Application of Support Vector Machine for Predicting Directional Movement of Stock Prices

MATH 6350 - Statistical Learning and Data Mining

Ravik Chand
Lucas Smith

MATH 6350 - Statistical Learning and Data Mining Ravik Chand
Project 5 Report Lucas Smith

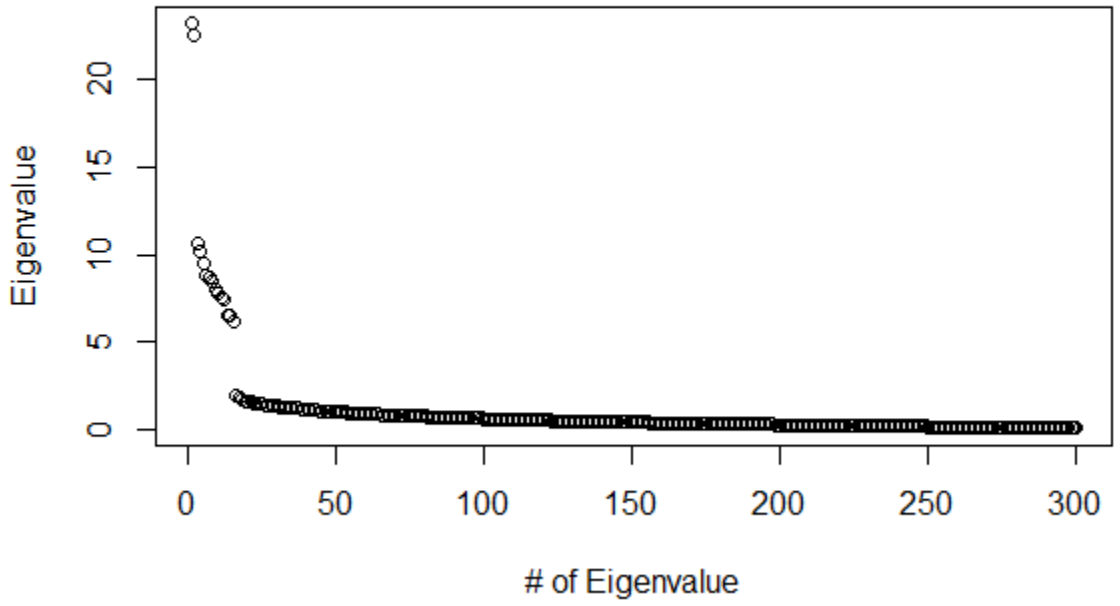
Project 5 was a demonstration of the application of the Support Vector Machine algorithm as a classifier. More specifically, the objective of Project 5 was to use the SVM model to predict whether the closing price of a particular stock traded on the U.S. market would exceed or fall below the previous closing price based on indicators from the preceding days. The data used was obtained from Yahoo! Finance in the form of .csv files containing 1258 observations of closing stock prices, among other variables, for 20 companies between January 1st, 2016 and December 30th, 2020. The companies selected were from a variety of business sectors and included in alphabetical order the Archer-Daniels-Midland Co. (ADM), Applied Materials (AMAT), Air Products & Chemicals, Inc. (APD), BlackRock, Inc. (BLK), Crane Holdings Co. (CR), Ford Motor Co. (F), General Electric Co. (GE), International Business Machines Corp. (IBM), JPMorgan Chase & Co. (JPM), Kuehne + Nagel International AG (KHNGF), Martin Marietta Materials Inc. (MLM), Microsoft Corp. (MSFT), Northrop Grumman Corp. (NOC), News Corp. (NWS), Panasonic Holding Corp. (PCRFF), Pfizer Inc. (PFE), PPG Industries Inc. (PPG), AT&T Inc. (T), UnitedHealth Group Inc. (UNH), and Visa Inc. (V). These files were imported as data frames into R, ranging from *comp1* to *comp20*. The data frames were compiled into the list *comps* for the batch application of functions for data processing. The closing cost was isolated as the column **s**, and calendar dates were replaced with a progressive count of days **t** then returned from *comps* to replace the existing *compJ* data frames.

No missing values were found and data frames contained a uniform count of rows. A function get.y.fxn was created to generate a new column **y** which would contain the percent change in **s** on day **t** relative to the previous day, **t**-1. This function was applied to the list *comps* to generate the rate of return column **y** for all data frames which then replaced the *compJ* data frames.

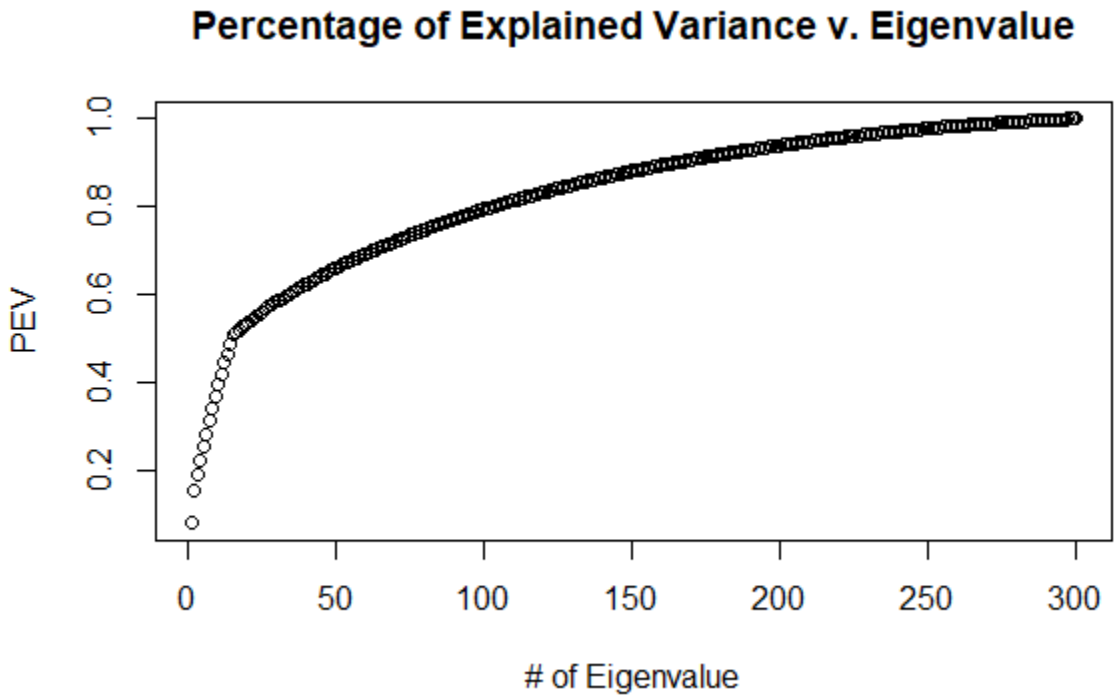
The function get.v.fxn was created to generate a new data frame **x.v**, in which **x** represents an input data frame *compJ*, which would contain the columns that represent the rate of return from the preceding 14 days leading up to and including day **t**. For each data frame **x**, this function obtained the preceding 14 values from the column **y**, beginning with **y** where **t** is equal to **t**-15, to form the columns **x.1** through **x.15** which were bound together to form the data frame *v*. This function was applied to all 20 data frames *compJ*, creating the data frames *compJ.v* which were then bound together to form the data frame *comp.v* containing 1243 observations of 300 features each.

Of the data frames *compJ*, *comp20* or Visa Inc. was selected as the stock for which predictions would be made. A vector *comp20.labels* was created to hold the true classification of **y** on a specific day **t** for days 1 through 1243. Days for which the rate of return was greater than or equal to 0.6% were designated as HIGH, while days below a 0.6% return were classified as LOW. The percentage of HIGH returns was found to be 32.9%, and LOW returns accounted for 67.02%.

A 300x300 correlation matrix *CORR* was computed using the cor function on the data frame *comp.v*. The eigen function was then applied to the *CORR* matrix to generate the matrix *L* and vector *W* which contained the eigenvectors and eigenvalues respectively. The sum of values in vector *L* was confirmed to equal 300 and individual values were plotted against the total count of values in the graph below.



The percentage of explained variance was charted in the below graph and of the 300 observations in *L*, 212 were determined to capture 95% of variance. The matrix *redW* was created from the first 212 columns of the matrix *W*. Then, through the matrix multiplication of *comp.v* and *redW*, the 1243x212 matrix *Z* was created.

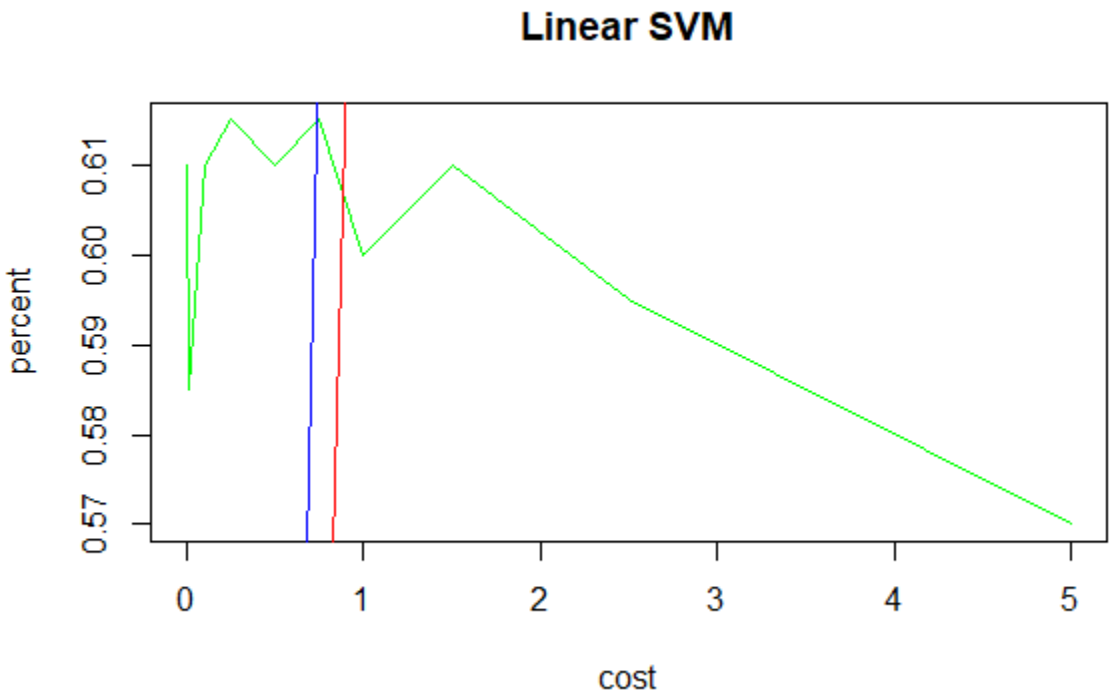


The matrix Z was split into a *trainset* and *testset* based on the random sample of index values *idx*. The resulting matrices contained imbalanced ratios between the HIGH and LOW classes. The HIGH class was oversampled through cloning for both the *trainset* and *testset* separately such that HIGH observations amounted to about 40% of observations.

A linear SVM algorithm was applied using the svm function from the "e1071" package. A vector *costval* was created housing 10 cost values ranging 0.001 to 5 for provision to the SVM algorithm. The svm function was trained using the *trainset* within a for-loop iterating on the cost values followed by predictions made using the predict function and the *testset*. Each iteration deposited results into corresponding vectors. The count of support vectors as *svm.v.count*, accuracy observed in training as *svm.train.acc*, the accuracy observed in testing as *svm.test.acc*, the accuracy regarding the prediction of HIGH observations in testing as *svm.high.acc*, and the accuracy regarding the prediction of LOW observations in testing as *svm.low.acc*. The ratio of test accuracy and training accuracy per iteration, *svm.ratio*, was calculated as *svm.test.acc* divided by *svm.train.acc*. Likewise, the percentage of support vectors per iteration was calculated as the quotient of *svm.v.count* and the count of observations in the *trainset*. These results appear in the below table. The subsequent graph displays the percentage accuracy of test set predictions in **green**, accuracy of training set predictions in **red**, and percentage of support vectors in **blue**, each in relation to the cost coefficient.

Ravik Chand
Lucas Smith

Cost	Train Acc.	Train Acc.	Test/Train	SVM %
0.001	68.51%	61.00%	0.890	83.21%
0.010	72.71%	58.50%	0.805	75.57%
0.100	72.63%	61.00%	0.840	71.96%
0.250	72.29%	61.50%	0.851	72.04%
0.500	71.96%	61.00%	0.848	72.12%
0.750	72.12%	61.50%	0.853	72.46%
1.000	72.46%	60.00%	0.828	72.04%
1.500	72.46%	61.00%	0.842	71.96%
2.500	72.46%	59.50%	0.821	72.12%
5.000	72.46%	57.00%	0.787	74.64%



The optimal cost observed amongst the coefficients in *costval* was 0.25, at which point a test accuracy of 61.5% was achieved, similar to cost value of 0.75, however with a smaller count of support vectors.

Train	Truth	
Pred.	HIGH	LOW
HIGH	258	110
LOW	220	603

Test	Truth	
Pred.	HIGH	LOW
HIGH	33	30
LOW	47	90

The svm function was again applied, this time using a radial kernel. A matrix *hil.dif* was created to facilitate the calculation of Hilbert distances between HIGH and LOW class observations. These Hilbert distances were then used to define a range of gamma values from 0.125 to 0.5 at increments of 0.125 which were then provided to the SVM algorithm. A for-loop was implemented at each of the 4 gamma values iterating through cost values that were defined earlier by *costval*. The results of the SVM algorithm running at each value of gamma were tabled as follows.

Gamma 0.125			
Cost	**Train Acc.**	**Test Acc.**	**SVM %**
0.001	59.87%	60.00%	100.00%
0.010	59.87%	60.00%	100.00%
0.100	59.87%	60.00%	100.00%
0.250	59.87%	60.00%	100.00%
0.500	82.70%	60.00%	100.00%
0.750	100.00%	60.00%	100.00%
1.000	100.00%	60.00%	100.00%
1.500	100.00%	60.00%	88.58%
2.500	100.00%	60.00%	88.58%
5.000	100.00%	60.00%	88.58%

Ravik Chand
Lucas Smith

Gamma 0.250			
Cost	**Train Acc.**	**Test Acc.**	**SVM %**
0.001	59.87%	60.00%	100.00%
0.010	59.87%	60.00%	100.00%
0.100	59.87%	60.00%	100.00%
0.250	59.87%	60.00%	100.00%
0.500	82.70%	60.00%	100.00%
0.750	100.00%	60.00%	100.00%
1.000	100.00%	60.00%	100.00%
1.500	100.00%	60.00%	88.58%
2.500	100.00%	60.00%	88.58%
5.000	100.00%	60.00%	88.58%

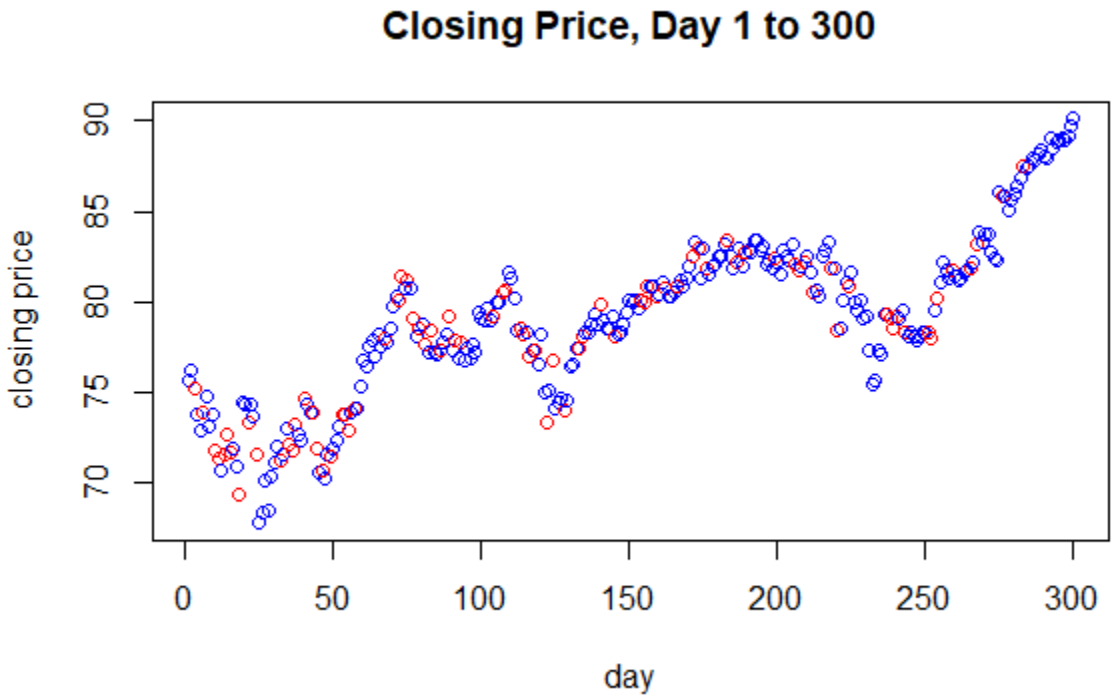
Gamma 0.375			
Cost	**Train Acc.**	**Test Acc.**	**SVM %**
0.001	59.87%	60.00%	100.00%
0.010	59.87%	60.00%	100.00%
0.100	59.87%	60.00%	100.00%
0.250	59.87%	60.00%	100.00%
0.500	82.70%	60.00%	100.00%
0.750	100.00%	60.00%	100.00%
1.000	100.00%	60.00%	100.00%
1.500	100.00%	60.00%	88.58%
2.500	100.00%	60.00%	88.58%
5.000	100.00%	60.00%	88.58%

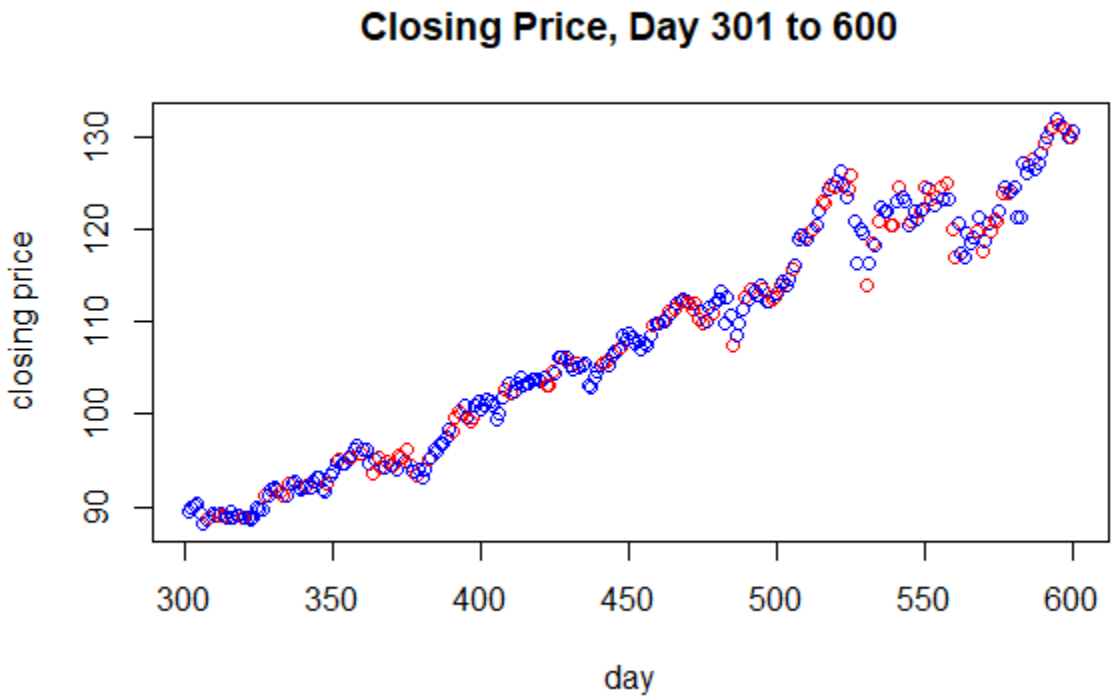
Ravik Chand
Lucas Smith

Gamma 0.500			
Cost	**Train Acc.**	**Test Acc.**	**SVM %**
0.001	59.87%	60.00%	100.00%
0.010	59.87%	60.00%	100.00%
0.100	59.87%	60.00%	100.00%
0.250	59.87%	60.00%	100.00%
0.500	82.70%	60.00%	100.00%
0.750	100.00%	60.00%	100.00%
1.000	100.00%	60.00%	100.00%
1.500	100.00%	60.00%	88.58%
2.500	100.00%	60.00%	88.58%
5.000	100.00%	60.00%	88.58%

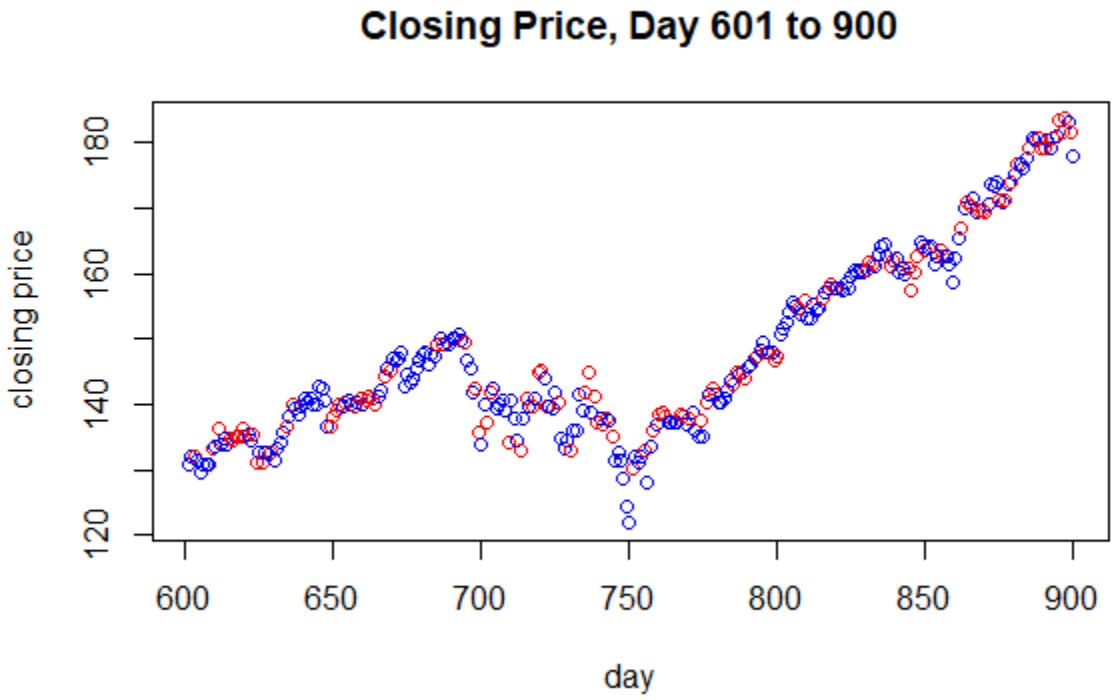
The below graphs show the closing stock price **s** for company 20 by the end of day **t**. Separate graphs show the percent change **y** in stock price for company 20 by the end of day **t** relative to the previous day. In both sets of graphs, HIGH percent change observations are denoted in **red**, while LOW percent change observations are shown in **blue**.

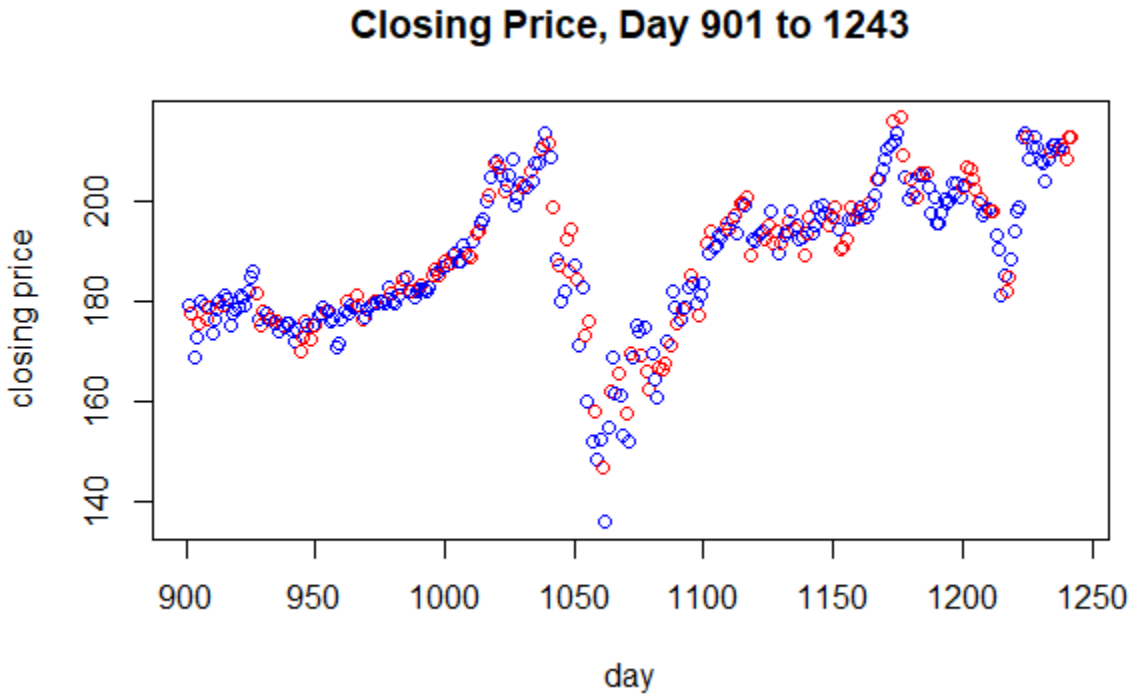
Ravik Chand
Lucas Smith



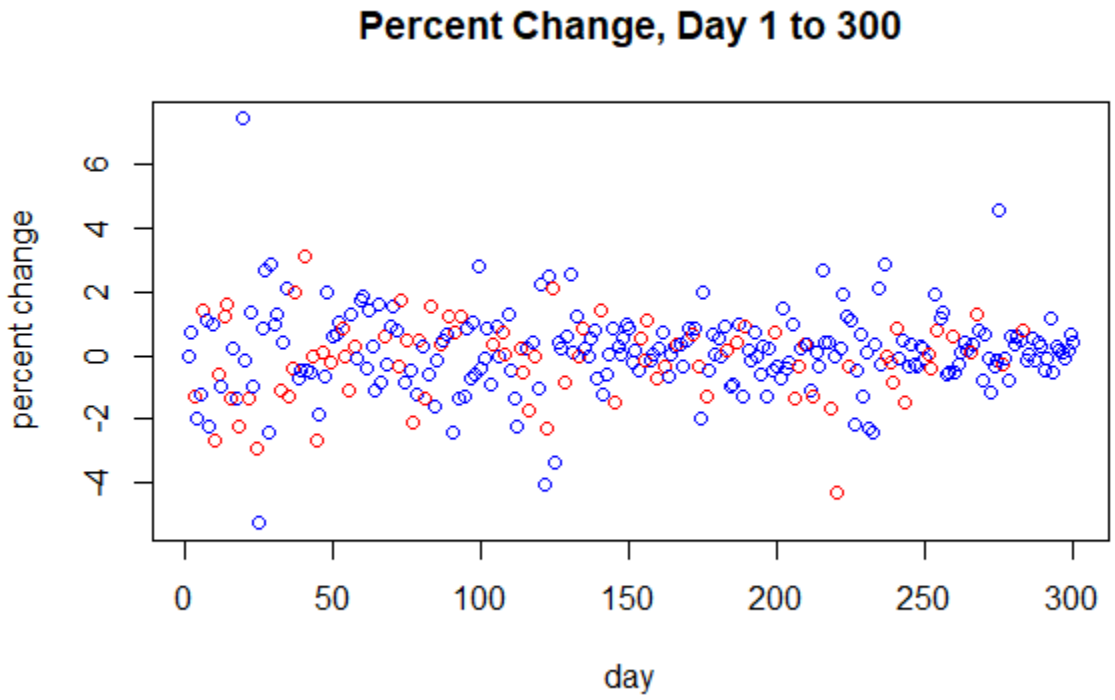


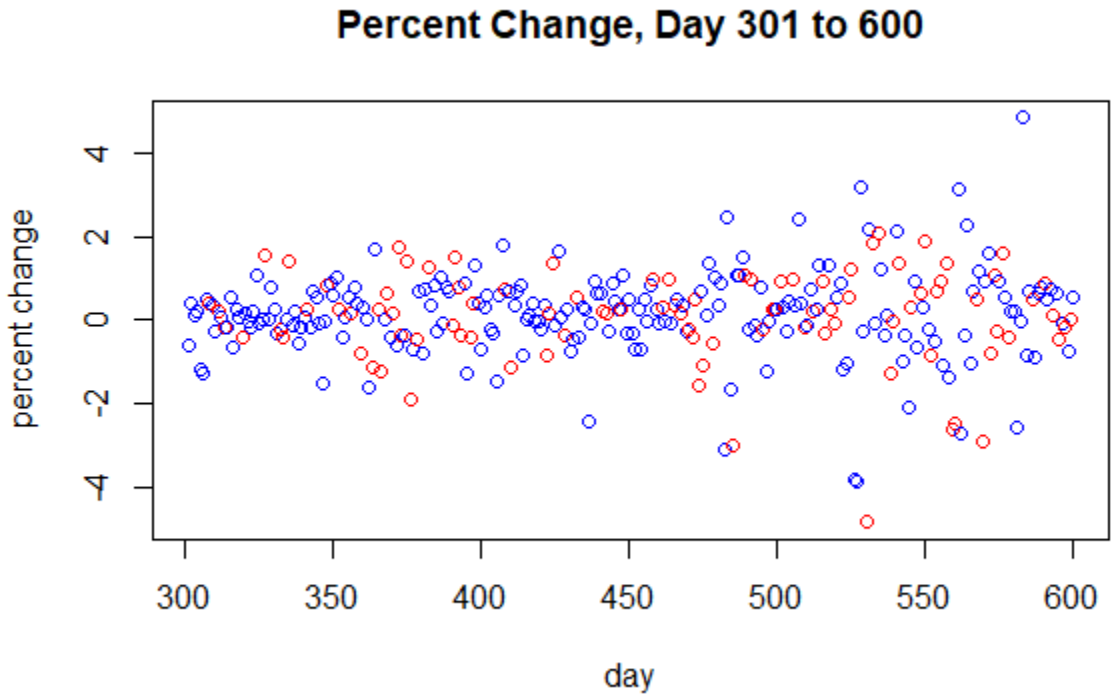
Ravik Chand
Lucas Smith





Ravik Chand
Lucas Smith





Ravik Chand
Lucas Smith