INFX 573: Problem Set 6 - Regression

Rajendran Seetharaman

Due: Tuesday, November 21, 2017

Problem Set 6

Collaborators:

Instructions:

Before beginning this assignment, please ensure you have access to R and RStudio.

1. Replace the "Insert Your Name Here" text in the `author:` field with your own name. List all collaborators on the top of your assignment.
2. Be sure to include well-documented (e.g. commented) code chucks, figures and clearly written text chunk explanations as necessary. Any figures should be clearly labeled and appropriately referenced within the text.
3. Collaboration on problem sets is fun and useful but turn in an individual write-up in your own words and involving your own code. Do not just copy-and-paste from others' responses or code.
4. When you have completed the assignment and have **checked** that your code both runs in the Console and knits correctly when you click `Knit PDF`, rename the R Markdown file to `YourLastName_YourFirstName_ps6.Rmd`, knit a PDF and submit the PDF file on Canvas.

1. Housing Values in Suburbs of Boston

In this problem we will use the Boston dataset that is available in *MASS* package. This dataset contains information about median house value for 506 neighborhoods in Boston, MA.

```
library(MASS)
```

1.1 Describe data

Describe the data and variables that are part of the *Boston* dataset. Tidy data as necessary.

```
library(tidyverse)
```

```
## Loading tidyverse: ggplot2
## Loading tidyverse: tibble
## Loading tidyverse: tidyr
## Loading tidyverse: readr
## Loading tidyverse: purrr
## Loading tidyverse: dplyr
```

```
## Conflicts with tidy packages ----------------------------------------------
```

```
## filter(): dplyr, stats
## lag():    dplyr, stats
## select(): dplyr, MASS
```

```
#convert chas to a categorical variable for analysis
Boston <- Boston %>% mutate(chas=as.factor(chas))
```

Ans. The Boston dataset contains information collected by the U.S Census Service concerning housing in the area of Boston Mass. It has 506 case records, one for each of the census tracts in the Boston area, with 14 attributes per case.

The dataset has the following variables- crim- Per capita crime rate per town zn- proportion of residential land zoned for lots over 25,000 sq.ft indus- proportion of non-retail business acres per town chas- Charles River dummy variable (1 if tract bounds river; 0 otherwise) nox- nitric oxides concentration (parts per 10 million) rm- average number of rooms per dwelling age- proportion of owner-occupied units built prior to 1940 dis- weighted distances to five Boston employment centres rad- index of accessibility to radial highways tax- full-value property-tax rate per $10,000 ptratio- pupil-teacher ratio by town black- 1000(Bk - 0.63)ˆ2 where Bk is the proportion of blacks by town lstat- lower status of the population (percent). medv- Median value of owner-occupied homes in $1000's

Source: Harrison, D. and Rubinfeld, D.L. (1978) Hedonic prices and the demand for clean air. J. Environ. Economics and Management 5, 81-102.

Belsley D.A., Kuh, E. and Welsch, R.E. (1980) Regression Diagnostics. Identifying Influential Data and Sources of Collinearity. New York: Wiley.

1.2 Variable of interest

Consider this data in context, what is the response variable of interest? Discuss how you think some of the possible predictor variables might be associated with this response.

Ans. The response variable of interest here is the median value of owner occupied homes (medv) Several of the variables in the dataset could be possible predictor variables which might be associated with the median house price. The per capita crime rate could be one possible predictor, more the crime rate, lesser the median price. The proportion of residential and business acres could be two other predictors. More business acres could indicate higher development, leading to higher house prices. Proximity to the charles river could be another predictor, houses in proximity to Charles might have higher median prices. Nox concentration might indicate pollution levels, higher nox concentration could lead tolower prices. The number of rooms in the dwelling would very likely influence the median prices of the homes. More the rooms, greater the price. The proximity to employment centres and radial highways might also drive up the home prices. High property tax rates might also drive up the median prices. pupil teacher ratio could be a predictor, but it is not very certain as parents generally prefer lower pupil teacher ratios, but major cities with higher house prices generally have big schools with high pupil teacher ratio. The number of blacks might influence prices but it is not certain The percent lower status of the population might very likely influence median prices. Prices with a higher proportion of lower status people might be lower. The proportion of owner occupied units might influence the price.

1.3 Simple Regression

For each predictor, fit a simple linear regression model to predict the response. In which of the models is there a statistically significant association between the predictor and the response? Create some plots to back up your assertions.

Ans. The data indicates that each of the predictor variables is statistically significant while attempting to determine an association with the response variables. The p values for each of the 13 predictors is extremely small except for the charles river dummy variable which also has a small p value.

Although each of the variables are statistically significant, the small adjusted R squared value for all of the models indicates that each of these variables individually only describe a small variation in the response, so a model with multiple predictors might work better.

```r
library(tidyverse)
col_lst <- colnames(Boston)
#get a list of all columns except median price
col_lst <- col_lst[1:length(col_lst)-1]
#function to create linear model with predictor as x for outcome medv
lin_model <- function(x)
  {
    pred_mod <- lm(medv ~ get(x),data=Boston)
    return(pred_mod)
}
#run function for each of the 13 predictors to get individual models and summary of models
prediction_model <- lapply(col_lst,lin_model)
lapply(prediction_model,function(x){summary(x)})
```

```
## [[1]]
##
## Call:
## lm(formula = medv ~ get(x), data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -16.957  -5.449  -2.007   2.512  29.800
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 24.03311    0.40914   58.74   <2e-16 ***
## get(x)      -0.41519    0.04389   -9.46   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.484 on 504 degrees of freedom
## Multiple R-squared:  0.1508, Adjusted R-squared:  0.1491
## F-statistic: 89.49 on 1 and 504 DF,  p-value: < 2.2e-16
##
##
## [[2]]
##
## Call:
## lm(formula = medv ~ get(x), data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.918  -5.518  -1.006   2.757  29.082
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 20.91758    0.42474  49.248   <2e-16 ***
## get(x)       0.14214    0.01638   8.675   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.587 on 504 degrees of freedom
## Multiple R-squared:  0.1299, Adjusted R-squared:  0.1282
## F-statistic: 75.26 on 1 and 504 DF,  p-value: < 2.2e-16
```

```
##
##
## [[3]]
##
## Call:
## lm(formula = medv ~ get(x), data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -13.017  -4.917  -1.457   3.180  32.943
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 29.75490    0.68345   43.54   <2e-16 ***
## get(x)      -0.64849    0.05226  -12.41   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.057 on 504 degrees of freedom
## Multiple R-squared:  0.234,  Adjusted R-squared:  0.2325
## F-statistic:   154 on 1 and 504 DF,  p-value: < 2.2e-16
##
##
## [[4]]
##
## Call:
## lm(formula = medv ~ get(x), data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -17.094  -5.894  -1.417   2.856  27.906
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  22.0938     0.4176  52.902  < 2e-16 ***
## get(x)1       6.3462     1.5880   3.996 7.39e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.064 on 504 degrees of freedom
## Multiple R-squared:  0.03072,    Adjusted R-squared:  0.02879
## F-statistic: 15.97 on 1 and 504 DF,  p-value: 7.391e-05
##
##
## [[5]]
##
## Call:
## lm(formula = medv ~ get(x), data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -13.691  -5.121  -2.161   2.959  31.310
##
## Coefficients:
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   41.346      1.811   22.83   <2e-16 ***
## get(x)       -33.916      3.196  -10.61   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.323 on 504 degrees of freedom
## Multiple R-squared:  0.1826, Adjusted R-squared:  0.181
## F-statistic: 112.6 on 1 and 504 DF,  p-value: < 2.2e-16
##
##
## [[6]]
##
## Call:
## lm(formula = medv ~ get(x), data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -23.346  -2.547   0.090   2.986  39.433
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -34.671      2.650  -13.08   <2e-16 ***
## get(x)         9.102      0.419   21.72   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.616 on 504 degrees of freedom
## Multiple R-squared:  0.4835, Adjusted R-squared:  0.4825
## F-statistic: 471.8 on 1 and 504 DF,  p-value: < 2.2e-16
##
##
## [[7]]
##
## Call:
## lm(formula = medv ~ get(x), data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.097  -5.138  -1.958   2.397  31.338
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 30.97868    0.99911  31.006   <2e-16 ***
## get(x)      -0.12316    0.01348  -9.137   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.527 on 504 degrees of freedom
## Multiple R-squared:  0.1421, Adjusted R-squared:  0.1404
## F-statistic: 83.48 on 1 and 504 DF,  p-value: < 2.2e-16
##
##
## [[8]]
```

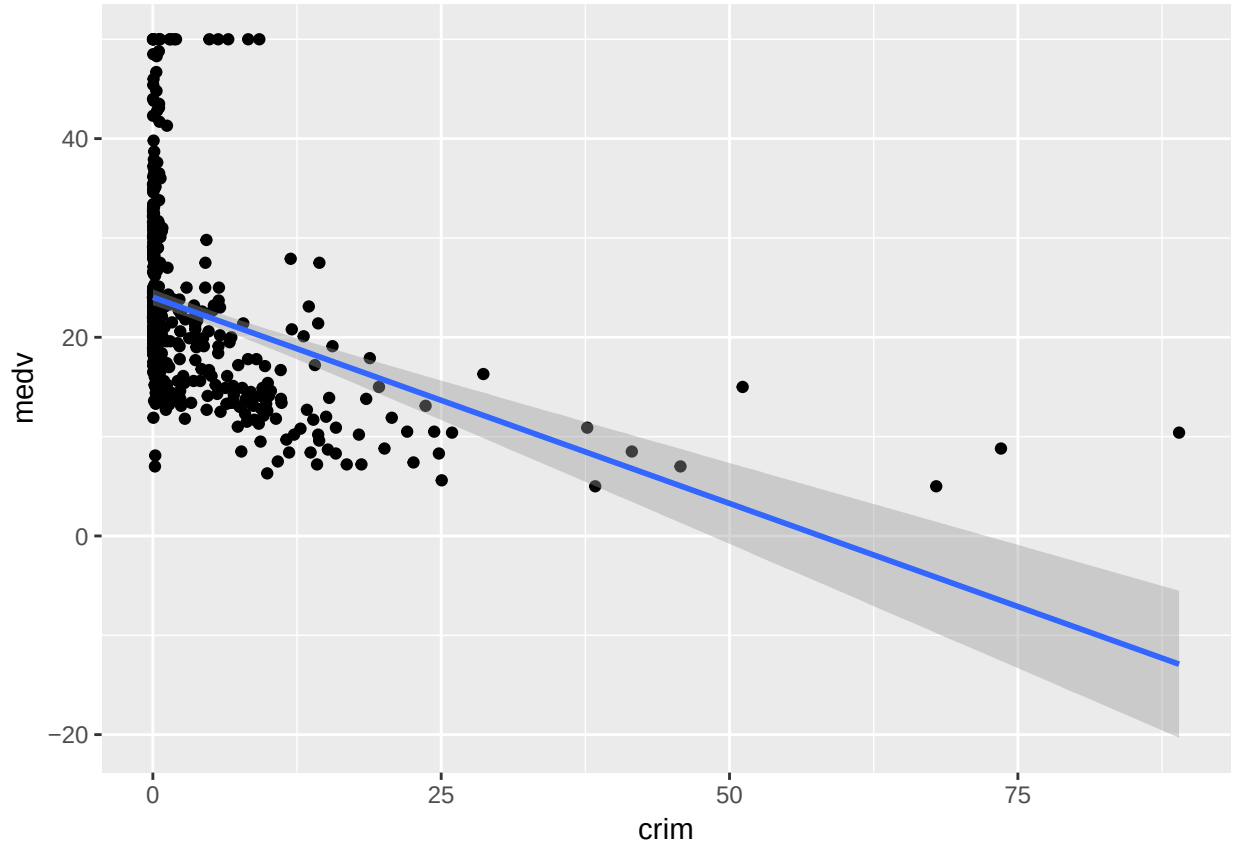
```
##
## Call:
## lm(formula = medv ~ get(x), data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.016  -5.556  -1.865   2.288  30.377
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  18.3901     0.8174  22.499  < 2e-16 ***
## get(x)        1.0916     0.1884   5.795 1.21e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.914 on 504 degrees of freedom
## Multiple R-squared:  0.06246,    Adjusted R-squared:  0.0606
## F-statistic: 33.58 on 1 and 504 DF,  p-value: 1.207e-08
##
##
## [[9]]
##
## Call:
## lm(formula = medv ~ get(x), data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -17.770  -5.199  -1.967   3.321  33.292
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 26.38213    0.56176  46.964   <2e-16 ***
## get(x)      -0.40310    0.04349  -9.269   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.509 on 504 degrees of freedom
## Multiple R-squared:  0.1456, Adjusted R-squared:  0.1439
## F-statistic: 85.91 on 1 and 504 DF,  p-value: < 2.2e-16
##
##
## [[10]]
##
## Call:
## lm(formula = medv ~ get(x), data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -14.091  -5.173  -2.085   3.158  34.058
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 32.970654   0.948296   34.77   <2e-16 ***
## get(x)      -0.025568   0.002147  -11.91   <2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.133 on 504 degrees of freedom
## Multiple R-squared:  0.2195, Adjusted R-squared:  0.218
## F-statistic: 141.8 on 1 and 504 DF,  p-value: < 2.2e-16
##
##
## [[11]]
##
## Call:
## lm(formula = medv ~ get(x), data = Boston)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -18.8342  -4.8262  -0.6426   3.1571  31.2303
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   62.345      3.029   20.58   <2e-16 ***
## get(x)        -2.157      0.163  -13.23   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.931 on 504 degrees of freedom
## Multiple R-squared:  0.2578, Adjusted R-squared:  0.2564
## F-statistic: 175.1 on 1 and 504 DF,  p-value: < 2.2e-16
##
##
## [[12]]
##
## Call:
## lm(formula = medv ~ get(x), data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -18.884  -4.862  -1.684   2.932  27.763
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 10.551034   1.557463   6.775 3.49e-11 ***
## get(x)       0.033593   0.004231   7.941 1.32e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.679 on 504 degrees of freedom
## Multiple R-squared:  0.1112, Adjusted R-squared:  0.1094
## F-statistic: 63.05 on 1 and 504 DF,  p-value: 1.318e-14
##
##
## [[13]]
##
## Call:
## lm(formula = medv ~ get(x), data = Boston)
```

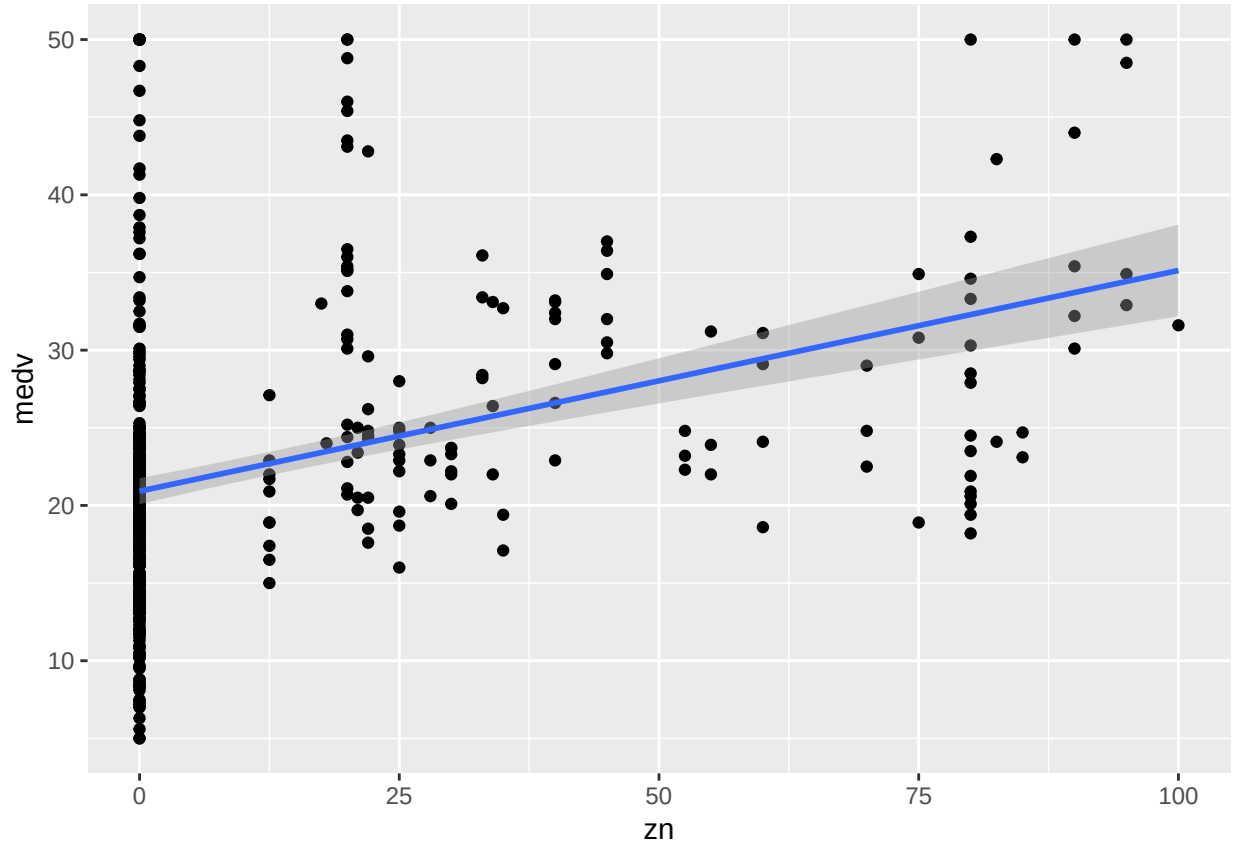
```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.168  -3.990  -1.318   2.034  24.500
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 34.55384    0.56263   61.41   <2e-16 ***
## get(x)      -0.95005    0.03873  -24.53   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.216 on 504 degrees of freedom
## Multiple R-squared:  0.5441, Adjusted R-squared:  0.5432
## F-statistic: 601.6 on 1 and 504 DF,  p-value: < 2.2e-16
```

```
#function to create plot of individual predictor against outcome
plotfunc <- function(i)
{ggplot(data=Boston,aes_string(x=colnames(Boston[i]),y=colnames(Boston[14])))+
    geom_point()+geom_smooth(method='lm')}
lapply(seq(col_lst),plotfunc)
```

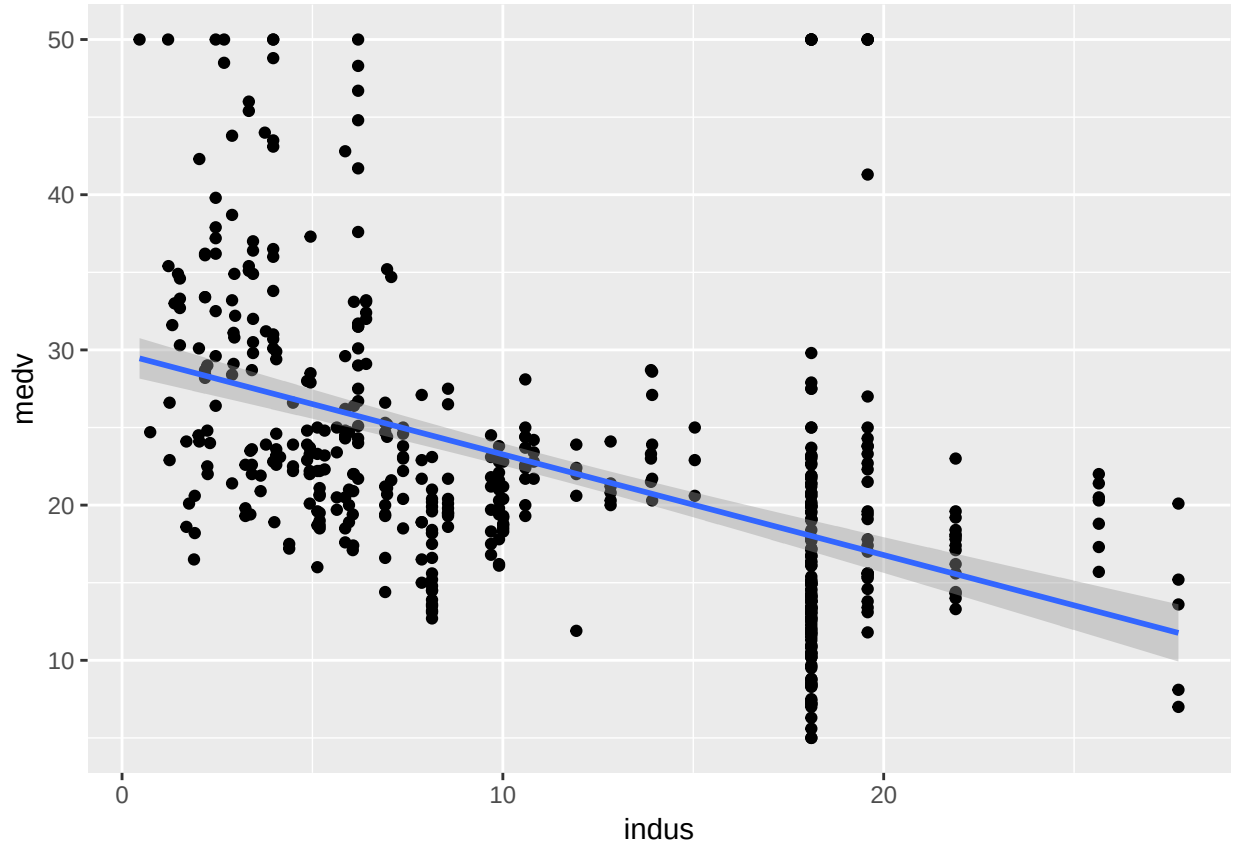
```
## [[1]]
```



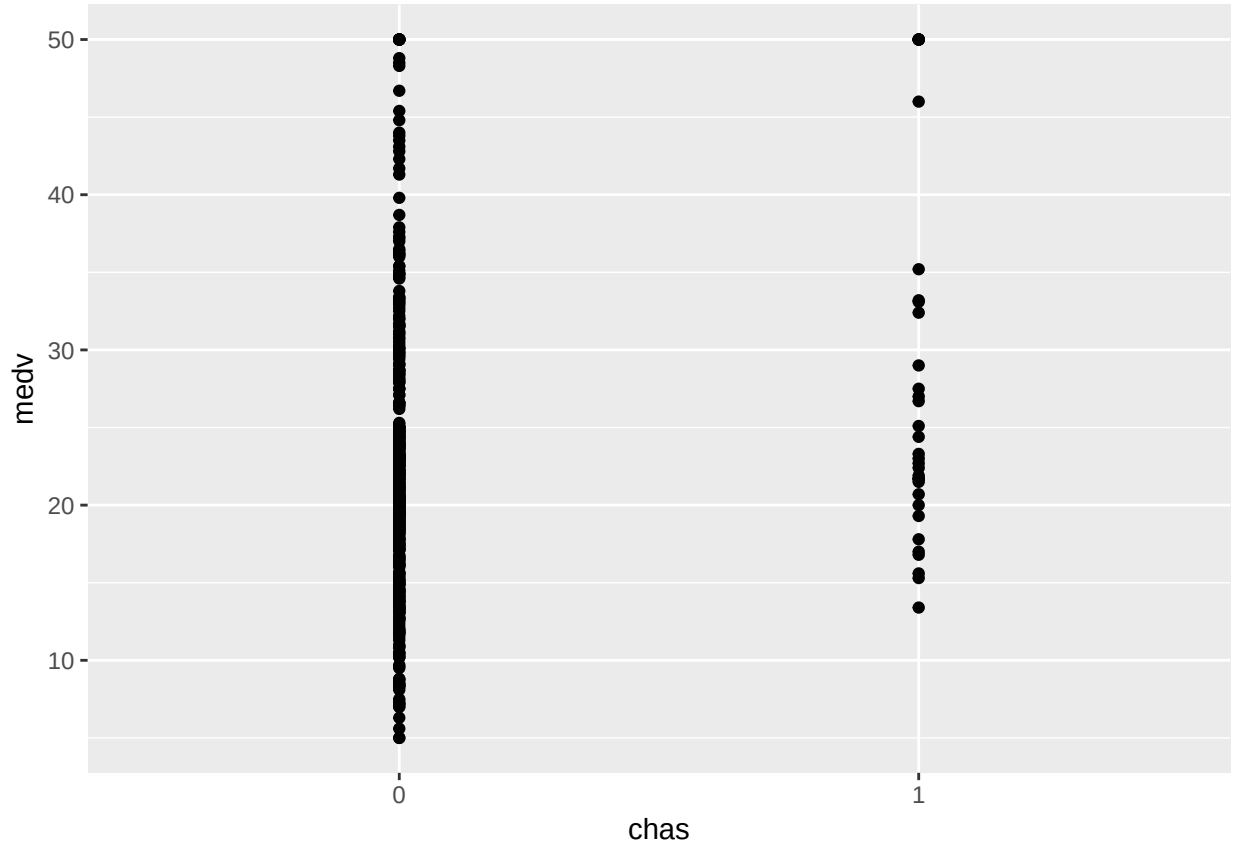

```
##
## [[2]]
```



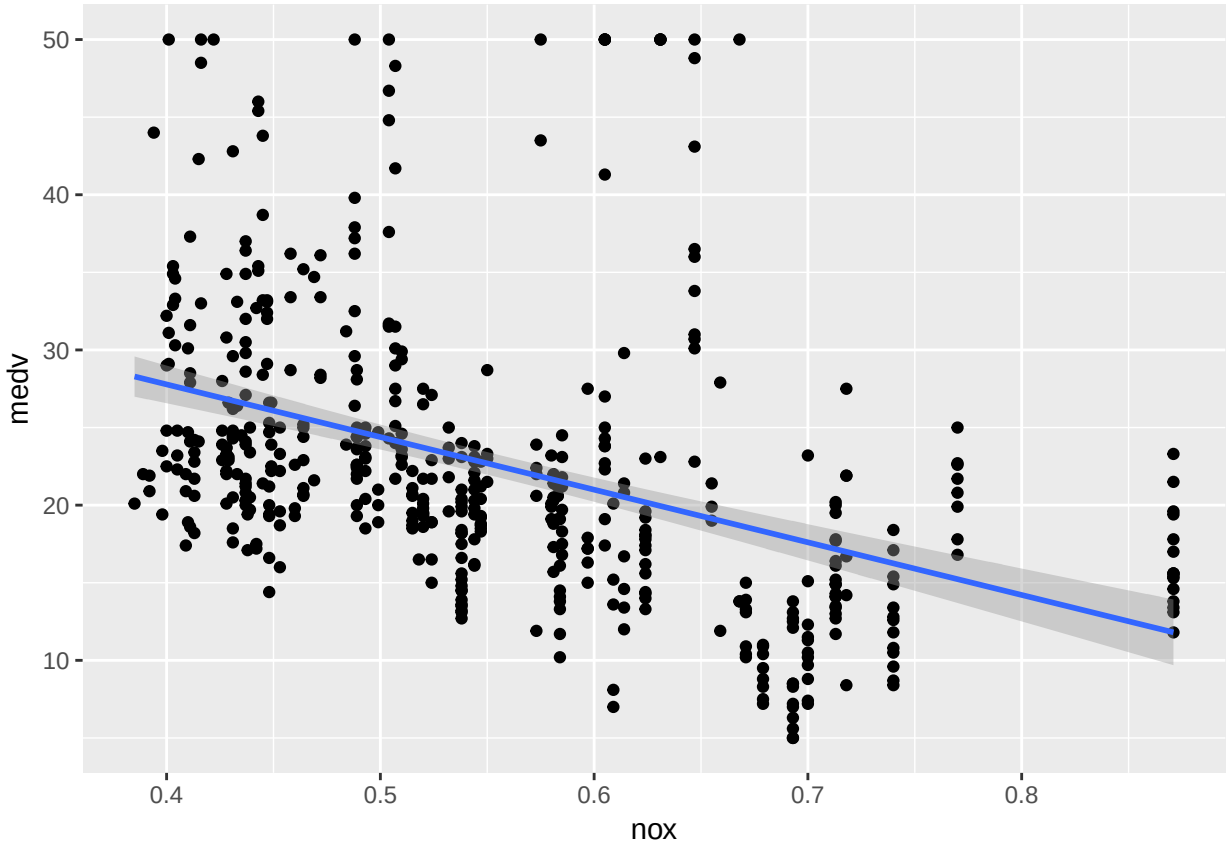

```
##
## [[3]]
```



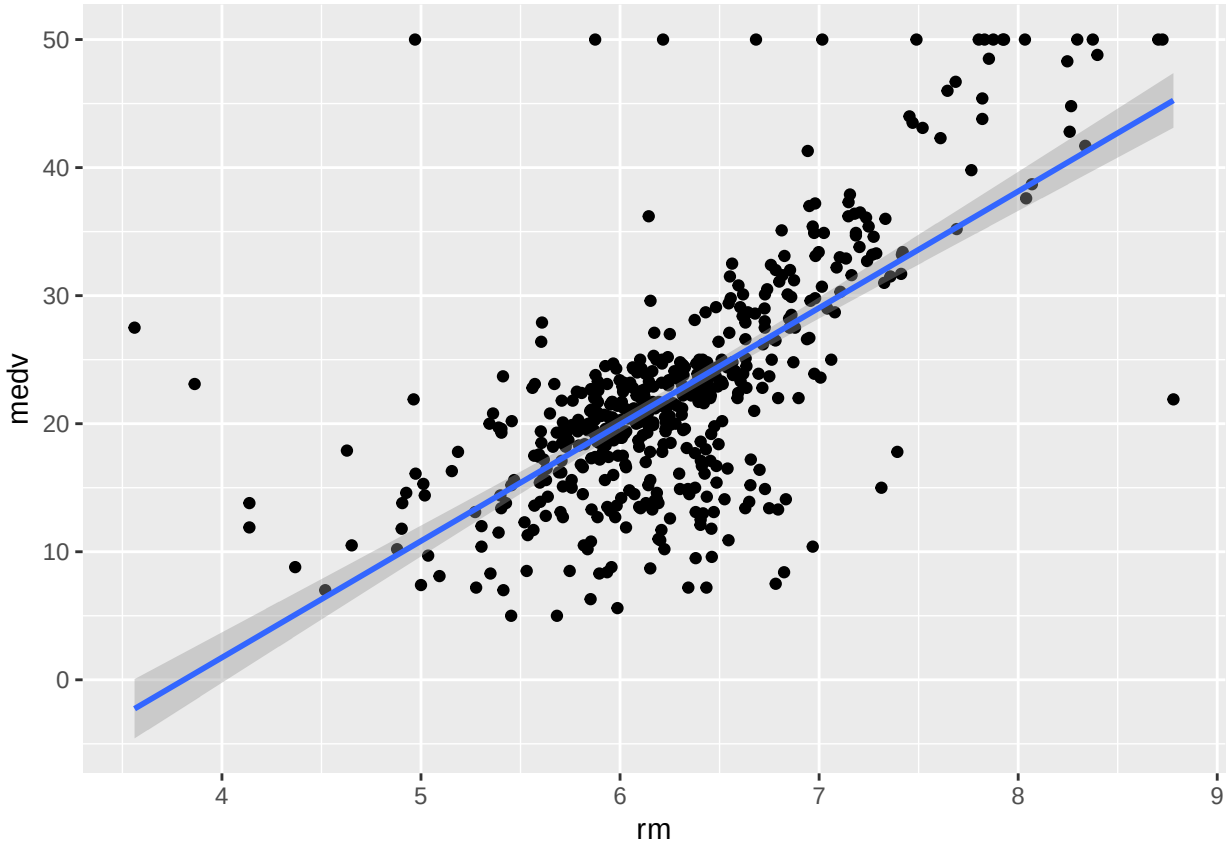

```
##
## [[4]]
```



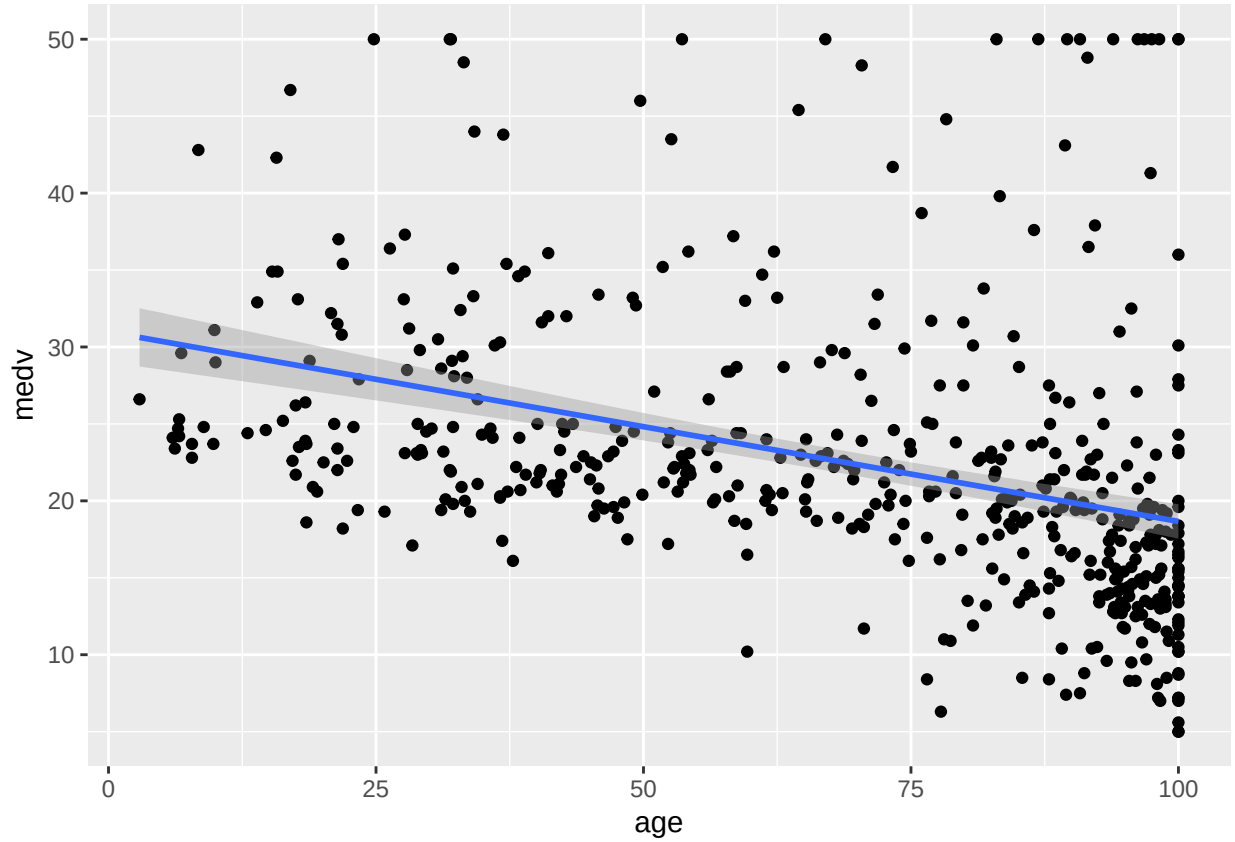

```
##
## [[5]]
```



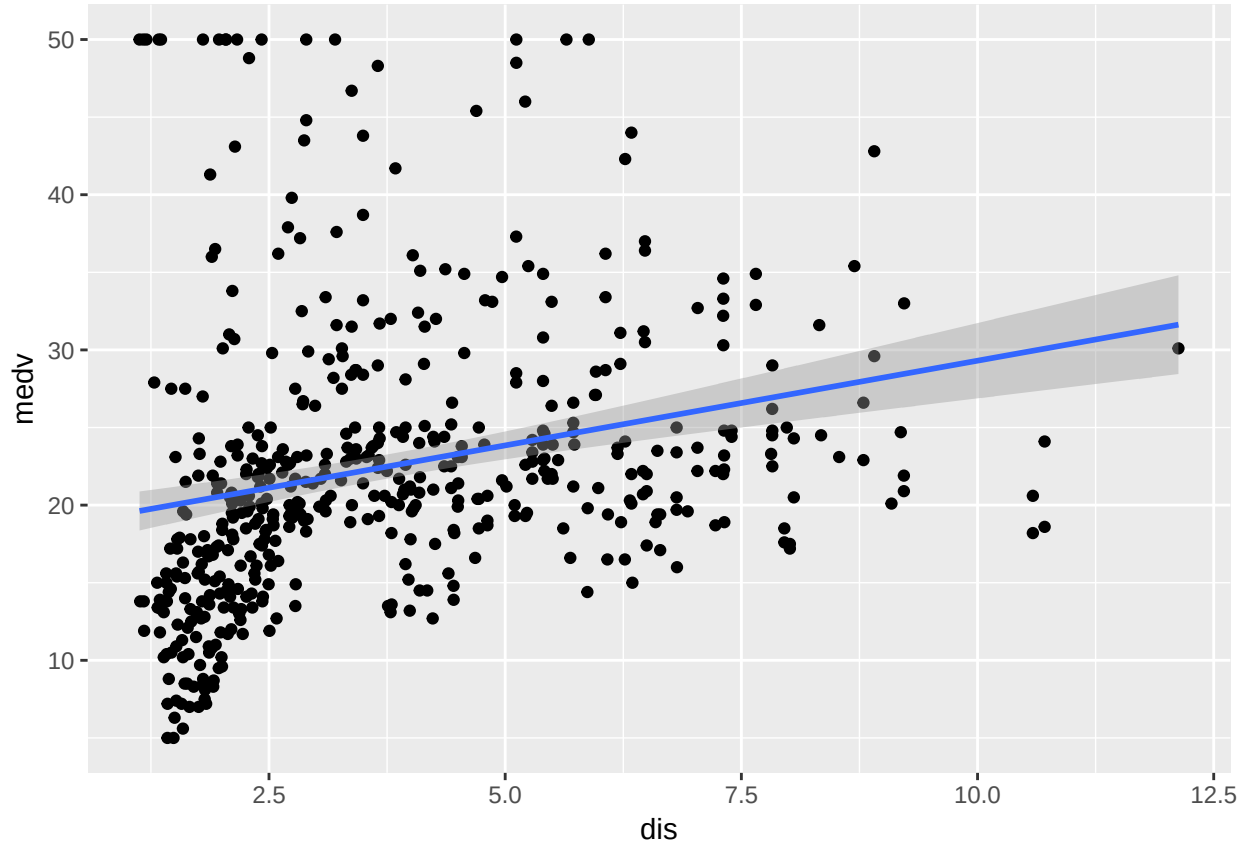

```
##
## [[6]]
```



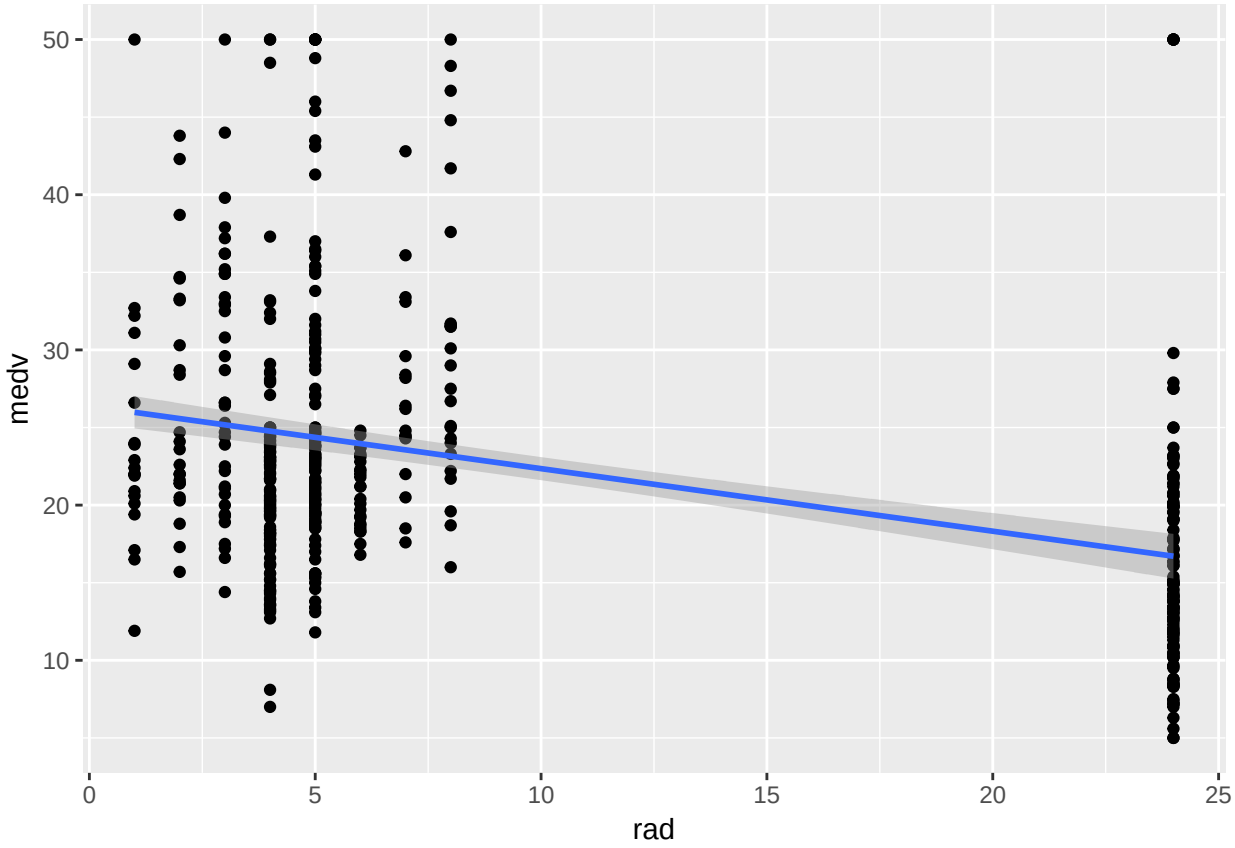

```
##
## [[7]]
```



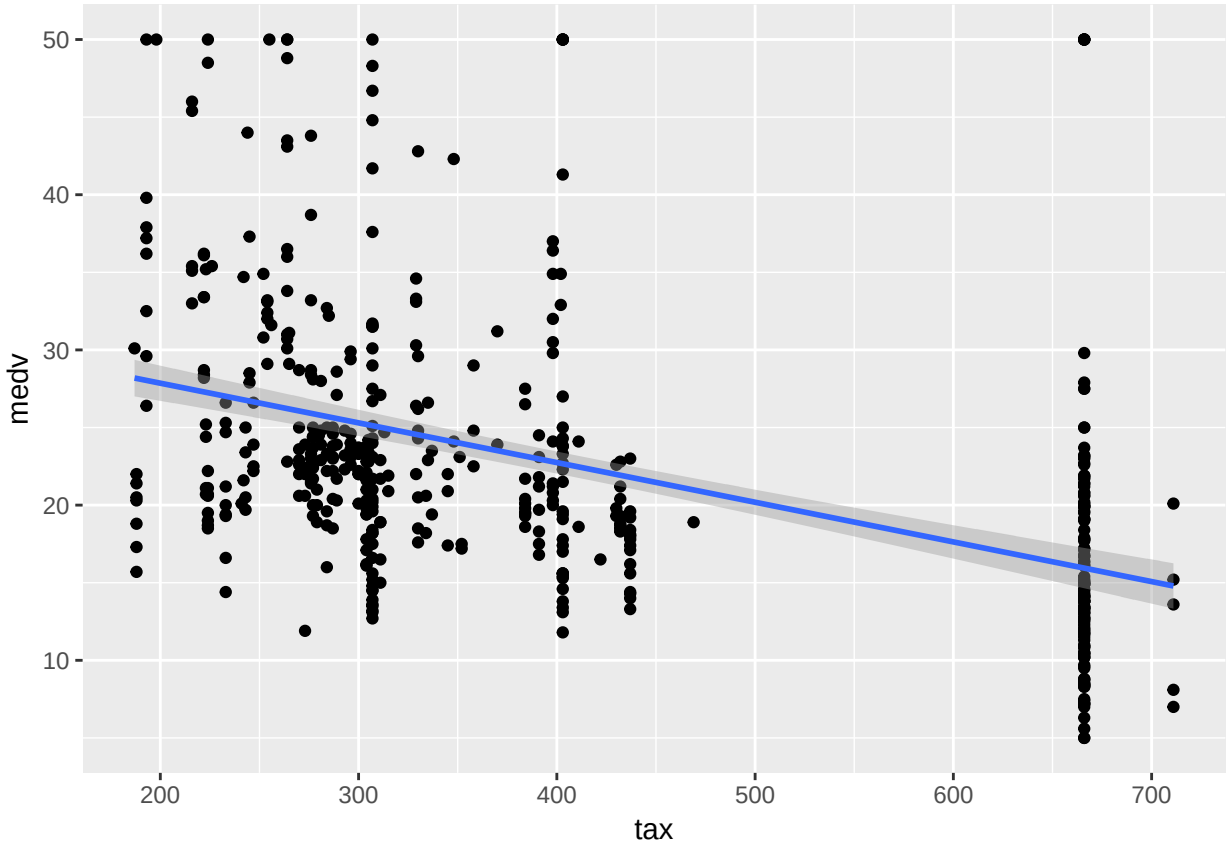

```
##
## [[8]]
```



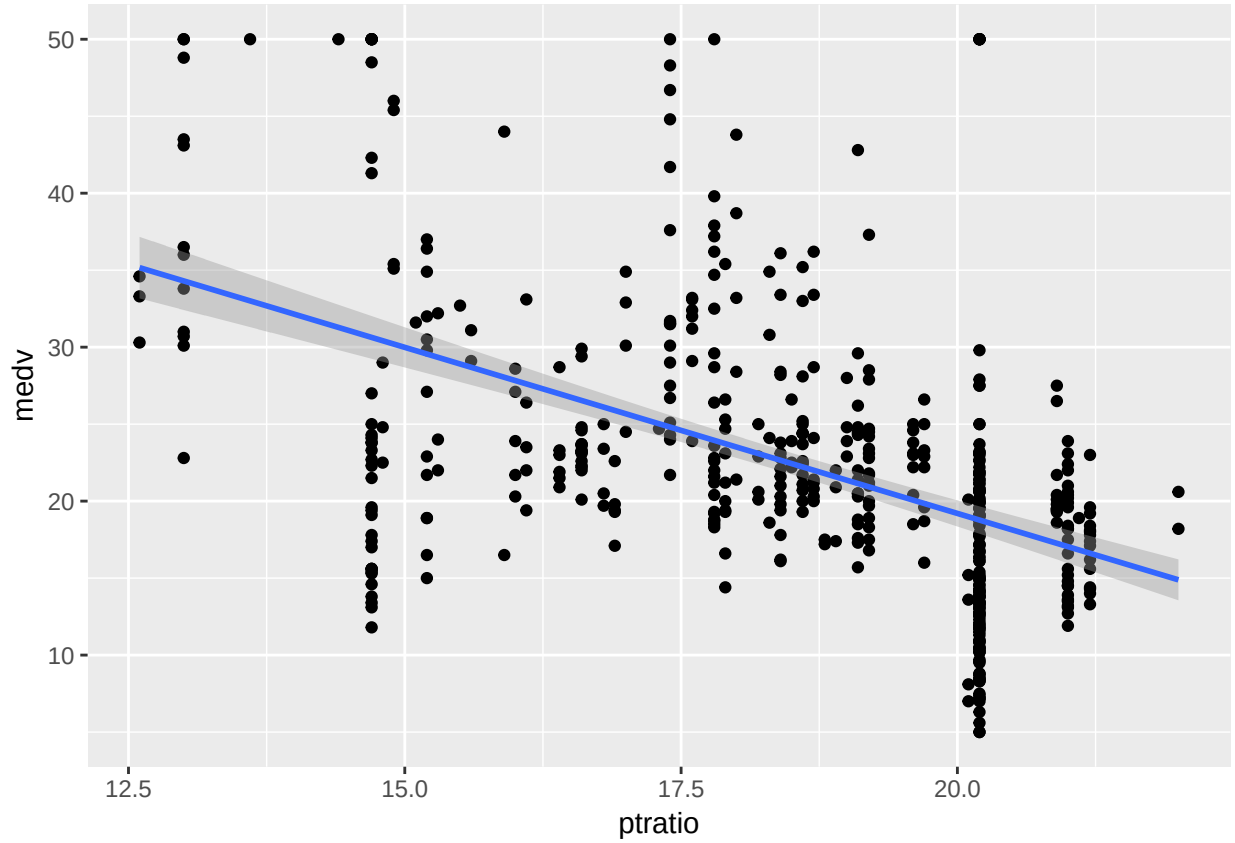

```
##
## [[9]]
```



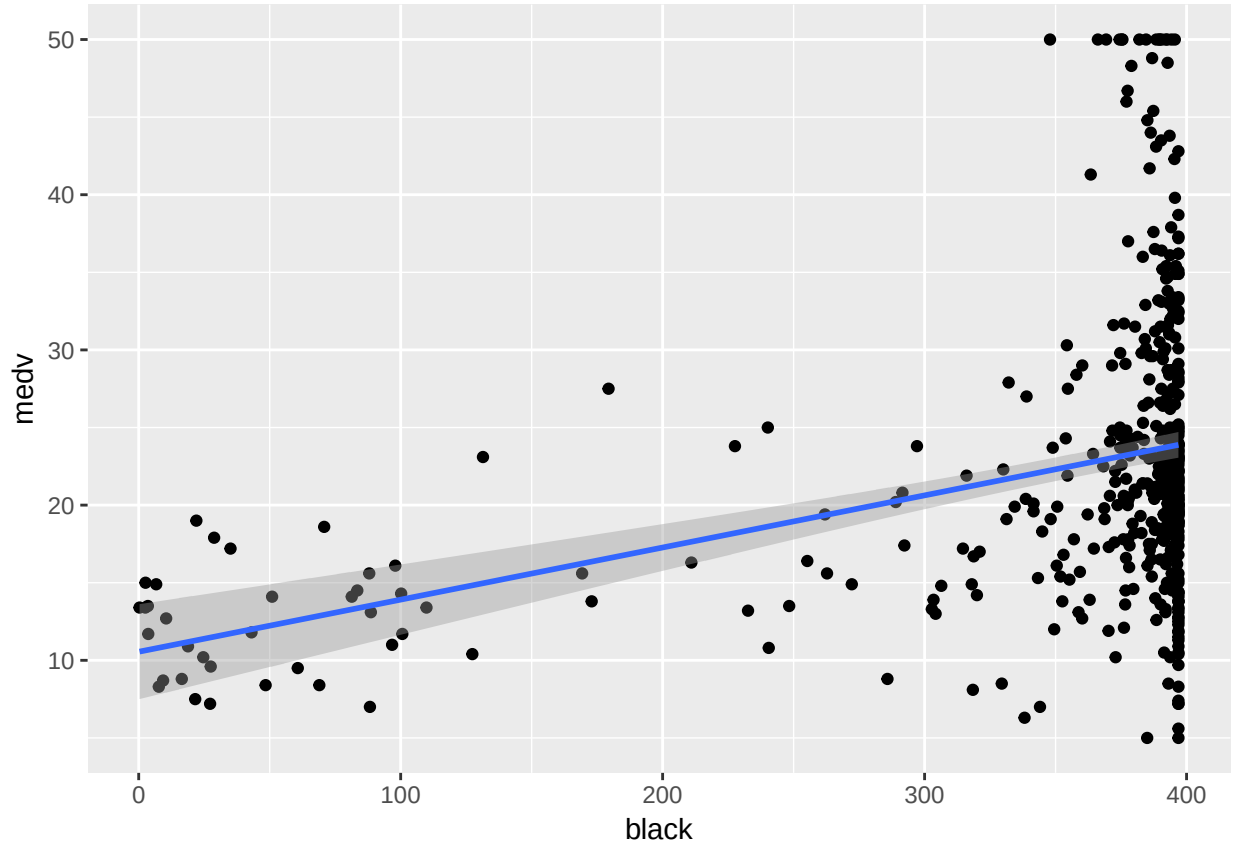

```
##
## [[10]]
```



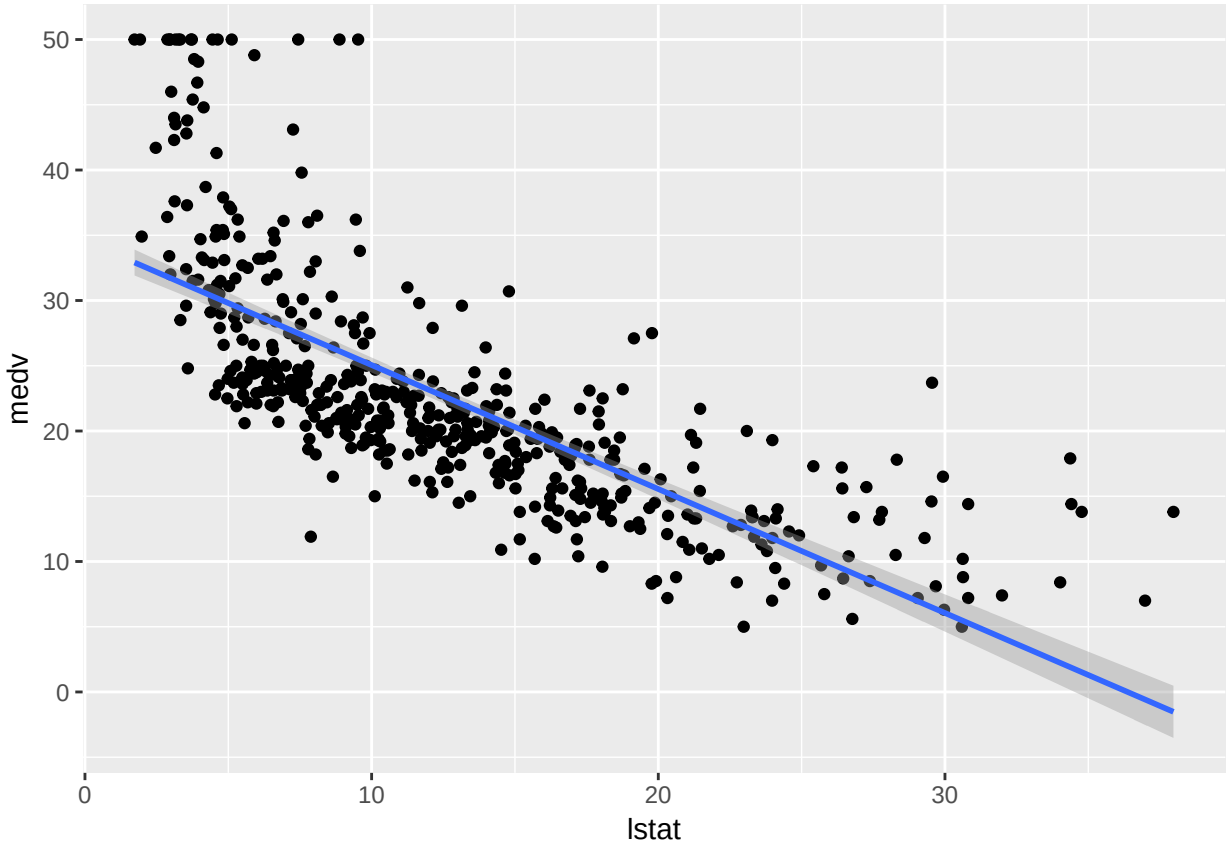

```
##
## [[11]]
```




```
##
## [[12]]
```




```
##
## [[13]]
```



1.4 Multiple Regression

Make sure you are familiar with multiple regression (Openintro Statistics, Ch 8.1-8.3).

Fit a multiple regression model to predict the response using all of the predictors. Describe your results. For which predictors can we reject the null hypothesis $H_0 : \beta_j = 0$?

Ans. When we use all the predictors, the resulting model performs much better than each of the models where the response variable is associated with a single predictor variable. This is indicated by a high adjusted R squared value of 0.7338 which indicates the strength of the fit. This means that there was a 73.38% reduction in the variation of the data by using the additional variables in the model. The residual standard error is 4.745 on 492 degrees of freedom. The data indicates that the variables crime rate, nox concentration, distance from employment centres, taxation, pupil teacher ratio, and lower status of popultation have a negative correlation with median house prices, while the rest of the variables have a positive correlation. The data suggests that for every unit increase in crime rate, the median house price goes down by 1.080e-01 (Standard error- 5.103e+00 and t-value- 7.144) . (Similar interpretations can be made of the coefficients of the other 12 predictors)

As all of the predictor p values fall below the critical p value of 0.05 for a 95% confidence interval, we can reject the null hypothesis for all of the above predictors except for indus and age, for which the p value falls above the critical p value of 0.05.

```
#create multiple regression model for all 13 predictors combined in a linear fashion
multi_model <- lm(data=Boston,medv ~ crim + zn + indus + chas + nox +
                    rm + age + dis + rad + tax + ptratio + black + lstat)
summary(multi_model)
```

```
##
## Call:
## lm(formula = medv ~ crim + zn + indus + chas + nox + rm + age +
##     dis + rad + tax + ptratio + black + lstat, data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.595  -2.730  -0.518   1.777  26.199
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.646e+01  5.103e+00   7.144 3.28e-12 ***
## crim        -1.080e-01  3.286e-02  -3.287 0.001087 **
## zn           4.642e-02  1.373e-02   3.382 0.000778 ***
## indus        2.056e-02  6.150e-02   0.334 0.738288
## chas1        2.687e+00  8.616e-01   3.118 0.001925 **
## nox         -1.777e+01  3.820e+00  -4.651 4.25e-06 ***
## rm           3.810e+00  4.179e-01   9.116  < 2e-16 ***
## age          6.922e-04  1.321e-02   0.052 0.958229
## dis         -1.476e+00  1.995e-01  -7.398 6.01e-13 ***
## rad          3.060e-01  6.635e-02   4.613 5.07e-06 ***
## tax         -1.233e-02  3.760e-03  -3.280 0.001112 **
## ptratio     -9.527e-01  1.308e-01  -7.283 1.31e-12 ***
## black        9.312e-03  2.686e-03   3.467 0.000573 ***
## lstat       -5.248e-01  5.072e-02 -10.347  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.745 on 492 degrees of freedom
## Multiple R-squared:  0.7406, Adjusted R-squared:  0.7338
## F-statistic: 108.1 on 13 and 492 DF,  p-value: < 2.2e-16
```

1.5 Compare Regressions

How do your results from (3) compare to your results from (4)? Create a plot displaying the univariate regression coefficients from (3) on the x-axis and the multiple regression coefficients from part (4) on the y-axis. Use this visualization to support your response.

Ans. When we use all the predictors, the resulting model performs much better than each of the models where the response variable is associated with a single predictor variable. This is indicated by a high adjusted R squared value of 0.7338 which indicates the strength of the fit. This means that there was a 73.38% reduction in the variation of the data by using the additional variables in the model. Also, while the univariate regression model suggests that all predictors might be statistically significant, the p values of the multi regression model suggest that indus and age are not statistically significant as their p values are above 0.05.

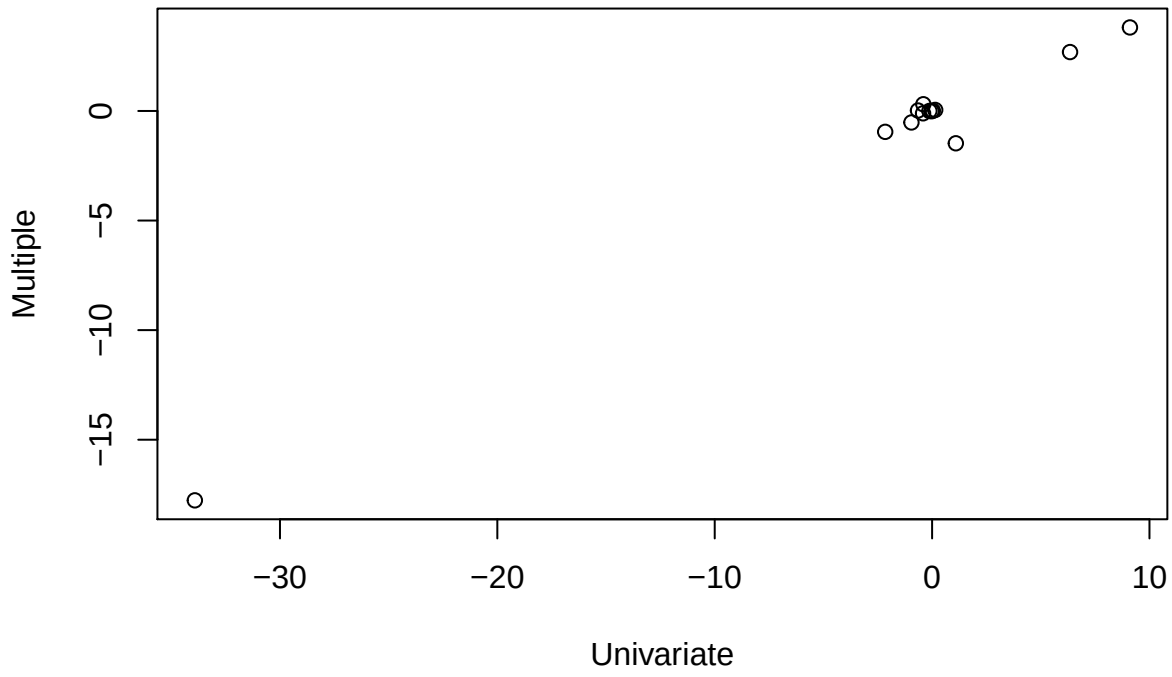
```
# create a plot of multiple regression coefficients against univariate coefficients
multi_model$coefficients[2:14]
```

```
##          crim            zn         indus         chas1           nox
## -1.080114e-01  4.642046e-02  2.055863e-02  2.686734e+00 -1.776661e+01
##            rm           age           dis           rad           tax
##  3.809865e+00  6.922246e-04 -1.475567e+00  3.060495e-01 -1.233459e-02
##       ptratio         black         lstat
## -9.527472e-01  9.311683e-03 -5.247584e-01
```

```
ind_coeff <- list()
for(i in prediction_model)
{
  ind_coeff <- append(ind_coeff,i$coefficients[2])
}
ind_coeff <- unlist(ind_coeff)
print(ind_coeff)
```

```
##       get(x)       get(x)       get(x)      get(x)1       get(x)
##  -0.41519028   0.14213999  -0.64849005   6.34615711 -33.91605501
##       get(x)       get(x)       get(x)       get(x)       get(x)
##   9.10210898  -0.12316272   1.09161302  -0.40309540  -0.02556810
##       get(x)       get(x)       get(x)
##  -2.15717530   0.03359306  -0.95004935
```

```
plot(ind_coeff, multi_model$coefficients[2:14], xlab = 'Univariate', ylab = 'Multiple')
```



1.6 Non-linearities

Is there evidence of a non-linear association between any of the predictors and the response? To answer this question, for each predictor X fit a model of the form:

$$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \epsilon$$

Ans.

Approach 1: Using p values of coefficients of squared anc cubed terms. Checking which p values of squared or cubed terms in all the models is lower than the critical p value of 0.05, the predictors - crim, zn, indus, nox, rm, dis, rad, lstat seem to have a non linear relationship with median house prices (medv).

Approach 2: Using metric R square The data seems to present evidence that there might be a strong non linear relationship between the variables rm and medv, and lstat and medv. This is indicated by moderately high R squared values (Good model fit) when non linear models are fit on these 2 variables. The non linear model fit using average number of rooms (rm) has an R squared value of 0.5586 (Adjusted) and the linear model fit using the lower status proportion variable (lstat) has an R squared value of 0.6558 (Adjusted).

```
# create univariate non linear model for each of the predictors except chas
for(i in col_lst)
{
  if(i!='chas')
  {
  non_linear_mod <- lm(data=Boston,medv ~ I(get(i)) + I((get(i))^2) + I((get(i))^3))
  print(i)
  print(summary(non_linear_mod))
  }

}
```

```
## [1] "crim"
##
## Call:
## lm(formula = medv ~ I(get(i)) + I((get(i))^2) + I((get(i))^3),
##     data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -17.983  -4.975  -1.940   2.881  33.391
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    2.519e+01  4.355e-01  57.846  < 2e-16 ***
## I(get(i))     -1.136e+00  1.444e-01  -7.868 2.24e-14 ***
## I((get(i))^2)  2.378e-02  6.808e-03   3.494 0.000518 ***
## I((get(i))^3) -1.489e-04  6.641e-05  -2.242 0.025411 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.159 on 502 degrees of freedom
## Multiple R-squared:  0.2177, Adjusted R-squared:  0.213
## F-statistic: 46.57 on 3 and 502 DF,  p-value: < 2.2e-16
##
## [1] "zn"
##
## Call:
## lm(formula = medv ~ I(get(i)) + I((get(i))^2) + I((get(i))^3),
##     data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.449  -5.549  -1.049   3.225  29.551
##
```

```
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)   20.4485972  0.4359536  46.905  < 2e-16 ***
## I(get(i))      0.6433652  0.1105611   5.819 1.06e-08 ***
## I((get(i))^2) -0.0167646  0.0038872  -4.313 1.94e-05 ***
## I((get(i))^3)  0.0001257  0.0000316   3.978 7.98e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.43 on 502 degrees of freedom
## Multiple R-squared:  0.1649, Adjusted R-squared:  0.1599
## F-statistic: 33.05 on 3 and 502 DF,  p-value: < 2.2e-16
##
## [1] "indus"
##
## Call:
## lm(formula = medv ~ I(get(i)) + I((get(i))^2) + I((get(i))^3),
##     data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.760  -4.725  -1.009   2.932  32.038
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   37.080160   1.663326  22.293  < 2e-16 ***
## I(get(i))     -2.806994   0.509349  -5.511 5.71e-08 ***
## I((get(i))^2)  0.140462   0.041554   3.380 0.000781 ***
## I((get(i))^3) -0.002399   0.001011  -2.373 0.018026 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.844 on 502 degrees of freedom
## Multiple R-squared:  0.2768, Adjusted R-squared:  0.2725
## F-statistic: 64.06 on 3 and 502 DF,  p-value: < 2.2e-16
##
## [1] "nox"
##
## Call:
## lm(formula = medv ~ I(get(i)) + I((get(i))^2) + I((get(i))^3),
##     data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -13.104  -5.020  -2.144   2.747  32.416
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -22.49      38.52  -0.584   0.5596
## I(get(i))       315.10     195.10   1.615   0.1069
## I((get(i))^2)  -615.83     320.48  -1.922   0.0552 .
## I((get(i))^3)   350.19     170.92   2.049   0.0410 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 8.282 on 502 degrees of freedom
## Multiple R-squared:  0.1939, Adjusted R-squared:  0.189
## F-statistic: 40.24 on 3 and 502 DF,  p-value: < 2.2e-16
##
## [1] "rm"
##
## Call:
## lm(formula = medv ~ I(get(i)) + I((get(i))^2) + I((get(i))^3),
##     data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -29.102  -2.674   0.569   3.011  35.911
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    241.3108    47.3275   5.099 4.85e-07 ***
## I(get(i))     -109.3906    22.9690  -4.763 2.51e-06 ***
## I((get(i))^2)   16.4910     3.6750   4.487 8.95e-06 ***
## I((get(i))^3)   -0.7404     0.1935  -3.827 0.000146 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.11 on 502 degrees of freedom
## Multiple R-squared:  0.5612, Adjusted R-squared:  0.5586
## F-statistic:   214 on 3 and 502 DF,  p-value: < 2.2e-16
##
## [1] "age"
##
## Call:
## lm(formula = medv ~ I(get(i)) + I((get(i))^2) + I((get(i))^3),
##     data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -16.443  -4.909  -2.234   2.185  32.944
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    2.893e+01  2.992e+00   9.668   <2e-16 ***
## I(get(i))     -1.224e-01  2.014e-01  -0.608    0.544
## I((get(i))^2)  2.355e-03  3.930e-03   0.599    0.549
## I((get(i))^3) -2.318e-05  2.279e-05  -1.017    0.310
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.472 on 502 degrees of freedom
## Multiple R-squared:  0.1566, Adjusted R-squared:  0.1515
## F-statistic: 31.06 on 3 and 502 DF,  p-value: < 2.2e-16
##
## [1] "dis"
##
## Call:
```

```
## lm(formula = medv ~ I(get(i)) + I((get(i))^2) + I((get(i))^3),
##     data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -12.571  -5.242  -2.037   2.397  34.769
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    7.03789    2.91134   2.417  0.01599 *
## I(get(i))      8.59284    2.06633   4.158 3.77e-05 ***
## I((get(i))^2) -1.24953    0.41235  -3.030  0.00257 **
## I((get(i))^3)  0.05602    0.02428   2.307  0.02146 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.727 on 502 degrees of freedom
## Multiple R-squared:  0.105,  Adjusted R-squared:  0.09968
## F-statistic: 19.64 on 3 and 502 DF,  p-value: 4.736e-12
##
## [1] "rad"
##
## Call:
## lm(formula = medv ~ I(get(i)) + I((get(i))^2) + I((get(i))^3),
##     data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -16.630  -5.151  -2.017   3.169  33.594
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   30.251303   2.567860  11.781  < 2e-16 ***
## I(get(i))     -3.799454   1.307156  -2.907 0.003815 **
## I((get(i))^2)  0.616347   0.186057   3.313 0.000991 ***
## I((get(i))^3) -0.020086   0.005717  -3.514 0.000482 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.37 on 502 degrees of freedom
## Multiple R-squared:  0.1767, Adjusted R-squared:  0.1718
## F-statistic: 35.91 on 3 and 502 DF,  p-value: < 2.2e-16
##
## [1] "tax"
##
## Call:
## lm(formula = medv ~ I(get(i)) + I((get(i))^2) + I((get(i))^3),
##     data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.109  -4.952  -1.878   2.957  33.694
##
## Coefficients:
```

```
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    5.222e+01  1.397e+01   3.739 0.000206 ***
## I(get(i))     -1.635e-01  1.133e-01  -1.443 0.149646
## I((get(i))^2)  3.029e-04  2.872e-04   1.055 0.292004
## I((get(i))^3) -2.079e-07  2.236e-07  -0.930 0.353061
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.115 on 502 degrees of freedom
## Multiple R-squared:  0.2261, Adjusted R-squared:  0.2215
## F-statistic: 48.89 on 3 and 502 DF,  p-value: < 2.2e-16
##
## [1] "ptratio"
##
## Call:
## lm(formula = medv ~ I(get(i)) + I((get(i))^2) + I((get(i))^3),
##     data = Boston)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -17.7795  -5.0364  -0.9778   3.4766  31.1636
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   312.28642  152.48693   2.048   0.0411 *
## I(get(i))     -48.69114   26.88441  -1.811   0.0707 .
## I((get(i))^2)   2.83995    1.56413   1.816   0.0700 .
## I((get(i))^3)  -0.05686    0.03005  -1.892   0.0590 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.898 on 502 degrees of freedom
## Multiple R-squared:  0.2669, Adjusted R-squared:  0.2625
## F-statistic: 60.91 on 3 and 502 DF,  p-value: < 2.2e-16
##
## [1] "black"
##
## Call:
## lm(formula = medv ~ I(get(i)) + I((get(i))^2) + I((get(i))^3),
##     data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -19.005  -4.802  -1.613   2.852  28.051
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    1.260e+01  2.517e+00   5.006  7.7e-07 ***
## I(get(i))     -1.703e-02  6.150e-02  -0.277    0.782
## I((get(i))^2)  2.036e-04  3.258e-04   0.625    0.532
## I((get(i))^3) -2.224e-07  4.765e-07  -0.467    0.641
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 8.685 on 502 degrees of freedom
## Multiple R-squared:  0.1135, Adjusted R-squared:  0.1082
## F-statistic: 21.43 on 3 and 502 DF,  p-value: 4.463e-13
##
## [1] "lstat"
##
## Call:
## lm(formula = medv ~ I(get(i)) + I((get(i))^2) + I((get(i))^3),
##     data = Boston)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.5441  -3.7122  -0.5145   2.4846  26.4153
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)   48.6496253  1.4347240  33.909  < 2e-16 ***
## I(get(i))     -3.8655928  0.3287861 -11.757  < 2e-16 ***
## I((get(i))^2)  0.1487385  0.0212987   6.983 9.18e-12 ***
## I((get(i))^3) -0.0020039  0.0003997  -5.013 7.43e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.396 on 502 degrees of freedom
## Multiple R-squared:  0.6578, Adjusted R-squared:  0.6558
## F-statistic: 321.7 on 3 and 502 DF,  p-value: < 2.2e-16
```

1.7 Stepwise Model Selection

Consider performing a stepwise model selection procedure to determine the best fit model (consult Openintro Statistics, 8.2.2). Discuss your results. How is this model different from the model in (4)?

Ans. This model eliminates the predictors age and indus from the multiple regression model. The resulting model seems to be more accurate as the adjusted R squared value of the model has gone up to 0.7348, indicating a better model fit.

I have used 2 methods, the stepAIC function and the forward selection method based on R squared values for this question. Both lead to the same model

1. StepAIC function method

```
#create model using backward elemination using the stepAIC function and the AIC metric
stepwise_model <- stepAIC(multi_model,direction = 'backward')
```

```
## Start:  AIC=1589.64
## medv ~ crim + zn + indus + chas + nox + rm + age + dis + rad +
##     tax + ptratio + black + lstat
##
##           Df Sum of Sq   RSS    AIC
## - age      1      0.06 11079 1587.7
## - indus    1      2.52 11081 1587.8
## <none>                 11079 1589.6
## - chas     1    218.97 11298 1597.5
## - tax      1    242.26 11321 1598.6
## - crim     1    243.22 11322 1598.6
## - zn       1    257.49 11336 1599.3
```

```
## - black    1    270.63 11349 1599.8
## - rad      1    479.15 11558 1609.1
## - nox      1    487.16 11566 1609.4
## - ptratio  1   1194.23 12273 1639.4
## - dis      1   1232.41 12311 1641.0
## - rm       1   1871.32 12950 1666.6
## - lstat    1   2410.84 13490 1687.3
##
## Step:  AIC=1587.65
## medv ~ crim + zn + indus + chas + nox + rm + dis + rad + tax +
##     ptratio + black + lstat
##
##           Df Sum of Sq   RSS    AIC
## - indus    1      2.52 11081 1585.8
## <none>                 11079 1587.7
## - chas     1    219.91 11299 1595.6
## - tax      1    242.24 11321 1596.6
## - crim     1    243.20 11322 1596.6
## - zn       1    260.32 11339 1597.4
## - black    1    272.26 11351 1597.9
## - rad      1    481.09 11560 1607.2
## - nox      1    520.87 11600 1608.9
## - ptratio  1   1200.23 12279 1637.7
## - dis      1   1352.26 12431 1643.9
## - rm       1   1959.55 13038 1668.0
## - lstat    1   2718.88 13798 1696.7
##
## Step:  AIC=1585.76
## medv ~ crim + zn + chas + nox + rm + dis + rad + tax + ptratio +
##     black + lstat
##
##           Df Sum of Sq   RSS    AIC
## <none>                 11081 1585.8
## - chas     1    227.21 11309 1594.0
## - crim     1    245.37 11327 1594.8
## - zn       1    257.82 11339 1595.4
## - black    1    270.82 11352 1596.0
## - tax      1    273.62 11355 1596.1
## - rad      1    500.92 11582 1606.1
## - nox      1    541.91 11623 1607.9
## - ptratio  1   1206.45 12288 1636.0
## - dis      1   1448.94 12530 1645.9
## - rm       1   1963.66 13045 1666.3
## - lstat    1   2723.48 13805 1695.0
```

```
print(summary(stepwise_model))
```

```
##
## Call:
## lm(formula = medv ~ crim + zn + chas + nox + rm + dis + rad +
##     tax + ptratio + black + lstat, data = Boston)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.5984  -2.7386  -0.5046   1.7273  26.2373
```

```
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  36.341145   5.067492   7.171 2.73e-12 ***
## crim         -0.108413   0.032779  -3.307 0.001010 **
## zn            0.045845   0.013523   3.390 0.000754 ***
## chas1         2.718716   0.854240   3.183 0.001551 **
## nox         -17.376023   3.535243  -4.915 1.21e-06 ***
## rm            3.801579   0.406316   9.356  < 2e-16 ***
## dis          -1.492711   0.185731  -8.037 6.84e-15 ***
## rad           0.299608   0.063402   4.726 3.00e-06 ***
## tax          -0.011778   0.003372  -3.493 0.000521 ***
## ptratio      -0.946525   0.129066  -7.334 9.24e-13 ***
## black         0.009291   0.002674   3.475 0.000557 ***
## lstat        -0.522553   0.047424 -11.019  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.736 on 494 degrees of freedom
## Multiple R-squared:  0.7406, Adjusted R-squared:  0.7348
## F-statistic: 128.2 on 11 and 494 DF,  p-value: < 2.2e-16
```

2. Forward selection method based on R squaed values

```
#create stepwise model using forward selection
#choose predictors which increase r squared value
for(i in col_lst)
{
  print(i)
  print(summary(lm(data=Boston,medv ~ get(i)))$r.squared)
}
```

```
## [1] "crim"
## [1] 0.1507805
## [1] "zn"
## [1] 0.1299208
## [1] "indus"
## [1] 0.23399
## [1] "chas"
## [1] 0.03071613
## [1] "nox"
## [1] 0.182603
## [1] "rm"
## [1] 0.4835255
## [1] "age"
## [1] 0.1420947
## [1] "dis"
## [1] 0.06246437
## [1] "rad"
## [1] 0.1456386
## [1] "tax"
## [1] 0.2195259
## [1] "ptratio"
## [1] 0.2578473
## [1] "black"
```

```
## [1] 0.1111961
## [1] "lstat"
## [1] 0.5441463
```

```
print('adding lstat to model as it has highest
      r squared value of 0.5441463')
```

```
## [1] "adding lstat to model as it has highest \n      r squared value of 0.5441463"
```

```
for(i in col_lst)
{
  print(i)
  print(summary(lm(data=Boston,medv ~ lstat + get(i)))$r.squared)
}
```

```
## [1] "crim"
## [1] 0.5475862
## [1] "zn"
## [1] 0.5478995
## [1] "indus"
## [1] 0.5464578
## [1] "chas"
## [1] 0.5625543
## [1] "nox"
## [1] 0.5442586
## [1] "rm"
## [1] 0.6385616
## [1] "age"
## [1] 0.5512689
## [1] "dis"
## [1] 0.5622284
## [1] "rad"
## [1] 0.5447338
## [1] "tax"
## [1] 0.5505702
## [1] "ptratio"
## [1] 0.6066546
## [1] "black"
## [1] 0.5487897
## [1] "lstat"
## [1] 0.5441463
```

```
print('adding rm to model as it increases the r
      squared value the greatest to 0.6385616')
```

```
## [1] "adding rm to model as it increases the r \n      squared value the greatest to 0.6385616"
```

```
for(i in col_lst)
{
  print(i)
  print(summary(lm(data=Boston,medv ~ lstat + rm + get(i)))$r.squared)
}
```

```
## [1] "crim"
## [1] 0.6458521
## [1] "zn"
## [1] 0.6398856
```

```
## [1] "indus"
## [1] 0.6399917
## [1] "chas"
## [1] 0.6514028
## [1] "nox"
## [1] 0.6389104
## [1] "rm"
## [1] 0.6385616
## [1] "age"
## [1] 0.6390341
## [1] "dis"
## [1] 0.6467821
## [1] "rad"
## [1] 0.6427861
## [1] "tax"
## [1] 0.6485148
## [1] "ptratio"
## [1] 0.6786242
## [1] "black"
## [1] 0.6505548
## [1] "lstat"
## [1] 0.6385616
```

```
print('adding ptratio to model as it increases the
      r squared value the greatest to 0.6786242')
```

```
## [1] "adding ptratio to model as it increases the \n      r squared value the greatest to 0.6786242"
```

```
for(i in col_lst)
{
  print(i)
  print(summary(lm(data=Boston,medv ~ lstat + rm +
                     ptratio + get(i)))$r.squared)
}
```

```
## [1] "crim"
## [1] 0.6814923
## [1] "zn"
## [1] 0.6789745
## [1] "indus"
## [1] 0.6786435
## [1] "chas"
## [1] 0.6874723
## [1] "nox"
## [1] 0.6792051
## [1] "rm"
## [1] 0.6786242
## [1] "age"
## [1] 0.6801748
## [1] "dis"
## [1] 0.6903077
## [1] "rad"
## [1] 0.6787664
## [1] "tax"
## [1] 0.6796626
```

```
## [1] "ptratio"
## [1] 0.6786242
## [1] "black"
## [1] 0.6877468
## [1] "lstat"
## [1] 0.6786242
```

```
print('adding dis to model as it increases the
      r squared value the greatest to 0.6903077')
```

```
## [1] "adding dis to model as it increases the \n      r squared value the greatest to 0.6903077"
```

```
for(i in col_lst)
{
  print(i)
  print(summary(lm(data=Boston,medv ~ lstat + rm +
                     ptratio + dis + get(i)))$r.squared)
}
```

```
## [1] "crim"
## [1] 0.695775
## [1] "zn"
## [1] 0.6936977
## [1] "indus"
## [1] 0.6959881
## [1] "chas"
## [1] 0.6965682
## [1] "nox"
## [1] 0.7080893
## [1] "rm"
## [1] 0.6903077
## [1] "age"
## [1] 0.6917442
## [1] "dis"
## [1] 0.6903077
## [1] "rad"
## [1] 0.6908321
## [1] "tax"
## [1] 0.6959341
## [1] "ptratio"
## [1] 0.6903077
## [1] "black"
## [1] 0.7020747
## [1] "lstat"
## [1] 0.6903077
```

```
print('adding nox to model as it increases the
      r squared value the greatest to 0.7080893')
```

```
## [1] "adding nox to model as it increases the \n      r squared value the greatest to 0.7080893"
```

```
for(i in col_lst)
{
  print(i)
  print(summary(lm(data=Boston,medv ~ lstat + rm +
                     ptratio + dis + nox + get(i)))$r.squared)
}
```

```
## [1] "crim"
## [1] 0.7114003
## [1] "zn"
## [1] 0.7116409
## [1] "indus"
## [1] 0.7084897
## [1] "chas"
## [1] 0.7157742
## [1] "nox"
## [1] 0.7080893
## [1] "rm"
## [1] 0.7080893
## [1] "age"
## [1] 0.708095
## [1] "dis"
## [1] 0.7080893
## [1] "rad"
## [1] 0.7093413
## [1] "tax"
## [1] 0.7083351
## [1] "ptratio"
## [1] 0.7080893
## [1] "black"
## [1] 0.7153894
## [1] "lstat"
## [1] 0.7080893
```

```
print('adding chas to model as it increases the
      r squared value the greatest to 0.7157742')
```

```
## [1] "adding chas to model as it increases the \n      r squared value the greatest to 0.7157742"
```

```
for(i in col_lst)
{
  print(i)
  print(summary(lm(data=Boston,medv ~ lstat + rm +
                     ptratio + dis + nox + chas
                   + get(i)))$r.squared)
}
```

```
## [1] "crim"
## [1] 0.7184975
## [1] "zn"
## [1] 0.719623
## [1] "indus"
## [1] 0.7163893
## [1] "chas"
## [1] 0.7157742
## [1] "nox"
## [1] 0.7157742
## [1] "rm"
## [1] 0.7157742
## [1] "age"
## [1] 0.7158288
## [1] "dis"
```

```
## [1] 0.7157742
## [1] "rad"
## [1] 0.717145
## [1] "tax"
## [1] 0.7158722
## [1] "ptratio"
## [1] 0.7157742
## [1] "black"
## [1] 0.7221614
## [1] "lstat"
## [1] 0.7157742
```

```
print('adding black to model as it increases the
      r squared value the greatest to 0.7221614')
```

```
## [1] "adding black to model as it increases the \n      r squared value the greatest to 0.7221614"
```

```
for(i in col_lst)
{
  print(i)
  print(summary(lm(data=Boston,medv ~ lstat + rm +
                     ptratio + dis + nox + chas +
                     black + get(i)))$r.squared)
}
```

```
## [1] "crim"
## [1] 0.7234638
## [1] "zn"
## [1] 0.7266079
## [1] "indus"
## [1] 0.7225262
## [1] "chas"
## [1] 0.7221614
## [1] "nox"
## [1] 0.7221614
## [1] "rm"
## [1] 0.7221614
## [1] "age"
## [1] 0.7223825
## [1] "dis"
## [1] 0.7221614
## [1] "rad"
## [1] 0.72554
## [1] "tax"
## [1] 0.7222247
## [1] "ptratio"
## [1] 0.7221614
## [1] "black"
## [1] 0.7221614
## [1] "lstat"
## [1] 0.7221614
```

```
print('adding zn to model as it increases the
      r squared value the greatest to 0.7266079')
```

```
## [1] "adding zn to model as it increases the \n      r squared value the greatest to 0.7266079"
```

```
for(i in col_lst)
{
  print(i)
  print(summary(lm(data=Boston,medv ~ lstat + rm +
                     ptratio + dis + nox + chas +
                     black + zn + get(i)))$r.squared)
}
```

```
## [1] "crim"
## [1] 0.7288251
## [1] "zn"
## [1] 0.7266079
## [1] "indus"
## [1] 0.7269835
## [1] "chas"
## [1] 0.7266079
## [1] "nox"
## [1] 0.7266079
## [1] "rm"
## [1] 0.7266079
## [1] "age"
## [1] 0.7266428
## [1] "dis"
## [1] 0.7266079
## [1] "rad"
## [1] 0.7287994
## [1] "tax"
## [1] 0.7267004
## [1] "ptratio"
## [1] 0.7266079
## [1] "black"
## [1] 0.7266079
## [1] "lstat"
## [1] 0.7266079
```

```
print('adding crim to model as it increases the
      r squared value the greatest to 0.7288251')
```

```
## [1] "adding crim to model as it increases the \n      r squared value the greatest to 0.7288251"
```

```
for(i in col_lst)
{
  print(i)
  print(summary(lm(data=Boston,medv ~ lstat + rm +
                     ptratio + dis + nox + chas +
                     black + zn + crim + get(i)))$r.squared)
}
```

```
## [1] "crim"
## [1] 0.7288251
## [1] "zn"
## [1] 0.7288251
## [1] "indus"
## [1] 0.7291943
## [1] "chas"
```

```
## [1] 0.7288251
## [1] "nox"
## [1] 0.7288251
## [1] "rm"
## [1] 0.7288251
## [1] "age"
## [1] 0.7288829
## [1] "dis"
## [1] 0.7288251
## [1] "rad"
## [1] 0.7341768
## [1] "tax"
## [1] 0.7288556
## [1] "ptratio"
## [1] 0.7288251
## [1] "black"
## [1] 0.7288251
## [1] "lstat"
## [1] 0.7288251
```

```
print('adding rad to model as it increases the
      r squared value the greatest to  0.7341768')
```

```
## [1] "adding rad to model as it increases the \n      r squared value the greatest to  0.7341768"
```

```
for(i in col_lst)
{
  print(i)
  print(summary(lm(data=Boston,medv ~ lstat + rm +
                     ptratio + dis + nox + chas +
                     black + zn + crim + rad + get(i)))$r.squared)
}
```

```
## [1] "crim"
## [1] 0.7341768
## [1] "zn"
## [1] 0.7341768
## [1] "indus"
## [1] 0.7349702
## [1] "chas"
## [1] 0.7341768
## [1] "nox"
## [1] 0.7341768
## [1] "rm"
## [1] 0.7341768
## [1] "age"
## [1] 0.734179
## [1] "dis"
## [1] 0.7341768
## [1] "rad"
## [1] 0.7341768
## [1] "tax"
## [1] 0.7405823
## [1] "ptratio"
## [1] 0.7341768
```

```
## [1] "black"
## [1] 0.7341768
## [1] "lstat"
## [1] 0.7341768
```

```
print('adding tax to model as it increases the
      r squared value the greatest to 0.7405823')
```

```
## [1] "adding tax to model as it increases the \n      r squared value the greatest to 0.7405823"
```

```
for(i in col_lst)
{
  print(i)
  print(summary(lm(data=Boston,medv ~ lstat + rm +
                     ptratio + dis + nox + chas + black + zn +
                     crim + rad + tax + get(i)))$r.squared)
}
```

```
## [1] "crim"
## [1] 0.7405823
## [1] "zn"
## [1] 0.7405823
## [1] "indus"
## [1] 0.7406412
## [1] "chas"
## [1] 0.7405823
## [1] "nox"
## [1] 0.7405823
## [1] "rm"
## [1] 0.7405823
## [1] "age"
## [1] 0.7405837
## [1] "dis"
## [1] 0.7405823
## [1] "rad"
## [1] 0.7405823
## [1] "tax"
## [1] 0.7405823
## [1] "ptratio"
## [1] 0.7405823
## [1] "black"
## [1] 0.7405823
## [1] "lstat"
## [1] 0.7405823
```

```
print('As model r squared is not improving any
      further, this is the optimal stepwise selected model')
```

```
## [1] "As model r squared is not improving any \n      further, this is the optimal stepwise selected m
```

```
stepwise_model1 <- lm(data=Boston,medv ~ lstat + rm +
                        ptratio + dis + nox +
                        chas + black + zn + crim + rad + tax)
print(summary(stepwise_model1))
```

```
##
## Call:
```

```
## lm(formula = medv ~ lstat + rm + ptratio + dis + nox + chas +
##     black + zn + crim + rad + tax, data = Boston)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.5984  -2.7386  -0.5046   1.7273  26.2373
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  36.341145   5.067492   7.171 2.73e-12 ***
## lstat        -0.522553   0.047424 -11.019  < 2e-16 ***
## rm            3.801579   0.406316   9.356  < 2e-16 ***
## ptratio      -0.946525   0.129066  -7.334 9.24e-13 ***
## dis          -1.492711   0.185731  -8.037 6.84e-15 ***
## nox         -17.376023   3.535243  -4.915 1.21e-06 ***
## chas1         2.718716   0.854240   3.183 0.001551 **
## black         0.009291   0.002674   3.475 0.000557 ***
## zn            0.045845   0.013523   3.390 0.000754 ***
## crim         -0.108413   0.032779  -3.307 0.001010 **
## rad           0.299608   0.063402   4.726 3.00e-06 ***
## tax          -0.011778   0.003372  -3.493 0.000521 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.736 on 494 degrees of freedom
## Multiple R-squared:  0.7406, Adjusted R-squared:  0.7348
## F-statistic: 128.2 on 11 and 494 DF,  p-value: < 2.2e-16
```

1.8 Do Assumptions Hold?

Evaluate the statistical assumptions in your regression analysis from (1.7) by performing a basic analysis of model residuals and any unusual observations (consult Openintro Statistics 7.2). Discuss any concerns you have about your model.

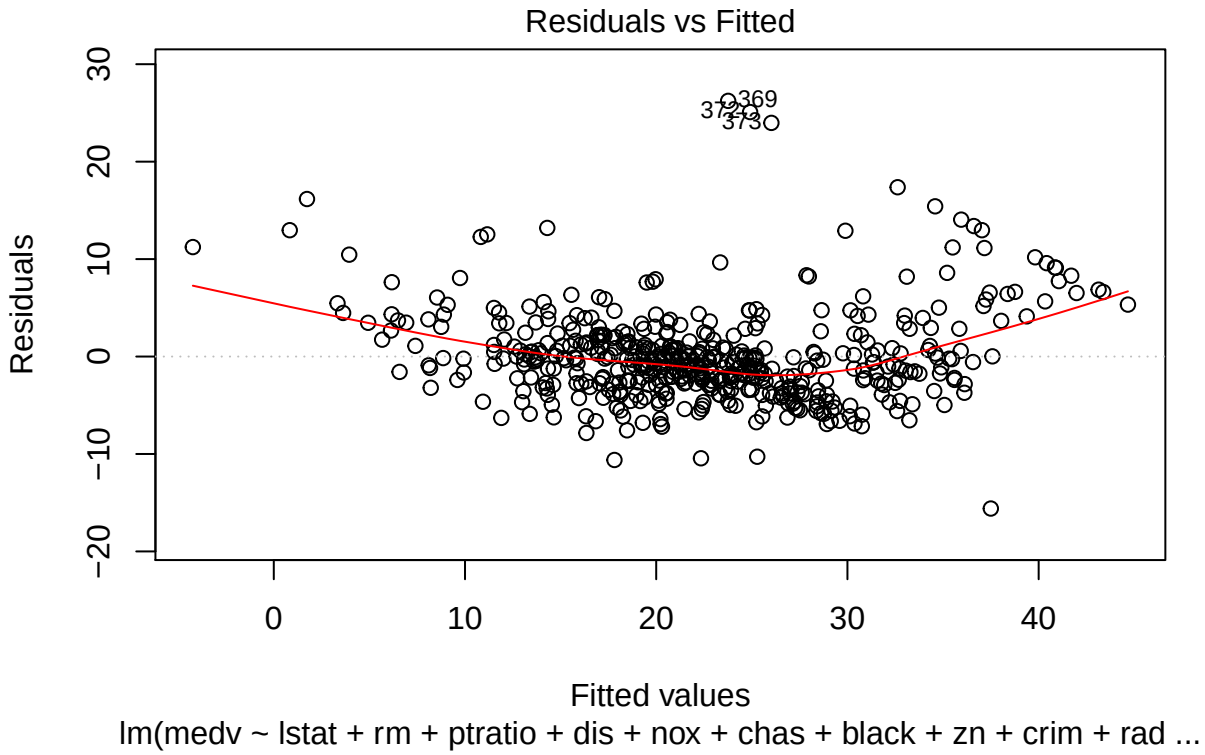
Ans. The statistical assumptions in the analysis are Normality, Homoscedasticity, Independence, and Linearity.

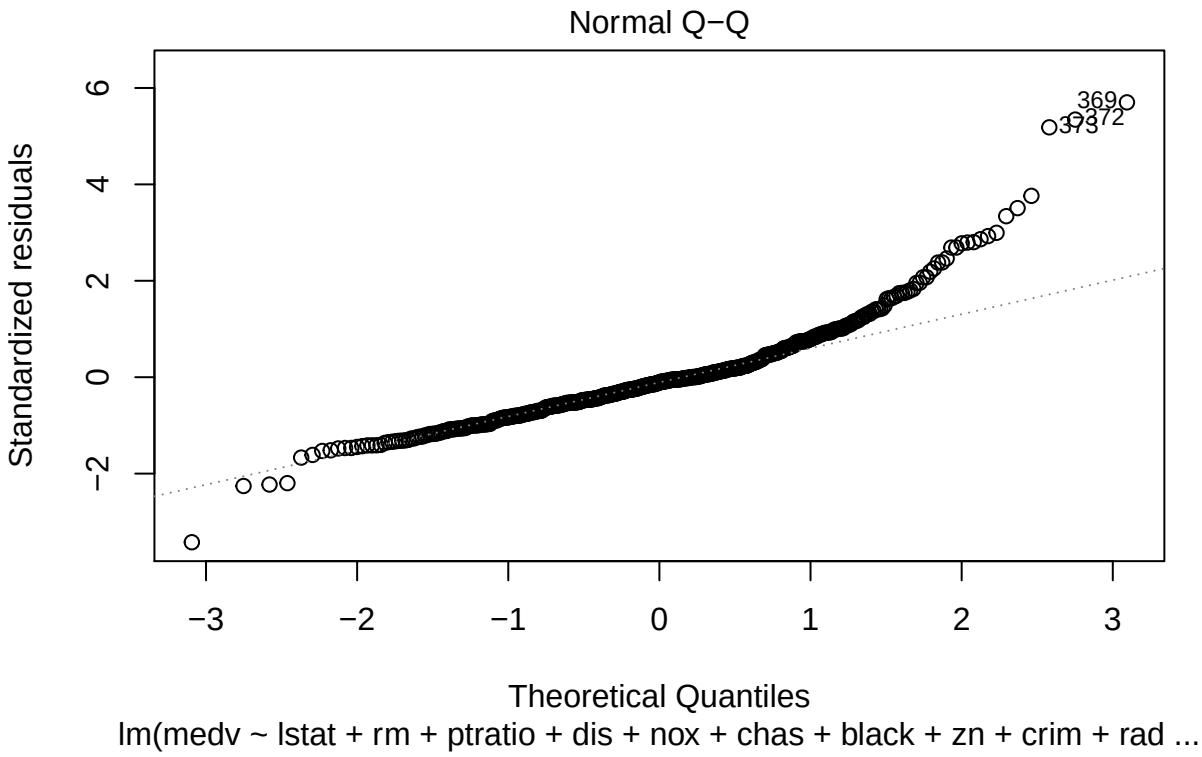
1. The QQ plot of the residuals is shown below in #1. There is a slight deviation from the normal distribution at the right end which indicates that the residuals might not be normally distributed, which violates the assumptions of the analysis.
2. Absolute values of residuals against fitted values #2 - There is a slight dip in the variance of residuals vs the fitted values for certain values of fitted values which violates the constant variance assumption for the linear model. It also suggests that the model might not be a linear one.
3. Residuals in order of data collection- In order to check if residuals are independent, residuals are plotted in the order of data collection in #3. The assumption is that the data records in Boston dataset are ordered in the order of data collection. The data seems to indicate that consecutive observations tend to be close to each other, violating the statistical assumptions of the analysis. But there are clear deviations from this trend around observations 150 and 370, indicating that successive observations may in fact not be related, and there is no clear evidence that consecutive observations are related to each other.
4. Residuals plotted against each predictor variable- For the variables rm and lstat, the curvature in the plot of residuals indicates that the relationship with these variables might not be linear, which is also indicated by the previous questions. There is also a slight curvature in the plot of the residuals crim, zn, rad, dis which might indicate that the relationship of these variables with median value of homes

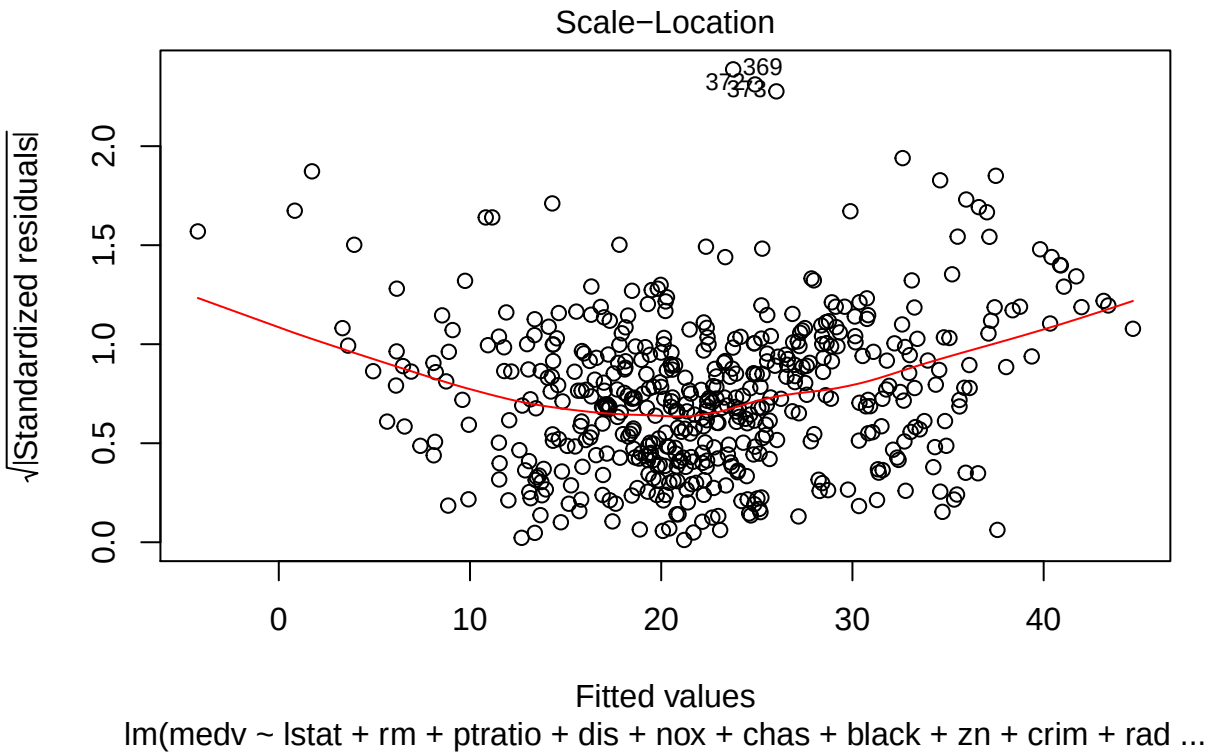
might not be linear. This might violate the assumptions of the analysis that all predictors are mostly linearly associated with the response variable.

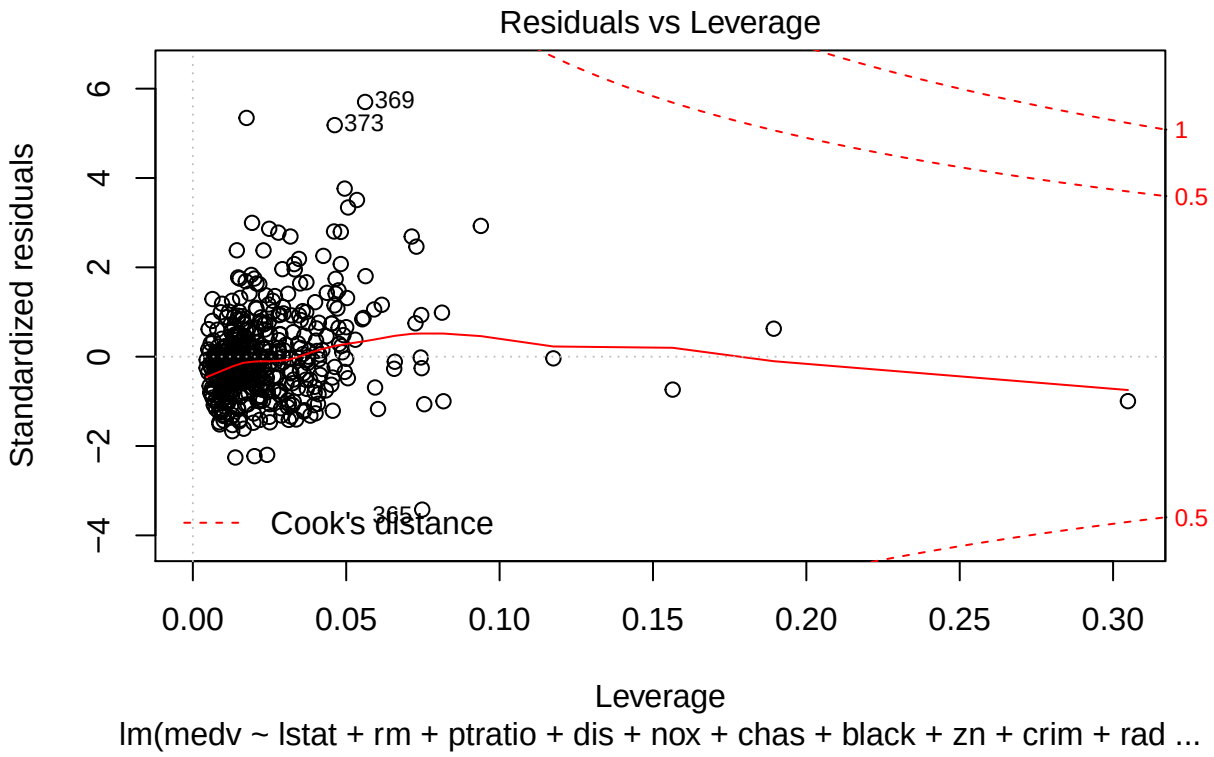
```
#checking for the linearmodel assumptions using plots
mod_residuals <- stepwise_model1$residuals
fitted_vals <- stepwise_model1$fitted.values
mod_pred <- col_lst[col_lst != 'age' & col_lst!='indus']

#checking for normal dist and homoedasticity
#1,2
plot(stepwise_model1)
```

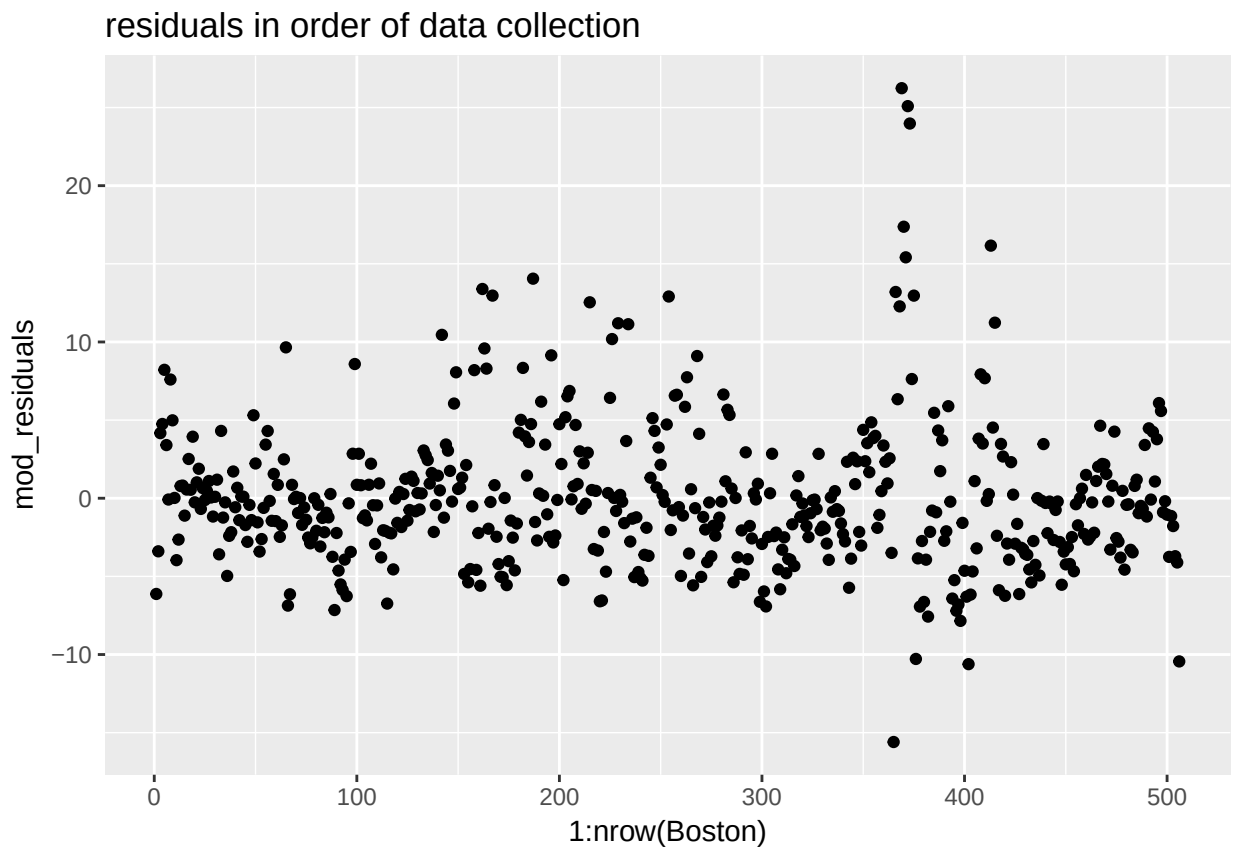



Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(medv ~ lstat + rm + ptratio + dis + nox + chas + black + zn + crim + rad ...
369
372
373
−3
−2
−1
0
1
2
3
−2
0
2
4
6

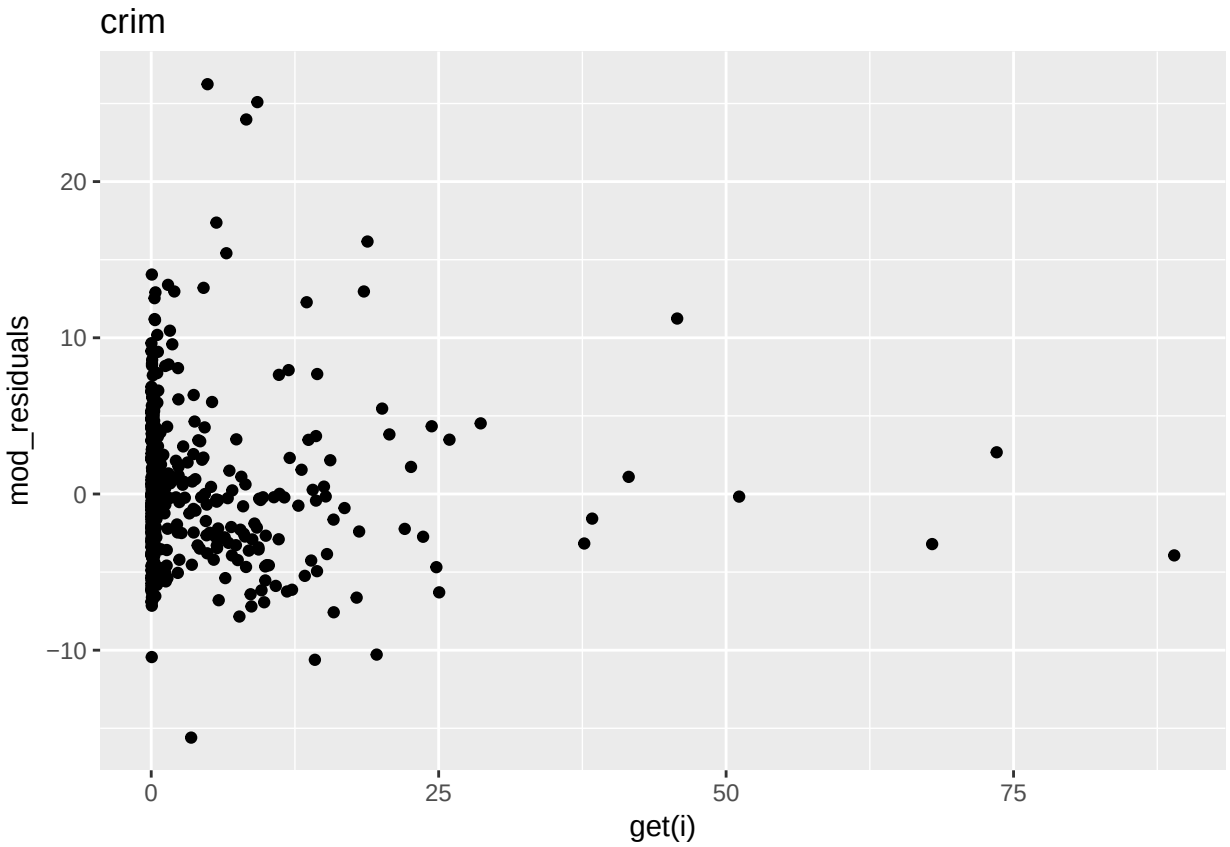

Scale−Location
369
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
0
10
20
30
40
Fitted values
lm(medv ~ lstat + rm + ptratio + dis + nox + chas + black + zn + crim + rad ...

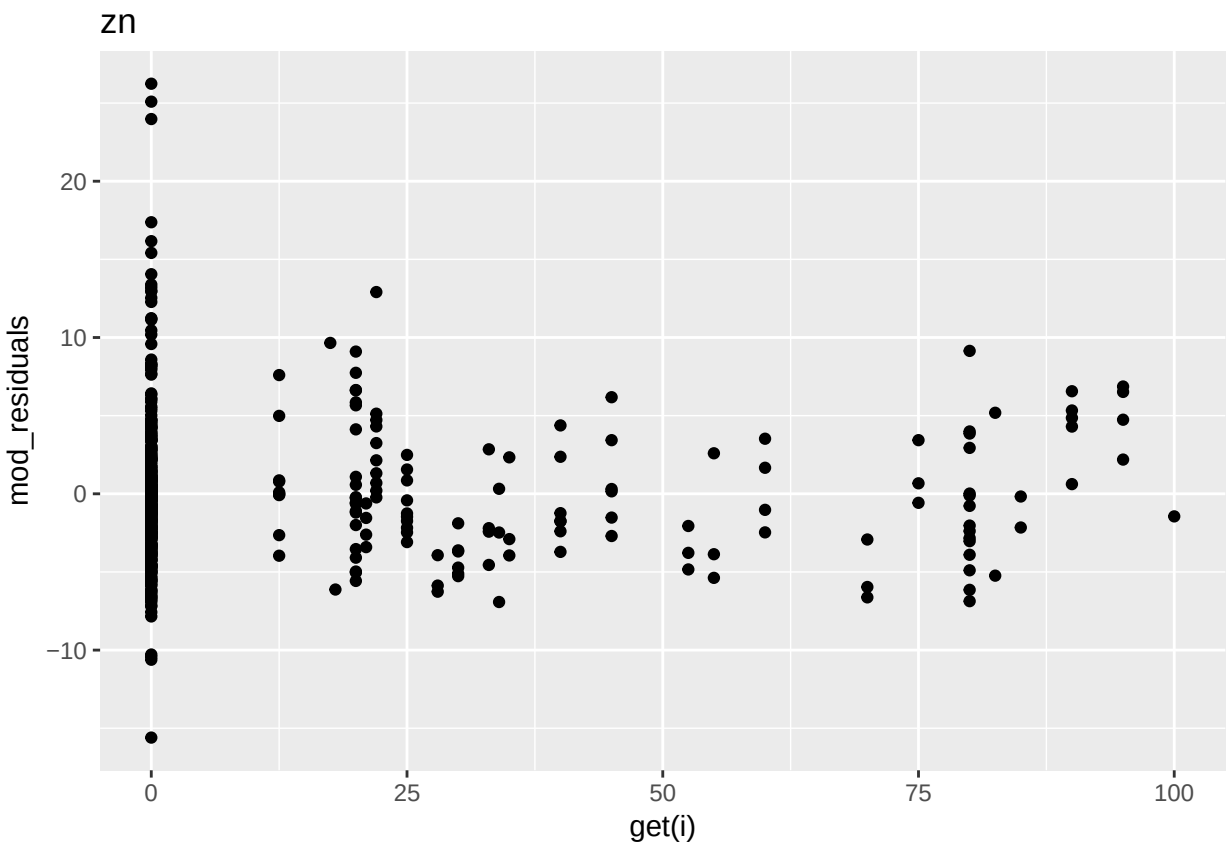


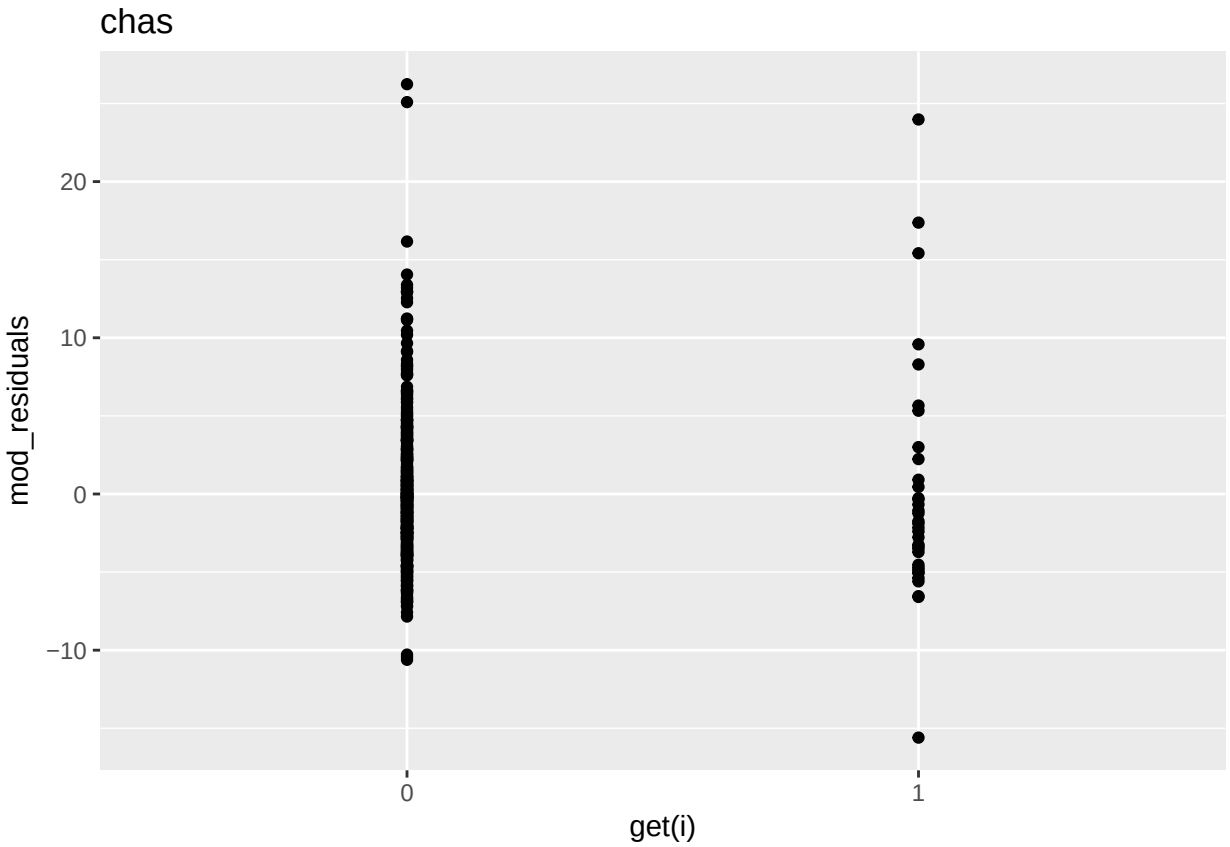

```
#checking for independency
#3
ggplot()+geom_point(data=Boston,aes(x=1:nrow(Boston),y=mod_residuals))+
  ggtitle("residuals in order of data collection")
```

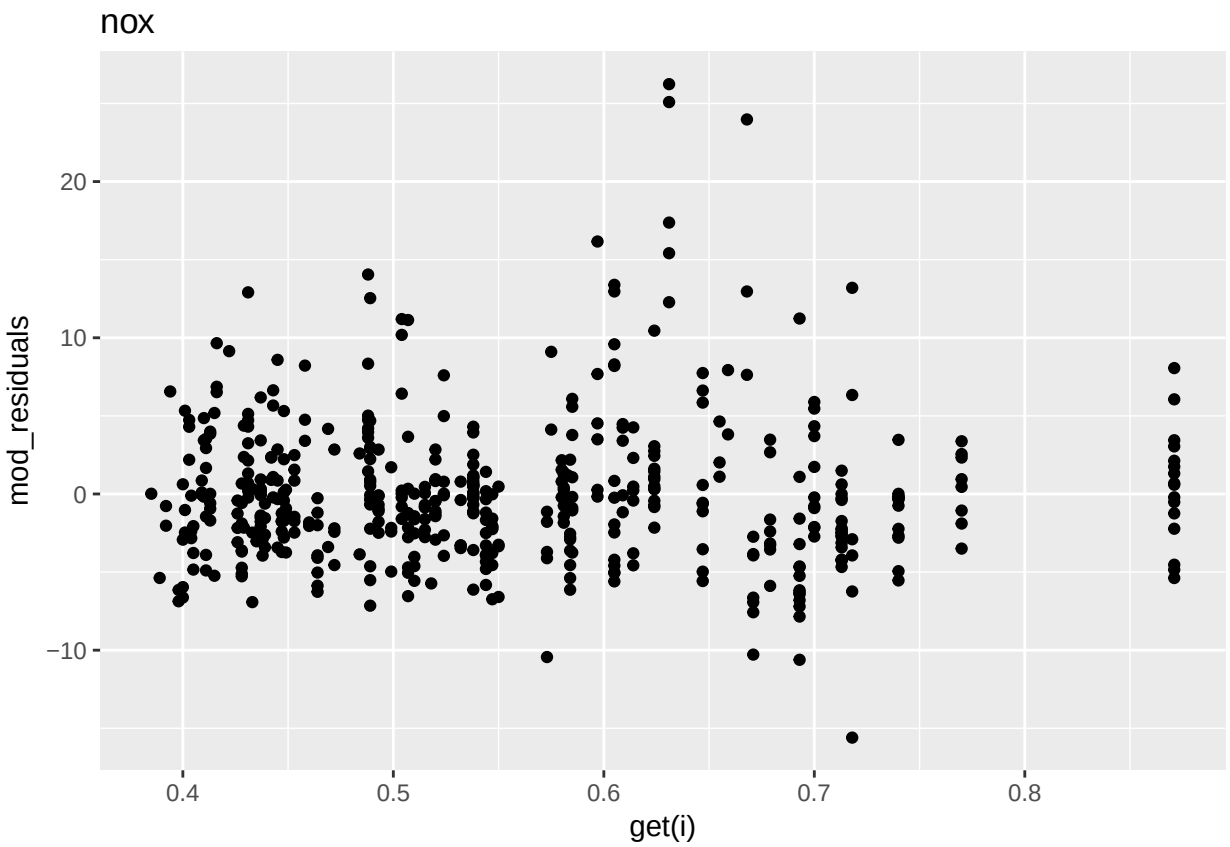


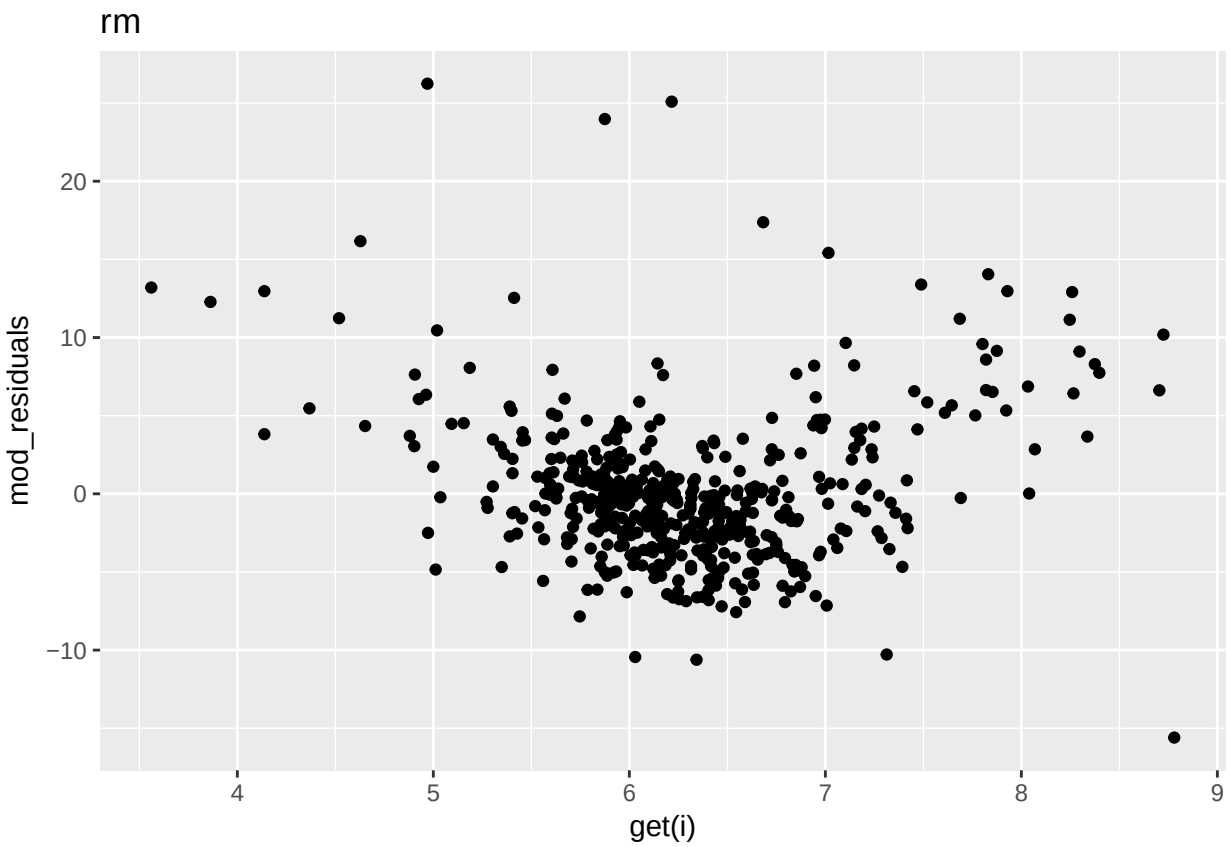

```
#checking for linearity
#4
for(i in mod_pred)
{
print(ggplot()+geom_point(data=Boston,
                          aes(x=get(i),y=mod_residuals))+ggtitle(i))
}
```

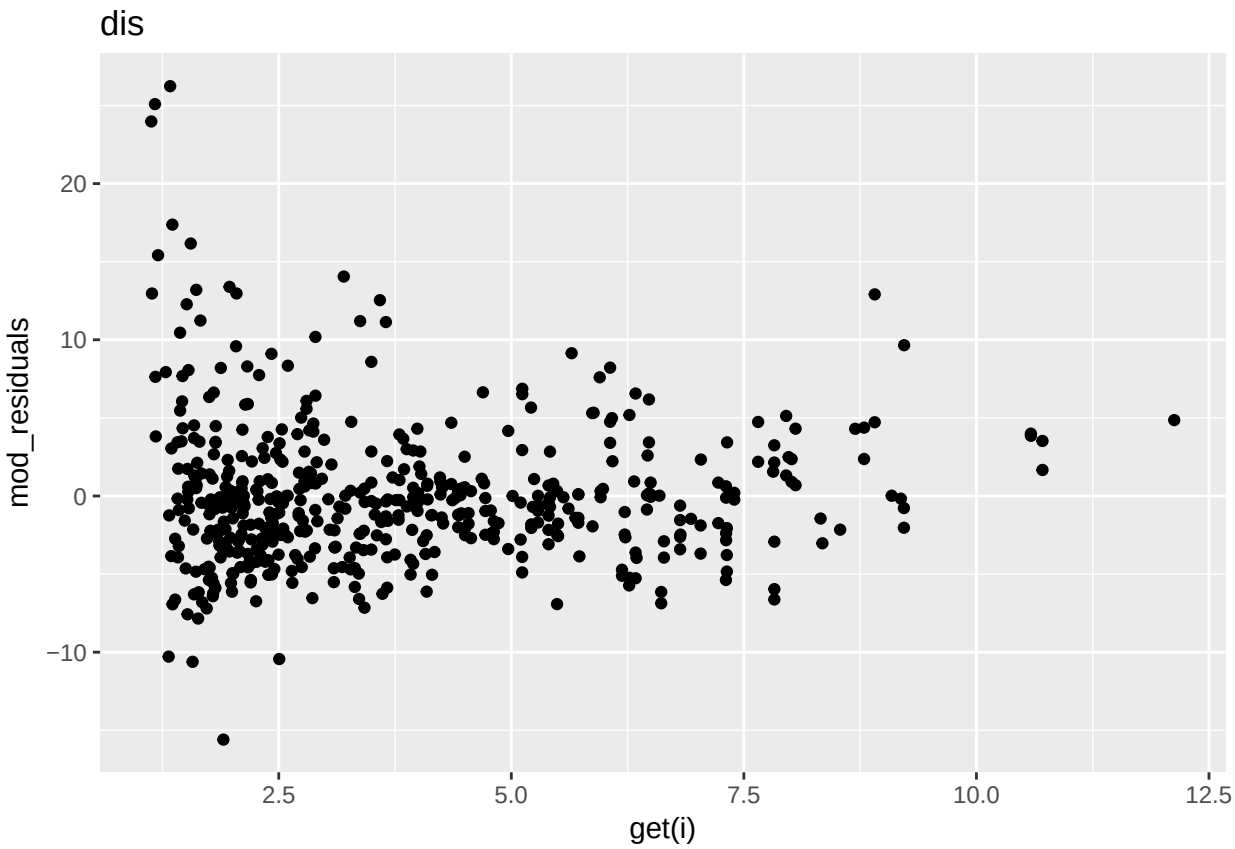

crim
mod_residuals
20
10
0
−10
0
25
50
75
get(i)

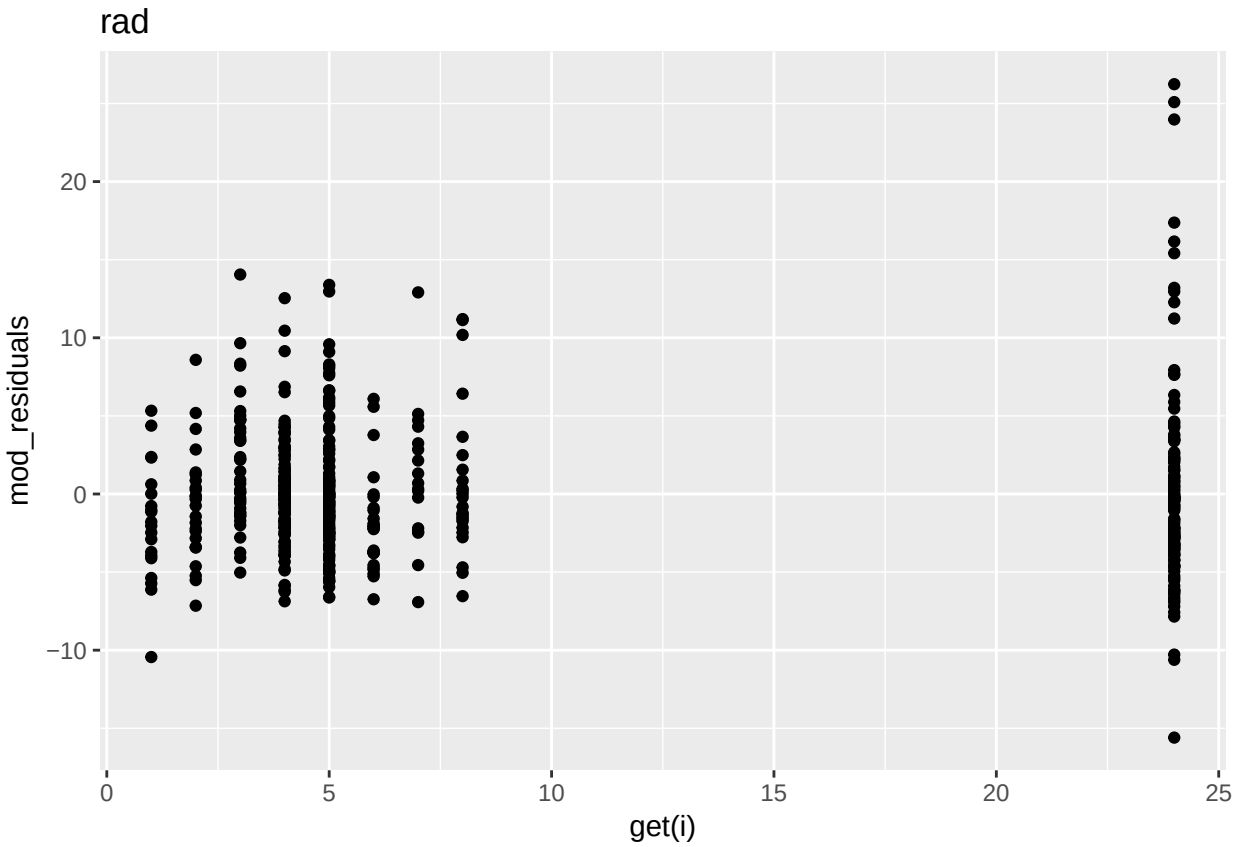

zn
mod_residuals
20
10
0
−10
0
25
50
75
100
get(i)

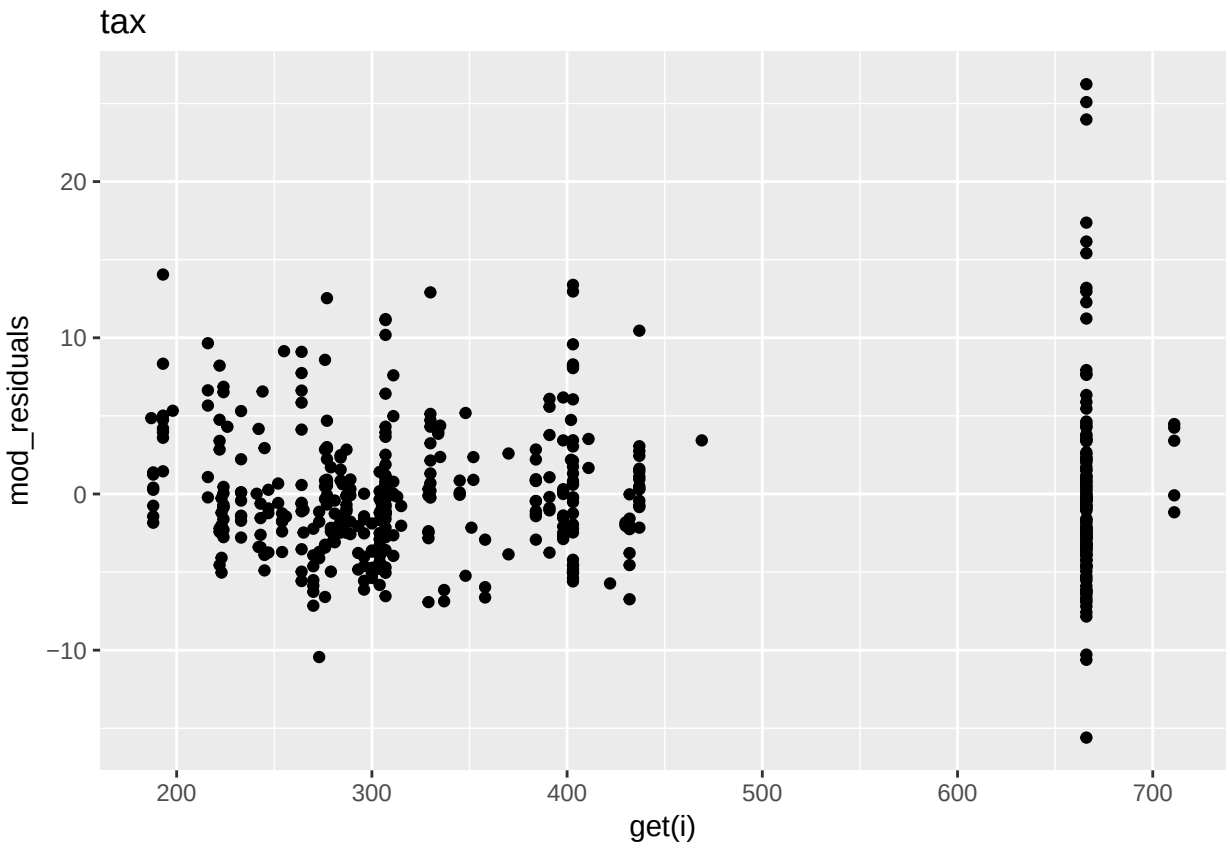

chas
mod_residuals
20
10
0
−10
0
1
get(i)

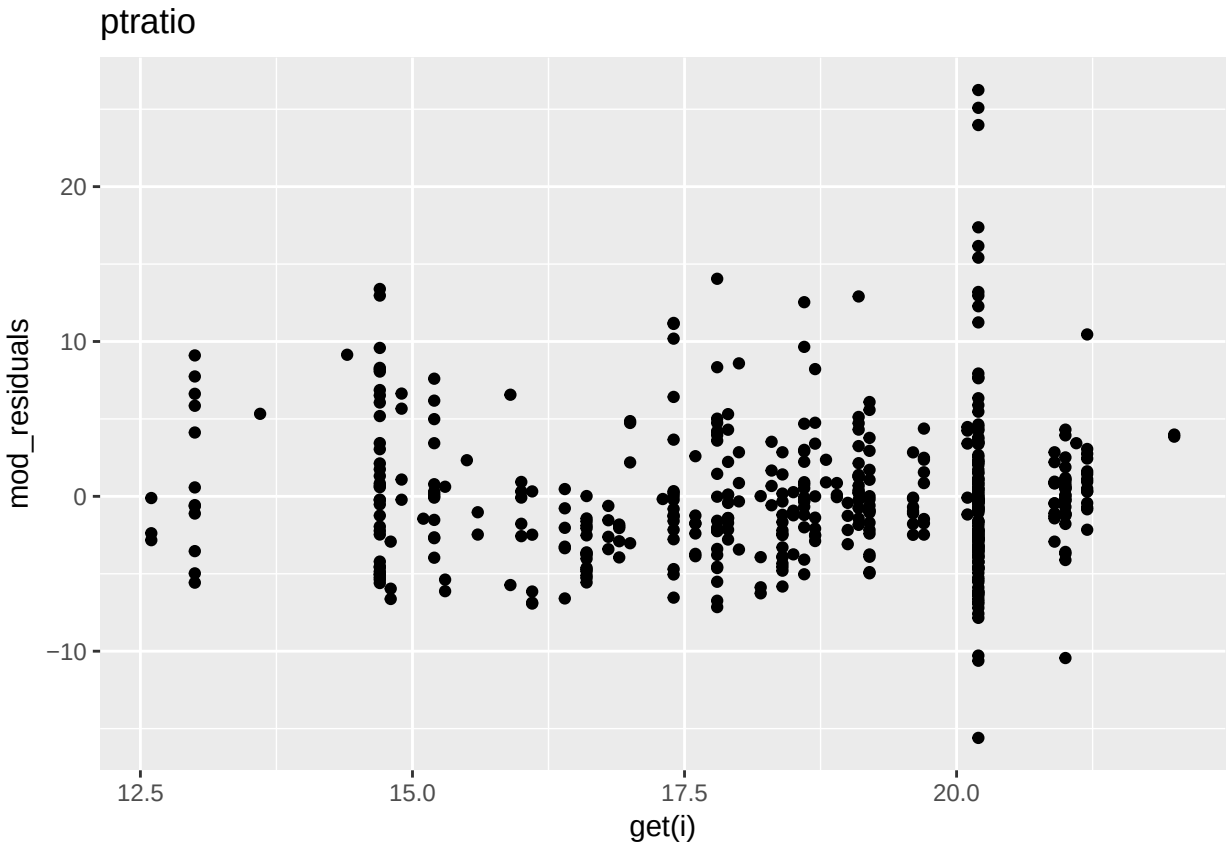

nox
mod_residuals
20
10
0
−10
0.4
0.5
0.6
0.7
0.8
get(i)

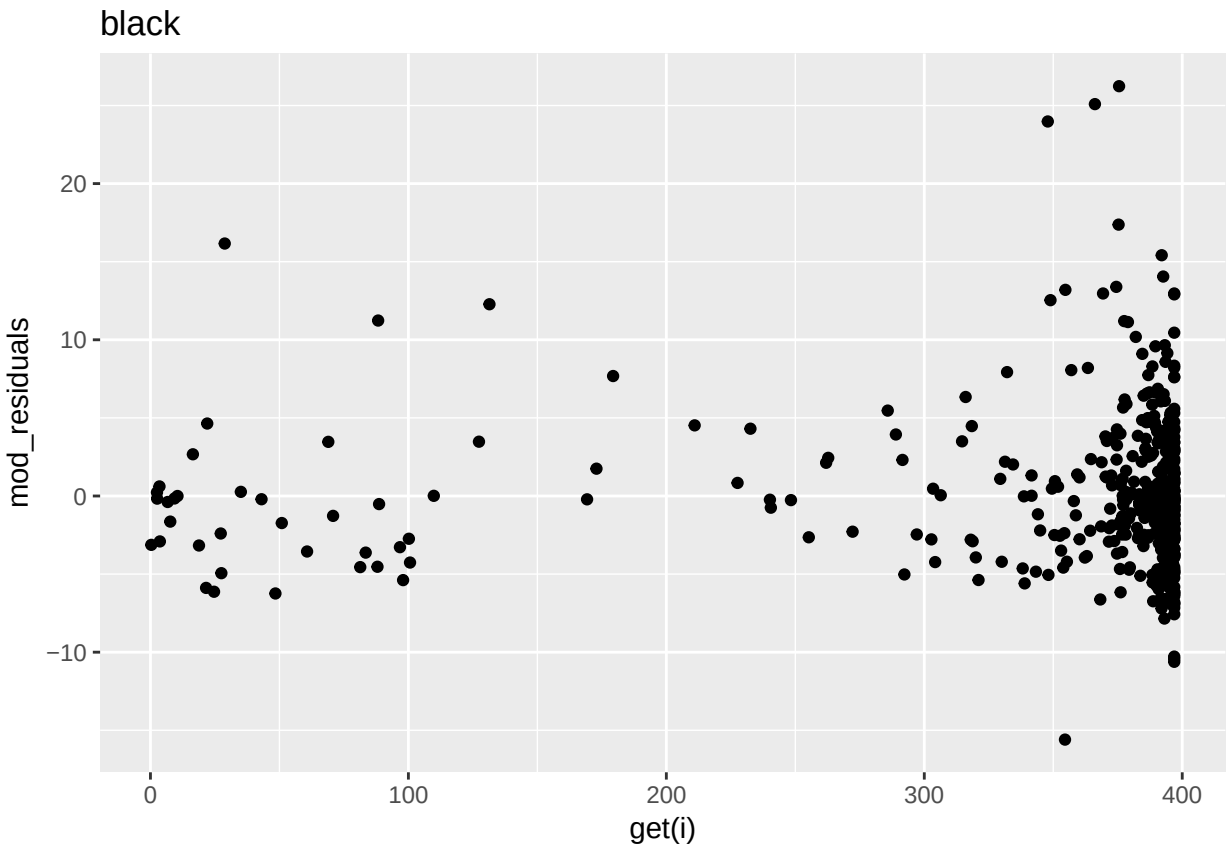

rm
mod_residuals
20
10
0
−10
4
5
6
7
8
9
get(i)

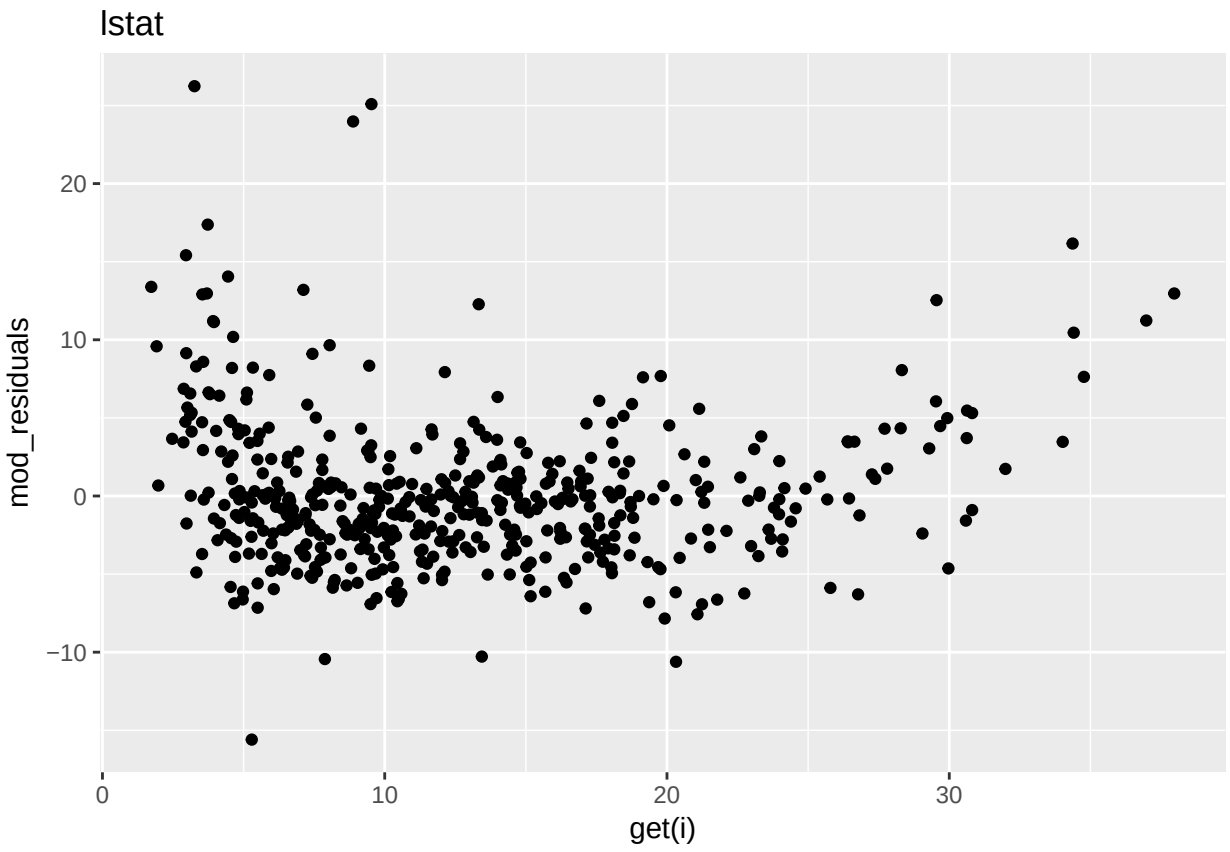

dis
mod_residuals
20
10
0
−10
2.5
5.0
7.5
10.0
12.5
get(i)


rad
mod_residuals
20
10
0
−10
0
5
10
15
20
25
get(i)


tax
mod_residuals
20
10
0
−10
200
300
400
500
600
700
get(i)


ptratio
mod_residuals
20
10
0
−10
12.5
15.0
17.5
20.0
get(i)


black
mod_residuals
20
10
0
−10
0
100
200
300
400
get(i)



2. Diamonds' Price

Let's look at the *diamonds* dataset from *ggplot2* package. Your task is to find which parameters influence the price of diamonds.

I recommend to transform the ordered factors (such as *cut*, *clarity*) to unordered factors with a command like `factor(cut, ordered=FALSE)` in order to give more easily interpretable results.

```
#transfrom categorical variables to unordered factors
library(tidyverse)
library(ggplot2)
diamonds1 <- ggplot2::diamonds %>%
  mutate(cut=factor(cut, ordered=FALSE),
         clarity=factor(clarity,ordered=FALSE),
         color=factor(color,ordered=FALSE))
str(diamonds1)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    53940 obs. of  10 variables:
##  $ carat  : num  0.23 0.21 0.23 0.29 0.31 0.24 0.24 0.26 0.22 0.23 ...
##  $ cut    : Factor w/ 5 levels "Fair","Good",..: 5 4 2 4 2 3 3 3 1 3 ...
##  $ color  : Factor w/ 7 levels "D","E","F","G",..: 2 2 2 6 7 7 6 5 2 5 ...
##  $ clarity: Factor w/ 8 levels "I1","SI2","SI1",..: 2 3 5 4 2 6 7 3 4 5 ...
##  $ depth  : num  61.5 59.8 56.9 62.4 63.3 62.8 62.3 61.9 65.1 59.4 ...
##  $ table  : num  55 61 65 58 58 57 57 55 61 61 ...
##  $ price  : int  326 326 327 334 335 336 336 337 337 338 ...
##  $ x      : num  3.95 3.89 4.05 4.2 4.34 3.94 3.95 4.07 3.87 4 ...
```

```
##  $ y       : num  3.98 3.84 4.07 4.23 4.35 3.96 3.98 4.11 3.78 4.05 ...
##  $ z       : num  2.43 2.31 2.31 2.63 2.75 2.48 2.47 2.53 2.49 2.39 ...
```

2.1 Describe the variables.

What do you think, which variables are relevant in determining the price? Describe your thought before you do any formal analysis.

Ans. It is a dataset containing the prices and other attributes of almost 54,000 diamonds. It has 53940 rows and 10 variables.

The variables in the dataset are price: price in US dollars of the diamond carat: weight of the diamond cut: quality of the cut (Fair, Good, Very Good,Premium, Ideal) color: diamond colour, from J (worst) to D (best) clarity: A measurement of how clear the diamond is (I1 (worst), SI2, SI1, VS2, VS1, VVS2, VVS1, IF (best)) x: length in mm y: width in mm z: depth in mm depth: total depth percentage = z / mean(x, y) = 2 * z / (x + y) table: width of top of diamond relative to widest point

Source: http://ggplot2.tidyverse.org/reference/diamonds.html

I feel that the variables which might have the strongest association with the price of the diamond are carat, color, cut, and clarity of the diamond. The x,y,z, depth percentage, and table might also be associated as these variables indicate the volume of the diamond, which i feel might strongly associated with the diamond price.

2.2 Multiple regression

Select a number of variables you consider the most relevant. Estimate a multiple regression model in the form

$$\text{price}_i = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + \cdots + \epsilon_i.$$

Interpret the coefficient values.

- if you are able to, give the literal interpretation of the numeric value
- if there is no easy literal interpretation, broadly explain what it means, and interpret at least the sign.

Ans. The intercept has no literal interpretation as a diamond would never weigh 0 carats, have all the size dimensions as 0, and not belong to any of the color, cut, clarity combinations. For the color characteristic, having a color in the set [E,F,H,I,J] would signify a change in the price of the diamond by [-209.118,-272.854,-482.039,-980.267,-1466.244,-2369.398] respectively. This indicates that other factors kept constant, the prices for diamonds might take the following order- E>F>G>H>I>J. A similar thing can be said about the clarity and cut categorical predictors using the respective beta estmates. For the carat predictor, a unit change in the carat value of the diamond signifies a price change of 11256.978 units. A similar kind of thing could be said about the relationship between the quantitative variables x,y,z, depth, and table using the respective values of the beta estimates.

```
#create linear model
diamond_model <- lm(data=diamonds1, price ~ carat+color+cut+
                      clarity+x+y+z+table+depth)
summary(diamond_model)
```

```
##
## Call:
## lm(formula = price ~ carat + color + cut + clarity + x + y +
##     z + table + depth, data = diamonds1)
##
## Residuals:
```

```
##      Min       1Q   Median       3Q      Max
## -21376.0   -592.4   -183.5    376.4  10694.2
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2184.477    408.197   5.352 8.76e-08 ***
## carat        11256.978     48.628 231.494  < 2e-16 ***
## colorE        -209.118     17.893 -11.687  < 2e-16 ***
## colorF        -272.854     18.093 -15.081  < 2e-16 ***
## colorG        -482.039     17.716 -27.209  < 2e-16 ***
## colorH        -980.267     18.836 -52.043  < 2e-16 ***
## colorI       -1466.244     21.162 -69.286  < 2e-16 ***
## colorJ       -2369.398     26.131 -90.674  < 2e-16 ***
## cutGood        579.751     33.592  17.259  < 2e-16 ***
## cutVery Good   726.783     32.241  22.542  < 2e-16 ***
## cutPremium     762.144     32.228  23.649  < 2e-16 ***
## cutIdeal       832.912     33.407  24.932  < 2e-16 ***
## claritySI2    2702.586     43.818  61.677  < 2e-16 ***
## claritySI1    3665.472     43.634  84.005  < 2e-16 ***
## clarityVS2    4267.224     43.853  97.306  < 2e-16 ***
## clarityVS1    4578.398     44.546 102.779  < 2e-16 ***
## clarityVVS2   4950.814     45.855 107.967  < 2e-16 ***
## clarityVVS1   5007.759     47.160 106.187  < 2e-16 ***
## clarityIF     5345.102     51.024 104.757  < 2e-16 ***
## x            -1008.261     32.898 -30.648  < 2e-16 ***
## y                9.609     19.333   0.497    0.619
## z              -50.119     33.486  -1.497    0.134
## table          -26.474      2.912  -9.092  < 2e-16 ***
## depth          -63.806      4.535 -14.071  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1130 on 53916 degrees of freedom
## Multiple R-squared:  0.9198, Adjusted R-squared:  0.9198
## F-statistic: 2.688e+04 on 23 and 53916 DF,  p-value: < 2.2e-16
```

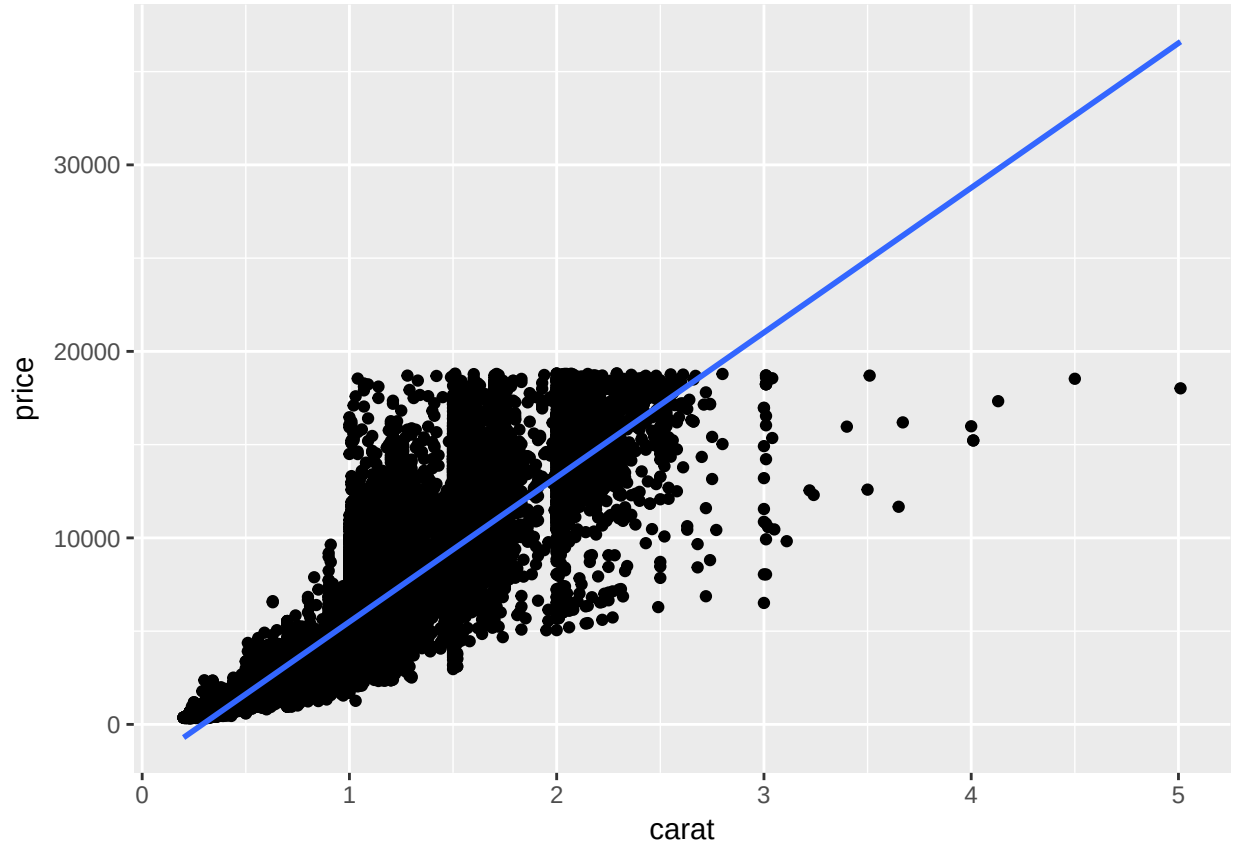
2.3 Other specifications

Select 2-3 different sets of explanatory variables or change the model specification in other ways, for instance by using log of the outcome or explanatory variables, adding interactions and squares, cubes of the variables, normalizing variables, or something else.

Which specification gives you the highest R^2? Comment your results.

Ans. diamond_model2 below gives the highest r squared value of 0.9701. It relates the log value of the outcome with a linear combination of predictors. This might be because the price of the diamond probably might have somewhat of exponential relationship with the predictors.
The other models give r squared values of 0.9198 and 0.9049 for diamond_model1 and diamond_model3 respectively. This might be because the price might not have a strong linear linear relationship with all variables.

```
ggplot(data=diamonds1,aes(x=carat,y=price))+
  geom_point()+geom_smooth(method = 'lm')
```




```
# create 3 models to compare the goodness of fit of each model
diamond_model1 <- lm(data=diamonds1, price ~ carat+
                       color+cut+clarity+x+table+depth)
summary(diamond_model1)
```

```
##
## Call:
## lm(formula = price ~ carat + color + cut + clarity + x + table +
##     depth, data = diamonds1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -21385.0   -592.4   -183.7    376.5  10694.6
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2366.086    390.351   6.061 1.36e-09 ***
## carat        11256.968     48.600 231.626  < 2e-16 ***
## colorE        -209.237     17.893 -11.694  < 2e-16 ***
## colorF        -272.834     18.093 -15.080  < 2e-16 ***
## colorG        -481.943     17.716 -27.204  < 2e-16 ***
## colorH        -980.122     18.836 -52.035  < 2e-16 ***
## colorI       -1466.182     21.162 -69.283  < 2e-16 ***
## colorJ       -2369.504     26.131 -90.678  < 2e-16 ***
## cutGood        580.240     33.572  17.283  < 2e-16 ***
## cutVery Good   726.820     32.212  22.564  < 2e-16 ***
```

```
## cutPremium     762.759     32.225  23.670  < 2e-16 ***
## cutIdeal       833.260     33.396  24.951  < 2e-16 ***
## claritySI2    2702.077     43.812  61.674  < 2e-16 ***
## claritySI1    3664.905     43.627  84.005  < 2e-16 ***
## clarityVS2    4266.612     43.847  97.308  < 2e-16 ***
## clarityVS1    4577.589     44.535 102.786  < 2e-16 ***
## clarityVVS2   4950.168     45.847 107.972  < 2e-16 ***
## clarityVVS1   5007.061     47.152 106.190  < 2e-16 ***
## clarityIF     5344.338     51.015 104.761  < 2e-16 ***
## x            -1029.478     20.549 -50.098  < 2e-16 ***
## table          -26.457      2.911  -9.089  < 2e-16 ***
## depth          -66.769      4.091 -16.322  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1130 on 53918 degrees of freedom
## Multiple R-squared:  0.9198, Adjusted R-squared:  0.9198
## F-statistic: 2.944e+04 on 21 and 53918 DF,  p-value: < 2.2e-16
```

```
diamond_model2 <- lm(data=diamonds1, I(log(price)) ~ carat+
                       color+cut+clarity+x+y+z+table+depth)
summary(diamond_model2)
```

```
##
## Call:
## lm(formula = I(log(price)) ~ carat + color + cut + clarity +
##     x + y + z + table + depth, data = diamonds1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.2544 -0.0911  0.0015  0.0902 10.2241
##
## Coefficients:
##                Estimate Std. Error  t value Pr(>|t|)
## (Intercept)  -3.1460012  0.0634200  -49.606  < 2e-16 ***
## carat        -0.6120540  0.0075551  -81.012  < 2e-16 ***
## colorE       -0.0581373  0.0027800  -20.913  < 2e-16 ***
## colorF       -0.0894535  0.0028110  -31.823  < 2e-16 ***
## colorG       -0.1574203  0.0027525  -57.192  < 2e-16 ***
## colorH       -0.2582798  0.0029265  -88.257  < 2e-16 ***
## colorI       -0.3846735  0.0032879 -116.996  < 2e-16 ***
## colorJ       -0.5243942  0.0040599 -129.166  < 2e-16 ***
## cutGood       0.0911177  0.0052191   17.459  < 2e-16 ***
## cutVery Good  0.1244075  0.0050091   24.836  < 2e-16 ***
## cutPremium    0.1102122  0.0050071   22.011  < 2e-16 ***
## cutIdeal      0.1557466  0.0051904   30.007  < 2e-16 ***
## claritySI2    0.4409340  0.0068079   64.768  < 2e-16 ***
## claritySI1    0.6078471  0.0067793   89.663  < 2e-16 ***
## clarityVS2    0.7503510  0.0068134  110.129  < 2e-16 ***
## clarityVS1    0.8184076  0.0069209  118.251  < 2e-16 ***
## clarityVVS2   0.9380828  0.0071243  131.674  < 2e-16 ***
## clarityVVS1   1.0047321  0.0073270  137.127  < 2e-16 ***
## clarityIF     1.0953087  0.0079274  138.167  < 2e-16 ***
## x             1.1646945  0.0051112  227.871  < 2e-16 ***
## y             0.0325386  0.0030037   10.833  < 2e-16 ***
```

```
## z             0.0427049  0.0052026    8.208 2.29e-16 ***
## table         0.0089978  0.0004524   19.890  < 2e-16 ***
## depth         0.0521543  0.0007045   74.028  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1756 on 53916 degrees of freedom
## Multiple R-squared:  0.9701, Adjusted R-squared:  0.9701
## F-statistic: 7.597e+04 on 23 and 53916 DF,  p-value: < 2.2e-16
```

```
diamond_model3 <- lm(data=diamonds1, price ~ I((carat)^2) +
                       color+cut+clarity+x+table+depth)
summary(diamond_model3)
```

```
##
## Call:
## lm(formula = price ~ I((carat)^2) + color + cut + clarity + x +
##     table + depth, data = diamonds1)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -38344   -673   -179    422  20036
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -17411.138    403.026 -43.201   <2e-16 ***
## I((carat)^2)   2027.328     10.567 191.854   <2e-16 ***
## colorE         -207.209     19.483 -10.635   <2e-16 ***
## colorF         -259.373     19.703 -13.164   <2e-16 ***
## colorG         -452.142     19.291 -23.438   <2e-16 ***
## colorH         -919.281     20.501 -44.840   <2e-16 ***
## colorI        -1405.356     23.034 -61.012   <2e-16 ***
## colorJ        -2313.994     28.452 -81.330   <2e-16 ***
## cutGood         680.760     36.553  18.624   <2e-16 ***
## cutVery Good    855.817     35.064  24.407   <2e-16 ***
## cutPremium      817.486     35.085  23.300   <2e-16 ***
## cutIdeal        908.618     36.359  24.990   <2e-16 ***
## claritySI2     2828.950     47.755  59.238   <2e-16 ***
## claritySI1     3767.299     47.571  79.193   <2e-16 ***
## clarityVS2     4395.871     47.804  91.957   <2e-16 ***
## clarityVS1     4703.055     48.554  96.863   <2e-16 ***
## clarityVVS2    5113.616     49.966 102.342   <2e-16 ***
## clarityVVS1    5184.475     51.372 100.919   <2e-16 ***
## clarityIF      5521.450     55.574  99.353   <2e-16 ***
## x              1938.711     10.170 190.631   <2e-16 ***
## table            -3.917      3.168  -1.236    0.216
## depth            70.389      4.359  16.146   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1231 on 53918 degrees of freedom
## Multiple R-squared:  0.9049, Adjusted R-squared:  0.9049
## F-statistic: 2.443e+04 on 21 and 53918 DF,  p-value: < 2.2e-16
```

2.4 Visualize your best model

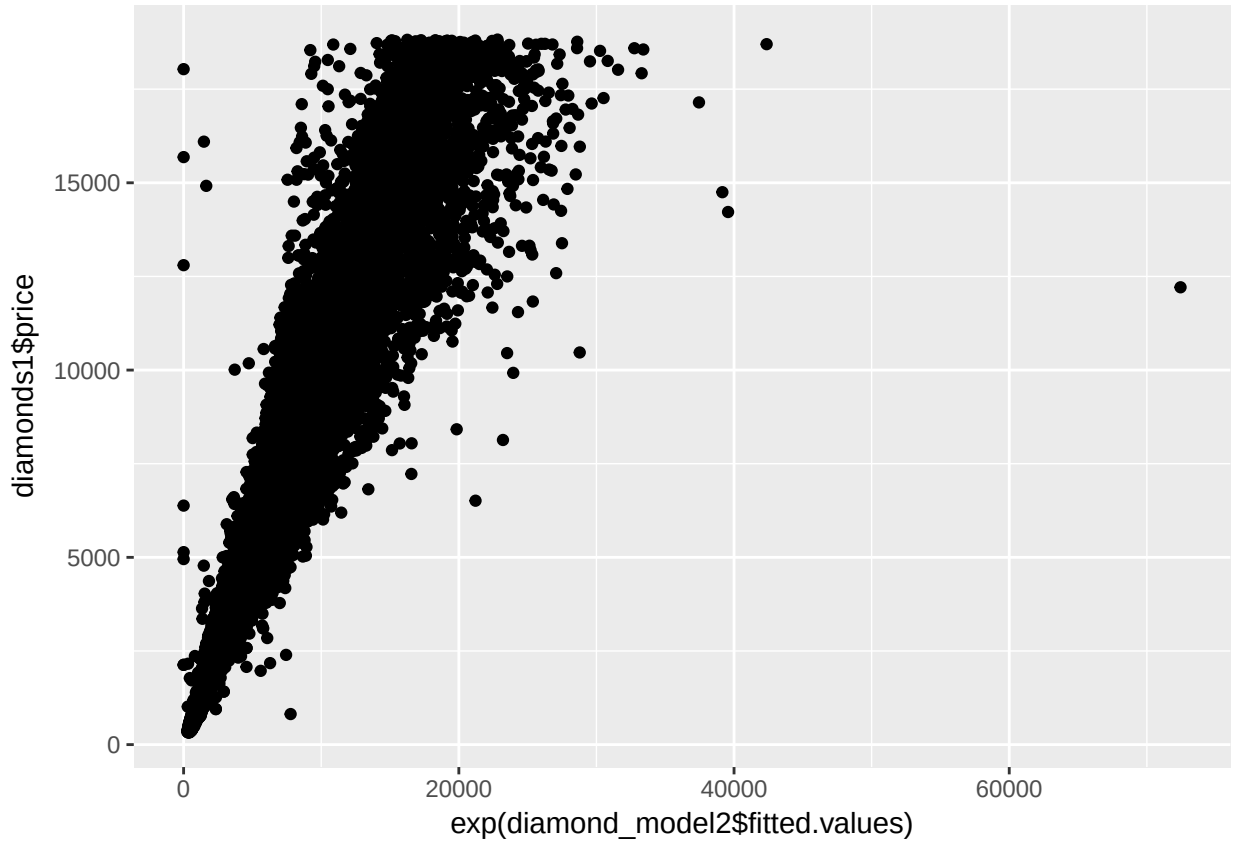
Visualize your best and your worst model's predictions on a true-predicted price scatterplot. Explain the differences.

Ans. My best model, diamond_model2's predicted prices when plotted against the true values of diamond prices give fairly accurate predictions for the data used to train the model. Since its R squared value is higher, it has a higher chance of better fitting the training data set.

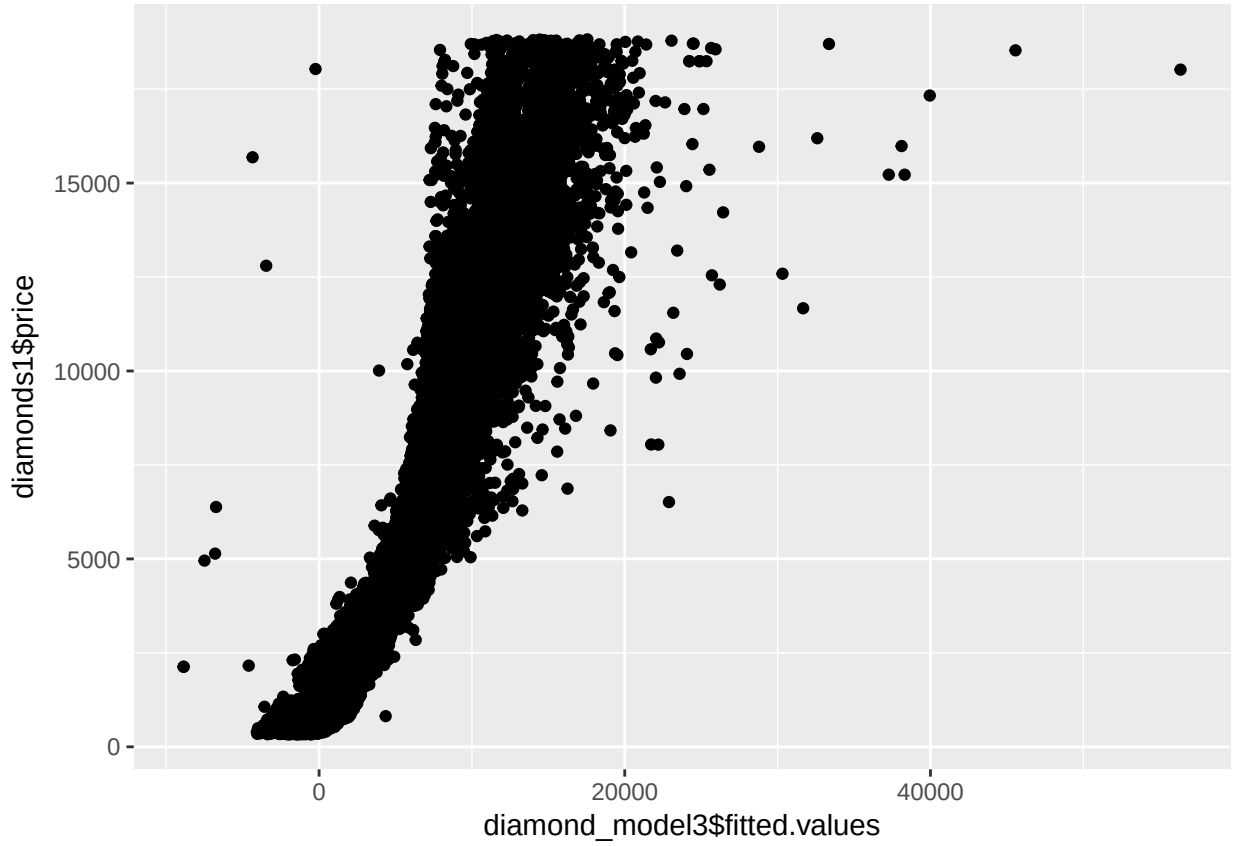
My worst model, diamond_model3's predicted prices when plotted against the true values of diamond prices give somewhat accurate predictions for the data used to train the model, but its accuracy seems to be clearly lower that diamond_model2. Since its R squared value is lower than diamond_model2, it has a lower chance of fitting the training data set well when compared to the previous model.

Both models have some very clear outliers which show large deviations from the true price.

```
#visualize best model
ggplot(mapping=aes(x=exp(diamond_model2$fitted.values),
                   y=diamonds1$price))+geom_point()
```




```
#visualize worst model
ggplot(mapping=aes(x=diamond_model3$fitted.values,
                   y=diamonds1$price))+geom_point()
```

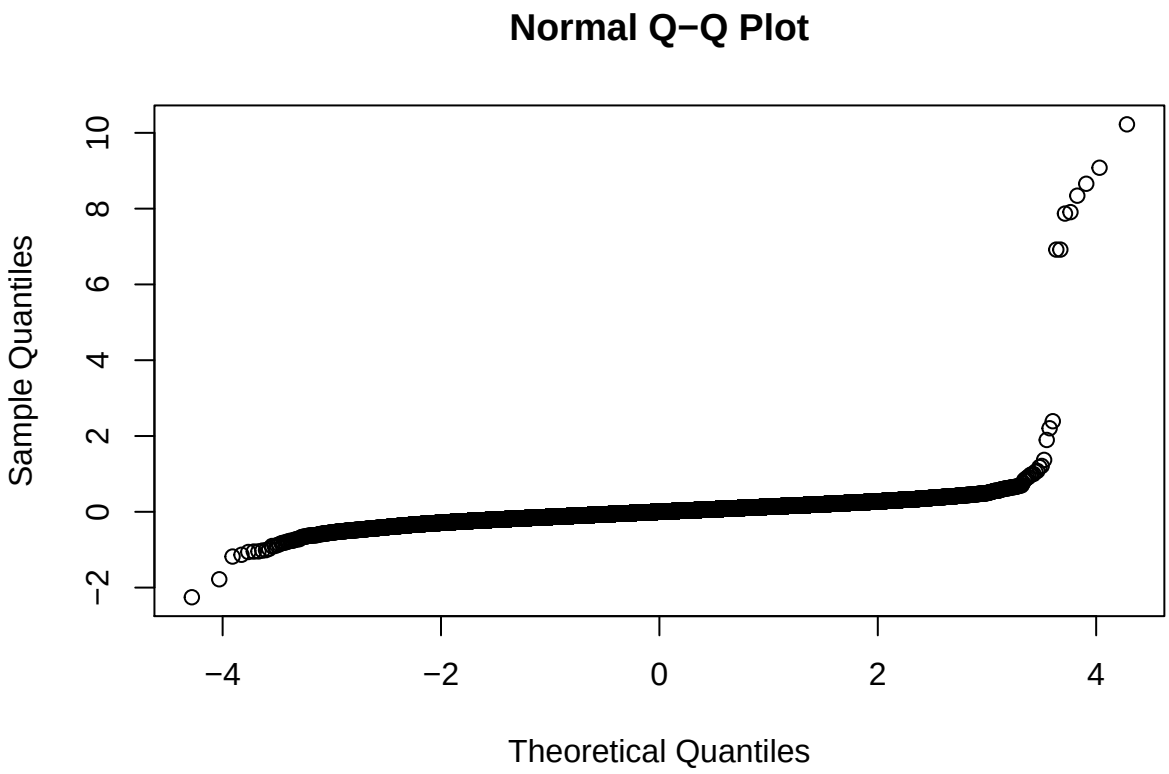


2.5 Residuals

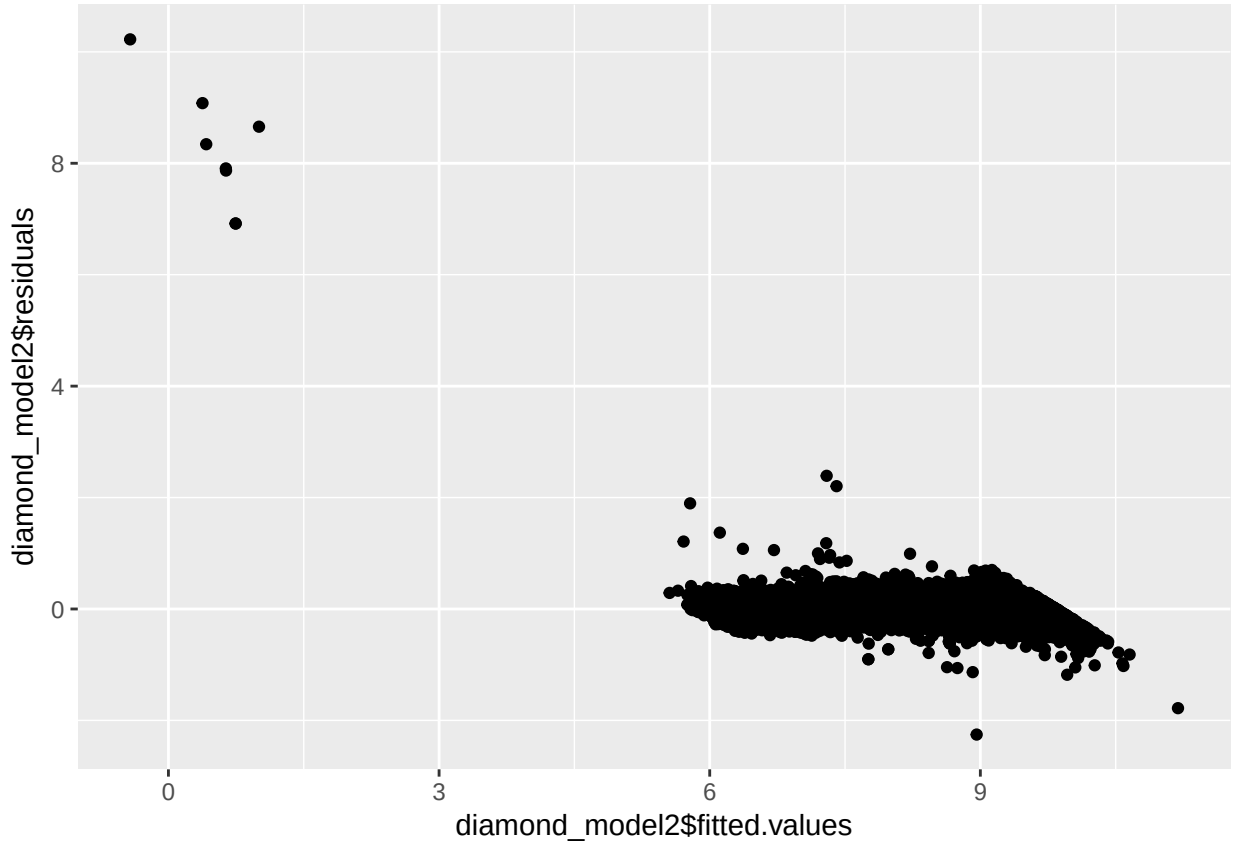
- Show the distribution of residuals (difference between the actual and predicted price). Does it look normal?
- Analyze a few largest outliers. Anything special with those diamonds? Ans. The QQplot of residuals seems to tend towards a normal distribution with the exception of certain outliers whose price predictions seem to greatly exceed or fall below the predicted price values.

Analyzing a few outliers, these are diamonds who price far exceeds the price predicted by our model, given a certain combination of predictors and our model. There are also diamonds whose price falls below our predictions to a great extent. This might be because our model is inadequete to explain these outliers because for example certain important features may not have been considered in the model, or existing features may not have been modeled properly.

```
#check distribution of residuals
qqnorm(diamond_model2$residuals)
```




```
ggplot(mapping=aes(x=diamond_model2$fitted.values,
                   y=diamond_model2$residuals))+geom_point()
```



3. How much work?

Tell us, roughly how many hours did you spend on this homework.

Ans. Around 6-7 hours.