Simulated Data Regressions

Signal Data Science

We'll work with simulated data to explore some of the theory behind linear regressions.

The normal distribution

Read the indicated answers to these Quora questions:

- Why do we use the normal distribution? by Paul King, Ralph Winters and Peter Flom.
- How would you explain the Gaussian distribution in layman's terms? by Breno Sakaguti.

Then go through Chapter 3: The Normal Distribution of the book Tutorial for the integration of the software, R, with introductory statistics. (*Note:* If these links are broken, skip them and move on.)

Normal distributions and linear regression

The theory of linear regression applies when the variables involved are normally distributed. We'll be exploring this in the special case of two variables y and x that are normally distributed with mean 0 and standard deviation 1.[1]

Suppose that $y = ax + \text{error}$, where:

- x is normally distributed with mean 0 and standard deviation 1, and
- error is normally distributed with mean 0 and standard deviation b such that $a^2 + b^2 = 1$.[2]

[1]Any normally distributed variable can be converted to such a variable using the `scale()` function in R.

[2]If distributions A and B have variances σ_A^2 and σ_B^2, then their sum has variance $\sigma_A^2 + \sigma_B^2$ and therefore standard deviation $\sqrt{\sigma_A^2 + \sigma_B^2}$. (That is to say, *variances are additive*.) Therefore under the specified conditions y also has mean 0 and standard deviation 1.

Then a is the *correlation* between x and y. The *percent variance in y explained by x* is just a^2. This is also referred to as R^2.

Regressions with simulated data

In this exercise we'll explore the approximations to a that come from applying ordinary least squares regression to a finite sample of data. In the material below, try $a = 0.1, 0.2, 0.3, \ldots, 0.9$ and sample size $n = 100, 500, 2500, 10000$.

- Write a function `getSamples(a, n)` that takes a value of `a` and a sample size `n`, returning a dataframe with two columns:
  - `x`, obtained using `rnorm()`, and
  - `y` as defined above.
- Use `ggplot()` to make a scatter plot of `y` against `x` for various values of `a` and `n` to get intuition for what linear relationships between two normally distributed variables looks like. Graph a linear best fit line along with the points using `geom_smooth()` and the right choice for the `method` parameter.

The distributions of slope estimates

- Write a function `estimateSlopes(a, n, numTrials = 500)` which returns an array of estimates of `a` for each of `numTrials` batches with sample size `n`. You'll want to call `coef()` on the output of `lm()`.
- Using `geom_histogram()`, make histograms of the output of `estimateSlopes()` for some values of `a` and `n`. Based on your reading of the Quora answers above, speculate on why the values might be normally distributed.
- Make a dataframe `dfSD` with rows corresponding to values of `n` and columns corresponding to values of `a`. You may find `rownames()` helpful for this. Fill the entries with the standard deviations of the outputs of `estimateSlopes()`. Do the answers depend on the value of `a`?
- Let `a = 0.1`. Determine how the standard deviations of the outputs of `estimateSlopes()` vary with `n`. You'll likely find it useful to make a dataframe `dfSD2` with a larger range of values of `n`, and plot the entries as a function of `n`.

p-values

- Modify `estimateSlopes()` to make a function `estimateSlopesWithPVals()` to return a dataframe with estimated slopes and p-values associated with the slopes being nonzero. You'll have to figure out how to extract the latter from the `lm()` object in R.
- Call the function with `a = 0.1, n = 500, numTrials = 10000`. Plot the slopes and the p-values. Compare the median p-value with the fraction of slopes that are less than or equal to zero.