B555: Assignment 3

Pawan Patel

November 18, 2015

1. a) See code for implementation. Naive Bayes gives a better prediction than random and usually beats linear regression, though it depends on the training set chosen. Naive Bayes with a column of ones does worse and produces an error. Since we are using a gaussian distribution for $P(x_i|y)$, we expectedly get a standard deviation of 0 for the column of ones, which produces an error when one tries to apply the pdf of a guassian to a test sample for prediction.

   b) + c) See code for implementation.

   d) In general, we can see that all three of these algorithms seem to do about slightly better than linear regression most of the time. Though, with a statistical test we cannot say the difference is significant. Occasionally, I have observed the nueral network do much worse and most of the time do just about as well, if not slightly worse, than linear regression. This is not entirely surprising as the only difference in our implementation of the two layer neural network than that of linear regression is in the use of a transfer function between two linear maps. Logistic regression seems to do the best most of the time, followed by Naive Bayes without ones. All of the implementations produce around 70 to 80 percent accuracy.

2. We compute maximum likelihood estimation with this model for $P(y|x,w)$.

We have

$$\text{argmax}_w \prod_{i=1}^{n} [\frac{1}{2}(1+\frac{w^T x_i}{\sqrt{1+(w^T x_i)^2}})]^{y_i} + [\frac{1}{2}(1-\frac{w^T x_i}{\sqrt{1+(w^T x_i)^2}})]^{1-y_i}$$

$$\implies ll(w) = \sum_{i=1}^{n} y_i \log(\frac{1}{2}(1+\frac{w^T x_i}{\sqrt{1+(w^T x_i)^2}}) + (1-y_i)\log(\frac{1}{2}(1-\frac{w^T x_i}{\sqrt{1+(w^T x_i)^2}})$$

$$= \sum_{i=1}^{n} y_i \log(\frac{1}{2}) + y_i \log(1+\frac{w^T x_i}{\sqrt{1+(w^T x_i)^2}}) + \log(\frac{1}{2}) - y_i \log(\frac{1}{2}) - y_i \log(1-\frac{w^T x_i}{\sqrt{1+(w^T x_i)^2}})$$

$$+ \log(1-\frac{w^T x_i}{\sqrt{1+(w^T x_i)^2}})$$

$$= \sum_{i-1}^{n} y_i \log(1+\frac{w^T x_i}{\sqrt{1+(w^T x_i)^2}}) - y_i \log(1-\frac{w^T x_i}{\sqrt{1+(w^T x_i)^2}}) + \log(\frac{1}{2}) + \log(1-\frac{w^T x_i}{\sqrt{1+(w^T x_i)^2}})$$

Taking derivatives, we arrive at:

$$\frac{dll(w)}{dw_k} = \sum_{i=1}^{n} y_i \frac{1}{(1+\frac{w^T x_i}{\sqrt{1+(w^T x_i)^2}})}(\frac{\sqrt{1+(w^T x_i)^2}(x_{ik}) - (w^T x_i)^2(1+(w^T x_i)^2)^{-\frac{1}{2}}(x_{ik})}{1+(w^T x_i)^2}$$

$$+ \sum_{i=1}^{n} y_i \frac{1}{(1-\frac{w^T x_i}{\sqrt{1+(w^T x_i)^2}})}(\frac{\sqrt{1+(w^T x_i)^2}(x_{ik}) - (w^T x_i)^2(1+(w^T x_i)^2)^{-\frac{1}{2}}(x_{ik})}{1+(w^T x_i)^2}$$

$$- \sum_{i=1}^{n} \frac{1}{(1-\frac{w^T x_i}{\sqrt{1+(w^T x_i)^2}})}(\frac{\sqrt{1+(w^T x_i)^2}(x_{ik}) - (w^T x_i)^2(1+(w^T x_i)^2)^{-\frac{1}{2}}(x_{ik})}{1+(w^T x_i)^2}$$

$$= \sum_{i=1}^{n} (\frac{y_i}{A_i} + \frac{y_i - 1}{B_i}) C x_{ik} \implies \nabla ll(w) = X^T[y/A + (y-1)/B] \circ C$$

where we have: $A_i = \frac{1}{(1+\frac{w^T x_i}{\sqrt{1+(w^T x_i)^2}})}$, $B_i = \frac{1}{(1-\frac{w^T x_i}{\sqrt{1+(w^T x_i)^2}})}$, and

$$C_i = (\frac{\sqrt{1+(w^T x_i)^2} - (w^T x_i)^2(1+(w^T x_i)^2)^{-\frac{1}{2}}}{1+(w^T x_i)^2}$$

Where by squaring, multiplication, and division of vectors, we mean entry wise. We implement gradient descent with 10000 iterations to get our weights. We could find the Hessian and use newton method, but its not so slow with the usual gradient descent so I chose not to compute the Hessian.

We find that logistic regression with this new transfer function is about as good as the previous one. Occasionally, it does slightly better and sometimes slightly worse, but its not clear there is any statistically significant difference.

3. a) Let $R_1 = \sum_{i=1}^{k} |w_i|$ and $R_2 = \sum_{i=1}^{k} |w_i|^2$ denote L1 and L2 regularizers, respectively. As derivatives are linear, the gradient and Hessian of the error terms will increase by the gradient and hessions of the regularizers. Note that

$$\nabla R_1 = (sign(w_1), ..., sign(w_k)) = w/abs(w)$$

and

$$\nabla R_2 = 2|w|$$

Thus, the Hessian of R_1 is zero and $HR_2 = 2I$ where I denots the identity matrix. In the implementation, we drop coefficients as they can be absorbed into the multiplier α.

In the implementation, one notices that since the Hessian of R_1 is 0, we can get noninvertibilities for the Hessian of the entire error. As such, we modify the code to use the usual gradient descent in this case. For the L2 Regularizer, this is not an issue.

b) As a third regularizer, I used the L3 regularizer. One computes $\nabla R_3 = 3w^2$ (where we mean entire-wise squaring) and $HR_3 = 2diag(w)$, where here we mean a square matrix whose diagonal entries are the entries of w, and are zero elsewhere. We add these to the gradient and hessian of the error, just as above.

c) These regularizers don't seem to do too well. L1 regularization does the poorest at just about the same as random. L2 and L3 do slightly better at about 53 percent accuracy. Linear Regression beats them all at a consistent 57-58 percent accuracy.