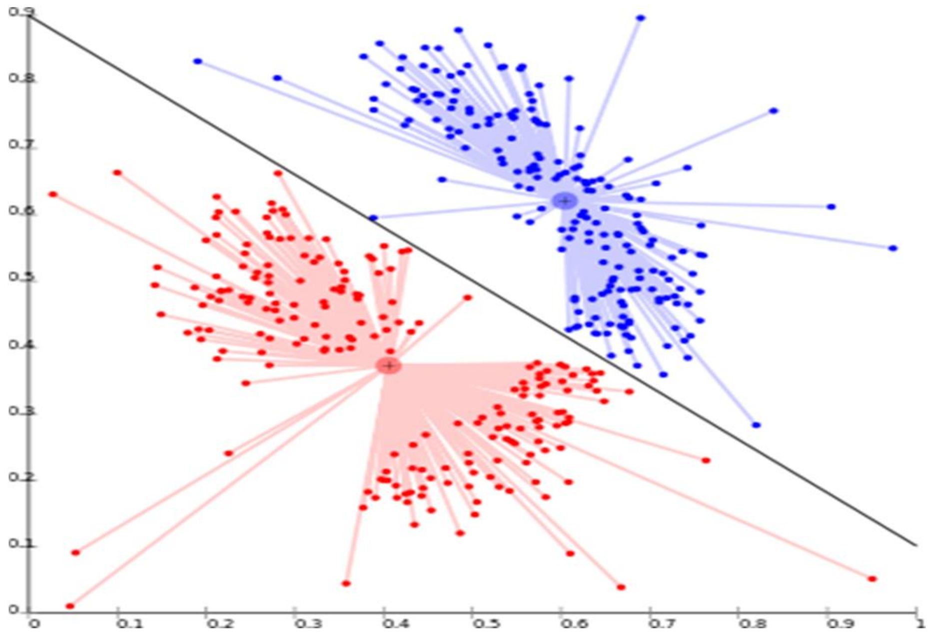
Data Science: Clustering

Won Kim

2022

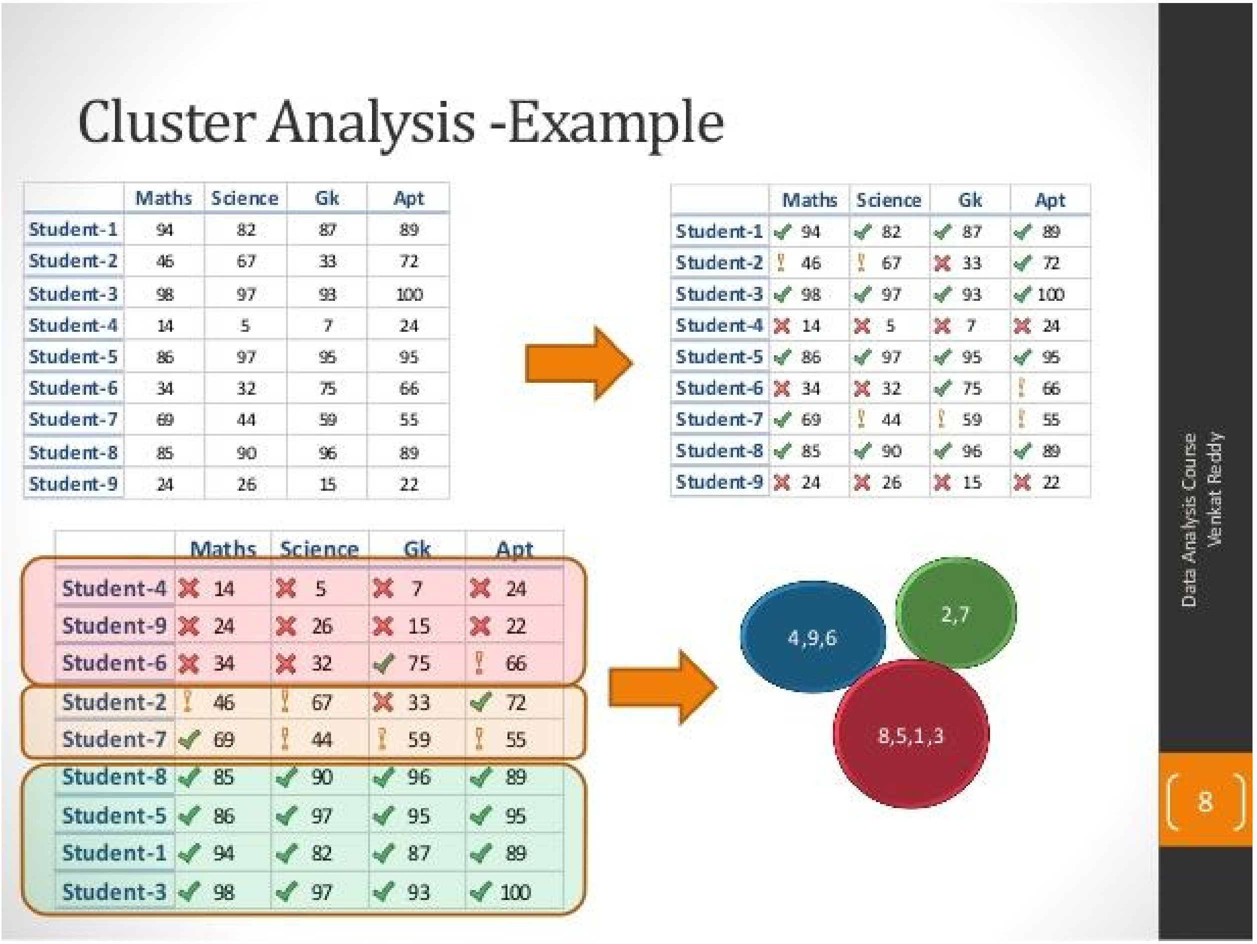
Clustering

- Divides a dataset into subgroups (clusters) based on similarity properties.
- Unlike classification, the training dataset has no target variable (or correct results).
- It is very difficult to know what is the result that meets the objective of data science project. (2 or 4 clusters? in the figure below)



Cluster Analysis Example

- The dataset has 4 features (scores) and 9 samples.
- Divides students into 3 clusters based on all 4 scores



Cluster Analysis Applications

- Biology & Bioinformatics
  - gene sequence analysis
- Medicine
  - medical imaging
- Business and Marketing
  - market research
  - shopping items grouping
  - recommendation system
- Web
  - search result grouping
  - social network analysis
- …

Cluster Models

- Centroid model (vector-center-based cluster)
  - k-means algorithm
- Connectivity/Hierarchical model (distance-based cluster)
  - agglomerative
  - divisive
- Distribution model (statistical distribution-based cluster)
  - Expectation-Maximization algorithm
- Density model (density-based cluster)
  - DBSCAN, OPTICS
- Graph model
  - clique

Roadmap: Clustering

- Overview
- Similarity
- k-Means Clustering
- Hierarchical Agglomerative Clustering

Acknowledgments

- http://www.cse.ust.hk/~qyang/337/slides/dist.ppt

Similarity and Dissimilarity

- Similarity
  - numerical measure of how alike two data objects are
  - is higher when objects are more alike.
  - often falls in the range [0,1]
- Dissimilarity
  - numerical measure of how different two data objects are
  - lower when objects are more alike
  - minimum dissimilarity is often 0
  - upper limit varies
- Proximity refers to similarity or dissimilarity

Distance Measures

- Euclidean Distance – most commonly used
- Manhattan Distance
- Minkowski Distance
- Mahalanobis Distance

Euclidean Distance

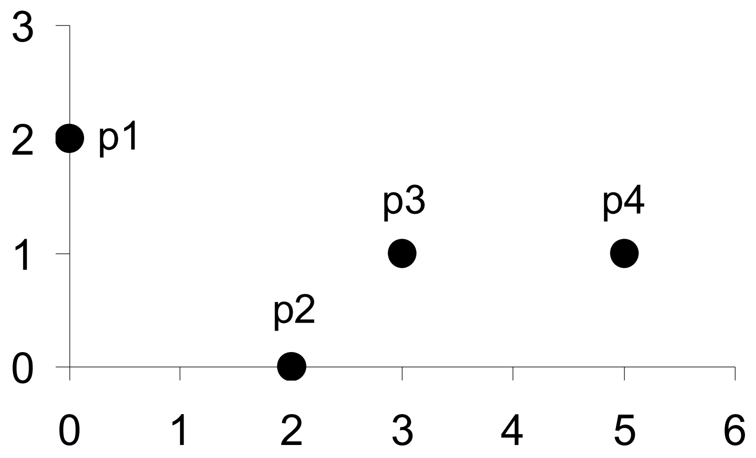
- Euclidean Distance formula
  (by Euclid of Alexandria, B.C. 300,
  father of geometry)

$$dist = \sqrt{\sum_{k=1}^{n} (p_k - q_k)^2}$$

Where n is the number of dimensions (attributes) and p_k and q_k are, respectively, the k[th] attributes (components) of data objects p and q.

- Standardization is necessary, if scales differ.

Example



point	x	y
p1	0	2
p2	2	0
p3	3	1
p4	5	1

	p1	p2	p3	p4
p1	0	2.828	3.162	5.099
p2	2.828	0	1.414	3.162
p3	3.162	1.414	0	2
p4	5.099	3.162	2	0

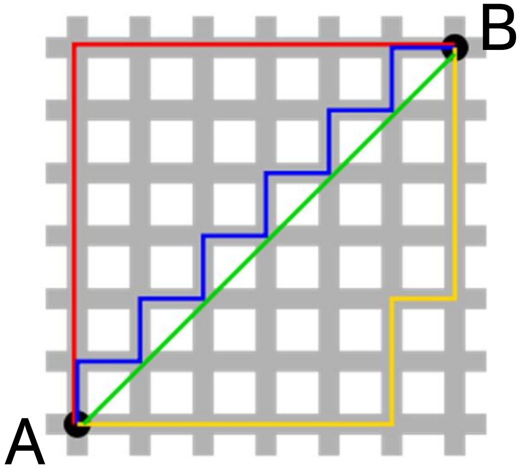
Distance Matrix

Manhattan Distance

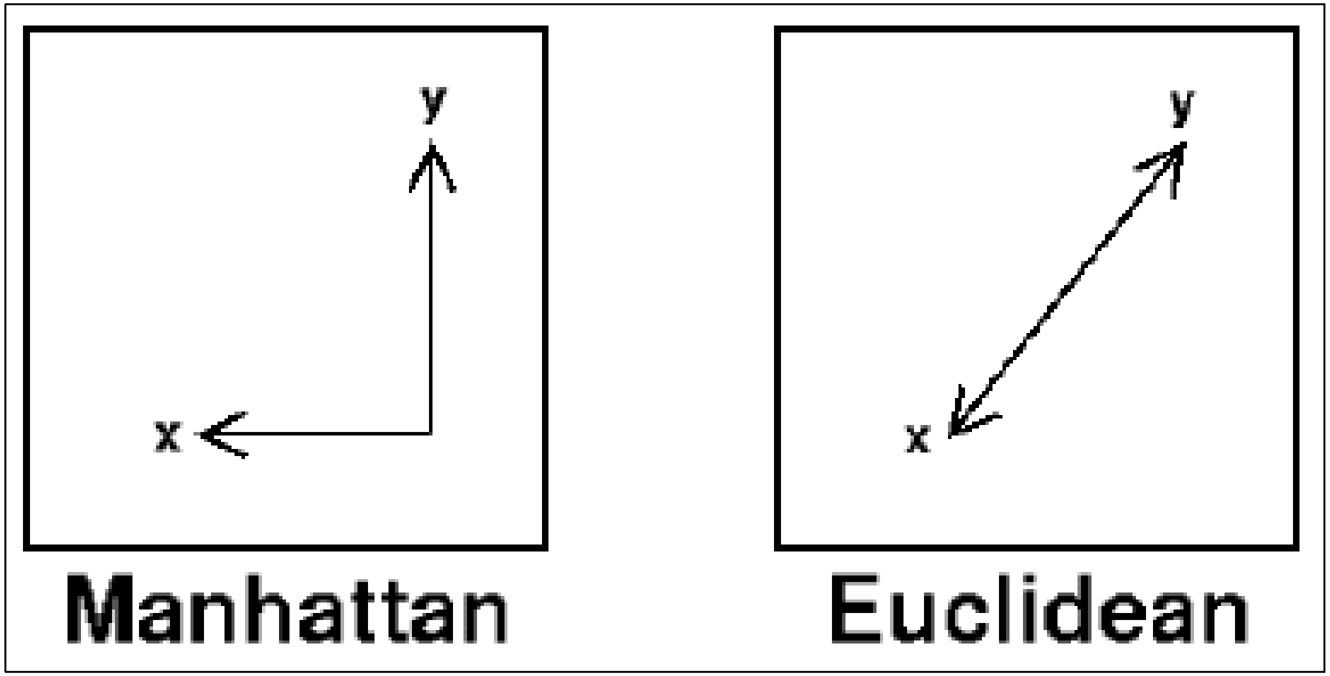
- Named after the shape of the streets in the borough of Manhattan in New York City

$$Manhattan\ Distance = \sum_{i=1}^{n} |p_i - q_i|$$

- Advantage over Euclidean distance
  - No need to compute multiplication and square root
- Multiple shortest paths (below: red, blue, yellow lines)



Symbols for Euclidean and Manhattan Distance



Minkowski Distance

- By German mathematician Hermann Minkowski
- Measure of distance between two points in the normed vector space (N dimensional real space)
- Generalization of the Euclidean distance and the Manhattan distance.

$$\left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{\frac{1}{p}}$$

- If parameter p is 1, this is Manhattan distance
- If parameter p is 2, this is Euclidean distance

Mahalanobis Distance

- By P. C. Mahalanobis in 1936
- Measure of the distance between a point P and a distribution D,
- It is a multi-dimensional generalization of the idea of measuring how many standard deviations away P is from the mean of D.
- This distance is zero for P at the mean of D and grows as P moves away from the mean along each principal component axis.

Cosine Similarity

- Widely used measure of similarity between text documents.
- Suppose we have two text documents, Hamlet and Macbeth.
- How can we determine how similar the two ducuments are?
  - We collect all unique words that appear in each document.
  - We then convert each word in each document into a number. Then each document becomes a vector.
  - We then compute the cosine similarity between the two vectors.
- Sounds incredible, but the result is pretty good.

Illustration: Computing Cosine Similarity

- If d_1 and d_2 are two document vectors, then

  $\cos(d_1, d_2) = (d_1 \bullet d_2) / ||d_1|| \, ||d_2||$,

  where $\bullet$ indicates vector dot product and

  $|| d ||$ is the length of vector d.
- Example:

  $d_1 = 3\ 2\ 0\ 5\ 0\ 0\ 0\ 2\ 0\ 0$

  $d_2 = 1\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 2$

  $d_1 \bullet d_2 = 3*1 + 2*0 + 0*0 + 5*0 + 0*0 + 0*0 + 0*0 + 2*1 + 0*0 + 0*2 = 5$

  $||d_1|| = (3*3+2*2+0*0+5*5+0*0+0*0+0*0+2*2+0*0+0*0)^{0.5} = (42)^{0.5} = 6.481$

  $||d_2|| = (1*1+0*0+0*0+0*0+0*0+0*0+0*0+1*1+0*0+2*2)^{0.5} = (6)^{0.5} = 2.245$

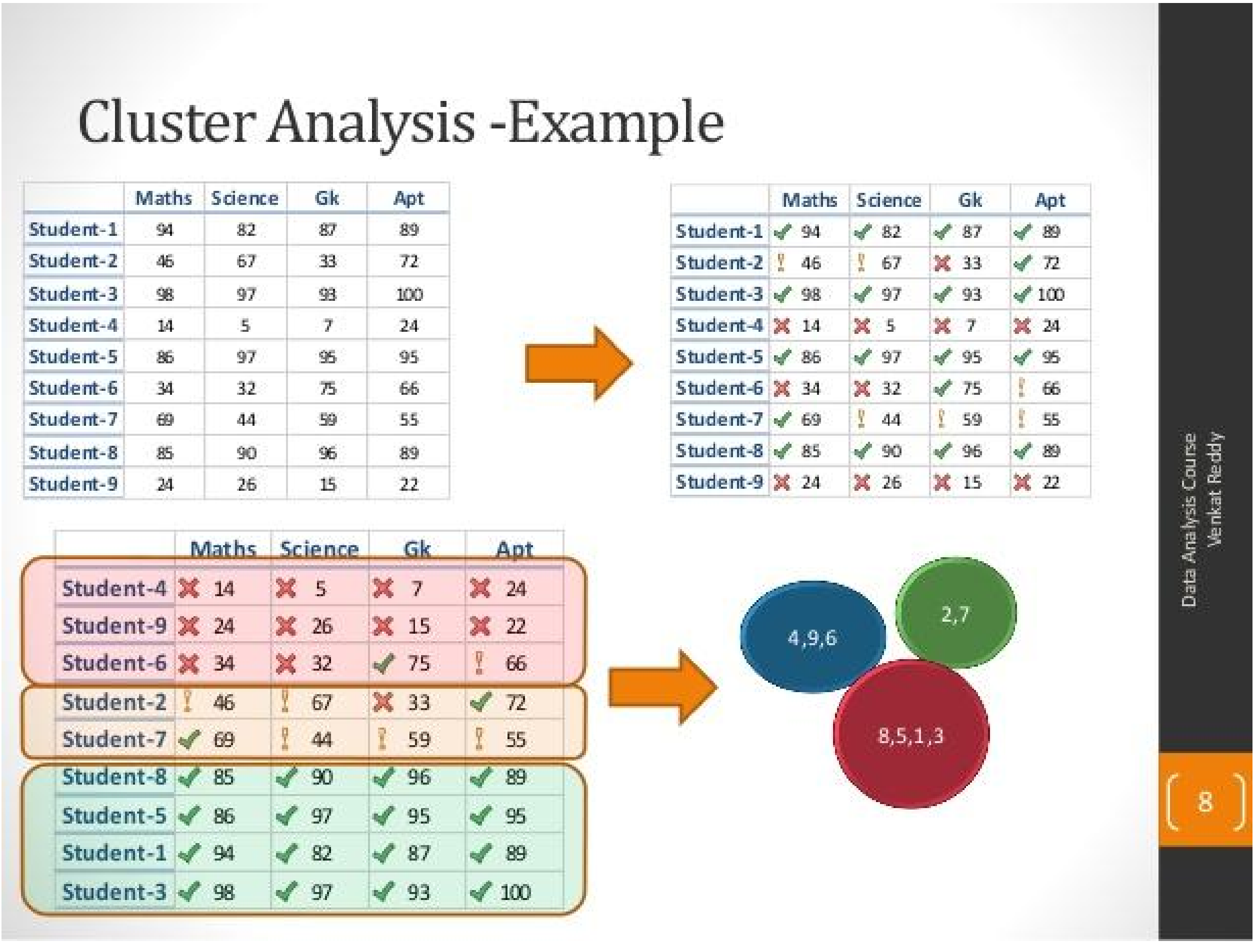
  $\cos(d_1, d_2) = .3150$, distance=1-cos(d1,d2)

Roadmap: Clustering

- Overview
- Similarity
- k-Means Clustering
- Hierarchical Agglomerative Clustering

Cluster Analysis Example

- The dataset has 4 features (scores) and 9 samples.
- Divides students into 3 clusters based on all 4 scores



Cluster Analysis -Example

	Maths	Science	Gk	Apt
Student-1	94	82	87	89
Student-2	46	67	33	72
Student-3	98	97	93	100
Student-4	14	5	7	24
Student-5	86	97	95	95
Student-6	34	32	75	66
Student-7	69	44	59	55
Student-8	85	90	96	89
Student-9	24	26	15	22

	Maths	Science	Gk	Apt
Student-1	✓ 94	✓ 82	✓ 87	✓ 89
Student-2	! 46	! 67	✗ 33	✓ 72
Student-3	✓ 98	✓ 97	✓ 93	✓ 100
Student-4	✗ 14	✗ 5	✗ 7	✗ 24
Student-5	✓ 86	✓ 97	✓ 95	✓ 95
Student-6	✗ 34	✗ 32	✓ 75	! 66
Student-7	✓ 69	! 44	! 59	! 55
Student-8	✓ 85	✓ 90	✓ 96	✓ 89
Student-9	✗ 24	✗ 26	✗ 15	✗ 22

	Maths	Science	Gk	Apt
Student-4	✗ 14	✗ 5	✗ 7	✗ 24
Student-9	✗ 24	✗ 26	✗ 15	✗ 22
Student-6	✗ 34	✗ 32	✓ 75	! 66
Student-2	! 46	! 67	✗ 33	✓ 72
Student-7	✓ 69	! 44	! 59	! 55
Student-8	✓ 85	✓ 90	✓ 96	✓ 89
Student-5	✓ 86	✓ 97	✓ 95	✓ 95
Student-1	✓ 94	✓ 82	✓ 87	✓ 89
Student-3	✓ 98	✓ 97	✓ 93	✓ 100

Clustering Is Very Difficult

- In general there are many possible groupings based on the objective and criteria you choose.
- For the current example, the following objectives may make sense.
  - Select students for scholarship award
  - Understand correlation between different tests/subjects
  - Understand score distribution for each test/subject
- Also, the following criteria may make sense.
  - (1) Use 1, 2, or 3 of the scores (not all 4)
  - (2) Create a different number of clusters (not 3); 2, 4, 5
  - (3) use various combinations of (1) and (2) above.

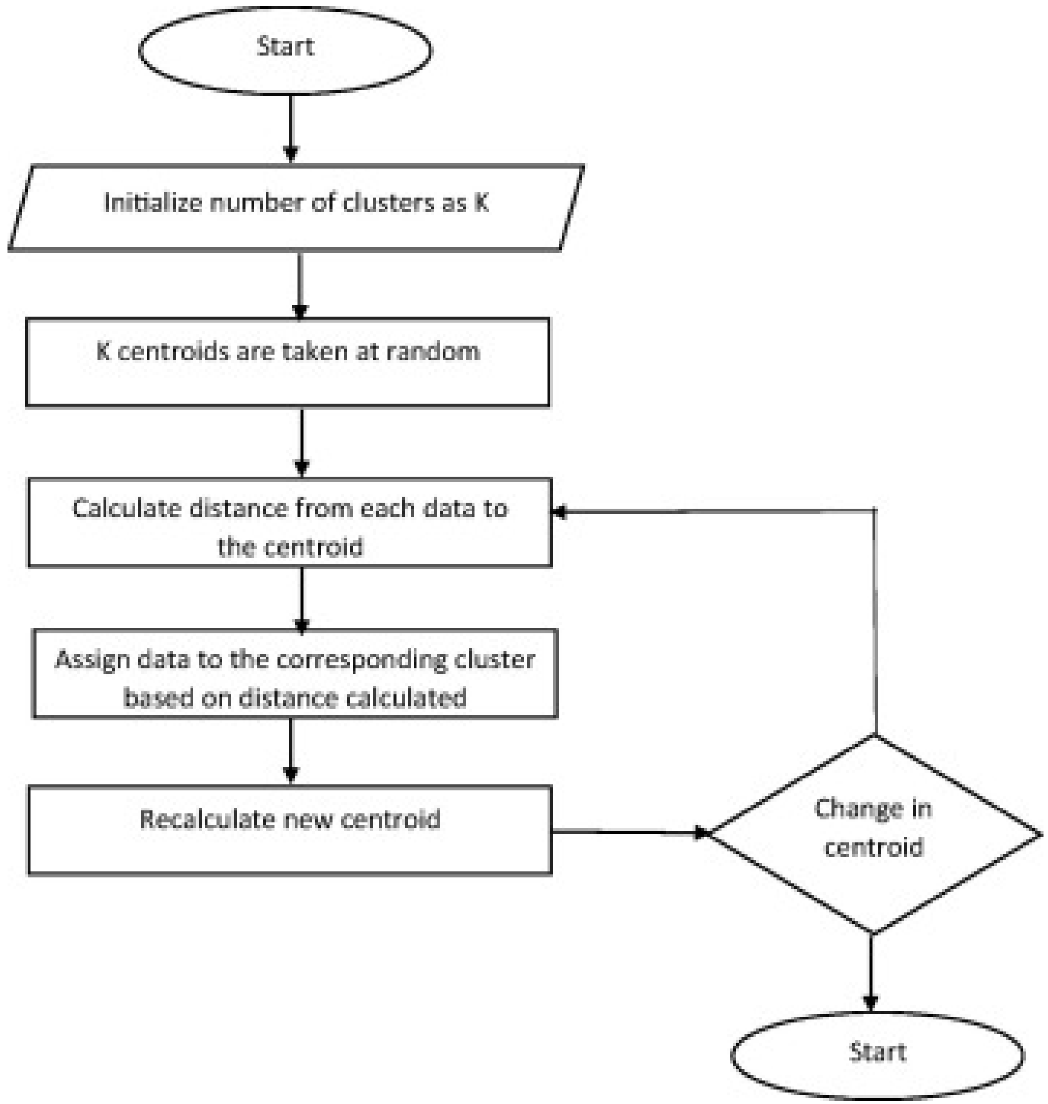
k-Means Clustering

- Simplest partitioning method for clustering analysis and widely used in data mining applications.
- k-means clustering is an algorithm to classify or group data based on attributes/features into k groups.
- The grouping is done by minimizing the distance between data and the corresponding cluster centroid.

Acknowledgments

- https://kjambi.kau.edu.sa/GetFile.aspx?id=187901&Lng=AR&fn=k-mean-clustering.ppt
- http://mnemstudio.org/clustering-k-means-example-1.htm

Flow Graph of the k-Means Clustering Algorithm



Computing Steps (1/2)

- Step 1
  - Begin with a decision on the value of k (the desired number of clusters).
- Step 2
  - Create an initial partition that classifies the data into k clusters, as follows:

    - Take the first k training data points as single-element clusters.
    - Assign each of the remaining (N-k) training samples to the cluster with the nearest centroid.
    - After each assignment, recompute the centroid of the gaining cluster.

Computing Steps (2/2)

- Step 3
  - Take each sample in sequence and compute its distance from the centroid of each of the clusters.
  - If a sample is not currently in the cluster with the closest centroid,

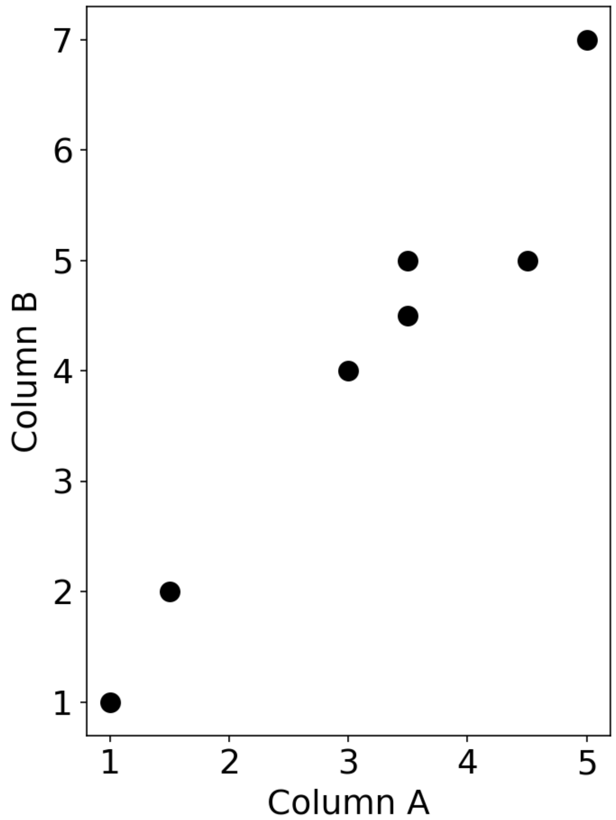
    - switch this sample to that cluster and

    - update the centroid of the cluster gaining the sample and the cluster losing the sample.
- Step 4
  - Repeat Step 3 until convergence, that is, until a pass through the training samples causes no new assignments.

Walkthrough Example 1

- Given a dataset consisting of 7 records, each with 2 features (A, B)
- Group the data into 2 clusters (i.e., k=2).

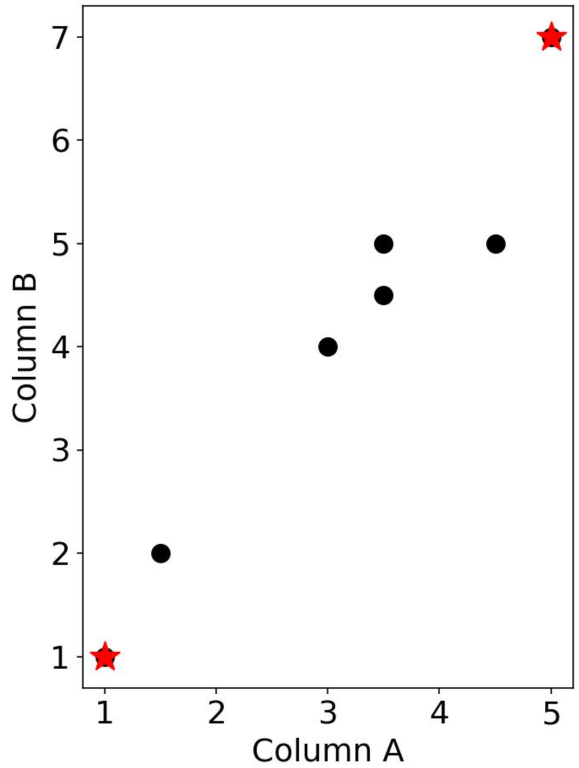
Record	A	B
1	1.0	1.0
2	1.5	2.0
3	3.0	4.0
4	5.0	7.0
5	3.5	5.0
6	4.5	5.0
7	3.5	4.5



Computing Steps (1/5)

- Select the initial partition (i.e., initial 2 clusters)
  - We may select two records randomly.
  - But let us select records 1 and 4, whose A & B feature values are farthest apart (using the Euclidean distance measure).
  - The initial clusters, and their centroids, are as follows:

	Record	Mean Vector (centroid)
Cluster 1	1	(1.0, 1.0)
Cluster 2	4	(5.0, 7.0)



Computing Steps (2/5)

- Examine the remaining records one at a time, and add it to the closest of the 2 clusters (in terms of Euclidean distance to the cluster centroid).
- The mean vector of the cluster is recalculated each time a new record is added to the cluster.
  - record 1 (1.0, 1.0), record 2 (1.5, 2.0) → centroid ((1.0+1.5)/2, (1.0+2.))/2) = (1.3, 1.5)
- The following is the result of adding records 2 and 3 (both to Cluster-1). Note the changes in the Cluster-1 centroid.

	Cluster 1		Cluster 2	
Step	Record	Mean Vector (centroid)	Record	Mean Vector (centroid)
1	1	(1.0, 1.0)	4	(5.0, 7.0)
2	1, 2	(1.3, 1.5)	4	(5.0, 7.0)
3	1, 2, 3	(1.8, 2.3)	4	(5.0, 7.0)

Computing Steps (3/5)

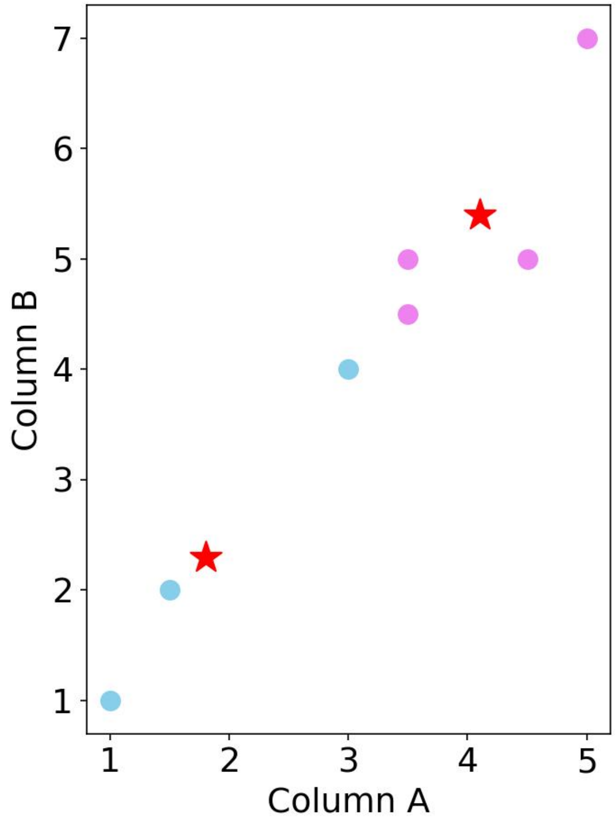
- The following is the result of adding records 5,6, and 7 (all to Cluster-2). Note the changes in the Cluster-2 centroid.
- Note: The centroid is a computer point (not a real point)

	Cluster 1		Cluster 2	
Step	Record	Mean Vector (centroid)	Record	Mean Vector (centroid)
1	1	(1.0, 1.0)	4	(5.0, 7.0)
2	1, 2	(1.2, 1.5)	4	(5.0, 7.0)
3	1, 2, 3	(1.8, 2.3)	4	(5.0, 7.0)
4	**1, 2, 3**	**(1.8, 2.3)**	**4, 5**	**(4.2, 6.0)**
5	**1, 2, 3**	**(1.8, 2.3)**	**4, 5, 6**	**(4.3, 5.7)**
6	**1, 2, 3**	**(1.8, 2.3)**	**4, 5, 6, 7**	**(4.1, 5.4)**

Computing Steps (3/5) cont'd

- The following is the (initial) clustering result.

	Record	Mean Vector (centroid)
Cluster 1	**1, 2, 3**	**(1.8, 2.3)**
Cluster 2	**4, 5, 6, 7**	**(4.1, 5.4)**



Computing Steps (4/5)

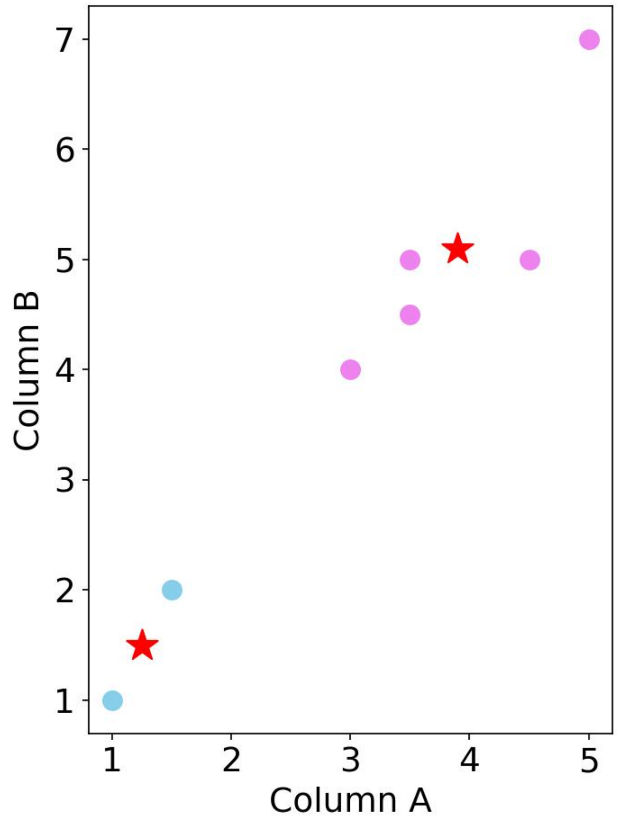
- We need to make sure every record is in the correct cluster (i.e., closer to its own cluster centroid than the other cluster).
- The result of comparison is as follows.
- Record 3 (in Cluster-1) is actually closer to Cluster-2 !

Record	Distance to mean (centroid) of Cluster 1	Distance to mean (centroid) of Cluster 2
1	1.5	5.4
2	0.4	4.3
3	2.1	1.8
4	5.7	1.8
5	3.2	0.7
6	3.8	0.6
7	2.8	1.1

Computing Steps (5/5)

- Move record 3 to Cluster 2.
- The following is the result.

	Record	Mean Vector (centroid)
Cluster 1	**1, 2**	**(1.3, 1.5)**
Cluster 2	**3, 4, 5, 6, 7**	**(3.9, 5.1)**



- In this example, this is the final cluster solution.
- However, the iterative movement of records would continue until no more movements occur.

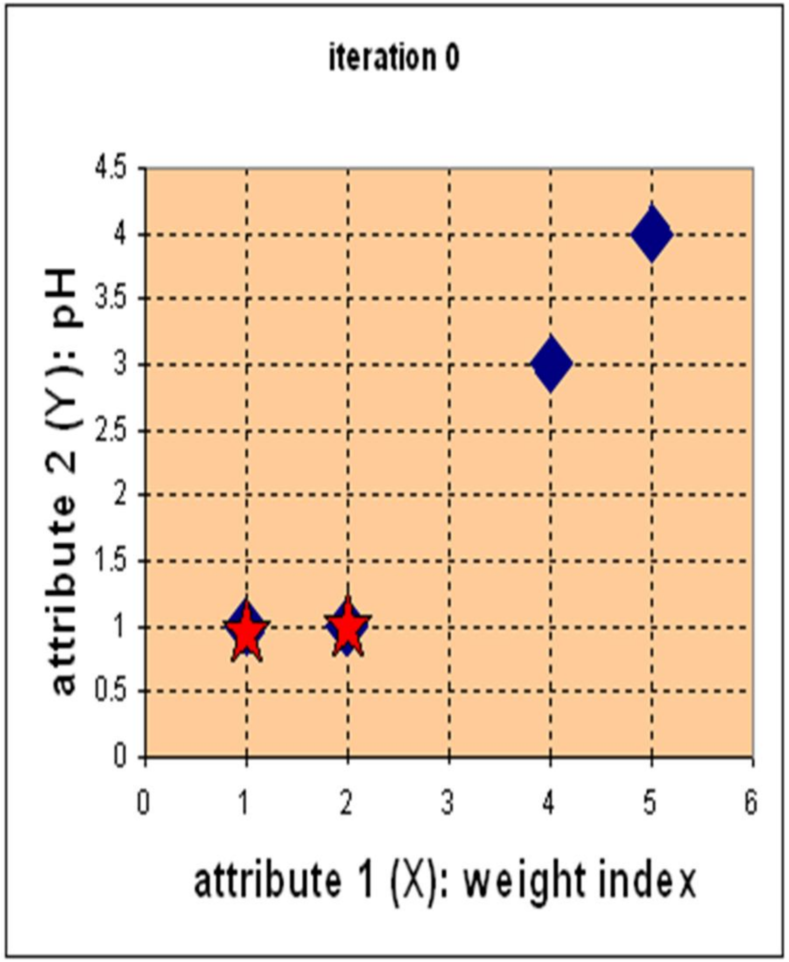
Walkthrough Example 2

- The dataset has data about 4 types of medicine, and each medicine has 2 attributes (weight, pH (measure of acidity or alkalinity).
- We want to group them into 2 clusters

object	weight(X)	pH(Y)
medicine A	1	1
medicine B	2	1
medicine C	4	3
medicine D	5	4

Computing Steps (1/7)

- Randomly select medicine A and medicine B as the first centroids (red stars in the plot).
- Let c_1 and c_2 denote the coordinates of the centroids. Then $c_1=(1,1)$ and $c_2=(2,1)$



Computing Steps (2/7)

- Compute the Euclidean distance between cluster centroid and each object.
- The distance matrix at iteration 0 is shown below.

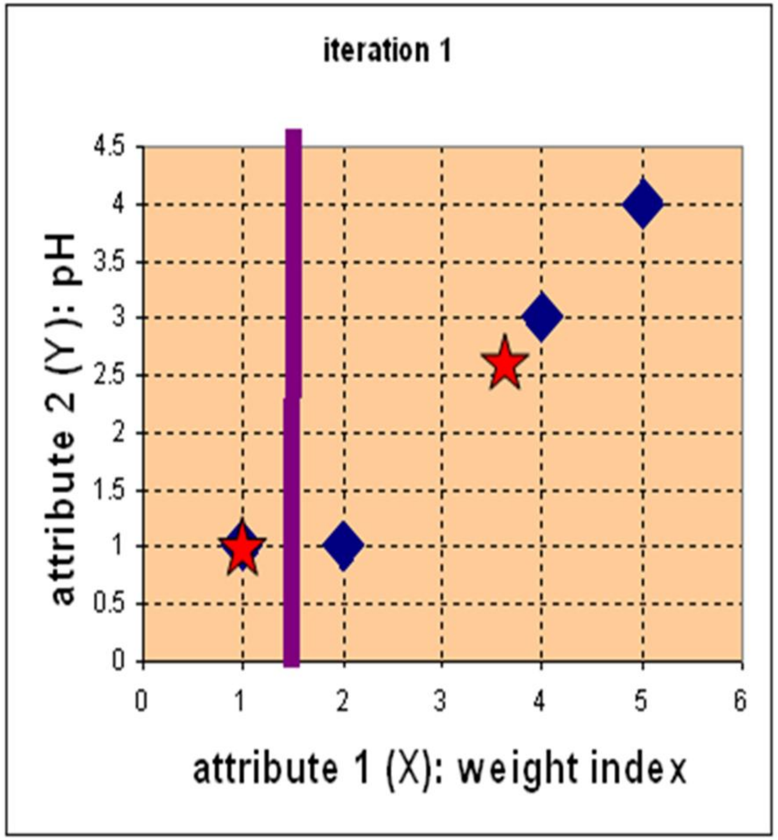
	A	B	C	D
c_1	0	1	3.61	5
c_2	1	0	2.83	4.24

$c_1 = (1, 1)$ cluster-1
$c_2 = (2, 1)$ cluster-2

- The first row of the distance matrix corresponds to the distance of each object to the first centroid, and the second row the second centroid.
- (e.g.) distance from medicine C = (4, 3) to the first centroid c_1 is $\sqrt{(4-1)^2+(3-1)^2}=3.61$ and distance to the second centroid c_2 is $\sqrt{(4-2)^2+(3-1)^2}=2.83$ etc.

Computing Steps (3/7)

- Assign each object to the closest cluster.
  - medicine C and D both go to cluster-2.
- Now, cluster-1 has medicine A.
  cluster-2 has medicine B, C, D.
  (The cluster-2 centroid is shown
  as a red star in the plot.)



Computing Steps (4/7)

- Compute the distance of all objects to the new centroids.
- The revised distance matrix is shown below.

	A	B	C	D
c_1	0	1	3.61	5
c_2	3.14	2.36	0.47	1.89

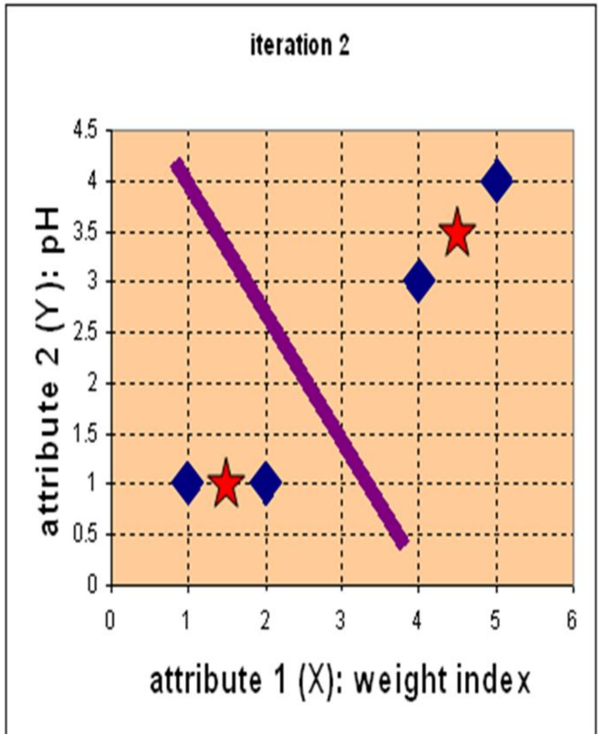
$c_1 = (1, 1)$ cluster-1
$c_2 = (11/3, 8/3)$ cluster-2

Computing Steps (5/7)

- The new distance matrix shows medicine B is closer to cluster-1 than to cluster-2.
- So, medicine B is moved to cluster-1.
- Now calculate the new centroids for both cluster-1 and cluster-2, as cluster-1 gained an object, and cluster-2 lost an object.
- The new centroids are

$$\mathbf{c}_1 = \left(\frac{1+2}{2}, \frac{1+1}{2}\right) = \left(1\tfrac{1}{2}, 1\right)$$

$$\mathbf{c}_2 = \left(\frac{4+5}{2}, \frac{3+4}{2}\right) = \left(4\tfrac{1}{2}, 3\tfrac{1}{2}\right)$$



Computing Steps (6/7)

- Compute the distance of all objects to the new centroids.
- The revised distance matrix is shown below.

	A	B	C	D
c_1	0.5	0.5	3.20	4.61
c_2	4.30	3.54	0.71	0.71

$c_1 = (1.5, 1)$ cluster-1
$c_2 = (4.5, 3.5)$ cluster-2

Computing Steps (7/7)

- Assign each object to the closest cluster.
- No object moves to a different cluster.
- Thus, the k-means clustering has reached its stability and no more iteration is needed.
- Final Result

object	weight(X)	pH(Y)	Group
medicine A	1	1	1
medicine B	2	1	1
medicine C	4	3	2
medicine D	5	4	2

Weaknesses of k-Means Clustering

- Applicable only when the mean of data can be defined
  - (i.e.) applicable to numerical data, but not categorical data (e.g., shirt size S,M,L,XL).
- Unable to handle noisy data and outliers
- The number of clusters, K, must be determined up front.
  - Since experiments must be run with different k values anyway, this is not a serious weakness.
- When the dataset is small, selection of initial clusters can impact the result significantly.

Variants of k-Means

- Objectives are to overcome its weaknesses
  - *k*-medoids: less affected by noise and outliers
  - *k*-modes: applies to categorical data
  - CLARA: deals with large data sets
  - mixture models (EM algorithm): addresses uncertainty of clusters

Notes

- For large datasets, it takes hundreds of iterations for the *k*-means algorithm to complete.
- In the Scikit-learn library, the default maximum iteration count is 300.
- The performance of the algorithm is affected by initialization and distance measure chosen (Euclidean vs. Manhattan).

Exercise

- (Note: total 4 written problems.)
- 1. Group each of the two walkthrough example datasets into 2 clusters, using two different initial clusters.
- 2. Group each of the datasets into 3 clusters.
- * Submit to CyberCampus all 4 solutions in one WORD file.

k-Means Clustering
Using Pandas and Scikit-Learn

Acknowledgments

- https://datatofish.com/k-means-clustering-python/

Import Libraries

```
import numpy as np
from pandas import DataFrame
from matplotlib import pyplot as plt
from sklearn.cluster import KMeans
```

Create a Pandas Data Frame

```
Data = {'x':
[25,34,22,27,33,33,31,22,35,34,67,54,57,43,50,57,59,52,65,47,49,48,35,33,44,45,38,43,51,46],
      'y':
[79,51,53,78,59,74,73,57,69,75,51,32,40,47,53,36,35,58,59,50,25,20,14,12,20,5,29,27,8,7]
      }

df = DataFrame(Data,columns=['x','y'])
print (df)
```

Data Frame Created

	x	y
0	25	79
1	34	51
2	22	53
3	27	78
4	33	59
5	33	74
6	31	73
7	22	57
8	35	69
9	34	75
10	67	51
11	54	32
12	57	40
13	43	47
14	50	53
15	57	36
16	59	35
17	52	58
18	65	59
19	47	50
20	49	25
21	48	20
22	35	14
23	33	12
24	44	20
25	45	5
26	38	29
27	43	27
28	51	8
29	46	7

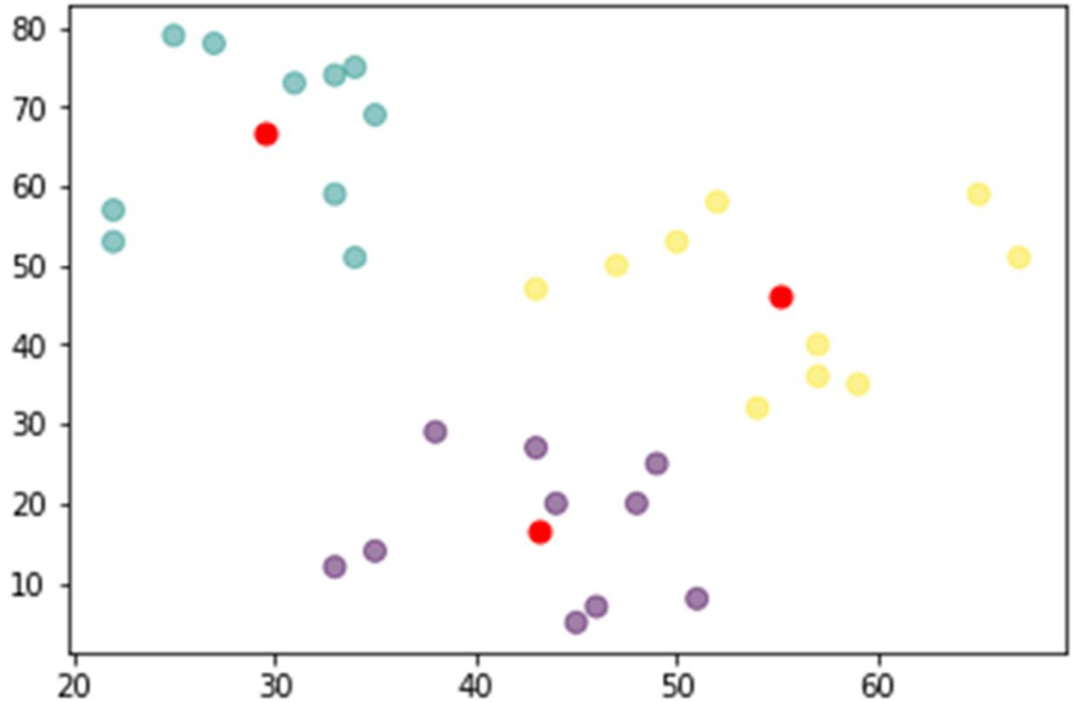
Run K-Means and Display the Clusters

```
# Create 3 clusters

Kmeans = Kmeans(n_clusters=3).fit(df)
Centroids = kmeans.cluster_centers_
Print(centroids)

# Display the 3 clusters

plt.scatter(df['x'], df['y'], c= kmeans.labels_.astype(float),
s=50, alpha=0.5)
plt.scatter(centroids[:, 0], centroids[:, 1], c='red', s=50)
plt.show()
```

Results of Creating 3 Clusters

- 3 centroids, and 3 clusters
- 3 centroids are highlighted in red

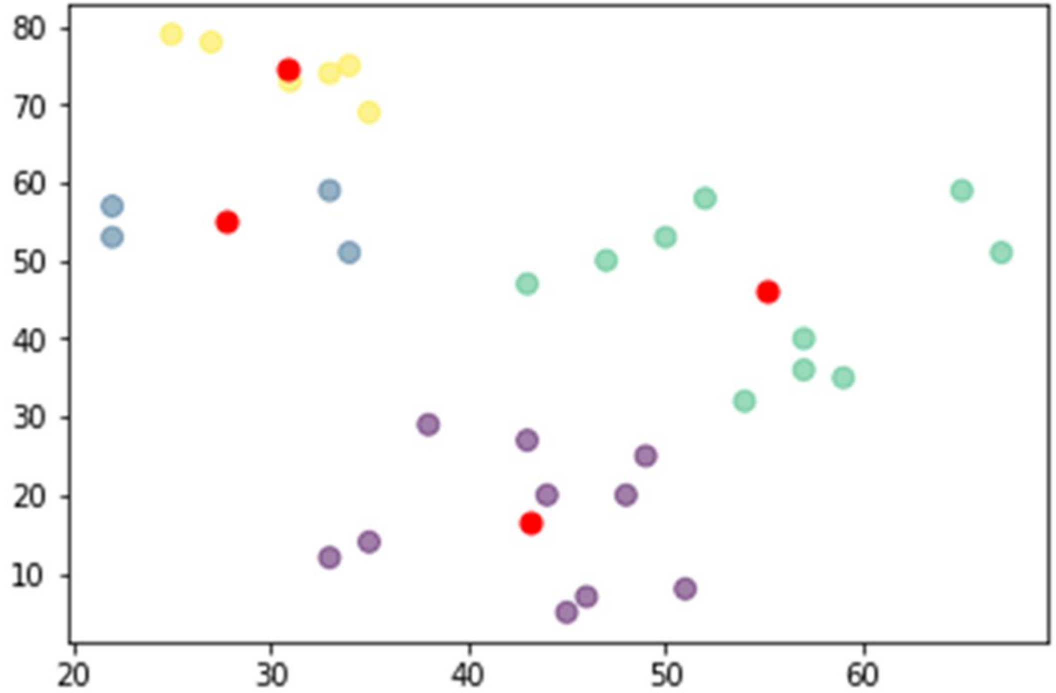
```
[[43.2 16.7]
 [29.6 66.8]
 [55.1 46.1]]
```



Results of Creating 4 Clusters

- By just changing n_clusters=3 to n_clusters=4

```
[[43.2        16.7       ]
 [27.75       55.        ]
 [55.1        46.1       ]
 [30.83333333 74.66666667]]
```



Many Parameters

- Some of the k-Means parameters that can be tuned to improve the model.
  - n_init: number of times the k-means algorithm will be run with different initial centroids (the best result of the total n_init runs is chosen)
  - max_iter: maximum number of iterations of the k-means algorithm for a single run
  - algorithm: (auto, elkan, full) selects a k-means algorithm to use. "auto" chooses "elkan" for dense data and "full" for sparse data.

Roadmap: Clustering

- Overview
- Similarity
- k-Means Clustering
- Hierarchical Agglomerative Clustering

Acknowledgment

- https://onlinecourses.science.psu.edu/stat555

Hierarchical Agglomerative Clustering (HAC)

- “agglomerate” means “gather into a cluster”
- Define each data point as a cluster and merge existing clusters at each step.
- Four different methods
  - Single (Minimum) Linkage
  - Complete (Maximum) Linkage
  - Average Linkage
  - Centroid Method

Hierarchical Agglomerative Clustering Methods (1/2)

- Single Linkage
  - define the distance between two clusters as the minimum between any data point in the first cluster and any data point in the second cluster.
  - At each cluster merging stage, merge the two clusters with the smallest single linkage distance.
- Complete Linkage
  - Define the distance between two clusters as the maximum between any data point in the first cluster and any data point in the second cluster.
  - At each cluster merging stage, merge the two clusters with the smallest complete linkage distance.

Hierarchical Agglomerative Clustering Methods (2/2)

- Average Linkage
  - Define the distance between two clusters to be the average between data points in the first cluster and data points in the second cluster.
  - At each cluster merging stage, merge the two clusters that have the smallest average linkage distance.
- Centroid Method
  - The distance between two clusters is the distance between the two mean vectors of the clusters.
  - At each cluster merging stage, merge the two clusters that have the smallest centroid distance.

Walkthrough Example: Complete Linkage Clustering (1/4)

- 5 data points = 5 initial clusters
- We want to merge 2 clusters into 1, and have 4 clusters.
- Let us create a distance matrix between clusters

	1	**2**	**3**	**4**	**5**
1	0	9	3	6	11
2	9	0	7	5	10
3	3	7	0	9	2
4	6	5	9	0	8
5	11	10	2	8	0

- The smallest distance, 2, is between 3 and 5.
- Remove(merge) the 3 and 5 entries, and replace them by a new entry "35".

Walkthrough Example: Complete Linkage Clustering (2/4)

- Update the distance matrix
  - For every other entry n, compute the "maximum" distance between (n,3) and (n,5): n=1,2,4
  - (e.g.) distance between 4 and "35" is

    max((4,3), (4,5)) = max(9,8) = 9
- The updated distance matrix is shown below.

	1	2	3	4	5
1	0	9	3	6	11
2	9	0	7	5	10
3	3	7	0	9	2
4	6	5	9	0	8
5	11	10	2	8	0

→

	35	1	2	4
35	0	11	10	9
1	11	0	9	6
2	10	9	0	5
4	9	6	5	0

Walkthrough Example: Complete Linkage Clustering (3/4)

- Now, the smallest distance, 5, is between 2 and 4.
- Remove the 2 and 4 entries, and replace them by a new entry "24".
- Update the distance matrix
  - For every other entry n, compute the "maximum" distance between (n,2) and (n,4): n=1, 35.
- The updated distance matrix is shown below.

	35	1	2	4
35	0	11	10	9
1	11	0	9	6
2	10	9	0	5
4	9	6	5	0

→

	24	35	1
24	0	10	9
35	10	0	11
1	9	11	0

Walkthrough Example: Complete Linkage Clustering (4/4)

- Now, the smallest distance, 9, is between 1 and 24.
- Remove the 1 and 24 entries, and replace them by a new entry “124”.
- Update the distance matrix
  - For every other entry n, compute the “maximum” distance between (n,1) and (n,24): n=35.
- The updated distance matrix is shown below.
- * We are done. It makes no sense to create one cluster.

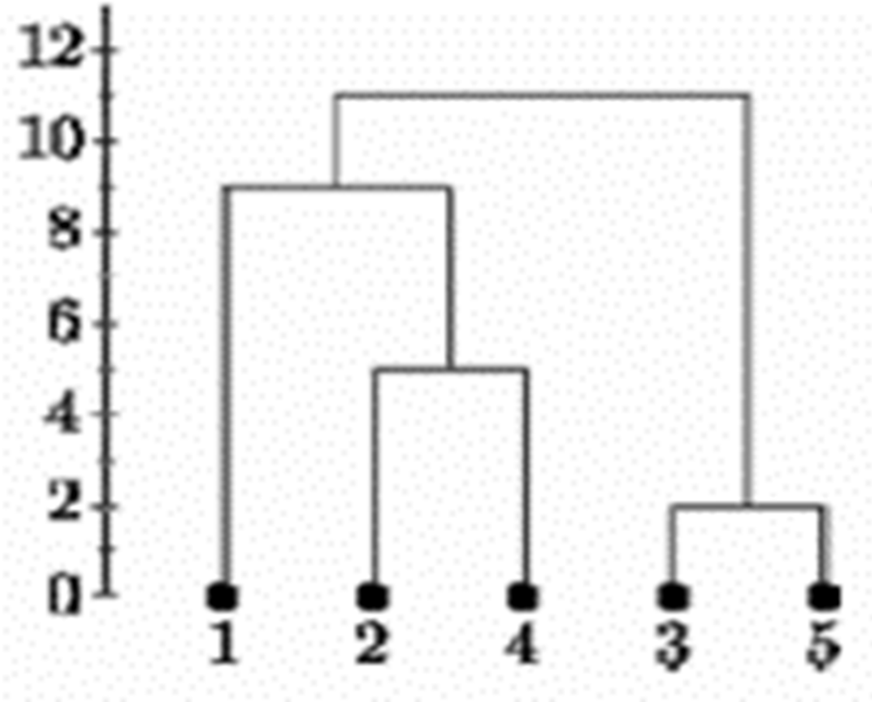
	24	35	1
24	0	10	9
35	10	0	11
1	9	11	0

→

	124	35
124	0	11
35	11	0

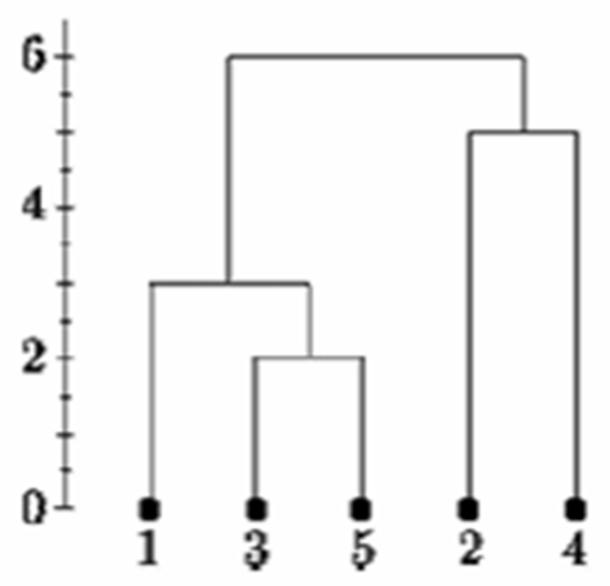
Visual Representation of Complete Linkage Clustering Steps

- The y-axis (cluster height) shows the distance between entries at the time they were clustered.
- It also shows the order in which the entries are merged.



Visual Representation of Single Linkage Clustering Steps

- Below is the single linkage dendrogram for the same distance matrix.
- It starts out with cluster "35" also. However, the distance between every other entry n and "35" is the minimum of (n,3) and (n,5).



Walkthrough Example: Single Linkage Clustering (1/5)

- 6 initial clusters, or a distance matrix between 6 cities in Italy
- Start by finding two entries (cities) with the shortest distance. (MI-TO 138)

	BA	FI	MI	NA	RM	TO
BA	0	662	877	255	412	996
FI	662	0	295	468	268	400
MI	877	295	0	754	564	138
NA	255	468	754	0	219	869
RM	412	268	564	219	0	669
TO	996	400	138	869	669	0

legend: BA (Bari), FI (Firenze), MI (Milano)
NA (Napoli), RM (Rome), TO (Torino)

Walkthrough Example: Single Linkage Clustering (2/5)

- Merge the MI and TO entries into a new entry MI/TO.
- For each entry in the distance matrix, compute the minimum distance between it and {MI,TO}.
- The new distance matrix is shown below.
- Find the next two cities with the shortest distance. (NA-RM 219)

	BA	FI	MI/TO	NA	RM
BA	0	662	877	255	412
FI	662	0	295	468	268
MI/TO	877	295	0	754	564
NA	255	468	754	0	219
RM	412	268	564	219	0

Walkthrough Example: Single Linkage Clustering (3/5)

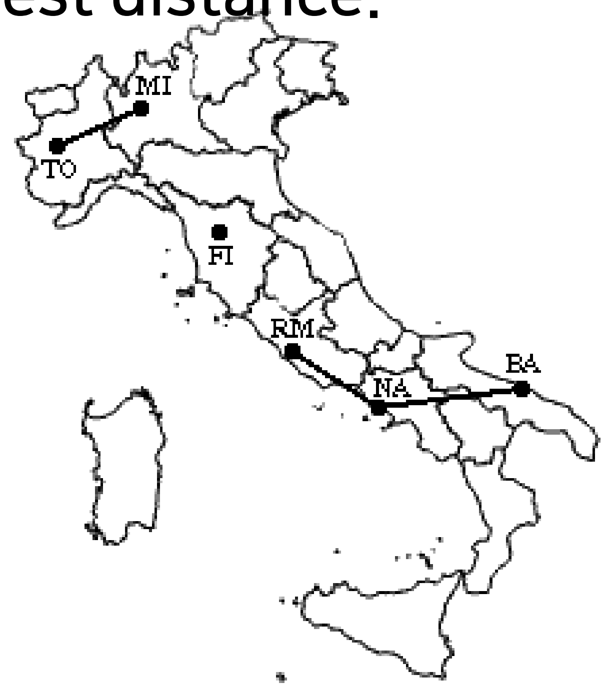
- Merge the NA and RM entries into NA/RM, and
- For each entry in the distance matrix, compute the minimum distance between it and {NA,RM}.
- The new distance matrix is shown below.
- Find the next two cities with the shortest distance. (BA-(NA/RM) 255)

	BA	FI	MI/TO	NA/RM
BA	0	662	877	255
FI	662	0	295	268
MI/TO	877	295	0	564
NA/RM	255	268	564	0

Walkthrough Example: Single Linkage Clustering (4/5)

- Merge the BA and NA/RM entries into BA/NA/RM, and
- For each entry in the distance matrix, compute the minimum distance between it and {BA,NA,RM}.
- The new distance matrix is shown below.
- Find the next two cities with the shortest distance. (FI-(BA/NA/RM) 268)

	BA/NA/RM	FI	MI/TO
BA/NA/RM	0	268	564
FI	268	0	295
MI/TO	564	295	0



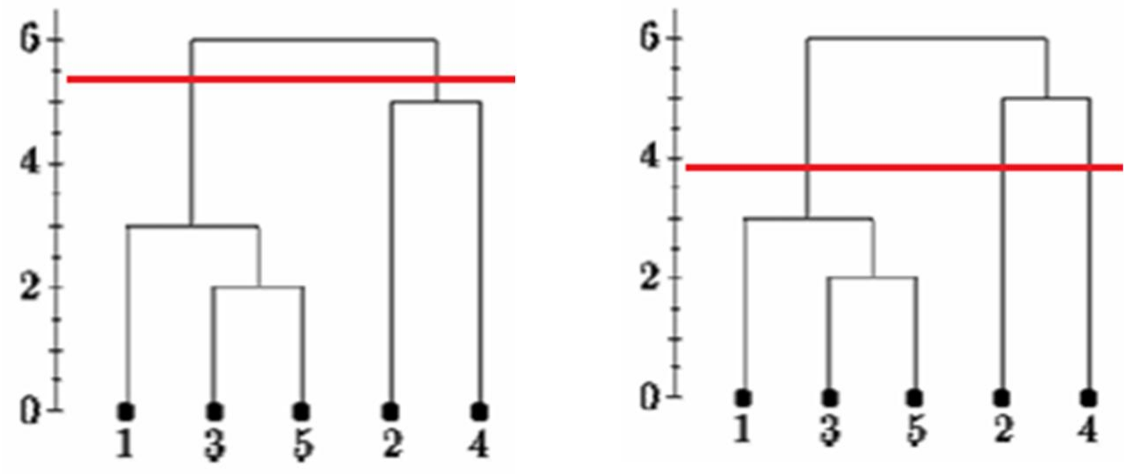
Walkthrough Example: Single Linkage Clustering (5/5)

- Merge the FI and BA/NA/RM entries into BA/FI/NA/RM, and
- For each entry in the distance matrix, compute the minimum distance between it and {BA,FI,NA,RM}.
- The new distance matrix is shown below.
- We have only two clusters and we are done.

	BA/FI/NA/RM	MI/TO
BA/FI/NA/RM	0	295
MI/TO	295	0

Determining Clusters

- There is no objective way to know how many clusters we want to create.
- If we cut the single linkage tree as shown below on the left, there would be 2 clusters.
- However, if we cut the same tree lower as shown below on the right, there would be 1 cluster and 2 singletons.



Exercise

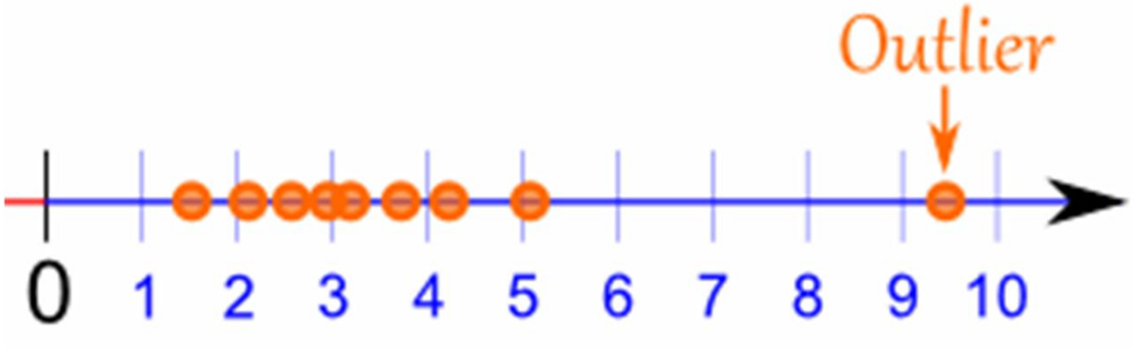
- Using the “distance between 6 Italian cities“ dataset, work through the complete, and average linkage methods.
- Submit a single WORD file to CyberCampus.

Roadmap: Data Mining Algorithms

- Regression
- Classification
- Clustering
- Outlier Detection

Outlier Detection

- Discover data that is very different from the rest of the data in a dataset



- Two scenarios for outlier detection
  - When outliers are unimportant: drop them, to avoid wasted computation time and memory, and prevent them from influencing the learning models
  - When outliers are important: drop non-outliers, and keep only the outliers; focus on it for in-depth analysis

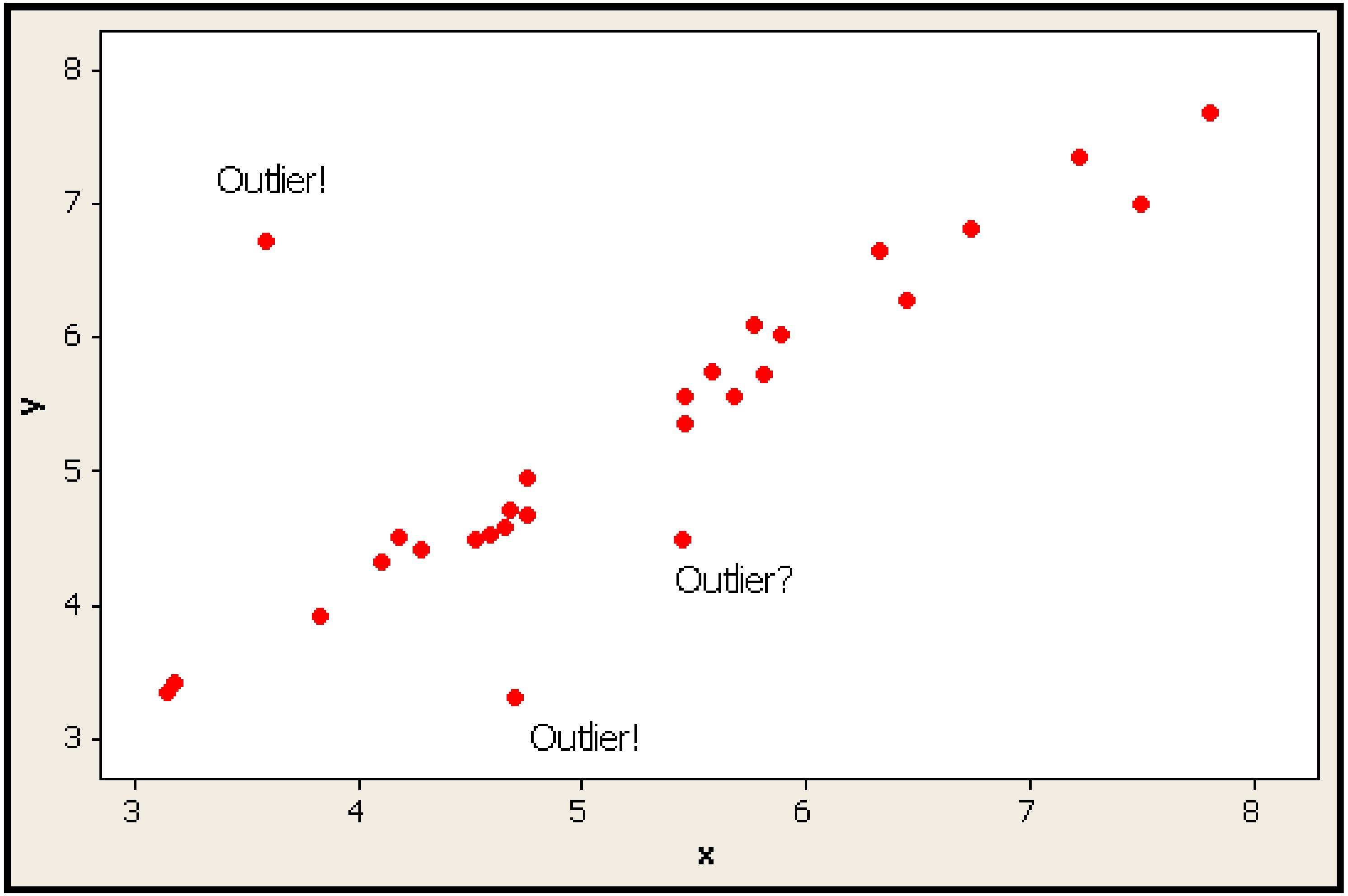
Applications Where Outliers Are Important

- Fraud detection, intrusion detection
- Medicine (cancer diagnosis), public health (virus infection)
- Mechanical parts failure
- Measurement error detection

Outlier Detection Techniques

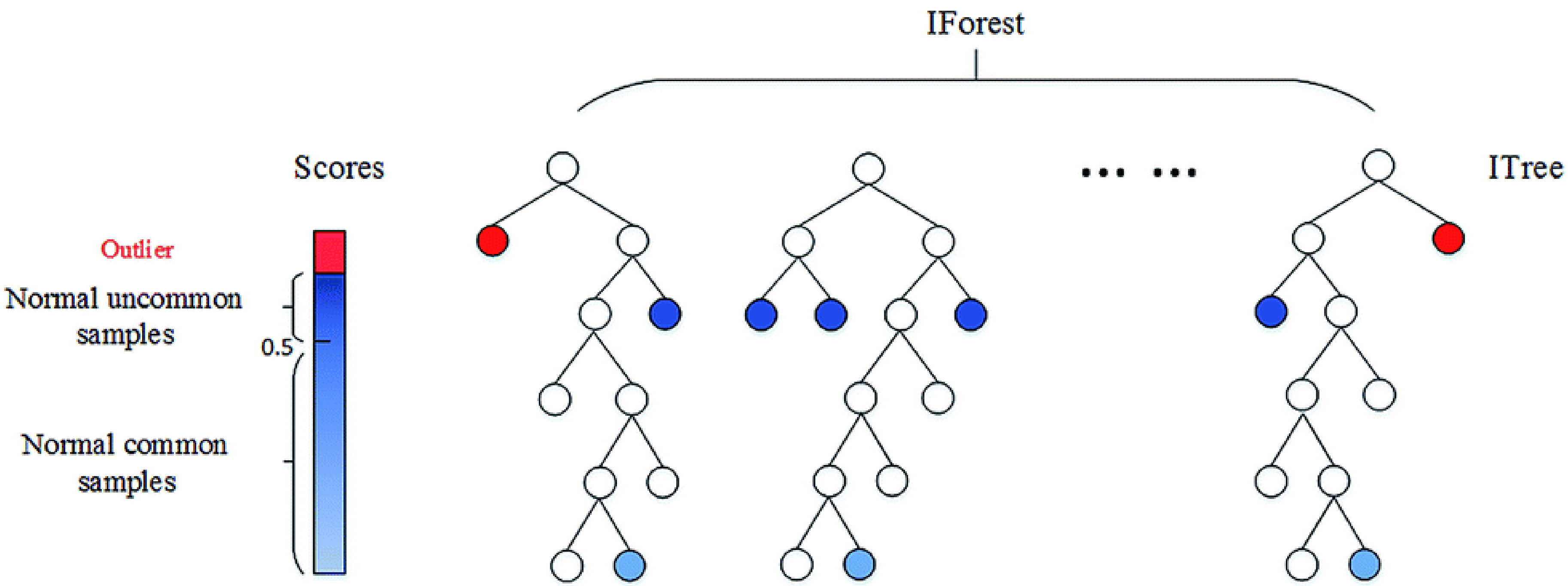
- statistics and linear algebra
  - central tendency and dispersion
  - distribution curve, histogram, boxplot, etc.
- machine learning and data visualization
  - regression, classification, clustering

Using Regression

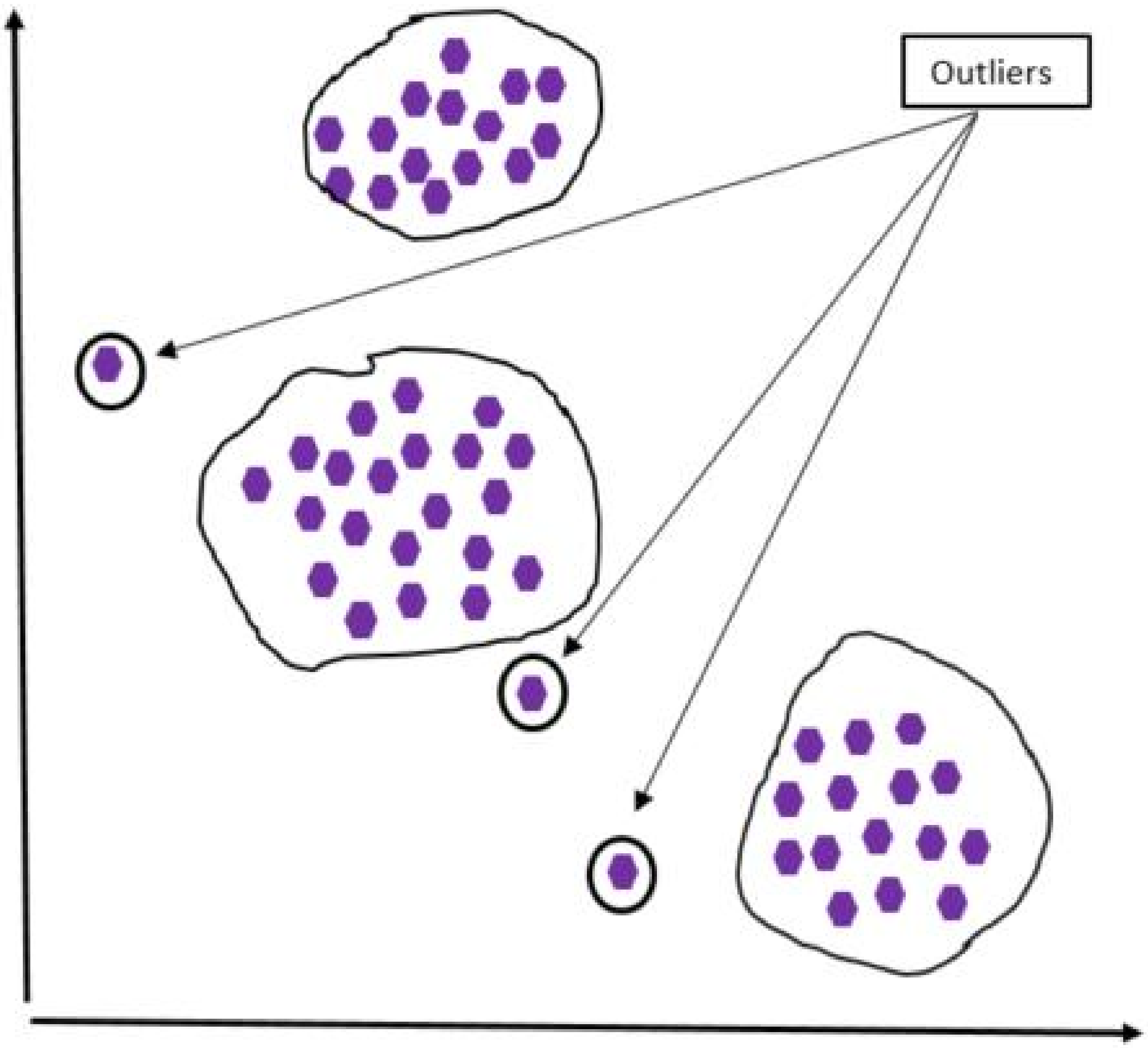


Using Classification Algorithms

- Isolation Forest



Using Clustering Algorithms



End of Class