National and Kapodistrian University of Athens

Department of Computer Science

Software Development for Algorithmic Problems

K23γ

Clustering Algorithms over vectors and polygonal curves

STUDENT ID :
sdi1600151
sdi1700115

Contents

1 ABSTRACT

In this project, the above Clustering algorithms have been implemented: Initialization {Random Selection - K-Means++} Assignment {Simplest - By Range Search with LSH for vectors} Update {PAM - Mean Vector - DTW Centroid Curve}

2 Directories and Files

Our implementation code is derived in the following directories:

- app: includes utilization functions called from all interfaces
  - includes: includes utilization libraries
  - src: includes the implementation of utilization functions
- config: includes configuration file
- core: includes functions called from all interfaces
  - cluster
    - clustering: includes cluster's class implementation. Cluster Class combines all 8 algorithms implemented.
    - initialization: includes the implementation of initialization algorithms.
    - assignment: includes the implementation of assignment algorithms.
    - update: includes the implementation of update algorithms.
    - search: includes the implementation of functions commonly used by different search algorithms in different interfaces.
    - utils: includes the implementation of interfaces' utilisation.
  - hash: includes hash function's and amplified hash function's implementation
  - metric: includes the implementation of different metrics (ex:DTW,Manhattan, Euclidean)
  - search: includes the implementation of functions commonly used by different search algorithms in different interfaces.
  - utils: includes the implementation of interfaces' utilisation.
- datasets: includes curves' and vectors' dataset and query files.

- results: includes search algorithms' results
- scripts: includes compilation script for each interface
- unit_testing: includes gtests implementations

3 Compilation and Running

Clustering for Vectors

Navigate to **scripts** file, then run **sh run_vectors_clustering.sh**. Modify compilation scripts in case you want to change arguments. Argument Parameters:

- -i input file
- -c configuration file
- -o output file
- –complete
- –init initialization algorithm
- –assign assignment algorithm
- –update update algorithm

Default hyperparameters: $init = k - means + +, assign = lloyd, update = pam$.

Clustering for Curves

Navigate to **scripts** file, then run **sh run_curves_clustering.sh**. Modify compilation scripts in case you want to change arguments. Argument Parameters:

- -i input file
- -c configuration file
- -o output file
- –complete
- –init initialization algorithm
- –assign assignment algorithm
- –update update algorithm

Default hyperparameters: $init = k - means + +, assign = lloyd, update = pam$.

4 Implementation Details

General

- All arguments are validated. In case of no arguments, user is asked to provide them from input.
- All d-dimensional points are stored in in 1D vector allongside with their offsets and ids, gaining a better performance.
- Metric: Euclidean Distance.

Initialization

Vectors

1. Random Initialization: Choose randomly n = number of clusters vectors and set them as centers.
2. ParkJunk: Uniformly set n = number of clusters vectors as centers. Compute distances of closest non center vectors to each center and get a probabilistic value (prob). Set as new center the one with minimum prob. Repeat until no changes occur.

Curves

1. Random Initialization: Choose randomly n = number of clusters curves and set them as centers.
2. ParkJunk: Uniformly set n = number of clusters vectors as centers. Compute distances of closest curves to each center and set new center the one closer to previous centers given by a probability.

Assignment

Vectors

1. Simplest: Assign each vector to its closer center, given by Euclidean Distance. Compute also the sum of distances of all vectors assigned to each cluster from its center and set it as cost of assignment.
2. Range-LSH: Hashing all vectors into LSH Stractures. Perform Range NN to assign vectors to nearest clusters.

Curves

1. Simplest: Assign each curve to its closer center, given by DTW Distance. Compute also the sum of distances of all curves assigned to each cluster from its center and set it as cost of assignment.
2. Range-LSH: Hashing all curves into LSH Stractures. Perform Range NN to assign vectors to nearest clusters.

Update

Vectors

1. Pam: For each vector in cluster compute distances between all vectors in same cluster. Set as center the vector with minimum sum of distances.
2. Mean: Generate a new vector by calculating the average value of the same points of all vectors assigned in a specific cluster and set it as new center.

Curves

1. Pam: For each curve in cluster compute distances between all curves in same cluster. Set as center the curve with minimum sum of distances.
2. Mean: Generate a new curve of λ length by calculating the average distance of best pair of points of all curves assigned in a specific cluster and set it as new center.

5 Results

5.1 Vectors

5.1.1 Random - LLoyd's - PAM

- Input
  - dataset: DataVectors_5_500x100.csv
  - number of clusters: 5
  - max iterations: 100
- Output
  - Silhouette: 0.457086

5.1.2 K-means++ - Lloyd's - PAM

- Input
  - dataset: DataVectors_5_500x100.csv
  - number of clusters: 5
  - max iterations: 100
- Output
  - Silhouette: 0.903341

5.1.3 Random - Range-LSH - Pam

- Input
  - dataset: DataVectors_5_500x100.csv
  - number of clusters: 5
  - max iterations: 100
- Output
  - Silhouette: 0.02532

5.1.4 K-Means++ - Range-LSH - Pam

- Input
  - dataset: DataVectors_5_500x100.csv
  - number of clusters: 5
  - max iterations: 100
- Output
  - Silhouette: 0.03478

5.1.5 Random - Lloyd's - Mean

- Input
  - dataset: DataVectors_5_500x100.csv
  - number of clusters: 5
  - max iterations: 100

- Output
    - Silhouette: 0.40373

5.1.6 K-Means++ - Lloyd's - Mean

- Input
    - dataset: DataVectors_5_500x100.csv
    - number of clusters: 5
    - max iterations: 100
- Output
    - Silhouette: 0.903341

5.1.7 Random - Range-LSH - Mean

- Input
    - dataset: DataVectors_5_500x100.csv
    - number of clusters: 5
    - max iterations: 100
- Output
    - Silhouette: 0.100672

5.1.8 K-Means++ - Range-LSH - Mean

- Input
    - dataset: DataVectors_5_500x100.csv
    - number of clusters: 5
    - max iterations: 100
- Output
    - Silhouette: 0.199499

5.2 Curves

5.2.1 Random - LLoyd's - PAM

- Input
  - dataset: input_projection6.csv
  - number of clusters: 6
  - max iterations: 10
- Output
  - Silhouette: 0.490926

5.2.2 K-means++ - Lloyd's - PAM

- Input
  - dataset: input_projection6.csv
  - number of clusters: 6
  - max iterations: 10
- Output
  - Silhouette: 0.626284

5.2.3 Random - Range-LSH - Pam

- Input
  - dataset: input_projection6.csv
  - number of clusters: 2
  - max iterations: 10
- Output
  - Silhouette: 0.994

5.2.4 K-Means++ - Range-LSH - Pam

- Input
  - dataset: input_projection6.csv
  - number of clusters: 6

- max iterations: 10

- Output
  - Silhouette: 0.996

5.2.5 Random - Lloyd's - Mean

- Input
  - dataset: input_projection6.csv
  - number of clusters: 3
  - max iterations: 10
- Output
  - Silhouette: 0.496017

5.2.6 K-Means++ - Lloyd's - Mean

- Input
  - dataset: input_projection6.csv
  - number of clusters: 6
  - max iterations: 10
- Output
  - Silhouette: 0.666667

5.2.7 Random - Range-LSH - Mean

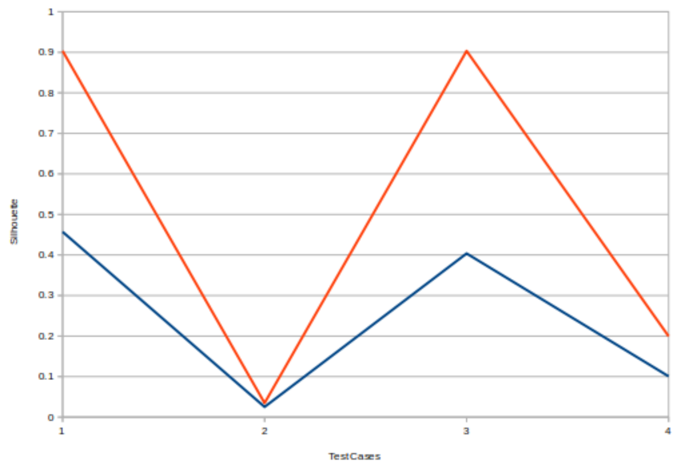
- Input
  - dataset: input_projection6.csv
  - number of clusters: 2
  - max iterations: 10
- Output
  - Silhouette: 0.86

5.2.8 K-Means++ - Range-LSH - Mean

- Input
  - dataset: input_projection6.csv
  - number of clusters: 2
  - max iterations: 10
- Output
  - Silhouette: 0.97

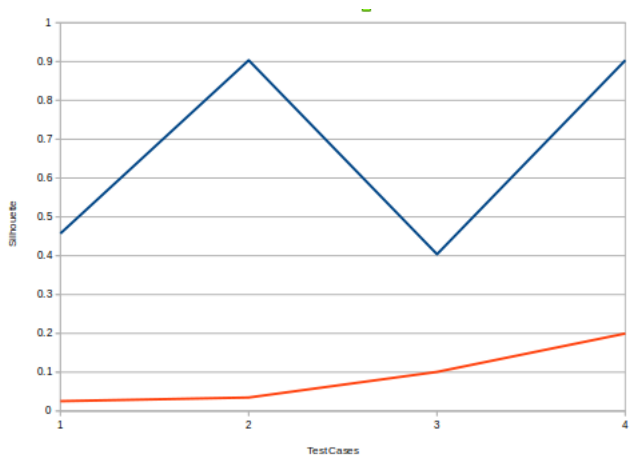
6 Conclusion

6.0.1 Random Selection vs K means++



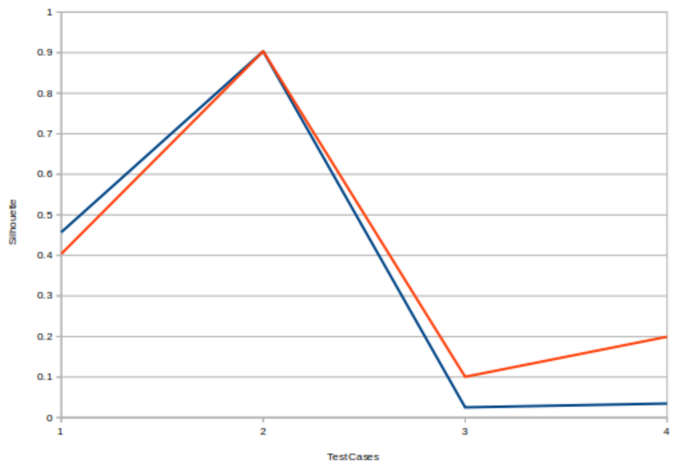
Regarding the above results we conclude that K-means++ initializes clusters better than random selection though it is slightly slower.

6.0.2 Lloyd's Approach vs Range-LSH



In addition, Lloyd's approach for assignment has slightly better results than Range-LSH Assignment but it lacks performance.

6.0.3 PAM vs Mean



To conclude, PAM update algorithm has almost the same results as Mean Vector / Curve computation. Results depend on the other arguments used as well as the dataset. Nevertheless, Mean computation algorithm performs better than PAM algorithm.