Capstone Project-3

Project Title

COMPANY BANKRUPTCY PREDICTION

By-Bhushan Patil


$
BANKRUPTCY

Problem Statement

Prediction of bankruptcy is a phenomenon of increasing interest to firms who stand to loose money because on unpaid debts. Since computers can store huge dataset pertaining to bankruptcy making accurate predictions from them before hand is becoming important.

The data were collected from the Taiwan Economic Journal for the years 1999 to 2009. Company bankruptcy was defined based on the business regulations of the Taiwan Stock Exchange.

Data Summary

There are 3 categorical variables and 93 numerical variables

Categorical variables

Bankrupt?

1 : Yes , 0 : No

Liability-Assets Flag

1 : if Total Liability exceeds Total Assets, 0 : otherwise

Net Income Flag

1 : if Net Income is Negative for the last two years, 0 : otherwise

Dataset

- There are 6819 entries and 96 columns
- 93 out of 96 are numeric (float)
- 3 out of 96 are categorical

  Bankrupt?
  Liability-Assets Flag

  Net Income Flag

```
<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)

Data Description

Y = Output feature
Y - Bankrupt?

X = Input features
X - 95 features

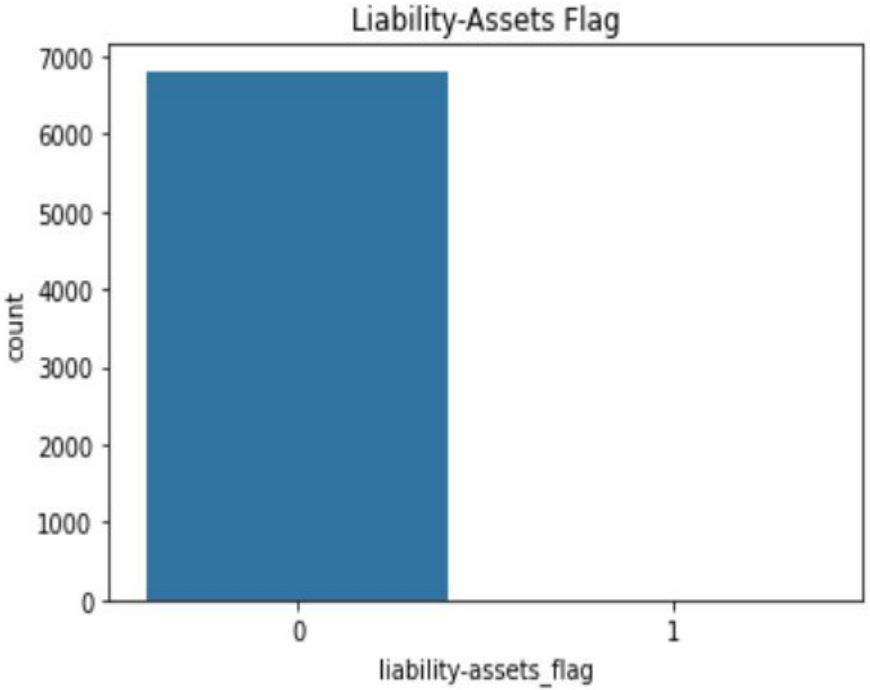

Exploratory Data Analysis

Value Count of Liability-Assets Flag

0 : otherwise

1 : if Total Liability exceeds Total Assets

```
0    6811
1       8
Name: liability-assets_flag, dtype: int64
```

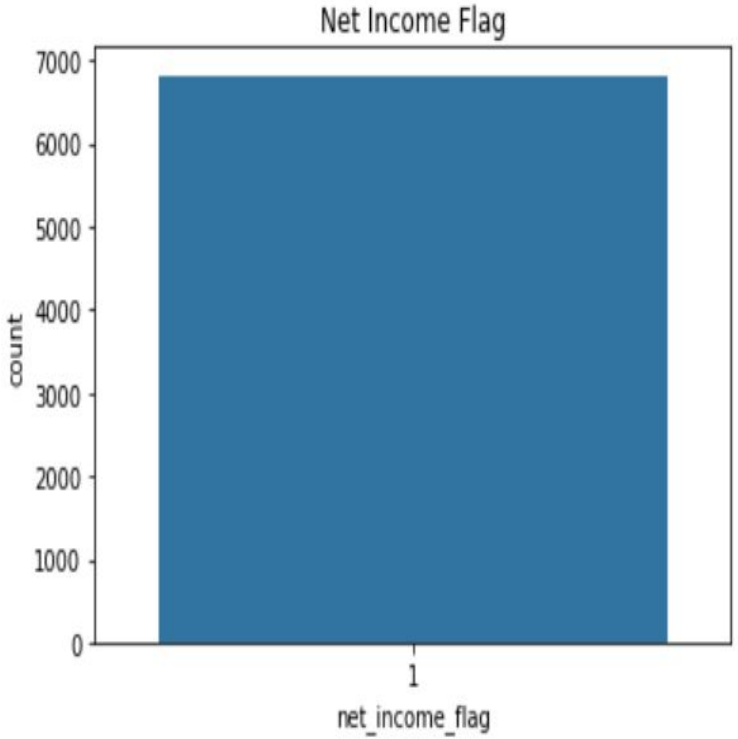


Value Count of Net Income Flag

1 : if Net Income is Negative for the last two years

0 : otherwise

```
1    6819
Name: net_income_flag, dtype: int64
```

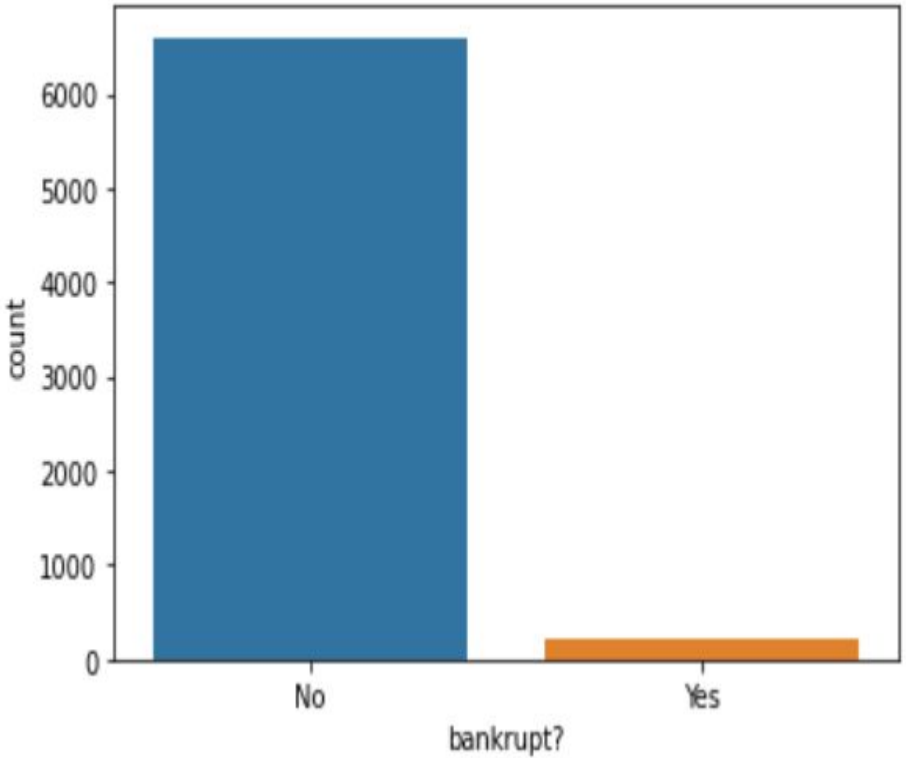


Value Count of Target variable

0 : No

1 : Yes

```
0    6599
1     220
Name: bankrupt?, dtype: int64
```



The Metric Trap

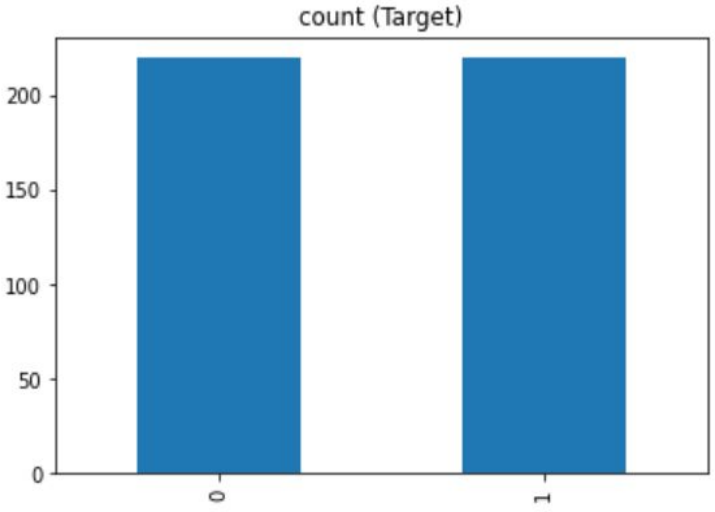
```
Distribution of classes of dependent variable in train :
bankrupt?
0            5274
1             181
dtype: int64

 Distribution of classes of dependent variable in test :
bankrupt?
0            1325
1              39
dtype: int64
```

By using XGB Classifier we can get 97% accuracy.
We are getting very high accuracy because it is predicting mostly the majority class that is 0 (Non-fraudulent).

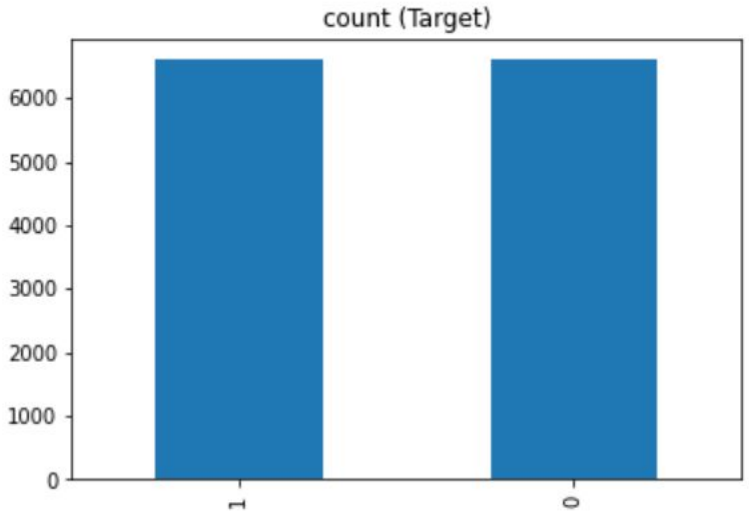
Resampling Techniques

1. Random Under-Sampling
2. Random Over-Sampling
3. Random under-sampling with imblearn
4. Random over-sampling with imblearn
5. Under-sampling: Tomek links
6. Synthetic Minority Oversampling Technique (SMOTE)
7. Near Miss
8. Penalize Algorithms (Cost-Sensitive Training): Support vector classifier

Random Under-Sampling and Over-Sampling



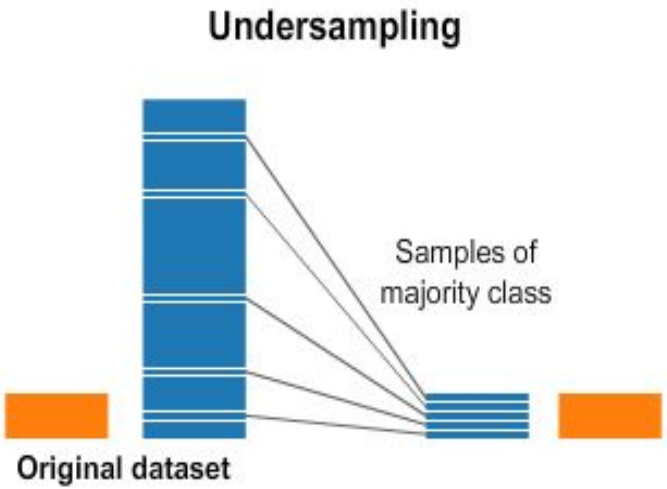
Random Under-Sampling

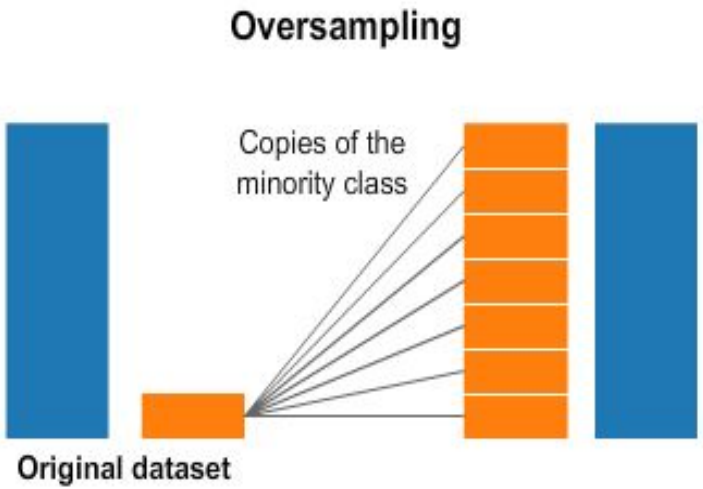


Random Over-Sampling

Imbalance problem

Apply both Under-Sampling and Over-Sampling techniques.





Feature Selection in Machine Learning

Recursive feature elimination (RFE) is a feature selection process that suits a model and eliminates the weakest feature (or features) before the required number of features is achieved. Features are rated by the model's coef or feature importances attributes, and RFE aims to remove dependencies and collinearity by recursively deleting a small number of features per loop.

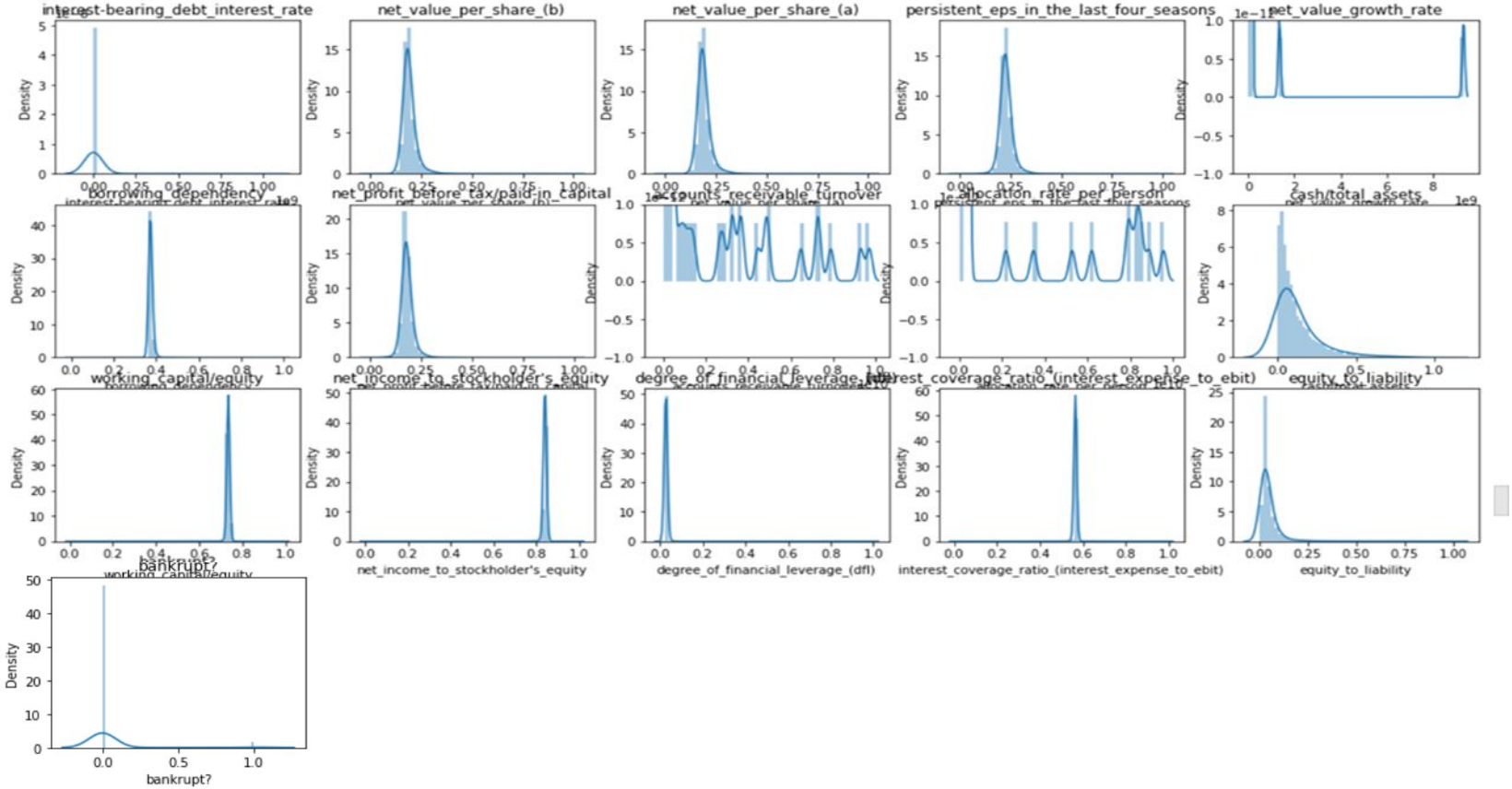
Feature Selection

```
Test score: 0.968  number of features: 15
```

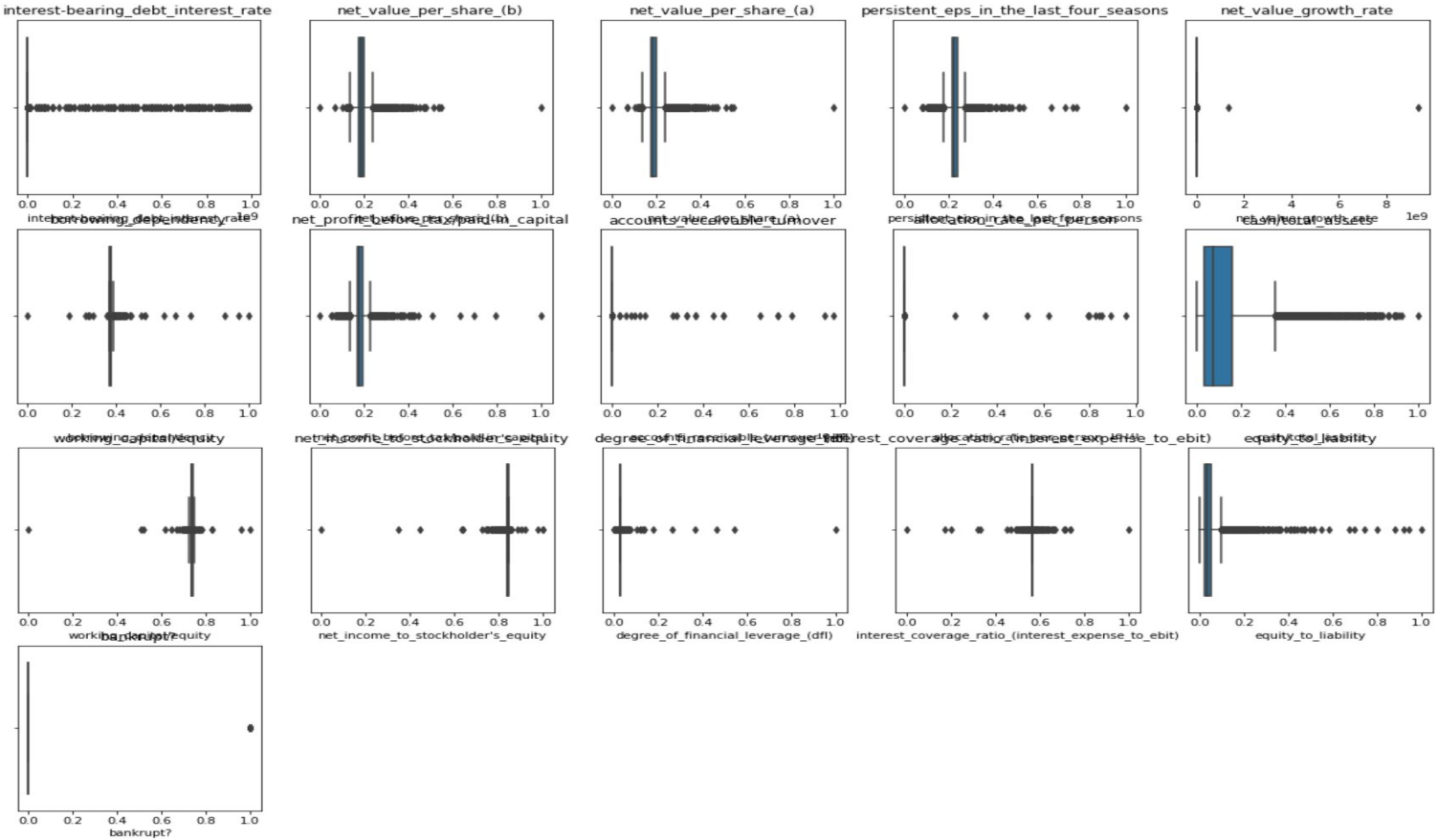
```
RangeIndex: 6819 entries, 0 to 6818
Data columns (total 16 columns):
 #   Column                                            Non-Null Count  Dtype
---  ------                                            --------------  -----
 0   interest-bearing_debt_interest_rate               6819 non-null   float64
 1   net_value_per_share_(b)                           6819 non-null   float64
 2   net_value_per_share_(a)                           6819 non-null   float64
 3   persistent_eps_in_the_last_four_seasons           6819 non-null   float64
 4   net_value_growth_rate                             6819 non-null   float64
 5   borrowing_dependency                              6819 non-null   float64
 6   net_profit_before_tax/paid-in_capital             6819 non-null   float64
 7   accounts_receivable_turnover                      6819 non-null   float64
 8   allocation_rate_per_person                        6819 non-null   float64
 9   cash/total_assets                                 6819 non-null   float64
 10  working_capital/equity                            6819 non-null   float64
 11  net_income_to_stockholder's_equity                6819 non-null   float64
 12  degree_of_financial_leverage_(dfl)                6819 non-null   float64
 13  interest_coverage_ratio_(interest_expense_to_ebit)  6819 non-null   float64
 14  equity_to_liability                               6819 non-null   float64
 15  bankrupt?                                         6819 non-null   int64
dtypes: float64(15), int64(1)
```

Step 1: Univariate Analysis

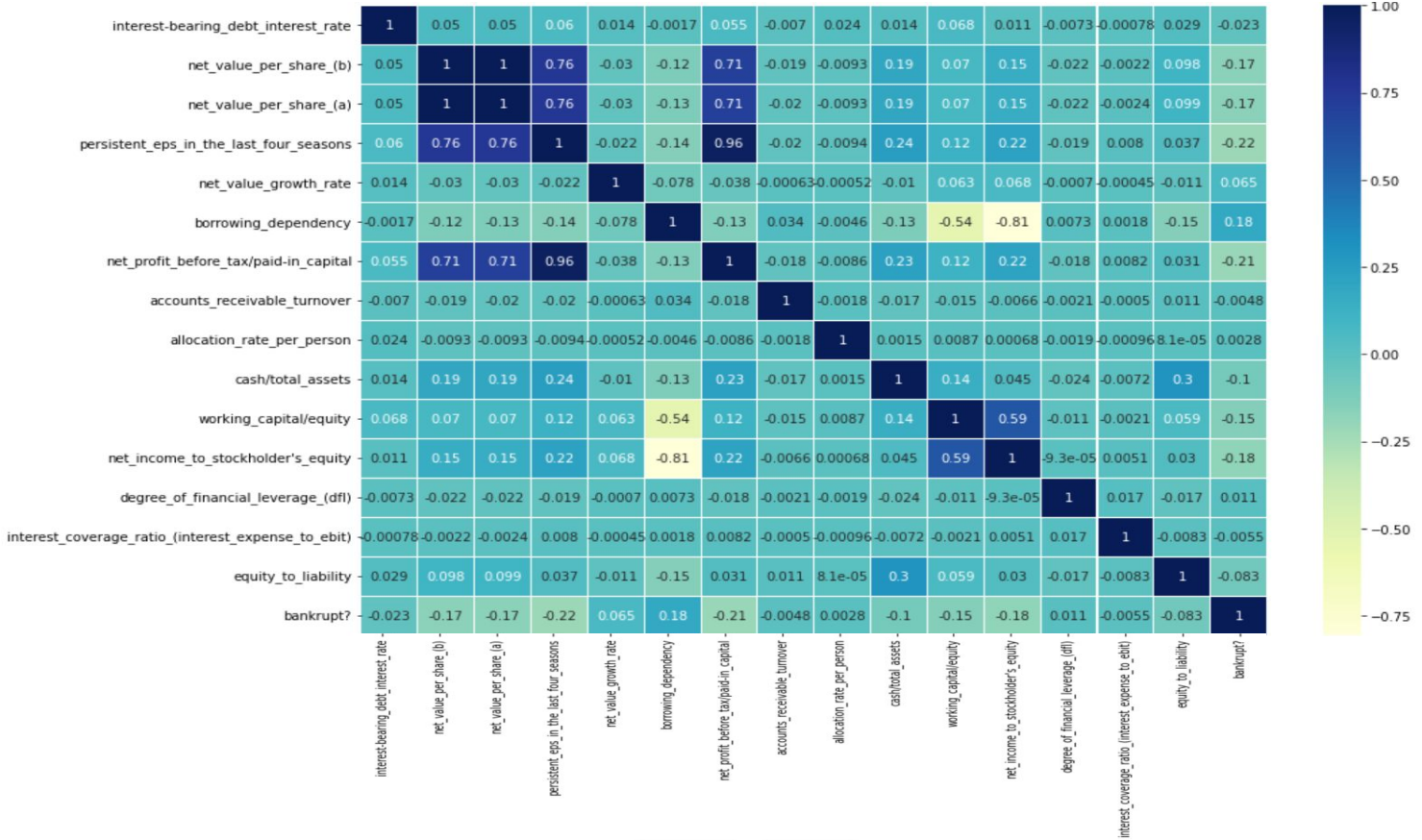
Explore data



Using Boxplot



Step 2: Bivariate/Multivariate Analysis



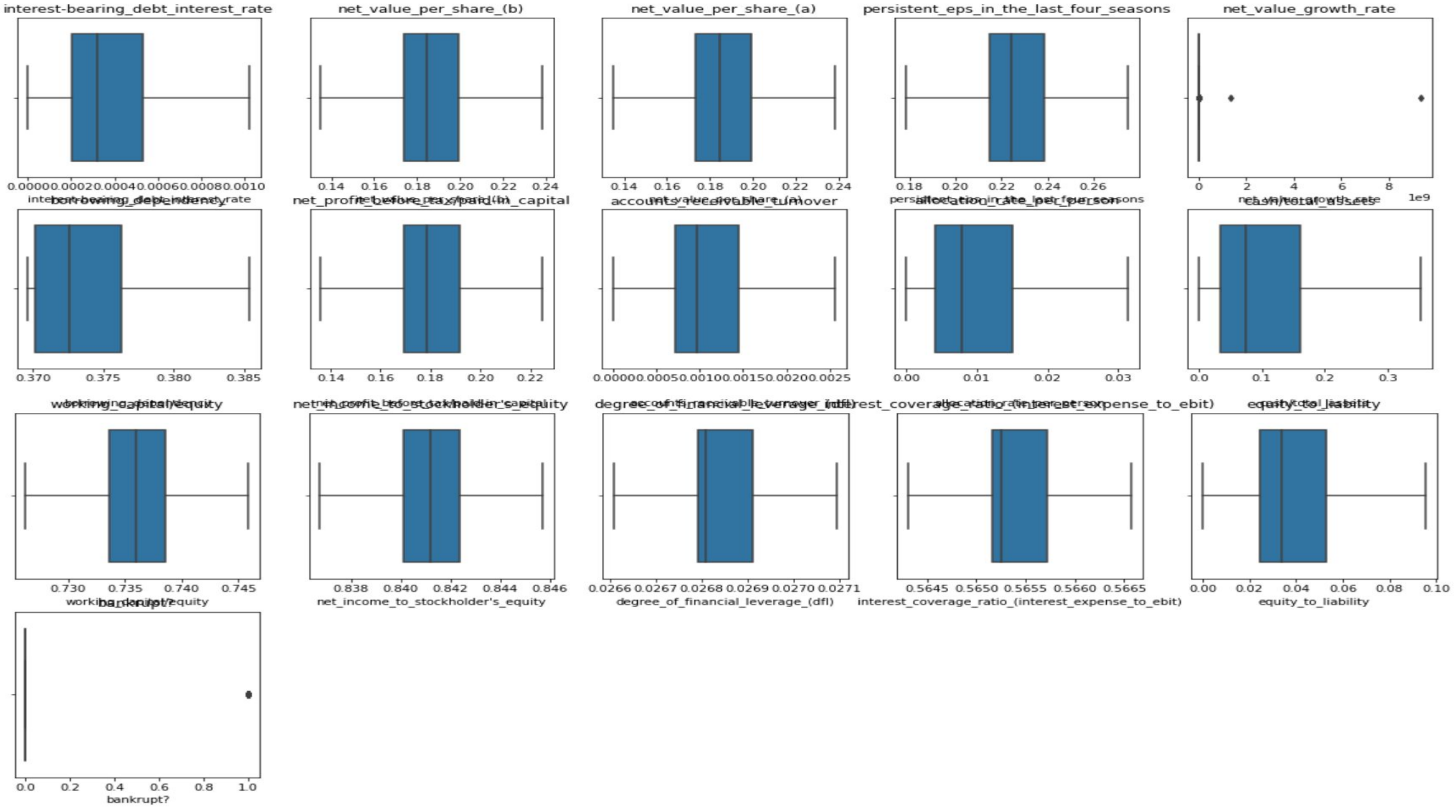
Handling Outliers

Using Outlier Insensitive Algorithms

Some algorithms that are not sensitive to outliers are Naive Bayes Classifier, Support Vector Machine, Decision Tree, Ensemble Techniques, and K-Nearest Neighbours. We can use these algorithms to get rid of outliers.

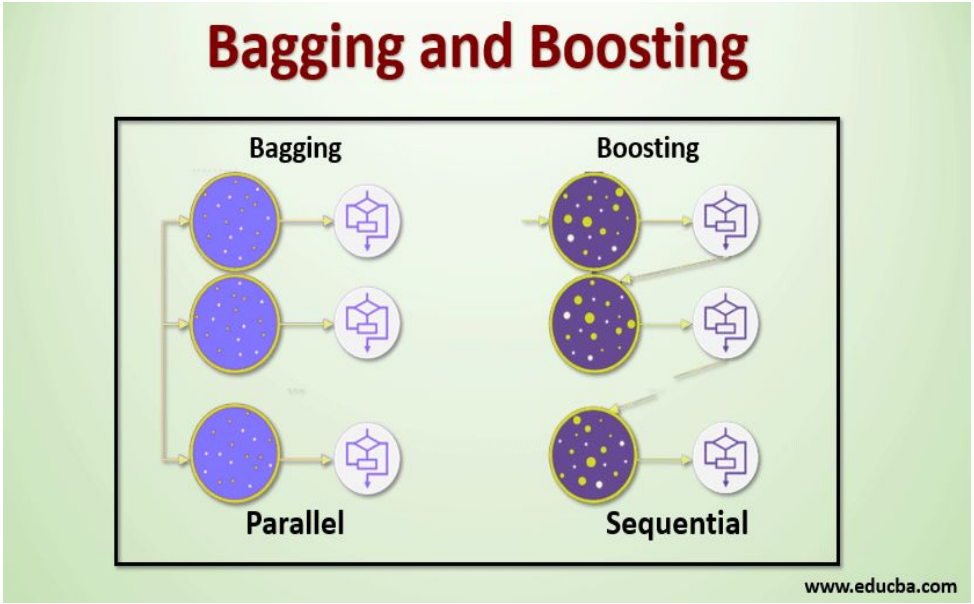
Caps outliers to closest existing value within threshold (IQR)

Verification of Deleting Outliers



Model Used

- Random Forest Classifier
- Gradient Boosting Classifier
- XGB Classifier
- Cat Boost Classifier
- LGBM Classifier
- SVM Classifier
- Decision Tree Classifier
- Logistic Regression



Model Testing Report On Overall Data

	support_vector_classifier	random_forest	XGBOOST	cat_boost_classifier	LGBMClassifier	Decision_tree_classifier
test_acc	0.805718	0.923754	0.978739	0.975806	0.978006	0.970674
train_acc	0.799633	0.917507	1.000000	0.999083	1.000000	0.970852
test_precision	0.055118	0.255639	0.692308	0.750000	0.846154	0.482759
train_precision	0.060694	0.278418	1.000000	1.000000	1.000000	0.585938
test_recall	0.358974	0.871795	0.461538	0.230769	0.282051	0.358974
train_recall	0.348066	0.933702	1.000000	0.972376	1.000000	0.414365
test_f1-score	0.095563	0.395349	0.553846	0.352941	0.423077	0.411765
train_f1-score	0.103363	0.428934	1.000000	0.985994	1.000000	0.485437
test_auc	0.586541	0.963580	0.947615	0.958897	0.960252	0.936701
train_auc	0.639762	0.973236	1.000000	0.999998	1.000000	0.938089

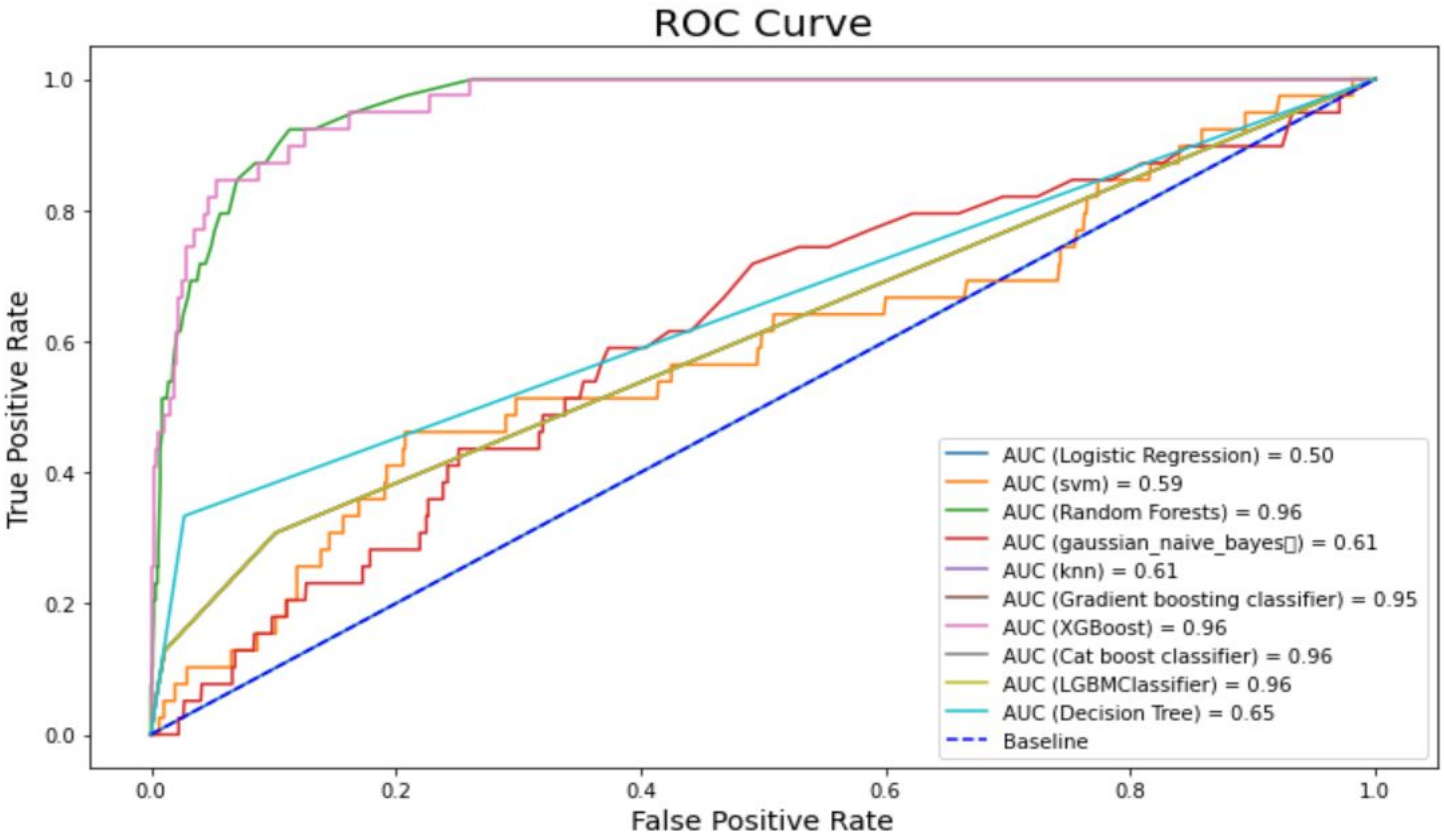
Under-Sampling report

	random_forest_classifier	gradient_boosting_classifier	XGBClassifier	cat_boost_classifier	LGBMClassifier
test_acc	0.755865	0.641496	0.720674	0.730205	0.714809
train_acc	0.941989	1.000000	1.000000	1.000000	1.000000
test_precision	0.100543	0.072243	0.090909	0.093827	0.091121
train_precision	0.944444	1.000000	1.000000	1.000000	1.000000
test_recall	0.948718	0.974359	0.974359	0.974359	1.000000
train_recall	0.939227	1.000000	1.000000	1.000000	1.000000
test_f1-score	0.181818	0.134513	0.166302	0.171171	0.167024
train_f1-score	0.941828	1.000000	1.000000	1.000000	1.000000
test_auc	0.951524	0.939468	0.959632	0.954891	0.955123
train_auc	0.992674	1.000000	1.000000	1.000000	1.000000

Over-Sampling report

	SVMClassifier	gradient_boosting_classifier	XGBClassifier	cat_boost_classifier	LGBMClassifier
test_acc	0.785924	0.943548	0.942815	0.977273	0.978006
train_acc	0.621350	0.979048	0.978385	1.000000	1.000000
test_precision	0.059233	0.302083	0.290323	0.633333	0.666667
train_precision	0.677580	0.959782	0.958561	1.000000	1.000000
test_recall	0.435897	0.743590	0.692308	0.487179	0.461538
train_recall	0.463026	1.000000	1.000000	1.000000	1.000000
test_f1-score	0.104294	0.429630	0.409091	0.550725	0.545455
train_f1-score	0.550124	0.979478	0.978842	1.000000	1.000000
test_auc	0.520368	0.952356	0.956284	0.953033	0.961974
train_auc	0.584433	0.997387	0.998103	1.000000	1.000000

Using ROC and AUC



Model used after feature selection

```
XGBClassifier(eval_metric='logloss')
Score for XGBClassifier(eval_metric='logloss') method: 0.9384848484848485
Score for XGBClassifier(eval_metric='logloss') method using cross_val_score: 0.9357440941262987
              precision    recall  f1-score   support

           0       0.96      0.91      0.94      1612
           1       0.92      0.97      0.94      1688

    accuracy                           0.94      3300
   macro avg       0.94      0.94      0.94      3300
weighted avg       0.94      0.94      0.94      3300

0.937830753942587
```

Roc auc score

```
RandomForestClassifier()
Score for RandomForestClassifier() method: 0.9724242424242424
Score for RandomForestClassifier() method using cross_val_score: 0.9712050739957718
              precision    recall  f1-score   support

           0       0.99      0.96      0.97      1612
           1       0.96      0.99      0.97      1688

    accuracy                           0.97      3300
   macro avg       0.97      0.97      0.97      3300
weighted avg       0.97      0.97      0.97      3300

0.9720953923770771
```
Roc auc score
```
LogisticRegression()
Score for LogisticRegression() method: 0.8939393939393939
Score for LogisticRegression() method using cross_val_score: 0.8875530839231546
              precision    recall  f1-score   support

           0       0.90      0.88      0.89      1612
           1       0.89      0.91      0.90      1688

    accuracy                           0.89      3300
   macro avg       0.89      0.89      0.89      3300
weighted avg       0.89      0.89      0.89      3300

0.8935619112579822
```
Roc auc score

CONCLUSION

1. Random forest classifier, XGB Classifier, LGBM Classifier give best results.
2. Random forests, Cat boost classifier, LGBM classifier, XGBoost give best roc-auc value above 95%.
3. Oversampling method perform better than undersampling method.
4. Model testing report on overall data give best results because we use non sensitive algorithms.
5. Random forest classifier model used after feature selection by using Recursive feature elimination (RFE) process give best results.


THANK YOU