Lecture 9 Notes

2024-07-17

```
require(tidyverse)
```

```
## Loading required package: tidyverse
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ dplyr     1.1.4     ✔ readr     2.1.5
## ✔ forcats   1.0.0     ✔ stringr   1.5.1
## ✔ ggplot2   3.5.1     ✔ tibble    3.2.1
## ✔ lubridate 1.9.3     ✔ tidyr     1.3.1
## ✔ purrr     1.0.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts t
o become errors
```

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

Compare two regressions

```
# Step 1: Prepare the data
mv_analysis <- mv %>%
  mutate(gross_log = log(gross),
         budget_log = log(budget)) %>%
  drop_na(gross_log,budget_log,score)

# Step 2: Run regressions
m_budget <- lm(gross_log ~ budget_log,
               data = mv_analysis)

m_score <- lm(gross_log ~ score,
              data = mv_analysis)

# Step 3: Translate to plain language
require(broom)
```

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

```
tidy(m_budget)
```

```
## # 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
```

```
tidy(m_score)
```

```
## # A tibble: 2 × 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)   16.1      0.211      76.5  0
## 2 score          0.279    0.0325      8.58 1.49e-17
```

```
exp(16.1)
```

```
## [1] 9820671
```

```
(exp(0.279)-1)*100
```

```
## [1] 32.18073
```

Looking at errors

- Step 1: Create the errors

```
mv_analysis <- mv_analysis %>%
  mutate(Yhat_budget = predict(m_budget),
         Yhat_score = predict(m_score))

mv_analysis <- mv_analysis %>%
  mutate(error_budget = gross_log - Yhat_budget,
         error_score = gross_log - Yhat_score)
```

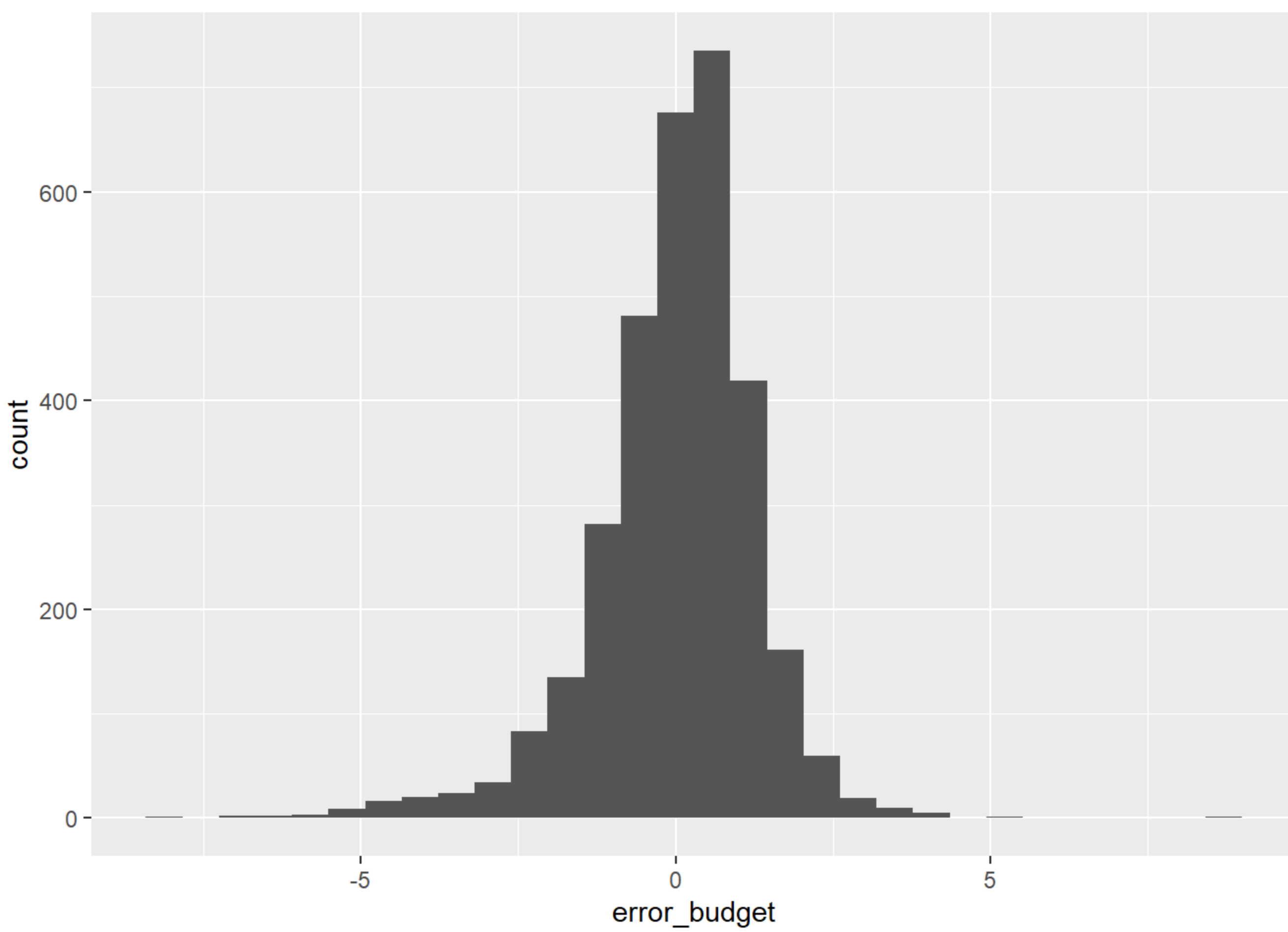
- Step 2: univariate visualization of errors

```
mv_analysis %>%
  select(title,error_budget,error_score)
```

```
## # A tibble: 3,179 × 3
##    title                  error_budget error_score
##    <chr>                         <dbl>       <dbl>
##  1 Almost Famous                -0.834      -0.220
##  2 American Psycho               0.913      -0.460
##  3 Gladiator                     0.930       1.90
##  4 Requiem for a Dream          -0.195      -2.19
##  5 Memento                       0.826      -0.527
##  6 Cast Away                     0.980       2.01
##  7 Scary Movie                   2.04        2.02
##  8 The Perfect Storm             0.286       2.14
##  9 Coyote Ugly                   0.321       1.27
## 10 X-Men                         0.784       1.75
## # ℹ 3,169 more rows
```

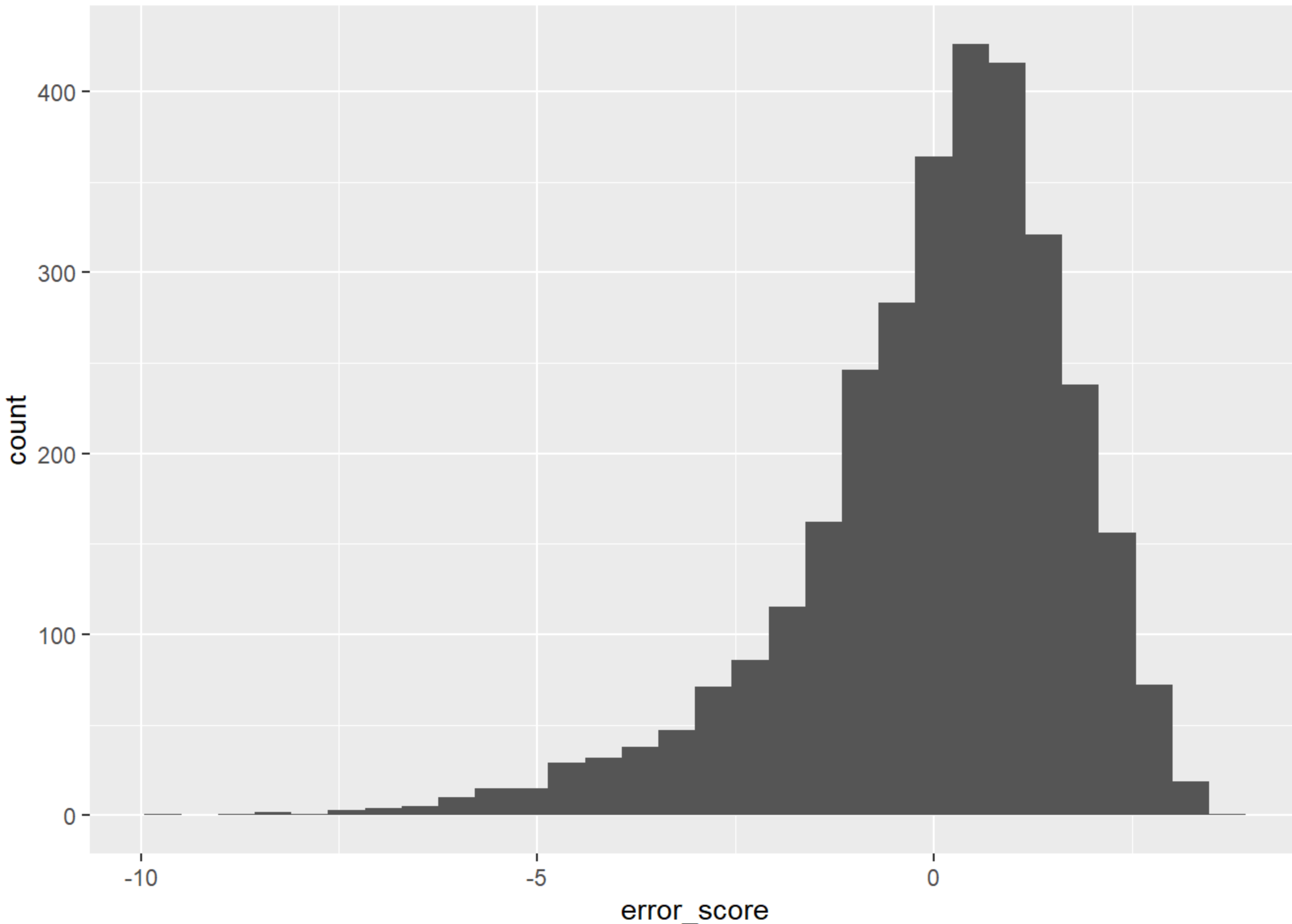
```
mv_analysis %>%
  ggplot(aes(x = error_budget)) +
  geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
mv_analysis %>%
  ggplot(aes(x = error_score)) +
  geom_histogram()
```

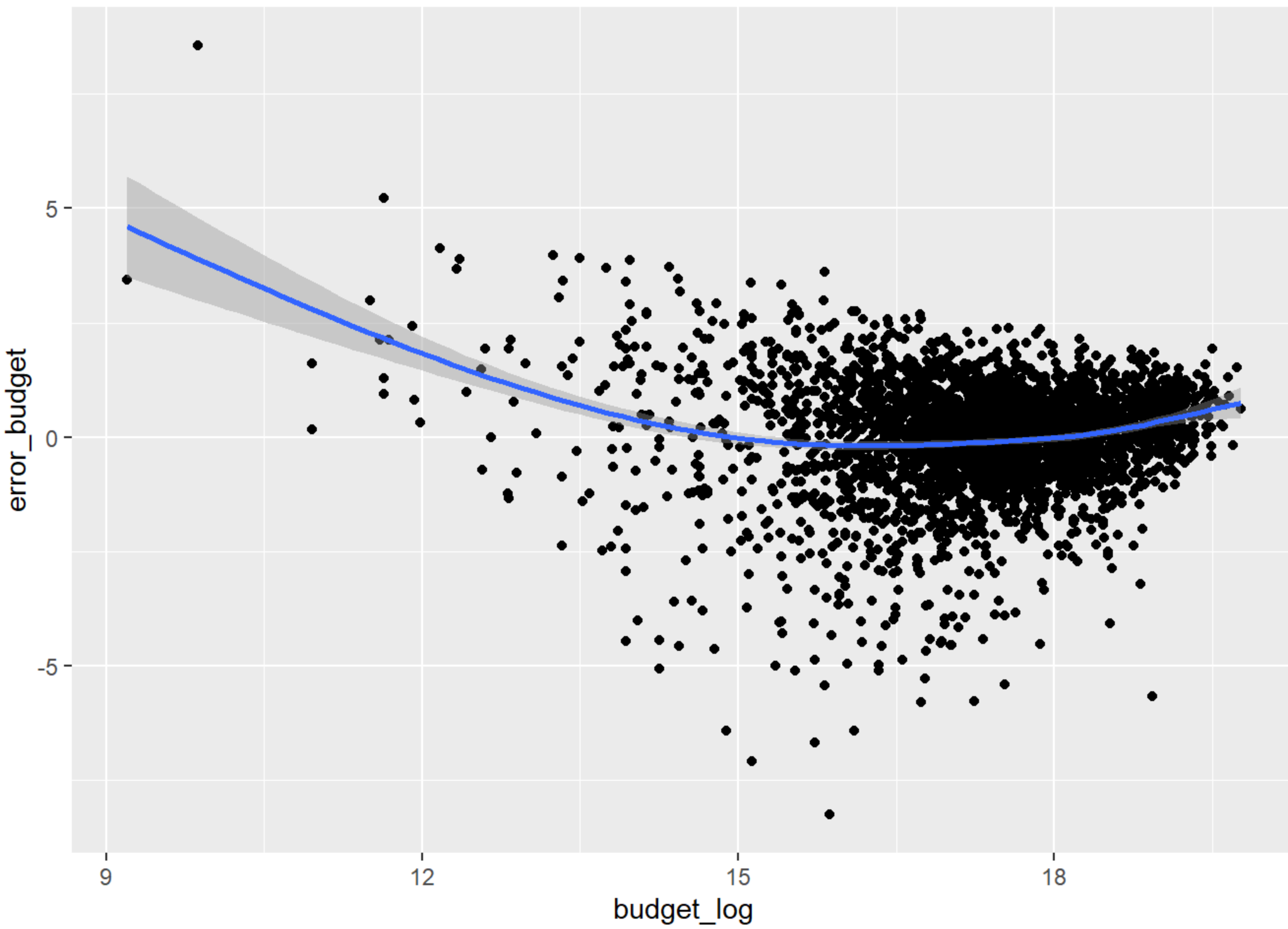
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



- Step 3: Multivariate visualization

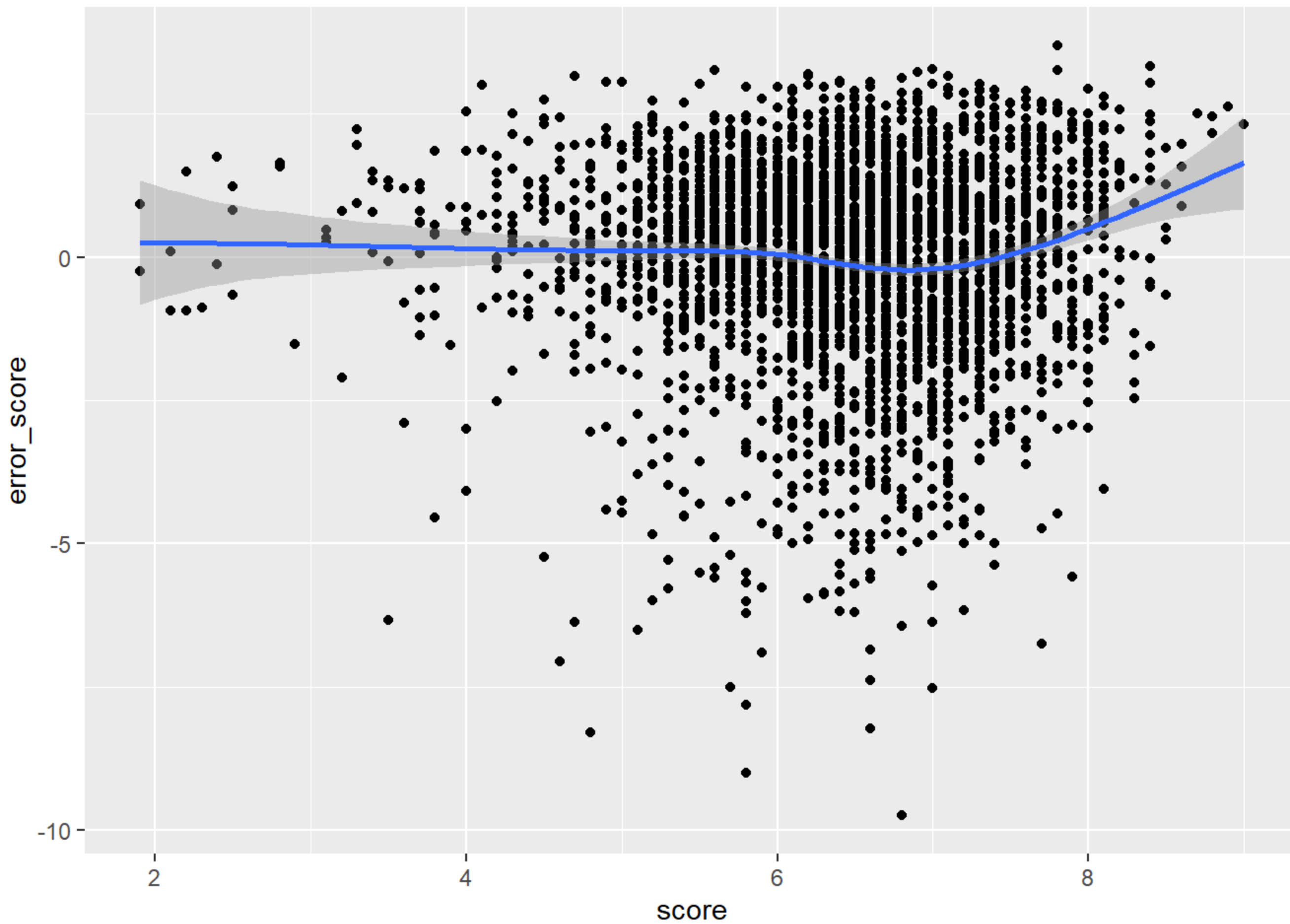
```
mv_analysis %>%
  ggplot(aes(x = budget_log,
             y = error_budget)) +
  geom_point() +
  geom_smooth()
```

```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```




```
mv_analysis %>%
  ggplot(aes(x = score,
             y = error_score)) +
  geom_point() +
  geom_smooth()
```

```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```



- Step 4: Cross validated RMSE

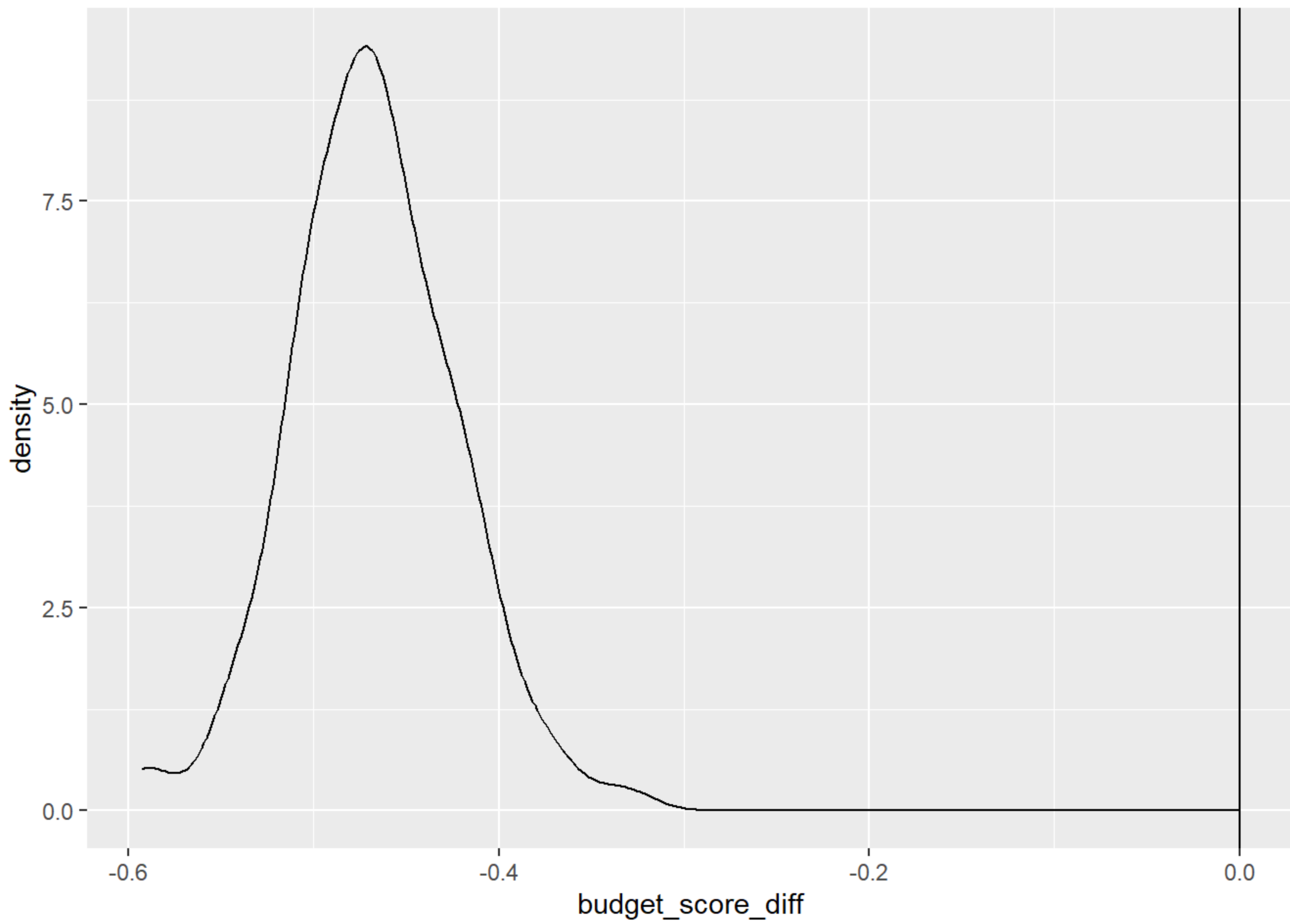
```r
set.seed(123)
cvRes <- NULL
for(i in 1:100) {
  # First, split the data
  train <- mv_analysis %>%
    sample_n(size = round(nrow(mv_analysis)*.8),
             replace = F)
  test <- mv_analysis %>%
    anti_join(train)

  # Second, estimate models in train data
  mTmp_budget <- lm(gross_log ~ budget_log,
                    data = train)
  mTmp_score <- lm(gross_log ~ score,
                   data = train)

  # Third, calculate RMSE
  answer <- test %>%
    mutate(Yhat_budget = predict(mTmp_budget,
                                 newdata = test),
           Yhat_score = predict(mTmp_score,
                                newdata = test)) %>%
    mutate(error_budget = gross_log - Yhat_budget,
           error_score = gross_log - Yhat_score) %>%
    summarise(rmse_budget = sqrt(mean(error_budget^2)),
              rmse_score = sqrt(mean(error_score^2))) %>%
    mutate(cv_number = i)

  # Save result to cvRes object
  cvRes <- cvRes %>%
    bind_rows(answer)
}
```

Looking at CV results

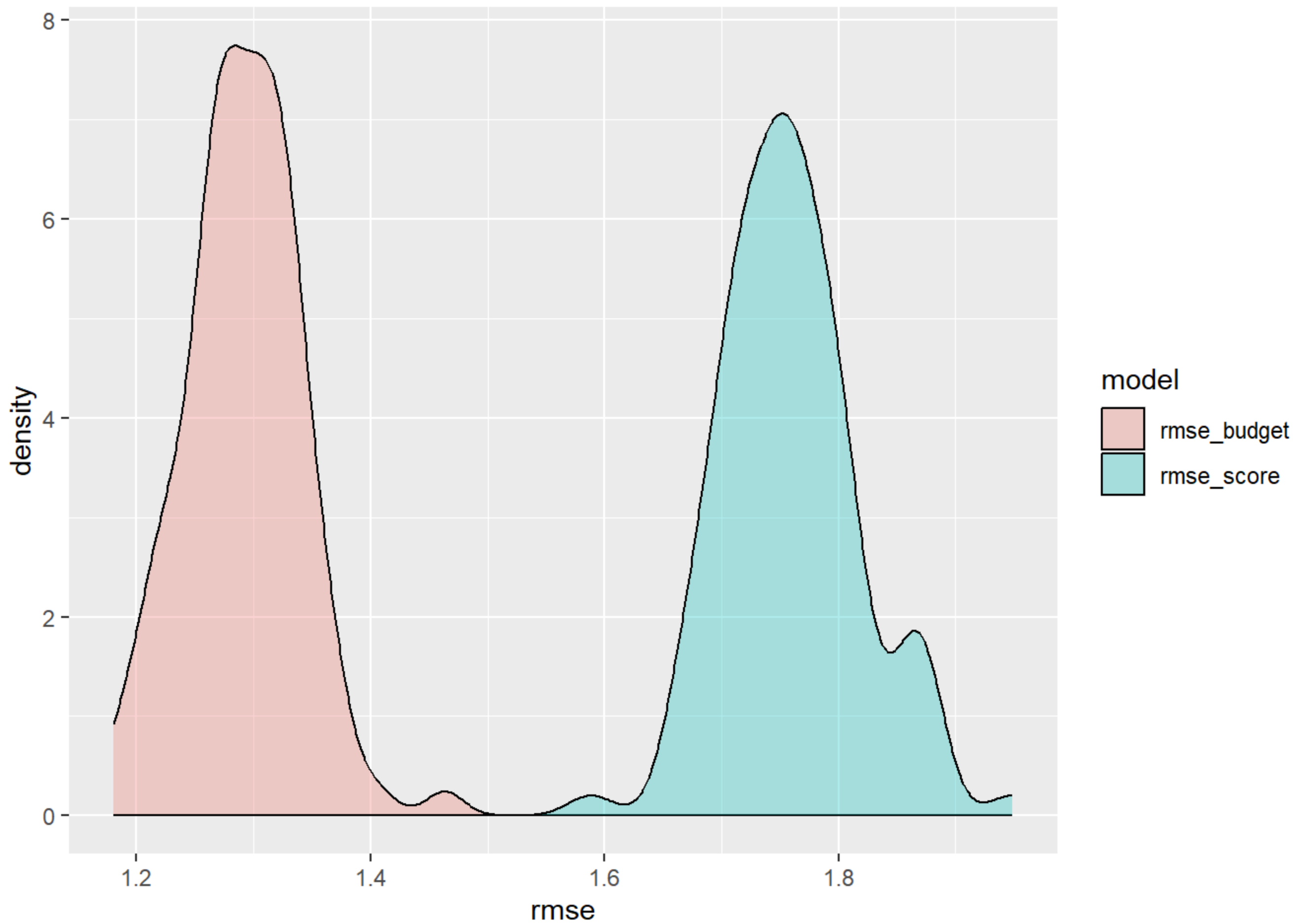
```r
cvRes %>%
  summarise(rmse_budget_avg = mean(rmse_budget),
            rmse_score_avg = mean(rmse_score))
```

```
## # A tibble: 1 × 2
##   rmse_budget_avg rmse_score_avg
##             <dbl>          <dbl>
## 1            1.29           1.76
```

```r
# Visualize the answer
cvRes %>%
  mutate(budget_score_diff = rmse_budget - rmse_score) %>%
  ggplot(aes(x = budget_score_diff)) +
  geom_density() +
  geom_vline(xintercept = 0)
```




```r
cvRes %>%
  pivot_longer(cols = c(rmse_budget,rmse_score),
               names_to = "model",
               values_to = "rmse") %>%
  ggplot(aes(x = rmse,
             fill = model)) +
  geom_density(alpha = .3)
```



Bechdel Test

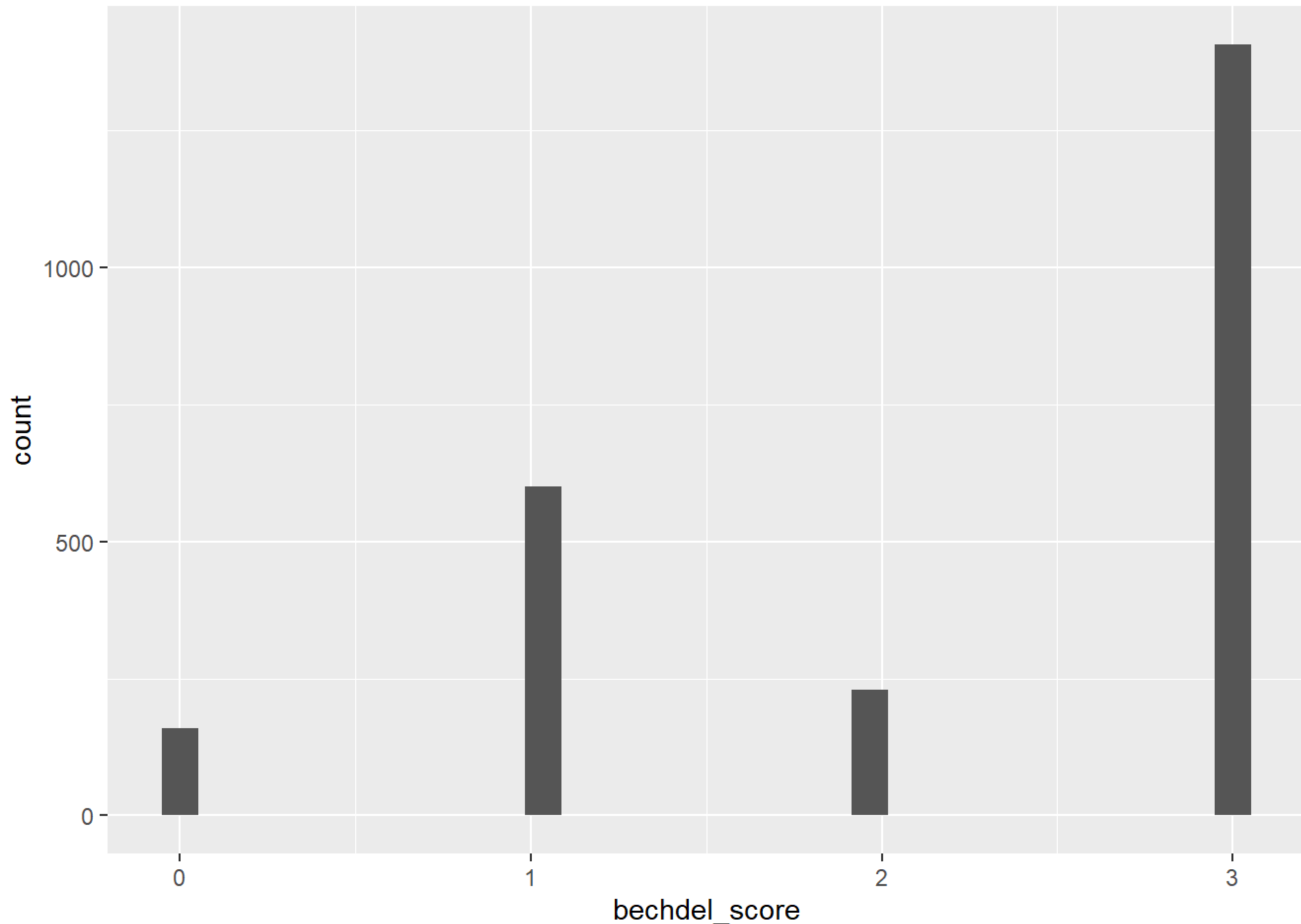
- First, look at the data

```
summary(mv %>%
  select(bechdel_score))
```

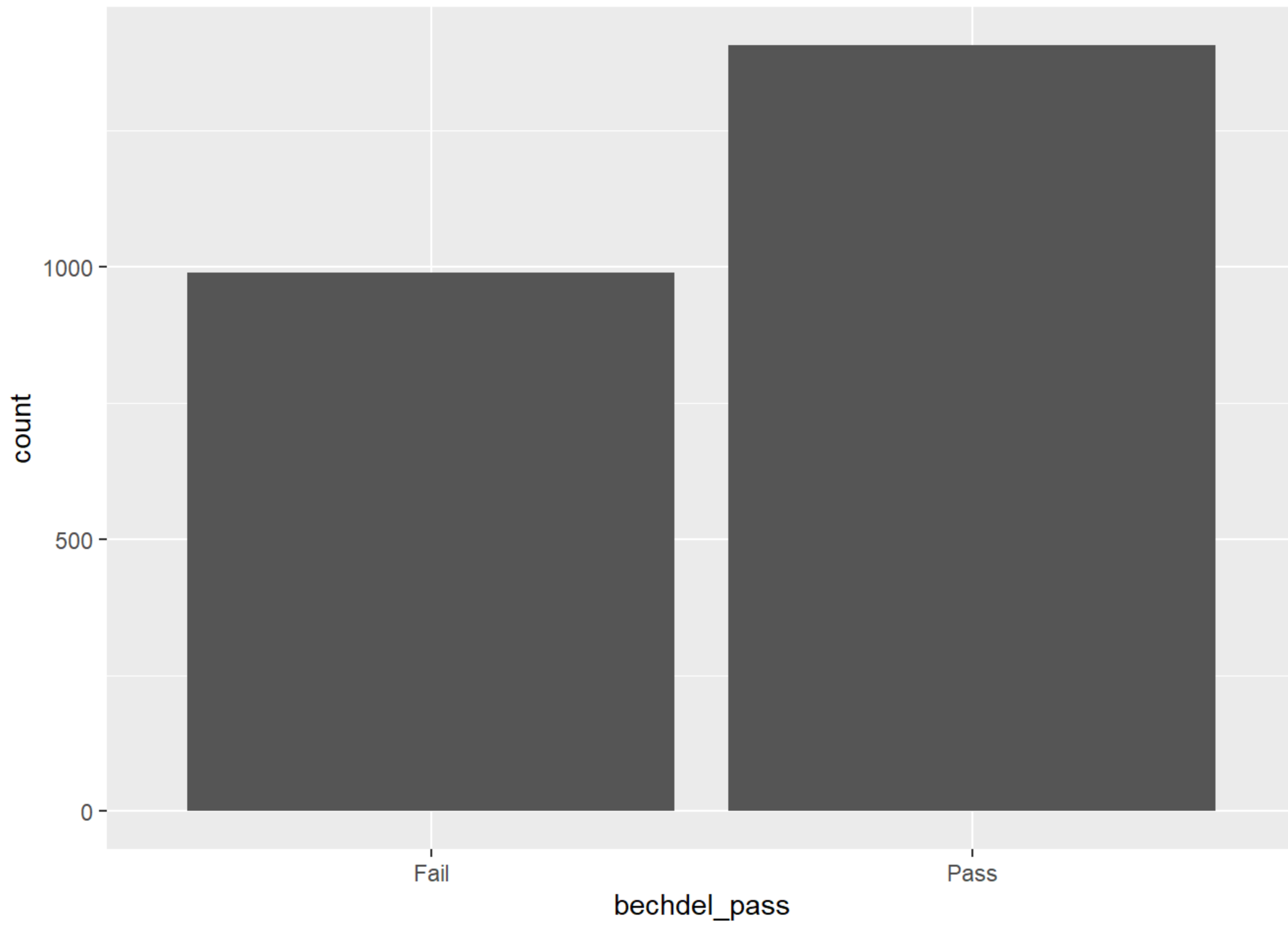
```
##  bechdel_score
##  Min.   :0.000
##  1st Qu.:1.000
##  Median :3.000
##  Mean   :2.171
##  3rd Qu.:3.000
##  Max.   :3.000
##  NA's   :4162
```

```
mv_analysis <- mv %>%
  drop_na(bechdel_score,gross) %>%
  mutate(gross_log = log(gross))

mv_analysis %>%
  ggplot(aes(x = bechdel_score)) +
  geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
# Wrangle data to create binary bechdel score
mv_analysis <- mv_analysis %>%
  mutate(bechdel_pass = ifelse(bechdel_score == 3,
                               "Pass",
                               "Fail"))

mv_analysis %>%
  ggplot(aes(x = bechdel_pass)) +
  geom_bar()
```



Regression

```
m_bechdel_binary <- lm(gross_log ~ bechdel_pass,
                       data = mv_analysis)

tidy(m_bechdel_binary)
```

```
## # A tibble: 2 × 5
##   term             estimate std.error statistic p.value
##   <chr>               <dbl>     <dbl>     <dbl>   <dbl>
## 1 (Intercept)         18.0      0.0629    286.    0
## 2 bechdel_passPass   -0.233     0.0821     -2.84 0.00451
```

```
(exp(-.233)-1)*100
```

```
## [1] -20.78464
```

Controlling for budget

```
mv_analysis <- mv_analysis %>%
  mutate(budget_log = log(budget)) %>%
  drop_na(gross_log,budget_log,bechdel_pass)

m_bechdel_control <- lm(gross_log ~ bechdel_pass + budget_log,
                        data = mv_analysis)

tidy(m_bechdel_control)
```

```
## # A tibble: 3 × 5
##   term             estimate std.error statistic   p.value
##   <chr>               <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept)         2.12     0.353       6.01 2.25e-  9
## 2 bechdel_passPass    0.188    0.0525      3.58 3.56e-  4
## 3 budget_log          0.921    0.0199     46.2  3.57e-320
```

```
exp(2.12)
```

```
## [1] 8.331137
```

```
(exp(.188)-1)*100
```

```
## [1] 20.68335
```

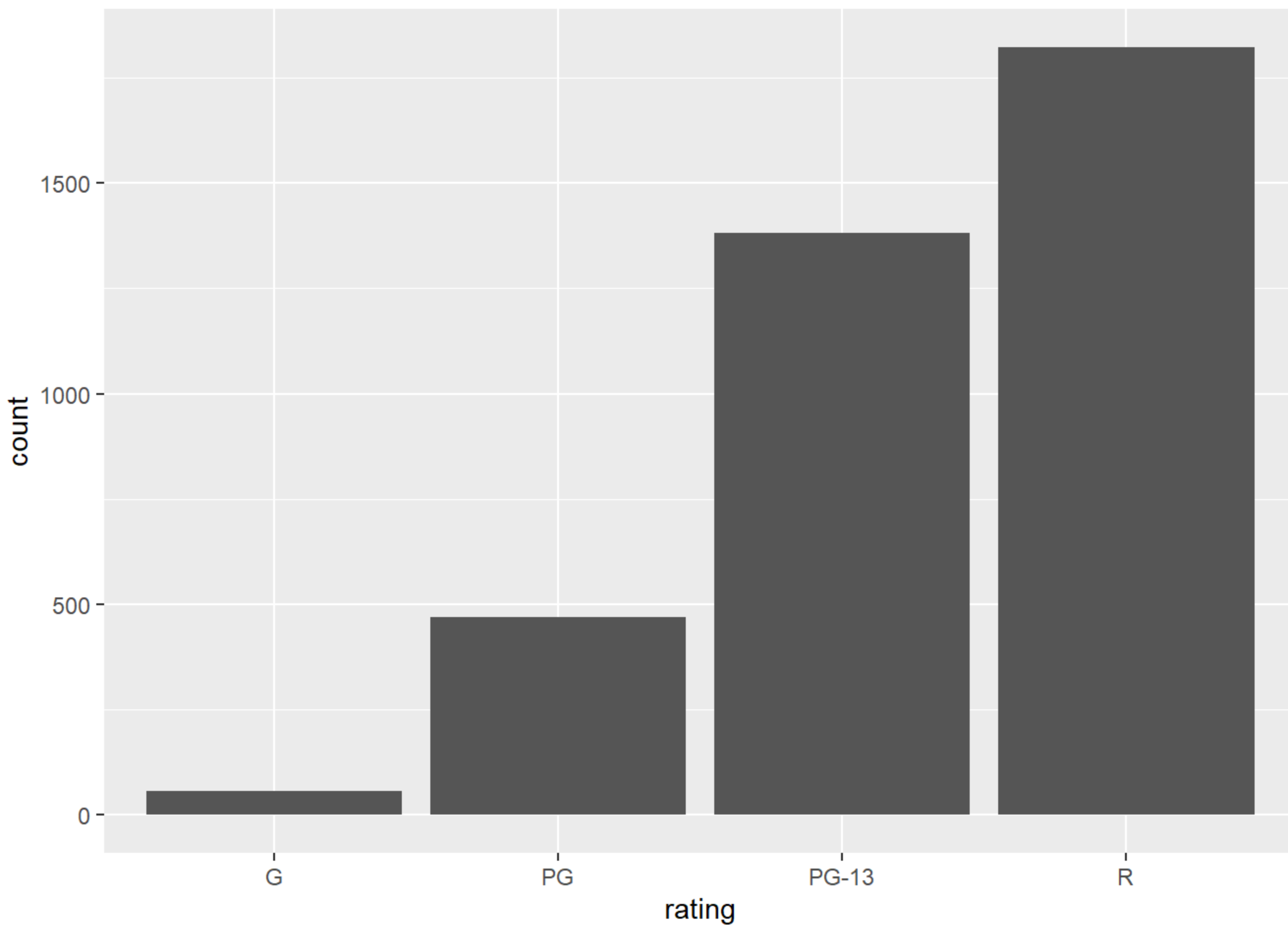
New variable: categorical X

- Rating: A movie's rating is assigned to inform people how violent / drug / alcohol / sexy the movie

```
mv_analysis %>%
  count(rating)
```

```
## # A tibble: 8 × 2
##   rating        n
##   <chr>     <int>
## 1 G            33
## 2 NC-17         4
## 3 Not Rated    23
## 4 PG          278
## 5 PG-13       837
## 6 R           878
## 7 TV-MA         1
## 8 Unrated       4
```

```r
# Wrangling the data
mv_analysis <- mv %>%
  mutate(rating = ifelse(rating %in% c("NC-17","Not Rated",
                                       "TV-MA","Unrated",
                                       "TV-14","TV-PG"),
                         NA,
                         rating),
         gross_log = log(gross)) %>%
  drop_na(gross_log,rating)

# Univariate visualization
mv_analysis %>%
  ggplot(aes(x = rating)) +
  geom_bar()
```



Run the regression

```r
m_rating <- lm(gross_log ~ rating,
               data = mv_analysis)
tidy(m_rating)
```

```
## # A tibble: 4 × 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)   19.1       0.252     76.0  0
## 2 ratingPG      -0.554     0.266     -2.08 3.75e- 2
## 3 ratingPG-13   -0.919     0.257     -3.58 3.48e- 4
## 4 ratingR       -2.43      0.255     -9.53 2.71e-21
```