Unsupervised Machine Learning

Part One: Clustering

Agenda

- Overview of Unsupervised Learning
- Distance Metrics
- Clustering Algorithms

Unsupervised Learning

Definition

Unsupervised machine learning is the machine learning task of inferring a function to describe hidden structure from "unlabeled" data (a classification or categorization is not included in the observations). Since the examples given to the learner are unlabeled, there is no evaluation of the accuracy of the structure that is output by the relevant algorithm.

https://en.wikipedia.org/wiki/Unsupervised_learning

Important Points

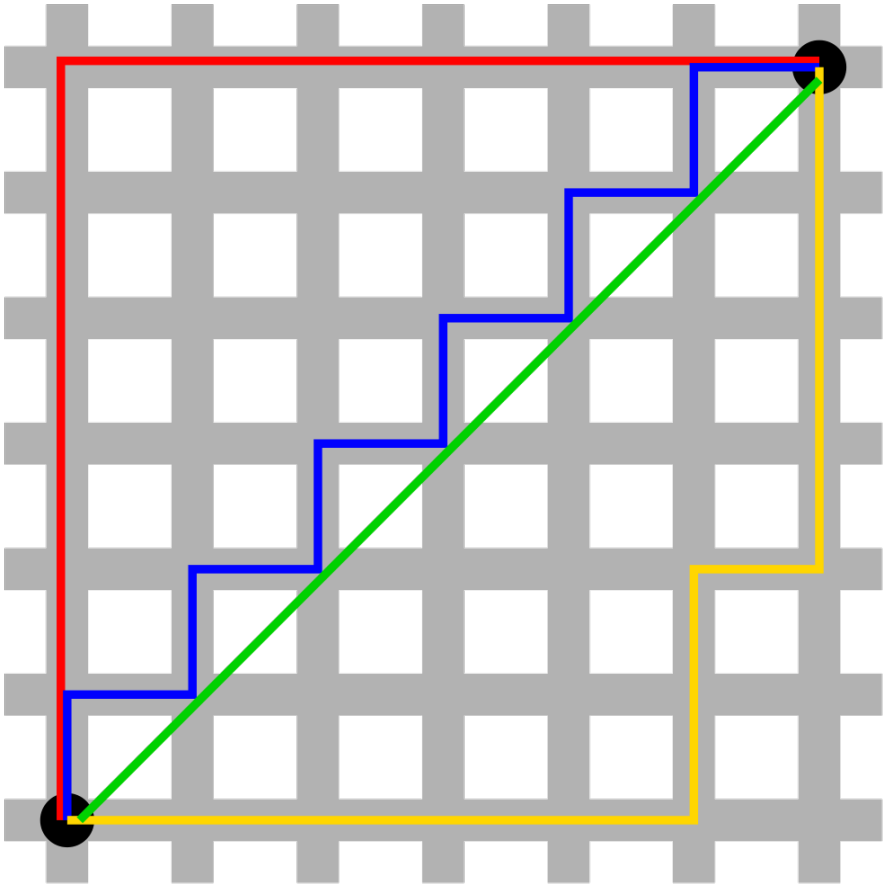
- Unlabeled training data
- Labels are produced by the algorithm
- Produces an inferred function
- No evaluation of accuracy

Approaches

- Clustering
- Neural networks
- Latent variable models
- Anomaly detection

Distance Metrics

Manhattan Distance

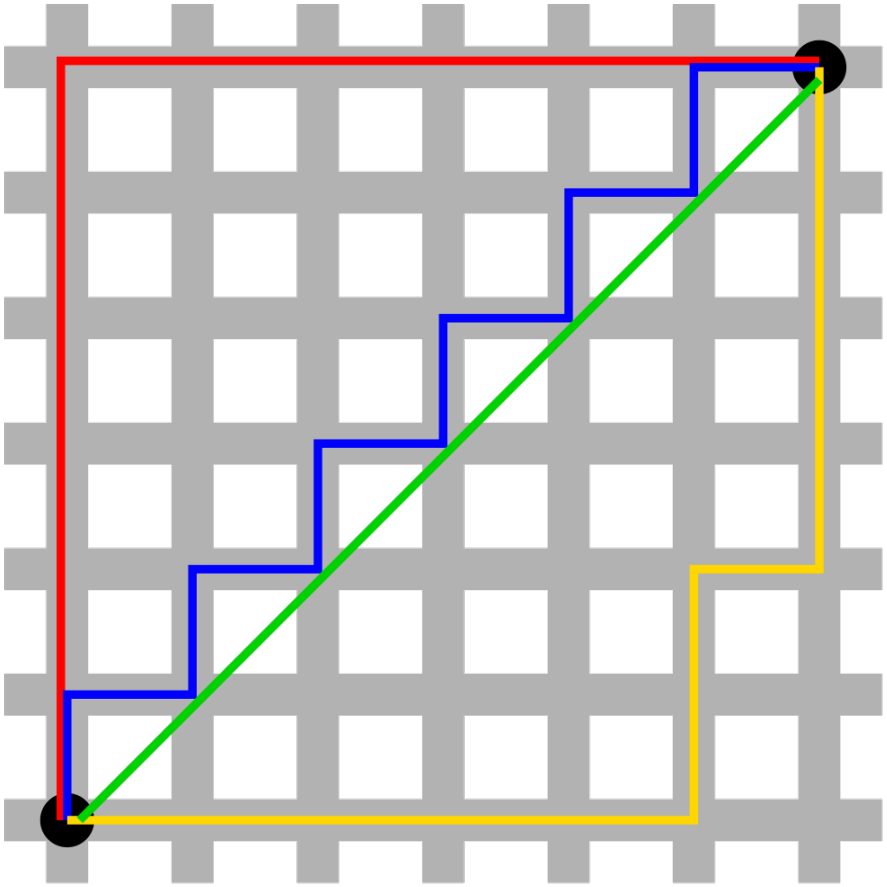


Taxicab (Manhattan)

Distance is the Red, Yellow, and Blue lines.

$D_{manhattan} = 12$

Euclidean Distance



Green line is Euclidean distance.

$D_{euclidean} = 6\sqrt{2} = 8.49$

Minkowski Distance

- A generalization of both Manhattan and Euclidean distance.
- Assumes all data is symmetric (distance is the same in all dimensions).

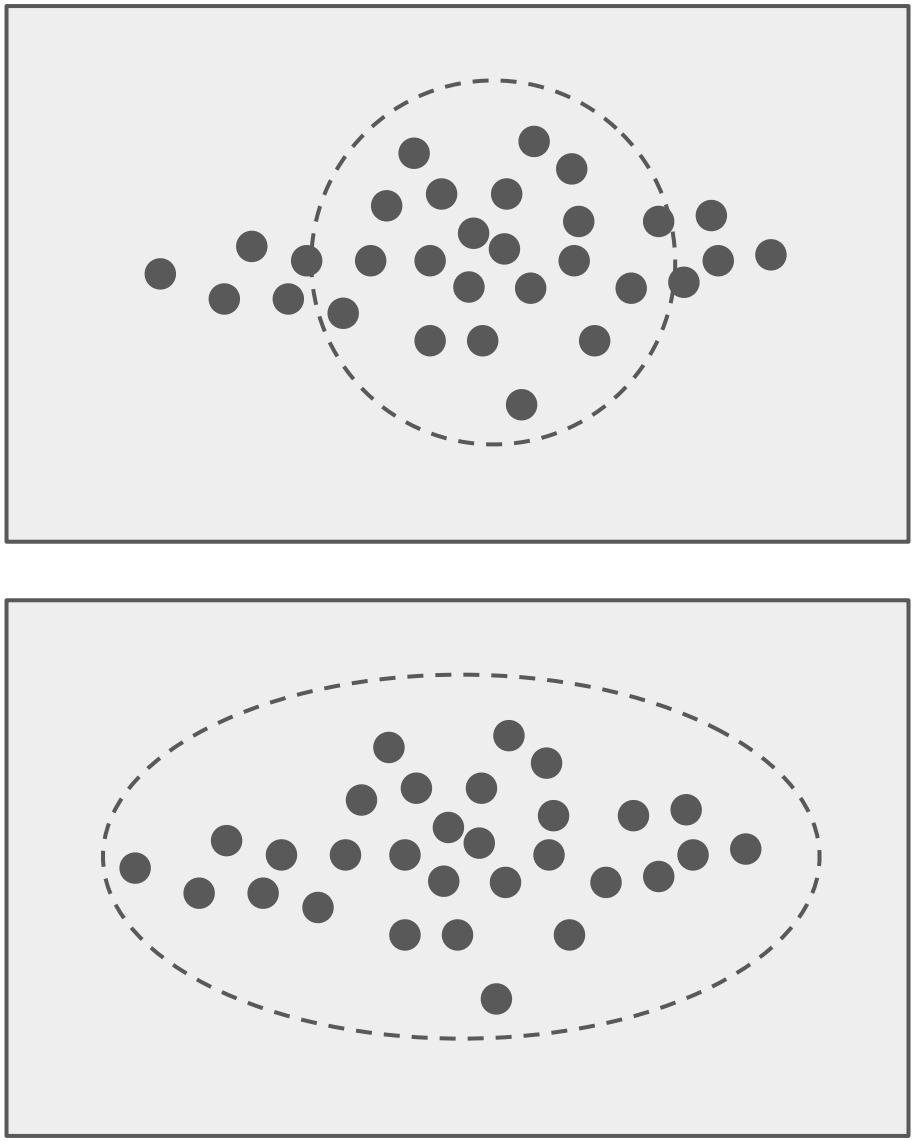
Mahalanobis Distance

Considers the volatility of each dimensions; creating a variable s_i for each dimension, the standard deviation of that set.

Mahalanobis Distance

By taking into account the shape of the data you can minimize the impact of high variance data.

Other techniques include PCA or LDA for dimension

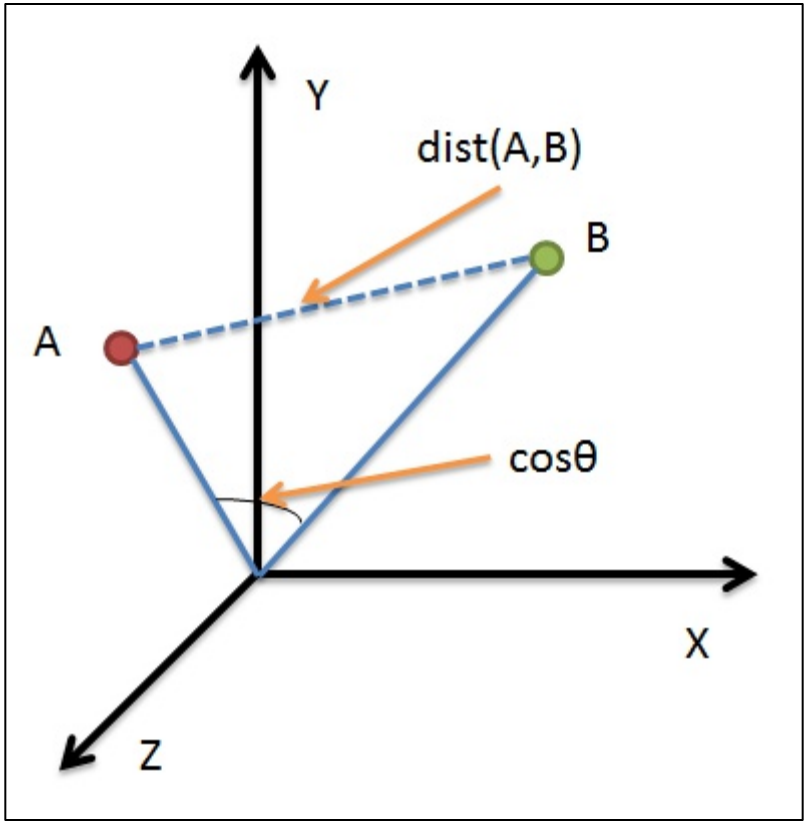


Cosine Similarity

Measures the angle between the vectors of two points in feature space.

Faster to compute, closer to Pearson correlation.

Not a true distance metric but is often used for spherical clusters.



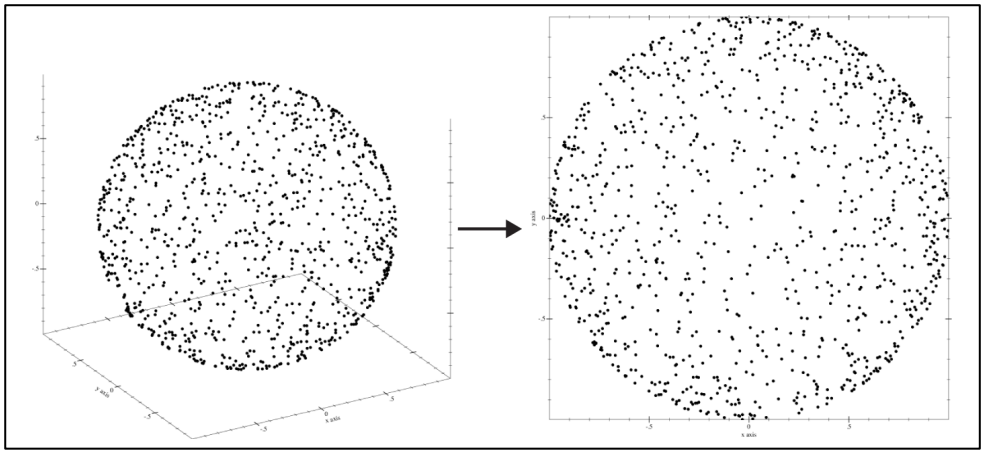
Match Scores Reference

String Matching	Distance Metrics	Relational Matching	Other Matching
Edit Distance - Levenstein - Smith-Waterman - Affine **Alignment** - Jaro-Winkler - Soft-TFIDF - Monge-Elkan **Phonetic** - Soundex - Translation	- Euclidean - Manhattan - Minkowski **Text Analytics** - Jaccard - TFIDF - Cosine similarity	**Set Based** - Dice - Tanimoto (Jaccard) - Common Neighbors - Adar Weighted **Aggregates** - Average values - Max/Min values - Medians - Frequency (Mode)	- Numeric distance - Boolean equality - Fuzzy matching - Domain specific **Gazettes** - Lexical matching - Named Entities (NER)

Curse of Dimensionality

High dimensional data tends to be sparse and far apart.

Algorithms that are based in locality need to determine nearness, but reducing or increasing dimensions can skew results.



Moving from 3 dimensions to 2, average distance between points is smaller.

Clustering Algorithms

Clustering Algorithms

- K-Means
- Hierarchical
- Parallel canopy

K-Means

K-Means

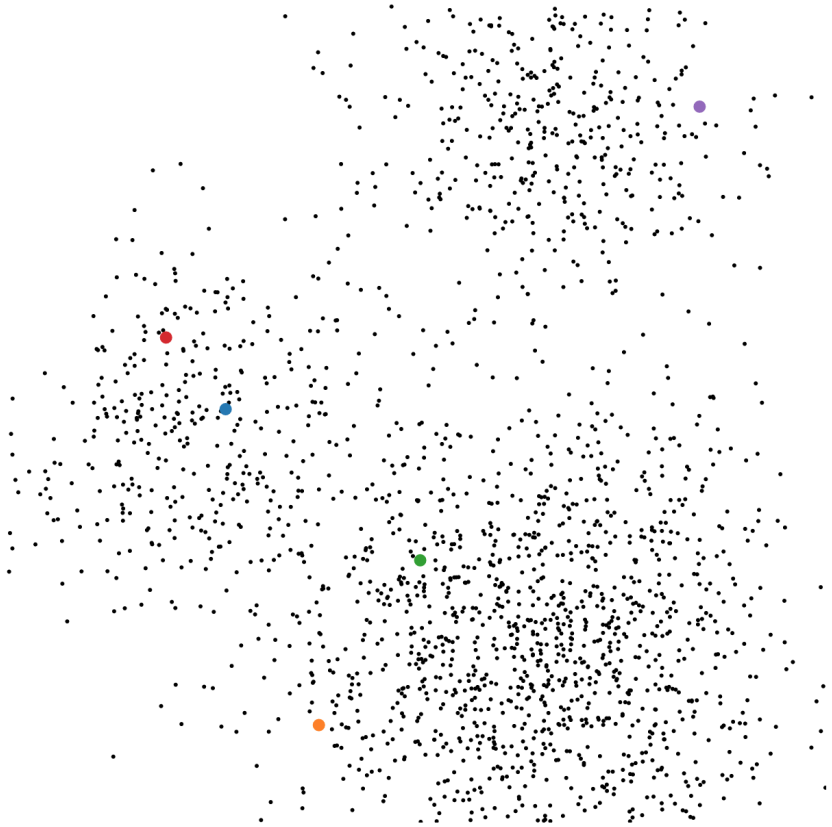
Straight Forward, Mature Algorithm

- Select predefined K
- Pick K random points in the data set to be centroids
- For each point
  - Assign it to closest centroid
  - Compute middle of cluster, move centroid
  - Repeat previous 2 steps until max iterations or min change between last two iterations is met

K-Means Considerations

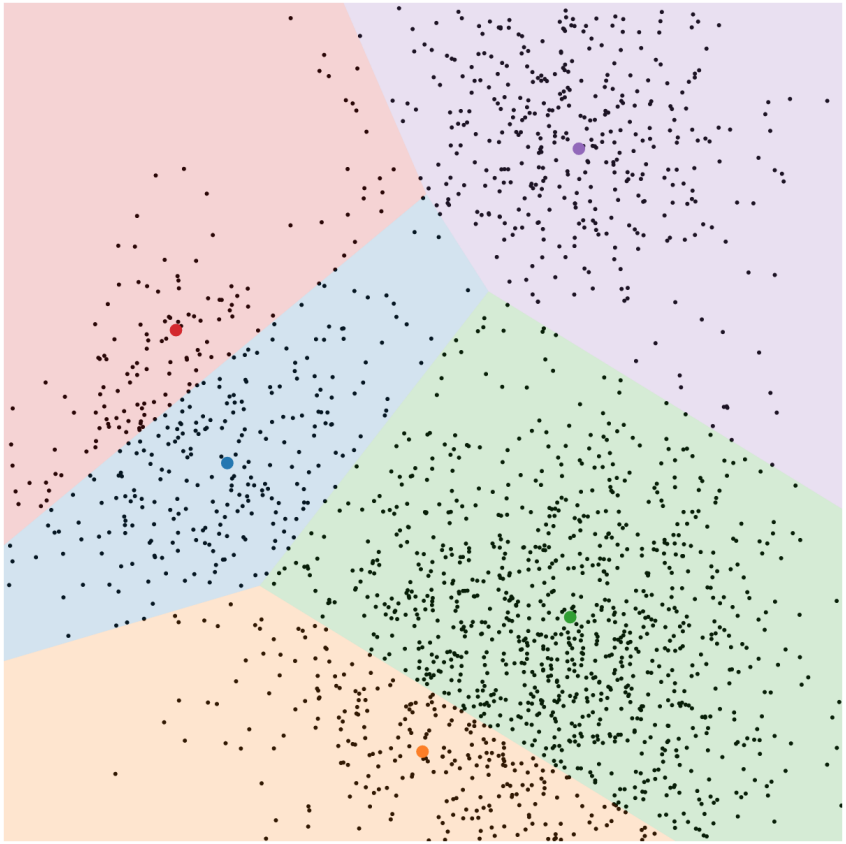
- Which distance metric to use?
- How do you choose K?

K-Means: Step 0



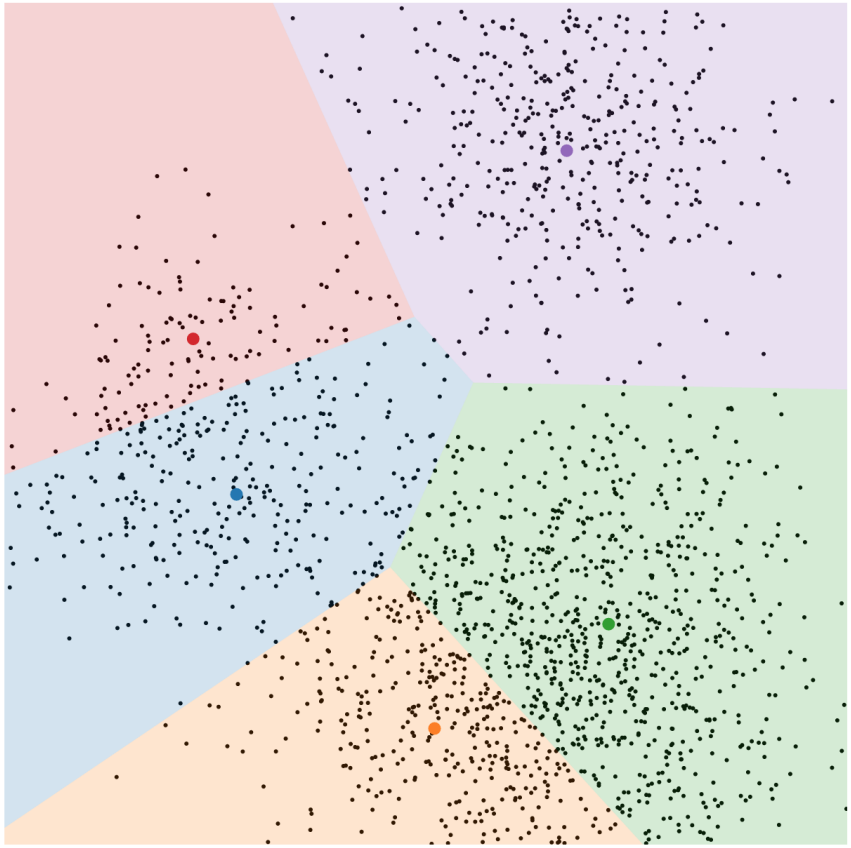
http://bit.ly/1BCeGFY

K-Means: Step 1



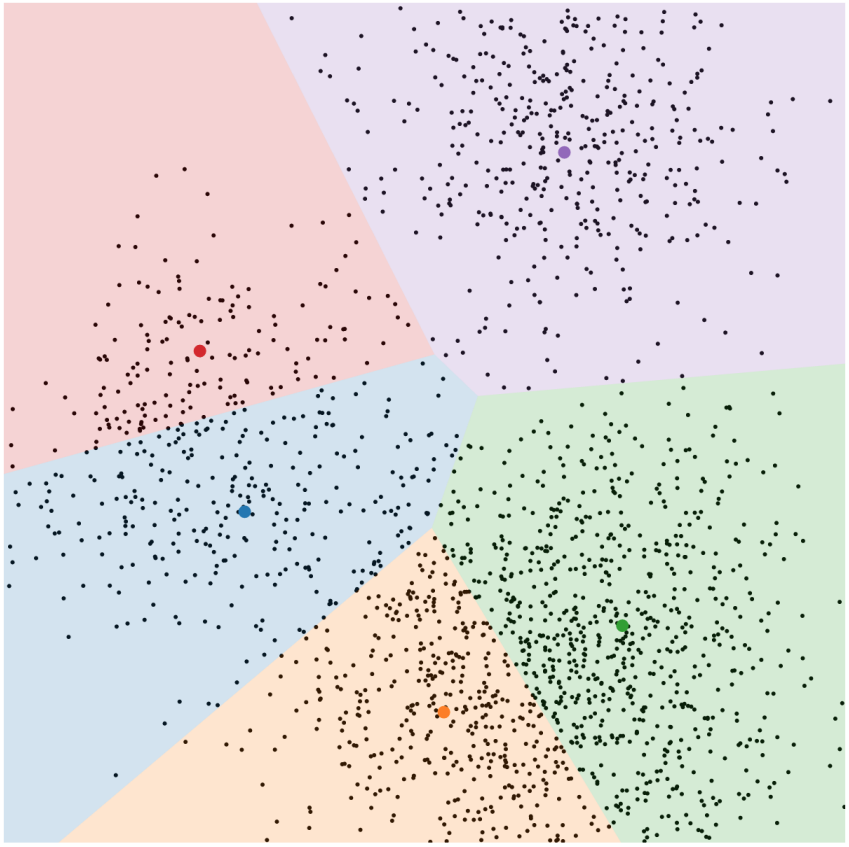
http://bit.ly/1BCeGFY

K-Means: Step 2



http://bit.ly/1BCeGFY

K-Means: Step 3



http://bit.ly/1BCeGFY

K-Means

Pros:

- Will find most optimal centers for the cluster
- Clusters are spherical
- Will converge to a solution

Cons:

- What K to use?
- Hard boundaries (no gray areas)
- Spherical clusters
- Stochastic: randomly determined

Impossibility Theorem

No free lunch!

Cannot have more than two of the following:

- **Richness**: there exists a distance function that can yield all different types of partitions.
- **Scale Invariance:** if numbers are scaled, same clusters should happen - units don't matter
- **Consistency**: shrinking distance between points, then expanding them should yield the same result

Most clustering has Richness and Scale-Invariance, but not consistency.

Lab: K-Means Clustering

Sklearn & MLlib

Hierarchical Clustering

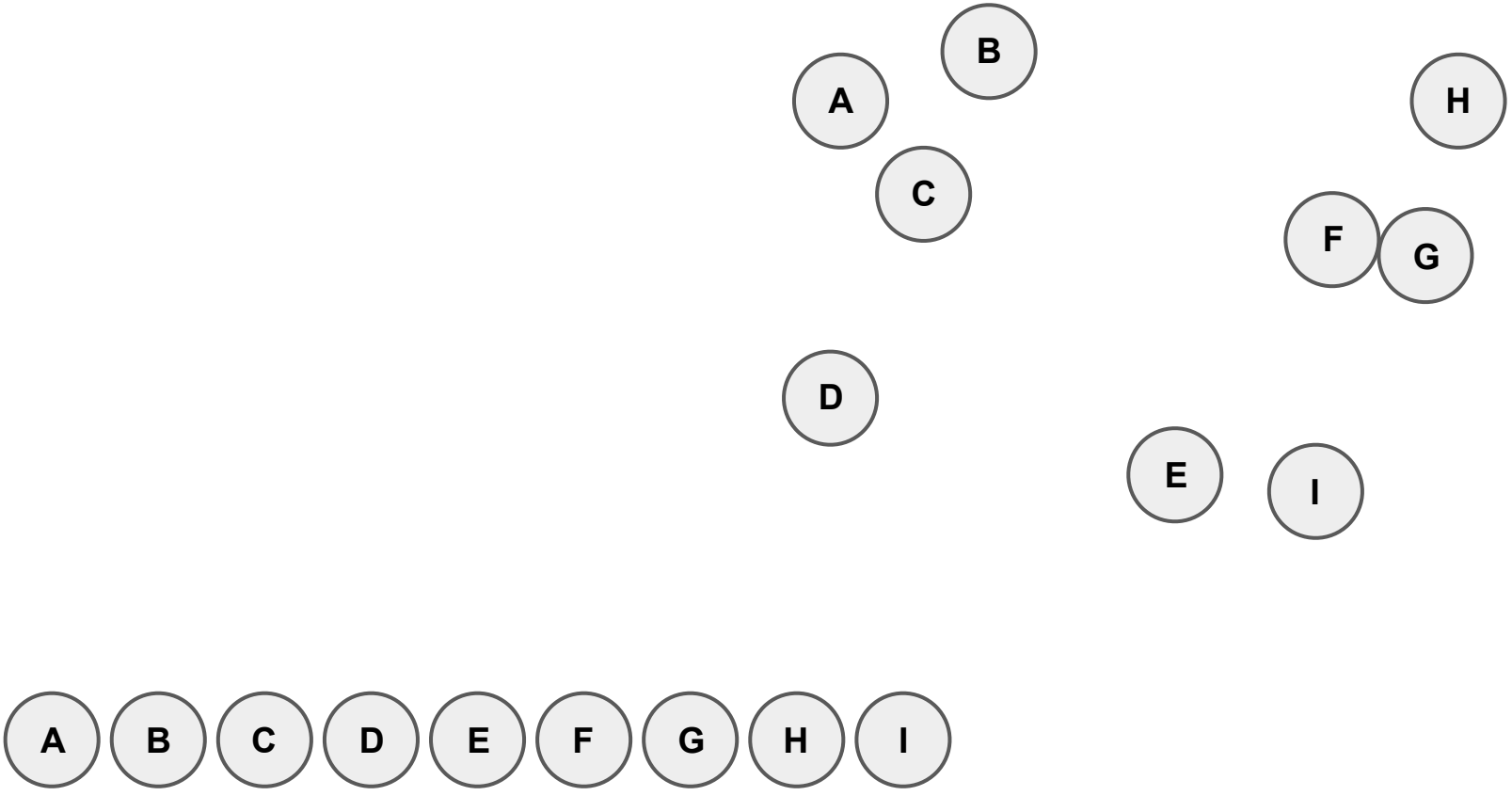
Hierarchical Clustering

We want strong membership as a hierarchy.

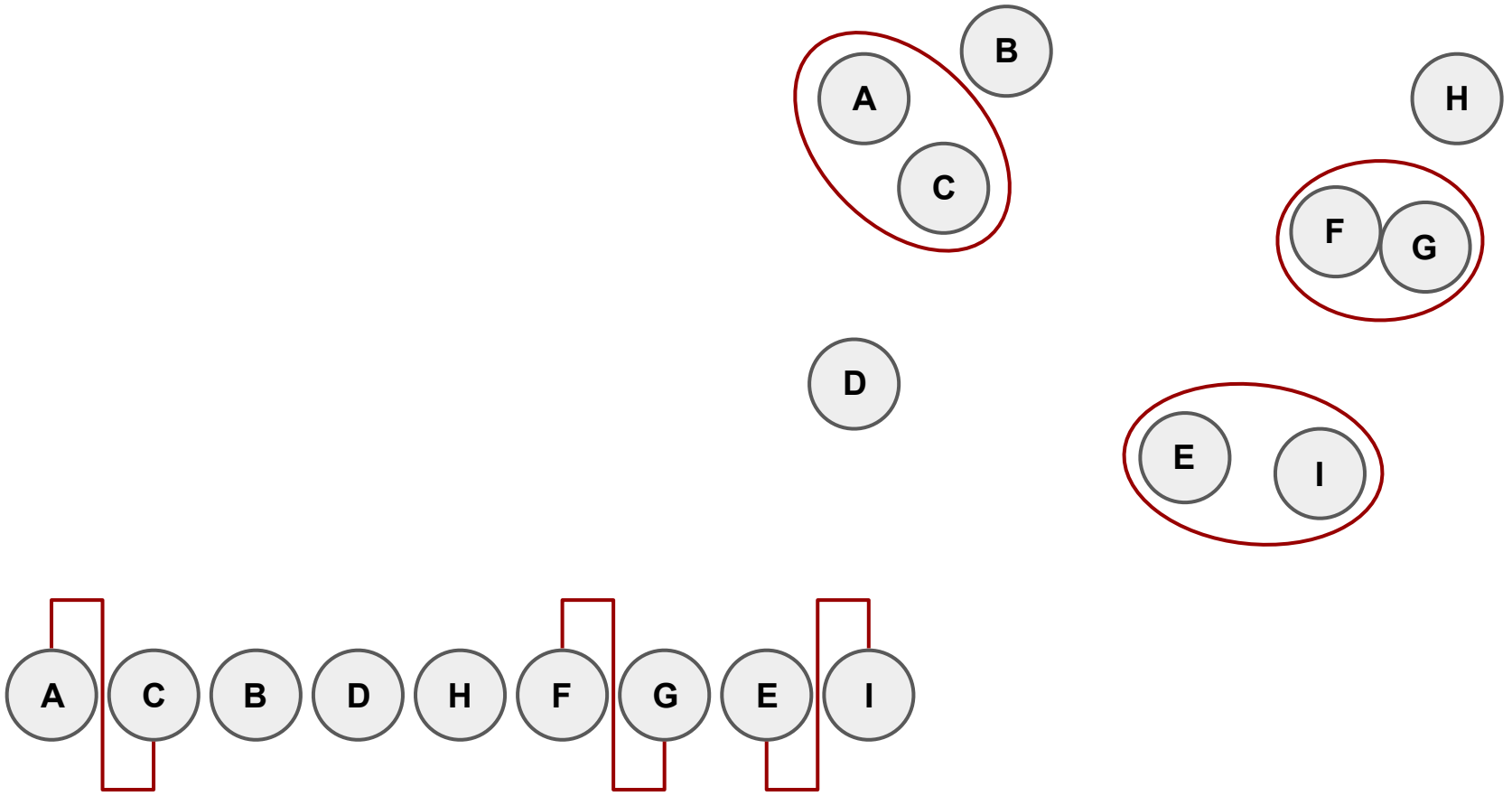
- Start with all data points as their own cluster
- Repeat until only a single cluster is left:
  - Find 2 closest points x_i and x_j
  - Merge points into a single cluster
  - Remove previous singleton clusters

This method creates a dendrogram of clusters- a hierarchical tree representing the cluster structure!

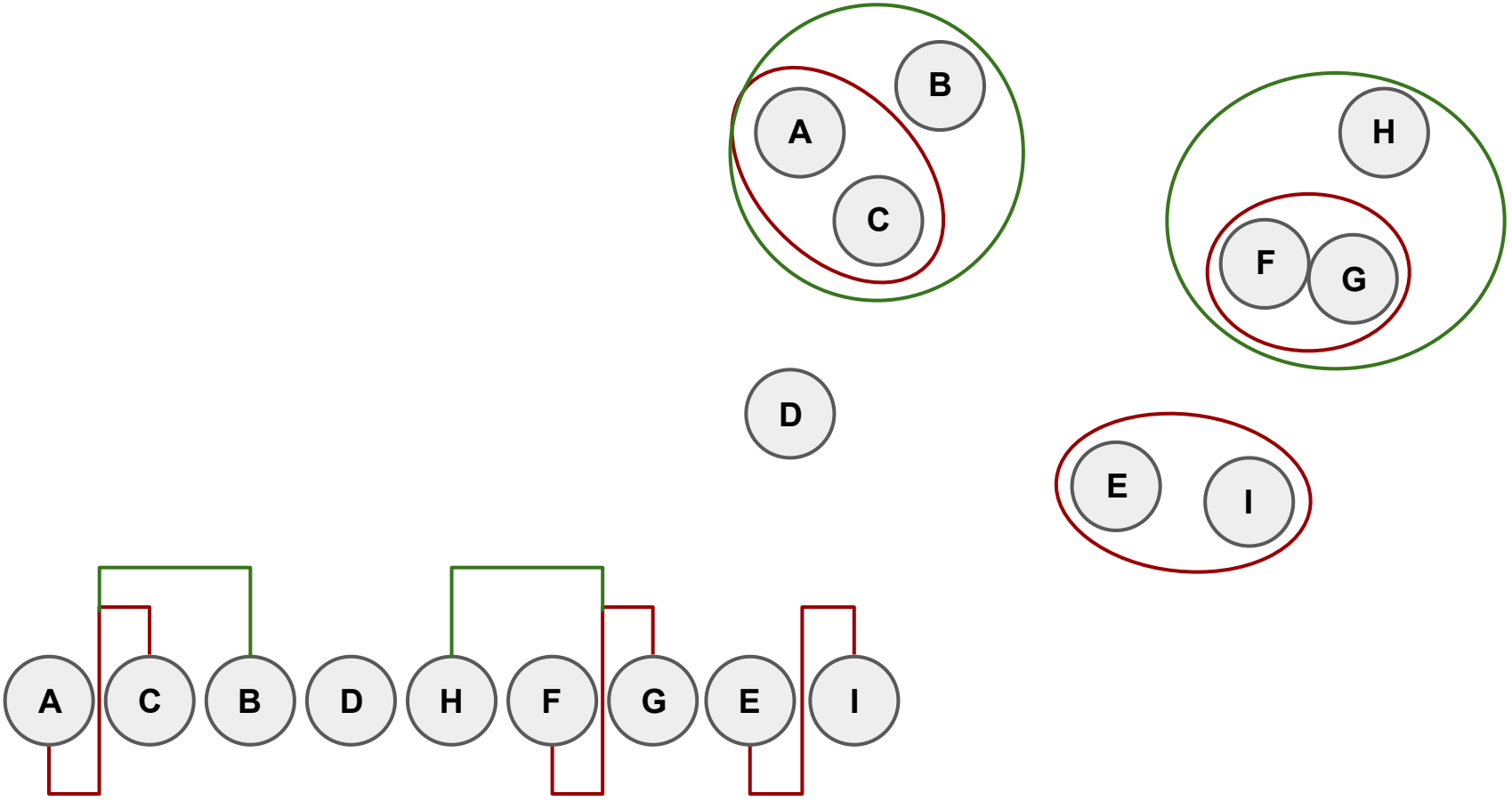
Hierarchical Clustering Example



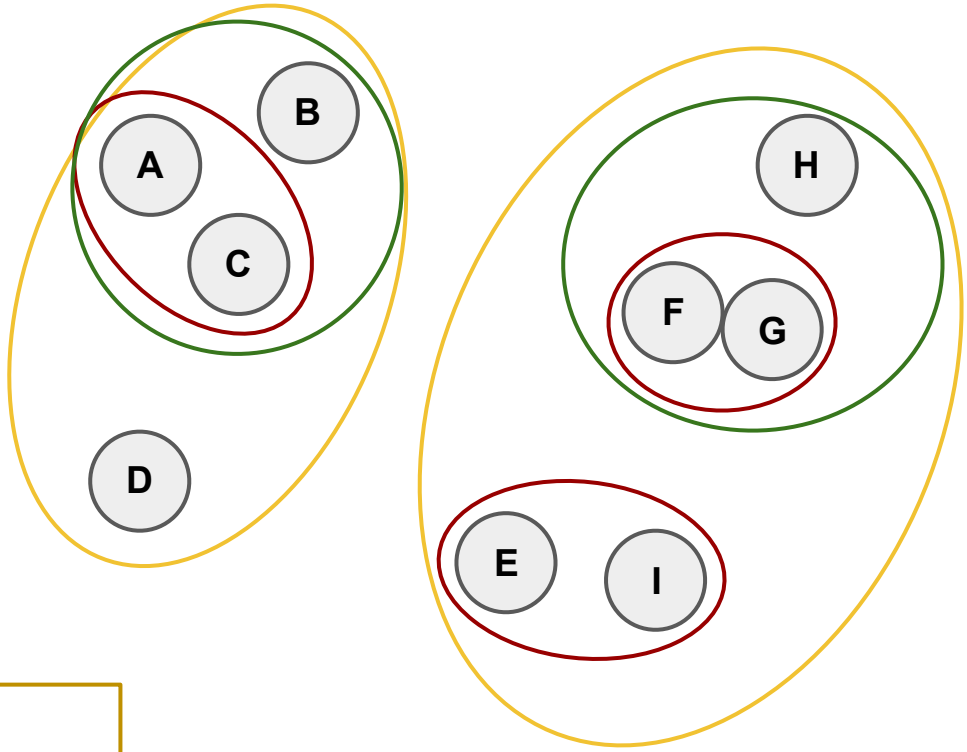
Hierarchical Clustering Example: Steps 1-3

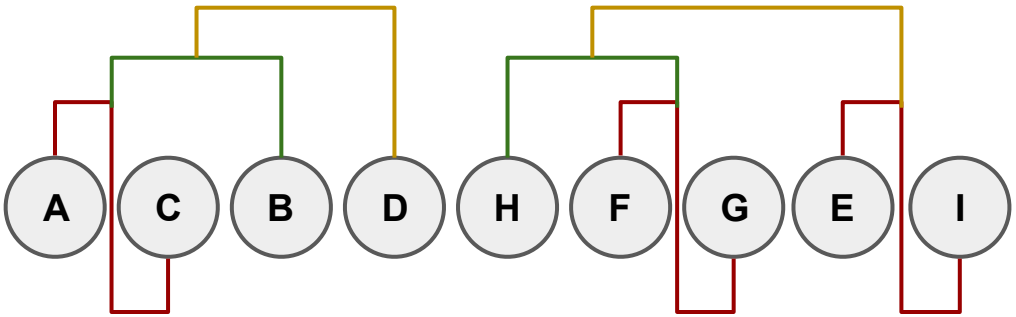


Hierarchical Clustering Example: Steps 4-5

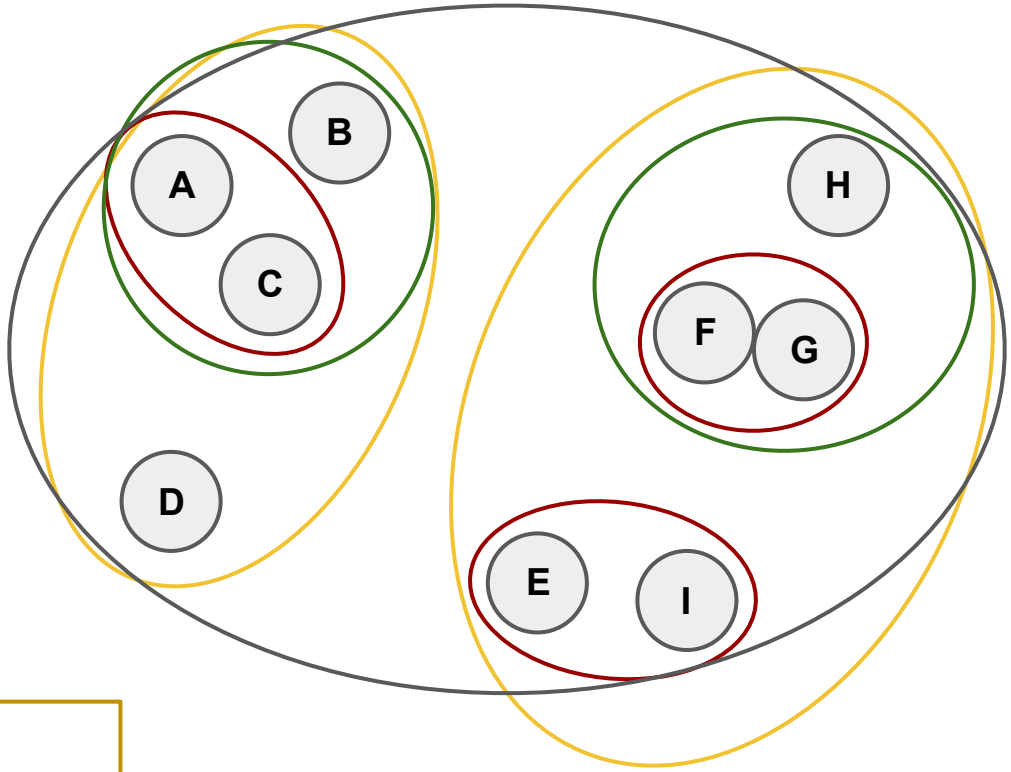


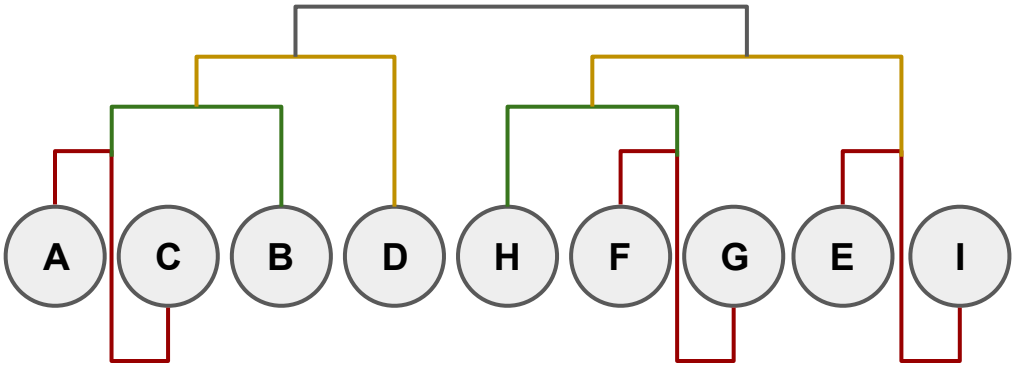
Hierarchical Clustering Example: Step 6





Hierarchical Clustering Example: Step 7





Lab: Hierarchical Clustering

Sklearn & MLlib

Parallel Canopy Clustering

Canopy Clustering

An unsupervised *pre-clustering* algorithm that is often used as a preprocessing step for K-Means or Hierarchical clustering.

This algorithm is intended to speed up other clustering algorithms, particularly in large data sets that make these algorithms impractical.

Basically canopies are a form of “blocking” - reducing the computational space and the number of required pairwise distance comparisons.

Canopy Clustering

The algorithm begins with two thresholds, T_1 and T_2 where $T_1 > T_2$ - these thresholds are called the “loose” and “tight” distances.

Algorithm:

1. Remove a point from the set and start a new “canopy”
2. For each point in the set, assign it to the new canopy if the distance is less than the loose distance T_1.
3. If the distance is less than T_2 remove it from the original set completely.
4. Repeat until there are no more data points to cluster.

Distance Metrics

1. Euclidean Distance
2. Cosine Similarity
3. Jaccard Distance
4. Manhattan Distance
5. Levenshtein Distance
6. … more

In MapReduce, you can implement a Mapper with one of these distance metrics to come up with new canopy centers by iterating through points and determining if it is already within a canopy threshold.

Parallelism in Canopy Clustering

A mapping of the points to the canopies is used to assign centers (this can be done iteratively to refine the canopies).

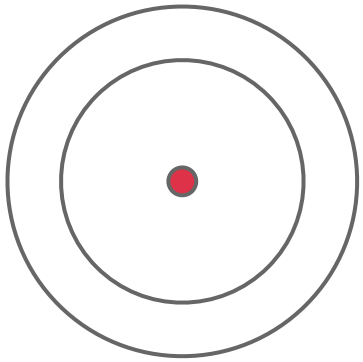
A reduction phase will combine centers that are redundant, in that they are within the threshold of each other.

Points can belong to multiple canopies for more rigorous clustering algorithms applied downstream.

Advantages: single-pass, fast enough to iterate different runs with different thresholds.

Canopy Clustering

Canopy Clustering

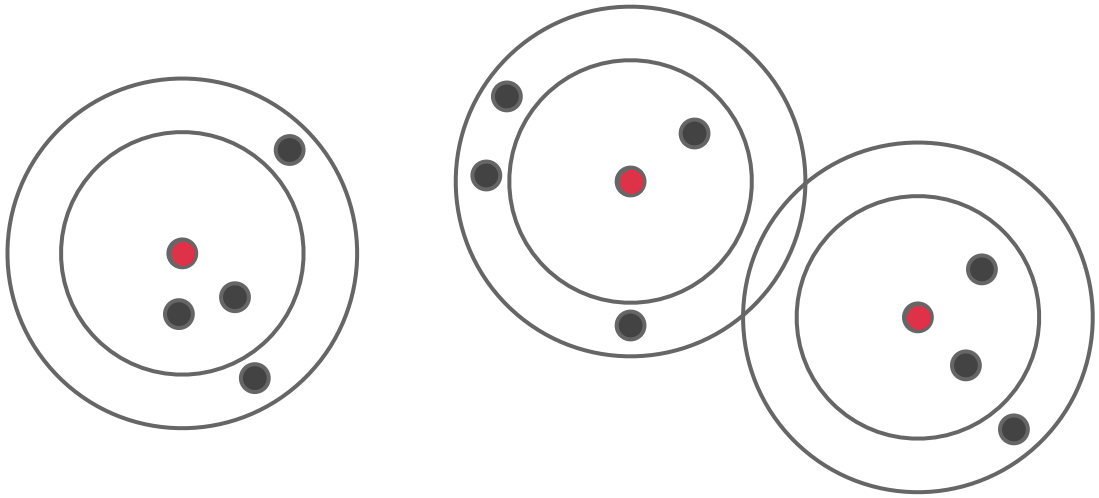


Canopy Clustering

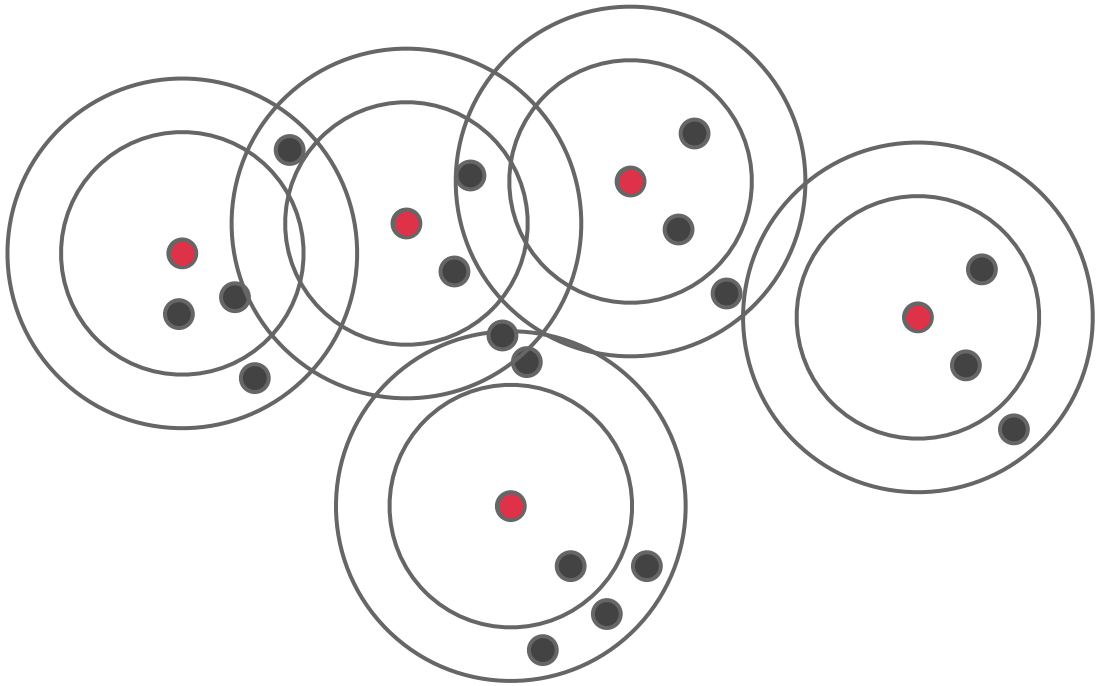
Canopy Clustering

Canopy Clustering

Canopy Clustering

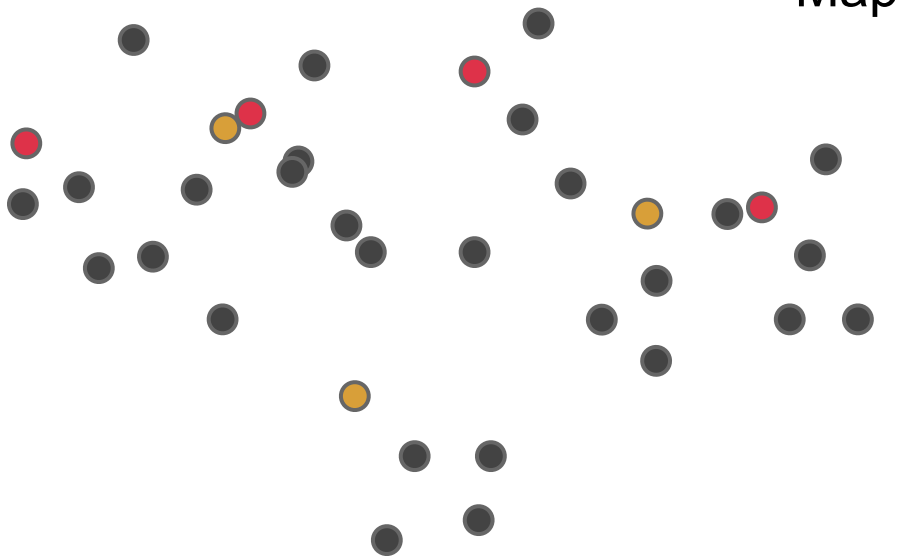


Canopy Clustering

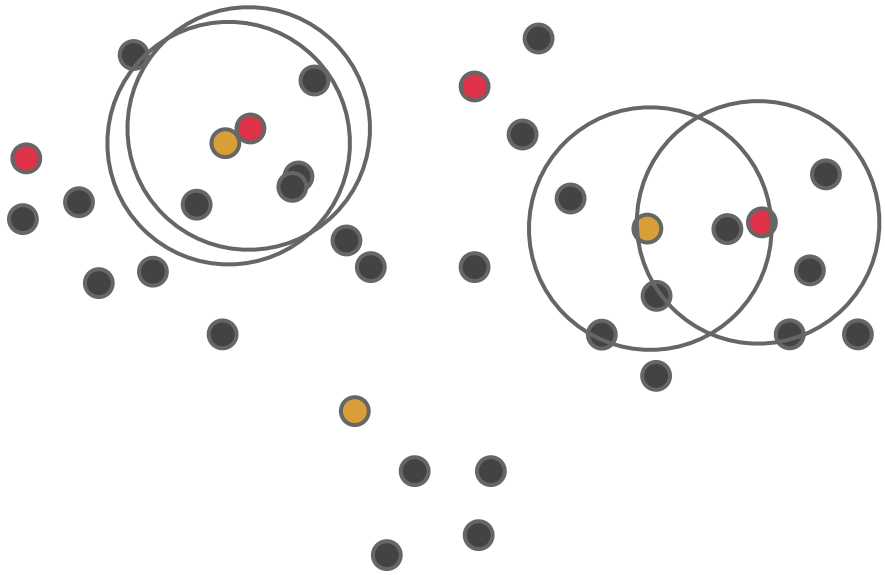


Canopy Clustering

Mapper A - Red Center
Mapper B - Gold Center



Canopy Clustering



Reducer computes redundant centers that are within the threshold of each other.

Clustering Text Documents

Lab: MiniBatch K-Means

Sklearn

Lab: LDA Clustering

MLlib