Stock Price Prediction with Machine Learning

Independent Study Final Project

Yuchen Liang Vishnu Gosai Daniel Xu

Mr. Wizzard

IS: Machine Learning

November 14, 2019

Abstract

This project implemented a customized Linear Regression model with Gradient Descent to predict stock prices based on past prices and other features. The results demonstrate that our customized Linear Regression model performs at the same level as generic Linear Regression model and the Support Vector Regression model in the scikit-learn library, with a mean absolute error of 29.762 and a R^2 value of 0.979.

Keywords: **Linear Regression, Gradient Descent, back propagation**.

Introduction

In the midst of the craziness that is Wall Street, very few investors, which include the likes of Warren Buffet and Ray Dalio, haven risen to the very top through the stock market. It's a near-insurmountable task to predict the stock market accurately, especially using only human brainpower. Using machine learning, we hope to simplify the task. Computers are able to parse through enormous amounts of significantly faster than any human, allowing them a wide range that's easily accessible. With machine learning, computers can train themselves using the data provided to make their predictions more accurate without any explicit programming, automating the process. The purpose of this project was to predict the future stock price of companies using past share prices and additional features from existing stock market data. We use the features to generate a Linear Regression model that we can use to estimate future stock prices.

Using machine Learning to Predict stock prices

While using machine learning to predict stock prices takes away the task of having to process massive amounts of data to come to a conclusion of which stocks to purchase and trade, there are still many variables that have to be taken into account when building a program and interpreting data. This passage primarily focuses on the relationship between Goodhart's Law and the Efficient Market Hypothesis and the issues surrounding predicting a stocks price.

First a definition of Goodhart's Law and the Efficient Market Hypothesis. The Efficient Market Hypothesis states that people participate in the stock market solely based on all available information, and do not participate in artificial trading(artificial trading being emotion trading or trading on impulse and feeling). A definition of Goodhart's law has been phrased by Marilyn Strathern as "When a measure becomes a target, it ceases to be a good measure."[1] An example of this in the context of the stock market would be that there is some company A. They hire a firm to make a model to predict the direction of the market, and the company's future shares and price. Company A then uses the model to reoptimize there sales and purchases of shares and

[1] YouTube (YouTube), accessed November 15, 2019, https://www.youtube.com/watch?v=SwcK7NI_i98)

assets. When company A decided to use the model in this way they have then fundamentally changed something and there will be a measurable drift in the model based off of this. From this we can say that Goodhart's law is that any data set that is dependent upon time and human interaction will be subject to some drift in the predicted value. This human factor is a big issue when using machine learning.

If the stock market operated on the Efficient Market Hypothesis then that would mean that trading occurs only with all the available knowledge available to all investors. This would mean that insider trading does not occur, and artificial trading. However, trading based on feeling and insider trading most certainly occur. In addition to the human factor being a big impact on predicting stock prices, statistically speaking you cannot predict a stock priced off of historical trends alone.

Returning to Goodhart's Law again, fundamental changes in an object/entity will result in a measurable drift and as time passes things will change in what affects a company's stock price. The fundamental change I would like to talk about is where the revenue comes from in a companies. Specifically what I am referring to is the globalization of nations economy. Data taken from quandl begins in 2004 since then the U.S. interactions with outside economies has been variable. We formed the Trans-Pacific Partnership (TPP) during the Obama administration, and withdrew from it in 2017 and are now in a "trade-war". As time passes, changes will occur and looking at a single stock by itself becomes unreasonable. Later in the write-up we will discuss how to take into account other markets in the model.

Related Works

A project created by researchers at the Hong Kong University was similar to ours in that they built a model that had the goal of predicting the S&P 500. The S&P 500 is as close of a representation as you can get of the U.S. economy. In the paper it shows that 70.4% of the revenue of the 500 companies that make up the S&P 500 index comes from the U.S. and 99.8% of those companies main offices are in the U.S.[2] To achieve a higher level of efficacy they still needed a way to take into account the last 30%. To do so, they used a correlation matrix that showed the relationship between the S&P 500 index and other major nation world indexes. They found that FTSE, the London Stock Exchange, comprised of the 100 highest capitalization companies, NIKKEI, the Tokyo Index, DJIA, Dow Jones Industrial Average, and the NASDAQ have a measurable effect on the S&P 500.[3] From the model they built they were able to take into account foreign markets and therefore take into account an issue discussed early. The features for their program included RSI, relative strength index, CCI, commodity channel index, and Accumulation/Distribution, which became more useful for data analysis. RSI, CCI, and

[2] Ismail, Hossain. "Forecasting the S&P 500 Index using a Machine Learning Approach", November 15, 2019. https://pdfs.semanticscholar.org/c00d/797bce4098d3fa3c996a3280489b9500eb84.pdf
[3]Ibid

accumulation functions give a value on a scale from 1-100 that shows whether a stock is overvalued or undervalued. Now when these values for American companies are compared to the model that is imitating the U.S. economy, the conclusion of whether a stock will drop or not can be made and the weights in the model can be changed in this regard.

Core Method

This section illustrates the structure of our customized multivariate Linear Regression model based on Gradient Descent.

The idea of Linear Regression is to approximate a target value as a linear function of x.

$$h(x) = z = \sum_{i=0}^{n} w_i x_i + b_i = w^T x + b$$

where w is the weight and b is the bias.

Given a $n \times m$ matrix X, and ground truth value $\widehat{y}$ $(1 \times m)$, w is initialized as a vector that has a size n with each element randomly generated in a standard normal distribution, and b is initialized as 0.

Next, the least-squares cost function was defined as below:

$$J = \frac{1}{2m} \sum_{i=0}^{m} (h(x)^{(i)} - y^{(i)})^2$$

The task for the model, therefore, is to repeatedly change the parameters w and b until they converge to a value that minimizes the cost J. This minimization can be achieved with back propagation and gradient descent. In this model, the following update rule is adopted:

$$w_j := w_j - \alpha \cdot \frac{\delta}{\delta w_j} J(w, b)$$

$$b_j := b_j - \alpha \cdot \frac{\delta}{\delta b_j} J(w, b)$$

where α is the learning rate of the model.

During implementation, the partial derivatives can be simplified as following:

$$\frac{dJ}{dw_j} = \frac{\delta J}{\delta z} \cdot \frac{\delta z}{\delta w_j} = \frac{\delta J}{\delta z} \cdot X^T$$

$$\frac{dJ}{db_j} = \frac{\delta J}{\delta z} \cdot \frac{\delta z}{\delta b_j} = \frac{\delta J}{\delta z} \cdot 1 = \frac{\delta J}{\delta z}$$

Feature Engineering

The features used for the customized multivariate Linear Regression model are pulled from stock market data we gathered on Quandl. From various companies, the features include adjusted opening price, the share price at the beginning of the trading day; adjusted high price, the highest a share price is during that trading day; adjusted low price, the lowest a share price is during that trading day; adjusted closing price, the share price just before the market closes; and adjusted volume, the amount of shares bought or sold on that trading day.

We used the adjusted high and low prices to calculate the high-low percentage, which is what percentage the share price fluctuated between based on the closing price of that daw. Additionally, we calculated the percent change of the share price, telling us how much a share increased or decreased in proportion to its opening price.

Eventually, we have four features for our input:

A. ***Adjusted Close Price****: Adj. Close Column in the dataset*

B. ***High/Low Percentage*** *= (Adj. High - Adj. Low) / Adj. Close × 100%*

C. Percent Change = *(Adj. Close- Adj. Open) / Adj. Open × 100%*

D. Adjusted Volume*: Adj. Volume in the dataset*

Results

A. Model Comparison

Two methods were used to evaluate the performance of the algorithms.

The R^2 measures the confidence level of the model, while the MAE (Mean absolute error) measures the difference between two continuous variable as defined below:

$$\mathrm{MAE} = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n} = \frac{\sum_{i=1}^{n} |e_i|}{n}.$$

The result in Table I shows that our Customized Linear Regression model's performance is as good as Scikit-learn's LR model and is even better than SVR.

Table I: Model Performance Comparison

Model	***R^2***	***MAE (Mean absolute error)***
Customized Linear Regression	0.979	29.761
Sklearn Linear Regression	0.979	29.761
Sklearn Support Vector Regression	0.977	30.964

Other parameters: eta(learning rate)=0.4, epochs=500, test_size=0.2, random_state=5

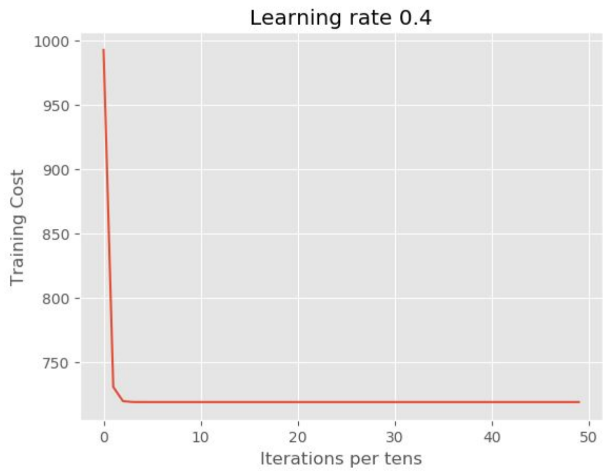

Figure 1-A: Training cost over iterations with learning rate of 0.4

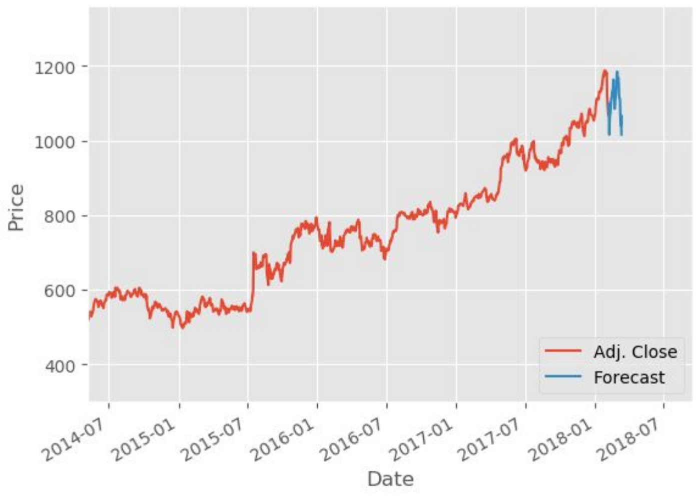

Figure 1-B: Adj. Close Price over time with learning rate of 0.4

B. *Learning rate*

Learning rate was also altered to see if there would be a change in the model's performance. Figure 2-A and 2-B demonstrated that except at eta=0.001, changing the learning rate has little impact on the performance of the algorithm.

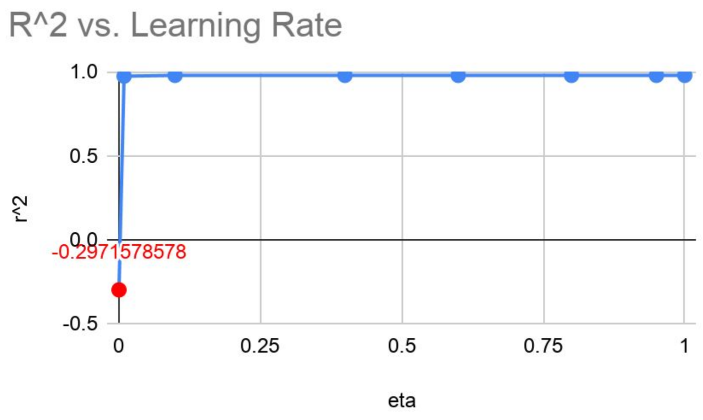

Figure 2-A R^2 over change in learning rate

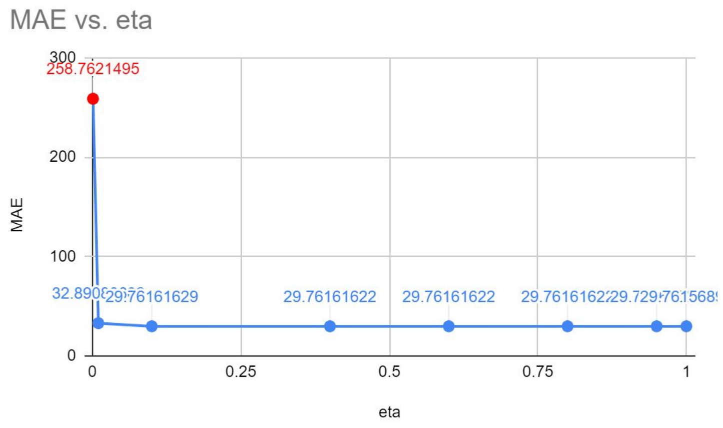

Figure 2-B Mean absolute error over change in learning rate

Conclusion

In conclusion our model was effective. It has a mean absolute error of 29.762 and a R^2 value of 0.979. The source code of the entire project is posted on Github for use and reproducing the same result. Our model is as good as the SK learning python model, but it can still be improved. What we can do to improve our model is to add features that are momentum calculations such as RSI, CCI, and accumulation/distribution. Momentum calculations are rate of change calculations that indicate whether a stock is overvalued or undervalued and is therefore projected to increase or decrease in price. So adding this would give us a more effective way of changing the weights in our model. Another point to make about RSI, CCI, etc. is that since they return a value from 1-100 they need a reference point to say what is overvalued or undervalued. The typical convention is that 70 and above is overvalued, and 30 and below is undervalued. But it has been recorded where an RSI of 55 showed that the company was overvalued.[4] So a more effective way to make a comparison would be to build a model of the S&P 500 and use it to make comparison for the corresponding time periods.

[4] James Chen, "Relative Strength Index – RSI," Investopedia (Investopedia, June 10, 2019), https://www.investopedia.com/terms/r/rsi.asp)

Bibliography

YouTube. YouTube. Accessed November 15, 2019.
https://www.youtube.com/watch?v=SwcK7NI_i98.

Chen, James. "Relative Strength Index – RSI." Investopedia. Investopedia, November 15, 2019.
https://www.investopedia.com/terms/r/rsi.asp.

Ismail, Hossain. "Forecasting the S&P 500 Index using a Machine Learning Approach", November 15, 2019.
https://pdfs.semanticscholar.org/c00d/797bce4098d3fa3c996a3280489b9500eb84.pdf