# Lab 7: command line interface for clustering analytics using K-means

In this assignment, you are asked to implement the K-means clustering algorithm. A clustering problem can be formulated as: Let $X = \{x_1, x_2, x_3, \ldots, x_n\}$ be the set of data points and $V = \{v_1, v_2, v_3, \ldots, v_K\}$ be the set of centers. We need to find the optimal values of these centers in accordance with some distance metric (e.g. Euclidean distance) such that the distances between points within a single cluster are minimized, while the distances between points in different clusters are maximized. This can be achieved by using the K-means algorithm which can be described as follows:

```
Randomly select V (a set of cluster centers) from X.

for N iterations
```

1. Using some distance metric calculate the distance between each data point and cluster centers.
2. Assign each data point to the closest cluster center.
3. Recalculate the new cluster centers using the following formula

$$v_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$$

where $C_i$ represents the cluster (a set of data points) with the center $v_i$

```
Return cluster_ids
```

The initial skeleton of the algorithm is provided in lab7.py file and you required to use the following parameters to complete your implementation:

- `input_file`: the name of the file containing a dataset. The file should be in the same folder as `lab7.py` .
- `features`: a list of features used for clustering. For example: " -f 0,2,3 " would ask the program to use features with implicit indices 0, 2 and 3. The default option should be "all".
- `number_of_clusters`: the number of clusters. The default option should be 3.
- `distance_metric`: the id of a distance metric. The default option should be 2.

| ID | Distance metric |
|---|---|
| 0 | Cosine |
| 1 | Manhattan (city block) |
| 2 | Euclidean |
| 3 | Minkowski (p = 3) |

- `random_seed`: a random number generator seed. The default option should be 777.
- `number_of_iterations`: the number of iterations (N) to run the algorithm. The default option should be 10.
- `output_file`: a file where the final `cluster_ids` of data points should be saved. The values should be separated by a comma "," and have the same order as their corresponding data points in the input file.

These parameters should be provided to the program as command line arguments in the following way:

```
lab7.py -i input_file -f features -k number_of_clusters -d distance_metric
-s random_seed -n number_of_iterations -o output_file
```

We recommend to use `argparse` (https://docs.python.org/3/library/argparse.html) to parse input arguments.

You are allowed to use any built-in functions to calculate the distance.

# Lab 7: command line interface for clustering analytics using K-means

You can write your program in any IDE of your choice. We would recommend using the community version of Pycharm. Use local functions to encapsulate your code into smaller subroutines when you implement the execution for each of the options. This will help you to improve readability of your code, and thus, to elevate debugging your code.

To assess your implementation we ask you to perform clustering on Iris dataset. You should provide the following values to the parameters: `number_of_clusters` = 3, `number_of_iterations` = 10 and `random_seed` = 111. As you have seen, the dataset contains 4 features. We ask you to explore all possible two-feature combinations using all four distance metrics.

For example, if you would like to explore the combination of the first two features using the Manhattan distance, then your command could be:

```
lab7.py -i iris.csv -f 0,1 -k 3 -d 1 -s 111 -n 10 -o out.txt
```

To determine the best two-feature combination and distance metric, compute the purity score after each experiment. The purity score is a commonly used evaluation metric for clustering algorithms.
(Check the following links to understand the purity score:
https://nlp.stanford.edu/IR-book/html/htmledition/evaluation-of-clustering-1.html,
https://stats.stackexchange.com/questions/95731/how-to-calculate-purity)

In addition, your program should make a 2D scatter plot of the data, such that each data point is colored in accordance with its cluster and the cluster centers are distinctively marked. The axes should be labeled in accordance with the selected features and the legend should explain the cluster colors and ids. This plot should be generated at every iteration to show the progression of the algorithm. The title should contain the current number of iteration, the current purity score and the used distance metric. As a result, you will have 10 plots for 10 iterations.

You should submit the 10 plots corresponding to the experiment with the highest purity score. Save the plots as pdf files and merge them into a single pdf file before submission.

Here is a link to an overview of K-means:
https://www.analyticsvidhya.com/blog/2019/08/comprehensive-guide-k-means-clustering/