551 HW 3

Matthew Stoebe

2024-10-03

#10.6:

Regression models with interactions: The folder Beauty contains data (use file beauty.csv) from Hamermesh and Parker (2005) on student evaluations of instructors' beauty and teaching quality for several courses at the University of Texas. The teaching evaluations were conducted at the end of the semester, and the beauty judgments were made later, by six students who had not attended the classes and were not aware of the course evaluations. (a) Run a regression using beauty (the variable beauty) to predict course evaluations (eval), adjusting for various other predictors. Graph the data and fitted model, and explain the meaning of each of the coefficients along with the residual standard deviation. Plot the residuals versus fitted values.

(b) Fit some other models, including beauty and also other predictors. Consider at least one model with interactions. For each model, explain the meaning of each of its estimated coefficients.

```
# Load necessary libraries
library(ggplot2)
library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(broom) # for tidy model summaries

# Load the data
df <- read.csv("./Other Data/HW_3/beauty.csv")

# Inspect the first few rows of the data
head(df)

##   eval     beauty female age minority nonenglish lower course_id
## 1  4.3  0.2015666      1  36        1          0     0         3
## 2  4.5 -0.8260813      0  59        0          0     0         0
## 3  3.7 -0.6603327      0  51        0          0     0         4
```

```
## 4  4.3 -0.7663125      1  40        0          0     0         2
## 5  4.4  1.4214450      1  31        0          0     0         0
## 6  4.2  0.5002196      0  62        0          0     0         0

# Model 1: Simple linear regression with beauty and other predictors
model1 <- lm(eval ~ beauty + age + female + minority + nonenglish, data = df)

# Model 2: Adding an interaction term between beauty and female
model2 <- lm(eval ~ beauty * female + age + minority + nonenglish, data = df)

# Model 3: Adding interactions between beauty and other variables
model3 <- lm(eval ~ beauty * female + beauty * age + beauty * minority +
beauty * nonenglish, data = df)


# Summary of each model
summary(model1)

##
## Call:
## lm(formula = eval ~ beauty + age + female + minority + nonenglish,
##     data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.87797 -0.35784  0.04323  0.37956  1.02073
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.254132   0.142588  29.835  < 2e-16 ***
## beauty       0.140942   0.032938   4.279 2.29e-05 ***
## age         -0.002707   0.002750  -0.984  0.32545
## female      -0.207452   0.052563  -3.947 9.17e-05 ***
## minority    -0.044374   0.075725  -0.586  0.55817
## nonenglish  -0.313490   0.108630  -2.886  0.00409 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5324 on 457 degrees of freedom
## Multiple R-squared:  0.08919,    Adjusted R-squared:  0.07922
## F-statistic:  8.95 on 5 and 457 DF,  p-value: 4.001e-08

summary(model2)

##
## Call:
## lm(formula = eval ~ beauty * female + age + minority + nonenglish,
##     data = df)
##
## Residuals:
```

```
##      Min       1Q   Median       3Q      Max
## -1.84616 -0.34549  0.04303  0.39253  1.05515
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    4.234142   0.142766  29.658  < 2e-16 ***
## beauty         0.193681   0.045052   4.299 2.10e-05 ***
## female        -0.214027   0.052593  -4.069 5.55e-05 ***
## age           -0.002169   0.002763  -0.785  0.43271
## minority      -0.017367   0.077195  -0.225  0.82210
## nonenglish    -0.330643   0.108864  -3.037  0.00252 **
## beauty:female -0.111446   0.065107  -1.712  0.08763 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5313 on 456 degrees of freedom
## Multiple R-squared:  0.095,  Adjusted R-squared:  0.08309
## F-statistic: 7.978 on 6 and 456 DF,  p-value: 3.372e-08
```

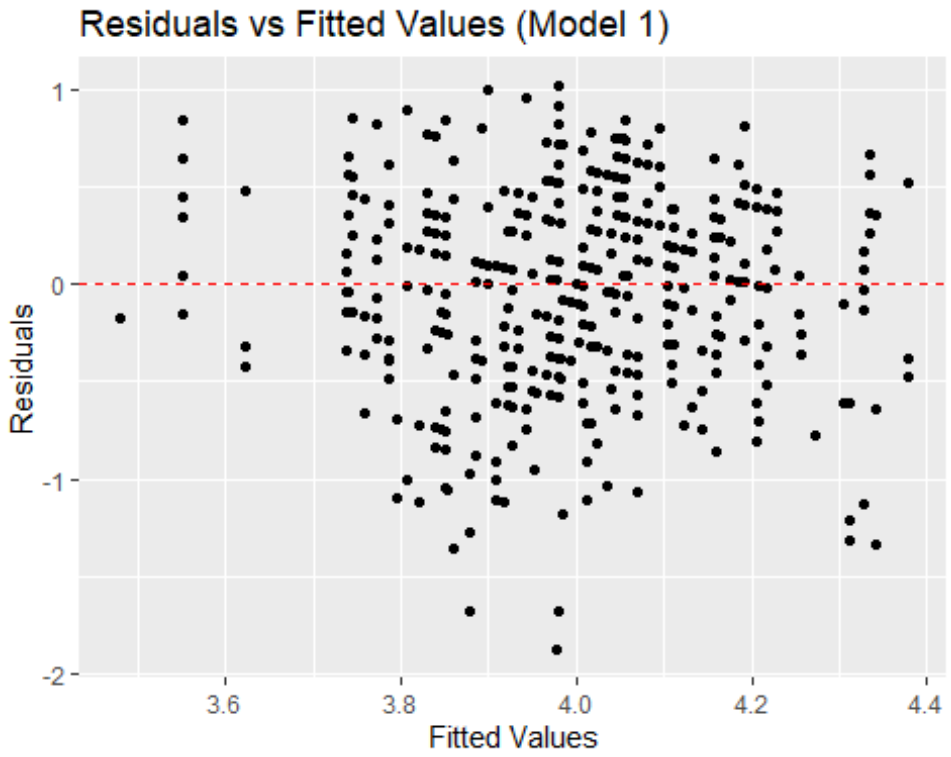
```
summary(model3)
```

```
##
## Call:
## lm(formula = eval ~ beauty * female + beauty * age + beauty *
##     minority + beauty * nonenglish, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.79912 -0.34949  0.02668  0.40794  1.15316
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)        4.214913   0.142404  29.598  < 2e-16 ***
## beauty            -0.442954   0.188034  -2.356 0.018912 *
## female            -0.201594   0.053459  -3.771 0.000184 ***
## age               -0.001198   0.002753  -0.435 0.663744
## minority          -0.087540   0.080492  -1.088 0.277365
## nonenglish        -0.302712   0.112018  -2.702 0.007144 **
## beauty:female      0.031452   0.074512   0.422 0.673142
## beauty:age         0.012537   0.003532   3.550 0.000426 ***
## beauty:minority   -0.139384   0.098975  -1.408 0.159738
## beauty:nonenglish  0.286041   0.358166   0.799 0.424925
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5246 on 453 degrees of freedom
## Multiple R-squared:  0.1236, Adjusted R-squared:  0.1062
## F-statistic:   7.1 on 9 and 453 DF,  p-value: 1.211e-09
```

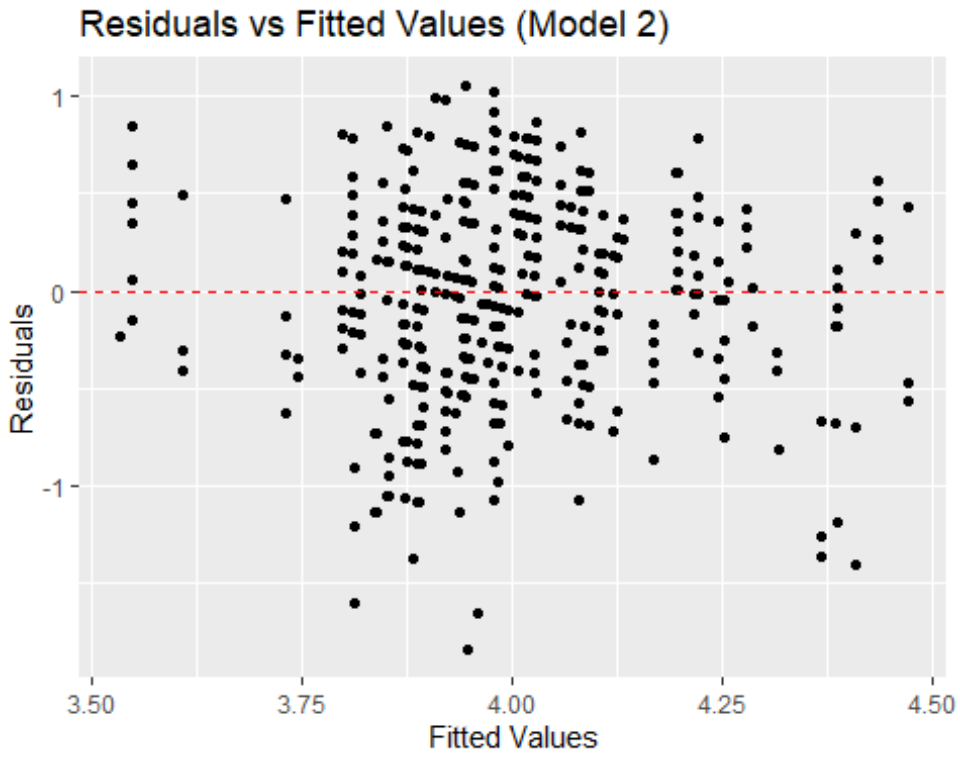
```r
# Create a table comparing the coefficients of each model using broom's
tidy() function
coef_model1 <- tidy(model1) %>% mutate(model = "Model 1")
coef_model2 <- tidy(model2) %>% mutate(model = "Model 2")
coef_model3 <- tidy(model3) %>% mutate(model = "Model 3")

# Combine the coefficients from all models
coef_comparison <- bind_rows(coef_model1, coef_model2, coef_model3)


coef_combined <- coef_comparison %>% filter(term != "(Intercept)")


# Print the comparison of coefficients across models
coef_comparison %>%
  select(term, estimate, std.error, statistic, p.value, model) %>%
  arrange(term, model) %>%
  print()
```

```
## # A tibble: 23 × 6
##    term        estimate std.error statistic   p.value model
##    <chr>          <dbl>     <dbl>     <dbl>     <dbl> <chr>
##  1 (Intercept)  4.25      0.143      29.8   2.41e-109 Model 1
##  2 (Intercept)  4.23      0.143      29.7   1.79e-108 Model 2
##  3 (Intercept)  4.21      0.142      29.6   6.13e-108 Model 3
##  4 age         -0.00271   0.00275    -0.984 3.25e-  1 Model 1
##  5 age         -0.00217   0.00276    -0.785 4.33e-  1 Model 2
##  6 age         -0.00120   0.00275    -0.435 6.64e-  1 Model 3
##  7 beauty       0.141     0.0329      4.28  2.29e-  5 Model 1
##  8 beauty       0.194     0.0451      4.30  2.10e-  5 Model 2
##  9 beauty      -0.443     0.188      -2.36  1.89e-  2 Model 3
## 10 beauty:age   0.0125    0.00353     3.55  4.26e-  4 Model 3
## # ℹ 13 more rows
```

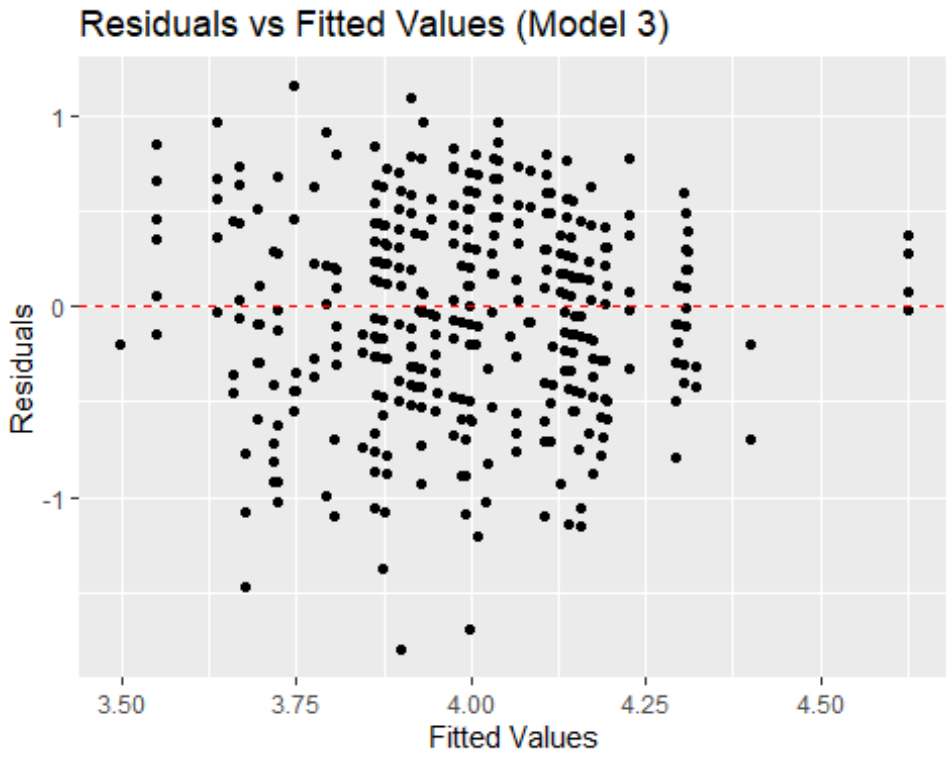
```r
# Plot residuals vs fitted values for each model

# Model 1 Residuals vs Fitted
residuals1 <- resid(model1)
fitted_values1 <- fitted(model1)

ggplot(data.frame(fitted = fitted_values1, residuals = residuals1), aes(x =
fitted, y = residuals)) +
  geom_point() +
  geom_hline(yintercept = 0, linetype = "dashed", color = "red") +
  labs(title = "Residuals vs Fitted Values (Model 1)", x = "Fitted Values", y
= "Residuals")
```



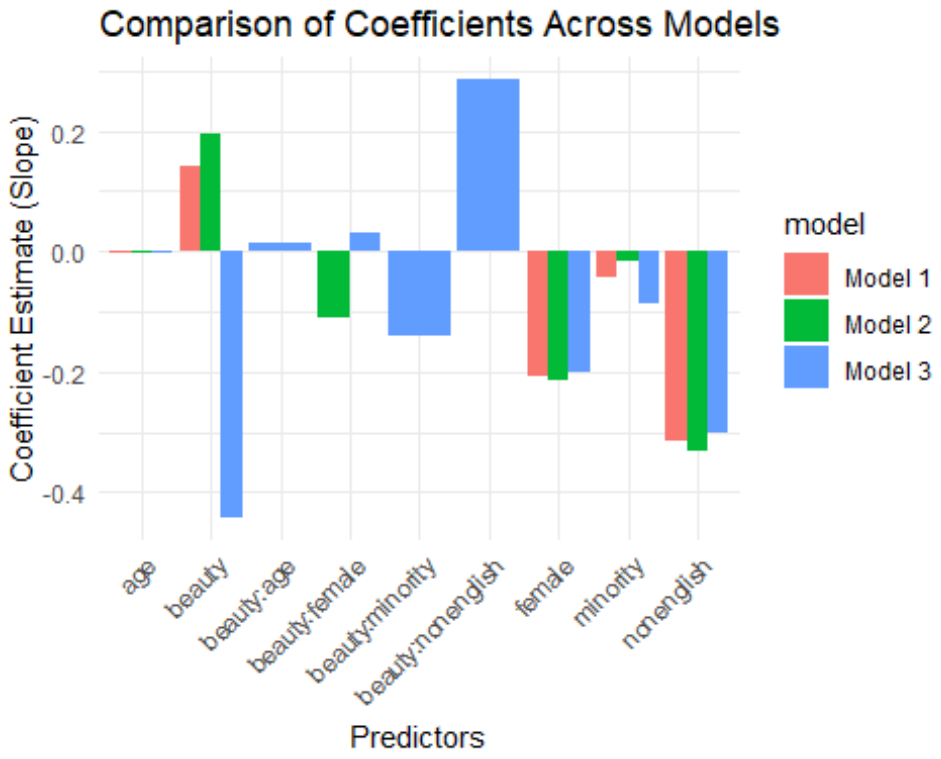

```r
# Model 2 Residuals vs Fitted
residuals2 <- resid(model2)
fitted_values2 <- fitted(model2)

ggplot(data.frame(fitted = fitted_values2, residuals = residuals2), aes(x =
fitted, y = residuals)) +
  geom_point() +
  geom_hline(yintercept = 0, linetype = "dashed", color = "red") +
  labs(title = "Residuals vs Fitted Values (Model 2)", x = "Fitted Values", y
= "Residuals")
```




```r
# Model 3 Residuals vs Fitted
residuals3 <- resid(model3)
fitted_values3 <- fitted(model3)

ggplot(data.frame(fitted = fitted_values3, residuals = residuals3), aes(x =
fitted, y = residuals)) +
  geom_point() +
  geom_hline(yintercept = 0, linetype = "dashed", color = "red") +
  labs(title = "Residuals vs Fitted Values (Model 3)", x = "Fitted Values", y
= "Residuals")
```




```r
ggplot(coef_combined, aes(x = term, y = estimate, fill = model)) +
  geom_bar(stat = "identity", position = position_dodge()) +
  labs(title = "Comparison of Coefficients Across Models",
       x = "Predictors",
       y = "Coefficient Estimate (Slope)") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



a: Coefficient and residual standard deviation Interpretation 1. Beauty: 0.141. For every unit increase in the beauty rating, the course evaluation is expected to increase by approximately 0.141, holding all other variables constant. This indicates a positive effect of perceived beauty on evaluations. 2. Age: -0.0027. There is a slight negative association between age and course evaluation, but the effect is small and not statistically significant. 3. Female: -0.207. Female instructors receive evaluations that are, on average, 0.207 points lower than male instructors, holding other factors constant. This result is statistically significant. 4, Minority: -0.044. Minority instructors receive evaluations that are 0.044 points lower than non-minority instructors, but this effect is not statistically significant. 5. Non-English: -0.313. Instructors who are non-native English speakers tend to receive evaluations that are 0.313 points lower than native English speakers, a statistically significant result.

Residual Standard Deviation: 0.532. This indicates the typical deviation of observed evaluations from the predicted values by the model.

b: Interpretation of coefficients for other models including interactions

Model 2:

1. Beauty: 0.194. The effect of beauty on course evaluations for male instructors (reference group) is a positive increase of 0.194 per unit increase in beauty.
2. Female: -0.214. Female instructors receive evaluations that are 0.214 points lower than male instructors when beauty is held constant.
3. Female:Beauty : -0.111. This indicates that for female instructors, the effect of beauty is smaller. In particular, for every unit increase in beauty, female instructors

receive an increase of 0.111 points less compared to male instructors. This interaction is marginally significant.

4. Other coefficients remain similar to the first model.

Model 3:

1. Beauty: -0.443. In this model, the direct effect of beauty is negative. However, its interactions with other variables need to be considered for interpretation.
2. female:Beauty: 0.0125. This suggests that the effect of beauty increases with age. Older instructors benefit more from beauty in their evaluations.
3. minority:Beauty: -0.139. The negative interaction indicates that for minority instructors, beauty has a smaller positive effect on evaluations.
4. Beauty:nonenglish: 0.286. Non-native English speakers benefit more from beauty in their evaluations.

Residual standard error: 0.524. This model has a slightly lower residual standard error, indicating a better fit than the previous two models.

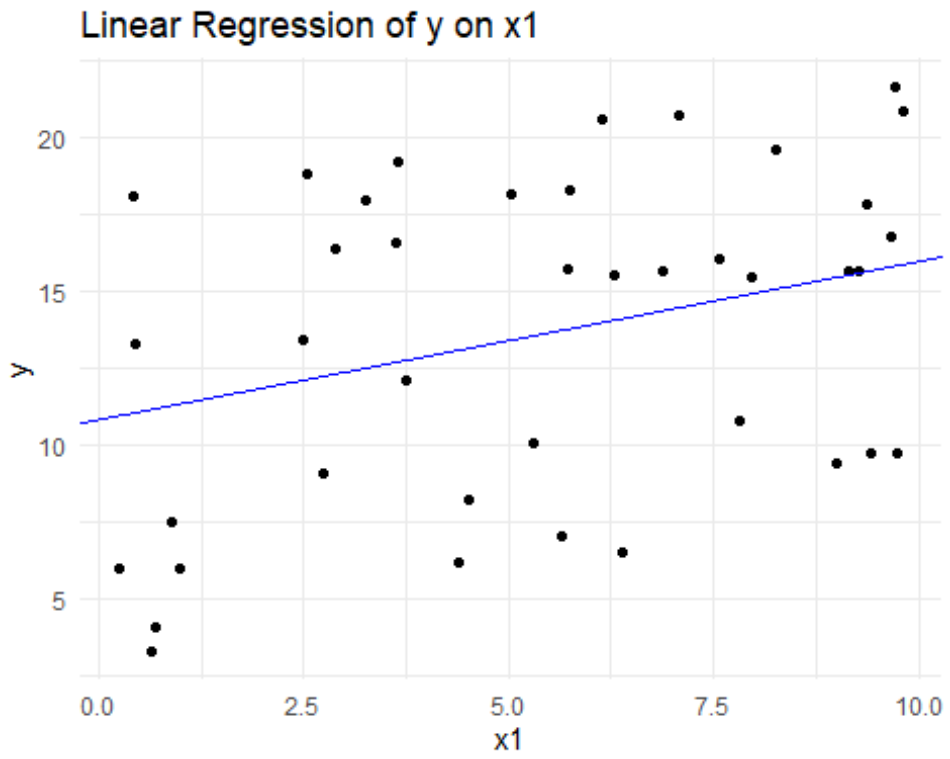
#11.5

Residuals and predictions: The folder Pyth contains outcome y and predictors x1, x2 for 40 data points, with a further 20 points with the predictors but no observed outcome. Save the file to your working directory, then read it into R using read.table(). (a) Use R to fit a linear regression model predicting y from x1, x2, using the first 40 data points in the file. Summarize the inferences and check the fit of your model. (b) Display the estimated model graphically as in Figure 11.2. (c) Make a residual plot for this model. Do the assumptions appear to be met? (d) Make predictions for the remaining 20 data points in the file. How confident do you feel about these predictions?

```
# (a) Read the data with proper column names
data <- read.table("./Other Data/HW_3/pyth.txt", header = TRUE)

# Convert columns to numeric
data$x1 <- as.numeric(data$x1)
data$x2 <- as.numeric(data$x2)
data$y <- as.numeric(data$y)
train_data <- data[1:40, ]
test_data <- data[41:60, c("x1", "x2")]

# (a) Fit the linear regression model using the first 40 data points
model <- lm(y ~ x1 + x2, data = train_data)

summary(model)

##
## Call:
## lm(formula = y ~ x1 + x2, data = train_data)
##
## Residuals:
```

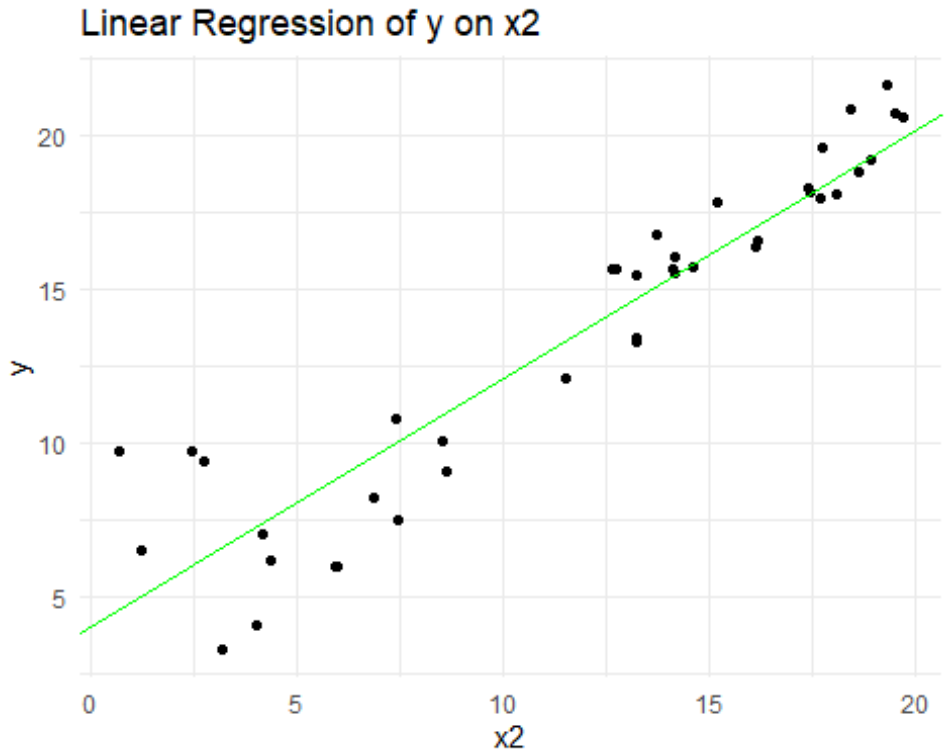
```
##     Min      1Q  Median      3Q     Max
## -0.9585 -0.5865 -0.3356  0.3973  2.8548
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.31513    0.38769   3.392  0.00166 **
## x1           0.51481    0.04590  11.216 1.84e-13 ***
## x2           0.80692    0.02434  33.148  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9 on 37 degrees of freedom
## Multiple R-squared:  0.9724, Adjusted R-squared:  0.9709
## F-statistic: 652.4 on 2 and 37 DF,  p-value: < 2.2e-16
```

```r
# (b) Display the estimated model graphically for 'x1' with regression line
ggplot(train_data, aes(x = x1, y = y)) +
  geom_point() +
  geom_abline(intercept = coef(model)[1] + coef(model)[3] *
mean(train_data$x2), slope = coef(model)[2], color = "blue") +  # Add
regression line
  labs(title = "Linear Regression of y on x1", x = "x1", y = "y") +
  theme_minimal()
```

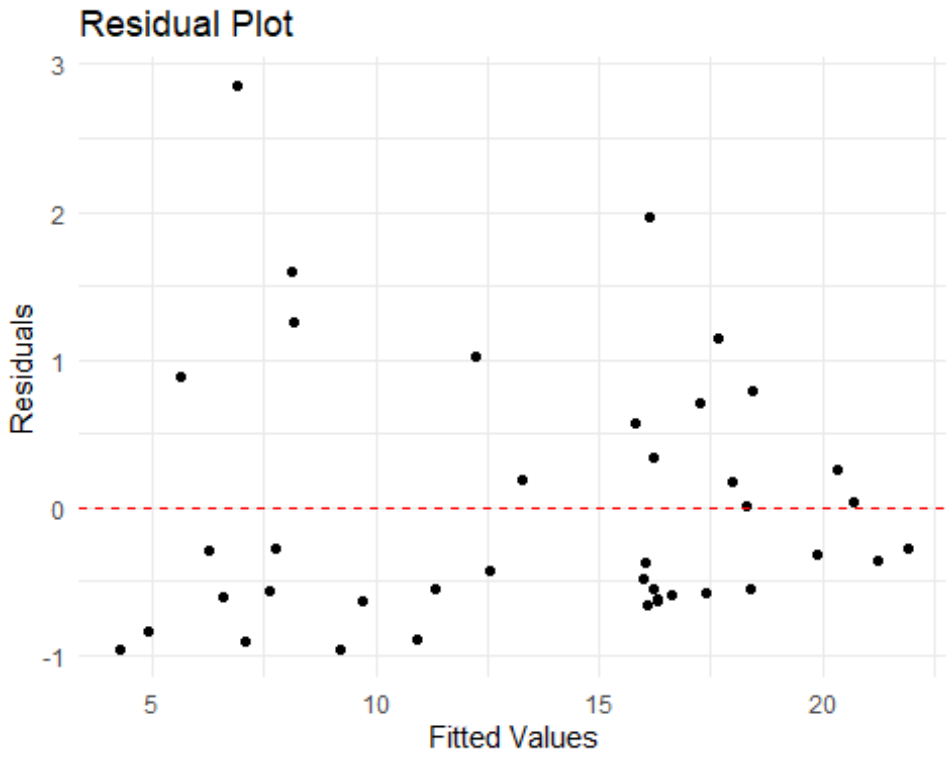



```r
# (b) Display the estimated model graphically for 'x2' with regression line
ggplot(train_data, aes(x = x2, y = y)) +
  geom_point() +
```

```
  geom_abline(intercept = coef(model)[1] + coef(model)[2] *
mean(train_data$x1), slope = coef(model)[3], color = "green") +  # Add
regression line
  labs(title = "Linear Regression of y on x2", x = "x2", y = "y") +
  theme_minimal()
```




```
# (c) Residual plot: plot residuals against fitted values
residuals <- resid(model)
fitted_values <- fitted(model)

ggplot(data.frame(fitted = fitted_values, residuals = residuals), aes(x =
fitted, y = residuals)) +
  geom_point() +
  geom_hline(yintercept = 0, linetype = "dashed", color = "red") +
  labs(title = "Residual Plot", x = "Fitted Values", y = "Residuals") +
  theme_minimal()
```




```
# (d) Predictions for the remaining 20 data points (test_data)
predictions <- predict(model, newdata = test_data)

# Print the predictions
print(predictions)
```

```
##        41        42        43        44        45        46        47
48
## 14.812484 19.142865  5.916816 10.530475 19.012485 13.398863  4.829144
9.145767
##        49        50        51        52        53        54        55
56
##  5.892489 12.338639 18.908561 16.064649  8.963122 14.972786  5.859744
7.374900
##        57        58        59        60
##  4.535267 15.133280  9.100899 16.084900
```

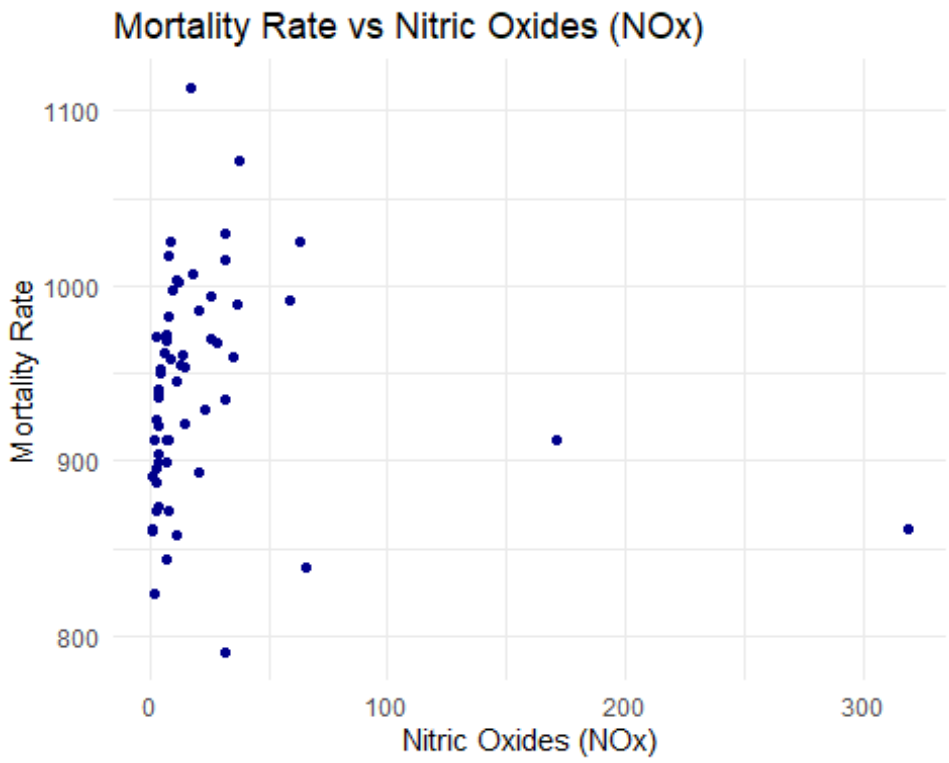
#12.6 12.6 Logarithmic transformations: The folder Pollution contains mortality rates and various environmental factors from 60 U.S. metropolitan areas (see McDonald and Schwing, 1973). For this exercise we shall model mortality rate given nitric oxides, sulfur dioxide, and hydrocarbons as inputs. This model is an extreme oversimplification, as it combines all sources of mortality and does not adjust for crucial factors such as age and smoking. We use it to illustrate log transformations in regression. (a) Create a scatterplot of mortality rate versus level of nitric oxides. Do you think linear regression will fit these data well? Fit the regression and evaluate a residual plot from the regression. (b) Find an appropriate transformation that will result in data more appropriate for linear regression.

Fit a regression to the transformed data and evaluate the new residual plot. (c) Interpret the slope coefficient from the model you chose in (b) (d) Now fit a model predicting mortality rate using levels of nitric oxides, sulfur dioxide, and hydrocarbons as inputs. Use appropriate transformations when helpful. Plot the fitted regression model and interpret the coefficients. (e) Cross validate: fit the model you chose above to the first half of the data and then predict for the second half. You used all the data to construct the model in (d), so this is not really cross validation, but it gives a sense of how the steps of cross validation can be implemented.

```r
# Load necessary libraries
library(ggplot2)
library(dplyr)

# Load and Inspect the Dataset

# Define column names
column_names <- c("prec", "jant", "jult", "ovr65", "popn", "educ", "hous",
"dens",
                  "nonw", "wwdrk", "poor", "hc", "nox", "so2", "humid",
"mort")

# Load the dataset
data <- read.csv("./Other Data/HW_3/pollution.csv", header = TRUE, col.names
= column_names)

# Inspect the first few rows of the dataset
head(data)
```

```
##   prec jant jult ovr65 popn educ hous dens nonw wwdrk poor hc nox so2
humid
## 1   36   27   71   8.1 3.34 11.4 81.5 3243  8.8  42.6 11.7 21  15  59
59
## 2   35   23   72  11.1 3.14 11.0 78.8 4281  3.5  50.7 14.4  8  10  39
57
## 3   44   29   74  10.4 3.21  9.8 81.6 4260  0.8  39.4 12.4  6   6  33
54
## 4   47   45   79   6.5 3.41 11.1 77.5 3125 27.1  50.2 20.6 18   8  24
56
## 5   43   35   77   7.6 3.44  9.6 84.6 6441 24.4  43.7 14.3 43  38 206
55
## 6   53   45   80   7.7 3.45 10.2 66.8 3325 38.5  43.1 25.5 30  32  72
54
##       mort
## 1  921.870
## 2  997.875
## 3  962.354
## 4  982.291
## 5 1071.289
## 6 1030.380
```

```r
# Part (a): Scatterplot and Linear Regression

# Create scatterplot of mortality rate vs nitric oxides (nox)
ggplot(data, aes(x = nox, y = mort)) +
  geom_point(color = "darkblue") +
  labs(title = "Mortality Rate vs Nitric Oxides (NOx)",
       x = "Nitric Oxides (NOx)",
       y = "Mortality Rate") +
  theme_minimal()
```

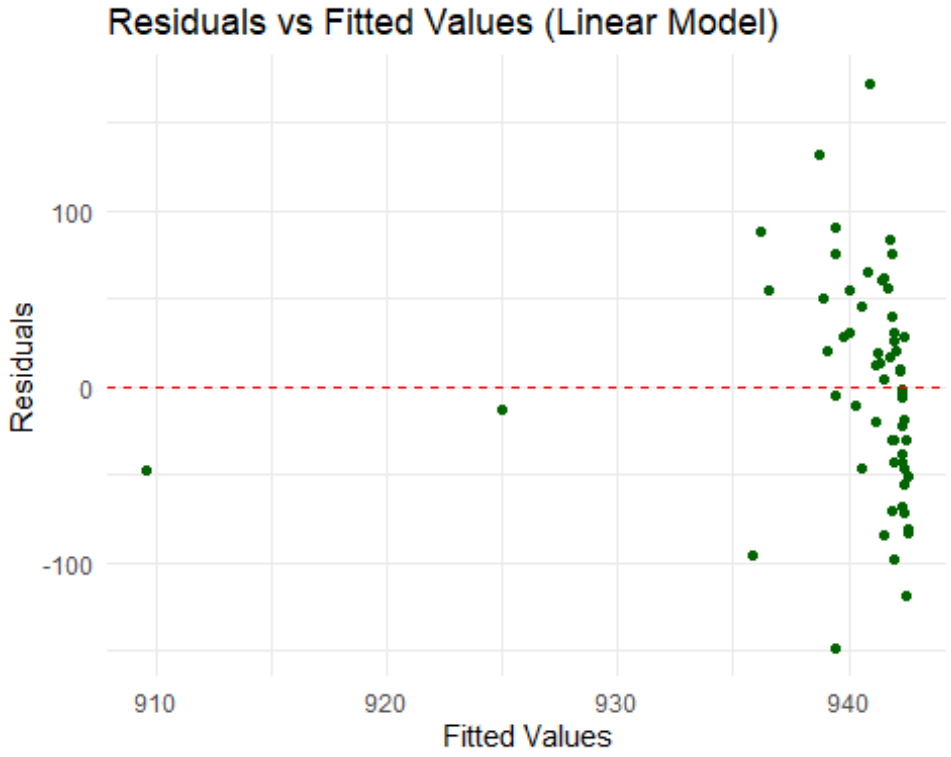



```r
# Fit a simple linear regression model: mort ~ nox
linear_model <- lm(mort ~ nox, data = data)

# Summary of the linear regression model
summary(linear_model)
```

```
##
## Call:
## lm(formula = mort ~ nox, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -148.654  -43.710    1.751   41.663  172.211
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 942.7115     9.0034 104.706   <2e-16 ***
```

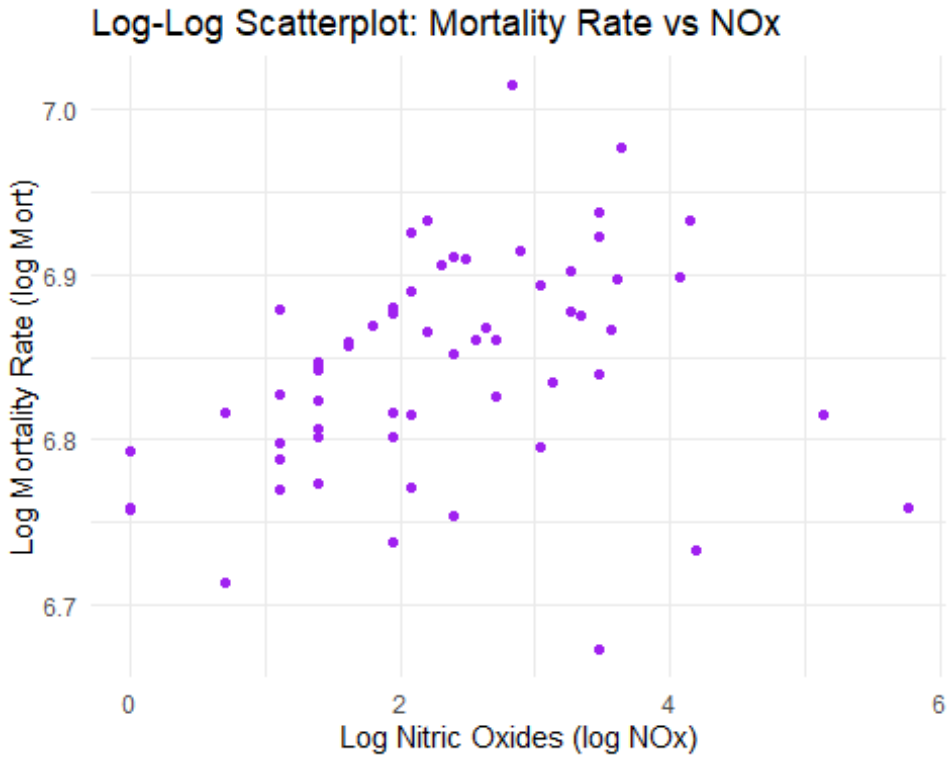
```
## nox          -0.1039     0.1758  -0.591    0.557
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 62.55 on 58 degrees of freedom
## Multiple R-squared:  0.005987,   Adjusted R-squared:  -0.01115
## F-statistic: 0.3494 on 1 and 58 DF,  p-value: 0.5568
```

```r
# Residuals vs Fitted Values Plot
ggplot(data, aes(x = fitted(linear_model), y = resid(linear_model))) +
  geom_point(color = "darkgreen") +
  geom_hline(yintercept = 0, linetype = "dashed", color = "red") +
  labs(title = "Residuals vs Fitted Values (Linear Model)",
       x = "Fitted Values",
       y = "Residuals") +
  theme_minimal()
```




```r
# Part (b): Logarithmic Transformation and Regression

# Check for zero or negative values before log transformation
if(any(data$mort <= 0) | any(data$nox <= 0)) {
  stop("Mortality rate and NOx levels must be positive for log
transformation.")
}

# Apply logarithmic transformation to both response and predictor
data <- data %>%
```

```r
  mutate(log_mort = log(mort),
         log_nox = log(nox))

# Create scatterplot of log-transformed variables
ggplot(data, aes(x = log_nox, y = log_mort)) +
  geom_point(color = "purple") +
  labs(title = "Log-Log Scatterplot: Mortality Rate vs NOx",
       x = "Log Nitric Oxides (log NOx)",
       y = "Log Mortality Rate (log Mort)") +
  theme_minimal()
```

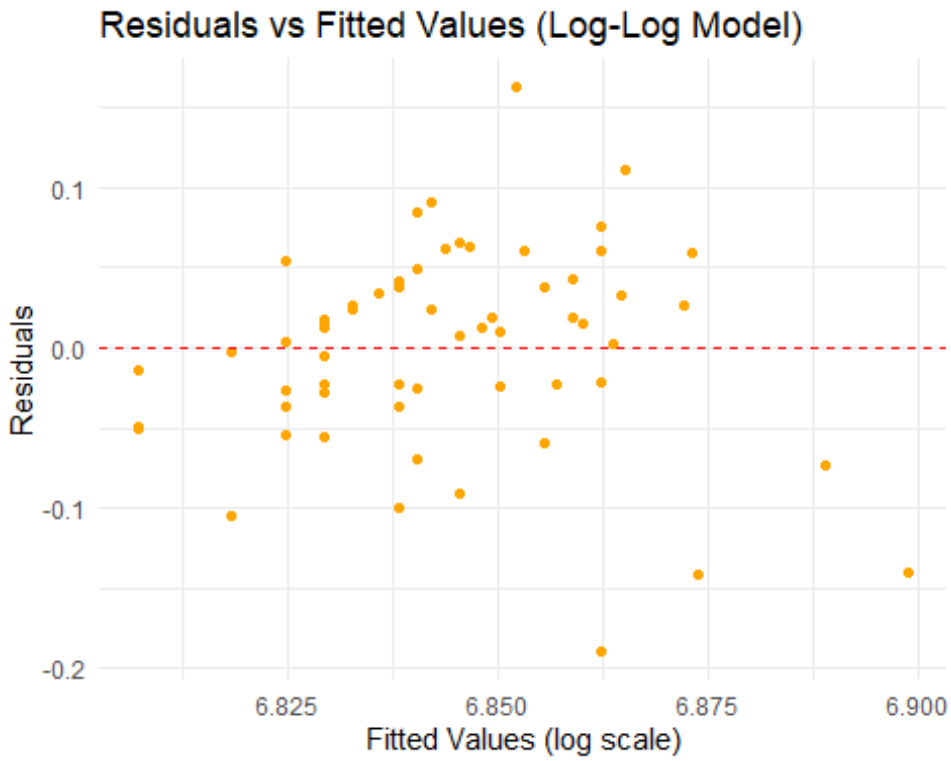



```r
# Fit a linear regression model on log-transformed data
log_linear_model <- lm(log_mort ~ log_nox, data = data)

# Summary of the log-log regression model
summary(log_linear_model)
```

```
##
## Call:
## lm(formula = log_mort ~ log_nox, data = data)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.18930 -0.02957  0.01132  0.03897  0.16275
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept) 6.807175   0.018349 370.975   <2e-16 ***
## log_nox     0.015893   0.007048   2.255   0.0279 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.06412 on 58 degrees of freedom
## Multiple R-squared:  0.08061,    Adjusted R-squared:  0.06476
## F-statistic: 5.085 on 1 and 58 DF,  p-value: 0.02792
```

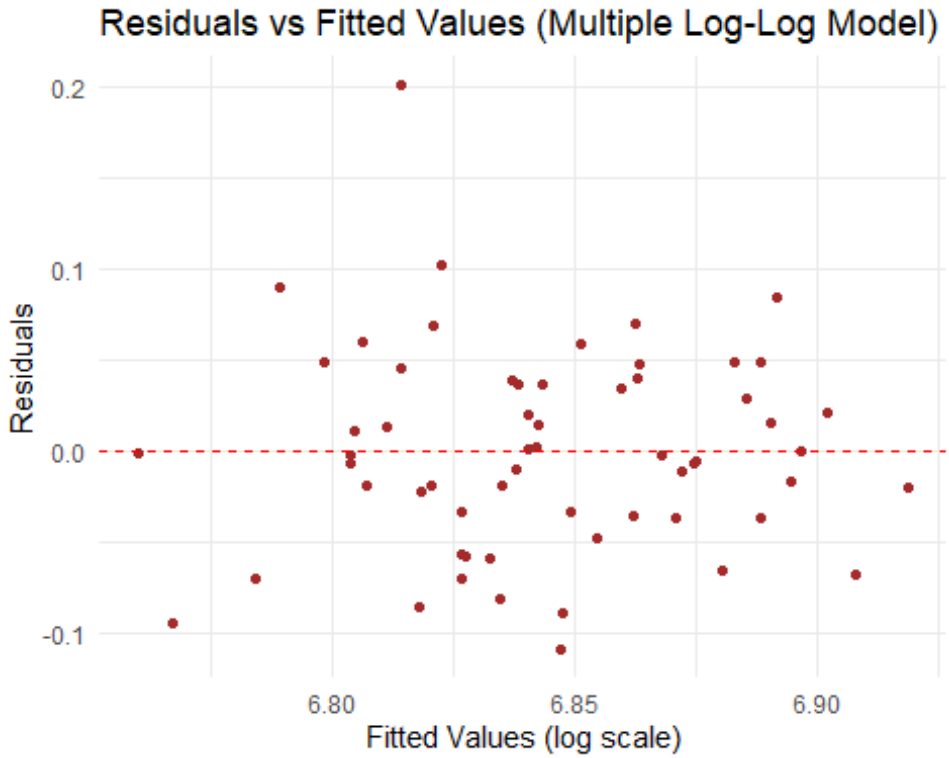
```r
# Residuals vs Fitted Values Plot for Log-Transformed Model
ggplot(data, aes(x = fitted(log_linear_model), y = resid(log_linear_model)))
+
  geom_point(color = "orange") +
  geom_hline(yintercept = 0, linetype = "dashed", color = "red") +
  labs(title = "Residuals vs Fitted Values (Log-Log Model)",
       x = "Fitted Values (log scale)",
       y = "Residuals") +
  theme_minimal()
```




```r
# Part (c): Interpretation of the Slope Coefficient

# Extract slope coefficient
slope_coef <- coef(log_linear_model)["log_nox"]

# Interpretation statement
cat(sprintf("Interpretation of Slope Coefficient:\nA 1%% increase in nitric
oxides (NOx) level is associated with approximately %.2f%% increase in
```

```
mortality rate, holding other factors constant.\n",
            slope_coef * 100))

## Interpretation of Slope Coefficient:
## A 1% increase in nitric oxides (NOx) level is associated with
approximately 1.59% increase in mortality rate, holding other factors
constant.

# Part (d): Multiple Regression with Transformed Data

# Check for zero or negative values in additional predictors
if(any(data$so2 <= 0) | any(data$hc <= 0)) {
  stop("SO2 and HC levels must be positive for log transformation.")
}

# Apply logarithmic transformation to additional predictors
data <- data %>%
  mutate(log_so2 = log(so2),
         log_hc = log(hc))

# Fit multiple linear regression model on log-transformed data
multi_log_model <- lm(log_mort ~ log_nox + log_so2 + log_hc, data = data)

# Summary of the multiple regression model
summary(multi_log_model)

##
## Call:
## lm(formula = log_mort ~ log_nox + log_so2 + log_hc, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.10874 -0.03574 -0.00218  0.03709  0.20085
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.826749   0.022701 300.726  < 2e-16 ***
## log_nox      0.059837   0.023021   2.599  0.01192 *
## log_so2      0.014309   0.007584   1.887  0.06436 .
## log_hc      -0.060812   0.020553  -2.959  0.00452 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.05753 on 56 degrees of freedom
## Multiple R-squared:  0.2852, Adjusted R-squared:  0.2469
## F-statistic: 7.449 on 3 and 56 DF,  p-value: 0.0002777

# Residuals vs Fitted Values Plot for Multiple Log-Transformed Model
ggplot(data, aes(x = fitted(multi_log_model), y = resid(multi_log_model))) +
  geom_point(color = "brown") +
```

```
  geom_hline(yintercept = 0, linetype = "dashed", color = "red") +
  labs(title = "Residuals vs Fitted Values (Multiple Log-Log Model)",
       x = "Fitted Values (log scale)",
       y = "Residuals") +
  theme_minimal()
```




```
# Part (e): Cross-Validation

# Split the data into first half and second half
set.seed(123) # For reproducibility
n <- nrow(data)
half <- floor(n / 2)

# Ensure data is shuffled before splitting to avoid any ordering bias
data_shuffled <- data %>% sample_frac(1)

first_half <- data_shuffled[1:half, ]
second_half <- data_shuffled[(half + 1):n, ]

# Fit the multiple regression model on the first half
cv_model <- lm(log_mort ~ log_nox + log_so2 + log_hc, data = first_half)

# Predict on the second half
cv_predictions_log <- predict(cv_model, newdata = second_half)

# Convert predictions back to the original scale
```

```r
cv_predictions <- exp(cv_predictions_log)

# Actual mortality rates in the second half
actual_mort <- second_half$mort

# Calculate prediction errors (e.g., Mean Absolute Error)
mae <- mean(abs(cv_predictions - actual_mort))
rmse <- sqrt(mean((cv_predictions - actual_mort)^2))

# Output prediction results and error metrics
cat("Cross-Validation Predictions:\n")
```

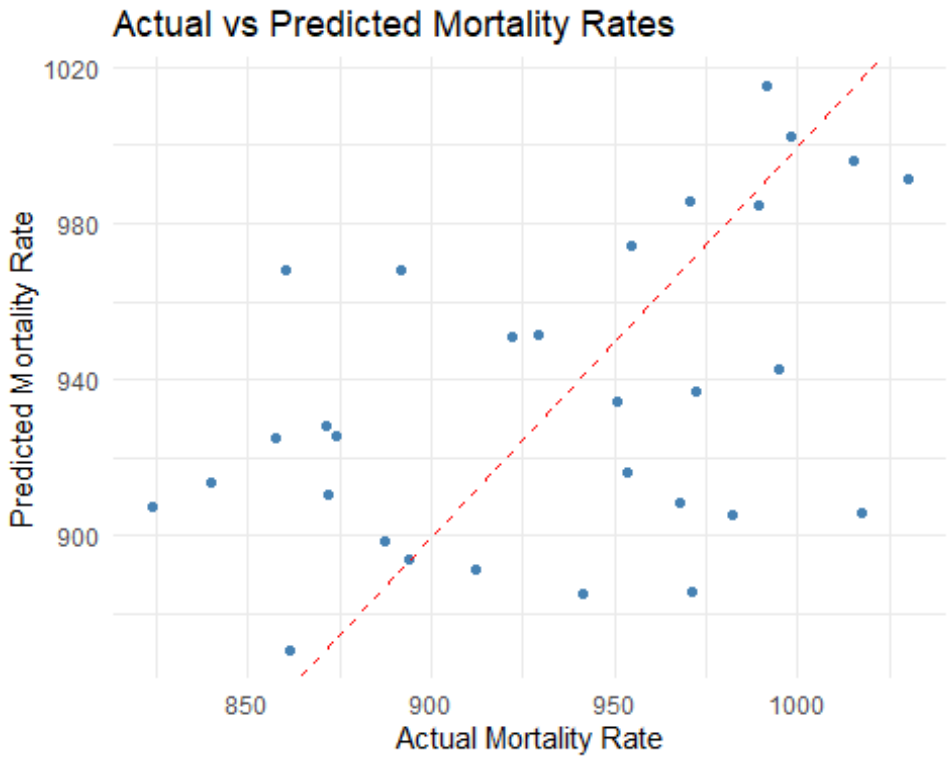
```
## Cross-Validation Predictions:
```

```r
print(data.frame(Actual = actual_mort, Predicted = cv_predictions))
```

```
##      Actual Predicted
## 31  970.467  985.5975
## 32 1015.023  995.7878
## 33  994.648  942.7096
## 34  874.281  925.3358
## 35  954.442  974.4098
## 36  971.122  885.7972
## 37  891.708  968.0315
## 38  972.464  937.0718
## 39  887.466  898.3322
## 40 1030.380  991.2014
## 41  997.875 1002.0066
## 42  839.709  913.5646
## 43  893.991  894.1658
## 44  929.150  951.2347
## 45  861.439  870.5871
## 46  989.265  984.5507
## 47  941.181  885.0865
## 48  921.870  951.0910
## 49  953.560  916.2000
## 50 1017.613  905.6202
## 51  857.622  925.2661
## 52  871.338  928.0023
## 53  950.672  934.5089
## 54  912.202  891.4351
## 55  860.101  968.0315
## 56  967.803  908.2659
## 57  982.291  905.4423
## 58  823.764  907.2175
## 59  871.766  910.6883
## 60  991.290 1015.1801
```

```r
cat(sprintf("\nPrediction Error Metrics:\nMean Absolute Error (MAE):
%.2f\nRoot Mean Squared Error (RMSE): %.2f\n", mae, rmse))
```

```
##
## Prediction Error Metrics:
## Mean Absolute Error (MAE): 43.50
## Root Mean Squared Error (RMSE): 53.21

# Optional: Plot Actual vs Predicted Mortality Rates
ggplot(data.frame(Actual = actual_mort, Predicted = cv_predictions), aes(x =
Actual, y = Predicted)) +
  geom_point(color = "steelblue") +
  geom_abline(slope = 1, intercept = 0, linetype = "dashed", color = "red") +
  labs(title = "Actual vs Predicted Mortality Rates",
       x = "Actual Mortality Rate",
       y = "Predicted Mortality Rate") +
  theme_minimal()
```



#12.15

```
# Load libraries
library("rprojroot")
library("rstantools")

## This is rstantools version 2.4.0

library("rstanarm")

## Loading required package: Rcpp

## This is rstanarm version 2.32.1
```

```
## - See https://mc-stan.org/rstanarm/articles/priors for changes to default
priors!

## - Default priors may change, so it's safest to specify priors, even if
equivalent to the defaults.

## - For execution on a local, multicore CPU with excess RAM we recommend
calling

##   options(mc.cores = parallel::detectCores())
```

```
library("loo")
```

```
## This is loo version 2.8.0

## - Online documentation and vignettes at mc-stan.org/loo

## - As of v2.0.0 loo defaults to 1 core but we recommend using as many as
possible. Use the 'cores' argument or set options(mc.cores = NUM_CORES) for
an entire session.

## - Windows 10 users: loo may be very slow if 'mc.cores' is set in your
.Rprofile file (see https://github.com/stan-dev/loo/issues/94).
```

```
library("ggplot2")
library("bayesplot")
```

```
## This is bayesplot version 1.11.1

## - Online documentation and vignettes at mc-stan.org/bayesplot

## - bayesplot theme set to bayesplot::theme_default()

##    * Does _not_ affect other ggplot2 plots

##    * See ?bayesplot_theme_set for details on theme setting
```

```
theme_set(bayesplot::theme_default(base_family = "sans"))
```

```
SEED=123
```

```
data_path <- "./Other Data/HW_3/portugese_language.csv"  # Ensure the correct
path and filename
```

```
data <- read.csv(data_path, header = TRUE)
```

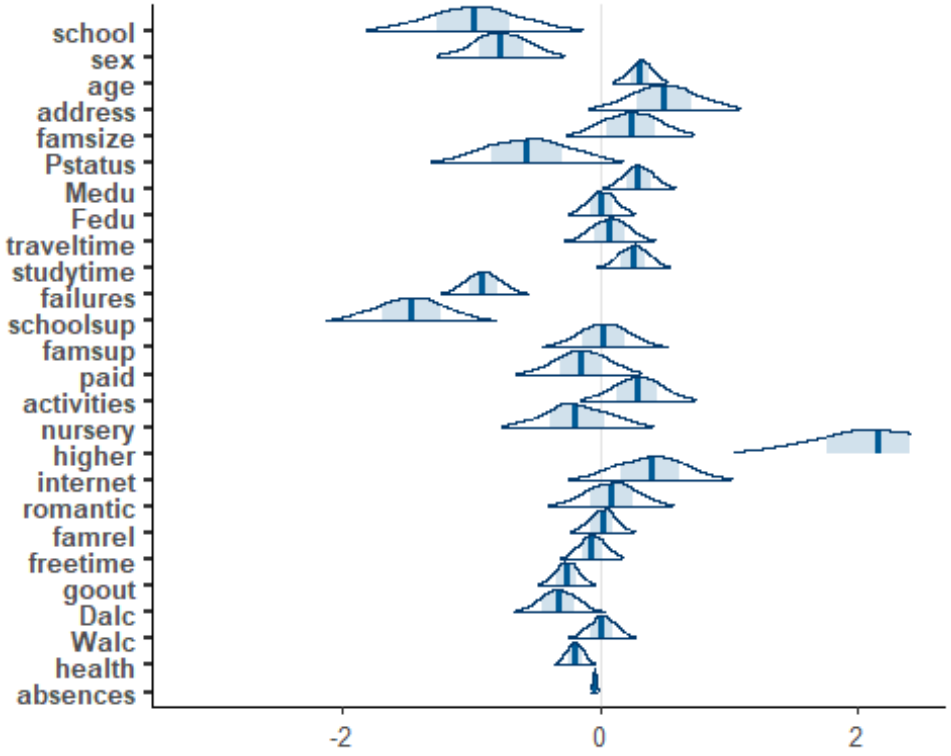
```
# Inspect the first few rows of the dataset
head(data)
```

```
##   G3por school sex age address famsize Pstatus Medu Fedu traveltime
studytime
## 1    13      0   0  15       0       0       1    1    1          2
4
```

```
## 2    11      0   0  15       0       0       1    1    1          1
2
## 3    12      0   0  15       0       0       1    2    2          1
1
## 4    10      0   0  15       0       0       1    2    4          1
3
## 5    13      0   0  15       0       0       1    3    3          2
3
## 6    12      0   0  15       0       0       1    3    4          1
3
##   failures schoolsup famsup paid activities nursery higher internet
romantic
## 1        1         1      1    1          1       1      1        1
0
## 2        2         1      1    0          0       0      1        1
1
## 3        0         1      1    1          1       1      1        0
0
## 4        0         1      1    1          1       1      1        1
0
## 5        2         0      1    1          1       1      1        1
1
## 6        0         1      1    1          1       1      1        1
0
##   famrel freetime goout Dalc Walc health absences
## 1      3        1     2    1    1      1        2
## 2      3        3     4    2    4      5        2
## 3      4        3     1    1    1      2        8
## 4      4        3     2    1    1      5        2
## 5      4        2     1    2    3      3        8
## 6      4        3     2    1    1      5        2
```

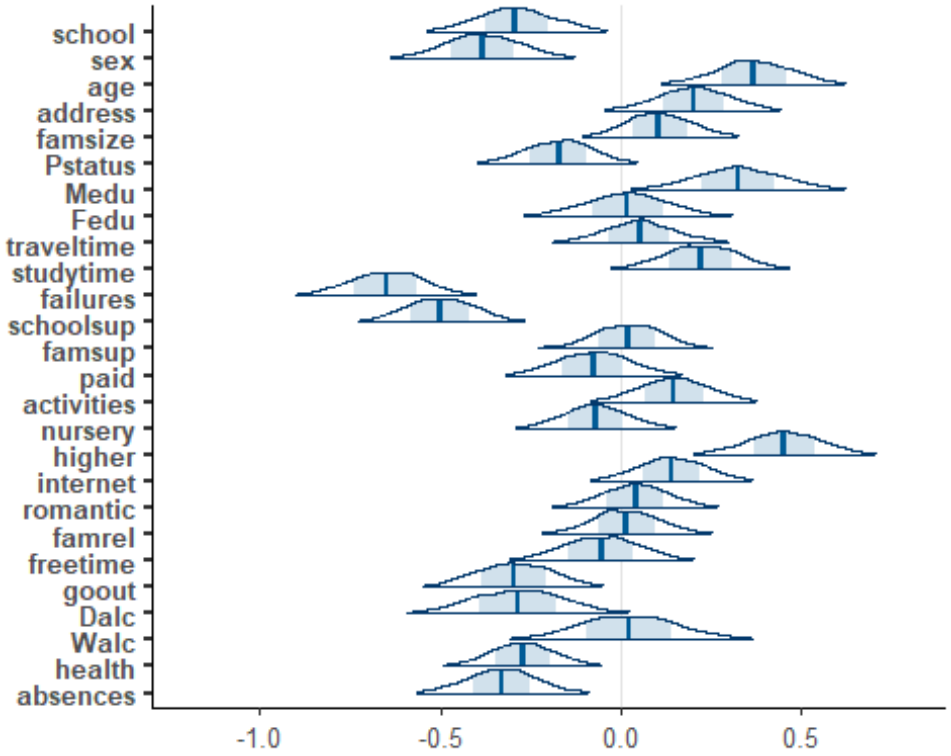
```r
# Define the predictors as per Section 12.7
predictors <- c("school", "sex", "age", "address", "famsize", "Pstatus",
                "Medu", "Fedu", "traveltime", "studytime", "failures",
                "schoolsup", "famsup", "paid", "activities", "nursery",
                "higher", "internet", "romantic", "famrel", "freetime",
                "goout", "Dalc", "Walc", "health", "absences")

p <- length(predictors)


# Define the outcome variable
outcome <- "G3por"  # Assuming the Portuguese grade column is named 'G3port'

data_G3por <- subset(data, subset=G3por>0, select=c("G3por",predictors))
n <- nrow(data_G3por)

fit0 <- stan_glm(G3por ~ ., data = data_G3por, refresh=0)
p0 <- mcmc_areas(as.matrix(fit0), pars=vars(-'(Intercept)',-sigma),
```

```
                  prob_outer=0.95, area_method = "scaled height") +
  xlim(c(-3.2,2.4))

## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.

p0 <- p0 + scale_y_discrete(limits = rev(levels(p0$data$parameter)))

## Scale for y is already present.
## Adding another scale for y, which will replace the existing scale.

p0

## Warning: Removed 1 row containing missing values or values outside the
scale range
## (`geom_segment()`).
```




```
datastd_G3por <- data_G3por
datastd_G3por[,predictors] <-scale(data_G3por[,predictors])

fit1 <- stan_glm(G3por ~ ., data = datastd_G3por, seed = SEED, refresh=0)

p1 <- mcmc_areas(as.matrix(fit1), pars=vars(-'(Intercept)',-sigma),
                  prob_outer=0.95, area_method = "scaled height") +
  xlim(c(-1.2,0.8))

## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
```

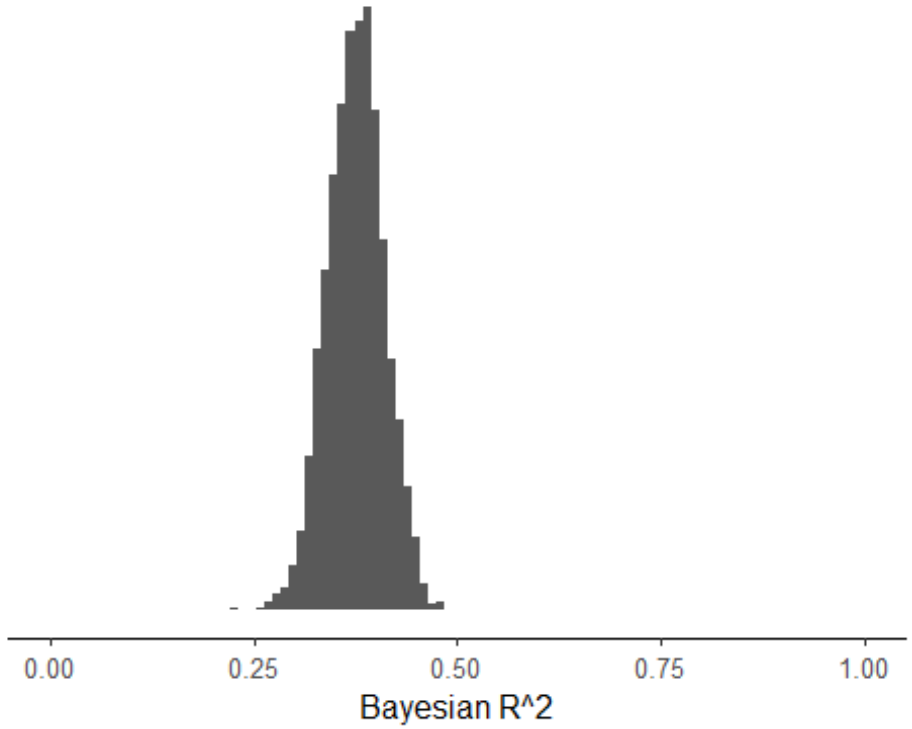
```
p1 <- p1 + scale_y_discrete(limits = rev(levels(p1$data$parameter)))

## Scale for y is already present.
## Adding another scale for y, which will replace the existing scale.

p1
```




```
round(median(bayes_R2(fit1)), 2)

## [1] 0.38

round(median(loo_R2(fit1)), 2)

## [1] 0.26

(loo1 <- loo(fit1))

## Warning: Found 1 observation(s) with a pareto_k > 0.7. We recommend
calling 'loo' again with argument 'k_threshold = 0.7' in order to calculate
the ELPD without the assumption that these observations are negligible. This
will refit the model 1 times to compute the ELPDs for the problematic
observations directly.

##
## Computed from 4000 by 377 log-likelihood matrix.
##
##          Estimate   SE
## elpd_loo   -836.5 16.0
## p_loo        28.5  2.8
```

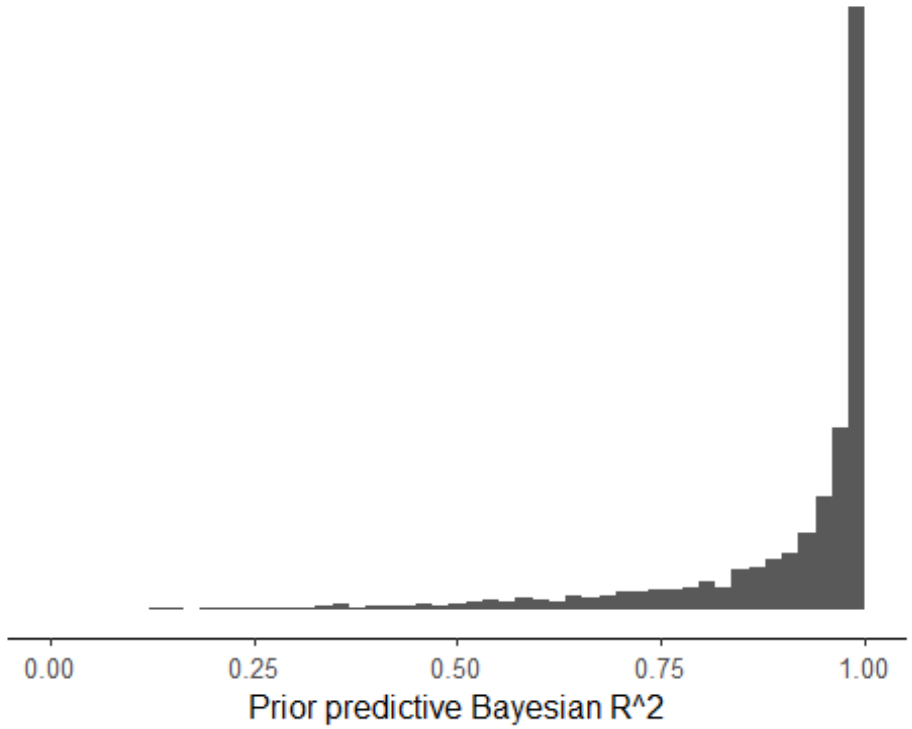
```
## looic      1672.9 32.0
## ------
## MCSE of elpd_loo is NA.
## MCSE and ESS estimates assume MCMC draws (r_eff in [0.5, 2.2]).
##
## Pareto k diagnostic values:
##                          Count Pct.    Min. ESS
## (-Inf, 0.7]   (good)     376   99.7%   545
##    (0.7, 1]   (bad)        1    0.3%   <NA>
##    (1, Inf)   (very bad)   0    0.0%   <NA>
## See help('pareto-k-diagnostic') for details.
```

```
ggplot() + geom_histogram(aes(x=bayes_R2(fit1)),
breaks=seq(0,1,length.out=100)) +
  xlim(c(0,1)) +
  scale_y_continuous(breaks=NULL) +
  labs(x="Bayesian R^2", y="")
```

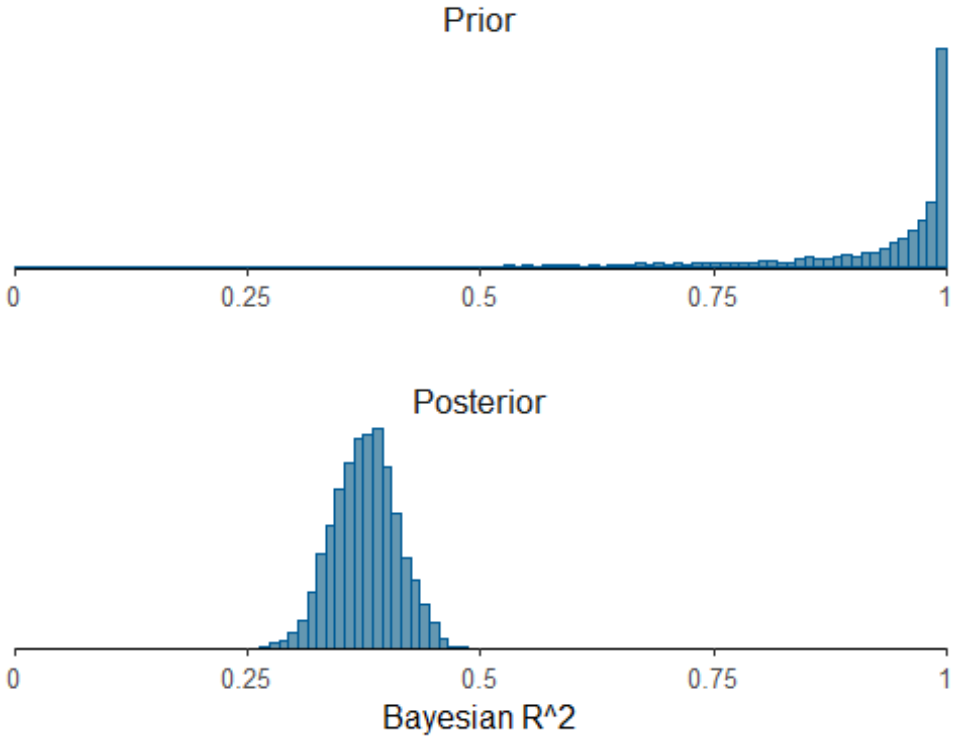



```
ppR2<-numeric()
for (i in 1:4000) {
  sigma2 <- rexp(1,rate=0.3)^2;
  muvar <- var(as.matrix(datastd_G3por[,2:27]) %*% rnorm(26)*2.5)
  ppR2[i] <- muvar/(muvar+sigma2)
}
ggplot()+geom_histogram(aes(x=ppR2), breaks=seq(0,1,length.out=50)) +
  xlim(c(0,1)) +
```

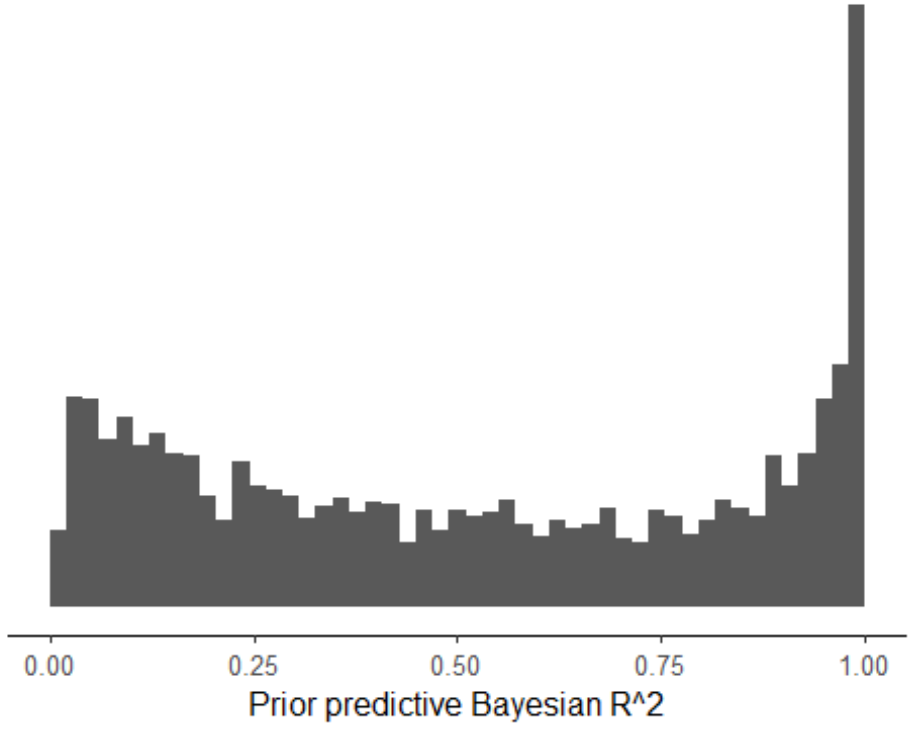
```
  scale_y_continuous(breaks=NULL) +
  labs(x="Prior predictive Bayesian R^2",y="")
```



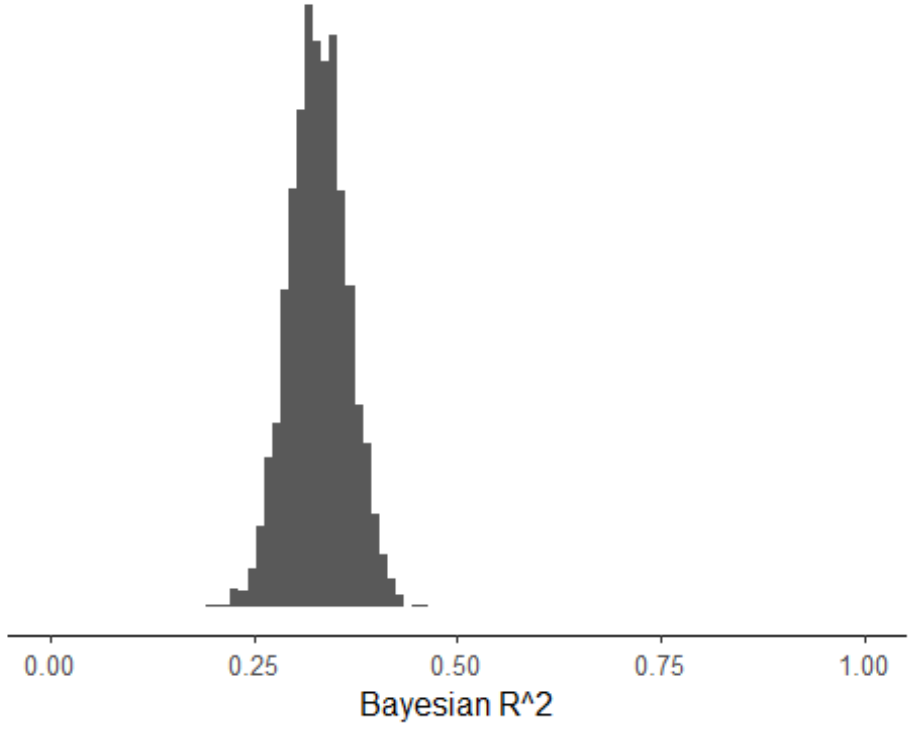

```
pp1 <- mcmc_hist(data.frame(Prior=ppR2,Posterior=bayes_R2(fit1)),
                 breaks=seq(0,1,length.out=100),
                 facet_args = list(nrow = 2)) +
  facet_text(size = 13) +
  scale_x_continuous(limits = c(0,1), expand = c(0, 0),
                     labels = c("0","0.25","0.5","0.75","1")) +
  theme(axis.line.y = element_blank()) +
  xlab("Bayesian R^2")

pp1
```



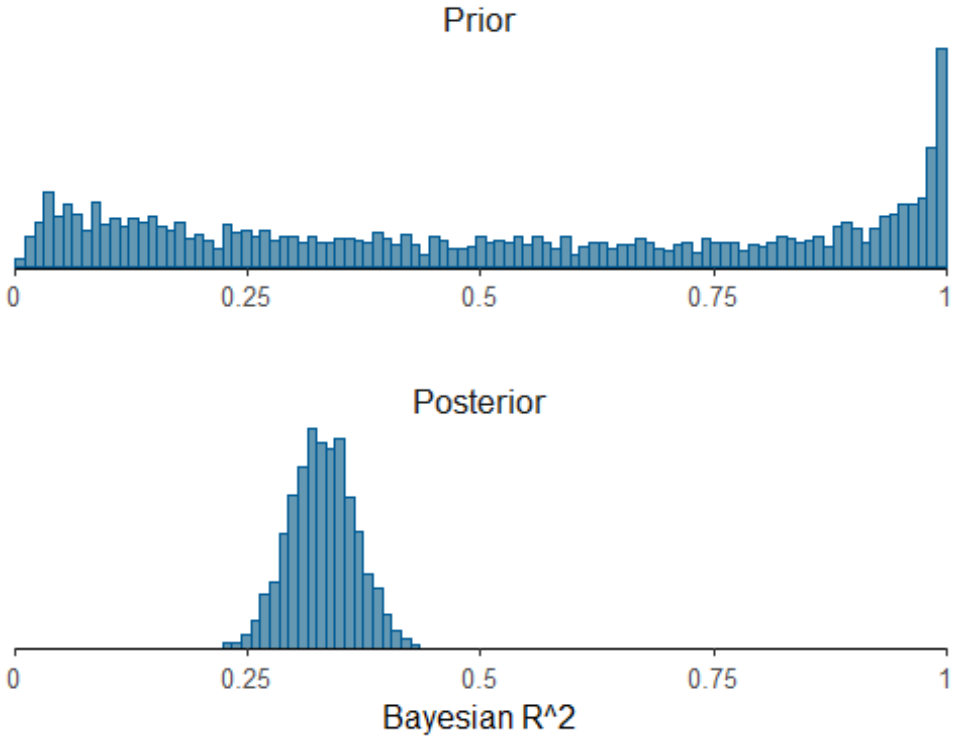

```
ppR2<-numeric()
for (i in 1:4000) {
  sigma2 <- 0.7*rexp(1, rate=1/sd(datastd_G3por$G3por))^2
  muvar <- var(as.matrix(datastd_G3por[,2:27]) %*% rnorm(26,
sd=sd(datastd_G3por$G3por)/sqrt(26)*sqrt(0.3)))
  ppR2[i] <- muvar/(muvar+sigma2)
}
ggplot()+geom_histogram(aes(x=ppR2), breaks=seq(0,1,length.out=50)) +
  xlim(c(0,1)) +
  scale_y_continuous(breaks=NULL) +
  labs(x="Prior predictive Bayesian R^2",y="")
```




```
fit2 <- stan_glm(G3por ~ ., data = datastd_G3por, seed = SEED,

prior=normal(scale=sd(datastd_G3por$G3por)/sqrt(26)*sqrt(0.3),
                                autoscale=FALSE),
                 refresh=0)

round(median(loo_R2(fit2)), 2)

## [1] 0.27

round(median(bayes_R2(fit2)), 2)

## [1] 0.33

ggplot()+geom_histogram(aes(x=bayes_R2(fit2)),
breaks=seq(0,1,length.out=100)) +
  xlim(c(0,1)) +
  scale_y_continuous(breaks=NULL) +
  labs(x="Bayesian R^2",y="")
```




```
pp2 <- mcmc_hist(data.frame(Prior=ppR2,Posterior=bayes_R2(fit2)),
                 breaks=seq(0,1,length.out=100),
                 facet_args = list(nrow = 2)) +
  facet_text(size = 13) +
  scale_x_continuous(limits = c(0,1), expand = c(0, 0),
                     labels = c("0","0.25","0.5","0.75","1")) +
  theme(axis.line.y = element_blank()) +
  xlab("Bayesian R^2")
pp2
```




```
(loo2 <- loo(fit2))
```

```
##
## Computed from 4000 by 377 log-likelihood matrix.
##
##          Estimate   SE
## elpd_loo   -833.3 16.4
## p_loo        24.2  2.5
## looic      1666.6 32.8
## ------
## MCSE of elpd_loo is 0.1.
## MCSE and ESS estimates assume MCMC draws (r_eff in [0.4, 2.5]).
##
## All Pareto k estimates are good (k < 0.7).
## See help('pareto-k-diagnostic') for details.
```

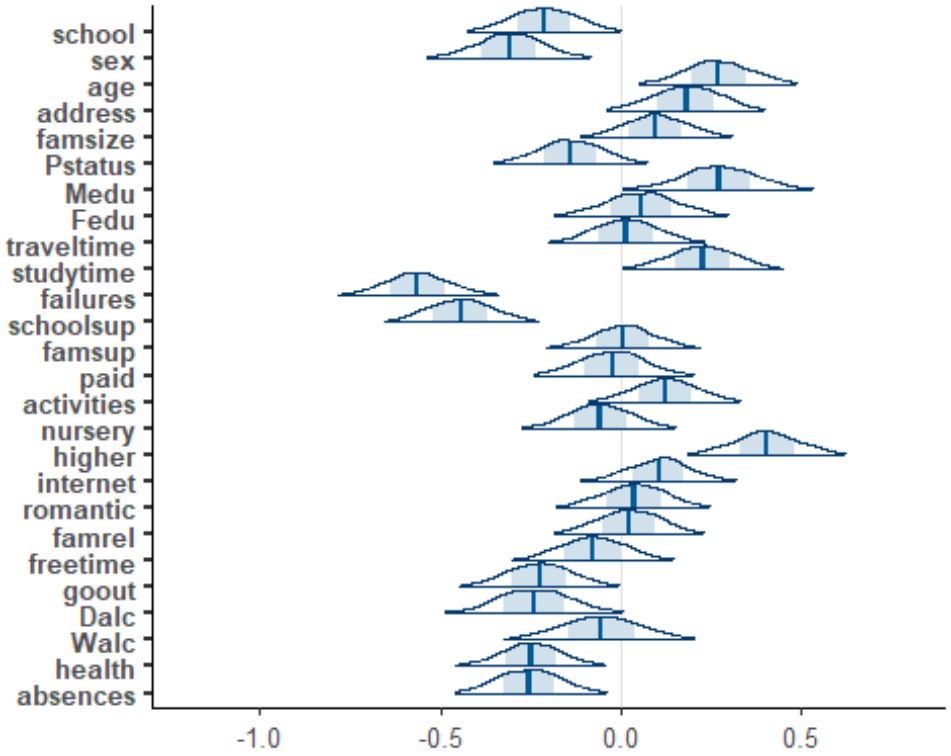
```
loo_compare(loo1,loo2)
```

```
##      elpd_diff se_diff
## fit2  0.0       0.0
## fit1 -3.2       1.8
```

```
p2 <- mcmc_areas(as.matrix(fit2), pars=vars(-'(Intercept)',-sigma),
                 prob_outer=0.95, area_method = "scaled height") +
  xlim(c(-1.2,0.8))
```

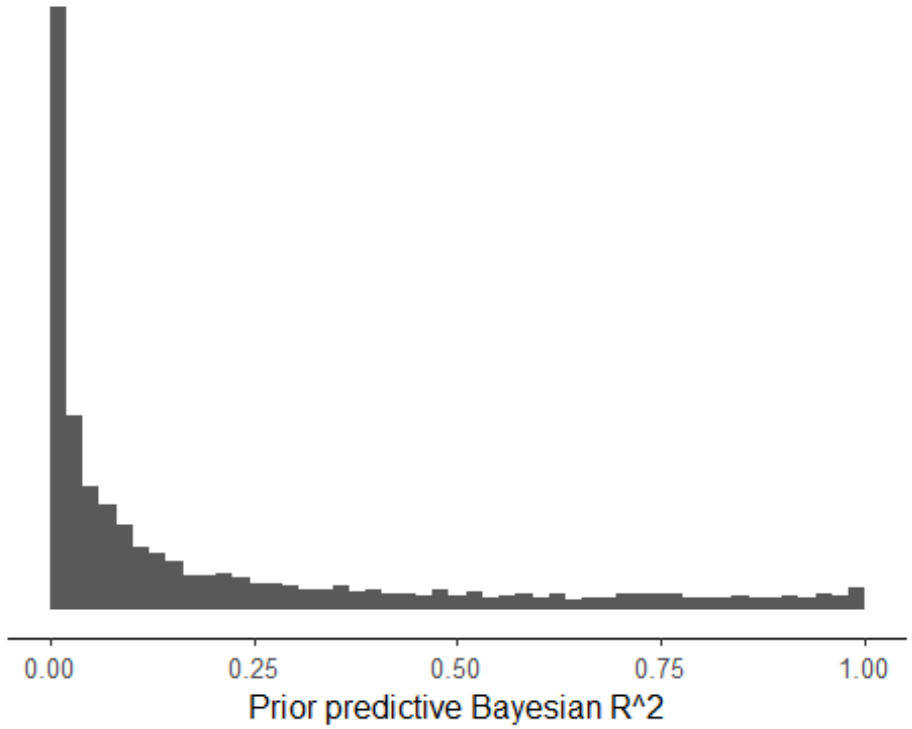
```
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
```

```
p2 <- p2 + scale_y_discrete(limits = rev(levels(p2$data$parameter)))

## Scale for y is already present.
## Adding another scale for y, which will replace the existing scale.

p2
```




```
p0 <- 6
slab_scale <- sd(datastd_G3por$G3por)/sqrt(p0)*sqrt(0.3)
#
ppR2<-numeric()
for (i in 1:4000) {
  sigma2 <- 0.7*rexp(1,rate=1/sd(datastd_G3por$G3por))^2;
  global_scale <- p0 / (p - p0) * sqrt(sigma2) / sqrt(n)
  z <- rnorm(p)
  lambda <- rcauchy(p)
  tau <- rcauchy(1, scale = global_scale)
  caux <- 1/rgamma(1, shape=0.5, rate=0.5)
  c <-  slab_scale * sqrt(caux)
  lambda_tilde <- sqrt(c^2 * lambda^2 / (c^2 + tau^2*lambda^2))
  beta <- rnorm(p) * lambda_tilde * tau
  muvar <- var(as.matrix(datastd_G3por[,2:27]) %*% beta)
  ppR2[i] <- muvar/(muvar+sigma2)
}
ggplot()+geom_histogram(aes(x=ppR2), breaks=seq(0,1,length.out=50)) +
  xlim(c(0,1)) +
```

```
  scale_y_continuous(breaks=NULL) +
  labs(x="Prior predictive Bayesian R^2",y="")
```




```
p0 <- 6
slab_scale <- sd(datastd_G3por$G3por)/sqrt(p0)*sqrt(0.3)
# global scale without sigma, as the scaling by sigma happens in stan_glm
global_scale <- p0 / (p - p0) / sqrt(n)
fit3 <- stan_glm(G3por ~ ., data = datastd_G3por, seed = SEED,
                 prior=hs(global_scale=global_scale, slab_scale=slab_scale),
                 refresh=0)

round(median(loo_R2(fit3)), 2)

## [1] 0.27

round(median(bayes_R2(fit3)), 2)

## [1] 0.32

(loo3 <- loo(fit3))

##
## Computed from 4000 by 377 log-likelihood matrix.
##
##          Estimate   SE
## elpd_loo   -834.3 16.4
## p_loo        23.4  2.4
## looic      1668.6 32.9
```

```
## ------
## MCSE of elpd_loo is 0.1.
## MCSE and ESS estimates assume MCMC draws (r_eff in [0.5, 1.2]).
##
## All Pareto k estimates are good (k < 0.7).
## See help('pareto-k-diagnostic') for details.
```

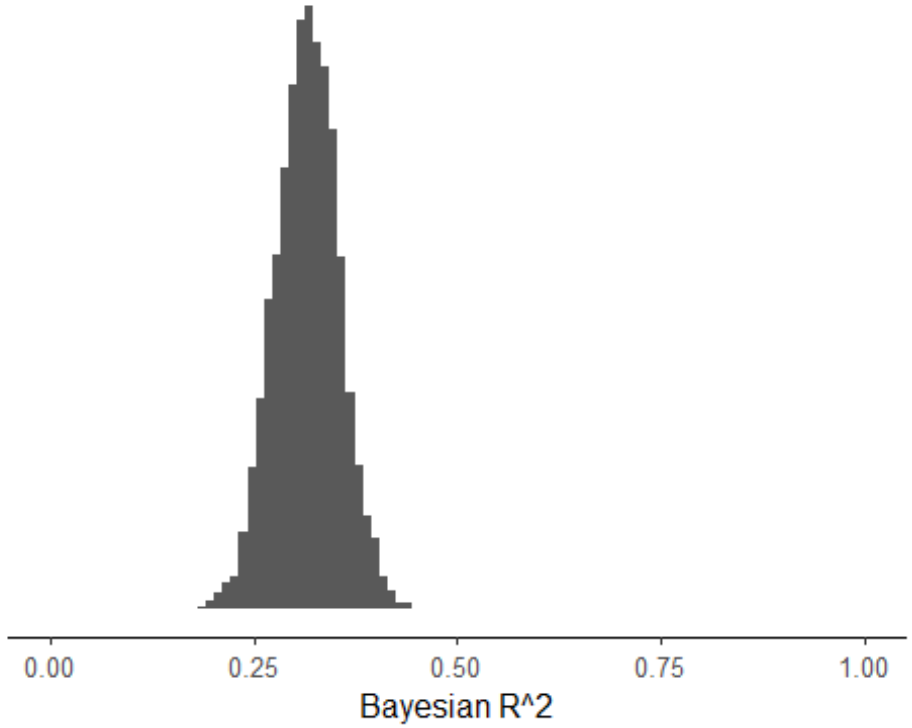
```
loo_compare(loo1,loo3)
```

```
##      elpd_diff se_diff
## fit3  0.0       0.0
## fit1 -2.2       2.7
```

```
loo_compare(loo2,loo3)
```

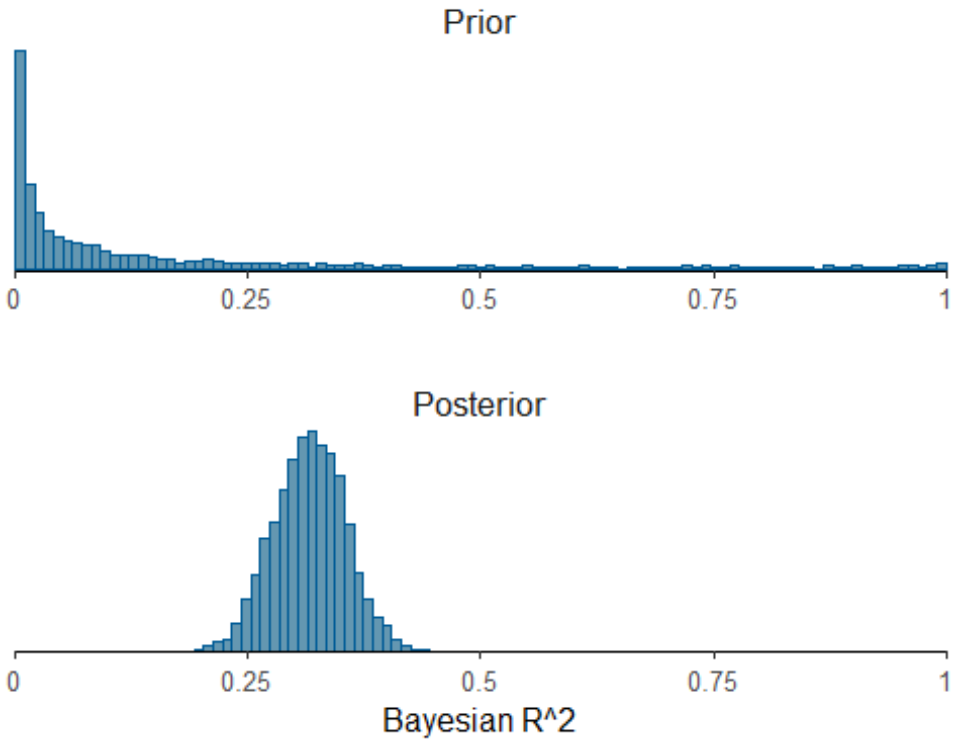
```
##      elpd_diff se_diff
## fit2  0.0       0.0
## fit3 -1.0       1.4
```

```
ggplot()+geom_histogram(aes(x=bayes_R2(fit3)),
breaks=seq(0,1,length.out=100)) +
  xlim(c(0,1)) +
  scale_y_continuous(breaks=NULL) +
  labs(x="Bayesian R^2",y="")
```

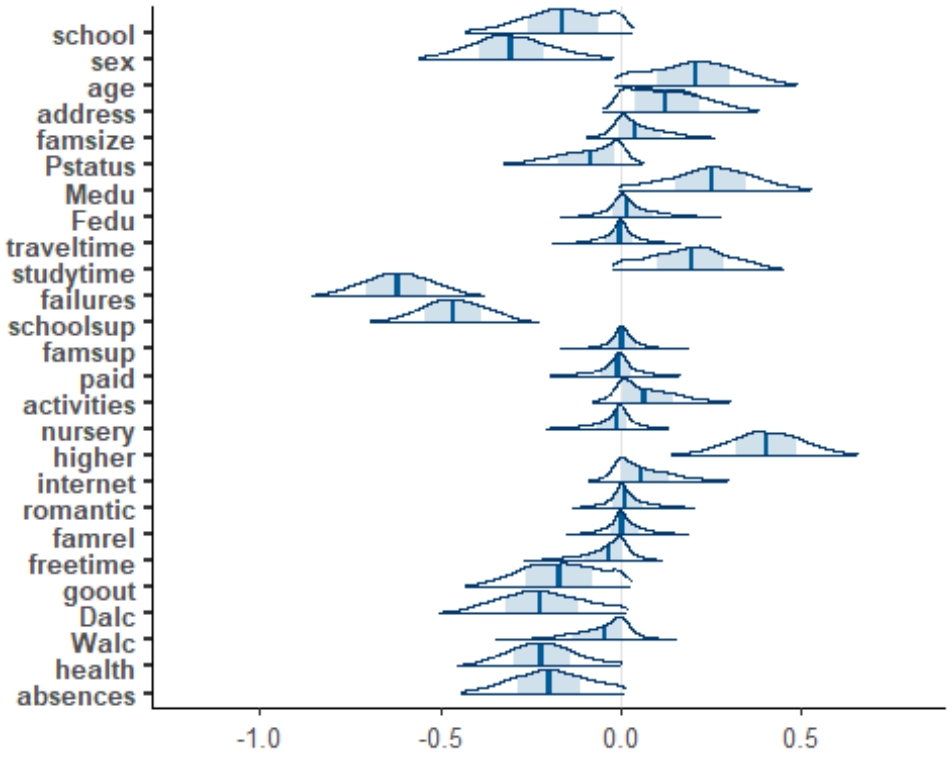



```
pp3 <- mcmc_hist(data.frame(Prior=ppR2,Posterior=bayes_R2(fit3)),
                 breaks=seq(0,1,length.out=100),
                 facet_args = list(nrow = 2)) +
```

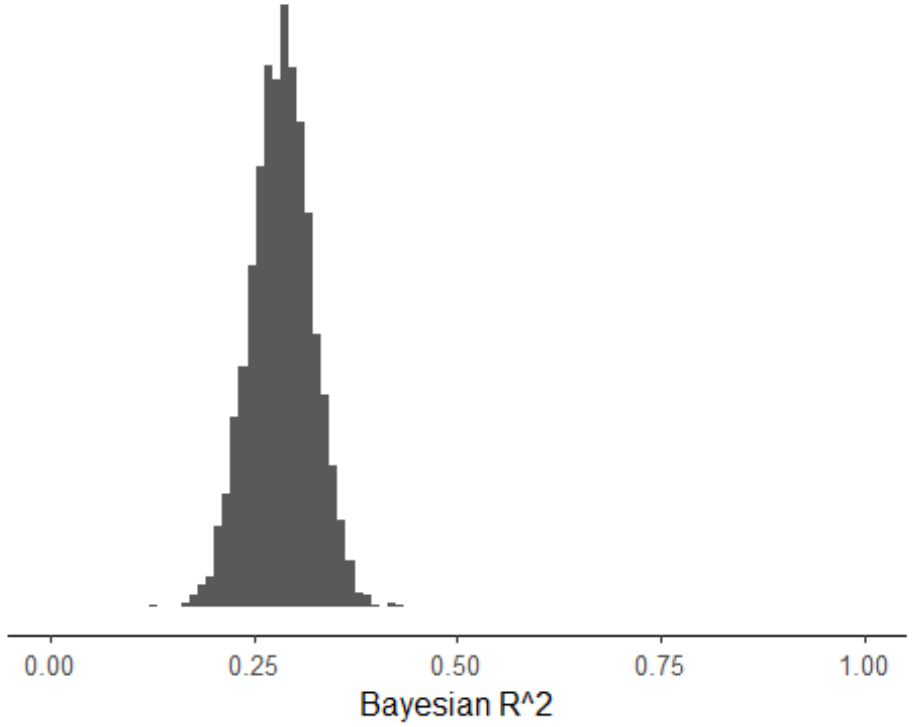
```
  facet_text(size = 13) +
  scale_x_continuous(limits = c(0,1), expand = c(0, 0),
                     labels = c("0","0.25","0.5","0.75","1")) +
  theme(axis.line.y = element_blank()) +
  xlab("Bayesian R^2")
pp3
```




```
p3 <- mcmc_areas(as.matrix(fit3), pars=vars(-'(Intercept)',-sigma),
                 prob_outer=0.95, area_method = "scaled height") +
  xlim(c(-1.2,0.8))

## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.

p3 <- p3 + scale_y_discrete(limits = rev(levels(p3$data$parameter)))

## Scale for y is already present.
## Adding another scale for y, which will replace the existing scale.

p3
```




```
fit4 <- stan_glm(G3por ~ failures + schoolsup + goout + absences +higher +
school +sex,
                 data = datastd_G3por, seed = SEED, refresh=0)

round(median(loo_R2(fit4)), 2)

## [1] 0.25

round(median(bayes_R2(fit4)), 2)

## [1] 0.29

ggplot()+geom_histogram(aes(x=bayes_R2(fit4)),
breaks=seq(0,1,length.out=100)) +
  xlim(c(0,1)) +
  scale_y_continuous(breaks=NULL) +
  labs(x="Bayesian R^2",y="")
```




```
(loo4 <- loo(fit4))
```

```
##
## Computed from 4000 by 377 log-likelihood matrix.
##
##          Estimate   SE
## elpd_loo   -839.5 16.2
## p_loo         9.1  1.3
## looic      1678.9 32.4
## ------
## MCSE of elpd_loo is 0.1.
## MCSE and ESS estimates assume MCMC draws (r_eff in [0.5, 2.1]).
##
## All Pareto k estimates are good (k < 0.7).
## See help('pareto-k-diagnostic') for details.
```

```
loo_compare(loo3,loo4)
```

```
##      elpd_diff se_diff
## fit3  0.0       0.0
## fit4 -5.2       6.0
```

```
loo_compare(loo2,loo4)
```

```
##      elpd_diff se_diff
## fit2  0.0       0.0
## fit4 -6.2       6.8
```

```r
loo_compare(loo1,loo4)
```

```
##      elpd_diff se_diff
## fit1  0.0       0.0
## fit4 -3.0       7.3
```

```r
p4 <- mcmc_areas(as.matrix(fit4), pars=vars(-'(Intercept)',-sigma),
                 prob_outer=0.99, area_method = "scaled height") +
  xlim(c(-1.3,0.1))
```

```
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
```

```r
p4 <- p4 + scale_y_discrete(limits = rev(levels(p4$data$parameter)))
```

```
## Scale for y is already present.
## Adding another scale for y, which will replace the existing scale.
```

```r
p4
```

```
## Warning: Removed 1 row containing missing values or values outside the
scale range
## (`geom_segment()`).
```