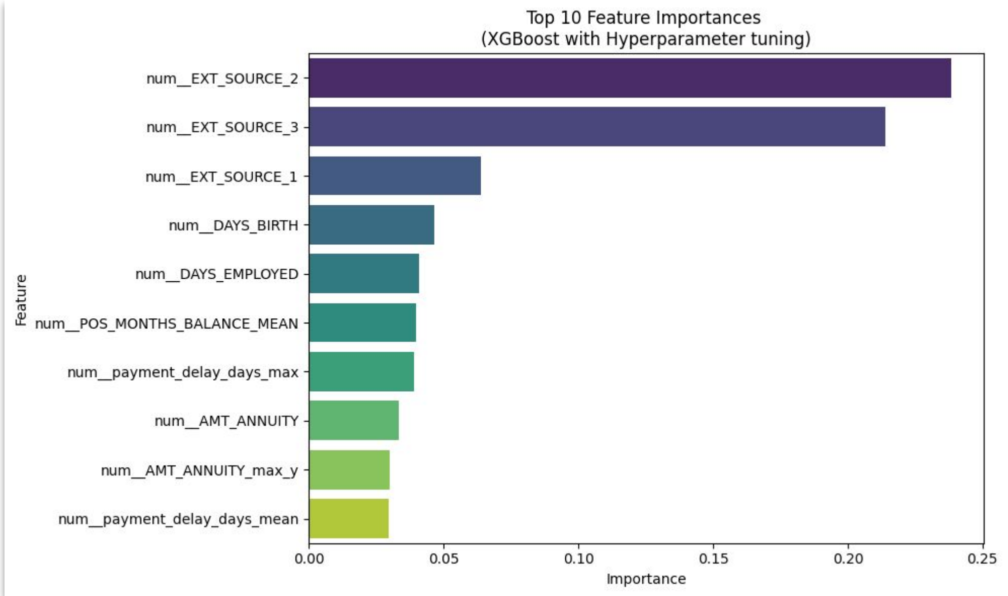
Overview of Modeling Results minus Neural Networks





Overview of Modeling Results minus Neural Networks





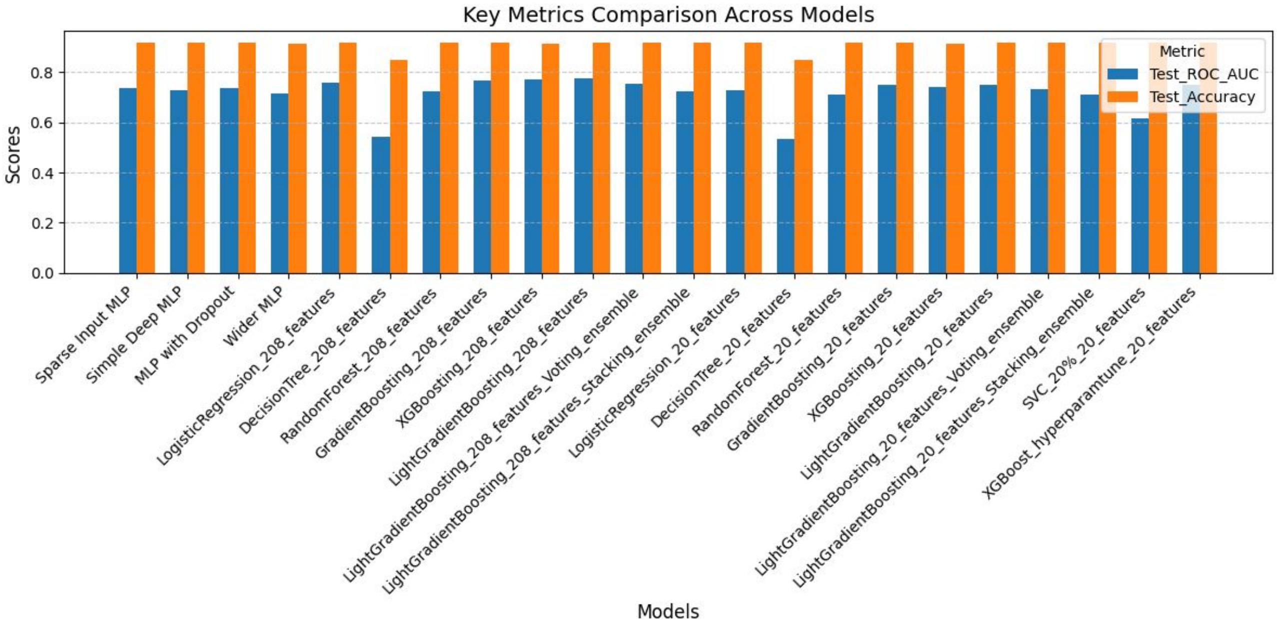
Best Parameters:
Model: XGBoost
  model__colsample_bytree: 0.8
  model__learning_rate: 0.1
  model__max_depth: 5
  model__subsample: 1.0

Multi-Layer Perceptron (MLP)

- **Sparse Input MLP:** A lightweight baseline with fewer neurons to benchmark
- **Simple Deep MLP:** Added depth (128, 64, 32 units) to capture complex patterns in the data
- **MLP with Dropout:** Incorporated dropout (rate = 0.2) to mitigate overfitting, improving generalization
- **Wider MLP:** Increased width (512, 256, 128 units) for greater representational power

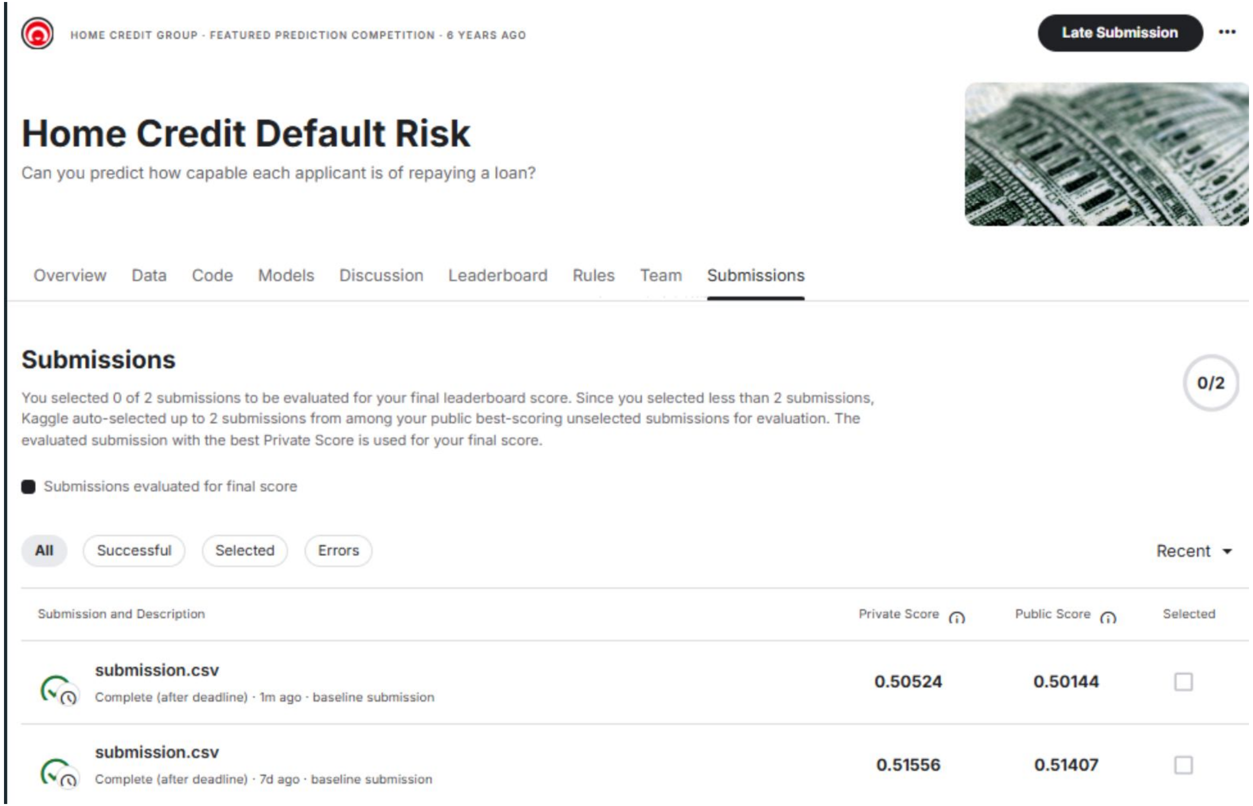
Experiment_Name	Hidden_Units	Dropout_Rate	Test_Accuracy	Test_Loss	Test_Precision	Test_Recall	Test_ROC_AUC	Test_F1
Sparse Input MLP	(64, 32)	NaN	0.9168	0.2605	0.0522	0.0094	0.7345	0.0156
Simple Deep MLP	(128, 64, 32)	NaN	0.9165	0.2645	0.0898	0.0178	0.7260	0.0285
MLP with Dropout	(128, 64, 32)	0.2	0.9175	0.2597	0.0082	0.0011	0.7372	0.0020
Wider MLP	(512, 256, 128)	NaN	0.9126	0.2804	0.1572	0.0417	0.7152	0.0621

Results and discussion



- MLP models performed comparably to other models but didn’t outperform them.
- The class imbalance remained a significant challenge, affecting minority class predictions.

Kaggle Submission



Conclusions

- Many of our models performed similarly, with and without feature selection
- MLP models performed comparably to other models but didn’t outperform them
- We struggled with class imbalance, which impacted our ability to model this data
- Our model of choice performs very well in predicting non-defaulters, as evident by the large number of true negatives and very few false positive, which might help people not being unfairly rejected

Next Steps

- From a consumer perspective we’re not unfairly rejecting applicants
- From a business perspective, we are incorrectly predicting applicants that might defect
- Our recommendation is to deal with the class imbalance by using methods like SMOTE (Synthetic Minority Oversampling Technique) to generate synthetic samples for the minority class or implementing cost-sensitive learning to penalize misclassifications of the minority class
- Some ensemble techniques like balanced bagging or boosting could also help

Challenges Faced

- Complex Dimensionality Reduction process
- Computational limitations of Google Colab even with Pro Account
- Challenges with the imbalance of Target variable
- Challenges with multiple people editing the same file