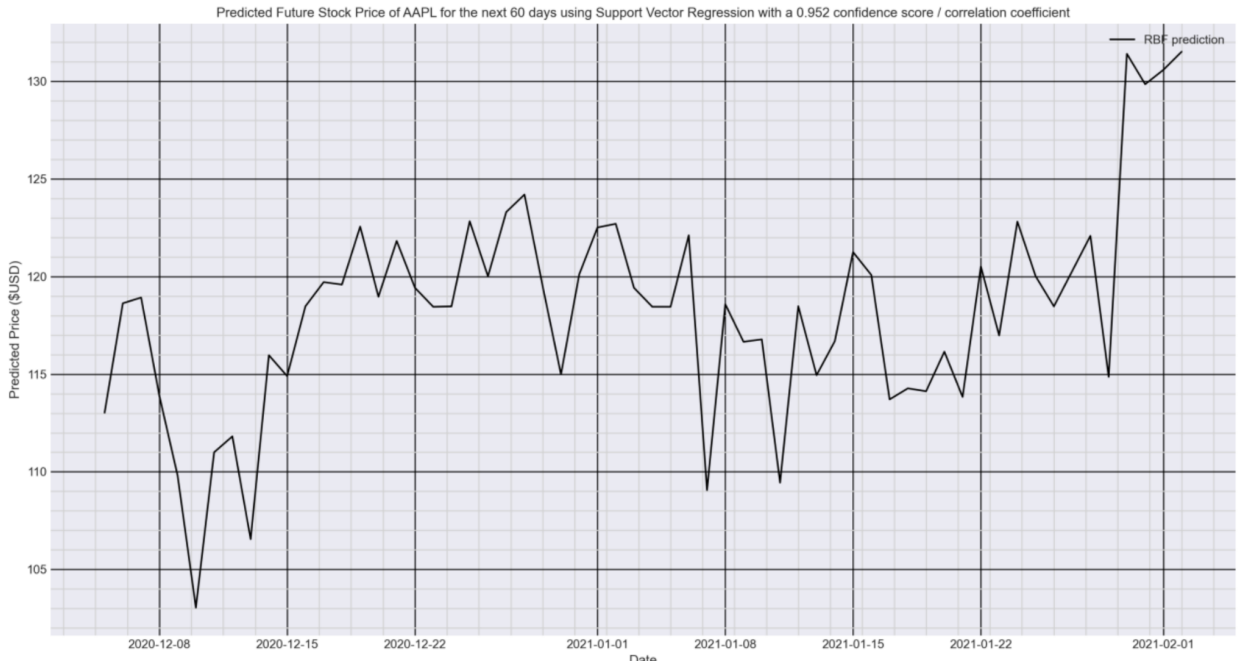
Dataset: https://ca.finance.yahoo.com/

For my project, I didn't have one specific dataset, but rather I called the dataset from Yahoo Finance's stock database using inputs and pandas' datareader. You can type in the stock ticker of any stock that exists on the stock market, and my program will automatically search for the data for that respective stock, and use that information for the models.

For my first classifier, I chose to use support vector machines to predict the price of a stock for x amount of days in the future. The number of days to predict into the future is up to the user to input as well. I was originally thinking of using a basic linear regression model for my predictions, but I found that nonlinear models tended to have a lower predictive error, so I decided to go with support vectors and logistic regression for both of my models. As well, linear regression isn't appropriate for data that might not have a linear relationship, which is the case for this project.

Since SVM's provide the optimal next datapoint based on historical data very effectively, I believed it would've been very effective if I used that model, since for this project, I was only considering the effect of historical prices on the future prices of a stock. I will go into a little more detail later as to why this reasoning would potentially contribute to ineffective predictions. For the purposes of this assignment, however, it was a great option to use since the classes from the data were very easily separable. While I wasn't working with data that had many dimensions, which is the kind of data that works best with SVM models, I still believed it was a good starting choice to predicting future prices based on historical prices.

Here is a sample output for the SVM model, with inputs of ticker = AAPL and days = 60 (seeing the future predicted stock price of Apple for the next 60 days). Again, since the user can type in any stock and prediction time length they want, the number of possible visualizations is infinite.



I also had the program produce the raw prediction values and confidence score:

```
svm confidence (correlation coefficient): 0.9522851953411691

svm prediction values:  [113.02617268 118.63311119 118.92552726 113.89390269 109.80646396
 103.03922321 110.9984555  111.81424243 106.54717173 115.97026566
 114.90128161 118.47694769 119.71773121 119.59064857 122.56248852
 118.96766353 121.82772359 119.42799447 118.45294443 118.47276436
 122.83698735 120.01272243 123.3056293  124.20227317 119.48464483
 114.99250677 120.11358005 122.51532395 122.70472054 119.42265924
 118.45449093 118.45348726 122.11405031 109.05878558 118.58661112
 116.65928293 116.78389067 109.44177116 118.48151454 114.94895749
 116.69287385 121.25453568 120.08126566 113.70753025 114.27325885
 114.12571106 116.15156079 113.83432059 120.51750414 116.98274203
 122.81704872 120.01645961 118.47399308 120.27725484 122.08675557
 114.86361461 131.41083981 129.85524869 130.59203986 131.52354881]
```

I found that the SVM model tended to work better with less volatile stocks (stocks that had less drastic variance in price in the past), and it worked better with shorter time periods. Since my program takes historical data up to the present, there is no way to actually see if the predicted values were in fact accurate other than to wait and see what the prices turn out to be, so the measure I used to see if the program ran well was to use the confidence score. The raw prediction values referred to the values that my SVM model produced. The SVM confidence score is a measure of how well the predicted values fit with the actual values when training the variables – the closer the value was to 1.0, the better the fit.

The model evidently worked well with some stocks but worse for others. Stocks such as AAPL, NVDA, AMZN (Apple, Nvidia, and Amazon respectively) had quite high correlation coefficient scores, which meant that the predicted stock values would be a relatively good fit considering their historical prices. However, for other stocks like UBER, where the stock was more volatile and unpredictable, the SVM model did not work at all, giving correlation scores of close to 0.

I'd conclude that this model can only possibly work for stocks with low volatility, for short timeframes, and overall if the stocks had a high correlation score with the model. Most stocks actually performed above average with the model – I tested the model with many different stocks and most tended to have a score of around 0.7 or higher, which demonstrated a relatively good fit. I'd therefore say that the SVM model definitely has some strong points in terms of considering historical behavior of the data.

For my second classifier, I chose to use logistic regression because of the efficiency and ease of implementation of the model. Logistic regression tends to work well with simpler datasets, and the biggest factor I wanted to take advantage of for this model was the ability to see the direction of association and the coefficient size, which were aspects I believed would have been effective in creating accurate predictions.

Though having the binomial aspect of logistic regression was going to be a little unconventional, I thought it was going to be interesting seeing how well the predicted values would have fit with the original values through this method.

In my second classifier, I had the program produce a graph of the same selected stock, and instead, I had the original returns overlay the predicted returns up to the present date. This was to see how accurate the model was going to be.

As you can see, the predicted returns during this time period were not very accurate. However, we have to consider that the original returns were heavily influenced by the impact of COVID-19, which is a huge factor that this logistic regression model is not able to account for.

Here, I also had the script provide a synthesis table to see the raw data:

```
logistic regression synthesis table:
             0         1  Prediction Up/Down
0     0.468791  0.531209           1      Up
1     0.450127  0.549873           1      Up
2     0.440776  0.559224           1      Up
3     0.434967  0.565033           1      Up
4     0.453198  0.546802           1      Up
...        ...       ...         ...     ...
1046  0.717977  0.282023          -1    Down
1047  0.318390  0.681610           1      Up
1048  0.425489  0.574511           1      Up
1049  0.324044  0.675956           1      Up
1050  0.394795  0.605205           1      Up

[1051 rows x 4 columns]
```

While I don't believe that either of these results are extremely accurate, I do think that it can provide a very, very basic preview of what the general trend of a particular stock might be in the future.

I felt that each model had its strengths and weaknesses, but I think overall, the logistic regression model was more accurate in learning about the historical stock movements. The SVM model has high risk to overfitting, which is why the score might be so accurate; Objectively, the SVM model performed better in terms of accuracy, but because of this high chance that the data is overfit, it might perform poorly when it comes to estimating future prices. Usually, if a SVM is paired with a LSTM neural network in this case, it would've led to more effective, unbiased predictions. Logistic regression is also a model that is based more upon probability, while SVM is more based upon how the values performed geometrically in the past. Depending on your view, you could believe that stock prices movements are more technically-based (using technical indicators, following patterns to predict future movements) or probability-based, and I believe that the probability-based approach offers more accuracy when estimating if a stock goes up or not since it tracks the actual historical activity of how likely a stock is going to go up or down and applies that to the current price to estimate future prices, rather than relying more on the shape of the graph to make predictions, as is in SVM's case. I also used a moving average in my logistic regression, which I believe can help increase the accuracy of predicting if a stock will go up or down based on their position relative to that average.

Again, however, when I used the logistic regression model, the probabilities of 0 or 1 were always relatively close to each other (both hovered around the 40-60% probability mark, meaning the prediction of whether the stock went up or down was always a relatively close call). Therefore, it obviously led to discrepancies as we saw in the second visualization with AAPL, but since I believe that data was less overfit, it would portray a more accurate prediction of future prices.

In general, since these approaches are best for situations where future values depended on past values, the glaring downfall to these models are that both are solely reliant on the historical movements of the stock price to help them predict the future prices. However, we know that realistically, stock prices can be

extremely random and are not solely based on historical prices and technical factors; economic events, news, unexpected company events are all things that have massive influences on the stock price regardless of how the stock performed in the past. I had the idea to combine news sentiment about economic events and about the particular stock the user wants to analyze alongside these models, but that step would have required significantly more time to code, so I was unable to program it. However, I believe that if these basic SVM and regression models were combined together alongside these other models (sentiment analysis from news, social media, etc.), then I believe there would be much more accuracy predicting stock movements.