CS 4850/6850: Introduction to Machine Learning **Fall 2018**

Topic 5: Random Forests

Instructor: Daniel L. Pimentel-Alarcón

5.1 Introduction

Random forests are one of the most widely used machine learning algorithms for prediction and classification. They aim to address questions such as:

- Will I get *El Cáncer*?
- Will I develop diabetes?
- Is my boyfriend/girlfriend cheating on me?
- Will this bacteria develop antibiotic resistance?

To this end they use a *bootstrap*, or *bagging* version of *decision trees*.

5.2 Setup

Suppose you have a data matrix $\mathbf{X} \in \mathbb{R}^{\mathrm{d}\times\mathrm{n}}$ containing d features about a collection of n bacteria, such as size, shape, and whether specific genes are active or inactive. Also suppose you have a data matrix $\mathbf{Y} \in \mathbb{R}^{1\times\mathrm{n}}$ containing a variable of interest about the n bacteria, for example, whether they are resistant to penicillin. Given a new bacteria with feature vector $\mathbf{x} \in \mathbb{R}^{\mathrm{d}}$, our goal is to determine its variable of interest $\mathrm{y} \in \{0, 1\}$ indicating whether it is resistant to penicillin. To this end, decision trees can use a variety of tools. In this lecture we will use a measure of information called *entropy*.

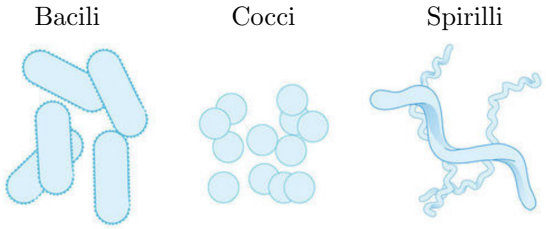


Figure 5.1: Different shapes of bacteria.

5.3 Entropy

As the name suggests, information theory deals with quantifying information in data, efficient ways to store it, and reliable ways to communicate it. One of its most important concepts is that of entropy, which measures the amount of information in a random variable. For a discrete random variable x with support on a set $\mathcal{X}$, entropy is defined as follows:

$$H(x) \;:=\; \mathbb{E}\left[\log_2\left(\frac{1}{\mathbb{P}(x)}\right)\right] \;=\; \sum_{\mathrm{x}\in\mathcal{X}} \mathbb{P}(x=\mathrm{x})\log_2\left(\frac{1}{\mathbb{P}(x=\mathrm{x})}\right).$$

Intuitively, $\log_2\left(\frac{1}{\mathbb{P}(x=\mathrm{x})}\right)$ quantifies the number of bits that one should spend encoding outcome x: the higher the probability of x, the more frequent we will see it, and so the fewer bits we should spend on it; the lower the probability of x the less likely it will appear, and so we can spend more bits on it.

Example 5.1. Consider horse races of 8 horses with the following odds:

Horse	1	2	3	4	5	6	7	8
$\mathbb{P}$(winning)	1/2	1/4	1/8	1/16	1/64	1/64	1/64	1/64

Suppose you want to keep track of which horse wins each race. You can use the following code, which has an average length of 3 bits per race:

Horse	1	2	3	4	5	6	7	8
Code	000	001	010	011	100	101	110	111

However, as entropy suggests, if you assign each horse (outcome x) a code of length $\log_2\left(\frac{1}{\mathbb{P}(x=\mathrm{x})}\right)$, such as the following, then you can achieve an average length of 2 bits per race:

Horse	1	2	3	4	5	6	7	8
Code	0	10	110	1110	111100	111101	111110	111111

To summarize, entropy quantifies the amount of information in a variable. Decision trees essentially *split* data iteratively, according to the most informative variables, and predict according to this split.

5.4 Decision Trees

Suppose we have the following data:

Bacteria	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Gene 1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1
Gene 2	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
Gene 3	0	1	1	0	0	0	1	1	1	1	0	0	0	1	0	1
Resistance?	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**

Table 5.1: Data indicating active genes of 16 bacteria, and whether they are antibiotic resistant.

We will use x_1, x_2, x_3 to refer to our features, in this case whether Genes 1, 2 and 3 are active or inactive. Similarly, we will use y to refer to our variable of interest, in this case whether a bacteria is antibiotic resistant or not.

Step 1: Compute Entropy

To build a decision tree we first compute the entropy for each variable. For example, for Gene 1:

$$\begin{aligned} H(x_1) &= \sum_{\mathrm{x} \in \boldsymbol{\mathcal{X}}} \mathbb{P}(x_1 = \mathrm{x}) \log_2 \left(\frac{1}{\mathbb{P}(x_1 = \mathrm{x})} \right) \\ &= \mathbb{P}(x_1 = 0) \log_2 \left(\frac{1}{\mathbb{P}(x_1 = 0)} \right) + \mathbb{P}(x_1 = 1) \log_2 \left(\frac{1}{\mathbb{P}(x_1 = 1)} \right) \\ &= \frac{1}{16} \log_2(16) + \frac{15}{16} \log_2(\frac{16}{15}) \\ &= \frac{1}{16}(4) + \frac{15}{16}(0.0931) \\ &= 0.3373. \end{aligned}$$

Similarly, for Gene 2 we have:

$$\begin{aligned} H(x_2) &= \sum_{\mathrm{x} \in \boldsymbol{\mathcal{X}}} \mathbb{P}(x_2 = \mathrm{x}) \log_2 \left(\frac{1}{\mathbb{P}(x_2 = \mathrm{x})} \right) \\ &= \mathbb{P}(x_2 = 0) \log_2 \left(\frac{1}{\mathbb{P}(x_2 = 0)} \right) + \mathbb{P}(x_2 = 1) \log_2 \left(\frac{1}{\mathbb{P}(x_2 = 1)} \right) \\ &= \frac{15}{16} \log_2(\frac{16}{15}) + \frac{1}{16} \log_2(16) \\ &= \frac{15}{16}(0.0931) + \frac{1}{16}(4) \\ &= 0.3373, \end{aligned}$$

and for Gene 3:

$$\begin{aligned} H(x_3) &= \sum_{\mathrm{x}\in\mathbf{x}} \mathbb{P}(x_3 = \mathrm{x}) \log_2\left(\frac{1}{\mathbb{P}(x_3 = \mathrm{x})}\right) \\ &= \mathbb{P}(x_3 = 0) \log_2\left(\frac{1}{\mathbb{P}(x_3 = 0)}\right) + \mathbb{P}(x_3 = 1) \log_2\left(\frac{1}{\mathbb{P}(x_3 = 1)}\right) \\ &= \frac{1}{2}\log_2(2) + \frac{1}{2}\log_2(2) \\ &= 1. \end{aligned}$$

Step 2: Split Data

We then split our data according to the most informative variable, and make such variable a node in our decision tree. For instance, in our example we would split data according to Gene 3:

	Bacteria	1	4	5	6	11	12	13	15	2	3	7	8	9	10	14	16
	Gene 1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1
	Gene 2	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
→	Gene 3	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
	Resistance?	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**

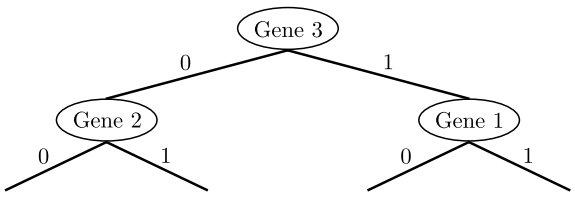
and our tree would start to look as follows:

Step 3: Compute Entropy in Each Subset

We then compute the Entropy of the remaining variables, restricted to each subset of the data. In our example:

$$H(x_1|x3 = 0) = 0, \qquad H(x_1|x3 = 1) = 0.5435,$$
$$H(x_2|x3 = 0) = 0.5435, \qquad H(x_2|x3 = 1) = 0.$$

We thus conclude that given $x_3 = 0$ the most informative variable is x_1, and given $x_3 = 1$ the most informative variable is x_2. We thus add these variables as nodes in our decision tree:



Step 4: Iterate Steps 2 and 3

A decision tree then iterates Step 2 (split data) and 3 (compute entropy in each subset of data) until the entropy of the variable of interest (in each subset of data) is zero (or close to zero).

For instance, in the 2nd iteration of our example, we would split data as follows:

	Bacteria	1	4	5	11	12	13	15	6		14	2	3	7	8	9	10	16	
	Gene 1	1	1	1	1	1	1	1	1		0	1	1	1	1	1	1	1	←
→	Gene 2	0	0	0	0	0	0	0	1		0	0	0	0	0	0	0	0	
	Gene 3	0	0	0	0	0	0	0	0		1	1	1	1	1	1	1	1	
	Resistance?	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**		**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	

Computing the entropy of our variable of interest in each subset, we obtain:

$$H(y|x3=0,x2=0)=0, \qquad H(y|x3=0,x2=1)=0,$$
$$H(y|x3=1,x1=0)=0, \qquad H(y|x3=1,x1=1)=0.$$

At this point we assign the corresponding variable of interest to each leaf. In our example we obtain the following decision tree:

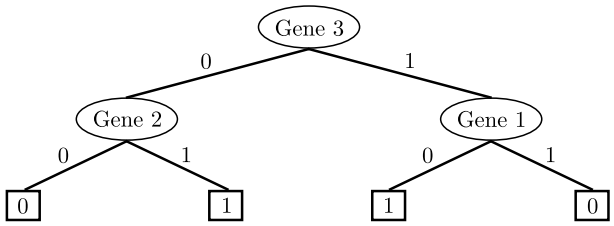


Figure 5.2: Decision tree for the data in Table 5.3.

5.5 Using a Decision Tree

Once we have a decision tree, we can follow it from the root to predict/classify new data. For instance, given the following data:

Bacteria	17	18	19	20
Gene 1	0	1	1	0
Gene 2	1	0	1	0
Gene 3	0	1	0	1

Table 5.2: Data of 4 additional bacteria; it is unknown whether they are antibiotic resistant.

we can use the tree in Figure 5.2 to classify/predict whether these additional bacteria are antibiotic resistant:

Bacteria	17	18	19	20
Gene 1	0	1	1	0
Gene 2	1	0	1	0
Gene 3	0	1	0	1
Resistance?	**1**	**0**	**1**	**1**

Table 5.3: Prediction of whether the bacteria in Table 5.2 are are antibiotic resistant, using the decision tree in Figure 5.2.

5.6 From Decision Trees to Random Forests

As you can imagine, decision trees can be very sensitive to the specific data we observe. In fact, they can be quite biased, and tend to overfit. To overcome these drawbacks, people extend the idea of decision trees to random forests.

As the name suggests, the main idea of random forests is to obtain many decision trees, each time starting with a random subset of the data, either in terms of samples, or features. For example, suppose that instead of Table 5.3 we have the following data:

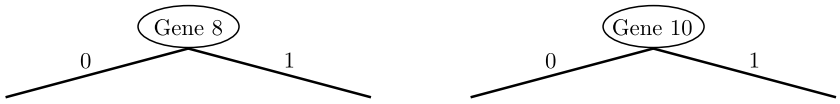
Bacteria	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Gene 1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1
Gene 2	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
Gene 3	0	1	1	0	0	0	1	1	1	1	0	0	0	1	0	1
Gene 4	1	1	0	0	1	1	1	0	1	1	0	0	1	0	0	0
Gene 5	0	0	0	0	0	0	0	1	0	0	0	1	0	0	1	0
Gene 6	0	0	1	0	0	1	1	1	0	0	1	0	1	1	0	0
Gene 7	0	0	0	1	0	0	0	1	1	0	0	0	1	0	1	0
Gene 8	0	1	0	0	0	0	1	1	1	0	1	1	0	0	0	1
Gene 9	0	0	1	0	0	1	0	1	0	0	1	0	1	1	1	1
Gene 10	0	1	0	1	1	0	0	1	1	0	0	1	0	1	0	1
Resistance?	**1**	**0**	**1**	**0**	**0**	**1**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**1**	**0**	**0**

To obtain a random forest, we select a random subset $\mathcal{S}_1 \subset [\mathrm{n}] := \{1, \ldots, \mathrm{n}\}$ of the samples in this table. For example, letting $\mathcal{S}_1 = [\mathrm{n}] \backslash \{3, 10, 14, 16\}$ we obtain:

Bacteria	1	2	4	5	6	7	8	9	11	12	13	15
Gene 1	1	1	1	1	1	1	1	1	1	1	1	1
Gene 2	0	0	0	0	1	0	0	0	0	0	0	0
Gene 3	0	1	0	0	0	1	1	1	0	0	0	0
Gene 4	1	1	0	1	1	1	0	1	0	0	1	0
Gene 5	0	0	0	0	0	0	1	0	0	1	0	1
Gene 6	0	0	0	0	1	1	1	0	1	0	1	0
Gene 7	0	0	1	0	0	0	1	1	0	0	1	1
Gene 8	0	1	0	0	0	1	1	1	1	1	0	0
Gene 9	0	0	0	0	1	0	1	0	1	0	1	1
Gene 10	0	1	1	1	0	0	1	1	0	1	0	0
Resistance?	**1**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

Using this subset of the data we build a decision tree (as described before), which we will denote as $\mathfrak{T}_1$.

We repeat this process T times, each with a different random subset of data $\mathcal{S}_t$ to obtain a sequence of decision trees $\mathcal{T}_1, \ldots, \mathcal{T}_T$, which together form a random forest. Notice that these decision trees are not necessarily going to be the same, because they are run on different subsets of the data. For example, with $\mathcal{S}_1$ above, the most informative variables will be Genes 8 and 10. We can choose amongst them randomly, and so $\mathcal{T}_1$ will start looking like one of the following:



In contrast, if $\mathcal{S}_2 = [n] \backslash \{3, 6, 10, 14\}$, then our data looks as follows:

Bacteria	1	2	4	5	7	8	9	11	12	13	15	16
Gene 1	1	1	1	1	1	1	1	1	1	1	1	1
Gene 2	0	0	0	0	0	0	0	0	0	0	0	0
Gene 3	0	1	0	0	1	1	1	0	0	0	0	1
Gene 4	1	1	0	1	1	0	1	0	0	1	0	0
Gene 5	0	0	0	0	0	1	0	0	1	0	1	0
Gene 6	0	0	0	0	1	1	0	1	0	1	0	0
Gene 7	0	0	1	0	0	1	1	0	0	1	1	0
Gene 8	0	1	0	0	1	1	1	1	1	0	0	1
Gene 9	0	0	0	0	0	1	0	1	0	1	1	1
Gene 10	0	1	1	1	0	1	1	0	1	0	0	1
Resistance?	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

In this case the most informative variable will be Gene 4, and so $\mathcal{T}_2$ will start looking as follows:

Consequently, $\mathcal{T}_1$ will not be the same as $\mathcal{T}_2$. In fact, each tree may produce a different prediction/classification. In a random forest, the final prediction/classification is obtained by consensus over all the predictions/classifications given by all its decision trees. For example, if our random forest has 5 decision trees $\mathcal{T}_1, \ldots, \mathcal{T}_5$, classifying drug resistance of bacteria 17 as $\{11101\}$, then we would conclude that bacteria 17 is drug resistant.