ISYE 6501 - Homework 5

Justin Lewis

February 7, 2019

Start by clearing variables and importing packages

```
rm(list=ls())
library(kernlab, quietly=TRUE)
```

```
## Warning: package 'kernlab' was built under R version 3.5.2
```

```
library(knitr, quietly=TRUE)
library(ggplot2, quietly=TRUE)
```

```
## Warning: package 'ggplot2' was built under R version 3.5.2
```

```
##
## Attaching package: 'ggplot2'
```

```
## The following object is masked from 'package:kernlab':
##
##     alpha
```

Question 8.1

Q:

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.

A:

In day to day life I drive a bit of an older Jeep, which is not very good on gas. In order to model the gas mileage I can expect we could use a linear regression model with the weight of the vehicle and passengers, the speed I'm going at any one time, and the grade of the hill I am driving on at any time. Most likely the relationship is not nicely linear, but by creating polynomial features from these three we could probably fit a reasonable model.

Question 8.2

Q:

Using crime data from http://www.statsci.org/data/general/uscrime.txt (http://www.statsci.org/data/general/uscrime.txt) (file uscrime.txt, description at http://www.statsci.org/data/general/uscrime.html (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:

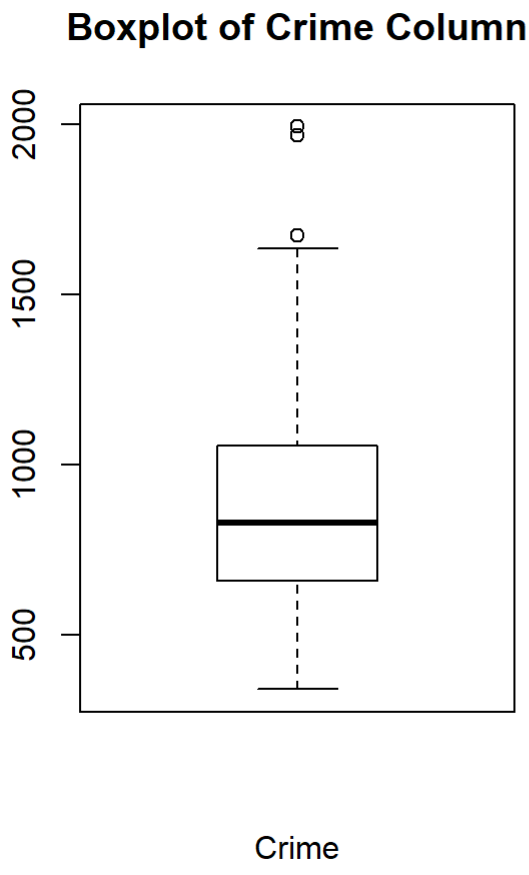
```
input = list(14.0, 0, 10.0, 12.0, 15.5, 0.640, 94.0, 150, 1.1, 0.120, 3.6, 3200, 20.1, 0.04, 39.
0)
```

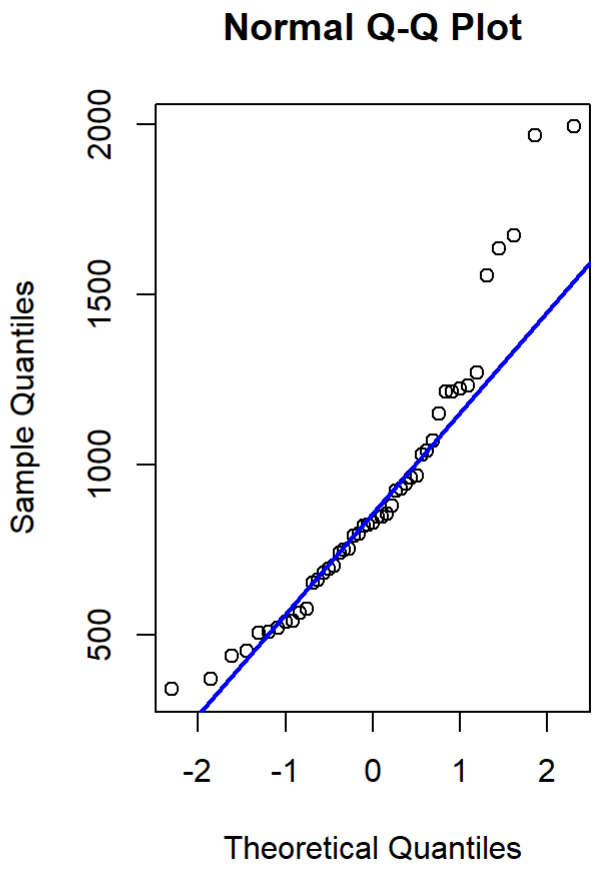
A:

First we are going to need to import our data, and investigate:

```
#Import data
data <- as.data.frame(read.csv('uscrime.txt', header=TRUE, sep='\t'))

#Check the data for outliers using boxplot and qqplot
par(mfrow=c(1,2))
boxplot(data$Crime, xlab='Crime', main='Boxplot of Crime Column')
qqnorm(data$Crime)
qqline(data$Crime, col='Blue', lw=2)
```





So it looks like our data is not normally distributed, and has a few outliers. That is good to know for when we begin looking at our model results.

Now going to define the attributes to be used in the model, and with R we can immediately just fit a model to every column naively, and examine the results.

```
#Define the data we can use for predictions
attribs <- data[, 1:ncol(data)-1]
lm_all <- lm(data$Crime~., attribs)
```

Now we have a model fit using every attribute, linearly attempting to predict the data['Crime'] column. This method is not ideal, because we do not really know if the attributes we supplied are the optimal ones. To check this we can look over the summary of our model.

```
naivemodel_summary <- summary(lm_all)
print(naivemodel_summary)
```

```
##
## Call:
## lm(formula = data$Crime ~ ., data = attribs)
##
## 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 *
## So          -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
```

From this summary we can see a lot of information, including the calculated p-values for each attribute of the model. Recalling that a typical p-value threshold is 0.05, we can see which attributes are most significant, and which ones most likely have no real relationship with our response variable (Crime).

Using this summary value, we can grab all the attributes with p-values over 0.05 and remove them. However, we should not remove them all at once since as each attribute is removed, re-fitting our model will produce different p-values. First, the attribute with the highest p-value will be removed. Next, the model will be re-fit with only the remaining attributes. A new summary will be produced, and this will be repeated until there are only significant attributes remaining in the model

```
#Create a new df from the coefficients table in the summary
nms_df <- as.data.frame(naivemodel_summary['coefficients'])

#Order the df based on the values in the p-values column
nms_df <- nms_df[order(nms_df$coefficients.Pr...t..),]

#Reverse the resulting df since I could not get a descending option in order() to work
nms_df <- nms_df[dim(nms_df)[1]:1,]

#Grab only the data where the p-value is above 0.05
high_pv <- nms_df[nms_df['coefficients.Pr...t..'] > 0.05,]['coefficients.Pr...t..']

#Now check which attribute has the highest p-value
print(paste('The attribute with the highest p-value is', row.names(high_pv)[[1]]))
```

```
## [1] "The attribute with the highest p-value is So"
```

With this, we can now make new attributes and build a new model.

```
#Create new attributes with the highest o
new_attribs <- attribs[, !(names(attribs) %in% c(row.names(high_pv)[[1]]))]
new_m <- lm(data$Crime~., new_attribs)
new_s <- as.data.frame(summary(new_m)['coefficients'])

print(new_s['coefficients.Pr...t..'])
```

```
##             coefficients.Pr...t..
## (Intercept)          0.0007113266
## M                    0.0395203338
## Ed                   0.0041698787
## Po1                  0.0741517742
## Po2                  0.3508553433
## LF                   0.6157358444
## M.F                  0.3878763591
## Pop                  0.5675732764
## NW                   0.4800023797
## U1                   0.1411060133
## U2                   0.0420807933
## Wealth               0.3430412342
## Ineq                 0.0018339190
## Prob                 0.0353441414
## Time                 0.6245556161
```

Looking at this we can see that our new summary has different p-values for the remaining attributes which were used in the model. Now we are basically going to want to repeat the same process until all p-values are below our significance level. This is tedious, so I'll define a function to do it using recursion.

```
back_prop <- function(summ, atts, response){
  #Take in summary, grab coefficients, order the p-values, remove the highest from attributes
  #Create a new df from the coefficients table in the summary
  ms_df <- as.data.frame(summ['coefficients'])[2:nrow(as.data.frame(summ['coefficients'])),]
  ms_df <- ms_df[order(ms_df$coefficients.Pr...t..),]
  ms_df <- ms_df[dim(ms_df)[1]:1,]
  high_pvs <- ms_df[ms_df['coefficients.Pr...t..'] > 0.05,]['coefficients.Pr...t..']

  #If there are p-values > 0.05, make a new model after removing the highest p-value attr
  if (nrow(high_pvs) > 0){
    print(paste('Removing:', row.names(high_pvs)[[1]], 'from attributes.'))
    atts <- atts[, !(names(atts) %in% c(row.names(high_pvs)[[1]]))]
    new_m <- lm(response~., atts)
    new_s <- summary(new_m)
    back_prop(new_s, atts, response)
  }
  #Otherwise, return the good attributes
  else if (nrow(high_pvs) == 0){
    print(paste('Remaining attributes:', list(colnames(atts))))
    return(atts)
  }
}
```

Now we can try it using the results from our first iteration.

```
#Use the back_prop algorithm to get the relevant attributes
rel_atts <- back_prop(summary(new_m), new_attribs, data$Crime)
```

```
## [1] "Removing: Time from attributes."
## [1] "Removing: LF from attributes."
## [1] "Removing: NW from attributes."
## [1] "Removing: Po2 from attributes."
## [1] "Removing: Pop from attributes."
## [1] "Removing: Wealth from attributes."
## [1] "Removing: M.F from attributes."
## [1] "Removing: U1 from attributes."
## [1] "Remaining attributes: c(\"M\", \"Ed\", \"Po1\", \"U2\", \"Ineq\", \"Prob\")"
```

```
#Make a new model
model <- lm(data$Crime~., rel_atts)

#Lets check the summary with our new model
summary(model)
```

```
##
## Call:
## lm(formula = data$Crime ~ ., data = rel_atts)
##
## 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 ***
## M             105.02      33.30   3.154  0.00305 **
## Ed            196.47      44.75   4.390 8.07e-05 ***
## Po1           115.02      13.75   8.363 2.56e-10 ***
## U2             89.37      40.91   2.185  0.03483 *
## Ineq           67.65      13.94   4.855 1.88e-05 ***
## Prob        -3801.84    1528.10  -2.488  0.01711 *
## ---
## 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
```

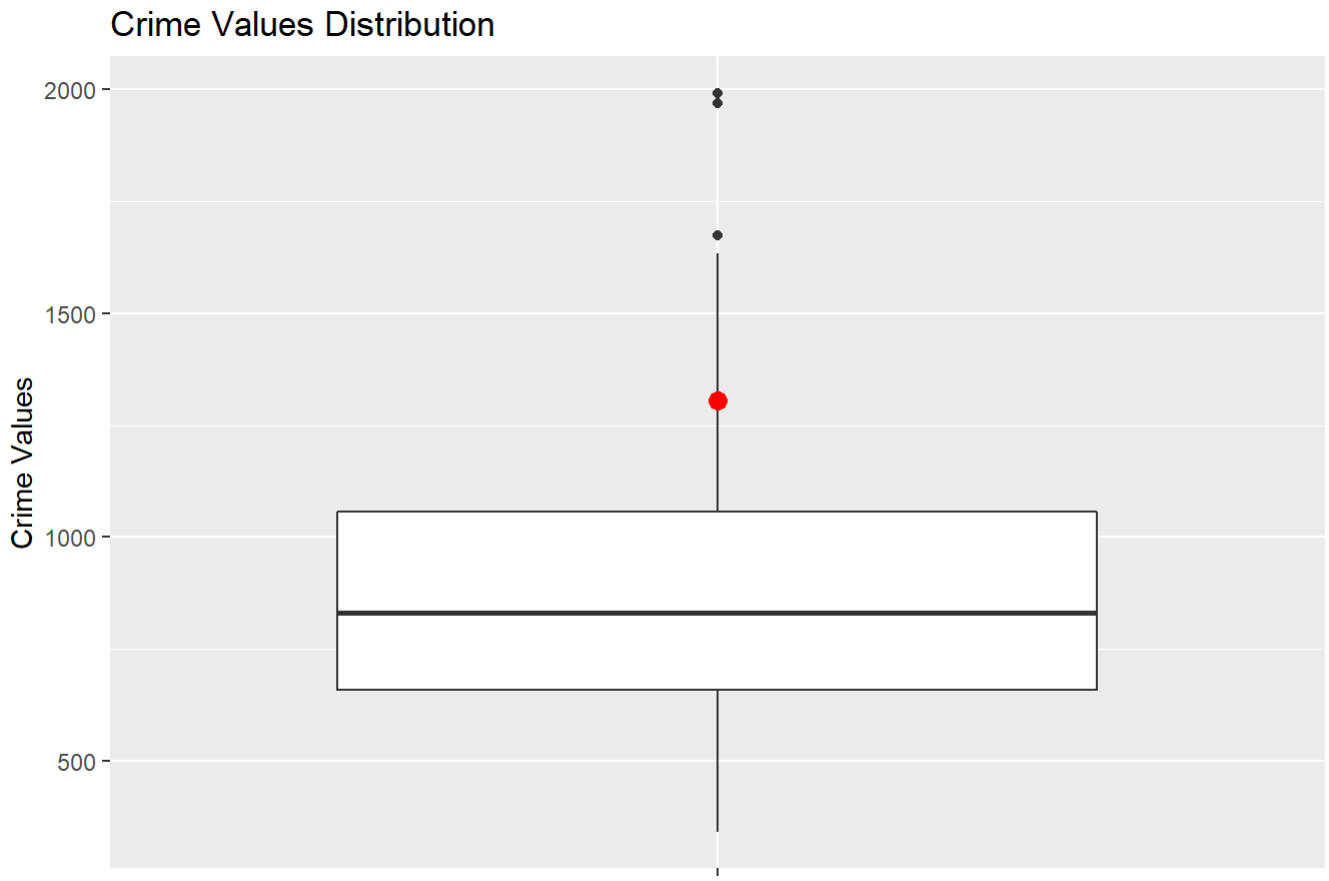
We can see that our R-squared and adjusted R-squared values are still fairly high, all of the p-values for the remaining attributes are well within the 0.05 threshold, and the overall p-value of the plot is very low. If we compare the R-squared value to the first initial model, we see a small reduction, but the adjusted R-squared value has actually increased. Additionally, the estimated coefficients for the attributes are all far away from zero, telling us that they are very relevant to the model. Overall with these results I am content with this model, and will use these attributes in the final model.

At this point we can attempt to make our prediction for the input data, but only using the relevant attributes.

```
#Build the prediction for the data
nd <- t(as.vector(c(14.0, 10.0, 12.0, 3.6, 20.1, 0.04)))
coefs <- row.names(summary(model)$coefficients)[c(1:length(nd)+1)]
df = as.data.frame(nd)
colnames(df) = coefs
p <- predict(model, newdata=df)
predicted_value <- p[[1]]
```

Using the resulting model from the feature selection process, and the provided data's relevant attributes, the predicted Crime value is 1304.2452107. To see if this is a realistic value within the range of the 'Crime' values, we can check a boxplot with the resulting point overlayed. If it is within the whiskers it is a good indication that our prediction is a reasonable value.

```
qplot(x='', y=data$Crime, geom='boxplot', ylab='Crime Values', main='Crime Values Distribution')
+ geom_point(aes(x='', y=c(predicted_value)), color='red',size=3)
```



The red point is the resulting prediction, which does seem to be a reasonable value.

This was all done without any kind of scaling. Now using the same back_prop function, I can easily check how much of a difference scaling the data before analysis will make.

```
#Scale the data
scaled_data <- as.data.frame(scale(data))

#Get our scaled attributes
scaled_attribs <- as.data.frame(scaled_data[, 1:ncol(data)-1])

#Build the first model with everything
scm_1 <- lm(scaled_data$Crime~., scaled_attribs)

#Now look at the summary
scm_1_summ <- summary(scm_1)
print(scm_1_summ)
```

```
##
## Call:
## lm(formula = scaled_data$Crime ~ ., data = scaled_attribs)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.02321 -0.25361 -0.01731  0.29214  1.32554
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.212e-16  7.885e-02   0.000  1.00000
## M            2.854e-01  1.355e-01   2.106  0.04344 *
## So          -4.710e-03  1.842e-01  -0.026  0.97977
## Ed           5.447e-01  1.796e-01   3.033  0.00486 **
## Po1          1.482e+00  8.154e-01   1.817  0.07889 .
## Po2         -7.911e-01  8.493e-01  -0.931  0.35883
## LF          -6.936e-02  1.536e-01  -0.452  0.65465
## M.F          1.326e-01  1.551e-01   0.855  0.39900
## Pop         -7.215e-02  1.269e-01  -0.568  0.57385
## NW           1.118e-01  1.723e-01   0.649  0.52128
## U1          -2.716e-01  1.963e-01  -1.384  0.17624
## U2           3.664e-01  1.798e-01   2.038  0.05016 .
## Wealth       2.399e-01  2.586e-01   0.928  0.36075
## Ineq         7.290e-01  2.343e-01   3.111  0.00398 **
## Prob        -2.854e-01  1.336e-01  -2.137  0.04063 *
## Time        -6.375e-02  1.313e-01  -0.486  0.63071
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5405 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
```

Looks pretty comparable to the previous model, but note that the intercept now has a p-value of 1, with an extremely small coefficient. Likely a result of the scaling making it so that the intercept goes through zero and does not move. We can run through the function to select relevant attributes.

```
screl_atts <- back_prop(scm_1_summ, scaled_attribs, scaled_data$Crime)
```

```
## [1] "Removing: So from attributes."
## [1] "Removing: Time from attributes."
## [1] "Removing: LF from attributes."
## [1] "Removing: NW from attributes."
## [1] "Removing: Po2 from attributes."
## [1] "Removing: Pop from attributes."
## [1] "Removing: Wealth from attributes."
## [1] "Removing: M.F from attributes."
## [1] "Removing: U1 from attributes."
## [1] "Remaining attributes: c(\"M\", \"Ed\", \"Po1\", \"U2\", \"Ineq\", \"Prob\")"
```

```
scmodel <- lm(scaled_data$Crime~., screl_atts)
print(summary(scmodel))
```

```
##
## Call:
## lm(formula = scaled_data$Crime ~ ., data = screl_atts)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.21696 -0.20274 -0.05089  0.34419  1.43817
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.378e-16  7.569e-02   0.000  1.00000
## M            3.413e-01  1.082e-01   3.154  0.00305 **
## Ed           5.683e-01  1.295e-01   4.390 8.07e-05 ***
## Po1          8.838e-01  1.057e-01   8.363 2.56e-10 ***
## U2           1.951e-01  8.932e-02   2.185  0.03483 *
## Ineq         6.979e-01  1.438e-01   4.855 1.88e-05 ***
## Prob        -2.235e-01  8.983e-02  -2.488  0.01711 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5189 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
```

And we end up with an identical model, but the intercept now goes through 0. Additionally we have a much smaller residual standard error on the degrees of freedom.