Machine Learning with Applications in Finance - COMP0050 Coursework

Avlonitis Ektor

March 2024

Task 1

Introduction

The study focuses on constructing a predictive model capable of determining the likelihood of bank defaults. Given the complexities of financial datasets and the critical nature of accurate default predictions, the study explores a variety of methodological approaches. By employing and comparing different machine learning techniques the goal is to identify the method that provides the highest accuracy in predicting defaults.

Methodology

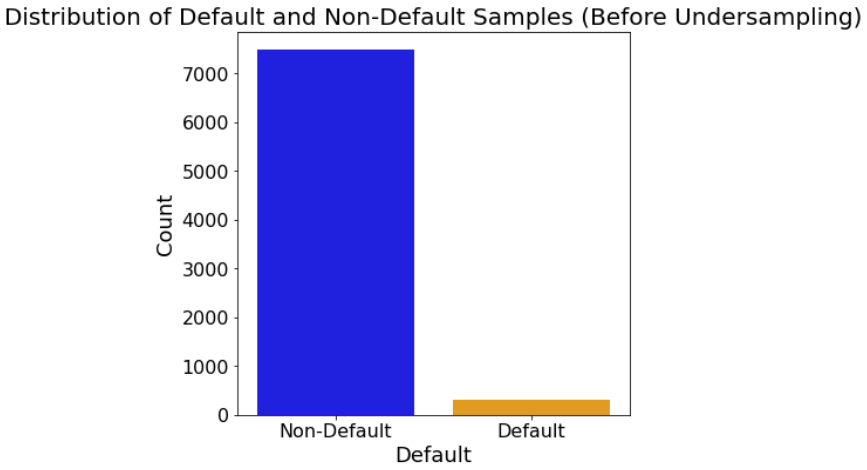

Figure 1: Distribution of Default and Non-Default Samples

In the analysis of the dataset concerning default and non-default samples, it is observed that there is a significant imbalance between the two classes. Specifically, the number of non-default samples significantly exceeds that of default samples, as seen in Figure 1. This imbalance presents a challenge for machine learning algorithms, as it can lead to a bias towards the majority class, in this case, the non-default samples. One effective approach to address this problem is through the process of undersampling. Undersampling involves reducing the number of samples in the majority class to match the number of samples in the minority class. By implementing undersampling, it can be ensured that the algorithm does not become biased towards predicting non-default cases, thus improving its ability to accurately identify default cases as well.

Next, it's important to select the appropriate features for the machine learning model. The "Default" column is excluded from the features set as it serves as the target variable for the predictive modeling. Additionally, the columns *Loans secured by 1-4 family residential properties* (Column 3) and *Banks debt* (Column 15) are also removed from the feature set. The decision to exclude these columns is based on observations from the correlation matrix seen in the appendix. These features have extremely high correlations with most of the other features in the dataset. This multicollinearity can degrade the performance of some machine learning algorithms by making the model training process less efficient and the model itself less interpretable.

Results

The models used for the analysis and subsequently compared were Logistic Regression, Decision Trees, Random Forest, and Neural Networks. These diverse approaches were evaluated to identify the most effective method for the given dataset, taking into account various performance metrics such as accuracy, precision, recall, F1 score, and ROC AUC.

Logistic Regression

The different Logistic Regression models used were: Logistic regression without regularization parameter, logistic regression with ridge regularization, logistic regression with lasso regularization and logistic regression with PCA (preprocessing the features with Principal Component Analysis).

Table 1: Logistic Regression Results

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC
Ridge Regularization	66.30%	81.13%	45.26%	58.11%	73.92%
PCA	59.78%	70.59%	37.89%	49.32%	74.30%
No Regularization	67.93%	80.00%	50.53%	61.94%	72.28%
Lasso Regularization	65.22%	78.18%	45.26%	57.33%	72.58%

It is observed from Table 1 that the Logistic Regression models exhibit varying degrees of effectiveness across different metrics, such as Accuracy, Precision, Recall, F1 Score, and ROC AUC. Specifically, the model with Ridge Regularization demonstrates a balanced performance with a relatively high Precision of 81.13% and also high ROC AUC of 73.92% among the logistic regression tests, indicating its effectiveness in distinguishing between the classes. The model without regularization shows the best Accuracy (67.93%) and F1 Score (61.94%), suggesting its proficiency in general prediction tasks and balance between Precision and Recall. Meanwhile, models employing PCA and Lasso Regularization exhibit lower performance across most metrics, with PCA showing the lowest Accuracy and F1 Score, which might be attributed to the reduced feature space impacting the model's ability to capture the complexity of the data. Overall, the choice of regularization technique and the decision to use PCA significantly impact the model's performance.

Decision Trees

The different Decision Tree models used were: Standard decision tree (without any modification to its default parameters), decision tree with PCA (preprocessing the features with Principal Component Analysis), and decision trees with best parameters. Using Grid Search, the best parameters for Decision Trees were identified as follows:

Table 2: Best Parameters for Decision Trees

Parameter	Value
max_depth	3
max_features	None
min_samples_leaf	1
min_samples_split	2

Table 3: Decision Tree Model Results

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC
Decision Tree	70.65%	73.03%	68.42%	70.65%	70.73%
Decision Tree (best parameters)	72.83%	73.20%	74.74%	73.96%	77.45%
Decision Tree with PCA	73.91%	80.52%	65.26%	72.09%	74.20%

It is observed from Table 3 that the application of PCA in Decision Tree models significantly improves performance metrics compared to the standard Decision Tree and even the model tuned with the best parameters. The choice of 5 components for PCA yielded the best results in terms of model performance. The Decision Tree with PCA achieves the highest Accuracy (73.91%) and Precision (80.52%), demonstrating its superior capability to classify instances correctly and its effectiveness in predicting positive instances as positive. While the Recall is slightly lower than that observed in the Decision Tree with the best parameters, the F1 Score and ROC AUC values are notably competitive, indicating a balanced performance between precision and recall and an excellent ability to distinguish between classes. These results highlight the positive impact of dimensionality reduction through PCA on the model's predictive power.

Below, the decision tree with best parameters is visualized, showing how it classifies banks based on their features.

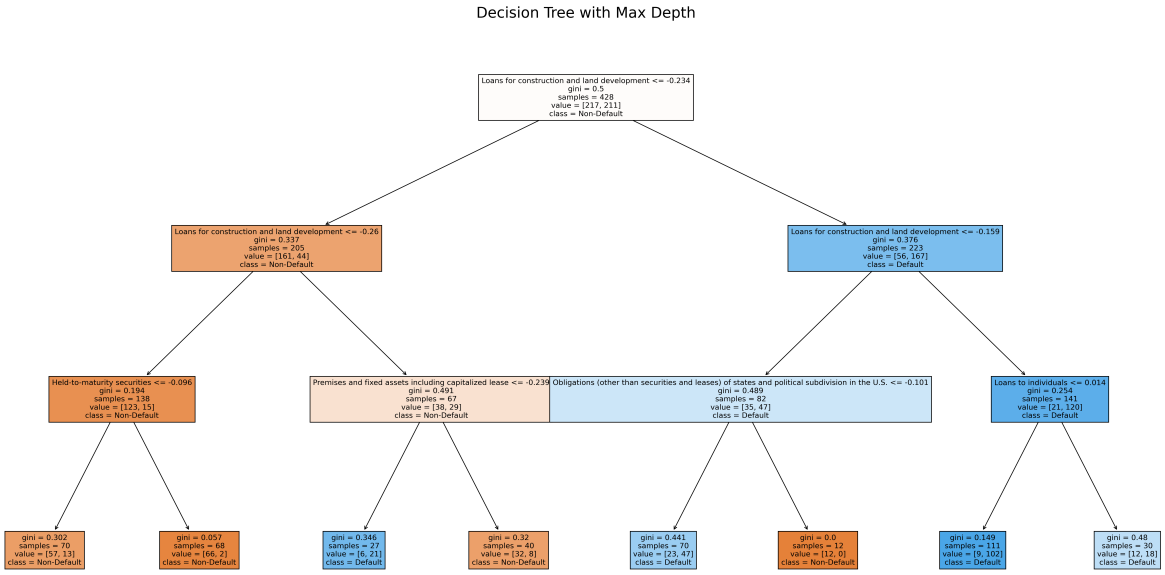

Figure 2: Decision tree with best parameters

Random Forest

The different Random Forest models used were: Standard random forest (without any modification to its default parameters), random forest with PCA (preprocessing the features with Principal Component Analysis), and random forest with best parameters. Using Grid Search, the best parameters for Random Forest were identified as follows:

Table 4: Best Parameters for Random Forest

Parameter	Value
max_depth	10
max_features	auto
min_samples_leaf	2
min_samples_split	5
n_estimators	100

Table 5: Random Forest Model Results

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC
Random Forest	72.28%	75.00%	69.47%	72.13%	82.24%
Random Forest (best parameters)	73.37%	75.56%	71.58%	73.51%	82.01%
Random Forest with PCA	76.63%	83.33%	68.42%	75.14%	80.03%

From Table 5, it is observed that applying PCA to the Random Forest model enhances its Precision to 83.33%, the highest among the variants, indicating a strong ability to correctly predict positive instances. The choice of 4 components for PCA yielded the best results in terms of model performance. However, this comes with a slight trade-off in Recall and ROC AUC compared to the model with the best parameters, suggesting a strong impact of dimensionality reduction. While the PCA-enhanced model demonstrates competitive Accuracy and F1 Score, the best parameters without PCA slightly outperform in overall model effectiveness. This underscores the importance of balancing dimensionality reduction and parameter optimization to achieve optimal model performance.

Neural Network

Using Grid Search, the best parameters for the Neural Network were identified as follows:

Table 6: Best Parameters for Neural Network Keras Classifier

Parameter	Value
Activation	relu
Batch Size	20
Epochs	100
Layers	2
Neurons	10

Table 7: Neural Network Keras Classifier Test Set Metrics

Metric	Value
Accuracy	76.63%
Precision	81.71%
Recall	70.53%
F1 Score	75.71%
ROC AUC	84.53%

Table 7 shows the Neural Network Keras Classifier achieving a solid performance with an accuracy of 76.63% and a precision of 81.71%. Its F1 Score and ROC AUC of 75.71% and 84.53%, respectively, indicate a balanced and effective model in distinguishing between classes, highlighting the success of the optimized parameters.

All Models

Table 8: Best Performing Models Across Different Approaches

Approach	Model	Accuracy	Precision	Recall	F1 Score	ROC AUC
Logistic Regression	No Regularization	67.93%	80.00%	50.53%	61.94%	72.28%
Decision Trees	with PCA	73.91%	80.52%	65.26%	72.09%	74.20%
Random Forest	with PCA	76.63%	83.33%	68.42%	75.14%	80.03%
Neural Network	Keras Classifier	76.63%	81.71%	70.53%	75.71%	84.53%

Table 8 compares the top-performing models across various approaches, demonstrating their effectiveness in classification tasks. The Neural Network Keras Classifier stands out with an accuracy of 76.63% and the highest ROC AUC value of 84.53%, indicating its superior ability to distinguish between classes. Additionally, it achieves the second highest precision rate of 81.71%, reflecting its accuracy in identifying positive instances. Both the Decision Trees and Random Forest models, enhanced with PCA, also display great performances, especially in terms of precision and F1 scores, showcasing PCA's role in refining model predictions.

For the Random Forest model enhanced with PCA, the ROC curve below offers a visual representation of its classification performance.

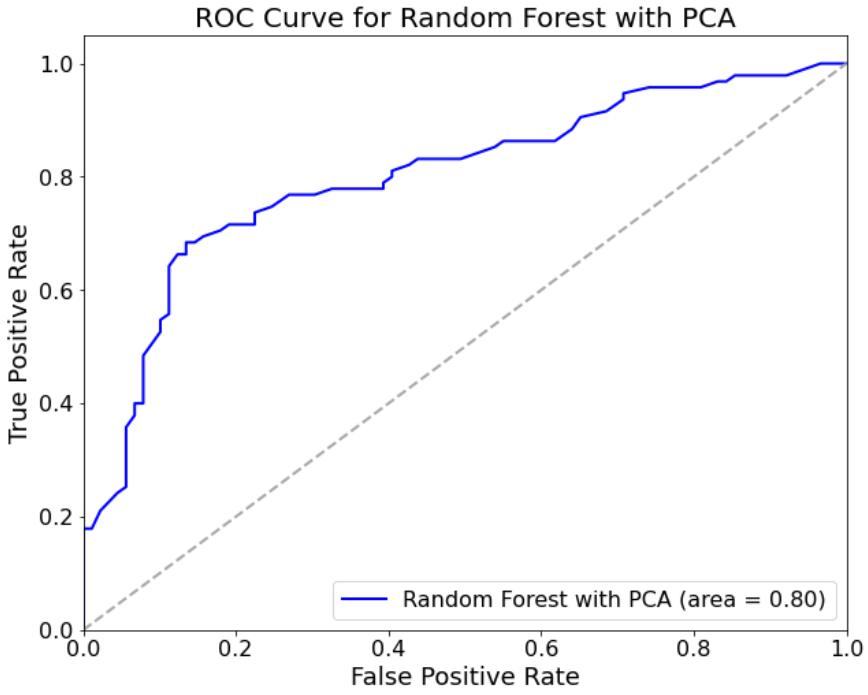

Figure 3: ROC Curve for Random Forest with PCA

This study encounters limitations due to the imbalanced nature of the dataset and the potential loss of predictive information from excluding features based on multicollinearity. Future work could improve upon these areas by employing sophisticated imbalance correction techniques like statistical advanced techniques and exploring more advanced feature selection methods. Additionally, integrating complex models such as deep learning could better capture the dataset's patterns and thus improve the predictive accuracy.

Task 2

Introduction

This study examines the daily returns of 48 industry portfolios, ranging across diverse sectors. The objective is to determine the optimal portfolio composition across different time windows, analyzing how portfolio variance is influenced by the choice of weighting scheme and the implications for portfolio performance.

Methodology

Data Analysis & Processing

The data was organized into two key datasets: one based on market-capitalization weighted returns and another on equal-weighted average returns, covering the period from July 1, 1926, to October 31, 2017. The market-cap weighted dataset was chosen for analysis, which includes the influence of larger companies within each sector on the portfolio's performance. During the preprocessing of the dataset, missing values were addressed using the forward fill method. This approach propagates the last observed non-null value forward until another non-null value is encountered, thus maintaining continuity in the dataset's time series without introducing bias.

Portfolio Optimization

The project aims to identify the optimal asset weights that minimize the total variance of the portfolio, over three distinct time windows: 5-Year, 10-Year, and 20-Year. This involves calculating the covariance matrix from historical returns and solving for the weight distribution that minimizes portfolio variance under the constraint that the sum of weights equals 1. The following equation was used:

$$\text{Portfolio Variance} = \mathbf{w}^T \Sigma \mathbf{w} \tag{1}$$

as initially proposed by Markowitz. (Markowitz, 1959)

Regularization and Constraints

Regularization introduces a penalty term to the optimization process, which was optimally calculated using an 80% training set and validated on the remaining 20%. This way historical variance was minimized and the validation helped in reducing overfitting. Including regularization the equation is the following:

$$\text{Regularized Portfolio Variance} = \mathbf{w}^T \Sigma \mathbf{w} + \lambda \sum \mathbf{w}^2 \tag{2}$$

which reflects the approach by DeMiguel et al. (DeMiguel et al., 2007)

Ban on Short Selling

Applying a ban on short selling creates an additional constraint: all portfolio weights must be non-negative. This change limits investment strategies to long positions, increasing the portfolio's risk profile by removing the ability to hedge against downturns. (Jagannathan and Ma, 2003)

Results

The optimal portfolio was computed for different lengths of training time series.

Time Window	**5-Year**	**10-Year**	**20-Year**
Total Variance	0.3231	0.5376	0.5840
Agriculture	0.0462	0.0191	0.0716
Food	-0.0983	0.1792	0.2430
...	...	...	...
Other	0.1606	0.1978	0.0921

Table 9: Optimal Portfolio Weights and Total Variance for Different Time Windows

Following the table, Figure 4 offers a bar representation of the optimal portfolio weights across all sectors for the entire dataset. This representation helps to understand the distribution of weights better.

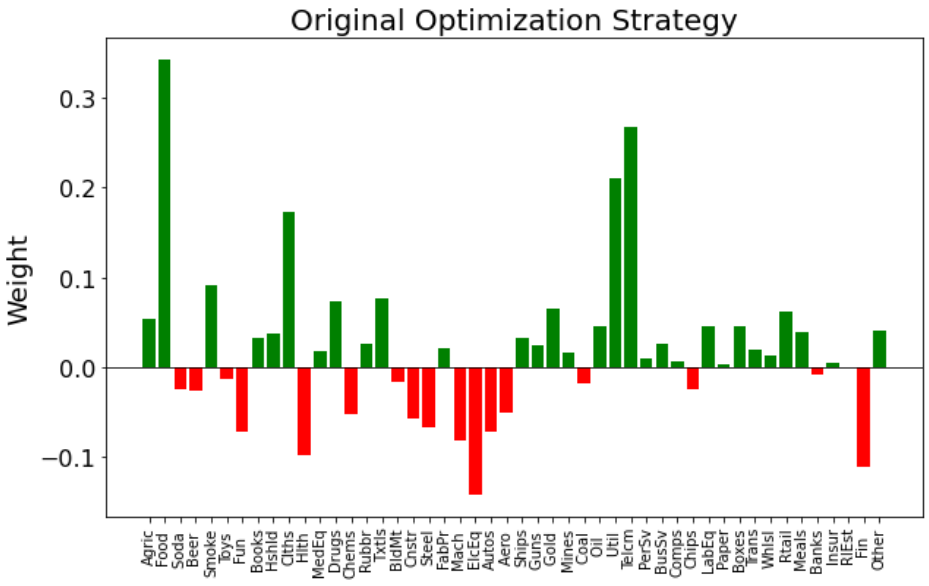

Figure 4: Optimal portfolio weights

As seen in Table 9 the total portfolio variance increases as the time window extends from 5 years to 20 years, from 0.3231 to 0.5840. This suggests that longer time windows, which incorporate more historical data, lead to portfolios with higher variance, indicating greater risk. One possible reason for that is that longer time windows encompass a broader range of market conditions, including various economic cycles such as expansions, recessions, booms. This wider range of conditions can lead to a greater historical volatility being captured in the covariance matrix used for portfolio optimization.

Larger companies have a higher influence on the industry average returns in a market-cap weighted dataset like this one. The total variance of the industry will be greatly impacted if these big businesses are highly volatile. In contrast, equal-weighted averages may reduce the influence of extremely volatile large corporations because every company, regardless of size, contributes equally to the industry average.

Regularization effect

Next, a regularization scheme was tested on the entire dataset. The effect of applying regularization, especially with an optimally tuned λ, reflects a balance between minimizing historical variance and avoiding overfitting to past returns. The portfolio constructed was not only optimized based on historical data but also more likely to perform well in the future under different market conditions. The optimized parameter is: $\lambda_{\text{opt}} = 0.1$

Asset	Weight
Asset 1	0.0485
Asset 2	0.2147
Asset 3	0.0070
...	...
Asset 47	-0.0758
Asset 48	0.0226

Table 10: Optimal Portfolio Weights with Regularization Parameter $\lambda = 0.1$

The following bar chart visualizes the optimal portfolio weights across the entire dataset after applying regularization with $\lambda = 0.1$.

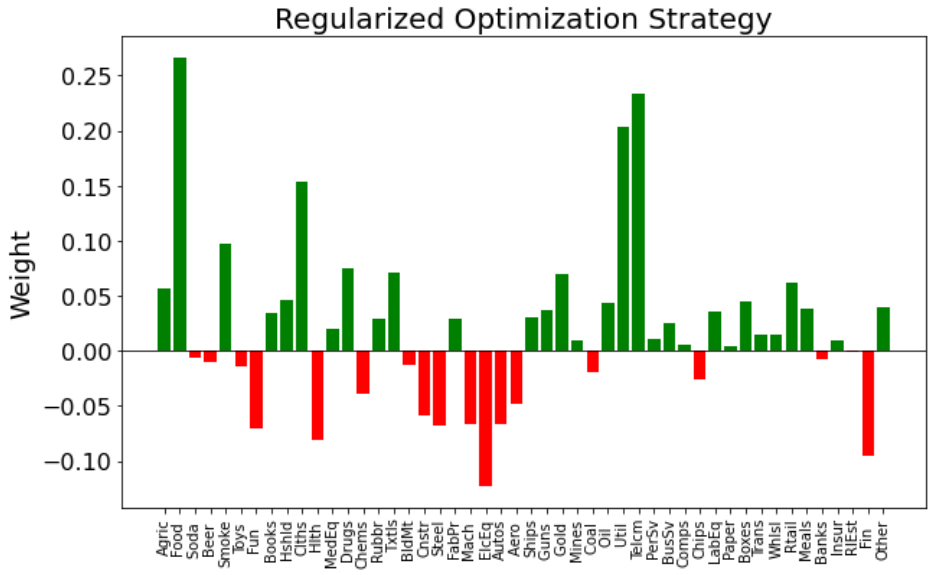

Figure 5: Optimal portfolio weights with regularization

As observed from Table 10 and Figure 5, the implementation of a regularization scheme with a regularization parameter (λ) set to 0.1 significantly influences the portfolio optimization process. This regularization approach leads to a well-diversified portfolio by ensuring that no single asset's

weight is excessively high, and thus reducing the risk of overweighting in few assets. The portfolio now is less likely to underperform and also includes an investment strategy that is better suited for varying future market conditions.

Ban on Short Selling

When a ban on short selling is applied to the un-regularized portfolio optimization problem, it changes the constraints of the optimization process by restricting the portfolio weights (x) to be non-negative. This means that all investments must be long positions. Below is a table with the results obtained from the portfolio optimization process with and without a ban on short selling for the entire dataset:

Metric	With Short Selling	Without Short Selling
Optimization Status	Successful	Successful
Portfolio Variance (fun)	0.4799	0.6589
Number of Iterations (nit)	15	11
Number of Function Evaluations (nfev)	751	544
Number of Jacobian Evaluations (njev)	15	11

Table 11: Comparison of Portfolio Optimization Results

Following the table, the bar chart shows:

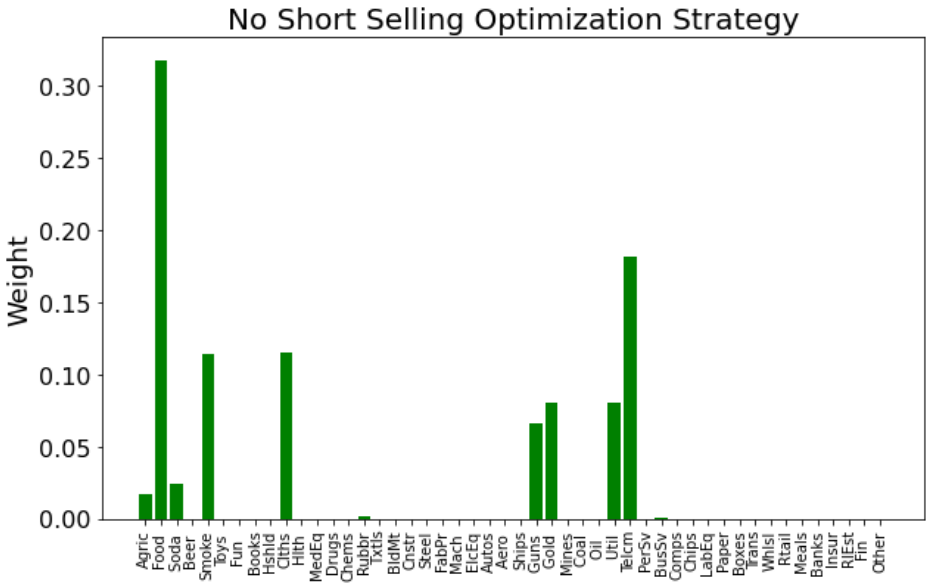

Figure 6: Optimal portfolio weights with ban on short selling

Notable shifts are observed in strategy and risk profile. The portfolio now maintains exclusively non-negative weights, eliminating the capacity to gain from any decreases in asset values. This adjustment leads to a heightened focus on specific assets, increasing the portfolio's overall risk, as indicated by the higher portfolio variance observed.

The reliance on historical data for optimizing weight allocations in portfolios is limited by its inability to fully predict or account for sudden market shifts or broader economic changes. Future research work could include adopting dynamic models that anticipate market conditions more

accurately, using broader economic indicators. Moreover, considering maximizing performance metrics, such as the Sharpe ratio or incorporating ESG criteria, could yield more sophisticated investment strategies. This approach would ensure that the weight allocation process is more relevant and adaptive to current market realities.

Bibliography

DeMiguel, V., Garlappi, L., & Uppal, R. (2007). Optimal versus naive diversification: How inefficient is the 1/n portfolio strategy? *Review of Financial Studies*, *22*(5), 1915–1953.

Jagannathan, R., & Ma, T. (2003). Risk reduction in large portfolios: Why imposing the wrong constraints helps. *The Journal of Finance*, *58*(4), 1651–1683.

Markowitz, H. (1959). *Portfolio selection: Efficient diversification of investments*. Wiley.

Appendix

This appendix provides further insights into the selection of features for the machine learning model. The decision to exclude certain features was based on their high correlation with other variables in the dataset, as demonstrated in the correlation matrix below.

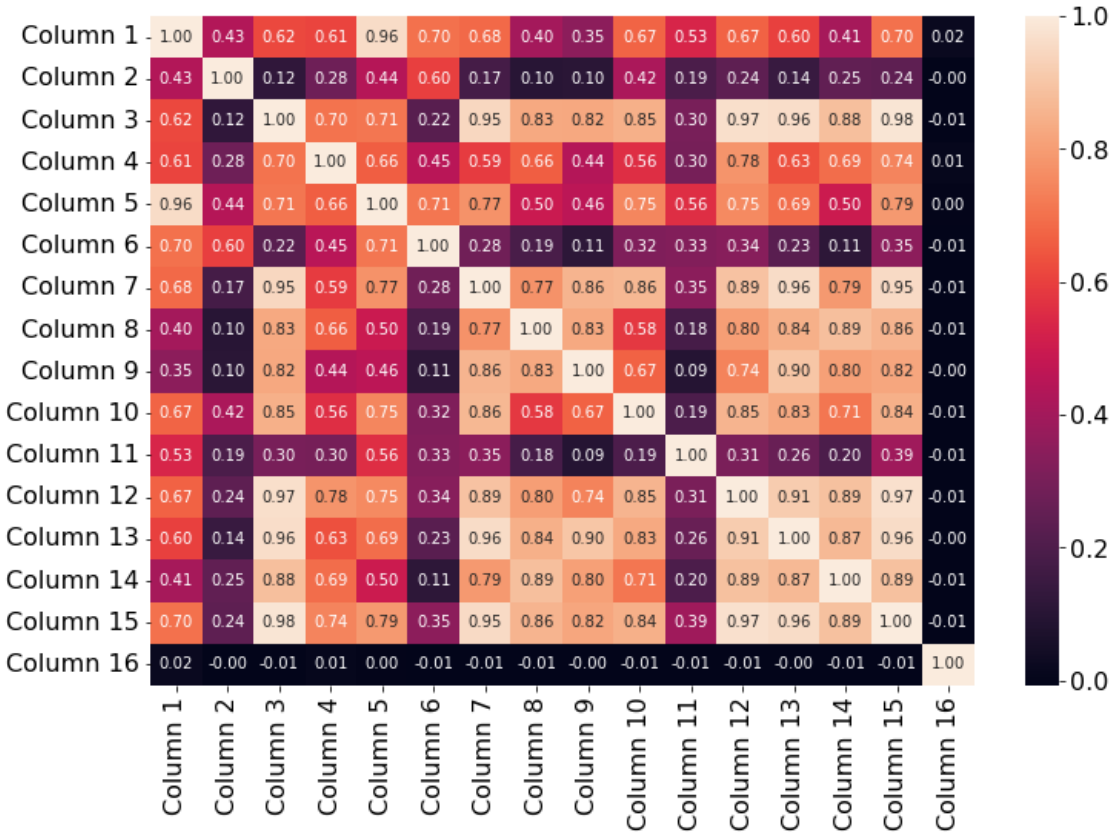

Figure 7: Correlation Matrix

The correlation matrix (Figure 7) highlights the rationale for removing the features related to Column 3 - *Loans secured by 1-4 family residential properties* and Column 15 - *Banks debt* due to their multicollinearity with other variables.