Time Series Forecasting using Machine Learning

Vittorio Bisin and Raghav Singhal

December 20, 2016

1 Introduction

Our project is fundamentally divided into two parts: *a*) where we apply a clustering algorithm onto stock market data and *b*) where we use the resultant clusters to add features into a previously developed method using Support Vector Machines for predicting individual stock prices. The motivation for the project was initially conceived as a response to the high degree of stochasticity in predicting future stock prices, and attempting to group stock performances together, so as to decrease possibile individual randomness in predicting stock prices tomorrow. Furthermore we also noticed that several key technical indicators showed a delayed response to a developing trend (which however were captured by other members of the cluster). Our project makes short-term predictions (1,5, and 10 days) using time-series data and feeds this data to train a SVM with a polynomial kernel.

2 Methods

We initially downloaded the Yahoo! Finance Python module, giving us quick access to each companys: name, stock market, daily opening price, daily closing price, daily highest price, daily lowest price, daily volume traded.

We then also programmed a web crawler to go on Google Finance and download the sector and industry for each of our stocks. We downloaded stock quotes from 2010-2016 for the three major American stock markets: NYSE, NASDAQ, and NYSE MKT (formerly known as AMEX).

Given such a large set of data, we had to complete a fair amount of pre-processing. Firstly we deleted any stocks that had any missing quotes or whose industry the Google Finance crawler was unable to find. At this point we had 1,510 days worth of quotes but roughly 2,500 companies per stock market (AMEX is a bit smaller), meaning that for us to create a graph structure between stocks (explained later) we needed to have at most as many companies as samples. We firstly decided to discard stock funds from our sample because their price changes would be fundamentally dependent on other stocks performance, thus making the clustering question less interesting (but perhaps, easier). For better results, we then decided that we wanted to cluster stocks for 3 separate time intervals within these 1510 days, so to

do so we narrowed down our samples to 400 stocks for NYSE and NASDAQ and 100 for AMEX. We did so by choosing the top 400/100 companies with the greatest daily traded market capitalization.

Then we decided what clustering algorithm to implement on any given feature, using the Scikit Learn package in Python we decided to use the Affinity Propagation algorithm.
This algorithm uses a graph distance metric (e.g. nearest neighbor) while also allowing for many clusters (of different sizes), without needing to specify the number of clusters (unlike the classic clustering algorithm Kmeans). Following an affinity propagation implementation on the Sckikit Learn website (see Varoquaux) we first use a sparse inverse covariance estimation to determine which quotes are conditionally corrrelated on others in the stock market. We thought it a good idea to firstly promote sparsity in that we wanted to separate many stocks being correlated to each other, rather than any one individual stock. We also used inverse covariance estimation to solve for the partial correlations between variables, that is solving for the correlation between two variables, while holding all the other variables constant. Which is extremely important in our setting since there will be a great deal of correlation betweem variables.

Our next question was to determine which features to cluster over, which features best define and differentiate a companys change in stock price. We found several papers (see papers by Kim, Kim et all, and Di) suggesting 12 optimal features for predicting future stock prices(see feature space section)

Given our later described clustering algorithm over multiple graphs, it is important to minimize the number of features over which we cluster. That is, because every feature will be given equal weight, then it is important to determine the relevance of each feature in discerning the change in stock price. To determine which subset of these features to run the algorithm on we ran a Ridge regression and Elastic Net regression (with varying values of alpha). Our rationale for choosing these methods was that we wanted to be careful for any possible multcollinearity between variables (as we expected there to be a great deal), while also trying to promote a zero biasing metric (i.e. sparsity) to narrow down the subset of relevant features. Fortunately, independently of the chosen regression, these 7 of the 12 original features had significantly greater weights than the others: Momentum, ROC, A/D Oscillator, Disparity 5, Disparity 10, OSCP, and RSI.

Our objective was to determine a novel way to cluster stocks using these 7 features as well as each stocks sector. Each of these features would be calculated for each stock, and then we would find each stocks conditional covariance on the others. This conditional covariance matrix can also be interpreted as a graph where each node represents a stock and each weighted edge between two nodes represents the conditional covariance of that feature between the two stocks. In other words, our problem dissolved into finding a method to cluster over multiple graphs.
Our solution to this problem is the following algorithm: fixing the first time interval, we computed an inverse covariance matrix and clustered using affinity propagation once for

every of the seven features. We then repeated this operation for each of the two remaining time intervals. After this we took an average of the number of times that any two stocks were in the same cluster for a given feature and time interval. Furthermore, if two stocks were in the same sector we also added to this sum

$$W_{ij} = \mathbf{1}_{sector_i=sector_j} + \mathbf{1}_{industry_i=industry_j} + \sum_{feature=1}^{7} \frac{\sum_{time=1}^{3} \mathbf{1}_{C_i^{feature}=C_j^{feature}}}{3} \tag{1}$$

In such a way each of all our features had uniform weight. This summation was then set to be the weight of the edge between the nodes and then using the weight matrix as a measure for the similarity between the nodes we clustered on this resultant graph to achieve our final cluster.

3 Feature Space

Let C_t be the closing price at time t, H_t be the high price at time t, L_t be the low price at time t , and O_t be the opening price at time t.

The output is defined as :

$$\begin{aligned} Y1_t &= sign(C_{t+1} - C_t) \\ Y5_t &= sign(C_{t+5} - C_t) \\ Y10_t &= sign(C_{t+10} - C_t) \end{aligned}$$

And the input X has the following features for each day t :

$$K_t = \frac{C_t - LL_{t-n}}{HH_{t-n} - LL_{t-n}} * 100$$

where HH_{t-n} and LL_{t-n} are the highest high and lowest low respectively

$$D_t = \frac{\sum_{i=0}^{n-1} K_{t-i}}{n}$$

$$SlowD_t = \frac{\sum_{i=0}^{n-1} D_{t-i}}{n}$$

$$Momentum_t = C_t - C_{t-4}$$

$$ROC_t = \frac{C_t}{C_{t-n}} * 100$$

$$WilliamsR_t = \frac{H_n - C_t}{H_n - C_n} * 100$$

$$(A/D)oscillator_t = \frac{H_t - C_{t-1}}{H_t - L_t}$$

$$Disparity5_t = \frac{C_t}{MA_5} * 100$$

$$Disparity10_t = \frac{C_t}{MA_{10}} * 100$$

$$OSCP_t = \frac{MA_5 - MA_{10}}{MA_5}$$

$$CCI_t = \frac{M_t - SM_t}{0.015|R_t|} * 100 \; where$$
$$M_t = \frac{H_t + C_t + L_t}{3}$$
$$SM_t = \frac{\sum_{i=1}^{n} M_{t-i+1}}{n}$$
$$R_t = \frac{\sum_{i=0}^{n-1} M_{t-i} - SM_t}{n}$$

$$RSI_t = 100 - \frac{100}{1 + \frac{\sum_{i=1}^{n} Up_{t-i+1}}{\sum_{i=1}^{n} Dw_{t-i+1}}} \text{ where Up means upward price change and Dw means downward price change}$$

Now after we have clustered the three markets we then add 12 more features to the feature space but where H_t, L_t, C_t, and O_t are the combined prices for each stock in the cluster.

4 Results

After each iteration of our clustering algorithm we obtained the three clusters for each stock market: AMEX (see appendix 1), NASDAQ (see appendix 2), and NYSE (see apprendix 3). These clusters were graphed building on code written by Varoquaux, which combines three different characteristics: i) each node has a color defining its cluster ii) the covariance matrix is used to display the strength of the edges (warmer colors for stronger connections) and iii) 2D embedding is used to position the nodes. Given the large number of stocks and (unintuitive) 2D embedding, it is somewhat hard to intuitively understand the accuracy of the clusters. Nevertheless, we can visualize how separate stocks clearly fall into various clusters.

As described in the Methods section, to greater assess the validity of our clusters, we ran the SVM algorithm with the combined features from the individual stock as well as the cluster. We compare our results (see appendix 4) with those performed without the added cluster features (See Kim and Di). In both Di and Kims paper they identified a slightly better than random (circa 55 percent) accuracy for next day prediction using their algorithm, similar to our results. However, for the 5 and 10 day predictions we achieved higher accuracy results with an average greater than 70 percent accuracy, whereas in Dis paper he reaches a high of 70-74 percent (our high is 84 percent). However, it is in the long term that our results are far better, in predicting 10 days ahead we achieve over an 83 percent accuracy rate (with a high of 95

5 Discussion

Our inital motivation for this project was to be able to control daily indiviudal stock perturbations by assigning the stock a cluster and visualizing group effects. As we mentioned in the introduction, we noticed that when predicting individual stocks, for certain technical indicators, there was a lag in a developing trend. By assigning each stock into a cluster, we prevent this from occurring, since this trend will be picked up by the cluster. Our results show that the cluster controls for individual stock perturbation signficantly better than solely including the individual stock features, especially in the long run. This may be due to the fact that the stock perturbations have greater effect over the course of one day, but are distributed evenly as we begin to take greater time intervals. This then allows our clustering features to more accurately predict stock prices for these larger time intervals.

6 Works Cited

1. Bonde, Ganesh, and Rasheed Khaled. "Extracting the Best Features for Predicting Stock Prices Using Machine Learning.
2. ataltepe, Zehra, Sava zer, and Vahide Unutmaz Barn. "Feature Selecton for Price Change Prediction." (2011)
2. Di, Xinjie. "Stock Trend Prediction with Technical Indicators Using SVM."
4. Frey, Brendan. "Affinity Propagation." Lecture notes. University of Toronto,
5. Friedman, Jerome, Trevor Hastie, and Robert Tibshirani. "Sparse Inverse Covariance Estimation with the Graphical Lasso." Biostatistics 9.3 (2007): 432-41.
6. Kim, Kyoung-Jae, and Ingoo Han. "Genetic Algorithms Approach to Feature Discretization in Artificial Neural Networks for the Prediction of Stock Price Index."Expert Systems with Applications 19.2 (2000): 125-32.
7. Kim, Kyoung-Jae. "Financial Time Series Forecasting Using Support Vector Machines." Neurocomputing 55.1-2 (2003): 307-19.
8. Kim, Kyoung-Jae. "Financial Time Series Forecasting Using Support Vector Machines." Neurocomputing 55.1-2 (2003): 307-19.
9. Tang, Wei, Zhengdong Lu, and Inderjit S. Dhillon. "Clustering with Multiple Graphs." 2009 Ninth IEEE International Conference on Data Mining (2009).
10.Tay, Francis E.h, and Lijuan Cao. "Application of Support Vector Machines in Financial Time Series Forecasting." Omega 29.4 (2001): 309-17.
11. Varoquaux, Gael. "Visualizing the Stock Market Structure." Scikit-Learn Documentation.
12. Walter, Sebastian F. Clustering by Affinity Propagation. Thesis. ETH Zurich (Eidgenssische Technische Hochschule Zrich), 2007.

Appendix

Below we have the visualiztion for the Market Clusters:

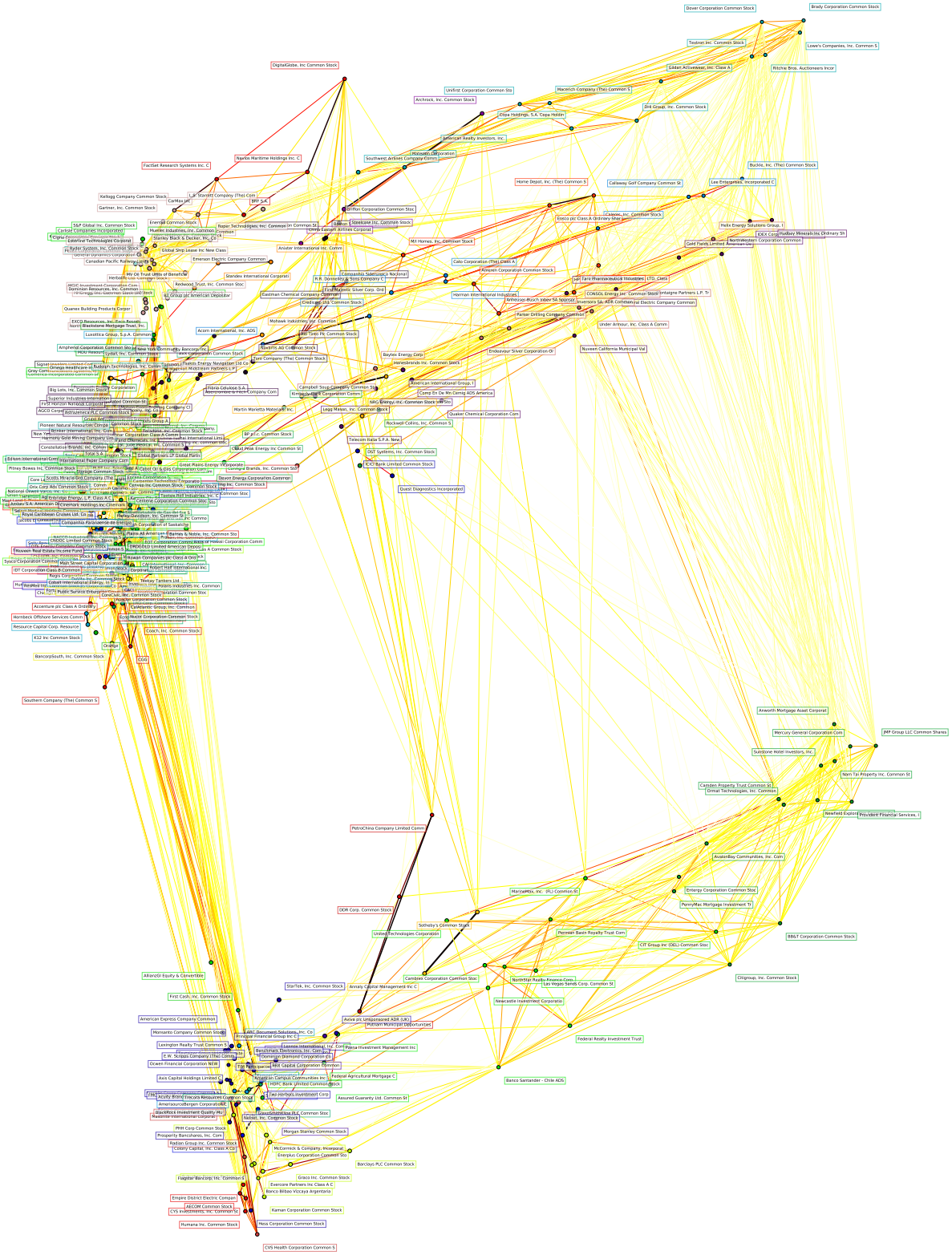

Figure 1: Visualization of NYSE clusters

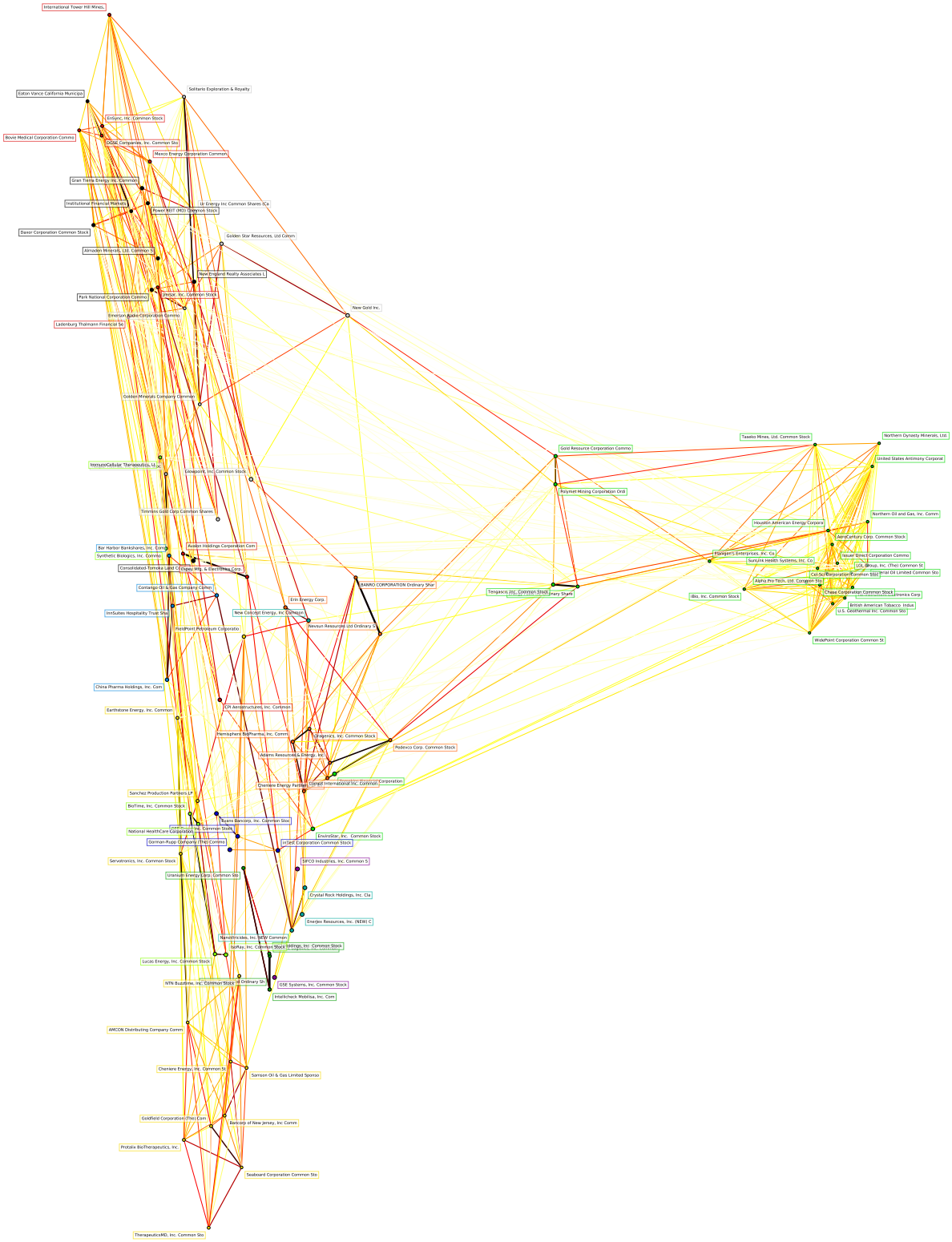

Figure 2: Visualization of AMEX clusters

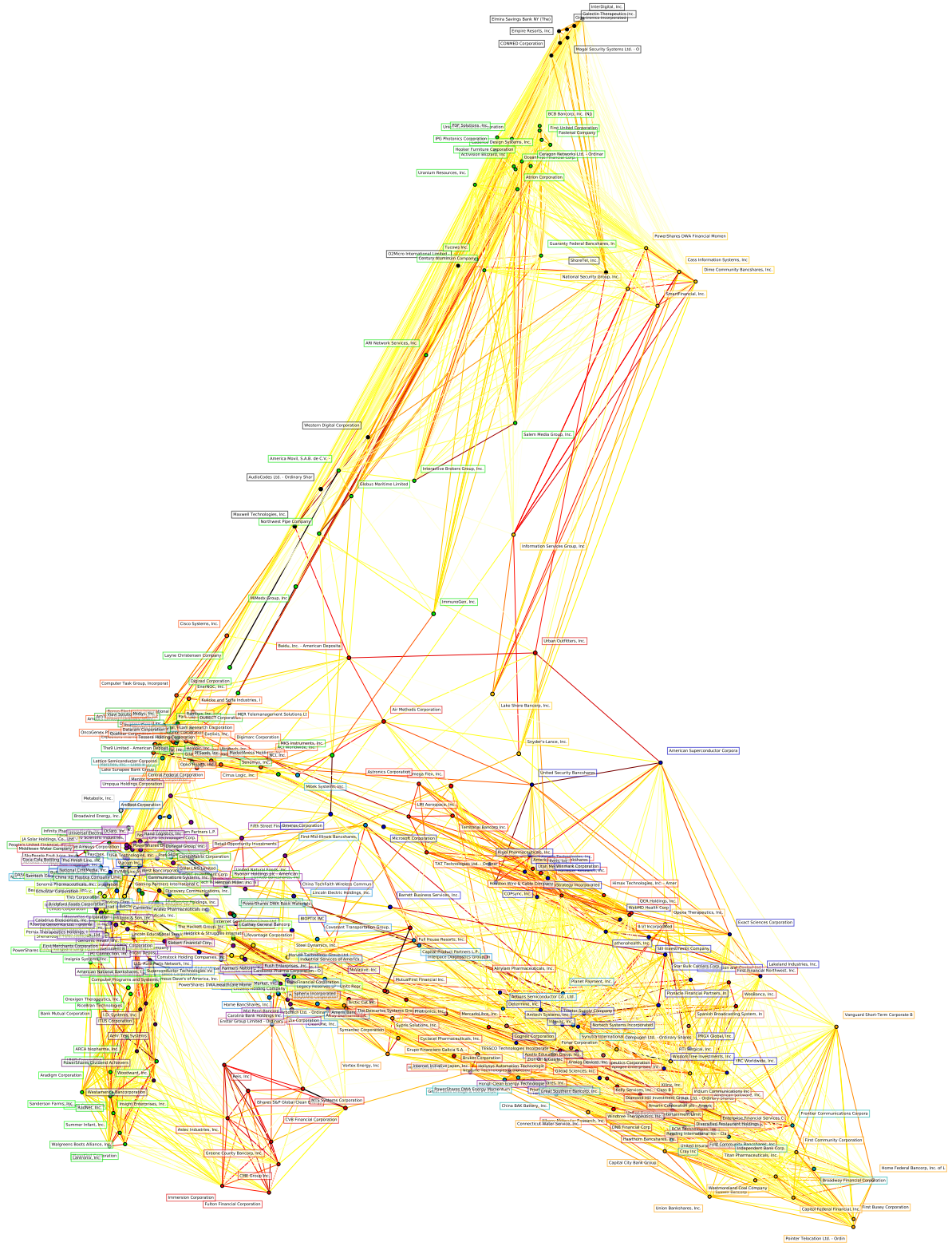

Figure 3: Visualization of NASDAQ clusters

AMEX		1 Day Prediction	5 Day Prediction	10 Day Prediction
	Mean	0.520	0.706	0.798
	Variance	0.007	0.005	0.004
	High	0.690	0.837	0.837
	Low	0.172	0.298	0.574
NASDAQ				
	Mean	0.569	0.725	0.821
	Variance	0.004	0.007	0.007
	High	0.689	0.880	0.956
	Low	0.317	0.0284	0.368
NYSE				
	Mean	0.588	0.742	0.834
	Variance	0.003	0.005	0.005
	High	0.718	0.915	0.949
	Low	0.021	0.312	0.169

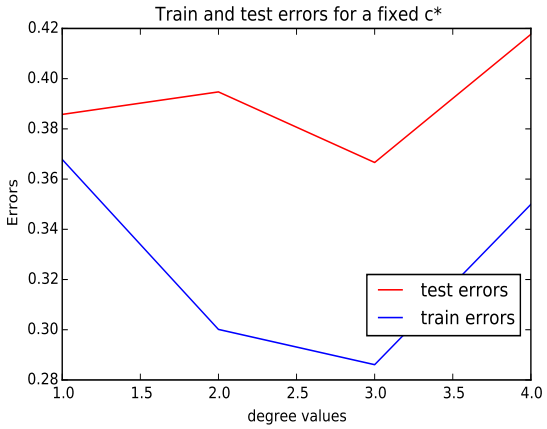

Figure 4: 1-day Training and Testing prediction error for DTE energy with fixed C^*

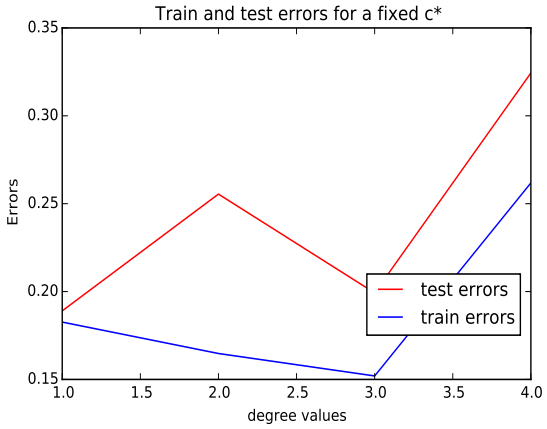

Figure 5: 5-day Training and Testing prediction error for DTE energy with fixed C^*

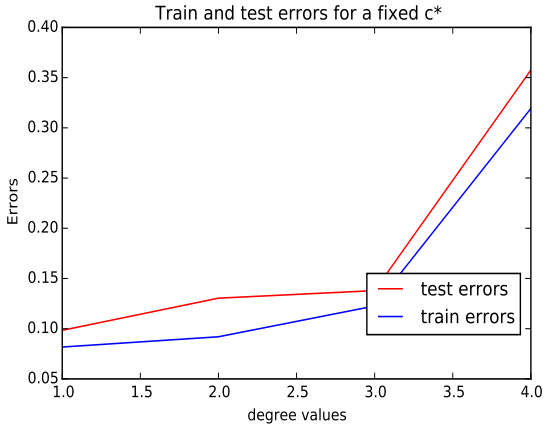

Figure 6: 10-day Training and Testing prediction error for DTE energy with fixed C^*

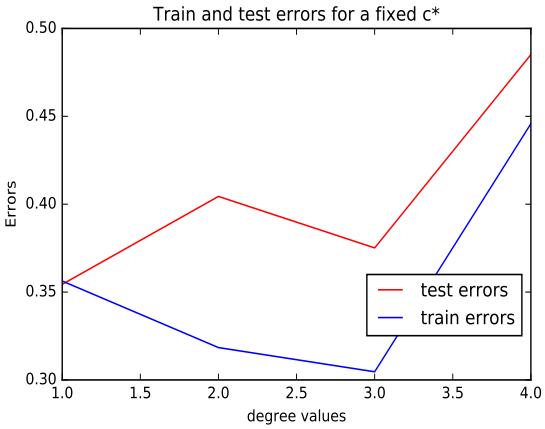

Figure 7: 1-day Training and Testing prediction error for FedEx with fixed C^*

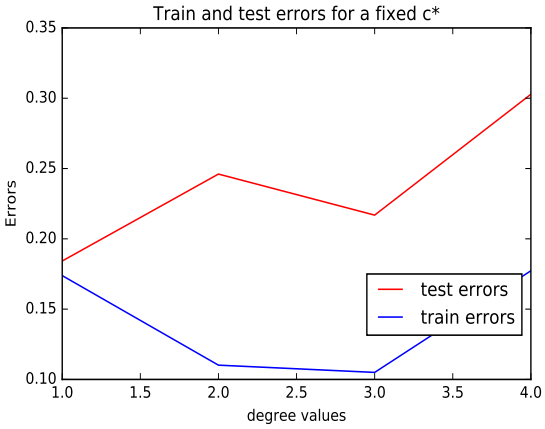

Figure 8: 5-day Training and Testing prediction error for FedEx with fixed C^*

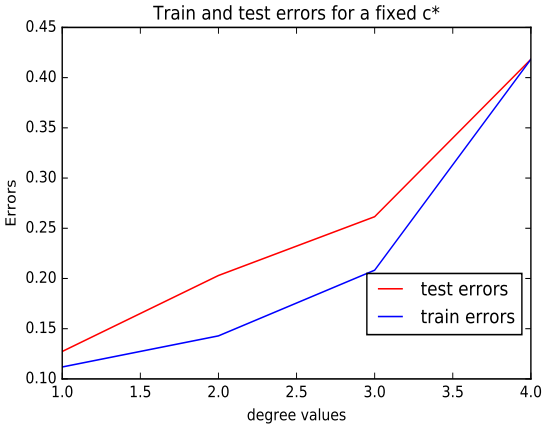

Figure 9: 10-day Training and Testing prediction error for FedEx with fixed C^*

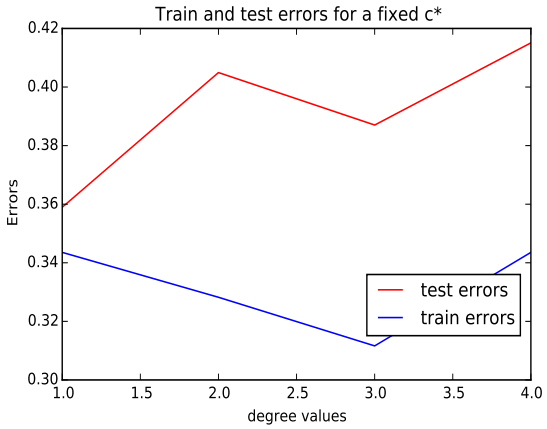

Figure 10: 1-day Training and Testing prediction error for ViaCom with fixed C^*

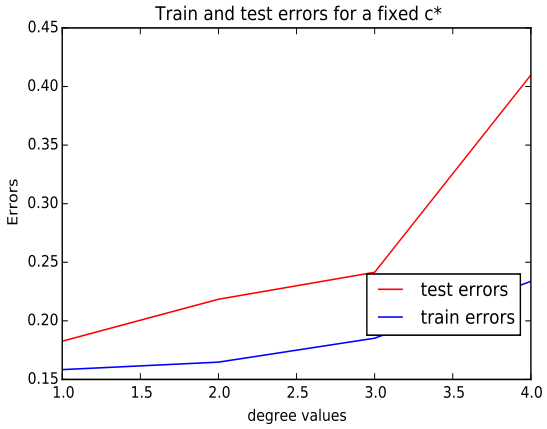

Figure 11: 5-day Training and Testing prediction error for ViaCom with fixed C^*

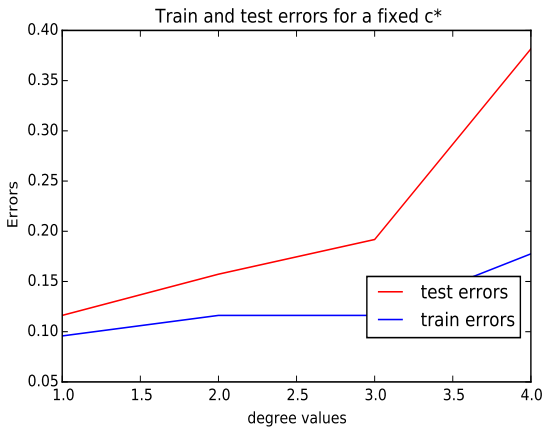

Figure 12: 10-day Training and Testing prediction error for ViaCom with fixed C^*