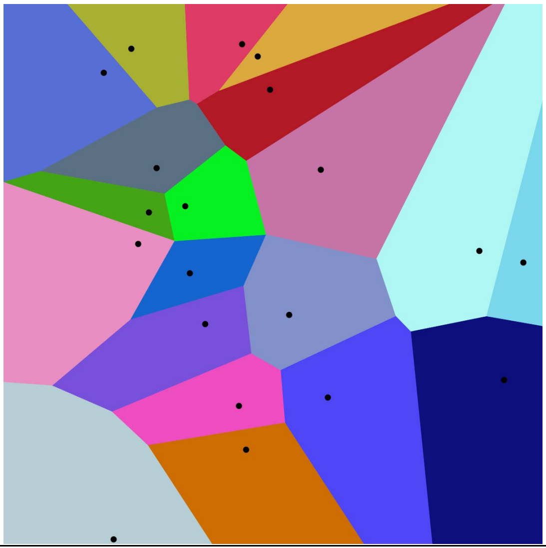


# Unit 2: Things to Consider when Choosing the 'Right' Clustering Algorithm

***Case Study*: How do we go about choosing the 'right' clustering algorithm? We'll look at several 2-d artificial datasets to showcase the various goals for using and choosing a clustering algorithm.**

## Purpose of this Lecture:

- **Questions**:
  - Provide a **road-map** for the different clustering algorithms we will learn in this class.
  - In this lecture we will introduce and **categorize different types of clustering algorithms.**
  - Specifically we will introduce the ifferent types of questions to consider when **choosing the 'right' clustering algorithm** for a given dataset, research question, or research goal.
  - Using **2-d datasets**, we will develop an **intuition** for:
    - why the k-means clustering algorithm performs effectively for certain types of datasets,
    - why the k-means clustering algorithm does not perform well for other types of datasets,
    - what other types of algorithms will work well for these types of datasets.
  - Become more acclimated with using Python.

## Summary of Concepts:

- Nature of The Data:
  - What **type of attributes** is this clustering algorithm designed for (ie. numerical attributes? categorical attributes? a mixture of both?)
- "Unclean" Data Considerations:
  - Will the algorithm **perfom effectively** when the data has **noise** and/or **outliers**?
  - What will the algorithm **do with a noisy object**?
    - Identify it as noise?
    - Put it in a cluster with other elements?
    - Make it it's own cluster?
  - What will the algorithm **do with an outlier**?
    - Identify it as an outlier?
    - Put it in a cluster with other elements?
    - Make it it's own cluster?
  - *How do we know if a dataset with 4 or more attributes has noise or outliers?*

- Definition of a "Cluster"
    - If there is some "clustering" structure in the dataset, what is the best way to **define the nature of a given "cluster"**?
        - Well-Separated cluster
        - Prototype-based cluster
        - Graph-based cluster
        - Density-based cluster
        - Shared-property (conceptual) cluster
    - *How do we know what the clusters look like (ie. what they should be defined by) in a dataset with 4 or more attributes*?
- Types of Clustering Results
    - What would we like our clustering result to tell us about the "grouping" nature of the data?
        - **Partition Results vs. Hierarchical Results**
            - Partition Results: Do we want just one grouping?
            - Hierarchical Results: Are our clusters "nested"? Are some clusters closer to other clusters? Do we want to see this "nested" cluster nature in our results?
        - **Exclusive vs. Overlapping vs. Fuzzy**
            - Exclusive Clustering Result: Do we want our objects to belong to just one cluster?
            - Overlapping Clustering Result: Can objects in our clustering belong to more than one cluster
            - Fuzzy Clustering Result: Instead of assigning an object to a cluster(s), what if wanted to attain a cluster membership score for each object i and each cluster j (ie. $score(x_{ij}) \in [0,1]$, represents the "percent" to which object i belongs to cluster j).
- Types of Algorithm Performance Measures to Consider
    - (will discuss in future lecture)

---

Review of Concepts from Lecture 02

08/27/20

# Recap of What We Learned

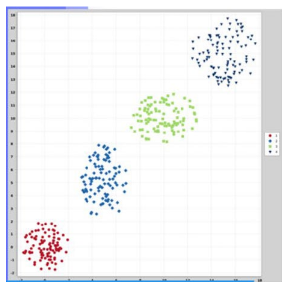
- **More about k-means algorithm**
  - When k-means will work well.
  - How k-means performs under certain types of data.
  - Properties of k-means clustering algorithm results.

- **Definitions of a Cluster**
  - If there is some "clustering" structure in the dataset, what is the best way to **define the nature of a given "cluster"**?
    - Well-Separated cluster
    - Prototype-based cluster
    - Graph-based cluster
    - Density-based cluster
    - Shared-property (conceptual) cluster

- **Types of Clustering Results**
  - What would we like our clustering result to tell us about the "grouping" nature of the data?
    - Partition Results vs. Hierarchical Results
    - Exclusive vs. Overlapping vs. Fuzzy

- **New Algorithm**:
  - **K-medoids clustering**

# When the k-means algorithm works well



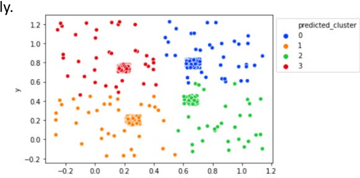
- The data is:
  - All numerical variables
  - Doesn't have noise or outliers

- The clusters are:
  - Spherical (globular)
  - Well-separated
  - Have the same size (ie. number of objects in them)
  - Have the same sparsity

- The algorithm:
  - Uses the "inherent" number of clusters in the data (ie. "right" k).

# How does **k-means** perform with **noisy data**?

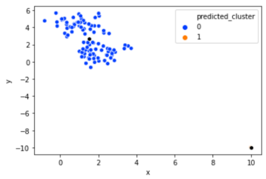
- With noisy data k-means will do the following.
  - Performance for identifying the *actual* clusters may be affected.
    - It *could* not identify all the clusters.
    - It *could* split the clusters.
    - Or it *could* identify the clusters appropriately.
  - Noise objects in the data will:
    - Be place in a cluster.
    - NOT be explicitly identified as noise.



# How does **k-means** perform with **outliers**?

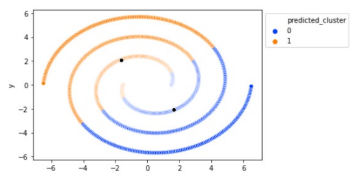
- When a dataset has outliers k-means will do the following.
  - Performance for identifying the *actual* clusters may be affected.
    - It *could* not identify all the clusters.
    - It *could* split the clusters.
    - Or it *could* identify the clusters appropriately.
  - Outliers in the data will:
    - Not be *explicitly* identified.
    - The prototypes (centroids) may not necessarily identified as clusters.

**Question:** What are some ways you might try to identify if some objects are outliers *after* k-means clustering has taken place?



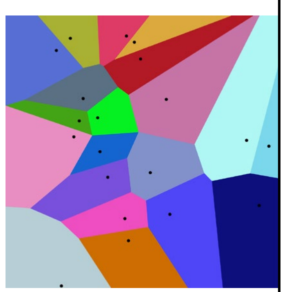

# How does **k-means** perform with **non-spherical shaped clusters**?

- When a dataset has clusters that are non-spherical (ie. non-globular), then k-means will do the following.
  - Performance for identifying the *actual* clusters *may* be affected.
    - It *could* not identify all the clusters.
    - It *could* split the clusters.
    - Or it *could* identify the clusters appropriately.



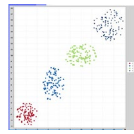

# After running k-means we can **cluster new data** by finding the closest centroid to that new object and assigning it to this cluster.

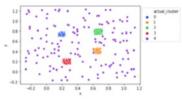
- This post-hoc assignment splits up the space of data into a **voronoi diagram.**
  - This diagram partitions the plane that the data resid es in into a set of regions bounded by intersecting lin e segments. These line segments represent the point s in the plane that are equidistant to the two nearest centroids.

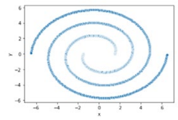


# Definitions of a Cluster

- If there is some "clustering" structure in the dataset, what is the best way to **define the nature of a given "cluster"**?
  - **Well-Separated cluster** defines a cluster only when the data contains natural clusters that are quite far away from one another.
  - **Prototype-based cluster** defines a cluster as a set of objects in which each object is closer (or more similar) to the prototype that defines the cluster than to the prototype of any other cluster.
    - Types of prototype-based clustering algorithms:
      - **Ex**: **k-means** (mean is the prototype)
      - **Ex**: **k-medoids** (medoid is the prototype)
  - **Density-based cluster** defines a cluster a dense region of objects that is surrounded by a region of low density.
    - Types of density-based clustering algorithms:
      - **Ex**: **DBSCAN** (identifies noise objects as noise)
  - **Graph-based cluster** a group of objects that are connected to one another, but have no connection to objects outside the group.
    - **Contiguity-based cluster** (a type of graph-based cluster definition) two objects are connected only if they are within a specified distance of one another.
    - Types of contiguity-based clustering algorithms:
      - **Ex**: **Agglomerative Hierarchical Clustering Algorithms (single linkage, complete linkage, average linkage, Ward's linkage)**
  - **Shared-property (conceptual) cluster** defines a cluster as a set of objects that share some property.







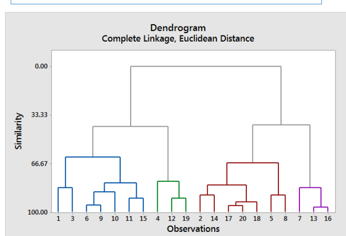
# Types of Clustering Results

- Partitional vs. Hierarchical Clustering

**Partitional Clustering**
a division of the set of data objects into non-overlapping subsets (clusters) such that each obje ct is in exactly one subset.

**Hierarchical Clustering**
we allow for clusters to have sub-clusters. A hierarchical clustering is displayed as a set of nested clusters displayed as a tree.

Ex: Clustering =
*Cluster 1*: {1, 3, 6, 9, 10, 11, 15}
*Cluster 2*: {4, 12, 19}
*Cluster 3*: {2, 14, 17, 20, 18, 5, 8}
*Cluster 4*: {7, 13, 16}




# Types of Clustering Results

- Exclusive vs. Overlapping vs. Fuzzy Clustering Results
  - **Exclusive Clustering** will assign an object to a single cluster.
  - **Overlapping Clustering** *can* allow for an object to be assigned to more than one cluster.
  - In a **Fuzzy Clustering** every object belongs to every cluster with a membership weight that is between 0 (absolutely doesn't belong) to 1 (absolutely belongs).
    - Usually the sum of each objects weights must sum to 1.
    - $w_{ij}$ =the probability that object i belongs to cluster j

Ex: Exclusive Clustering =
*Cluster 1*: {1, 3, 5}
*Cluster 2*: {2, 4}

Ex: Overlapping Clustering =
*Cluster 1*: {1, 3, 5}
*Cluster 2*: {2, 4, 5}

Ex: Fuzzy Clustering =

| | Cluster 1 Membership Weight | Cluster 2 Membership Weight |
|---|---|---|
| Object 1 | 0.33 | 0.67 |
| Object 2 | 0.5 | 0.5 |
| Object 3 | 1 | 0 |
| Object 4 | 0 | 1 |
| Object 5 | 0.25 | 0.75 |

# K-Medoids Clustering

1. Select initial medoids randomly.
2. While the cost decreases
   a) Set a new medoid:
      - For each cluster, set medoid = the **point** in the cluster that minimizes the sum of the distances within the cluster to the medoid.
   
   b) Reassign clusters:
      - Assign each point to it's closest **medoid.**

**Additional Information:**

- This is a prototype-based clustering algorithm.
  - The medoid is the prototype.
  - The medoid is always an actual point (as opposed to the centroid in k-means)
- Guaranteed to converge to a local minimum.
- Performs better than k-means algorithm when there are outliers.