

# Unsupervised Learning

Jayanth Rasamsetti  
Founder [www.sgmoid.com](http://www.sgmoid.com)  
Columbia University (MS)  
IIT-Madras (B.Tech & M.Tech)



# Fundamentals

Clustering - Understanding Distance

Unsupervised Learning

K-Means and K-medoids

# Why clustering? Applications of Clustering

## Why clustering?

- 1) To group similar objects/data points
- 2) To find homogeneous sets of customers
- 3) To segment the data in similar groups

## Applications:

- 1) Marketing: Customer Segmentation & Profiling
- 2) Libraries: Book classification
- 3) Retail: Store Categorization

# What is Clustering?

Clustering is a technique for finding similar groups in data, called clusters

Clustering is an Unsupervised Learning Technique

Clustering can also be thought of as a case reduction technique wherein it groups together similar records in cluster

Clustering helps simplify data by reducing many data points into a few clusters (segments)

# What is a Cluster?

A cluster can be defined as a collection of objects which are “similar” between them and are “dissimilar” to the objects belonging to other clusters

How do we define “Similar” in clustering?

Based on Distance

Shoppers	Price Conscious	Brand Loyalty
A	2	4
B	8	2
C	9	3
D	1	5
E	8	1



# How do we define “(dis) Similar” ?

Similar in clustering is based on Distance

Various distance measures

Euclidean Distance



Chebyshev Distance



$$\text{Manhattan Distance} = 8 + 4 = 12$$

Manhattan Distance ...and more



$$\text{Chebyshev Distance} = \text{Max} (8, 4) = 8$$



$$\text{Euclidean Distance} = \text{sqrt} ( 8^2 + 4^2 ) = 8.94$$



# Chebyshev Distance

In mathematics, Chebyshev distance is a metric defined on a vector space where the distance between two vectors is the greatest of their differences along any coordinate dimension

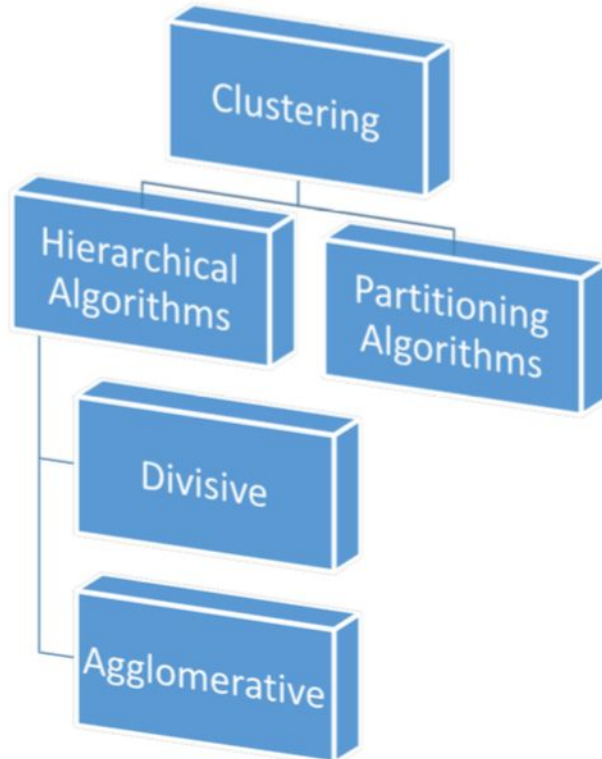
Assume two vectors: A ( $x_1, y_1, \dots, z_1$ ) & B ( $x_2, y_2, \dots, z_2$ )

Chebyshev Distance =  $\text{Max} ( |x_2 - x_1| , |y_2 - y_1| , \dots , |z_2 - z_1| )$

Application: Survey / Research Data where the responses are Ordinal Reference

Link: [https://en.wikipedia.org/wiki/Chebyshev\\_distance](https://en.wikipedia.org/wiki/Chebyshev_distance)

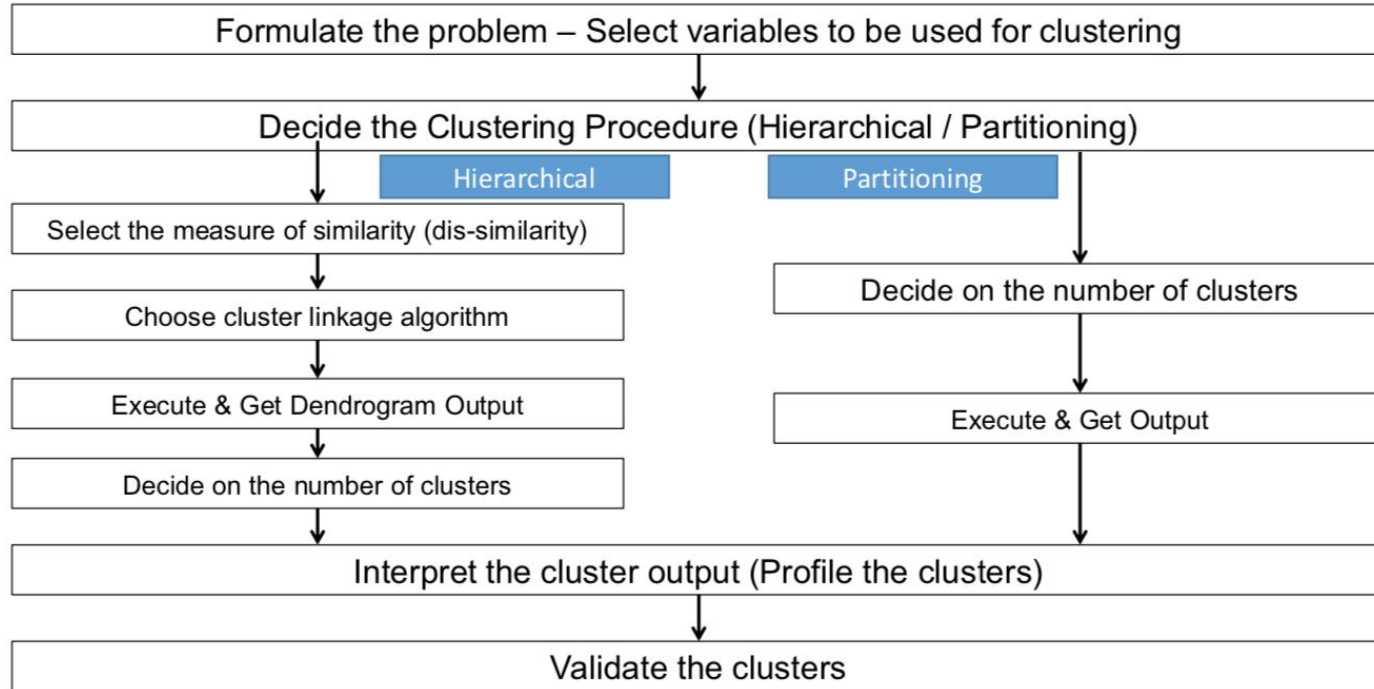
# Types of Clustering Procedures



- Hierarchical clustering is characterized by a tree like structure and uses distance as a measure of (dis)similarity
- Partitioning Algorithms starts with a set of partitions as clusters and iteratively refines the partitions to form stable clusters



# Steps involved in Clustering



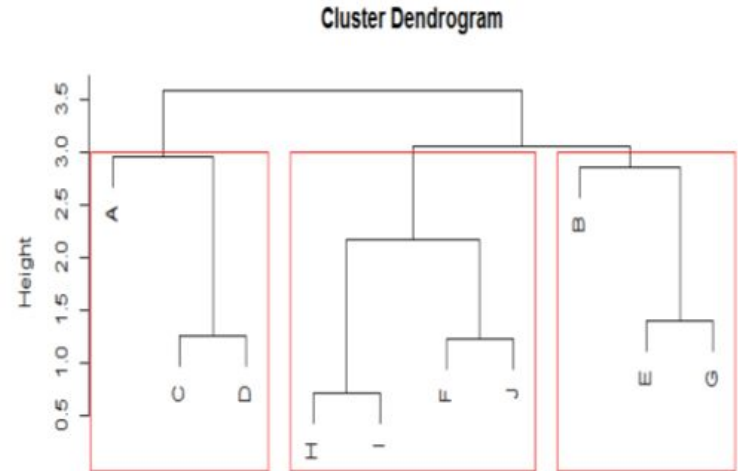
# Hierarchical Clustering

# Hierarchical Clustering

Hierarchical Clustering is a clustering techniques which tends to create clusters in a hierarchical tree like structure

Hierarchical clustering makes use of Distance as a measure of similarity

Cluster tree like output is called Dendrogram

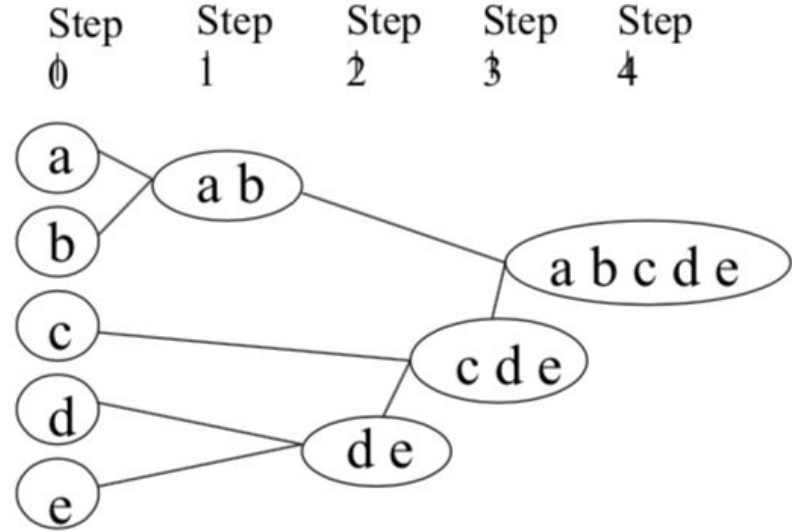


# Hierarchical Clustering | Agglomerative Clustering Steps

Starts with each record as a cluster of one record each

Sequentially merges 2 closest records by distance as a measure of (dis)similarity to form a cluster. This reduces the number of records by 1

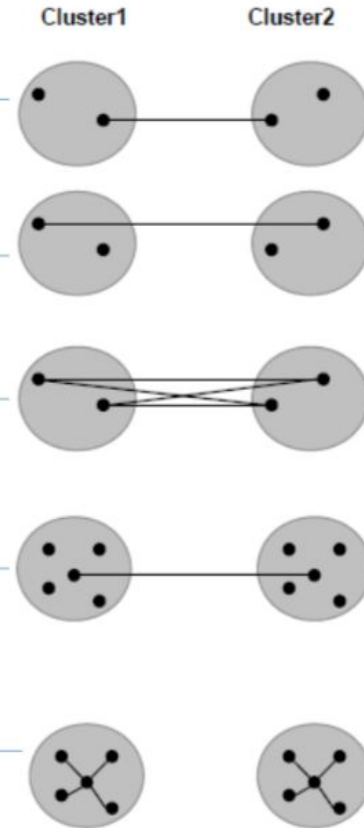
Repeat the above step with new cluster and all remaining clusters till we have one big cluster



How do you measure the distance between cluster (a,b) and (c) or the cluster (a,b) and (d,e)?

# Agglomerative Clustering Linkage Algorithms

- Single linkage – Minimum distance or Nearest neighbour rule
- Complete linkage – Maximum distance or Farthest distance
- Average linkage – Average of the distances between all pairs
- Centroid method – combine cluster with minimum distance between the centroids of the two clusters
- Ward's method – Combine clusters with which the increase in within cluster variance is to the smallest degree



# Hierarchical Clustering for Retail Customers

Case Study: Indian Retail is a USD 125 Billion Industry. It is extremely important to segment customers and target the appropriate offers. Q Mart has hired you as a Machine Learning Expert to find out the patterns from their shopping purchase data.



# Hierarchical Clustering for Retail Customers

Index	Cust_ID	Name	Avg_Mthly_Spend	No_Of_Visits	Apparel_Items	FrV_Items	Staples_Items
0	1	A	10000	2	1	1	0
1	2	B	7000	3	0	10	9
2	3	C	7000	7	1	3	4
3	4	D	6500	5	1	1	4
4	5	E	6000	6	0	12	3
5	6	F	4000	3	0	1	8
6	7	G	2500	5	0	11	2
7	8	H	2500	3	0	1	1
8	9	I	2000	2	0	2	2
9	10	J	1000	4	0	1	7

Let us find the clusters in given Retail Customer Spends data

We will use Hierarchical Clustering technique

**AVG\_Mthly\_Spend**

**No\_of\_Visits**

**Counts**

## Building the hierarchical clusters (without variable scaling)

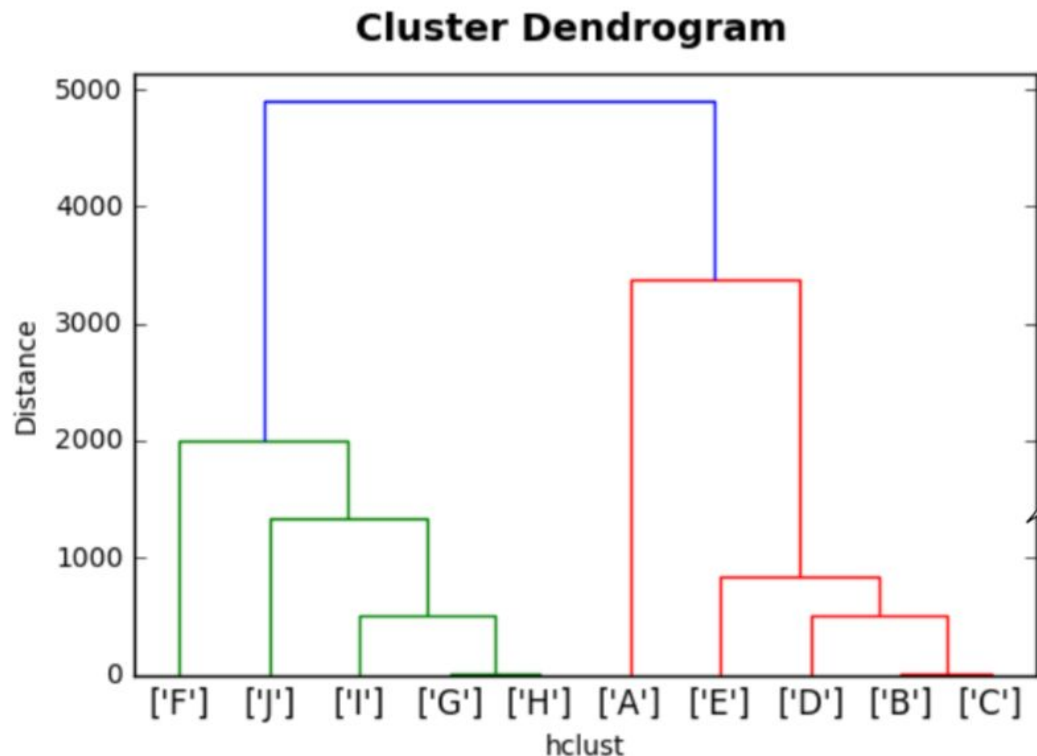
```
## Distance Computation
from scipy.spatial.distance import pdist, squareform
pdist ## Ctrl + i
help(pdist)
d_euc = pdist(RCDF.ix[:,2:7], metric = "euclidean")

## Building the Clusters
from scipy.cluster.hierarchy import linkage, dendrogram, cut_tree
help(linkage)
clus1 = linkage(d_euc, method = "average")

## Displaying the Clusters in Dendrogram
import matplotlib.pyplot as plt
dendrogram(clus1, labels=RCDF[[1]].values.tolist())
plt.xlabel('hclust')
plt.ylabel('Distance')
plt.suptitle('Cluster Dendrogram', fontweight='bold', fontsize=14);
```



# Cluster Dendrogram




Note: Two broad clusters are formed. The clusters are primarily on the basis of AVG\_MTHLY\_SPEND

Euclidian Distance computation in this case is influenced by AVG\_MTHLY\_SPEND variable as the range of this variable is too large compared to the other variables


To avoid this problem, we should scale the variables used for clustering

# Scaling the dataset

```
## Hierarchical Clustering with Scaling
from sklearn.preprocessing import scale as scale
## scale function standardizes the values
## Note - The scale function calculates the
## Std. Dev assuming data is Sample
scaled_RCDF = scale(KRCDF.ix[:,2:7])
scaled_RCDF
```



```
## Note - Here we are scaling taking
## the data as Population
scaled_RCDF = KRCDF.ix[:,2:7].apply(
    lambda x: (x- x.mean())/x.std())
scaled_RCDF
```



	0	1	2	3	4
0	1.886	-1.240	1.528	-0.741	-1.380
1	0.788	-0.620	-0.655	1.281	1.725
2	0.788	1.861	1.528	-0.292	0.000
3	0.604	0.620	1.528	-0.741	0.000
4	0.421	1.240	-0.655	1.730	-0.345
5	-0.311	-0.620	-0.655	-0.741	1.380
6	-0.861	0.620	-0.655	1.505	-0.690
7	-0.861	-0.620	-0.655	-0.741	-1.035
8	-1.044	-1.240	-0.655	-0.517	-0.690
9	-1.410	0.000	-0.655	-0.741	1.035

Index	Avg_Mthly_Spend	No_Of_Visits	Apparel_Items	FnV_Items	Staples_Items
0	1.79	-1.18	1.45	-0.703	-1.31
1	0.747	-0.588	-0.621	1.21	1.64
2	0.747	1.77	1.45	-0.277	0
3	0.573	0.588	1.45	-0.703	0
4	0.4	1.18	-0.621	1.64	-0.327
5	-0.295	-0.588	-0.621	-0.703	1.31
6	-0.817	0.588	-0.621	1.43	-0.655
7	-0.817	-0.588	-0.621	-0.703	-0.982
8	-0.99	-1.18	-0.621	-0.49	-0.655
9	-1.34	0	-0.621	-0.703	0.982

# Understanding the Distance Calculation

```
import numpy as np
print(np.round(squareform(d_euc).tolist(),3))
```

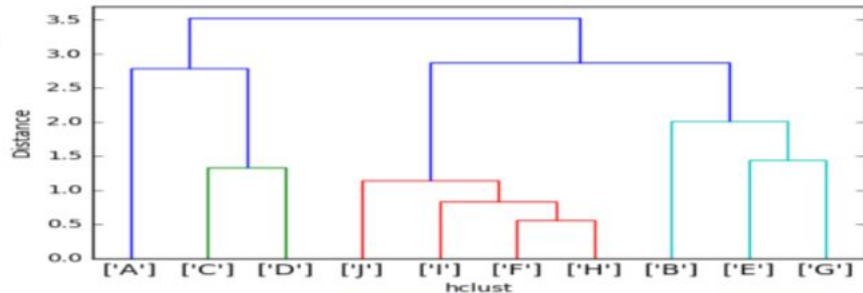
[ [ 0.	4.482	3.596	2.648	4.499	4.195	4.615	3.58	3.725	4.796]
[ 4.482	0.	4.045	3.66	2.843	2.327	3.184	3.798	3.578	3.127]
[ 3.596	4.045	0.	1.332	3.08	3.772	3.567	3.861	4.273	3.785]
[ 2.648	3.66	1.332	0.	3.377	3.007	3.526	3.085	3.386	3.206]
[ 4.499	2.843	3.08	3.377	0.	3.617	1.483	3.419	3.67	3.592]
[ 4.195	2.327	3.772	3.007	3.617	0.	3.343	2.477	2.293	1.308]
[ 4.615	3.184	3.567	3.526	1.483	3.343	0.	2.589	2.754	2.951]
[ 3.58	3.798	3.861	3.085	3.419	2.477	2.589	0.	0.767	2.23 ]
[ 3.725	3.578	4.273	3.386	3.67	2.293	2.754	0.767	0.	2.168]
[ 4.796	3.127	3.785	3.206	3.592	1.308	2.951	2.23	2.168	0. ]]

```
distance = clus2[:,2]
```

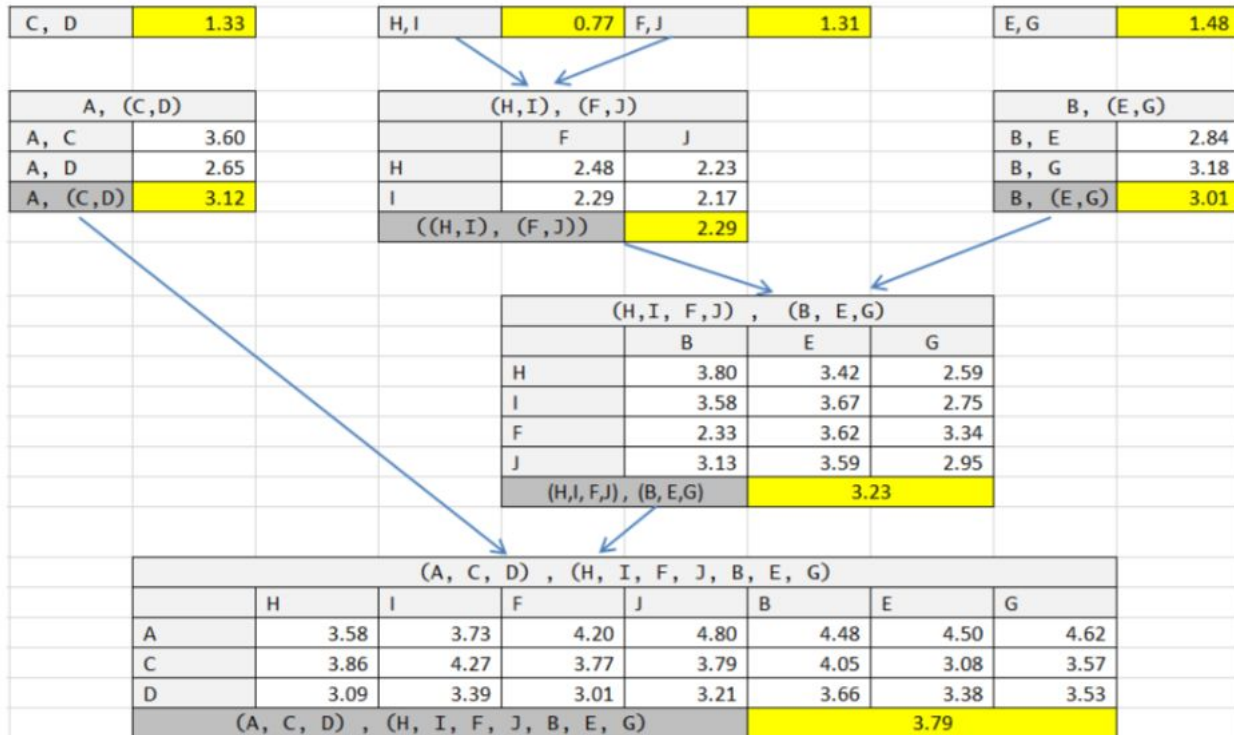
```
distance
```

```
array([ 0.76660834,  1.30817103,  1.33188797,  1.48252699,  2.29187962,  
       3.01368978,  3.12171991,  3.23053961,  3.78685213])
```

Cluster Dendrogram



Dist	A	B	C	D	E	F	G	H	I	J
A	0.00	4.48	3.60	2.65	4.50	4.20	4.62	3.58	3.73	4.80
B	4.48	0.00	4.05	3.66	2.84	2.33	3.18	3.80	3.58	3.13
C	3.60	4.05	0.00	1.33	3.08	3.77	3.57	3.86	4.27	3.79
D	2.65	3.66	1.33	0.00	3.38	3.01	3.53	3.09	3.39	3.206
E	4.50	2.84	3.08	3.38	0.00	3.62	1.48	3.42	3.67	3.59
F	4.20	2.33	3.77	3.01	3.62	0.00	3.34	2.48	2.29	1.31
G	4.62	3.18	3.57	3.53	1.48	3.34	0.00	2.59	2.75	2.95
H	3.58	3.80	3.86	3.09	3.42	2.48	2.59	0.00	0.77	2.23
I	3.73	3.58	4.27	3.39	3.67	2.29	2.75	0.77	0.00	2.17
J	4.80	3.13	3.79	3.21	3.59	1.31	2.95	2.23	2.17	0.00



# Profiling the clusters

```
## Profiling Step
```

```
RCDF['Clusters'] = cut_tree(clus2, 3)
```

```
clus_profile = RCDF.ix[:,2:8].groupby(['Clusters'], as_index=False).mean()
```

```
clus_profile
```

Index	Clusters	Avg_Mthly_Spend	No_Of_Visits	Apparel_Items	FnV_Items	Staples_Items
0	0	7833	4.667	1	1.667	2.667
1	1	5167	4.667	0	11	4.667
2	2	2375	3	0	1.25	4.5

# Partitioning Clustering

K Means Clustering

# K Means Clustering

K-Means is the most used, non-hierarchical clustering technique

It is not based on Distance

It is based on within cluster Variation, in other words Squared Distance from the Centre of the Cluster

The algorithm aims at segmenting data such that within cluster variation is reduced

# K Means Algorithm

Input: No of Clusters to be formed. (Say K)

Steps:

1. Assume K Centroids (for K Clusters)
2. Compute Euclidean distance of each objects with these Centroids
3. Assign the objects to clusters with shortest distance
4. Compute the new centroid (mean) of each cluster based on the objects assigned to each clusters. The K number of means obtained will become the new centroids for each cluster
5. Repeat step 2 to 4 till there is convergence
  - a) i.e. there is no movement of objects from one cluster to another
  - b) Or threshold number of iterations have occurred



# K-means advantages

K-means is superior technique compared to Hierarchical technique as it is less impacted by outliers

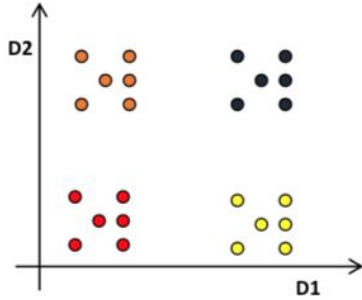
Computationally it is more faster compared to Hierarchical

Preferable to use on interval or ratio-scaled data as it uses Euclidian distance... desirable to avoid using on ordinal data

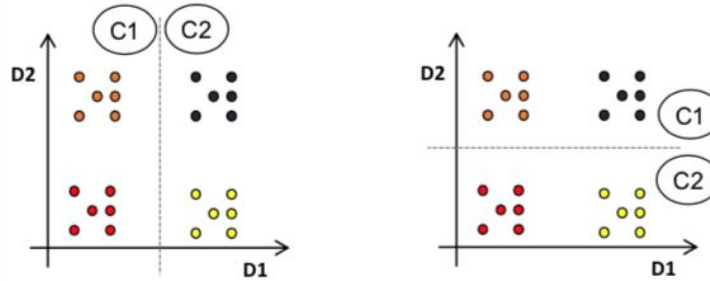
Challenge – Number of clusters are to be pre-defined and to be provided as input to the process

# Why find optimal No. of Clusters?

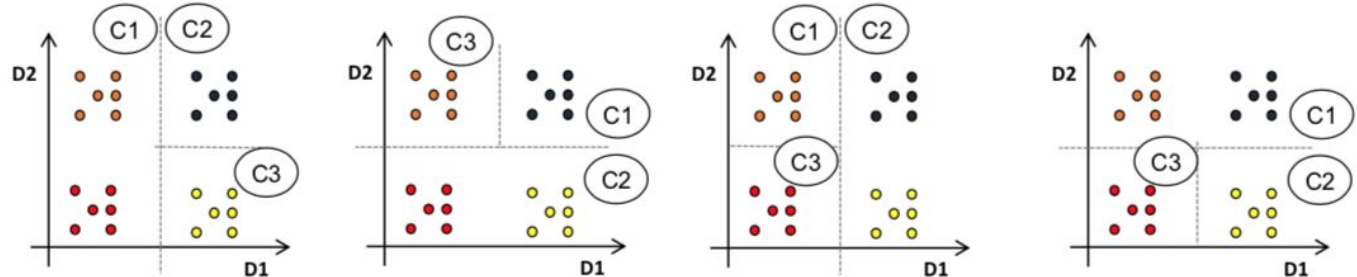
## Data to be clustered



### Two Clusters – 2 possible solution



### Three Clusters – Multiple possible solution

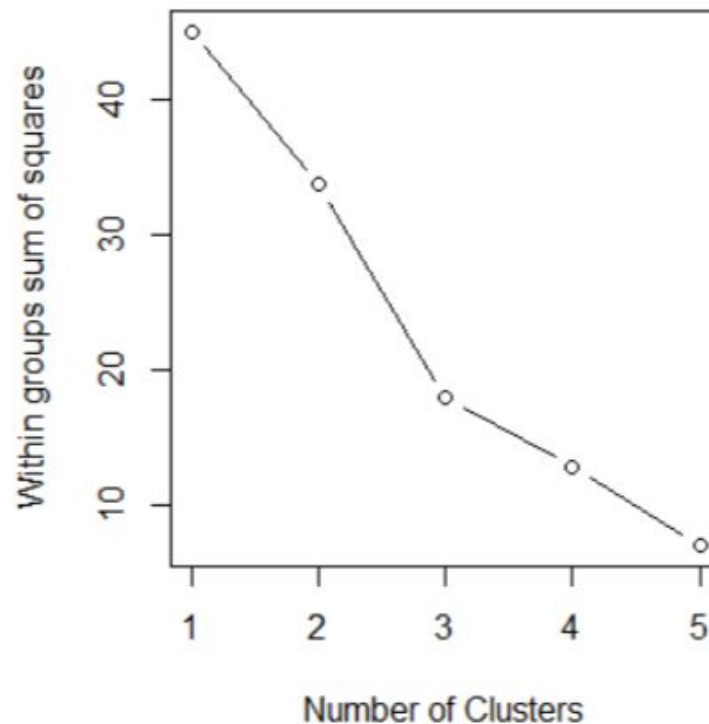


# R code to get Optimal No. of Clusters (In Class Exercise: 10m Python)

Identifying the optimal number of clusters form

```
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(scaled.RCDF, nc=5)
```

<http://www.r-statistics.com/2013/08/k-means-clustering-from-r-in-action/>



# Using NbClust to get optimal No. of Clusters

## Identifying the optimal number of clusters

```
## install.packages("NbClust")
```

```
library(NbClust)
```

```
set.seed(1234)
```

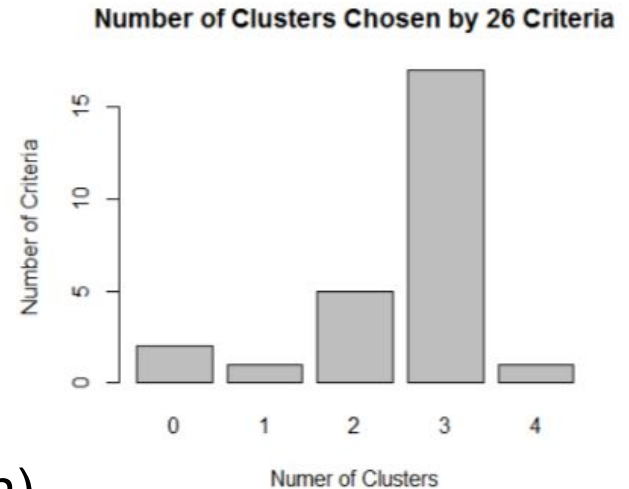
```
nc <- NbClust(KRCDF[,c(-1,-2)], min.nc=2, max.nc=4, method="kmeans")
```

```
table(nc$Best.n[1,])
```

```
barplot(table(nc$Best.n[1,]),
```

```
      xlab="Numer of Clusters", ylab="Number of Criteria",
```

```
      main="Number of Clusters Chosen by 26 Criteria")
```



(In Class Exercise: 10 min Equivalent Python)

# K Means Clustering R Code

```
?kmeans
```

```
kmeans.clus = kmeans(x=scaled.RCDF, centers = 3, nstart = 25)
```

```
## x = data frame to be clustered
```

```
## centers = No. of clusters to be created
```

```
## nstart = No. of random sets to be used for clustering
```

```
kmeans.clus
```

```
K-means clustering with 3 clusters of sizes 4, 3, 3
```

```
Cluster means:
```

	Avg_Mthly_Spend	No_of_Visits	Apparel_Items	FnV_Items	Staples_Items
1	-0.8600931	-0.5883484	-0.621059	-0.6500980	0.1636634
2	1.0367452	0.3922323	1.449138	-0.5612868	-0.4364358
3	0.1100456	0.3922323	-0.621059	1.4280842	0.2182179

```
Clustering vector:
```

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

```
within cluster sum of squares by cluster:
```

```
[1] 5.256752 6.514105 6.126217  
(between_SS / total_SS = 60.2 %)
```

```
Available components:
```

```
[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss" "betweenss"  
[7] "size"         "iter"         "ifault"
```

(In Class Exercise: 10  
min Equivalent  
Python)

# Profiling the clusters

## profiling the clusters

```
KRCDF$Clusters <- kmeans.clus$cluster
```

```
aggr = aggregate(KRCDF[, -c(1, 2, 8)], list(KRCDF$Clusters), mean)
```

```
clus.profile <- data.frame( Cluster=aggr[, 1],  
                           Freq=as.vector(table(KRCDF$Clusters)),  
                           aggr[, -1])
```

View(clus.profile)

Cluster	Freq	Avg_Mthly_Spend	No_Of_Visits	Apparel_Items	FnV_Items	Staples_Items
1	4	2375.000	3.000000	0	1.250000	4.500000
2	3	7833.333	4.666667	1	1.666667	2.666667
3	3	5166.667	4.666667	0	11.000000	4.666667

# Next steps after clustering

Clustering provides you with clusters in the given dataset

Clustering does not provide you rules to classify future records

To be able to classify future records you may do the following

Build Discriminant Model on Clustered Data

Build Classification Tree Model on Clustered Data

<https://towardsdatascience.com/the-5-clustering-algorithms-data-scientists-need-to-know-a36d136ef68>

# Clustering applications - Astrostatistics

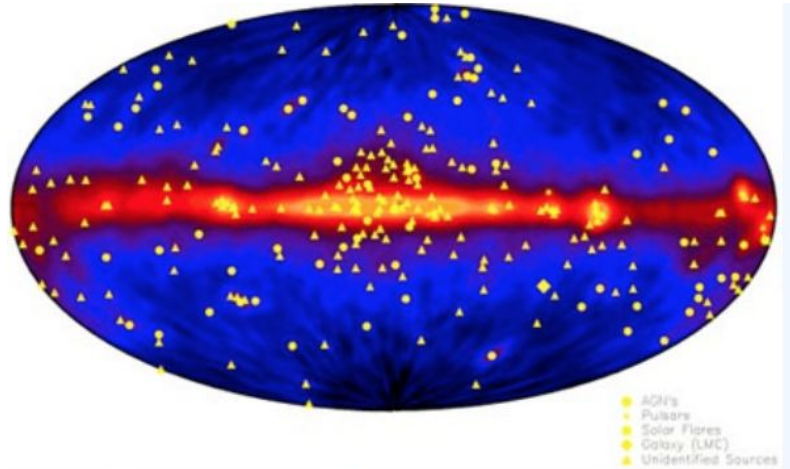


Image: <http://science.hq.nasa.gov>

## THREE TYPES OF GAMMA-RAY BURSTS

SOMA MUKHERJEE,<sup>1,2,3</sup> ERIC D. FEIGELSON,<sup>4</sup> GUTTI JOGESH BABU,<sup>5</sup> FIONN MURTAGH,<sup>6,7</sup>  
CHRIS FRALEY,<sup>8</sup> AND ADRIAN RAFTERY<sup>8</sup>

*Received 1998 February 9; accepted 1998 June 25*



# “One” schematic for addressing problems in machine learning...

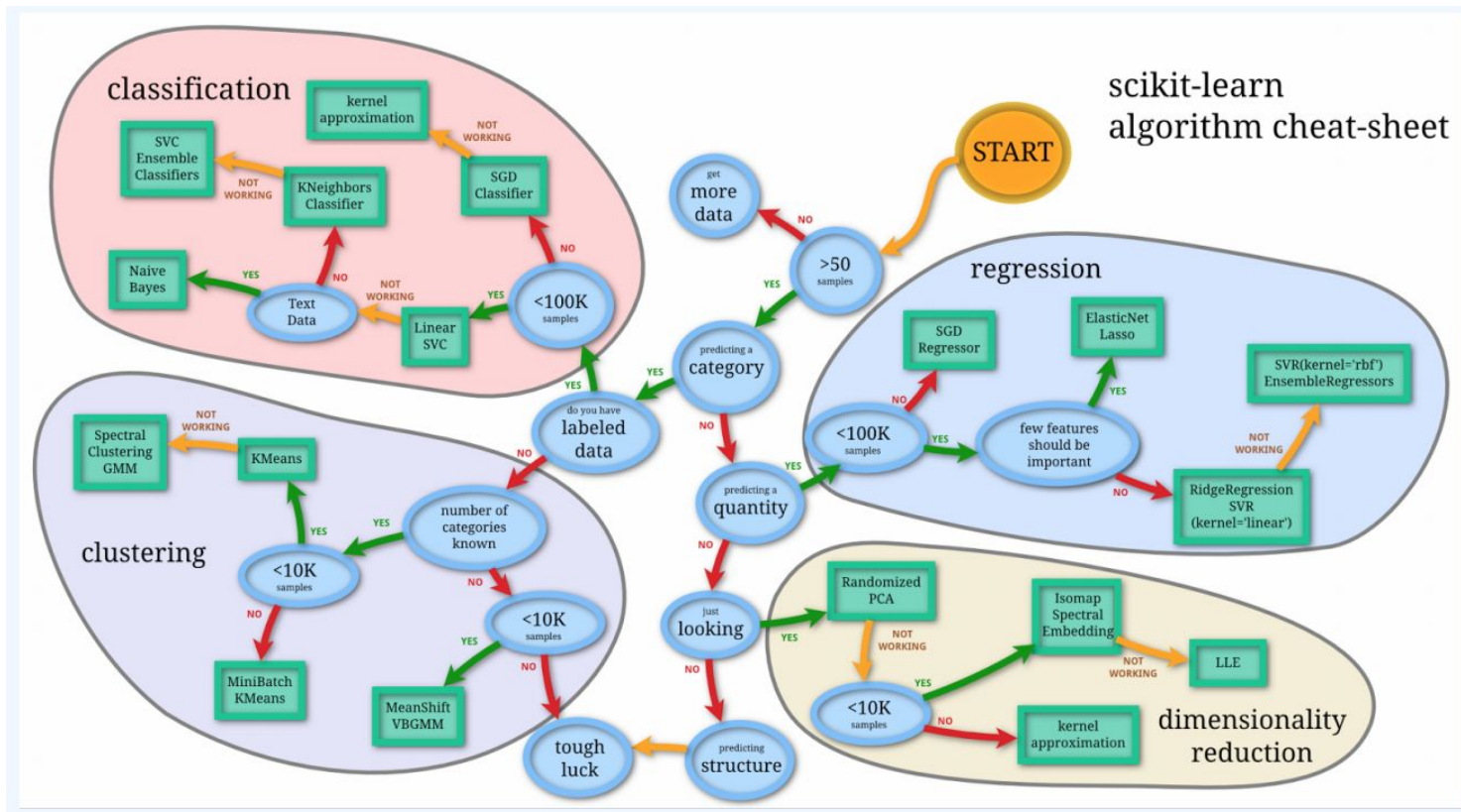


Image: Andy's Computer Vision and Machine Learning Blog <http://peekaboo-vision.blogspot.com>

# References

## References

Chapter 9: Cluster Analysis (<http://www.springer.com>)

Google search : “www.springer.com cluster analysis chapter 9”

[http://sites.stat.psu.edu/~ajw13/stat505/fa06/19\\_cluster/09\\_cluster\\_wards.html](http://sites.stat.psu.edu/~ajw13/stat505/fa06/19_cluster/09_cluster_wards.html) •

[https://home.deib.polimi.it/matteucc/Clustering/tutorial\\_html/](https://home.deib.polimi.it/matteucc/Clustering/tutorial_html/)

Thank you! Questions