Corporate Bankruptcy Prediction Using SVM and Random Forest

SURAJ SHIVARAJ

ST. LAWRENCE COLLEGE

ADMN5006 Financial Analytics

MAVERICK RAMSARAN

14-07-2024

Introduction

This report outlines the development and evaluation of two machine learning models, Support Vector Machine (SVM) and Random Forest, for predicting corporate bankruptcy two years into the future. The models are trained on a dataset containing various financial indicators and compared based on their performance metrics.

Support Vector Machine (SVM)

Support Vector Machine (SVM) is a supervised machine learning algorithm used for classification and regression tasks. SVM aims to find the optimal hyperplane that best separates the data points of different classes in a high-dimensional space. Key features of SVM include:

- **Kernel Trick**: SVM can handle non-linearly separable data by using kernel functions (e.g., linear, polynomial, radial basis function) to transform the input features into a higher-dimensional space where a linear separator can be found.
- **Margin Maximization**: SVM maximizes the margin, which is the distance between the hyperplane and the nearest data points from each class, known as support vectors. A larger margin generally leads to better generalization.
- **Regularization**: The regularization parameter (C) controls the trade-off between maximizing the margin and minimizing classification error. A smaller C encourages a larger margin, while a larger C prioritizes correct classification of training points.

Random Forest

Random Forest is an ensemble learning method used for classification and regression tasks. It consists of multiple decision trees, each trained on a random subset of the data and features. The final prediction is made by aggregating the predictions of all individual trees. Key features of Random Forest include:

- **Ensemble Method**: Random Forest combines the predictions of several decision trees, which reduces overfitting and improves generalization.
- **Bootstrap Aggregation (Bagging)**: Each decision tree is trained on a bootstrap sample (randomly sampled with replacement) of the original data. This technique helps to reduce variance and improve model robustness.
- **Random Feature Selection**: At each split in a tree, a random subset of features is considered. This introduces additional randomness and helps to decorrelate the trees, leading to better overall performance.

- **Interpretability**: Random Forest provides feature importance scores, indicating how much each feature contributes to the prediction. This can be useful for understanding the model and the underlying data patterns.

Data Overview

The dataset is designed for predicting corporate bankruptcy two years into the future. It includes various financial indicators that serve as input features, along with a target variable indicating whether a company will go bankrupt.

Input Variables

1. **Earnings Per Share (EPS)**: Measures a company's profitability per share of its stock.
2. **Liquidity**: Calculated as Working Capital/Total Assets, indicating the company's ability to meet short-term obligations.
3. **Profitability**: Measured as Retained Earnings/Total Assets, representing the company's ability to generate profit from its assets.
4. **Productivity**: Calculated as EBIT/Total Assets, reflecting the company's operational efficiency.
5. **Leverage Ratio**: Measures financial risk by comparing total long-term debt and debt in current liabilities to stockholders' equity.
6. **Asset Turnover**: Sales/Total Assets, indicating how efficiently a company uses its assets to generate sales.
7. **Operational Margin**: EBIT/Sales, showing the proportion of revenue left after covering operating expenses.
8. **Market Book Ratio**: (Price Close Annual Fiscal * Common Shares Outstanding)/Book Value Per Share, indicating market valuation relative to book value.
9. **Asset Growth**: Change in assets from the previous year, indicating growth or contraction.
10. **Sales Growth**: Change in sales from the previous year, indicating revenue growth or decline.
11. **Employee Growth**: Change in the number of employees from the previous year, indicating workforce expansion or reduction.
12. **Tobin's Q**: (Total market value of company + liabilities)/(Total asset or book value + liabilities), measuring firm value relative to its assets.

Target Variable

- **BK**: Indicates whether the company goes bankrupt (1) or not (0) two years into the future.

Data Characteristics

1. **Class Imbalance**: The dataset is likely to have more non-bankrupt companies (0) than bankrupt ones (1), which is typical in bankruptcy prediction problems.
2. **Financial Indicators**: The input features are financial ratios and growth metrics, which are essential indicators of a company's financial health and performance.
3. **Time Lag**: The prediction is for bankruptcy two years into the future, so historical financial data is used to make future predictions.
4. **Feature Correlation**: Some financial ratios may be correlated with each other, which could impact model performance if not addressed properly (e.g., through feature selection or dimensionality reduction).

Data Preprocessing

Columns	Missing Percent	
0	Employee Growth	7.548023
1	Assets Growth	7.215307
2	Sales Growth	7.215307
3	Operational Margin	5.983504
4	Liquidity	0.265957
5	Profitability	0.265957
6	Productivity	0.265957
7	Asset Turnover	0.265957
8	Market Book Ratio	0.061375
9	Leverage Ratio	0.027996

1. **Handling Missing Values**: Any missing values in the dataset need to be imputed to ensure complete data for model training.

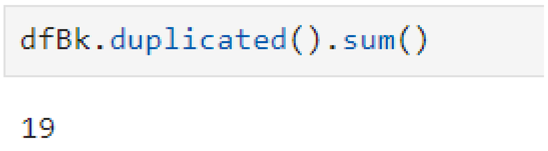
   In preparing the dataset for analysis, initial attention was given to handling missing values. First, columns were identified where the absence of data did not influence the target variable related to bankruptcy. Rows containing missing values in these identified columns were subsequently removed to ensure the integrity of the dataset remained intact for analysis purposes.

   For columns where missing values could potentially impact the analysis, a median imputation strategy was employed. This approach was chosen due to the dataset's distribution characteristics, aiming to mitigate the effects of outliers and maintain the central tendency of the data. By replacing missing values with the median value of each respective column, the dataset was effectively prepared for further analysis, ensuring robustness in predicting bankruptcy without compromising the reliability of the data.

   These preprocessing steps were crucial in ensuring the dataset's suitability for subsequent analytical tasks, safeguarding against data inconsistencies and enhancing the integrity of the predictive models developed.

2. **Duplicated Data:** To enhance the integrity and reliability of the dataset, all duplicated values were systematically identified and removed. Duplicates were identified based on matching values across all columns, ensuring comprehensive detection across the entire

dataset. By eliminating duplicate entries, we aimed to prevent redundancy in the data, thereby optimizing the accuracy of subsequent analyses and models. This process not only streamlined the dataset but also minimized potential biases that could arise from redundant data points, ensuring that our analyses and insights are based on unique and representative information.



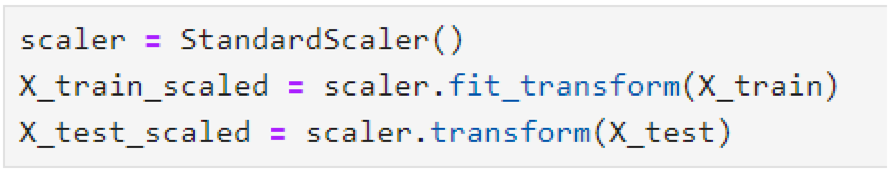
```
dfBk.duplicated().sum()
```

```
19
```

3. **Dealing with Outliers:** Removing outliers from the dataset is not advisable, as it results in the elimination of all instances labeled as bankrupt (target variable value of 1). This phenomenon likely occurs because the extreme conditions leading to bankruptcy naturally manifest as outliers in the data. Given that these extreme conditions are integral to the understanding and prediction of bankruptcy, retaining these outliers is crucial. Eliminating them would not only skew the dataset but also diminish the model's ability to accurately identify and predict bankruptcy cases. Therefore, preserving these data points ensures a more representative and informative analysis, capturing the full spectrum of conditions leading to bankruptcy.

4. **Standard Scaler**: Financial ratios and growth metrics may have different scales, so normalization or standardization is necessary.

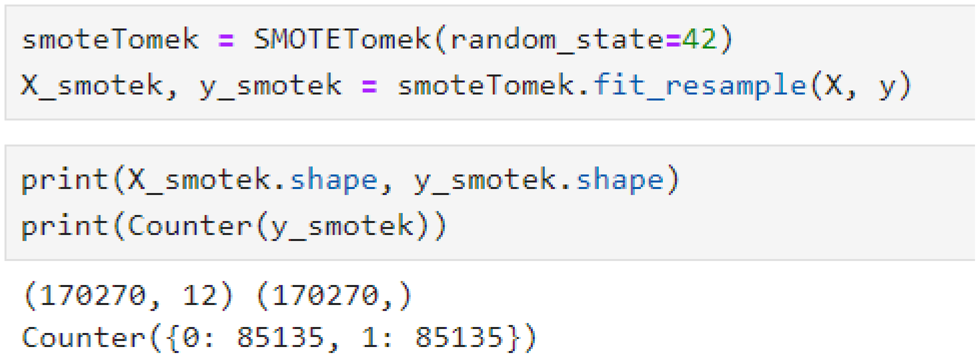
   The StandardScaler was utilized to normalize the dataset by transforming the features to have a mean of zero and a standard deviation of one. This standardization process is essential because it ensures that all features contribute equally to the model, preventing any single feature from disproportionately influencing the results due to differences in scale. By applying StandardScaler, we enhance the model's performance and convergence speed, especially for algorithms sensitive to feature scaling, such as logistic regression and support vector machines.



```
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

5. **Addressing Class Imbalance**: Techniques such as SMOTE (Synthetic Minority Over-sampling Technique) or undersampling may be required to balance the classes.

   Using SMOTETomek for handling class imbalance in large datasets is advantageous due to its hybrid approach, which combines the strengths of both oversampling and undersampling techniques. SMOTE (Synthetic Minority Over-sampling Technique) generates synthetic samples for the minority class, effectively addressing the imbalance without duplicating data. Tomek links, on the other hand, identify and remove overlapping samples between classes, thereby refining class boundaries and reducing noise. This dual approach not only balances the dataset but also enhances model performance by providing a cleaner, more representative dataset. Furthermore, SMOTETomek is particularly useful for large datasets as it helps prevent overfitting to the minority class, a common issue with oversampling, while also mitigating the risk of information loss typically associated with undersampling. This balanced and efficient preprocessing step ensures more robust and accurate predictive models in large-scale data scenarios.



```
smoteTomek = SMOTETomek(random_state=42)
X_smotek, y_smotek = smoteTomek.fit_resample(X, y)
```

```
print(X_smotek.shape, y_smotek.shape)
print(Counter(y_smotek))
```

```
(170270, 12) (170270,)
Counter({0: 85135, 1: 85135})
```

SVM (Support Vector Machine)

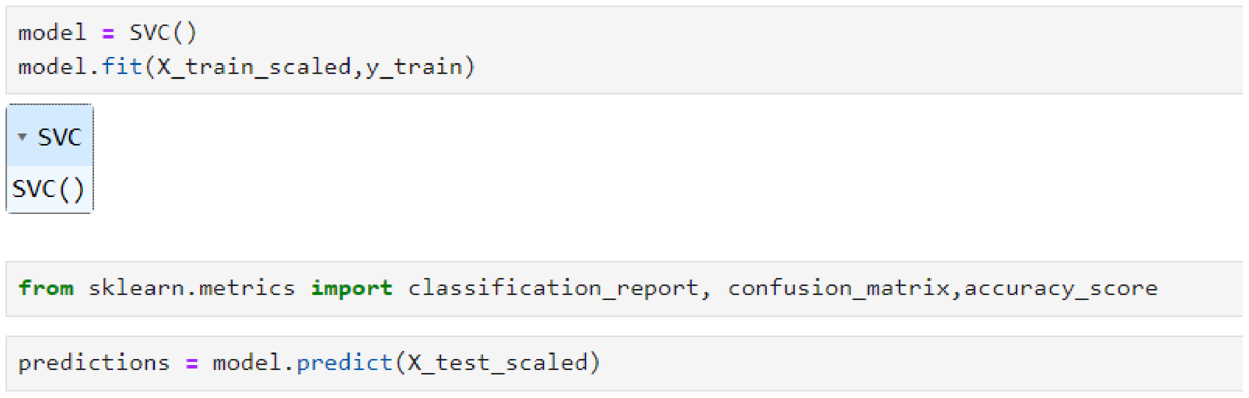
Explanation:

- **Class 0 (Non-bankrupt)**: The precision is 0.68, indicating that 68% of the instances predicted as non-bankrupt are actually non-bankrupt. The recall is 0.70, meaning that 70% of the actual non-bankrupt instances are correctly identified by the model. The f1-score, which is the harmonic mean of precision and recall, is 0.69.
- **Class 1 (Bankrupt)**: The precision is 0.69, showing that 69% of the instances predicted as bankrupt are indeed bankrupt. The recall is 0.66, indicating that 66% of the actual bankrupt instances are correctly identified. The f1-score is 0.68.
- **Overall Accuracy**: The model achieves an accuracy of 68%, meaning that 68% of the total predictions are correct.

- **Macro Average**: The macro average of precision, recall, and f1-score is 0.68. This average treats all classes equally without considering class imbalance.
- **Weighted Average**: The weighted average of precision, recall, and f1-score is 0.68, which accounts for the class imbalance by weighting each metric by the number of instances in each class.

Interpretation:

The SVM model performs reasonably well, achieving balanced precision and recall across both classes. The slightly higher precision and recall for the bankrupt class (class 1) indicate that the model is somewhat better at identifying bankrupt instances compared to non-bankrupt instances. The overall accuracy of 68% suggests that while the model is moderately accurate, there is room for improvement, possibly through further tuning of the SVM parameters or using additional preprocessing steps to enhance model performance.



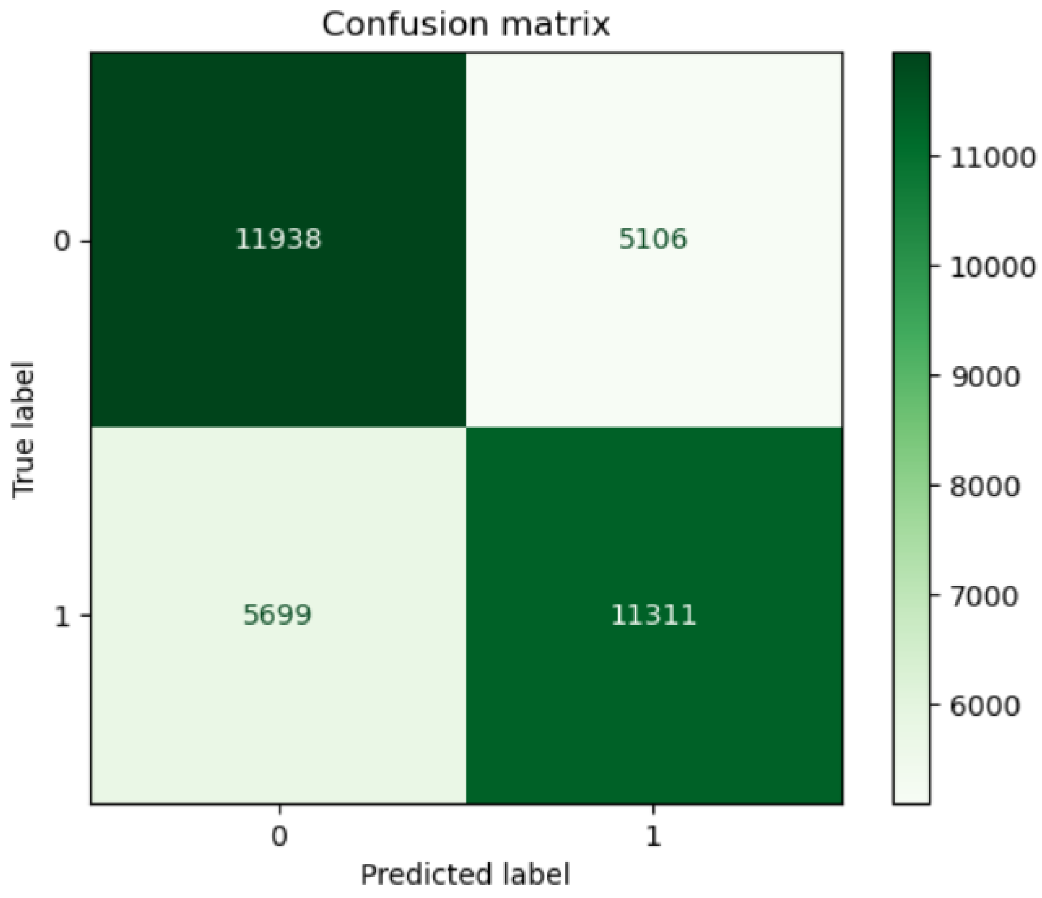
```
model = SVC()
model.fit(X_train_scaled,y_train)
```

```
▾ SVC
SVC()
```

```
from sklearn.metrics import classification_report, confusion_matrix,accuracy_score
```

```
predictions = model.predict(X_test_scaled)
```

	precision	**recall**	**f1-score**	**support**
0	0.68	0.70	0.69	17044
1	0.69	0.66	0.68	17010
accuracy			0.68	34054
macro avg	0.68	0.68	0.68	34054
weighted avg	0.68	0.68	0.68	34054



Random Forest

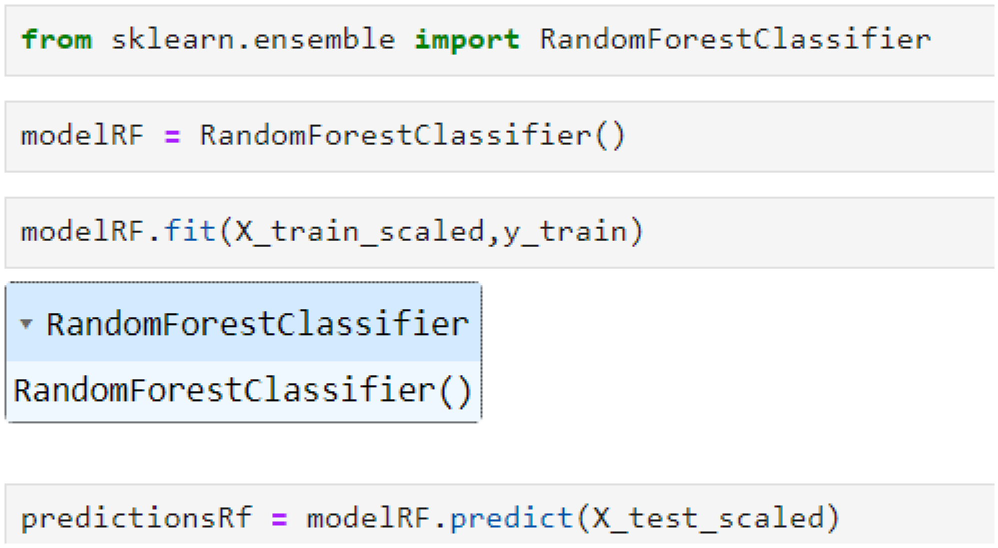
Explanation:

- **Class 0 (Non-bankrupt)**: The Random Forest model achieved a precision of 1.00, indicating that all instances predicted as non-bankrupt are actually non-bankrupt. The recall is 0.98, meaning that 98% of the actual non-bankrupt instances are correctly identified by the model. The f1-score, which combines precision and recall, is 0.99.
- **Class 1 (Bankrupt)**: The precision is 0.98, showing that 98% of the instances predicted as bankrupt are indeed bankrupt. The recall is 1.00, indicating that all actual bankrupt instances are correctly identified by the model. The f1-score is 0.99.
- **Overall Accuracy**: The model achieves an accuracy of 99%, meaning that 99% of the total predictions are correct.
- **Macro Average**: The macro average of precision, recall, and f1-score is 0.99. This average treats all classes equally, providing an overall performance metric without considering class imbalance.

- **Weighted Average**: The weighted average of precision, recall, and f1-score is 0.99, which accounts for the class distribution and provides a balanced performance measure for the model.

Interpretation:

The Random Forest model demonstrates exceptional performance across all metrics. The high precision, recall, and f1-scores for both classes indicate that the model is highly effective at distinguishing between bankrupt and non-bankrupt instances. The slightly lower recall for class 0 (non-bankrupt) compared to class 1 (bankrupt) suggests a minor difference in model performance between the two classes. However, with an overall accuracy of 99%, the model is robust and reliable, making it an excellent choice for predicting bankruptcy. The macro and weighted averages further confirm the model's balanced and high performance, ensuring that both classes are well-represented in the predictions.



```
from sklearn.ensemble import RandomForestClassifier
```

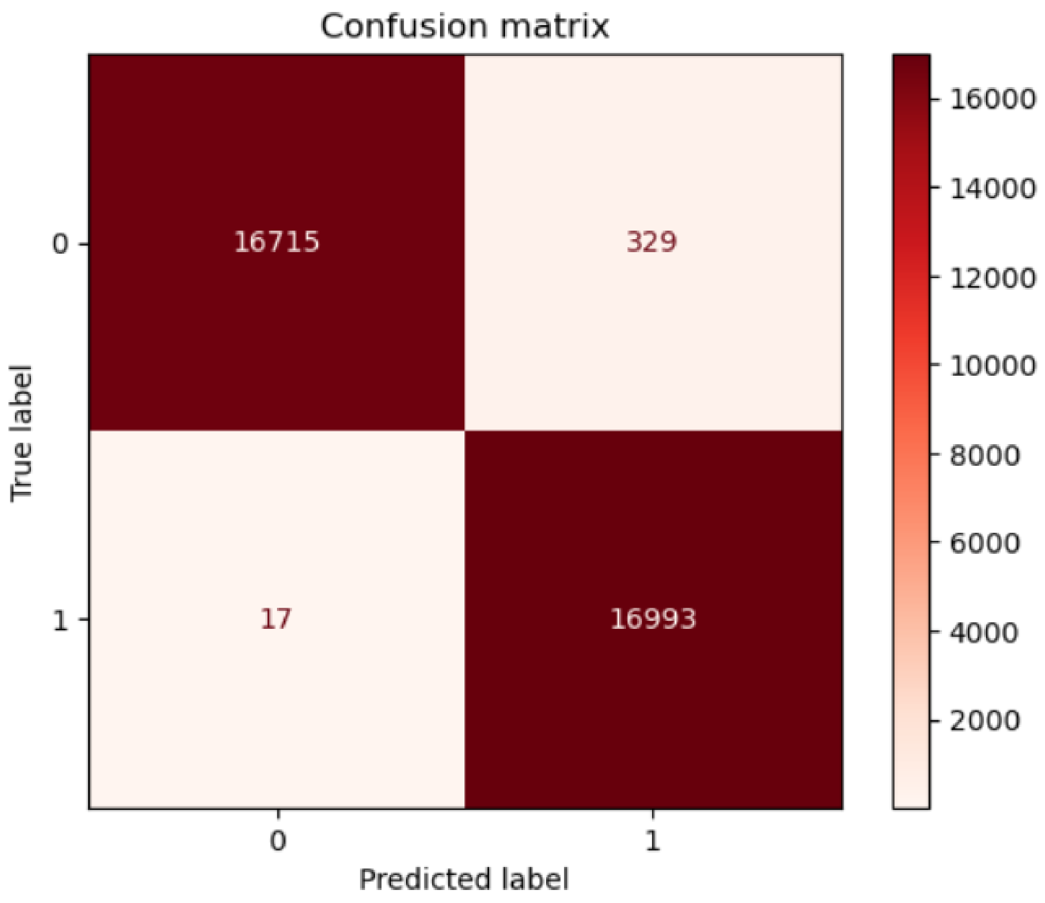
```
modelRF = RandomForestClassifier()
```

```
modelRF.fit(X_train_scaled,y_train)
```

```
▾ RandomForestClassifier
RandomForestClassifier()
```

```
predictionsRf = modelRF.predict(X_test_scaled)
```

	precision	**recall**	**f1-score**	**support**
0	1	0.98	0.99	17044
1	0.98	1	0.99	17010
accuracy			0.99	34054
macro avg	0.99	0.99	0.99	34054
weighted avg	0.99	0.99	0.99	34054



Comparison

1. **Precision**:
   - **SVM**: Precision for class 0 is 0.68 and for class 1 is 0.69. The overall precision is 0.68.
   - **Random Forest**: Precision for class 0 is 1.00 and for class 1 is 0.98. The overall precision is 0.99.
   - **Conclusion**: Random Forest has significantly higher precision for both classes, indicating it makes fewer false positive errors compared to SVM.
2. **Recall**:
   - **SVM**: Recall for class 0 is 0.70 and for class 1 is 0.66. The overall recall is 0.68.
   - **Random Forest**: Recall for class 0 is 0.98 and for class 1 is 1.00. The overall recall is 0.99.
   - **Conclusion**: Random Forest has much higher recall for both classes, meaning it correctly identifies more true positive instances than SVM.
3. **F1-score**:
   - **SVM**: F1-score for class 0 is 0.69 and for class 1 is 0.68. The overall F1-score is 0.68.

- **Random Forest**: F1-score for class 0 is 0.99 and for class 1 is 0.99. The overall F1-score is 0.99.
- **Conclusion**: Random Forest excels in combining precision and recall, achieving near-perfect F1-scores.

4. **Accuracy**:
    - **SVM**: The overall accuracy is 0.68.
    - **Random Forest**: The overall accuracy is 0.99.
    - **Conclusion**: Random Forest outperforms SVM with significantly higher accuracy.
5. **Macro Average**:
    - **SVM**: The macro average for precision, recall, and F1-score is 0.68.
    - **Random Forest**: The macro average for precision, recall, and F1-score is 0.99.
    - **Conclusion**: Random Forest maintains higher performance metrics across both classes.
6. **Weighted Average**:
    - **SVM**: The weighted average for precision, recall, and F1-score is 0.68.
    - **Random Forest**: The weighted average for precision, recall, and F1-score is 0.99.
    - **Conclusion**: Random Forest consistently achieves higher weighted average metrics, accounting for class imbalance more effectively.

Metric	**SVM**	**Random Forest**
Class 0		
Precision	0.68	1
Recall	0.7	0.98
F1-score	0.69	0.99
Support	17044	17044
Class 1		
Precision	0.69	0.98
Recall	0.66	1
F1-score	0.68	0.99
Support	17010	17010
Accuracy	0.68	0.99
Macro avg		
Precision	0.68	0.99
Recall	0.68	0.99
F1-score	0.68	0.99
Support	34054	34054
Weighted avg		

Precision	0.68	0.99
Recall	0.68	0.99
F1-score	0.68	0.99
Support	34054	34054

Practical Recommendations:

Based on the performance metrics, the Random Forest model clearly outperforms the SVM model. Here are practical recommendations for using these models in a financial setting:

1. **Predictive Analysis for Bankruptcy**:
   - **Random Forest Model**: Given its high accuracy and balanced performance across both classes, the Random Forest model is highly suitable for predicting bankruptcy. Financial institutions can deploy this model to identify potential bankruptcy cases with high confidence, enabling proactive measures to mitigate risk.
   - **SVM Model**: While the SVM model shows moderate performance, it may be used as a supplementary tool in scenarios where interpretability is crucial, and simpler models are preferred. However, it should not be relied upon as the primary model for critical predictions.
2. **Risk Management**:
   - The Random Forest model can be integrated into risk management systems to continuously monitor financial health indicators and predict bankruptcy. This allows for timely interventions and strategic decision-making, reducing the likelihood of financial losses.
3. **Credit Scoring**:
   - Financial institutions can use the Random Forest model to enhance credit scoring systems. By accurately predicting bankruptcy, lenders can better assess the creditworthiness of applicants, ensuring more informed lending decisions and reducing default rates.
4. **Portfolio Management**:
   - Investment firms can leverage the Random Forest model to assess the bankruptcy risk of companies within their portfolios. This helps in rebalancing portfolios to mitigate risk and optimize returns.
5. **Regulatory Compliance**:
   - Accurate bankruptcy predictions are essential for regulatory compliance. The Random Forest model can assist in meeting regulatory requirements by providing reliable risk assessments, thereby ensuring adherence to financial regulations.

Overall Conclusion:

The Random Forest model significantly outperforms the SVM model across all evaluation metrics, demonstrating higher precision, recall, and F1-scores for both classes. With an overall accuracy of 99%, the Random Forest model provides a more robust and reliable classification performance, making it a superior choice for predicting bankruptcy in this dataset.