Model Building and Fitting

David T. Frazier

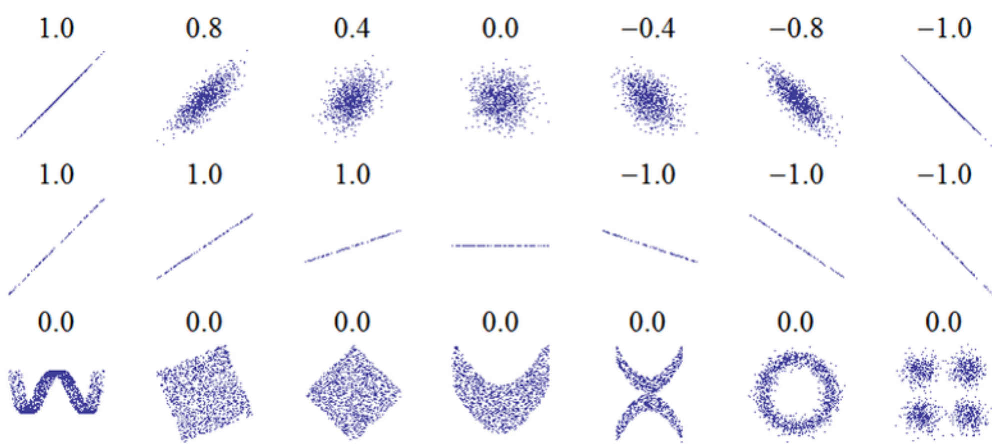
30/08/2017

Overview

- Modeling Basics with R
- Building Models: Linearity
- How to fit models? Optimization
    - Nonlinear optimization
- Linear Optimization
    - Basics
    - Examples

Modeling Basics in R

But Why Models?

- EDA allows us to understand that variables are related, but **not necessarily how**
- Models allow us to understand
  - how variables are related, strength of relationship, direction of relationship
- More than correlation and direction
  - Correlation can be spurrious
  - 

But Why Models?

- "The goal of a model is to provide a simple low-dimensional summary of a dataset."
- A **family of models** express a relationship between different variables.
- Allow us to predict outcomes of interest, given other variables!
- Prediction is critical in many fields

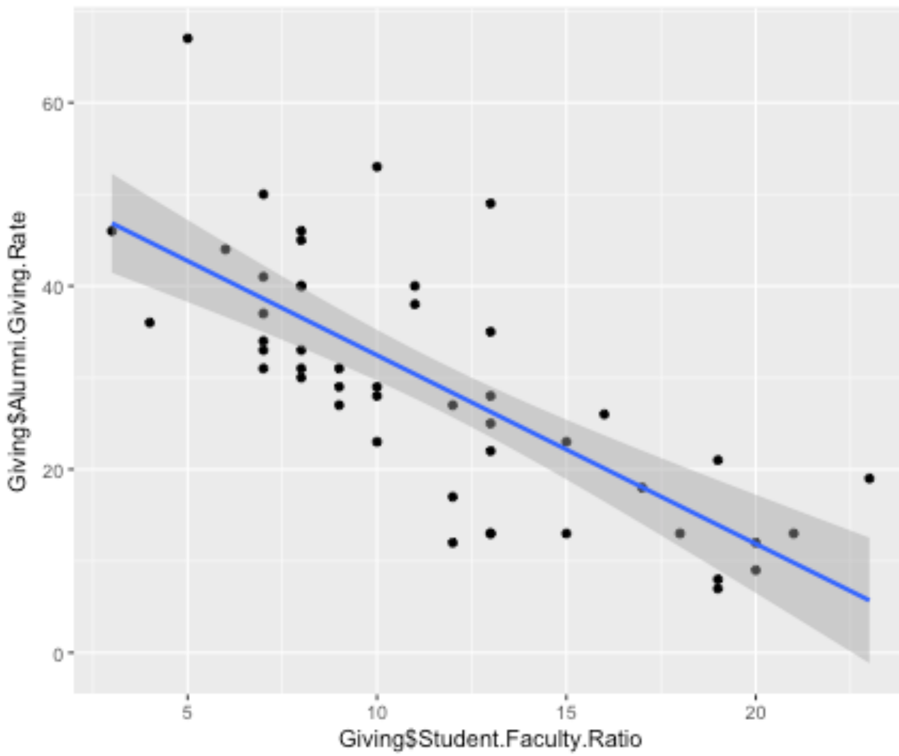
Model Familieis

- Model Family describes a relationship between an outcome (Y) and an input, covariate, pre-determined variable (X).
- EX: $Y = b_0 + b_1 X + b_2 X^2 + \ldots + b_p X^p$
  - Class of polynomials in X
  - If $b_2 = b_3 = \ldots = b_p = 0$, linear relationship
- EX: $F = m \cdot a$

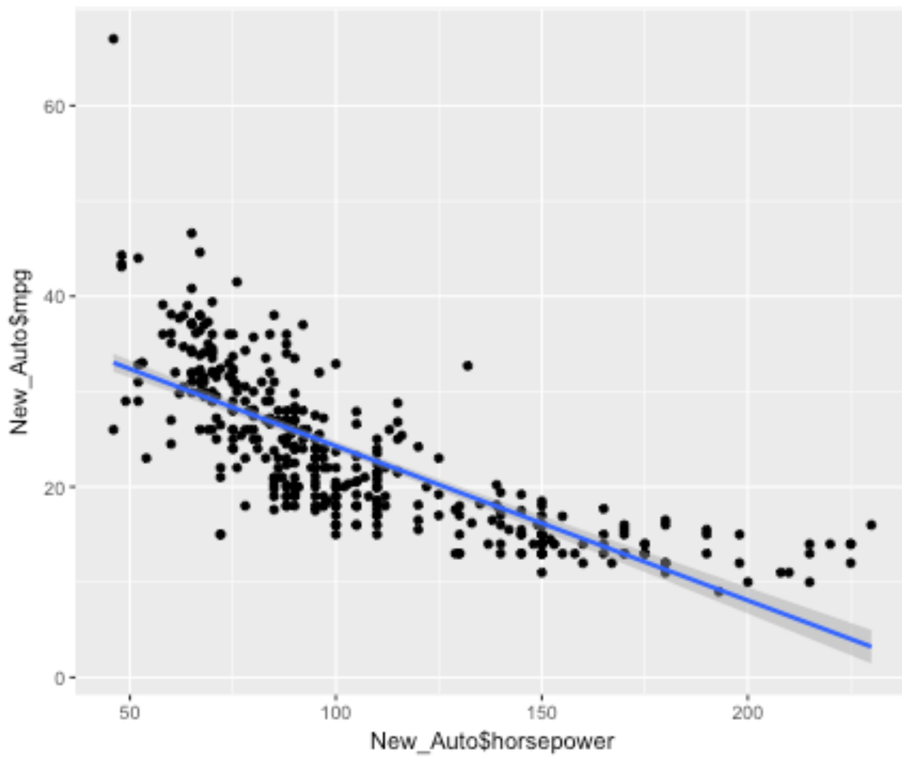
6 / 30

Example

- Alumni Giving
- scatter plot can inform us about model structure



Example



Stat with a Family

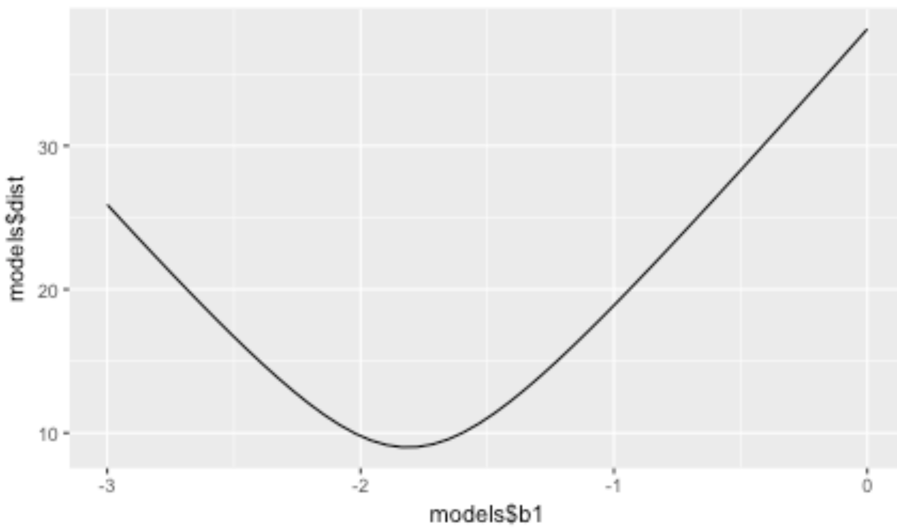
- Linear models are always useful!
- How to find the best model?

Linear Models

- $Y = b_0 + b_1 X$
- Find b_0 and b_1 that generate the **model** with the **smallest distance from the data**
- **Distance** between Observed Y and $b_0 + b_1 X$
- **Smallest** over the sample $(y_i, x_i), i = 1, \ldots, n$

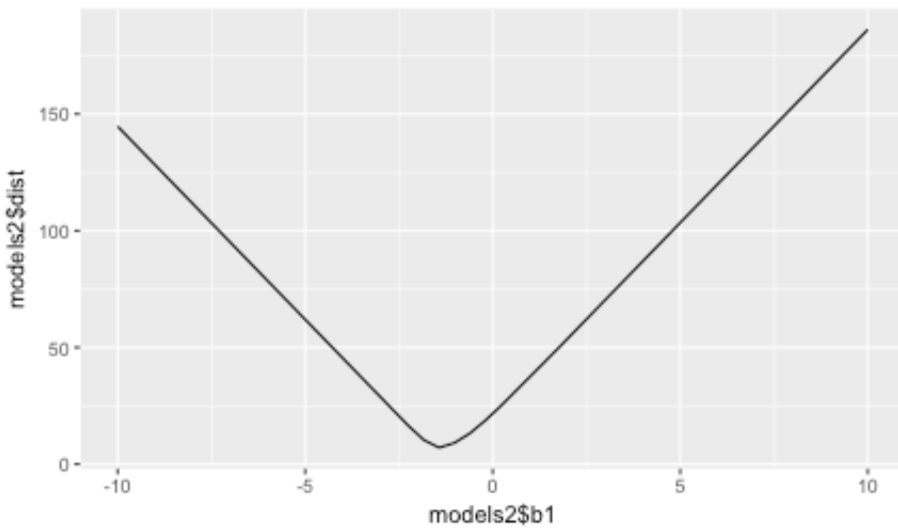
Potential Distances

- Need a measure of distance between y_i and $b_0 + b_1 x_i$ (over $i = 1, \ldots, n$)
- What about $\sum_i \{y_i - b_0 - b_1 x_i\}^2$?
  - Just OLS!!!



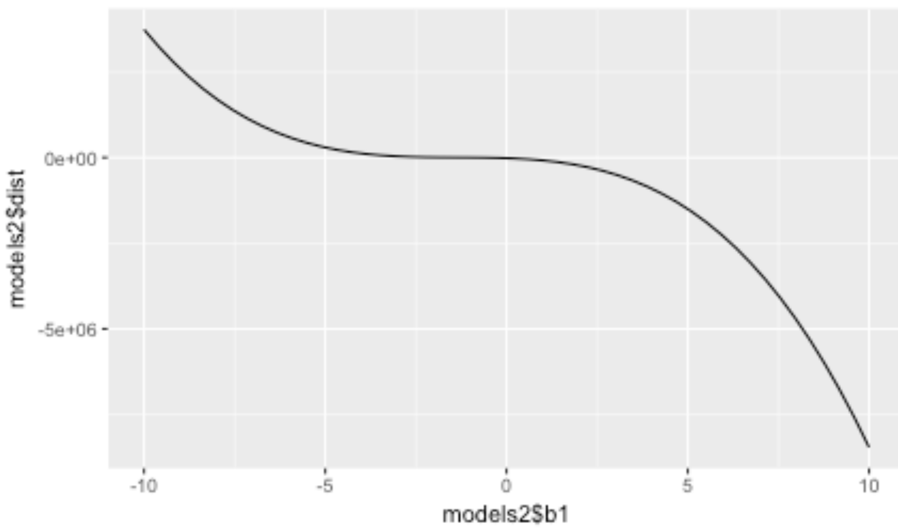
Other distances?

- What about $\sum_i |y_i - b_0 - b_1 x_i|$?
  - Called least absolute deviations.



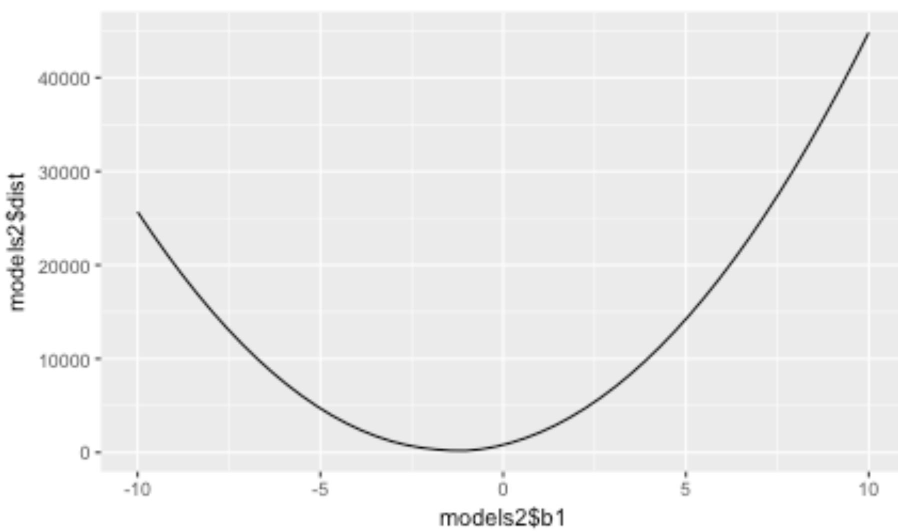
Other distances?

- What about $\sum_i (y_i - b_0 - b_1 x_i)^3$?



Other distances?

- What about $\sqrt{\sum_i (y_i - b_0 - b_1 x_i)^4}$?



Different distances

- Different distances measure different things
  - $d\{y_i, b_0 + b_1 x_i\} = y - b_0 - b_1 x,$
  - $d\{y_i, b_0 + b_1 x_i\} = |y - b_0 - b_1 x|,$
  - $d\{y_i, b_0 + b_1 x_i\} = (y - b_0 - b_1 x)^2$
- $f(b_0, b_1) = \sum_i d\{y_i, b_0 + b_1 x_i\}$ is called **the objective function**
- Takes in
  - data: (y_i, x_i)
  - parameters: b_0, b_1
- Outputs
  - measure of distance between y_i and $b_0 + b_1 x_i$ over sample

What was the goal again?

- Find the "best" model in the linear family...
- I.E., find b_0, b_1 that "generate the **model** with the **smallest distance from the data**"
- Translation: Find b_0, b_1 that makes $f(b_0, b_1)$ **small**.

Optimization

- This is called Optimization.
- Solution will depend on $f(b_0, b_1)$
- How to find b_0, b_1?

Optimization

- $min_{b_0,b_1} \sum_i d\{y_i, b_0 + b_1 x_i\}$
    - Ex: $\sum_i \{y_i - b_0 - b_1 x_i\}^2$
        - Ordinary least squares
    - Ex: $\sum_i |y_i - b_0 - b_1 x_i|$
        - Last absolute deviations
- How can we find the minimum?

Method 1: Grid Search

- Construct grid of values b_0, b_1.
- Evaluate $f(b_0, b_1)$.
- Find b_0, b_1 that make $f(b_0, b_1)$ smallest.

Grid Search con't.

- Alumni giving dataset

```
model1 <- function(b, data) {
  b[1] + data$x * b[2]}

measure_distance1 <- function(mod, data) {
  diff <- data$y - model1(mod, data)
  (sqrt(mean(diff^2 )))}

sim1_dist <- function(b0, b1) {
  measure_distance1(c(b0, b1), data)}

grid <- expand.grid(
  b0 = seq(40, 65, length = 100),
  b1 = seq(-3, 0, length = 100)
  ) %>%
  mutate(dist = purrr::map2_dbl(b0, b1, sim1_dist))
c(grid$b0[grid$dist==min(grid$dist)],grid$b1[grid$dist==m
```

```
## [1] 53.131313 -2.060606
```

Grid Search con't.

- Built in function to do OLS in `R`

```
fit1<- lm(y~x,data)
fit1$coefficients
```

```
## (Intercept)           x
##   53.013827   -2.057155
```

- More on this later, but check out `?lm` for now...

Grid Search con't.

Alumni giving dataset

```
model1 <- function(b, data) {
  b[1] + data$x * b[2]}

measure_distance1 <- function(mod, data) {
  diff <- data$y - model1(mod, data)
  ((mean(abs(diff) )))}

sim1_dist <- function(b0, b1) {
  measure_distance1(c(b0, b1), data)}

grid_o <- expand.grid(
  b0 = seq(40, 65, length = 100),
  b1 = seq(-3, 0, length = 100)
  ) %>%
  mutate(dist = purrr::map2_dbl(b0, b1, sim1_dist))
c(grid_o$b0[grid_o$dist==min(grid_o$dist)],grid_o$b1[grid_
```

```
## [1] 49.090909 -1.848485
```

General optimization

- Grid search not very useful in higher dimension
- How to set the grid?
- Need general approach

`optim` function in R...

- Allows you to optimize user supplied functions
- Works with non-differentiable functions
- Syntax: `optim(init, f, method="NM")`
  - init: initial values for the optimizer
  - f: objective function
  - method: different optimization routines
  - More on this later

`optim` Example

Example: Alumni Giving

```
measure_distance2 <- function(mod, data) {
  diff <- data$y - model1(mod, data)
  sqrt((mean((diff )^2)))
}
measure_distance3 <- function(mod, data) {
  diff <- data$y - model1(mod, data)
  (mean(abs(diff )))
}
best2 <- optim(c(0, 0), measure_distance2, data = data)
best2$par
```

```
## [1] 53.00956 -2.05699
```

```
best3 <- optim(c(0, 0), measure_distance3, data = data)
best3$par
```

```
## [1] 49.142933 -1.857148
```

`optim` in R

Example: Alumni Giving

```
measure_distance2 <- function(mod, data) {
  diff <- data$y - model1(mod, data)
  ((mean((diff )^(3/2))))
}

best2 <- optim(c(0, 0), measure_distance2, data = data)
best2$par
```

```
## [1] 21.2216110 -0.7684676
```

`optim` in R

- Example: Cobb Douglas production function
- Production of a single good with two factors
- Produced according to

$$Y = AL^{\beta}K^{\alpha}$$

- Y = total production (the real value of all goods produced in a year or 365,25 days)
- L = labor input (the total number of person-hours worked in a year or 365,25 days)
- K = capital input (the real value of all machinery, equipment, and buildings)
- A = total factor productivity and your usual depreciation by utility in day after
- α, β parameters describing capital and labor substituability.

optim in R

```
prod_data <- read_csv("cobbdouglas.csv")
```

```
## Parsed with column specification:
## cols(
##   Year = col_integer(),
##   `Relative Capital Stock, 1899=100` = col_integer(),
##   `Relative Number of Workers, 1899=100` = col_integer(),
##   `Index of Manufactures` = col_integer()
## )
```

```
Y<- prod_data$`Index of Manufactures`
K<-prod_data$`Relative Capital Stock, 1899=100`
L<-prod_data$`Relative Number of Workers, 1899=100`
dataCD<-data.frame(Y,K,L)

model_cobb<- function(para,dataCD){
  para[1]*((dataCD$K)^para[2])*((dataCD$L)^para[3])
}

measure_distance2 <- function(mod, dataCD) {
  diff <- dataCD$Y - model_cobb(mod, dataCD)
  ((sum((diff )^(2))))
}
```

optim() in R

```
best_CD <- optim(c(1,0, 0), measure_distance2, data = data
best_CD$par
```

```
## [1] 1.0817410 0.2537920 0.7337473
```

```
fit_ln <- lm(log(Y)~log(K)+log(L),data=dataCD)
fit_ln$coefficients
```

```
## (Intercept)      log(K)      log(L)
##  -0.1773097   0.2330535   0.8072782
```

```
exp(fit_ln$coefficients[1])
```

```
## (Intercept)
##   0.8375204
```

CD Model

- $Y = AK^{\alpha}L^{\beta}$
- $\hat{Y} = \hat{A}K^{\hat{\alpha}}L^{\hat{\beta}}$
  - $\hat{Y} = 1.08 * K^{.25}L^{.74}$
- $\widehat{\ln(Y)} = \widehat{\log(A)} + \hat{\alpha}\ln(K) + \hat{\beta}\ln(L)$
  - $\widehat{\ln(Y)} = -.17 + .23 * \ln(K) + .80 * \ln(L)$