

A candy manufacturer interviews a customer on his willingness to eat a candy of a particular color or flavor. The following table shows the collected responses:

| Color | Flavor | Edibility |
|-------|--------|-----------|
| Red | Grape | Yes |
| Red | Cherry | Yes |
| Green | Grape | Yes |
| Green | Cherry | No |
| Blue | Grape | No |
| Blue | Cherry | No |

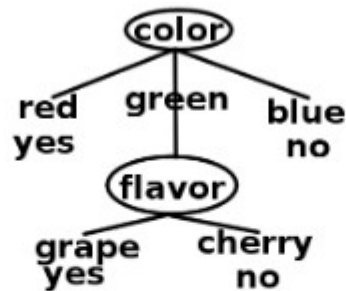
1. (2 points) What is $H(\text{edibility})$? It is not necessary to show your work.

$$-\left(\frac{1}{2}\log_2\frac{1}{2} + \frac{1}{2}\log_2\frac{1}{2}\right) = 1$$

2. (2 points) What is $H(\text{edibility} \mid \text{color})$? It is not necessary to show your work.

$$-\left(\frac{1}{3}(1\log_2 1 + 0\log_2 0) + \frac{1}{3}\left(\frac{1}{2}\log_2\frac{1}{2} + \frac{1}{2}\log_2\frac{1}{2}\right) + \frac{1}{3}(1\log_2 1 + 0\log_2 0)\right) = \frac{1}{3}$$

4. (2 points) Draw the decision tree for predicting edibility that maximizes the information gain.



5. (2 points) Using your decision tree, what would you predict for the edibility of a blue, blueberry-flavored candy?

Edibility: No

We would like to predict the sex of a person based on two binary attributes: leg-cover (pants or skirts) and facial-hair (some or none). We have a data set of 2,000 individuals, half male and half female. 75% of the males have no facial hair. Skirts are worn by 50% of the females. All females are barefaced and no male wears a skirt.

| | |
|---------------|-----------|
| $\log_2(1/8)$ | $= -3.0p$ |
| $\log_2(1/4)$ | $= -2.00$ |
| $\log_2(1/3)$ | $= -1.58$ |
| $\log_2(3/8)$ | $= -1.42$ |
| $\log_2(3/7)$ | $= -1.22$ |
| $\log_2(1/2)$ | $= -1.00$ |
| $\log_2(4/7)$ | $= -0.81$ |
| $\log_2(5/8)$ | $= -0.68$ |
| $\log_2(2/3)$ | $= -0.58$ |
| $\log_2(3/4)$ | $= -0.42$ |
| $\log_2(7/8)$ | $= -0.19$ |

(a) What is the initial entropy in the system? [5]

$$\text{initial entropy} = I(1/2, 1/2) = -(1/2 * \log(1/2) + 1/2 * \log(1/2)) = 1$$

(b) Compute the information gain of initially choosing the attribute leg-cover and for initially choosing facial-hair. [15]

$$\text{Entropy after leg-cover} = 1/4 * I(0, 1) + 3/4 * I(1/3, 2/3) \approx 0 + 0.69 = 0.69$$

$$\text{Information gain for leg cover} = 0.31$$

$$\text{Entropy after facial-hair} = 1/8 * I(0, 1) + 7/8 * I(3/7, 4/7) \approx 0 + 0.86$$

$$\text{Information gain for facial hair} = 0.14$$

(c) Based on your answers, which attribute should be used as the root of a decision tree? [10]

Leg cover

(d) How can the information gain heuristic that was originally designed for symbolic attributes be generalized to cope with continuous attributes? [5]

Quantize the variable into a fixed number of ranges. For example, if T represents the temperature and is assumed to be a real number ranging from -50 to +120, we might quantize it into ranges like (<0, 0-32, 32-90, >90)

(e) A friend notes that in some countries men do wear skirts or skirt-like garments. To ensure a more accurate decision tree, he suggests adding another variable: the person's ID number represented as an integer between 0 and one billion. Is this a good idea? Why or why not, assuming you are using ID3 and the simple information gain heuristic. [10]

Using this variable would produce the largest information gain, since it would split all 2000 observations into 200 categories, each with just one instance and thus a known class. So ID# would choose that variable and then stop. Of course, this would be useless as a decision tree, since any new examples, would most likely not match any of the IDs in the decision tree and would not be classified.

We have some data about when people go hiking. The data take into effect, whether hike is on a weekend or not, if the weather is rainy or sunny, and if the person will have company during the hike. Find the optimum decision tree for hiking habits, using the training data below. When you

You may find the following useful in your calculations: $H(x) = H(1-x)$, $H(0) = 0$, $H(1/5) = 0.72$, $H(1/4) = 0.8$, $H(1/3) = 0.92$, $H(2/5) = 0.97$, $H(3/7) = 0.99$, $H(0.5) = 1$.

| Weekend? | Company? | Weather | Go Hiking? |
|----------|----------|---------|------------|
| Y | N | R | N |
| Y | Y | R | N |
| Y | Y | R | Y |
| Y | Y | S | Y |
| Y | N | S | Y |
| Y | N | S | N |
| Y | Y | R | N |
| Y | Y | S | Y |
| N | Y | S | N |
| N | Y | R | N |
| N | N | S | N |

- (a) [13 points] Draw your decision tree. **solution:** We want to choose attributes that maximize $mH(p) - m_rH(p_r) - m_lH(p_l)$. This means that at each step, we need to choose the attributes for which $m_rH(p_r) + m_lH(p_l)$ is minimum. For the first step, the *Weekend* attribute achieve this:

$$\text{Weekend} : m_rH(p_r) + m_lH(p_l) = 8H(1/2) + 3H(0) = 8$$

$$\text{Weather} : m_rH(p_r) + m_lH(p_l) = 5H(1/5) + 6H(1/2) \approx 9.6$$

$$\text{Company} : m_rH(p_r) + m_lH(p_l) = 4H(1/4) + 7H(3/7) \approx 10.1$$

Therefore we first split on weekend attribute.

If weekend = NO: then Go Hiking = NO.

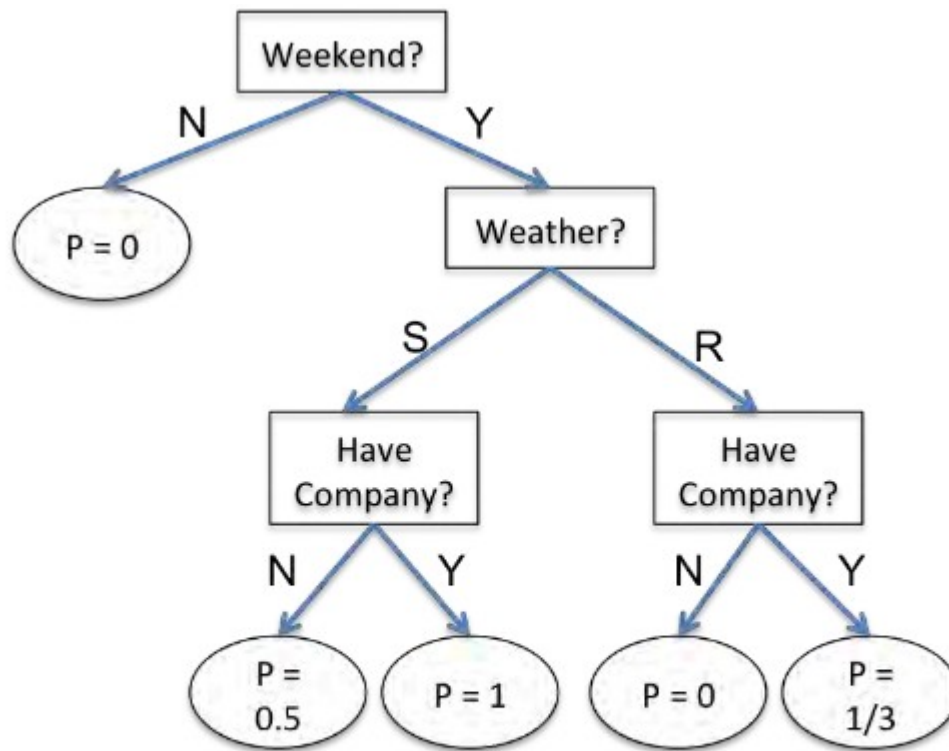
If weekend = YES, we need to choose second attribute to split on:

$$\text{Weather} : m_rH(p_r) + m_lH(p_l) = 4H(1/4) + 4H(1/4) \approx 6.4$$

$$\text{Company} : m_rH(p_r) + m_lH(p_l) = 5H(2/5) + 3H(1/3) \approx 7.6$$

Therefore the second attribute will be *Weather* attribute, and third one will be *Company* attribute. The decision tree will be as follows:

- (b) [1 point] According to your decision tree, what is the probability of going to hike on a rainy week day, without any company? **Answer:** 0



- (c) [1 point] How about probability of going to hike on a rainy weekend when having some company?
Answer: 1/3.

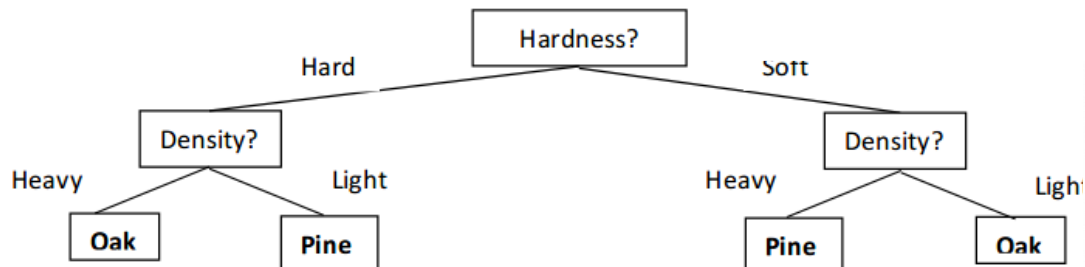
1. (10 pts total) Decision Tree Classifier Learning. You are a robot in a lumber yard, and must learn to discriminate Oak wood from Pine wood. You choose to learn a Decision Tree classifier. You are given the following examples:

| Example | Density | Grain | Hardness | Class |
|------------|---------|-------|----------|-------|
| Example #1 | Heavy | Small | Hard | Oak |
| Example #2 | Heavy | Large | Hard | Oak |
| Example #3 | Heavy | Small | Hard | Oak |
| Example #4 | Light | Large | Soft | Oak |
| Example #5 | Light | Large | Hard | Pine |
| Example #6 | Heavy | Small | Soft | Pine |
| Example #7 | Heavy | Large | Soft | Pine |
| Example #8 | Heavy | Small | Soft | Pine |

1a. (2 pts) Which attribute would information gain choose as the root of the tree?

Hardness

1b. (4 pts) Draw the decision tree that would be constructed by recursively applying information gain to select roots of sub-trees, as in the



Classify these new examples as Oak or Pine using your decision tree above.

1c. (2 pts) What class is [Density=Light, Grain=Small, Hardness=Hard]? Pine

1d. (2 pts) What class is [Density=Light, Grain=Small, Hardness=Soft]? Oak

For neural network classification, it is typical to train k networks and average the results. Why not run your decision tree (using all of the data) k times and then average the results?

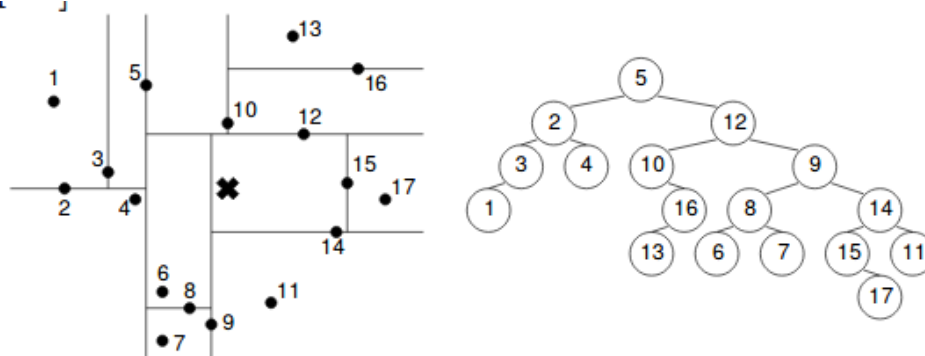
Decision trees return the same result each time you run them.

[2 pts] True or false: Selecting the decision tree split (at each node as you move down the tree) that *minimizes classification error* will guarantee an optimal decision tree.

☐ True ☒ False

[2 pts] True or false: Selecting the decision tree split (at each node as you move down the tree) that *maximizes information gain* will guarantee an optimal decision tree.

☐ True ☒ False



- (1) [5 pts] Above, we have two depictions of the same k -d tree, which we have built to solve nearest neighbor queries. Each node of the tree at right represents a rectangular box at left, and also stores one of the sample points that lie inside that box. (The root node represents the whole plane \mathbb{R}^2 .) If a treenode stores sample point i , then the line passing through point i (in the diagram at left) determines which boxes the child treenodes represent.

Simulate running an exact 1-nearest neighbor query, where the bold X is the query point. Recall that the query algorithm visits the treenodes in a smart order, and keeps track of the nearest point it has seen so far.

- Write down the numbers of all the sample points that serve as the “nearest point seen so far” sometime while the query algorithm is running, in the order they are encountered.
- Circle all the subtrees in the k -d tree at upper right that are never visited during this query. (This is why k -d tree search is usually faster than exhaustive search.)

Nearest point seen so far: first 5, then 12, then 10.

The unvisited subtrees are rooted at 2, 13, 7, and 17.

- (2) [5 pts] We are building a decision tree for a 2-class classification problem. We have n training points, each having d real-valued features. At each node of the tree, we try every possible univariate split (i.e. for each feature, we try every possible splitting value for that feature) and choose the split that maximizes the information gain.

Explain why it is possible to build the tree in $O(ndh)$ time, where h is the depth of the tree’s deepest node. Your explanation should include an analysis of the time to choose one node’s split. Assume that we can radix sort real numbers in linear time.

Consider choosing the split at a node whose box contains n' sample points. For each of the d features, we can sort the sample points in $O(n'd)$ time. Then we can compute the entropy for the first split (separating the first sample in the sorted list from the others) in $O(n')$ time, then we can walk through the list and update the entropy for each successive split in $O(1)$ time, summing to a total of $O(n')$ time for each of the d features. So it takes $O(n'd)$ time overall to choose a split.

Each sample point participates in at most h treenodes, so each sample point contributes at most dh to the running time, for a total running time of at most $O(ndh)$.