**Data Mining Algorithms for Financial Domain**

**Megan Smith**

**Sundarkumar Ramsundar**

**Table of Contents**

Novelty of the Problem Page 3

Data Preprocessing Page 3

Problem Formulation Page 3

Data Mining Algorithms, Evaluation, and Experimental Results Page 4-

Data Summary Page 4

Histograms Page 6-7

Density Page 8-9

Scatter Plots Page 10-11

Pair Plots Page 12-13

Forecasting Page 14-20

Decomposition Page 21-22

Classification Page 23-25

Clustering Page 26-28

Neural Networking Page 29-32

References Page 33

**Novelty of the Problem**

Financial markets are highly unpredictable. This is because of various known and unknown factors influencing the markets in direct and indirect ways. We want to apply data mining techniques to this problem domain to find out whether we can predict future values of financial entities from previous data.

Financial predictability is one of the corner stones for any organization or individual. Most of the individual investors in stock markets and other financial markets go by speculation and incomplete analysis of previous data. This is mainly because of the number crunching required for analyzing the previous data, which is time consuming and an individual is unable to afford that time for the decisions. Impact of wrong decision by the customers affects the feedback loop of demand-supply in the market and this makes the markets further unpredictable. Hence, it’s a vicious cycle.

Now a days computing power is cheaper, we can use the data mining methods, which are iterative and numerical, to solve the problem and suggest customers a way to invest or purchase optimally. This will help the users to save money and get better goods in the short term, while making the markets more predictable and transparent in the long term.

**Data Preprocessing**

We first performed data cleanings on the Dow Jones Industrial Average dataset to remove all the null values from the data and skipped incomplete rows for certain algorithms. Following the cleaning, we performed a number of data reductions depending on the algorithm or desired plot graph of the data. For instance, several plots only used portions of the data for reason such as: saving space and testing the accuracy of forecasting algorithms. We also had one instance of data integration involving the action attribute for the Dow Jones Index dataset. The action attribute did not originally come with the dataset; however, due to the want of developing a decision tree for predicting whether to buy, hold, or sell a particular stock, we researched and added in the approximate values.

**Problem Formulation**

The problem formulation for each problem is defined separately for each data mining algorithm in the succeeding section.

**Data Mining Algorithms, Evaluations and Experimental Results**

We began our problem formulation by exploring the data. Using the R function summary(), we returned the minimum, maximum, mean, median, and the first (25%) and third (75%) quartiles. For categorical variables, summary() also shows the frequency of every level.

**The summary for the Dow Jones Industrial Average dataset:**

> summary(mydata)

DATE VALUE

Min. :2005-04-01 Min. : 6547

1st Qu.:2007-09-26 1st Qu.:10724

Median :2010-03-24 Median :12215

Mean :2010-03-24 Mean :12417

3rd Qu.:2012-09-17 3rd Qu.:13540

Max. :2015-03-19 Max. :18289

**The summary for the Dow Jones Index dataset:**

> summary(mydata)

QUARTER STOCK DATE OPEN

Min. :1.00 AA : 25 Min. :2011-01-07 Min. : 10.59

1st Qu.:1.00 AXP : 25 1st Qu.:2011-02-18 1st Qu.: 29.83

Median :2.00 BA : 25 Median :2011-04-01 Median : 45.97

Mean :1.52 BAC : 25 Mean :2011-03-31 Mean : 53.65

3rd Qu.:2.00 CAT : 25 3rd Qu.:2011-05-13 3rd Qu.: 72.72

Max. :2.00 CSCO : 25 Max. :2011-06-24 Max. :172.11

(Other):600

HIGH LOW CLOSE VOLUME

Min. : 10.94 Min. : 10.40 Min. : 10.52 Min. :9.719e+06

1st Qu.: 30.63 1st Qu.: 28.72 1st Qu.: 30.36 1st Qu.:3.087e+07

Median : 46.88 Median : 44.80 Median : 45.93 Median :5.306e+07

Mean : 54.67 Mean : 52.64 Mean : 53.73 Mean :1.175e+08

3rd Qu.: 74.29 3rd Qu.: 71.04 3rd Qu.: 72.67 3rd Qu.:1.327e+08

Max. :173.54 Max. :167.82 Max. :170.58 Max. :1.453e+09

PERCENT\_CHANGE\_PRICE PERCENT\_CHANGE\_VOLUME\_OVER\_LAST\_WK PREVIOUS\_WEEKS\_VOLUME

Min. :-15.42290 Min. :-61.4332 Min. :9.719e+06

1st Qu.: -1.26550 1st Qu.:-19.8043 1st Qu.:3.068e+07

Median : 0.00000 Median : 0.5126 Median :5.295e+07

Mean : 0.05642 Mean : 5.5936 Mean :1.174e+08

3rd Qu.: 1.65089 3rd Qu.: 21.8006 3rd Qu.:1.333e+08

Max. : 9.88223 Max. :327.4089 Max. :1.453e+09

NA's :30 NA's :30

NEXT\_WEEKS\_OPEN NEXT\_WEEKS\_CLOSE PERCENT\_CHANGE\_NEXT\_WEEKS\_PRICE

Min. : 10.52 Min. : 10.52 Min. :-15.4229

1st Qu.: 30.32 1st Qu.: 30.46 1st Qu.: -1.2221

Median : 46.02 Median : 46.12 Median : 0.1012

Mean : 53.70 Mean : 53.89 Mean : 0.2385

3rd Qu.: 72.72 3rd Qu.: 72.92 3rd Qu.: 1.8456

Max. :172.11 Max. :174.54 Max. : 9.8822

DAYS\_TO\_NEXT\_DIVIDENE PERCENT\_RETURN\_NEXT\_DIVIDEND ACTION

Min. : 0.00 Min. :0.06557 BUY : 5

1st Qu.: 24.00 1st Qu.:0.53455 HOLD: 2

Median : 47.00 Median :0.68107 SELL: 5

Mean : 52.53 Mean :0.69183 NA's:738

3rd Qu.: 69.00 3rd Qu.:0.85429

Max. :336.00 Max. :1.56421

We measured the distribution of the two datasets with a histogram using the R function hist().

**The histogram for the Dow Jones Industrial Average dataset:**

Here we see the distribution of the numeric values of the Dow Jones Industrial Average throughout our dataset.

> hist(mydata$VALUE, sub = "Histogram of DJIA dataset")



**The histogram for the Dow Jones Index dataset:**

Here we see the distribution of the percent of return for all the stocks in our dataset. We focused on percent of return most often to determine which stock will produce the greatest rate of return in the following week. Stocks with greater rates of return in the following week should be sold in order to gain the greatest profit.

> hist(mydata$PERCENT\_CHANGE\_NEXT\_WEEKS\_PRICE, sub = "Histogram of DJI dataset" )



We also produced a distribution of the data with density graphs using the R function density().

**The density for the Dow Jones Industrial Average dataset:**

Here we see a more fluid version of the Dow Jones Industrial Average histogram, displaying the distribution of the numeric values of the Dow Jones Industrial Average throughout our dataset.

> plot(density(mydata$VALUE), sub = "Density of DJIA dataset")

****

**The density for the Dow Jones Index dataset:**

Here we see a more fluid version of the Dow Jones Index histogram, showing the distribution of the percent of return for all the stocks in our dataset.

> plot(density(mydata$PERCENT\_CHANGE\_NEXT\_WEEKS\_PRICE), sub = "Density of DJI dataset")



Next we made a general plot graph displaying the relationship between our datasets versus the time in which they occur using the R function plot().

**The general plot for the Dow Jones Industrial Average dataset:**

Here we see a graphical representation of the Dow Jones Industrial Average Value as it progresses through 2005-2015.

> plot(mydata$DATE, mydata$VALUE, sub = "DJIA dataset")



**The general plot for the Dow Jones Index dataset:**

Here we see a representation of the percent return for each stock per time interval (by weeks).

> plot(mydata$DATE, mydata$PERCENT\_CHANGE\_NEXT\_WEEKS\_PRICE, sub = "DJI dataset")



We included a pairs plot (a matrix of scatter plots) of both datasets to thoroughly represent both datasets in every circumstance for future reference.

**The pairs plot for the Dow Jones Industrial Average dataset:**

Here we see a matrix of scatter plots using all the attributes of the Dow Jones Industrial Average dataset : DATE, VALUE.

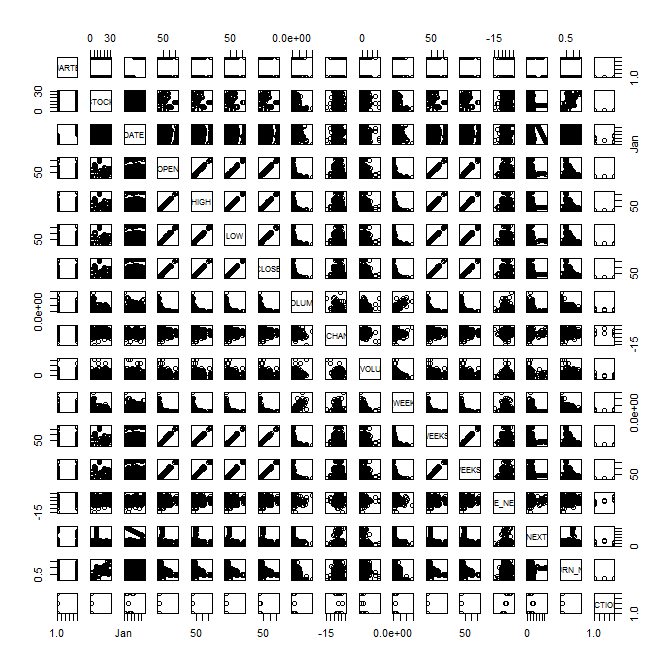
> pairs(mydata)



**The pairs plot for the Dow Jones Index dataset:**

Here we see a matrix of scatter plots using all the attributes of the Dow Jones Index dataset : QUARTER, STOCK,DATE,OPEN,HIGH,LOW,CLOSE,VOLUME,PERCENT\_CHANGE\_PRICE,PERCENT\_CHANGE\_VOLUME\_OVER\_LAST\_WK,PREVIOUS\_WEEKS\_VOLUME,NEXT\_WEEKS\_OPEN,NEXT\_WEEKS\_CLOSE,PERCENT\_CHANGE\_NEXT\_WEEKS\_PRICE,DAYS\_TO\_NEXT\_DIVIDEND,PERCENT\_RETURN\_NEXT\_DIVIDEND, ACTION

> pairs(mydata)



One of the main data mining algorithms we used was time series forecasting in order to predict future events based on historical data. We used a popular model for time series forecasting called autoregressive integrated moving average (ARIMA), which is a generalization of an autoregressive moving average model (ARMA). Our dataset, being economic data, can be considered seasonal/dependent on a non-stationary price level. Therefore, we utilize the ARIMA model to reduce the non-stationarity by applying an initial differencing step. We used as our (p,d,q) values (2,0,2) because we believed it provided the best forecast with the smallest Akaike Information Criterion (AIC) value. This AIC value is a measure of the relative quality of a model, indicating the preferred model as the one with the minimum AIC value.

**The general plot and forecast for the Dow Jones Industrial Average dataset:**

Here we see a general plot of the Dow Jones Industrial Average Value versus Time. This graph shows all of the current data in order to provide a reference for the forecast plot that follows.

> plot(x.ts2,col=3,ylim=range(6000:20000), xaxt="n",ylab = "Value", sub = "DJIA dataset" )



Here we see the forecast of the Dow Jones Industrial Average Value versus Time for the first few days of August 2014 (starting at data tuple 2351).

> plot(forecast(arima(x.ts,order=c(2,0,2)),10), xaxt="n", sub ="Forcast of DJIA dataset", ylab = "Value", xlab = "Time")



Here we see a summary of the forecast results. A set of error measures are provided within this summary, some of those errors including mean absolute error (MAE), root mean squared error (RMSE), mean absolute percentage error (MAPE), and mean absolute scaled error (MASE). Next the summary lists the calculated forecasts. For each forecast point, confidence values are displayed for the low 80%, high 80%, low 90%, and high 90% confidence levels. All together though, the evaluation of the forecast can ultimately boil down to its comparison to the actual plot of the dataset, in which it accurately predicts the Dow Jones Industrial Average Values for the first few data values in August 2014.

> summary(forecast(arima(x.ts,order=c(2,0,2)), 10))

Forecast method: ARIMA(2,0,2) with non-zero mean

Model Information:

Call: arima(x = x.ts, order = c(2, 0, 2))

Coefficients:

ar1 ar2 ma1 ma2 intercept

0.0601 0.9381 0.8335 -0.0940 12088.067

s.e. 0.0711 0.0710 0.0744 0.0221 1961.292

sigma^2 estimated as 16176: log likelihood = -14724.82, aic = 29461.64

Error measures:

ME RMSE MAE MPE MAPE MASE ACF1

Training set 2.895915 127.1854 87.68103 0.01054005 0.7721873 0.0374739 0.003221416

Forecasts:

Point Forecast Lo 80 Hi 80 Lo 95 Hi 95

7.438356 16594.47 16431.48 16757.46 16345.19 16843.75

7.441096 16586.70 16368.11 16805.30 16252.39 16921.02

7.443836 16585.97 16322.91 16849.02 16183.66 16988.28

7.446575 16578.64 16278.05 16879.22 16118.93 17038.34

7.449315 16577.50 16243.33 16911.68 16066.42 17088.58

7.452055 16570.56 16206.19 16934.93 16013.31 17127.81

7.454795 16569.08 16176.66 16961.50 15968.92 17169.24

7.457534 16562.48 16144.13 16980.83 15922.67 17202.29

7.460274 16560.69 16117.79 17003.59 15883.33 17238.05

7.463014 16554.39 16088.43 17020.34 15841.77 17267.01

**The general plot for AA Stock and the forecast for the AA Stock of the Dow Jones Index dataset:**

Here we see a general plot of Percent Return of the AA Stock for the 1st Quarter of 2011 versus Time.

> plot(Time, rateOfReturn, type = "b", sub = "AA STOCK - QUARTER 1", ylab = "PERCENT\_CHANGE\_NEXT\_WEEKS\_PRICE")



Here we see a general plot of Percent Return of the AA Stock for both Quarters of 2011. We supplied this graph as a reference for the forecast plot that follows.

> plot(Time2, rateOfReturn2, type = "b", sub = "AA STOCK - QUARTER 1 and 2", ylab = "PERCENT\_CHANGE\_NEXT\_WEEKS\_PRICE")



Here we see the forecast Percent Return of the AA Stock versus Time immediately following Quarter 1, showing us the expected Percent Return of AA Stock at the beginning weeks of Quarter 2.

>plot(forecast(arima(m,order=c(2,0,2)), 10), xaxt="n",type = "b", xlab = "Time",ylab = "PERCENT\_CHANGE\_NEXT\_WEEKS\_PRICE", sub = "Forecast of AA STOCK" )



Here we see a summary of the forecast results predicting the future Percent Returns of AA Stock in the 2nd Quarter of 2011. A set of error measures are provided within this summary, some of those errors including mean absolute error (MAE), root mean squared error (RMSE), mean absolute percentage error (MAPE), and mean absolute scaled error (MASE). Next the summary lists the calculated forecasts. For each forecast point, confidence values are displayed for the low 80%, high 80%, low 90%, and high 90% confidence levels. All together though, the evaluation of the forecast can ultimately boil down to its comparison to the actual plot of the dataset, in which it roughly predicts the Percent Returns of AA Stock for the first few weeks of the 2nd Quarter in 2011.

> summary(forecast(arima(m,order=c(2,0,2)), 10))

Forecast method: ARIMA(2,0,2) with non-zero mean

Model Information:

Call: arima(x = m, order = c(2, 0, 2))

Coefficients:

ar1 ar2 ma1 ma2 intercept

1.1477 -0.9352 -1.8945 1.000 -0.0199

s.e. 0.0926 0.0653 0.3671 0.363 0.0601

sigma^2 estimated as 1.132: log likelihood = -22.24, aic = 56.48

Error measures: ME RMSE MAE MPE MAPE MASE ACF1

Training set -0.06186096 1.064144 0.9019508 -71.63861 97.20537 NaN -0.2046073

Forecasts:

Point Forecast Lo 80 Hi 80 Lo 95 Hi 95

2011.033 -0.4936555 -1.9603908 0.9730797 -2.736834 1.7495224

2011.036 -2.7395995 -4.4573620 -1.0218370 -5.366691 -0.1125085

2011.038 -2.6982193 -4.7542322 -0.6422063 -5.842620 0.4461812

2011.041 -0.5503841 -2.6548485 1.5540803 -3.768885 2.6681164

2011.044 1.8759718 -0.2927908 4.0447344 -1.440864 5.1928079

2011.047 2.6520868 0.2478299 5.0563437 -1.024907 6.3290804

2011.049 1.2737683 -1.2279440 3.7754807 -2.552271 5.0998072

2011.052 -1.0339178 -3.5401480 1.4723123 -4.866866 2.7990304

2011.055 -2.3934666 -5.0391768 0.2522436 -6.439731 1.6527979

2011.058 -1.7957272 -4.5629001 0.9714457 -6.027753 2.4362985

Next we made a time series decomposition to decompose the Dow Jones Industrial Average dataset into trend, seasonal, observed and random components. The trend component stands for long term trend, the seasonal component is seasonal variation, the observed component is a general plot of the data, and the random component is the model of the unpredictable component in the times series data

**The decomposition and seasonal component plot for the Dow Jones Industrial Average dataset:**

Here we see the seasonal component of the Dow jones Industrial Average dataset for one year. Seasonality in a time series can be identified by regularly spaced peaks and troughs which have a consistent direction and approximately the same magnitude every year, relative to the trend.

> plot(f$figure, type="b", xaxt="n", xlab="Time",col=10)



Here we see the total decomposition of the Dow Jones Industrial Average dataset. The trend component shows a smooth, best-fit trend of the data and the random component is the remaining components after removing trend and seasonal factors.

> plot(f, xaxt="n")



Another main data mining algorithm that we used was time series classification. We used this to build a classification model based on labeled time series values and then used the model to predict the label of unlabeled time series values. In the case of the Dow Jones Industrial Average dataset, we used the labeled Average Values of the entire dataset to predict the unlabeled forecast of a single succeeding Average Value. The classification method we used for the following trees is conditional inference trees (ctree), an embedded tree structure regression model.

**The Classification for the Dow Jones Industrial Average dataset:**

Here we see the decision tree built to demonstrate classification of time series with the Dow Jones Industrial Average dataset. We used the first two months of Average Values to build the tree. Using the first 64 data values, one could predict the 65th data value.

> ct <- ctree(VALUE ~ (VALUE+1), data=mydata)

>plot(ct)

****

**The Classification for the first 9 Stocks in the Dow Jones Index dataset:**

Here we see the classification of the first 9 stocks present in the Dow Jones Index dataset in the form of a decision tree. This tree uses all the attributes present in the Dow Jones Index dataset to predict the name of the corresponding stock.

> ct <- ctree(STOCK ~ QUARTER+OPEN+HIGH+LOW+CLOSE+VOLUME+PERCENT\_CHANGE\_NEXT\_WEEKS\_PRICE+PREVIOUS\_WEEKS\_VOLUME+PERCENT\_CHANGE\_PRICE+PERCENT\_CHANGE\_VOLUME\_OVER\_LAST\_WK+NEXT\_WEEKS\_OPEN+NEXT\_WEEKS\_CLOSE+DAYS\_TO\_NEXT\_DIVIDENE+PERCENT\_RETURN\_NEXT\_DIVIDEND, data=mydata2)

> plot(ct)



**The Decision Tree on whether to buy/sell/hold a stock for the Dow Jones Index dataset:**

Here we see a decision tree that predicts on whether or not to buy, sell, or hold a particular stock. Here we used recursive partitioning for classification, regression and survival trees (rpart) as our data mining algorithm, which creates a decision tree to classify members based on independent variables. Every attribute of the Dow Jones Index dataset was used in the tree to predict what action (buy, sell, or hold) should be taken with each stock depends on that stocks information. Only the information pertaining to the stock AA was used to develop the tree.

> plot(fit, uniform=TRUE, main="Decision Tree - Buy or Sell or Hold?")



**Chapter Clustering Using Kmeans**

We considered the financial problem of correlation exchange rates of currencies. There are various factors which affect the exchange rate of currency. Can we identify clusters of currencies which vary in similar fashion? Datamining can answer this question. The method to this is called clustering.

Clustering is an unsupervised learning. One of the famous methods to make clusters is K means. In this method, K is the parameter and the distance measurement method is also variable.

**Dataset**

For our experiment, we choose the following time series data format:

usd/DZD 0.013467 0.013496 0.013501 0.013533 0.013595 0.013604 0.013585 …. (1001 such values)

usd/ARS 0.24284 0.24255 0.24179 0.24240 0.24202 0.24224 0.24155 0.24127 …… (1001 such values)

Here each row represents the value of that currency vs USD ($) for 1001 successive time units.

**Data cleaning**

We skip rows of the input with missing values. So, after cleaning, we get dataset with no missing values. The dataset after cleaning resulted in 74 rows. Each row having 1001 days of exchange rate for a particular currency.

**K-Means parameters**

We vary K between 1 and 30. Number of clusters ideally suited for clustering is . We go well beyond that till .

**Evaluation**

The optimal value of K can be got by trying various values of K (K =1 to K=30) and choosing the one which gives the minimum intra-cluster distance.

**Using Euclidean Distances**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| K | Distance | K | Distance | K | Distance |
| 1 | 7174.9 | 11 | 1873.88 | 21 | 1034.75 |
| 2 | 7174.9 | 12 | 1599.16 | 22 | 1033.88 |
| 3 | 4530.85 | 13 | 1565.53 | 23 | 1034.75 |
| 4 | 4530.85 | 14 | 1565.53 | 24 | 1034.75 |
| 5 | 5060.92 | 15 | 1365.94 | 25 | 1034.75 |
| 6 | 3562.11 | 16 | 1365.94 | 26 | 1034.75 |
| 7 | 3504.17 | 17 | 1365.94 | 27 | 1034.75 |
| 8 | 3504.17 | 18 | 1034.75 | 28 | 1034.75 |
| 9 | 4555.59 | 19 | 1034.75 | 29 | 1034.75 |
| 10 | 3002.32 | 20 | 1034.75 | 30 | 1034.75 |

**Figure Shows the plot of intracluster distance vs K (Euclidean distance)**

**Using Manhattan distance**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| K | Distance | K | Distance | K | Distance |
| 1 | 7175.38 | 11 | 1836 | 21 | 1034.75 |
| 2 | 7175.38 | 12 | 1565.53 | 22 | 1033.88 |
| 3 | 5218.26 | 13 | 1365.94 | 23 | 1034.75 |
| 4 | 5218.26 | 14 | 1365.94 | 24 | 1034.75 |
| 5 | 6463.92 | 15 | 1365.94 | 25 | 1034.75 |
| 6 | 4737.61 | 16 | 1365.94 | 26 | 1034.75 |
| 7 | 4616.25 | 17 | 1365.94 | 27 | 1034.75 |
| 8 | 4604.38 | 18 | 1034.75 | 28 | 1034.75 |
| 9 | 4596.16 | 19 | 1034.75 | 29 | 1034.75 |
| 10 | 3002.32 | 20 | 1034.75 | 30 | 1034.75 |

**Figure Shows the plot of intracluster distance vs K (Manhattan distance)**

**Conclusion**

The clustering of this data follows the general principle of number of clusters . For some other applications, K = 11 would be optimal.

**Chapter Stock Market Prediction using KNN and ANN**

Uncertainty in stock markets is a fact. Can we use datamining methods to improve the predictability of the stock-market. We explored this problem in the context of Dow jones Industrial Average (DJIA). We used K Nearest Neighbor (KNN) and Artificial Neural Network (ANN) to predict the future values of DJIA.

**Dataset**

For our experiment, we choose the following time series data format:

10565.39

10470.51

10456.02

10442.87

10485.65

10405.70

10540.93

10503.76

10404.30

Here each row represents the value of DJIA successive time units.

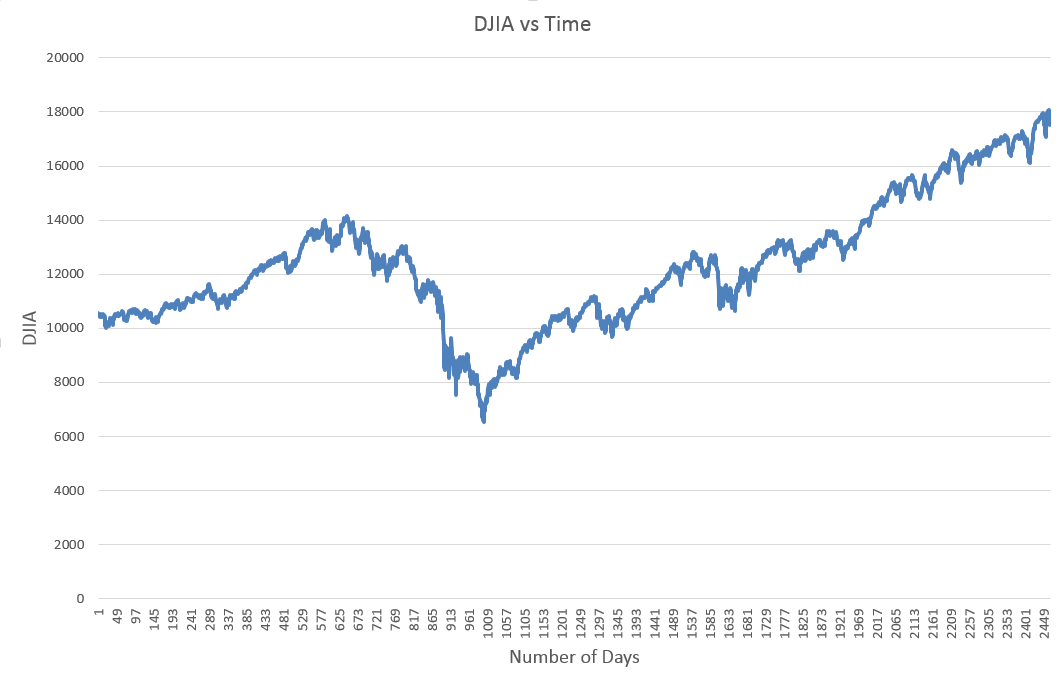


Figure Shows the variation of DJIA over time

**Data for prediction**

x0 = 10565.39

x1 =10470.51

x2 =10456.02

x3 =10442.87

x4 =10485.65

x5 =10405.70

x6 =10540.93

x7 =10503.76

x8 =10404.30

This data will be converted to the format called **tuples**.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tuple** | **Input (xi)** | **Input (xi+1)** | **Input (xi+2)** | **Output (xi+3)** |
| 0 | 10565.39 | 10470.51 | 10456.02 | 10442.87 |
| 1 | 10470.51 | 10456.02 | 10442.87 | 10485.65 |
| 2 | 10456.02 | 10442.87 | 10485.65 | 10405.7 |
| 3 | 10442.87 | 10485.65 | 10405.7 | 10540.93 |
| 4 | 10485.65 | 10405.7 | 10540.93 | **To be predicted** |
| 5 | 10405.7 | 10540.93 | 10503.76 | **To be predicted** |

The number of inputs can be varied based on the method used to predict the values.

**K Nearest Neighbor (KNN)**

The dataset has complete rows and no missing data in the inputs. So, we can use KNN algorithm directly.

**Methodology**

We iterated over the possible values of :

* Kfold validation (0 to 8 )
* Number of inputs (10 to 20 in steps of 2)
* K (40 to 60 in steps of 2)

To find the best combination of parameters to minimize the NRMSE.

**Using Euclidean distance**

Optimal Error = 0.0774328 Optimal History = 10 Optimal K = 40 Optimal\_kfold = 0

Optimal Error = 0.0772718 Optimal History = 10 Optimal K = 42 Optimal\_kfold = 0

Optimal Error = 0.0772377 Optimal History = 10 Optimal K = 44 Optimal\_kfold = 0

Optimal Error = 0.072944 Optimal History = 20 Optimal K = 40 Optimal\_kfold = 0

Optimal Error = 0.0692062 Optimal History = 10 Optimal K = 40 Optimal\_kfold = 1

Optimal Error = 0.0691492 Optimal History = 10 Optimal K = 42 Optimal\_kfold = 1

Optimal Error = 0.0688926 Optimal History = 10 Optimal K = 44 Optimal\_kfold = 1

Optimal Error = 0.068883 Optimal History = 14 Optimal K = 40 Optimal\_kfold = 4

Optimal Error = 0.0672009 Optimal History = 14 Optimal K = 40 Optimal\_kfold = 5

Optimal Error = 0.0648096 Optimal History = 20 Optimal K = 40 Optimal\_kfold = 5

**Kfold Validation**

9 fold validation has been used. Optimal\_kfold variable tells us which data chunk gave the least error.

**Result**

**Optimal Error = 0.0648096 // NRMSE error**

Optimal History = 20 // Optimal number of inputs

Optimal K = 40 // Optimal K

Optimal\_kfold = 5 // Optimal kfold partition number

**Using Manhattan distance**

Optimal Error = 0.0798606 Optimal History = 10 Optimal K = 40 Optimal\_kfold = 0

Optimal Error = 0.07986 Optimal History = 10 Optimal K = 42 Optimal\_kfold = 0

Optimal Error = 0.0796785 Optimal History = 10 Optimal K = 44 Optimal\_kfold = 0

Optimal Error = 0.0791764 Optimal History = 10 Optimal K = 50 Optimal\_kfold = 0

Optimal Error = 0.0791053 Optimal History = 10 Optimal K = 54 Optimal\_kfold = 0

Optimal Error = 0.076265 Optimal History = 20 Optimal K = 40 Optimal\_kfold = 0

Optimal Error = 0.0762472 Optimal History = 20 Optimal K = 42 Optimal\_kfold = 0

Optimal Error = 0.0759768 Optimal History = 20 Optimal K = 44 Optimal\_kfold = 0

Optimal Error = 0.0754583 Optimal History = 20 Optimal K = 46 Optimal\_kfold = 0

Optimal Error = 0.0717671 Optimal History = 10 Optimal K = 40 Optimal\_kfold = 1

Optimal Error = 0.0716917 Optimal History = 10 Optimal K = 42 Optimal\_kfold = 1

Optimal Error = 0.0716462 Optimal History = 10 Optimal K = 44 Optimal\_kfold = 1

Optimal Error = 0.071634 Optimal History = 10 Optimal K = 46 Optimal\_kfold = 1

Optimal Error = 0.0715459 Optimal History = 10 Optimal K = 48 Optimal\_kfold = 1

Optimal Error = 0.0714912 Optimal History = 10 Optimal K = 50 Optimal\_kfold = 1

Optimal Error = 0.0714206 Optimal History = 14 Optimal K = 40 Optimal\_kfold = 4

Optimal Error = 0.0690106 Optimal History = 14 Optimal K = 40 Optimal\_kfold = 5

Optimal Error = 0.0685351 Optimal History = 20 Optimal K = 40 Optimal\_kfold = 5

Optimal Error = 0.0685158 Optimal History = 20 Optimal K = 42 Optimal\_kfold = 5

**Kfold Validation**

9 fold validation has been used. Optimal\_kfold variable tells us which data chunk gave the least error.

**Result**

**Optimal Error = 0.0685158 // NRMSE error**

Optimal History = 20 // Optimal number of inputs

Optimal K = 42 // Optimal K

Optimal\_kfold = 5 // Optimal kfold partition number

**Artificial Neural Network (ANN)**

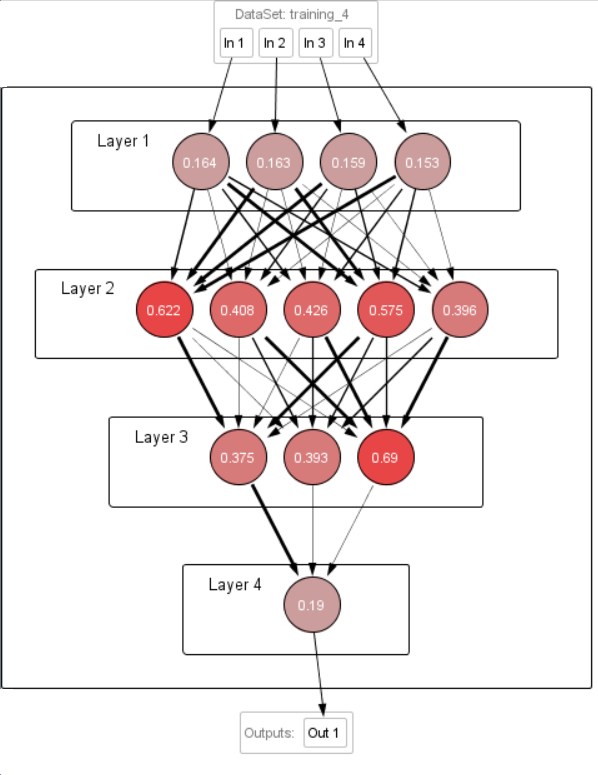
ANN can be used for predicting the dataset that we have built. We try to compare the performance of ANN with KNN.

**Methodology**

We used 4 input variables and one output.

Maximum iterations = 1000

Learning rate = 0.01



**Figure Shows the structure of the ANN used to predict**

**Result**

The ANN tool uses normalized values.

Input: 0.0026; 0.002; 0; 0.0014; Output: 0.1435; Desired output: 0.0061; Error: 0.1374;

Input: 0.002; 0; 0.0014; 0.0061; Output: 0.1438; Desired output: 0.02; Error: 0.1237;

Input: 0; 0.0014; 0.0061; 0.02; Output: 0.1452; Desired output: 0.0192; Error: 0.126;

Input: 0.0014; 0.0061; 0.02; 0.0192; Output: 0.1466; Desired output: 0.048; Error: 0.0987;

**Normalized Mean Square Error: 0.00561320442784034**

**Conclusion**

ANN gives better result than KNN. It is because of the backpropagation method of ANN to reduce the error rate.

**References:**

1. <http://research.stlouisfed.org/fred2/series/DJIA/downloaddata>
2. Jiawei Han, Micheline Kamber, and Jian Pei, “Data Mining: Concepts and Techniques”, 3rd edition
3. <http://archive.ics.uci.edu/ml/datasets/Dow+Jones+Index#>
4. <http://www.rdatamining.com/examples/time-series-clustering-classification>
5. <http://cran.r-project.org/doc/contrib/Zhao_R_and_data_mining.pdf>
6. <file:///C:/Users/megan/Downloads/RDataMining-introduction-slides.pdf>
7. http://neuroph.sourceforge.net/