CS373: Homework 4

Due Date: 11:59 PM, Nov 15, 2019

Instructions for submission: In this programming assignment you will implement the k-means clustering algorithm. Instructions below detail how to turn in your code and assignment on `data.cs.purdue.edu`

You are given a skeleton code with an existing folder structure. Do not modify the folder structure. You may add any extra files you want to add in the `cs373-hw4/src/` folder. For all other parts write your answers in a single PDF. Name this `report.pdf`. Place your report in the `cs373-hw4/report/` folder. Label all the plots with the question number. Your homework must contain your name and Purdue ID at the start of the file. If you omit your name or the question numbers for the plots, you will be penalized.

Failure to follow the instructions along with the format specifications in this handout will result in a loss of points.

To submit your assignment, log into `data.cs.purdue.edu` (physically go to the lab or use ssh remotely) and follow these steps:

1. Place all of your code in the `cs373-hw4/src/` folder and your report in `cs373-hw4/report/` folder.
2. Change directory to outside of `cs373-hw4/` folder (run `cd ..` from inside `cs373-hw4/` folder)
3. Execute the following command to turnin your code: `turnin -c cs373 -p hw4 cs373-hw4`
4. To overwrite an old submission, simply execute this command again.
5. To verify the contents of your submission, execute this command: `turnin -v -c cs373 -p hw4`. Do not forget the `-v` option, else your submission will be overwritten with an empty submission.

1 Specification

1.1 Dataset

In this homework, you will be working with the 'Yelp' dataset. It contains 19 attributes: 15 diescrete and 4 continuous (**{latitude, longitude, reviewCount, checkins}**). Your task for this homework is to implement the k-means algorithm and apply it to the continuous attributes in the data.

Your submission will be run on a hidden dataset which is different from given dataset. The hidden dataset will have the same column names for the continuous attributes as the given dataset.

The features you will use are the 4 continuous attributes in `yelp3.csv`.

1.2 Skeleton Code

You are provided a skeleton structure with this homework:

```
cs373-hw4/
├─ handout.pdf
└─ data/
   └─ given/
      └─ yelp3.csv
```

```
│
├── src/
│   ├── __init__.py
│   └── kmeans.py
└── report/
    └── report.pdf
```

Do not modify the given folder structure. You should not need to, but you may, add any extra files of code in `cs373-hw4/src/` folder. You should place your report (`report.pdf`) in the `cs373-hw4/report/` folder. You **must not** modify `__init__.py`. All your coding work for this homework must be done inside `cs373-hw4/src/` folder. You are not allowed to use any external libraries for classifier implementations such as `scikit-learn` etc.

Your code will be tested using `python3` from inside the `cs373-hw4/src/` folder. Make sure that you test your code on `data.cs.purdue.edu` before submitting.

1.3 Expected Output

Your python script should take three arguments as input:

1. `trainingDataFileName`: corresponds to a subset of the data that should be used as the training set for your algorithm.
2. `K`: the value of k to use when clustering.
3. `clustering option`: takes one of the following six values, 1 (use the four original attributs for clustering, which corresponds to Q3.1), 2 (apply a log transform to reviewCount and checkins, which corresponds to Q3.2), 3 (use the standardized four attributes for clustering, which corresponds to Q3.3, 4 (use the four original attributes and Manhattan distance for clustering, which corresponds to Q3.4), 5 (use 3% random sample of data for clustering, which corresponds to Q3.5), and 6 (use the improved score function, which corresponds to Q3.6).

Your code should read in the training sets from the csv file, cluster the training set using the specified value of `k`, and output the within-cluster sum of squared error and cluster centroids. For the centroid of each cluster report the values for each of the four attributes in the following order.
{latitude, longitude, reviewCount, checkins}

The expected output is given below. Note that this will run your algorithm with the `yelp.csv` file, a K value of 4, and clustering option 1. Please make sure you follow this output else you **WILL** loose points. Note that this is only a sample output and the numbers are not representative of the actual results.

```
$ python kmeans.py yelp.csv 4 1
WC-SSE=15.2179
Centroid1=[49.00895,8.39655,12,3]
...
CentroidK=[33.33548605,-111.7714182,9,97]
```

2 Kmeans (25 points)

2.1 Theory (10 points)

1. (2 points) What are some limitations of using k-means algorithm?
2. (2 points) Can you always discover structure in data using k-means algorithm? Does the algorithm always generate meaningful clusters (e.g., in uniformly distributed data, in binary data)?

3. (2 points) What is the theoretical time complexity of the algorithm? Discuss how you can parallelize the algorithm to improve the complexity.
4. (2 points) Consider dataset X with p discrete, q continuous attributes and 1 binary class label. Imagine you learnt cluster memberships for each of the data instances using the k-means algorithm on the q continuous attributes. Let approach A represent the Naive Bayes classifier learned only with the p discrete features, and approach B represent the NBC model using the p discrete features and a new cluster membership feature (learnt using k-means). Discuss how the accuracy (or 0/1-loss or squared loss) of approach A is expected to be compared to that of approach B.
5. (2 points) Improve the score function. To evaluate the clustering, it is not sufficient to measure only the within-cluster sum of squares (wc) defined above. It is also desired to have each cluster separate from others as much as possible. To improve the resulting clustering, define your own score function that takes into account not only the compactness of the clusters but also the separation of the clusters. Write a formal mathematical expression of your score function and explain why you think your score function is better than the within-cluster sum of squares.

2.2 Implementation (15 Points)

You need to implement the k-means clustering algorithm for this part. This part could be completed by editing only `kmeans.py`. You need to follow the description of the models discussed in the lecture slides (link) with the following specifications.

Features: Consider the 4 continuous attributes in `yelp.csv` for **X**.

Distance: Use Euclidean distance unless otherwise specified.

Score function: Use within-cluster sum of squared error (where r_k is the centroid of cluster C_k, d is the distance function.).

$$wc(C) = \sum_{k=1}^{K} \sum_{x(i) \in C_k} d(x(i), r_k)^2 \tag{1}$$

Make sure to also implement the following cluster options as described in Section 1.4.

1. The four original attributes for clustering (corresponding to 3.1).
2. A log transform to reviewCount and checkins (corresponding to 3.2).
3. Standardize the 4 attributes for clustering (corresponding to 3.3).
4. Four original attributes and Manhattan distance for clustering (corresponding to 3.4).
5. A random sample of the data for clustering (corresponding to 3.5).
6. Use your improved score function from Theory Question 5 (corresponding to 3.6)

Report the results obtained on the given train set in your report.

3 Analysis (30 points)

You only need to include your plots and discussions in your report. Make sure that the code you submit doesn't include any changes you don't want to be included.

1. (5 points) Cluster the Yelp data using k-means.

(a) Use a random set of examples as the initial centroids.

(b) Use values of K = [3,6,9,12,24].

(c) Plot the within-cluster sum of squares (wc) as a function of K.

(d) Choose an appropriate K from the plot and argue why you choose this particular K.

(e) For the chosen value of K, plot the clusters with their centroids in two ways: first using `latitude` vs. `longitude` and second using `reviewCount`, `checkins`. Discuss whether any patterns are visible.

2. (5 points) Do a log transform of `reviewCount`, `checkins`. Describe how you expect the transformation to change the clustering results. Then repeat the analysis (1). Discuss any differences in the results.

3. (5 points) Transform the four original attributes so that each attribute has mean = 0 and stdev = 1. You can do this with the numpy functions, numpy.mean() and numpy.std() (i.e., subtract mean, divide by stdev). Describe how you expect the transformation to change the clustering results. Then repeat the analysis (1). Discuss any differences in the results.

4. (5 points) Use Manhattan distance instead of Euclidean distance in the algorithm. Describe how you expect the change in the clustering results. Then repeat the analysis (1). Discuss any differences in the results.

5. (5 points) Take a 6% random sample of the data. Describe how you expect the downsampling to change the clustering results. Then run the analysis (i) five times and report the average performance. Specially, you should use a single random 6% sample of the data. Then run 5 trials where you start k-means from different random choices of the initial centroids. Report the average wc when you plot wc vs. K. For your chosen K, determine which trial had performance closest to the reported average. Plot the centroids from that trial. Discuss any differences in the results and comment on the variability you observe.

6. (5 points) Improved score function. In this case, you will use the score function you proposed in Theory (2.1) question 5. Using the best configuration from Questions 1-5, plot the results of your score function for K = [3, 6, 9, 12, 24], and compare the results to the appropriate algorithm from Question 1-5.

4 Time Limit

Your code must terminate with in 10 minutes for each clustering model with 4 or less clusters. If it doesn't terminate with in 10 minutes, you will be graded for the output your code generates at the end of the 10 minute time limit. If your code doesn't converge by then, it would be a good idea to print the results you have at that point or use a different convergence criteria.