MA678 Homework 2 Form Yuxi Wang

9/10/2020

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.

```
# Loading the data and tidy it
library("tidyverse")
```

```
## ── Attaching packages ──────────────────────────────────────────────────────
──────────── tidyverse 1.3.0 ──
```

```
## ✓ tibble  3.0.3     ✓ dplyr   1.0.2
## ✓ tidyr   1.1.2     ✓ stringr 1.4.0
## ✓ readr   1.3.1     ✓ forcats 0.5.0
## ✓ purrr   0.3.4
```

```
## ── Conflicts ─────────────────────────────────────────────────────────────────
──────── tidyverse_conflicts() ──
## x tidyr::expand() masks Matrix::expand()
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x tidyr::pack()   masks Matrix::pack()
## x dplyr::select() masks MASS::select()
## x tidyr::unpack() masks Matrix::unpack()
```

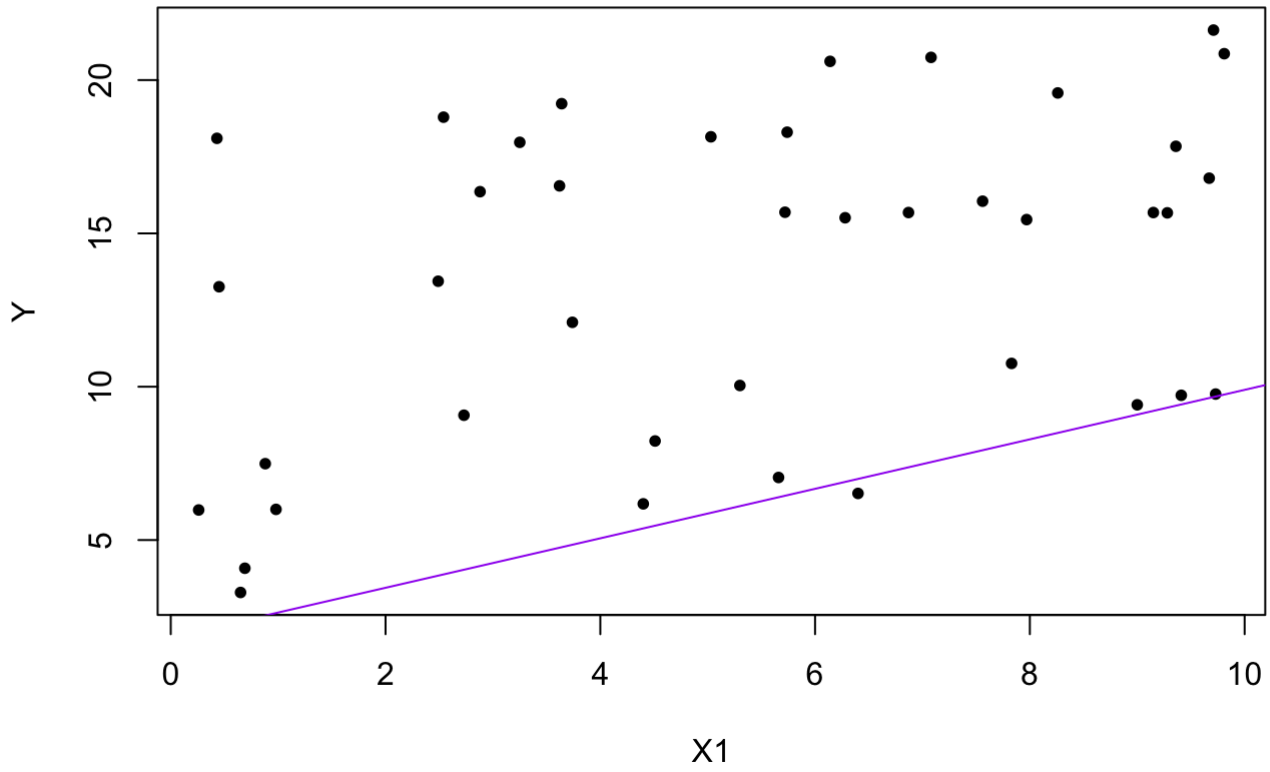
```
data <- read.table(file = '/Users/mac/Desktop/BU Mssp/MA678/ROS-Examples-master/Pyth/
pyth.txt',sep = '\t',header = TRUE)
tidydata <- data %>%
  separate(y.x1.x2, into = c("y", "x1", "x2"), sep = " ",convert = TRUE )
y <-  tidydata[1:40,1]
x1 <- tidydata[1:40, 2]
x2 <- tidydata[1:40, 3]
pyth <- data.frame(x1,x2,y)
# Fitting the model
fit <- stan_glm(y ~x1+x2, data=pyth,refresh=0)
summary(fit)
```

```
##
## Model Info:
##  function:     stan_glm
##  family:       gaussian [identity]
##  formula:      y ~ x1 + x2
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 40
##  predictors:   3
##
## Estimates:
##               mean   sd   10%   50%   90%
## (Intercept) 1.3    0.4  0.8   1.3   1.8
## x1          0.5    0.0  0.5   0.5   0.6
## x2          0.8    0.0  0.8   0.8   0.8
## sigma       0.9    0.1  0.8   0.9   1.1
##
## Fit Diagnostics:
##            mean   sd   10%   50%   90%
## mean_PPD 13.6    0.2 13.3  13.6  13.9
##
## The mean_ppd is the sample average posterior predictive distribution of the outcom
e variable (for details see help('summary.stanreg')).
##
## MCMC diagnostics
##               mcse Rhat n_eff
## (Intercept)   0.0  1.0  4983
## x1            0.0  1.0  4618
## x2            0.0  1.0  4444
## sigma         0.0  1.0  3984
## mean_PPD      0.0  1.0  4475
## log-posterior 0.0  1.0  1602
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure o
f effective sample size, and Rhat is the potential scale reduction factor on split ch
ains (at convergence Rhat=1).
```

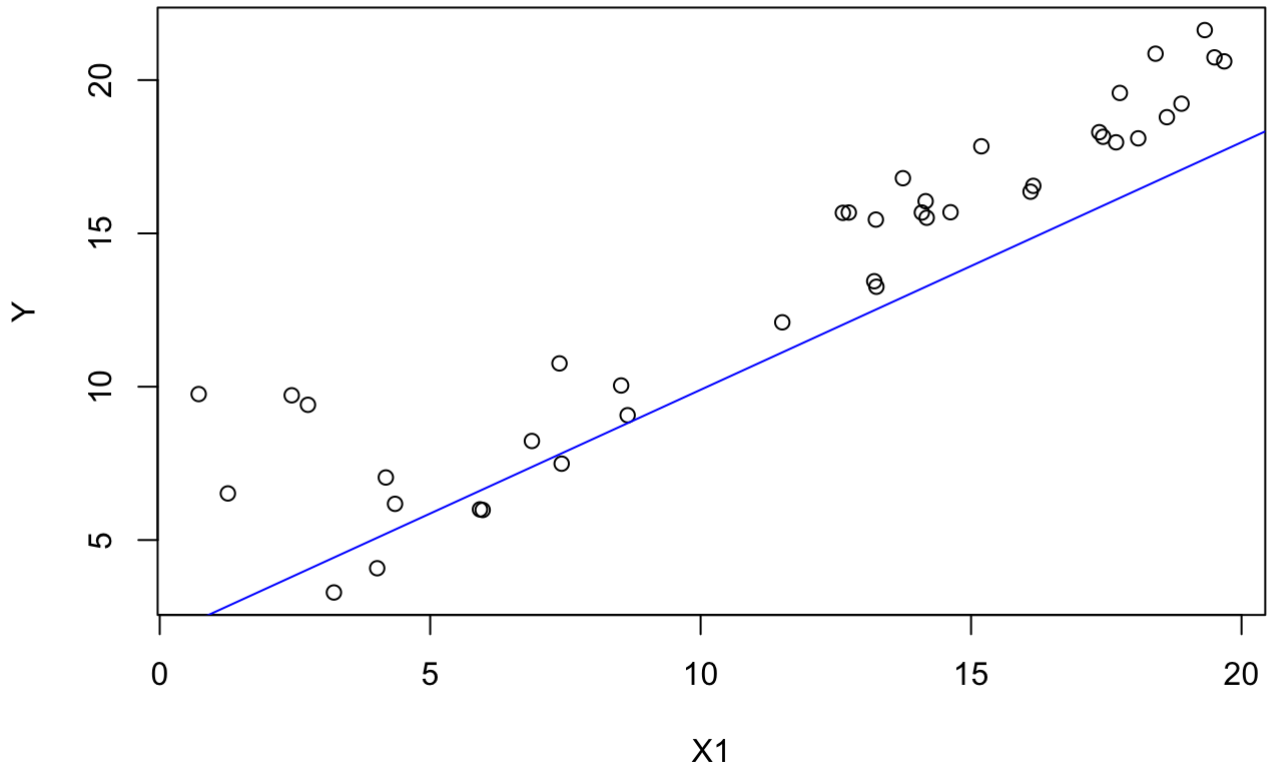
(b)

Display the estimated model graphically as in Figure 10.2

```
# Displaying using one plot for each input variable
plot(pyth$x1, pyth$y, xlab="X1", ylab="Y", pch=20)
b_hat <- coef(fit)
abline(b_hat[1] + b_hat[2], b_hat[3], col="purple")
```



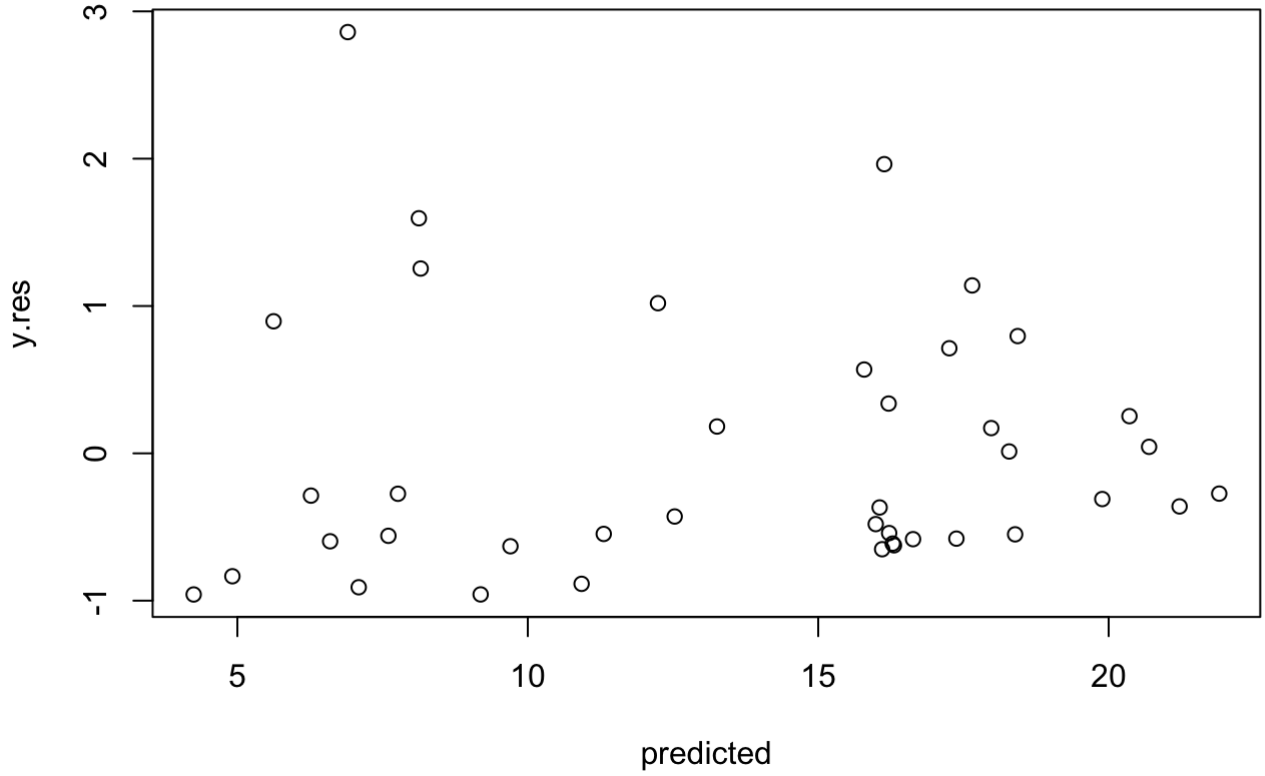

```
plot(pyth$x2, pyth$y, xlab="X1", ylab="Y", pch=1)
b2_hat <- coef(fit)
abline(b2_hat[1] + b2_hat[2], b2_hat[3], col="blue")
```



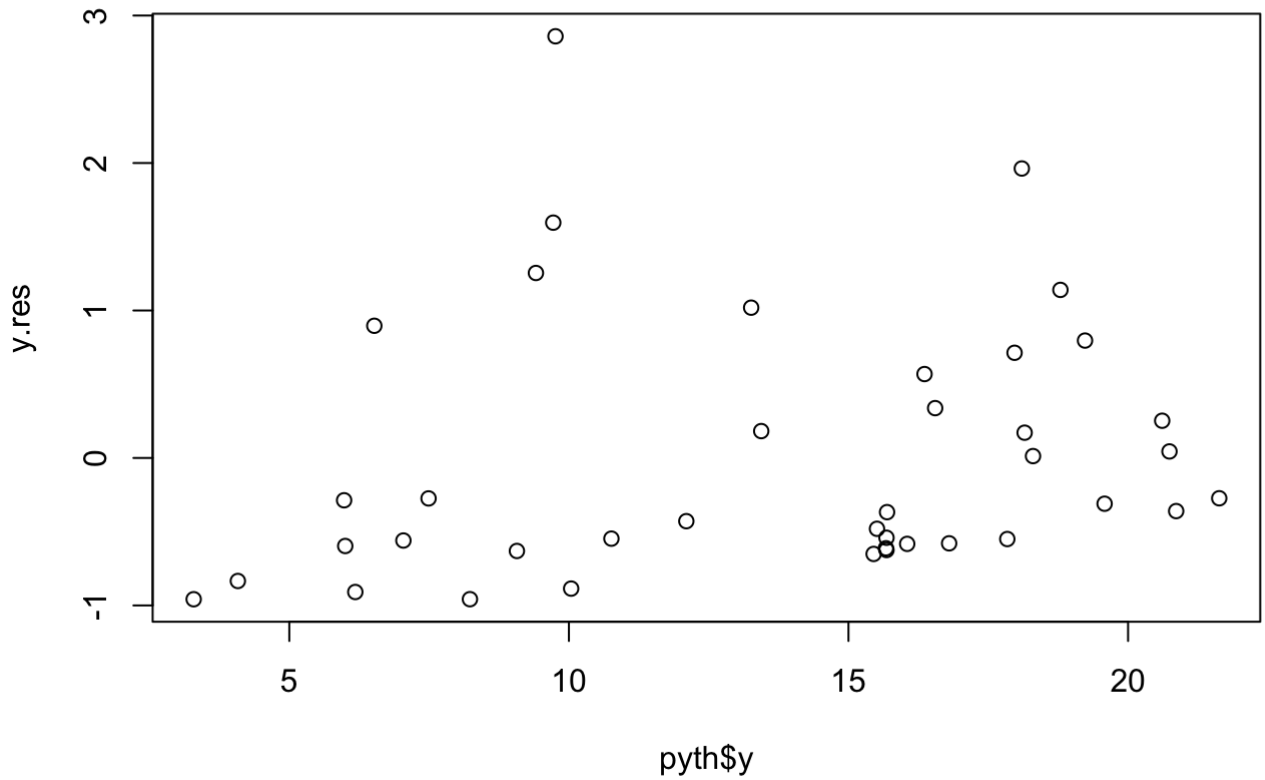
(c)

Make a residual plot for this model. Do the assumptions appear to be met?

```
# To make a residual plot
y.res <- resid(fit)
predicted <- predict(fit)
plot(y.res~predicted)
```




```
plot(y.res~pyth$y)
```



```
# We know that for the assumption of regression analysis, the error should have Indep
endence, Equal variance, Normality.
# In both of these two residual plot, most of the point are in -1~1, and do not have
 trend.
# So that we can briefly believe that 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?

```
# Making predictions
x11 <- tidydata[41:60, 2]
x12 <- tidydata[41:60, 3]
y_predictions <- coef(fit)[1] + coef(fit)[2]*x11 + coef(fit)[3]*x12
c(y_predictions)
```

```
##  [1] 14.810107 19.141662  5.912742 10.532116 19.011481 13.399002  4.827893
##  [8]  9.142383  5.889750 12.341152 18.910645 16.065223  8.960970 14.974800
## [15]  5.859136  7.375219  4.534453 15.132829  9.101424 16.083027
```

```
# The confident of these predictions depend on the fit diagnostics, because we use th
e stan_glm function to fit the model, the mean_PPD of it is 13.6, and the sd of it is
0.2. So that we can believe the prediction is great.
```

12.5

Logarithmic transformation and regression: Consider the following regression: log(weight)=-3.8+2.1log(height)+error, with errors that have standard deviation 0.25. Weights are in pounds and heights are in inches.

(a)

Fill in the blanks: Approximately 68% of the people will have weights within a factor of **-1.28** and **1.28** of their predicted values from the regression.

(b)

Using pen and paper, sketch the regression line and scatterplot of log(weight) versus log(height) that make sense and are consistent with the fitt2ed model. Be sure to label the axes of your graph.

12.6

Logarithmic transformations: The folder Pollution contains mortality rates and various environmental factors from 60 US metropolitan areas. For this exercise we shall model mortality rate given nitric oxides, sulfur dioxide, and hydrocarbons as inputs. this model is an extreme oversimplication, 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 transformation in regression.

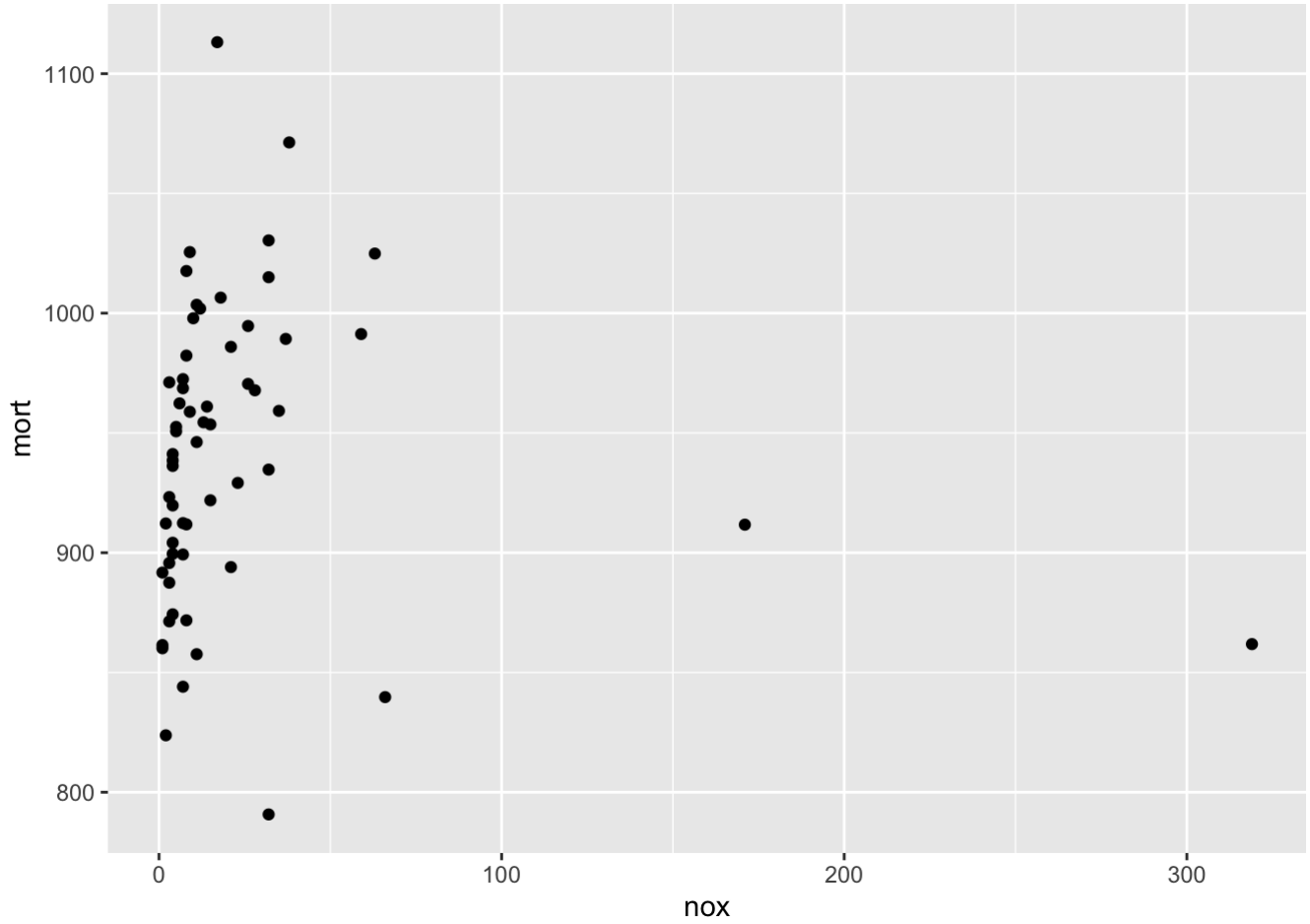
(a)

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.

```
#Loading the data
pollution <- read.csv(file = "/Users/mac/Desktop/BU Mssp/MA678/ROS-Examples-master/Po
llution/data/pollution.csv", head=TRUE)
summary(pollution)
```

```
##       prec            jant            jult           ovr65
##  Min.   :10.00   Min.   :12.00   Min.   :63.00   Min.   : 5.600
##  1st Qu.:32.75   1st Qu.:27.00   1st Qu.:72.00   1st Qu.: 7.675
##  Median :38.00   Median :31.50   Median :74.00   Median : 9.000
##  Mean   :37.37   Mean   :33.98   Mean   :74.58   Mean   : 8.798
##  3rd Qu.:43.25   3rd Qu.:40.00   3rd Qu.:77.25   3rd Qu.: 9.700
##  Max.   :60.00   Max.   :67.00   Max.   :85.00   Max.   :11.800
##       popn            educ            hous            dens           nonw
##  Min.   :2.920   Min.   : 9.00   Min.   :66.80   Min.   :1441   Min.   : 0.80
##  1st Qu.:3.210   1st Qu.:10.40   1st Qu.:78.38   1st Qu.:3104   1st Qu.: 4.95
##  Median :3.265   Median :11.05   Median :81.15   Median :3567   Median :10.40
##  Mean   :3.263   Mean   :10.97   Mean   :80.91   Mean   :3876   Mean   :11.87
##  3rd Qu.:3.360   3rd Qu.:11.50   3rd Qu.:83.60   3rd Qu.:4520   3rd Qu.:15.65
##  Max.   :3.530   Max.   :12.30   Max.   :90.70   Max.   :9699   Max.   :38.50
##      wwdrk            poor             hc              nox
##  Min.   :33.80   Min.   : 9.40   Min.   :  1.00   Min.   :  1.00
##  1st Qu.:43.25   1st Qu.:12.00   1st Qu.:  7.00   1st Qu.:  4.00
##  Median :45.50   Median :13.20   Median : 14.50   Median :  9.00
##  Mean   :46.08   Mean   :14.37   Mean   : 37.85   Mean   : 22.65
##  3rd Qu.:49.52   3rd Qu.:15.15   3rd Qu.: 30.25   3rd Qu.: 23.75
##  Max.   :59.70   Max.   :26.40   Max.   :648.00   Max.   :319.00
##       so2             humid            mort
##  Min.   :  1.00   Min.   :38.00   Min.   : 790.7
##  1st Qu.: 11.00   1st Qu.:55.00   1st Qu.: 898.4
##  Median : 30.00   Median :57.00   Median : 943.7
##  Mean   : 53.77   Mean   :57.67   Mean   : 940.4
##  3rd Qu.: 69.00   3rd Qu.:60.00   3rd Qu.: 983.2
##  Max.   :278.00   Max.   :73.00   Max.   :1113.2
```

```
# Creating a scatterplot of mortality rate versus level of nitric oxides
ggplot(data=pollution, mapping= aes(x=nox, y=mort))+
  geom_point()
```



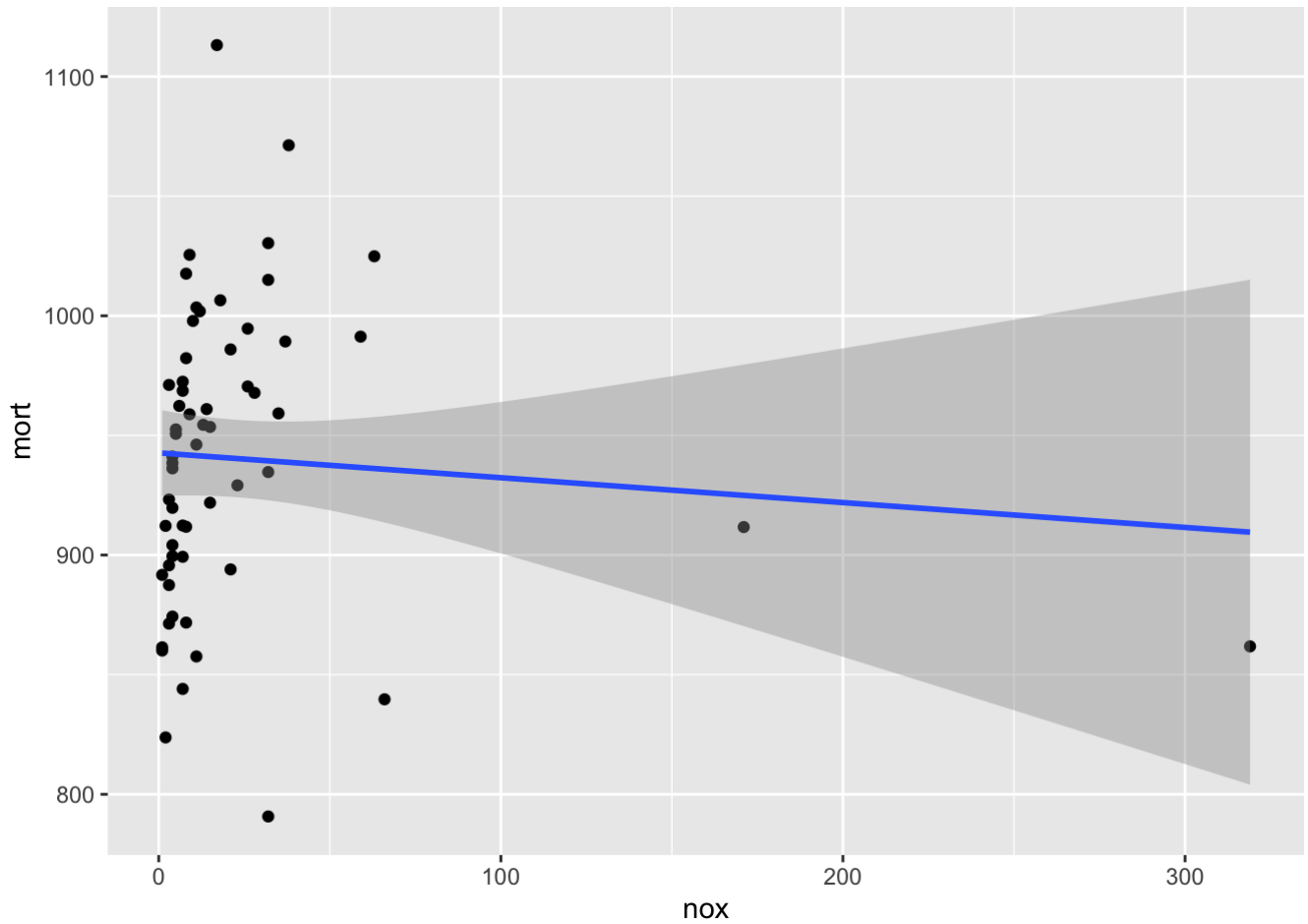

```
#linear regression will fit these data well?
print("I donot think the linear regression will fit these data well. Unless we elimin
ate all points far away from the main data group. So, I think it is better to transfo
rm all data")
```

```
## [1] "I donot think the linear regression will fit these data well. Unless we elimi
nate all points far away from the main data group. So, I think it is better to transf
orm all data"
```

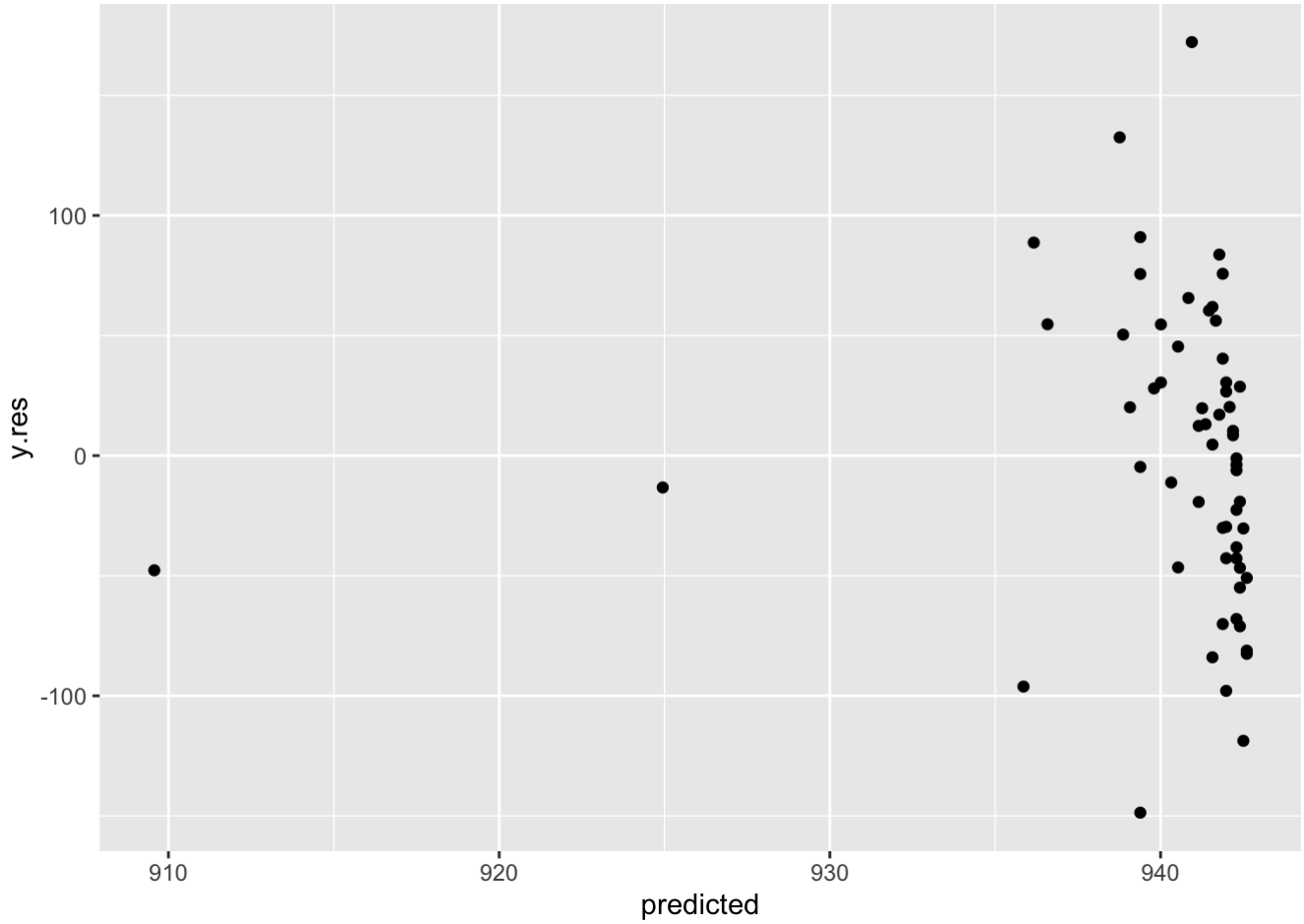
```
## Fitting the model and make the plot of fitted model
fit <- lm(mort ~ nox, data=pollution)
summary(fit)
```

```
##
## Call:
## lm(formula = mort ~ nox, data = pollution)
##
## 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
```

```
ggplot(data=pollution, mapping= aes(x=nox, y=mort)) +
  geom_point() +
  stat_smooth(method="lm", formula=y ~ x, se=TRUE)
```



```
# The residual plot of the model
y.res <- resid(fit)
predicted <- predict(fit)
data_a <- data.frame(y.res,predicted)
ggplot(data=data_a, mapping= aes(x=predicted, y=y.res)) +
  geom_point()
```



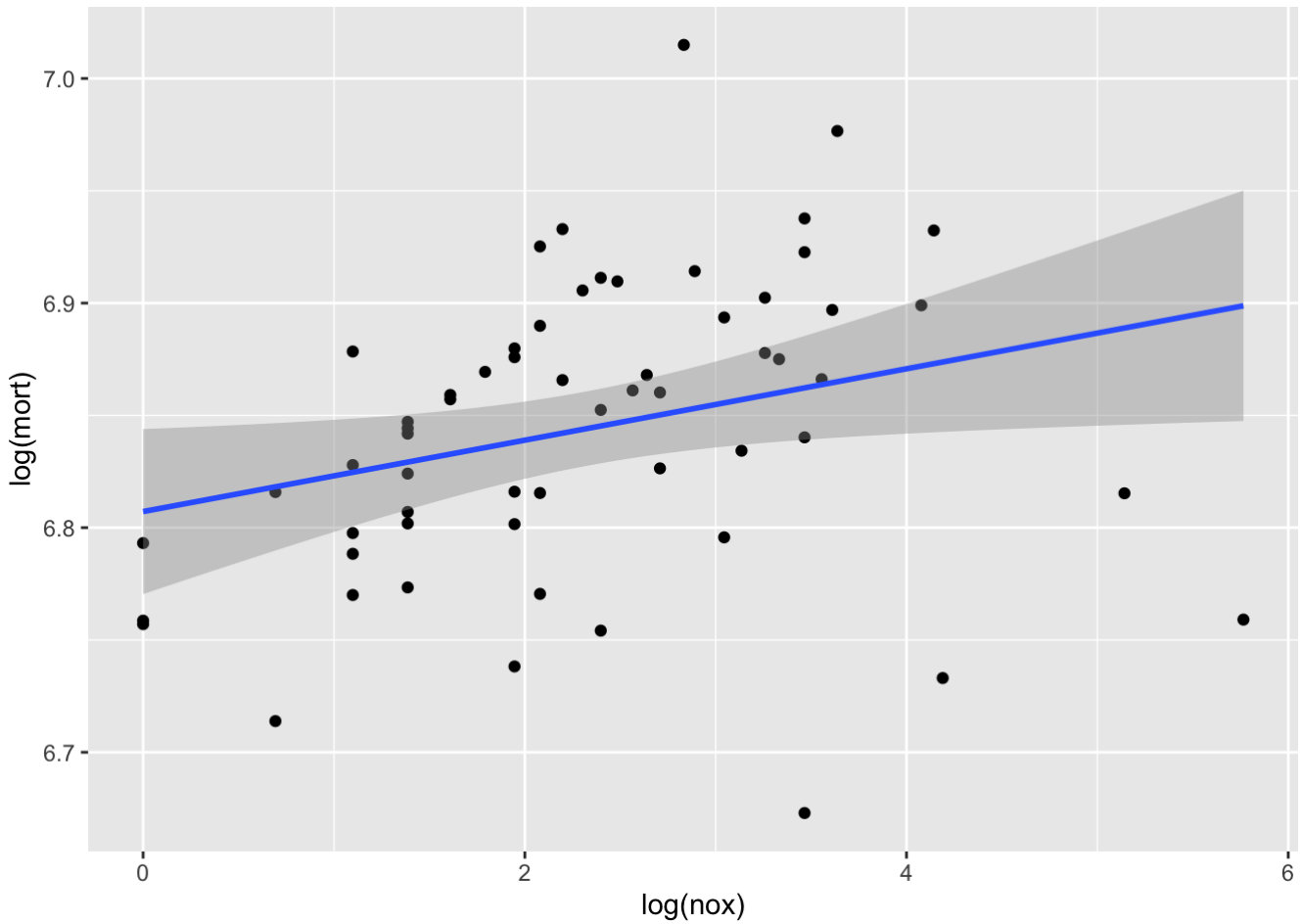
(b)

Find an appropriate reansformation that will result in data more appropriate for linear regression. Fit a regression to the transformed data and evaluate the new residual plot.

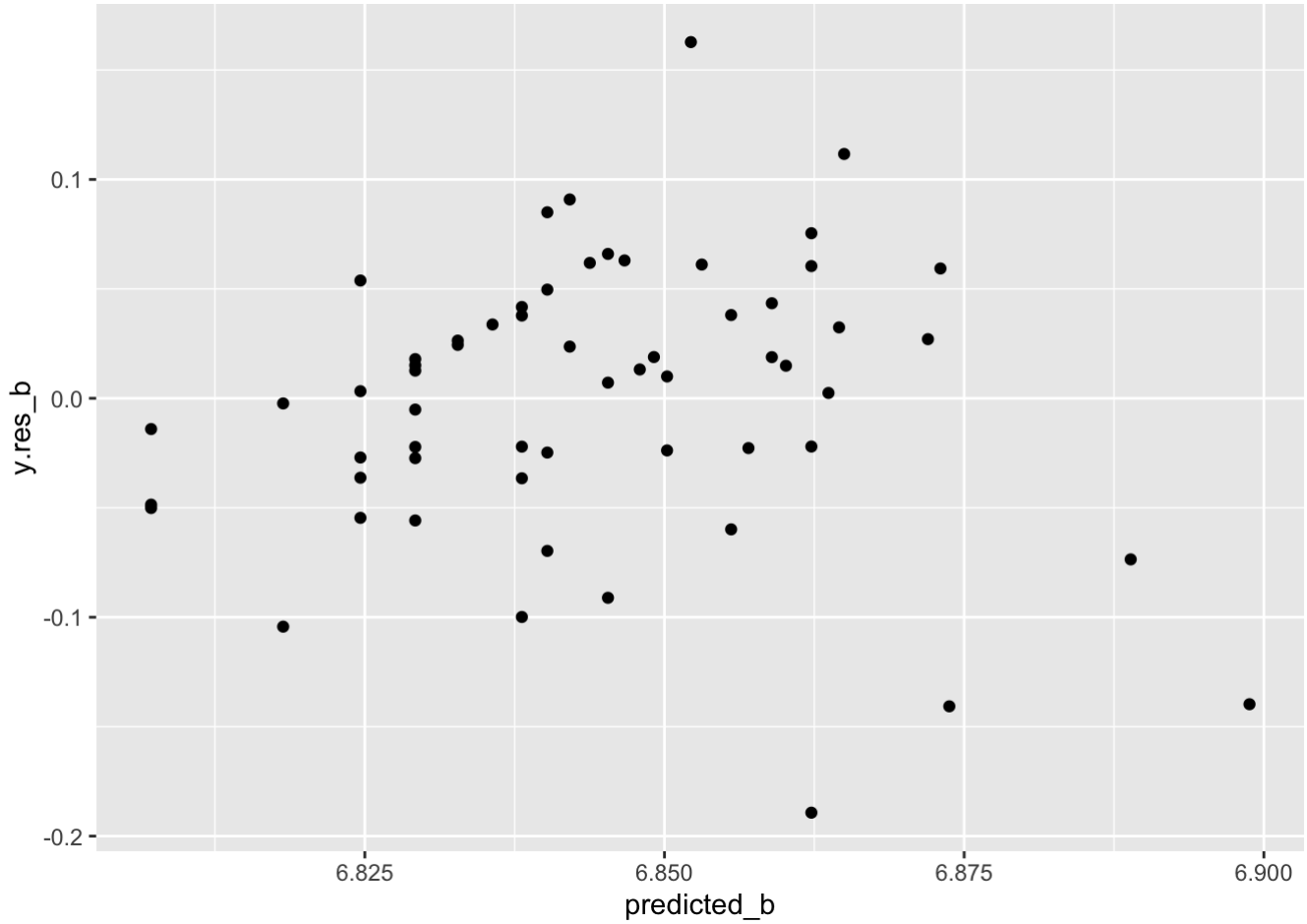
```
#First, try to use logarithmic method.
fit_b <- lm(log(mort) ~ log(nox), data=pollution)
summary(fit_b)
```

```
##
## Call:
## lm(formula = log(mort) ~ log(nox), data = pollution)
##
## 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
```

```
#By using some plots to see whether the logarithmic method is well worked.
ggplot(data=pollution, aes(x=log(nox), y=log(mort))) +
  geom_point() +
  stat_smooth(method="lm", formula=y ~ x, se=TRUE)
```



```
y.res_b <- resid(fit_b)
predicted_b <- predict(fit_b)
data_b <- data.frame(y.res_b ,predicted_b)
ggplot(data=data_b, mapping= aes(x=predicted_b, y=y.res_b)) +
  geom_point()
```




```
print("In this plot, the result of the fitted plot and the residual plot are better t
han before.")
```

```
## [1] "In this plot, the result of the fitted plot and the residual plot are better
than before."
```

(c)

Interpret the slope coefficient from the model you chose in (b)

#Answer: The slope coefficient in this model is 0.015893. Which means that if the variance log(nox) have 1% difference in nitric oxides, the difference in mort will get bigger in 0.0159.

(d)

Now fit a model predicting mortality rate using levels of nitric oxides, sulfur dioxide, and hydrocarbons as inputs. Use appropriate transformation when helpful. Plot the fitted regression model and interpret the coefficients.

```
# Firstly, checking IQR for each predictors we have to use in the new model
apply(pollution[, c("hc", "nox", "so2")], FUN=IQR, MARGIN = 2)
```

```
##    hc   nox   so2
## 23.25 19.75 58.00
```

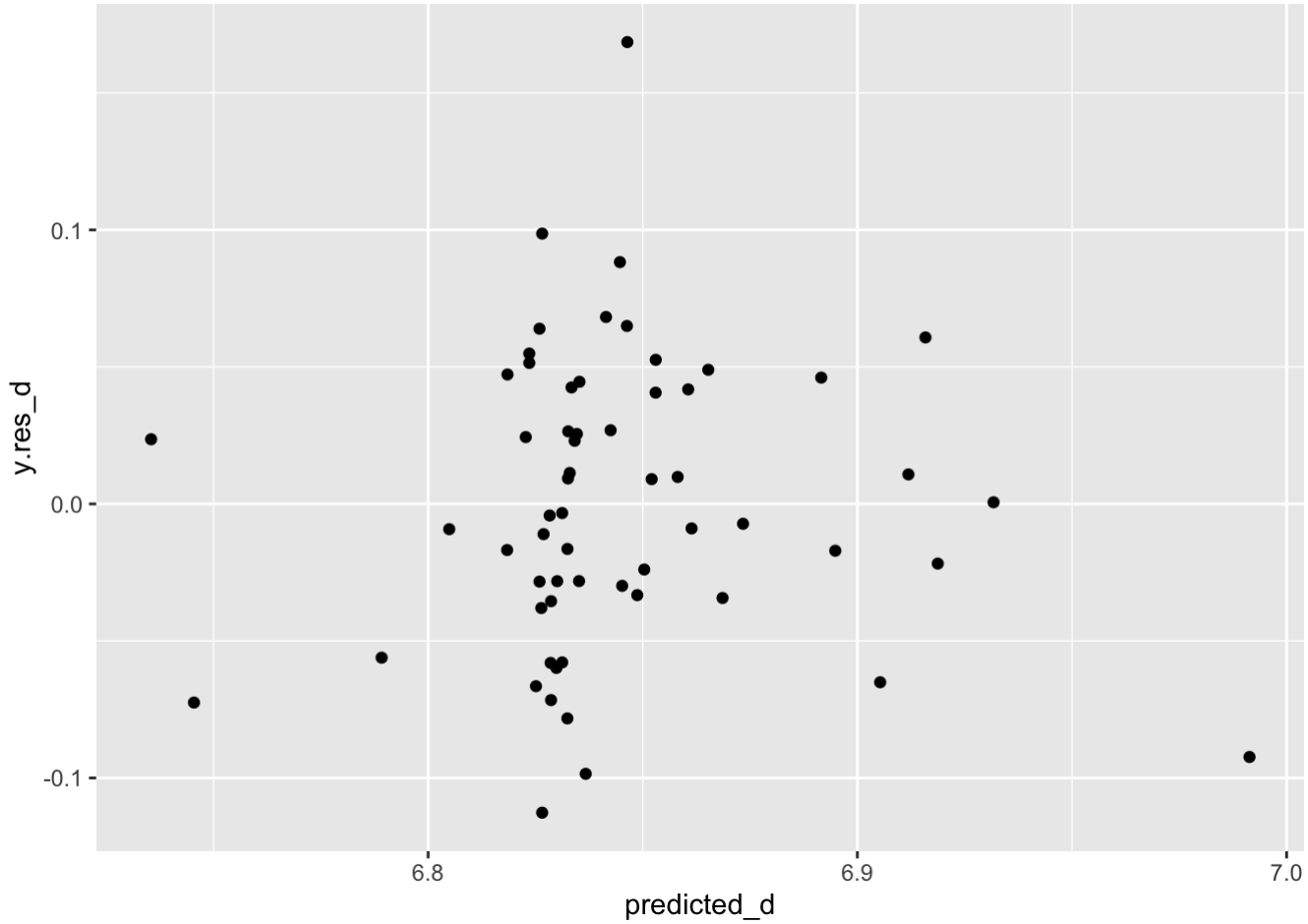
```
# Then, we can scale predictors
scale <- function(X) (X - mean(X)) / (2*sd(X))
pollution[, c("hc_1", "nox_1", "so2_1")] <- apply(pollution[, c("hc", "nox", "so2")],
FUN=scale, MARGIN = 2)
apply(pollution[, c("hc_1", "nox_1", "so2_1")], FUN=IQR, MARGIN = 2)
```

```
##      hc_1     nox_1     so2_1
## 0.1263894 0.2131297 0.4574820
```

```
# In this situation, we can use linear model to fit these predictors.
fit_d <- lm(log(mort) ~ hc_1+ nox_1 + so2_1, data=pollution)
summary(fit_d)
```

```
##
## Call:
## lm(formula = log(mort) ~ hc_1 + nox_1 + so2_1, data = pollution)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.112676 -0.033540 -0.003781  0.041982  0.168553
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.844105   0.007098 964.211  < 2e-16 ***
## hc_1        -0.323153   0.118398  -2.729  0.00846 **
## nox_1        0.296217   0.124494   2.379  0.02077 *
## so2_1        0.026428   0.023236   1.137  0.26022
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.05498 on 56 degrees of freedom
## Multiple R-squared:  0.3473, Adjusted R-squared:  0.3123
## F-statistic: 9.931 on 3 and 56 DF,  p-value: 2.39e-05
```

```
# By using some plots to see the result.
y.res_d <- resid(fit_d)
predicted_d <- predict(fit_d)
data_d <- data.frame(y.res_d ,predicted_d)
ggplot(data=data_d, mapping= aes(x=predicted_d, y=y.res_d)) +
  geom_point()
```




```
print("The coefficients:
     Interception: The death rate of people exposed to average levels of nitric oxid
e, sulfur dioxide and hydrocarbons is exp(6.84) = 934.4891
     hc_1:  The standard deviation of hydrocarbons, and the rest are average. The cor
responding mortality rate is reduced by 0.726149 times and 27%
     nox_1: The standard deviation of nitric oxide, the rest are average, and the cor
responding mortality rate is 1.34985 times higher, which is 35% higher
     so2_1: One standard deviation difference of sulfur dioxide corresponds to a 0.0
3% increase in mortality")
```

```
## [1] "The coefficients:\n     Interception: The death rate of people exposed to ave
rage levels of nitric oxide, sulfur dioxide and hydrocarbons is exp(6.84) = 934.4891
\n     hc_1:  The standard deviation of hydrocarbons, and the rest are average. The c
orresponding mortality rate is reduced by 0.726149 times and 27%\n     nox_1: The sta
ndard deviation of nitric oxide, the rest are average, and the corresponding mortalit
y rate is 1.34985 times higher, which is 35% higher\n     so2_1: One standard deviati
on difference of sulfur dioxide corresponds to a 0.03% increase in mortality"
```

(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.

```
# seprete the data to train and test
train <- pollution[1:30, ]
test <- pollution[31:60, ]

# To fit a linear model by using training dataset, and make a prediction
fit_e <- lm(log(mort) ~ nox_1 + so2_1 + hc_1, data=train)
summary(fit_e)
```

```
##
## Call:
## lm(formula = log(mort) ~ nox_1 + so2_1 + hc_1, data = train)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.104750 -0.029486 -0.004945  0.036401  0.088267
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.849793   0.009281 738.045   <2e-16 ***
## nox_1        0.095321   0.214682   0.444    0.661
## so2_1        0.054983   0.032040   1.716    0.098 .
## hc_1        -0.128191   0.202298  -0.634    0.532
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.05038 on 26 degrees of freedom
## Multiple R-squared:  0.3757, Adjusted R-squared:  0.3037
## F-statistic: 5.216 on 3 and 26 DF,  p-value: 0.005925
```

```
prediction <- predict(fit_e, test)
cbind(predictions=exp(prediction), observed=test$mort)
```

```
##    predictions observed
## 31    946.1116 1006.490
## 32    924.1960  861.439
## 33    976.9405  929.150
## 34    932.8144  857.622
## 35    959.0416  961.009
## 36    928.2680  923.234
## 37    928.4681 1113.156
## 38    967.3537  994.648
## 39   1004.3378 1015.023
## 40   1067.3597  991.290
## 41    926.3130  893.991
## 42    931.9622  938.500
## 43    964.6836  946.185
## 44    945.0756 1025.502
## 45    931.3130  874.281
## 46    946.5939  953.560
## 47    903.0423  839.709
## 48    921.5209  911.701
## 49    889.4473  790.733
## 50    926.5700  899.264
## 51    933.4211  904.155
## 52    934.4545  950.672
## 53    934.4234  972.464
## 54    927.8740  912.202
## 55    950.7915  967.803
## 56    924.5027  823.764
## 57    943.2453 1003.502
## 58    926.3294  895.696
## 59    947.1499  911.817
## 60    943.9589  954.442
```

```
# Finally to compute RMSE
sqrt(mean((test$mort-exp(prediction))^2))
```

```
## [1] 58.10359
```

12.7

Cross validation comparison of models with different transformations of outcomes: when we compare models with transformed continuous outcomes, we must take into account how the nonlinear transformation warps the continuous outcomes. Follow the procedure used to compare models for the mesquite bushes example on page 202.

(a)

Compare models for earnings and for log(earnings) given height and sex as shown in page 84 and 192. Use earnk and log(earnk) as outcomes.

```
# Loading the data and reset the model from page 84 and page 192
earnings <- read.csv(file = '/Users/mac/Desktop/BU Mssp/MA678/ROS-Examples-master/Ear
nings/data/earnings.csv',header = TRUE)
earnings$earnk <- earnings$earn/1000
fit_1 <- stan_glm(earnk ~ height + male, data=earnings, refresh=0)
print(fit_1)
```

```
## stan_glm
##  family:       gaussian [identity]
##  formula:      earnk ~ height + male
##  observations: 1816
##  predictors:   3
## ------
##             Median MAD_SD
## (Intercept) -25.9   12.0
## height        0.6    0.2
## male         10.7    1.4
##
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 21.4    0.3
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

```
fit_2 <- stan_glm(log(earnk) ~ height + male, data=earnings, subset=earn>0,refresh=0)
print(fit_2)
```

```
## stan_glm
##  family:       gaussian [identity]
##  formula:      log(earnk) ~ height + male
##  observations: 1629
##  predictors:   3
## ------
##             Median MAD_SD
## (Intercept) 1.1    0.5
## height      0.0    0.0
## male        0.4    0.1
##
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 0.9    0.0
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

```
# Using the Jacobian to adjust the predictive comparison after a transformation
loo_1 <- loo(fit_1)
```

```
## 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.
```

```
kfold_1 <- kfold(fit_1, K=10)
```

```
## Fitting model 1 out of 10
```

```
## Fitting model 2 out of 10
```

```
## Fitting model 3 out of 10
```

```
## Fitting model 4 out of 10
```

```
## Fitting model 5 out of 10
```

```
## Fitting model 6 out of 10
```

```
## Fitting model 7 out of 10
```

```
## Fitting model 8 out of 10
```

```
## Fitting model 9 out of 10
```

```
## Fitting model 10 out of 10
```

```
loo_2_with_jacobian <- loo(fit_2)
earnings_1 <- earnings[which(earnings$earnk>0),]
loo_2_with_jacobian$pointwise[,1] <- loo_2_with_jacobian$pointwise[,1]-
  log(earnings_1$earnk)
sum(loo_2_with_jacobian$pointwise[,1])
```

```
## [1] -6663.319
```

```
library(bayesplot)
```

```
## This is bayesplot version 1.7.2
```

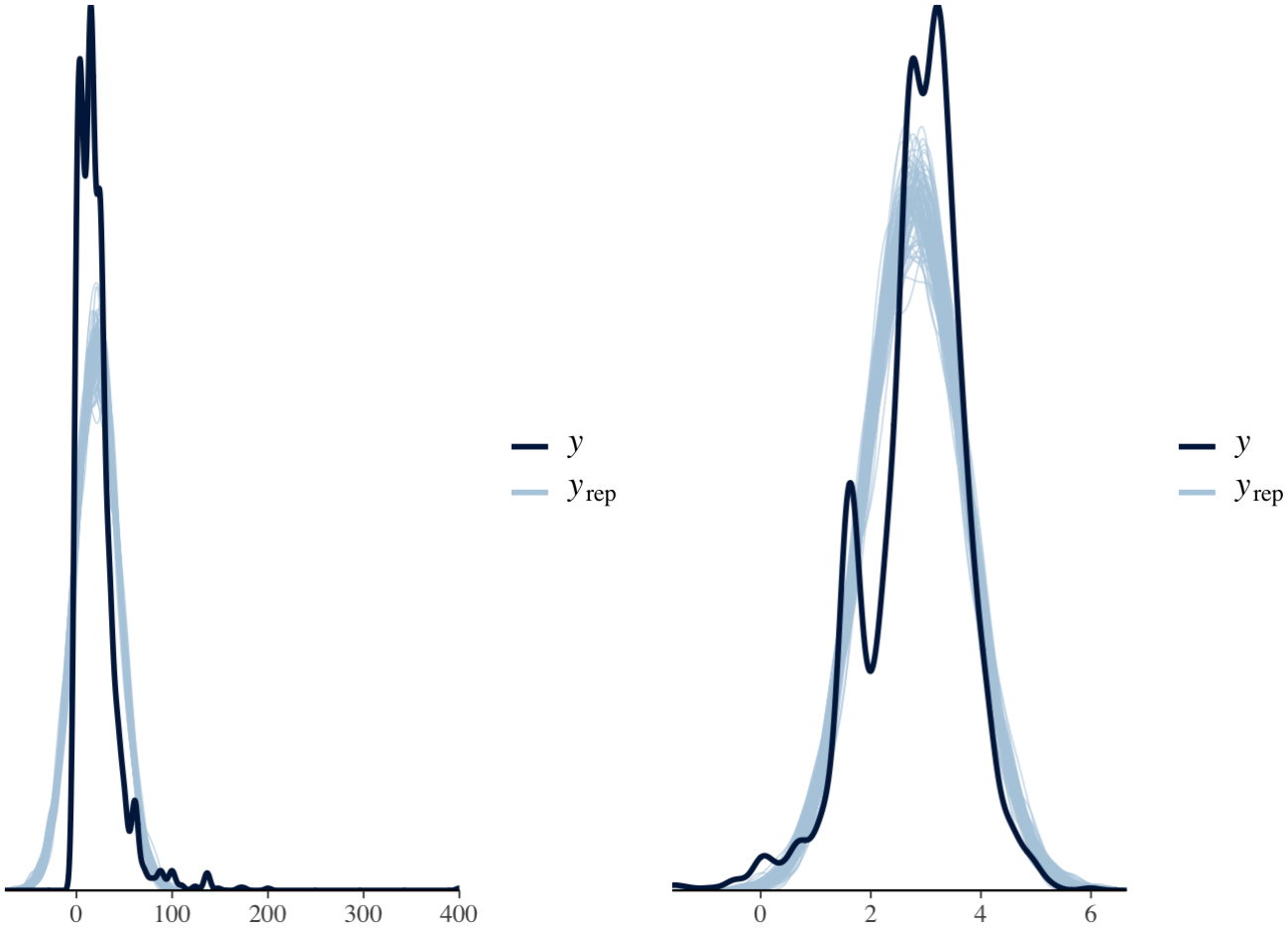
```
## - Online documentation and vignettes at mc-stan.org/bayesplot
```

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

```
##    * Does _not_ affect other ggplot2 plots
```

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

```
p1 <- posterior_predict(fit_1)
n_sims1 <- nrow(p1)
sims_display1 <- sample(n_sims1,100)
p2 <- posterior_predict(fit_2)
n_sims2 <- nrow(p2)
sims_display2 <- sample(n_sims2,100)
ppc_1 <- ppc_dens_overlay(earnings$earnk,p1[sims_display1,])+ theme(axis.line.y = ele
ment_blank())
ppc_2 <- ppc_dens_overlay(log(earnings_1$earnk),p2[sims_display2,])+ theme(axis.line.
y = element_blank())
plot <- bayesplot_grid(
  ppc_1,ppc_2,
  grid_args = list(ncol=2)
)
plot
```



(b)

Compare models from other exercises in this chapter.

12.8

Log-log transformations: Suppose that, for a certain population of animals, we can predict log weight from log height as follows:

- An animal that is 50 centimeters tall is predicted to weigh 10 kg.

- Every increase of 1% in height corresponds to a predicted increase of 2% in weight.
- The weights of approximately 95% of the animals fall within a factor of 1.1 of predicted values.

(a)

Give the equation of the regression line and the residual standard deviation of the regression.

#Answer: The equation of the regression line is: log(weight)= -5.52 + 2log(height). the residual standard deviation of the regression is: log(1.1)/2=0.0477.

(b)

Suppose the standard deviation of log weights is 20% in this population. What, then, is the R^2 of the regression model described here?

#Answer: To calculate it: 1-((log(1.1)/2)^2)/(0.2)2=0.9432248 So, R^{2} of the regression model 0.9432248

12.9

Linear and logarithmic transformations: For a study of congressional elections, you would like a measure of the relative amount of money raised by each of the two major-party candidates in each district. Suppose that you know the amount of money raised by each candidate; label these dollar values Di and Ri. You would like to combine these into a single variable that can be included as an input variable into a model predicting vote share for the Democrats. Discuss the advantages and disadvantages of the following measures:

(a)

The simple difference, $D_i - R_i$

Answer: By trying to use congressional elections data to see the outcome. It is easy to find the advantage of this simple difference that is symmetric and centered at zero.

However, a disadvantage of this simple difference transformation is only numbers of the difference are taken into account, not proportions. The mere difference in the number of votes cannot explain the difference in candidates. Because the total number of voters is often unknown and

changes over time. If only this transformation is used for modeling, the results of the model are often not easy to interpret.

(b)

The ratio, D_i/R_i

Answer: This transformation is just the opposite of the first transformation. It has advantages that is proportions. It can show the gap in percentage. At the same time, its shortcomings are asymmetrical, and also, it is centered on 1, not 0.

(c)

The difference on the logarithmic scale, $log\ D_i - log\ R_i$

Answer: The advantage of this transformation is that it is centered at zero and symmetrical. In addition, because logarithmization is used, the difference is not sensitive to outliers. At the same time, its disadvantage is only numbers of the difference are taken into account, not proportions.

(d)

The relative proportion, $D_i/(D_i + R_i)$.

#Answer: This transformation uses relative proportions. Unlike the second transformation, it has symmetry, which is its advantage. The disadvantage is that it is not centered on 0.

12.11

Elasticity: An economist runs a regression examining the relations between the average price of cigarettes, P, and the quantity purchased, Q, across a large sample of counties in the United States, assuming the functional form, $logQ = \alpha + \beta logP$. Suppose the estimate for β is 0.3. Interpret this coefficient.

#Answer： Take the Euler's Number in both sides of the equation. We can get the equation: #Q = (e^(alpha)) * P^(beta) #In this situation, for each 1% difference in cigarettes price, the predicted difference in quantity purchased is (e^(alpha)) * 0.01^(0.3).

12.13

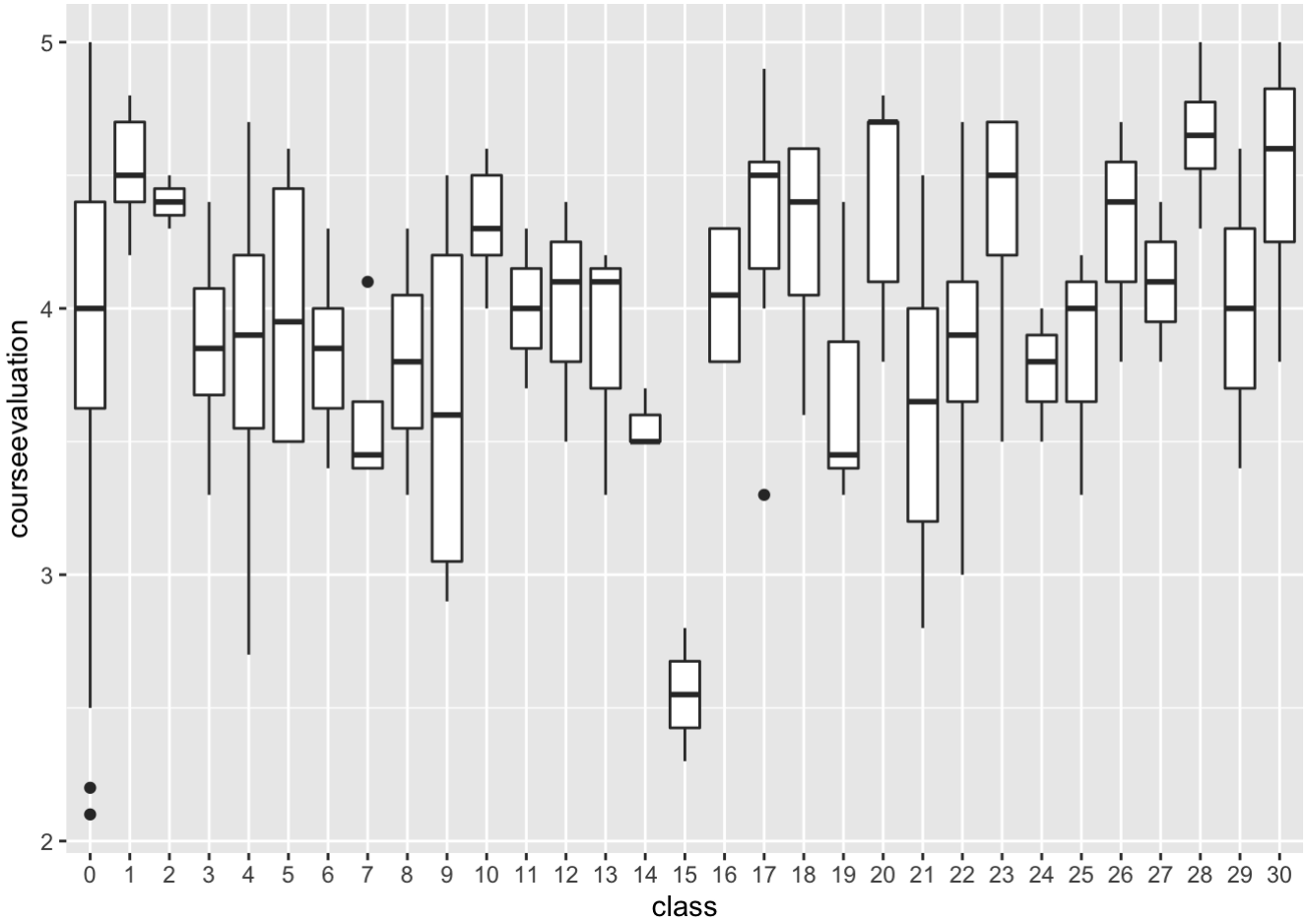
Building regression models: Return to the teaching evaluations data from Exercise 10.6. Fit regression models predicting evaluations given many of the inputs in the dataset. Consider interactions, combinations of predictors, and transformations, as appropriate. Consider several models, discuss in detail the final model that you choose, and also explain why you chose it rather than the others you had considered.

```
prof <- read.csv("/Users/mac/Desktop/BU Mssp/MA678/ROS-Examples-master/Beauty/data/Pr
ofEvaltnsBeautyPublic.csv")
summary(prof)
```

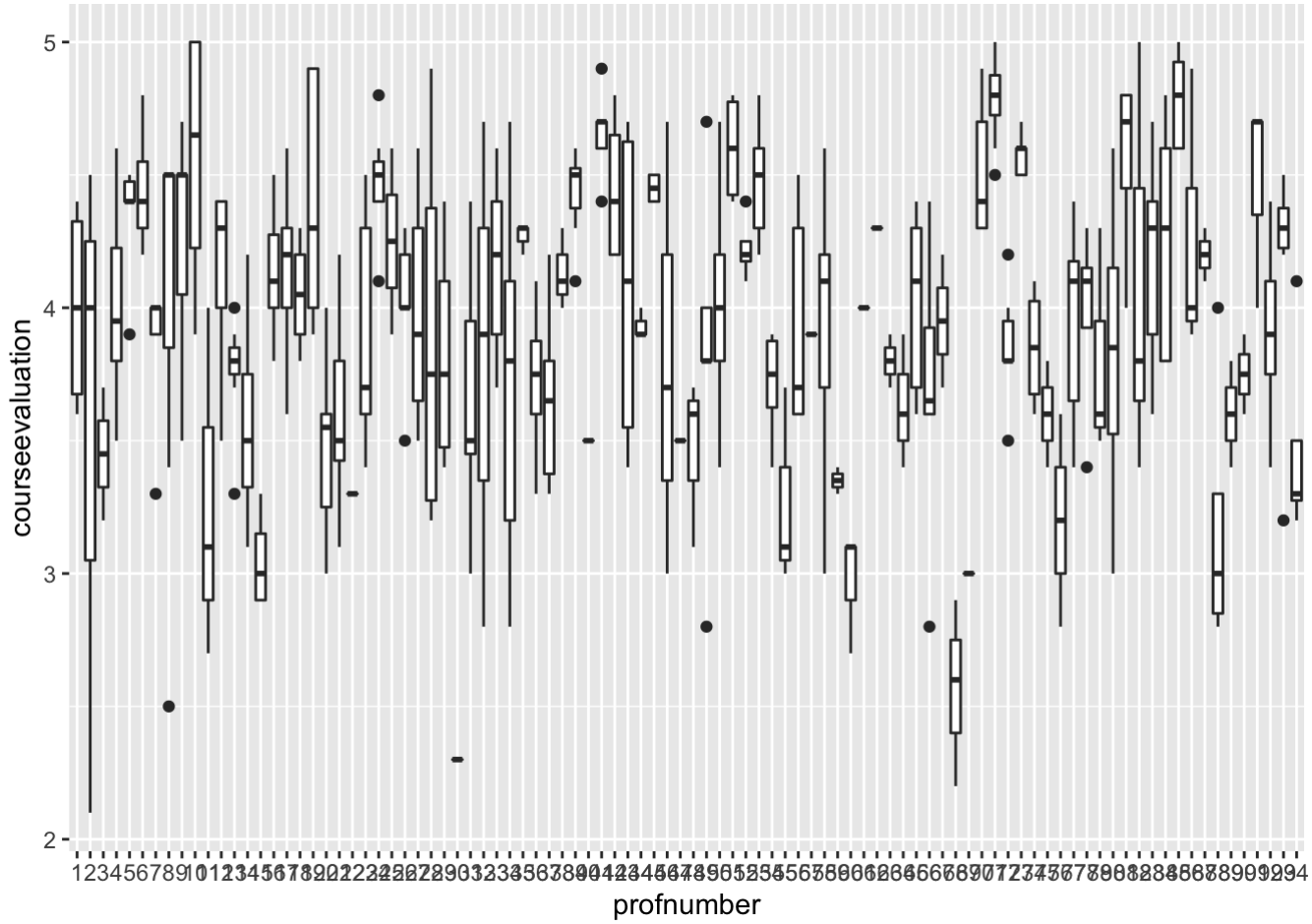
```
##     tenured         profnumber       minority           age
##  Min.   :0.0000   Min.   : 1.00   Min.   :0.0000   Min.   :29.00
##  1st Qu.:0.0000   1st Qu.:20.00   1st Qu.:0.0000   1st Qu.:42.00
##  Median :1.0000   Median :44.00   Median :0.0000   Median :48.00
##  Mean   :0.5464   Mean   :45.43   Mean   :0.1382   Mean   :48.37
##  3rd Qu.:1.0000   3rd Qu.:70.50   3rd Qu.:0.0000   3rd Qu.:57.00
##  Max.   :1.0000   Max.   :94.00   Max.   :1.0000   Max.   :73.00
##  beautyf2upper    beautyflowerdiv beautyfupperdiv beautym2upper
##  Min.   : 1.000   Min.   :1.000   Min.   :1.000   Min.   :1.000
##  1st Qu.: 4.000   1st Qu.:2.000   1st Qu.:4.000   1st Qu.:4.000
##  Median : 5.000   Median :4.000   Median :5.000   Median :5.000
##  Mean   : 5.214   Mean   :3.963   Mean   :5.019   Mean   :4.752
##  3rd Qu.: 6.000   3rd Qu.:5.000   3rd Qu.:7.000   3rd Qu.:6.000
##  Max.   :10.000   Max.   :8.000   Max.   :9.000   Max.   :9.000
##  beautymlowerdiv beautymupperdiv   btystdave          btystdf2u
##  Min.   :1.000   Min.   :1.000   Min.   :-1.53884   Min.   :-2.09653
##  1st Qu.:2.000   1st Qu.:3.000   1st Qu.:-0.74462   1st Qu.:-0.66500
##  Median :3.000   Median :4.000   Median :-0.15636   Median :-0.18782
##  Mean   :3.413   Mean   :4.147   Mean   :-0.08835   Mean   :-0.08579
##  3rd Qu.:5.000   3rd Qu.:5.000   3rd Qu.: 0.45725   3rd Qu.: 0.28935
##  Max.   :7.000   Max.   :9.000   Max.   : 1.88167   Max.   : 2.19806
##     btystdfl          btystdfu          btystdm2u          btystdml
##  Min.   :-1.66803   Min.   :-2.02983   Min.   :-2.34098   Min.   :-1.48761
##  1st Qu.:-1.13652   1st Qu.:-0.57701   1st Qu.:-0.52736   1st Qu.:-0.90006
##  Median :-0.07351   Median :-0.09273   Median : 0.07718   Median :-0.31252
##  Mean   :-0.09302   Mean   :-0.08018   Mean   :-0.07298   Mean   :-0.07015
##  3rd Qu.: 0.45800   3rd Qu.: 0.87581   3rd Qu.: 0.68172   3rd Qu.: 0.86256
##  Max.   : 2.05253   Max.   : 1.84436   Max.   : 2.49533   Max.   : 2.03765
##     btystdmu           class1           class2           class3
##  Min.   :-1.5731   Min.   :0.0000   Min.   :0.00000   Min.   :0.00000
##  1st Qu.:-0.6547   1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:0.00000
##  Median :-0.1954   Median :0.0000   Median :0.00000   Median :0.00000
##  Mean   :-0.1280   Mean   :0.0108   Mean   :0.00432   Mean   :0.01728
##  3rd Qu.: 0.2638   3rd Qu.:0.0000   3rd Qu.:0.00000   3rd Qu.:0.00000
##  Max.   : 2.1007   Max.   :1.0000   Max.   :1.00000   Max.   :1.00000
##      class4            class5            class6            class7
##  Min.   :0.00000   Min.   :0.000000   Min.   :0.00000   Min.   :0.000000
##  1st Qu.:0.00000   1st Qu.:0.000000   1st Qu.:0.00000   1st Qu.:0.000000
##  Median :0.00000   Median :0.000000   Median :0.00000   Median :0.000000
##  Mean   :0.04104   Mean   :0.008639   Mean   :0.01296   Mean   :0.008639
##  3rd Qu.:0.00000   3rd Qu.:0.000000   3rd Qu.:0.00000   3rd Qu.:0.000000
##  Max.   :1.00000   Max.   :1.000000   Max.   :1.00000   Max.   :1.000000
##      class8            class9           class10          class11
##  Min.   :0.00000   Min.   :0.00000   Min.   :0.0000   Min.   :0.00000
##  1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.0000   1st Qu.:0.00000
##  Median :0.00000   Median :0.00000   Median :0.0000   Median :0.00000
##  Mean   :0.00432   Mean   :0.01728   Mean   :0.0108   Mean   :0.00432
##  3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.0000   3rd Qu.:0.00000
##  Max.   :1.00000   Max.   :1.00000   Max.   :1.0000   Max.   :1.00000
##     class12           class13           class14           class15
##  Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.   :0.00000
##  1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000
##  Median :0.00000   Median :0.00000   Median :0.00000   Median :0.00000
##  Mean   :0.00648   Mean   :0.00648   Mean   :0.00648   Mean   :0.00432
##  3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000
##  Max.   :1.00000   Max.   :1.00000   Max.   :1.00000   Max.   :1.00000
##     class16           class17           class18           class19
```

```
##  Min.   :0.000000   Min.   :0.00000   Min.   :0.000000   Min.   :0.00000
##  1st Qu.:0.000000   1st Qu.:0.00000   1st Qu.:0.000000   1st Qu.:0.00000
##  Median :0.000000   Median :0.00000   Median :0.000000   Median :0.00000
##  Mean   :0.008639   Mean   :0.01512   Mean   :0.008639   Mean   :0.01296
##  3rd Qu.:0.000000   3rd Qu.:0.00000   3rd Qu.:0.000000   3rd Qu.:0.00000
##  Max.   :1.000000   Max.   :1.00000   Max.   :1.000000   Max.   :1.00000
##     class20          class21           class22           class23
##  Min.   :0.0000   Min.   :0.00000   Min.   :0.00000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.0000
##  Median :0.0000   Median :0.00000   Median :0.00000   Median :0.0000
##  Mean   :0.0108   Mean   :0.03024   Mean   :0.02376   Mean   :0.0108
##  3rd Qu.:0.0000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.0000
##  Max.   :1.0000   Max.   :1.00000   Max.   :1.00000   Max.   :1.0000
##     class24           class25           class26           class27
##  Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.   :0.00000
##  1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000
##  Median :0.00000   Median :0.00000   Median :0.00000   Median :0.00000
##  Mean   :0.00648   Mean   :0.00648   Mean   :0.00648   Mean   :0.00432
##  3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000
##  Max.   :1.00000   Max.   :1.00000   Max.   :1.00000   Max.   :1.00000
##     class28           class29           class30        courseevaluation
##  Min.   :0.000000   Min.   :0.00000   Min.   :0.00000   Min.   :2.100
##  1st Qu.:0.000000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:3.600
##  Median :0.000000   Median :0.00000   Median :0.00000   Median :4.000
##  Mean   :0.008639   Mean   :0.00432   Mean   :0.01728   Mean   :3.998
##  3rd Qu.:0.000000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:4.400
##  Max.   :1.000000   Max.   :1.00000   Max.   :1.00000   Max.   :5.000
##  didevaluation        female           formal          fulldept
##  Min.   :  5.00   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.: 15.00   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:1.0000
##  Median : 23.00   Median :0.0000   Median :0.0000   Median :1.0000
##  Mean   : 36.62   Mean   :0.4212   Mean   :0.1663   Mean   :0.8942
##  3rd Qu.: 40.00   3rd Qu.:1.0000   3rd Qu.:0.0000   3rd Qu.:1.0000
##  Max.   :380.00   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##      lower        multipleclass      nonenglish        onecredit
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.00000   Min.   :0.00000
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:0.00000
##  Median :0.0000   Median :0.0000   Median :0.00000   Median :0.00000
##  Mean   :0.3391   Mean   :0.3391   Mean   :0.06048   Mean   :0.05832
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:0.00000   3rd Qu.:0.00000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.00000   Max.   :1.00000
##  percentevaluating profevaluation     students       tenuretrack
##  Min.   : 10.42    Min.   :2.300   Min.   :  8.00   Min.   :0.0000
##  1st Qu.: 62.70    1st Qu.:3.800   1st Qu.: 19.00   1st Qu.:1.0000
##  Median : 76.92    Median :4.300   Median : 29.00   Median :1.0000
##  Mean   : 74.43    Mean   :4.175   Mean   : 55.18   Mean   :0.7797
##  3rd Qu.: 87.25    3rd Qu.:4.600   3rd Qu.: 60.00   3rd Qu.:1.0000
##  Max.   :100.00    Max.   :5.000   Max.   :581.00   Max.   :1.0000
##   blkandwhite     btystdvariance     btystdavepos     btystdaveneg
##  Min.   :0.0000   Min.   :0.08503   Min.   :0.0000   Min.   :-1.5388
##  1st Qu.:0.0000   1st Qu.:0.82837   1st Qu.:0.0000   1st Qu.:-0.7446
##  Median :0.0000   Median :1.56579   Median :0.0000   Median :-0.1564
##  Mean   :0.1685   Mean   :1.84263   Mean   :0.2824   Mean   :-0.3708
##  3rd Qu.:0.0000   3rd Qu.:2.68229   3rd Qu.:0.4573   3rd Qu.: 0.0000
##  Max.   :1.0000   Max.   :5.79167   Max.   :1.8817   Max.   : 0.0000
```

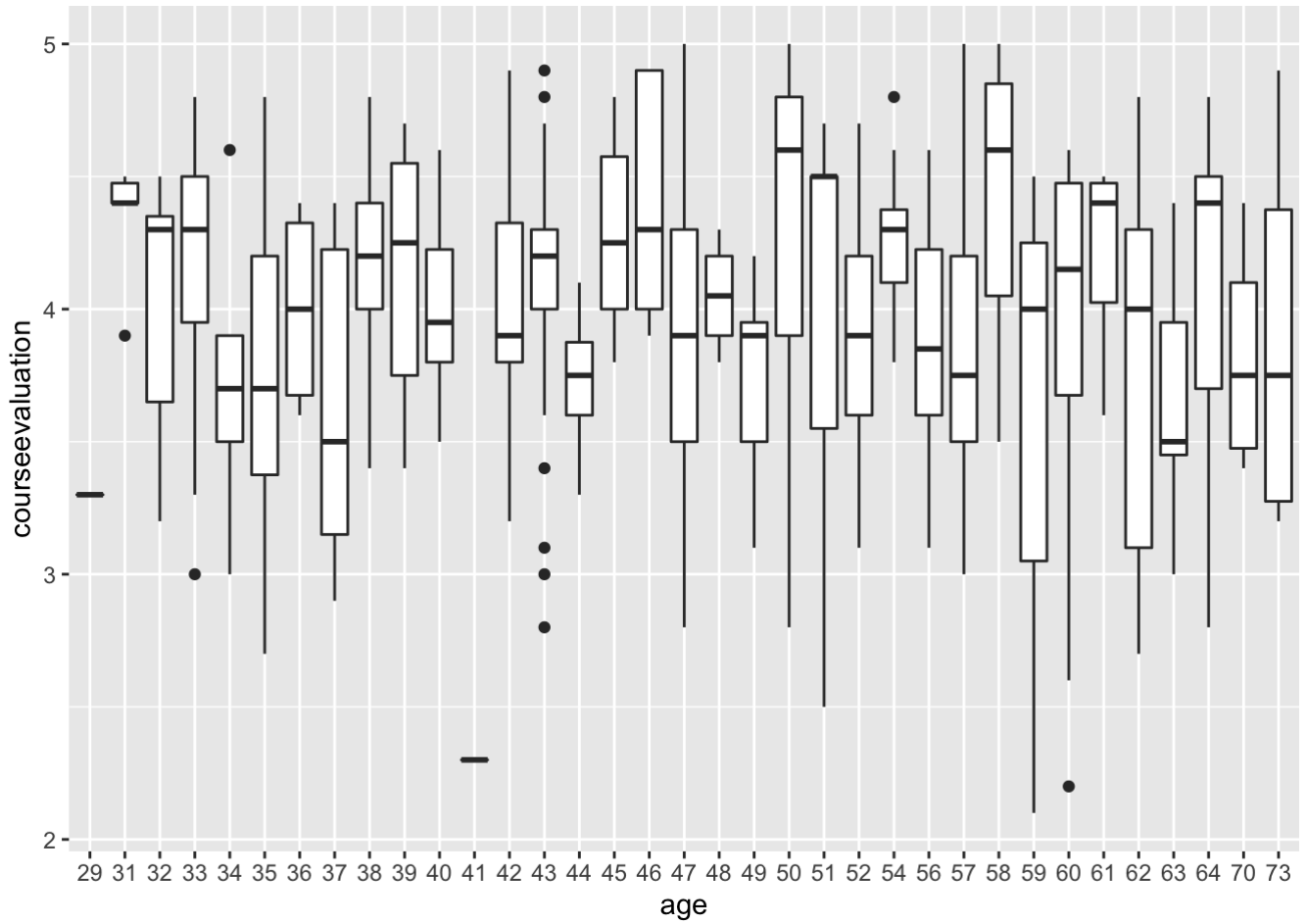
```
# Divided into several parts according to the variable's category.
prof$profnumber <- as.factor(prof$profnumber)
prof$female <- as.factor(prof$female)
prof$age <- as.factor(prof$age)
prof$class <- factor(apply(prof[ ,18:47], FUN=function(r) r %*% 1:30, MARGIN=1))


# Making four boxplot of four main dataset.
ggplot(data=prof, aes(x=class, y=courseevaluation)) + geom_boxplot()
```



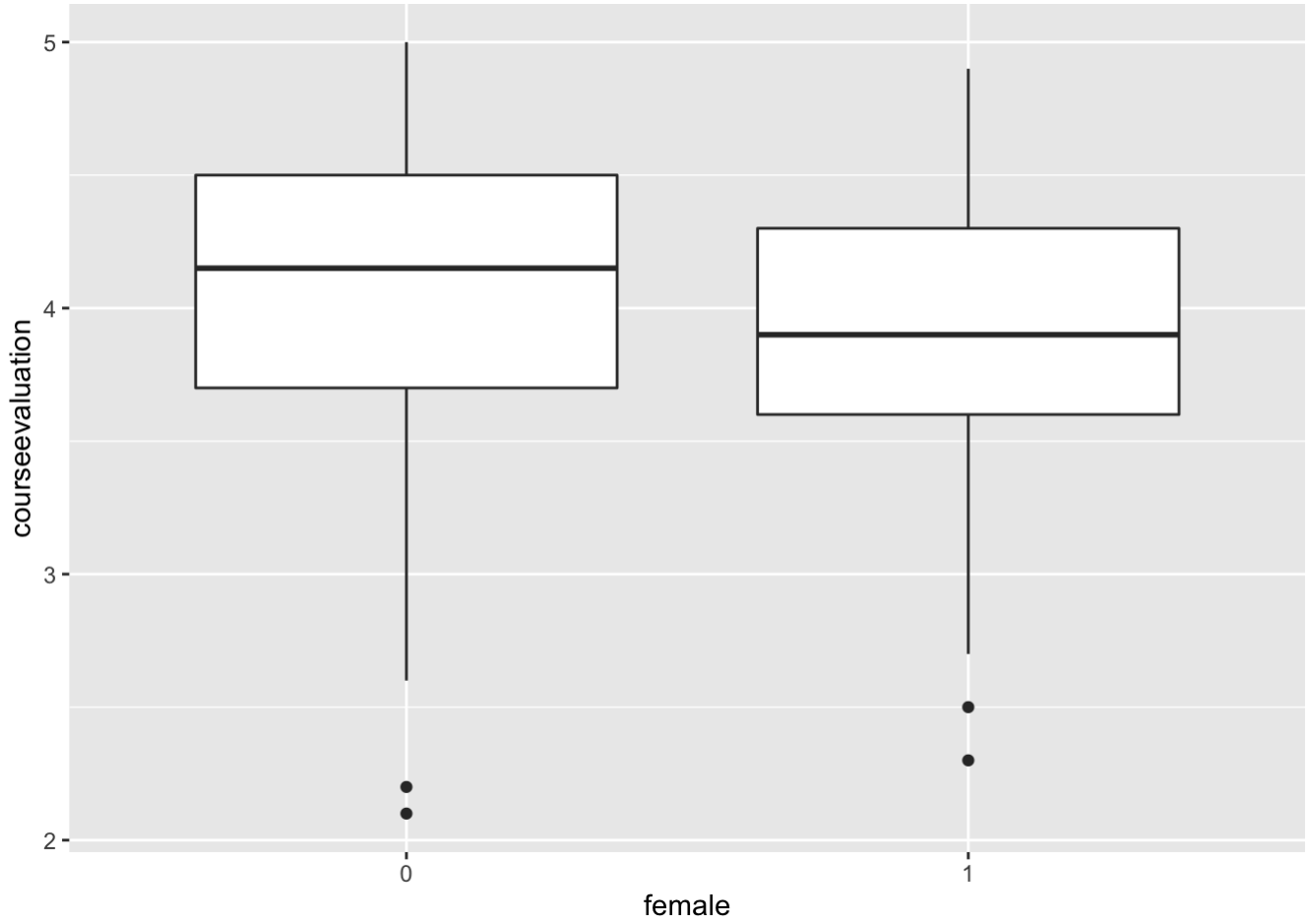

```
ggplot(data=prof, aes(x=profnumber, y=courseevaluation)) + geom_boxplot()
```




```
ggplot(data=prof, aes(x=age, y=courseevaluation)) + geom_boxplot()
```



```
ggplot(data=prof, aes(x=female, y=courseevaluation)) + geom_boxplot()
```




```
# To fit the model.
fit_1 <- lm(courseevaluation ~ female + profnumber + class, data=prof, refresh=0)
```

```
## Warning: In lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##  extra argument 'refresh' will be disregarded
```

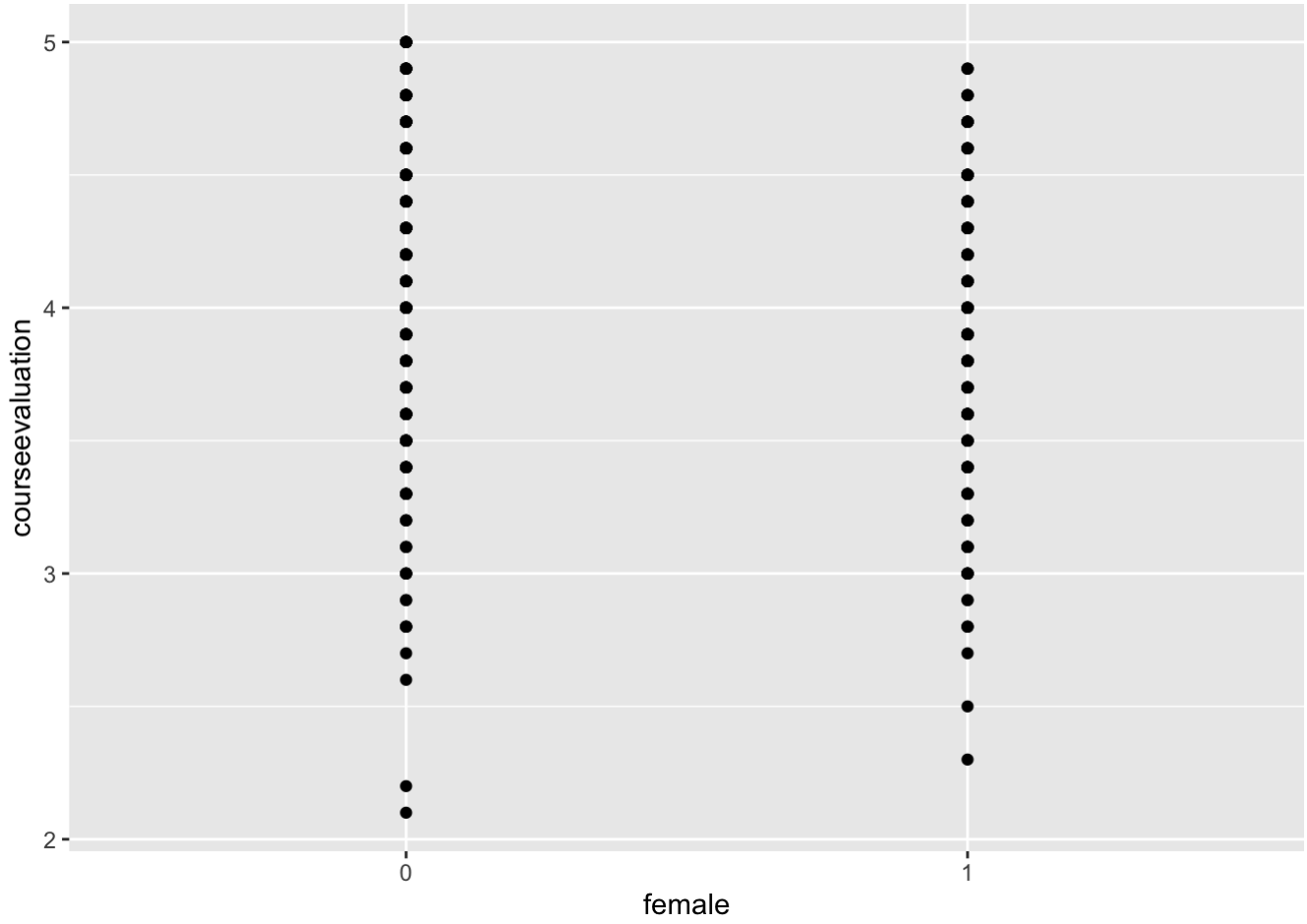
```
summary(fit_1)
```

```
##
## Call:
## lm(formula = courseevaluation ~ female + profnumber + class,
##     data = prof, refresh = 0)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.5286 -0.2062  0.0000  0.2000  0.9667
##
## Coefficients: (1 not defined because of singularities)
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   4.150e+00  1.595e-01  26.019  < 2e-16 ***
## female1      -2.539e-02  3.557e-01  -0.071 0.943130
## profnumber2  -6.167e-01  2.763e-01  -2.232 0.026253 *
## profnumber3  -4.477e-01  3.515e-01  -1.274 0.203696
## profnumber4  -1.664e-01  3.520e-01  -0.473 0.636665
## profnumber5   2.042e-01  3.576e-01   0.571 0.568428
## profnumber6   3.650e-01  2.214e-01   1.648 0.100220
## profnumber7  -5.782e-02  3.377e-01  -0.171 0.864156
## profnumber8  -9.604e-02  3.505e-01  -0.274 0.784277
## profnumber9   1.068e-01  3.529e-01   0.303 0.762334
## profnumber10  5.109e-01  2.102e-01   2.431 0.015595 *
## profnumber11 -6.310e-01  3.132e-01  -2.014 0.044754 *
## profnumber12 -4.354e-02  2.443e-01  -0.178 0.858651
## profnumber13 -3.395e-01  2.425e-01  -1.400 0.162463
## profnumber14 -4.488e-01  2.628e-01  -1.708 0.088539 .
## profnumber15 -1.062e+00  4.164e-01  -2.550 0.011218 *
## profnumber16  9.558e-02  3.372e-01   0.283 0.777001
## profnumber17 -2.674e-01  4.048e-01  -0.661 0.509321
## profnumber18 -1.000e-01  2.110e-01  -0.474 0.635844
## profnumber19  2.198e-01  3.436e-01   0.640 0.522683
## profnumber20 -6.446e-01  3.411e-01  -1.890 0.059626 .
## profnumber21 -5.189e-01  3.651e-01  -1.421 0.156094
## profnumber22 -1.050e+00  7.871e-01  -1.333 0.183281
## profnumber23 -3.141e-01  3.640e-01  -0.863 0.388866
## profnumber24  3.214e-01  2.174e-01   1.479 0.140123
## profnumber25  1.254e-01  4.212e-01   0.298 0.766101
## profnumber26 -7.500e-02  4.885e-01  -0.154 0.878083
## profnumber27 -1.926e-01  2.802e-01  -0.687 0.492273
## profnumber28 -2.628e-01  3.029e-01  -0.868 0.386150
## profnumber29  3.611e-01  4.098e-01   0.881 0.378819
## profnumber30 -3.246e-01  6.947e-01  -0.467 0.640631
## profnumber31 -4.850e-01  2.230e-01  -2.175 0.030319 *
## profnumber32  1.500e-01  3.190e-01   0.470 0.638495
## profnumber33 -2.738e-02  2.404e-01  -0.114 0.909386
## profnumber34 -2.642e-01  3.628e-01  -0.728 0.467015
## profnumber35 -1.327e-01  3.604e-01  -0.368 0.712983
## profnumber36 -2.281e-01  3.818e-01  -0.597 0.550629
## profnumber37 -6.435e-01  2.278e-01  -2.825 0.005012 **
## profnumber38 -1.115e-01  2.836e-01  -0.393 0.694316
## profnumber39  2.303e-01  2.349e-01   0.980 0.327547
## profnumber40 -2.926e-01  5.545e-01  -0.528 0.598049
## profnumber41  5.100e-01  2.366e-01   2.156 0.031804 *
## profnumber42  5.500e-01  3.190e-01   1.724 0.085589 .
## profnumber43  1.316e-01  3.834e-01   0.343 0.731664
## profnumber44  2.833e-01  4.785e-01   0.592 0.554153
## profnumber45  4.812e-01  2.672e-01   1.801 0.072573 .
```

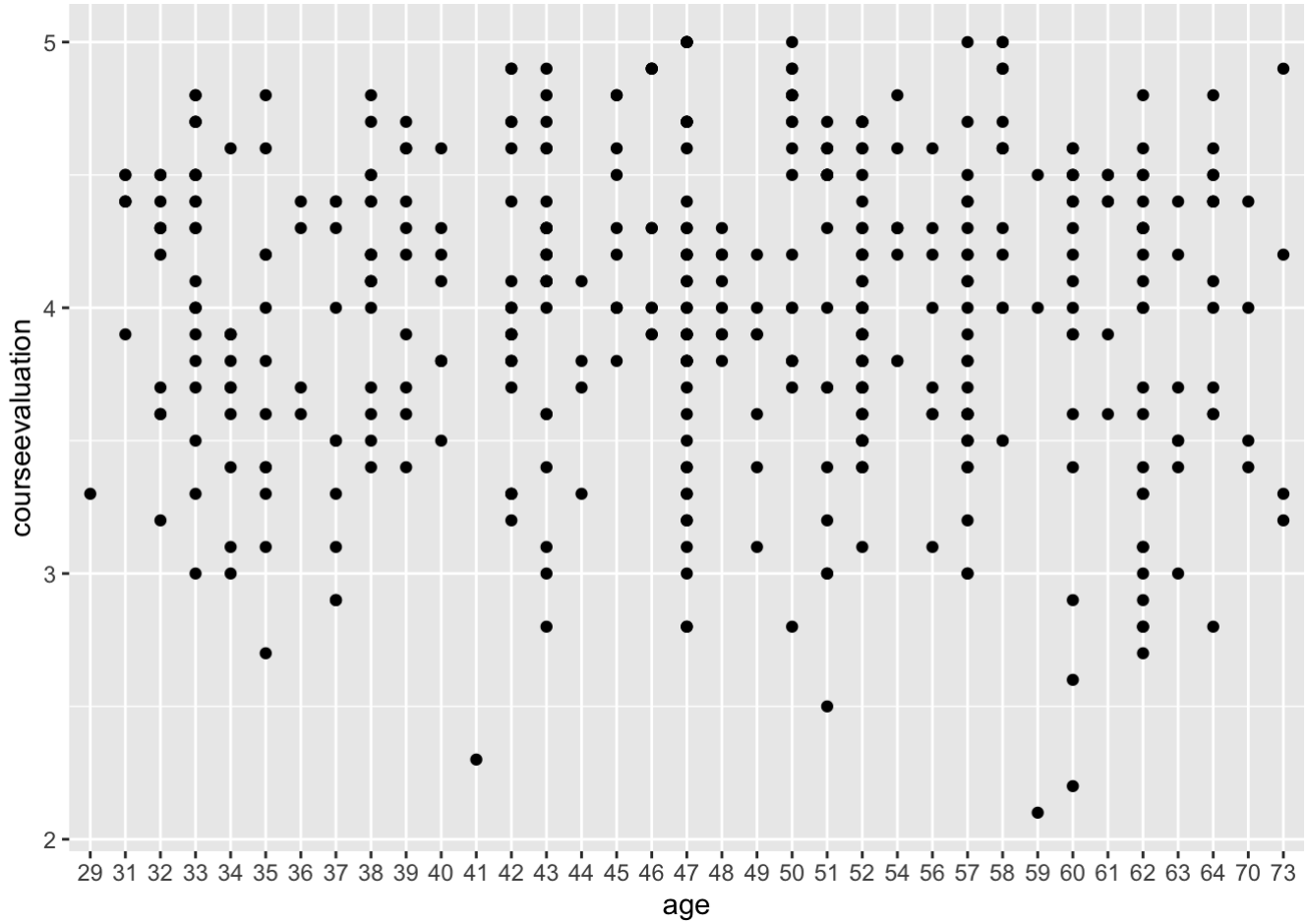
```
## profnumber46  1.500e-01  4.785e-01   0.313 0.754105
## profnumber47 -1.350e+00  5.557e-01  -2.429 0.015678 *
## profnumber48 -1.579e-01  5.519e-01  -0.286 0.774916
## profnumber49 -3.695e-01  3.582e-01  -1.031 0.303056
## profnumber50 -3.500e-01  2.059e-01  -1.700 0.090091 .
## profnumber51  4.754e-01  3.557e-01   1.337 0.182266
## profnumber52  2.536e-02  2.597e-01   0.098 0.922275
## profnumber53  2.334e-01  3.783e-01   0.617 0.537722
## profnumber54 -4.079e-01  3.557e-01  -1.147 0.252227
## profnumber55 -1.064e+00  2.964e-01  -3.588 0.000382 ***
## profnumber56 -3.182e-01  2.452e-01  -1.298 0.195316
## profnumber57 -2.246e-01  4.212e-01  -0.533 0.594187
## profnumber58 -2.146e-01  3.411e-01  -0.629 0.529644
## profnumber59 -8.000e-01  3.888e-01  -2.058 0.040375 *
## profnumber60 -1.158e+00  3.898e-01  -2.971 0.003185 **
## profnumber61 -1.500e-01  4.220e-01  -0.355 0.722467
## profnumber62  1.500e-01  4.220e-01   0.355 0.722467
## profnumber63 -3.246e-01  4.212e-01  -0.771 0.441414
## profnumber64 -4.913e-01  3.898e-01  -1.260 0.208432
## profnumber65 -1.103e-01  3.505e-01  -0.315 0.753168
## profnumber66 -4.667e-01  2.256e-01  -2.069 0.039313 *
## profnumber67 -2.000e-01  3.190e-01  -0.627 0.531101
## profnumber68 -1.583e+00  2.763e-01  -5.731 2.20e-08 ***
## profnumber69 -1.125e+00  5.037e-01  -2.233 0.026221 *
## profnumber70  3.611e-01  2.059e-01   1.754 0.080380 .
## profnumber71  6.300e-01  2.017e-01   3.123 0.001946 **
## profnumber72 -3.000e-01  2.256e-01  -1.330 0.184411
## profnumber73  4.300e-01  2.366e-01   1.818 0.070004 .
## profnumber74 -3.000e-01  2.522e-01  -1.190 0.235039
## profnumber75 -7.461e-02  5.298e-01  -0.141 0.888085
## profnumber76 -9.246e-01  4.212e-01  -2.195 0.028821 *
## profnumber77  1.254e-01  3.860e-01   0.325 0.745464
## profnumber78  2.570e-02  2.790e-01   0.092 0.926647
## profnumber79 -3.500e-01  2.763e-01  -1.267 0.206048
## profnumber80 -2.996e-01  3.731e-01  -0.803 0.422573
## profnumber81  2.780e-01  2.577e-01   1.078 0.281596
## profnumber82 -6.270e-02  2.034e-01  -0.308 0.758140
## profnumber83  5.539e-02  3.628e-01   0.153 0.878731
## profnumber84  8.613e-02  4.583e-01   0.188 0.851032
## profnumber85  6.643e-01  3.102e-01   2.141 0.032972 *
## profnumber86  1.167e-01  2.763e-01   0.422 0.673065
## profnumber87  5.000e-02  3.190e-01   0.157 0.875541
## profnumber88 -9.929e-01  2.174e-01  -4.568 6.91e-06 ***
## profnumber89 -5.246e-01  3.898e-01  -1.346 0.179266
## profnumber90 -4.000e-01  3.190e-01  -1.254 0.210728
## profnumber91  3.421e-01  3.898e-01   0.878 0.380836
## profnumber92 -2.103e-01  3.505e-01  -0.600 0.548912
## profnumber93         NA         NA      NA       NA
## profnumber94 -6.496e-01  3.731e-01  -1.741 0.082604 .
## class1        1.986e-01  2.489e-01   0.798 0.425551
## class2        3.052e-01  3.021e-01   1.010 0.313120
## class3       -1.246e-01  2.508e-01  -0.497 0.619662
## class4       -2.523e-01  1.476e-01  -1.709 0.088355 .
## class5        1.869e-01  2.144e-01   0.872 0.383960
## class6       -2.074e-01  3.220e-01  -0.644 0.519901
## class7       -6.000e-01  2.996e-01  -2.002 0.046040 *
## class8        2.250e-01  6.049e-01   0.372 0.710143
## class9       -1.706e-02  2.465e-01  -0.069 0.944860
```

```
## class10       4.713e-01  2.992e-01   1.575 0.116151
## class11       5.409e-01  3.226e-01   1.677 0.094519 .
## class12      -1.419e-01  2.612e-01  -0.543 0.587402
## class13      -1.167e-01  3.041e-01  -0.384 0.701456
## class14      -3.574e-01  3.597e-01  -0.994 0.321115
## class15      -1.500e+00  4.785e-01  -3.135 0.001870 **
## class16       2.881e-01  2.821e-01   1.021 0.307973
## class17       2.846e-01  1.920e-01   1.482 0.139175
## class18       1.719e-01  2.629e-01   0.654 0.513643
## class19      -6.861e-01  3.230e-01  -2.124 0.034375 *
## class20       4.882e-01  2.127e-01   2.296 0.022307 *
## class21      -3.624e-01  1.764e-01  -2.054 0.040698 *
## class22      -5.000e-01  3.907e-01  -1.280 0.201491
## class23       7.250e-01  2.348e-01   3.088 0.002181 **
## class24      -3.000e-01  4.642e-01  -0.646 0.518544
## class25       3.858e-15  4.444e-01   0.000 1.000000
## class26       1.272e-01  2.476e-01   0.514 0.607819
## class27       6.769e-02  3.044e-01   0.222 0.824151
## class28       1.375e-01  3.383e-01   0.406 0.684712
## class29      -4.508e-01  3.202e-01  -1.408 0.160153
## class30      -9.570e-03  2.792e-01  -0.034 0.972677
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3907 on 339 degrees of freedom
## Multiple R-squared:  0.6362, Adjusted R-squared:  0.5042
## F-statistic:  4.82 on 123 and 339 DF,  p-value: < 2.2e-16
```

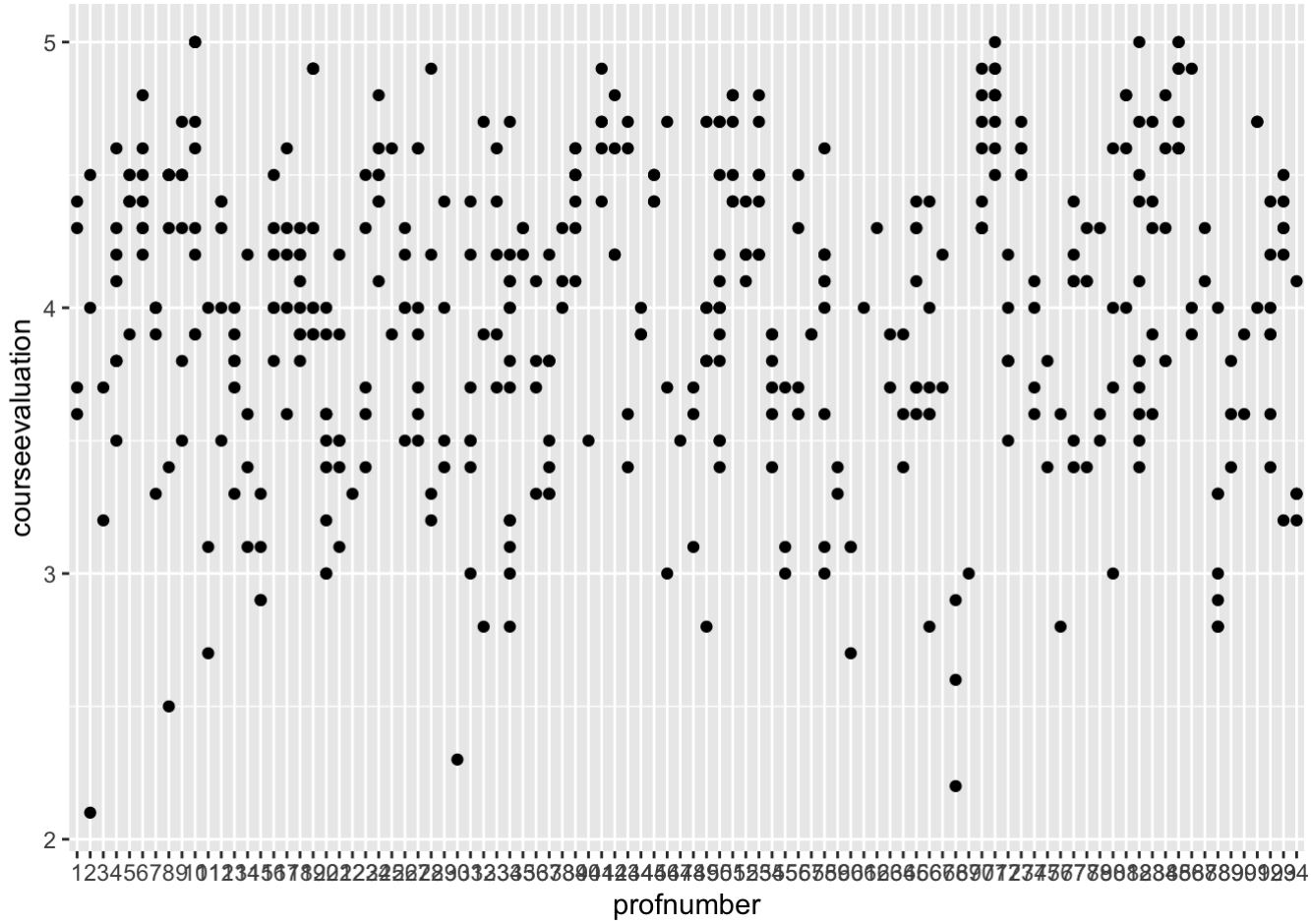
```
# plot the fitted model and its residual.
ggplot(data=prof, mapping= aes(x=female, y=courseevaluation)) +
  geom_point() +
  stat_smooth(method="lm", formula=y ~ x, se=TRUE)
```



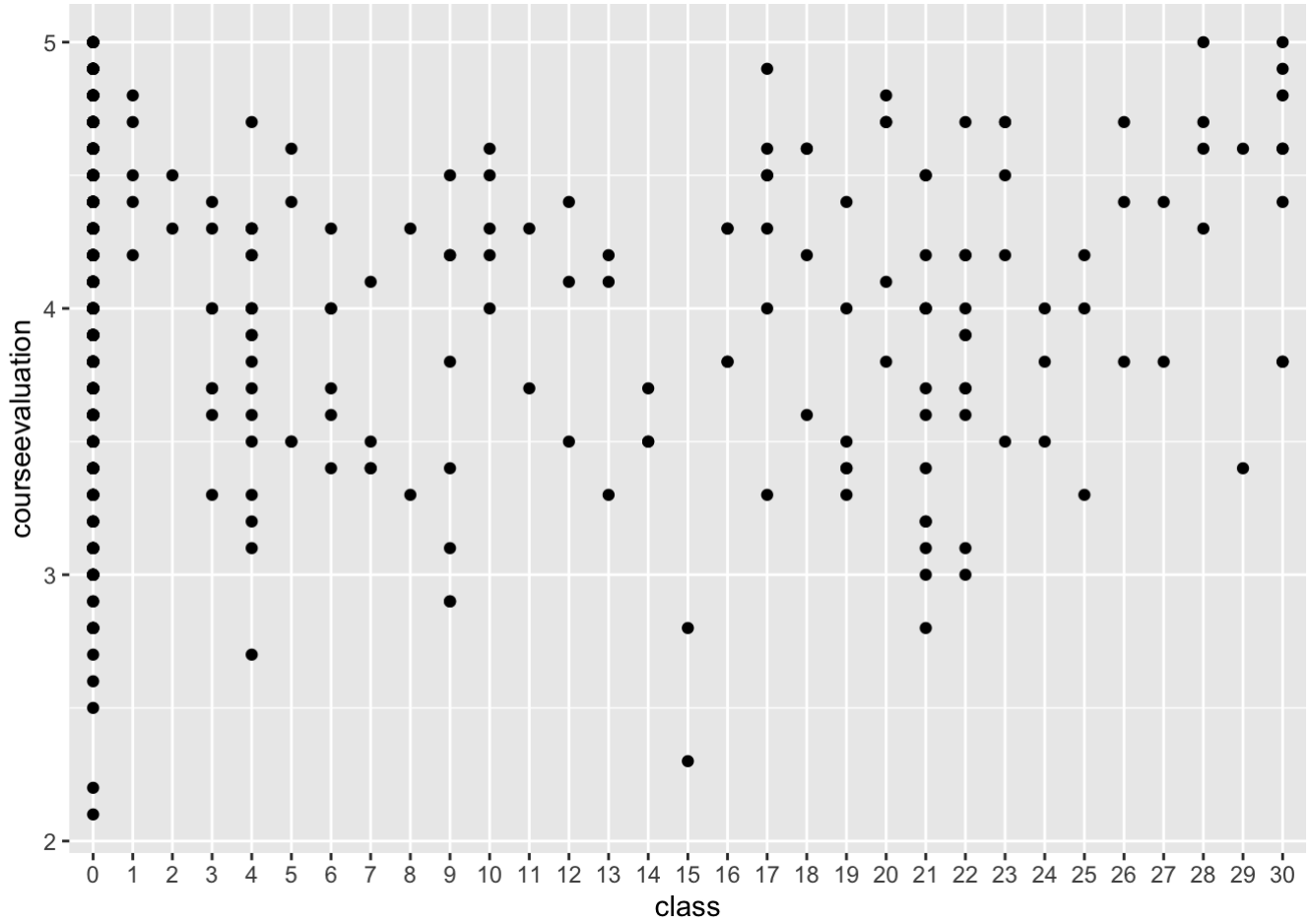

```
ggplot(data=prof, mapping= aes(x=age, y=courseevaluation)) +
  geom_point() +
  stat_smooth(method="lm", formula=y ~ x, se=TRUE)
```



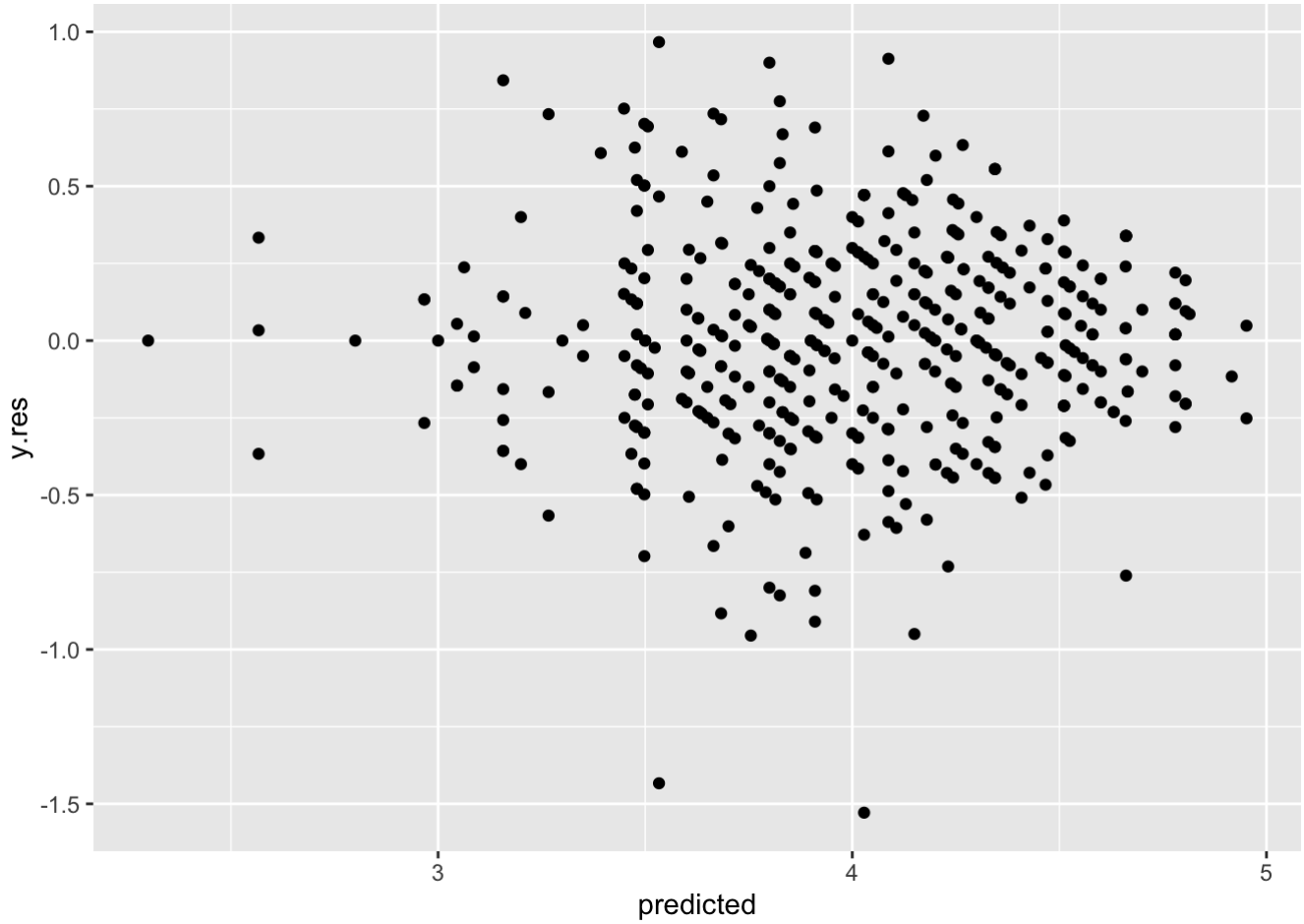

```
ggplot(data=prof, mapping= aes(x=profnumber, y=courseevaluation)) +
  geom_point() +
  stat_smooth(method="lm", formula=y ~ x, se=TRUE)
```




```
ggplot(data=prof, mapping= aes(x=class, y=courseevaluation))+
  geom_point() +
  stat_smooth(method="lm", formula=y ~ x, se=TRUE)
```




```
y.res <- resid(fit_1)
predicted <- predict(fit_1)
data <- data.frame(y.res,predicted)
ggplot(data=data, mapping= aes(x=predicted, y=y.res)) +
  geom_point()
```




```
# display the number of observations for each combination of professor and class
fit_2 <- stan_glm(courseevaluation ~ female + profevaluation, data=prof)
```

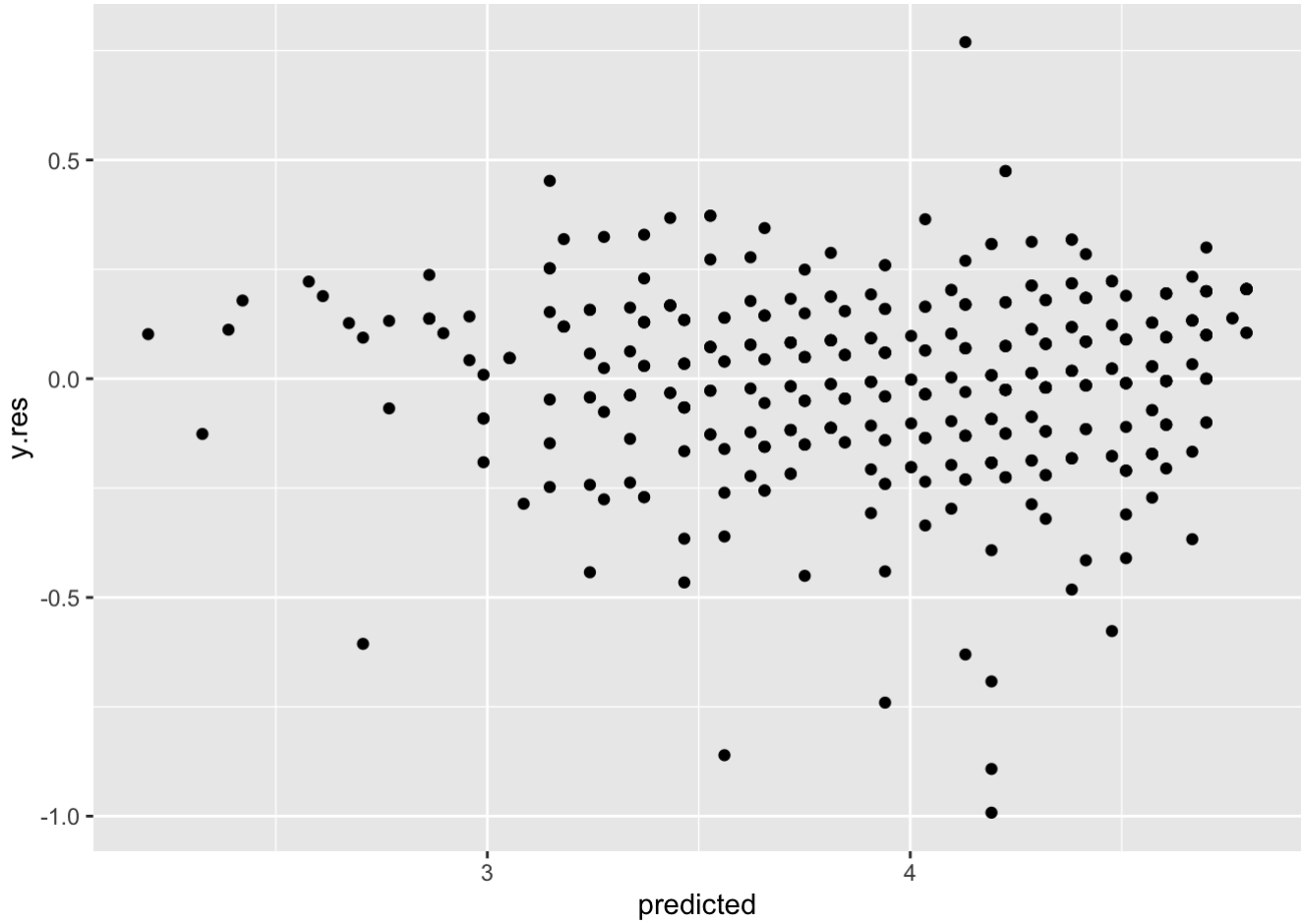
```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 1.9e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.19 s
econds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.038365 seconds (Warm-up)
## Chain 1:                0.072082 seconds (Sampling)
## Chain 1:                0.110447 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 1.9e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.19 s
econds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.034345 seconds (Warm-up)
## Chain 2:                0.07332 seconds (Sampling)
## Chain 2:                0.107665 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 1.9e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.19 s
```

```
econds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.03747 seconds (Warm-up)
## Chain 3:                0.067676 seconds (Sampling)
## Chain 3:                0.105146 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 1.5e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.15 s
econds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.037082 seconds (Warm-up)
## Chain 4:                0.071901 seconds (Sampling)
## Chain 4:                0.108983 seconds (Total)
## Chain 4:
```

```
summary(fit_2)
```

```
##
## Model Info:
##  function:     stan_glm
##  family:       gaussian [identity]
##  formula:      courseevaluation ~ female + profevaluation
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 463
##  predictors:   3
##
## Estimates:
##                  mean   sd   10%   50%   90%
## (Intercept)     0.0    0.1  0.0   0.0   0.1
## female1         0.0    0.0 -0.1   0.0   0.0
## profevaluation  0.9    0.0  0.9   0.9   1.0
## sigma           0.2    0.0  0.2   0.2   0.2
##
## Fit Diagnostics:
##            mean   sd   10%   50%   90%
## mean_PPD 4.0    0.0  4.0   4.0   4.0
##
## The mean_ppd is the sample average posterior predictive distribution of the outcom
e variable (for details see help('summary.stanreg')).
##
## MCMC diagnostics
##                mcse Rhat n_eff
## (Intercept)    0.0  1.0  4472
## female1        0.0  1.0  4929
## profevaluation 0.0  1.0  4484
## sigma          0.0  1.0  4684
## mean_PPD       0.0  1.0  4186
## log-posterior  0.0  1.0  1758
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure o
f effective sample size, and Rhat is the potential scale reduction factor on split ch
ains (at convergence Rhat=1).
```

```
y.res <- resid(fit_2)
predicted <- predict(fit_2)
data <- data.frame(y.res,predicted)
ggplot(data=data, mapping= aes(x=predicted, y=y.res)) +
  geom_point()
```




```
fit_3 <- stan_glm(courseevaluation ~ female + age, data=prof)
```

```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 9.6e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.96 s
econds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.225505 seconds (Warm-up)
## Chain 1:                0.332946 seconds (Sampling)
## Chain 1:                0.558451 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 1.9e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.19 s
econds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.197677 seconds (Warm-up)
## Chain 2:                0.307262 seconds (Sampling)
## Chain 2:                0.504939 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 1.9e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.19 s
```

```
econds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.277393 seconds (Warm-up)
## Chain 3:                0.327956 seconds (Sampling)
## Chain 3:                0.605349 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 2e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.2 se
conds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.265979 seconds (Warm-up)
## Chain 4:                0.388715 seconds (Sampling)
## Chain 4:                0.654694 seconds (Total)
## Chain 4:
```

```
## Warning: Bulk Effective Samples Size (ESS) is too low, indicating posterior means
and medians may be unreliable.
## Running the chains for more iterations may help. See
## http://mc-stan.org/misc/warnings.html#bulk-ess
```

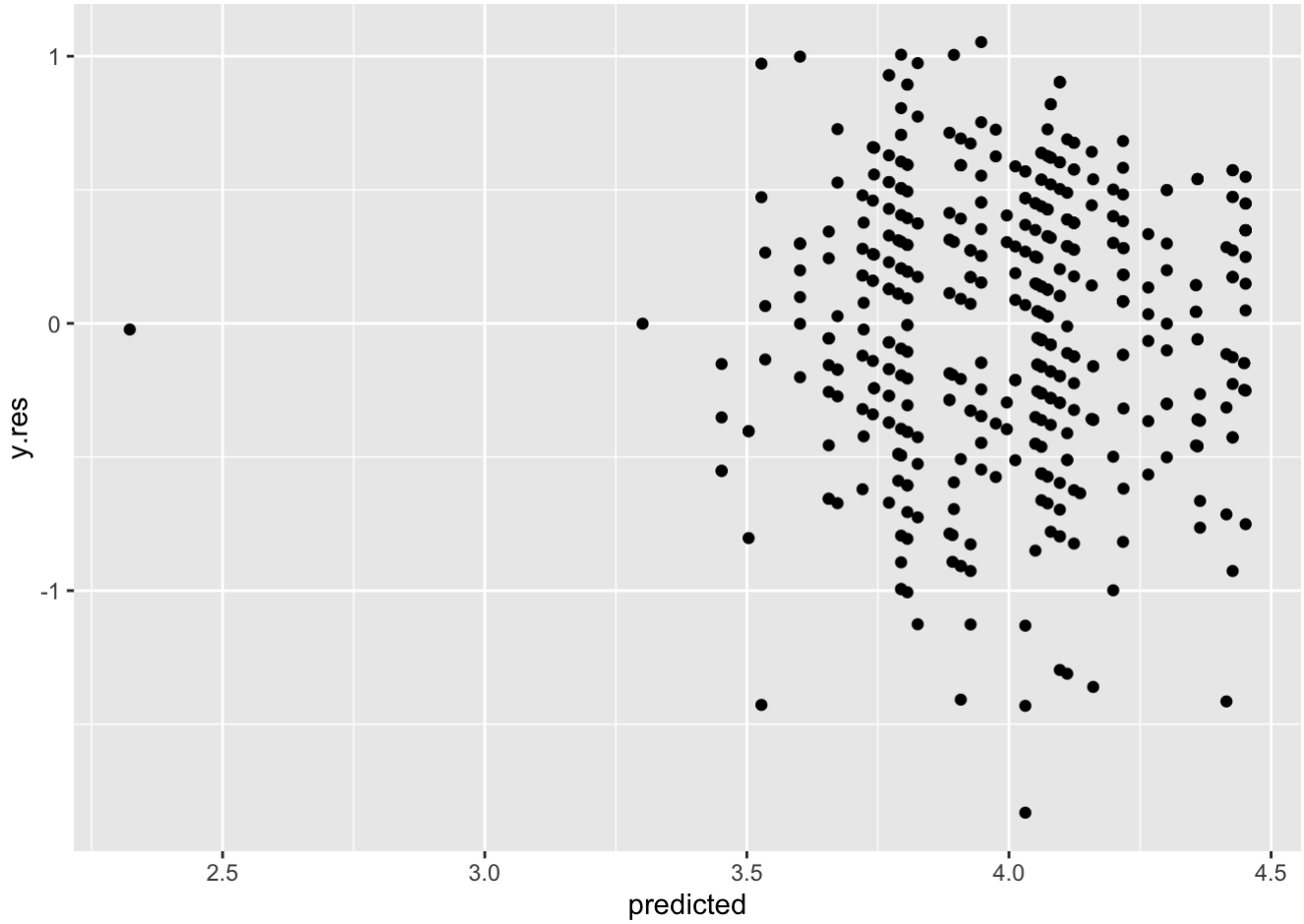
```
## Warning: Tail Effective Samples Size (ESS) is too low, indicating posterior varian
ces and tail quantiles may be unreliable.
## Running the chains for more iterations may help. See
## http://mc-stan.org/misc/warnings.html#tail-ess
```

```
summary(fit_3)
```

```
##
## Model Info:
##  function:     stan_glm
##  family:       gaussian [identity]
##  formula:      courseevaluation ~ female + age
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 463
##  predictors:   36
##
## Estimates:
##               mean   sd   10%   50%   90%
## (Intercept)  3.6    0.5  3.0   3.6   4.2
## female1     -0.3    0.1 -0.4  -0.3  -0.2
## age31        1.0    0.5  0.3   1.1   1.7
## age32        0.4    0.5 -0.2   0.5   1.1
## age33        0.8    0.5  0.2   0.8   1.5
## age34        0.3    0.5 -0.4   0.3   0.9
## age35        0.2    0.5 -0.5   0.2   0.9
## age36        0.7    0.6  0.0   0.7   1.4
## age37        0.1    0.5 -0.5   0.2   0.8
## age38        0.8    0.5  0.1   0.8   1.4
## age39        0.7    0.5  0.0   0.7   1.3
## age40        0.7    0.5  0.0   0.7   1.4
## age41       -1.0    0.7 -2.0  -1.0  -0.1
## age42        0.5    0.5 -0.2   0.5   1.1
## age43        0.6    0.5  0.0   0.6   1.3
## age44        0.4    0.6 -0.3   0.4   1.1
## age45        0.7    0.5  0.0   0.7   1.4
## age46        1.0    0.5  0.4   1.1   1.7
## age47        0.5    0.5 -0.2   0.5   1.1
## age48        0.4    0.5 -0.2   0.5   1.1
## age49        0.1    0.5 -0.6   0.1   0.8
## age50        0.8    0.5  0.2   0.9   1.5
## age51        0.6    0.5 -0.1   0.6   1.2
## age52        0.5    0.5 -0.2   0.5   1.1
## age54        0.8    0.5  0.2   0.9   1.5
## age56        0.6    0.5 -0.1   0.6   1.2
## age57        0.3    0.5 -0.3   0.4   1.0
## age58        0.8    0.5  0.2   0.8   1.5
## age59       -0.1    0.6 -0.8  -0.1   0.7
## age60        0.4    0.5 -0.2   0.4   1.1
## age61        0.6    0.5 -0.1   0.6   1.3
## age62        0.2    0.5 -0.5   0.2   0.8
## age63        0.1    0.5 -0.6   0.1   0.8
## age64        0.5    0.5 -0.1   0.5   1.1
## age70        0.2    0.6 -0.5   0.2   0.9
## age73        0.3    0.6 -0.4   0.3   1.0
## sigma        0.5    0.0  0.5   0.5   0.5
##
## Fit Diagnostics:
##            mean   sd   10%   50%   90%
## mean_PPD 4.0    0.0  4.0   4.0   4.0
##
## The mean_ppd is the sample average posterior predictive distribution of the outcom
e variable (for details see help('summary.stanreg')).
```

```
##
## MCMC diagnostics
##               mcse Rhat n_eff
## (Intercept)   0.0  1.0   183
## female1       0.0  1.0  1687
## age31         0.0  1.0   219
## age32         0.0  1.0   202
## age33         0.0  1.0   191
## age34         0.0  1.0   203
## age35         0.0  1.0   197
## age36         0.0  1.0   230
## age37         0.0  1.0   205
## age38         0.0  1.0   196
## age39         0.0  1.0   201
## age40         0.0  1.0   203
## age41         0.0  1.0   343
## age42         0.0  1.0   192
## age43         0.0  1.0   192
## age44         0.0  1.0   227
## age45         0.0  1.0   207
## age46         0.0  1.0   207
## age47         0.0  1.0   188
## age48         0.0  1.0   205
## age49         0.0  1.0   200
## age50         0.0  1.0   193
## age51         0.0  1.0   193
## age52         0.0  1.0   188
## age54         0.0  1.0   204
## age56         0.0  1.0   206
## age57         0.0  1.0   193
## age58         0.0  1.0   194
## age59         0.0  1.0   239
## age60         0.0  1.0   195
## age61         0.0  1.0   217
## age62         0.0  1.0   192
## age63         0.0  1.0   211
## age64         0.0  1.0   197
## age70         0.0  1.0   231
## age73         0.0  1.0   232
## sigma         0.0  1.0  1312
## mean_PPD      0.0  1.0  2197
## log-posterior 0.1  1.0  1344
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure o
f effective sample size, and Rhat is the potential scale reduction factor on split ch
ains (at convergence Rhat=1).
```

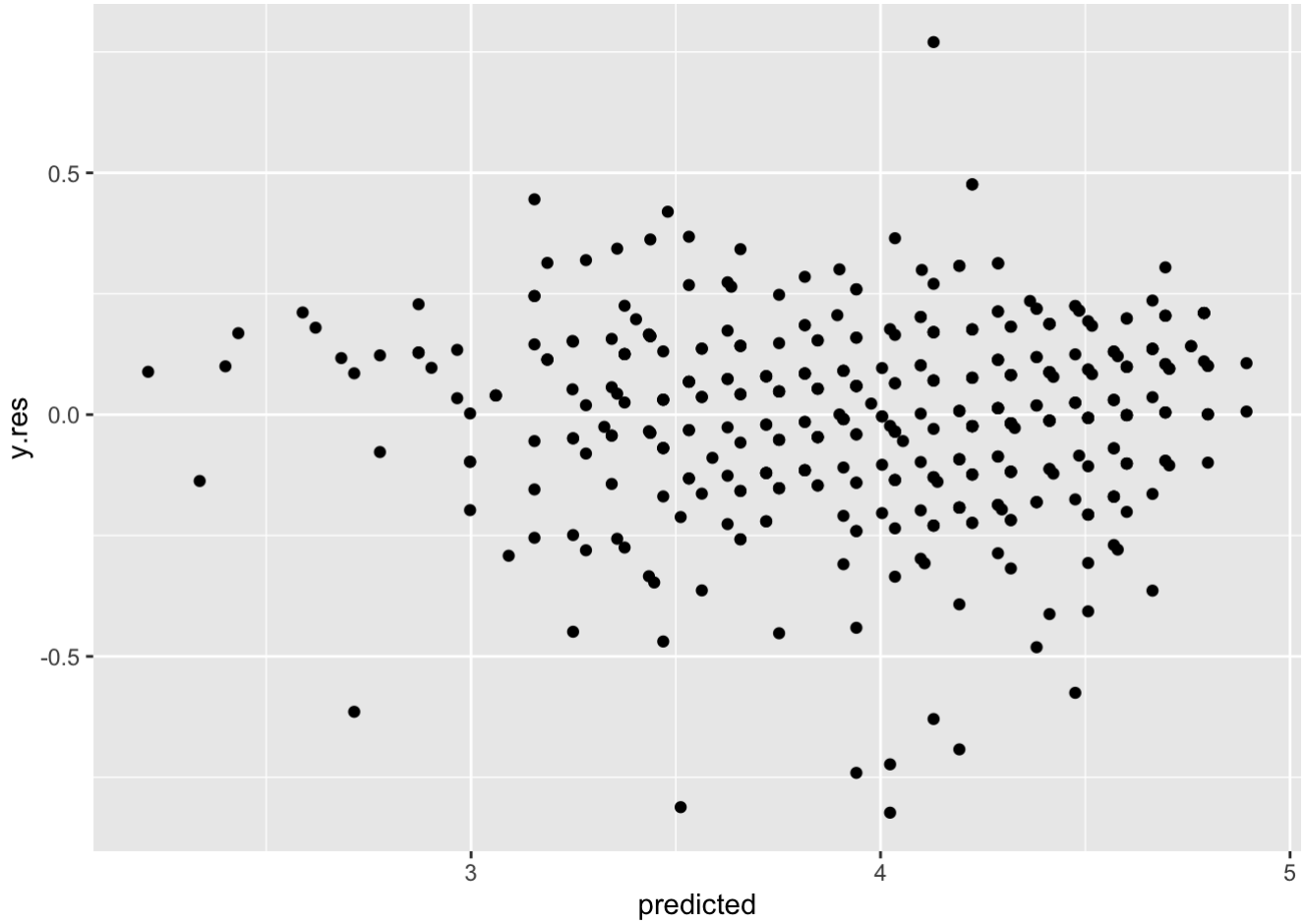
```
y.res <- resid(fit_3)
predicted <- predict(fit_3)
data <- data.frame(y.res,predicted)
ggplot(data=data, mapping= aes(x=predicted, y=y.res)) +
  geom_point()
```




```
prof$profevaluation <- (prof$profevaluation - mean(prof$profevaluation)) / (2 * sd(pr
of$profevaluation))
fit_4 <- stan_glm(courseevaluation ~ female + onecredit + profevaluation*nonenglish,
 data=prof, refresh=0)
summary(fit_4)
```

```
##
## Model Info:
##  function:     stan_glm
##  family:       gaussian [identity]
##  formula:      courseevaluation ~ female + onecredit + profevaluation * nonenglish
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 463
##  predictors:   6
##
## Estimates:
##                              mean   sd   10%   50%   90%
## (Intercept)                 4.0    0.0  4.0   4.0   4.0
## female1                     0.0    0.0 -0.1   0.0   0.0
## onecredit                   0.1    0.0  0.1   0.1   0.2
## profevaluation              1.0    0.0  1.0   1.0   1.0
## nonenglish                 -0.1    0.0 -0.2  -0.1  -0.1
## profevaluation:nonenglish -0.2    0.1 -0.3  -0.2  -0.1
## sigma                       0.2    0.0  0.2   0.2   0.2
##
## Fit Diagnostics:
##            mean   sd   10%   50%   90%
## mean_PPD 4.0    0.0  4.0   4.0   4.0
##
## The mean_ppd is the sample average posterior predictive distribution of the outcom
e variable (for details see help('summary.stanreg')).
##
## MCMC diagnostics
##                            mcse Rhat n_eff
## (Intercept)                0.0  1.0  4872
## female1                    0.0  1.0  4807
## onecredit                  0.0  1.0  5503
## profevaluation             0.0  1.0  4567
## nonenglish                 0.0  1.0  4250
## profevaluation:nonenglish 0.0  1.0  4208
## sigma                      0.0  1.0  4276
## mean_PPD                   0.0  1.0  4339
## log-posterior              0.0  1.0  1855
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure o
f effective sample size, and Rhat is the potential scale reduction factor on split ch
ains (at convergence Rhat=1).
```

```
y.res <- resid(fit_4)
predicted <- predict(fit_4)
data <- data.frame(y.res,predicted)
ggplot(data=data, mapping= aes(x=predicted, y=y.res)) +
  geom_point()
```




```
# In this situation, I choose model 4 as my best choice. It is because in this situat
ion, where residual plot are more likely centralized in 0, and have no trend in it, s
ince profevaluation have a transformation.
```

12.14

Prediction from a fitted regression: Consider one of the fitted models for mesquite leaves, for example fit_4, in Section 12.6. Suppose you wish to use this model to make inferences about the average mesquite yield in a new set of trees whose predictors are in data frame called new_trees. Give R code to obtain an estimate and standard error for this population average. You do not need to make the prediction; just give the code.

```
fit_4 <- lm(log(mort) ~ hc_1+ nox_1 + so2_1, data=pollution)
summary(fit_4)
```

```
##
## Call:
## lm(formula = log(mort) ~ hc_1 + nox_1 + so2_1, data = pollution)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.112676 -0.033540 -0.003781  0.041982  0.168553
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.844105   0.007098 964.211  < 2e-16 ***
## hc_1        -0.323153   0.118398  -2.729  0.00846 **
## nox_1        0.296217   0.124494   2.379  0.02077 *
## so2_1        0.026428   0.023236   1.137  0.26022
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.05498 on 56 degrees of freedom
## Multiple R-squared:  0.3473, Adjusted R-squared:  0.3123
## F-statistic: 9.931 on 3 and 56 DF,  p-value: 2.39e-05
```

```
# Imagine that we have a new data set called new_trees, which contain hc_new, nox_ne
w, so2_new. So, we can:
prediction <- predict(fit_4, data=new_trees)
mean <- mean(prediction)
standard_error <- sd(prediction)
```