Stat 425 Case Study 1

Carrie Mecca, Charlie Marcou, Jack Hanley, Jessie Bustin

2022-10-19

```r
# Libraries
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6     v purrr   0.3.4
## v tibble  3.1.8     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.1
## v readr   2.1.2     v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(faraway)
library(lmtest)
```

```
## Loading required package: zoo
##
## Attaching package: 'zoo'
##
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```r
library(MASS)
```

```
##
## Attaching package: 'MASS'
##
## The following object is masked from 'package:dplyr':
##
##     select
```

```r
# Load Data & Name Columns
data <- read_table("CDI.txt", col_names = FALSE)
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   X1 = col_double(),
```

```
##   X2 = col_character(),
##   X3 = col_character(),
##   X4 = col_double(),
##   X5 = col_double(),
##   X6 = col_double(),
##   X7 = col_double(),
##   X8 = col_double(),
##   X9 = col_double(),
##   X10 = col_double(),
##   X11 = col_double(),
##   X12 = col_double(),
##   X13 = col_double(),
##   X14 = col_double(),
##   X15 = col_double(),
##   X16 = col_double(),
##   X17 = col_double()
## )
```

```
data <- data %>%
  rename(id = X1,
         county = X2,
         state = X3,
         land_area = X4,
         total_pop = X5,
         pop_18to24 = X6,
         pop_over65 = X7,
         num_physicians = X8,
         num_hospital_beds = X9,
         serious_crimes = X10,
         highschool_rate = X11,
         bachelors_rate = X12,
         poverty_rate = X13,
         unemployment_rate = X14,
         per_capita_income = X15,
         total_personal_income = X16,
         region = X17)
```

```
# Check Variable Types
str(data)
```

```
## spec_tbl_df [440 x 17] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ id                   : num [1:440] 1 2 3 4 5 6 7 8 9 10 ...
##  $ county               : chr [1:440] "Los_Angeles" "Cook" "Harris" "San_Diego" ...
##  $ state                : chr [1:440] "CA" "IL" "TX" "CA" ...
##  $ land_area            : num [1:440] 4060 946 1729 4205 790 ...
##  $ total_pop            : num [1:440] 8863164 5105067 2818199 2498016 2410556 ...
##  $ pop_18to24           : num [1:440] 32.1 29.2 31.3 33.5 32.6 28.3 29.2 27.4 27.1 32.6 ...
##  $ pop_over65           : num [1:440] 9.7 12.4 7.1 10.9 9.2 12.4 12.5 12.5 13.9 8.2 ...
##  $ num_physicians       : num [1:440] 23677 15153 7553 5905 6062 ...
##  $ num_hospital_beds    : num [1:440] 27700 21550 12449 6179 6369 ...
##  $ serious_crimes       : num [1:440] 688936 436936 253526 173821 144524 ...
##  $ highschool_rate      : num [1:440] 70 73.4 74.9 81.9 81.2 63.7 81.5 70 65 77.1 ...
##  $ bachelors_rate       : num [1:440] 22.3 22.8 25.4 25.3 27.8 16.6 22.1 13.7 18.8 26.3 ...
```

```
##  $ poverty_rate         : num [1:440] 11.6 11.1 12.5 8.1 5.2 19.5 8.8 16.9 14.2 10.4 ...
##  $ unemployment_rate    : num [1:440] 8 7.2 5.7 6.1 4.8 9.5 4.9 10 8.7 6.1 ...
##  $ per_capita_income    : num [1:440] 20786 21729 19517 19588 24400 ...
##  $ total_personal_income: num [1:440] 184230 110928 55003 48931 58818 ...
##  $ region               : num [1:440] 4 2 3 4 4 1 4 2 3 3 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   X1 = col_double(),
##   ..   X2 = col_character(),
##   ..   X3 = col_character(),
##   ..   X4 = col_double(),
##   ..   X5 = col_double(),
##   ..   X6 = col_double(),
##   ..   X7 = col_double(),
##   ..   X8 = col_double(),
##   ..   X9 = col_double(),
##   ..   X10 = col_double(),
##   ..   X11 = col_double(),
##   ..   X12 = col_double(),
##   ..   X13 = col_double(),
##   ..   X14 = col_double(),
##   ..   X15 = col_double(),
##   ..   X16 = col_double(),
##   ..   X17 = col_double()
##   .. )
```

```r
# Check for NA's and INF's
complete_rows <- data[complete.cases(data), ]
nrow(data) == nrow(complete_rows)
```

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

```r
# Change region type because it is not numeric
data <- data %>%
  mutate(region = as.factor(region))

# Check correlation for numeric features
data %>%
  dplyr::select(-id, -county, -state, -region) %>%
  cor() %>%
  round(digits = 2)
```

```
##                      land_area total_pop pop_18to24 pop_over65 num_physicians
## land_area                 1.00      0.17      -0.05       0.01           0.08
## total_pop                 0.17      1.00       0.08      -0.03           0.94
## pop_18to24               -0.05      0.08       1.00      -0.62           0.12
## pop_over65                0.01     -0.03      -0.62       1.00           0.00
## num_physicians            0.08      0.94       0.12       0.00           1.00
## num_hospital_beds         0.07      0.92       0.07       0.05           0.95
## serious_crimes            0.13      0.89       0.09      -0.04           0.82
## highschool_rate          -0.10     -0.02       0.25      -0.27           0.00
## bachelors_rate           -0.14      0.15       0.46      -0.34           0.24
## poverty_rate              0.17      0.04       0.03       0.01           0.06
```

```
## unemployment_rate          0.20      0.01      -0.28       0.24           -0.05
## per_capita_income         -0.19      0.24      -0.03       0.02            0.32
## total_personal_income      0.13      0.99       0.07      -0.02            0.95
##                       num_hospital_beds serious_crimes highschool_rate
## land_area                          0.07           0.13           -0.10
## total_pop                          0.92           0.89           -0.02
## pop_18to24                         0.07           0.09            0.25
## pop_over65                         0.05          -0.04           -0.27
## num_physicians                     0.95           0.82            0.00
## num_hospital_beds                  1.00           0.86           -0.11
## serious_crimes                     0.86           1.00           -0.11
## highschool_rate                   -0.11          -0.11            1.00
## bachelors_rate                     0.10           0.08            0.71
## poverty_rate                       0.17           0.16           -0.69
## unemployment_rate                  0.01           0.04           -0.59
## per_capita_income                  0.19           0.12            0.52
## total_personal_income              0.90           0.84            0.04
##                       bachelors_rate poverty_rate unemployment_rate
## land_area                      -0.14         0.17              0.20
## total_pop                       0.15         0.04              0.01
## pop_18to24                      0.46         0.03             -0.28
## pop_over65                     -0.34         0.01              0.24
## num_physicians                  0.24         0.06             -0.05
## num_hospital_beds               0.10         0.17              0.01
## serious_crimes                  0.08         0.16              0.04
## highschool_rate                 0.71        -0.69             -0.59
## bachelors_rate                  1.00        -0.41             -0.54
## poverty_rate                   -0.41         1.00              0.44
## unemployment_rate              -0.54         0.44              1.00
## per_capita_income               0.70        -0.60             -0.32
## total_personal_income           0.22        -0.04             -0.03
##                       per_capita_income total_personal_income
## land_area                         -0.19                  0.13
## total_pop                          0.24                  0.99
## pop_18to24                        -0.03                  0.07
## pop_over65                         0.02                 -0.02
## num_physicians                     0.32                  0.95
## num_hospital_beds                  0.19                  0.90
## serious_crimes                     0.12                  0.84
## highschool_rate                    0.52                  0.04
## bachelors_rate                     0.70                  0.22
## poverty_rate                      -0.60                 -0.04
## unemployment_rate                 -0.32                 -0.03
## per_capita_income                  1.00                  0.35
## total_personal_income              0.35                  1.00
```

```r
# We have 4 variables highly correlated with total_pop so will transform them to per 100,000 people for
# Will drop total income because we already have per capita
# We can also drop the ID column
model_data <- data.frame(data %>%
  dplyr::select(-id, -total_personal_income) %>%
  mutate(hospital_beds_percap = num_hospital_beds / (total_pop / 100000),
         serious_crimes_percap = serious_crimes / (total_pop / 100000)) %>%
  dplyr::select(-num_hospital_beds, -serious_crimes))
```

```r
# Check county levels
model_data %>%
  group_by(county) %>%
  summarise(counts = n()) %>%
  summarise(min = min(counts), max(counts))
```

```
## # A tibble: 1 x 2
##     min `max(counts)`
##   <int>         <int>
## 1     1             7
```

```r
# Drop Counties and check states
model_data <- model_data %>%
  dplyr::select(-county)
model_data %>%
  group_by(state) %>%
  summarise(counts = n()) %>%
  arrange(counts) %>%
  head()
```

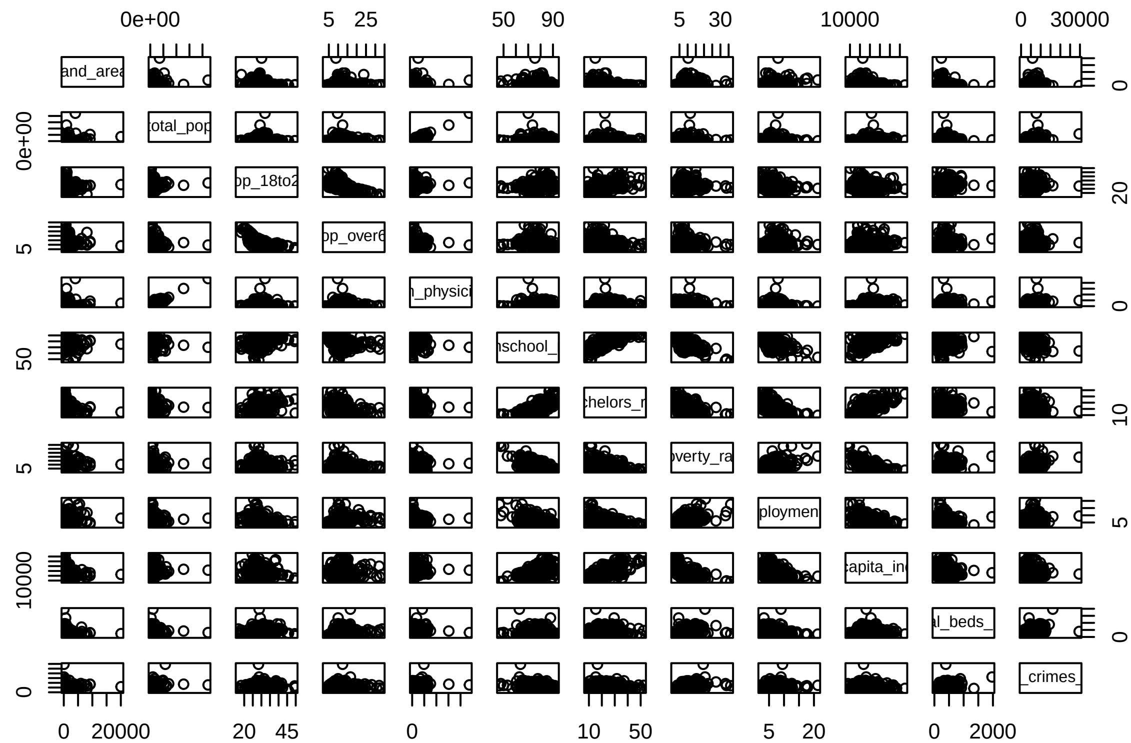
```
## # A tibble: 6 x 2
##   state counts
##   <chr>  <int>
## 1 DC         1
## 2 ID         1
## 3 MT         1
## 4 ND         1
## 5 SD         1
## 6 VT         1
```

```r
# Drop States and we will use regions
model_data <- model_data %>%
  dplyr::select(-state)

# Recheck correlation
model_data %>%
  dplyr::select(-region) %>%
  cor() %>%
  round(digits = 2)
```

```
##                         land_area total_pop pop_18to24 pop_over65 num_physicians
## land_area                    1.00      0.17      -0.05       0.01           0.08
## total_pop                    0.17      1.00       0.08      -0.03           0.94
## pop_18to24                  -0.05      0.08       1.00      -0.62           0.12
## pop_over65                   0.01     -0.03      -0.62       1.00           0.00
## num_physicians               0.08      0.94       0.12       0.00           1.00
## highschool_rate             -0.10     -0.02       0.25      -0.27           0.00
## bachelors_rate              -0.14      0.15       0.46      -0.34           0.24
## poverty_rate                 0.17      0.04       0.03       0.01           0.06
## unemployment_rate            0.20      0.01      -0.28       0.24          -0.05
## per_capita_income           -0.19      0.24      -0.03       0.02           0.32
```

```
## hospital_beds_percap      -0.14      0.02       0.03      0.25              0.19
## serious_crimes_percap      0.04      0.28       0.19     -0.07              0.31
##                        highschool_rate bachelors_rate poverty_rate
## land_area                        -0.10          -0.14         0.17
## total_pop                        -0.02           0.15         0.04
## pop_18to24                        0.25           0.46         0.03
## pop_over65                       -0.27          -0.34         0.01
## num_physicians                    0.00           0.24         0.06
## highschool_rate                   1.00           0.71        -0.69
## bachelors_rate                    0.71           1.00        -0.41
## poverty_rate                     -0.69          -0.41         1.00
## unemployment_rate                -0.59          -0.54         0.44
## per_capita_income                 0.52           0.70        -0.60
## hospital_beds_percap             -0.21          -0.05         0.37
## serious_crimes_percap            -0.23           0.04         0.47
##                        unemployment_rate per_capita_income hospital_beds_percap
## land_area                           0.20             -0.19                -0.14
## total_pop                           0.01              0.24                 0.02
## pop_18to24                         -0.28             -0.03                 0.03
## pop_over65                          0.24              0.02                 0.25
## num_physicians                     -0.05              0.32                 0.19
## highschool_rate                    -0.59              0.52                -0.21
## bachelors_rate                     -0.54              0.70                -0.05
## poverty_rate                        0.44             -0.60                 0.37
## unemployment_rate                   1.00             -0.32                -0.06
## per_capita_income                  -0.32              1.00                -0.05
## hospital_beds_percap               -0.06             -0.05                 1.00
## serious_crimes_percap               0.04             -0.08                 0.36
##                        serious_crimes_percap
## land_area                               0.04
## total_pop                               0.28
## pop_18to24                              0.19
## pop_over65                             -0.07
## num_physicians                          0.31
## highschool_rate                        -0.23
## bachelors_rate                          0.04
## poverty_rate                            0.47
## unemployment_rate                       0.04
## per_capita_income                      -0.08
## hospital_beds_percap                    0.36
## serious_crimes_percap                   1.00
```

```r
## Let's also look at a scattermatrix for correlation
model_data %>%
  dplyr::select(-region) %>%
  pairs()
```




```
# With a cutoff of 0.75 we have no highly correlated pairs other than population and our target

# With a cutoff of 0.7 we have the 2 following pairs:
# high_school_rate & bachelors_rate
# bachelors_rate & per_capita_income
```

```
# Start by just looking at total_pop for fun
slr_model <- lm(num_physicians~total_pop, data = model_data)
summary(slr_model)
```

```
##
## Call:
## lm(formula = num_physicians ~ total_pop, data = model_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1969.4  -209.2   -88.0    27.9  3928.7
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.106e+02  3.475e+01  -3.184  0.00156 **
## total_pop    2.795e-03  4.837e-05  57.793  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 610.1 on 438 degrees of freedom
## Multiple R-squared:  0.8841, Adjusted R-squared:  0.8838
## F-statistic:  3340 on 1 and 438 DF,  p-value: < 2.2e-16
```

```
# Full Model
mlr_full_model <- lm(num_physicians~., data = model_data)
summary(mlr_full_model)
```

```
##
## Call:
## lm(formula = num_physicians ~ ., data = model_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1904.06  -241.05   -30.21   185.69  2702.83
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)          -1.284e+03  7.154e+02  -1.795 0.073363 .
## land_area            -6.667e-02  1.734e-02  -3.844 0.000139 ***
## total_pop             2.708e-03  4.194e-05  64.571  < 2e-16 ***
## pop_18to24            1.860e+01  8.558e+00   2.174 0.030266 *
## pop_over65            8.983e+00  7.922e+00   1.134 0.257460
## highschool_rate      -1.372e+01  6.937e+00  -1.978 0.048613 *
## bachelors_rate        1.549e+01  7.499e+00   2.065 0.039493 *
## poverty_rate          2.493e+01  1.030e+01   2.421 0.015875 *
## unemployment_rate    -1.479e+01  1.381e+01  -1.071 0.284764
## per_capita_income     4.812e-02  1.218e-02   3.952 9.07e-05 ***
## region2              -3.810e+01  7.129e+01  -0.534 0.593340
## region3              -6.191e+01  7.379e+01  -0.839 0.401954
## region4               1.700e+02  8.910e+01   1.908 0.057117 .
## hospital_beds_percap  1.448e+00  1.511e-01   9.582  < 2e-16 ***
## serious_crimes_percap -2.903e-02  1.117e-02  -2.599 0.009668 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 462.3 on 425 degrees of freedom
## Multiple R-squared:  0.9354, Adjusted R-squared:  0.9333
## F-statistic: 439.6 on 14 and 425 DF,  p-value: < 2.2e-16
```

```
# Remove some fields based on EDA and full model summary
sub1_data <- model_data %>%
  dplyr::select(-pop_over65, -unemployment_rate, -highschool_rate)
mlr_sub1_model <- lm(num_physicians~., data = sub1_data)
anova(mlr_sub1_model, mlr_full_model)
```

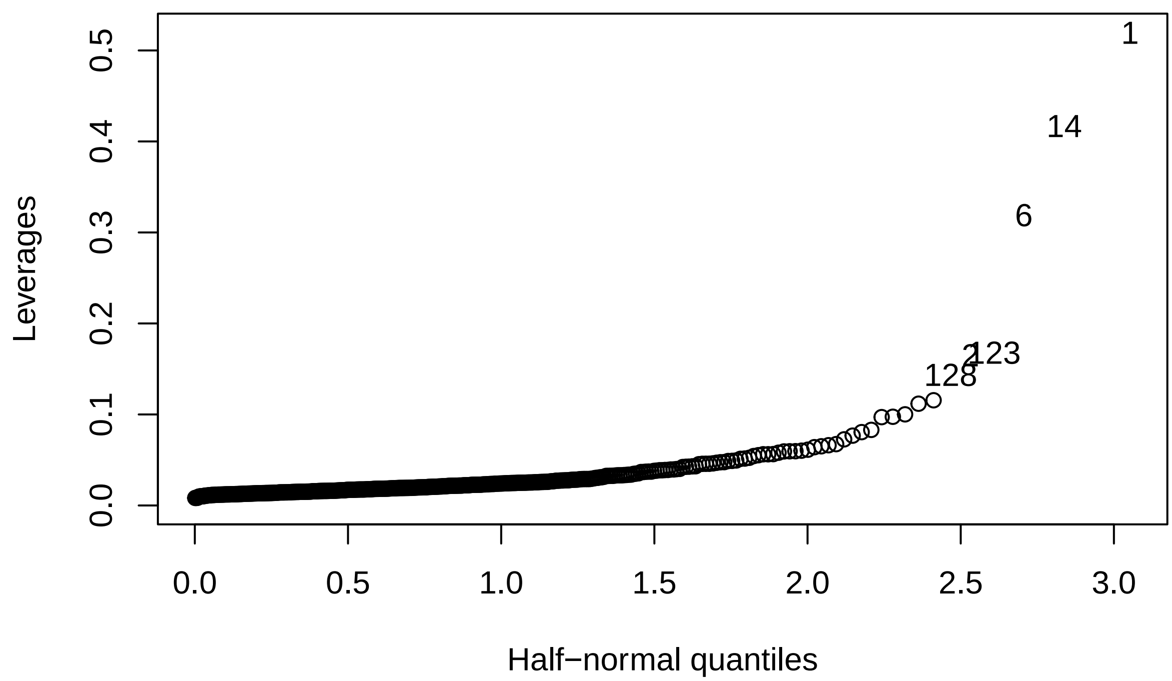
```
## Analysis of Variance Table
##
## Model 1: num_physicians ~ land_area + total_pop + pop_18to24 + bachelors_rate +
##     poverty_rate + per_capita_income + region + hospital_beds_percap +
##     serious_crimes_percap
## Model 2: num_physicians ~ land_area + total_pop + pop_18to24 + pop_over65 +
##     highschool_rate + bachelors_rate + poverty_rate + unemployment_rate +
```

```
##     per_capita_income + region + hospital_beds_percap + serious_crimes_percap
##   Res.Df      RSS Df Sum of Sq      F Pr(>F)
## 1    428 92073649
## 2    425 90825598  3   1248050 1.9467 0.1214
```

```
## Checking high-leverage points
leverages=lm.influence(mlr_sub1_model)$hat
head(leverages)
```

```
##          1          2          3          4          5          6
## 0.51964431 0.16519116 0.04881486 0.04305255 0.05235414 0.31889087
```

```
## Plot to help identify high leverage observations
halfnorm(leverages, nlab=6, labs=as.character(1:length(leverages)), ylab="Leverages")
```




```
## Determining leverages that exceed a 2p/n threshold
n = dim(model_data)[1]
p = length(variable.names(mlr_sub1_model))
leverages.high = leverages[leverages>(2*p/n)]
leverages.high
```

```
##          1          2          6          7         14         42         48
## 0.51964431 0.16519116 0.31889087 0.08057521 0.41713279 0.05953933 0.05626570
##         49         65         67         85         95        123        128
```

```
## 0.06403587 0.07669668 0.06125503 0.05643133 0.05796254 0.16774044 0.14336932
##        187        188        206        235        303        337        357
## 0.05963580 0.11180418 0.09706308 0.06618486 0.11559560 0.10010948 0.05624120
##        363        392        396        400        405        412        418
## 0.07263222 0.05937063 0.06016817 0.06739193 0.06502508 0.08302403 0.09746487
##        433
## 0.05524000
```

```
## We currently have many high leverage points (29), They represent only about 6.6% of the data.
## Before continuing, let us look at what high leverage points are good and bad
## Calculate IQR for number of physicians
IQR_y = IQR(model_data$num_physicians)
## Define range with its lower limit being (Q1 - IQR) and upper limit being (Q3 + IQR)
QT1_y = quantile(model_data$num_physicians,0.25)
QT3_y = quantile(model_data$num_physicians,0.75)
lower_lim_y = QT1_y - IQR_y
upper_lim_y = QT3_y + IQR_y
vector_lim_y = c(lower_lim_y,upper_lim_y)
## Range for number of physicians
vector_lim_y
```

```
##     25%     75%
## -670.50 1889.25
```

```
## Extract observations with high leverage points from the original data frame
highlev = model_data[leverages>2*p/n,]
## Select only the observations with leverage points outside the range
highlev_lower = highlev[highlev$num_physicians < vector_lim_y[1], ]
highlev_upper = highlev[highlev$num_physicians > vector_lim_y[2], ]
highlev2 = rbind(highlev_lower,highlev_upper)
## This is not outputting the observation number like her example did. It is probably because we're usi
##I switched model_data to be a dataframe which I believe solves this issue-Carrie
highlev2
```

```
##     land_area total_pop pop_18to24 pop_over65 num_physicians highschool_rate
## 1        4060   8863164       32.1        9.7          23677            70.0
## 2         946   5105067       29.2       12.4          15153            73.4
## 6          71   2300664       28.3       12.4           4861            63.7
## 7        9204   2122101       29.2       12.5           4320            81.5
## 14      20062   1418380       30.1        8.8           2463            75.4
## 48        495    757027       28.6       10.2           4635            90.6
## 67         59    663906       39.2       12.1           5674            75.4
## 95        181    496938       28.3       13.0           2500            68.1
## 123        62    396685       28.7       16.6           4189            62.8
##     bachelors_rate poverty_rate unemployment_rate per_capita_income region
## 1             22.3         11.6               8.0             20786      4
## 2             22.8         11.1               7.2             21729      2
## 6             16.6         19.5               9.5             16803      1
## 7             22.1          8.8               4.9             18042      4
## 14            14.9         10.3               8.0             16399      4
## 48            49.9          2.7               3.3             30081      3
## 67            27.7         14.4               8.7             23150      1
## 95            22.4         27.3               6.1             16578      3
```

```
## 123           15.3         20.6               9.0             18113      2
##     hospital_beds_percap serious_crimes_percap
## 1              312.5295              7773.026
## 2              422.1296              8558.869
## 6              388.6704             29598.672
## 7              287.6395              8368.735
## 14             236.1144              5859.502
## 48             199.0682              4590.853
## 67             926.9385             10364.118
## 95             808.5516             10914.440
## 123           1969.8249             16159.673
```

```
## Computing Studentized Residuals
mlr_sub1_model.resid = rstudent(mlr_sub1_model);
## Critical value with Bonferroni correction
## Note: Compare to t-value later at the alpha we choose
bonferroni_cv = qt(.05/(2*n), n-p-1)
bonferroni_cv
```

```
## [1] -3.895681
```

```
## Sorting residuals to find outliers
mlr_sub1_model.resid.sorted = sort(abs(mlr_sub1_model.resid), decreasing=TRUE)[1:10]
print(mlr_sub1_model.resid.sorted)
```

```
##       50       67       48       19        8       53       11       32
## 6.267323 5.790836 5.067090 4.513103 4.376934 4.185103 4.059736 3.088757
##       15        2
## 3.059593 2.722946
```

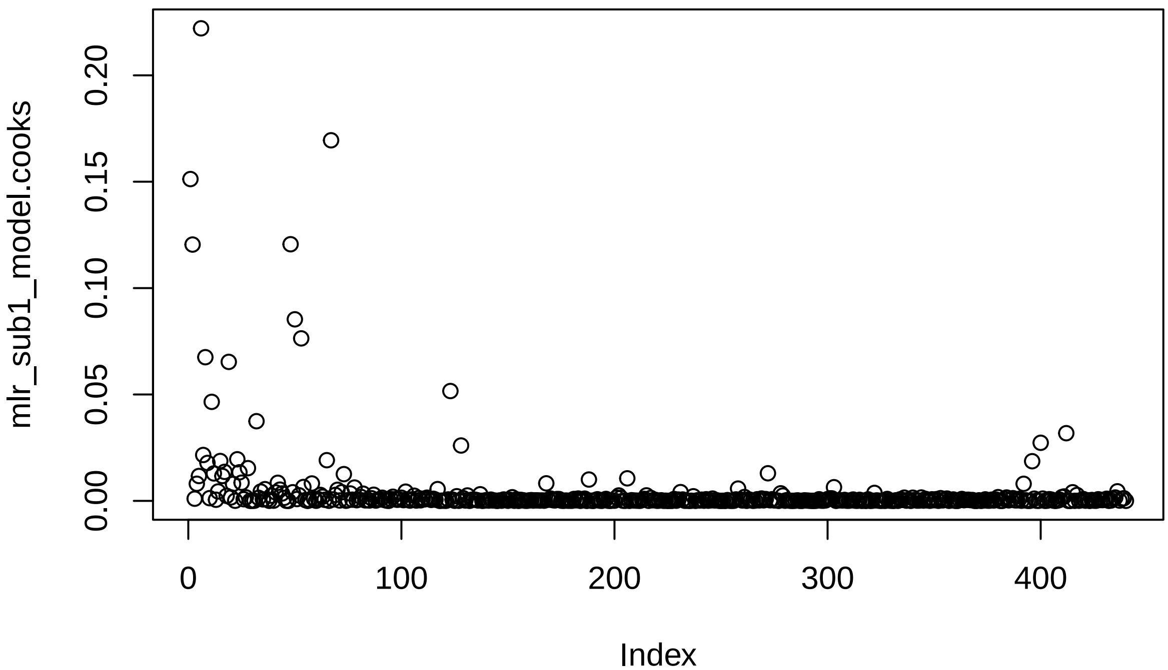
```
## Printing those above the value
## We can see observations 50, 67, 48, 19, 8, 53, and 11 are outliers.
mlr_sub1_model.outliers = mlr_sub1_model.resid.sorted[abs(mlr_sub1_model.resid.sorted) > abs(bonferroni_
print(mlr_sub1_model.outliers)
```

```
##       50       67       48       19        8       53       11
## 6.267323 5.790836 5.067090 4.513103 4.376934 4.185103 4.059736
```

```
## Finding high cook's distance observations
mlr_sub1_model.cooks = cooks.distance(mlr_sub1_model)
sort(mlr_sub1_model.cooks, decreasing = TRUE)[1:10]
```

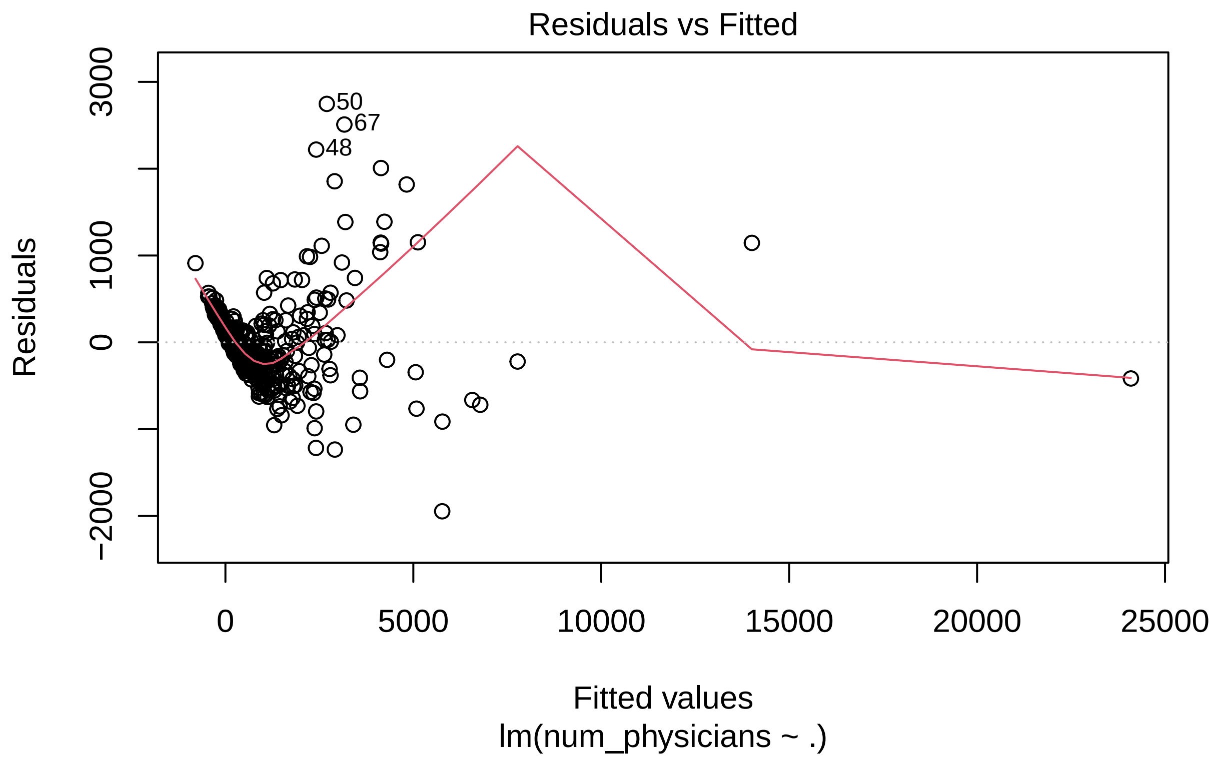
```
##          6         67          1         48          2         50         53
## 0.22208251 0.16946408 0.15123922 0.12061098 0.12045815 0.08532309 0.07635770
##          8         19        123
## 0.06751845 0.06531750 0.05162090
```

```
## Plotting cook's distance
plot(mlr_sub1_model.cooks)
```




```
## Some observations have high cook's distance relative to other observations, but none have cook's d >
```

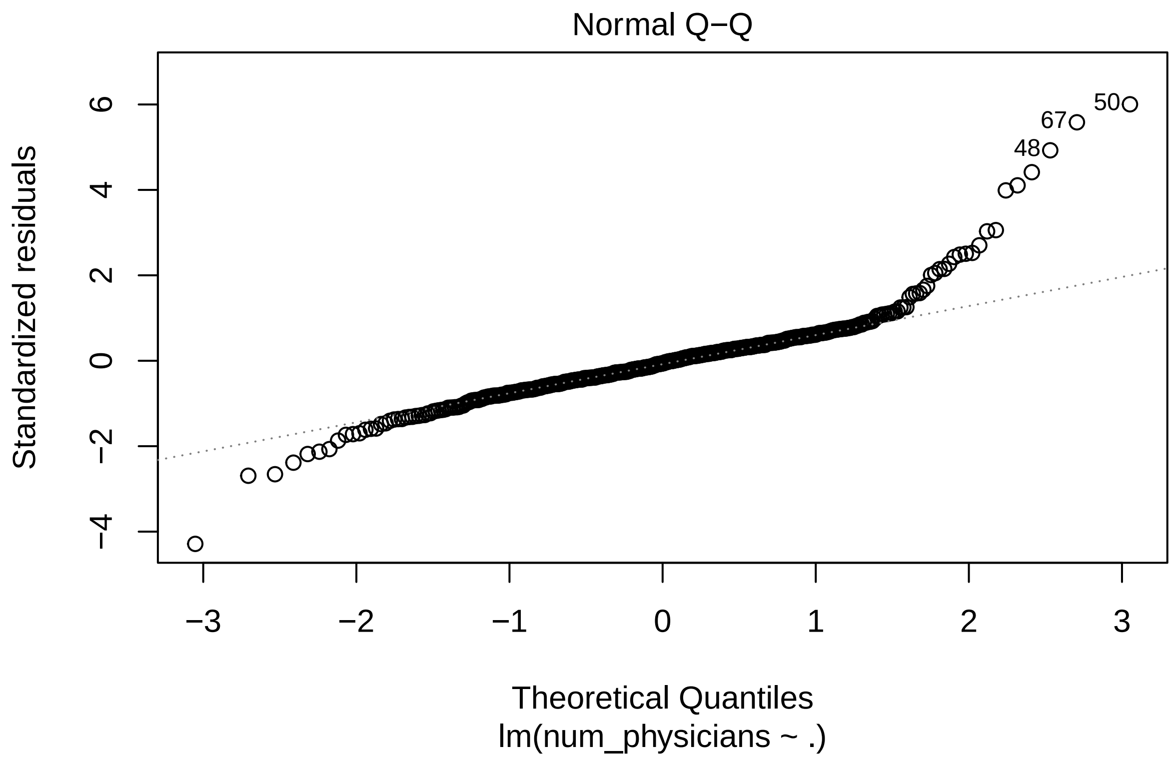
```
## Checking Constant Variance
plot(mlr_sub1_model, which=1)
```



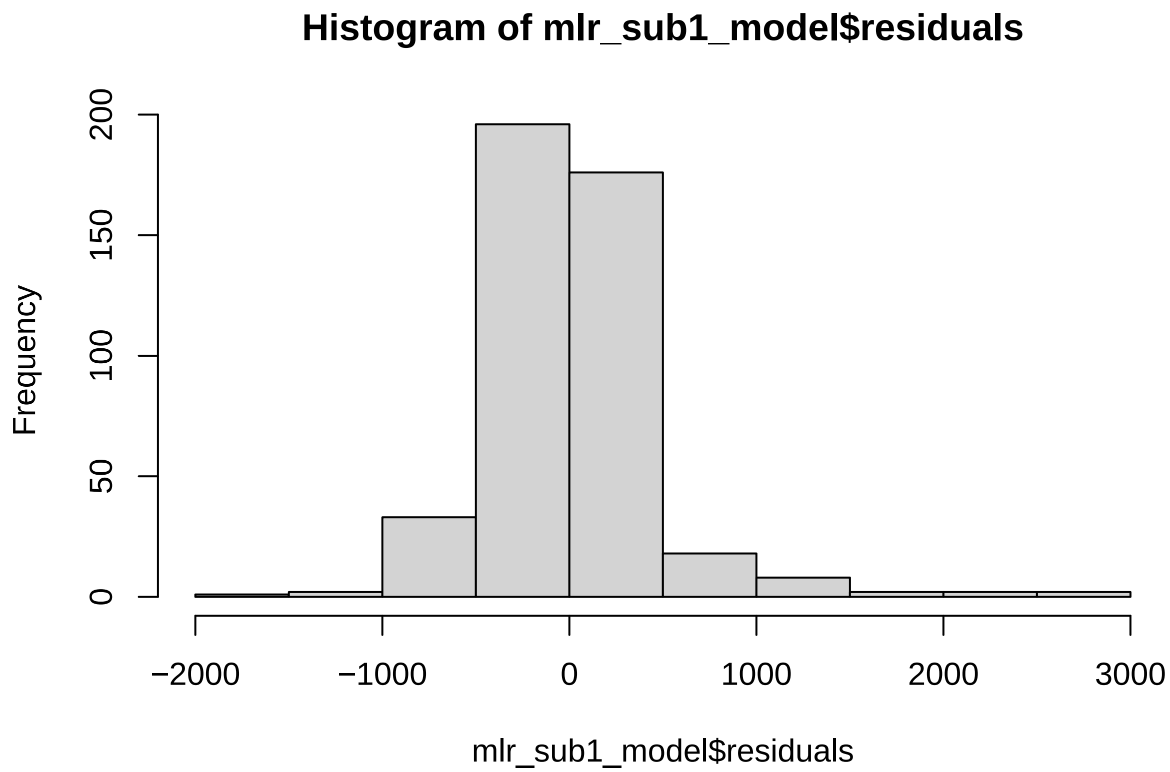

```
bptest(mlr_sub1_model)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  mlr_sub1_model
## BP = 77.914, df = 11, p-value = 3.73e-12
```

```
## Constant Variance seems to be violated
## Checking Normality
plot(mlr_sub1_model, which=2)
```




```
hist(mlr_sub1_model$residuals)
```



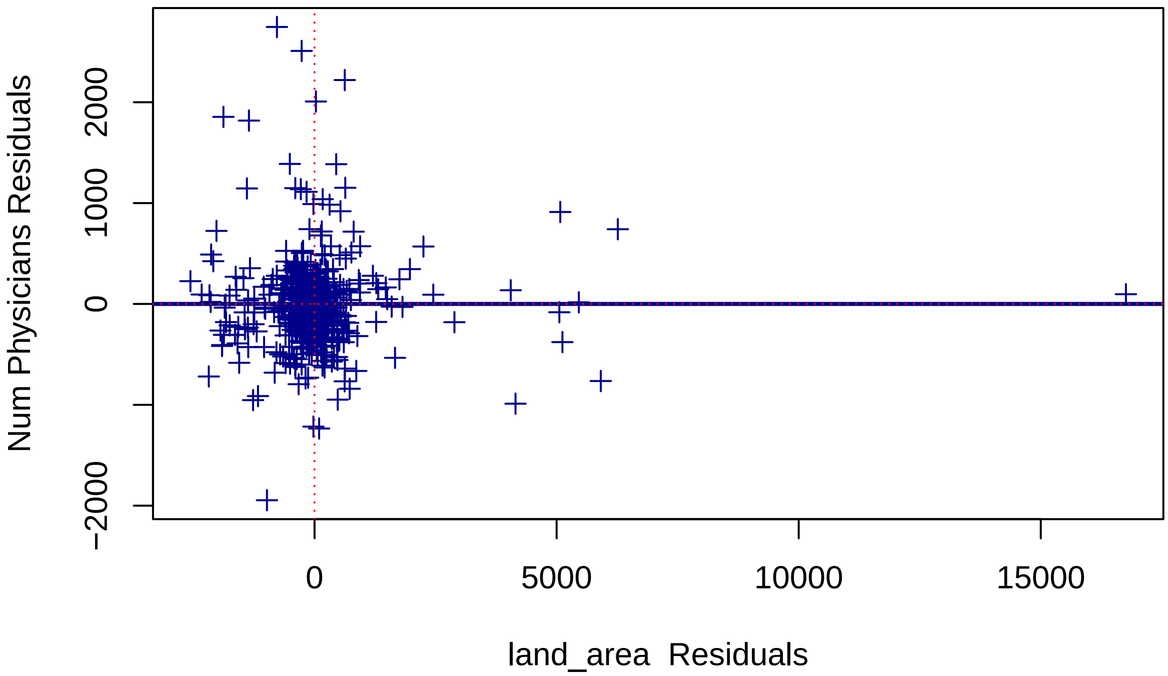

```r
### We can use the KS test to assess normality because n>50.
ks.test(mlr_sub1_model$residuals, 'pnorm')  ## We may want to check that this is the right syntax for t
```

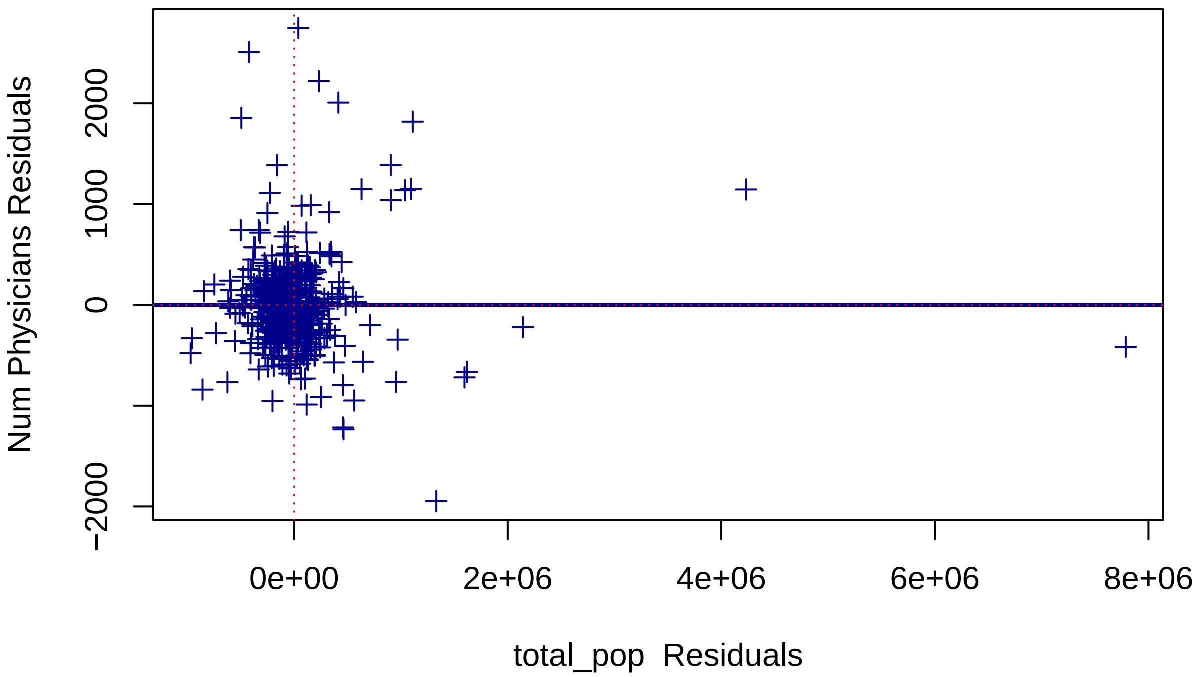
```
##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  mlr_sub1_model$residuals
## D = 0.52223, p-value < 2.2e-16
## alternative hypothesis: two-sided
```

```r
## Next step is to check linearity of each variable
```

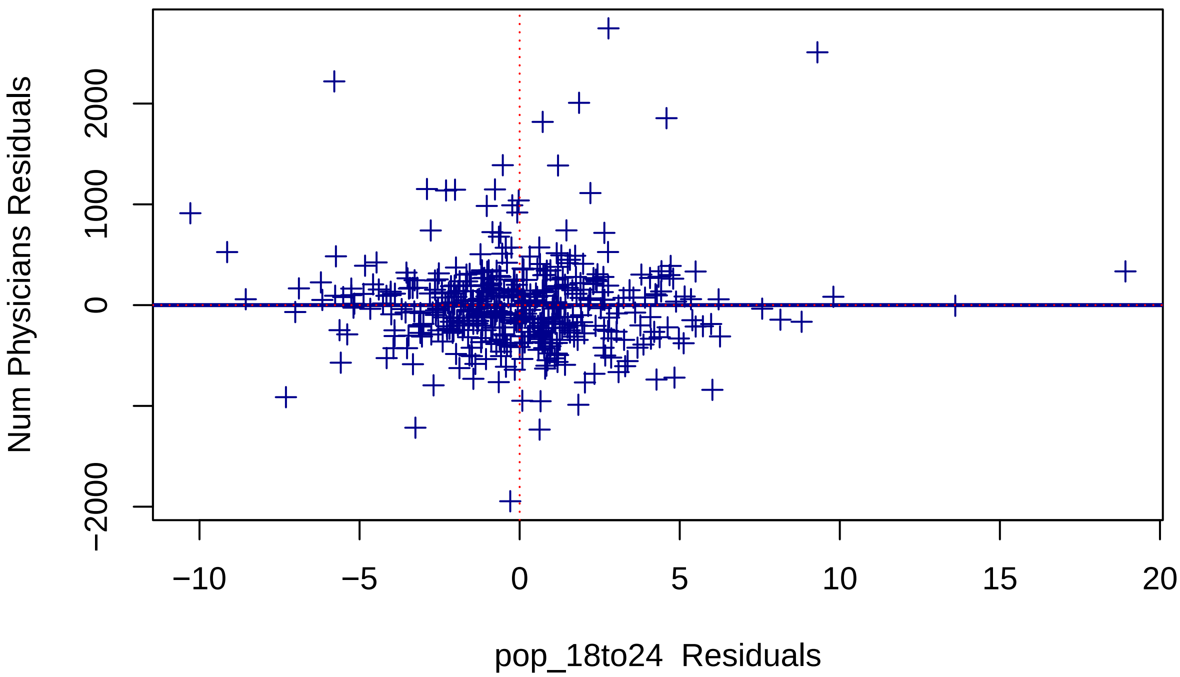
```r
checkLinearity <- function(var) {
  var_idx = which( colnames(sub1_data)==var )
  y.var = update(mlr_sub1_model, .~. -c(var_idx))$res
  #remove response + the variable itself
  x.var = lm(sub1_data[,var_idx] ~ . ,sub1_data[,-c(var_idx,4)])$res

  plot(x.var, y.var, xlab=paste(var," Residuals"), ylab="Num Physicians Residuals",   col='Darkblue', p
  abline(lm(y.var ~ x.var), col='Darkblue', lwd=2,xlim = c(quantile(x.var,.005),quantile(x.var,.995)))
  abline(v = 0, col="red", lty=3)
  abline(h = 0, col="red", lty=3)
}
predictors = names(sub1_data)
#remove the response variable (and region since it's a factor (?))
```

```r
predictors = predictors[!(predictors %in% c("num_physicians","region"))]
#check linearity for each predictor
for (var in predictors) {
  checkLinearity(var)
}
```

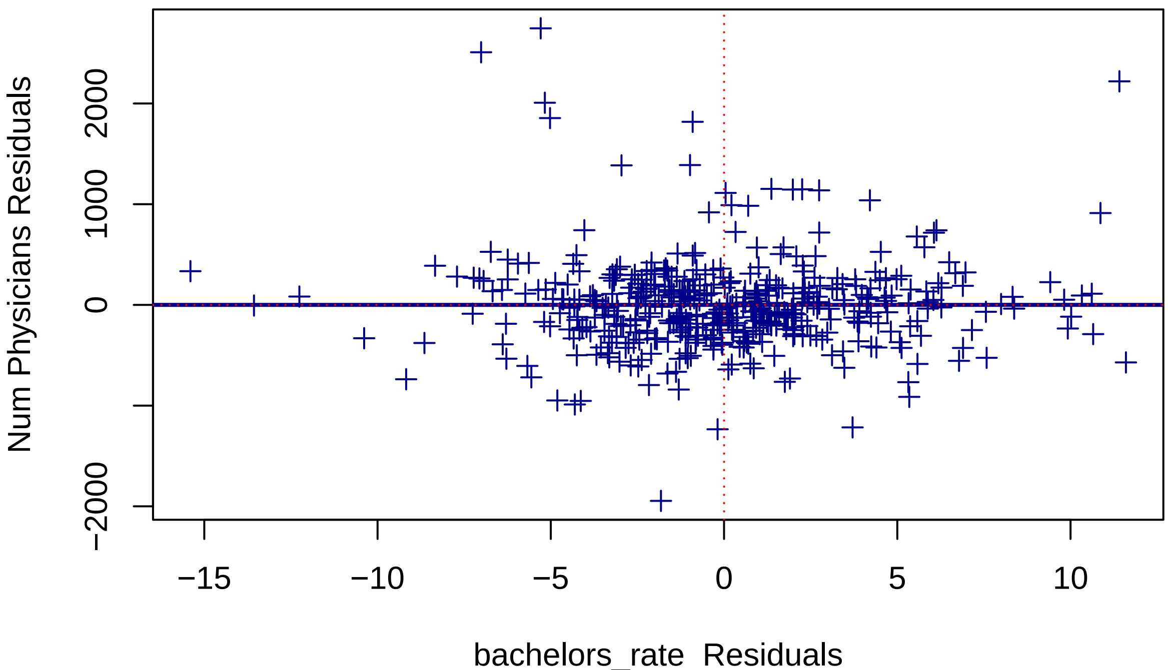




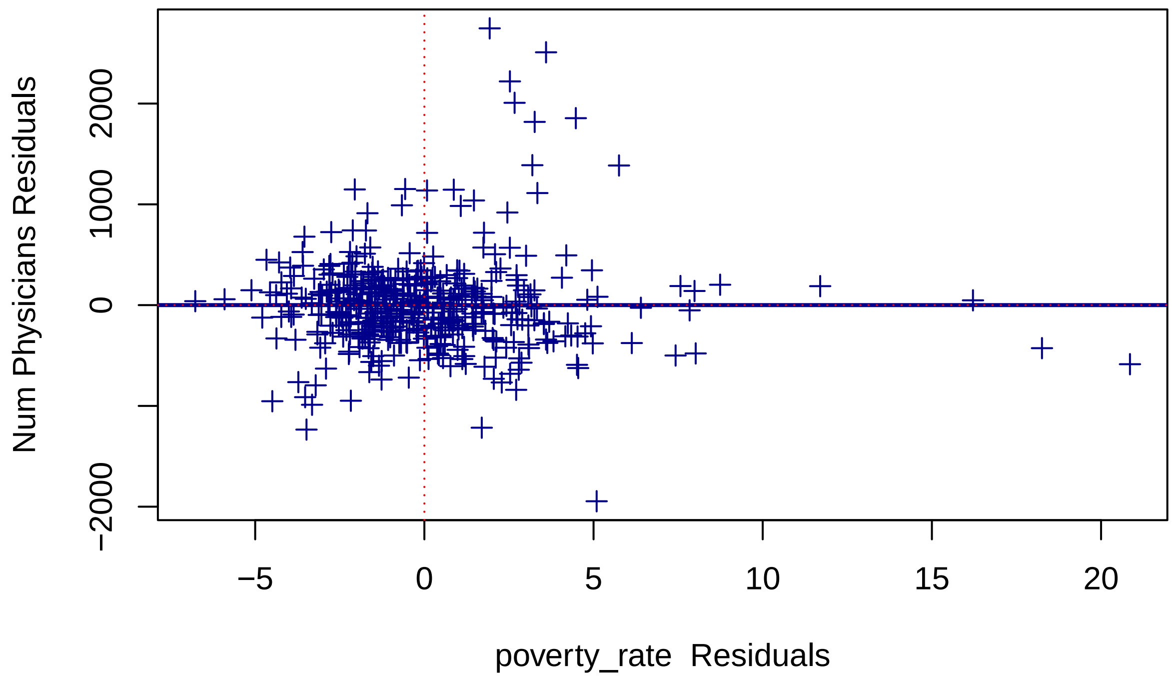
Num Physicians Residuals
2000
1000
0
−2000
0e+00
2e+06
4e+06
6e+06
8e+06
total_pop Residuals

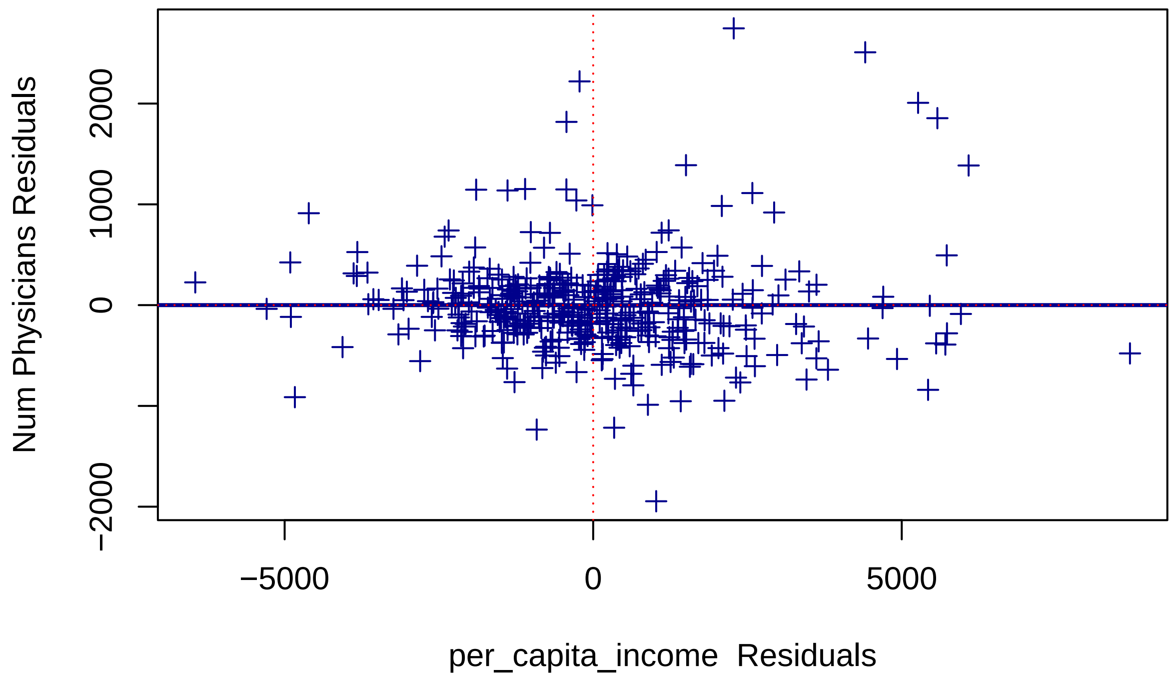


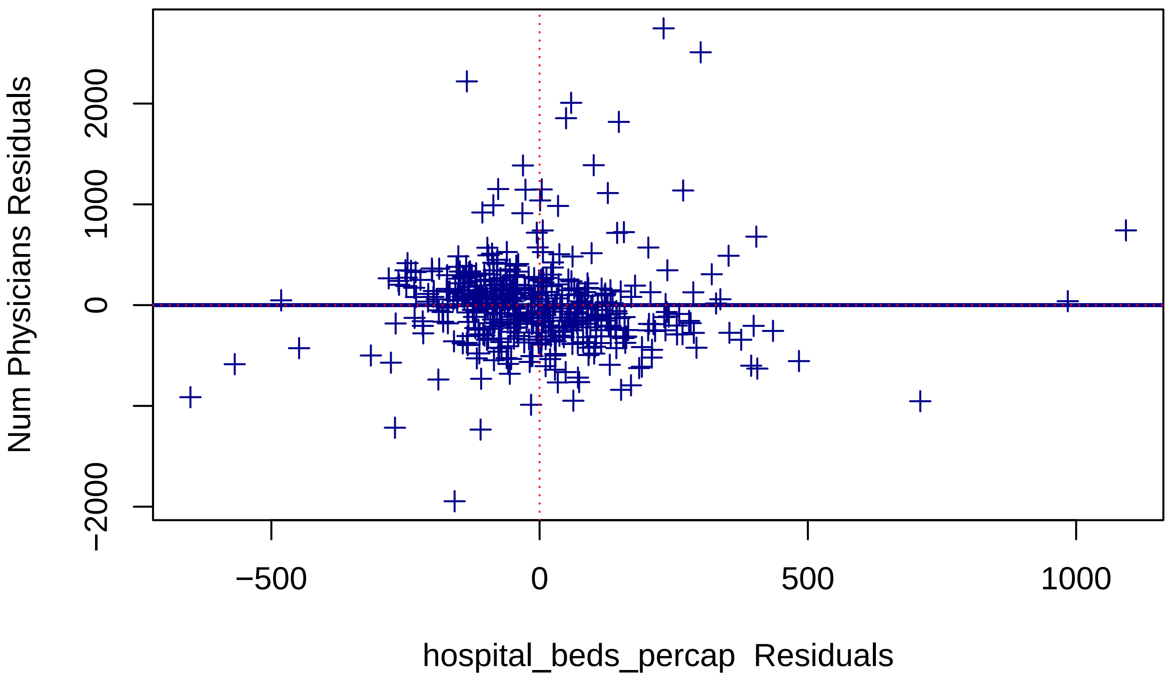
Num Physicians Residuals
2000
1000
0
−2000
−10
−5
0
5
10
15
20
pop_18to24 Residuals

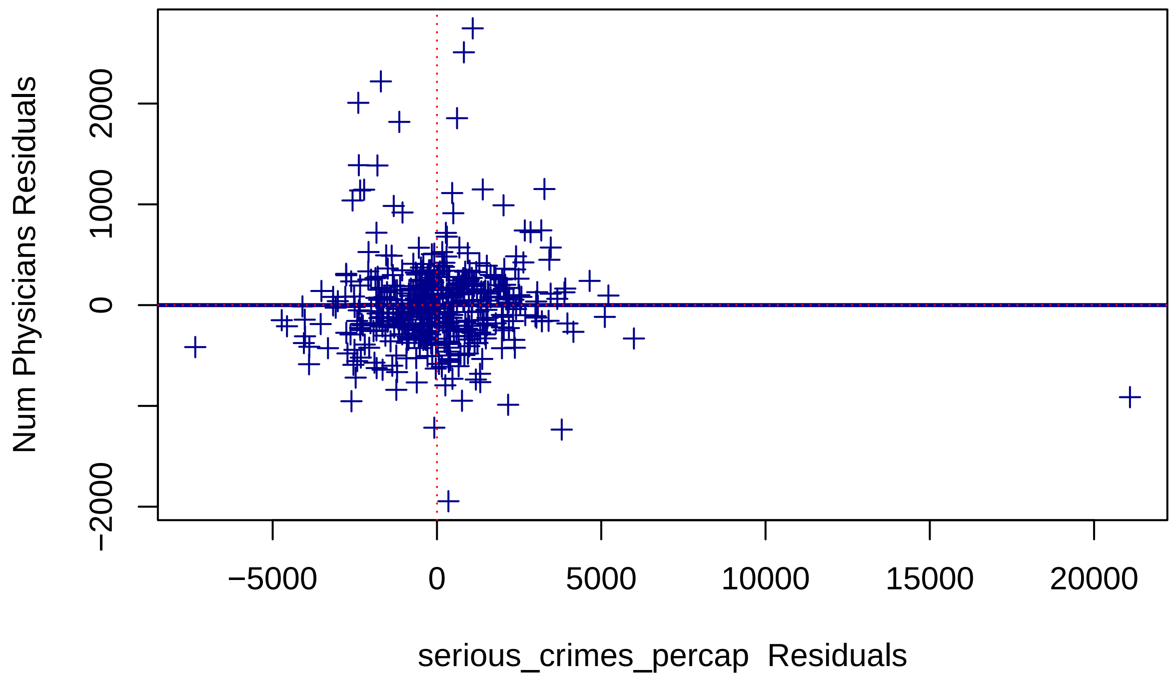


Num Physicians Residuals
2000
1000
0
−2000
−15
−10
−5
0
5
10
bachelors_rate Residuals

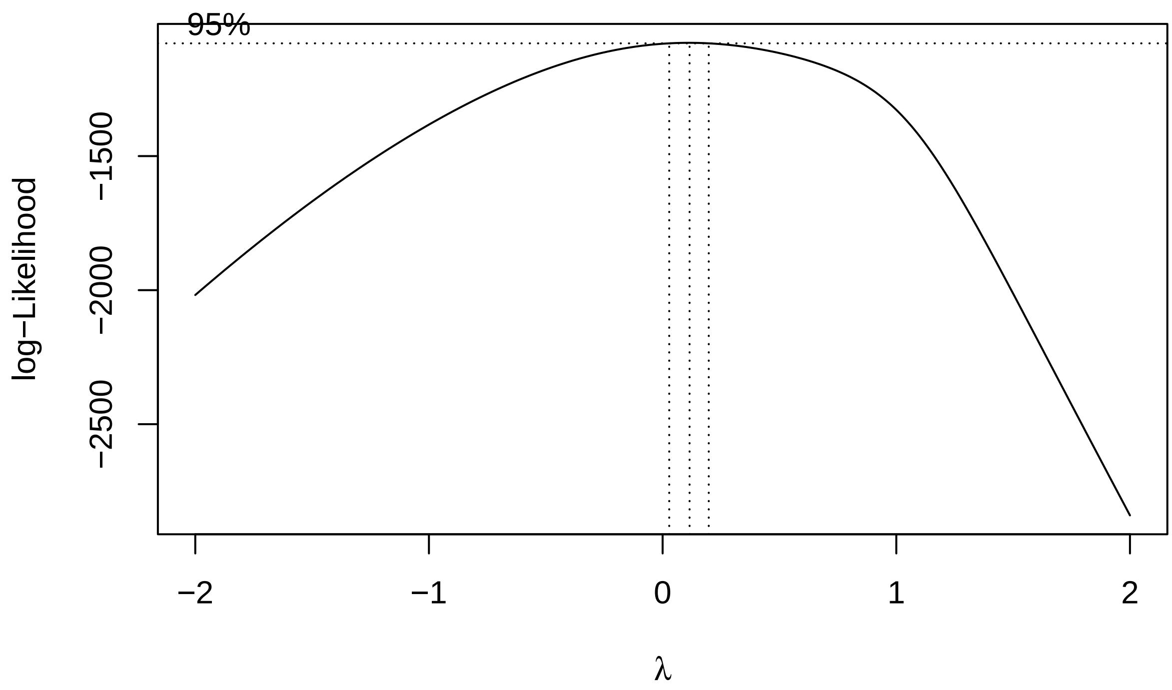

Num Physicians Residuals
2000
1000
0
−2000
−5
0
5
10
15
20
poverty_rate Residuals


Num Physicians Residuals
-2000
0
1000
2000
-5000
0
5000
per_capita_income Residuals


Num Physicians Residuals
2000
1000
0
−2000
−500
0
500
1000
hospital_beds_percap Residuals



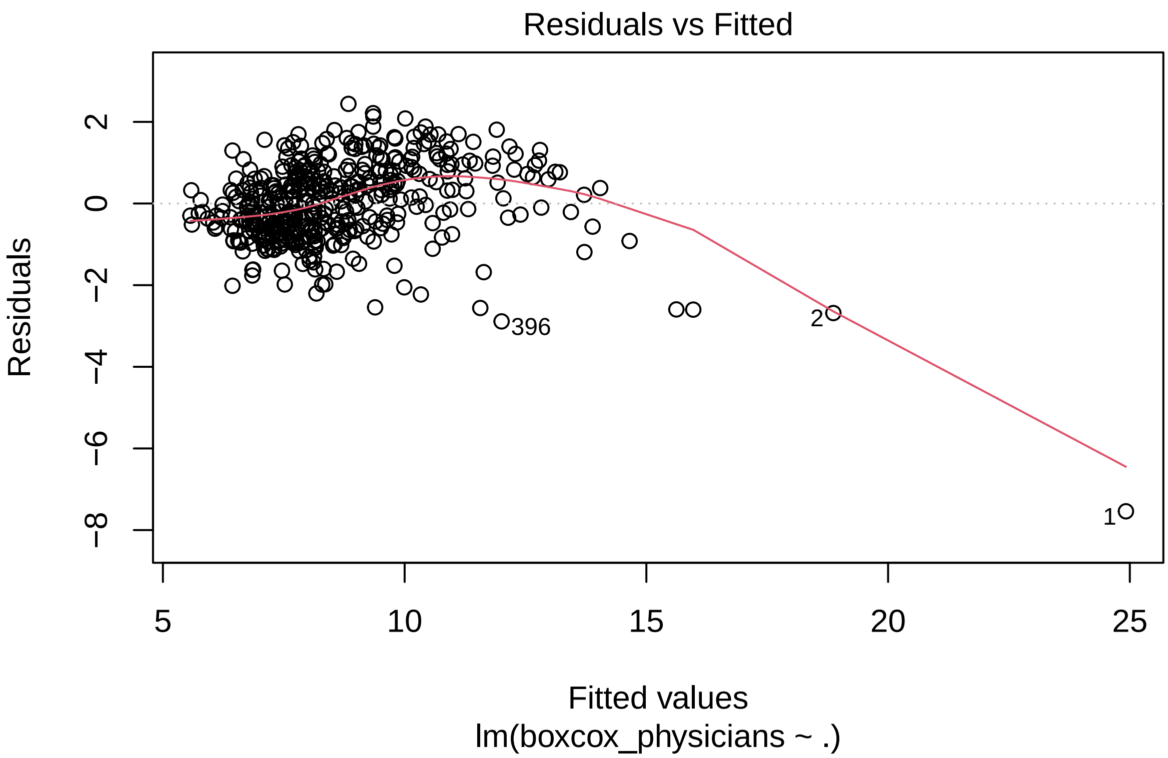

```
# log transform target - Didn't help
sub2_data <- sub1_data %>%
  mutate(log_physicians = log(num_physicians))
mlr_sub2_model <- lm(log_physicians~., data = sub2_data)

# Checking Box Cox
physician.transformation = boxcox(mlr_sub1_model, lambda=seq(-2,2, length=400))
```




```
lambda <- physician.transformation$x[which.max(physician.transformation$y)]
lambda
```

```
## [1] 0.1152882
```

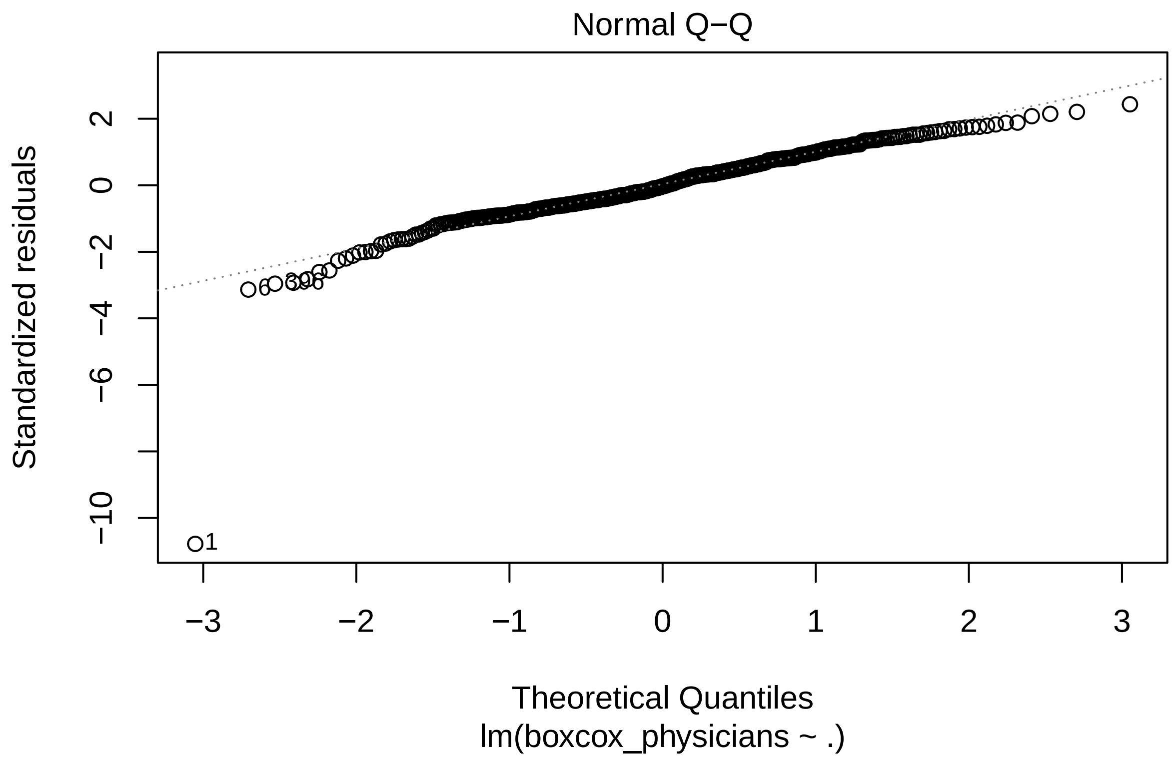
```
# Using 0.1 for box cox - Didn't help
lambda <- 0.1
sub2_data <- sub1_data %>%
  mutate(boxcox_physicians = (num_physicians^lambda - 1)/ lambda)
mlr_sub2_model <- lm(boxcox_physicians~., data = sub2_data)

## Checking Constant Variance
plot(mlr_sub2_model, which=1)
```




```
library(lmtest)
bptest(mlr_sub2_model)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  mlr_sub2_model
## BP = 231.29, df = 12, p-value < 2.2e-16
```

```
## Constant Variance seems to be violated
## Checking Normality
plot(mlr_sub2_model, which=2)
```

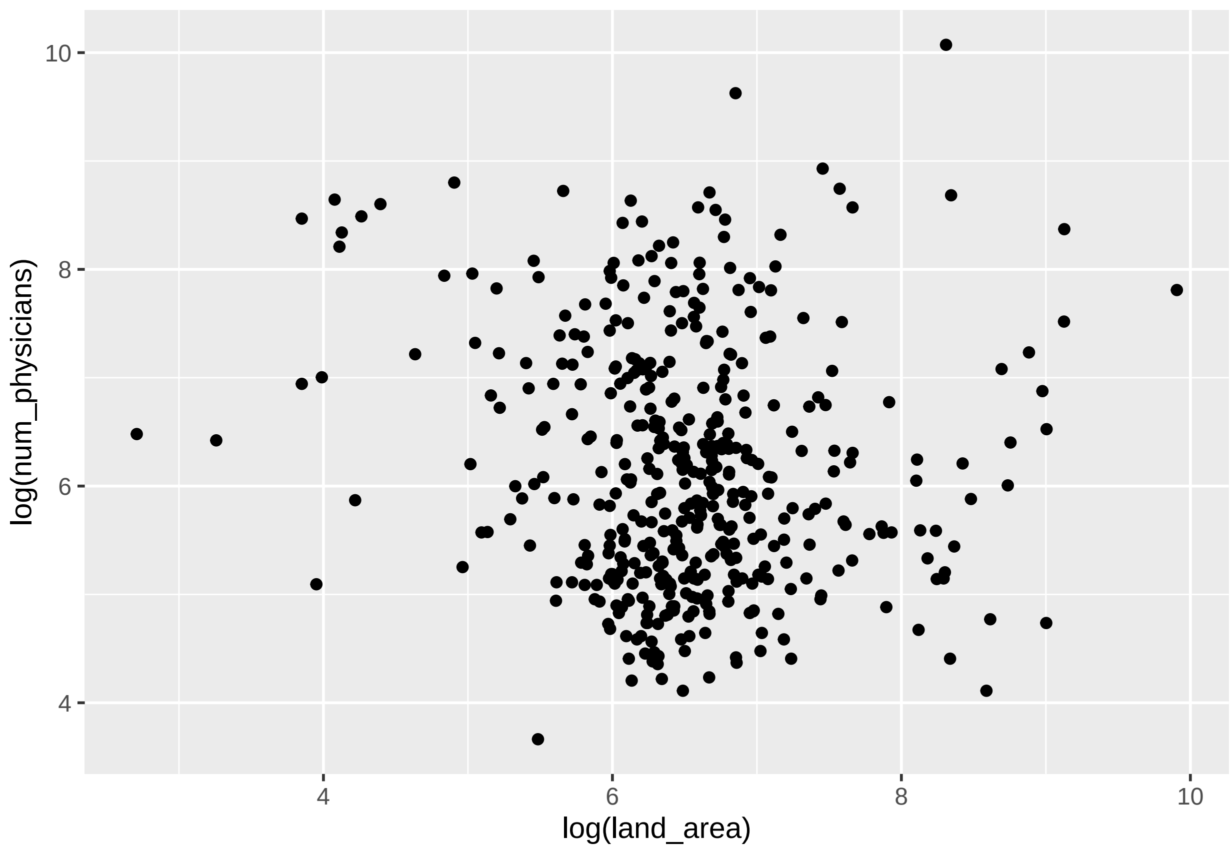



```
hist(mlr_sub2_model$residuals)
```

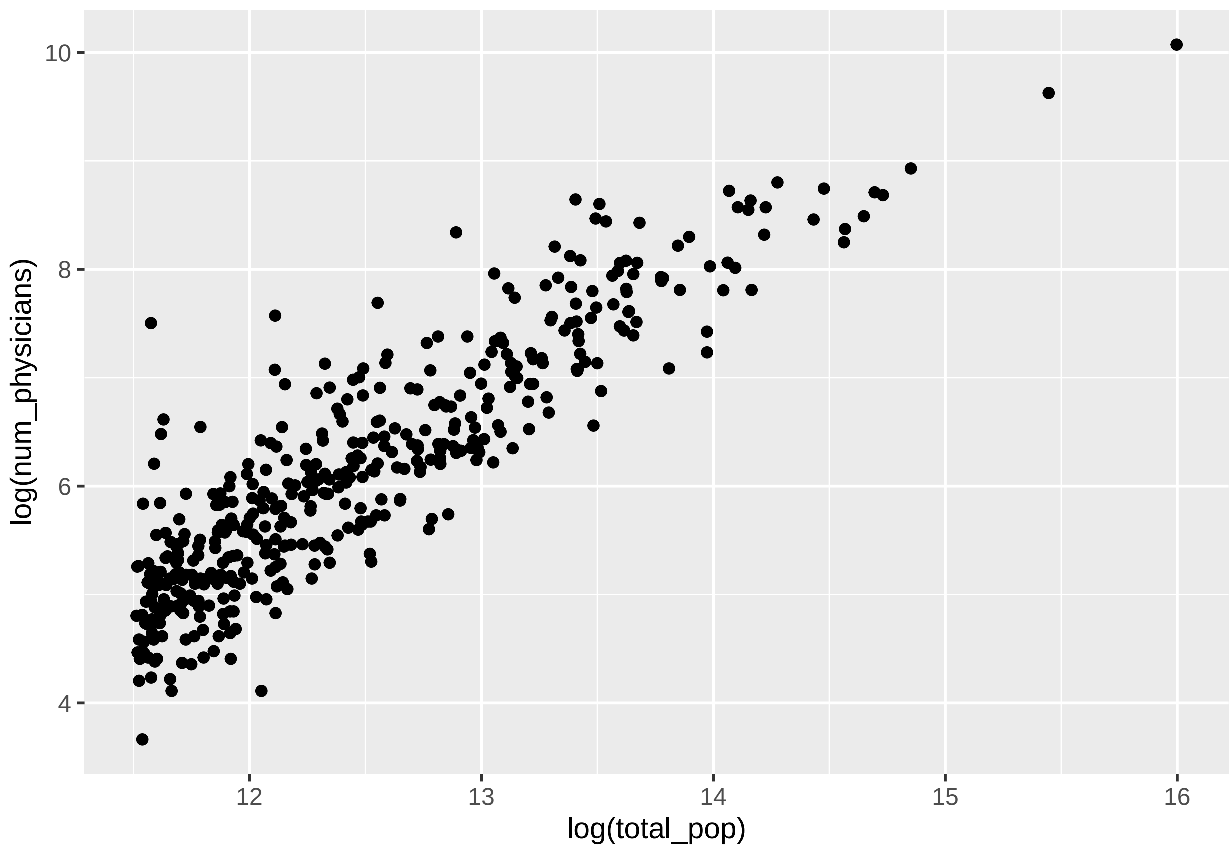

```r
### We can use the KS test to assess normality because n>50.
ks.test(mlr_sub2_model$residuals, 'pnorm')
```

```
##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  mlr_sub2_model$residuals
## D = 0.055808, p-value = 0.129
## alternative hypothesis: two-sided
```

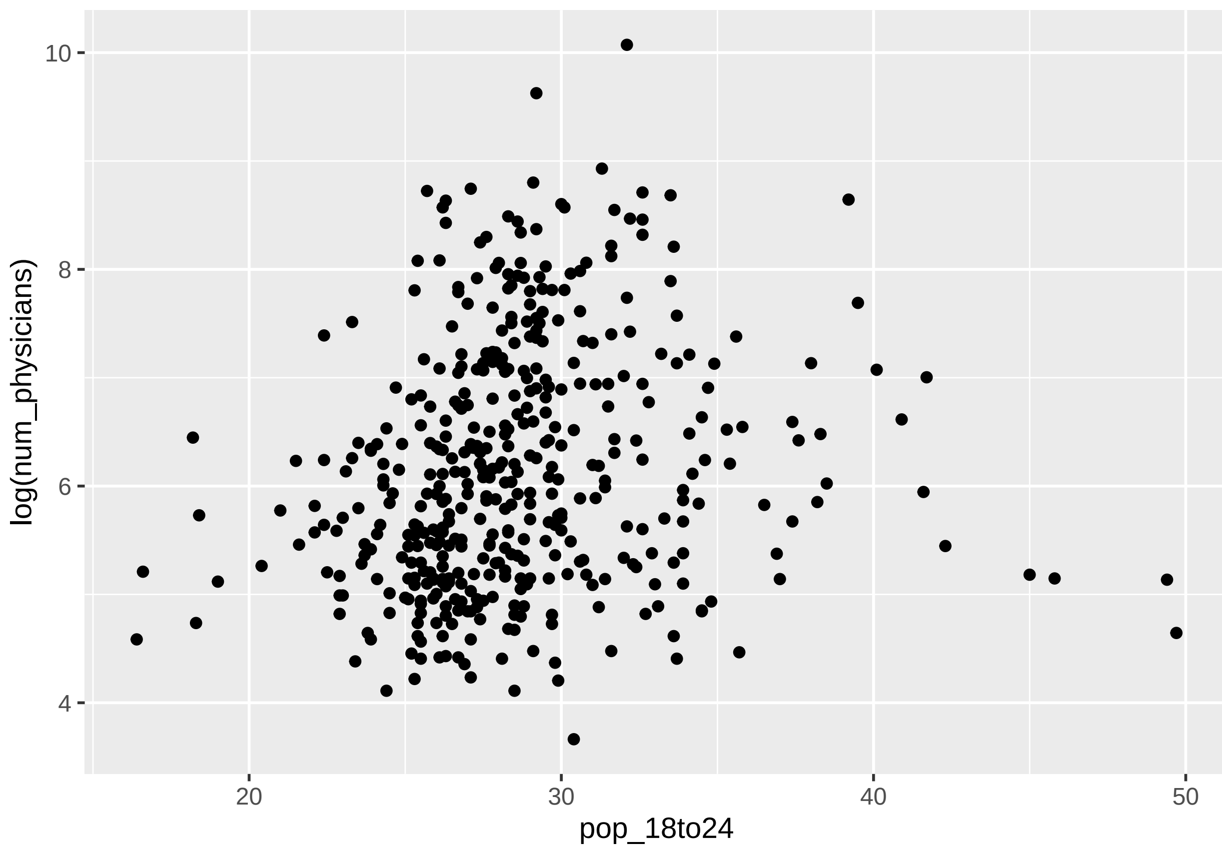
```r
# Let's look at each variable and the target graphically
# Need a log transformation on land_area
model_data %>% ggplot(aes(x = log(land_area), y = log(num_physicians))) +
  geom_point()
```



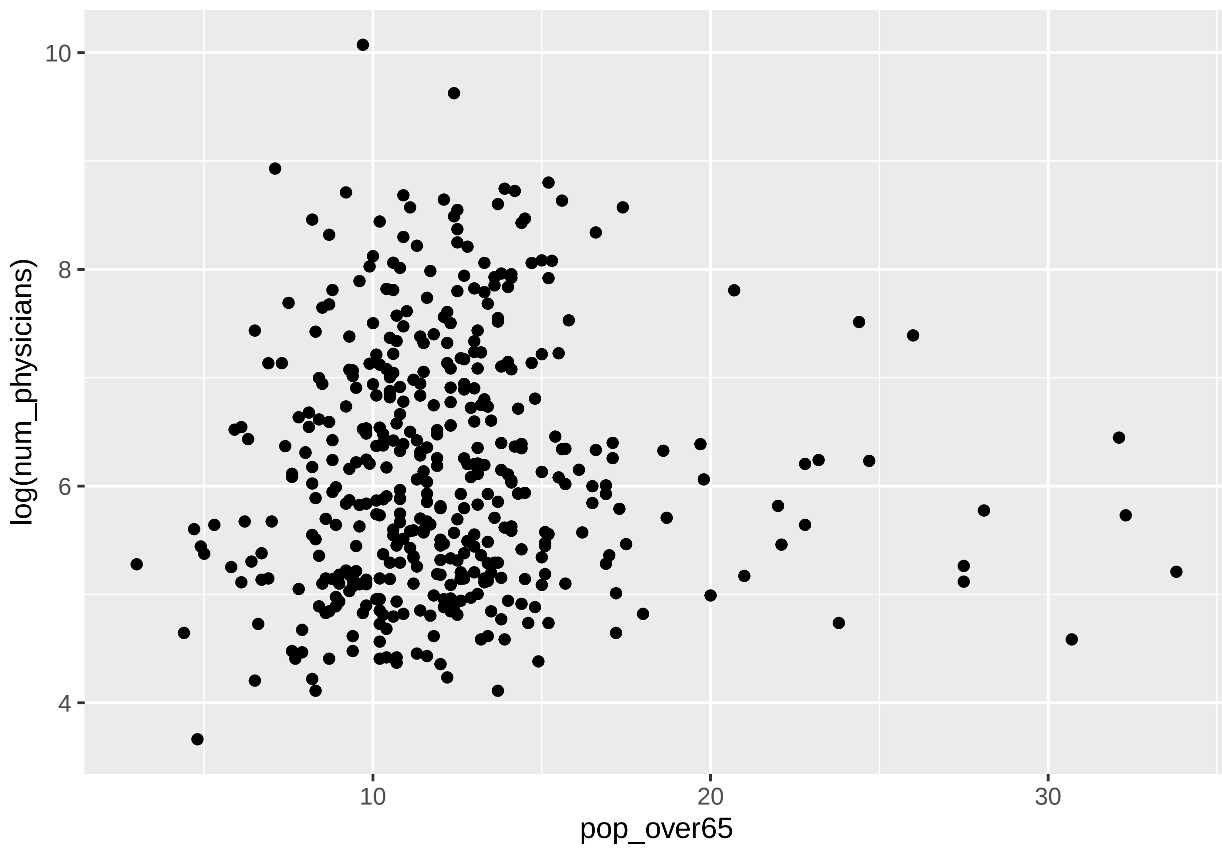

```
# log of total_pop will do wonders!!!
model_data %>% ggplot(aes(x = log(total_pop), y = log(num_physicians))) +
  geom_point()
```



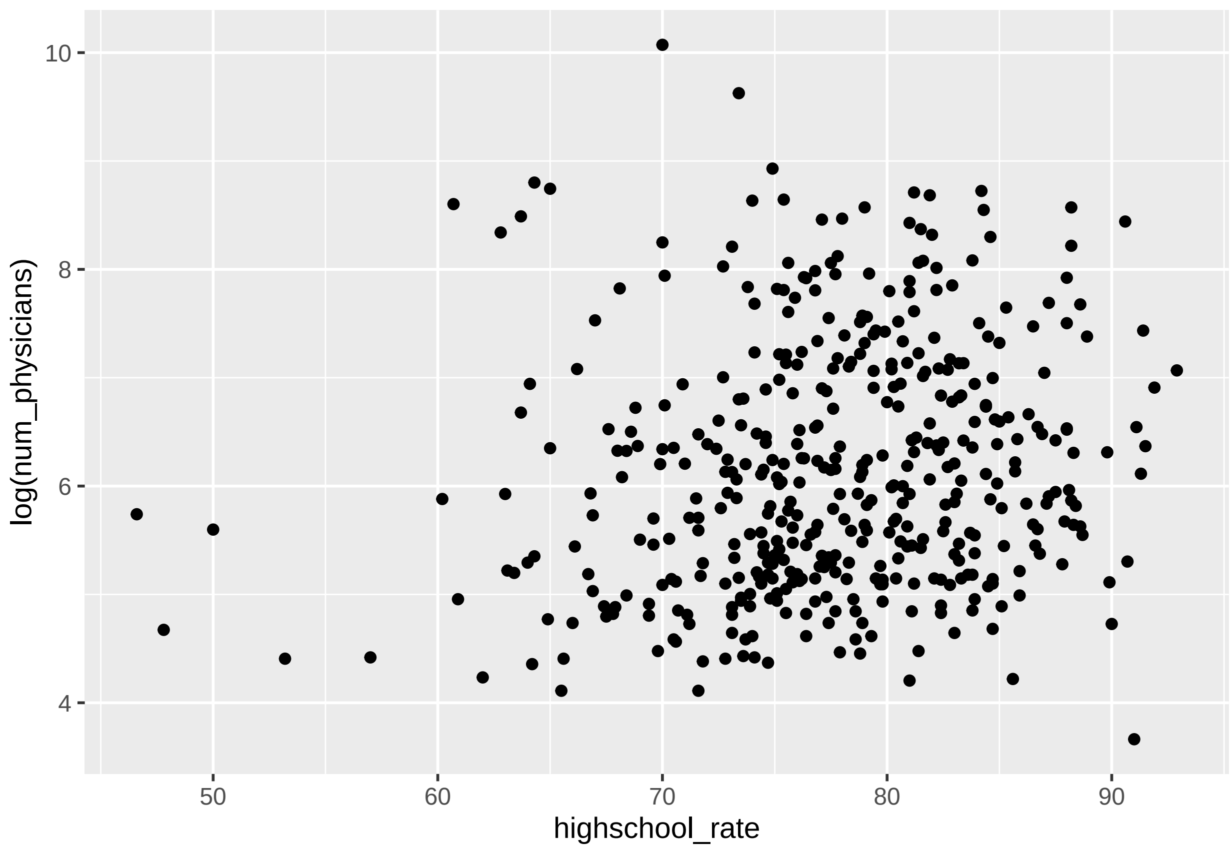

```r
# looks ok as is
model_data %>% ggplot(aes(x = pop_18to24, y = log(num_physicians))) +
  geom_point()
```



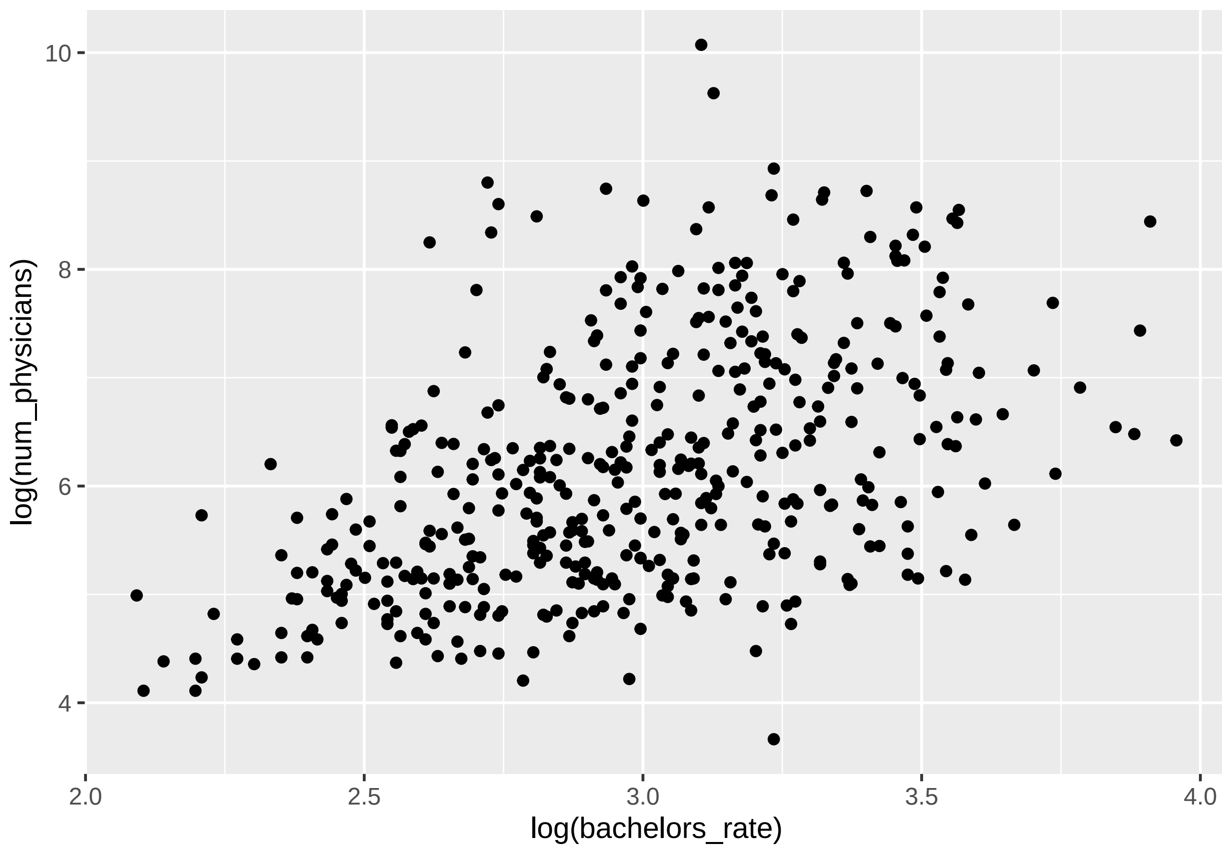

```
# looks ok as is
model_data %>% ggplot(aes(x = pop_over65, y = log(num_physicians))) +
  geom_point()
```



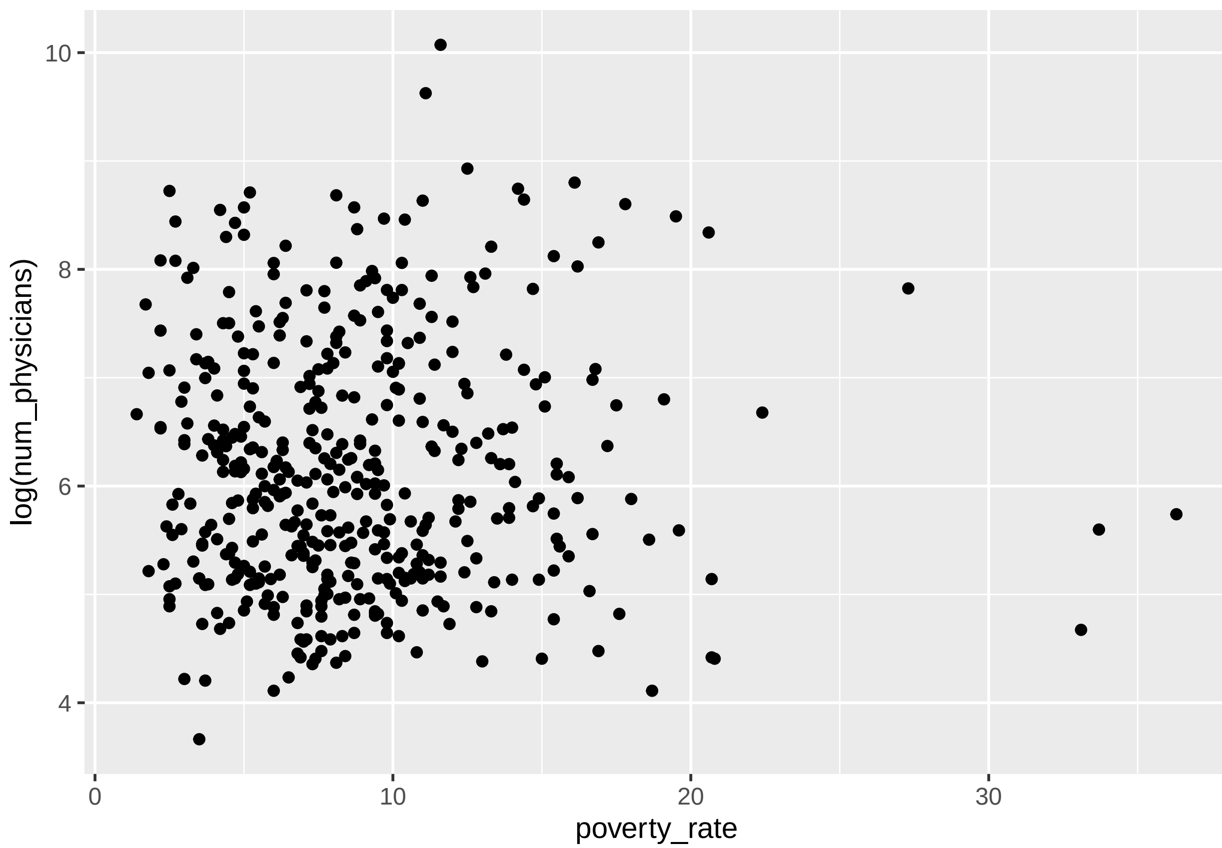

```
# looks ok as is - tried 1/x, x + x^2, sqrt and log but non look better
model_data %>% ggplot(aes(x = highschool_rate, y = log(num_physicians))) +
  geom_point()
```



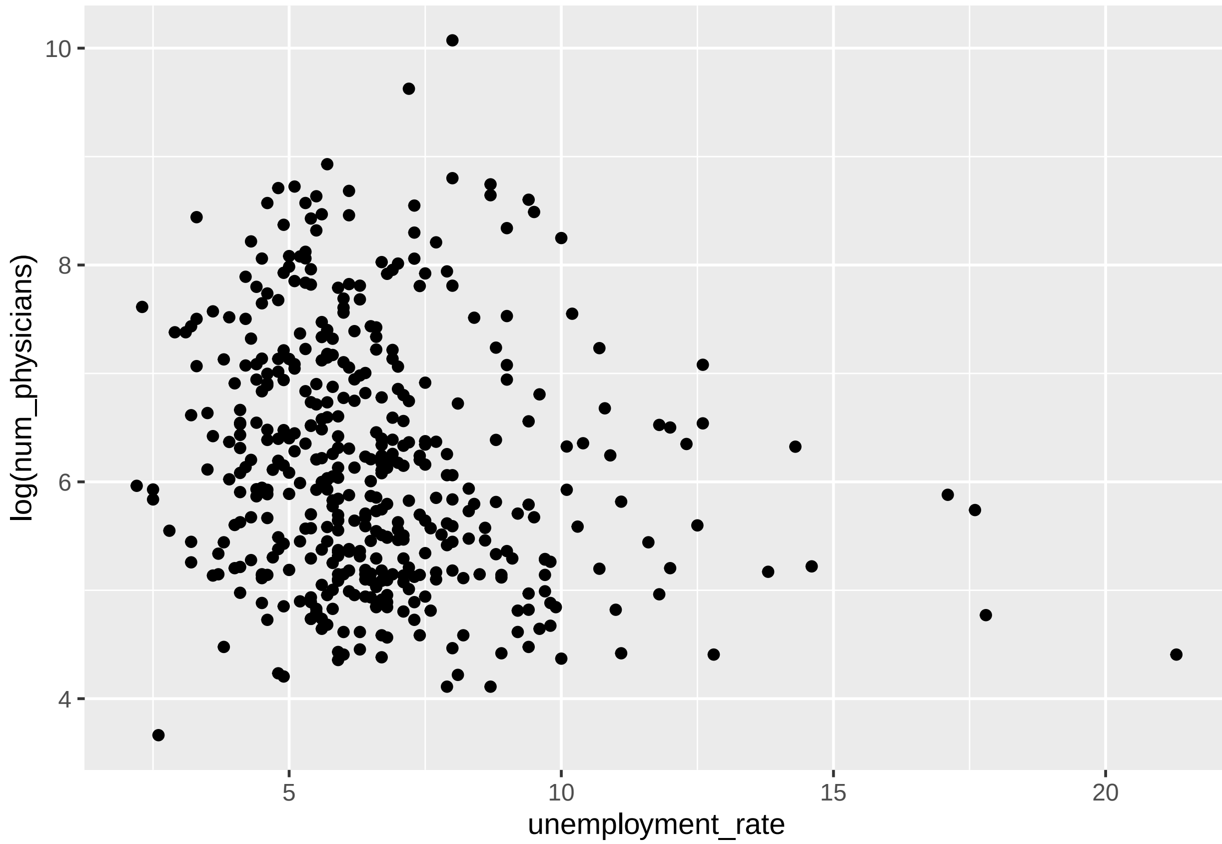

```
# looks ok as is but log looks better
model_data %>% ggplot(aes(x = log(bachelors_rate), y = log(num_physicians))) +
  geom_point()
```



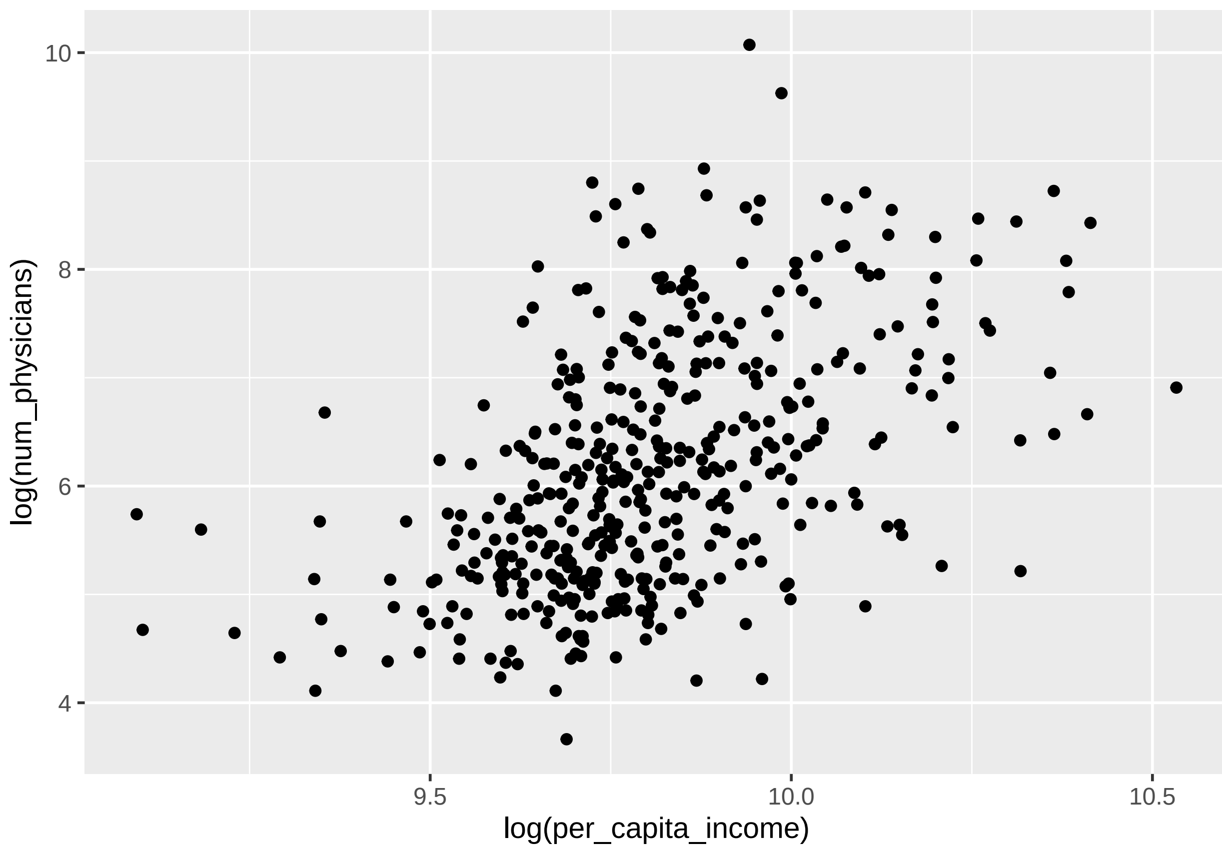

```
# looks ok as is
model_data %>% ggplot(aes(x = poverty_rate, y = log(num_physicians))) +
  geom_point()
```



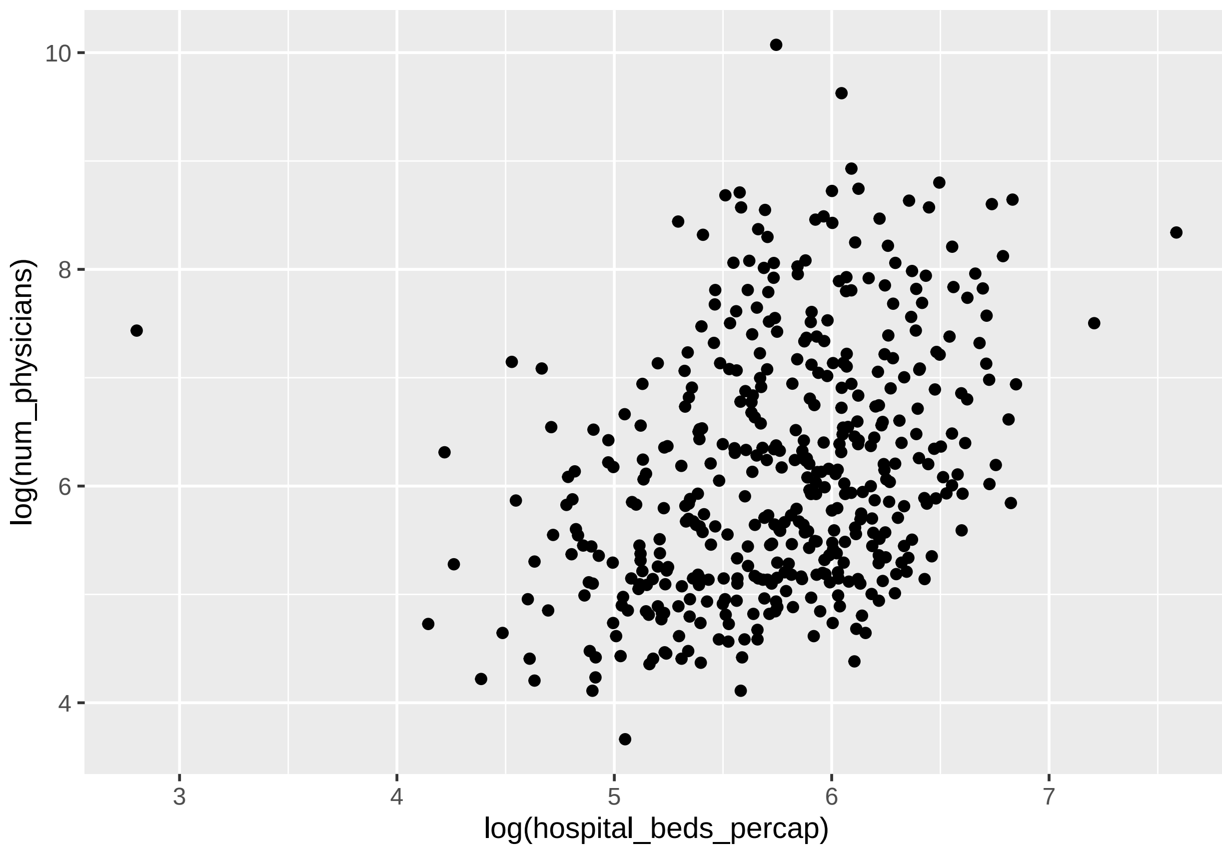

```
# looks ok as is
model_data %>% ggplot(aes(x = unemployment_rate, y = log(num_physicians))) +
  geom_point()
```



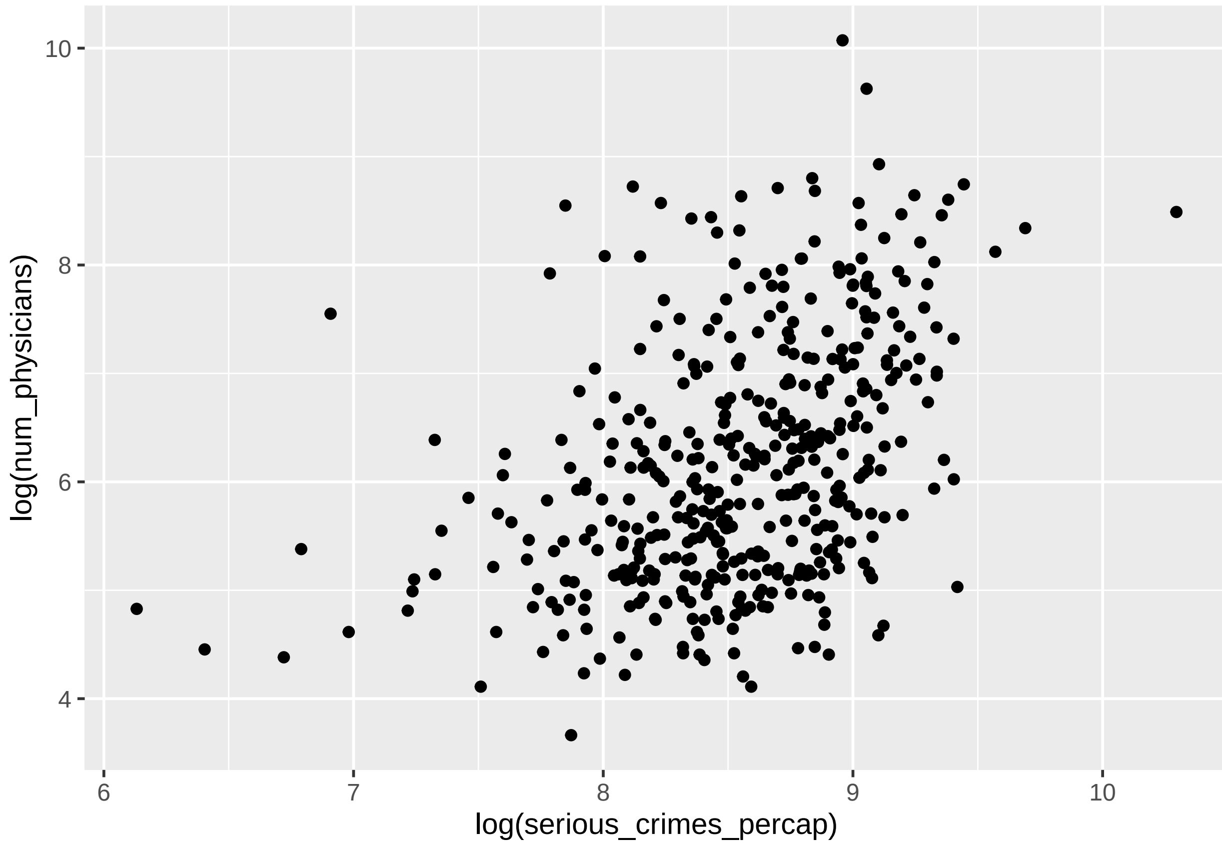

```
# looks ok as is but log is better
model_data %>% ggplot(aes(x = log(per_capita_income), y = log(num_physicians))) +
  geom_point()
```




```
# log helps!
model_data %>% ggplot(aes(x = log(hospital_beds_percap), y = log(num_physicians))) +
  geom_point()
```




```r
# looks ok as is but log is better
model_data %>% ggplot(aes(x = log(serious_crimes_percap), y = log(num_physicians))) +
  geom_point()
```




```r
# log transformations based on above EDA
transformed_data <- model_data %>%
  mutate(log_physicians = log(num_physicians),
         log_pop = log(total_pop),
         log_land = log(land_area),
         log_bachelors = log(bachelors_rate),
         log_percap_income = log(per_capita_income),
         log_hospital_beds = log(hospital_beds_percap),
         log_crimes = log(serious_crimes_percap)) %>%
  dplyr::select(-c(num_physicians, total_pop, land_area, bachelors_rate, per_capita_income,
          hospital_beds_percap, serious_crimes_percap))
```

```r
# Start by just looking at total_pop for fun
slr_model <- lm(log_physicians~log_pop, data = transformed_data)
summary(slr_model)
```

```
##
## Call:
## lm(formula = log_physicians ~ log_pop, data = transformed_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.48926 -0.33040 -0.04263  0.27043  2.52197
##
## Coefficients:
```

```
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -10.06656    0.38025  -26.47   <2e-16 ***
## log_pop       1.29996    0.03042   42.74   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5037 on 438 degrees of freedom
## Multiple R-squared:  0.8066, Adjusted R-squared:  0.8061
## F-statistic:  1826 on 1 and 438 DF,  p-value: < 2.2e-16
```

```r
# Full Model
mlr_full_model <- lm(log_physicians~., data = transformed_data)
summary(mlr_full_model)
```

```
##
## Call:
## lm(formula = log_physicians ~ ., data = transformed_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.94655 -0.16551 -0.01646  0.14326  1.37012
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)        -19.899033   1.433753 -13.879  < 2e-16 ***
## pop_18to24           0.013936   0.004977   2.800 0.005348 **
## pop_over65           0.015032   0.004772   3.150 0.001748 **
## highschool_rate     -0.002384   0.004195  -0.568 0.570094
## poverty_rate         0.023823   0.006340   3.757 0.000196 ***
## unemployment_rate   -0.021619   0.008181  -2.643 0.008527 **
## region2             -0.096126   0.042693  -2.252 0.024858 *
## region3             -0.051341   0.045770  -1.122 0.262620
## region4              0.161340   0.054295   2.972 0.003131 **
## log_pop              1.072263   0.021622  49.591  < 2e-16 ***
## log_land            -0.044456   0.019109  -2.326 0.020466 *
## log_bachelors        0.569723   0.097012   5.873 8.66e-09 ***
## log_percap_income    0.771188   0.152647   5.052 6.50e-07 ***
## log_hospital_beds    0.543203   0.033362  16.282  < 2e-16 ***
## log_crimes           0.014359   0.037690   0.381 0.703406
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2689 on 425 degrees of freedom
## Multiple R-squared:  0.9465, Adjusted R-squared:  0.9447
## F-statistic: 537.1 on 14 and 425 DF,  p-value: < 2.2e-16
```

```r
# Remove some fields based on EDA and full model summary
# The final group here was chosen based on EDA, p-values, and trial and error
sub1_data <- transformed_data %>%
  dplyr::select(-highschool_rate, -log_crimes, -log_land)
mlr_sub1_model <- lm(log_physicians~., data = sub1_data)
summary(mlr_sub1_model)
```

```
##
```

```
## Call:
## lm(formula = log_physicians ~ ., data = sub1_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.90134 -0.18275 -0.01244  0.14497  1.34825
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -21.634618   1.165895 -18.556  < 2e-16 ***
## pop_18to24          0.016813   0.004822   3.486  0.00054 ***
## pop_over65          0.015297   0.004784   3.198  0.00149 **
## poverty_rate        0.028214   0.005233   5.392 1.16e-07 ***
## unemployment_rate  -0.023414   0.008070  -2.901  0.00391 **
## region2            -0.098454   0.039829  -2.472  0.01383 *
## region3            -0.042164   0.040105  -1.051  0.29370
## region4             0.120411   0.045368   2.654  0.00825 **
## log_pop             1.066383   0.019930  53.506  < 2e-16 ***
## log_bachelors       0.507090   0.075970   6.675 7.69e-11 ***
## log_percap_income   0.925893   0.137381   6.740 5.15e-11 ***
## log_hospital_beds   0.546572   0.032444  16.846  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2699 on 428 degrees of freedom
## Multiple R-squared:  0.9457, Adjusted R-squared:  0.9443
## F-statistic: 677.9 on 11 and 428 DF,  p-value: < 2.2e-16
```

```
anova(mlr_sub1_model, mlr_full_model)
```

```
## Analysis of Variance Table
##
## Model 1: log_physicians ~ pop_18to24 + pop_over65 + poverty_rate + unemployment_rate +
##     region + log_pop + log_bachelors + log_percap_income + log_hospital_beds
## Model 2: log_physicians ~ pop_18to24 + pop_over65 + highschool_rate +
##     poverty_rate + unemployment_rate + region + log_pop + log_land +
##     log_bachelors + log_percap_income + log_hospital_beds + log_crimes
##   Res.Df    RSS Df Sum of Sq     F Pr(>F)
## 1    428 31.189
## 2    425 30.740  3   0.44894 2.069 0.1037
```

```
# Adding a test for slr model vs full to show the the slr is rejected
anova(slr_model, mlr_full_model)
```

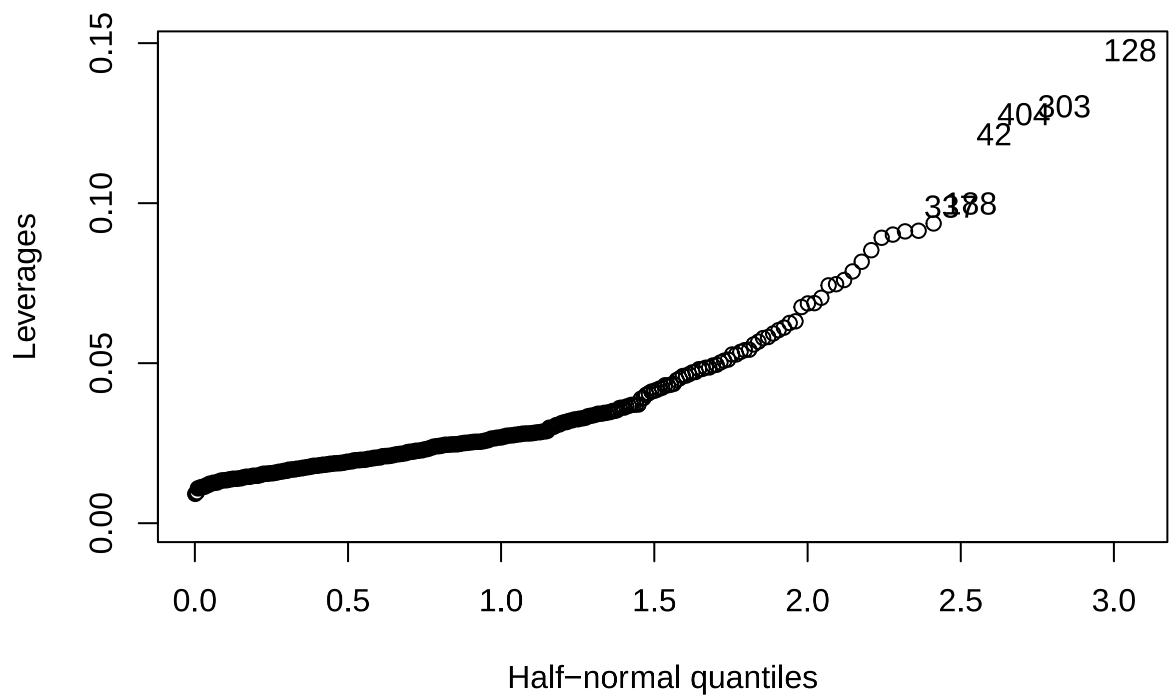
```
## Analysis of Variance Table
##
## Model 1: log_physicians ~ log_pop
## Model 2: log_physicians ~ pop_18to24 + pop_over65 + highschool_rate +
##     poverty_rate + unemployment_rate + region + log_pop + log_land +
##     log_bachelors + log_percap_income + log_hospital_beds + log_crimes
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1    438 111.14
## 2    425  30.74 13    80.397 85.504 < 2.2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## Checking high-leverage points
leverages=lm.influence(mlr_sub1_model)$hat
head(leverages)
```

```
##          1          2          3          4          5          6
## 0.06110336 0.04819432 0.03643543 0.03598304 0.03700481 0.05271986
```

```
## Plot to help identify high leverage observations
halfnorm(leverages, nlab=6, labs=as.character(1:length(leverages)), ylab="Leverages")
```




```
## Determining leverages that exceed a 2p/n threshold
n = dim(model_data)[1]
p = length(variable.names(mlr_sub1_model))
leverages.high = leverages[leverages>(2*p/n)]
leverages.high
```

```
##          1         42         67         95        128        155        171
## 0.06110336 0.12159176 0.05817187 0.07467461 0.14776376 0.06307226 0.08918100
##        173        184        187        188        206        246        249
## 0.08529432 0.05673314 0.06256259 0.09986278 0.07043394 0.05927733 0.06031901
##        272        301        303        334        337        357        363
```

```
## 0.05584776 0.06875972 0.13022693 0.05784573 0.09888911 0.06753811 0.09138324
##        396        398        404        405        412        415        433
## 0.07866738 0.09365475 0.12777068 0.07597522 0.08168668 0.09117746 0.06866181
##        436        437
## 0.07431743 0.09017867
```

```
## We currently have many high leverage points (30)
## Before continuing, let us look at what high leverage points are good and bad
## Calculate IQR for number of physicians
IQR_y = IQR(model_data$num_physicians)
## Define range with its lower limit being (Q1 - IQR) and upper limit being (Q3 + IQR)
QT1_y = quantile(model_data$num_physicians,0.25)
QT3_y = quantile(model_data$num_physicians,0.75)
lower_lim_y = QT1_y - IQR_y
upper_lim_y = QT3_y + IQR_y
vector_lim_y = c(lower_lim_y,upper_lim_y)
## Range for number of physicians
vector_lim_y
```

```
##     25%     75%
## -670.50 1889.25
```

```
## Extract observations with high leverage points from the original data frame
highlev = data[leverages>2*p/n,]
## Select only the observations with leverage points outside the range
highlev_lower = highlev[highlev$num_physicians < vector_lim_y[1], ]
highlev_upper = highlev[highlev$num_physicians > vector_lim_y[2], ]
highlev2 = rbind(highlev_lower,highlev_upper)

# Only 3 bad high leverage points
highlev2
```

```
## # A tibble: 3 x 17
##      id county     state land_~1 total~2 pop_1~3 pop_o~4 num_p~5 num_h~6 serio~7
##   <dbl> <chr>      <chr>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1     1 Los_Angel~ CA       4060 8863164    32.1     9.7   23677   27700  688936
## 2    67 Suffolk    MA         59  663906    39.2    12.1    5674    6154   68808
## 3    95 Orleans    LA        181  496938    28.3    13      2500    4018   54238
## # ... with 7 more variables: highschool_rate <dbl>, bachelors_rate <dbl>,
## #   poverty_rate <dbl>, unemployment_rate <dbl>, per_capita_income <dbl>,
## #   total_personal_income <dbl>, region <fct>, and abbreviated variable names
## #   1: land_area, 2: total_pop, 3: pop_18to24, 4: pop_over65,
## #   5: num_physicians, 6: num_hospital_beds, 7: serious_crimes
```

```
## Computing Studentized Residuals
mlr_sub1_model.resid = rstudent(mlr_sub1_model);

## Critical value with Bonferroni correction
## Note: Compare to t-value later at the alpha we choose
bonferroni_cv = qt(.05/(2*n), n-p-1)
bonferroni_cv
```

```
## [1] -3.895681
```

```r
## Sorting residuals to find outliers
mlr_sub1_model.resid.sorted = sort(abs(mlr_sub1_model.resid), decreasing=TRUE)[1:10]
print(mlr_sub1_model.resid.sorted)
```

```
##      418       42      431      291      258       50      248      282
## 5.285196 4.004943 3.439459 3.369727 3.266127 2.922149 2.750065 2.687347
##      346      241
## 2.608512 2.548376
```

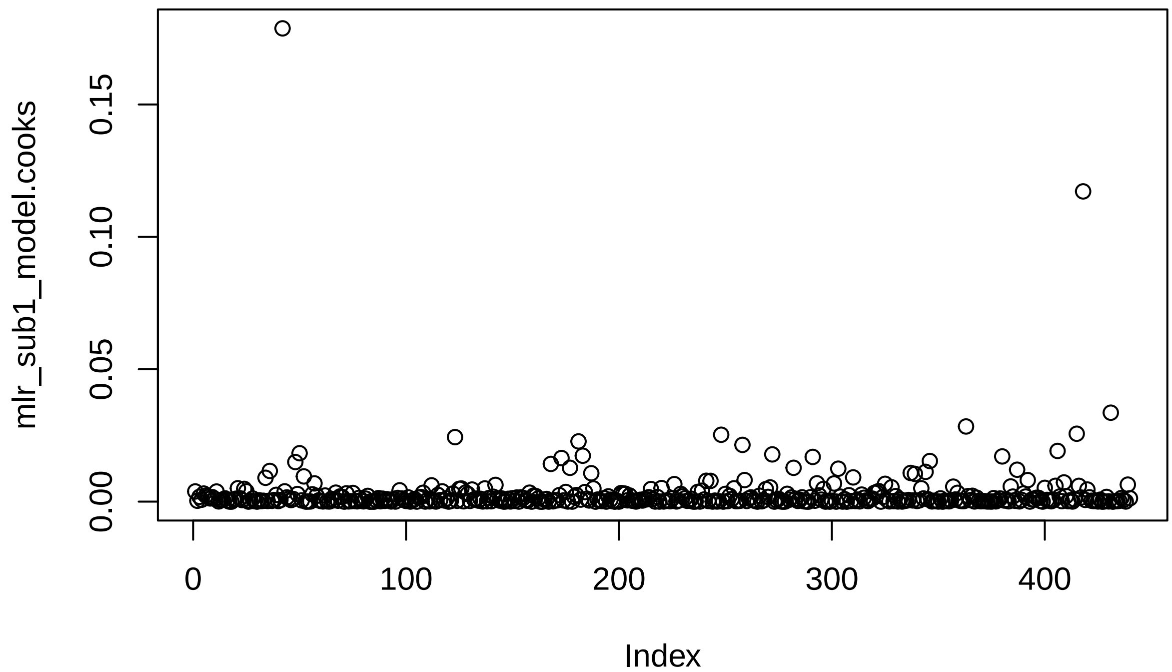
```r
## 2 points are outliers (418, 42)
mlr_sub1_model.outliers = mlr_sub1_model.resid.sorted[abs(mlr_sub1_model.resid.sorted) > abs(bonferroni_
print(mlr_sub1_model.outliers)
```

```
##      418       42
## 5.285196 4.004943
```

```r
## Finding high cook's distance observations
mlr_sub1_model.cooks = cooks.distance(mlr_sub1_model)
sort(mlr_sub1_model.cooks, decreasing = TRUE)[1:10]
```

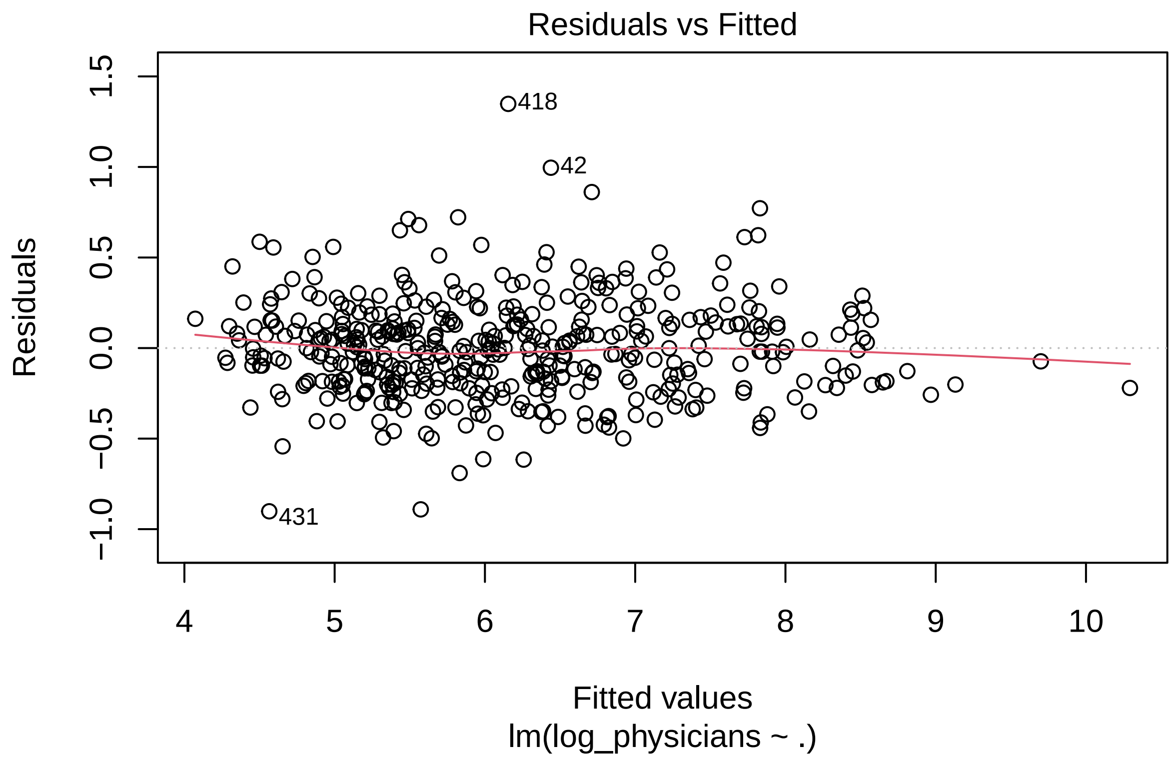
```
##         42        418        431        363        415        248        123
## 0.17873948 0.11716642 0.03358325 0.02838177 0.02564037 0.02525072 0.02435829
##        181        258        406
## 0.02274234 0.02142160 0.01913981
```

```r
## Plotting cook's distance
plot(mlr_sub1_model.cooks)
```




```
## Some observations have high cook's distance relative to other observations, but none have cook's d >
```

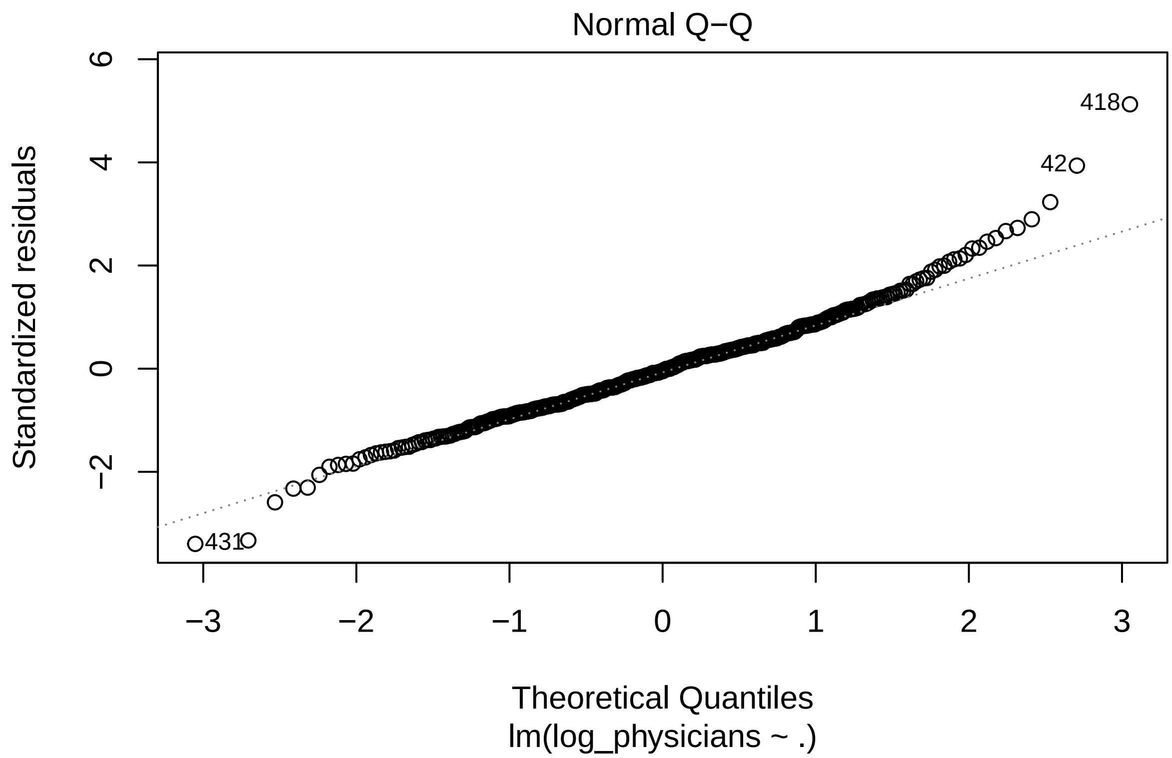
```
## Checking Constant Variance
plot(mlr_sub1_model, which=1)
```



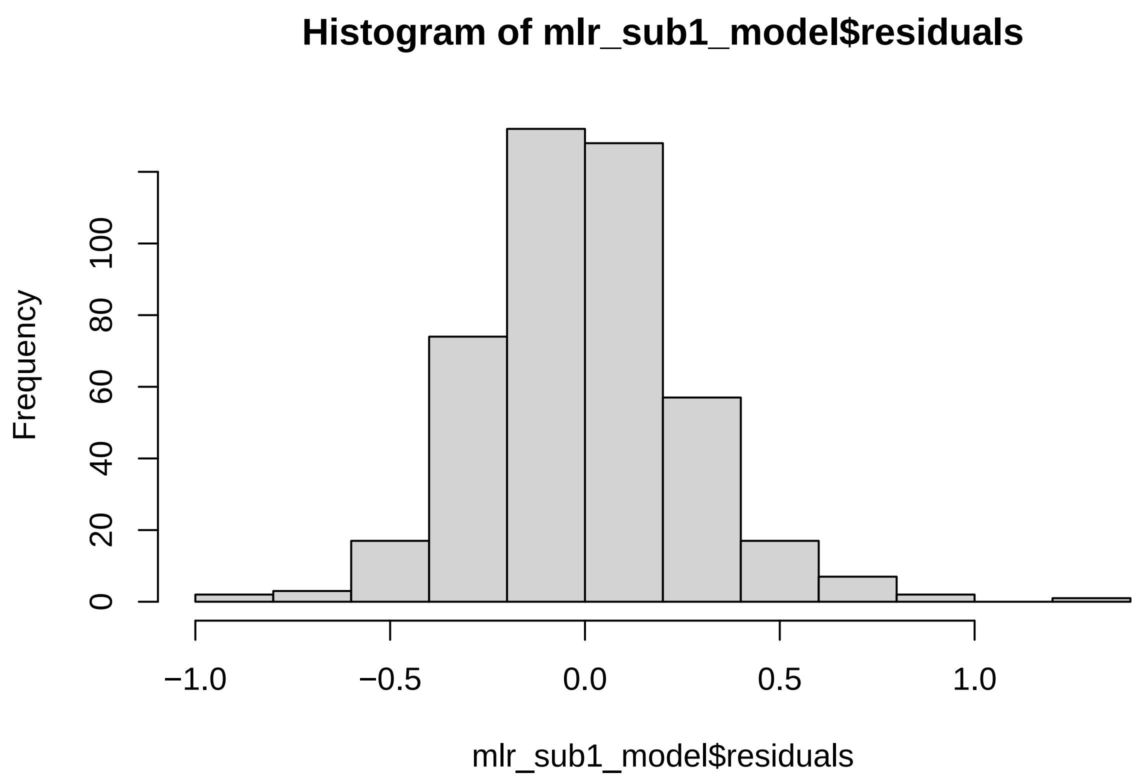

```
bptest(mlr_sub1_model)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  mlr_sub1_model
## BP = 37.381, df = 11, p-value = 9.945e-05
```

```
## Constant Variance seems to be violated based on p-value, the plot looks OK though
## Checking Normality
plot(mlr_sub1_model, which=2)
```




```
hist(mlr_sub1_model$residuals)
```



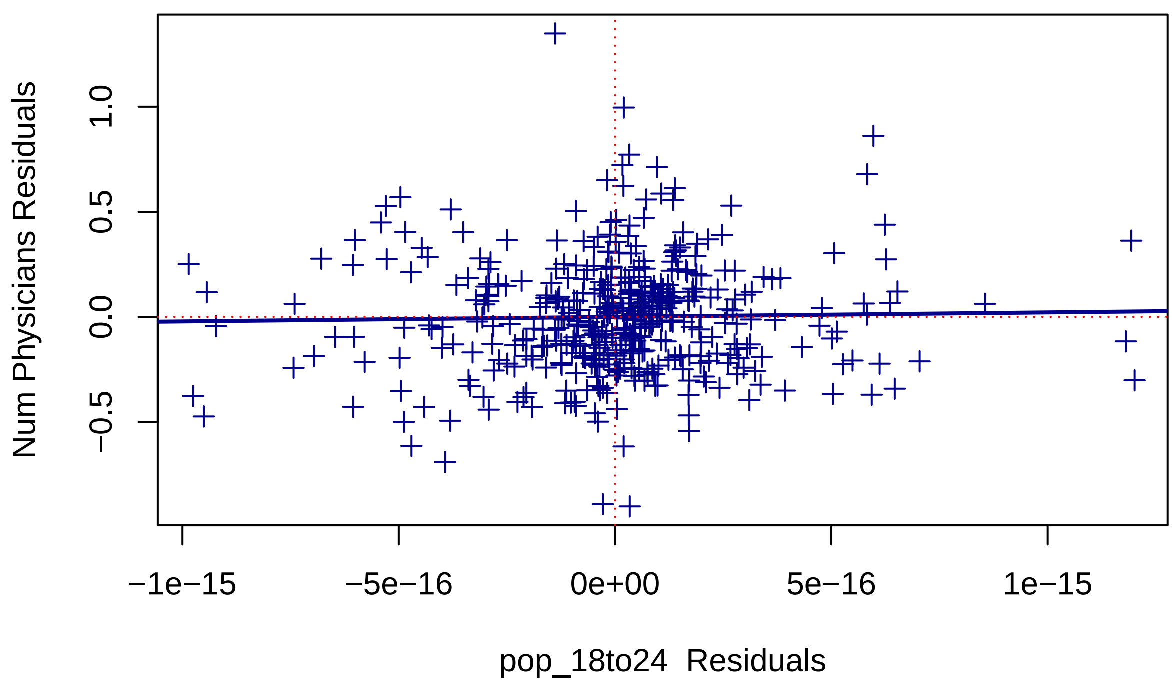

```
### We can use the KS test to assess normality because n>50.
ks.test(mlr_sub1_model$residuals, 'pnorm')  ## We may want to check that this is the right syntax for t
```

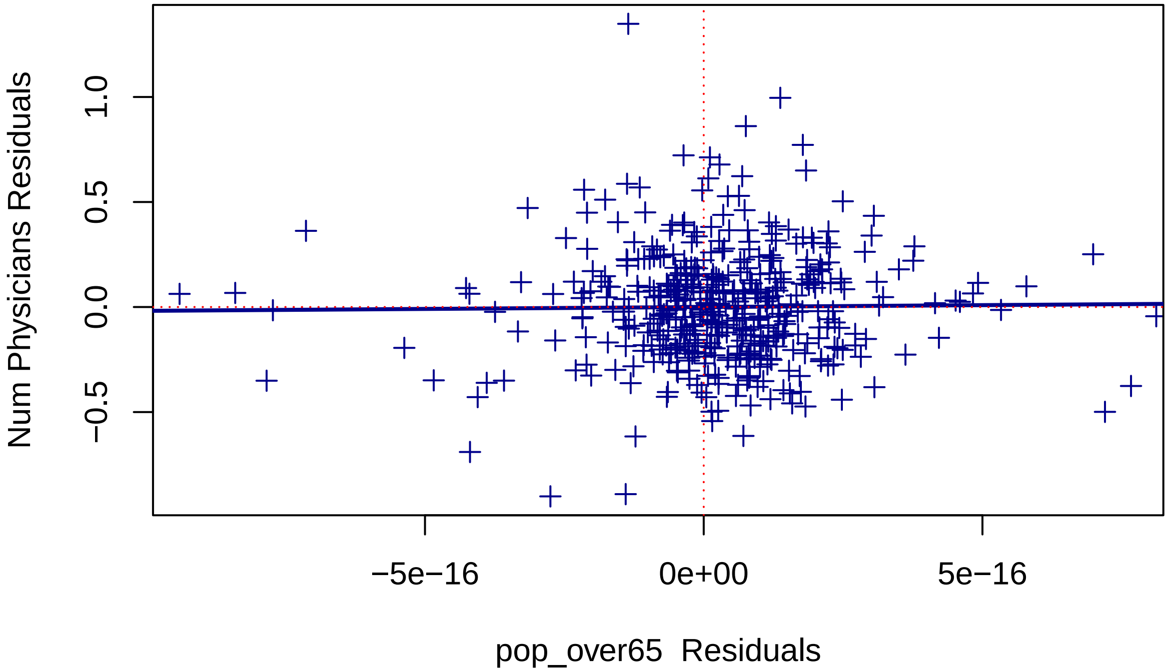
```
##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  mlr_sub1_model$residuals
## D = 0.30228, p-value < 2.2e-16
## alternative hypothesis: two-sided
```

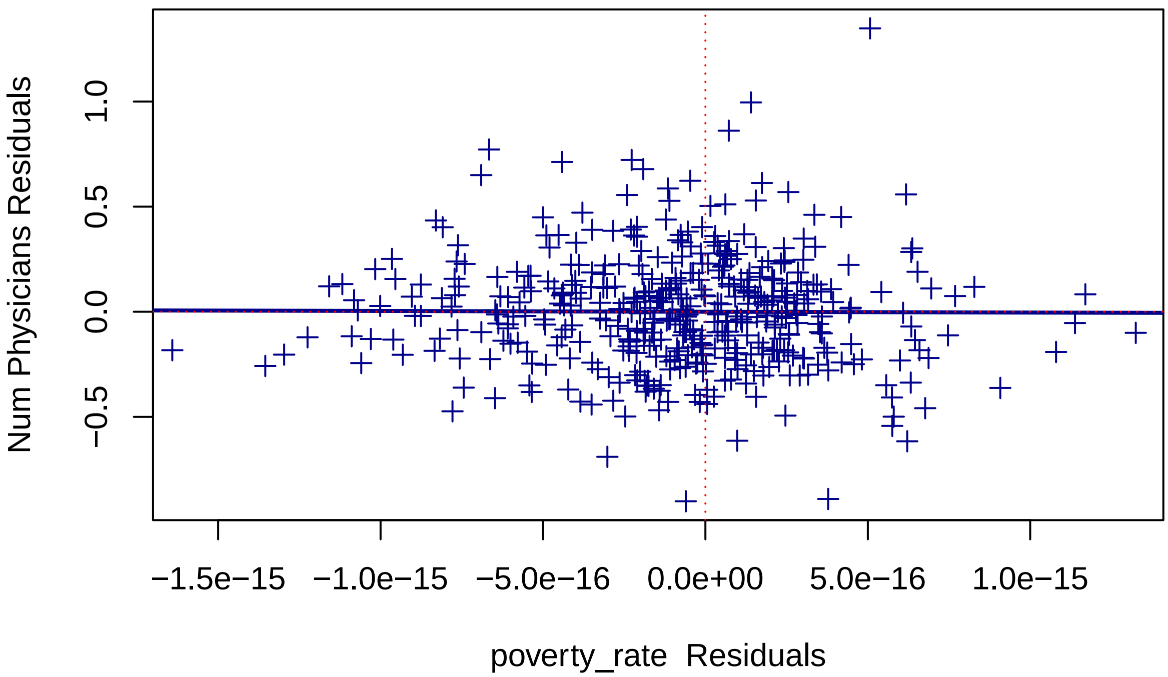
```
## Next step is to check linearity of each variable
```

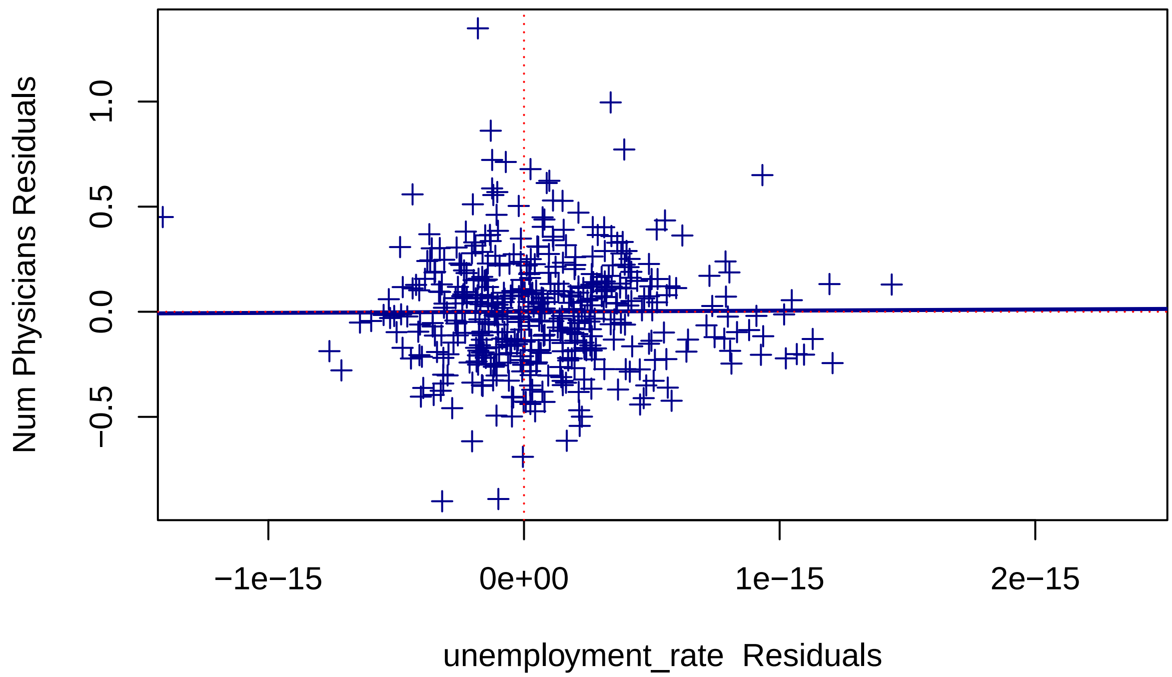
```
sub1_data <- data.frame(sub1_data)

checkLinearity <- function(var) {
  var_idx = which( colnames(sub1_data)==var )
  y.var = update(mlr_sub1_model, .~. -c(var_idx))$res
  #remove response + the variable itself
  x.var = lm(sub1_data[,var_idx] ~ . ,sub1_data[,-c(6)])$res

  plot(x.var, y.var, xlab=paste(var," Residuals"), ylab="Num Physicians Residuals",   col='Darkblue', p
  abline(lm(y.var ~ x.var), col='Darkblue', lwd=2)
  abline(v = 0, col="red", lty=3)
  abline(h = 0, col="red", lty=3)
}
```

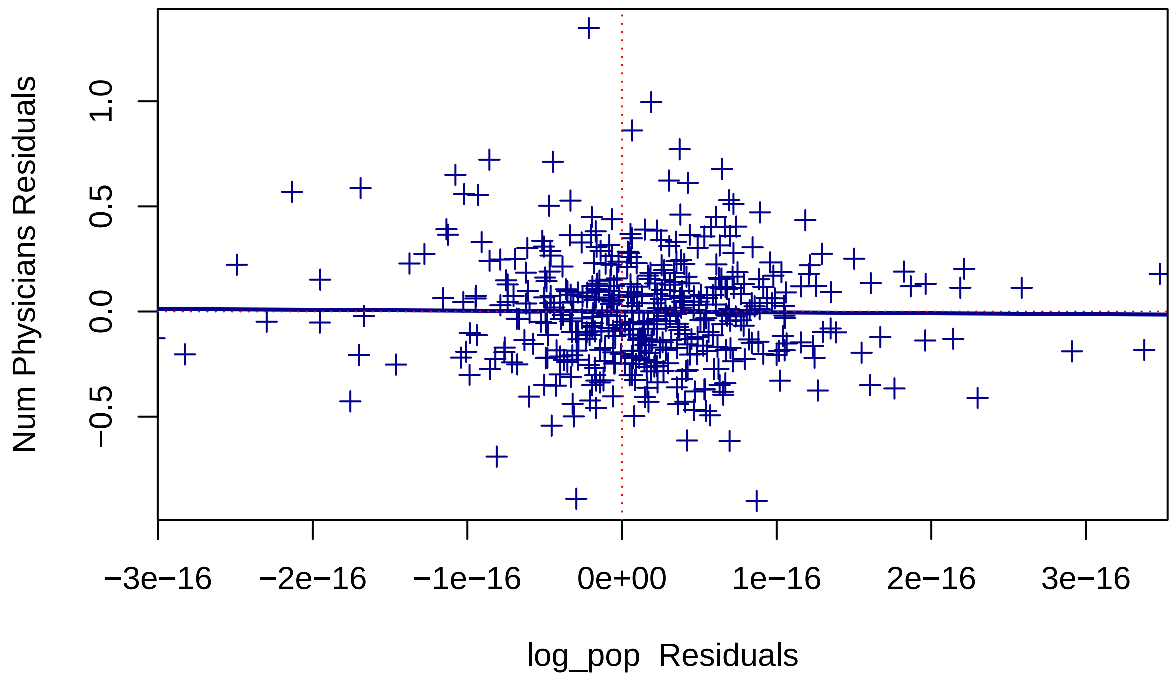
```r
predictors = names(sub1_data)
#remove the response variable (and region since it's a factor (?))
predictors = predictors[!(predictors %in% c("log_physicians","region"))]
#check linearity for each predictor
for (var in predictors) {
  checkLinearity(var)
}
```



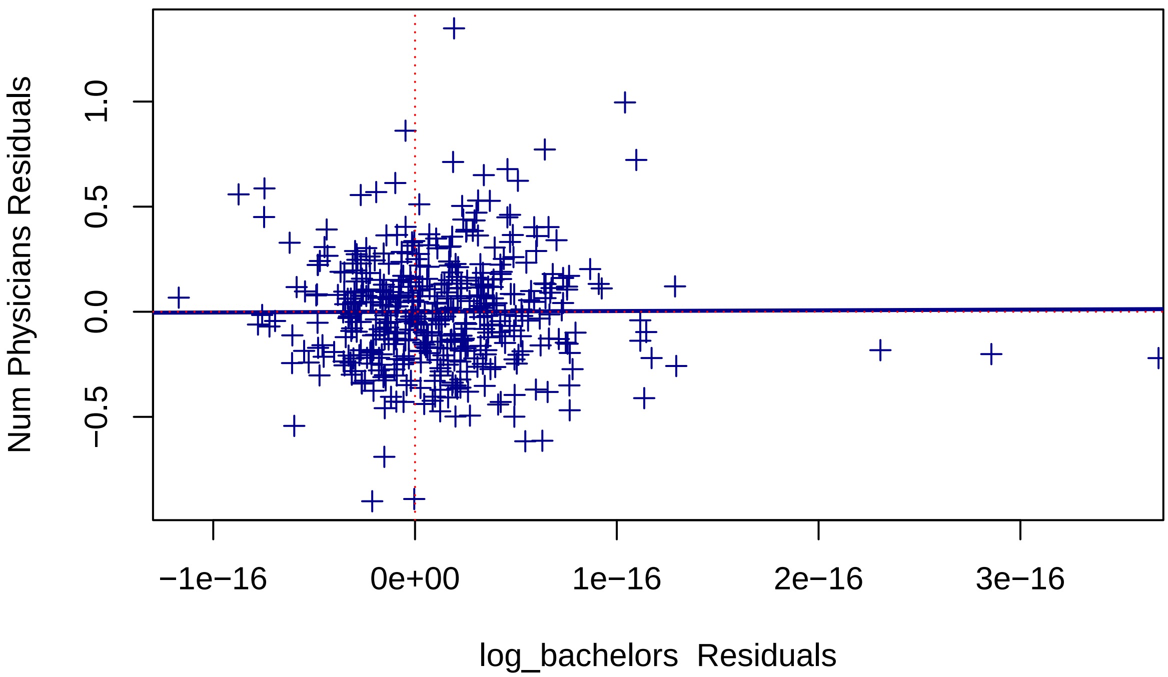

Num Physicians Residuals
1.0
0.5
0.0
−0.5
−5e−16
0e+00
5e−16
pop_over65 Residuals

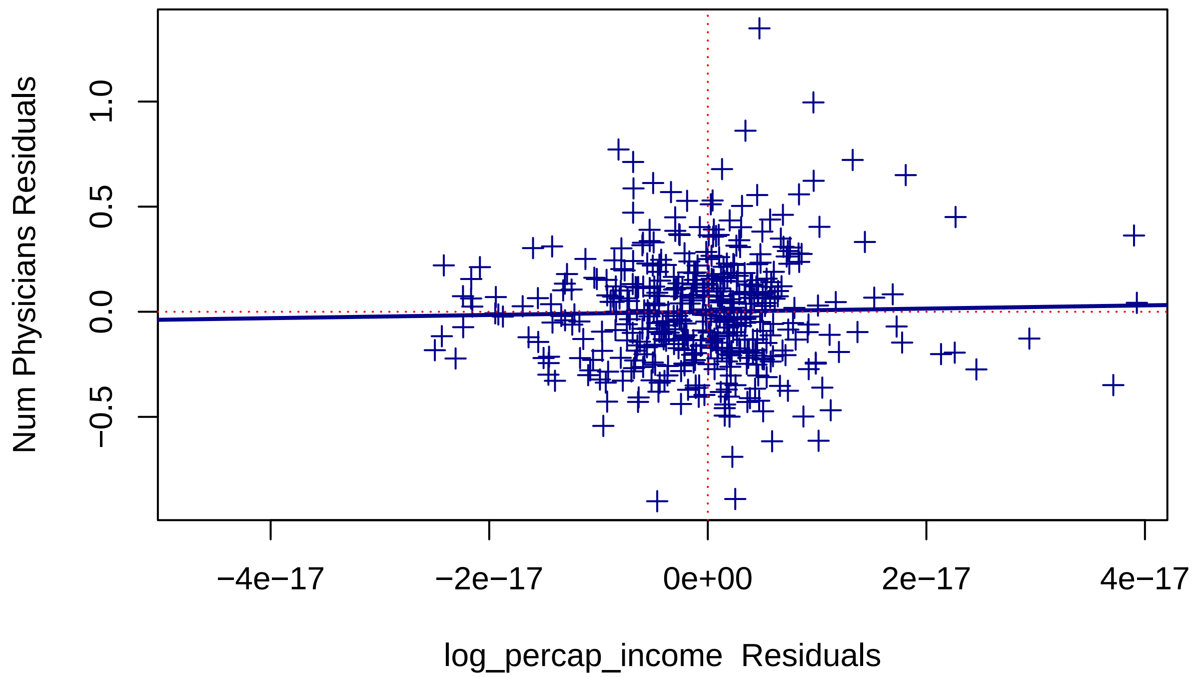

Num Physicians Residuals
−0.5
0.0
0.5
1.0
−1.5e−15
−1.0e−15
−5.0e−16
0.0e+00
5.0e−16
1.0e−15
poverty_rate Residuals

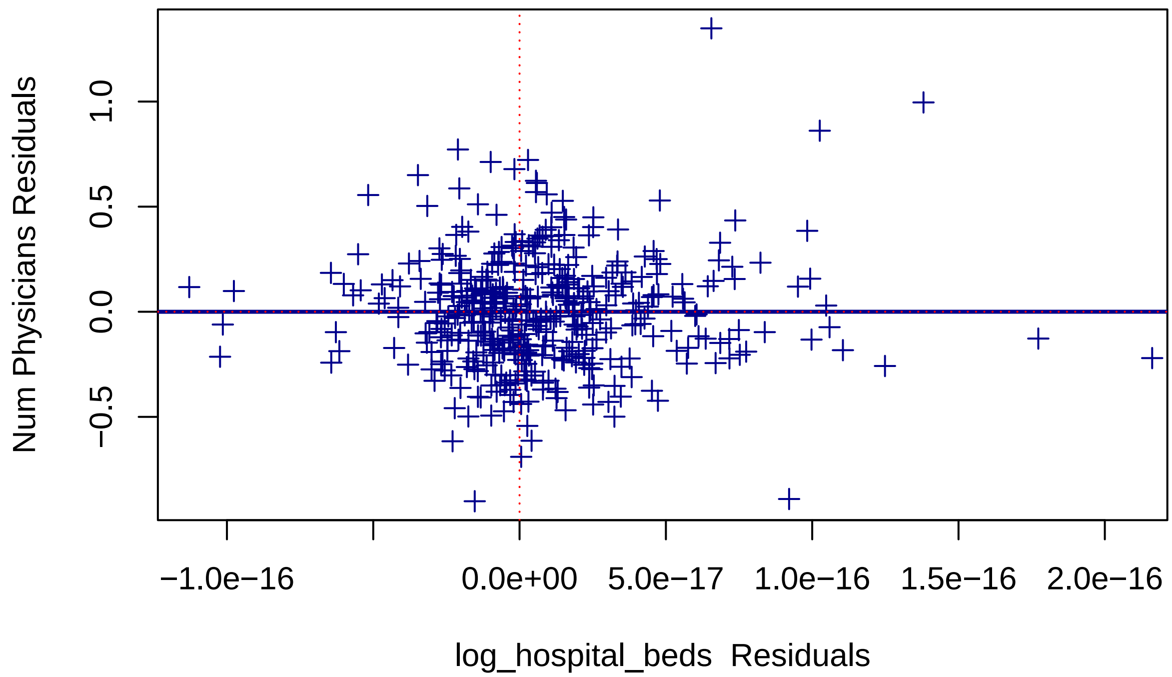

Num Physicians Residuals
unemployment_rate Residuals
−0.5
0.0
0.5
1.0
−1e−15
0e+00
1e−15
2e−15



Num Physicians Residuals
−0.5
0.0
0.5
1.0
−3e−16
−2e−16
−1e−16
0e+00
1e−16
2e−16
3e−16
log_pop Residuals


Num Physicians Residuals
−0.5
0.0
0.5
1.0
−1e−16
0e+00
1e−16
2e−16
3e−16
log_bachelors Residuals


Num Physicians Residuals
1.0
0.5
0.0
−0.5
−4e−17
−2e−17
0e+00
2e−17
4e−17
log_percap_income Residuals




```r
## Checking VIF
round(vif(mlr_sub1_model),3) #log_bachelors and log_percap_income are kind of concerning to me, but are
```

```
##        pop_18to24        pop_over65      poverty_rate unemployment_rate
##             2.461             2.198             3.577             2.144
##           region2           region3           region4           log_pop
##             1.774             2.196             1.794             1.495
##     log_bachelors log_percap_income log_hospital_beds
##             4.368             4.859             1.924
```

```r
## Grabbing design matrix
x = model.matrix(mlr_sub1_model)[,-1]

## Standardize the matrix
x = x - matrix(apply(x,2, mean), 440,11, byrow=TRUE)
x = x / matrix(apply(x, 2, sd), 440,11, byrow=TRUE)


## Compute the eigenvalues of the matrix
eigenvalues.x = eigen(t(x) %*% x)
eigenvalues.x$val
```

```
##  [1] 1275.27414  803.29800  668.63050  624.17316  530.54827  321.79620
##  [7]  219.22558  147.41423  111.62601   84.01384   43.00007
```

```r
## Compute Condition Number
sqrt(eigenvalues.x$val[1]/eigenvalues.x$val[8]) ## Is less than 30, looks good.
```

```
## [1] 2.941251
```