Midterm: CS 6375
Spring 2015
Solutions

The exam is closed book. You are allowed a one-page cheat sheet. Answer the questions in the spaces provided on the question sheets. If you run out of room for an answer, use an additional sheet (available from the instructor) and staple it to your exam.

- **NAME** ______________________________
- **UTD-ID if known** ______________________________

Question	Points	Score
Decision Trees	10	
Linear Classifiers	10	
Neural Networks	20	
Support Vector Machines	10	
Point Estimation	10	
Naive Bayes	10	
Instance Based Learning	10	
Extra Credit: Linear Regression	20	
Total:	100	

Question 1: Decision Trees (10 points)

Consider the training dataset given below. A, B, and C are the attributes and Y is the class variable.

X	A	B	C	Y
	0	1	0	Yes
1	1	0	1	Yes
	0	0	0	No
6	1	0	1	No
	0	1	1	No
	1	1	0	Yes

(a) (2 points) Can you draw a decision tree having 100% accuracy on this training set? If you answer is yes, draw the decision tree in the space provided below. If your answer is no, explain why?

Solution: No. Because Examples #2 and #4 have the same feature vector but different classes.

(b) (3 points) Which attribute among A, B and C has the highest information gain? Explain your answer.

Solution:
$IG(A) = H(3,3) - \frac{1}{2}H(2,1) - \frac{1}{2}H(2,1)$
$IG(B) = H(3,3) - \frac{1}{2}H(2,1) - \frac{1}{2}H(2,1)$
$IG(C) = H(3,3) - \frac{1}{2}H(2,1) - \frac{1}{2}H(2,1)$
Therefore, all three attributes have the same information gain.

(c) (3 points) You are given a collection of datasets that are linearly separable. Is it always possible to construct a decision tree having 100% accuracy on such datasets? True or False. Explain your answer.

Solution: True. A decision tree can represent any Boolean function. Since a linear classifier can be represented using a Boolean function and any linear classifier will have 100% accuracy on the training dataset, the decision tree representing the Boolean function associated with the linear classifier will also have 100% accuracy on the training dataset.

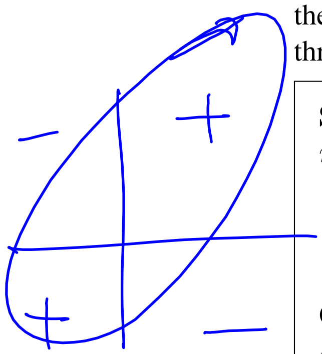
(d) (2 points) You are given two decision trees that represent the same target concept. In other words, given the same input the two decision trees will always yield the same output. Does it imply that the two decision trees have the same number of nodes? True or False. Explain your answer.

Solution: False. Consider two decision trees representing the target concept $(x_1 \vee x_2) \wedge (x_2 \vee x_3)$. Draw one along the ordering x_2, x_1 and x_3 and draw the second one along the ordering x_1, x_2, and x_3. The first decision tree has 7 nodes, while the second one has 9 nodes. (Please verify).

Question 2: Linear Classifiers (10 points)

Recall that a linear threshold function or a linear classifier is given by: If $(w_0 + \sum_i w_i x_i) > 0$ then class is positive, otherwise it is negative. Assume that 1 is true and 0 is false.

(a) (5 points) Consider a function over n Binary features, defined as follows. If at least k variables are false, then the class is **positive**, otherwise the class is **negative**. Can you represent this function using a linear threshold function. If your answer is **YES**, then give a precise numerical setting of the weights. Otherwise, clearly explain, why this function cannot be represented using a linear threshold function.



Solution: YES. Consider the following setting of weights: $w_i = -1$ for all i and $w_0 = n - k + 0.5$. If at least k variables are false, then $\sum_i w_i x_i \geq -n + k$. Therefore:

$$w_0 + \sum_i w_i x_i \geq -n + k + n - k + 0.5 \geq 0.5 > 0$$

On the other hand, if fewer than k variables are false, then $\sum_i w_i x_i \leq -(n - (k-1)) = -n + k - 1$. Therefore:

$$w_0 + \sum_i w_i x_i \leq -n + k - 1 + n - k + 0.5 \leq -0.5 < 0$$

(b) (5 points) Consider a linear threshold function over n real-valued features having $w_0 = 0$ (namely the bias term is zero). Can you always represent such a function using a linear threshold function having only $n - 1$ features? Answer **YES** or **NO** and briefly explain your answer. Note that no credit will be given if your explanation is incorrect.

Solution: NO. Because we have to find a weight vector $(v_0, v_1, \ldots, v_{n-1})$ such that

$$\sum_{i=1}^{n} w_i x_i = v_0 + \sum_{i=1}^{n-1} v_i x_i$$

This is not possible because the left hand side has n variables and the right hand side has only $n - 1$ variables.

Question 3: Neural Networks (20 points)

(a) (10 points) Draw a neural network that represents the function $f(x_1, x_2, x_3)$ defined below. You can only use two types of units: linear units and sign units. Recall that the linear unit takes as input weights and attribute values and outputs $w_0 + \sum_i w_i x_i$, while the sign unit outputs $+1$ if $(w_0 + \sum_i w_i x_i) > 0$ and -1 otherwise.

x_1	x_2	x_3	$f(x_1, x_2, x_3)$
0	0	0	10
0	0	1	-5
0	1	0	-5
0	1	1	10
1	0	0	-5
1	0	1	10
1	1	0	10
1	1	1	10

function

Note that to get full credit, you have to write down the precise numeric weights (e.g., -1, -0.5, $+1$, etc.) as well as the precise units used at each hidden and output node.

Solution: Several solutions are possible. The simplest solution is to have two hidden layers and one output node. The first hidden layer has three nodes with each unit being a "sign unit", the second layer has one node, which is also a sign unit and the output node is a linear unit. The three hidden nodes in the first hidden layer represent the following functions respectively:

1. If $\neg x_1 \wedge \neg x_2 \wedge x_3$ evaluates to true, output a $+1$ otherwise output a -1. An example setting of weights is $w_1 = -1$, $w_2 = -1$ and $w_3 = 1$ and $w_0 = -0.5$.
2. If $\neg x_1 \wedge x_2 \wedge \neg x_3$ evaluates to true, output a $+1$ otherwise output a -1. An example setting of weights is $w_1 = -1$, $w_2 = 1$ and $w_3 = -1$ and $w_0 = -0.5$.
3. If $x_1 \wedge \neg x_2 \wedge \neg x_3$ evaluates to true, output a $+1$ otherwise output a -1. An example setting of weights is $w_1 = 1$, $w_2 = -1$ and $w_3 = -1$ and $w_0 = -0.5$.

The input to the hidden unit in the second hidden layer is the output of all hidden nodes in the first hidden layer. It represents the following function:

1. If at least one of the inputs is a $+1$, the unit outputs a $+1$, otherwise it outputs a -1. It thus represents an OR node. An example setting of weights is $w_0 = -0.5$ and $w_1 = w_2 = w_3 = 1$.

The input to the output node is the output of the node in the second hidden layer. It represents the following function

1. If the input is $+1$ then output -5, otherwise output 10. An example setting of weights is $w_0 = 5/2$ and $w_1 = -15/2$.

(b) (10 points) Derive a gradient descent training algorithm for the following "special" unit. The "special" unit takes as input a vector $(x_1, \ldots, x_n)$ of feature values and outputs o, where o is given by the following equation:

$$o = w_0 + \sum_{i=1}^{n} w_i(x_i + x_i^2 + x_i^3)$$

Here, $w_0, w_1, \ldots, w_n$ are the parameters which you have to learn from the training dataset D. Use the batch gradient descent approach. Use the following notation: $x_{i,d}$ denotes the value of the i-th attribute (feature) in the d-th example in the training set D.

Solution: The error function is $\frac{1}{2}\sum_{d=1}^{m}(t_d - o_d)^2$ where m is the number of data points and t_d is the class of the d-th example.
The gradient of the error function w.r.t. w_i is (please verify this):

$$-\sum_{d=1}^{m}(t_d - o_d)(x_{i,d} + x_{i,d}^2 + x_{i,d}^3)$$

The partial derivative of the error function w.r.t. w_0 is (please verify this):

$$-\sum_{d=1}^{m}(t_d - o_d)$$

Therefore, the batch gradient descent algorithm is given by:

```
• Initialize all $w_i$'s and $w_0$ to some finite randomly generated real numbers
• Repeat Until convergence
    – Initialize $\Delta w_i = 0$ for $i = 0$ to $n$
    – **for** each data point indexed by $d$ do
        * Calculate $o_d = w_0 + \sum_{i=1}^{n} w_i(x_{i,d} + x_{i,d}^2 + x_{i,d}^3)$
        * $\Delta w_0 = \Delta w_0 + (t_d - o_d)$
        * **for** each attribute indexed by $i$ do
            · $\Delta w_i = \Delta w_i + (t_d - o_d)(x_{i,d} + x_{i,d}^2 + x_{i,d}^3)$
    – **for** each attribute indexed by $i$ do
        * $w_i = w_i + \eta\Delta w_i$ where $\eta$ is the learning rate
• **return** $(w_0, w_1, \ldots, w_n)$
```

Question 4: Support Vector Machines (10 points)

Consider the training data given below (X is the attribute, and Y is the class variable)

X	Y
-2	-1
-1	+1
1	+1
3	-1

(a) (3 points) Assume that you are using a linear SVM. Let $\alpha_1, \alpha_2, \alpha_3$ and α_4 be the lagrangian mutlipliers for the four data points. Write the precise expression for the lagrangian dual optimization problem that needs to be solved in order to compute the values of $\alpha_1, \ldots, \alpha_4$ for the dataset given above.

Solution: The dual formulation is:

$$maximize \sum_{i=1}^{4} \alpha_i - \frac{1}{2} \sum_{i=1}^{4} \sum_{j=1}^{4} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

subject to the constraints: $\alpha_1, \alpha_2, \alpha_3, \alpha_4 \geq 0$ and $-\alpha_1 + \alpha_2 + \alpha_3 - \alpha_4 = 0$. The quantity $y_i y_j \mathbf{x}_i^T \mathbf{x}_j$ for different values of i and j is given by the cells of the following matrix

$$\begin{pmatrix} 4 & -2 & 2 & -6 \\ -2 & 1 & -1 & 3 \\ 2 & -1 & 1 & -3 \\ -6 & 3 & -3 & 9 \end{pmatrix}$$

(b) (2 points) Do you think, you will get zero training error on this dataset if you use linear SVMs (Yes or No)? Explain your answer.

Solution: No, because the data is not linearly separable. Plot it on a line and see for yourself.

The dataset is replicated here for convenience (X is the attribute, and Y is the class variable).

X	Y
-2	-1
-1	+1
1	+1
3	-1

(c) (3 points) Now assume that you are using a quadratic kernel: $(1 + x_i^T x_j)^2$. Again, let $\alpha_1, \alpha_2, \alpha_3$ and α_4 be the lagrangian mutlipliers for the four data points. Write the precise expression for the lagrangian dual optimization problem that needs to be solved in order to compute the values of $\alpha_1, \ldots, \alpha_4$ for the dataset and the quadratic kernel given above.

Solution:

$$maximize \sum_{i=1}^{4} \alpha_i - \frac{1}{2} \sum_{i=1}^{4} \sum_{j=1}^{4} \alpha_i \alpha_j y_i y_j (1 + \mathbf{x}_i^T \mathbf{x}_j)^2$$

subject to the constraints: $\alpha_1, \alpha_2, \alpha_3, \alpha_4 \geq 0$ and $-\alpha_1 + \alpha_2 + \alpha_3 - \alpha_4 = 0$. The quantity $y_i y_j (1 + \mathbf{x}_i^T \mathbf{x}_j)^2$ for different values of i and j is given by the cells of the following matrix

$$\begin{pmatrix} 25 & -9 & -1 & 25 \\ -9 & 4 & 0 & -4 \\ -1 & 0 & 4 & -16 \\ 25 & -4 & -16 & 100 \end{pmatrix}$$

(d) (2 points) Do you think, you will get zero training error on this dataset if you use the quadratic kernel (Yes or No)? Explain your answer.

Solution: Yes, the quadratic kernel will correctly classify this data because it will add the feature x^2 to the dataset. You can plot the points in 2-D with this added dimension and see for yourself.

Question 5: Point Estimation (10 points)

Given that it is virtually impossible to find a suitable "date" for boring, geeky computer scientists, you start a dating website called "www.csdating.com." Before you launch the website, you do some tests in which you are interested in estimating the failure probability of a "potential date" that your website recommends. In order to do that, you perform a series on experiments on your classmates (friends). You ask them to go on "dates" until they find a suitable match. The number of failed dates, k, is recorded.

(a) (5 points) Given that p is the failure probability, what is the probability of k failures before a suitable "match" is found by your friend.

Solution:

$$p^k(1-p)$$

(b) (5 points) You have performed m independent experiments of this form (namely, asked m of your friends to go out on dates until they find a suitable match), recording $k_1, \ldots, k_m$. Estimate the most likely value of p as a function of m and $k_1, \ldots, k_m$.

Solution:

$$\begin{aligned} LL &= \sum_{i=1}^{m} \ln(p_i^k(1-p)) \\ &= \sum_{i=1}^{m} k_i \ln(p) + \ln(1-p) \\ &= m\ln(1-p) + \sum_{i=1}^{m} k_i \ln(p) \end{aligned}$$

Taking the derivative of LL w.r.t. to p and setting it to zero we get:

$$\frac{-m}{1-p} + \sum_{i=1}^{m} \frac{k_i}{p} = 0$$

Solving for p, we get:

$$p = \frac{\sum_{i=1}^{k} k_i}{m + \sum_{i=1}^{m} k_i}$$

Question 6: Naive Bayes (10 points)

Consider the following training dataset with two real-valued inputs X_1 and X_2 and a class variable Y that takes two values, $+$ and $-$.

X_1	X_2	Y
0	5	$+$
2	9	$+$
1	3	$-$
2	4	$-$
3	5	$-$
4	6	$-$
5	7	$-$

We assume that the data is generated by a Gaussian naive Bayes model, and we will use the data to develop a naive Bayes classifier.

(a) (5 points) Assuming that the variance is independent of the class, estimate the parameters of the Gaussian Naive Bayes model from the given dataset.

Solution: The parameters are $P(Y = +) = 2/7$ and $P(Y = -) = 5/7$ The mean of the Gaussian representing $P(X_1|Y = +)$ is $(0 + 2)/2 = 1$ and with $P(X_1|Y = -)$ is $(1 + 2 + 3 + 4 + 5)/5 = 3$. Since the variance is independent of the class, the variance associated with both Gaussians is the variance of the vector $(0, 2, 1, 2, 3, 4, 5)$. The mean of the Gaussian representing $P(X_2|Y = +)$ is $(5 + 9)/2 = 7$ and with $P(X_2|Y = -)$ is $(3 + 4 + 5 + 6 + 7)/5 = 5$. Since the variance is independent of the class, the variance associated with both Gaussians is the variance of the vector $(5, 9, 3, 4, 5, 6, 7)$.

(b) (5 points) Assuming that the variance is independent of the features X_1 and X_2, estimate the parameters of the Gaussian Naive Bayes model from the given dataset.

Solution: The parameters are $P(Y = +) = 2/7$ and $P(Y = -) = 5/7$ The mean of the Gaussian representing $P(X_1|Y = +)$ is $(0 + 2)/2 = 1$ and with $P(X_1|Y = -)$ is $(1 + 2 + 3 + 4 + 5)/5 = 3$. The mean of the Gaussian representing $P(X_2|Y = +)$ is $(5 + 9)/2 = 7$ and with $P(X_2|Y = -)$ is $(3 + 4 + 5 + 6 + 7)/5 = 5$.

Since the variance is independent of the features, the variance associated with Gaussians representing $P(X_1|Y = +)$ and $P(X_2|Y = +)$ is the variance of the vector $(0, 2, 5, 9)$ while the variance associated with Gaussians $P(X_1|Y = -)$ and $P(X_2|Y = -)$ is the variance of the vector $(1, 2, 3, 4, 5, 3, 4, 5, 6, 7)$.

Question 7: Instance Based Learning (10 points)

Consider the training data given below. x is the attribute and y is the class variable.

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
y	A	A	A	A	B	A	A	A	A	B	B	B	B	A	B	B	B	B

(a) (2 points) What would be the classification of a test sample with $x = 4.2$ according to 1-NN ?

Solution: B

(b) (2 points) What would be the classification of a test sample with $x = 4.2$ according to 3-NN ?

Solution: A

(c) (3 points) What is the "leave-one-out" cross validation error of 1-NN. If you need to choose between two or more examples of identical distance, make your choice so that the number of errors is maximized.

Solution: 8 out of 18

(d) (3 points) What is the "leave-one-out" cross validation error of 17-NN. If you need to choose between two or more examples of identical distance, make your choice so that the number of errors is maximized.

Solution: 18 out of 18

Question 8: Extra Credit: Linear Regression (20 points)

Attempt this question, only after answering the rest of the questions. This question has three parts and each part will be graded on a binary scale; either you will get full points or no points at all.

Consider fitting the model: $\underline{y = w_0 + w_1 x}$ (x is 1-dimensional) using the squared loss function that we discussed in class:

$$J(w_0, w_1) = \sum_{i=1}^{m} (y_i - (w_0 + w_1 x_i))^2$$

where $[(x_1, y_1), \ldots, (x_m, y_m)]$ are the data points.

Unfortunately we did not keep the original data, but we did store the following five quantities (statistics) that we computed from the data:

1. $\overline{x}^{(m)} = \frac{1}{m}\sum_{i=1}^{m} x_i$
2. $\overline{y}^{(m)} = \frac{1}{m}\sum_{i=1}^{m} y_i$
3. $C_{xx}^{(m)} = \frac{1}{m}\sum_{i=1}^{m} (x_i - \overline{x}^{(m)})^2$
4. $C_{xy}^{(m)} = \frac{1}{m}\sum_{i=1}^{m} (x_i - \overline{x}^{(m)})(y_i - \overline{y}^{(m)})$
5. $C_{yy}^{(m)} = \frac{1}{m}\sum_{i=1}^{m} (y_i - \overline{y}^{(m)})^2$

(a) (5 points) What are the minimal set of statistics that we need to estimate w_1. Namely, which of the above five statistics do you need to estimate w_1. Explain your answer by giving a precise expression for w_1 in terms of your chosen statistics. No credit without correct explanation.

Solution: See class slides for the closed form solution for w_1, we can show that it is a ratio of covariance of X and Y, and variance of X. Therefore, we need $C_{xx}^{(m)}$ and $C_{xy}^{(m)}$.

(b) (5 points) What are the minimal set of statistics that we need to estimate w_0. Namely, which of the above five statistics do you need to estimate w_0. Explain your answer by giving a precise expression for w_0 in terms of your chosen statistics. No credit without correct explanation.

Solution: See class slides for the closed form solution for w_0, we can see that it depends on $\overline{x}^{(m)}$ and $\overline{y}^{(m)}$.

(c) (10 points) Suppose a new data point (x_{m+1}, y_{m+1}) arrives, and we want to update our sufficient statistics without looking at the old data, which we have not stored. Give precise expression for the new statistics in terms of the old statistics and the new data point. (This is useful for online learning.)

Solution:

$$\overline{x}^{(m+1)} = \frac{(m\overline{x}^{(m)} + x_{m+1})}{m+1}$$

$$\overline{y}^{(m+1)} = \frac{(m\overline{y}^{(m)} + y_{m+1})}{m+1}$$

$$\overline{C}_{xy}^{(m+1)} = \frac{x_{m+1}y_{m+1} + mC_{xy}^{(m+1)} + m\overline{x}^{(m)}\overline{y}^{(m)} - (m+1)\overline{x}^{(m+1)}\overline{y}^{(m+1)}}{m+1}$$

$C_{xx}^{(m+1)}$, $C_{yy}^{(m+1)}$: Skipped. Homework problem!

1 Appendix: Representing Arbitrary Boolean Functions using a Perceptron and a Neural Network

AND Function Let $\{X_1, \ldots, X_n\}$ be the binary input attributes taking values from the set $\{0, 1\}$. Let $f(X_1, \ldots, X_n)$ be an arbitrary AND function over the literals of the n attributes. Recall that a literal is a propositional variable or its negation. For example, the literals of X_3 are X_3 and $\neg X_3$ respectively. Without loss of generality, $X_1, \ldots, X_k$ be the positive literals and $X_{k+1}, \ldots, X_n$ be the negative literals in the AND function f. In other words, f is given by:

$$f(X_1, \ldots, X_n) = X_1 \wedge \ldots \wedge X_k \wedge \neg X_{k+1} \wedge \ldots \wedge \neg X_n$$

We can represent this using a perceptron (with sign unit) having the following weights: $w_0 = -k + 0.5$; $w_1 = \ldots = w_k = 1$ and $w_{k+1} = \ldots = w_n = -1$.

The output of this perceptron will be $+1$ if f is true and -1 otherwise.

OR Function Let g be an OR function over the literals of binary variables in the set $\{X_1, \ldots, X_n\}$. Without loss of generality, $X_1, \ldots, X_k$ be the positive literals and $X_{k+1}, \ldots, X_n$ be the negative literals in the OR function g. In other words, g is given by:

$$g(X_1, \ldots, X_n) = X_1 \vee \ldots \vee X_k \vee \neg X_{k+1} \vee \ldots \vee \neg X_n$$

We can represent this using a perceptron (with sign unit) having the following weights: $w_0 = n - k - 0.5$; $w_1 = \ldots = w_k = 1$ and $w_{k+1} = \ldots = w_n = -1$.

The output of this perceptron will be $+1$ if g is true and -1 otherwise.

Special Case: where X_i's take values from the set $\{+1, -1\}$ instead of $\{0, 1\}$.

For this case, we can represent the AND function using a perceptron (with sign unit) having the following weights: $w_0 = -n + 0.5$; $w_1 = \ldots = w_k = 1$ and $w_{k+1} = \ldots = w_n = -1$.

For this case, we can represent the OR function using a perceptron (with sign unit) having the following weights: $w_0 = n - 0.5$; $w_1 = \ldots = w_k = 1$ and $w_{k+1} = \ldots = w_n = -1$.

Representing Arbitrary Boolean functions using a Neural Network Express the Boolean function either in CNF or a DNF form. Recall that a CNF is a conjunction (AND) of clauses where a clause is a disjunction (OR) of literals. A DNF is a disjunction of conjunctive formulas where a conjunctive formula is a conjunction of literals.

For a DNF $c_1 \vee \ldots \vee c_t$: Use one hidden layer with t hidden nodes, each hidden node indexed by i represents the AND function given by the conjunctive formula c_i. The output node represents the OR function of the outputs of the hidden nodes. However, notice that the output of the hidden nodes is $\{+1, -1\}$ and therefore, we need to use the weights for the OR function given in the special case.

For a CNF $c_1 \wedge \ldots \wedge c_t$: Use one hidden layer with t hidden nodes, each hidden node indexed by i represents the OR function given by the clause c_i. The output node represents the AND function of the outputs of the hidden nodes. However, notice that the output of the hidden nodes is $\{+1, -1\}$ and therefore, we need to use the weights for the AND function given in the special case.

What if the output range is $\{a, b\}$ where a and b are integers?
Here, the function is:

```
If (DNF or CNF is satisfied) then output is a, else output is b.
```

Here, we use two hidden layers.

- The first hidden layer is the same as the hidden layer of the neural network for the CNF (or the DNF) described above. All units in this hidden layer are sign units.
- The second hidden layer has just one node and is this node is the same as the output node of the neural network for the CNF (or the DNF) described above. Again the unit associated with the node is the sign unit.
- The output node is a linear unit and has just one input, which comes from the only node in the second hidden layer. Note that the input is $+1$ if the CNF (or DNF) is satisfied and -1 otherwise. We have to change $+1$ to a and -1 to b. The following setting of weights for the linear unit achieves this: $w_0 = \frac{a+b}{2}$ and $w_1 = \frac{a-b}{2}$.