CMSC-478/CMSC-678 Machine Learning - Spring 2013
Homework Assignment 2
Due at the start of class on March 7^{th}

1. (Probability decision boundary) Consider a case where we have learned a conditional probability distribution $P(y|\mathbf{x})$. Suppose there are only two classes, and let $p_0 = P(y = 0|\mathbf{x})$ and let $p_1 = P(y = 1|\mathbf{x})$. Consider the following loss matrix:

	$y = 0$ (true)	$y = 1$ (true)
$\hat{y} = 0$ (predicted)	0	10
$\hat{y} = 1$ (predicted)	5	0

   Show that the decision $\hat{y}$ that minimizes the expected loss is equivalent to setting a probability threshold θ and predicting $\hat{y} = 0$ if $p_1 < \theta$ and $\hat{y} = 1$ if $p_1 \geq \theta$. What is this threshold for this loss matrix?

   Recall that the expected loss is the sum over all ways in which you can be wrong of the probability of being wrong times the loss incurred. You should predict class 1 if the expected loss for doing so is less than or equal to the expected loss for predicting class 0.

2. **CMSC-678 students only** (Dual perceptron algorithm) Note that the answer to this question is not at all obvious. You'll need to struggle with it a bit and do some reading. However, the insight you'll gain will be extremely useful as we move on to other algorithms.

   Consider the following learning algorithm known as the dual perceptron algorithm:

   Let $\alpha_i = 0$ for $i = 0, \ldots, N$
   Repeat forever
      Accept training example $(\mathbf{x}_i, y_i)$
      if $\sum_l \alpha_l \mathbf{x}_l \mathbf{x}_i y_i y_l \leq 0$
         $\alpha_i = \alpha_i + 1$

   Note that α_i is a counter of the number of times training example $\mathbf{x}_i$ has been misclassified. The outer loop that repeats forever cycles through the N training instances.

   Explain intuitively what this algorithm does and why it is sensible. You can use the web to help you answer this question. If you do, cite the sources you used. Note that this algorithm is equivalent to the online perceptron algorithm with learning rate 1 and weight vector $w = \sum_l \alpha_l \mathbf{x}_l y_l$.

3. (Double counting the evidence) Consider a problem in which the class label $y \in \{T, F\}$ and each training example X has 2 binary attributes $X_1, X_2 \in \{T, F\}$.

   Let the class prior be $p(Y = T) = 0.5$ and $p(X_1 = T|Y = T) = 0.8$ and $p(X_2 = T|Y = T) = 0.5$. Likewise, $p(X_1 = F|Y = F) = 0.7$ and $p(X_2 = F|Y = F) = 0.9$. Attribute X_1 provides slightly stronger evidence about the class label than X_2.

   - Assume X_1 and X_2 are truly independent given Y. Write down the naive Bayes decision rule.
   - What is the expected error rate of naive Bayes if it uses only attribute X_1? What if it uses only X_2?

     The expected error rate is the probability that each class generates an observation where the decision rule is incorrect. If Y is the true class label, let $\hat{Y}(X_1, X_2)$ be the predicted class label. Then the expected error rate is $p(X_1, X_2, Y|Y \neq \hat{Y}(X_1, X_2))$.

- Show that if naive Bayes uses both attributes, X_1 and X_2, the error rate is 0.235, which is better than if using only a single attribute (X_1 or X_2).
- Now suppose that we create new attribute X_3 which is an exact copy of X_2. So for every training example, attributes X_2 and X_3 have the same value. What is the expected error of naive Bayes now?
- Explain what is happening with naive Bayes? Does logistic regression suffer from the same problem? Explain why.

4. (Logistic regression) Implement 2-class logistic regression in any language of your choice. Run your code on any dataset chosen from the UC Irvine Machine Learning Repository which can be found here:

   `http://archive.ics.uci.edu/ml/`

   To test your implementation, load the dataset using (either write throwaway code to do this, i.e., don't worry about making it generic, or "embed" the data into your program), randomize the order of the instances, use the first two-thirds for training and the final one-third for testing. Turn in your code, the name of the dataset you ran it on, the weight vector found for the training instances, and the classification accuracy on the test instances.