# ISYE 6501 Homework 05

*EYu*

*February 10, 2019*

## Contents



## Question 8.1

### Describe a situation or problem from your job...

Describe a situation or problem from your job, everyday life, current events, etc., for which a linear regression model would be appropriate. List some (up to 5) predictors that you might use.

We rent out a couple rooms via AirBnB and I think to determine at what level we want to rent them out at, linear regression might be a good solution. Listing data is readily available at Airbnb's website and we could leverage this information, first, to explore what variables seem to play a role in listing price, second, to create an informed decision in deciding listing price. Some variable that would be useful:

- Room type: full house vs private room vs shared room
- Minimum/maximum # nights for rent
- Number of reviews
- Reviews for cleanliness/communication/etc.
- Availability

The linear model would predict price per night based on the above variables. I would probably log transform the price as monetary values tend to follow an exponential pattern.

## Question 8.2

### Using crime data from...

Using crime data from http://www.statsci.org/data/general/uscrime.txt (file uscrime.txt, description at http://www.statsci.org/data/general/uscrime.html ), use regression (a useful R function is lm or glm) to predict the observed crime rate in a city with the following data: M = 14.0
So = 0
Ed = 10.0
Po1 = 12.0
Po2 = 15.5
LF = 0.640
M.F = 94.0

Pop = 150
NW = 1.1
U1 = 0.120
U2 = 3.6
Wealth = 3200
Ineq = 20.1
Prob = 0.04
Time = 39.0

Show your model (factors used and their coefficients), the software output, and the quality of fit.

Note that because there are only 47 data points and 15 predictors, you'll probably notice some overfitting. We'll see ways of dealing with this sort of problem later in the course.

**Description of variables in dataset**

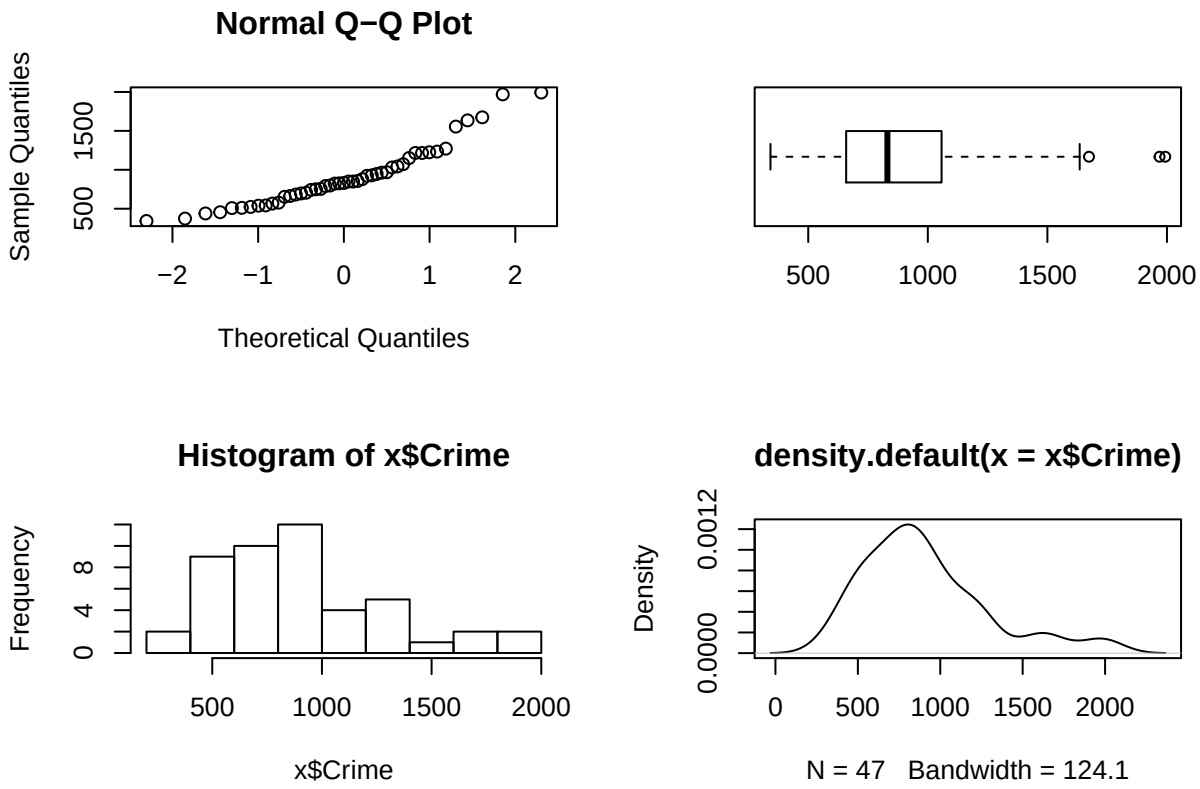
| | Description |
|---|---|
| M | percentage of males aged 14–24 in total state population |
| So | indicator variable for a southern state |
| Ed | mean years of schooling of the population aged 25 years or over |
| Po1 | per capita expenditure on police protection in 1960 |
| Po2 | per capita expenditure on police protection in 1959 |
| LF | labour force participation rate of civilian urban males in the age-group 14-24 |
| M.F | number of males per 100 females |
| Pop | state population in 1960 in hundred thousands |
| NW | percentage of nonwhites in the population |
| U1 | unemployment rate of urban males 14–24 |
| U2 | unemployment rate of urban males 35–39 |
| Wealth | wealth: median value of transferable assets or family income |
| Ineq | income inequality: percentage of families earning below half the median income |
| Prob | probability of imprisonment: ratio of number of commitments to number of offenses |
| Time | average time in months served by offenders in state prisons before their first release |
| Crime | crime rate: number of offenses per 100,000 population in 1960 |

## Import and peak data

We load packages, import, and convert the TRUE/FALSE indicator variable for being in a southern state to a logical. We also perform a quick Shapiro-Wilk test to determine if our data is normally distributed. The Shapiro-Wilk's null hypothesis states that the *data is normally distributed*. The alternative hypothesis that the data *is not* normally distributed.

```
# load packages
if(!require(tidyverse)) install.packages("tidyverse")
library(tidyverse)
if(!require(DAAG)) install.packages("DAAG")
library(DAAG)
if(!require(GGally)) install.packages("GGally")
library(GGally)
if(!require(MASS)) install.packages("MASS")
library(MASS)
if(!require(leaps)) install.packages("leaps")
library(leaps)

setwd("I:/College-Courses-Documents/OMSA-edX/ISYE 6501/hw05")
# setwd("/home/e/Documents/OMSA/ISYE 6501/hw03")
```

```
x <- read_table2("uscrime.txt", col_names=T)
# glimpse(x)

# convert 'So' to logical
x <- x %>%
  mutate(So = as.logical(So))

# change viewport size
par(mfrow=c(2,2))

# plot the plots
qqnorm(x$Crime)
boxplot(x$Crime, horizontal=T)
hist(x$Crime)
plot(density(x$Crime))
```




```
# check Shapiro-Wilk
# reject null hypothesis that data IS normally distributed
shapiro.test(x$Crime)
```

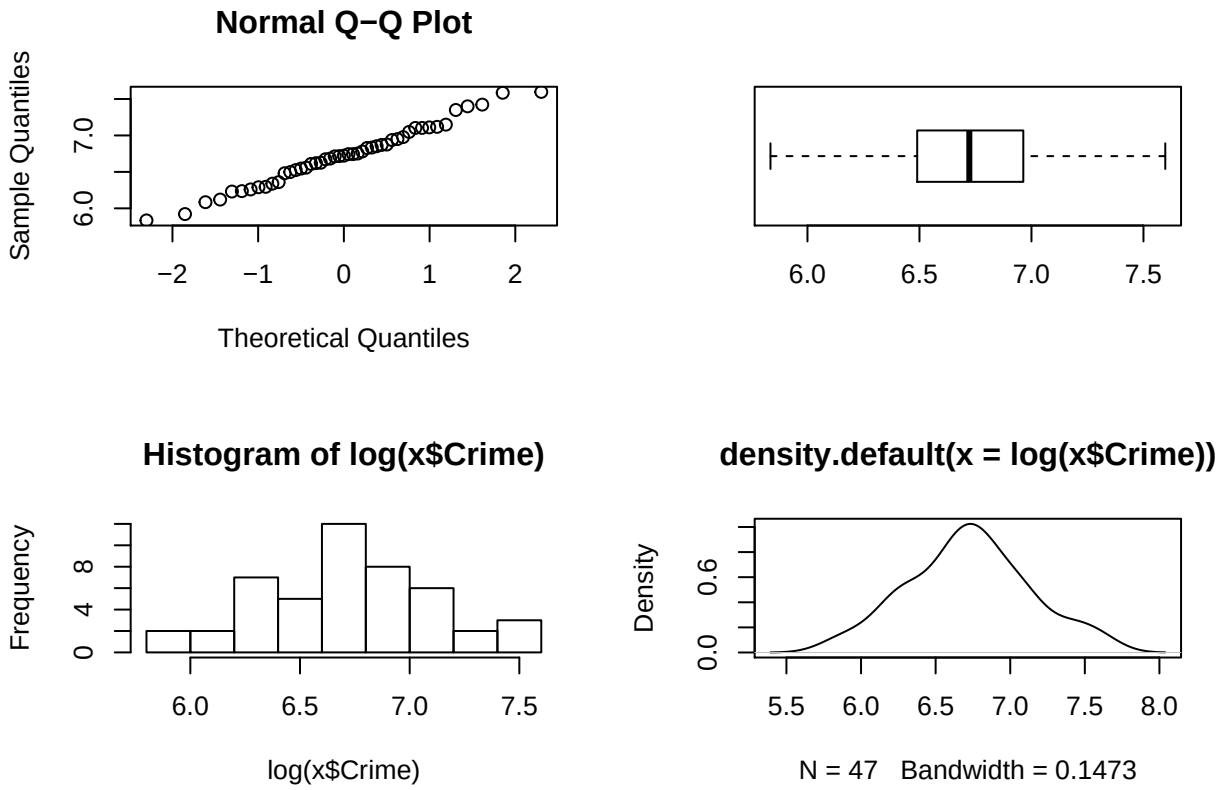
```
##
##  Shapiro-Wilk normality test
##
## data:  x$Crime
## W = 0.91273, p-value = 0.001882
```

Our p-value was low enough that we must reject the null hyopthesis that our data is normally distributed.

Let's try a simple transformation.

### Log transform

```
# change viewport size
par(mfrow=c(2,2))

# plots again
qqnorm(log(x$Crime))
boxplot(log(x$Crime), horizontal=T)
hist(log(x$Crime))
plot(density(log(x$Crime)))
```




```
# check Shapiro-Wilk
# keep alternative hypothesis that data is normally distributed
shapiro.test(log(x$Crime))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  log(x$Crime)
## W = 0.98709, p-value = 0.8778
```

A log transform seems to do the job and now our p-value is high enough that we accept the null hypothesis. We can try to create a linear model now, we'll try it with the log transformed as well as the standard data.

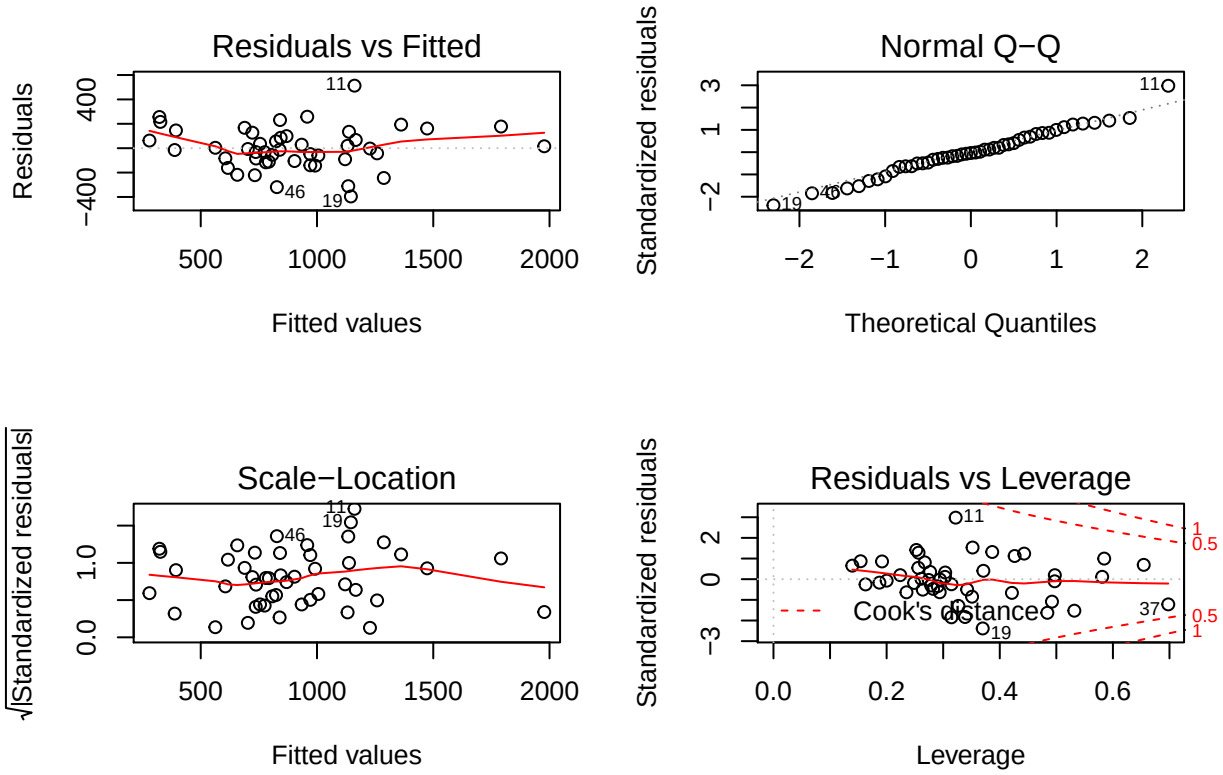
## Create linear models

We create the models, look at the summary stats, and look at the diagnostic plots.

```
# Linear model
crime.lm     <- lm(Crime ~ ., data = x)
summary(crime.lm)
```

```
##
## Call:
## lm(formula = Crime ~ ., data = x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -395.74  -98.09   -6.69  112.99  512.67
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -5.984e+03  1.628e+03  -3.675 0.000893 ***
## M            8.783e+01  4.171e+01   2.106 0.043443 *
## SoTRUE      -3.803e+00  1.488e+02  -0.026 0.979765
## Ed           1.883e+02  6.209e+01   3.033 0.004861 **
## Po1          1.928e+02  1.061e+02   1.817 0.078892 .
## Po2         -1.094e+02  1.175e+02  -0.931 0.358830
## LF          -6.638e+02  1.470e+03  -0.452 0.654654
## M.F          1.741e+01  2.035e+01   0.855 0.398995
## Pop         -7.330e-01  1.290e+00  -0.568 0.573845
## NW           4.204e+00  6.481e+00   0.649 0.521279
## U1          -5.827e+03  4.210e+03  -1.384 0.176238
## U2           1.678e+02  8.234e+01   2.038 0.050161 .
## Wealth       9.617e-02  1.037e-01   0.928 0.360754
## Ineq         7.067e+01  2.272e+01   3.111 0.003983 **
## Prob        -4.855e+03  2.272e+03  -2.137 0.040627 *
## Time        -3.479e+00  7.165e+00  -0.486 0.630708
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 209.1 on 31 degrees of freedom
## Multiple R-squared:  0.8031, Adjusted R-squared:  0.7078
## F-statistic: 8.429 on 15 and 31 DF,  p-value: 3.539e-07
```

```
par(mfrow=c(2,2))
plot(crime.lm)
```

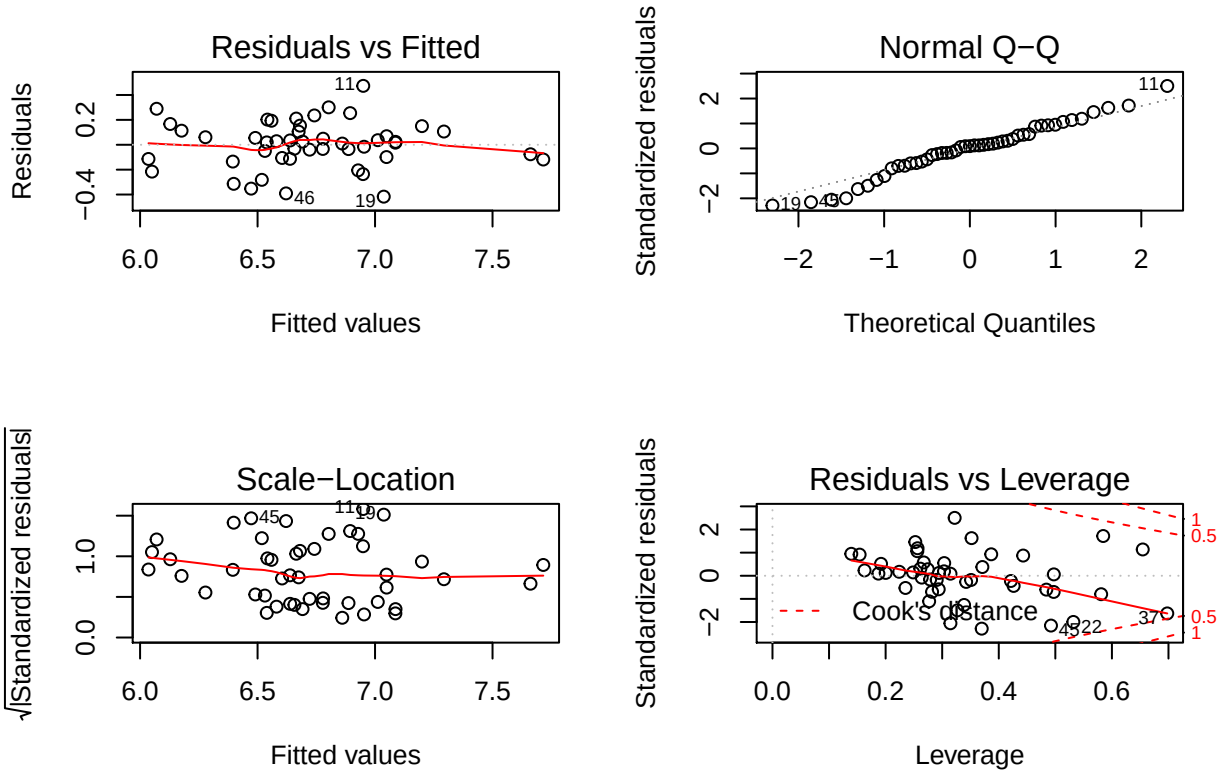



```
# Log-linear model
crime.log.lm <- lm(log(Crime) ~ ., data = x)
summary(crime.log.lm)
```

```
##
## Call:
## lm(formula = log(Crime) ~ ., data = x)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.41697 -0.10961  0.01903  0.10971  0.47322
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.4345908  1.7884057   0.243  0.80960
## M            0.1160700  0.0458149   2.533  0.01657 *
## SoTRUE       0.0917576  0.1633799   0.562  0.57841
## Ed           0.2147289  0.0681926   3.149  0.00361 **
## Po1          0.1862712  0.1165418   1.598  0.12012
## Po2         -0.1077276  0.1290273  -0.835  0.41015
## LF           0.1874034  1.6142245   0.116  0.90833
## M.F         -0.0061943  0.0223549  -0.277  0.78355
## Pop         -0.0009638  0.0014163  -0.681  0.50124
## NW           0.0047976  0.0071181   0.674  0.50530
## U1          -4.3033459  4.6242214  -0.931  0.35925
## U2           0.1717980  0.0904308   1.900  0.06680 .
```

```
## Wealth       0.0001737  0.0001139   1.526  0.13715
## Ineq         0.0808730  0.0249499   3.241  0.00284 **
## Prob        -6.0950555  2.4957820  -2.442  0.02050 *
## Time        -0.0080381  0.0078697  -1.021  0.31497
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2296 on 31 degrees of freedom
## Multiple R-squared:  0.7897, Adjusted R-squared:  0.688
## F-statistic: 7.761 on 15 and 31 DF,  p-value: 8.862e-07
```

```
par(mfrow=c(2,2))
plot(crime.log.lm)
```




It's not obvious looking at the summary output of the models which to choose. The F-statistics are very close. Looking at the graph we do see the logarithmic model looks more normally distributed. Let's get some predictions.

```
# PREDICT ON NEW DATA
new.data <- data.frame("M"= 14,
                       "So" = FALSE,
                       "Ed" = 10,
                       "Po1" = 12,
                       "Po2" = 15.5,
                       "LF" = .640,
                       "M.F" = 94,
                       "Pop" = 150,
```

```
                        "NW" = 1.1,
                        "U1" = .120,
                        "U2" = 3.6,
                        "Wealth" = 3200,
                        "Ineq" = 20.1,
                        "Prob" = .04,
                        "Time" = 39)

predict.lm(crime.lm, newdata = new.data) # 155.4349
```

```
##        1
## 155.4349
```

```
predict.lm(crime.log.lm, newdata = new.data) # 5.897035
```

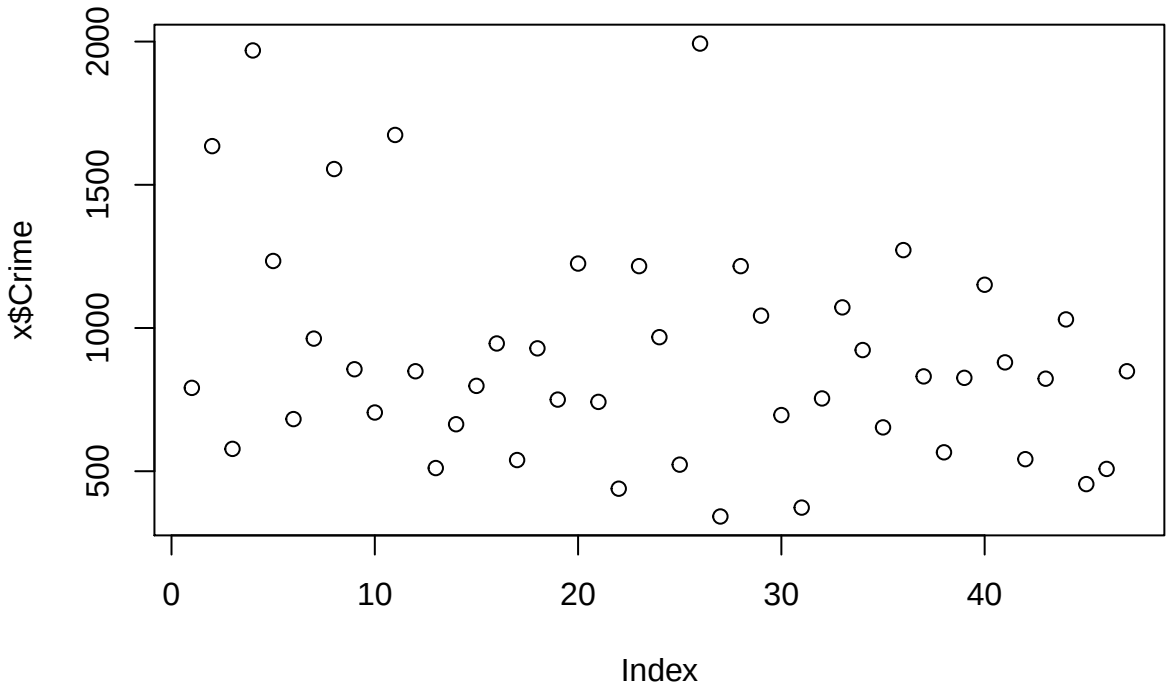
```
##        1
## 5.897035
```

```
exp(5.897035) # 363.9567
```

```
## [1] 363.9567
```

These values do seem quite low–as mentioned in the office hours–when we look at a scatterplot of the actual crime values. The values values range from 500-2000, well above our predicted results for either model! Our model needs tuned a bit.

```
plot(x$Crime)
```




Thus far we see the qqplot and the residuals vs fitted plots do show improvement with the log-linear model.

Interestingly, both the F-statistic and the adjusted R-squared values are higher in the standard, non-log-transformed model! Seems as though transformation of the `x$Crime` variable did not help as anticipated (if we use those measures as a measure of model accuracy, that is).

So we've got one potential linear model that predicts Crime based on *all* variables. We also see from the `summary()` output that some variables are flagged as being more significant (having lower p-values) than others. It seems intuitive to simply select the significant variables and make a new model. However, there are many, many posts on StackExchange that cite this idea as a bad one, here are just a couple:

> If you are putting in a predictor variable with multiple levels, you either put in the variable or you don't, you can't pick and choose levels. You might want to restructure the levels of your predictor variable to decrease the number of levels (if that makes sense in the context of your analysis.) However, I'm not sure if this would cause some type of statistical invalidation if you're collapsing levels because you see they are not significant [, https://stats.stackexchange.com/users/9686/ellie].

> If you are putting in a variable with a number of levels, you need to retain all those levels in your analysis. Picking and choosing based on significance level will both bias your results and do very weird things to your inference, even if by some miracle your estimates manage to stay the same, as you'll have gaping holes in your estimated effects over different levels of the variable [, https://stats.stackexchange.com/users/5836/fomite].

Stepwise selection is yet another method looked down on by, seemingly, the whole StackExchange community, for example:

> I think this approach is mistaken, but perhaps it will be more helpful if I explain why. Wanting to know the best model given some information about a large number of variables is quite understandable. Moreover, it is a situation in which people seem to find themselves regularly. In addition, many textbooks (and courses) on regression cover stepwise selection methods, which implies that they must be legitimate. Unfortunately however, they are not, and the pairing of this situation and goal are quite difficult to successfully navigate. The following is a list of problems with automated stepwise model selection procedures [gung , https://stats.stackexchange.com/users/7290/gung].

Now that I've appropriately disclaimed the methods I'll be using, let's try them out.

But first... let's consider variance inflation factors!

## Variance inflation factors

Essentially, when two variables happen to trend together this can create a false significance in our model. Explained more clearly in this post:

> If x1 and x2 are strongly correlated then that means that they 'move together'. What linear regression tries to do is to ''assign" a change in the dependent variable y to either x1 or x2. Obviously, if both 'move together' (because of high correlation) then it will be difficult to 'decide' which of the x's is 'responsible' for the change in y (because they both change). Therefore the estimates of the $[\beta_i]$ coefficients will be less precise. A VIF of four means that the variance (a measure of imprecision) of the estimated coefficients is four times higher because of correlation between the two independent variables [user83346].

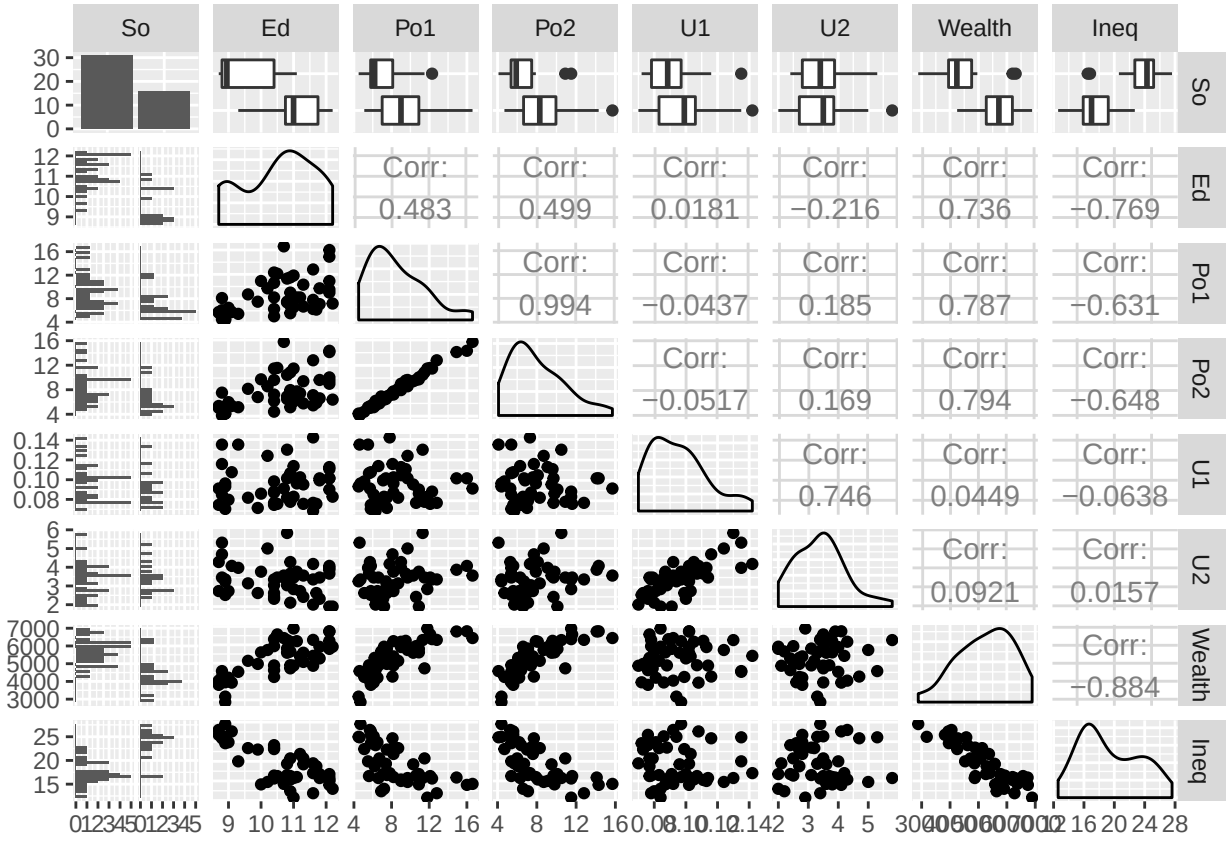
In general, it seems the rule of thumb is to reject at least one of the highly correlated factors from the model to reduce the ill effects. Let's take a look at our vif values and plot them against one another to see how they look.

```
# vif > 5 or 10 problematic
vif(crime.lm)
```

```
##        M   SoTRUE       Ed      Po1      Po2       LF      M.F      Pop
##   2.8924   5.3428   5.0774 104.6600 113.5600   3.7127   3.7859   2.5367
##       NW       U1       U2   Wealth     Ineq     Prob     Time
```

```
##    4.6741    6.0639    5.0889   10.5300    8.6445    2.8095    2.7138
```

```
# correlation plot
x %>%
  dplyr::select(So, Ed, Po1, Po2, U1, U2, Wealth, Ineq) %>%
  ggpairs()
```




There are a few variables that are very correlated with one another, such as: Po1:Po2, Wealth:Ineq, Po2:Wealth. And these correlations are very likely being misrepresented in our model. Because removing one variable will likely change the vif values, what we can do is remove the highest vif and check our model sequentially until there are no vif values above 5.

### Remove highest VIF and recreate model

```
# check max vif, remove
max(vif(crime.lm))
```

```
## [1] 113.56
```

```
new.lm <- lm(Crime ~ . -Po2,x)

max(vif(new.lm))
```

```
## [1] 10.497
```

```
new.lm <- lm(Crime ~ . -Po2,x)

max(vif(new.lm))
```

```
## [1] 10.497
```

```
new.lm <- lm(Crime ~ . -Po2-Wealth,x)

max(vif(new.lm))
```

```
## [1] 5.9483
```

```
new.lm <- lm(Crime ~ . -Po2-Wealth-U1,x)

max(vif(new.lm))
```

```
## [1] 5.4376
```

```
new.lm <- lm(Crime ~ . -Po2-Wealth-U1-Ineq,x)

# now our max vif value is 3.77, acceptable
max(vif(new.lm))
```

```
## [1] 3.7718
```

```
# check new model and make a prediction
crime.vif.lm <- lm(Crime ~ . -Po2-Wealth-U1-Ineq,x)
summary(crime.vif.lm)
```

```
##
## Call:
## lm(formula = Crime ~ . - Po2 - Wealth - U1 - Ineq, data = x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -427.18 -142.11    2.68  109.47  531.16
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -4.874e+03  1.642e+03  -2.968  0.00537 **
## M            9.431e+01  4.697e+01   2.008  0.05242 .
## SoTRUE       2.721e+02  1.417e+02   1.921  0.06293 .
## Ed           8.472e+01  6.114e+01   1.386  0.17459
## Po1          8.709e+01  1.914e+01   4.551 6.18e-05 ***
## LF           1.631e+03  1.406e+03   1.160  0.25394
## M.F          1.755e+01  1.921e+01   0.914  0.36715
## Pop          7.083e-02  1.405e+00   0.050  0.96008
## NW           4.247e+00  6.670e+00   0.637  0.52851
## U2           9.330e+01  5.867e+01   1.590  0.12080
## Prob        -5.023e+03  2.483e+03  -2.023  0.05080 .
## Time        -8.250e-01  7.578e+00  -0.109  0.91393
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 239.5 on 35 degrees of freedom
## Multiple R-squared:  0.7082, Adjusted R-squared:  0.6164
## F-statistic: 7.721 on 11 and 35 DF,  p-value: 1.467e-06
```

```
predict.lm(crime.vif.lm, new.data) # 1149.968
```

```
##        1
## 1149.968
```

So we got a prediction that's actually within the range represented in our response values! Now what? I think we can optimize even more. Now that autocorrelation is mitigated we can try to use our stepwise selection which will attempt to choose the best predictors based on AIC values.

## Model selection

We'll try:

1. Selecting only significant variables
2. Forward stepwise
3. Backward stepwise
4. Regsubsets exhaustive

### Pick the significant variables only

This seemed like the right approach at first. We'll try it anyways to see what happens.

```
sig.model <- lm(Crime ~ Ed + Ineq + M + Prob + U2 + Po1, data = x)
summary(sig.model)
```

```
##
## Call:
## lm(formula = Crime ~ Ed + Ineq + M + Prob + U2 + Po1, data = x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -470.68  -78.41  -19.68  133.12  556.23
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -5040.50     899.84  -5.602 1.72e-06 ***
## Ed            196.47      44.75   4.390 8.07e-05 ***
## Ineq           67.65      13.94   4.855 1.88e-05 ***
## M             105.02      33.30   3.154  0.00305 **
## Prob        -3801.84    1528.10  -2.488  0.01711 *
## U2             89.37      40.91   2.185  0.03483 *
## Po1           115.02      13.75   8.363 2.56e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 200.7 on 40 degrees of freedom
## Multiple R-squared:  0.7659, Adjusted R-squared:  0.7307
## F-statistic: 21.81 on 6 and 40 DF,  p-value: 3.418e-11
```

```
predict.lm(sig.model, new.data) # 1304.245
```

```
##        1
## 1304.245
```

### Forward stepwise

We create an empty linear model, and then build up one at a time, selecting the lowest AIC value variable and adding sequentially until adding an additional variable leads to no improvement in the model

```
# forward stepwise
# create empty model
lowfit <- lm(Crime ~ 1, x)
```

```
stepf <- stepAIC(lowfit, direction = "forward", trace = FALSE,
                 scope = list(upper = crime.vif.lm,
                              lower = lowfit))

summary(stepf)
```

```
##
## Call:
## lm(formula = Crime ~ Po1 + M + M.F + Prob + So, data = x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -457.01 -121.83   15.94  121.44  504.02
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -4354.40    1297.99  -3.355  0.00172 **
## Po1           100.53      14.62   6.878 2.45e-08 ***
## M              74.01      37.79   1.958  0.05702 .
## M.F            35.93      12.85   2.796  0.00784 **
## Prob        -4988.42    1955.15  -2.551  0.01455 *
## SoTRUE        241.04     107.50   2.242  0.03041 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 236.6 on 41 degrees of freedom
## Multiple R-squared:  0.6665, Adjusted R-squared:  0.6258
## F-statistic: 16.39 on 5 and 41 DF,  p-value: 7.203e-09
```

```
predict.lm(stepf, new.data) # 1066.205
```

```
##        1
## 1066.205
```

### Backward stepwise

We start with our full model and remove the highest AIC value variable, one at a time, until removing no longer improves the model.

```
stepb <- stepAIC(crime.vif.lm, direction = "backward", trace = FALSE)

summary(stepb)
```

```
##
## Call:
## lm(formula = Crime ~ M + So + Ed + Po1 + LF + U2 + Prob, data = x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -494.61 -131.42   -3.34  118.38  536.45
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -4155.45    1042.45  -3.986 0.000285 ***
## M             116.81      39.09   2.988 0.004835 **
```

```
## SoTRUE        308.36     120.12   2.567 0.014205 *
## Ed             95.68      51.97   1.841 0.073247 .
## Po1            89.55      15.30   5.851 8.33e-07 ***
## LF           2414.78    1147.91   2.104 0.041911 *
## U2            121.69      49.50   2.458 0.018508 *
## Prob        -4324.24    1904.07  -2.271 0.028743 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 231.1 on 39 degrees of freedom
## Multiple R-squared:  0.6973, Adjusted R-squared:  0.643
## F-statistic: 12.84 on 7 and 39 DF,  p-value: 2.029e-08
```

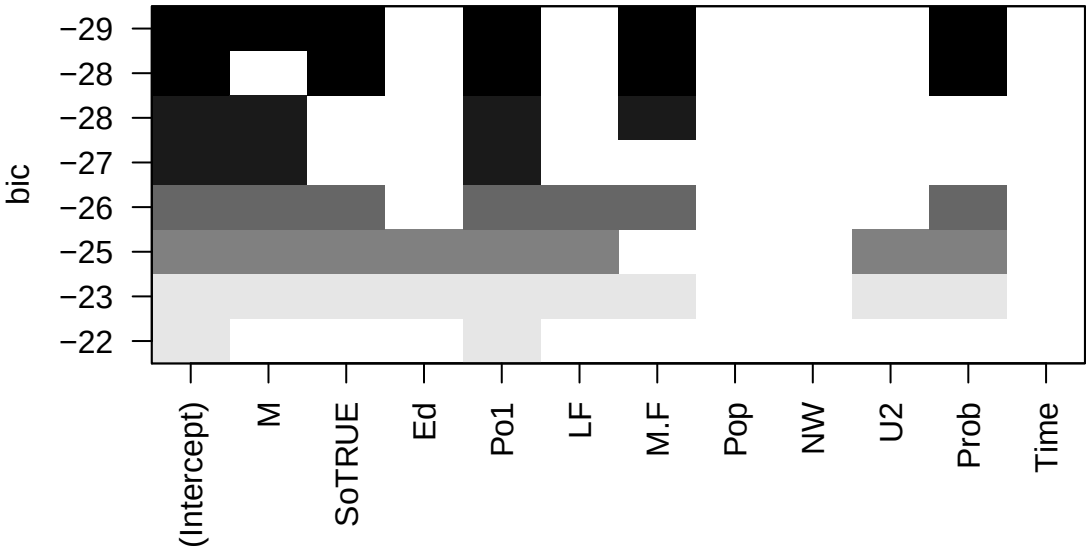
```
predict.lm(stepb, new.data) # 1321.79
```

```
##       1
## 1321.79
```

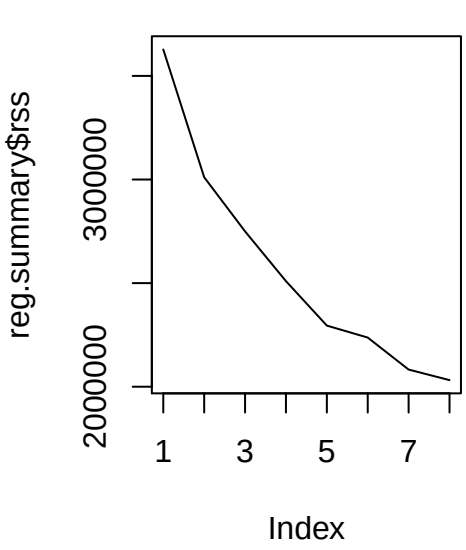
### Regsubsets

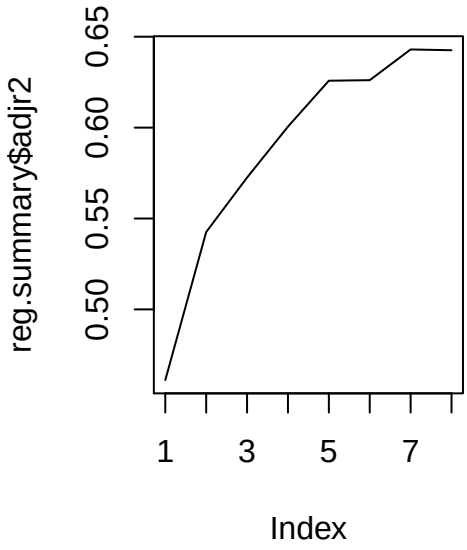
This seems to use a combination of stepwise methods, the method used according to the function output is *exhaustive*. We build a model using the `regsubsets` function, can plot which variables used together create the lowest BIC value, and can also plot number of variables used versus various measures of model performance (i.e., RSS, r2. adjr2, Cp, BIC). When plotted we can make a judgement on how many variables we want to use. In this case I've chosen to use the BIC measure as a guide. Choosing 5 variables grants to lowest BIC, so I recreate the model using the 5 most significant variables, as reported by the function.

```
reg.Crime.vif <- regsubsets(Crime ~. -Po2-Wealth-U1-Ineq, x)
par(mfrow=c(1,1))
plot(reg.Crime.vif)
```

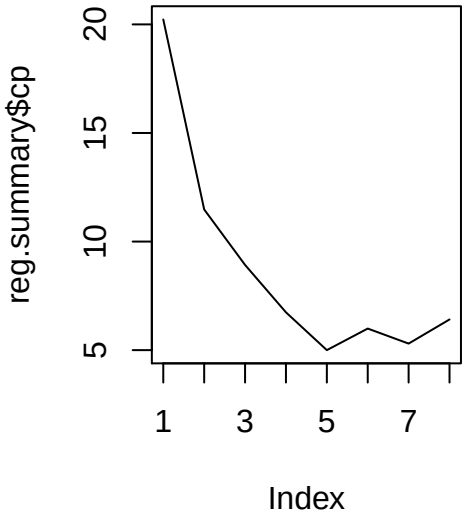


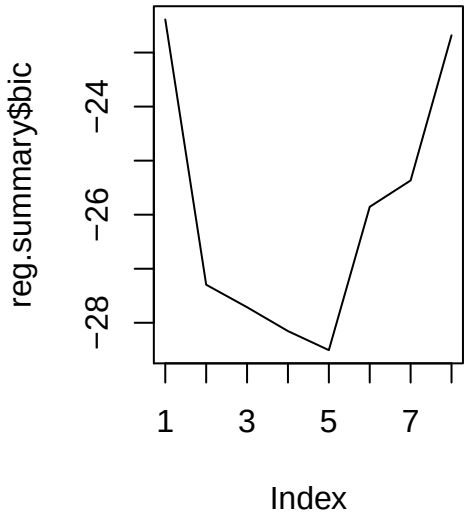

```
# plot(reg.Crime.vif,scale="r2")
reg.summary <- summary(reg.Crime.vif)
# plot
par(mfrow=c(1,2))
plot(reg.summary$rss, type="l") # lower rss more variables
plot(reg.summary$adjr2, type="l")
```




```
plot(reg.summary$cp, type="l")
plot(reg.summary$bic, type="l")
```




```
which.max(reg.summary$adjr2) # tells us highest r2 uses 7 variables
```

```
## [1] 7
```

```
which.min(reg.summary$cp)    # smallest Cp uses 5 variables
```

```
## [1] 5
```

```
which.min(reg.summary$bic)   # smallest bic uses 5 variables
```

```
## [1] 5
```

```
# select the top 5 variables
coef(reg.Crime.vif,5)
```

```
## (Intercept)           M      SoTRUE         Po1         M.F        Prob
##  -4354.4006     74.0109    241.0437    100.5256     35.9328  -4988.4198
```

```
# create a model
reg.Crime.vif.lm <- lm(Crime ~ M + So + Po1 + M.F + Prob, x)
summary(reg.Crime.vif.lm)
```

```
##
## Call:
## lm(formula = Crime ~ M + So + Po1 + M.F + Prob, data = x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -457.01 -121.83   15.94  121.44  504.02
```

```
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -4354.40    1297.99  -3.355  0.00172 **
## M              74.01      37.79   1.958  0.05702 .
## SoTRUE        241.04     107.50   2.242  0.03041 *
## Po1           100.53      14.62   6.878 2.45e-08 ***
## M.F            35.93      12.85   2.796  0.00784 **
## Prob        -4988.42    1955.15  -2.551  0.01455 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 236.6 on 41 degrees of freedom
## Multiple R-squared:  0.6665, Adjusted R-squared:  0.6258
## F-statistic: 16.39 on 5 and 41 DF,  p-value: 7.203e-09
```

```
predict.lm(reg.Crime.vif.lm, new.data) # 1066.205
```

```
##        1
## 1066.205
```

### Regsubsets without vif-removal

We'll try a regsubsets selection on the original linear model as well for comparison sake.

```
# we'll also try regsubsets using the standard non-vif adjusted model
reg.Crime <- regsubsets(Crime ~., x)
reg.summary <- summary(reg.Crime)

which.max(reg.summary$adjr2) # tells us best model is with 8 variables
```

```
## [1] 8
```

```
which.min(reg.summary$cp)     # 6 vars
```

```
## [1] 6
```

```
which.min(reg.summary$bic)    # 6 vars
```

```
## [1] 6
```

```
# select top 6 coeffs
coef(reg.Crime.vif,6)
```

```
## (Intercept)           M      SoTRUE         Po1          LF         M.F
## -4377.22872    70.00154   282.34106   100.84595  1137.12074    30.01780
##        Prob
## -4883.95124
```

```
#create model
reg.Crime.lm <- lm(Crime ~ M + Ed + Po1 + U2 + Ineq + Prob, x)
predict.lm(reg.Crime.lm, new.data) # 1304
```

```
##        1
## 1304.245
```

### Summary of model selections

Okay we've gone thru a lot of models, let's list off a few we want to compare:

1. All predictors
2. Regsubsets variable selection
3. Keep only high p-value predictors
4. High VIF removed
5. High VIF removed, forward stepwise
6. High VIF removed, backward stepwise
7. High VIF removed, regsubsets

To compare models to one another, generally the F-statistic is used. To give an estimate of how well the model fits to the test data we can report the $R^2_{adj}$ as listed in most of the `summary` calls to each function. We can also perform cross validation to produce a more realistic $R^2_{adj}$ value that is produced from fitting to a hold-out set.

We'll create a function to predict $R^2$ using cross-validation of our models.

### R-squared using k-fold cross-validation

```
# function to return rsq and adjrsq, needs linear model + number of folds
calc_r2 <- function(lmod, kfold){
  r <- tibble("r2"    = numeric(0),
              "adjr2" = numeric(0))
  y <- as.data.frame(x)
  y <- cv.lm(y, lmod, m = kfold)

  y.resid <- sum((y$Crime - y$cvpred)^2)
  y.sst   <- sum((y$Crime - mean(y$Crime))^2)

  rsq <- 1-(y.resid/y.sst)

  k <- length(lmod$coefficients)-1
  n <- as.numeric(length(x$Crime))

  adjr2 <- 1-(((1-rsq)*(n-1))/(n-k-1))

  r[1,1] <- rsq
  r[1,2] <- adjr2
  return(r)
}

# commented out because the output takes up several pages
# calc_r2(crime.lm, 4)         # .470, .214
# calc_r2(reg.Crime.lm,4)      # .719, .677
# calc_r2(sig.model,4)         # .719, .677
# calc_r2(crime.vif.lm,4)      # .358, .156
# calc_r2(stepf,4)             # .555, .500
# calc_r2(stepb,4)             # .447, .348
# calc_r2(reg.Crime.vif.lm,4) # .719, .677
```

Sparing you all the output produced from calling the function, a table formatted with the relevant data looks like this:

#### Model, predictors, model $R^2_{adj}$ & F-statistic

```
knitr::kable(summ[1:5])
```

| model.used | predictors.used | adj.r2 | f.statistic | prediction |
|---|---|---|---|---|
| all.pred | M,So,Ed,Po1,Po2,LF,M.F,Pop,NW,U1,U2,Wealth,Ineq,Prob,Time | 0.7078 | 8.43 | 155 |
| regsub.all | M,Ed,Po1,U2,Ineq,Prob | 0.7660 | 21.81 | 1304 |
| high.pvals | M,Ed,Po1,U2,Ineq,Prob | 0.7660 | 21.81 | 1304 |
| vif.adj | M,So,Ed,Po1,LF,M.F,Pop,NW,U2,Prob,Time | 0.6160 | 7.72 | 1150 |
| vif.stepf | M,So,Po1,M.F,Prob | 0.6260 | 16.40 | 1066 |
| vif.stepb | M,So,Ed,Po1,LF,U2,Prob | 0.6430 | 12.80 | 1322 |
| vif.regsub | M,Ed,Po1,U2,Ineq,Prob | 0.7310 | 21.80 | 1304 |

**Model, cross-validated $R^2$ & $R^2_{adj}$**

```
knitr::kable(summ[c(1,6:7)])
```

| model.used | cv.r2 | cv.adjr2 |
|---|---|---|
| all.pred | 0.470 | 0.214 |
| regsub.all | 0.719 | 0.677 |
| high.pvals | 0.719 | 0.677 |
| vif.adj | 0.358 | 0.156 |
| vif.stepf | 0.555 | 0.500 |
| vif.stepb | 0.447 | 0.348 |
| vif.regsub | 0.719 | 0.677 |

## Summary

We started with a simple linear model that used all predictors. We tried a log transform on the model which seemed to lead to a more normal distribution of the crime data. When comparing models, however, it seemed there was not much improvement based on f-statistic, $R^2$, and predicted results.

Our next plan of attack was to remove variables which had high variance inflation factor values. These variables were more than likely causing very small yet misleading p-values due to autocorrelation among the variables. Once removed, we then refined our variable selection by performing various stepwise selection methods.

Finally, so that we could measure the performance of the models, we measured $R^2$ and $R^2_{adj}$ and threw data into a table along with the F-statistic, predict.lm output, and variables used.

Looking at the table we see, interestingly, when the regsubset method of variable selection was used on the linear model with all predictors, we get the exact same results as well hand-picking predictors with high p-values. It's even the same when using regsubsets on the VIF-removed linear model! I can't quite explain the intricacies of this selection but I'm hesitant to choose it due to all the concern I find online about using these methods.

At this point it seems like a toss up. I would probably further explore the model without high VIF variables which we used forward stepwise selection on (vif.stepf). It has a decent $R^2$ and $R^2_{adj}$ when compared to the standard model (all predictors) and it's predicted value is actually within the range of our expected values (1066).

It may actually be beneficial to try our original log-transformation idea again.

**Log-transform crime data on optimal model****

```
# try log transform on optimal model
# vif adjust lm, convert to log-linear
# create starting model with log(crime) as response
```

```r
lowfit <- lm(log(Crime) ~ 1, x)
# remove high vif and create full log(crime) model
crime.log.lm <- lm(log(Crime) ~ . -Po2-Wealth-U1-Ineq, x)
# use forward stepwise to select best predictors
stepf.log <- stepAIC(lowfit, direction = "forward", trace = FALSE,
                     scope = list(upper = crime.log.lm,
                              lower = lowfit))
summary(stepf.log)
```

```
##
## Call:
## lm(formula = log(Crime) ~ Po1 + M + Prob + So + Ed + U2 + LF,
##     data = x)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.45168 -0.16536  0.00834  0.14471  0.63155
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.22040    1.11415   1.095  0.28008
## Po1          0.08966    0.01636   5.482 2.71e-06 ***
## M            0.12191    0.04178   2.918  0.00582 **
## Prob        -5.22680    2.03505  -2.568  0.01416 *
## SoTRUE       0.41610    0.12838   3.241  0.00244 **
## Ed           0.12119    0.05555   2.182  0.03522 *
## U2           0.12825    0.05291   2.424  0.02008 *
## LF           2.56854    1.22688   2.094  0.04285 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.247 on 39 degrees of freedom
## Multiple R-squared:  0.6939, Adjusted R-squared:  0.639
## F-statistic: 12.63 on 7 and 39 DF,  p-value: 2.495e-08
```

```r
predict.lm(stepf.log, new.data) # 7.111562
```

```
##        1
## 7.111562
```

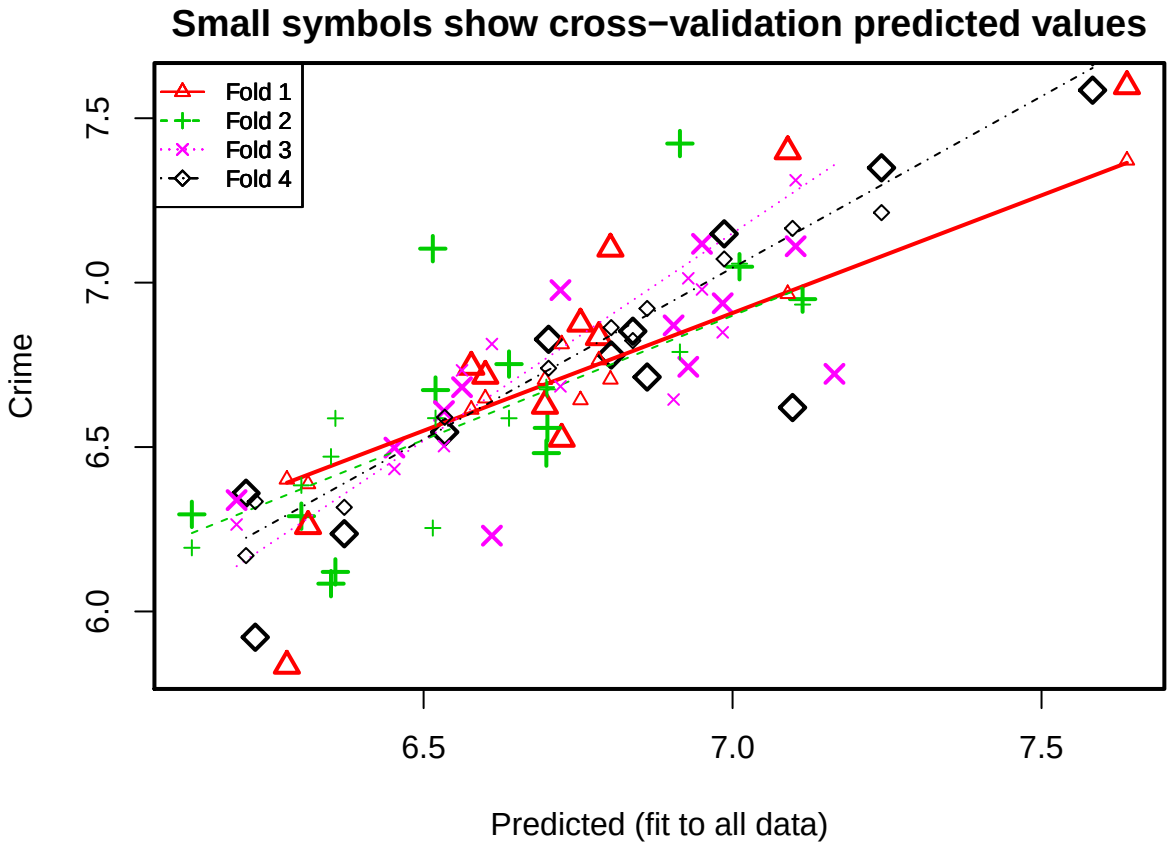
```r
exp(7.111562) #1226.061
```

```
## [1] 1226.061
```

```r
# calculate r2 and adjr2
xx <- as.data.frame(x)
xx <- xx %>%
  mutate(Crime = log(Crime))

crime.log.lm <- lm(Crime ~ . - Po2 - Wealth - U1 - Ineq, data = xx)

y <- cv.lm(xx, crime.log.lm, m=4)
```

```
## Analysis of Variance Table
##
## Response: Crime
```

```
##           Df Sum Sq Mean Sq F value  Pr(>F)
## M          1   0.02    0.02    0.37 0.54778
## So         1   0.00    0.00    0.00 0.99944
## Ed         1   1.19    1.19   17.83 0.00016 ***
## Po1        1   3.24    3.24   48.62 4.1e-08 ***
## LF         1   0.14    0.14    2.10 0.15656
## M.F        1   0.04    0.04    0.53 0.47254
## Pop        1   0.05    0.05    0.76 0.38907
## NW         1   0.00    0.00    0.06 0.81069
## U2         1   0.32    0.32    4.75 0.03613 *
## Prob       1   0.41    0.41    6.10 0.01858 *
## Time       1   0.03    0.03    0.40 0.53355
## Residuals 35   2.33    0.07
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```




```
##
## fold 1
## Observations in test set: 11
##                 2      6   12     18    24     25    26     27    28
## Predicted   7.089  6.723 6.58 6.7837 6.754  6.313 7.638  6.279 6.802
## cvpred      6.966  6.813 6.61 6.7643 6.643  6.387 7.372  6.402 6.706
## Crime       7.399  6.525 6.74 6.8341 6.875  6.260 7.597  5.835 7.103
## CV residual 0.434 -0.288 0.13 0.0698 0.232 -0.128 0.226 -0.567 0.398
##                32     39
## Predicted    6.6963 6.5998
```

```
## cvpred       6.7049 6.6484
## Crime        6.6254 6.7166
## CV residual -0.0795 0.0682
##
## Sum of squares = 0.9    Mean square = 0.08    n = 11
##
## fold 2
## Observations in test set: 12
##                  1     9     10    11      17     22   23     29    35
## Predicted   6.5193 6.638  6.700 6.915  6.3022  6.350 6.51 7.1133  6.70
## cvpred      6.5879 6.588  6.676 6.789  6.3835  6.471 6.25 6.9333  6.68
## Crime       6.6733 6.752  6.558 7.423  6.2897  6.084 7.10 6.9499  6.48
## CV residual 0.0854 0.165 -0.117 0.634 -0.0938 -0.386 0.85 0.0165 -0.20
##                   40    42     45
## Predicted    7.01121 6.125  6.357
## cvpred       7.05760 6.194  6.587
## Crime        7.04839 6.295  6.120
## CV residual -0.00921 0.102 -0.467
##
## Sum of squares = 1.6    Mean square = 0.13    n = 12
##
## fold 3
## Observations in test set: 12
##                 5     7     14      15     20    21    33    37     38
## Predicted   6.951 6.905 6.4527  6.5622  7.102 6.533 6.721  7.16 6.1973
## cvpred      6.979 6.645 6.4331  6.7332  7.311 6.503 6.685  7.93 6.2644
## Crime       7.118 6.870 6.4983  6.6821  7.111 6.609 6.977  6.72 6.3386
## CV residual 0.139 0.226 0.0652 -0.0511 -0.201 0.107 0.293 -1.21 0.0742
##                 44     46     47
## Predicted   6.9840  6.611  6.928
## cvpred      6.8488  6.813  7.013
## Crime       6.9373  6.230  6.744
## CV residual 0.0885 -0.583 -0.269
##
## Sum of squares = 2.1    Mean square = 0.17    n = 12
##
## fold 4
## Observations in test set: 12
##                3       4     8      13     16     19      30     31    34
## Predicted   6.21  7.5827 7.241  6.3716 6.8386  7.097  6.5342  6.228 6.702
## cvpred      6.17  7.6837 7.213  6.3167 6.8239  7.165  6.5904  6.334 6.740
## Crime       6.36  7.5853 7.349  6.2364 6.8522  6.620  6.5453  5.922 6.828
## CV residual 0.19 -0.0984 0.136 -0.0803 0.0284 -0.545 -0.0451 -0.413 0.088
##                36      41     43
## Predicted   6.9865  6.8036  6.862
## cvpred      7.0719  6.8632  6.921
## Crime       7.1483  6.7799  6.713
## CV residual 0.0765 -0.0832 -0.208
##
## Sum of squares = 0.6    Mean square = 0.05    n = 12
##
## Overall (Sum over all 12 folds)
##    ms
## 0.111
```

```
y.resid <- sum((y$Crime - y$cvpred)^2)
y.sst   <- sum((y$Crime - mean(y$Crime))^2)
rsq <- 1-(y.resid/y.sst)
k <- length(crime.log.lm$coefficients)-1
n <- as.numeric(length(y$Crime))
adjr2 <- 1-(((1-rsq)*(n-1))/(n-k-1))

rsq   # 0.33
```

```
## [1] 0.33
```

```
adjr2 # 0.12
```

```
## [1] 0.12
```

**Model, predictors, model $R^2_{adj}$ & F-statistic**

```
knitr::kable(summ2[1:5])
```

| model.used | predictors.used | adj.r2 | f.statistic | prediction |
|---|---|---|---|---|
| stepf.log | M,So,Ed,Po1,LF,U2,Prob | 0.639 | 12.6 | 1224 |

**Model, cross-validated $R^2$ & $R^2_{adj}$**

```
knitr::kable(summ2[c(1,6:7)])
```

| model.used | cv.r2 | cv.adjr2 |
|---|---|---|
| stepf.log | 0.33 | 0.12 |

Looks like we got a decent value and a not so bad $R^2_{adj}$ for the model. We see some odd values when calculated $R^2$ & $R^2_{adj}$ on the cross validated results. Regardless of the low R values, I think I would certainly research a better way to cross validate, report model performance, and try to use this model. To make a more precise model likely requires more domain knowledge and expertise on the subject. It seems obvious that Wealth and Inequality would suffer from autocorrelation in hindsight. Someone with more background would likely be able to filter through some of these predictors and make better decisions on which to choose.

# References


gung (https://stats.stackexchange.com/users/7290/gung). Algorithms for automatic model selection. Cross Validated. URL https://stats.stackexchange.com/q/20856. URL:https://stats.stackexchange.com/q/20856 (version: 2012-01-11).

Fomite (https://stats.stackexchange.com/users/5836/fomite). Can i ignore coefficients for non-significant levels of factors in a linear model? Cross Validated. URL https://stats.stackexchange.com/q/24310. URL:https://stats.stackexchange.com/q/24310 (version: 2012-03-08).

Ellie (https://stats.stackexchange.com/users/9686/ellie). Can i ignore coefficients for non-significant levels of factors in a linear model? Cross Validated. URL https://stats.stackexchange.com/q/24305. URL:https://stats.stackexchange.com/q/24305 (version: 2012-03-08).

user83346. How to interpret a vif of 4? Cross Validated. URL https://stats.stackexchange.com/q/169449. URL:https://stats.stackexchange.com/q/169449 (version: 2015-08-31).