Chapter 13

Random Forests

In Chapter 7, we saw how individual decision trees are learned from training data. These days, decision trees are mainly used as parts of **ensembles**, collections of models whose predictions are combined to produce a final answer.

A **random forest** is an ensemble of decision trees whose predictions are [mostly] uncorrelated. Each tree is built using a subset of the training data and a subset of the features. The trees' predictions are then combined using a voting or weighting scheme. The same basic methodology works for several different supervised learning problems, including classification (Chapter 2), regression (Chapter 3), and survival analysis (Chapter 11). A key advantage of random forests is that they are non-parametric, meaning that they make no distributional or functional assumptions about the relationships between the predictors and the outcome. Another advantage is that the fitting of individual trees happens independently and can be **parallelized**.

Question 13.1

If random forests are so great, why are linear, logistic, and Cox proportional hazards regression models still the standard approaches to predicting continuous, categorical, and survival outcomes in clinical research? What do you think are some of the main drawbacks of random forests and other ensemble methods?

13.1 Building a Random Forest

Assume we have a training dataset of N samples and P predictors. The basic strategy for building a random forest is as follows[1]:

1. For $b = 1, \ldots, B$, where B is the desired number of trees:
    (a) Draw a bootstrap sample of size n from the training data.
    (b) Grow a tree, T_b, on the bootstrap sample by recursively repeating the following steps for each terminal node of the tree until the minimum node size, n_{min}, is reached:
        i. Select p predictors at random.
        ii. Pick the best predictor and split point.
        iii. Split the node into two child nodes.
2. Output the ensemble of trees, $T_1, \ldots, T_B$.

Question 13.2

This algorithm leaves us with many choices. We must choose B, the number of trees; n, the size of each bootstrap sample; p, the number of predictors evaluated for each split; and n_{min}, the minimum node size. What impact does each parameter choice have on the properties of the forest, both in terms of the individual trees and the forest as a whole?

Question 13.3

What does the "best" predictor and split point refer to? How are these choices made for classification and regression models? Think back to our discussion in Chapter 7.

The term **bootstrapping** refers to random sampling with replacement. To create a bootstrap sample of size n from a larger dataset of size N, we simply select n items from among the N, one at a time, being careful to put each

[1] See *Elements of Statistical Learning*, Chapter 15, Algorithm 15.1.

item back between selections. Because we are sampling with replacement, a bootstrap sample will likely contain repeats; this is fine and expected.

The process of averaging model predictions built on different bootstrap samples is called bootstrap aggregating, or **bagging**. We will learn more about why bagging works in Chapter 15.

13.2 Classification Example: Breast Cancer Diagnosis

Let's revisit the classification example for which we built a single decision tree in Chapter 7. The Wisconsin Breast Cancer Dataset contains information about 30 different imaging features of fine needle aspirate (FNA) samples from breast masses in 569 study participants. Here are tabular representations of two trees from a 100-tree random forest built on this dataset:

node_id	left_child	right_child	split_variable	split_point	prediction
1	2	3	area_mean	694.10	
2	4	5	symmetry_worst	0.37	
3	6	7	texture_mean	14.09	
4	8	9	concave.points_mean	0.05	
5	10	11	radius_worst	14.84	
6	12	13	compactness_se	0.03	
7	14	15	concave.points_mean	0.05	
8	16	17	area_se	42.19	
9	18	19	smoothness_worst	0.13	
10	0	0		0.00	B
11	0	0		0.00	M
12	0	0		0.00	B
13	0	0		0.00	M
14	0	0		0.00	M
15	0	0		0.00	M
16	0	0		0.00	B
17	0	0		0.00	M
18	0	0		0.00	B
19	0	0		0.00	M

node_id	left_child	right_child	split_variable	split_point	prediction
1	2	3	perimeter_worst	106.10	
2	4	5	radius_se	0.63	
3	6	7	radius_mean	15.04	
4	8	9	compactness_worst	0.76	
5	10	11	smoothness_se	0.01	
6	12	13	smoothness_mean	0.09	
7	14	15	radius_worst	18.23	
8	16	17	concave.points_worst	0.18	
9	0	0		0.00	M
10	18	19	compactness_se	0.01	
11	0	0		0.00	B
12	0	0		0.00	B
13	0	0		0.00	M
14	0	0		0.00	M
15	0	0		0.00	M
16	0	0		0.00	B
17	0	0		0.00	M
18	0	0		0.00	M
19	0	0		0.00	B

Question 13.4

Draw the two classification trees from the Wisconsin Breast Cancer Dataset random forest that are represented by these tables. Note: if a variable's value is less than the split point at a particular node, the training sample goes to the left.

13.3 Regression Example: Insurance Costs

The following dataset comes from the book *Machine Learning with R*, by Brett Lantz. It's unclear whether it is real or simulated, but it provides insurance cost information on 1338 subjects, as well as information about the following predictors:

(a) age (age of primary beneficiary)

(b) sex (sex of primary beneficiary, labeled "female" or "male")

(c) bmi (body mass index of beneficiary)

(d) children (number of children/dependents covered by beneficiary's health insurance)

(e) smoker (smoking status of beneficiary)

(f) region (the beneficiary's residential area in the U.S.: northeast, southeast, southwest, northwest)

The variable *charges* is the outcome of interest; it is the total individual medical costs (in thousands of dollars) billed by the beneficiary's health insurance. Here are tabular representations of two trees from a 100-tree random forest built on this dataset:

node_id	left_child	right_child	split_variable	split_point	prediction
1	2	3	smoker	yes	13.19
2	4	5	region	NE	8.65
3	6	7	region	NE, SE, SW	31.94
4	8	9	children	2.50	7.85
5	10	11	children	0.50	8.92
6	12	13	bmi	30.01	29.21
7	14	15	bmi	30.30	36.26
8	16	17	bmi	25.19	7.29
9	18	19	children	3.50	11.61
10	0	0		0.00	8.11
11	0	0		0.00	9.55
12	0	0		0.00	20.72
13	0	0		0.00	41.93
14	0	0		0.00	23.15
15	0	0		0.00	44.51
16	0	0		0.00	10.95
17	0	0		0.00	6.82
18	0	0		0.00	12.49
19	0	0		0.00	8.09

node_id	left_child	right_child	split_variable	split_point	prediction
1	2	3	smoker	yes	13.24
2	4	5	bmi	31.30	8.33
3	6	7	region	NE, NW	31.71
4	8	9	sex	2.00	7.56
5	10	11	region	NE, NW, SE	9.24
6	12	13	children	2.50	29.45
7	14	15	bmi	30.10	33.69
8	16	17	children	0.50	7.27
9	18	19	children	1.50	7.82
10	0	0		0.00	8.69
11	0	0		0.00	11.09
12	0	0		0.00	28.39
13	0	0		0.00	33.71
14	0	0		0.00	22.11
15	0	0		0.00	41.00
16	0	0		0.00	6.79
17	0	0		0.00	7.72
18	0	0		0.00	7.34
19	0	0		0.00	9.04

Question 13.5

Draw the two regression trees from the Insurance Cost Dataset random forest that are represented by these tables. Note: if a variable's value is less than the split point at a particular node, the training sample goes to the left. For categorical variables, if the variable's value is one of the categories listed under "split point", the training sample goes to the right.

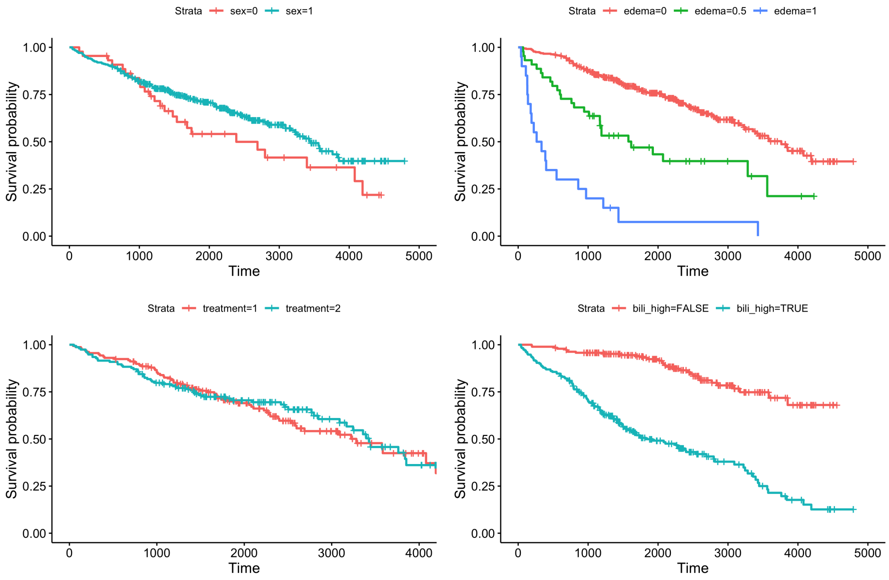
13.4 Model Parameters

13.4.1 Splitting Criteria

We already discussed how individual trees are built in Chapter 7. One of the choices we made then was which **splitting criterion** to use in building the tree. The splitting criterion is some way of deciding which variables are "good" to split on. For classification trees, the most common criteria are Gini index and information gain (the Gini index is by far the most popular). For regression, **variance reduction** (the same idea as standard deviation reduction) is the most common criterion.

Question 13.6

The following data come from a Mayo Clinic trial of the drug D-penicillamine for primary biliary cirrhosis (PBC) of the liver. The trial was conducted between 1974 and 1984. The data shown here are for 418 patients who completed the trial. The dataset comes from the `survival` package in R and contains information on 17 predictors, as well as the follow-up time and outcome (death or censoring) for each patient. Here are Kaplan-Meier curves (see Chapter 11) for four predictors, one of which (bilirubin) I manually binarized. Bilirubin is considered high if it is greater than 1.2 mg/dL.



Say you wanted to build a decision tree to predict survival in PBC. You would want to choose splits for which survival looks very *different* on either side of the split. This is analogous to choosing splits that increase the purity of the outcome (for classification) or reduce the variance of the outcome (regression). Speculate on how you might build such a tree. We will discuss the process of constructing **random survival forests** in much greater detail after we've seen a bit more survival analysis.

13.4.2 Creating Split Points

Each node within a tree signifies a division of one of the predictors into two groups. **Deterministic splitting** means considering all possible splits and identifying the best one. For a numeric predictor, this involves considering all of the values of the predictor represented in the dataset; there may be as many as n possible values, where n is the number of training samples included in the tree. For a categorical predictor, this involves dividing the possible categories into two **complementary groups**. For example, the insurance cost dataset in Section 13.3 contains a predictor `region` with four categories: northeast, southeast, southwest, and northwest. Each split decision must consider all

$2^{k-1} - 1$ possible divisions[2] of those categories, where k is the number of categories. For `region`, the possible splits are:

Group 1	Group 2
NE, SE, SW	NW
NE, SW, NW	SE
NE, SE, NW	SW
SE, SW, NW	NE
NE, SE	NW, SW
NE, SW	NW, SE
NE, NW	SE, SW

Deterministic splitting becomes problematic when the number of possible splits is large. In that case, software packages often employ **random splitting**, in which a predetermined number of possible splits (the exact number is set using a parameter) are randomly chosen from among all the possibilities.

Question 13.7

A known issue with decision trees is their tendency to prefer to split on continuous predictors over discrete predictors. Why do you think this is? It can be avoided, in part, by using random splitting with a fixed number of possible splits.

13.4.3 Bag Size and Number of Predictors

Each tree in a random forest is built using a "bagged" sample of n training examples from the original N examples. Typically around $2/3$ of training examples are used per tree, but most software packages include a parameter that allows the user to set this value. In addition, at each split, a tree considers only a randomly-chosen subset of p predictor variables; the default number is usually $\sqrt{P}$ (for classification problems) or $P/3$ (for regression problems), where P is the total number of predictors. In software, this will also be a

[2] This comes from taking 2^k (total combinations of k categories), subtracting 2 (all in or all out, neither of which is possible), and then dividing the whole thing by 2 (because the ordering of the groups doesn't matter). $(2^k - 2)/2 = 2^{k-1} - 1$

settable parameter. Usually it's fine to leave these parameters at their default values.

Question 13.8

For an ensemble to be more accurate than any of its individual members, the learners comprising the ensemble must be *accurate* and *diverse*. Accuracy means that the learners must perform better than random on their designated task. Diversity means that the classifiers must make different errors on new data points. How do the parameters n and p impact diversity? How does the way the trees are trained ensure accuracy?

13.4.4 Node Size and Tree Depth

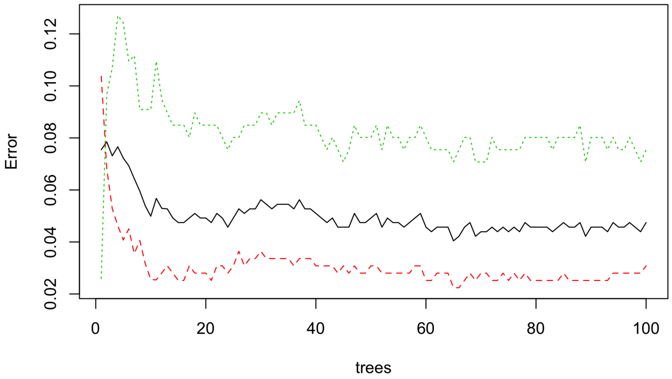
Software packages generally allow the user to control the growth of individual trees within a random forest by specifying node size and tree depth parameters. The **node size** parameter governs the minimum number of training samples present at a node for a split to be considered. The **tree depth** parameter governs the maximum number of connections between the root of the tree and one of its leaves. A split at a particular node will only be considered when there is still some impurity in the outcome at that node (see Chapter 7) and when:

1. The current tree depth is less than the maximum allowed tree depth.
2. The number of samples at a node is at least 2x the minimum node size (since a binary split on a smaller node would result in leaves with less than the minimum required node size).

13.5 The Out-of-Bag Error

The random forest will report a number called the **out-of-bag (OOB) error** as it runs. To calculate OOB error, each tree makes a prediction for each of the training samples *not* used in its construction. This provides an ongoing estimate of the generalization error of the forest.

Here is an OOB error plot for the random classification forest built on the Wisconsin Breast Cancer dataset:



The black line shows the overall OOB error (the percent of OOB points misclassified) after the addition of each new tree. The red line shows the OOB error for points of class 1, which in this case is *B* (benign), and the green line shows the OOB error for points of class 2, which in this case is *M* (malignant).

Question 13.9

What does it mean that the green line is so much higher than the red line? What does this tell you about the relative rates of false positives and false negatives for this random forest?

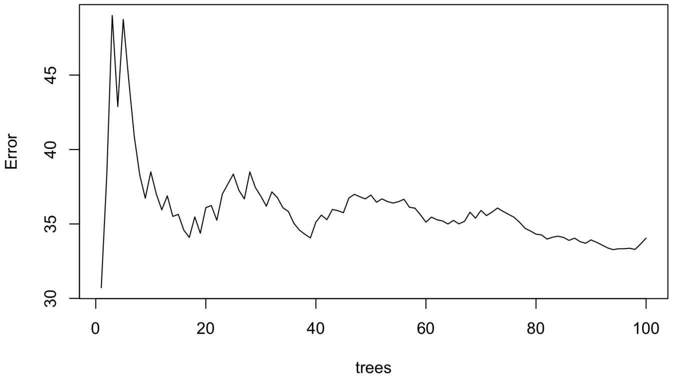
Question 13.10

Here is the final **confusion matrix** for the Wisconsin Breast Cancer random forest.

```
      B    M  class.error
B   346   11   0.03081232
M    16  196   0.07547170
```

It is probably more important to avoid false negatives (*M* tumors that are classified as *B*) than false positives. Speculate on ways in which you could force the forest to produce a lower rate of false negatives, even if it means increasing the number of false positives.

For regression forests, the OOB error is calculated differently. It is defined as the mean square error among the OOB samples. The square root of this error is the average absolute value of the difference between the predicted and actual costs.



13.6 Variable Importance Measures

One of the main disadvantages of random forests is their lack of clarity around which variables are "important". In a regression model, the model output contains hypothesis tests and coefficients for each variable that provide the user with an interpretable importance ranking. Nothing this simple exists for random forests. However, there are some heuristics for ranking variables. These fall into two camps.

13.6.1 Impurity-Based Importance

Trees are built by choosing splits that reduce uncertainty, or impurity, in the outcome. This impurity reduction is a measure of how much splitting on that variable "helps" in purifying the outcome. One way to measure the importance of a variable, therefore, is to average the decrease in node impurity across all splits involving that variable, across all trees in the random forest[3]. This importance measure is called the **Mean Decrease in Impurity (MDI)**. It

[3] Because splits occur at different heights, impurity reduction is typically weighted by how many samples reach a given node.

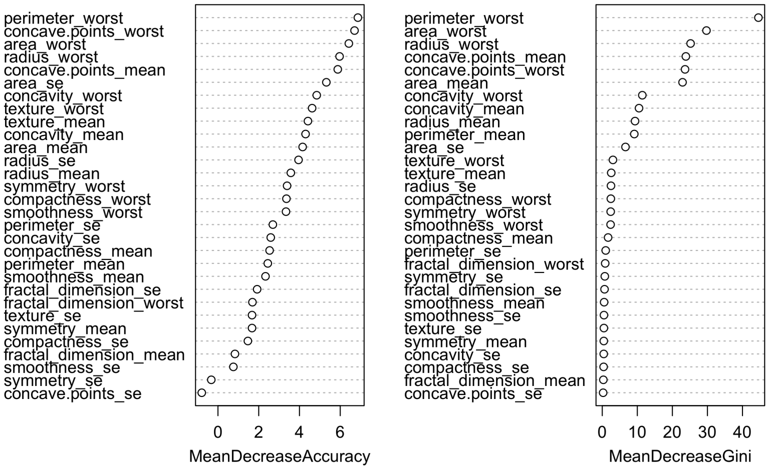
works no matter what your outcome is. Its main advantage is that because the reduction in impurity is what is already used to determine the splits, it requires very little additional computation.

13.6.2 Permutation-Based Importance

An alternative way of measuring the importance of variable *j* is to see how much it affects the predictive accuracy of trees across the whole forest. We assess this using the OOB samples. First the real OOB error is calculated (by running each sample through the trees for which it is OOB). Then the values of variable *j* across the OOB samples are randomly permuted, and the OOB error is calculated again. We expect the OOB error to go up in the second case by an amount proportional to how important variable *j* is. We then average this difference for each variable across all trees. This permutation-based importance measure is called the **Mean Decrease in Accuracy (MDA)**. Again, it works no matter what the outcome is. Its main advantage is that it is more interpretable than MDI.

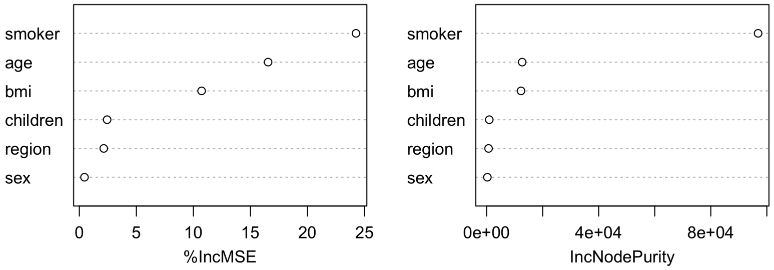
Question 13.11

Here is a variable importance plot for the Wisconsin Breast Cancer classification forest. Which side is MDI and which is MDA? Which variables are most and least important?



Question 13.12

Here is a variable importance plot for the insurance cost dataset regression forest. Which side is MDI and which is MDA? Which variables are most and least important?



13.7 A Note on Software Packages

As we have seen in Chapter 7, there are many different ways to build and optimize decision trees. There are even more ways to build and optimize random forests. This chapter uses the `randomForest` R package by Andy Liaw, a faithful implementation of the original random forest implementation suggested by Breiman (2003), as well as `randomForestSRC`, a faster and more recent package by Ishwaran and Kogalur that provides a unified interface for random forest-based classification, regression, and survival analysis. The `scikit-learn` package in Python provides implementations of random forests for both classification and regression.

Chapter 14

Introduction to Boosting

Random forests (Chapter 13) are one approach to ensemble learning. Today we will examine another approach, **boosting**, that relies on a completely different set of ideas. The first practical boosting algorithm, AdaBoost, was invented in 1995 by Freund and Schapire. However, boosting is a general approach to choosing a set of weak learners that, together, create a strong learner. Since the same idea has spawned many different approaches, boosting is referred to as a **meta-algorithm**.

14.1 AdaBoost

Assume we are trying to solve a supervised learning problem. As usual, our data look like this:

$$\{(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)})\}$$

where the $x^{(i)}$ are feature vectors of length p. For now, we will assume that the outcome, Y, is binary (so this is a binary classification problem – see Chapter 2). We'll make one small notational change from earlier chapters. Instead of $Y \in \{0, 1\}$, we'll say $Y \in \{-1, 1\}$. The fact that the positive and negative training examples have opposite sign will help us write the algorithm more concisely.

The basic strategy behind AdaBoost is to build classifiers in series while maintaining a set of weights for the different training examples. The weights change with each subsequent classifier, increasing if previous classifiers have made incorrect predictions and decreasing if previous classifiers were correct. In this way, the importance of difficult-to-classify training examples increases, leading us to prefer future classifiers that successfully predict these examples.

Here is the algorithm:

1. Initialize the observation weights to $w_i^{(1)} = \frac{1}{N}$ for $i = 1, \ldots, N$.
2. For $m = 1, \ldots, M$:
   (a) Select a classifier, $G_m(x)$, that minimizes the weighted training error according to the current set of weights, $w_i^{(m)}$. Depending on the algorithm, it may be possible to train a single classifier on the weighted training set; in other cases, one may need to select the best-performing classifier from among a predefined set.

   (b) Compute
   $$\mathrm{err}_m = \frac{\sum_{i=1}^{N} w_i^{(m)} \cdot \mathcal{I}(y^{(i)} \neq G_m(x^{(i)}))}{\sum_{i=1}^{N} w_i^{(m)}}$$

   (c) Compute voting weight for classifier m:
   $$\alpha_m = \log\left(\frac{1 - \mathrm{err}_m}{\mathrm{err}_m}\right)$$

   (d) Set
   $$w_i^{(m+1)} := w_i^{(m)} \cdot \exp\left[\alpha_m \cdot \mathcal{I}(y^{(i)} \neq G_m(x^{(i)}))\right]$$
   for $i = 1, \ldots, N$.
3. Output
   $$G(x) = \mathrm{sign}\left[\sum_{m=1}^{M} \alpha_m G_m(x)\right]$$

The output definition in step 3 is really the key to the whole thing. Your goal is to construct a set of classifiers whose votes will be added together to produce an overall decision. AdaBoost provides one possible set of instructions for how to choose/build the individual classifiers, $G_m(x)$, and how to weight their votes.

14.2 Visualizing the Steps of AdaBoost

We will now train AdaBoost on the happiness dataset from Section **??**. Here is the dataset, using the new notation for Y. The astute observer will notice that we've added a fourth covariate, X_4 (whether or not the person has a pet). The reason is that subjects 5 and 10 are exactly the same otherwise, so we can never get perfect separation using the original dataset.

Subject ID	friends (X_1)	money (X_2)	free time (X_3)	pet (X_4)	happy (Y)
1	1	1	0	0	-1
2	1	1	1	0	-1
3	0	1	1	0	-1
4	0	0	0	0	-1
5	1	0	0	0	-1
6	0	0	0	0	-1
7	1	2	1	0	1
8	1	0	1	0	1
9	0	0	1	1	1
10	1	0	0	1	1

$$X_1 = \begin{cases} 0 & \text{no friends} \\ 1 & \text{friends} \end{cases} \qquad X_2 = \begin{cases} 0 & \text{poor} \\ 1 & \text{enough money} \\ 2 & \text{rich} \end{cases}$$

$$X_3 = \begin{cases} 0 & \text{no free time} \\ 1 & \text{some free time} \end{cases} \qquad X_4 = \begin{cases} 0 & \text{no pet} \\ 1 & \text{has a pet} \end{cases}$$

Now, let's go through the process of applying AdaBoost to this dataset, step by step.

(a) Initialize the observation weights for the training data. (Make them uniform.)

$w_1^{(1)} =$

$w_2^{(1)} =$

$w_3^{(1)} =$

$w_4^{(1)} =$

$w_5^{(1)} =$

$w_6^{(1)} =$

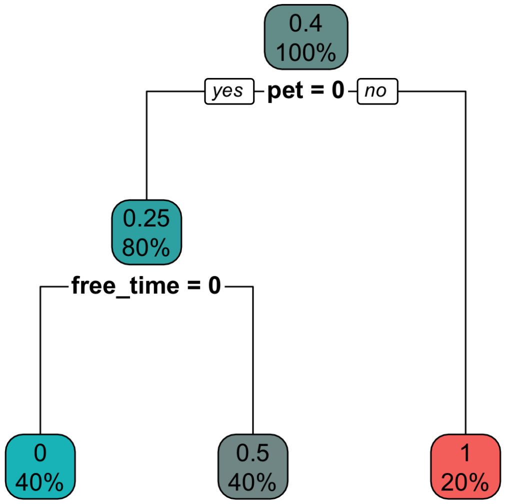
$w_7^{(1)} =$

$w_8^{(1)} =$

$w_9^{(1)} =$

$w_{10}^{(1)} =$

(b) We will now grow a decision tree on this dataset, $G_1(x)$, using these weights. We'll use the `rpart` package in R, just as in Section **??**. Here is the first tree.



Here are the predictions for this tree, along with the current weights.

Datapoint ID	$w_i^{(1)}$	happy (Y)	$G_1(x)$
1	0.1	-1	-1
2	0.1	-1	-1
3	0.1	-1	-1
4	0.1	-1	-1
5	0.1	-1	-1
6	0.1	-1	-1
7	0.1	1	-1
8	0.1	1	-1
9	0.1	1	1
10	0.1	1	1

Compute the misclassification error of this tree, err_1. Compare this tree to the one we constructed by hand in Chapter 7.

$$\text{err}_1 =$$

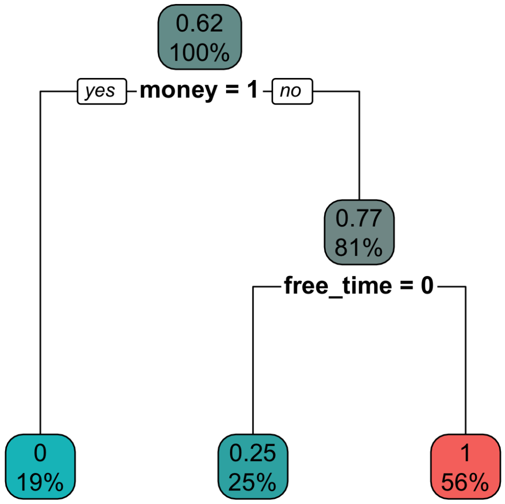
(c) Based on how $G_1(x)$ performs, calculate α_1, its voting weight.

$$\alpha_1 =$$

(d) Re-weight the observation weights for the training examples.

$w_1^{(2)} =$ $w_6^{(2)} =$
$w_2^{(2)} =$ $w_7^{(2)} =$
$w_3^{(2)} =$ $w_8^{(2)} =$
$w_4^{(2)} =$ $w_9^{(2)} =$
$w_5^{(2)} =$ $w_{10}^{(2)} =$

(e) Now we grow another decision tree, $G_2(x)$, using these new weights as inputs to the `rpart` package.



Here are the predictions for this tree, along with the current weights.

Datapoint ID	$w_i^{(2)}$	happy (Y)	$G_2(x)$
1	0.1	-1	-1
2	0.1	-1	-1
3	0.1	-1	-1
4	0.1	-1	-1
5	0.1	-1	-1
6	0.1	-1	-1
7	0.4	1	1
8	0.4	1	1
9	0.1	1	1
10	0.1	1	-1

Compute the misclassification error of this tree, err_2.

$$\text{err}_2 =$$

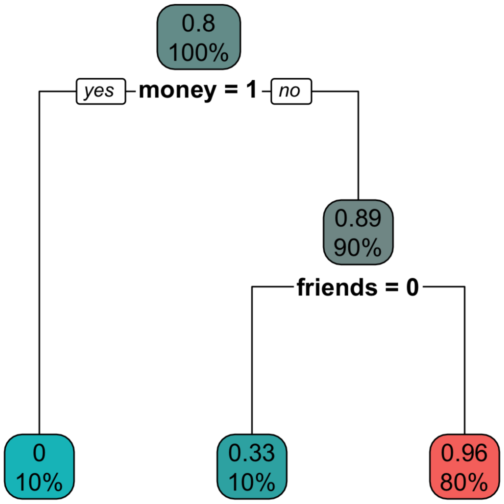
(f) Based on how $G_2(x)$ performs, calculate α_2, its voting weight.

$$\alpha_2 =$$

(g) Re-weight the observation weights for the training examples.

$w_1^{(3)} =$ $w_6^{(3)} =$
$w_2^{(3)} =$ $w_7^{(3)} =$
$w_3^{(3)} =$ $w_8^{(3)} =$
$w_4^{(3)} =$ $w_9^{(3)} =$
$w_5^{(3)} =$ $w_{10}^{(3)} =$

(h) Now we grow another decision tree, $G_3(x)$, using these new weights as inputs to the `rpart` package.



Here are the predictions for this tree, along with the current weights.

Datapoint ID	$w_i^{(3)}$	happy (Y)	$G_3(x)$
1	0.1	-1	-1
2	0.1	-1	-1
3	0.1	-1	-1
4	0.1	-1	-1
5	0.1	-1	1
6	0.1	-1	-1
7	0.4	1	1
8	0.4	1	1
9	0.1	1	-1
10	1.5	1	1

(i) Compute the misclassification error of this tree, err_3.

$$\text{err}_3 =$$

(j) Based on how $G_3(x)$ performs, calculate α_3, its voting weight.

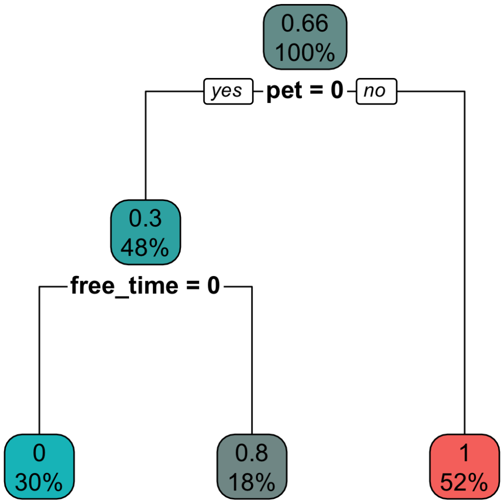
$$\alpha_3 =$$

(k) Re-weight the observation weights for the training examples.

$w_1^{(4)} =$
$w_2^{(4)} =$
$w_3^{(4)} =$
$w_4^{(4)} =$
$w_5^{(4)} =$

$w_6^{(4)} =$
$w_7^{(4)} =$
$w_8^{(4)} =$
$w_9^{(4)} =$
$w_{10}^{(4)} =$

(l) Now we grow another decision tree, $G_4(x)$, using these new weights as inputs to the `rpart` package.



Here are the predictions for this tree, along with the current weights.

Datapoint ID	$w_i^{(4)}$	happy (Y)	$G_4(x)$
1	0.1	-1	-1
2	0.1	-1	-1
3	0.1	-1	-1
4	0.1	-1	-1
5	1.4	-1	1
6	0.1	-1	-1
7	0.4	1	1
8	0.4	1	1
9	1.4	1	1
10	1.5	1	1

(m) Compute the misclassification error of this tree, err_4.

$$\text{err}_4 =$$

(n) Based on how $G_4(x)$ performs, calculate α_4, its voting weight.

$$\alpha_4 =$$

(o) Output the weighted average of the four classifiers' votes for each training example:

$$G(x) = \text{sign}\left[\alpha_1 G_1(x) + \alpha_2 G_2(x) + \alpha_3 G_3(x) + \alpha_4 G_4(x)\right]$$

What is the final training error?

Datapoint ID	happy (Y)	$G_1(x)$	$G_2(x)$	$G_3(x)$	$G_4(x)$	$G(x)$
1	-1	-1	-1	-1	-1	
2	-1	-1	-1	-1	1	
3	-1	-1	-1	-1	1	
4	-1	-1	-1	-1	-1	
5	-1	-1	-1	1	-1	
6	-1	-1	-1	-1	-1	
7	1	-1	1	1	1	
8	1	-1	1	1	1	
9	1	1	1	-1	1	
10	1	1	-1	1	1	

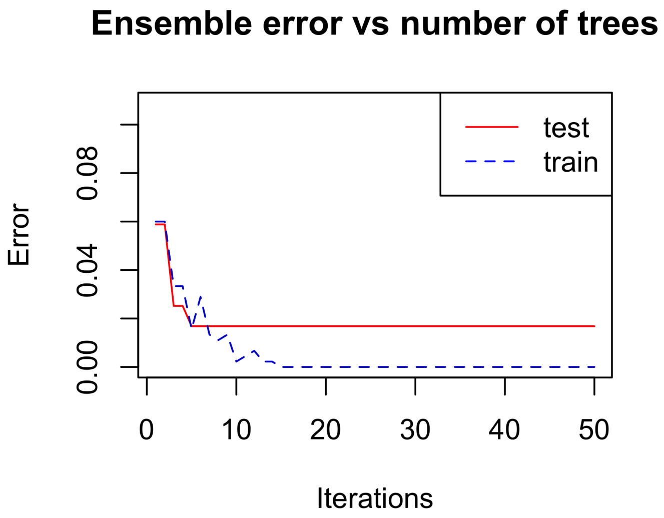
Question 14.1

The most difficult thing to understand in all of this is how the updated weights play into the construction of subsequent trees. A clue comes from the dataset percentages shown in the nodes of each tree. For example, trees 1 and 3 actually have the same structure, but the impurities at each node are different due to the different weights. Discuss how the weights could inform the variables the splitting algorithm chooses to split on (revisit Chapter 7, if necessary, to see the math).

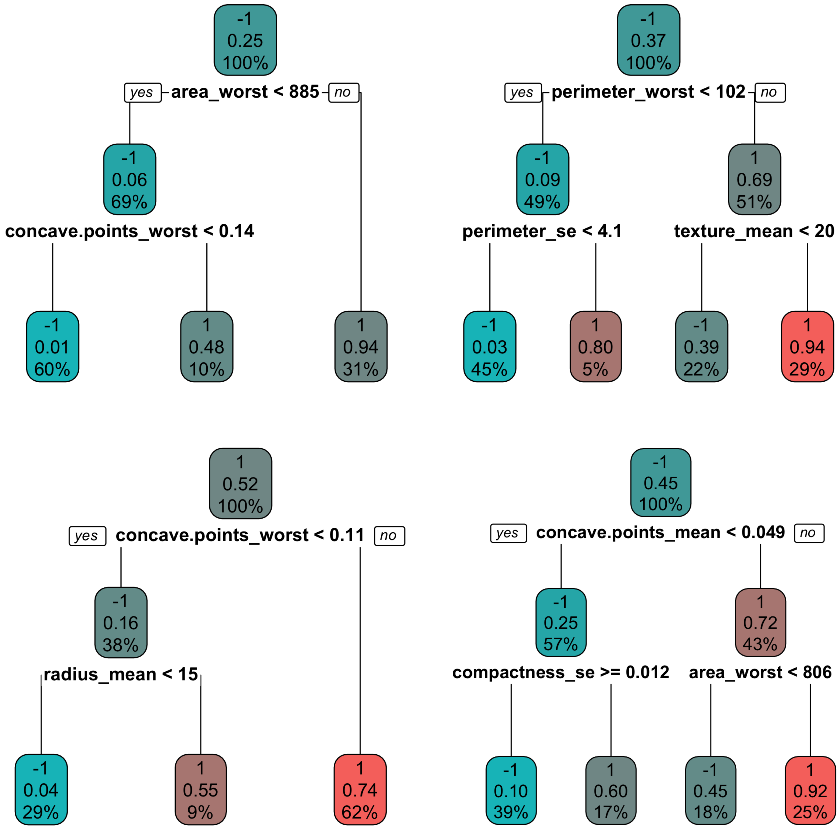
14.3 Revisiting Breast Cancer Classification

Let's once again revisit the classification example for which we built a single decision tree in Chapter 7 and a random forest in Chapter 13. The Wisconsin Breast Cancer Dataset contains information about 30 different imaging features of fine needle aspirate (FNA) samples from breast masses in 569 study participants.

Here is a plot showing the evolution of the training and test error as a function of the number of trees:



where 450 samples are used for training and the remaining 119 for testing. Here are tree numbers 1, 3, 6, and 8:



Question 14.2

Compare and contrast boosting and random forests on the basis of:

- Whether each tree uses all or part of the dataset
- Whether they consider a subset of the predictors at each split or all the predictors
- Whether you can parallelize the construction of different trees (i.e., build them at the same time on different processors)
- Whether the votes of different classifiers are independent
- Ease of use and interpretability

Chapter 15

Model Quality and the Bias-Variance Tradeoff

We have now seen several different algorithms (and meta-algorithms) for supervised learning. We've examined linear models for classification and regression (Chapters 8 and 9), KNN (Chapters 2 and 3), decision trees (Chapters 7), and ensemble methods – usually based on decision trees – including random forests (Chapter 13) and boosting (Chapter 14). In this short chapter, we'll take a step back and talk about some key themes that are relevant to all of these methods.

15.1 Measuring Error: Loss and Objective Functions

Most of us think of error as a fairly intuitive concept, and so far in our discussions of model building and model quality, we've avoided any formal definitions. However, as we begin to extend the concepts we've learned from classification and regression to more challenging problem classes (e.g., survival analysis), a more formal conception of what error means and how to minimize it algorithmically becomes increasingly crucial. Here are some examples:

- In classification, error typically means either the total number of misclassified points,

$$\text{error} = \sum_{i=1}^{n} \mathcal{I}\left(y^{(i)} \neq \hat{y}^{(i)}\right),$$

or a weighted average of the number of misclassified points per class. Another way of quantifying classification error is **AUC**, the area under the receiver operating characteristic curve, which is the probability that a randomly chosen positive example (where $y = 1$) will receive a higher score from the model (higher $\hat{y}$) than a randomly chosen negative example (where $y = 0$).

- In regression, error typically means the **mean-squared error (MSE)**,

  $$\text{error} = \frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)} - \hat{y}^{(i)}\right)^2,$$

  although there are also other ways of quantifying error. For example, you could use the **mean absolute error**

  $$\text{error} = \frac{1}{n}\sum_{i=1}^{n}|y^{(i)} - \hat{y}^{(i)}|.$$

  Neither is "wrong" or "right", but they have different properties. For example, the MSE gives a higher weight to large errors (outliers). It's also differentiable, which is why it's used so much more often in, e.g., neural networks than the mean absolute error.

- In survival analysis, error is typically quantified using something called **Harrell's concordance index**, which takes into account censored observations[1].

Question 15.1

Harrell's concordance (C) index is probably unfamiliar to you. It is calculated like this:

[1] For a detailed explanation and some experimental results, see Kattan MW, Hess KR and Beck JR (1998). "Experiments to determine whether recursive partitioning (CART) or an artificial neural network overcomes theoretical limitations of Cox proportional hazards regression." *Computers and Biomedical Research*, 31(5), 363-373.

1. Create a list of all possible pairs of patients. There will be $n(n-1)/2$ pairs.
2. Eliminate all pairs for which the patient with the shorter follow-up time does not experience the event of interest (i.e., is censored). The remaining patient pairs are considered "usable" since the patient with the shorter time-to-event is identifiable.
3. Count the number of usable patient pairs for which the patient with the shorter follow-up time had the higher predicted hazard for the event. That is, you want the number of pairs for which the model's predictions about relative times to event are consistent with the observed data.
4. The C statistic is the number of consistent pairs divided by the number of usable pairs. The error is defined as $1 - C$.

Here are four patients and the predicted mean time to event from two different survival models (low value means lower time to event). All times are in years. Calculate the C statistic for both models.

Patient	Follow-up Time	Observed?	Model 1 Score	Model 2 Score
1	8.3	1	4.6	5.2
2	6.5	0	2.3	7.1
3	2.7	1	0.6	6.7
4	7.4	1	4.7	6.6

First Patient	Second Patient	Usable	Model 1 Consistent	Model 2 Consistent
1	2			
1	3			
1	4			
2	3			
2	4			
3	4			

You should find that the C statistic for Model 1 is 0.75, while the C statistic for Model 2 is only 0.25. Model 1 is clearly the better model.

The definition of error used to optimize a particular model is called a **loss function**. Loss functions are a general concept from optimization and decision theory; they represent the cost associated with a decision. If we think of a supervised learning model as an engine for making decisions, the loss is how "bad", on average, those decisions will be. Loss functions are, in turn, part of a broader class of functions called **objective functions**. Most learning algorithms work by either minimizing some measure of "badness" (a loss function) or maximizing some measure of "goodness" (negative of the loss, alternatively called a **reward function**).

Question 15.2

Imagine what would happen if, in Question 15.1, we simply set a follow-up time of 7 years and treated the problem as a classification problem, calculating the AUC for the two models based on that time horizon and ignoring censoring.

Positive Patient	Negative Patient	Model 1 Ranks Pos Higher?	Model 2 Ranks Pos Higher?
3	1	1	0
3	2	1	1
3	4	1	1

We would calculate an AUC of 1.0 for Model 1 and 0.67 for Model 2. Do you think this is a good approach to quantifying error for this problem? Why or why not?

The choice of how to define error, and which loss function to use for quantifying that error, ultimately rests with the model builder. The process of model training is about adjusting the available parameters of the model to minimize the loss.

15.2 Goodness of Fit vs. Generalizability

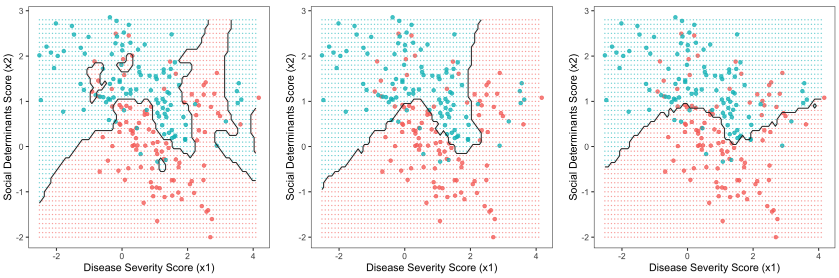
Once we have an appropriate objective function, we can set about building a model to optimize it. This requires us to think about what constitutes a "good model". It's a bit more complicated than simply minimizing the loss on our training data, because ideally we want a model that is both accurate and **parsimonious**, meaning that it is as simple as possible without sacrificing performance. Another way of thinking about this is that we create models to tell stories about the data we see. If they are good stories, they will do two things:

1. Explain the structure of the data that are used to train them (high **goodness of fit**)
2. Make accurate predictions on new data (good **generalizability**)

For a supervised learning model, we quantify (1) using the **training error** (error on the training set) and (2) using the **test error** (error on an independent test set).

Question 15.3

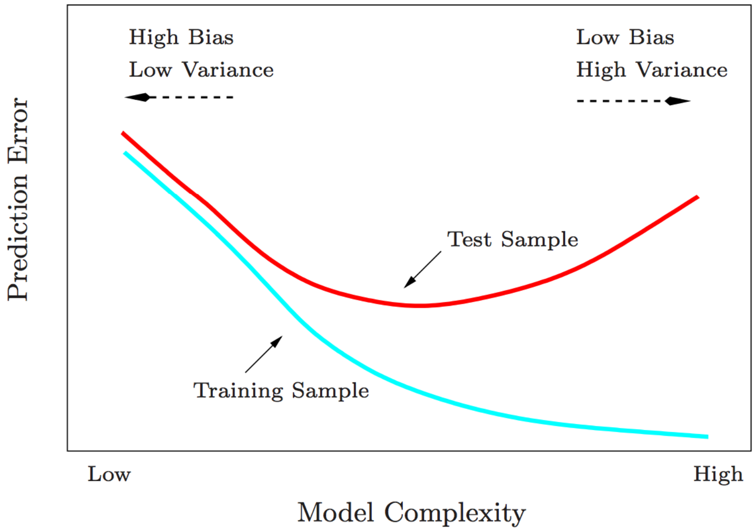
Here we see three decision boundaries for KNN with different values of K (the number of neighbors considered in making a prediction). The data are for the two-class classification problem first discussed in Chapter 2. From left to right, $K = 3, 15$, and 50. What are the tradeoffs in moving from left to right in terms of (a) training error/goodness of fit and (b) test error/generalizability?



In supervised learning, **model complexity** (loosely defined as the effective number of parameters the model must fit) is, therefore, a very important consideration. A model that is not complex enough will fail to capture all of the structure in the training data, and both its training and test error will suffer as a result. We call this situation **underfitting**. Conversely, a model that is too complex may fit the training data very well – maybe perfectly – but will fail to generalize well to new data. We call this situation **overfitting**.

15.3 Bias vs. Variance

It turns out that there is a general principle governing model complexity in supervised learning called the **bias-variance tradeoff**. It is perhaps best illustrated by this figure, which comes from the excellent (and free) book *Elements of Statistical Learning*, by Tibshirani, Hastie, and Friedman (Figure 2.11):



The figure shows us that test error, our measurement of generalization error, comes from two different sources:

$$\text{test error} = \text{bias} + \text{variance}.$$

The term **bias** refers to error that results from underfitting, while **variance** results from overfitting.

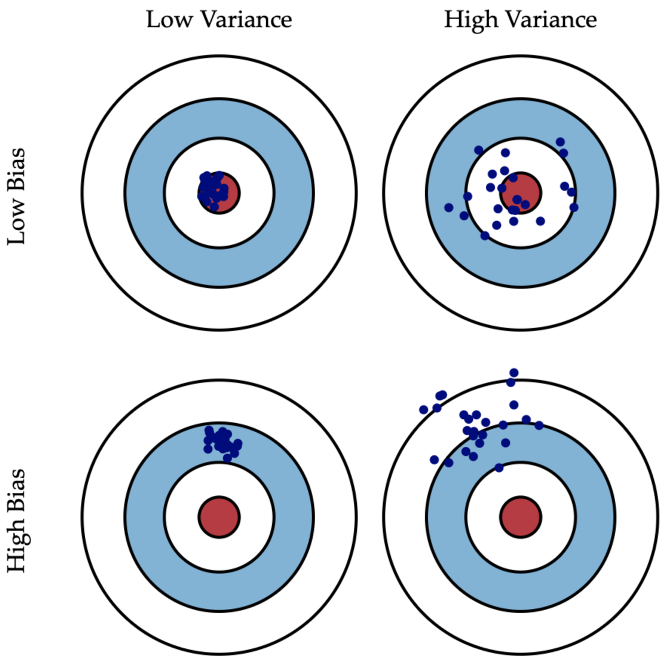
Error due to bias can often be reduced by introducing more/different features or by using a different learning algorithm. Error due to variance can often be reduced by increasing the size or diversity of the training data. That's why it's important to know which situation you're in; gathering more training data when your model is too simple is unlikely to help you, and deploying an extremely fancy (and complicated) deep learning model when logistic regression has already overfit your training data is also unlikely to help.

Another consideration is that even the "sweet spot" (minimum test error) on the bias-variance tradeoff curve can still be high. This occurs when the data you have available simply aren't sufficient to answer the question – maybe you have the wrong features or there is no relationship between the features and the outcome. Or maybe your features aren't measured correctly. In my experience, these types of issues are not considered often enough in the clinical domain.

Question 15.4

A recent review article had the provocative title "A systematic review shows no performance benefit of machine learning over logistic regression for clinical prediction models." (Christodoulou E, Ma J, Collins GS, Steyerberg EW, Verbakel JY, Van Calster B. *Journal of Clinical Epidemiology*, 2019, 110:12-22). Assuming this finding is true, what is it telling us about the nature of the error in most clinical risk prediction models? What does it suggest about what we should be doing to improve the performance of these models?

Another way of thinking about bias and variance is in terms of what happens when you make slight changes to the training data (for example, by collecting new data from the same patient population, or drawing repeated bootstrap samples from the same training set). The figure below is from the article *Understanding the Bias-Variance Tradeoff*, by Scott Fortmann-Roe:



Think of each dot as representing a single test example evaluated under the same model trained on slightly different datasets. The center of the target is the prediction the model should make for that test example. In the case of high bias and low variance, all of the models are off, but they are “wrong in the same way”. Even if you average their predictions, the answer is still way off the mark. In the case of high variance, the models all make incorrect predictions, but their predictions are off in different directions. As a result, if you average their outputs, you’ll get closer to the right answer.

Question 15.5

We saw random forests in Chapter 13 and were introduced to boosting in Chapter 14. It is interesting to compare the two methods because they reduce test error in different ways: one primarily tackles variance, while the other primarily tackles bias. Which one is which, and how does each algorithm succeed in reducing its respective form of error?

Chapter 16

Regularization

The ideas we examined in Chapter 15

16.1 Lasso, Ridge, and Elastic Net

Sometimes when building regression models, you run into issues like the following:

- You have more predictors, p, than you have samples, n.
- Your predictors are highly correlated.

Both of these conditions can lead to models that are highly unstable. Maybe they fit your training data well, but if you change your training set even a tiny bit, the coefficients shift wildly. It becomes very hard to trust the coefficient values under these circumstances. One way to combat this is to introduce a **penalty** on the values of the coefficients. There are different types of penalty (see slides) that do different things. Relevant terms include: **ridge regression**, **Lasso**, and **elastic net**.

16.2 Gradient Boosting

Jerome Friedman and Leo Breiman generalized AdaBoost into a general framework called **gradient boosting**. In this framework, of which AdaBoost is a subset, there are three components:

1. A loss function to be optimized.
2. Weak learners to make predictions.
3. An additive model that adds the contributions of different weak learners to minimize the loss function.

We don’t have time to get into the details of the gradient boosting framework today, but its basic advantage is that it formulates the boosting process in such a way that any differentiable loss function can be used. In addition, although classification trees or regression trees are usually the weak learners (and technically, Friedman defined “gradient boosting” as a model that uses trees as learners) the framework is general enough to encompass other types of weak learners.

Chapter 17

Bias in EHR Studies

In recent years, observational data have begun to

> Bias: Prejudice in favor of or against one thing, person, or group compared with another, usually in a way considered to be unfair. (from Oxford English Dictionary)

> "[EHRs] present an opportunistic and non-random snapshot of patient interactions, capturing only the information that is relevant to each specific encounter and across a variety of contexts that are represented by how a patient interacts with the health system."
> -Phelan *et al*

this chapter, we pause our discussion of methods to reflect on some of the key sources of bias in studies that use EHR data.

When we build a supervised learning model or apply a hypothesis test, we are attempting to use data to draw a conclusion about the world.

[1]

[1] Gianfrancesco MA, Tamang S, Yazdany J, Schmajuk G. Potential biases in machine learning algorithms using electronic health record data. JAMA internal medicine. 2018 Nov 1;178(11):1544-7.

[2]

[3]

[4]

Studies based on EHR data are part of an older tradition of **retrospective observational studies**. The potential sources of bias in these types of studies have been known for decades - in some cases, centuries.

Many of the definitions of biases that we will investigate today are from the wonderful website `catalogofbias.org`.

Question 17.1

As clinicians, what information do you think is over-represented in EHRs? What information do you tend *not* to record?

Question 17.2

It gets a little tiresome hearing all the problems and potential sources of bias in EHR studies. There are positive aspects to working with EHR data as well that cannot be accomplished by other means. What are some reasons that it is worth it to attempt to mitigate the effects of bias in EHR studies instead of giving up?

Question 17.3

In each case, think about how each of the following studies would be affected. (1) Predictive model of in-hospital mortality. (2) Study of the effectiveness of a particular treatment among patients admitted to the hospital for an acute illness. (3) Observational study of the utility of a particular lab value as a predictive biomarker for long-term development of a chronic illness. (4) Same as (3) except it's a machine learning-based study that uses 200 different patient characteristics as predictors.

[2] Goldstein BA, Navar AM, Pencina MJ. Risk prediction with electronic health records: the importance of model validation and clinical context. JAMA cardiology. 2016 Dec 1;1(9):976-7.

[3] Phelan M, Bhavsar NA, Goldstein BA. Illustrating informed presence bias in electronic health records data: how patient interactions with a health system can impact inference. eGEMs. 2017;5(1).

[4] Agniel D, Kohane IS, Weber GM. Biases in electronic health record data due to processes within the healthcare system: retrospective observational study. Bmj. 2018 Apr 30;361.

Imagine you are trying to decide if you believe a particular study. You're basically asking yourself whether the study's result is true or whether it could be due to (a) random chance, (b) confounding, or (c) bias. If it's unlikely that the result was due to one of these factors, the study is considered **internally valid**. However, the study is not deemed **externally valid** if it cannot be expected to generalize well to other circumstances (e.g., other health systems or patient populations).

The authors of one article (Phelan *et al*) call the set of biases introduced by how a patient interacts with a health system **informed presence bias**. The idea is that patient interactions with a health system are typically informative.

17.1 Confounding

What happens if the potential confounder is unknown or not available in the dataset? For example, access to housing or transportation may be the direct cause of adverse outcomes in patients with low socioeconomic status, but we may attribute the association to insurance status or race because those things are more easily measured and recorded in the EHR.

Example: Confounding by indication is a distortion that modifies an association between an exposure and an outcome, caused by the presence of an indication for the exposure that is the true cause of the outcome.

A third factor that affects both exposure and outcome. Can be adjusted for, assuming you can measure it.

Difference with bias: you can't adjust for it. It's just part of your study. Bias arises whenever you treat the exposed/unexposed group differently. Tends to bias toward the null or away from the null. We often don't know the direction of bias but we know the bias is there.

17.2 Selection Bias

Problem: different groups within the study are selected in different ways; undermines internal validity

Definition: occurs when individuals or groups in a study differ systematically from the population of interest leading to a systematic error in an association or outcome.

Messes up internal validity of study, meaning you just did the thing wrong.

A distinction of sampling bias (albeit not a universally accepted one) is that it undermines the external validity of a test (the ability of its results to be generalized to the rest of the population), while selection bias mainly addresses internal validity for differences or similarities found in the sample at hand.

Examples:

1. In a case-control study, choosing a control population that is not representative of the population that produced the cases
2. The likelihood of being lost to follow-up is related to outcome or exposure status
3. Patients self-select to be part of the study, so refusal, non-response, or agreement to participate is related to exposure and/or disease
4. Differential referral or diagnosis of subjects from different groups within the study (e.g., cases defined using one technology; controls another)
5. HbA1c measured in the ED is higher (clinically and statistically) than HbA1c measured at home or in an outpatient clinic. Positive association between HbA1c and future myocardial infarction attenuated when controlling for location of measurement (Phelan et al).

Immortal time bias and length time bias are examples of selection biases.

Sampling bias is a form of selection bias.

Examples: Differential loss to follow up (differences between treatment and control groups)

17.3 Sampling Bias

Problem: study population differs from overall population; undermines external validity

ALSO KNOWN AS Ascertainment Bias

This messes up external validity of a study. The internal analysis may not be wrong as with selection bias. Sometimes people say this is a form of selection bias.

In Phelan et al, they discuss admixture bias, which occurs because the referral population for a particular institution is different than the local population. I would consider this a sampling bias, since you'd try to extrapolate findings to the local population without understanding that the referral population has biased the result. Also affects estimates of disease prevalence if, e.g., the facility is a referral facility for a particular disease/condition.

Systematic differences in the identification of individuals included in a study or distortion in the collection of data in a study.

Ascertainment bias arises when data for a study or an analysis are collected (or surveyed, screened, or recorded) such that some members of the target population are less likely to be included in the final results than others. The resulting study sample becomes biased, as it is systematically different from the target population.

Ascertainment bias can happen when there is more intense surveillance or screening for outcomes among exposed individuals than among unexposed individuals, or differential recording of outcomes.

Ascertainment bias can occur in screening, where take-up can be influenced by factors such as cultural differences. It can occur in case-control studies in the initial identification of cases and controls, which can be skewed by relevant exposures, leading to biased estimates of associations.

Examples: selection from a specific geographic area, self-selection when people get to choose whether to enter the study, pre-screening of trial participants or advertising for volunteers, healthy user bias, Berkson's fallacy

1. Many – perhaps most – psychology studies are done using college students as subjects.
2. Studies developed at university hospitals will be developed using individuals with generally higher income, younger age, and who are on average more white than the surrounding area. Results may not be applicable to a community hospital.

Solution is often stratification. Accounting for factors that influence a patient's pattern of encounters with a health system.

Question 17.4

How does the end user of an analysis (health system, patient, physicians, other scientists) impact one's concern for various kinds of bias?

17.4 Information Bias

> Information bias: Information bias is any systematic difference from the truth that arises in the collection, recall, recording and handling of information in a study, including how missing data is dealt with.

Information bias is probably more important than selection/ascertainment bias in EHR studies because its effects are more insidious; after all, one could make the argument that EHRs represent reasonably random samples of the surrounding area and documentation for different patients is unlikely to be influenced by features of the patient in a non-random way (potentially with the exception of insurance status).

17.4.1 Informed Presence/Absence Bias

Also known as: missing/enriched data

Examples:

1. Individuals from vulnerable populations, including those with low socioeconomic status, those with psychosocial issues, and immigrants, are more likely to visit multiple institutions or health care systems to receive care. Patients who use both VA and non-VA services have fewer measured comorbidities when only a single health system EHR was used in analyses (Phelan).

2. Patients with low socioeconomic status may receive fewer diagnostic tests and medications for for chronic diseases.

3. Patients with lower technological literacy/access may not be able to access patient portals or document self-reported outcomes.

4. Patients with diabetes have more recorded HbA1c measurements within their EHR records than patients without diabetes (Phelan).

Question 17.5

One of my pet peeves are machine learning models that predict a future diagnostic outcome using all available information: medications, labs, vitals, other diagnoses, procedures. Why is this practice problematic?

Kohane paper results: presence of lab tests, value of lab tests, time between successive tests all informative of mortality

17.4.2 Detection Bias

Systematic differences between groups in how outcomes are determined.

17.4.3 Misclassification Bias

Misclassification bias: When the probability of a measurement/misclassification error differs based on some other relevant property (e.g. age, gender, body weight, etc.).

1. Patients of low socioeconomic status may be more likely to be seen in teaching clinics, where data input or clinical reasoning may be less accurate. Implicit bias by healthcare practitioners could also lead to differences in how data are recorded and whether the results are accurate (e.g., the types of diagnoses considered as potential causes of a problem).

2. Women may be less likely to receive lipid-lowering medications and in-hospital procedures, as well as optimal care at discharge, compared with men, despite being more likely to present with hypertension and heart failure.

17.4.4 Recall Bias

Recall bias occurs when participants do not remember previous events or experiences accurately or omit details: the accuracy and volume of memories may be influenced by subsequent events and experiences. due to differences in accuracy or completeness of recall to memory of past events or experiences.

17.4.5 Reporting Bias

Reporting bias occurs when the dissemination of research findings is influenced by the nature and direction of the results, for instance in systematic reviews. Positive results is a commonly used term to describe a study finding that one intervention is better than another. A systematic distortion that arises from the selective disclosure or withholding of information by parties involved in the design, conduct, analysis, or dissemination of a study or research findings

17.5 Social Bias vs. Scientific Bias

Bias could mean "the model is wrong or does not generalize" or bias could mean "discrimination against one or more people/groups". In machine learning, bias means a model that is oversimplified and makes the same errors each time.

Remember: a model can only recapitulate patterns found in the underlying data.

All observational studies may be biased; however, bias in machine learning is particularly insidious because it can affect clinical decision support tools. Clinical studies that use EHR data are also at risk, but these may be discussed and results replicated elsewhere. The process is somewhat longer. There is a significant risk that health systems that are developing their own internal machine learning tools may incorporate bias into real-time healthcare decisions.

17.6 Availability Bias

A distortion that arises from the use of information which is most readily available, rather than that which is necessarily most representative. Think Trump: opinion is whichever one he heard last. Same goes for doctors; they're more likely to use a diagnosis that they already used recently.

All research questions and decisions, whether considering diagnostic accuracy of a test or effectiveness of an intervention, involve interpretation of data. Clinical decisions are based on data, which may be from routine care, published evidence, guidelines or clinician preference or experience.

1. Certain diseases, groups of people, and outcomes are easier to study than others because more data are available.
2. Is the EHR really the best instrument to use to ask a particular question? If we devote resources toward these studies, are we taking them away from study types that may be more accurate?