Stock Portfolio Creator

Table of Contents

Introduction

The stock market has been a part of the global economy since the 17th century when the Dutch promoted the growth of several financial innovations that lead to the creation of the modern financial system.[1]

It has been long believed that the stock market is highly unpredictable and its movement cannot be 'predicted' by any form of statistical measure. I, on the other hand, believed that every number has to have a cause. Predicting future stock prices can be considered the holy-grail of quant. Edward Thorp, a scientist, demonstrated in the 1960s and 1970s that a small edge in the stock market can provide massive returns. *The main objective of this project was to use machine learning and big data to create a portfolio of stocks that comprised mostly, if not completely, of winners.* Winners refer to a set of stocks that gave an overwhelmingly positive return. On the other hand, losers refer to a set of stocks that underperformed and gave poor returns. The algorithm would rank stocks on the basis of their predicted quarterly return and the portfolio would be readjusted quarterly. Extensive backtesting would also be done to tweak the model and algorithm to ensure the creation of an excellent portfolio. The data being used for this project is for the BSE500 from the year 2009 to the year 2015.

The algorithm at the core of this project is iterative gradient descent - a supercharged version of the gradient descent algorithm. The algorithm not only calculates the weightage assigned to each factor, but also selects the factors that best describe a security's stock price trajectory.

[1] Goetzmann, William N.; Rouwenhorst, K. Geert (2005). The Origins of Value: The Financial Innovations that Created Modern Capital Markets. (Oxford University Press, ISBN 978-0195175714))

Chosen Factors and Why They Were Chosen.

There are several categories of factors that affect the movement of a stock - fundamental, momentum, quality, growth and technical factors. The factors selected as ‘variables’ for our gradient descent model were chosen to ensure an equitable representation of all the categories of factors. They included:

1. Quarterly Change in Oil Prices (in %), in Gold Prices (in %) and in INR to $ Prices (in %) - These were some macro-factors that were added to account for the influence of the global market on the local Indian market.
2. EV/EBITDA - EV/EBITDA is a ratio of the enterprise value (EV) of a company to the company’s earnings before interest, taxes, depreciation and amortization (EBITDA).
3. PE Ratio - The price to earning ratio is a ratio of the current per-share price to the earning per share.
4. Momentum - Momentum is a factor that indicates the performance of the stock in the last 11 months. It was included as a means to provide a holistic view to each data point in the data set.
5. Sharpe Ratio - The Sharpe ratio is the average return earned in excess of the risk-free rate per unit of volatility or total risk. It is a measure used for calculating risk adjusted return.[2]
6. Asset Turnover - Asset Turnover is a ratio of net sales to average total assets. It is commonly used as a measure of efficiency. If Company A has an asset turnover of 0.7, then it means that the company generates $0.70 in sales for each dollar of its assets.
7. ROE - Return on Equity (also return on net worth) is a measure that indicates the how many dollars of profit a company makes with each dollar of shareholders’ equity. It is commonly used as a measure of profitability.
8. ROIC - Return on Invested Capital is ratio of the net operating profit (after taxes) to total invested capital.
9. Volatility - Volatility is a measure that indicates the dispersion of returns for the given stock. A high volatility indicates that a stock’s price can fluctuate dramatically in a short

[2] http://www.investopedia.com/terms/s/sharperatio.asp

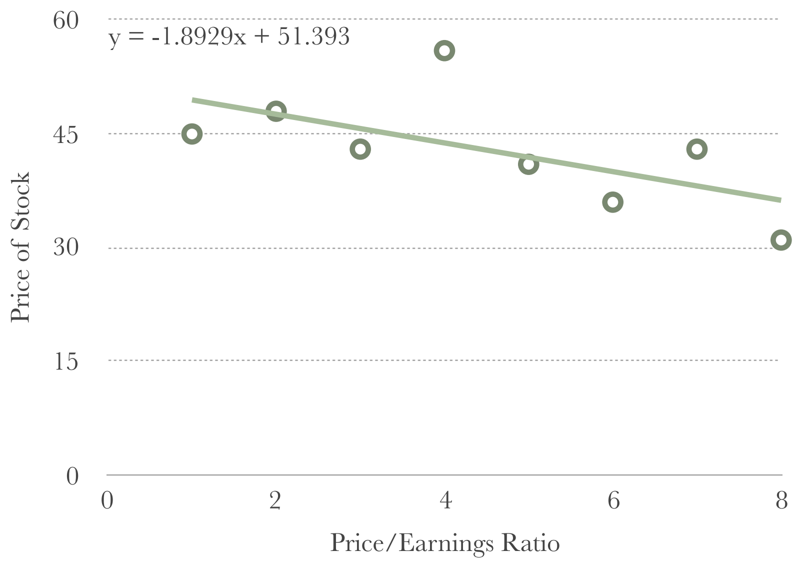
period of time. On the other hand, a low volatility indicates that a stock's price will change steadily over a period of time.

10. Earning Surprise - Earning Surprise is a ratio of year-over-year quarterly growth in earnings to the standard deviation in quarterly growth. The higher the earning surprise, the better the stock will perform.

Core Algorithm and Engineering Process.

The core algorithm used for the portfolio generator is gradient descent, a supervised learning algorithm. Gradient descent can also be classified as a discriminative learning algorithm where it tries to calculate p(y|x; ø) - the probability of y given x. Gradient descent is primarily used to find a linear numerical relationship between a given set of factors and a resultant final value - in this case x and y respectively - allowing us to 'predict' a future y-value using a x-value.

Gradient descent was chosen for this project for two primary reasons namely, the lack of computation power to build a neural network and the algorithm's ability to be fast and reasonably accurate.

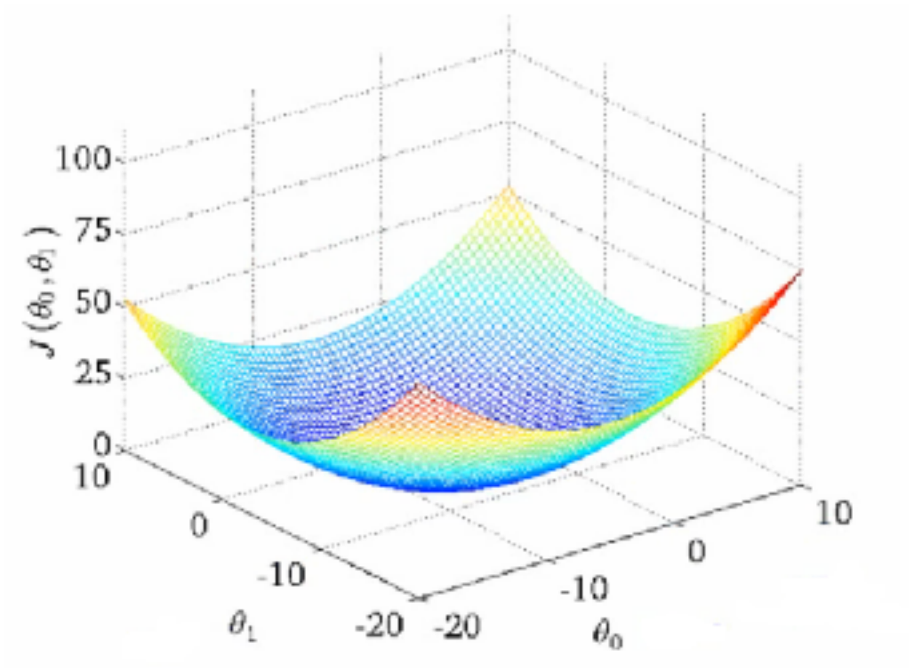


The following graph explains the mental reasoning for choosing gradient descent as our machine learning algorithm of choice. The graph highlights the relationship between the stock's price and the Price/Earnings Ratio. The straight line (y = -1.8929x + 51.393) describes the relationship, in numerical terms, in a form that has the least amount of error - this is also known as linear regression. In the context of the portfolio creator, gradient descent allows us to find the weights associated with each parameter, enabling us to 'predict' future returns of a stock.

As described earlier, the main aim of the gradient descent algorithm is to reduce the cost function. The cost function used for this application is as follows:

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

To visualize the working of the cost function, consider the figure below where the cost function is plotted as a contour graph.[3]



The main objective is to reach the global minimum of the cost function, the deep-blue part of the contour map. By starting at a random location on this map, we take small steps towards the minima by using the derivative of the cost function. This method is slow if the chosen is too small or is unpredictable if the chosen is too big. To overcome this issue I used the normalized version of gradient descent. The normalized equation form for gradient descent is as follows:

$$\theta=(X.X^T)^{-1}.X^T.Y$$

[3] http://www.holehouse.org/mlclass/01_02_Introduction_regression_analysis_and_gr.html

Magic Number

An internal method called the 'Magic Number' was used by the company to rank stocks to create a portfolio. Magic Number, an homage to Joel Greenblatt's 'The Little Book That Beats The Market', is calculated by taking the sum of individual rankings of the stock in 8 categories with respect to the rest of the stocks in the index. The eight factors used were dividend, sharpe ratio, CFO-NI, delta asset turnover, CROIC, delta CROIC, B/P and earnings yield. Each of these factors had a weight associated with them.

For example, a typical data point for an index would look something like this:

Associated Weights	1	2	0.5	3	1	2	0.75	1
Factors	**Dividend Rank**	**Sharpe Rank**	**CFO-NI**	**Delta Asset Turnover**	**CROIC**	**Delta CROIC**	**B/P**	**Earnings Yield**
Stock A	263	206	186	72	180	183	219	120
Stock B	242	279	61	293	234	31	44	211

* the greater the rank, the better.

The Magic Number for Stock A = 1 * 263 + 2 * 206 + 0.5 * 186 + 3 * 72 + 1 * 180 + 2 * 183 + 0.75 * 219 + 1 * 120 = 1,814.25.

The Magic Number for Stock B = 1 * 242 + 2 * 279 + 0.5 * 61 + 3 * 293 + 1 * 234 + 2 * 31 + 0.75 * 44 + 1 * 211 = 2,249.5.

We notice that the Magic Number for Stock B is greater than that of Stock A, hence (based on our current weights), Stock B would be a better investment choice. After calculating the Magic Numbers for all the stocks in an index, we choose the top 20% of the stocks - 20 stocks in a 100 stock index - and make a portfolio out of those stocks. However, we don't invest more than 8% of the total available capital in any stock or more than 13% in any sector - this is to prevent the portfolio from becoming company-heavy and/or sector-heavy.

Challenges and Solutions

Even though the initial build of the algorithm gave us decent results, there were a number of problems with our approach.

Initially, instead of letting the computer decide on the correct combination of factors that would accurately describe the trajectory of a stock, we were limiting it to a single combination derived through human instinct as well as trial and error. By allowing the computer to iterate through all possible combinations of the provided factors, not only did the accuracy of stock return predictions increase, but also led to the creation of a portfolio with more winners.

Secondly, because of the high number of training data points for each stock, the iterative gradient descent algorithm was taking too long to run for a set of 100 stocks. To counter this problem, I decided to use the *normal equation form of gradient descent.* This allowed us cut the calculation time by a huge margin. For reference purposes, calculations for 10 stocks that were taking us around 700-800 seconds on average were cut down to 10-20 seconds when we switched to using the *normal equation form of gradient descent*.

Further Work and Conclusion

The final version of the application tested all possible permutations of the factors and choose the one with the lowest overall cost. Mathematically, this resulted in the program going through 2^n permutations, where n is the number of factors that were being used. Due to this exponential time complexity, the machine being used to run this algorithm/application[4] wasn't able to handle more than 12 factors at a time. This leaves room for future experimentation with more factors and more complex combinations (multiplication of factors, etc).

Another variation of the application that uses neural networks instead of gradient descent can be created - this would ideally result in much higher accuracy, but much slower training speed. This is a variation that I aim to tackle in the future.

Overall, the objective of the project was met. The algorithm and model was able to create a portfolio that was comprised mostly of winners (60% winners and 40% losers).

[4] Late 2012 MacBook Pro Retina 13"