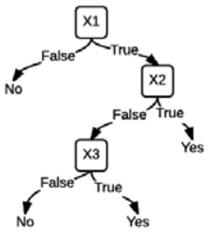
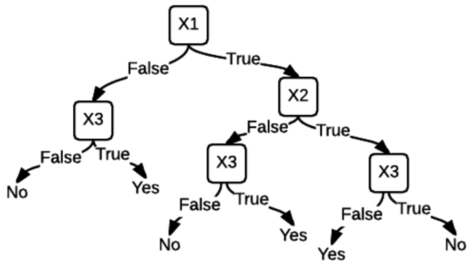
1 Decision Trees

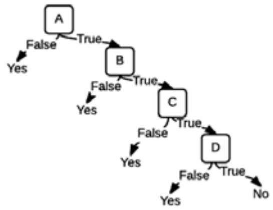
1. (a)



(b)



(c)



2. All of log's are in base 2.

(a) 2 * 2 * 3 * 4 = 48, so 2^48 = **281474976710656**. **2^39 functions are consistent with given data set.**

(b) p = 5/9, n = 4/9 ==> Entropy = -(5/9)log(5/9) - (4/9)log(4/9) = **0.991**

(c) Technology: No: p = 4/6, n = 2/6. So, -2/3log(2/3) – 1/3log(1/3) = 0.918295834

Yes: p = 1/3, n = 2/3. So, -1/3log(1/3) – 2/3log(2/3) = 0.918295834

6/9(0.9183) + 3/9(0.918295834) = 9/9(0.918295834) = 0.918295834

Information Gain = 0.991076067 – 0.9183 = **0.073**

Environment: No: p = 1/4, n = 3/4. So, -1/4log(1/4) – 3/4log(3/4) = 0.811278124

Yes: p = 4/5, n = 1/5. So, -4/5log(4/5) – 1/5log(1/5) = 0.721928095

4/9(0.811278124) + 5/9(0.721928095) = 0.761639219

Information Gain = 0.991076067 – 0.761639219 = **0.229**

Human: Like: p = 1/4, n = 3/4. So, -1/4log(1/4) – 3/4log(3/4) = 0.811278124

Not Care: p = 4/4, n = 0/4. So, -(1)log(1) – (0)log(0) = 0

Hate: p = 0/1, n = 1/1. So, -(0)log(0) – (1)log(1) = 0

4/9(0.811278124) + 4/9(0) + 1/9(0) = 0.360568055

Information Gain = 0.991076067 – 0.360568055 = **0.631**

Distance: 1: p = 1/2, n = 1/2. So, -1/2log(1/2) – 1/2log(1/2) = 1

2: p = 1/1, n = 0/1. So, -(1)log(1) – (0)log(0) = 0

3: p = 2/3, n = 1/3. So, -(2/3)log(2/3) – (1/3)log(1/3) = 0.918295834

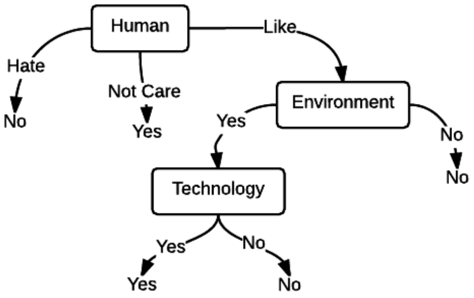
4: p = 1/3, n = 2/3. So, -(1/3)log(1/3) – (2/3)log(2/3) = 0.918295834

2/9(1) + 1/9(0) + 3/9(0.918295834) + 3/9(0.918295834) = 0.834419445

Information Gain = 0.991076067 – 0.834419445 = **0.157**

(d) I will choose the **Human** attribute because it has the highest Information Gain.

(e)



(f) First Example: **Yes**, Second Example: **No**, Third Example: **Yes**, two out of the three were correct, so the accuracy is 2/3 = **66.667%**

3. (a) Total Error: 1 – 5/9 = 4/9
   Technology: No: 1 – 4/6 = 2/6, Yes: 1 – 2/3 = 1/3, so 6/9(2/6) + (3/9)(1/3) = 3/9, so 4/9 – 3/9 = **1/9 = 0.111**
   Environment: Yes: 1 – 4/6 = 2/6, No: 1 – ¾ = ¼, so 5/9(1/3) + 4/9(¼) = 8/27, so 12/27 – 8/27 = **4/27 = 0.148**
   Human: Not Care: 1-4/4 = 0, Like: 1- ¾ = ¼, Hate: 1-1/1 = 0, so 0 + 4/9(¼) + 0 = 1/9, so 4/9-1/9 = **3/9 = 0.333**
   Distance: 1: 1-½ = ½, 2: 1-1/1 = 0, 3: 1-2/3 = 1/3, 4: 1-2/3 = 1/3, so 2/9(½) + 0 + 3/9(1/3) + 3/9(1/3) = 3/9
   So 4/9 – 3/9 = **1/9 = 0.111**
   (b) **Human** should be the attribute for the root for the decision tree.

2 Linear Classifiers

1. W = [1 1 0 1], b = -1
2. S

X1	X2	X3	X4	o	
0	0	0	1	1	0+0+0+1= 1-1 = 0>=0, output should be 1 (correct)
0	0	1	1	1	0+0+0+1= 1-1 = 0>=0, output should be 1 (correct)
0	0	0	0	-1	0+0+0+0= 0-1 = -1<0, output should be -1 (correct)
1	0	1	0	1	1+0+0+0= 1-1 = 0>=0, output should be 1 (correct)
1	1	0	0	1	1+1+0+0= 2-1 = 1>=0, output should be 1 (correct)
1	1	1	1	1	1+1+0+1= 3-1 = 2>=0, output should be 1 (correct)
1	1	1	0	1	1+1+0+0= 2-1 = 1>=0, output should be 1 (correct)

It correctly represents the dataset **100%**

3. W = [1 0 0 1], b = -1

X1	X2	X3	X4	o	
First Data Set					
1	0	1	1	1	1+0+0+1= 2-1 = 1>=0, output should be 1 (correct)
0	1	0	1	1	0+0+0+1= 1-1 = 0>=0, output should be 1 (correct)
0	0	1	0	-1	0+0+0+0= 0-1 = -1<0, output should be -1 (correct)
Second Data Set					
0	0	0	1	1	0+0+0+1= 1-1 = 0>=0, output should be 1 (correct)
0	0	1	1	1	0+0+0+1= 1-1 = 0>=0, output should be 1 (correct)
0	0	0	0	-1	0+0+0+0= 0-1 = -1<0, output should be -1 (correct)
1	0	1	0	1	1+0+0+0= 1-1 = 0>=0, output should be 1 (correct)
1	1	0	0	1	1+0+0+0= 1-1 = 0>=0, output should be 1 (correct)
1	1	1	1	1	1+0+0+1= 2-1 = 1>=0, output should be 1 (correct)
1	1	1	0	1	1+0+0+0= 1-1 = 1>=0, output should be 1 (correct)
Third Data Set					
0	1	0	0	-1	0+0+0+0= 0-1 = -1<0, output should be -1 (correct)
0	1	1	0	-1	0+0+0+0= 0-1 = -1<0, output should be -1 (correct)
0	1	1	1	1	0+0+0+1= 1-1 = 0>=0, output should be 1 (correct)
1	0	0	0	1	1+0+0+0= 1-1 = 1>=0, output should be 1 (correct)
1	0	0	1	1	1+0+0+1= 2-1 = 1>=0, output should be 1 (correct)
1	1	0	1	1	1+0+0+1= 2-1 = 1>=0, output should be 1 (correct)

3 Experiments

1. (a) Approaches or choices made. I first decided to create a class called data that would pretty much take all the data from the file and parse it out to a List of Entry's where an Entry represents one line in the data files given. Then I took the List of Entry's, asked the six questions for each entry, and stored their result along with their label as a List of TrainingData. Each TrainingData stores the results for one line in the data file given. I then pass the TrainingData by reference into the constructor of the DecisionTree. I got to a point where I had two

TrainingData objects in the list had the same exact results for the six questions, but had a different label which caused a stackoverflow. So what I did is found out if this happened and set the label as the one that occurred the most of the duplicates. In my Decision Tree Class, I had a LeftTree and a RightTree. The LeftTree represented sub-tree of the question answered as "No" and the RightTree represented the sub-tree of question answered as "Yes". Each DecisionTree had a Value that could be nullable. It was null if it was not a leaf, and if it was a leaf it had the value it should for the specific leaf. There was also a bool value for each DecisionTree that said if the Tree was a Leaf or not (called "IsLeaf"). To be able to make sure I was getting the correct data passed down to each sub-tree, I did a LINQ statement on the training data. Once I was completed making the full DecisionTree from the TrainingData List, I realized that my tree was not collapsed correctly. What I mean is, at the far bottom left of the tree, beneath one of the features, both of its leaves were the same value. I then decided to create a method in DecisionTee that collapsed the tree to exactly how it should look. When it came to determining what would be the first name, middle name, and last name, I obviously made the first name until a space as the first name. The last name was the very last name on each line. The middle name was everything in between the first and last name. So spaces pretty much determined what the names were. For example, for "- Willem van der Poel", "Willem" is the first name, "van der" is the middle name, and "Poel" was the last name.

(b) Do they have more than two spaces in their name?
Is there was a period anywhere in their first name?
Is there was a hyphen in their last name?
Is the number of letters in their first name even?

(c) The error of my decision tree on the Training Data is: **4.944%**

(d) The error of my decision tree on the Test Data is: **3.604%**

(e) The maximum depth of my decision tree is: **6**

2. For this section, I had to clone my code and add my additional 14 features to my code. I added the following features:

```
First name has period
Middle name has period
First name has hyphen
Last name has hyphen
First name has even letters
Middle name has even letters
First name is bigger than four letters
Last name is bigger than four letters
Last name starts and ends with the same letter
Middle name starts and ends with the same letter
Second letter of last name is a vowel
Third letter of first name is a vowel
Third letter of last name is a vowel
The entire name has at least three spaces
```

(a) Depth set to 1: 82.658% Accuracy
Depth set to 2: 88.063% Accuracy
Depth set to 3: 89.189% Accuracy
Depth set to 4: 92.793% Accuracy
Depth set to 5: 94.144% Accuracy
Depth set to 10: 83.333% Accuracy
Depth set to 15: 83.333% Accuracy
Depth set to 20: 82.658% Accuracy

(b) Accuracy on Test Data: 96.396%

(c) The depth limited tree has a much better performance than the full decision tree because as you get deeper and deeper into the tree (as in 20 deep), the information gain approaches 0. When all the features approach 0 for their information gain, it does not help the accuracy of the data. In fact, it actually makes the accuracy worse. I do think that limiting the depth is a good idea because of these reasons and because the accuracy of the tree will be a higher value.

4 Decision Lists

For (X1, X2…..Xn), the b1, b2….bn+1, where b1 to bn is 1 and bn+1 is 0. The Wn values are all 1 and the b = -1. All the b1's to bn's results will always be 1 with the set w and b because all the values in the w vector are 1. There is at least one 1 result in b1 to bn. Bn+1 will always return 0 in this because all the xn's will be 0's and all the wn's are 1's. So all zeros multiplied by all ones will add up to 0 and 0 will not be greater than the bias of -1.