Introduction

(Mostly based on McElreath's book)

McElreath has this notion of *golem*, essentially a mechanical device that for a given input produces some statistical test output. In our world these are the *models* we work with.

For example, if I have two sample populations with mean x_i and x_j, st dev s_i, s_j, then my golem might tell me to use a t-test to compare the means. And importantly (hence the golem metaphor) without any awareness of what the inputs are, and especially with no idea if that is the right golem to use for the problem. Thus we should be very careful when relying only on statistical tests, as their results may not be accurate.

(Ch 1)

Bayesian Data Analysis

> Count all the ways data can happen under the assumptions. Assumptions with more ways the data can happen are more plausible.

Model **plausibility**: instead of falsifying some (often arbitrary and unrealistic) null hypothesis model (e.g., programming skill has NO EFFECT on task time), compare credible models (or hypotheses). Which one best describes the data?

- Watch for overfitting - if we train on the data, the model knows the **sample** really well, but not the broader **world** (which is what we care about!)
- Use information theory (AIC, cross-validation) to compare

Marbles example - see text chapter 2

Bayes as marbles

> A conjectured proportion of blue marbles, *p*, is usually called a **PARAMETER** value. It's just a way of indexing possible explanations of the data.

> The relative number of ways that a value *p* can produce the data is usually called a **LIKELIHOOD**. It is derived by enumerating all the possible data sequences that could have happened and then eliminating those sequences inconsistent with the data.

> The prior plausibility of any specific *p* is usually called the **PRIOR PROBABILITY**.

> The new, updated plausibility of any specific *p* is usually called the **POSTERIOR PROBABILITY**.

Than we can use Bayes's law to say `Posterior prob is proportional to Prob of data X Prior`

Bayesian inference: the ICSE example.

What is happening in the code (see Github).

A Bayesian Workflow

See Fig 1 in https://arxiv.org/pdf/2011.01808.pdf

1. Pick a model
2. Check the model and prior against domain knowledge
3. Fit the model (e.g. call `ulam()` or `brms()` or similar)
4. Validate the computation/sampling: convergence diagnostics, r-hat
5. Fix the computation - rewrite the model
6. Evaluate and Use Model (predict, cross-validate, posterior predictive checks)
7. Modify the model - new priors, more data, new predictors
8. Compare models - leave-one-out information criterion, model averaging

Building Models

Note: we can write linear regression equations as either $y = mx + b$ or as the model variant of that, $y \sim Normal(\mu, \sigma)$ (And possibly some error term ϵ)

From Chapter 4.2 (a language for describing models):

1. What measurements (variables) are we hoping to predict/understand? (y, here)
2. For each variable, what is a likelihood that describes the plausibility of the observations? (Here, Normal())
3. What predictor variables will help us find the outcomes?
4. Use these predictors to define the shape and location of the likelihood (here, μ, σ)
5. Choose priors to define our *initial* information state for all model parameters. (Where should the mean be? What should sigma be?)

Statistical Model Specification in R and Stan

We now can specify our statistical model using the tilde notation. For variables W (for waistline), D (for Donuts), we can say W is influenced by / predicated by / modelled by D, i.e., donut consumption influences waist line. We write this as:

$$W \sim D$$

A full model formula will include the priors for the parameters and look more like

$$\begin{aligned}
\mathrm{L}_i &\sim \mathrm{Binomial}(n_i, p_i) \\
\mathrm{logit}(p_i) &= \alpha_{\mathrm{SUBJECT}[i]} + (\beta_P + \beta_{PC} C_i) P_i \\
\alpha_{\mathrm{SUBJECT}} &\sim \mathrm{Normal}(0, 10) \\
\beta_P &\sim \mathrm{Normal}(0, 10) \\
\beta_{PC} &\sim \mathrm{Normal}(0, 10)
\end{aligned}$$

Once we have this we translate it into Stan's DSL (see 4.3):

```
flist = alist(
        height ~ dnorm(mu,sigma),
        mu ~ dnorm(178,20),
  sigma ~ dunif(0,40)
)
```

and then model it using Stan