# Capstone 1 Milestone

1. This project has proven to be a very fruitful research exercise in the areas of portfolio optimization.  The initial problem was one of prediction and forecasting. How does one successfully predict price movements of a particular company or set of companies and even more so, an industry given another industry? This is a question that is most relevant to hedge funds and just as well government agencies who, at the very least, may wish to entice their employees with the perks of investing their wages in stable securities. Likewise, any given company may wish to understand its market capacity within its own industry, its profitability trends, and its competitors price behavior. Accurate forecasting can potentially lead to monetary gains or protect a company from systemic risk. In general, the study of market prices can be quite useful for risk assessment as well as profit discovery. The existence of hedge funds is just an example. The initial question will more than likely be reduced to a much simpler one of how to properly predict a single company's prices within the loosely based guidelines of a portfolio optimization methodology.

   An important note regarding portfolio optimization, the methodology is rather simple and a whole lot of algorithms to carry out the process already exist in python. In other words, the process was made rather simple since all of the necessary items for research and analysis of stock prices in this fashion already exist.

The main reason and most useful application of portfolio optimization was in the process of stock selection. When dealing with stocks from a particular industry, the analyst needs to reduce the size of stocks being dealt with and needs to do it in a reasonable way. A good rule of thumb and probably the only rule is to use statistically based methods.This is the methodology used in portfolio optimization for stock selection.

All of the python packages used in the project for portfolio selection were taken from DataCamp's Portfolio Analysis course. An important note regarding DataCamp, although DataCamp is an incredibly useful tool for implementation and understanding more difficult tasks in statistical and machine learning, it by no means serves as a sufficient source of information on these matters. Mathematics is rarely explained, explained in depth or even covered and coding implementation can often be rather basic. Often times, you may find that it is the only source for learning how to code algorithms or concepts that may otherwise be difficult to approach. DataCamp is a great resource and launching point. For that reason, it is valuable. I found myself returning to it often when searching for appropriate literature in pursuit of my project goals.

2. To speak more concretely about the process of portfolio selection, we will assume the data has already been acquired and is in a dataframe. It's a two step process. We will calculate returns with the appropriate python method and use returns for the analysis. We'll then do statistics with a focus on screening returns based on their normality. We want returns with a kurtosis less than or equal to 3

and skew between -0.5 and 0.5. That's it for stats. The next and last step is the sortino ratio. This is variant of the sharpe ratio. These ratios give you an idea of the risk to reward tradeoff where the goal is to achieve a higher reward. So we are always looking for larger positive ratios. Sortino uses the standard deviation of negative returns rather than all returns. This is the part of the ratio and formula that's appealing. The advantage here is that it's focus is on negative volatility of returns whereas the sharpe ratio is concerned with both positive and negative volatility, which might be considered a disadvantage. Essentially, there is no penalty for large positive swings in return values. When this calculation is complete, order your tickers in descending order. We want the most positive values. Select however many tickers you wish to become a part of your study. A crucial note: individual tickers were treated as individual portfolios. Technically, a portfolio is a set of stocks or tickers. Weights are normally assigned to each stock to indicate the quantities invested in each stock so that the sum of the weights is 1. To prevent arbitrarily initializing weights, this detail was overlooked. It was overlooked despite Markowitz Efficient Frontier Optimization technique, which minimize variance of the portfolio weights. Altogether, the issue was deliberately overlooked because it is currently an open problem in portfolio analysis and requires quite a lot of time for experimental design and validation. In addition, the goal of this project is not necessarily to maximize profit but more so to understand volatility as it relates to forecasting techniques especially while our forecasting is constrained to univariate time series. The traditional way is to minimize variance of a portfolio Sharpe/Sortino Ratios with the Efficient Frontier approach but we did not use weights to indicate how much of each stock we invested in. According to Modern Portfolio Theory, the traditional approach to optimization is to minimize portfolio variance using weight but we did not use weights to indicate how much of each stock we invested in. Below is an image of the Efficient Frontier with tangents representing optimal portfolios along the utility curve where risk and reward are tradeoffs on the x and y axis, respectively.

The theh

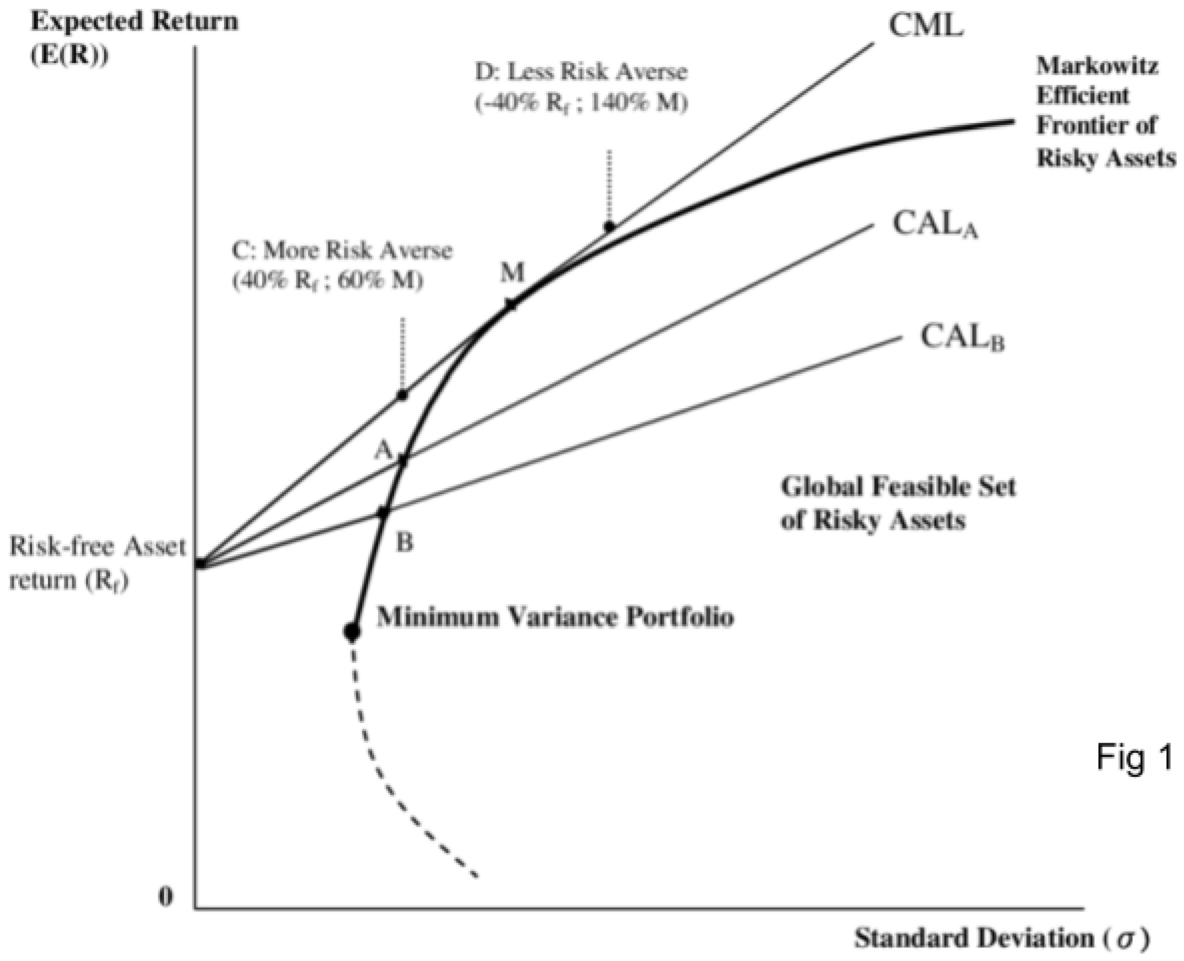
Expected Return (E(R))
CML
D: Less Risk Averse (-40% $R_f$ ; 140% M)
Markowitz Efficient Frontier of Risky Assets
$CAL_A$
C: More Risk Averse (40% $R_f$ ; 60% M)
M
$CAL_B$
A
Global Feasible Set of Risky Assets
Risk-free Asset return ($R_f$)
B
Minimum Variance Portfolio
0
Standard Deviation ($\sigma$)

Fig 1

The Efficient Frontier Curve is a parabolic boundary for all feasible assets. For a two asset example, the boundary is governed by assets that are situated near perfectly positive, perfectly negative and zero correlations.

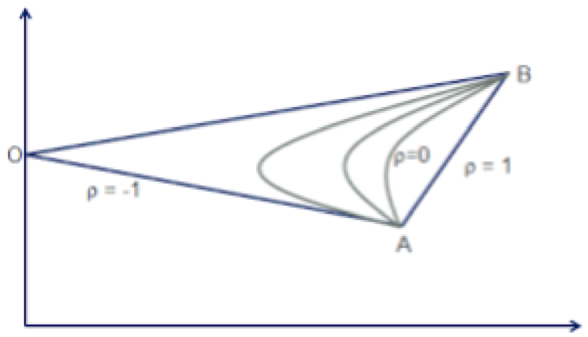

Fig 2

For more details:
https://www.youtube.com/watch?v=fNIoVdDNFmk
https://www.youtube.com/watch?v=lPKtI90f_sE
https://www.math.ust.hk/~maykwok/courses/ma362/Topic2.pdf

Once again, the decision to skip the Markowitz technique was deliberate and reconsidered due to the issue of using weights. A traditional portfolio set of assets were not needed for the task of univariate modeling. However the use of the Sortino Ratio was crucial to making final decisions on selection.

3. The dataset was taken from Yahoo's Finance API. The best part about Yahoo's API is that pandas datareader is able to read the data into a data frame with very little effort, in just one line to be exact. An issue encountered was with the data frame itself, which comes as a multi index data frame. The columns are multi indexed, which means the data frame has to be taken care of in a similar way as a multi index data frame indexed by row. At first, multi indexing can be tricky but

the trick appears to be .loc and slice(None). Once you become familiar with the multi index, you realize that you can hold lots of information in a data frame, which is a fantastic advantage since data frames are useful when it comes time to visualization and modeling. As I’ve become more experienced with python, I’ve realized that I do prefer the data frame as a visualization tool just as much as a model implementation tool. In fact, I have come to realize that I would like to extend that tool to image processing. I have not yet encountered a tool that allows you to visualize images in a dataframe in the same way you can display a matrix of distributions plotted using matplotlib. This tool may already exist. Nevertheless, data frames have paved the way to my understanding of analysis and I have come to find that so much analysis defaults to their use.

Furthermore, cleaning the data frame for my project has proven to be a rather simple task. In my experience, it generally takes a good portion of time if you are dealing with string elements. Many tools for parsing strings already exist and that is one big reason why cleaning is such an easy task. I only look forward to the day that all data fall prey to just a few cleaning algorithms that can fashion it depending on the way it needs to be presented. I hope to participate in this data engineering venture.

Dealing with numerical data has not been that problematic, except when dealing with nans. And in that case, it is not a matter of coding expertise but one of statistics. Do you need to use statistics to make a case for or against dropping columns or rows with nans? For this project, nans were dealt with in a very simple manner. Since there was an abundance of data, there was no need to dwell on the issue of dropping tickers with nans. Upon inspection, nans appeared to arise from the company's lifeline. Some companies had not yet entered the market. As a result, I dropped tickers with nans for the selected time period. Even after removing those tickers, there were still plenty to choose from and clearly needed to be reduced to a smaller, finer pool of tickers.

Returning to the issue of a multi index data frame, you can simply drop the column level you wish to keep and save it to df.columns. This is an easy solution if you wish to move from a multi to a single index data frame.

Reading in the full set of 245 tickers with get_data_yahoo() caused errors, which was resolved via a method for making an exception when errors arose in reading tickers. I'm uncertain about the ultimate meaning of the error because it could not have been solely based on nonexistent values in the period of time selected. I still got a data frame with tickers and nans, meaning there weren't values for some of these tickers for the entire time period.

Aside from multi index manipulations, nan and/or column removal, I had a relatively easy time cleaning the data and did not have to deal with merging data sets. If I had chosen to use macroeconomic variables such as federal interest rates or consumer price indices, I would have had to do a little more cleaning. I would have had to duplicate values since the frequency on many of these variables is much slower. Generally, they don't have a daily frequency. According to portfolio optimization techniques, prediction accuracy may improve if macro variables are used. This is something to verify in future projects since it would not be too taxing to incorporate this data, nor difficult to acquire.

4. Are there variables that are particularly significant in terms of explaining the answer to your project questions?

The variables at play are all the same. They are all prices. To say something about the variety, the prices range from high, low, adjusted close, close and open prices. Once again, these are the prices taken from yahoo finance api. There is a multitude of ways to collect and explore financial data. After all was said and done, I only used adjusted close prices.

To speak more closely about project goals, I considered predicting prices from some industry based on the prices from another related industry. This concept was interesting due to the relationship across certain industries and their potential dependencies. In my opinion, prices alone are sufficient for studying price movement and creating models but in practice, it is likely the case that prediction may have more accurate outcomes if you take into account macroeconomic factors and certain exogenous events. Also, adding these factors doesn’t necessarily complicate the task of prediction but it does affect time to completion for the entire project. Altogether, beginning with a simplified model may afterall verify the difference between using and not using exogenous variables. If predictions solely based on prices can compare well with predictions using exogenous variables, then something should be said about the power of the model and the theory behind exogenous variables.

As pertaining to the methodology of portfolio optimization, that is, optimizing returns, it is recommended in DataCamp's Portfolio Analysis course to use exogenous variables. Once again, I agree that it is a good idea to use relevant variables to explain the variation in prices and error in prediction. These are fruitful tasks to be implemented in the future.

5. Are there significant differences between subgroups in your data that may be relevant to your project aim?

   As of yet, there are no subgroups in the prices. However, exploring the relationship between values across tickers by methods of correlation may be a fruitful path to research. Clustering methods are other methods I highly desire to use in the near future.

   The final set of tickers chosen through simple methods of portfolio selection gave us prices that behave normally based on skew and kurtosis testing. Aside from that, the prices levels between tickers can differ substantially. Exploring price levels and in relation to volume is certainly another path for future research and analysis.

   Personally, I believe using fundamentals would be the simplest and most reasonable path to a solution for profitability, though an expensive one. Fundamentals data is downloadable for free at yahoo finance for no more than three years into the past from current and provided at a much slower frequency. This type of data coupled with macroeconomic data will certainly be fruitful in an attempt to optimize returns and reduce error in prediction.

I have several other methods in mind but unfortunately I cannot share them at this time. Stay tuned to my github and you will see the development of my ideas over time in an ongoing project to model volatile markets accurately, especially the biotech markets.

6. Are there strong correlations between pairs of independent variables or between an independent and a dependent variable?

After careful consideration, the decision to work with univariate data was better for this project, especially since I was more curious about the use of classical statistical models such as autoregression.

In order to use standard autoregressive models, you must become familiar with the parameters p(lags),d(difference lags),q(error lags).  Let me return to these parameters later.

Autoregressive models use lags, which are unique types of variables. These are time shifted variables and they are exact duplicates of the original time series down shifted in a data frame.They are missing future values and that depends upon how many periods you wish to shift the series by.

If you wish to create (t-1) lag variable from an original time series with a daily frequency, this means you would duplicate the series and lose the last data point in the time series and get a nan where the first data point would be in the original time series. Likewise, if you wanted to create a (t-2) lag variable, you would lose the last two data points from the original series and have two nans where the first two data points would be in the original series. All else, remains the same. The original series is just shifted down. This is a convenient matrix structure for the task of differencing. In time series, differencing may be necessary as this may have been the reason for creating lags in the first place. Differencing is necessary for non stationary data, especially if you wish to do modeling. Accurate modeling will be difficult without a stationary time series and that is often the major drawback of time series. Working with time series often requires methods of detrending and that is the purpose of differencing.

Returning to parameter p and d, p stands for the number of lags and d for the number of lags to difference. You can use as many lags as you wish and assess whether they are useful through autocorrelation and partial autocorrelation methods, which are, in many cases, already written into the autoregressive models in stats packages like statsmodels. In particular, the arima function already has a built in optimizer that implements these methods to find the best set of parameters for prediction forecasting. The last parameter q in the arima model delivers the number of lag error terms on which to perform the moving average calculation. In most cases, the order of any of the three parameters will not exceed 3.

There is no shortage of resources for these classical type autoregressive models. The decision to implement a univariate model was due to the fact that these autoregressive models best stand alone as univariate models but apparently can be applied one at a time to many variables, which is ultimately not computationally efficient.

In the arima model, the equation is linear and the weights contribute some amount to each lag variable. The ACF, which is written into the auto arima model, tells us given some arbitrary number, shows us how many lag variables are statistically significant and if there ultimately exists a significant correlation between lags and the original time series. Positive lags mean stickiness between sequential prices. So high prices will be followed by high prices and negative lags mean swings. So high prices could be followed by low prices. In addition, the series may contain seasonality, especially if lag variables show patterns in multiples every 30 days, 60 days, or 6 months, etc. The ACF can pick up this information. What it ultimately means is that correlation is a useful property of time series. It means there is more information with which we can improve our forecast.

It is important to note ACF quite versatile in its use. It’s not just limited to giving us information about the significance of lags, it gives us an indication of stationarity. This is an important property for time series prior to prediction modeling. The mean and variance must be fixed with a constant autocorrelation structure. And stock prices are notorious for being noisy, trendy and highly correlated. In addition to the ACF plot, we performed a DIcky Fuller Test with the null hypothesis that the time series is non-stationary. This was the last step in the statistical analysis before moving toward modeling.

Additionally, t tests were performed on the final selection of stocks to decide whether mean returns varied across any of them. Means varied across all stocks thus we reject the null on every t-test and concluded that if we were to run a regression we would have a suitable set of predictors.