

# Regression

## Part 2

Prof. Bisbee

Vanderbilt University

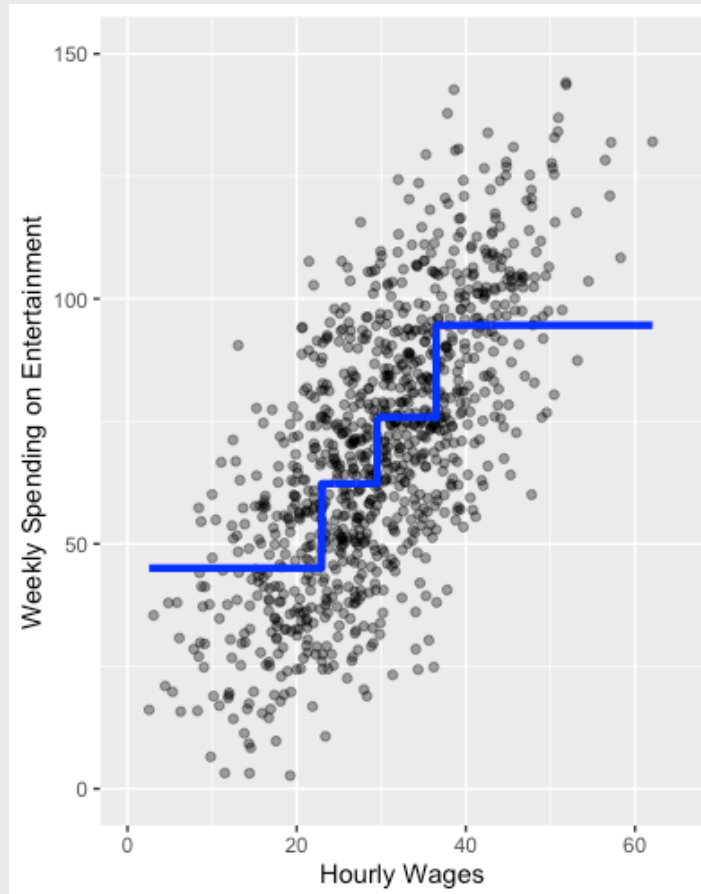
Slides Updated: 2024-08-10

# 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

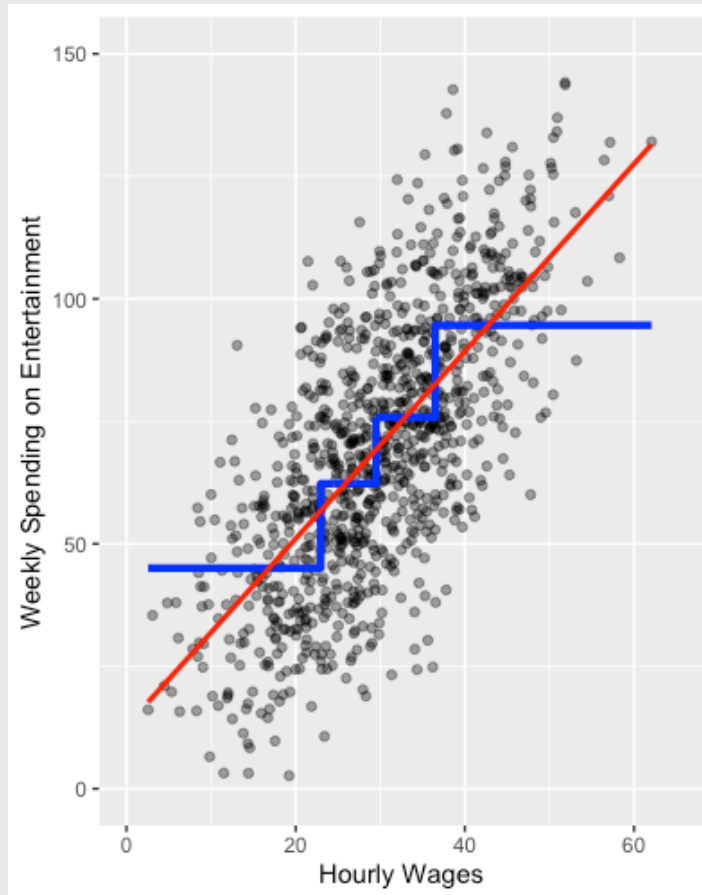
# Regression Recap

- Regression very similar to **conditional means**



# Regression Recap

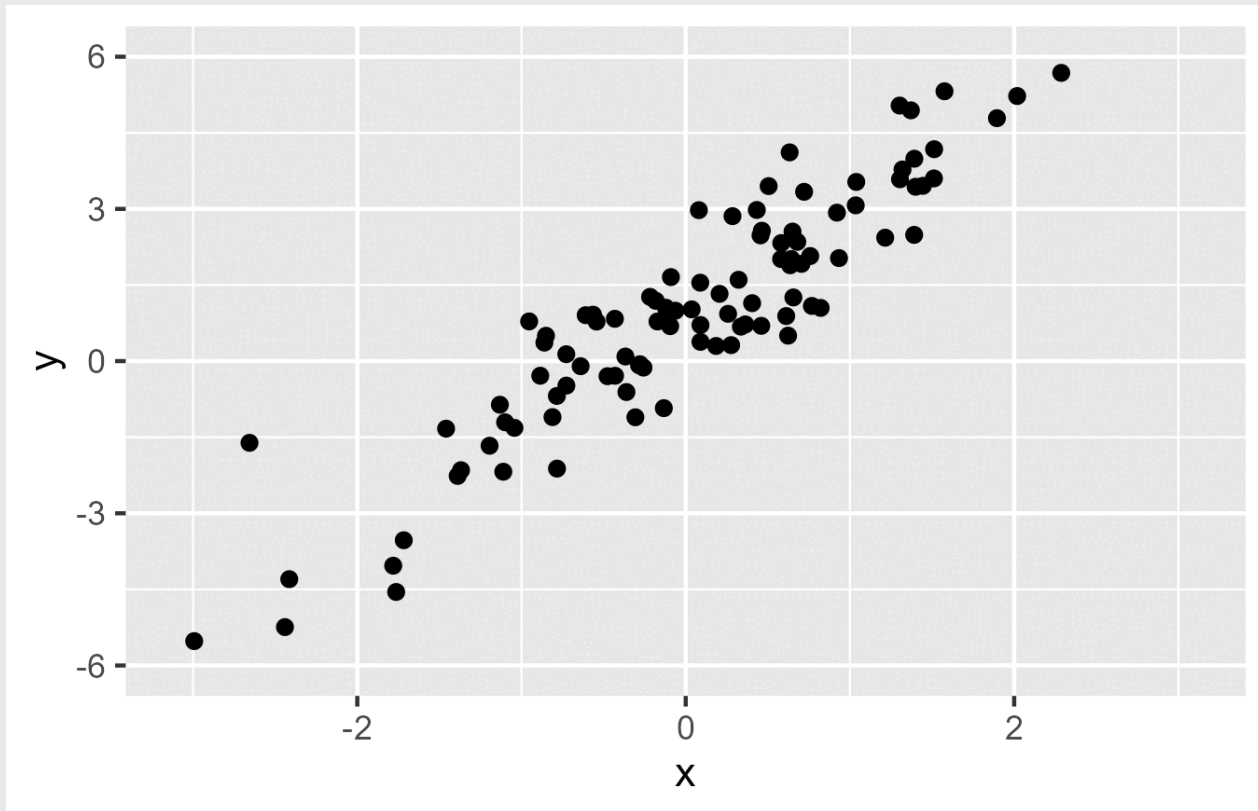
- Regression very similar to **conditional means**



# Regression Recap

- For this class, don't need to know how it happens
  - But the intuition is obvious
  - Given  $Y = \alpha + \beta X$ , just tweak  $\alpha$  and  $\beta$  to reduce **errors**
  - Once you've minimized all the errors, you have the **line of best fit**

# Visual Intuition



# 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**
  - Download `mv.Rds` to your `data` folder and load to object `mv`
  - `require tidyverse`, and `plotly` packages

```
require(tidyverse)

mv <-
read_rds('https://github.com/jbisbee1/DS1000_F2024/raw/main/data/mv.Rds')
```



# 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

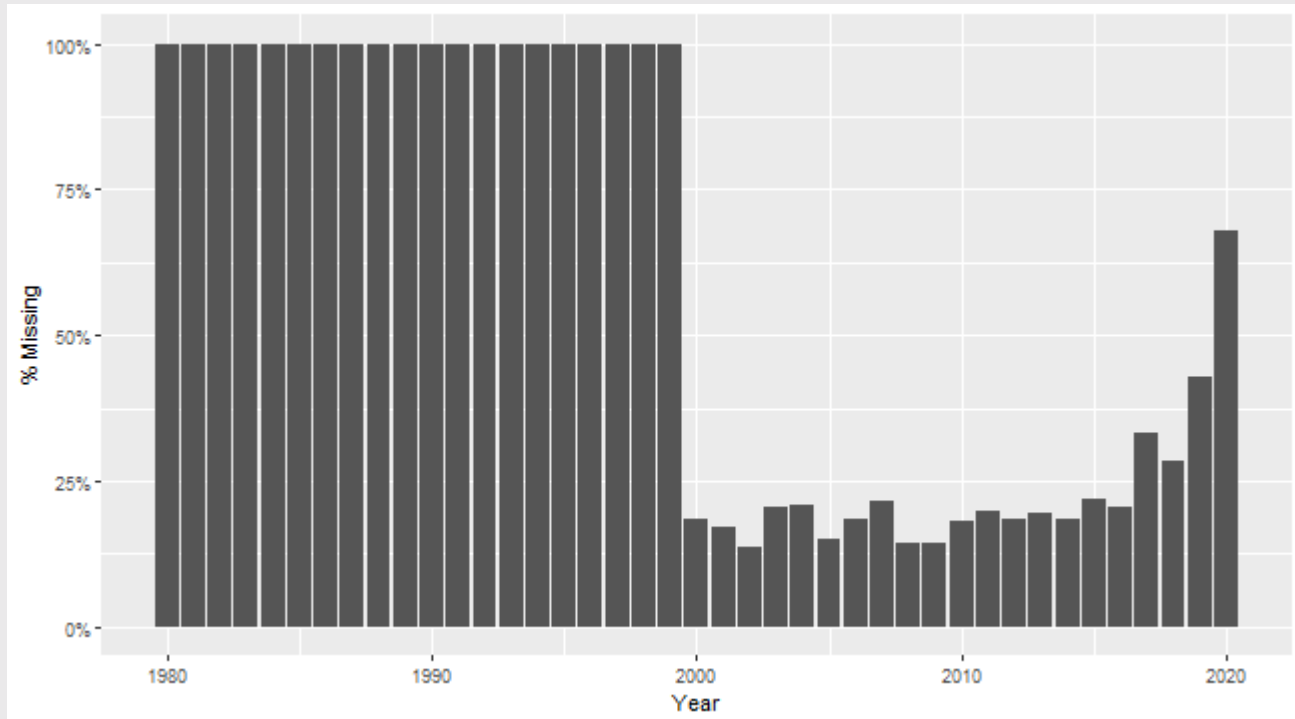
- What does this mean for "generalizability"
  - "Generalizability": Do our conclusions from this data extend ("generalize") to the population at large?

```
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**)?

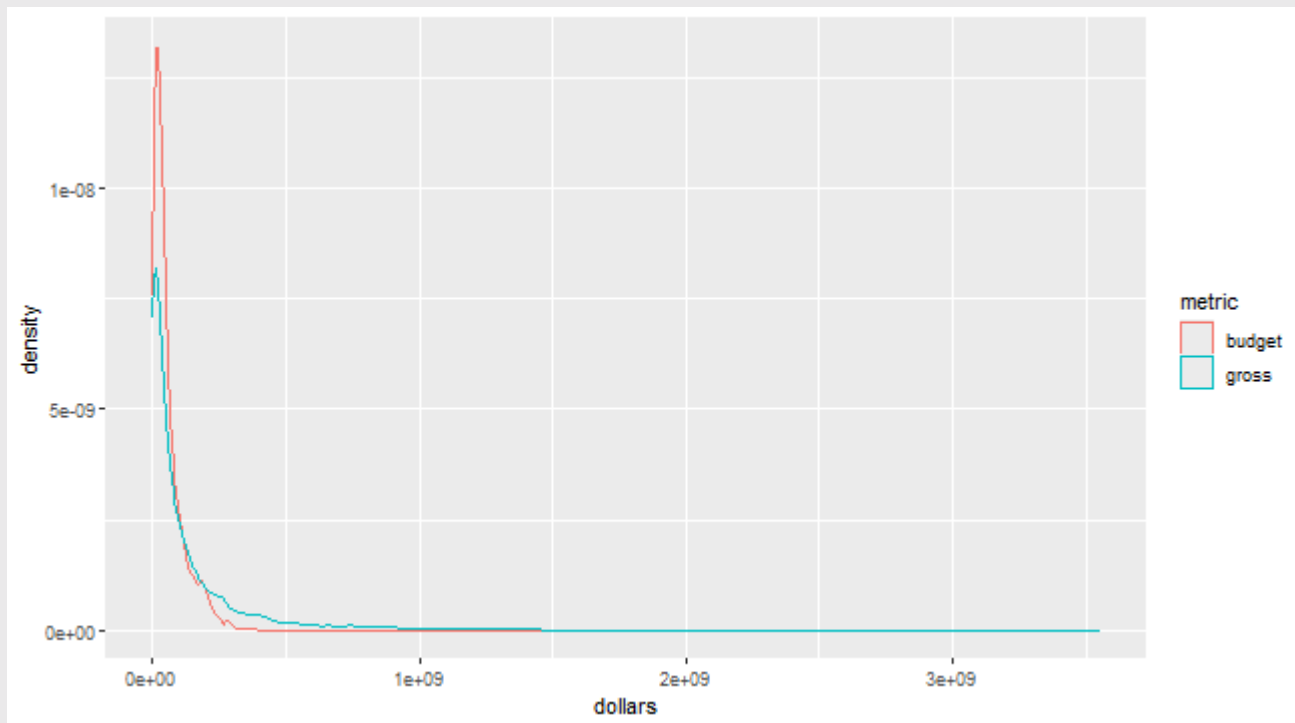
```
mv %>%  
  drop_na(gross,budget) %>%  
  select(gross,budget) %>% glimpse()
```

```
## 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) %>%  
  gather(metric,dollars,-title) %>%  
  ggplot(aes(x = dollars,color = metric)) +  
  geom_density()
```



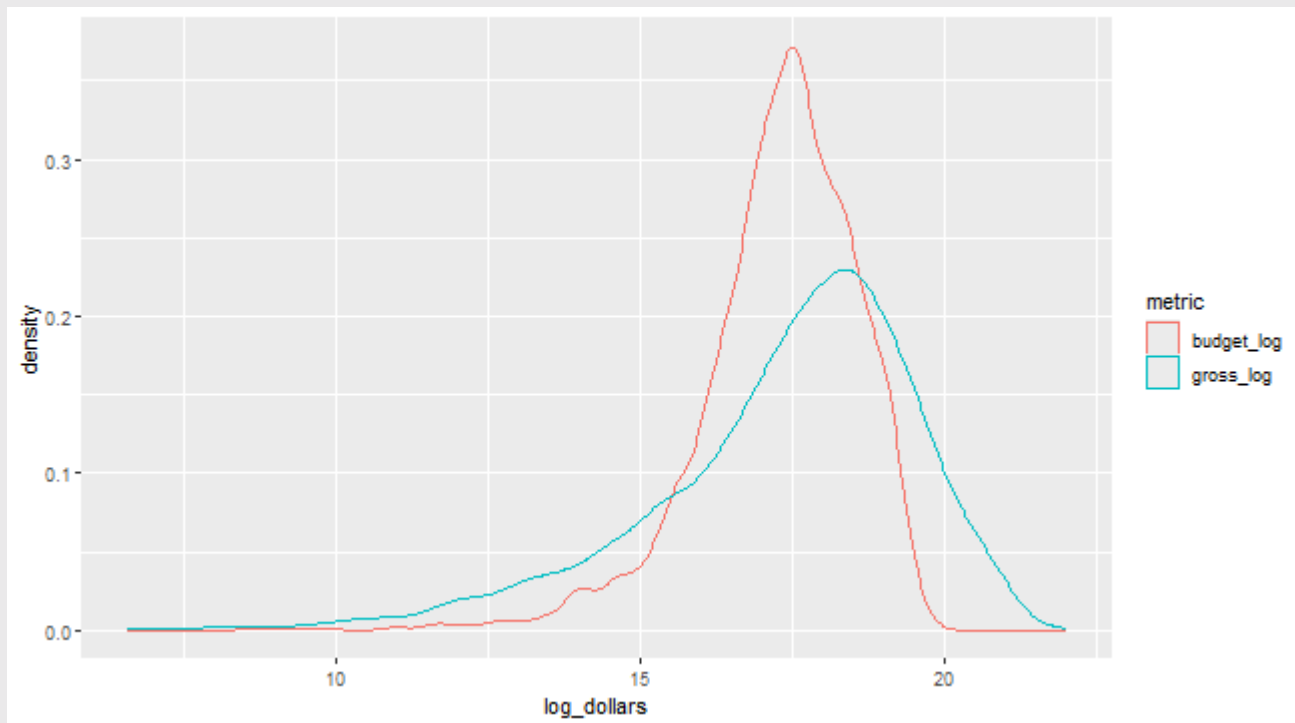
# Log and Skew

- 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) %>%  
  gather(metric,log_dollars,-title) %>%  
  ggplot(aes(x = log_dollars,color = metric)) +  
  geom_density()
```

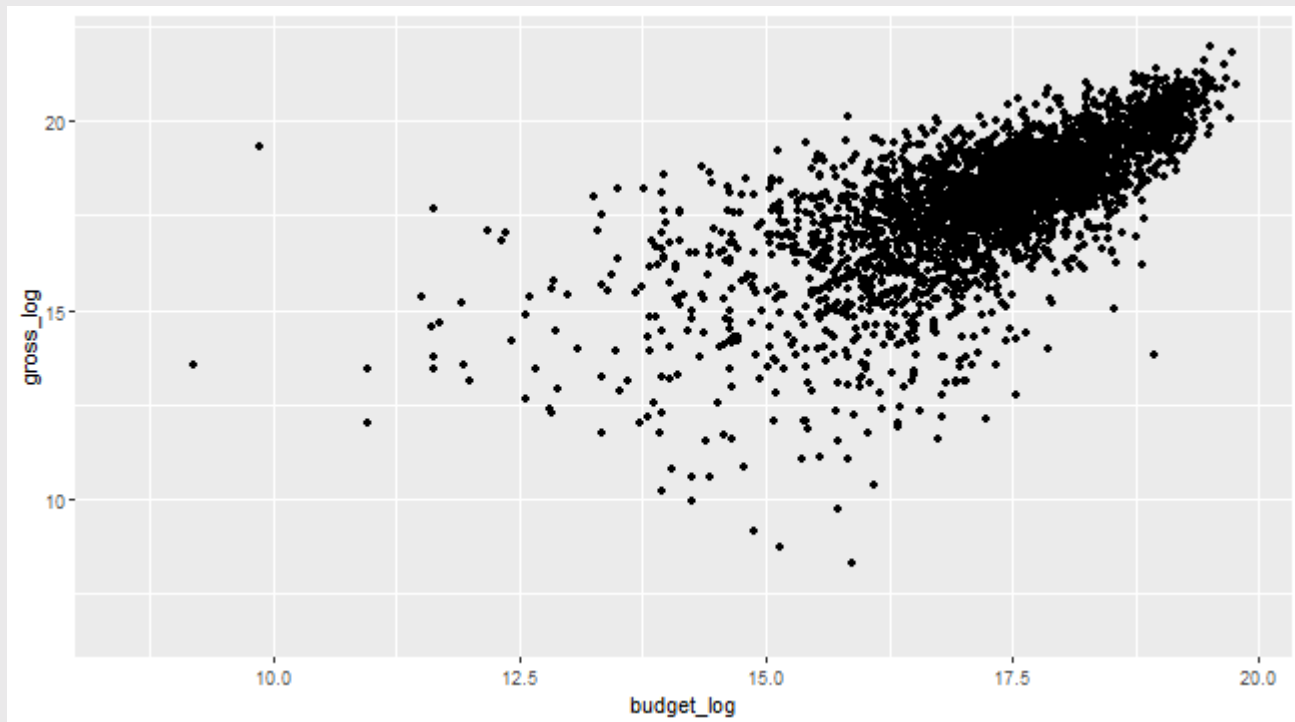




# 3. Conditional Analysis

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

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



# 3. Conditional Analysis

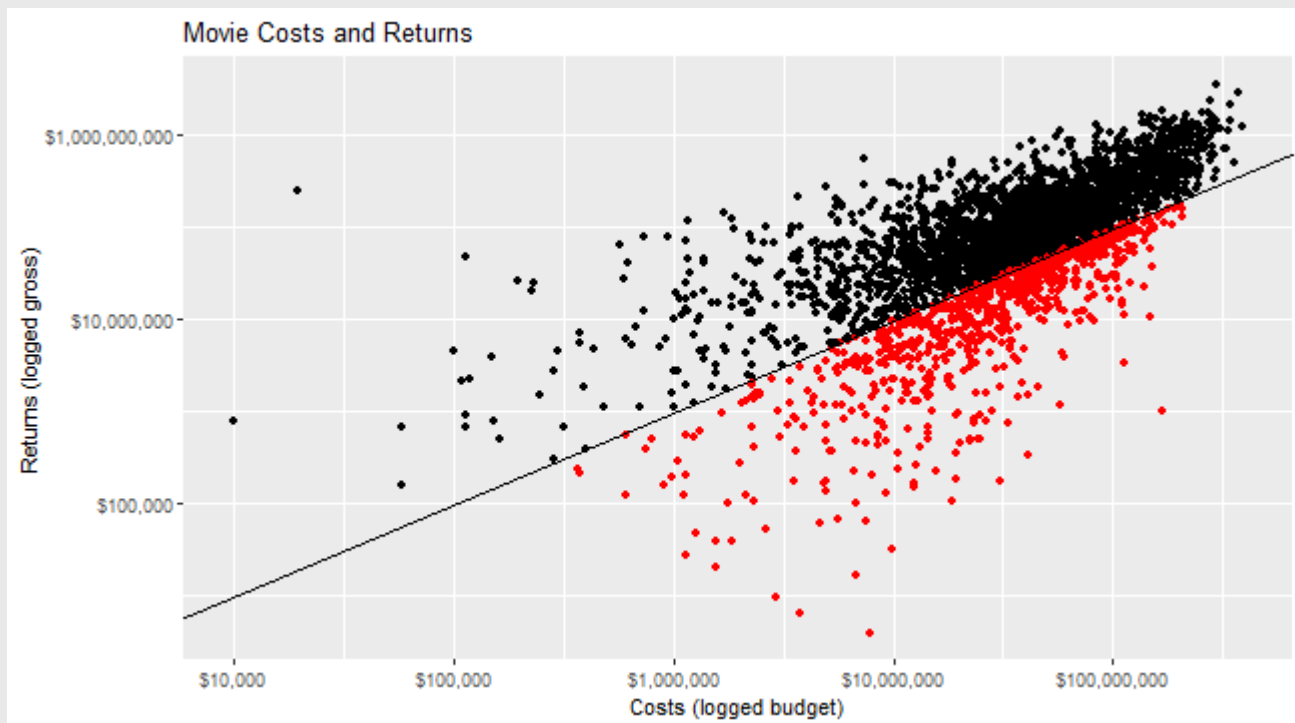
- (BTW, I know I've been violating the tenets of data viz for several slides now. Let's fix that.)

```
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')))) +
  geom_abline(intercept = 0,slope = 1)
```

# 3. Conditional Analysis

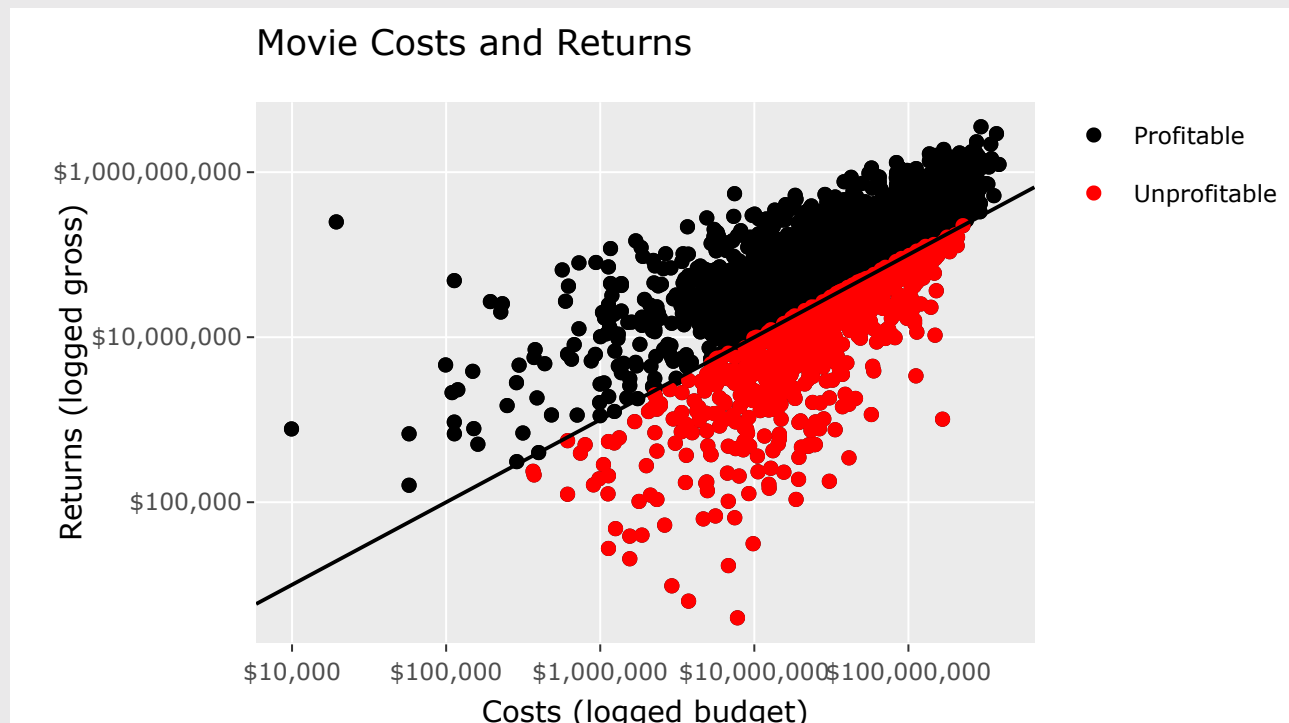
pFancy



# Look with `plotly`

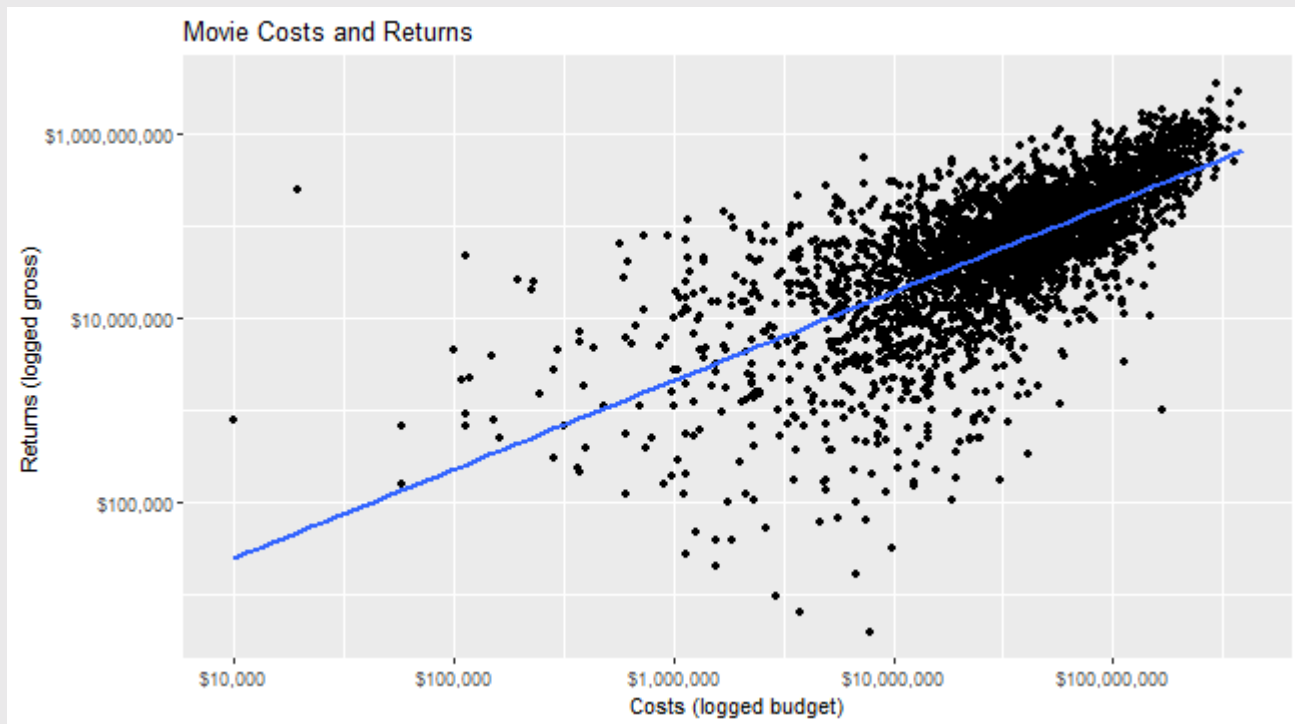
- If curious, can use `plotly` to see outliers

```
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)
summary(m)
```

```
##
## Call:
## lm(formula = gross_log ~ budget_log, data = mv)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -8.2672 -0.6354  0.1648  0.7899  8.5599
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.26107     0.30953   4.074 4.73e-05 ***
## budget_log    0.96386     0.01786  53.971 < 2e-16 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.281 on 3177 degrees of freedom
## (4494 observations deleted due to missingness)
## Multiple R-squared:  0.4783,    Adjusted R-squared:  0.4782
```

# 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  $\alpha$ ? What is  $\beta$ ?

$$gross\_log = 1.26 + 0.96 * budget\_log$$

# Interpreting with Logs

- Previously, we said:
  - $\alpha$  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  $\beta$  coefficient, it depends on where the logged variable appears:
  1.  $\log(Y) \sim X$ : 1 unit change in  $X \rightarrow (\exp(b)-1)*100\%$  change in  $Y$
  2.  $Y \sim \log(X)$ : 1% increase in  $X \rightarrow b/100$  unit change in  $Y$
  3.  $\log(Y) \sim \log(X)$ : 1% increase in  $X \rightarrow b\%$  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  $\alpha$  and  $\beta$  to minimize mistakes
  - Mistakes (aka "errors" or "residuals") are captured in the  $\varepsilon$  term
  - We can apply the **process** to these!

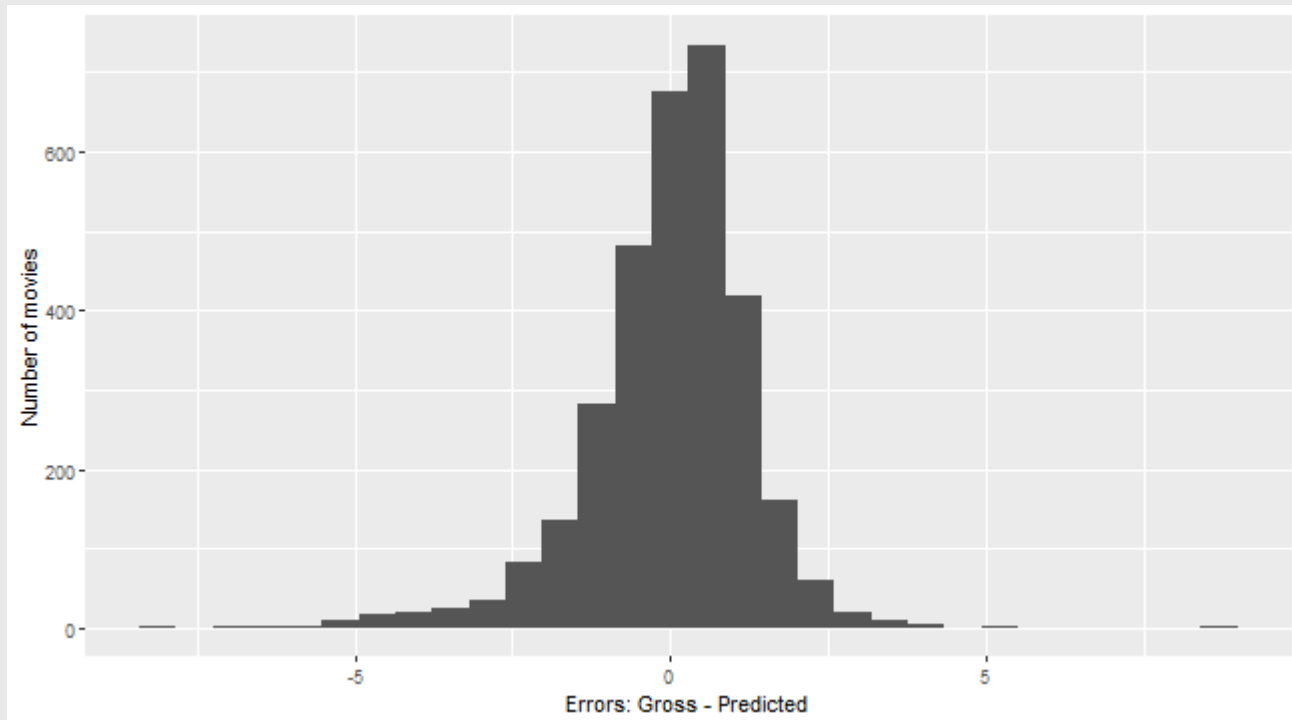
```
# Wrangle data to drop missingness!
mv_analysis <- mv %>% drop_na(gross_log,budget_log)
m <- lm(gross_log ~ budget_log,data = mv_analysis)
mv_analysis$predictions <- predict(m)
mv_analysis$errors <- mv_analysis$gross_log - mv_analysis$predictions

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')
```



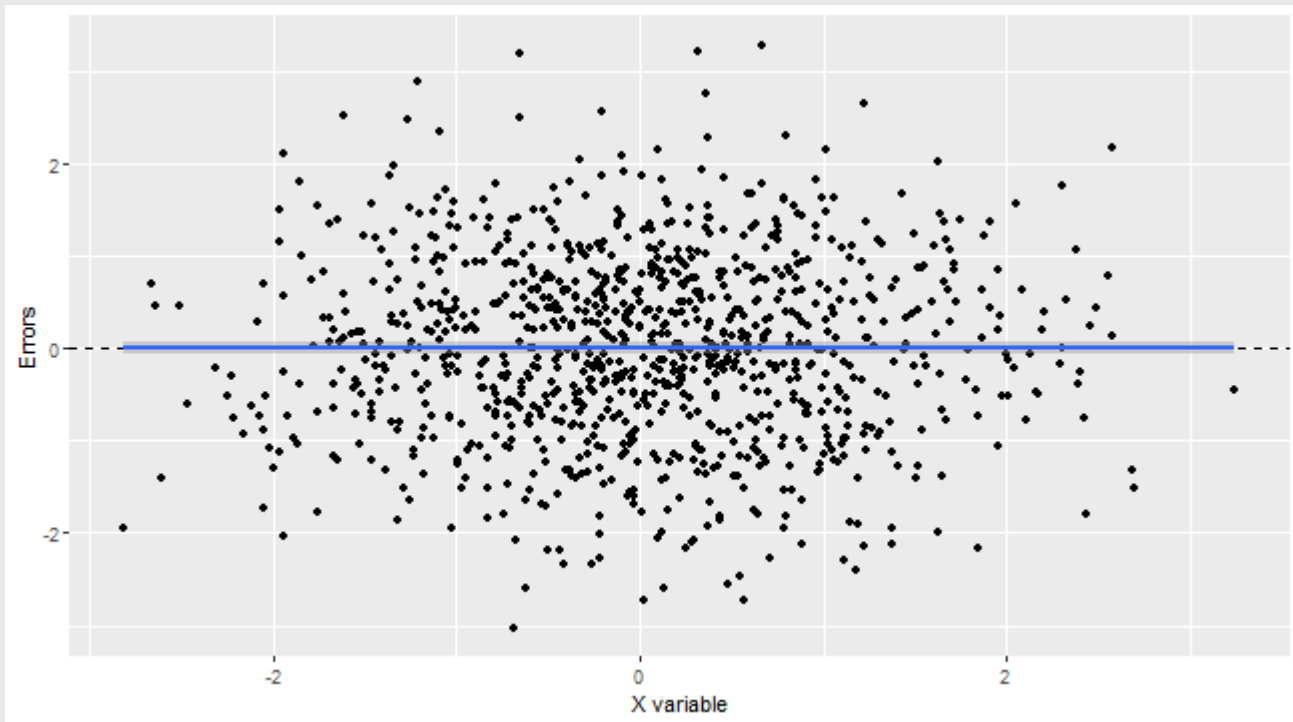
# Univariate Viz of Errors

- 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

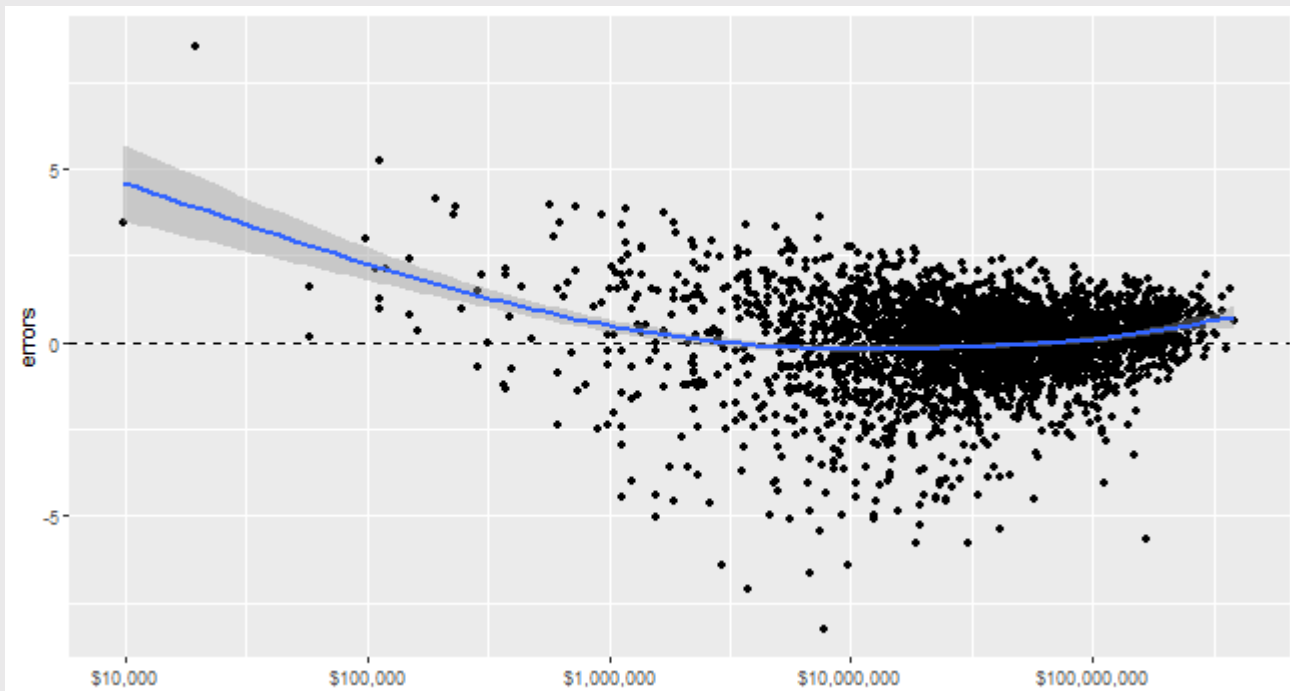
pIdeal



# 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

- **Error** (  $\varepsilon$  ): actual outcome (  $Y_i$  ) - predicted outcome (  $\hat{Y}_i$  )
  - The "distance" between the data and the model
- **Squared**:  $\varepsilon^2$ 
  1. Makes all values positive
  2. **Exaggerates** the presence of larger errors
- **Mean**: average these squared errors
- **Root**: take their square root (**un-exaggerate**)

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

# RMSE

- **Error** (  $\epsilon$  ): actual outcome (  $Y_i$  ) - predicted outcome (  $\hat{Y}_i$  )
  - The "distance" between the data and the model
- **Squared**:  $\epsilon^2$ 
  1. Makes all values positive
  2. Exaggerates the presence of larger errors
- **Mean**: average these squared errors
- **Root**: take their square root (un-exaggerate)

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

# RMSE

- **Error** (  $\varepsilon$  ): actual outcome (  $Y_i$  ) - predicted outcome (  $\hat{Y}_i$  )
  - The "distance" between the data and the model
- **Squared**:  $\varepsilon^2$ 
  1. Makes all values positive
  2. Exaggerates the presence of larger errors
- **Mean**: average these squared errors
- **Root**: take their square root (un-exaggerate)

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

# RMSE

- **Error** (  $\varepsilon$  ): actual outcome (  $Y_i$  ) - predicted outcome (  $\hat{Y}_i$  )
  - The "distance" between the data and the model
- **Squared**:  $\varepsilon^2$ 
  1. Makes all values positive
  2. Exaggerates the presence of larger errors
- **Mean**: average these squared errors
- **Root**: take their square root (un-exaggerate)

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

# RMSE

- **Error** (  $\varepsilon$  ): actual outcome (  $Y_i$  ) - predicted outcome (  $\hat{Y}_i$  )
  - The "distance" between the data and the model
- **Squared**:  $\varepsilon^2$ 
  1. Makes all values positive
  2. Exaggerates the presence of larger errors
- **Mean**: average these squared errors
- **Root**: 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) 0.04104524 0.03182923   1.289545 0.1975073  
## X           2.08804729 0.03210760  65.032814 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)
# Create list of row numbers at random
inds <- sample(1:nrow(mv_analysis),
               size = round(nrow(mv_analysis)/2),
               replace = F)

# Use slice(inds) to get training data
train <- mv_analysis %>%
  slice(inds)

# Use slice(-inds) to get test data
test <- mv_analysis %>%
  slice(-inds)
```

- We now have two datasets of roughly the same number of observations!

# 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) {
  inds <- sample(1:nrow(mv_analysis),
                 size = round(nrow(mv_analysis)/2),
                 replace = F)

  train <- mv_analysis %>% slice(inds)
  test <- mv_analysis %>% slice(-inds)

  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)
}

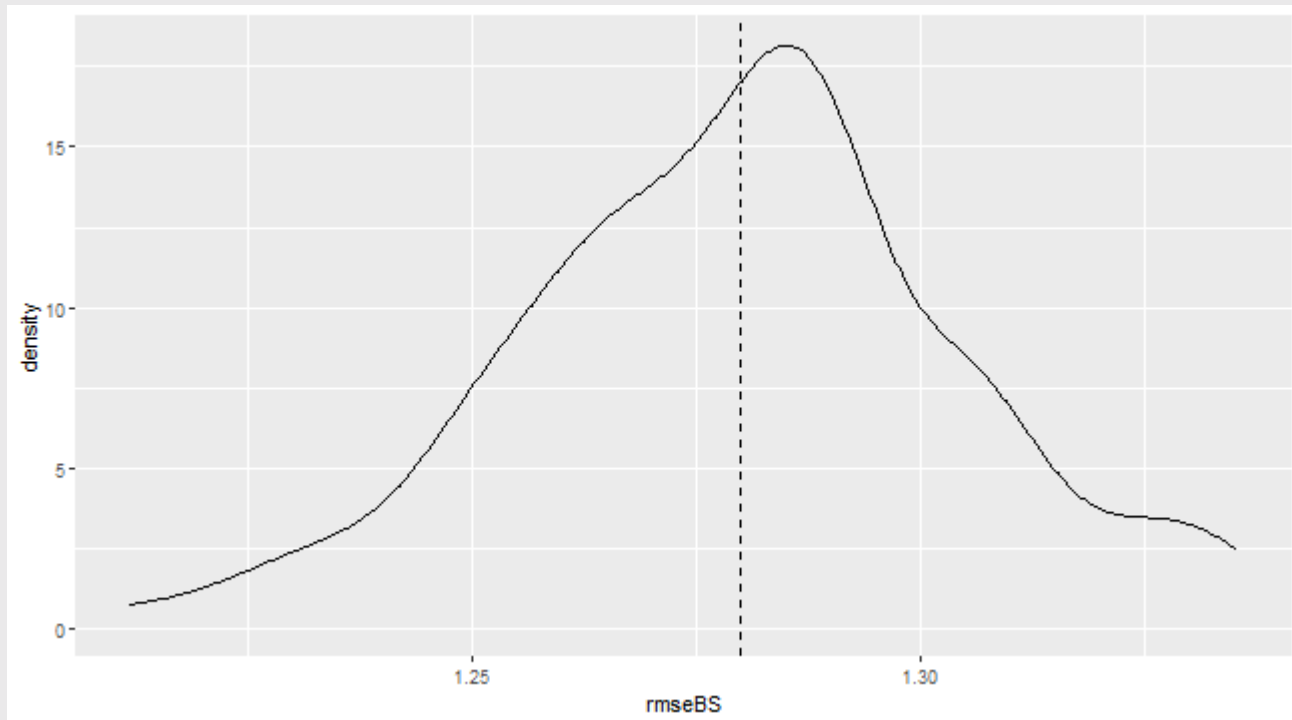
mean(bsRes)
```

```
## [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

```
inds <- sample(1:nrow(mv_analysis),  
              size = round(nrow(mv_analysis)*.8),  
              replace = F)
```

# Quiz & Homework

- Go to Brightspace and take the **12th** quiz
  - The password to take the quiz is ####
- **Homework:**
  1. Work through ds1000\_hw\_12.Rmd
  2. Problem Set 7 (due Friday)

