Homework 5

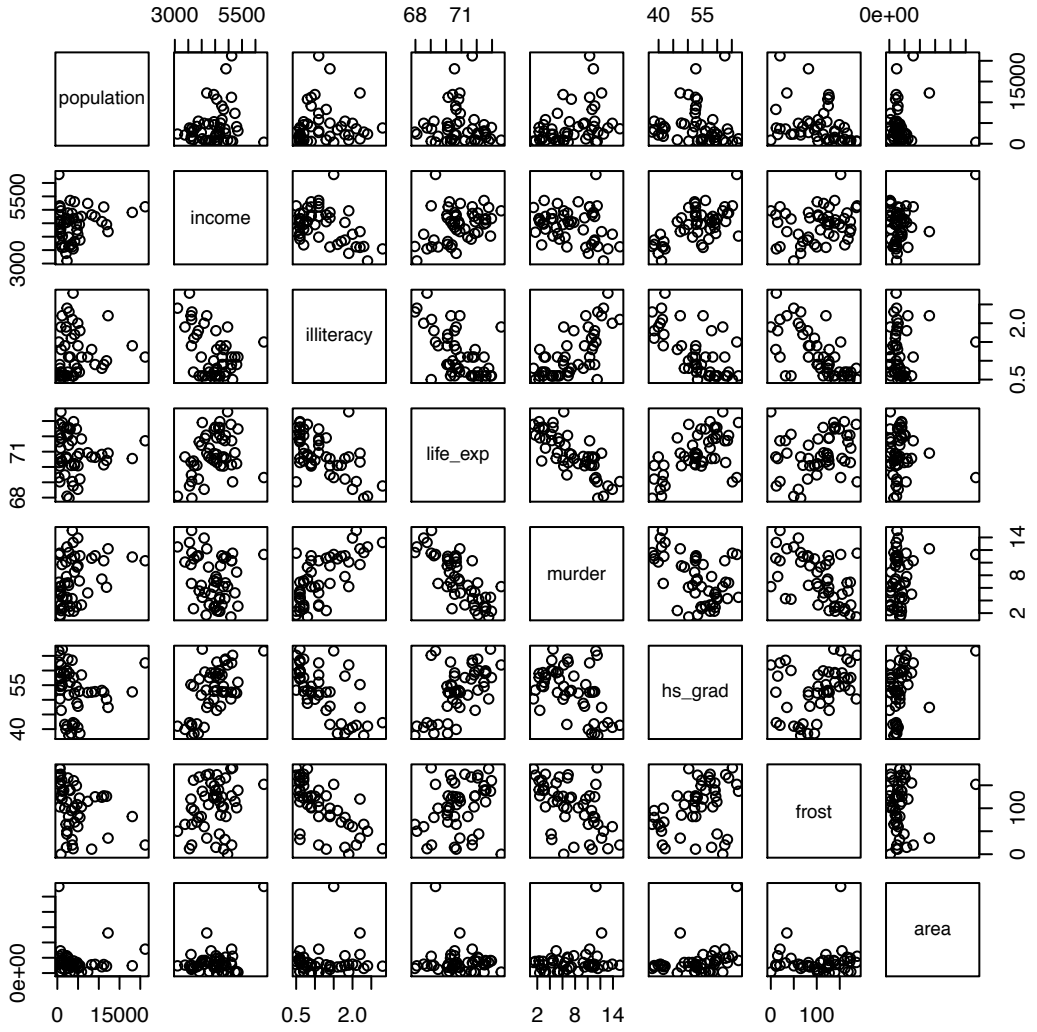
Alyssa Vanderbeek (amv2187)

3 December 2018

1. Explore the dataset and generate appropriate descriptive statistics and relevant graphs for all variables of interest (continuous and categorical) – no test required. Be selective! Even if you create 20 plots, you don't want to show them all.

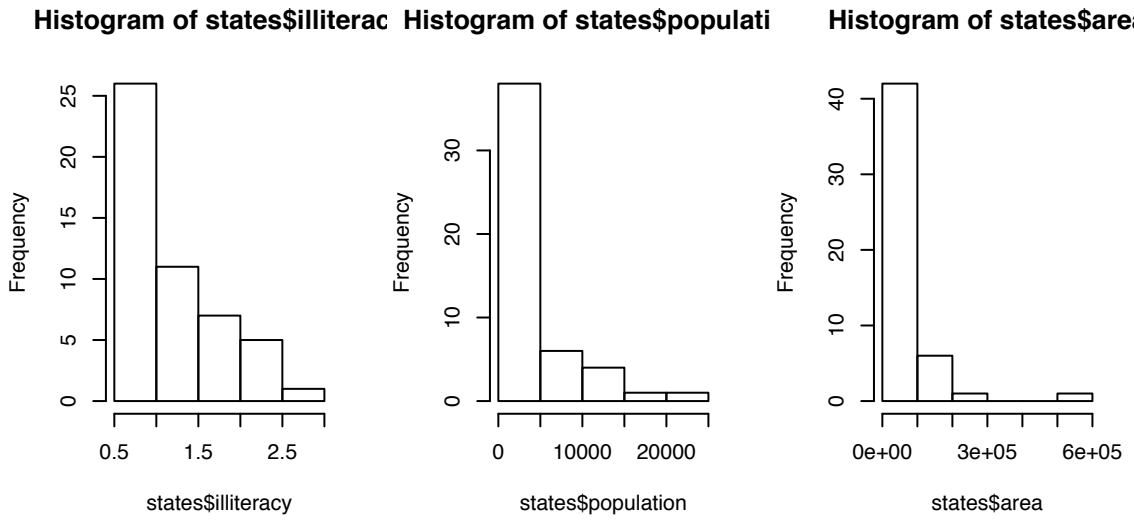
Table 1: Summary statistics

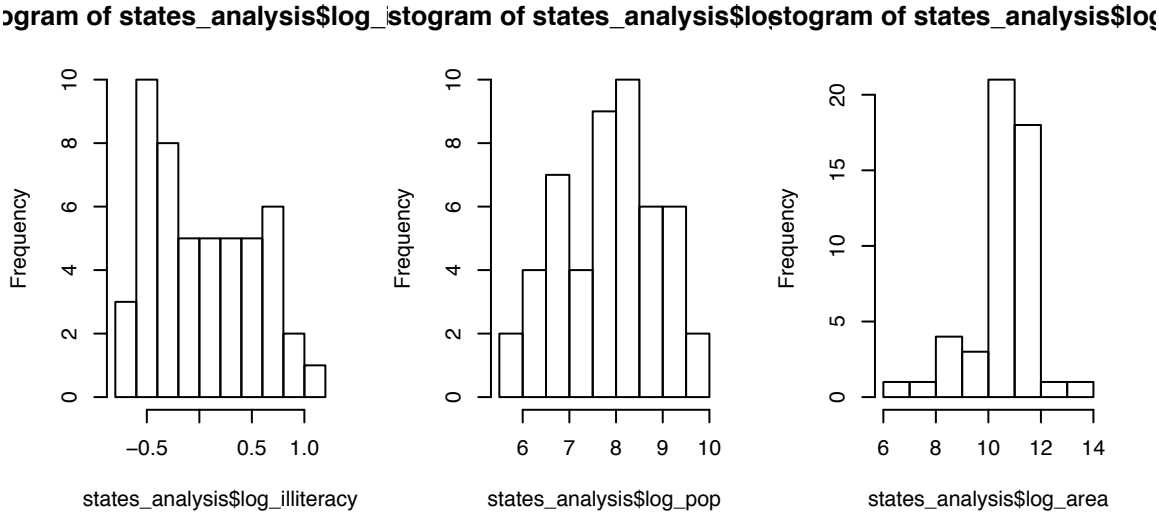
	NA	Mean	Std. Dev.	Min	1st Q	Median	3rd Q	Max
area	0	70735.88	85327.3	1049	36985.25	54277	81162.5	566432
frost	0	104.46	51.98	0	66.25	114.5	139.75	188
hs_grad	0	53.11	8.08	37.8	48.05	53.25	59.15	67.3
illiteracy	0	1.17	0.61	0.5	0.62	0.95	1.58	2.8
income	0	4435.8	614.47	3098	3992.75	4519	4813.5	6315
life_exp	0	70.88	1.34	67.96	70.12	70.67	71.89	73.6
murder	0	7.38	3.69	1.4	4.35	6.85	10.67	15.1
population	0	4246.42	4464.49	365	1079.5	2838.5	4968.5	21198



It looks like murder is correlated both with life expectancy (-0.7808458) and illiteracy (0.7029752), suggesting that it is a potential confounder. Specifically, murder is positively associated with illiteracy (higher murder rate = higher illiteracy rate) and negatively associated with life expectancy (higher murder rate = lower life expectancy).

After examining the distribution of each variable in the dataset, I chose to perform a log transformation on the estimates for area size, illiteracy rate, and population size, which were all skewed.





2. Use automatic procedures to find a 'best subset' of the full model.

Final model using backwards elimination:

```
##
## Call:
## lm(formula = life_exp ~ murder + hs_grad + frost + log_pop, data = states_analysis)
##
## 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 ***
## frost       -0.005174   0.002482  -2.085 0.042779 *
## log_pop      0.246836   0.112539   2.193 0.033491 *
## ---
```

```
## 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
```

Final model using forwards process:

```
##
## Call:
## lm(formula = life_exp ~ murder + hs_grad + log_pop + frost, data = states_analysis)
##
## 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_pop      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
```

Final model using a stepwise process:

```
## Start:  AIC=-23.71
## life_exp ~ income + murder + hs_grad + frost + log_area + log_illiteracy +
##     log_pop
##
##                  Df Sum of Sq    RSS     AIC
## - income          1    0.0002 22.596 -25.712
## - log_illiteracy  1    0.1079 22.704 -25.475
## - log_area        1    0.2368 22.833 -25.192
## <none>                        22.596 -23.713
## - frost           1    1.1645 23.760 -23.200
## - log_pop         1    2.0155 24.611 -21.441
## - hs_grad         1    2.4822 25.078 -20.502
## - murder          1   24.0347 46.631  10.512
##
## Step:  AIC=-25.71
## life_exp ~ murder + hs_grad + frost + log_area + log_illiteracy +
##     log_pop
##
##                  Df Sum of Sq    RSS      AIC
## - log_illiteracy  1    0.1095 22.705 -27.4708
## - log_area        1    0.2616 22.858 -27.1370
## <none>                        22.596 -25.7125
## - frost           1    1.2628 23.859 -24.9936
## - log_pop         1    2.3859 24.982 -22.6937
```

```
## - hs_grad         1    4.4112 27.007 -18.7959
## - murder          1   24.4834 47.079   8.9907
##
## Step:  AIC=-27.47
## life_exp ~ murder + hs_grad + frost + log_area + log_pop
##
##            Df Sum of Sq    RSS     AIC
## - log_area  1    0.2157 22.921 -28.998
## <none>                  22.705 -27.471
## - log_pop   1    2.2792 24.985 -24.688
## - frost     1    2.3760 25.082 -24.495
## - hs_grad   1    4.9491 27.655 -19.612
## - murder    1   29.2296 51.935  11.899
##
## Step:  AIC=-29
## life_exp ~ murder + hs_grad + frost + log_pop
##
##           Df Sum of Sq    RSS     AIC
## <none>                 22.921 -28.998
## - frost    1     2.214 25.135 -26.387
## - log_pop  1     2.450 25.372 -25.920
## - hs_grad  1     6.959 29.881 -17.741
## - murder   1    34.109 57.031  14.578
```

(a) Do the procedures generate the same model?

All automatic processes conclude the same model, using percent increase in population size (log(population)), rate of high school graduation (hs_grad), murder rate per 100,000 (murder), and average number of days annually with temperatures below freezing (frost) as predictors of life expectancy.

(b) Is there any variable a close call? What was your decision: keep or discard? Provide arguments for your choice. (Note: this question might have more or less relevance depending on the 'subset' you choose).

No variables were seen to be a "close call" at the 5% significance level. 'Frost' is the least significant predictor, with a p-value of 0.043.

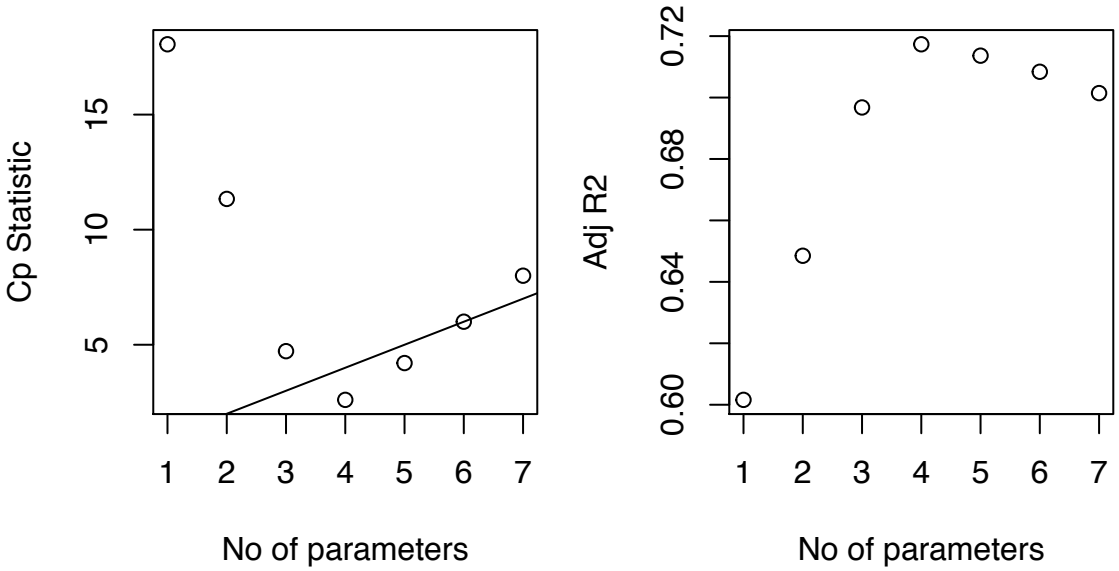
(c) Is there any association between 'Illiteracy' and 'HS graduation rate'? Does your 'subset' contain both?

There is an observed correlation between illiteracy (with and without log transformation) and high school graduation rate (-0.6571886). This is intuitive in that we would expect that the more people who graduate from high school (high HS graduation rate), there are fewer people who are illiterate. However, my model includes only high school graduation rate as a predictor.

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

Table 2: Criterion-based model building

p	(Intercept)	income	murder	hs_grad	frost	log_area	log_illiteracy	log_pop	rss	rsq	adjr2	cp	bic
1	1	0	1	0	0	0	0	0	34.46133	0.6097201	0.6015893	18.054999	-39.22051
2	1	0	1	1	0	0	0	0	29.77036	0.6628461	0.6484991	11.335656	-42.62472
3	1	0	1	1	0	0	0	1	25.13538	0.7153378	0.6967729	4.720403	-47.17452
4	1	0	1	1	1	0	0	1	22.92123	0.7404135	0.7173392	2.604837	-47.87315
5	1	0	1	1	1	1	0	1	22.70549	0.7428568	0.7136360	4.203829	-44.43397
6	1	0	1	1	1	1	1	1	22.59600	0.7440968	0.7083894	6.000318	-40.76364
7	1	1	1	1	1	1	1	1	22.59583	0.7440987	0.7014485	8.000000	-36.85199

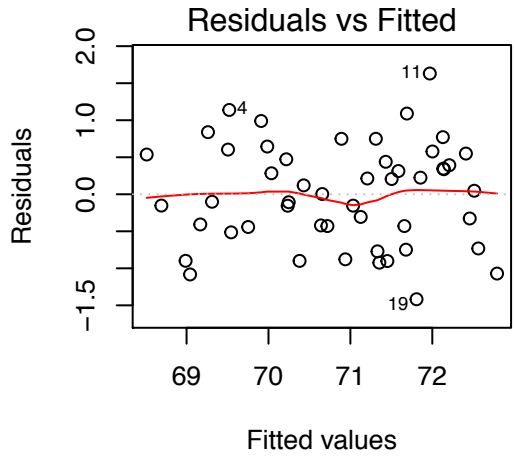


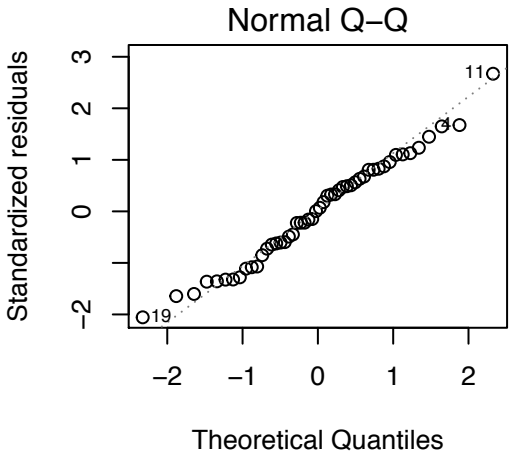
According to the Cp statistics and Adjusted R^2, the ideal number of parameters is 4; as seen in the table above, those parameters are murder, hs_grad, frost, and log_pop - the same as what was concluded in the automatic process.

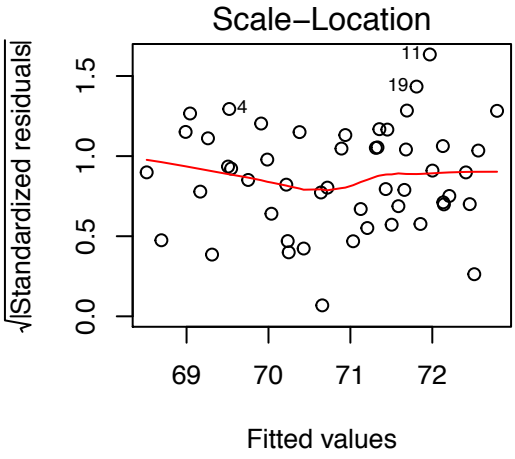
4. Compare the two 'subsets' from parts 2 and 3 and recommend a 'final' model. Using this 'final' model do the following. a) Identify any leverage and/or influential points and take appropriate measures. b) Check the model assumptions.

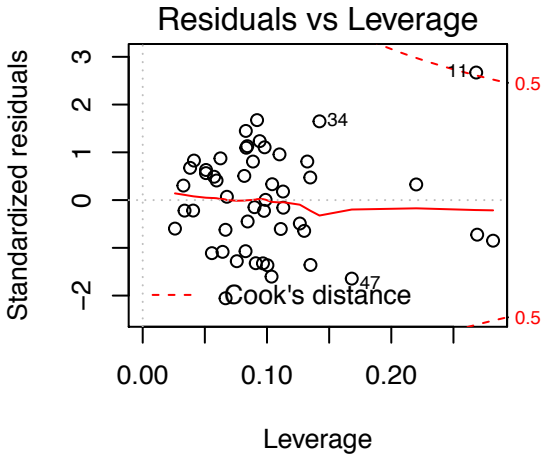
All analyses above recommend the same model using percent increase in population size (log(population)), rate of high school graduation (hs_grad), murder rate per 100,000 (murder), and average number of days annually with temperatures below freezing (frost) as predictors of life expectancy.

The 11th entry (Hawaii) showed evidence of being an influential outlier (according to Cook's distance and measure of influence). To check to see whether this entry had a significant impact on the model and its assumptions, I compared diagnostics of the model with and without the 11th point. The diagnostic plots below show that, in fact, the model assumptions (1. residuals have mean zero, 2. residuals have equal variance, 3. residuals are independent) are met for both models - with and without the potential influential point. However, we can see that without the point, the 'frost' variable is no longer a significant predictor of life expectancy, with a p-value of 0.53.



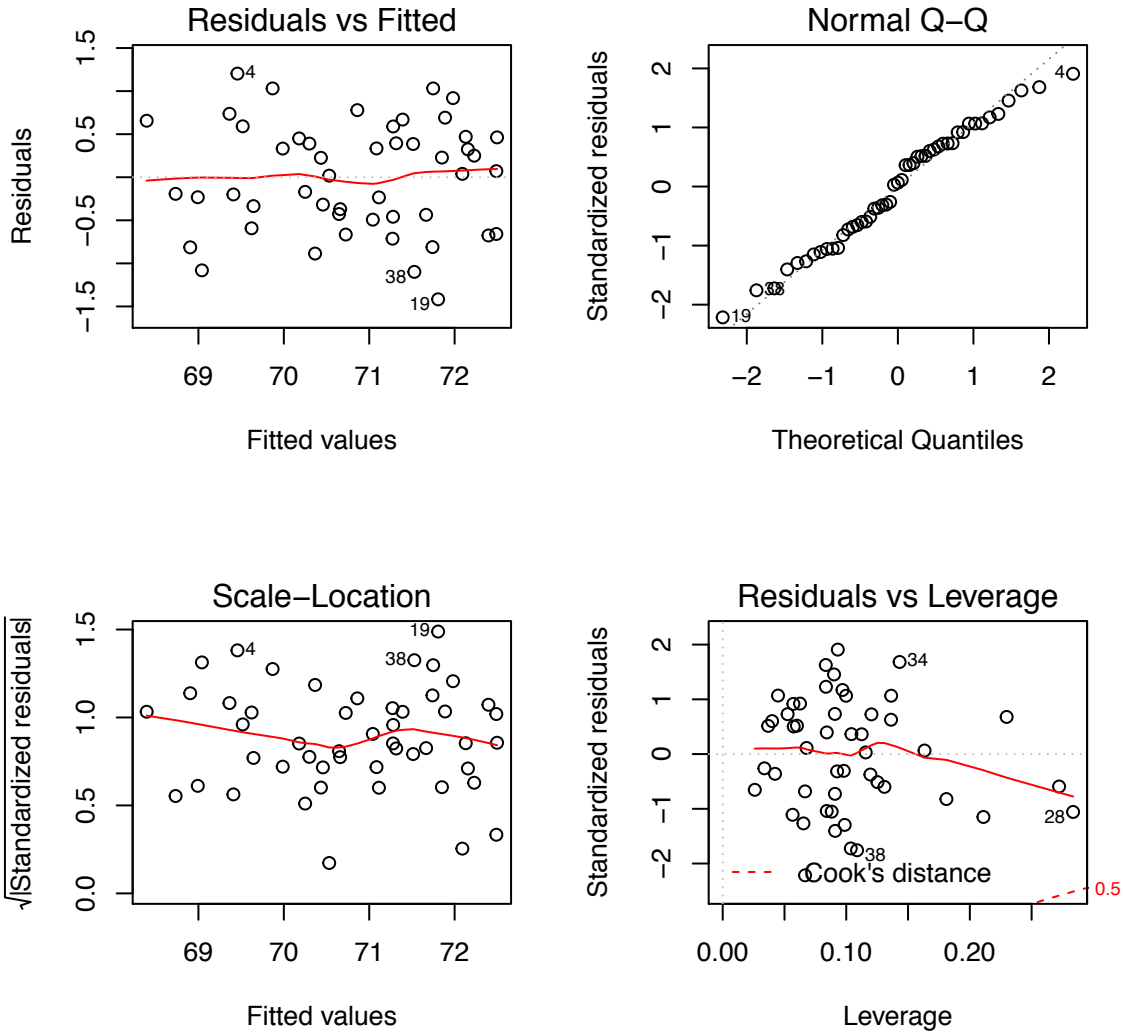







```
##
## Call:
## lm(formula = life_exp ~ murder + hs_grad + log_pop + frost, data = states_analysis[-11,
##     ])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.41708 -0.45880  0.03924  0.46286  1.20332
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 67.906960   1.344438  50.510  < 2e-16 ***
## murder      -0.276679   0.033203  -8.333 1.35e-10 ***
## hs_grad      0.046799   0.013953   3.354  0.00165 **
## log_pop      0.337449   0.109043   3.095  0.00342 **
## frost       -0.001632   0.002610  -0.625  0.53499
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6621 on 44 degrees of freedom
## Multiple R-squared:  0.7611, Adjusted R-squared:  0.7394
```

```
## F-statistic: 35.05 on 4 and 44 DF,  p-value: 3.709e-13
```



Using the 'final' model chosen in part 4, focus on MSE to test the model predictive ability

(a) Use a 10-fold cross-validation (10 repeats).

```
##         RMSE       MSE    RMSESD  std.error
## 1  0.7524920 0.5662442 0.2755474 0.03061638
## 2  0.7510947 0.5641433 0.1558614 0.01731793
## 3  0.7436424 0.5530040 0.2433998 0.02704442
## 4  0.7584087 0.5751837 0.1412344 0.01569271
## 5  0.7576462 0.5740278 0.2156512 0.02396125
## 6  0.7565450 0.5723603 0.2247771 0.02497524
## 7  0.7517577 0.5651396 0.2362289 0.02624765
## 8  0.7347655 0.5398804 0.2517595 0.02797328
## 9  0.7850556 0.6163123 0.1649644 0.01832938
## 10 0.7506681 0.5635027 0.2157190 0.02396878
```

```
##         mse        se
## 1 0.5689798 0.0236127
```

(b) Experiment a new, but simple bootstrap technique called "residual sampling". Summarize the MSE.

```
## # A tibble: 1 x 4
##    RMSE     bias std.error   MSE
##   <dbl>    <dbl>     <dbl> <dbl>
## 1 0.689 -0.00659    0.0422 0.475
```

```
## # A tibble: 1 x 4
##    RMSE    bias std.error   MSE
##   <dbl>   <dbl>     <dbl> <dbl>
## 1 0.730 -0.0599    0.0289 0.533
```

The MSE of the bootstrap is slightly smaller than that of the 10-fold CV, however the standard error of this estimate is lower for the 10-fold CV than for the bootstrap. This is reflective of the bias-variance tradeoff, and we can conclude then that the 10-fold CV has larger bias and smaller variance, and the residual bootstrap has smaller bias and larger variance. Which method we use to cross validate the model will depend on which of these two (bias vs. variance) we are more interested in minimizing. However, as both methods are computationally inexpensive and produce slightly different MSE (one more conservative than the other), I recommend using both to test the model.

Overall, the MSE is a bit high for the model built above. After testing the predictive ability here, I would return to the model building steps, and examine potential for effect modifiers or higher order terms. On the other hand, it may be that this model is the best we can build for the outcome with the provided predictors. After all, the criterion-based analysis suggested that 4 was the ideal number of predictors.

```
library(tidyverse)
library(faraway)
library(HH)
library(leaps)
library(caret)

states = state.x77 %>% # load data from faraway package
  as.data.frame() %>%
  janitor::clean_names()

# table of summary stats
states %>%
  skimr::skim_to_list() %>%
  as.data.frame %>%
  dplyr::select(1, 2, 5:11) %>%
  `colnames<-`(c(' ', 'NA', 'Mean', 'Std. Dev.', 'Min', '1st Q',
'Median', '3rd Q', 'Max')) %>%
  knitr::kable(caption = 'Summary statistics')

# scatterplot to assess correlation between vars
states %>% pairs

# correlation matrix to evaluate what is seen in scatterplots
# states %>%
#   cor

states_analysis = states %>%
  mutate(log_area = log(area),
         log_illiteracy = log(illiteracy),
         log_pop = log(population)) %>%
  dplyr::select(-area, -population, -illiteracy)

par(mfrow = c(2, 3))
hist(states$illiteracy)
hist(states$population)
hist(states$area)
hist(states_analysis$log_illiteracy)
hist(states_analysis$log_pop)
hist(states_analysis$log_area)

## backwards elimination
# summary(lm(life_exp ~ ., data = states_analysis))
# summary(lm(life_exp ~ murder + hs_grad + frost + log_area +
log_illiteracy + log_pop, data = states_analysis))
# summary(lm(life_exp ~ murder + hs_grad + frost + log_illiteracy +
log_pop, data = states_analysis))

b.fit = lm(life_exp ~ murder + hs_grad + frost + log_pop, data =
states_analysis)
summary(b.fit)
```

```
## forwards process
# summary(lm(life_exp ~ murder, data = states_analysis))
# summary(lm(life_exp ~ hs_grad, data = states_analysis))
# summary(lm(life_exp ~ frost, data = states_analysis))
# summary(lm(life_exp ~ log_area, data = states_analysis))
# summary(lm(life_exp ~ log_illiteracy, data = states_analysis))
# summary(lm(life_exp ~ log_pop, data = states_analysis))
#
# # murder has lowest p-val. Start adding secondary vars
# summary(lm(life_exp ~ murder + hs_grad, data = states_analysis))
# summary(lm(life_exp ~ murder + frost, data = states_analysis))
# summary(lm(life_exp ~ murder + log_area, data = states_analysis))
# summary(lm(life_exp ~ murder + log_illiteracy, data =
states_analysis))
# summary(lm(life_exp ~ murder + log_pop, data = states_analysis))
#
# # murder + hs_grad
# summary(lm(life_exp ~ murder + hs_grad + frost, data =
states_analysis))
# summary(lm(life_exp ~ murder + hs_grad + log_area, data =
states_analysis))
# summary(lm(life_exp ~ murder + hs_grad + log_illiteracy, data =
states_analysis))
# summary(lm(life_exp ~ murder + hs_grad + log_pop, data =
states_analysis))
#
# # murder + hs_grad + log_pop
# summary(lm(life_exp ~ murder + hs_grad + log_pop + frost, data =
states_analysis))
# summary(lm(life_exp ~ murder + hs_grad + log_pop + log_area, data =
states_analysis))
# summary(lm(life_exp ~ murder + hs_grad + log_pop + log_illiteracy,
data = states_analysis))
#
# # murder + hs_grad + log_pop + frost
# summary(lm(life_exp ~ murder + hs_grad + log_pop + frost + log_area,
data = states_analysis))
# summary(lm(life_exp ~ murder + hs_grad + log_pop + frost +
log_illiteracy, data = states_analysis))

f.fit = lm(life_exp ~ murder + hs_grad + log_pop + frost, data =
states_analysis)
summary(f.fit)

## Stepwise
step.fit = step(lm(life_exp ~ ., data = states_analysis))

# function to select the 'best' model
best <- function(model, ...)
```

```
{
  subsets <- regsubsets(formula(model), model.frame(model), ...)
  subsets <- with(summary(subsets),
                  cbind(p = as.numeric(rownames(which)), which, rss,
rsq, adjr2, cp, bic))

  return(subsets)
}

best(lm(life_exp ~ ., data = states_analysis)) %>%
  knitr::kable(., 'latex', caption = 'Criterion-based model building')
%>%
  kableExtra::kable_styling(latex_options = c("hold_position")) %>%
  kableExtra::landscape()

# leaps::leaps(x = states_analysis[, c(1, 3:8)], y =
states_analysis$life_exp, nbest = 2, method = "Cp")

# leaps::leaps(x = states_analysis[, c(1, 3:8)], y =
states_analysis$life_exp, nbest = 2, method = "adjr2")

# Summary of models for each size (one model per size)
b = leaps::regsubsets(life_exp ~ ., data = states_analysis)
rs = summary(b)

# Plots of Cp and Adj-R2 as functions of parameters
par(mar = c(4, 4, 1, 1))
par(mfrow = c(1, 2))

plot(1:7, rs$cp, xlab = "No of parameters", ylab = "Cp Statistic")
abline(0, 1)
plot(1:7, rs$adjr2, xlab = "No of parameters", ylab = "Adj R2")

life_exp_fit = b.fit

# rstandard function gives the INTERNALLY studentized residuals

stu_res = rstandard(life_exp_fit)
outliers_y = stu_res[abs(stu_res) > 2.5]

# Measures of influence:
# Gives DFFITS, Cook's Distance, Hat diagonal elements, and others.

# influence.measures(life_exp_fit)

# Look at the Cook's distance lines / influential point output and
notice obs 11 as potential Y outlier / influential point

par(mfrow = c(2, 2))
plot(life_exp_fit)
```

```
# Examine results with and without observations 5 and 28 that have
very high survivals (>2000)
fit_nooutlier = lm(life_exp ~ murder + hs_grad + log_pop + frost, data
= states_analysis[-11, ])
summary(fit_nooutlier) # look at the results of the fitted model
without the influential point

plot(fit_nooutlier)

## 10-fold CV
kfold_cv = lapply(1:10, function(i){
  # create 10-fold training datasets
  data_train <- trainControl(method = "cv", number = 10)

  # Fit the model used above
  model_caret <- train((life_exp ~ murder + hs_grad + log_pop +
frost),
                       data = states_analysis,
                       trControl = data_train,
                       method = 'lm',
                       na.action = na.pass)

  #return(list(model_caret$results, model_caret$resample))
  return(model_caret$results)
})

do.call("rbind", kfold_cv) %>%
  dplyr::select(RMSE, RMSESD) %>% # summarise(mse = mean(RMSE))
  mutate(MSE = RMSE^2,
         std.error = RMSESD / 9) %>%
  dplyr::select(1, 3, 2, 4) # %>% summarise(se = mean(std.error))

do.call("rbind", kfold_cv) %>%
  dplyr::select(RMSE, RMSESD) %>% # summarise(mse = mean(RMSE))
  mutate(MSE = RMSE^2,
         std.error = RMSESD / 9) %>%
  dplyr::select(1, 3, 2, 4) %>% summarise(mse = mean(MSE),
                                          se = mean(std.error))

## Bootstrap
set.seed(1)

# Perform a regression model with the original sample; calculate
predicted values and residuals.
states_analysis = states_analysis %>%
  modelr::add_predictions(life_exp_fit) %>% # add predicted
birthweight
  modelr::add_residuals(life_exp_fit) %>% # residual of observed bwt -
predicted bwt
```

```
  rename('pred1' = pred)


# function to bootstrap residuals and regress new predictions
boot.res <- function(data, index){
  data = data %>%
    rowwise %>%
    mutate(rand_res = sample(resid, replace = T, size = 1), # Randomly
resample the residuals (with replacement), but leave the X values and
predicted values unchanged.
           boot_y = pred1 + rand_res) %>% # New observations by adding
the original predicted values to the bootstrap residuals
    modelr::add_predictions(lm(boot_y ~ murder + hs_grad + log_pop +
frost, data = .)) %>%
    mutate(sq = (boot_y - pred)^2)

  mse = (1/(nrow(data)) * sum(data$sq)) # calculate mse
  root.mse = sqrt(mse) # rmse
  return(root.mse)
}

broom::tidy(boot::boot(states_analysis, boot.res, 10)) %>%
  rename('RMSE' = statistic) %>%
  mutate(MSE = RMSE^2)
broom::tidy(boot::boot(states_analysis, boot.res, 1000)) %>%
  rename('RMSE' = statistic) %>%
  mutate(MSE = RMSE^2)
```