Chapter 5

Nonparametric statistics and model selection

In Chapter 2, we learned about the t-test and its variations. These were designed to compare sample means, and relied heavily on assumptions of normality. We were able to apply them to non-Gaussian populations by using the central limit theorem, but that only really works for the mean (since the central limit theorem holds for averages of samples). Sometimes, we're interested in computing other sample statistics and evaluating their distributions (remember that all statistics computed from samples are random variables, since they're functions of the random samples) so that we can obtain confidence intervals for them. In other situations, we may not be able to use the central limit theorem due to small sample sizes and/or unusual distributions.

In this chapter, we'll focus on techniques that don't require these assumptions. Such methods are usually called *nonparametric* or *distribution-free*. We'll first look at some statistical tests, then move to methods outside the testing framework.

■ 5.1 Estimating distributions and distribution-free tests

So far, we've only used "eyeballing" and visual inspection to see if distributions are similar. In this section, we'll look at more quantitative approaches to this problem. Despite this, don't forget that visual inspection is usually an excellent place to start!

We've seen that it's important to pay attention to the assumptions inherent in any test. The methods in this section make fewer assumptions and will help us test whether our assumptions are accurate.

■ 5.1.1 Kolmogorov-Smirnov test

The **Kolmogorov-Smirnov test** tests whether two arbitrary distributions are the same. It can be used to compare two empirical data distributions, or to compare one empirical data distribution to any reference distribution. It's based on comparing two cumulative distribution functions (CDFs).

Remember that the CDF of a random variable x is the probability that the random variable

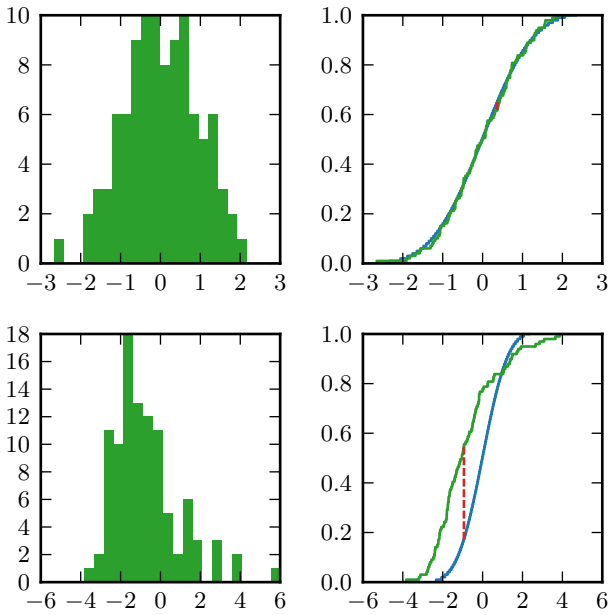


Figure 5.1: Two Kolmogorov-Smirnov test plots (right column) with histograms of the data being tested (left column). On the top row, the empirical CDF (green) matches the test CDF (blue) closely, and the largest difference (dotted vertical red line, near 0.5) is very small. On the bottom, the empirical CDF is quite different from the test CDF, and the largest difference is much larger.

is less than or equal to some value. To be a bit more precise, it's a function F such that $F(a) = P(x \leq a)$. When talking about data, it's often useful to look at empirical CDFs: $F_n(a) = \frac{1}{n}\sum_i \mathbb{I}(x_i < a)$[1] is the CDF of n observed data points.

Now suppose we want to compare two CDFs, F^1 and F^2. They might be empirical CDFs (to compare two different datasets and see whether they're significantly different) or one might be a reference CDF (to see whether a particular distribution is an appropriate choice for a dataset). The Kolmogorov-Smirnov test computes the statistic D_n:

$$D_n = \max_x |F_n^1(x) - F_n^2(x)|$$

This compares the two CDFs and looks at the point of maximum discrepancy; see Figure 5.1 for an example. We can theoretically show that if F^1 is the empirical distribution of x and F^2 is the true distribution x was drawn from, then $\lim_{n\to\infty} D_n = 0$. Similarly, if the two distributions have no overlap at all, the maximum difference will be 1 (when one CDF is 1 and the other is 0). Therefore, we can test distribution equality by comparing the statistic D_n to 0 (if D_n is significantly larger than 0 and close to 1, then we might conclude that the distributions are not equal).

Notice that this method is only defined for one-dimensional random variables: although there

[1] Remember that $\mathbb{I}$ is a function that returns 1 when its argument is true and 0 when its argument is false.

are extensions to multiple random variables; they are more complex than simply comparing joint CDFs.

Also notice that this test is sensitive to any differences at all in two distributions: two distributions with the same mean but significantly different shapes will produce a large value of D_n.

■ 5.1.2 Shapiro-Wilk test

The **Shapiro-Wilk test** tests whether a distribution is Gaussian using quantiles. It's a bit more specific than the Kolmogorov-Smirnov test, and as a result tends to be more powerful. We won't go into much detail on it in this class, but if you're interested, the Wikipedia page has more detail.

■ 5.1.3 Wilcoxon's signed-rank test

The two-sample t test we discussed in Chapter 2 requires us to use the central limit theorem to approximate the distribution of the sample mean as Gaussian. When we can't make this assumption (i.e., when the number of samples is small and/or the distributions are very skewed or have very high variance), we can use Wilcoxon's signed-rank test for matched pairs. This is a paired test that compares the medians of two distributions[2]. For example, when comparing the incomes of two different groups (especially groups that span the socioeconomic spectrum), the distributions will likely be highly variable and highly skewed. In such a case, it might be better to use a nonparametric test like Wilcoxon's signed-rank test. In contrast to the Kolmogorov-Smirnov test earlier, this test (like its unpaired cousin the Mann-Whitney U) is only sensitive to changes in the median, and not to changes in the shape.

The null hypothesis in this test states that the median difference between the pairs is zero. We'll compute a test statistic and a corresponding p-value, which give us a sense for how likely our data are under the null hypothesis. We compute this test statistic as follows:

(1) For each pair i, compute the difference, and keep its absolute value d_i and its sign S_i (where $S_i \in \{-1, 0, +1\}$). We'll exclude pairs with $S_i = 0$.

(2) Sort the absolute values d_i from smallest to largest, and rank them accordingly. Let R_i be the rank of pair i (for example, if the fifth pair had the third smallest absolute difference, then $R_5 = 3$).

(3) Compute the test statistic:

$$W = \left| \sum_i S_i R_i \right|$$

[2]For unmatched pairs, we can use the Mann-Whitney U test, described in the next section

W has a known distribution. In fact, if N is greater than about 10, it's approximately normally distributed (if not, it still has a known form). So, we can evaluate the probability of observing it under a null hypothesis and thereby obtain a significance level.

Intuitively, if the median difference is 0, then half the signs should be positive and half should be negative, and the signs shouldn't be related to the ranks. If the median difference is nonzero, W will be large (the sum will produce a large negative value or a large positive value). Notice that once we constructed the rankings and defined R_i, we never used the actual differences!

■ 5.1.4 Mann-Whitney U Test

The Mann-Whitney U test is similar to the Wilcoxon test, but can be used to compare multiple samples that aren't necessarily paired. The null hypothesis for this test is that the two groups have the same distribution, while the alternative hypothesis is that one group has larger (or smaller) values than the other[3]. Computing the test statistic is simple:

(1) Combine all data points and rank them (largest to smallest or smallest to largest).

(2) Add up the ranks for data points in the first group; call this R_1. Find the number of points in the group; call it n_1. Compute $U_1 = R_1 - n_1(n_1+1)/2$. Compute U_2 similarly for the second group.

(3) The test statistic is defined as $U = \min(U_1, U_2)$.

As with W from the Wilcoxon test, U has a known distribution. If n_1 and n_2 are reasonably large, it's approximately normally distributed with mean $n_1 n_2/2$ under the null hypothesis.

If the two medians are very different, U will be close to 0, and if they're similar, U will be close to $n_1 n_2/2$. Intuitively, here's why:

- If the values in the first sample were all bigger than the values in the second sample, then $R_1 = n_1(n_1+1)/2$[4]: this is the smallest possible value for R_1. U_1 would then be 0.
- If the ranks between the two groups aren't very different, then U_1 will be close to U_2. With a little algebra, you can show that the sum $U_1 + U_2$ will always be $n_1 n_2$. If they're both about the same, then they'll both be near half this value, or $n_1 n_2/2$.

[3] Under some reasonable assumptions about the distributions of the data (see the Mann-Whitney U article on Wikipedia for more details), this test can be used with a null hypothesis of equal medians and a corresponding alternative hypothesis of a significant difference in medians.

[4] $R_1 = n_1(n_1+1)/2$ because in this case, the ranks for all the values from the first dataset would be 1 through n_1, and the sum of these values is $n_1(n_1+1)/2$.

5.2 Resampling-based methods

All the approaches we've described involve computing a test statistic from data and measuring how unlikely our data are based on the distribution of that statistic. If we don't know enough about the distribution of our test statistic, we can use the data to tell us about the distribution: this is exactly what resampling-based methods do. **Permutation tests** "sample" different relabelings of the data in order to give us a sense for how significant the true relabeling's result is. **Bootstrap** creates "new" datasets by resampling several times from the data itself, and treats those as separate samples. The next example illustrates a real-world example where these methods are useful.

Example: Chicago teaching scandal

In 2002, economists Steven Levitt and Brian Jacob investigated cheating in Chicago public schools, but not in the way you might think: they decided to investigate cheating by teachers, usually by changing student answers after the students had taken standardized tests[a]

So, how'd they do it? Using statistics! They went through test scores from thousands of classrooms in Chicago schools, and for each classroom, computed two measures:

(1) How unexpected is that classroom's performance? This was computed by looking at every student's performance the year before and the year after. If many students had an unusually high score one year that wasn't sustained the following year, then cheating was likely.

(2) How suspicious are the answer sheets? This was computed by looking at how similar the A-B-C-D patterns on different students' answer sheets were.

Unfortunately, computing measures like performance and answer sheet similarity is tricky, and results in quantities that don't have well-defined distributions! As a result, it isn't easy to determine a null distribution for these quantities, but we still want to evaluate how unexpected or suspicious they are.

To solve this problem, Levitt and Jacob use two nonparametric methods to determine appropriate null distributions as a way of justifying these measures. In particular:

- They assume (reasonably) that most classrooms have teachers who don't cheat, so by looking at the 50th to 75th percentiles of both measures above, they can obtain a null distribution for the correlation between the two.
- In order to test whether the effects they observed are because of cheating teachers, they randomly re-assign all the students to new, hypothetical classrooms and repeat their analysis. As a type of *permutation test*, this allows them to establish a baseline level for these measures by which they can evaluate the values they observed.

While neither of these methods is exactly like what we'll discuss here, they're both examples of a key idea in nonparametric statistics: using the data to generate a null hypothesis rather than assuming any kind of distribution.

What'd they find? 3.4% of classrooms had teachers who cheated on at least one standardized test when the two measures above were thresholded at the 95th percentile. They also used regression with a variety of classroom demographics to determine that academically poorer classrooms were more likely to have cheating teachers, and that policies that put more weight on test scores correlated with increased teacher cheating.

[a] see Jacob and Levitt. Rotten Apples: An Investigation of the Prevalence and Predictors of Teacher Cheating. For more economic statistics, see Steven Levitt's book with Stephen Dubner, Freakonomics.

5.2.1 Permutation tests

Suppose we want to see if a particular complex statistic is significantly different between two groups. If we don't know the distribution of the statistic, then we can't assign any particular probabilities to particular values, and so we can't say anything interesting after computing a statistic from data. The idea behind **permutation tests** is to use the data to generate the null hypothesis.

The key idea is that if there weren't a significant difference between this statistic across the two groups in our data, then the statistic should have the same value for any other two arbitrarily selected "groups" of the same sizes. We'll make up a bunch of fake "groups" and see if this is the case.

We'll demonstrate the idea with comparing sample means, but note that in practice sample means could be compared using a t-test, and that permutation tests are generally used for more complicated statistics whose distribution can't be deduced or approximated.

Let's call the two groups in our data A and B. Our test statistic of interest will be $w^* = \bar{x}_A - \bar{x}_B$: the difference in the means of the two groups. Our null hypothesis will be $H_0 : \bar{x}_A = \bar{x}_B$. As before, we want to see whether this difference is significantly large or small. In order to get a sense for the null distribution of w, we can try relabeling the points many different times and recomputing the statistic. Suppose the original set A has n data points, and B has m points. We'll randomly pick n points out of all $n+m$, assign those to our new "group" A_i, and assign the others to our new "group" B_i (where the subscript i indicates the iteration). We'll recompute w_i based on these new groups, and repeat the process. We can then see whether our value w^* is unusual based on the collection of w_i, perhaps by looking at what fraction of them are more extreme than w^*.

We could either repeat this for every possible labeling (usually called an *exact test*), or randomly choose a subset and evaluate only on those (usually called a *Monte Carlo approximation*). So, this entire procedure requires only a procedure for computing the statistic of interest (in this case, w): the rest is determined by the data.

We can apply permutations elsewhere, too! For example, if we suspect there is some kind of trend in time series data, we can permute the time indices and compare the correlation. For example, if we observe the time series data $(8, 12, 15.8, 20.3)$ and suspect there's a trend, we can permute the order, obtaining permutations like $(15.8, 12, 20.3, 8)$ or $(20.3, 8, 15.8, 12)$, fit linear models to each one, and evaluate how unlikely the observed fit is.

5.2.2 Bootstrap

Suppose we have some complicated statistic y that we computed from our data x. If we want to provide a confidence interval for this statistic, we need to know its variance. When our statistic was simply $\bar{x}$, we could compute the statistic's standard deviation (i.e., the standard error of that statistic) from our estimated standard deviation using $s_x/\sqrt{n}$. But for more complicated statistics, where we don't know the distributions, how do we provide a confidence interval?

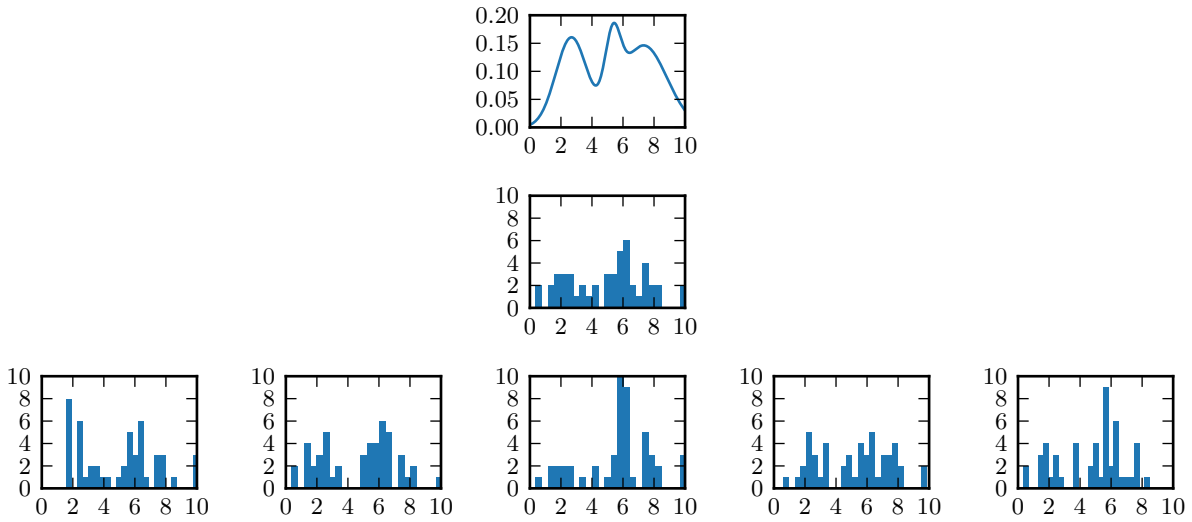


Figure 5.2: An illustration of bootstrap sampling. The top figure shows the true distribution that our data points are drawn from, and the second figure shows a histogram of the particular data points we observed ($N = 50$). The bottom row shows various bootstrap resamplings of our data (with $n = N = 50$). Even though they were obtained from our data, they can be thought of as samples from the true distribution (top).

One approach is a method called **bootstrap**. The key idea here is that we can resample points from our data, compute a statistic, and repeat several times to look at the variance across different resamplings.

Recall that our original data (N points) are randomly generated from some true distribution. If we randomly sample n points ($n \leq N$, and often $n = N$) from our data *with replacement*[5], these points will also be random samples from our true distribution, as shown in Figure 5.2. So, we can compute our statistic over this smaller random sample and repeat many times, measuring the variance of the statistic across the different sample runs.

Everything we've talked about so far has been based on the idea of trying to approximating the true distribution of observed data with samples. Bootstrap takes this a step further and samples from the samples to generate more data.

A similar method, known as **jackknife**, applies a similar process, but looks at $N - 1$ points taken without replacement each time instead of n with replacement. Put more simply, we remove one point at a time and test the model. Notice that we've seen a similar idea before: our initial definition of Cook's distance was based on the idea of removing one point at a time. In practice, boostrap is more widely used than jackknife; jackknife also has very different theoretical properties.

[5]This means that a single data point can be sampled more than once.

5.3 Model selection

When fitting models (such as regression) to real data, we'll often have a choice to make for model complexity: a more complex model might fit the data better but be harder to interpret, while a simpler model might be more interpretable but produce a larger error. We've seen this before when looking at polynomial regression models and LASSO in Chapters 3 and 4.

In this section, we'll learn how to pick the "best" model for some data among several choices.

5.3.1 Minimum generalization error

First, we'll define "best" by how well our model *generalizes* to new data that we'll see in the future. In particular, we'll define it this way to avoid *overfitting*.

How can we figure out how well our model generalizes? The most common way is to divide our data into three parts:

- The **training set** is what we'll use to *fit the model* for each possible value of the manually-set parameters,
- the **validation set** is what we'll use to *determine parameters* that require manual settings,
- and the **test set** is what we use to *evaluate our results* for reporting, and get a sense for how well our model will do on new data in the real world.

It's critically important to properly separate the test set from the training and validation sets! At this point you may be wondering: why do we need separate test and validation sets? The answer is that we *choose a model based on its validation set performance*: if we really want to see *generalization error*, we need to see how it does on some new data, not data that we used to pick it.

A good analogy is to think of model fitting and parameter determination as a student learning and taking a practice exam respectively, and model evaluation as that student taking an actual exam. Using the test data in any way during the training or validation process is like giving the student an early copy of the exam: it's cheating!

Figure 5.3 illustrates a general trend we usually see in this setup: as we increase model complexity, the training error (i.e. the error of the model on the training set) will go down, while the testing error will hit a "sweet spot" and then start increasing due to overfitting.

For example, if we're using LASSO (linear regression with sum-of-absolute-value penalty) as described in Chapter4, we need to choose our regularization parameter λ. Recall that λ controls model sparsity/complexity: small values of λ lead to complex models, while large values lead to simpler models. One approach is:

(a) Choose several possibilities for λ and, for each one, compute coefficients using the training set.

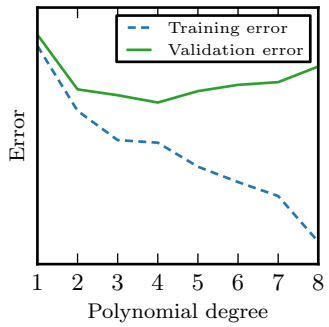


Figure 5.3: Training and validation error from fitting a polynomial to data. The data were generated from a fourth-order polynomial. The validation error is smallest at this level, while the training error continues to decrease as more complex models overfit the training data.

(b) Then, look at how well each one does on the validation set. Separating training and validation helps guard against overfitting: if a model is overfit to the training data, then it probably won't do very well on the validation data.

(c) Once we've determined the best value for λ (i.e., the one that achieves minimum error in step (b)), we can fit the model on all the training and validation data, and then see how well it does on the test data. The test data, which the model/parameters have *never seen before*, should give a measure of how well the model will do on arbitrary new data that it sees.

The procedure described above completely separates our fitting and evaluation processes, but it does so at the cost of preventing us from using much of the data. Recall from last week that using more data for training typically decreases the variance of our estimates, and helps us get more accurate results. We also need to have enough data for validation, since using too little will leave us vulnerable to overfitting.

One widely used solution that lets us use more data for training is **cross-validation**. Here's how it works:

(1) First, divide the *non-test* data into K uniformly sized blocks, often referred to as *folds*. This gives us K training-validation pairs: in each pair, the training set consists of $K-1$ blocks, and the validation set is the remaining block.

(2) For each training/validation pair, repeat steps (a) and (b) above: this gives us K different errors for each value of λ. We can average these together to get an average error for each λ, which we'll then use to select a model.

(3) Repeat step (c) above to obtain the test error as an evaluation of the model.

Although these examples were described with respect to LASSO and the parameter λ, the procedures are much more general: we could have easily replaced "value for λ" above with a different measure of model complexity.

Also note that we could use a bootstrap-like approach here too: instead of deterministically dividing our dataset into K parts, we could have randomly subsampled the non-test data K different times and applied the same procedure.

■ 5.3.2 Penalizing complexity

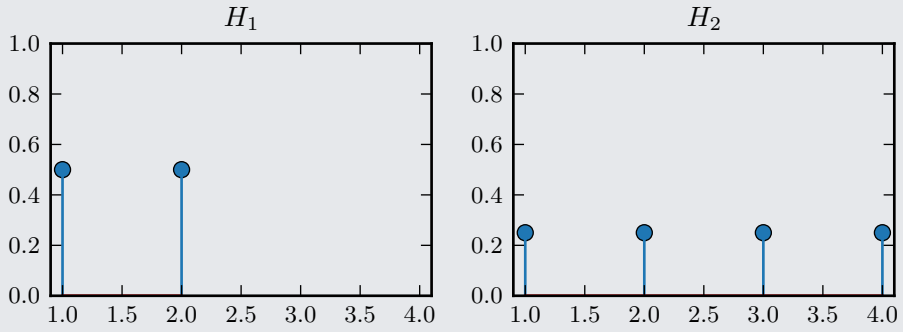
Another way to address the tradeoff between model complexity and fit error is to directly penalize model complexity. That is, we can express the "badness" of a model as the sum of an error term (recall the sum of squared error cost that we used in linear regression) and some penalty term that increases as the model becomes more complex. Here, the penalty term effectively serves as a proxy for minimizing generalization error. Two common penalty terms are **AIC**, or Akaike Information Criterion, which adds a penalty term equal to $2p$ (where p is the number of parameters), and **BIC**, or Bayesian Information Criterion, which adds a penalty term equal to $p \ln n$ (where n is the number of data points).

BIC is closely related to another criterion known as **MDL**, or minimum description length. This criterion tries to compress the model and the error: if the model and the error are both small and simple, they'll be highly compressible. A complex model that produces low error will be hard to compress, and an overly simplistic model will produce high error that will be hard to compress. This criterion tries to achieve a balance between the two.

Example: Bayesian Occam's Razor

Another idea commonly used is **Bayesian Occam's Razor**. In Bayesian models, simpler models tend to have higher probability of observing data.

Suppose we observe a number and want to decide which of two models generate it. The simpler model, H_1, says that it's equally likely to be 1 or 2, while the more complex model, H_2, says that it's equally likely to be 1, 2, 3, or 4. The following figure shows the two distributions:



If we observe the number 2, the likelihood of this observation under H_1 is 0.5, while the likelihood under H_2 is 0.25. Therefore, by choosing the model with the highest probability placed on our observation, we're also choosing the simpler model.

Intuitively, the more values that a model allows for, the more it has to spread out its probability for each value.