Artificial Intelligence – Assignment 5

1. [25 pts.] Assume that you are given the set of labeled training examples below, where each of three features has four possible values: a, b, c, or d. You choose to apply the ID3 decision tree induction algorithm to this data.

	F1	F2	F3	Output
ex1	a	a	b	-
ex2	b	c	d	+
ex3	b	b	a	+
ex4	c	c	a	-
ex5	a	a	b	+
ex6	c	d	c	-
ex7	c	b	d	-

⇨ 1. I(T) = $\sum_{i=1}^{\{k\}} \left[\frac{|T_{C_i}|}{T}\right] X \log_2 \frac{|T_{C_i}|}{T}$

I(T) = - 3/7$\log_2 3/7$ - 4/7$\log_2 4/7$ = 0.523 + 0.461 = 0.984

2. Test F1:

1. I($T_{\{F1 <-- a\}}$} = -1/2$\log_2 1/2$ - 1/2$\log_2 1/2$ = ½ + ½ = 1
2. I($T_{\{F1 <-- b\}}$} = -2/2$\log_2 2/2$ = 0
3. I($T_{\{F1 <-- c\}}$} = -0/3$\log_2 0/3$ = 0

I(F1, T) = 2/7. I($T_{\{F1 <-- a\}}$} + 2/7. I($T_{\{F1 <-- b\}}$} + 3/7 I($T_{\{F1 <-- c\}}$}
= 2/7.1 + 0 + 0
= 0.285

Gain = I(T) – I(F1, T) = 0.984 – 0.285 = 0.699

3. Test F2:

1. I($T_{\{F2 <-- a\}}$} = -1/2$\log_2 1/2$ - 1/2$\log_2 1/2$ = ½ + ½ = 1
2. I($T_{\{F2 <-- b\}}$} = -1/2$\log_2 1/2$ - 1/2$\log_2 1/2$ = ½ + ½ = 1
3. I($T_{\{F2 <-- c\}}$} = -1/2$\log_2 1/2$ - 1/2$\log_2 1/2$ = ½ + ½ = 1
4. I($T_{\{F2 <-- d\}}$} = -0/1$\log_2 0/1$ = 0

I(F2, T) = 2/7. I($T_{\{F2 <-- a\}}$} + 2/7. I($T_{\{F2 <-- b\}}$} + 2/7 I($T_{\{F2 <-- c\}}$} + 1/7 I($T_{\{F2 <-- d\}}$}
= 2/7.1 + 2/7.1 + 2/7.1 + 1/7.0
= 0.857

Gain = I(T) – I(F2, T) = 0.984 – 0.857 = 0.127

4. Test F3:

1. $I(T_{\{F3 \leftarrow a\}}) = -1/2\log_2 1/2 - 1/2\log_2 1/2$ = ½ + ½ = 1
2. $I(T_{\{F3 \leftarrow b\}}) = -1/2\log_2 1/2 - 1/2\log_2 1/2$ = ½ + ½ = 1
3. $I(T_{\{F3 \leftarrow c\}}) = -0/1\log_2 0/1 = 0$
4. $I(T_{\{F3 \leftarrow d\}}) = -1/2\log_2 1/2 - 1/2\log_2 1/2$ = ½ + ½ = 1

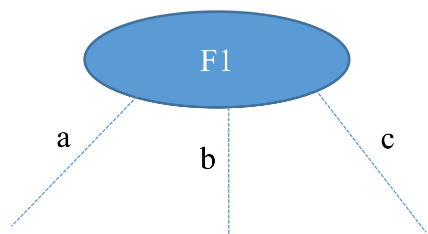
$I(F3, T) = 2/7.\ I(T_{\{F3 \leftarrow a\}}) + 2/7.\ I(T_{\{F3 \leftarrow b\}}) + 0/7\ I(T_{\{F3 \leftarrow c\}}) + 2/7\ I(T_{\{F3 \leftarrow d\}})$
$= 2/7.1 + 2/7.1 + 0 + 2/7.1 = 6/7 = 0.857$

Gain = I(T) – I(F3, T) = 0.984 – 0.857 = 0.127

Comparing Gains:

Gain(F1) > Gain(F2) = Gain(F3) :- 0.699 > 0.127 = 0.127

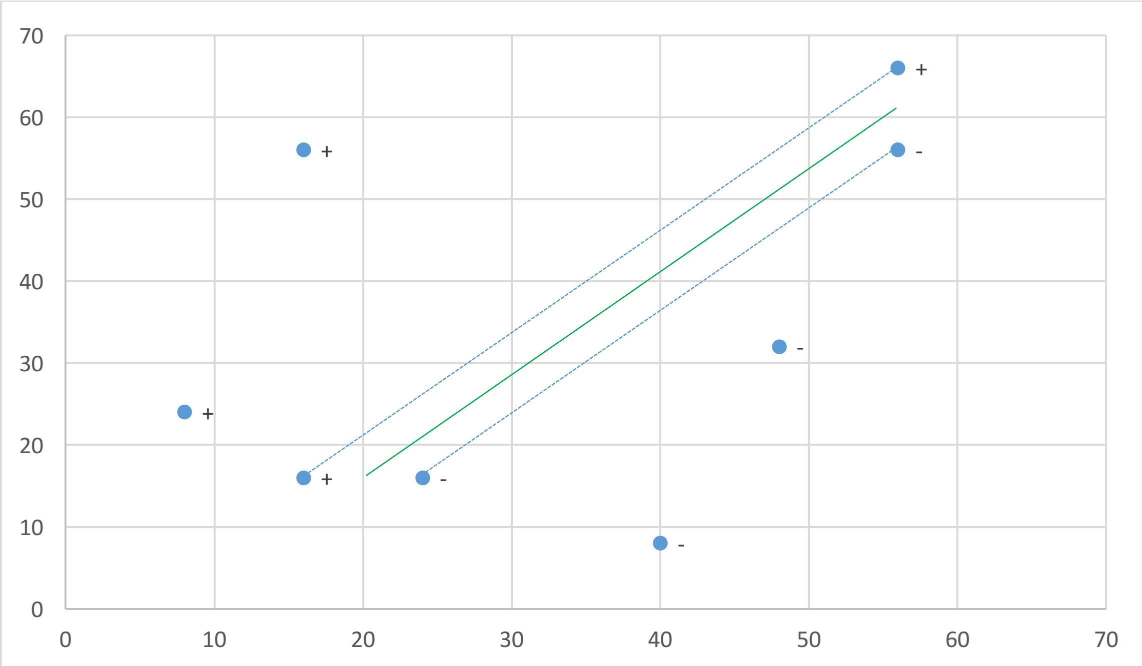
Hence the root attribute can be selected as F1 as it has the maximum gain.



2. [25 pts.] Derive the equation for the maximum margin separating hyperplane that a Support Vector Machine would find to classify the following set of points.

- **Positive:** (56,66), (16,16), (16,56), (8,24)
- **Negative:** (48,32), (40,8), (24,16), (56,56)

⇨ To get the maximum margin separating hyperplane found by Support Vector Machine(SVM), we need to find the examples in both the sets which are nearest to each other. Calculating the distance between each –ve class example with each +ve class example, we get to the conclusion that.



+(16,16) and –(24,16) have a distance between them which equals 8 units.

+(56, 66) and –(56,56) have a distance between them which equals 10 units and these 2 are the 2 examples from each class which are closest to each other.

Consider a line passing through the two positive examples (16,16) and –(24,16). Then the equation of this line would be:

$$y - y_1 = \frac{y_2 - y_1}{x_2 - x_1}(x - x_1)$$

$$y - 66 = \frac{16-66}{16-56}(x - 56)$$

$$4y = 5x - 16$$

Now consider another line passing through the two negative examples (24,16) and (56,56). Then the equation of this line would be:

$$y - y_1 = \frac{y_2 - y_1}{x_2 - x_1}(x - x_1)$$

$$y - 56 = \frac{16-56}{24-56}(x - 56)$$

$$4y = 5x - 56$$

The maximum margin separating hyperplane will be the line equidistant and parallel to the 2 lines.

$$\frac{4y - 5x + 56}{\sqrt{16+25}} = \frac{4y - 5x + 16}{\sqrt{16+25}}$$

$$8y - 10x + 72 = 0$$

$$4y - 5x + 36 = 0$$

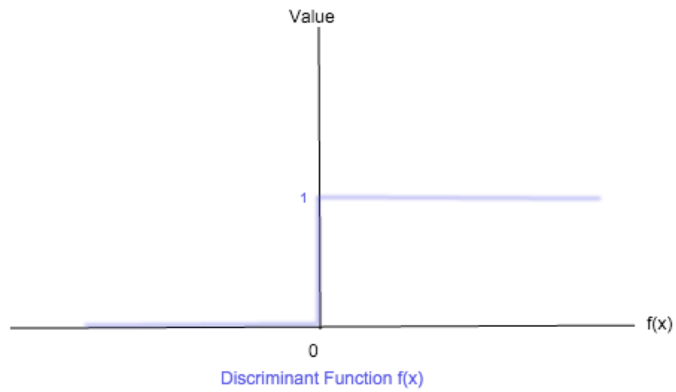
Hence, the equation of the maximum margin separating hyperbole is 4y – 5x + 36 = 0.

3. [25 pts.] Consider a neuron with two inputs, one output, and a threshold activation function. If the two weights are w_1 = 1 and w_2 = 1, and the bias is b = -1.5, then what is the output for input (0,0)? What about for inputs (1,0), (0,1), and (1,1)? Draw the discriminant function for this neuron, and write down its equation. Does it correspond to any particular logic gate?

- We have the following set of input sets:
  (0, 0), (0, 1), (1, 0) and (1, 1).
  We are also given a couple of weights:
  W1 = 1, W2 = 1.
  and a bias:
  B = -1.5

I1	I2	Value($\sum IW$+ B)	O/P(Threshold = 0)
0	0	-1.5	0
1	0	-0.5	0
0	1	-0.5	0
1	1	0.5	1

Discriminant Function f(x) = $\begin{cases} 0 \text{ } if \text{ } value \leq 0 \\ 1 \text{ } if \text{ } value > 0 \end{cases}$



4. [25 pts.] Suppose you are running a learning experiment on a new algorithm for Boolean classification. You have a data set consisting of 100 positive examples and 100 negative examples. You plan to use leave-one-out cross-validation and compare your algorithm to a baseline function, a simple majority classifier that outputs the class that is in the majority in the training set, regardless of its input. You expect the majority classifier to score about 50% on leave-one-out, but to your surprise, it scores zero every time. Explain why.

⇨ Each time we run the algorithm, an example is selected from the 100 positive examples and 100 negative examples, randomly. This is used for the test set and the rest of the examples are used for the training set. So assuming, the algorithm chose a positive example from the data set for the test set.
The test set will have only one example which would be positive and the majority classifier will give an output as positive. But the same majority classifier will give a negative output in the training set because there are 100 negative examples and 99 positive examples. Hence the output for the test set will always be different from the output in training set and hence majority classifier will score 0 always.

BONUS QUESTIONS:

1. (40 pts.) **The Training Set:** Construct a training set having the appropriate format for the classifier you are using, which, when used with the classifier, comes as close as you can to outputting the following decision tree (the format is roughly that of the Prolog ID3; Weka J48 will be different):

⇨ The following is the training set:

```
attribute(income, [low, medium, high]).
attribute(history, [good, bad, unknown]).
attribute(debt, [low, high]).
attribute(collateral, [none, adequate]).

example(high, [income = low, history = good, debt = low, collateral =
none]).
example(high, [income = low, history = good, debt = low, collateral =
adequate]).
example(high, [income = low, history = good, debt = high, collateral =
none]).
example(high, [income = low, history = good, debt = high, collateral =
adequate]).

example(high, [income = low, history = bad, debt = low, collateral =
none]).
example(high, [income = low, history = bad, debt = low, collateral =
adequate]).
example(high, [income = low, history = bad, debt = high, collateral =
none]).
example(high, [income = low, history = bad, debt = high, collateral =
adequate]).

example(high, [income = low, history = unknown, debt = low, collateral
= none]).
example(high, [income = low, history = unknown, debt = low, collateral
= adequate]).
example(high, [income = low, history = unknown, debt = high, collateral
= none]).
example(high, [income = low, history = unknown, debt = high, collateral
= adequate]).


example(high, [income = medium, history = bad, debt = low, collateral =
none]).
example(high, [income = medium, history = bad, debt = low, collateral =
adequate]).
example(high, [income = medium, history = bad, debt = high, collateral
= none]).
example(high, [income = medium, history = bad, debt = high, collateral
= adequate]).
```

```
example(high, [income = medium, history = unknown, debt = high,
collateral = none]).
example(high, [income = medium, history = unknown, debt = high,
collateral = adequate]).


example(moderate, [income = medium, history = good, debt = low,
collateral = none]).
example(moderate, [income = medium, history = good, debt = low,
collateral = adequate]).
example(moderate, [income = medium, history = good, debt = high,
collateral = none]).
example(moderate, [income = medium, history = good, debt = high,
collateral = adequate]).



example(moderate, [income = medium, history = unknown, debt = low,
collateral = none]).
example(moderate, [income = medium, history = unknown, debt = low,
collateral = adequate]).


example(moderate, [income = high, history = bad, debt = low, collateral
= none]).
example(moderate, [income = high, history = bad, debt = low, collateral
= adequate]).
example(moderate, [income = high, history = bad, debt = high,
collateral = none]).
example(moderate, [income = high, history = bad, debt = high,
collateral = adequate]).


example(low, [income = high, history = good, debt = low, collateral =
none]).
example(low, [income = high, history = good, debt = low, collateral =
adequate]).
example(low, [income = high, history = good, debt = high, collateral =
none]).
example(low, [income = high, history = good, debt = high, collateral =
adequate]).



example(low, [income = high, history = unknown, debt = low, collateral
= none]).
example(low, [income = high, history = unknown, debt = low, collateral
= adequate]).
example(low, [income = high, history = unknown, debt = high, collateral
= none]).
example(low, [income = high, history = unknown, debt = high, collateral
= adequate]).
```

The above training set is consulted along with **ID3b.pl** file to get the following decision tree:

```
income = low : high
income = medium
    history = good : moderate
    history = bad : high
    history = unknown
        debt = high : high
        debt = low : moderate
income = high
    history = good : low
    history = bad : moderate
    history = unknown : low
```

2. **The Report:** In addition to the training set, write and submit a short report describing

    1. your approach to solving the problem,
    2. what happened, and
    3. what conclusions you draw from your results.

⇨ The problem was solved with the basis on impurity, variance and Gain. The attribrute with the highest gain and lowest impurity is selected as the root of the tree. So that it divides the examples according to the classes.

In this question, the approach was to create a tranining set which considered first the attribrute with the highest gain and lowest impurity, which was 'income'. And it started the decision tree based on income. Next we found that 'history' had the next best gain and the 2^{nd} to worst impurity, hence it was considered as the node for the next level and classification was done based on it. Next debt was considered because it had the 3^{rd} best gain and the 3^{rd} to worst impurity. In the end collateral was selected because it had no impact on the classification of the examples.

From the above question we can conclude that if the gain of an attribrute is high it can efficiently classify the examples, while on the other hand if the gain of an attribute is low, it wont be efficient in classifying the problems and hence the the attribrute with lower gain is placed at a lower level in the decision tree.

Similarly, impurity decides how purely(or impurely) classified are the solutions. If the impurity is high then we can say that the given attribrute couldn't classify the data properly and hence the attribrute should be kept low in the decision table.