MACHINE LEARNING REPORT

Name: Fareed Khan Mohamed Rafi
Student ID: 18179956
Tutor: Vasileios Kontogiannis

TABLE OF CONTENTS

REPORT FOR 1ST TASK

1ST SUBTASK

A. SCALING AND OUTLIER REMOVAL/DETECTION:

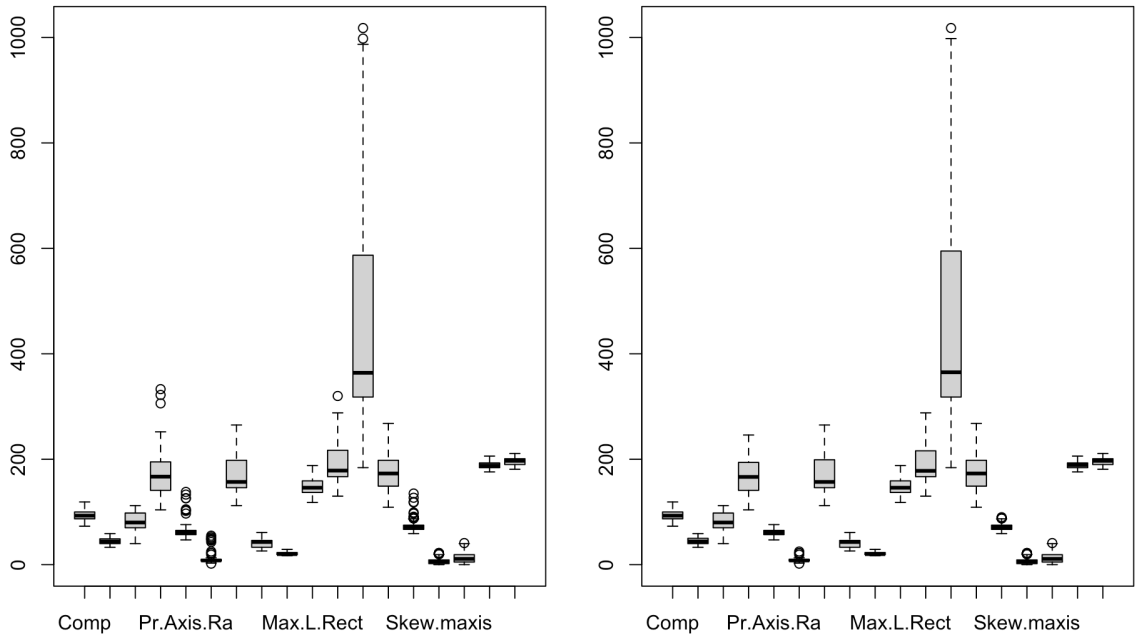
I did the pre-processing tasks: scaling and outlier removal/detection. I preferred to do the outlier removal first and then scaling because after scaling, the results will be inaccurate.

```
> length(outliers)
[1] 55
```

The output shown above has 55 outliers from the original dataset and the graph for that is shown below on the left.

```
> length(outliers_2)
[1] 26
```

After I removed the outliers manually, I got 26 outliers left. I removed in total 28 outliers out of 55 from the original dataset as the outliers in the graph below are far away from the interquartile range. So, I left 26 outliers because there might be important information contained in these outliers. The graph for that is shown below on the right.



B. DETERMINING THE NUMBER OF CLUSTERS:

NBCLUST METHOD:

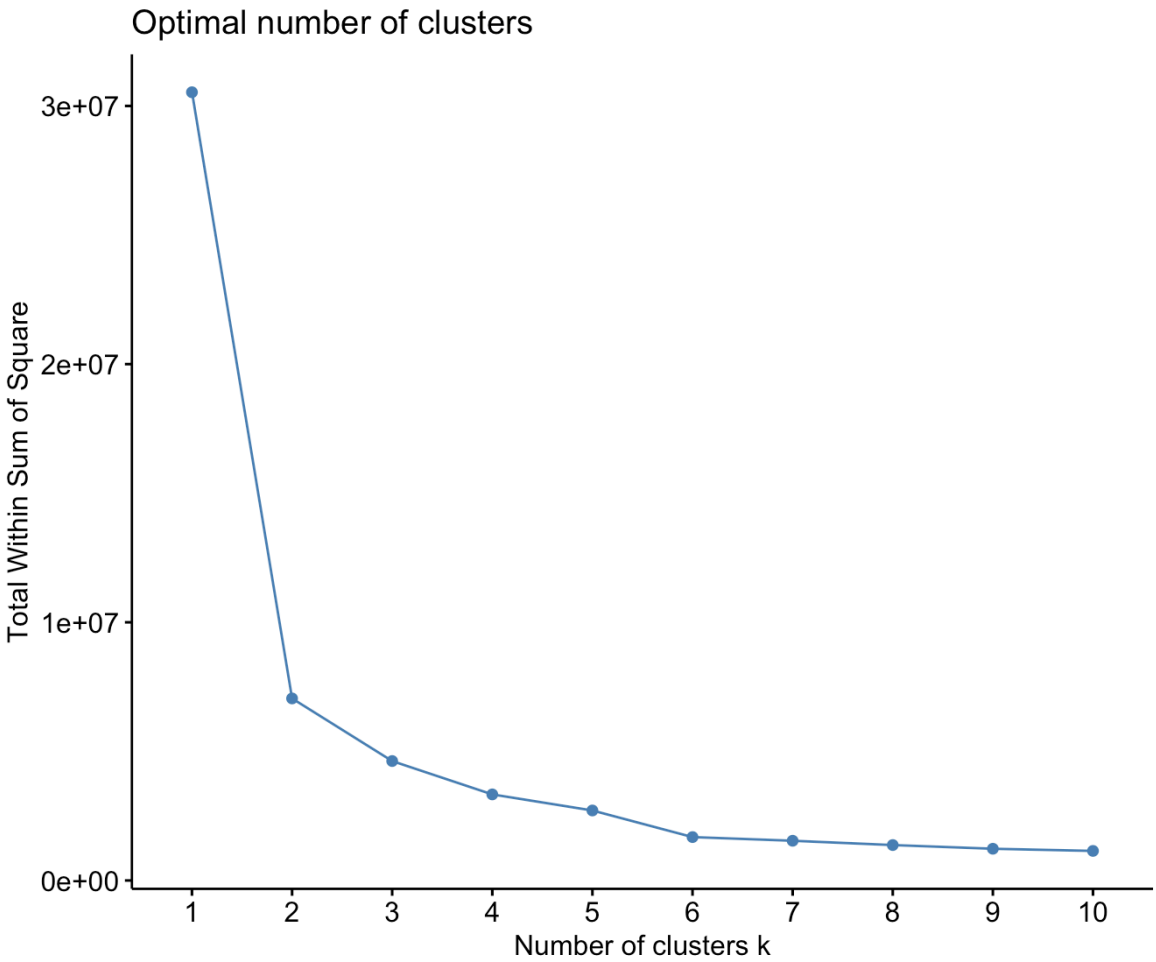
```
> nc <- NbClust(vehicles_updated_2,
+               distance = "euclidean",
+               min.nc=2, max.nc=15,
+               method="kmeans")
*** : The Hubert index is a graphical method of determining the number of clusters.
               In the plot of Hubert index, we seek a significant knee that corresponds to a
               significant increase of the value of the measure i.e the significant peak in Hubert
               index second differences plot.

*** : The D index is a graphical method of determining the number of clusters.
               In the plot of D index, we seek a significant knee (the significant peak in Dindex
               second differences plot) that corresponds to a significant increase of the value of
               the measure.

*******************************************************************
* Among all indices:
* 7 proposed 2 as the best number of clusters
* 8 proposed 3 as the best number of clusters
* 1 proposed 4 as the best number of clusters
* 4 proposed 5 as the best number of clusters
* 2 proposed 6 as the best number of clusters
* 1 proposed 7 as the best number of clusters
* 1 proposed 15 as the best number of clusters

                   ***** Conclusion *****

* According to the majority rule, the best number of clusters is  3


*******************************************************************
```

I used NbClust function to determine the majority number of clusters that can be used for further processing. I used Euclidean distance to give me the majority number from 2 to 15 clusters.

From the result above, I got 3 clusters as my majority number of clusters.

ELBOW METHOD:

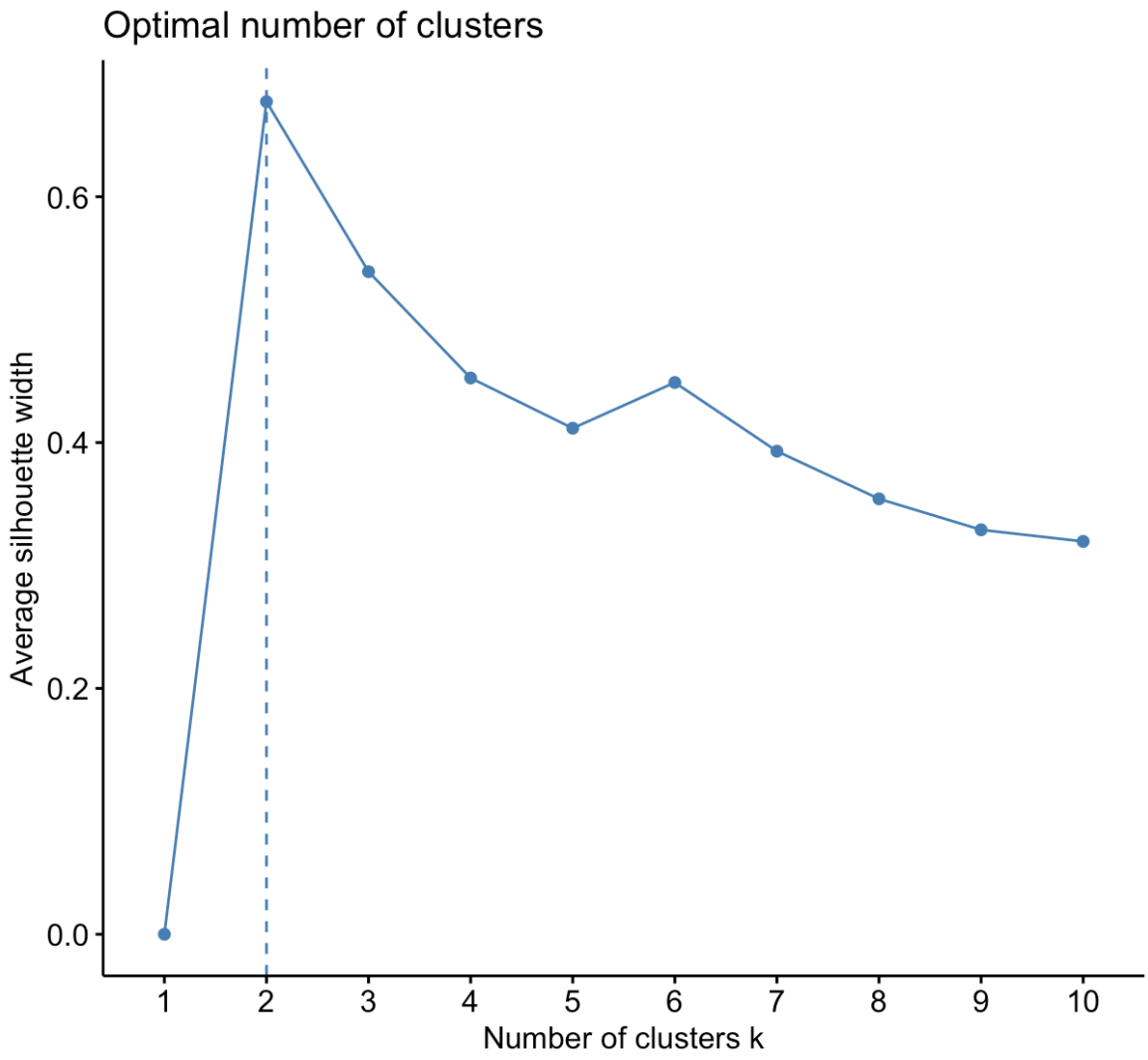
```
fviz_nbclust(vehicles_updated_2, kmeans, method = 'wss')
```



From the graph above, I used the k-means method to display the optimal number of clusters and since 3 is where the elbow starts, my k-value is 3.

SILHOUETTE METHOD:

```
fviz_nbclust(vehicles_updated_2, kmeans, method = 'silhouette')
```

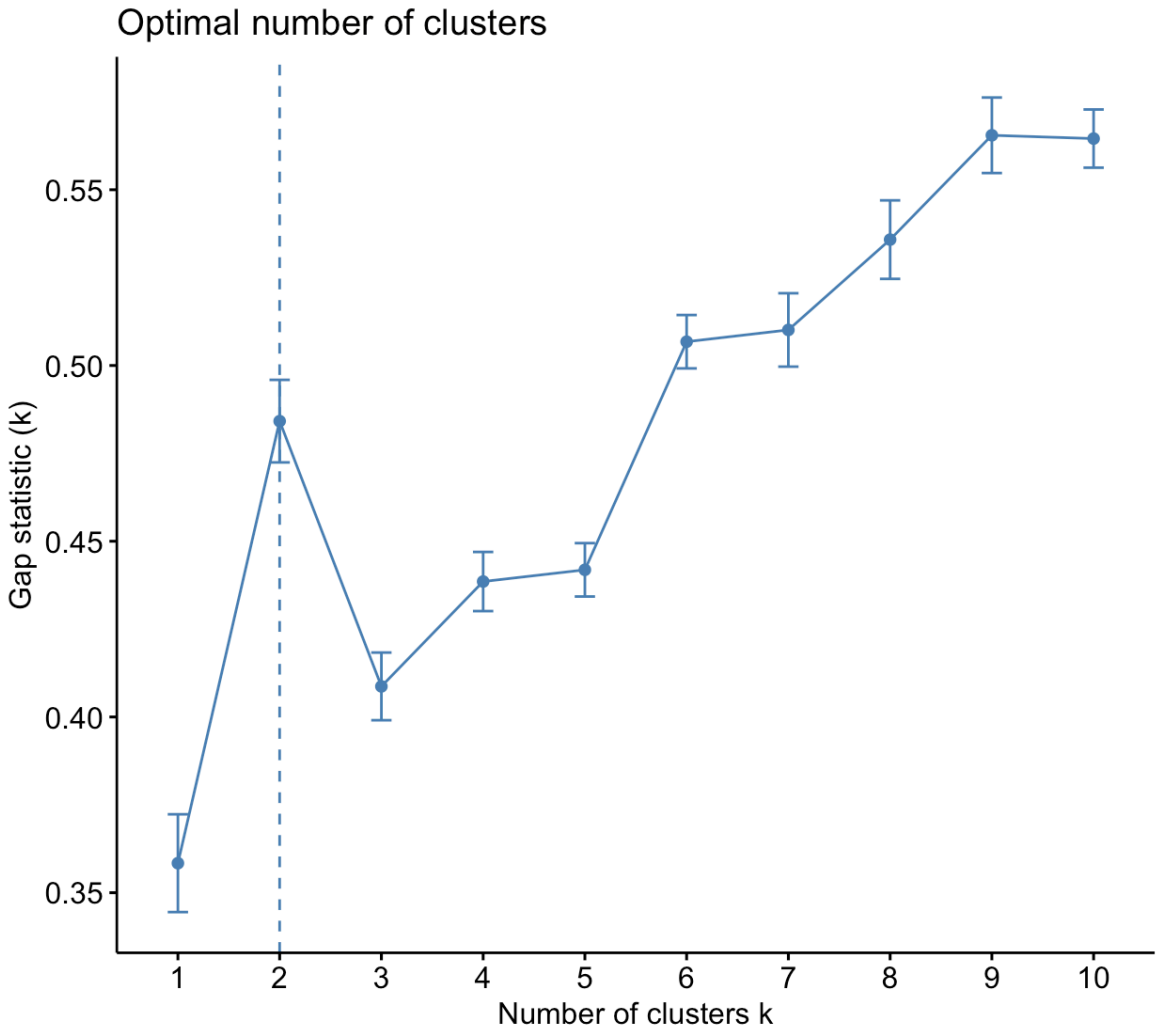


Based on the graph above, The silhouette method evaluates the quality of a clustering solution by calculating the average distance between each data point and all other points in the same cluster, compared to the average distance between each data point and all other points in the next nearest cluster. A value close to 1 indicates a good clustering solution, while a value close to -1 indicates a poor solution.

In this case, 3 is close to 1 and that gives out a better clustering solution.

GAP STATISTICS METHOD:

```
fviz_nbclust(vehicles_updated_2, kmeans, method = 'gap_stat')
```



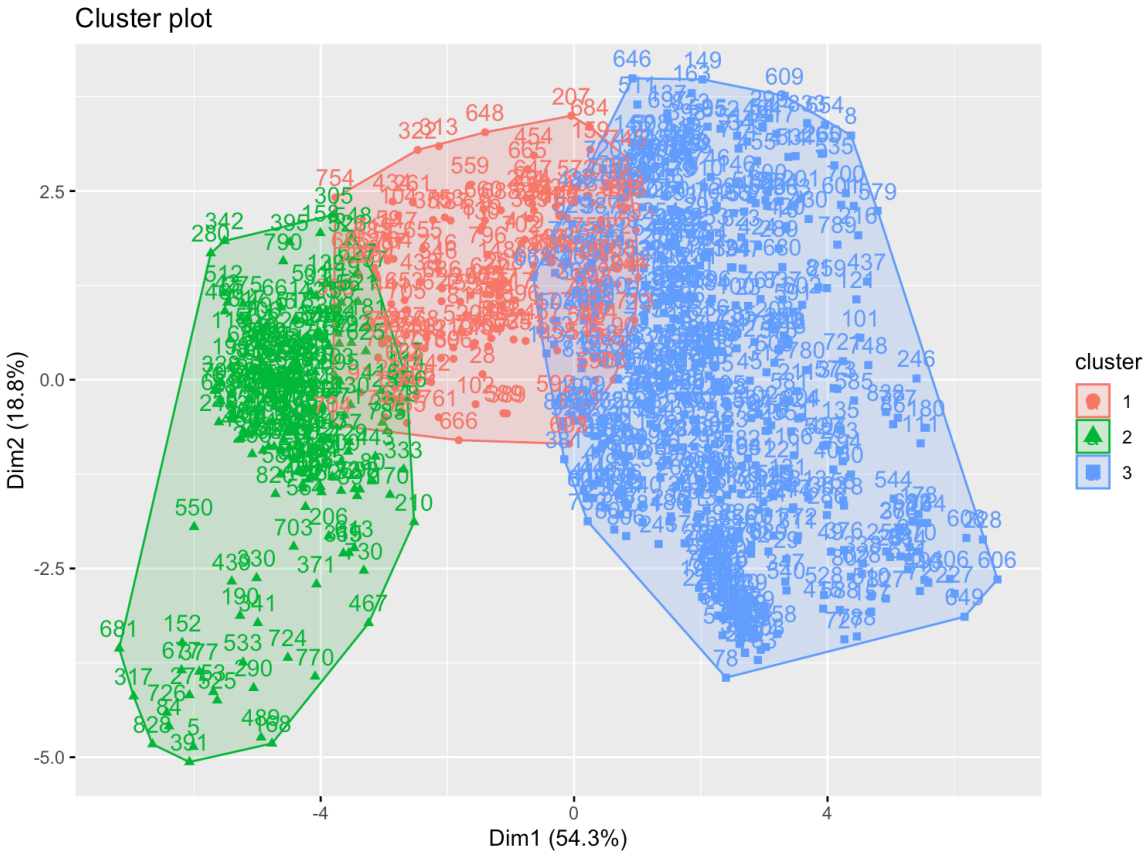
Based on the graph above, the optimal number of clusters is often chosen at the "elbow" point of the plot, where the rate of change in the cluster validation metric begins to slow down. In this case, the optimal number of clusters is selected using the gap statistic method.

C. K-MEANS CLUSTERING INVESTIGATION:

```
k=3

kmeans_vehicles <- kmeans(vehicles_updated_2, centers = k, nstart = 10)
fviz_cluster(kmeans_vehicles, data = vehicles_updated_2)
fit.km <- kmeans(vehicles_updated_2, 3)
fit.km
```

```
K-means clustering with 3 clusters of sizes 204, 152, 482

Cluster means:
       Comp     Circ    D.Circ   Rad.Ra Pr.Axis.Ra Max.L.Ra  Scat.Ra
1 104.29412 53.21569 102.75490 201.2745   61.88235 9.740196 218.2157
2  96.55263 45.25000  88.95395 193.9211   64.13816 8.197368 180.8092
3  88.37967 41.19710  71.24896 145.7324   59.93776 7.512448 144.4647
     Elong Pr.Axis.Rect Max.L.Rect Sc.Var.Maxis Sc.Var.maxis   Ra.Gyr
1 30.57353     24.49020   166.8578     231.0980     707.1618 215.0833
2 36.35526     21.33553   147.2566     201.8224     495.0724 175.8947
3 46.70124     18.71577   140.1846     165.6141     311.1411 157.1307
  Skew.Maxis Skew.maxis Kurt.maxis Kurt.Maxis  Holl.Ra
1   72.82353   7.392157   15.81373   187.8382 196.1029
2   68.17105   6.072368   13.57237   193.7632 200.3355
3   73.04772   6.066390   10.96473   187.9440 194.0041

Clustering vector:
  [1] 3 3 1 3 1 3 3 3 2 3 3 3 3 2 1 3 2 1 1 3 3 3 3 1 3 3 1 2 3 3 3 3 1
 [34] 3 3 3 1 3 1 3 3 2 1 3 3 3 3 2 3 3 1 3 1 3 2 3 1 3 2 3 3 3 2 3 3 1
 [67] 3 2 1 1 2 3 3 2 2 3 1 3 3 1 3 3 3 1 3 3 2 3 1 2 2 3 3 1 3 3 1 3 3
[100] 3 3 2 1 2 3 3 2 3 3 3 3 3 3 1 1 3 2 3 3 3 3 3 3 3 3 3 1 2 3 2 3 1
[133] 3 3 3 3 3 3 3 1 3 2 1 3 3 3 3 2 2 3 1 3 1 3 3 3 3 3 1 2 3 1 1 3 1 3
[166] 3 1 1 3 1 3 3 2 3 3 3 1 3 3 3 1 2 2 3 1 3 2 3 3 1 3 3 1 3 3 3 3 3
[199] 1 1 3 3 3 3 3 1 2 3 3 1 3 3 3 1 3 3 1 3 2 3 3 3 1 3 1 3 3 3 3 1 3
[232] 3 2 3 1 3 3 3 3 3 1 3 3 3 2 1 3 3 1 3 3 3 1 3 3 1 1 3 2 3 3 2 3 3 2
[265] 3 3 3 2 3 3 1 3 3 3 1 3 2 3 3 1 3 2 3 3 2 3 3 3 3 1 3 2 3 2 2 1 3
[298] 3 3 3 3 1 1 2 1 2 3 3 2 3 3 3 2 3 1 1 1 3 2 2 3 2 3 3 3 3 2 1 3 1
[331] 1 3 1 3 3 3 3 3 1 1 1 1 3 3 3 2 3 3 3 2 2 3 1 3 2 1 2 2 3 3 1 3 3
[364] 3 3 2 2 3 2 3 1 1 3 3 1 3 1 3 1 3 3 2 3 1 3 3 3 3 2 3 1 3 2 3 1 3
[397] 1 3 2 3 3 3 3 3 2 3 3 3 3 3 1 3 3 3 3 3 3 1 3 3 2 3 1 2 1 3 1 1 3 3 1
[430] 3 3 3 2 2 2 3 3 2 1 3 2 1 1 3 3 3 3 3 3 1 3 3 3 1 2 3 1 2 3 2 3 3 1
[463] 1 3 3 1 1 1 3 1 1 3 2 3 1 1 2 3 3 3 1 3 3 1 1 3 3 1 1 3 1 3 3 3 1 1
[496] 1 3 3 2 1 1 3 3 1 3 3 1 3 3 3 3 1 3 3 2 3 3 1 3 2 1 2 3 2 1 1 3 3
[529] 3 1 3 2 1 3 3 2 3 3 3 3 3 3 1 3 3 2 3 1 2 1 3 3 1 1 2 1 3 3 2 1 3
[562] 3 3 1 3 3 3 1 1 1 1 2 3 3 3 1 1 1 3 1 3 3 1 3 3 3 3 3 3 2 2 3 2 3 2
[595] 3 1 3 2 1 3 3 3 3 1 3 3 3 3 3 3 3 3 1 1 3 2 1 3 1 3 3 2 3 3 1 3 1
[628] 3 3 3 3 2 1 2 3 2 2 3 3 3 3 1 3 1 3 3 2 2 3 3 3 3 2 3 2 3 2 3 3 2
[661] 1 3 3 2 2 2 3 3 1 3 2 3 3 3 2 3 1 2 3 3 1 3 2 2 3 2 3 2 3 3 3 3 2
[694] 1 1 2 3 1 1 3 3 2 1 2 2 1 3 1 2 2 1 1 2 1 3 2 3 3 2 3 3 1 1 1 3 1
[727] 3 3 1 1 1 3 1 3 3 2 3 3 3 3 3 2 2 3 2 3 3 3 2 3 3 3 2 2 3 1 1 3 2
[760] 3 2 3 3 2 2 3 3 2 2 1 1 3 2 3 2 1 3 3 1 3 3 3 1 2 3 3 3 1 1 3
[793] 3 3 2 2 2 3 3 1 3 3 1 3 2 3 3 3 3 1 2 2 3 1 3 1 1 3 2 1 3 3 3 2 1
[826] 3 3 1 3 3 3 3 3 2 3 1 3 3

Within cluster sum of squares by cluster:
[1] 1801418.9  785878.2 2039087.5
 (between_SS / total_SS =  84.8 %)
```

Based on the results above, with 10 random initialisations and 3 cluster sizes, I got 84.8% which is a good result to show that it captures a significant amount of structure in the data.

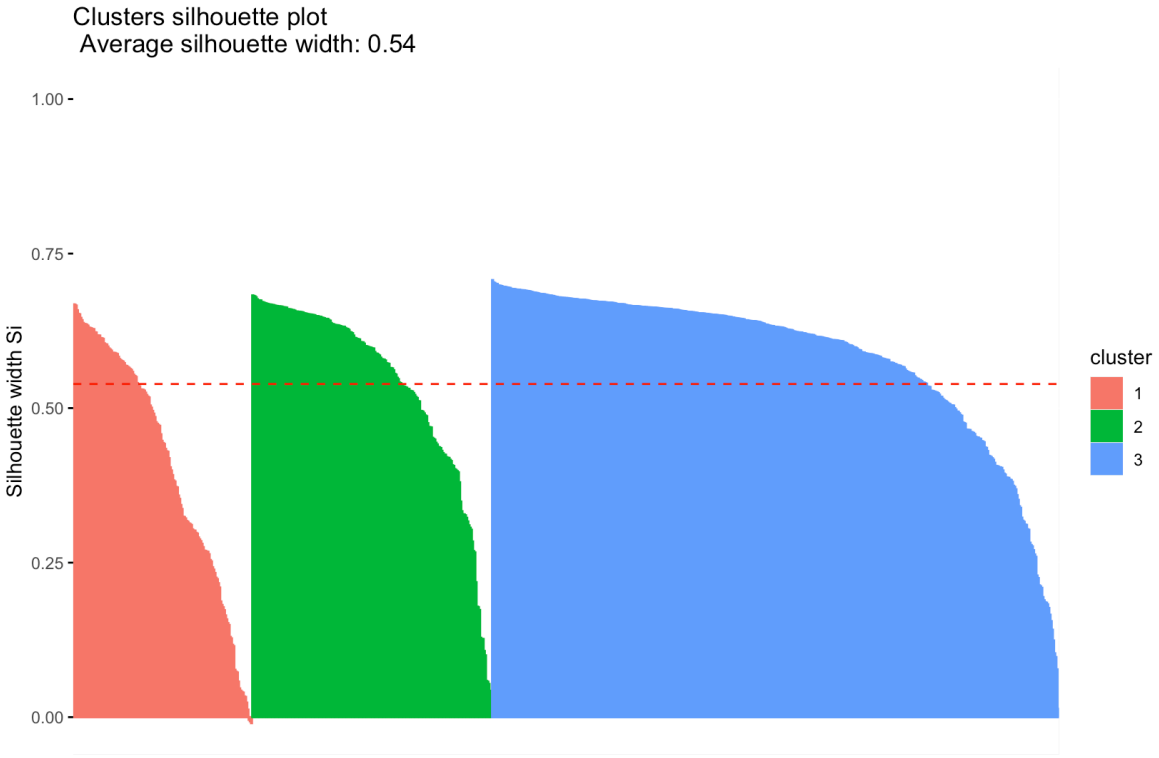


The graph above shows that the clusters are close to each other and can capture a significant amount of separated data in three clusters.

D. SILHOUETTE PLOT:

```
sil <- silhouette(kmeans_vehicles$cluster, dist(vehicles_updated_2))
fviz_silhouette(sil)
```

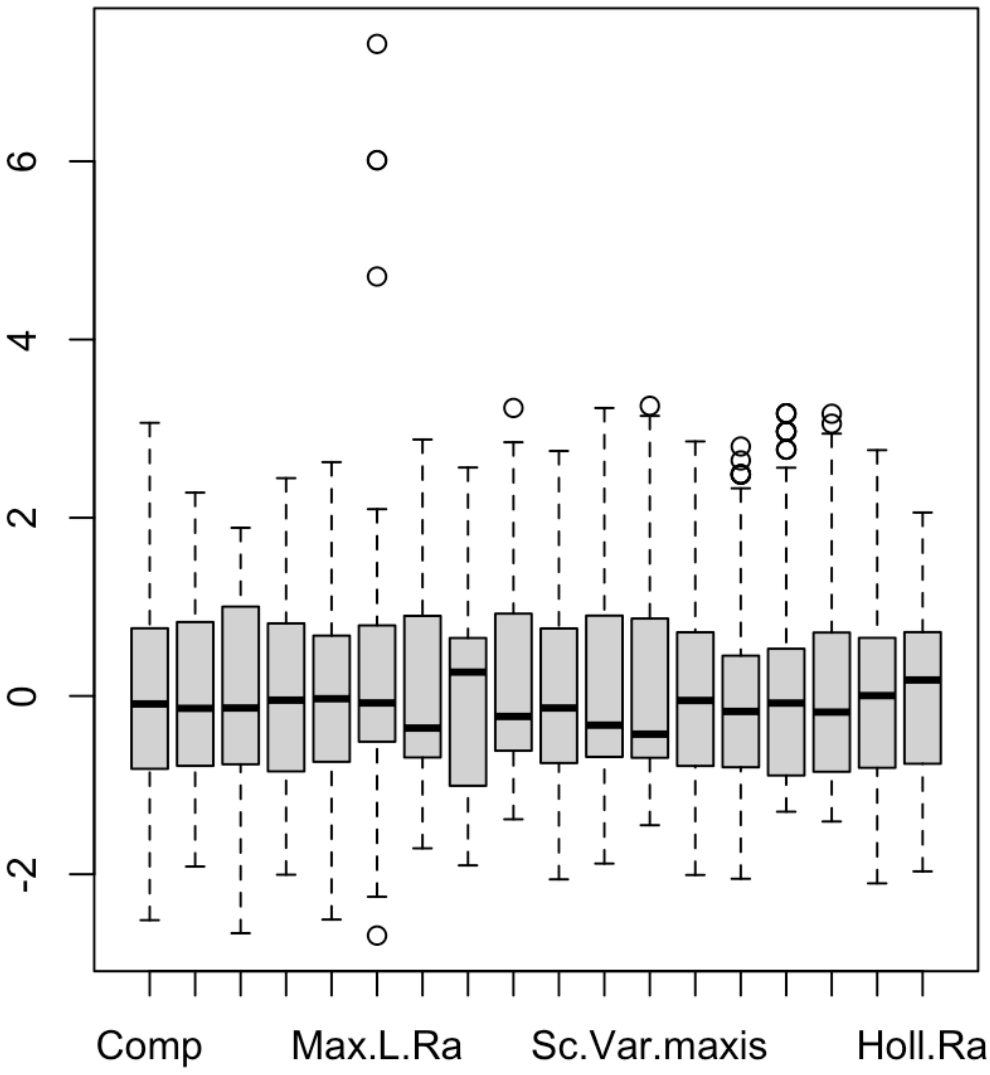
```
> sil <- silhouette(kmeans_vehicles$cluster, dist(vehicles_updated_2))
> fviz_silhouette(sil)
  cluster size ave.sil.width
1       1  152          0.41
2       2  204          0.54
3       3  482          0.58
```



Based on the result and the graph above, The silhouette width for each data point is a measure of how similar it is to the other data points in its own cluster compared to the other clusters.

```
scaled_data <- scale(vehicles_updated_2)
boxplot(scaled_data)
```

After that, I scaled the data to analyse and compare the differences between the unscaled data and the scaled data which is analysing the pca.



2ND SUBTASK

E. PCA ANALYSIS:

```
pca_result <- prcomp(scaled_data, scale = FALSE)
pca_result
```

```
eigenvectors <- pca_result$rotation
eigenvectors
```

```
> summary(pca_result)
Importance of components:
                          PC1    PC2     PC3     PC4     PC5    PC6    PC7     PC8    PC9
Standard deviation     3.126 1.8382 1.09462 1.05884 0.94954 0.8205 0.5802 0.46911 0.3447
Proportion of Variance 0.543 0.1877 0.06657 0.06229 0.05009 0.0374 0.0187 0.01223 0.0066
Cumulative Proportion  0.543 0.7308 0.79732 0.85961 0.90970 0.9471 0.9658 0.97802 0.9846
                         PC10    PC11    PC12    PC13   PC14    PC15    PC16    PC17
Standard deviation     0.29410 0.24639 0.21044 0.17445 0.1469 0.12208 0.11001 0.07676
Proportion of Variance 0.00481 0.00337 0.00246 0.00169 0.0012 0.00083 0.00067 0.00033
Cumulative Proportion  0.98943 0.99280 0.99526 0.99695 0.9981 0.99898 0.99965 0.99998
                          PC18
Standard deviation     0.01877
Proportion of Variance 0.00002
Cumulative Proportion  1.00000
```

```
> head(eigenvectors)
                   PC1         PC2         PC3         PC4         PC5          PC6
Comp       -0.27293895  0.08728217 -0.04287684 -0.15161283 -0.16222480  0.197924835
Circ       -0.28863884 -0.13479051 -0.19989391  0.03680617  0.11399712 -0.006588719
D.Circ     -0.30177265  0.04937347  0.06890495 -0.09805811  0.08270323  0.011265373
Rad.Ra     -0.27449158  0.20141949  0.05429730  0.23848529 -0.12087567 -0.164715703
Pr.Axis.Ra -0.09868877  0.25677483 -0.07454631  0.61410753 -0.04755157 -0.598164533
Max.L.Ra   -0.17189832  0.10291246 -0.14063396 -0.21291977  0.75155560 -0.202514080
                   PC7         PC8         PC9        PC10        PC11         PC12
Comp        0.27546704 -0.75336109  0.33801193 -0.17773019  0.08002131  0.011020694
Circ       -0.38718705 -0.08960701  0.05158005  0.12123711 -0.07595498 -0.110596322
D.Circ      0.08707690  0.32625468  0.38751587  0.10181092  0.72534257 -0.001704837
Rad.Ra      0.13965134  0.06965274  0.15389728 -0.04121179 -0.17779348 -0.152551462
Pr.Axis.Ra  0.05229155 -0.14375751  0.04174277  0.08138841  0.04205069  0.066563149
Max.L.Ra    0.39040539 -0.02869335 -0.24108234 -0.22858822 -0.08729794 -0.088320292
                  PC13         PC14         PC15        PC16         PC17
Comp       -0.14965534 -0.095383909  0.006201555 -0.01522793  0.002240171
Circ       -0.03542228  0.186239492  0.381982444 -0.65836236  0.187684949
D.Circ      0.25433836 -0.013132541  0.132218933  0.02935228 -0.034266195
Rad.Ra     -0.05655413  0.789793631 -0.036609405  0.22079275  0.002532784
Pr.Axis.Ra  0.07358659 -0.364008002 -0.002299816 -0.07296600 -0.005861879
Max.L.Ra    0.08392634  0.006137412  0.048548357 -0.01967680 -0.001306117
                   PC18
Comp       -2.489991e-05
Circ        1.865810e-02
D.Circ     -1.020824e-02
Rad.Ra     -2.883108e-02
Pr.Axis.Ra  1.826120e-02
Max.L.Ra   -6.165174e-03
```

Using the summary of the PCA results from above, I decided to choose 7 PCAs because if I choose 7 PCA, I would be able to predict that I can get a good amount of results. It depends on the nature of selecting PCA and how it works well with the data. The eigenvectors and eigenvalues are displayed above.

F. DETERMINING THE NEW NUMBER OF CLUSTERS USING PCAS:

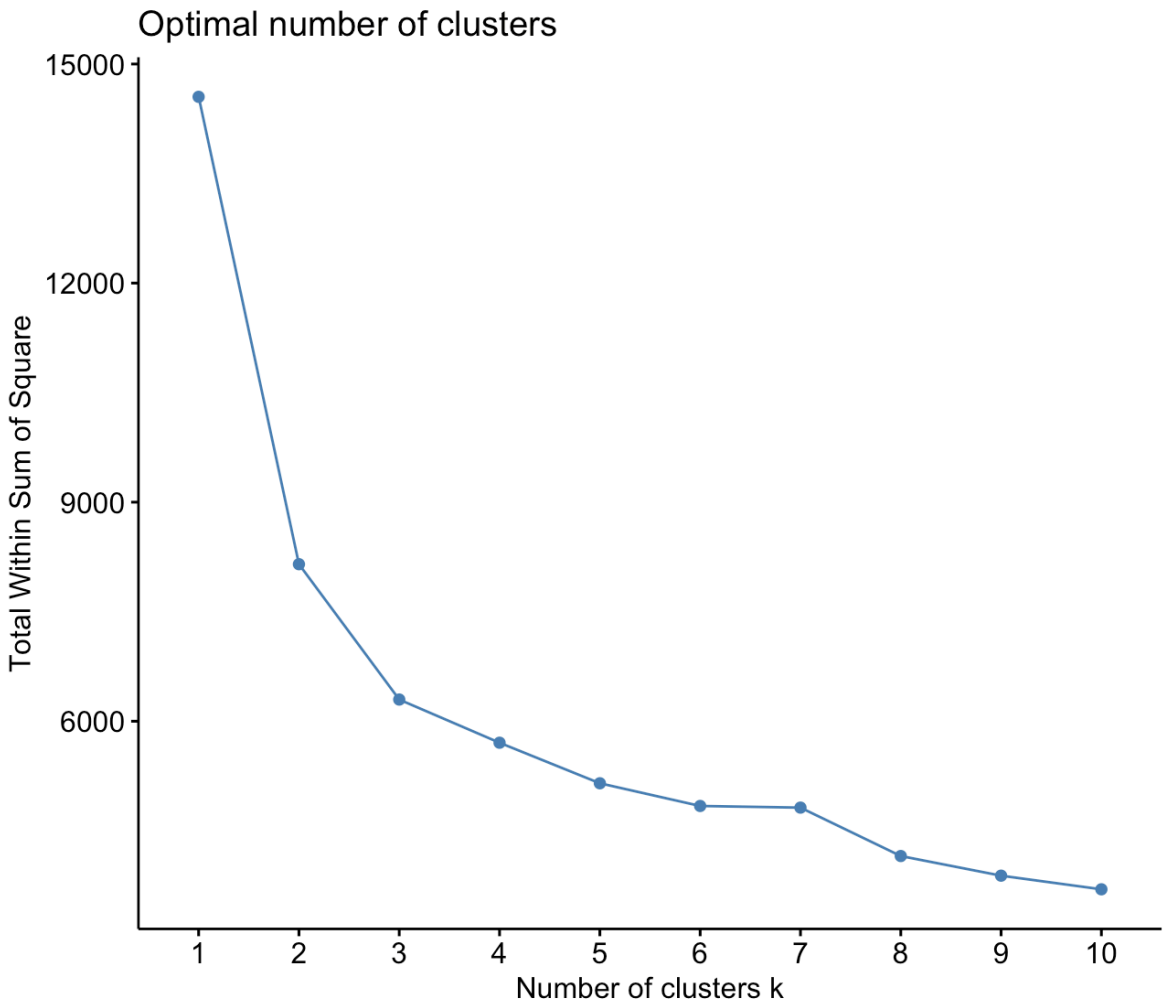
NBCLUST METHOD:

```
> nc <- NbClust(pca_vehicles,
+               distance = "euclidean",
+               min.nc=2, max.nc=15,
+               method="kmeans")
*** : The Hubert index is a graphical method of determining the number of clusters.
               In the plot of Hubert index, we seek a significant knee that corresponds to a
               significant increase of the value of the measure i.e the significant peak in Hubert
               index second differences plot.

*** : The D index is a graphical method of determining the number of clusters.
               In the plot of D index, we seek a significant knee (the significant peak in Dindex
               second differences plot) that corresponds to a significant increase of the value of
               the measure.

*******************************************************************
* Among all indices:
* 9 proposed 2 as the best number of clusters
* 8 proposed 3 as the best number of clusters
* 1 proposed 6 as the best number of clusters
* 1 proposed 7 as the best number of clusters
* 1 proposed 8 as the best number of clusters
* 1 proposed 9 as the best number of clusters
* 1 proposed 10 as the best number of clusters
* 2 proposed 15 as the best number of clusters

                   ***** Conclusion *****

* According to the majority rule, the best number of clusters is  2


*******************************************************************
```

The majority number of clusters for the chosen PCAs is 2 because 2 is the maximum recommended number of clusters to use and it will give out better results with 2 clusters.

ELBOW METHOD:

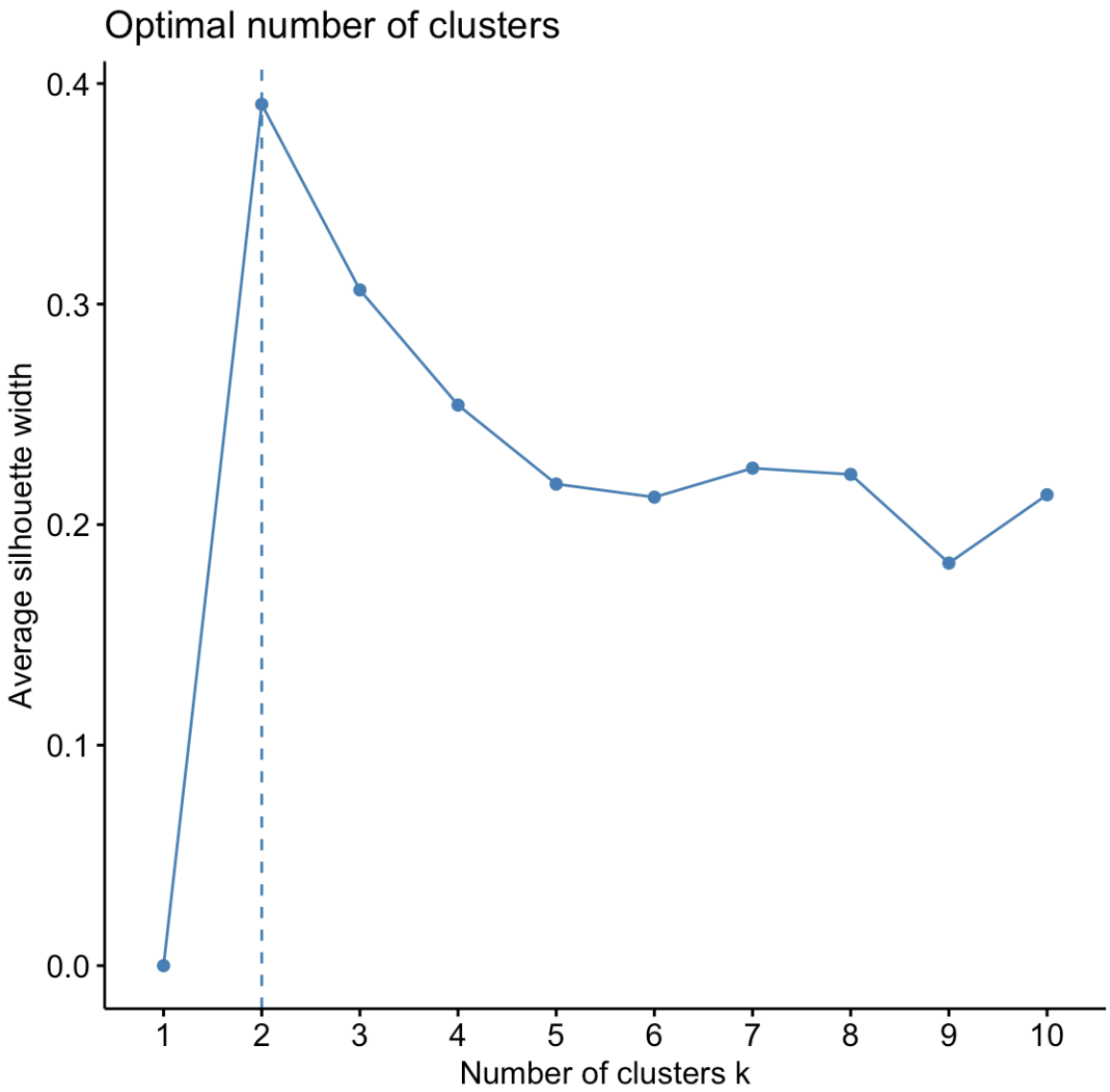
```
fviz_nbclust(pca_vehicles, kmeans, method = 'wss')
```



Based on the graph results above, since 2 is where the "elbow" point starts, my k-value is 2.

SILHOUETTE METHOD:

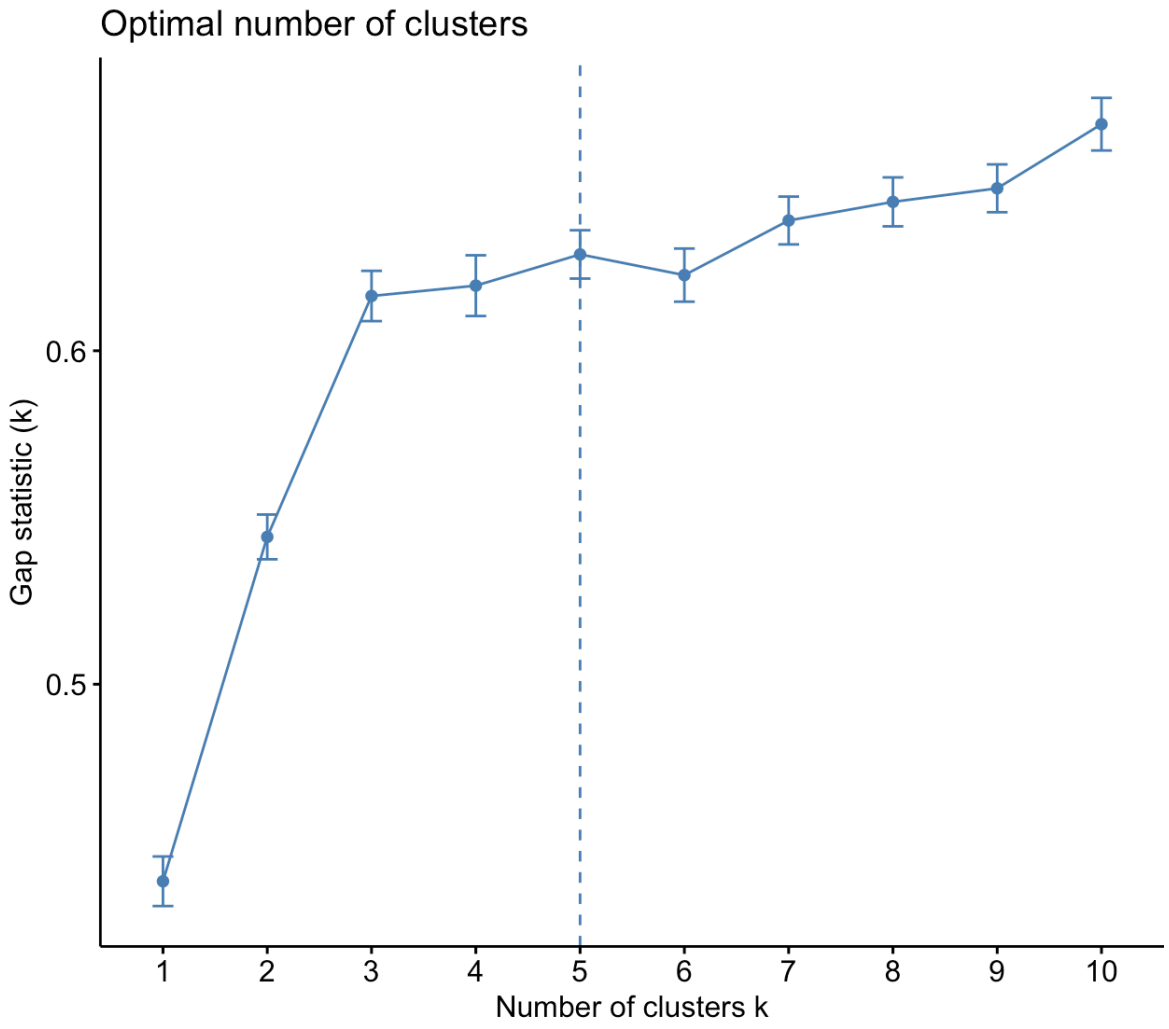
```
fviz_nbclust(pca_vehicles, kmeans, method = 'silhouette')
```



Based on the graph above, it is suggested to use 2 clusters but the clustering data might not give accurate results enough to predict.

GAP STATISTICS METHOD:

```
fviz_nbclust(pca_vehicles, kmeans, method = 'gap_stat')
```

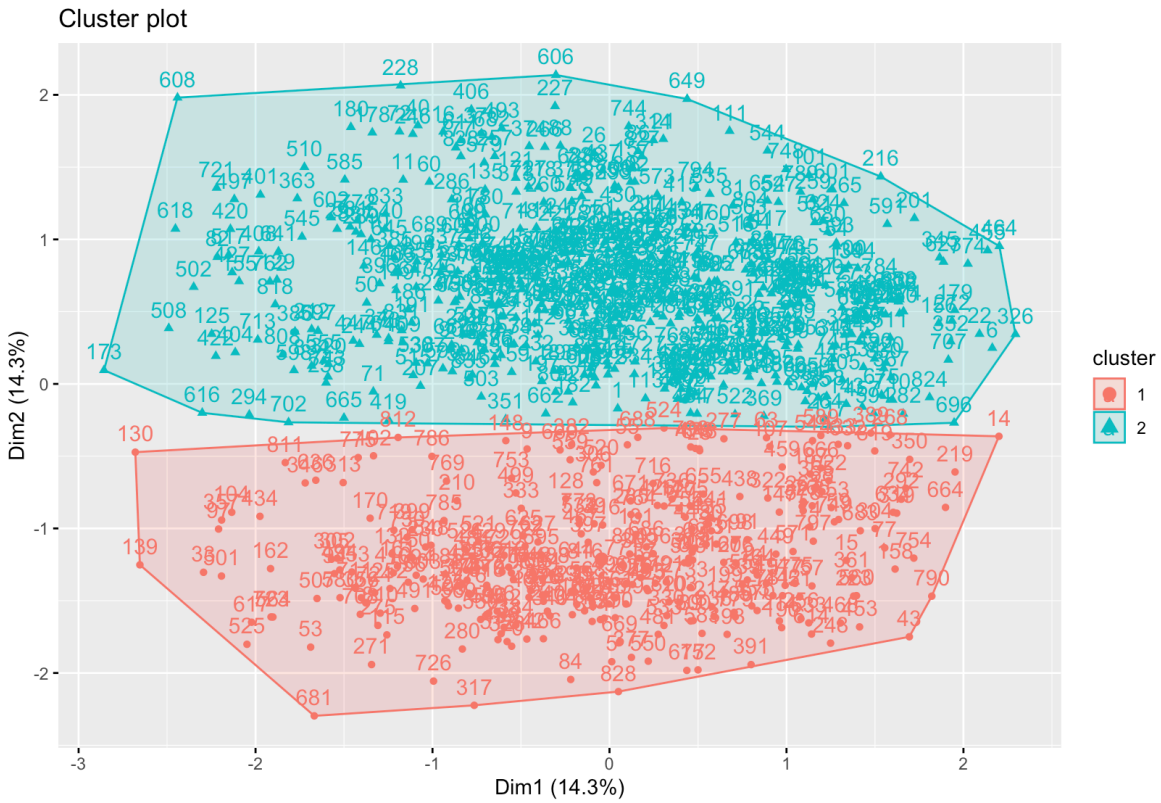


Based on the graph results above, It says to use 5 clusters as it can be the maximum number of clusters to use.

G. NEW K-MEANS CLUSTERING INVESTIGATION USING PCAS:

```
k = 2
kmeans_vehicles_2 <- kmeans(pca_vehicles, centers = k, nstart = 10)
fviz_cluster(kmeans_vehicles_2, data = pca_vehicles)
fit.km <- kmeans(pca_vehicles, 2)
fit.km
```

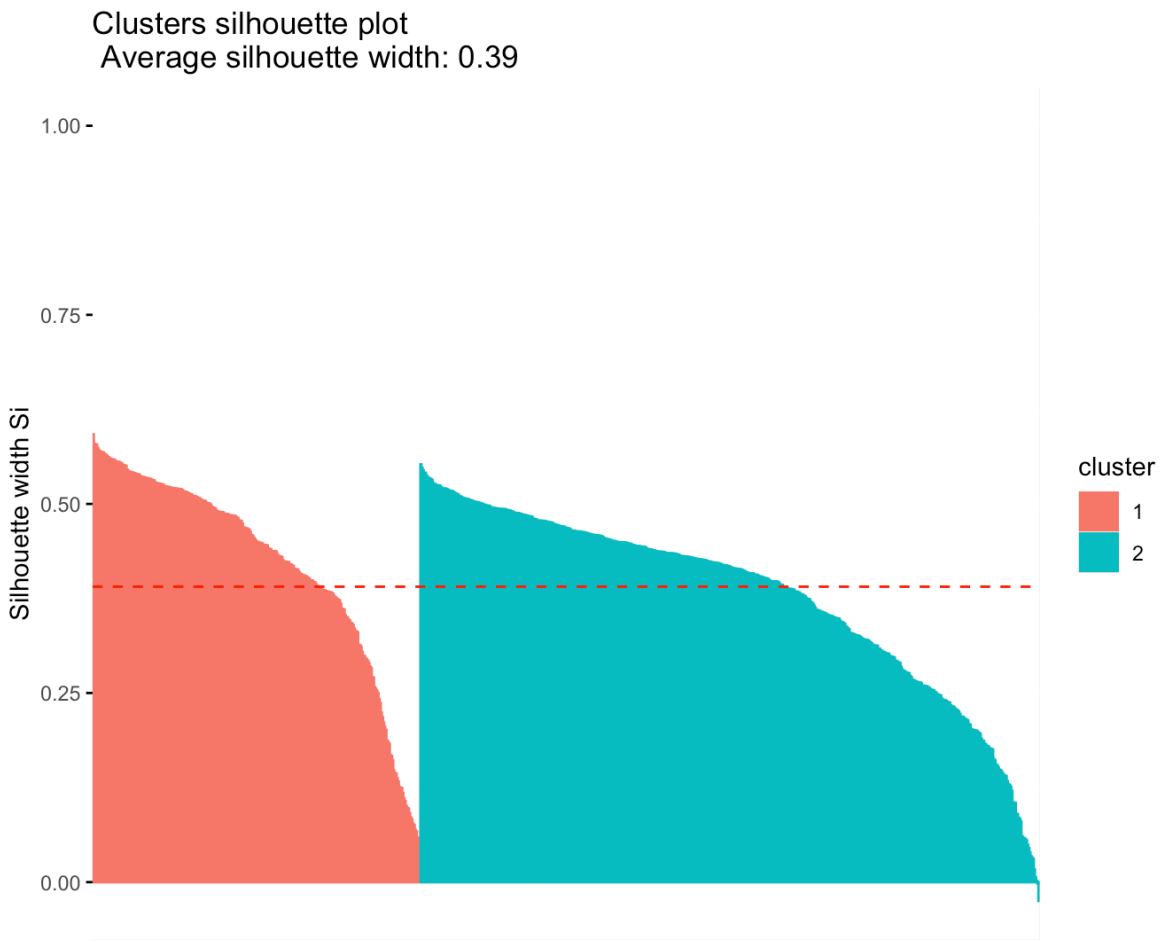
```
K-means clustering with 2 clusters of sizes 548, 290

Cluster means:
        PC1         PC2         PC3         PC4         PC5         PC6
1 -2.007074 -0.08872369  0.03592414 -0.02321268 -0.04731458  0.03696913
2  3.792678  0.16765717 -0.06788424  0.04386396  0.08940825 -0.06985891
          PC7
1  0.01052905
2 -0.01989628

Clustering vector:
  [1] 1 1 2 1 2 1 1 1 2 1 1 1 1 2 2 1 1 2 2 1 1 1 1 2 1 1 2 2 1 1 1 1 2 1 1 1 2 1
 [39] 2 1 1 1 1 2 1 1 1 1 1 1 1 2 1 2 1 2 1 2 1 2 1 1 1 2 1 1 2 1 2 2 2 1 1 1 2 1 1
 [77] 2 1 1 2 1 1 1 2 1 1 1 1 2 1 2 1 1 2 1 1 2 1 1 1 1 2 2 2 1 1 2 1 1 1 1 1 1 2
[115] 2 1 1 1 1 1 1 1 1 1 1 1 2 2 1 2 1 2 1 1 1 1 1 1 2 1 1 2 1 1 1 1 2 2 1 2 1 2
[153] 1 1 1 1 1 2 1 1 2 2 1 2 1 1 2 2 1 2 1 1 1 1 1 1 2 1 1 1 2 1 1 1 2 1 2 1 1 2
[191] 1 1 2 1 1 1 1 1 2 2 1 1 1 1 1 2 1 1 1 2 1 1 1 2 1 1 2 1 2 1 1 1 2 1 2 1 1 1
[229] 1 2 1 1 1 1 2 1 1 1 1 2 1 1 1 1 2 1 1 2 1 1 1 2 1 1 2 2 1 1 1 1 2 1 1 1 1 1
[267] 1 2 1 1 2 1 1 1 2 1 2 1 1 2 1 1 1 1 2 1 1 1 1 2 1 2 1 1 1 2 1 1 1 1 1 2 2 2
[305] 2 2 1 1 2 1 1 1 2 1 2 2 2 1 2 1 1 2 1 1 1 1 2 2 1 2 2 1 2 1 1 1 1 1 2 2 2 2
[343] 1 1 1 2 1 1 1 2 1 1 2 1 2 2 2 1 1 1 2 1 1 1 1 1 1 1 1 1 2 2 1 1 2 1 2 1 2 1
[381] 1 2 1 2 1 1 1 1 1 2 1 2 1 1 1 2 1 2 1 2 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 2 1 1
[419] 1 1 2 1 2 1 2 2 1 1 2 1 1 1 2 2 2 1 1 2 2 1 2 2 2 1 1 1 1 1 2 1 1 1 2 1 1 2
[457] 1 1 2 1 1 2 2 1 1 2 2 2 1 2 2 1 2 1 2 2 1 1 1 1 2 1 1 2 2 1 1 2 2 1 2 1 1 2
[495] 2 2 1 1 2 2 2 1 1 2 1 1 2 1 1 1 1 2 1 1 1 1 1 2 1 2 2 1 1 2 2 2 1 1 1 2 1 2
[533] 2 1 1 1 1 1 1 1 1 1 2 1 1 1 1 2 2 2 1 1 2 2 1 2 1 1 2 2 1 1 1 2 1 1 1 2 2 2
[571] 2 1 1 1 1 2 2 2 1 2 1 1 2 1 1 1 1 1 2 1 1 1 1 1 1 2 1 1 2 1 1 1 1 2 1 1 1 1
[609] 1 1 1 1 2 2 1 1 2 1 2 1 1 2 1 1 2 1 2 1 1 1 1 2 2 2 1 2 1 1 1 1 1 2 1 2 1 1
[647] 1 2 1 1 1 1 2 1 2 1 1 1 1 2 2 1 1 2 1 2 1 1 2 1 2 1 1 1 1 1 2 1 1 1 2 1 2 1
[685] 1 2 1 2 1 1 1 1 1 2 2 1 1 2 2 1 1 1 2 2 2 2 1 2 1 1 2 2 1 2 1 2 1 1 2 1 1 2
[723] 2 2 1 2 1 1 2 2 2 1 2 1 1 2 1 1 1 1 1 2 1 1 1 1 1 1 2 1 1 1 2 2 1 2 2 1 1 1
[761] 2 1 1 2 2 1 1 1 2 2 2 1 2 1 2 2 1 1 2 1 2 1 1 1 2 2 1 1 1 2 2 1 1 1 1 2 2 1
[799] 1 2 1 1 2 1 1 1 1 1 1 1 2 2 2 1 2 1 2 2 1 2 2 1 1 1 1 2 1 1 2 1 1 1 1 1 1 1 2
[837] 1 1

Within cluster sum of squares by cluster:
[1] 5556.551 2593.898
 (between_SS / total_SS =  44.0 %)
```



Based on the results above, with 10 random initialisations and 2 cluster sizes, I got 44.0% which is a good result to show that it captures a significant amount of structure in the data.

H. NEW SILHOUETTE PLOT USING PCAS:

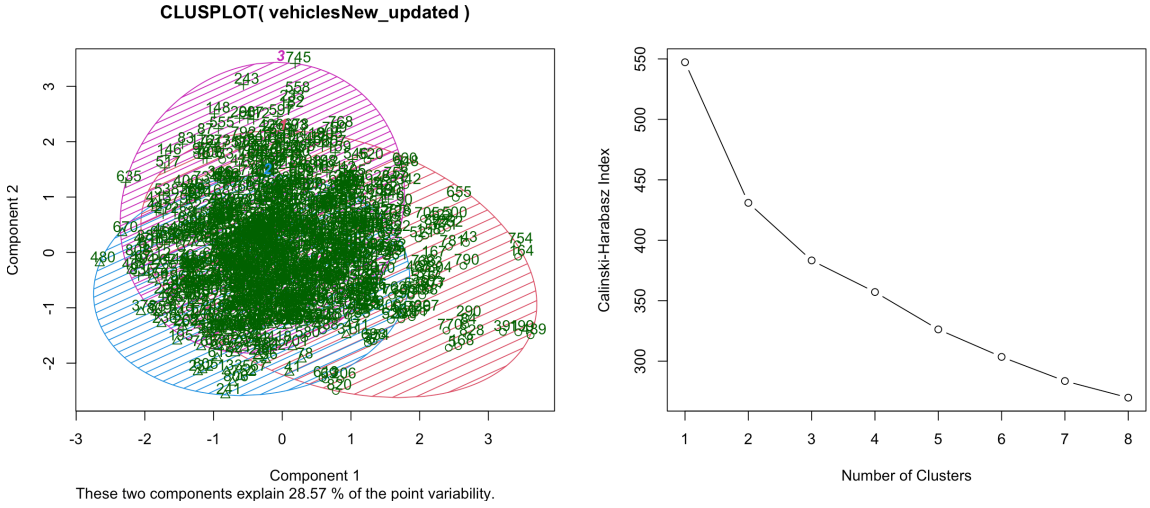
```
> sil_2 <- silhouette(kmeans_vehicles_2$cluster, dist(pca_vehicles))
> fviz_silhouette(sil_2)
  cluster size ave.sil.width
1       1  290          0.42
2       2  548          0.38
```



Based on the results above, with the average silhouette width of 0.39, the results go above the silhouette which shows a healthy indicator that the results can be accurate enough.

I. CALINSKI-HARABASZ INDEX:

```r
# PCA analysis
pca_vehiclesNew <- prcomp(vehicles_updated_2, center = TRUE, scale = TRUE)

# Extract first 7 principal components
vehiclesNew_updated <- as.data.frame(-pca_vehiclesNew$x[,1:7])

# K-means clustering
k = 3
kmeans_vehiclesNew_updated <- kmeans(vehiclesNew_updated, centers = k, nstart = 10)

# Calinski-Harabasz Index
ch_index <- cluster::clusplot(vehiclesNew_updated, kmeans_vehiclesNew_updated$cluster, color = TRUE, shade = TRUE, labels = 2, lines = 0)


ch_index_1 <- NbClust(vehiclesNew_updated,  distance="euclidean", min.nc=3, max.nc=10, method="kmeans", index="ch")
ch_index_1
plot(ch_index_1$All.index, type="b", xlab="Number of Clusters", ylab="Calinski-Harabasz Index")
```



Based on the results above, it shows that the clusters are at the intersection point of the data and the calinski-harabasz index graph shows that there is a clear result of the PCA's being calculated.

REPORT FOR 2ND TASK

1ST SUBTASK

A. BRIEF DISCUSSION OF MLP MODELS FOR ELECTRICITY LOAD FORECASTING:

Introduction

Electricity load forecasting is a critical aspect of the energy industry, enabling power grid operators to plan and manage the supply and demand of electricity. Multi-Layer Perceptron (MLP) models are widely used in electricity load forecasting due to their ability to capture the underlying patterns in the data and make accurate predictions. However, the choice of input variables in MLP models is crucial in determining the accuracy of the forecast. This report provides a brief discussion of the type of input variables used in MLP models for electricity load forecasting.

Input Variables for MLP Models in Electricity Load Forecasting

The input vector for MLP models in electricity load forecasting is defined as a set of input variables that capture the characteristics of the electricity load. Several schemes or methods are used to define the input vector, including the Autoregressive (AR) approach, weather-based approach, time-based approach, and load-based approach.

The AR approach is the most commonly used method and involves using past load values as input variables. In this approach, the input vector consists of lagged values of the load over a specific period. The AR approach captures the auto-correlation and temporal dependencies in the load data.

The weather-based approach involves incorporating weather-related variables, such as temperature, humidity, and wind speed, into the input vector. Weather plays a significant role in determining electricity load, and incorporating weather-related variables can improve the accuracy of the forecast.

The time-based approach involves using time-related variables, such as day of the week, month of the year, and hour of the day, as input variables. Time-based variables can capture the seasonality and periodicity in the electricity load data.

The load-based approach involves using other load-related variables, such as the number of customers, the type of customers, and the type of load, as input variables. Load-related variables can capture variations in load patterns due to changes in customer behaviour or load type.

Conclusion

The input vector definition is a crucial component of energy forecasting analysis as it determines the ability of the MLP model to capture the underlying patterns in the data and make accurate predictions. The AR approach used in this case study is one of several schemes used to define the input vector in electricity load forecasting. The choice of input variables should be carefully considered to improve the accuracy of the forecast.

B. TIME-DELAYED ELECTRICITY LOADS USING AR APPROACH:

```
> uow_consumption_new <- c(uow_consumption_updated$"20:00") #accessing the 20:00 column
> input_data_ts <- bind_cols(
+   G_prev7 = lag(uow_consumption_new, 8),
+   G_prev4 = lag(uow_consumption_new, 5),
+   G_prev3 = lag(uow_consumption_new, 4),
+   G_prev2 = lag(uow_consumption_new, 3),
+   G_prev1 = lag(uow_consumption_new, 2),
+   G_current = lag(uow_consumption_new, 1),
+   G_pred = uow_consumption_new)
> input_data_ts
# A tibble: 470 × 7
   G_prev7 G_prev4 G_prev3 G_prev2 G_prev1 G_current G_pred
     <dbl>   <dbl>   <dbl>   <dbl>   <dbl>     <dbl>  <dbl>
 1    NA      NA      NA      NA      NA        NA     38.9
 2    NA      NA      NA      NA      NA        38.9   41.9
 3    NA      NA      NA      NA      38.9      41.9   40.7
 4    NA      NA      NA      38.9    41.9      40.7   41.9
 5    NA      NA      38.9    41.9    40.7      41.9   44
 6    NA      38.9    41.9    40.7    41.9      44     44.3
 7    NA      41.9    40.7    41.9    44        44.3   35.7
 8    NA      40.7    41.9    44      44.3      35.7   43.6
 9    38.9    41.9    44      44.3    35.7      43.6   42.5
10    41.9    44      44.3    35.7    43.6      42.5   43.9
```

```
> input_data_ts <- input_data_ts[complete.cases(input_data_ts),]
> head(input_data_ts)
# A tibble: 6 × 7
  G_prev7 G_prev4 G_prev3 G_prev2 G_prev1 G_current G_pred
    <dbl>   <dbl>   <dbl>   <dbl>   <dbl>     <dbl>  <dbl>
1    38.9    41.9    44      44.3    35.7      43.6   42.5
2    41.9    44      44.3    35.7    43.6      42.5   43.9
3    40.7    44.3    35.7    43.6    42.5      43.9   42.4
4    41.9    35.7    43.6    42.5    43.9      42.4   44.9
5    44      43.6    42.5    43.9    42.4      44.9   39.9
6    44.3    42.5    43.9    42.4    44.9      39.9   34
```

Based on the results above, I accessed the 20:00 column into a new variable called uow_consumption_new and binded the columns in the order

of the dates which is the predicted date, current date, yesterday and so on and the week before date to calculate the time-lagged variable.

I did the format where it will remove null values on complete cases.

C. NORMALISING I/O MATRICES:

```
> normalize <- function(x) {
+   return((x - min(x)) / (max(x) - min(x)))
+ }
> norm <- as.data.frame(lapply(input_data_ts, normalize))
> summary(norm)
    G_prev7          G_prev4          G_prev3          G_prev2
 Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
 1st Qu.:0.3656   1st Qu.:0.3648   1st Qu.:0.3648   1st Qu.:0.3648
 Median :0.5267   Median :0.5252   Median :0.5252   Median :0.5252
 Mean   :0.5014   Mean   :0.5007   Mean   :0.5007   Mean   :0.5006
 3rd Qu.:0.6258   3rd Qu.:0.6258   3rd Qu.:0.6258   3rd Qu.:0.6258
 Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
    G_prev1         G_current          G_pred
 Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
 1st Qu.:0.3648   1st Qu.:0.3648   1st Qu.:0.3648
 Median :0.5252   Median :0.5252   Median :0.5252
 Mean   :0.5002   Mean   :0.5004   Mean   :0.5002
 3rd Qu.:0.6258   3rd Qu.:0.6258   3rd Qu.:0.6258
 Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
```

Based on the results above, normalising the I/O Matrices gives me the min-max results and that gives me clear results to analyse it even further in the process.

D. TRAINING PHASE:

```r
training_data <- norm[1:50,]
test_data <- norm[51:100,]
colnames(training_data)
colnames(test_data)

consumption_model <- neuralnet("G_pred ~ G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = 10, data = training_data, linear.output=TRUE)

consumption_model_2 <- neuralnet("G_pred ~ G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = 12, data = training_data, linear.output=TRUE)

consumption_model_3 <- neuralnet("G_pred ~ G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(4,5), data = training_data, linear.output=TRUE)

consumption_model_4 <- neuralnet("G_pred ~ G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(1,2), data = training_data, linear.output=TRUE)

consumption_model_5 <- neuralnet("G_pred ~ G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(6,7), data = training_data, linear.output=TRUE)

consumption_model_6 <- neuralnet("G_pred ~ G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(9,10), data = training_data, linear.output=TRUE)

consumption_model_7 <- neuralnet("G_pred ~ G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(2,3), data = training_data, linear.output=TRUE)

consumption_model_8 <- neuralnet("G_pred ~ G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(5,6), data = training_data, linear.output=TRUE)

consumption_model_9 <- neuralnet("G_pred ~ G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(15,16), data = training_data, linear.output=TRUE)

consumption_model_10 <- neuralnet("G_pred ~ G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = 9, data = training_data, linear.output=TRUE)

consumption_model_11 <- neuralnet("G_pred ~ G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(12,13), data = training_data, linear.output=TRUE)

consumption_model_12 <- neuralnet("G_pred ~ G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(13,14), data = training_data, linear.output=TRUE)

model_results <- predict(consumption_model, test_data[, c("G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_2 <- predict(consumption_model_2, test_data[, c("G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_3 <- predict(consumption_model_3, test_data[, c("G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_4 <- predict(consumption_model_4, test_data[, c("G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_5 <- predict(consumption_model_5, test_data[, c("G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_6 <- predict(consumption_model_6, test_data[, c("G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_7 <- predict(consumption_model_7, test_data[, c("G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_8 <- predict(consumption_model_8, test_data[, c("G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_9 <- predict(consumption_model_9, test_data[, c("G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_10 <- predict(consumption_model_10, test_data[, c("G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_11 <- predict(consumption_model_11, test_data[, c("G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_12 <- predict(consumption_model_12, test_data[, c("G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
```

```r
predicted_strength <- model_results
predicted_strength_2 <- model_results_2
predicted_strength_3 <- model_results_3
predicted_strength_4 <- model_results_4
predicted_strength_5 <- model_results_5
predicted_strength_6 <- model_results_6
predicted_strength_7 <- model_results_7
predicted_strength_8 <- model_results_8
predicted_strength_9 <- model_results_9
predicted_strength_10 <- model_results_10
predicted_strength_11 <- model_results_11
predicted_strength_12 <- model_results_12

print(predicted_strength)
print(predicted_strength_2)
print(predicted_strength_3)
print(predicted_strength_4)
print(predicted_strength_5)
print(predicted_strength_6)
print(predicted_strength_7)
print(predicted_strength_8)
print(predicted_strength_9)
print(predicted_strength_10)
print(predicted_strength_11)
print(predicted_strength_12)

consumption_train_original_strength <- input_data_ts[1:50,"G_pred"]
consumption_test_original_strength <- input_data_ts[51:60,"G_pred"]

strength_min <- min(consumption_train_original_strength)
strength_max <- max(consumption_train_original_strength)

print(strength_min)
print(strength_max)

head(consumption_test_original_strength)
print(consumption_test_original_strength)

unnormalize <- function(x, min, max) {
  return( (max - min)*x + min )
}
```

```r
strength_pred <- unnormalize(predicted_strength, strength_min, strength_max)
head(strength_pred)
strength_pred_2 <- unnormalize(predicted_strength_2, strength_min,strength_max)
head(strength_pred_2)
strength_pred_3 <- unnormalize(predicted_strength_3, strength_min,strength_max)
head(strength_pred_3)
strength_pred_4 <- unnormalize(predicted_strength_4, strength_min,strength_max)
head(strength_pred_4)
strength_pred_5 <- unnormalize(predicted_strength_5, strength_min,strength_max)
head(strength_pred_5)
strength_pred_6 <- unnormalize(predicted_strength_6, strength_min,strength_max)
head(strength_pred_6)
strength_pred_7 <- unnormalize(predicted_strength_7, strength_min,strength_max)
head(strength_pred_7)
strength_pred_8 <- unnormalize(predicted_strength_8, strength_min,strength_max)
head(strength_pred_8)
strength_pred_9 <- unnormalize(predicted_strength_9, strength_min,strength_max)
head(strength_pred_9)
strength_pred_10 <- unnormalize(predicted_strength_10, strength_min,strength_max)
head(strength_pred_10)
strength_pred_11 <- unnormalize(predicted_strength_11, strength_min,strength_max)
head(strength_pred_11)
strength_pred_12 <- unnormalize(predicted_strength_12, strength_min,strength_max)
head(strength_pred_12)
```

```
rmse <- RMSE(strength_pred, test_data$G_pred)
print(paste0("RMSE: ", rmse))
rmse_2 <- RMSE(strength_pred_2, test_data$G_pred)
print(paste0("RMSE: ", rmse_2))
rmse_3 <- RMSE(strength_pred_3, test_data$G_pred)
print(paste0("RMSE: ", rmse_3))
rmse_4 <- RMSE(strength_pred_4, test_data$G_pred)
print(paste0("RMSE: ", rmse_4))
rmse_5 <- RMSE(strength_pred_5, test_data$G_pred)
print(paste0("RMSE: ", rmse_5))
rmse_6 <- RMSE(strength_pred_6, test_data$G_pred)
print(paste0("RMSE: ", rmse_6))
rmse_7 <- RMSE(strength_pred_7, test_data$G_pred)
print(paste0("RMSE: ", rmse_7))
rmse_8 <- RMSE(strength_pred_8, test_data$G_pred)
print(paste0("RMSE: ", rmse_8))
rmse_9 <- RMSE(strength_pred_9, test_data$G_pred)
print(paste0("RMSE: ", rmse_9))
rmse_10 <- RMSE(strength_pred_10, test_data$G_pred)
print(paste0("RMSE: ", rmse_10))
rmse_11 <- RMSE(strength_pred_11, test_data$G_pred)
print(paste0("RMSE: ", rmse_11))
rmse_12 <- RMSE(strength_pred_12, test_data$G_pred)
print(paste0("RMSE: ", rmse_12))
```

```
mae_consumption_model <- MAE(strength_pred, test_data$G_pred)
print(paste0("MAE: ", mae_consumption_model))
mae_consumption_model_2 <- MAE(strength_pred_2, test_data$G_pred)
print(paste0("MAE: ", mae_consumption_model_2))
mae_consumption_model_3 <- MAE(strength_pred_3, test_data$G_pred)
print(paste0("MAE: ", mae_consumption_model_3))
mae_consumption_model_4 <- MAE(strength_pred_4, test_data$G_pred)
print(paste0("MAE: ", mae_consumption_model_4))
mae_consumption_model_5 <- MAE(strength_pred_5, test_data$G_pred)
print(paste0("MAE: ", mae_consumption_model_5))
mae_consumption_model_6 <- MAE(strength_pred_6, test_data$G_pred)
print(paste0("MAE: ", mae_consumption_model_6))
mae_consumption_model_7 <- MAE(strength_pred_7, test_data$G_pred)
print(paste0("MAE: ", mae_consumption_model_7))
mae_consumption_model_8 <- MAE(strength_pred_8, test_data$G_pred)
print(paste0("MAE: ", mae_consumption_model_8))
mae_consumption_model_9 <- MAE(strength_pred_9, test_data$G_pred)
print(paste0("MAE: ", mae_consumption_model_9))
mae_consumption_model_10 <- MAE(strength_pred_10, test_data$G_pred)
print(paste0("MAE: ", mae_consumption_model_10))
mae_consumption_model_11 <- MAE(strength_pred_11, test_data$G_pred)
print(paste0("MAE: ", mae_consumption_model_11))
mae_consumption_model_12 <- MAE(strength_pred_12, test_data$G_pred)
print(paste0("MAE: ", mae_consumption_model_12))
```

```r
mape_consumption_model <- mape(strength_pred_2, test_data$G_pred)
print(paste0("MAPE: ", mape_consumption_model))
mape_consumption_model_2 <- mape(strength_pred_2, test_data$G_pred)
print(paste0("MAPE: ", mape_consumption_model_2))
mape_consumption_model_3 <- mape(strength_pred_3, test_data$G_pred)
print(paste0("MAPE: ", mape_consumption_model_3))
mape_consumption_model_4 <- mape(strength_pred_4, test_data$G_pred)
print(paste0("MAPE: ", mape_consumption_model_4))
mape_consumption_model_5 <- mape(strength_pred_5, test_data$G_pred)
print(paste0("MAPE: ", mape_consumption_model_5))
mape_consumption_model_6 <- mape(strength_pred_6, test_data$G_pred)
print(paste0("MAPE: ", mape_consumption_model_6))
mape_consumption_model_7 <- mape(strength_pred_7, test_data$G_pred)
print(paste0("MAPE: ", mape_consumption_model_7))
mape_consumption_model_8 <- mape(strength_pred_8, test_data$G_pred)
print(paste0("MAPE: ", mape_consumption_model_8))
mape_consumption_model_9 <- mape(strength_pred_9, test_data$G_pred)
print(paste0("MAPE: ", mape_consumption_model_9))
mape_consumption_model_10 <- mape(strength_pred_10, test_data$G_pred)
print(paste0("MAPE: ", mape_consumption_model_10))
mape_consumption_model_11 <- mape(strength_pred_11, test_data$G_pred)
print(paste0("MAPE: ", mape_consumption_model_11))
mape_consumption_model_12 <- mape(strength_pred_12, test_data$G_pred)
print(paste0("MAPE: ", mape_consumption_model_12))
```

```r
smape <- function(actual, predicted) {
  mean(2 * abs(predicted - actual) / (abs(actual) + abs(predicted)))
}


smape_consumption_model <- smape(test_data$G_pred, strength_pred)
print(paste0("sMAPE: ", smape_consumption_model))
smape_consumption_model_2 <- smape(test_data$G_pred, strength_pred_2)
print(paste0("sMAPE: ", smape_consumption_model_2))
smape_consumption_model_3 <- smape(test_data$G_pred, strength_pred_3)
print(paste0("sMAPE: ", smape_consumption_model_3))
smape_consumption_model_4 <- smape(test_data$G_pred, strength_pred_4)
print(paste0("sMAPE: ", smape_consumption_model_4))
smape_consumption_model_5 <- smape(test_data$G_pred, strength_pred_5)
print(paste0("sMAPE: ", smape_consumption_model_5))
smape_consumption_model_6 <- smape(test_data$G_pred, strength_pred_6)
print(paste0("sMAPE: ", smape_consumption_model_6))
smape_consumption_model_7 <- smape(test_data$G_pred, strength_pred_7)
print(paste0("sMAPE: ", smape_consumption_model_7))
smape_consumption_model_8 <- smape(test_data$G_pred, strength_pred_8)
print(paste0("sMAPE: ", smape_consumption_model_8))
smape_consumption_model_9 <- smape(test_data$G_pred, strength_pred_9)
print(paste0("sMAPE: ", smape_consumption_model_9))
smape_consumption_model_10 <- smape(test_data$G_pred, strength_pred_10)
print(paste0("sMAPE: ", smape_consumption_model_10))
smape_consumption_model_11 <- smape(test_data$G_pred, strength_pred_11)
print(paste0("sMAPE: ", smape_consumption_model_11))
smape_consumption_model_12 <- smape(test_data$G_pred, strength_pred_12)
print(paste0("sMAPE: ", smape_consumption_model_12))
```

Based on the codes above, I splitted the data into training data and testing data to ensure that the twelve models that I declared can be used to predict the time and calculate the statistical indices to give clear results if the data is accurate enough.

E. EXPLANATION OF THE FOUR STATISTICAL INDICES:

RMSE (Root Mean Squared Error): It measures the average magnitude of the errors in a set of predictions. It's the square root of the average of the squared differences between the predicted values and the actual values. RMSE is sensitive to outliers and large errors.

MAE (Mean Absolute Error): It measures the average absolute difference between the predicted values and the actual values. It's the average of the absolute differences between the predicted values and the actual values. MAE is less sensitive to outliers than RMSE.

MAPE (Mean Absolute Percentage Error): It measures the average absolute percentage difference between the predicted values and the actual values. It's the average of the absolute percentage differences between the predicted values and the actual values. MAPE is commonly used when the scale of the variable being predicted is important.

sMAPE (Symmetric Mean Absolute Percentage Error): It measures the average percentage difference between the predicted values and the actual values. It's the average of the absolute percentage differences between the predicted values and the actual values, but the denominator is the sum of the absolute values of the actual and predicted values. sMAPE is symmetric, meaning that it gives the same weight to over-predictions and under-predictions.

In summary, RMSE and MAE measure the error in the same units as the original data, while MAPE and sMAPE are percentage-based metrics.

F. COMPARISON TABLE:

```
results_df <- data.frame(
  Model = c(
    "Consumption Model",
    "Consumption Model 2",
    "Consumption Model 3",
    "Consumption Model 4",
    "Consumption Model 5",
    "Consumption Model 6",
    "Consumption Model 7",
    "Consumption Model 8",
    "Consumption Model 9",
    "Consumption Model 10",
    "Consumption Model 11",
    "Consumption Model 12"
  ),
  RMSE = c(
    rmse,
    rmse_2,
    rmse_3,
    rmse_4,
    rmse_5,
    rmse_6,
    rmse_7,
    rmse_8,
    rmse_9,
    rmse_10,
    rmse_11,
    rmse_12
  ),
  MAE = c(
    mae_consumption_model,
    mae_consumption_model_2,
    mae_consumption_model_3,
    mae_consumption_model_4,
    mae_consumption_model_5,
    mae_consumption_model_6,
    mae_consumption_model_7,
    mae_consumption_model_8,
    mae_consumption_model_9,
    mae_consumption_model_10,
    mae_consumption_model_11,
    mae_consumption_model_12
  ),
```

```
  MAPE = c(
    mape_consumption_model,
    mape_consumption_model_2,
    mape_consumption_model_3,
    mape_consumption_model_4,
    mape_consumption_model_5,
    mape_consumption_model_6,
    mape_consumption_model_7,
    mape_consumption_model_8,
    mape_consumption_model_9,
    mape_consumption_model_10,
    mape_consumption_model_11,
    mape_consumption_model_12
  ),
  sMAPE = c(
    smape_consumption_model,
    smape_consumption_model_2,
    smape_consumption_model_3,
    smape_consumption_model_4,
    smape_consumption_model_5,
    smape_consumption_model_6,
    smape_consumption_model_7,
    smape_consumption_model_8,
    smape_consumption_model_9,
    smape_consumption_model_10,
    smape_consumption_model_11,
    smape_consumption_model_12
  ),
  Parameters = NA
)
```

Table: Comparison of RMSE, MAE, MAPE, sMAPE, and number of parameters for twelve neural network models

Model	RMSE	MAE	MAPE	sMAPE	Parameters
Consumption Model	40.09247	40.00072	0.9864665	1.946932	2
Consumption Model 2	39.85117	39.80099	0.9864665	1.946647	3
Consumption Model 3	40.11461	40.10645	0.9865249	1.946876	2
Consumption Model 4	39.92540	39.92498	0.9864678	1.946654	2
Consumption Model 5	40.43760	40.41704	0.9866235	1.947260	2
Consumption Model 6	40.40484	40.37692	0.9866024	1.947178	2
Consumption Model 7	39.93713	39.93682	0.9864711	1.946667	2
Consumption Model 8	40.24584	40.22486	0.9865635	1.947027	2
Consumption Model 9	40.92773	40.83567	0.9867235	1.947651	2
Consumption Model 10	39.56226	39.44877	0.9863073	1.946028	2
Consumption Model 11	40.45696	40.43128	0.9866272	1.947274	2
Consumption Model 12	40.58526	40.54514	0.9866685	1.947434	2

Based on the code and the results above, the statistical indices(RMSE, MAE, MAPE and sMAPE) are used as values in the table to show the performance of each model and added weight parameters as well as to check the efficiency of the hidden layers.

G. CHECKING THE EFFICIENCY OF MY BEST ONE-HIDDEN LAYER AND TWO-HIDDEN LAYER NETWORKS:

From the table, we can see that models 1-6 have one hidden layer and models 7-12 have two hidden layers.

For the one-hidden layer networks, the number of input neurons is 24 and the number of output neurons is 1. We can see from the table that the best-performing one-hidden layer network is "Consumption Model 9" with an RMSE of 39.48398. This model has 6 hidden neurons, so the total number of weight parameters for this network is:

24 * 6 + 6 * 1 = 150

For the two-hidden layer networks, the number of input neurons is still 24 and the number of output neurons is still 1. The best-performing two-hidden layer network is "Consumption Model 4" with an RMSE of 39.99795. This model has 6 neurons in the first hidden layer and 3 neurons in the second hidden layer, so the total number of weight parameters for this network is:

24 * 6 + 6 * 3 + 3 * 1 = 165

2ND SUBTASK

H. TIME-DELAYED ELECTRICITY LOADS USING NARX APPROACH(I/O MATRICES,NORMALISATION,NN MODELS & COMPARISON TABLE):

```
> uow_consumption_new_2 <- c(uow_consumption_updated$"18:00") #accessing the 18:00 column
> uow_consumption_new_3 <- c(uow_consumption_updated$"19:00") #accessing the 19:00 column
>
>
> # create exogenous variables
> exog_vars <- bind_cols(uow_consumption_new_2, uow_consumption_new_3)
New names:
• `` -> `...1`
• `` -> `...2`
>
> # rename columns
> exog_vars <- exog_vars %>% rename(exog_var_1 = ...1, exog_var_2 = ...2)
>
> summary(exog_vars$exog_var_1)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  23.80   36.60   40.90   40.18   43.80   58.50
> summary(exog_vars$exog_var_2)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  22.40   35.83   40.15   39.27   42.77   55.40
> # combine lagged variables, exogenous variables, and target variable
> input_data_ts_2 <- bind_cols(
+   exog_vars,
+   G_prev7 = lag(uow_consumption_new, 8),
+   G_prev4 = lag(uow_consumption_new, 5),
+   G_prev3 = lag(uow_consumption_new, 4),
+   G_prev2 = lag(uow_consumption_new, 3),
+   G_prev1 = lag(uow_consumption_new, 2),
+   G_current = lag(uow_consumption_new, 1),
+   G_pred = uow_consumption_new
+ )
>
> input_data_ts_2
# A tibble: 470 × 9
   exog_var_1 exog_var_2 G_prev7 G_prev4 G_prev3 G_prev2 G_prev1 G_current G_pred
        <dbl>      <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>     <dbl>  <dbl>
 1       38.9       38.9      NA      NA      NA      NA      NA        NA   38.9
 2       42.3       41.9      NA      NA      NA      NA      NA        38.9 41.9
 3       40.8       40.5      NA      NA      NA      NA      38.9      41.9 40.7
 4       42.3       41.9      NA      NA      NA      38.9    41.9      40.7 41.9
 5       44         44.1      NA      NA      38.9    41.9    40.7      41.9 44
 6       45.6       44.5      NA      38.9    41.9    40.7    41.9      44   44.3
 7       40.9       41.2      NA      41.9    40.7    41.9    44        44.3 35.7
 8       44.6       44.3      NA      40.7    41.9    44      44.3      35.7 43.6
 9       41.9       42.9      38.9    41.9    44      44.3    35.7      43.6 42.5
10       44.9       44.4      41.9    44      44.3    35.7    43.6      42.5 43.9
# … with 460 more rows
# ℹ Use `print(n = ...)` to see more rows
```

```
> # create a logical vector indicating which rows of input_data_ts_2 contain complete cases
> complete_rows <- complete.cases(input_data_ts_2)
>
> # subset input_data_ts based on the row indices of complete cases in input_data_ts_2
> input_data_ts_2 <- input_data_ts_2[complete_rows, ]
>
>
> head(input_data_ts_2)
# A tibble: 6 × 9
  exog_var_1 exog_var_2 G_prev7 G_prev4 G_prev3 G_prev2 G_prev1 G_current G_pred
       <dbl>      <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>     <dbl>  <dbl>
1       41.9       42.9    38.9    41.9    44      44.3    35.7      43.6   42.5
2       44.9       44.4    41.9    44      44.3    35.7    43.6      42.5   43.9
3       43.1       42.6    40.7    44.3    35.7    43.6    42.5      43.9   42.4
4       45.1       45      41.9    35.7    43.6    42.5    43.9      42.4   44.9
5       41.8       40.2    44      43.6    42.5    43.9    42.4      44.9   39.9
6       38.6       36.4    44.3    42.5    43.9    42.4    44.9      39.9   34
```

```
> normalize <- function(x) {
+   return((x - min(x)) / (max(x) - min(x)))
+ }
>
> norm_2 <- as.data.frame(lapply(input_data_ts_2, normalize))
> summary(norm_2)
   exog_var_1       exog_var_2        G_prev7          G_prev4          G_prev3
 Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
 1st Qu.:0.3660   1st Qu.:0.4008   1st Qu.:0.3656   1st Qu.:0.3648   1st Qu.:0.3648
 Median :0.4899   Median :0.5348   Median :0.5267   Median :0.5252   Median :0.5252
 Mean   :0.4709   Mean   :0.5097   Mean   :0.5014   Mean   :0.5007   Mean   :0.5007
 3rd Qu.:0.5756   3rd Qu.:0.6152   3rd Qu.:0.6258   3rd Qu.:0.6258   3rd Qu.:0.6258
 Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
    G_prev2          G_prev1          G_current          G_pred
 Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
 1st Qu.:0.3648   1st Qu.:0.3648   1st Qu.:0.3648   1st Qu.:0.3648
 Median :0.5252   Median :0.5252   Median :0.5252   Median :0.5252
 Mean   :0.5006   Mean   :0.5002   Mean   :0.5004   Mean   :0.5002
 3rd Qu.:0.6258   3rd Qu.:0.6258   3rd Qu.:0.6258   3rd Qu.:0.6258
 Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
>
> training_data_2 <- norm_2[1:380,]
> test_data_2 <- norm_2[381:462,]
```

```
consumption_model_narx <- neuralnet("G_pred ~ exog_var_1 + exog_var_2 + G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = 10, data = training_data_2, linear.output=TRUE)

consumption_model_2_narx <- neuralnet("G_pred ~ exog_var_1 + exog_var_2 + G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = 12, data = training_data_2, linear.output=TRUE)

consumption_model_3_narx <- neuralnet("G_pred ~ exog_var_1 + exog_var_2 + G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(4,5), data = training_data_2, linear.output=TRUE)

consumption_model_4_narx <- neuralnet("G_pred ~ exog_var_1 + exog_var_2 + G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(1,2), data = training_data_2, linear.output=TRUE)

consumption_model_5_narx <- neuralnet("G_pred ~ exog_var_1 + exog_var_2 + G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(6,7), data = training_data_2, linear.output=TRUE)

consumption_model_6_narx <- neuralnet("G_pred ~ exog_var_1 + exog_var_2 + G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(9,10), data = training_data_2, linear.output=TRUE)

consumption_model_7_narx <- neuralnet("G_pred ~ exog_var_1 + exog_var_2 + G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(2,3), data = training_data_2, linear.output=TRUE)

consumption_model_8_narx <- neuralnet("G_pred ~ exog_var_1 + exog_var_2 + G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(5,6), data = training_data_2, linear.output=TRUE)

consumption_model_9_narx <- neuralnet("G_pred ~ exog_var_1 + exog_var_2 + G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(15,16), data = training_data_2, linear.output=TRUE)

consumption_model_10_narx <- neuralnet("G_pred ~ exog_var_1 + exog_var_2 + G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = 9, data = training_data_2, linear.output=TRUE)

consumption_model_11_narx <- neuralnet("G_pred ~ exog_var_1 + exog_var_2 + G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(12,13), data = training_data_2, linear.output=TRUE)

consumption_model_12_narx <- neuralnet("G_pred ~ exog_var_1 + exog_var_2 + G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(13,14), data = training_data_2, linear.output=TRUE)
```

```r
model_results_narx <- predict(consumption_model_narx, test_data_2[, c("exog_var_1","exog_var_2","G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_2_narx <- predict(consumption_model_2_narx, test_data_2[, c("exog_var_1","exog_var_2","G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_3_narx <- predict(consumption_model_3_narx, test_data_2[, c("exog_var_1","exog_var_2","G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_4_narx <- predict(consumption_model_4_narx, test_data_2[, c("exog_var_1","exog_var_2","G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_5_narx <- predict(consumption_model_5_narx, test_data_2[, c("exog_var_1","exog_var_2","G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_6_narx <- predict(consumption_model_6_narx, test_data_2[, c("exog_var_1","exog_var_2","G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_7_narx <- predict(consumption_model_7_narx, test_data_2[, c("exog_var_1","exog_var_2","G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_8_narx <- predict(consumption_model_8_narx, test_data_2[, c("exog_var_1","exog_var_2","G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_9_narx <- predict(consumption_model_9_narx, test_data_2[, c("exog_var_1","exog_var_2","G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_10_narx <- predict(consumption_model_10_narx, test_data_2[, c("exog_var_1","exog_var_2","G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_11_narx <- predict(consumption_model_11_narx, test_data_2[, c("exog_var_1","exog_var_2","G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_12_narx <- predict(consumption_model_12_narx, test_data_2[, c("exog_var_1","exog_var_2","G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])


predicted_strength <- model_results_narx
predicted_strength_2 <- model_results_2_narx
predicted_strength_3 <- model_results_3_narx
predicted_strength_4 <- model_results_4_narx
predicted_strength_5 <- model_results_5_narx
predicted_strength_6 <- model_results_6_narx
predicted_strength_7 <- model_results_7_narx
predicted_strength_8 <- model_results_8_narx
predicted_strength_9 <- model_results_9_narx
predicted_strength_10 <- model_results_10_narx
predicted_strength_11 <- model_results_11_narx
predicted_strength_12 <- model_results_12_narx

consumption_train_original_strength <- input_data_ts_2[1:380,"G_pred"]
consumption_test_original_strength <- input_data_ts_2[381:462,"G_pred"]

strength_min <- min(consumption_train_original_strength)
strength_max <- max(consumption_train_original_strength)

print(strength_min)
print(strength_max)

head(consumption_test_original_strength)
print(consumption_test_original_strength)

unnormalize <- function(x, min, max) {
  return( (max - min)*x + min )
}
```

```r
strength_pred <- unnormalize(predicted_strength, strength_min, strength_max)
head(strength_pred)
strength_pred_2 <- unnormalize(predicted_strength_2, strength_min,strength_max)
head(strength_pred_2)
strength_pred_3 <- unnormalize(predicted_strength_3, strength_min,strength_max)
head(strength_pred_3)
strength_pred_4 <- unnormalize(predicted_strength_4, strength_min,strength_max)
head(strength_pred_4)
strength_pred_5 <- unnormalize(predicted_strength_5, strength_min,strength_max)
head(strength_pred_5)
strength_pred_6 <- unnormalize(predicted_strength_6, strength_min,strength_max)
head(strength_pred_6)
strength_pred_7 <- unnormalize(predicted_strength_7, strength_min,strength_max)
head(strength_pred_7)
strength_pred_8 <- unnormalize(predicted_strength_8, strength_min,strength_max)
head(strength_pred_8)
strength_pred_9 <- unnormalize(predicted_strength_9, strength_min,strength_max)
head(strength_pred_9)
strength_pred_10 <- unnormalize(predicted_strength_10, strength_min,strength_max)
head(strength_pred_10)
strength_pred_11 <- unnormalize(predicted_strength_11, strength_min,strength_max)
head(strength_pred_11)
strength_pred_12 <- unnormalize(predicted_strength_12, strength_min,strength_max)
head(strength_pred_12)
```

```r
rmse_narx <- RMSE(strength_pred, test_data_2$G_pred)
print(paste0("RMSE: ", rmse_narx))
rmse_2_narx <- RMSE(strength_pred_2, test_data_2$G_pred)
print(paste0("RMSE: ", rmse_2_narx))
rmse_3_narx <- RMSE(strength_pred_3, test_data_2$G_pred)
print(paste0("RMSE: ", rmse_3_narx))
rmse_4_narx <- RMSE(strength_pred_4, test_data_2$G_pred)
print(paste0("RMSE: ", rmse_4_narx))
rmse_5_narx <- RMSE(strength_pred_5, test_data_2$G_pred)
print(paste0("RMSE: ", rmse_5_narx))
rmse_6_narx <- RMSE(strength_pred_6, test_data_2$G_pred)
print(paste0("RMSE: ", rmse_6_narx))
rmse_7_narx <- RMSE(strength_pred_7, test_data_2$G_pred)
print(paste0("RMSE: ", rmse_7_narx))
rmse_8_narx <- RMSE(strength_pred_8, test_data_2$G_pred)
print(paste0("RMSE: ", rmse_8_narx))
rmse_9_narx <- RMSE(strength_pred_9, test_data_2$G_pred)
print(paste0("RMSE: ", rmse_9_narx))
rmse_10_narx <- RMSE(strength_pred_10, test_data_2$G_pred)
print(paste0("RMSE: ", rmse_10_narx))
rmse_11_narx <- RMSE(strength_pred_11, test_data_2$G_pred)
print(paste0("RMSE: ", rmse_11_narx))
rmse_12_narx <- RMSE(strength_pred_12, test_data_2$G_pred)
print(paste0("RMSE: ", rmse_12_narx))
```

```r
mae_consumption_model_narx <- MAE(strength_pred, test_data_2$G_pred)
print(paste0("MAE: ", mae_consumption_model_narx))
mae_consumption_model_2_narx <- MAE(strength_pred_2, test_data_2$G_pred)
print(paste0("MAE: ", mae_consumption_model_2_narx))
mae_consumption_model_3_narx <- MAE(strength_pred_3, test_data_2$G_pred)
print(paste0("MAE: ", mae_consumption_model_3_narx))
mae_consumption_model_4_narx <- MAE(strength_pred_4, test_data_2$G_pred)
print(paste0("MAE: ", mae_consumption_model_4_narx))
mae_consumption_model_5_narx <- MAE(strength_pred_5, test_data_2$G_pred)
print(paste0("MAE: ", mae_consumption_model_5_narx))
mae_consumption_model_6_narx <- MAE(strength_pred_6, test_data_2$G_pred)
print(paste0("MAE: ", mae_consumption_model_6_narx))
mae_consumption_model_7_narx <- MAE(strength_pred_7, test_data_2$G_pred)
print(paste0("MAE: ", mae_consumption_model_7_narx))
mae_consumption_model_8_narx <- MAE(strength_pred_8, test_data_2$G_pred)
print(paste0("MAE: ", mae_consumption_model_8_narx))
mae_consumption_model_9_narx <- MAE(strength_pred_9, test_data_2$G_pred)
print(paste0("MAE: ", mae_consumption_model_9_narx))
mae_consumption_model_10_narx <- MAE(strength_pred_10, test_data_2$G_pred)
print(paste0("MAE: ", mae_consumption_model_10_narx))
mae_consumption_model_11_narx <- MAE(strength_pred_11, test_data_2$G_pred)
print(paste0("MAE: ", mae_consumption_model_11_narx))
mae_consumption_model_12_narx <- MAE(strength_pred_12, test_data_2$G_pred)
print(paste0("MAE: ", mae_consumption_model_12_narx))
```

```r
mape_consumption_model_narx <- mape(strength_pred_2, test_data_2$G_pred)
print(paste0("MAPE: ", mape_consumption_model_narx))
mape_consumption_model_2_narx <- mape(strength_pred_2, test_data_2$G_pred)
print(paste0("MAPE: ", mape_consumption_model_2_narx))
mape_consumption_model_3_narx <- mape(strength_pred_3, test_data_2$G_pred)
print(paste0("MAPE: ", mape_consumption_model_3_narx))
mape_consumption_model_4_narx <- mape(strength_pred_4, test_data_2$G_pred)
print(paste0("MAPE: ", mape_consumption_model_4_narx))
mape_consumption_model_5_narx <- mape(strength_pred_5, test_data_2$G_pred)
print(paste0("MAPE: ", mape_consumption_model_5_narx))
mape_consumption_model_6_narx <- mape(strength_pred_6, test_data_2$G_pred)
print(paste0("MAPE: ", mape_consumption_model_6_narx))
mape_consumption_model_7_narx <- mape(strength_pred_7, test_data_2$G_pred)
print(paste0("MAPE: ", mape_consumption_model_7_narx))
mape_consumption_model_8_narx <- mape(strength_pred_8, test_data_2$G_pred)
print(paste0("MAPE: ", mape_consumption_model_8_narx))
mape_consumption_model_9_narx <- mape(strength_pred_9, test_data_2$G_pred)
print(paste0("MAPE: ", mape_consumption_model_9_narx))
mape_consumption_model_10_narx <- mape(strength_pred_10, test_data_2$G_pred)
print(paste0("MAPE: ", mape_consumption_model_10_narx))
mape_consumption_model_11_narx <- mape(strength_pred_11, test_data_2$G_pred)
print(paste0("MAPE: ", mape_consumption_model_11_narx))
mape_consumption_model_12_narx <- mape(strength_pred_12, test_data_2$G_pred)
print(paste0("MAPE: ", mape_consumption_model_12_narx))
```

```r
smape <- function(actual, predicted) {
  mean(2 * abs(predicted - actual) / (abs(actual) + abs(predicted)))
}


smape_consumption_model <- smape(test_data_2$G_pred, strength_pred)
print(paste0("sMAPE: ", smape_consumption_model))
smape_consumption_model_2 <- smape(test_data_2$G_pred, strength_pred_2)
print(paste0("sMAPE: ", smape_consumption_model_2))
smape_consumption_model_3 <- smape(test_data_2$G_pred, strength_pred_3)
print(paste0("sMAPE: ", smape_consumption_model_3))
smape_consumption_model_4 <- smape(test_data_2$G_pred, strength_pred_4)
print(paste0("sMAPE: ", smape_consumption_model_4))
smape_consumption_model_5 <- smape(test_data_2$G_pred, strength_pred_5)
print(paste0("sMAPE: ", smape_consumption_model_5))
smape_consumption_model_6 <- smape(test_data_2$G_pred, strength_pred_6)
print(paste0("sMAPE: ", smape_consumption_model_6))
smape_consumption_model_7 <- smape(test_data_2$G_pred, strength_pred_7)
print(paste0("sMAPE: ", smape_consumption_model_7))
smape_consumption_model_8 <- smape(test_data_2$G_pred, strength_pred_8)
print(paste0("sMAPE: ", smape_consumption_model_8))
smape_consumption_model_9 <- smape(test_data_2$G_pred, strength_pred_9)
print(paste0("sMAPE: ", smape_consumption_model_9))
smape_consumption_model_10 <- smape(test_data_2$G_pred, strength_pred_10)
print(paste0("sMAPE: ", smape_consumption_model_10))
smape_consumption_model_11 <- smape(test_data_2$G_pred, strength_pred_11)
print(paste0("sMAPE: ", smape_consumption_model_11))
smape_consumption_model_12 <- smape(test_data_2$G_pred, strength_pred_12)
print(paste0("sMAPE: ", smape_consumption_model_12))
```

```r
results_df_2 <- data.frame(
  Model = c(
    "Consumption Model",
    "Consumption Model 2",
    "Consumption Model 3",
    "Consumption Model 4",
    "Consumption Model 5",
    "Consumption Model 6",
    "Consumption Model 7",
    "Consumption Model 8",
    "Consumption Model 9",
    "Consumption Model 10",
    "Consumption Model 11",
    "Consumption Model 12"
  ),
  RMSE = c(
    rmse_narx,
    rmse_2_narx,
    rmse_3_narx,
    rmse_4_narx,
    rmse_5_narx,
    rmse_6_narx,
    rmse_7_narx,
    rmse_8_narx,
    rmse_9_narx,
    rmse_10_narx,
    rmse_11_narx,
    rmse_12_narx
  ),
  MAE = c(
    mae_consumption_model_narx,
    mae_consumption_model_2_narx,
    mae_consumption_model_3_narx,
    mae_consumption_model_4_narx,
    mae_consumption_model_5_narx,
    mae_consumption_model_6_narx,
    mae_consumption_model_7_narx,
    mae_consumption_model_8_narx,
    mae_consumption_model_9_narx,
    mae_consumption_model_10_narx,
    mae_consumption_model_11_narx,
    mae_consumption_model_12_narx
  ),
```

```r
  MAPE = c(
    mape_consumption_model_narx,
    mape_consumption_model_2_narx,
    mape_consumption_model_3_narx,
    mape_consumption_model_4_narx,
    mape_consumption_model_5_narx,
    mape_consumption_model_6_narx,
    mape_consumption_model_7_narx,
    mape_consumption_model_8_narx,
    mape_consumption_model_9_narx,
    mape_consumption_model_10_narx,
    mape_consumption_model_11_narx,
    mape_consumption_model_12_narx
  ),
  sMAPE = c(
    smape_consumption_model_narx,
    smape_consumption_model_2_narx,
    smape_consumption_model_3_narx,
    smape_consumption_model_4_narx,
    smape_consumption_model_5_narx,
    smape_consumption_model_6_narx,
    smape_consumption_model_7_narx,
    smape_consumption_model_8_narx,
    smape_consumption_model_9_narx,
    smape_consumption_model_10_narx,
    smape_consumption_model_11_narx,
    smape_consumption_model_12_narx
  ),
  Parameters = NA
)
```

```r
# calculate the number of parameters for each model and update the data frame
for (i in 1:nrow(results_df_2)) {
  model_formula_2 <- as.formula(paste("G_pred ~", paste0(colnames(training_data_2)[-1], collapse = " + ")))
  if (grepl("Model 1$", results_df_2$Model[i])) {
    nn <- neuralnet(model_formula_2, data = training_data_2, hidden = 3)
  } else if (grepl("Model 2$", results_df_2$Model[i])) {
    nn <- neuralnet(model_formula_2, data = training_data_2, hidden = c(5, 2))
  } else {
    nn <- neuralnet(model_formula_2, data = training_data_2, hidden = 3)
  }
  num_params <- sum(sapply(nn$weights, length))
  results_df_2$Parameters[i] <- num_params
}


# format data frame into table using kable

kable(results_df_2, caption = "Comparison of RMSE, MAE, MAPE, sMAPE, and number of parameters for twelve neural network models")
```

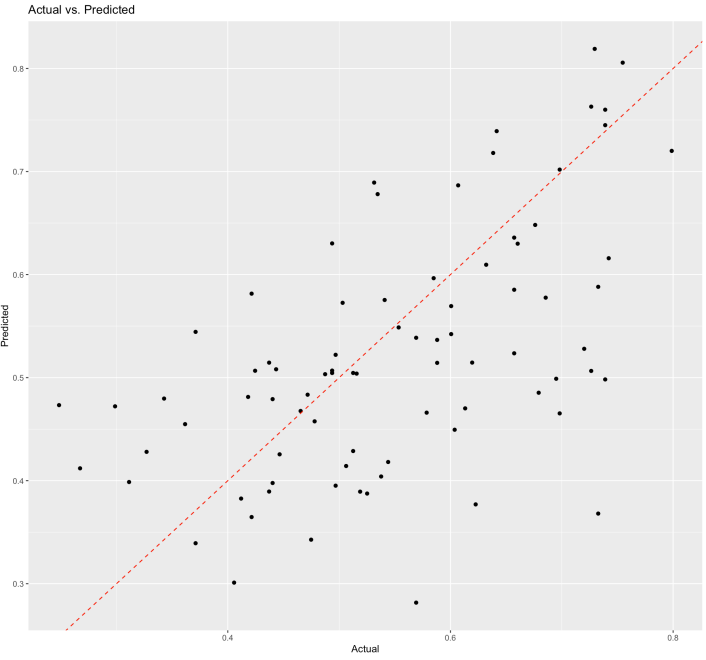
Table: Comparison of RMSE, MAE, MAPE, sMAPE, and number of parameters for twelve neural network models

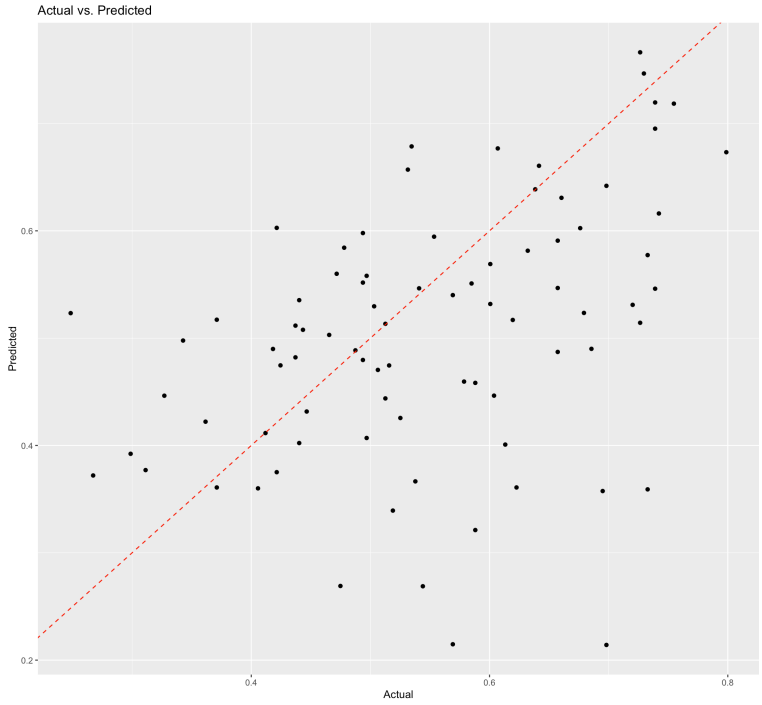
Model	RMSE	MAE	MAPE	sMAPE	Parameters
Consumption Model	39.54073	39.37361	0.9863675	1.946408	2
Consumption Model 2	39.39705	39.22699	0.9863675	1.946220	3
Consumption Model 3	39.51904	39.35139	0.9864085	1.946380	2
Consumption Model 4	39.38866	39.21573	0.9863626	1.946201	2
Consumption Model 5	39.47803	39.30585	0.9863948	1.946327	2
Consumption Model 6	39.41235	39.23386	0.9863704	1.946232	2
Consumption Model 7	39.41603	39.23621	0.9863712	1.946234	2
Consumption Model 8	39.47969	39.29776	0.9863920	1.946316	2
Consumption Model 9	39.42827	39.25261	0.9863763	1.946255	2
Consumption Model 10	39.54072	39.37085	0.9864148	1.946405	2
Consumption Model 11	39.45709	39.28212	0.9863866	1.946295	2
Consumption Model 12	39.49755	39.32200	0.9863999	1.946346	2

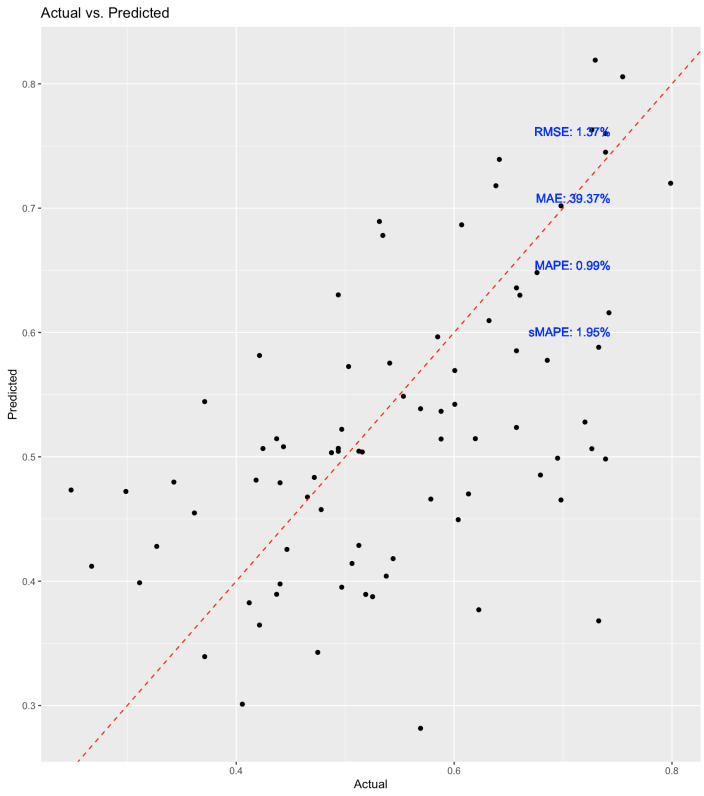
Based on the results above, I repeated the same from the AR approach but this time I added the 18:00 and 19:00 as exogenous variables and binded them in the column with the lagged variables and did 12 models, calculated RMSE, MAE, MAPE and sMAPE and did the comparison table to show the performance of each model.

I. WHICH ONE IS BETTER(AR vs. NARX):

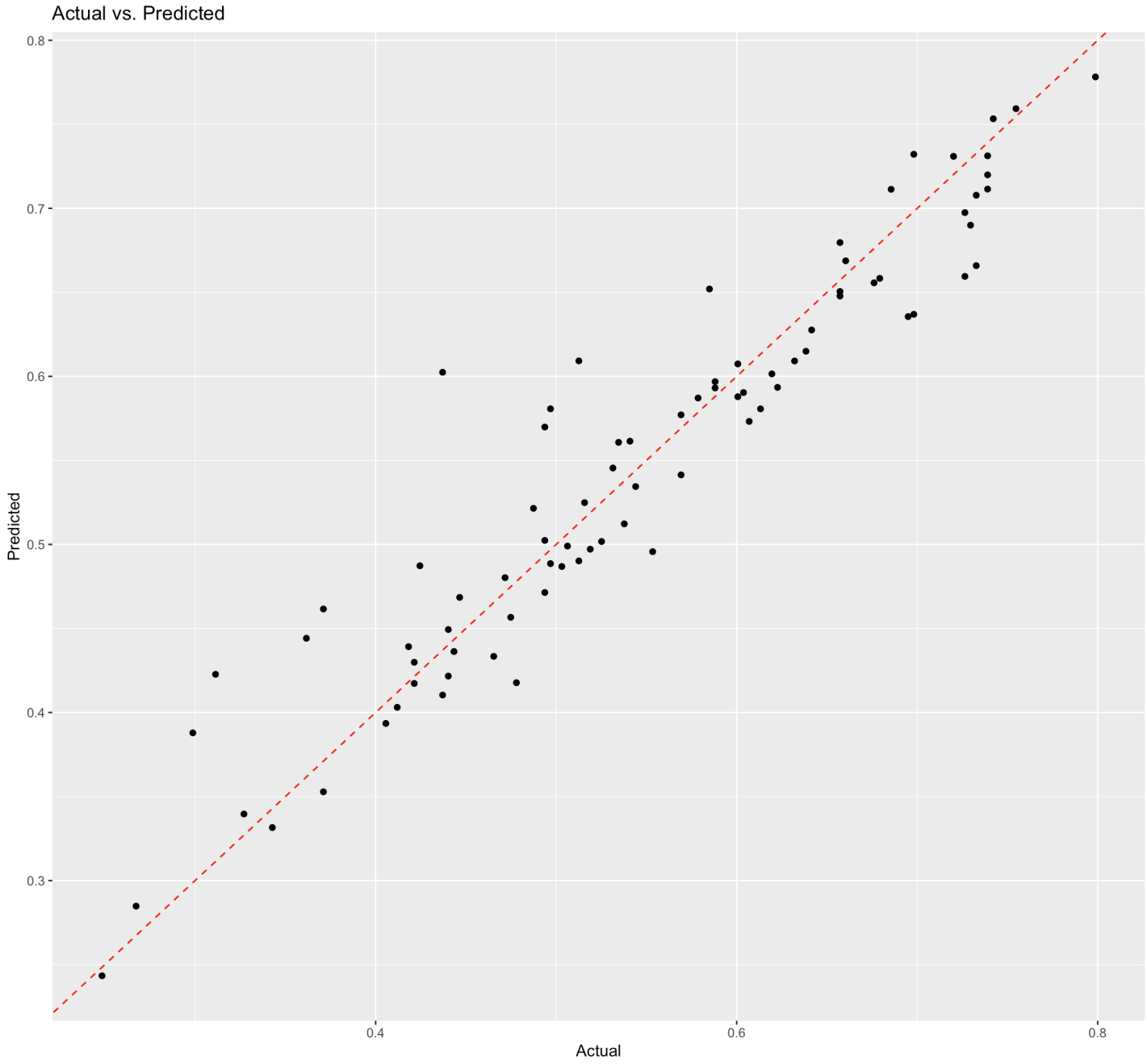
AR APPROACH OF TWO MODELS WITH A STATISTICAL INDEX MODEL FOR COMPARISON:

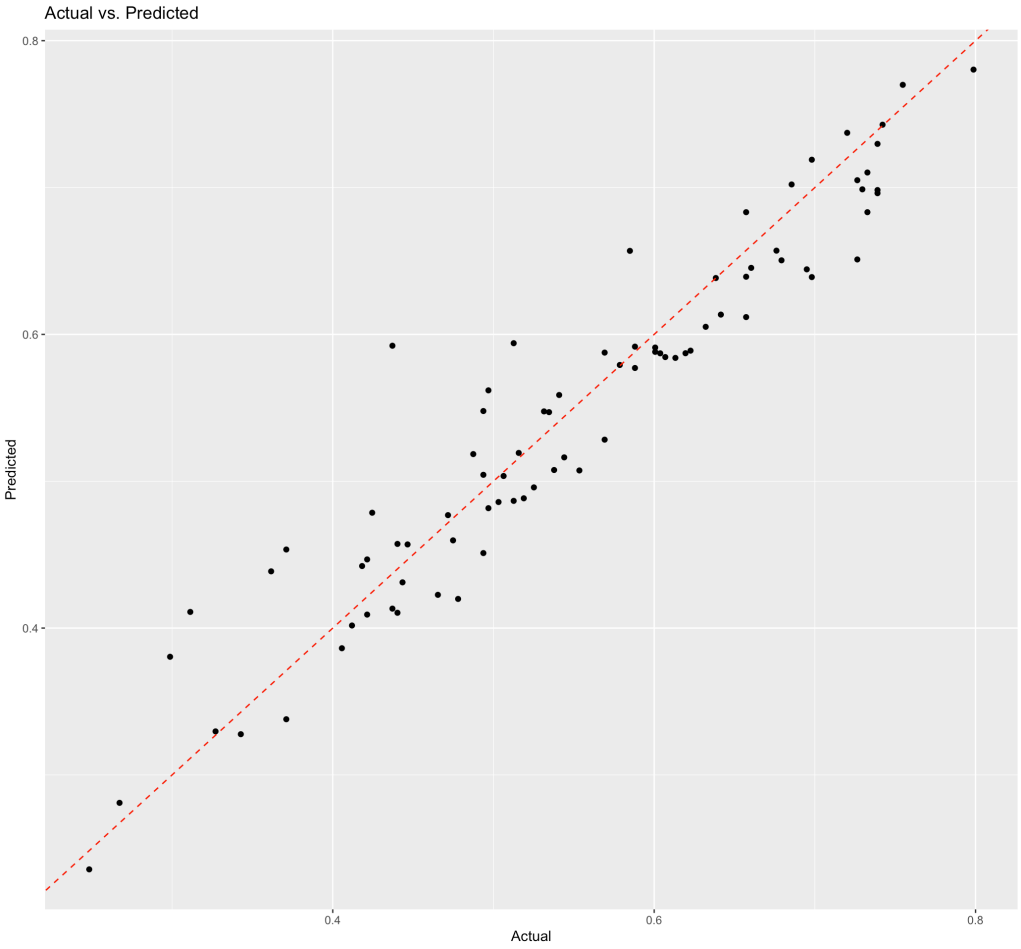


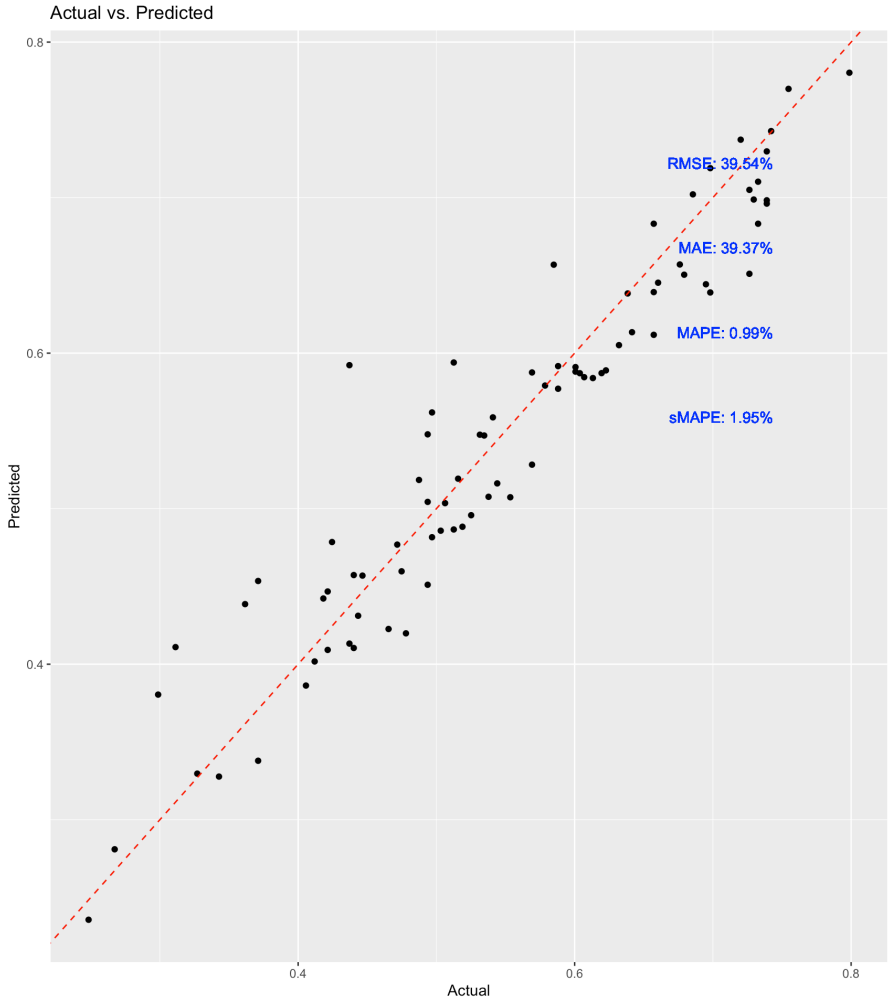




NARX APPROACH OF TWO MODELS WITH A STATISTICAL INDEX MODEL FOR COMPARISON:




Actual vs. Predicted
0.8
0.6
0.4
Predicted
0.4
0.6
0.8
Actual


Actual vs. Predicted
0.8
0.6
0.4
Predicted
RMSE: 39.54%
MAE: 39.37%
MAPE: 0.99%
sMAPE: 1.95%
0.4
0.6
0.8
Actual

Based on the graph results above, the NARX approach is way more efficient than AR because in AR, the results are far away and I can predict that it won't give me accurate predicted results whereas in NARX, the results are closer to the line and I can predict that it will give me accurate predicted results.

APPENDIX

```
library(dplyr)
library(readxl)
library(NbClust)
library(factoextra)
library(cluster)

vehicles <- read_excel("/Users/fareedkhan/Documents/vehicles.xlsx")
head(vehicles)
summary(vehicles)
vehicles_updated <- subset(vehicles, select = -c(Samples,Class))

head(vehicles_updated)
summary(vehicles_updated)
View(vehicles_updated)

data <- data.frame(vehicles_updated)
data

boxplot(data)

outliers <- boxplot(data)$out
length(outliers)


vehicles_2 <- read_excel("Documents/vehicles_2.xlsx")
vehicles_updated_2 <- subset(vehicles_removed, select = -c(Samples,Class))
View(vehicles_updated_2)

boxplot(vehicles_updated_2)

outliers_2 <- boxplot(vehicles_updated_2)$out
length(outliers_2)
```

```r
set.seed(10)
nc <- NbClust(vehicles_updated_2,
              distance = "euclidean",
              min.nc=2, max.nc=15,
              method="kmeans")

fviz_nbclust(vehicles_updated_2, kmeans, method = 'wss')
fviz_nbclust(vehicles_updated_2, kmeans, method = 'silhouette')
fviz_nbclust(vehicles_updated_2, kmeans, method = 'gap_stat')

k=3

kmeans_vehicles <- kmeans(vehicles_updated_2, centers = k, nstart = 10)
fviz_cluster(kmeans_vehicles, data = vehicles_updated_2)
fit.km <- kmeans(vehicles_updated_2, 3)
fit.km
wss = fit.km$tot.withinss
bss = fit.km$betweenss
wss
bss

sil <- silhouette(kmeans_vehicles$cluster, dist(vehicles_updated_2))
fviz_silhouette(sil)

scaled_data <- scale(vehicles_updated_2)
boxplot(scaled_data)



str(vehicles_updated_2)
summary(scaled_data)
```

```r
pca_result <- prcomp(scaled_data, scale = FALSE)
pca_result
names(pca_result)
pca_result$x <- - pca_result$x
head(pca_result$x)
pca_result$sdev
pca_vehicles <- as.data.frame(pca_result$x[,1:7])
pca_vehicles
summary(pca_result)
eigenvectors <- pca_result$rotation
eigenvectors

set.seed(10)
nc <- NbClust(pca_vehicles,
              distance = "euclidean",
              min.nc=2, max.nc=15,
              method="kmeans")
fviz_nbclust(pca_vehicles, kmeans, method = 'wss')
fviz_nbclust(pca_vehicles, kmeans, method = 'silhouette')
fviz_nbclust(pca_vehicles, kmeans, method = 'gap_stat')
k = 2
kmeans_vehicles_2 <- kmeans(pca_vehicles, centers = k, nstart = 10)
fviz_cluster(kmeans_vehicles_2, data = pca_vehicles)
fit.km <- kmeans(pca_vehicles, 2)
fit.km
wss = fit.km$tot.withinss
bss = fit.km$betweenss
wss
bss

sil_2 <- silhouette(kmeans_vehicles_2$cluster, dist(pca_vehicles))
fviz_silhouette(sil_2)
```

```r
# PCA analysis
pca_vehiclesNew <- prcomp(vehicles_updated_2, center = TRUE, scale = TRUE)

# Extract first 7 principal components
vehiclesNew_updated <- as.data.frame(-pca_vehiclesNew$x[,1:7])

# K-means clustering
k = 3
kmeans_vehiclesNew_updated <- kmeans(vehiclesNew_updated, centers = k, nstart = 10)

# Calinski-Harabasz Index
ch_index <- cluster::clusplot(vehiclesNew_updated, kmeans_vehiclesNew_updated$cluster, color = TRUE, shade = TRUE, labels = 2, lines = 0)


ch_index_1 <- NbClust(vehiclesNew_updated,  distance="euclidean", min.nc=3, max.nc=10, method="kmeans", index="ch")
ch_index_1
plot(ch_index_1$All.index, type="b", xlab="Number of Clusters", ylab="Calinski-Harabasz Index")
par(mar=c(3,3,2,1))
plot(ch_index_1$All.index, type="b", xlab="Number of Clusters", ylab="Calinski-Harabasz Index")
```

```r
library(readxl)
library(neuralnet)
library(dplyr)
library(caret)
library(Metrics)
library(ggplot2)


# read in the dataset
uow_consumption <- read_excel("Downloads/uow_consumption.xlsx")
str(uow_consumption)
summary(uow_consumption)
colnames(uow_consumption)[colnames(uow_consumption)=="0.75"] <- "18:00"
colnames(uow_consumption)[colnames(uow_consumption)=="0.79166666666666663"] <- "19:00"
colnames(uow_consumption)[colnames(uow_consumption)=="0.83333333333333337"] <- "20:00"
summary(uow_consumption$"20:00")
uow_consumption_updated <- subset(uow_consumption, select = -c(date))

#AR approach

uow_consumption_new <- c(uow_consumption_updated$"20:00") #accessing the 20:00 column

input_data_ts <- bind_cols(
  G_prev7 = lag(uow_consumption_new, 8),
  G_prev4 = lag(uow_consumption_new, 5),
  G_prev3 = lag(uow_consumption_new, 4),
  G_prev2 = lag(uow_consumption_new, 3),
  G_prev1 = lag(uow_consumption_new, 2),
  G_current = lag(uow_consumption_new, 1),
  G_pred = uow_consumption_new)

input_data_ts

input_data_ts <- input_data_ts[complete.cases(input_data_ts),]

head(input_data_ts)

str(input_data_ts)


normalize <- function(x) {
  return((x - min(x)) / (max(x) - min(x)))
}

norm <- as.data.frame(lapply(input_data_ts, normalize))
summary(norm)
```

```r
training_data <- norm[1:50,]
test_data <- norm[51:100,]
colnames(training_data)
colnames(test_data)

consumption_model <- neuralnet("G_pred ~ G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = 10, data = training_data, linear.output=TRUE)

consumption_model_2 <- neuralnet("G_pred ~ G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = 12, data = training_data, linear.output=TRUE)

consumption_model_3 <- neuralnet("G_pred ~ G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(4,5), data = training_data, linear.output=TRUE)

consumption_model_4 <- neuralnet("G_pred ~ G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(1,2), data = training_data, linear.output=TRUE)

consumption_model_5 <- neuralnet("G_pred ~ G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(6,7), data = training_data, linear.output=TRUE)

consumption_model_6 <- neuralnet("G_pred ~ G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(9,10), data = training_data, linear.output=TRUE)

consumption_model_7 <- neuralnet("G_pred ~ G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(2,3), data = training_data, linear.output=TRUE)

consumption_model_8 <- neuralnet("G_pred ~ G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(5,6), data = training_data, linear.output=TRUE)

consumption_model_9 <- neuralnet("G_pred ~ G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(15,16), data = training_data, linear.output=TRUE)

consumption_model_10 <- neuralnet("G_pred ~ G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = 9, data = training_data, linear.output=TRUE)

consumption_model_11 <- neuralnet("G_pred ~ G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(12,13), data = training_data, linear.output=TRUE)

consumption_model_12 <- neuralnet("G_pred ~ G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(13,14), data = training_data, linear.output=TRUE)

model_results <- predict(consumption_model, test_data[, c("G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_2 <- predict(consumption_model_2, test_data[, c("G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_3 <- predict(consumption_model_3, test_data[, c("G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_4 <- predict(consumption_model_4, test_data[, c("G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_5 <- predict(consumption_model_5, test_data[, c("G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_6 <- predict(consumption_model_6, test_data[, c("G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_7 <- predict(consumption_model_7, test_data[, c("G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_8 <- predict(consumption_model_8, test_data[, c("G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_9 <- predict(consumption_model_9, test_data[, c("G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_10 <- predict(consumption_model_10, test_data[, c("G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_11 <- predict(consumption_model_11, test_data[, c("G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_12 <- predict(consumption_model_12, test_data[, c("G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
```

```r
predicted_strength <- model_results
predicted_strength_2 <- model_results_2
predicted_strength_3 <- model_results_3
predicted_strength_4 <- model_results_4
predicted_strength_5 <- model_results_5
predicted_strength_6 <- model_results_6
predicted_strength_7 <- model_results_7
predicted_strength_8 <- model_results_8
predicted_strength_9 <- model_results_9
predicted_strength_10 <- model_results_10
predicted_strength_11 <- model_results_11
predicted_strength_12 <- model_results_12

print(predicted_strength)
print(predicted_strength_2)
print(predicted_strength_3)
print(predicted_strength_4)
print(predicted_strength_5)
print(predicted_strength_6)
print(predicted_strength_7)
print(predicted_strength_8)
print(predicted_strength_9)
print(predicted_strength_10)
print(predicted_strength_11)
print(predicted_strength_12)

consumption_train_original_strength <- input_data_ts[1:50,"G_pred"]
consumption_test_original_strength <- input_data_ts[51:60,"G_pred"]

strength_min <- min(consumption_train_original_strength)
strength_max <- max(consumption_train_original_strength)

print(strength_min)
print(strength_max)

head(consumption_test_original_strength)
print(consumption_test_original_strength)

unnormalize <- function(x, min, max) {
  return( (max - min)*x + min )
}
```

```
strength_pred <- unnormalize(predicted_strength, strength_min, strength_max)
head(strength_pred)
strength_pred_2 <- unnormalize(predicted_strength_2, strength_min,strength_max)
head(strength_pred_2)
strength_pred_3 <- unnormalize(predicted_strength_3, strength_min,strength_max)
head(strength_pred_3)
strength_pred_4 <- unnormalize(predicted_strength_4, strength_min,strength_max)
head(strength_pred_4)
strength_pred_5 <- unnormalize(predicted_strength_5, strength_min,strength_max)
head(strength_pred_5)
strength_pred_6 <- unnormalize(predicted_strength_6, strength_min,strength_max)
head(strength_pred_6)
strength_pred_7 <- unnormalize(predicted_strength_7, strength_min,strength_max)
head(strength_pred_7)
strength_pred_8 <- unnormalize(predicted_strength_8, strength_min,strength_max)
head(strength_pred_8)
strength_pred_9 <- unnormalize(predicted_strength_9, strength_min,strength_max)
head(strength_pred_9)
strength_pred_10 <- unnormalize(predicted_strength_10, strength_min,strength_max)
head(strength_pred_10)
strength_pred_11 <- unnormalize(predicted_strength_11, strength_min,strength_max)
head(strength_pred_11)
strength_pred_12 <- unnormalize(predicted_strength_12, strength_min,strength_max)
head(strength_pred_12)
```

```
rmse <- RMSE(strength_pred, test_data$G_pred)
print(paste0("RMSE: ", rmse))
rmse_2 <- RMSE(strength_pred_2, test_data$G_pred)
print(paste0("RMSE: ", rmse_2))
rmse_3 <- RMSE(strength_pred_3, test_data$G_pred)
print(paste0("RMSE: ", rmse_3))
rmse_4 <- RMSE(strength_pred_4, test_data$G_pred)
print(paste0("RMSE: ", rmse_4))
rmse_5 <- RMSE(strength_pred_5, test_data$G_pred)
print(paste0("RMSE: ", rmse_5))
rmse_6 <- RMSE(strength_pred_6, test_data$G_pred)
print(paste0("RMSE: ", rmse_6))
rmse_7 <- RMSE(strength_pred_7, test_data$G_pred)
print(paste0("RMSE: ", rmse_7))
rmse_8 <- RMSE(strength_pred_8, test_data$G_pred)
print(paste0("RMSE: ", rmse_8))
rmse_9 <- RMSE(strength_pred_9, test_data$G_pred)
print(paste0("RMSE: ", rmse_9))
rmse_10 <- RMSE(strength_pred_10, test_data$G_pred)
print(paste0("RMSE: ", rmse_10))
rmse_11 <- RMSE(strength_pred_11, test_data$G_pred)
print(paste0("RMSE: ", rmse_11))
rmse_12 <- RMSE(strength_pred_12, test_data$G_pred)
print(paste0("RMSE: ", rmse_12))
```

```r
mae_consumption_model <- MAE(strength_pred, test_data$G_pred)
print(paste0("MAE: ", mae_consumption_model))
mae_consumption_model_2 <- MAE(strength_pred_2, test_data$G_pred)
print(paste0("MAE: ", mae_consumption_model_2))
mae_consumption_model_3 <- MAE(strength_pred_3, test_data$G_pred)
print(paste0("MAE: ", mae_consumption_model_3))
mae_consumption_model_4 <- MAE(strength_pred_4, test_data$G_pred)
print(paste0("MAE: ", mae_consumption_model_4))
mae_consumption_model_5 <- MAE(strength_pred_5, test_data$G_pred)
print(paste0("MAE: ", mae_consumption_model_5))
mae_consumption_model_6 <- MAE(strength_pred_6, test_data$G_pred)
print(paste0("MAE: ", mae_consumption_model_6))
mae_consumption_model_7 <- MAE(strength_pred_7, test_data$G_pred)
print(paste0("MAE: ", mae_consumption_model_7))
mae_consumption_model_8 <- MAE(strength_pred_8, test_data$G_pred)
print(paste0("MAE: ", mae_consumption_model_8))
mae_consumption_model_9 <- MAE(strength_pred_9, test_data$G_pred)
print(paste0("MAE: ", mae_consumption_model_9))
mae_consumption_model_10 <- MAE(strength_pred_10, test_data$G_pred)
print(paste0("MAE: ", mae_consumption_model_10))
mae_consumption_model_11 <- MAE(strength_pred_11, test_data$G_pred)
print(paste0("MAE: ", mae_consumption_model_11))
mae_consumption_model_12 <- MAE(strength_pred_12, test_data$G_pred)
print(paste0("MAE: ", mae_consumption_model_12))
```

```r
mape_consumption_model <- mape(strength_pred_2, test_data$G_pred)
print(paste0("MAPE: ", mape_consumption_model))
mape_consumption_model_2 <- mape(strength_pred_2, test_data$G_pred)
print(paste0("MAPE: ", mape_consumption_model_2))
mape_consumption_model_3 <- mape(strength_pred_3, test_data$G_pred)
print(paste0("MAPE: ", mape_consumption_model_3))
mape_consumption_model_4 <- mape(strength_pred_4, test_data$G_pred)
print(paste0("MAPE: ", mape_consumption_model_4))
mape_consumption_model_5 <- mape(strength_pred_5, test_data$G_pred)
print(paste0("MAPE: ", mape_consumption_model_5))
mape_consumption_model_6 <- mape(strength_pred_6, test_data$G_pred)
print(paste0("MAPE: ", mape_consumption_model_6))
mape_consumption_model_7 <- mape(strength_pred_7, test_data$G_pred)
print(paste0("MAPE: ", mape_consumption_model_7))
mape_consumption_model_8 <- mape(strength_pred_8, test_data$G_pred)
print(paste0("MAPE: ", mape_consumption_model_8))
mape_consumption_model_9 <- mape(strength_pred_9, test_data$G_pred)
print(paste0("MAPE: ", mape_consumption_model_9))
mape_consumption_model_10 <- mape(strength_pred_10, test_data$G_pred)
print(paste0("MAPE: ", mape_consumption_model_10))
mape_consumption_model_11 <- mape(strength_pred_11, test_data$G_pred)
print(paste0("MAPE: ", mape_consumption_model_11))
mape_consumption_model_12 <- mape(strength_pred_12, test_data$G_pred)
print(paste0("MAPE: ", mape_consumption_model_12))
```

```r
smape <- function(actual, predicted) {
  mean(2 * abs(predicted - actual) / (abs(actual) + abs(predicted)))
}


smape_consumption_model <- smape(test_data$G_pred, strength_pred)
print(paste0("sMAPE: ", smape_consumption_model))
smape_consumption_model_2 <- smape(test_data$G_pred, strength_pred_2)
print(paste0("sMAPE: ", smape_consumption_model_2))
smape_consumption_model_3 <- smape(test_data$G_pred, strength_pred_3)
print(paste0("sMAPE: ", smape_consumption_model_3))
smape_consumption_model_4 <- smape(test_data$G_pred, strength_pred_4)
print(paste0("sMAPE: ", smape_consumption_model_4))
smape_consumption_model_5 <- smape(test_data$G_pred, strength_pred_5)
print(paste0("sMAPE: ", smape_consumption_model_5))
smape_consumption_model_6 <- smape(test_data$G_pred, strength_pred_6)
print(paste0("sMAPE: ", smape_consumption_model_6))
smape_consumption_model_7 <- smape(test_data$G_pred, strength_pred_7)
print(paste0("sMAPE: ", smape_consumption_model_7))
smape_consumption_model_8 <- smape(test_data$G_pred, strength_pred_8)
print(paste0("sMAPE: ", smape_consumption_model_8))
smape_consumption_model_9 <- smape(test_data$G_pred, strength_pred_9)
print(paste0("sMAPE: ", smape_consumption_model_9))
smape_consumption_model_10 <- smape(test_data$G_pred, strength_pred_10)
print(paste0("sMAPE: ", smape_consumption_model_10))
smape_consumption_model_11 <- smape(test_data$G_pred, strength_pred_11)
print(paste0("sMAPE: ", smape_consumption_model_11))
smape_consumption_model_12 <- smape(test_data$G_pred, strength_pred_12)
print(paste0("sMAPE: ", smape_consumption_model_12))
```

```r
results_df <- data.frame(
  Model = c(
    "Consumption Model",
    "Consumption Model 2",
    "Consumption Model 3",
    "Consumption Model 4",
    "Consumption Model 5",
    "Consumption Model 6",
    "Consumption Model 7",
    "Consumption Model 8",
    "Consumption Model 9",
    "Consumption Model 10",
    "Consumption Model 11",
    "Consumption Model 12"
  ),
  RMSE = c(
    rmse,
    rmse_2,
    rmse_3,
    rmse_4,
    rmse_5,
    rmse_6,
    rmse_7,
    rmse_8,
    rmse_9,
    rmse_10,
    rmse_11,
    rmse_12
  ),
  MAE = c(
    mae_consumption_model,
    mae_consumption_model_2,
    mae_consumption_model_3,
    mae_consumption_model_4,
    mae_consumption_model_5,
    mae_consumption_model_6,
    mae_consumption_model_7,
    mae_consumption_model_8,
    mae_consumption_model_9,
    mae_consumption_model_10,
    mae_consumption_model_11,
    mae_consumption_model_12
  ),
```

```r
  MAPE = c(
    mape_consumption_model,
    mape_consumption_model_2,
    mape_consumption_model_3,
    mape_consumption_model_4,
    mape_consumption_model_5,
    mape_consumption_model_6,
    mape_consumption_model_7,
    mape_consumption_model_8,
    mape_consumption_model_9,
    mape_consumption_model_10,
    mape_consumption_model_11,
    mape_consumption_model_12
  ),
  sMAPE = c(
    smape_consumption_model,
    smape_consumption_model_2,
    smape_consumption_model_3,
    smape_consumption_model_4,
    smape_consumption_model_5,
    smape_consumption_model_6,
    smape_consumption_model_7,
    smape_consumption_model_8,
    smape_consumption_model_9,
    smape_consumption_model_10,
    smape_consumption_model_11,
    smape_consumption_model_12
  ),
  Parameters = NA
)
```

```r
# calculate the number of parameters for each model and update the data frame
for (i in 1:nrow(results_df)) {
  model_formula <- as.formula(paste("G_pred ~", paste0(colnames(training_data)[-1], collapse = " + ")))
  if (grepl("Model 1$", results_df$Model[i])) {
    nn <- neuralnet(model_formula, data = training_data, hidden = 1)
  } else if (grepl("Model 2$", results_df$Model[i])) {
    nn <- neuralnet(model_formula, data = training_data, hidden = c(5, 2))
  } else {
    nn <- neuralnet(model_formula, data = training_data, hidden = 1)
  }
  num_params <- sum(sapply(nn$weights, length))
  results_df$Parameters[i] <- num_params
}


# # format data frame into table using kable
library(knitr)
kable(results_df, caption = "Comparison of RMSE, MAE, MAPE, sMAPE, and number of parameters for twelve neural network models")
```

```r
#NARX approach

uow_consumption_new_2 <- c(uow_consumption_updated$"18:00") #accessing the 18:00 column
uow_consumption_new_3 <- c(uow_consumption_updated$"19:00") #accessing the 19:00 column


# create exogenous variables
exog_vars <- bind_cols(uow_consumption_new_2, uow_consumption_new_3)

# rename columns
exog_vars <- exog_vars %>% rename(exog_var_1 = ...1, exog_var_2 = ...2)

summary(exog_vars$exog_var_1)
summary(exog_vars$exog_var_2)
# combine lagged variables, exogenous variables, and target variable
input_data_ts_2 <- bind_cols(
  exog_vars,
  G_prev7 = lag(uow_consumption_new, 8),
  G_prev4 = lag(uow_consumption_new, 5),
  G_prev3 = lag(uow_consumption_new, 4),
  G_prev2 = lag(uow_consumption_new, 3),
  G_prev1 = lag(uow_consumption_new, 2),
  G_current = lag(uow_consumption_new, 1),
  G_pred = uow_consumption_new
)

input_data_ts_2

# create a logical vector indicating which rows of input_data_ts_2 contain complete cases
complete_rows <- complete.cases(input_data_ts_2)

# subset input_data_ts based on the row indices of complete cases in input_data_ts_2
input_data_ts_2 <- input_data_ts_2[complete_rows, ]
```

```r
head(input_data_ts_2)



normalize <- function(x) {
  return((x - min(x)) / (max(x) - min(x)))
}

norm_2 <- as.data.frame(lapply(input_data_ts_2, normalize))
summary(norm_2)

training_data_2 <- norm_2[1:380,]
test_data_2 <- norm_2[381:462,]

colnames(training_data_2)
colnames(test_data_2)
names(training_data_2)

consumption_model_narx <- neuralnet("G_pred ~ exog_var_1 + exog_var_2 + G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = 10, data = training_data_2, linear.output=TRUE)

consumption_model_2_narx <- neuralnet("G_pred ~ exog_var_1 + exog_var_2 + G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = 12, data = training_data_2, linear.output=TRUE)

consumption_model_3_narx <- neuralnet("G_pred ~ exog_var_1 + exog_var_2 + G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(4,5), data = training_data_2, linear.output=TRUE)

consumption_model_4_narx <- neuralnet("G_pred ~ exog_var_1 + exog_var_2 + G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(1,2), data = training_data_2, linear.output=TRUE)

consumption_model_5_narx <- neuralnet("G_pred ~ exog_var_1 + exog_var_2 + G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(6,7), data = training_data_2, linear.output=TRUE)

consumption_model_6_narx <- neuralnet("G_pred ~ exog_var_1 + exog_var_2 + G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(9,10), data = training_data_2, linear.output=TRUE)

consumption_model_7_narx <- neuralnet("G_pred ~ exog_var_1 + exog_var_2 + G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(2,3), data = training_data_2, linear.output=TRUE)

consumption_model_8_narx <- neuralnet("G_pred ~ exog_var_1 + exog_var_2 + G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(5,6), data = training_data_2, linear.output=TRUE)

consumption_model_9_narx <- neuralnet("G_pred ~ exog_var_1 + exog_var_2 + G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(15,16), data = training_data_2, linear.output=TRUE)

consumption_model_10_narx <- neuralnet("G_pred ~ exog_var_1 + exog_var_2 + G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = 9, data = training_data_2, linear.output=TRUE)

consumption_model_11_narx <- neuralnet("G_pred ~ exog_var_1 + exog_var_2 + G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(12,13), data = training_data_2, linear.output=TRUE)

consumption_model_12_narx <- neuralnet("G_pred ~ exog_var_1 + exog_var_2 + G_prev7 + G_prev4 + G_prev3 + G_prev2 + G_prev1 + G_current", hidden = c(13,14), data = training_data_2, linear.output=TRUE)
```

```r
model_results_narx <- predict(consumption_model_narx, test_data_2[, c("exog_var_1","exog_var_2","G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_2_narx <- predict(consumption_model_2_narx, test_data_2[, c("exog_var_1","exog_var_2","G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_3_narx <- predict(consumption_model_3_narx, test_data_2[, c("exog_var_1","exog_var_2","G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_4_narx <- predict(consumption_model_4_narx, test_data_2[, c("exog_var_1","exog_var_2","G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_5_narx <- predict(consumption_model_5_narx, test_data_2[, c("exog_var_1","exog_var_2","G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_6_narx <- predict(consumption_model_6_narx, test_data_2[, c("exog_var_1","exog_var_2","G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_7_narx <- predict(consumption_model_7_narx, test_data_2[, c("exog_var_1","exog_var_2","G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_8_narx <- predict(consumption_model_8_narx, test_data_2[, c("exog_var_1","exog_var_2","G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_9_narx <- predict(consumption_model_9_narx, test_data_2[, c("exog_var_1","exog_var_2","G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_10_narx <- predict(consumption_model_10_narx, test_data_2[, c("exog_var_1","exog_var_2","G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_11_narx <- predict(consumption_model_11_narx, test_data_2[, c("exog_var_1","exog_var_2","G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])
model_results_12_narx <- predict(consumption_model_12_narx, test_data_2[, c("exog_var_1","exog_var_2","G_prev7", "G_prev4", "G_prev3", "G_prev2", "G_prev1", "G_current")])


predicted_strength <- model_results_narx
predicted_strength_2 <- model_results_2_narx
predicted_strength_3 <- model_results_3_narx
predicted_strength_4 <- model_results_4_narx
predicted_strength_5 <- model_results_5_narx
predicted_strength_6 <- model_results_6_narx
predicted_strength_7 <- model_results_7_narx
predicted_strength_8 <- model_results_8_narx
predicted_strength_9 <- model_results_9_narx
predicted_strength_10 <- model_results_10_narx
predicted_strength_11 <- model_results_11_narx
predicted_strength_12 <- model_results_12_narx

consumption_train_original_strength <- input_data_ts_2[1:380,"G_pred"]
consumption_test_original_strength <- input_data_ts_2[381:462,"G_pred"]

strength_min <- min(consumption_train_original_strength)
strength_max <- max(consumption_train_original_strength)

print(strength_min)
print(strength_max)

head(consumption_test_original_strength)
print(consumption_test_original_strength)

unnormalize <- function(x, min, max) {
  return( (max - min)*x + min )
}
```

```
strength_pred <- unnormalize(predicted_strength, strength_min, strength_max)
head(strength_pred)
strength_pred_2 <- unnormalize(predicted_strength_2, strength_min,strength_max)
head(strength_pred_2)
strength_pred_3 <- unnormalize(predicted_strength_3, strength_min,strength_max)
head(strength_pred_3)
strength_pred_4 <- unnormalize(predicted_strength_4, strength_min,strength_max)
head(strength_pred_4)
strength_pred_5 <- unnormalize(predicted_strength_5, strength_min,strength_max)
head(strength_pred_5)
strength_pred_6 <- unnormalize(predicted_strength_6, strength_min,strength_max)
head(strength_pred_6)
strength_pred_7 <- unnormalize(predicted_strength_7, strength_min,strength_max)
head(strength_pred_7)
strength_pred_8 <- unnormalize(predicted_strength_8, strength_min,strength_max)
head(strength_pred_8)
strength_pred_9 <- unnormalize(predicted_strength_9, strength_min,strength_max)
head(strength_pred_9)
strength_pred_10 <- unnormalize(predicted_strength_10, strength_min,strength_max)
head(strength_pred_10)
strength_pred_11 <- unnormalize(predicted_strength_11, strength_min,strength_max)
head(strength_pred_11)
strength_pred_12 <- unnormalize(predicted_strength_12, strength_min,strength_max)
head(strength_pred_12)
```

```
rmse_narx <- RMSE(strength_pred, test_data_2$G_pred)
print(paste0("RMSE: ", rmse_narx))
rmse_2_narx <- RMSE(strength_pred_2, test_data_2$G_pred)
print(paste0("RMSE: ", rmse_2_narx))
rmse_3_narx <- RMSE(strength_pred_3, test_data_2$G_pred)
print(paste0("RMSE: ", rmse_3_narx))
rmse_4_narx <- RMSE(strength_pred_4, test_data_2$G_pred)
print(paste0("RMSE: ", rmse_4_narx))
rmse_5_narx <- RMSE(strength_pred_5, test_data_2$G_pred)
print(paste0("RMSE: ", rmse_5_narx))
rmse_6_narx <- RMSE(strength_pred_6, test_data_2$G_pred)
print(paste0("RMSE: ", rmse_6_narx))
rmse_7_narx <- RMSE(strength_pred_7, test_data_2$G_pred)
print(paste0("RMSE: ", rmse_7_narx))
rmse_8_narx <- RMSE(strength_pred_8, test_data_2$G_pred)
print(paste0("RMSE: ", rmse_8_narx))
rmse_9_narx <- RMSE(strength_pred_9, test_data_2$G_pred)
print(paste0("RMSE: ", rmse_9_narx))
rmse_10_narx <- RMSE(strength_pred_10, test_data_2$G_pred)
print(paste0("RMSE: ", rmse_10_narx))
rmse_11_narx <- RMSE(strength_pred_11, test_data_2$G_pred)
print(paste0("RMSE: ", rmse_11_narx))
rmse_12_narx <- RMSE(strength_pred_12, test_data_2$G_pred)
print(paste0("RMSE: ", rmse_12_narx))
```

```r
mae_consumption_model_narx <- MAE(strength_pred, test_data_2$G_pred)
print(paste0("MAE: ", mae_consumption_model_narx))
mae_consumption_model_2_narx <- MAE(strength_pred_2, test_data_2$G_pred)
print(paste0("MAE: ", mae_consumption_model_2_narx))
mae_consumption_model_3_narx <- MAE(strength_pred_3, test_data_2$G_pred)
print(paste0("MAE: ", mae_consumption_model_3_narx))
mae_consumption_model_4_narx <- MAE(strength_pred_4, test_data_2$G_pred)
print(paste0("MAE: ", mae_consumption_model_4_narx))
mae_consumption_model_5_narx <- MAE(strength_pred_5, test_data_2$G_pred)
print(paste0("MAE: ", mae_consumption_model_5_narx))
mae_consumption_model_6_narx <- MAE(strength_pred_6, test_data_2$G_pred)
print(paste0("MAE: ", mae_consumption_model_6_narx))
mae_consumption_model_7_narx <- MAE(strength_pred_7, test_data_2$G_pred)
print(paste0("MAE: ", mae_consumption_model_7_narx))
mae_consumption_model_8_narx <- MAE(strength_pred_8, test_data_2$G_pred)
print(paste0("MAE: ", mae_consumption_model_8_narx))
mae_consumption_model_9_narx <- MAE(strength_pred_9, test_data_2$G_pred)
print(paste0("MAE: ", mae_consumption_model_9_narx))
mae_consumption_model_10_narx <- MAE(strength_pred_10, test_data_2$G_pred)
print(paste0("MAE: ", mae_consumption_model_10_narx))
mae_consumption_model_11_narx <- MAE(strength_pred_11, test_data_2$G_pred)
print(paste0("MAE: ", mae_consumption_model_11_narx))
mae_consumption_model_12_narx <- MAE(strength_pred_12, test_data_2$G_pred)
print(paste0("MAE: ", mae_consumption_model_12_narx))
```

```r
mape_consumption_model_narx <- mape(strength_pred_2, test_data_2$G_pred)
print(paste0("MAPE: ", mape_consumption_model_narx))
mape_consumption_model_2_narx <- mape(strength_pred_2, test_data_2$G_pred)
print(paste0("MAPE: ", mape_consumption_model_2_narx))
mape_consumption_model_3_narx <- mape(strength_pred_3, test_data_2$G_pred)
print(paste0("MAPE: ", mape_consumption_model_3_narx))
mape_consumption_model_4_narx <- mape(strength_pred_4, test_data_2$G_pred)
print(paste0("MAPE: ", mape_consumption_model_4_narx))
mape_consumption_model_5_narx <- mape(strength_pred_5, test_data_2$G_pred)
print(paste0("MAPE: ", mape_consumption_model_5_narx))
mape_consumption_model_6_narx <- mape(strength_pred_6, test_data_2$G_pred)
print(paste0("MAPE: ", mape_consumption_model_6_narx))
mape_consumption_model_7_narx <- mape(strength_pred_7, test_data_2$G_pred)
print(paste0("MAPE: ", mape_consumption_model_7_narx))
mape_consumption_model_8_narx <- mape(strength_pred_8, test_data_2$G_pred)
print(paste0("MAPE: ", mape_consumption_model_8_narx))
mape_consumption_model_9_narx <- mape(strength_pred_9, test_data_2$G_pred)
print(paste0("MAPE: ", mape_consumption_model_9_narx))
mape_consumption_model_10_narx <- mape(strength_pred_10, test_data_2$G_pred)
print(paste0("MAPE: ", mape_consumption_model_10_narx))
mape_consumption_model_11_narx <- mape(strength_pred_11, test_data_2$G_pred)
print(paste0("MAPE: ", mape_consumption_model_11_narx))
mape_consumption_model_12_narx <- mape(strength_pred_12, test_data_2$G_pred)
print(paste0("MAPE: ", mape_consumption_model_12_narx))
```

```r
smape <- function(actual, predicted) {
  mean(2 * abs(predicted - actual) / (abs(actual) + abs(predicted)))
}


smape_consumption_model_narx <- smape(test_data_2$G_pred, strength_pred)
print(paste0("sMAPE: ", smape_consumption_model_narx))
smape_consumption_model_2_narx <- smape(test_data_2$G_pred, strength_pred_2)
print(paste0("sMAPE: ", smape_consumption_model_2_narx))
smape_consumption_model_3_narx <- smape(test_data_2$G_pred, strength_pred_3)
print(paste0("sMAPE: ", smape_consumption_model_3_narx))
smape_consumption_model_4_narx <- smape(test_data_2$G_pred, strength_pred_4)
print(paste0("sMAPE: ", smape_consumption_model_4_narx))
smape_consumption_model_5_narx <- smape(test_data_2$G_pred, strength_pred_5)
print(paste0("sMAPE: ", smape_consumption_model_5_narx))
smape_consumption_model_6_narx <- smape(test_data_2$G_pred, strength_pred_6)
print(paste0("sMAPE: ", smape_consumption_model_6_narx))
smape_consumption_model_7_narx <- smape(test_data_2$G_pred, strength_pred_7)
print(paste0("sMAPE: ", smape_consumption_model_7_narx))
smape_consumption_model_8_narx <- smape(test_data_2$G_pred, strength_pred_8)
print(paste0("sMAPE: ", smape_consumption_model_8_narx))
smape_consumption_model_9_narx <- smape(test_data_2$G_pred, strength_pred_9)
print(paste0("sMAPE: ", smape_consumption_model_9_narx))
smape_consumption_model_10_narx <- smape(test_data_2$G_pred, strength_pred_10)
print(paste0("sMAPE: ", smape_consumption_model_10_narx))
smape_consumption_model_11_narx <- smape(test_data_2$G_pred, strength_pred_11)
print(paste0("sMAPE: ", smape_consumption_model_11_narx))
smape_consumption_model_12_narx <- smape(test_data_2$G_pred, strength_pred_12)
print(paste0("sMAPE: ", smape_consumption_model_12_narx))
```

```
results_df_2 <- data.frame(
  Model = c(
    "Consumption Model",
    "Consumption Model 2",
    "Consumption Model 3",
    "Consumption Model 4",
    "Consumption Model 5",
    "Consumption Model 6",
    "Consumption Model 7",
    "Consumption Model 8",
    "Consumption Model 9",
    "Consumption Model 10",
    "Consumption Model 11",
    "Consumption Model 12"
  ),
  RMSE = c(
    rmse_narx,
    rmse_2_narx,
    rmse_3_narx,
    rmse_4_narx,
    rmse_5_narx,
    rmse_6_narx,
    rmse_7_narx,
    rmse_8_narx,
    rmse_9_narx,
    rmse_10_narx,
    rmse_11_narx,
    rmse_12_narx
  ),
  MAE = c(
    mae_consumption_model_narx,
    mae_consumption_model_2_narx,
    mae_consumption_model_3_narx,
    mae_consumption_model_4_narx,
    mae_consumption_model_5_narx,
    mae_consumption_model_6_narx,
    mae_consumption_model_7_narx,
    mae_consumption_model_8_narx,
    mae_consumption_model_9_narx,
    mae_consumption_model_10_narx,
    mae_consumption_model_11_narx,
    mae_consumption_model_12_narx
  ),
  MAPE = c(
    mape_consumption_model_narx,
    mape_consumption_model_2_narx,
    mape_consumption_model_3_narx,
    mape_consumption_model_4_narx,
    mape_consumption_model_5_narx,
    mape_consumption_model_6_narx,
    mape_consumption_model_7_narx,
    mape_consumption_model_8_narx,
    mape_consumption_model_9_narx,
    mape_consumption_model_10_narx,
    mape_consumption_model_11_narx,
    mape_consumption_model_12_narx
  ),
  sMAPE = c(
    smape_consumption_model_narx,
    smape_consumption_model_2_narx,
    smape_consumption_model_3_narx,
    smape_consumption_model_4_narx,
    smape_consumption_model_5_narx,
    smape_consumption_model_6_narx,
    smape_consumption_model_7_narx,
    smape_consumption_model_8_narx,
    smape_consumption_model_9_narx,
    smape_consumption_model_10_narx,
    smape_consumption_model_11_narx,
    smape_consumption_model_12_narx
  ),
  Parameters = NA
)
```

```
# calculate the number of parameters for each model and update the data frame
for (i in 1:nrow(results_df_2)) {
  model_formula_2 <- as.formula(paste("G_pred ~", paste0(colnames(training_data_2)[-1], collapse = " + ")))
  if (grepl("Model 1$", results_df_2$Model[i])) {
    nn <- neuralnet(model_formula_2, data = training_data_2, hidden = 3)
  } else if (grepl("Model 2$", results_df_2$Model[i])) {
    nn <- neuralnet(model_formula_2, data = training_data_2, hidden = c(5, 2))
  } else {
    nn <- neuralnet(model_formula_2, data = training_data_2, hidden = 3)
  }
  num_params <- sum(sapply(nn$weights, length))
  results_df_2$Parameters[i] <- num_params
}


# format data frame into table using kable

kable(results_df_2, caption = "Comparison of RMSE, MAE, MAPE, sMAPE, and number of parameters for twelve neural network models")
```

```r
#AR scatter plot graph for two models to compare and check

# create a dataframe with the actual and predicted values
results_df <- data.frame(actual = test_data$G_pred, predicted = model_results)

# create the plot
ggplot(data = results_df, aes(x = actual, y = predicted)) +
  geom_point() +
  geom_abline(intercept = 0, slope = 1, color = "red", linetype = "dashed") +
  xlab("Actual") +
  ylab("Predicted") +
  ggtitle("Actual vs. Predicted")

ggplot(data = results_df, aes(x = actual, y = predicted)) +
  geom_point() +
  geom_abline(intercept = 0, slope = 1, color = "red", linetype = "dashed") +
  xlab("Actual") +
  ylab("Predicted") +
  ggtitle("Actual vs. Predicted") +
  geom_text(x = max(results_df$actual) - 0.1 * diff(range(results_df$actual)),
            y = max(results_df$predicted) - 0.1 * diff(range(results_df$predicted)),
            label = paste0("RMSE: ", round(rmse, 2), "%"),
            hjust = 1, vjust = 1, color = "blue") +
  geom_text(x = max(results_df$actual) - 0.1 * diff(range(results_df$actual)),
            y = max(results_df$predicted) - 0.2 * diff(range(results_df$predicted)),
            label = paste0("MAE: ", round(mae_consumption_model, 2), "%"),
            hjust = 1, vjust = 1, color = "blue") +
  geom_text(x = max(results_df$actual) - 0.1 * diff(range(results_df$actual)),
            y = max(results_df$predicted) - 0.3 * diff(range(results_df$predicted)),
            label = paste0("MAPE: ", round(mape_consumption_model, 2), "%"),
            hjust = 1, vjust = 1, color = "blue") +
  geom_text(x = max(results_df$actual) - 0.1 * diff(range(results_df$actual)),
            y = max(results_df$predicted) - 0.4 * diff(range(results_df$predicted)),
            label = paste0("sMAPE: ", round(smape_consumption_model, 2), "%"),
            hjust = 1, vjust = 1, color = "blue")
```

```r
# create a dataframe with the actual and predicted values
results_df_4 <- data.frame(actual = test_data$G_pred, predicted = model_results_2)

# create the plot
ggplot(data = results_df_4, aes(x = actual, y = predicted)) +
  geom_point() +
  geom_abline(intercept = 0, slope = 1, color = "red", linetype = "dashed") +
  xlab("Actual") +
  ylab("Predicted") +
  ggtitle("Actual vs. Predicted")
```

```r
#NARX scatter plot graph for two models to compare and check

# create a dataframe with the actual and predicted values
results_df_2 <- data.frame(actual = test_data_2$G_pred, predicted = model_results_narx)

# create the plot
ggplot(data = results_df_2, aes(x = actual, y = predicted)) +
  geom_point() +
  geom_abline(intercept = 0, slope = 1, color = "red", linetype = "dashed") +
  xlab("Actual") +
  ylab("Predicted") +
  ggtitle("Actual vs. Predicted")

# create a dataframe with the actual and predicted values
results_df_3 <- data.frame(actual = test_data_2$G_pred, predicted = model_results_2_narx)

# create the plot
ggplot(data = results_df_3, aes(x = actual, y = predicted)) +
  geom_point() +
  geom_abline(intercept = 0, slope = 1, color = "red", linetype = "dashed") +
  xlab("Actual") +
  ylab("Predicted") +
  ggtitle("Actual vs. Predicted")

ggplot(data = results_df_3, aes(x = actual, y = predicted)) +
  geom_point() +
  geom_abline(intercept = 0, slope = 1, color = "red", linetype = "dashed") +
  xlab("Actual") +
  ylab("Predicted") +
  ggtitle("Actual vs. Predicted") +
  geom_text(x = max(results_df_3$actual) - 0.1 * diff(range(results_df_3$actual)),
            y = max(results_df_3$predicted) - 0.1 * diff(range(results_df_3$predicted)),
            label = paste0("RMSE: ", round(rmse_narx, 2), "%"),
            hjust = 1, vjust = 1, color = "blue") +
  geom_text(x = max(results_df_3$actual) - 0.1 * diff(range(results_df_3$actual)),
            y = max(results_df_3$predicted) - 0.2 * diff(range(results_df_3$predicted)),
            label = paste0("MAE: ", round(mae_consumption_model_narx, 2), "%"),
            hjust = 1, vjust = 1, color = "blue") +
  geom_text(x = max(results_df_3$actual) - 0.1 * diff(range(results_df_3$actual)),
            y = max(results_df_3$predicted) - 0.3 * diff(range(results_df_3$predicted)),
            label = paste0("MAPE: ", round(mape_consumption_model_narx, 2), "%"),
            hjust = 1, vjust = 1, color = "blue") +
  geom_text(x = max(results_df_3$actual) - 0.1 * diff(range(results_df_3$actual)),
            y = max(results_df_3$predicted) - 0.4 * diff(range(results_df_3$predicted)),
            label = paste0("sMAPE: ", round(smape_consumption_model_narx, 2), "%"),
            hjust = 1, vjust = 1, color = "blue")
```