Application of Deep Learning to the Random Walk Theory

Matt Liston

University of Edinburgh
S1580449@ed.ac.uk

Abstract

The theory of random walks in stock market data has been a debated subject in the field of quantitative finance for at least fifty years. Over that time, many chartists and technical theorists have attempted to disprove the theory by pointing out various anomalies using statistical methods. Using these anomalies, they hope to provide evidence for some sort of meaningful pattern in the daily fluctuations of stock prices. Until recently, this task has been nearly impossible as the statistical methods used are too weak to provide convincing evidence that proves the random walk hypothesis is false. This paper will empirically prove that there is dependency in the daily movements of stock market data from the New York Stock Exchange (NYSE) and NASDAQ (NDAQ) using results obtained from a trained deep neural network. The network is able to discriminate between real and randomly permuted closing price sequences with 70.55% accuracy. This paper will also speculate about various ways these results could be interpreted so that they are valuable for prediction.

1. Background

In Fama's article "Random Walks in Stock-Market Prices" (1965), he states that "the chartist would like to have a more sophisticated method for identifying movements – a method which does not always predict the termination of the movement simply because the price level has temporarily changed direction" (Fama, 1965). Recently, deep neural networks have been shown to be powerful at finding abstract patterns in datasets. They do this by learning an arbitrary function which is a composition of layered features. The deeper layers are able to disentangle complex patterns in high dimensional data. Applied to daily stock price changes, these neural networks are a relatively new tool for analyzing financial data.

2. Discussion of Dependence

In order to show that daily stock price changes exhibit non-random dependence, a deep neural network is taught to discriminate between real and synthetic closing price data. The dataset is constructed according to the flowchart in Figure 1.

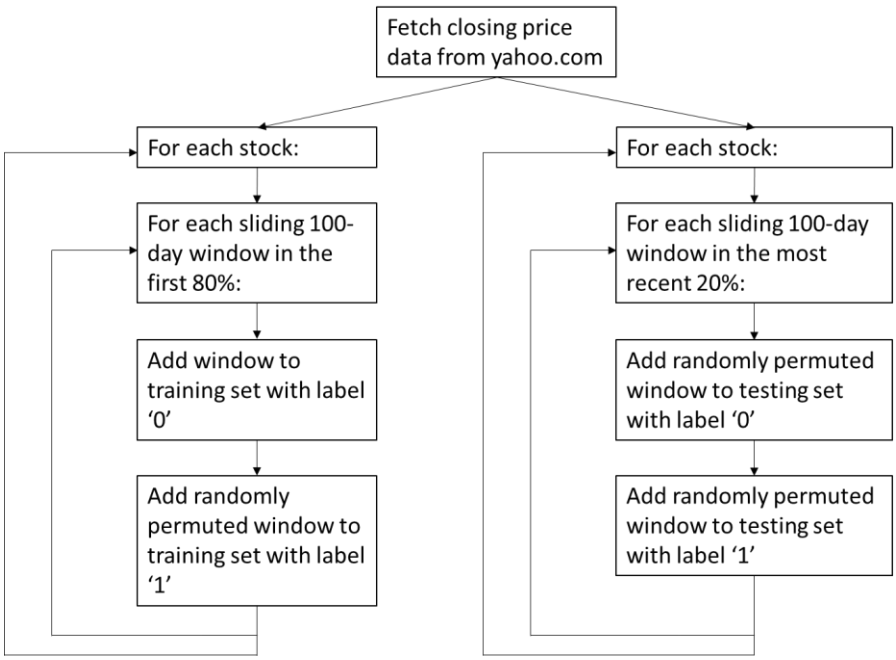

Figure 1. Procedure for generating dataset

First, the daily prices for June 2, 2006 to June 2, 2016 are downloaded from the Yahoo financial database. Then the daily price changes are normalized using the formula $(day_n/day_{n-1}) - 1$. Next, all possible 100 day windows are generated from the daily price changes using a sliding window. Each unaltered sliding window is labeled '0', and a randomly permuted copy is labeled '1'. Note that there are an exactly balanced number of data points with label '0' and label '1'. It is important to split the total dataset into a training portion and a testing portion which are independent of each other. The first chronological 80% of the windows are used for training, and the most recent 20% are used for testing. In this way, approximately 2600 stocks * 2500 windows * 2 permutations * 0.8 ≈ 10.4M {data, label} pairs are created for training. Similarly, 2600 stocks * 2500 windows * 2 permutations * 0.2 ≈ 2.6M {data, label} pairs. This dataset is illustrated Figure 2.

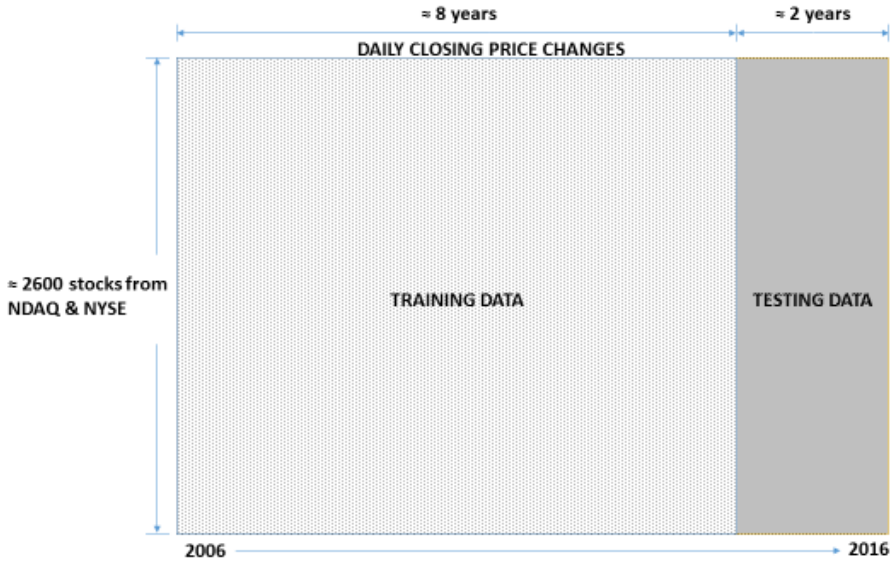

Figure 2. Dataset split between training and testing

As a control, another dataset is created using the same procedure as described in the previous paragraph, except that both label ‘0’ and label ‘1’ are randomly permuted. The expectation is that since random data does not contain any information, the deep neural network should not be able to discriminate between the labels.

To help visualize the dataset, Figure 3(a-d) show a sample 100-day window of AAPL stock from the test dataset. Figure 3a is the real unaltered data, and Figure 3(b-d) are randomly permuted copies of the same data. Clearly, discriminating between real and permuted data is a challenging task.

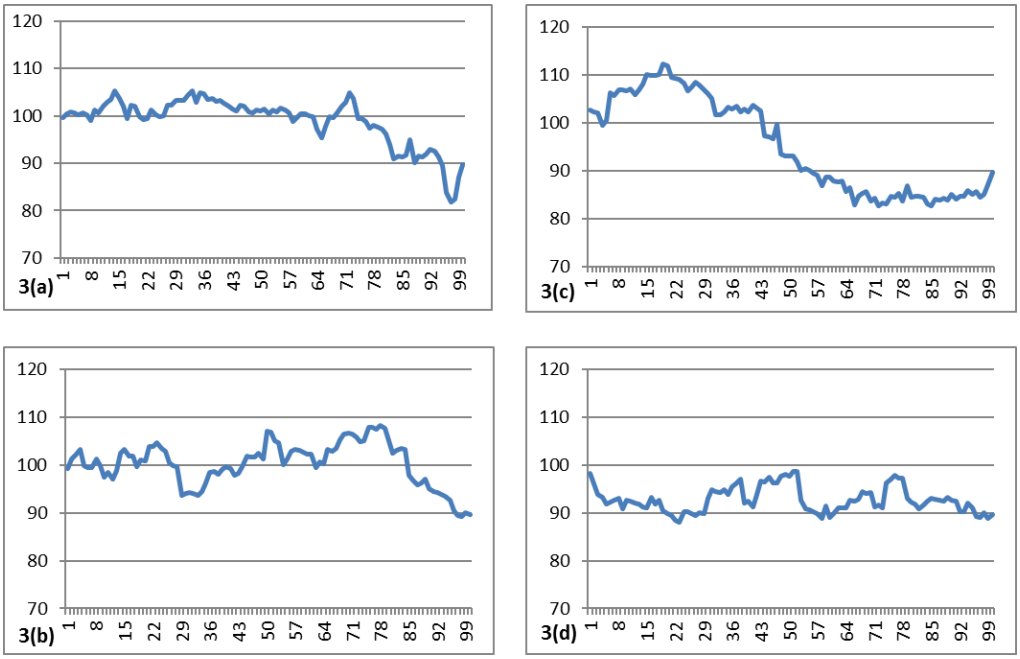

Figure 3(a-d). 100-day closing price windows for AAPL

A deep neural network is then trained using the dataset described in the previous paragraph. The network will try to learn to predict the label given a 100-day window of closing price changes. Remember that the label is ‘0’ if the window of closing price changes is unaltered, and ‘1’ if it is randomly permuted. The network is shown in Figure 4.

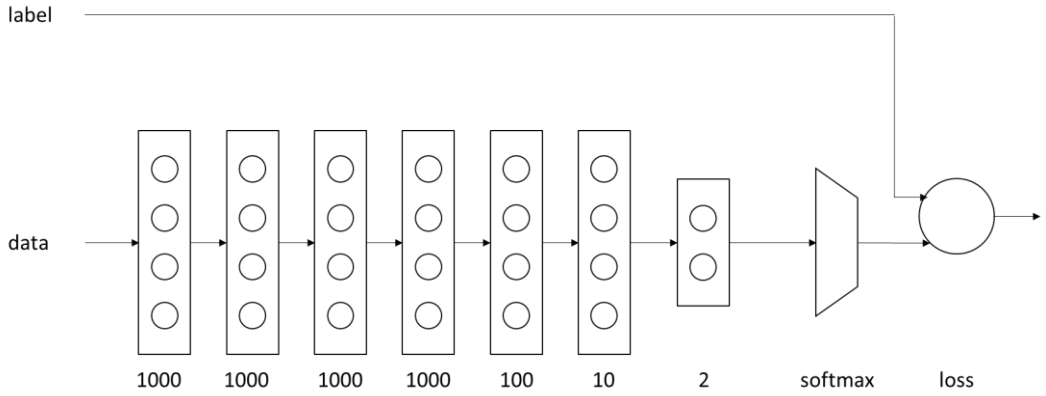

Figure 4. Deep neural network architecture

It takes as input a 100-dimensional data vector of closing price changes which corresponds to the 100-day window. The network contains six layers of fully connected inner product units. The six layers are composed of 1000-1000-1000-1000-100-10 units. The depth of the network allows it to learn increasingly abstract representations of the input data. The network narrows at the deep layers in order to help prevent overfitting by constraining the information flow. The second to last layer contains 10 units which form a compact representation of the features in the 100-dimensional input vector. A final layer contains 2 units which feed into a softmax, followed by a loss function which is driven by the ground truth label from the dataset. This is a state of the art deep neural network.

The network is implemented and trained using the Caffe library on a GPU enabled laptop running Ubuntu linux. The Caffe solver uses the Adam algorithm with default parameters. Adam is an adaptive gradient descent algorithm. The dataset is shuffled during training and the batch size is set to 50. The network is trained for 10M iterations with testing performed every 100K iterations. The total training time was approximately 15 hours.

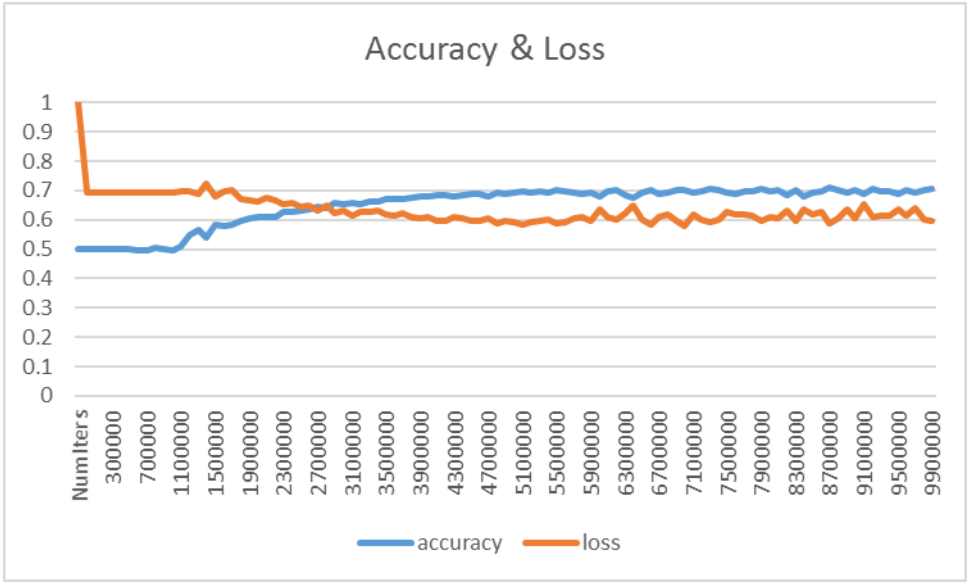

Figure 5. Training results showing 70.55% accuracy

The results for experiment are shown in Figure 5. The deep neural network was able to successfully identify real vs. randomly permuted closing price windows with a 70.55% accuracy. This is a key result of this paper. Note that the accuracy number comes from the testing phase, which was run on data not used during training. Therefore, there is no possibility that the network just memorized the dataset.

The results for the control are shown in Figure 6.

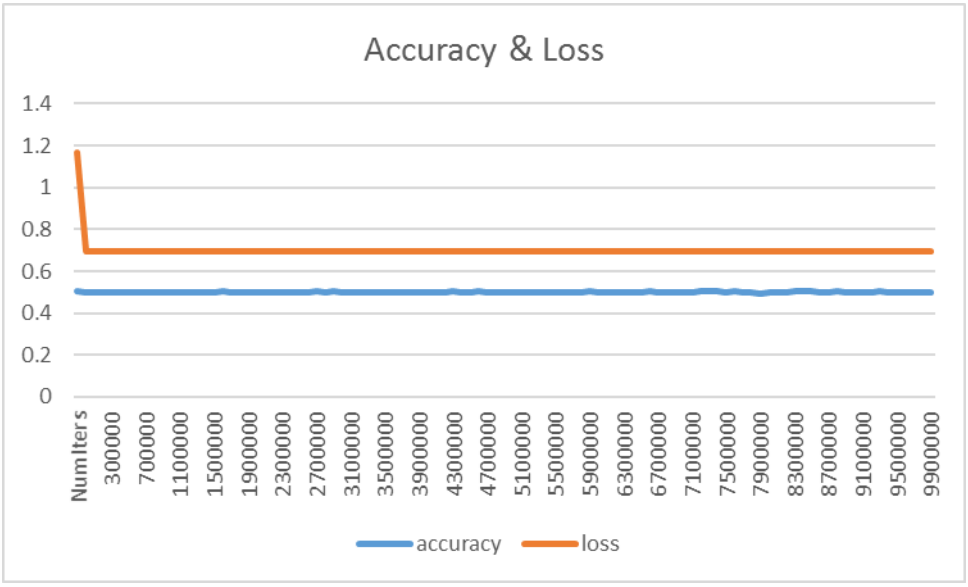

Figure 6. Training results with both labels randomly permuted

As expected, the accuracy after training is 50%. This confirms that the deep neural network is unable to find a pattern in random data.

Referring back to Figure 3(a-d), each of the four windows was fed into the trained network, and the output probabilities were: 0.998061, 0.036104, 0.091689, and 0.042465 respectively. This example serves as a check to show that the deep neural network is generating correct results.

Based on these results, the deep neural network is picking up on patterns that are undetectable to humans. Although it is not possible to characterize the patterns due to the opaque nature of the deep neural network, it is empirically clear that the network has learned patterns because it is able to discriminate between real and randomly permuted data.

3. Discussion of Meaning

Section 2 showed that there is dependence in the daily changes of stock prices. The second element of Fama's random walk theory states that even if there is a dependence, there is no useful way to exploit that dependence. Fama refers to this as meaning.

Building on the results from Section 2, this section will attempt to use the trained deep neural network to make meaningful predictions about the future. The basic idea is to treat the last 10-dimensional layer of the network as a feature vector. Using these feature vectors, k-means is used to create cluster centroids from the training dataset. The cluster centroids are then applied to the features from the testing dataset to assign each window to a cluster. These clusters are then used to append the next day price change after the last day of the window to a list. There is a separate list for each cluster. Once the lists are fully populated, the mean and standard deviation are calculated per list. In this way, it is possible to see the next day change statistics as a function of the cluster. If these statistics deviate significantly from the overall market average, there may be a way to exploit this information. This procedure is illustrated in Figures 7 and 8.

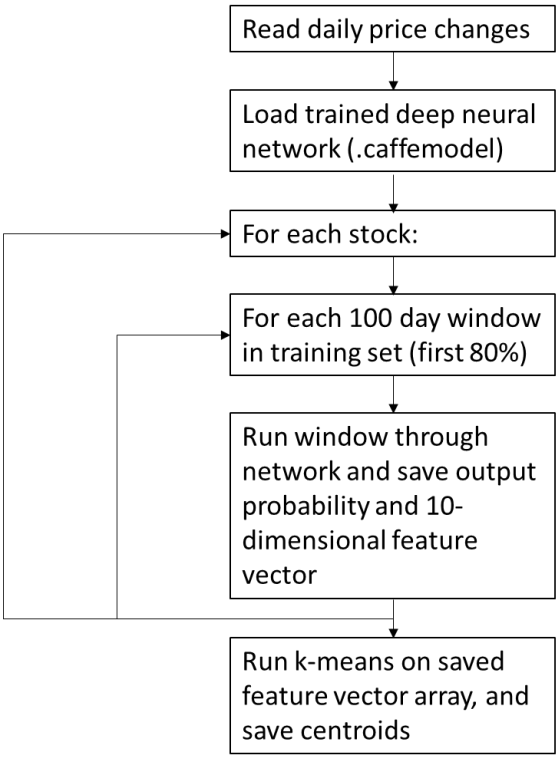

Figure 7. Finding centroids using k-means

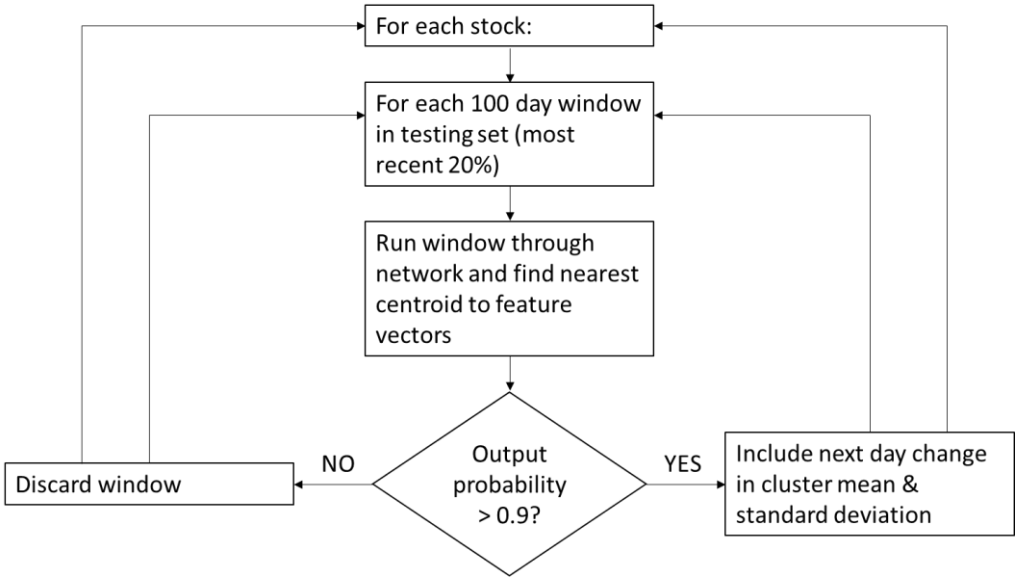

Figure 8. Using the centroids to find high probability clusters with test data

The previously described clustering algorithm was run with the number of clusters k=50. The results are shown in Table 1. The first column in Table 1 refers to the cluster number. The special cluster 'overall market' refers to all stocks and daily changes in the test dataset. The second column is the number of items in the cluster 'n'. The third column is the mean value of the next daily change for the cluster. The fourth column is the standard deviation of the next daily change for the cluster. The fifth column is a measure of the fraction of up versus down days, computed as ∑sign(next daily change) / n. The rows of Table 1 correspond to each of the non-zero sized clusters generated by k-means.

Consider cluster number 42. It has mean daily change which is more than 3x the market average with a standard deviation which is roughly half the market average. Now consider cluster number 14. When the network indicates this pattern is occurring, the average next day change is negative. Lastly, consider cluster number 33. This cluster has a next day volatility which is 4x the market average.

Given this information, it may be possible to construct a trading strategy which can exceed the market average. For example, near the end of the trading day, the 100-day window including the current day could be fed into the network, producing a cluster number. Based on this cluster number, a trade can be executed based on the expected market direction or volatility.

cluster_number	n	mean	std	quantized mean
overall market	1334459	0.000201836	0.0476476	0.007108499
40	20944	0.000457478	0.0242995	0.022536287
49	19928	0.000707479	0.022901	0.029506222
32	18544	0.000927559	0.0707432	0.019089733
3	18047	0.000796483	0.0257643	0.038898432
30	17518	-6.02E-06	0.022457	0.006393424
11	15485	0.000570583	0.0235353	0.022408783
16	15150	0.000282539	0.0243102	0.026732673
42	14143	0.000627832	0.0298771	0.033302694
43	13966	0.00115363	0.0912134	0.017256194
17	13571	0.000320243	0.0234858	0.03212733
47	13029	0.000710993	0.033996	0.031468263
20	10168	0.000315326	0.0270217	0.024881983
44	6210	0.00248319	0.12952	0.012560386
14	3510	-0.000171644	0.0318736	0.003133903
15	3302	-8.71E-05	0.0337375	0.013628104
33	2997	0.00302106	0.167111	0.011011011
21	2841	0.00134054	0.0274764	0.039422738
45	2168	0.00278856	0.0569086	0.052583026
29	2027	0.000662511	0.0469722	0.020226936
38	1903	0.000374697	0.0167048	0.037834997
41	965	0.00380551	0.0651681	-0.042487047
10	335	-0.00163728	0.046604	-0.068656716
26	229	0.0120781	0.108034	-0.065502183
23	67	-0.00168325	0.0226066	-0.059701493
13	22	0.000857478	0.077323	0.181818182
36	4	0.0516202	0.02776	1
7	1	0.0343643	0	1

Table 1. Results from k-means clustering

Another way to examine the possibility of meaning in the deep neural network is by looking at the time series for a specific stock. In Figure 9, the daily price of AAPL is plotted for the 2-year test dataset. The output probability from

the network is superimposed along with the daily cluster number. It may be possible to correlate the sequence of probabilities and cluster numbers with subsequent trends in the stock price. This is a topic for future research.

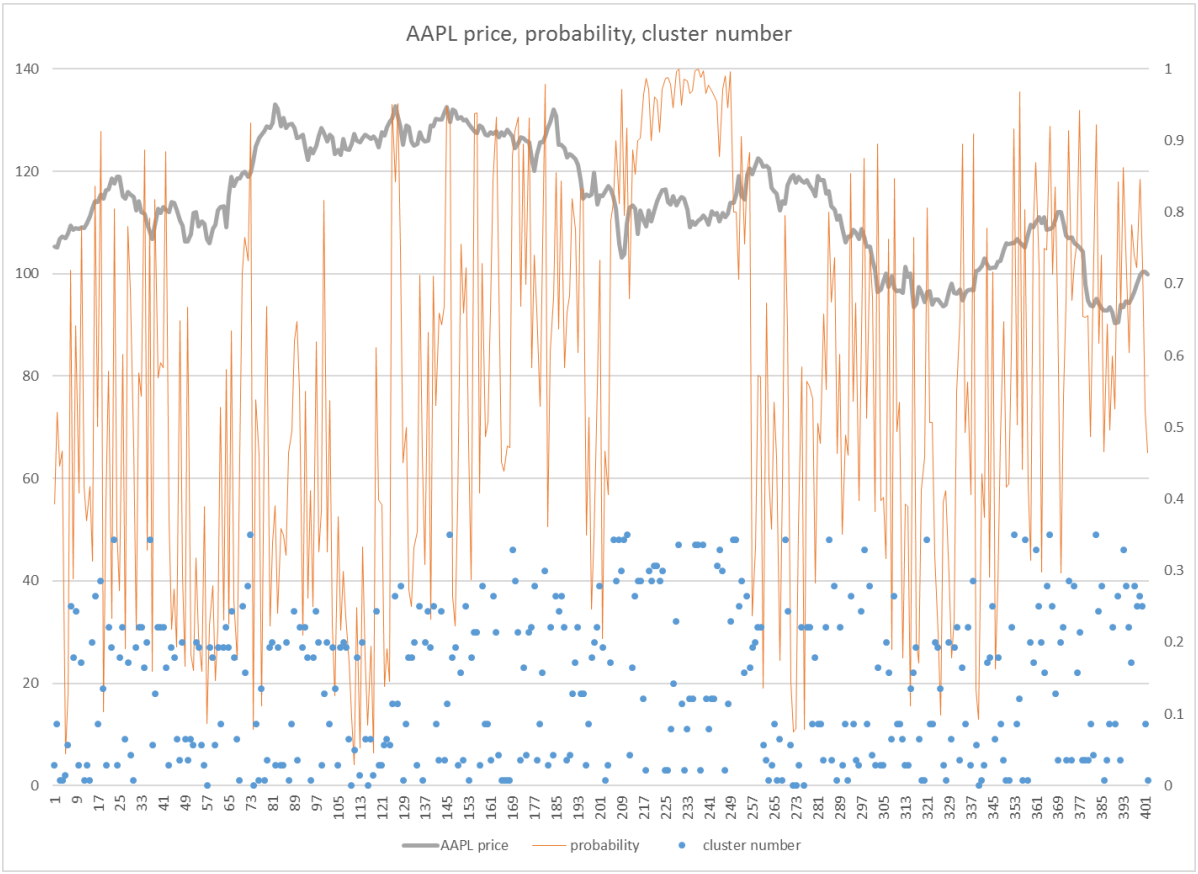

Figure 9.

4. Conclusion

Although the random walk theory is decades old, deep neural networks are a relatively new and powerful tool for finding patterns in data. The main contribution of this paper is a robust application of deep neural network technology to the random walk theory. The elements which make it robust are the careful separation of test versus training data and equal number of unmodified and randomly permuted closing price sequences. If there was no pattern in the data, the probability of guessing whether a sequence is unmodified or randomly permuted is 0.50. After training, the deep network achieved an accuracy of 0.7055. This constitutes conclusive evidence that there are hidden patterns in daily stock price changes.

The next interesting question is whether there is meaningful information in these patterns. This paper contributed towards this question by applying clustering methods to the untangled, 10-dimensional feature vector which is produced by the trained deep neural network. Some future directions for research include using density clustering instead centroid based clustering, predicting future multi-day trends instead of the next day change, and using sequences in feature space to predict future price changes.

In order to help validate these results and facilitate collaboration, the code has been committed to http://www.github.com/mattliston/randomwalk.

Work Cited

@ARTICLE{Fama65randomwalks,
   author = {Eugene F. Fama},
   title = {Random Walks in Stock-Market Prices},
   journal = {Financial Analysts Journal},
   year = {1965},
   volume = {21},
   pages = {55--59}
}

@article{jia2014caffe,
  Author = {Jia, Yangqing and Shelhamer, Evan and Donahue, Jeff and Karayev, Sergey and Long, Jonathan and Girshick, Ross and Guadarrama, Sergio and Darrell, Trevor},
  Journal = {arXiv preprint arXiv:1408.5093},
  Title = {Caffe: Convolutional Architecture for Fast Feature Embedding},
  Year = {2014}
}