p8130_hw5_brm2150

Brooklynn McNeil

2024-12-03

A.

R data set state.x77 from library(faraway) contains information on 50 states from 1970s collected by US Census Bureau. The goal is to predict 'life expectancy' using a combination of remaining variables.The data has the outcome variable `life_exp` and the predictor variables `population`, `income`, `illiteracy`, `murder`, `hs_grad`, `frost`, and `area`. All of the predcitors we have are continuous variables.

Load Data

```
data = datasets::state.x77 |>
  as.tibble() |>
  janitor::clean_names()
```

```
Warning: 'as.tibble()' was deprecated in tibble 2.0.0.
i Please use 'as_tibble()' instead.
i The signature and semantics have changed, see '?as_tibble'.
This warning is displayed once every 8 hours.
Call 'lifecycle::last_lifecycle_warnings()' to see where this warning was
generated.
```

Let's take a look at the summary of our predictors of interest.

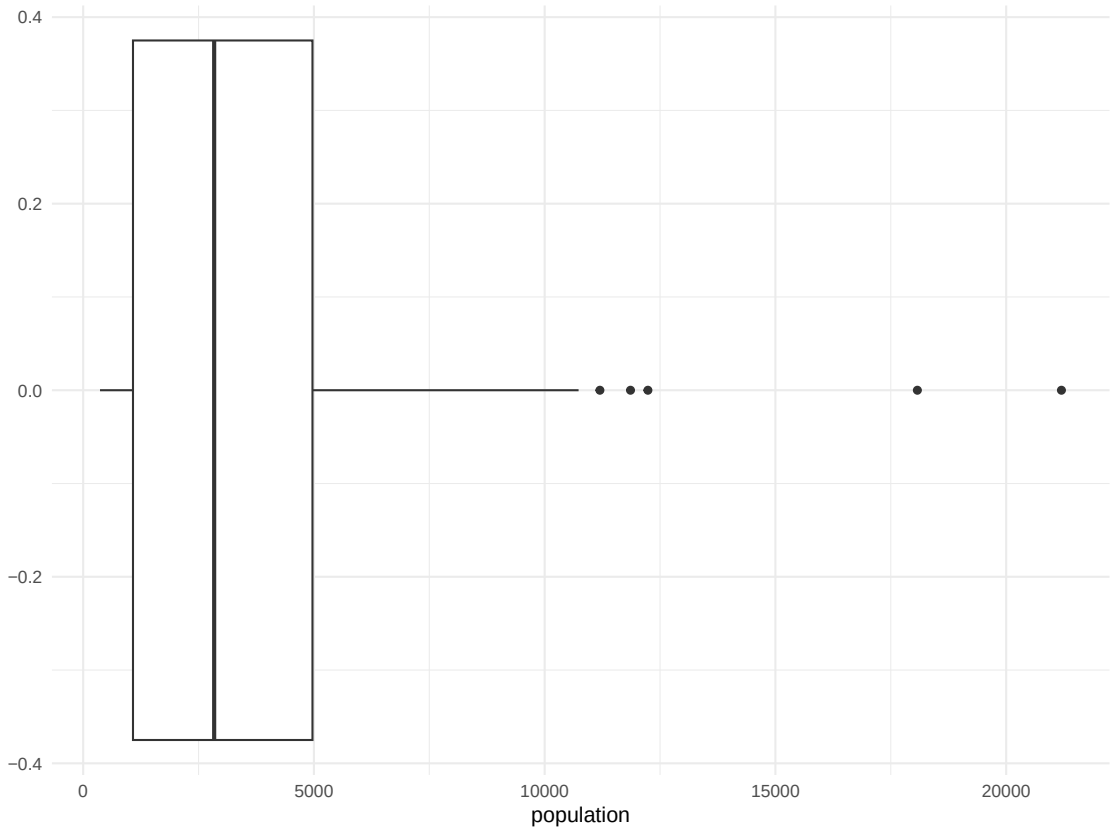
```
data |>
  summary() |>
  knitr::kable()
```

population	income	illiteracy	life_exp	murder	hs_grad	frost	area
Min. : 365	Min. :3098	Min. :0.500	Min. :67.96	Min. : 1.400	Min. :37.80	Min. : 0.00	Min. : 1049
1st Qu.: 1080	1st Qu.:3993	1st Qu.:0.625	1st Qu.:70.12	1st Qu.: 4.350	1st Qu.:48.05	1st Qu.: 66.25	1st Qu.: 36985
Median : 2838	Median :4519	Median :0.950	Median :70.67	Median : 6.850	Median :53.25	Median :114.50	Median : 54277
Mean : 4246	Mean :4436	Mean :1.170	Mean :70.88	Mean : 7.378	Mean :53.11	Mean :104.46	Mean : 70736
3rd Qu.: 4968	3rd Qu.:4814	3rd Qu.:1.575	3rd Qu.:71.89	3rd Qu.:10.675	3rd Qu.:59.15	3rd Qu.:139.75	3rd Qu.: 81162

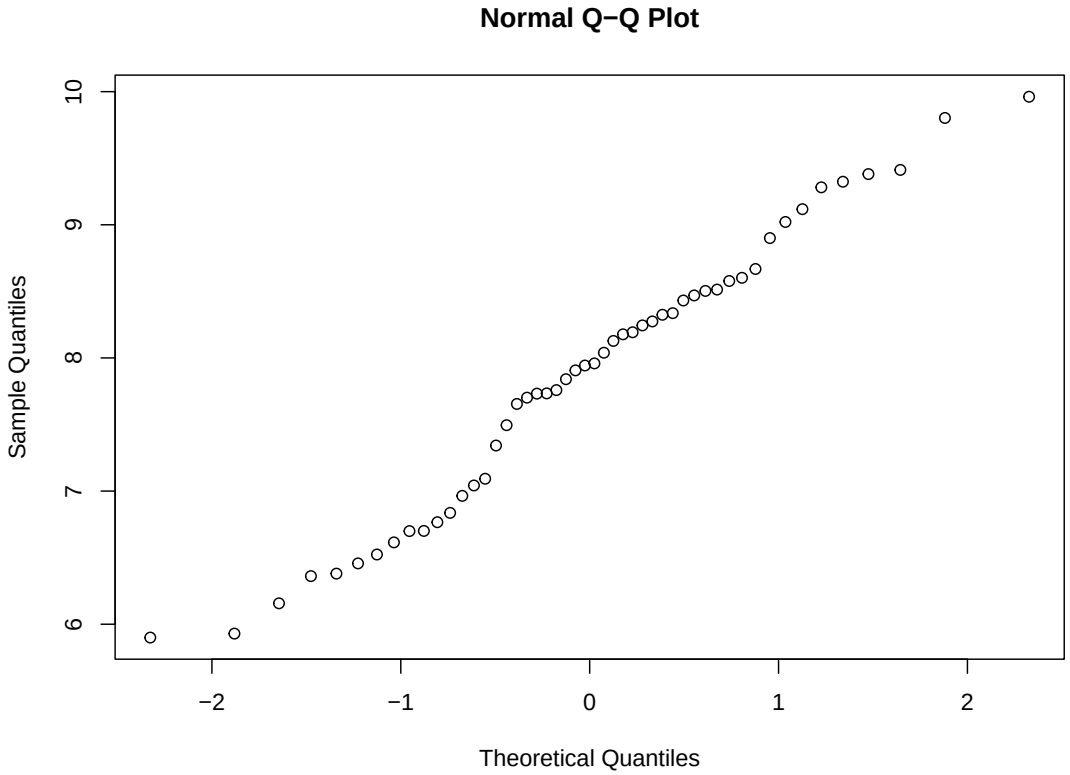
	population	income	illiteracy	life_exp	murder	hs_grad	frost	area
	Max. :21198	Max. :6315	Max. :2.800	Max. :73.60	Max. :15.100	Max. :67.30	Max. :188.00	Max. :566432

B. Exploratory analysis

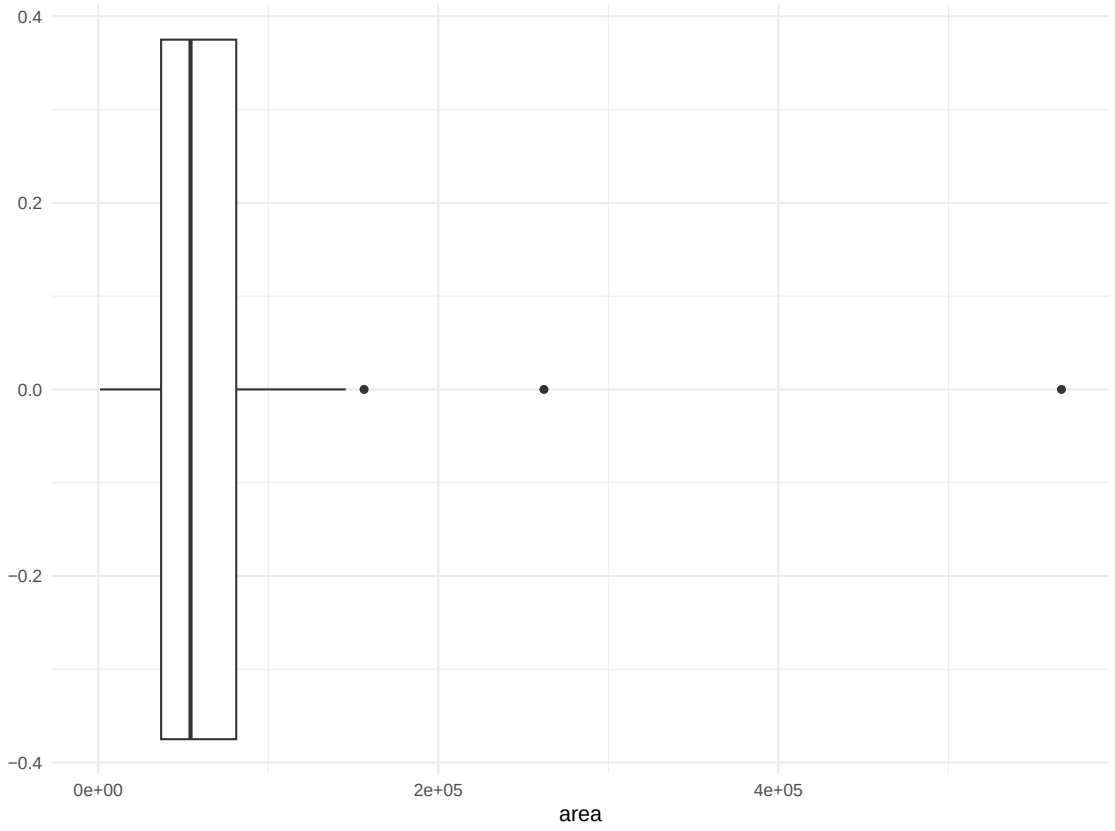
```
# basic histograms for potentially not normal continuous variables

data |>
  ggplot(aes(x = population)) +
  geom_boxplot()
```



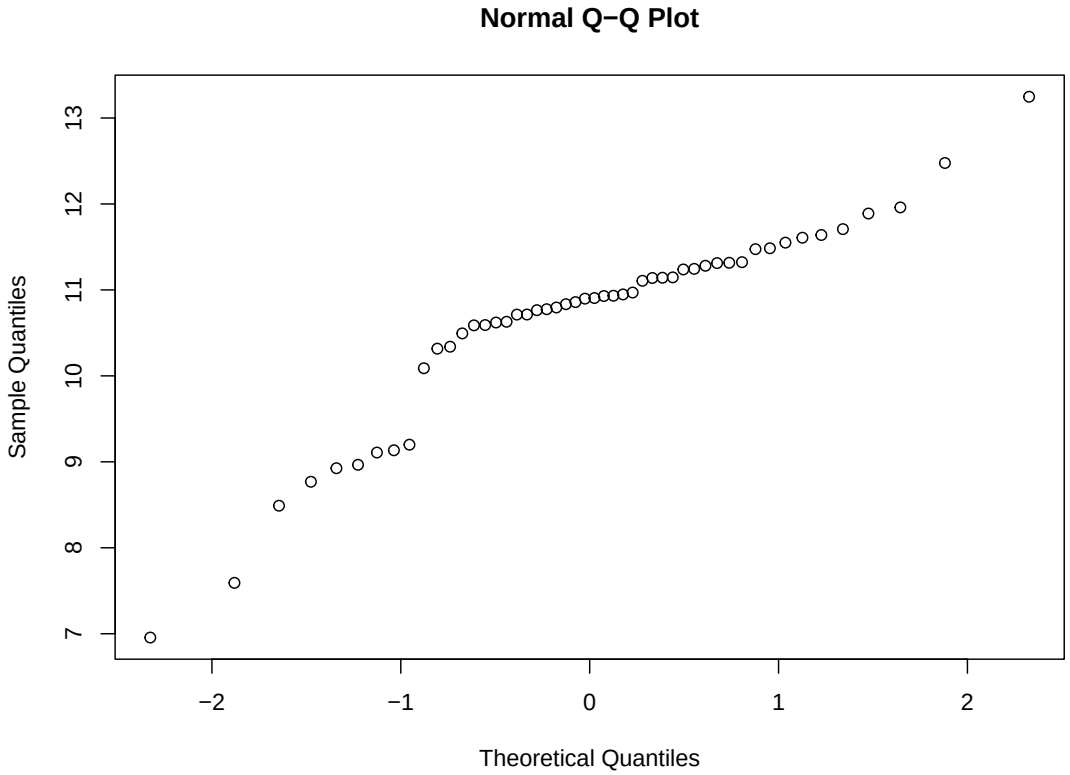

```
qqnorm(log(data$population))
```



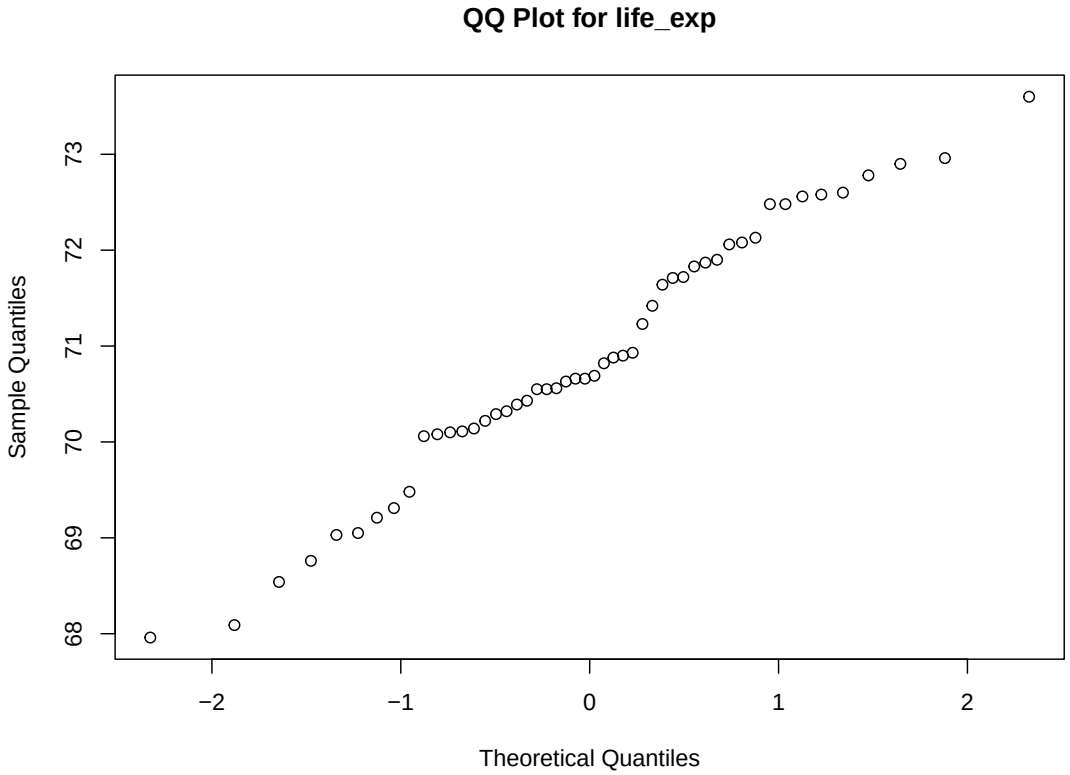

```
data |>
  ggplot(aes(x = area)) +
  geom_boxplot()
```



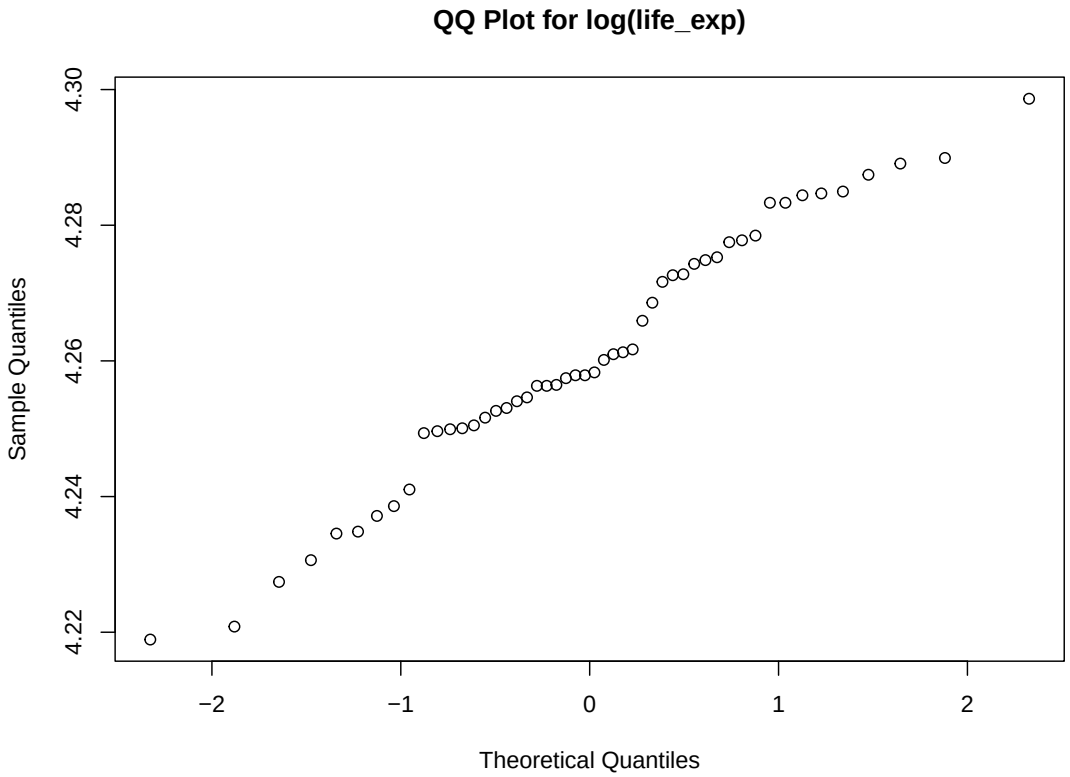

```
qqnorm(log(data$area))
```



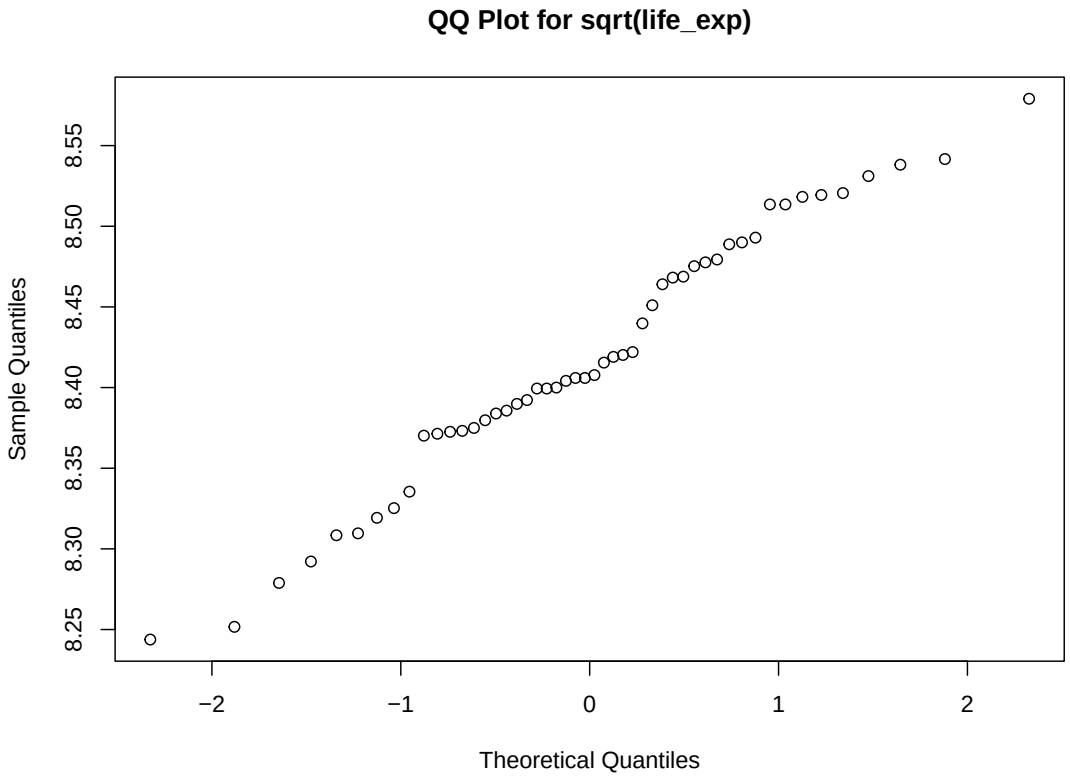

```
# check for transformations of the outcome
qqnorm(data$life_exp,main = paste("QQ Plot for", "life_exp")) # identitiy
```



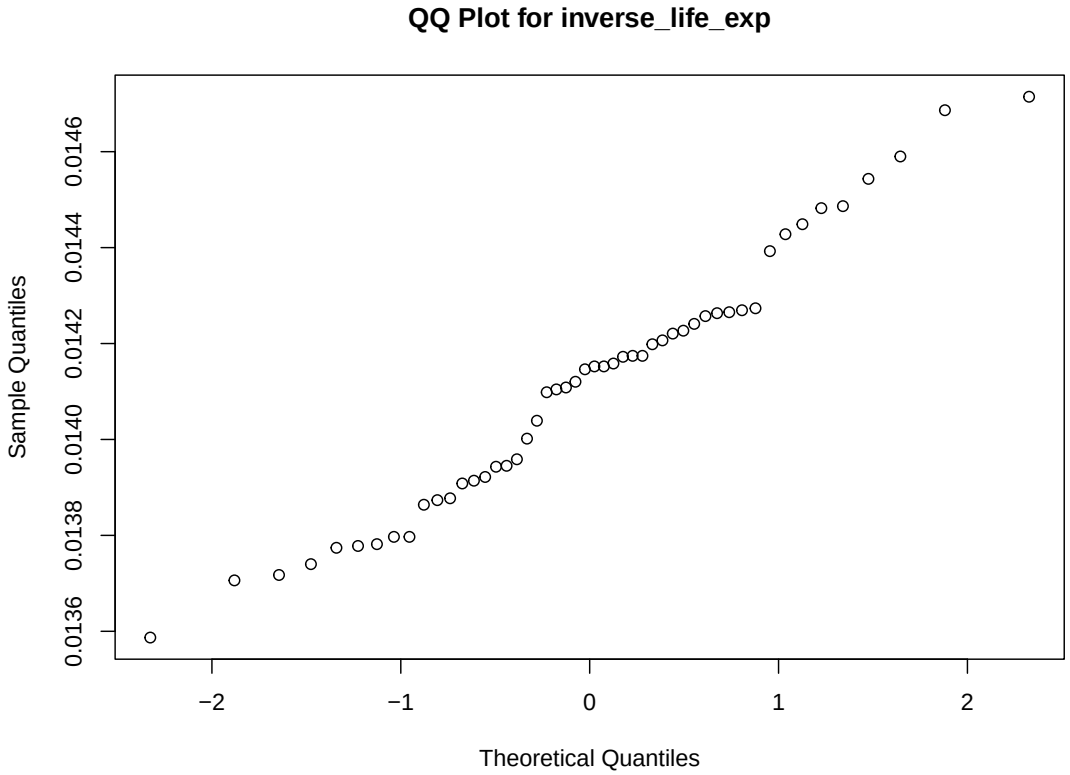

```r
qqnorm(log(data$life_exp), main = paste("QQ Plot for", "log(life_exp)")) # log
```




```
qqnorm(sqrt(data$life_exp), main = paste("QQ Plot for", "sqrt(life_exp)")) # square root
```

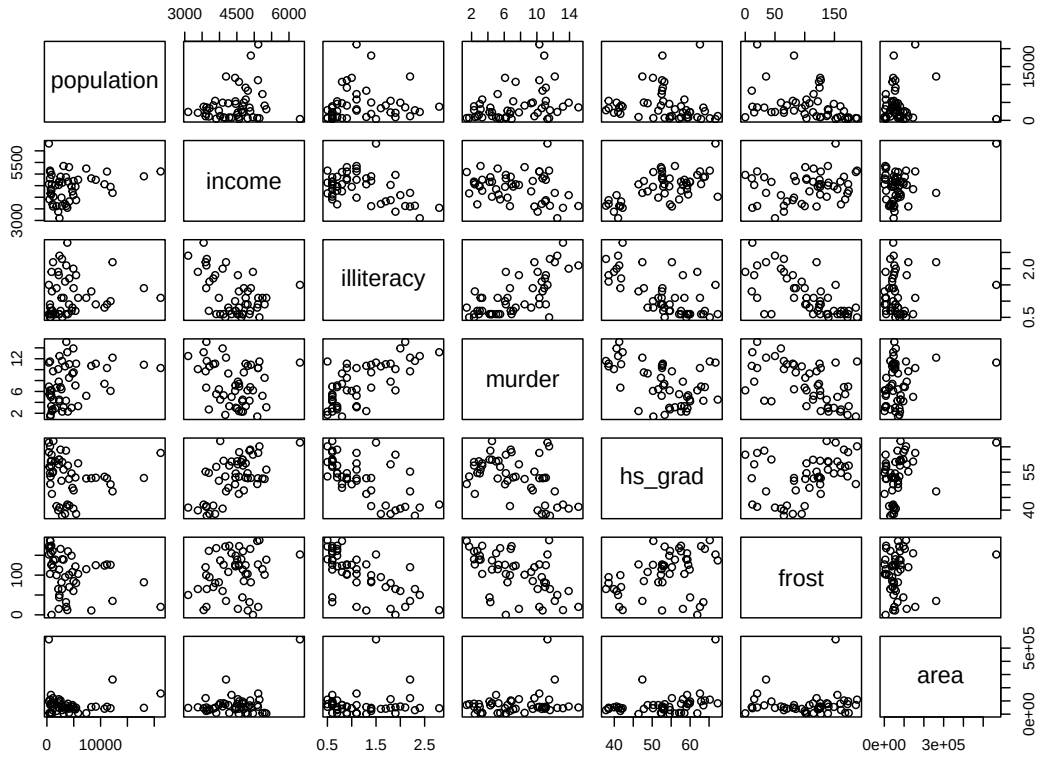



```r
qqnorm(1/(data$life_exp), main = paste("QQ Plot for", "inverse_life_exp")) # inverse
```



Let's look at correlation of the predictors to make sure there isn't collinearity that's obvious at the start.

```
data |>
  select(-life_exp) |>
  pairs()
```



C.

Use automatic procedures to find a 'best subset' of the full model. Backward selection, forward selection, stepwise. It doesn't look like there is any obvious highly correlated variables, so we don't need to remove anything at this point. It looks like the forward and backward step approaches led to the same model with final predictors as `population`, `murder`, `hs_grad`, and `frost`.

```
# backward selection
fit.mult = lm(life_exp ~ log(population) + income + illiteracy + murder + hs_grad + frost + log(area), d
summary(fit.mult)
```

```
Call:
lm(formula = life_exp ~ log(population) + income + illiteracy +
    murder + hs_grad + frost + log(area), data = data)

Residuals:
     Min       1Q   Median       3Q      Max
-1.43084 -0.45559  0.02759  0.49618  1.70215

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)      6.795e+01  2.092e+00  32.478  < 2e-16 ***
log(population)  2.527e-01  1.351e-01   1.870   0.0685 .
income           1.396e-05  2.444e-04   0.057   0.9547
```

```
illiteracy       1.126e-01  3.507e-01   0.321   0.7497
murder          -3.092e-01  4.706e-02  -6.570 6.01e-08 ***
hs_grad          5.278e-02  2.483e-02   2.126   0.0394 *
frost           -4.869e-03  3.215e-03  -1.515   0.1373
log(area)        6.862e-02  1.098e-01   0.625   0.5354
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.7343 on 42 degrees of freedom
Multiple R-squared:  0.7435,    Adjusted R-squared:  0.7008
F-statistic: 17.39 on 7 and 42 DF,  p-value: 1.433e-10
```

```
fit.back = step(fit.mult, direction = 'backward', trace = FALSE)
summary(fit.back)
```

```

Call:
lm(formula = life_exp ~ log(population) + murder + hs_grad +
    frost, data = data)

Residuals:
     Min       1Q   Median       3Q      Max
-1.41760 -0.43880  0.02539  0.52066  1.63048

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)     68.720810   1.416828  48.503  < 2e-16 ***
log(population)  0.246836   0.112539   2.193 0.033491 *
murder          -0.290016   0.035440  -8.183 1.87e-10 ***
hs_grad          0.054550   0.014758   3.696 0.000591 ***
frost           -0.005174   0.002482  -2.085 0.042779 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.7137 on 45 degrees of freedom
Multiple R-squared:  0.7404,    Adjusted R-squared:  0.7173
F-statistic: 32.09 on 4 and 45 DF,  p-value: 1.17e-12
```

```
# forward selection

intercept_only = lm (life_exp ~ 1, data = data, trace = FALSE)
```

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

```
fit.forward = step(intercept_only, direction = "forward", scope = formula(fit.mult), trace = FALSE)
```

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

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

```r
summary(fit.forward)
```

```
Call:
lm(formula = life_exp ~ murder + hs_grad + log(population) +
    frost, data = data, trace = FALSE)

Residuals:
     Min       1Q   Median       3Q      Max
-1.41760 -0.43880  0.02539  0.52066  1.63048

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)     68.720810   1.416828  48.503  < 2e-16 ***
murder          -0.290016   0.035440  -8.183 1.87e-10 ***
hs_grad          0.054550   0.014758   3.696 0.000591 ***
log(population)  0.246836   0.112539   2.193 0.033491 *
frost           -0.005174   0.002482  -2.085 0.042779 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.7137 on 45 degrees of freedom
Multiple R-squared:  0.7404,    Adjusted R-squared:  0.7173
F-statistic: 32.09 on 4 and 45 DF,  p-value: 1.17e-12
```

```r
# stepwise
```

The `log(population)` and `frost` variables are close call variables. So, let's remove therm and refit the model. Removing the variables reduced the Adjusted R-squared value, therefore reducing the performance of the model. So we will keep them

```r
fit.3 = lm(life_exp ~ murder + hs_grad,
    data = data, trace = FALSE)
```

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

```r
summary(fit.3)
```

```
Call:
lm(formula = life_exp ~ murder + hs_grad, data = data, trace = FALSE)

Residuals:
     Min       1Q   Median       3Q      Max
-1.66758 -0.41801  0.05602  0.55913  2.05625
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 70.29708    1.01567  69.213  < 2e-16 ***
murder      -0.23709    0.03529  -6.719 2.18e-08 ***
hs_grad      0.04389    0.01613   2.721  0.00909 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.7959 on 47 degrees of freedom
Multiple R-squared:  0.6628,    Adjusted R-squared:  0.6485
F-statistic:  46.2 on 2 and 47 DF,  p-value: 8.016e-12
```

Let's add an interaction variable between `illiteracy` and `hs_grad` to see if there is an interacting effect there. The interaction term is not significant with a p value of 0.4072 and the addition of the interaction didn't increased the Adjusted R-squared value, so we will not include it.

```
fit.4 = lm(life_exp ~ murder + hs_grad + frost + log(population) + illiteracy + hs_grad * illiteracy, da
summary(fit.4)
```

```
Call:
lm(formula = life_exp ~ murder + hs_grad + frost + log(population) +
    illiteracy + hs_grad * illiteracy, data = data)

Residuals:
     Min       1Q   Median       3Q      Max
-1.43853 -0.46114 -0.00391  0.56559  1.39067

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)        70.314362   2.899659  24.249  < 2e-16 ***
murder             -0.286047   0.042549  -6.723 3.25e-08 ***
hs_grad             0.024206   0.039397   0.614   0.5422
frost              -0.004788   0.003095  -1.547   0.1291
log(population)     0.235437   0.127148   1.852   0.0709 .
illiteracy         -1.155971   1.359603  -0.850   0.3999
hs_grad:illiteracy  0.023582   0.024529   0.961   0.3417
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.7216 on 43 degrees of freedom
Multiple R-squared:  0.7464,    Adjusted R-squared:  0.7111
F-statistic:  21.1 on 6 and 43 DF,  p-value: 2.338e-11
```

D. Use criterion-based procedures to guide your selection of the 'best subset'. Summarize your results (tabular or graphical).

```
fit.5 = MASS::stepAIC(fit.mult, trace = FALSE)
summary(fit.5)
```

```
Call:
```

```
lm(formula = life_exp ~ log(population) + murder + hs_grad +
    frost, data = data)

Residuals:
     Min       1Q   Median       3Q      Max
-1.41760 -0.43880  0.02539  0.52066  1.63048

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)     68.720810   1.416828  48.503  < 2e-16 ***
log(population)  0.246836   0.112539   2.193 0.033491 *
murder          -0.290016   0.035440  -8.183 1.87e-10 ***
hs_grad          0.054550   0.014758   3.696 0.000591 ***
frost           -0.005174   0.002482  -2.085 0.042779 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.7137 on 45 degrees of freedom
Multiple R-squared:  0.7404,    Adjusted R-squared:  0.7173
F-statistic: 32.09 on 4 and 45 DF,  p-value: 1.17e-12
```

```
broom::tidy(fit.5) |> knitr::kable()
```

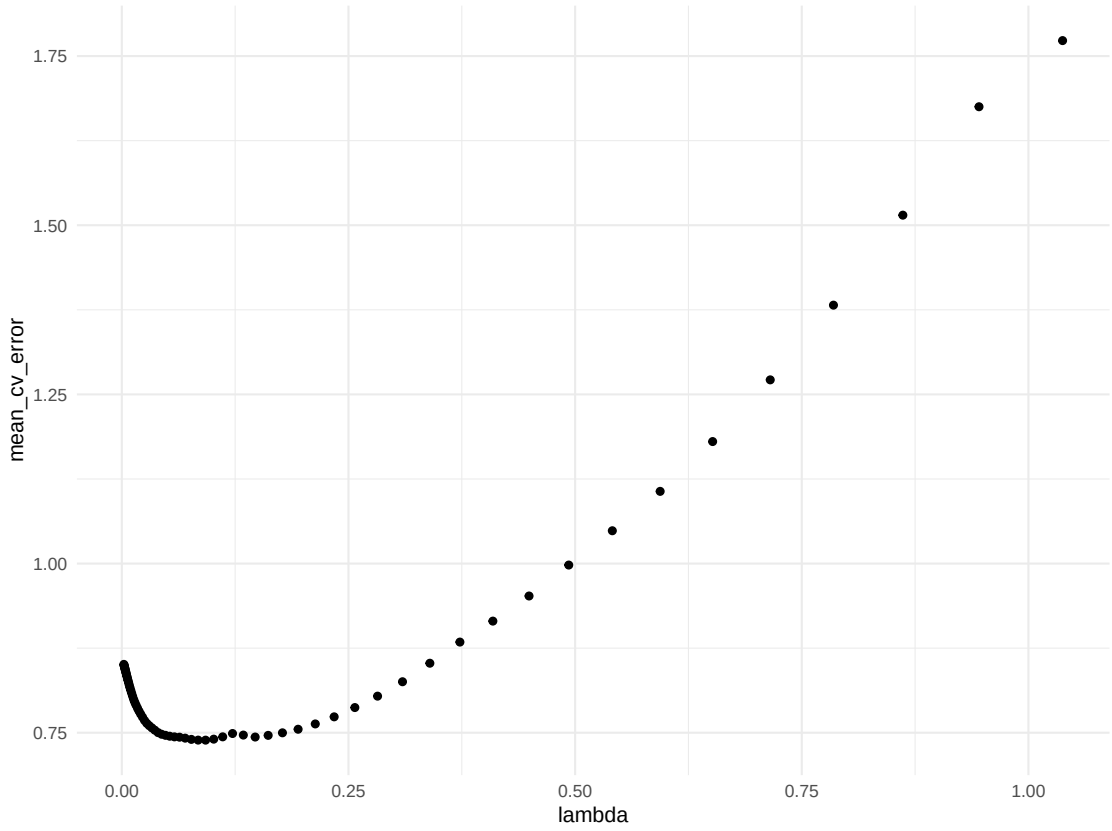
term	estimate	std.error	statistic	p.value
(Intercept)	68.7208105	1.4168278	48.503289	0.0000000
log(population)	0.2468363	0.1125391	2.193338	0.0334912
murder	-0.2900161	0.0354403	-8.183231	0.0000000
hs_grad	0.0545504	0.0147580	3.696332	0.0005915
frost	-0.0051744	0.0024818	-2.084931	0.0427794

E. Use the LASSO method to perform variable selection. Make sure you choose the "best lambda" to use and show how you determined this. It looks like this model has also dropped `income`, `illiteracy` and `area` from the model, which is the same as the stepAIC results.

```
library(glmnet)
```

```
Loading required package: Matrix


Attaching package: 'Matrix'

The following objects are masked from 'package:tidyr':

    expand, pack, unpack

Loaded glmnet 4.1-8
```

```
y = data |> pull(life_exp)

x = data |> select(-life_exp) |>as.matrix()

# find optimal lambda value

cv_model = cv.glmnet(x, y, alpha = 1)

best_lambda = cv_model$lambda.min

# plot the CV results
  tibble(lambda = cv_model$lambda,
         mean_cv_error = cv_model$cvm) |>
    ggplot(aes(x = lambda, y = mean_cv_error)) +
    geom_point()
```




```
# create best model with best lambda

fit.lasso = glmnet(x, y, alpha = 1, lambda = best_lambda)
coef(fit.lasso)
```

```
8 x 1 sparse Matrix of class "dgCMatrix"
                      s0
(Intercept) 70.8740153109
population   0.0000268723
```

```
income       .
illiteracy   .
murder      -0.2475895138
hs_grad      0.0367388352
frost       -0.0022394891
area         .
```

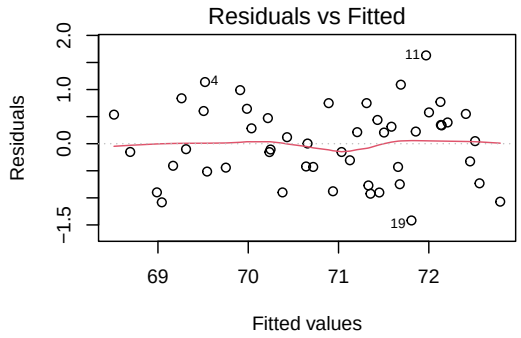
F. Comparing subsets and final model.

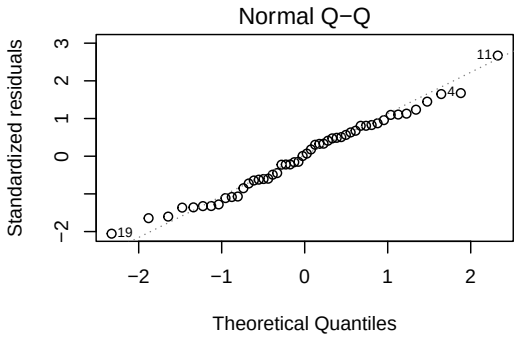
All of the methods (forward, backward, stepAIC, and LASSO) chose the same model with 4 predictors: `population`, `murder`, `hs_grad`, and `frost`. After a boxcox transformation we see that taking the inverse of life expectancy gives us better noramilty with the residuals. We

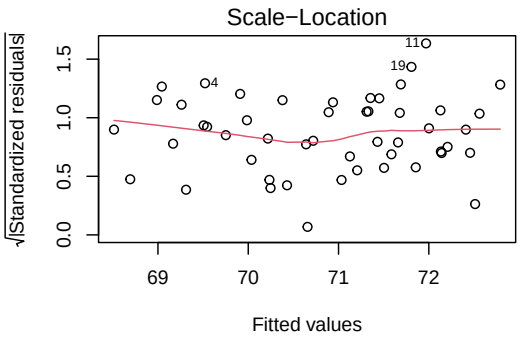
```
par(mfrow = c(2, 2))
# refit linear model with top parameters
fit.best = lm(life_exp ~ log(population) + murder + hs_grad +
    frost, data = data)
summary(fit.best)
```

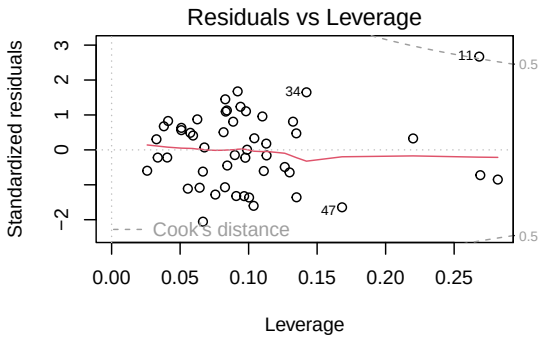
```
Call:
lm(formula = life_exp ~ log(population) + murder + hs_grad +
    frost, data = data)

Residuals:
     Min       1Q   Median       3Q      Max
-1.41760 -0.43880  0.02539  0.52066  1.63048

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)     68.720810   1.416828  48.503  < 2e-16 ***
log(population)  0.246836   0.112539   2.193 0.033491 *
murder          -0.290016   0.035440  -8.183 1.87e-10 ***
hs_grad          0.054550   0.014758   3.696 0.000591 ***
frost           -0.005174   0.002482  -2.085 0.042779 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.7137 on 45 degrees of freedom
Multiple R-squared:  0.7404,    Adjusted R-squared:  0.7173
F-statistic: 32.09 on 4 and 45 DF,  p-value: 1.17e-12
```

```
plot(fit.best)
```







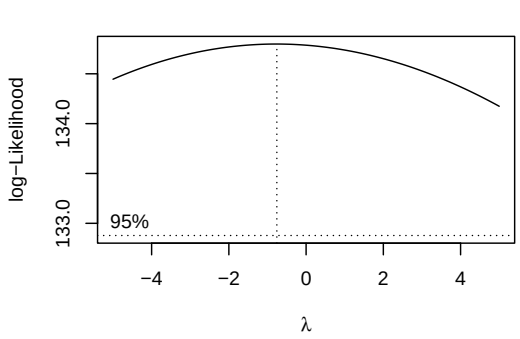


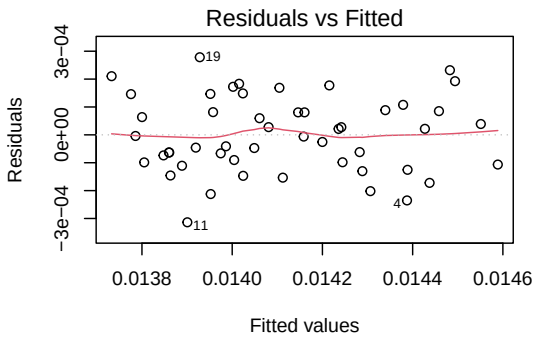

```r
# do boxcox transformation to see if there is a better fit
MASS::boxcox(fit.best, lambda = seq(-5, 5, by = 0.25))

fit.best = lm(1/life_exp ~ log(population) + murder + hs_grad +
    frost, data = data)
summary(fit.best)
```

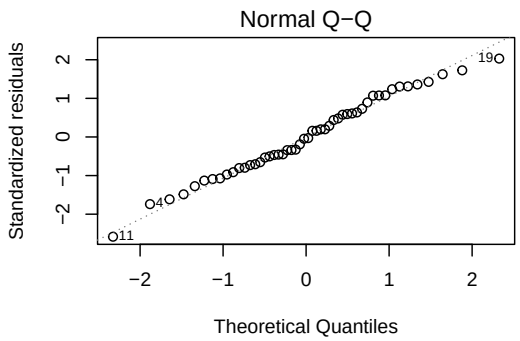
```
Call:
lm(formula = 1/life_exp ~ log(population) + murder + hs_grad +
    frost, data = data)

Residuals:
       Min         1Q     Median         3Q        Max
-3.136e-04 -9.869e-05 -5.110e-06  8.766e-05  2.784e-04

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)     1.457e-02  2.818e-04  51.704  < 2e-16 ***
log(population) -5.131e-05  2.238e-05  -2.292 0.026615 *
murder           5.785e-05  7.049e-06   8.208 1.72e-10 ***
hs_grad         -1.099e-05  2.935e-06  -3.745 0.000511 ***
frost            1.000e-06  4.936e-07   2.027 0.048624 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

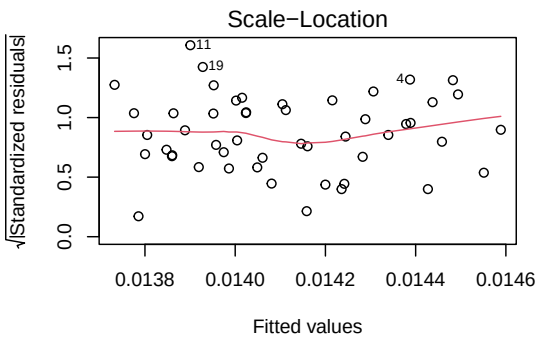
```
Residual standard error: 0.0001419 on 45 degrees of freedom
Multiple R-squared:  0.7429,    Adjusted R-squared:  0.7201
F-statistic: 32.52 on 4 and 45 DF,  p-value: 9.415e-13
```

```r
plot(fit.best)
```



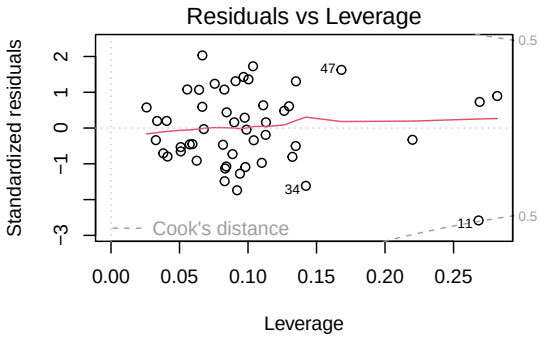







```r
# check collinearity with values >5
vif(fit.best)
```

```
log(population)          murder         hs_grad           frost
       1.321673        1.646571        1.366860        1.600993
```



Cross validation

```
library(caret)
set.seed(12345)
# Use 10-fold validation and create the training sets
  train = trainControl(method = "cv", number = 10)
# Fit the 4-variables model that we discussed in previous lectures
model_caret = train(1/life_exp ~ log(population) + murder + hs_grad + frost,
                    data = data,
                    trControl = train,
                    method = 'lm',
                    na.action = na.pass)
model_caret$finalModel
```

```
Call:
lm(formula = .outcome ~ ., data = dat)

Coefficients:
      (Intercept)  `log(population)`             murder            hs_grad
        1.457e-02         -5.131e-05          5.785e-05         -1.099e-05
            frost
        1.000e-06
```

```
print(model_caret)
```

```
Linear Regression

50 samples
 4 predictor

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 44, 46, 45, 46, 43, 46, ...
Resampling results:

  RMSE          Rsquared  MAE
  0.0001385569  0.754282  0.0001227942

Tuning parameter 'intercept' was held constant at a value of TRUE
```

G. Summary

Using a dataset of 50 U.S. states, we developed a regression model to predict life expectancy in the 1970s. Variables like population, high school graduation, frost, and murder had linear relationships with life expectancy. The population variable showed a distribution that was not norma, but could be addressed by a log transformation. I built the model using forward and backward automatic procedures, as well as stepAIC criterion procdure and, finally LASSO. All models returned the same set of predictors, so that increased confidence in the finally model including log_population, high school graduation, murder and frost. Assessing the model with a boxcox transformation indicated than an inverse transformation of life expectancy gaves us more noramlly distributed residuals. The final model explains 75.43% of the total variance of the dataset. The cross-validation showed that the RMSE and MAE are low and showing that there is low predictive error.