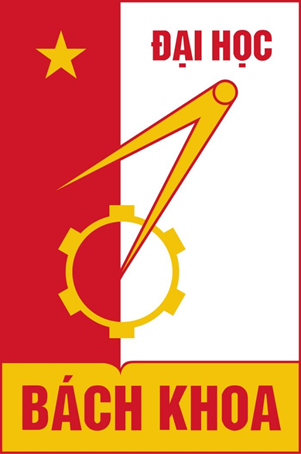
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

SCHOOL OF INFORMATION COMMUNICATION TECHNOLOGY

– – – – – – – – – – –



Machine Learning and Data Mining

*Team members:*

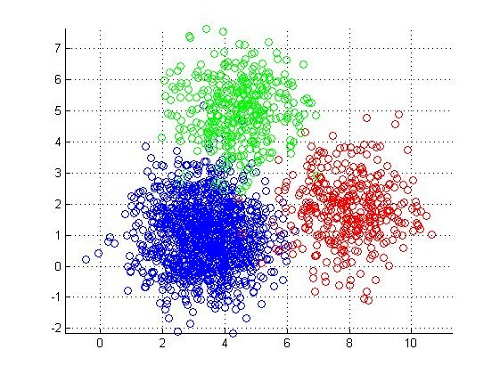
Hoang Van Phuong – 20200478

Truong Anh Huy – 20200287

*Lecturer:*

Nguyen Nhat Quang

Semester 20222



**Clustering student study results**

**MACHINE LEARNING AND DATA MINING**

**PROJECT REPORT**

1. **Introduction**

Clustering is an unsupervised machine learning technique with a lot of applications in the areas of pattern recognition, image analysis, customer analytics, market segmentation, social network analysis, and more. A broad range of industries use clustering, from airlines to healthcare and beyond.

In this project, we will show you two efficient clustering algorithms and the differences between them.

1. **Problem**

The system needs to group the students based on a number of predefined attributes (e.g., age, gender, score, the number of registered courses for the semester, etc..).

Input: a file containing a set of vectors presenting students’ study results. Each element of a vector is equivalent to an attribute of a student.

Output: group of students after being clustered.

Approach: using K-means clustering or FCM clustering

1. **Solution**
   1. **K-means**
      1. **Introduction**

K-means is a crucial algorithm that is popularly used in machine learning, particularly in clustering problems. K-means belongs to the partition-based clustering and its main target is maximizing the distance between clusters and minimizing the distance between samples that are in the same cluster.

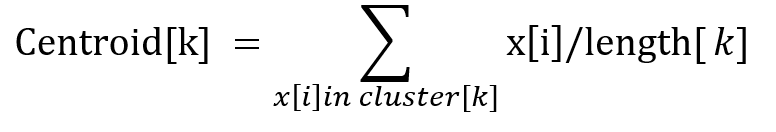
* + 1. **Algorithm**

Step 1: Choose randomly K centroids for K clusters. Each centroid represents a cluster.

Step 2: Calculate the distance between all samples of the dataset and K centroids (usually Euclidean distance)

Step 3: Assign each sample to the cluster that is nearest to it.

Step 4: Re-calculate centroids as formular:



Where:

Centroid[k]: center of cluster k-th

x[i]: sample i-th in the cluster

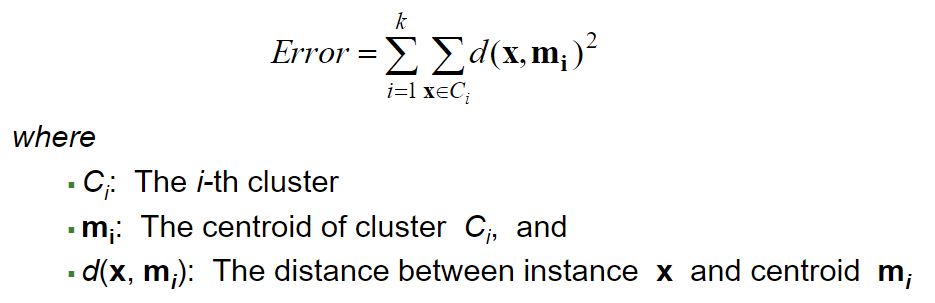
length[k]: the number of samples of cluster k-th

cluster[k]: cluster k-th

Step 5: if the algorithm converges, then stop; otherwise, go to step 2.

Convergence criterion:

* No (or insignificant) re-assignment of samples to different clusters, or
* No (or insignificant) change of centroids, or
* Insignificant decrease in the sum of error



The output of algorithm is a set of several attributes. But there are 2 features we need to focus on. The 2D array refers to the last centroids, the 1D array refers to the labels of each sample in the dataset.

Label[i] is an integer in range [0, k - 1]

Where: k is the number of clusters

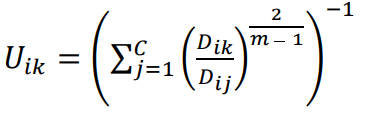
* + 1. **Summary**
       1. **Advantages**
* Easy to understand
* Easy to install and execute
  + - 1. **Disadvantage**
* Need to know the number of cluster and do not know how many clusters are optimal
* The last result depends on the random of the initial centroids
* Clusters need to have roughly the same number of points
* Clusters should be circular
* The algorithm cannot converge if there is a cluster lying inside another cluster
* Sensitive to outliers
  1. **Fuzzy-c-means (FCM)**
     1. **Introduction**

The idea of FCM is quite similar to K-means. It also belongs to the partition-based clustering and based on the separation among samples.

* + 1. **Algorithm**

Step 1: Choose randomly K centroids for K clusters. Each centroid represents a cluster.

Step 2: Calculate the membership matrix as the formular:



Where:

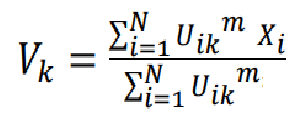
Uik: the characteristic value identifying element xi belonging to cluster k

Dik: the distance between xi and cluster k

Dij: the distance between xi and cluster j

m: the fuzzy parameter

Step 3: Re-calculate centroids as the formular:



Step 4: Repeat step 2, 3 until ‖ 𝑉 (𝑙) − 𝑉 (𝑙+1) ‖ < 𝜀

When the program stops, we receive:

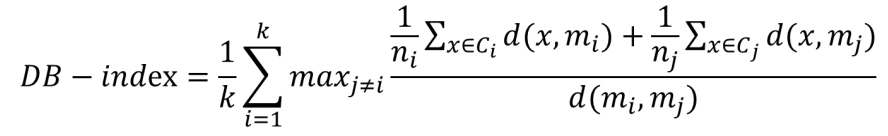
* A 2D array refers to the centroids
* A 2D array refers to the characteristic value identifying each sample belonging to each cluster. From this 2D array, we use function argmax () to choose the index of the largest number in each row. This sample will be assigned to the cluster with this index
  + 1. **Summary**
       1. **Advantages**

Improve the quality of clustering comparing to K-means due to the complex of formulars

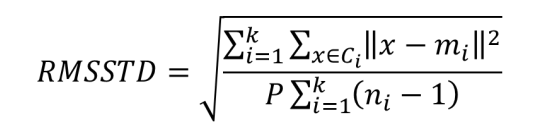
* + - 1. **Disadvantages**
* Need to know the number of cluster and do not know how many clusters are optimal
* The last result depends on the random of the initial centroids
* Sensitive to outliers
* The formulars are complex

1. **Program results**
   1. **Evaluation metrics**

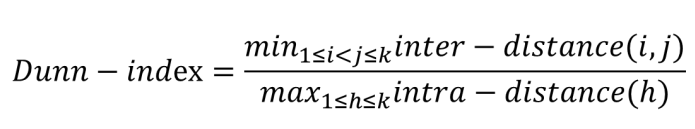
* Davies-Bouldin index: expected that the Davies-Bouldin index is as small as possible!



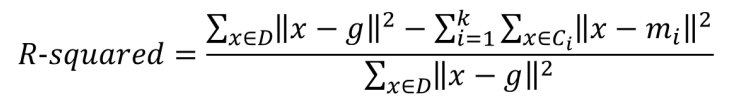
* + k: The number of clusters
  + ni, mi: The number of examples and the centroid of cluster i
  + nj, mj: The number of examples and the centroid of cluster j
  + d (mi, mj): The distance between the 2 cluster centroids mi and m
* Root-mean square standard deviation: expected that the RMSSTD value is as small as possible!



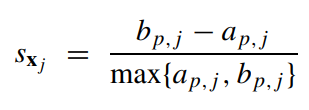
* + k: The number of clusters
  + ci: Cluster i
  + mi: The center (centroid) of cluster Ci
  + P: The number of dimensions (i.e., the number of attributes) used to represent examples
  + ni: The number of examples in cluster C
* Dunn index: expected that the Dunn index is as large as possible!



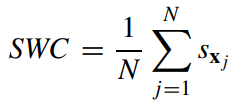
* + k: The number of clusters
  + inter-distance (i, j): The distance between the 2 clusters i and j
  + intra-distance(h): The distance (dissimilarity) between the examples of cluster h
* R-squared: expected that the R-squared value is as large as possible!



* + k: The number of clusters
  + ci: Cluster i
  + mi: The center (centroid) of cluster Ci
  + D: The entire set of examples
  + g: The center (centroid) of the entire set of examples
* Silhouette width criterion: this index is expected to be pointed out when it is maximized



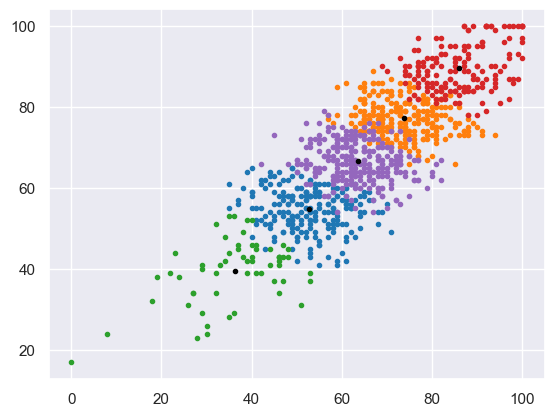
* + dq, j: the average distance of this sample to all samples in another cluster q
  + bp, j: the minimum dq, j computed over q = 1, …, k, q ≠ p
  + ap, j: the average distance of this object to all other objects in cluster p



where N is the number of samples in the dataset

* 1. **Results**
     1. **K-means**

Data after clustering without any techniques



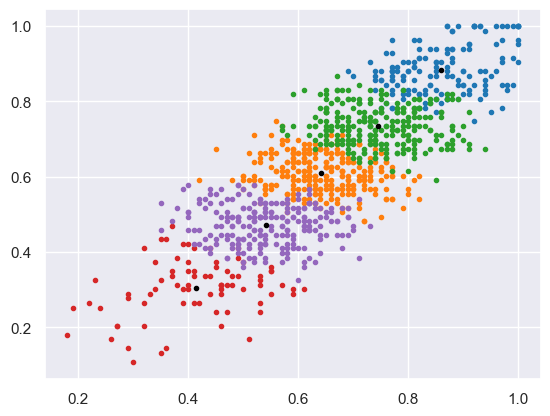
* + Evaluations:
    - Davies–Bouldin index: 0.9320
    - RMSSTD (Root-mean-square standard deviation): 6.1066
    - Dunn index: 0.0154
    - R-squared: 0.8347
    - Silhouette with criterion: 0.3320

Data after clustering with normalization (using min-max scalar)



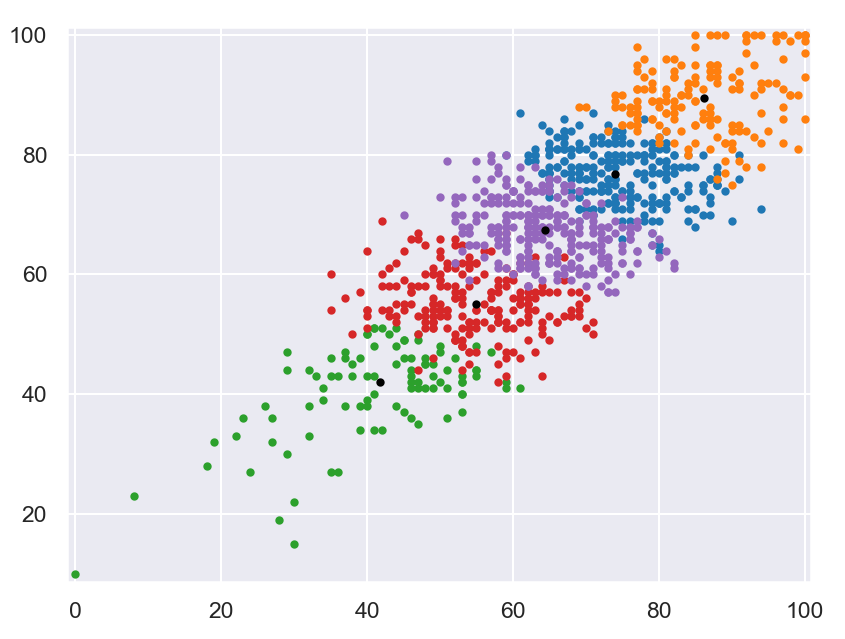
* + Evaluations:
    - Davies–Bouldin index: 0.9137
    - RMSSTD (Root-mean-square standard deviation): 0.0658
    - Dunn index: 0.0215
    - R-squared: 0.8429
    - Silhouette with criterion: 0.3388

Data after clustering with normalization and removing outliers



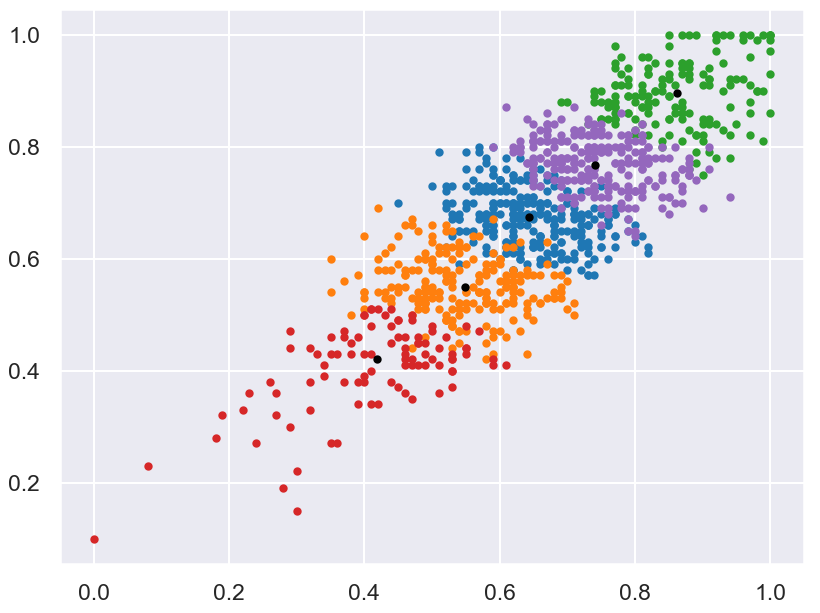
* + Evaluations:
    - Davies–Bouldin index: 0.9233
    - RMSSTD (Root-mean-square standard deviation): 0.0639
    - Dunn index: 0.0213
    - R-squared: 0.8455
    - Silhouette with criterion: 0.3351
    1. **FCM**

Data after clustering without any techniques



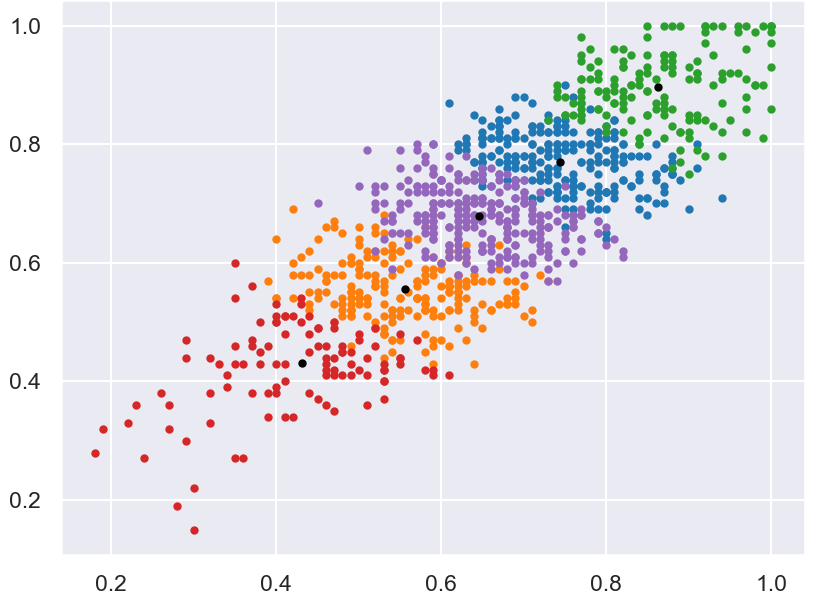
* + Evaluations:
    - Davies–Bouldin index: 0.9284
    - RMSSTD (Root-mean-square standard deviation): 6.1071
    - Dunn index: 0.0155
    - R-squared: 0.8347
    - Silhouette with criterion: 0.3326

Data after clustering with normalization (using min-max scalar)



* + Evaluations:
    - Davies–Bouldin index: 0.9154
    - RMSSTD (Root-mean-square standard deviation): 0.0658
    - Dunn index: 0.0191
    - R-squared: 0.8429
    - Silhouette with criterion: 0.3383

Data after clustering with normalization and removing outliers



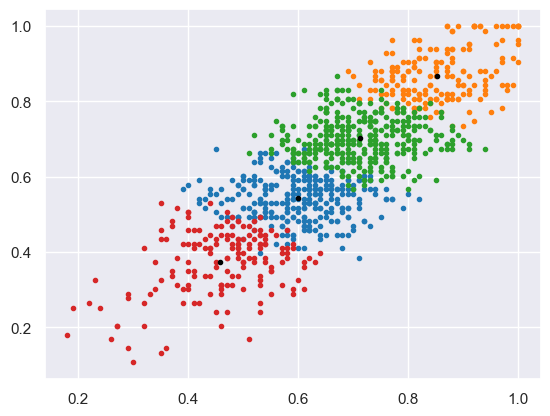
* + Evaluations:
    - Davies–Bouldin index: 0.9241
    - RMSSTD (Root-mean-square standard deviation): 0.0639
    - Dunn index: 0.0273
    - R-squared: 0.8455
    - Silhouette with criterion: 0.3352

1. **Comparation**

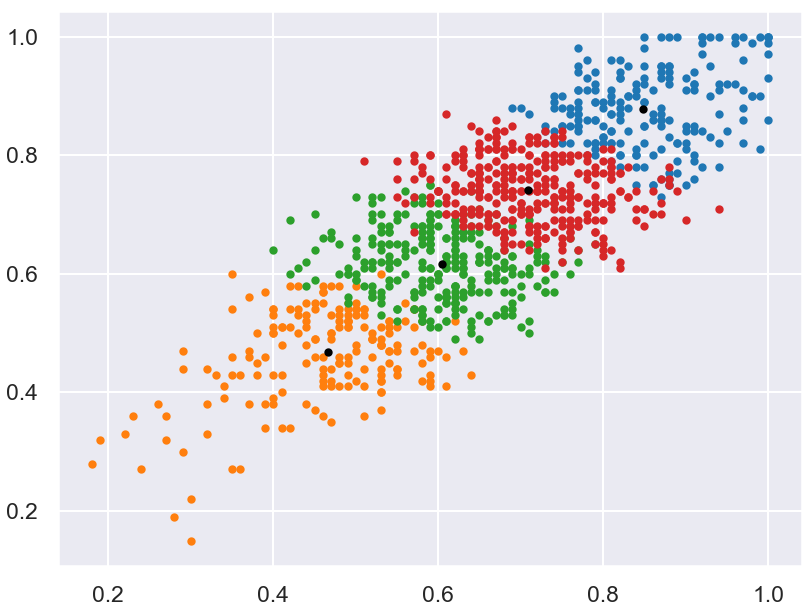
With 4 clusters:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Davies-Bouldin index | Root- mean square standard deviation | Dunn index | R squared | Silhouette width criterion |
| K-means | 0.8631 | 0.0707 | 0.0196 | 0.8108 | 0.3585 |
| FCM | 0.8975 | 0.0661 | 0.0170 | 0.8011 | 0.3477 |

K-means:



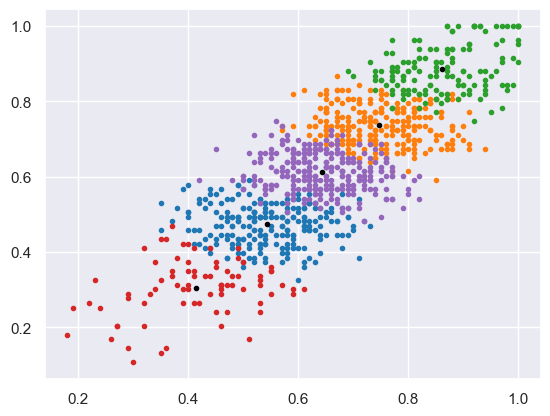
FCM:



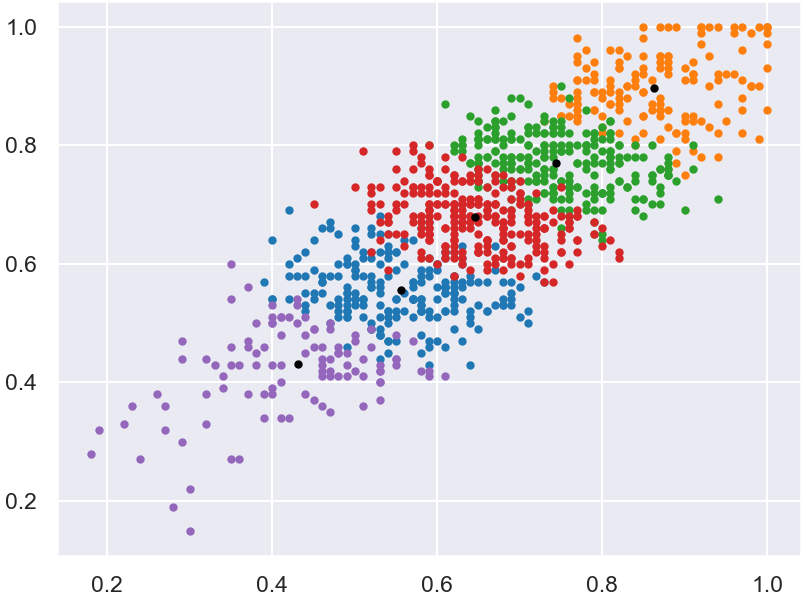
With 5 clusters:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Davies-Bouldin index | Root- mean square standard deviation | Dunn index | R squared | Silhouette width criterion |
| K-means | 0.9241 | 0.0639 | 0.0273 | 0.8455 | 0.3352 |
| FCM | 0.9835 | 0.0602 | 0.0182 | 0.8348 | 0.3138 |

K-means:



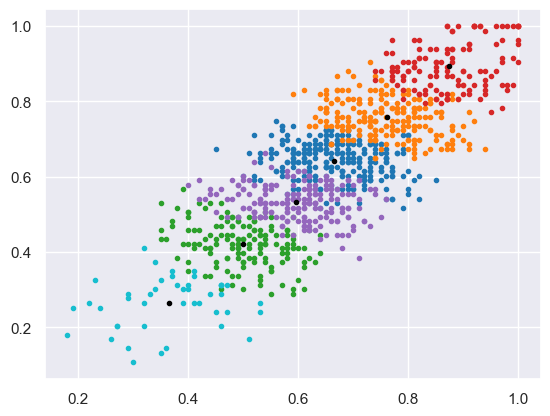
FCM:



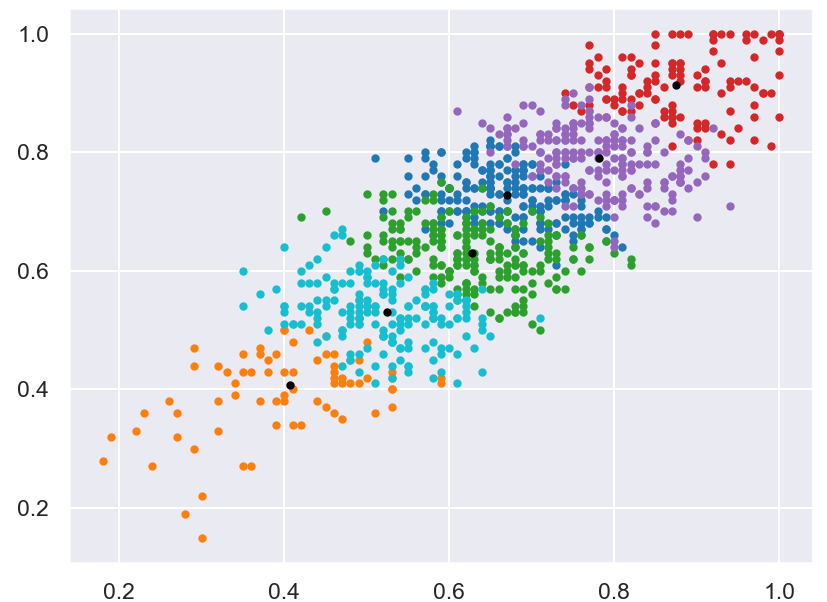
With 6 clusters:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Davies-Bouldin index | Root- mean square standard deviation | Dunn index | R squared | Silhouette width criterion |
| K-means | 1.0006 | 0.0602 | 0.0257 | 0.8630 | 0.3002 |
| FCM | 1.0628 | 0.0567 | 0.0218 | 0.8543 | 0.2811 |

K-means:



FCM:



Comments:

* With this dataset, both algorithms give acceptable results. Although the elements in the same cluster are close, the clusters are not too far from each other.
* In 2D pictures, we can see that the clusters are overlapped. This is because we run the algorithms with several dimensions so in 2 random dimensions, this happens.
* About the evaluation index, although FCM is installed from the idea of K-means but the calculation formulars in order to have better results of clustering, in this project, we see that the results of FCM is not really greater than K-means.
* This is because in certain cases, if the data does not exhibit fuzzy boundaries or if the assumptions of FCM are violated, the result of FCM can be worse than the results of K-means.

**References**

douglasrizzo (2022) dunn-sklearn.py [source code] <https://gist.github.com/douglasrizzo/cd7e792ff3a2dcaf27f6>

Lucas Vendramin, Ricardo J. G. B. Campello∗ and Eduardo R. Hruschka (2010) Relative Clustering Validity Criteria: A Comparative Overview

Machine Learning cơ bản (2017) K-means clustering

<https://machinelearningcoban.com/2017/01/01/kmeans>

Yufeng (2021) Fuzzy C-Means Clustering with Python

<https://towardsdatascience.com/fuzzy-c-means-clustering-with-python-f4908c714081>