Name: _____

To complete this homework, you do not need to use any special software. In fact, please write out your answers on a paper copy of *this* document. Please do *not* write your answers in Word, Google docs, etc. or hand write them on some other piece of paper. You may wish to consult the readings and lecture notes (available online) related to this material (Lectures 14-18).

This assignment is **due on Wednesday, April 26th, 2017, at 11:59pm**. Please turn it in by uploading to Gradescope. Note: if you write out your answers by hand, you will need to scan or upload a photo of your work. Be sure to leave enough time for this.

Questions

1. **Decision Trees** This question is based on the dataset shown in Table 1.

A	B	T
a_1	b_1	-
a_1	b_2	-
a_2	b_1	+
a_2	b_2	+
a_2	b_2	+
a_3	b_1	+
a_3	b_2	-
a_3	b_2	-
a_3	b_1	+
a_3	b_2	-

Table 1: Labeled dataset: The target attribute is T and A and B are predictor attributes.

(a) (5 points) Calculate the entropy of A. Your final answer should be a single number. Show your work.

Solution:

A takes on three values, a_1, a_2 and a_3 each of these occuring 2, 3, and 5 times respectively.

$$Entropy(A) = -\left(\frac{2}{10}\lg\frac{2}{10} + \frac{3}{10}\lg\frac{3}{10} + \frac{5}{10}\lg\frac{5}{10}\right) \approx 1.485$$

`Entropy(A) 1.48547529723`

(b) (5 points) Calculate the information gain of segmenting on attribute A. Your final answer should be a single number. Show your work.

Solution:

$InfoGain(A, T, D) = Entropy(T, D) - SegmentEntropy(A, T, D)$.

T takes on two values, $+$ and $-$ and $p(+) = p(-) = \frac{1}{2}$.

$$Entropy(T) = -(\frac{1}{2}\lg\frac{1}{2} + \frac{1}{2}\lg\frac{1}{2}) = -\lg\frac{1}{2} = 1$$

Segmenting on A splits into three groups:

- $p(A = a_1) = \frac{2}{10}$, with all $T = -$, entropy is 0.
- $p(A = a_2) = \frac{3}{10}$, with all $T = +$, entropy is 0.
- $p(A = a_3) = \frac{5}{10}$, with entropy $= -(\frac{2}{5}\lg\frac{2}{5} + \frac{3}{5}\lg\frac{3}{5})$.

Putting it all together:

$$SegmentEntropy(A, T, D) = \frac{2}{10} \times 0 + \frac{3}{10} \times 0 + \frac{5}{10} \times \left(-(\frac{2}{5}\lg\frac{2}{5} + \frac{3}{5}\lg\frac{3}{5})\right)$$

```
Entropy(T) 1.0
SegmentEntropy(A, T) 0.485475297227
InfoGain(A, T) 0.514524702773
```

(c) (5 points) Suppose use this data to fit a decision tree using the algorithm described in class (and in the *DSB* reading). Draw the resulting tree.

Solution: A has higher information gain than B.

It would split on A into three groups. If $A = a_1$, it predicts "-"; if $A = a_2$, it predicts "+"; if $A = a_3$, it splits on B and if $B = b_1$, it predicts "+" otherwise "-".

2. **Perceptrons**

Suppose you are hired at a bank and your job is to design an automated credit approval program. The bank has provided you with a collection of past applications along with the decision that were made. To be more precise, you have data with the following attributes:

- X_1 the customer's years in current residence
- X_2 the customer's monthly income (thousands of dollars)
- X_3 the customer's gender (encoded as 1 for male and 0 for female)
- Y which is equal to 1 if the customer was approved for credit and -1 if the customer was denied credit.

Your task is to predict Y given predictor attributes X_1, X_2, and X_3. You decide to train a perceptron on the available data. It comes back with the following weights: $w_0 = -12$, $w_1 = 6$, $w_2 = 2$, and $w_3 = 2$.

Recall that a perceptron makes a prediction by calculating

$$h(x) = \text{sign}\left(\sum_{i=0}^{d} w_i x_i\right)$$

(a) (2 points) What prediction would the perceptron make for a woman making \$1,000/month who has lived 2 years in her current residence?

Solution: For this formulation, the perceptron computes:

$$\text{sign}\left(-12 + 6 \times X_1 + 2 \times X_2 + 2 \times X_2\right)$$

Approve credit because:

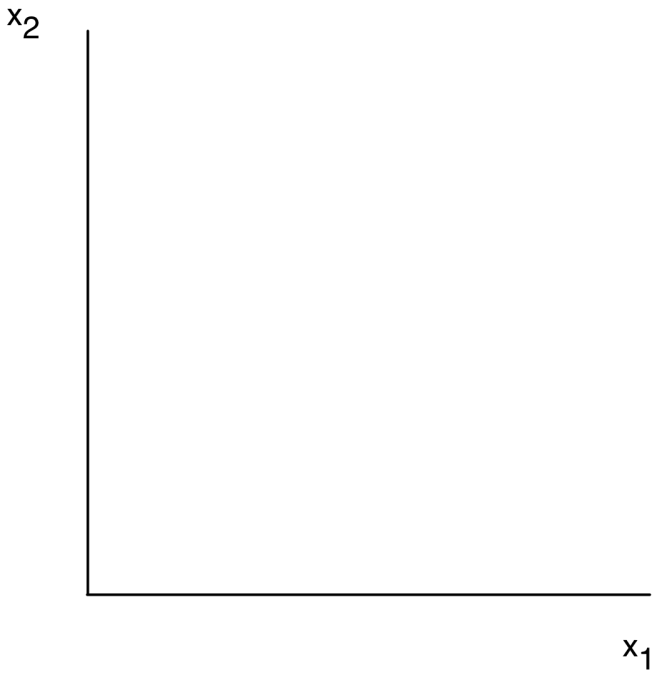
$$-12 + 6 \times 2 + 2 \times 1 + 2 \times 0 = 2 \geq 0$$

(b) (2 points) What prediction would the perceptron make for a man making \$4,500/month who just moved (0 years in current residence)?

Solution: Deny credit because:

$$-12 + 6 \times 0 + 2 \times 4.5 + 2 \times 1 = -1 < 0$$

(c) (5 points) Illustrate the decision boundary in the feature space below for *female* applicants. In other words, show which points will be labeled "Yes" and which ones labeled "No."

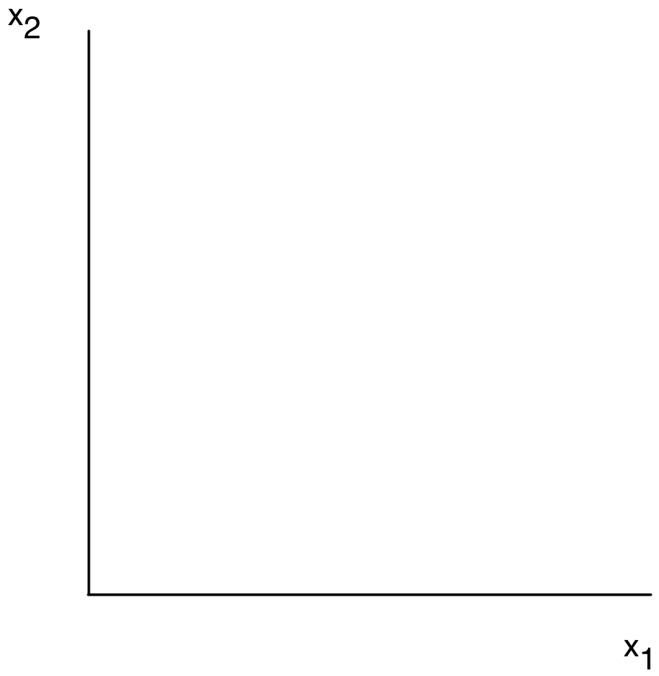


Solution:

$$\begin{aligned} x_2 &= \frac{w_1}{-w_2}x_1 + \frac{w_0}{-w_2} \\ &= \frac{6}{-2}x_1 + \frac{-12}{-2} \\ &= -3x_1 + 6 \end{aligned}$$

Everything above this line is classified as approve.

(d) (5 points) Illustrate the decision boundary in the feature space below for *male* applicants.



Solution:

$$\begin{aligned} x_2 &= \frac{w_1}{-w_2}x_1 + \frac{w_0 + w_3}{-w_2} \\ &= \frac{6}{-2}x_1 + \frac{-12+2}{-2} \\ &= -3x_1 + 5 \end{aligned}$$

Everything above this line is classified as approve.

(e) (1 point) What is the effect of the binary attribute gender? Describe its effect on the decision boundary. Does it change the slope? The intercept? Both? Neither?

Solution: Intercept.

3. **Avoiding Overfitting**

Machine learning can be used to discover genetic patterns associated with certain diseases. The dataset might consist of a collection of patients, some of whom are healthy and the rest suffer from a certain disease. For each patient, you might have a large large number of genomic measurements. For example, a common measurement is gene expression: a numerical measurement that indicates whether a particular gene is being

expressed or not. The machine learning task is to use the gene expression measurements to predict whether the patient has the disease.

The challenge is this: you might have a *small number of labelled examples* (say, 200 patients) and *a large number of predictor attributes* (say, 10,000 numerical gene expression attributes). When the number of examples is smaller than the number of attributes, any machine learning algorithm is at risk for overfitting.

Suppose you run the decision tree algorithm as described in class and you notice that it keeps splitting on attributes until each leaf has only 1 or zero examples. You're concerned that your tree has severely overfit the data.

So you modify the algorithm as follows. Your algorithm takes a new parameter k which will be used in a new stopping criterion. The criterion is that if on any branch of the tree, if it has fewer than k examples, the tree will stop segmenting and turn this branch into a leaf. The predicted value at this leaf will be the most common value of the target attribute among the examples.

The problem is that you don't know how to set k. If $k = 1$, your new stopping criterion will have no effect. If $k = 200$, you won't split the data at all and simply predict the most common value (healthy or diseased, whichever is more common in the labeled examples). You can think of k as an inverted measure of complexity: when k is small, you can build complex models; when k is large, you can only build simple models.

(a) (5 points) Explain how you might use your labeled data to figure out how to set k. You may wish to review the lecture on overfitting as well as the reading (*DSB* Ch. 5).

Solution: Here's a quick summary, but you are encouraged to review Ch. 5.

Split the data into SubTrain, Validate, and Test sets.

For each setting of k, train a tree on SubTrain. Then evaluate that tree on Validate and record its accuracy. Repeat this for each setting of k and keep track of which value of k achieves the highest accuracy on the Validation set. Let's call this k^*.

Now, train a tree with $k = k^*$ on a dataset which is the combination of *both* the SubTrain and Validate sets. Let's call this tree T. This is our final tree.

Finally, evaluate the accuracy of T on the test dataset. The sole purpose of the test dataset is to see how good T is. If it's good, you might choose to deploy T in the real-world (i.e., use it to predict the disease of new patients). If it's bad, you might call the experiment a failure and go back to the drawing board.

(b) (5 points) You only have a small amount of data. Explain how you might incorporate *cross validation* into your solution for part (a).

Solution: You can use cross validation any time you have a collection of data and you want to use it for two purposes (e.g., train and test).

Given the above, we actually want to use the data for three purposes: training, validation, and testing. Thus, we can do what is called nested cross validation, which is described in the book (p. 134-135). The basic idea is we use cross-validation to split the data into Train and Test, then we use cross-validation *again* to split Train into SubTrain and Validation.

Please write your answers to both parts on the next page.

Write your answer to the previous question here.

Extra space should you need it.