

# Lecture 9: Clustering Part I

## Statistical Methods for Data Science

**Yinan Yu**

Department of Computer Science and Engineering

December 3, 2020

# Today

- 1 Introduction
- 2 Modeling for clustering
- 3 Clustering tendency
  - Are there clusters in the data?
  - Distance based approach
  - Hopkins statistic
  - Histogram based technique
- 4 First clustering model: K-means
- 5 Summary

## Learning outcome

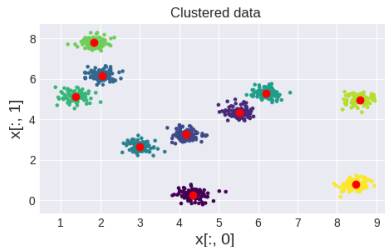
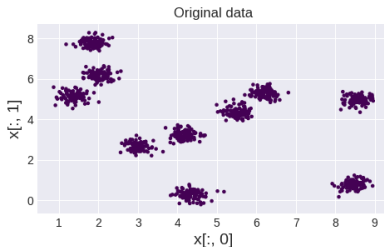
- Understand the difference between supervised learning and unsupervised learning
- Understand how to apply clustering algorithms to the applications discussed in this lecture
- Be able to compute histograms for high dimensional data
- Be able to compute the dissimilarity matrix with the Euclidean distance
- Be able to explain how to identify clusterability using the Hopkins statistic
- Be able to implement the K-means algorithm

# Today

- 1 Introduction
- 2 Modeling for clustering
- 3 Clustering tendency
- 4 First clustering model: K-means
- 5 Summary

# Clustering

- We start with blobs of data
- We assign some semantics to each of these data points



- Each of these semantics is called a **cluster**
- The process of finding clusters is called **clustering**

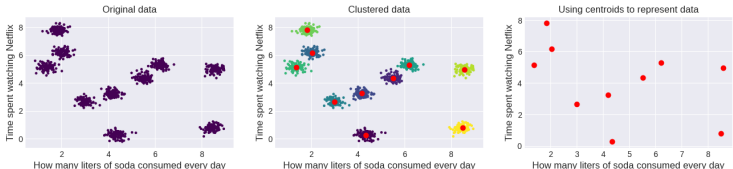
# Application

Clustering is widely used in different applications - clustering algorithm development **does not require expensive annotations**

## 1. Clustering as a preprocessing method

### 1.1 To summarize a large amount of data using their clusters

**Example:** you have access to the time people spend on Netflix and the amount of soda they consume everyday; you want to make a more advanced summary from this data set

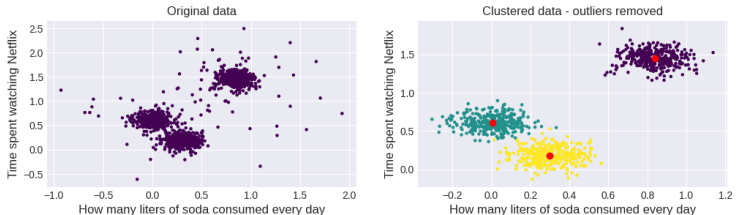


Group these people into clusters and correlate these patterns with other data sources

## Application (cont.)

### 1. Clustering as a preprocessing method (cont.)

#### 1.2 To detect and remove **outliers** - data points that are far away from any clusters



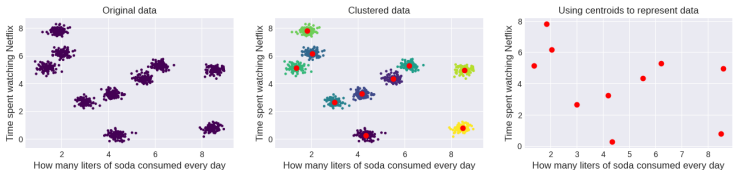
Without clustering, it is hard to define what should be considered outliers when the data distribution is **complex**:

- High dimensionality
- Data cannot be modeled with a single probability distribution

## Application (cont.)

### 2. Clustering as a data reduction technique

2.1 To reduce a large amount of data into fewer data points by, e.g., representing the data set with only the centroids - the set up is similar to 1.1



One important application is the **recommender system**

- Task: find patterns in preferred items from massive amount of users
- Challenge: there are too many users
- Solution: we recommend items to users on a cluster level



## Application (cont.)

### 2. Clustering as a data reduction technique (cont.)

#### 2.2 Image compression



- Each data point is a pixel in the image, i.e.  $\mathbf{x} = [\text{red}, \text{green}, \text{blue}] = [x_1, x_2, x_3]$ , where  $\text{red}, \text{green}, \text{blue} \in [0, 255]$  integers
- Run clustering algorithms in this RGB color space and find  $K$  centroids
- Replace each pixel by its closest centroid
- Now we only use  $3 \times K$  unique values to represent the image instead of  $3 \times 256$  values
- In this example, with  $K = 10$  centroids, when we save the .png image, we have a reduction from 328.5 kB to 43.4 kB

# Today

- 1 Introduction
- 2 Modeling for clustering
- 3 Clustering tendency
- 4 First clustering model: K-means
- 5 Summary

# Clustering modeling

- Modeling for clustering

$$y = g(x; \theta \mid h)$$

- Clustering:

- $y$ : **categorical (nominal)**, scalar - each category is called a **cluster**
- $x$ : typically **continuous numerical**; feature vector  $\mathbf{x} = [x_1, \dots, x_d]$  (similar to classification problems in lecture 5)
- $g$ : **clustering model**, e.g. K-means, Gaussian mixture models, hierarchical clustering models, etc

There are mainly four categories of clustering models

- **Centroid clustering**
- **Distribution clustering**
- Density clustering
- Hierarchical clustering
- $\theta$  (parameters) and  $h$  (hyperparameters) depend on  $g$

## Parameter estimation

- Clustering models are **unsupervised learning** algorithms
- In unsupervised learning, the parameters are estimated from an **unlabeled data set**, that is, a data set contains only the feature vectors  $\{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ , e.g.



where  $\mathbf{x}_i$  = pixel values in a picture and the task is to group **similar** ducks into the same cluster

- **Similarity** is not well defined
- Clustering tasks do not require annotations - it is cheaper, but also more difficult because there are no predefined clusters!
- In this course, we will look at one commonly used parameter estimation technique called the **Expectation-Maximization (EM)** algorithm

# Models in this course

We are going to introduce various categories of clustering techniques; then we focus on two clustering models

- K-means
  - **Parameters:**  $K$  centroids
  - **Hyperparameters:**  $K$
  - **Parameter estimation:** an iterative method to update the centroids until convergence; this method can be interpreted as a simplified version of the Expectation-Maximization algorithm
- Gaussian mixture models
  - **Parameters:**  $K$  priors,  $K$  Gaussian likelihood (the big two!)
  - **Hyperparameters:** the number of Gaussian components  $K$
  - **Parameter estimation:** the Expectation-Maximization algorithm

# Today

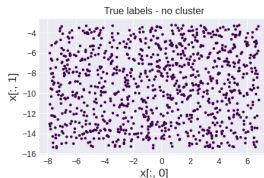
- 1 Introduction
- 2 Modeling for clustering
- 3 **Clustering tendency**
  - Are there clusters in the data?
  - Distance based approach
  - Hopkins statistic
  - Histogram based technique
- 4 First clustering model: K-means
- 5 Summary



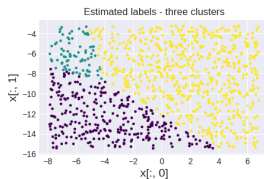
Are there clusters in the data?

# Let's try something out!

- Generate some data  $\{[x_1^1, x_1^2], \dots, [x_N^1, x_N^2]\}$  from a uniform distribution for  $i = 1, \dots, N, j = 1, 2$



- Run a clustering algorithm - go you magical beast!





# Take a step back: is the data “clusterable”?

- Do you see any clusters in the following plots?



- Figure 1: data is generated from a uniform distribution - no cluster
  - Figure 2: data is generated from three different Gaussian distributions - three clusters
  - Figure 3: data is generated from two different Gaussian distributions - two clusters
  - Figure 4: data is generated from one Gaussian distribution - one cluster
- How to decide if the data is clusterable
    - Need to define what a cluster is
    - Need to define the “null hypothesis”, i.e. the situation where there are no clusters **Note:** the “null hypothesis” is in quotes because it does not have to be described by a probabilistic distribution
  - There is no ground truth label - there are various ways of defining these prerequisites, which makes it a difficult task!
  - Now spend 30 secs staring at the plots and try to think how you can measure if the data is clusterable

# Cluster tendency

The general idea is to compare the data distribution with a theoretical distribution with no cluster tendency!

Let  $\mathbf{x}_i = [x_1^i, \dots, x_d^i]$  be a feature vector when we need to index both the dimension and the data point, we use superscript to index the data point and use subscript to index the dimension

- For example, we can make a qq-plot to compare the set  $\{x_j^1, \dots, x_j^N\}$  and a non-clusterable theoretical probability distribution, e.g. a uniform distribution



- We can repeat this for all dimensions  $j = 1, \dots, d$
- But then the question is how to aggregate all these  $d$  dimensions? - Not easy!
- Comparing distributions gets trickier when  $d > 1$ !

## Cluster tendency (cont.)

- Luckily, we have some other techniques that can help us!
- In this course, we briefly introduce the following techniques
  - Distance based technique
    - Distance measure
    - Pairwise distance
    - Dissimilarity matrix
  - Hopkins statistic
  - Histogram based technique
    - Histogram for high dimensional data

## Distance based approach

# Distance based approach

- Distance measure
  - Defines how “similar” two items are
  - The most commonly used distance is the Euclidean distance
  - Example: let  $\mathbf{x} = [x_1, x_2, x_3]$  and  $\mathbf{y} = [y_1, y_2, y_3]$  be two feature vectors, the Euclidean distance is defined as

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + (x_3 - y_3)^2}$$

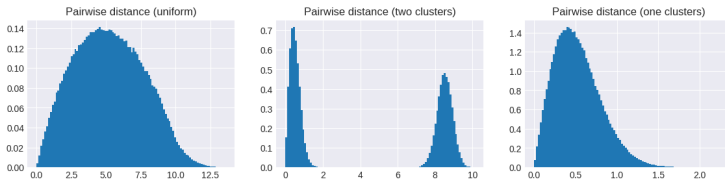
- Pairwise distance
  - Distances between all pairs of data points from two sets
  - Example: let  $\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$  and  $\{\mathbf{y}_1, \mathbf{y}_2\}$  be two sets, the pairwise distance is defined as

$$\{d(\mathbf{x}_1, \mathbf{y}_1), d(\mathbf{x}_1, \mathbf{y}_2), d(\mathbf{x}_2, \mathbf{y}_1), d(\mathbf{x}_2, \mathbf{y}_2), d(\mathbf{x}_3, \mathbf{y}_1), d(\mathbf{x}_3, \mathbf{y}_2)\}$$

- The general idea is to compare the **distribution of the pairwise distance computed from the data** to the one computed from a distribution without clustering tendency, e.g. a **uniform distribution**

## Distance based approach (cont.)

- Pairwise distance (cont.)
  - A very simplistic example



- Dissimilarity matrix
    - A matrix that contains pairwise distance  $d(\mathbf{x}_i, \mathbf{y}_j)$  on its  $(i, j)^{th}$  position
- |                                 |                                 |                                 |
|---------------------------------|---------------------------------|---------------------------------|
| $d(\mathbf{x}_1, \mathbf{y}_1)$ | $d(\mathbf{x}_1, \mathbf{y}_2)$ | $d(\mathbf{x}_1, \mathbf{y}_3)$ |
| $d(\mathbf{x}_2, \mathbf{y}_1)$ | $d(\mathbf{x}_2, \mathbf{y}_2)$ | $d(\mathbf{x}_2, \mathbf{y}_3)$ |
- It is very useful in many machine learning algorithms
  - Ordered dissimilarity matrix:** reorder the similarity matrix to group similar items together

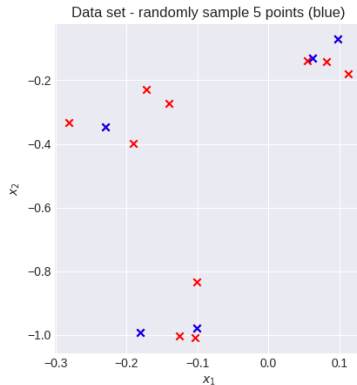
## Hopkins statistic

# Hopkins statistic for testing cluster tendency

- **Data:**  $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$  from unknown distribution
- **Null hypothesis  $H_0$ :** there is no cluster tendency in the data set
- **Test statistic  $h$ :** Hopkins statistic *Just when you thought you'd never see hypothesis testing ever again... Bam!*
- **Computation**
  - 1: Choose an integer  $M \ll N$  (sparse sampling)
  - 2: Generate a sample of uniformly distributed data with sample size  $M$ :  $\{\mathbf{y}_1, \dots, \mathbf{y}_M\}$
  - 3: Randomly choose  $M$  data points (without replacement) from  $\mathcal{X}$ :  $\{\mathbf{x}_{m_1}, \dots, \mathbf{x}_{m_M}\}$
  - 4: **for**  $i = 1$  to  $M$  **do**
  - 5: Find the **nearest neighbor of  $\mathbf{y}_i$**  in  $\mathcal{X}$ :  $\mathbf{y}$
  - 6: Compute the distance between  $\mathbf{y}_i$  and  $\mathbf{y}$ :  $u_i = \text{dist}(\mathbf{y}_i, \mathbf{y})$
  - 7: Find the **nearest neighbor of  $\mathbf{x}_{m_i}$**  in  $\mathcal{X}$ :  $\mathbf{x}$
  - 8: Compute the distance between  $\mathbf{x}_{m_i}$  and  $\mathbf{x}$ :  $w_i = \text{dist}(\mathbf{x}_{m_i}, \mathbf{x})$
  - 9: **end for**
- 10: 
$$h_0 = \frac{\sum_{i=1}^M u_i^d}{\sum_{i=1}^M u_i^d + \sum_{i=1}^M w_i^d}$$



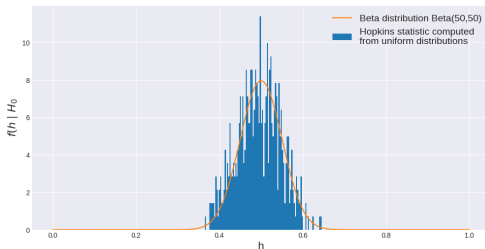
# Hypothesis testing using Hopkins statistic (cont.)



## Hypothesis testing using Hopkins statistic (cont.)

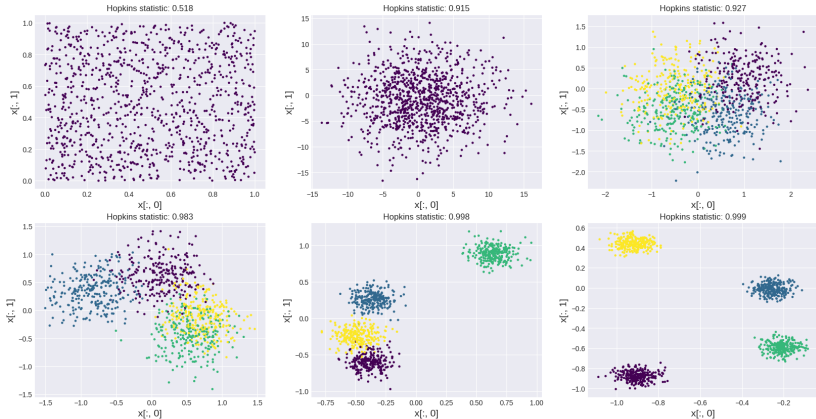
- **Null distribution:**

- PDF: Beta distribution with parameters  $a = M$  and  $b = M$
- Python: `stats.beta.pdf(x, M, M)`



- Note: there are variations of the Hopkins statistic; in general, when the Hopkins statistic deviates from 0.5 significantly, it indicates cluster tendency

# Hypothesis testing using Hopkins statistic (cont.)



## Histogram based technique

## Histogram for high dimensional data

- High dimensional histogram - empirical joint distribution

$$f_{X_1, \dots, X_d}(X_1, \dots, X_d)$$

- **Compute histogram for  $d$  dimensional data**

1: **for**  $i = 1$  to  $d$  **do**

2:   For dimension  $i$ , divide the range of data into  $n$  bins with the same size

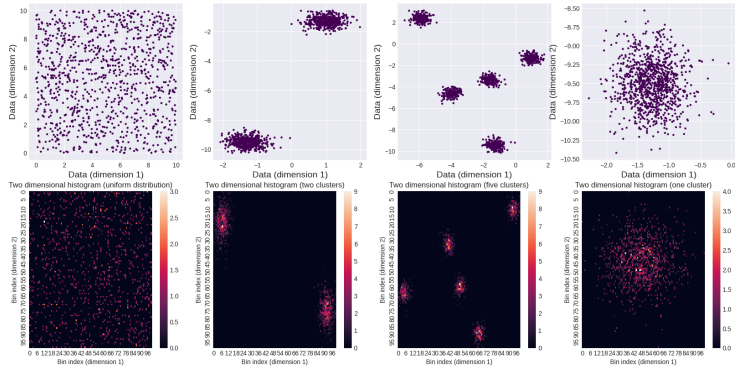
3: **end for**

4: **for**  $j = 1$  to  $n$  **do**

5:   Count the number of points in each cell  $j$  - each cell is a  $d$  dimensional cell

6: **end for**

# Histogram for high dimensional data (cont.)



# Compare two distributions using $d$ dimensional histograms

- Recall that our task here is to compare two distributions: a high dimensional data distribution and a theoretical distribution without cluster tendency, e.g. a uniform distribution - now we would like to compare this  $d$  dimensional histogram to a  $d$  dimensional theoretical distribution
- But high dimensional theoretical distribution can be hard to manipulate, for example, the area under the surface with integration is difficult
- We typically approximate high dimensional theoretical distributions using sampling techniques
- Pseudo-algorithm to illustrate the idea
  - 1: Given a data set  $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$
  - 2: Compute the  $d$  dimensional histogram for  $\mathcal{X}$
  - 3: Sample  $N$  data points from a  $d$  dimensional uniform distribution and compute the  $d$  dimensional histogram
  - 4: Compare these two histograms using, e.g. the **Kullback–Leibler divergence**

# What we have seen so far

- Definition and modeling of clustering
- Applications of clustering
  - As a preprocessing technique, e.g. summarize data, detect outliers
  - As a data reduction technique, e.g. recommender system on a cluster level, image compression
- Testing cluster tendency by comparing two distributions using 1) pairwise distance, 2) Hopkins statistic and 3)  $d$  dimensional histograms

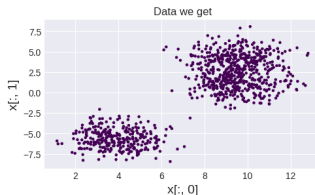


# Today

- 1 Introduction
- 2 Modeling for clustering
- 3 Clustering tendency
- 4 First clustering model: K-means**
- 5 Summary

# K-means

- **Data**  $\mathbf{x}$ :  $d$  dimensional feature vector  $\mathbf{x}$



- **Target**  $\mathbf{y}$ :

$$y = \arg \min_{k \in \{1, \dots, K\}} \text{dist}(\mathbf{x}, \boldsymbol{\mu}_k)$$

where  $\text{dist}(\cdot, \cdot)$  is a distance measure; in this course, we use the Euclidean distance (cf. page 21)

- **Parameters**:  $K$  centroids
- **Hyperparameters**:  $K$
- **Parameter estimation**: an iterative method to update the centroids until convergence
- It is a **hard clustering** technique - one data point is assigned to only one cluster

# K-means parameter estimation algorithm

- Algorithm

- **Randomly choose  $K$  centroids  $\mu_k$**  for  $k = 1, \dots, K$ , e.g. randomly choose  $K$  data points from  $\mathcal{X}$
- Repeat the two steps below until convergence, e.g.  $\mu_k$  does not change anymore

- For all  $i = 1, \dots, N$ , assign  $x_i$  to a cluster  $\hat{k}_i$  by computing

$$\hat{k}_i = \arg \min_{k \in \{1, \dots, K\}} \text{dist}(x_i, \mu_k)$$

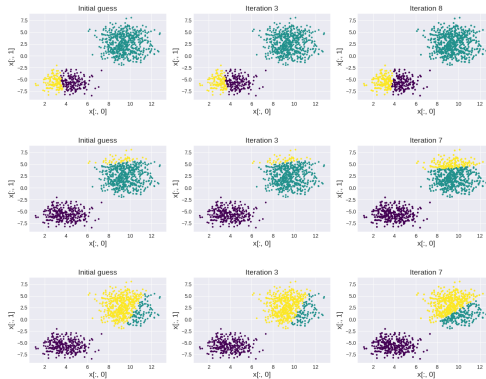
- Let  $\mathcal{X}_k$  be the set of all  $x_i$  assigned to cluster  $k$  and  $N_k$  be the size of  $\mathcal{X}_k$ , compute

$$\mu_k \leftarrow \frac{1}{N_k} \sum_{x_j \in \mathcal{X}_k} x_j$$

- There is some **randomness** in the algorithm - we should always be careful when there is randomness

# K-means initial guess

Different initializations result in different clusters



A typical solution is to run the algorithm multiple times with different initial points and aggregate the results

## K-means parameter estimation pseudocode

- 1: Given a data set  $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$
- 2: Randomly choose  $K$  data points from  $\mathcal{X}$  as the centroids  $\mu_k$  for  $k = 1, \dots, K$
- 3: **while** true **do**
- 4:   Assign  $\mathbf{x}_i$  to the closest  $\mu_k$  for all  $i = 1, \dots, N$
- 5:   For all  $k = 1, \dots, K$ , compute  $\mu_k^{new}$  as the center of all  $\mathbf{x}_i$  assigned to class  $k$
- 6:   **if**  $\mu_k^{new} == \mu_k$  for all  $k$  **then**
- 7:     **break**
- 8:   **else**
- 9:      $\mu_k \leftarrow \mu_k^{new}$
- 10:   **end if**
- 11: **end while**

# K-means: pros and cons

- Pros:
  - Convergence guaranteed
  - Easy to implement
  - Scale to large data sets
- Cons - **potential improvement**:
  - Need to choose the hyperparameter  $K$  manually - **gradually increase  $K$  and monitor the loss during parameter estimation**  
- discussed in the next lecture
  - Dependence on random initial values - **multiple initial values**
  - Do not work well on very high dimensional data - **apply dimensionality reduction techniques before clustering**
  - Not robust to outliers - **try to remove outliers before clustering**

# Today

- 1 Introduction
- 2 Modeling for clustering
- 3 Clustering tendency
- 4 First clustering model: K-means
- 5 Summary**

# Summary

So far:

- Data types and data containers
- Descriptive data analysis: descriptive statistics, visualization
- Probability distributions, events, random variables, PMF, PDF, parameters
- CDF, Q-Q plot, how to compare two distributions (data vs theoretical, data vs data)
- Modeling
- Parameter estimation: maximum likelihood estimation (MLE) and maximum a posteriori estimation (MAP)
- Classification, multinomial naive Bayes classifier, Gaussian naive Bayes classifier
- Central limit theorem, interval estimation
- Hypothesis tests, comparison of two classifiers
- Clustering, cluster tendency, k-means

Next:

- More clustering models

Before next lecture:

- Gaussian distribution
- The Bayes' rule

