Being conservative

When the exact right answer cannot be reliably determined from a dataset, you need to make a choice about what types of errors are most harmful. You often want to be conservative in some sense, but that can take different forms. In particular:

- You might want to declare that an effect exists only if there is strong evidence for it.
- You may want to allow for its existence even if the evidence is not very strong.

In at least one very common domain, there are fundamental tradeoffs at play so that you cannot do both. Today, we will get into some of the details and see a version of this tradeoff in action.

Learning objectives

- Describe linear regression.
- Fit linear regression models in R. Use them to extract predictions and control the rate of false positives.
- Describe two model selection criteria – hypothesis testing and cross-validation. Explain the differences between them in terms of end goals and in terms of typical outcomes. Implement them in R to demonstrate the differences in their results.

Linear regression

In order to have enough context to continue on this topic, I need to take a detour and discuss *linear regression*. You saw linear regression in the session on outliers. We used it to try to capture the line in this plot, and we tried to modify it to ignore outliers.

Linear regression just means fitting a line to a set of data on an X-Y scatterplot. If $Y \approx \beta_0 + \beta_1 X$, the task is to choose the slope β_1 and the y-intercept β_0.

Today we will carry out linear regression use a built-in function in R called `lm`. `lm` chooses coefficients by minimizing the sum of squared errors:

$$(Y_1 - \beta_0 + \beta_1 X_1)^2 + ... + (Y_N - \beta_0 + \beta_1 X_N)^2$$

. This is the sum of the squares of the lengths of the *vertical green lines* in the picture.

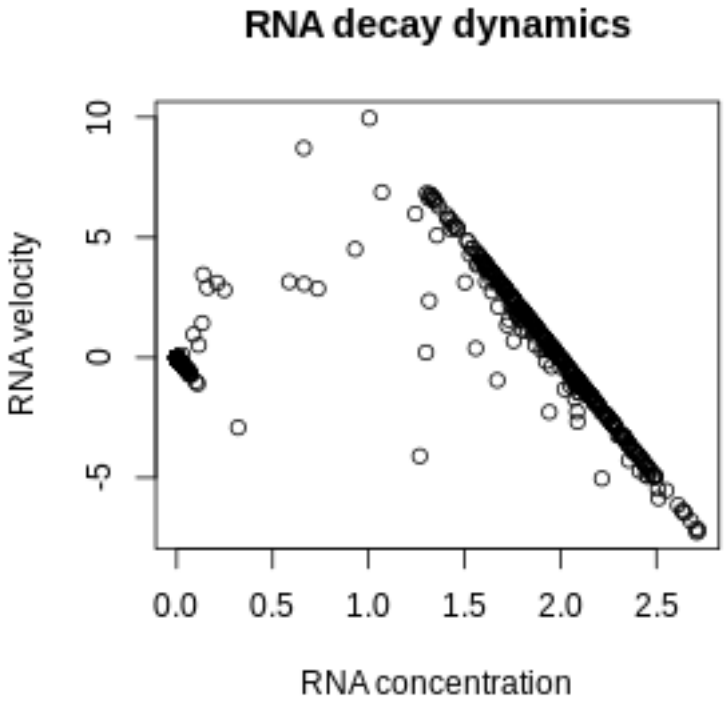

Figure 1: The chart showing RNA levels (x axis) versus RNA velocity (y axis) for gene 7 from the outliers session.

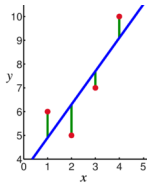

Figure 2: From the Wikipedia article on linear regression. Public domain; by Oleg Alexandrov.

I anticipate that this formula will raise some questions, but the main point of this session has not yet arrived, so let me anticipate two questions and discuss only briefly before continuing. First, it's great that `lm` can minimize the sum of squared errors, but how? For better or worse, this class is about not "How?" but "Why?", so I decline to discuss this. One good way to learn how `lm` works is to take a class in linear algebra, and towards the end, ask about the QR matrix decomposition.

Second, why use the sum of squared errors, and not some other "goodness of fit" measure? We are using it for the class because it is one of the simplest things to do, both in terms of mathematical analysis and algorithmic implementation. But much of the time, the sum of squared errors is not the best thing to minimize – for example, the sum of squares will be very sensitive to outliers. There are many other options, and to learn about them, you could look for classes in regression methods, generalized linear models, or supervised machine learning. One very useful mathematical finding is the Gauss-Markov Theorem. It describes limited circumstances when minimizing the sum of squares is the best possible choice.

Variable selection

Linear regression is often done using more than one X value for each observation. In those cases, it is harder to picture because the XY scatterplot only fits one variable on the X axis, but the formula is easy to write:

$$Y_i \approx \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \beta_3 X_{i3}$$

.

Today, we ask: which parts of X should we use and which should we ignore? If the estimated value for β_2 is small, should we set it to 0 and continue as if X_{i2} has no association with Y_i? This problem is called "variable selection" or "subset selection".

Countless papers, several books, and probably hundreds of computer programs have been written on this question, and different methods get different answers. Sometimes, one method is just better than another, but often, the difference is not about performance but rather priorities. In particular, these methods may prioritize either *prediction* or *inference*.

- Inference means we want to learn something about the internal mechanisms of the system under study.
- Prediction means we don't care if the internals are right or wrong: if someone gives us a new value of X, we just want to guess Y accurately.

Algorithms or decision rules that are optimized for one of these goals will often fall short of the other goal. In some cases, it is guaranteed that no algorithm can

perform optimally at both prediction and inference. Here is one very technical way of mathematically proving that the prediction-inference tradeoff cannot be avoided. Today, we will demonstrate a similar tradeoff through simulation in R.

Simulated data

Today's demo centers around a dataset generated like this:

- X_1 contains the numbers from 1 to 10, with each number occuring 5 times (total length: 50). They are in a random order.
- X_2 and X_3 are generated the same way as X_1, but the order is different (all three are statistically independent).
- ϵ is distributed as $Binomial(0.5, 10)$.
- $Y_i = 1.1 * X_{1,i} + 0.1 * X_{2,i} + 0 * X_{3,i} + \epsilon_i$.

So, in truth, Y depends on X_1 and X_2, but not X_3. We will act as if we don't know which of the X's is relevant, and we will consider models that include different combinations of X's.

Variable selection methods

We'll study three methods for variable selection.

Error after fitting

This is a simple method.

- Fit some models to the dataset.
- For each model, make predictions (we'll call them $\hat{Y}$).
- Measure the sum of squared errors $(Y_1 - \hat{Y}_1)^2 + ... + (Y_{50} - \hat{Y}_{50})^2$.
- Select the model with the lowest error.

Leave-one-out cross-validation

This works much like predictive error. You can follow almost the same steps. The key difference is to *fit* the model and *evaluate* the model on separate subsets of data. You'll fit each model on 49 data points and evaluate on the 50th. You'll repeat this for all 50 data points and average the results. This is demonstrated in the code.

Hypothesis testing

This method is completely separate from cross-validation: different history, different priorities, different mathematical basis. The goal here is to control the *false positive rate*: "If $\beta_3 = 0$, then I want less than a 10% chance of including it in my model."

To achieve this, we need the concept of a *sampling distribution*. Estimates of the β's take a simple form, depending on quantities we have already described. If the estimate of β_3 is called $\hat{\beta}_3$, there is a function f such that

$$\hat{\beta}_3 = f(X_1, X_2, X_3, Y, \epsilon, \beta)$$

. If you take typical college classes in linear algebra and probabilty & statistics, you can learn what f is and where f comes from. Since this class is about "why", not "how", let's assume someone else has done the math and figure out how to use it.

Functions of random variables are also random variables, so $\hat{\beta}_3$ is itself a random variable. It follows some mathematical law, which we call its *sampling distribution*. It has an expectation, a variance, and a probability mass function. All of them can be computed and evaluated (at least approximately) via existing software.

This software provides a very simple procedure. It will output a number called a "p-value". If you want to control the false positive rate at 10% for a given decision, then select a variable whenever its p-value is below 0.1. P-values are discussed a little more in a footnote.

Exercises

After doing the exercises below, send your results and your R code by email as `LAST_FIRST_regression.docx` and `LAST_FIRST_regression.R`.

Starting from the linear regression demo:

1. Compare these three models:
   - $Y \approx \beta_1 * X_1$,
   - $Y \approx \beta_1 * X_1 + \beta_2 * X_2$, and
   - $Y \approx \beta_1 * X_1 + \beta_2 * X_2 + \beta_3 * X_3$.

   Compare them using the error after fitting and the error from leave-one-out cross-validation. Generate 1,000 different datasets. What percent of the time is X_2 included in the model? X_3?
2. Fit the model $Y \approx \beta_1 * X_1 + \beta_2 * X_2 + \beta_3 * X_3$. Test the null hypotheses $\beta_2 = 0$ and $\beta_3 = 0$, controlling the false positive rate at 10%. Repeat 1,000 times.

Report your results in a table like this. (These numbers are made-up.)

Method	includes X_2	includes X_3
Fitting error	80%	10%
LOOCV	60%	20%
Hypothesis tests	40%	30%

Footnotes

You don't have to read these. But you can if you want.

P-values

A p-value is a probability. The event that this probability describes is a little complicated. Suppose you do an analysis and observe an estimate $\hat{\beta}_3$. Suppose you were to re-do the entire study, obtaining a new dataset in the exact same way as the first. The p-value associated with $\hat{\beta}_3$ is the probability of observing a result more extreme than $\hat{\beta}_3$ upon redoing the study. This probability is defined and calculated only under a strong assumption called a "null hypothesis", to be explained below.

"More extreme" can be measured in different ways, and when you report a p-value, you need to explain how you measured extremeness. If you hear people talking about "one-tailed" and "two-tailed" tests, they are measuring extremeness in different ways. If all else remains the same, the p-value is always decreasing with the measure of extremeness.

P-values assume a certain "null hypothesis". The same probability calculations cannot apply both when the true value β_3 is 0 and when β_3 is 10,000, and in this lecture, the null hypothesis is that the coefficient being tested equals 0.

In probability and statistics, we are accustomed to probabilities describing random variables, but being fixed themselves. For a Binomial random variable with success probability 0.5, the outcome is random, but the 0.5 is fixed. P-values break this dichotomy: they are stated as probabilities, but they are also outcomes of a random experiment. This can be very confusing. It may be easier at times to retreat from p-values and consider only sampling distributions of more tangible quantities, like β_3. Often, you can still get the job done without using p-values at all.

Prediction and inference within a single research problem

The "prediction versus inference" debate can become more complicated when you use a predictive model as a component of a downstream inference method. You might want to test whether the gene ZBTB4 contributes to younger age at

onset of Alzheimer's disease. To reduce randomness in age at onset, you might first subtract out variation due to known causes, like biological sex or the gene APOE4. For testing ZBTB4, the objective is inference, but when modeling known causes, you do not need inferences. You need the best possible predictions. Sometimes you need to add complexity to a *predictive* model in order to achieve the best possible *inferences* further downstream. (This example is from a 2018 paper I contributed to in the lab of Elizabeth Blue at the University of Washington.)