Homework 3
CS 7301: Advanced Machine Learning

1 Perceptrons, and Neural Networks (50 points)

In this homework, you will implement the Perceptron algorithm and compare it with WEKA implementation of Neural networks. You will also compare it with either your own or WEKA implementations of Logistic Regression and Naive Bayes.

TASK 1: Download the datasets available on the class webpage or elearning. As in homework 2, the classification task is spam/ham.

30 points Implement the perceptron algorithm (use the perceptron training rule and not the gradient descent rule). Your task here is to experiment with different values of number of iterations and the learning rate. Report the accuracy for 20 suitable combinations of number of iterations and the learning rate. Repeat your experiment by filtering out the stop words. Compare the accuracy of your perceptron implementation with that of Naive Bayes and Logistic Regression (implemented in Homework 2). If you did not implement Naive Bayes (NB) and/or Logistic regression (LR) in homework 2 or you are not sure that your implementation is correct, please use the LR and NB implementations available in WEKA.

20 points Neural networks in WEKA.

- Download WEKA `http://www.cs.waikato.ac.nz/ml/weka/`. You can also use **scikit-learn** for this part if you are more comfortable with python.
- Convert the spam/ham dataset into the ARFF format used by WEKA (or write code to input data in scikit-learn).

– Using the Neural networks implementation in *WEKA* (called MultiLayered Perceptron) or *scikit-learn*, report the accuracy on the test set. ***Use just one hidden layer.*** Experiment with different number of hidden units in the hidden layer. Report on how the number of hidden units as well as other options such as momentum, number of iterations, and learning rate affect the accuracy.

2 Collaborative Filtering on Netflix Ratings (50 points)

- Read the paper Empirical Analysis of Predictive Algorithms for Collaborative Filtering linked on the course web page. You need to read up to Section 2.1, and are encouraged to read further if you have time.
- The dataset we will be using is a subset of the movie ratings data from the Netflix Prize. You need to download it from the course web page. It contains a training set, a test set, a movies file, a dataset description file, and a README file. The training and test sets are both subsets of the Netflix training data. You will use the ratings provided in the training set to predict those in the test set. You will compare your predictions with the actual ratings provided in the test set. The evaluation metrics you need to measure are the Mean Absolute Error and the Root Mean Squared Error. The dataset description file further describes the dataset, and will help you get started. The README file is from the original set of Netflix files, and has been included to comply with the terms of use for this data.
- Implement the collaborative filtering algorithm described in Section 2.1 of the paper (Equations 1 and 2; ignore Section 2.1.2) for making the predictions. You may program in C, C++, Java or Python.

WRITTEN QUESTIONS: (Will not be graded. However, it will help you understand the concepts)

- Design a two-layer neural network that implements $A\ XOR\ B$. Namely, we have two Boolean features A and B, each of which take values from the domain $\{0, 1\}$. Use a simple threshold unit.
- Problem 4.8 from Tom Mitchell's book. Revise the BACKPROPAGATION Algorithm so that it operates on units using the squashing function *tanh* in

place of the sigmoid function. That is, assume the output of a single unit is $o = tanh(\overrightarrow{w}.\overrightarrow{x})$. Give the weight update rule for output layer weights and hidden layer weights. **Hint:** Derivative of $tanh(x)$ is $1 - tanh^2(x)$.

- Problem 4.10 from Tom Mitchell's book. Consider the alternative error function defined below:

$$E(\overrightarrow{w}) = \frac{1}{2} \sum_{d \in D} \sum_{k \in outputs} (t_{kd} - o_{kd})^2 + \gamma \sum_{i,j} w_{i,j}^2$$

  Derive the gradient descent update rule for this definition of E. Show that it can be implemented by multiplying each weight by some constant before performing the standard gradient descent update in the BACKPROPAGATION algorithm.

What to Turn in:

- Part 1
  - Your Perceptron source code.
  - Script/code for converting a zip file containing spam/ham emails to ARFF (Weka) format or scikit-learn.
  - Your report describing the results of your experimental analysis. Make sure that you report your results on all the three data sets.
- Part 2
  - Code for collaborative filtering and a report describing the experimental evaluation on the netflix dataset.
- Answers to the written questions (will not be graded).