CIS 5930: Data Mining

Fall 2019

Assignment 3

Instructions

- **Submission Deadline**: **Thursday, October 31 2019, 11:59pm.** See the submission instructions below
- There are two problems. Total points possible: 25
- You have to use Matlab or Python for Problem 2
- Submit the solutions through Canvas

Problem 1 (10 points)

Consider the posterior probabilities obtained by applying a trained classification model on a test set with 10 instances. Plot the ROC curve.

Instance	True Class	P(+|A)
1	+	0.61
2	+	0.03
3	-	0.68
4	-	0.31
5	+	0.45
6	+	0.09
7	-	0.38
8	-	0.05
9	+	0.01
10	-	0.04

Problem 2 (15 points)

A practical challenge in training a reliable machine learning model is the requirement of a large amount of labeled training data. While gathering unlabeled data is cheap and easy, labeling the data is an expensive process in terms of time, labor and human expertise. Active learning algorithms automatically identify the unlabeled samples that need to be labeled to train a reliable machine learning model. When exposed to large amounts of unlabeled data, they select the salient and exemplar samples for manual annotation. In this project, you will study the application of active learning in the classification setting, on two datasets.

Classification Model: We will use the Logistic Regression (LR) model in this project (Note: LR is a classifier, not a regression model). We have not covered LR in class; you are given the

necessary functions to train and test an LR model in Matlab. Copy all the files in your current Matlab directory. The following two functions should be used for training and testing:

train_LR_Classifier: Function to train a LR classifier. It takes the training samples, labels and the number of classes as inputs and returns the parameters of the trained model.

test_LR_Classifier: Function to test the LR classifier. It takes a test sample, the trained weights and the number of classes as inputs and returns a vector containing the probabilities of the sample corresponding to all the classes.

If you are using Python, you should be able to find built-in functions to train and test an LR classifier.

Active Learning Outline: The functioning of an active learning system can be outlined as follows:

Given: Small amount of initial training set (training samples and labels); large amount of unlabeled data with NO labels (the labels of the unlabeled samples should be used only in response to a user query); a test set to judge the performance of the system; a batch size k (number of unlabeled samples to be queried in each iteration) and the number of iterations N

Loop over N iterations

Step 1: Train a machine learning model using the current training set

Step 2: Apply the model on the test set and obtain the accuracy

Step 3: Apply the model on the unlabeled set and select a batch of k unlabeled samples based on an active learning strategy (see below)

Step 4: Obtain the labels of the selected k samples from a human expert (you will use the provided labels of the unlabeled samples here to simulate the human expert)

Step 5: Remove those samples from the unlabeled set and add them to the current training set

End Loop

In each iteration, you will get an accuracy value. After the active learning iterations are over, you will get a vector containing N accuracy values. Our objective is to study the rate of increase of accuracy over iterations. This is expressed as a plot of the number of iterations (on the x-axis) and the accuracy (on the y-axis). An active learning algorithm is better than another if its accuracy grows at a faster rate.

In this project, use k = 10 and N = 50. Also, for each dataset, repeat the process 3 times using 3 different initial training sets, unlabeled sets and test sets and report the average accuracy results.

Active Learning Strategies: We will study the performance of two active learning strategies in this project:

(1) Random Sampling: Select a batch of k samples from the unlabeled set at random

(2) Uncertainty-based Sampling: For each unlabeled sample, compute the classification entropy as $e = - \sum p_i \log (p_i)$, where i runs from 1 to the number of classes and p_i is the probability that the sample belongs to class i. Select the k samples producing the highest entropy.

For each dataset, plot one graph containing the performance of Random Sampling and Uncertainty-based Sampling (on the same graph). Write your observations in your report.

Datasets: We will use two facial expression recognition datasets in this problem: MindReading and MMI. They both contain samples belonging to 6 classes. Each dataset contains an initial labeled training set, an unlabeled set and a test set. Also, each experiment needs to be run 3 times and the average results should be reported. For the training, testing and unlabeled matrices, each row denotes a sample and each column denotes a feature.

Please submit the following:

- The Matlab / Python code files through Canvas (a submission link has been created)
- A ReadMe file with clear instructions on how to run your code (through Canvas)
- A brief report (about 1 page) summarizing your findings. Include two accuracy graphs, one for the MMI and another for the MindReading datasets. Each graph should contain a plot of the accuracy against the number of iterations for the two active learning strategies (Random and Uncertainty Sampling). Mention your conclusions.

Zip all the files (the solution to Problem 1, the report, source codes and ReadMe files for Problem 2) in a single folder and submit through Canvas.