ENVIRONMENTAL DATASET

Data Analysis & Statistical Learning Report in R

USAMA ALI

Description

The dataset contains measurements of ozone, radiation, temperature, and wind from a monitoring station in New York. The data was collected over a period of 153 days, with 111 observations per day.

The variables and the units in the dataset are as follows:

ozone : the amount of ozone in parts per billion
radiation : the amount of solar radiation in Langleys
temperature : the temperature in degrees Fahrenheit
wind : the wind speed in miles per hour

We can have a look of some of the values of all four variables

```
> str(environmental)
```

```
'data.frame':   111 obs. of  4 variables:
 $ ozone      :  num  41 36 12 18 23 19 8 16 11 14 ...
 $ radiation  :  num  190 118 149 313 299 99 19 256 290 274 ...
 $ temperature: num  67 72 74 62 65 59 61 69 66 68 ...
 $ wind       :  num  7.4 8 12.6 11.5 8.6 13.8 20.1 9.7 9.2 10.9 ...
```

Univariate Analysis:

In univariate analysis we analyze each variable separately. So, we begin from here:

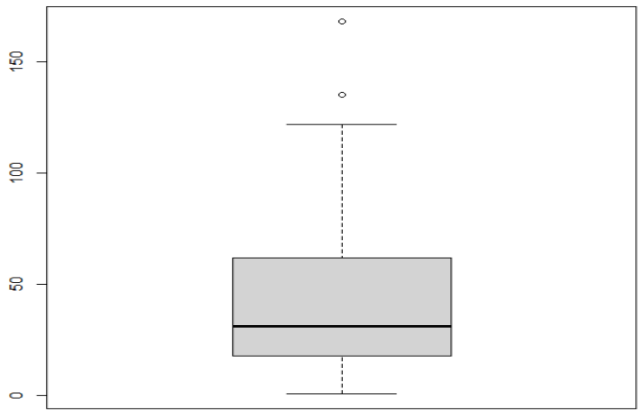
Ozone:

First, we need to look at the basic statistics of the ozone variable

```
> summary(ozone)
```

```
   Min. 1st Qu.  Median    Mean   3rd Qu.    Max.
    1.0    18.0    31.0    42.1    62.0    168.0
```

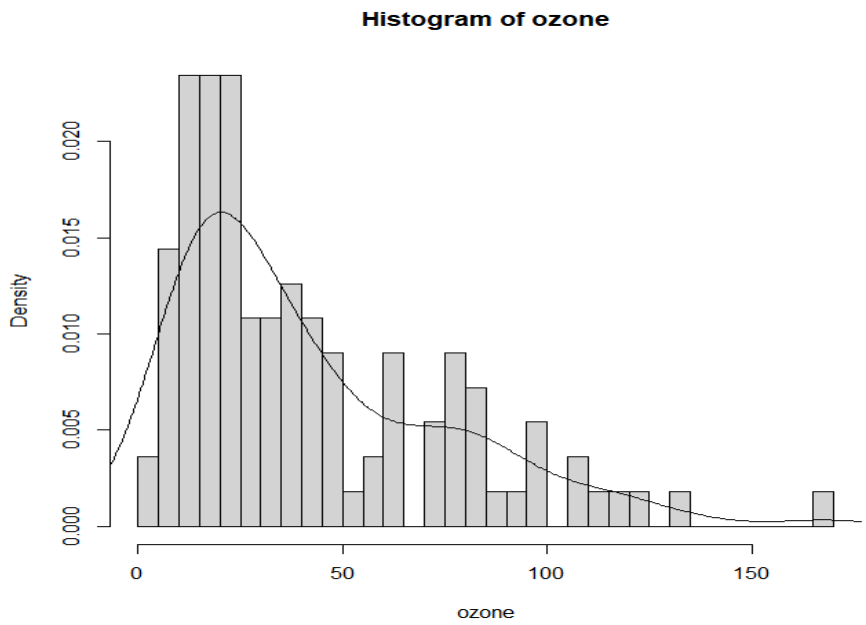
```
> boxplot(ozone)
```



The boxplot represents the summary of the distribution of the ozone variable.
The median is 31.0. This means that 50% of the data falls below 31.0.
The mean is 42.1, since the mean is greater than median hence the data is positively skewed.
The third quartile (Q3) is 62.0. This means that 75% of the data falls below 62.0.

The better representation of distribution of ozone can be visualized by histogram:

```
> hist(ozone,breaks=30,freq = FALSE)
> lines(density(ozone))
```

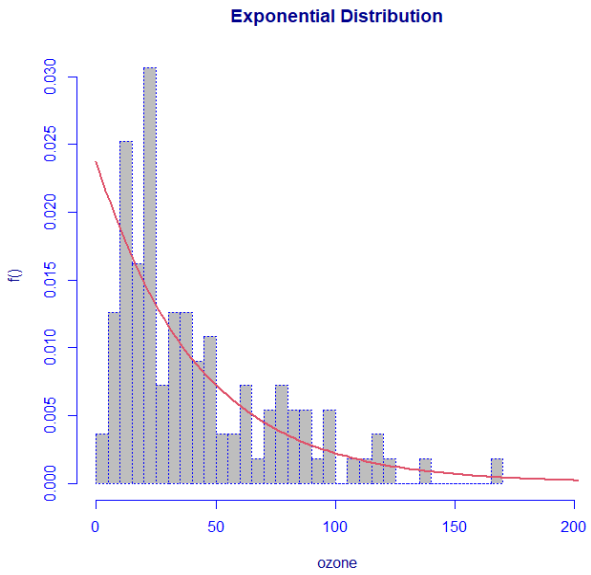


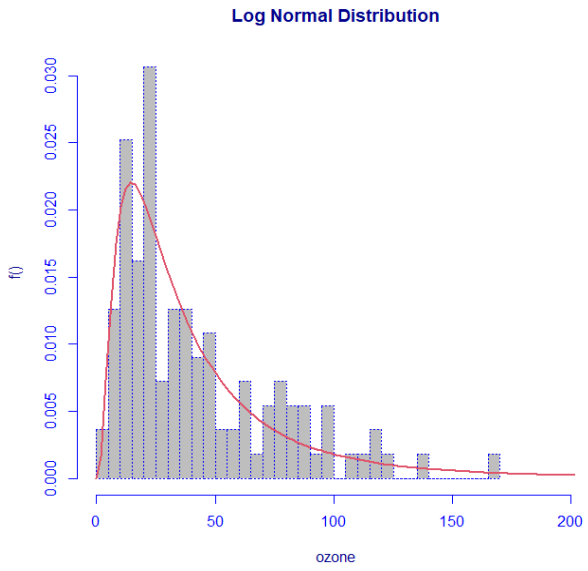

```
> skewness(ozone)
[1] 1.231275
> kurtosis(ozone)
[1] 1.128994
```

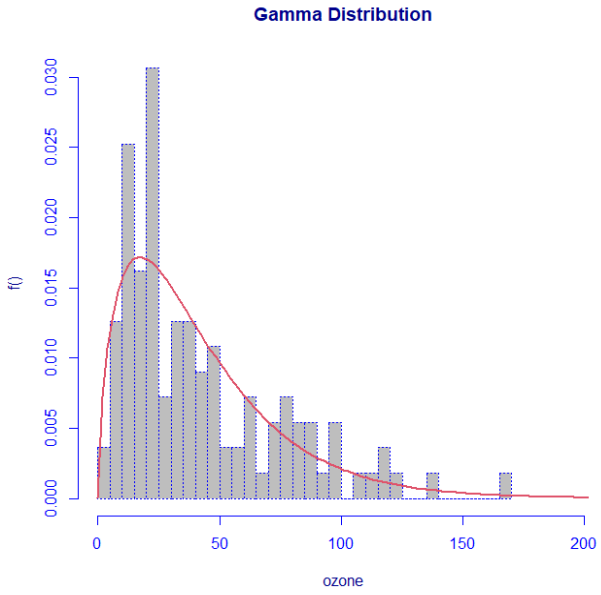
We can see from the histogram that it has long tail on right hand side and value of ozone is more concentrated on the left-hand side that can also be verified from the value of skewness. Here we also measured the excess kurtosis, the value of kurtosis shows that distribution of ozone is more peaked and heavily tailed than normal distribution.

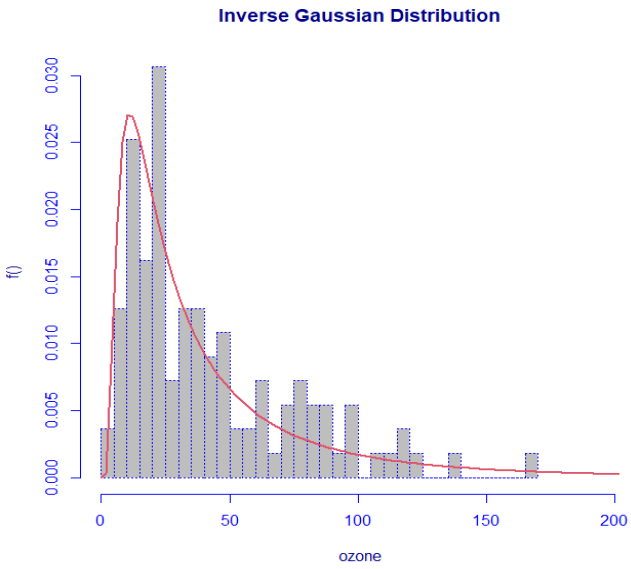
Now fitting some of the common distributions for ozone variable:

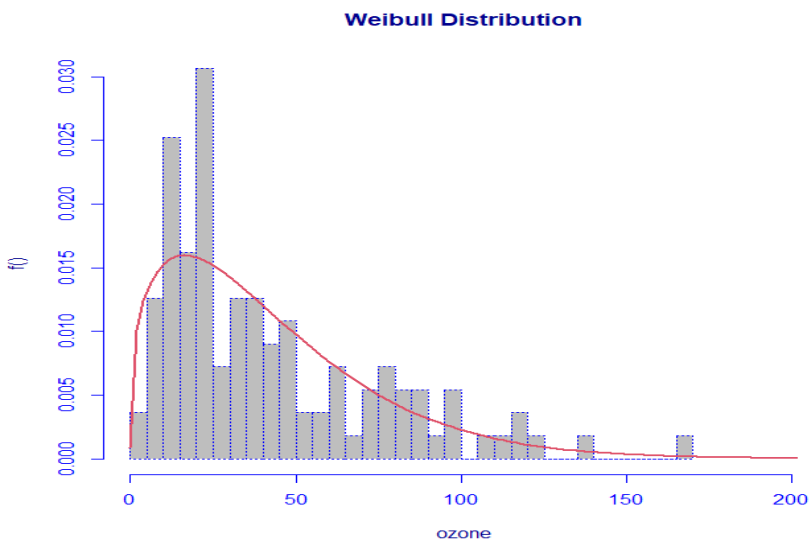
```
> ozone.fit.exp <-histDist(ozone,family = EXP,nbins = 30,main = "Exponential Distribution")
> ozone.fit.lognorm <- histDist(ozone,family = LOGNO,nbins=30,main="Log Normal Distribution")
> ozone.fit.GA <- histDist(ozone,family = GA,nbins = 30,main="Gamma Distribution")
> ozone.fit.IG <- histDist(ozone,family = IG,nbins = 30,main="Inverse Gaussian Distribution")
> ozone.fit.WEI <- histDist(ozone,family=WEI,nbins=30,main = "Weibull Distribution")
```


Exponential Distribution
f()
0.000
0.005
0.010
0.015
0.020
0.025
0.030
0
50
100
150
200
ozone


Log Normal Distribution
f()
0.000
0.005
0.010
0.015
0.020
0.025
0.030
0
50
100
150
200
ozone


Gamma Distribution
f()
0.000
0.005
0.010
0.015
0.020
0.025
0.030
0
50
100
150
200
ozone


Inverse Gaussian Distribution
f()
0.000
0.005
0.010
0.015
0.020
0.025
0.030
0
50
100
150
200
ozone


Weibull Distribution
f()
0.000
0.005
0.010
0.015
0.020
0.025
0.030
0
50
100
150
200
ozone

```
> > data.frame(row.names = c("Exponential",'Log Normal','Gamma ',"Inverse Gussian","Weibull"),
+         AIC =
c(AIC(ozone.fit.exp),AIC(ozone.fit.lognorm),AIC(ozone.fit.GA),AIC(ozone.fit.IG),AIC(ozone.fit.WEI)),
+         SBC =
c(ozone.fit.exp$sbc,ozone.fit.lognorm$sbc,ozone.fit.GA$sbc,ozone.fit.IG$sbc,ozone.fit.WEI$sbc),
+         logLik= c(logLik(ozone.fit.exp),logLik(ozone.fit.lognorm),logLik(ozone.fit.GA),
+                 logLik(ozone.fit.IG),logLik(ozone.fit.WEI)))
```

	AIC	SBC	logLik
Exponential	1054.286	1056.995	-526.1429
Log Normal	1044.363	1049.782	-520.1813
Gamma	1040.520	1045.939	-518.2601
Inverse Gussian	1062.103	1067.522	-529.0513
Weibull	1042.776	1048.195	-519.3882

The Akaike Information Criteria (AIC) and the Bayesian Information Criteria (BIC) are model selection metrics that help to compare the relative quality of different models. AIC and BIC balance the goodness of fit of a model with the complexity of the model. Lower AIC and BIC values indicate a better model fit. Also we find the log-likelihood which is the measure of the goodness of fit of the model and it's value should be maximized for the good fitted model. The Gamma distribution has the lowest AIC and BIC values, indicating that it is the best fitting distribution for the ozone data among the considered distributions.

```
> LR.test(ozone.fit.WEI,ozone.fit.GA)
 Null model: deviance= 1038.776 with  2 deg. of freedom
Altenative model: deviance= 1036.52 with  2 deg. of freedom
 LRT = 2.256101 with 0 deg. of freedom and p-value= 0
```

The output provides information about the deviance (a measure of the goodness-of-fit of the model) and the degrees of freedom for each model. The difference in deviances (2.256101) is the test statistic for the likelihood ratio test. The test statistic is used to compute the p-value, which is equal to 0 in this case. A p-value of 0 indicates that the alternative model (the gamma distribution in this case) is significantly better than the null model (the Weibull distribution).

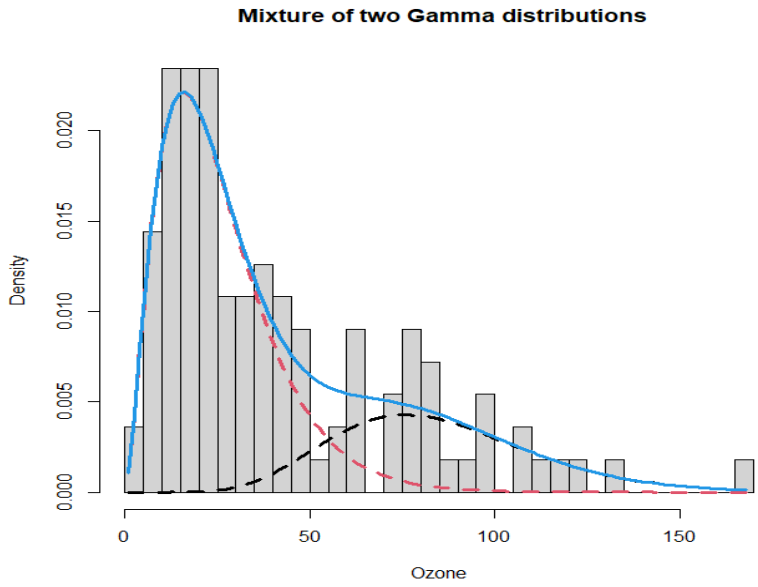
Now fitting the model with Gamma distribution using gamlssMXfits() function:

```
> ozone.mix.GA <- gamlssMXfits(n=5,ozone~1,family = GA,K=2,data=environmental)
> ozone.mix.GA$aic                    > ozone.mix.GA$sbc
[1] 1039.16                           [1] 1052.708
```

We are now plotting this distribution,

```
> hist(ozone, breaks = 30,freq = FALSE, xlab = "Ozone" ,main=" Mixture of two Gamma distributions")
> lines(seq(min(ozone),max(ozone),length=length(ozone)),
+     ozone.mix.GA[["prob"]][1]*dGA(seq(min(ozone),max(ozone),length=length(ozone)),
+                         mu = mu.hat1, sigma = sigma.hat1),lty=2,lwd=3,col=1)
> lines(seq(min(ozone),max(ozone),length=length(ozone)),
+     ozone.mix.GA[["prob"]][2]*dGA(seq(min(ozone),max(ozone),length=length(ozone)),
+                         mu = mu.hat2, sigma = sigma.hat2),lty=2,lwd=3,col=2)
> lines(seq(min(ozone),max(ozone),length=length(ozone)),
+     ozone.mix.GA[["prob"]][1]*dGA(seq(min(ozone),max(ozone),length=length(ozone)),
```

```
+                                mu = mu.hat1, sigma = sigma.hat1)+
+       ozone.mix.GA[["prob"]][2]*dGA(seq(min(ozone),max(ozone),length=length(ozone)),
+                                mu = mu.hat2, sigma = sigma.hat2),lty=1,lwd=3,col=4)
```

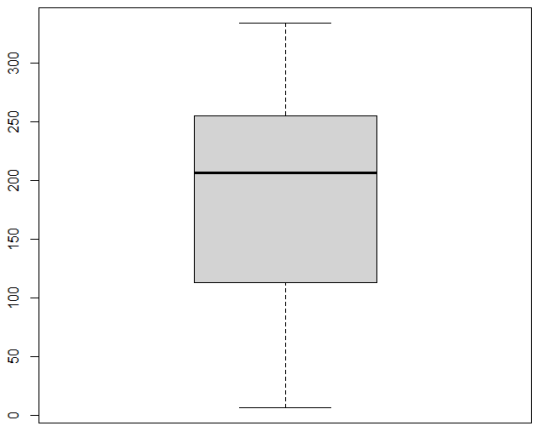


As ozone consists of mixture of two gamma distributions, the distribution on left hand side has higher probability and captures the most of the data when compared to the right hand side of distribution.

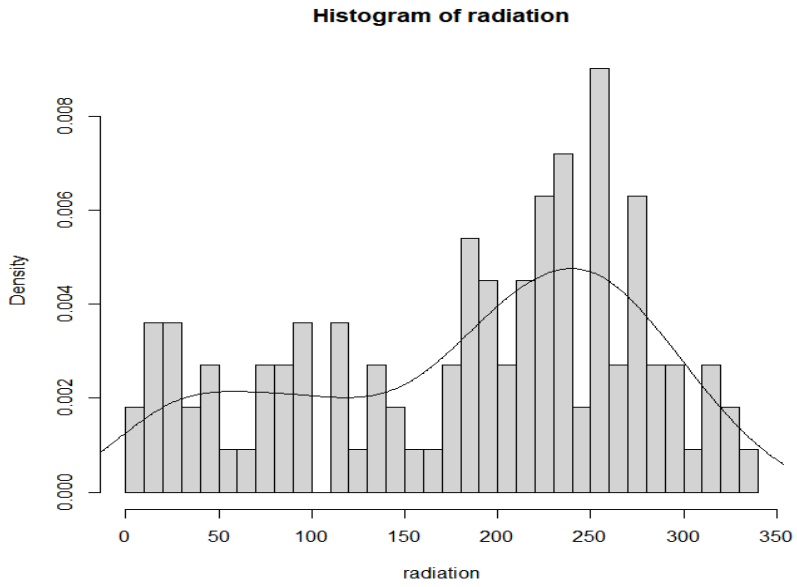
Radiation:

```
> summary(radiation)
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
7.0	113.5	207.0	184.8	255.5	334.0



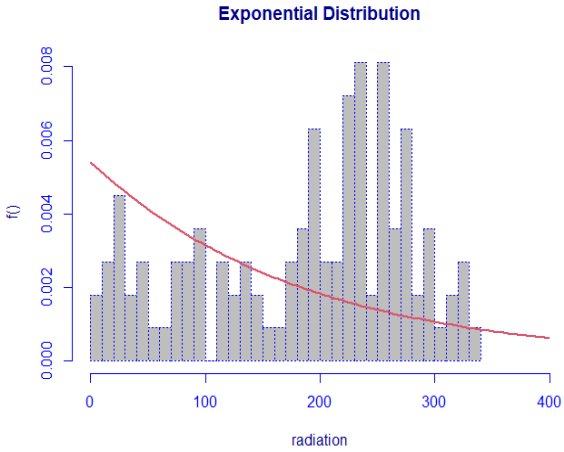
The boxplot show that values of radiation is widely spread over the range of 7 Langley to 334 Langley but the higher concentration of data is between 180 to 340. Also the mean is less than the median which is indication of positively skewed data that can be later confirmed from histogram.

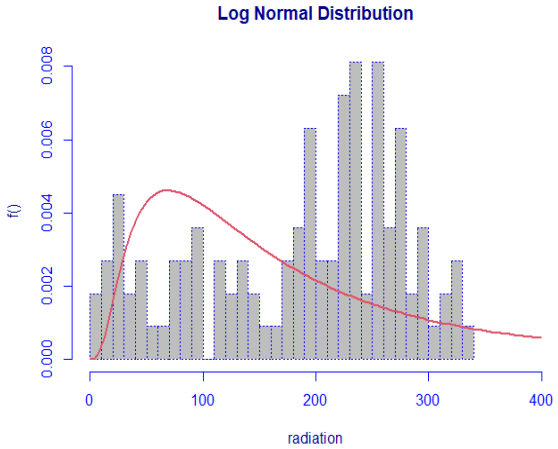


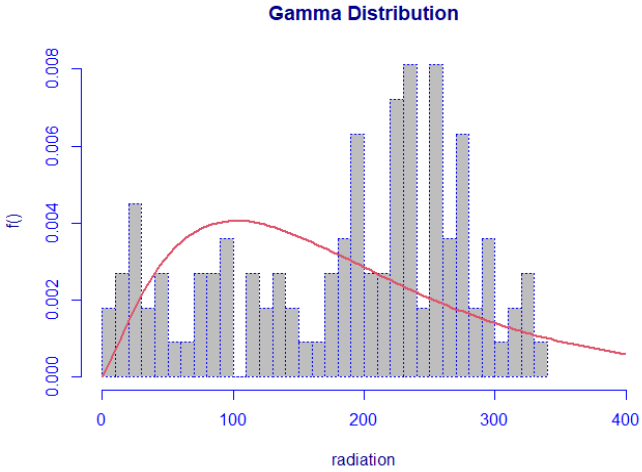

```
> skewness(radiation)
[1] -0.4796906
> kurtosis(radiation)
[1] -0.9663681
```

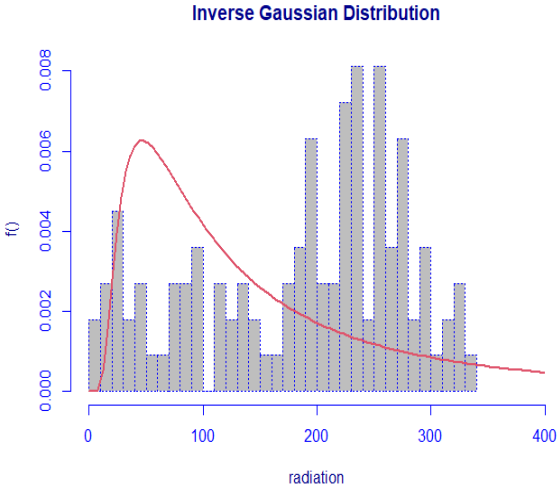
Now we try to fit some of the common distributions,

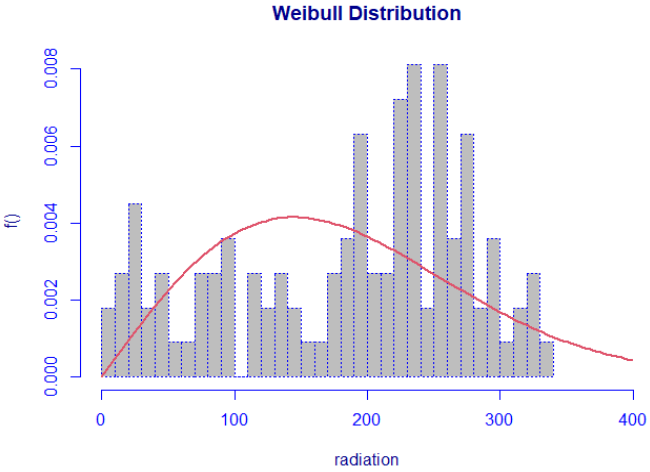
```
> radiation.fit.exp <-histDist(radiation,family = EXP,nbins = 30,main = "Exponential Distribution")
> radiation.fit.lognorm <- histDist(radiation,family = LOGNO,nbins=30,main="Log Normal
Distribution")
> radiation.fit.GA <- histDist(radiation,family = GA,nbins = 30,main="Gamma Distribution")
> radiation.fit.IG <- histDist(radiation,family = IG,nbins = 30,main="Inverse Gaussian Distribution")
> radiation.fit.WEI <- histDist(radiation,family=WEI,nbins=30,main = "Weibull Distribution")
```


Exponential Distribution
f()
0.000
0.002
0.004
0.006
0.008
0
100
200
300
400
radiation


Log Normal Distribution
f()
0.000
0.002
0.004
0.006
0.008
0
100
200
300
400
radiation


Gamma Distribution
f()
0.000
0.002
0.004
0.006
0.008
0
100
200
300
400
radiation


Inverse Gaussian Distribution
f()
0.000
0.002
0.004
0.006
0.008
0
100
200
300
400
radiation


Weibull Distribution
f()
0.000
0.002
0.004
0.006
0.008
0
100
200
300
400
radiation

```
> data.frame(row.names = c("Exponential",'Log Normal','Gamma ',"Inverse Gussian","Weibull"),
+         AIC = c(AIC(radiation.fit.exp),AIC(radiation.fit.lognorm),AIC(radiation.fit.GA),
AIC(radiation.fit.IG),AIC(radiation.fit.WEI)),
+         SBC = c(radiation.fit.exp$sbc,radiation.fit.lognorm$sbc,radiation.fit.GA$sbc,
radiation.fit.IG$sbc,radiation.fit.WEI$sbc))
```

	AIC	SBC
Exponential	1382.681	1385.391
Log Normal	1391.109	1396.528
Gamma	1351.547	1356.966
Inverse Gussian	1421.650	1427.069
Weibull	1332.112	1337.531

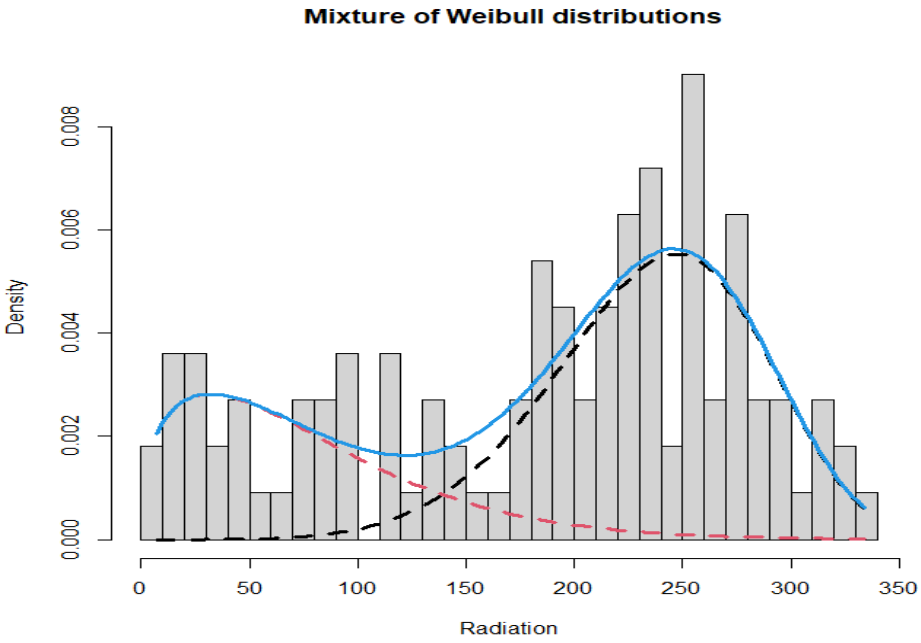
These values are the Akaike Information Criterion (AIC) and the Schwarz Bayesian Criterion (SBC) values for different probability distributions applied to the data. AIC and SBC are both measures of model fit, where a lower value indicates a better fit. In this case, the Weibull distribution has the lowest AIC and SBC values, indicating that it is the best fit for the data.

Fitting the Weibull distribution using gamlssMXfits() function,

```
> radiation.mix.WEI <- gamlssMXfits(n=5,radiation~1,family=WEI,K=2,data=environmental)
> radiation.mix.WEI$aic                    > radiation.mix.WEI$sbc
[1] 1283.983                               [1] 1297.531
```

Then plotting this distribution:

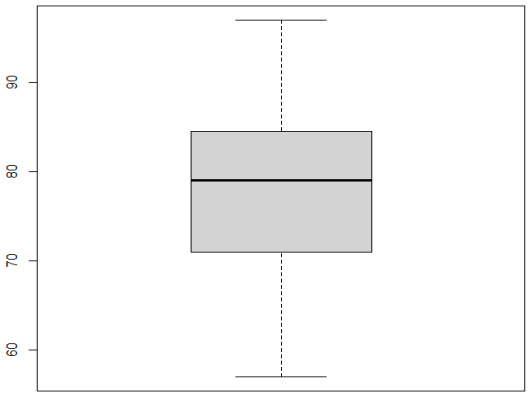
```
> hist(radiation, breaks = 30,freq = FALSE, xlab = "Radiation" ,main="Radiation Mixture of Gamma
distributions")
> lines(seq(min(radiation),max(radiation),length=length(radiation)),
+
radiation.mix.GA[["prob"]][1]*dWEI(seq(min(radiation),max(radiation),length=length(radiation)),
+                    mu = mu.hat1, sigma = sigma.hat1),lty=2,lwd=3,col=1)
> lines(seq(min(radiation),max(radiation),length=length(radiation)),
+
radiation.mix.GA[["prob"]][2]*dWEI(seq(min(radiation),max(radiation),length=length(radiation)),
+                    mu = mu.hat2, sigma = sigma.hat2),lty=2,lwd=3,col=2)
> lines(seq(min(radiation),max(radiation),length=length(radiation)),
+
radiation.mix.GA[["prob"]][1]*dWEI(seq(min(radiation),max(radiation),length=length(radiation)),
+                    mu = mu.hat1, sigma = sigma.hat1)+

radiation.mix.GA[["prob"]][2]*dWEI(seq(min(radiation),max(radiation),length=length(radiation)),
+                     mu = mu.hat2, sigma = sigma.hat2),lty=1,lwd=3,col=4)
```

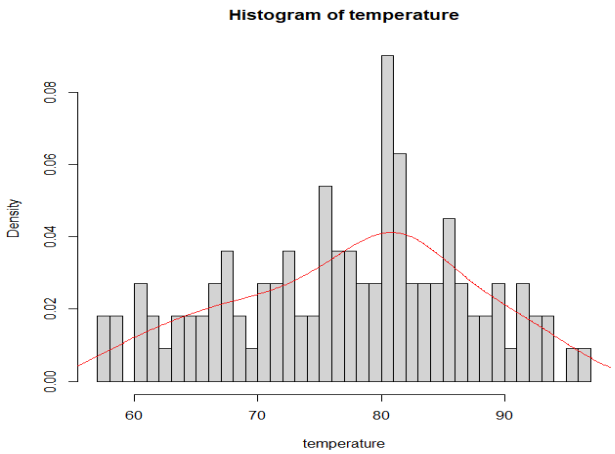


Temperature:

```
> summary(temperature)
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
57.00	71.00	79.00	77.79	84.50	97.00



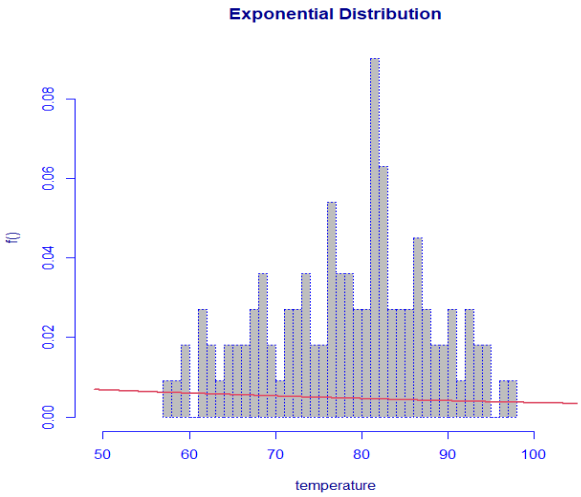


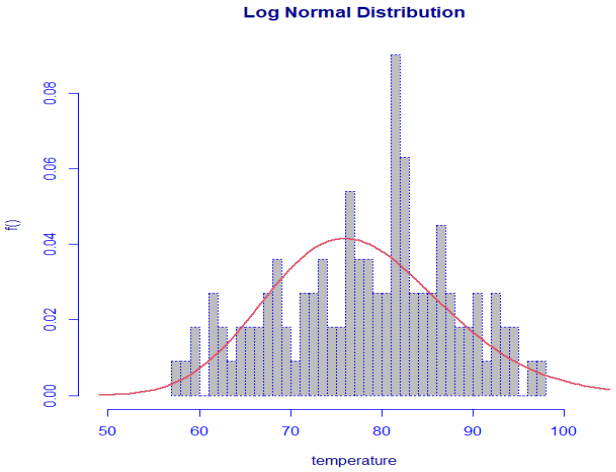

```
> skewness(temperature)
```

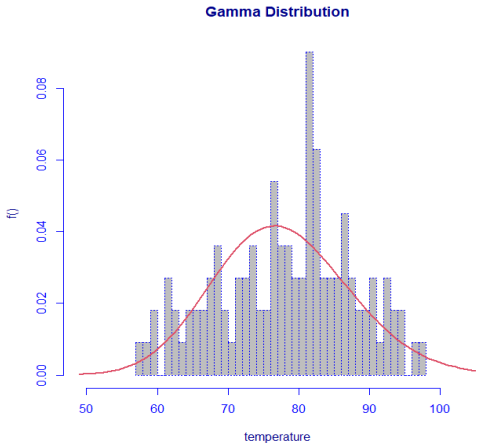
[1] -0.2220609

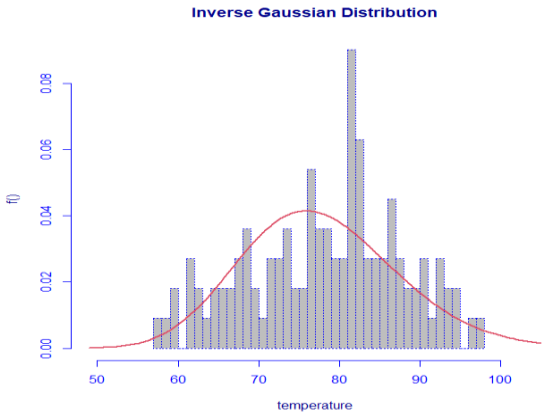
```
> kurtosis(temperature)
```

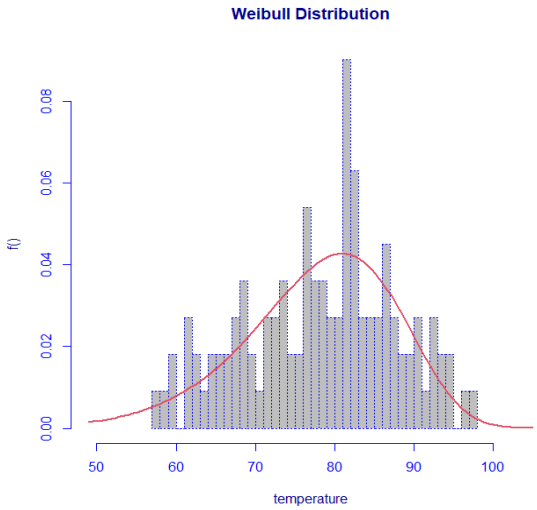
[1] -0.7098729


Exponential Distribution
f()
0.00
0.02
0.04
0.06
0.08
50
60
70
80
90
100
temperature


Log Normal Distribution
f()
0.00
0.02
0.04
0.06
0.08
50
60
70
80
90
100
temperature


Gamma Distribution
f()
0.00
0.02
0.04
0.06
0.08
50
60
70
80
90
100
temperature


Inverse Gaussian Distribution
f()
0.00
0.02
0.04
0.06
0.08
50
60
70
80
90
100
temperature


Weibull Distribution
f()
0.00
0.02
0.04
0.06
0.08
50
60
70
80
90
100
temperature

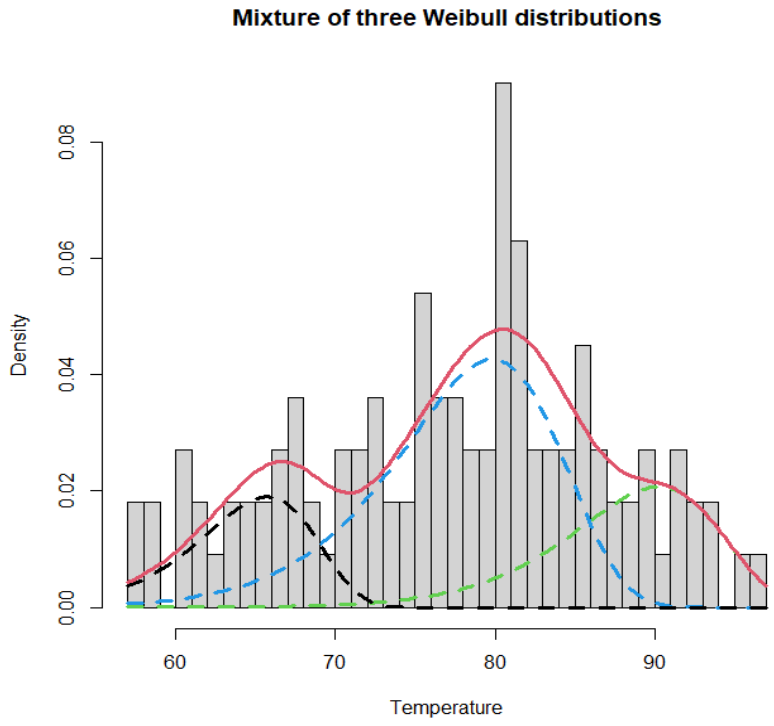
```
> data.frame(row.names = c("Exponential",'Log Normal','Gamma ',"Inverse Gussian","Weibull"),
+                     AIC = c(AIC(temp.fit.exp),AIC(temp.fit.lognorm),AIC(temp.fit.GA),
AIC(temp.fit.IG),AIC(temp.fit.WEI)),
+                     SBC = c(temp.fit.exp$sbc,temp.fit.lognorm$sbc,temp.fit.GA$sbc,
temp.fit.IG$sbc,temp.fit.WEI$sbc))
```

	AIC	SBC
Exponential	1190.5988	1193.3084
Log Normal	823.0498	828.4688
Gamma	821.0899	826.5089
Inverse Gussian	822.9935	828.4126
Weibull	817.4259	822.8449

In this table, the Weibull distribution has the lowest AIC and SBC values, which suggests that it is the best fit model.

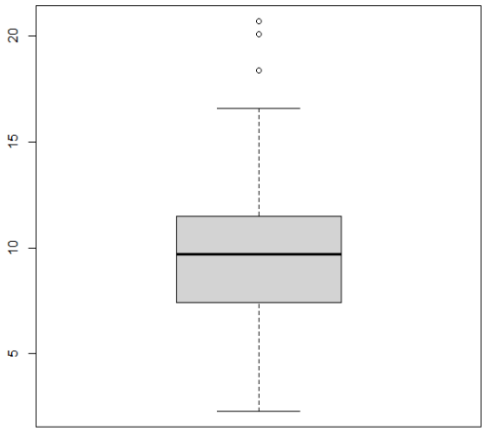
Now fitting the model with Weibull distribution,

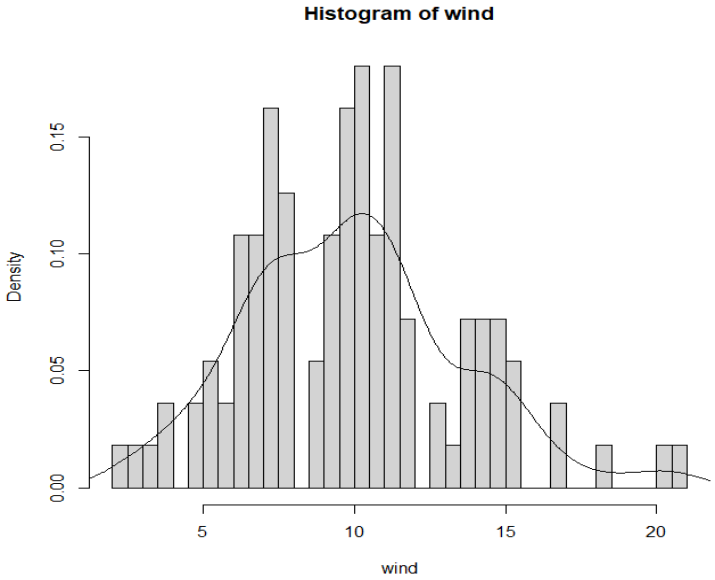
```
> temp.mix.WEI <- gamlssMXfits(n=5,temperature~1,family=WEI,K=3,data=environmental)
> temp.mix.WEI$aic                    > temp.mix.WEI$sbc
[1] 841.0137                          [1] 819.3375
```



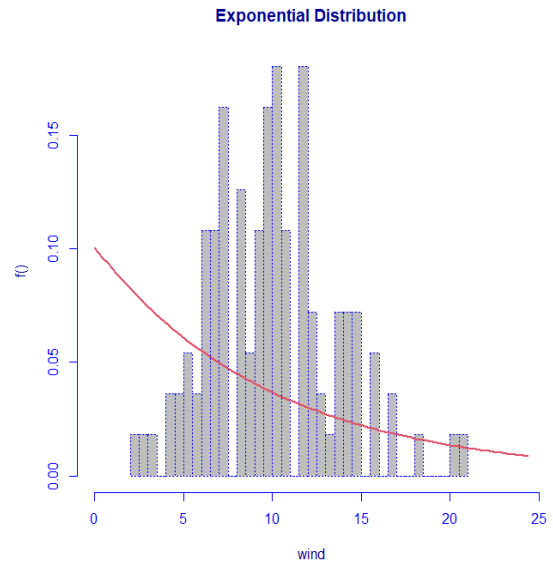
Wind

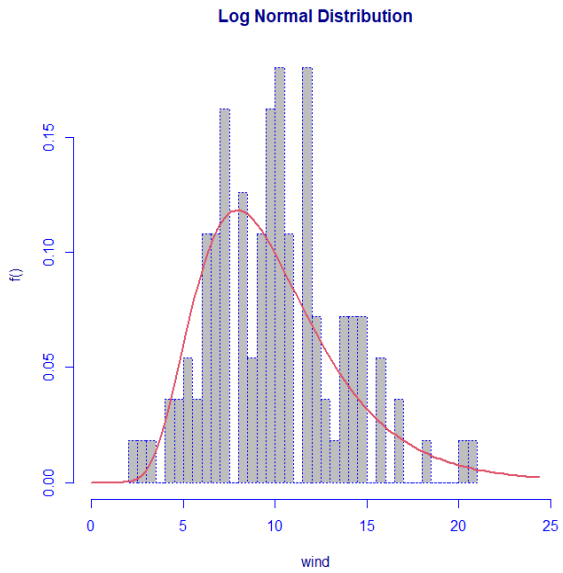
```
> summary(wind)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  2.300   7.400   9.700   9.939  11.500  20.700
> boxplot(wind)
```

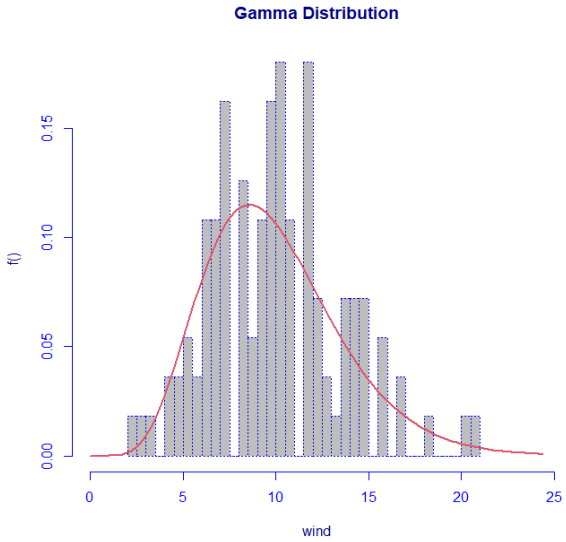


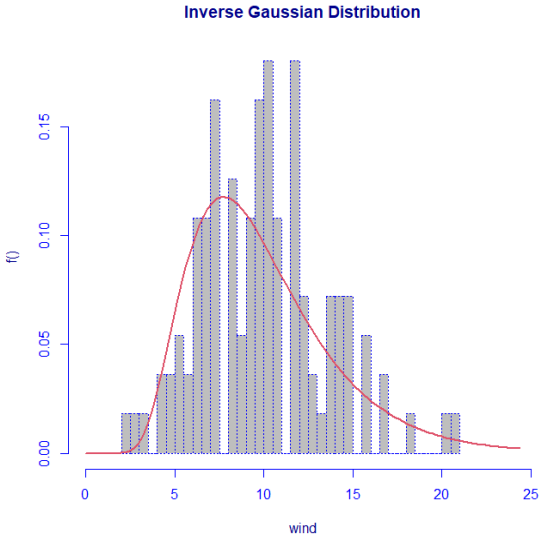


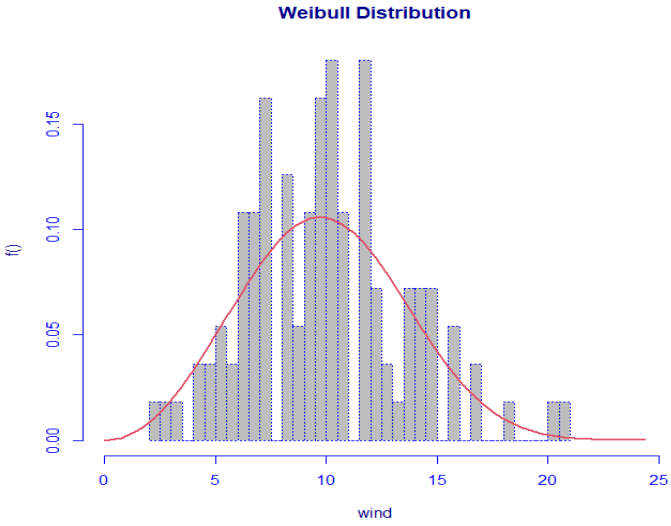

```
> skewness(wind)
[1] 0.4476014
> kurtosis(wind)
[1] 0.2220383
```


Exponential Distribution
f()
0.00
0.05
0.10
0.15
0
5
10
15
20
25
wind


Log Normal Distribution
f()
0.00
0.05
0.10
0.15
0
5
10
15
20
25
wind


Gamma Distribution
f()
0.00
0.05
0.10
0.15
0
5
10
15
20
25
wind


Inverse Gaussian Distribution
f()
0.00
0.05
0.10
0.15
0
5
10
15
20
25
wind


Weibull Distribution
f()
0.00
0.05
0.10
0.15
0
5
10
15
20
25
wind

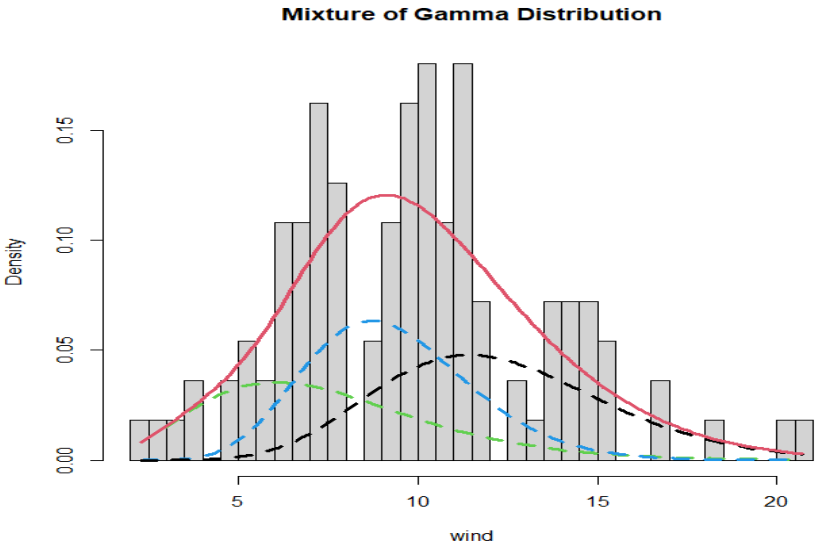
```
> data.frame(row.names = c("Exponential",'Log Normal','Gamma ',"Inverse Gussian","Weibull"),
+          AIC = c(AIC(wind.fit.exp),AIC(wind.fit.lognorm),AIC(wind.fit.GA),
AIC(wind.fit.IG),AIC(wind.fit.WEI)),
+          SBC = c(wind.fit.exp$sbc,wind.fit.lognorm$sbc,wind.fit.GA$sbc,
wind.fit.IG$sbc,wind.fit.WEI$sbc))
```

	AIC	SBC
Exponential	733.8097	736.5192
Log Normal	606.0190	611.4381
Gamma	598.0889	603.5080
Inverse Gussian	608.5869	614.0060
Weibull	598.1419	603.5610

```
> wind.mix.GA <- gamlssMXfits(n=6,wind~1,family = GA,K=3,data = environmental)
```

```
> wind.mix.GA$aic
[1] 607.3494
```

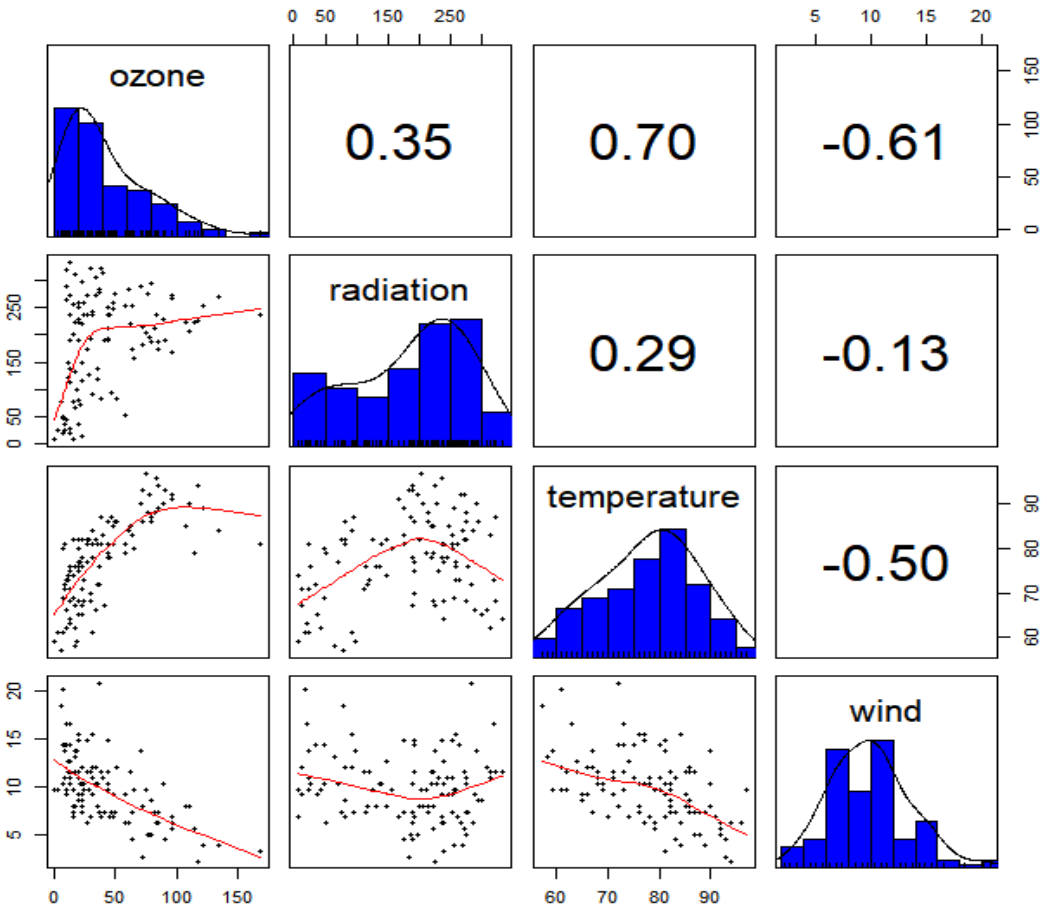
```
> wind.mix.GA$sbc
[1] 629.0256
```



Premilinary Analysis

Preliminary Analysis is crucial for finding the relation between the variables as if there exists some relation, we can do Principal Component Analysis and clustering techniques.

```
> pairs.panels(environmental, hist.col="blue", density = TRUE, ellipses = FALSE)
```



Here we are visualizing the scatterplots, distributions and correlation between variables. There is strong positively correlation between temperature and ozone, and the negatively moderated correlation between ozone and wind correlation. These values are indicates we can represent the same data in low dimensional space as there exists correlation between variables. Also the scattering plots gives idea that there is at least some clustering structure in our dataset but detailed pre analysis for clustering is present in later section of this report.

Principle Component Analysis

```
> pc <- prcomp(environmental,scale=TRUE)
```

The output of the prcomp() function is a list that contains the principal components of the data. The $rotation component of this list gives the loadings of the original variables on the principal components. The loading of a variable on a principal component is a measure of the correlation between the variable and the principal component.

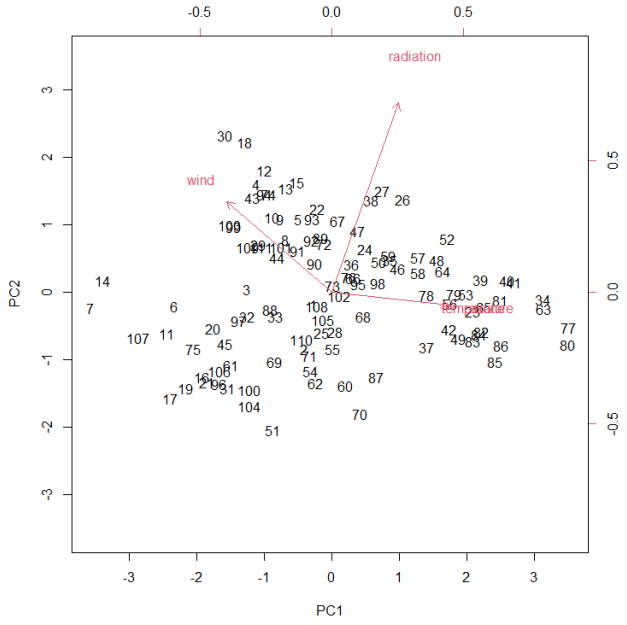
```
> pc$rotation[,1:4]
```

	PC1	PC2	PC3	PC4
ozone	0.5890271	-0.06304115	0.1137638	0.7975780
radiation	0.3168987	0.89855477	-0.2773707	-0.1234503
temperature	0.5527125	-0.06128476	0.6585842	-0.5069713
wind	-0.4971228	0.42996431	0.6902102	0.3026705

It is evident from first principal component ozone and temperature is strongly positively correlated while wind is negatively correlated. Radiation is slightly less correlated with PC1 while radiation very strongly correlated with PC2 having value of 0.89.

The effect of each variable can be visually explained using biplot:

```
> biplot(pc,scale=0)
```



Unit 63, has high value of temperature also indicates the high value of ozone

Unit 100, high value of wind but low value of ozone and temperature

Unit 27, high value of radiation does not correspond to significant value of ozone.

We applied some techniques to find that number of PCs we should retain

1. Proportion of Variance Explained:

```
> sdev <- pc$sdev
> pve <- sdev^2/sum(sdev^2)
[1] 0.59004947 0.22363944 0.11894698 0.06736411
```

PVE is a measure of how much of the total variance in the data is explained by each principal component. In this case, the first principal component explains 59% of the variance in the data, the second principal component explains 22.36% of the variance.

```
> cumsum(pve)
```

[1] 0.5900495 0.8136889 0.9326359 1.0000000

We can see from cumulative sum of PVE that only first two PC components includes 81 % of information. Also visualizing,

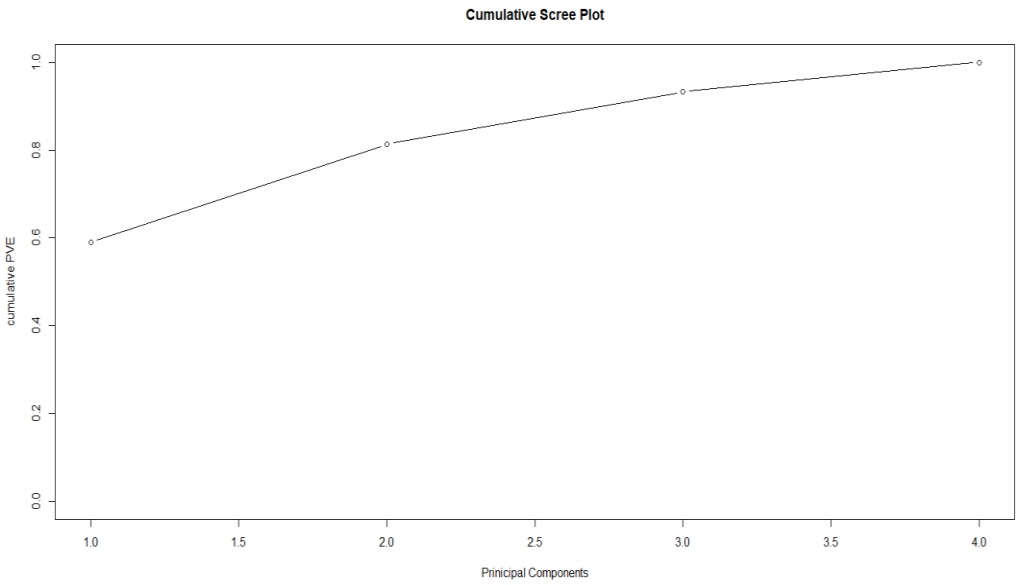
2. Kaiser Rule:

```
> pc.var <- pc$sdev^2
> pc.var
```

[1] 2.3601979 0.8945578 0.4757879 0.2694564)
Kaiser rule states that retain as many pcs as for which the variance is greater than 1. So this rule suggests that retain only one PC.

3. Scree Plot:

```
> plot(cumsum(pve),ylim = c(0,1),type = 'b',main = "Cumulative Scree Plot",
+     ylab = "cumulative PVE",
+     xlab= "Prinicipal Components")
```



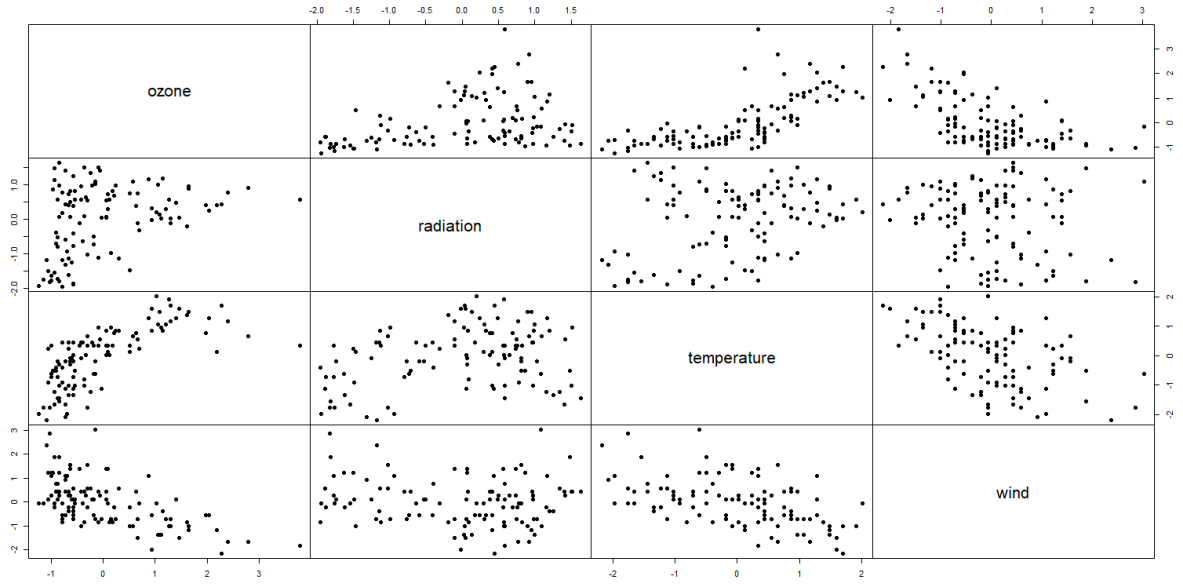
Clustering

The task of clustering is to find the homogenous groups inside the data. This homogeneity can be achieved by finding the distances between units. The units which are dissimilar has large distance among them. Hence finding the distance among units are the first step in finding the dissimilarity. We begin with applying hierarchical clustering methods to our data. Also there are different types of distances used in clustering. We analyze each common distance strategies separately.

1. Accessing Cluster Tendency:

It means that data has the capability of having clusters or not. Here we

```
> pairs(scaled.data,gap=0,pch=16)
```



We can see from scatterplots of the data that it has atleast the capability of having 2 cluster. But how many clusters should be in our data, it is the subject of next step.

Methods for assessing cluster tendency

i) Statistical methods:

Hopkins statistics:

The value of zero indicates the highly cluster data and 0.5 represents the unclustered data.

```
> hopkins(environmental,n=nrow(environmental)-1)
```

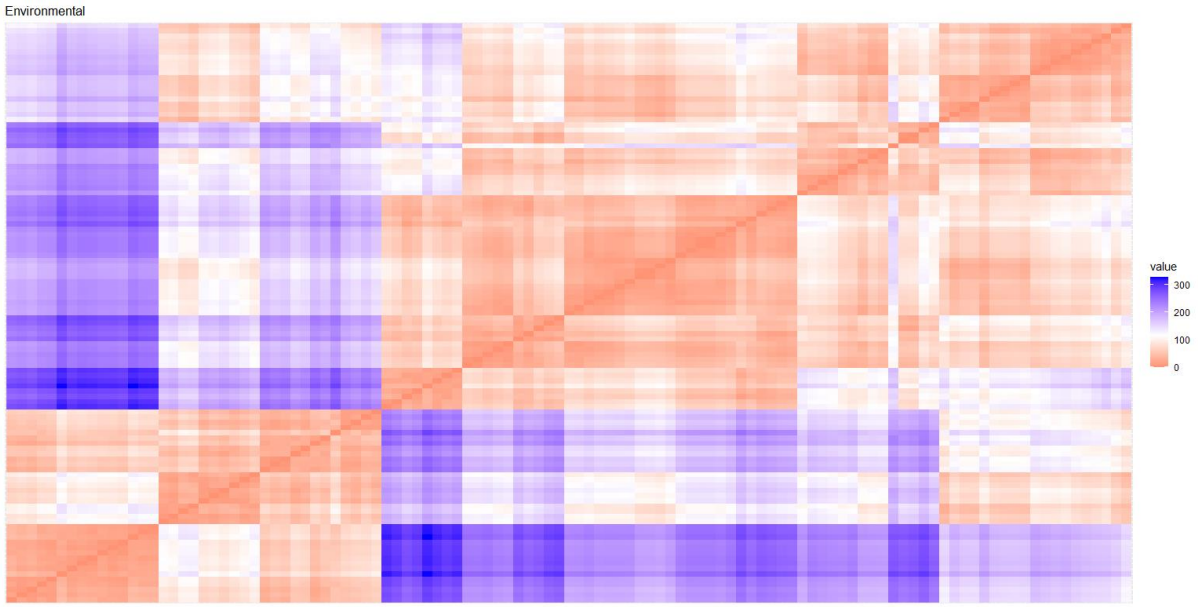
[1] 0.2675046

Although it is true that Hopkins statistics for environmental dataset is not approaching the value of zero which is indication of highly cluster data but the value of 0.26 is also enough for us to do clustering in this dataset.

ii) Visualization Method:

The VAT algorithm detects the clustering tendency in a visual form by counting the number of square shaped red blocks along the diagonal in a VAT image

```
> fviz_dist(dist(environmental),show_labels=FALSE)+labs(title="Environmental")
```



The color level is proportional to the value of the dissimilarity between observations:

red denotes high similarity (i.e. low dissimilarity);

blue denotes low similarity (i.e. high dissimilarity).

The dissimilarity matrix shows that there is the clustering structure in this dataset.

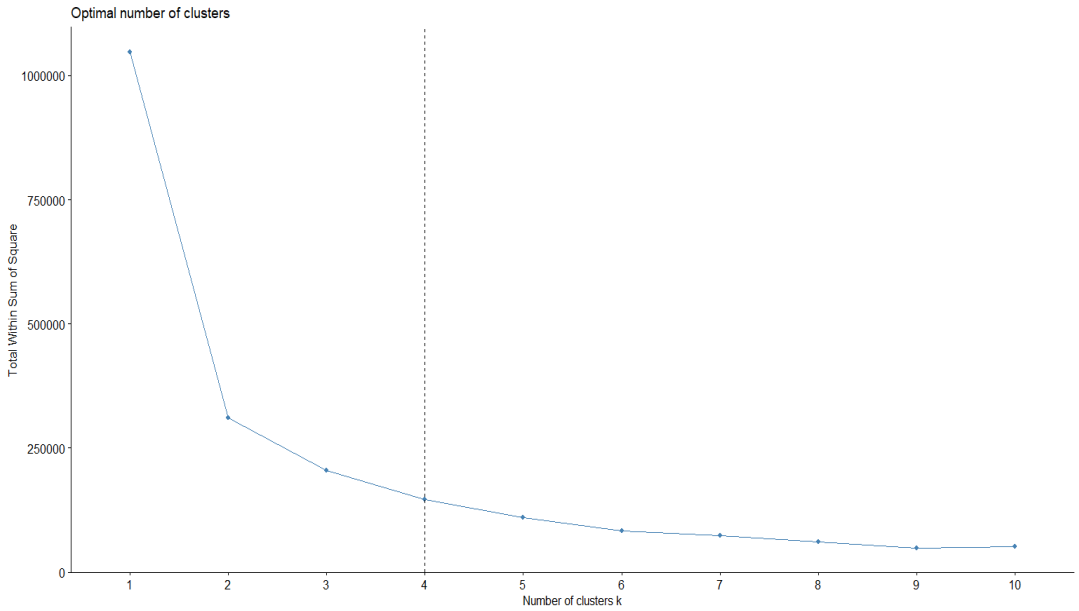
2. Optimal Number of Clusters:

Now we are sure that environmental dataset possessed the clustering structure, the next step is to find how many clusters enough for the best analysis of the data. Since there is no definitive method to determine the optimal number of clusters, so it's best to use a combination of different methods and to make a subjective decision based on the results.

i) Elbow Method:

The elbow method roughly measures the quality of a clustering by determining how compact clusters are in terms of within-cluster sum of squares (WSS), a classical measure of compactness or cohesion.

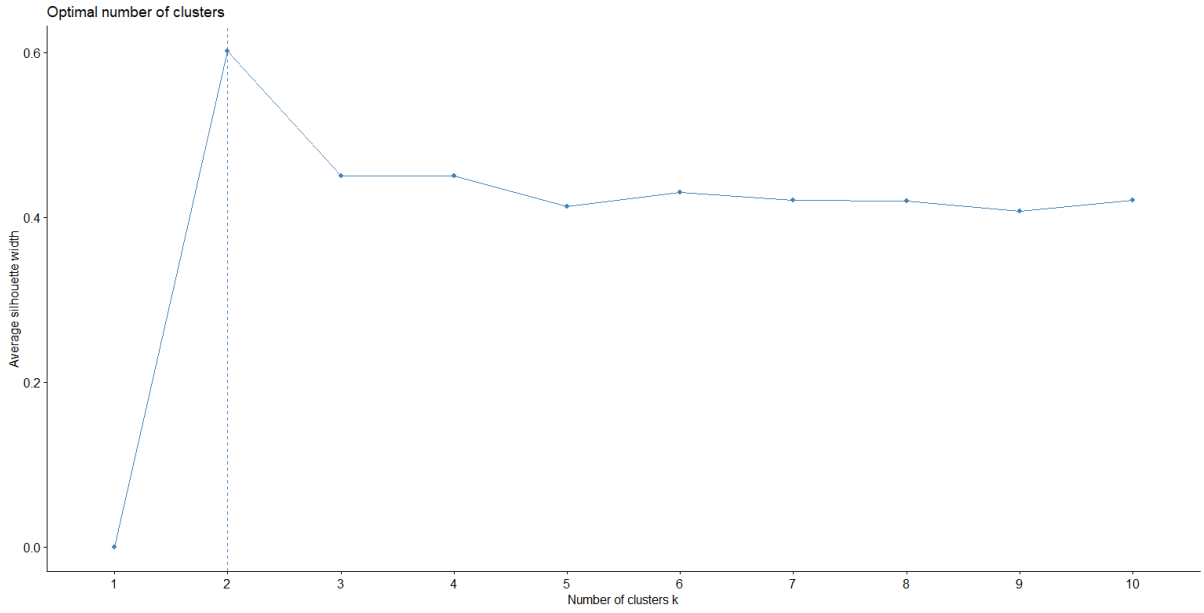
> fviz_nbclust(environmental,kmeans, method="wss")+geom_vline(xintercept=4,linetype=2)



It is evident from figure that total within sum of squares significantly drops if the data goes from cluster 1 to cluster 2. Onwards to cluster 4, there is only gradual change of total within sum of square.

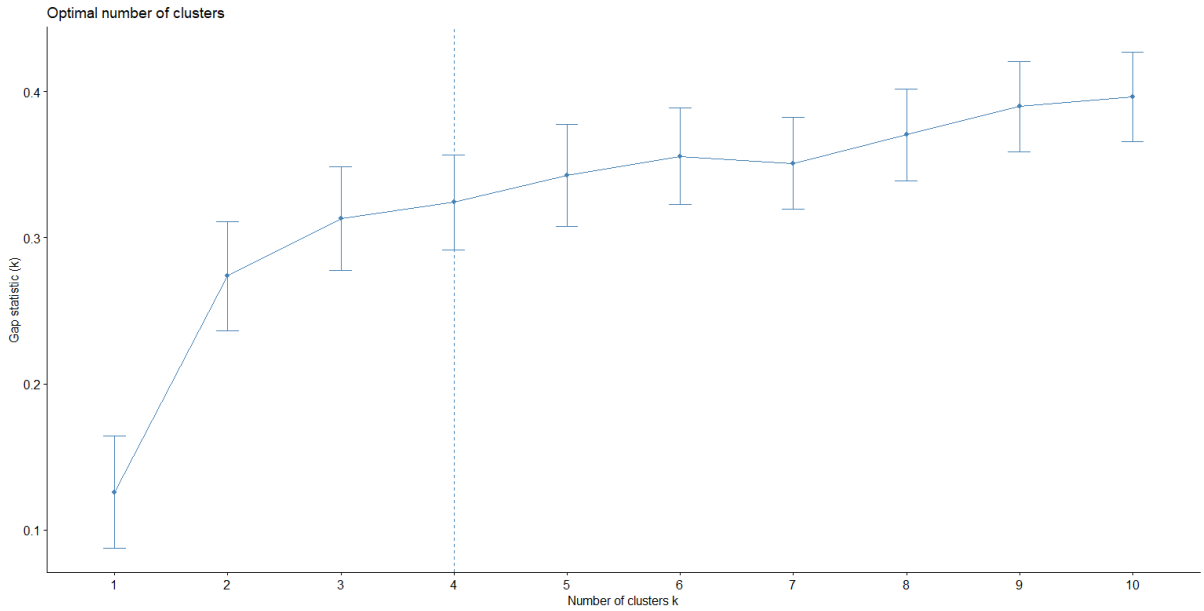
ii) Average Silhouette Method:

> fviz_nbclust(environmental,kmeans,method = "silhouette")



iii) Gap Statistics Method:

```
> fviz_nbclust(environmental,kmeans,nstart=50, method = "gap_stat",nboot = 50)
```



After applying three different methods, the two methods

The optimal number of clusters is 4 as two methods out of three gives this result. Now we do our analysis by considering this information.

3. Hierarchical Clustering (HC):

i) Agglomerative HC with Euclidean distance and wards linkage method:

```
> dist.euclidean <- dist(scaled.data,method="euclidean")
```

Here we find the Euclidean distance, the result is not easy ready for analysis. Therefore, need to convert it into matrix form.

```
> round(as.matrix(dist.euclidean)[1:15,1:15],2)
```

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	0.00	0.97	1.91	1.98	1.37	2.32	4.20	1.25	1.51	1.58	2.43	2.14	1.84	3.64	1.88
2	0.97	0.00	1.53	2.63	2.16	2.20	3.84	1.73	2.15	2.05	2.22	2.78	2.44	3.47	2.49
3	1.91	1.53	0.00	2.23	2.23	1.71	2.89	1.53	2.00	1.58	1.93	2.31	2.04	2.54	2.09
4	1.98	2.63	2.23	0.00	0.90	2.46	4.04	1.09	0.84	0.79	2.79	0.33	0.66	3.29	0.73
5	1.37	2.16	2.23	0.90	0.00	2.71	4.50	0.73	0.42	0.82	2.97	0.95	1.02	3.80	0.93
6	2.32	2.20	1.71	2.46	2.71	0.00	2.01	2.32	2.58	2.29	0.42	2.71	2.49	1.39	2.72
7	4.20	3.84	2.89	4.04	4.50	2.01	0.00	4.01	4.30	3.88	2.05	4.23	4.01	0.91	4.23
8	1.25	1.73	1.53	1.09	0.73	2.32	4.01	0.00	0.53	0.41	2.59	1.13	1.06	3.39	0.98
9	1.51	2.15	2.00	0.84	0.42	2.58	4.30	0.53	0.00	0.56	2.85	0.84	1.06	3.61	0.95
10	1.58	2.05	1.58	0.79	0.82	2.29	3.88	0.41	0.56	0.00	2.61	0.80	0.79	3.23	0.73
11	2.43	2.22	1.93	2.79	2.97	0.42	2.05	2.59	2.85	2.61	0.00	3.06	2.85	1.52	3.07
12	2.14	2.78	2.31	0.33	0.95	2.71	4.23	1.13	0.84	0.80	3.06	0.00	0.72	3.50	0.65
13	1.84	2.44	2.04	0.66	1.02	2.49	4.01	1.06	1.06	0.79	2.85	0.72	0.00	3.34	0.32
14	3.64	3.47	2.54	3.29	3.80	1.39	0.91	3.39	3.61	3.23	1.52	3.50	3.34	0.00	3.57
15	1.88	2.49	2.09	0.73	0.93	2.72	4.23	0.98	0.95	0.73	3.07	0.65	0.32	3.57	0.00

Now applying the method of hclust() for hierarchichal clustering.

```
> res.hc <- hclust(d=dist.euclidean,method = "ward.D2")
```

One way to measure how well the cluster tree generated by the hclust() function reflects our data is to compute the correlation between the cophenetic distances and the original distances generated by the dist() function. The cophenetic dissimilarity or cophenetic distance of two units is a measure of how similar those two units have to be in order to be grouped into the same cluster.

```
> res.coph <- cophenetic(res.hc)
> cor(res.coph,dist.euclidean)
[1] 0.6352151
```

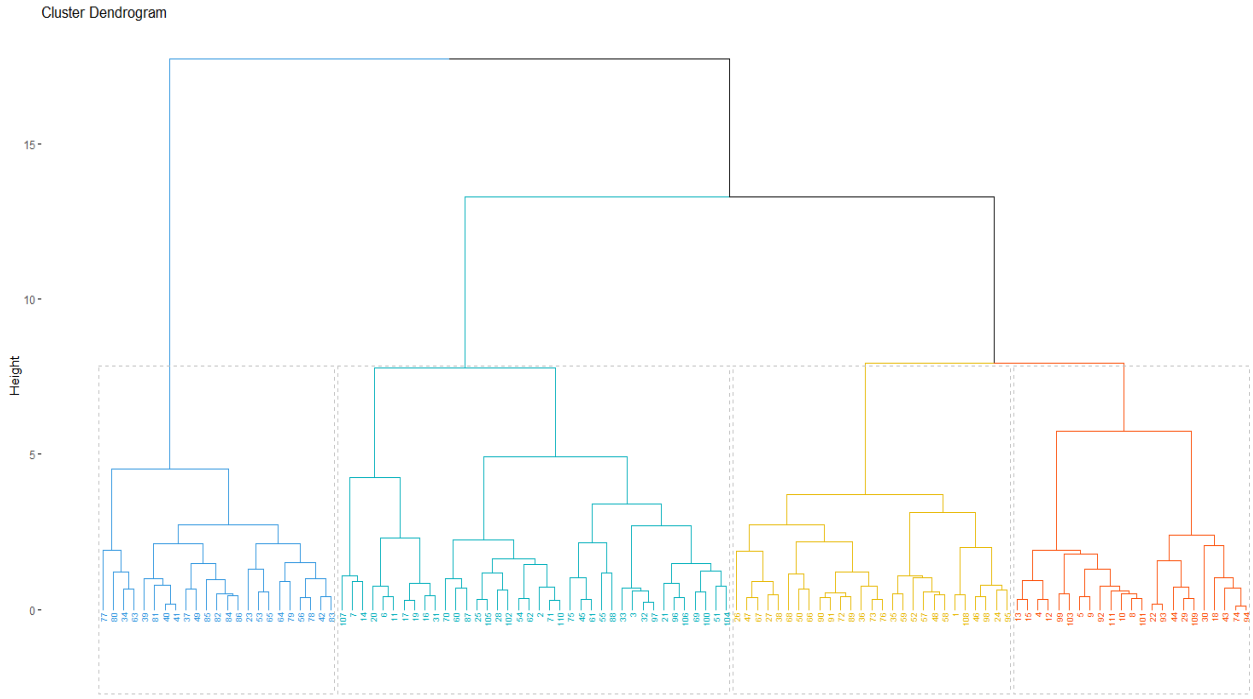
It indicates a good moderate value of correlation between cophenetic dissimilarities and original distance matrix.

We have already seen our data contains four optimal cluster. Therefore, we are cutting the tree into this number.

```
> grp <- cutree(res.hc,k=4)
```

The cluster assignment to first 6 units.

```
> head(grp)
[1] 1 2 2 3 3 2
```



In dendrogram, height represents the distance between units proportional to dissimilarity. As we can see the unit 13, 15 and 4,12 are close to each other, hence fuse together forming a branch. These both branch then merged together as their Euclidean distance is less with other units. This hierarchy continues until it become a single cluster.

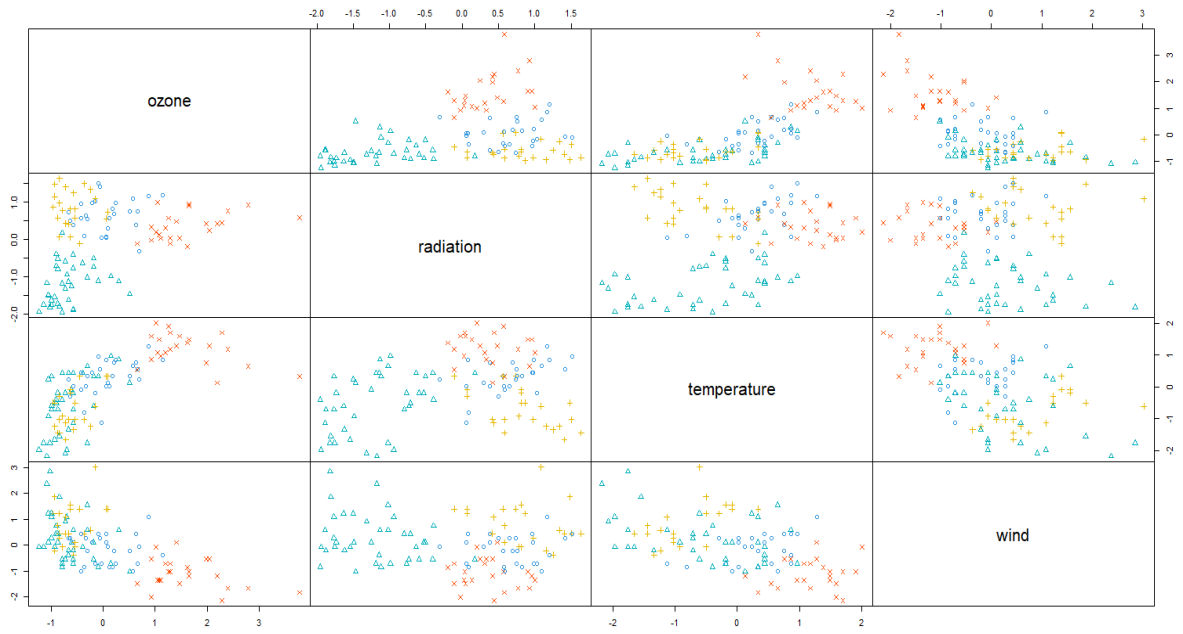
The number of units each cluster contain can be viewed as:

```
> table(grp)
1  2  3  4
27 38 23 23
```

Second cluster contains highest number of units.

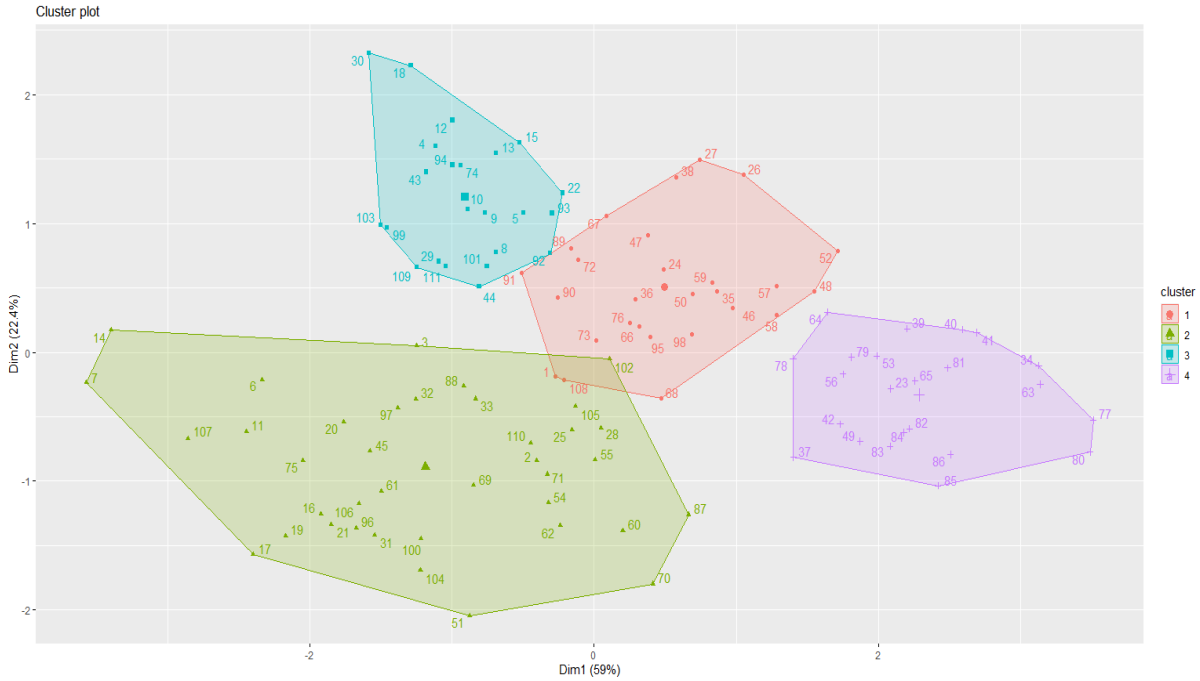
```
> pairs(scaled.data,gap=0,pch=grp,col=c("#2E95DF","#00AFBB","#E7B800","#FC4E07")[grp])
```

We can see now the scatterplots in the original space. The different colors and symbols shows the clusters made by hierarchichal clustering.



We can also visualize the results in the scatter plot (space) of the first 2 PCs (since d > 2)

```
> fviz_cluster(list(data=scaled.data,cluster=grp),
+              pallete=c("#2E95DF","#00AFBB","#E7B800","#FC4E07"),
+              ellipse.type = "convex",
+              repel=TRUE)
```



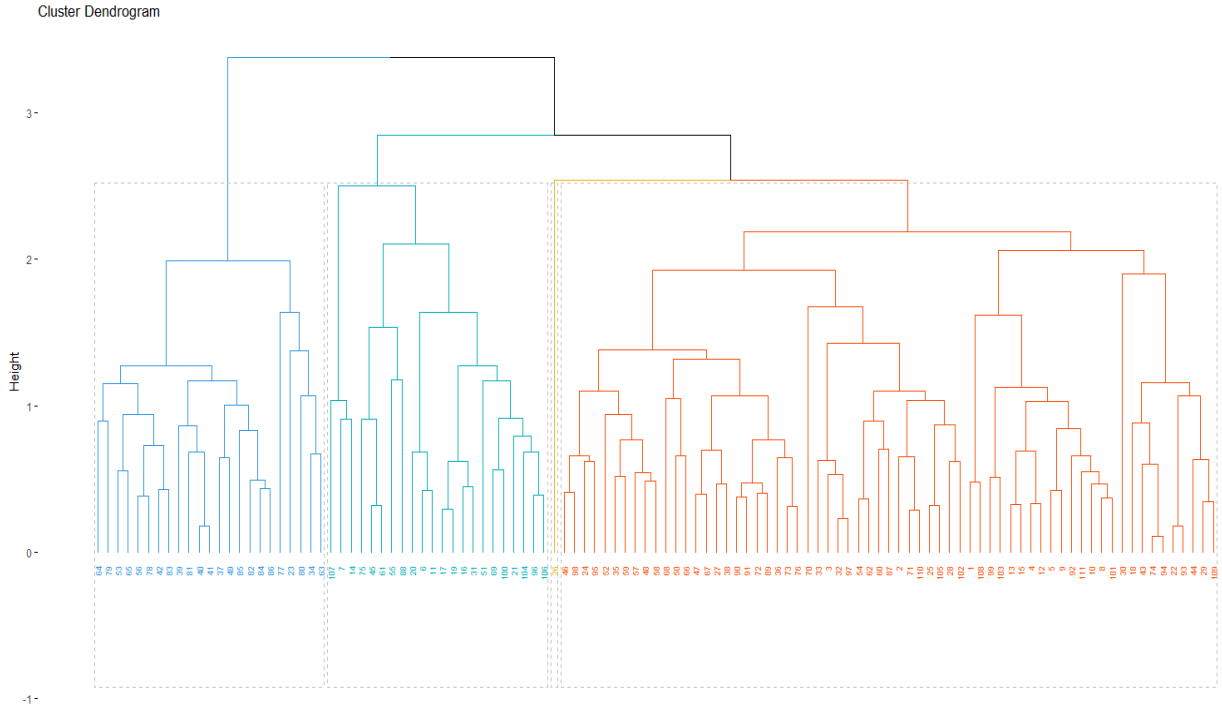
The agglomerative clustering using euclidean distance with wards linkage method clearly clusterized the data except very few points are misclassified.

ii) Agglomerative HC with euclidean and average linkage method

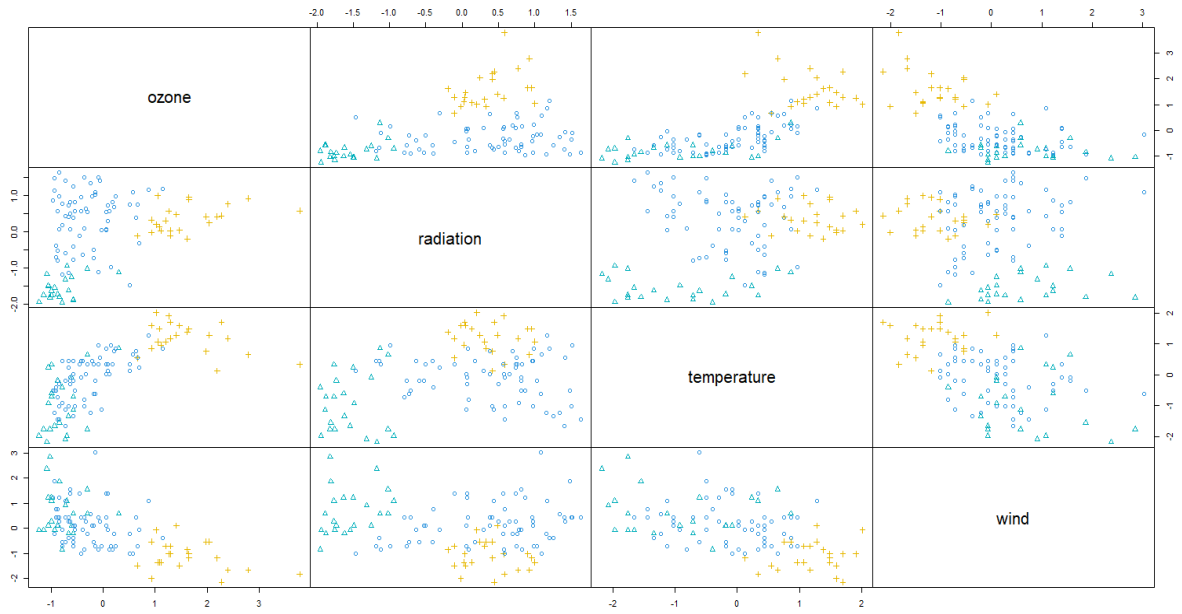
```
> res.hc2 <- hclust(d=dist.euclidean,method = "average")
> grp2 <- cutree(res.hc2,k=4)

> head(grp2)
[1] 1 1 1 1 1 2


> table(grp2)
 1  2  3  4
65 22 23  1
```

The last cluster has just one unit which average linkage method could not be group to other clusters.

```
> fviz_dend(res.hc2,k=4,cex = 0.5, k_colors =c("#2E95DF","#00AFBB","#E7B800","#FC4E07"),
        color_labels_by_k = TRUE, rect = TRUE)
```



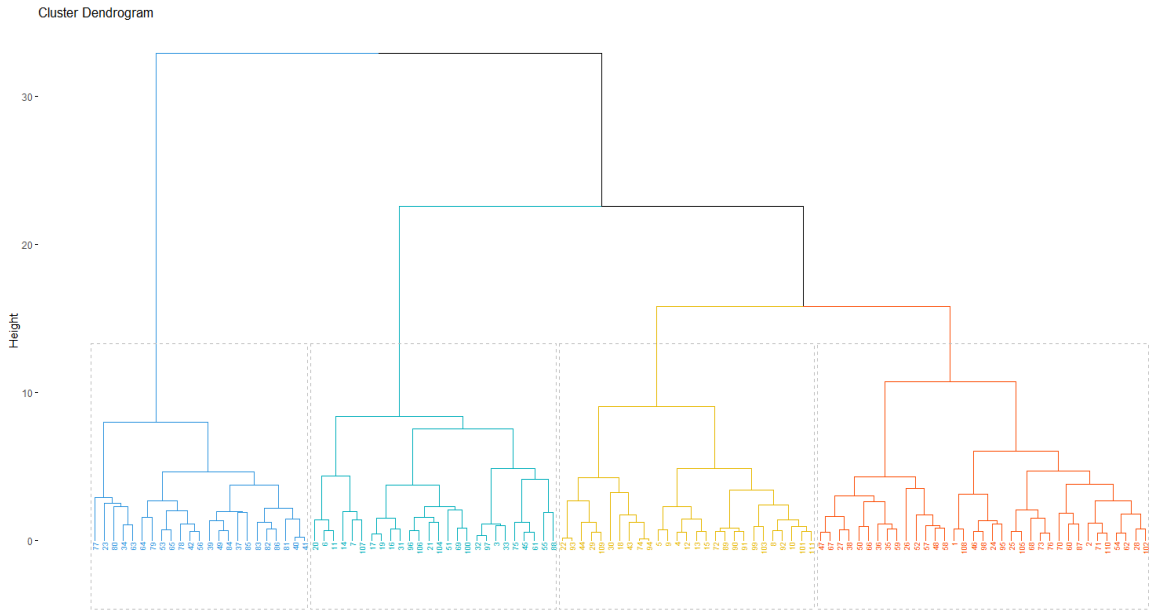

```
> res.coph2 <- cophenetic(res.hc2)
> cor(res.coph2,dist.euclidean)
[1] 0.6649177


> pairs(scaled.data, gap=0,pch=grp2,col=c("#2E95DF","#00AFBB","#E7B800")[grp2])
```



iii) Agglomerative HC based on Manhattan and wards linkage method

```
> dist.manhattan <- dist(scaled.data,method = "manhattan")

> round(as.matrix(dist.manhattan)[1:10,1:10],2)

    1    2    3    4    5    6    7    8    9   10
1  0.00 1.63 3.52 3.72 2.28 4.30 7.07 2.33 2.61 2.82
2  1.63 0.00 2.56 4.71 3.28 3.71 6.48 2.91 3.60 3.61
3  3.52 2.56 0.00 3.55 4.04 2.67 5.02 2.63 3.37 2.54
4  3.72 4.71 3.55 0.00 1.43 3.34 6.05 1.93 1.53 1.35
5  2.28 3.28 4.04 1.43 0.00 4.40 7.17 1.41 0.73 1.51
6  4.30 3.71 2.67 3.34 4.40 0.00 3.19 4.01 4.36 3.83
7  7.07 6.48 5.02 6.05 7.17 3.19 0.00 6.60 6.65 6.30
8  2.33 2.91 2.63 1.93 1.41 4.01 6.60 0.00 0.98 0.70
9  2.61 3.60 3.37 1.53 0.73 4.36 6.65 0.98 0.00 0.95
10 2.82 3.61 2.54 1.35 1.51 3.83 6.30 0.70 0.95 0.00

> res.hc.man <- hclust(d=dist.manhattan,method = "ward.D2")
> res.coph.man <- cophenetic(res.hc.man)
> cor(res.coph.man,dist.manhattan)
[1] 0.6358156
```

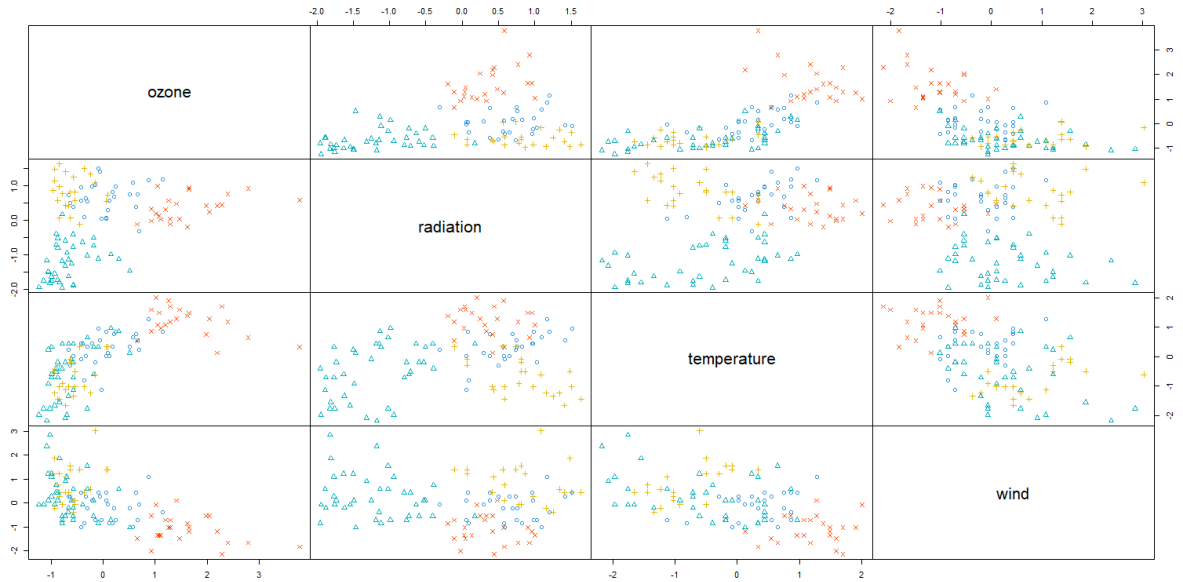
```
> grp <- cutree(res.hc.man,k=4)
> head(grp)
[1] 1 1 2 3 3 2

> fviz_dend(res.hc.man,k=4,cex = 0.5, k_colors =c("#2E95DF","#00AFBB","#E7B800","#FC4E07"),
+         color_labels_by_k = TRUE)
```




```
> table(grp)
 1  2  3  4
35 26 27 23
```

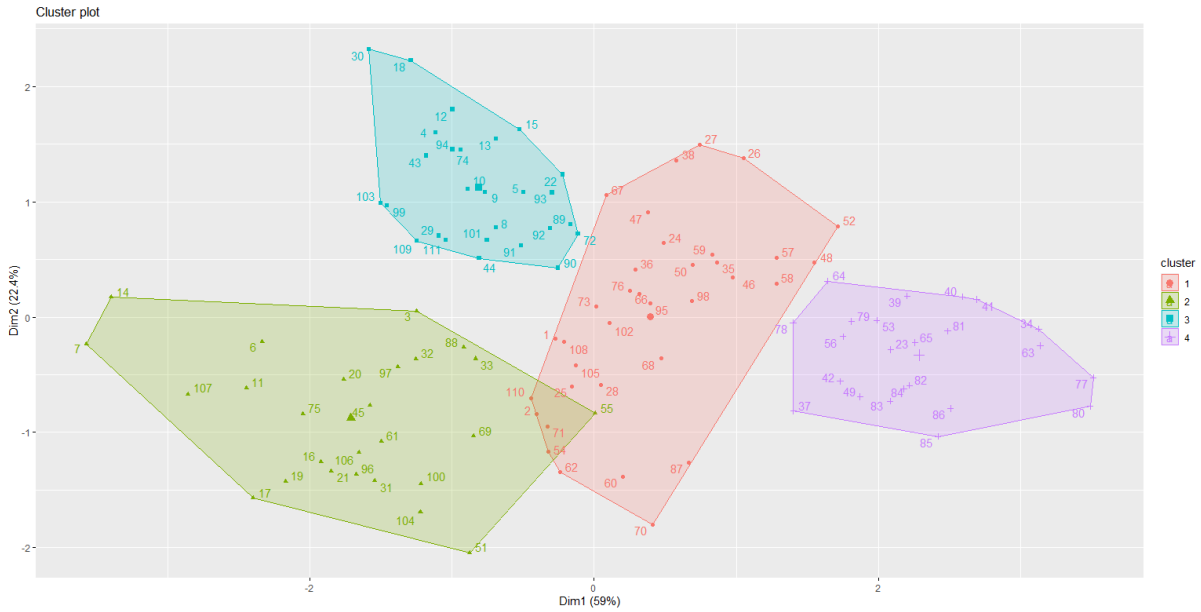
Except first cluster which contains high number of 35 units all other cluster has nearly equal number of units. There are still many units that we see in the case of Euclidean Distance cluster together, like unit 5 and 9, 4 and 12, 13 and 15.

```
> pairs(scaled.data,gap=0,pch=grp,col=c("#2E95DF","#00AFBB","#E7B800","#FC4E07")[grp])
```



This is depicting some scatterplots has better clustering capability while in some cases not. It is better to visualize it using fviz_cluster().

```
> fviz_cluster(list(data=environmental,cluster=grp),
+              pallete=c("#2E95DF","#00AFBB","#E7B800","#FC4E07"),
+              ellipse.type = "convex",
+              repel=TRUE)
```

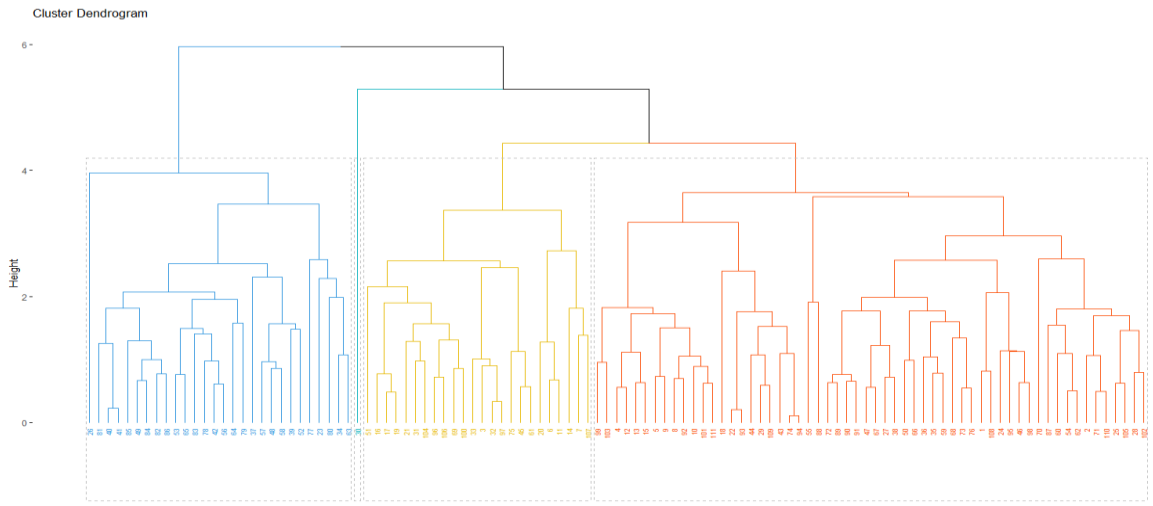


It shows four clear cluster excepts some units in two clusters are misclassify.

Agglomerative HC based on Manhattan and average linkage

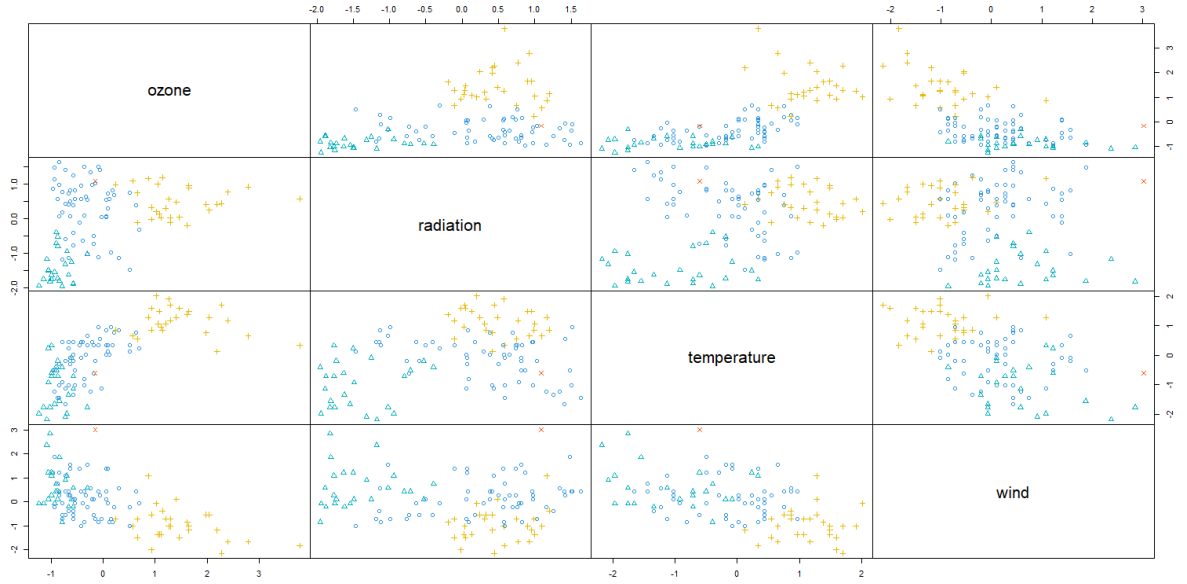
```
> res.hc.man2 <- hclust(d=dist.manhattan,method = "average")
> res.coph2 <- cophenetic(res.hc.man2)
> grp <- cutree(res.hc.man2,k=4)
> head(grp)
[1] 1 1 2 1 1 2
```

```
> fviz_dend(res.hc.man2,k=4,cex = 0.5, k_colors =c("#2E95DF","#00AFBB","#E7B800","#FC4E07"),
+           color_labels_by_k = TRUE, rect= TRUE)
```

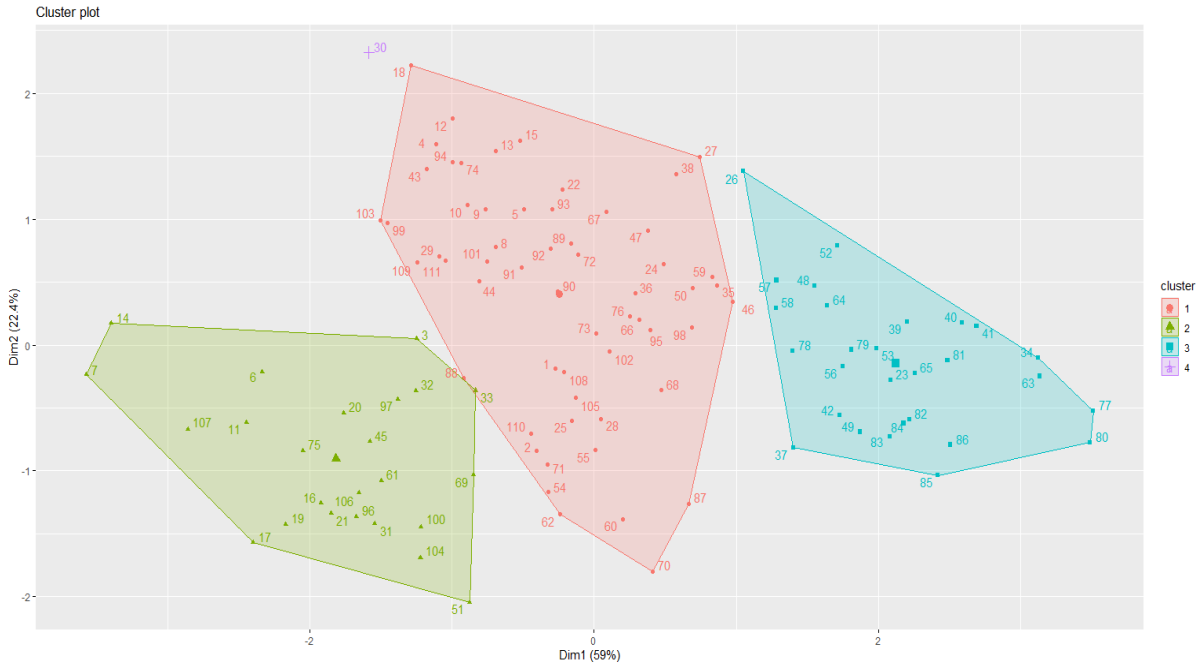


```
> table(grp)
 1  2  3  4
58 24 28  1
```

```
> pairs(environmental,gap=0,pch=grp,col=c("#2E95DF","#00AFBB","#E7B800","#FC4E07")[grp])
```




```
> fviz_cluster(list(data=environmental,cluster=grp),
+              pallete=c("#2E95DF","#00AFBB","#E7B800","#FC4E07"),
+              ellipse.type = "convex",
+              repel=TRUE)
```

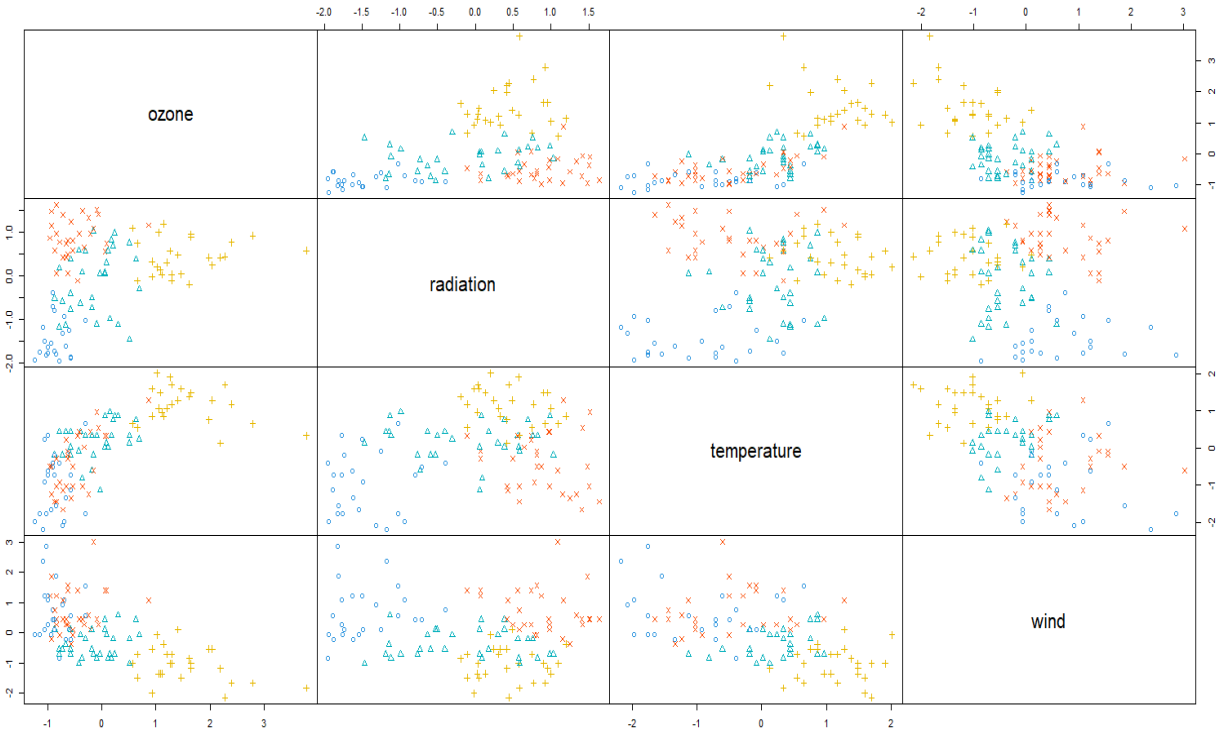


4. Partioning Clustering

i) K means

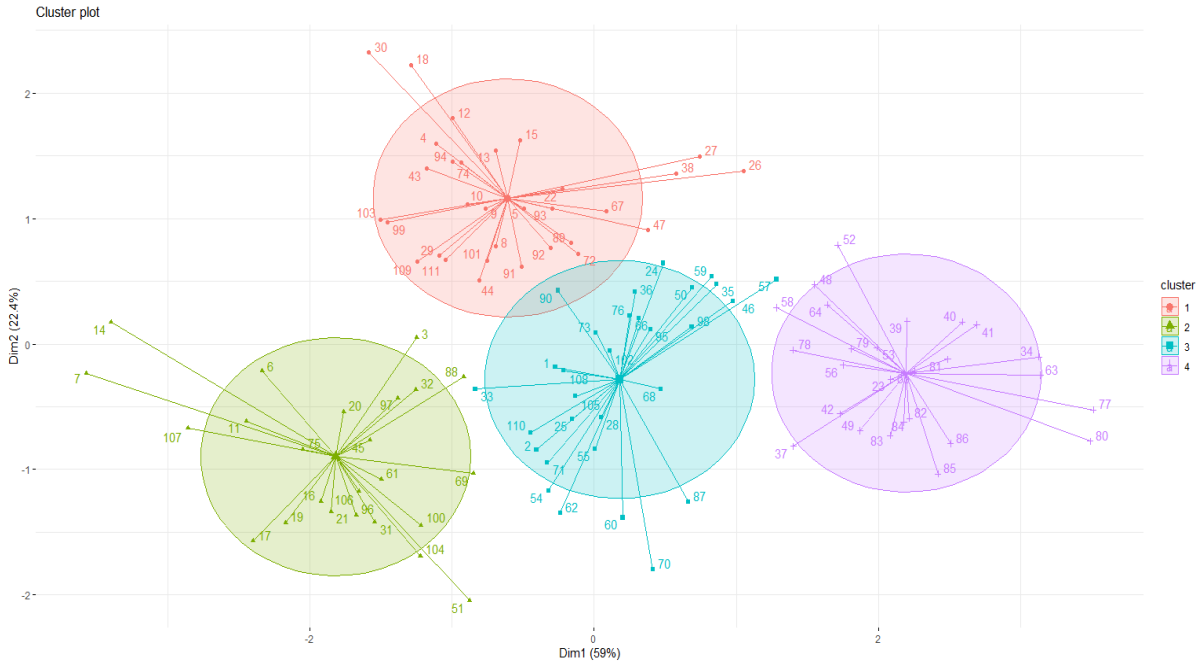
```
> km.res <- kmeans(scaled.data,4,nstart=25)
> c1 <- km.res$cluster
> km.res$size
[1] 31 24 30 26
```

```
> pairs(environmental,gap=0,pch=c1,col=c("#2E95DF","#00AFBB","#E7B800","#FC4E07")[c1])
```



We can also visualize the results in the scatter plot (space) of the first 2 PCs (since d > 2)

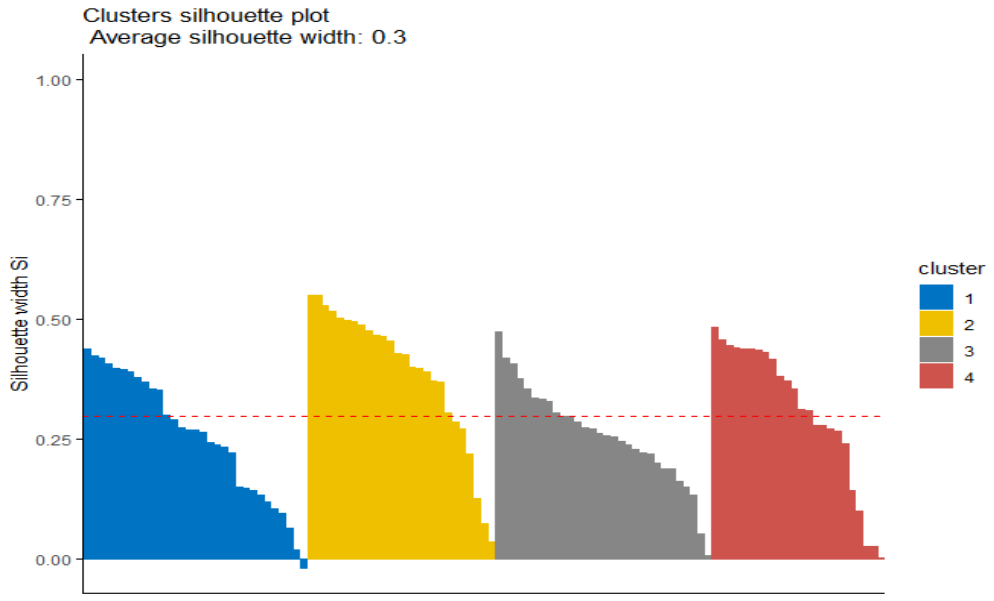
```
> fviz_cluster(km.res,data= scaled.data,pallete=c("#2E95DF","#00AFBB","#E7B800","#FC4E07"),
+          ellipse.type = "euclid", star.plot=TRUE,
+          repel = TRUE,ggtheme=theme_minimal())
```



Most of the units are correctly classified by K means but still cluster 1 and 3 are not well separated. There are some units which can be considered as outliers, like unit 7 and 14 here is the part of cluster 2.

Inter Cluster Validation:

```
> km.res <- eclust(scaled.data,"kmeans",k=4,nstart=50)
> fviz_silhouette(km.res,palette="jco",ggtheme=theme_classic())
  cluster size ave.sil.width
1       1   31          0.25
2       2   26          0.39
3       3   30          0.26
4       4   24          0.31
```



```
> sil[neg_sil_index,,drop=FALSE]
  cluster neighbor   sil_width
47       1        3       -0.01968863

> silinfo <- km.res$silinfo
> head(silinfo$widths[,1:3],10)
  cluster neighbor sil_width
43       1        3 0.4378999
18       1        3 0.4229450
94       1        3 0.4186369
74       1        3 0.4078874
13       1        3 0.3966955
15       1        3 0.3960033
12       1        3 0.3908122
4        1        3 0.3784156
10       1        3 0.3700032
99       1        4 0.3546167
```

Dunn Index:

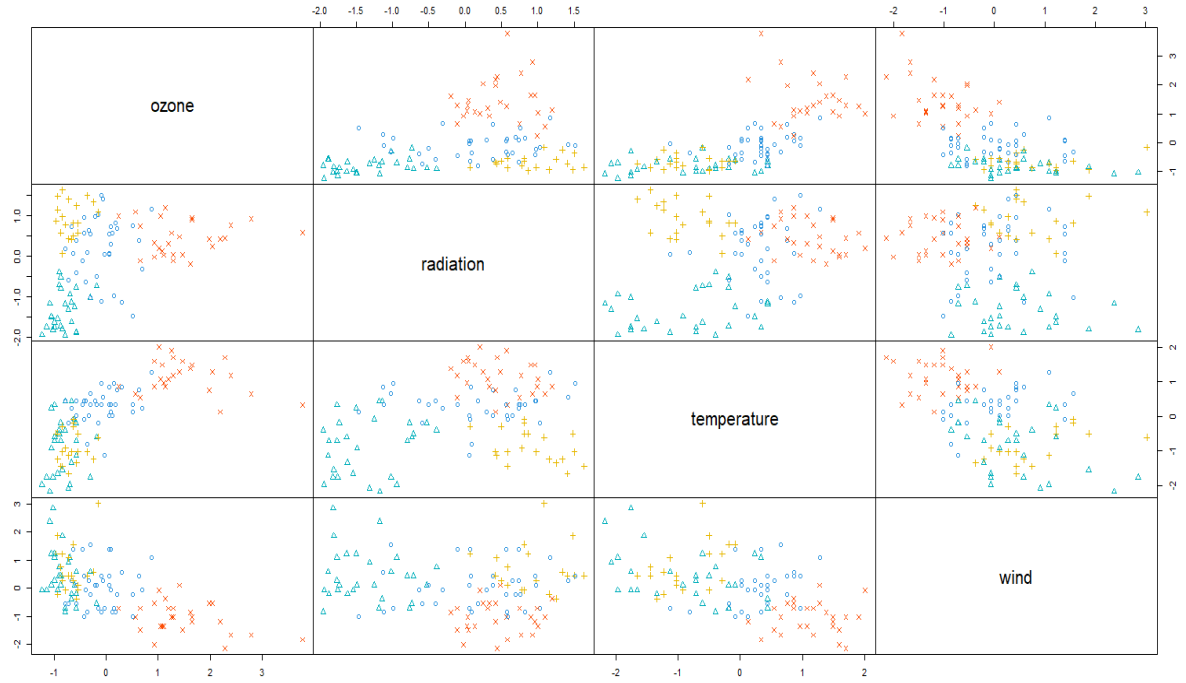
```
> km_stats <- cluster.stats(dist(scaled.data),km.res$cluster)
> km_stats$dunn
[1] 0.09552124

> km_stats$min.separation
[1] 0.3783685
> km_stats$max.diameter
[1] 3.961093
```

ii. K mediods

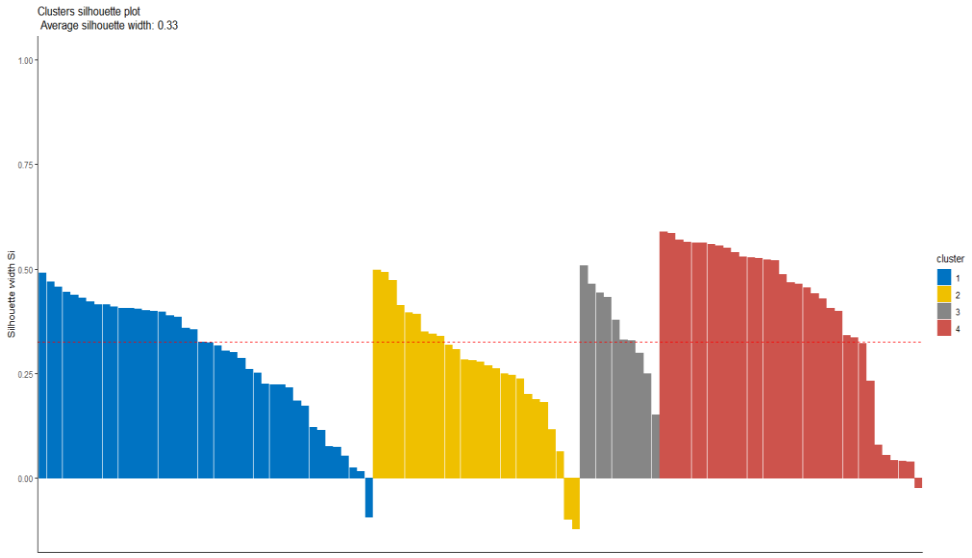
```
> pam.res <- pam(scaled.data,4)
> pam.res$medoids
        ozone  radiation temperature       wind
[1,]  0.08717705  0.2983819   0.1266748 -0.06707618
[2,] -1.05478820 -1.4898340  -0.9226465  0.10150018
[3,] -0.84442618  0.9785622  -1.0275787  0.27007655
[4,]  1.19909059  0.3093526   1.0710640 -0.71328559
```

Each medoid is a representative of the cluster and it's a point of the cluster with the lowest dissimilarity to all other points in the cluster. The values in the medoids represent the coordinates of the medoids in the feature space.

```
> pairs(scaled.data,gap = 0,pch=clust,col=c("#2E95DF","#00AFBB","#E7B800","#FC4E07")[clust])
```




```
> pam_stats <- cluster.stats(dist(scaled.data),pam.res$cluster)
> fviz_silhouette(pam.res,palette="jco",ggtheme=theme_classic())
  cluster size ave.sil.width
1       1   42          0.29
2       2   26          0.27
3       3   10          0.36
4       4   33          0.40
```



```
> head(silinfo$widths[,1:3],10)
   cluster neighbor sil_width
43       1        3  0.4378999
18       1        3  0.4229450
94       1        3  0.4186369
74       1        3  0.4078874
13       1        3  0.3966955
15       1        3  0.3960033
12       1        3  0.3908122
4        1        3  0.3784156
10       1        3  0.3700032
99       1        4 0.3546167

> sil <- pam.res$silinfo$widths[,1:3]
> neg_sil_index <-which(sil[,"sil_width"]<0)
> sil[neg_sil_index,,drop=FALSE]
   cluster neighbor   sil_width
3        1        2 -0.09268899
75       2        3 -0.09722765
31       2        3 -0.12088006
68       4        2 -0.02266437
```

Dunn Index:

```
> pam_stats$dunn
[1] 0.09544065

> pam_stats$min.separation # inter-cluster separation
[1] 0.4078935

> pam_stats$max.diameter # intra-cluster compactness
[1] 4.273792
```

Model Based Clustering:

We begin by fitting the Gaussian Finite Mixture Model using Mclust() function.

```
> mod <-Mclust(scaled.data,G=1:9,modelNames=NULL)
```

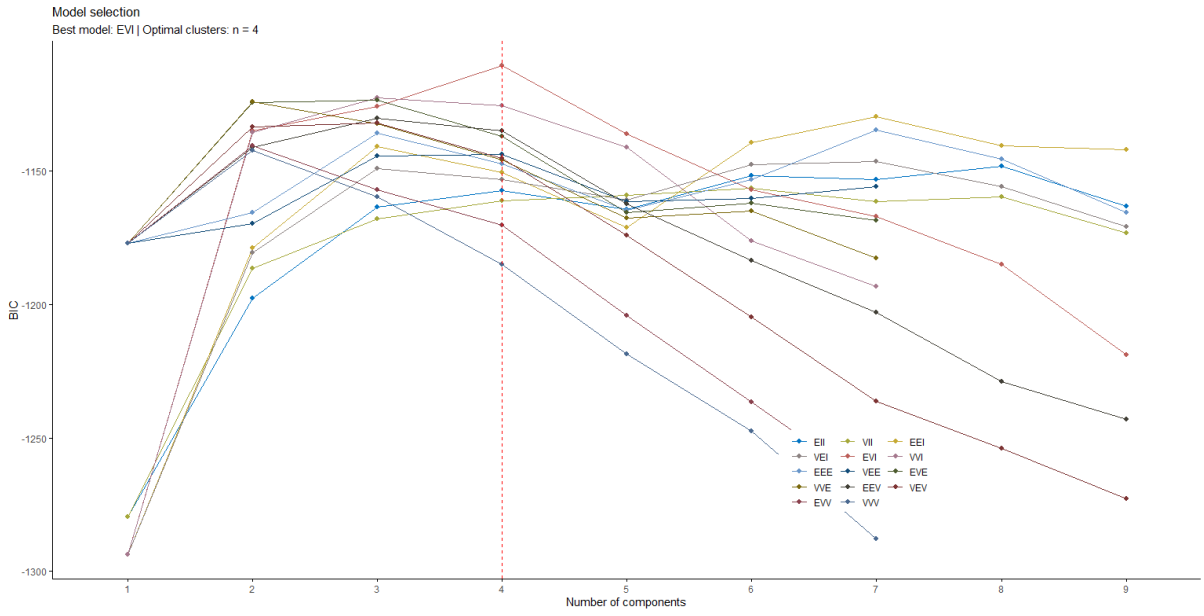
```
> summary(mod$BIC)
```

Best BIC values:

	EVI,4	VVI,3	EVE,3
BIC	-1110.501	-1122.2996	-1123.3632
BIC diff	0.000	-11.7983	-12.8619

In this output, the best BIC value is -1110.501, which corresponds to the EVI (same volume, varying shape, axis alinged) model with 4 clusters. The second best BIC value is -1122.2996 which corresponds to the EVE (equal volume, varying shape, same orientation) model with 2 clusters. And the third best BIC value -1123.3632 which corresponds to the EVE (same volume, varying shape, same orientation) model with 4 clusters.

```
>  fviz_mclust(mod, "BIC", palette = "jco")
```




```
> summary(mod)
```

Gaussian finite mixture model fitted by EM algorithm

Mclust EVI (diagonal, equal volume, varying shape) model with 4 components:

log-likelihood	n	df	BIC	ICL
-479.8982	111	32	-1110.501	-1131.055

Clustering table:

1	2	3	4
32	21	34	24

The clustering table shows the number of observations that are assigned to each of the 4 clusters. In this case, cluster 1 has 32 observations, cluster 2 has 21 observations, cluster 3 has 34 observations and cluster 4 has 24 observations.

```
> head(round(mod$z, 6), 15)
```

```
        [,1]     [,2]     [,3]    [,4]
 [1,] 0.000158 0.000006 0.999819 1.7e-05
 [2,] 0.045096 0.020605 0.934289 1.0e-05
 [3,] 0.003950 0.027694 0.968357 0.0e+00
 [4,] 0.000000 0.000000 1.000000 0.0e+00
 [5,] 0.000000 0.000000 1.000000 0.0e+00
 [6,] 0.000000 0.949062 0.050938 0.0e+00
 [7,] 0.000000 0.999901 0.000099 0.0e+00
 [8,] 0.000015 0.000000 0.999985 0.0e+00
 [9,] 0.000000 0.000000 1.000000 0.0e+00
[10,] 0.000003 0.000000 0.999997 0.0e+00
[11,] 0.000000 0.996486 0.003514 0.0e+00
[12,] 0.000000 0.000000 1.000000 0.0e+00
[13,] 0.000002 0.000000 0.999998 0.0e+00
[14,] 0.000000 0.999482 0.000518 0.0e+00
[15,] 0.000015 0.000000 0.999985 0.0e+00
```

mod$z is a matrix of posterior probabilities of each observation belonging to each of the clusters. The first observation has a very low probability 0.000171 of belonging to the first cluster and a very high probability 0.999797 of belonging to the third cluster which means it is likely that this observation belongs to the third cluster.

```
> head(mod$classification, 15)
```

```
 [1] 3  3  3  3  3 2  2  3  3  3   2   3  3 2 3
```