200+ Financial Indicators Visualization and Analytics of US Stocks in 2015-2019

Honglong LI, Supervised by Huamin QU

The Hong Kong University of Science and Technology, Hong Kong 999077, China

Abstract. This project makes an indicator analysis and indicator learning on financial indicators in US stocks. During the data preprocessing part, extreme value, missing value, wrong value, and imbalanced data are handled carefully. In the indicator learning part, I use the XGBoost model to train these indicators, so as to predict the price variation. In some sectors, its testing MSE can be lower than 300 and has a nearly 90% classification accuracy. During the indicator learning, I also find that the feature importance to each sector and each year is quite different, and feature popularity does not imply the feature importance. In the indication correlation part, I find the correlation between price variation and indicators (positive, negative, no influence) in each year and each sector, which can help make conclusions that which indicator usually has what influence on which sector. Also, whether the indicator's correlation has significant influence does not imply whether it is important for making contributions to the price variation prediction.

Keywords: Stock Indicators · Data Visualization · Data Analytics

1 Introduction

Stock data visualization and analytics is always a popular topic in recent years. For one stock in the stock market, it has more than two hundred indicators, such as the revenue and the gross profit. For an outsider, it is hard to understand every indicator's concept and how it affects the price variation (positive, negative, or no influence), and how significant its influence. Finding value in stocks is an art that very few people mastered. Fortunately, data visualization and analytics technique can be applied to the financial indicators of the stock market.

Dataset Description The stock market information, including the indicators' change and stock price variation, is from Kaggle dataset ***200+ Financial Indicators of US stocks (2014-2018)*** [5]. There are five datasets for each year. Each dataset contains 200+ financial indicators, that are commonly found in the 10-K filings each publicly traded company releases yearly, for a plethora of US stocks. On average, there are 4k stocks listed in each dataset. Here, I would like to introduce some of the indicators [4]:

– Revenue: The income or increase in net assets that an entity has from its normal activities

- EPS: The monetary value of earnings per outstanding share of common stock for a company.
- Operating income: An accounting figure that measures the amount of profit realized from a business's operations, after deducting operating expenses such as wages
- Gross margin: A company's net sales revenue minus its cost of goods sold (COGS). In other words, it is the sales revenue a company retains after incurring the direct costs associated with producing the goods it sells, and the services it provides.

1.1 Objective

The first objective is to do data preprocessing. It is always important to make data suitable for data visualization and machine learning. The second objective is indicator learning. In this part, I not only need to predict the price variation but also find what indicators contribute most to the price variation change. The third objective is the indicator correlation. After finding what indicators contribute most to the price variation change, I need to find what kinds of influence the indicator has on the price variation: positive, negative, or no influence. The source code is put in GitHub: https://github.com/EririSawamura/200-Financial-Indicators-Visualization-and-Analytics-of-US-Stocks-in-2015-2019.

2 Data Preprocessing

After a sparse data analysis, I find that the stock data source is not clean, so I need to do some data preprocessing.

2.1 Imbalanced Data

An imbalanced dataset is relevant primarily in the context of supervised machine learning involving two or more classes. Imbalance means that the number of data points available for different classes is different: If there are two classes, then ideally balanced data would mean 50% points for each of classes. Imbalanced data is not a new problem. As Krawczyk [7] says in his paper, this problem has been widely discussed in the last two decades. One of the first deeper analyses of learning from imbalanced data was related to convergence rates of back-propagation-trained neural networks in Anand, Rangachari, et al's paper[3]. In their paper, they conclude that an imbalanced training set will cause the slow rate of convergence of net output error when using the back-propagation algorithm to train neural networks for a two-class problems in which the numbers of exemplars for the two classes differ greatly. For most machine learning techniques, little imbalance is not a problem. For example, if there are 60% points for one class and 40% for the other class, it should not cause any significant performance degradation. Only when the class imbalance is high, e.g. 90% points for one class and 10% for the other, standard optimization criteria or performance measures may not be

effective and would need modification. A typical example of imbalanced data is encountered in an e-mail classification problem where emails are classified into ham or spam. The number of spam emails is usually lower than the number of relevant (ham) emails. Therefore, using the original distribution of two classes leads to an imbalanced dataset. Using accuracy as a performance measure for highly imbalanced datasets is not a good idea. For example, if 90% points belong to the true class in a binary classification problem, a default prediction of true for all data points leads to a classifier which is 90% accurate, even though the classifier has not learned anything about the classification problem at hand!

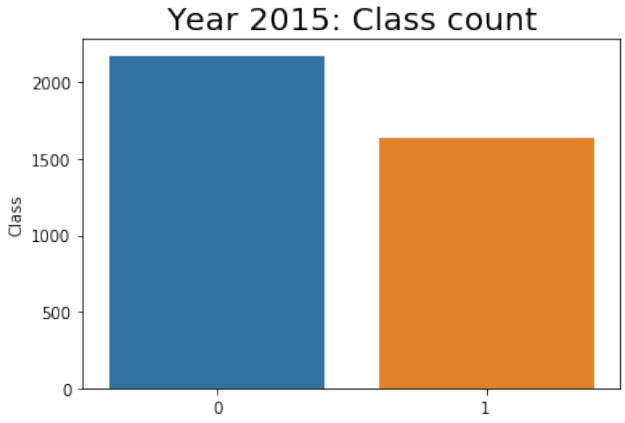

Fig. 1. Stock class distribution

Figure 1 is the stock class distribution in 2015 and figure 2 is the stock sector distribution in 2015. Neither class nor sector is the stock indicator. For class, its value is 1 if this stock's price variation is positive, otherwise, it becomes 0. The sector is like the group name of the stocks. As shown in figure 1 and figure 2, the class distribution is not so balanced, but it is ok. However, the stock sector is quite imbalanced: The sector ***Financial Services*** has more than 600 stocks but the sector ***Communication Services*** has only around 100 stocks. The imbalanced sector may lead to overfitting if I use the whole stock data in 2015 for training. To address this problem, it is useful to use the stratify option available within train_test_split functions

2.2 Find Outliers and Errors in Price Variations

It is always important to make sure that the target data makes sense. The price variations may contain some mistakes (for instance typing mistakes or unreasonable values). A quick plot of price variations in each sector allows us to find outliers and errors.

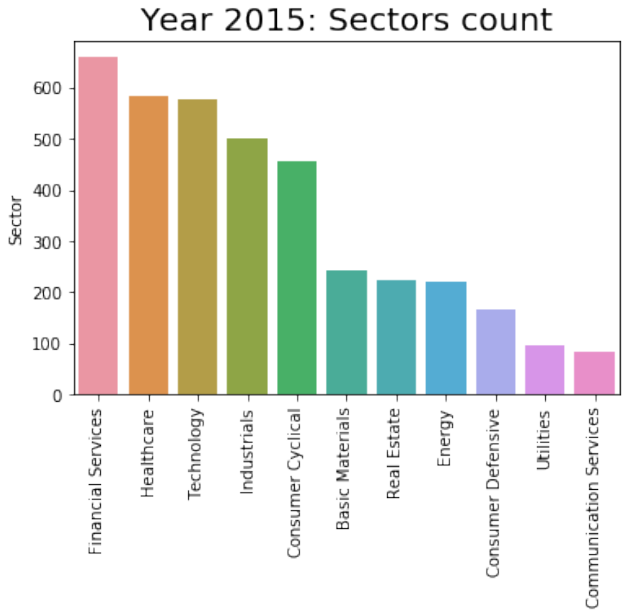

Fig. 2. Stock sector distribution

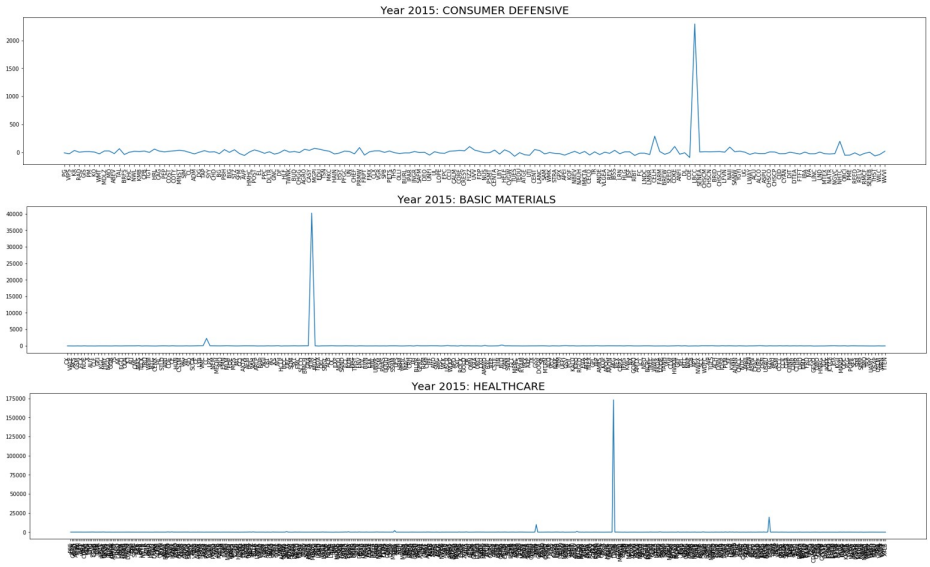

Fig. 3. Stock outliers example

Figure 3 is stock price variation (%) for sectors in 2015. As shown in this figure, some stocks experience an incredible gain in 2015. However, I am not sure that each of these gains is due to the trading activity. To address this problem, I check the price trend in a whole year for those stocks that increased their stock price by more than 500%. Here, I use pandas_datareader to query the ***Adjusted Close Price*** in 2015 from Yahoo! Finance. As described in this tutorial [10], Yahoo! Finance is a media property that provides financial news, data including stock quotes, and financial reports. It also offers some online tools for personal finance management. Learning from this tutorial and using pandas_datareader, I can find the ***Adjusted Close Price*** of the required stocks during 2015 and make some plots to represent them.

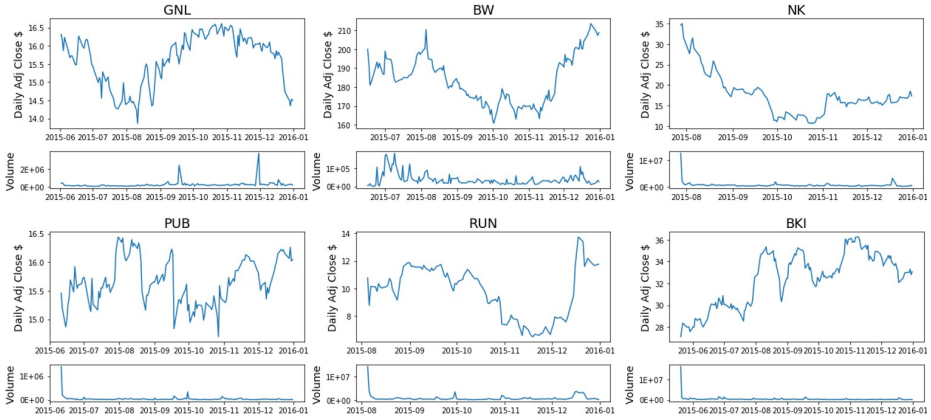

Fig. 4. Stock example who has more than 500% gains in 2015

As shown in Figure 4, these stocks do not experience an incredible gain in 2015, which means that their Price variations in 2015 are wrong. In this way, we need to delete these stocks. As shown in Figure 5, these stocks' price variation in 2015 distribution becomes much more normal after stock deletion.

.

2.3 Handle Missing Values, 0-values

Next, we need to check the presence of missing values(NaN) and the number of 0-values in each indicator. Missing data is a common problem in statistical analysis. Rates of less than 1% missing data are generally considered trivial, 1-5% manageable. However, 5-15% require sophisticated methods to handle, and more than 15% may severely impact any kind of interpretation [2]. According to Acuna's statement, I make some bar charts for the counts of both missing values and 0-values to have a first look.

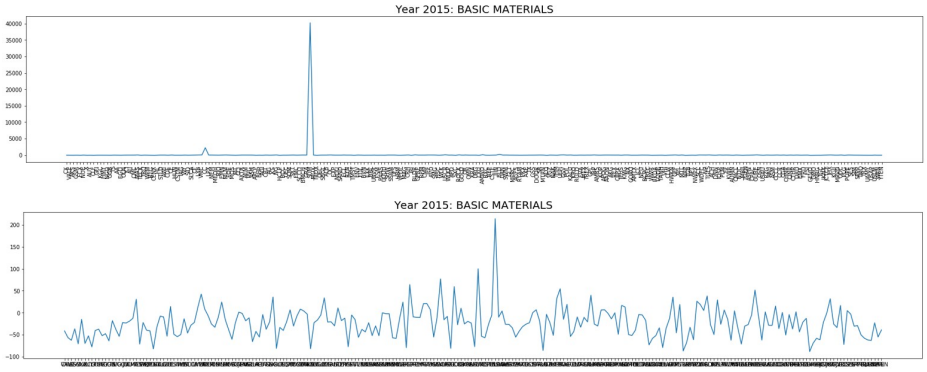

Fig. 5. Comparison of price variation distribution between stock deletion

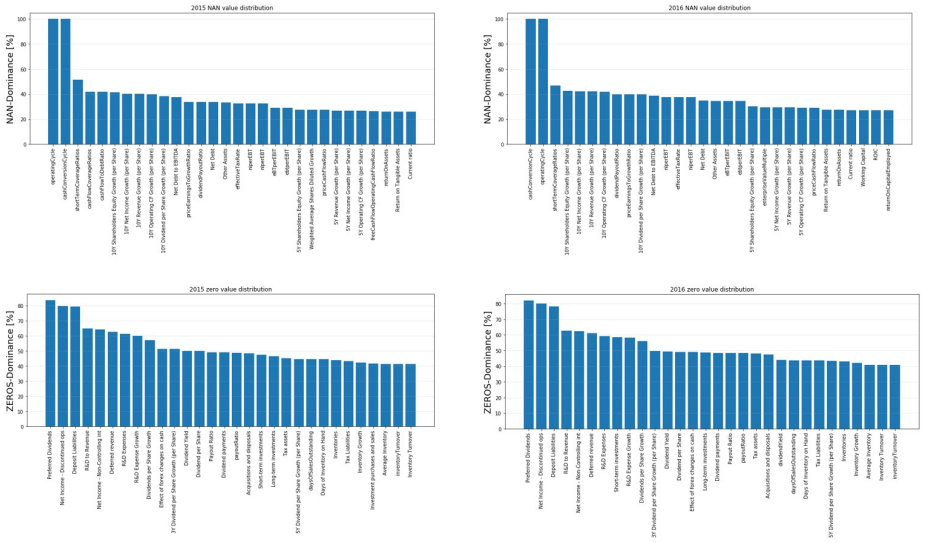

Fig. 6. Count of NaN/zero value in 2015&2016

As shown in Figure 6, There are quite a lot of missing values & zero-values. For some financial indicators, almost every value is 0. Because there are too many NAN values or zero values in some indicators, these indicators need to be dropped. There are two methods to handle NAN/zero-values: 1. Filling the missing data; 2. Fill or drop those indicators that are heavy zeros-dominant. However, I cannot choose only one of these two methods to handle missing values/zero values, because there is a trade-off of the drop threshold. The decision of dropping top N% NaN-dominant & zero-dominant financial indicators is not like the less, the better: If N is too high (like 98%), the remaining indicators are less than 10, which is not sufficient for indicator analysis; If N is too low, there are too many NAN value & zero value in the statistics. Even if we can fill the NaN value by 1. Fill NaN with 0; 2. Fill NaN with the mean value of the column; 3. Fill NaN with mode value of column; 4. Fill NaN with the previous value, the indicator analysis may not be accurate because of too many NaN value / zero value.

Table 1. Maximum NaN/zero-value percentile in different thresholds

Threshold	Maximum percentile
NaN-20%	13.4
NaN-30%	14.65
NaN-40%	16.41
Zero-40%	1.69
Zero-50%	4.39
Zero-60%	6.48
Zero-70%	16.82

In this way, the levels of NaN-dominance and zeros-dominance should be decided for tolerance. According to Acuna's paper, I determine a threshold level for both NaN-dominance and zeros-dominance. It means: If a column has a percentage of NaN-values and/or zero-valued entries higher than the threshold, drop it. Firstly, I drop all the columns and rows that have a percentage of NaN-values 100%. Table 1 represents the maximum NaN/zero-value percentile in different thresholds. According to my previous statement, to be consecutive, there is a conclusion that more than 15% NaN-value and zero-value may severely impact any kind of interpretation. After the threshold trade-off experiment, I finally decide to drop the top 70% NaN-dominant financial indicators and drop the top 40% zero-dominant financial indicators.

As shown in Figure 7, in each year, the NaN-dominance threshold is 4-15% and the zero-dominance threshold is between 6-8%, which is suitable, compared with Acuna's paper. As shown in Figure 8, there are still around 40 remaining indicators, which is large enough. For the indicators that still have NaN values, I fill them with that indicator's mean value.

```
Total NaN      156.000000
Percent NaN      4.096639
Name: 0.3, dtype: float64
Total Zeros      290.200000
Percent Zeros      7.620798
Name: 0.6, dtype: float64

Total NaN      250.900000
Percent NaN      6.089806
Name: 0.3, dtype: float64
Total Zeros      262.800000
Percent Zeros      6.378641
Name: 0.6, dtype: float64

Total NaN      671.000000
Percent NaN     13.987909
Name: 0.3, dtype: float64
Total Zeros      301.800000
Percent Zeros      6.291432
Name: 0.6, dtype: float64
```

```
Total NaN      726.70000
Percent NaN     14.65121
Name: 0.3, dtype: float64
Total Zeros      301.800000
Percent Zeros      6.084677
Name: 0.6, dtype: float64

Total NaN      207.200000
Percent NaN      4.717668
Name: 0.3, dtype: float64
Total Zeros      285.800000
Percent Zeros      6.507286
Name: 0.6, dtype: float64
```

Fig. 7. Decision threshold of NaN/zero value in year 2015-2019

```
INITIAL NUMBER OF VARIABLES: 224

NUMBER OF VARIABLES AFTER NaN THRESHOLD 4.10%: 68

NUMBER OF VARIABLES AFTER Zeros THRESHOLD 7.62%: 36

INITIAL NUMBER OF VARIABLES: 224

NUMBER OF VARIABLES AFTER NaN THRESHOLD 6.09%: 67

NUMBER OF VARIABLES AFTER Zeros THRESHOLD 6.38%: 39

INITIAL NUMBER OF VARIABLES: 224

NUMBER OF VARIABLES AFTER NaN THRESHOLD 13.99%: 68

NUMBER OF VARIABLES AFTER Zeros THRESHOLD 6.29%: 43

INITIAL NUMBER OF VARIABLES: 224

NUMBER OF VARIABLES AFTER NaN THRESHOLD 14.65%: 67

NUMBER OF VARIABLES AFTER Zeros THRESHOLD 6.08%: 44

INITIAL NUMBER OF VARIABLES: 224

NUMBER OF VARIABLES AFTER NaN THRESHOLD 4.72%: 67

NUMBER OF VARIABLES AFTER Zeros THRESHOLD 6.51%: 45
```

Fig. 8. Remaining indicators in year 2015-2019

2.4 Handle Extreme Value

Figure 9 is the indicator value distribution in 2015. Although for each indicator, I only record its mean, min, max, and standard deviation values, it is still hard to find other indicator's values. Therefore, I drop two most highest indicators ***Total assets*** and ***Total liabilities***, and figure 9 becomes figure 10. At this time, it is not difficult to find that there are still many extreme values in most indicators, and their standard deviation is very large.

By Analyzing the dataframe with the method ***describe()*** I can see that some financial indicators show a large discrepancy between max value and 75% quantile. Furthermore, we also have standard deviation values that are very large! This could be a sign of the presence of outliers: to be conservative, I will drop the top 5% and bottom 5% of the data for each financial indicator.

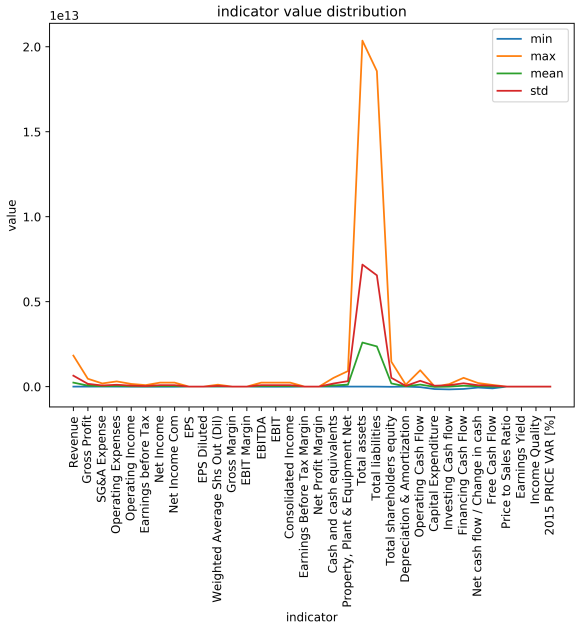

Fig. 9. Indicator value distribution in 2015

3 Indicator Visualization & Analytics

3.1 Indicator Learning

Given indicators, we may predict the price variation in that year. Moreover, it is also important to find which indicators contribute most to the price variation prediction. In this part, I use the XGBoost model to address the indicator learning problem. I choose all stock data from 2015 to 2019 as a source.

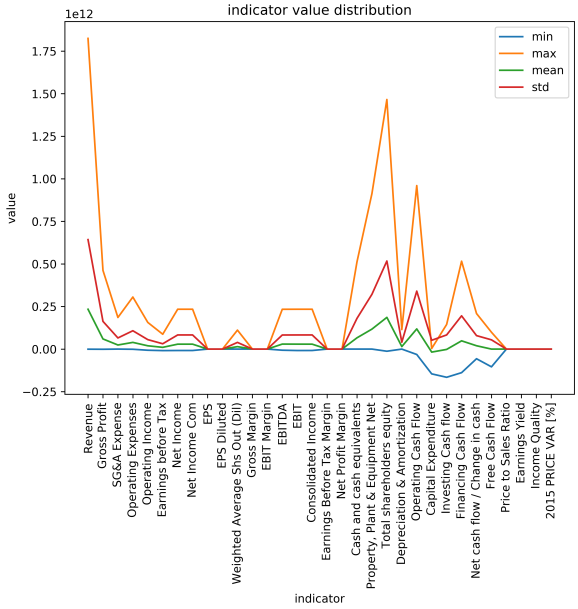

Fig. 10. Indicator value distribution in 2015 after modification

Data preprocessing Although there are already many operations in section 2 for data detection, data cleaning, and data filling, I still need some other operations to make stock data suitable for model training. As shown in section 2.4, there are many extreme values for me to handle. Here I choose to use ***IsolationForest*** for outlier detection and cleaning.

IsolationForest IsolationForest is an unsupervised learning algorithm for outlier detection that works on the principle of isolating anomalies, which returns the anomaly score of each sample using the IsolationForest algorithm. The IsolationForest "isolates" observations by randomly selecting a feature and then randomly selecting a split value between the maximum and minimum values of the selected feature. Since recursive partitioning can be represented by a tree structure, the number of splittings required to isolate a sample is equivalent to the path length from the root node to the terminating node. This path length, averaged over a forest of such random trees, is a measure of normality and our decision function. Random partitioning produces noticeably shorter paths for anomalies. Hence, when a forest of random trees collectively produce shorter path lengths for particular samples, they are highly likely to be anomalies. In addition to the IsolationForest algorithm, to be conservative, I drop the top 5% and bottom 5% of the data for each financial indicator, to make sure that there is no such extreme values.

Moreover, I use ***StandardScaler*** method to normalize all feature data. ***StandardScaler*** is a module in ***sklearn.preprocessing***, which standardizes

features by removing the mean and scaling to unit variance. Finally, before model training, to make a comparison of the ground truth and prediction result, I split the dataset to 70% training case and 30% testing case, for model evaluation.

Introduction of XGBoost As described in their documentation [1], XGBoost is an open-source software library that provides a gradient boosting framework for Python. It works on Windows. It aims to provide a "Scalable, Portable and Distributed Gradient Boosting (GBM, GBRT, GBDT) Library". I choose it because it not only provides many hyperparameters for me to create a good model, but also provides feature importance for each indicator who gives significant contribution. Some features make XGBoost different from other gradient boosting algorithms:

- A proportional shrinking of leaf nodes
- Newton Boosting
- Extra randomization parameter
- Implementation on single, distributed systems and out-of-core computation

Hyperparameter tuning Hyperparameter tuning, also known as hyperparameter optimization, is referred to as tuning within the machine learning community. Commonly recurring tuning parameters are related to regularization, kernels, learning rate. There are some important general observations for hyperparameter tuning [6]:

- The effect of tuning parameters can differ greatly
- Many tuning parameters have (box) constraints
- The importance of a parameter can vary per problem
- Typically only a subset of tuning parameters significantly affect the quality of the model

XGBoost model provides many parameters for us to do hyperparameter tuning. I would like to introduce some parameters:

- booster: Specify which booster to use: gbtree, gblinear or dart.
- objective (string or callable) – Specify the learning task and the corresponding learning objective or a custom objective function to be used.
- gamma (float) – Minimum loss reduction required to make a further partition on a leaf node of the tree.
- max_depth (int) – Maximum tree depth for base learners.
- min_child_weight (float) – Minimum sum of instance weight(hessian) needed in a child.
- max_delta_step (int) – Maximum delta step we allow each tree's weight estimation to be.
- subsample (float) – Subsample ratio of the training instance.
- colsample_bytree (float) – Subsample ratio of columns when constructing each tree.

– learning_rate (float) – Boosting learning rate (XGB's "eta")

To find every optimal parameter in a short time, our method is that we find some optimal parameters first because one parameter will not affect the other parameter's performance. We set a range for each parameter to run the model. If the optimal parameter is neither max value nor min value, this parameter will not change anymore; Otherwise, we need to expand the parameter range to find if there is any other optimal parameter.

Prediction result After hyperparameter tuning, we finally find the best model for each year. Table 2 is the training mean squared error, testing mean squared error and classification accuracy of prediction in each year, and Figure 11 is the comparison of the prediction value and the true value. Note that for each stock, if the prediction and the ground truth are all positive or negative, we think that this prediction is accurate; Otherwise, it is not accurate.

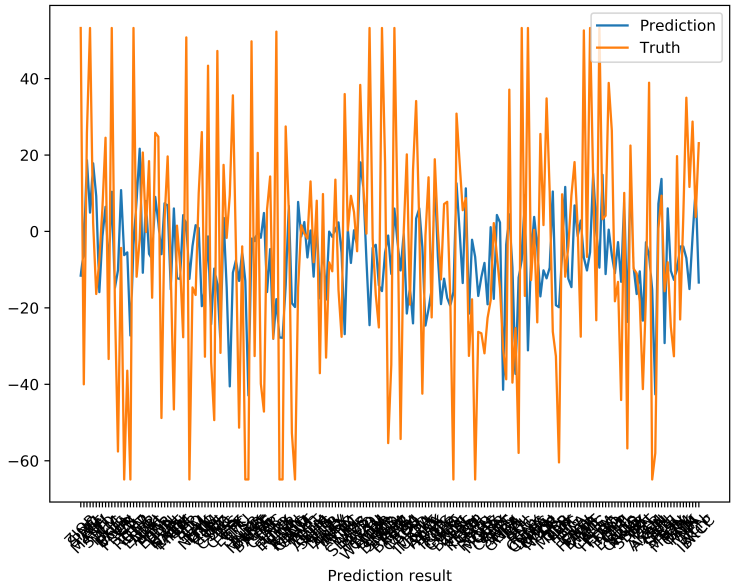

Fig. 11. Stock prediction in year 2015

As shown in table 2 and figure 11, the average prediction accuracy is around 70%, which is a relatively good result. However, although I have sampled the stock dataset to make a balanced dataset, I want to find that if I can get the same prediction for each sector. In this way, I divide the stock dataset by the sector in 2015 and use the same model to train them.

Table 3 is the training mean squared error, testing mean squared error, and classification accuracy of each sector in 2015. As shown in table 3, the accuracy

Table 2. Prediction MSE and accuracy in each year

Year	train MSE	test MSE	Accuracy
2015	522.78	911.88	0.644
2016	696.55	1181.39	0.763
2017	736.16	1223.58	0.701
2018	426.28	788.32	0.765
2019	822.49	1455.38	0.729

Table 3. Prediction MSE and accuracy in 2015

Sector	train MSE	test MSE	Accuracy
Consumer Defensive	702.54	1094.47	0.614
Basic Materials	602.48	710.17	0.797
Healthcare	1452.93	2040.98	0.624
Consumer Cyclical	539.30	667.22	0.589
Industrials	643.58	1093.1	0.716
Real Estate	253.44	456.74	0.778
Communication Services	671.08	698.85	0.652
Energy	529.51	783.79	0.873
Financial Services	248.54	291.69	0.704
Utilities	596.19	560.42	0.630
Industrials	797.31	869.12	0.607

change in each sector is not as stable as that in table 2. The highest accuracy, which is in sector ***Energy***, is nearly 90%, but the lowest accuracy is only around 60%. In this way, I can conclude that the accuracy distribution by sector is not as stable as that by year: Some sector is easy to make a prediction, but some are not.

Feature importance When training the stock dataset, it is meaningful to find what indicator contributes most to the price variation prediction. Fortunately, the XGBoost model provides a method to plot the F-score distributions of indicators in the stock dataset. F-score is the feature score in the XGBoost model. The higher the feature score, then this feature is more important.

Figures 12-16 are the F-scores of indicators in 2015-2019. As shown in these figures, their top 5 indicators are quite different. In this way, to make it more clear, I choose the top 5 indicators in each year and count them in Figure 17. As shown in figure 17, in around a half of sectors, indicator ***Operating Income Growth***, ***Operating Cash Flow growth***, ***Weighted Average Shs Out (Dil)*** and ***Gross Margin*** has a significant influence on the price variation change. However, it does not seem that the most popular indicator is the most important indicator. For example, in 2015, although indicator ***Earnings Yield*** is not so popular, it contributes most to the price variation prediction in this year.

However, In addition to considering the F-scores of indicators in all sectors, for each sector, will these indicators still have the most significant contributions?

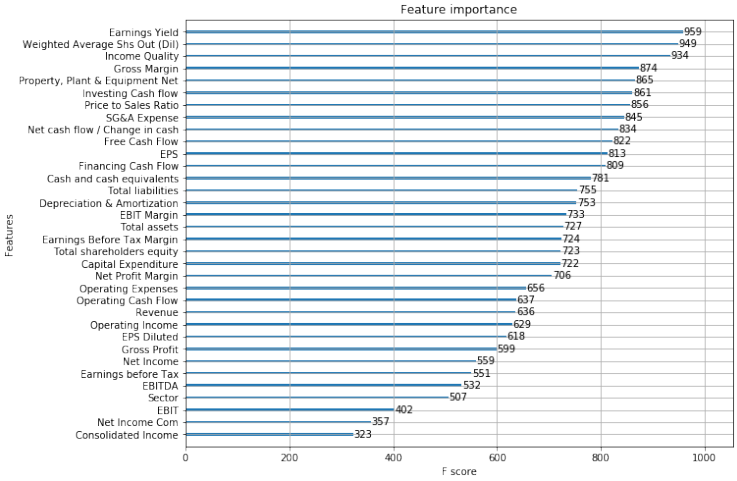

Fig. 12. F-scores of indicators in year 2015

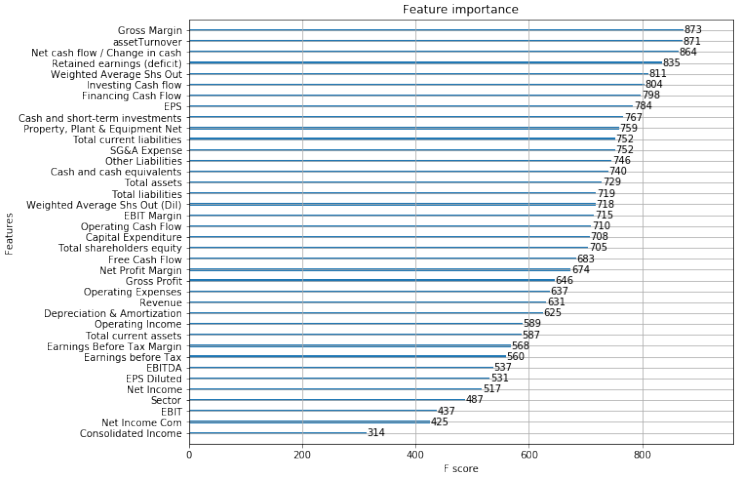

Fig. 13. F-scores of indicators in year 2016

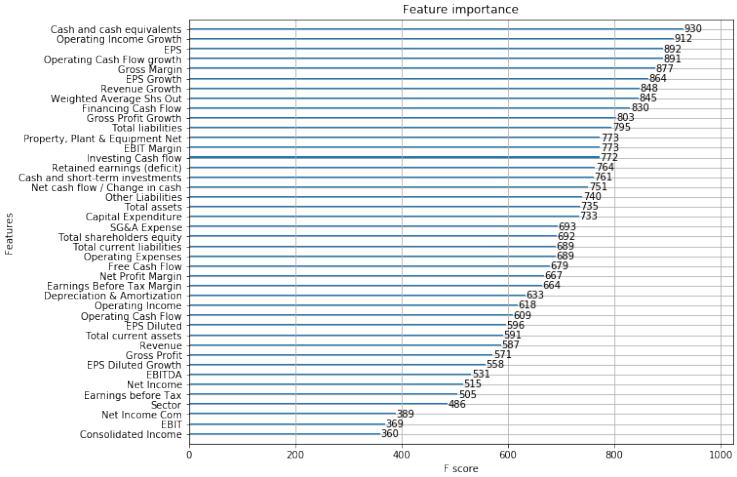

Fig. 14. F-scores of indicators in year 2017

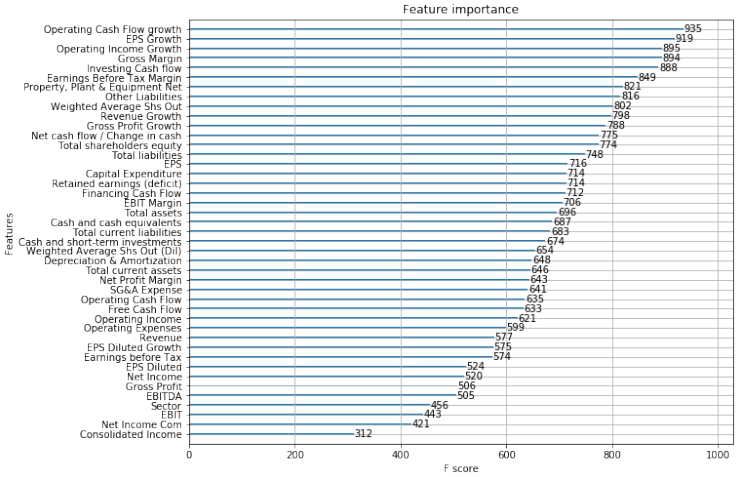

Fig. 15. F-scores of indicators in year 2018

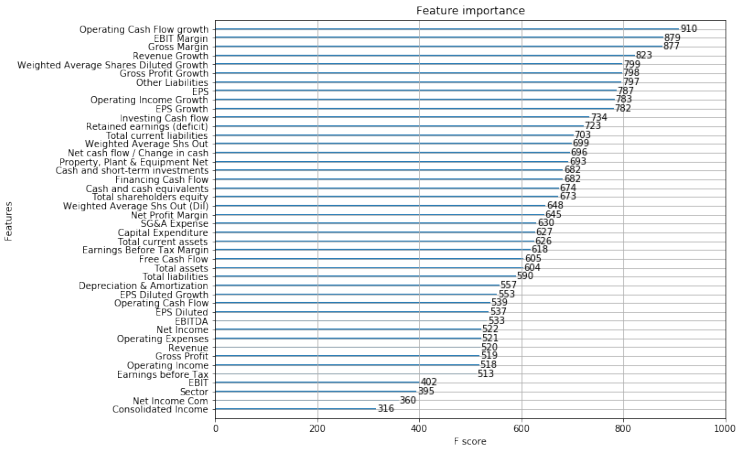

Fig. 16. F-scores of indicators in year 2019

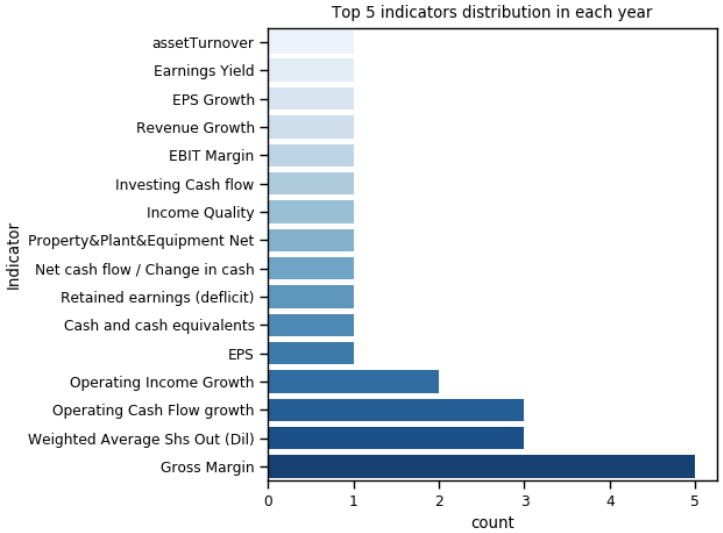

Fig. 17. Top 5 indicators distribution in each year

To address this problem, I sample the data by the sector name in 2015, and use the same model to train them, so as to find which indicators contribute most to the price variation.

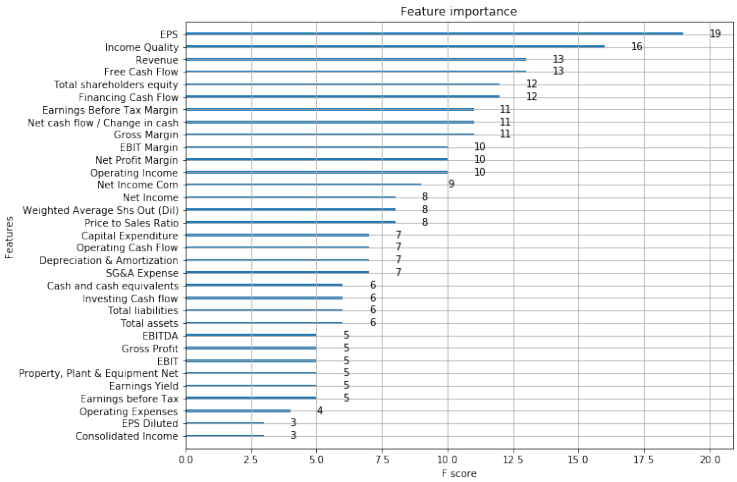

Fig. 18. F-scores of indicators in sector Consumer Defensive(year 2015)

Figures 18-28 are the F-scores of indicators for each sector in 2015. As shown in these figures, their top 5 indicators are quite different. In this way, to make it more clear, I choose the top 5 indicators in each sector and count them in Figure 29. As shown in figure 29, in around a half of sectors, indicators ***Price to Sales Ratio***, ***Income Quality*** and ***Net cash flow/Change in cash*** have a significant influence on the price variation prediction. However, it does not seem that the most popular indicator is the most important indicator. For example, in the sector ***Communication Services***, although indicators ***EBIT Margin*** and ***Earnings Yield*** are not so popular, they contribute most to the price variation in this sector.

Compared with the conclusion in the year's summary, I can say that although one indicator is popular in nearly a half of sectors to contribute to the price variation prediction, such as the ***Financing Cash Flow***, it may not contribute much to the whole sector's price variation prediction.

3.2 Indicator correlation

After finding what indicators contribute most to the price variation prediction, I need to find what kinds of influence the indicator has on the price variation. As Ross [9] discusses in the paper, in statistics, correlation is the scaled version of covariance, which measures whether variables are positively or inversely related. Correlation is a very important concept in technical stock market analysis, as

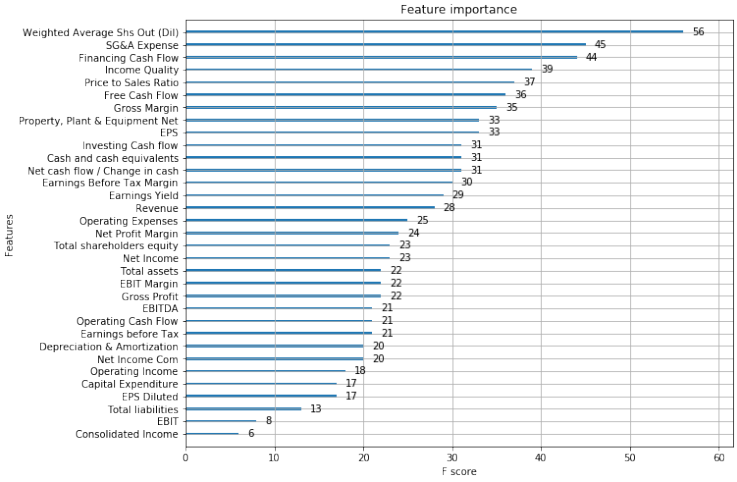

Fig. 19. F-scores of indicators in sector Basic Materials(year 2015)

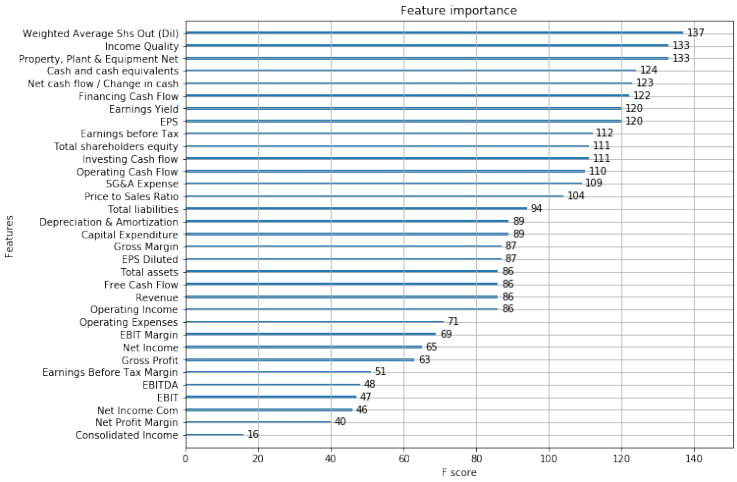

Fig. 20. F-scores of indicators in sector Healthcare(year 2015)

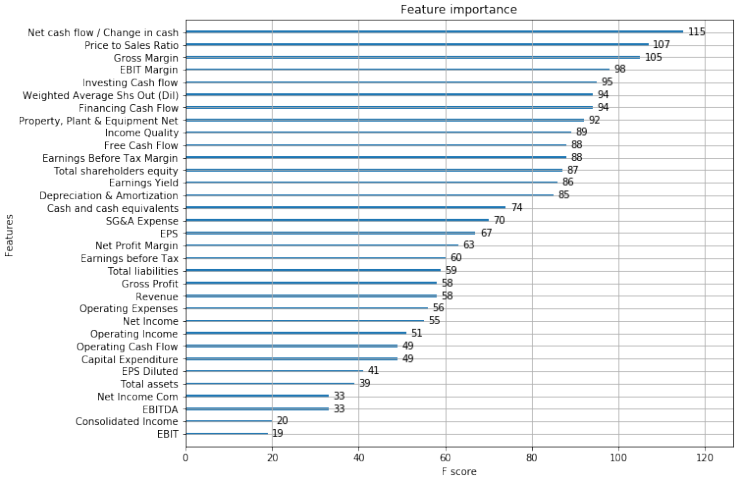

Fig. 21. F-scores of indicators in sector Consumer Cyclical(year 2015)

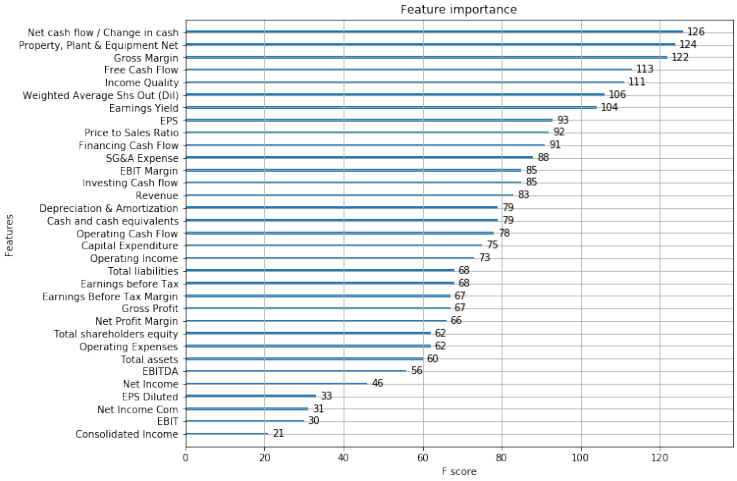

Fig. 22. F-scores of indicators in sector Industrials(year 2015)

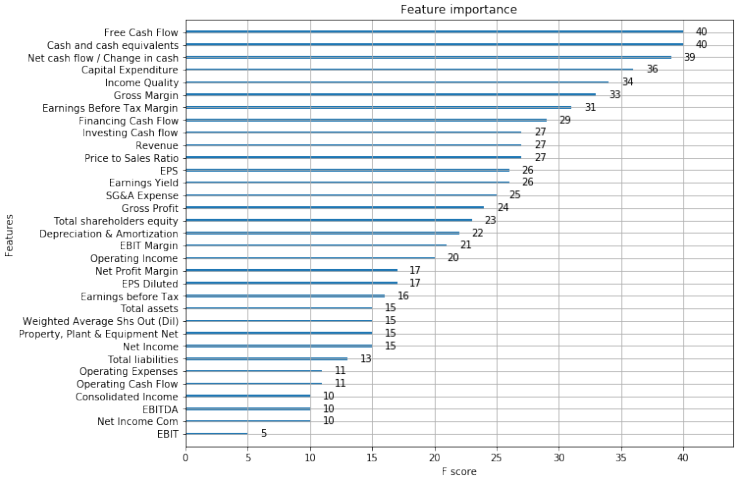

Fig. 23. F-scores of indicators in sector Real Estate(year 2015)

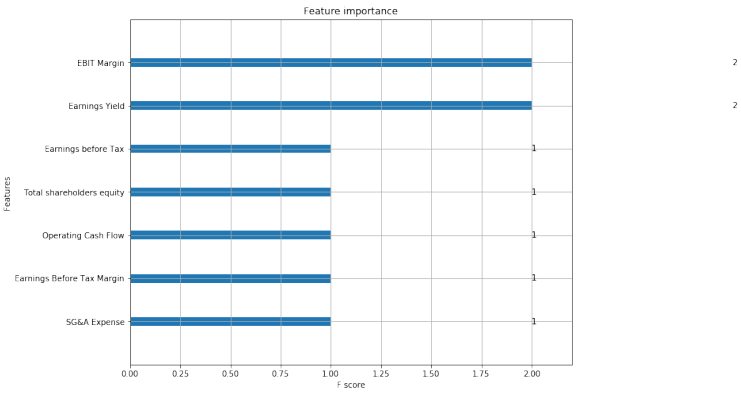

Fig. 24. F-scores of indicators in sector Communication Services(year 2015)

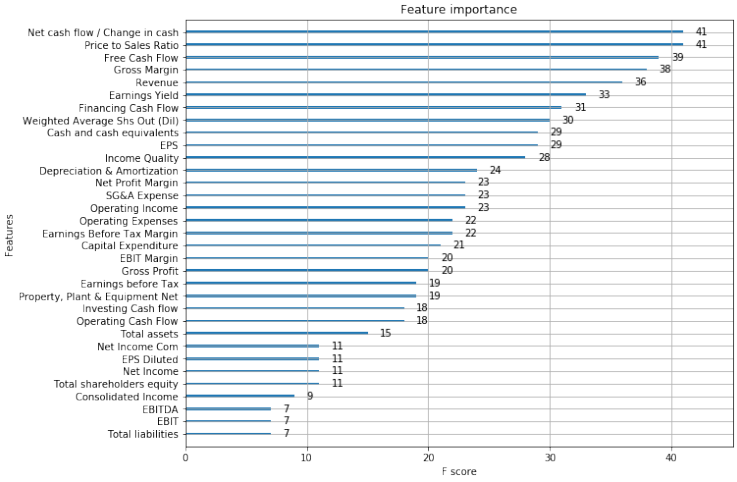

Fig. 25. F-scores of indicators in sector Energy(year 2015)

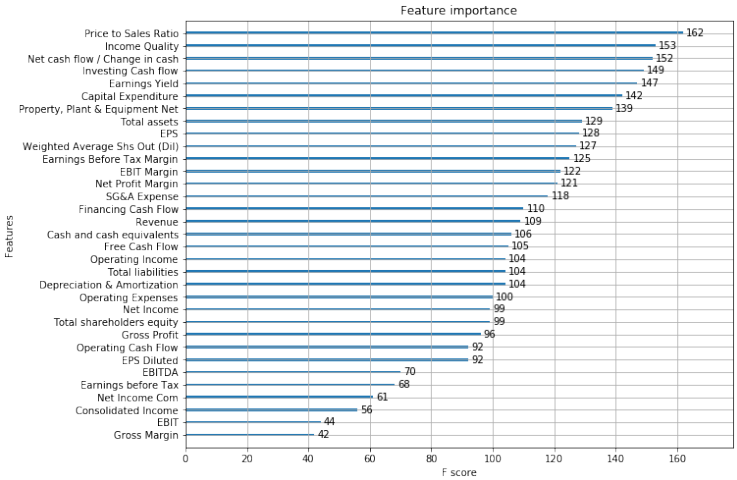

Fig. 26. F-scores of indicators in sector Financial Services(year 2015)

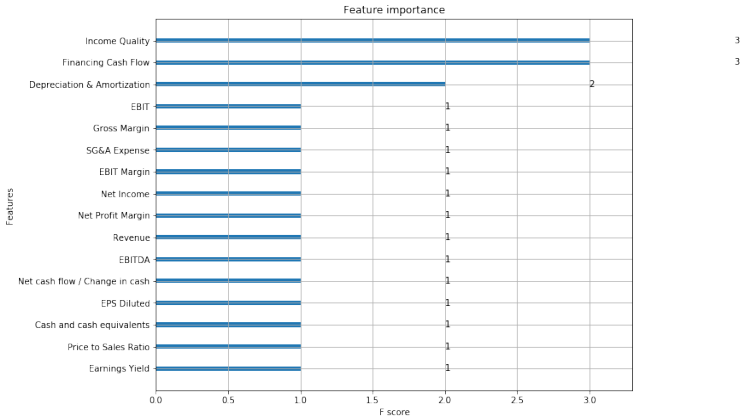

Fig. 27. F-scores of indicators in sector Utilities(year 2015)

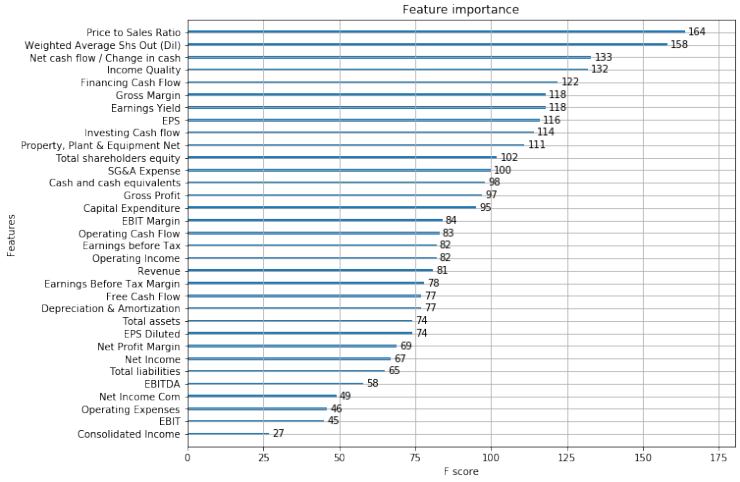

Fig. 28. F-scores of indicators in sector Technology(year 2015)

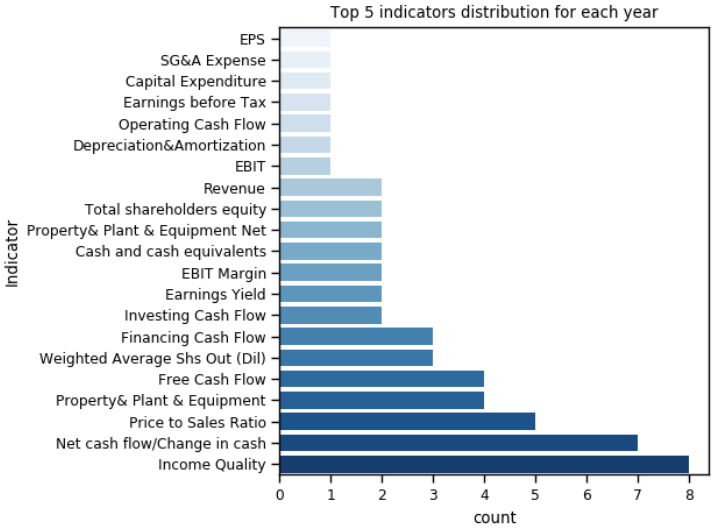

Fig. 29. Top 5 indicators distribution in all sectors(year 2015)

it makes it possible to guess at the mechanics of price patterns. The indicator correlation calculation is shown in figure 30.

$$\text{Correlation Coefficient} = \frac{COV}{SDMI \times SDSP}$$

where:

COV = Market indicator, stock price

$SDMI$ = Standard deviation for market indicator

$SDSP$ = Standard deviation for stock price

Fig. 30. Indicator correlation formula

In this project, there are three correlations: positive, negative, and no influence. In this way, it is important to set a threshold that determines whether one indicator has an influence on the price variation or not. Here, according to Ratner[8], Values between -0.3 and 0.3 indicate a weak positive (negative) linear relationship through a shaky linear rule. However, if I set the absolute correlation threshold to 0.3, there will be too many indicators in some sectors that have no influence, and I cannot even draw the correlation heatmap in 2017. Therefore, to be more consecutive, I set that if the absolute correlation t is less or equal to 0.15, this indicator is assumed to have no significant influence on the price variation. Moreover, it is possible that no indicator has a significant influence on

the price variation of one sector. In this case, this sector will not be represented in the correlation figure.

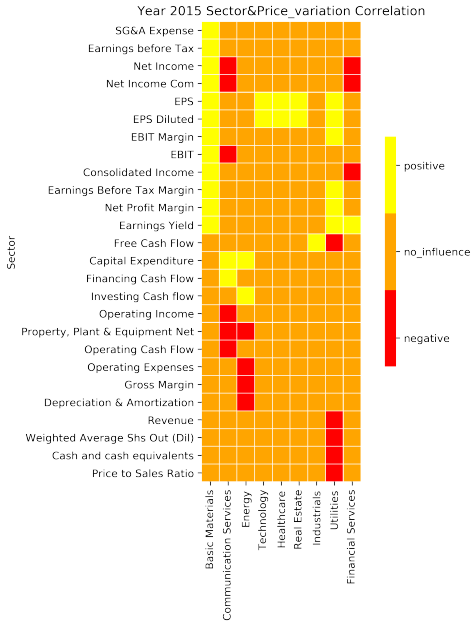

Fig. 31. Indicator correlation in year 2015

Figures 31-35 are the indicator correlations in 2015-2019. Here, I want to make an analytics sample for the sector ***Utilities***: According to the below correlation heatmaps, the indicator ***EPS*** has a positive significant impact on this sector in 2015, and it has no significant impact to this sector in 2016, 2017, 2018 and 2019. In general, I have confidence to say that the indicator ***EPS*** has a positive correlation with the sector ***Utilities***. Also, according to this indicator's correlation with every other sector, it is not difficult to find that this indicator never gives a negative influence on the stock price variation.

However, whether the indicator's correlation is significant does not imply whether it is important for making contributions to the price variation prediction. For example, in figure 19 and figure 31, for sector Basic Materials, ***Weighted Average Shs Out (Dil)*** and ***SG&A Expense*** are important for making contributions to price variation prediction, but in correlation heatmap, ***Weighted Average Shs Out (Dil)*** seems to have no influence on this sector's price variation. In this way, we can conclude that the relationship is not as simple as a correlation heatmap or feature importance distribution chart. I need to consider both of them.

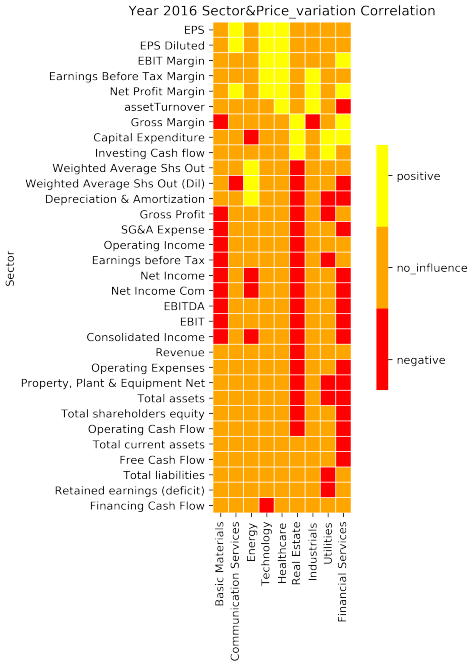

Fig. 32. Indicator correlation in year 2016

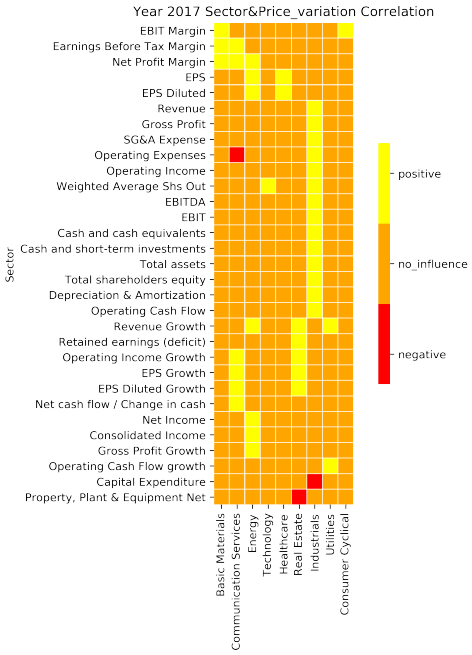

Fig. 33. Indicator correlation in year 2017

4 Conclusion

This project makes an indicator analysis and indicator learning on financial indicators in US stocks. During the data preprocessing part, extreme value, missing value, wrong value, and imbalanced data are handled carefully. In the indicator learning part, I use the XGBoost model to train these indicators, to predict the price variation. In some sectors, its testing MSE can be lower than 300 and has a nearly 90% classification accuracy. During the indicator learning, we also find that the feature importance to each sector and each year is quite different, and feature popularity does not imply the feature importance. In the indication correlation part, I find the correlation between price and variation (positive, negative, no influence) in each year and each sector, which can help make conclusions that which indicator usually has what influence on which sector. Also, whether the indicator's correlation has significant influence does not imply whether it is important for making contributions to the price variation prediction.

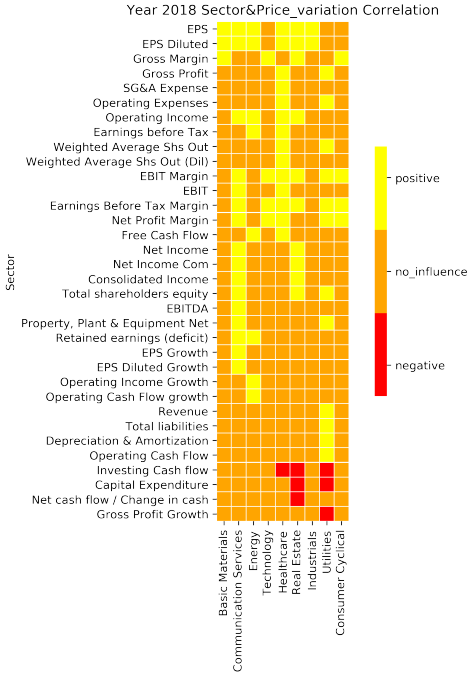

Fig. 34. Indicator correlation in year 2018

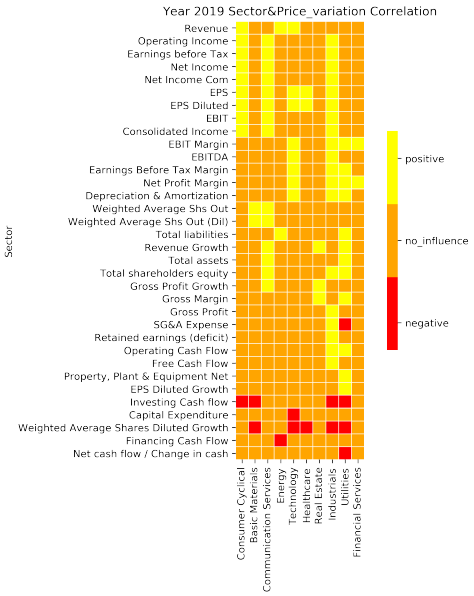

Fig. 35. Indicator correlation in year 2019

References

1. Xgboost documentation¶, https://xgboost.readthedocs.io/en/latest/
2. Acuna, E., Rodriguez, C.: The treatment of missing values and its effect on classifier accuracy. In: Classification, clustering, and data mining applications, pp. 639–647. Springer (2004)
3. Anand, R., Mehrotra, K.G., Mohan, C.K., Ranka, S.: An improved algorithm for neural network classification of imbalanced training sets. IEEE Transactions on Neural Networks **4**(6), 962–969 (1993)
4. Bloomenthal, A.: Gross margin defined (Nov 2020), https://www.investopedia.com/terms/g/grossmargin.asp
5. Carbone, N.: 200+ financial indicators of us stocks (2014-2018)
6. Claesen, M., Simm, J., Popovic, D., Moor, B.: Hyperparameter tuning in python using optunity. In: Proceedings of the International Workshop on Technical Computing for Machine Learning and Mathematical Engineering. vol. 1, p. 3 (2014)
7. Krawczyk, B.: Learning from imbalanced data: open challenges and future directions. Progress in Artificial Intelligence **5**(4), 221–232 (2016)
8. Ratner, B.: The correlation coefficient: Its values range between+ 1/- 1, or do they? Journal of targeting, measurement and analysis for marketing **17**(2), 139–142 (2009)
9. Ross, S.: How do i calculate correlation between market indicators and specific stocks? (Sep 2020), https://www.investopedia.com/ask/answers/032315/how-do-i-calculate-correlation-between-market-indicators-and-specific-stocks.asp
10. Staff, R., Malika, Susana, Malika, Rao, V.: How to use the yahoo finance api (in 2020) [tutorial]: Rapidapi (Sep 2020), https://rapidapi.com/blog/how-to-use-the-yahoo-finance-api/