

Informatics Institute of Technology

Machine Learning & Data Mining

5DATA002W

Course Work Report

Pasindu Anuradha Godakanda

Uow Number – w1871482

IIT Number – 20212026

Table of Contents

[1 Objective 1: 4](#_Toc134061606)

[2 Partitioning Clustering 4](#_Toc134061607)

[2.1 Data Pre-processing 5](#_Toc134061608)

[2.2 Determine the number of Clusters. 9](#_Toc134061609)

[2.2.1 NBclust 9](#_Toc134061610)

[2.2.2 Elbow Method 11](#_Toc134061611)

[2.2.3 Silhouette Method 12](#_Toc134061612)

[2.2.4 Gap Statistic Method 13](#_Toc134061613)

[2.3 K Means Clustering Implementation 14](#_Toc134061614)

[2.4 Silhouette Plot 16](#_Toc134061615)

[3 PCA Implementation 17](#_Toc134061616)

[3.1 Find Optimal Number of Clusters 19](#_Toc134061617)

[3.1.1 NBclust 19](#_Toc134061618)

[3.1.2 Elbow Method 21](#_Toc134061619)

[3.1.3 Silhouette Method 21](#_Toc134061620)

[3.1.4 Gap Statistics Method 22](#_Toc134061621)

[3.2 Implement K Means 23](#_Toc134061622)

[3.3 Silhouette Plot 25](#_Toc134061623)

[3.4 Calinski-Harabasz Index 25](#_Toc134061624)

[4 Objective 2: 27](#_Toc134061625)

[5 MLP Models for Energy Forecasting 27](#_Toc134061626)

[5.1 Construct the Input/output Matrices 27](#_Toc134061627)

[5.2 Normalization 28](#_Toc134061628)

[5.3 Training phase 29](#_Toc134061629)

[5.3.1 Model 1 (Matrix 1, hidden Layer = 1, hidden layer nodes = 6, linear = true) 29](#_Toc134061630)

[5.3.2 Model 2 (Matrix 1, hidden Layer = 2, hidden layer nodes = (6, 8), linear = true) 30](#_Toc134061631)

[5.3.3 Model 3 (Matrix 1, hidden Layer = 1, hidden layer nodes = 6, linear = false, act.fct = tanh) 31](#_Toc134061632)

[5.3.4 Model 4 (Matrix 1, hidden Layer = 2, hidden layer nodes = (6, 8), linear = false, act.fct = tanh) 32](#_Toc134061633)

[5.3.5 Model 5 (Matrix 2, hidden Layer = 1, hidden layer nodes = 6, linear = true) 33](#_Toc134061634)

[5.3.6 Model 6 (Matrix 2, hidden Layer = 1, hidden layer nodes = 6, linear = false, act.fct = tanh) 34](#_Toc134061635)

[5.3.7 Model 7 (Matrix 2, hidden Layer = 2, hidden layer nodes = (6,8), linear = true) 35](#_Toc134061636)

[5.3.8 Model 8 (Matrix 2, hidden Layer = 2, hidden layer nodes = (6, 8), linear = false, act.fct = tanh) 36](#_Toc134061637)

[5.3.9 Model 9 (Matrix 3, hidden Layer = 1, hidden layer nodes = 6, linear = true) 37](#_Toc134061638)

[5.3.10 Model 10 (Matrix 3, hidden Layer = 1, hidden layer nodes = 6, linear = false, act.fct = logistic) 38](#_Toc134061639)

[5.3.11 Model 11 (Matrix 3, hidden Layer = 2, hidden layer nodes = (6, 8), linear = true) 39](#_Toc134061640)

[5.3.12 Model 12 (Matrix 3, hidden Layer = 2, hidden layer nodes = (6, 8), linear = false, act.fct = logistic) 40](#_Toc134061641)

[5.3.13 Model 13 (Matrix 3, hidden Layer = 2, hidden layer nodes = (8,10), linear = true) 41](#_Toc134061642)

[5.3.14 Model 14 (Matrix 4, hidden Layer = 1, hidden layer nodes = 6, linear = false, act.fct = logistic) 42](#_Toc134061643)

[5.3.15 Model 15 (Matrix 4, hidden Layer = 2, hidden layer nodes = (4, 6), linear = false, act.fct = tanh) 44](#_Toc134061644)

[5.4 Evaluate the Performance 45](#_Toc134061645)

[5.4.1 RMSE (Root Mean Squared Error) 45](#_Toc134061646)

[5.4.2 MAE (Mean Absolute Error) 45](#_Toc134061647)

[5.4.3 MAPE (Mean Absolute Percentage Error) 45](#_Toc134061648)

[5.4.4 sMAPE (Symmetric Mean Absolute Percentage Error) 45](#_Toc134061649)

[5.5 Comparison Table 46](#_Toc134061650)

[5.6 Efficiency of the selected models 47](#_Toc134061651)

[5.6.1 One layer model (Model 14) 47](#_Toc134061652)

[5.6.2 Two layer model (Model 15) 48](#_Toc134061653)

# **Objective 1:**

## **Partitioning Clustering**

Clustering is a technique use in unsupervised machine learning which is grouping unlabeled data into smaller groups or clusters based on the similarities between data points. Partitioning clustering and Hierarchical clustering are the most popular clustering techniques in use.

For the given task we have to use the partitioning clustering where we need to decide the number of clustering beforehand and algorithm start by randomly assign data points for the centers. K means is the most common partitioning clustering algorithm which is clustering using the distance between the data points.

### **Data Pre-processing**

Before we start clustering we need to preprocess the given dataset which is a very important step in data analysis and machine learning. Data preprocessing is a process of cleaning raw data to improve the quality of data and enhance the accuracy and the efficiency of the machine learning model. Data preprocessing included deal with missing data, handle outliers and normalization.

Any of given raw dataset is often incomplete, noisy, and could have missing values and outliers. This kind of errors can be affect in the outcome of the machine learning model. And also a given data set could have data from different ranges, different formats, or scales. Normalization convert data into consistent format and scale make it easier to analyze and compare data.

As for the given task, first we need to import the dataset to our workplace.

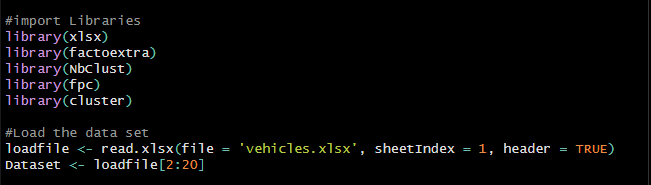


Figure 1 import dataset

The dataset that given to us in the “.xlsx” format which means it is a excel file. There are many ways to read excel file in R language and I used external package called “xlsx” for this purpose. As show in the *Figure 1.1*  I imported the “vehicles.xlsx” file as a data frame. Then we need to remove first column, which contains the number of the sample since we don’t need that column for our task.

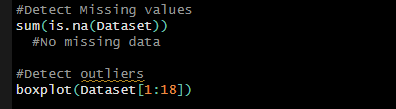


Figure 2 Detecting missing values and outliers

Then I checked for any missing data and no missing data found. Now we need to handle the outliers and for that first, we should detect outliers. In R Language “boxplot” is the easiest way to detect outliers. I used boxplot function to visualize and detect outliers in the dataset.

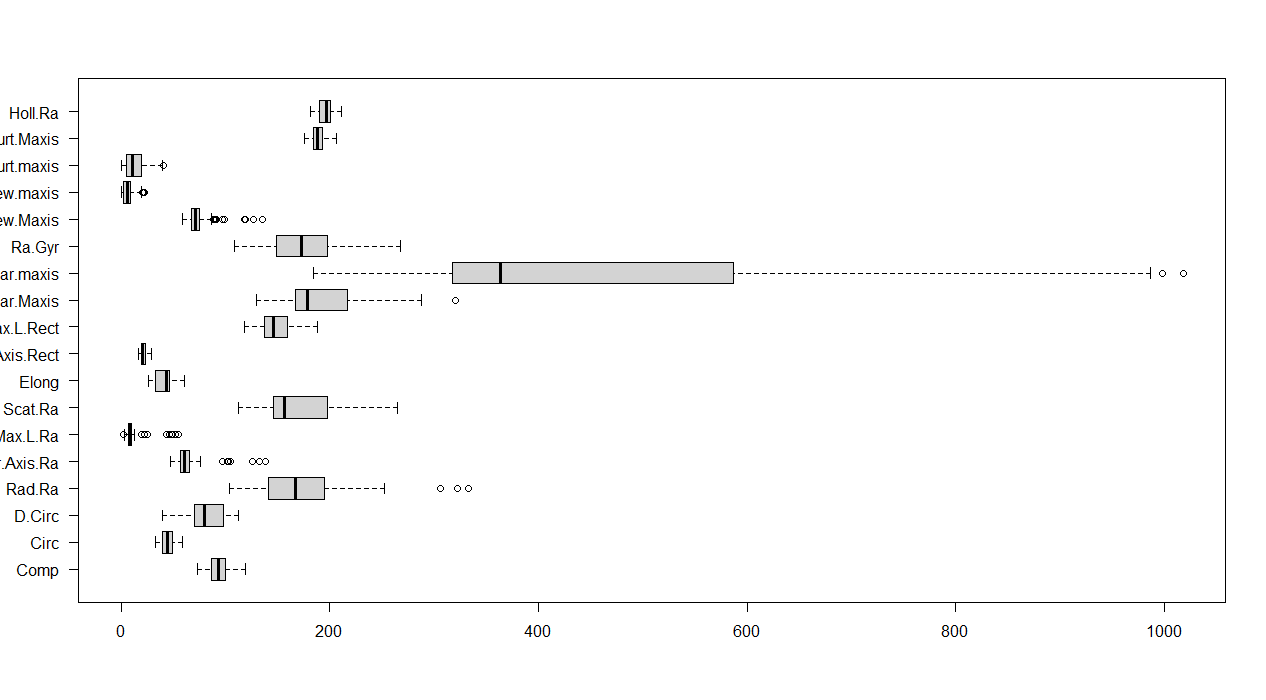


Figure 3 boxplot for detecting outliers

As shown in this boxplot, Skew.maxis, Pr.Axis.Ra, Sc.Var.Maxis, Rad.Ra, Kurt.maxis, Skew.maxis, Max.L.Ra, Sc.Var.maxis variables contains outliers. For Remove outliers I used following R code snippet.

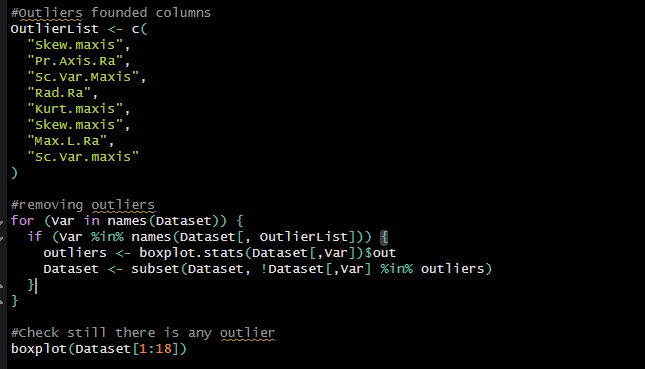


Figure 4

After removing outliers there were still remaining outliers in the Skew.Maxis and Sc.Var.maxis columns. Hence following the steps that we do before I cleared every outliers.

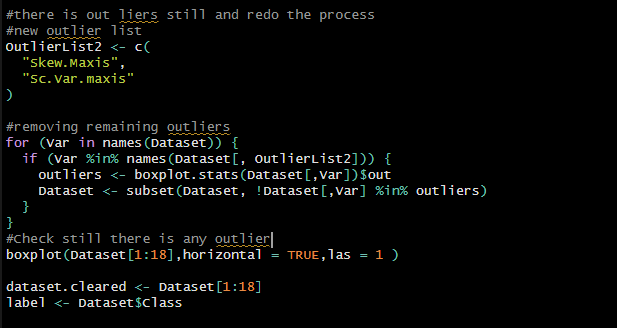


Figure 5

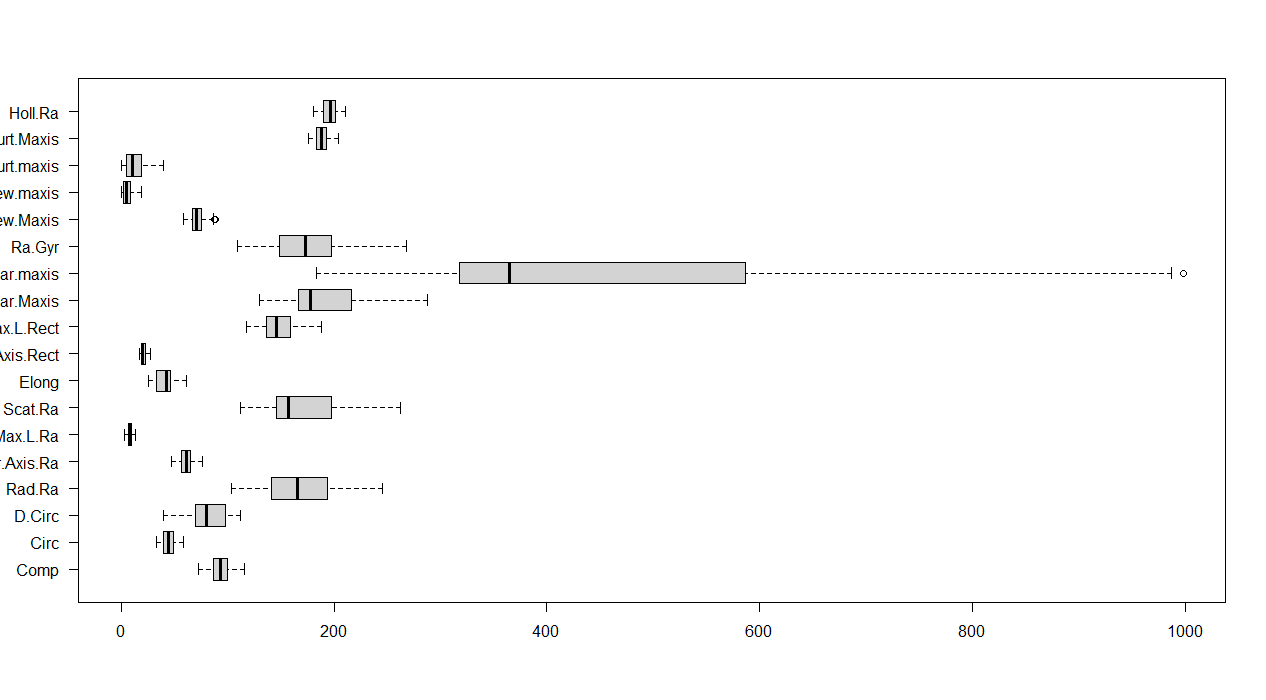


Figure 6

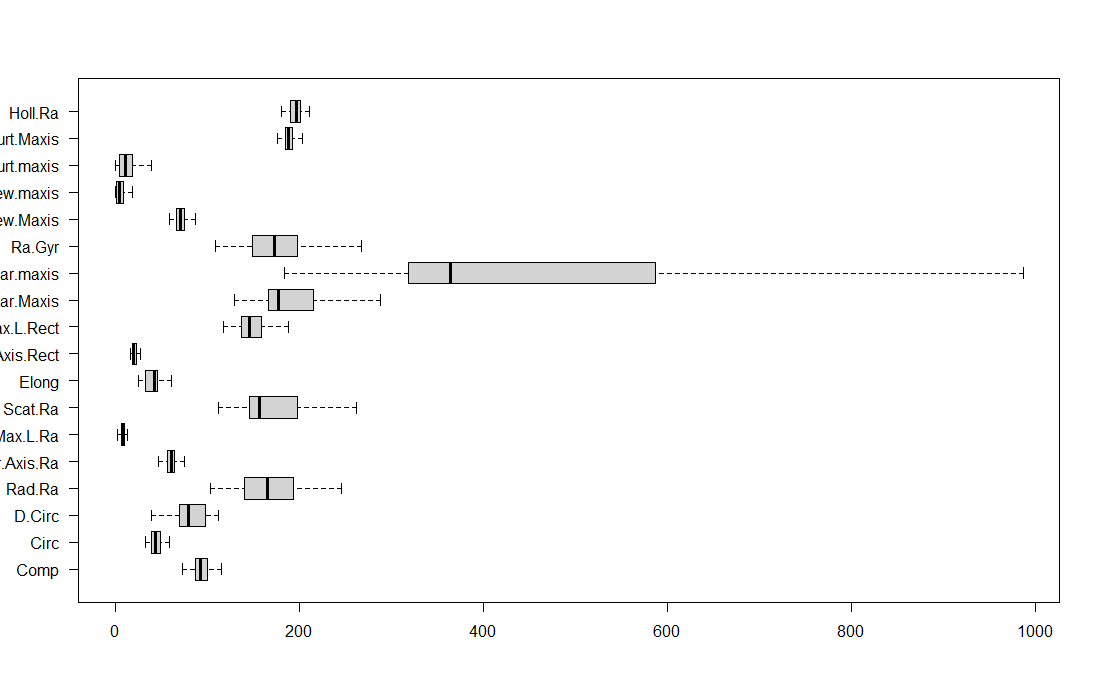


Figure 7 After removing all outliers

As shown in the boxplot data in each column scatter through different data ranges. We need to bring them in to a same range or we need to scale them. Following boxplot visualize how data scatter after scaling.

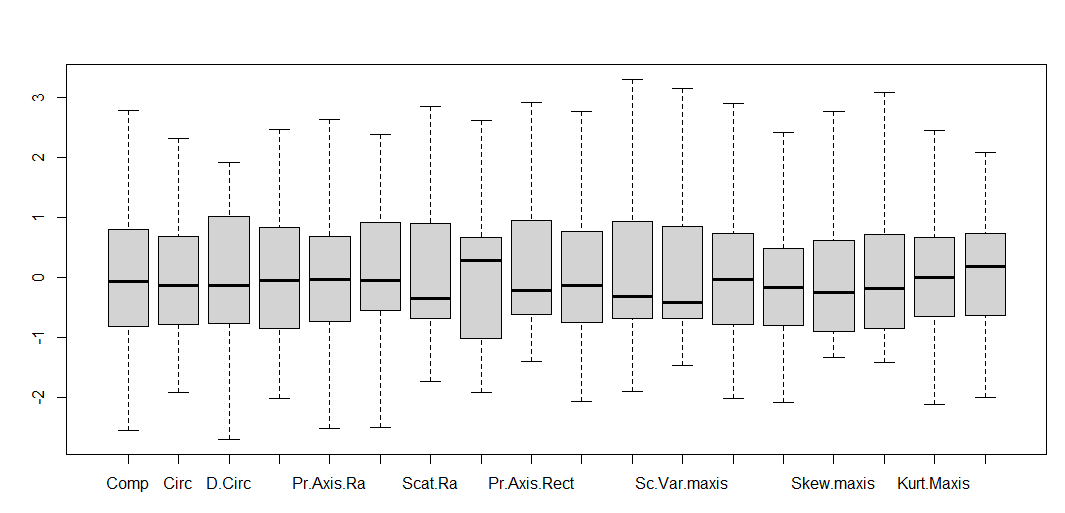


Figure 8 After scaling

### **Determine the number of Clusters.**

After completing the data preprocessing stage, we can perform the k means clustering. For perform k means clustering we need to determine ‘k’ or the number of clusters. We can use few automated tools for determine the number of clusters. In this task I used the following automated tools to determine the k.

* NBclust
* Elbow method
* Silhouette method
* Gap statistics method

#### **NBclust**

I have used NBclust method with Euclidean distance and Manhattan distance. As shown in the following figures, according to NBclust method, Optimal number of clusters is 3 when using Euclidean distance. When Using Manhattan distance NBclust method suggest optimal number of cluster is 2.

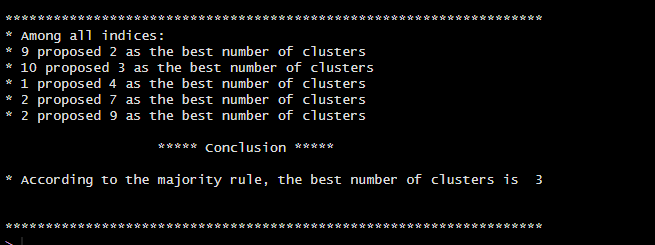


Figure 9 Euclidean distance

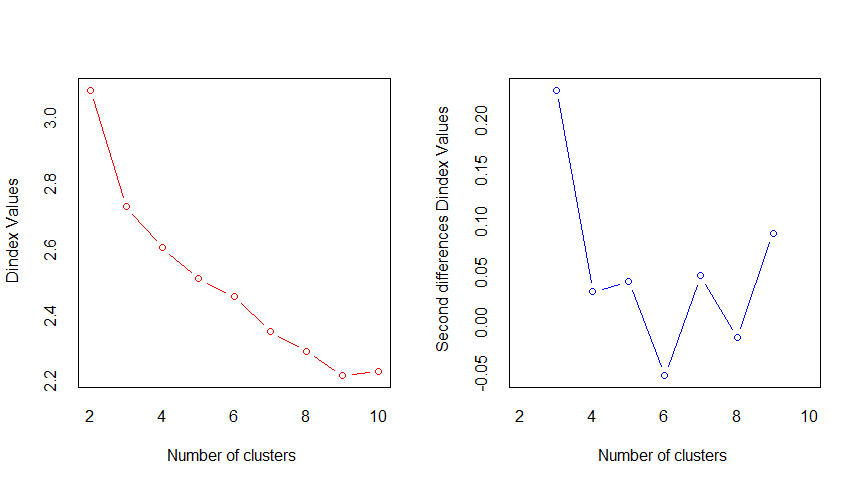


Figure 10 Euclidean Distance

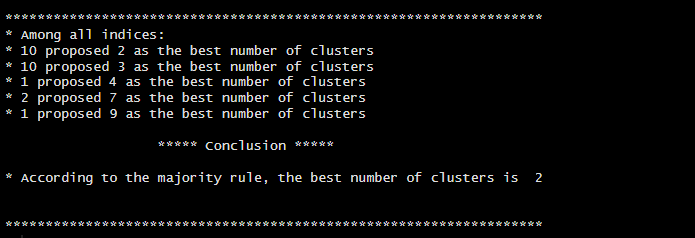


Figure 11 Manhattan Distance

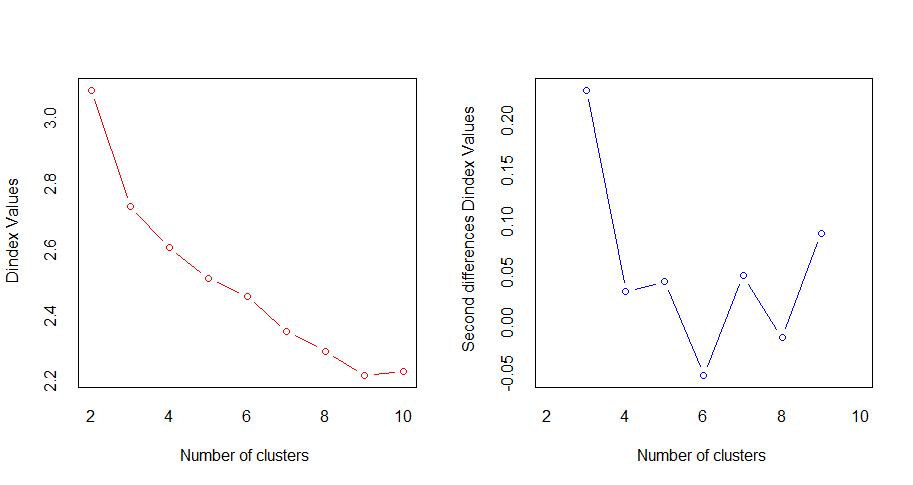


Figure 12 Manhattan Distance

#### **Elbow Method**

Elbow method is a Graphical approach to determine optimal number of clusters. It involves plotting sum of squared distance of data points from their centroids based on number of clusters. We can find optimal number of clusters using the bending point of the plot. As the Following diagram suggest, the optimal number of clusters is 3 according to the elbow method.

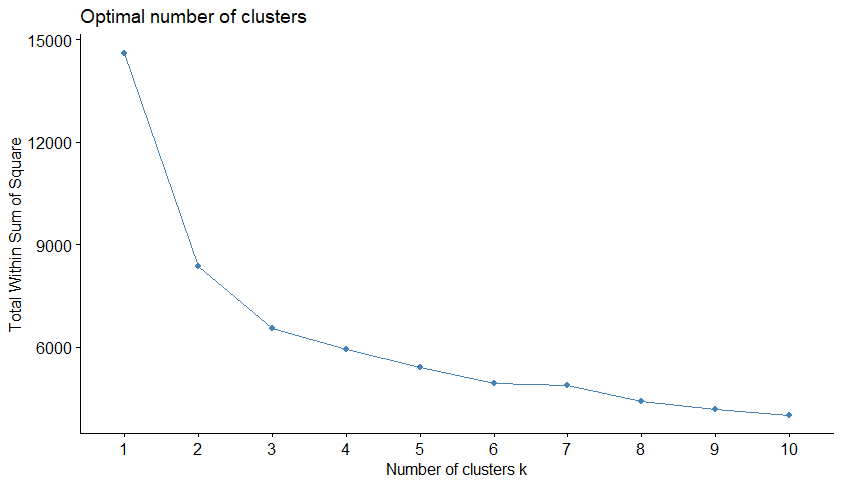


Figure 13 Elbow Method

#### **Silhouette Method**

According to the Silhouette method the optimal number of clusters is 2 as shown in the following graph.

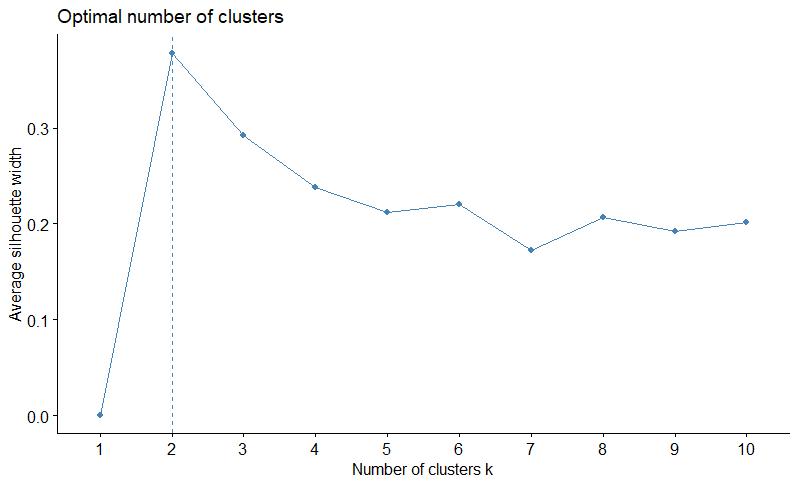


Figure 14 Silhouette Method

#### **Gap Statistic Method**

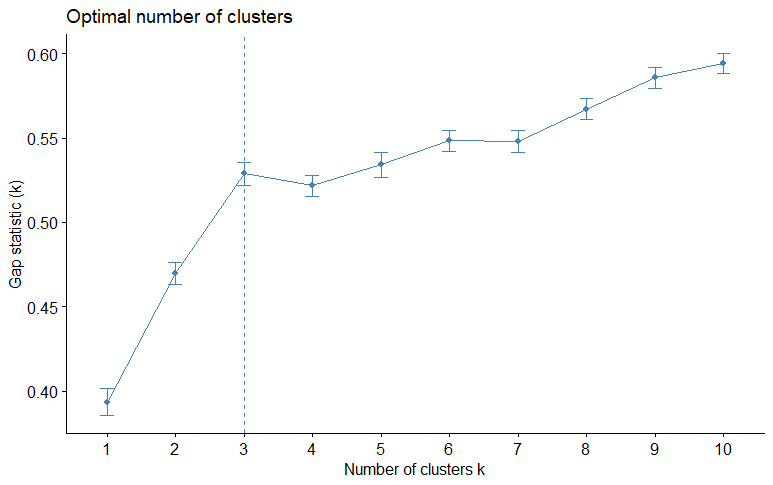


Figure 15 Gap Statistic Method

Above Diagram suggest according to the gap statistic method optimal number of clusters is 3.

|  |  |
| --- | --- |
| **Method** | **Number Of Clusters** |
| NBclust with Euclidean Distance | 3 |
| NBclust with Manhattan Distance | 2 |
| Elbow Method | 3 |
| Silhouette Method | 2 |
| Gap Statistic Method | 3 |

When we compare the results, majority of the automated methods suggest the optimal number of clusters is 3.

### **K Means Clustering Implementation**

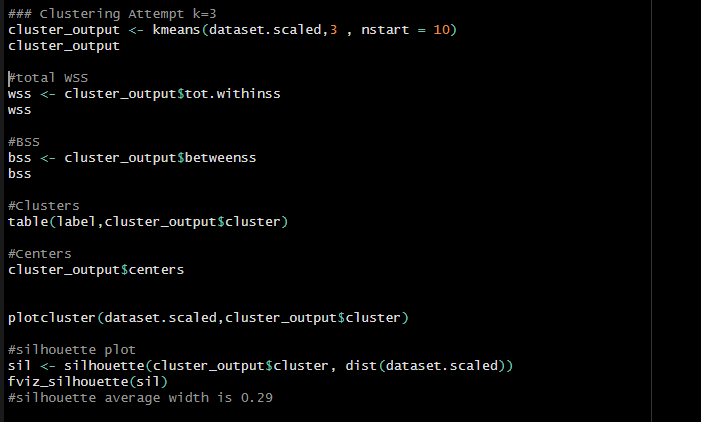
As we found the most favorable optimal number of clusters in the previous step, now we can perform the k means clustering for k = 3.

Figure 16

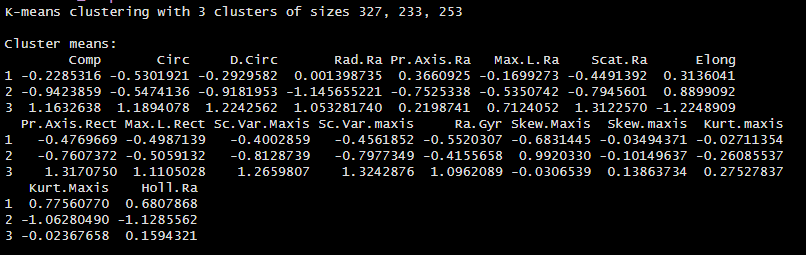


Figure 17

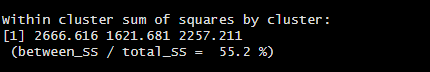


Figure 18

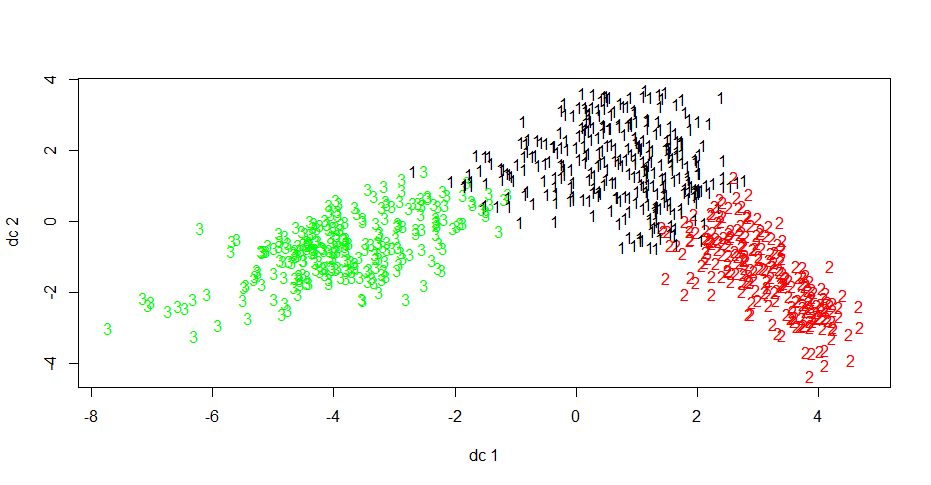


Figure 19

**Cluster Results**

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **Cluster 1** | **Cluster 2** | **Cluster 3** |
| Bus | 82 | 80 | 46 |
| Van | 109 | 80 | 0 |
| Opel | 63 | 35 | 110 |
| Saab | 73 | 38 | 97 |
| **Total** | **327** | **233** | **253** |

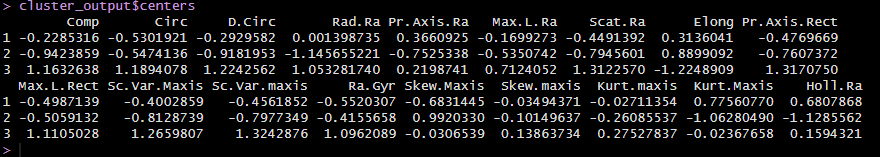


Figure 20 Cluster Centers

**Summary**

|  |  |
| --- | --- |
| Within Cluster Sum of Squares (WSS) | 6545.508 |
| Between Cluster Sum of Squares (BSS) | 8070.492 |
| Total Sum of Squares (TSS) | 14616 |
| Ratio BSS over TSS | 55.22% |

### **Silhouette Plot**

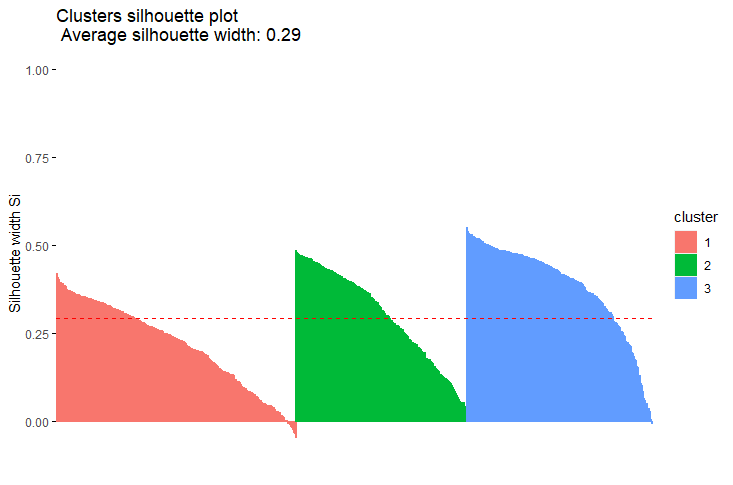


Figure 21 Silhouette Plot

|  |  |  |
| --- | --- | --- |
| **Cluster** | **Size** | **Average silhouette width** |
| 1 | 327 | 0.22 |
| 2 | 233 | 0.30 |
| 3 | 253 | 0.38 |

As shown in the Silhouette plot, the average silhouette score is 0.29, which is not a bad silhouette score considering the best clustering silhouette score in generally above 0.5 and we consider below the 0.2 silhouette score as a poor clustering. However, the interpretation of the silhouette score can vary depending on the dataset and the clustering method used. In some cases lower silhouette scores may still indicate reasonable clustering results if the dataset is difficult to cluster or the pattern of the dataset make clustering challenging. As the all data points are above the average silhouette score, we can consider this as a good sign. Also the thickness of the plots are more or less similar in size which indicate the sizes of clusters are also more or less similar to each other. Considering all of these observations, we can consider this as a good clustering attempt.

## **PCA Implementation**

Principal Component Analysis (PCA) is a dimensionality reduction technique which reduce the original number of variables while keeping as much as the information of the original variables. To perform PCA first we need to scale the dataset. Hence I have used previously scaled dataset as the input.

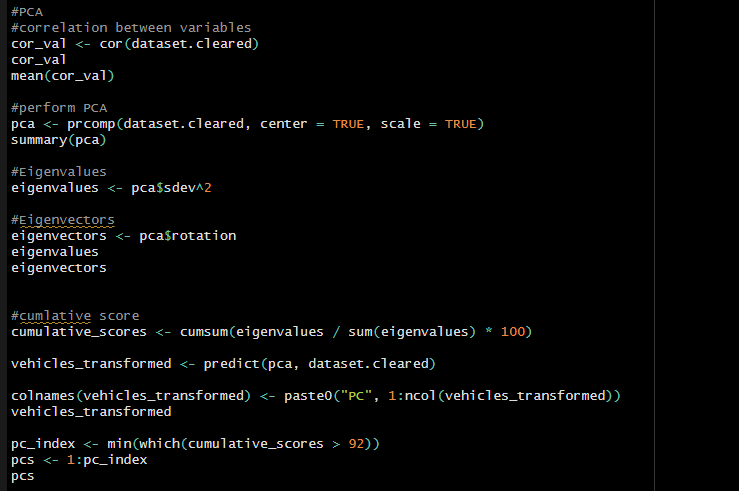


Figure 22

After perform PCA to the scaled dataset using “prcomp” function we can see the details of the PCs that have created using dataset using “summary” function as following *figure 23.*

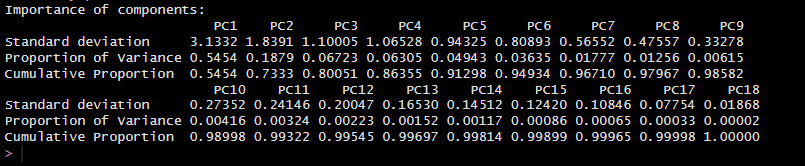


Figure 23 Details of PCs

Additionally we can extract eigenvalues and eigenvectors from the PCA results.

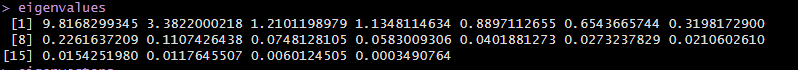


Figure 24 Eigenvalues

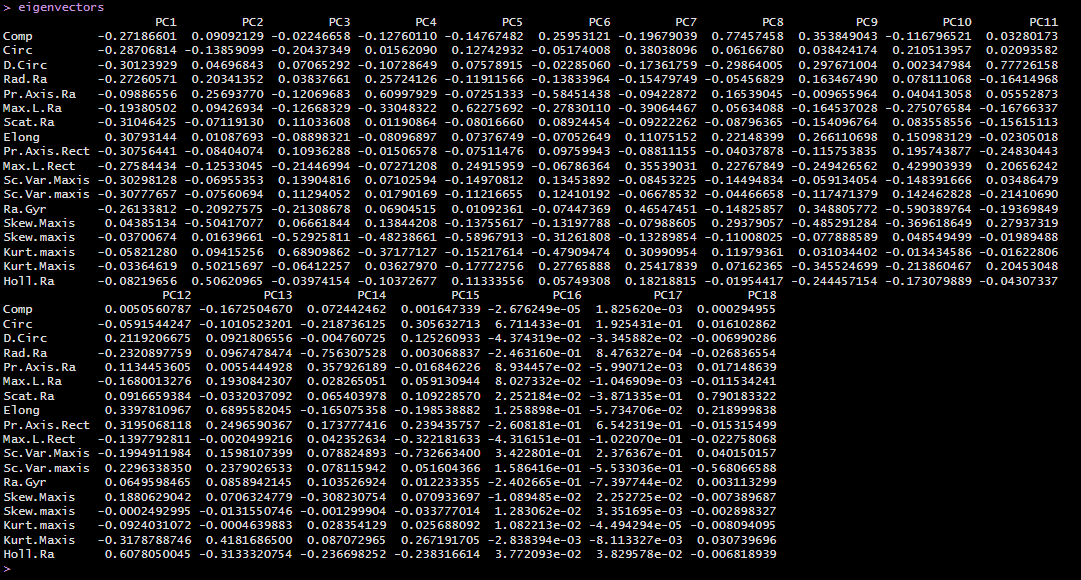


Figure 25 Eigenvectors

Next we need to choose PCs that explains the maximum amount of variation in the dataset. To do that we need to cumulative scores which represent the cumulative amount of variations in the data which explain by certain number of PCs. Then as the coursework specification suggest we need to find the smallest number of PCs as over at least 92% of cumulative score. Using above code snippet we can find the PCs that fit to our requirements which is first 6 PCs. Then we need to create new transformed dataset from using those PCs.

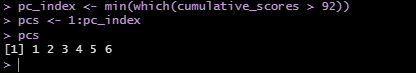


Figure 26 Number of PCs

### **Find Optimal Number of Clusters**

As we have a new dataset, we need to find the optimal number of clusters before applying the K Means to the new PCA dataset. For that I used previous 4 automated tools again.

#### **NBclust**

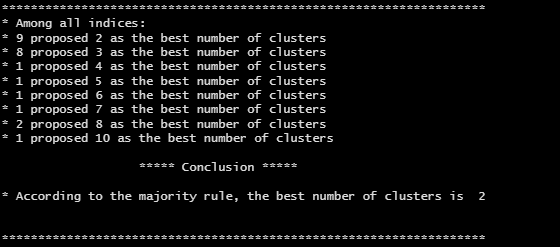


Figure 27 Using Euclidean Distance

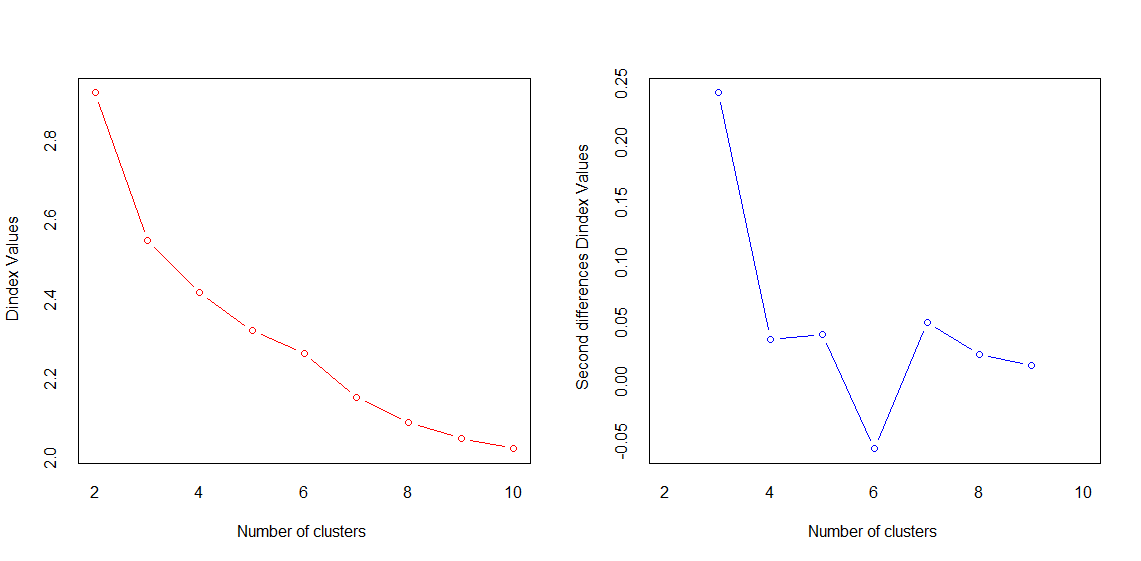


Figure 28 Using Euclidean distance

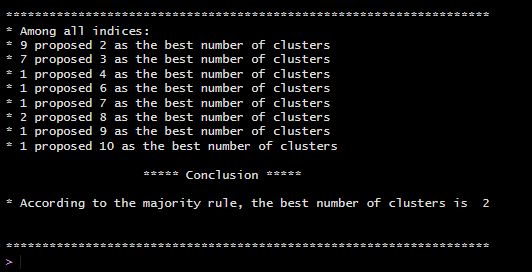


Figure 29 Using Manhattan Distance

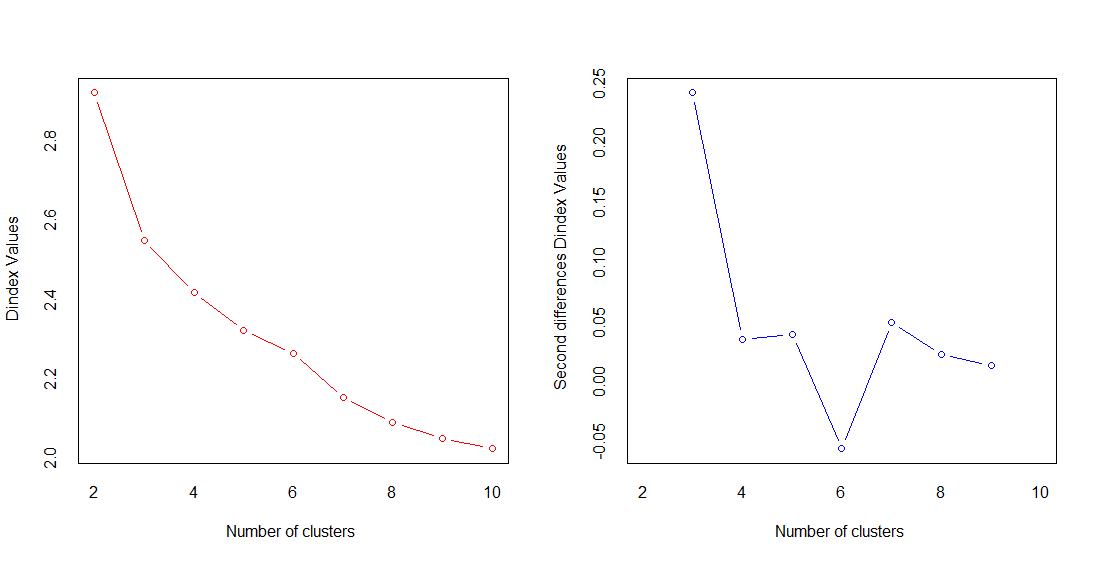


Figure 30 Using Manhattan Distance

According to the NBclust method optimal number of clusters are 2 in both Manhattan distance and Euclidean distance options.

#### **Elbow Method**

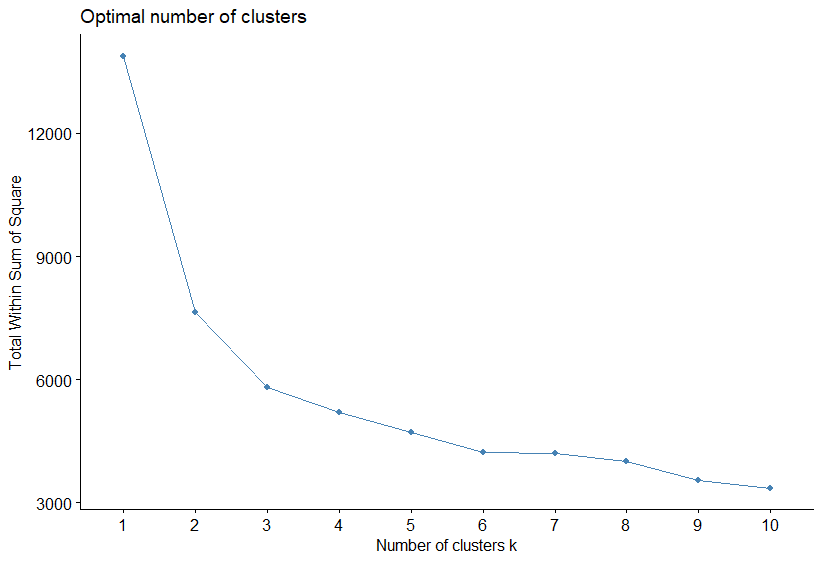


Figure 31 Elbow Method

According to the above diagram optimal number of clusters is 3 as Elbow method suggest.

#### **Silhouette Method**

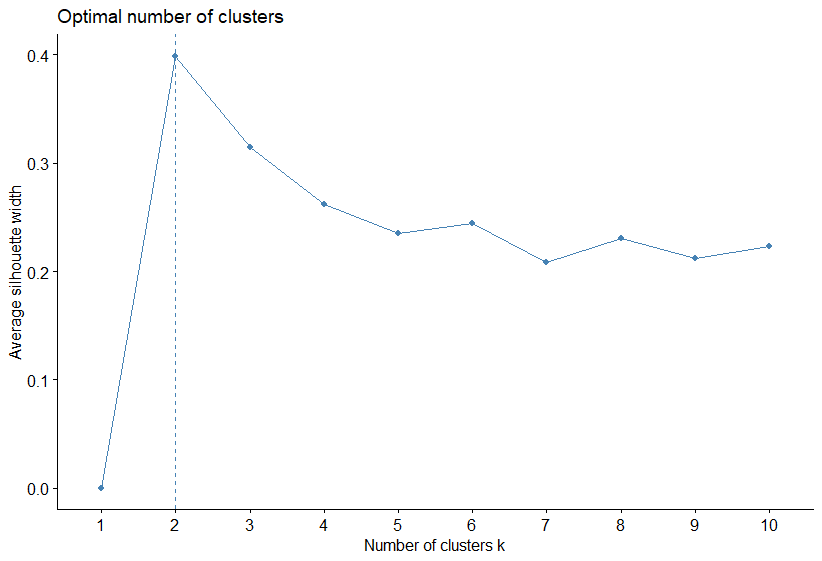


Figure 32 Silhouette Method

As Silhouette method suggest the optimal number of cluster is 2 according to the above diagram.

#### **Gap Statistics Method**

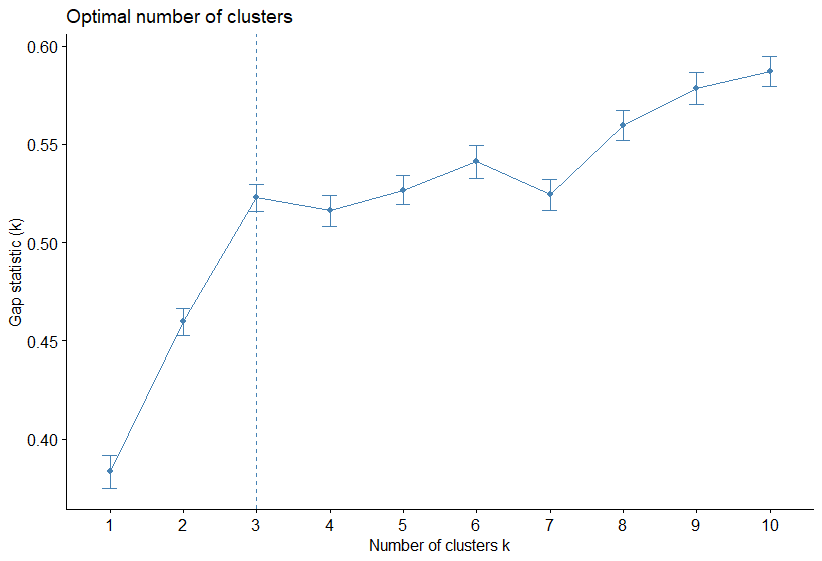


Figure 33 Gap Statistics Method

Gap statistics Method Suggest that the best number of clusters is 3.

|  |  |
| --- | --- |
| **Method** | **Number Of Clusters** |
| NBclust with Euclidean Distance | 2 |
| NBclust with Manhattan Distance | 2 |
| Elbow Method | 3 |
| Silhouette Method | 2 |
| Gap Statistic Method | 3 |

When we compare the results, majority of the automated methods suggest the optimal number of clusters is 2. It seems that when using new dataset number of clusters reduced to 2 from 3. This can be result of dimensionality reduction and it may be have reduce the complexity of the dataset.

### **Implement K Means**

After Determine the Optimal number of clusters we can perform K means clustering to our new dataset.

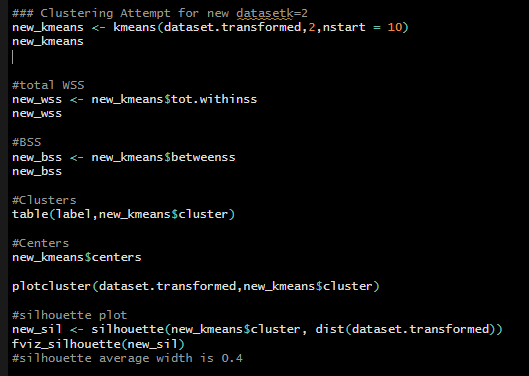


Figure 34

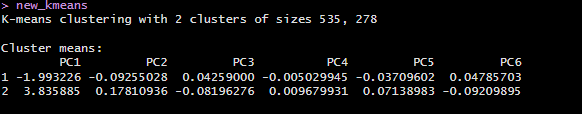


Figure 35

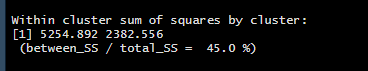


Figure 36

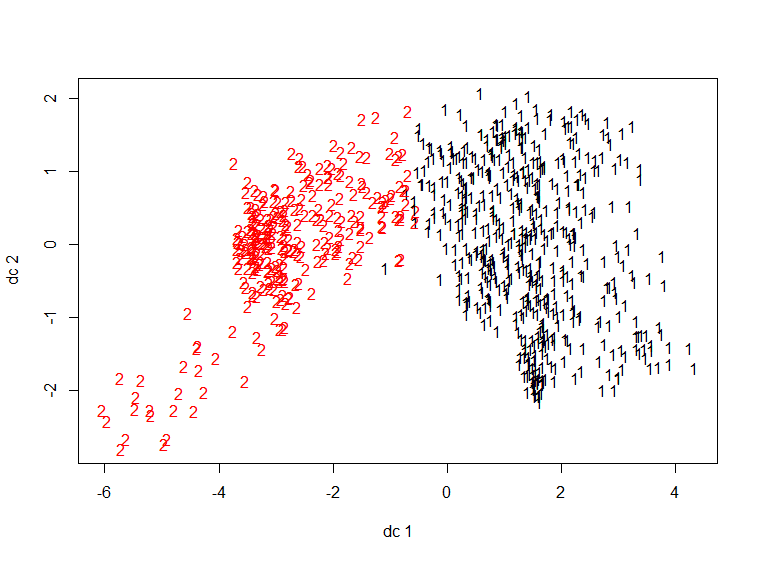


Figure 37

**Cluster Results**

|  |  |  |
| --- | --- | --- |
| **Class** | **Cluster 1** | **Cluster 2** |
| Bus | 155 | 53 |
| Van | 189 | 0 |
| Opel | 92 | 116 |
| Saab | 99 | 109 |
| **Total** | **535** | **278** |



Figure 38 Cluster Centers

**Summary**

|  |  |
| --- | --- |
| Within Cluster Sum of Squares (WSS) | 7637.447 |
| Between Cluster Sum of Squares (BSS) | 6238.04 |
| Total Sum of Squares (TSS) | 13875.49 |
| Ratio BSS over TSS | 45% |

### **Silhouette Plot**

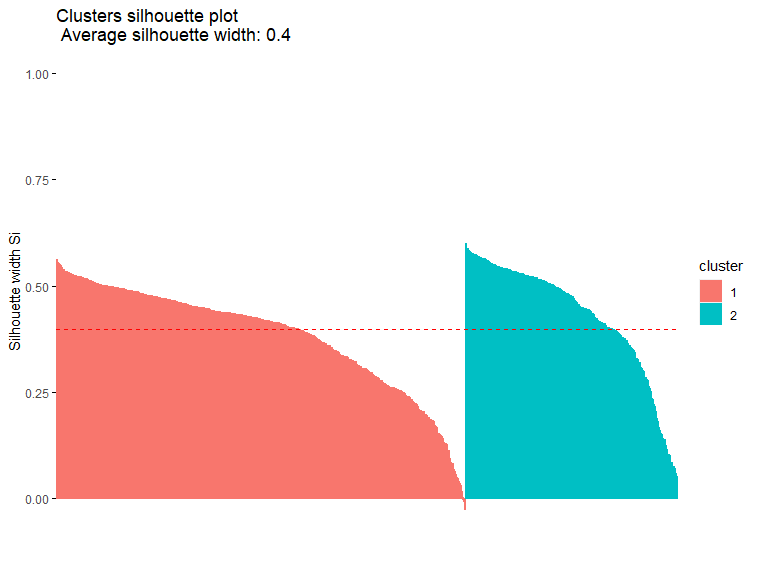


Figure 39 Silhouette Plot

|  |  |  |
| --- | --- | --- |
| **Cluster** | **Size** | **Average silhouette width** |
| 1 | 535 | 0.38 |
| 2 | 278 | 0.43 |

According to the above silhouette plot we can see the average silhouette width is this time higher than the previous one. And also we can see the all data points are well above the average silhouette width and it makes this good attempt of clustering. Even though, the thicknesses of the plots having difference which indicate the different size of clusters we can consider this as a good attempt of clustering.

### **Calinski-Harabasz Index**

The Calinsk-Harabasz index is a measure of clustering quality that can be used to determine the optimal number of clusters. It based on the ratio of the between cluster variance to the within cluster variance and also it measures the extent to which the clusters are well separated from each other.

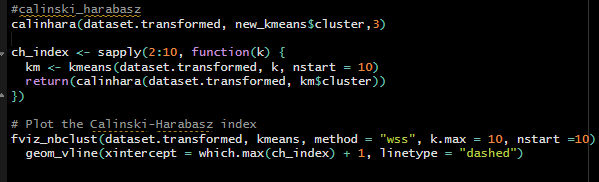


Figure 40

Calinski-harabasz index of the new transformed data set which used in PCA part is 662.4007, which is very higher value when compare in the following diagram and it suggest that this is a good clustering attempt.

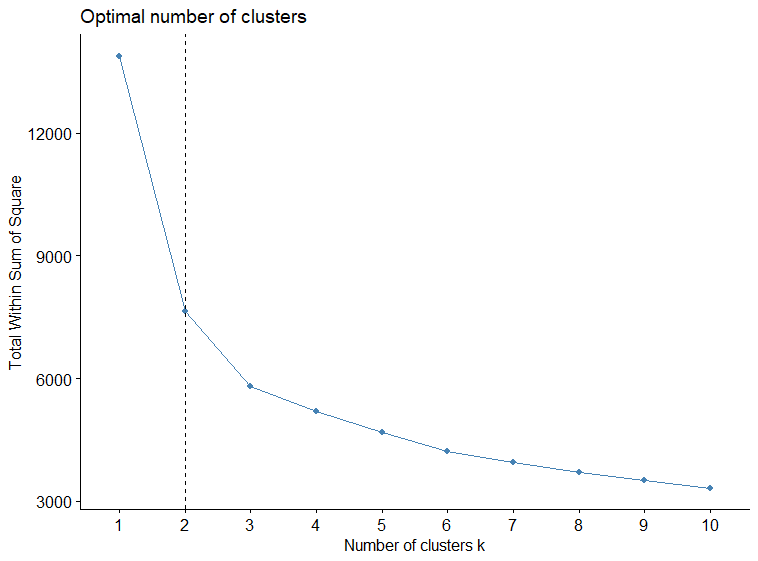


Figure 41 Calinski Harabasz Index plot

# **Objective 2:**

## **MLP Models for Energy Forecasting**

MLP (Multi – Layer Perceptron) models are commonly used for electricity load forecasting due to their ability to capture complex relationships between the input and the target variables. In the context of electricity forecasting, MLP models can be trained using historical load demand data and a set of input variables to predict the future load demand.

The input vector typically consist of a set of features or input variables that used to predict the future electricity load demand. There are several schemes and methods that are commonly used to define the input vector in electricity forecasting such as Autoregressive approach which uses lagged values of the electricity load demand as input variables, Seasonal Decomposition which uses decomposed seasonal and non-seasonal components from electricity load demand, Weather Variables which involves weather related variables base on the assumption of the weather conditions can have a clear impact on the electricity load demand, Calendar Variables which involves calendar related variables based on the assumption of the they may have different electricity load demand and Exogenous Variables which are additional exogenous variable that may be relevant to the electricity load demand.

As for the Coursework specification in the 1st sub task we are going to use Autoregressive (AR) approach.

### **Construct the Input/output Matrices**

As the order of the AR approach is unknown, we need to experiment with various input vectors and create an Input/output matrix for each of them. Hence, I used the following code snippet to construct I/O matrices.

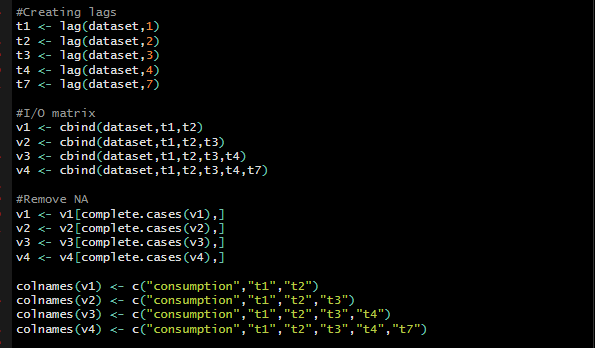


Figure 42

As suggest in the subtask 1, we only have to use the 20th hour attribute for the forecasting process. Which means we need to predict 20th hour attribute using 20th hour attribute of previous days. First I separated 20th hour attribute column and lagged it into (t-1), (t-2), (t-3), (t-4) and (t-5) inputs. Then I created 4 different data frames using those inputs and 20th hour attribute as the output. Then I completed my 4 I/O matrices from removing NA values. My four I/O matrices are,

* Matrix 1 (20th hour attribute, (t-1), (t-2) )
* Matrix 2 (20th hour attribute, (t-1), (t-2), (t-3) )
* Matrix 3 (20th hour attribute, (t-1), (t-2) (t-3), (t-4) )
* Matrix 4 (20th hour attribute, (t-1), (t-2) (t-3), (t-4), (t-7) )

### **Normalization**

Normalization is an important preprocessing step in any kind of machine learning task including MLP neural networks. Normalization is a process of adjusting the futures of the dataset into common range. Normalization ensures that the all input variables contribute equally to the model without biased towards variables with larger magnitudes or ranges. Normalization also can be helped in accelerating the training of MLP models by reducing the number of iterations required to reach good solution because normalization make it easy for the optimization algorithm to find the optimal weights.

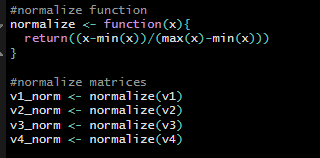


Figure 43

In the end of the normalization we need to split normalized matrices into training and testing datasets.

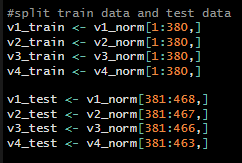


Figure 44

### **Training phase**

#### **Model 1 (Matrix 1, hidden Layer = 1, hidden layer nodes = 6, linear = true)**

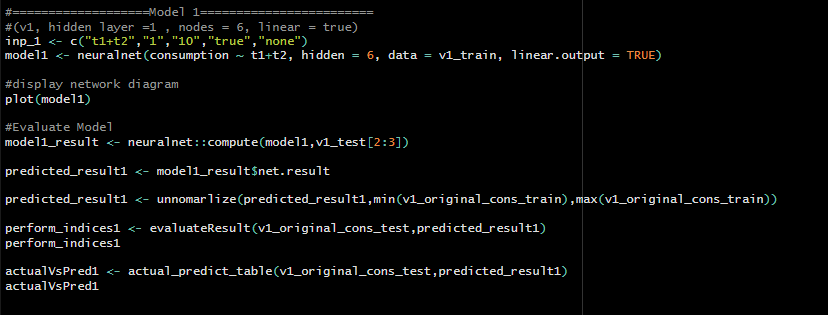


Figure 45

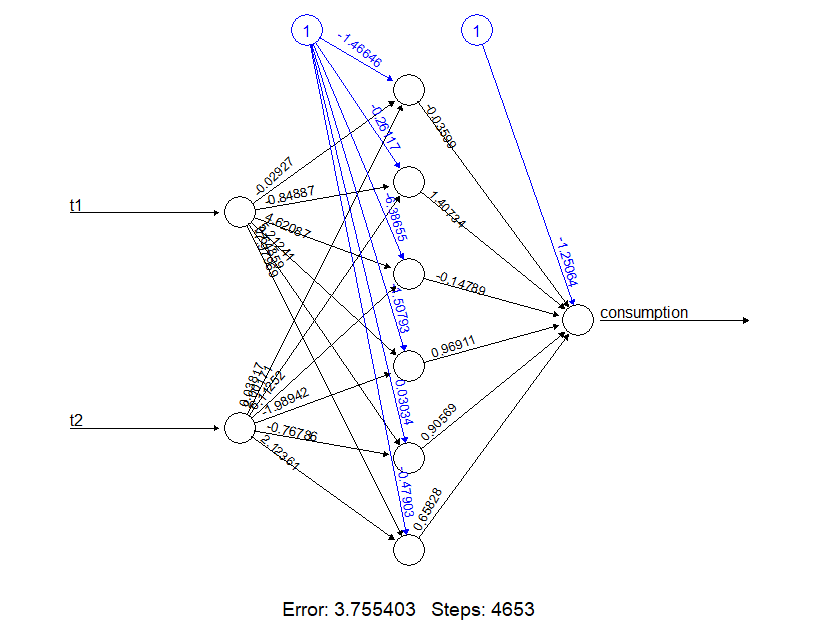


Figure 46

#### **Model 2 (Matrix 1, hidden Layer = 2, hidden layer nodes = (6, 8), linear = true)**

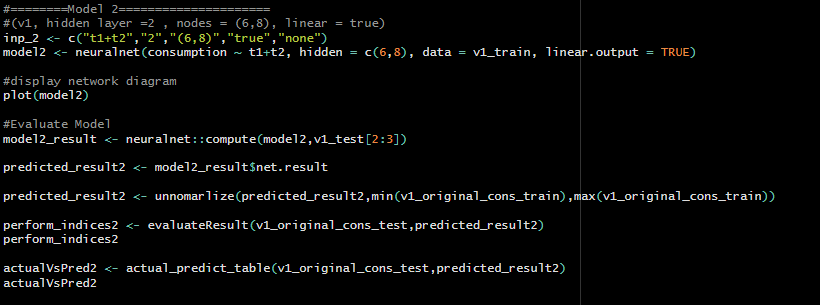


Figure 47

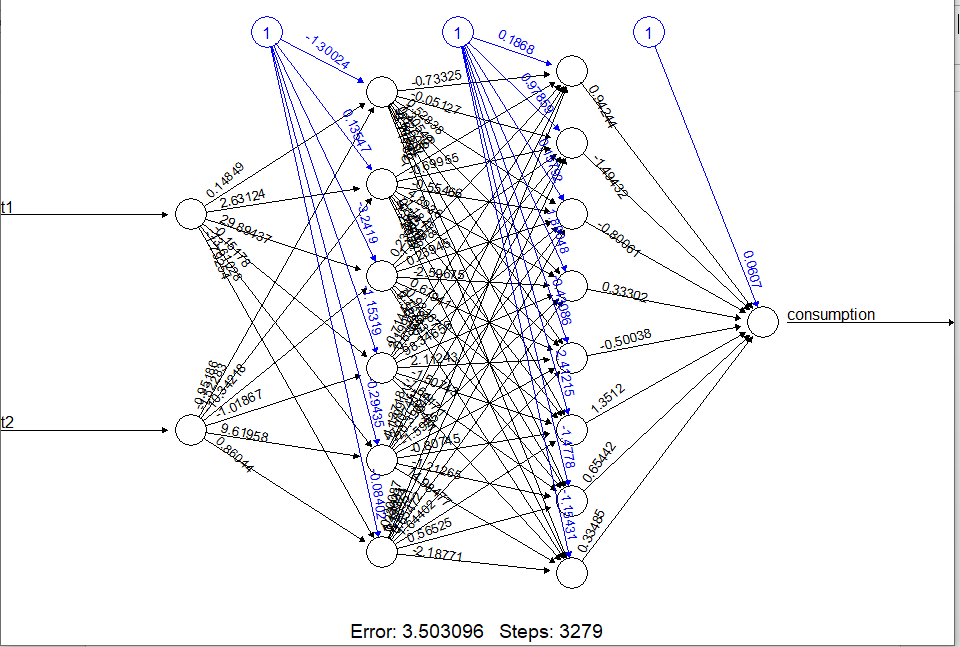


Figure 48

#### **Model 3 (Matrix 1, hidden Layer = 1, hidden layer nodes = 6, linear = false, act.fct = tanh)**

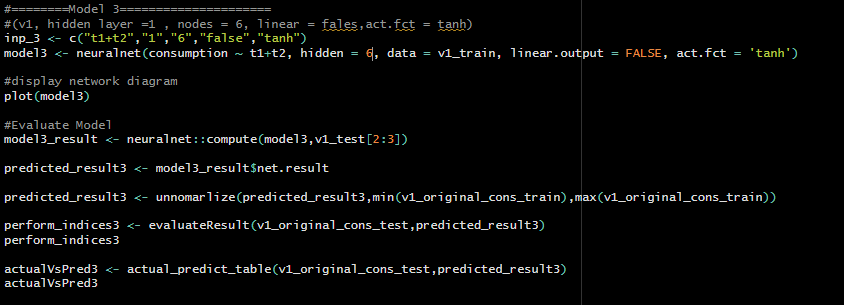


Figure 49

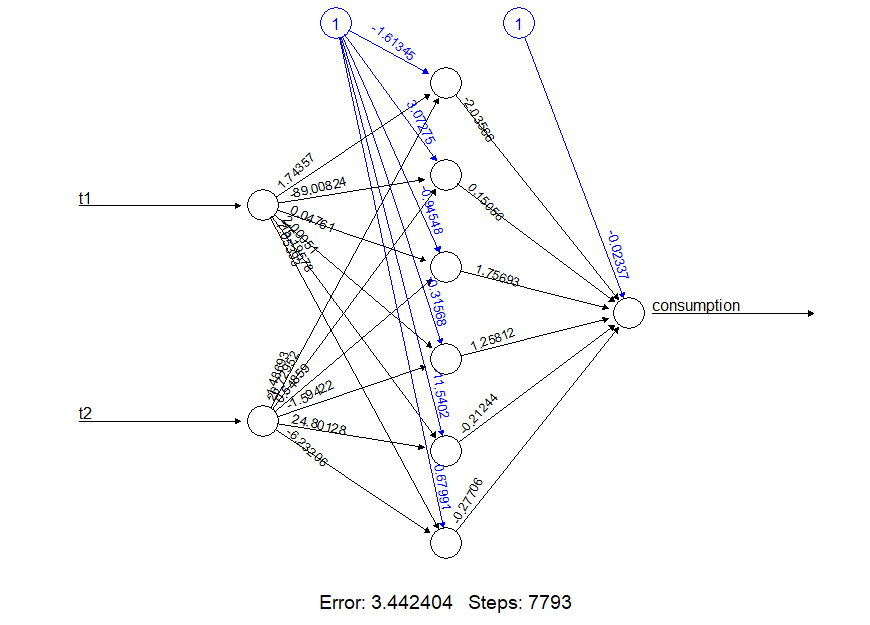


Figure 50

#### **Model 4 (Matrix 1, hidden Layer = 2, hidden layer nodes = (6, 8), linear = false, act.fct = tanh)**

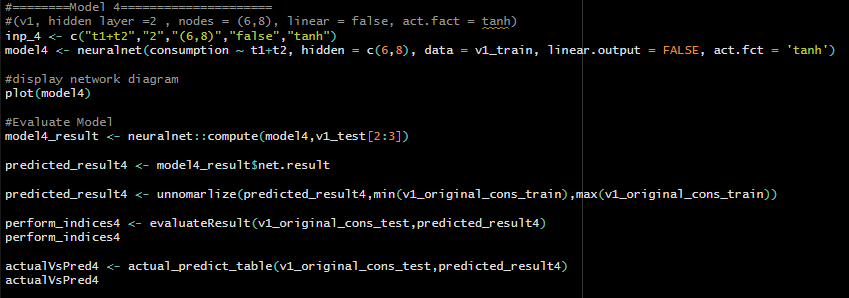


Figure 51

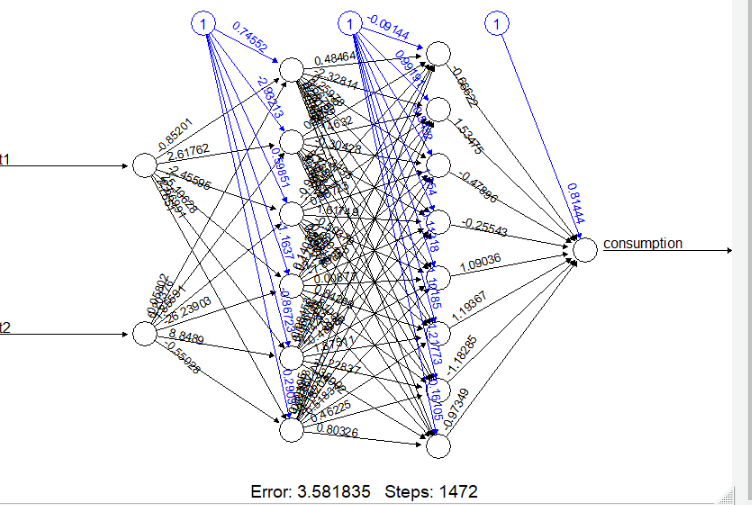


Figure 52

#### **Model 5 (Matrix 2, hidden Layer = 1, hidden layer nodes = 6, linear = true)**

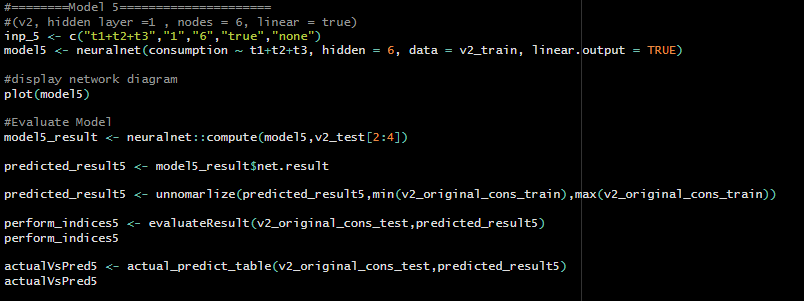


Figure 53

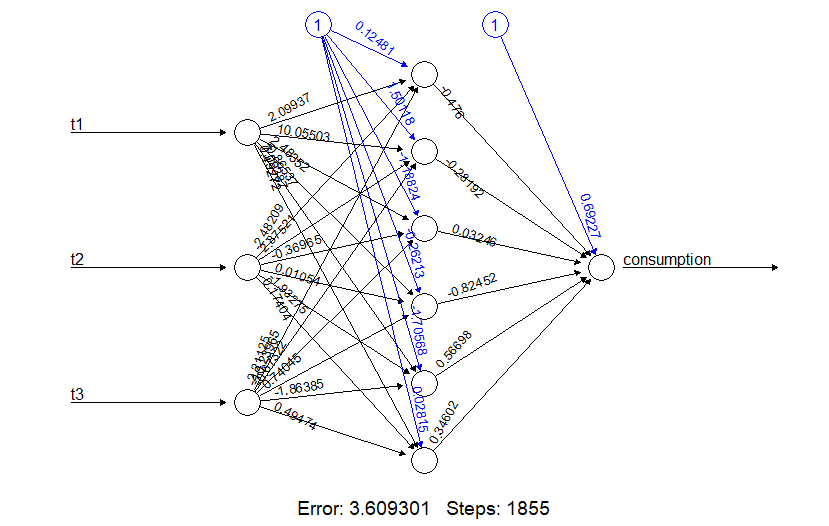


Figure 54

#### **Model 6 (Matrix 2, hidden Layer = 1, hidden layer nodes = 6, linear = false, act.fct = tanh)**

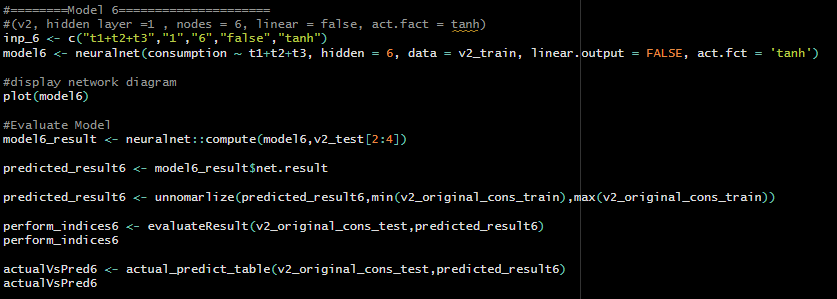


Figure 55

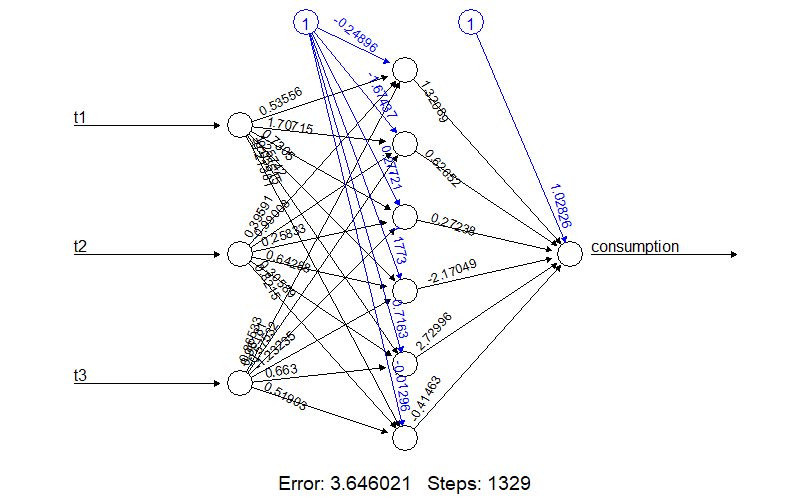


Figure 56

#### **Model 7 (Matrix 2, hidden Layer = 2, hidden layer nodes = (6,8), linear = true)**

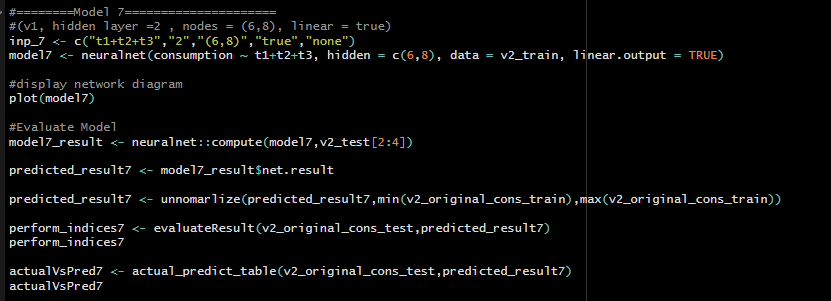


Figure 57

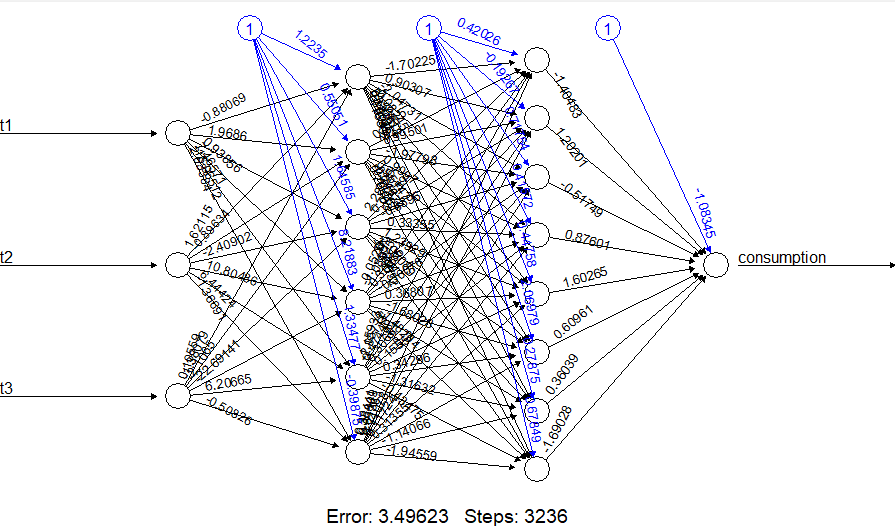


Figure 58

#### **Model 8 (Matrix 2, hidden Layer = 2, hidden layer nodes = (6, 8), linear = false, act.fct = tanh)**

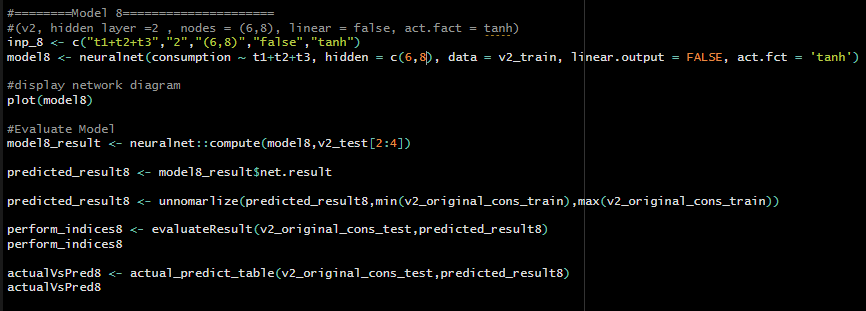


Figure 59

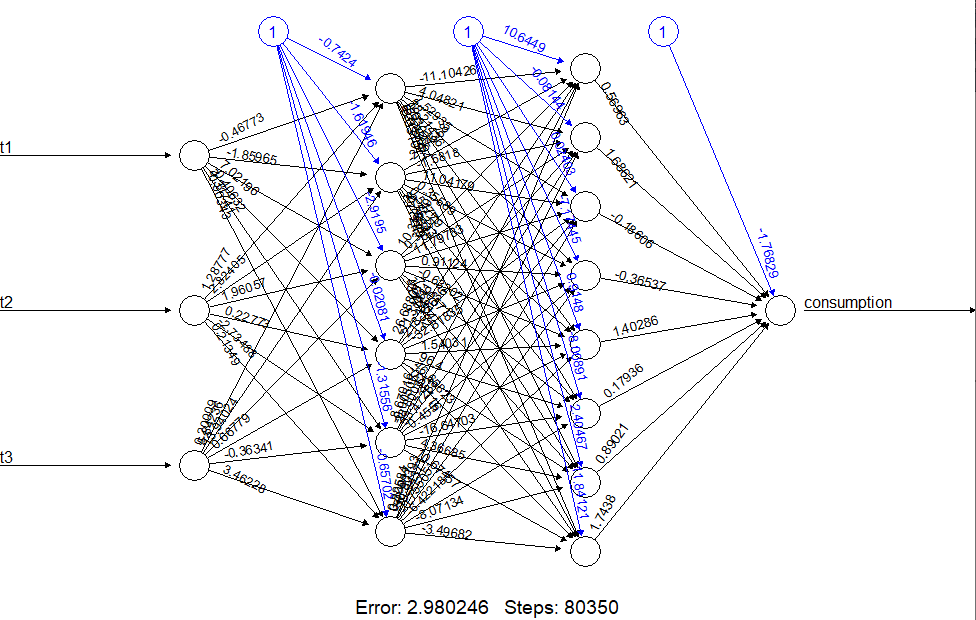


Figure 60

#### **Model 9 (Matrix 3, hidden Layer = 1, hidden layer nodes = 6, linear = true)**

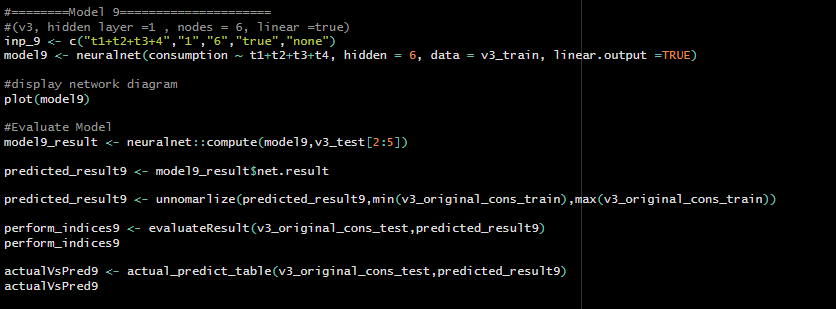


Figure 61

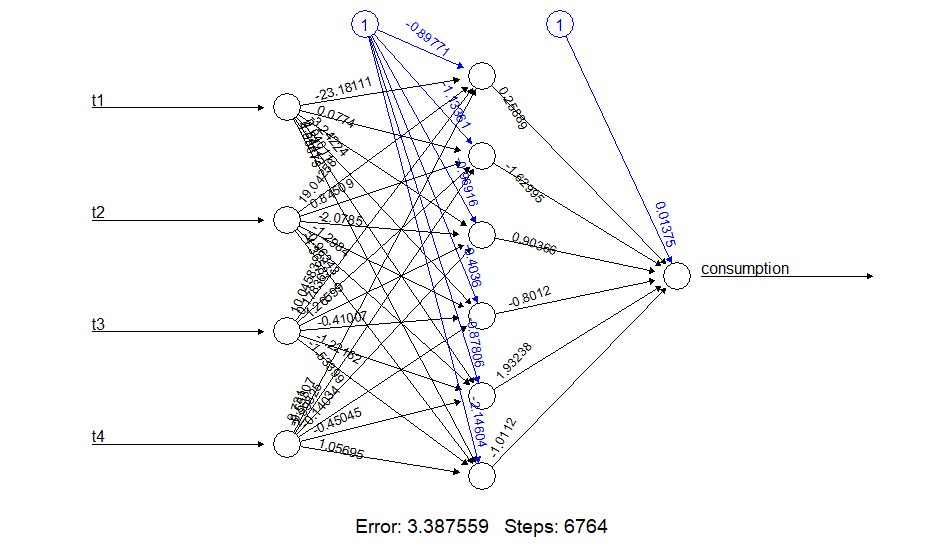


Figure 62

#### **Model 10 (Matrix 3, hidden Layer = 1, hidden layer nodes = 6, linear = false, act.fct = logistic)**

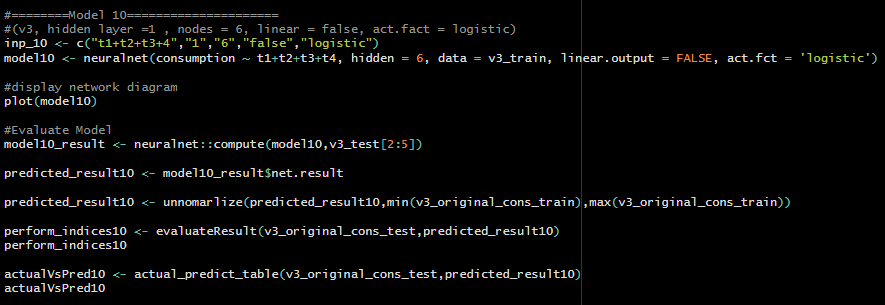


Figure 63

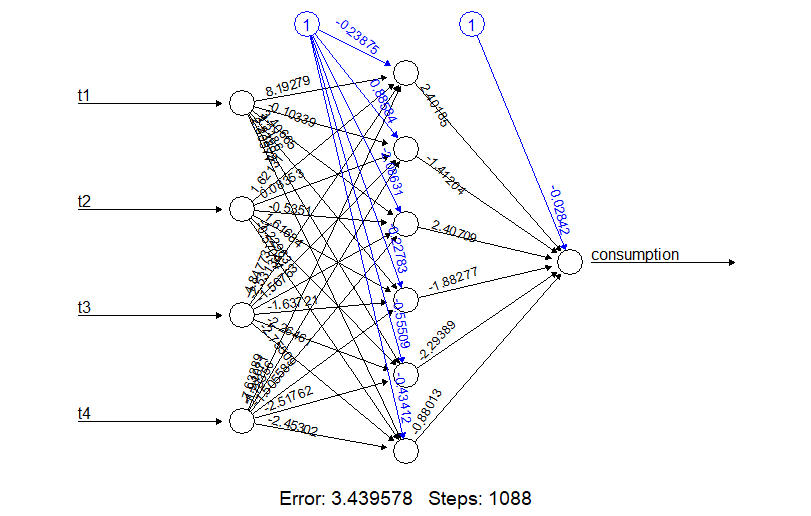


Figure 64

#### **Model 11 (Matrix 3, hidden Layer = 2, hidden layer nodes = (6, 8), linear = true)**

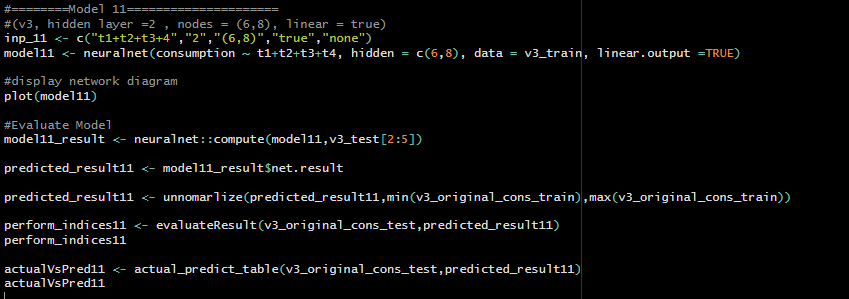


Figure 65

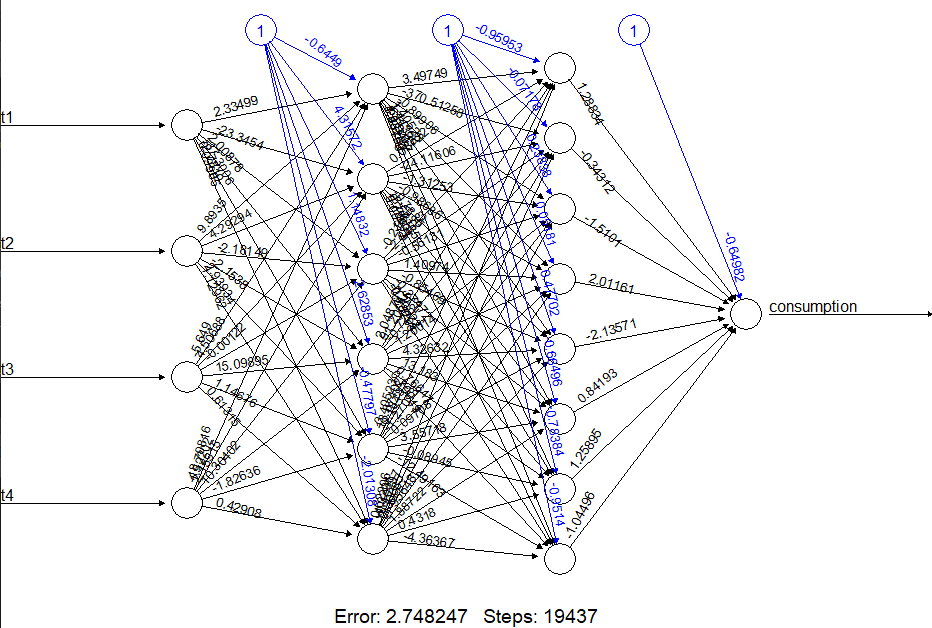


Figure 66

#### **Model 12 (Matrix 3, hidden Layer = 2, hidden layer nodes = (6, 8), linear = false, act.fct = logistic)**

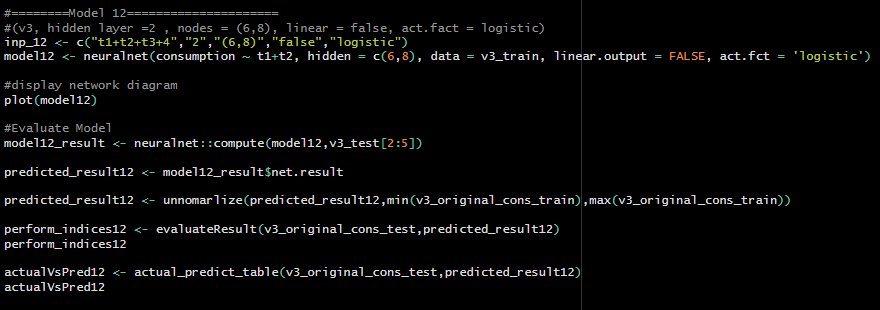


Figure 67

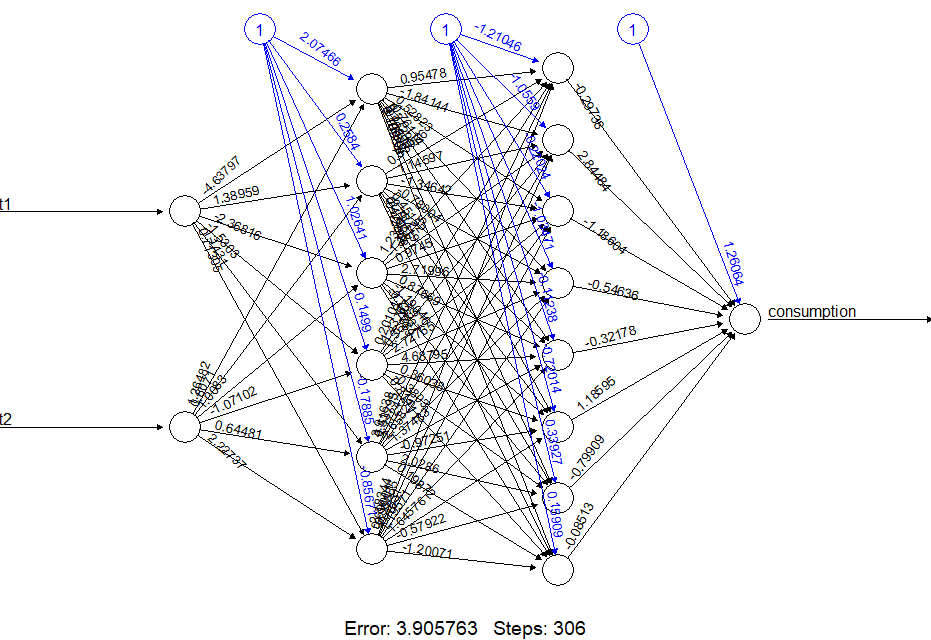


Figure 68

#### **Model 13 (Matrix 3, hidden Layer = 2, hidden layer nodes = (8,10), linear = true)**

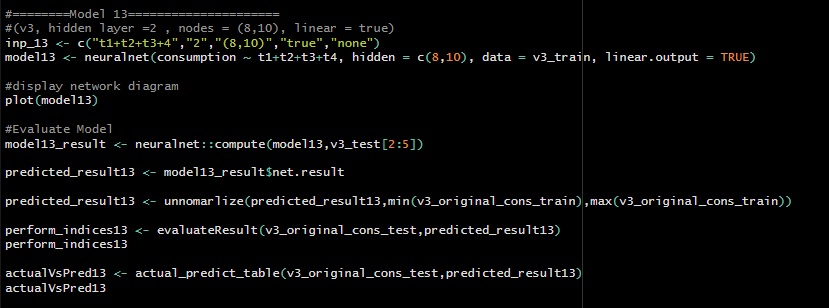


Figure 69

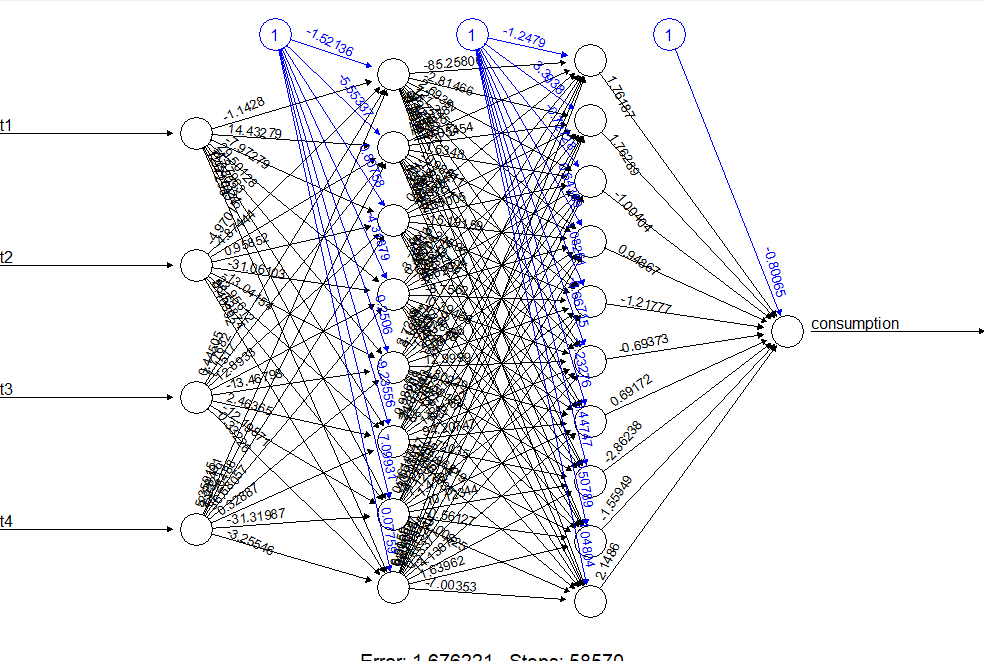


Figure 70

#### **Model 14 (Matrix 4, hidden Layer = 1, hidden layer nodes = 6, linear = false, act.fct = logistic)**

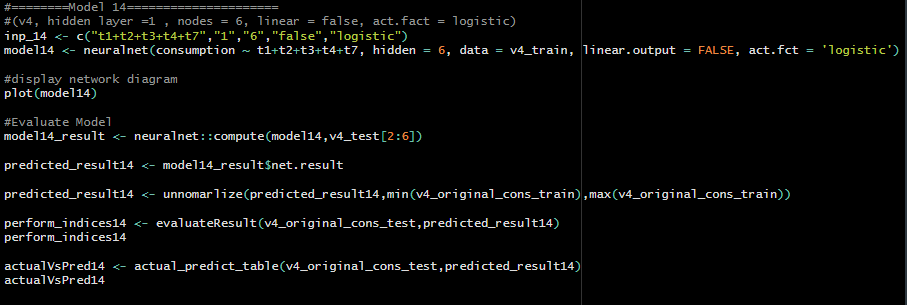


Figure 71

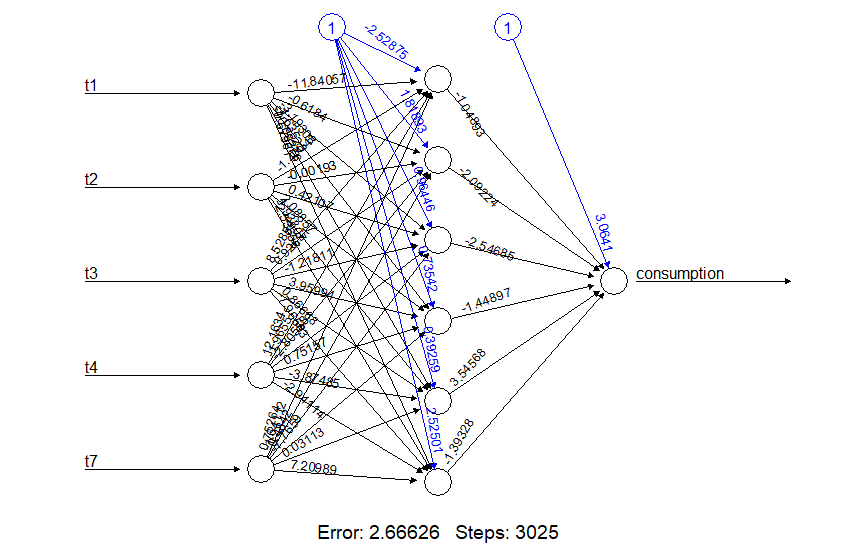


Figure 72

#### **Model 15 (Matrix 4, hidden Layer = 2, hidden layer nodes = (4, 6), linear = false, act.fct = tanh)**

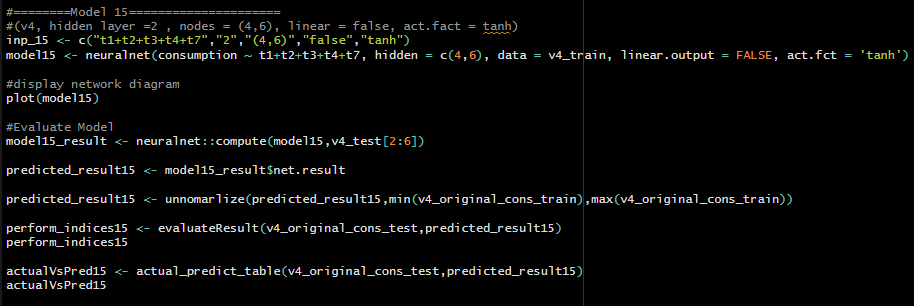


Figure 73

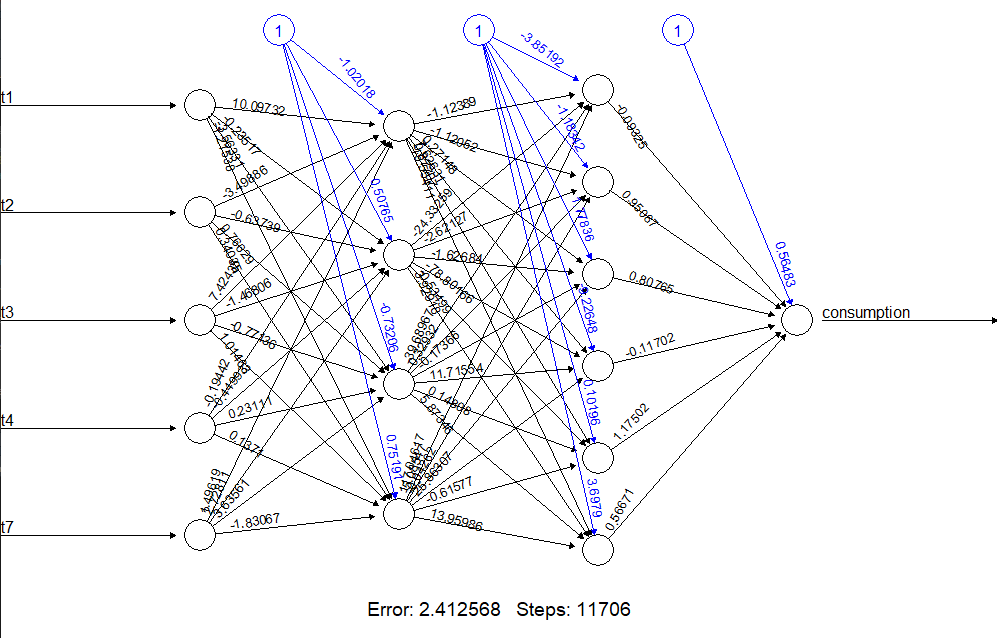


Figure 74

### **Evaluate the Performance**

For evaluate the performances of the trained model we can use standard statistical indices. In this task I have used RMSE, MAE, MAPE and sMAPE standard statistical indices. For ease of use I created a user defined function to evaluate above indices.

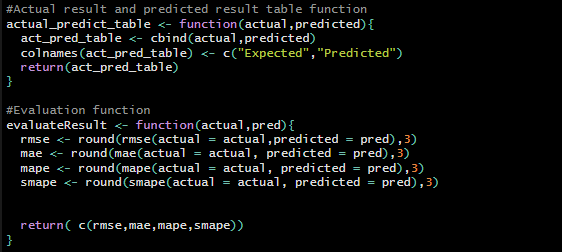


Figure 75

#### **RMSE (Root Mean Squared Error)**

RMSE is a measure of the differences between predicted and actual values using the square root of the average squared differences between predicted and actual value. This is sensitive to larger errors and outliers which make it useful when the accuracy of large errors is important.

#### **MAE (Mean Absolute Error)**

MAE calculates the average absolute difference between predicted and actual values. Basically, MAE measure the average magnitude of the errors in a set of predictions without considering their direction. This is less sensitive to outliers than RMSE which is useful when accuracy of small errors is important.

#### **MAPE (Mean Absolute Percentage Error)**

MAPE measures the percentage difference between predicted and actual values. This calculate the average of absolute percentage differences between predicted values and actual values. This method useful when we need to accuracy of the model need to be expressed as a percentage. However, it has a limitations when actual values are close to zero.

#### **sMAPE (Symmetric Mean Absolute Percentage Error)**

This is a symmetric version of MAPE that also measure the average percentage difference between predicted and actual values, where both predicted and actual values are in the denominator. It is less sensitive to outliers than other methods, and it can be useful when there is no clear directionality in the errors.

### **Comparison Table**

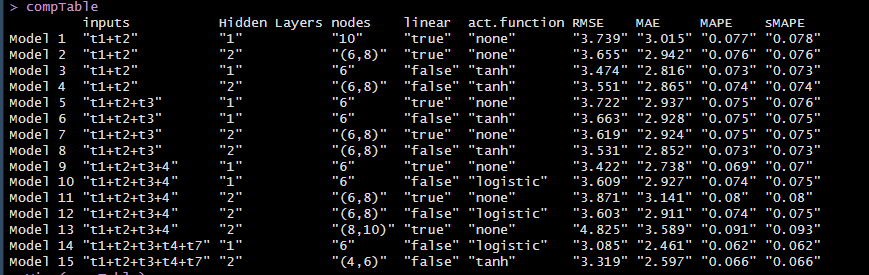


Figure 76 Comparison Table



Figure 77 Comparison Table

From analyze above comparison table we can clearly identify the best model from the lowest of all the evaluation indices. For one layer model best one is Model 14 and as for the two layer model best one is Model 15.

### **Efficiency of the selected models**

The number of weight parameters in a neural network is an important factor in determining its efficiency. Generally, a network with fewer weight parameters will be more efficient, as it requires less computational resources and can be trained faster.

#### **One layer model (Model 14)**

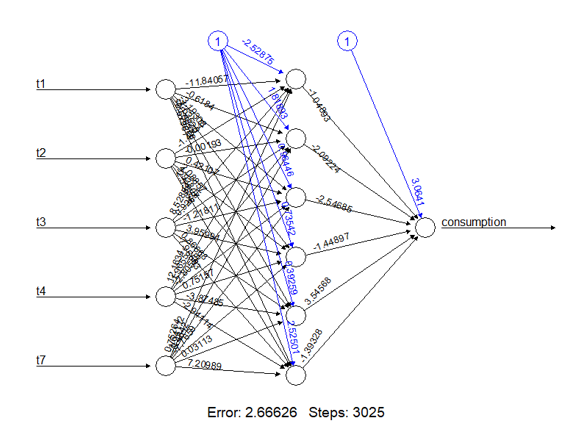


Figure 78 Mode 14

Weights between Input and Hidden Layer = no. of Inputs \* no. of node in Hidden Layer

= (5+1) \* 6

= 36 parameters

Weights between Hidden and Output Layers = no. of nodes in Hidden Layer \* no. of nodes in Output Layer

= (6+1) \* 1

= 7 parameters

Total Number of Weight Parameters = 36 + 7

= 43 parameters

#### **Two layer model (Model 15)**

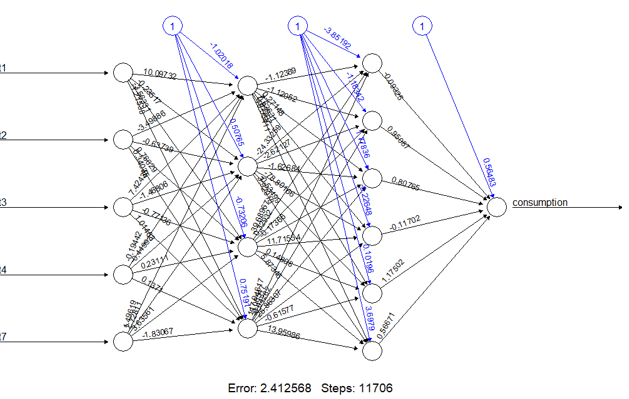


Figure 79 Model 15

Weights between Input and Hidden Layer = no. of Inputs \* no. of node in Hidden Layer

= (5+1) \* 4

= 24 parameters

Weights between 1st Hidden and 2nd Hidden Layers = no. of nodes in 1st Hidden Layer \* no. of nodes in 2nd Hidden Layers

= (4+1) \* 6

= 30 parameters

Weights between 2nd Hidden and Output Layers = no. of nodes in 2nd Hidden Layer \* no. of nodes in Output Layer

= (6+1) \* 1

= 7 parameters

Total Number of Weight Parameters = 24 + 30 + 7

= 61 parameters

From above 2 models I prefer Model 14 which is one layer model because comparing those two models which are best two models we choose from the comparison table according to the highest accuracy, model 14 has the best performance results and also considering the parameters model 15 is more complex than model 14 which may also be prone to overfitting, particularly if the dataset is small.