

Lecture 8 Notes

2024-07-15

Regression using mv.rds data

```
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 to
o become errors
```

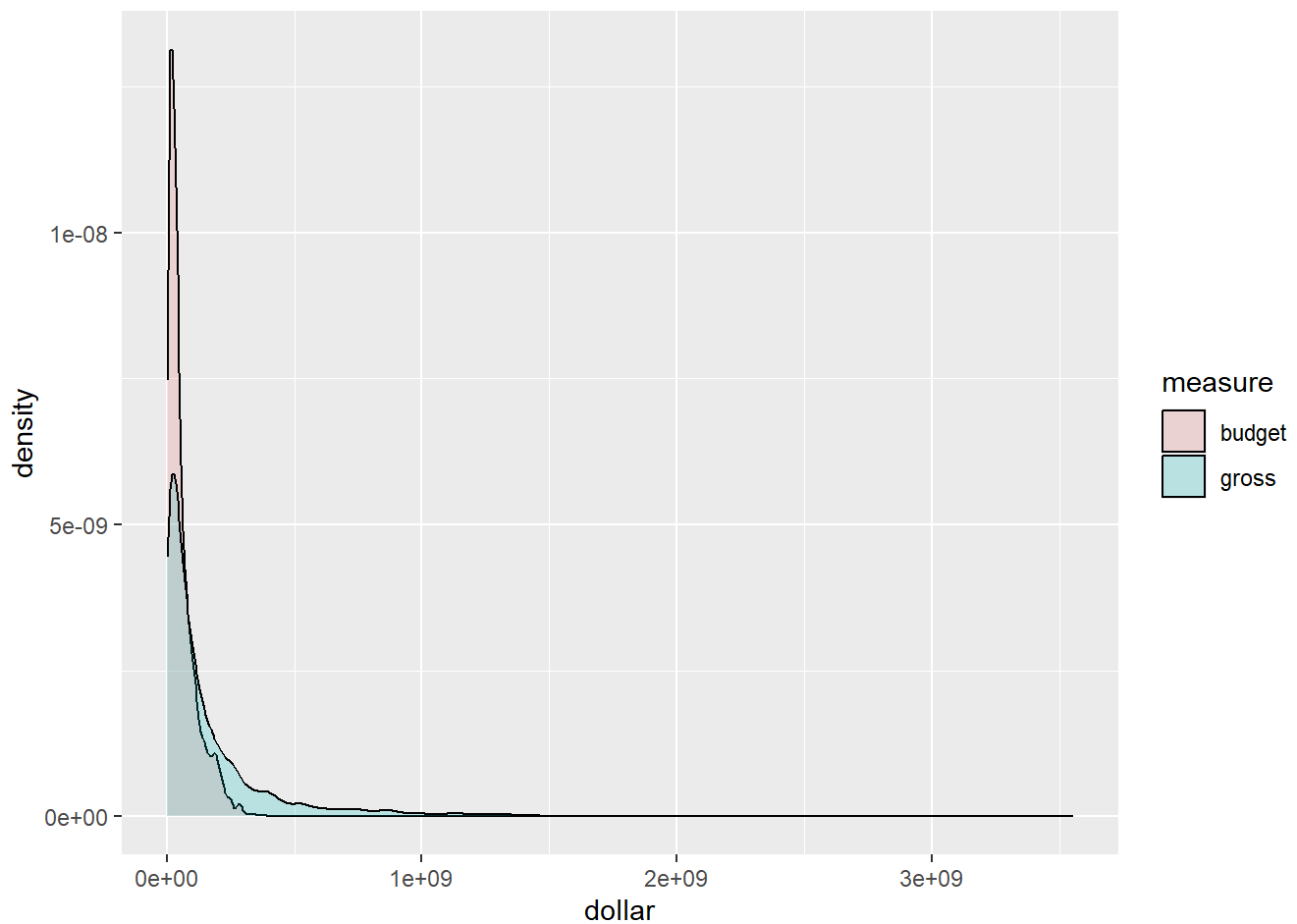
```
mv <- read_rds("https://github.com/jbisbee1/ISP_Data_Science_2024/raw/main/data/mv.Rds")
```

pivot_longer()

```
mv_long <- mv %>%
  select(title,budget,gross) %>%
  drop_na() %>%
  pivot_longer(cols = c("budget","gross"),
               names_to = "measure",
               values_to = "dollar")
```

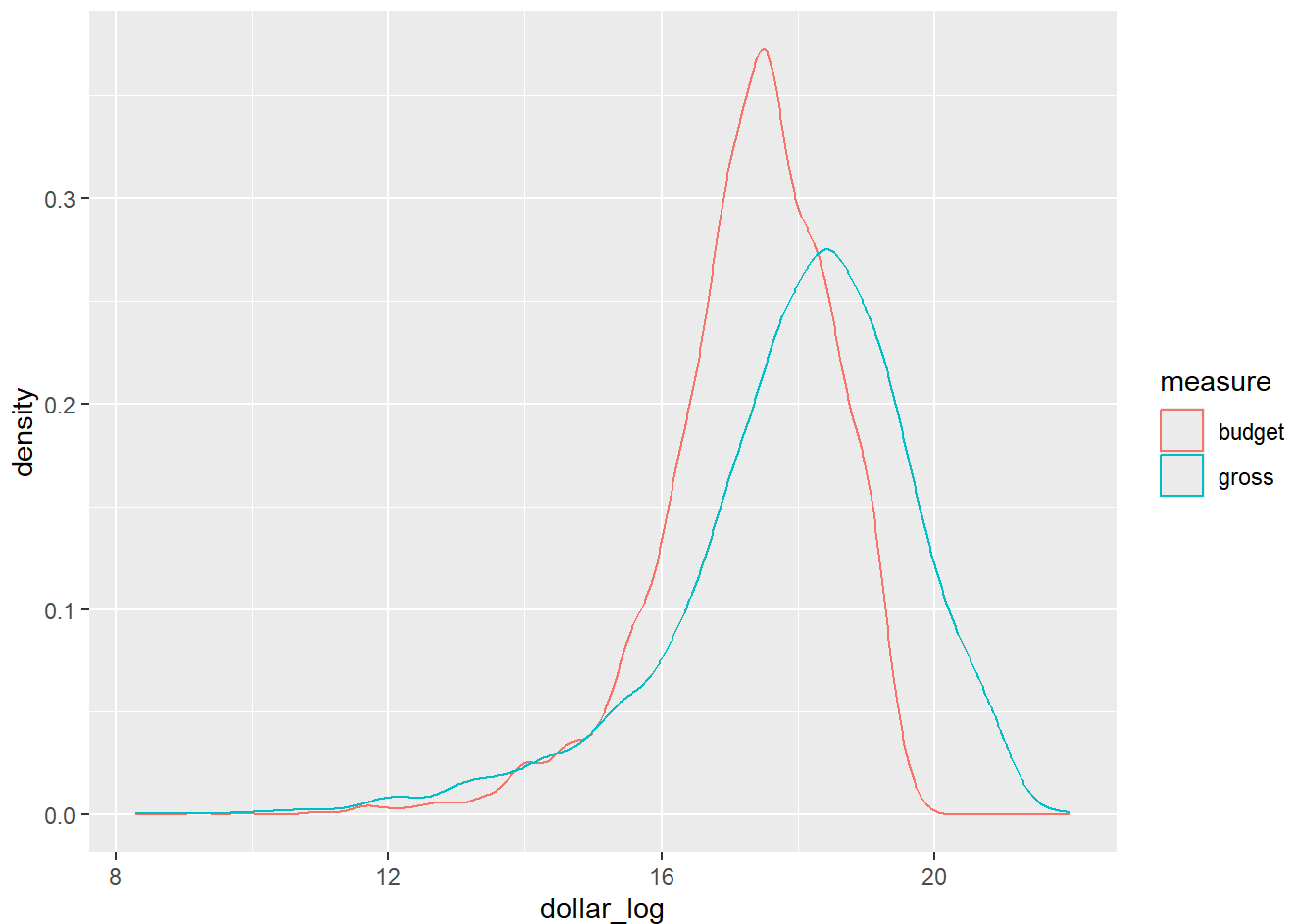
Create univariate visualization of both X and Y together

```
mv_long %>%
  ggplot(aes(x = dollar,
             fill = measure)) +
  geom_density(alpha = .2)
```



Log to get rid of extreme skew

```
mv_long %>%  
  mutate(dollar_log = log(dollar)) %>%  
  ggplot(aes(x = dollar_log,  
             color = measure)) +  
  geom_density()
```

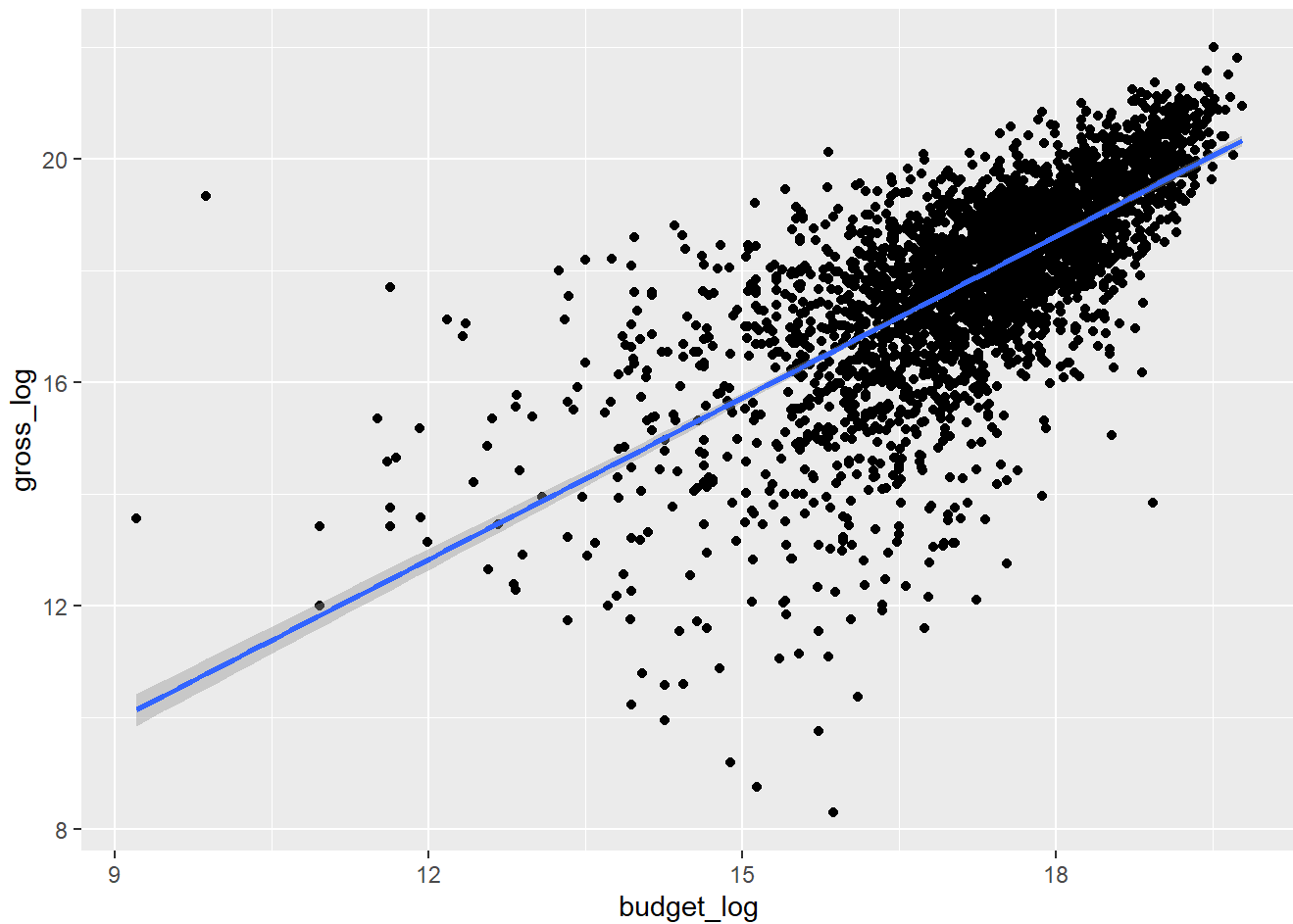


Running multivariate viz and regression

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

```
# Multivariate visualization  
mv_analysis %>%  
  ggplot(aes(x = budget_log,  
            y = gross_log)) +  
  geom_point() +  
  geom_smooth(method = 'lm')
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



Running regression with the `lm()` function

```
m1 <- lm(gross_log ~ budget_log,  
  data = mv_analysis)  
  
# Looking at result: two methods  
# Method 1: use summary()  
summary(m1)
```

```
##
## Call:
## lm(formula = gross_log ~ budget_log, data = mv_analysis)
##
## 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
## Multiple R-squared:  0.4783, Adjusted R-squared:  0.4782
## F-statistic: 2913 on 1 and 3177 DF, p-value: < 2.2e-16
```

```
# Method 2: use tidy() from broom package
require(broom)
```

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

```
tidy(m1)
```

```
## # 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
```

Tangent: log rules

```
(exp(0.96)-1)*100
```

```
## [1] 161.1696
```

Evaluating model performance

Step 1: Look at errors

First, calculate the errors

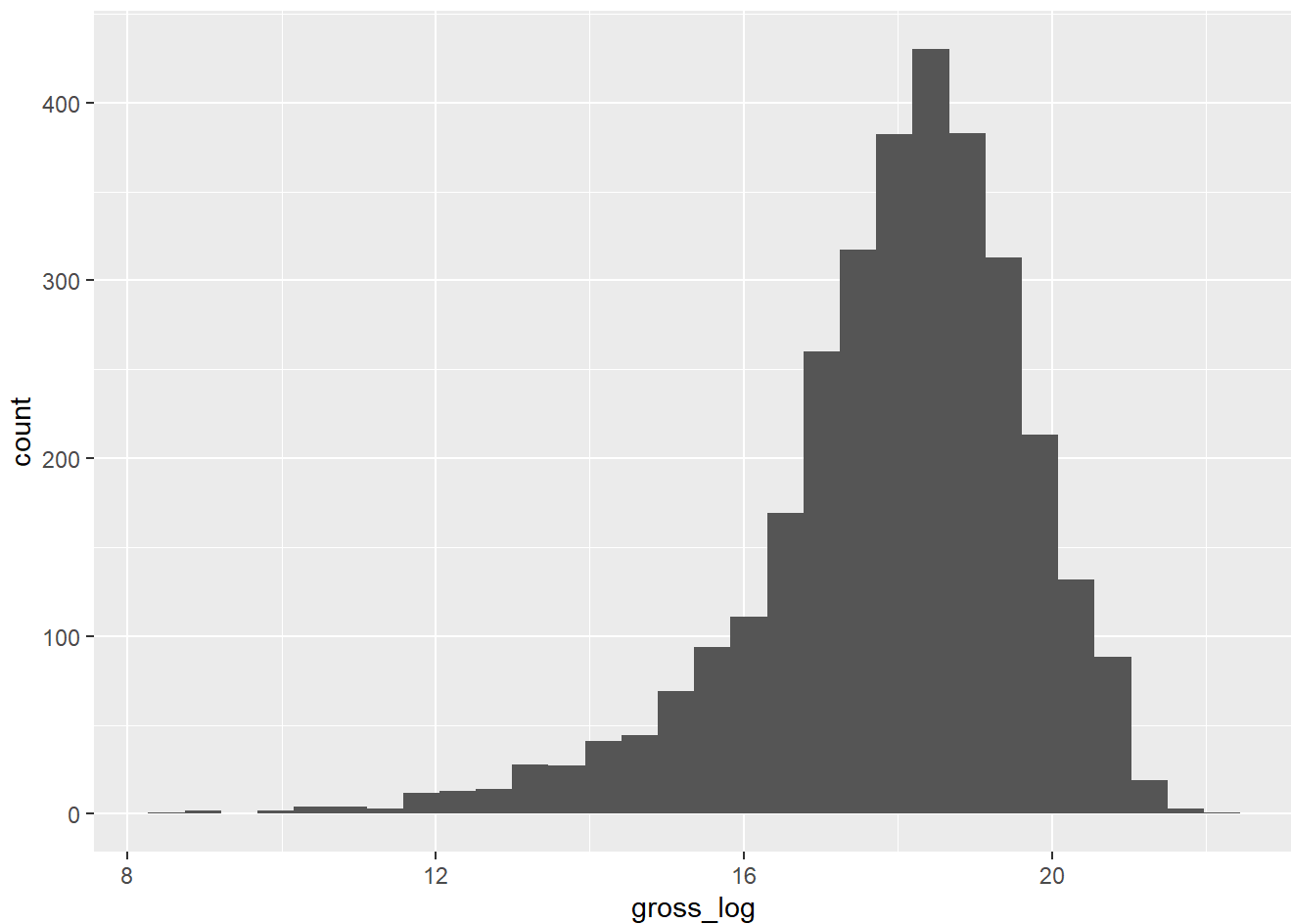
```
mv_analysis <- mv_analysis %>%
  mutate(Yhat = predict(m1)) %>%
  mutate(error = gross_log - Yhat)

# Summarize the errors
summary(mv_analysis %>%
  select(error))
```

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

```
# Reminder of what Y looks like
mv_analysis %>%
  ggplot(aes(x = gross_log)) +
  geom_histogram()
```

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

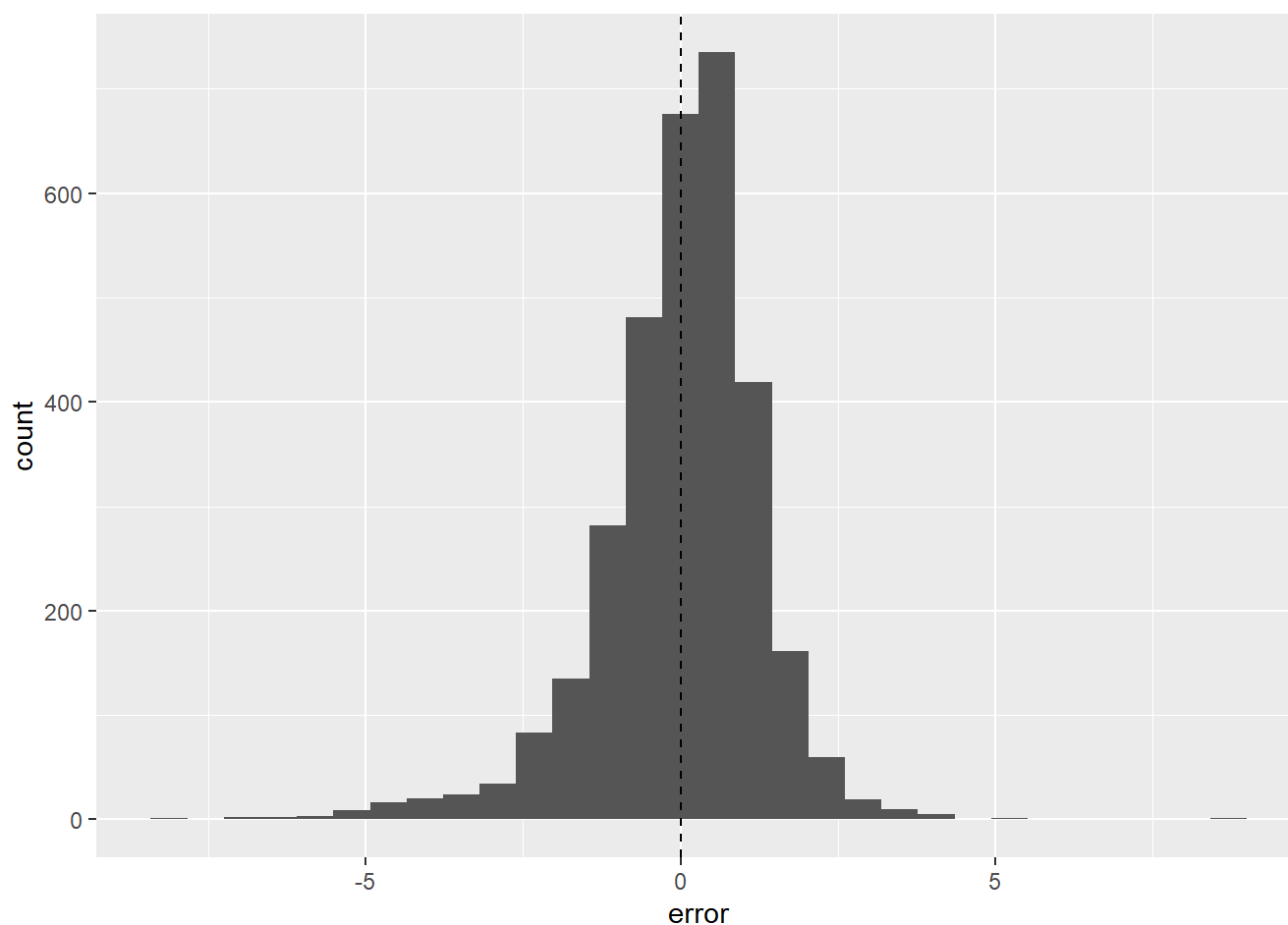


Looking at the shape of error

First, univariate visualization of the errors

```
mv_analysis %>%  
  ggplot(aes(x = error)) +  
  geom_histogram() +  
  geom_vline(xintercept = 0,  
            linetype = 'dashed')
```

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



Second, multivariate visualization of the errors

```
mv_analysis %>%  
  ggplot(aes(x = budget_log,  
            y = error)) +  
  geom_point() +  
  geom_hline(yintercept = 0,  
            linetype = 'dashed') +  
  geom_smooth()
```

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

