# Definition

*(approximately 1 - 2 pages)*

### Project Overview

### The Stock Market is a large collection of markets and exchanges where equities of publicly held companies can be traded. Trading in stocks has many benefits, and can be a lucrative income if the market is well understood. Companies are able to gain access to capital by selling of slices of ownership to investors, and the investors have the opportunity to gain income and assets. The Stock Market can also be one of the better predictors in determining the health and direction of an economy. It can help predict if economic and political policies are paying off, whether or not housing prices are going to rise, and influence the size of a nation’s workforce [1].

### For these reasons, predicting the Stock Market can be very advantageous. The issue is the market is very volatile and influenced by a large number of factors. Many argue that the market cannot be predicted, and in a strong definition that is likely correct. However, the market often does have concrete trends that can be analyzed, and therefore may still be able to be reasonably predicted in short periods of time.

In this project, I am going to combine Machine Learning concepts with a technical analysis of individual stocks from the NYSE or NASDAQ (American Stock Markets) in an attempt to predict their stock prices. I will report the stock predictions over a 14-day period in intervals of 7 days. Ex: 1 day out, 7 days out, 14 days out from the current date. All data sets were obtained from Yahoo Finances [6]. The final implementation will be in the form of a web application that allows users to select up to 4 stock symbols, and the application will then output the predicted results and analysis of each stock.

# My prior knowledge of the Stock Market was minimal, which was also a driving factor in my personal reasoning for taking on this project. Much of my understanding and knowledge of the Stock Market and how to interpret it through technical analysis came from Udacity’s course Machine Learning for Trading [2], Khan Academy’s overview of Stocks and Bonds [3], and the book “How to Day Trade for a Living: Tools, Tactics, Money Management, Discipline and Trading Psychology” [4].

### Problem Statement

### The goal of this project is to reasonably predict the price of individual stocks in the American Stock Market over a short period of time in the future, 14-days, with the expectation that the 14th day will be less accurate than the 1st. The following outlines the steps involved in accomplishing the task:

### Obtain data sets of 4 individual stocks and the S&P500 (explained in detail in coming sections) from Yahoo Finances.

### Process the data to ensure it includes any missing information, and does not include any dates the market was not trading.

### Analyze each of the stocks individually, compare them against the S&P500, and determine the best features to utilize in my predictions.

### Train the data against a few classifiers to determine which delivers the best results.

### Tweak the results of the classifiers if necessary.

### Develop a web application that allows a user to select up to four stocks, run the stocks through the classifier, and present them an analysis of the stocks with predictions based on my chosen classifier.

### The web application will only be intended only for educational purposes, and not to be utilized in actual market trading.

### Metrics

In order to provide metrics for my solution I will be using the coefficient of R^2 (R squared) of the prediction. R^2 is used in models with the purpose of predicting future outcomes. The coefficient R^2 is defined as:

*(1 - u/v), where u is the residual sum of squares, and v is the total sum of squares* [7].

### Given all final predictions in my model occur in future dates, the metrics will be calculated by backtesting and scoring how accurately the classifier was able to predict historical dates.

# Analysis

*(approximately 2 - 4 pages)*

### Data Exploration

### During the data exploration phase of the project multiple sets of historical stock data was obtained. The data was downloaded from Yahoo finance in the form of CSV files. The reason for downloading the files in the beginning stages, instead of using the Yahoo API utilized in the final implementation, was to ensure any procedures were reproducible and not affected by stock market factors such as stock splits. The data sets used during this phase are included in the final project files, so any procedures can be reproduced in the future.

### There are two main components needed for data exploration to satisfy the problem domain; A date range to gather historical values, and stock indexes that represent a company traded in the American Stock Market (NYSE or NASDAQ) on those dates. To satisfy the former the dates from July 7th, 2015 – July 7th, 2017 where chosen (At the time July 7th was the most recent trading date). This results in 2 years of historical data. To satisfy the latter stock indexes for Google (GOOG), Apple (AAPL), IBM (IBM), Nvidia (NVDA), and the S&P 500 (SPY) were chosen. Google, Apple, IBM, and Nvidia where chosen at random, and four indexes were chosen since that will be the max allowed indexes in the final implementation of the user interface. The S&P 500, however, is a unique stock market index in that it is a weighted representation of around 500 American Stock Market indexes with the nice side effect that any null values in the data set will represent days not traded. By removing these dates, we are left with only dates the stock market was open and trading. This is useful information as it helps determine true null data values in other stocks such as dates it may not have existed, dates when the stock market was trading but the stock was not, or just missing data. For this reason, the S&P 500 has been chosen as a benchmark index to evaluate all stock data against.

### When retrieving data from Yahoo Finances multiple features of the stock are returned. These features include the chosen stocks “Open”, “High”, “Close”, “Adjusted Close”, and “Volume” and are indexed by the dates traded. The below graph is a visual representation from the S&P 500 data.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| SPY | | | | | | |
| Date | Open | High | Low | Close | Adj Close | Volume |
| 2010-07-15 | 109.610001 | 110.059998 | 108.169998 | 95.017952 | 109.680000 | 232337900 |
| 2010-07-16 | 109.089996 | 109.209999 | 106.449997 | 92.401672 | 106.660004 | 282693400 |
| 2010-07-19 | 107.050003 | 107.629997 | 106.220001 | 92.947449 | 107.290001 | 186709000 |
| 2010-07-20 | 105.870003 | 108.559998 | 105.820000 | 93.978371 | 108.480003 | 258162400 |
| 2010-07-21 | 109.040001 | 109.070000 | 106.629997 | 92.756844 | 107.070000 | 264527000 |

### The above data example represents various features of the S&P 500 over a 5-day period. They each have their own significance depending on what is trying to be determined, but ultimately, they mostly describe the price of a given Stock at different portions of the day. In the defined domain problem of this project, our target is the Adjusted Close prices of each day since our ultimate goal is to predict future Adjusted Close prices, and the other data pieces are not useful features in determining this. Therefore, the other data features will be discarded and not used in our exploration and implementation. This is an interesting problem and divergence from many other data sets used for predictions. After dropping the other data features we are left only with our target values and no features in the given set to learn from. The solution will be to derive new features that are calculated from basic statistics and the previous days Adjusted Close. A few of these new features that will be processed and explored are the Standard Moving Average, Momentum, and the Bollinger Bands ratios . In upcoming sections, there will be discussion on how these features were processed and used, but for now they will only be defined.

### Simple Moving Average (SMA): The SMA is derived by calculating the mean of the data over intervals. For example, an interval may be every 20 days, and the SMA for a particular day would be represented by the mean calculated from the previous 20 days. This is also referred to as the Rolling Mean. In order to use this as a technical analysis we use the following equation to obtain values around -0.50 to 0.50.

### *Simple Moving[day] = (price[day] / price[day - interval] .mean()) - 1*

### Bollinger Bands (BB): BB bands are simply the upper and lower bands created when calculating data points two standard deviations above and below the SMA. In order to use this as a technical analysis we use the following equation to obtain values around -1.0 to 1.0.

### *BB[day] = price[day] - sma[day] / 2 \* std[day]*

### Momentum: Momentum gives directionality to a stock over a given interval. It indicates if the stock has a positive momentum (going up), or a negative momentum (going down). In order to use this as a technical analysis we use the following equation to obtain values around -0.50 to 0.50.

### *momentum[day] = price[day] / price[day - interval] - 1*

### Now that we have our feature set the below table gives an example of what it would look like for the S&P500.

|  |  |  |  |
| --- | --- | --- | --- |
|  | BB | Momentum | SMA |
| 2015-07-28 | -0.305356 | -0.006408 | -0.025181 |
| 2015-07-29 | 0.053972 | 0.000760 | -0.018476 |
| 2015-07-30 | 0.108938 | -0.006971 | -0.018243 |
| 2015-07-31 | 0.082941 | -0.009319 | -0.019733 |
| 2015-08-03 | -0.045573 | -0.013171 | -0.023039 |

### Given the high volatility of the Stock Market it was decided it was not necessary to locate or remove outliers. In this situation, what may be seen as an outlier is still valuable information in understanding how a particular stock is behaving and therefore is not truly an outlier.

### The below table list the basic statistics of the data sets used. The adjusted closing price has been normalized so each stock is evaluated on a level playing field. This allows for a more accurate representation of how the stocks are behaving compared to each other.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Mean | Median | Standard Deviation |
| SPY | 1.021891 | 1.008686 | 0.072933 |
| NVDA | 3.246513 | 2.654098 | 1.982375 |
| IBM | 0.910470 | 0.909703 | 0.078761 |
| GOOG | 1.357082 | 1.347870 | 0.154974 |
| AAPL | 0.927906 | 0.899809 | 0.134810 |

### Of these basic statistical measurements, the Standard Deviation holds a lot of value. Higher values indicate the stock is more volatile which could result in less certain predictions. As we can see NVDA has a much larger standard deviation than the other stocks.

### Exploratory Visualization

### The follwing graph is a visualization of the Rolling Mean and the Bollinger Bands. The visualization is useful to see how they relate to the Adjusted Close values over a given period of time.

### SMA/Bollinger Bands of SPY

### 

### The following graph illustrates the correlations between the selected features. If a feature is correlated to another feature it would be visualized in a linear fashion. If two features are highly correlated they may not offer unique enough information to the final model. Based on the visualization it would appear the features are unique enough to be useful.

### Feature Correlation Based on SPY

### 

### Algorithms and Techniques

### The problem is a Supervised Learning problem given we are attempting to predict future stock prices based on learning and training from historical data. The continuous nature of the data makes it particularly suitable for regression type algorithms. After evaluating the types of algorithms available I have decided to use an Ordinary Least Squares Linear Regression algorithm, the K-Nearest Neighbor Regressor algorithm, and a Random Forest Regressor algorithm to evaluate which returns the best results. Each of these algorithms are available in the Scikit Learn Python library.

### Linear Regression: In this model, one technique that will be utilized is to pass the model what is known as a Standard Scalar. The idea of the is to bring each of the features passed to the linear regression within the same numerical scale as well as try representing the features as a normal distribution. The goal of the Linear Regression model is to fit a line to the data that can best generalize any trends. The output of the model will be an equation that can then be used to make the future predictions.

### K-Nearest Neighbor Regressor: The K-Nearest Neighbor Regressor is a regression variant of the class base K-Nearest Neighbor algorithm. In this model, the same Standard Scalar technique utilized in the Linear Regression model will be utilized. One of the fundamental differences in this model is the historical data the model is trained on needs to be retained in order to make future predictions, whereas in Linear Regression the historical data is no longer needed once the model equation is obtained. The idea behind this model is it will use a variable amount of the closest historical values, take their average, and use that to make its predictions. Different variable amounts will be tested to see what number of neighbors return the best results. The weights parameter will also be set to “distance” so that closer query points will have greater influence than those further away.

### Random Forest Regressor: A Random Forest Regressor is a model that fits multiple Decision Trees and uses averaging to improve predictive accuracy and control over-fitting. In this model, we will not need to use the Standard Scalar used on the previous models because the decision-making process is not affected by features of differing scales. The only parameter that will be tuned is the n\_estimators parameter which determines the number of trees the model will use.

### One technique that will be utilized for all three models is Time Series Splitting for cross-validation. Cross-validation is a useful technique for training and testing a model by utilizing all the available data for testing and training. The goal is to reduce overfitting. However, traditional cross-validation does not preserve data ordering in the process which is important in our problem domain. Time Series Split will preserve the ordering solving this issue.

### Benchmark

### As mentioned earlier the Benchmark for the model will be the S&P 500. The S&P 500 is a weighted representation of around 500 American Stock Market indexes and attempts to capture the market as a whole. By using this as the Benchmark we can properly prepare the data for our predictions, and ensure we are not using erroneous values generated from non-trading days. We can also visualize our data against the S&P 500 along the way to evaluate how each stock is doing in comparison, and we can use it to help determine if our features are appropriate. In the end, we can attempt to predict the S&P 500 along with the other stocks and see if it follows a similar trend.

# Methodology

*(approximately 3 - 5 pages)*

### Data Preprocessing

### During the preprocessing stage I began by pulling data for the S&P500, and used it as a benchmark for finding null values in other stocks. The following steps were taken:

### Decide on a historical date range to explore. The dates July 7th, 2015 – July 7th, 2017 where used in the analysis process.

### Gather data on the S&P500 based on the chosen dates.

### Gather data on 4 other stocks based on the chosen dates. GOOG, NVDA, AAPL, and IBM where used during this phase.

### Drop all data values except for the Adjusted Close values.

### Drop all dates in all chosen stocks where the S&P500 is null. These represent dates the stock market was closed.

### Locate null values in the other stocks and fill in values forward. This means taking the last known value that was not-null, and filling that value forward until you reach the next non-null value.

### Locate any more null values in the other stocks and fill backwards. If a particular stock has null values at the start of the date range, find the first non-null value and use that value to fill back to the start of the date range.

### At this point our data is ready to be explored. There is no need to search for outliers, as all historical data values represent the nature of the stocks.

### Now that the data has been pre-processed the next step is to calculate the desired features from the Adjusted Close values. The technical indicators chosen for the purposes of this project where the stocks Momentum, the Bollinger Band values, and Simple Moving Average values. These indicators were talked about more thoroughly in the Data Exploration section.

### The features are not on the same scale and so they were scaled using Scikit Learn’s Standard Scalar. This was required for the Linear Regression classifier and the KNN classifier.

### Implementation

Once the data was fully processed and features were evaluated I decided to run the data through 3 different types of classifiers to determine which gave the best results. The algorithms chosen were a simple Linear Regression, A KNN Regression, and a Random Forest Regression.

To begin I ran the data and the classifiers through the Scikit Leanr Pipeline. A pipeline is a chaining of estimators applying a fixed sequence of steps. It was used to first scale the features before running the code through the classifier. This was used for the Linear Regression classifier as well as the KNN classifier. There was no need to use it for the Random Forest as that particular classifier does not benefit from scaling of data.

A common procedure in training a classifier is cross-validation (CV) with K-folds being one of the more common CV techniques. In K-folds you split the data into k smaller sets, then use k – 1 folds as training data, validate on the remaining data, and then repeat until all the folds have been used for training and testing. However, this technique does not work in situations where the order of the data is important such as a time series. In order to achieve similar effects, a technique known as a Time Series Split was used. In each split of a Time Series Split CV, the test indices are higher than the previous split preserving order while still maximizing the data used for training and testing.

The CV was set to N splits, and therefore the data was put through N sets of training and testing for each algorithm. The R^2 scores for each split was collected and overall mean score was calculated as the metrics value. The score was calculated for the training sets as well as the testing sets to determine if there was any overfitting.

During the final split, where the training set was the largest, the predicted values and actual values where plotted on a scatter plot and fit with a linear line as a visual metric.

Once the final classifier was decided upon, the final application was built using the Python web development micro framework Flask for the server side code, Angular JS to handle the front-end code presentation, and D3.js for data visualization. During each load of the application, a user selects up to 4 stock symbols to explore, then each stock is trained on the classifier, and the user is presented with the predictions and analysis of each stock.

### Refinement

### After the first few runs of training the classifiers the results were not as expected. The KNN classifier and the Random Forrest Classifier performed very poorly. The Linear Regression classifier, however, had pleasing results. I decided to move forward with the Linear Regression classifier and see if it could be improved upon.

### Over all the scores of the Linear Regression where considered acceptable compared to the metrics, so refinement strategies were minimal. The first parameter that was tested was the number of splits in the Time Series Split CV. After multiple trial and error attempts setting the number of splits to 3 had good results while still performing at reasonable speeds.

### A few other areas that were tweaked to increase performance did not relate directly to the classifier itself, but how the data was processed, and the calculation of the features. Multiple date ranges were testing to see what was the least amount of data needed to increase speed and still obtain good metric scores. Ultimately, I settled on 1 year of data to be analyzed during the classifier training.

### During feature selections, a portion of the data gets lost in order to calculate the Simple Moving Average, Momentum, and Bollinger Bands. The reason for this loss in data is it requires X amount of days to calculate the first usable day of data. This was referred to as the window size in the documentation and project code. The window size was adjusted to see if it had an effect on the final results. The final chosen window size was 10 days.

# Results

*(approximately 2 - 3 pages)*

### Model Evaluation and Validation

In this section, the final model and any supporting qualities should be evaluated in detail. It should be clear how the final model was derived and why this model was chosen. In addition, some type of analysis should be used to validate the robustness of this model and its solution, such as manipulating the input data or environment to see how the model’s solution is affected (this is called *sensitivity analysis*). Questions to ask yourself when writing this section:

* *Is the final model reasonable and aligning with solution expectations? Are the final parameters of the model appropriate?*
* *Has the final model been tested with various inputs to evaluate whether the model generalizes well to unseen data?*
* *Is the model robust enough for the problem? Do small perturbations (changes) in training data or the input space greatly affect the results?*
* *Can results found from the model be trusted?*

### Justification

In this section, your model’s final solution and its results should be compared to the benchmark you established earlier in the project using some type of statistical analysis. You should also justify whether these results and the solution are significant enough to have solved the problem posed in the project. Questions to ask yourself when writing this section:

* *Are the final results found stronger than the benchmark result reported earlier?*
* *Have you thoroughly analyzed and discussed the final solution?*
* *Is the final solution significant enough to have solved the problem?*

# Conclusion

*(approximately 1 - 2 pages)*

### Free-Form Visualization

In this section, you will need to provide some form of visualization that emphasizes an important quality about the project. It is much more free-form, but should reasonably support a significant result or characteristic about the problem that you want to discuss. Questions to ask yourself when writing this section:

* *Have you visualized a relevant or important quality about the problem, dataset, input data, or results?*
* *Is the visualization thoroughly analyzed and discussed?*
* *If a plot is provided, are the axes, title, and datum clearly defined?*

### Reflection

In this section, you will summarize the entire end-to-end problem solution and discuss one or two particular aspects of the project you found interesting or difficult. You are expected to reflect on the project as a whole to show that you have a firm understanding of the entire process employed in your work. Questions to ask yourself when writing this section:

* *Have you thoroughly summarized the entire process you used for this project?*
* *Were there any interesting aspects of the project?*
* *Were there any difficult aspects of the project?*
* *Does the final model and solution fit your expectations for the problem, and should it be used in a general setting to solve these types of problems?*

### Improvement

In this section, you will need to provide discussion as to how one aspect of the implementation you designed could be improved. As an example, consider ways your implementation can be made more general, and what would need to be modified. You do not need to make this improvement, but the potential solutions resulting from these changes are considered and compared/contrasted to your current solution. Questions to ask yourself when writing this section:

* *Are there further improvements that could be made on the algorithms or techniques you used in this project?*
* *Were there algorithms or techniques you researched that you did not know how to implement, but would consider using if you knew how?*
* *If you used your final solution as the new benchmark, do you think an even better solution exists?*

Bibliogorpahy

<https://www.thebalance.com/economic-indicators-are-your-secret-weapon-2637037>

7 http://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsRegressor.html