k-means limitation:

~ Relies on random initialization and so the outcome may change depending upon this initialization.

~ K-Means clustering requires to specify the number of clusters in advance.

~ Very often you do not know the centers in advance. The elbow method or the silhouette method to find the optimal number of clusters are not always easy to interpret.

~ Each point has to have a cluster assignment.

~ The boundaries between clusters are linear; It fails to identify clusters with complex shapes. Because K-Means partitions the space based on the closest mean. Each cluster is defined solely by its center and so it can only capture relatively simple shapes.

Failure 1Chart, scatter chart

Description automatically generated

Failure 2

Chart, scatter chart

Description automatically generated

Failure 3

Chart, scatter chart

Description automatically generated

15.1 DBSCAN Motivation

k-means's decision boundary is linear, it cannot identify complex cluster shapes.

15.2 DBSCAN - density based clustering

two hyper params

1.**eps**: determines what it means for points to be close (low eps = more noise points, higher eps = more accepting)

2.**min\_samples**: determines the number of neighbouring points we require to consider in order for a point to be part of a cluster.

~ we cannot use **elbow method** to examine the goodness of clusters created with DBSCAN

~ but **silhouette** method could

K-Means vs. DBSCAN

* In DBSCAN, you do not have to specify the number of clusters!
  + Instead, you have to tune eps and min\_samples.
* Unlike K-Means, DBSCAN doesn’t have to assign all points to clusters.
  + The label is -1 if a point is unassigned.
* **Unlike K-Means, there is no predict method.**
  + **DBSCAN only really clusters the points you have, not “new” or “test” points.**

There are three kinds of points.

* **Core points** are the points that have at least min\_samples points in the neighborhood.
* **Border points** are the points with fewer than min\_samples points in the neighborhood, but are connected to a core point.
* **Noise points** are the points which do not belong to any cluster. In other words, the points which have less that min\_samples point within distance eps of the starting point are noise points.
* We cannot use the elbow method to examine the goodness of clusters created with DBSCAN.
* But we can use the silhouette method because it’s not dependent on the idea of cluster centers.
* **Pros**
  + **Can learn arbitrary cluster shapes**
  + **Can detect outliers**
* **Cons**
  + **Cannot predict on new examples.**
  + **Needs tuning of two non-obvious hyperparameters**

### DBSCAN: failure cases[¶](https://ubc-cs.github.io/cpsc330/lectures/15_recommender-systems.html#dbscan-failure-cases)

* DBSCAN is able to capture complex clusters. But this doesn’t mean that DBSCAN always works better. It has its own problems!
* DBSCAN doesn’t do well when we have clusters with different densities.
  + You can play with the hyperparameters but it’s not likely to help much.

Chart, scatter chart

Description automatically generated

Recommender system

* Most often the data for recommender systems come in as **ratings** for a set of items from a set of users.
* We have two entities: N **users** and M **items**.
* **Users** are consumers.
* **Items** are the products or services offered.
  + E.g., movies (Netflix), books (Amazon), songs (spotify), people (tinder)

### Utility matrix

* Below is a toy utility matrix. Here N = 6 and M = 5.
* Each entry yij (ith row and jth column) denotes the rating given by the user i to item j.
* We represent users in terms of items and items in terms of users.
* Given a utility matrix of N users and M items, **complete the utility matrix**. In other words, **predict missing values in the matrix**.

Baselines

Let’s first try some simple approaches to predict missing entries.

1. Global average baseline:

In this baseline we predict everything as the global average rating

Graphical user interface, text, application, email

Description automatically generated

1. [k-Nearest Neighbours imputation](https://scikit-learn.org/stable/modules/generated/sklearn.impute.KNNImputer.html)

Impute missing values using the mean value from k nearest neighbours found in the training set.

Calculate distances between examples using features where neither value is missing

Better approach:

1. Collaborative Filtering: Unsupervised approach: Only uses the user-item interactions given in the ratings matrix.

**Intuition**

We may have similar users and similar items which can help us predict missing entries.

Leverage social information to provide recommendations.

The collaborative filtering algorithm we use in this package is called SVD.

### content-based filtering

* Supervised machine learning approach
* In collaborative filtering we assumed that we only have ratings data.
* Usually there is some information on items and users available.
* Examples
  + Netflix can describe movies as action, romance, comedy, documentaries.
  + Amazon could describe books according to topics: math, languages, history.
  + Tinder could describe people according to age, location, employment.
* Can we use this information to predict ratings in the utility matrix?
  + Yes!

1. Hybrid filtering: combination of 3 and 4