8) Clustering

- Part of unsupervised learning, meaning we have data points with unknown outcomes (labels)
- Some common use cases of clustering are:
  - Classification
  - Anomaly/outlier detection
  - Customer segregation
  - Improve supervised learning models. For example, by training different models for clusters rather than training a model for the whole data

Distance metrics

- "Distance" between two points can be measured in various ways
- Choice of our distance metric is incredibly important during clustering
- Each metric has its strengths, weaknesses, and use cases

Manhattan / city block (L1)

The concept here is that "you cannot walk between two points directly, you have to turn at a corner first"

$|\Delta x| + |\Delta y|$

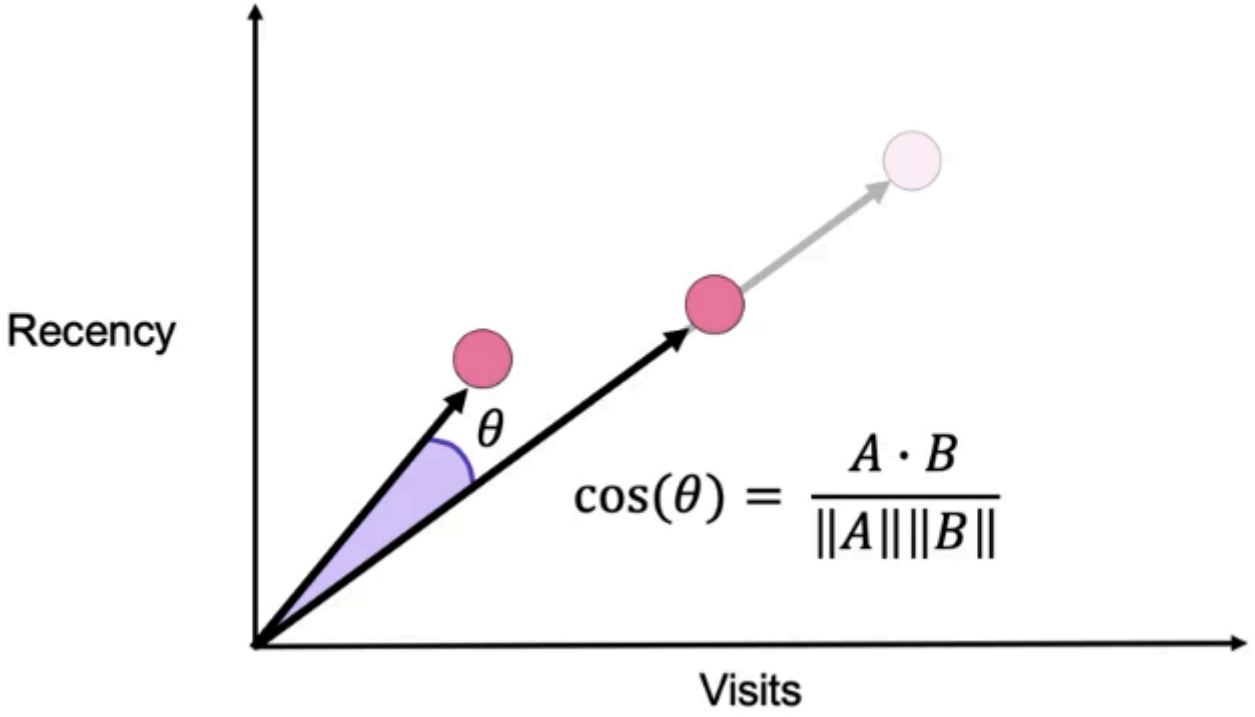
Euclidean (L2)

The most common and easy to understand method, basically how we were taught in school to calculate distance between two points. It works on the pythagorus theorem and calculates the literal distances between the points. It is useful for coordinate based measurements. It is more sensitive (compared to cosine) to the curse of dimensionality

$\text{sqrt}(\Delta x^2 + \Delta y^2)$

Cosine

Here we draw lines/vectors of the two points and take the cosine of the angle between them. It remains insensitive to the scaling in respect to the origin. Meaning the distance will remain same even if a point is moved back/forward on the line. It is better for data such as text where the location of occurrence in less important. It is more robust/insensitive (compared to euclidean) to the curse of dimensionality



Jaccard

Useful for sets (like word occurrences)

- **Sentence A**: “I like chocolate ice cream.”
- set A = {I, like, chocolate, ice, cream}
- **Sentence B**: “Do I want chocolate cream or vanilla cream?”
- set B = {Do, I, want, chocolate, cream, or, vanilla}

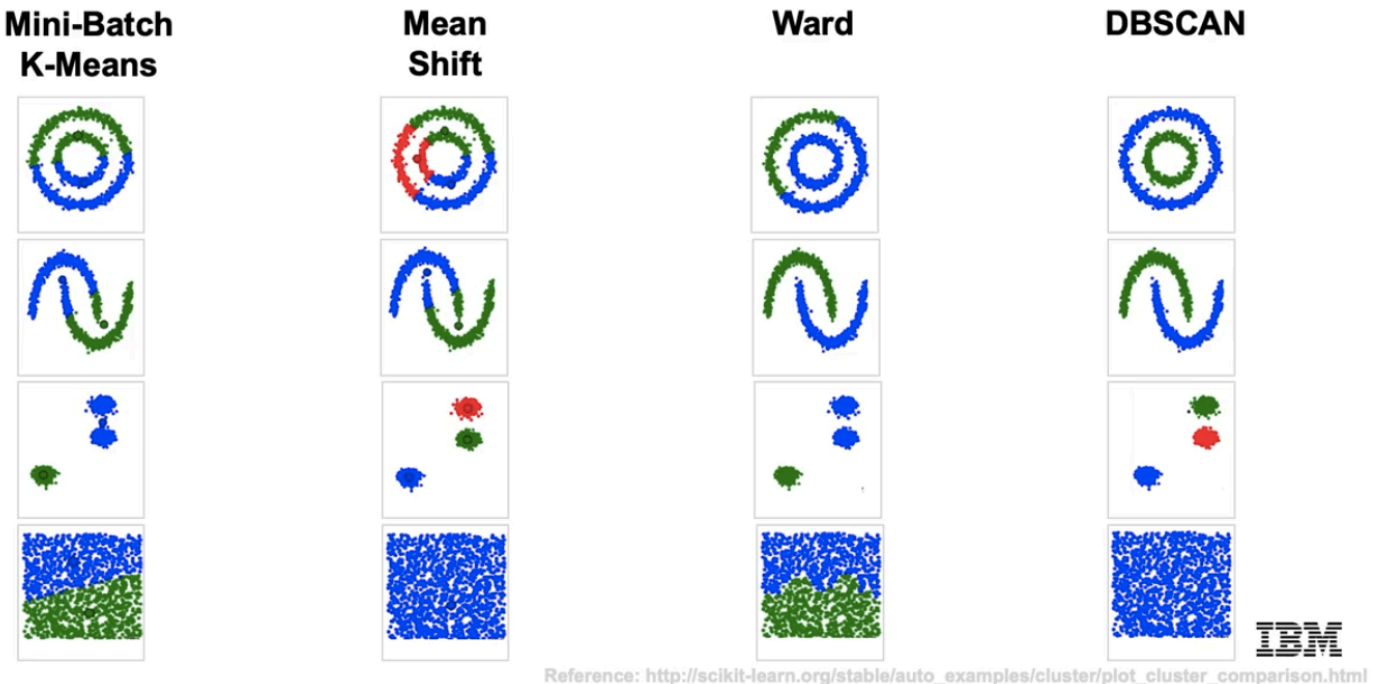
$$1 - \frac{A \cap B}{A \cup B} = 1 - \frac{len(shared)}{len(unique)}$$

- **Sentence A**: “I like chocolate ice cream.”
- set A = {I, like, **chocolate**, ice, **cream**}
- **Sentence B**: “Do I want chocolate cream or vanilla cream?”
- set B = {Do, I, want, **chocolate**, **cream**, or, vanilla}

$$1 - \frac{A \cap B}{A \cup B} = 1 - \frac{3}{9}$$

Comparisons

("Ward" means the HAC algo with ward linkage)



A summary of approaches to clustering, along with general use cases and applications.

Method name	*K-means*	*Mean-shift*	*Hierarchical clustering*	*DBSCAN*
Parameters	Number of clusters	Bandwidth	Number of clusters	Neighborhood size
Scalability	Very large *n_samples* medium *n_clusters* with *MiniBatch code*	Not scalable with n_samples	Large *n_samples* and *n_clusters*	Very large *n_samples*, medium *n_clusters*
General use case	General purpose, even cluster size, flat geometry, not too many clusters	Many clusters, uneven cluster size, non-flat geometry	Many clusters, possibly connectivity constraints	Non-flat geometry, uneven cluster sizes, outlier detection
Applications	Find few clusters of roughly the same size	Can identify number of clusters, often used in video	Clusters may be of different size, does not identify outliers	Often used in computer vision applications

From https://scikit-learn.org/stable/modules/clustering.html#overview-of-clustering-methods

---

1) K-Means

- MiniBatch is fast
- Gotta finetune the `n_clu` hyperparameter (can use that elbow method)
- Tends to find even sized clusters
- Bad with non-spherical cluster shapes

2) Mean shift

- We dont have to guess `n_clu`
- Can find uneven clusters
- Slow with a lot of data
- Does a good job of finding a lot of clusters (if they exist)
- Does not do a great job of finding weird shapes
- Uses euclidean distances only

3) HAC (ward)

- Useful for getting a "full hierarchal tree", meaning how some groups are maybe sub groups of others
- Gotta finetune the `n_clu` hyperparameter
- Can find uneven cluster sizes
- A lot of distance metrics and linkage options (may make it difficult to finetune)
- Can be slow to calculate (as number of observations increase)

4) DBSCAN

- "Can get the best of both worlds, if the right parameters are chosen"

- Gotta finetune the `epsilon` and `n_clu` hyperparameters
- Can find uneven sized clusters
- Various distance metrics
- Can handle tons of data and weird shapes
- Too small epsilon is not trustworthy (as it will result in too many clusters)
- Does not do well with clusters of different densities