Company Bankruptcy Prediction Report

Introduction: Company bankruptcy prediction plays a vital role in financial decision-making, helping stakeholders assess the financial health and stability of businesses. This report explores the effectiveness of SVM, Logistic Regression, and Naive Bayes models in predicting bankruptcy using financial data from the Taiwan Economic Journal.

Methodology: We utilized a dataset from 1999 to 2009, encompassing 95 financial indicators, to train and evaluate our models. The data was split into training and validation sets, with preprocessing steps including feature scaling and oversampling to handle imbalance.

Results:

Support Vector Machine (SVM):

- **Accuracy:** 88.83%
- **Precision and Recall:** Achieved balanced performance with approximately 91% precision and recall for both bankrupt and non-bankrupt classes.
- **Confusion Matrix:** Showed 1154 true negatives and 1191 true positives, indicating effective discrimination between classes.

Logistic Regression:

- **Accuracy:** Disappointingly low at 35.61%, indicating poor predictive performance.
- **Precision and Recall:** Particularly weak recall for bankrupt class (8%), highlighting challenges in identifying financially distressed companies.
- **Challenges:** The model struggled with class imbalance and may require further feature engineering or alternative approaches.

Naive Bayes:

- **Accuracy:** 53.67%
- **Precision and Recall:** Demonstrated higher recall (93%) for bankrupt class but at the expense of precision, indicating a tendency to misclassify non-bankrupt companies.
- **Performance:** Moderate effectiveness in predicting bankruptcies, with an F1-score of 66% for bankrupt class.

Conclusion: The SVM model outperformed both Logistic Regression and Naive Bayes in predicting company bankruptcies based on financial indicators. Its balanced precision, recall, and overall accuracy make it a reliable tool for stakeholders looking to assess financial risk. However, the Logistic Regression and Naive Bayes models showed limitations in handling the complexities of the dataset, especially in dealing with class imbalance and predicting bankruptcies accurately.

Recommendations:

- **Feature Engineering:** Further exploration of feature selection and engineering techniques could enhance model performance, especially for Logistic Regression and Naive Bayes.
- **Ensemble Methods:** Consider ensemble methods such as Random Forest to leverage the strengths of different models and improve predictive accuracy.
- **Continuous Monitoring:** Implement a system for continuous monitoring and updating of predictive models with real-time financial data to adapt to changing business conditions.

Implications: Effective bankruptcy prediction models are crucial for investors, creditors, and other stakeholders to make informed decisions and mitigate financial risks. As financial markets evolve, robust predictive analytics will remain essential for ensuring sustainable business practices and economic stability.

Module 4 Assignment 2 - Company Bankruptcy Prediction (Kaggle)

Sachin Sharma

MSDS-422

07/13/2024

Management/Research Question

In layman's terms, what is the management/research question of interest, and why would anyone care?

Requirements

Split the training set into an 80% training and 20% validation set and conduct / improve upon previous EDA. Build at least three models: an SVM, a logistic regression model, a Naïve Bayes model. Evaluate each of the models' assumptions. Conduct hyperparameter tuning for the SVM kernel. Evaluate goodness of fit metrics including TPR, FPR, precision, recall, and accuracy on the training and validation sets. Build ROC and Precision / Recall graphs. Evaluate your models' performance on the validation set using the F1-score. Python scikit-learn should be your primary environment for conducting this research.

Libraries to be loaded:

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import KFold, train_test_split
import seaborn as sns
from sklearn.metrics import confusion_matrix, classification_report,
precision_recall_curve, roc_curve, accuracy_score
from sklearn.svm import SVC
from sklearn.naive_bayes import GaussianNB
from imblearn.over_sampling import SMOTE
from sklearn.ensemble import RandomForestClassifier
from sklearn.feature_selection import RFE
```

Read Data into Pandas DF

```
train_df = pd.read_csv("data.csv")
train_df.head()
```

```
   Bankrupt?   ROA(C) before interest and depreciation before interest
\
0          1                                            0.370594

1          1                                            0.464291

2          1                                            0.426071

3          1                                            0.399844

4          1                                            0.465022


    ROA(A) before interest and % after tax  \
0                                 0.424389
1                                 0.538214
2                                 0.499019
3                                 0.451265
4                                 0.538432

    ROA(B) before interest and depreciation after tax  \
0                                            0.405750
1                                            0.516730
2                                            0.472295
3                                            0.457733
4                                            0.522298

    Operating Gross Margin   Realized Sales Gross Margin  \
0                 0.601457                      0.601457
1                 0.610235                      0.610235
2                 0.601450                      0.601364
3                 0.583541                      0.583541
4                 0.598783                      0.598783

    Operating Profit Rate   Pre-tax net Interest Rate  \
0                0.998969                    0.796887
1                0.998946                    0.797380
2                0.998857                    0.796403
3                0.998700                    0.796967
4                0.998973                    0.797366

    After-tax net Interest Rate   Non-industry income and
expenditure/revenue  \
0                      0.808809
0.302646
1                      0.809301
0.303556
2                      0.808388
0.302035
3                      0.808966
0.303350
```

```
4                      0.809304
0.303475

   ...   Net Income to Total Assets   Total assets to GNP price  \
0  ...                     0.716845                    0.009219
1  ...                     0.795297                    0.008323
2  ...                     0.774670                    0.040003
3  ...                     0.739555                    0.003252
4  ...                     0.795016                    0.003878

    No-credit Interval   Gross Profit to Sales  \
0             0.622879                0.601453
1             0.623652                0.610237
2             0.623841                0.601449
3             0.622929                0.583538
4             0.623521                0.598782

    Net Income to Stockholder's Equity   Liability to Equity  \
0                             0.827890              0.290202
1                             0.839969              0.283846
2                             0.836774              0.290189
3                             0.834697              0.281721
4                             0.839973              0.278514

    Degree of Financial Leverage (DFL)  \
0                             0.026601
1                             0.264577
2                             0.026555
3                             0.026697
4                             0.024752

    Interest Coverage Ratio (Interest expense to EBIT)   Net Income
Flag  \
0                                            0.564050
1
1                                            0.570175
1
2                                            0.563706
1
3                                            0.564663
1
4                                            0.575617
1

    Equity to Liability
0              0.016469
1              0.020794
2              0.016474
3              0.023982
4              0.035490
```

```
[5 rows x 96 columns]
```

```
train_df.shape
```

```
(6819, 96)
```

```
train_df.describe()
```

```
         Bankrupt?   ROA(C) before interest and depreciation before
interest  \
count  6819.000000                                      6819.000000
mean      0.032263                                         0.505180
std       0.176710                                         0.060686
min       0.000000                                         0.000000
25%       0.000000                                         0.476527
50%       0.000000                                         0.502706
75%       0.000000                                         0.535563
max       1.000000                                         1.000000

        ROA(A) before interest and % after tax  \
count                              6819.000000
mean                                  0.558625
std                                   0.065620
min                                   0.000000
25%                                   0.535543
50%                                   0.559802
75%                                   0.589157
max                                   1.000000

        ROA(B) before interest and depreciation after tax  \
count                                         6819.000000
mean                                             0.553589
std                                              0.061595
min                                              0.000000
25%                                              0.527277
50%                                              0.552278
75%                                              0.584105
max                                              1.000000

        Operating Gross Margin   Realized Sales Gross Margin  \
count              6819.000000                   6819.000000
mean                  0.607948                      0.607929
```

```
std                        0.016934                              0.016916
min                        0.000000                              0.000000
25%                        0.600445                              0.600434
50%                        0.605997                              0.605976
75%                        0.613914                              0.613842
max                        1.000000                              1.000000

        Operating Profit Rate   Pre-tax net Interest Rate  \
count             6819.000000                 6819.000000
mean                 0.998755                    0.797190
std                  0.013010                    0.012869
min                  0.000000                    0.000000
25%                  0.998969                    0.797386
50%                  0.999022                    0.797464
75%                  0.999095                    0.797579
max                  1.000000                    1.000000

        After-tax net Interest Rate  \
count                   6819.000000
mean                       0.809084
std                        0.013601
min                        0.000000
25%                        0.809312
50%                        0.809375
75%                        0.809469
max                        1.000000

        Non-industry income and expenditure/revenue  ...  \
count                                   6819.000000  ...
mean                                       0.303623  ...
std                                        0.011163  ...
min                                        0.000000  ...
25%                                        0.303466  ...
50%                                        0.303525  ...
75%                                        0.303585  ...
max                                        1.000000  ...

        Net Income to Total Assets   Total assets to GNP price  \
count                  6819.000000                6.819000e+03
mean                      0.807760                1.862942e+07
std                       0.040332                3.764501e+08
min                       0.000000                0.000000e+00
25%                       0.796750                9.036205e-04
50%                       0.810619                2.085213e-03
75%                       0.826455                5.269777e-03
max                       1.000000                9.820000e+09

        No-credit Interval   Gross Profit to Sales  \
count          6819.000000             6819.000000
mean              0.623915                0.607946
```

```
std                 0.012290                0.016934
min                 0.000000                0.000000
25%                 0.623636                0.600443
50%                 0.623879                0.605998
75%                 0.624168                0.613913
max                 1.000000                1.000000

        Net Income to Stockholder's Equity   Liability to Equity  \
count                          6819.000000          6819.000000
mean                              0.840402             0.280365
std                               0.014523             0.014463
min                               0.000000             0.000000
25%                               0.840115             0.276944
50%                               0.841179             0.278778
75%                               0.842357             0.281449
max                               1.000000             1.000000

        Degree of Financial Leverage (DFL)  \
count                          6819.000000
mean                              0.027541
std                               0.015668
min                               0.000000
25%                               0.026791
50%                               0.026808
75%                               0.026913
max                               1.000000

        Interest Coverage Ratio (Interest expense to EBIT)   Net
Income Flag  \
count                                        6819.000000
6819.0
mean                                            0.565358
1.0
std                                             0.013214
0.0
min                                             0.000000
1.0
25%                                             0.565158
1.0
50%                                             0.565252
1.0
75%                                             0.565725
1.0
max                                             1.000000
1.0

        Equity to Liability
count           6819.000000
mean               0.047578
std                0.050014
```

```
min                  0.000000
25%                  0.024477
50%                  0.033798
75%                  0.052838
max                  1.000000

[8 rows x 96 columns]
```

EDA

Get dataframe information:

```
train_df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6819 entries, 0 to 6818
Data columns (total 96 columns):
 #   Column                                                        Non-
Null Count  Dtype
---  ------
-------------  -----
 0   Bankrupt?                                                     6819
non-null   int64
 1    ROA(C) before interest and depreciation before interest     6819
non-null   float64
 2    ROA(A) before interest and % after tax                      6819
non-null   float64
 3    ROA(B) before interest and depreciation after tax           6819
non-null   float64
 4    Operating Gross Margin                                      6819
non-null   float64
 5    Realized Sales Gross Margin                                 6819
non-null   float64
 6    Operating Profit Rate                                       6819
non-null   float64
 7    Pre-tax net Interest Rate                                   6819
non-null   float64
 8    After-tax net Interest Rate                                 6819
non-null   float64
 9    Non-industry income and expenditure/revenue                 6819
non-null   float64
 10   Continuous interest rate (after tax)                        6819
non-null   float64
 11   Operating Expense Rate                                      6819
non-null   float64
 12   Research and development expense rate                       6819
non-null   float64
 13   Cash flow rate                                              6819
non-null   float64
```

```
 14   Interest-bearing debt interest rate                       6819
non-null   float64
 15   Tax rate (A)                                              6819
non-null   float64
 16   Net Value Per Share (B)                                   6819
non-null   float64
 17   Net Value Per Share (A)                                   6819
non-null   float64
 18   Net Value Per Share (C)                                   6819
non-null   float64
 19   Persistent EPS in the Last Four Seasons                   6819
non-null   float64
 20   Cash Flow Per Share                                       6819
non-null   float64
 21   Revenue Per Share (Yuan ¥)                                6819
non-null   float64
 22   Operating Profit Per Share (Yuan ¥)                       6819
non-null   float64
 23   Per Share Net profit before tax (Yuan ¥)                  6819
non-null   float64
 24   Realized Sales Gross Profit Growth Rate                   6819
non-null   float64
 25   Operating Profit Growth Rate                              6819
non-null   float64
 26   After-tax Net Profit Growth Rate                          6819
non-null   float64
 27   Regular Net Profit Growth Rate                            6819
non-null   float64
 28   Continuous Net Profit Growth Rate                         6819
non-null   float64
 29   Total Asset Growth Rate                                   6819
non-null   float64
 30   Net Value Growth Rate                                     6819
non-null   float64
 31   Total Asset Return Growth Rate Ratio                      6819
non-null   float64
 32   Cash Reinvestment %                                       6819
non-null   float64
 33   Current Ratio                                             6819
non-null   float64
 34   Quick Ratio                                               6819
non-null   float64
 35   Interest Expense Ratio                                    6819
non-null   float64
 36   Total debt/Total net worth                                6819
non-null   float64
 37   Debt ratio %                                              6819
non-null   float64
 38   Net worth/Assets                                          6819
```

```
non-null   float64
 39   Long-term fund suitability ratio (A)                                6819
non-null   float64
 40   Borrowing dependency                                                6819
non-null   float64
 41   Contingent liabilities/Net worth                                    6819
non-null   float64
 42   Operating profit/Paid-in capital                                    6819
non-null   float64
 43   Net profit before tax/Paid-in capital                               6819
non-null   float64
 44   Inventory and accounts receivable/Net value                         6819
non-null   float64
 45   Total Asset Turnover                                                6819
non-null   float64
 46   Accounts Receivable Turnover                                        6819
non-null   float64
 47   Average Collection Days                                             6819
non-null   float64
 48   Inventory Turnover Rate (times)                                     6819
non-null   float64
 49   Fixed Assets Turnover Frequency                                     6819
non-null   float64
 50   Net Worth Turnover Rate (times)                                     6819
non-null   float64
 51   Revenue per person                                                  6819
non-null   float64
 52   Operating profit per person                                         6819
non-null   float64
 53   Allocation rate per person                                          6819
non-null   float64
 54   Working Capital to Total Assets                                     6819
non-null   float64
 55   Quick Assets/Total Assets                                           6819
non-null   float64
 56   Current Assets/Total Assets                                         6819
non-null   float64
 57   Cash/Total Assets                                                   6819
non-null   float64
 58   Quick Assets/Current Liability                                      6819
non-null   float64
 59   Cash/Current Liability                                              6819
non-null   float64
 60   Current Liability to Assets                                         6819
non-null   float64
 61   Operating Funds to Liability                                        6819
non-null   float64
 62   Inventory/Working Capital                                           6819
non-null   float64
```

```
 63   Inventory/Current Liability                              6819
non-null   float64
 64   Current Liabilities/Liability                            6819
non-null   float64
 65   Working Capital/Equity                                   6819
non-null   float64
 66   Current Liabilities/Equity                               6819
non-null   float64
 67   Long-term Liability to Current Assets                    6819
non-null   float64
 68   Retained Earnings to Total Assets                        6819
non-null   float64
 69   Total income/Total expense                               6819
non-null   float64
 70   Total expense/Assets                                     6819
non-null   float64
 71   Current Asset Turnover Rate                              6819
non-null   float64
 72   Quick Asset Turnover Rate                                6819
non-null   float64
 73   Working capitcal Turnover Rate                           6819
non-null   float64
 74   Cash Turnover Rate                                       6819
non-null   float64
 75   Cash Flow to Sales                                       6819
non-null   float64
 76   Fixed Assets to Assets                                   6819
non-null   float64
 77   Current Liability to Liability                           6819
non-null   float64
 78   Current Liability to Equity                              6819
non-null   float64
 79   Equity to Long-term Liability                            6819
non-null   float64
 80   Cash Flow to Total Assets                                6819
non-null   float64
 81   Cash Flow to Liability                                   6819
non-null   float64
 82   CFO to Assets                                            6819
non-null   float64
 83   Cash Flow to Equity                                      6819
non-null   float64
 84   Current Liability to Current Assets                      6819
non-null   float64
 85   Liability-Assets Flag                                    6819
non-null   int64
 86   Net Income to Total Assets                               6819
non-null   float64
 87   Total assets to GNP price                                6819
```

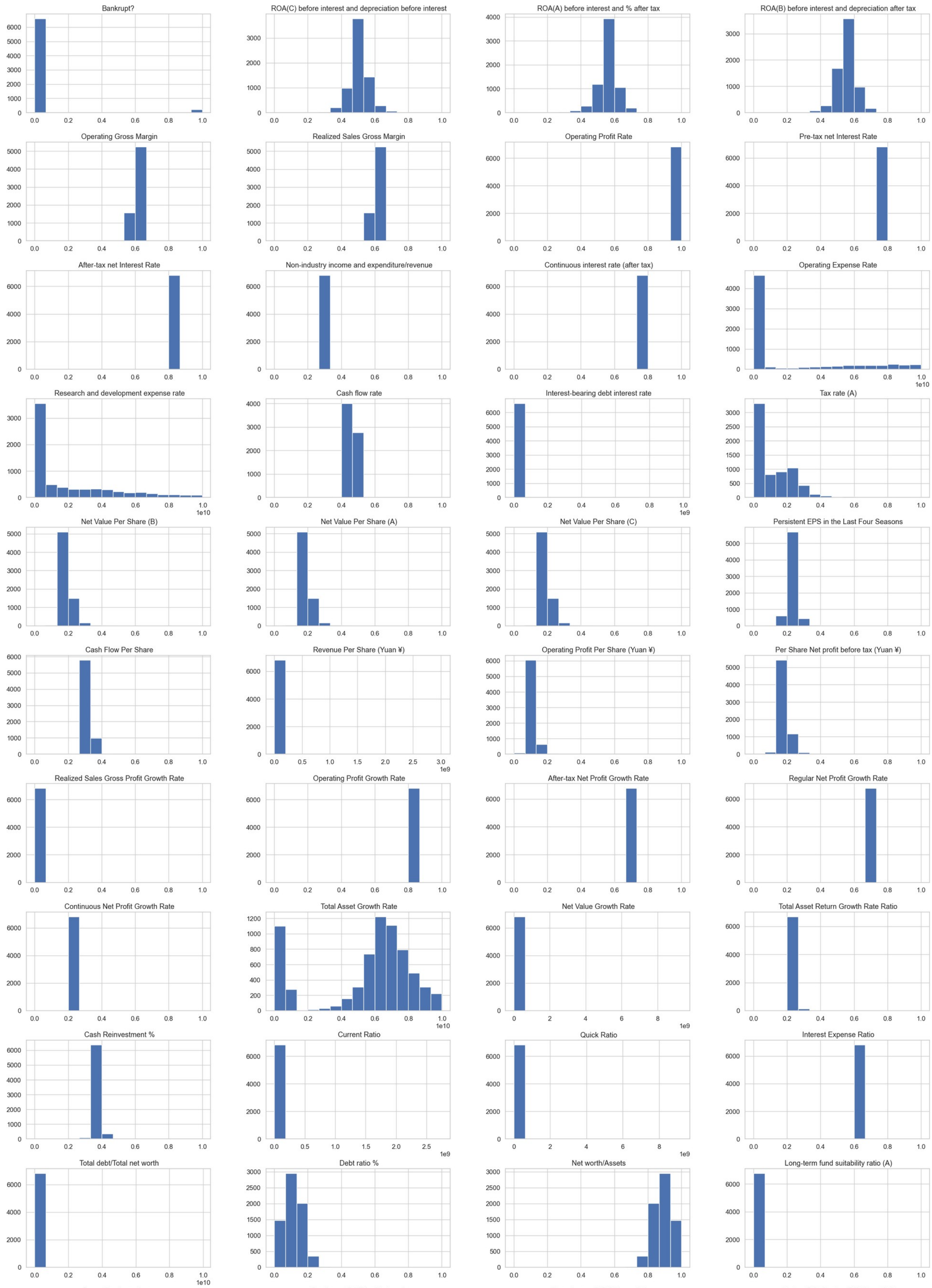
```
non-null   float64
 88   No-credit Interval                                              6819
non-null   float64
 89   Gross Profit to Sales                                           6819
non-null   float64
 90   Net Income to Stockholder's Equity                              6819
non-null   float64
 91   Liability to Equity                                             6819
non-null   float64
 92   Degree of Financial Leverage (DFL)                              6819
non-null   float64
 93   Interest Coverage Ratio (Interest expense to EBIT)              6819
non-null   float64
 94   Net Income Flag                                                 6819
non-null   int64
 95   Equity to Liability                                             6819
non-null   float64
dtypes: float64(93), int64(3)
memory usage: 5.0 MB
```

There are no null values.

```
train_df.isna().sum().sum()

0

# Separate numeric (quantitative) and categorical (nominal) variables
numeric_vars = train_df.select_dtypes(include=['float64',
'int64']).columns.tolist()
categorical_vars =
train_df.select_dtypes(include=['object']).columns.tolist()

print("\nNumeric Variables:")
print(len(numeric_vars))

print("\nCategorical Variables:")
print(len(categorical_vars))


Numeric Variables:
96

Categorical Variables:
0
```

Let's plot a histogram of the variables.

```
sns.set(style='whitegrid', font_scale=1.1, rc={'figure.figsize': [30,
102]})
train_df[train_df.columns].hist(bins=15, layout=(24, 4));
```


Bankrupt?
ROA(C) before interest and depreciation before interest
ROA(A) before interest and % after tax
ROA(B) before interest and depreciation after tax
Operating Gross Margin
Realized Sales Gross Margin
Operating Profit Rate
Pre-tax net Interest Rate
After-tax net Interest Rate
Non-industry income and expenditure/revenue
Continuous interest rate (after tax)
Operating Expense Rate
Research and development expense rate
Cash flow rate
Interest-bearing debt interest rate
Tax rate (A)
Net Value Per Share (B)
Net Value Per Share (A)
Net Value Per Share (C)
Persistent EPS in the Last Four Seasons
Cash Flow Per Share
Revenue Per Share (Yuan ¥)
Operating Profit Per Share (Yuan ¥)
Per Share Net profit before tax (Yuan ¥)
Realized Sales Gross Profit Growth Rate
Operating Profit Growth Rate
After-tax Net Profit Growth Rate
Regular Net Profit Growth Rate
Continuous Net Profit Growth Rate
Total Asset Growth Rate
Net Value Growth Rate
Total Asset Return Growth Rate Ratio
Cash Reinvestment %
Current Ratio
Quick Ratio
Interest Expense Ratio
Total debt/Total net worth
Debt ratio %
Net worth/Assets
Long-term fund suitability ratio (A)
Borrowing dependency
Contingent liabilities/Net worth
Operating profit/Paid-in capital
Net profit before tax/Paid-in capital

Let's create a correlation heatmap.

```
correlation = train_df.corr()
fig, ax = plt.subplots(figsize = (15,15));
sns.heatmap(correlation, ax = ax, linewidth = 0.1);
```

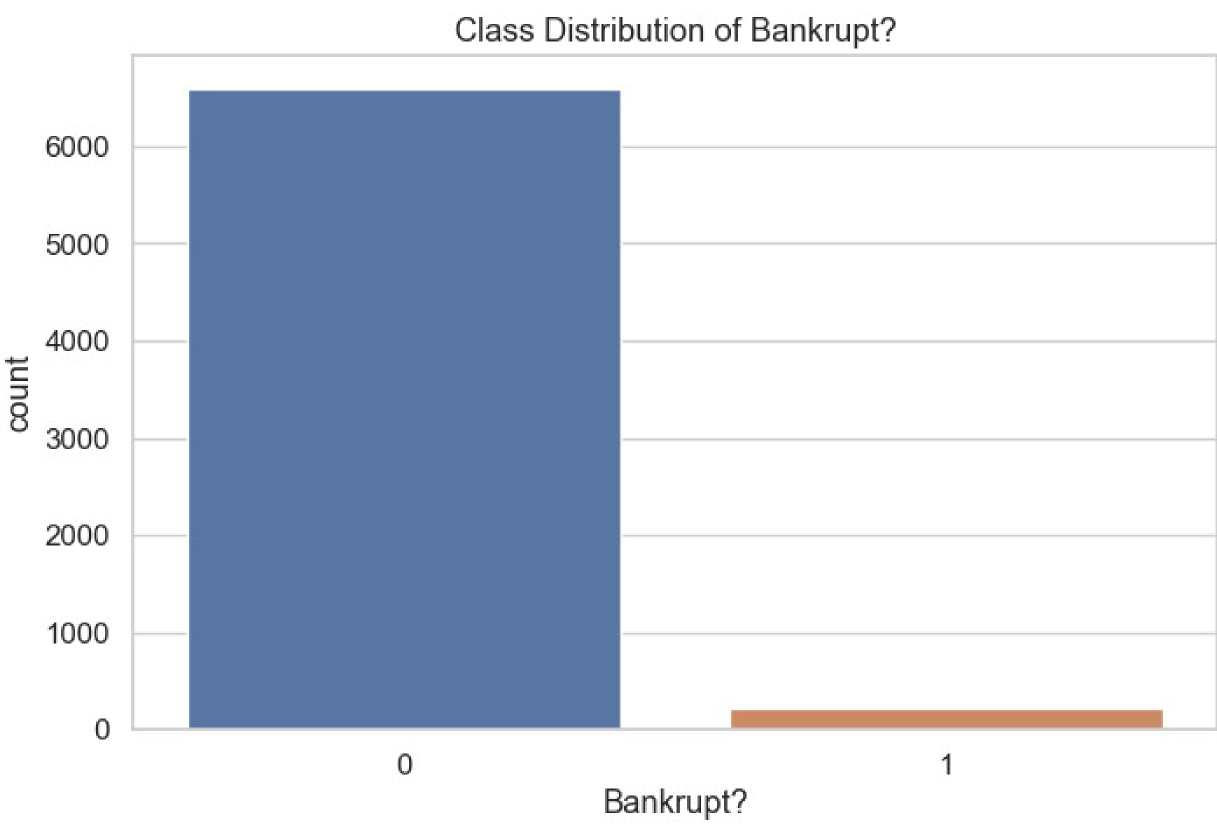
Just over three percent of the data contain rows listed as 'Bankrupt?' (the dependent variable) = 1.

Can we mitigate the overwhelming 'Bankrupt?' = 0 bias using some oversampling methodology...?

```
total_bankrupt = train_df['Bankrupt?'].sum()
pct_bankrput = total_bankrupt/len(train_df['Bankrupt?'])*100
print('Num bankrupt: %d, %% of sample: %.2f%%' %(total_bankrupt,
pct_bankrput))
```

```
Num bankrupt: 220, % of sample: 3.23%
```

```
# Class distribution of the target variable
sns.set(style='whitegrid', font_scale=1.1, rc={"figure.figsize": [8,
5]})
sns.countplot(x='Bankrupt?', data=train_df)
plt.title('Class Distribution of Bankrupt?')
plt.show()
```




```
print('Financially stable:',
round(train_df['Bankrupt?'].value_counts()[0] / len(train_df) * 100,2)
,'%')
print('Financially unstable:',
round(train_df['Bankrupt?'].value_counts()[1] / len(train_df) * 100,
2), '%')
```

```
Financially stable: 96.77 %
Financially unstable: 3.23 %
```

We see the data is highly skewed towards, Financially stable. If we train the model on this dataset, our prediction will be biased towards Financially stabled.

We will balance the dataset, to train our model.

```
train_df_X = train_df.copy()
train_df_y = train_df_X['Bankrupt?']
train_df_X.drop(['Bankrupt?'], axis=1, inplace=True)

train_df_X.shape
```

```
(6819, 95)
```

Oversampling

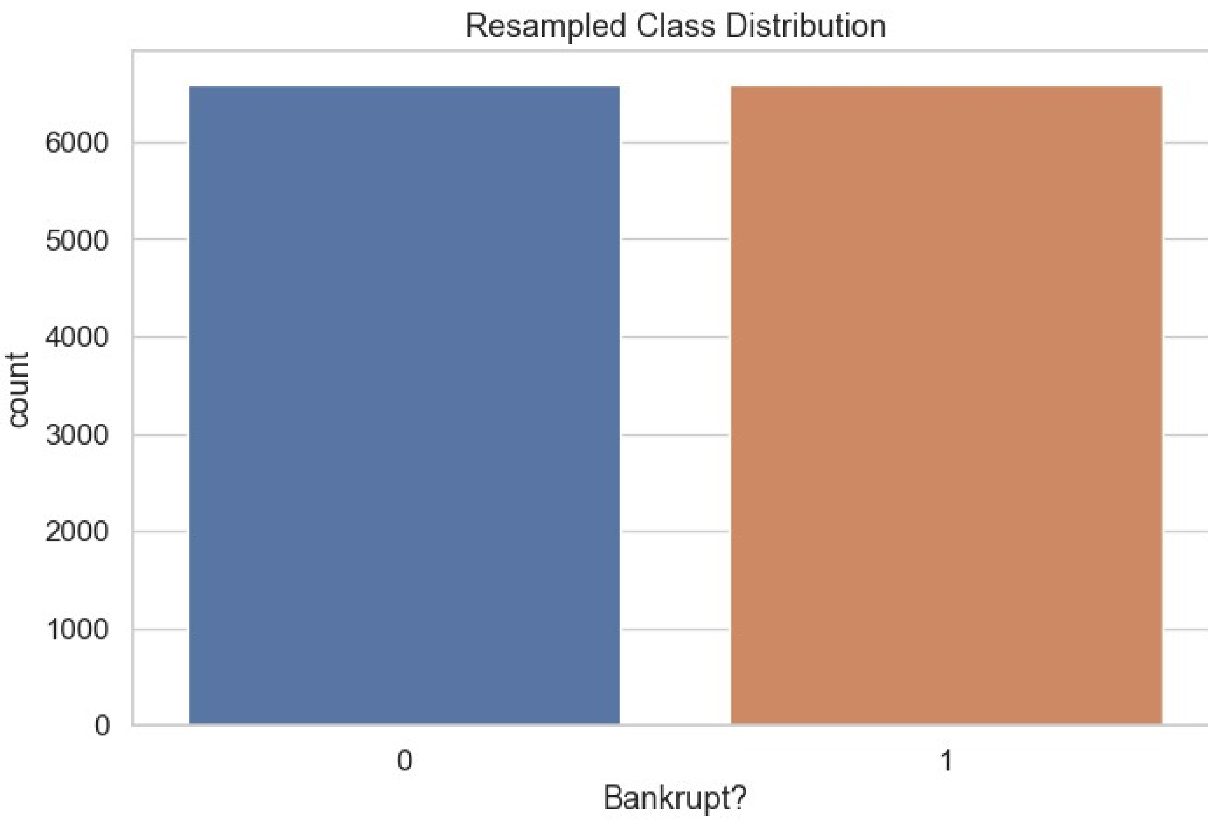
We have the data between 'Bankrupt?' and not 'Bankrupt?' now even using SMOTE oversampling.

```
oversample = SMOTE(random_state=42)
train_df_X,train_df_y = oversample.fit_resample(train_df_X,train_df_y)

total_bankrupt = train_df_y.sum()
pct_bankrput = total_bankrupt/len(train_df_y)*100
print('Num bankrupt: %d, %% of sample: %.2f%%' %(total_bankrupt,
pct_bankrput))
```

```
Num bankrupt: 6599, % of sample: 50.00%
```

```
# Display the new class distribution
print("Resampled class distribution:\n", train_df_y.value_counts())
sns.countplot(x=train_df_y)
plt.title('Resampled Class Distribution')
plt.show()
```

```
Resampled class distribution:
 Bankrupt?
1    6599
0    6599
Name: count, dtype: int64
```



Feature Selection

Let's attempt to reduce the 95 independent variables into a smaller subset of more correlated features using a Feature Selection methodology.

We will use Recursive Feature Elimination with the Random Forest Classifier to narrow down from 95 to 15 features.

```
X_train,X_val,y_train,y_val =
train_test_split(train_df_X,train_df_y,test_size=0.2,random_state=42)

%%time
select = RFE(RandomForestClassifier(n_estimators=100,
random_state=42), n_features_to_select=15)

select.fit(X_train, y_train)

mask = select.get_support()

X_train_rfe = select.transform(X_train)
X_test_rfe = select.transform(X_val)

score = RandomForestClassifier().fit(X_train_rfe,
y_train).score(X_test_rfe, y_val)
```

```
print("Test score: {:.3f}".format(score), " number of features:
{}".format(15))

features = pd.DataFrame({'features':list(train_df.iloc[:,1:].keys()),
'select':list(mask)})
features = list(features[features['select']==True]['features'])
features.append('Bankrupt?')
```

```
Test score: 0.963  number of features: 15
CPU times: user 5min 20s, sys: 309 ms, total: 5min 20s
Wall time: 5min 21s
```

```
features = features[:-1]
```

```
features
```

```
[' Pre-tax net Interest Rate',
 ' After-tax net Interest Rate',
 ' Continuous interest rate (after tax)',
 ' Interest-bearing debt interest rate',
 ' Persistent EPS in the Last Four Seasons',
 ' Quick Ratio',
 ' Interest Expense Ratio',
 ' Total debt/Total net worth',
 ' Debt ratio %',
 ' Borrowing dependency',
 ' Net profit before tax/Paid-in capital',
 ' Retained Earnings to Total Assets',
 ' Cash Turnover Rate',
 ' Net Income to Total Assets',
 " Net Income to Stockholder's Equity"]
```

```
train_df_X_reduced = train_df_X[features]
train_df_X_reduced.shape
```

```
(13198, 15)
```

Scale Data

```
scaler = StandardScaler()
train_df_X_reduced_scaled = scaler.fit_transform(train_df_X_reduced)
train_df_X_reduced_scaled =
pd.DataFrame(scaler.transform(train_df_X_reduced),
index=train_df_X_reduced.index, columns=train_df_X_reduced.columns)
```

```
train_df_X_reduced_scaled.head(5)
```

```
   Pre-tax net Interest Rate  After-tax net Interest Rate  \
0                  -0.007885                    -0.004726
1                   0.044809                     0.044772
2                  -0.059589                    -0.047220
```

```
3                   0.000646                            0.011013
4                   0.043301                            0.045073

   Continuous interest rate (after tax)  Interest-bearing debt
interest rate  \
0                             -0.018885                            -
0.123653
1                              0.037666                            -
0.123653
2                             -0.094945                            -
0.123653
3                              0.008911                            -
0.123653
4                              0.042448                            -
0.123653

   Persistent EPS in the Last Four Seasons  Quick Ratio  \
0                                -1.107635    -0.046312
1                                -0.005620    -0.046312
2                                -0.790904    -0.046312
3                                -0.427056    -0.046312
4                                 0.093849    -0.046312

   Interest Expense Ratio  Total debt/Total net worth  Debt ratio %  \
0               -0.065798                   -0.044923      0.942596
1                0.359610                   -0.044923      0.358094
2               -0.091859                   -0.044923      0.941625
3               -0.043224                   -0.044923      0.041570
4                0.431516                   -0.044923     -0.680319

   Borrowing dependency  Net profit before tax/Paid-in capital  \
0              0.179148                              -0.866768
1             -0.140027                               0.101639
2             -0.084970                              -0.547788
3             -0.069635                              -0.562532
4             -0.180963                               0.055067

   Retained Earnings to Total Assets  Cash Turnover Rate  \
0                          -0.555638           -0.701870
1                           0.346746            0.091643
2                          -0.339167           -0.583546
3                          -0.436444           -0.087991
4                          -0.211240           -0.558944

   Net Income to Total Assets  Net Income to Stockholder's Equity
0                   -0.896223                           -0.117565
1                    0.320238                            0.150567
2                    0.000392                            0.079645
3                   -0.544098                            0.033535
4                    0.315877                            0.150644
```

- Split the training set into an 80% training and 20% validation set.

```
# un-scaled split data
X_train,X_val,y_train,y_val=train_test_split(train_df_X_reduced,train_
df_y,test_size=0.2,random_state=42)

# scaled split data
X_train_scaled,X_val_scaled,y_train_scaled,y_val_scaled =
train_test_split(train_df_X_reduced_scaled,train_df_y,test_size=0.2,ra
ndom_state=42)
```

MODELING

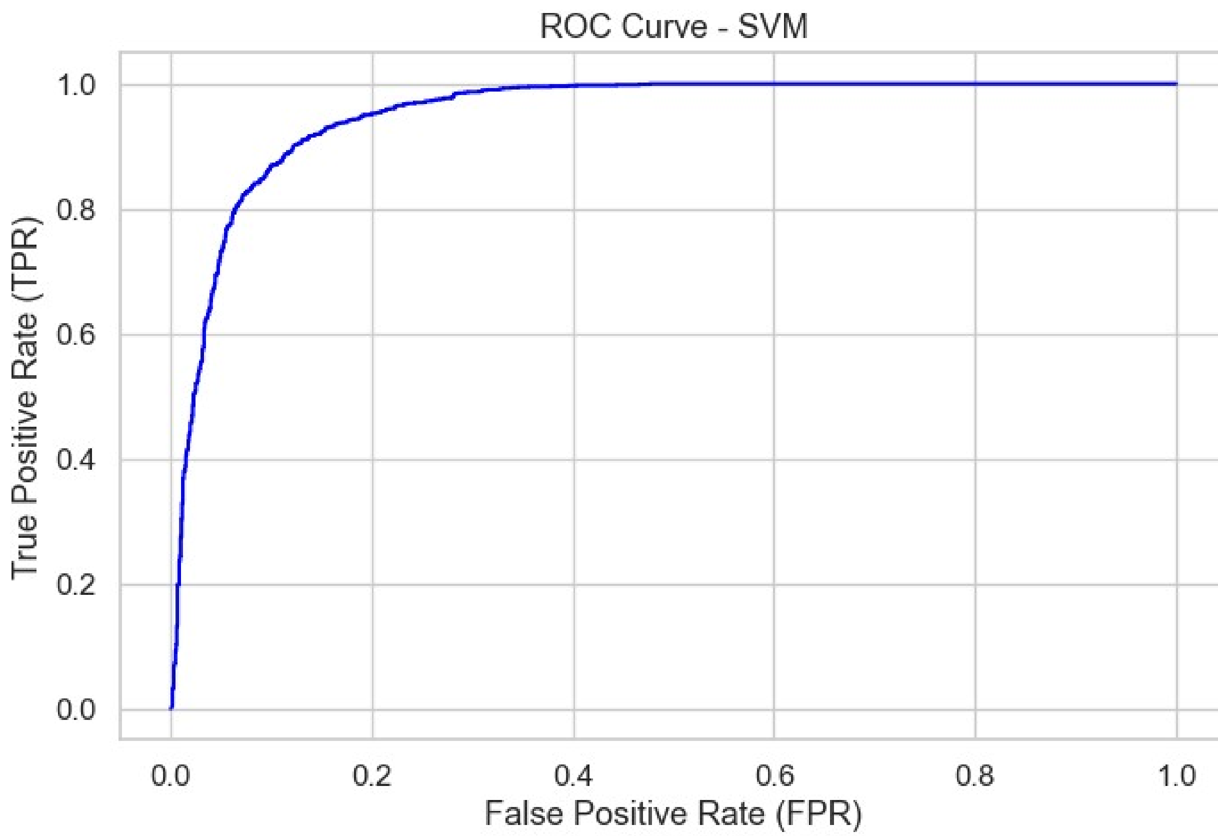
```
# set up a KFold cross-validation rule
K = 10
kf = KFold(n_splits=K, shuffle=True, random_state=42)
```

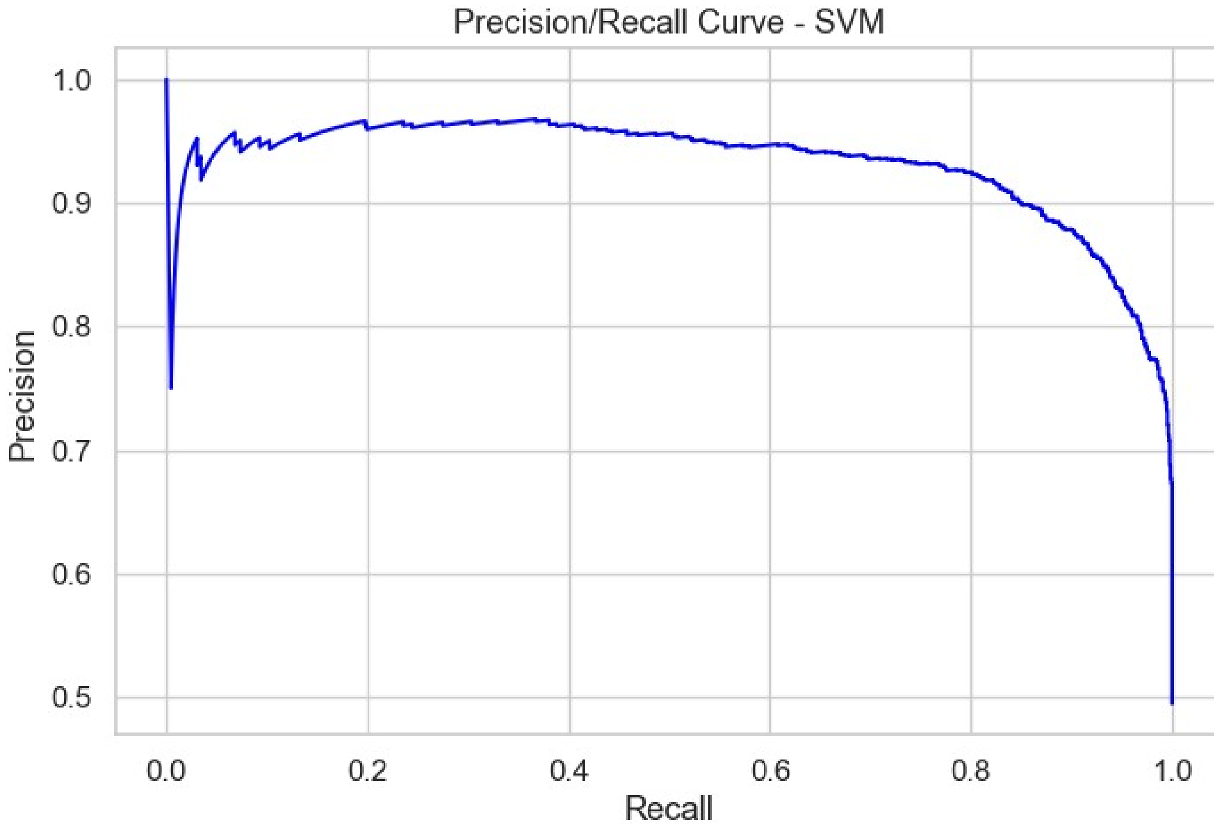
Support Vector Machine

```
#training model
svc = SVC(kernel='linear', gamma=0.01, C=2, probability=True)
svc.fit(X_train_scaled,y_train_scaled)

#getting confusion matrix
y_pred = svc.predict(X_val_scaled)
cm = confusion_matrix(y_val_scaled,y_pred)
print('confusion matrix:\n',cm)
print('accuracy score = ',accuracy_score(y_val_scaled,y_pred))
print("Classification Report:\
n",classification_report(y_val_scaled,y_pred))
```

```
confusion matrix:
 [[1154  180]
 [ 115 1191]]
accuracy score =  0.8882575757575758
Classification Report:
               precision    recall  f1-score   support

           0       0.91      0.87      0.89      1334
           1       0.87      0.91      0.89      1306

    accuracy                           0.89      2640
   macro avg       0.89      0.89      0.89      2640
weighted avg       0.89      0.89      0.89      2640
```

```
y_pred = svc.predict_proba(X_val_scaled)[::,1]
fpr, tpr, _ = roc_curve(y_val_scaled, y_pred)

#create ROC curve
plt.plot(fpr,tpr, color="blue")
plt.title('ROC Curve - SVM')
plt.xlabel('False Positive Rate (FPR)')
plt.ylabel('True Positive Rate (TPR)')
plt.show()

#calculate precision and recall
precision, recall, thresholds = precision_recall_curve(y_val_scaled,
y_pred)

#create precision recall curve
fig, ax = plt.subplots()
ax.plot(recall, precision, color='blue')

#add axis labels to plot
ax.set_title('Precision/Recall Curve - SVM')
ax.set_ylabel('Precision')
ax.set_xlabel('Recall')

#display plot
plt.show()
```


ROC Curve - SVM
True Positive Rate (TPR)
False Positive Rate (FPR)
0.0
0.2
0.4
0.6
0.8
1.0



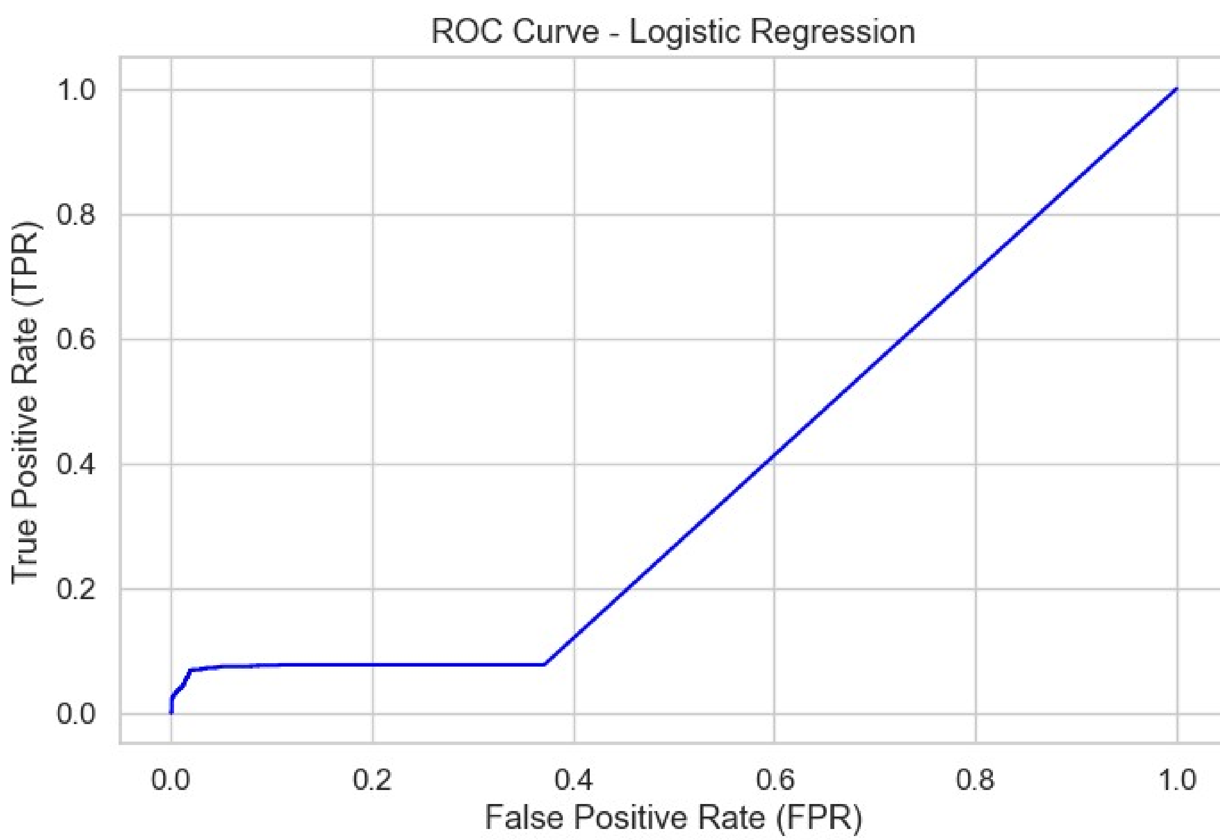
The SVM model applied to the dataset achieved a commendable accuracy of 88.83%. With precision and recall scores around 89% for both bankrupt and non-bankrupt classes, the model demonstrates robust performance in distinguishing between financially stable and distressed companies. The balanced performance metrics indicate that the SVM model effectively utilizes the dataset's features to predict company bankruptcy with high accuracy and reliability

Logistic Regression

```
#training model
lr = LogisticRegression(max_iter = 10000)
lr.fit(X_train_scaled,y_train_scaled)

#getting confusion matrix
y_pred = lr.predict(X_val)
cm = confusion_matrix(y_val,y_pred)
print('confusion matrix:\n',cm)
lra = accuracy_score(y_val,y_pred)
print('accuracy score = ',lra)
print("Classification Report:\n",classification_report(y_val,y_pred))
```

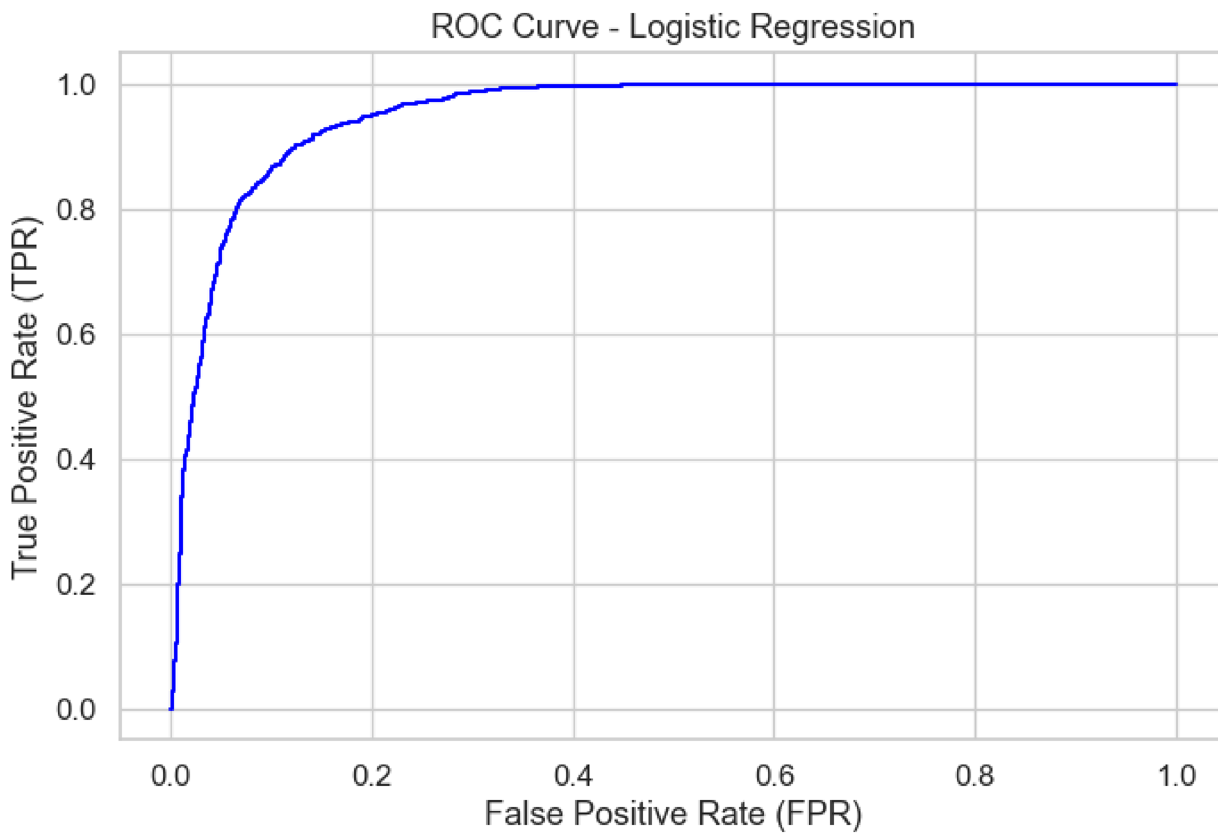
```
confusion matrix:
 [[ 840  494]
 [1206  100]]
```

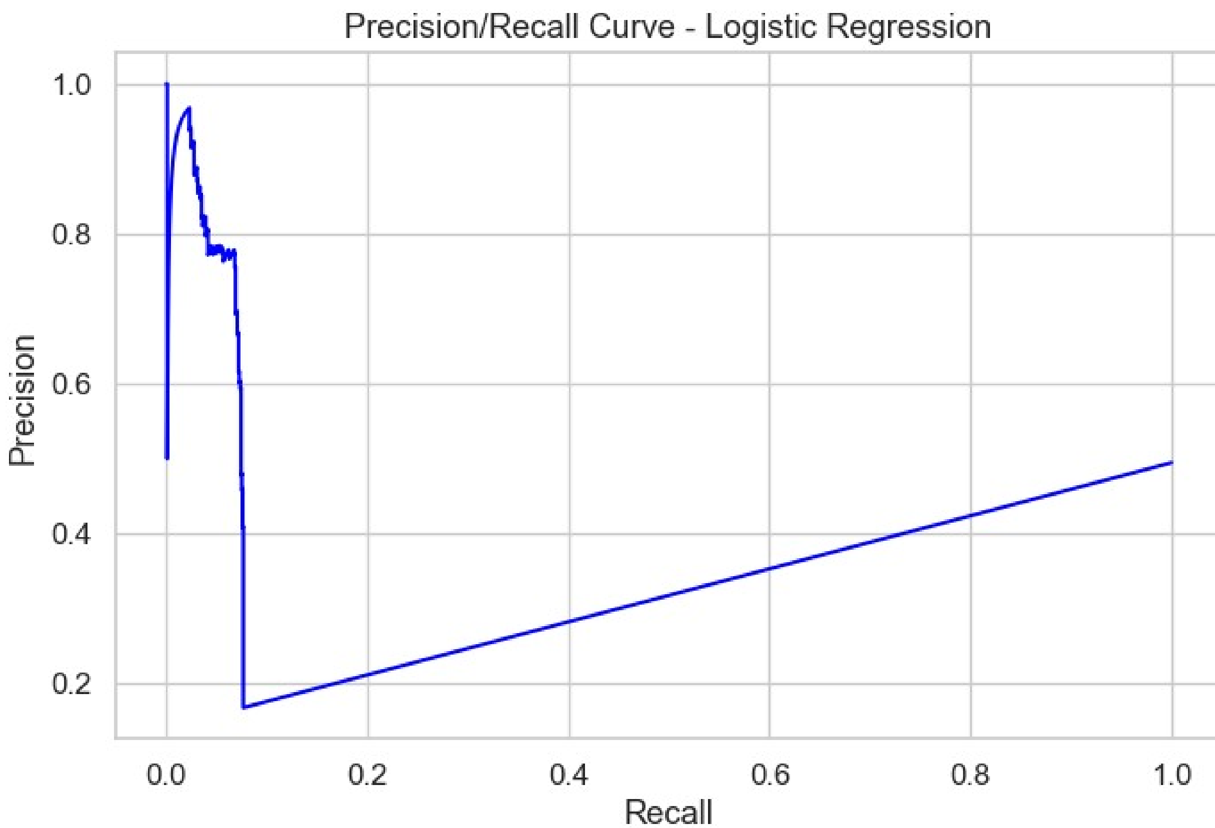
```
accuracy score =  0.3560606060606061
Classification Report:
              precision    recall  f1-score   support

           0       0.41      0.63      0.50      1334
           1       0.17      0.08      0.11      1306

    accuracy                           0.36      2640
   macro avg       0.29      0.35      0.30      2640
weighted avg       0.29      0.36      0.30      2640


#define metrics
y_pred = lr.predict_proba(X_val)[::,1]
fpr, tpr, _ = roc_curve(y_val,  y_pred)

#create ROC curve
plt.plot(fpr,tpr, color="blue")
plt.title('ROC Curve - Logistic Regression')
plt.xlabel('False Positive Rate (FPR)')
plt.ylabel('True Positive Rate (TPR)')
plt.show()
```



Precision/Recall Curve

```
# ROC Curve
y_pred_proba = lr.predict_proba(X_val_scaled)[::, 1]
fpr, tpr, _ = roc_curve(y_val, y_pred_proba)

plt.figure()
plt.plot(fpr, tpr, color="blue")
plt.title('ROC Curve - Logistic Regression')
plt.xlabel('False Positive Rate (FPR)')
plt.ylabel('True Positive Rate (TPR)')
plt.show()

#calculate precision and recall
precision, recall, thresholds = precision_recall_curve(y_val, y_pred)

#create precision recall curve
fig, ax = plt.subplots()
ax.plot(recall, precision, color='blue')

#add axis labels to plot
ax.set_title('Precision/Recall Curve - Logistic Regression')
ax.set_ylabel('Precision')
ax.set_xlabel('Recall')

#display plot
plt.show()
```


ROC Curve - Logistic Regression
True Positive Rate (TPR)
False Positive Rate (FPR)
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0



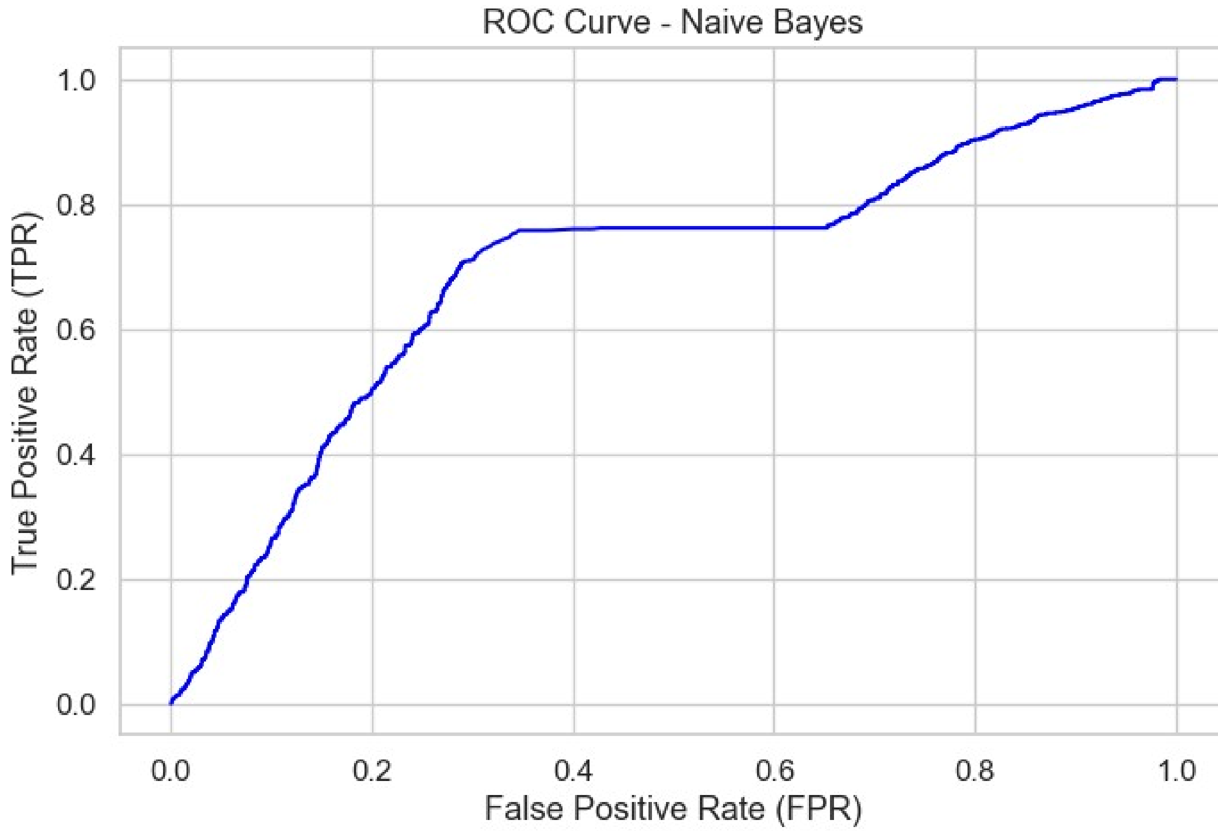
The Logistic Regression model applied to the dataset yielded disappointing results with an accuracy of only 35.61%. The model struggled particularly with recall, achieving 8% for class 1 (bankrupt), indicating a high rate of false negatives. Precision was also low, with values of 41% for class 0 (non-bankrupt) and 17% for class 1, highlighting challenges in correctly identifying bankrupt companies. Overall, the model's performance suggests limitations in effectively leveraging the dataset's features for bankruptcy prediction compared to the SVM model.

Naive Bayes

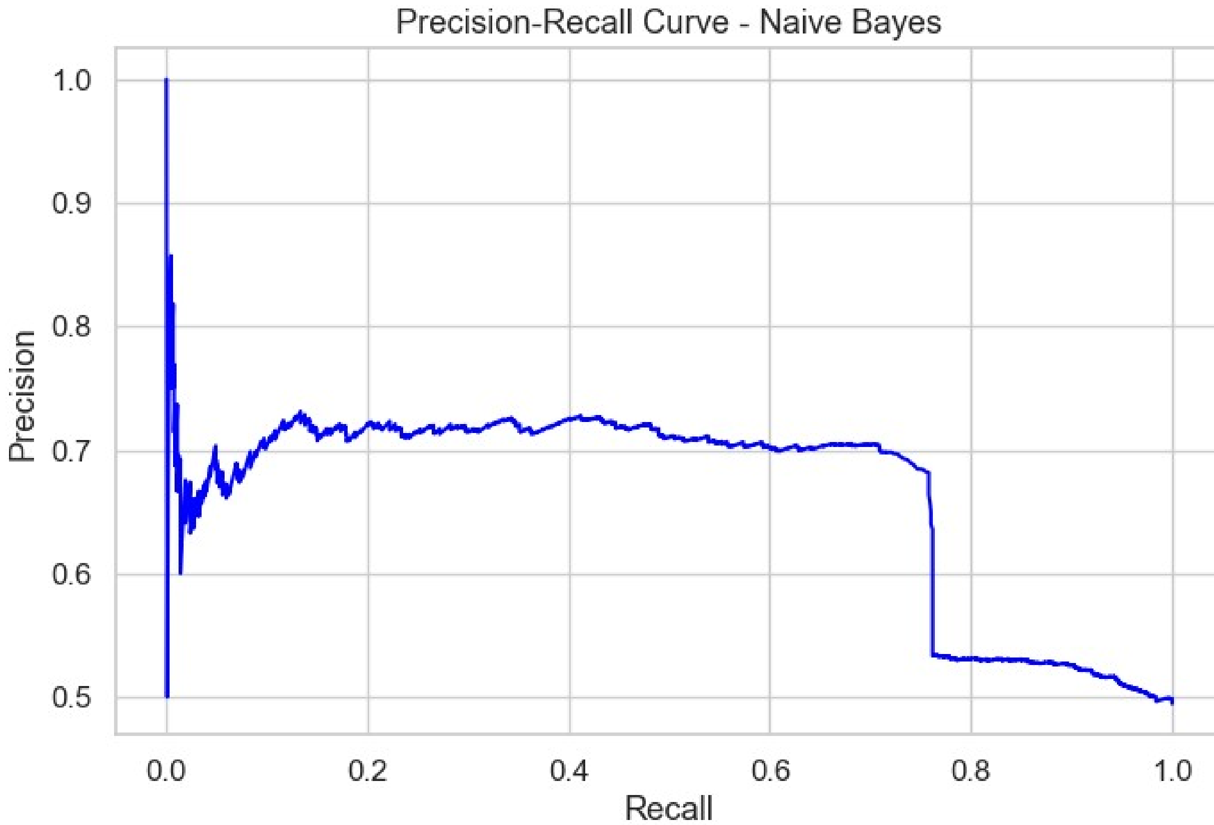
```
#training model
nb = GaussianNB()
nb.fit(X_train,y_train)

#getting confusion matrix
y_pred = nb.predict(X_val)
cm = confusion_matrix(y_val,y_pred)
print('confusion matrix:\n',cm)

#checking accuracy
nba = accuracy_score(y_val,y_pred)
print('accuracy score = ',accuracy_score(y_val,y_pred))
print("Classification Report:\n",classification_report(y_val,y_pred))
```

```
confusion matrix:
 [[ 207 1127]
 [  96 1210]]
accuracy score =  0.5367424242424242
Classification Report:
               precision    recall  f1-score   support

           0       0.68      0.16      0.25      1334
           1       0.52      0.93      0.66      1306

    accuracy                           0.54      2640
   macro avg       0.60      0.54      0.46      2640
weighted avg       0.60      0.54      0.46      2640
```

```python
# ROC Curve
y_pred_proba = nb.predict_proba(X_val)[:, 1]
fpr, tpr, _ = roc_curve(y_val, y_pred_proba)

plt.figure()
plt.plot(fpr, tpr, color="blue")
plt.title('ROC Curve - Naive Bayes')
plt.xlabel('False Positive Rate (FPR)')
plt.ylabel('True Positive Rate (TPR)')
plt.show()

# Precision-Recall Curve (optional)
precision, recall, _ = precision_recall_curve(y_val, y_pred_proba)

plt.figure()
plt.plot(recall, precision, color='blue')
plt.title('Precision-Recall Curve - Naive Bayes')
plt.xlabel('Recall')
plt.ylabel('Precision')
plt.show()
```



ROC Curve - Naive Bayes
True Positive Rate (TPR)
False Positive Rate (FPR)
0.0
0.2
0.4
0.6
0.8
1.0



The Naive Bayes model applied to the dataset achieved an accuracy of 53.67%. It showed a significant disparity in performance between classes, with a higher recall of 93% for class 1 (bankrupt) compared to 16% for class 0 (non-bankrupt). Precision was also higher for class 1 at 52%, indicating the model's ability to better identify bankrupt companies but at the cost of misclassifying non-bankrupt companies. The overall F1-score was 25% for class 0 and 66% for class 1, reflecting the model's moderate performance in predicting bankruptcies based on the dataset's features.

CONCLUSION

Management/Research Question

In layman's terms, what is the management/research question of interest, and why would anyone care?

The management/research question of interest could be:

Research Question: Can we accurately predict whether a company is likely to go bankrupt based on financial indicators?

Layman's Explanation: The question aims to determine if we can use financial information to forecast whether a company might face financial distress and potentially go out of business.

Why it Matters: Understanding and predicting company bankruptcy is crucial for various stakeholders, including investors, creditors, and even employees. It helps them make informed decisions about investments, loans, and employment stability. By identifying early warning signs of financial distress, stakeholders can take proactive measures to mitigate risks or capitalize on opportunities effectively.