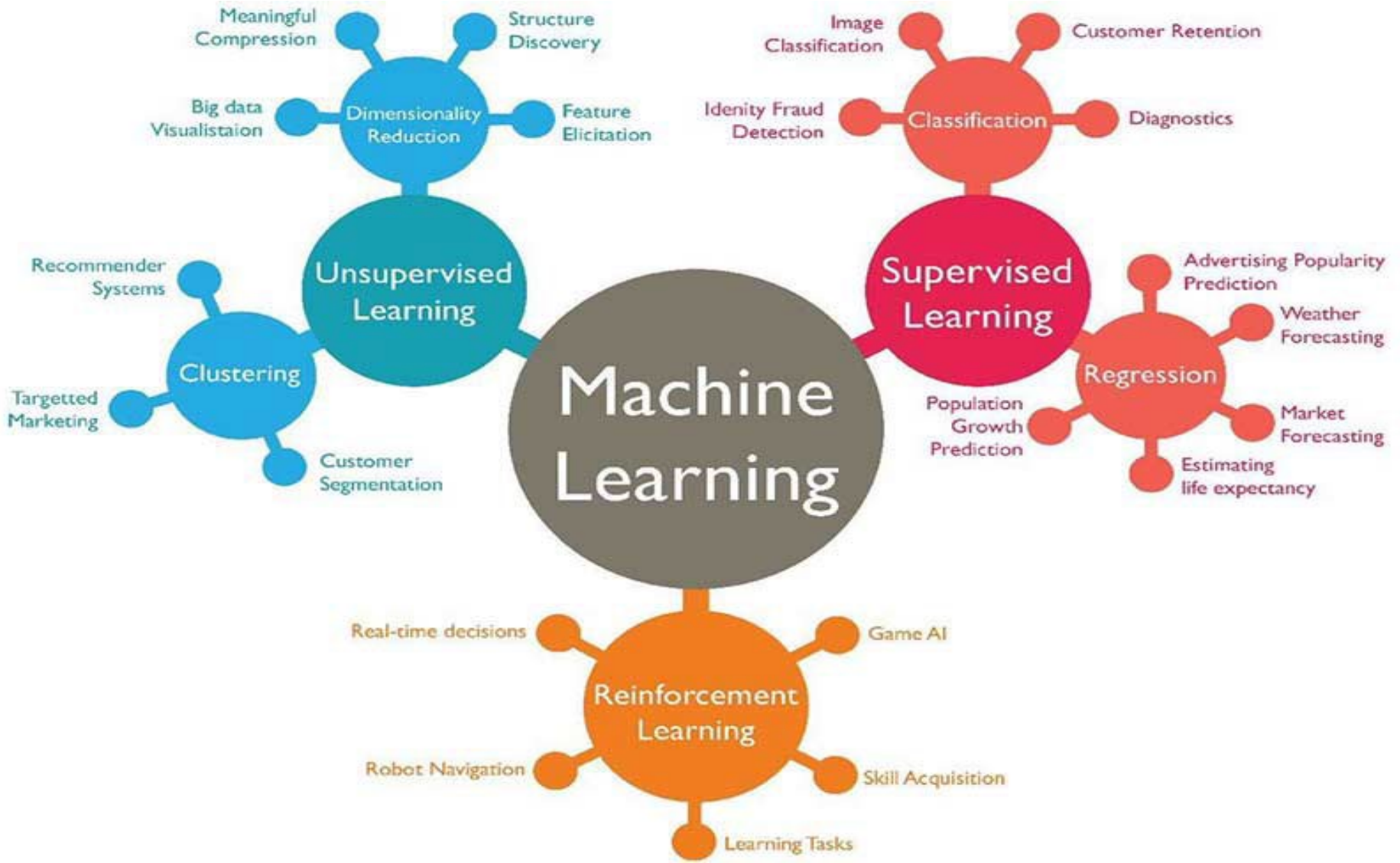
NLP for SE/AI techs

Intro to machine learning part 2

Agenda

- Basic Clustering algorithms
  - K-mean
  - HAC

Introduction to machine- learning

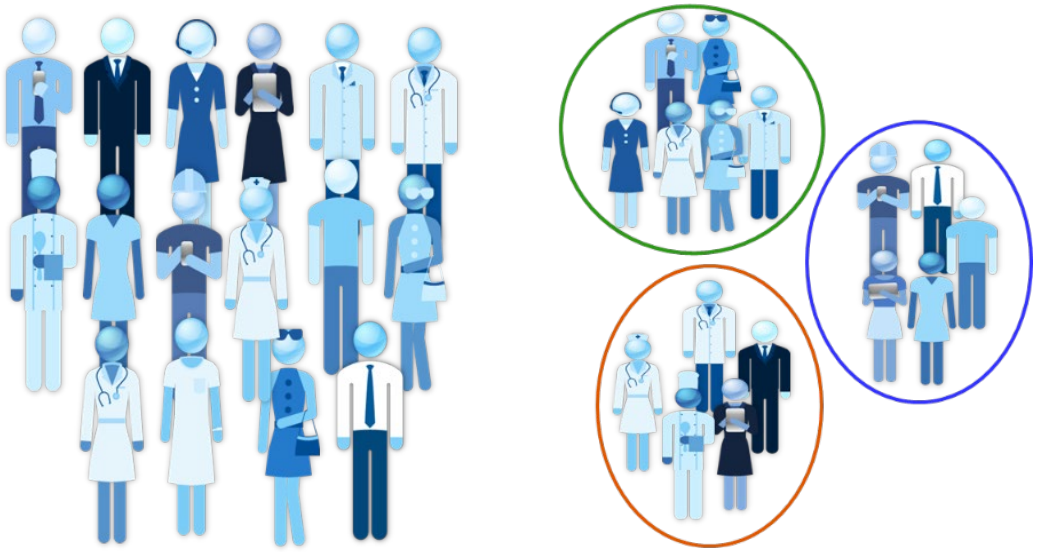


Classification vs Clustering

Criteria	Classification	Clustering
Prior Knowledge of classes	Yes	No
Use case	Classify new sample into known classes	Suggest groups based on patterns in data
Algorithms	Decision Trees, Bayesian classifiers	K-means, Expectation Maximization
Data Needs	Labeled samples from a set of classes	Unlabeled samples

Clustering

- **Belong to unsupervised learning**
- **Goal:** Grouping a set of objects that has similarity to each other than those in the other groups



Cluster Analysis Applications

- Broadly used in many applications
  - market research, pattern recognition, data analysis, and image processing.
- Help marketers **discover distinct groups** in their customer base, such that they can characterize their customer groups based on the purchasing patterns
- Used in **outlier detection** applications such as detection of credit card fraud
- As a data mining function, cluster analysis serves as a tool to **gain insight into the distribution** of data to observe characteristics of each cluster.

Clustering examples

- Image segmentation
- Goal: Break up the image into similar regions

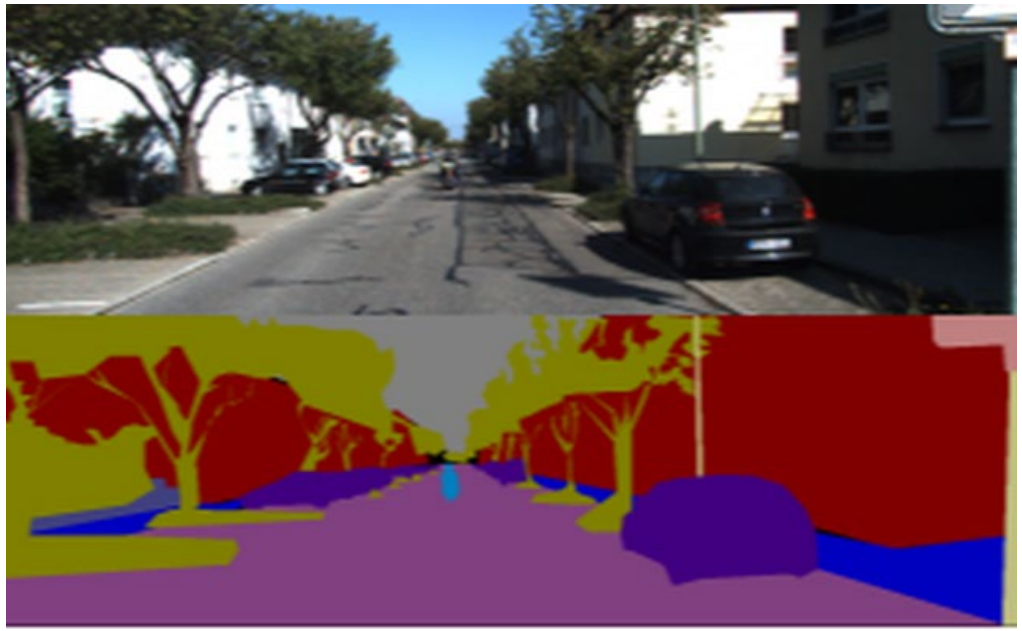


Image source: http://adas.cvc.uab.es/elektra/datasets/semantic-segmentation/

Clustering Methods

- Partitioning Method
  - K-means
- Hierarchical algorithms
  - Agglomerative (HAC) – Bottom up approach (begin with each element as a separate cluster and merge them into successively larger cluster)
  - Divisive – top-down approach (begin with the whole set and process split it into successively smaller clusters

K-means clustering

- The most well-know cluster algorithm
- Easy to understand and implement in code

K-mean applications

- Behavioral segmentation
  - Segment by purchase history
  - Segment by activities on application (e.g. website)
- Inventory categorization
  - Group inventory by sales activity
- Sorting sensor measurements
  - Group images
  - Separate audio
  - Detect activity types in motion sensors.
- NLP
  - Group sentences
- Time series
  - Group Similar series

Visualizing K-means Clustering

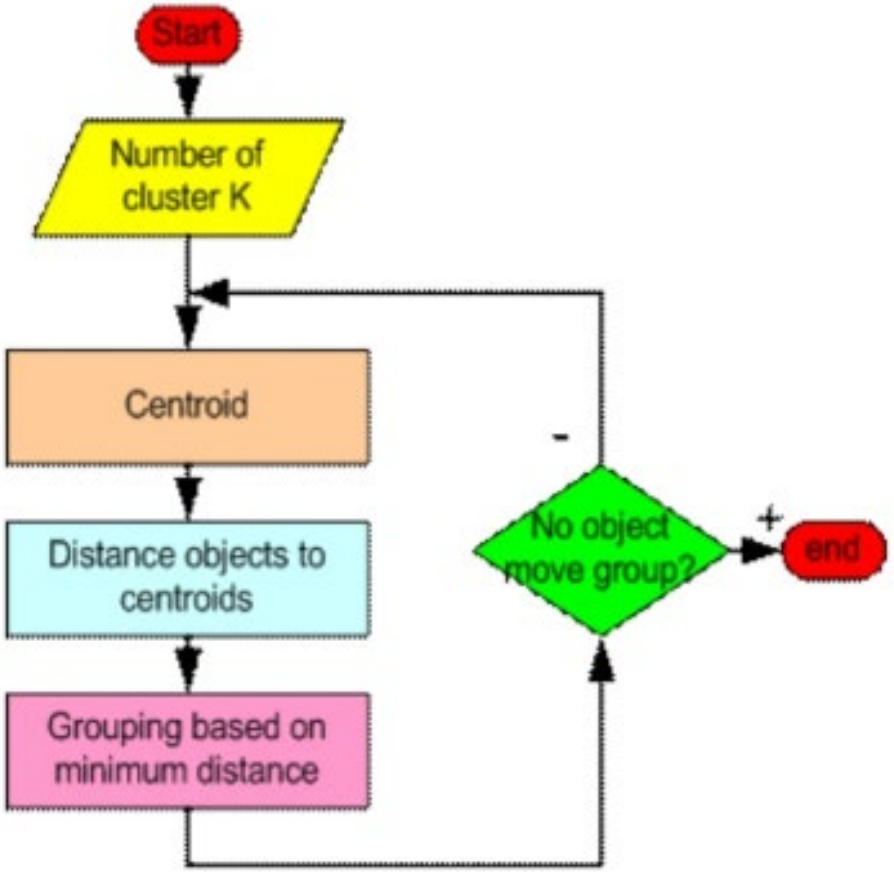
- https://www.naftaliharris.com/blog/visualizing-k-means-clustering/
- https://hckr.pl/k-means-visualization/

- Let take a look at the algo., so we can see through how things works.

Similarity measurement

- Euclidean distance*
- Manhattan distance
- Cosine Distance
- Squared Euclidean Distance

K-means in one picture



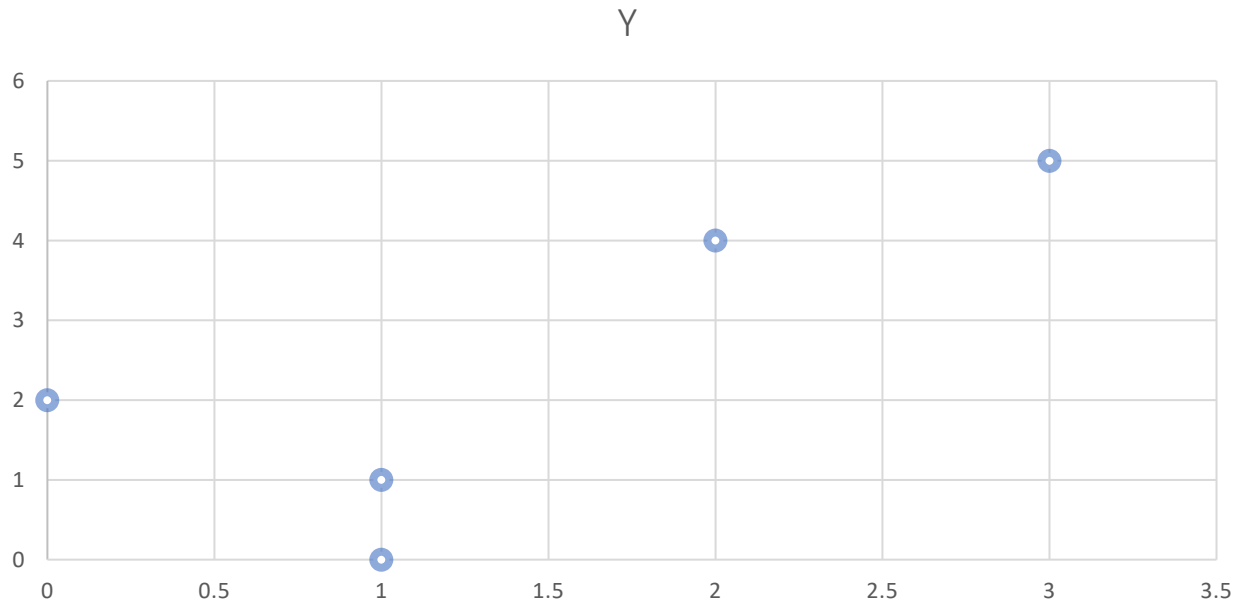
Iterate until cluster converge:

1. Determine the centroid coordinate
2. Determine the distance of each objects to the centroid
3. Group the object based on minimum distance (find the closest centroid)

K-Mean Clustering

Apply K-mean clustering for the following data sets for two clusters.

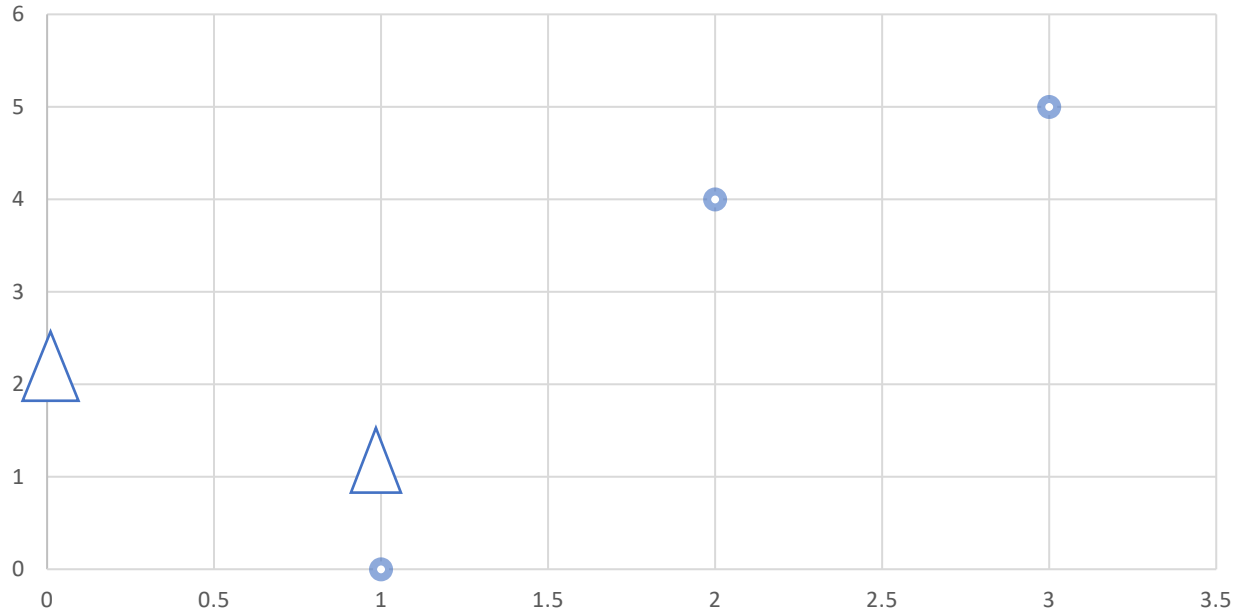
points	X	Y
A	1	1
B	1	0
C	0	2
D	2	4
E	3	5



K-Mean Clustering

Given k=2, we randomly pick two centroids (1,1) and (0,2).
Mean for cluster 1 is A, Mean for cluster 2 is C

points	X	Y
A	1	1
B	1	0
C	0	2
D	2	4
E	3	5



K-Mean Clustering

- $Eucledian\ Distance[(x, y), (a, b)] = \sqrt{(x - a)^2 + (y - b)^2}$

centriod	X	Y
C1	1	1
C2	0	2

Initial Centroids

K-Mean Clustering

- Use distance function
- C1 [(1, 1), (1, 1)] = $\sqrt{(1-1)^2+(1-1)^2}$ =0
- Distance from C2 [(1, 1), (0,2)] = $\sqrt{(0-1)^2+(2-1)^2}$
  - = $\sqrt{(-1)^2+(1)^2}$ = 1.4

K-Mean Clustering

Compute all the distances between **each of the cluster means** and **all the other points**

We assign the cluster number based on the closet centroid

points	C1	C2	Assignment to Cluster
A	0	1.4	1
B	1	2.2	1
C	1.4	0	2
D	3.2	2.8	2
E	4.5	4.2	2

*https://www.calculatorsoup.com/calculators/geometry-plane/distance-two-points.php

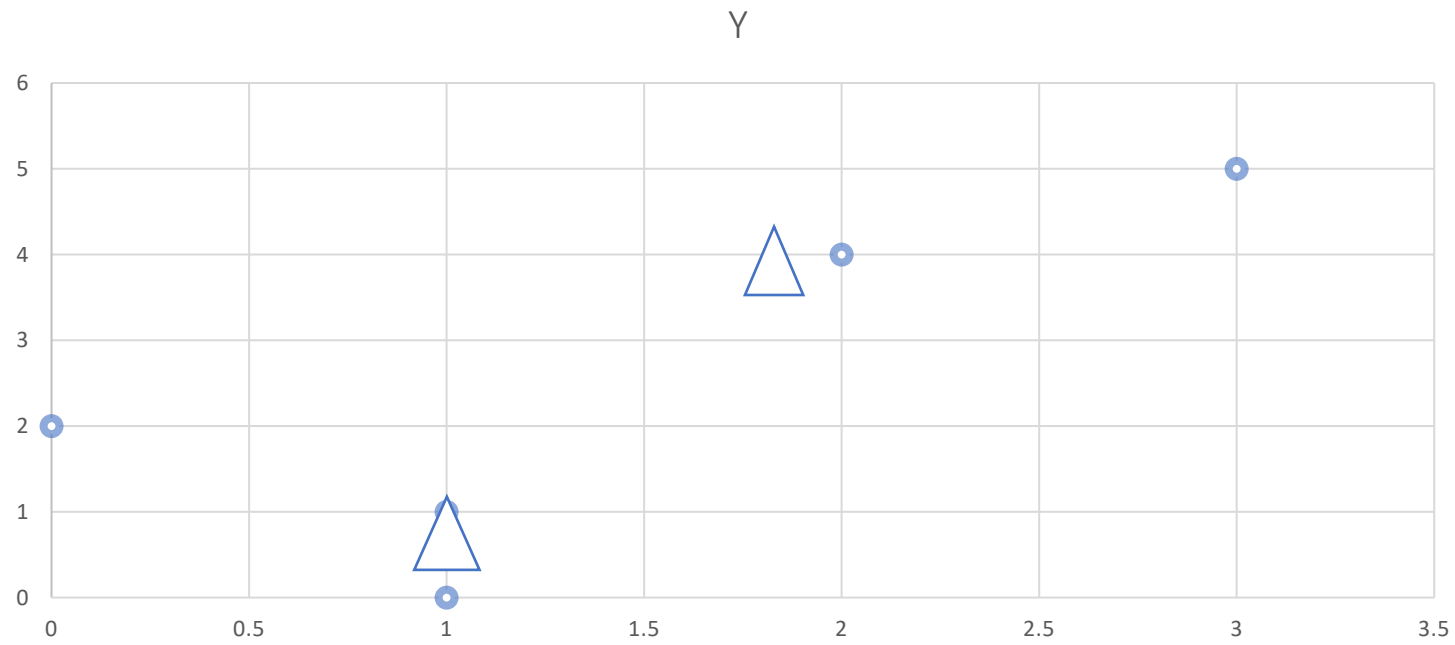
K-Mean Clustering

- Update centroid points by taking average
- New centroid C1 ((1+1)/2), ((1+0)/2) = (1, 0.5)
- New centroid C2 ((0+2+3)/3) , ((2+4+5)/3) = (1.7, 3.7)

points	X	Y
A	1	1
B	1	0
C	0	2
D	2	4
E	3	5

K-Mean Clustering

- Now, see the plots C1 = (1, 0.5), C2 = (1.7, 3.7)



K-Mean Clustering

- C1 = (1, 0.5) and C2 = (1.7, 3.7)
- Use the new centroid Compute all the distances between **each of the cluster means** and **all the other points (step 2 and 3)**

points	X	Y
A	1	1
B	1	0
C	0	2
D	2	4
E	3	5

First initial points

point	C1	C2
A	0.5	2.7
B	0.5	3.7
C	1.8	2.4
D	3.6	0.5
E	4.9	1.9

After the 1st iteration of update

K-Mean Clustering

Recalculate the cluster means, C1 = (0.7, 1), C2 = (2.5, 4.5)

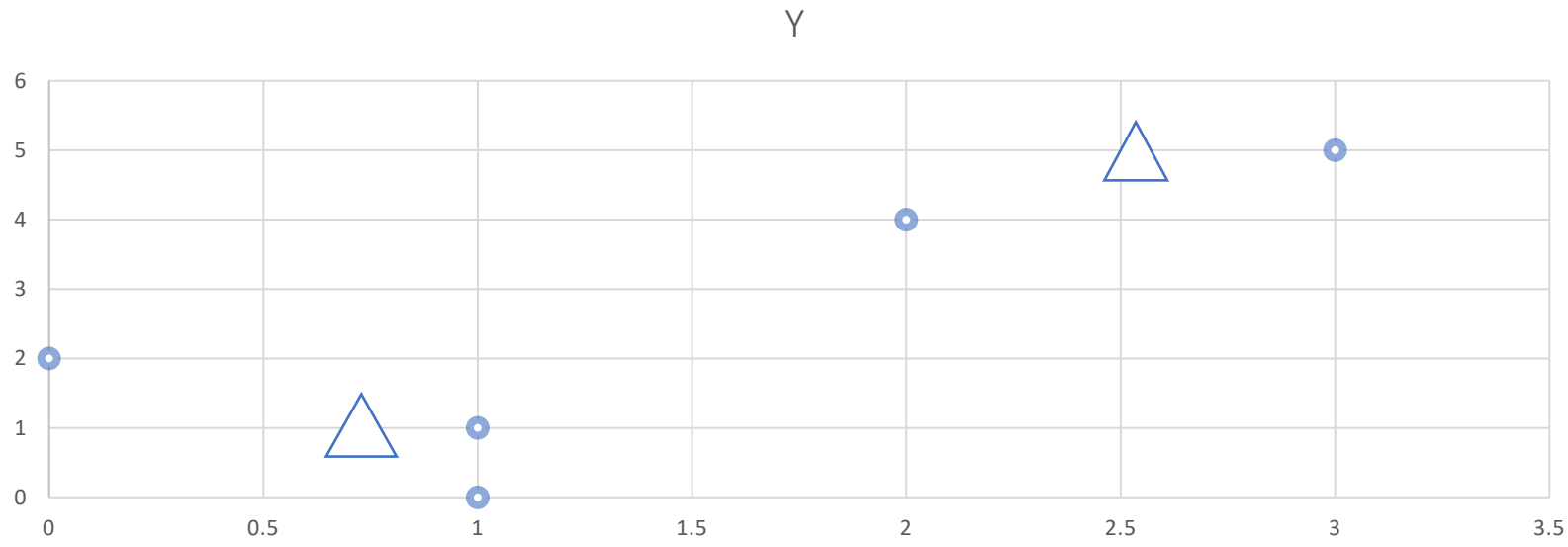
Points	C1	C2	Assign to Cluster
A	0.5	2.7	1
B	0.5	3.7	1
C	1.8	2.4	1
D	3.6	0.5	2
E	4.9	1.9	2

Points	X	Y
A	1	1
B	1	0
C	0	2
D	2	4
E	3	5

K-Mean Clustering

Algorithm has converged, no more change to the centroids if we keep calculating

C1 = (0.7, 1), C2 = (2.5, 4.5)



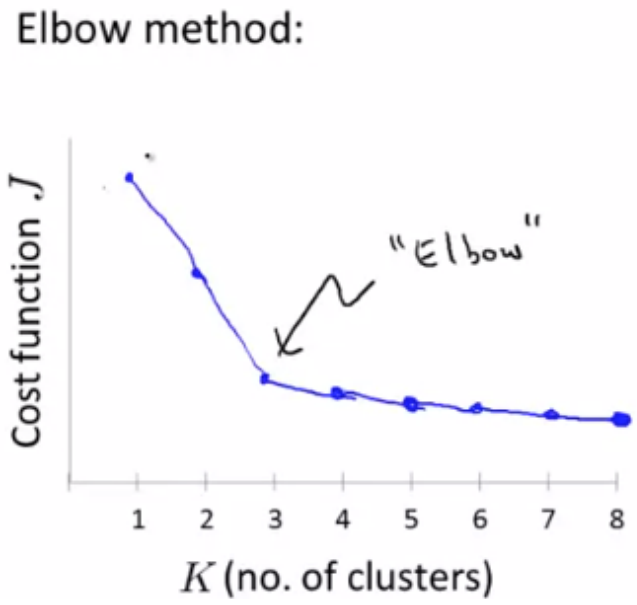
How to select an efficient set of initial centroids?

- Its NP hard problem, we need heuristic function.
- you can follow K-mean++ from the link here:
- http://ilpubs.stanford.edu:8090/778/1/2006-13.pdf
- http://www.csc.kth.se/utbildning/kth/kurser/DD143X/dkand13/Group4Per/report/40-eliasson-rosen.pdf

- For now, we select the initial centroid randomly (deal?).

How to choose the right value of k

- Manually choose
- Elbow method
  - a method of interpretation and validation of consistency within cluster analysis designed to help finding the appropriate number of clusters in a dataset.



How do we choose the number of cluster k?

- Try different k values, (Brute force)
  - best = kmean(points)
    - For k in range (n)
      - C = kmean(points)
      - if(dissimilarity (C) < dissimilarity(best)
        - best = C
    - Return best
- **dissimilarity(C)** is the sum of all the variabilities of k clusters
- **variability** is the sum of all Euclidean distances between the centroid and each example in the cluster.

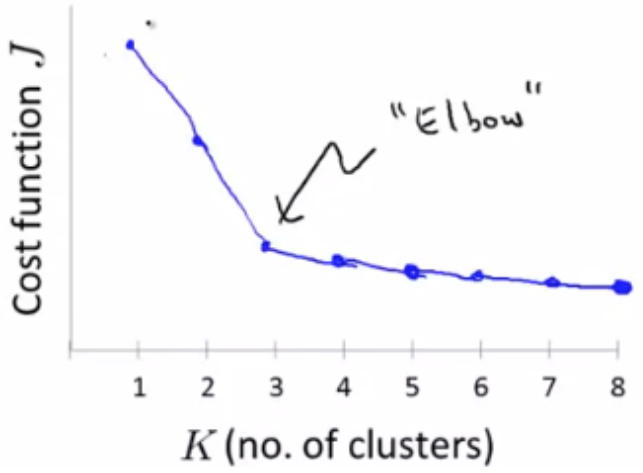
How do we choose the number of cluster k?

```python
inertia_list = []
for num_clusters in range(1, 11):
    kmeans_model = KMeans(n_clusters=num_clusters, init="k-means++")
    kmeans_model.fit(X)
    inertia_list.append(kmeans_model.inertia_
```

- With in cluster of sum of square = Inertia

$$\sum_{i=1}^{N}(x_i - C_k)^2$$

Elbow method:



Evaluation of clustering algorithm

- By Visualization
- (Required knowledge of the ground truth classes
  - Accuracy rate:

  measure the accuracy of the new labelling by comparing it with the original labelling (original labelling is the ground truth)
  - Adjusted Rand index – measure the similarity of two assignments ignore permutations.
  - Mutual information based scores

K-mean in Python

- https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html

Hierarchical Agglomerative Clustering

- Ensure nearby points merge into the same cluster
- Bottom-up approach
- Dendrogram – a tree data structure to represent hierarchical methods

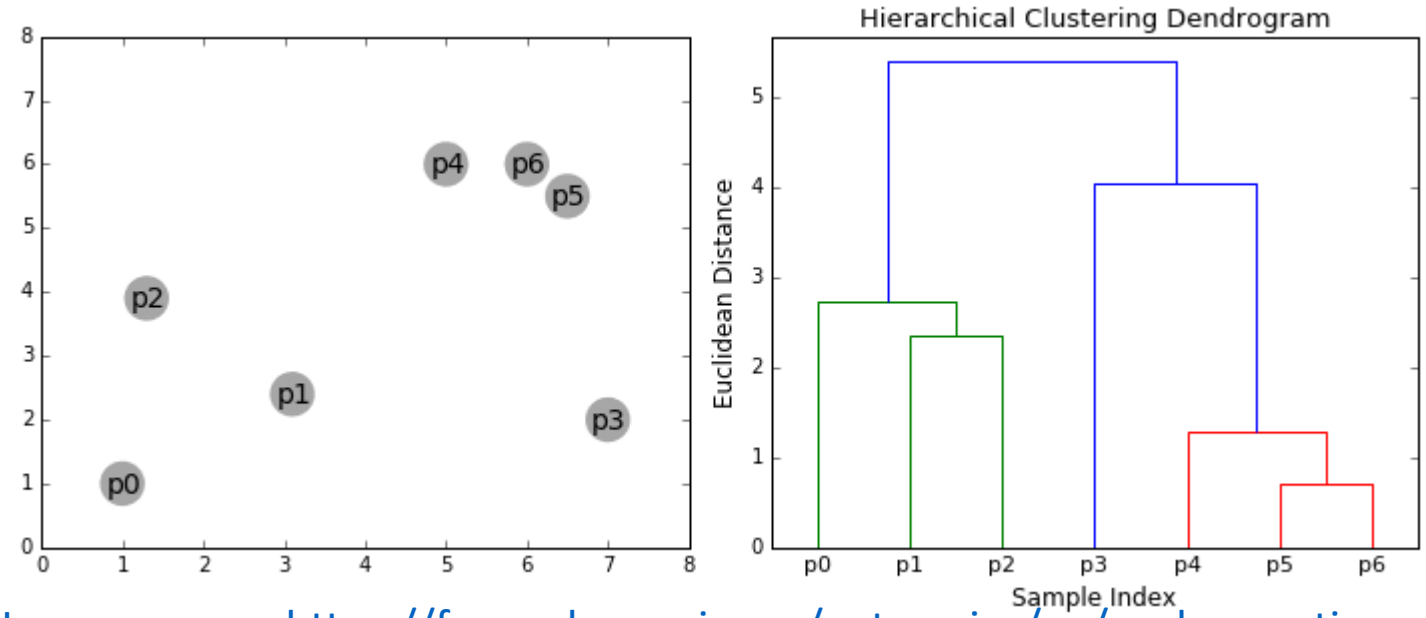
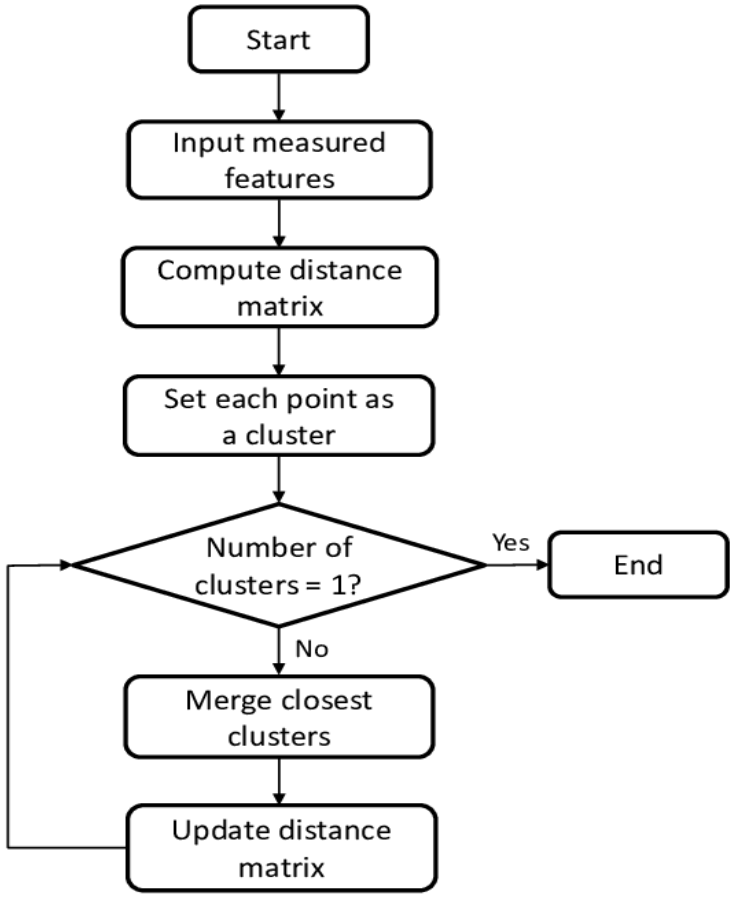


Image source: https://forum.huawei.com/enterprise/en/agglomerative-hierarchical-clustering/thread/609856-893

HAC in one picture



1. Compute the proximity matrix
2. Let each data point be a cluster

    Repeat:

        Merge the two closest clusters

        Update the matrix

    Until only 1 cluster remains.

Data points

*P*1 (1.3,4.7)

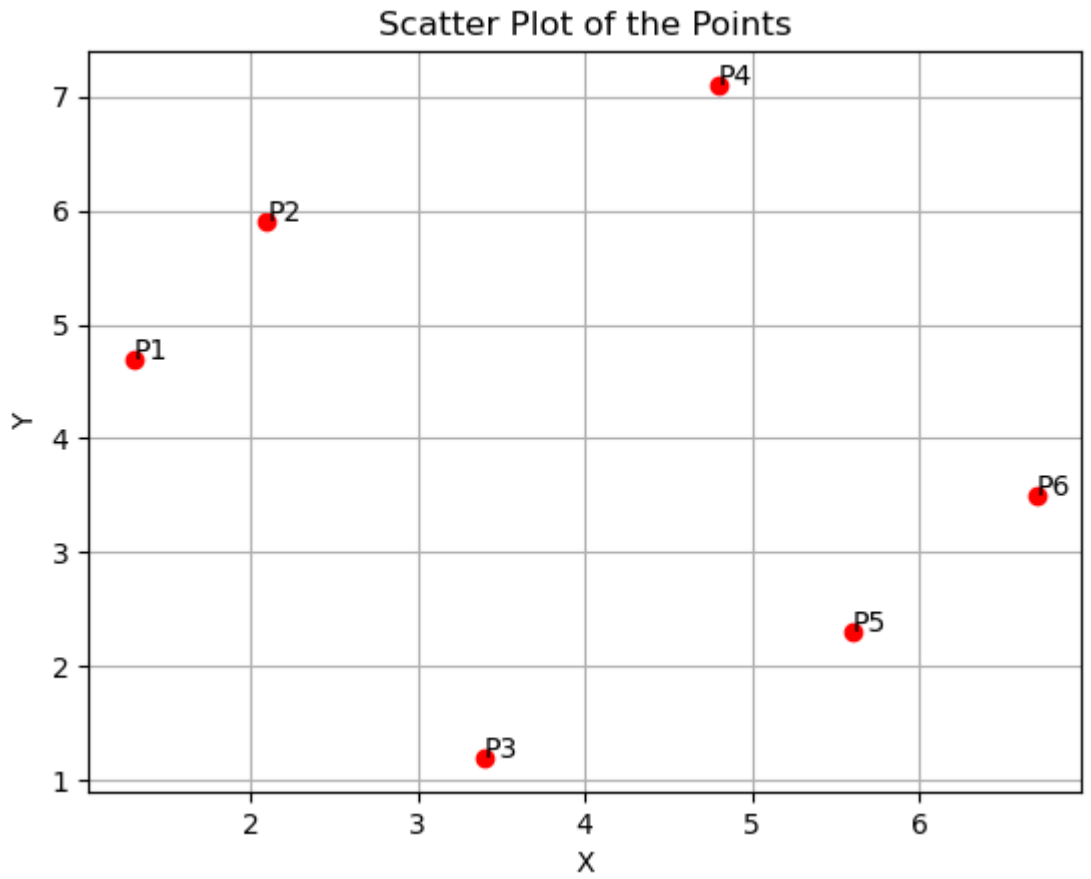
*P*2 (2.1,5.9)

*P*3 (3.4,1.2)

*P*4 (4.8,7.1)

*P*5 (5.6,2.3)

*P*6 (6.7,3.5)



Linkage criteria: Single Linkage

- The distance between two clusters is defined as the shortest distance between two points in each cluster.
- $D(C_i, D_j)=\min d(x,y)$

Distance matrix

–	$P1$	$P2$	$P3$	$P4$	$P5$	$P6$
$P1$	0	–	–	–	–	–
$P2$	d_{12}	0	–	–	–	–
$P3$	d_{13}	d_{23}	0	–	–	–
$P4$	d_{14}	d_{24}	d_{34}	0	–	–
$P5$	d_{15}	d_{25}	d_{35}	d_{45}	0	–
$P6$	d_{16}	d_{26}	d_{36}	d_{46}	d_{56}	0

We can see that the minum dist is P1 and P2
And we can form a cluster here C1={p1, p2}

	$P1$	$P2$	$P3$	$P4$	$P5$	$P6$
$P1$	0.00	–	–	–	–	–
$P2$	1.44	0.00	–	–	–	–
$P3$	3.81	5.04	0.00	–	–	–
$P4$	4.24	2.88	6.44	0.00	–	–
$P5$	4.92	4.70	2.24	5.10	0.00	–
$P6$	5.83	5.29	3.33	4.02	1.50	0.00

merge these two points (or existing clusters) that are closest to each other according to the distance metric we are using.

Single linkage

- The distance between {P1, P2} and P3 is min(d(P1, P3), d(P2, P3)) = min(3.81, 4.92) = 3.81
- The distance between {P1, P2} and P4 is min(d(P1, P4), d(P2, P4)) = min(3.68, 2.88) = 2.88
- The distance between {P1, P2} and P5 is min(d(P1, P5), d(P2, P5)) = min(5.01, 4.70) = 4.70
- The distance between {P1, P2} and P6 is min(d(P1, P6), d(P2, P6)) = min(5.72, 5.83) = 5.83

Distance matrix (cont.)

- d(C1, p3) = min (d(p1,p3), d(p2,p3))
- Using the single linkage method, the distance between C1 and p3 is given by the minimum of the distances between P3 and all points in C1 (p1, p2).
- And this continue in loop..

	$C1$	$P3$	$P4$	$P5$	$P6$
$C1$	0.00	–	–	–	–
$P3$	3.81	0.00	–	–	–
$P4$	2.88	6.44	0.00	–	–
$P5$	4.70	2.24	5.10	0.00	–
$P6$	5.29	3.33	4.02	1.50	0.00

	$C1$	$P3$	$P4$	$C2$
$C1$	0.00	–	–	–
$P3$	3.81	0.00	–	–
$P4$	2.88	6.44	0.00	–
$C2$	4.70	2.24	5.10	0.00

- C2={p5,p6}

Single linkage

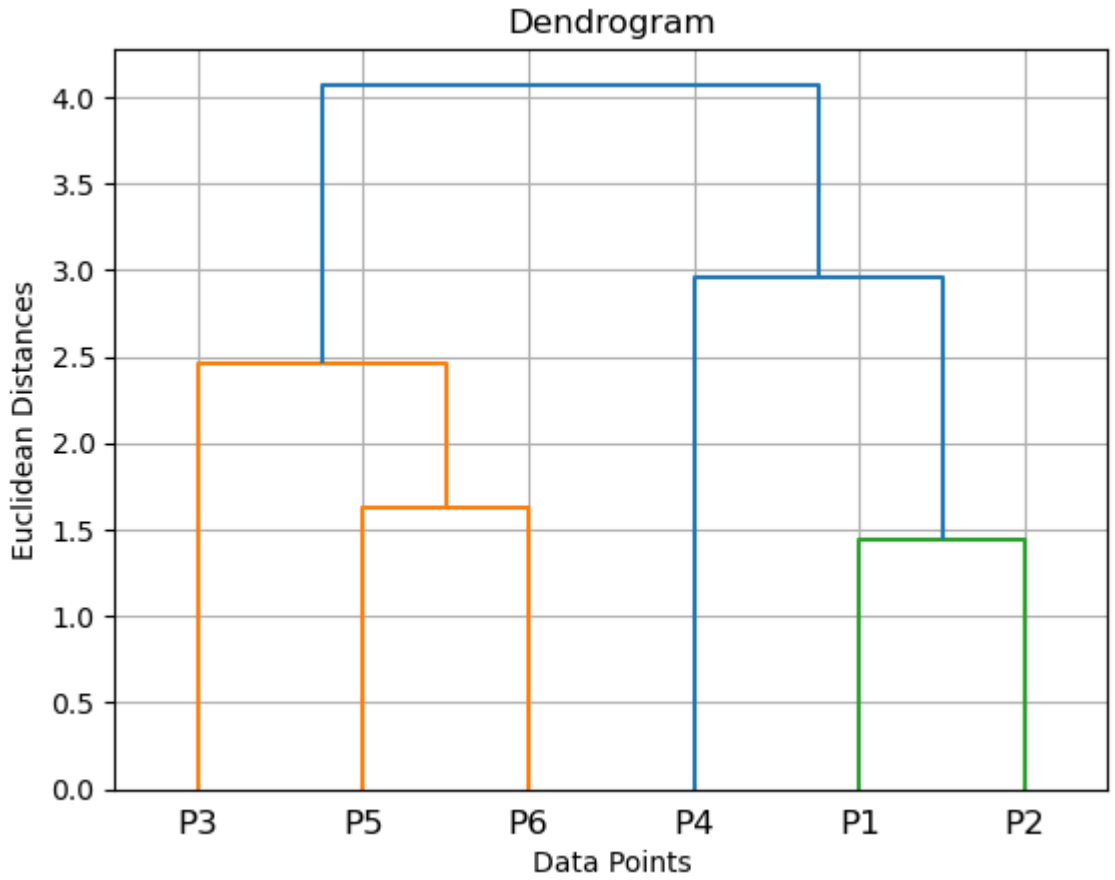
- So, let's find out the distances between C2{P5, P6} and other existing clusters including C1{P1, P2}:
- The distance between {P5, P6} and {P1, P2} is min(d(P5, P1), d(P5, P2), d(P6, P1), d(P6, P2)) = min(5.01, 4.86, 5.72, 5.08) = 4.86.
- The distance between {P5, P6} and P3 is min(d(P5, P3), d(P6, P3)) = min(2.36, 3.39) = 2.36.
- The distance between {P5, P6} and P4 is min(d(P5, P4), d(P6, P4)) = min(5.06, 4.06) = 4.06.

Distance matrix (cont.)

	{P1, P2}	P3	P4	{P5, P6}
{P1, P2}	0.00			
P3	3.65	0.00		
P4	2.77	6.43	0.00	
{P5, P6}	4.86	2.36	4.06	0.00

And this keep going till all the data points are merge in to a single clustered.

Dendrogram


Dendrogram
Euclidean Distances
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
P3
P5
P6
P4
P1
P2
Data Points

HAC in Python

- https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html

Class activity as *Workshop6*: Now, its your turn

1. Use K-mean this dataset (by hand -2 iterations)
2. Use Dendogram and elbow ---find the optimal number of cluster.

sample	A	B
P1	1.0	1.5
P2	1.5	2.0
P3	3.0	4.0
P4	5.0	7.0
P5	3.5	5.0
P6	4.5	5.0
P7	3.5	4.5

Last workshop Workshop 7: Customer segmentation with K-Mean (due 7th Oct)

- Download the dataset at the MSTeam, customer.csv
- Check for Missing value and outlier (1 point)
- Perform EDA (3 points)
  - Write summary talk about each feature and their distribution
- Scaling using standard scalar (1 point)
- Modelling with k-mean (4 points)
- Find and optimal number of cluster. (1 point)
- Plot elbow (extra credit 3 points)

References

- https://forum.huawei.com/enterprise/en/agglomerative-hierarchical-clustering/thread/609856-893
- https://www.naftaliharris.com/blog/visualizing-k-means-clustering/
- https://hckr.pl/k-means-visualization/