

Clustering (k-means)

K-Means is a clustering approach that belongs to the class of unsupervised statistical learning methods. The general idea of a clustering algorithm is to partition a given dataset into distinct, exclusive clusters so that the data points in each group are quite similar to each other.

Let's read our building records as one large dataset of around 6 million and try to find cluster's.

- 1) Reading the csv and removing features which we would not use and will not make sense in the clustering

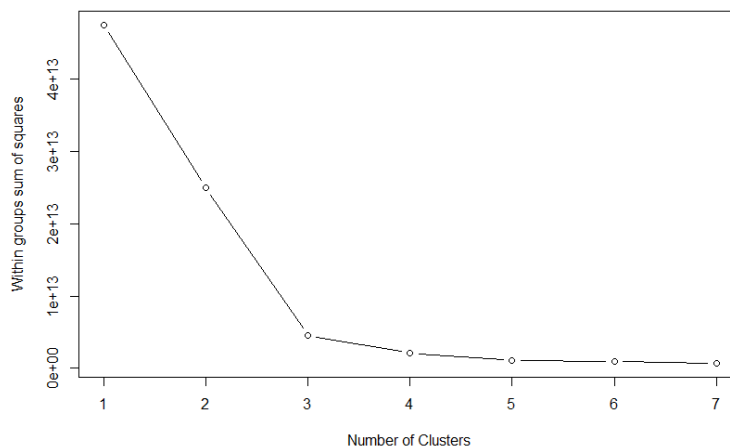
```
k$sv <- read.csv("finland_allBuilding_allweather.csv", stringsAsFactors = FALSE)
require(dplyr)
mergedData <- csv %>%
  select(-c(X, date,
            vac, BuildingID,
            meternumb, type, Consumption, City, wind_Direction, Conditions))
```

- 2) Converting the features to appropriate data type

```
## Converting non numeric features to numeric
mergedData$hour <- as.numeric(mergedData$hour)
mergedData$consumption.kwh.sqm <- as.numeric(mergedData$consumption.kwh.sqm)
mergedData$month <- as.numeric(mergedData$month)
mergedData$Day.of.week <- as.numeric(mergedData$Day.of.week)
mergedData$weekday <- as.numeric(mergedData$weekday)
mergedData$Holiday <- as.numeric(mergedData$Holiday)
mergedData$Base_Hour_Class <- as.factor(mergedData$Base_Hour_Class)
```

- 3) A plot of the within groups sum of squares by number of clusters extracted can help determine the appropriate number of clusters. We will use bend graph function to check within groups sum of squares by number of clusters and deciding the optimal number of clusters

```
wssplot <- function(data, nc=15, seed=1234){
  wss <- (nrow(data)-1)*sum(apply(data, 2, var))
  for (i in 2:nc){
    set.seed(seed)
    wss[i] <- sum(kmeans(data, centers=i)$withinss)}
  plot(1:nc, wss, type="b", xlab="Number of clusters",
       ylab="within groups sum of squares")}
wssplot(mergedData[, -10], nc=7)
```



- 4) From the graph we can see that, within groups sum of squares is not improving after K =3
- 5) Using k=3, Let's use the **kmeans** function from R base stats package. Removing Base_Hour_Class from the feature list. Hence mergedata[, -10]

```
## Using k=3 as seen from the Bend graph
k.means.fit <- kmeans(mergedata[, -10], 3)
```

- 6) We can see the Within cluster sum of squares by cluster percentage of 90.5% which suggests good variance in between cluster

```
within cluster sum of squares by cluster:
[1] 1.500202e+09 2.977045e+12 1.524417e+12
(between_ss / total_ss = 90.5 %)
```

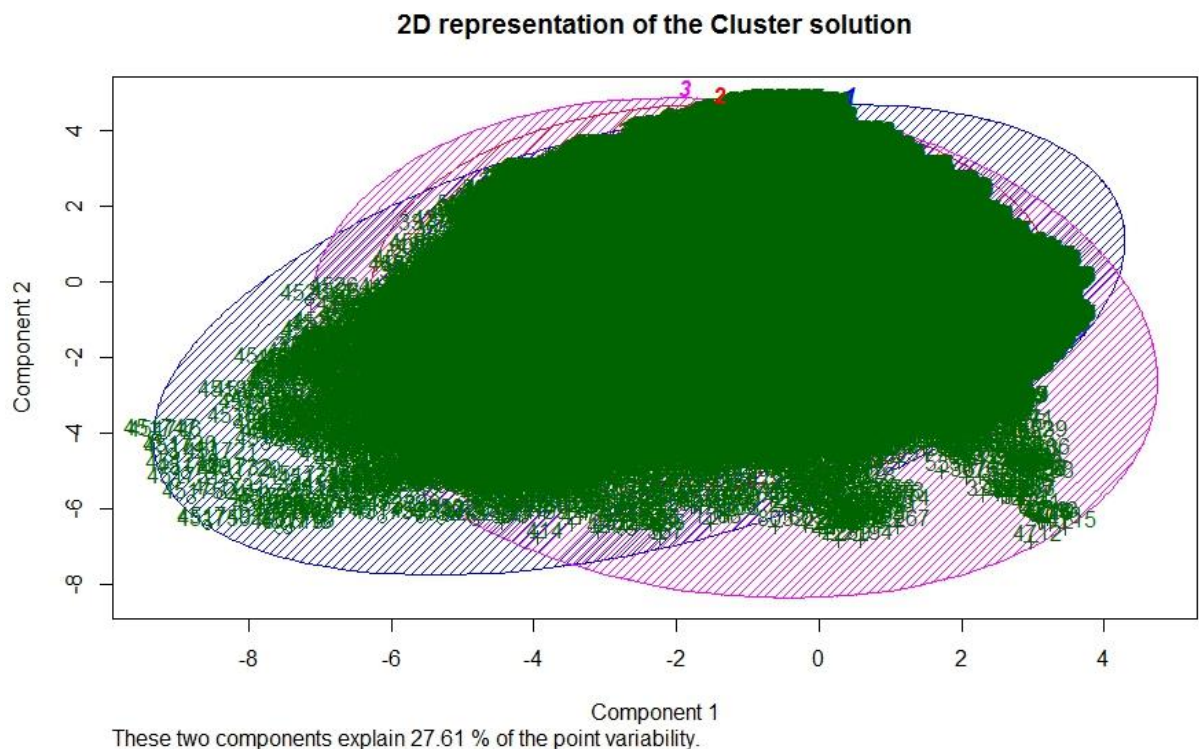
- 7) Further analyzing the results of our model, we can find out the centers of our clusters according to the feature

```
> k.means.fit$centers
```

	hour	month	Day.of.week	weekday	Base_hour_Flag	Holiday	area_floor._m.sqr	consumption.kwh.sqm	base_hr_usage	TemperatureF	Dew_PointF
1	11.5	6.111233	4.005333	0.7137052	0.2916667	0.04689749	11986.321	0.012968686	131.59166	44.16141	37.78187
2	11.5	6.415397	4.003208	0.7145148	0.2916667	0.04570970	41339.000	0.006159809	205.01885	45.64414	39.11011
3	11.5	6.182115	4.005292	0.7135964	0.2916667	0.04677407	3558.796	0.022182093	48.95188	44.27843	37.86958

	Humidity	Sea_Level_PressureIn	VisibilityMPH	wind_SpeedMPH	windDirDegrees
1	80.61580	29.90159	5.506452	6.453348	177.8844
2	80.27218	29.89641	5.525084	6.512837	178.1044
3	80.52758	29.90055	5.528377	6.507188	178.2768

- 8) We can see good separation in terms of **area_floor, consumption, base_hr_usage**.
- 9) Plotting the result in 2-D space would give some idea about our cluster. Since the data is huge, we will not be able to analyze the cluster's well



10) Plotting against our Base_Hour_Flag, we can see the below result

```
> table(mergeData[,10],k.means.fit$cluster)

      1      2      3
High 162047 14099 177735
Low  121465 15829 130641
```

11) We can say from this interpretation is that, the Base_Hour_Flag is missing a cluster of records. It should be **High**, **Low** and maybe **Neutral (for the consumption which is equal to the base_hour_usage)**

Clustering (hierarchical clustering)

- 1) K-means clustering requires us to specify the number of clusters, and finding the optimal number of clusters can often be hard. **Hierarchical clustering** is an alternative approach which builds a hierarchy from the bottom-up, and doesn't require us to specify the number of clusters beforehand.
- 2) Hierarchical methods use a distance matrix as an input for the clustering algorithm. The choice of an appropriate metric will influence the shape of the clusters, as some elements may be close to one another according to one distance and farther away according to another
- 3) We can use hclust for this. hclust requires us to provide the data in the form of a distance matrix. We can do this by using dist. By default, **the complete linkage method is used**.
- 4) Now since the data is huge, we will first transpose our data to a data matrix, and calculate the distance matrix

```
# Create transposed data matrix
data.matrix.t <- t(as.matrix(mergeData[, -10]))

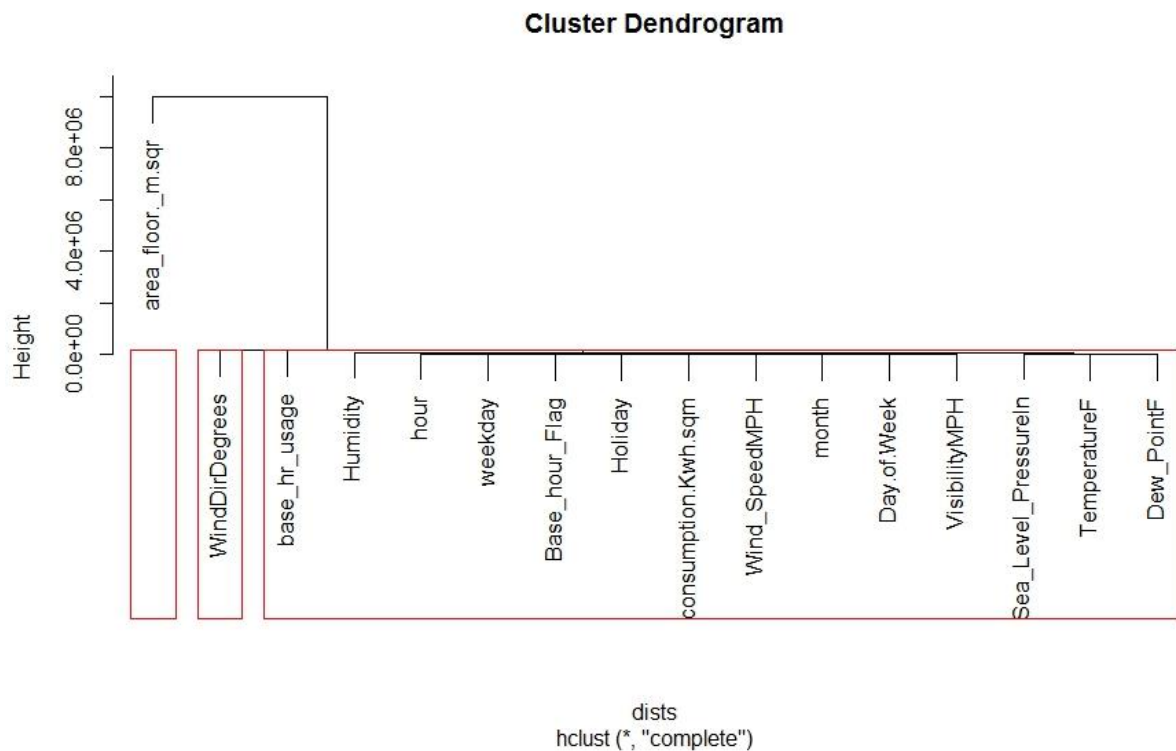
# Create distance matrix
dists <- dist(data.matrix.t)
```

5) Now calculating the cluster

```
# clustering
hcl <- hclust(dists)
```

6) We can draw the dendrogram and see the results

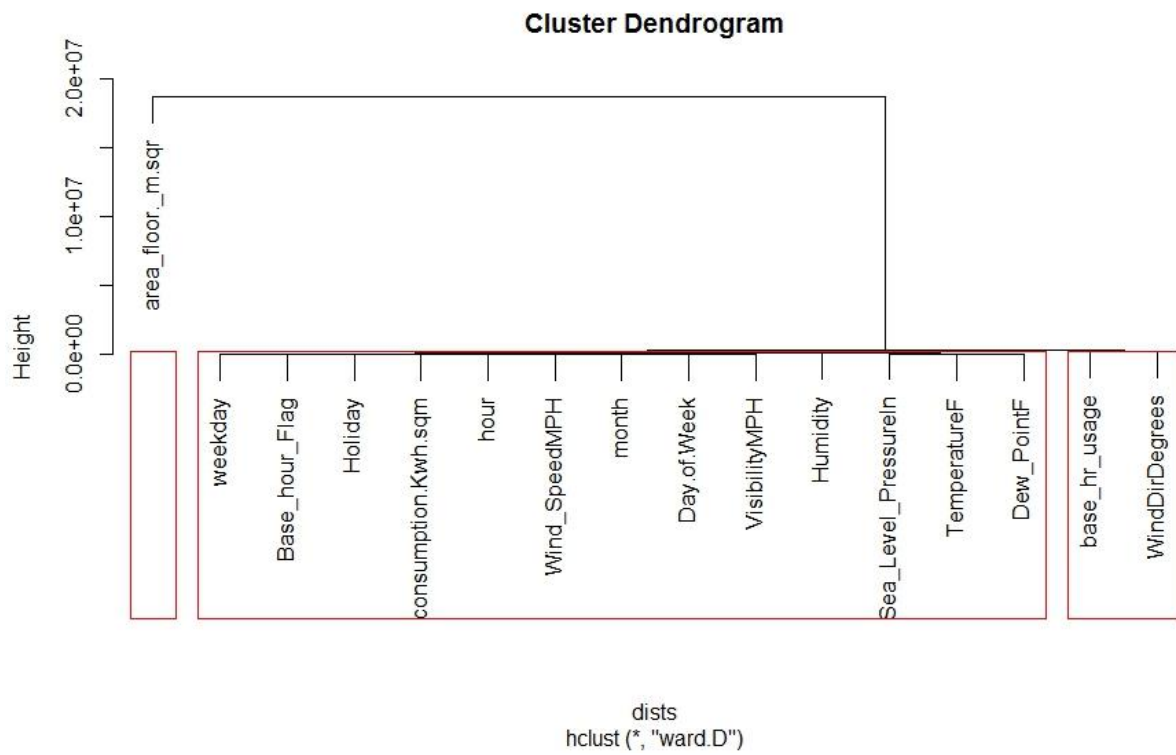
```
# draw dendrogram with red borders around the 3 clusters
rect.hclust(hcl, k=3, border="red")
```



- 7) From the hierarchical clustering we can see that, 3 clusters are formed. With area_floor again driving the cluster segmentation. WindDirDegrees a major contributor in a cluster and the rest for the third cluster
- 8) We can also use Ward's minimum variance criterion to minimize the total within-cluster variance and plot the result

```
#We use the Euclidean distance as an input for the clusterii
H.fit <- hclust(dists, method="ward.D")

#The clustering output can be displayed in a dendrogram
plot(H.fit) # display dendrogram
groups <- cutree(H.fit, k=3) # cut tree into 3 clusters
# draw dendrogram with red borders around the 3 clusters
rect.hclust(H.fit, k=3, border="red")
```



- 9) A different clustering group can be analyzed from here. Base_hr_usage now influencing the second cluster and rest remains the same. This is pretty much in alliance with our K-means clustering algorithm