CIS 521: HW 7 - Machine Learning

1 Instructions

This homework consists of a pair programming assignment along with a discussion of the results. As usual, please make sure you do the following:

1. BOTH partners should submit a "soft copy" of your write-up (with both names on it) and code on blackboard prior to the beginning of class on **Thursday, April 5th**.

START EARLY! This homework is more complicated than previous ones and requires running relatively long tests. We recommend sampling a small portion of the files used to test more quickly. Once you are sure your algorithm works, then run it on the larger files. Also be sure to see the "supplementary" description with key hints and details

2 Programming Portion

In this section you will apply several different methods to the problem of *document classification*. The goal of this problem is to automatically label documents as one of several "types;" for this homework, the inputs are chunks of text posted by users, and the corresponding "type" label is which *newsgroup*[1] the user posted in. The data is in `hw4.zip`. You will be considering postings from 4 different newsgroups:

1. `rec.sport.baseball.txt` - Baseball discussion
2. `rec.sport.hockey.txt` - Hockey discussion
3. `comp.sys.ibm.pc.hardware.txt` - PC hardware discussion
4. `comp.sys.mac.hardware.txt` - Mac hardware discussion

Each line of one of the data files is the entire text of a single posting. Also though there are 4 newsgroups that we will consider, at any given time your algorithm will only have to decide between 2 labels for a given posting.

[1]Note: A newsgroup is an older form of internet discussion board, like a forum or e-mail list.

Getting started. Because machine learning is numerically intensive, we'll be using NumPy, a numerical package for Python, on this assignment. See Blackboard for information on downloading, installing, and getting started with NumPy.

We are providing you with a helper class `Dataset` (in file "Dataset.py") that reads in the dataset and converts it to NumPy data structures. `Dataset` will read in the text files and create a dictionary `wordmap` that maps words to sequential id's. Then, `Dataset` converts each posting to a single row vector $\mathbf{x}$, where each element x_i is 1 if the i'th word (defined by the dictionary) occurs in the posting and 0 otherwise. The corresponding label y is either +1 or -1, depending on which file the post came from.

This is a standard representation that's a good starting point for document classification. Note that `Dataset` by default only uses the 2000 most frequent words in the dataset, but you should feel free to experiment with this parameter. Finally, `Dataset` provides the `getTrainAndTestSets` method that randomly splits the complete dataset into a training and test set, so you can evaluate the different approaches.

Requirements. The requirements for this assignment are as follows:

- Implement each of the following machine learning algorithms for binary classification:
  - Naive Bayes
  - Ridge regression
  - Streamwise regression
  - Stepwise regression
  - Perceptron

  Note that if you use the n most frequent words in the dataset, then solving ridge regression requires inverting a $n \times n$ matrix and thus $O(n^3)$ time and space. This can take a while on a laptop if n is larger than 1000, so you don't have to use the default value of 2000. For the other methods, you will definitely want to try considering more words as features.

  Note also that for ridge regression, you will need to choose a λ regularization parameter, and for perceptron, a learning rate.

  Finally, note that for stepwise and streamwise regression, you will need to choose when to stop adding features to the procedure. See the lecture notes on Blackboard for more information.

- Evaluate each method on 3 binary classification tasks: PC vs. Mac, Baseball vs. Hockey, and PC vs. Baseball (pass the corresponding `.txt` files to the `Dataset` class.) For each comparison, run the `getTrainAndTestSets` method to generate 80-20 splits with five different random seeds (the optional `seed` parameter), and report the average training and test error across random splits.[2] Make sure that you use the same random seeds for all algorithms; in other words, your code should generate one train/test split and then run *all* of the algorithms on that data.

- Write a report discussing your findings. *Include the table of results (training and testing error for each method on each of the three pairs of text)*, and address the following questions:

[2]For those of you that are familiar with hold-one-out cross-validation, this is NOT that type of cross-validation. This simpler procedure generates five different training and test sets that may overlap.

1. What choices did you make when implementing the algorithms? List the settings of parameters like learning rate, number of words considered for each algorithm, or the λ parameter. Do you think these choices affected the results in any way? Why or why not?
2. Of the three comparisons, which was the most difficult comparison across all algorithms? Why do you think it was the hardest?
3. For streamwise and stepwise regression, list the top ten features added for each of the three comparisons. (Make sure to report the *words*, not just feature indices.) Do these features "make sense," i.e., would you have predicted that these words were useful for distinguishing which newsgroup a post belongs in? Why or why not?

The text of your report should not exceed 1 single-spaced page (not including the main table of results, the lists of most discriminatory words, or any supplementary tables or figures). Also, if you run out of time or are having trouble with any of the algorithms, you can discuss any issues you had for partial credit. Remember to include whatever Python code you do have, even if it's not complete.