IT496: Introduction to Data Mining

Lecture 35-36

Clustering Analysis

Arpit Rana

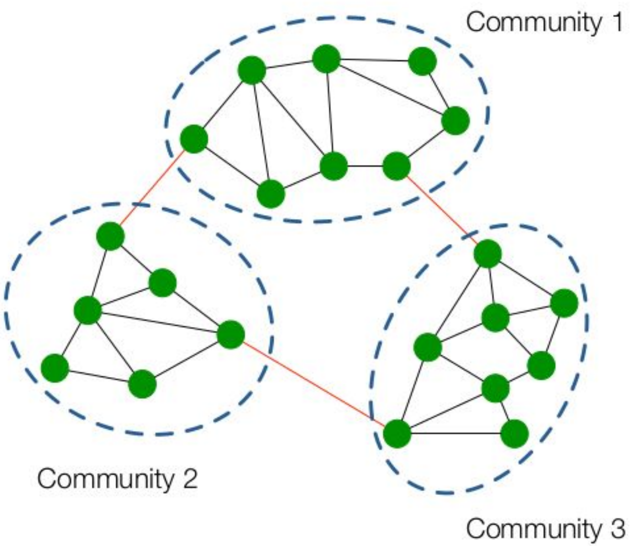
21st / 23rd November 2023

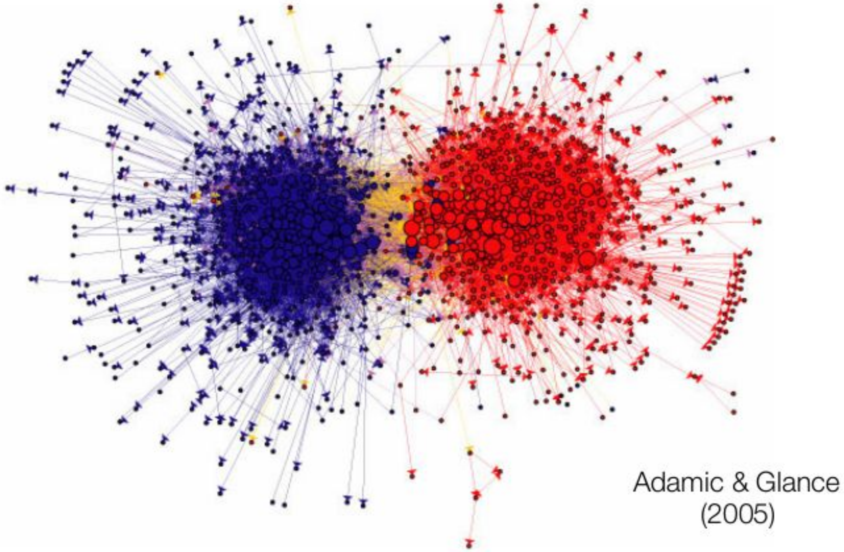
Applications of Clustering

Market segmentation: Unsupervised task that attempts to automatically grouping customers into separate clusters, so that customers in the same cluster have similar needs and respond similarly to a marketing action.

Applications of Clustering

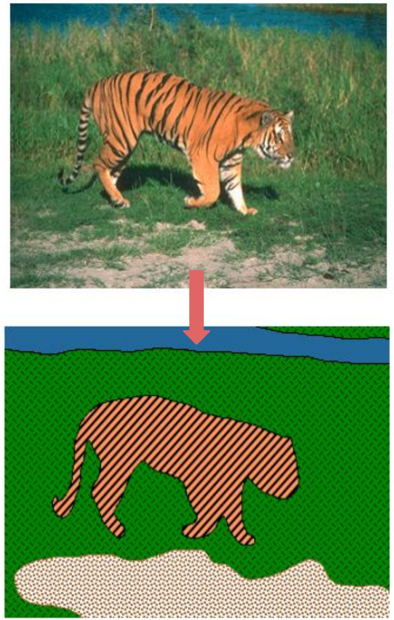
Community Detection: Given a social network, apply clustering to identify communities of users who are well-connected to one another, and who are separated from other communities.

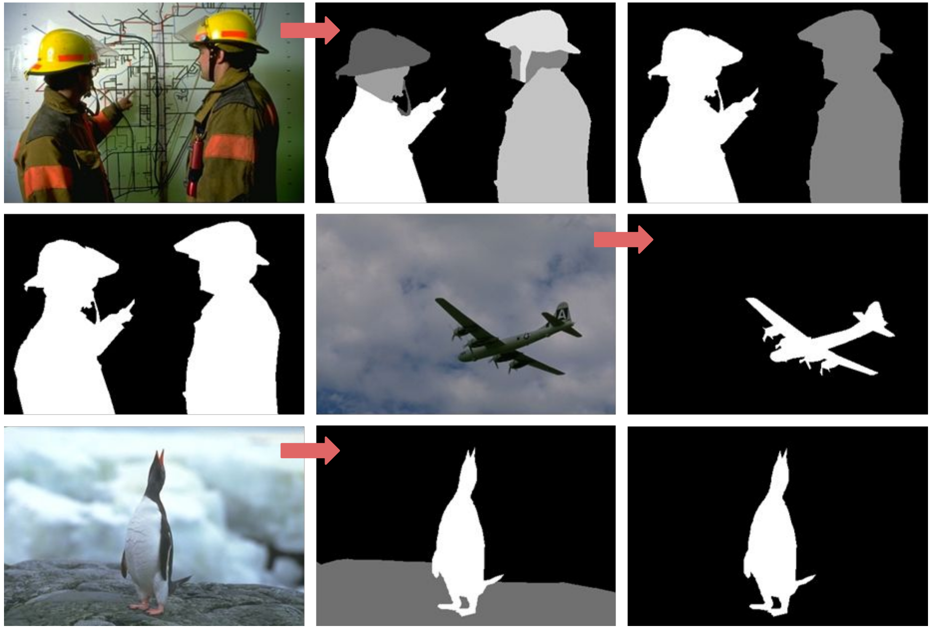




Applications of Clustering

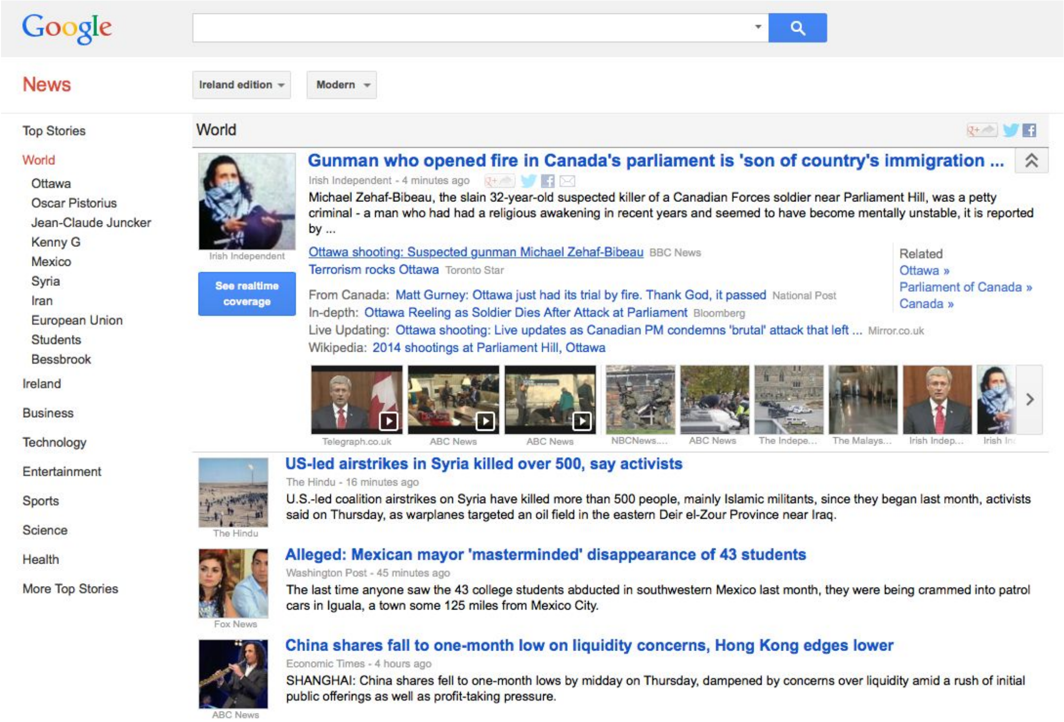
Image segmentation: Unsupervised task in computer vision that attempts to automatically split an image into regions with similar *colour* or *texture*, or *both*. Aim is to partition the image into its constituent "objects".





Applications of Clustering

Document Clustering: Automatically group related documents together based on similar content (e.g. related articles on Google News).



Applications of Clustering

Topic modeling: Unsupervised task of discovering the underlying thematic structure in a text corpus - i.e. the key “topics” in the data.

Clustering

Grouping examples in the absence of any external information is called *Clustering*.

- No labelled training examples to learn from.
- Generally we will not know in advance how many clusters are present in the data.

Clustering

Clusters are inferred from the data such that -

- Examples within a cluster should be similar.
- Examples from different clusters should be dissimilar.

Secondary goals in clustering

- Avoid very small and very large clusters
- Define clusters that are easy to explain to the user
- Many others . . .

Clustering

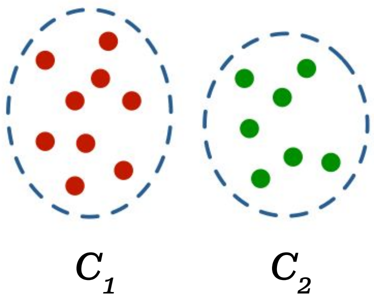
Clusters are inferred from the data without human input.

- However, there are many ways of influencing the outcome of clustering:
  - number of clusters,
  - similarity measure,
  - representation of examples (e.g., documents),
  - . . .

Clustering: Types

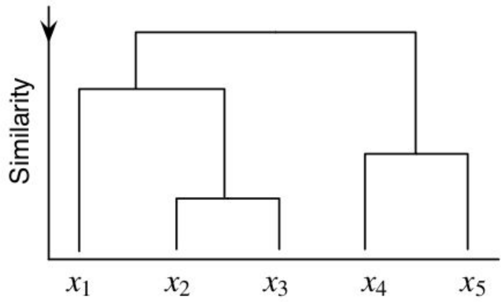
Flat algorithms (Partitioning)

- Usually start with a random (partial) partitioning of examples into groups
- Refine iteratively
- Main algorithm: *K-means*



Hierarchical algorithms

- Create a hierarchy
- Bottom-up, *agglomerative*
- Top-down, *divisive*



Clustering: Types

Hard clustering

- Each example in exactly one cluster.
    - More common and easier to do

Soft clustering

- An example can be in more than one cluster.
    - Makes more sense for browsable hierarchies
    - You may want to put sneakers in two clusters:
        - Sports apparel
        - Shoes
- You can only do that with a soft clustering approach.

We will do *flat, hard clustering* only in this course.

Clustering

Flat algorithms compute a partition of N examples into a set of K clusters.

- Given: a set of examples and the number K
- Find: a partition into K clusters that optimizes the chosen partitioning criterion
- Global optimization: exhaustively enumerate partitions, pick optimal one
  - Not tractable
- Effective heuristic method: K-*means* algorithm

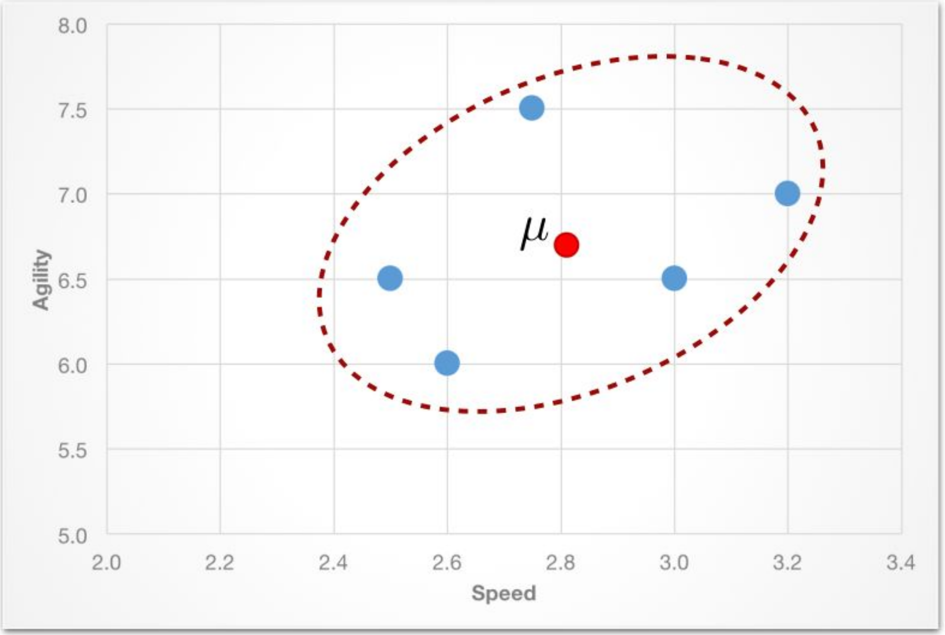
Clustering: What is Centroid?

Centroid: The mean vector of all items assigned to a given cluster (i.e. the mean of their feature vectors).

Athlete	*Speed*	*Agility*
1	2.6	6.0
2	3.0	6.5
3	2.5	6.5
4	3.2	7.0
5	2.8	7.5
Centroid	2.82	6.7

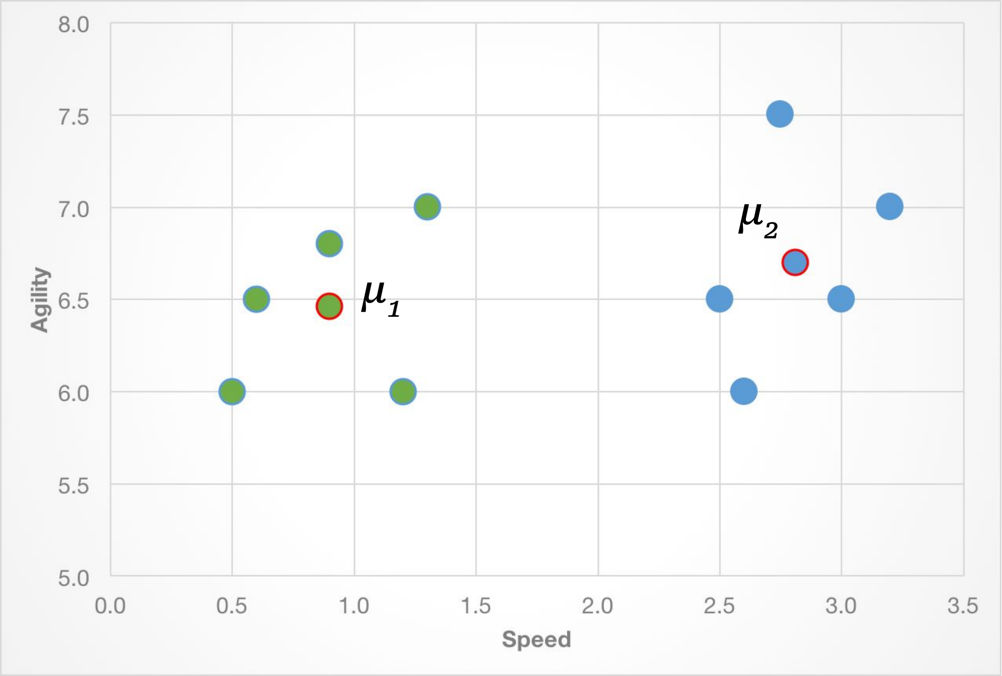
$(2.6 + 3.0 + 2.5 + 3.2 + 2.8)/5$
$= 2.82$

$(6.0 + 6.5 + 6.5 + 7.0 + 7.5)/5$
$= 6.7$



Clustering: Cluster Assignment based on Centroid

Each of the k clusters in a clustering can be represented by its own centroid μ_i. Example of two clusters, with centroids shown:



Clustering: Assignment based Clustering

Given a set X of data points, we want a set C of k centers $\{\mu_1, \mu_2, ..., \mu_k\}$ which minimises some cost function -

$$minimize \sum_{x \in X} d(x, C)^2$$

Here, $$d(x, C) = \min_{\mu_i \in C} d(x, \mu_i)$$

Often $d()$ is the Euclidean function $$d(x, \mu) = \sqrt{\sum_{j=1}^{m} (x_j - \mu_j)^2}$$

sum of squared difference over all m feature values

Clustering: Assignment based Clustering

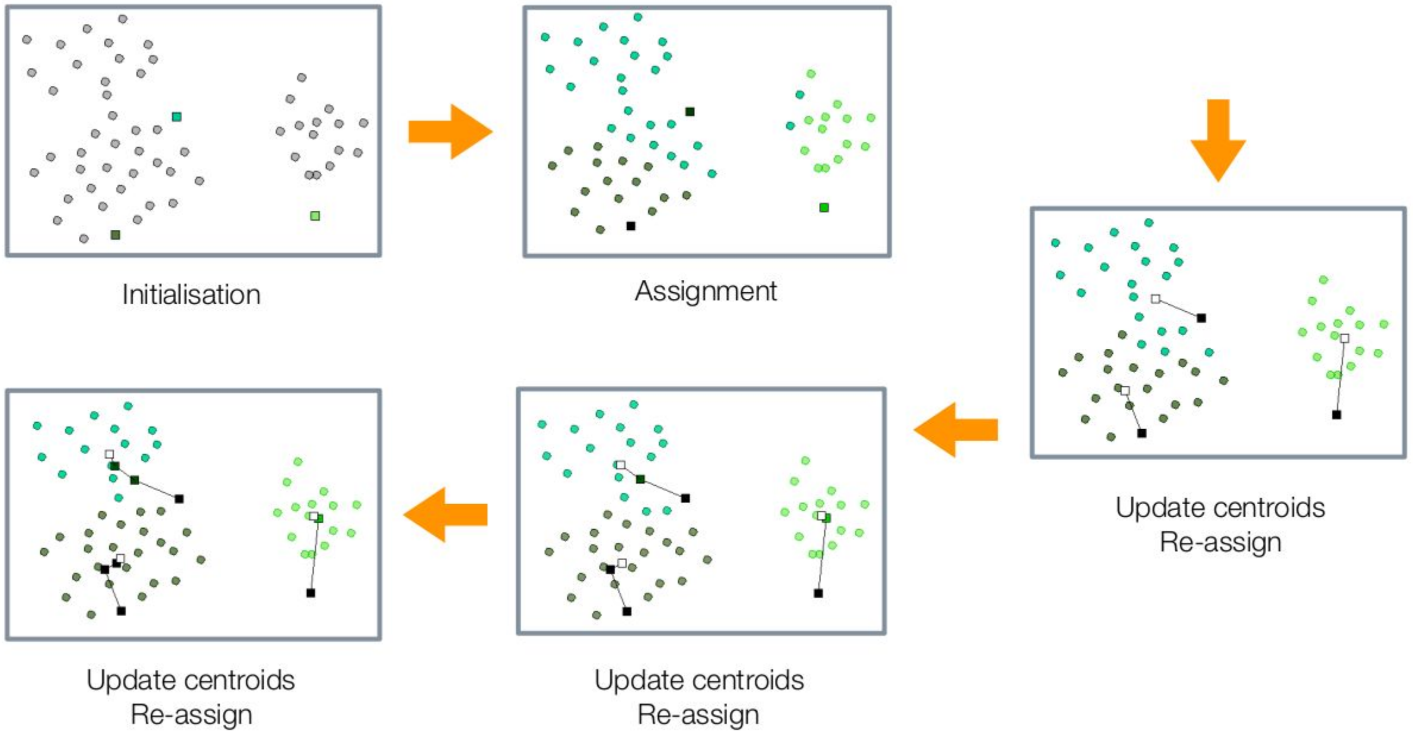
$$minimize \sum_{x \in X} d(x, C)^2 \quad with \quad d(x, C) = \min_{\mu_i \in C} d(x, \mu_i) \quad and \quad d(x, \mu) = \sqrt{\sum_{j=1}^{m} (x_j - \mu_j)^2}$$

- Minimising the k-Means objective is *NP-hard*
- Finding the optimal solution requires considering all K^n possible ways of assigning n data points to K clusters.
    - Each data point can be assigned to any one of the k clusters, leading to an exponential number of possible combinations.

Clustering: Assignment based Clustering

- ***Lloyd's* algorithm** is often used as a heuristic to minimise it
  - Reduce Sum of Squared Error (*SSE)* via a two step iterative process:
    - ***Reassign*** items to their nearest cluster centroid
    - ***Update*** the centroids based on the new assignments
  - Repeatedly apply these two steps until the algorithm converges to a final result

Clustering: Lloyd’s Algorithm



Clustering: Lloyd's Algorithm for K-means

Inputs

- Data: Set of unlabelled items
- k: User-specified target number of clusters
- Maximum number of iterations to run

Algorithm Steps

- Initialisation: Select k initial cluster centroids (e.g. at random)
- Assignment step: Assign every item to its nearest cluster centroid (e.g. using Euclidean distance).
- Update step: Recompute the centroids of the clusters based on the new cluster assignments, where a centroid is the mean point of its cluster.
- Go back to Step 2, until when no reassignments occur (or until a maximum number of iterations is reached).

Clustering: Lloyd's Algorithm for K-means

Lloyd's algorithm converges as the cost decreases monotonically.

- It may not converge in polynomial time -- there are examples where the algorithm takes exponentially many steps.
- The algorithm works well in practice.
- We often stop after a pre-defined number of iterations.
- No guarantee on the cost of the solution.

Clustering: Lloyd's Algorithm for K-means

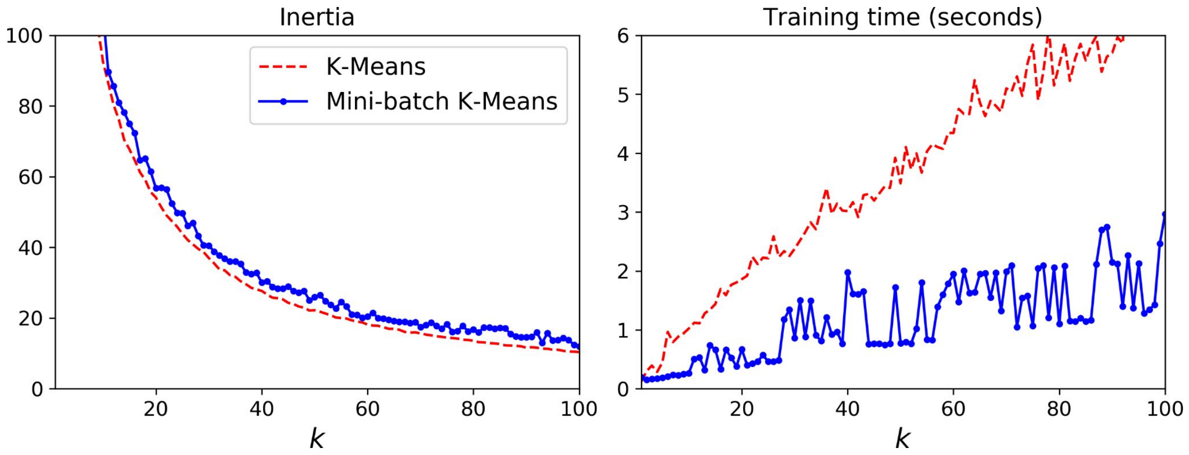
Complexity Analysis

- Computing a distance of two vectors is $O(M)$.
- Assignment step: $O(KNM)$ (we need to compute KN example-centroid distances)
- Update step: $O(NM)$ (we need to add each of the example's < M values to one of the centroids)
- Assume number of iterations bounded by I
  - Overall complexity: $O(IKNM)$ – linear in all important dimensions
- However: This is not a real worst-case analysis.

Mini-Batch K-means

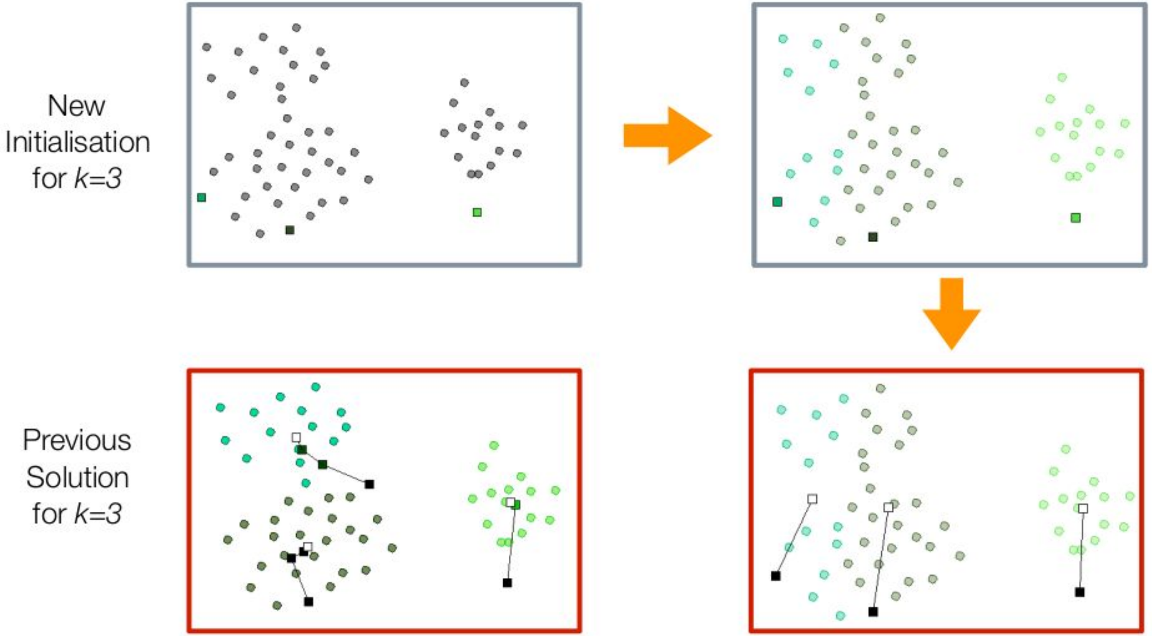
Instead of using the full dataset at each iteration,

- the algorithm is capable of using mini-batches, moving the centroids just slightly at each iteration.
- This speeds up the algorithm typically by a factor of 3 or 4 and makes it possible to cluster huge datasets that do not fit in memory.



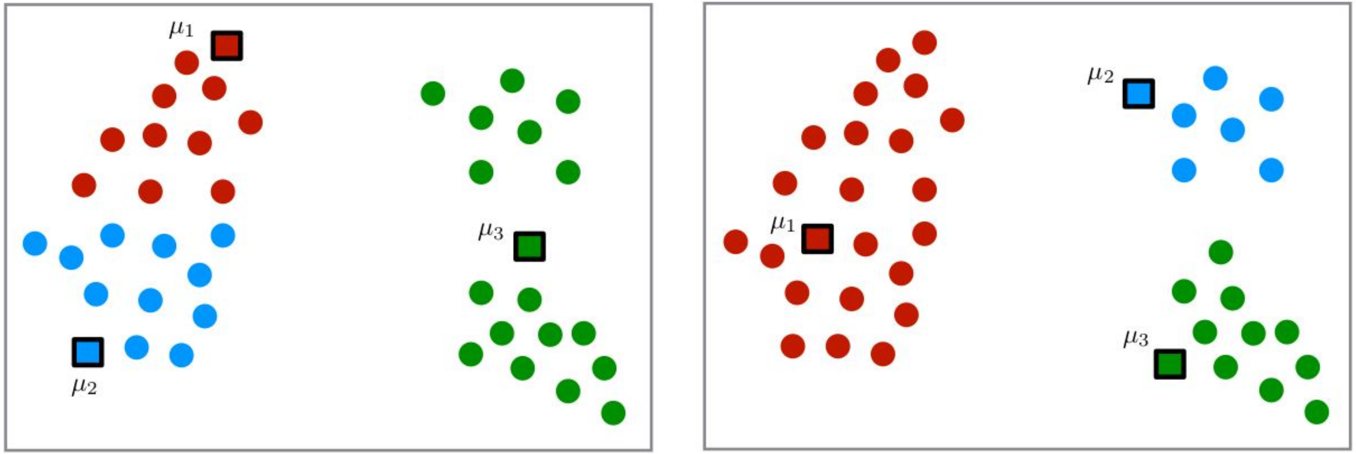
Lloyd’s Algorithm: Cluster Initialization

Results produced by Lloyd's algorithm are often highly dependent on the initial solution. Different starting positions can lead to different local minima - i.e. different clusterings of the same data.



Lloyd’s Algorithm: Cluster Initialization

A poor choice of initial centroids will often lead to a poor clustering that is not useful. A better initialisation will lead to different clusters.



Initialisation 1 Initialisation 2

Common strategy: Run the algorithm multiple times, select the solution(s) that scores well according to some validation measure.

k-Means++ Initialiser:

- Start with $C = \varnothing$
- Pick $x \in X$ uniformly at random and add it to C
- Repeat $k - 1$ times:
  - Pick an $x \in X$ with probability proportional to $d(x, C)^2$
  - Add x to C

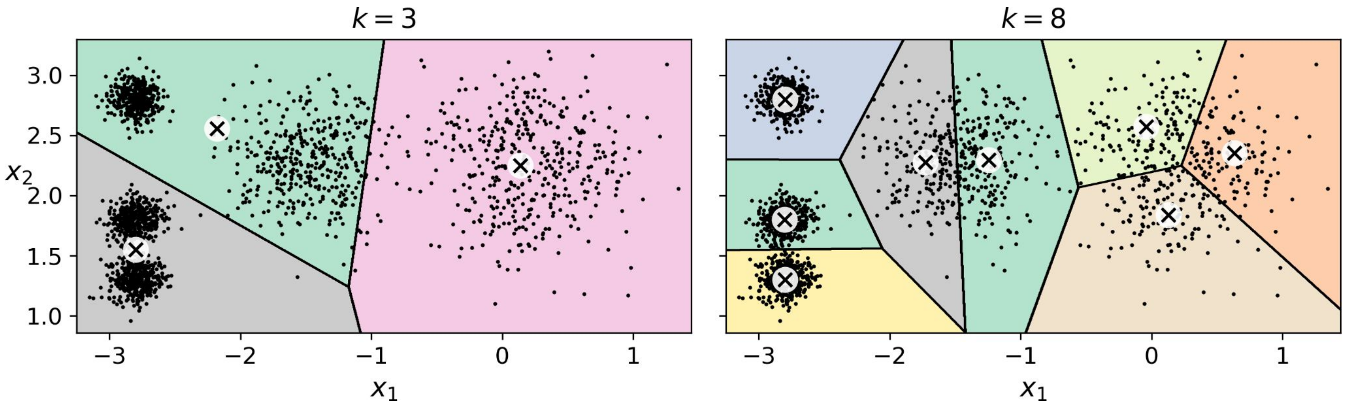
tends to select centroids that are distant from one another

Guarantee: Let C be the solution returned by k-Means++ and let C^* be the optimal solution. Then, $E[Cost(C)] \leq O(\log k) \cdot Cost(C^*)$

K-Means Clustering: How Many Clusters?

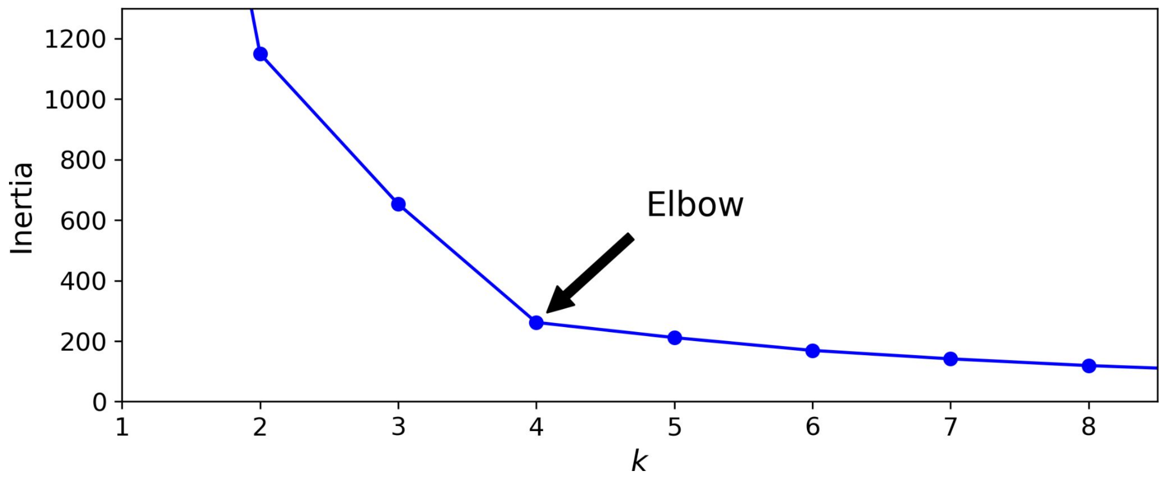
Key input parameter k - how many clusters?

- k too low → “smearing” of clusters that should not be merged.
- k too high → “over-clustering” of the data into many small, similar Clusters.



K-Means Clustering: How Many Clusters?

- The *inertia* is not a good performance metric when trying to choose k since it keeps getting lower as we increase k.
- The inertia drops very quickly as we increase k up to 4, but then it decreases much more slowly as we keep increasing k.



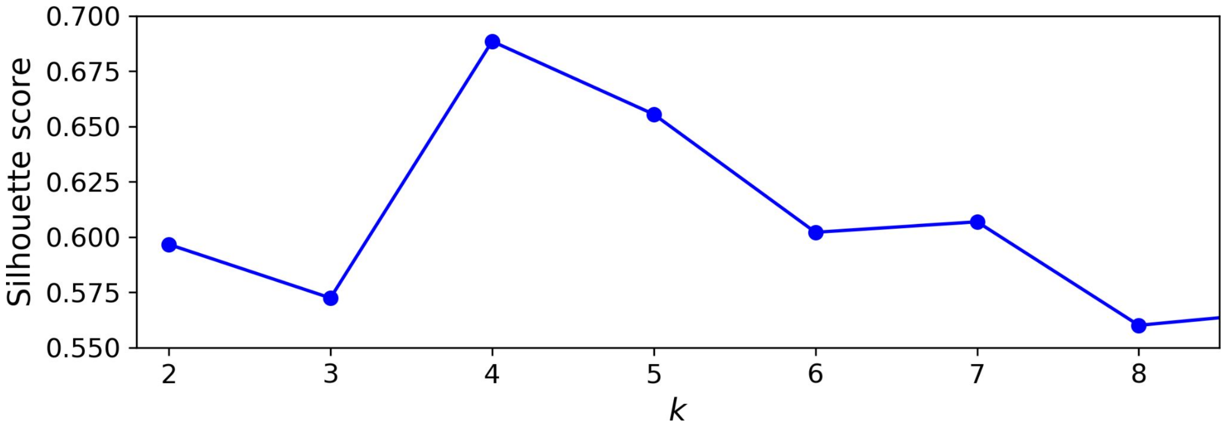
K-Means Clustering: How Many Clusters?

More precise way to use ***silhouette score***, which is the mean *silhouette coefficient* over all the instances.

- An instance's *silhouette coefficient* is equal to `(b−a)/max(a,b)` where
  - a is the mean distance to the other instances in the same cluster (it is the mean intra-cluster distance), and
  - b is the mean nearest-cluster distance, that is the mean distance to the instances of the next closest cluster (defined as the one that minimizes b, excluding the instance's own cluster).

K-Means Clustering: How Many Clusters?

- The *silhouette coefficient* can vary between -1 and +1:
  - a coefficient close to +1 means that the instance is well inside its own cluster and far from other clusters,
  - a coefficient close to 0 means that it is close to a cluster boundary, and
  - a coefficient close to -1 means that the instance may have been assigned to the wrong cluster.



K-Means Clustering

Advantages

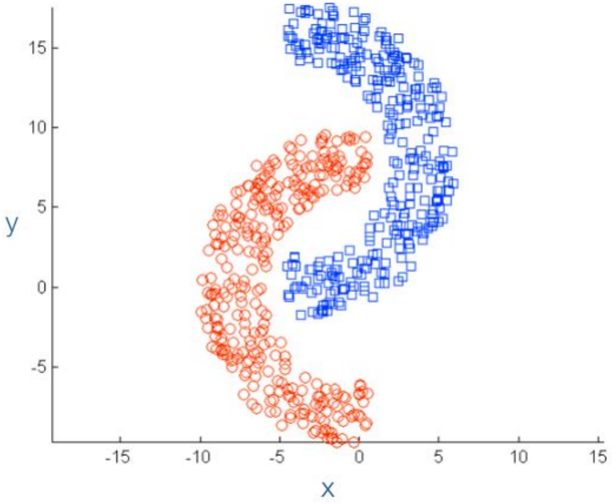
- Fast (for small dataset), easy to implement.
- “Good enough” in a wide variety of tasks and domains.

Disadvantages

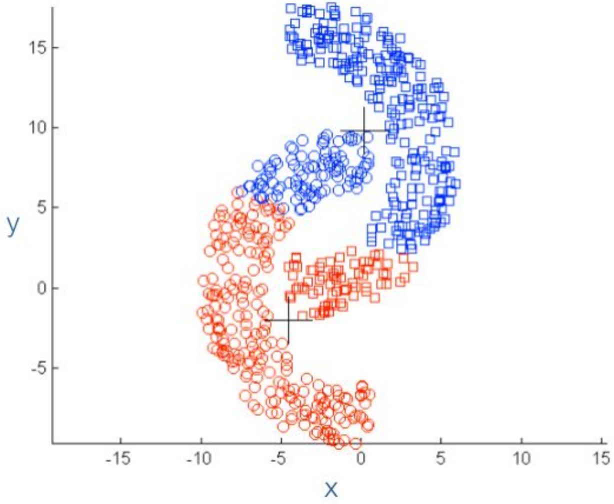
- Must pre-specify number of clusters k.
- Lloyd's algorithm is highly sensitive to choice of initial clusters.
- Assumes that each cluster is spherical in shape and data examples are largely concentrated near its centroid.
- Traditional objective can give undue influence to outliers.
- Iterative process can lead to empty clusters, particularly for higher values of k.

K-Means Clustering: Limitations

Example: k-Means assumes that clusters are spherical in shape and data examples are largely concentrated near its centroid.



Original "correct" groups in the data



Clusters identified by k-means for k=2

End of the Course