Boosting Credit-bility: Improving Predictive Performance of Credit Default using XGBoost

Riya Mhatre rmhatre@ucsd.edu

Emily Chin e1chin@ucsd.edu

Zaki Ahmed zahmed@ucsd.edu

Zhen Fang zhfang@ucsd.edu

Mentor: Rod Albuyeh
ralbuyeh@ucsd.edu

Abstract

This project explores the enhancement of credit default predictions through sophisticated data processing techniques applied to machine learning models. Contrasting with studies that rely on raw data, we emphasize the value of processed features in boosting the efficacy of these models. An ablation study is conducted to evaluate the impact of feature engineering by comparing the performance, specifically AUC scores, of a model trained on processed data against the same model using raw inputs and a traditional credit scoring approach. Our findings aim to highlight the pivotal role of data processing in refining the predictive accuracy of machine learning algorithms in the context of financial risk assessment.

Code: https://github.com/prod-lab/RiskNet

1 Introduction . . . 2
2 Methods . . . 3
3 Results . . . 6
4 Discussion . . . 8
References . . . 10

1 Introduction

The foundation of the banking sector rests on addressing a predictive challenge known as default prediction. This inquiry revolves around anticipating who will repay a loan and, conversely, who is likely to default on it. Presently, the banking industry relies on a metric called credit score—a single numerical value ranging from 350 to 850, determined by various factors like credit payment history, outstanding debt, and credit mix (Akin 2023). However, with the increasing accessibility of computing power, scholars and economists are turning to machine learning as a potential means to enhance the speed and accuracy of loan default predictions.

Contemporary machine learning (ML) research like that done by Alonso and Albanesi, seeks to demonstrate that ML models can effectively analyze raw banking data and swiftly generate reliable predictions (Alonso and Carbo 2021; Albanesi and Vamossy 2019). Nevertheless, many research inquiries in this field face challenges such as limited data access and the absence of shareable, packaged code, rendering them unreplicable. This paper endeavors to replicate previous efforts, placing a strong emphasis on open-source and replicable methodologies. This approach aims to assist fellow researchers by establishing a definitive benchmark illustrating the superior performance of XGBoost with robust features over credit/FICO score predictions. It also serves as a practical and clear starting point for our quarter 2 project.

In this investigation, we opted to utilize the Freddie Mac Single-Family Loan-Level dataset for three main reasons: data quality, robustness, and accessibility. The Freddie Mac (FM) dataset stands out for its data quality, being one of the largest and most reliable credit performance datasets for mortgages. In this context, ”high-quality” implies a high level of consistency, accuracy, and validity (Sakpal 2021). Major corporations and the United States Federal Housing Finance Agency rely on this dataset to address housing loan inequality (Freddie Mac 2024). Robustness, our second criterion, gauges the dataset's volume. The FM dataset is highly dimensional, encompassing 32 variables and millions of loan instances recorded annually from 1999 to 2023. This substantial size is crucial for addressing the default prediction challenge, as defaults are less frequent than non-defaulting loans in lending scenarios. Accumulating years of loan data enables our future machine learning models to offer a more nuanced representation of defaults. Moreover, the abundance of variables and observations allows us to (1) create multiple features and (2) achieve statistically significant results when comparing machine learning models. Lastly, aside from its high quality and robustness, the Freddie Mac dataset is freely accessible for download. Prioritizing accessibility, coupled with converting our code into a pip-installable package, ensures that our results can be easily reproduced and repeated.

2 Methods

2.1 Data

For our predictive loan default analysis, we initially utilized a dataset comprising 500,000 housing loan instances. Of this sample, 1.9% represented loan defaults, a relatively rare occurrence even considering the historical mid-2000s housing crisis from 2007 to late 2009. Focusing on data from the first quarter of 2009, post-crisis, our study incorporated two main components: a loan origination data file and a monthly loan performance dataset. The former provides static information such as Credit Score, Loan-To-Value ratio, Debt-To-Income ratio, and Interest Rate at loan origination, while the latter details individual loan performance over time, including payment history and delinquency status.

The loan origination dataset records static information such as Credit Score, Loan-To-Value ratio, Debt-To-Income ratio, and the Interest Rate at the time of loan origination while the second dataset contains details of each individual loan over time. Monthly performance data includes insights such as payment history and delinquency status for a specific loan, extending until the occurrence of a termination event or the conclusion of the selected timeframe.

Both datasets were merged on each loan's unique Loan Sequence Number, ensuring the retention of loans present in both datasets post-merge. Notably, termination events, denoted as ”zero balance codes,” and delinquency status emerged as pivotal features in the final combined raw dataset, as both serve as indicators of loan default.

2.2 Data Processing

It is crucial to define loan default in the context of the dataset. In the original dataset defaults were implicitly coded as specific “zero balance codes”- encoding why a loan was terminated- as well as a feature describing the delinquency status of loans over time. From these two features, a default feature was generated as any loans with a zero balance code out of 3, 9, and 6 or loans that were more than 90 days delinquent. 90 days was primarily chosen due to our timeframe being a financial quarter in which a loan can only be delinquent for a maximum of 91 days. More generally, 90 days was also selected because a borrower who is 90 days delinquent is already three payments behind, a situation typically deemed as a loan default by servicers.

During data processing, null values were mapped to recurrent 9's (e.g. “999” or 99) to avoid future issues with possibly disparate behavior on missing data. Features with a high proportion of missing data and features were automatically removed.

We employed a time-series split for the training, testing, and validation data. Specifically, the model was trained on the earliest data, which — when using Q1 2009 data — was from January and February, while the more recent data from February and March was used for validation and testing, respectively. This is standard operating procedure for working with time series data to prevent data leakage and ensure the model is capturing useful information (and therefore can predict future instances). Specifically, we ensured that the testing set is always the last/most recent 1,000 data points. This ensures comparability in results for every model.

Additionally, for our latter two models (described in the Models section later), we ran a Parquet conversion function on the original text file based dataset to alleviate runtime issues. While the Parquet conversion itself required an expensive operation to convert our data to Parquet, the overhead runtime cost was deemed acceptable for faster read operations during the model pipeline. For our last model feature-engineered features are added to the dataset.

2.3 Model

The selection of the XGBoost algorithm as the primary model stems from its demonstrated effectiveness in handling tabular time series data and capturing evolving patterns crucial for loan default prediction. During our preliminary testing of various machine learning models, XGBoost consistently outperformed others, showcasing its prowess in iteratively generating decision trees to correct overall prediction errors. As a gradient boosting algorithm, XGBoost mitigates the risk of overfitting to the training data and has been widely acknowledged for its superior predictive performance in diverse domains. We specifically chose XGBoost for its ability to handle the complexity of credit data and extract valuable insights from a dataset with multiple features. The interpretability of the model's decision-making process, coupled with its high predictive accuracy, made XGBoost the optimal choice for our study. Additionally, its popularity and widespread use in the machine learning community contribute to the reproducibility and comparability of our results with existing literature.

In this study, we trained 4 models, a baseline model and 3 others which used a combination of engineered features and utilization of parquet data storage.

The first model, which was our baseline model, was trained only using credit score as an input. In other words, this machine learning model demonstrated the trustworthiness of credit score as a predictor. Isolating credit score as the only input for the baseline model clarifies the value of additional processed features in enhancing prediction reliability. By including the first baseline of only credit score, we can contextualize how large the impact of additional (and engineered) features is against a regular machine learning model.

Our next model used the same architecture as the first, but included all relevant features provided by the Freddie Mac dataset.

Our third model used the same architecture as the second, but added the use Parquet as the primary data medium for runtime performance uplift.

Finally, our last model included all advantages of our third model, but also built feature engineered features for comparison purposes.

2.4 Performance Metrics

The selection of the Area Under the Receiver Operating Characteristic curve (AUC-ROC) as our performance metric is rooted in the inherent characteristics of our loan default prediction problem. Given the highly imbalanced nature of our dataset, where defaults are infrequent compared to non-defaults, precision and accuracy metrics would not be suitable as they treat false positives and false negatives equally. AUC-ROC, on the other hand, provides a comprehensive evaluation of the model's ability to discriminate between positive and negative instances across different probability thresholds. By focusing on the trade-off between true positive rate and false positive rate, AUC-ROC captures the model's overall performance regardless of the chosen decision threshold. This metric is particularly valuable in scenarios where the cost of false positives and false negatives is uneven, as is often the case in credit default prediction. The consistent AUC performance across training, validation, and testing sets in our study reflects the robustness of XGBoost in handling imbalanced datasets and reinforces the reliability of our model in predicting loan defaults. Additionally, AUC performance on a static test set is used to compare incremental changes such as adding automated feature engineering to the pipeline from baseline to final model.

Another performance metric that was measured was class precision. Precision was calculated as the number of true positives divided by the total predicted and true positives. This metric was chosen for two reasons: (1) for better programming evaluation, and (2) for more relevant insight in terms of banking cost. Firstly, due to the class imbalance, instances of "positives" (defaults) are very rare, accounting for about 1.9% of the overall dataset. Measuring the performance in relation to positive predictions is therefore important to compare against AUC, which considers both positive and negative predictions. Secondly, precision was measured for each model because this metric is especially significant to banking institutions. A false positive represents a charge-off: when the bank loans out money and does not recover it in full, therefore losing money. This metric will help banking institutes contextualize performance.

The final performance metric that was measured was time. Specifically, we measured time taken to train the parameters and hyperparameters of the model, assuming data loading and encoding took minimal time. Time acts as a proxy variable for cost to run the model(s).

2.5 Development Environment

The pipeline was developed on Python 3.10 with the key dependencies being Numpy 1.26.1 (Harris et al. 2020) and Pandas 2.1.2 (Pandas 2020). The development environment was UC San Diego's Data Science/Machine Learning Platform (DSMLP), which provides a Docker container running on a Linux operating system. This platform supports a range of languages and GPU-enabled frameworks. To ensure consistency and reproducibility, the project was housed within a virtual environment in a server configured with 8 CPU cores and 16 GB of memory. 32 GB of memory was utilized for overhead concerns during development especially for debugging. The configuration file accompanying the project code provides a comprehensive list of all other dependencies.

3 Results

After implementing and running the four different XGB models, the team discovered that increasing the size of the training data, either through Parquet loading or through feature engineering, led to improved performance of the models. As mentioned in Methods, we assessed the predictive accuracy of a machine learning model in comparison to credit score using ROC AUC as a metric. We opted for this metric as our preferred performance measure due to the dataset's pronounced imbalance.

The models visualized below are, from left to right, as follows. Firstly, in blue, the baseline model was trained using credit score as the sole feature. The purpose of this was to show how effective XGBoost is at leveraging that one metric to make decisions. Next, in orange, a second model was run using all the same XGBoost training and hyperparameter architecture, except this time, all the features in the dataset were utilized. The next area of improvement was the model's speed. To make the model faster, we converted the data files from txt files to Parquet. Parquet files allow data to be ingested at a faster rate and with lower memory cost. Because of this feature, we were able to load all the data rather than a sample of it, which helped improve the model performance. The implementation of Parquet was added to the third model, in green. Finally, the last thing we added to the model was feature engineering using the featuretools package. Featuretools is an automated program that performs Deep Feature Synthesis (DFS), which involves generating encodings, means, modes, etc. from the features in the dataset (Featuretools 2024). While this increases the data's dimensionality, it helps with breaking the data down, making it easier for XGBoost to comprehend. This, we assumed, would lead to a better performance. The model that included feature engineering is seen on the far right, in red.

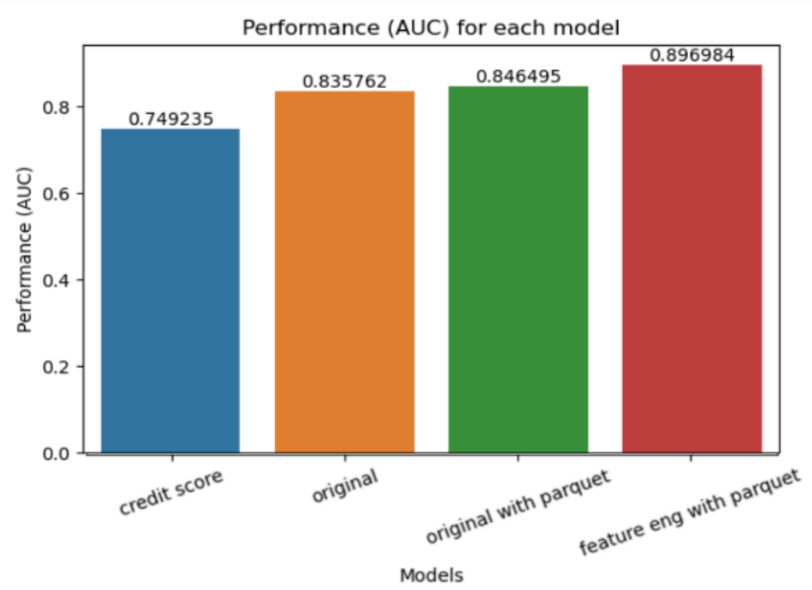

Figure 1: Performance as measured in ROC-AUC

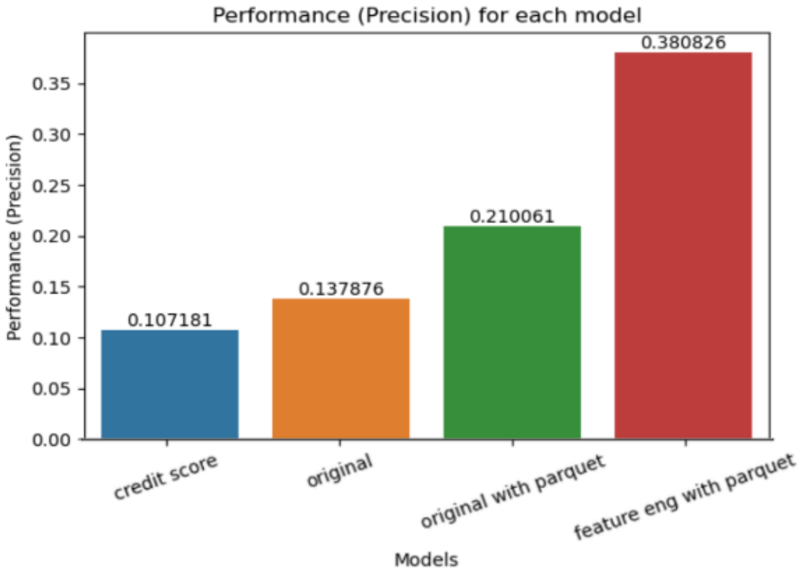

Figure 2: Performance as measured in simple precision

The results show that as more data was added to the model, performance improved. Specifically, we can see on Figure 1 and Figure 2 that the AUC and prediction values increased. As seen in Figure 1, the final model's AUC with feature engineering and parquet was 15% more than that of the baseline model, with AUC increasing somewhat linearly with each update. In contrast, precision increased almost exponentially moving from the baseline precision to the final model precision (Figure 2).

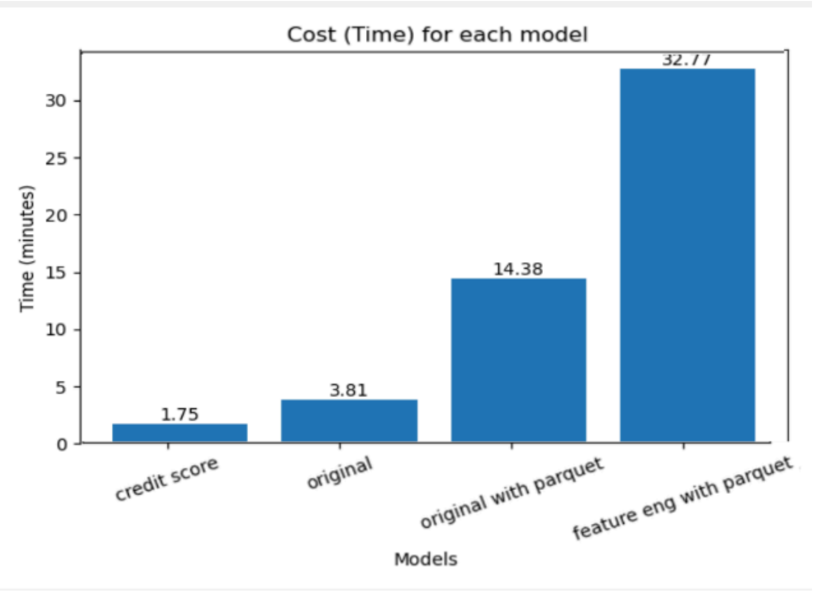

Figure 3: Cost of model as measured in time (minutes)

While the precision and AUC scores increased with complexity, we noticed that time also followed a similar trend. When evaluating the time necessary to initialize, train, and evaluate each model, we realized that as we added more features, the time increased.

As seen in Figure 3, time increased almost exponentially as the model and data became more complex. This is expected considering the addition of extra features would make the model processes, specifically data loading, more costly. In addition, implementing feature engineering and pre-loading the txt file into Parquet both added extra steps in the training process.

The time in minutes increased a factor of almost 30 when comparing the baseline credit score model to the final model. This is in contrast to a 3x increase seen in precision as we iterated through models. This suggests the time — which serves in our research as a proxy for cost — grows more steeply than precision as data dimensionality increases.

Overall, the model iterations did lead to a large increase in precision and AUC, making the model a better predictor than merely using credit score. However, the models also became more costly to use. We will discuss the implications of our findings in the next section.

4 Discussion

Our research emphasizes the importance of sophisticated data processing and feature engineering in improving the accuracy and reliability of credit default predictions using machine learning models, specifically XGBoost. Through comprehensive analysis, we have demonstrated that the integration of these advanced techniques substantially enhances model performance. It is our hope that this study will provide valuable insights for financial risk assessment and inspire further efforts to optimize the performance of machine learning applications in the domain of credit lending.

In terms of feature engineering, our study substantiates the significant role it plays in enhancing the predictive capabilities of machine learning models for credit default prediction. By employing advanced data processing techniques, such as Deep Feature Synthesis (DFS) via the featuretools package, we not only increased the dimensionality of our model but also facilitated a deeper understanding of the dataset by XGBoost (Featuretools 2024). This approach significantly improved our model's performance, as evidenced by a 15% increase in AUC and a nearly exponential rise in precision when compared to the baseline model that utilized only the credit score as a feature. This demonstrates the clear advantage of incorporating processed features over relying solely on raw data.

From a technical perspective, our journey to augment predictive performance uncovered a pivotal trade-off between accuracy and computational efficiency. The introduction of an intricate array of features and the application of advanced data processing techniques inevitably extended the model's training and evaluation timeframe. This observation emphasizes the necessity of balancing model precision with computational pragmatism, especially in scenarios where real-time predictions are paramount. Future research directions should thus pivot towards minimizing this computational burden without sacrificing the quality of predictions. Potential avenues include exploring lightweight yet powerful feature engineering methods, adopting more efficient data storage and processing formats, or investigating incremental learning approaches that adapt to new data without retraining from scratch. Moreover, considering the dynamic and ever-evolving landscape of financial markets, it becomes imperative to embed mechanisms for continuous model evaluation and adaptation, ensuring the sustained relevance and accuracy of the predictive models.

Within the context of banking institutes, our findings are exciting and significant. Discovering and visualizing the Pareto curve of performance versus cost (time) for XGB model prediction could open up valuable and thought-provoking questions for policy makers and bank managers. Our study could be used as a benchmark of how and where machine learning can be implemented in credit risk prediction cases. Furthermore, since our code is available for public usage, we hope to see more contributions and enhancements to our project in the future. This could revolutionize public involvement in the banking institutes that significantly impact our lives.

The practical implications of our findings for the financial sector cannot be overstated. By demonstrating the enhanced predictive capabilities through our methodology, we advocate for a shift towards more data-driven approaches in credit risk assessment. Such advancements could not only improve the accuracy of default predictions but also enable more personalized, fair, and efficient credit lending practices. Ultimately, as our models grow more sophisticated and attuned to the complexities of financial behaviors, it becomes an increasingly attractive tool that can more accurately provide mortgages to those who deserve them. Together, we can forge a future that leverages the cutting edge of machine learning to foster a more resilient and equitable financial ecosystem.

References

Akin, Jim. 2023. “What Affects Your Credit Scores?” *Experian*. [Link]

Albanesi, Stefania, and Domonkos F. Vamossy. 2019. “Predicting consumer default: A deep learning approach.” Technical Report, National Bureau of Economic Research. [Link]

Alonso, Andrés, and Jose Manuel Carbo. 2021. “Understanding the performance of machine learning models to predict credit default: a novel approach for supervisory evaluation.” [Link]

Featuretools. 2024. “featuretools.dfs.” February. [Link]

Freddie Mac. 2024. “Single-Family Loan-Level Dataset General User Guide.” January. [Link]

Harris, Charles R., K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. 2020. “Array programming with NumPy.” *Nature* 585 (7825): 357–362. [Link]

Pandas. 2020. “pandas-dev/pandas: Pandas.” February. [Link]

Sakpal, Manasi. 2021. “How to Improve Your Data Quality.” *Gartner*. [Link]