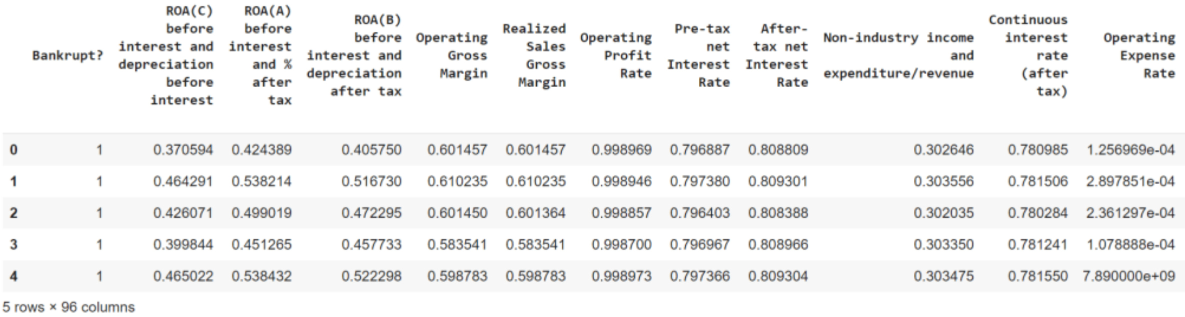
Bankruptcy Prediction Model

Dataset: Collected from Kaggle (Taiwan Economic Journal)

```
import numpy as np

import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LogisticRegression

from sklearn.metrics import accuracy_score


data = pd.read_csv("bank.csv")

data.head()
```

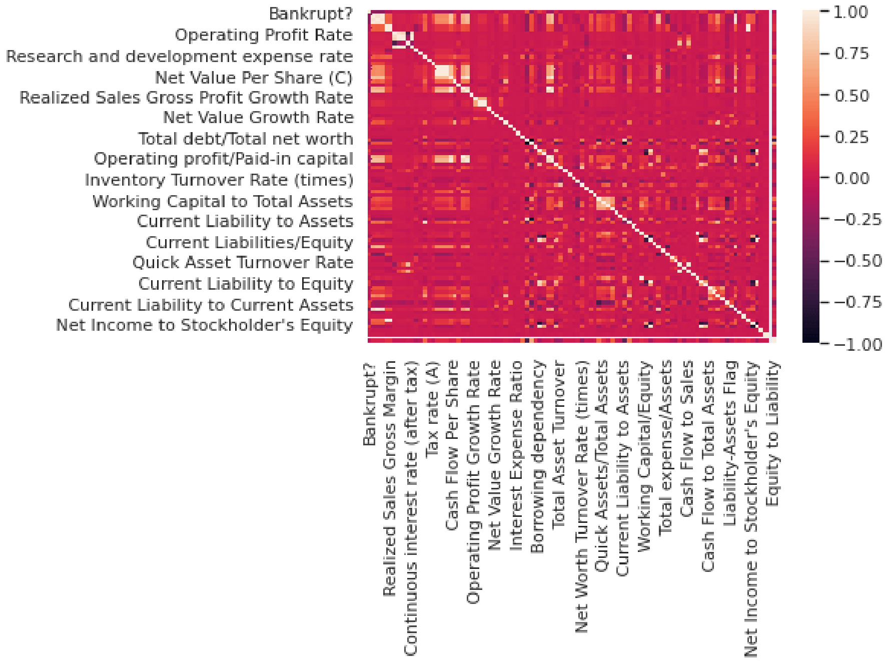


	Bankrupt?	ROA(C) before interest and depreciation before interest	ROA(A) before interest and % after tax	ROA(B) before interest and depreciation after tax	Operating Gross Margin	Realized Sales Gross Margin	Operating Profit Rate	Pre-tax net Interest Rate	After-tax net Interest Rate	Non-industry income and expenditure/revenue	Continuous interest rate (after tax)	Operating Expense Rate
0	1	0.370594	0.424389	0.405750	0.601457	0.601457	0.998969	0.796887	0.808809	0.302646	0.780985	1.256969e-04
1	1	0.464291	0.538214	0.516730	0.610235	0.610235	0.998946	0.797380	0.809301	0.303556	0.781506	2.897851e-04
2	1	0.426071	0.499019	0.472295	0.601450	0.601364	0.998857	0.796403	0.808388	0.302035	0.780284	2.361297e-04
3	1	0.399844	0.451265	0.457733	0.583541	0.583541	0.998700	0.796967	0.808966	0.303350	0.781241	1.078888e-04
4	1	0.465022	0.538432	0.522298	0.598783	0.598783	0.998973	0.797366	0.809304	0.303475	0.781550	7.890000e+09

5 rows × 96 columns

- ➢ Correlation (before training the model)

```
 import matplotlib.pyplot as plt

import seaborn as sns

sns.set()

sns.heatmap(data.corr())

plt.show()


X = data.drop(["Bankrupt?"], axis="columns")

y = data["Bankrupt?"]
```



- ➢ Split the dataset and use the logistic regression model to train the bankruptcy model:

```
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
logreg = LogisticRegression()
logreg.fit(x_train, y_train)
```

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
```

- ➢ Check the accuracy of the score on the training set

```
logreg.score(x_test, y_test)
```

0.9596774193548387 (Accuracy= 95%)