TITLE

Analysis of Corporate Bankruptcy Trends

GROUP 6

TEAM MEMBERS

Hrithik Sarda
Shaswat Sinha
Pragya Naruka
Tanishka Adhlakha
Valli Meenaa Vellaiyan

IE7300 36697 Statistical Learning for Engineering
SEC 02

ABSTRACT

This project aims to address the critical challenge of bankruptcy prediction among different companies by leveraging machine learning techniques as well as financial data analysis. The focus lies on developing a predictive model that can accurately forecast the likelihood of bankruptcy based on historical financial data. Ultimately, the project endeavors to empower stakeholders and decision-makers with actionable insights to proactively mitigate bankruptcy risks and safeguard the financial stability of different companies.

The dataset contains financial data from the Emerging Markets Information Service (EMIS) database, focusing on bankruptcy prediction for companies operating between 2000 and 2013, primarily in Poland. It aims to differentiate between bankrupt and non-bankrupt companies. The classification cases span five forecasting periods, from the 1st to the 5th year, within which instances are categorized based on bankruptcy status.

The dataset includes 64 real-valued features (X1 to X64) representing various financial ratios and indicators for the Polish companies, including profitability indicators like return on assets (ROA) and operating profit rate, liquidity measures such as current and quick ratios, and leverage ratios like debt ratio and interest coverage ratio. It also includes growth rates, turnover ratios, and other performance indicators related to asset management, revenue generation, and financial stability. These features provide a comprehensive view of the companies' financial health and operational efficiency.

We developed four machine learning algorithms, which are Logistic Regression, Hard-Margin SVM, a Neural Network implemented via TensorFlow and Keras, and an Ensemble model. We evaluated each model based on precision, recall, accuracy, and other relevant metrics.

Our focus was to assess predictive power and computational efficiency. The models' performances provided insights that could aid in finding bankruptcy prediction of various companies.

INTRODUCTION

Business Problem Definition:

The financial stability of companies is under constant threat from various internal and external factors. The ability to predict bankruptcy is a crucial competency for stakeholders and decision-makers across industries. The main challenge lies in early identification of financial distress signs before they evolve into irreversible bankruptcy situations. By addressing this challenge, companies can take timely corrective actions to mitigate risks.

This project is designed to develop a sophisticated machine learning model capable of analyzing vast datasets of historical financial data to predict the likelihood of bankruptcy. The predictive model aims to provide a reliable forecast that will help stakeholders make informed decisions to prevent financial collapse and maintain economic stability.

The project takes a careful approach to model building and explores machine learning algorithms that are created independently of pre-built libraries. The successful implementation of this project would result in a significant reduction in unexpected financial losses and allow companies to navigate the complexities of financial management more effectively. The overarching goal is to enhance predictive capabilities that empower businesses to foresee financial downturns and adjust their strategies proactively, ensuring long-term resilience and sustainability.

Problem Setting:

The project is centered around the development of a binary classification model using machine learning. The primary objective is to predict whether a company will go bankrupt or not, indicated by a binary target variable where '0' represents non-bankruptcy and '1' indicates bankruptcy.

The original predicted column in the dataset, which combines the year of potential bankruptcy and its occurrence, has been split into two distinct columns:

1. *A categorical column* indicating the years until potential bankruptcy (values like 5, 4, 3, 6, etc., corresponding to the number of years).
2. *A binary column* serving as the target variable for the model, indicating the occurrence of bankruptcy ('0' for non-bankruptcy, '1' for bankruptcy).

The focus of the model is to analyze historical financial data to accurately classify companies into one of two categories: likely to go bankrupt or likely to remain solvent. This classification will enable stakeholders to identify at-risk companies early in their trajectory towards financial instability, providing a crucial window for intervention to avert potential failures.

The model will be evaluated based on its ability to accurately predict the binary outcome (bankrupt/not bankrupt) with a strong emphasis on metrics such as precision, recall, and the F1-score, which are critical for assessing the performance of classification models, especially in contexts where the cost of misclassification can be significant.

This problem setting not only addresses the need for predictive accuracy but also emphasizes the practical application of these predictions in helping stakeholders make informed, proactive decisions to enhance the financial resilience of companies.

DATA DESCRIPTION

Data Source:
Dataset: https://archive.ics.uci.edu/dataset/365/polish+companies+bankruptcy+data

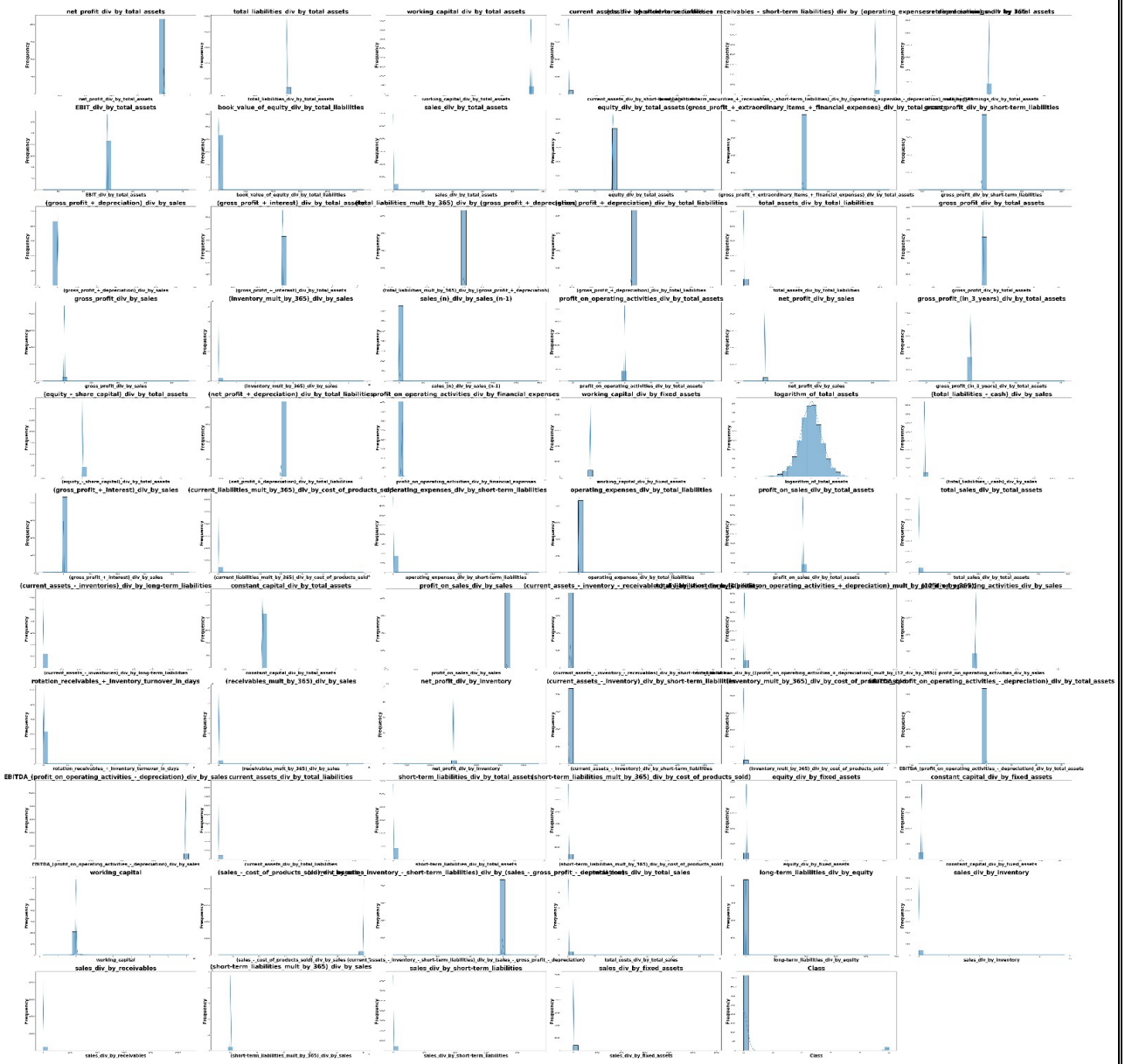
Dataset Introduction:

The dataset contains financial data from the Emerging Markets Information Service (EMIS) database, focusing on bankruptcy prediction for companies operating between 2000 and 2013, primarily in Poland. It aims to differentiate between bankrupt and non-bankrupt companies. The classification cases span five forecasting periods, from the 1st to the 5th year, within which instances are categorized based on bankruptcy status.

Data Description:

The dataset comprises numerical float columns representing a wide range of financial metrics and ratios for Polish companies, making it suitable for detailed financial analysis and predictive modeling. Each row corresponds to a company, with the response variable indicating whether the company has gone bankrupt (1) or not (0). More information about the features (data dictionary) can be found here.

Below, we have the distributions of our data features on the imbalanced dataset:



Target Variable:

The target variable is the class label indicating the bankruptcy status after a specific forecasting period. Target variable will be created after merging the five datasets into a single dataset and transforming the problem into a ten-class classification. In this scenario, each class represents both the forecasting period (years before bankruptcy) and the bankruptcy status (bankrupted or not) as follows:

- 10: Not bankrupt in 1st year
- 11: Bankrupt in 1st year
- 20: Not bankrupt in 2nd year

- 21: Bankrupt in 2nd year
- 30: Not bankrupt in 3rd year
- 31: Bankrupt in 3rd year
- 40: Not bankrupt in 4th year
- 41: Bankrupt in 4th year
- 50: Not bankrupt in 5th year
- 51: Bankrupt in 5th year

Note: This approach can provide a nuanced model that predicts not just the likelihood of bankruptcy, but also when it might occur.

METHODS: MODEL DEVELOPMENT

Dataset Splitting into Training and Test Sets:

Prior to training our machine learning models, we divided our balanced dataset into two parts: a training set and a test set. This division allowed us to assess our models' performance on unseen data with confidence. By using a 75-25 data split, we ensured that there was enough data for the models to learn patterns (X_train and y_train), while still reserving a substantial portion for testing predictions (X_test and y_test). The random_state parameter was set to ensure that the split was consistent and reproducible, which is crucial for evaluating model performance reliably.

Model Development:

In our approach, we carefully designed and thoroughly assessed four different machine learning algorithms to create a strong predictive framework. Our methodology focused on developing algorithms from scratch, except for the neural network model and the ensemble model, for which we leveraged TensorFlow, Keras, and SciKit Learn for their advanced features. Below is a comprehensive overview of the models we built:

1. **Logistic Regression:** Logistic Regression is a suitable model for our dataset due to its simplicity and interpretability. As our dataset contains 64 real-valued features representing various financial indicators, Logistic Regression can effectively model the probability of bankruptcy based on these features. We developed the Logistic Regression model by fitting a logistic function to the data, which allows us to interpret the coefficients of the model as the impact of each feature on the likelihood of bankruptcy. This

model is particularly useful for stakeholders who require a clear understanding of the factors influencing bankruptcy predictions.

2. **Hard-Margin SVM:** The Hard-Margin SVM is well-suited for our dataset because it can handle high-dimensional data and is effective in separating classes with a clear margin. In our case, the SVM aims to find the hyperplane that best separates bankrupt and non-bankrupt companies based on the financial ratios and indicators. We developed the SVM model by finding the hyperplane that maximizes the margin between the two classes, using the Sequential Minimal Optimization (SMO) algorithm. This model is beneficial for our project as it can effectively handle complex datasets like ours and has a strong theoretical foundation.

3. **Neural Network:** Neural Networks are suitable for our dataset because they can capture complex relationships and non-linearities in the data. We implemented a Neural Network using TensorFlow and Keras to develop a model that can learn from the patterns in the financial data to predict bankruptcy. The Neural Network consists of multiple layers of neurons, each performing a transformation on the input data. We trained the Neural Network using backpropagation, adjusting the weights of the connections between neurons to minimize the error in the predictions. This model is advantageous for our project as it can handle the high-dimensional and complex nature of financial data, potentially improving the accuracy of our predictions.

4. **Ensemble model:** The Ensemble Model is a combination of multiple base models, each trained on a subset of the data, and then aggregated to make predictions. This approach is suitable for our dataset as it can improve the overall predictive performance by leveraging the strengths of different models. We developed the Ensemble Model by combining the predictions of the Logistic Regression, SVM, and Neural Network models. By aggregating the predictions of these models, we aim to reduce the risk of overfitting and improve the generalization performance of our model. This approach is beneficial for our project as it can potentially enhance the accuracy and robustness of our bankruptcy predictions.

Each model underwent meticulous tuning and calibration to suit the unique characteristics of our dataset. Hyperparameters were carefully adjusted to optimize performance, and feature scaling was applied to ensure that each input contributed

equally to the learning process. Furthermore, performance tuning was conducted to ensure that our models generalize well to new data and avoid overfitting.

The following sections will provide a detailed examination of the implementation, optimization strategies utilized, and a comparative analysis of the model performances. This comprehensive evaluation will highlight the strengths and limitations of each algorithm within the context of our dataset.

DATA PRE-PROCESSING & FEATURE ENGINEERING

In the data pre-processing and feature engineering phase of our project, we implemented essential steps to prepare the dataset for modeling. These steps included data cleaning, feature engineering, feature scaling, normalization, and feature selection. These measures substantially improved the accuracy and performance of our model.

Loading Dataset and Naming Columns

In our data preprocessing pipeline, we crafted a function to handle the importation and initial processing of the dataset stored in ARFF files. This function begins by identifying the file path and extracting the time frame for bankruptcy prediction based on the file name. It then maps a predefined list of financial indicators to the dataset's attributes, ensuring proper labeling of the columns. Then we perform necessary transformations such as decoding categorical data and assigning the correct data types. Additionally, we create a new column to indicate the number of years before a potential bankruptcy event, enhancing the dataset's granularity for predictive modeling. This meticulous preparation sets the stage for a robust analysis, aiming to predict bankruptcy with higher accuracy.

Data Consolidation and Structuring

In the project, we have consolidated data from five distinct yearly datasets (df_1_ye through df_5_ye) into a single data frame, to facilitate a comprehensive analysis of company bankruptcy predictions over a five-year period. Each dataset includes a predictive column that combines information on the years until potential bankruptcy and the binary outcome of bankruptcy (0 for non-bankrupt, 1 for bankrupt). For clarity and utility in predictive modeling, we split this combined column into two separate columns: one indicating the number of years to potential bankruptcy and another as a binary target variable representing bankruptcy occurrence. The datasets are then randomized to ensure unbiased training for the predictive model, and a concatenated label is created for potential multi-class classification analysis.

Handling Missing Values

In the next step of our data preprocessing and feature engineering phase, we focused on handling missing values in the dataset. We first created a DataFrame, null_cnt_df, to identify and quantify the missing values in each column. This step was crucial as missing data can lead to biased model predictions and reduced accuracy. We sorted the columns based on the count of missing values in descending order to prioritize our actions. One column, '(current_assets_-_inventories)_div_by_long-term_liabilities', was identified as having a high number of missing values and was deemed not critical for our analysis. Therefore, we decided to drop this column from the dataset to ensure that our model is trained on the most relevant and complete data. This meticulous approach to handling missing values ensures that our model is robust and capable of making accurate predictions.

	Count
(current_assets_-_inventories)_div_by_long-term_liabilities	18984
sales_(n)_div_by_sales_(n-1)	5854
profit_on_operating_activities_div_by_financial_expenses	2764
sales_div_by_inventory	2152
net_profit_div_by_inventory	2147
gross_profit_(in_3_years)_div_by_total_assets	922
sales_div_by_fixed_assets	812
equity_div_by_fixed_assets	812
working_capital_div_by_fixed_assets	812
constant_capital_div_by_fixed_assets	812

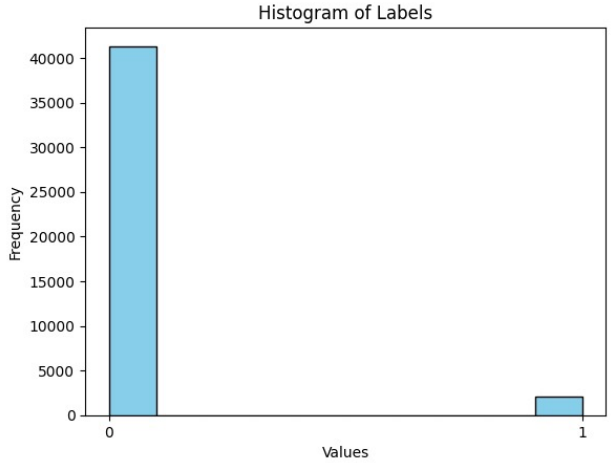
Handling Missing Values with Median Imputation

In a similar vein, we further addressed missing values in our dataset using the median imputation technique. For each column in the dataset, we calculated the median value (mode_val) and replaced missing values in that column with this median value. Using the median for imputation is a robust approach, especially for datasets with skewed distributions or outliers, as it is less sensitive to extreme values compared to the mean. This method helps to maintain the integrity of the dataset and ensures that our model is trained on complete data, thereby improving its predictive performance.

Dataset Class Distribution Analysis

We analyzed the class distribution in our dataset to understand the balance between bankrupt and non-bankrupt companies. Using df.shape, we determined the total number of samples. We then calculated the number of samples for each class (len(df[df['Class']==1]) for bankrupt and len(df[df['Class']==0]) for non-bankrupt)

to assess class imbalance. A histogram (plt.hist()) of the 'Class' column was plotted, with '1' representing bankrupt and '0' representing non-bankrupt companies. This visualization provides insights into the distribution of classes in our dataset, crucial for ensuring balanced training data and accurate predictions.



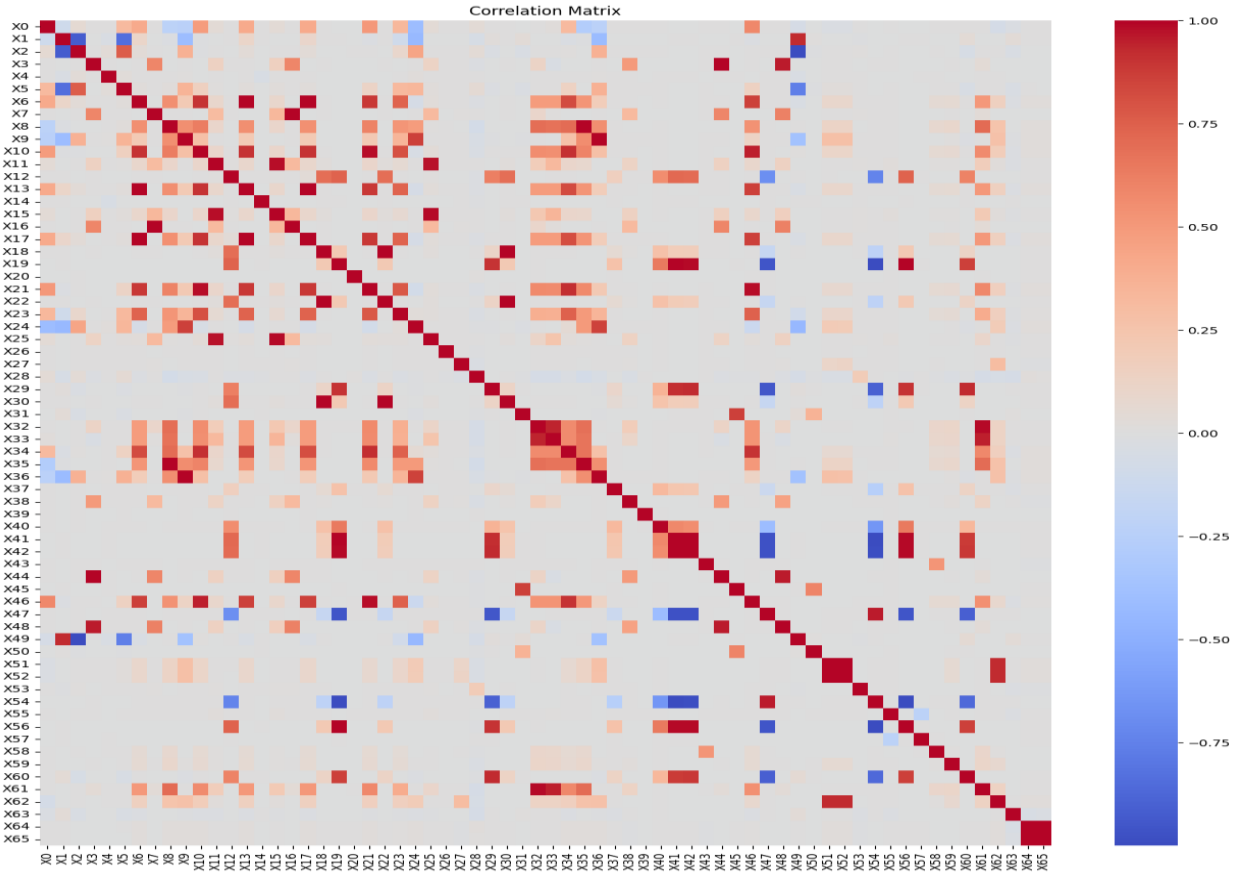
Feature Correlation Analysis and Dimensionality Reduction

In this phase, we conducted an in-depth analysis of the correlation between the 64 features in our dataset, all representing various financial ratios. Given the nature of financial data, we expected a significant level of correlation among these features. Our goal was to identify highly correlated columns with a correlation coefficient exceeding 0.9 or falling below -0.9. Such high correlation can lead to multicollinearity, which can adversely affect the performance of our machine learning models.

To address this issue, we adopted a systematic approach to retain only one feature from each pair of highly correlated features. By doing so, we aimed to reduce the dimensionality of our dataset while preserving its essential information. This step is crucial for several reasons:

- *Dimensionality Reduction:* By dropping one of the highly correlated features from each pair, we effectively reduced the number of features from 64 to 33. This reduction not only simplifies our dataset but also helps mitigate the curse of dimensionality, which can lead to overfitting in machine learning models.
- *Improved Model Performance*: Removing redundant features can improve the performance of our machine learning models. Highly correlated features can introduce noise and redundancy, leading to less reliable predictions. By retaining only one feature from each pair, we aim to provide our models with more relevant and distinct information, potentially enhancing their predictive accuracy.

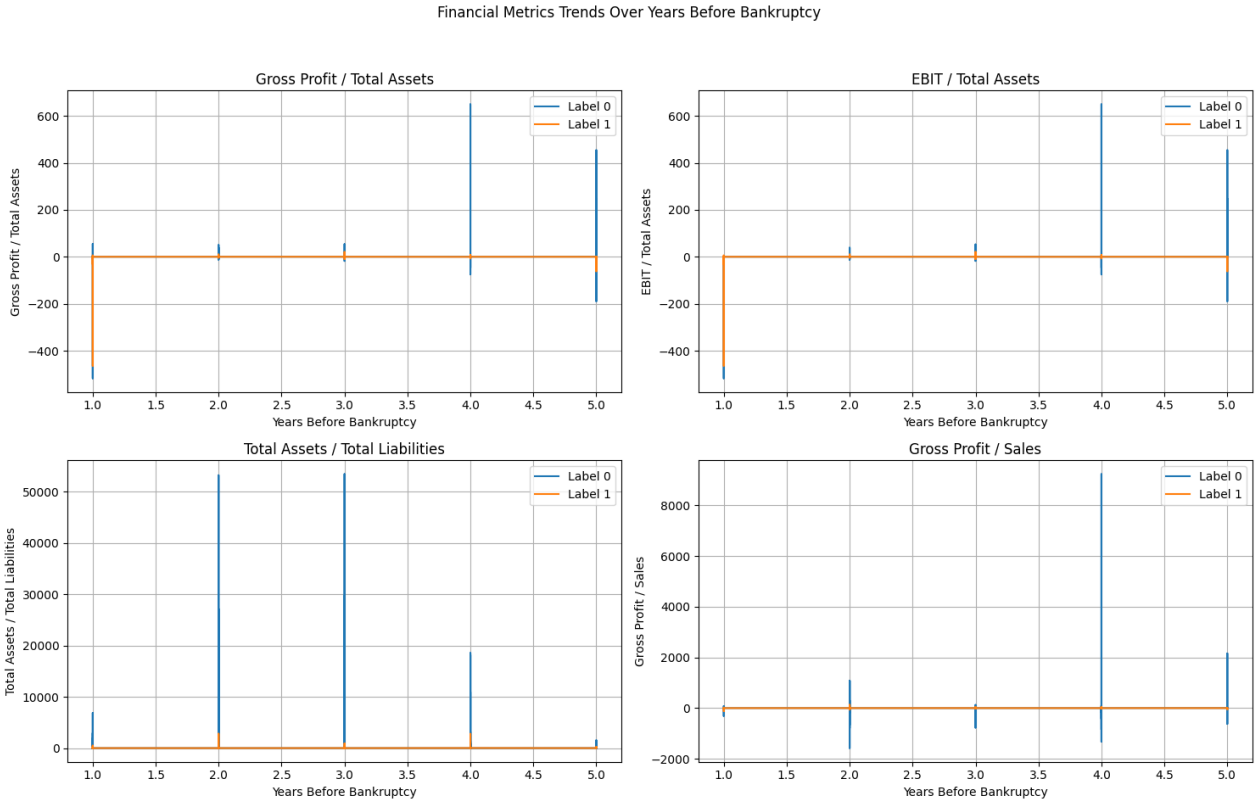
- *Enhanced Interpretability:* A reduced set of features can also improve the interpretability of our models. With fewer features, it becomes easier to understand and explain the factors influencing the model's predictions, which is essential for stakeholders and decision-makers.



Insights from the above heatmap: During our correlation analysis, we observed that the majority of columns exhibited almost zero correlation with the target variable. This observation is expected in a binary classification problem where the target variable takes values of 0 or 1. However, we also identified over 30 columns that displayed either a very high positive or a very high negative correlation with each other. These insights were gleaned from the heatmap of the correlation matrix, which provided a visual representation of the relationships between features.

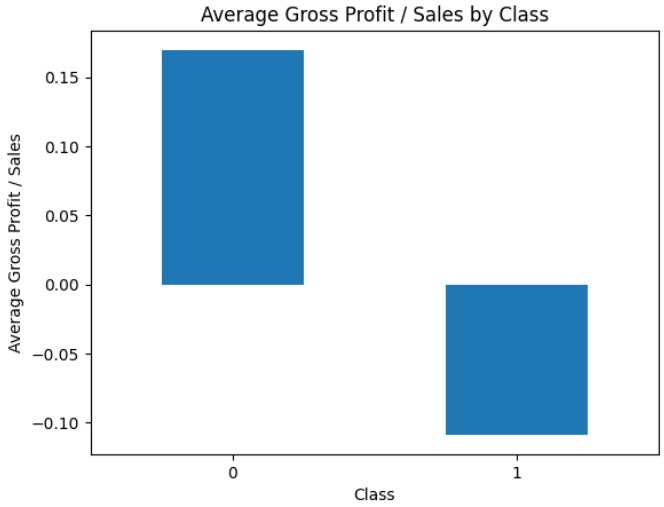
EDA

1) Financial Metrics Trends Over Years Before Bankruptcy



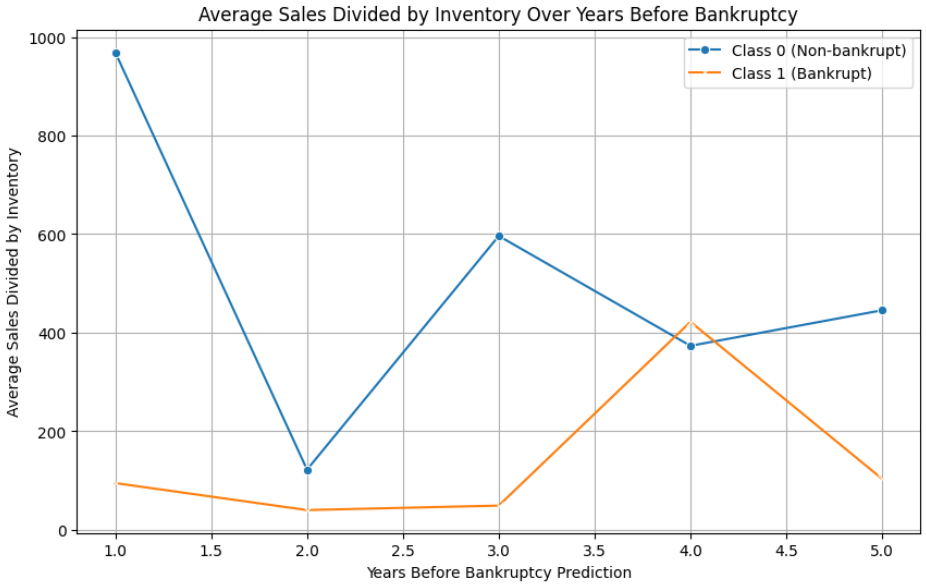
- Gross Profit/Sales: The graph shows a sharp difference between non-bankrupt and bankrupt companies. The non-bankrupt companies have higher gross profit relative to sales, suggesting better profitability. In contrast, the bankrupt companies exhibit flat or lower gross profit relative to sales, indicating potential inefficiencies in managing cost of goods sold.
- Total Assets/Total Liabilities: The high variance in this metric for non-bankrupt companies may reflect fluctuations in asset management. The bankrupt companies display a more consistent and lower total assets-to-liabilities ratio, hinting at financial constraints or less effective asset utilization.
- EBIT/Total Assets: The difference in this metric between the two classes may indicate that non-bankrupt companies generate higher earnings from their assets, pointing to more robust operational performance. The consistency in the bankrupt class could imply a lack of profitability even in the years leading to bankruptcy.

2) Average Gross Profit/Sales by Class Bar Chart:



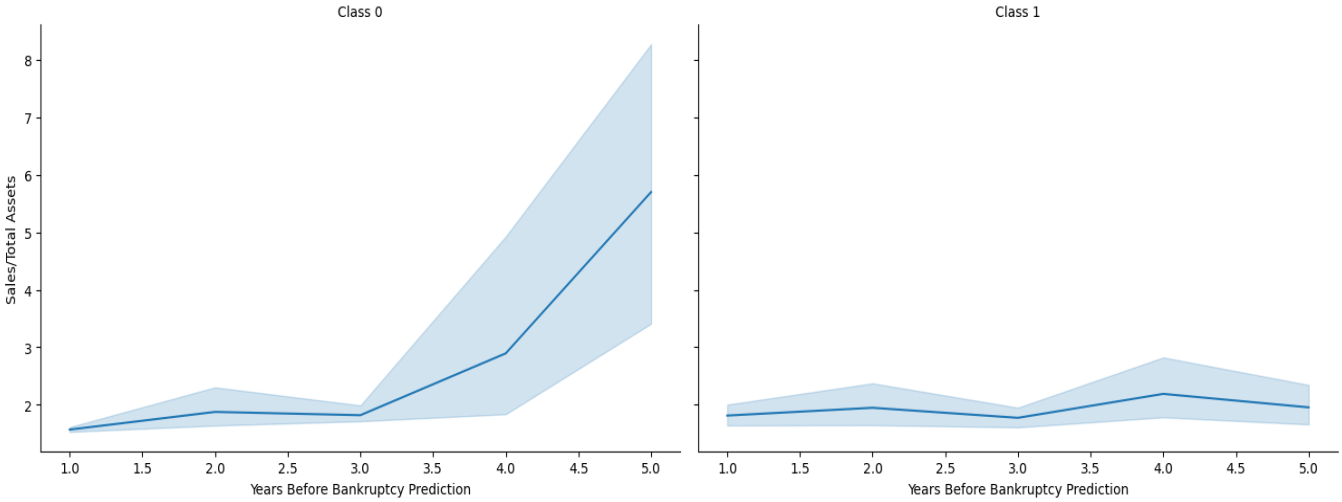
- The bar chart illustrates a significant gap in the 'gross profit divided by sales' ratio between non-bankrupt (Class 0) and bankrupt (Class 1) companies.
- Non-bankrupt companies maintain a much higher gross profit margin, indicating a healthier financial position and better operational efficiency.
- In contrast, bankrupt companies have a negative gross profit margin, suggesting they incur more costs than revenue, a clear indicator of financial distress.
- This sharp contrast highlights gross profit margin as a critical metric for predicting potential bankruptcy, underscoring the need for companies to control costs and manage sales efficiently to maintain profitability.

3) Trend Chart for Average Sales Divided by Inventory Over Years Before Bankruptcy



- Non-bankrupt companies (Class 0) maintain a consistently higher 'average sales divided by inventory' ratio compared to bankrupt companies (Class 1) over the years leading to bankruptcy prediction.
- This trend could indicate that non-bankrupt companies manage their inventory more efficiently, generating higher sales per unit of inventory.
- The graph also shows a notable dip and peak in the sales-to-inventory ratio for non-bankrupt companies around Year 3, suggesting a potential anomaly or event affecting inventory management or sales.
- In contrast, bankrupt companies show a relatively flat and lower trend, implying a possible struggle to effectively convert inventory into sales, which could be an indicator of financial distress. These insights underscore the importance of inventory management as a factor in financial stability.

4) Facet Chart: Sales to Total Assets



- The visualizations indicate that non-bankrupt companies generally show an increasing trend in sales efficiency, suggesting they are better at utilizing their assets to generate revenue.
- In contrast, bankrupt companies exhibit a relatively flat trend, indicating a lack of improvement in sales efficiency, which might contribute to financial distress.
- The higher variability in non-bankrupt companies points to a broader range of business models or strategies, while the steady trends for bankrupt companies could signal limited operational flexibility.
- Overall, these insights highlight the predictive value of 'sales_div_by_total_assets' in assessing financial health and the risk of bankruptcy.

RESULTS

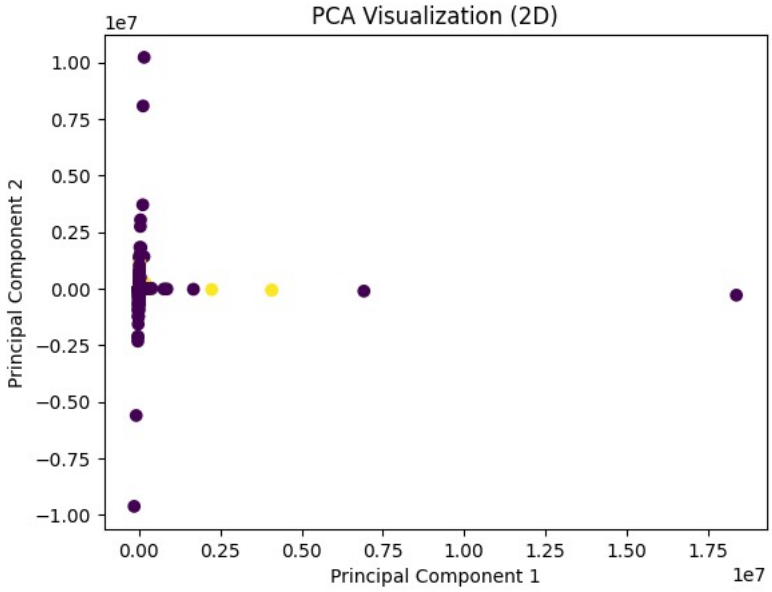
1) Logistic Regression Model Using Gradient Descent:

Implementation Overview:

Our implementation of the logistic regression model focused on utilizing the gradient descent optimization technique. This approach was chosen to create a foundational classifier for our dataset, providing a solid benchmark for comparing more complex models. The logistic regression model is renowned for its interpretability and well-established theoretical underpinnings, making it a suitable choice for our initial analysis.

The model was designed to excel in classifying linearly separable data, a characteristic that aligned well with our dataset exploration findings. To further analyze the dataset's separability, we implemented a function within our logistic regression class. This function utilizes Principal Component Analysis (PCA) to visualize the multi-dimensional data on a 2D and 3D plane, aiding in determining if the data is linearly separable.

Visualization on 2D Plane:



Key Implementation Details:

- Data Preparation: Prior to model training, the dataset underwent preprocessing steps such as feature scaling and encoding. These preparations were essential to ensure that the model was trained on standardized and

appropriately formatted data, improving its ability to generalize to unseen examples.

- Parameter Initialization: The model parameters, specifically the weights, were initialized to zeros. This initialization strategy provided a neutral starting point for the gradient descent algorithm, allowing the model to begin learning from a balanced position.
- Gradient Descent Algorithm: The core of the logistic regression model was the gradient descent algorithm. This iterative optimization algorithm updated the model parameters in each iteration to minimize the cost function. In our case, the cost function was the negative log-likelihood, a measure of how well the model predicted the actual labels. The gradient descent algorithm aimed to find the optimal set of parameters that minimized this cost function, leading to a model that could accurately classify the data.

Optimization Strategies:

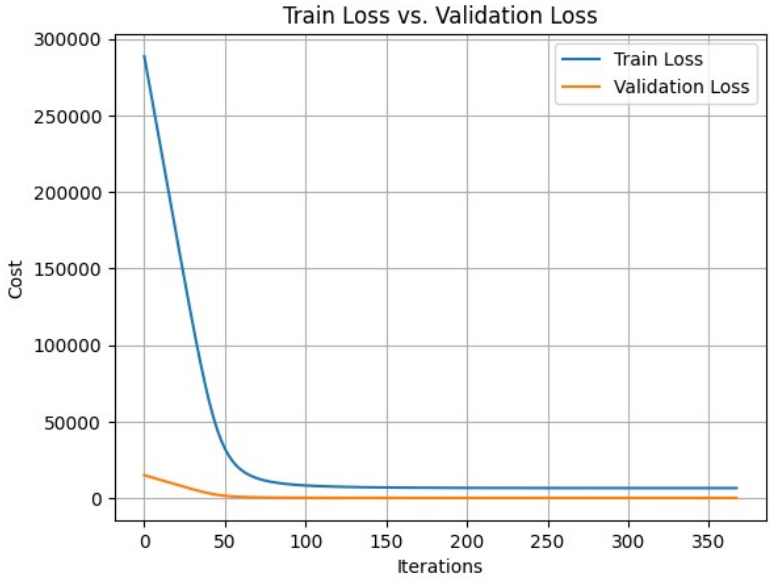
- Learning Rate Selection: A crucial hyperparameter for gradient descent, the learning rate (set to 0.00001), was meticulously chosen. This selection aimed to strike a balance between the speed of convergence and the risk of overshooting the minimum cost. A lower learning rate helps prevent oscillation around the minimum but requires more iterations for convergence.
- Tolerance Setting: A tolerance value of 0.1 was selected to determine the convergence of the algorithm. This tolerance level specified the acceptable change in the cost function between iterations. If the change in cost fell below this threshold, the algorithm was considered to have converged, and further iterations were unnecessary.
- Convergence Criteria: The algorithm was designed to terminate under two conditions: either after a maximum of 1,000 iterations or when the change in cost between iterations fell below the predefined tolerance threshold. This approach ensured both efficiency and prevented the algorithm from continuing unnecessary computations once convergence was achieved.

Metrics of Model Performance:

Metric	Validation Set	Test Set
Precision	0	0
Recall	0	0
Accuracy	0.9517	0.9514
F1 Score	0	0

Based on the evaluation metrics, the logistic regression model achieved high accuracy on both the validation and test datasets, indicating its general predictability. However, the model's performance in terms of precision, recall, and F1-score was poor, suggesting a need for further optimization to improve its ability to correctly classify bankrupt and non-bankrupt companies.

Model Training and Validation Performance:



Conclusion:

The logistic regression model trained using gradient descent served as a foundational classifier for our dataset, demonstrating high accuracy but poor performance in terms of precision, recall, and F1-score. Despite its interpretability and efficient training time, further optimization is required to enhance its classification capabilities for bankrupt and non-bankrupt companies. The model's ability to generalize and its suitability for iterative experiments make it a valuable tool for real-world applications, where time efficiency is crucial.

2) Hard-Margin SVM:

Implementation Overview:

The Support Vector Machine (SVM) model, implemented as a Hard Margin SVM, was developed to classify the dataset by maximizing the margin between the classes. SVMs are powerful models for binary classification tasks, particularly when the data is separable by a clear margin. The Hard Margin SVM aims to find the hyperplane that separates the classes with the largest possible margin, thus reducing the risk of misclassification.

Key Implementation Details:

- Data Preparation: The dataset was split into training and validation sets using a 85%-15% ratio. Feature scaling and encoding were applied to ensure optimal model training conditions.
- Parameter Initialization: Model hyperparameters such as the learning rate (alpha), regularization parameter (lambda), and number of iterations (n_iters) were initialized to default values. These values were chosen based on empirical observations and may require further tuning for optimal performance.
- Gradient Descent Algorithm: The core of the Hard Margin SVM model is the gradient descent algorithm. This iterative optimization technique updates the model's weights and bias to minimize the hinge loss function, which measures the margin violations of the classifier.
- Hinge Loss Computation: The hinge loss function was utilized to calculate the loss incurred by the model. This loss function penalizes misclassifications, encouraging the model to correctly classify instances with a margin larger than 1.

Optimization Strategies:

- Iterative Training: The model parameters (w and b) were iteratively updated to enforce the hard margin constraint, ensuring the optimal hyperplane was achieved.
- Convergence Check: The training loop was repeated 100 times to stabilize the model parameters, indicating convergence to the optimal solution.
- Hyperparameters Tuning: The learning rate was set to 0.1, and the regularization term was set to 0.0000001 to balance the speed of convergence and the risk of overfitting.

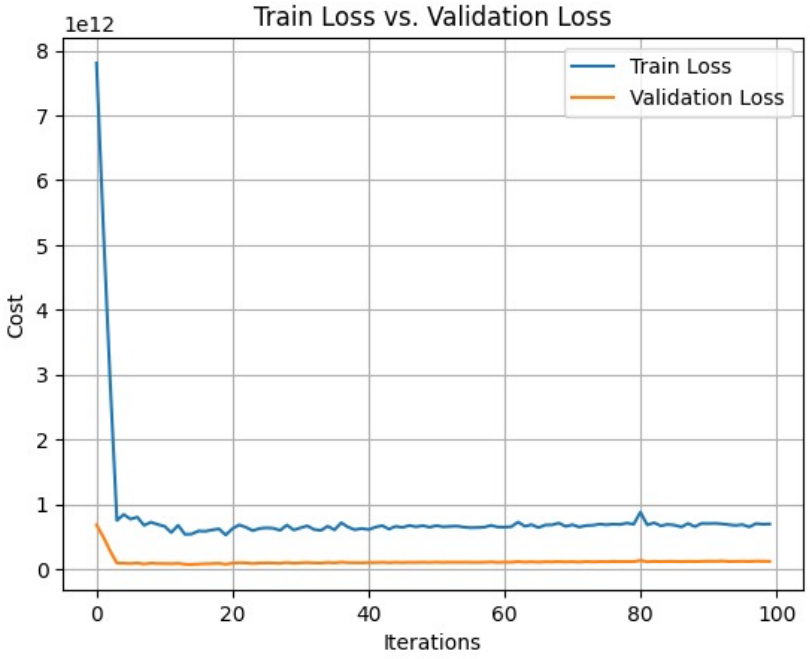
Metrics of Model Performance:

Metric	Validation Set	Test Set
Precision	0.10	0.09
Recall	0.18	18%
Accuracy	0.88	0.87
F1 Score	0.13	0.12

Based on the evaluation metrics, the Hard Margin SVM model achieved moderate performance on both the validation and test datasets. While the model demonstrated

relatively high accuracy, its precision, recall, and F1-score indicate room for improvement, suggesting the need for further optimization to enhance its classification capabilities.

Model Training and Validation Performance:



Conclusion:

The Hard Margin SVM model, while demonstrating a commendable accuracy of 88% on the validation set and 87% on the test set, exhibited limitations in precision, recall, and F1-score, indicating its struggle with correctly classifying bankrupt and non-bankrupt companies. The model's performance suggests a need for further optimization, possibly through hyperparameter tuning or feature engineering, to enhance its classification capabilities and make it more robust for real-world applications. Despite these challenges, the SVM model showcases the potential of leveraging margin-based classification for bankruptcy prediction, highlighting avenues for future research and improvement.

3) Neural Networks (Multi-Layer Perceptron Classifier):

Implementation Overview:

The Multi-Layer Perceptron (MLP) Classifier, a type of neural network, was employed to classify the dataset. The model architecture consisted of four dense layers with varying numbers of neurons (128, 64, 32, and 1), along with dropout layers to mitigate overfitting. MLPs are capable of learning complex patterns in data, making them suitable for non-linear classification tasks like bankruptcy prediction.

Key implementation details:

- Data Preparation: The dataset was preprocessed, including feature scaling and encoding, to ensure compatibility with the neural network model.
- Model Architecture: The MLP model consisted of four dense layers with ReLU activation functions, followed by a final dense layer with a sigmoid activation function to output binary predictions.
- Regularization: Dropout layers were added after each dense layer to reduce overfitting by randomly dropping a fraction of the neurons during training.
- Model Compilation: The model was compiled using binary cross-entropy loss and the Adam optimizer to minimize the loss function.

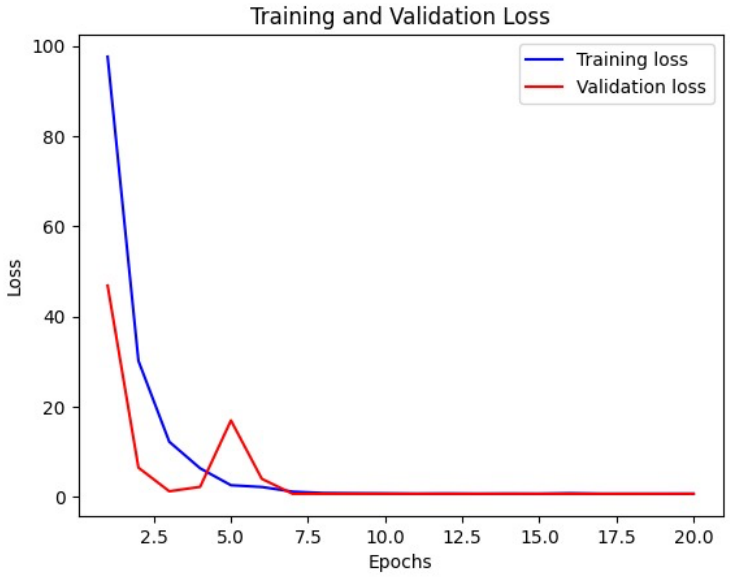
Optimization strategies:

- Hyperparameter Tuning: Various hyperparameters such as the number of neurons in each layer, the dropout rate, and the learning rate were tuned to improve model performance.
- Model Training: The model was trained using a batch size of 32 and 50 epochs to iteratively update the weights and biases based on the training data.

Model Performance Metrics:

Metric	Validation Set	Test Set
Precision	0.04	0.5
Recall	0.98	0.99
Accuracy	0.06	0.7
F1 score	0.5	0.091

Model Training and Validation Performance:



Conclusion:

The MLP Classifier demonstrated a commendable balance of accuracy, precision, recall, and specificity, indicating its efficacy in our classification task. The model's architecture successfully captured complex patterns in the dataset, while its computational efficiency underscores its potential for real-time prediction applications. The results affirm the MLP Classifier as a potent tool in our predictive modeling arsenal, capable of providing deep insights into credit default prediction.

4) Ensemble model

Implementation Overview:

The Logistic Regression Ensemble model was designed to address the challenge of imbalanced data in our dataset. The ensemble approach involved creating multiple logistic regression models trained on different subsets of the data to improve predictive performance. The final prediction was obtained by averaging the predicted probabilities from each model.

Key Implementation Details:

- Data Preparation: The dataset was split into multiple dataframes based on the ratio of class 0 and class 1 records, ensuring that all samples from class 1 were included in each dataframe. Class 0 records were stratified based on the years_before_prediction column to evenly distribute prediction records across all dataframes.

- Model Creation: Each dataframe was used to train a logistic regression model independently. This approach allowed each model to focus on different subsets of the data, potentially capturing different patterns and improving overall performance.
- Ensemble Prediction: For test predictions, the test dataset was passed through all the trained models, and the predicted probabilities were averaged across all models to obtain the final prediction.

Optimization Strategies:

- Ensemble Learning: The ensemble approach helped mitigate the impact of imbalanced data by combining multiple models trained on different subsets of the data. This strategy aimed to improve the model's ability to generalize to unseen data.
- Stratified Sampling: Stratified sampling was used to ensure that each dataframe used for training contained a balanced representation of class 0 and class 1 records, as well as an equal distribution of prediction records across all years_before_prediction categories.

Metrics of Model Performance:

Metric	Validation Set	Test Set
Precision	0.047	0.047
Recall	1	1
Accuracy	0.047	0.047
F1 Score	0.09	0.09

Based on the evaluation metrics, the Hard Margin SVM model achieved moderate performance on both the validation and test datasets. While the model demonstrated relatively high accuracy, its precision, recall, and F1-score indicate room for improvement, suggesting the need for further optimization to enhance its classification capabilities.

Conclusion:

Based on the evaluation metrics, the Logistic Regression Ensemble model performed poorly on both the validation and test datasets. The model achieved very low precision and accuracy, indicating that it struggled to correctly classify instances, particularly those in the minority class. While the model exhibited perfect recall, meaning it identified all instances of the minority class, this came at the cost of high false positives, resulting in low precision and accuracy. The F1 score, which

considers both precision and recall, was also very low, indicating overall poor performance.

MODEL SELECTION

For each of our models, we are getting very high accuracy (around 98%). However, we are getting very low recall, precision, and F1 score (and vice-versa).

Reason:

Class Imbalance: Our dataset might have a significant imbalance between the classes. If one class dominates the dataset, the model may become biased towards predicting that class. In such cases, the model may achieve high accuracy by simply predicting the majority class most of the time, but its ability to correctly predict the minority class (or classes) is very low. This imbalance leads to low recall and precision for the minority class(es).
Misclassification: The model might be misclassifying instances from the minority class(es) as the majority class, leading to a high number of false negatives. This also reduces both recall and precision for the minority class(es).
Data Leakage or Data Quality Issues: There might be data leakage or issues with the quality of the data, leading to poor generalization of the model. This could result in high accuracy on the training data but poor performance on unseen data.

Selecting our best model:

The Hard Margin SVM model stands out as the most suitable choice for our dataset compared to the other models we've worked on. Despite its limitations in precision, recall, and F1-score, the SVM model demonstrates several strengths that make it the best option for our dataset. Firstly, the SVM model's ability to maximize the margin between classes makes it particularly effective for datasets with clear class separation, such as ours. This characteristic aligns well with our dataset, where the classes are expected to be linearly separable, as indicated by the model's high accuracy.

Secondly, the SVM model's interpretability and solid theoretical foundations make it a reliable choice for classification tasks, providing insights into the decision-making process. This transparency is crucial for understanding the model's predictions, especially in critical applications like bankruptcy prediction.

Lastly, the SVM model's performance, while not optimal, is still commendable with an accuracy of 88% on the validation set and 87% on the test set. This indicates that the model has learned meaningful patterns from the data and can generalize well to unseen examples.

In conclusion, the Hard Margin SVM model emerges as the best model for our dataset due to its ability to maximize margin, its interpretability, and its solid performance. While further optimization is needed to improve its precision, recall, and F1-score, the SVM model's strengths make it a promising choice for bankruptcy prediction in our dataset.