Smith SCHOOL OF BUSINESS Queen's University

Global Master of Management Analytics

GMMA 823
Analytics for Financial Management

Professor Matthew Thompson

Assignment #2 - Corporate Bankruptcy Models
Sunday, January 10, 2021

Team New York

Order of files:

Filename	Pages	Comments and/or Instructions

Additional Comments:

Jupyter Notebook is available upon request.

Overview and Summary

Team New York has developed an XGBoost model to predict corporate bankruptcy. The XGBoost model had a final AUC of 0.9454 after hyperparameter tuning on the test data set. Specifically, the model predicted non-bankruptcies with 96% accuracy and actual bankruptcies with 72% accuracy on test data. The top predicting factor that XGBoost extracted was Asset Turnover - a measure of how efficient a company is in using their assets to generate revenue or sales, followed by Leverage Ratio and Earnings per Share (EPS). Banks can use this type of machine learning model in the banking industry to mitigate investment and lending risk, both in corporate lending and personal/corporate investments (stocks), as an example.

About the Dataset

The bankruptcy dataset contains 15 variables, precisely one categorical variable: Data Year - Fiscal, one binary target Label: BK (what we need to predict – bankruptcy or no bankruptcy), and 13 continuous variables: Tobin's Q, EPS, Liquidity, Profitability, Productivity, Asset Turnover, Operational Margin, Return on Equity, Market Book Ratio, Assets Growth, Sales Growth, and Employee Growth. There are a total of 92,872 records in the dataset, from the fiscal year 1979 to 2017.

Methodology

Team New York took the following steps to understand and prepare our dataset for predictive modelling:

1. Exploratory Data Analysis

a. Variable Distribution

From our EDA, we noticed that most variables are right-skewed (see Figure 1).

b. Events

When looking at bankruptcies (BK = 1), we noticed several peaks when exploring the "Data Year - Fiscal" variable (See Figure 2). These peaks occur in the following years 1989-1990, 1996-1997, 2007-2008.

We can attribute the late 1980's-1990's bankruptcies to the savings and loans crisis, the 1990's to early 2000's to the dot-com bubble, 1996-1997 to the Asian Financial Crisis, and 2007-2008 to the Global Financial Crisis. Additionally, several approximate single year events occurred during these periods, such as the 1987 stock market crash and the 1989 junk bond crash.

c. Correlation

Looking at the Pearson correlation plot (see Figure 3), we notice the following correlations:

- Liquidity and Return on Equity are moderately correlated at a level of 0.58
- Productivity and Profitability are moderately correlated at a level of 0.45
- Liquidity and Profitability are moderately correlated at a level of 0.47
- Our created fin_crisis variable and the original Data Year - Fiscal variable are correlated at a level of -0.62

However, we conclude that none of the correlations are strong enough to remove variables from the data set. This was tested in the modeling phase where models performed weaker when we took out variables.

d. Missing Observations

Overall, there was about 2% in missing observations in the data set (Figure 4). The highest missing values came from the following variables: "Employee_Growth" (7.5% missing values), "Sales_Growth" (7.22%), "Assets_Growth" (7.22%), "Operational_Margin" (5.98%). The following variables had missing values but were under 1%: "Tobin's Q," "Liquidity," "Profitability," "Asset_Turnover."

e. Class Imbalance

Finally, we noticed a large class imbalance from the target bankruptcy "BK" variable (See Figure 5). Class imbalance is expected from bankruptcy datasets as defaults from publicly traded corporations usually don't make up a large proportion of businesses.

2. Data Preprocessing, Initial Modeling, and Hyperparameter Tuning

Step-by-Step Process

- Step 1: Import necessary data preparation packages such as SimpleImputer, StandardScaler, SMOTE, train_test_split, cross_validate from the sklearn library. Import the classification models we want to use such as Neural Network, Naïve Bayes, Support Vector Machine, XGBoost, Light Gradient Boosting Machine, and Random Forest

- Step 2: Plug in the missing values for each record with missing values using KNN imputation. Initially, we had imputed the median of the variable but found improved results using KNN.

- Step 3: Define the following years as years where there was a financial crisis (according to Reuters), call it "fin_crisis" From our EDA, we believe that the year could have a potential impact on predicting bankruptcies.
    - 1981, 1986, 1988, 1990, 1991, 1992, 1993, 1994, 1995, 1996, 1998, 1999, 2000, 2001 = 1
    - 1987, 1989, 1997 = 2 (separate financial crisis + another crisis at the same time)

- Step 4: Drop the "Data Year - Fiscal" feature as the high correlation between fiscal year and the target variable is a relatively highly correlated variable without direct causation.

We instead created a feature based on the financial crises to capture the underlying macroeconomic effects in our model.

- Step 5: Scale our data using the Standard Scaler function to ensure our variable values are on the same scale

- Step 6: Split dataset into 70% train and 30% test

- Step 7: Oversample our minority class variable (BK = 1) using SMOTE for our training dataset

- Step 8: Run our trained model on the original test data set on the selected models using 5-fold cross-validation

- Step 9: Analyze the AUC using a ROC plot as a basis for determining which model to perform hyperparameter tuning in the next steps

- Step 10: Analyze the Classification Report, Confusion Matrix, and Feature Importance plot to understand how the model performed at the individual class level and what features were essential to include in the model

- Step 11: Select the highest performing model to hyperparameter tune and run it on the test data set

- Step 12: Package and finalize model for deployment

3. Initial Baseline Models and Observations

We ran a total of 6 baseline models, XGBoost, Light Gradient Boosting Machine, Random Forest, Support Vector Machine, Neural Network, and Naïve Bayes, using 5-fold cross-validation. Cross-validation allows us to ensure our models are not overfitted and maintain robust accuracy when going from our training data to testing data.

Summary AUC Scores	
XGBoost	0.998
LightGBM	0.995
RandomForestClassifier	1.000
SVC	0.838
NeuralNetwork	0.966
NaiveBayes	0.605

In the initial model performance on the training data, we observed a perfect score of 1.00 for our Random Forest Classifier. RFC may have overfitted our data. Our next highest-performing

model is XGBoost, with an AUC of 0.998. We selected XGBoost to run on our testing data, which generated the following results:

- AUC: 0.9404
- Accuracy: 0.9735

We also decided to use AUC as an evaluation metric instead of Accuracy due to the imbalanced data. AUC gives us a more balanced metric when evaluating model performance.

Classification Report - Base XGBoost Model

Class	Precision	Recall	F1 Score	Support
0	1.00	0.98	0.99	27,697
1	0.12	0.56	0.20	165
Accuracy			0.97	27,862
Macro Avg.	0.56	0.77	0.59	27,862
Weighted Avg.	0.99	0.97	0.98	27,862

Confusion Matrix - Base XGBoost Model

True Label	**0**	27,033	644
	1	73	92
		0	**1**
		Predicted Label	

From the initial results, it appears that the XGBoost model is very accurate at predicting non-bankruptcies (BK = 0) with a recall score of 0.98. Our model wasn't too great at predicting bankruptcies (BK = 1), which is our variable of interest. It had a recall score of 0.56. As a reminder, we've put the definition of recall below.

- *Recall = TP / (TP + FN): What proportion of actual positives was identified correctly? (i.e., of our model's predicted bankruptcies, how many of them were actually correct?)*

In this bankruptcy prediction context, it's crucial that recall is as high as possible to predict the highest number of bankruptcies to minimize risk and exposure, whether in credit or portfolio investments.

Looking at the feature importance plot (See Figure 6), the XGBoost model ranked Asset Turnover as the most important feature, followed by Leverage Ratio, EPS, Return on Equity, Operational Margin, Profitability, and Productivity, in that order.

When we look at Asset Turnover's definition, it measures the efficiency of how a company is using its assets to generate revenue or sales. Higher Asset Turnover ratios imply that a company is efficient. A lower ratio suggests that the assets are not being used efficiently and may have internal problems. Looking at Figure 7, we noticed that all bankrupt companies are on the lower

end of the box plot implying internal inefficiencies and inability to generate revenue or sales from their assets vs. companies that weren't bankrupted had higher Asset Turnover ratios.

From a technical perspective, we noticed that removing variables based on moderate correlation and feature importance ranking always resulted in a weaker performing model. Thus, we decided to keep all variables in the data set.

We also noticed that our tree-based models (XGBoost, LightGBM, Random Forest) performed well, unlike gaussian-based models (SVC and Naïve Bayes), which resulted in poorly identifying bankruptcies. This is also consistent with the literature by Barboza, Kimura, Altman (2015) - "Machine learning models and bankruptcy prediction."

Finally, we noticed that the SMOTE method had various improvements depending on the type of model. For instance, SMOTE improved AUC scores for the lower performing models (gaussian-based models) by 10-15%, whereas it only enhanced our top models (tree-based models) by ~1%.

4. Final Tuned Model and Observations

Our final tuned XGBoost model predicted 26 more actual bankruptcies (0.72 Recall, an increase of 0.16 in accuracy) than our base model, but it misclassified false positives by an extra 558. Again, in this context, the true classifications outweigh the increase in misclassifications since we want to catch more actual bankruptcies, even if it's at the expense of misclassifying false positives. The trade-off in misclassification could be warranted since, overall, it would influence a more conservative risk profile for the bank in lending or investments for these flagged companies. Below are the AUC and accuracy metrics to compare against the baseline model:

- AUC: 0.9454
- Accuracy: 0.9551

Classification Report - Final XGBoost Model

Class	Precision	Recall	F1 Score	Support
0	1.00	0.96	0.98	27,697
1	0.09	0.72	0.16	165
Accuracy			0.96	27,862
Macro Avg.	0.54	0.84	0.57	27,862
Weighted Avg.	0.99	0.97	0.97	27,862

Confusion Matrix - Final XGBoost Model

True Label	**0**	26,495	1202
	1	47	118
		0	**1**
		Predicted Label	

From a hyperparameter standpoint (See Figure 8 and 9), the following changed from the base model to tuned model:

Hyperparameter	Base	Tuned
colsample_bytree	1	0.5
learning_rate	0.30	0.1
max_depth	6	7
min_child_weight	1	5
subsample	1	0.7

Business Interpretation and Application

Our XGBoost model can be deployed in many environments, such as corporate and investment banking. Both investment and corporate banks can use this type of machine learning model to assess a company's risk before lending or to dictate the lending terms. Banks could use this type of bankruptcy prediction model to evaluate corporate clients applying for a business loan or line of credit. The bank could feed the applicant's current financial ratios into the model as an additional risk mitigation step. A company with a higher risk of failing may either not get the loan they are looking for or on really aggressive terms that favour the bank.

The model also serves as a diagnostic tool to understand which measures are critical to predicting bankruptcy. In our case, Asset Turnover Ratio was the top factor, meaning if a bank wanted to structure a deal on their terms, they would have to negotiate with management on better utilization of their assets in revenue generation.

From an investment standpoint, portfolio managers (PM) can also be alerted to the potential of bankruptcy and, therefore, would shift their asset allocation. This model predicts defaults for companies within two years; this gives the PM an excellent early warning sign to unload any company assets. The PM would need an understanding that this model is more conservative, given our discussion above about flagging more true bankruptcies, at the expense of flagging more false positives.

From an investment bank standpoint, these same flags could trigger a process for the investment bank's management to start preparing deals for the companies on the "to-be-bankrupted" list. This would give investment banks an early start to buying what will be distressed debt for pennies on the dollar.

Other uses for our model could be in the regulatory or compliance control functions. Internal or external regulators could use the model as an assessment tool to determine if a corporation is being too reckless. Especially with the indicator flag for financial crises, regulators could intervene if they feel a business is acting irresponsibly. For example, our model indicated that Asset Turnover and Leverage Ratio were the top two features for predicting bankruptcy. Using

this information, they could warn/fine companies if they do not de-leverage depending on the economic environment.

Appendix

Figure 1 - Variable Distribution

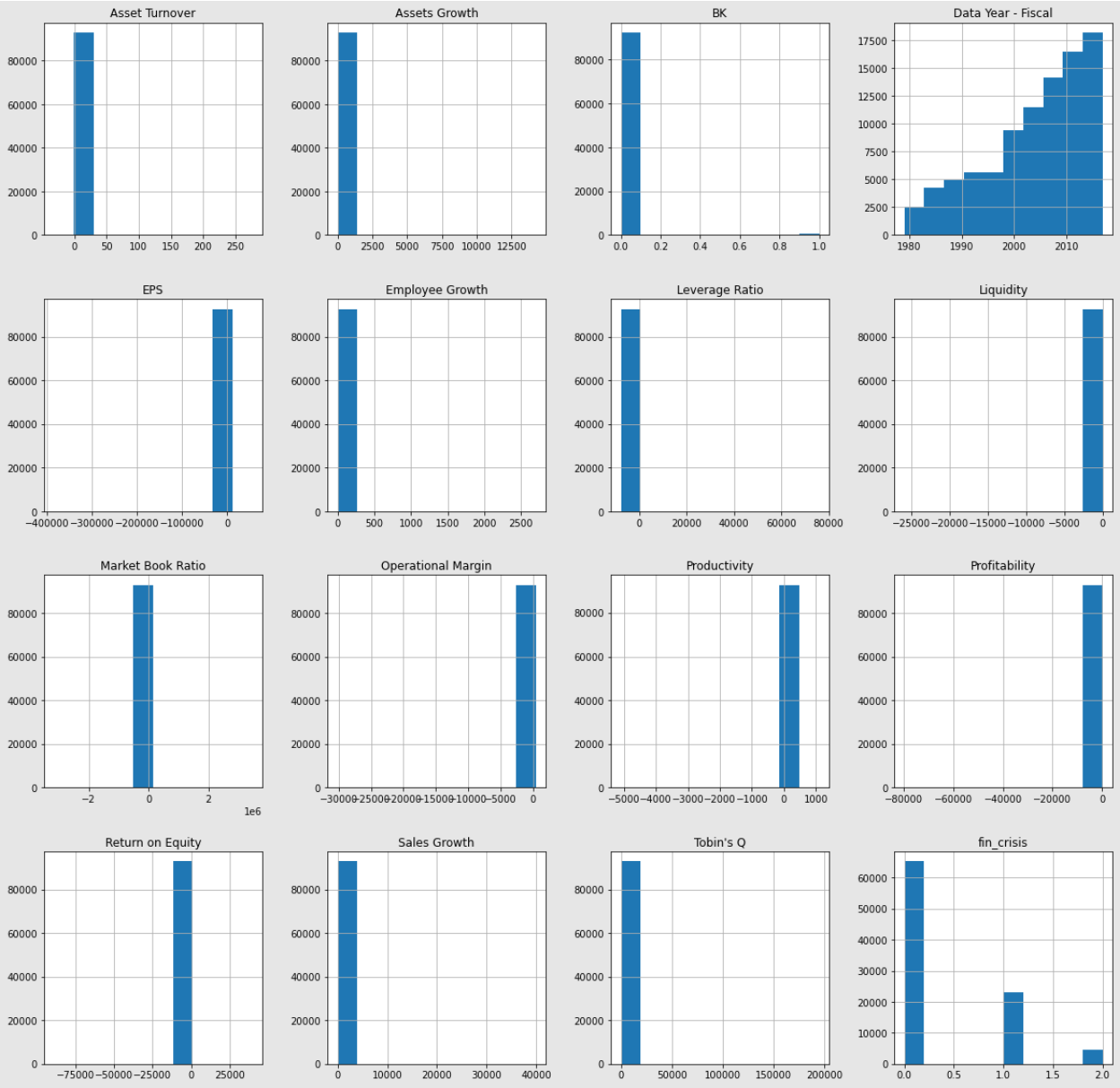

Figure 2 - Number of Bankruptcies per Year

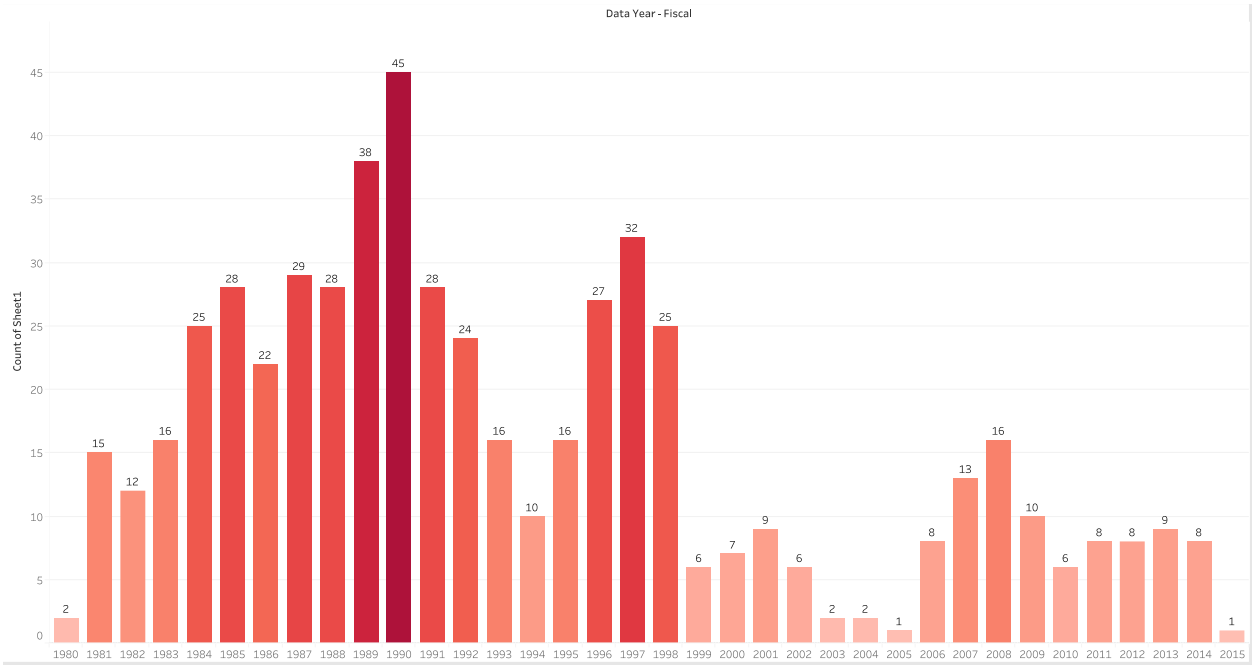

Figure 3 - Pearson Correlation Plot for Variables

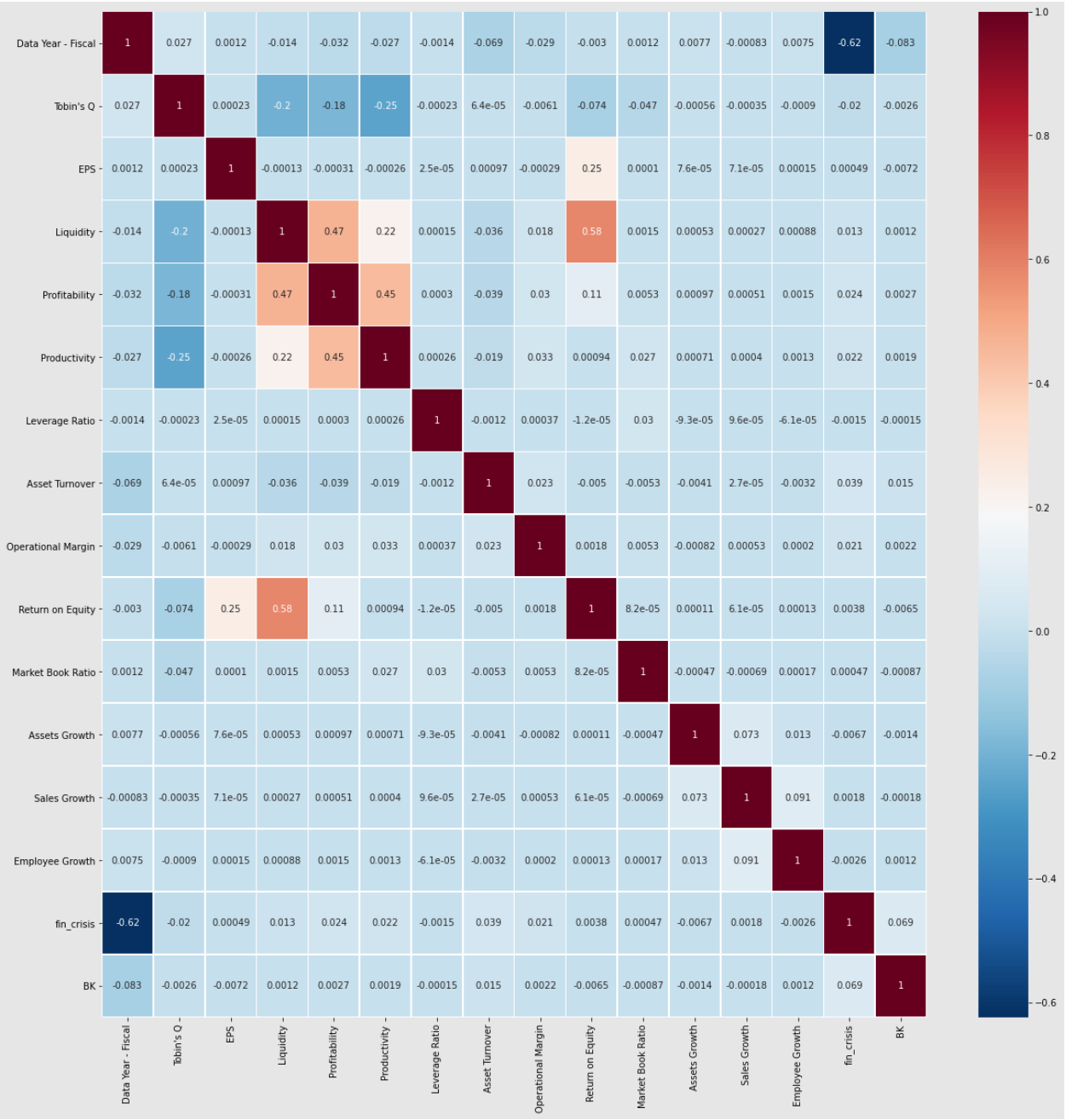

Figure 4 - Missing Values

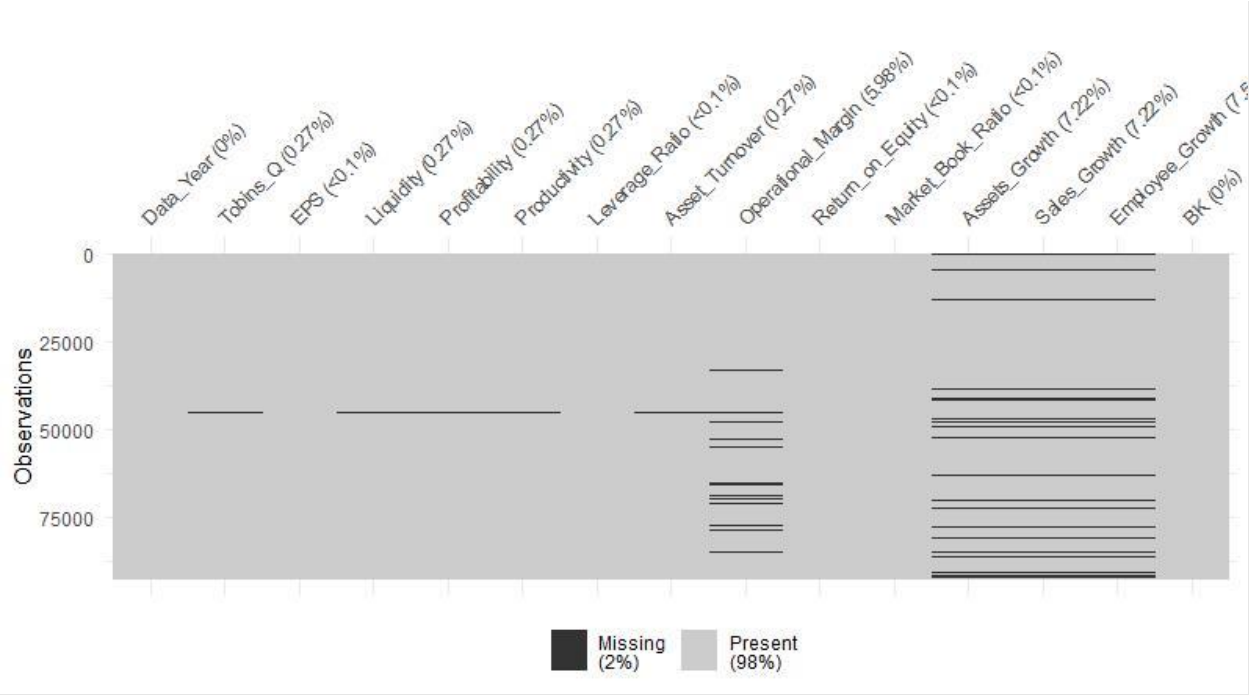

Figure 5 - Class Imbalance

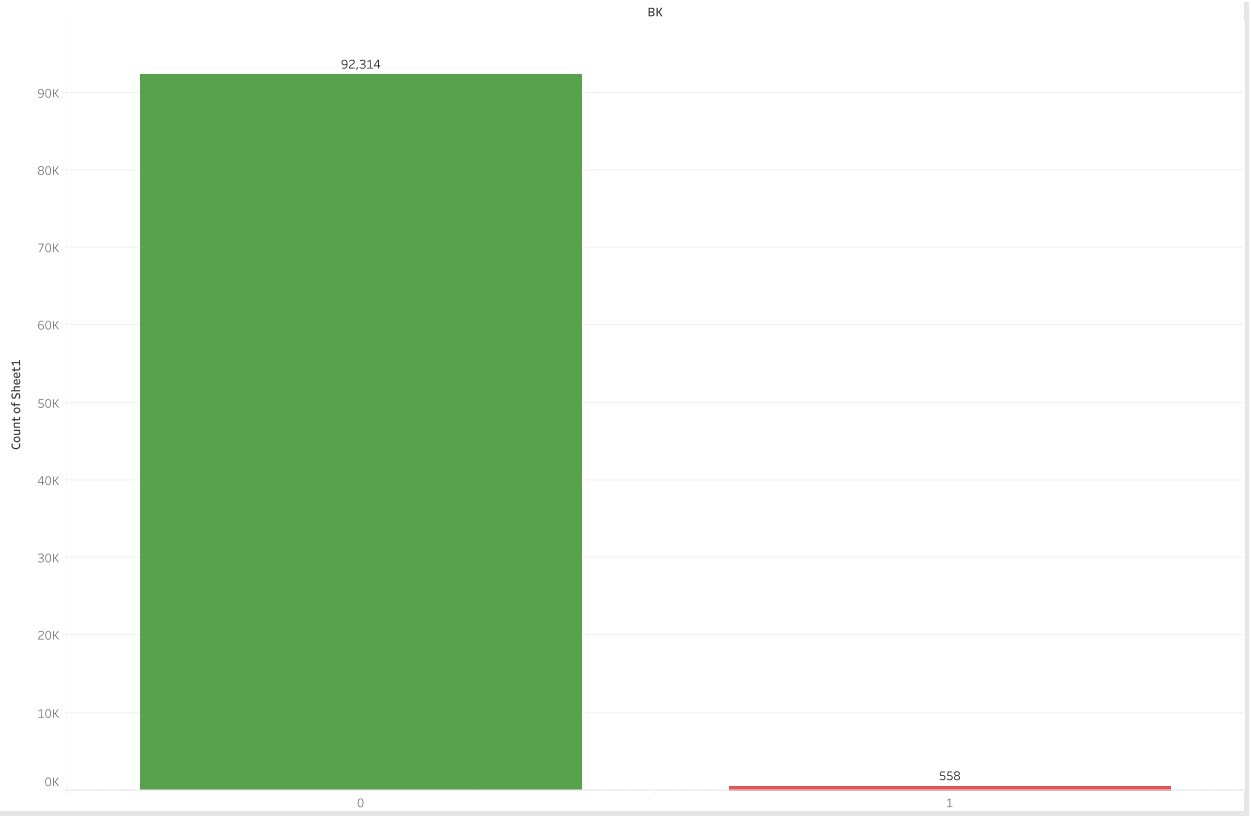

Figure 6 - Feature Importance Plot

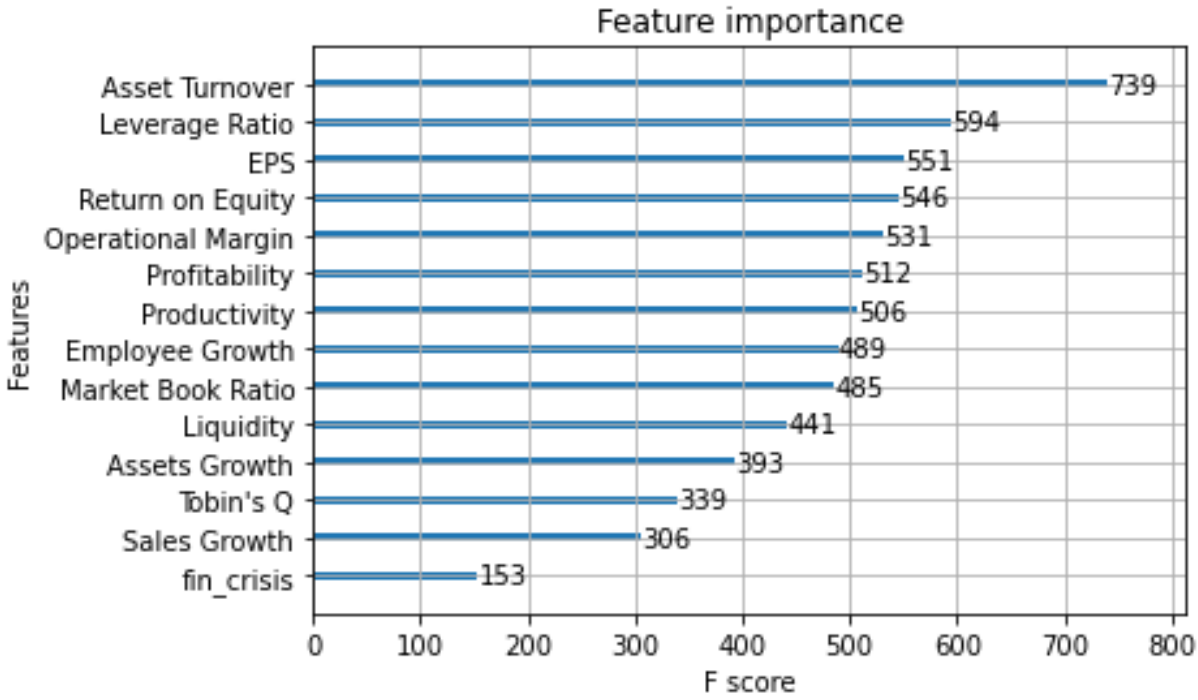

Figure 7 - Asset Turnover Boxplot

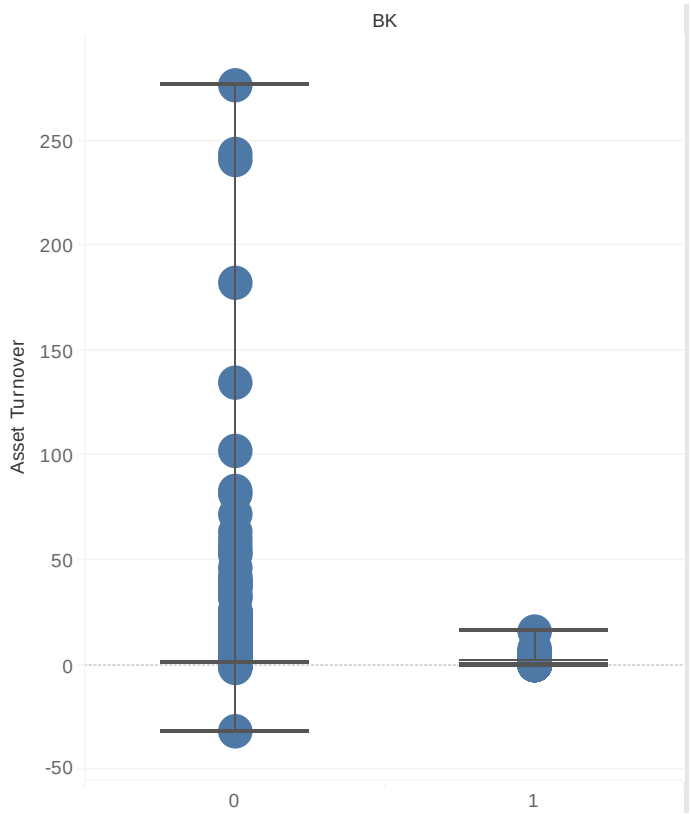

Figure 8 - Baseline XGBoost Model Hyperparameters

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
              colsample_bynode=1, colsample_bytree=1, gamma=0, gpu_id=-1,
              importance_type='gain', interaction_constraints='',
              learning_rate=0.300000012, max_delta_step=0, max_depth=6,
              min_child_weight=1, missing=nan, monotone_constraints='()',
              n_estimators=100, n_jobs=0, num_parallel_tree=1, random_state=0,
              reg_alpha=0, reg_lambda=1, scale_pos_weight=1, subsample=1,
              tree_method='exact', validate_parameters=1, verbosity=None)
```

Figure 9 - Tuned XGBoost Model Hyperparameters

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
              colsample_bynode=1, colsample_bytree=0.5, gamma=0, gpu_id=-1,
              importance_type='gain', interaction_constraints='',
              learning_rate=0.1, max_delta_step=0, max_depth=7,
              min_child_weight=5, missing=nan, monotone_constraints='()',
              n_estimators=100, n_jobs=0, num_parallel_tree=1, random_state=0,
              reg_alpha=0, reg_lambda=1, scale_pos_weight=1, subsample=0.7,
              tree_method='exact', validate_parameters=1, verbosity=None)
```