Report for CS971 - Evolutionary Computation for Finance

Assignment 3 – Forecasting and Prediction

Garry James McBride

20125102041663

University of Strathclyde 2018 – 2019

Introduction

The following report is an investigation on how to use Neural Networks when generating predictive functions for time series data from a choice of stock options selected. Using the programming language of R, the chosen data will be studied and understood by the means of averages and simple plots for general understanding before the Neural Network is used for the prediction. Prediction stock options is widely used in the Financial world to determine how secure a stock is, whether it is for an investor wanting to know which stock will perform well in the future or be financially secure, before or after they have attained said stock.

The Data Set

The chosen data has 888 entries for the stock value, over the course of a year. The data is from IBM's stock values, assessed over the time period. The International Business Machine Corporation (IBM) was founded in the United States of America in 1911 and was branded IBM in 1924. Manufacturing and selling computer hardware, Software, Middleware and contributes to a large amount of research in the technology industry. Listed in the New York Stock exchange and being one of the worlds most valuable companies, they will have many investors and new stock options around the world. Using neural networks to predict the stock options on such a massive organization will be helpful for potential current/future investors.

Mean	1.240212
Medium	1.2225
Range	-30.368 - 30.097

Table 1

Shown above in Table 1, the ***mean***, ***medium*** and ***range*** of the data is shown. The lowest and highest entry are a large number apart, roughly around 60, which means the stocks have dropped to a much lower value than expected but also has reached a higher number, the stock prices could be considered 'risky' based on this first look at the data.

The data is being analysed over the course of 12 months, and when put into months of the year, the rows are combined and scaled down to a total of 74 per month, shown below in ***Table 2***.

	Jan	Feb	Mar	Apr	May
1	-1.043	-2.478	-12.283	8.598	3.626
2	2.278	7.796	16.833	7.564	13.890
..	..	..	..	..	..
73	-5.779	5.791	-0.540	10.932	1.579
74	-0.612	-7.523	4.323	16.567	10.454

Table 2

Following the split into 12 months, the calculation of various moving averages (SMA) of the times series data was analysed, shown below in ***Table 3***.

	Jan	Feb	Mar	Apr	May
1	NA	NA	NA	NA	NA
2	1.825833e+00	2.682000e+00	5.108333e+00	5.022167e+00	5.877500e+00
..	..	..	..	..	..
73	1.990333e+00	3.180750e+00	3.521333e+00	3.128250e+00	2.615667e+00
74	5.210917e+00	4.101417e+00	4.506667e+00	4.976250e+00	5.715833e+00

Table 3

After the discovery of missing values, they were removed from the data set from the months of ***January*** – ***November***. Missing values or "missing data" can have a significant effect on the conclusions that can be drawn from the collected data. ***Figure 1*** Looks at current and historical values of the data entity, the share price. Viewing the data on a graph like this shows a visual trend in the time series across the length of time (12 months), investigating the past helps when predicting as past events can influence future outcomes or values. As shown the data is varied and the value often changes with no pattern to its higher and lower values.

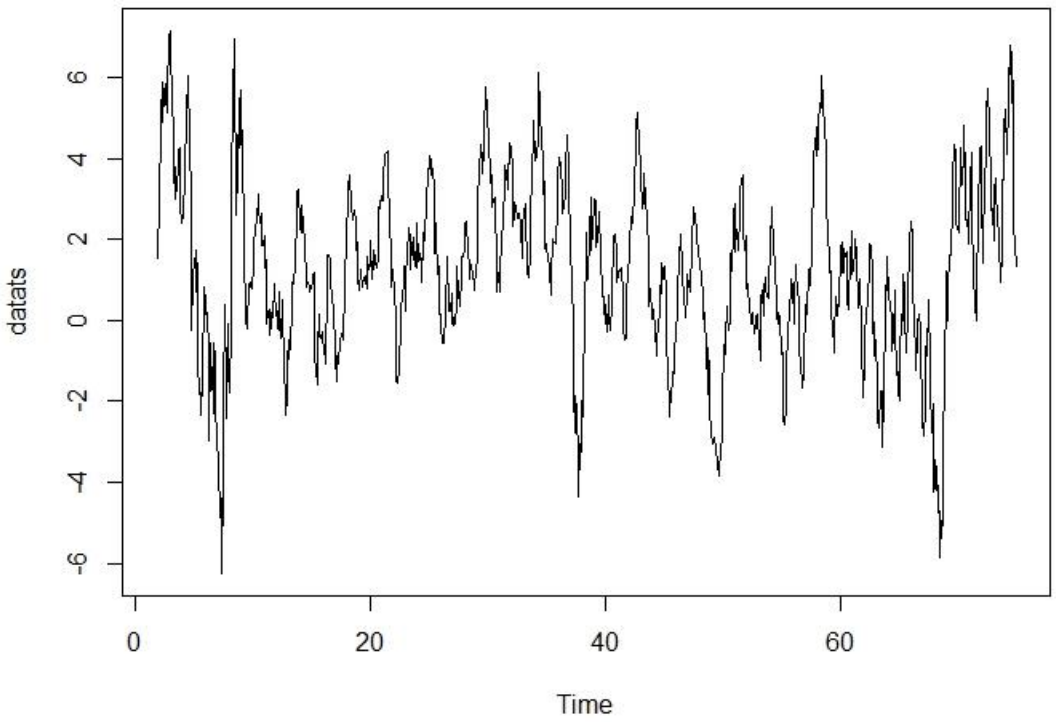

Figure 1

After a general visualization of the data, it is important to look more in-depth with aspects such as;

Trend *– A complicated and useful technique which looks at past sales, it used to determine possible trends from the given data to extrapolate what could be the future values of the stocks.*
Seasonality *– One of the most frequently used statically patterns, which creates a predictable cyclic version of the data, this depends on the time of the year.*
Cycles *– Used to predict changes in the data over the period collected.*

Shown below in ***Figure 2*** are these aspects with the data from IBM. When compared next to the data, you can see the methods taking aspects from the original data and applying what they do. This is a great first analysis for seeing what predictions of activity can be found from a small time period and first investigation.

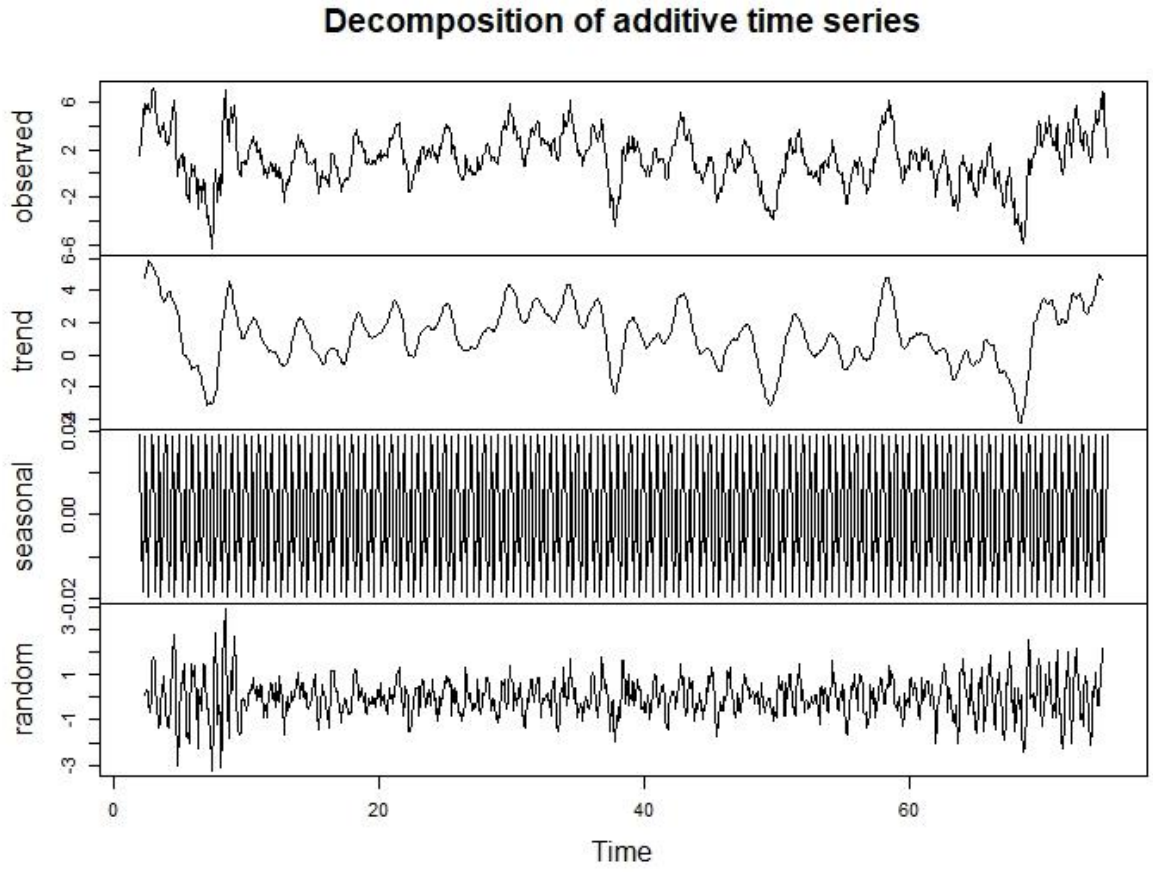

Figure 2

The data must be split into both training and testing. Training set is 75% and the reaming for testing. It is important to split the data, the training data is used for all the methods and changed to suit the Neural Network (NN), but it is interesting to see the results from the NN on unseen or new data… hence the training set of 25% as it has not been changed nor manipulated.

Neural Network

Neural networks are used for a vast amount of machine learning techniques such as; voice recognition or computer vision. The Neural Network is fed information and passes it through layers to find its strongest matching value or asset. Other results will be calculated at the same time, and they will be somewhat accurate, or wrong, but the strongest input based on the target (the output) will be passed through and end up matching with the intended target,

based on using weights and biases. The rest of the information fed to the NN will make it to the end, but they will not be the result we are looking for.

The R package “Neural Net” is used to produce the Network. The package can be seen and understood better within the code using the ‘help’ command, this gives a better insight to the package for experimentation and research for the best possible results.

To generate training data, inputs or windows are used to select certain parts of the data and slide. To generate a prediction, the window size must have a good amount of values to be accurate, 2 inputs would not be enough to make a prediction, so experimentation was done with using lower or higher inputs to change the range of the windows.

The neural network uses hidden layers to get an accurate result from the inputted data. The more hidden layers used does not necessarily mean the network is more accurate or faster, more layers take longer to be processed and the predication could be found with fewer layers.

Training Set Results

Inputs	*Hidden Layers*	*Threshold*	*Error*
9	15 15 15 15	0.1	3.54587
8	20 10 10	0.05	5.25689
7	20 20	0.01	6.25455
7	20 20 20	0.1	1.41915
9	15 15 20	0.01	2.21851
8	15 15 15	0.1	1.25486
9	15 10 20	0.1	9.25865

Table 4

Shown above in ***Table 4***, the results from 7 runs of the Neural Network with different input sizes, threshold rate and number of hidden layers. From the results it seems the higher the input and higher the threshold rates the better the error results. Using only 3 hidden players, with a 0.1 threshold and 9 inputs, the errors are much lower than others with either the same amount of inputs but a smaller number of hidden layers and a very low threshold value.

The Neural Network was not fast, it took long to run, and errors presented themselves when not using ‘etepmax = 1e+06’ and only using 6 inputs. Oddly the NN worked well and had a run

time of 5 minutes but with 3 hidden layers of 20 each, any less than 20 the NN would take longer to run and would end with an error as it is not complicated enough to work. Inputs were changed from 7-9 and the results with lower inputs depended on the threshold and number of layers. Another strange run with 4 layers of 15 did not work as well as 3 hidden layers of 20, the NN must have been too big to process and after a lengthy run it crashes, and an error presented itself.

Using the best results possible, they threshold could not be made bigger, as anymore than 0.1 errors allowed would not produce accurate results and using more inputs would cause the NN to process much more and take longer.

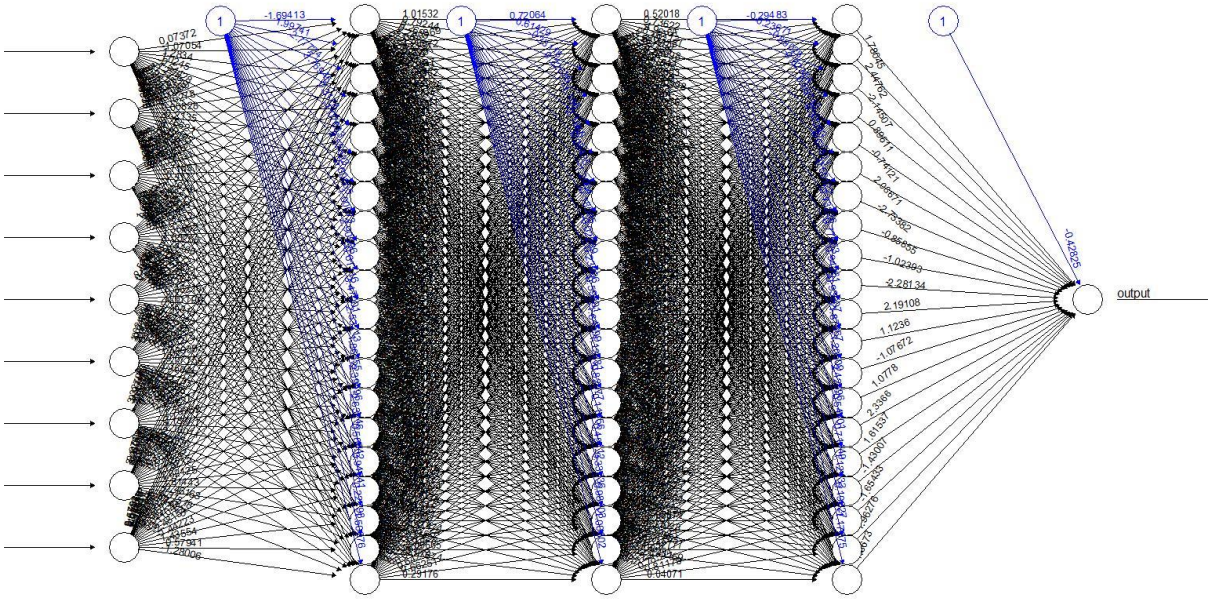

Figure 3

Testing

Results from testing

Inputs	*Hidden Layers*	*Threshold*	*Error*
9	15 15 15 15	0.1	15.25545
8	20 10 10	0.05	17.25456
7	20 20	0.01	19.55245
7	20 20 20	0.1	18.55968
9	15 15 20	0.01	21.25422
8	15 15 15	0.1	22.25789
9	15 10 20	0.1	19.25876

Using the same number of hidden layers and input sizes, the testing data was run and produced negative results for testing shown in ***Table 5***. The best results from training with the same number of attributes regarding inputs and layers, yet again produced the better results compared to the others. Based on the difference between the results, the NN did not work as well on unseen data as the training set.

From the analysis of the results from both testing and training, when using the NN on future or unseen data can't be the same as training. The code has to be changed and manipulated to achieve better results instead of using the same as training.

Comparisons of the network performance with other approaches

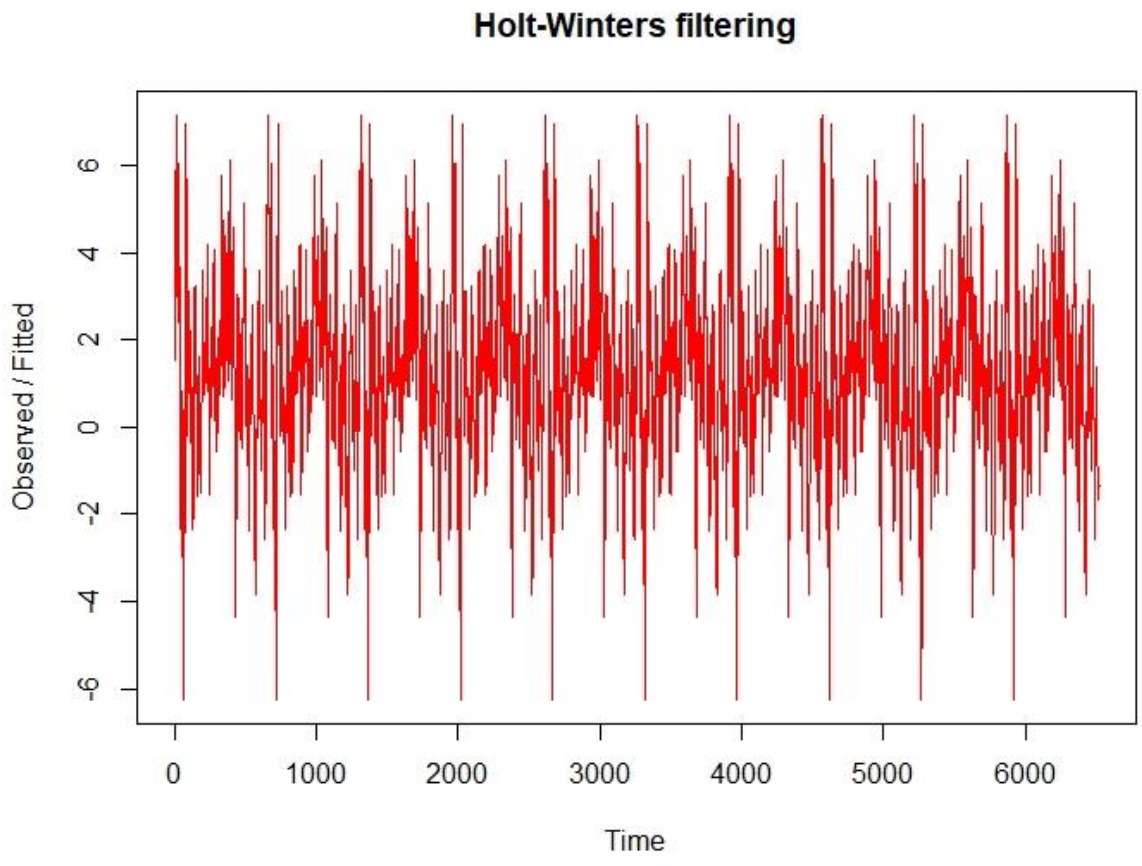

Figure 4

From ***Figure 4*** above, the method of Holt-winters filtering has been used on the time series data. Hot-Winters is used by means of the smoothing parameters, which are then chosen and will minimize the sum of all predictions, one step ahead of them.

The results show the predictions are not as accurate as the data, they are quite erratic and change from higher to lower a lot more than the data before hot-winters. Although this method does follow a trend, and patterns can be seen from the visual graph, it is not an accurate way of predicting the data set.