CS 4375 - ASSIGNMENT 3

Please read the instructions below before starting the assignment.

- This assignment has two parts. The first part requires you to answer theoretical questions, while the second part involves coding the K-means unsupervised learning algorithm.

- Please create two different folders, named part I, and part II, and keep your files in the respective folders.

- For each of the code folders, please include a README file indicating how to compile and run your code. Also, mention clearly which packages/libraries you have used.

- You should use a cover sheet, which can be downloaded at: http://www.utdallas.edu/~axn112530/cs4375/CS4375_CoverPage.docx

- You are allowed to work in pairs i.e. a group of two students is allowed. Please write the names of the group members on the cover page.

- You have a total of 4 free late days for the entire semester. You can use at most 2 days for any one assignment. After that, there will be a penalty of 10% for each late day. **The submission for this assignment will be closed 2 days after the due date.**

- Please ask all questions through Piazza, and not through email.

PART I

(10 points)
1. Consider a regression problem of trying to estimate the function f: X -> y where X is a vector of feature attributes and y is a continuous real valued output variable. You would like to use a bagging model where you first create M bootstrap samples and then use them to create M different models – h_1, h_2, …, $h_{M.}$ You can assume that all models are of the same type.

The error for each of the models would be described as:

$$\epsilon_i(x) = f(x) - h_i(x)$$

where x is the input data and h_i is the model created using i^{th} bootstrap sample

The expected value of the squared error for any of the models will be defined as:

$$E(\varepsilon_i(x)^2) = E[\big(f(x) - h_i(x)\big)^2]$$

The average value of the expected squared error for each of the models acting individually is defined as:

$$E_{avg} = \frac{1}{M}\sum_{i=1}^{M} E(\varepsilon_i(x)^2)$$

Now, you decide to aggregate the models using a committee approach as follows:

$$h_{agg}(x) = \frac{1}{M}\sum_{i=1}^{M} h_i\,(x)$$

The error using the aggregated model is defined as:

$$E_{agg}(x) = E[\left\{\frac{1}{M}\sum_{i=1}^{M} h_i\,(x) - f(x)\right\}^2]$$

which can be simplified as:

$$E_{agg}(x) = E[\left\{\frac{1}{M}\sum_{i=1}^{M} \epsilon_i\,(x)\right\}^2]$$

where we used the value of ε_i is defined above.

Prove that

$$E_{agg} = \frac{1}{M} E_{avg}$$

provided you make the following assumptions:

1. Each of the errors have a 0 mean

$$E(\epsilon_i(x)) = 0 \text{ for all i}$$

2. Errors are uncorrelated

$$E\left(\epsilon_i(x)\epsilon_j(x)\right) = 0 \;\; \text{for all i} \neq \text{j}$$

(10 points)
2. Jensen's inequality states that for any *convex* function *f*:

$$f(\sum_{i=1}^{M} \lambda_i x_i) \leq \sum_{i=1}^{M} \lambda_i f(x_i)$$

In question 1, we had assumed that each of the errors are uncorrelated i.e.

$$E\big(\epsilon_i(x)\epsilon_j(x)\big) = 0 \text{ for all i} \neq \text{j}$$

This is not really true, as the models are created using bootstrap samples and have correlation with each other. Now, let's remove that assumption. Show that using Jensen's inequality, it is still possible to prove that:

$$E_{agg} \leq E_{avg}$$

(10 points)
3. Deriving the training error for AdaBoost:
In class, we discussed the steps of Adaboost algorithm. Recall that the final hypothesis for a Boolean classification problem at the end of T iterations is given by:

$$H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$$

The above equation says that the final hypothesis is the weighted hypothesis generated at the end of each individual step.
Also recall that the weight for the point i at step t+1 is given by:

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times e^{-\alpha_t h_t(i) y(i)}$$

where:
$D_t(i)$ is the normalized weight of point i in step t
$h_t(i)$ is the hypothesis (prediction) at step t for point i
α_t is the final "voting power" of hypothesis h_t
$y(i)$ is the true label for point i
Z_t is the normalization factor at step t (it ensures that the weights sum up to 1.0)

Note that at step 1, the points have equal weight

$$D_1 = \frac{1}{N}$$

where N is the total number of data points.
At each of the steps, the total error of h_t will be defined as $\varepsilon_t = \frac{1}{2} - \gamma_t$, which is a way of saying that the error will be better than 50% by a value γ_t.

Prove that at the end of T steps, the overall training error will be bounded by:

$$\exp(-2 \sum_{i=1}^{T} \gamma_t^2)$$

That is, the overall training error of the hypothesis H will be less than or equal to the amount indicated above.

Part II (70 points)

Tweets Clustering using k-means

Twitter provides a service for posting short messages. In practice, many of the tweets are very similar to each other and can be clustered together. By clustering similar tweets together, we can generate a more concise and organized representation of the raw tweets, which will be very useful for many Twitter-based applications (e.g., truth discovery, trend analysis, search ranking, etc.)

In this assignment, you will learn how to cluster tweets by utilizing Jaccard Distance metric and K-means clustering algorithm.

Objectives:

- Compute the similarity between tweets using the Jaccard Distance metric.
- Cluster tweets using the K-means clustering algorithm.

Introduction to Jaccard Distance:

The Jaccard distance, which measures dissimilarity between two sample sets (A and B). It is defined as the difference of the sizes of the union and the intersection of two sets divided by the size of the union of the sets.

$$Dist(A,B) = 1 - \frac{|A \bigcap B|}{|A \bigcup B|} = \frac{|A \bigcup B| - |A \bigcap B|}{|A \bigcup B|}$$

For example, consider the following tweets:

Tweet A: the long march

Tweet B: ides of march

|A ∩ B | = 1 and |A U B | = 5, therefore the distance is 1 – (1/5)

In this assignment, a tweet can be considered as an unordered set of words such as {a,b,c}. By "unordered", we mean that {a,b,c}={b,a,c}={a,c,b}=...

Jaccard Distance Dist(A, B) between tweet A and B has the following properties:

- It is small if tweet A and B are similar.
- It is large if they are not similar.
- It is 0 if they are the same.
- It is 1 if they are completely different (i.e., no overlapping words).

Here is the reference for more details about Jaccard Distance:
http://en.wikipedia.org/wiki/Jaccard_index

Hint: Note that since the tweets do not have numerical coordinates as in Euclidean space, you might want to think of a sensible way to compute the "centroid" of a tweet cluster. *This could be the tweet having minimum distance to all of the other tweets in a cluster.*

Exercise:

Implement the tweet clustering function using the Jaccard Distance metric and K-means clustering algorithm to cluster redundant/repeated tweets into the same cluster. **Remember that you have to write your own code for K-means clustering**. It is acceptable to use external libraries for *data loading and pre-processing only*. Python is the preferred language for this assignment. If you want to use any other language, clearly specify how to compile and run your code in the README file.

Note that while the K-means algorithm is proved to converge, the algorithm is sensitive to the k initial selected cluster centroids (i.e., seeds) and the clustering result is not necessarily optimal on a random selection of seeds.

Steps of the exercise:

(1) We are going to use the following dataset for this exercise:
https://archive.ics.uci.edu/ml/datasets/Health+News+in+Twitter

Follow the “Data Folder” link and unzip the given file. You will file a folder containing tweets that contain links to various news sources e.g. the file “usnewshealth.txt” contains tweets that refer to articles published in US News. **You have to choose one such file and proceed.**

(2) Perform the following pre-processing steps:

- Remove the tweet id and timestamp
- Remove any word that starts with the symbol @ e.g. @AnnaMedaris
- Remove any hashtag symbols e.g. convert #depression to depression
- Remove any URL
- Convert every word to lowercase

(3) Perform K-means clustering on the resulting tweets using at least 5 different values of K and report your results in the format below
Note that the sum of squared error is defined as:

$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(m_i, x)$$

where K is the number of clusters and m_i is the centroid of the i^{th} cluster.

Value of K	**SSE**	**Size of each cluster**
10	200	1: 10 tweets 2: 25 tweets 3: 20 tweets …. 10: 100 tweets
….		

What to Turn In for Part II :
(1) Table of results as mentioned earlier.
(2) The source code including a README file indicating how to run your code.