

Final Project, K-Means Clustering

Song Liu (song.liu@bristol.ac.uk)

GA 18, Fry Building,

Microsoft Teams (search "song liu").

Clustering

- Clustering is a machine learning task that **groups similar objects together**.
- Given a dataset with n observations, $D := \{\mathbf{x}_i\}_{i=1}^n$, we would like to divide D into K disjoint subsets:

$$D = \cup_{i \in \{1 \dots K\}} D_i \text{ and } \cap_{i \in \{1 \dots K\}} D_i = \emptyset,$$

- Such that observations in each subset D_i **are similar** to each other.

Clustering Example

- Clustering of animals: Land vs. Sea.

$$D := \left\{ \begin{array}{ccc} \text{hippo} & \text{jellyfish} & \text{owl} \\ \text{giraffe} & \text{pig} & \text{whale} \end{array} \right\}$$

$$D_1 = \left\{ \text{hippo}, \text{giraffe}, \text{pig}, \text{owl} \right\}$$

$$D_2 = \left\{ \text{whale}, \text{jellyfish} \right\}$$

Clustering Example, 2

- Clustering of animals: Mammals vs. Non-mammals.

$$D := \left\{ \begin{array}{ccc} \text{hippo} & \text{jellyfish} & \text{owl} \\ \text{giraffe} & \text{pig} & \text{whale} \end{array} \right\}$$

$$D_1 = \left\{ \begin{array}{ccc} \text{hippo} & \text{giraffe} & \text{pig} \end{array} \right\}$$

$$D_2 = \left\{ \begin{array}{ccc} \text{owl} & \text{whale} & \text{jellyfish} \end{array} \right\}$$

Clustering Example

- The subsets found by clustering is **not** unique.
 - depending on **how the similarity is defined**, you can get different clustering results.
 - For example, I can also divide the animals in the previous case into mammals and non-mammals.
- You may want to define different types of similarities in different clustering applications.
 - For example, if you would like to group handwritten digits, you may want to consider hand writing features (directions of strokes, etc) when defining similarities between two handwritten digits.

Similarities

- In mathematics, similarities between objects are usually defined by a metric or distance function.
- One classic choice of distance is Euclidean distances.
- If two objects can be expressed as two points ***a*** and ***b*** in a *d*-dimensional Euclidean space, the Euclidean distance is

$$\text{dist}(\mathbf{a}, \mathbf{b}) := \sqrt{\sum_{i \in \{1 \dots d\}} (a_i - b_i)^2}$$

Similarities

Ronald Fisher created [Iris dataset](#), where he measured the length, width of the sepals and petals 150 iris flowers.

- In this dataset, each flower is a 4-dimensional vector.

```
> iris # load iris dataset in R by typing "iris".  
      Sepal.Length Sepal.Width Petal.Length Petal.Width  
1          5.1         3.5         1.4         0.2  
2          4.9         3.0         1.4         0.2  
3          4.7         x3.2         1.3         0.2  
...
```

- You can then measure the similarity between flowers by Euclidean distance.
 - The Euclidean distance between the first two observations is:

$$\sqrt{(5.1 - 4.9)^2 + (3.5 - 3.0)^2 + (1.4 - 1.4)^2 + (0.2 - 0.2)^2}$$

K-Means Clustering Algorithm

Given a dataset and a distance function, how to do clustering?

- **K-means** is a simple and popular choice.
- It measures the similarity between each observation and "centers" of each subset, then assigns observations to different subsets.
- After the assignments are made, it updates the centers to be the average of observations in all subsets.
- It repeatedly carries out the previous two steps until assignments do not change.

K-Means Clustering Algorithm

Suppose your dataset D contains n observations in d -dimensional space. Then K-Means divides D into D_1, \dots, D_K subsets using the following algorithm.

K-Means Clustering Algorithm

1. Randomly pick K observations from your dataset, as K centers: $\mathbf{c}_1, \dots, \mathbf{c}_K$

2. For each observation $\mathbf{x}_i \in D$,

i. For each $k \in \{1 \dots K\}$, compute the distance

$$d_{i,k} = \text{dist}(\mathbf{x}_i, \mathbf{c}_k)$$

ii. Assign \mathbf{x}_i to the subset $D_{k=k'}$, where

$$k' = \arg \min_k d_{i,k}$$

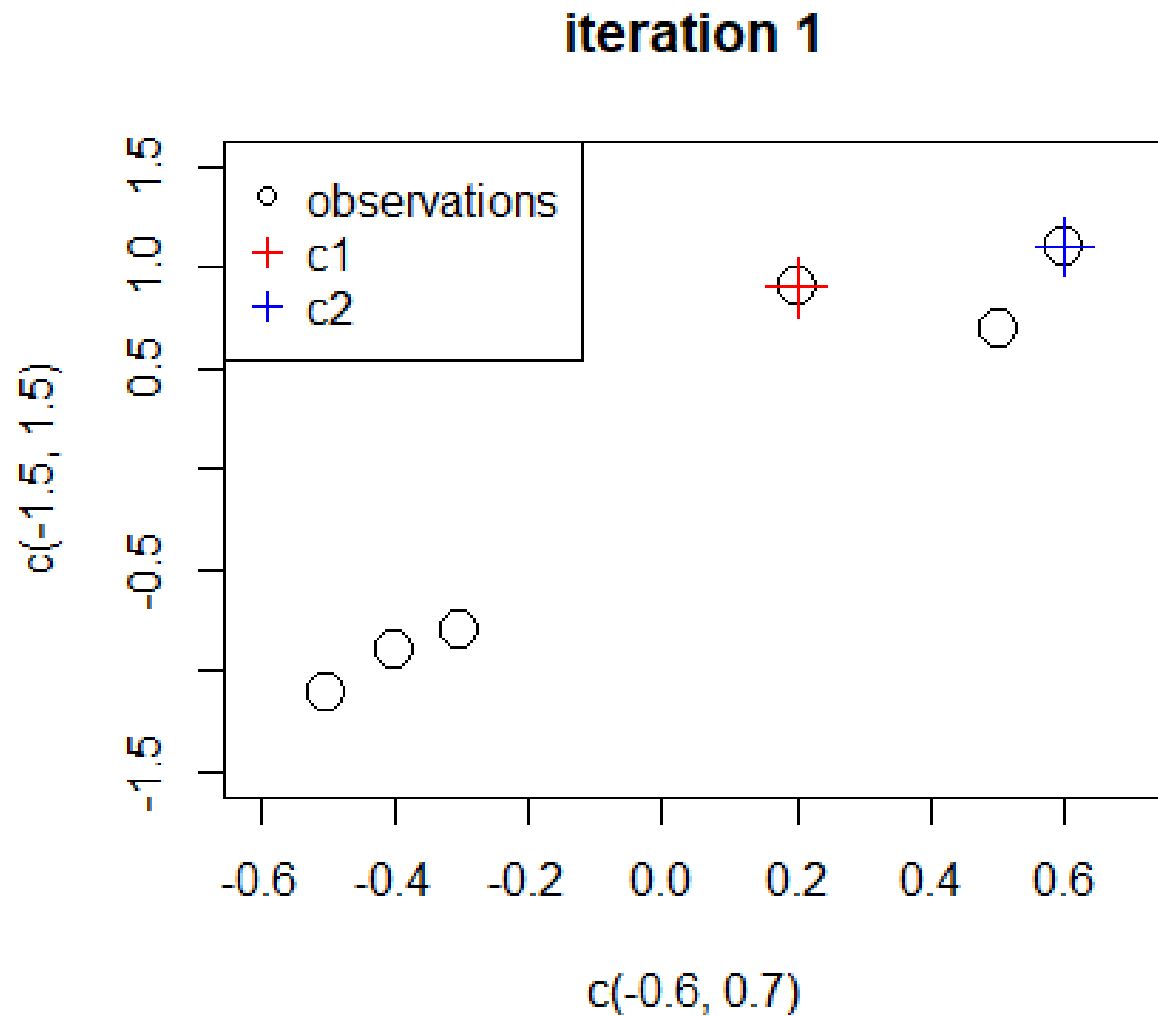
3. Compute the new centers: $\mathbf{c}_1, \dots, \mathbf{c}_K$, where

$$\mathbf{c}_k := \frac{1}{|D_k|} \sum_{\mathbf{x}_j \in D_k} \mathbf{x}_j,$$

i.e., the average of all observations in subset D_k .

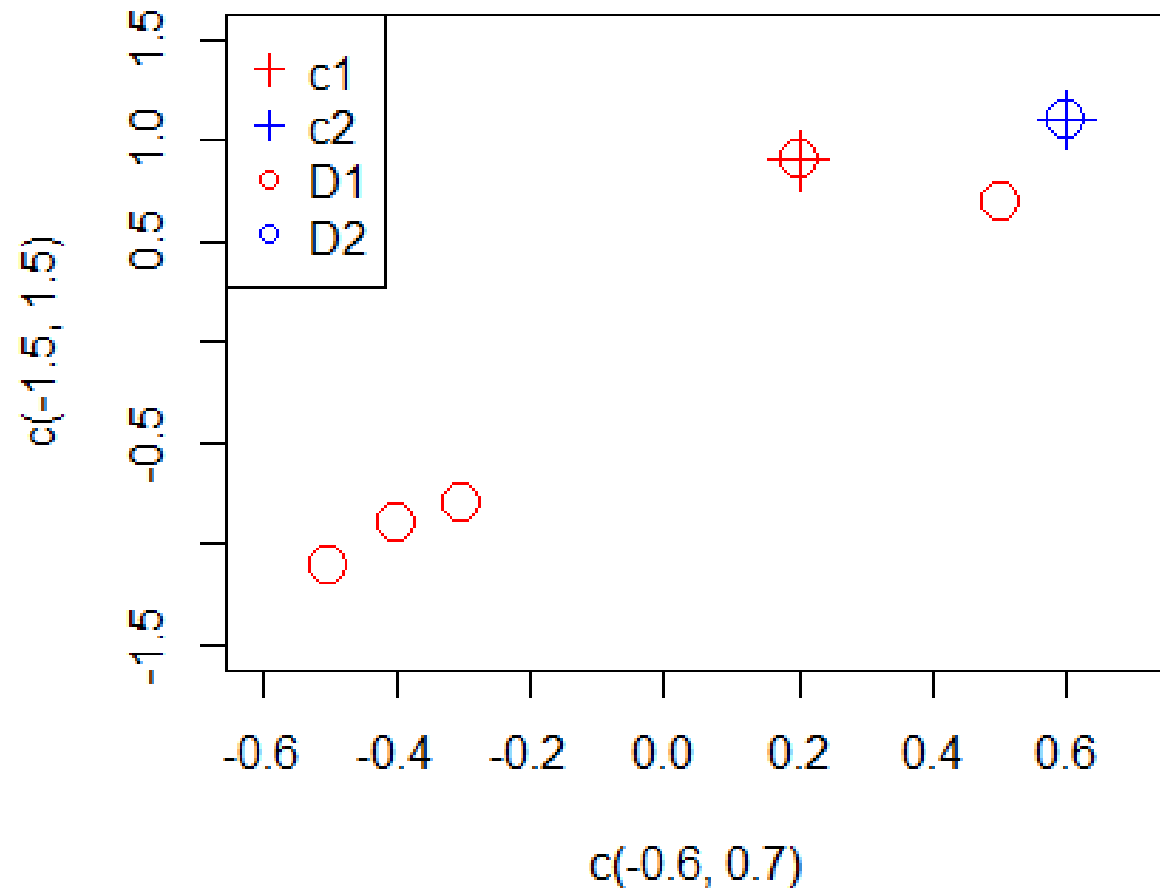
4. Repeat 2 and 3 until the assignment does not change any more.

K-Means Clustering Algorithm: Demo

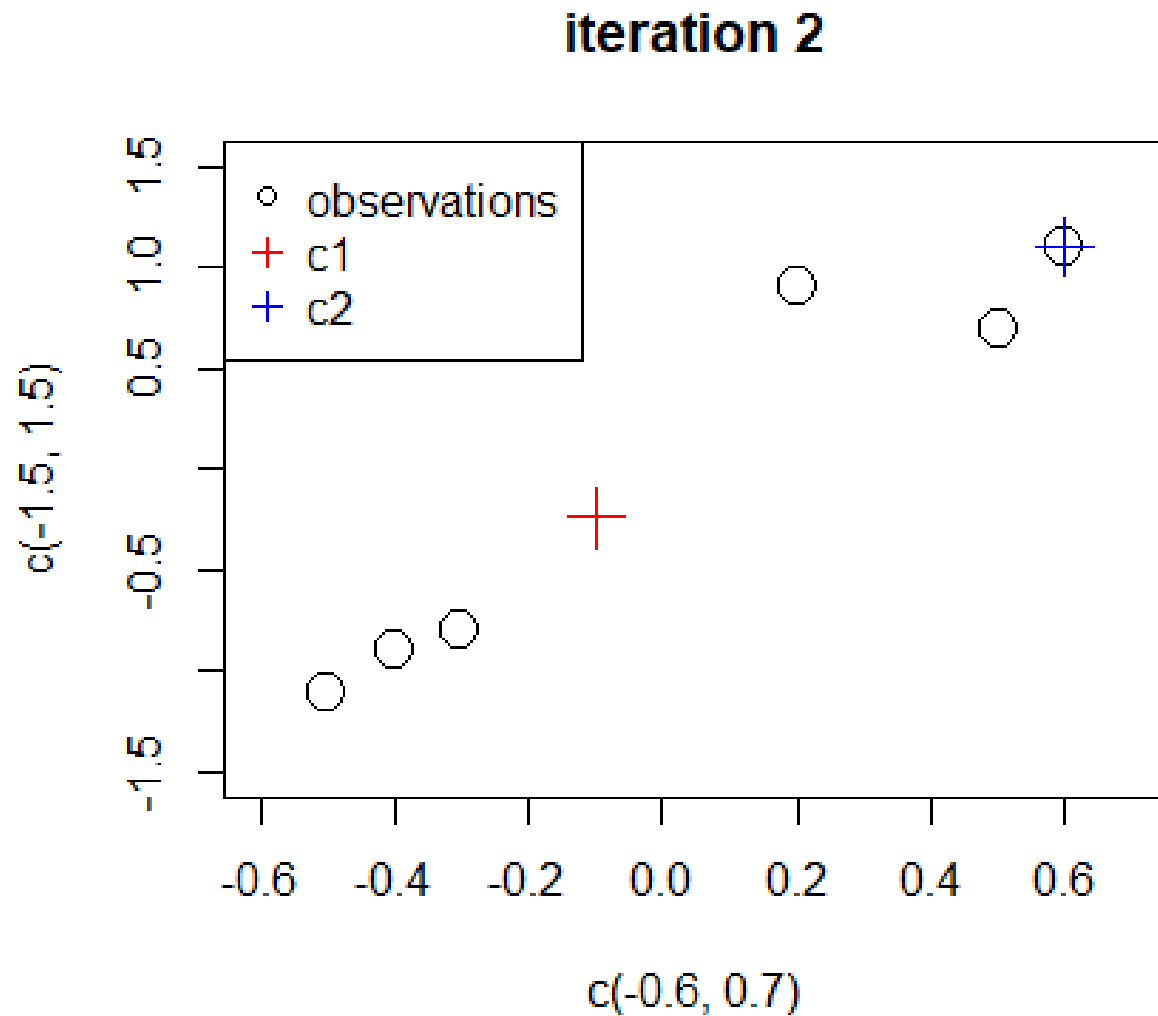


K-Means Clustering Algorithm: Demo

iteration 1 after assignment

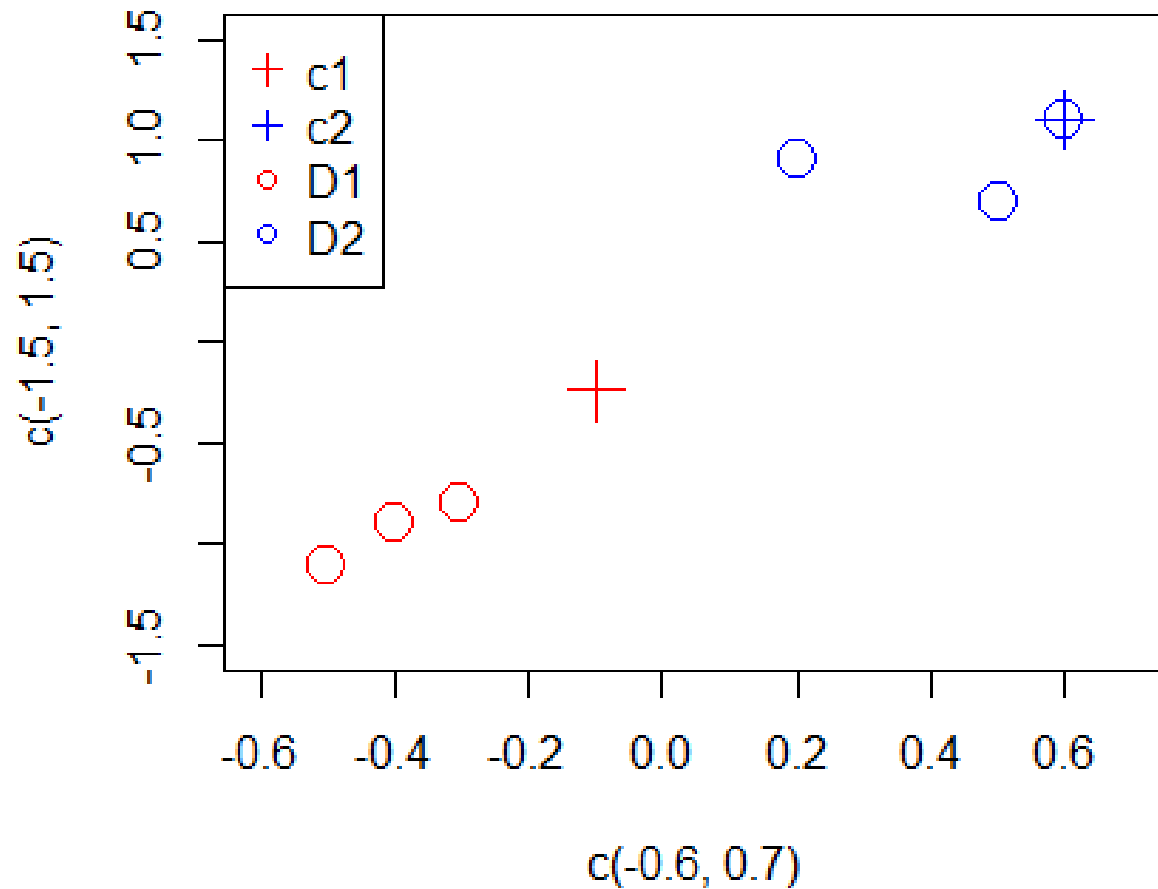


K-Means Clustering Algorithm: Demo

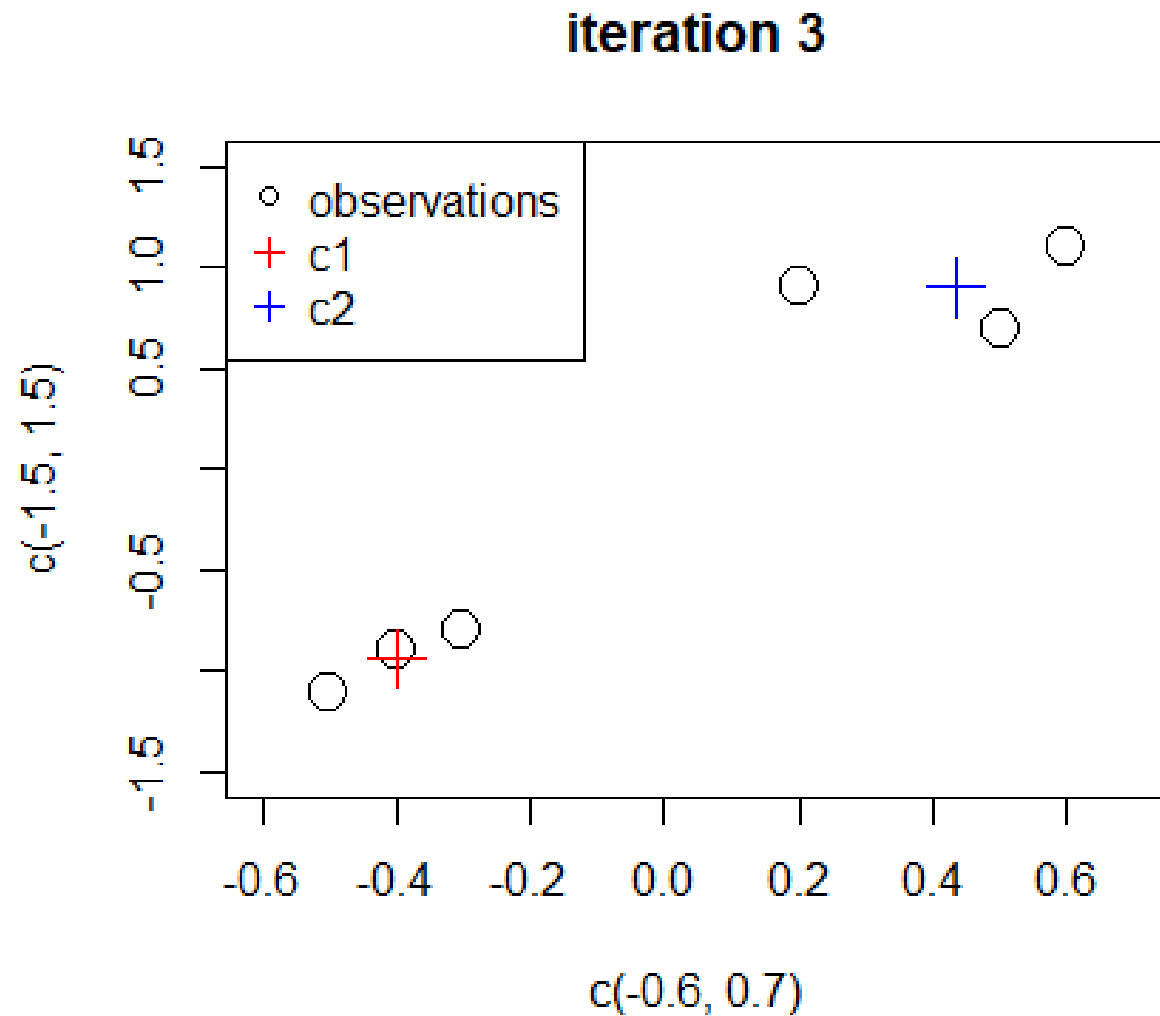


K-Means Clustering Algorithm: Demo

iteration 2 after assignment

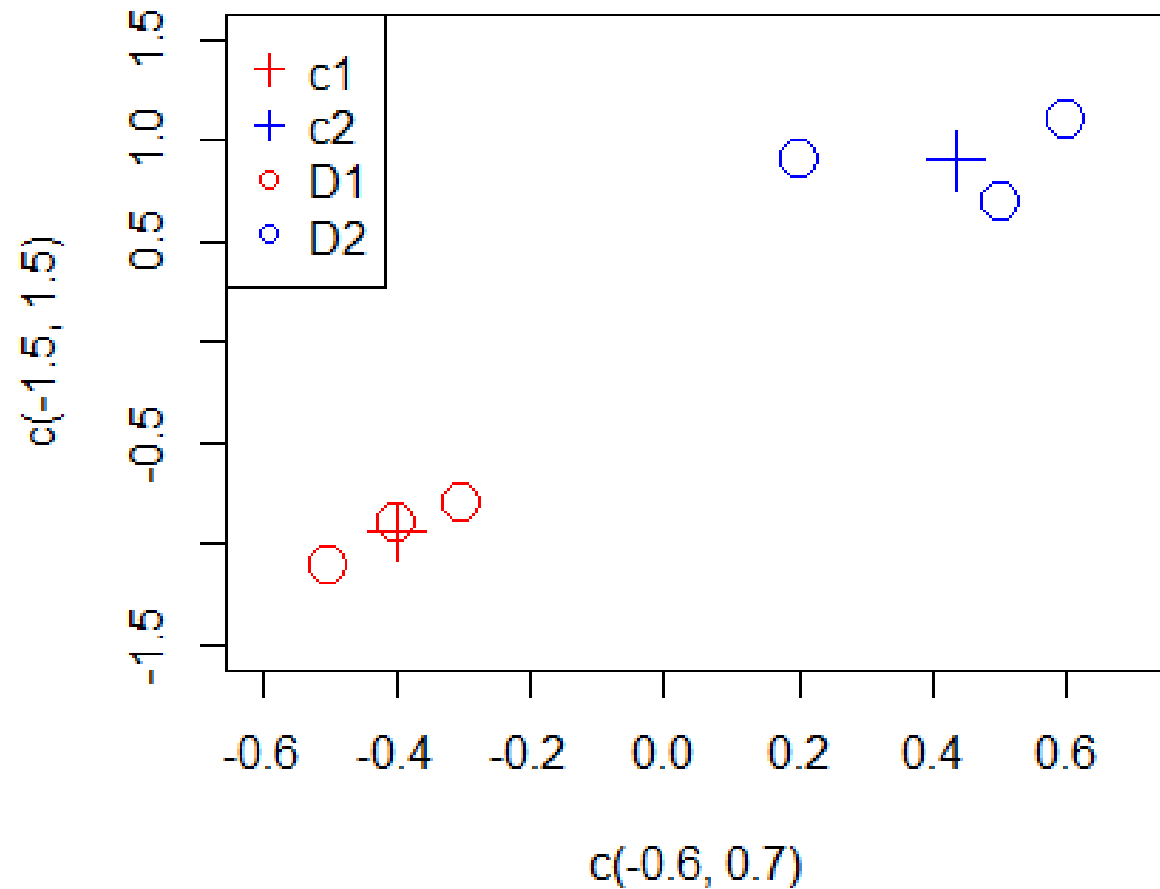


K-Means Clustering Algorithm: Demo



K-Means Clustering Algorithm: Demo

iteration 3 after assignment



K-Means Clustering Algorithm: Demo

The new centers do not lead to a different assignment comparing to iteration 2.

Stop.

Coursework: Part I (15 points)

The first part of this coursework is generating a toy dataset for our K-means algorithm. Let us assume $K = 2$ for now.

Coursework: Part I (15 points)

0. Set the random seed to 1.
1. Generate a dataset D , containing 100 random observations from two **2-dimensional** normal distribution with **different means**.
 - D can be either a matrix or data frame.
 - You should sample 50 observations from one normal distribution and 50 from the other normal distribution.
2. Create $C = \{c_1, c_2\}$ by randomly picking **2 centers** from your dataset.
 - The choice of centers needs to be random.
 - You are not allowed to specify centers yourself.

Coursework: Part I (15 points)

3. Visualize D using `points` function.

- It is up to you how to visualize your dataset.

Coursework: Part II (30 points)

- Now, let us write the K-means algorithm and test it on the toy dataset.
- Below is a list of suggested steps of writing your code. However, you can write your code differently.

Coursework: Part II (30 points)

1. Write a function `dist` that computes the distance $d_{i,k}$.
2. Write a function `assign` that assigns \mathbf{x}_i to $D_{k=k'}$.
3. Find a way to apply `assign` to all observations in D .
Obtain an 100-dimensional vector `k_prime` whose i -th component is the i -th observation's assignment k' .
4. Write a function `update_centers` that updates centers in C with the new assignments stored in `k_prime`.
5. Write a function `visualize` that visualizes $\mathbf{c}_1, \mathbf{c}_2$ and D_1, D_2 .

Coursework: Part II (30 points)

6. Write a loop that calls `assign`, `update_centers` and `visualize` repeatedly until the elements in `k_prime` do not change any more.

Coursework: Part III (15 points)

Now, let us apply K-means algorithm to a real-world dataset `iris`.

- Load iris dataset by typing `iris` and inspect the dataset.
- There are 5 variables in this dataset:
 - Sepal.Length
 - Sepal.Width
 - Petal.Length
 - Petal.Width
 - Species
- The first four variables are the properties of flowers. The fifth variable indicates the types of flowers.

Coursework: Part III (15 points)

0. Create a new R file.
1. Create a **list of 6 new datasets** from the iris dataset, by picking any pairs of variables from the first four variables.
2. Find a way to apply the K-means algorithm you previously wrote to the entire list of iris datasets and perform clustering analysis.
3. Visualize the assignments obtained from the K-means algorithm, and save your plots (6 in total) as 6 png files.

Coursework: Part III (15 points)

The following code saves a plot to the `points.png` file.

```
#create file
png("./points.png", width = 500, height = 500)
#create the plot
plot(c(-5,5),c(-5,5),type = "n")
points(rnorm(10), rnorm(10), cex = 2)
#close the file
dev.off()
```

Coursework: Part IV (20 points)

Part IV contains two slightly more difficult challenges. For each challenge, you need to create a new file.

Challenge 1:

1. You can use different distance functions in K-means algorithm. Consider the following distance functions:

- Taxi Cab distance: $\text{dist}(\mathbf{a}, \mathbf{b}) := \sum_i |a_i - b_i|$.
- Cosine distance: $\text{dist}(\mathbf{a}, \mathbf{b}) := 1 - \frac{\langle \mathbf{a}, \mathbf{b} \rangle}{\|\mathbf{a}\| \|\mathbf{b}\|}$.
- Chebyshev distance: $\text{dist}(\mathbf{a}, \mathbf{b}) := \max_i |a_i - b_i|$.

Coursework: Part IV (20 points)

2. Load [this dataset](#) to your R environment and visualize it.
3. Which distance do you think works the best for this dataset (choose from Taxi cab, Cosine, and Chebyshev)?
4. Modify your K-means code and replace the Euclidean distance with your choice. Test the new K-means algorithm on this dataset and visualize the result.

Coursework: Part IV (20 points)

Challenge 2:

- In the previous examples, we tested K-means algorithm for $K = 2$.
- Modify your K-means algorithm, so that it can divide your dataset D into arbitrary $K \geq 2$ subsets. Generate a toy dataset and test your modification.
 - K is provided to your algorithm as an input parameter.

Marking Criteria

- Part I: 15 points
- Part II: 30 points
- Part III: 15 points
- Part IV: 20 points
- Coding Style: 20 points.
 - Vectorization, Functional Programming, OOP.
 - Comments
 - Variable naming, Code formatting, etc.

DOs and DONTs

- You are encouraged to discuss this coursework with other students.
- All coursework questions should be addressed to the lecturer or TA during lab sessions or using the blackboard forum.
- You are only allowed to use the base R and `Rcpp`.
- You are not allowed to use external machine learning libraries without the permission from the lecturer.
- You are not allowed to copy other people's work.
 - You are not allowed to pass your work to other students.
- Discuss with others but **write the code by yourself!!**

Submission

- Deadline: 9th May.
- Submit a zip file containing all your R scripts (and C++ files if you use Rcpp, Rmd files if you use R markdown).
 - You do not need to submit any data file.
 - You do not need to submit any images your code generate.