Machine Learning Engineer Nanodegree

Capstone Project

Nobuyoshi Shimmen

July 11st, 2017

I. Definition

Project Overview

There are mainly two ways to evaluate stock market: technical analysis and fundamental analysis. Technical analysis assumes that a stock price reflects all the information that decides the price available to public and predicts how a stock moves by statically looking at the history of the price movements. Fundamental analysis, on the other hand, tries to see through intrinsic value of a company by analyzing corporate financial statements, like balance sheet and income statement.

They are used to see if a stock is undervalued or not. Technical analysis is said to be effective for predicting short-term trading trends because there is less uncertain elements than in the long term future. Fundamental analysis show its value in long term stock investment because you need time to wait before a company's true value is recognized in the market.

Hedge funds and institutional investors have been focusing on technical analysis because of the nature of the business they are in. In short period of time, they have to leverage their customers' money and make some profit for the customers. Also they do this for themselves because their salaries are greatly affected by how much profit they make. So they have been putting effort and money into applying machine learning to technical analysis for a long time.

Retail investors prefer fundamental analysis to technical analysis, because they simply cannot afford so much transaction cost for buying and selling stocks that technical analysis requires to make capital gains in short period time. Also, fundamental analysis has been proven effective and recommended by many famous investors like Warren Buffet. Interestingly and sadly, however, even though fundamental analysis have been seen as practical and helpful to individual investors, little research hasn't been done for applying machine learning to fundamental analysis.

Therefore, in this project, my focus is on fundamental analysis and I would like to see how machine learning can predict winning stocks in the long run by using fundamentals of companies.

Problem Statement

Fundamental analysis is suited for individual investors by nature but almost no one has tried to apply machine learning to this. My goal is to make a model that can pick winning stocks in the long run and outperform the stock market. In other words, a model would make a portfolio whose growth rate is higher than the market. Also, what I mean by the long run is one year. To be more clearly, I have a model predict the prices of stocks on December 30 or 31st of a year, using corporate fundamentals in December or January of the previous year, and corporate fundamentals are lie Net Income, Operating Income, and Earnings Per Share.

Metrics

In this project, there are two types of evaluations. One is used for evaluating the performance of a model, so that I can choose the best one.

I used coefficient of determination, which is often called R2. This metric is useful in regression analysis, because it describes how good a model is at making predictions. The maximum score of it is 1 and the minimum score of it is 0 and the greater the r2 score is, the better the model is fitting.

$$R^2 \equiv 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}}.$$

The other metric is for comparing the performance of a model to benchmark. I used compound annual growth rate (CAGR) for this.

Many stock investors and analysts use GAGR to evaluate the performance of investments. The definition is as follows.

$$\text{CAGR} = \left(\frac{\text{Ending Value}}{\text{Beginning Value}}\right)^{\left(\frac{1}{\#\text{ of years}}\right)} - 1$$

I had the best model predict prices of all the stocks in one year. Then I calculated the CAGR of 30 stocks that would get the highest growth in price, using the sum of original stock prices as the beginning value and the sum of the next year prices* as the ending value. Finally I compared the result to the CAGR of return on the market.

*These are not what are predicted by the model, but are the actual prices of the following year.

II. Analysis

Data Exploration & Exploratory Visualization

I prepared two kinds of datasets to make a model; one is fundamentals and the other is historical stock prices.

These are obtained from Kaggle (https://www.kaggle.com/dgawlik/nyse): fundamentals.csv and prices-split-adjusted.csv.

fundamentals.csv(modified-fundamentals.csv)

It contains corporate financial statements of 2012-2015.

I removed some of the columns based on this book (https://www.amazon.com/Investing-Dummies-Fifth-Eric-Tyson/dp/B0031569MO), and change the file name to modified-fundamentals.csv; I selected 14 important items that are used for fundamental analysis. For example, they are like...

- Ticker Symbol: This is an abbreviation used to identify a particular stock on a stock market
- Gross Profit: This is the profit a company makes after subtracting the costs associated with making and selling its products/services.
- Operating Income: This is the profit earned from a company's business operations
- Research and Development:
- Earnings Per Share: This is the amount of a company's profit allocated to each share of stock.

prices-split-adjusted.csv

It contains daily prices from 2010 to the end 2016, and the columns are like...

- symbol: This is an abbreviation used to identify a particular stock on a stock market
- date: This is a date of each price is set
- open: This is a price of stock when the market opens
- close: This is a price of stock when the market closes

I only used symol, date, and close columns. To be more specific, I used them on December 30/31 of each year.

I used things in modified-fundamentals.csv as features and the stock price of December 31 or 30 of the following year as label.

For example, when features are the fundamental data of a company in 2012, a label is the close price of the stock on December 31 in 2013.

I explored them as follows.

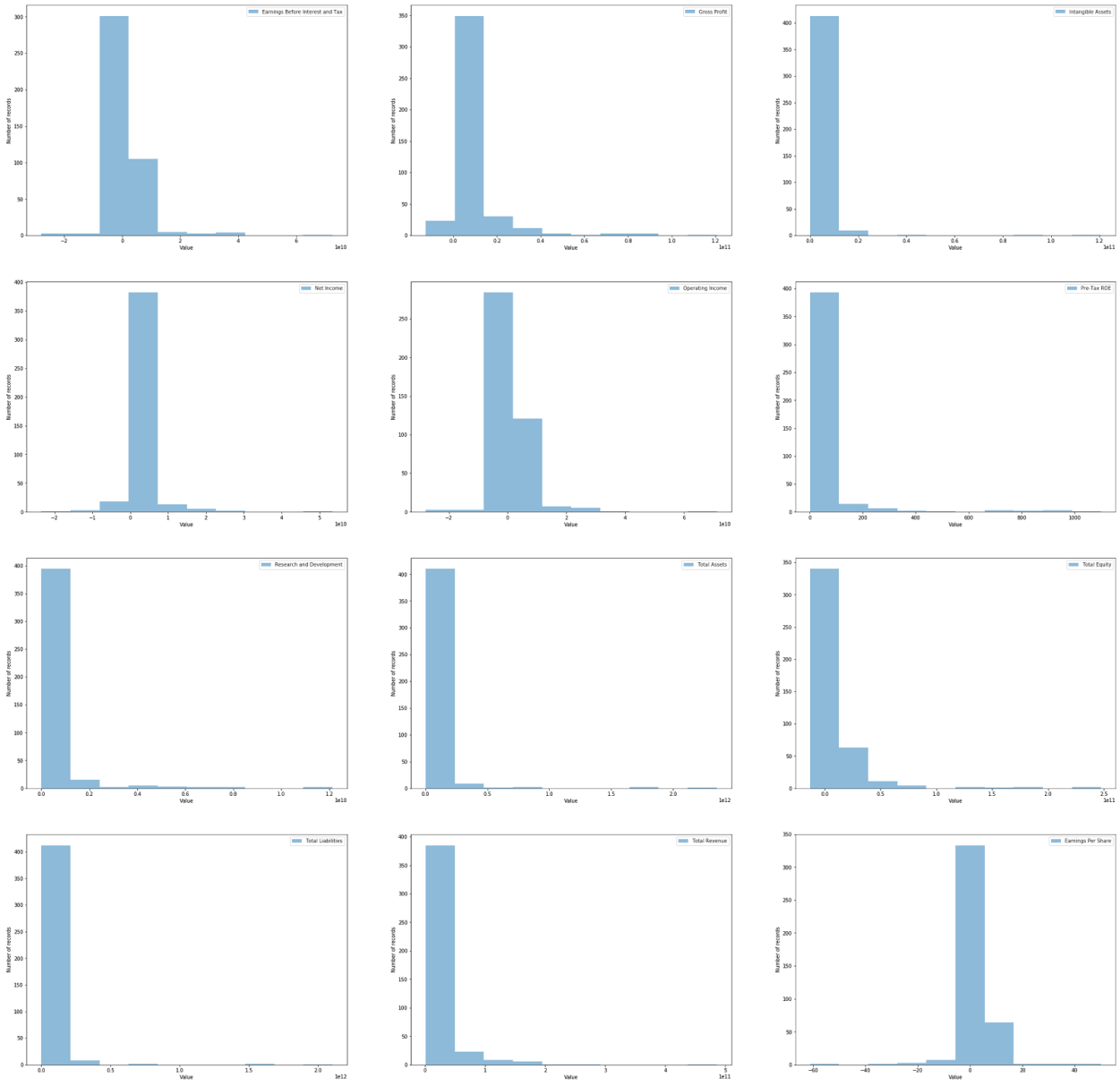
1. Get descriptive statistics of features for one random year
    2. - Min value, Max value, Mean value, Median value and Standard deviation of values
       - See if there is any feature that has highly-skewed distribution

3. Get descriptive statistics of labels for one random year
    4. - Min value, Max value, Mean value, Median value and Standard deviation of values

fundamental dataset has 1781 data points with 14 variables each.
split price dataset has 5339 data points with 3 variables each.

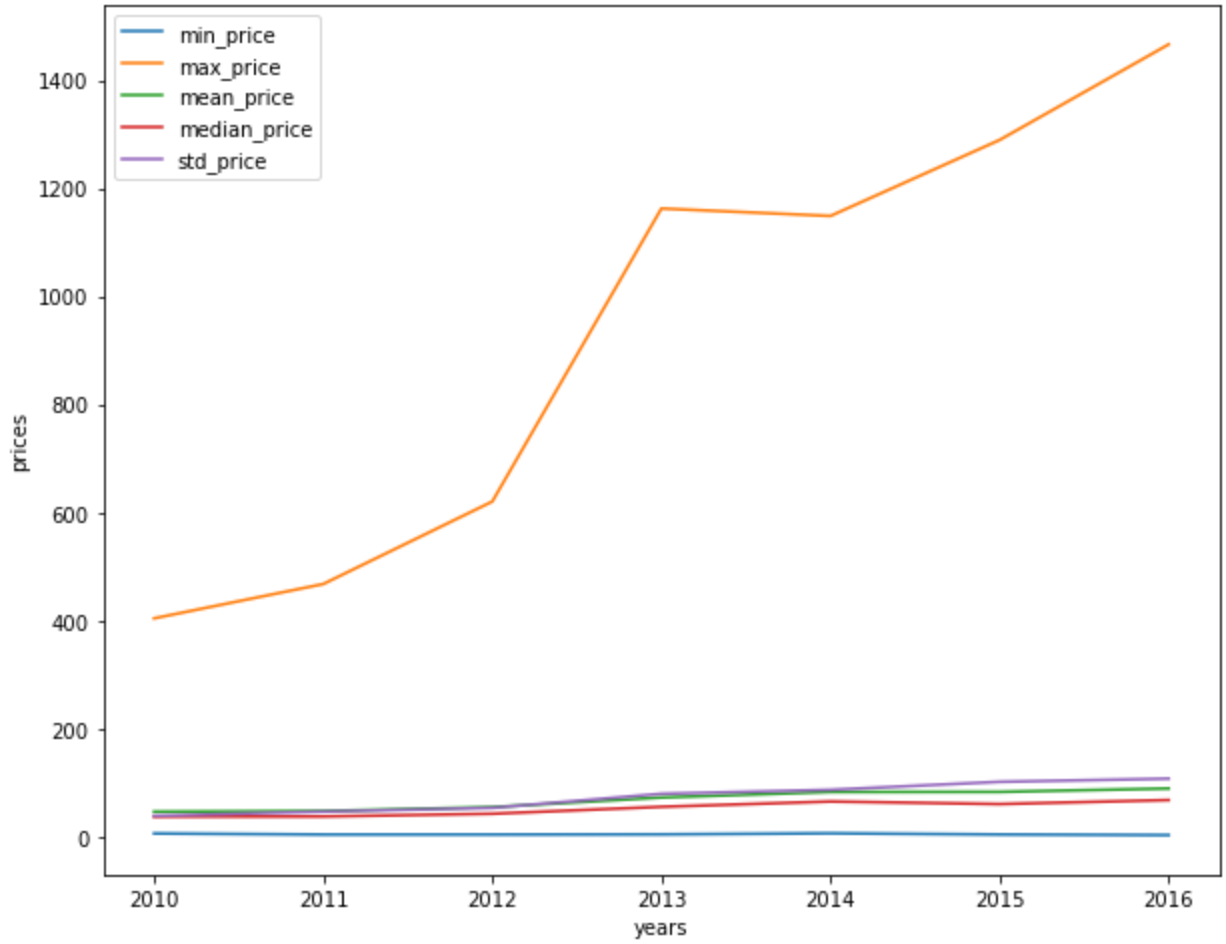
	Earnings Before Interest and Tax	Gross Profit	Intangible Assets	Net Income	Operating Income	Pre-Tax ROE	Research and Development	Total Assets	Total Equity	Total Liabilities	Total Revenue	For Year	Earnings Per Share	
count	4.250000e+02	4.250000e+02	4.250000e+02	4.250000e+02	4.250000e+02	425.00000	4.250000e+02	4.250000e+02	4.250000e+02	4.250000e+02	4.250000e+02	425.0	412.000000	
mean	2.457597e+09	7.384579e+09	2.209566e+09	1.575186e+09	2.116172e+09	54.249412	3.655569e+08	5.368703e+10	1.162822e+10	4.204838e+10	2.016654e+10	2015.0	3.019854	
std	6.344137e+09	1.351790e+10	8.175182e+09	4.409270e+09	5.763406e+09	130.080740	1.297193e+09	1.816241e+11	2.503089e+10	1.609421e+11	3.898290e+10	0.0	6.257429	

min	-2.792700e+10	-1.264700e+10	0.000000e+00	-2.352800e+10	-2.791000e+10	1.000000	0.000000e+00	1.474993e+09	-1.324400e+10	3.061520e+08	7.440120e+08	2015.0	-61.200000
25%	5.919080e+08	1.658870e+09	0.000000e+00	3.577960e+08	5.245270e+08	13.000000	0.000000e+00	7.278000e+09	2.349300e+09	4.572600e+09	3.956000e+09	2015.0	1.597500
50%	1.128286e+09	3.109281e+09	3.793050e+08	6.929570e+08	1.035000e+09	23.000000	0.000000e+00	1.590120e+10	5.104287e+09	1.013200e+10	8.592000e+09	2015.0	2.935000
75%	2.559000e+09	6.676000e+09	1.687326e+09	1.623600e+09	2.197000e+09	40.000000	6.371800e+07	3.887740e+10	1.061700e+10	2.453190e+10	1.748600e+10	2015.0	4.787500
max	7.251500e+10	1.205650e+11	1.207100e+11	5.339400e+10	7.123000e+10	1100.000000	1.212800e+10	2.351698e+12	2.475730e+11	2.104125e+12	4.856510e+11	2015.0	50.090000



From looking at the histograms, many features were positive-skewed, and I had to transform those values later.

As for outliers, I thought there were none, because those hight values, some may say call outliers in the features above, could be important characsteristics that decide whether it is a winning stock or not.



Mean, median, are gradually increaseing, which means the value of stock market as a whole is increasing. This could be either caused by some of the big companies that get super high values or a number of small companies increase their values little bit for each.
The company which has the highest price of stock was Priceline Group Inc(PCLN), and this was the case for all the years. Althought it was interesting that a company keeps the highest price of a stock for several years, I would save it for my future study because this was not super related to the focus of this project.

Algorithms and Techniques

I make three models and each model uses a different algorithms. Here're the algorithms to implement with the reasons I chose them.

- **Linear Regression**
  - characteristics
    - It's the most common and basic algorithm when it comes to regression problem.
    - It can be combined with neural network.

        - how it works
            - It assumes that the data follows a straight line, so it models the relationship between a label and feature, using a linear function. It creates a function that takes features as arguments and calculate a value, and compare it with a label. Then it measures the difference(error) between the predicted value and the true value. It does this for each data point, and minimize the errors by optimizing the function.
- **Decision Tree**
    - characteristics
        - It is easy to understand and to interpret how a result is decided.
        - It is robust to change and outliers.
        - It performs well with small amount of data points.
    - how it works
        - It creates a non-linear regression model between labels and features. The data is split at several split points for each feature. The error between the predicted value and the true values is calculated at each split point. The split point errors across the features are compared and the feature having the lowest error is chosen as the root split point. This process is recursively continued.
- **Deep Neural Network**
    - characteristics
        - It is a feedforward network with many hidden layers and modelled on the human brain (neural networks).
        - Even though the logic behind the result it produces is almost impossible to figure out, people say it works well.
    - how it works
        - Deep neural network is a feedforward network. It has three kinds of layers: input, hidden, and output. The layers are made of nodes. In hidden layers, a node combines a set of inputs from the given data with a set of weights. The sum of input-weight products is passed through a node's activation function, which decides whether and how much that signal goes further through the network to affect the final outcome. This computation happens at each node in hidden layers, so that it can model a complex non-linear relationship between features and labels. Also, it automatically updates the weights applied to the computation, according to the final outcome.

I use along the following techniques with these algorithms.

- **Hold-out validation**

    - It is a technique to see how well a model predict unseen data by splitting an original sample dataset into a training set to train the model, and a test set to evaluate.
- **Grid Search**
    - It is a technique to find parameters that perform best for a model. It can work with Cross Validation.

Benchmark

The benchmark is the average growth rate of return in stock market in general. This is because fundamental analysis is not a method but a concept of using fundamentals for evaluating stocks. The method of fundamental analysis varies depending on the person you ask. Therefore, there is no one best way to do that and I used the average returns of stock investment.

I based it on Standard & Poor's 500 Index, which has been seen as a reflection of the performance of American stocks in general. The data is obtained from this website (http://people.stern.nyu.edu/adamodar/New_Home_Page/datafile/histretSP.html). The owner of this website keeps track of CAGR of Standard & Poor's 500 Index for each year.

Year	CAGR
2012	15.89%
2013	32.15%
2014	13.52%
2015	1.38%
2016	11.74%

III. Methodology

Data Preprocessing

I processed the dataset as follows.

- Removed unnecessary rows

    - Feature dataset, which is fundamental data is from 2012 to 2016. Label dataset, which is stock price data is from 2010 to 2016. As you can see, the duration of the datasets are different. I dropped some of them so that they can match each other.
- Joined feature dataset and label dataset using (symbol/Ticker Symbol) If either label or features cannot be found with a Ticker Symbol drop them.
- Normalized features by using sklearn.preprocessing.MinMaxScaler
    - This is because the values of the feature dataset are too big to deal with.

Implementation

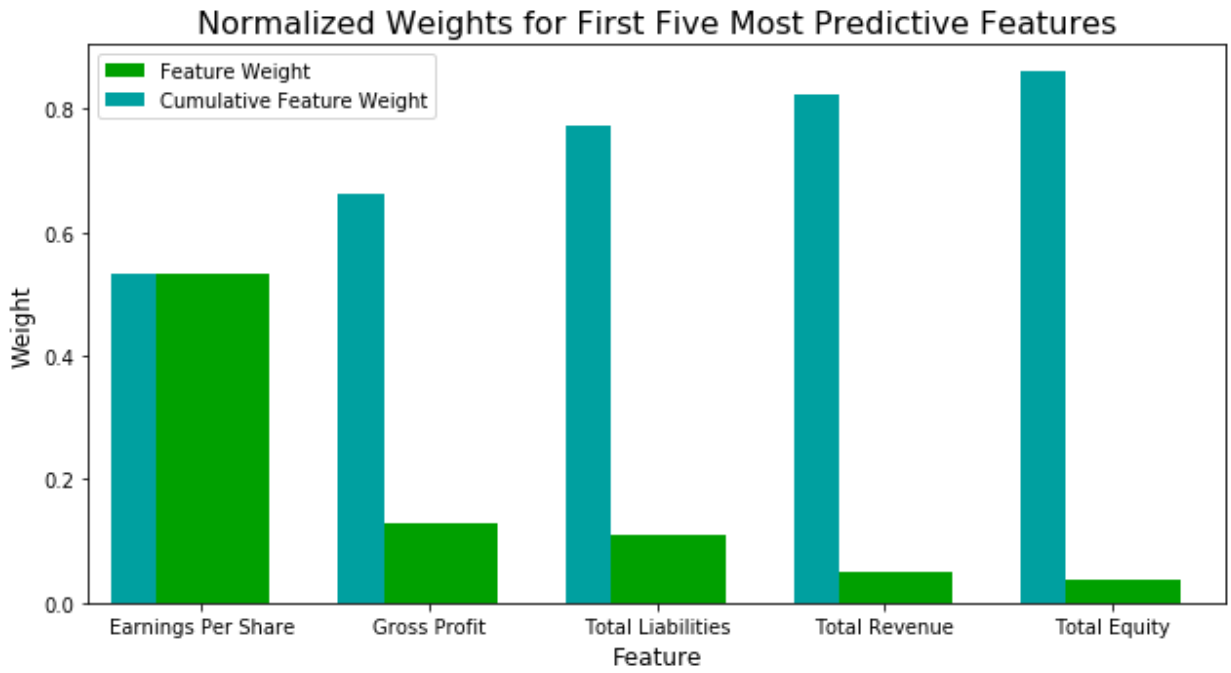
Define a Performance Metric

As I mentioned in Metrics section, I used coefficient of determination to measure the performance of the models.

Train and test a model: Decision Tree

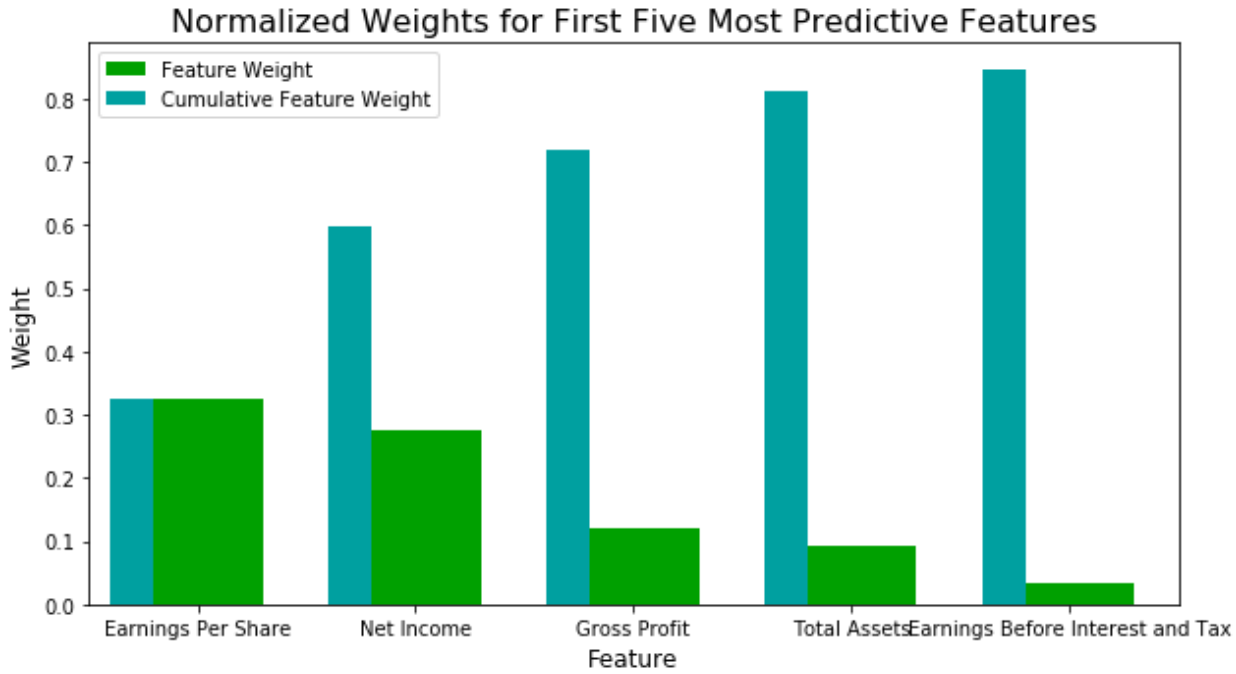
Between the three model, I first tried Decision Tree. As for splitting the dataset into train and test one, I was thinking of using TimeSeriesSplit from sklearn.model_selection. I realized, however, if I just predict the price of a stock in next year based on corporate fundamentals, I don't need to use it. That's because there is no correlation between observations that are near in time (autocorrelation).

R2 was 0.61



As you see the graph above, according to the decision tree model, Earnings Per Share is the most decisive factor of a stock price in next year.
Many people are probably more comfortable buying the stock of a profitable company. The higher the Earnings Per Share gets, the more people buy the stock.
I got curious what would happen if I expand the span; I used two years later stock prices as labels.

R2 is 0.53

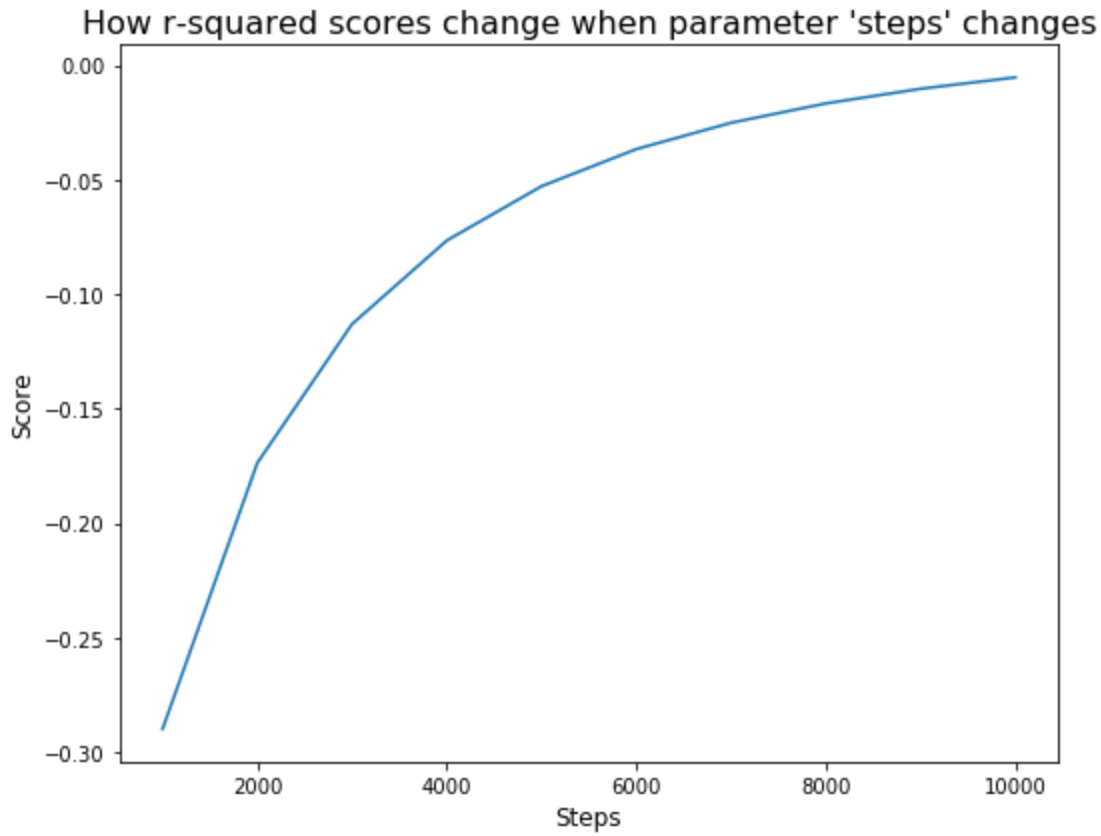


The r-squared score wend down by about 1%. This simply means it is harder to predict the price of a stock in the long run.
Also Net Income come in next to Earnings Per Share's weight, which made me think people prefer to buying even bigger and more profitable company's stock in two years.

Train and test a model: Linear Regression

Next I tried Linear Regression. I experimented with steps parameter which tells tensorflow to run iterations to train a model. The r-squared score was negative even when 10,000 steps(iterations) applied and from the graph of 'how r-squared score changes', I didn't think it would get better even if I increased the steps.

R2 was -0.01



Train and test a model: Deep Neural Network

Finally, I tried Deep Neural Network. The type of deep neural network I tried was called DNNRegressor. As the name implies, it construct a neural network regressions model. It takes a set of features and output the predictions. Then compare them to the labels(true value) to measure the error and tries to minimize it by updating the weights applied to inputs in hidden layers. This process is repeated to get a better result.

After more than 12 hours of the model being learning in my laptop, the r-squred score was only 0.14. I could've kept going, but I didn't because I couldn't imagine how much time would be required to get to a similar result to the one achieved by Decision Tree Regressor.

R2 was 0.14

Refinement

Between the models, in terms of the scores, and the learning time of the models, I decided to pick the decision tree regressor as the best one and to improve the score by optimizing parameters for the regressor with the grid search technique.

The grid search exhaustively generated candidates from a grid of parameter values that I gave as follows.

- max depth: The maximum depth of the tree
  - It controls the size of the tree to prevent overfitting.
- max_features: The number of features to consider when looking for the best split
  - If you increase it, it usually improves the performance of the model, however you might end up decreasing the speed of algorithm by doing so.
- min_samples_split: The minimum number of samples required to split an internal node
  - If it is a small number, the tree tends to overfit, wheres if it's a large number, it will prevent the tree from learning the data

As a result, it picked parameters: max_depth=35, min_samples_split=4, max_features=None, and r-squared score improved from 0.61 to 0.86.

IV. Results

Model Evaluation and Validation

The best performance parameters

- max_depth: 35
- min_samples_split: 4
- max_features: None
- r-squared score: 0.86

I picked the decision tree model with the parameters above as best because it's easy to see the logic of the model because it shows you what features are considered important. This is beneficial as fundamental analysis have been studied and practiced among investors, so it's easy to justify the logic.

As I mentioned earlier, there are three step I went through to evaluate the model.

1. I had the best model predict prices of all the stocks in one year that the model had never seen (I saved 2015 - 2016 year data for this evalation)
2. I calculated the CAGR of 30 stocks that would get the highest growth in price, using the sum of original price of stocks as the beginning value and the sum of the next year price as the ending value. These next year prices are not the ones predicted by the model, but are the actual prices of the following year.
3. I compared the result to the CAGR of return on the market.

The CAGR of the 30 stocks was 0.23 (22.85%)

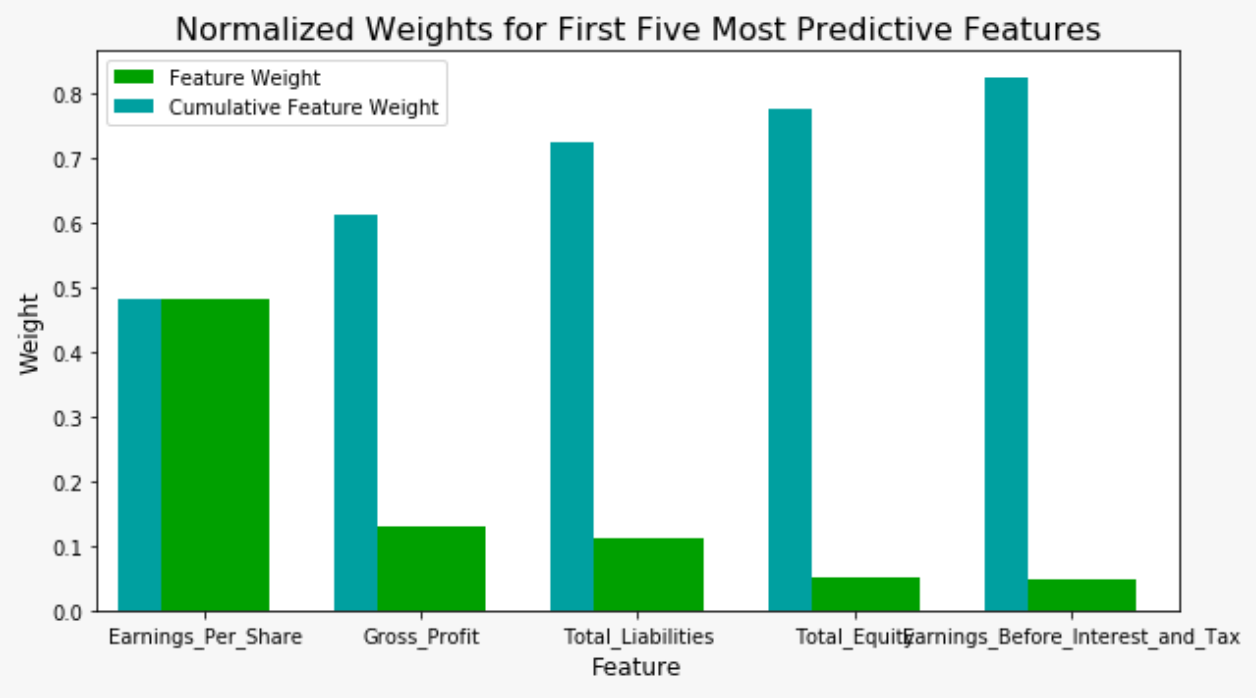
Compare the CAGR to the entire market's CAGR of 2016

30 stocks	Market	Diff
22.85%	11.74%	+ 11.11 %

Justification

As you see, the growth of the stocks picked by the model surpassed the growth of the market.

To justify the result, I looked at the features considered as important by this model.



From the graph above, Earnings Per Share is considered the most important by this model.
This makes sense from a perspective of investors. Investopedia, which is the world's leading source of financial/stock market content on the web, say the following about Earnings Per Share.

> Earnings per share is generally considered to be the single most important variable in determining a share's price. It is also a major component used to calculate the price-to-earnings valuation ratio.

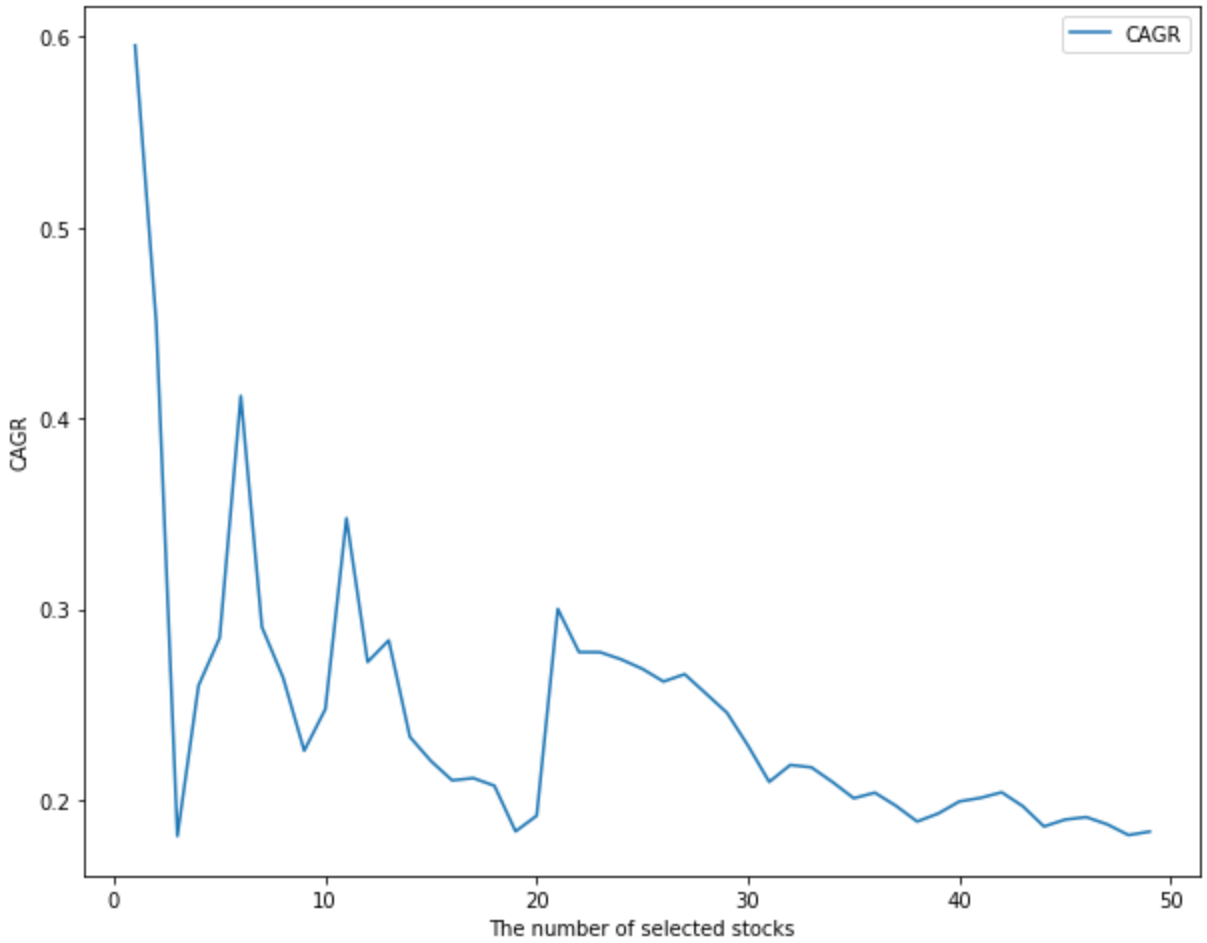
The quote obtained from this page: http://www.investopedia.com/terms/e/eps.asp

V. Conclusion

Free-Form Visualization

I wondered how many stocks are appropriate amount to buy, so I calculated CAGR of when the number of stocks is x with x being from 1 to 50.

The average of CAGR is 0.24(24.469886%)



As you see, the more stocks you buy, the less volatile CAGR becomes.
It is a common wisdom among investors that having more than 20 - 30 stocks at a time makes your portfolio more balanced.
This wisdom has been proven true here too.

Reflection

The most difficult challenge I faced in this project was processing the datasets.

The datasets are taken from Kaggle but to make it usable for this project, I had to do the following things.

- Refining the datasets
  - There were two datasets: one was a dataset of corporate fundamentals, and the other was a dataset of stock prices. Some companies were listed in the both datasets, but others were only in one dataset. I had to remove the latter.
- Associating the datasets
  - I had to associate corporate fundamentals of a year and stock prices of the following year.
- Filtering the dataset
  - While I was on Model Evaluation and Validation process, I discovered that the number of the stocks in 2015 and the number of the stocks in 2014 didn't match. That's because some might've gone bankrupt and got removed from the market or others might've joined the market from 2015. So I had to filter the companies with symbols.

This challenge gave me an opportunity to get more familiar with numpy, pandas, and matlab and taught me how important to be aware of what kind of data you deal with before you get started and stay careful of them during a project.

One last thing I wanted to mention was whether I would use this model or not. The answer was no, because it didn't take brokers' fees into account and inflation was not adjusted at all.

Improvement

Next time, I would implement ensemble method and RandomForestRegressor algorithm in order to improve generalizability or robustness.

Also, I would save more datasets for evaluation process, so that I can test the performance of a model more thoroughly with those unseen datasets. Also, if I do that, the amount of dataset I can use to feed a model would be small, so I would use k-fold cross-validation training technique, which makes it possible for a model to learn better when the amount of dataset is small.