CSE 6010 Assignment 2: Data Analysis

Due Dates:

- Due: 11:00 AM, Friday, September 22, 2017
- Revision (optional): 11:55 PM, Monday September 25, 2017

To complete this assignment you will work with one other student to develop two data analysis programs. The first program implements the k-means algorithm that clusters data samples from a data set into k distinct clusters, or categories. The second program uses the results produced by the K-means program to classify new data samples by assigning each sample to one of the k clusters using the K nearest neighbor algorithm. Each student in the team must implement one of these programs. You will need to work with the other student in your team to jointly define the format of data passed between the two programs. You will also need to work with your partner to analyze data provided to you, and write up your findings in a jointly-authored report.

The K-Means Problem and Algorithm

K-means is a well-known approach used in machine learning to classify data. The input to the algorithm is a set of data samples, where each sample is a point in an *n* dimensional space. In other words, each sample is a tuple of *n* data values $(x_1, x_2, \ldots x_n)$, where each dimension represents some attribute of interest (time, temperature, location, etc.). The goal is to partition the data samples into *k* distinct "clusters" where the samples within each cluster are "similar." For example, one of the earliest applications of clustering arose in the 19th century where the locations of deaths from a cholera outbreak were clustered, and used to identify the cause of the outbreak. In this case deaths were found to be clustered geographically around contaminated water wells.

The name "K-means" is commonly used to refer to both the clustering problem, and a specific algorithm used to solve it. The value of *k* is an input given to the algorithm. K-means is one algorithm in a class of machine learning techniques known as *unsupervised learning*.

Given a set of data points (samples) in an n-dimensional space, the *centroid* μ is the point defined by the arithmetic average of the data points along each dimension. Here, our interest is the centroids defined by the points within each cluster $(\mu_1, \mu_2, \ldots \mu_k)$. We would like the data points within the cluster to be "close" to each other, so a natural metric for a cluster is to consider the distance of each data point within the cluster from the cluster's centroid. In particular, the metric of interest is termed the *within cluster sum of squares distance* (or variance). The K-means problem is given a set of *n* data points $(x_1, x_2, \ldots x_n)$ and the value *k*, find $S = (S_1, S_2, \ldots S_k)$ that is a partition of the data points into *k* sets the minimizes the sum of squares distance:

$$\arg\min_{S} \sum_{i=1}^{k} \sum_{x \in S_i} \|x - \mu_i\|^2$$

The algorithm for solving this problem is commonly called the "k-means algorithm" or Lloyd's algorithm. The algorithm repeatedly performing the following steps:

1. Assign each data sample to the cluster with minimum distance between the data point and that cluster's centroid
2. After all samples have been assigned to clusters, update the cluster's centroid to the average of the points assigned to that cluster.

For this assignment, assume the first k data points in the data set are the initial centroids. Your program should repeat the above two steps until the root-mean-square falls below some pre-defined threshold (defined by you) or until the algorithm completes some maximum number of iterations if it is not able to reach this threshold.

To complete this part of the assignment, write a C program that takes three command line parameters:

1. The name of the input file containing the data to be clustered
2. The value of `k`
3. The name of the file where the output (to be used by KNN) is placed.

In addition, to creating the output file, for debugging purposes your program should print to the screen evidence that it is producing correct results. For example, you might print after each iteration the coordinates of the centers of each of the *k* clusters, the number of data items in each cluster, and the root-mean-square of the distances of each data point to its center.

K-Nearest Neighbors Algorithm

While K-means clusters data samples into distinct categories, K-nearest neighbors (KNN) is a simple approach to classify new samples, and label them based on a labeled data set, e.g., one produced by K-means. Specifically, KNN takes as input a labeled data set containing non-target data points, and target data points that we wish to label. KNN first finds the k non-target data points that are the closest to the target data item, and assigns a label to the target corresponding to the majority among these k nearest non-target items. It is common to choose k to be odd to reduce or eliminate the possibility of ties. Because KNN relies on a data set with known labels (also called the training data set), it is referred to as a *supervised learning* algorithm.

To complete this part of the assignment, write a C program that implements KNN. It should take the following command line parameters:

1. The name of the input file containing the non-target (training) set data produced by the K-means program.
2. The name of a second input file containing the data to be classified.
3. The name of the output file where results produced by KNN are placed.
4. The value of k used by KNN (not to be confused with the parameter used by K-means)

The output should indicate the label/cluster assigned to each target input sample.

Data Set Format

Assume the input data file for the K-means program and the target data for the KNN program has the following format:

```
num_items num_attrs
item1_attr1 item1_attr2 ... item1_attrm
item2_attr1 item2_attr2 ... item2_attrm
...
itemn_attr1 itemn_attr2 ... itemn_attrm
```

The first line contains the number of data items `num_items` and the number of attributes `num_attrs`. The following `num_items` lines contain the data items, with each line containing `num_attrs` attribute values.

You and your partner will need to define a file format to hold the results produced by K-means program that are passed to the KNN program.

Program Testing

Construct some synthetic data sets to verify that both of your programs are working correctly. In your report, describe your testing procedure and provide evidence of correct operation (e.g., test using different values of k, including invalid values, etc.) for each program. Explain why you think your programs are correct from these tests.

Data Analysis Problem

A sample data set from a biomedical application is provided on T-square in the file `gbm-KM.norm`. This data set comes from a microscope image of cancer tissue (see the file `gbm-image.jpeg`). The cell nuclei in the image were identified using image segmentation tools. Then 39 attributes of the nuclei, including size and elongation, were extracted and normalized. The nuclei of cancer cells are generally enlarged. We thank biomedical researchers Tony Pan and Joel Saltz who provided this data.

It is hypothesized that there is more than one type of cancer cell in the image, as well as possibly normal, undiseased cells. Use your K-means program to analyze and cluster this data set. Experiment with different values of `k`. Report on your studies on this data with your program.

Based on your results, suggest a suitable value of `k` and generate an output data file assigning each sample to one of the `k` clusters. Then apply your KNN program to classify the data points in the file `gbm-KNN.norm` using the file generated by your K-means program.

Final Comments

Turn your report and software in as a single zip file. Your software must be well documented and include comments so the code is easy to understand. You should include a README file with instructions on how to compile and run your program.

The web is an excellent resource to answer specific questions on C. We encourage you to use it, but be careful not to utilize code copied directly from the web. A reminder you must adhere to the Georgia Tech honor code, and the collaboration policy stated in the course syllabus. Specifically, you are encouraged to discuss the problem and possible solutions with other students (as well as the TA/instructor), however, all code that you turn in must be completely your own work. Disseminating your code to other students is strictly prohibited. Further, downloading code from the web or other sources other than examples provided in class is also prohibited.