Regression

Part 1

Prof. Bisbee

Seoul National University

Slides Updated: 2024-07-16

Agenda

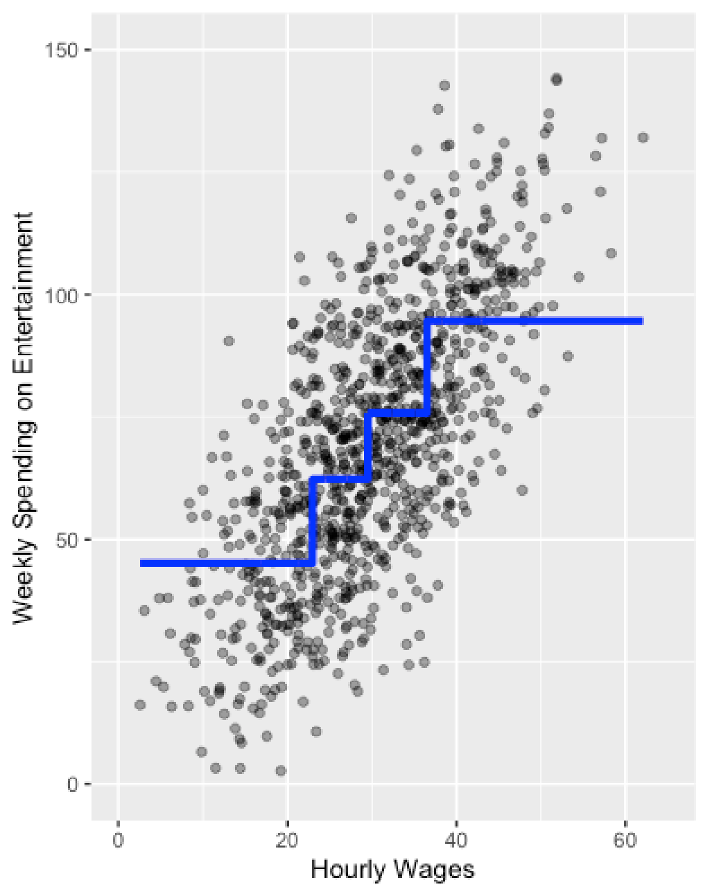
1. Modeling Conditional Variation
2. Adding Regression to the **Process**
3. Introducing the Data
4. Demonstrating Regressions

Regression & Conditional Analysis

- Recall our discussion of **conditional** analysis
  - Conditional → **depends on**
  - Analyze with **conditional means**

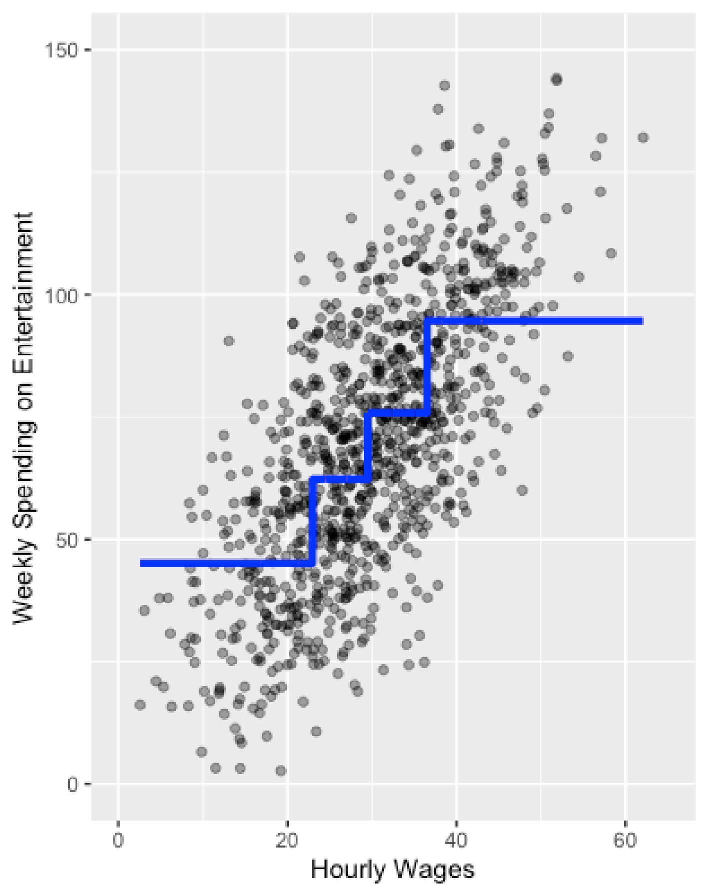
What is regression?

- Conditional means for continuous data



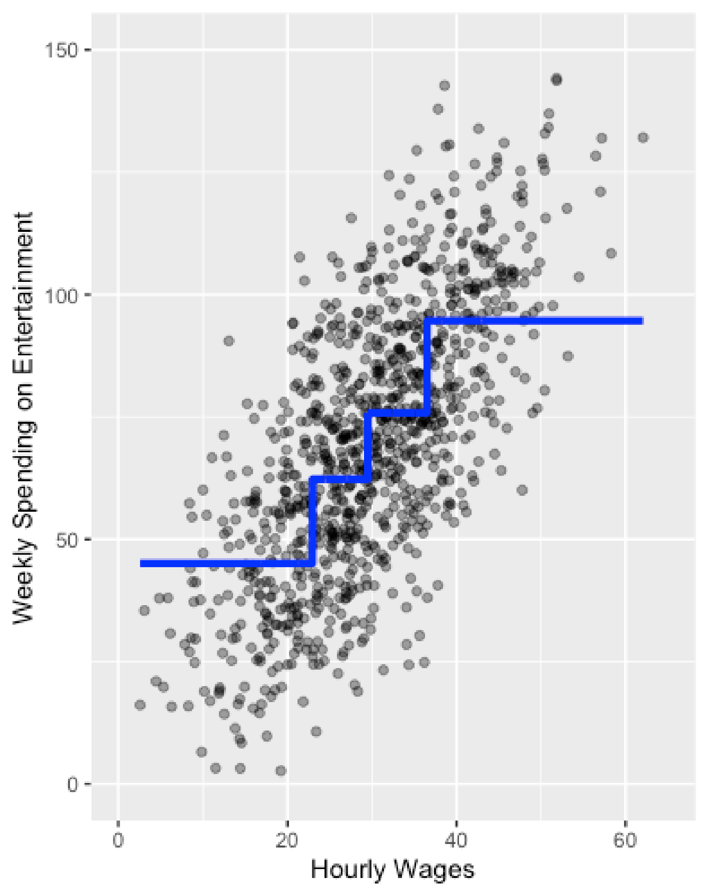
What is regression?

- People with hourly wages < \$20 spend ~\$50 on entertainment per week



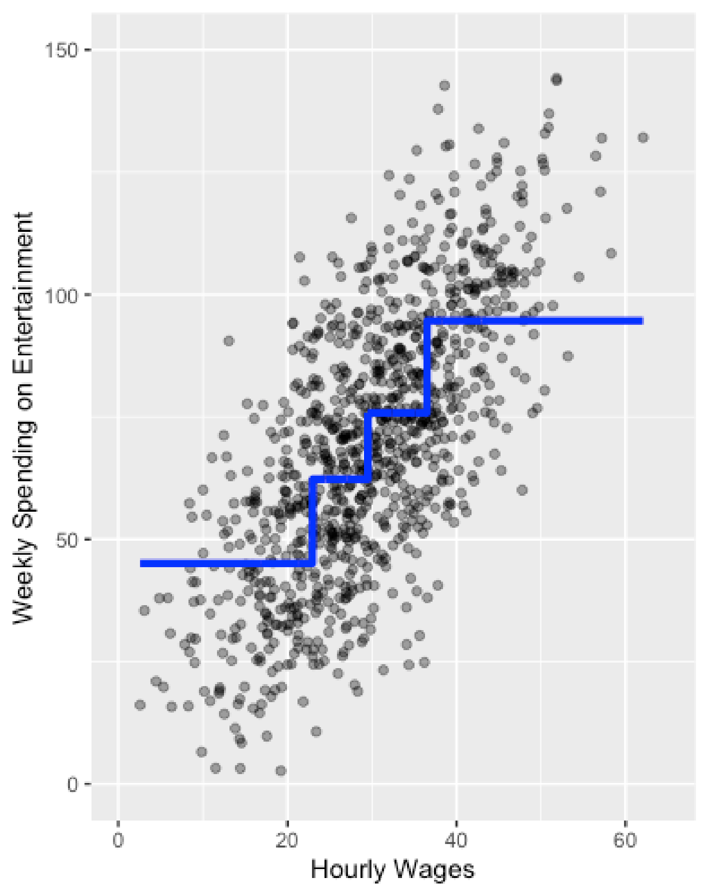
What is regression?

- People with hourly wages > $40 spend ~$95 on entertainment per week



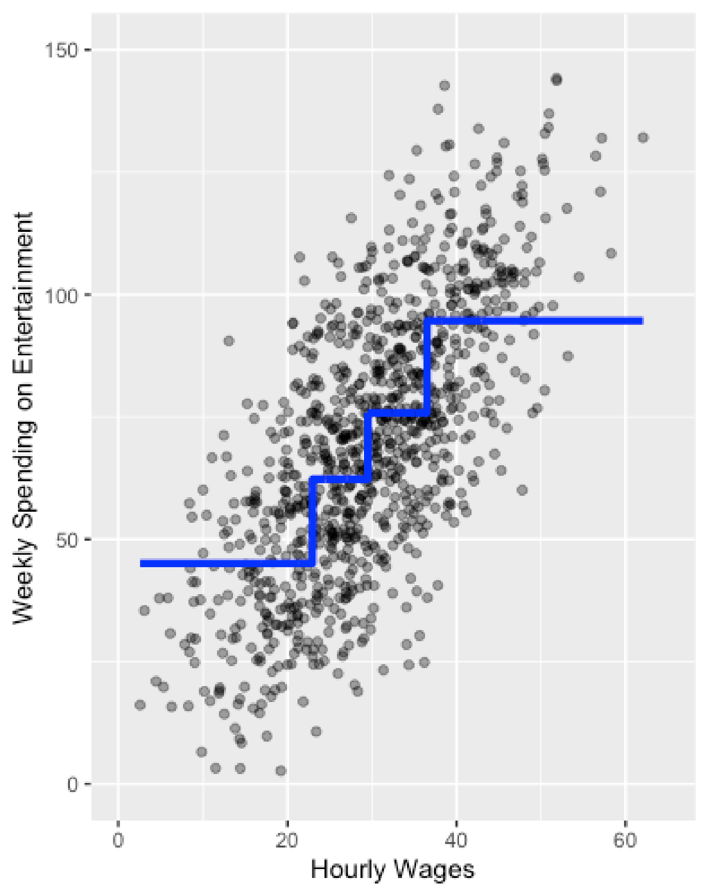
What is regression?

- Theory: the more you earn, the more you spend



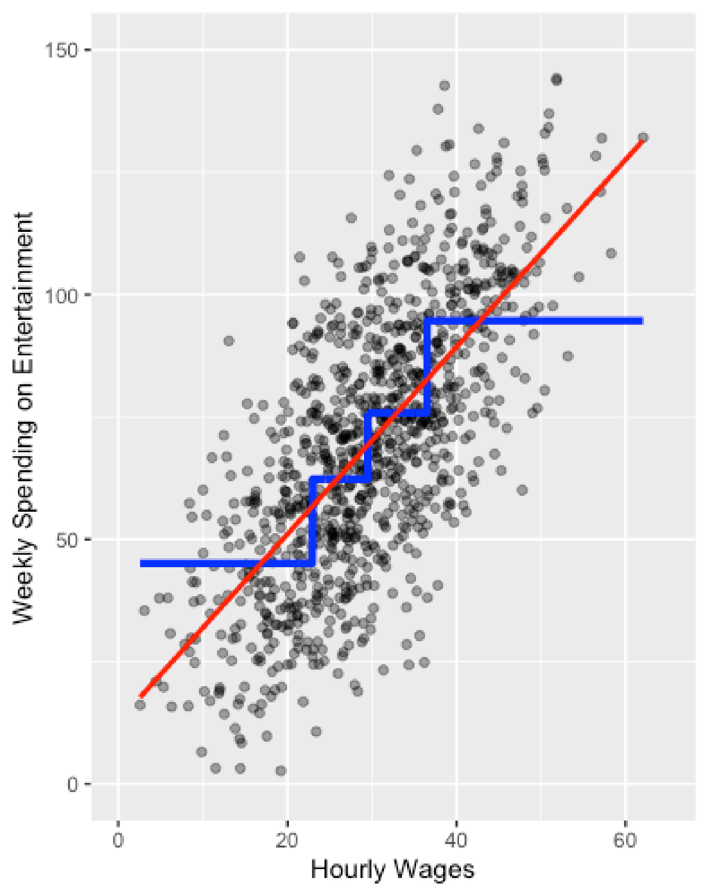
What is regression?

- But **conditional means** make a lot of mistakes. Can we do better?



What is regression?

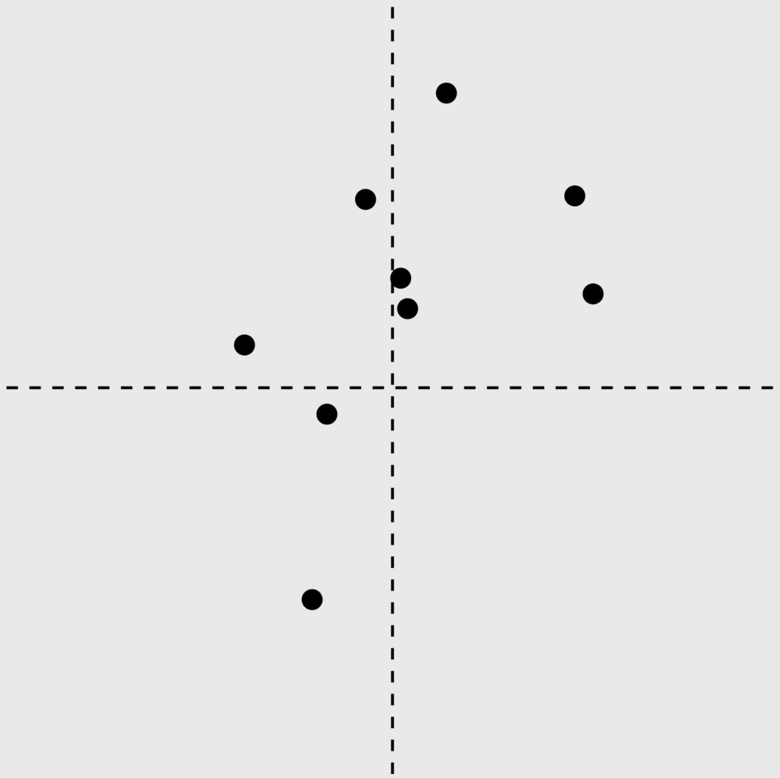
- But **conditional means** make a lot of mistakes. Can we do better?



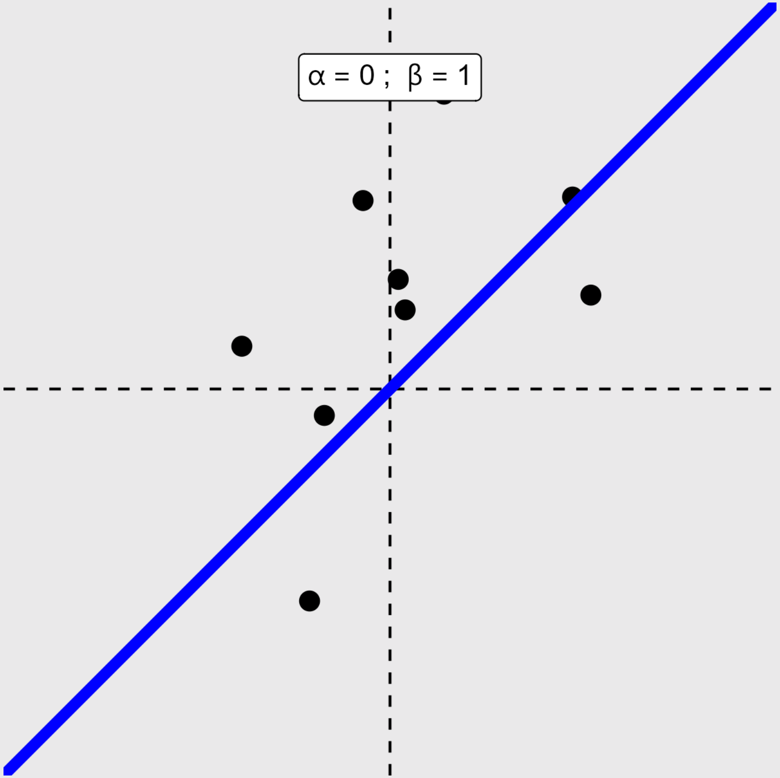
Regression

- Calculating a **line** that minimizes mistakes *for every observation*
  - NB: could be a curvey line! For now, just assume straight
- Recall from geometry how to graph a straight line
- $Y = a + bX$
  - a: the "intercept" (where the line intercepts the y-axis)
  - b: the "slope" (how much Y changes for each increase in X)
- (Data scientists use α and β instead of a and b b/c nerds)
- Regression analysis simply chooses the best line
  - "Best"?
  - The line that minimizes the mistakes (the **line of best fit**)

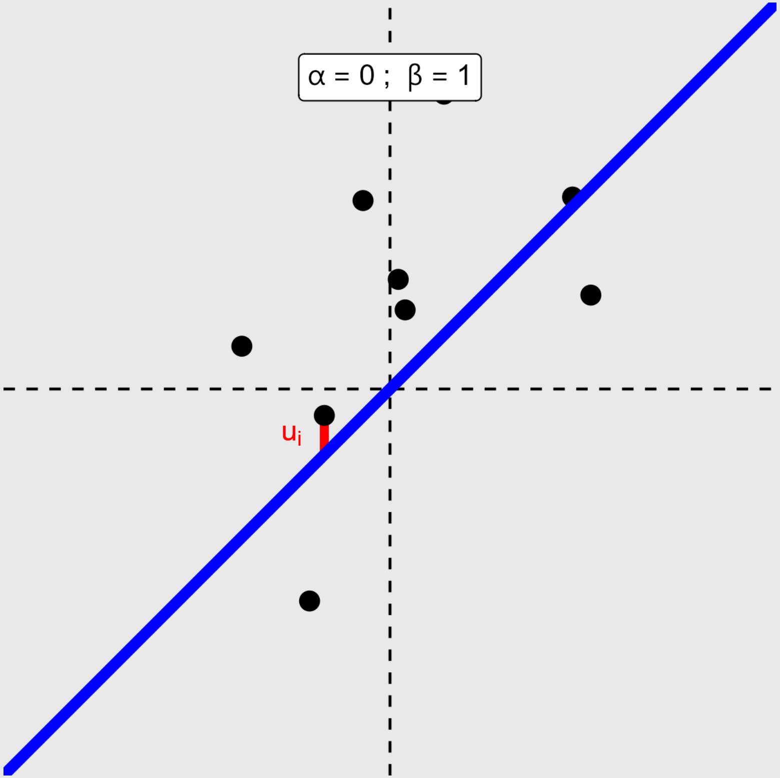
Linear Regression



Linear Regression

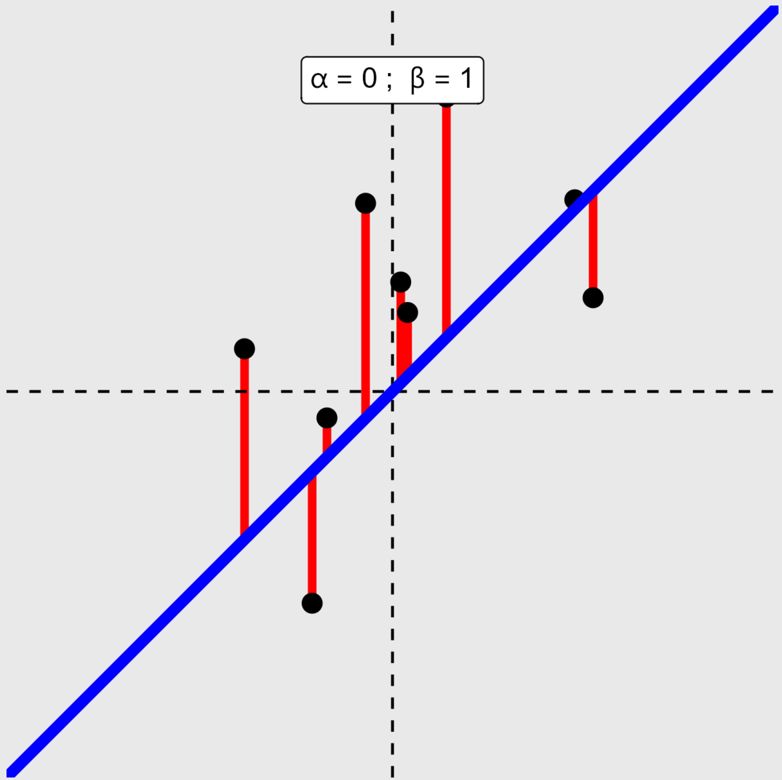


Linear Regression



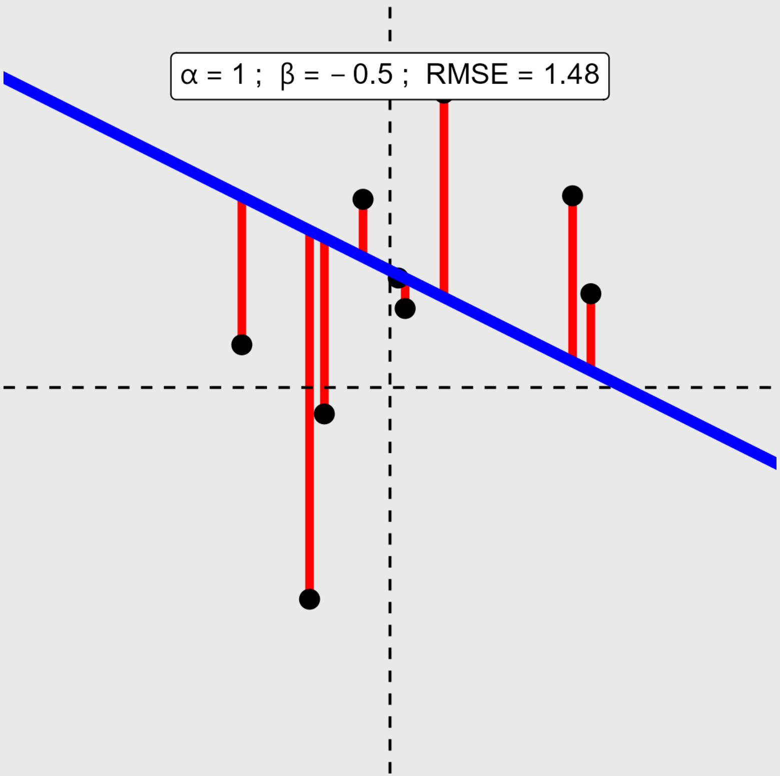
- **Error/Residual**: mistake made by a line
    - In math: $u_i = y_i - \hat{y}_i$
    - In English: difference between true outcome value (y_i) and prediction ($\hat{y}_i$)

Linear Regression



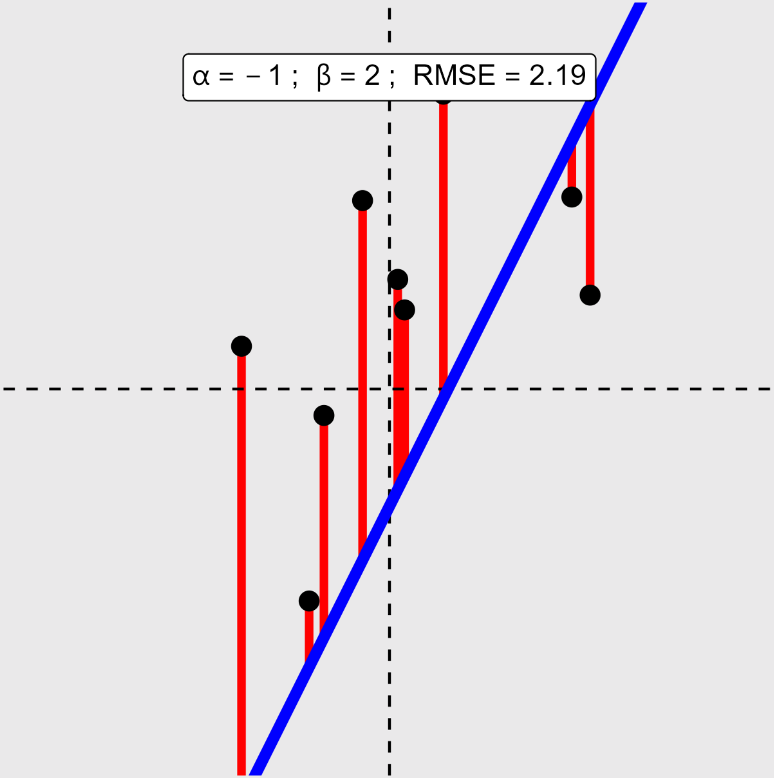
- Use **errors** to find **line of best fit**
- **RMSE** (**R**oot **M**ean **S**quared **E**rror)
  - Square the errors
  - Take their average
  - Take the square root
- **RMSE** = 1.23

Linear Regression



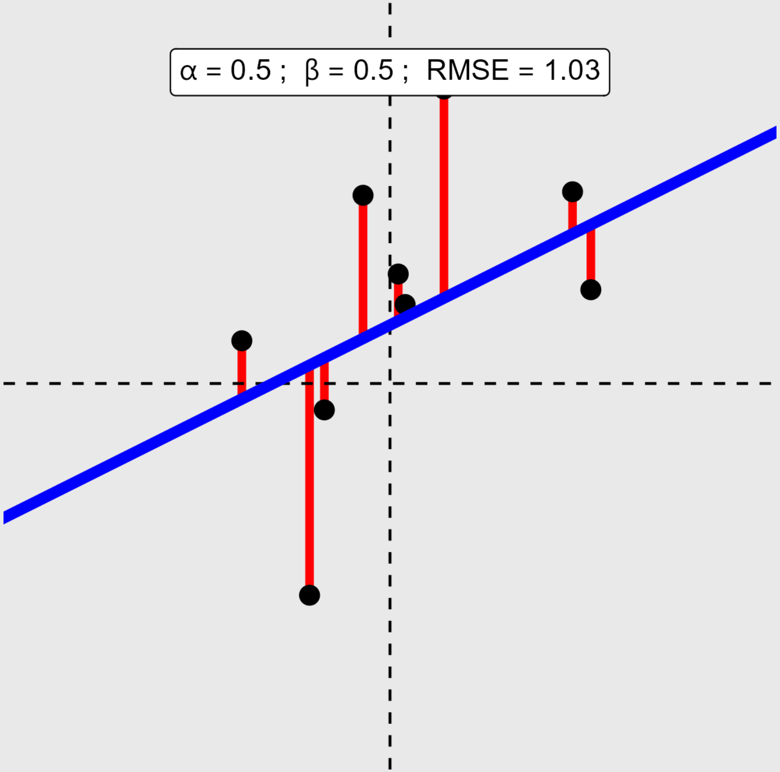
- Use **errors** to find **line of best fit**
- **RMSE** (**R**oot **M**ean **S**quared **E**rror)
  - Square the errors
  - Take their average
  - Take the square root
- **RMSE** = 1.48

Linear Regression



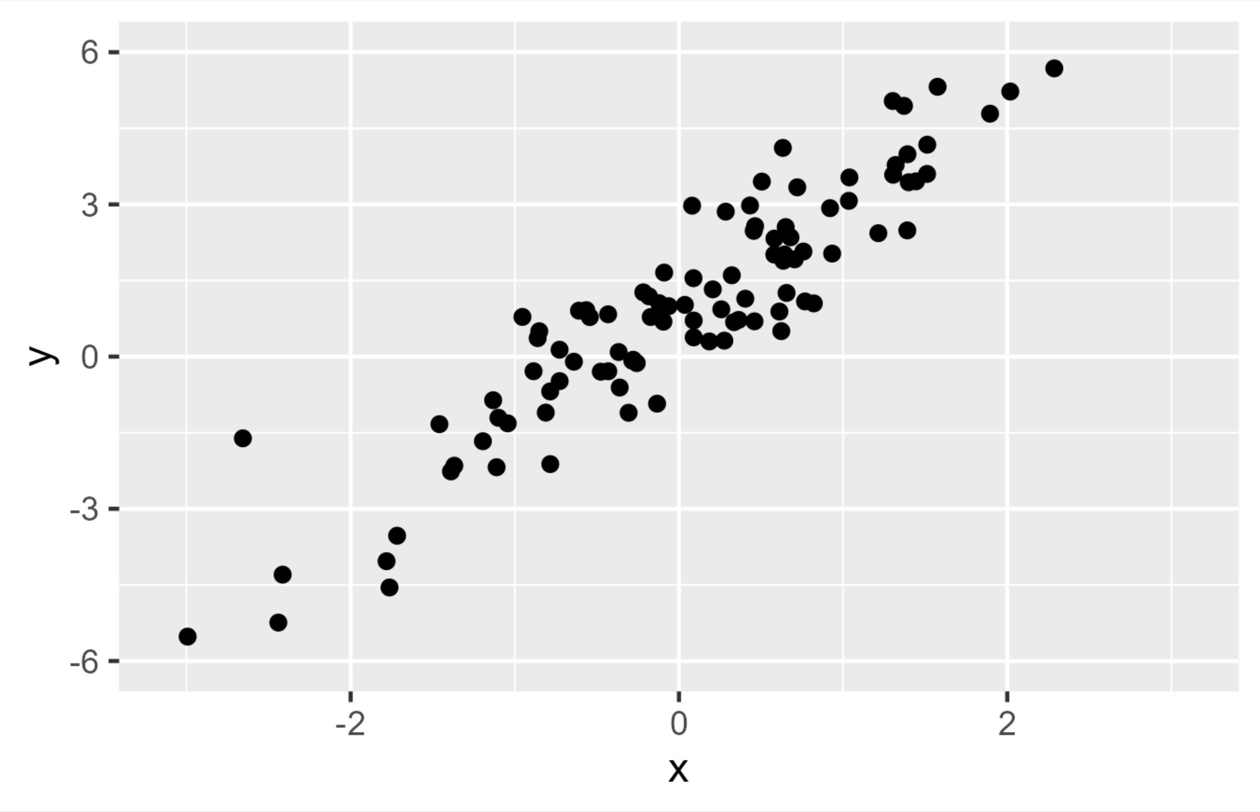
- Use **errors** to find **line of best fit**
- **RMSE** (**R**oot **M**ean **S**quared **E**rror)
  - Square the errors
  - Take their average
  - Take the square root
- **RMSE** = 2.19

Linear Regression

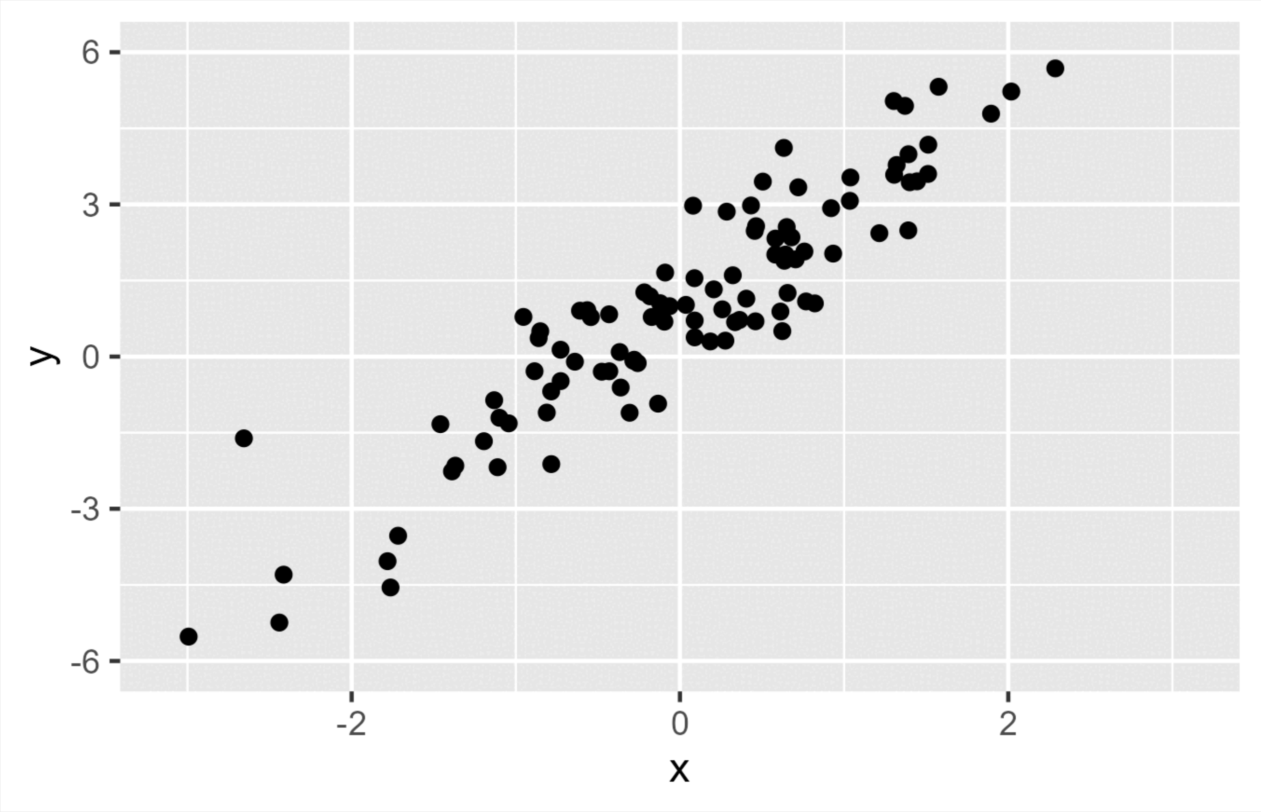


- Use **errors** to find **line of best fit**
- **RMSE** (**R**oot **M**ean **S**quared **E**rror)
  - Square the errors
  - Take their average
  - Take the square root
- **RMSE** = 1.03

Visual Intuition



Visual Intuition



Regression

- The line is substantively meaningful
- Red line on scatter plot of spending and wages: $Y = \underbrace{12}_{\alpha} + \underbrace{2}_{\beta} * X$
- α tells us the value of Y when X is zero
  - People who don't make any money spend $12 per week on entertainment
- β tells us how much Y increases for each additional X
  - People spend an additional $2 per week for each additional $1 in hourly wages

Regression

- These are called **"linear models"**
  - **Not** because the line is straight (it might not be)
  - but because the components are additive ($\alpha + \beta X$)
- Can extend to multiple predictors (X 's)
  - $Y = \alpha + \beta_1 X_1 + \beta_2 X_2 + \cdots + \varepsilon$
  - X_1 might be wages and X_2 might be age (for example)
  - The final term ε measures how bad our mistakes are

Regression

- Let's demonstrate with the `debt` data

```r
require(tidyverse)

debt <-
read_rds('https://github.com/jbisbee1/ISP_Data_Science_2024/raw/main/da

glimpse(debt)
```

```
## Rows: 2,546
## Columns: 16
## $ unitid        <int> 100654, 100663, 100690, 100706, 100…
## $ instnm        <chr> "Alabama A & M University", "Univer…
## $ stabbr        <chr> "AL", "AL", "AL", "AL", "AL", "AL",…
## $ grad_debt_mdn <int> 33375, 22500, 27334, 21607, 32000, …
## $ control       <chr> "Public", "Public", "Private", "Pub…
## $ region        <chr> "Southeast", "Southeast", "Southeas…
## $ preddeg       <chr> "Bachelor's", "Bachelor's", "Associ…
## $ openadmp      <int> 2, 2, 1, 2, 2, 2, 1, NA, 2, 2, 2, 1…
## $ adm_rate      <dbl> 0.9175, 0.7366, NA, 0.8257, 0.9690,…
## $ ccbasic       <int> 18, 15, 20, 16, 19, 15, 2, 22, 18, …
## $ sat_avg       <int> 939, 1234, NA, 1319, 946, 1261, NA,…
```

Research Camp

- Research Question: What is the relationship between SAT scores and median future earnings?
- Theory: Students with higher SAT scores work harder and have learned more. Employers reward these attributes with higher wages in the private market.
- Hypothesis: The relationship between SAT scores and future earnings should be positive.
    - **NB**: Important caveats to this simplistic theory!
    - Socioeconomic status: predicts both higher SAT scores and higher wages
    - **Correlation $\neq$ Causation**

Set Up

- Linking Theory to Data
- Our SAT scores are theorized to explain future earnings
    - Thus the SAT scores are the independent / explanatory / predictor variable X
    - And earnings are the dependent / outcome variable Y

Regression

- There is a simple recipe to follow
- And it is exactly how the syllabus for the class is designed!
  1. Look at your data to identify missingness (**Wrangling: Lecture 3**)
  2. **Univariate** visualization of your variables (**Lecture 4**)
  3. **Multivariate** visualization of your variables (**Lecture 5**)
  4. **Regression** (today)
  5. Evaluation of **errors** (today)

Step 1: Look

- Why worry about **missingness**?

1. **Substantive:** external validity

2. **Technical:** cross validation won't work!

```
summary(debt %>% select(sat_avg,md_earn_wne_p6))
```

```
##     sat_avg     md_earn_wne_p6
##  Min.   : 737   Min.   : 10600
##  1st Qu.:1053   1st Qu.: 26100
##  Median :1119   Median : 31500
##  Mean   :1141   Mean   : 33028
##  3rd Qu.:1205   3rd Qu.: 37400
##  Max.   :1557   Max.   :120400
##  NA's   :1317   NA's   :240
```

Step 2: Univariate Viz

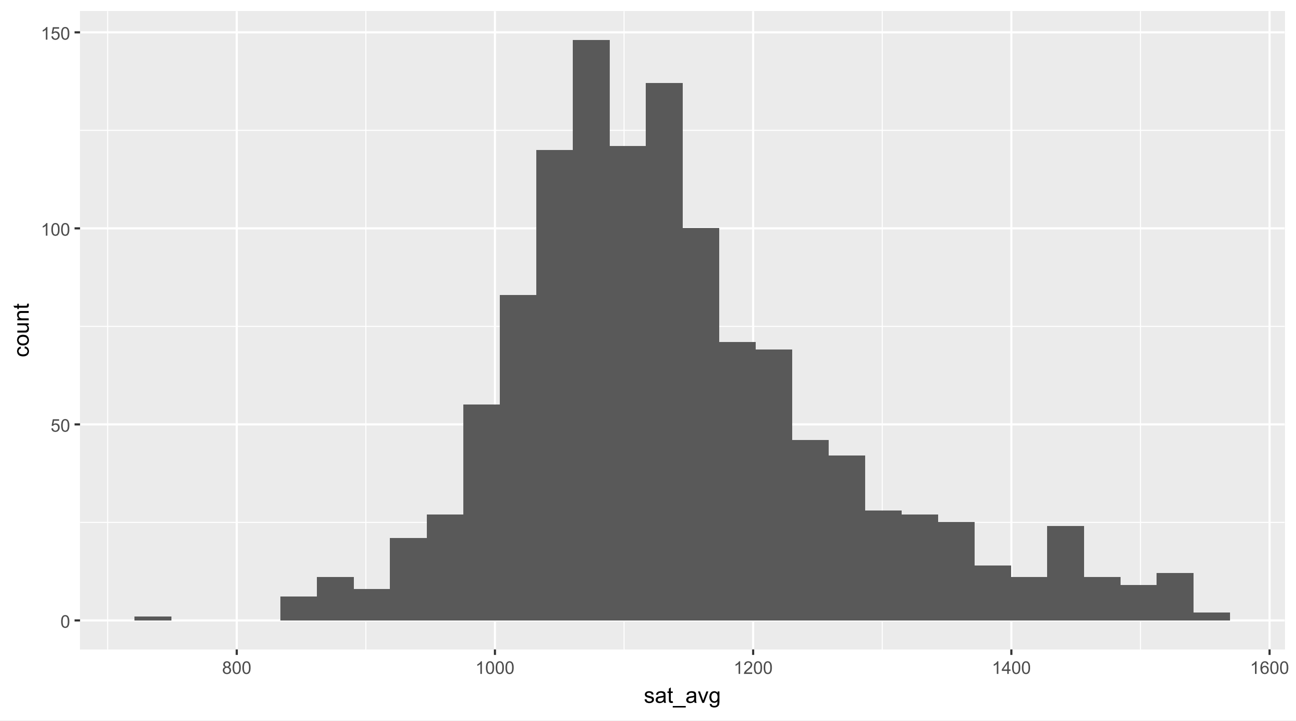
- Why visualize both Y and X?

1. **Substantive:** *See* which units you are talking about
2. **Technical:** Adjust for *skew*

Step 2: Univariate Viz

- Why visualize both Y and X?

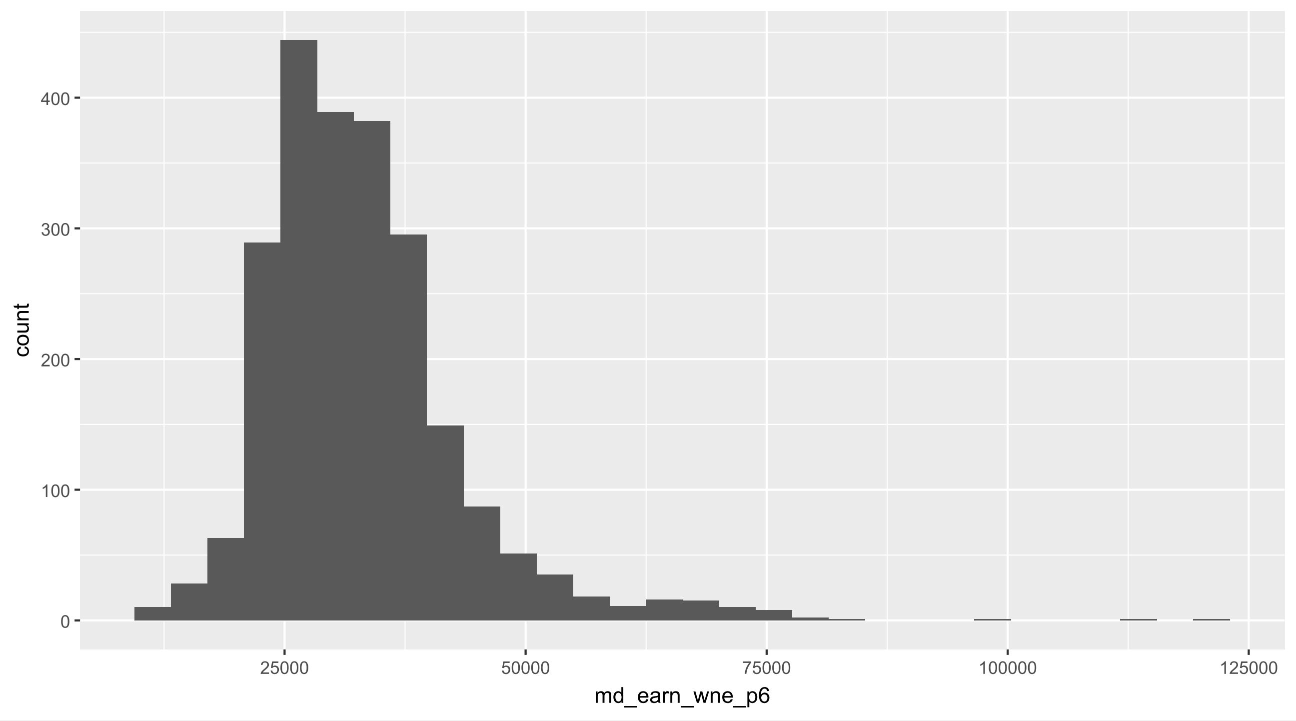
```
debt %>%
  ggplot(aes(x = sat_avg)) +
  geom_histogram()
```



Step 2: Univariate Viz

- Why visualize both Y and X?

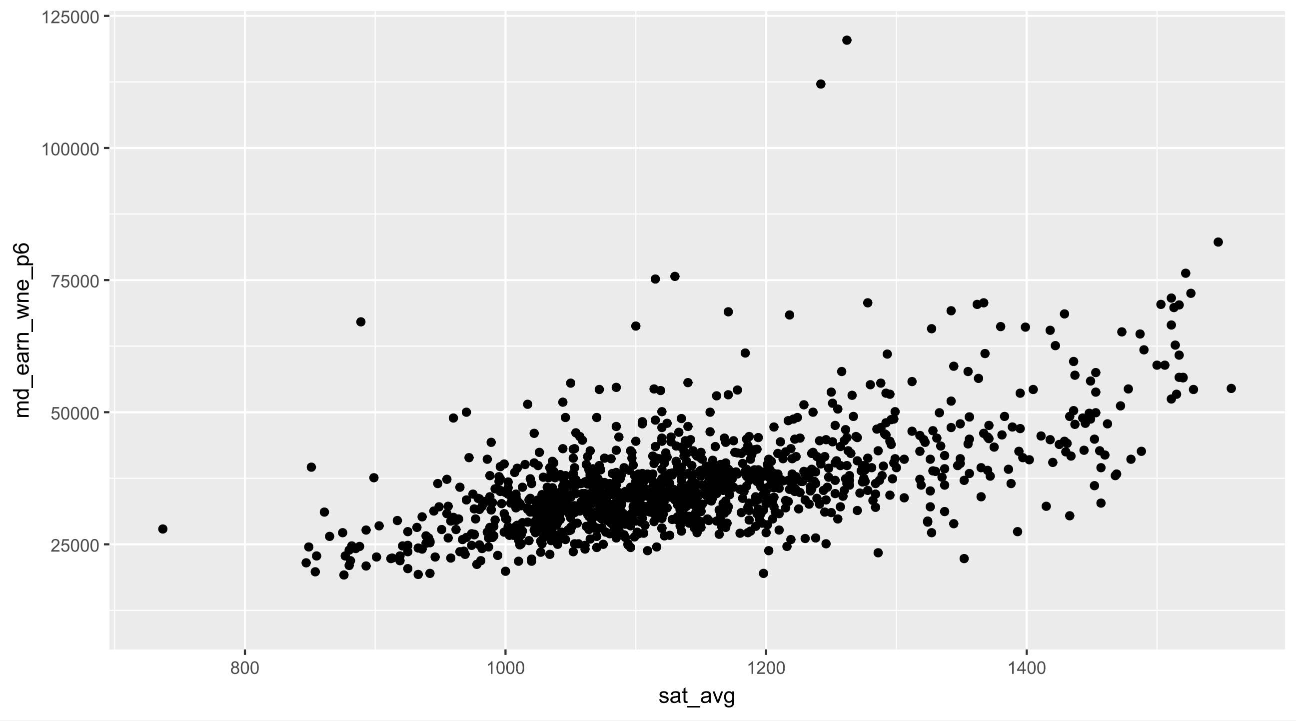
```
debt %>%
  ggplot(aes(x = md_earn_wne_p6)) +
  geom_histogram()
```



Step 3: Multivariate

- Eyeball the relationship first!

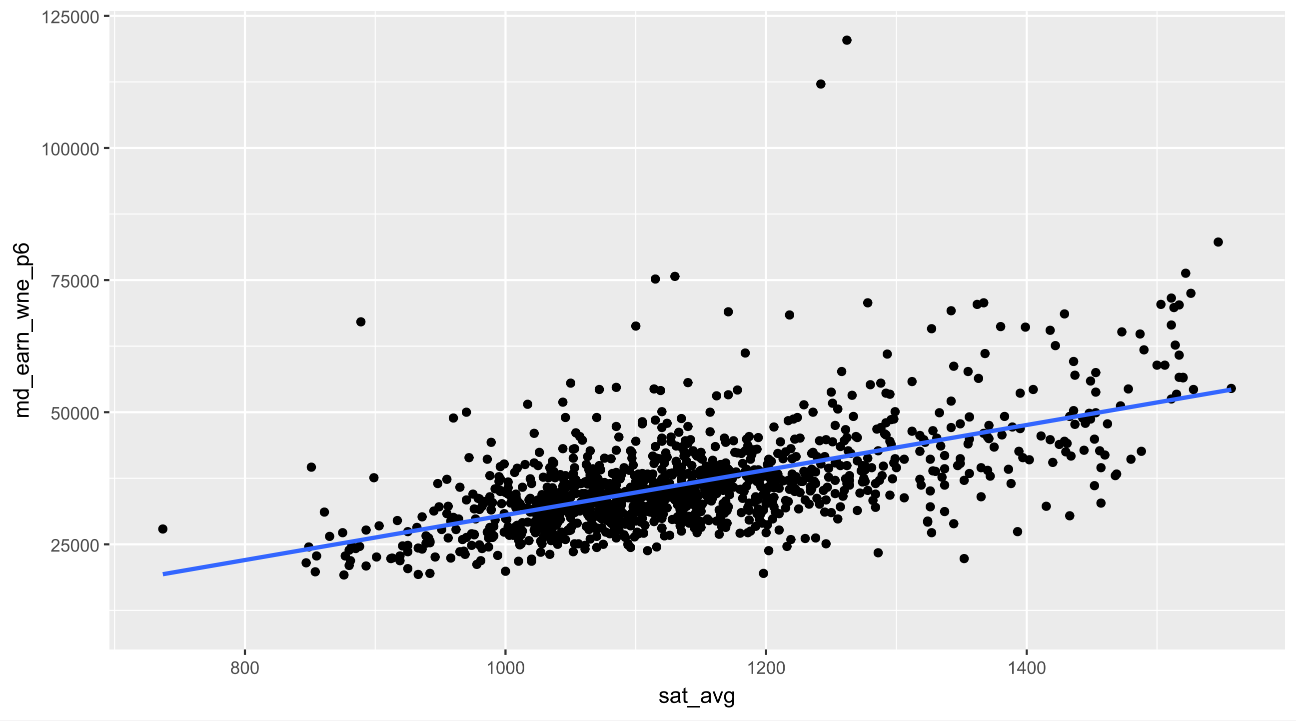
```
debt %>%
  ggplot(aes(x = sat_avg,y = md_earn_wne_p6)) +
  geom_point()
```



Step 3: Multivariate Viz

- Adding regression line

```
debt %>%
  ggplot(aes(x = sat_avg,y = md_earn_wne_p6)) +
  geom_point() + geom_smooth(method = 'lm',se = F)
```



Step 3: Multivariate Viz

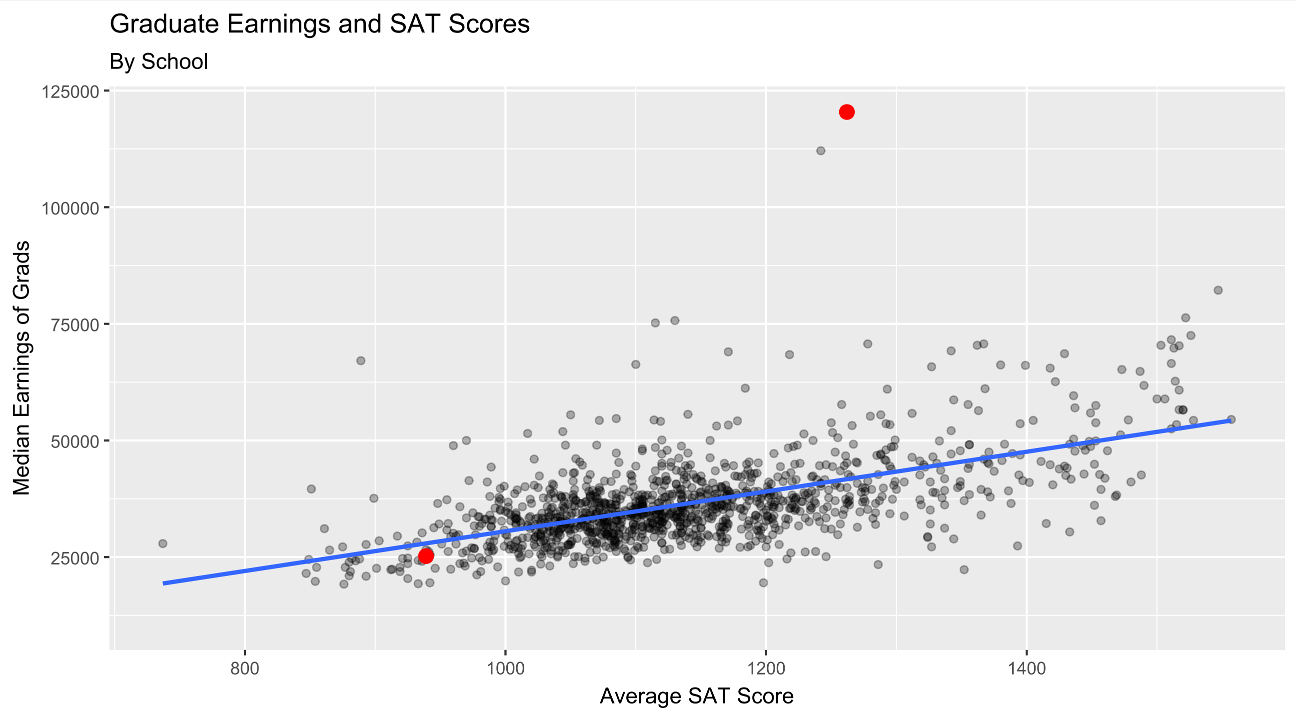
- Let's focus on two schools

```
toplot <- debt %>%
  mutate(hl = ifelse(unitid %in% c(100654,179265),'hl','none')) #
Choosing two examples
p2 <- toplot %>%
  ggplot(aes(x = sat_avg, y = md_earn_wne_p6,color = hl,group =
1,alpha = hl)) +
  geom_point(data = toplot %>% filter(hl == 'none')) +
  geom_point(data = toplot %>% filter(hl == 'hl'),size =3) +
  scale_alpha_manual(values = c(1,.3)) +
  scale_color_manual(values = c('red','black')) +
  geom_smooth(method = 'lm',se = F) +
  theme(legend.position = 'none') +
  labs(title = "Graduate Earnings and SAT Scores",
       subtitle = "By School",
       x = "Average SAT Score",
       y = "Median Earnings of Grads")
```

Step 3: Multivariate Viz

- Adding regression line

```
p2
```



Step 3: Multivariate Viz

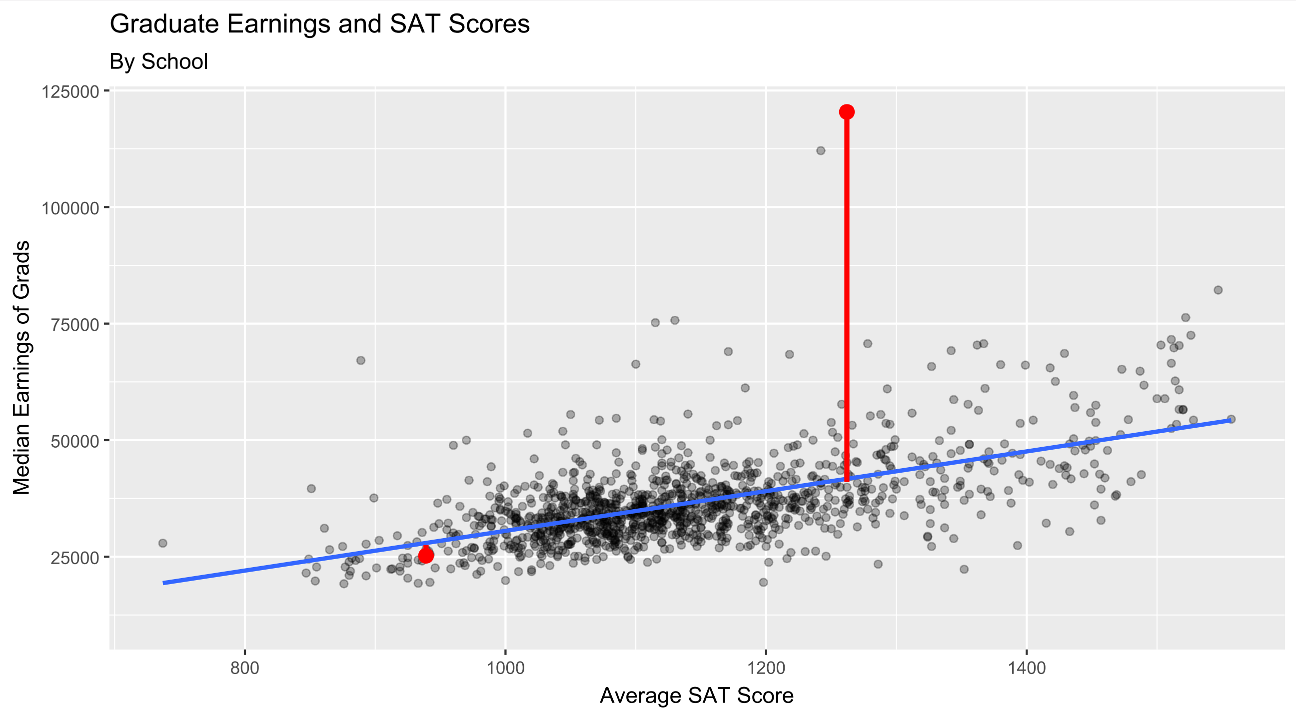
- Defining ε

```
p3 <- toplot %>%
  ggplot(aes(x = sat_avg, y = md_earn_wne_p6,color = hl,group =
1,alpha = hl)) +
  geom_point(data = toplot %>% filter(hl == 'none')) +
  geom_point(data = toplot %>% filter(hl == 'hl'),size =3) +
  scale_alpha_manual(values = c(1,.3)) +
  scale_color_manual(values = c('red','black')) +
  geom_smooth(method = 'lm',se = F) +
  annotate(geom = 'segment',
           x = toplot %>% filter(hl == 'hl') %>% .$sat_avg,
           y = toplot %>% filter(hl == 'hl') %>% .$md_earn_wne_p6,
           xend = toplot %>% filter(hl == 'hl') %>% .$sat_avg,
           yend = c(27500,41000),color = 'red',lwd = 1.2) +
  theme(legend.position = 'none') +
  labs(title = "Graduate Earnings and SAT Scores",
       subtitle = "By School",
       x = "Average SAT Score",
       y = "Median Earnings of Grads")
```

Step 3: Multivariate Viz

- Measuring errors

```
p3
```

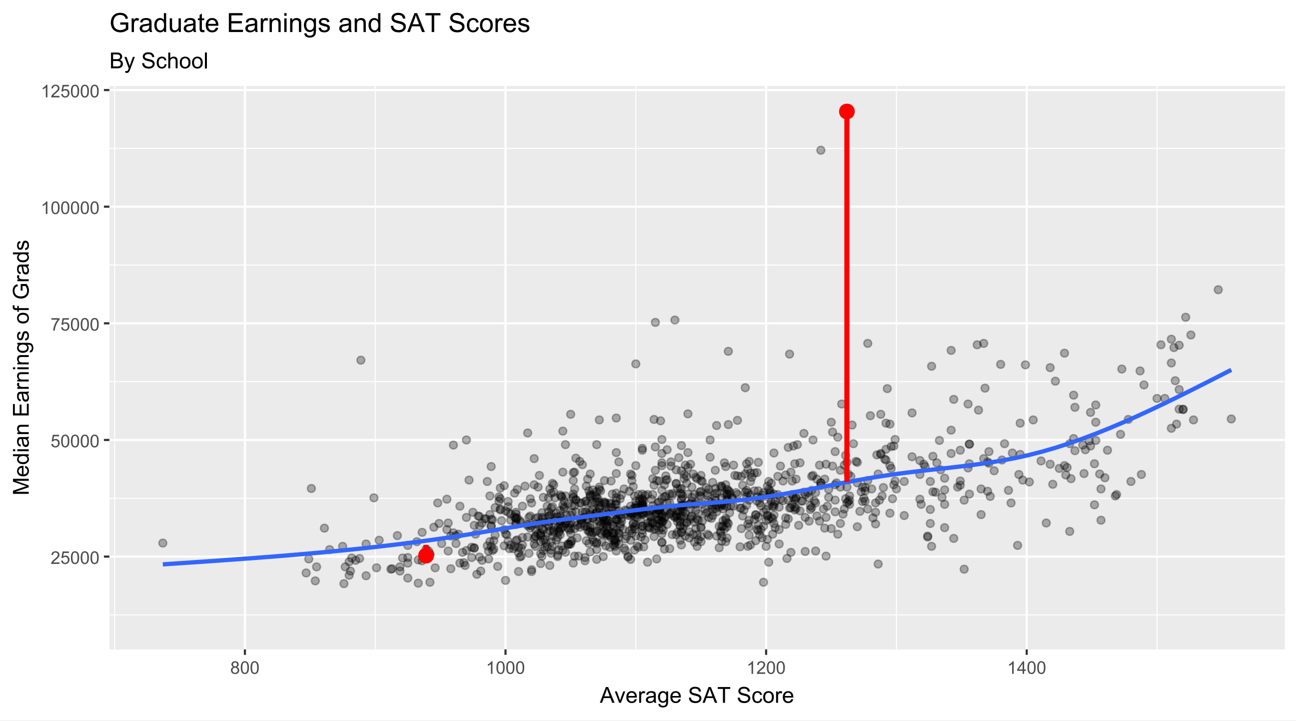


The Data Scientist's Trade-off

- Those mistakes seem pretty big!
- Why not use a curvier line?

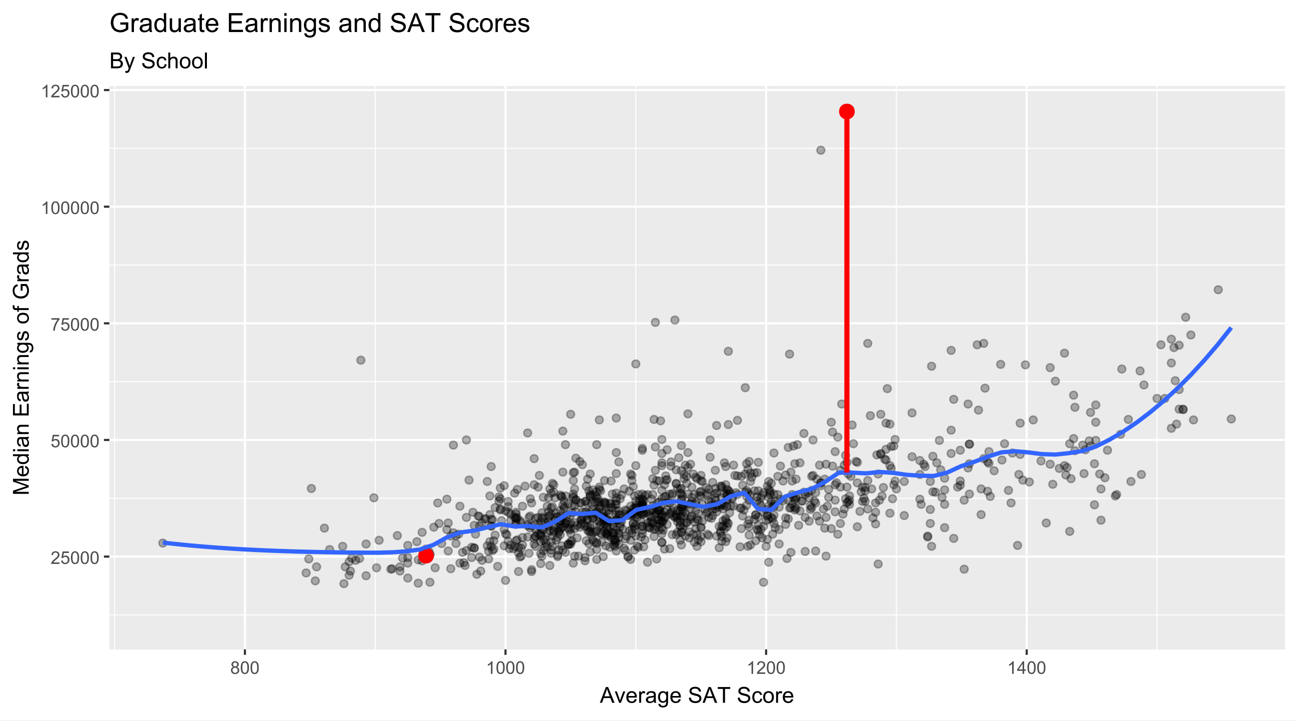
The Data Scientist's Trade-off

- Those mistakes seem pretty big!
- Why not use a curvier line?



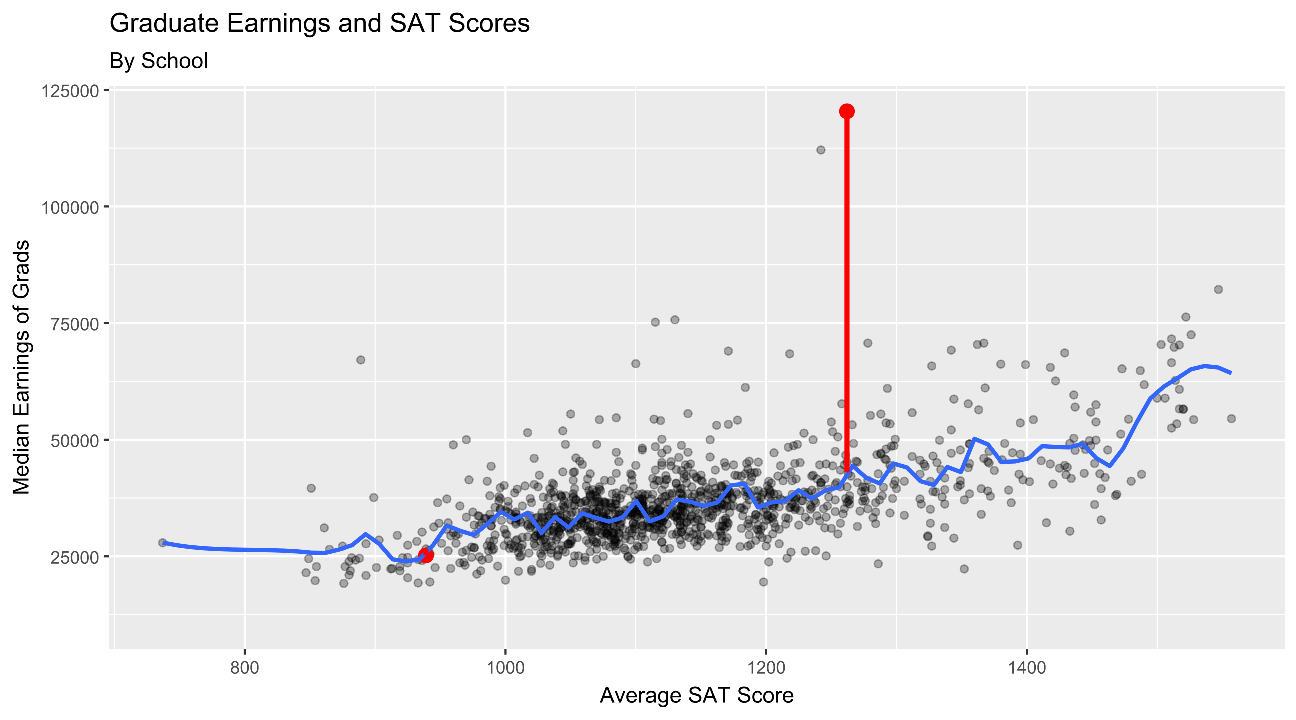
The Data Scientist's Trade-off

- Those mistakes seem pretty big!
- Why not use a curvier line?



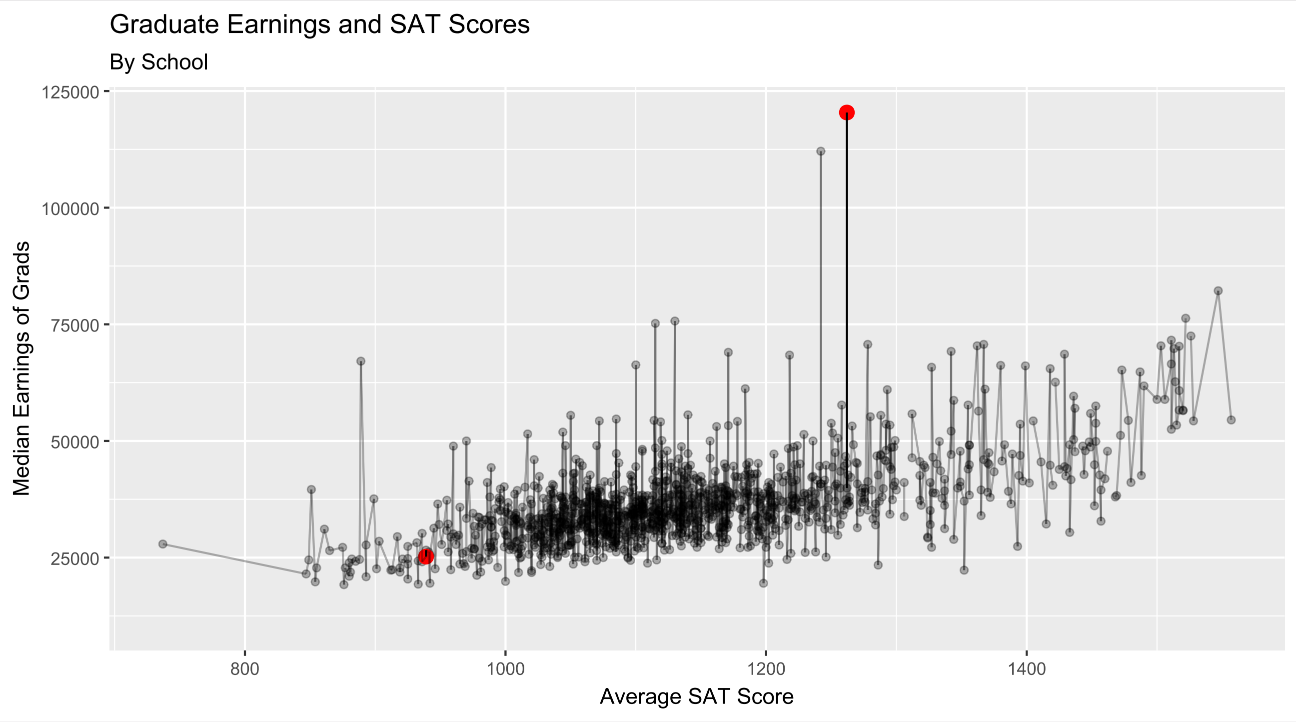
The Data Scientist's Trade-off

- Those mistakes seem pretty big!
- Why not use a curvier line?



The Data Scientist's Trade-off

- Those mistakes seem pretty big!
- Why not use a curvier line?



The Data Scientist's Trade-off

- Want to **reduce complexity**
- But also want to be **accurate**
- What is the right answer?
  - It depends on your theory and the data
  - It is context-dependent
- And this is still only using *linear regression models*!
  - This is a deep area of study, for those interested

Step 4: Regression

- Introducing the `lm(formula,data)` function
- Two inputs to care about:
  - `formula`: Code for $Y = \alpha + \beta X$
  - `data`: What is the data we are using?
- `formula` is written as `Y ~ X`
  - `R` will calculate α and β for us
  - Just need to tell it what is Y (`md_earn_wne_p6`) and X (`sat_avg`)
  - The tilde (`~`) is `R`'s version of the equals sign **in a regression equation**
- Save the model to an object

```
model_earn_sat <- lm(formula = md_earn_wne_p6 ~ sat_avg,data = debt)
```

Step 4: Interpretation

- What is in this object?
- The regression results! Look at them with `summary()`

```
summary(model_earn_sat)
```

```
##
## Call:
## lm(formula = md_earn_wne_p6 ~ sat_avg, data = debt)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -23239  -4311   -852   2893  78695
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -12053.87    1939.80  -6.214 7.12e-10 ***
## sat_avg         42.60       1.69  25.203  < 2e-16 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

Step 4: Interpretation

- Starting with the first column called `Estimate`
- 1st Row `(Intercept)` is α: the predicted value of Y when X is zero
  - Schools with average SAT scores of 0 produce graduates who earn -$12,053.87
  - Sensible?
- 2nd Row `sat_avg` is the β: the increase in Y when X increases by one
  - For each unit increase in the average SAT score, recent graduates earn $42.60 more
  - Sensible?

Step 4: Interpretation

- Other 3 columns?
  - `Std. Error` is the "standard error"
  - `t value` is the "t-statistic"
  - `Pr(>|t|)` is the "p-value"
- t-statistic = Estimate / standard error
- p-value = function(t-statistic)
  - Only really need to remember the p-value for this course
  - This is 1 minus confidence
  - The lower the p-value, the **more** confident we are that the `Estimate` is **not zero**

Step 4: Interpretation

```
summary(model_earn_sat)
```

```
##
## Call:
## lm(formula = md_earn_wne_p6 ~ sat_avg, data = debt)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -23239  -4311   -852   2893  78695
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -12053.87    1939.80  -6.214 7.12e-10 ***
## sat_avg         42.60       1.69  25.203  < 2e-16 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7594 on 1196 degrees of freedom
##   (1348 observations deleted due to missingness)
## Multiple R-squared:  0.3469,    Adjusted R-squared:  0.3463
## F-statistic: 635.2 on 1 and 1196 DF,  p-value: < 2.2e-16
```

Step 4: Interpretation

- Kinda ugly? Use `tidy()` function from the `broom` package

```
require(broom)
```

```
## Loading required package: broom
```

```
tidy(model_earn_sat)
```

```
## # A tibble: 2 × 5
##   term        estimate std.error statistic   p.value
##   <chr>          <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept) -12054.    1940.       -6.21 7.12e- 10
## 2 sat_avg         42.6      1.69     25.2  9.09e-113
```

Another Example

- We will come back to the RMSE after the break
- For now, let's try with a different research question!
- What is the relationship between admissions and future earnings?
  - Theory: More selective schools are more prestigious
  - Hypothesis: There should be a negative relationship between the admissions rate and future earnings

Do It Together!

1. Look at the data and acknowledge missingness
2. Univariate visualization of X and Y
3. Multivariate visualization of X and Y
4. Regression

BREAK

Learning Goals

1. Skew, logs, and coefficients
2. Evaluating a regression: Univariate and multivariate visualization of errors
3. Root Mean Squared Error (RMSE)
4. Cross Validation

Evaluating Regression Results

- Understanding the **errors** helps us evaluate the model
- Define the errors $\varepsilon = Y - \hat{Y}$
  - True outcome values Y
  - Predicted outcome values $\hat{Y}$
- Useful to assess model performance

1. **Look** with univariate and multivariate visualization of the errors
2. Calculate the **RMSE**

Introducing the Data

- New dataset on **movies**
  - `require` `tidyverse`, and `plotly` packages
  - Load `mv.Rds` from Github to object `mv`

```
require(tidyverse)

mv <-
read_rds('https://github.com/jbisbee1/ISP_Data_Science_2024/raw/main/da
```

RQ: Hollywood Finances

- Research Question: What is the relationship between a movie's budget (how much it costs to make a movie) and a movie's earnings (how much money people pay to see the movie in theaters)?
- Theory: More money spent means more famous actors, better special effects, stronger marketing
- Hypothesis: earnings (`gross`) and costs (`budget`) should be **positively** correlated
  - X: ?
  - Y: ?

Follow the process: Look

- TONS of missingness!

```
summary(mv %>% select(gross,budget))
```

```
##       gross                budget         
##  Min.   :7.140e+02   Min.   :     5172  
##  1st Qu.:1.121e+07   1st Qu.: 16865322  
##  Median :5.178e+07   Median : 37212044  
##  Mean   :1.402e+08   Mean   : 57420173  
##  3rd Qu.:1.562e+08   3rd Qu.: 77844746  
##  Max.   :3.553e+09   Max.   :387367903  
##  NA's   :3668        NA's   :4482
```

Missingness

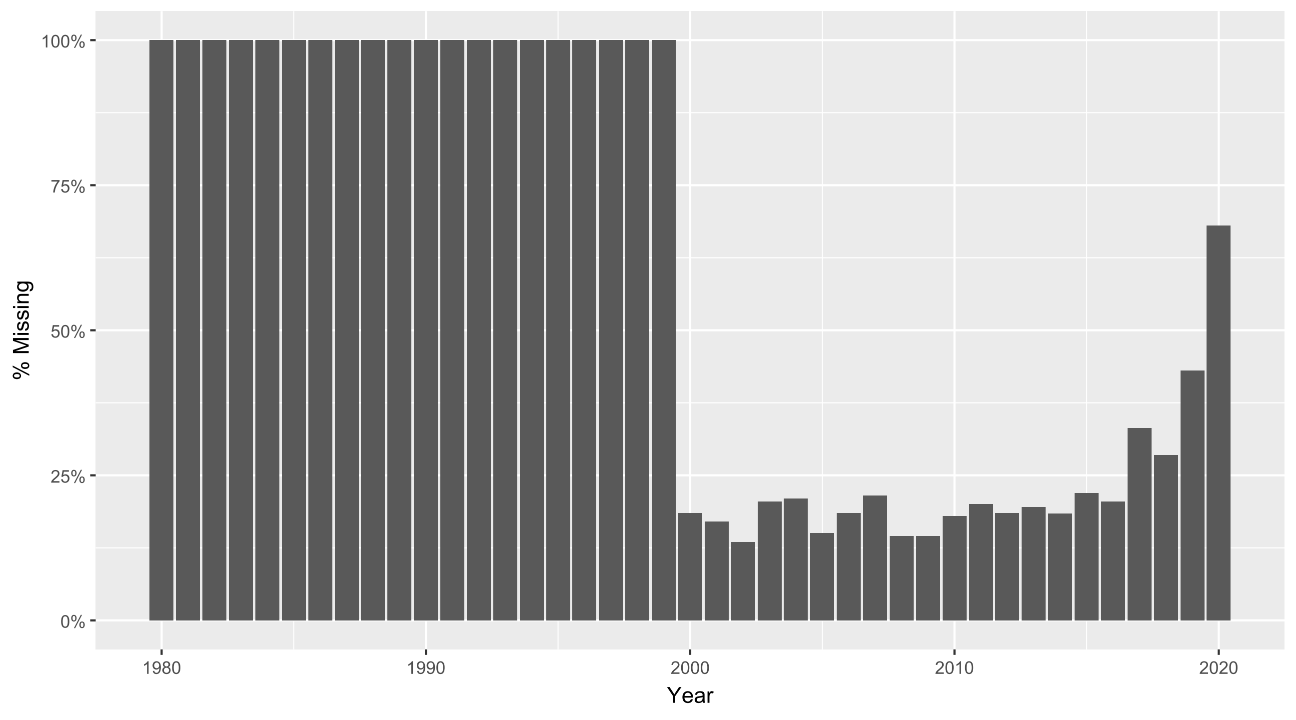
- What does this mean for "generalizability"
  - "Generalizability": Do our conclusions from this data extend ("generalize") to the population at large?

```
p <- mv %>%
  mutate(missing = ifelse(is.na(gross) | is.na(budget),1,0)) %>%
  group_by(year) %>%
  summarise(propMissing = mean(missing)) %>% # Calculate the
proportion of observations missing either gross or budget
  ggplot(aes(x = year,y = propMissing)) +
  geom_bar(stat = 'identity') +
  labs(x = 'Year',y = '% Missing') +
  scale_y_continuous(labels = scales::percent) # Format the y-axis
labels
```

Missingness

- We can only speak to post-2000s Hollywood!

```
p
```



Follow the process: Look

- What **type** of variables are earnings (`gross`) and costs (`budget`)?

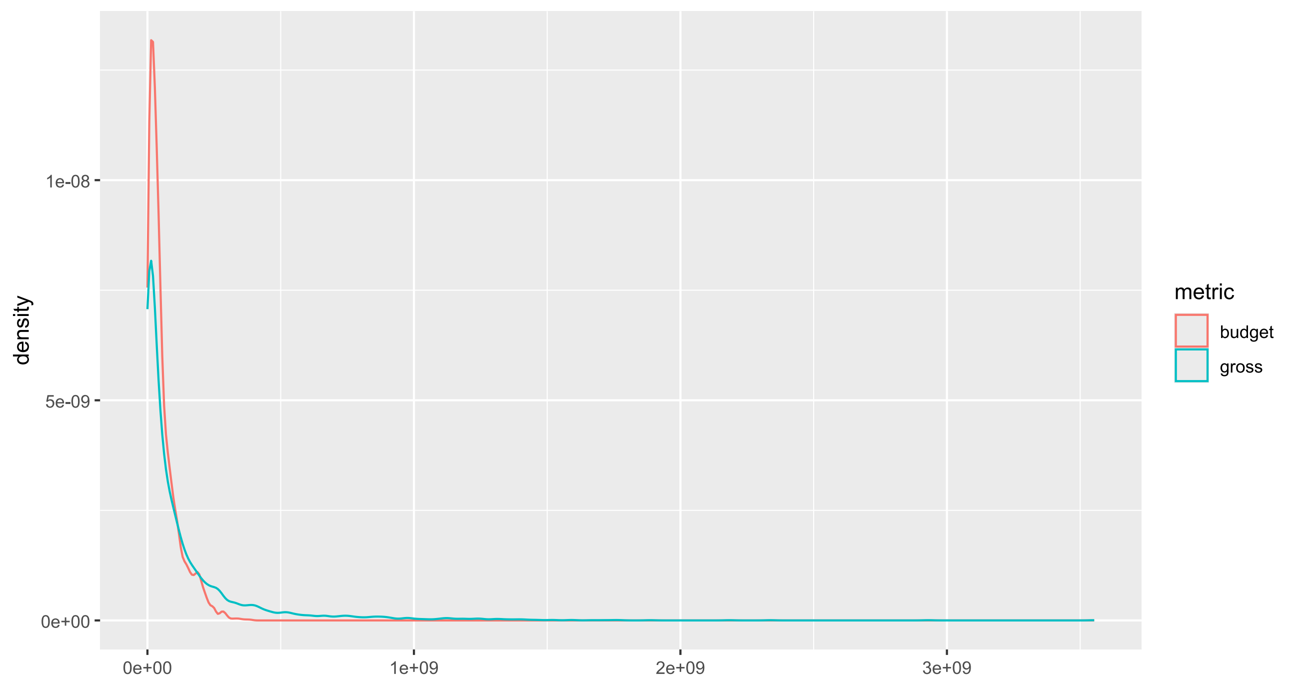
```
mv %>%
  drop_na(gross,budget) %>%
  select(gross,budget) %>% glimpse()
```

```
## Rows: 3,179
## Columns: 2
## $ gross  <dbl> 73677478, 53278578, 723586629, 11490339, 62…
## $ budget <dbl> 93289619, 10883789, 160147179, 6996721, 139…
```

- Looks like continuous measures to me!

2. Univariate Visualization

```
mv %>%
  select(title,gross,budget) %>%
  pivot_longer(names_to = "metric",values_to = "dollars",cols =
c("gross","budget")) %>%
  ggplot(aes(x = dollars,color = metric)) +
  geom_density()
```



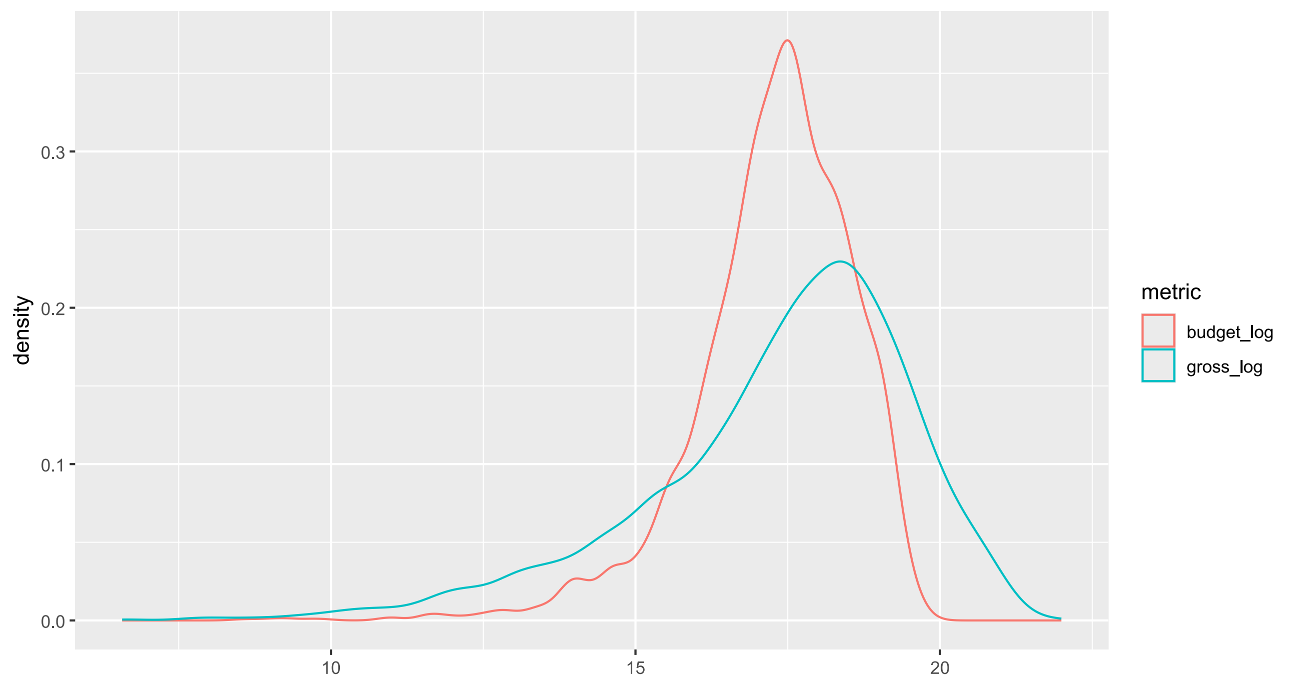
Log and Skew

- Univariate visualization highlights significant **skew** in both measures
  - Most movies don't cost a lot and don't make a lot, but there are a few blockbusters that pull the density way out
- Let's **wrangle** two new variables that take the log of these skewed measures
  - Logging transforms skewed measures to more "normal" measures
  - This is helpful for regression!

```
mv <- mv %>%
  mutate(gross_log = log(gross),
         budget_log = log(budget))
```

2. Univariate Visualization

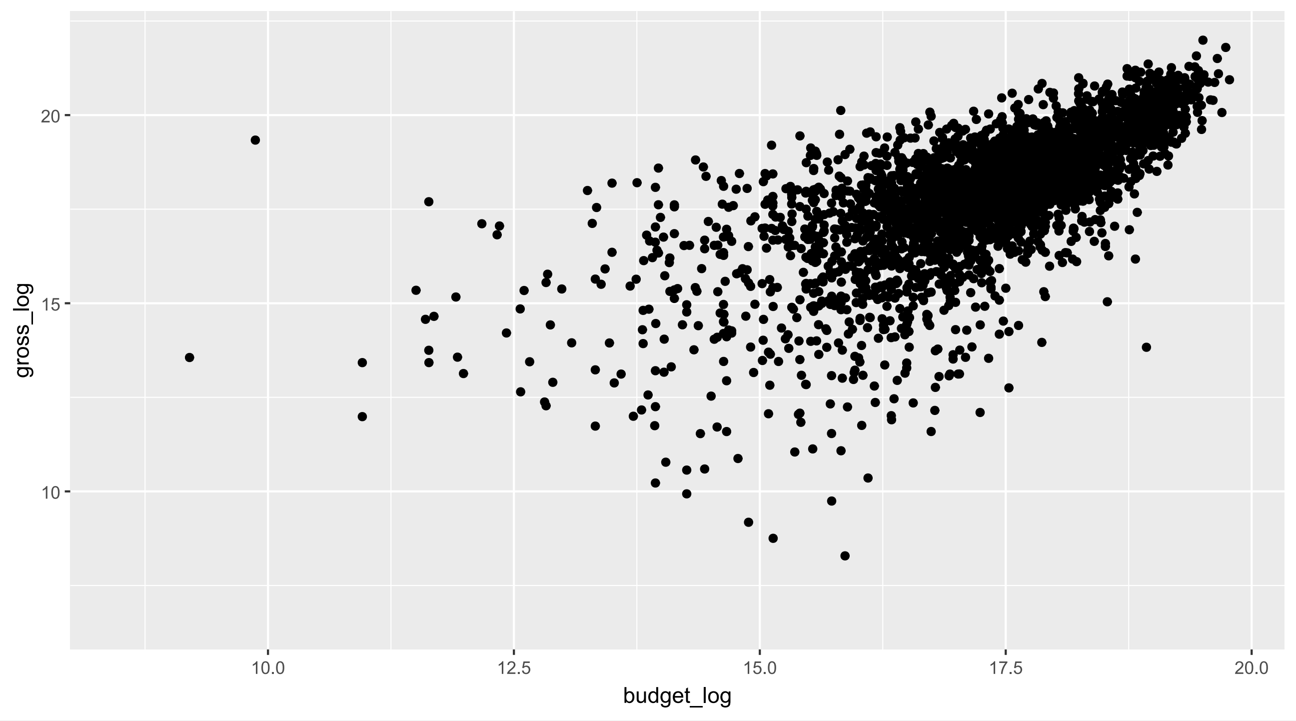
```
mv %>%
  select(title,gross_log,budget_log) %>%
  pivot_longer(names_to = "metric",values_to = "log_dollars",cols =
c("gross_log","budget_log")) %>%
  ggplot(aes(x = log_dollars,color = metric)) +
  geom_density()
```



3. Conditional Analysis

- Continuous X continuous variables? Scatter with `geom_point()`!

```
mv %>%
  ggplot(aes(x = budget_log,y = gross_log)) +
  geom_point()
```



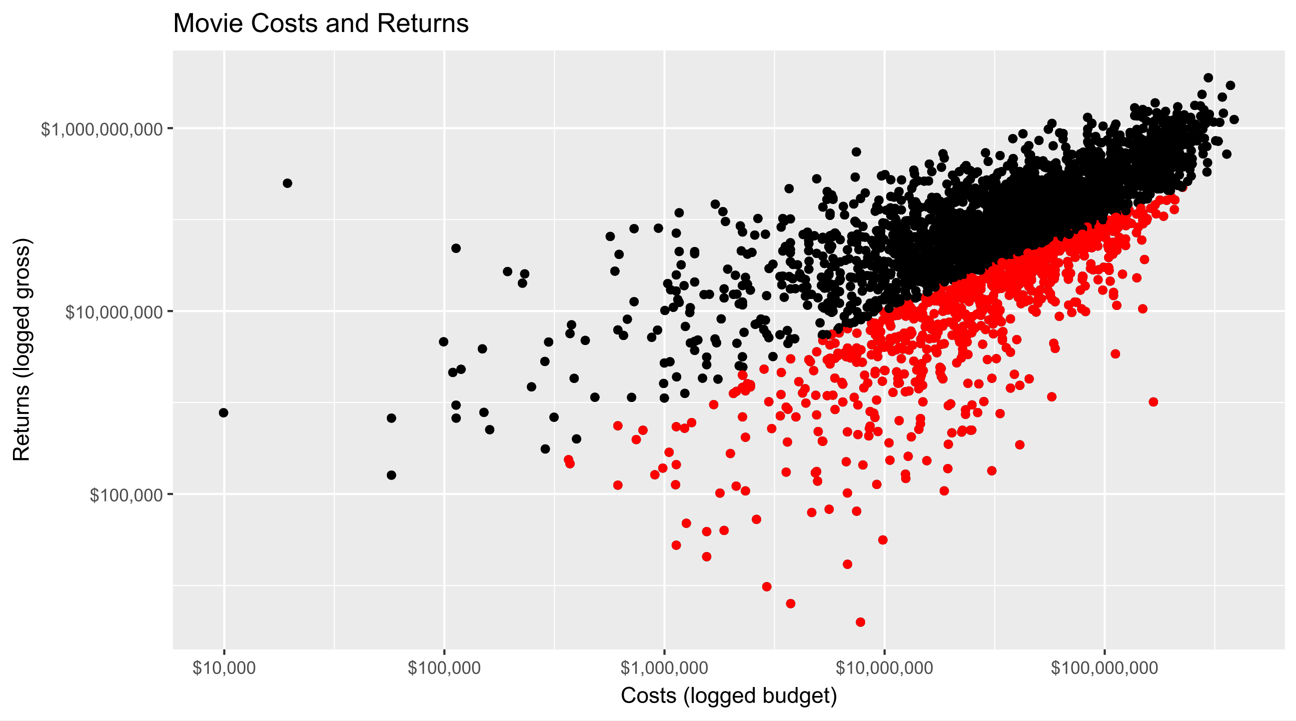
3. Conditional Analysis

- (BTW, I know I've been violating the tenets of data viz for several slides now. Let's fix that.)

```
pSimple <- mv %>%
  drop_na(budget,gross) %>%
  mutate(profitable = ifelse(gross >
budget,'Profitable','Unprofitable')) %>%
  ggplot(aes(x = budget,y = gross,text = paste0(title,' (',genre,',
',year,')'))) +
  geom_point() +
  scale_x_log10(labels = scales::dollar) +
  scale_y_log10(labels = scales::dollar) +
  labs(title = "Movie Costs and Returns",
       x = "Costs (logged budget)",
       y = "Returns (logged gross)")

pFancy <- pSimple + geom_point(aes(color = profitable)) +
  scale_color_manual(guide = 'none',values = rev(c('red','black')))
```

3. Conditional Analysis

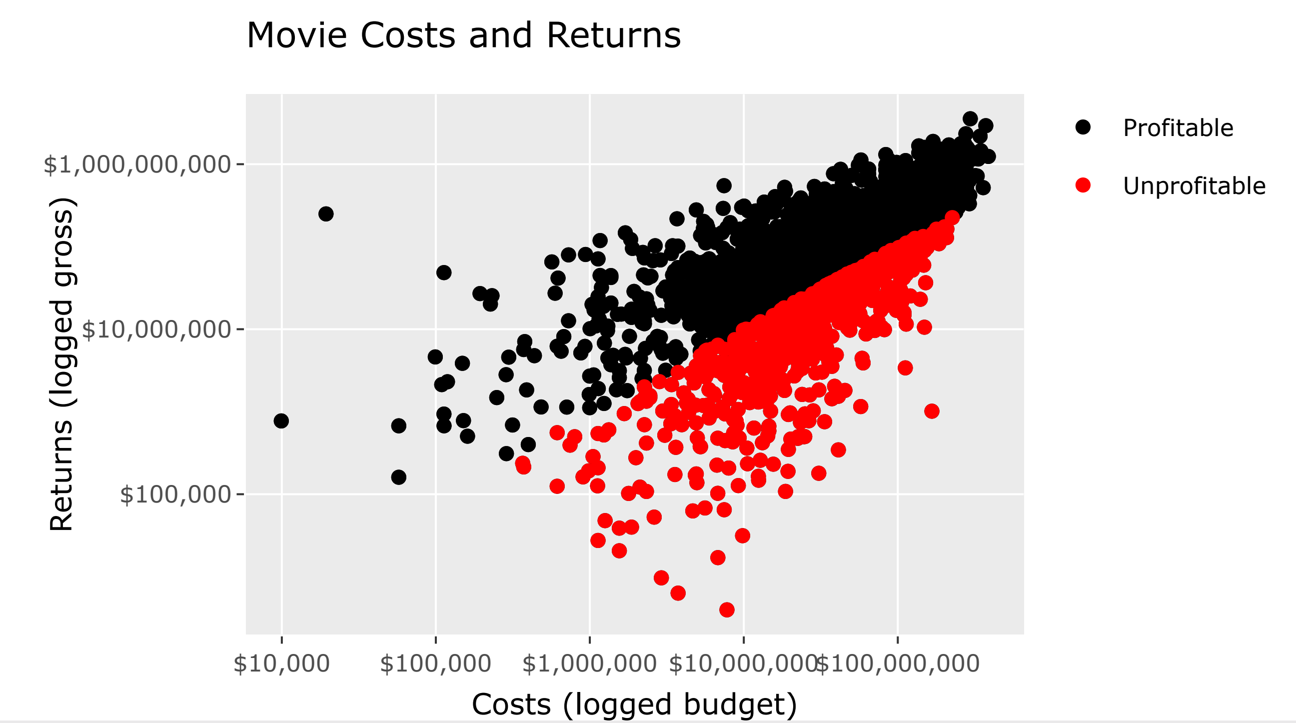
```
pFancy
```



Look with `plotly`

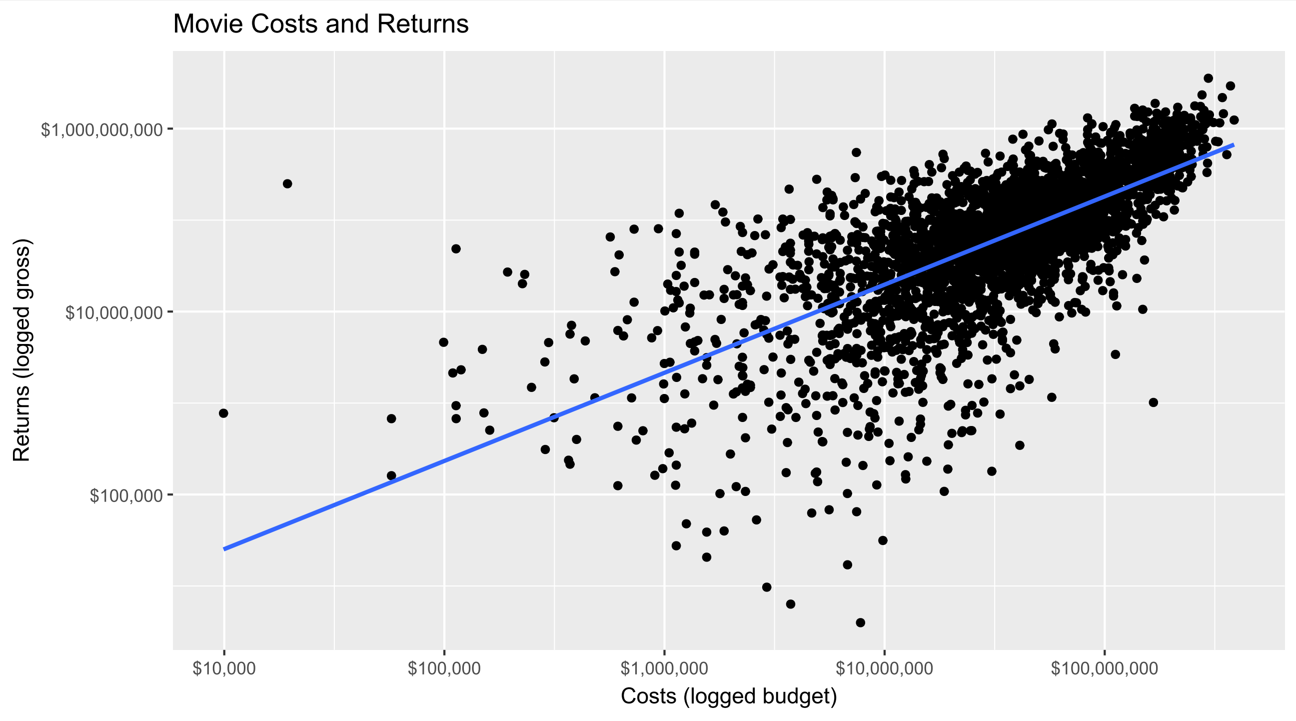
- If curious, can use `plotly` to see outliers

```
require(plotly)
ggplotly(pFancy,tooltip = 'text')
```



4. Regression!

```
pSimple +
  geom_smooth(aes(group = 1),method = 'lm',se = F)
```



4. Regression!

```
m <- lm(gross_log ~ budget_log,data = mv)
tidy(m)
```

```
## # A tibble: 2 × 5
##   term        estimate std.error statistic   p.value
##   <chr>          <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept)    1.26     0.310       4.07 0.0000473
## 2 budget_log     0.964    0.0179     54.0  0
```

Interpretation

- Remember the equation: $Y = \alpha + \beta * X$
- Our Y is logged gross
- Our X is logged budget
- Thus we can rewrite as $gross\_log = \alpha + \beta * budget\_log$
- What is α? What is β?

$$gross\_log = \underbrace{1.26}_{\alpha} + \underbrace{0.96}_{\beta} * budget\_log$$

Interpreting with Logs

- Previously, we said:
  - α is the value of Y when X is zero
  - We need to convert back out of logged values using the `exp()` function
  - When `budget_log` is zero, the budget is `exp(0)` or $1
- Thus, we say when the budget is \$1, the movie makes 1.26 logged dollars, or $3.53

```
exp(1.26107)
```

```
## [1] 3.529196
```

Interpreting with Logs

- For the β coefficient, it depends on where the logged variable appears:

  1. `log(Y) ~ X`: 1 unit change in $X \rightarrow (exp(\beta) - 1) * 100$ percent change in Y

  2. `Y ~ log(X)`: 1% increase in $X \rightarrow \beta/100$ unit change in Y

  3. `log(Y) ~ log(X)`: 1% increase in $X \rightarrow \beta$ percent change in Y

- In our example, a 1% increase in the budget corresponds to a 0.96% increase in gross

- You will either need to memorize these rules, or (like me) just look them up every time

Evaluation

- Every regression line makes mistakes
  - If they didn't, they wouldn't be good at **reducing complexity**!
- How bad do ours look?
  - How should we begin to answer this question!?
- Are there patterns to the mistakes?
  - We **overestimate** gross for movies that cost between $1m and $10m
  - These are the "indies"
  - We also **underestimate** gross to the "blockbusters"
- Why?

Understanding Regression Lines

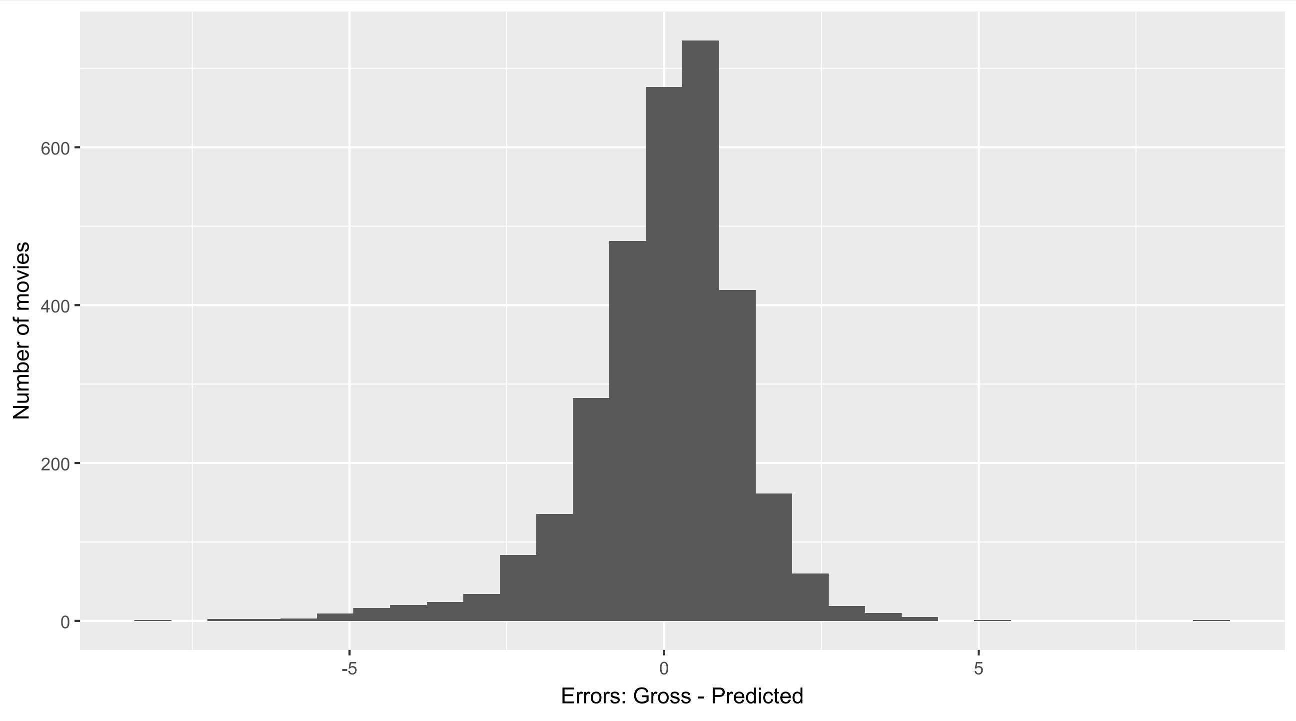
- Regression lines choose α and β to minimize mistakes
  - Mistakes (aka "errors" or "residuals") are captured in the ε term
  - We can apply the **process** to these!

```
# Wrangle data to drop missingness!
mv_analysis <- mv %>% drop_na(gross_log,budget_log)
m <- lm(gross_log ~ budget_log,data = mv_analysis)
mv_analysis$predictions <- predict(m)
mv_analysis$errors <- mv_analysis$gross_log - mv_analysis$predictions

summary(mv_analysis$errors)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -8.2672 -0.6354  0.1648  0.0000  0.7899  8.5599
```

Univariate Viz of Errors

```
mv_analysis %>%
  ggplot(aes(x = errors)) +
  geom_histogram() +
  labs(x = 'Errors: Gross - Predicted',y = 'Number of movies')
```



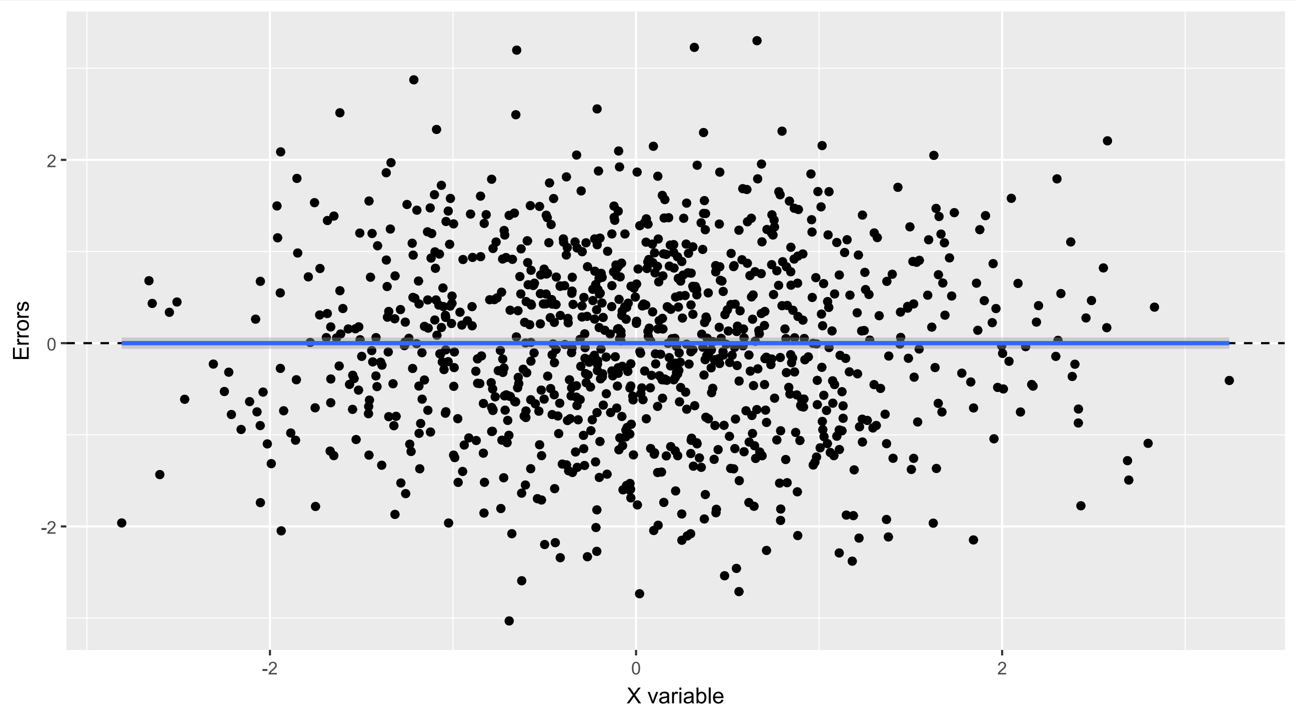
Univariate Viz of Errors

- Note that they are on average zero
  - Don't feel too proud! Mean 0 error is baked into the method
  - More concerned about **skew**...there is evidence of overestimating
- Can we do more? **Conditional Analysis**
  - Conditional on the **predictor** (the X variable)

Multivariate Viz of Errors

- Ideal is where errors are unrelated to predictor
  - This **should** appear as a rectangular cloud of points around zero

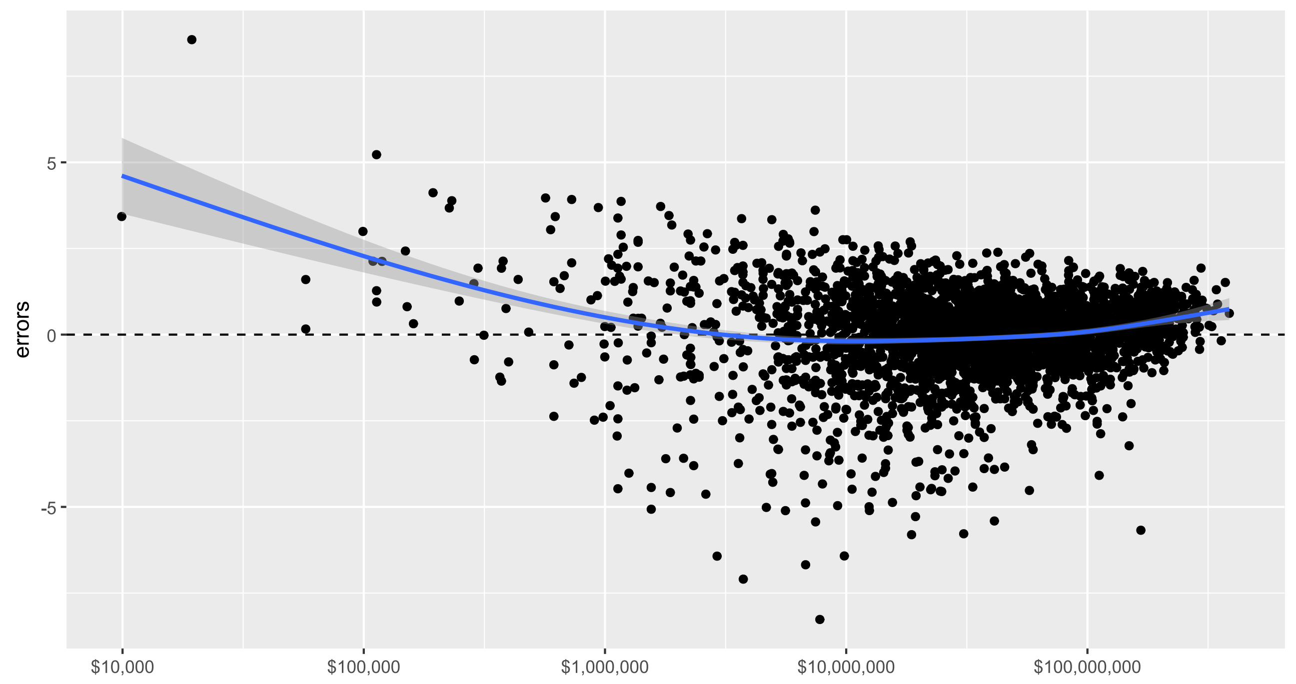
```
pIdeal
```



Multivariate Viz of Errors

- This is not the case for us!

```
mv_analysis %>%
  ggplot(aes(x = budget,y = errors)) +
  geom_point() + geom_hline(yintercept = 0,linetype = 'dashed') +
  scale_x_log10(label = scales::dollar) + geom_smooth()
```



Multivariate Viz of Errors

- Evidence of a U-shape → underpredict low and high budgets, overpredict middle budgets
- Ergo, our model is **not great**!
  - Could add additional predictors X_2, X_3, etc.
  - Next lecture!

RMSE

- Univariate / Multivariate visualization of errors is **important**
- But we want to summarize model quality in a simpler way
- **RMSE**: summarizes model performance with a *single number*
  - Useful for comparing multiple models to each other

RMSE

- **E**rror (ε): actual outcome (Y_i) - predicted outcome ($\hat{Y}_i$)
    - The "distance" between the data and the model
- **S**quared: ε^2
    1. Makes all values positive
    2. Exaggerates the presence of larger errors
- **M**ean: average these squared errors
- **R**oot: take their square root (un-exaggerate)

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2}$$

RMSE

- **E**rror (ε): actual outcome (Y_i) - predicted outcome ($\hat{Y}_i$)
    - The "distance" between the data and the model
- **S**quared: ε^2
    1. Makes all values positive
    2. Exaggerates the presence of larger errors
- **M**ean: average these squared errors
- **R**oot: take their square root (un-exaggerate)

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\underbrace{Y_i - \hat{Y}_i}_{\varepsilon})^2}$$

RMSE

- **E**rror (ε): actual outcome (Y_i) - predicted outcome ($\hat{Y}_i$)
    - The "distance" between the data and the model
- **S**quared: ε^2
    1. Makes all values positive
    2. Exaggerates the presence of larger errors
- **M**ean: average these squared errors
- **R**oot: take their square root (un-exaggerate)

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\underbrace{(\varepsilon)^2}_{S}}$$

RMSE

- **E**rror (ε): actual outcome (Y_i) - predicted outcome ($\hat{Y}_i$)
  - The "distance" between the data and the model
- **S**quared: ε^2
  1. Makes all values positive
  2. Exaggerates the presence of larger errors
- **M**ean: average these squared errors
- **R**oot: take their square root (un-exaggerate)

$$RMSE = \sqrt{\underbrace{\frac{1}{n}\sum_{i=1}^{n}}_{M}(SE)}$$

RMSE

- **E**rror (ε): actual outcome (Y_i) - predicted outcome ($\hat{Y}_i$)
  - The "distance" between the data and the model
- **S**quared: ε^2
  1. Makes all values positive
  2. Exaggerates the presence of larger errors
- **M**ean: average these squared errors
- **R**oot: take their square root (un-exaggerate)

$$RMSE = \sqrt{(MSE)}$$

RMSE

- RMSE is a **single measure that summarizes model performance**

```
e <- mv_analysis$gross_log - mv_analysis$predictions
se <- e^2
mse <- mean(se)
rmse <- sqrt(mse)
# Or
(rmseBudget <- sqrt(mean(mv_analysis$errors^2)))
```

```
## [1] 1.280835
```

- Is this good?

Predicting with uncertainty

- Say we're talking to investors about a new movie that costs $10m
  - How do we plug 10m into our model?

```
summary(m)$coefficients
```

```
##                Estimate Std. Error   t value  Pr(>|t|)
## (Intercept) 0.03776401 0.03153666  1.197464 0.2314101
## X           2.07852434 0.03152731 65.927738 0.0000000
```

- $\hat{Y}_i = \alpha + \beta * X$
  - $\alpha = 1.26$ and $\beta = 0.96$
  - where $\hat{Y}_i$ is predicted gross (log) and X is $10m budget (log)

```
pred_gross_log <- 1.26 + 0.96*log(1e7)
```

Predicted Gross

- Again, convert back out of logged values with `exp()`

```
scales::dollar(exp(pred_gross_log))
```

```
## [1] "$18,501,675"
```

- Cool! We'll make $8.5m!
  - But we know our model isn't perfect
  - Need to adjust for it's errors via **RMSE**

Incorporating RMSE

- Simple idea: add and subtract RMSE from this prediction

```
pred_gross_log_ub <- 1.26 + 0.96*log(1e7) + rmseBudget
pred_gross_log_lb <- 1.26 + 0.96*log(1e7) - rmseBudget
scales::dollar(exp(c(pred_gross_log_ub,pred_gross_log_lb)))
```

```
## [1] "$66,599,457" "$5,139,861"
```

- So we'll either make a $56m profit or we'll lose almost $5m?
- **CONCLUSION PART 2**: maybe our model isn't very good?

Introducing Cross Validation

- We ran a model on the full data and calculated the RMSE
- But this approach risks "overfitting"
  - Overfitting is when we get a model that happens to do well on our specific data, but isn't actually that useful for predicting elsewhere.
  - "Elsewhere": Other periods, other movies, other datasets
- Theory: Why care about **external validity**?
  - What is the point of measuring relationship if they don't generalize?

Introducing Cross Validation

- In order to avoid overfitting, we want to "train" our model on one part of the data, and then "test" it on a different part of the data.
  - Model "can't see" the test data → better way to evaluate performance
- Cross Validation: randomly split our data into a train set and test set
  - *Similar to bootstrapping*

Introducing Cross Validation (CV)

```
set.seed(1021)
# Sample our data WITHOUT replacement
train <- mv_analysis %>%
  sample_n(size = round(nrow(.)*.5),
           replace = F)

# New function...remove all the rows that are the same as train
test <- mv_analysis %>%
  anti_join(train)
```

```
## Joining with `by = join_by(title, rating, genre, year,
## released, score, votes, director, writer, star, country,
## budget, gross, company, runtime, id, imdb_id,
## bechdel_score, boxoffice_a, language, gross_log,
## budget_log, predictions, errors)`
```

- We now have two datasets of roughly the same number of observations, but none of them are the same!

CV to Calculate RMSE

- We want to estimate a model based on the **test** data
- And evaluate RMSE based on the **train** data

```
m2 <- lm(gross_log ~ budget_log,train)

# predict() function on a new dataset
test$preds <- predict(m2,newdata = test)

# Now calculate RMSE on the new dataset
e <- test$gross_log - test$preds
se <- e^2
mse <- mean(se,na.rm=T)
rmse <- sqrt(mse)
rmse
```

```
## [1] 1.28959
```

CV to Calculate RMSE

- We did worse with CV! This is a *feature*
  - We are not being overconfident
  - We are avoiding "overfitting"
- Want to do this many times (like bootstrapping)

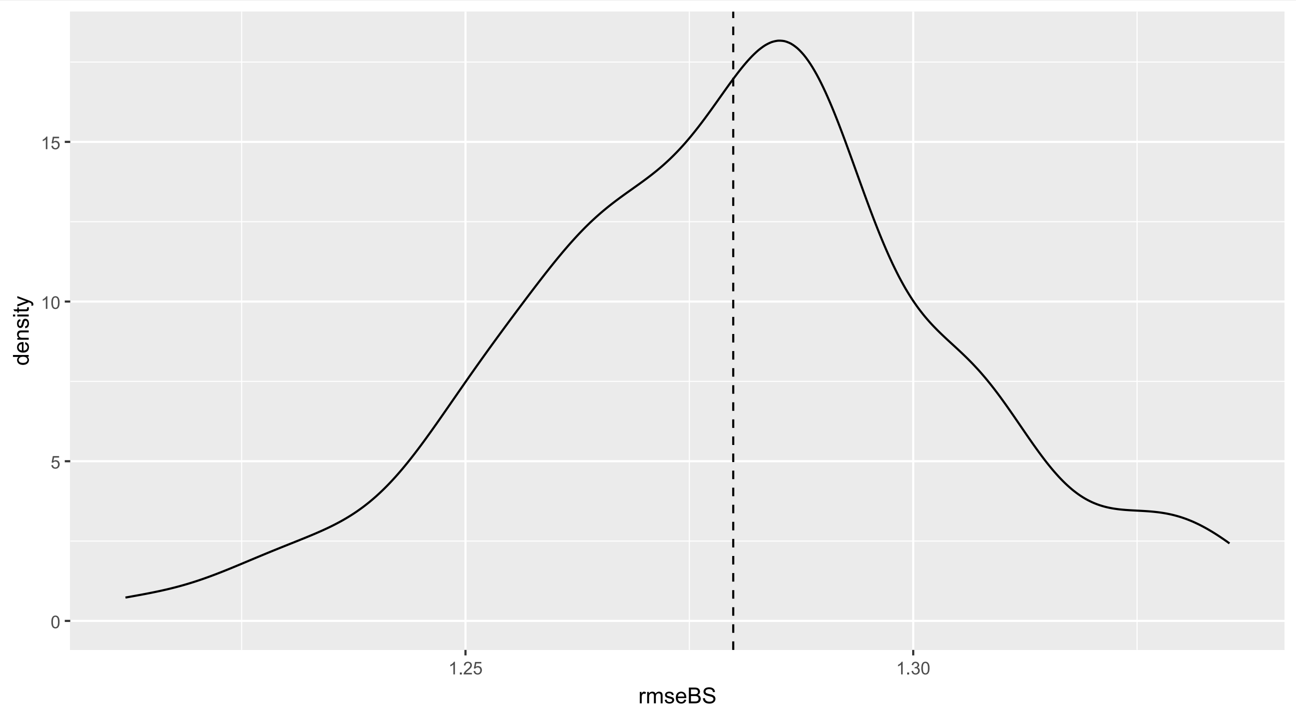
CV to Calculate RMSE

```
set.seed(123)
bsRes <- NULL
for(i in 1:100) {
  # Create training dataset
  train <- mv_analysis %>%
    sample_n(size = round(nrow(.)*.5),
             replace = F)

  # Create test dataset
  test <- mv_analysis %>%
    anti_join(train)

  mTrain <- lm(gross_log ~ budget_log,train)

  test$preds <- predict(mTrain,newdata = test)

  rmse <- sqrt(mean((test$gross_log - test$preds)^2,na.rm=T))
  bsRes <- c(bsRes,rmse)
}
mean(bsRes)
```

```
## [1] 1.279899
```

CV to Calculate RMSE

```
data.frame(rmseBS = bsRes) %>%
  ggplot(aes(x = rmseBS)) +
  geom_density() +
  geom_vline(xintercept = mean(bsRes),linetype = 'dashed')
```



Cross Validation

- In this example, we used a 50-50 split
- Often, data scientists prefer an 80-20 split
  - **Improves** the model (80% of the data is more to learn from)...
  - ...but still **protects** against overfitting

```
train <- mv_analysis %>%
  sample_n(size = round(nrow(mv_analysis)*.8),
           replace = F)
```