Network Event Correlation

Author: Christine Orosco
Objective: Determine validity of using Clustering, Network Diagrams, and Time-Series Anomaly Detection to identify anomalous computer network traffic.

Data Importing and Cleaning

```
test_data <- read.csv('~/clean_data.csv', header = TRUE, stringsAsFactors = FALSE)
```

Create Clustering Subset

Change Proto and IP to Numeric for clustering need all numeric except sav_index.

```
subset_clus <- test_data

sav_clus <- subset_clus
```

Convert Protocol to Factors

```
subset_clus$Proto <- as.factor(subset_clus$Proto)
subset_clus$Proto <- as.numeric(subset_clus$Proto)
```

Convert IPs to Numeric

```
subset_clus$Src_IP <- ip_to_numeric(subset_clus$Src_IP)
subset_clus$Dst_IP <- ip_to_numeric(subset_clus$Dst_IP)
```

Scale the dataframe

```
subset_clus <- scale(subset_clus)
```

Create Test Set for Network Diagrams

Change epoch to UTC, for node_edge, time series, and anomalzing

```
test_data$UTC <- anytime(test_data$UTC, asUTC=TRUE )
```

Create Subset

```
test.ts <- test_data[,c(1,3,5)]
test.ts[order(test.ts$UTC),]
```

```
                   UTC        Src_IP        Dst_IP
1   2020-04-01 11:39:53 10.100.10.123  192.168.1.10
2   2020-04-01 11:54:53 10.100.10.123  192.168.1.10
3   2020-04-01 12:09:53  192.168.1.10 10.100.10.123
4   2020-04-01 12:24:53 10.100.10.123  192.168.1.10
5   2020-04-01 12:39:53  192.168.1.10 10.100.10.123
6   2020-04-01 12:54:53 10.100.10.123  192.168.1.10
7   2020-04-01 13:09:53  192.168.1.10 10.100.10.123
8   2020-04-01 13:24:53 10.100.10.123  192.168.1.10
9   2020-04-01 13:39:53  192.168.1.10 10.100.10.123
10  2020-04-01 13:54:53 10.100.10.123  192.168.1.10
11  2020-04-01 14:09:53  192.168.1.10 10.100.10.123
12  2020-04-01 14:24:53 10.100.10.123  192.168.1.10
13  2020-04-01 14:39:53  192.168.1.10 10.100.10.123
14  2020-04-01 14:54:53 10.100.10.123  192.168.1.10
15  2020-04-01 15:09:53 10.101.10.124  192.168.1.10
16  2020-04-01 15:24:53  192.168.1.11 10.101.10.124
17  2020-04-01 15:39:53  192.168.1.10 10.101.10.124
18  2020-04-01 15:54:53 10.100.10.123  192.168.1.10
19  2020-04-01 16:09:53  192.168.1.10 10.101.10.124
20  2020-04-01 16:24:53  192.168.1.11  192.168.1.10
21  2020-04-01 16:39:53  192.168.1.11  192.168.1.10
22  2020-04-01 16:54:53  192.168.1.11  192.168.1.10
23  2020-04-01 17:09:53  192.168.1.10 10.101.10.124
24  2020-04-01 17:24:53  192.168.1.11 10.101.10.124
25  2020-04-01 17:39:53 10.101.10.125  192.168.1.12
26  2020-04-01 17:54:53  192.168.1.11 10.101.10.124
27  2020-04-01 18:09:53  192.168.1.11 10.101.10.124
28  2020-04-01 18:24:53  192.168.1.12 10.101.10.125
29  2020-04-01 18:39:53 10.101.10.125  192.168.1.12
30  2020-04-01 18:54:53 10.100.10.123  192.168.1.10
31  2020-04-01 19:09:53 10.101.10.125  192.168.1.12
32  2020-04-01 19:24:53  192.168.1.12 10.101.10.125
33  2020-04-01 19:39:53 10.101.10.125  192.168.1.12
```

34	2020-04-01	19:54:53	192.168.1.12	10.101.10.125
35	2020-04-01	20:09:53	10.101.10.125	192.168.1.12
36	2020-04-01	20:24:53	192.168.1.12	10.101.10.125
37	2020-04-01	20:39:53	192.168.1.12	10.101.10.125
38	2020-04-01	20:54:53	192.168.1.12	10.101.10.125
39	2020-04-01	21:09:53	10.101.10.125	192.168.1.12
40	2020-04-01	21:24:53	10.100.10.123	192.168.1.10
41	2020-04-01	21:39:53	10.100.10.123	192.168.1.12
42	2020-04-01	21:54:53	192.168.1.13	10.100.10.126
43	2020-04-01	22:09:53	192.168.1.13	10.100.10.126
44	2020-04-01	22:24:53	192.168.1.13	10.100.10.126
45	2020-04-01	22:39:53	192.168.1.13	10.100.10.126
46	2020-04-01	22:54:53	192.168.1.15	192.168.1.13
47	2020-04-01	23:09:53	10.100.10.123	192.168.1.10
48	2020-04-01	23:24:53	10.100.10.123	10.100.10.126
49	2020-04-01	23:39:53	192.168.1.14	10.100.10.126
50	2020-04-01	23:54:53	192.168.1.13	10.100.10.126
51	2020-04-02	00:09:53	192.168.1.13	10.100.10.126
52	2020-04-02	00:24:53	192.168.1.13	10.100.10.126
53	2020-04-02	00:39:53	192.168.1.12	10.101.10.125
54	2020-04-02	00:54:53	10.100.10.126	192.168.1.10
55	2020-04-02	01:09:53	192.168.1.13	10.100.10.126
56	2020-04-02	01:24:53	192.168.1.15	10.100.10.126
57	2020-04-02	01:39:53	10.100.10.123	192.168.1.10
58	2020-04-02	01:54:53	192.168.1.14	192.168.1.10
59	2020-04-02	02:09:53	192.168.1.14	192.168.1.14
60	2020-04-02	02:24:53	192.168.1.14	10.100.10.126
61	2020-04-02	02:39:53	192.168.1.14	10.100.10.126
62	2020-04-02	02:54:53	192.168.1.14	10.100.10.126
63	2020-04-02	03:09:53	192.168.1.14	10.100.10.126
64	2020-04-02	03:24:53	192.168.1.14	10.100.10.126
65	2020-04-02	03:39:53	192.168.1.14	10.100.10.126
66	2020-04-02	03:54:53	192.168.1.14	10.100.10.126
67	2020-04-02	04:09:53	192.168.1.14	10.100.10.126
68	2020-04-02	04:24:53	10.100.10.126	192.168.1.15
69	2020-04-02	04:39:53	192.168.1.14	10.100.10.126
70	2020-04-02	04:54:53	192.168.1.14	10.100.10.126
71	2020-04-02	05:09:53	10.100.10.126	192.168.1.14
72	2020-04-02	05:24:53	10.101.10.125	192.168.1.12
73	2020-04-02	05:39:53	192.168.1.14	10.100.10.126
74	2020-04-02	05:54:53	10.100.10.126	192.168.1.14
75	2020-04-02	06:09:53	192.168.200.1	192.168.200.1
76	2020-04-02	06:24:53	10.100.1.123	192.168.200.1

```
77  2020-04-02 06:39:53 10.100.10.126  192.168.1.14
78  2020-04-02 06:54:53 10.100.10.126  192.168.1.14
79  2020-04-02 07:09:53  192.168.1.14 10.100.10.126
80  2020-04-02 07:24:53 10.100.10.123  192.168.1.10
81  2020-04-02 07:39:53  192.168.1.14 10.100.10.126
82  2020-04-02 07:54:53 10.100.10.123  192.168.1.10
83  2020-04-02 08:09:53  192.168.1.14 10.100.10.126
84  2020-04-02 08:24:53  192.168.1.14 10.100.10.126
85  2020-04-02 08:39:53 10.100.10.126  192.168.1.14
86  2020-04-02 08:54:53  10.100.1.123 192.168.200.1
87  2020-04-02 09:09:53  192.168.1.12 10.101.10.125
88  2020-04-02 09:24:53  192.168.1.12  192.168.1.12
89  2020-04-02 09:39:53 10.100.10.123  192.168.1.12
90  2020-04-02 09:54:53 10.100.10.126  192.168.1.13
91  2020-04-02 10:09:53 10.100.10.126  192.168.1.13
92  2020-04-02 10:24:53 10.100.10.126  192.168.1.13
93  2020-04-02 10:39:53 10.100.10.126  192.168.1.13
94  2020-04-02 10:54:53  192.168.1.15  192.168.1.13
95  2020-04-02 11:09:53  192.168.1.14 10.100.10.126
96  2020-04-02 11:24:53 10.101.10.125  192.168.1.12
97  2020-04-02 11:39:53 10.100.10.126  192.168.1.14
98  2020-04-02 11:54:53 10.100.10.126  192.168.1.14
99  2020-04-02 12:09:53  192.168.1.15 10.100.10.126
100 2020-04-02 12:24:53  192.168.1.14  192.168.1.14
101 2020-04-02 12:39:53  192.168.1.14 10.100.10.126
102 2020-04-02 12:54:53 10.100.10.126  192.168.1.14
103 2020-04-02 13:09:53 10.100.10.126  192.168.1.14
104 2020-04-02 13:24:53 10.100.10.126  192.168.1.14
105 2020-04-02 13:39:53 10.100.10.126  192.168.1.14
106 2020-04-02 13:54:53 10.100.10.126  192.168.1.14
107 2020-04-02 14:09:53  192.168.1.12 10.101.10.125
108 2020-04-02 14:24:53 10.100.10.126  192.168.1.14
109 2020-04-02 14:39:53  192.168.1.14 10.100.10.126
110 2020-04-02 14:54:53  192.168.1.14 10.100.10.126
111 2020-04-02 15:09:53 10.100.10.123  192.168.1.12
112 2020-04-02 15:24:53 192.168.200.1 192.168.200.1
113 2020-04-02 15:39:53 192.168.200.1  10.100.1.123
114 2020-04-02 15:54:53 10.100.10.123  192.168.1.12
115 2020-04-02 16:09:53 10.100.10.126  192.168.1.13
116 2020-04-02 16:24:53 10.100.10.126  192.168.1.13
117 2020-04-02 16:39:53 10.100.10.126  192.168.1.13
118 2020-04-02 16:54:53 10.100.10.126  192.168.1.13
119 2020-04-02 17:09:53  192.168.1.14 10.100.10.126
```

```
120 2020-04-02 17:24:53  192.168.1.14 10.100.10.126
121 2020-04-02 17:39:53  192.168.1.12  192.168.1.12
122 2020-04-02 17:54:53 10.100.10.126  192.168.1.14
123 2020-04-02 18:09:53 10.100.10.126  192.168.1.14
124 2020-04-02 18:24:53 10.100.10.126  192.168.1.15
125 2020-04-02 18:39:53 10.100.10.123 10.100.10.126
126 2020-04-02 18:54:53 10.100.10.126  192.168.1.14
127 2020-04-02 19:09:53  192.168.1.14 10.100.10.126
128 2020-04-02 19:24:53 10.100.10.126  192.168.1.14
129 2020-04-02 19:39:53  192.168.1.14 10.100.10.126
130 2020-04-02 19:54:53 10.100.10.126  192.168.1.14
131 2020-04-02 20:09:53  192.168.1.14 10.100.10.126
132 2020-04-02 20:24:53 10.100.10.126  192.168.1.14
133 2020-04-02 20:39:53 10.100.10.126  192.168.1.14
134 2020-04-02 20:54:53  192.168.1.14 10.100.10.126
135 2020-04-02 21:09:53 192.168.200.1  10.100.1.123
136 2020-04-02 21:24:53  10.100.1.123   10.200.1.15
137 2020-04-02 21:39:53  10.100.1.123 192.168.200.1
138 2020-04-02 21:54:53 192.168.200.1  10.100.1.123
139 2020-04-02 22:09:53 192.168.200.1  10.100.1.123
140 2020-04-02 22:24:53  10.100.1.123   10.200.1.15
141 2020-04-02 22:39:53  10.100.1.123   10.200.1.15
142 2020-04-02 22:54:53 192.168.200.1  10.100.1.123
143 2020-04-02 23:09:53 192.168.200.1  10.100.1.123
144 2020-04-02 23:24:53  10.100.1.123   10.200.1.15
145 2020-04-02 23:39:53  10.100.1.123   10.200.1.15
146 2020-04-02 23:54:53 192.168.200.1  10.100.1.123
147 2020-04-03 00:09:53 192.168.200.1  10.100.1.123
148 2020-04-03 00:24:53  10.100.1.123   10.200.1.15
149 2020-04-03 00:39:53 192.168.200.1  10.100.1.123
```

```
test.ts$Src_IP <- as.factor(test.ts$Src_IP)
test.ts$Dst_IP <- as.factor(test.ts$Dst_IP)
```

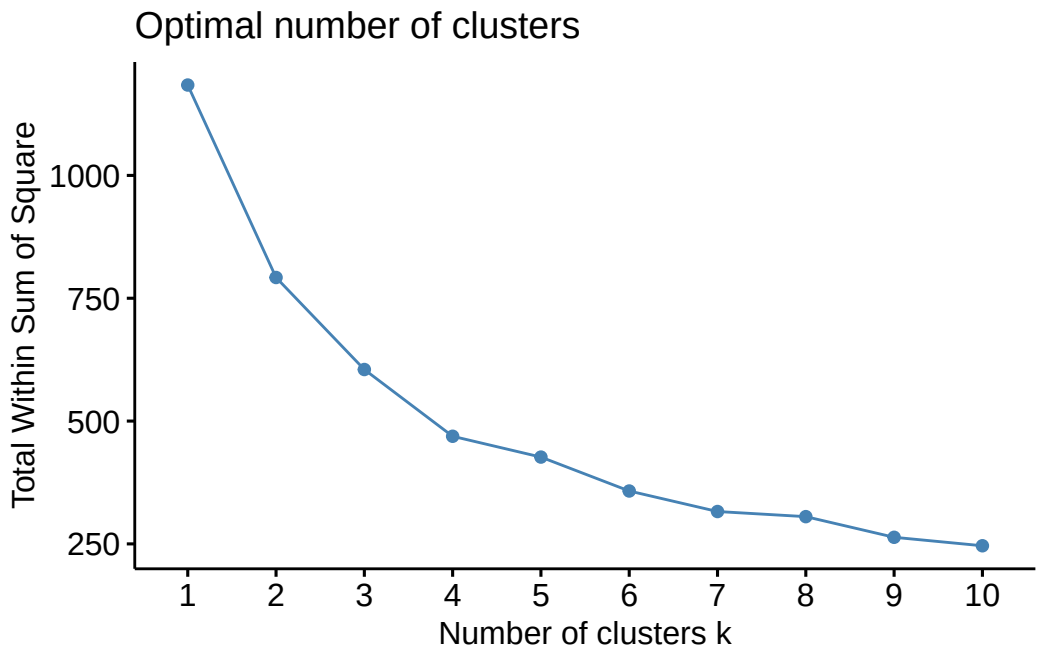
Create Clusters

K-means using WSS Method

```
k_clus <- fviz_nbclust(subset_clus, FUNcluster = kmeans,
                       nstart=2, nboot = 50,  method = "wss")
```

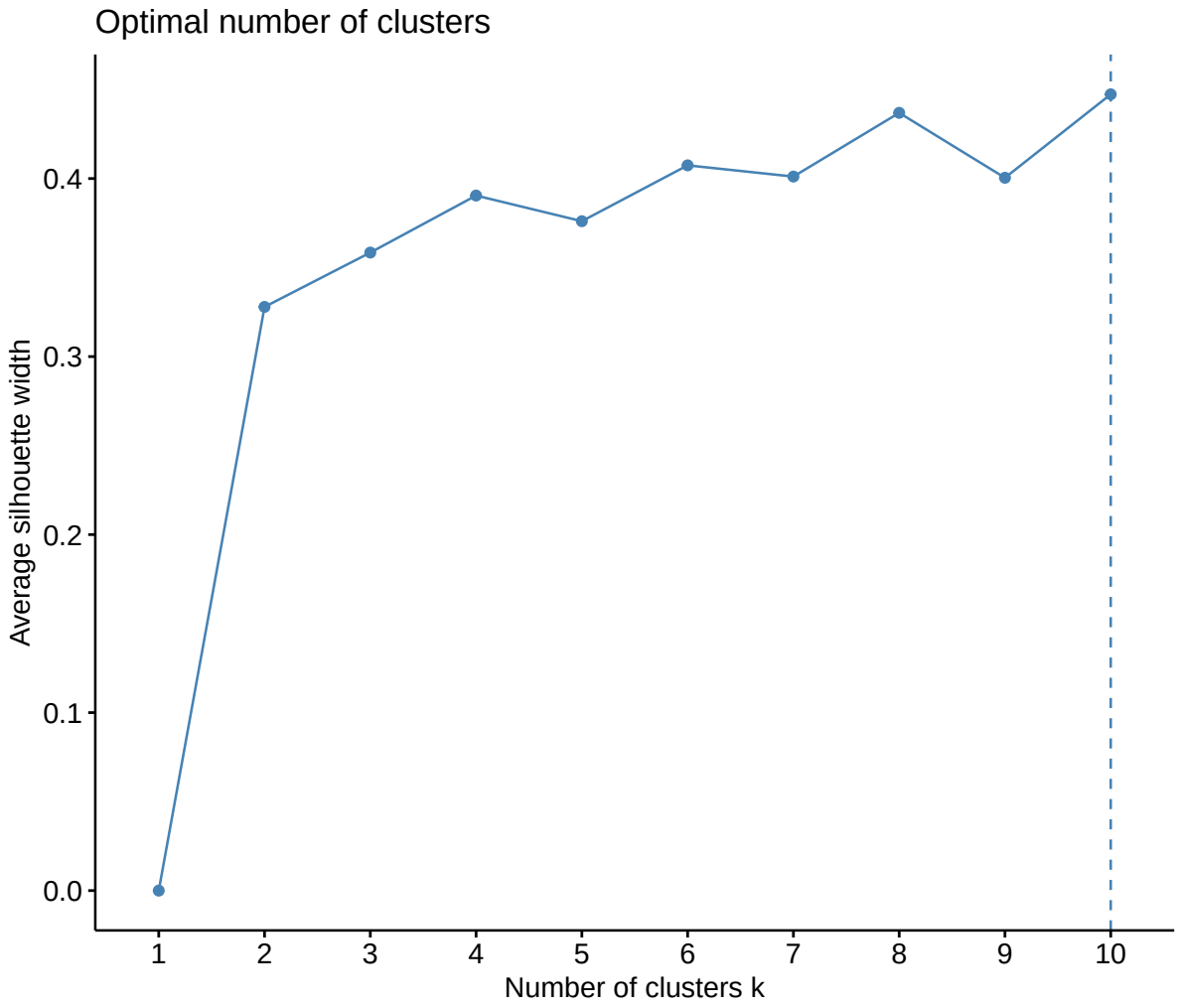
Display optimal nbr of clusters

```
k_clus
```



K-means Using Silhouette Method

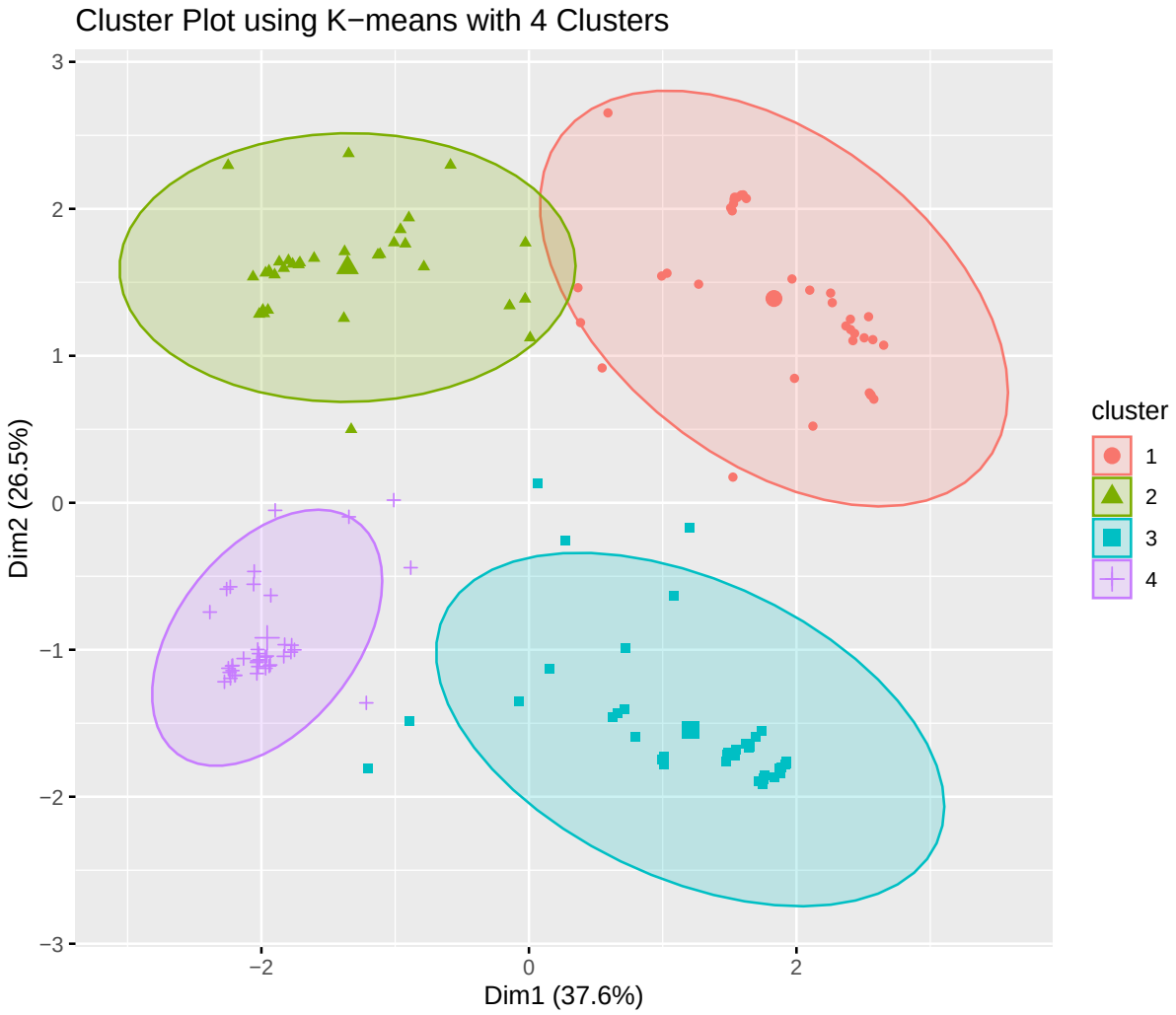
```
k_clus1 <- fviz_nbclust(subset_clus, FUNcluster = kmeans, nstart=2,
                        nboot = 50,method = "silhouette")
k_clus1
```



Plot K-means

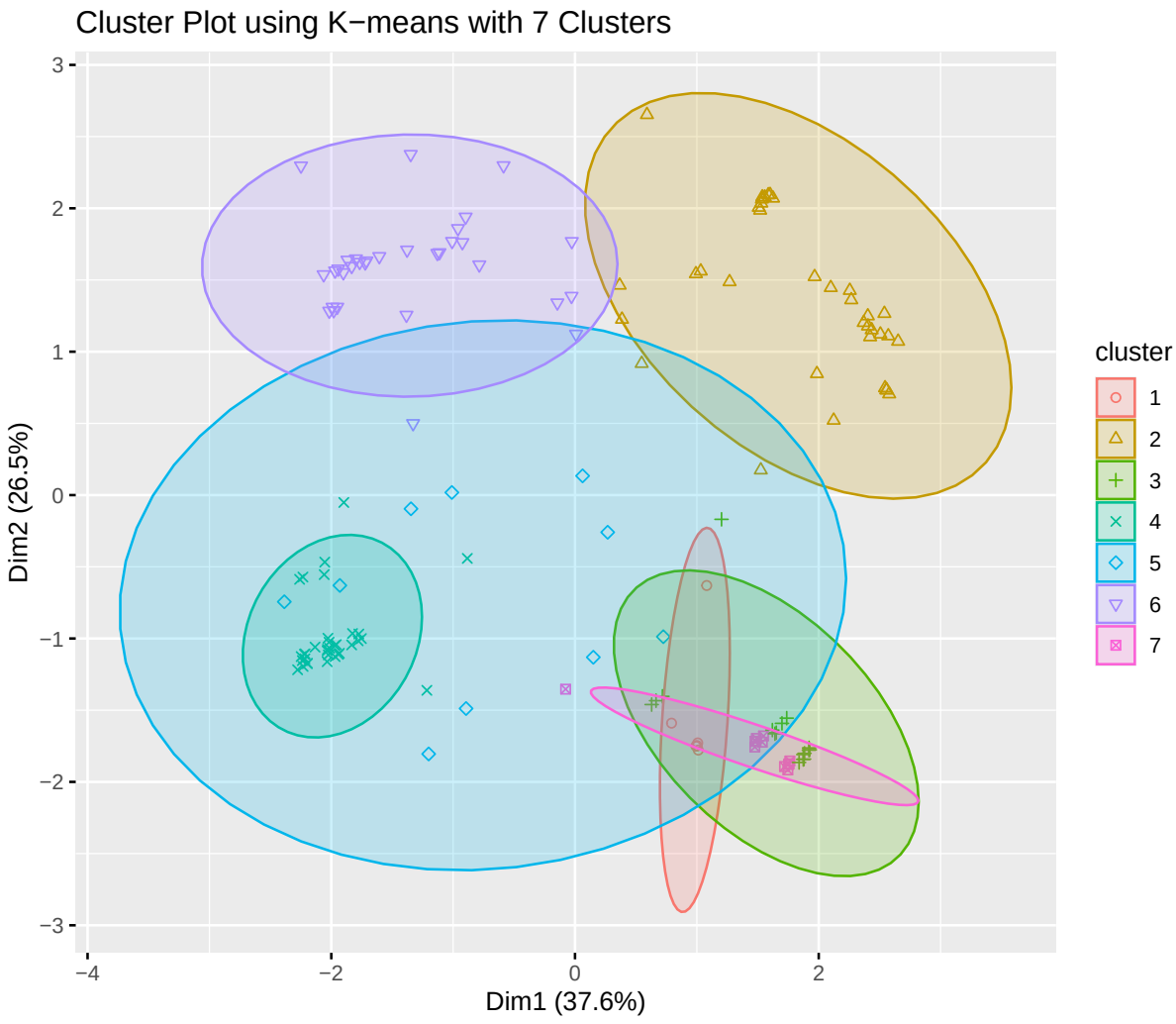
Plot with 4 Clusters

```
km.res <- kmeans(subset_clus, 4, nstart=4)

fviz_cluster(km.res, subset_clus,  geom = "point",
             ellipse.type = "norm", show.clust.cent = TRUE,
             main = "Cluster Plot using K-means with 4 Clusters")
```



Plot with 7 Clusters

```
km.res1 <- kmeans(subset_clus, 7, nstart=4)

fviz_cluster(km.res1, subset_clus,  geom = "point",
             ellipse.type = "norm", show.clust.cent = FALSE,
             main = "Cluster Plot using K-means with 7 Clusters")
```



Kmeans Summary Results

```
km.res1$centers
km.res1$withinss
km.res1$tot.withinss
km.res1$betweenss
#km.res1$size
km.res1$iter
km.res1$ifault
```

Cluster Analysis

K-means Cluster Analysis

```
best_kmeans <- kmeans(subset_clus, 4, nstart=4)
```

Display cluster members

```
head(subset_clus[best_kmeans$cluster==1],5)
```

```
[1] -1.714682 -1.691511 -1.645168 -1.598825 -1.552482
```

Append cluster assignment to data frame

```
best_kmeans_clus <- data.frame(subset_clus, best_kmeans$cluster)
best_kmeans_clus
tmp <- sav_clus
tmp1 <- sav_clus
```

Add best_kmeans.cluster to saved set

```
sav_clus <- data.frame(tmp, best_kmeans$cluster)
```

Determine number of clusters using another WSS method

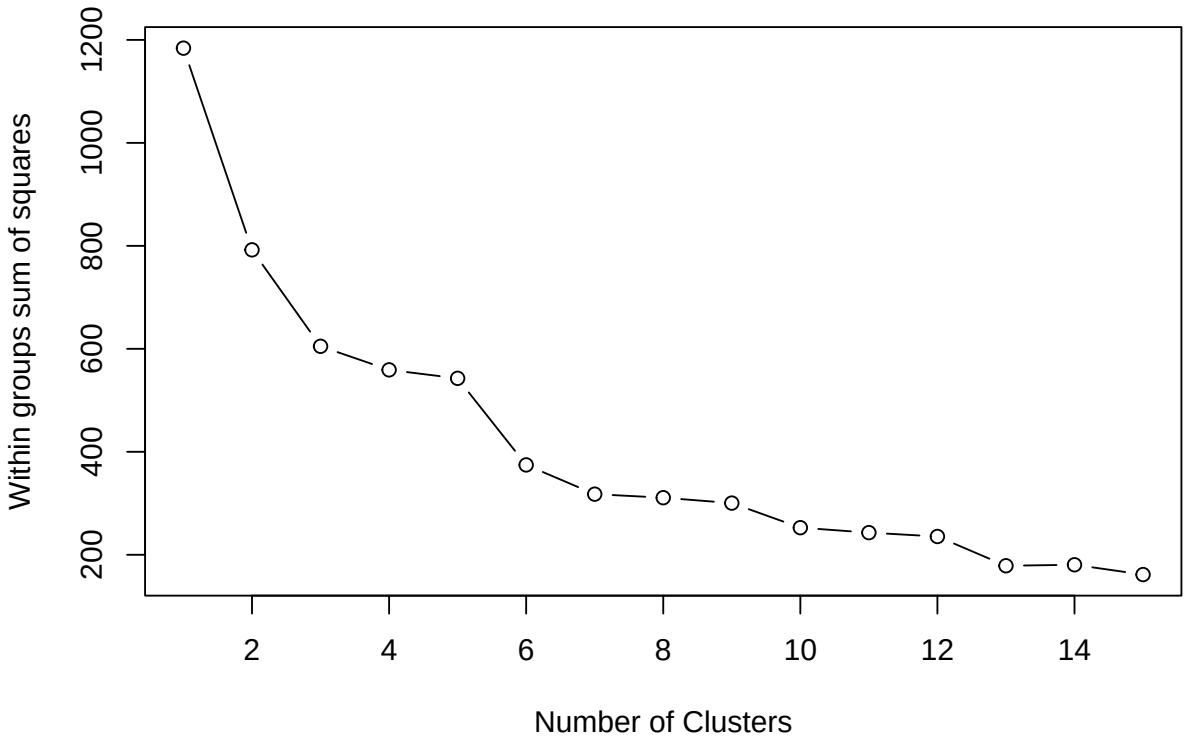
```
wss <- (nrow(subset_clus)-1)*sum(apply(subset_clus,2,var))
nrow(subset_clus)
```

```
[1] 149
```

```
for (i in 2:15) wss[i] <- sum(kmeans(subset_clus,
                                     centers=i)$withinss)
```

Plot K-means using WSS method

```
plot(1:15, wss, type="b", xlab="Number of Clusters",
     ylab="Within groups sum of squares")
```



Hierarchical Clusters

Compute dissimilarity matrix

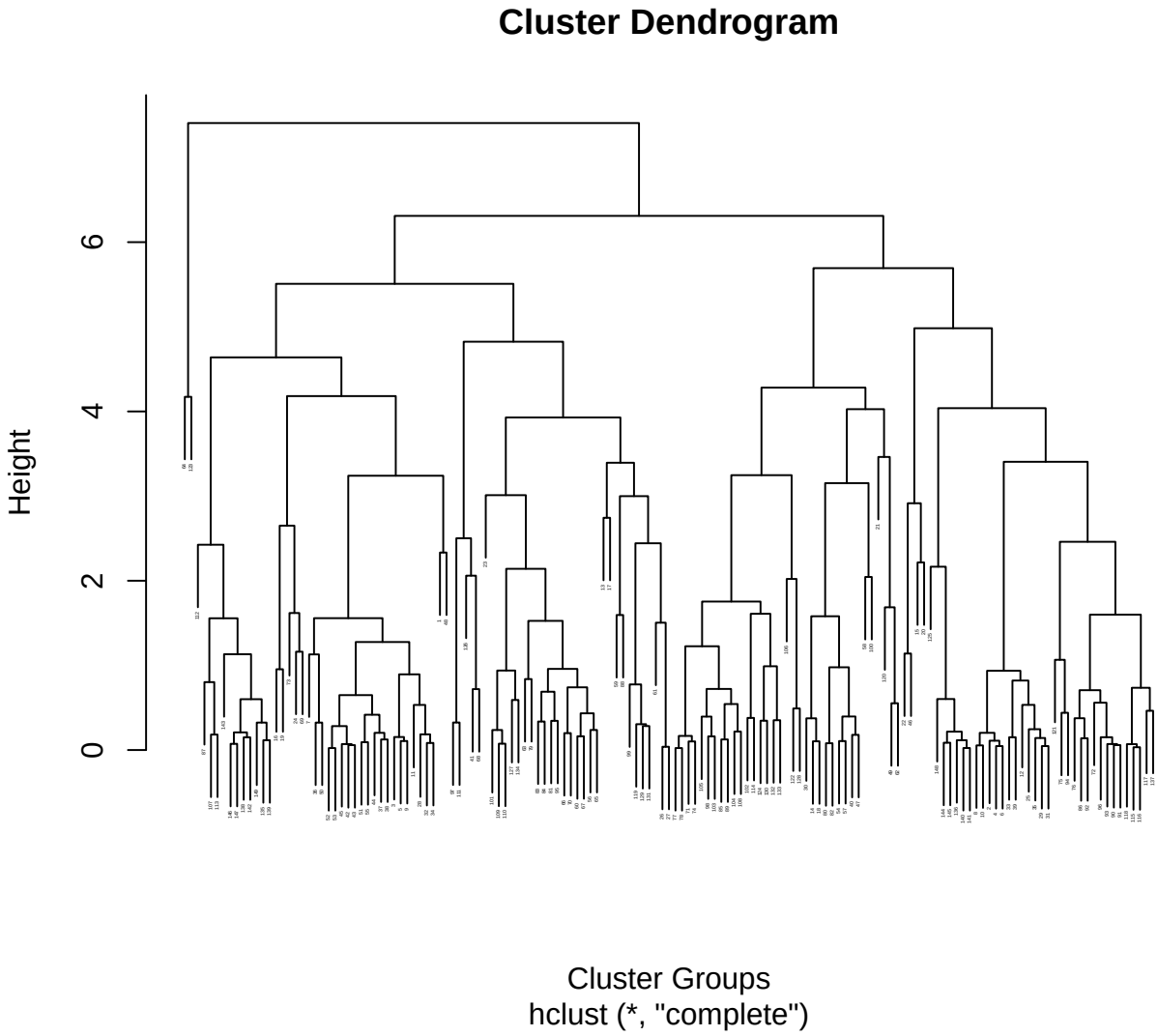
```
res.dist <- dist(as.matrix(subset_clus, method = "euclidean"))
```

Create hierarchical clustering

```
res.hc <- hclust(res.dist, method = "complete", members = NULL)
```

Plot Hierarchy

```
plot(res.hc, cex = 0.2, main = "Cluster Dendrogram", xlab = "Cluster Groups",
     ylab = "Height")
```



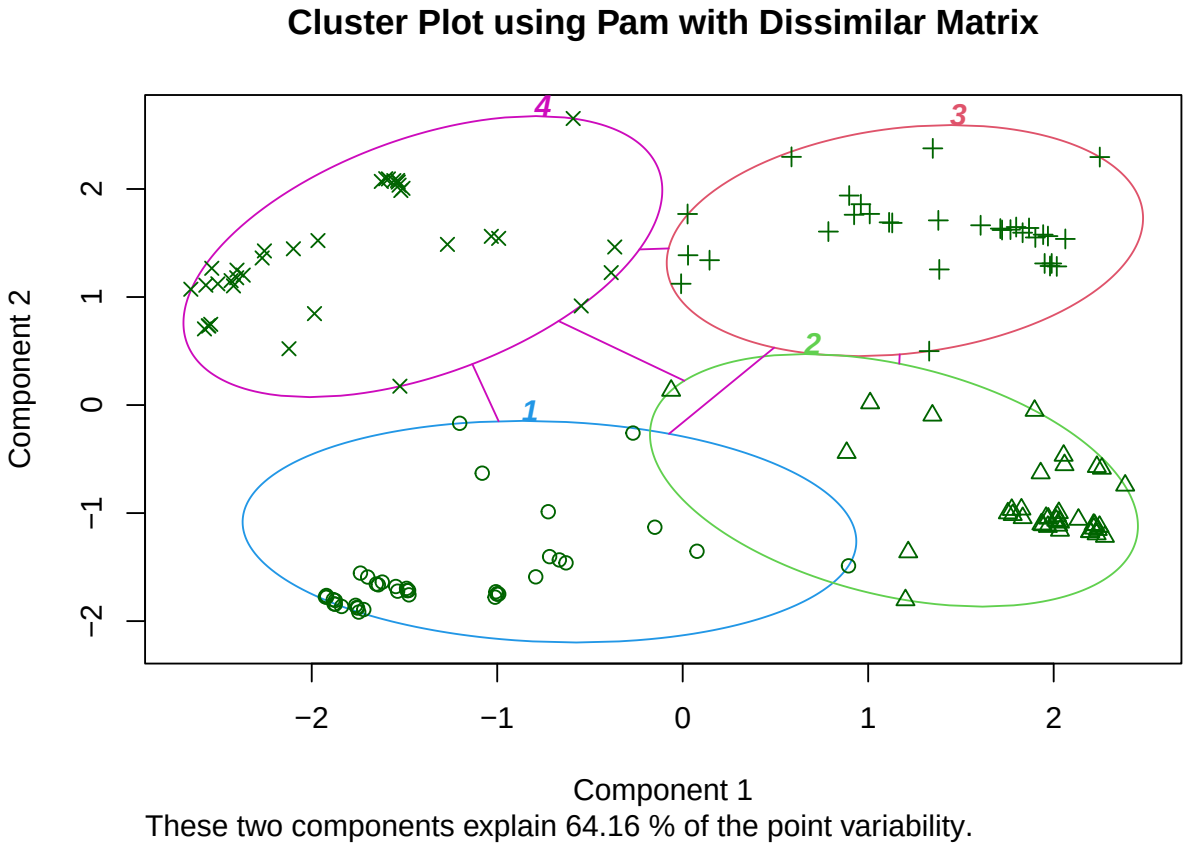
Clustering with K-medoids

With dissimilarity matrix from above

```
pam.res <- pam(res.dist, k=4)
summary(pam.res)
clus <- pam.res$clustering
```

```
clusplot(res.dist, clus, diss=TRUE,
         color = TRUE, labels=4,
```

```
         plotchar=TRUE,
         main = "Cluster Plot using Pam with Dissimilar Matrix")
```



Display Cluster Info

```
pam.res$isolation
```

```
 1  2  3  4
no no no no
Levels: no L L*
```

```
pam.res$clusinfo
```

```
     size max_diss  av_diss diameter separation
```

```
[1,]   41 5.118676 1.744605 5.836761   1.140114
[2,]   40 5.006735 1.534250 5.703791   1.140114
[3,]   32 3.165252 1.433966 4.823847   1.981160
[4,]   36 3.320892 1.561217 4.688911   2.009924
```

```
pam.res$call
```

```
pam(x = res.dist, k = 4)
```

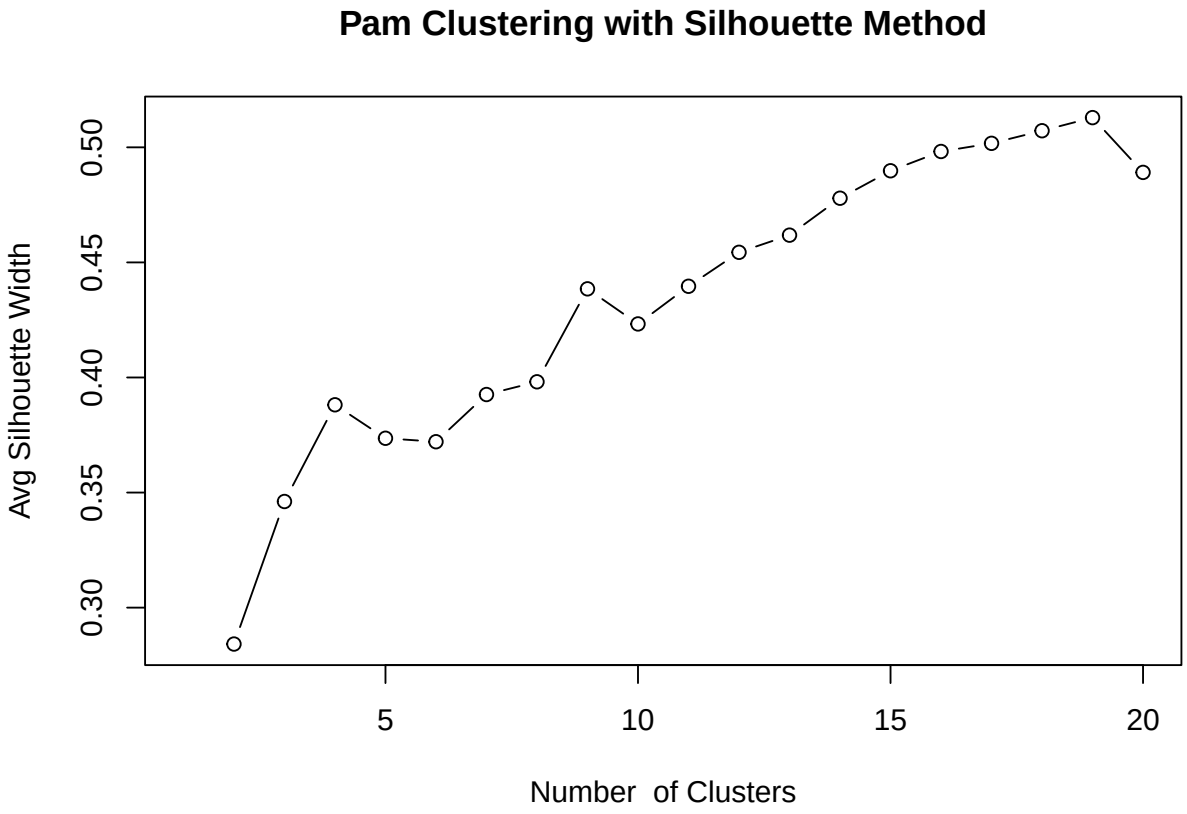
Plot PAM Using Silhouette Method

Create empty vector, display the cluster summary data, and plot

```
clus_res <- numeric()

for (k in 2:20) {
  clus_res[k] <- pam(subset_clus, k)$silinfo$avg.width
}

summary(clus_res)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
 0.2842  0.3904  0.4396  0.4342  0.4895  0.5129       1
```

```
plot(1:20, clus_res, xlab ="Number  of Clusters",
     main = "Pam Clustering with Silhouette Method",
     ylab = "Avg Silhouette Width",
     type = "b")
```



Clustering using DBSCAN

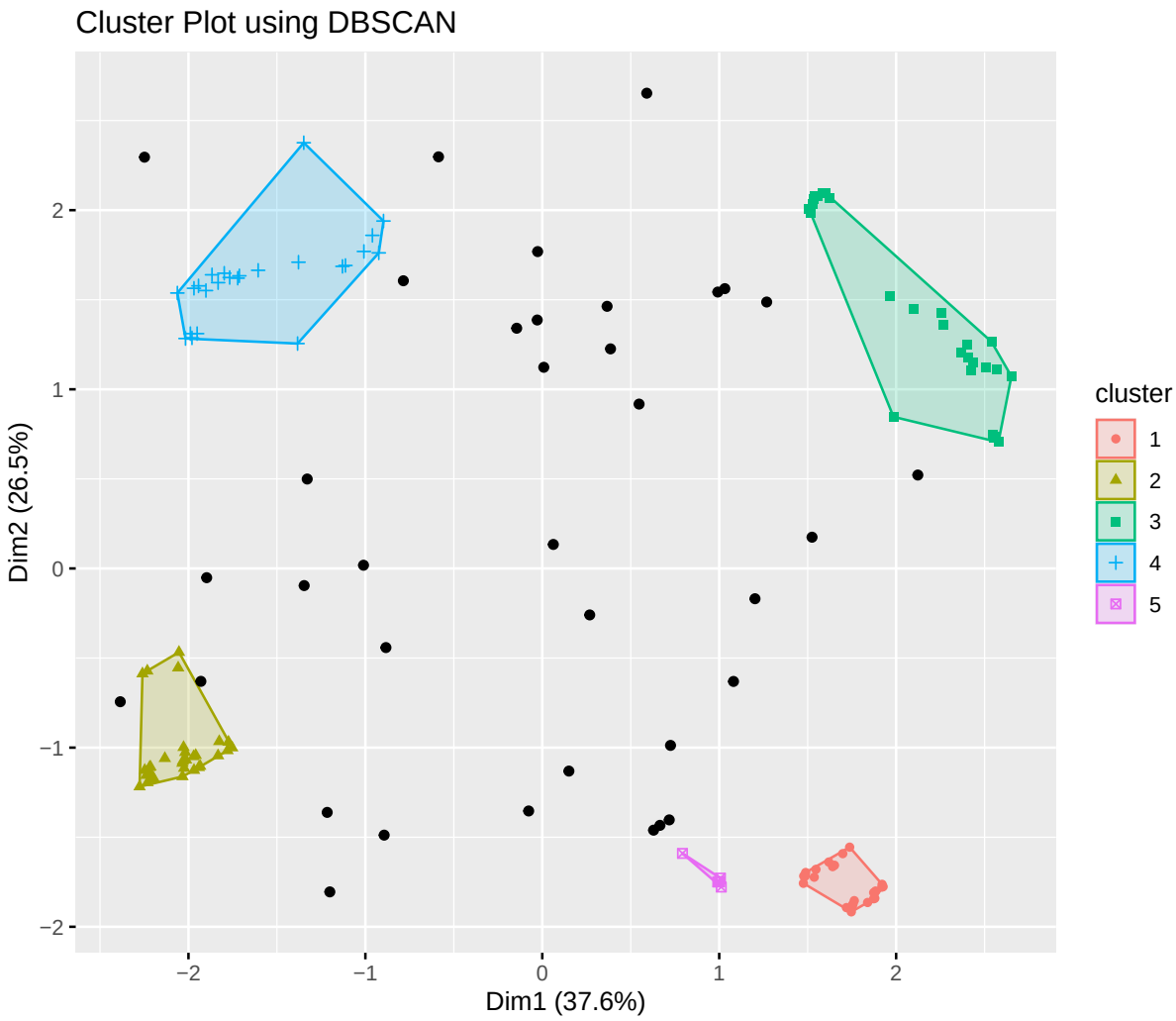
```
set.seed(123)
```

Create DBScan plot to compare against Kmeans

```
db_fpc <- fpc::dbscan(subset_clus, eps = 1.3, MinPts = 4)
```

Plot DBSCAN Plot

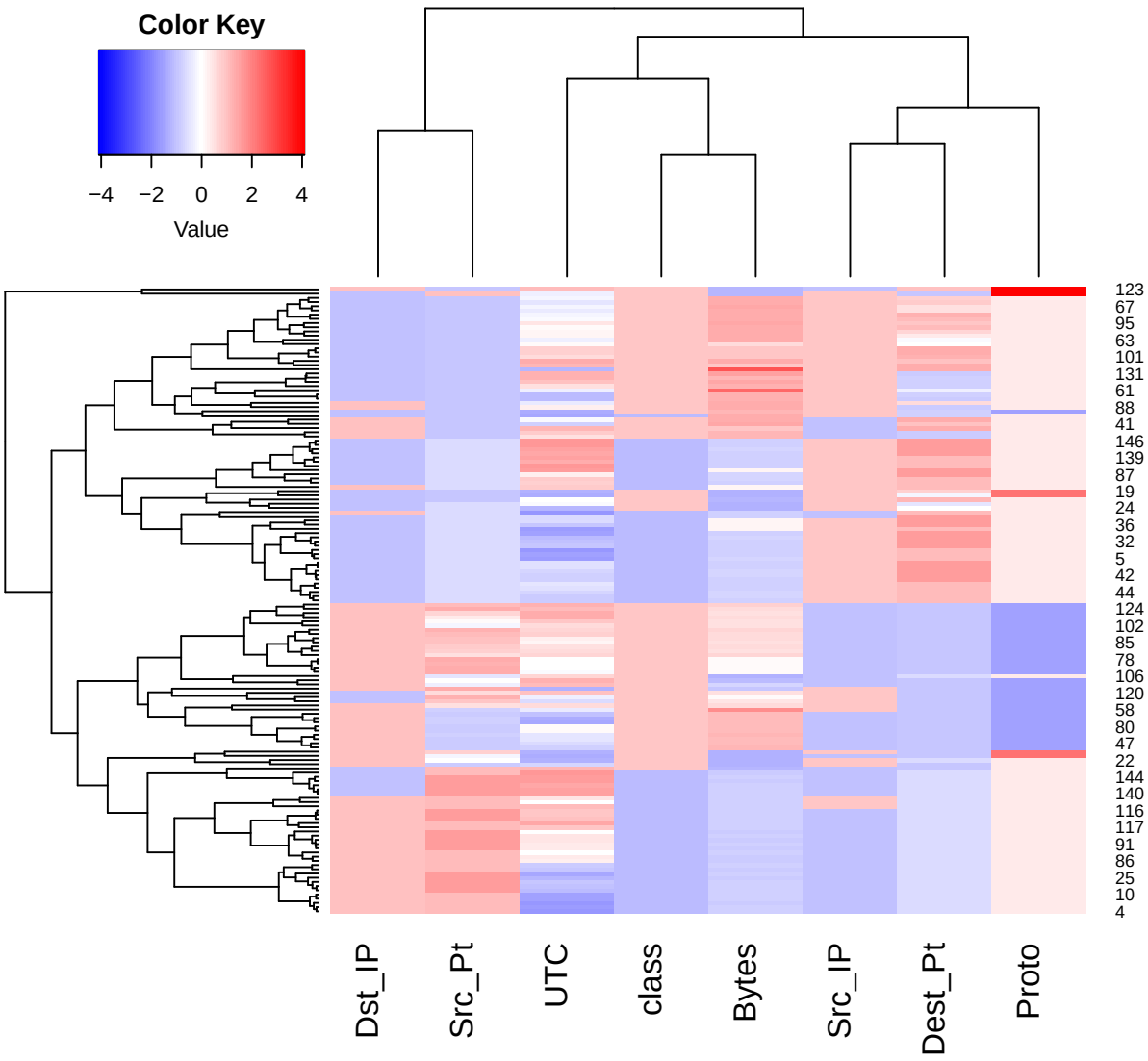
```
fviz_cluster(db_fpc, data = subset_clus, geom = "point",
             ellipse = TRUE, show.clust.cent = FALSE,
             main = "Cluster Plot using DBSCAN")
```



Heatmaps

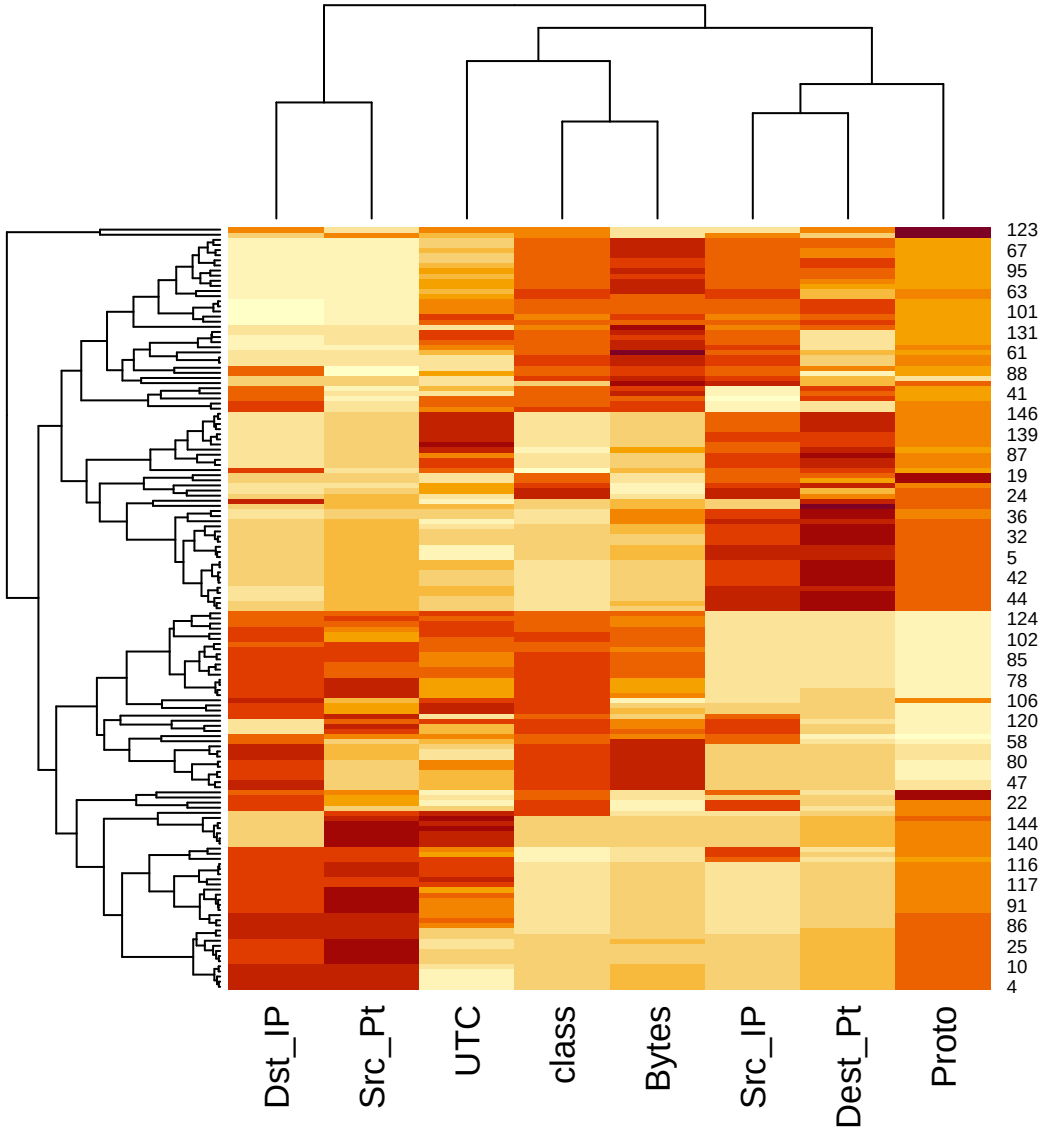
Use heatmaps to view correlations

```
heatmap.2(subset_clus, scale = "none", col = bluered(100),
          trace = "none", density.info = "none")
```



Split Heatmap by rows and by k-means

```
set.seed(123)
subset_mat <- as.matrix(subset_clus)
heatmap(subset_mat, k = 4)
```



Network Diagrams

Build network diagrams to view anomalies Create links and nodes, get unique list of IPs for the nodes list, SRC and Dst IPs

```
src <- test.ts %>%
  distinct(Src_IP) %>%
  rename(label = Src_IP)
```

```
dest <- test.ts %>%
  distinct(Dst_IP) %>%
  rename(label = Dst_IP)
```

Get unique IPs to create nodes list and add unique rowids

```
nodes <- full_join(src,dest, by = "label")

nodes <- nodes %>% rowid_to_column("id")
```

Create list of nodes per route

```
per_route <- test.ts %>%
  group_by(Src_IP, Dst_IP) %>%
  summarize(weight = n()) %>%
  ungroup()
```

```
`summarise()` has grouped output by 'Src_IP'. You can override using the
`.groups` argument.
```

Create the list of links aka edges

```
edges <- per_route %>%
  left_join(nodes, by = c("Src_IP" = "label")) %>%
  rename(from = id)

edges <- edges %>%
  left_join(nodes, by = c("Dst_IP" = "label")) %>%
  rename(to = id)
edges
```

```
# A tibble: 25 x 5
   Src_IP        Dst_IP         weight  from    to
   <fct>         <fct>           <int> <int> <int>
 1 10.100.1.123  10.200.1.15         6    12    13
 2 10.100.1.123  192.168.200.1       3    12    11
 3 10.100.10.123 10.100.10.126       2     1    10
 4 10.100.10.123 192.168.1.10       15     1     2
 5 10.100.10.123 192.168.1.12        4     1     6
```

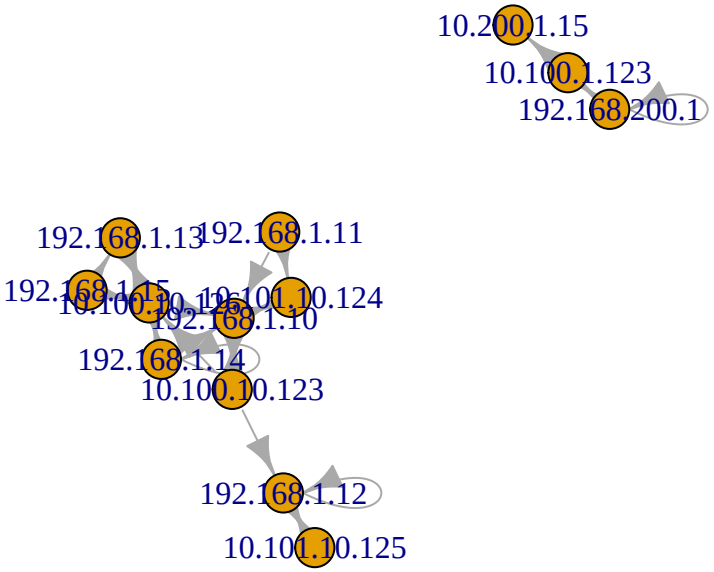
```
 6 10.100.10.126 192.168.1.10       1    10     2
 7 10.100.10.126 192.168.1.13       8    10     7
 8 10.100.10.126 192.168.1.14      20    10     9
 9 10.100.10.126 192.168.1.15       2    10     8
10 10.101.10.124 192.168.1.10       1     3     2
# ... with 15 more rows
```

Create a Community of Links for Network Plot

```
edges1 <- select(edges, Src_IP, Dst_IP)


edges <- select(edges, from, to, weight)
```

Plot Network Using igraph

```
routes_i <- graph_from_data_frame(d = edges, vertices = nodes, directed = TRUE)

plot(routes_i, layout = layout_with_graphopt, edge.arrow.linewidth = .01)
```



Build Network Using Tidygraph

```
routes_tidy <- tbl_graph(nodes = nodes, edges = edges, directed = TRUE)
```

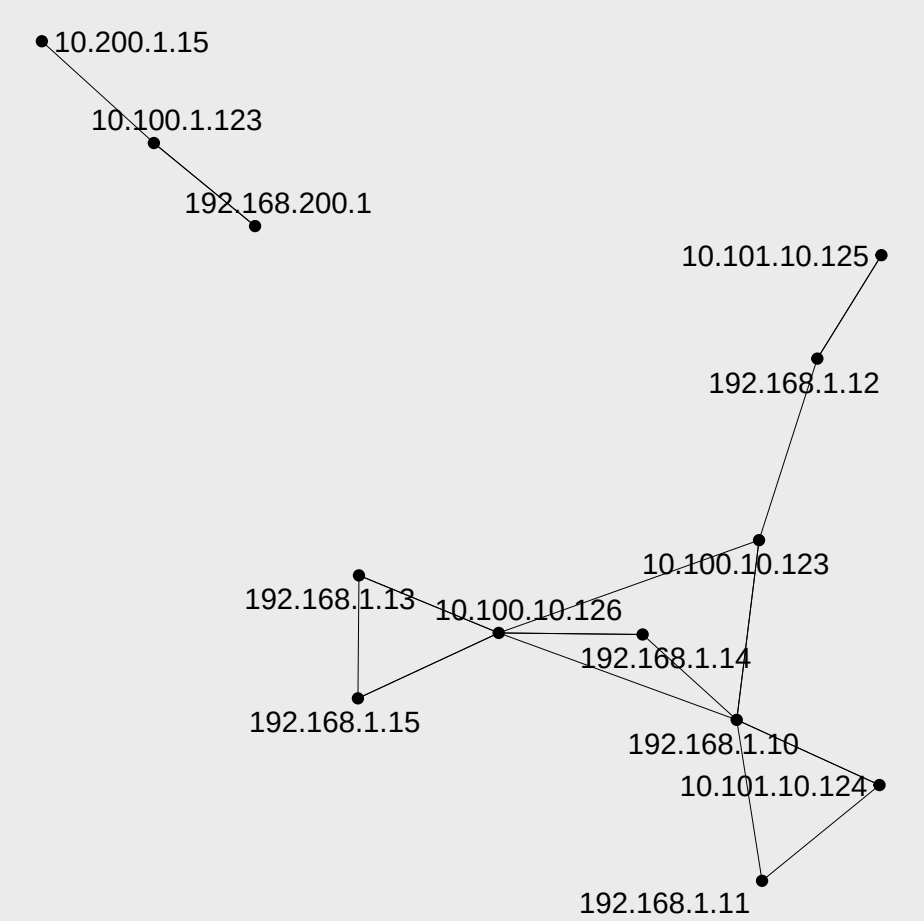
Plot Tidygraph Network

```
routes_tidy %>%
  activate(edges) %>%
  arrange(desc(weight))
```

```
# A tbl_graph: 13 nodes and 25 edges
#
# A directed multigraph with 2 components
#
# Edge Data: 25 x 3 (active)
   from    to weight
  <int> <int>  <int>
1     9    10     26
2    10     9     20
3     1     2     15
4     6     5      9
5    11    12      9
6    10     7      8
# ... with 19 more rows
#
# Node Data: 13 x 2
     id label
  <int> <fct>
1     1 10.100.10.123
2     2 192.168.1.10
3     3 10.101.10.124
# ... with 10 more rows
```

```
ggraph(routes_tidy, layout = "dh") +
geom_node_point() +
  geom_edge_link(edge_width = .09) +
  scale_edge_width(range = c(0.2, 2)) +
  geom_node_text(aes(label = label),  repel = TRUE) +
  labs(edge_width = "Connections")
```

```
Warning: Using the `size` aesthetic in this geom was deprecated in ggplot2 3.4.0.
i Please use `linewidth` in the `default_aes` field and elsewhere instead.
```



Anomaly Detection in Time Series Data using Anomalize

Create Tibble table

```
test.ts <- tibble(test.ts)

test.ts <- as_tbl_time(test.ts, index = UTC)
```

Plot Time Series Data using QESD Method

```
test.ts %>% time_decompose(Src_IP, method = "stl", frequency = "auto",
            trend = "auto") %>%
             anomalize(remainder, method = "gesd", alpha = 0.05,     max_anoms = 0.3) %>%
        time_recompose() %>%
```

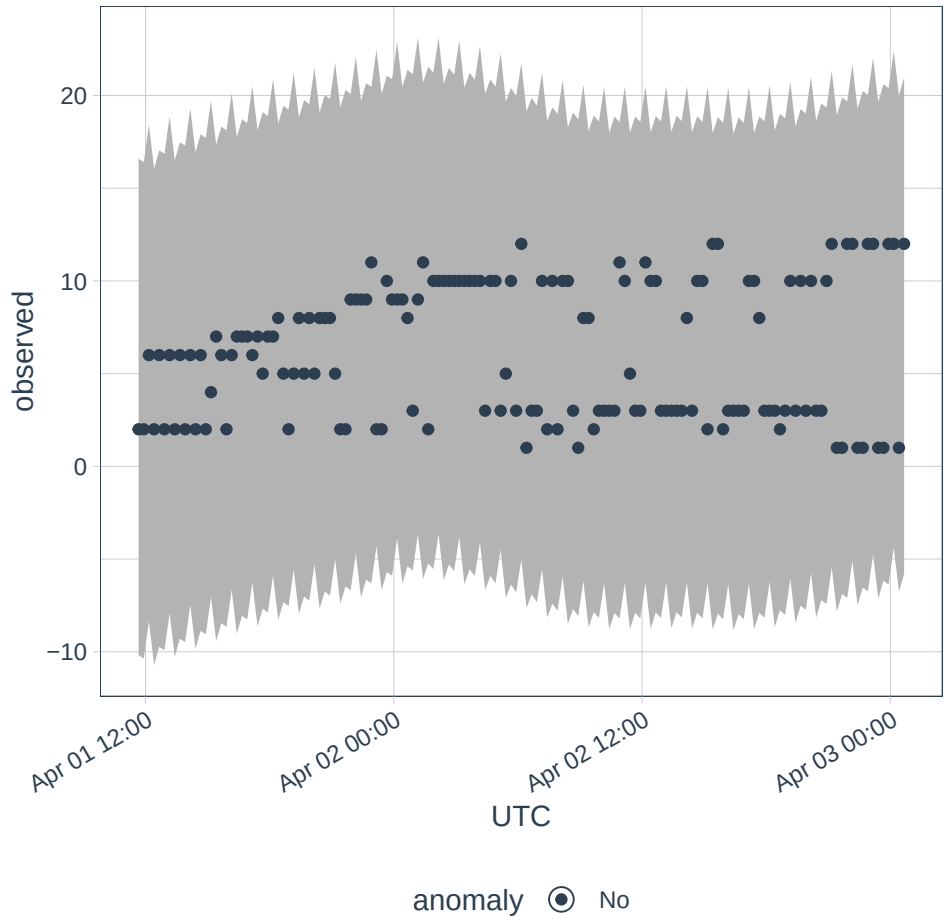
```
        plot_anomalies(time_recomposed = TRUE, ncol = 4)
```

```
frequency = 4 minutes
```

```
trend = 48 minutes
```

```
Registered S3 method overwritten by 'quantmod':
  method            from
  as.zoo.data.frame zoo
```

```
Registered S3 methods overwritten by 'forecast':
  method                 from
  autoplot.Arima         ggfortify
  autoplot.acf           ggfortify
  autoplot.ar            ggfortify
  autoplot.bats          ggfortify
  autoplot.decomposed.ts ggfortify
  autoplot.ets           ggfortify
  autoplot.forecast      ggfortify
  autoplot.stl           ggfortify
  autoplot.ts            ggfortify
  fitted.ar              ggfortify
  fortify.ts             ggfortify
  residuals.ar           ggfortify
```

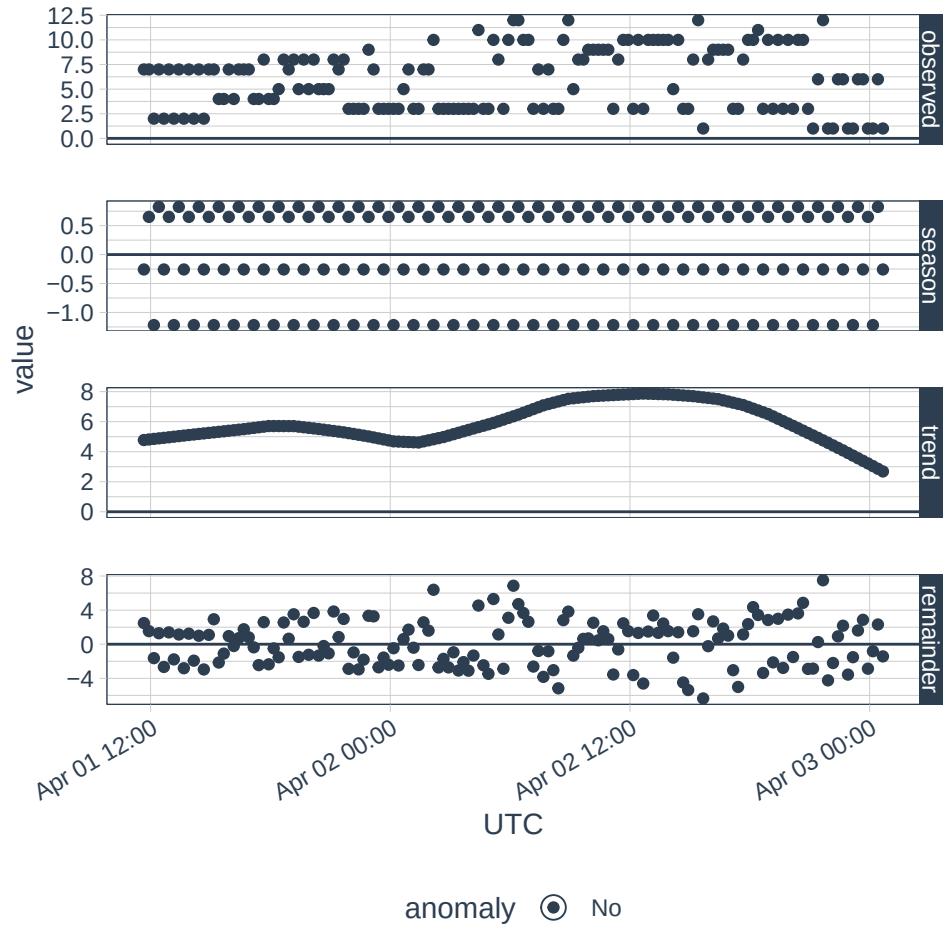


Plot Time Series Data using IQRD Method

```
test.ts %>% time_decompose(Dst_IP, method = "stl", frequency = "auto",
                           trend = "auto") %>%
  anomalize(remainder, method = "iqr", alpha = 0.05, max_anoms = 0.3) %>%
  plot_anomaly_decomposition()
```

```
frequency = 4 minutes
```

```
trend = 48 minutes
```



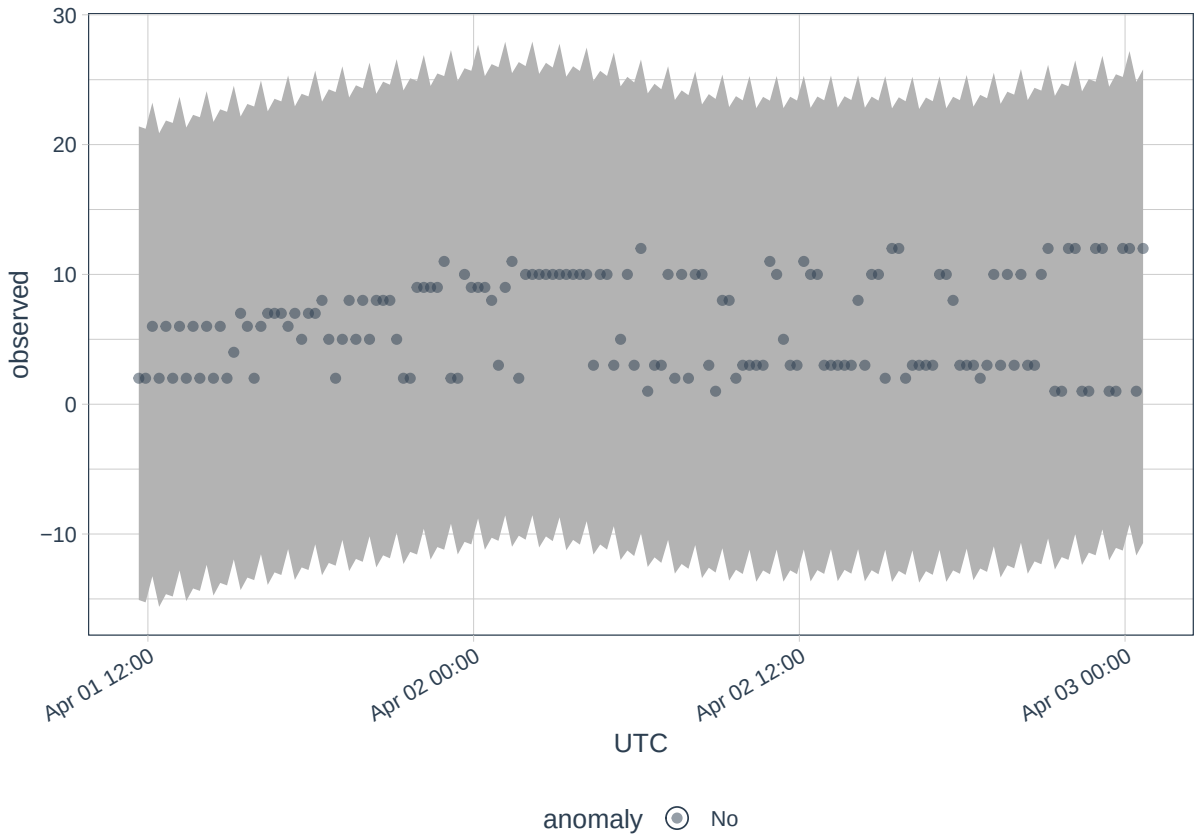
Plot Anomalies with SRC_IP

With SRC_IP

```
test.ts %>%
  time_decompose(Src_IP) %>%
  anomalize(remainder) %>%
  time_recompose() %>%
  plot_anomalies(time_recomposed = TRUE, ncol = 6, alpha_dots = 0.5)
```

```
frequency = 4 minutes
```

```
trend = 48 minutes
```

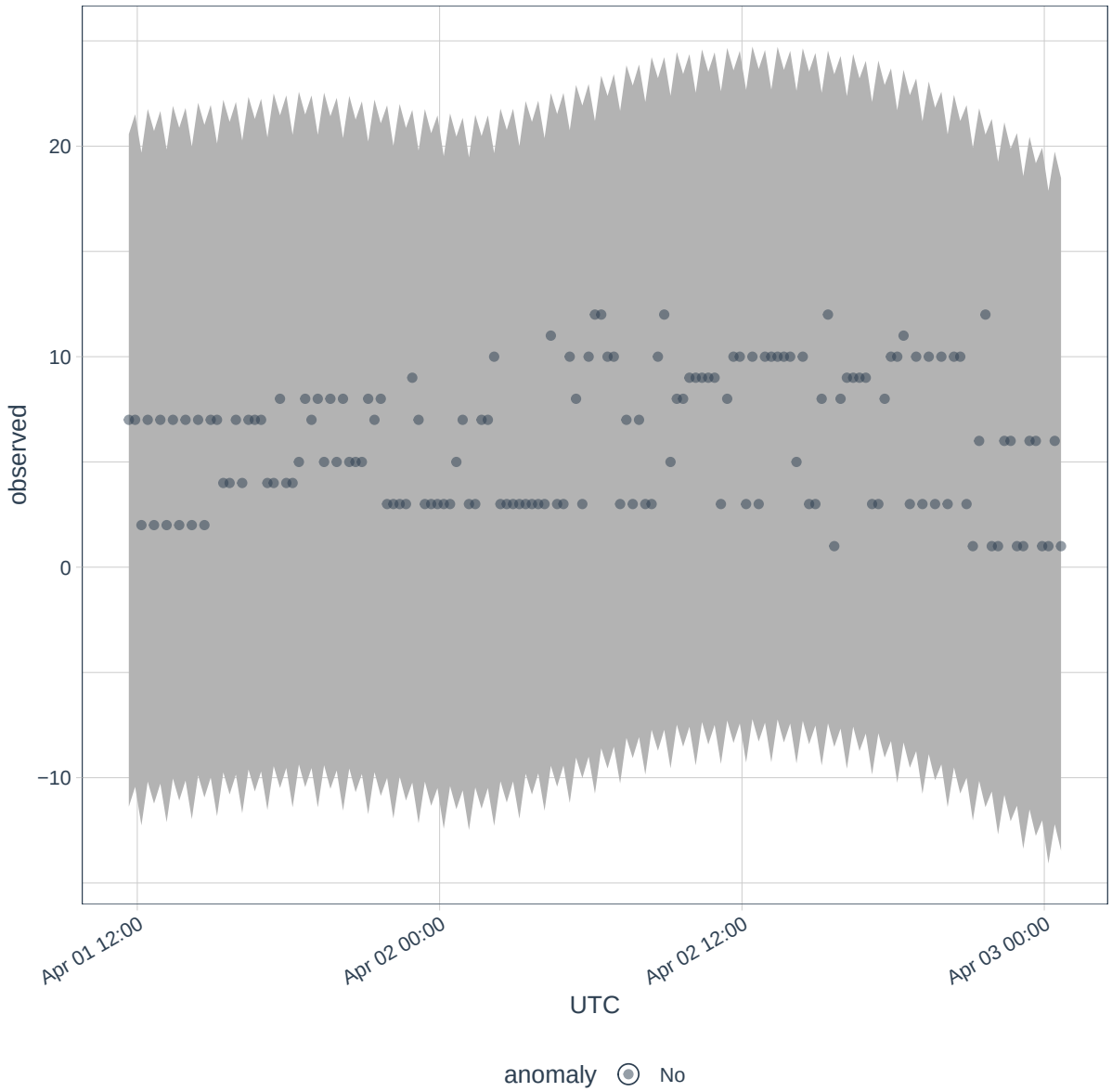


Plot Anomalies with DST_IP

```
test.ts %>%
  time_decompose(Dst_IP) %>%
  anomalize(remainder) %>%
  time_recompose() %>%
  plot_anomalies(time_recomposed = TRUE, ncol = 8, alpha_dots = 0.5)
```

```
frequency = 4 minutes
```

```
trend = 48 minutes
```



Extract Anomalous Datapoints

```
anomaly <- test.ts %>%
  time_decompose(Src_IP) %>%
  anomalize(remainder) %>%
  time_recompose() %>%
  filter(anomaly == 'Yes')
```

```
frequency = 4 minutes
```

```
trend = 48 minutes
```

```
  anomaly
```

```
# A time tibble: 0 x 10
# Index:         UTC
# ... with 10 variables: UTC <dttm>, observed <dbl>, season <dbl>, trend <dbl>,
#   remainder <dbl>, remainder_l1 <dbl>, remainder_l2 <dbl>, anomaly <chr>,
#   recomposed_l1 <dbl>, recomposed_l2 <dbl>
```