Mar | 2017

Systematic Equity Trading

Capstone Project

Machine Learning Engineer Nanodegree – Udacity

Faisal Hamid

Project Overview

Trading in financial securities began hundreds of years ago in Europe and revolutionized the way in which economies and global commerce functioned. Since then, people have been dedicating time and resources to research, in order to gain an advantage over competitors and other investors.

Global market capitalization of equities was approximately USD 62 trillion with the United states accounting for USD 25 trillion of that[1]. Most exchanges today no longer consist of people using hand signals on a trading floor, but rather of computer networks that discover prices and settle trades. This transition has led to investment funds relying on computer driven strategies to increase their profitability. In a 2017 article by Forbes, "four of the top 20 hedge funds that have generated the highest amounts of net returns are highly reliant on algorithmic trading". The availability of freely available historical stock data for a large number of highly traded companies has opened up opportunity for anyone to experiment with systematic trading strategies.

The purpose of this project is to use machine-learning techniques to discover a systematic trading strategy that can be used to generate profits in live trading. We begin with a brief discussion of the problem statement before proceeding to the detailed analysis of the individual steps taken to address the challenge.

Problem Statement

The purpose of this project is to develop a stock price predictor that applies machine-learning techniques to historical data and is capable of predicting prices 1 day in advance. The goal is to use these predictions to generate profits in the market by submitting trades based on the programs recommendations on which stocks to buy and sell for the following day.

The choice of companies shall be restricted to companies that are included in the S&P500 index, as these companies are generally the most liquid and data is available for a longer period of time. Specific features that shall be used are:

- Historical Price Data
- Historical Volume Data
- Sector Information

Using these features, we shall attempt to train a model that classifies the following day's stock price change for each stock, using the data we have available at that

[1] Data hosted by The World Bank (http://data.worldbank.org/indicator/CM.MKT.LCAP.CD?locations=US&name_desc=true)

point in time. The process that will be followed as well as some required parameters include:

1. Select a window size to determine the training set size (i.e. # of most recent trading days).
2. Standardize the price data by finding the percentage daily price change divided by the mean of the absolute value percentage change. This will ensure that the values are comparable across different stocks and may be more volatile than others.
3. Standardize the volume data by dividing the daily volume by the average volume over the training period.
4. Select and train a classifier using the 1-day forward returns converted to a binary variable representing 'up' days as 1 and down days as 0. We will look at data from all companies to train the classifier in order to capture any potential relationship not specific to any individual company.
5. Use time series cross validation as well as grid search to tune parameters and find the best model.
6. Test the results on actual data for the next day and repeat the exercise on a rolling window iterated over the remaining sample to generate an out of sample performance of the model.

Before we can start this exercise, it is important to determine what metrics to use in order to decide how to compare different models.

Metrics

There are two different types of metrics that we shall be using in this exercise. First, we are going to use the F1 score to train and test the model:

- To find the best prediction model, we shall use the F1 score where:

$$F1 = 2 * \frac{precision * recall}{(precision + recall)}$$

$$Precision = \frac{True\ Postives}{(True\ Positives + False\ Positives)}$$

$$Recall = \frac{True\ Positives}{(True\ Positives + False\ Negatives)}$$

Stock market returns tend to have a positive bias, as markets tend to go up over time[2]. Furthermore, we would also like to be able to correctly identify downward

[2] This is true for the companies that manage to survive and is known as *survivorship bias;* discussed in more detail later on.

movements in stock prices so that we can profit from those as well. The F1 score accomplishes this by also incorporating the number of positive samples that are missed by the model (in the recall score). However, correctly identifying up and down price movements is only an intermediary step. What we are actually concerned with is whether this information can be used to profit, consistently, in the market. As a result, the final evaluation metrics will be the total profit generated by using the model predictions to buy and sell specific stocks of interest to us, as well as a modified version of the Sharpe ratio:

$$Total\ Return = \prod_{d=1}^{Total\ Days} (1 + \sum_{i=1}^{Total\ Stocks} \omega_i * Stock\ Return_i)$$

$$Sharpe\ Ratio^3 = \frac{Annualized\ Return}{Annualized\ Volatility}$$

The total return is essentially the weighted sum of the percentage change in each stock we are buying (or selling) on any particular day. The weights are how much of our portfolio will be invested in any particular security; these will be equal in the default portfolio. Finally, the returns are then compounded over the total number of days in the sample to find the total return over the entire period. However, in the financial markets, we also have access to *leverage,* which allows us to magnify our total return by orders of magnitude using borrowed money. This can be quite risky unless our strategy is consistent enough to not get wiped out during a market crash. A portfolio with a high Sharpe ratio and low total return can actually be safely levered (i.e. using borrowed money to multiply the return) to yield a better portfolio than a comparable portfolio with a high return with low consistency.

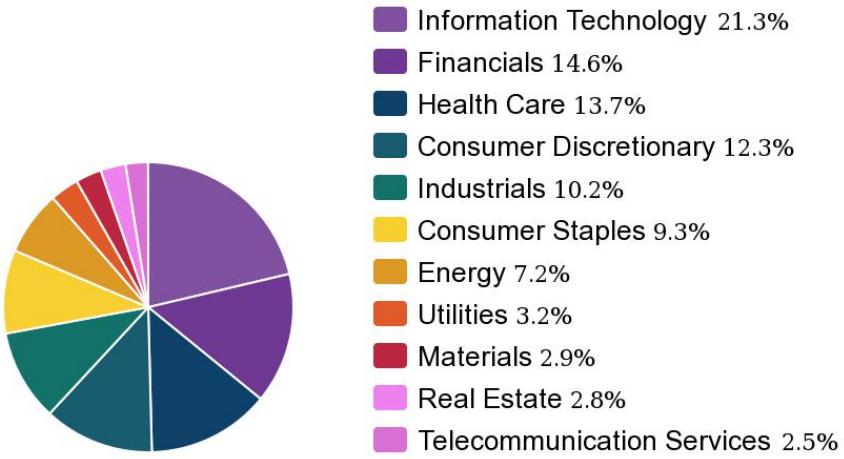
With these metrics in mind, we can begin exploring the data before discussing the algorithms and techniques in more detail.

Data Exploration

The S&P 500 is a market-cap weighted index of the 500 largest US companies. This is essentially a weighted average of the percentage change in prices of the companies where the weights correspond to the price multiplied by the number of shares outstanding for each company. The companies can be grouped by their respective sectors and companies in each sector tend to be correlated over time. According to the official S&P Dow Jones Indices website, the sector breakdown by market capitalization as of January 2017 is as follows:

---

[3] Usually, the sharpe ratio is calculated as the difference of the portfolio return and a risk free bond, divided by the volatility. However, in this case we will ignore the bond return and focus on just the consistency of returns for the portfolio itself.



Based on GICS® sectors
The weightings for each sector of the index are rounded to the nearest tenth of a percent; therefore, the aggregate weights for the index may not equal 100%.
As Of Jan 31, 2017

These breakdowns are not constant over time; in the United States, traditional sectors have slowly receded making way for a large information technology sector.

The index constituents change over time as new companies are added and poor performers removed; the idea is to attempt to track the largest 500 at any given point in time. As of January 2017, the Top 10 constituents by market cap include:

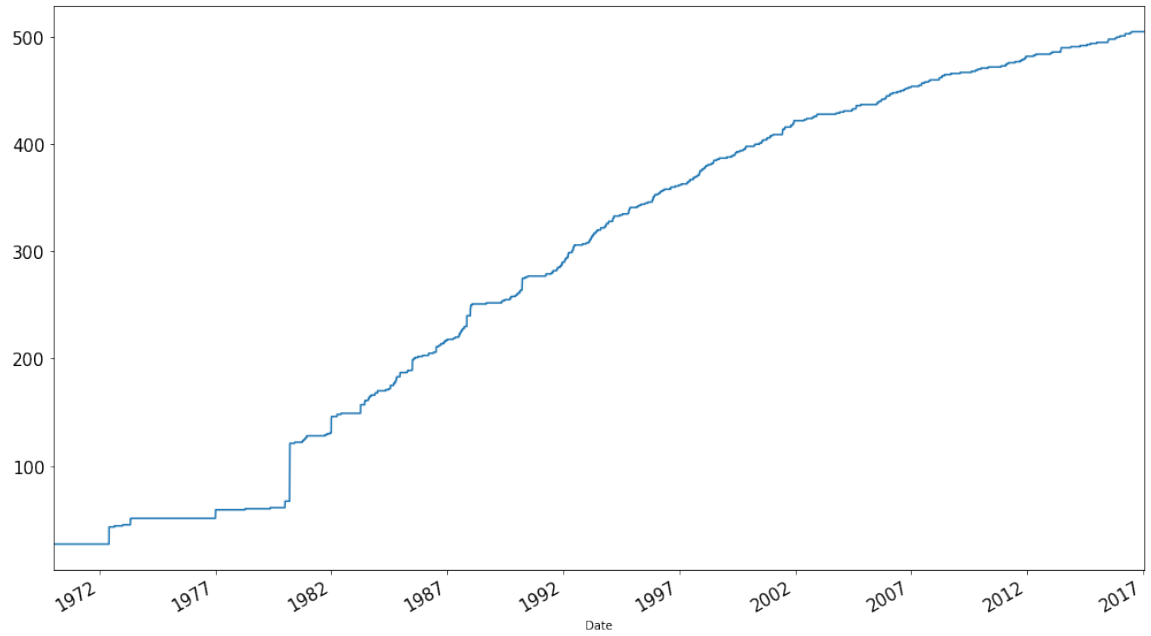
CONSTITUENT	SYMBOL	GICS® SECTOR
Apple Inc.	AAPL	Information Technology
Microsoft Corp	MSFT	Information Technology
Exxon Mobil Corp	XOM	Energy
Amazon.com Inc	AMZN	Consumer Discretionary
Berkshire Hathaway B	BRK.B	Financials
Johnson & Johnson	JNJ	Health Care
Facebook Inc A	FB	Information Technology
JP Morgan Chase & Co	JPM	Financials
General Electric Co	GE	Industrials
AT&T Inc	T	Telecommunication Services

The index was launched March 4th, 1957 and has gone through many changes since then. As a matter of fact, of the 505 tickers[4] that are part of the index as of October 2016, only 27 of them were part of the index in 1970. Unfortunately, we do not have a historical point in time database that will give us the historical companies and

[4] Some of the companies actually trade under two different tickers, Google or as it is now known, Alphabet is a an example.

their prices. Consequently, we will have to make a trade-off between the number of companies in our sample and the length of the sample time period. Figure 1 shows the number of companies that were also reporting data over different time periods over history:

Figure 1: Number of Companies reporting Data



The figure shows that the index actually changes quite a bit over time and that only 27 companies of the 500 today have been part of the index from the beginning. Furthermore, there was no way for us to know in 1970[5] that these companies would be part of the index in 2016 and that the others would not. This is known as survivorship bias, whereby only the companies that survive remain part of the index and as such the returns are slightly biased upward as the weak companies that may have underperformed are not part of our sample. Keeping this bias in mind we continue to explore the available features, which included historical daily prices, historical daily volume and sector classification.

The price and volume data is available from January 1970 until 2017, providing a total of 11,879 trading days, for a total for 505 stocks (not for all dates though). However, the data earlier data is restricted to a very small sample and summary statistics will be skewed if they are included. Since providing statistics for all companies is not very useful, the following table provides some summary statistics on the average values across companies for each specific sector. The summary is provided from the year 1990 onwards, which includes, at a minimum, little over half the reporting companies as well as 6,823 trading days:

	Consumer	Consum	Energ	Financia	Healt	Industria	Informatio	Material	Real	Tele-	Utilitie

[5] This is the earliest data point available on Yahoo Finance

	Discretionary	er Staples	y	ls	h Care	ls	n Technology	s	Estate	communications Services	s
mean	0.08	0.07	0.07	0.07	0.09	0.07	0.12	0.06	0.08	0.04	0.05
std	1.33	0.89	1.72	1.59	1.15	1.23	1.77	1.30	1.52	1.44	1.11
min	-10.95	-6.20	-16.89	-15.26	-7.72	-9.49	-9.51	-11.41	-17.38	-9.19	-9.40
25%	-0.55	-0.40	-0.73	-0.54	-0.49	-0.49	-0.76	-0.52	-0.45	-0.69	-0.44
50%	0.11	0.08	0.05	0.09	0.12	0.10	0.15	0.08	0.07	0.03	0.08
75%	0.73	0.55	0.94	0.69	0.72	0.65	1.01	0.68	0.62	0.76	0.57
max	11.02	8.17	21.84	15.43	12.37	11.52	15.74	12.37	24.45	11.39	14.95

The data is presented in percentages and we can see that the min and max increases are quite large relative to the mean values. This is a well-known attribute of financial market return distributions and can drastically impact the performance of any portfolio. Another useful statistic is the pairwise correlations for the sectors over the same time period:

	Consumer Discretionary	Consumer Staples	Energy	Financials	Health Care	Industrials	Information Technology	Materials	Real Estate	Telecommunications Services	Utilities
Consumer Discretionary	1.00	0.73	0.55	0.82	0.75	0.89	0.74	0.79	0.71	0.59	0.49
Consumer Staples	0.73	1.00	0.48	0.67	0.70	0.73	0.53	0.67	0.54	0.52	0.58
Energy	0.55	0.48	1.00	0.56	0.52	0.63	0.45	0.68	0.50	0.41	0.48
Financials	0.82	0.67	0.56	1.00	0.69	0.83	0.63	0.75	0.76	0.57	0.50
Health Care	0.75	0.70	0.52	0.69	1.00	0.75	0.68	0.65	0.55	0.52	0.48
Industrials	0.89	0.73	0.63	0.83	0.75	1.00	0.72	0.87	0.72	0.60	0.54
Information Technology	0.74	0.53	0.45	0.63	0.68	0.72	1.00	0.59	0.51	0.53	0.35
Materials	0.79	0.67	0.68	0.75	0.65	0.87	0.59	1.00	0.66	0.53	0.53
Real Estate	0.71	0.54	0.50	0.76	0.55	0.72	0.51	0.66	1.00	0.47	0.47
Telecommunications Services	0.59	0.52	0.41	0.57	0.52	0.60	0.53	0.53	0.47	1.00	0.44
Utilities	0.49	0.58	0.48	0.50	0.48	0.54	0.35	0.53	0.47	0.44	1.00

These correlations range between 0.35 and 0.89 and can change drastically depending on the time period in question. Consequently it is very important to control for the different sector grouping when training our models later on. The table below provides the same summary statistics for daily volume (in millions):

	Consumer Discretionary	Consumer Staples	Energy	Financials	Health Care	Industrials	Information Technology	Materials	Real Estate	Tele-communications Services	Utilities
mean	3.68	3.51	3.76	4.03	3.59	2.60	14.09	2.20	1.09	7.15	1.61
std	1.67	1.85	2.69	4.61	1.40	1.59	4.86	1.79	1.19	7.23	1.11
min	0.55	0.41	0.23	0.22	0.47	0.28	2.10	0.13	0.02	0.13	0.10
25%	2.54	2.05	1.32	1.12	2.58	1.39	10.99	0.75	0.19	1.08	0.60
50%	3.29	3.13	3.27	2.06	3.37	2.26	13.71	1.67	0.59	4.75	1.39
75%	4.36	4.51	5.61	5.49	4.33	3.30	16.71	3.22	1.82	11.73	2.47
max	16.61	17.63	24.46	46.03	14.08	18.46	53.85	17.01	10.41	133.73	8.06

The correlations can also vary quite a lot, with some being negative:

	Consumer Discretionary	Consumer Staples	Energy	Financials	Health Care	Industrials	Information Technology	Materials	Real Estate	Telecommunications Services	Utilities
Consumer Discretionary	1.00	0.87	0.76	0.79	0.80	0.88	0.25	0.85	0.81	0.70	0.76
Consumer Staples	0.87	1.00	0.84	0.73	0.79	0.88	0.19	0.86	0.79	0.75	0.80
Energy	0.76	0.84	1.00	0.70	0.69	0.83	-0.02	0.89	0.81	0.77	0.85
Financials	0.79	0.73	0.70	1.00	0.68	0.85	0.03	0.83	0.88	0.64	0.75
Health Care	0.80	0.79	0.69	0.68	1.00	0.81	0.30	0.74	0.68	0.62	0.69

Industrials	0.88	0.88	0.83	0.85	0.81	1.00	0.16	0.91	0.88	0.72	0.82
Information Technology	0.25	0.19	-0.02	0.03	0.30	0.16	1.00	0.03	-0.04	-0.05	-0.07
Materials	0.85	0.86	0.89	0.83	0.74	0.91	0.03	1.00	0.90	0.78	0.86
Real Estate	0.81	0.79	0.81	0.88	0.68	0.88	-0.04	0.90	1.00	0.73	0.84
Telecommunications Services	0.70	0.75	0.77	0.64	0.62	0.72	-0.05	0.78	0.73	1.00	0.80
Utilities	0.76	0.80	0.85	0.75	0.69	0.82	-0.07	0.86	0.84	0.80	1.00

Once again, we can notice that there is a very large range of values for each sector, some more so than others. What we are more interested in however, is how these returns and volume features evolve over time. The next section provides some illustrations of what these features look like for the aggregate market.

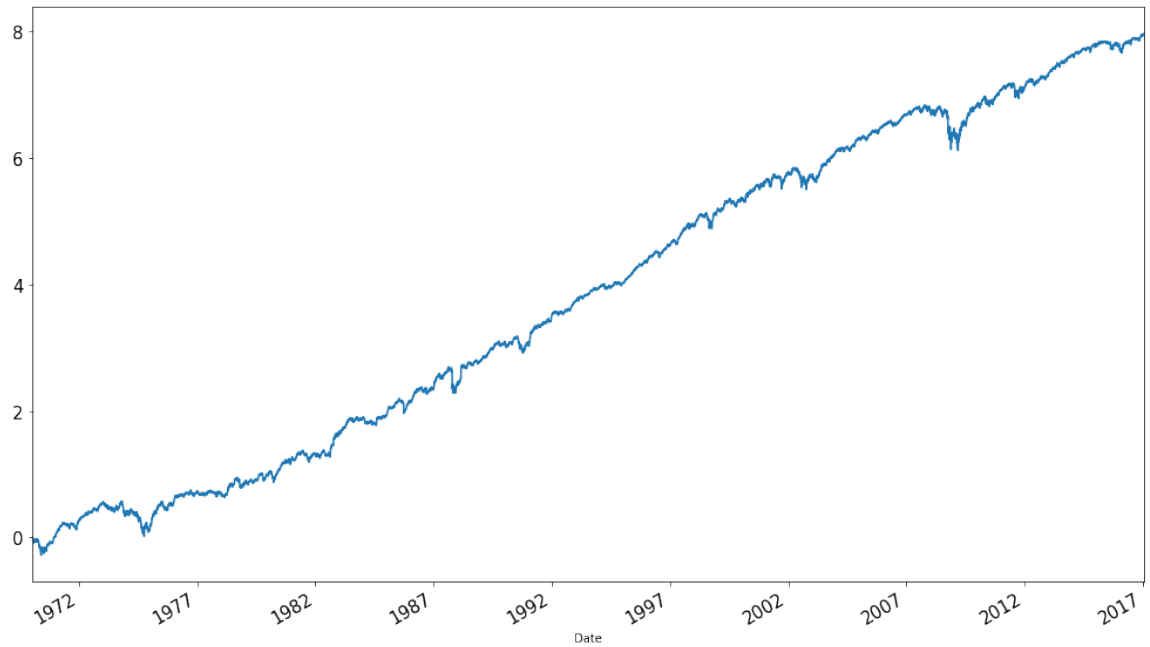
Exploratory Visualization

The three features for each company that we will be using are historical prices, volume and sector information. The following subsections present the a high level summary of the aggregate market rather than a detailed breakdown of each individual company or sector.

Historical Prices

Specifically, we will use the 'Adjusted Close' price from Yahoo. The adjusted close adjusts the price for stock splits/concentrations (these affect the price of the stock but not the value). The adjusted close price also adjusts the price for any dividends announced by the company. Figure 2 shows the average cumulative change (log scale) over time:

Figure 2: Cumulative Avg. Price Change - Log Scale

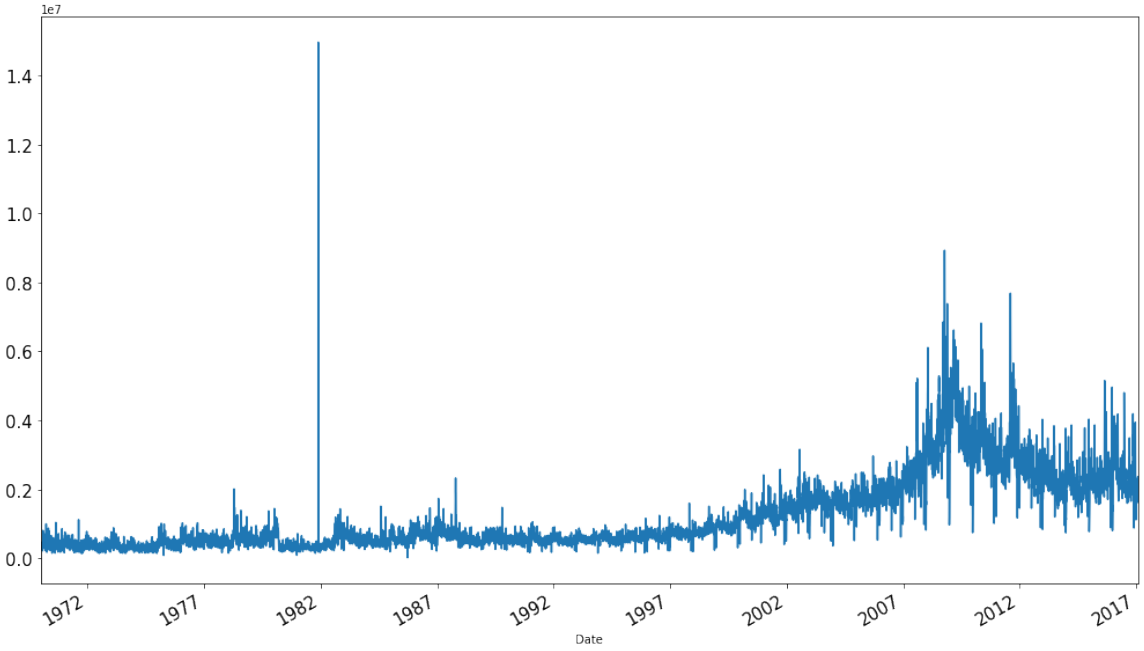


As expected, the market tends to go up over time but can be subject to some major corrections that can be damaging to portfolios. Our goal is to beat this return in terms of both return and volatility.

Historical Volume

Volume can be quite different for each company and can be skewed by outliers. Figure 3 plots the median volume over time:

Figure 3: Median Volume

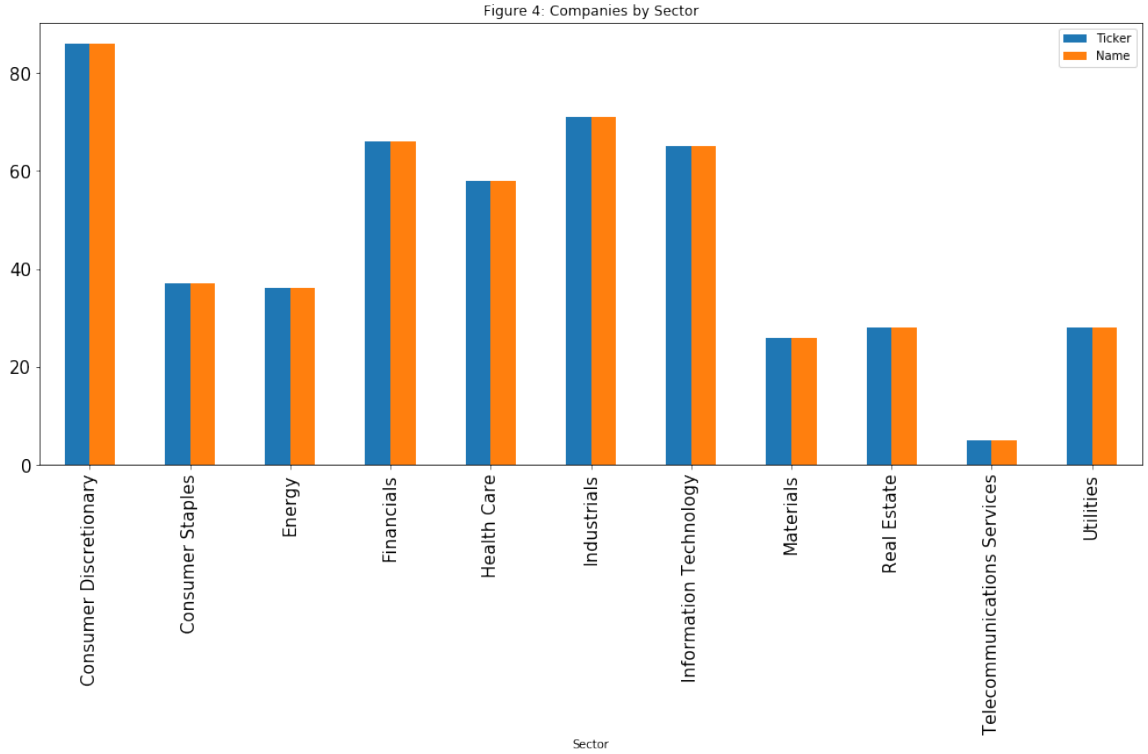


Median volume has also been trending upward over time; some of this can be attributed to the introduction of algorithmic trading and electronic communication networks. Volume has been trending lower more recently from a relatively high level during the financial crisis and market crash of 2008.

Sector Groups

The companies are grouped according to the Global Industry Classification Standard (GICS) and there are 11 different sectors at this point in time (the definitions are revised over time and the number of sectors evolves). Figure 4 provides a summary of the number of companies in each sector for our sample:

Figure 4: Companies by Sector



Note that these are simply the number of companies and will differ from the market cap breakdown provided earlier. The purpose of using sector breakdown is to control for the tendency for companies in the same sector to be highly correlated.

The final data set combines these features in to a multi-level pandas DataFrame that looks like this:

Table 1: Combined Data

Date	Ticker	Stock Price	Stock Volume	Sector Price	Sector Volume
T1	A	Daily Price Change	Daily Volume	Sector Daily Price Change	Sector Daily Volume
	B	…	…	…	…
	C	…	…	…	…
T2	A	…	…	…	…
	B	…	…	…	…
	C	…	…	…	…

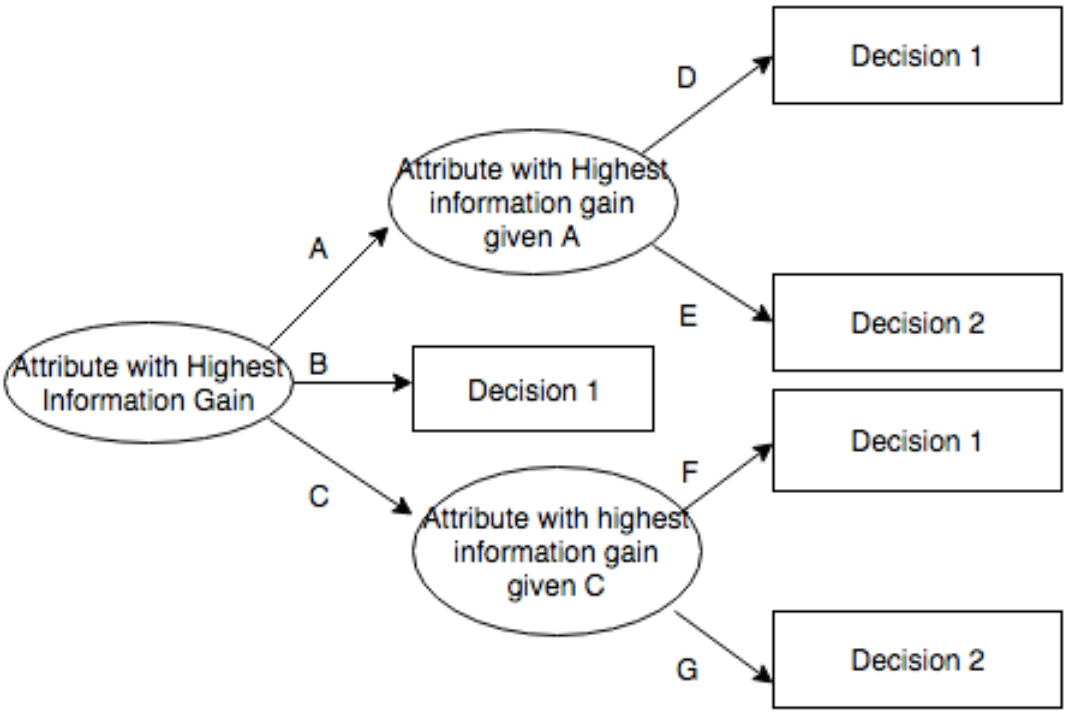
While the variable we are trying to predict will be a binary variable representing up days as a 1 and down days as a 0; so at T1, we are attempting to predict whether the stock will be up (1) or down (0) at T2. This approach essentially ignores the differences between individual companies, with the hope of capturing driving forces common across all companies (or sectors). We will have to standardize the data in some way so that they are comparable across companies and this is discussed further in the data preprocessing section later on.

Now that we have a brief understanding of the features that we shall be using, we move on to a discussion of the algorithms and techniques that we can use.

Algorithms and Techniques

In this exercise, there are quite a lot of data points and the primary criterion for classifier selection was performance. Of the different classifiers tested, only the Gaussian Naïve Bayes, Decision Tree and Logistic Regression classifiers could fit the data in a reasonable amount of time and these will be our choice of algorithms. The others tested include support vector machines (SVC) and a weak learner (AdaBoostClassifier), both of which failed to fit even a small training window in a reasonable amount of time. A quick summary of the algorithms and the relevant parameters specific to each classifier:

- **Decision Tree**: The decision tree model is based on a rule based tree structure that is estimated based on the information gain by restricting features to specific values. The algorithm takes the best attribute in terms of information gain and then resplits the data in the next nodes by the next best feature. The following diagram is taken from the Iterative Dichotomiser 3 (ID3) algorithm Wikipedia page:



The algorithm repeats the process until samples are classified correctly. The deeper the tree, the more perfectly the samples will be classified. However, this can lead to overfitting and our cross validation techniques will attempt to allows just enough depth to prevent overfitting.

Parameters:

    - Max Depth: We will have to test different depths to find the best performer at any given point in time.

- **Logistic Regression:** This type of classifier is a regression model that attempts to classify the data in to specific categories rather than predicting a numerical value, as most other regression models do. The independent variables can be continuous or discrete. The way the model works is by converting the conditional likelihood of a sample being part of a specific class given the attributes, in to a continuous function using a logistic transformation. Once this is done, we can use simple regression techniques to estimate the parameters.

  Parameters:
    - C: Will use a wide array of potential values to find the best fit.

- **GaussianNB:** Naïve Bayes classifier is based on the rules of conditional probability. The fundamental idea is that we can estimate the conditional likelihood of each attribute given a specific outcome or group directly from the data. These likelihoods can then be used to back out the reverse conditional likelihood of a specific outcome given the values of the attributes. One key assumption is that the features are independent of eachother. This is highly unlikely in practice; however, research has shown that decent predictions can be made even when this assumption is violated. We will not need to train any parameters for this classifier.

Other than the choice of classifier, there are also some choices of parameters that are common to all classifiers:

- Length of training window: How many days of historical data to include in our training sample (i.e. 5 days, 10 days, 50 days etc.)
- Number of Companies to include in sample. This is related to our discussion on the trade-off between sample size and number of companies.
- Choice of cross-validation type and folds. We are going to use the 'TimeSeriesSplit' cross validation technique, as it is most relevant to this dataset. The time series split method essentially splits the training set in to different folds, where the individual testing folds are always at a later date than the training counterparts.

Benchmark

The use of metrics and choice of classifiers are merely a means to an end. In this project, the ultimate goal is to discover a profitable trading strategy that can outperform readily available investment vehicles that everyone has access to. Specifically, we want to be able to beat the "market return" that is essentially a

simple buy and hold strategy that many people in the investment world use as a benchmark to compare their own performance to. We will not be using the actual S&P index returns as that may not be a fair comparison, but we will instead use an equal weighted simple average across all companies in our sample (as illustrated in Figure 2). Some summary statistics that we will revisit later:

252 Day Stats	Market
Mean	16.06%
Standard Deviation	19.14%
Sharpe	0.84
Max	110.53%
Min	-44.51%
Beta	1.00
Max Draw Down	-50.82%
Draw Down Date	2009-03-09

These are the benchmark numbers that we will attempt to beat. The next section deals data preprocessing that is required before we can move on with the implementation.

Data Pre-Processing

Since we are going to be training the classifiers across companies, they need to be comparable across the sample. For example, if a particular company is more volatile than all the others, we don't want that company skewing our model fit.

To solve this problem, we can scale each company's return, dividing by the mean of the absolute value of returns during our rolling time period and then subtracting 1. This process essentially assumes that the mean return for each company is 0, and then divides by the mean absolute deviation from that mean return. For example, lets take a look at Apple's returns for a 20-day window, before and after the transformation:

Table 2: Apple (AAPL) Before and After Transformation				
2007-12-17	-0.031462	Divide by mean of absolute values = 0.0229 and subtract 1 →	2007-12-17	-2.372326
2007-12-18	-0.007701		2007-12-18	-1.335893
2007-12-19	0.000765		2007-12-19	-0.966627
2007-12-20	0.022335		2007-12-20	-0.025770
2007-12-21	0.035789		2007-12-21	0.561062
2007-12-24	0.025218		2007-12-24	0.099973
2007-12-26	0.000755		2007-12-26	-0.967087
2007-12-27	-0.001910		2007-12-27	-1.083313
2007-12-28	0.006345		2007-12-28	-0.723224
2007-12-31	-0.008757		2007-12-31	-1.381991

2008-01-02	-0.016357		2008-01-02	-1.713473
2008-01-03	0.000462		2008-01-03	-0.979850
2008-01-04	-0.076335		2008-01-04	-4.329651
2008-01-07	-0.013385		2008-01-07	-1.583845
2008-01-08	-0.035972		2008-01-08	-2.569041
2008-01-09	0.047591		2008-01-09	1.075874
2008-01-10	-0.007692		2008-01-10	-1.335527
2008-01-11	-0.029940		2008-01-11	-2.305969
2008-01-14	0.035266		2008-01-14	0.538243
2008-01-15	-0.054480		2008-01-15	-3.376374

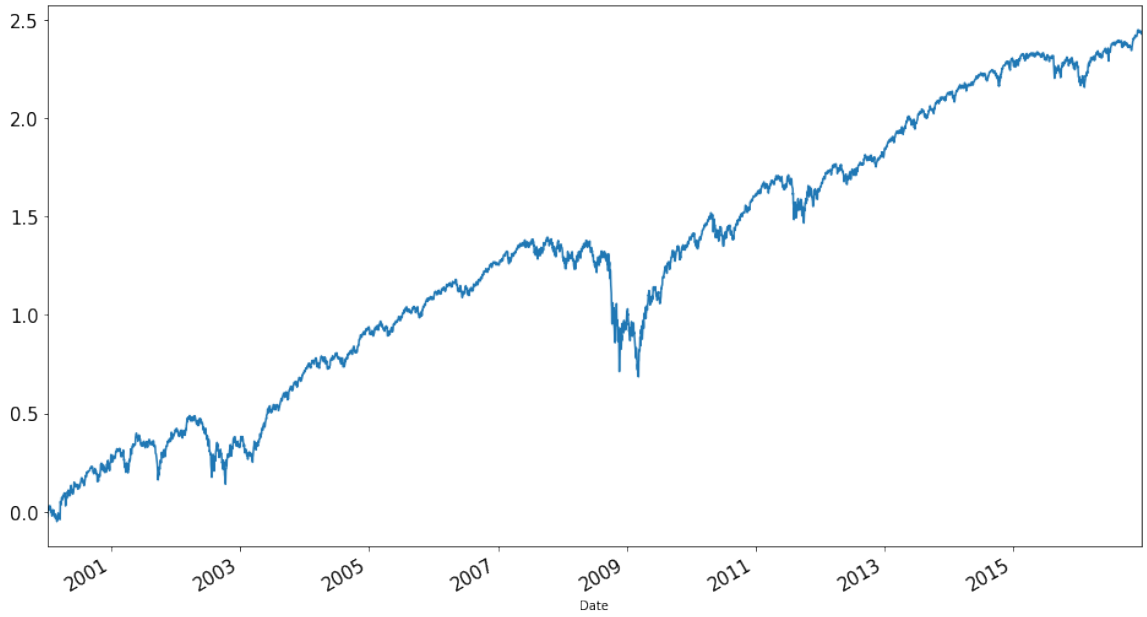
These values would be part of the training window that would be used to predict the direction of the stock for the next day, in this case Jan 16th, 2008. The same process can be applied to standardize the sector returns, by first finding the equally weighted average return across all companies in a particular sector, and then applying the transformation process described above. For volume data we can use the same transformation as for the returns.

The final pre-processing step is dealing with missing data. Missing data is an inevitable part of working with market data and there is never a 'correct' way of how to deal with it. Given our relatively large sample size, we can afford to fill missing values with 0. This assumes that the stock price did not change over the missing value period. For volume, this implies that a stock did not trade on that specific day.

Implementation

The first step is to decide on the choice of sample size keeping in mind the tradeoff between number of companies and number of years of history. In this case, we start with a start date of January 2000 to the latest available date, December 2016. This sample includes 398 companies as well as two full market cycles. The following graph shows the cumulative compounded return for that period of time:

Figure 5: Market Cumulative Return (Log Scale) : 2000-2016



This period consists of two significant market crashes (2002 and 2008) as well as sustained market rallies and should provide a good testing period for the strategy.

The approach we will take now is to select a smaller subset of this sample to train and optimize the classifiers and then test the model on the remaining sample. In addition, there are a few other parameters that will need to be selected and optimized in the model selection process:

- Length of the rolling window to train the classifier
- Choice of classifier
- Optimization of parameters specific to each classifier
- Total length of training window

The full process is illustrated below, along with the function names used in the IPython Notebook:

Essentially, this process attempts to find the best fit model at any given date and used it to make a prediction for the following day. As a result, the specific parameters chosen for a particular type of classifier can vary over time. For instance, the best decision tree model at January 4th, 2000 may have a max depth of 2 while a month later the best one may have a depth of 10. What we are most concerned with, is maintained a significant CV and average test score over time.

Some complications with this nested process is ensuring that we don't accidently use any data that was not available at a given point in time. For instance, if we are making predictions for January 2nd, our training window should not use any data that was not available as of the end of the trading day on January 1st. Since our model uses the 1-day ahead forward return to train the classifier, our training set

should set the cut-off date at December 31st to fit the classifier, then make predictions using the available data points on the 1st.

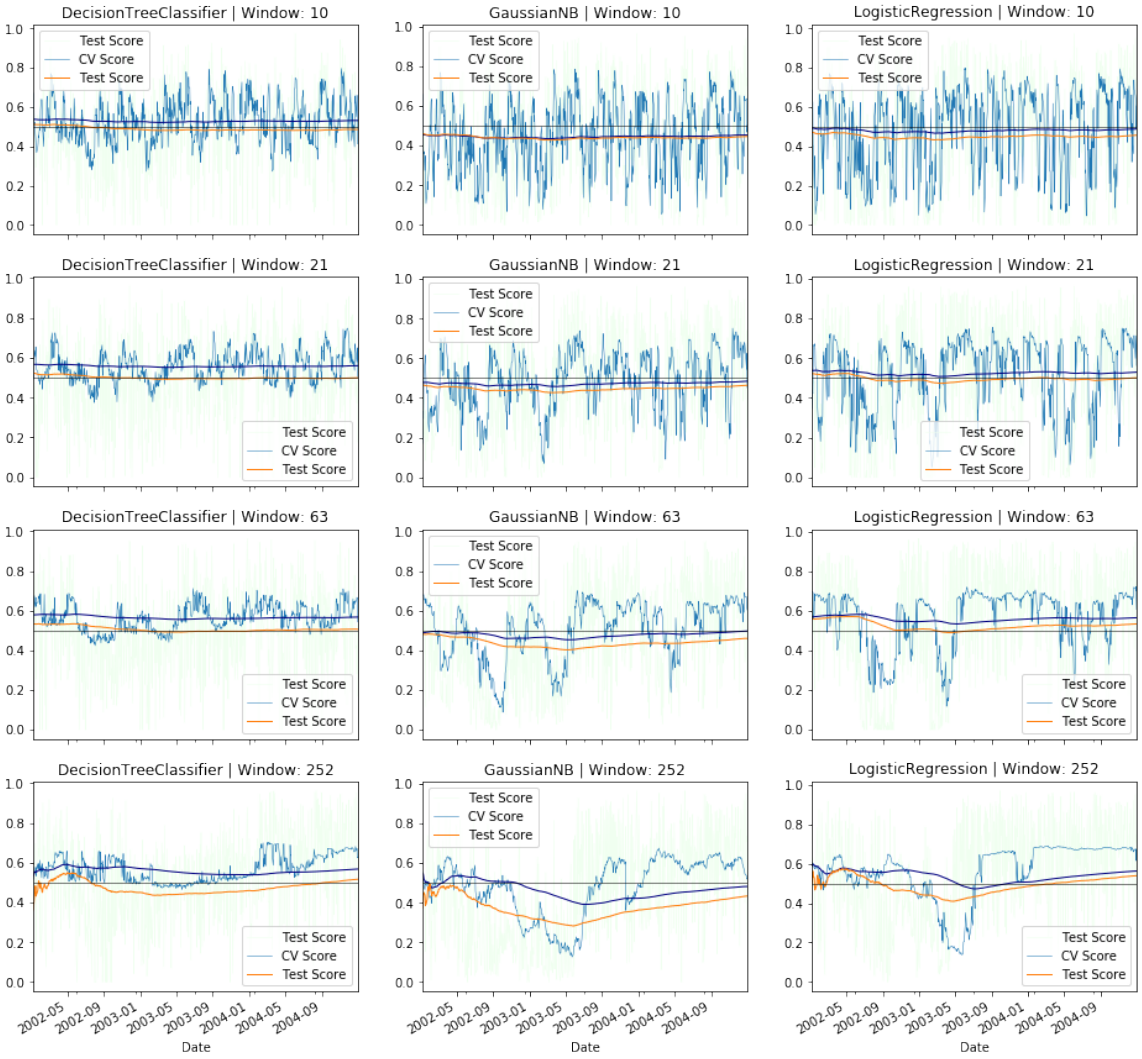
The implementation process discussed is quite a lengthy process and the number of combinations grows exponentially if too many parameters choices are passed. Consequently, the real challenge is to provide enough freedom in terms of parameter variability without slowing the process down too much. To start off, the following parameter choices are used:

Parameter	Parameter Values
Training Window Size	10 days 21 days 63 days 252 days
Classifier Type	Decision Tree Logistic Regression GaussianNB
Training Period	Start: 2000-01-31 End: 2004-12-31

The results of this initial test are discussed in the next section, along with some techniques to refine and improve upon them.

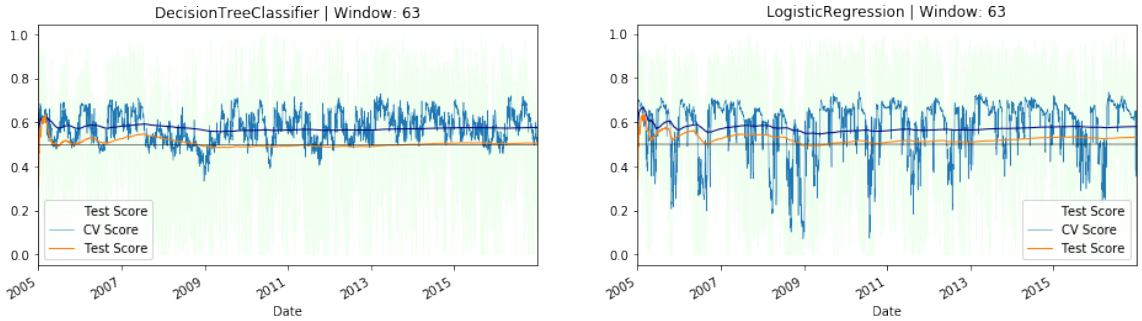
Refinement

The following graphs show the detailed results for cross validation scores from the training sets and the test scores for each date:



The navy blue line is the expanding mean of the CV score while the solid green line is the expanding mean of the test score. Of all the different models tested, what we are looking for is a high and consistent CV score as well as a test score that is above 0.5. This criteria is consistent with our choice of metrics discussed earlier on in the report. Of the models that meet this criteria, the best ones appear to be the Decision Tree and Logistic Regression with a 63 day training window. Note that we are not particularly concerned with what the specific parameters (max depth and 'C') are at any given point in time, since they will vary based on the best fit at that particular date.

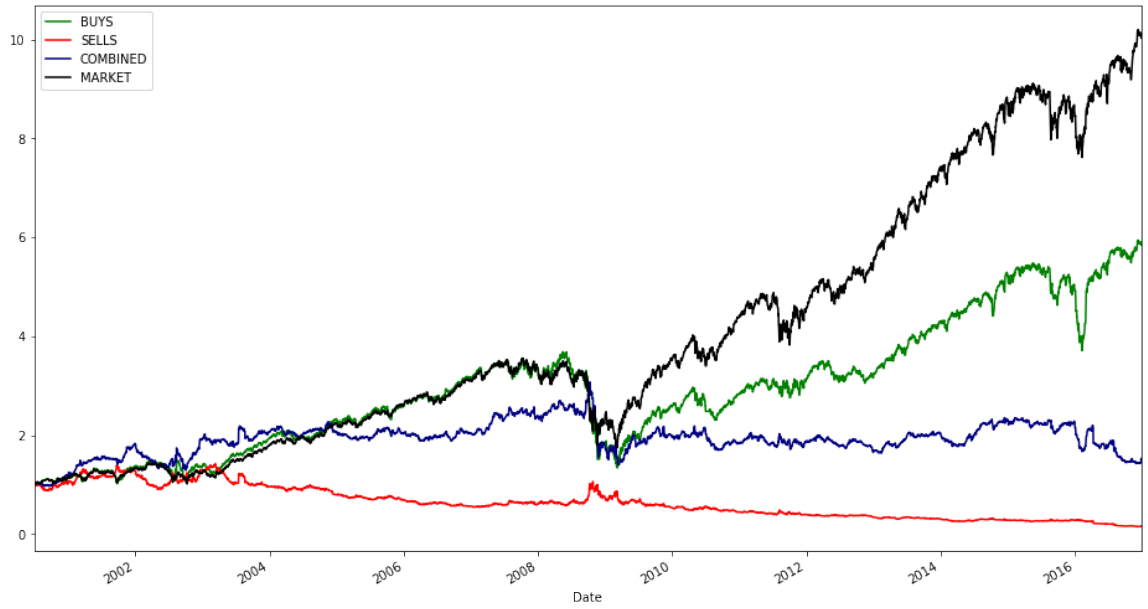
The next step is to repeat the exercise for these two models on the entire sample (2004-2016) and see what results we get. The following figures plots the CV and test scores for both models selected:



Of these two, the Decision tree model appears to be more consistent and we can expect it to perform better. In fact, a trading strategy that was formed on this model performs better as illustrated in the following two figures:

Figure 6: 63 Day DecisionTree - Cumulative Returns

Figure 7: 63 Day LogisticRegression - Cumulative Returns



The green and red lines represent the returns achieved by acting on the model predictions for up and down predictions respectively. The navy line is a combined portfolio while the black line is the market return (i.e. average daily return across all companies). The decision tree model performs better than the logistic regression, however, the sell predictions are just as poor. As a result, our final model will ignore the sell predictions and only focus on the buy predictions.

Model Evaluation and Validation

The final model we use is a long only Decision Tree model with a 63 day rolling window. The long only implies that we will only be buying stocks and will not be shorting (or selling) them in order to profit from downward movements. To summarize, the final model has the following characteristics:

Model Summary	
Classifier Type	Decision Tree
Classifier Parameters	Estimated Based on best fit in rolling window
Rolling Window Size	63 Trading Days
Cross Validation Type	Timeseries split (4 folds)

Based on the results from the previous section, we can see that the buy predictions by themselves do end up outperforming the market, as illustrated by the performance statistics below:

252 Day Stats	Decision Tree Buys	Market
Mean	17.37%	16.06%
Standard Deviation	17.54%	19.14%
Sharpe	0.99	0.84
Max	92.35%	110.53%
Min	-41.32%	-44.51%
Beta	0.94	1.00
Max Draw Down	-47.34%	-50.82%
Draw Down Date	2008-11-20	2009-03-09

The strategy has a better average annualized return of 17.36% compared to the market return of 16.46% while also maintaining a Sharpe ratio of 0.99 compared to 0.87 for the market. Furthermore, the strategy also has a better max draw down. As a final step, we can also attempt to see whether we can improve on this strategy by only trading when the CV Score is above a certain threshold. The underlying assumption is that periods of uncertainty tend to be persistent. For instance, we can trade using the model predictions when the 63-day rolling mean of the CV Scores is above a threshold of 0.525 and do nothing when it is below that level. The results from this slight modification to the model can be seen below:

252 Day Stats	Threshold Model	Market
Mean	13.78%	16.06%
Standard Deviation	12.3%	19.14%
Sharpe	1.11	0.84
Max	40.85%	110.53%
Min	-25.71%	-44.51%
Beta	0.49	1.00
Max Draw Down	-28.75%	-50.82%
Draw Down Date	2002-07-23	2009-03-09

At first glance this might seem like a worse model, as the average annualized return is lower. In the next section, we will discuss how leverage can be used to build an even more superior model that beats the benchmark in all categories.

Justification

The final version of the model presented in the previous section is capable outperforming the benchmark across by leveraging the strategy 1.5X. This essentially means we borrow money to magnify the profits, and losses, with the hope of making a higher return over time. The Threshold model with leverage has even better results:

252 Day Stats	Threshold Model (1.5X)	Market
Mean	20.94%	16.06%

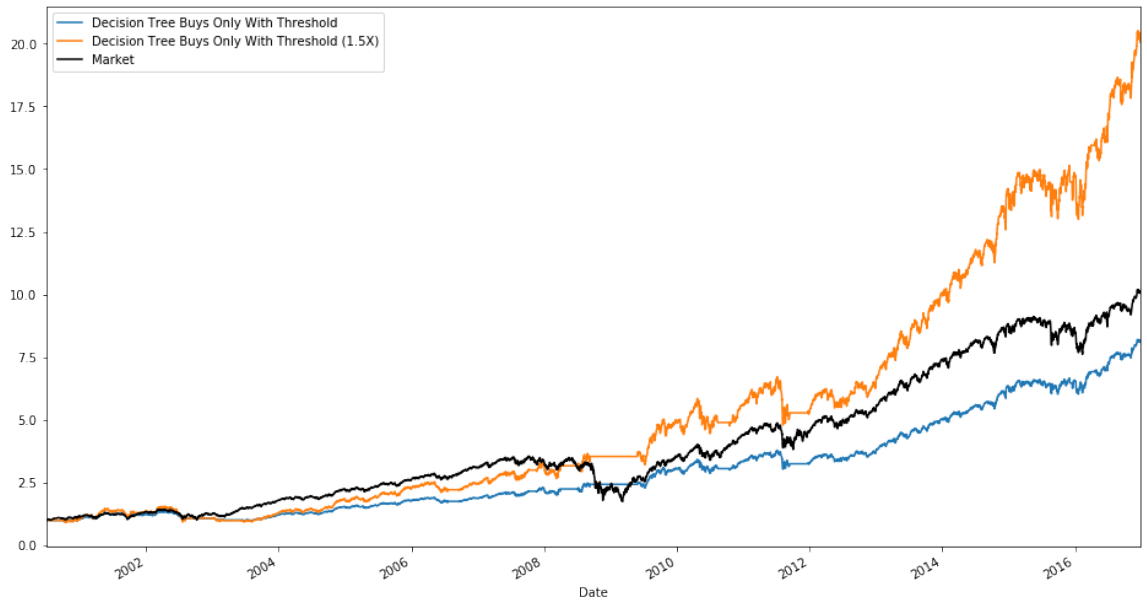
Standard Deviation	19.3%	19.14%
Sharpe	1.08	0.84
Max	65.13%	110.53%
Min	-36.64%	-44.51%
Beta	0.73	1.00
Max Draw Down	-40.18%	-50.82%
Draw Down Date	2002-07-23	2009-03-09

As seen in the table above, it has a higher average annualized return of 20.94% compared to the benchmark at 16.06%; more importantly, it still has a Sharpe ratio of 1.08 compared to the 0.84 of the market. The maximum drawdown is also significantly better at 40%, considering that we are also using borrowed money to magnify the returns. Overall, we can say that this final version successfully beats the benchmark established earlier, both in terms of annualized return and consistency of those returns.

Free-Form Visualization

Figure 8 shows the comparison of versions of the model discussed in the previous section:

Figure 8: Model with CV Threshold Comparison



This picture illustrates the difference between the models and benchmark (black). The visualization helps demonstrate how slight improvments in the performance metrics can compound over time to yield significant differences in portfolio value. A portfolio that was invested in the threshold model with 1.5X leverage would be worth twice as much as a portolio invested in the benchmark, with similar volatility

of returns. Furthermore, we can see the drastic differences during market crashes such as in 2008, where the strategy is actually flat while the market goes through a significant correction.

Reflection

This exercise has been lengthy but quite enlightening. Starting with quite a few models and parameters to choose from, we were able to narrow down a decision tree model with a 63-day rolling window as the one with the best chance at improving upon the overall market return. Even though the buy predictions appeared to perform well, the sell positions did not. Furthermore, we learnt that by restricting trading to only those days where the mean cross validation scores had been above a certain threshold in the recent past, we could improve the performance of the model when used in conjunction with leverage to magnify the returns. Despite this, there is still quite a lot of room for improvement.

Improvement

First, in order to truly have an unbiased estimate we require a point in time database. This means we need to have data for companies that were also removed from the index due to bad performance or bankruptcy. It is possible that our sell predictions would do a lot better if these companies were part of the sample. The impact of this survivorship bias is to bias the performance of buy positions upward while negatively affecting the performance of sell positions.

Second, we should have more features to group the different companies. In this project, we only used sector information. However, there is a lot of other fundamental data, such as financial statement data that can be used to distinguish between high growth, value or dividend paying companies. This can help to improve upon our predictions by grouping more relevant companies together.

Third, an ensemble of the different models could potentially perform better than just picking the top model. In this specific case, it is possible if we took all 12 of the models from Appendix I and created an ensemble of all of those, that we could beat our final model.

Finally, the returns in this case are free from transaction costs and in the real world trading costs can be quite significant. Since we are only trading some of the largest and most liquid stocks in the overall market, we should still be able to maintain our profitability. In this particular exercise, the average of the absolute value of daily returns for the strategy was around 0.9% while transaction costs are usually significantly less than that if trading with enough volume.

In conclusion, all these points can be used to improve upon the performance of the strategies that were discovered in this project. It also goes without saying that historic backtests do not imply that future performance will be similar.

More research is required to generate better trading strategies, as the potential gains are definitely worthwhile.