INF 553 – Fall 2018

Assignment 4: Clustering Yelp Reviews (Due: 11/16/2018)

Assignment Overview

In this assignment, you will experiment with several clustering techniques including K-Means, Bisecting K-Means, and Gaussian Mixture Models (GMM). Unsupervised data mining methods are fairly diverse. At one extreme, there are the pure statistical summarization techniques that are the mainstay of exploratory data analysis, somewhere in between are data transformations like principal components analysis, which have a deeper mathematical underpinning but a fairly mechanical formulation, and at the other extreme you have unsupervised models that make strong assumptions about the data distribution. Clustering methods fall into that latter category: they make an assumption about instance similarity given attributes and use this modeling assumption to provide an interpretable division of a dataset.

Environment Requirements

Python: 2.7 Scala: 2.11 Spark: 2.3.1

Coding

Do not share code with other students in the class!!
Here's why:

- The most obvious reason is that it will be a huge temptation to cheat: if you include code written by anyone else in your solution to the assignment, you will be cheating. As mentioned in the syllabus, this is a very serious offense, and may lead to you failing the class.
- However, even if you do not directly include any code you look at in your solution, it surely will influence your coding. Put another way, it will short-circuit the process of you figuring out how to solve the problem and will thus decrease how much you learn.

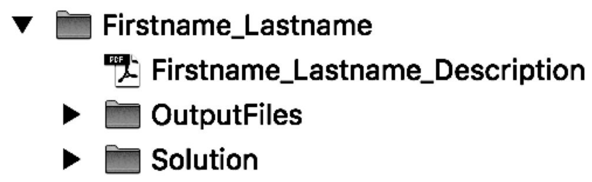
So, just don't look on the web for any code relevant to these problems. Don't do it.

For Problem 1 you must implement the K-Means algorithm. For Problem 2 and 3 you can use Spark APIs (https://spark.apache.org/docs/2.3.1/mllib-clustering.html)

Submission Details

For this assignment you will need to turn in a Python or Scala program depending on your language of preference. We will test your code using the same datasets but with different parameters. This assignment will surely need some time to be implemented so please plan accordingly and start early!
Your submission must be a .zip file with name: **<Firstname>_<Lastname>_hw4.zip**. The structure of your submission should be identical as shown below. The Firstname_Lastname_Description.pdf file contains helpful instructions on how to run your code along with other necessary information as described in the following sections. The *OutputFiles* directory contains the deliverable output files for each problem

and the *Solution* directory contains your source code.


Firstname_Lastname
Firstname_Lastname_Description
OutputFiles
Solution

Algorithm

Note: You need to develop the algorithm only for Problem 1. For Problem 2 and 3 you can use existing implementations.

In this assignment, you will look at three different clustering algorithms on the Yelp Dataset Challenge Round 12. You will cluster the reviews data. For the first problem you will implement K-Means algorithm and submit the code. For Problems 2 and 3, you will run K-Means, Bisecting K-Means and Gaussian Mixture Models based clustering. You can use existing libraries for this part of the assignment.

Note this dataset is different from the one used in last assignment. One of the challenges in data mining is preparing the data, extracting features, and presenting results. A significant portion of this assignment focusses on those aspects.

The dataset was prepared by extracting the text from reviews in Yelp Challenge Dataset Round 12. All special characters were removed from the reviews by running `sub('\W+', '',word)` and then the words were converted to lower case. Only reviews with more than 50 words were selected and one review is written per line. The small dataset (yelp_reviews_clustering_small) contains 1000 reviews. The large dataset (yelp_reviews_clustering_large) contains 100K reviews.

Each review should be treated as a document. You will use word count (only in Problem 1) and Term Frequency – Inverse Document Frequency (TF-IDF) features (remember, this was discussed in the class). The distance measure will be Euclidean Distance. For random seed or state use 20181031.

You can write your program in Python or Scala. For this assignment you will need to cluster the reviews in the Yelp dataset with which you are already familiar with. First you need to develop K-Means clustering algorithm and run it on **yelp_reviews_clustering_small** (Problem 1). Next you need to run K-Means, bisecting K-Means, and GMM clustering on **yelp_reviews_clustering_small** (Problem 2) dataset, and **yelp_reviews_clustering_large** (Problem 3) for large dataset.

At the end of the assignment, Appendix A specifies how to organize your Description pdf file.

Problem 1 (25 Points)
Develop K-Means Clustering Algorithm

You will implement the K-Means algorithm with 1) Word count and 2) TF-IDF features from reviews using Euclidean distance measure. The algorithm should take an input file with reviews, feature, number of clusters, and maximum number of iterations. For each cluster you need to identify the top ten frequently occurring features. Note these are the terms corresponding to the top ten values in the centroid vector.

Execution Requirements

Input Arguments:

1. **Input.txt:** This is the path to the input reviews file. Every line contains a word and reviews are separated by '---'. For Problem 1 you can use the *Data/yelp_reviews_clustering_small.txt* file to test the correctness of your algorithm.

2. **Feature=W**: W –for word count, T – for TF-IDF

3. **N=5:** Number of clusters.

4. **Iterations =20**: Maximum number of iterations

Output:

You should output the sizes of the clusters followed by the top 10 frequent terms in the cluster for each algorithm and feature type. The output should be in a JSON file in the format shown in the snapshot of the Execution Example section below. There should be 2 output files – one for Word Count and another one TF-IDF based clustering.

Execution Example

Following we present examples of how you can run your program with spark submit both when your application is a Java/Scala program or a Python script.

A. **Example of running a Java/Scala application with spark-submit:**
Notice that the argument class of the spark-submit specifies the main class of your application and it is followed by the jar file of the application.

```
$SPARK_HOME/bin/spark-submit  --class  Task1  Firstname_Lastname_KMeans.jar
<input_file> <feature> <N> <Iteration>
```

Example of running a Python application with spark-submit:

```
$SPARK_HOME/bin/spark-submit   Firstname_Lastname_KMeans.py   <input_file>
<feature> <N> <Iteration>
```

<u>Example output</u>

```
{
      "algorithm":"K-Means",
      "WSSE":0.52,
      "clusters":[ {
           "id":1,
           "size":50,
           "error":0.25,
           "terms": ["restaurant","food","good"…]
      },
….

}
```

<u>Deliverables for Problem 1</u>

1. **Script or Jar File and Source Code**
   Please name your Python script as: <firstname>_<lastname>_KMeans.py.
   Or if you submit a jar file as: <firstname>_<lastname>_KMeans.jar.

   The python script or the .jar file of your implementation should be inside the *Solution* directory of your submission along with your source codes.

2. **Output Files**
   We need two output files for Problem 1.
   Run your program against *Data/ yelp_reviews_clustering_small.txt*

   The format of the output should be exactly the same as the above snapshot for both cases.
   The names of the output files should be as:
   <firstname>_<lastname>_KMeans_small_W_5_20. json
   <firstname>_<lastname>_KMeans_small_T_5_20.json

Problem 2 (45 Points)
Implementing Clustering Algorithms using Spark with the Yelp Reviews Small Dataset

For this problem you will generate clusters of size **8** with random seed of **42** on Yelp Reviews dataset (**(***Data/yelp_reviews_clustering_small.txt)*. You will run K-Means, bisecting K-Means, and GMM algorithms in Spark. The maximum number of iterations should be set to **20**.

Deliverables for Problem 2

1. **Script or Jar File and Source Code**
   Please name your Python script as: <firstname>_<lastname>_Cluster.py.
   Or if you submit a jar file as: <firstname>_<lastname>_Cluster.jar.

   The python script or the .jar file of your implementation should be inside the *Solution* directory of your submission along with the source codes.

   Input Arguments:
   1. **Input.txt:** This is the path to the input reviews file. Every line contains a review. For Problem 2 you should use the *Data/yelp_reviews_clustering_small.txt* file

   2. **Algorithm=K**: K for K-Means, B for bisecting K-Means, G for GMM

   3. **N=8:** Number of clusters

   4. **Iterations =20**: Maximum number of iterations

2. **Output Files**
   There should be three output files corresponding to K-Means, bisecting K-Means, and GMMs. The K-Means and bisecting K-Means will contain the cluster sizes, within set sum of squared error, and top ten terms from the cluster. For GMM, the output will contain the weights, means, and standard deviation for each component. Example output is shown below.

Execution Example

Following we present examples of how you can run your program with spark submit both when your application is a Java/Scala program or a Python script.

B. **Example of running a Java/Scala application with spark-submit:**
   Notice that the argument class of the spark-submit specifies the main class of your application and it is followed by the jar file of the application.

```
$SPARK_HOME/bin/spark-submit  --class  Task2  Firstname_Lastname_Cluster.jar
<input_file> <algorithm> <N> <Iteration>
```

Example of running a Python application with spark-submit:

```
$SPARK_HOME/bin/spark-submit    Firstname_Lastname_Cluster.py    <input_file>
<algorithm> <N> <Iteration>
```

Example K-Means Output

```
{
      "algorithm":"K-Means",
      "WSSE":0.52,
      "clusters":[ {
            "id":1,
            "size":50,
            "error":0.25,
            "terms": ["restaurant","food","good"…]
      },
….

}
```

Example Bisecting K-Means Output

```
{
      "algorithm":"Bisecting K-Means",
      "WSSE":0.52,
      "clusters":[ {
            "id":1,
            "size":50,
            "error":0.25,
            "terms": ["restaurant","food","good"…]
      },
….

}
```

Example GMM Output

```
{
     "algorithm":"GMM",
     "clusters":[ {
          "id":1,
          "weight":0.2,
          "mean":0.4,
          "std": 0.01
     },
….

}
```

The output files should be named as:
<firstname>_<lastname>_Cluster_small_K_8_20.json
<firstname>_<lastname>_Cluster_small_B_8_20. json
<firstname>_<lastname>_Cluster_small_G_8_20. json

The above output files should be placed inside the *OutputFiles* directory of your submission.

Grade breakdown
a. Three correct output files (15pts each)

Problem 3 (30 Points)
Implementing Clustering using Spark with the Yelp Reviews Large Dataset

For this problem you will generate clusters of size **16** with random seed of **42** on Yelp Reviews dataset (**(***Data/yelp_reviews_clustering_large.txt)*. You will run K-Means, bisecting K-Means, and GMM algorithms in Spark.

<u>**Deliverables for Problem 3**</u>

1. Output Files

The format of the output should be exactly the same as the one for Problem 1.
The output files should be named as:
<firstname>_<lastname>_Cluster_large_K_16_20.json
<firstname>_<lastname>_Cluster_large_B_16_20. json
<firstname>_<lastname>_Cluster_large_G_16_20. json

The above output files should be placed inside the *OutputFiles* directory of your submission.

Grade breakdown
a. Three correct output files (10 pts each)

Bonus (5pts): Plot the clusters using your favorite visualization tool. You are to choose how you want to visualize the clusters. The visualization should be limited to 1 page including brief text describing the clusters. Include this as Firstname_LastName_Visualization.pdf

General Instructions:

1. Make sure your code compiles before submitting
2. Make sure to follow the output format and the naming format.

Grading Criteria:

1. If your programs cannot be run with the commands you provide, your submission will be graded based on the result files you submit and 20% penalty for it.
2. If the files generated by your programs are not sorted based on the specifications, there will be 20% penalty.
3. If your program generates more than one file, there will be 20% penalty.
4. **If you don't provide the source code and just the .jar file in case of a Scala application there will be 60% penalty.**
5. **If your submission does not state inside the Description pdf file how to run your code, which Spark version you used and which approach you followed to implement your algorithm there will be a penalty of 30%.**
6. **There will be a penalty of 20% if your class names do not match with the class names given for each task.**
7. **There will be a penalty of 20% if you do not follow the given folder structure.**
8. There will be 20% penalty for late submission.

APPENDIX A

Please include the following information inside your description document.

- Succinctly describe your approach to implement the algorithm.
- Commands to execute your program