Assignment 2

Due on November 10, 2016 (23:59:59)

Instructions. There are two parts in this assignment. The first part involves a series of theory questions and the second part involves coding. The goal of this problem set is to make you understand and familiarize with Naive Bayes algorithm.

PART I: Theory Questions

MLE

- Suppose you have N samples $x_1, x_2.....x_N$ from a univariate normal distribution with unknown mean μ and known variance σ^2. Derive the MLE estimator for the mean μ.
- Consider a dataset $(x^n, c^n), n = 1, ..., N$ of binary attributes, $x_i^n \in 0, 1, i = 1, ..., D$ and associated class label c^n. The number of datapoints from class $c = 0$ is denoted n_0 and the number from class $c = 1$ is denoted n_1. Estimate $p(x_i = 1|c) \equiv \theta_i^c$.
- A training set consists of one dimensional examples from two classes. The training examples from class 1 are {0.5, 0.1, 0.2, 0.4, 0.3, 0.2, 0.2, 0.1, 0.35, 0.25} and from class 2 are {0.9, 0.8, 0.75, 1.0}. Fit a (one dimensional) Gaussian using Maximum Likelihood to each of these two classes. You can assume that the variance for class 1 is 0.0149, and the variance for class 2 is 0.0092. Also estimate the class probabilities p1 and p2 using Maximum Likelihood. What is the probability that the test point x = 0.6 belongs to class 1?

Naive Bayes

- Whizzco decide to make a text classifier. To begin with they attempt to classify documents as either sport or politics. They decide to represent each document as a (row) vector of attributes describing the presence or absence of words.

  x = (goal,football,golf,defence,offence,wicket,office,strategy)

  Training data from sport documents and from politics documents is represented below in a

  matrix in which each row represents the 8 attributes.

$$x_{politics} = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

$$x_{sport} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \end{bmatrix}$$

Using a maximum likelihood naive Bayes classifier, what is the probability that the document x = (1,0,0,1,1,1,1,0) is about politics?

PART II: Sentiment Analysis

In this part of the assignment, you will try to predict the sentiment of movie reviews that are given. You will implement a Naive Bayes classifier and verify it's performance on a movie review dataset. As you learned in class, Naive Bayes is a simple classification algorithm that makes an assumption about the conditional independence of features, but it works quite well in practice.

The Dataset

- You have been provided with a movie review dataset [1] contains positive and negative reviews with a rating score (a negative review has a score smaller than or equal to 4 out of 10, while a positive review has a score bigger than or equal to 7 out of 10). You will try to implement Naive Bayes algorithm to predict the sentiment of the movie review.
- It contains 50,000 classified reviews as a text file in separate folders (25,000 for training and 25,000 for validation).
- Both of the training and validation sets include 12,500 positive reviews and 12,500 negative reviews). Each text file's name includes it's id and rating (`id_rating.txt`).
- You can download the dataset from `ftp://ftp.cs.hacettepe.edu.tr/pub/dersler/BBM4XX/BBM409_ML/Assignment_2/MRDataset.zip`. The original dataset also includes additional files, you may also investigate these files for your assignment.

Approach

You will represent your data with listed approaches and use them to learn a classifier via Naive Bayes algorithm. You have to implement your own Naive Bayes algorithm.

- Features
  You will use Bag of Words(BoW) model which learns a vocabulary from all of the documents, then models each document by counting the number of times each word appears. You will use BoW with two options:
  - Unigram: The occurrences of words in a document(frequency of the word).

  - Bigram: The occurrences of two adjacent words in a document.

- **Implementation Notes**
  - You should compute the log probabilities to prevent numerical underflow when calculating multiplicative probabilities.
  - You may encounter words during classification that you haven't during training. This may be for a particular class or over all. Your code should deal with that. *Hint: You can use Laplace smoothing*
  - You have to use a dictionary for BoW representation. You can implement your own method to obtain Bow model or you can use scikit-learn library `http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html`.
  - You can narrow down your dictionary by choosing specific words for positive and negative reviews. In other words, your classification results can be improved by selecting a subset of extremely effective words for the dictionary. TF-IDF (`http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfTransformer.html`) and Information Theory are good places to start looking.
- You will compute accuracy of your model to measure the success of your classification method:

$$\textbf{Accuracy} = 100 * \left(\frac{\textbf{number of correctly classified examples}}{\textbf{number of examples}}\right) \tag{1}$$

Submit

- report.pdf (PDF file containing your report)
- code/ (directory containing all your codes as Python file .py)

The ZIP file will be submitted via the department's submission system.

NOTE: To enter the competition, you have to register kaggle in Class with your department email account. The webpage of the competition will be announced later.

Grading

- Code (50): Naive Bayes: 30 points, BoW: 20
- Report(50): Theory part: 12 points, Analysis of the results for prediction: 38 points.

Notes for the report: You should explain your choices (Unigram, Bigram or both of their use for Bow) and their effects to the results. You can create a table to report your results like:

Feature	Accuracy
Unigram	0.3
Bigram	0.5
Unigram and Bigram	0.7

Late Policy

You may use up to five extension days (in total) over the course of the semester for the three problem sets you will take. Any additional unapproved late submission will be weighted by 0.5. You have to submit your solution in (rest of your late submission days + 5 days), otherwise it will not be evaluated.

Academic Integrity

All work on assignments must be done individually unless stated otherwise. You are encouraged to discuss with your classmates about the given assignments, but these discussions should be carried out in an abstract way. That is, discussions related to a particular solution to a specific problem (either in actual code or in the pseudocode) will not be tolerated. In short, turning in someone else's work, in whole or in part, as your own will be considered as a violation of academic integrity. Please note that the former condition also holds for the material found on the web as everything on the web has been written by someone else.

References

[1] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts, Learning Word Vectors for Sentiment Analysis, *The 49th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2011,