MIDTERM SOLUTIONS: FALL 2012
CS 6375
INSTRUCTOR: VIBHAV GOGATE

March 28, 2012

The exam is closed book. You are allowed a double sided one page cheat sheet. Answer the questions in the spaces provided on the question sheets. If you run out of room for an answer, continue on the back of the page. Attach your cheat sheet with your exam.

NAME _____

UTD-ID if known _____

- Problem 1: _____
- Problem 2: _____
- Problem 3: _____
- Problem 4: _____
- Problem 5: _____
- Problem 6: _____

- TOTAL: _____

PROBLEM 1: TRUE/FALSE QUESTIONS (10 points)

1. (2 points) Naive Bayes is a linear classifier. True or False. Explain.

   False. The decision rule for Naive Bayes cannot be written as: $\sum_i w_i x_i > k$.

2. (2 points) Classifier A has 90% accuracy on the training set and 75% accuracy on the test set. Classifier B has 78% accuracy on both the training and test sets. Therefore, we can conclude that classifier A is better than classifier B (because it has better mean accuracy). True or False. Explain.

   False: Classifier A is probably overfitting the training set and not generalizing as well as classifier B (although we do not have overwhelming evidence in favor of classifier B).

3. (2 points) Consider a data set that is linearly separable and two perceptrons, one trained using the gradient descent rule and the other trained using the perceptron rule. Both perceptrons will have the same accuracy on the training set and the test set. True or False. Explain.

   False: A perceptron does not learn a unique boundary.

4. (2 points) Given infinite training data over n Boolean attributes, we can always learn the target concept using a decision tree but not using Naive Bayes or Logistic Regression. True or False. Explain.

   True: A decision tree of depth n can represent any Boolean function perfectly. Thus, given infinite training data, it will learn the target concept perfectly. LR and Naive Bayes on the other hand cannot represent all possible Boolean functions perfectly. NB is in fact a probabilistic classifier. It does not represent a target concept.

5. (2 points) A decision tree is the smallest possible representation of the target concept. In other words, there exists no other representation that is smaller than the decision tree. True of False. Explain.

   False: A decision tree representation of a SAT problem can be exponential in size, while the SAT problem can be specified in linear space.

PROBLEM 2: LINEAR REGRESSION (10 points)

Consider a linear regression problem $y = w_1 x + w_0$, with a training set having m examples $(x_1, y_1), \ldots, (x_m, y_m)$. Suppose that we wish to minimize the mean $5 - th$ degree error (loss function) given by:

$$Loss = \frac{1}{m} \sum_{i=1}^{m} (y_i - w_1 x_i - w_0)^5$$

1. (3 points) Calculate the gradient with respect to the parameter w_1. Hint: $\frac{dx^k}{dx} = kx^{k-1}$.

   The gradient is :

   $$\frac{1}{m} \sum_{i=1}^{m} 5(y_i - w_1 x_i - w_0)^4 (-x_i)$$

2. (4 points) Write down pseudo-code for online gradient descent on w_1 for this problem. (You do not need to include the equations for w_0)

   Initialize w_0 and w_1. Assume a value for η
   Repeat Until Convergence
   For each example (x_i, y_i) in the training set do
   $w_1 = w_1 + \eta(y_i - w_1 x_i - w_0)^4 (x_i)$
   End For
   End Repeat

3. (3 points) Give one reason in favor of online gradient descent compared to batch gradient descent, and one reason in favor of batch over online.

   Online:

   - **it is often much faster, especially when the training set is redundant (contains many similar data points)**
   - **it can be used when there is no fixed training set (streaming data)**
   - **the noise in the gradient can help to escape from local minima**

   Batch:

   - **Less random and therefore easier to check for convergence**
   - **Often easier to set the learning rate η**

PROBLEM 3: CLASSIFICATION (10 points)

Imagine that you are given the following set of training examples. Each feature can take on one of three nominal values: a, b, or c.

F1	F2	F3	Category
a	c	a	+
c	a	c	+
a	a	c	−
b	c	a	−
c	c	b	−

1. (5 points) How would a Naive Bayes system classify the following test example? Be sure to show your work.

   F1 = a, F2 = c , F3 = b

   $P(Class = +) = 2/5$
   $P(F1 = a|Class = +) = 1/2, P(F1 = a|Class = -) = 1/3$
   $P(F1 = b|Class = +) = 0, P(F1 = b|Class = -) = 1/3$
   $P(F1 = c|Class = +) = 1/2, P(F1 = c|Class = -) = 1/3$
   $P(F2 = a|Class = +) = 1/2, P(F2 = a|Class = -) = 1/3$
   $P(F2 = c|Class = +) = 1/2, P(F2 = c|Class = -) = 2/3$
   $P(F3 = a|Class = +) = 1/2, P(F3 = a|Class = -) = 1/3$
   $P(F3 = b|Class = +) = 0, P(F3 = b|Class = -) = 1/3$
   $P(F3 = c|Class = +) = 1/2, P(F3 = c|Class = -) = 1/3$
   $P(F1 = a|Class = +)P(F2 = c|Class = +)P(F3 = b|Class = +)P(Class = +) = (1/2)(1/2)(0)(2/5) = 0$
   $P(F1 = a|Class = -)P(F2 = c|Class = -)P(F3 = b|Class = -)P(Class = -) = (1/3)(2/3)(1/3)(2/3)$
   The class is given by the equation: $argmax_c \prod_j P(F_j|C = c)P(C = c)$. Therefore the class is −.

2. (5 points) Describe how a 3-nearest-neighbor algorithm would classify the test example given above.

   The 3-nearest neighbor algorithm will use a distance measure to compute three nearest points to the test example and assign to it the majority class.

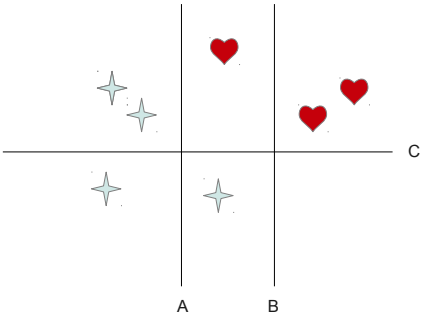
   Since the data is nominal, we will use hamming distance as the distance metric. Hamming distance is the number of attributes on which the two examples agree on.

   Hamming distance of the test point to the first five examples is: 2, 0, 1, 1, 2.

   Thus, we will assign to it the majority class of the first, third and fifth or that of first, fourth or fifth example. Both options yield majority class −.

PROBLEM 4: BOOSTING, LOGISTIC REGRESSION and POINT ESTIMATION(10 points)

1. (4 points) The diagram shows training data for a binary concept where positive examples are denoted by a heart. Also shown are three decision stumps (A, B and C) each of which consists of a linear decision boundary. Suppose that AdaBoost chooses A as the first stump in an ensemble and it has to decide between B and C as the next stump. Which will it choose? Explain. What will be the ϵ and α values for the first iteration?



It will choose B because the only example mis-classified by A is correctly classified by B (the misclassified examples are assigned higher weight in the next iteration).

In the first iteration $\epsilon = 1/7$

and $\alpha = \ln\{\frac{\epsilon}{1-\epsilon}\} = \ln(6)$.

2. (3 points) When learning a logistic regression classifier, you run gradient ascent for 50 iterations with the learning rate, $\eta = 0.3$, and compute the conditional log-likelihood $J(\theta)$ after each iteration (where θ denotes the weight vectors). You find that the value of $J(\theta)$ increases quickly then levels off. Based on this, which of the following conclusions seems most plausible? Explain your choice in a sentence or two.

   A. Rather than use the current value of η, it'd be more promising to try a larger value for the learning rate (say $\eta = 1.0$).

   B. $\eta = 0.3$ is an effective choice of learning rate.

   C. Rather than use the current value of η, it'd be more promising to try a smaller value (say $\eta = 0.1$).

B is the correct answer. $\eta = 0.3$ is an effective choice of learning rate because the algorithm seems to be converging after a few iterations.

3. (3 points) (**Point Estimation**) You are given a coin and a thumbtack and you put Beta priors $Beta(100, 100)$ and $Beta(1, 1)$ on the coin and thumbtack respectively. You perform the following experiment: toss both the thumbtack and the coin 100 times. To your surprise, you get 60 heads and 40 tails for both the coin and the thumbtack. Are the following two statements true or false.

   - The MLE estimate of both the coin and the thumbtack is the same but the MAP estimate is not.
   - The MAP estimate of the parameter θ (probability of landing heads) for the coin is greater than the MAP estimate of θ for the thumbtack.

   Explain your answer mathematically.

The first statement is True. MLE estimate = 6/10 for both cases. The MAP estimates are not because the beta priors are different.

The second statement is False.

MAP estimate for the coin: $\frac{\alpha_H+\beta_H-1}{\alpha_H+\beta_H+\alpha_T+\beta_T-2} = 60 + 100 - 160 + 100 + 40 + 100 - 2 = 159/298$

MAP estimate for the thumbtack: $\frac{\alpha_H+\beta_H-1}{\alpha_H+\beta_H+\alpha_T+\beta_T-2} = 60 + 1 - 160 + 1 + 40 + 1 - 2 = 60/100$

Therefore, the MAP estimate of thumbtack is greater than that of the coin.

PROBLEM 5: SUPPORT VECTOR MACHINES (10 points)

Consider the following 1-dimensional data:

x	-3	0	1	2	3	4	5
Class	$-$	$-$	$+$	$+$	$+$	$+$	$+$

1. (3 points) Draw the decision boundary of a linear support vector machine on this data and identify the support vectors.

   The decision boundary lies at 0.5. The support vectors are $x = 0$ and $x = 1$.

2. (3 points) Give the solution parameters w and b where the linear form is $wx + b$.

   $w(x = 0) + b = -1$ **and** $w(x = 1) + b = 1$ **Therefore,** $b = -1$ **and** $w = 2$

3. (2 points) Suppose we have another instance $(x = -5, Class = +)$. What kernel will you use to classify the training data perfectly.

   We should use a polynomial kernel with degree 2. (Intuitively we will need a circle around -3 and 0).

4. (2 points) Calculate the leave one out cross validation error for the data set.

   There are two possible answers here:

   The number of examples misclassified in leave one out cross validation are either $0, 1, 0, 0, 0, 0, 0$ **or** $0, 1, 1, 0, 0, 0, 0$**. Therefore the cross validation error is** $1/7$ **or** $2/7$**.**

PROBLEM 6: Complexity Analysis (10 points)

Provide pseudo-code for an extension of the standard decision tree algorithm that does a k-move lookahead. We discussed a 1-move lookahead algorithm in class. Assume that you have N instances in the training set and each instance is defined by M binary attributes. Provide a computational analysis of the time-complexity of the algorithm with a 2-move lookahead. Hypothesize whether this algorithm will perform better or worse than the standard decision tree and give a qualitative argument why.

The pseudo code for k-move lookahead is given below:

S: **Training data**
X_i **indexes the attributes.**

Procedure $ChooseBestAttribute(S, k)$**:**
  Return an attribute X_j **having the lowest** $Error(S, X_j, k-1)$

Procedure $Error(S, X_j, k)$
  If $k = 0$ **then Return total error if we split on feature** X_j
  Else
    $e = \infty$
    For every attribute X_k **we have not yet split on do**
      $e = min(e, Error(S|X_j = True, X_k, k-1) + Error(S|X_j = False, X_k, k-1))$
    Return e

At each step in the decision tree induction, the complexity of 1-move lookahead is $O(NM)$ **where** N **is the number of training examples and** M **is the dimensionality of the data (or the number of attributes). Since the recursive algorithm given above will be called** M **times for 2-move lookahead, the complexity of 2-move lookahead will be** $O(NM^2)$**. The complexity of** k**-move lookahead is** $O(NM^k)$

On average, the algorithm is likely to yield a smaller tree as compared with the 1-move look ahead algorithm because it has more knowledge about future splits. When $k \geq M$ **for example, we are guaranteed to get the smallest possible decision tree.**