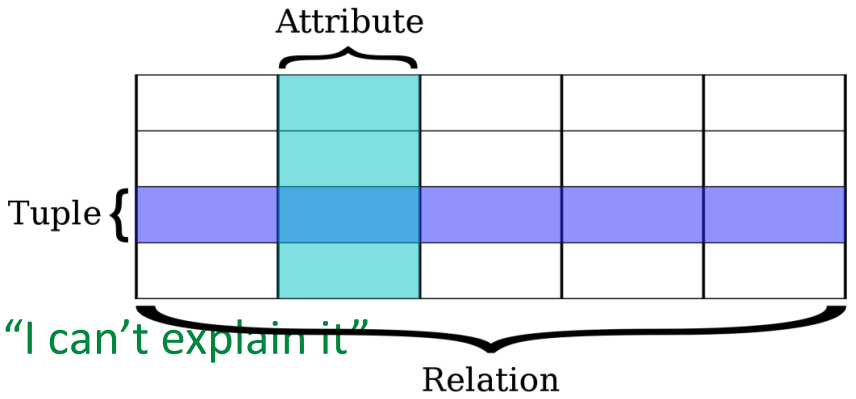
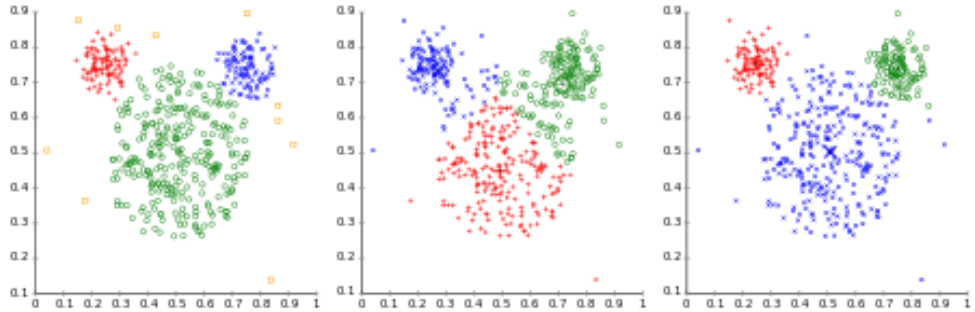
Clustering

Praphul Chandra

Clustering 101



- Motivation
  - Transaction data (Customer – Product matrix)
  - Test measurements from a fab plant for chips
  - Segmentation, Pre-processing, Multimodal distributions, “I can’t explain it”



- Clustering
  - Find elements (rows, tuples) which are similar.
  - Finding “areas” in space where data is concentrated
  - WYSIWYG : What You Select Is What You Get

- When are two elements / rows similar?
  - A measure of (dis)similarity.
  - Which dimensions (attributes) are relevant?
  - Normalization?
  - How many clusters?

Similarity, Dissimilarity & Distance

Measures

Distance in multiple dimensions

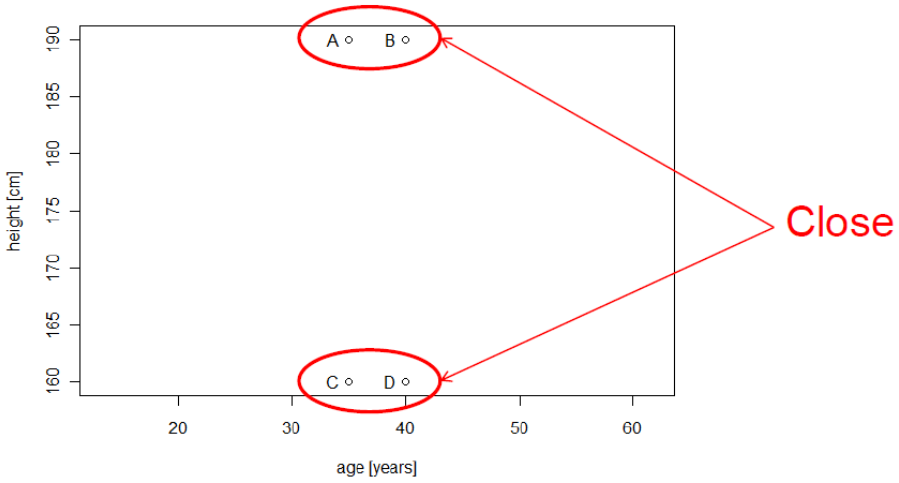
Euclidean distance:

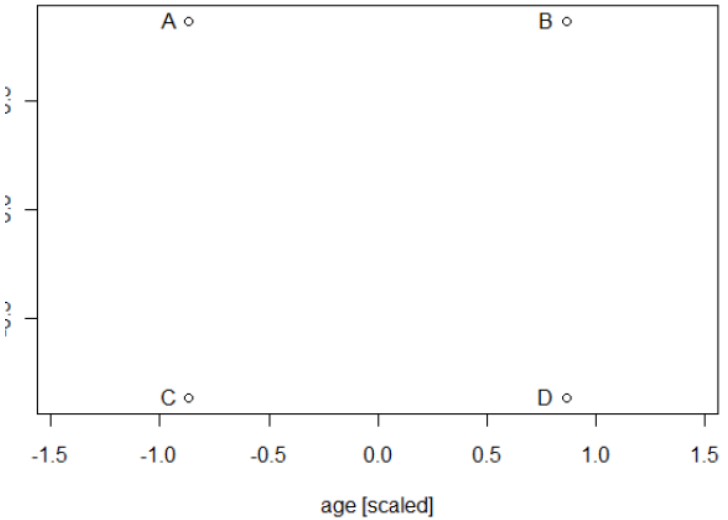
$$d(i,j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + ... + (x_{ip} - x_{jp})^2}$$

Person	Age [years]	Height [cm]
A	35	190
B	40	190
C	35	160
D	40	160

Person	Age [years]	Height [feet]
A	35	6.232
B	40	6.232
C	35	5.248
D	40	5.248

Person	Age [scaled]	Height [scaled]
A	-0.87	0.87
B	0.87	0.87
C	-0.87	-0.87
D	0.87	-0.87





Distance in multiple dimensions : To Scale or Not

- Distance depends on scale
- If variables are not scaled
  - Variable with largest range has most weight
- If variables are scaled
  - Every variable gets equal weight
  - Similar alternative is re-weighing

$$d(i,j) = \sqrt{w_1(x_{i1} - x_{j1})^2 + w_2(x_{i2} - x_{j2})^2 + ... + w_p(x_{ip} - x_{jp})^2}$$

- Scale
  - If variables measure different units (kg, meter, sec,…)
  - If you explicitly want to have equal weight for each variable
  - Default
- Don’t scale
  - if units are the same for all variables

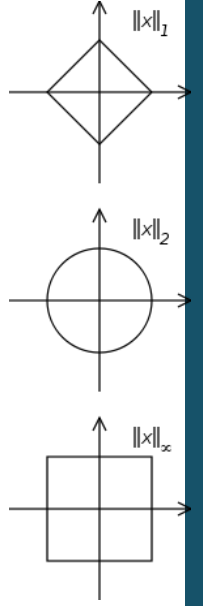
Measures of Dissimilarity : distance

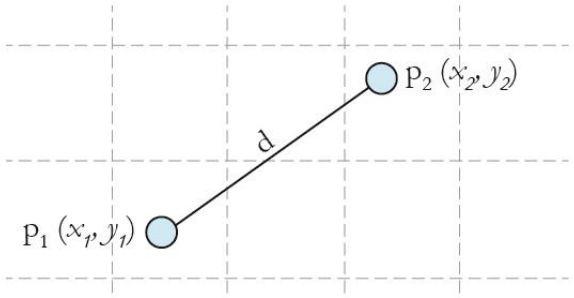
$$||x_i||_1 = \sum_p |x_j|$$

$$||x_i||_2 = \left(\sum_p |x_j|^2\right)^{\frac{1}{2}}$$

$$||x_i||_z = \left(\sum_p |x_j|^z\right)^{\frac{1}{z}}$$

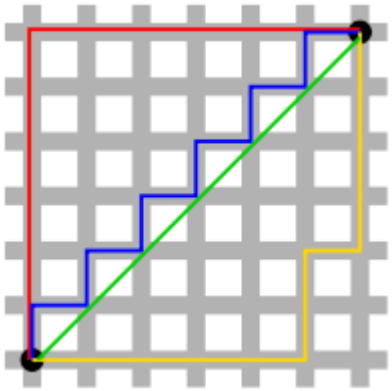
$$||x_i||_\infty = \max_p |x_j|$$





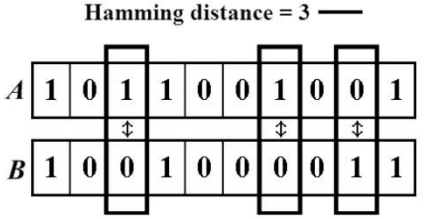
Euclidean distance: L2 distance

$$d(i,j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + ... + (x_{ip} - x_{jp})^2}$$



Manhattan distance: a.k.a. L1 distance a.k.a. Taxicab

$$d(i,j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + ... + |x_{ip} - x_{jp}|$$



Special cases of Minkowski distance:

$$d(i,j) = (|x_{i1} - x_{j1}|^q + |x_{i2} - x_{j2}|^q + ... + |x_{ip} - x_{jp}|^q)^{\frac{1}{q}}$$

Maximum distance: a.k.a. L_∞ distance a.k.a. supremum (max. dist. in any dimension)

$$d(i,j) = (|x_{i1} - x_{j1}|^\infty + |x_{i2} - x_{j2}|^\infty + ... + |x_{ip} - x_{jp}|^\infty)^{\frac{1}{\infty}} =$$
$$= max_{k=1}^{p} |x_{ik} - x_{jk}|$$

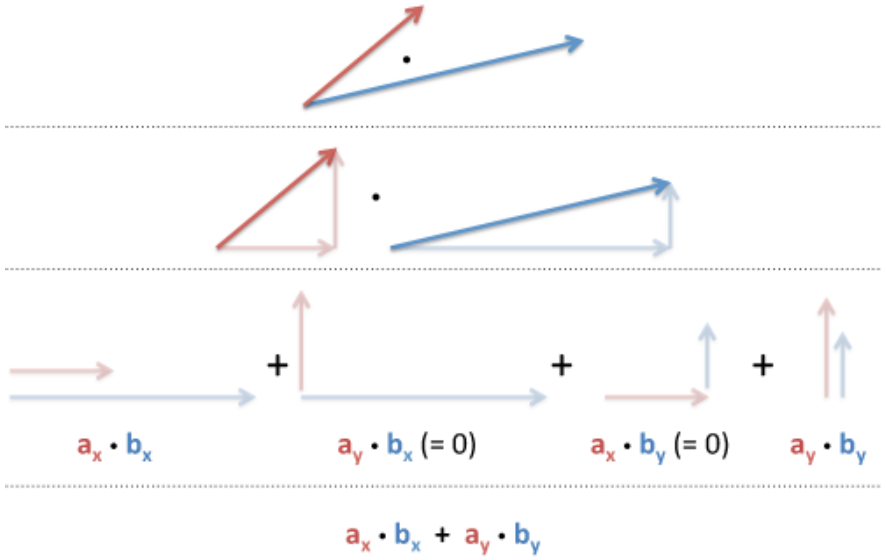
Measures of Similarity : The Dot Product

- Inner Product
  - Element wise product of two vectors
  - a.k.a. scalar product

$$\begin{pmatrix} 3 \\ -2 \\ 6 \end{pmatrix} \cdot \begin{pmatrix} 2 \\ 3 \\ -5 \end{pmatrix} = 3\times 2+(-2)\times 3+6\times(-5) = 6-6-30 = -30.$$

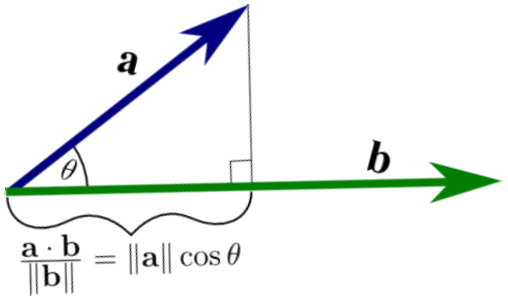
$$\mathbf{w} \in \mathbb{R}^p, \mathbf{x} \in \mathbb{R}^p$$

$$\mathbf{w}^T\mathbf{x} = \sum_{j=1}^{p} w_j x_j = \langle \mathbf{w}, \mathbf{x} \rangle$$



- Projection Product
  - Vector : Magnitude (Length) & Direction

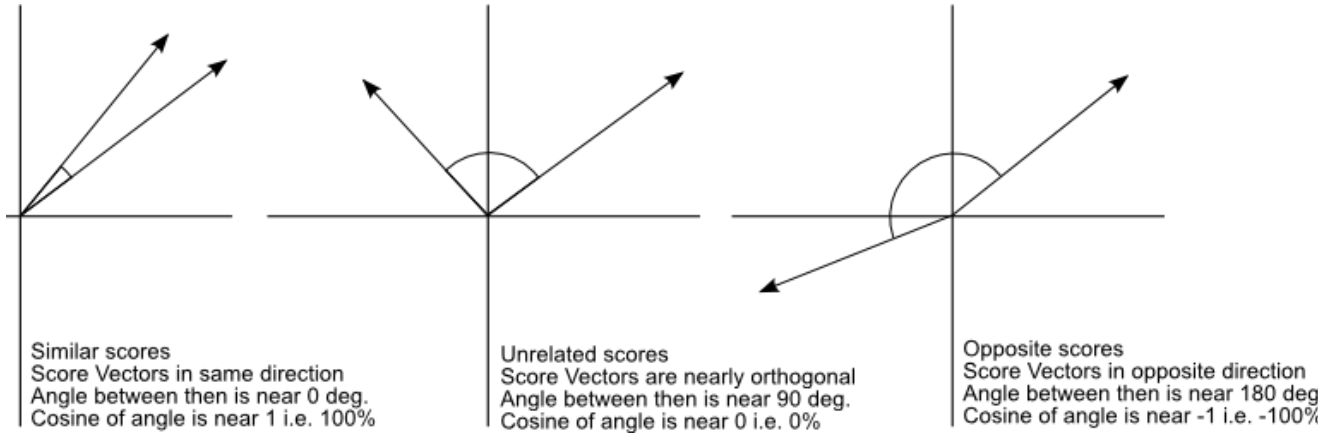
$$\begin{aligned} \mathbf{w}^T\mathbf{x} &= ||\mathbf{w}||||\mathbf{x}|| \cos\theta \\ \theta = 90 &\Rightarrow \mathbf{w}^T\mathbf{x} = 0 \\ \theta = 0 &\Rightarrow \mathbf{w}^T\mathbf{x} = ||\mathbf{w}||||\mathbf{x}|| \\ \mathbf{w}^T\mathbf{w} &= ||\mathbf{w}||^2 \end{aligned}$$



$$x_w = ||\mathbf{x}|| \cos\theta = \frac{||\mathbf{x}||||\mathbf{w}|| \cos\theta}{||\mathbf{w}||} = \frac{\mathbf{x}^T\mathbf{w}}{||\mathbf{w}||} = \mathbf{x}^T\left(\frac{\mathbf{w}}{||\mathbf{w}||}\right) = \mathbf{x}^T\hat{\mathbf{w}}$$

Measures of Similarity : Cosine Similarity

- Cosine of angle between two vectors
  - Distance between vectors captured by the cosine of the angle x between them.
- a.k.a. normalized inner product
  - Denominator involves the lengths of the vectors
- Dot-products and Cosine Similarity
  - Relevant in information retrieval
  - Documents (query) as vectors of words
  - Two documents similar if they contain the same words
  - Does size of the document matter?
    - Yes: Dot Product
    - No: Cosine similarity



$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|}$$

Measures of dissimilarity : Categorical Variables

- Categorical Variables
  - One hot encoding
- Hamming Distance
  - Also useful for strings

d("karolin“, "kathrin“) = 3.
d("karolin“, "kerstin“) = 3.
d(1011101, 1001001) = 2.
d(2173896, 2233796) = 3.

- Coefficients
  - SMC: Simple Matching Coeffient
  - Jaccard coefficient
  - Dice coefficient
  - Custom Variants: if one of the states is more important or more valuable than the other.
  - Distance = 1 - coefficient

a
1
2
3

→

a	a1	a2	a3
1	1	0	0
2	0	1	0
3	0	0	1

p = 1 0 0 0 0 0 0 0 0 0
q = 0 0 0 0 0 0 1 0 0 1

M_{01} = 2 (the number of attributes where p was 0 and q was 1)
M_{10} = 1 (the number of attributes where p was 1 and q was 0)
M_{00} = 7 (the number of attributes where p was 0 and q was 0)
M_{11} = 0 (the number of attributes where p was 1 and q was 1)

$$\text{SMC} = \frac{M_{11}+M_{00}}{M_{11}+M_{00}+M_{01}+M_{10}}$$

$$\text{Jaccard} = \frac{M_{11}}{M_{11}+M_{01}+M_{10}}$$

$$\text{Dice} = \frac{2M_{11}}{2M_{11}+M_{01}+M_{10}}$$

Measures of similarity : Ordinal Variables

- Normalize Ranks
  - Normalize Rank; Treat as Numeric

$$z_{if} = \frac{r_{if} - 1}{M_f - 1}$$

- Look up table

$$\begin{bmatrix} & 1 & 2 & 3 \\ 1 & 0 & 1 & 4 \\ 2 & 1 & 0 & 1 \\ 3 & 4 & 1 & 0 \end{bmatrix}$$

Distance in multiple dimensions : Which distance measure to use?

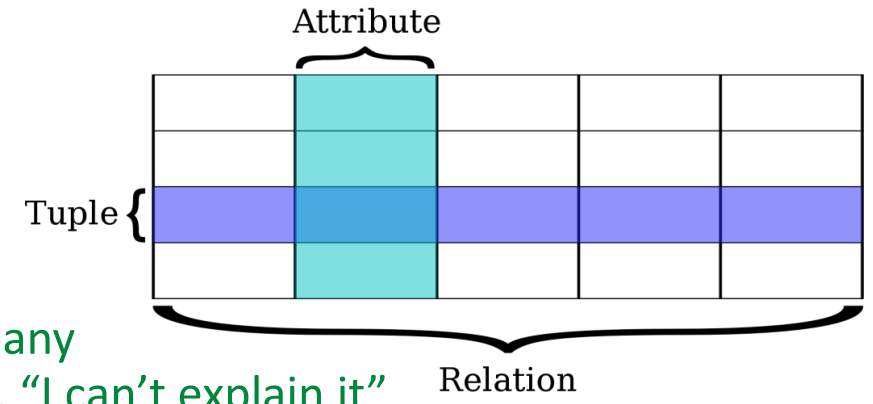
- Gower Distance
  - Idea: Use distance measure between 0 and 1 for each variable / dimension / feature $d_{ij}^{(f)}$
  - Aggregate: $d(i,j) = \frac{1}{p}\sum_{i=1}^{p} d_{ij}^{(f)}$

- Variable-type specific distance measure
  - Numeric Variables : Normalized Manhattan (L1 distance) $d_{ij}^{(f)} = \frac{|x_{if} - x_{jf}|}{R_f}$
    - x_{if}: Value for object i in variable f
    - R_f: Range of variable f for all objects
  - Categorical Variables : Dice coefficient
  - Ordinal Variables: Rank Normalized

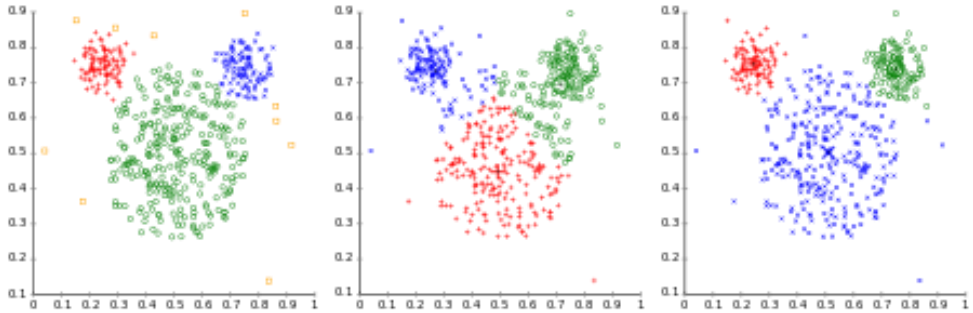
Algorithms

Clustering

Clustering 101

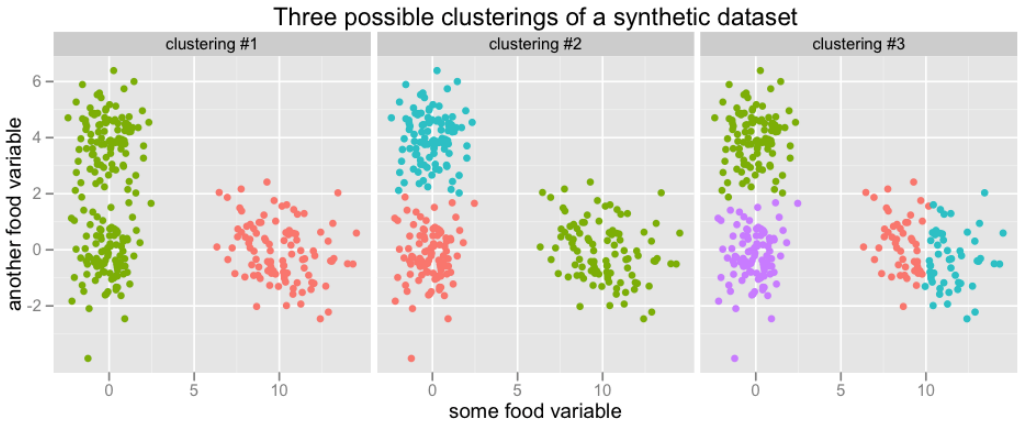


- Motivation
  - Transaction data (Customer – Product matrix)
  - Test measurements from a fab plant for chips
  - Predict shipping weight of a package for a logistics company
  - Segmentation, Pre-processing, Multimodal distributions, “I can’t explain it”



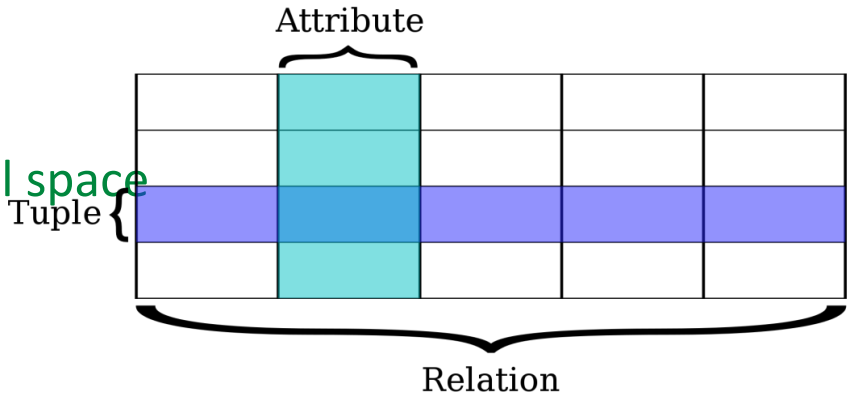
- Clustering
  - Find elements (rows, tuples) which are similar.
  - Finding “areas” in space where data is concentrated
  - WYSIWYG : What You Select Is What You Get

- When are two elements / rows similar?
  - A measure of (dis)similarity.
  - Which dimensions (attributes) are relevant?
  - Normalization?
  - How many clusters?



Dis-similarity / Distance based Clustering Framework

- Input
  - Data
  - Each data element (row, tuple) lives in a p-dimensional space
  - A dis-(similarity) / distance measure d()
  - k (Number of clusters)



$$X \in \mathbb{R}^{n \times p}$$

- Output
  - X partitioned into k-clusters;
  - Cluster-id (colour) for each element (row, tuple)

X *partitioned* into k-clusters; cluster-id for each $\mathbf{x} \in X$

$$C_1 \cup C_2 \cup \ldots C_k = \{1, 2, \ldots n\}$$

$$C_1 \cap C_2 \cap \ldots C_k = \phi$$

- Measure
  - Minimize within-sum-of-squares

k-means

- Initialize
  - Pick k data points randomly from X : centroids
- Iterate
  - Assign each data point to the closest / most-similar centroid
  - For each cluster, update centroid
  - Repeat
- Terminate when “change in within cluster variation” < threshold
  - Amount by which elements in the same clusters are different

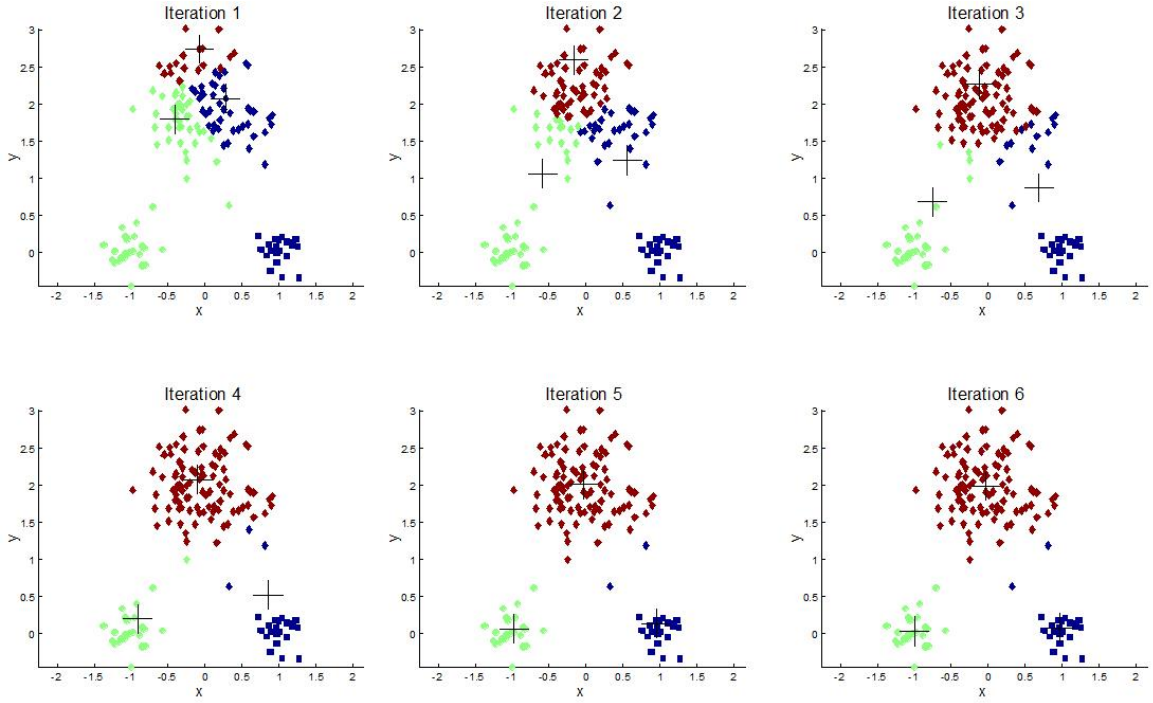
$$W(C_z) = \frac{1}{|C_z|} \sum_{i,j \in C_z} ||x_i - x_j||_2^2$$

$$= \frac{1}{|C_z|} \sum_{i,j \in C_z} \sum_{d=1}^{p} (x_{id} - x_{jd})^2$$

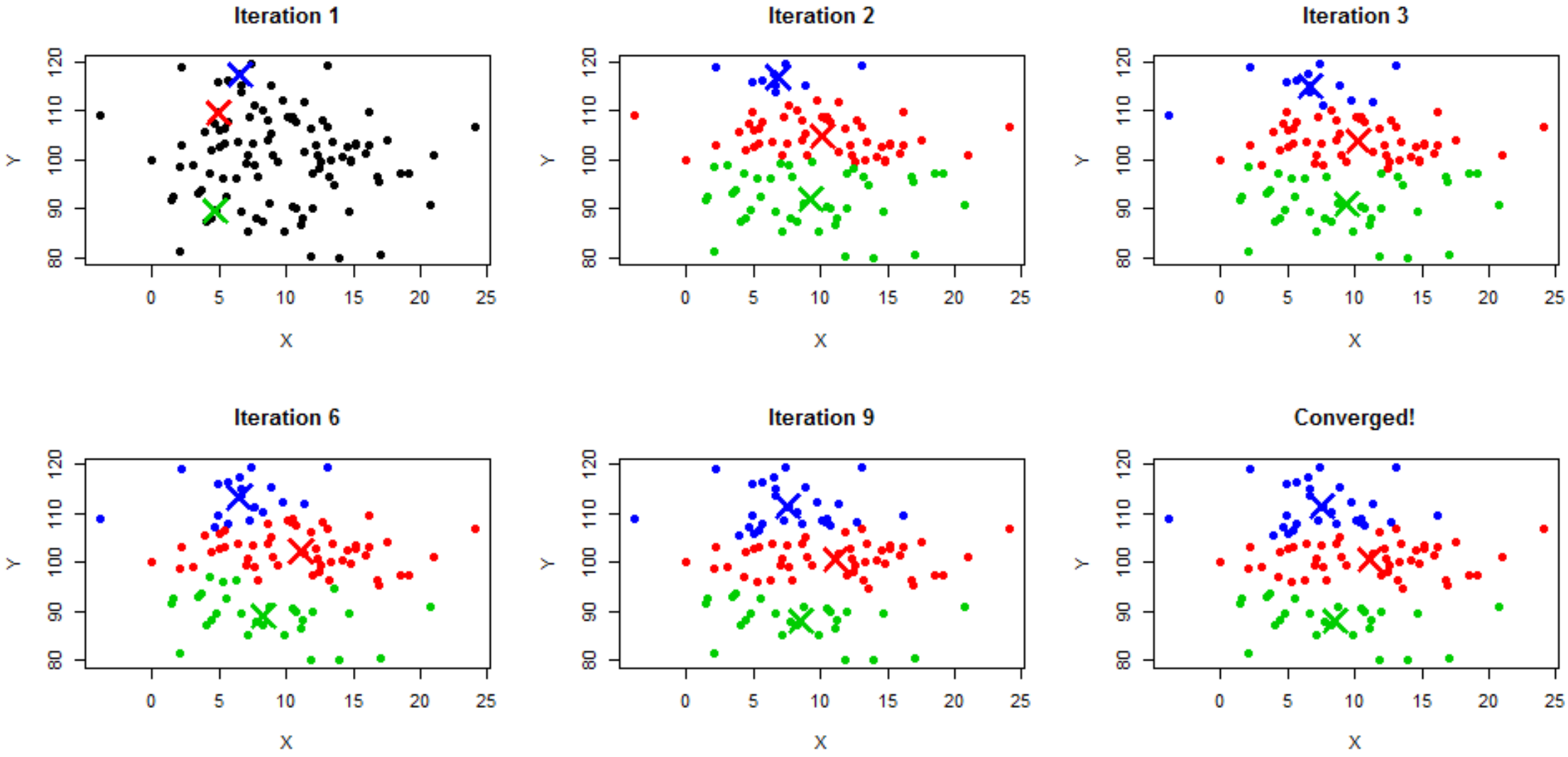
$$W(C) = \sum_{z=1}^{k} W(C_z)$$

Algorithm k-means (k , D)

1 Choose random k data points as initial Clusters Mean (cluster centers)
2 Repeat
3 For each data point x from D
4 Compute the distance between x and each cluster mean (centroid)
5 Assign x to the nearest cluster
6 End for
7 Re-compute the mean for current cluster collections
8 Until reaching stable clusters(current clusters means equals last clusters means)

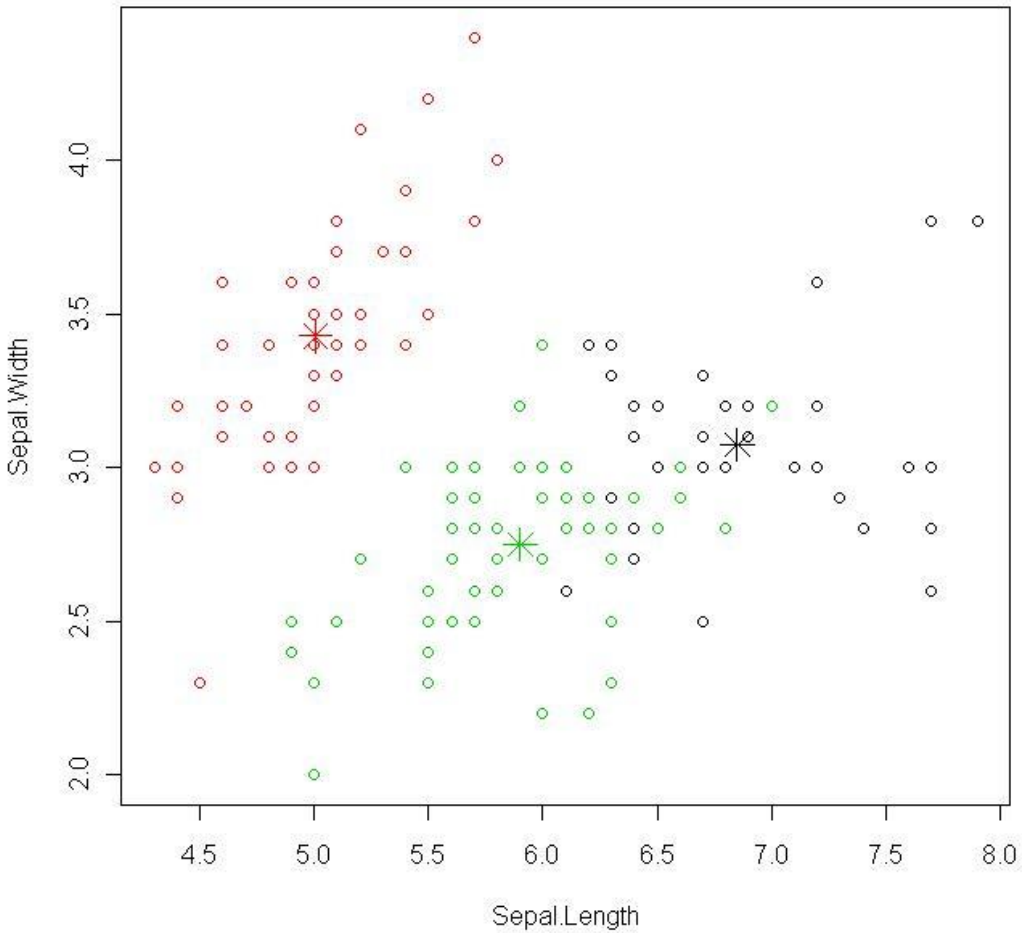


Are the clusters meaningful?



Example

```
> (kc <- kmeans(newiris, 3))
K-means clustering with 3 clusters of sizes 38, 50, 62

Cluster means:
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1     6.850000    3.073684     5.742105    2.071053
2     5.006000    3.428000     1.462000    0.246000
3     5.901613    2.748387     4.393548    1.433871

Clustering vector:
  [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 [30] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 1 3 3 3 3 3 3
 [59] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 3 3 3 3 3 3 3 3 3 3
 [88] 3 3 3 3 3 3 3 3 3 3 3 3 3 1 3 1 1 1 1 3 1 1 1 1 1 1 3 3 1
[117] 1 1 1 3 1 3 1 3 1 1 3 3 1 1 1 1 1 3 1 1 1 1 3 1 1 1 3 1 1
[146] 1 3 1 1 3

Within cluster sum of squares by cluster:
[1] 23.87947 15.15100 39.82097

Available components:
[1] "cluster"  "centers"  "withinss" "size"
```

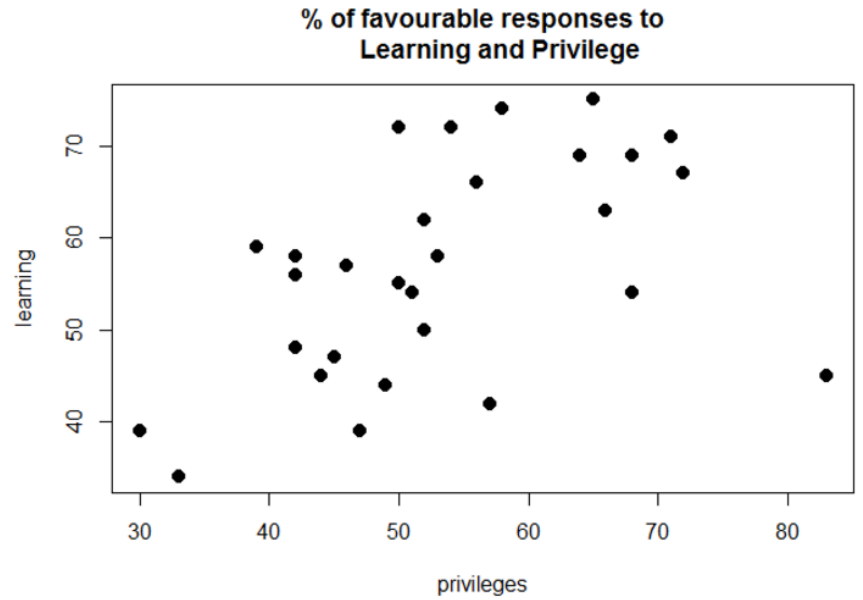



```
> table(iris$Species, kc$cluster)

              1  2  3
  setosa      0 50  0
  versicolor  2  0 48
  virginica  36  0 14
```

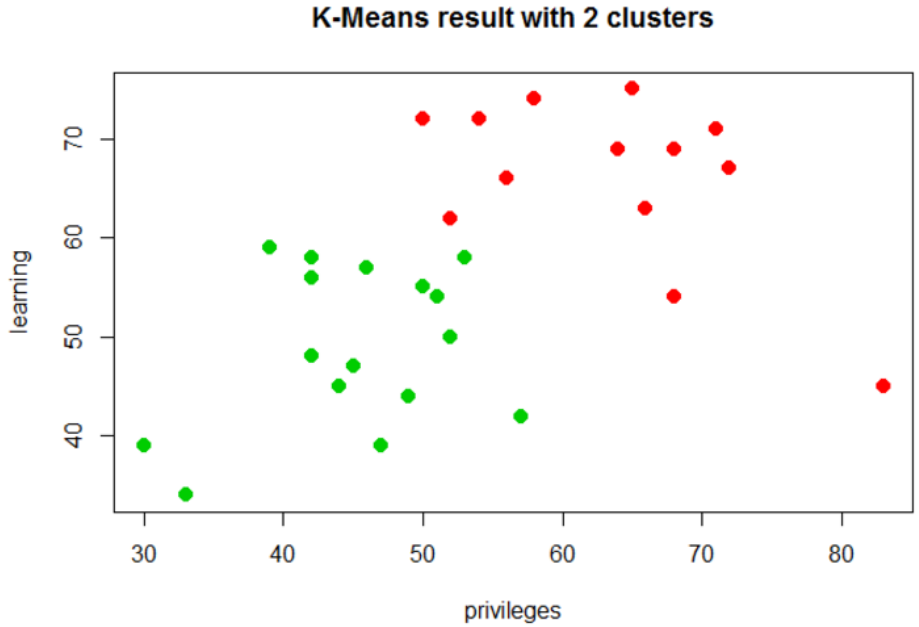
Example

```
## 'data.frame':    30 obs. of  7 variables:
##  $ rating    : num  43 63 71 61 81 43 58 71 72 67 ...
##  $ complaints: num  51 64 70 63 78 55 67 75 82 61 ...
##  $ privileges: num  30 51 68 45 56 49 42 50 72 45 ...
##  $ learning  : num  39 54 69 47 66 44 56 55 67 47 ...
##  $ raises    : num  61 63 76 54 71 54 66 70 71 62 ...
##  $ critical  : num  92 73 86 84 83 49 68 66 83 80 ...
##  $ advance   : num  45 47 48 35 47 34 35 41 31 41 ...
```

```
# Subset the attitude data
dat = attitude[,c(3,4)]

# Plot subset data
plot(dat, main = "% of favourable responses to
     Learning and Privilege", pch =20, cex =2)
```

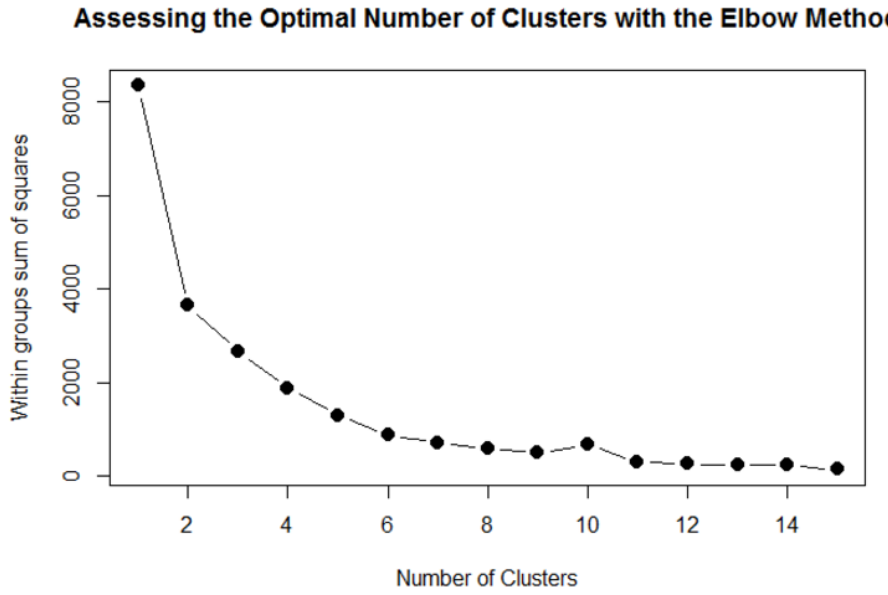



```
# Perform K-Means with 2 clusters
set.seed(7)
km1 = kmeans(dat, 2, nstart=100)

# Plot results
plot(dat, col =(km1$cluster +1) , main="K-Means result with 2 clusters", pch=20, cex=2)
```



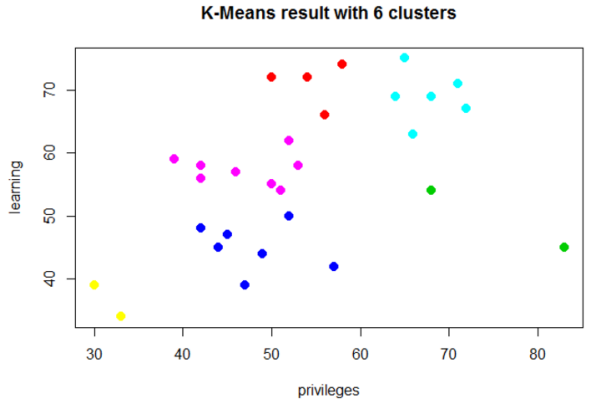
Example (cont'd)

```
# Check for the optimal number of clusters given the data

mydata <- dat
wss <- (nrow(mydata)-1)*sum(apply(mydata,2,var))
  for (i in 2:15) wss[i] <- sum(kmeans(mydata,
                                       centers=i)$withinss)
plot(1:15, wss, type="b", xlab="Number of Clusters",
     ylab="Within groups sum of squares",
     main="Assessing the Optimal Number of Clusters with the Elbow Method",
     pch=20, cex=2)
```




```
# Perform K-Means with the optimal number of clusters identified from the Elbow
set.seed(7)
km2 = kmeans(dat, 6, nstart=100)

# Examine the result of the clustering algorithm
km2
```

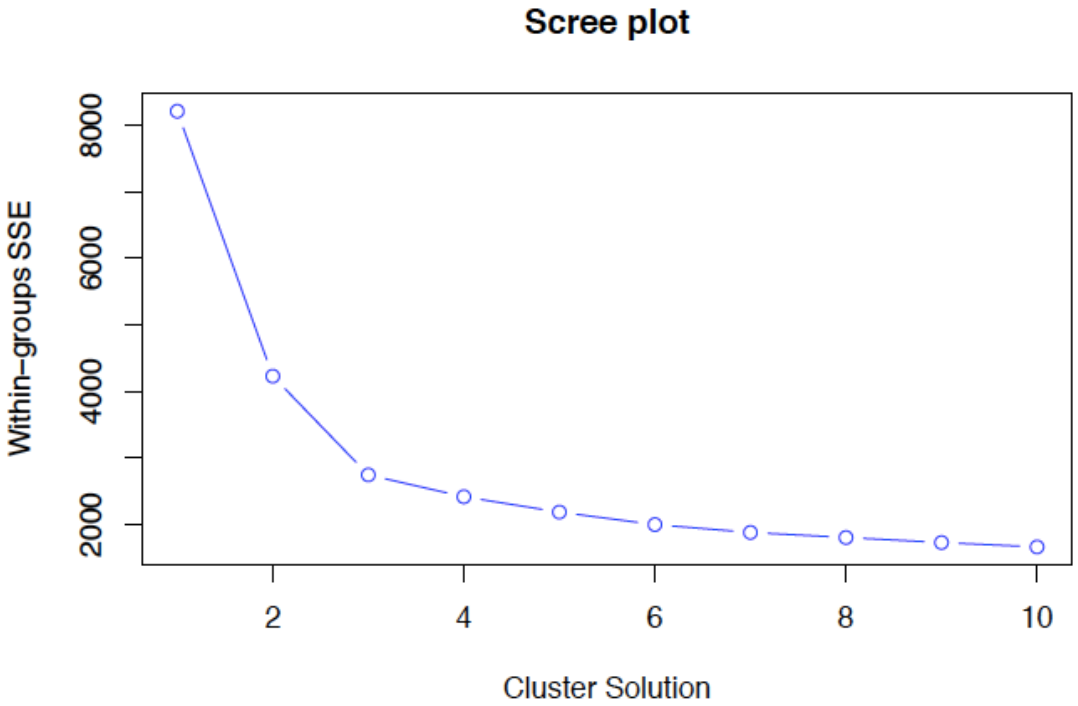
```
## K-means clustering with 6 clusters of sizes 4, 2, 8, 6, 8, 2
##
## Cluster means:
##   privileges learning
## 1   54.50000   71.000
## 2   75.50000   49.500
## 3   47.62500   45.250
## 4   67.66667   69.000
## 5   46.87500   57.375
## 6   31.50000   36.500
##
## Clustering vector:
##  [1] 6 5 4 3 1 3 5 5 4 3 5 3 3 2 1 1 4 4 5 2 6 5 3 5 3 4 1 3 4 5
##
## Within cluster sum of squares by cluster:
## [1]  71.0000 153.0000 255.3750 133.3333 244.7500  17.0000
##  (between_SS / total_SS =  89.5 %)
##
```

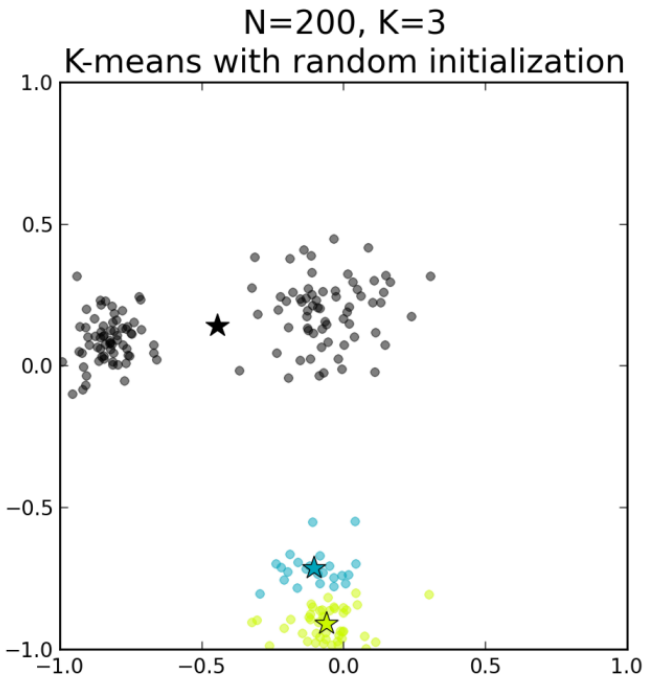


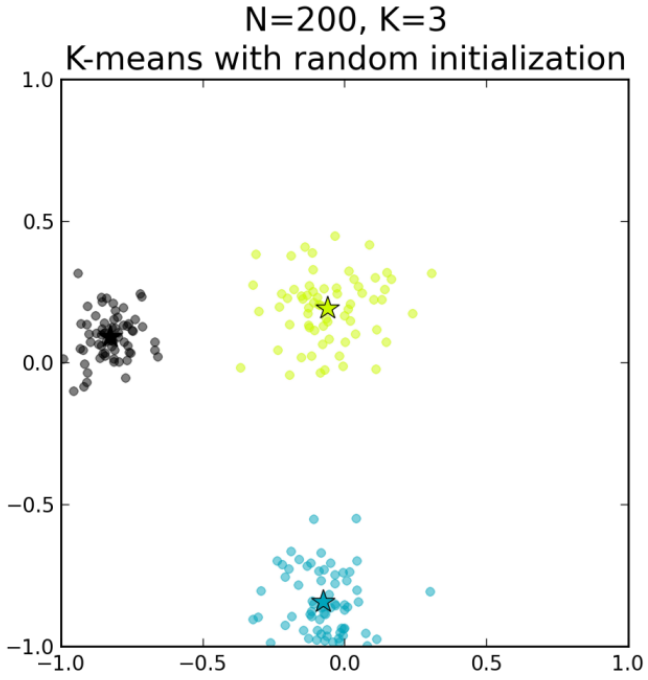
https://rpubs.com/FelipeRego/K-Means-Clustering

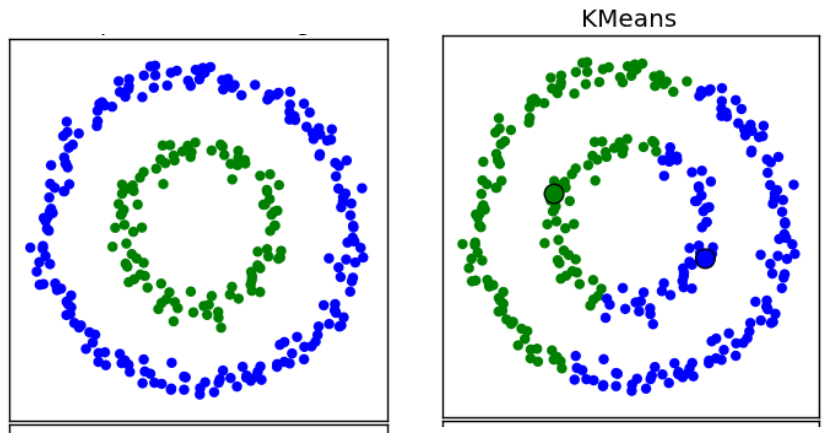
Limitations of k-means

- Hyperparameter : k
  - Not really a “limitation”
- Initial centroid at random
- Categorical (Mixed data?)
- Non-convex clusters







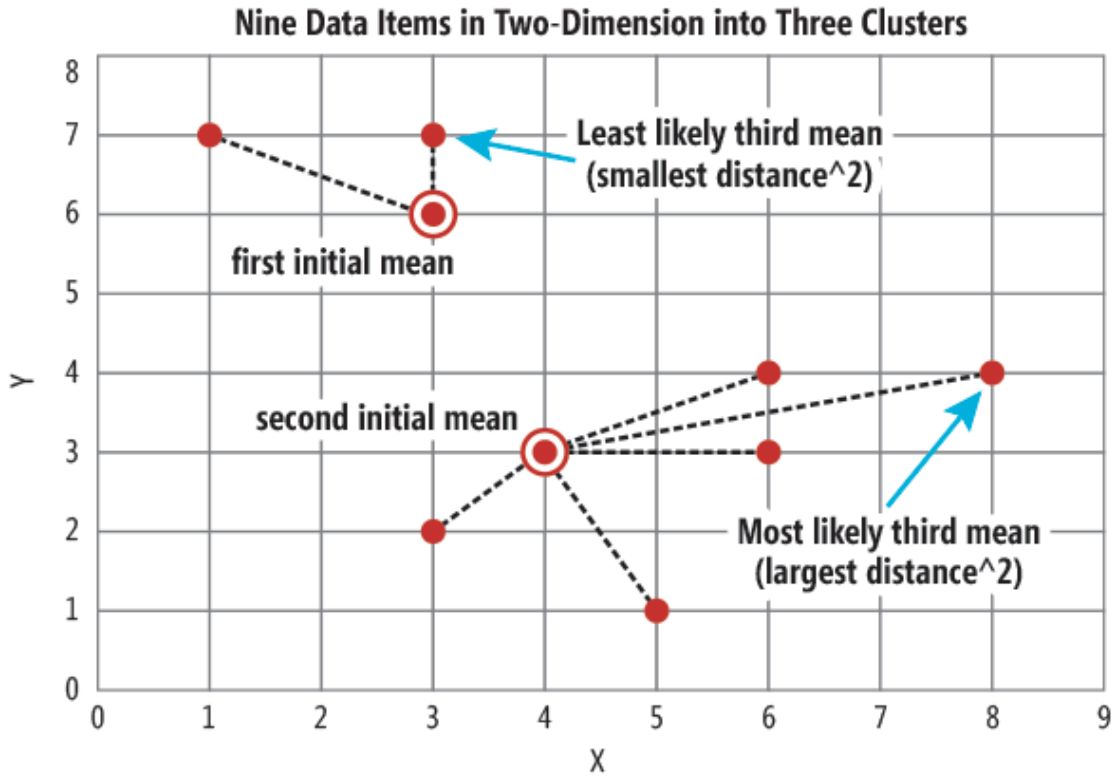


k-means++

- Motivation
  - Clusters in k-means depend on choice of initial cer
  - Each run (w/ different centroids) result in different
  - Can we do better?

- Key Idea: At Initialization,
  - Pick first centroid at random
  - For each x ε X; g(x) = d(x; $c_{nearest}$)
  - Pick x as next centroid with probability proportion
  - Repeat till all k centroids are picked

- Use k means with these initial centroids



Dealing with non-numeric data

```
Algorithm k-means (k , D)
1 Choose random k data points as initial Clusters Mean ( cluster centers)
2 Repeat
3     For each data point x from D
4         Compute the distance between x and each cluster mean (centroid)
5         Assign x to the nearest cluster
6     End for
7     Re-compute the mean for current cluster collections
8 Until reaching stable clusters(current clusters means equals last clusters
means)
```

- K-medoids
  - a.k.a. Partitioning around medoids (PAM)
- Takes custom distance matrix as input
  - e.g. output of a distance function (daisy() in R)
  - e.g. Gower distance metrics which works with mixed data type
- Uses median instead of mean to determine cluster centroids
  - a more *robust* estimate of a representative point than the mean
  - "Centroids" must be datapoints in the data set (medoids or exemplars)
- Computationally more expensive
- Others
  - Any clustering algorithm which takes dissimilarity (distance) matrix as input (hclust, dbscan)
  - Only categorical data : K-modes

A measure for how good a clustering is

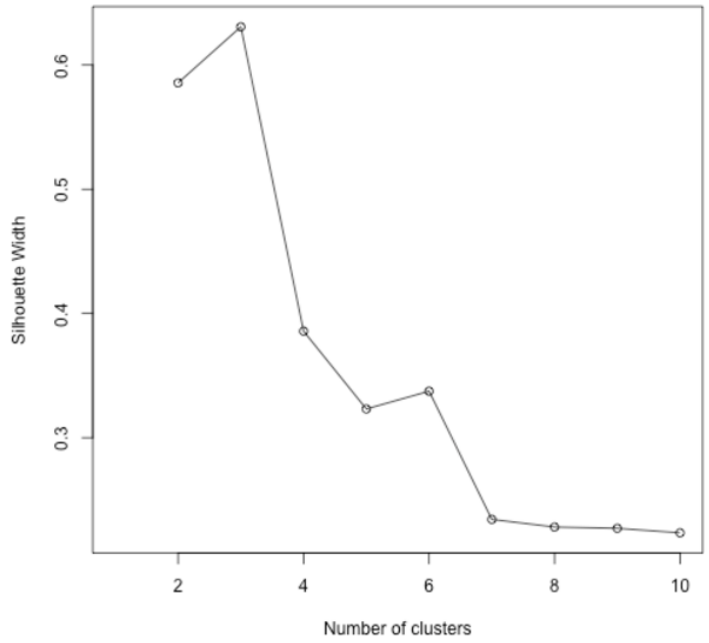
- Silhouette Value of an element
  - Each element is assigned to a cluster (partitioning)
  - A measure of how similar an element is to its own cluster compared to other clusters.
  - Can be calculated with any distance / dis-similarity / similarity metric
  - Normalized (Ranges from -1 to +1) ➔ Interpretable
  - High value (closer to +1) ➔ An element is similar to other elements of it's cluster
  - cohesion a(i) to separation b(i) ratio
    - a(i) : average dissimilarity of i with all other data within the same cluster
    - $b(i)_c$: average dissimilarity of i with all other data in cluster c
    - b(i) : average dissimilarity of i with all other data in the 'neighboring' cluster = min(b
    - s(i) → 1 ➔ a(i) << b(i) and s(i) → -1 ➔ a(i) >> b(i)

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

- Silhouette Value of a clustering
  - Average silhouette value
  - If most elements have a high value, clustering is "good"
  - If most elements have a low value (→ -1) , clustering is "not good"➔ Try fewer / more clusters

Example

- Continuous
  - Acceptance rate
  - Out of school tuition
  - Number of new students enrolled
- Categorical
  - Whether a college is public/private
  - Whether a college is “elite”



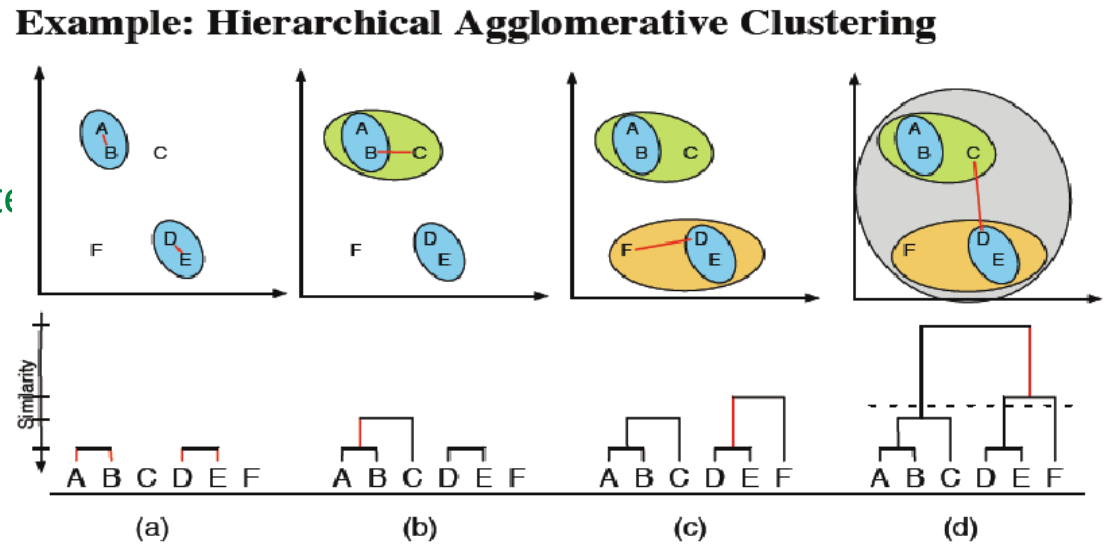

```
## Observations: 777
## Variables: 7
## $ name        (chr) "Abilene Christian University", "Ad...
## $ accept_rate (dbl) 0.7421687, 0.8801464, 0.7682073, 0....
## $ Outstate    (dbl) 7440, 12280, 11250, 12960, 7560, 13...
## $ Enroll      (dbl) 721, 512, 336, 137, 55, 158, 103, 4...
## $ Grad.Rate   (dbl) 60, 56, 54, 59, 15, 55, 63, 73, 80,...
## $ Private     (fctr) Yes, Yes, Yes, Yes, Yes, Yes, Yes,...
## $ isElite     (fctr) Not Elite, Not Elite, Not Elite, E...
```

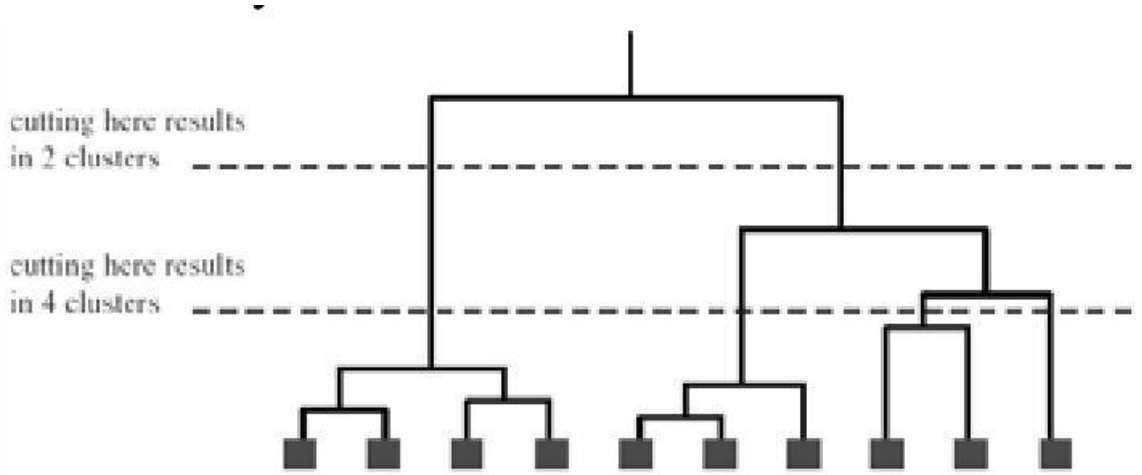
```
gower_dist <- daisy(college_clean[, -1],
                    metric = "gower",
                    type = list(logratio = 3))

sil_width <- c(NA)

for(i in 2:10){

  pam_fit <- pam(gower_dist,
                 diss = TRUE,
                 k = i)

  sil_width[i] <- pam_fit$silinfo$avg.width

}
```

l-types

Hierarchical

- Motivation
  - Instead or choosing k before clustering, choose it afte
  - Key Idea: Build a hierarchy of clusters.
- Agglomerative : Bottom-Up
  - Initialize
    - Each element is a cluster
  - Iterate
    - Calculate all pairwise distance between clusters
    - Merge the two nearest clusters
    - Repeat
  - Terminate
    - When all clusters merge into one.
    - Output dendogram : let user choose k
- Divisive : Top-Down
  - Reverse
- Distance between clusters?



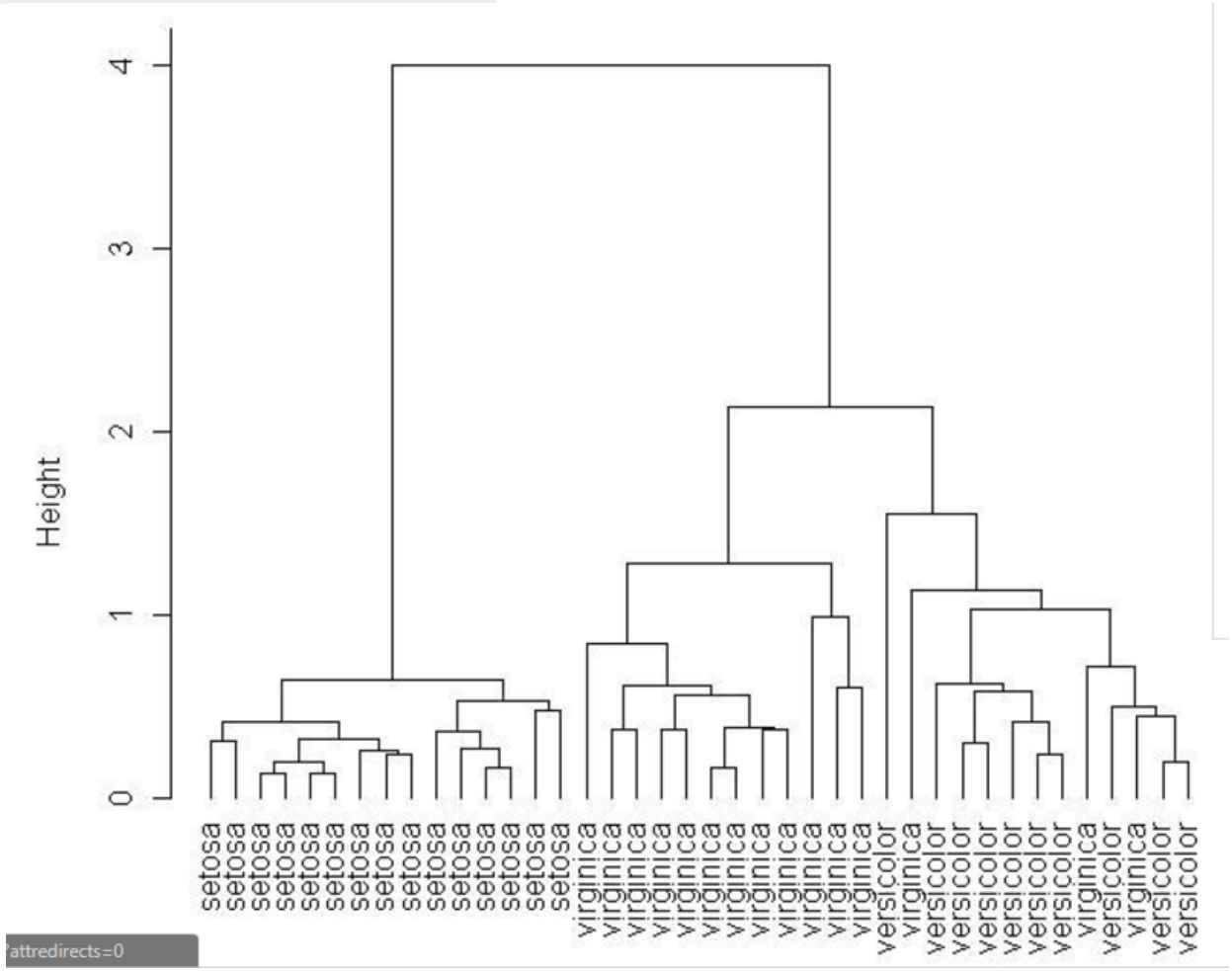


Distance between clusters?

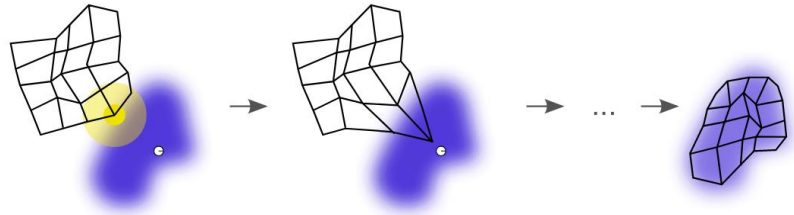
- Maximum or complete linkage clustering:
  - Max (all pairwise distances)
  - Tends to produce more compact clusters.
- Minimum or single linkage clustering:
  - Min (all pairwise distances)
  - Tends to produce long, “loose” clusters.
- Mean or average linkage clustering
  - Ave (all pairwise distances)
- Centroid linkage clustering:
  - Distance (Dissimilarity) between the centroid for cluster 1 and the centroid for cluster 2.
- Ward’s minimum variance method
  - Minimizes the total within-cluster variance.
  - At each step the pair of clusters with minimum between-cluster distance are merged.

Example

```
> hc <- hclust(dist(irisSample), method="ave")
> plot(hc, hang = -1, labels=iris$Species[idx])
```



Self Organizing Map



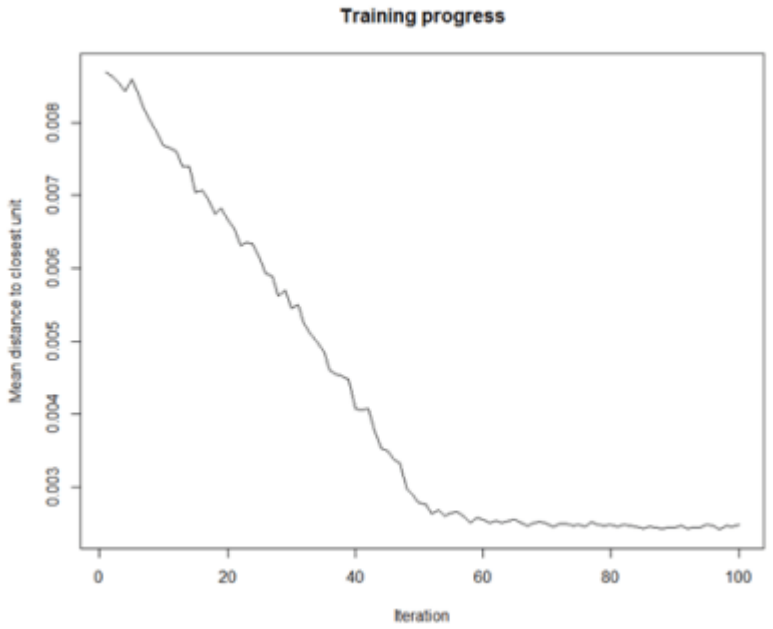
- Motivation
  - Cluster centroids must lie on a one or two dimensional manifold
  - Desirable for visualizing data (low dimensional space) s.t. similar items appear together
- Key Idea
  - Initialize a 2(low) dimensional manifold (rubber sheet) with m nodes (buttons sewn) in a grid
  - Bend the plane (by moving the centroids) so that the centroids are "close" to data
- Mechanics
  - Each node (button) is connected to each of the input elements with a random weight w_{iz}
  - Distance between an element and a node is used to increase the weight (Similarity ➔ Pull)
  - Nearby nodes are also “pulled” along with (“lower” weight update)

```
1: Initialize the centroids.
2: repeat
3:    Select the next object.
4:    Determine the closest centroid to the object.
5:    Update this centroid and the centroids that are close, i.e., in a specified neighborhood.
6: until The centroids don't change much or a threshold is exceeded.
7: Assign each object to its closest centroid and return the centroids and clusters.
```

Example

```
1: Initialize the centroids.
2: repeat
3:    Select the next object.
4:    Determine the closest centroid to the object.
5:    Update this centroid and the centroids that are close, i.e., in a specified neighborhood.
6: until The centroids don't change much or a threshold is exceeded.
7: Assign each object to its closest centroid and return the centroids and clusters.
```

- With each iteration
  - the distance from each node's weights to the samples represented by that node is reduced
- Convergence
  - Ideally, this distance should reach a minimum plateau.
  - If the curve is continually decreasing, more iterations are required.

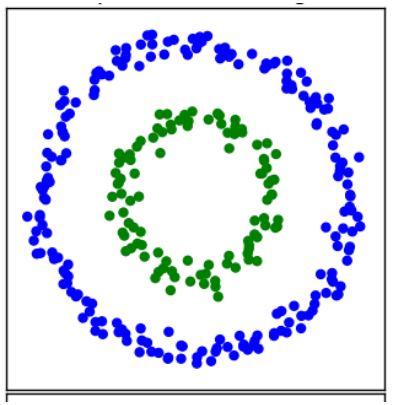


https://www.r-bloggers.com/self-organising-maps-for-customer-segmentation-using-r/

```
# Create the SOM Grid - you generally have to specify the size of the
# training grid prior to training the SOM. Hexagonal and Circular
# topologies are possible
som_grid <- somgrid(xdim = 20, ydim=20, topo="hexagonal")

# Finally, train the SOM, options for the number of iterations,
# the learning rates, and the neighbourhood are available
som_model <- som(data_train_matrix, grid=som_grid, rlen=100,
alpha=c(0.05,0.01), keep.data = TRUE, n.hood="circular" )
```

```
plot(som_model, type="changes")
```

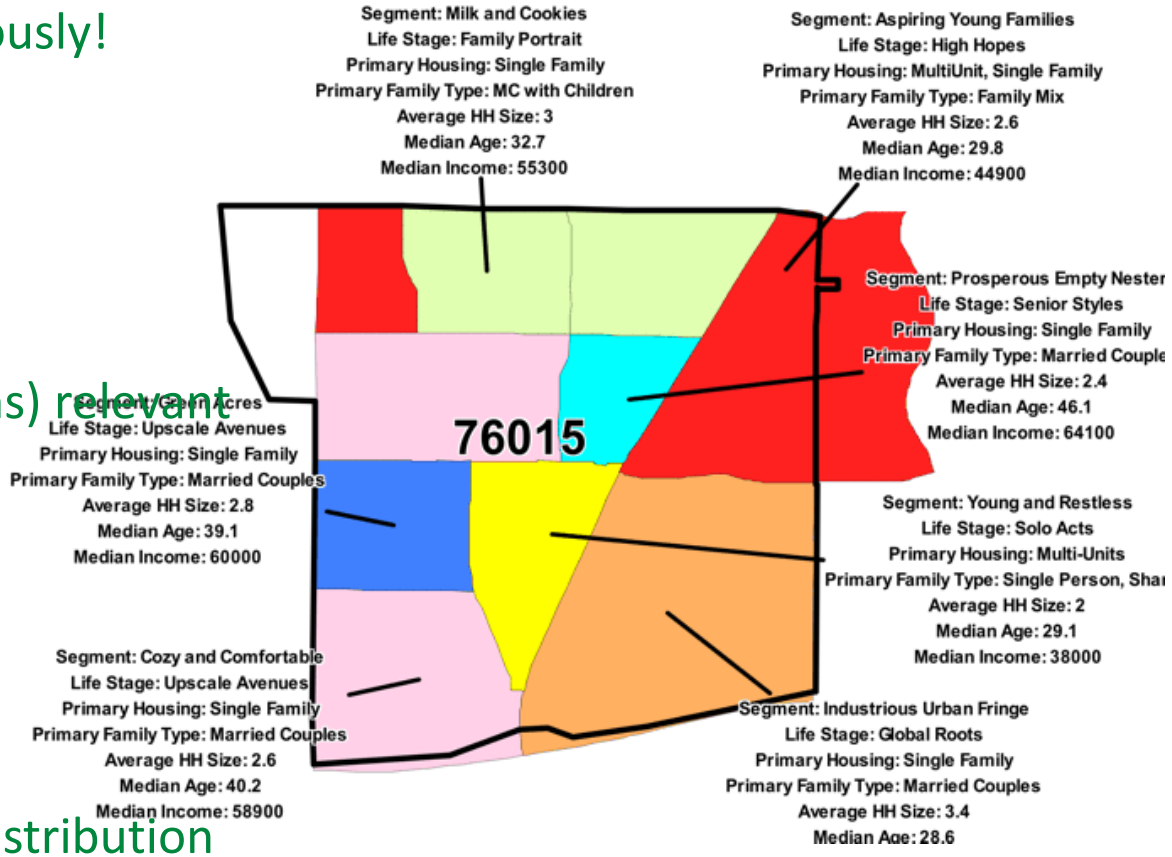
DBScan



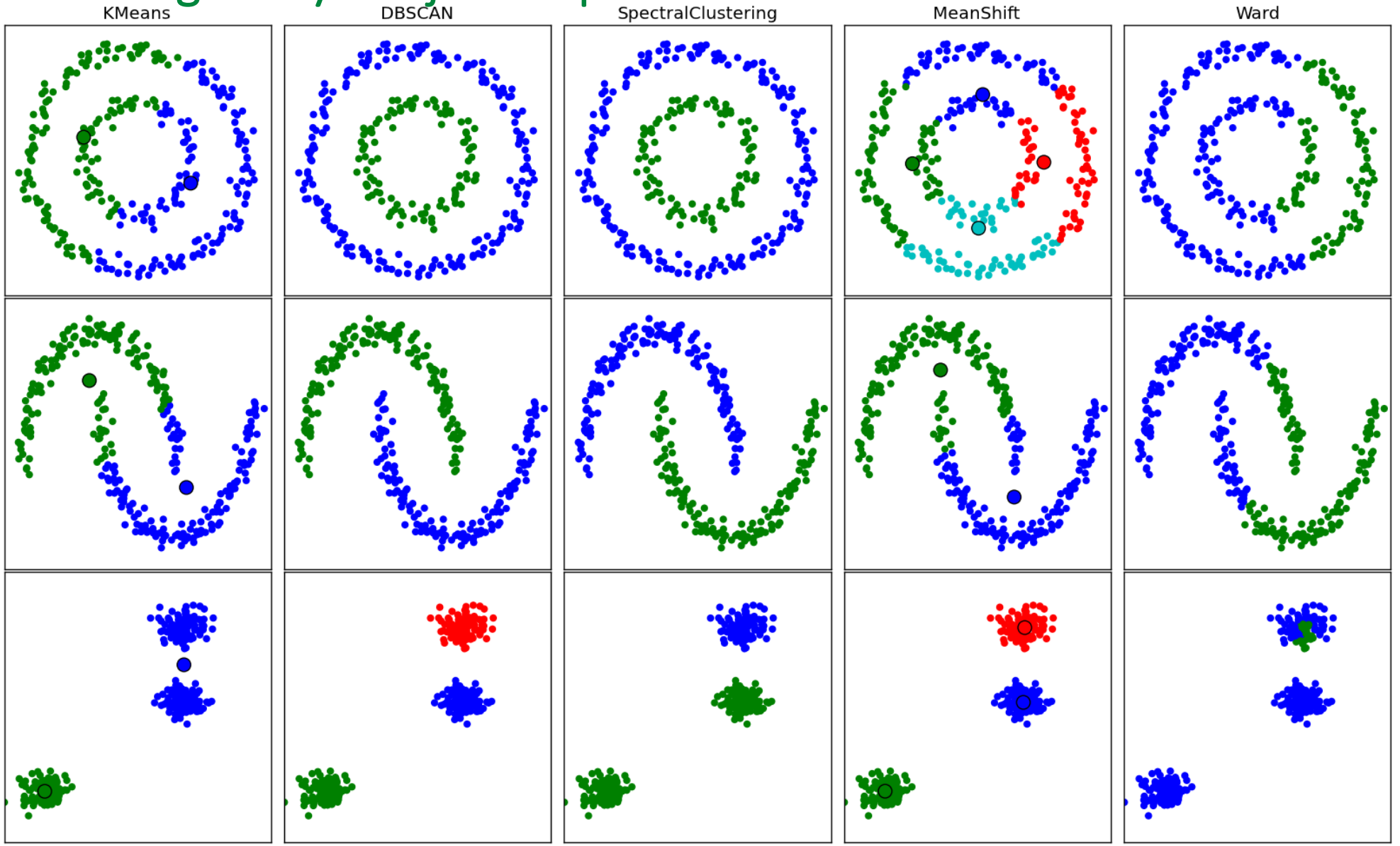
- Motivation
  - Some data does not lend itself into centroid-based clustering
- Key Idea
  - Use “density” as the defining characteristic
  - Partition points into dense regions separated by not-so-dense regions
  - What is dense?
    - Density at a point : # of points within a circle of radius Ɛ
    - Dense region: A circle of radius Ɛ that contains at-least m points
- Iterate
  i. Pick a data point, that has not been visited, randomly from X
  ii. If number of points within Ɛ > m, create a cluster; otherwise label it as noise
    i. For each point in this new cluster, repeat (ii)
  iii. Repeat till all points are exhausted
- Cluster
  - All points within the cluster are mutually density-connected (reachable by paths with each hop < Ɛ)

Clustering : Gotchas!

- Powerful tool
  - Will ALWAYS give you some clusters : Use cautiously!
  - What is a “good” clustering?
  - Unsupervised Learning
- Pre-processing
  - Choose relevant features (attributes, dimensions) relevant
  - Choose (dis-)similarity / distance metric
  - Normalize variables (or not)
  - These choices impact which clusters emerge
- Post-processing
  - Characterize each clusters in terms of feature distribution
  - Name your clusters



Clustering : Why not just stop here?



Q?

Praphul Chandra

Insofe