Homework #5 (Machine Learning)

Name: Spencer Fronberg
UID: u0766439

November 30, 2017

1 Logistic Regression

1. We need to take the derivative of the following function:

$$g(w) = log(1 + exp(-y_i w^T x_i))$$
$$\frac{d}{dw}(log(1 + exp(-y_i w^T x_i)))$$
$$\frac{1}{1 + exp(-y_i w^T x_i)} * exp(-y_i w^T x_i) * (-y_i x_i)$$
$$\frac{-y_i x_i}{exp(y_i w^T x_i) * (1 + exp(-y_i w^T x_i))}$$
$$\textbf{Final Answer: } - \frac{y_i x_i}{exp(y_i w^T x_i) + 1}$$

2. For SGD we can set m = 1 because with SGD we assume that we only have one example. We will then have to take the derivative of the following with respect to w:

$$g(w) = log(1 + exp(-y_i w^T x_i)) + \frac{1}{\sigma^2} w^T w$$

In the previous problem we already took the derivative of the log portion so:

$$-\frac{y_i x_i}{exp(y_i w^T x_i) + 1} + \frac{2w}{\sigma^2}$$

3. For each epoch 1....T

Foreach training example:

$$w \leftarrow (1 - \frac{2\gamma_t}{\sigma^2})w + \gamma_t(\frac{y_i x_i}{exp(y_i w^T x_i) + 1})$$

$$b \leftarrow (1 - \frac{2\gamma_t}{\sigma^2})b + \gamma_t(\frac{y_i}{exp(y_i b) + 1})$$

Shuffle Data

	Best hyper-parameters	Average cross-validation accuracy	Training accuracy	Test Accuracy
SVM	γ=0.0001,C=10	95.487	98.297	86.809
Logistic regression	γ=0.1, σ^2=1000	96.516	99.503	86.17
Naive Bayes	λ=0.5	69.5	74.344	68.83
Bagged Forests	No cross validation	No cross validation	65.459	60.251
SVM over trees	γ=0.00001,C=10	76.863	75.018	75.869
Logistic regression over trees	γ=0.001, σ^2=1000	78.212	78.531	79.595

Table 1: Result table

2 Experiments

For each algorithm, I completed them in C#

1. For SVM each of my vectors are represented as Dictionaries (key=int, value=double). I decided to not make an initial weight vector because I would have to go through 67692 examples every time I update, so in my algorithm when updating the w, if it doesn't exist, I then do the update and where the initial w would be, I generate a random number between -0.1 and 0.1. I do separate my bias from my weight vector just because it makes more sense for me to do that. It takes a total of 10 minutes to run my SVM.

2. For logistic Regression, I do the same exact thing as in SVM as in storing my vectors as Dictionaries and everything, but the only thing that is different is how I am updating my weight and bias which my pseudocode is shown in Question 1. It takes a total of 11 minutes to run my Logistic Regression.

3. For Naive Bayes, there is some overlap here into my Decision Tree. First off I create four dictionaries (key=int, value=double). Each of these store the total number of counts for the possibilities for each feature. Here are the four possibilities: positive label / positive feature, positive label / negative feature, negative label / positive feature, and negative label / negative feature. So I calculate the total count this way and run my Naive Bayes algorithm on these four dictionaries. In my algorithm, when I make a prediction for each example instead of multiplying each of the features' probabilities and the prior probability of that label, I take the log_{10} of each feature probability and the prior probability of the specific label and sum them up. Then I do the comparison to determine what the label is. It takes a total of 8 minutes to run my Naive Bayes 11:31

4. For this part I decided to store all my data's in Dictionaries. My training and test data stores a list of Entry where each entry stores the label and a Dictionary (key=int, value=double) for that label/example. My information gains that I calculated, I also stored in the same type of dictionary. My children nodes were stored in a list of

Children. For when I am splitting the data into two separate data's that will be assigned to their children tree, I determine if the Feature value contains the key, if it doesn't, I assign that example to the left tree if it does I assign it to the right tree. To store my Decision Tree I created a DecisionTree object that stored all the data needed for the decision tree (children, feature value, isLeaf, etc). It takes about an hour to an hour an a half to run my Bagged Forest. The files named "tree.test.data" and "tree.train.data" are the data files that my bagged forest generated to use for the next two parts. I used them while assisting myself in getting my accuracies and debugging so I did not have to re-run the entire bagged forest, but for my submission they are not used. I am providing them though.

5. As I said before, my submission does not actually use the data files provided, I just run my this section right after Bagged Forest and pass in the data that the trees create into SVM. Also I decided that if the feature value in any example is -1, I set it to be 0 rather than -1, but my +1 features remain to be 1 as you can see in the data files I submitted. When it comes to how I computed SVM, I did not change anything from my SMV algorithm for SVM over trees. Once my bagged forest has ran it takes about 5 to 10 minutes to run my SVM over trees

6. I do the same exact thing as I do in SVM when it comes to the data that I use. When it comes to how I computed Logistic Regression, I did not change anything from my Logistic Regression algorithm for Logistic Regression over trees. Once my bagged forest has ran it takes about 5 to 10 minutes to run my Logistic Regression over trees

7. Because there are a total of 67692 features and you have to calculate the information gain for 67692 features on a total of 2818 training examples 200 times because the depth is 200. On top of that, we initially had to find the best hyper-parameter a total of four different times then train and test on the best hyperparameter which adds up to 5 times. On top of that, for each cross-validation we had to create 5 full decision trees (for 5 fold cross-validation) and that is why it takes a very long time.