09/27 CS506 Notes
Hierarchical Clustering

- Output: show which clusters were merged in order to produce a dendrogram
- Cut the dendrogram at any point and then we have separated clusters.
  Give us a good picture of how data relates to each other

Two main types of hierarchical clustering:
Agglomerative: ***(main focus)***

1. Start with every point in its own cluster
2. Each step, merge the two closest clusters
3. Stop when every point is in the same cluster

Divisive:
…

Agglomerative algorithm:

1. Let each point be its own cluster
2. Compute the distance between all pairs of clusters
3. Merge the two closest clusters
4.repeat 3 and 4 until all points are in the same cluster

Any other ideas???
Compute the distance between the center of the clusters??
…. the means of the clusters???
Take the shortest paths???

Distance functions:

- **Single-link distance**
  The minimum of all pairwise distances between a point from one cluster and a point from the other cluster
  Take the minimum of the set

  But…. sensitive to noise points
  Tends to create elongated clusters

- **Complete-link distance**
  Maximum of all pairwise distances

  Less susceptible to noise
  But… tends to split up

- **Average-link distance**

The average of all pairwise distances

Less susceptible to noise and outliers
But… tends to be biased toward globular clusters

- **Ward's distance**
  Is the difference between the spread / variance of points in the merged cluster and the unmerged clusters
  Distance to the mean, proportional to the ring
  Close?? Then similar in spread

  Can ward's distance be negative??? Question for next time

Hierarchical clustering

- Used to expose a hierarchy in the data.

Density-based clustering

- Goal: cluster together points that are densely packed together
- Density definition???

  It helps to distinguish between points at the core of a dense region and points at the border of a dense region

  Core point, border point, noise point

  Create clusters by connecting core points
  Scan all core and border points into cluster, attach noise point to the closest cluster

DBScan algorithm:

- 2d parameters
- Epi and min-pts given:
  - Find the epi-neighborhood of each point
  - Label core
  - Label border
  - Label noise
  - Assign core points
  - Assign border points
  - Attach noise points
- Non-core point won't be in the neighborhood
- If one point is a border of a core, and it does not have enough data points around it, then it is a noise

DBScan - benefits:

- Can identify clusters of different shapes and sizes

- Resistant to noise

DBScan - limits:

- Can fail to identify clusters of varying densities
- Tends to create clusters of the same density
- Notion of density is problematic in high-dimensional spaces

**** every point treats as noises at the beginning, real noises will stay as noises till the end*

DBScan demo:

```
def dbscan(self):
    assignments = [0 for _ in range(len(self.dataset))]
    assignment = 1

    for i, x in enumerate(self.dataset):
        if epsilonNeighborhood(x) >= self.min_pts:
            # core point
            # explore x neighborhood for core points and give them the same as
            assignment = self._explore_neighborhood(x, assignments, assignment)
```