Machine Learning I ANSWERS

For the questions below, consider the data points given on the last page of the handout.

1 Conjunction Learning

1. Will our conjunction learning algorithm, if it's restricted to using only positive literals, be able to learn a hypothesis that has zero error for distinguishing between the red points and the points that are not red? If so, what is the hypothesis? If not, why not?

   yes. $f_2 \wedge f_3$

2. Now consider the green points. What will happen if we run the conjunction learning algorithm, still restricted to just positive literals, to learn a distinction between the green and non-green points? The pseudo-code for the conjunction learning algorithm is below.

```
N = negative examples in D
h = True
Loop until N is empty
  For every feature j that does not have value 0 on any positive examples
    nj := number of examples in N for which fj = 0
    j* := j for which nj is maximized
    h := h ^ fj*
    N := N - examples in N for which fj* = 0
  If no such feature found, fail
```

   There is no feature that doesn't have value 0 on any positive examples, so we will fail right away.

3. If we allow the algorithm to consider negative literals as well, will it find a hypothesis with zero error for the green vs. non-green points?

   Yes. The algorithm gives $\neg f_2 \wedge \neg f_3$.

4. What about the blue vs. non-blue points? Can the conjunction learning algorithm with positive and negative literals find a zero-error hypothesis for these points?

   No. We need disjunction for this one.

2 Disjunction Learning

1. First, let's simulate running the disjunction learning algorithm with only positive literals on the blue vs. non-blue points in our data set. The pseudo-code for the algorithm is given below. Remember that we select which feature to add to r by following the heuristic from lecture. What happens?

```
P = set of all positive examples
h = False
Loop until P is empty
  r = True
  N = set of all negative examples
  Loop until N is empty
    If all features are in r, fail
    Else, select a feature fj to add to r
      r = r ^ fj
      N = N - examples in N for which fj=0
  h = h v r;
  covered = examples in P covered by r
  if covered is empty, fail
  else P = P - covered
```

We fail because each of the literals gets added in the first inner loop (note that we never leave the inner loop because N is never empty). This means that the blue points are not representable with a disjunctive normal formula with only positive literals. Here's the simulation:

- $h = False$**,** $P = \{x^1, x^2, x^3, x^4\}$
  - $r = True$, $N = \{x^5, x^6, x^7, x^8, x^9\}$
    - $v_1 = \frac{3}{3}, \mathbf{v_2} = \mathbf{\frac{3}{2}}, v_3 = \frac{1}{2}, v_4 = \frac{3}{4}$
  - $r = f_2$, $N = \{x^5, x^6\}$
    - $\mathbf{v_1} = \mathbf{\frac{2}{1}}, v_3 = 0, v_4 = \frac{2}{2}$
  - $r = f_1 \wedge f_2$, $N = \{x^6\}$
    - $v_3 = 0, \mathbf{v_4 = 1/1}$
  - $r = f_1 \wedge f_2 \wedge f_4$, $N = \{x^6\}$
    - $\mathbf{v_3 = 0}$
  - $r = f_1 \wedge f_2 \wedge f_4 \wedge f_3$
  - **All features are in r, so fail**

2. Now let's consider allowing negative literals as well. How must we change the algorithm to cope with negative features?

   When considering which feature to add to the conjunction in the algorithm's inner loop, we need to consider the negation of each feature as well

3. Let's simulate the DNF algorithm *with* negative features on the blue points from our data set.

- $h = False$, $P = \{x^1, x^2, x^3, x^4\}$
  - $r = True$, $N = \{x^5, x^6, x^7, x^8, x^9\}$

        * $v_1 = \frac{3}{3}, \mathbf{v_2} = \mathbf{\frac{3}{2}}, v_3 = \frac{1}{2}, v_4 = \frac{3}{4}$
        * $v_{\neg 1} = \frac{1}{2}, v_{\neg 2} = \frac{1}{3}, v_{\neg 3} = \frac{3}{3}, v_{\neg 4} = \frac{1}{1}$
    - $r = f_2,\ N = \{x^5, x^6\}$
        * $v_1 = \frac{2}{1}, v_3 = 0, v_4 = \frac{2}{2}$
        * $v_{\neg 1} = \frac{1}{1}, \mathbf{v_{\neg 3}} = \mathbf{\frac{3}{0.001}}, v_{\neg 4} = \frac{1}{0.001}$
    - $r = f_2 \wedge \neg f_3,\ N = \{\}$
- $h = (f_2 \wedge \neg f_3),\ P = \{x^1\}$
    - $r = True,\ N = \{x^5, x^6, x^7, x^8, x^9\}$
        * $v_1 = \frac{1}{3}, v_2 = 0, \mathbf{v_3} = \mathbf{\frac{1}{2}}, v_4 = \frac{1}{4}$
        * $v_{\neg 1} = 0, v_{\neg 2} = \frac{1}{3}, v_{\neg 3} = 0, v_{\neg 4} = 0$
    - $r = f_3,\ N = \{x^5, x^6\}$
        * $v_1 = \frac{1}{1}, v_2 = 0, v_4 = \frac{1}{2}$
        * $v_{\neg 1} = 0, \mathbf{v_{\neg 2}} = \mathbf{\frac{1}{0.001}}, v_{\neg 4} = 0$
    - $r = f_3 \wedge \neg f_2,\ N = \{\}$
- $h = (f_2 \wedge \neg f_3) \vee (f_3 \wedge \neg f_2),\ P = \{\}$

4. Running the DNF algorithm (with both positive and negative literals) on the red and green points gives the same answers that we found doing conjunction learning from above. That is, the function for determining whether a point is red or not is $f_2 \wedge f_3$ and the function for determining whether or not a point is green is $\neg f_2 \wedge \neg f_3$. The following table shows all of the 4-bit inputs that were not in our training set. Figure out which category each point is in.

f_1	f_2	f_3	f_4	**class**
0	**0**	**0**	**0**	**green**
0	**0**	**0**	**1**	**green**
0	**0**	**1**	**0**	**blue**
0	**0**	**1**	**1**	**blue**
0	**1**	**0**	**0**	**blue**
0	**1**	**1**	**0**	**red**
1	**0**	**1**	**0**	**blue**

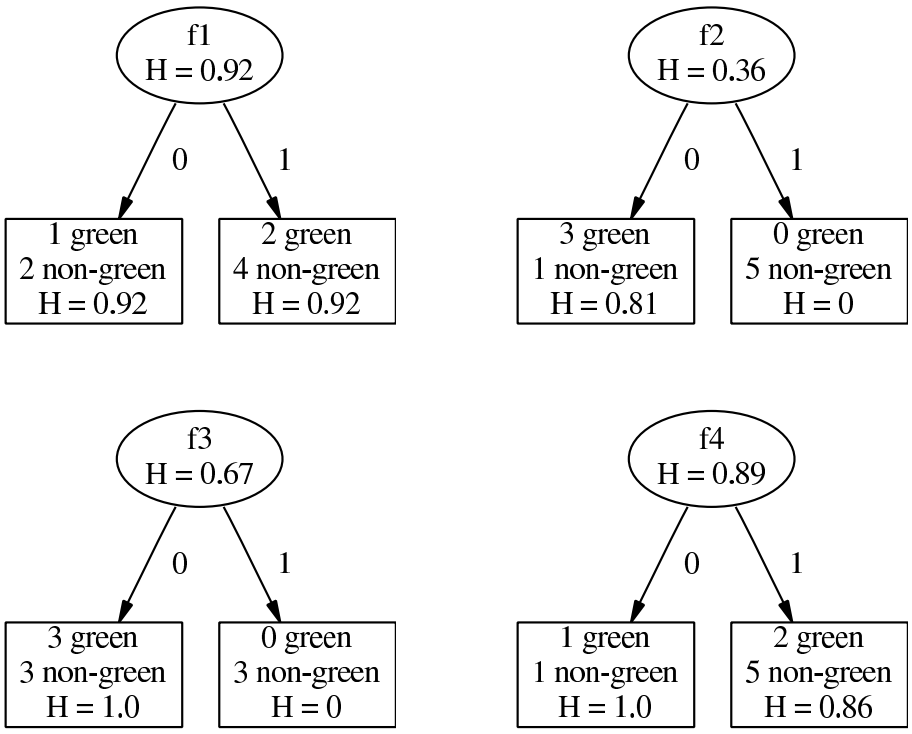
5. If we learn that the data point 0110 is actually green, this will cause our hypothesis for the red and green classes to shift. The hypothesis for the blue points will not change (depending on how ties are broken). Running the DNF learning algorithm on the new data set will give the function $f_2 \wedge f_3 \wedge f_4$ for the red points, and the function $(\neg f_2 \wedge \neg f_3) \vee (\neg f_1 \wedge \neg f_4)$ for the green points. How would we now classify the point 0010?

   Now we don't know how to classify the point. The XOR function for the blue points says that the point is blue, but our new function for green says it can be a green point as well. If we were using an algorithm that actually expressed a weight or a certainty associated with its prediction, we could choose based on which hypothesis is more certain, but since we just get a binary answer, we don't necessarily have the ability to classify all points based on the answers from our three separate hypotheses.

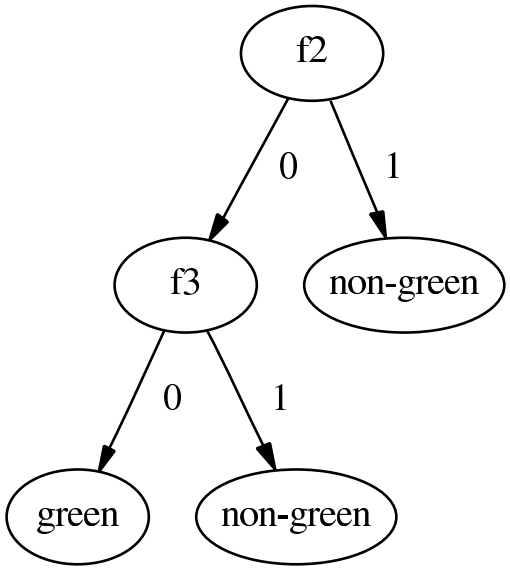
3 Decision Trees

Now let's consider how to learn the same data set using a decision tree. For this algorithm we will only consider one of the classes; that is, we will look at the green/non-green distinction.

1. Let's consider which split to make first, In each of the trees below, fill in the proportion of green and non-green points that go in the leaves. Without computing the entropy, can we make some guesses about which split might be the best one?



2. After choosing the first split, the algorithm will make one more split and give us the following tree as its hypothesis. What is the corresponding Boolean hypothesis for the green points?



3. Is the decision tree hypothesis for the green points the same as the hypothesis we got before with the conjunction learning algorithm? Will this always be the case?

   $\neg f_2 \wedge \neg f_3$**. Yes, they are the same, but this is not guaranteed.**

4 Naive Bayes

Lastly, let's consider the Naive Bayes algorithm (using the Laplace correction) on this data set.

1. Would we classify the point 0110 as red or non-red?

$$S(0110, red) = R_1(0, red) * R_2(1, red) * R_3(1, red) * R_4(0, red)$$

$$S(0110, red) = \frac{1+1}{2+2} * \frac{2+1}{2+2} * \frac{2+1}{2+2} * \frac{0+1}{2+2}$$

$$S(0110, red) = \frac{1}{2} * \frac{3}{4} * \frac{3}{4} * \frac{1}{4}$$

$$S(0110, red) = 0.0703$$

$$S(0110, nonred) = R_1(0, nonred) * R_2(1, nonred) * R_3(1, nonred) * R_4(0, nonred)$$

$$S(0110, nonred) = \frac{2+1}{7+2} * \frac{3+1}{7+2} * \frac{1+1}{7+2} * \frac{2+1}{7+2}$$

$$S(0110, nonred) = \frac{1}{3} * \frac{4}{9} * \frac{2}{9} * \frac{1}{3}$$

$$S(0110, nonred) = 0.0110$$

So we predict the point is red.

2. Would we classify that same point (0110) is green, or non-green?

$$S(0110, green) = \frac{1+1}{3+2} * \frac{0+1}{3+2} * \frac{0+1}{3+2} * \frac{1+1}{3+2}$$

$$S(0110, green) = \frac{2}{5} * \frac{1}{5} * \frac{1}{5} * \frac{2}{5}$$

$$S(0110, green) = 0.0064$$

$$S(0110, nongreen) = \frac{2+1}{6+2} * \frac{5+1}{6+2} * \frac{3+1}{6+2} * \frac{1+1}{6+2}$$

$$S(0110, nongreen) = \frac{3}{8} * \frac{3}{4} * \frac{1}{2} * \frac{1}{4}$$

$$S(0110, nongreen) = 0.035$$

So we predict the point is not green.

3. What about blue, or non-blue?

$$S(0110, blue) = \frac{1+1}{4+2} * \frac{3+1}{4+2} * \frac{1+1}{4+2} * \frac{1+1}{4+2}$$

$$S(0110, blue) = \frac{1}{3} * \frac{2}{3} * \frac{1}{3} * \frac{1}{3}$$

$$S(0110, blue) = 0.0247$$

$$S(0110, nonblue)\frac{2+1}{5+2} * \frac{2+1}{5+2} * \frac{2+1}{5+2} * \frac{1+1}{5+2}$$

$$S(0110, nonblue)\frac{3}{7} * \frac{3}{7} * \frac{3}{7} * \frac{2}{7}$$

$$S(0110, nonblue) = 0.0225$$

So we predict the point is blue. Note that it is possible to have a point which is predicted to be none of the available categories, or to be more than one. Because naive Bayes computes a score, we can deal with these cases by comparing the scores.

Dataset

The features in each point are binary, but there are three classes into which each of the points is classified.

pt	f_1	f_2	f_3	f_4	class
x^1	1	0	1	1	blue
x^2	1	1	0	0	blue
x^3	0	1	0	1	blue
x^4	1	1	0	1	blue
x^5	0	1	1	1	red
x^6	1	1	1	1	red
x^7	0	0	0	1	green
x^8	1	0	0	1	green
x^9	1	0	0	0	green