

# Statistics Foundation

## Unsupervised ML - Clustering

# Machine Learning

---

- **What is Machine Learning ?**
  - Consider you are trying to toss a paper to a dustbin.
  - After first attempt, you realize that you have put too much force in it.
  - After second attempt, you realize you are closer to target but you need to increase your throw angle.
  - What is happening here is basically after every throw we are learning something and improving the end result. We are programmed to learn from our experience..

# Machine Learning

---

- **Types of Machine Learning**

Machine learning implementations are classified into 2 major categories, depending on the nature of the learning “signal” or “response” available to a learning system which are as follows:-

**Supervised learning :**

- When an algorithm learns from example data and associated target responses that can consist of numeric values or string labels, such as classes or tags, in order to later predict the correct response when posed with new examples comes under the category of Supervised learning
- This approach is indeed similar to human learning under the supervision of a teacher
- The teacher provides good examples for the student to memorize, and the student then derives general rules from these specific examples

# Machine Learning

---

- **Types of Machine Learning**

- Unsupervised learning :**

- Whereas when an algorithm learns from plain examples without any associated response, leaving to the algorithm to determine the data patterns on its own
    - This type of algorithm tends to restructure the data into something else, such as new features that may represent a class or a new series of un-correlated values
    - They are quite useful in providing humans with insights into the meaning of data and new useful inputs to supervised machine learning algorithms
    - As a kind of learning, it resembles the methods humans use to figure out that certain objects or events are from the same class, such as by observing the degree of similarity between objects

# Machine Learning

---

- **Categorization based on output**

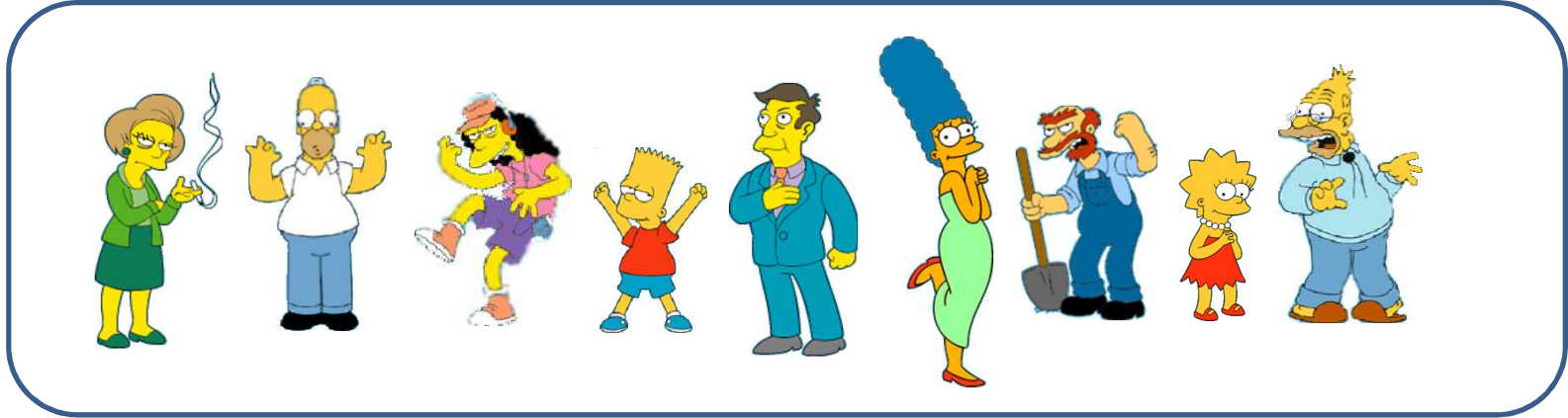
- **Classification** : When inputs are divided into two or more classes, and the learner must produce a model that assigns unseen inputs to one or more (multi-label classification) of these classes. This is typically tackled in a supervised way. Spam filtering is an example of classification, where the inputs are email (or other) messages and the classes are “spam” and “not spam”
- **Regression** : Which is also a supervised problem, A case when the outputs are continuous rather than discrete
- **Clustering** : When a set of inputs is to be divided into groups. Unlike in classification, the groups are not known beforehand, making this typically an unsupervised task

# Clustering

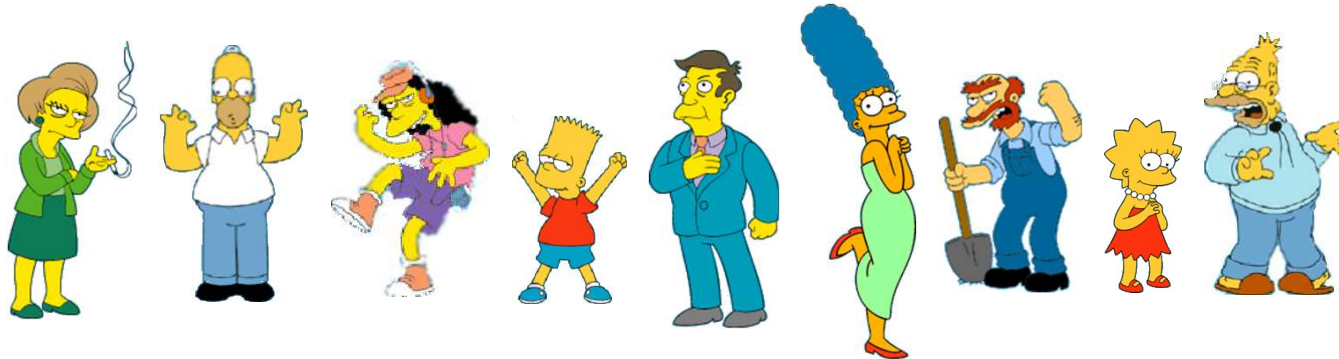
---

- **What is Clustering?**
  - *Also called unsupervised learning, sometimes called classification by statisticians and sorting by psychologists and segmentation by people in marketing*
  - Organizing data into classes such that there is
    - high intra-class similarity
    - low inter-class similarity
  - Finding the class labels and the number of classes directly from the data (in contrast to classification)
  - More informally, finding natural groupings among objects

What is a natural grouping among these objects?

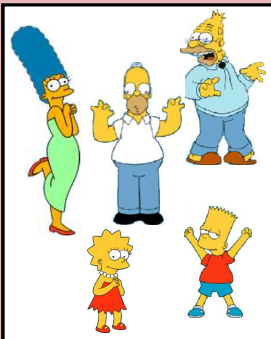


What is a natural grouping among these objects?



Clustering is subjective

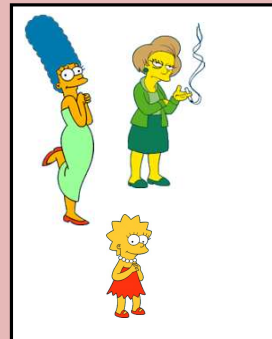
Simpson's Family



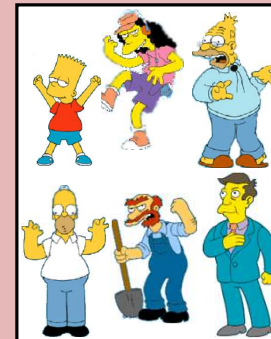
School Employees



Females



Males





# What is Similarity?

The quality or state of being similar; likeness; resemblance; as, a similarity of features.

**Webster's Dictionary**

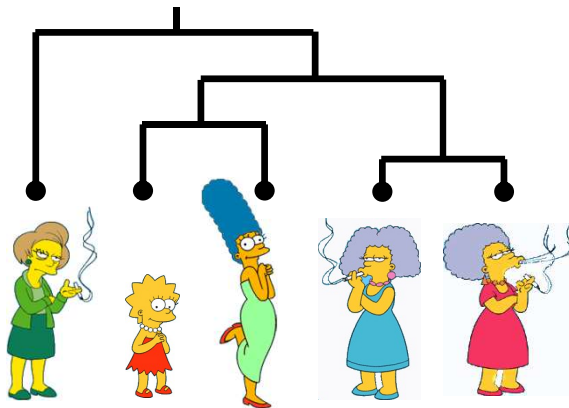


Similarity is hard to define, but...  
*"We know it when we see it"*

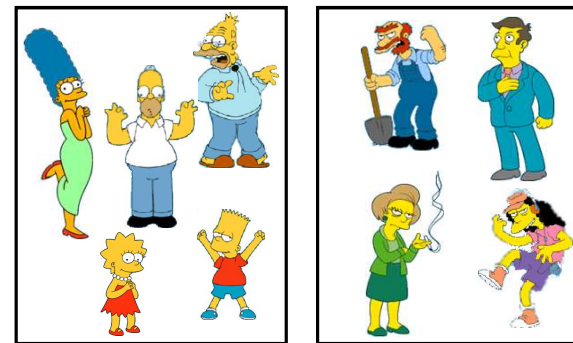
# Two Types of Clustering

- Partitional algorithms: Construct various partitions and then evaluate them by some criterion
- Hierarchical algorithms: Create a hierarchical decomposition of the set of objects using some criterion

**Hierarchical**



**Partitional**

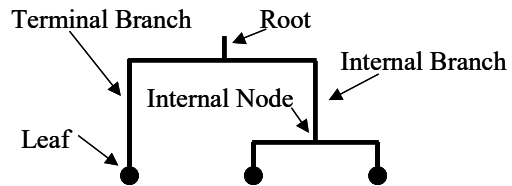


# **Desirable Properties of a Clustering Algorithm**

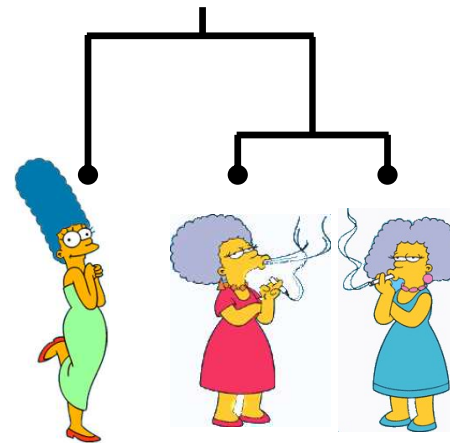
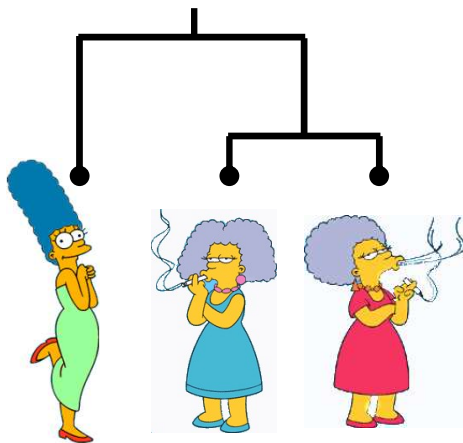
- Scalability (in terms of both time and space)
- Ability to deal with different data types
- Minimal requirements for domain knowledge to determine input parameters
- Able to deal with noise and outliers
- Insensitive to order of input records
- Incorporation of user-specified constraints
- Interpretability and usability

## A Useful Tool for Summarizing Similarity Measurements

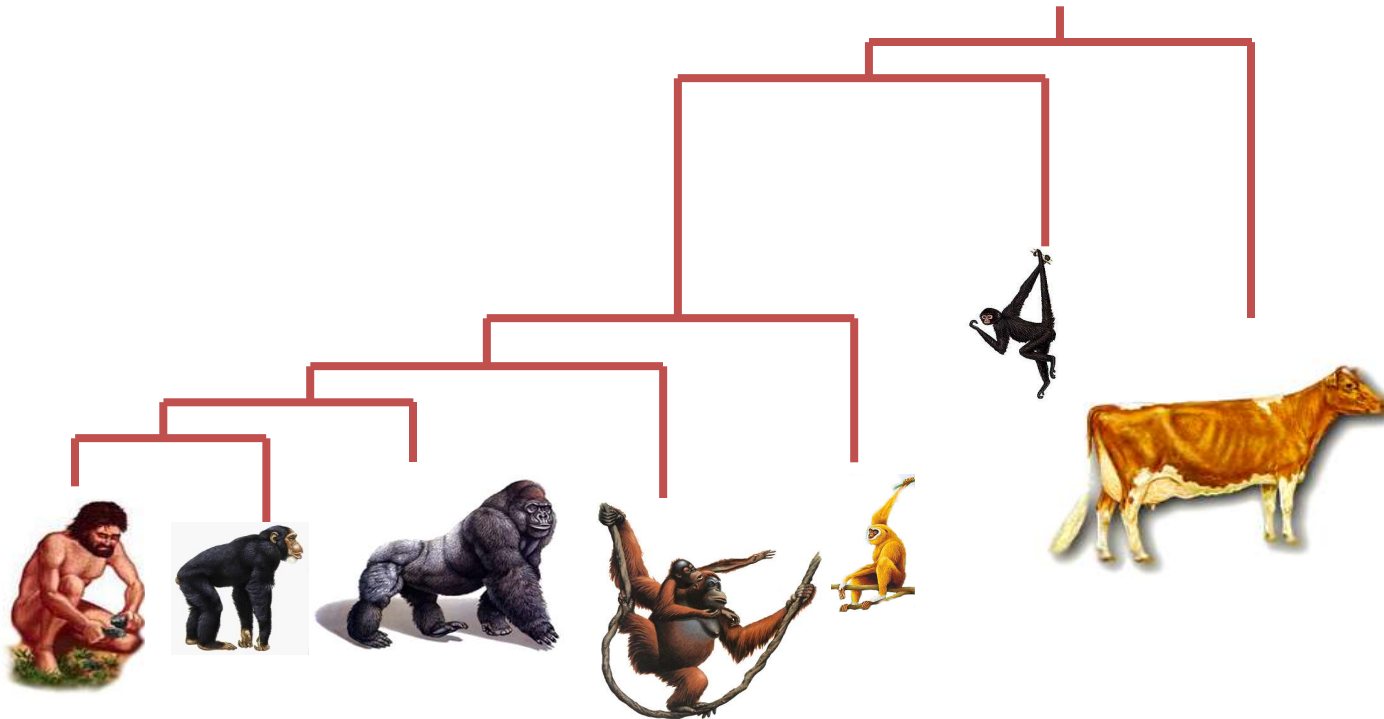
### Dendrogram



The similarity between two objects in a dendrogram is represented as the height of the lowest internal node they share.

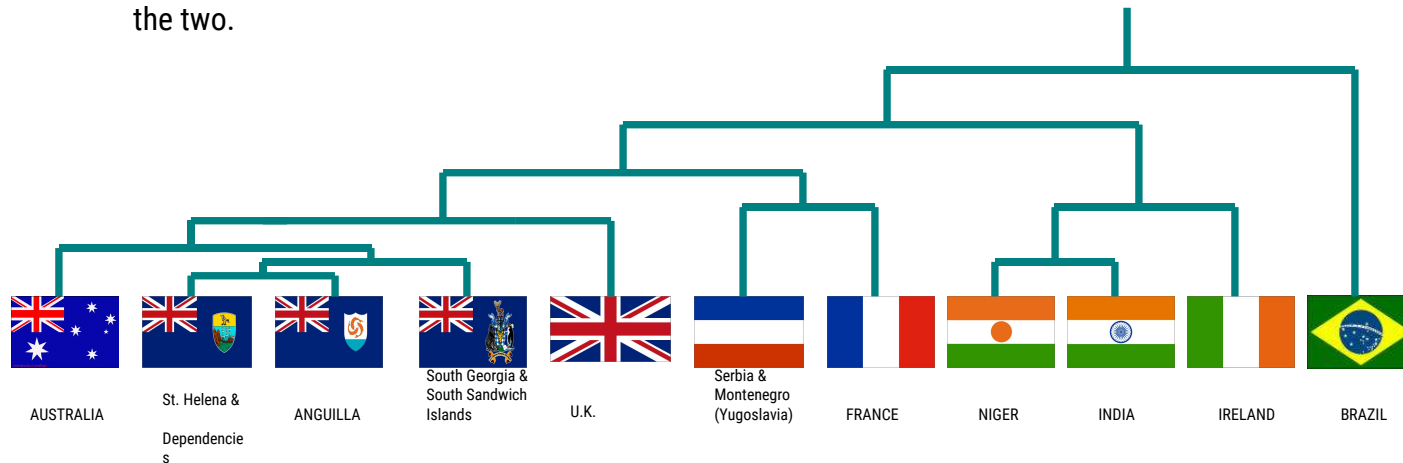


**There is only one dataset that can be perfectly clustered using a hierarchy...**



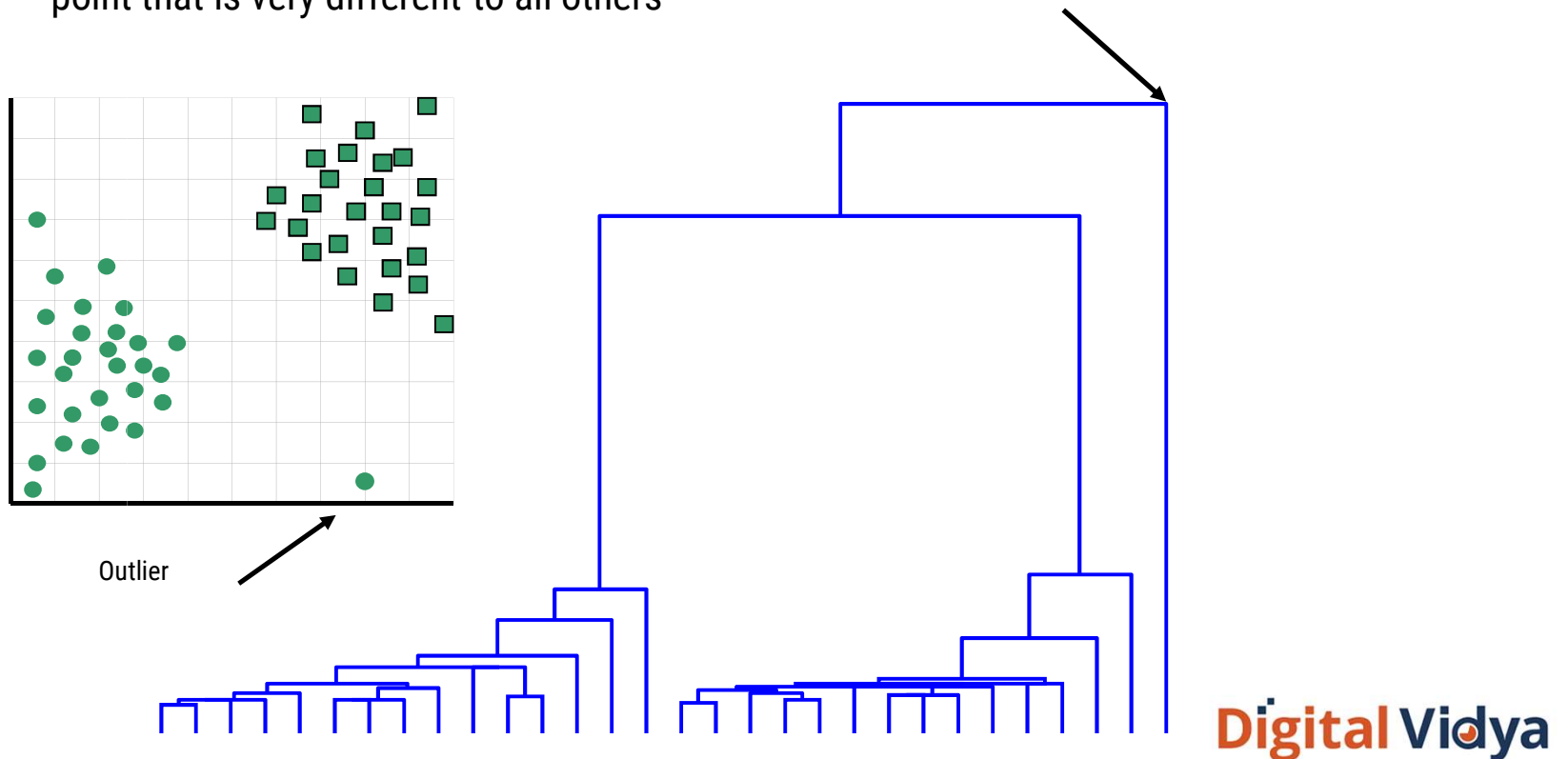
# Hierarchical clustering can sometimes show patterns that are meaningless or spurious

- For example, in this clustering, the tight grouping of Australia, Anguilla, St. Helena etc is meaningful, since all these countries are former UK colonies.
- However the tight grouping of Niger and India is completely spurious, there is no connection between the two.



## One potential use of a dendrogram is to detect outliers

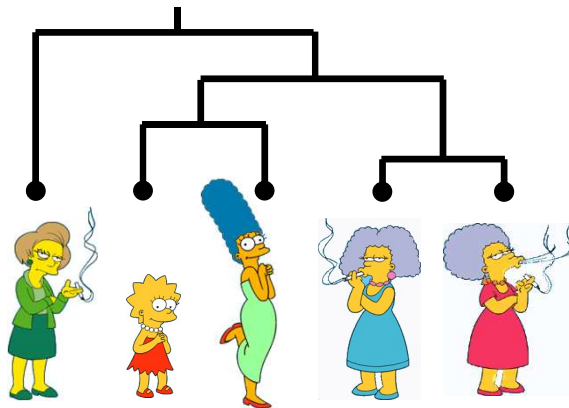
The single isolated branch is suggestive of a data point that is very different to all others



# (How-to) Hierarchical Clustering

The number of dendrograms with  $n$  leaves =  $(2n - 3)! / [(2^{n-2}) (n - 2)!]$

Number of Leafs	Number of Possible Dendrograms
2	1
3	3
4	15
5	105
...	...
10	34,459,425



Since we cannot test all possible trees we will have to heuristic search of all possible trees. We could do this..

**Bottom-Up (agglomerative):** Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.











**Top-Down (divisive):** Starting with all the data in a single cluster, consider every possible way to divide the cluster into two. Choose the best division and recursively operate on both sides.



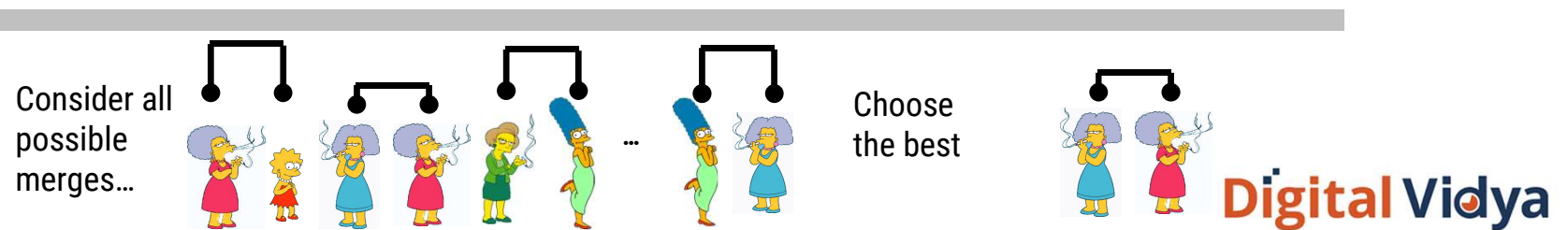
We begin with a distance matrix which contains the distances between every pair of objects in our database.

$$D(\text{Marge Simpson}, \text{Lisa Simpson}) = 8$$

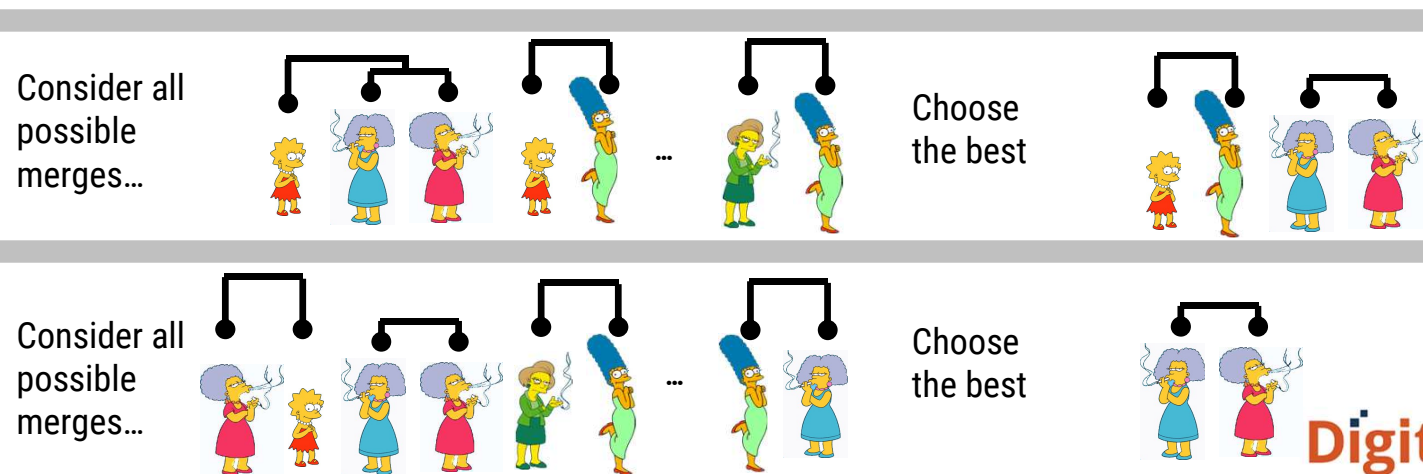
$$D(\text{Marge Simpson}, \text{Maggie Simpson}) = 1$$

					
	0	8	8	7	7
		0	2	4	4
			0	3	3
				0	1
					0

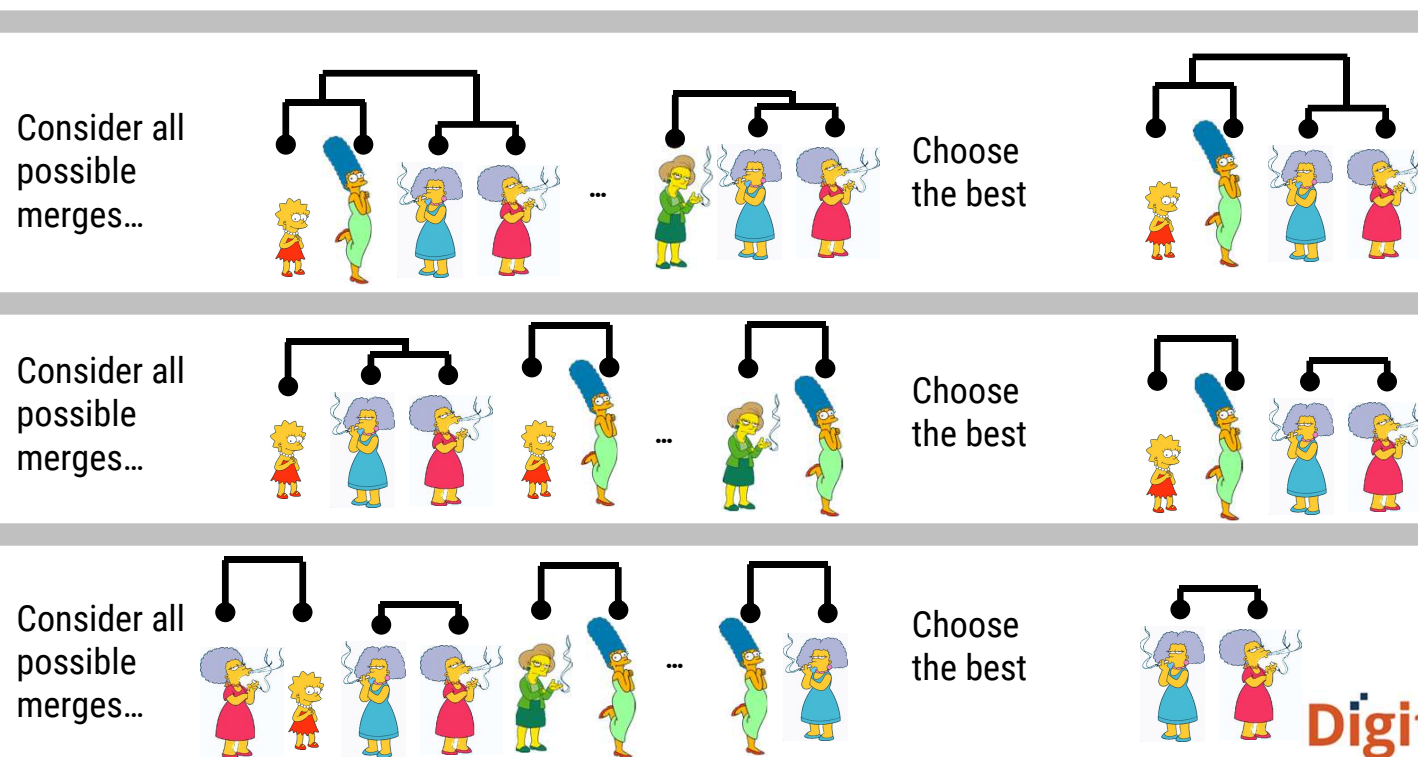
**Bottom-Up (agglomerative):** Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.



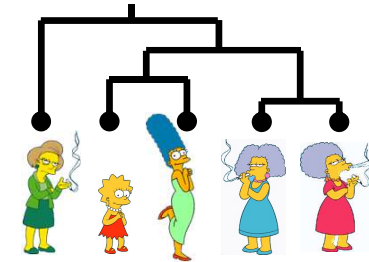
**Bottom-Up (agglomerative):** Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.



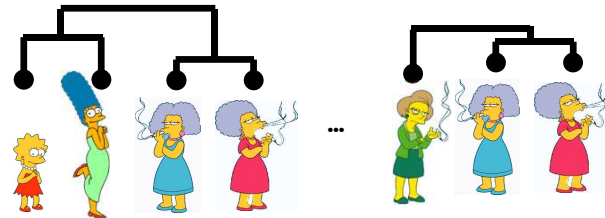
**Bottom-Up (agglomerative):** Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.



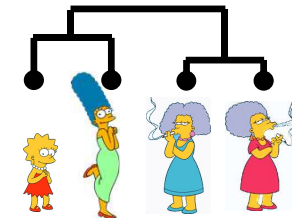
**Bottom-Up (agglomerative):** Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.



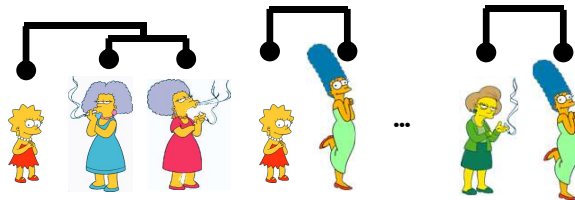
Consider all possible merges...



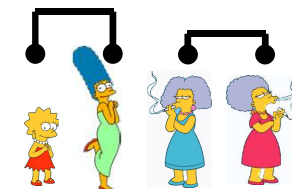
Choose the best



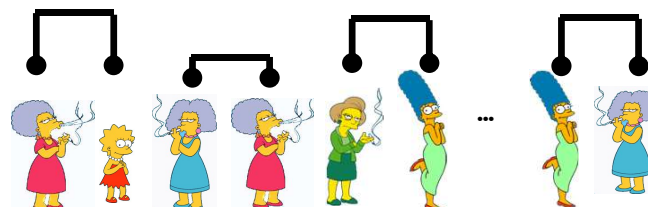
Consider all possible merges...



Choose the best



Consider all possible merges...

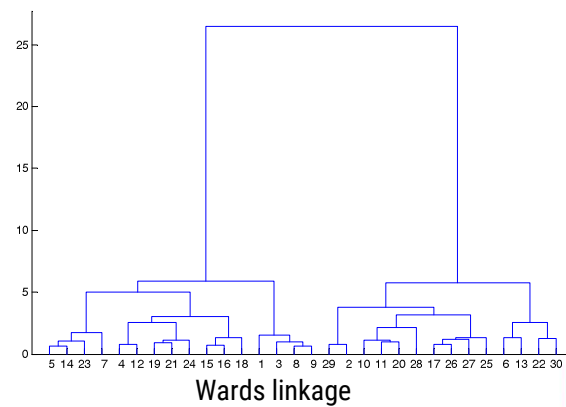
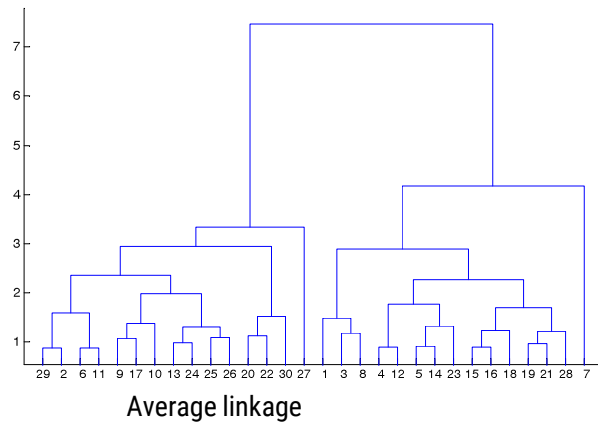
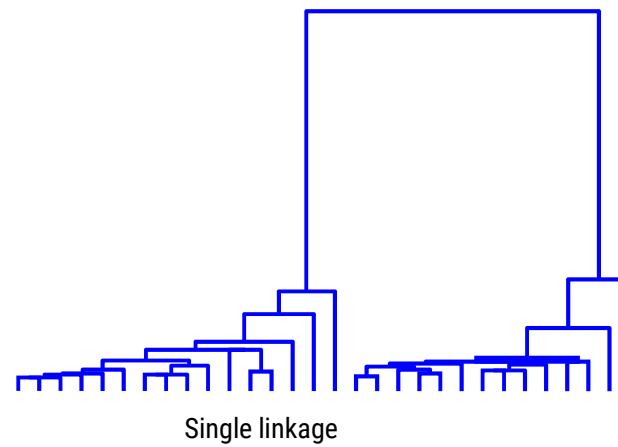
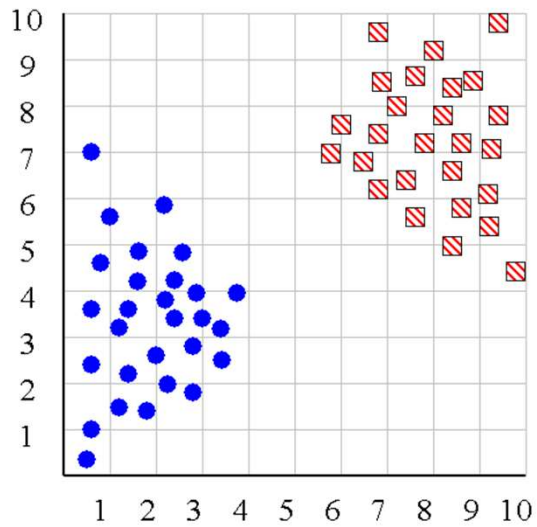


Choose the best



*We know how to measure the distance between two objects, but defining the distance between an object and a cluster, or defining the distance between two clusters is non obvious.*

- **Single linkage (nearest neighbor):** In this method the distance between two clusters is determined by the distance of the two closest objects (nearest neighbors) in the different clusters
- **Complete linkage (furthest neighbor):** In this method, the distances between clusters are determined by the greatest distance between any two objects in the different clusters (i.e., by the "furthest neighbors")
- **Group average linkage:** In this method, the distance between two clusters is calculated as the average distance between all pairs of objects in the two different clusters
- **Wards Linkage:** In this method, we try to minimize the variance of the merged clusters



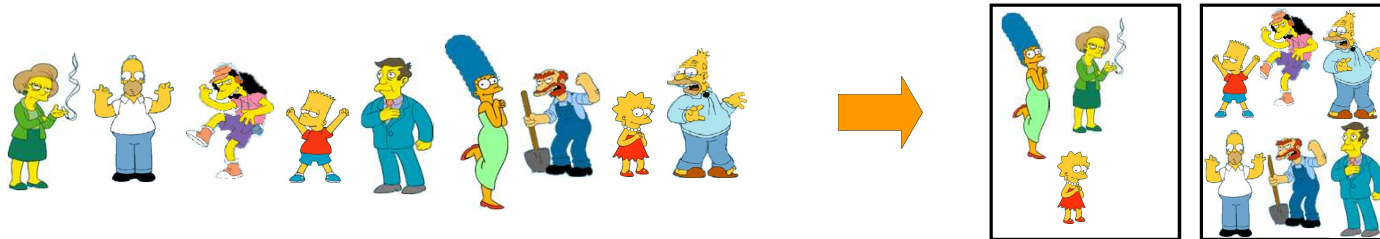
## Summary of Hierarchical Clustering Methods

- No need to specify the number of clusters in advance
- Hierarchical nature maps nicely onto human intuition for some domains
- They do not scale well: time complexity of at least  $O(n^2)$ , where  $n$  is the number of total objects
- Interpretation of results is (very) subjective



# Partitional Clustering

- Nonhierarchical, each instance is placed in exactly one of  $K$  non-overlapping clusters.
- Since only one set of clusters is output, the user normally has to input the desired number of clusters  $K$ .



# K-means

---

Most well-known and popular clustering algorithm:

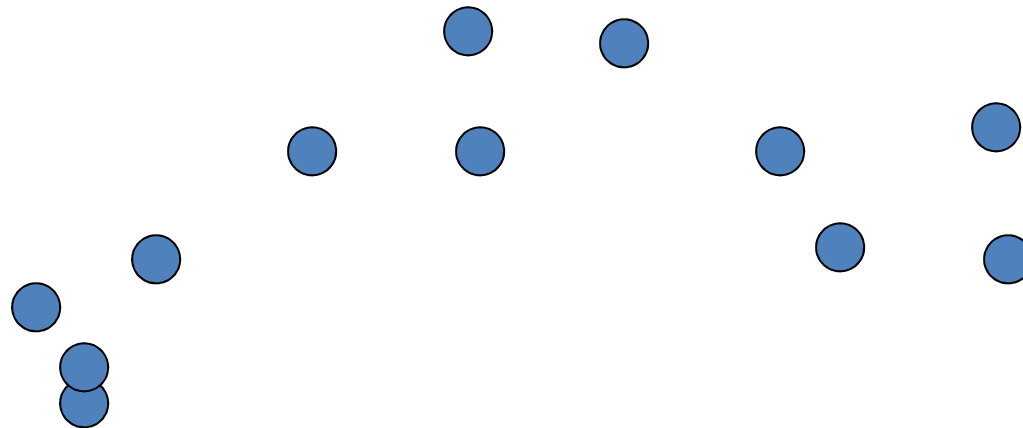
Start with some initial cluster centers

Iterate:

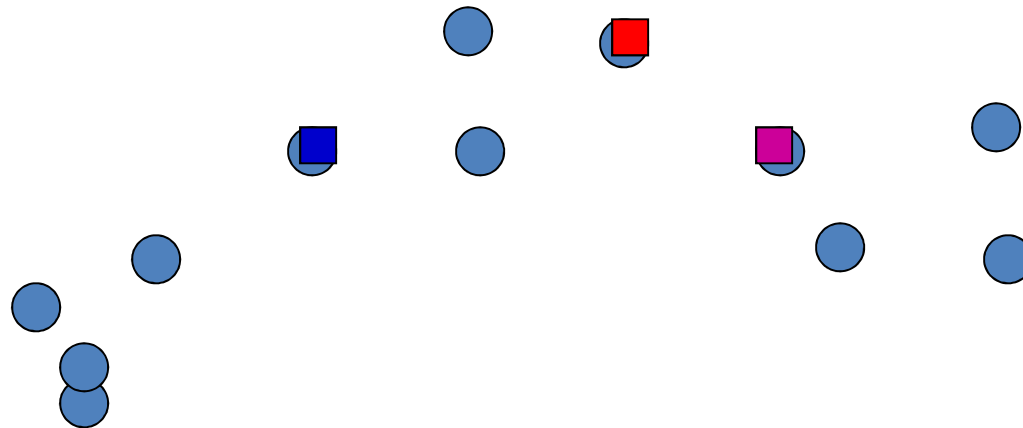
- Assign/cluster each example to closest center
- Recalculate centers as the mean of the points in a cluster

# K-means: an example

---

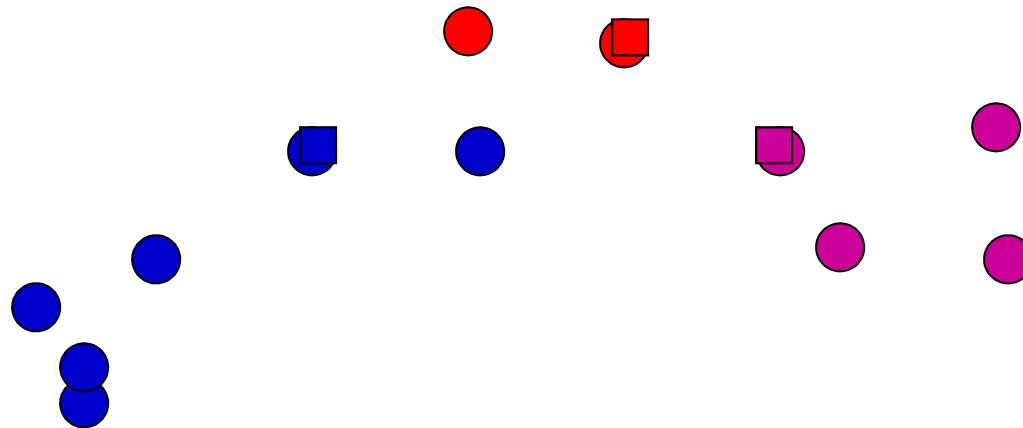


# K-means: Initialize centers randomly



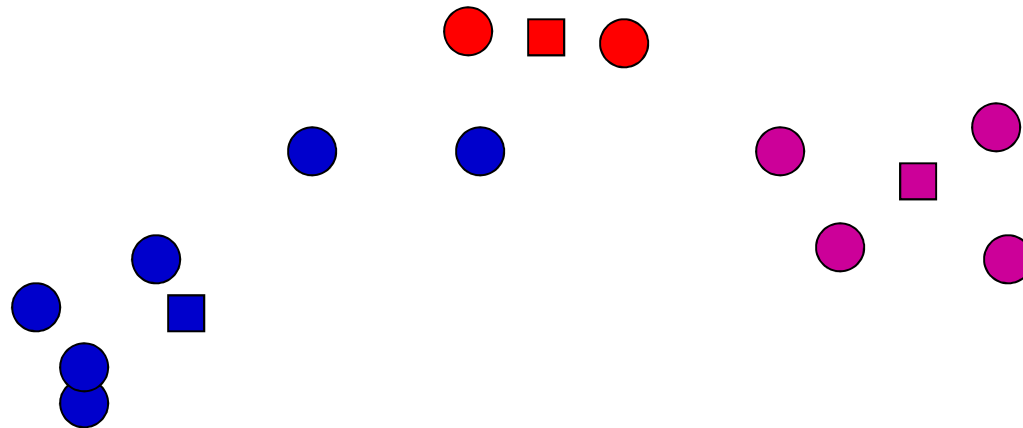
# K-means: assign points to nearest center

---



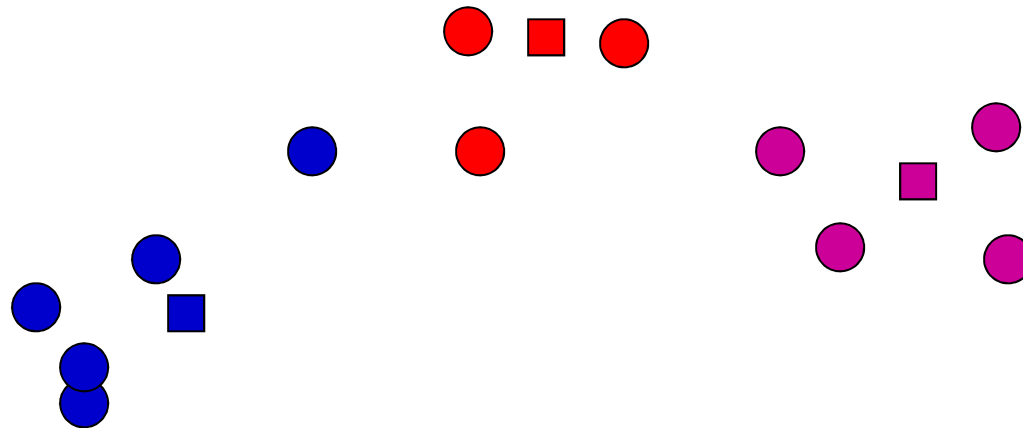
# K-means: readjust centers

---



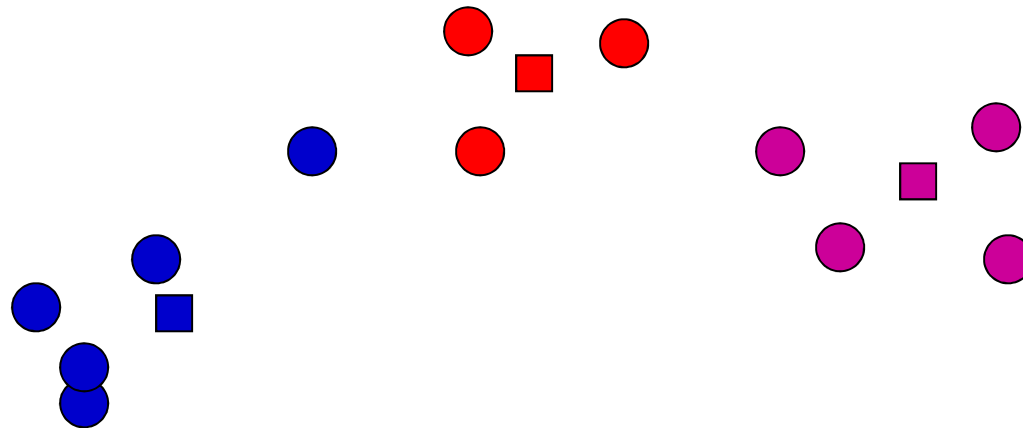
# K-means: assign points to nearest center

---



# K-means: readjust centers

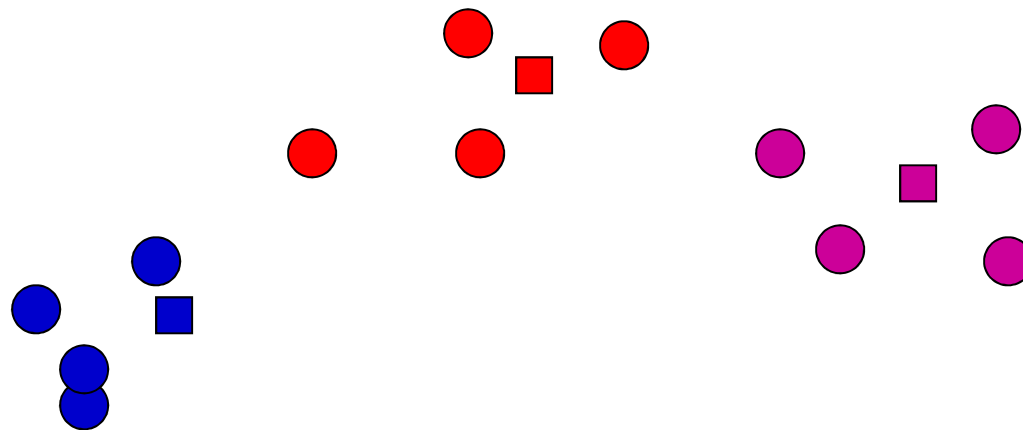
---





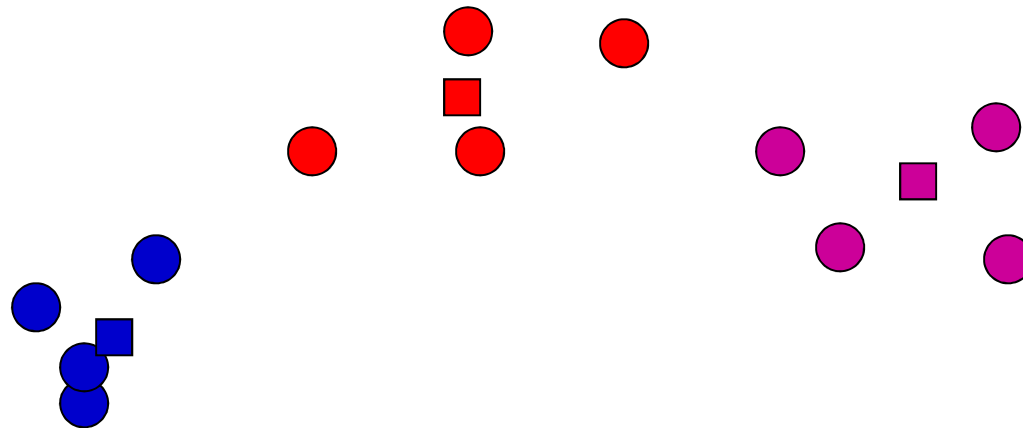
# K-means: assign points to nearest center

---



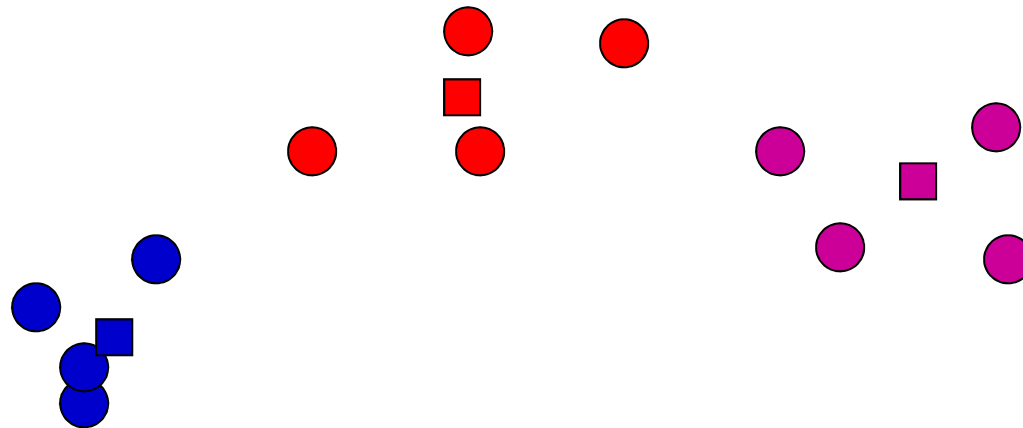
# K-means: readjust centers

---



# K-means: assign points to nearest center

---



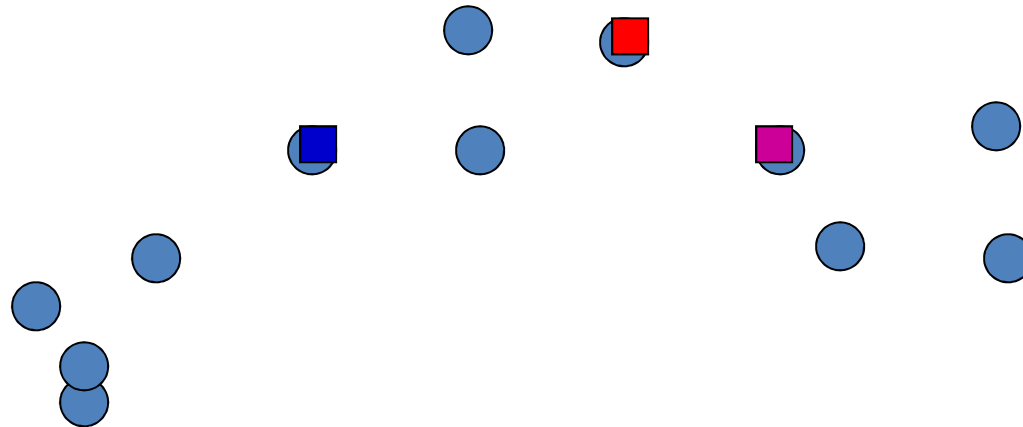
No changes: Done

# K-means

---

Iterate:

- **Assign/cluster each example to closest center**
- Recalculate centers as the mean of the points in a cluster



# K-means

---

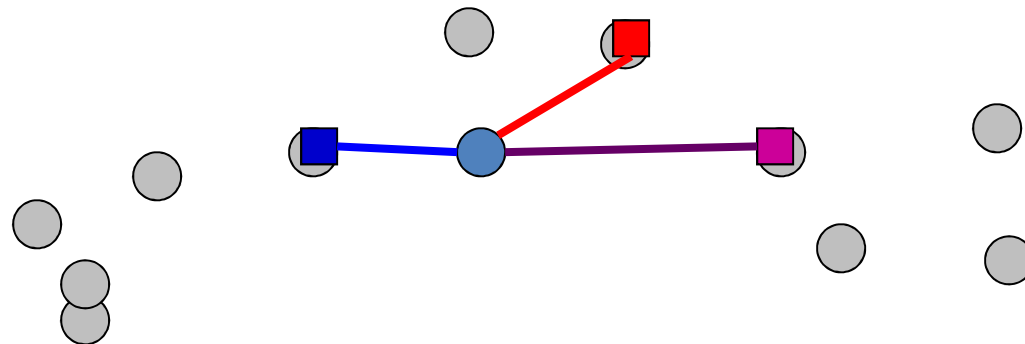
Iterate:

- **Assign/cluster each example to closest center**

iterate over each point:

- get distance to each cluster center
- assign to closest center (hard cluster)

- Recalculate centers as the mean of the points in a cluster



# K-means

---

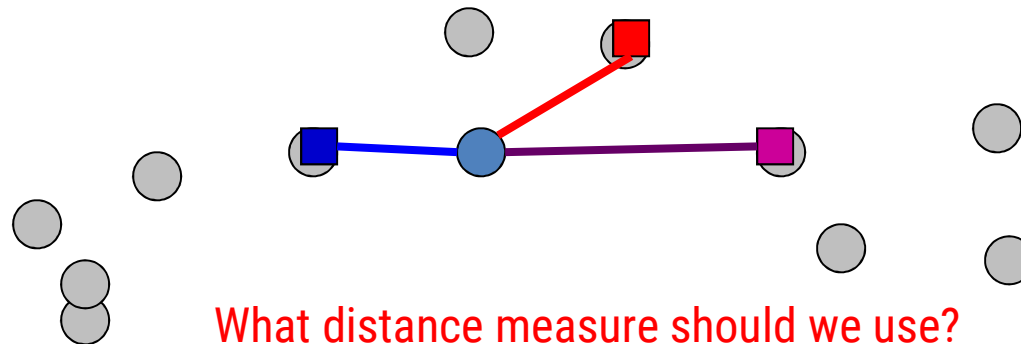
Iterate:

- **Assign/cluster each example to closest center**

iterate over each point:

- get **distance** to each cluster center
- assign to closest center (hard cluster)

- Recalculate centers as the mean of the points in a cluster



# Distance measures

---

Euclidean:

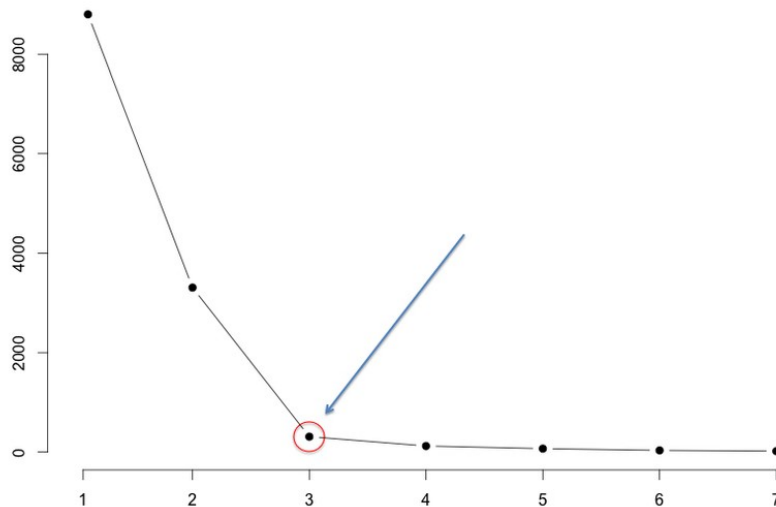
$$d(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

good for spatial data

# Choosing the Value of K

---

- **Choosing the Value of K**
  - We often know the value of K. In that case we use the value of K
  - Else we use the Elbow Method

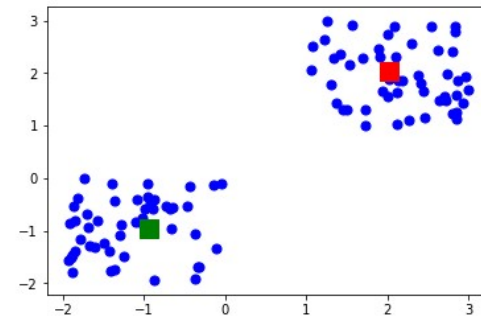




# Clustering using Python

## K-Means

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
%matplotlib inline
X = -2 * np.random.rand(100, 2)
X1 = 1 + 2 * np.random.rand(50, 2)
X[50:100, :] = X1
plt.scatter(X[:, 0], X[:, 1], s = 50, c = 'b')
plt.show()
```



# Clustering using Python

---

## K-Means

```
from sklearn.cluster import KMeans
Kmean = KMeans(n_clusters=2)
Kmean.fit(X)
Kmean.cluster_centers_
plt.scatter(X[:, 0], X[:, 1], s=50, c='b')
plt.scatter(-0.94665068, -0.97138368, s=200, c='g', marker='s')
plt.scatter(2.01559419, 2.02597093, s=200, c='r', marker='s')
plt.show()
Kmean.labels_
sample_test=np.array([-3.0,-3.0])
second_test=sample_test.reshape(1, -1)
Kmean.predict(second_test)
```

# Clustering using Python

---

## Hierarchical

```
from matplotlib import pyplot as plt
from scipy.cluster.hierarchy import dendrogram, linkage
import numpy as np
# generating two clusters: x with 10 points and y with 20:
np.random.seed(1234)
x = np.random.multivariate_normal([10, 0], [[3, 1], [1, 4]], size=[10,])
y = np.random.multivariate_normal([0, 20], [[3, 1], [1, 4]], size=[20,])
X = np.concatenate((x, y),)
print(X.shape) # 150 samples with 2 dimensions
plt.scatter(X[:,0], X[:,1])
plt.show()
```

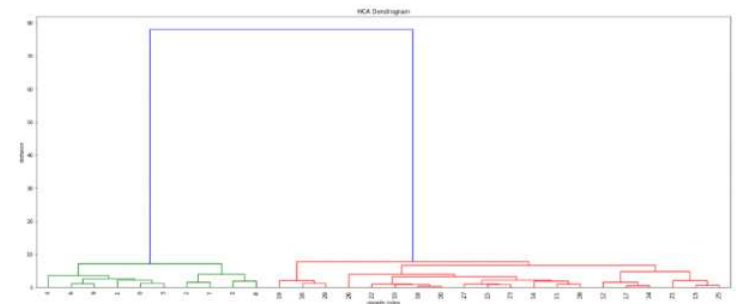
# Clustering using Python

## Hierarchical

```
# generate the linkage matrix
Z = linkage(X, 'ward')
#print(Z)

from scipy.cluster.hierarchy import cophenet
from scipy.spatial.distance import pdist
coph_dists = cophenet(Z, pdist(X))
#coph_dists

plt.figure(figsize=(25, 13))
plt.title('HCA Dendrogram')
plt.xlabel('sample index')
plt.ylabel('distance')
dendrogram(Z, leaf_rotation=90, leaf_font_size=12,)
plt.show()
```



# Clustering using Python

---

## Hierarchical

```
from sklearn.cluster import KMeans
Kmean = KMeans(n_clusters=2)
Kmean.fit(X)
Kmean.cluster_centers_
plt.scatter(X[:, 0], X[:, 1], s=50, c='b')
plt.scatter(-0.94665068, -0.97138368, s=200, c='g', marker='s')
plt.scatter(2.01559419, 2.02597093, s=200, c='r', marker='s')
plt.show()
Kmean.labels_
sample_test=np.array([-3.0,-3.0])
second_test=sample_test.reshape(1, -1)
Kmean.predict(second_test)
```

**Thank You**

**Digital Vidya**