Decision Trees and Naive Bayes

- The idea of **learning** is that an agent observes some data and uses it to develop some behavior or characterization, such that when he sees new data, he can generate appropriate outputs. In **supervised** learning, the agent is given inputs along with the desired outputs, whereas in **unsupervised** learning, he is given only inputs. In **classification**, the outputs are discrete, whereas in **regression**, the outputs are continuous. Generally, an input is called $\mathbf{x}_i$ (and can be multidimensional), and the corresponding output is y_i (usually scalar).

- Learning can be characterized as choosing a single **model** (aka hypothesis) from a **hypothesis class**. In choosing the model, we want to minimize the amount of error it makes; for example, the **mean square error** of a hypothesis f is

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - f(\mathbf{x}_i))^2,$$

  where the inputs are $\mathbf{x}_1, ..., \mathbf{x}_n$ and the correct outputs are $y_1, ..., y_n$. Note that the error can only be computed with respect to inputs for which the correct outputs are known.

- We want to avoid **overfitting** the model to the specific examples we've seen: the purpose of learning isn't to perform perfectly on known examples, it's to perform well on future examples. There is a tradeoff between fitting the data and generalizing well; **Occam's razor** is the concept of "don't make things more complicated than they need to be".

- One way to evaluate an algorithm and avoid overfitting is **cross-validation**. In k-fold cross-validation, the training data are divided in to k "folds"; for each fold, the algorithm is trained on the remaining $k-1$ folds and evaluated on the held-out one.

- A **decision tree** is a method for classification: each internal node in the tree is a "question" that gets applied to any data points that get there (for example "how many legs does it have" or "is its x_5 value bigger than 22"), and depending on the "answer," a different branch is followed, until reaching a leaf node, which specifies a label.

- One algorithm to construct a decision tree is to greedily minimize entropy. The **entropy** (in bits) of a set of labeled things is

$$H = \sum_{i=1}^{n} -\frac{N_i}{N} \log_2 \frac{N_i}{N},$$

  where N_i is the number of things with the ith label and $N = \sum_{i=1}^{n} N_i$ is the total number of things. A question divides the data points into sets, so we can choose the question that minimizes the weighted average of the entropies of the sets.

- Unless we have two identical data points with differing labels, it is possible to construct a decision tree that perfectly classifies the training data, but that will probably be overfitting. We can stop building the tree when the overall entropy in each region is small enough, or when number of elements per region is small enough. Or, we can construct a full tree and prune it back until it has good empirical performance.

- **Naive Bayes** is a method for classification that simply outputs the maximum a posteriori label assuming that all the features are independent. Recall that the posterior probability of a label y_j given a data point $\mathbf{x} = \langle x_1, ..., x_n \rangle$ is

$$P(y_j|\mathbf{x}) \propto P(y_j) \cdot P(\mathbf{x}|y_j) = P(y_j) \cdot P(x_1, ..., x_n|y_j),$$

  and using our independence assumption, we can assume that this is equal to

$$P(y_j) \cdot P(x_1|y_j) \cdot ... \cdot P(x_n|y_j),$$

  which is easy to compute from data: $P(x_i|y_j)$ is just the proportion of data points with label y_j that have the specified value of x_i. For the ith feature, we can define the quantity $R_i(a, b) = P(x_i = a|y = b)$.

- If there are lots of features and not many data points, many of the observed $P(x_i|y_j)$ might be 0, and if even one of them is 0, it will cause the estimated $P(y_j|\mathbf{x})$ to be 0, which can be undesirable. To avoid this, we can use the **Laplace correction**: instead of using the actual counts of how many data points with label y_j have the specified x_i, we add 1 to each count.

Exercises

1. (AIMA 18.6) Consider the following data set comprised of three binary input attributes (A_1, A_2, and A_3) and one binary output:

Example	A_1	A_2	A_3	Output y
$\mathbf{x}_1$	1	0	0	0
$\mathbf{x}_2$	1	0	1	0
$\mathbf{x}_3$	0	1	0	0
$\mathbf{x}_4$	1	1	1	1
$\mathbf{x}_5$	1	1	0	1

   Learn a decision tree for these data. Show the computations made to determine the attribute to split at each node.

2. Calculate the R_1, R_2, and R_3 values for a naive Bayes classifier using the above data set. Classify the following data points: $\mathbf{x}_6 = \{0, 0, 0\}$ and $\mathbf{x}_7 = \{0, 1, 1\}$. Repeat this exercise using Laplace smoothing. Do your predictions differ from those obtained by the decision tree from the previous exercise?