Math 651 Final Project

Mary Peng, Hillary Dunn, and Max Kearns

December 17, 2018

1 Abstract

2 Introduction

2.0.0.0.1 needs editing

Every four summers, the Olympic Games become the center of the world's attention, as elite athletes seek honor for both themselves and for their countries. Many countries associate tremendous national pride with their medal counts, since a nation's athletic competence also projects its soft power. For this last reason, some governments invest generously in their sports programs, in hopes of an elevated medal count in the next Olympics.

Given the fierce competition and high profile nature of medal counts, one may wonder what factors influence the number of medals that a country wins at the summer Olympics. Certainly countries with the largest economies and populations, such as the United States and China, commonly dominate the top of the billboard. However, Azerbaijan, which ranks 91st in population and 72nd in total GDP, also ranked in the top 20 countries by total medal count in the 2016 Summer Olympics.

Our team will develop multiple regression models using various predictors, such as GDP, population size, size of athletic investment, and geographic location, to predict medal count. We will build the model on countries' total medal count for the 2004, 2008, and 2012 Olympics, and then project the model onto the 2016 Olympics to understand the accuracy of the model's predictions.

3 Methods and Materials

3.1 Data

The purpose of this reasearch is to determine whether there is a predictive relationship between popluation, GDP per capita, being a host nation, or at one point being a Communist nation or a member of the Soviet Union and number of medals won at the Olympics in the associated year by country.

3.1.1 EDA–choose what we want to keep, and then format as figures in Appendix

As we can see IN FIGURE. . . , the average number of medals won by a country in a year is 11.79 medals and the minimum number of medals is one. This is meaningful because this means we are only assessing countries that have won at least one medal in at least one of the years we are analyzing.

Below we see that the medal count is distributed exponentially, which could indicate a need to transform the response variable if we need to fit a linear regression model.

X_1: This variable indicates population of people in the country in the associated year. Below, we can see a histogram of the population data. On the left, is the original distribution of the data points and on the right, we've transformed the data to look more normally distributed.

X_2: This variable indicates GDP per capita in the associated year. Below, we can see a histogram of the GDP per capita data. On the left, is the original distribution of the data points and on the right, we've transformed the data to look more normally distributed.

X_3: This binary predictor variable indicates with a 1 if a country hosted the Olympics within the previous 8 years, whether the country is hosting the Olympics in that year, or if the country is hosting within 8 years in the future. Of all the data points, 25 are classified as host within 8 years prior, current host or future host within 8 years. This makes sense because we are considering 5 years in our training data and each year has five countries that can be classified by this predictor variable.
X_4: This binary predictor variable indicates whether the country was once a member of the Soviet Union or if they were indicated as ever being a Communist country. From _______________ we aggregated a binary predictor variable that identifies countries that were once members of the Soviet Union or at one time classified as communist countries. 111 data points are classified as former Soviet Union or Communist.(INSERT SOURCE AGAIN???)

In addition to the predictors and response variable, each data point has an associated country and year.

INSERT WHY WE DECIDED TO GO WITH LOG OF VARIABLES

To properly determine an ideal model for the data, we need to address the relationships between variables. We can do this be analyzing both the scatterplot matrix of variables and the correlation matrix.

The scatterplot matrix does not obviously show us many relationships between the predictor variables and their effect on the response variable. While there might be a relationship between the log of the population and the medal count, it is difficult to determine from the plot if this is a linear relationship. Additional information can be collected from the correlation matrix below.

This correlation matrix suggests that collinearity between variables will not be a significant issue. The most significant correlation between predictor variables is between the log of the GDP per capita and the indicator of Communism or Soviet Union inclusion variable. However, the correlation coefficient is only -0.2989.

Summary of Medal Count:

Distribution of Medal Count:

Distribution of Medal Count in 2008:

GDP Summary:

Population Summary:

Total Count of Communist / comm_soviet Countries:

Count of host, hosted within 8 years prior, or will be hosting within 8 years:

Histograms of Population:

Boxplot of log(GDP):

Boxplot of log(Population):

Scatterplot Matrix of Variables:

Correlation Matrix of Predictor Variables and Response Variable:

3.2 Model Building and Selection

3.2.1 Linear Model

According to the selected model diagnostics, the model with the smallest C_p statistic, largest adjusted R^2 value, smallest AIC and smallest PRESS value is the model that includes all four predictor variables.

```r
olympic.lmfinal <- lm(log_count ~ log_pop + log_gdp_per_cap + host + comm_soviet, data = Olympic_v2)
summary(olympic.lmfinal)
```

```
##
## Call:
## lm(formula = log_count ~ log_pop + log_gdp_per_cap + host + comm_soviet,
##     data = Olympic_v2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.3699 -0.6238  0.0294  0.6488  2.0421
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -9.63058    0.63841 -15.085  < 2e-16 ***
## log_pop          0.44275    0.02937  15.077  < 2e-16 ***
## log_gdp_per_cap  0.40830    0.03212  12.714  < 2e-16 ***
## host             0.86726    0.19029   4.558 6.98e-06 ***
## comm_soviet      1.03281    0.10469   9.866  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8666 on 382 degrees of freedom
## Multiple R-squared:  0.531,  Adjusted R-squared:  0.526
## F-statistic: 108.1 on 4 and 382 DF,  p-value: < 2.2e-16
```

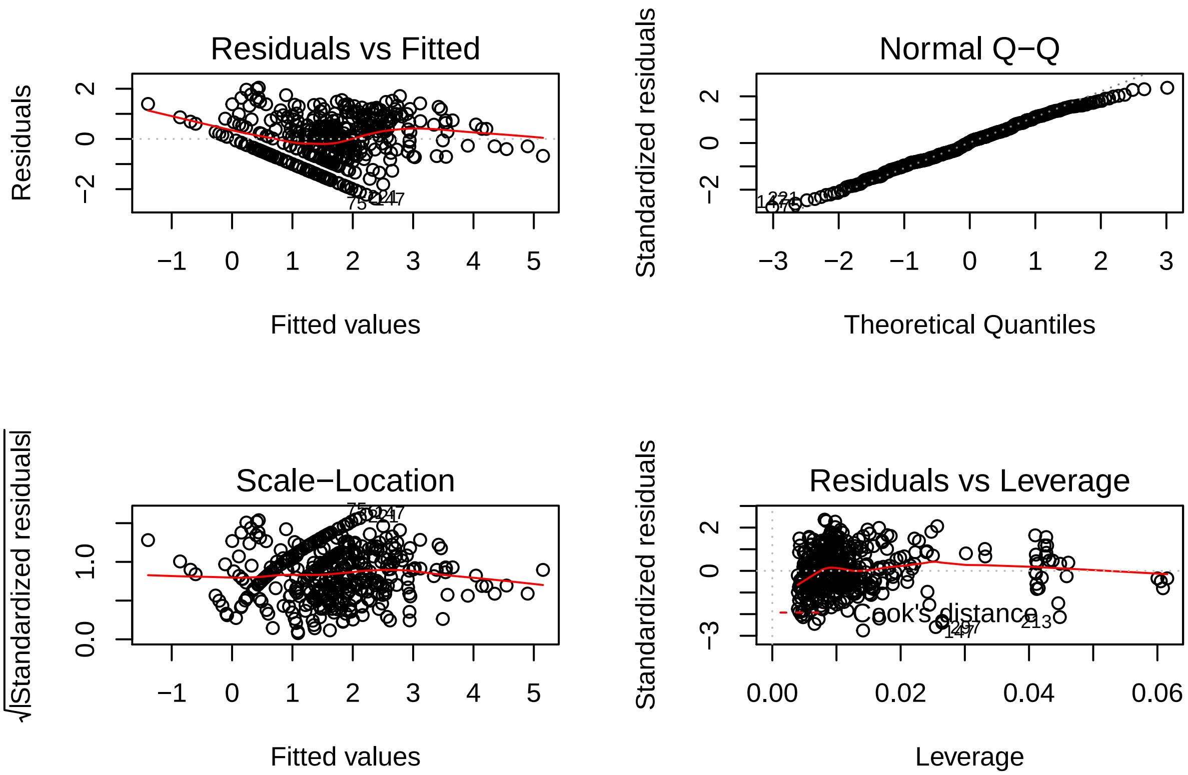
$$X_1 = \log(\text{population})$$

$$X_2 = \log(\text{gdp/capita})$$

$$X_3 = \text{host}$$

$$X_4 = \text{Soviet Country or Communist}$$

$$\hat{Y} = -9.63058 + 0.44275X_1 + 0.40830X_2 + 0.86726X_3 + 1.03281X_4$$



ssumptions: According to the Normal Q-Q plot, it is reasonable to assume the residuals follow a normal distribution. Hoever, the Residuals vs Fitted plot indicates possible heteroskedasticity or non-constant variance of the errors. The Breusch-Pagan test will help us determine if heteroskedasticity is affecting the model.

Breusch Pagan Test: Assuming $Var(\epsilon_i) = \sigma_i^2$ such that $\log_e \sigma_i^2 = \gamma_0 + \gamma_1 X_i$: Alternatives:

$$H_0 : \gamma_1, \text{vs. } H_a : \gamma_1 \neq 0$$

Decision Rule: At significance level $\alpha = 0.05$, if the p-value of the Breusch-Pagan test is less than α then we reject H_0 and accept H_a, otherwise we fail to reject H_0. Accepting H_a means we accept that the variance is non-constant.

The p-value $= 0.001587 < \alpha$, therefore we accept the alternative hypothesis that our variance is not constant.

Test for normality (Lilliefors Test):
Alternatives:

$$H_0 : \text{Sample comes from a N}(\mu, \sigma^2) \text{ distribution}$$

$$H_a : \text{Sample does not come from a N}(\mu, \sigma^2) \text{ distribution}$$

Decision Rule: At signifcance level $\alpha = 0.05$, if the p-value is less than α then we accept H_a and reject H_0, otherwise we fail to reject H_0.

```r
lillie.test(olympic.lmfinal$residuals)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  olympic.lmfinal$residuals
## D = 0.036681, p-value = 0.2332
```

The p-value= $0.2332 > \alpha$, thus we fail to reject H_0. Therefore, it is reasonable to assume that the error terms are distributed normally.

#Diagnostics vif(olympic.lmfinal)

Influential Cases:

```
#Influential cases
olympic.lm_inf=influence.measures(olympic.lmfinal)$is.inf
idx=which(apply(olympic.lm_inf,1,any))

Olympic[idx,]
```

```
##             country count year          gdp        pop host comm_soviet
## 2             China   100 2008 4.598206e+12 1324655000    1           1
## 4    United Kingdom    47 2008 2.890564e+12   61806995    1           0
## 5         Australia    46 2008 1.052585e+12   21249200    1           0
## 17           Brazil    15 2008 1.695825e+12  192979029    1           0
## 20          Jamaica    11 2008 1.367861e+10    2790122    0           0
## 48           Greece     4 2008 3.544608e+11   11077841    1           0
## 75          Vietnam     1 2008 9.913030e+10   86707801    0           1
## 86    United States   101 2004 1.227493e+13  292805298    1           0
## 88            China    63 2004 1.955347e+12 1296075000    1           1
## 90        Australia    49 2004 6.119043e+11   20127400    1           0
## 94   United Kingdom    31 2004 2.398555e+12   59987905    1           0
## 102          Greece    16 2004 2.405213e+11   10955141    1           0
## 147           India     1 2004 6.996889e+11 1126135777    0           0
## 238           China    88 2012 8.560547e+12 1350695000    1           1
## 240  United Kingdom    65 2012 2.662085e+12   63700300    1           0
## 242           Japan    38 2012 6.203213e+12  127629000    1           0
## 252          Brazil    17 2012 2.465189e+12  200560983    1           0
## 258         Jamaica    12 2012 1.480017e+10    2840992    0           0
## 294          Greece     2 2012 2.456707e+11   11045011    1           0
## 302    Saudi Arabia     1 2012 7.359748e+11   29086357    0           0
## 303       Venezuela     1 2012 3.812862e+11   29893080    0           0
## 318   United States    94 2000 1.028478e+13  282162411    1           0
## 320           China    58 2000 1.211347e+12 1262645000    1           1
## 321       Australia    58 2000 4.150342e+11   19153000    1           0
## 336          Greece    13 2000 1.301338e+11   10805808    1           0
## 340           Spain    11 2000 5.954026e+11   40567864    1           0
## 343         Jamaica     9 2000 8.985353e+09    2656864    0           0
## 378           India     1 2000 4.621468e+11 1053050912    0           0
## 395   United States   101 1996 8.100201e+12  269394000    1           0
## 398           China    50 1996 8.637467e+11 1217550000    0           1
## 399       Australia    41 1996 4.003027e+11   18311000    1           0
## 402     South Korea    27 1996 5.980991e+11   45524681    1           0
## 409           Spain    17 1996 6.409983e+11   39889852    1           0
## 420          Greece     8 1996 1.458616e+11   10608800    1           0
## 452           India     1 1996 3.876560e+11  978893217    0           0
## 453          Mexico     1 1996 4.109756e+11   95687452    0           0
## 468           Tonga     1 1996 2.195836e+08      96369    0           0
```

APPENDIX: A. Influential Cases by test:

```
olympic.lm_inf[idx,]
```

```
##     dfb.1_ dfb.lg_p dfb.l___ dfb.host dfb.cmm_ dffit cov.r cook.d   hat
```

```
## 2    FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE  TRUE
## 4    FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE  TRUE
## 5    FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE  TRUE
## 17   FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE  TRUE
## 20   FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE FALSE
## 48   FALSE    FALSE    FALSE    FALSE    FALSE FALSE FALSE  FALSE  TRUE
## 75   FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE FALSE
## 86   FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE  TRUE
## 88   FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE  TRUE
## 90   FALSE    FALSE    FALSE    FALSE    FALSE FALSE FALSE  FALSE  TRUE
## 94   FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE  TRUE
## 102  FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE  TRUE
## 147  FALSE    FALSE    FALSE    FALSE    FALSE  TRUE  TRUE  FALSE FALSE
## 157  FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE  TRUE
## 159  FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE  TRUE
## 161  FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE  TRUE
## 171  FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE  TRUE
## 177  FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE FALSE
## 213  FALSE    FALSE    FALSE    FALSE    FALSE  TRUE FALSE  FALSE  TRUE
## 221  FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE FALSE
## 222  FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE FALSE
## 237  FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE  TRUE
## 239  FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE  TRUE
## 240  FALSE    FALSE    FALSE    FALSE    FALSE FALSE FALSE  FALSE  TRUE
## 255  FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE  TRUE
## 259  FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE  TRUE
## 262  FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE FALSE
## 297  FALSE    FALSE    FALSE    FALSE    FALSE  TRUE FALSE  FALSE FALSE
## 314  FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE  TRUE
## 317  FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE FALSE
## 318  FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE  TRUE
## 321  FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE  TRUE
## 328  FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE  TRUE
## 339  FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE  TRUE
## 371  FALSE    FALSE    FALSE    FALSE    FALSE  TRUE FALSE  FALSE FALSE
## 372  FALSE    FALSE    FALSE    FALSE    FALSE FALSE  TRUE  FALSE FALSE
## 387  FALSE    FALSE    FALSE    FALSE    FALSE FALSE FALSE  FALSE  TRUE
```

3.2.2 Generalized Linear Model

The first choice for a glm is typically a Poisson model, which may provide a reasonable model for these data. This generalized model was created with the same variables as the linear model; population, GDP per capita, the host dummy variable, and the communist/soviet dummy variable. Like the linear model, the population and GDP per capita were log-transformed, but in this model it is unnecessary to transform the medal count. The full poisson model showed highly significant parameters, but the dispersion of this model was much greater than 1 (disp. = 6.621), so a negative binomial model would likely provide a better fit for these data.

A negative binomial regression model assumes that dispersion is greater than 1, which is consistent with these data. Therefore, it allows for more accurate tests of the parameters. Among the negative binomial models, the best model again proved to be the full model, after a comparison of AIC between all subsets of variables (Table 2).

4 Results

5 Discussion and Conclusions

6 Bibliography

7 Appendix A

Table 1: Model Selection Diagnostics for a Negative Binomial Model

Pop	GDP/C	Host	Soviet	Parameters	Cp.nb	AIC.nb
1	0	0	0	2	198.46	2497.82
0	0	1	0	2	215.98	2633.68
0	1	0	0	2	336.41	2658.69
0	0	0	1	2	370.52	2695.53
1	0	1	0	3	108.70	2474.29
1	1	0	0	3	119.18	2428.33
1	0	0	1	3	180.88	2489.15
0	1	1	0	3	199.48	2601.80
0	0	1	1	3	206.80	2626.84
0	1	0	1	3	318.21	2614.08
1	1	0	1	4	58.39	2338.79
1	1	1	0	4	60.95	2417.52
1	0	1	1	4	87.77	2457.19
0	1	1	1	4	178.38	2556.83
1	1	1	1	5	5.00	2319.23

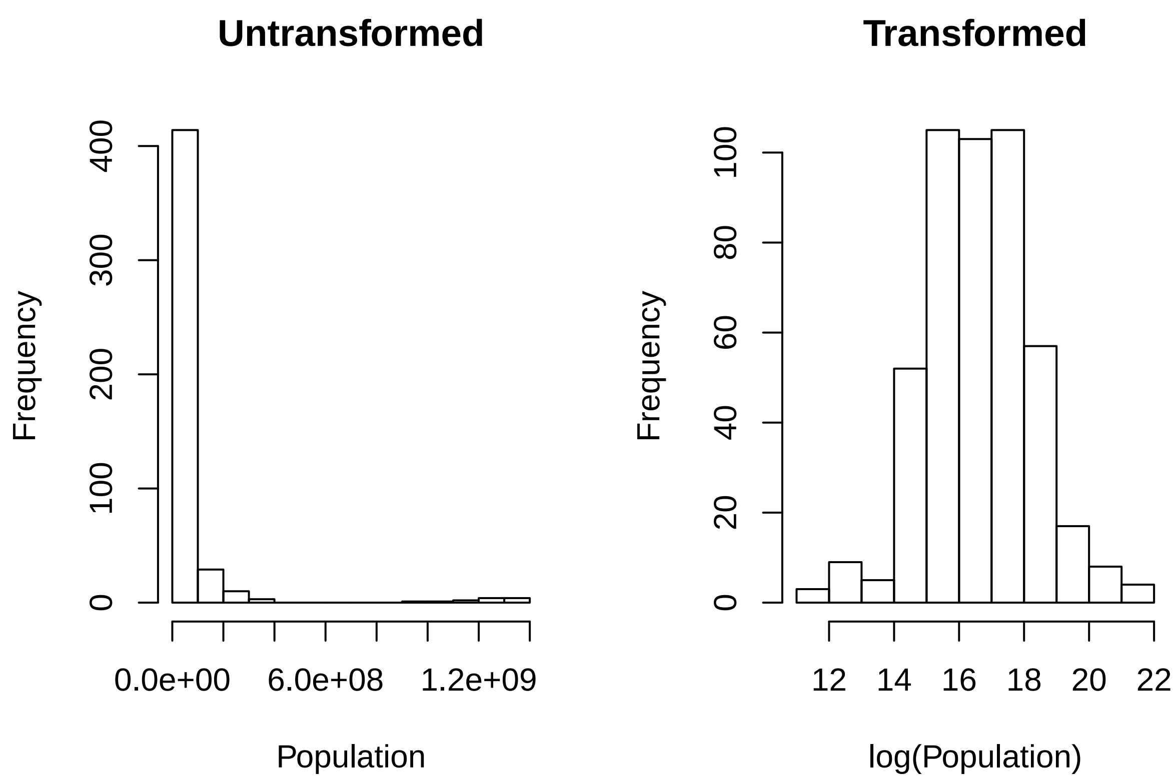


Figure 1: This is a test figure.

Table 2: Negative Binomial Model Coefficients

	Estimate	Std. Error	z value	p-value
Intercept	-10.1657683	0.6468624	-15.715504	0.00e+00
log(Population)	0.4984760	0.0288138	17.299910	0.00e+00
log(GDP/Capita)	0.4023884	0.0317426	12.676614	0.00e+00
Host	0.6926724	0.1593195	4.347693	1.38e-05
Comm/Soviet	1.0337603	0.0984874	10.496376	0.00e+00

8 Appendix B

```
knitr::opts_chunk$set(echo = FALSE, comment = NA)
base<-read.csv('data/base_data.csv', stringsAsFactors = F)
P__disp <- function(x) {
  pr <- sum(residuals(x, type="pearson")^2)
  dispersion <- pr/x$df.residual
  c(pr, dispersion)
}
library(dplyr)
library(qpcR)
library(MuMIn)

base.total = base[which(base$year!=2016),]
Olympic = base.total[,c(2,3,4,5,6,7,8)]

#Clean the data (Mary to add)

#Make new dataframe with GDP / capita
Olympic_v2 <- data.frame(year = Olympic$year, country = Olympic$country, count = Olympic$count, log_pop
library(knitr)
summary(base$count)
hist(base$count)
hist(base[which(base$year == c("2008")),c("count")])
summary(base$gdp)
summary(base$pop)
sum(base$comm_soviet)
nrow(base)-sum(base$comm_soviet)
sum(base$host)
par(mfrow=c(1,2))
#Histogram Population
hist(base$pop,main = "Untransformed",xlab = "Population")
#Histogram log(Population)
hist(log(base$pop),main = "Transformed",xlab = "log(Population)")
#Log(GDP)
boxplot(log(base$gdp),ylab = "log(GDP)")
#log(Population)
boxplot(log(base$pop),ylab = "log(Population)")
pairs(Olympic[,c(2,4,5,6,7)])
kable(cor(Olympic[,c(2,4,5,6,7)]))
library(leaps)
#CP
olympic.leapCP <- leaps(y=log(Olympic_v2$count), x=Olympic_v2[,4:7])
#R2a
olympic.leapR2a <- leaps(y=log(Olympic_v2$count), x=Olympic_v2[,4:7], method = 'adjr2')

  ### Code for AIC
xList <- names(Olympic_v2)[4:7]
  #### Remove the last row that has all False's
vec <- olympic.leapCP$which
  ### Name the columns in the grid
names(vec) <- paste("X", 1:4, sep="")
  #### Build matrix of formula for every row
```

```r
allModelsList <- apply(vec, 1, function(x) as.formula(
  paste(c("count ~ 1", xList[x]), collapse = "+")))
  ### Calculate the coefficients for all 16 models
allModelsResults <- lapply(allModelsList,
                           function(x) lm(x, data=Olympic_v2))

#PRESS (Non-Mac)
library(qpcR)
olympic.lm = PRESS(lm(log_count~log_pop+log_gdp_per_cap+host+comm_soviet, data = Olympic_v2))
olympic.lmX1 = PRESS(lm(log_count~log_pop, data = Olympic_v2))
olympic.lmX2 = PRESS(lm(log_count~log_gdp_per_cap, data = Olympic_v2))
olympic.lmX3 = PRESS(lm(log_count~host, data = Olympic_v2))
olympic.lmX4 = PRESS(lm(log_count~comm_soviet, data = Olympic_v2))
olympic.lmX1X2 = PRESS(lm(log_count~log_pop+log_gdp_per_cap, data = Olympic_v2))
olympic.lmX1X3 = PRESS(lm(log_count~log_gdp_per_cap+host, data = Olympic_v2))
olympic.lmX1X4 = PRESS(lm(log_count~log_pop+comm_soviet, data = Olympic_v2))
olympic.lmX2X3 = PRESS(lm(log_count~log_gdp_per_cap+host, data = Olympic_v2))
olympic.lmX2X4 = PRESS(lm(log_count~log_gdp_per_cap+comm_soviet, data = Olympic_v2))
olympic.lmX3X4 = PRESS(lm(log_count~host+comm_soviet, data = Olympic_v2))
olympic.lmX1X2X3 = PRESS(lm(log_count~log_pop+log_gdp_per_cap+host, data = Olympic_v2))
olympic.lmX1X2X4 = PRESS(lm(log_count~log_pop+log_gdp_per_cap+comm_soviet, data = Olympic_v2))
olympic.lmX2X3X4 = PRESS(lm(log_count~log_gdp_per_cap+host+comm_soviet, data = Olympic_v2))
olympic.lmX1X3X4 = PRESS(lm(log_count~log_pop+host+comm_soviet, data = Olympic_v2))

olympic.lm_press <- rbind(olympic.lmX1$stat,
                          olympic.lmX3$stat,
                          olympic.lmX2$stat,
                          olympic.lmX4$stat,
                          olympic.lmX1X3$stat,
                          olympic.lmX1X2$stat,
                          olympic.lmX1X4$stat,
                          olympic.lmX2X3$stat,
                          olympic.lmX3X4$stat,
                          olympic.lmX2X4$stat,
                          olympic.lmX1X2X3$stat,
                          olympic.lmX1X2X4$stat,
                          olympic.lmX1X3X4$stat,
                          olympic.lmX2X3X4$stat,
                          olympic.lm$stat)

#Summary
Diagnostics = cbind(olympic.leapCP$which, Cp=round(olympic.leapCP$Cp,2), aR2=round(olympic.leapR2a$adjr
      AIC=matrix(unlist(lapply(allModelsResults, function(x) round(extractAIC(x),2))), ncol=2, byrow=TRU
#PRESS wasn't showing as column name
colnames(Diagnostics) = c("1","2","3","4","Cp","aR2","AIC","PRESS")
Diagnostics
olympic.lmfinal <- lm(log_count ~ log_pop + log_gdp_per_cap + host + comm_soviet, data = Olympic_v2)
summary(olympic.lmfinal)
par(mfrow=c(2,2))
plot(olympic.lmfinal)
library(lmtest)

bptest(log_count ~ log_pop + log_gdp_per_cap + host + comm_soviet,data = Olympic_v2,studentize = FALSE)
```

```r
library(nortest)
lillie.test(olympic.lmfinal$residuals)
#Influential cases
olympic.lm_inf=influence.measures(olympic.lmfinal)$is.inf
idx=which(apply(olympic.lm_inf,1,any))

Olympic[idx,]
olympic.lm_inf[idx,]
Olympic.pois<-glm(count~log_pop + log_gdp_per_cap + host + comm_soviet, data = Olympic_v2, family = pois
P__disp(Olympic.pois)
library(MASS)
olympic.nb <- glm.nb(count~log_pop + log_gdp_per_cap + host + comm_soviet, data = Olympic_v2)
library(leaps)
olympic.nb_leap <- leaps(y=Olympic_v2$count, x=Olympic_v2[,4:7])
Cp.nb<-round(olympic.nb_leap$Cp, 2)

which<-olympic.nb_leap$which
rownames(which) <-NULL
colnames(which)<-c('Pop', 'GDP/C', 'Host', 'Soviet')
xList <- names(Olympic_v2)[4:7]
vec <- olympic.nb_leap$which

#Name the columns in the grid
names(vec) <- paste("X", 1:4, sep="")

#Build matrix of formula for every row
allModelsList <- apply(vec, 1, function(x) as.formula(
  paste(c("count ~ 1", xList[x]), collapse = "+")))

#Calculate the coefficients for all 16 models
allModelsResults <- lapply(allModelsList,
                           function(x) glm.nb(x, data=Olympic_v2))


AIC.nb<-matrix(unlist(lapply(allModelsResults, function(x) round(extractAIC(x),2))), ncol = 2, byrow = T
library(knitr)
kable(cbind(which, Parameters = olympic.nb_leap$size, Cp.nb, AIC.nb), caption = 'Model Selection Diagnos
par(mfrow=c(1,2))
#Histogram Population
hist(base$pop,main = "Untransformed",xlab = "Population")
#Histogram log(Population)
hist(log(base$pop),main = "Transformed",xlab = "log(Population)")
tab<-summary(olympic.nb)$coefficients
colnames(tab)[4]<-'p-value'
rownames(tab)<-c('Intercept', 'log(Population)', 'log(GDP/Capita)', 'Host', 'Comm/Soviet')
kable(tab, format.args = list(justify = 'centre'), caption = 'Negative Binomial Model Coefficients')
```