HOMEWORK #1

Software Project (0368-2161)

Due Date: 11/05/2021

1 Introduction

The K-means algorithm is a popular clustering method for finding a partition of N unlabeled observations into K distinct clusters, where K is a parameter of the method. In this assignment you will implement this algorithm in both Python and C. The goals of the assignment are:

- Practice the material taught in class both for C and Python.
- Transform a known algorithm into working executable code.
- Read input stream and process it.
- Create an interface for programs.
- Experience the difference in the programming effort and the running time of both languages.

1.1 K-means

Given a set of N datapoints $x_1, x_2, \ldots, x_N \in R^d$, the goal is to group the data into K clusters, each datapoint is assigned to exactly one cluster and the number of clusters K is such that $K < N$. We will denote the group of clusters by $S_1, S_2, \ldots, S_K$, each cluster S_j is represented by it's centroid which is the mean $\mu_j \in R^d$ of the cluster's members.

Algorithm 1 k-means clustering algorithm

```
Initialize centroids $\mu_1, \mu_2, \ldots, \mu_K$ as first $k$ datapoints $x_1, x_2, \ldots, x_K$
repeat
    Assign $x_i$ to the closest cluster $S_j$:
        $$\underset{S_j}{\text{argmin}}\ (x_i - \mu_j)^2,\ \forall j\ 1 \leq j \leq K$$
    update all centroids as follows:
        $$\mu_i = \frac{\sum_{x_l \in S_i} x_l}{|S_i|}$$
until convergence: ($no\ change\ in\ \mu$) $OR$ ($iteration\_number = max\_iter$)
```

2 Assignment Description

Implement the k-means algorithm as detailed in 1.1 both in C and Python.
The behavior of the program is as follows:

1. The program receives the arguments: K, filename and optional(max_iter):
   (a) K – the number of clusters required.
   (b) filename - *.txt file that contains datapoints separated by commas.
   (c) max_iter – the maximum number of iterations of the K-means algorithm, if not provided the default value is 200.
2. The final centroids are returned to the cmd.

2.1 Compile and Running

2.1.1 C

The program must compile cleanly (no errors, no warnings) when running the following command:

```
$gcc -ansi -Wall -Wextra -Werror -pedantic-errors kmeans.c -lm -o hw1
```

After successfully running the above command, an executable file called hw1 will be created. Now you can run your program by executing the following line on Nova(assuming K = 3, max_iter = 100, filename = "input.txt"):

```
$./hw1 3 100 < input.txt
```

See example below:

```
$#providing max_iter=100
$./hw1 3 100 < input.txt
-4.2435,9.1568,5.4105,9.6870,-5.7564,-7.2314
3.3226,-1.3896,-9.1927,-6.0907,-0.9954,-8.7412
8.2239,-8.5714,-8.4985,0.8969,-8.2158,-2.3753
```

```
$#not providing max_iter
$./hw1 3 < input.txt
-4.2435,9.1568,5.4105,9.6870,-5.7564,-7.2314
3.3226,-1.3896,-9.1927,-6.0907,-0.9954,-8.7412
8.2239,-8.5714,-8.4985,0.8969,-8.2158,-2.3753
```

2.1.2 Python

Your program must be executed by (no errors, no warnings) the following line on Nova(assuming K = 3, max_iter = 100, filename = "input.txt"):

```
$python3 kmeans.py 3 100 < input.txt
```

See examples below:

```
$#providing max_iter=100
$python3 kmeans.py 3 100 < input.txt
-4.2435,9.1568,5.4105,9.6870,-5.7564,-7.2314
3.3226,-1.3896,-9.1927,-6.0907,-0.9954,-8.7412
8.2239,-8.5714,-8.4985,0.8969,-8.2158,-2.3753
```

```
$#not providing max_iter
$python3 kmeans.py 3 < input.txt
-4.2435,9.1568,5.4105,9.6870,-5.7564,-7.2314
3.3226,-1.3896,-9.1927,-6.0907,-0.9954,-8.7412
8.2239,-8.5714,-8.4985,0.8969,-8.2158,-2.3753
```

2.2 Assumptions and requirements:

Note that the following list applies to both programs in this assignment:

1. You may assume that the input file is in the correct format and that it is supplied.
2. Validate that the command line arguments are in correct format.
3. Outputs must be formatted to 4 decimal places (no rounding).
4. In the C implementation, you should use assert to exit on any error in allocating memory.
5. You may not import external includes (in C) or modules (in Python) that are not mentioned in this document.
6. Learn about and use input() and split() in your Python implementation.
7. 3 input files and their corresponding output files examples will be provided within the assignment in Moodle.
8. Make your code (C and Python) as efficient as possible, you'll notice that the Python's runtime becomes increasingly slow as the number of observations, clusters and dimensions grow, to measure the runtime, use the "time" command in Linux. **There is no requirement to output your runtimes**.

2.3 Submission

1. Please submit a file named id1_id2_assignment1.zip via Moodle, where id1 and id2 are the ids of the partners.
   (a) In case of individual submission, id2 must be 111111111
   (b) Create the zip file using **only** linux cmd:
   ```
   $zip -r id1_id2_assignment1.zip your_directory_name
   ```
2. The zipped file must contain **only** the following files (at the root, no folders):
   (a) kmeans.c – Your source code for the C program.
   (b) kmeans.py – Your source code for the Python program.

2.4 Remarks

For any question regarding the assignment, please post at the HW_1 discussion forum.