CS 1675 Introduction to Machine Learning
Lecture 18

Clustering

Milos Hauskrecht
milos@cs.pitt.edu
5329 Sennott Square

K-means clustering algorithm

- an iterative clustering algorithm
- works in the d-dimensional R space representing **x**

K-Means clusterting algorithm:

Initialize randomly *k* values of means (centers)

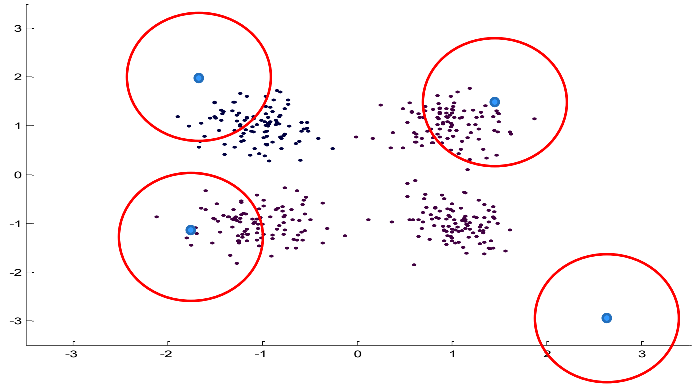
Repeat

- Partition the data according to the current set of means (using the similarity measure)
- Move the means to the center of the data in the current partition

Until no change in the means

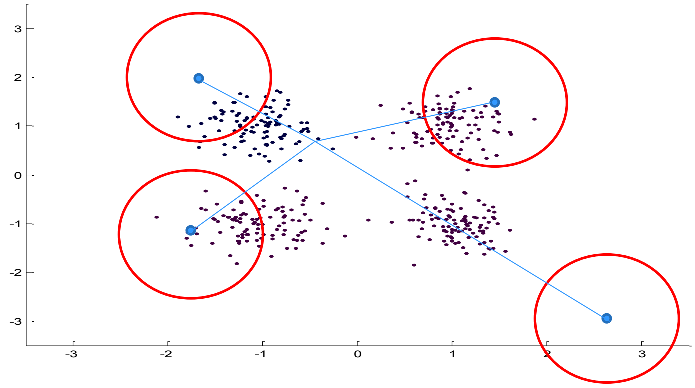
K-means: example

- **Initialize the cluster centers**



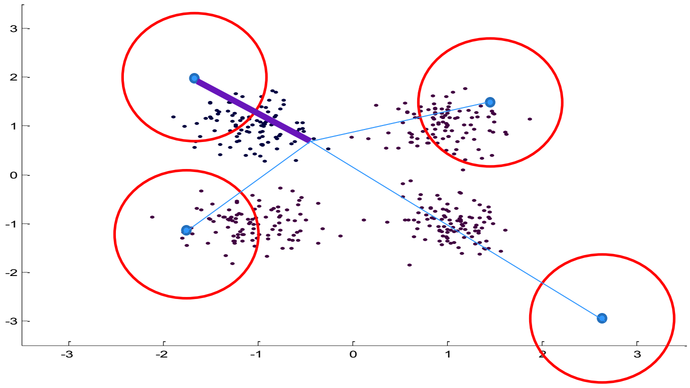
K-means: example

- **Calculate the distances of each point to all centers**



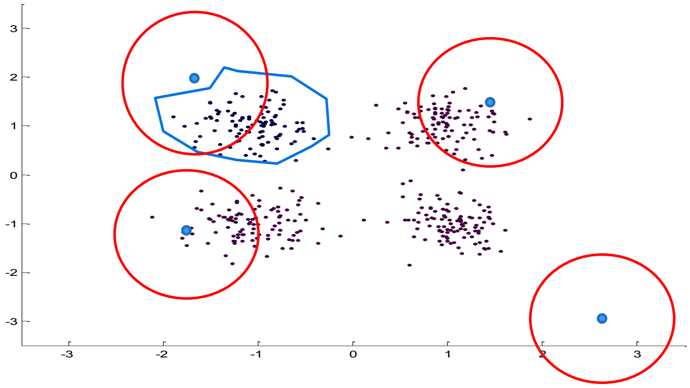
K-means: example

- For each example pick the best (closest) center



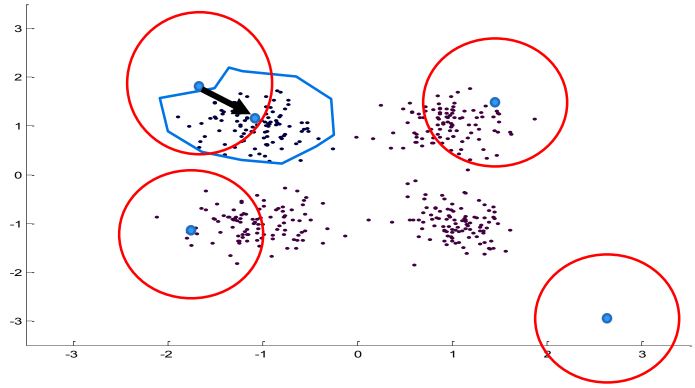
K-means: example

- Recalculate the new mean from all data examples assigned to the same cluster center



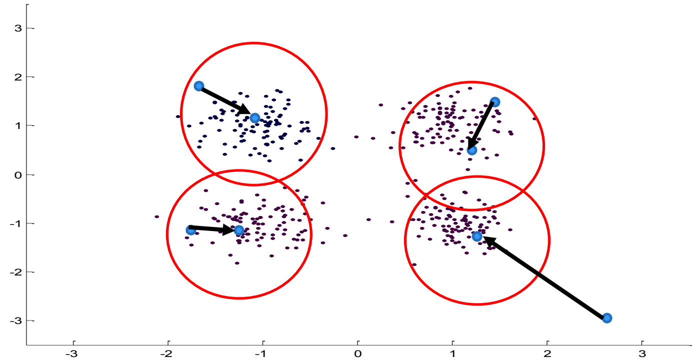
K-means: example

- Shift the cluster center to the new mean



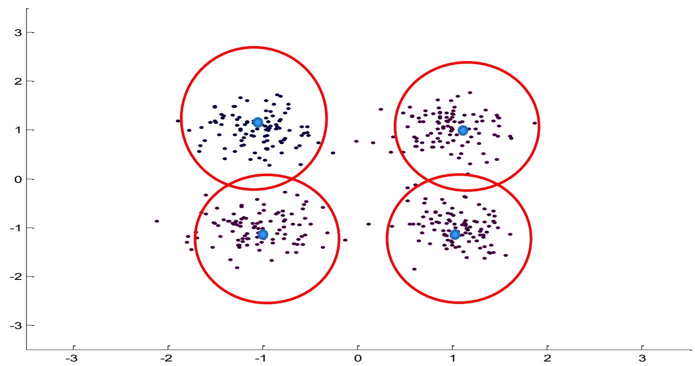
K-means: example

- Shift the cluster centers to the new calculated means



K-means: example

- **And repeat the iteration …**
- **Till no change in the centers**



K-means clustering algorithm

K-Means algorithm:

Initialize randomly k values of means (centers)

Repeat

- Partition the data according to the current set of means (using the similarity measure)
- Move the means to the center of the data in the current partition

Until no change in the means

Properties:

- Minimizes the sum of **squared center-point distances** for all clusters

$$\min_S \sum_{i=1}^{k} \sum_{x_j \in S_i} \| x_j - u_i \|^2 \qquad u_i = \text{center of cluster } S_i$$

K-means clustering algorithm

- **Properties:**
  - **converges** to centers minimizing the sum of squared center-point distances (still local optima)
  - The result is **sensitive** to the initial means' values
- **Advantages:**
  - Simplicity
  - Generality – can work for more than one distance measure
- **Drawbacks:**
  - Can perform poorly with overlapping regions
  - Lack of robustness to outliers
  - Good for attributes (features) with continuous values
    - Allows us to compute cluster means
    - k-medoid algorithm used for discrete data

Probabilistic (EM-based) algorithms

- **Latent variable models**

  Examples: Naïve Bayes with hidden class
  Mixture of Gaussians
- **Partitioning:**
  - the data point belongs to the class with the highest posterior
- **Advantages:**
  - Good performance on overlapping regions
  - Robustness to outliers
  - Data attributes can have different types of values
- **Drawbacks:**
  - EM is computationally expensive and can take time to converge
  - Density model should be given in advance

Hierarchical clustering

Uses an arbitrary similarity/dissimilarity measure

Typical similarity measures *d(a,b)* :

Pure real-valued data-points:

– Euclidean, Manhattan, Minkowski distances

Pure categorical data:

– Hamming distance, Number of matching values

Combination of real-valued and categorical attributes

– Weighted, or Euclidean

Hierarchical clustering

Two versions of the hierarchical clustering

– **Agglomerative approach**
  - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters

– **Divisive approach:**
  - Splits clusters in top-down fashion, starting from one complete cluster

Hierarchical (agglomerative) clustering

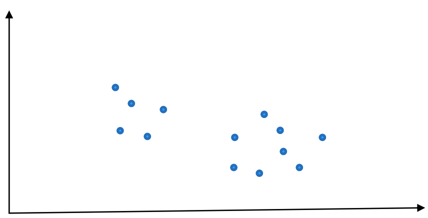
Approach:

- **Compute dissimilarity matrix for all pairs of points**
  - uses standard or other distance measures
- **Construct clusters greedily:**
  - **Agglomerative approach**
    - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters
- **Stop the greedy construction** when some criterion is satisfied
  - E.g. fixed number of clusters

Hierarchical (agglomerative) clustering

Approach:

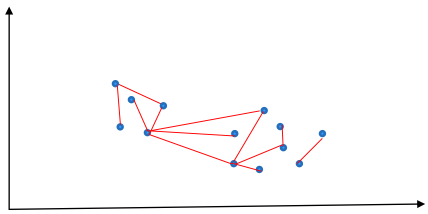
- **Compute dissimilarity matrix for all pairs of points**
  - uses standard or other distance measures



Hierarchical (agglomerative) clustering

Approach:

- **Compute dissimilarity matrix for all pairs of points**
  - uses standard or other distance measures

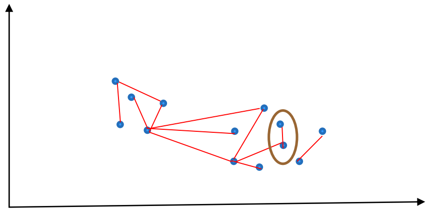


N points, $O(N^2)$ pairs, $O(N^2)$ distances

Hierarchical (agglomerative) clustering

Approach:

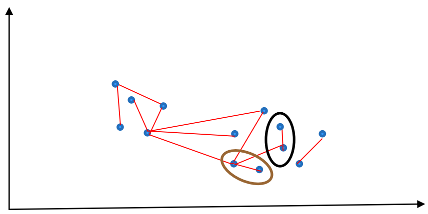
- **Compute dissimilarity matrix for all pairs of points**
  - uses standard or other distance measures
- **Construct clusters greedily:**
  - **Agglomerative approach**
    - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters



Hierarchical (agglomerative) clustering

Approach:

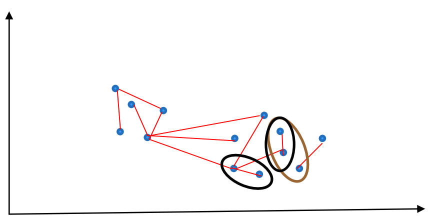
- **Compute dissimilarity matrix for all pairs of points**
  - uses standard or other distance measures
- **Construct clusters greedily:**
  - **Agglomerative approach**
    - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters



Hierarchical (agglomerative) clustering

Approach:

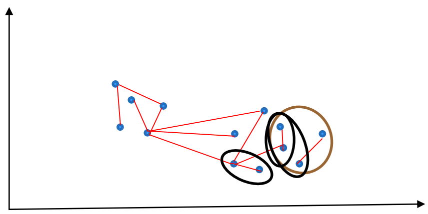
- **Compute dissimilarity matrix for all pairs of points**
  - uses standard or other distance measures
- **Construct clusters greedily:**
  - **Agglomerative approach**
    - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters



Hierarchical (agglomerative) clustering

Approach:

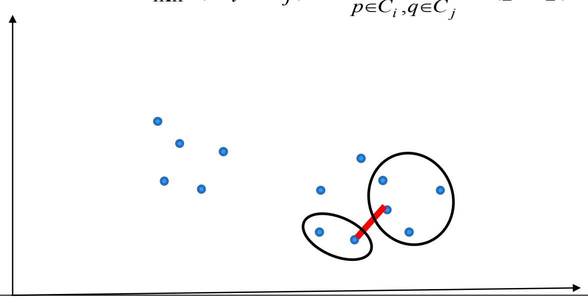
- **Compute dissimilarity matrix for all pairs of points**
  - uses standard or other distance measures
- **Construct clusters greedily:**
  - **Agglomerative approach**
    - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters



Cluster merging

- **Agglomerative approach**
  - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters
  - Merge clusters based on **cluster (or linkage) distances**. Defined in terms of point distances. **Examples:**

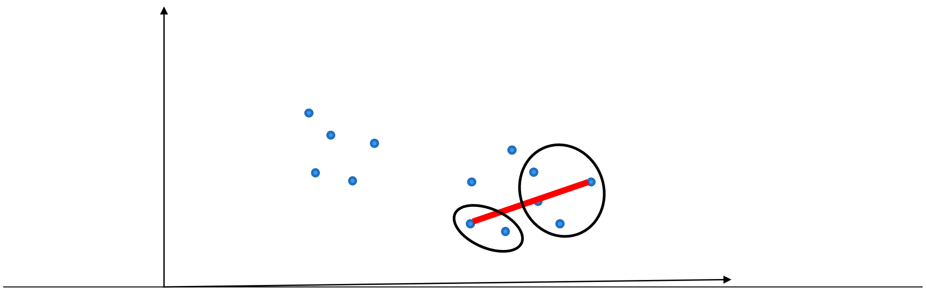
Min distance $$d_{\min}(C_i, C_j) = \min_{p \in C_i, q \in C_j} d(p,q)$$



Cluster merging

- **Agglomerative approach**
  - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters
  - Merge clusters based on **cluster (or linkage) distances**. Defined in terms of point distances. **Examples:**

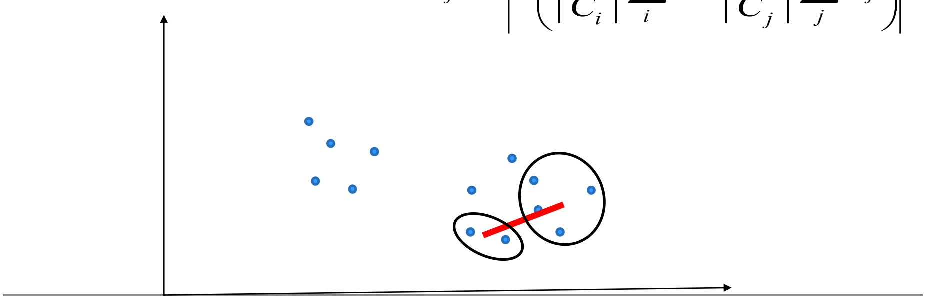
Max distance $$d_{\max}(C_i, C_j) = \max_{p \in C_i, q \in C_j} d(p, q)$$



Cluster merging

- **Agglomerative approach**
  - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters
  - Merge clusters based on **cluster (or linkage) distances**. Defined in terms of point distances. **Examples:**

Mean distance $$d_{mean}(C_i, C_j) = \left| d\left(\frac{1}{|C_i|} \sum_i p_i ; \frac{1}{|C_j|} \sum_j q_j \right) \right|$$



Hierarchical (agglomerative) clustering

Approach:

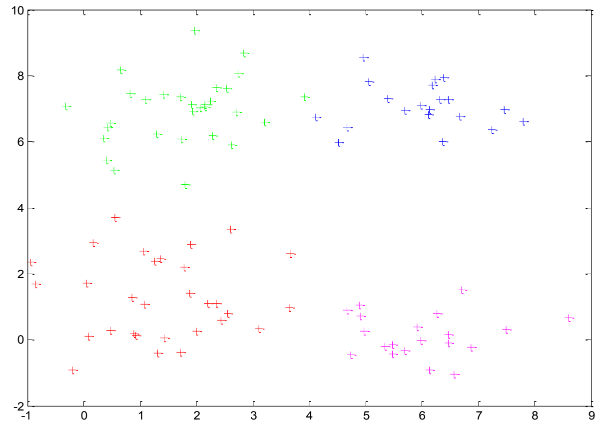
- **Compute dissimilarity matrix for all pairs of points**
  - uses standard or other distance measures
- **Construct clusters greedily:**
  - **Agglomerative approach**
    - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters
- **Stop the greedy construction** when some criterion is satisfied
  - E.g. fixed number of clusters

Hierarchical (divisive) clustering

Approach:

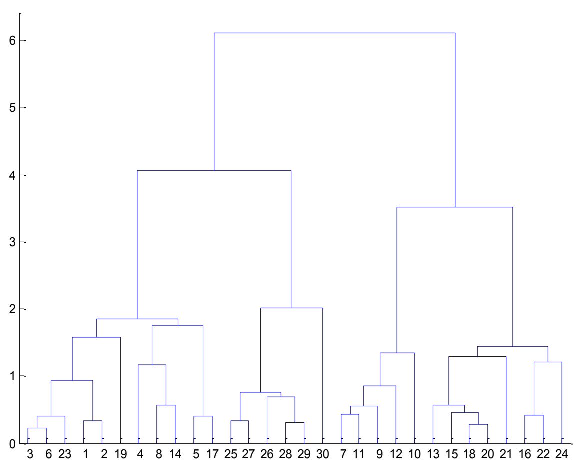
- **Compute dissimilarity matrix for all pairs of points**
  - uses standard or other distance measures
- **Construct clusters greedily:**
  - **Agglomerative approach**
    - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters
  - **Divisive approach:**
    - Splits clusters in top-down fashion, starting from one complete cluster
- **Stop the greedy construction** when some criterion is satisfied
  - E.g. fixed number of clusters

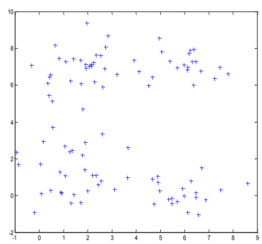
Hierarchical clustering example



Hierarchical clustering example

- **Dendogram**





Hierarchical clustering

- **Advantage:**
  - Smaller computational cost; avoids scanning all possible clusterings
- **Disadvantage:**
  - Greedy choice fixes the order in which clusters are merged; cannot be repaired
- **Partial solution:**
    - combine hierarchical clustering with iterative algorithms like k-means algorithm

Other clustering methods

- **Spectral clustering**
  - Uses similarity matrix and its spectral decomposition (eigenvalues and eigenvectors)
- **Multidimensional scaling**
  - techniques often used in data visualization for exploring similarities or dissimilarities in data.

CS 1675 Introduction to Machine Learning
Lecture 18

Ensemble methods

Milos Hauskrecht
milos@cs.pitt.edu
5329 Sennott Square

Ensemble methods

We know how to build different classification or regression models from data

- **Question:**
  - Is it possible to learn and combine multiple (classification/regression) models and improve their predictive performance ?
- **Answer:** **yes**
- There are different ways of how to do it…

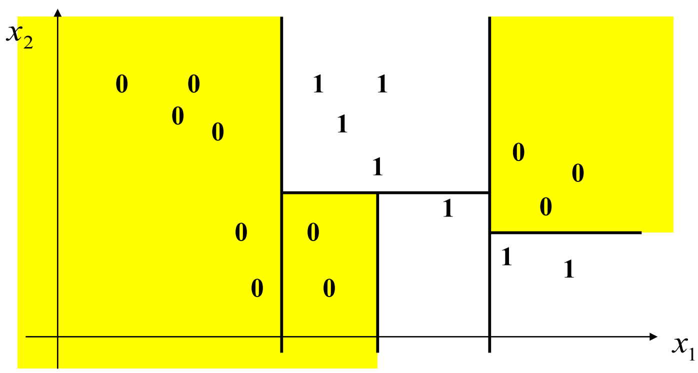
Ensemble methods

- **Question:**
  - Is it possible to learn and combine multiple (classification/regression) models and improve their predictive performance ?
- There are different ways of how to do it…

- Assume you have models M1, M2, … Mk
- **Approach 1:** use the different models (classifiers, regressors) to cover the different parts of the input (x) space
- **Approach 2:** use the models (classifiers, regressors) that cover the complete input (x) space

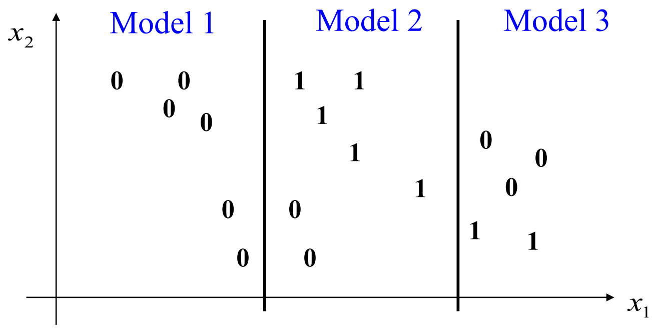
Approach 1

- Recall the decision tree:
  - **It partitions the input space to regions**
  - **It classifies independently in every region**



Approach 1

- Recall the decision tree:
  - **It partitions the input space to regions**
  - **It classifies independently in every region**
- **What if we define a more general partitions of the input space and learn a model specific to these partitions**



Approach 1

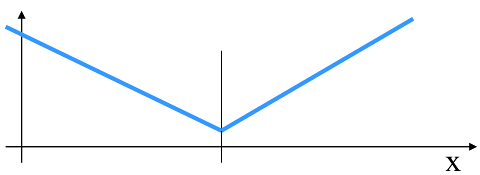
- Approach 1: **define a more general partitions of the input space and learn a model specific to these partitions**

Example:

- **Mixture of expert model:**
  - Different input regions covered with different learners
  - A “soft” switching between learners

- **Mixture of experts**

  Expert = learner



Approach 2

- **Approach 2:** use multiple models (classifiers, regressors) that cover the complete input (x) space
- **Committee machines:**
  - Each base model is trained on a slightly different train set
  - Combine predictions of all models to produce the output
    - **Goal:** Improve the accuracy of the 'base' model

- **Methods:**
    - **Bagging**
    - **Boosting**
    - Stacking (not covered)

Bagging (Bootstrap Aggregating)

- **Given:**
  - Training set of N examples
  - A class of learning models (e.g. decision trees, neural networks, …)
- **Method:**
  - Train multiple (k) models on slightly different datasets
  - Predict (test) by averaging the results of k models
- **Goal:**
  - Improve the accuracy of one model by using its multiple copies
  - Average of misclassification errors on different data splits gives a better estimate of the predictive ability of a learning method

Bagging algorithm

- **Training**
- For each model M1, M2, … Mk
  - Randomly sample with replacement *N* samples from the training set
  - Train a chosen "base model" (e.g. neural network, decision tree) on the samples
- **Test**
  - For each test example
    - Run all base models M1, M2, … Mk
    - Predict by combining results of all T trained models:
      - **Regression:** averaging
      - **Classification:** a majority vote

Class decision via majority voting