Stat 340 – Updating a discrete prior in R

Today, we talked about how to use a discrete prior distribution to draw inference about a binomial proportion. In this handout, I'll discuss how to do this in R two ways.

The textbook's way

Your textbook author's created an R package, `ProbBayes`, to make some things easier. For example, they created the `bayesian_crank()` function to calculate the posterior probability table given the prior probabilities and the likelihood.

To use the `bayesian_crank()` you first need to create a data frame with columns for the prior support, the prior probabilities, and the likelihood. The below code chunk does this for the blindsight example.

```
library(dplyr)
bayes_table <- data.frame(
  p = seq(0.1, 0.9, by = .1),
  Prior = 1 / 9  #R recycles automatically
) %>%
  mutate(Likelihood = dbinom(14, size = 17, prob = p))
```

Notes:

- In the above code, I use `dplyr::mutate()` to add a new column to the data set.
- The capitalization of `Prior` and `Likelihood` is (annoyingly) necessary.

Once we have this table, we can pass it to the `bayesian_crank()`:

```
posterior_table <- bayesian_crank(bayes_table)
```

The `tidyverse` way

If you took data science or have used `dplyr` to wrangle data before, then you don't need to use the `bayesian_crank()` function. Instead, you can add a bit more wrangling to the `mutate()` step:

```
library(dplyr)
blindsight <-
  data.frame(
    p =  seq(0.1, 0.9, by = .1),
    prior = 1 / 9) %>%
  mutate(
    likelihood =  dbinom(14, size = 17, prob = p),  # binomial likelihood
    unstd.prior = likelihood * prior,               # posterior plausibility
    posterior = unstd.prior / sum(unstd.prior)      # posterior probability
  )
```