Financial Indicators of U.S Stock Market

May 1, 2020

Provided by Xiaoqi Hu, Yanni Lan, Chenhang Niu, Yuyang Shu, Siqi Zhang
Questrom School of Business, MSBA Capstone Project Summary Report

MSBA Capstone Project Summary Report

Team 1: Xiaoqi Hu, Yanni Lan, Chenhang Niu, Yuyang Shu, Siqi Zhang
Github: https://github.com/CNIU1997/BA_888.git

I. Problem Statement

In the FinTech Industry, algorithmic trading strategies are expanding with new methods. We noticed machine learning is gradually being applied to the finance industry, helping asset management companies to find new investment opportunities and seek for better alpha. On the other hand, Many IPO companies have invested tremendous money on Infrastructure and R&D to raise more capital from the Stock Market.

One great example will be Tesla. After the fourth-quarter earnings reports were released, the stock price rose to 23%, around $960 per share. The quarterly report reveals the profitability of the current operation, and Tesla worked on the transformation from a niche manufacturer into a mass-market electric-vehicle producer (Sergei Klebnikov, 2020). One great lesson learned from this breaking news is, the movement of time series data from financial markets is influenced by a rich mixture of quantitative information from the dynamics of the system and company operations. Company operations, as the data we get from annual financial reports, are related to investor's confidence in stock investment. Regarding our project, We plan to investigate how close the relationship is, and what kind of indicators would be most related to the stock performance. The information extracted from financial reports better at predicting the direction of underlying asset volatility movement, or its second-order statistics, rather than its direction of price movement.

The Annual financial reports usually include detailed information about the company's operation, cash flow, and investment. Some time would be hard for investor's get concise and valuable information among the over 20 pages of documents. Thus, in this project, our goal is trying to look at over 200 Financial Indicators found in 10-K filings among all different industry and then build machine learning model to learn the differences between stocks that perform well and those that don't through the price variation, and then leverage this knowledge to predict which stock will be worth buying. The biggest challenge and our project goal are to feature predictable variables that could contribute to the stock recommendation.

II. Dataset Introduction

The datasets we used are from **Kaggle**[1] which contains around four thousand individual stocks of different companies with more than 200 financial indicators. This topic originally has five-years of data from 2014 to 2018. We further combined it into one consolidated dataset. The dataset includes stocks from eleven industrial sectors, and

[1] *Kaggle Resource: https://www.kaggle.com/cnic92/200-financial-indicators-of-us-stocks-20142018/kernels

the variables are the terms directly from the financial reports. The price variation is a key part of this dataset, which means price increases or decreases compared to the year before and it is the output variable that we are trying to predict. In the dataset, the positive and negative returns are being classified as 1 and 0. We also have revenue, cost, dividend increase rate, and all the probable financial indicators from the 10-K filings trying to find the relationship with the stock price. However, not all the variables are effective work for the models, some redundant financial indicators such as revenue and revenue growth. Our main focus for this project is trying different feature engineer methods to filter out similar variables, reduce the coefficient, and improve the model accuracy.

Data Cleaning

The process of data cleaning involved the following procedures: inspected the original datasets, dealt with NAs and outsiders, and merged datasets. As we noticed, each dataset includes more than three thousand stocks across five years and 226 financial indicators. Before merging the data, we first cleaned up the column names by replacing the space with underscores. To be noted, both lower cases and upper cases exit in the names due to problems caused by similar names after converting cases. We inspect the correlation equal to 1 to remove the variables that have different names but in exactly the same values. Additionally, the target variables, named '201X_PRICE_VAR_[%]', were replaced by the uniformed name 'PRICE_VAR' for the merging purpose. Besides, we replaced all the NAs(omitting NAs would result in an eighty percent loss of the data). In our dataset, we have some extreme outliers, so we looked into outliers and also inspected the distribution of data without outliers but decided to keep outliers for now considering the size of training and testing. Lastly, for model training, we take out year labels and use randomization to split the train and test data set by 80% and 20%.

Distribution by Class

In the final result, we use the class to determine the company's stock performance indicator. Class equal to 0 means stock price decreased compared to previous fiscal year and classified as not worth buying, and class equal to 1 means stock price increased compared to previous fiscal year, therefore, it's worth buying. Firstly, we are going to do a balance check for both the train and test dataset. The 0 and 1 categories are equally distributed into two datasets, and we notice the class1, which means worth buying stock is slightly more than not worth buying stock. The reasonable explanation behind this is the United States stock market has a great performance and more stocks have better performance over these 4 years. (See Exhibit 1).

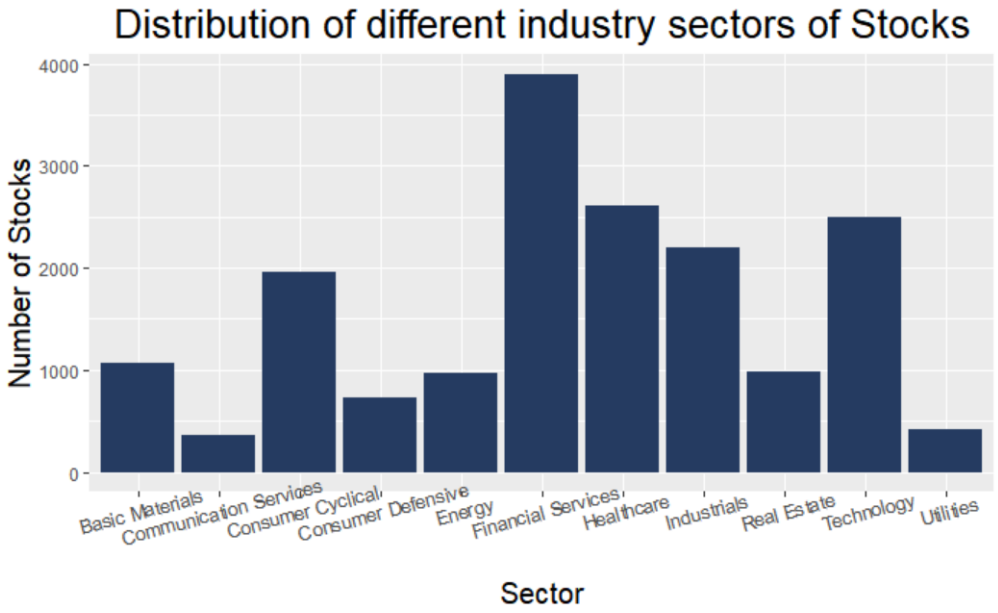
Exhibit 1: Train and Test Dataset Distribution

Data Type	Number of Class 0	Percentage of Class 0	Number of Class 1	Percentage of Class1	Total Number of Observations
Train Data	7,975	45.2%	9,685	54.8%	17,660
Test Data	1,942	44.0%	2,473	56.0%	4,415

Distribution by Sectors

The dataset includes eleven sectors in total, but observations are not equally distributed among all eleven sectors. Particularly in four of the following categories contains an extremely large number of observations, specifically over 2000 stocks: Financial Service, Healthcare, Industrials, and Technology (See Exhibit 2).

Exhibit 2



Default Limitation

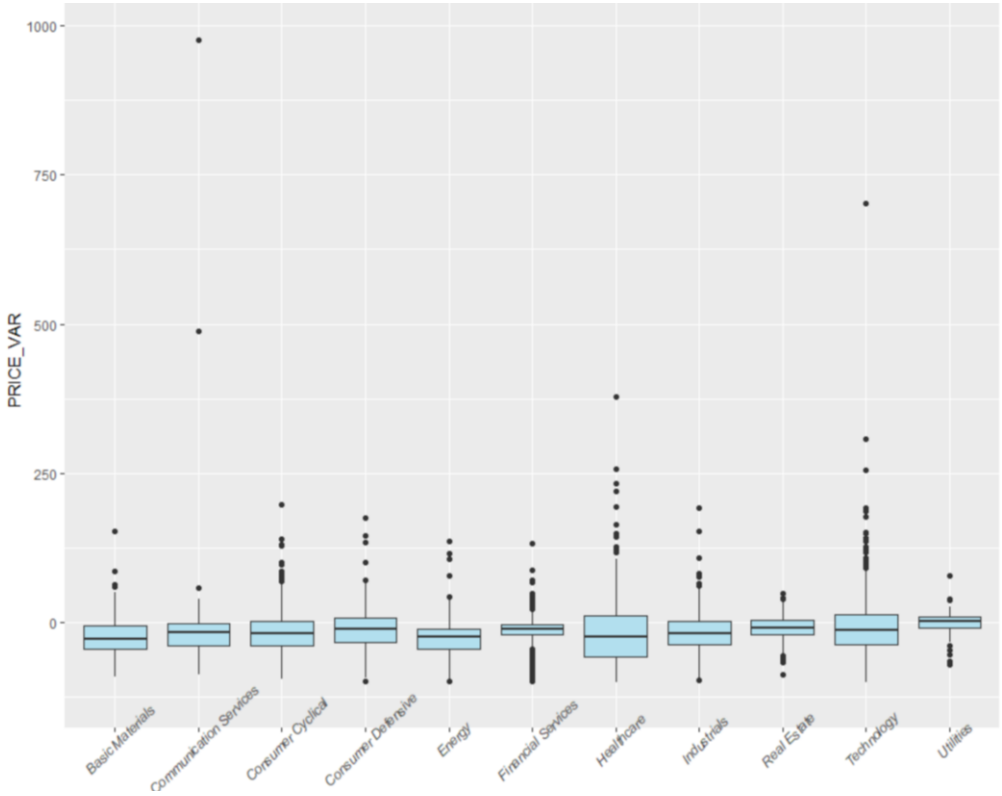
Looking in the summary of our train dataset, we have 22,075 rows of entries and 226 variables. The 226 variables mostly include all the financial ratios. The ratio between data and its variables is 77.91. We have too many variables that might create noise to the model predictions and are not efficient in model building. With this limitation, we have to narrow down the number of variables that are most related to the stock performances by applying feature engineering, otherwise, we would spend most of the time on the model running.

III. Exploratory Data Analysis

Price Variation by Industry Sectors

To explore the dataset further, we make a boxplot with price variation on the y-axis, and sectors on the x-axis. To understand the stock performance by sectors and identify the variances between each company's value, we further investigated via boxplot. Compared with different sectors, utilities have the highest median value. The basic materials sector has a low median value, it matches today's market. In today's stock market, investors pay more attention to those high-profit margin companies such as healthcare and technology. Low-profit margin companies like basic materials are always contempt, and with low P/B ratio the price variation is low. We can also see in those high-profit margin sectors such as communication services, technology, and healthcare sectors, they have higher outliers than other sectors (See Exhibit 3).

Exhibit 3



Valuable Stocks

In the stock market, there are always some stocks prices can increase at a crazy rate. In our dataset, we picked stocks with an increased rate of more than 100 percent, and group them by their sectors. We can find Healthcare and Technology has the most counts. It makes sense because the healthcare sector has the most companies associated with high increase rates. In 2017, the spread of deadly flu caused high demand in healthcare-related goods and flu shots, and this could be the direct reason

for the increase of healthcare stocks price, just like what is happening today, Clorox a disinfecting wipe making company has experienced a price skyrocket since early 2020 when coronavirus 19 outbreak. We also notice that the technology sector has the highest increase rate among all the sectors. The mean increase rate reaches 2326 percent. This happens because technology companies have very high gross margins and most companies in the technology sector are in the rapid development stage (See Exhibit 4).

Exhibit 4

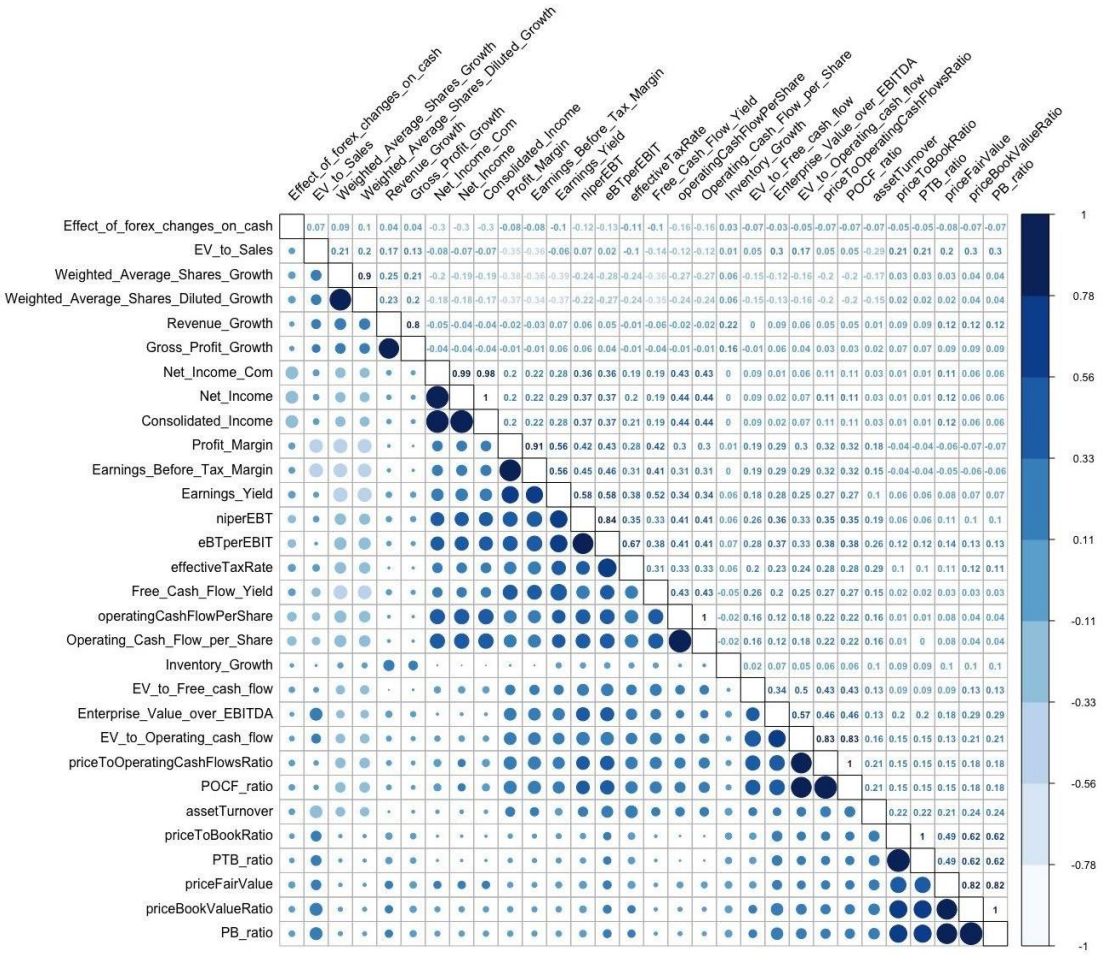
Sector	count	meanincrase
Healthcare	25	196.0345
Technology	20	2326.2810
Consumer Cyclical	5	139.4767
Consumer Defensive	4	138.8442
Energy	3	119.7087
Industrials	3	151.4514
Communication Services	2	731.6235
Basic Materials	1	152.6585
Financial Services	1	132.9897

Visualization of Correlation Matrix

The following graph demonstrates the correlation of our dataset's 30 variables for predicting classification tasks. As we can see from the graph's legend the darkest blue represents a highly positive relationship between two variables, and the lightest blue represents a highly negative relationship between two variables. For example in the following graph, "Net_ Income" and "Net_Income_Com" have a very high positive relationship among all variable pairs, their correlation reaches 0.99 which means if "net income" increases then "net income com" follows in tandem. We also found that there are few variable pairs are uncorrelated since their correlation equal to 0, they are: "Revenue_Growth" to "EV_to_Free_cash_flow"; "Net_Income_Com" to "Inventory_Growth"; "Net_Income" to "Inventory_Growth"; "Consolidated_Income" to "Inventory_Growth"; "Earnings_Before_Tax_Margin" to "Inventory_Growth"; and "Operating_Cash_Flow_per_Share" to "PTB_ratio", these variable pairs have no correlation between each other. There are also negative correlation scores that can be found in the following charts, which means variables in these variable pairs move in counter-directions. Lastly, through this correlation plot, we also found there are five pairs of variables share exactly same data inputs by taking different variables names, the first one is "Net Income" and "Consolidated_Income"; the second one is "operatingCashFlowPerShare" and "Operating_Cash_Flow_per_Share"; the third one is "priceToOperatingCashFlowsRatio" and "POCF_ratio"; the fourth one is "priceToBookRatio" and "PTB_ratio"; the fifth one is "priceBookValueRatio" and

"PB_ratio". These duplicated columns will need to remove one or other for future model data inputs to avoid trash in trash out (See Exhibit 5).
By digging into our merged datasets, we found out there are multiple duplicate columns more than what shows in the following graph with slightly different names that need to be removed before using it. We approach this by first selecting out variable pairs with a correlation equal to 1. Then we inspected these pairs individually and finally decided to remove 26 variables. The inspection and selection process can be found in the "Duplicated Column Remove Selection Process.xlsx" stored in the Github repository.

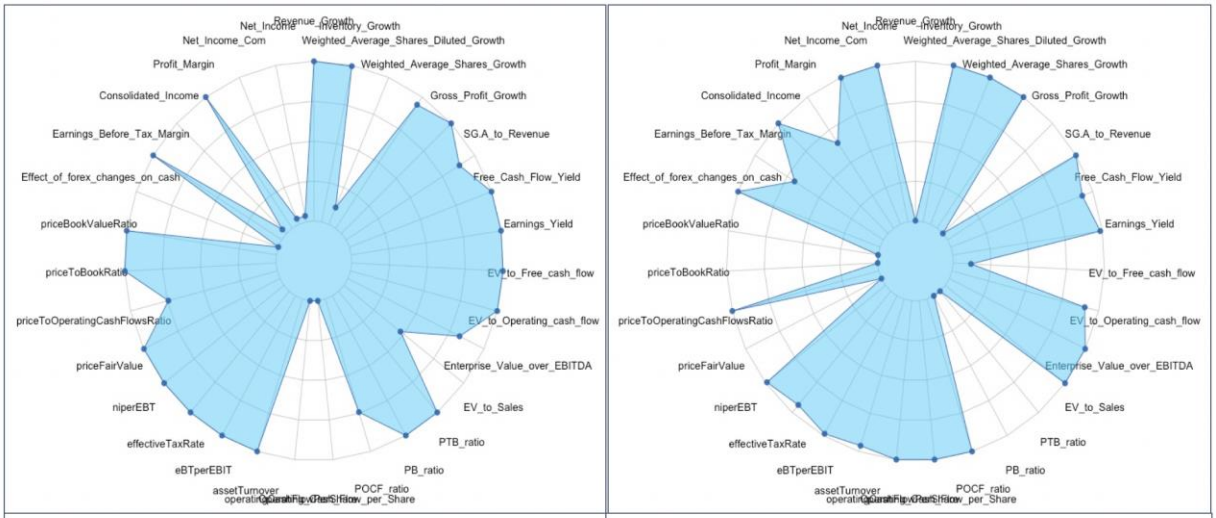
Exhibit 5: Correlation Matrix



Principal Component Analysis

Because the nature of our datasets contains too many variables and is hard to use a single plot to demonstrate the distribution of two different classes. We decided to use principal component analysis as another approach to show the data. After redundant few variables from the training dataset. We performed a principal components analysis on the training dataset, according to the results from visualized eigenvalues, k-means with silhouette scores and clustering results visualization, we decided to choose 2 clusters as our final decision for the number of clusters. Then we added back our clusters and 2 PCA variables to the original training dataset. By grouping the newly created dataset through clusters and take means for all the variables, we get our data summaries, by extracting maximum values from the summary dataset, we get a sense of the general characteristics based on all the variables of two different class, which can be used for the recommendation for general investment advice. Such as when customers are looking for stocks to invest which things they will need to pay attention to when they read companies 10K
reports. The following charts represent the characteristics of two different clusters from a radar chart perspective (See Exhibit 6).

Exhibit 6: PCA Radar Visuals



IV. Methodology

To investigate the informative variables and machine learning methods, we follow the following procedures. The initial step started with merging five datasets into one consolidated data frame. After splitting the data into train and test, we further implemented initial data cleaning and conducted an exploratory analysis in the R environment.

As introduced above, our dataset includes more than 200 financial indicators. Some variables come from financial statements such as total assets, while others are ratios such as earnings per share. Therefore, the additional data cleaning is required to deal with outliers and to standardize. In Google Colab, we removed the outliers using the function Winsorize and standardizing the data with z-scores. As a result, the values left were within the 5th percentile and 95th percentile and were rescaled, so each variable has the properties of a standard normal distribution with a mean of zero and standardization of 1.The next step is to implement features selection. We approached three methods for feature selections, including Light Gradient Boosted Machines (GBM), Random Forest, and Boruta. The Light GBM and Random Forest were conducted in the python environment, Google Colab, while the Boruta was completed in the local R studio. Last but not least, for three feature selection methods, we generated three subsets of variables that are informative to classifying the recommended stock. To continue investigating the most informative financial indicators, we applied various supervised machine learning models on the three subsets of variables, including Random Forests classifier, Naive Bayes classifier, K-nearest neighbor (KNN) classifier, Logistic Regression and more.

V. Initial Results

We are focusing on feature selection to reduce the number of variables in the dataset so that we can make a better model in the future analysis. Feature selection included three methods: wrapper method, filter method, and embedded method. In our initial results, we tried two methods under Python by using different packages.

- Filter method -- Mutual information

Mutual information is a measure of the mutual dependence between the two variables. This is one of the filter methods for feature selection, which only considers statistical probability on the training data, not using any learning algorithms. This is non-parametric tests that use k-nearest neighbors to measure the scale of the relationship between the predictor values and the target variable. The quantities reflect how much each feature tells about the response. The following plot shows each variable's importance in this dataset (See Appendix 4).

- Embedded method -- Decision tree

Instead of the statistical test, we tried using one of the learning algorithms which is the decision tree. In our initial trial, we set the threshold of our features selection as two times the mean, however, we used other criteria under our *Final Results* part (See Appendix 5).

Since our training data is large and there are multiple ways in doing the feature selection, we continued to dig in analyzing our data by using other feature engineering methods. After systematically studying feature engineering, we decided to dive into the methods mentioned in the *Final results* part. Finally, we would compare our results between those various methods and analyze the best performance model on our dataset.

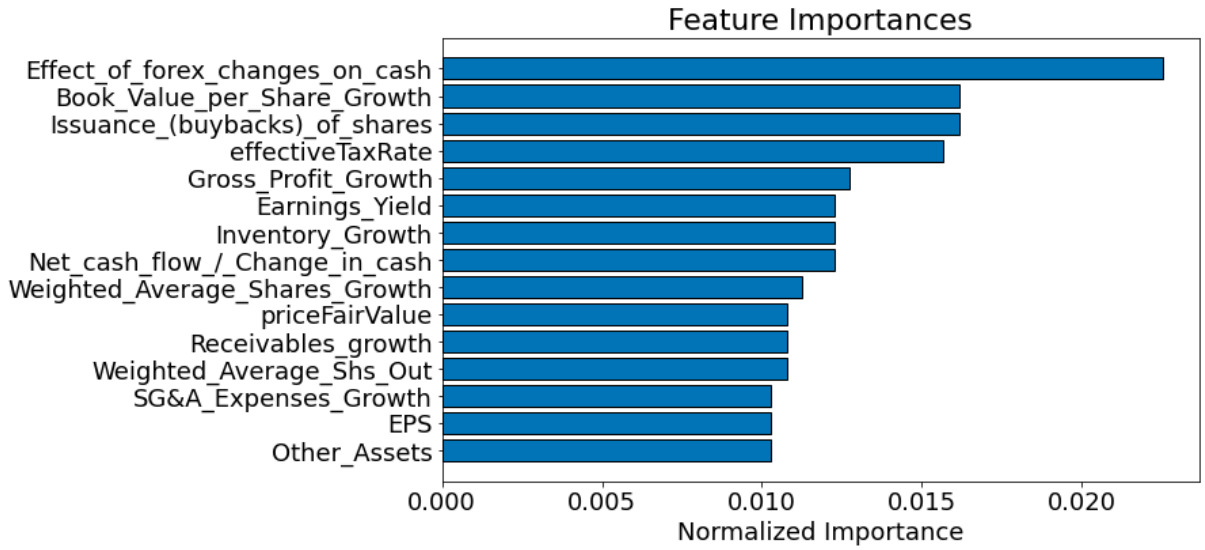
VI. Final Results

As a reference, before feature engineering, we applied the Random Forest and Gaussian Naive Bayes on the whole train dataset with 226 features and reached an accuracy of 50.8% and 49.97% correspondingly. The following feature engineering methods aim to improve the accuracy of modeling.

Feature Engineering I: *Light GBM*

LightGBM stands for lightweight gradient boosting machines, which expects to convert categorical features to integers. So it's necessary to encode the variable `sector` here to do the feature selection. This gradient boosting framework is based on the decision tree learning algorithm. Light GBM grows trees vertically while other algorithms grow trees horizontally, hence results in much better accuracy than others. It can perform very well with large datasets under less training time compared to other boosting methods. In order to avoid overfitting and randomness of the variable selection, we re-split the train data again by using the 20% of data as a validate set, then fit the light GBM model twice. 137 features reached 90% of cumulative importance, but we only choose the top 15 features in our later models based on the feature importance ranking (See Exhibit 7).

Exhibit 7: Top 15 Important Features by LightGBM

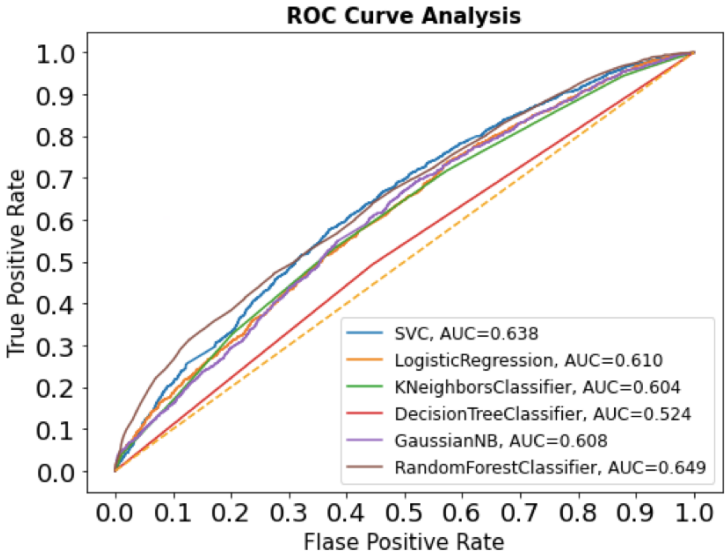


Modeling based on FE I

- Results before tuning hyper-parameters

We trained those 15 features into the training dataset and predicted our test dataset. The following are the model accuracies and the ROC curves (See Exhibit 8).

Exhibit 8: ROC Curve and Accuracy Table



Model	Accuracy
SVM	61.3%
Logistic Regression	59.8%
Decision Tree	52.0%
KNN	58.1%
Naive Bayes	59.7%
Random Forest	60.5%

Based on the results we got, we can see that random forest and SVM models return the best two results among the 6 models. The reason we plot the ROC curve is that only the accuracy cannot indicate that the model is good due to a bad model still can get higher accuracy. Since accuracy is computed at the threshold of value 0.5, AUC (area under the curve) is the average of all threshold values. According to our results, both AUC and

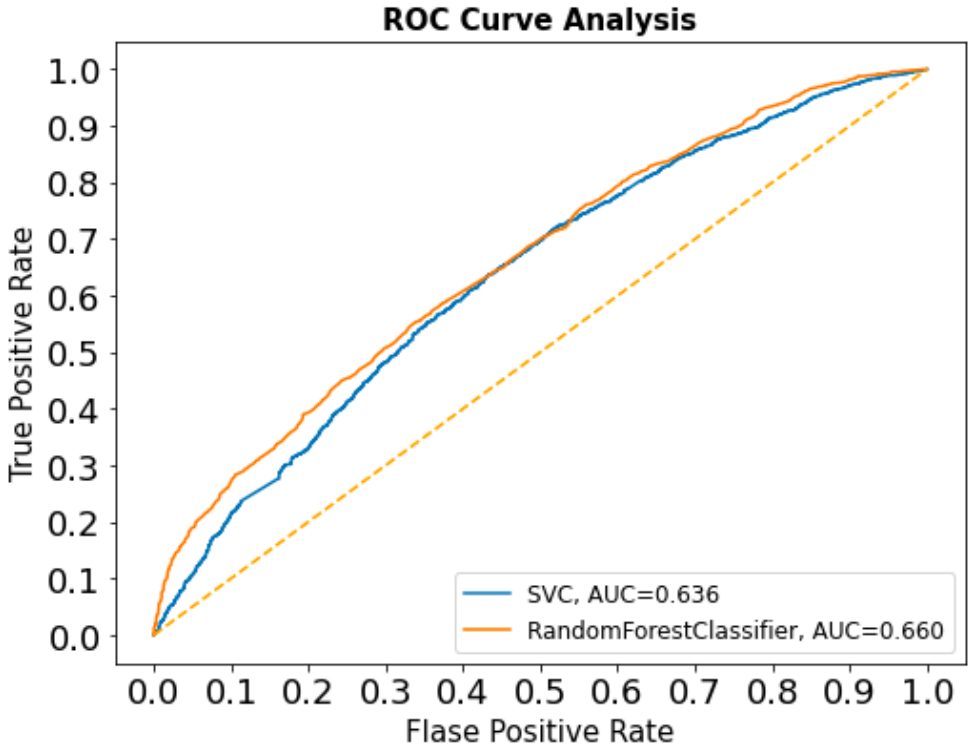
accuracy show that random forest and SVM are the best two models under the default setting value.

However, the machine learning model seems like a Black-Box, we can tune the hyper-parameters under different classifiers so that to improve our model. Hence, we make more effort on analyzing these two models which tuning parameter C and gamma in the SVM and n-estimators in the random forest.

- Results after tuning hyper-parameters

After tuning the hyper-parameter for both models, we can find that both models' accuracy is 61.3%, the value of AUC changed inversely. Particularly, the AUC value increased by over 1% under the random forest model. Combined with these two values, we will choose the random forest as our best classifier to predict our models (See Exhibit 9).

Exhibit 9: ROC Curve after Tuning Hyper-parameter

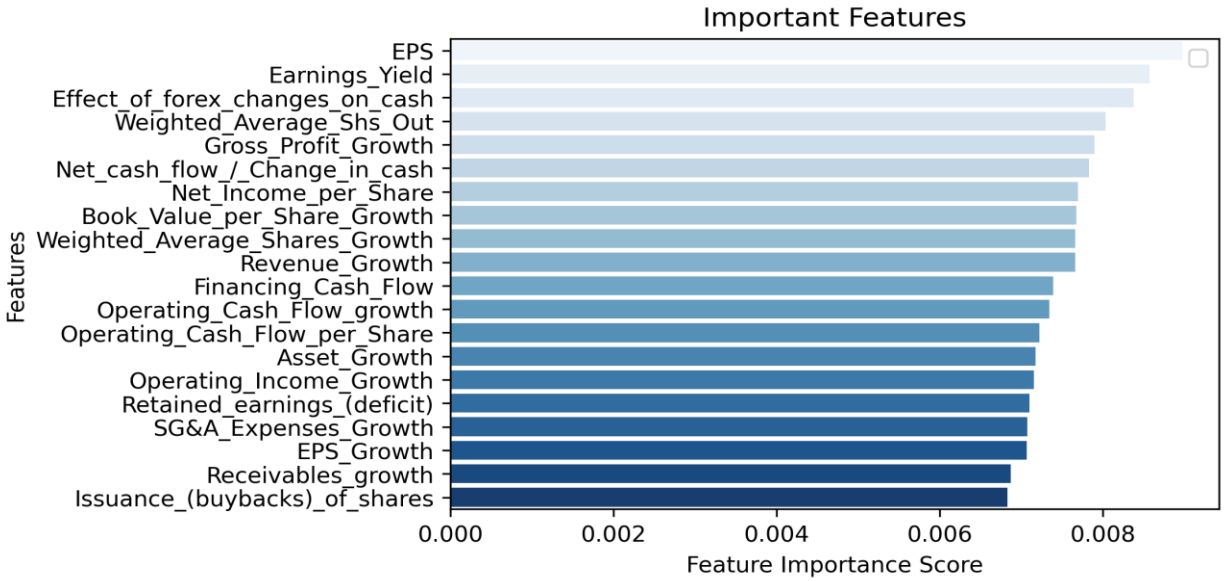


Feature Engineering II: *Random Forest*

Random Forests as an ensemble model made up of decision trees are often used for feature selection in a data science workflow because tree-based methods naturally rank by how well they improve the purity of the node. The random forests used in this project used 100 classifier trees which is the hyper-parameter for the random forest algorithm. To ensure the stability of the result generated by the model, we utilized a while loop to

examine the model 30 times using training data to ensure some features appearing to be important were not random cases. After sorting the importance based on the mean importance of each feature, we found that achieving 90% of cumulative importance requires 47 features. We finally filter 20 variables out of 206 variables (See Exhibit 10).

Exhibit 10: Top 20 Important Features by Random Forest



According to the result of feature engineering via Random Forest, Earnings per Share (EPS), Earnings Yield, and Effect of Foreign Exchange Rate on Cash are the top three outstanding features, while the Gross Profit Margin, Net Cash Flow/ Change in Cash follow. The Net Income per Share, Book Value per Share Growth, Weighted Average Share Growth, and Revenue Growth contribute the relatively same amount. From the perspective of finance and accounting, the selected features are informative. EPS calculated by net income diverge by the shares of outstanding is one of the essential variables in determining a share's price. Similar to EPS, Earnings Yield, as the inverse of the P/E ratio, shows the percentage of how much a company earned per share and if often used by investors to determine which assets seem underpriced or overpriced. Different from EPS and Earning yield that presents the operating ability of the company, the Effect of Foreign Exchange Rate on Cash measures the impact on the value of existing cash balance in the reported currency due to fluctuations in foreign exchange rates given that some or all of the cash may be held in foreign currencies. Another thing worthy of notice is half of the selected features are growth rate. One plausible explanation is that the growth rate compared to numbers reveals more information because of accounting for the change in a period. Besides, among elected features, the cash related features appear frequently.

Modeling based on FE II

With the 20 variables generated by the feature engineering of random forest (FE II), we further investigated various supervised machine learning methods including Radial Support Vector Machine, K-nearest neighbors, Gaussian Naive Bayes, Logistic regression, Decision Tree, Random Forest, Bagged KNN, Adaptive Boosting, Stochastic Gradient Boosting and eXtreme Gradient Boosting (XGBoost). Random Forest, XGBoost, Stochastic Gradient Boosting, and SVM outperform the rest of the models with accuracies around 60.14% (See Exhibit 11). Due to the top four outstanding models are tree-based methods except for SVM, visualizing the features playing an important role (see Appendix 2) is available. In addition, Random Forest, Stochastic Gradient Boosting, and XGBoost maintain consistent performance in ROC analysis and cross-validation. However, the ROC curve points out that SVM is not a qualified model with an AUC score of less than 0.5 (See Exhibit 12). Moreover, we continued conducting hyper-parameter tuning for the best models Random Forest and XGBoost. For the Random Forest, we further the number of estimators(trees) for Random Forest, n_estimators, max depth, and learning rate for the XGBoost. The results show a slight improvement. XGBoost improved the accuracy from 60.14% to 60.20% with 200 number of estimators XGBoost, max depth of 4, the learning rate of 0.001. The best number of estimators for Random Forest is 300, and the best score improved from 60.14% to 61.33%. The improvement of both methods is minor. In the end, the best-performed model is Random Forest for the 20 variables selected by FE II. The ROC curve and the feature importances for two methods are presented below (See Exhibit 13).

Exhibit 11: Accuracy Table

Method	Accuracy
Random Forest	60.14%
XGBoost	60.14%
Stochastic Gradient Boosting	60.14%
Adaptive Boosting	60.14%
SVM	60.14%
KNN	58.82%
Logistic Regression	58.05%
GNB	57.92%
Decision Tree	54.70%

Exhibit 12: ROC Curve

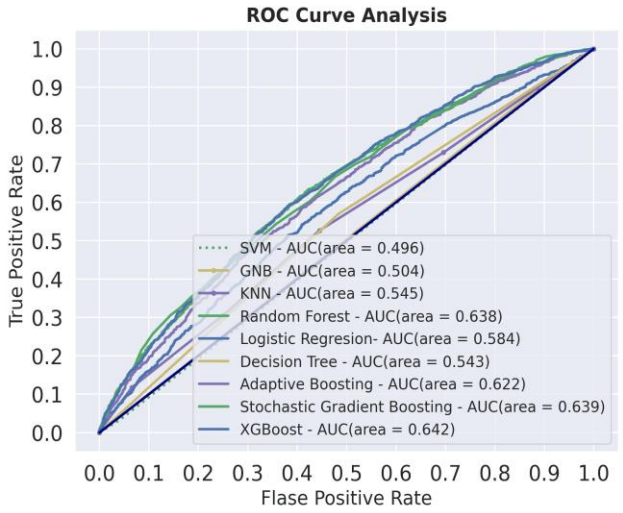
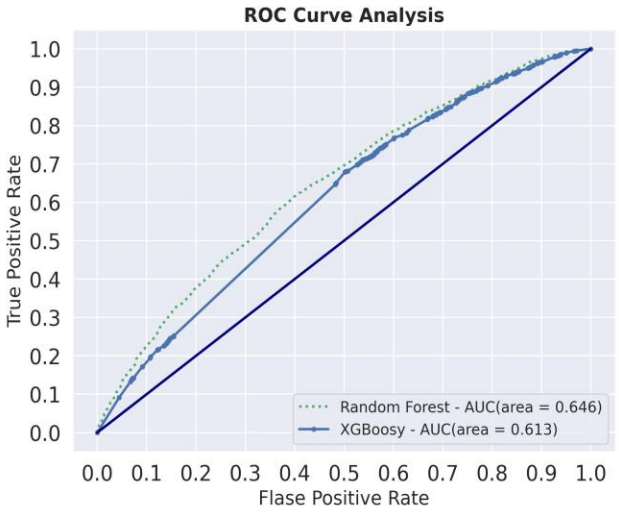


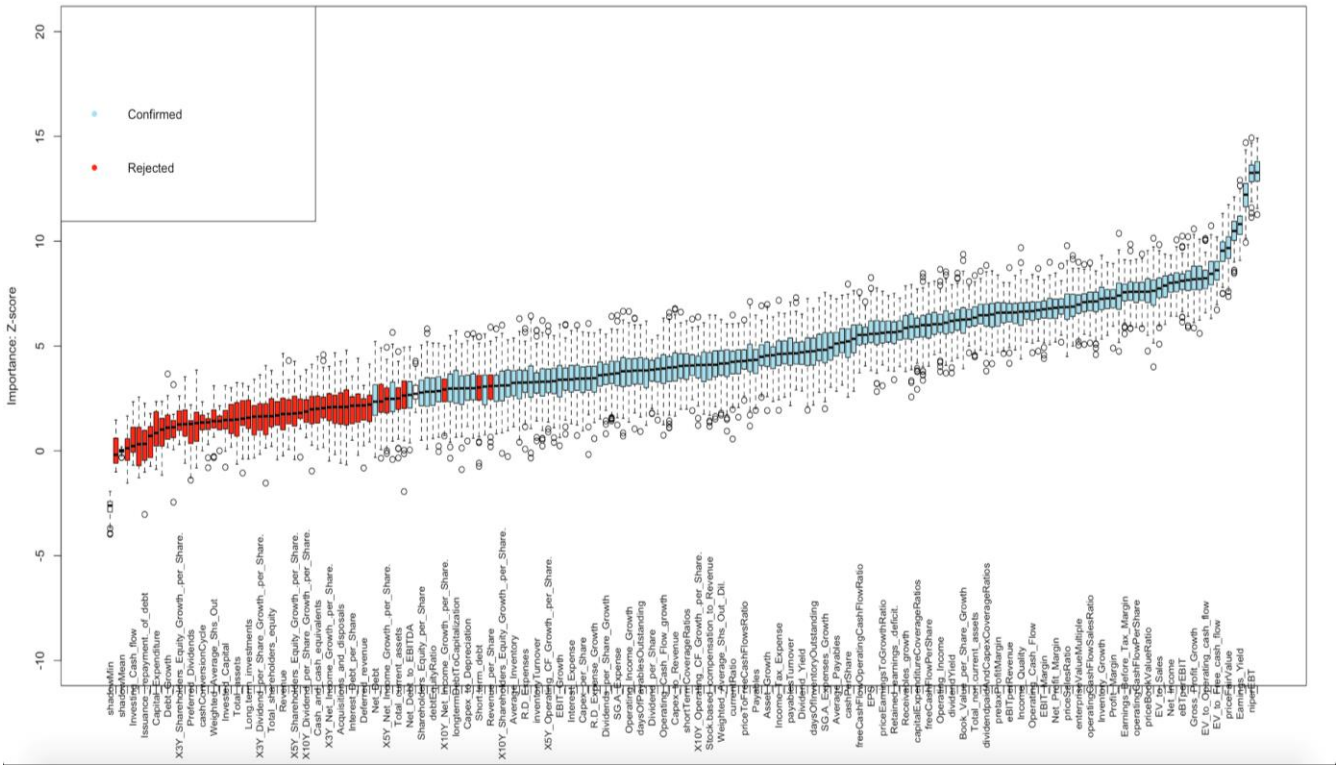
Exhibit 13: After Tuning Hyper-parameter



Feature Engineering III: *Boruta*

Boruta as a wrapper method of feature selection. In the iteration of feature selection, the boruta algorithm first duplicates our training datasets and shuffles each column values which are called shadow features. Then by performing a random forest classification, each shadow feature gets its importance. By comparing the real features' importance with shadow features' importance, the algorithm will record features with higher importance into the predefined vector and call them hits (Manish Pathak, 2018). After 200 iterations (200 is a predefined set of iterations, the default number is 100) we get a table of hits. And within each iteration by comparing the importance of the features from duplicated datasets and original datasets, we can define which hits are important or unimportant by assigning them "Confirmed" or "Rejected" or "Tentative", if the feature is assigned tentative it will be brought into the next iteration until it is defined or predefined iterations runs out. The model will stop itself under two conditions, first is when the predefined iterations run out, second is when all the features have been either confirmed or rejected. In our case after the 5.83 hours selection process, we have 20 variables defined as tentative when our model stops while 131 variables have been confirmed, and 46 variables have been rejected. The following graph shows both confirmed and rejected variables with their importance score ranked (See Exhibit 14).

Exhibit 14: Z_score of every features in the shuffled dataset

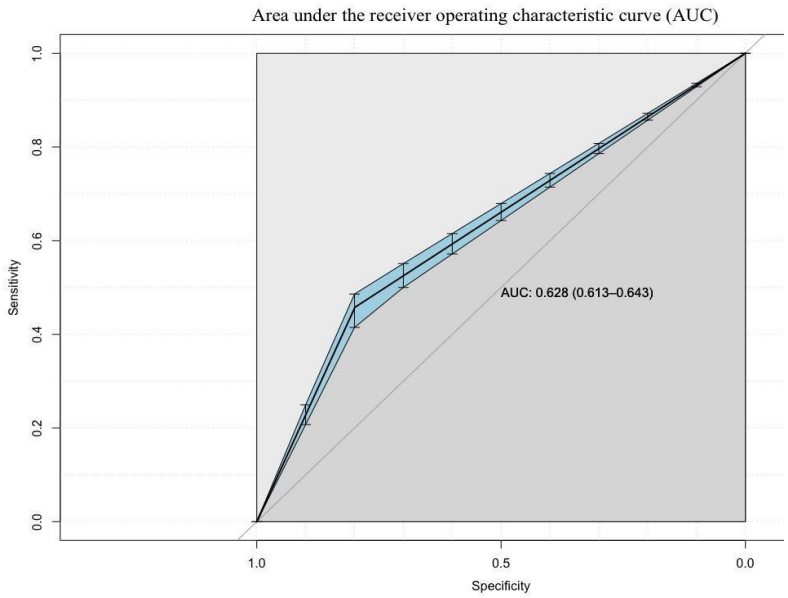


Modeling based on FE III

With 131 confirmed variables generated by the feature engineering of Boruta (FE III), we performed a generalized linear model by using different variable combinations according to variables mean importance ranking. The reason we choose using logistic regression is that we have a binary outcome “0” and “1” which represents “Not Buy” and “Buy” besides almost all of our predictor variables are continuous variables. After many iterations of using different variable sets, this model reaches its state of arts of the accuracy of 69.4 within 95% of the confidence interval of 68.1% to 70.8% with 42 variables. And this model has 2434 true positive cases out of 4392 cases within our test dataset. It got an AUC (area under the receiver operating characteristic curve) value equal to 0.628 (0.613-0.643) which is better than randomly picking a stock for investment (See Exhibit 15).

Exhibit 15: Confusion Matrix and AUC

```
Confusion Matrix and Statistics

          Reference
Prediction    1    0
         1 2434  730
         0  612  616

               Accuracy : 0.694
                 95% CI : (0.681, 0.708)
    No Information Rate : 0.694
    P-Value [Acc > NIR] : 0.4552

                  Kappa : 0.263

 Mcnemar's Test P-Value : 0.0014

            Sensitivity : 0.799
            Specificity : 0.458
         Pos Pred Value : 0.769
         Neg Pred Value : 0.502
             Prevalence : 0.694
         Detection Rate : 0.554
   Detection Prevalence : 0.720
      Balanced Accuracy : 0.628

       'Positive' Class : 1
```

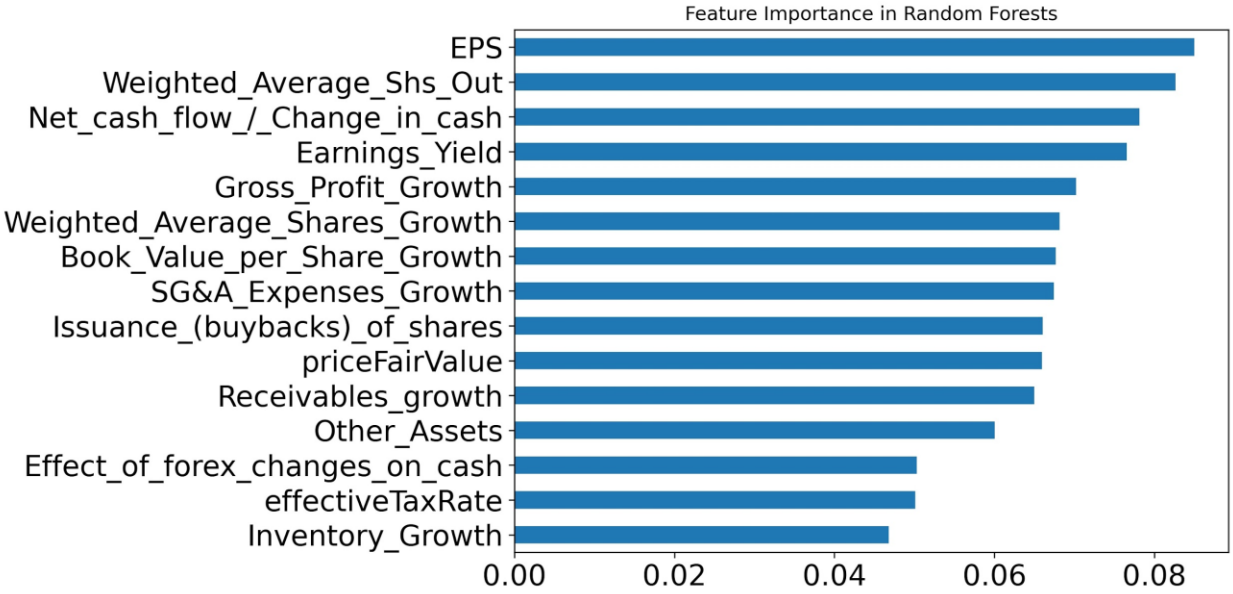


Lastly, we demonstrated two different classes' characteristics based on selected predictable variables in the following two radar charts (See Appendix 6).

VII. Discussion

To summarize the modeling results with three subsets of variables selected from three types of feature engineering, the model with the highest accuracy was generated by the Random Forest classifier using the 15 features selected from the feature engineering method, light GBM (*FE I*). As mentioned in the `Feature Engineering I` part, Light GBM grows tree leaf-wise while other algorithms grow level-wise. In other words, the leaf-wise algorithm can reduce more loss when compared to a level-wise algorithm. Moreover, Light GBM provides a parallel learning algorithm that could parallel features and data. The best model under Light GBM feature selection is the random forest, with accuracy 61.3% and the value of AUC 66%. As shown in Exhibit 16, the top three features are `EPS`, `Weighted Average Shs Out`, and `Net Cash Flow / Change in Cash` (See Exhibit 16), in which those features are based on the feature selection of Light GBM.

Exhibit 16: Feature Importance In Random Forest



The above result is also consistent with the three feature engineering methods. `Effect of Foreign Exchange Rate on Cash`, `Earnings Yield`, and `Effective Tax Rate` are the three most common features that were captured by the feature selection method of Light GBM, Random Forest, and Boruta. However, after applying various features that were selected by Light GBM, random forest, Burota into several models, `EPS`, `Earnings Yield`, and `Weighted Average Shs Out` became the most common features. From the perspective of finance and accounting, the selected features are informative. EPS calculated by net income diverge by the shares of outstanding is one of the essential variables in determining a share's price. Similar to EPS, Earnings Yield, as the inverse of the P/E ratio, shows the percentage of how much a company earned per share and if often used by investors to determine which assets seem underpriced or overpriced. Different from EPS and Earning yield that presents the operating ability of the company. The weighted average of outstanding shares is a calculation that incorporates any changes in the number of outstanding shares over a reporting period. It is also an important number in calculating the earnings per share (EPS). All these top three features are related to each other, which could reflect the company's operating ability.

VIII. Criticism and Future Improvement

The difficulty level of our project objective is out of our expectation from the beginning. The process of data inspection, data cleaning, and feature selection for these unorganized datasets is very time-consuming. Data cleaning was not a one-off process for our project based on the nature of our dataset. It's often the case that we have found minor problems related to the dataset during the process of feature selection and modeling, so we have to go back and reclean the dataset based on previous approaches then dive into feature selection and modeling. In order to test result accuracy, we take different approaches to inspect our model performance. Throughout the whole project, we applied our knowledge of coding into this specific task and interpreted the results with our understanding of the financial industry.

When it comes to modeling with bulk data and limited compute ability, we have to make trade-offs between accuracy and running time. We ran into a situation many times for models like Boruta and cross-validation for the SVM could take more than 7 to 8 hours to run. Therefore, we choose to save running time on running at the sacrifice of accuracy. For instance, we chose a k folder of 5 instead of 10 majorities of the time when doing k-folder cross-validation and also narrow down the range of parameters when tuning the hyper-parameter. We are also aware that the decision we made might be the cause of the modest improvement of hyper-parameters running. Besides, trade-offs also happen between the interpretability and complexity of the model. After evaluating various models, we often found it is very challenging to explain why some models outperform others in theory.

One future improvement for this project is that we could get more data related to individual companies and possibly analyze COVID-19 impact on the stock market. As we noticed, we have experienced unusual changes in the stock market in 2020. With the work from home policy, global recessions have affected the economy globally. The dataset we used from Kaggle only collected the data from 2014 to 2018, but the year 2020 market performance could be great practice on evaluating the model.

IX. Conclusions

With regard to the purpose of our project, we aim to discover informative features in the financial statements to help investors make better decisions. Along with the dataset with more than 200 financial indicators, we completed feature engineering in three aspects and implemented various models to distinguish valuable stocks over 22,075 observations. The model with the highest accuracy is Random Forest, with features selected through LightGBM. The Random Forest generates an accuracy of 61.3% with an AUC 66%. According to the modeling result, the most influential variables are earnings per share (EPS), Weighted Average Share Outstanding, and Net Cash Flow

which are also consistent with the outcomes generated by the other two features engineering methods. Indeed, the unpredictability of the stock market with tremendous randomness is shared knowledge. To conclude, we advise potential feature investors to make their decisions of picking stocks based on features from our best performed model results and lay their eyesights on features with top importance scores since the majority of these features reflect either the ability to generate the profits or the change in cash to identify value stocks and make wise decisions.

X. References

Bhattacharyya, Indresh. “Feature Selection (Boruta /Light GBM/Chi Square)-Categorical Feature Selection.” Medium, Medium, 18 Sept. 2018, medium.com/@indreshbhattacharyya/feature-selection-categorical-feature-selection-boruta-light-gbm-chi-square-bf47e94e2558.

Chuan, Yu, “Common Methods For Feature Selection You Should Know.” Medium, 17 Jun. 2018, https://medium.com/@cxu24/common-methods-for-feature-selection-you-should-know-2346847fdf31

“Figure 2f from: Irimia R, Gottschling M (2016) Taxonomic Revision of Rochefortia Sw. (Ehretiaceae, Boraginales). Biodiversity Data Journal 4: e7720. Https://Doi.org/10.3897/BDJ.4.e7720.” doi:10.3897/bdj.4.e7720.figure2f.

Daniel Homola. “BorutaPy – an All Relevant Feature Selection Method.” Daniel Homola, 8 Feb. 2016, danielhomola.com/2015/05/08/borutapy-an-all-relevant-feature-selection-method/.

Klebnikov, S. (2020, February 4). Here's Why Tesla Stock Just Surged Past A Record $900 Per Share. Retrieved from https://www.forbes.com/sites/sergeiklebnikov/2020/02/03/heres-why-tesla-stock-just-surged-to-a-record-780-per-share/#16ce29485ee7

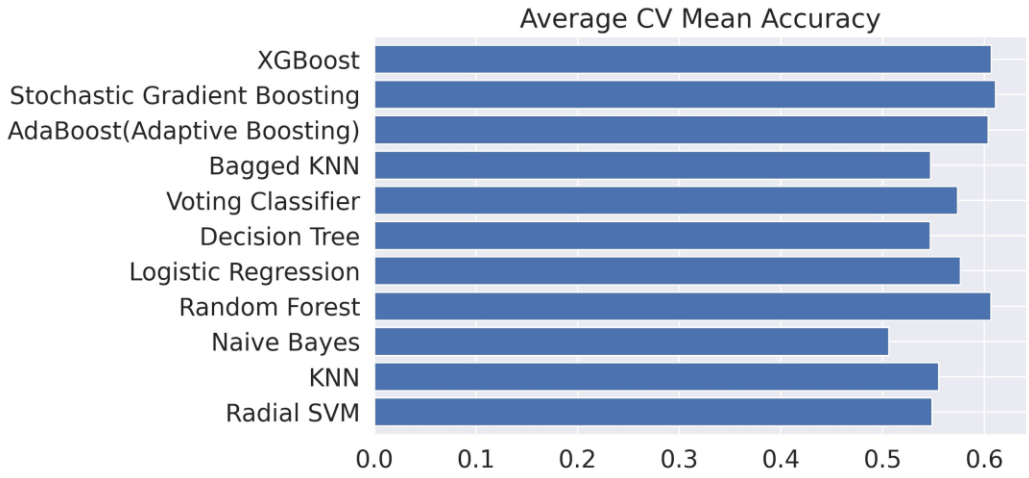
Nguyen, Vu. “Bayesian Optimization for Accelerating Hyper-Parameter Tuning.” 2019 IEEE Second International Conference on Artificial Intelligence and Knowledge Engineering (AIKE), 2019, doi:10.1109/aike.2019.00060.

Pathak, Manish. “Boruta Feature Selection in R.” *DataCamp Community*, 7 Mar. 2018, www.datacamp.com/community/tutorials/feature-selection-R-boruta.

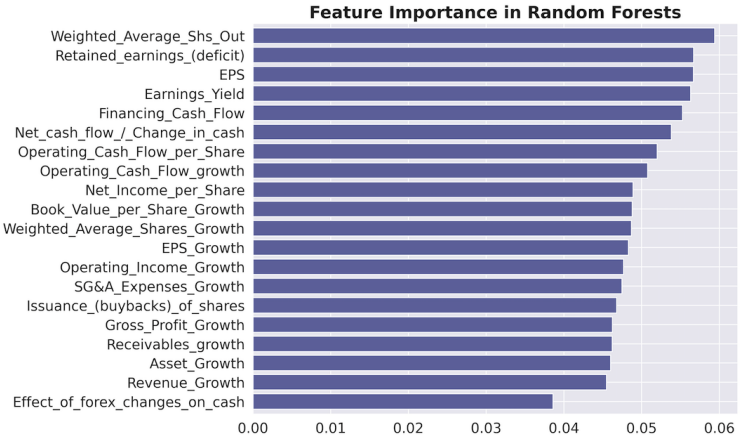
Mandot, Pushkar, “What Is LightGBM, How To Implement It? How To Fine Tune The Parameters?”, Medium, 17 May, 2017, https://medium.com/@pushkarmandot/https-medium-com-pushkarmandot-what-is-lightgbm-how-to-implement-it-how-to-fine-tune-the-parameters-60347819b7fc

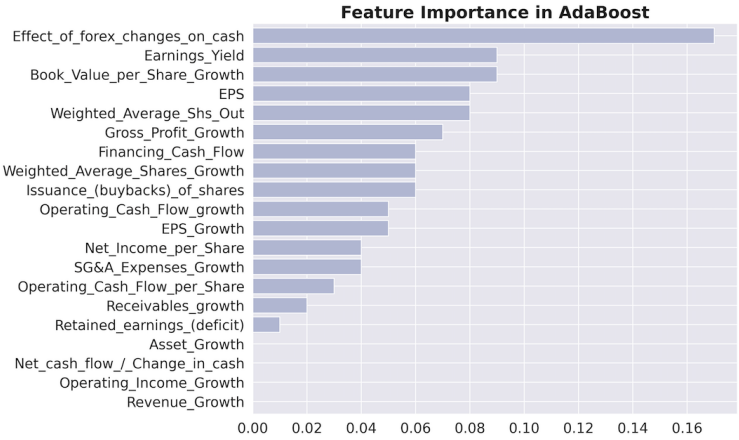
XI. Appendix

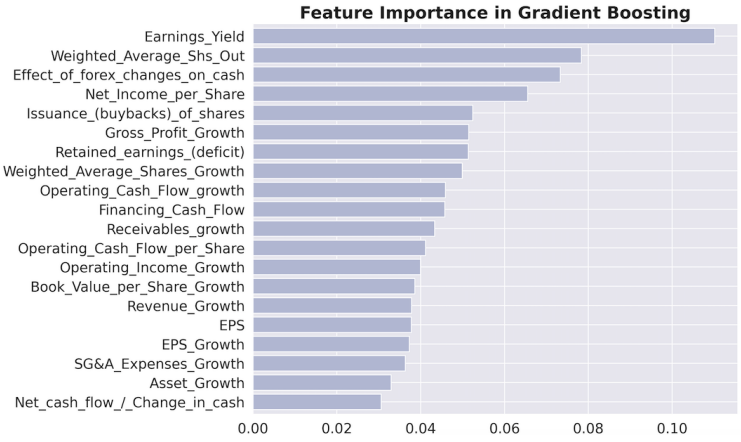
Appendix 1: Models of FE II--Average CV Accuracy Plot

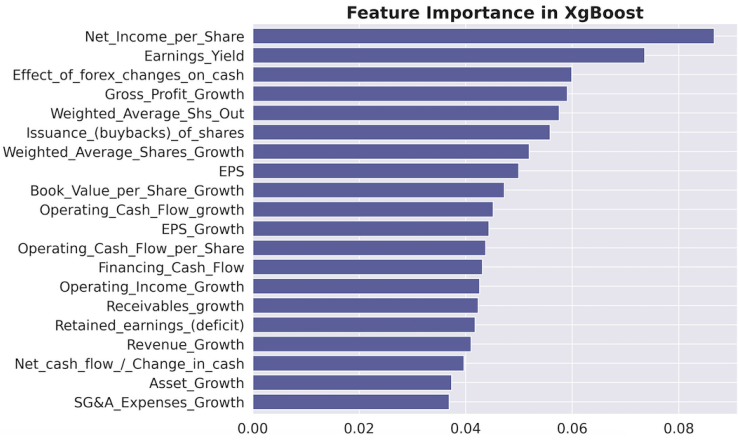


Appendix 2: Models of FE II--Feature Importance of Four Models

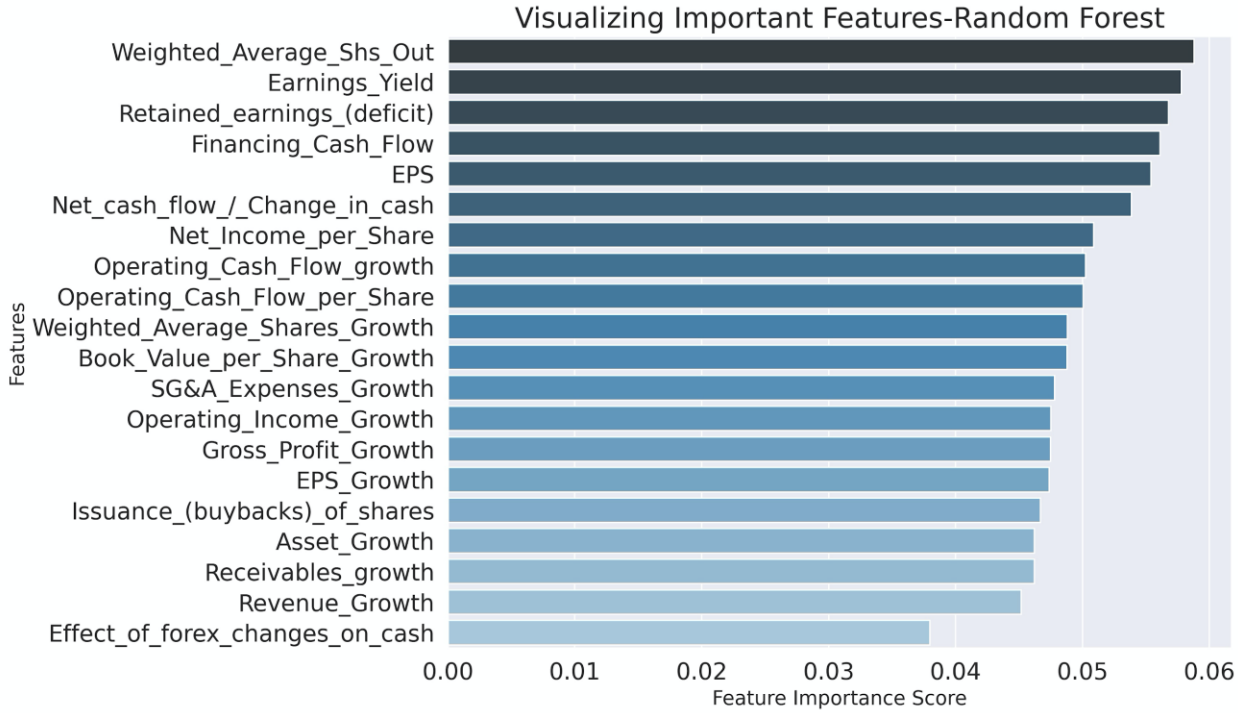


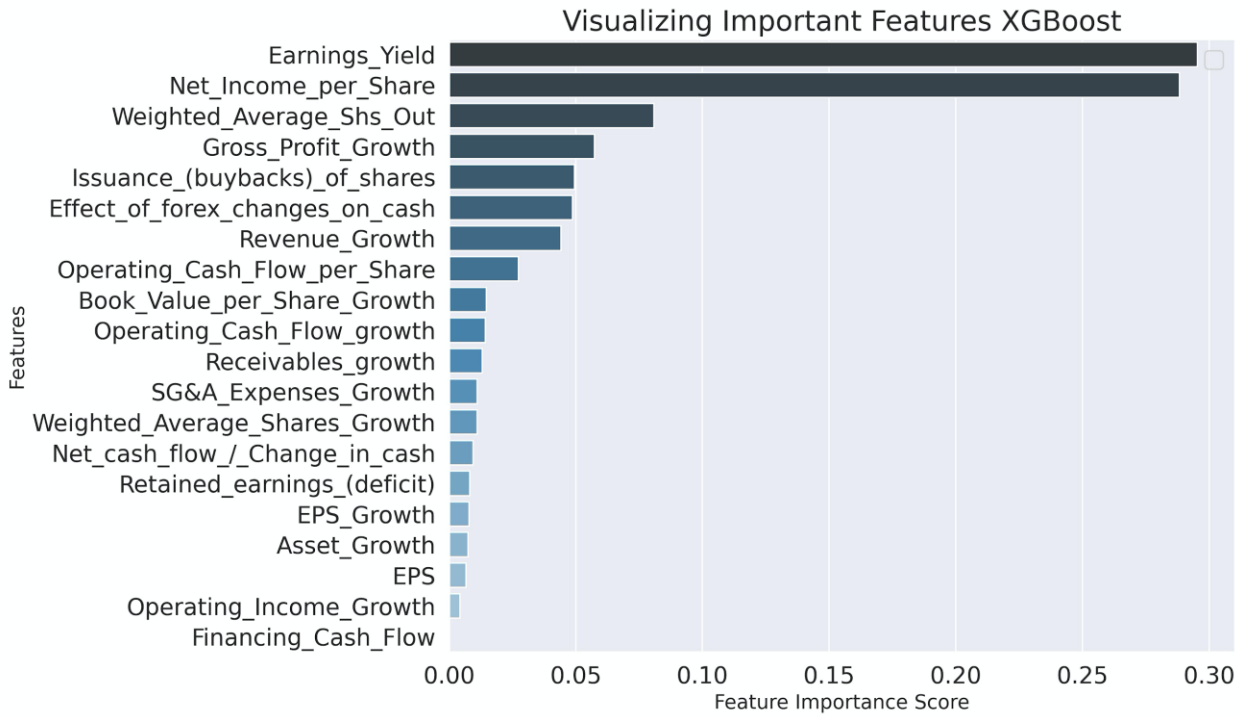




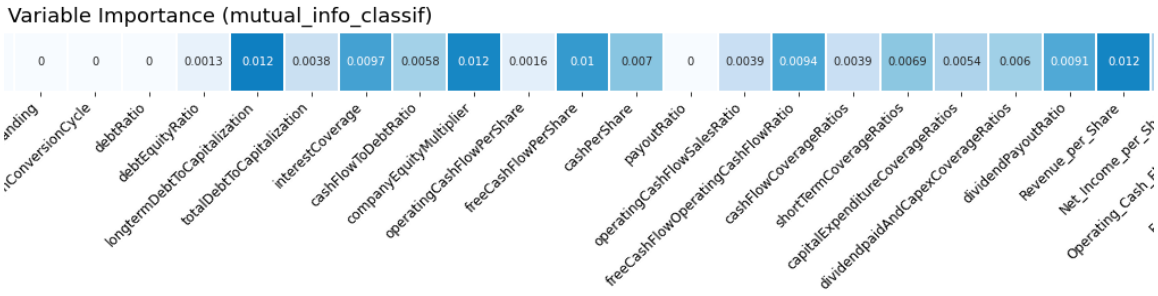


Appendix 3: Models of FE II--Feature Importance After Tuning Hyper-parameter

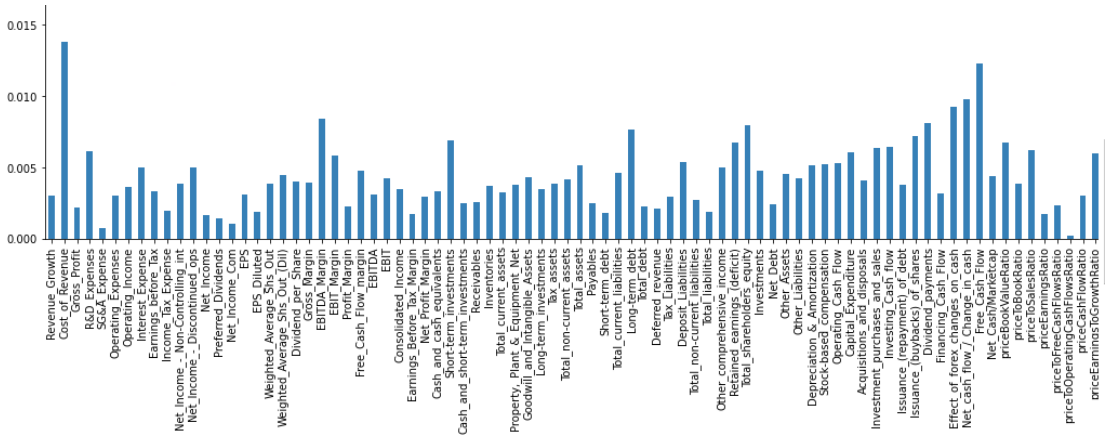




Appendix 4: Feature Importance for Mutual Information



Appendix 5: Feature Importance for Decision Tree Classifier



Appendix 6: Radar Charts based on Class