Introduction to Basic Statistical Models

John Tipton

Readings

- R for data science
  - Introduction
  - Chapters 18 (Model basics with `modelr`), 19 (Model building), and 20 (Building many models with `purrr` and `broom`)

Modeling

- Data analysis goes:
    - explore the data
    - visualize the data
    - construct models
    - repeat
- Basic analysis framework
    - train: set aside 60% of the data for model development
    - test: set aside 20% of the data for repeated testing of model performance
    - validate: set aside 20% of the data for the final validation (never use this data until the very end of the analysis)

Modeling

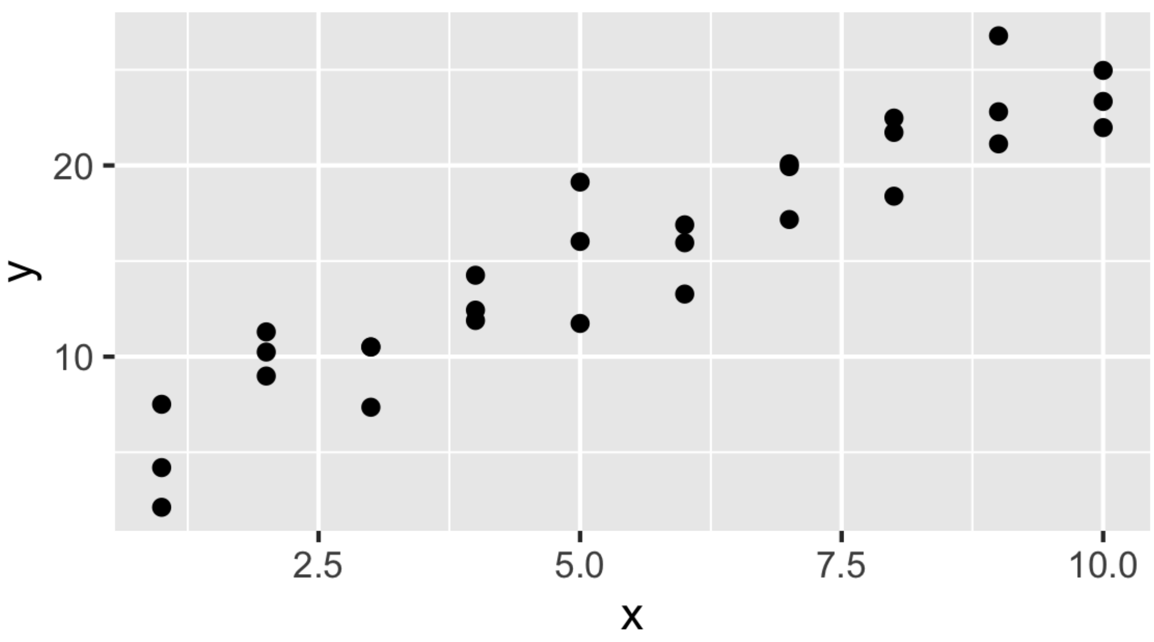
> George Box: All models are wrong, but some are useful

```
library(tidyverse)
library(modelr)
set.seed(111)
```

Modeling

- Use the `sim1` dataset from `modelr`

```
sim1 %>%
  ggplot(aes(x, y)) +
  geom_point()
```



Modeling

- Many different models with different slopes and intercepts

```
models <- tibble(
  intercept = runif(50, 0, 8),
  slope     = runif(50, 1, 3)
)

ggplot(sim1, aes(x, y)) +
  geom_abline(aes(intercept = intercept, slope = slope), data = models, alpha = 1/4) +
  geom_point()
```

Modeling

- How do we choose between models?
- Idea: predict the data for each set of coefficients and choose the "best" model

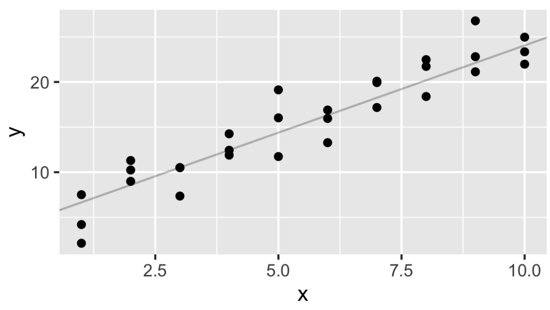
```
model_predictions <- function(intercept, slope, data) {
  intercept + data$x * slope
}

models <- models %>%
  mutate(preds = map2(intercept, slope, model_predictions, data = sim1))
```

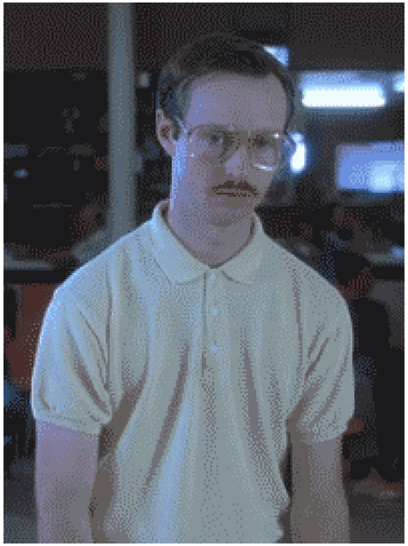
```
sum_of_squared_deviations <- function(intercept, slope, data) {
  preds <- model_predictions(intercept, slope, data)
  sum((preds - data$y)^2)
}
models <- models %>%
  mutate(deviations = map2_dbl(intercept, slope, sum_of_squared_deviations, data = sim1)

best_idx <- which.min(abs(models$deviations))
```

Modeling

- Alright!

```
ggplot(sim1, aes(x, y)) +
  geom_abline(
    data = models[best_idx, ],
    aes(intercept = intercept, slope = slope),
    alpha = 1/4
  ) +
  geom_point()
```





Modeling

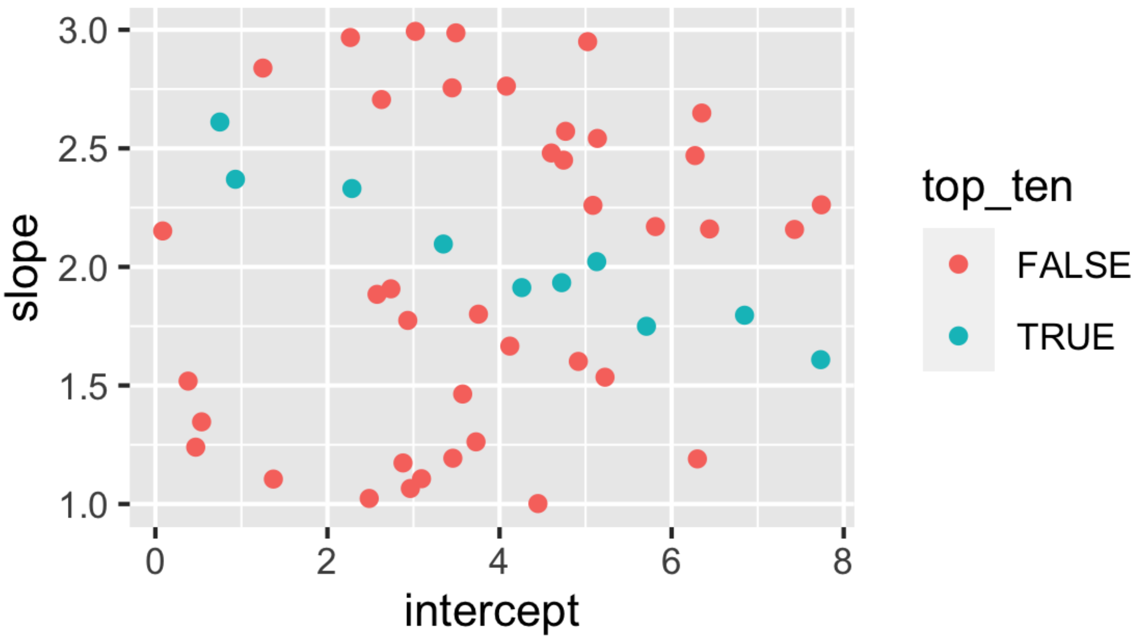
- Filter and plot the top 10 models

```
ggplot(sim1, aes(x, y)) +
  geom_point() +
  geom_abline(
    data = filter(models, rank(deviations) <= 10),
    aes(intercept = intercept, slope = slope),
    alpha = 1/4
  )
```

Modeling

- What values of coefficients fit the data the best?

```
models %>%
  mutate(top_ten = rank(deviations) <= 10) %>%
  ggplot(aes(x = intercept, y = slope, color = top_ten)) +
  geom_point()
```



Modeling

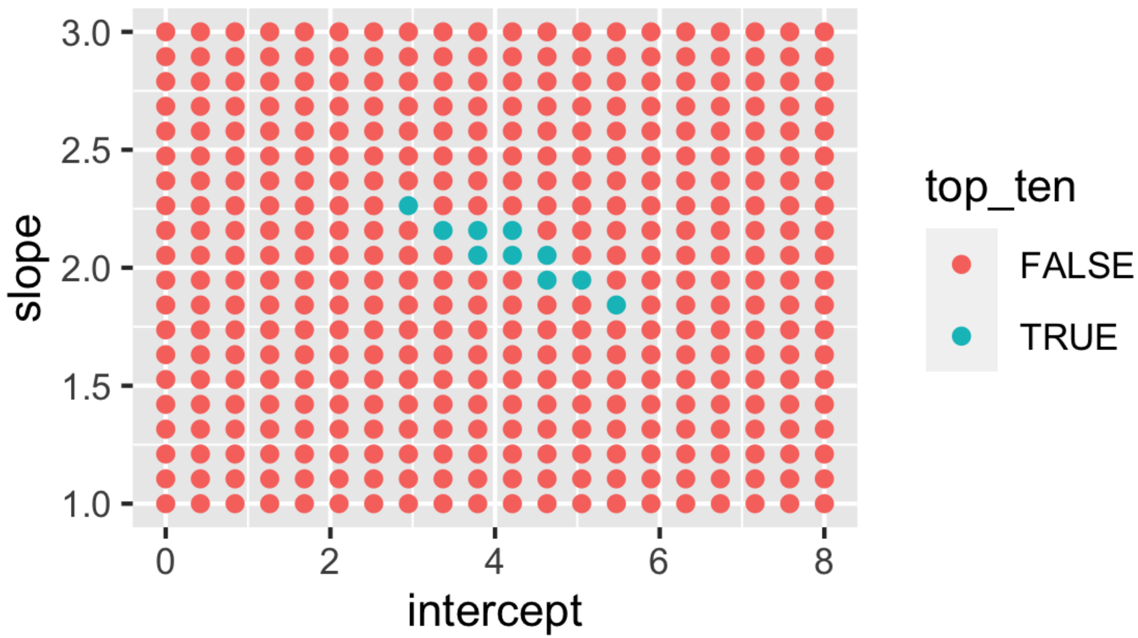
- Search over grid of parameters

```
models <- expand.grid(
  intercept = seq(0, 8, length = 20),
  slope     = seq(1, 3, length = 20)
)

models <- models %>%
  mutate(deviations = map2_dbl(intercept, slope, sum_of_squared_deviations, data = sim1)
  mutate(top_ten = rank(deviations) <= 10)
```

Modeling

- Search over grid of parameters

```
models %>%
  ggplot(aes(x = intercept, y = slope, color = top_ten)) +
  geom_point()
```



Modeling

- Find the optimal (lowest squared deviation) solution using the `optim()` function

```
#? optim
measure_distance <- function(coefficients, data) {
  sum_of_squared_deviations(coefficients[1], coefficients[2], data)
}
best <- optim(c(0, 0), measure_distance, data = sim1)

## optimal model
best$par
```

```
## [1] 4.222247799614617491670 2.051203813178364754322
```