p8130 Homework 5

Eleanor Zhang uni: zz2602

12/1/2018

Read Data

R dataset '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.

Here the main response is life expectancy. The rest variables constitute the pool of variables that may be selected for regression model.

```
library(faraway)
data(state)
state <- as.tibble(state.x77) %>%
  janitor::clean_names() # clean variable names
```

Explore the data

data description

```
str(state) # 50 rows, 8 variables
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    50 obs. of  8 variables:
##  $ population: num  3615 365 2212 2110 21198 ...
##  $ income    : num  3624 6315 4530 3378 5114 ...
##  $ illiteracy: num  2.1 1.5 1.8 1.9 1.1 0.7 1.1 0.9 1.3 2 ...
##  $ life_exp  : num  69 69.3 70.5 70.7 71.7 ...
##  $ murder    : num  15.1 11.3 7.8 10.1 10.3 6.8 3.1 6.2 10.7 13.9 ...
##  $ hs_grad   : num  41.3 66.7 58.1 39.9 62.6 63.9 56 54.6 52.6 40.6 ...
##  $ frost     : num  20 152 15 65 20 166 139 103 11 60 ...
##  $ area      : num  50708 566432 113417 51945 156361 ...
```

The dataset contains 50 observations and 8 variables

Data description:

- population: population estimate as of July 1, 1975
- income: per capita income (1974)
- illiteracy: illiteracy (1970, percent of population)
- life_exp (main response): life expectancy in years (1969–71)
- murder: murder and non-negligent manslaughter rate per 100,000 population (1976)
- hs_grad: percent high-school graduates (1970)
- frost: mean number of days with minimum temperature below freezing (1931–1960) in capital or large city
- area: land area in square miles

Problem 1 Explore the data and summary

Number summary

```
summary(state)
```

```
##    population        income       illiteracy       life_exp
##  Min.   :  365   Min.   :3098   Min.   :0.500   Min.   :67.96
##  1st Qu.: 1080   1st Qu.:3993   1st Qu.:0.625   1st Qu.:70.12
##  Median : 2838   Median :4519   Median :0.950   Median :70.67
```

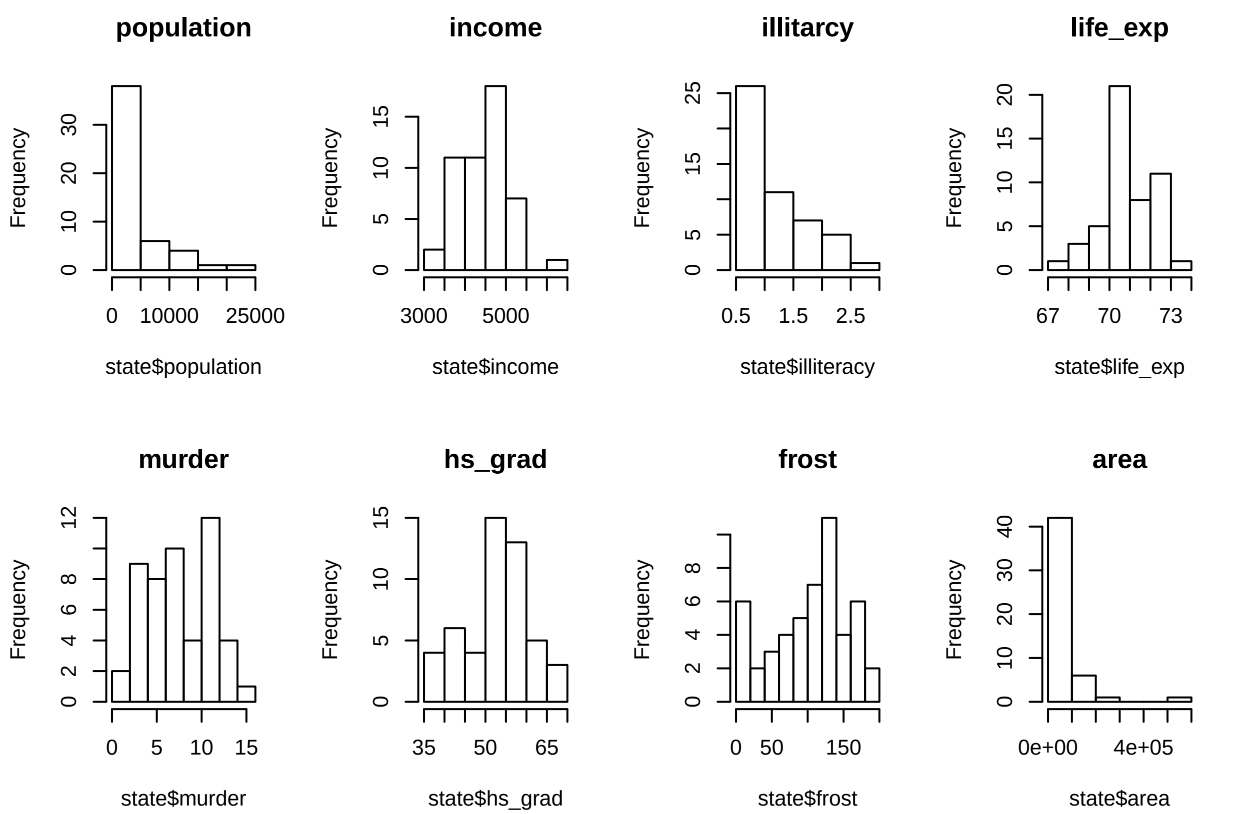
```
##  Mean   : 4246   Mean   :4436   Mean   :1.170   Mean   :70.88
##  3rd Qu.: 4968   3rd Qu.:4814   3rd Qu.:1.575   3rd Qu.:71.89
##  Max.   :21198   Max.   :6315   Max.   :2.800   Max.   :73.60
##      murder          hs_grad          frost             area
##  Min.   : 1.400   Min.   :37.80   Min.   :  0.00   Min.   :  1049
##  1st Qu.: 4.350   1st Qu.:48.05   1st Qu.: 66.25   1st Qu.: 36985
##  Median : 6.850   Median :53.25   Median :114.50   Median : 54277
##  Mean   : 7.378   Mean   :53.11   Mean   :104.46   Mean   : 70736
##  3rd Qu.:10.675   3rd Qu.:59.15   3rd Qu.:139.75   3rd Qu.: 81162
##  Max.   :15.100   Max.   :67.30   Max.   :188.00   Max.   :566432
```

```r
anyNA(state) # NO missing value
```

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

Display distributin of variables in order described above

```r
par(mfrow = c(2,4))
hist(state$population, main = "population")
hist(state$income, main = "income")
hist(state$illiteracy, main = "illitarcy")
hist(state$life_exp, main = "life_exp")
hist(state$murder, main = "murder")
hist(state$hs_grad, main = "hs_grad")
hist(state$frost, main = "frost")
hist(state$area, main = "area")
```

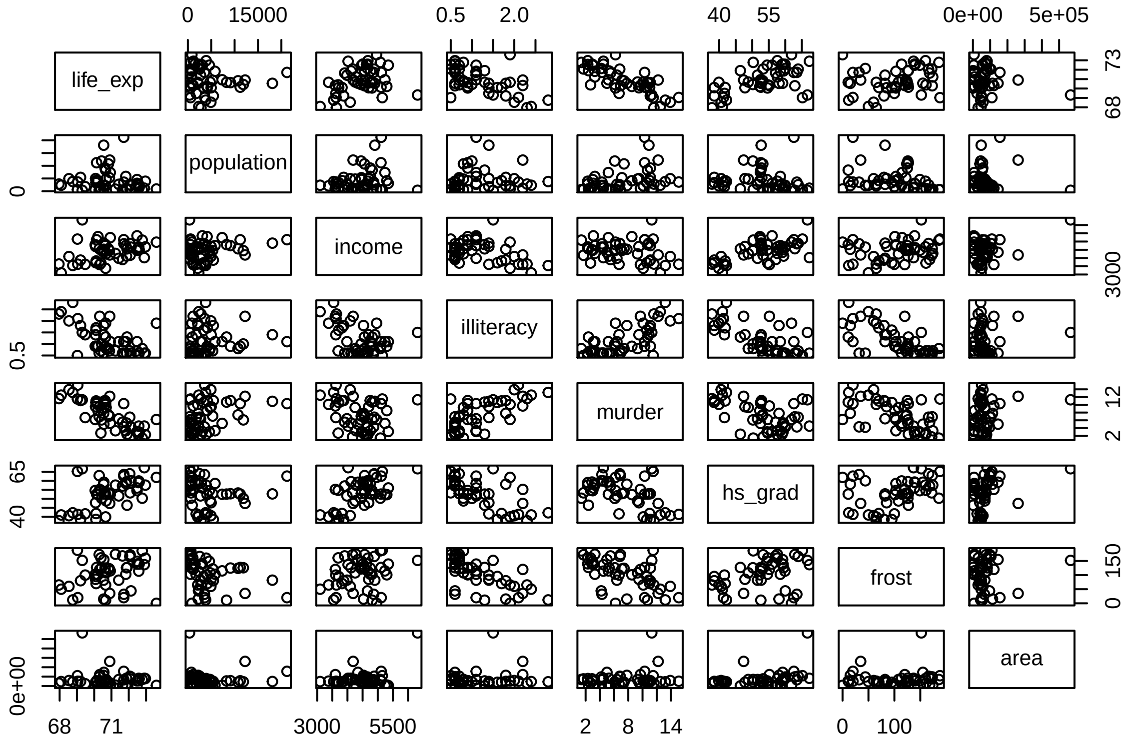


Observe:

- skewed: population size, illteracy, area (reported by median and IQR)
- the other distribution looks evenly shaped (reported by mean and sd)

relationship between covariates

```r
state %>% select(life_exp, everything()) %>% pairs()
```




```r
cor(state) %>% knitr::kable()
```

	population	income	illiteracy	life_exp	murder	hs_grad	frost	area
population	1.0000000	0.2082276	0.1076224	-0.0680520	0.3436428	-0.0984897	-0.3321525	0.0225438
income	0.2082276	1.0000000	-0.4370752	0.3402553	-0.2300776	0.6199323	0.2262822	0.3633154
illiteracy	0.1076224	-0.4370752	1.0000000	-0.5884779	0.7029752	-0.6571886	-0.6719470	0.0772611
life_exp	-0.0680520	0.3402553	-0.5884779	1.0000000	-0.7808458	0.5822162	0.2620680	-0.1073319
murder	0.3436428	-0.2300776	0.7029752	-0.7808458	1.0000000	-0.4879710	-0.5388834	0.2283902
hs_grad	-0.0984897	0.6199323	-0.6571886	0.5822162	-0.4879710	1.0000000	0.3667797	0.3335419
frost	-0.3321525	0.2262822	-0.6719470	0.2620680	-0.5388834	0.3667797	1.0000000	0.0592291
area	0.0225438	0.3633154	0.0772611	-0.1073319	0.2283902	0.3335419	0.0592291	1.0000000

Observe:

- murder and illiteracy seems to have exponential relation
- Area may need to be categorized
- life expectancy are negatively and linearly associated with murder rate and illiteracy repectively. There is some positive linear relation between life expectancy and high school graduates percentage and frost days.
- Some potential colinearity: hs_grad and income, hs_grad and illiteracy,

Problem 2 Automatic procedure

```r
multi.fit <- lm(life_exp ~ ., data = state)
summary(multi.fit)
```

```
##
## Call:
## lm(formula = life_exp ~ ., data = state)
##
## Residuals:
```

```
##      Min       1Q   Median       3Q      Max
## -1.48895 -0.51232 -0.02747  0.57002  1.49447
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  7.094e+01  1.748e+00  40.586  < 2e-16 ***
## population   5.180e-05  2.919e-05   1.775   0.0832 .
## income      -2.180e-05  2.444e-04  -0.089   0.9293
## illiteracy   3.382e-02  3.663e-01   0.092   0.9269
## murder      -3.011e-01  4.662e-02  -6.459 8.68e-08 ***
## hs_grad      4.893e-02  2.332e-02   2.098   0.0420 *
## frost       -5.735e-03  3.143e-03  -1.825   0.0752 .
## area        -7.383e-08  1.668e-06  -0.044   0.9649
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7448 on 42 degrees of freedom
## Multiple R-squared:  0.7362, Adjusted R-squared:  0.6922
## F-statistic: 16.74 on 7 and 42 DF,  p-value: 2.534e-10
```

Comment: murder is the most significant predictor. hs_grad is significant at 0.05 level. The other predictors are not very significant when including all other variables in the model. The adjusted R-square is penalized such that it is significantly smaller than the unadjusted one. This implies we have included unnecessary predictors in the model.

1) **Method I: Backward elimination (choose alpha_to_remove > 0.2)**

Start from there, we use backward elimination to find the "best" subset:

By looking at the summary of full model regression, backward elimination starts eliminating the one with largest p value, so we **remove area** first

```
step1 <- update(multi.fit, . ~ . -area)
summary(step1)
```

```
##
## Call:
## lm(formula = life_exp ~ population + income + illiteracy + murder +
##     hs_grad + frost, data = state)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.49047 -0.52533 -0.02546  0.57160  1.50374
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  7.099e+01  1.387e+00  51.165  < 2e-16 ***
## population   5.188e-05  2.879e-05   1.802   0.0785 .
## income      -2.444e-05  2.343e-04  -0.104   0.9174
## illiteracy   2.846e-02  3.416e-01   0.083   0.9340
## murder      -3.018e-01  4.334e-02  -6.963 1.45e-08 ***
## hs_grad      4.847e-02  2.067e-02   2.345   0.0237 *
## frost       -5.776e-03  2.970e-03  -1.945   0.0584 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7361 on 43 degrees of freedom
## Multiple R-squared:  0.7361, Adjusted R-squared:  0.6993
## F-statistic: 19.99 on 6 and 43 DF,  p-value: 5.362e-11
```

Then we **remove illiteracy**

```
step2 <- update(step1, . ~ . -illiteracy)
summary(step2)
```

```
##
## Call:
## lm(formula = life_exp ~ population + income + murder + hs_grad +
##     frost, data = state)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.4892 -0.5122 -0.0329  0.5645  1.5166
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  7.107e+01  1.029e+00  69.067  < 2e-16 ***
## population   5.115e-05  2.709e-05   1.888   0.0657 .
## income      -2.477e-05  2.316e-04  -0.107   0.9153
## murder      -3.000e-01  3.704e-02  -8.099 2.91e-10 ***
## hs_grad      4.776e-02  1.859e-02   2.569   0.0137 *
## frost       -5.910e-03  2.468e-03  -2.395   0.0210 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7277 on 44 degrees of freedom
## Multiple R-squared:  0.7361, Adjusted R-squared:  0.7061
## F-statistic: 24.55 on 5 and 44 DF,  p-value: 1.019e-11
```

Then we **remove income**

```
step3 <- update(step2, . ~ . -income)
summary(step3)
```

```
##
## Call:
## lm(formula = life_exp ~ population + murder + hs_grad + frost,
##     data = state)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.47095 -0.53464 -0.03701  0.57621  1.50683
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  7.103e+01  9.529e-01  74.542  < 2e-16 ***
## population   5.014e-05  2.512e-05   1.996  0.05201 .
## murder      -3.001e-01  3.661e-02  -8.199 1.77e-10 ***
## hs_grad      4.658e-02  1.483e-02   3.142  0.00297 **
## frost       -5.943e-03  2.421e-03  -2.455  0.01802 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7197 on 45 degrees of freedom
## Multiple R-squared:  0.736,  Adjusted R-squared:  0.7126
## F-statistic: 31.37 on 4 and 45 DF,  p-value: 1.696e-12
```

As we set alpha_to_remove = 0.2 at the beginning. There is no further reduction of variable at the stage.

Result: backward selection model is

life expectancy = 71 + 0.00005population - 0.3Murder + 0.047hs_grad - 0.006frost

2) **Method II: Forward elimination (choose alpha to enter < 0.2)**

We begin with regression with ech single predictor and obtain their summaries

```
fit_pop <- lm(life_exp ~ population, data = state)
result <- tibble(model = map(state[-4], ~lm(life_exp ~ .x, data = state))) %>%
  mutate(result = map(model, broom::tidy)) %>%
  select(-model) %>%
  unnest() %>%
  filter(term == ".x") %>%
  select(-statistic) %>%
  mutate(term = c("population", "income", "illiteracy", "murder", "hs_grad", "frost", "area"),
         estimate = round(estimate, digits = 6),
         std.error = round(std.error, digits = 6))
result %>% arrange(p.value) # rank by p value
```

```
## # A tibble: 7 x 4
##   term        estimate std.error  p.value
##   <chr>          <dbl>     <dbl>    <dbl>
## 1 murder     -0.284     0.0328   2.26e-11
## 2 illiteracy -1.30      0.257    6.97e- 6
## 3 hs_grad     0.0968    0.0195   9.20e- 6
## 4 income      0.000743  0.000297 1.56e- 2
## 5 frost       0.00677   0.00360  6.60e- 2
## 6 area       -0.000002  0.000002 4.58e- 1
## 7 population -0.00002   0.000043 6.39e- 1
```

Enter variable with smallest p value: murder

```
library(broom)
```

```
##
## Attaching package: 'broom'

## The following object is masked from 'package:modelr':
##
##     bootstrap
```

```
forward1 <- lm(life_exp ~ murder, data = state)
```

Enter variable with the smallest p value among the rest:

```
fit1 <- update(forward1, . ~ . +population)
fit2 <- update(forward1, . ~ . +income)
fit3 <- update(forward1, . ~ . +illiteracy)
fit4 <- update(forward1, . ~ . +hs_grad)
fit5 <- update(forward1, . ~ . +frost)
fit6 <- update(forward1, . ~ . +area)

result2 <- tibble(model = map(list(fit1, fit2, fit3, fit4, fit5, fit6), summary)) %>%
  mutate(result = map(model, tidy)) %>%
  select(-model) %>%
  unnest(result)

result2 %>%
  filter(!term %in% c("(Intercept)", "murder")) %>%
  mutate(rank_p_value = rank(p.value)) %>%
  right_join(., result2)
```

```
## Joining, by = c("term", "estimate", "std.error", "statistic", "p.value")
```

```
## # A tibble: 18 x 6
##    term            estimate  std.error statistic  p.value rank_p_value
##    <chr>              <dbl>      <dbl>     <dbl>    <dbl>        <dbl>
##  1 (Intercept)  72.9        0.258         282.    1.55e-77           NA
##  2 murder       -0.312      0.0332         -9.42  2.15e-12           NA
##  3 population    0.0000683  0.0000274       2.49  1.64e- 2            2
##  4 (Intercept)  71.2        0.967          73.6   3.32e-50           NA
##  5 murder       -0.270      0.0328         -8.21  1.22e-10           NA
##  6 income        0.000370   0.000197        1.88  6.66e- 2            4
##  7 (Intercept)  73.0        0.286         256.    1.56e-75           NA
##  8 murder       -0.264      0.0464         -5.69  7.96e- 7           NA
##  9 illiteracy   -0.172      0.281          -0.613 5.43e- 1            6
## 10 (Intercept)  70.3        1.02           69.2   5.91e-49           NA
## 11 murder       -0.237      0.0353         -6.72  2.18e- 8           NA
## 12 hs_grad       0.0439     0.0161          2.72  9.09e- 3            1
## 13 (Intercept)  73.9        0.500         148.    2.36e-64           NA
## 14 murder       -0.328      0.0375         -8.74  2.05e-11           NA
## 15 frost        -0.00578    0.00266        -2.17  3.52e- 2            3
## 16 (Intercept)  72.9        0.275         265.    2.73e-76           NA
## 17 murder       -0.290      0.0338         -8.58  3.47e-11           NA
## 18 area          0.00000118 0.00000146      0.806 4.24e- 1            5
```

Enter variable: hs_grad

```r
forward2 <- lm(life_exp ~ murder + hs_grad, data = state)
tidy(forward2)
```

```
## # A tibble: 3 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)  70.3       1.02       69.2  5.91e-49
## 2 murder       -0.237     0.0353     -6.72 2.18e- 8
## 3 hs_grad       0.0439    0.0161      2.72 9.09e- 3
```

Enter variable with the smallest p value among the rest:

```r
fit1 <- update(forward2, . ~ . +population)
fit2 <- update(forward2, . ~ . +income)
fit3 <- update(forward2, . ~ . +illiteracy)
fit4 <- update(forward2, . ~ . +frost)
fit5 <- update(forward2, . ~ . +area)

result3 <- tibble(model = map(list(fit1, fit2, fit3, fit4, fit5), summary)) %>%
  mutate(result = map(model, tidy)) %>%
  select(-model) %>%
  unnest(result)

result3 %>%
  filter(!term %in% c("(Intercept)", "murder", "hs_grad")) %>%
  mutate(rank_p_value = rank(p.value)) %>%
  right_join(., result3)
```

```
## Joining, by = c("term", "estimate", "std.error", "statistic", "p.value")
```

```
## # A tibble: 20 x 6
##    term            estimate  std.error statistic  p.value rank_p_value
```

```
##   <chr>              <dbl>      <dbl>     <dbl>    <dbl>         <dbl>
##  1 (Intercept)  70.4         0.969         72.7    3.95e-49          NA
##  2 murder       -0.266       0.0357        -7.45   1.91e- 9          NA
##  3 hs_grad       0.0407      0.0154         2.64   1.12e- 2          NA
##  4 population    0.0000625   0.0000259      2.41   1.99e- 2           2
##  5 (Intercept)  70.1         1.10          64.0    1.33e-46          NA
##  6 murder       -0.239       0.0358        -6.66   2.92e- 8          NA
##  7 hs_grad       0.0391      0.0203         1.92   6.05e- 2          NA
##  8 income        0.0000953   0.000239       0.398  6.92e- 1           5
##  9 (Intercept)  69.7         1.22          57.1    2.41e-44          NA
## 10 murder       -0.258       0.0435        -5.93   3.63e- 7          NA
## 11 hs_grad       0.0518      0.0188         2.76   8.25e- 3          NA
## 12 illiteracy    0.254       0.305          0.833  4.09e- 1           3
## 13 (Intercept)  71.0         0.983         72.2    5.25e-49          NA
## 14 murder       -0.283       0.0367        -7.71   8.04e-10          NA
## 15 hs_grad       0.0499      0.0152         3.29   1.95e- 3          NA
## 16 frost        -0.00691     0.00245       -2.82   6.99e- 3           1
## 17 (Intercept)  69.9         1.16          60.1    2.30e-45          NA
## 18 murder       -0.224       0.0404        -5.56   1.30e- 6          NA
## 19 hs_grad       0.0504      0.0190         2.65   1.10e- 2          NA
## 20 area         -0.00000106 0.00000162    -0.658  5.14e- 1           4
```

Enter: frost

```
forward3 <- lm(life_exp ~ murder + hs_grad + frost, data = state)
summary(forward3)
```

```
##
## Call:
## lm(formula = life_exp ~ murder + hs_grad + frost, data = state)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.5015 -0.5391  0.1014  0.5921  1.2268
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 71.036379   0.983262  72.246  < 2e-16 ***
## murder      -0.283065   0.036731  -7.706 8.04e-10 ***
## hs_grad      0.049949   0.015201   3.286  0.00195 **
## frost       -0.006912   0.002447  -2.824  0.00699 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7427 on 46 degrees of freedom
## Multiple R-squared:  0.7127, Adjusted R-squared:  0.6939
## F-statistic: 38.03 on 3 and 46 DF,  p-value: 1.634e-12
```

Enter variable with the smallest p value among the rest:

```
fit1 <- update(forward3, . ~ . +population)
fit2 <- update(forward3, . ~ . +income)
fit3 <- update(forward3, . ~ . +illiteracy)
fit4 <- update(forward3, . ~ . +area)

result4 <- tibble(model = map(list(fit1, fit2, fit3, fit4), summary)) %>%
  mutate(result = map(model, tidy)) %>%
  select(-model) %>%
  unnest(result)
```

```r
result4 %>%
  filter(!term %in% c("(Intercept)", "murder", "hs_grad", "frost")) %>%
  mutate(rank_p_value = rank(p.value)) %>%
  right_join(., result4)
```

```
## Joining, by = c("term", "estimate", "std.error", "statistic", "p.value")

## # A tibble: 20 x 6
##    term             estimate  std.error statistic  p.value rank_p_value
##    <chr>               <dbl>      <dbl>     <dbl>    <dbl>        <dbl>
##  1 (Intercept)  71.0          0.953         74.5   8.61e-49           NA
##  2 murder       -0.300        0.0366        -8.20  1.77e-10           NA
##  3 hs_grad       0.0466       0.0148         3.14  2.97e- 3           NA
##  4 frost        -0.00594      0.00242       -2.46  1.80e- 2           NA
##  5 population    0.0000501    0.0000251      2.00  5.20e- 2            1
##  6 (Intercept)  70.8          1.05          67.4   7.53e-47           NA
##  7 murder       -0.286        0.0373        -7.66  1.07e- 9           NA
##  8 hs_grad       0.0436       0.0190         2.30  2.64e- 2           NA
##  9 frost        -0.00698      0.00247       -2.83  6.96e- 3           NA
## 10 income        0.000127     0.000223       0.571 5.71e- 1            2
## 11 (Intercept)  71.5          1.32          54.2   1.28e-42           NA
## 12 murder       -0.273        0.0411        -6.64  3.50e- 8           NA
## 13 hs_grad       0.0450       0.0178         2.53  1.49e- 2           NA
## 14 frost        -0.00768      0.00283       -2.72  9.36e- 3           NA
## 15 illiteracy   -0.182        0.328         -0.554 5.82e- 1            3
## 16 (Intercept)  70.9          1.15          61.7   3.92e-45           NA
## 17 murder       -0.279        0.0427        -6.52  5.34e- 8           NA
## 18 hs_grad       0.0519       0.0179         2.91  5.66e- 3           NA
## 19 frost        -0.00682      0.00251       -2.71  9.40e- 3           NA
## 20 area         -0.000000329 0.00000154    -0.214 8.32e- 1            4
```

Add population

```r
forward4 <- lm(life_exp ~ murder + hs_grad + frost + population, data = state)
summary(forward4)
```

```
##
## Call:
## lm(formula = life_exp ~ murder + hs_grad + frost + population,
##     data = state)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.47095 -0.53464 -0.03701  0.57621  1.50683
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  7.103e+01  9.529e-01  74.542  < 2e-16 ***
## murder      -3.001e-01  3.661e-02  -8.199 1.77e-10 ***
## hs_grad      4.658e-02  1.483e-02   3.142  0.00297 **
## frost       -5.943e-03  2.421e-03  -2.455  0.01802 *
## population   5.014e-05  2.512e-05   1.996  0.05201 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7197 on 45 degrees of freedom
## Multiple R-squared:  0.736,  Adjusted R-squared:  0.7126
```

```
## F-statistic: 31.37 on 4 and 45 DF,  p-value: 1.696e-12
```

Enter variable with the smallest p value among the rest:

```
fit1 <- update(forward4, . ~ . +income)
fit2 <- update(forward4, . ~ . +illiteracy)
fit3 <- update(forward4, . ~ . +area)

result5 <- tibble(model = map(list(fit1, fit2, fit3), summary)) %>%
  mutate(result = map(model, tidy)) %>%
  select(-model) %>%
  unnest(result)

result5 %>%
  filter(!term %in% c("(Intercept)", "murder", "hs_grad", "frost", "population")) %>%
  mutate(rank_p_value = rank(p.value)) %>%
  right_join(., result5)
```

```
## Joining, by = c("term", "estimate", "std.error", "statistic", "p.value")
```

```
## # A tibble: 18 x 6
##    term        estimate  std.error statistic  p.value rank_p_value
##    <chr>          <dbl>      <dbl>     <dbl>    <dbl>        <dbl>
##  1 (Intercept)  7.11e+1 1.03          69.1    1.66e-46          NA
##  2 murder      -3.00e-1 0.0370        -8.10   2.91e-10          NA
##  3 hs_grad      4.78e-2 0.0186         2.57   1.37e- 2          NA
##  4 frost       -5.91e-3 0.00247       -2.39   2.10e- 2          NA
##  5 population   5.11e-5 0.0000271      1.89   6.57e- 2          NA
##  6 income      -2.48e-5 0.000232      -0.107  9.15e- 1           1
##  7 (Intercept)  7.09e+1 1.32          53.8    8.77e-42          NA
##  8 murder      -3.02e-1 0.0428        -7.05   9.57e- 9          NA
##  9 hs_grad      4.73e-2 0.0173         2.73   9.00e- 3          NA
## 10 frost       -5.81e-3 0.00292       -1.99   5.32e- 2          NA
## 11 population   5.09e-5 0.0000269      1.89   6.51e- 2          NA
## 12 illiteracy   2.91e-2 0.338          0.0861 9.32e- 1           2
## 13 (Intercept)  7.10e+1 1.12          63.7    5.81e-45          NA
## 14 murder      -2.99e-1 0.0428        -7.00   1.16e- 8          NA
## 15 hs_grad      4.69e-2 0.0175         2.68   1.03e- 2          NA
## 16 frost       -5.93e-3 0.00248       -2.39   2.11e- 2          NA
## 17 population   5.00e-5 0.0000255      1.96   5.61e- 2          NA
## 18 area        -5.79e-8 0.00000150    -0.0386 9.69e- 1           3
```

There is no additional predictor with $p < 0.2$, so we will not enter any other predictor. Hence, the forward selection model:

life_exp ~ 71 - 0.3murder + 0.047hs_grad - 0.006frost + 0.00005population

Method III: stepwise regression

```
mult.fit <- lm(life_exp ~ ., data = state)
step(mult.fit, direction = 'both') # select by AIC
```

```
## Start:  AIC=-22.18
## life_exp ~ population + income + illiteracy + murder + hs_grad +
##     frost + area
##
##              Df Sum of Sq    RSS     AIC
## - area        1    0.0011 23.298 -24.182
## - income      1    0.0044 23.302 -24.175
## - illiteracy  1    0.0047 23.302 -24.174
```

```
## <none>                    23.297 -22.185
## - population  1    1.7472 25.044 -20.569
## - frost       1    1.8466 25.144 -20.371
## - hs_grad     1    2.4413 25.738 -19.202
## - murder      1   23.1411 46.438  10.305
##
## Step:  AIC=-24.18
## life_exp ~ population + income + illiteracy + murder + hs_grad +
##     frost
##
##              Df Sum of Sq    RSS     AIC
## - illiteracy  1    0.0038 23.302 -26.174
## - income      1    0.0059 23.304 -26.170
## <none>                    23.298 -24.182
## - population  1    1.7599 25.058 -22.541
## + area        1    0.0011 23.297 -22.185
## - frost       1    2.0488 25.347 -21.968
## - hs_grad     1    2.9804 26.279 -20.163
## - murder      1   26.2721 49.570  11.569
##
## Step:  AIC=-26.17
## life_exp ~ population + income + murder + hs_grad + frost
##
##              Df Sum of Sq    RSS     AIC
## - income      1     0.006 23.308 -28.161
## <none>                    23.302 -26.174
## - population  1     1.887 25.189 -24.280
## + illiteracy  1     0.004 23.298 -24.182
## + area        1     0.000 23.302 -24.174
## - frost       1     3.037 26.339 -22.048
## - hs_grad     1     3.495 26.797 -21.187
## - murder      1    34.739 58.041  17.456
##
## Step:  AIC=-28.16
## life_exp ~ population + murder + hs_grad + frost
##
##              Df Sum of Sq    RSS     AIC
## <none>                    23.308 -28.161
## + income      1     0.006 23.302 -26.174
## + illiteracy  1     0.004 23.304 -26.170
## + area        1     0.001 23.307 -26.163
## - population  1     2.064 25.372 -25.920
## - frost       1     3.122 26.430 -23.877
## - hs_grad     1     5.112 28.420 -20.246
## - murder      1    34.816 58.124  15.528
```

```
##
## Call:
## lm(formula = life_exp ~ population + murder + hs_grad + frost,
##     data = state)
##
## Coefficients:
## (Intercept)   population       murder      hs_grad        frost
##   7.103e+01    5.014e-05   -3.001e-01    4.658e-02   -5.943e-03
```

We choose the one with smallest AIC, hence the model selected by stepwise regression procedure is:

life_exp = 71 + 0.00005population - 0.3murder + 0.047hs_grad - 0.006frost

Answer questions:

a) All the three procedures end up with the same model: life_exp ~ population + murder + hs_grad + frost.

b) During the forward and backward elimination procedures, the variable population is close to the not rejection region in terms of p value if we choose alpha to be 0.05. However, at this stage of exploratory analysis, we want to leverage the critical alpha value to be more inclusive and less stringent in variable selection. Therefore we keep this variable "population" in the model.

c) illteracy vs. HS graduation rate

```
cor(state$illiteracy, state$hs_grad)
```

```
## [1] -0.6571886
```

The linear correlation between illeteracy and HS graduation rate is -0.66. This makes sense because lower high graduation rate can be associated with higher rate of illiteracy. The subsets in the above do not contain both variable.

Problem 3 Criterion based procedure

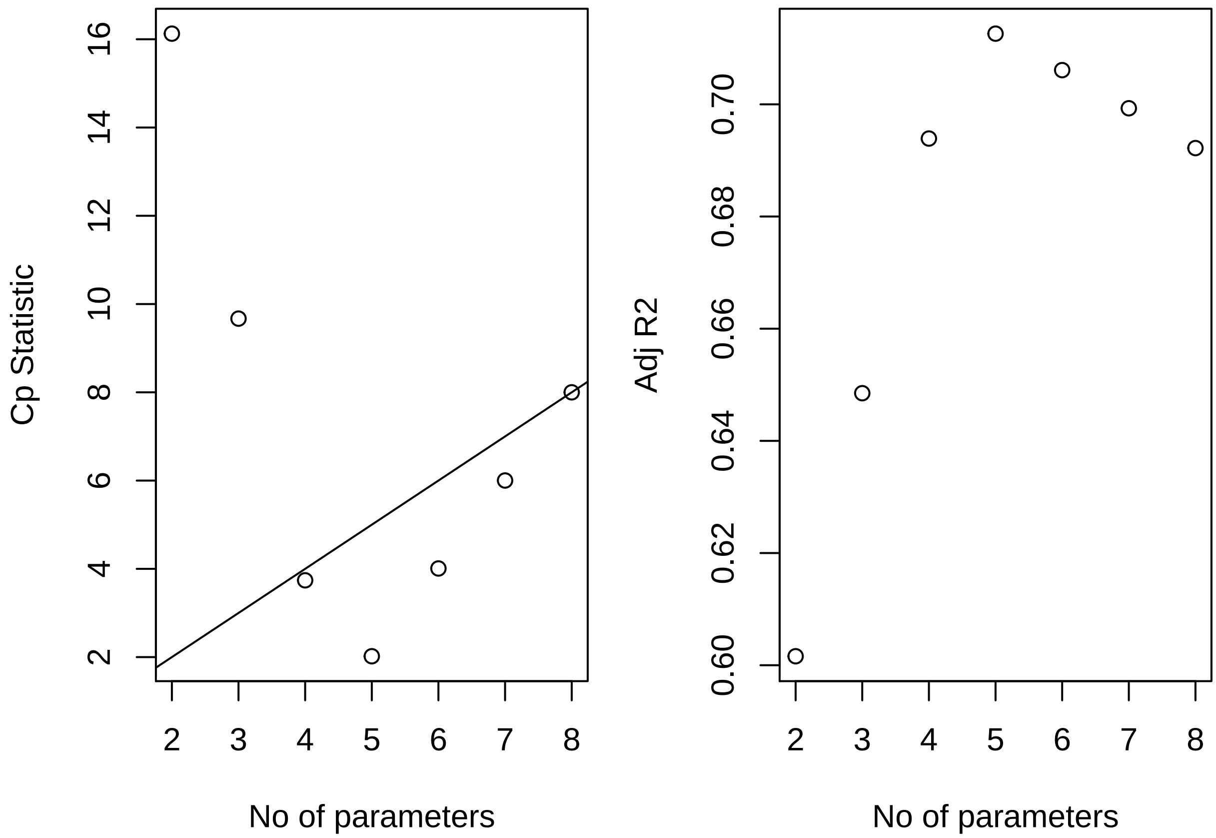
We used criterion of Cp and adjusted R square to select for the best model

```
library(leaps)
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_result <- round(best(multi.fit), 4) %>% as.tibble()
best_result %>% knitr::kable()
```

p	(Intercept)	population	income	illiteracy	murder	hs_grad	frost	area	rss	rsq	adjr2	cp	
1	1	0	0	0	1	0	0	0	34.4613	0.6097	0.6016	16.1268	-3
2	1	0	0	0	1	1	0	0	29.7704	0.6628	0.6485	9.6699	-4
3	1	0	0	0	1	1	1	0	25.3716	0.7127	0.6939	3.7399	-4
4	1	1	0	0	1	1	1	0	23.3080	0.7360	0.7126	2.0197	-4
5	1	1	1	0	1	1	1	0	23.3020	0.7361	0.7061	4.0087	-4
6	1	1	1	1	1	1	1	0	23.2982	0.7361	0.6993	6.0020	-3
7	1	1	1	1	1	1	1	1	23.2971	0.7362	0.6922	8.0000	-3

```
par(mar=c(4,4,1,1))
par(mfrow=c(1,2))


plot(2:8, best_result$cp, xlab="No of parameters", ylab="Cp Statistic")
abline(0,1)

plot(2:8, best_result$adjr2, xlab="No of parameters", ylab="Adj R2")
```



Comment: From the criterion of Cp and Adjusted R square, 5 parameters reach to the summit of adjusted R square with Cp smaller than number of parameters. So we decide to choose the model with 5 parameter (4 predictors): life_exp ~ population + murder + hs_grad + frost. The model we achieved here is consistent with the automatic procedure result above.

Problem 4 choose final model and checking assumption

Given the automatic procedure and criterion based procedure arrive at the same model, we will recommend this consistent result as our final model with 4 predictors: life_exp ~ population + murder + hs_grad + frost

```
multi.fit4 <- lm(life_exp ~ population + murder + hs_grad + frost, data = state)
summary(multi.fit4)
```

```
##
## Call:
## lm(formula = life_exp ~ population + murder + hs_grad + frost,
##     data = state)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.47095 -0.53464 -0.03701  0.57621  1.50683
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  7.103e+01  9.529e-01  74.542  < 2e-16 ***
## population   5.014e-05  2.512e-05   1.996  0.05201 .
## murder      -3.001e-01  3.661e-02  -8.199 1.77e-10 ***
## hs_grad      4.658e-02  1.483e-02   3.142  0.00297 **
## frost       -5.943e-03  2.421e-03  -2.455  0.01802 *
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7197 on 45 degrees of freedom
## Multiple R-squared:  0.736,  Adjusted R-squared:  0.7126
## F-statistic: 31.37 on 4 and 45 DF,  p-value: 1.696e-12
```

a) Identify leverage and/or influential points

1. check outliers in outcome (life_exp)

```
stu_res <- rstandard(multi.fit4) # calculate studentized residuals
outliers_y <- stu_res[abs(stu_res)>2.5]
outliers_y
```

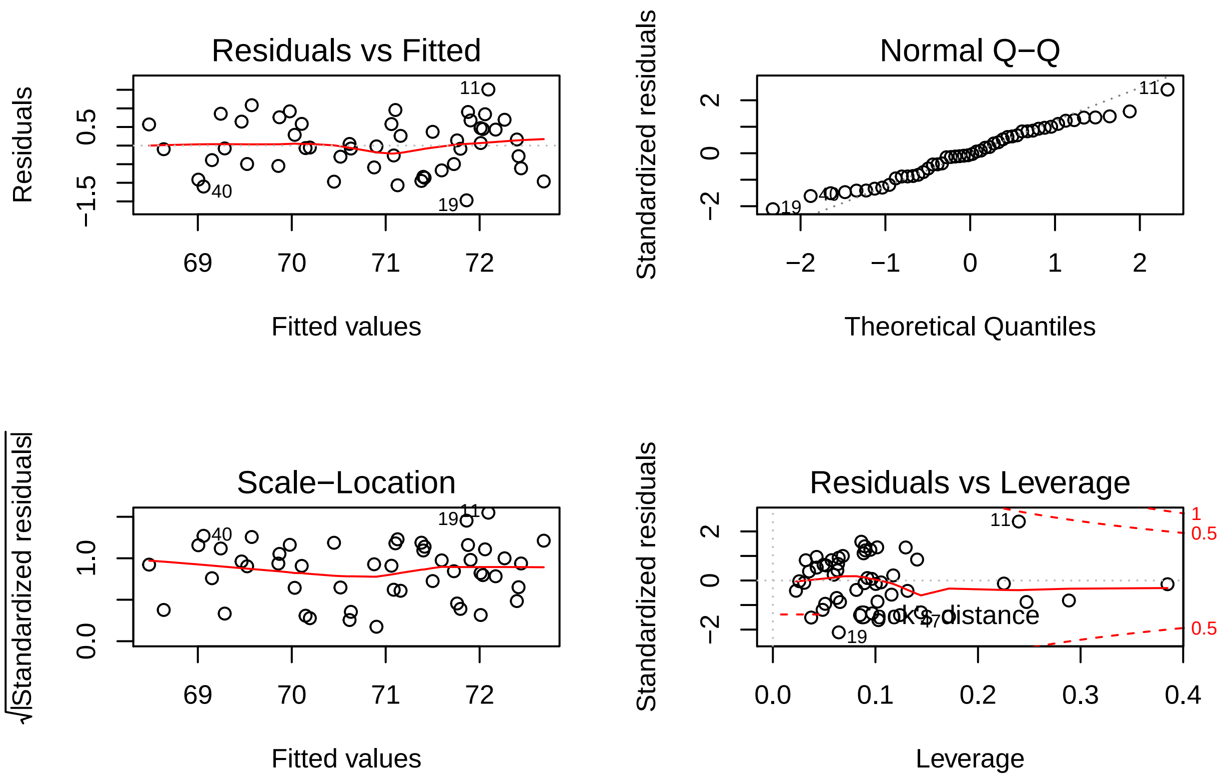
```
## named numeric(0)
```

Comment: we did not find any outlier in life expectancy (response)

2. check leverage and infulential points

Some influential points can be identified on diagnostic plot:

```
par(mfrow = c(2,2))
plot(multi.fit4)
```



Numerical measure of influential points:

```
influ.point <- influence.measures(multi.fit4)
summary(influ.point) %>% knitr::kable()
```

```
## Potentially influential observations of
##   lm(formula = life_exp ~ population + murder + hs_grad + frost,      data = state) :
##
##    dfb.1_ dfb.pplt dfb.mrdr dfb.hs_g dfb.frst dffit   cov.r   cook.d
## 2   0.41   0.18    -0.40    -0.35    -0.16    -0.50    1.36_*  0.05
## 5   0.04  -0.09     0.00    -0.04     0.03    -0.12    1.81_*  0.00
```

```
## 11 -0.03  -0.57     -0.28      0.66     -1.24_*    1.43_*  0.74     0.36
## 28  0.40    0.14    -0.42     -0.29     -0.28     -0.52    1.46_*  0.05
## 32  0.01  -0.06      0.00      0.00     -0.01     -0.07    1.44_*  0.00
##    hat
## 2   0.25
## 5   0.38_*
## 11  0.24
## 28  0.29
## 32  0.23
```

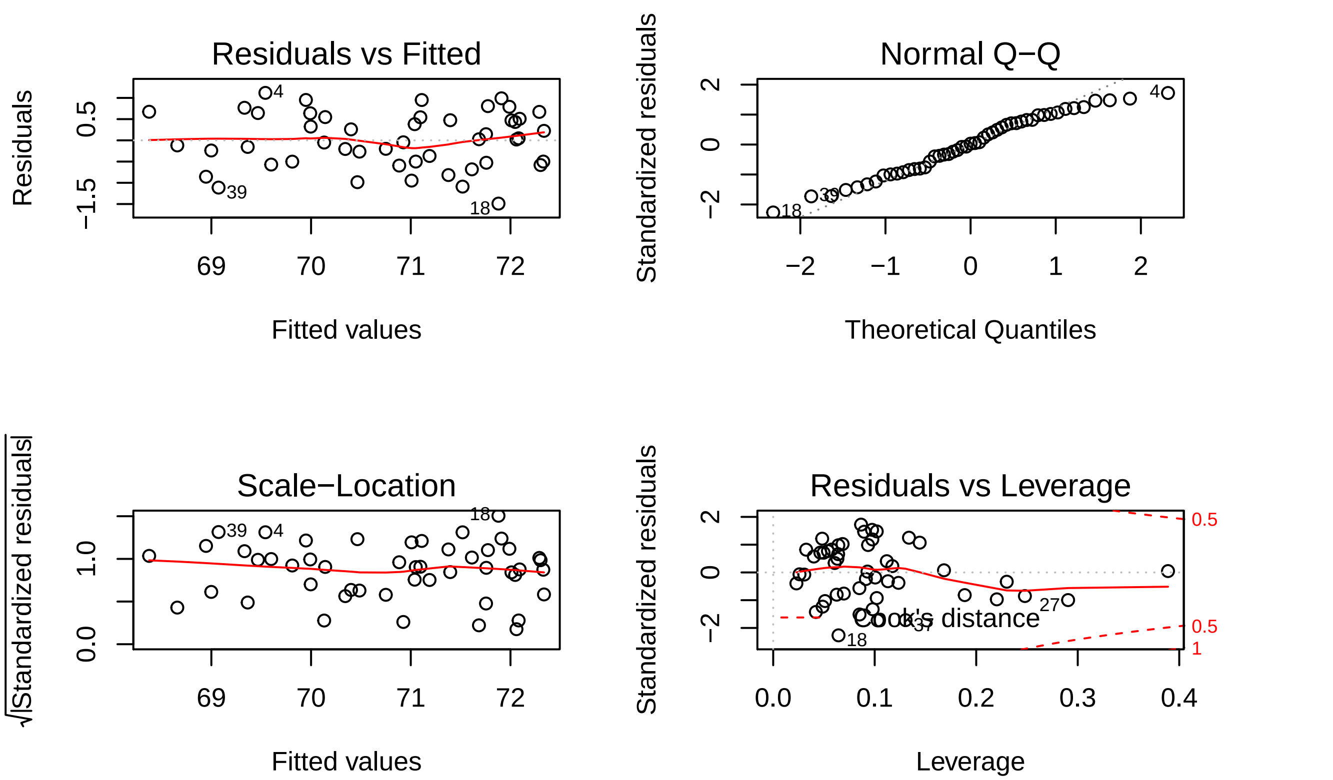
	dfb.1_	dfb.pplt	dfb.mrdr	dfb.hs_g	dfb.frst	dffit	cov.r	cook.d	hat
2	0.4103335	0.1833463	-0.4019774	-0.3492225	-0.1640490	-0.5011665	1.3638202	0.0504978	0.2472792
5	0.0355410	-0.0913986	0.0040451	-0.0441775	0.0269879	-0.1186700	1.8140241	0.0028791	0.3847592
11	-0.0330091	-0.5685627	-0.2759245	0.6644435	-1.2440260	1.4282387	0.7414739	0.3637786	0.2397924
28	0.4029918	0.1431347	-0.4243758	-0.2880959	-0.2832758	-0.5172512	1.4601117	0.0539178	0.2886092
32	0.0113929	-0.0600615	-0.0048877	-0.0024875	-0.0126398	-0.0668113	1.4416747	0.0009127	0.2252274

Comment: obseravtion 5 is an influential point in terms of predictor with high leverage value. observation 11 is identified with high DFFITS value so it affects the observation 11 fitted value. On the diagnostic plot, case 11 appears problematic on each plot. Therefore, we remove this point and do analysis again.

b) check model assumption

From previous conclusion, here we remove the observation 11 and compare the residuals plots with previous ones.

```r
state_no_11 <- state[-11,]
multi.fit4.no11 <- lm(life_exp ~ population + murder + hs_grad + frost, data = state_no_11)
par(mfrow = c(2,2))
plot(multi.fit4.no11)
```



Comment: After removing the influential point observation 11, we observed the residuals variances are stabilized and normality is improved as well. So we will continue the following analysis based on the dataset without observation 11.

Problem 5

a) 10 fold cross validation

Final Model: life_exp ~ population + murder + hs_grad + frost

```
data_train <- trainControl(method="cv", number=10)
```

Fit for 4 predictor model

```
model_caret <- train(life_exp ~ population + murder + hs_grad + frost,
                   data = state_no_11,
                   trControl=data_train,
                   method='lm',
                   na.action=na.pass)
model_caret
```

```
## Linear Regression
##
## 49 samples
##  4 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 44, 43, 45, 44, 45, 43, ...
## Resampling results:
##
##   RMSE       Rsquared   MAE
##   0.6904492  0.8057901  0.615514
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

```
sd(model_caret$resample$Rsquared) # training data R2
```

```
## [1] 0.15525
```

Comment: the RMSE is 0.695 over the 10 folds of testing data. R square is 0.8. The R square shows that 80% the variation in life expectancy can be explained by these four predictors.

b) A new bootstrap : residual sampling

i) fit model with full dataset, get predicted value and resididuals

```
model.fit <- lm(life_exp ~ population + murder + hs_grad + frost, data = state_no_11)

data_pred_res <- state_no_11 %>%
  add_predictions(model.fit) %>%
  add_residuals(model.fit)
```

ii) randomly resample the residuals (with replacement), leaving X and fitted value unchanged

```
set.seed(1)
sample_res <- as.tibble(sample(data_pred_res$resid, nrow(data_pred_res), replace = TRUE))
new_data_pred_res <- cbind(data_pred_res, sample_res) %>% rename("resid_sample" = value)
```

iii) add new sampled residuals to fitted value

```
new_data_pred_res <- new_data_pred_res %>% mutate(new_fitted = pred + resid_sample)
```

iv) regress new fitted value ("new" observations) with origianl predictors

```r
new_model_fit <- lm(new_fitted ~ population + murder + hs_grad + frost, data = new_data_pred_res)
anova(new_model_fit)["Residuals","Mean Sq"] # get the MSE
```

```
## [1] 0.3753003
```

Put everything into function and repeat for 10 and 1000 times:

```r
new_bootstrap <- function(model, n) {
  model_output <- vector("list", length = n)
  MSE_output <- vector("list", length = n)

  model.fit <- lm(life_exp ~ population + murder + hs_grad + frost, data = state_no_11)

  data_pred_res <- state_no_11 %>% add_predictions(model.fit) %>% add_residuals(model.fit)

  for (i in 1:n) {

    sample_res <- as.tibble(sample(data_pred_res$resid, nrow(data_pred_res), replace = TRUE))

    new_data_pred_res <- cbind(data_pred_res, sample_res) %>% rename("resid_sample" = value) %>%
      mutate(new_fitted = pred + resid_sample)

    new_model_fit <- lm(new_fitted ~ population + murder + hs_grad + frost, data = new_data_pred_res)

    model_output[[i]] <- new_model_fit
    MSE_output[i] <- anova(new_model_fit)["Residuals","Mean Sq"]

  }
   tibble(model_output,
          MSE_output = MSE_output %>% as.numeric())
}
```

repeat for 10 and 1000 times:

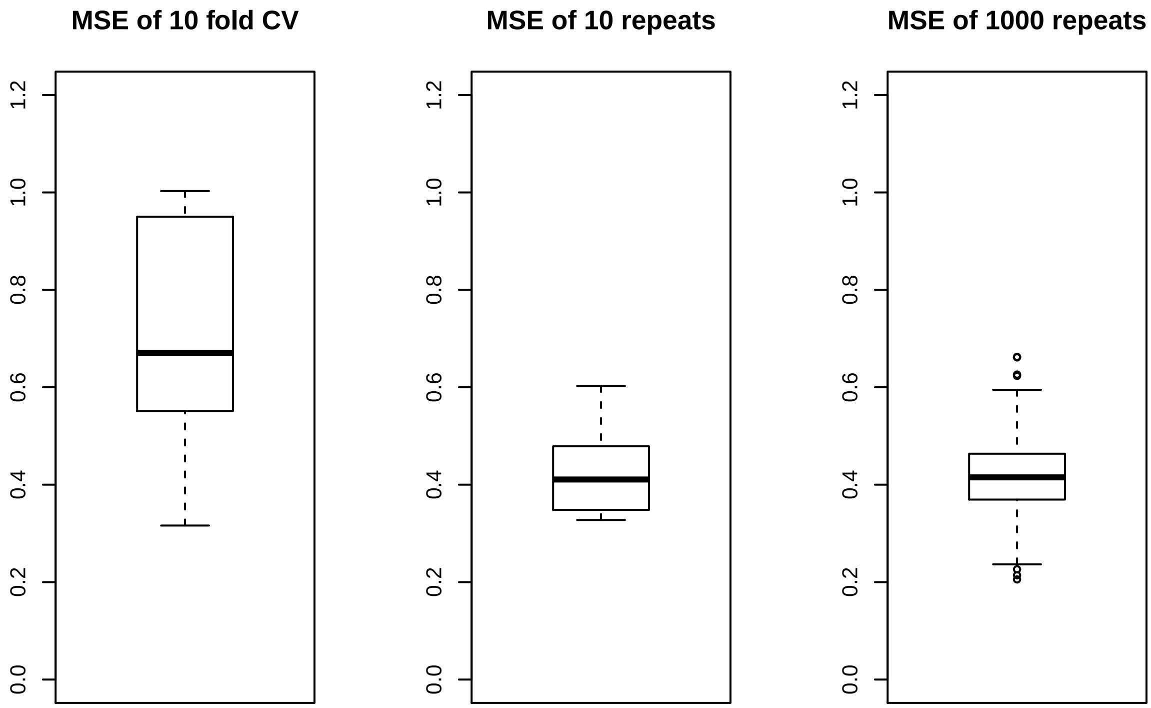
```r
set.seed(2)
newboot_10 <- new_bootstrap(model, 10)
newboot_1000 <- new_bootstrap(model, 1000)

summary(newboot_10$MSE_output)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.3275  0.3537  0.4106  0.4316  0.4743  0.6026
```

```r
summary(newboot_1000$MSE_output)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.2056  0.3696  0.4151  0.4160  0.4636  0.6625
```

```r
# compare previous 10 folds Cross validation method:
par(mfrow = c(1,3))
boxplot(model_caret$resample$RMSE, main = "MSE of 10 fold CV", ylim = c(0, 1.2))
boxplot(newboot_10$MSE_output, main = "MSE of 10 repeats", ylim = c(0, 1.2) )
boxplot(newboot_1000$MSE_output, main = "MSE of 1000 repeats", ylim = c(0, 1.2))
```



Comment: The new bootstrap method achieved a lower prediction MSE with less variance compared to cross validation method. This method relies on resampling residual errors and add to predicted value to create a new set of pseudo "new observations", then refit the model. We tested the predictive ability of the model after generating a new set of "observations" in each cycle. Here we can examine the mean value and variability of MSE. I would recommend the new boostrap method because it does not leave out any data from the full dataset. In addition, it is capable of generating "new" data point for us to test for our model predictibility. So I would say the second method is more reliable.