olympics_prediction

Introduction

The Olympic games is international sporting events featuring winter and summer sporting competitions for men and women. The Olympics games is arguably the most prestigious sporting event in the world and its popularity is on the increase. In the 2016 edition held in Rio-Brazil, there were 306 events compared to the very first Olympic games in 1896 held in Athens-Greece which only had 43 events. The growth of the Olympics has been a trend from the early years of the event. In the 1950s, after the second world war and the cusp of the cold war the number of events had tripled to 150 from the very first Olympics. The growth continued till present time although it slowed down a bit. There might be an increase in the number of unique events over the next few years, but it would most likely be minute. The focus of our study is to build on our previous work and create a model that will be able to predict the proportion of medals won at future Olympic Games. Our SMART question is: What proportion of medals will Japan (the next Olympic host country) win during the 2020 Summer Olympic games. Previously we were able to answer the question: Does the host country have an advantage in the olympics? The answer was yes, that the host team preforms better than their performances when not a host team. Drawing on previous Olympic model building research we will be taking the following variable into consideration: GDP per Capita, Population, historical medal counts, planned economy (communist economy), soviet (or was once a soviet) nation, host country (is or isn't). The previous research that we referenced used the variables that were listed ans used the log of GDP, GDP per Capita, and Population, so we followed suit.

Exploratory Data Analysis

Olympic Data Set

We first started our EDA process by importing the Olympic data from: https://www.kaggle.com/heesoo37/120-years-of-olympic-history-athletes-and-results.

The first thing we did was add a Host_Country Column.

We then noticed that each athlete was awarded a medal including each team member, we were able to subset the data so that each team was awarded one medal so that there was no discrepency. We changed the structure of some of the columns such as medal, changed to ordered factor. We dropped Age, Height, Weight, City and Games.

Next we insert host country data into the data frame. Change some country codes to simplify analysis and select only the Summer Olympics data.

Next we created a seperate dataframe with only athletes that have won a medal.

We also created an additional data frame with year and total medals won. We Examined the structure of the data frame and then examined the five statistic summary.

```r
#Find the total medal count per year
total_medals <- winners_only %>% group_by(Year) %>% summarize(Medal_Count = n())
total_medals
```

```
## # A tibble: 28 x 2
##     Year Medal_Count
##    <int>       <int>
##  1  1896         143
##  2  1900         604
## 3  1904         486
## 4  1908         831
## 5  1912         941
## 6  1920        1308
## 7  1924         832
## 8  1928         734
## 9  1932         647
## 10 1936         917
## # ... with 18 more rows
```

```
## 'data.frame':    220819 obs. of  11 variables:
##  $ Year    : int  1896 1896 1896 1896 1896 1896 1896 1896 1896 1896 ...
##  $ ID      : int  41160 74612 105467 35741 35741 85456 89416 28169 56757 56757 ...
##  $ Name    : chr  "Dimitrios P. Golemis" "Julius Carl Fritz Manteuffel" "Sanidis" "Gustav Felix Flat
##  $ Sex     : chr  "M" "M" "M" "M" ...
##  $ Team    : chr  "Greece" "Germany" "Greece" "Germany" ...
##  $ NOC     : chr  "GRE" "GER" "GRE" "GER" ...
##  $ Season  : chr  "Summer" "Summer" "Summer" "Summer" ...
##  $ Sport   : chr  "Athletics" "Gymnastics" "Shooting" "Gymnastics" ...
##  $ Event   : chr  "Athletics Men's 800 metres" "Gymnastics Men's Parallel Bars, Teams" "Shooting Men
##  $ Medal   : Ord.factor w/ 3 levels "Bronze"<"Silver"<..: 1 3 NA NA NA NA NA NA NA NA ...
##  $ Host_NOC: chr  "GRE" "GRE" "GRE" "GRE" ...
```

```
##       Year            ID              Name               Sex           
##  Min.   :1896   Min.   :     1   Length:220819      Length:220819     
##  1st Qu.:1960   1st Qu.: 33988   Class :character   Class :character  
##  Median :1984   Median : 68266   Mode  :character   Mode  :character  
##  Mean   :1977   Mean   : 67978                                        
##  3rd Qu.:2000   3rd Qu.:101862                                        
##  Max.   :2016   Max.   :135568                                        
##      Team               NOC               Season             Sport          
##  Length:220819      Length:220819      Length:220819      Length:220819     
##  Class :character   Class :character   Class :character   Class :character  
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character  
##                                                                             
##                                                                             
##                                                                             
##     Event              Medal          Host_NOC        
##  Length:220819      Bronze: 11264   Length:220819     
##  Class :character   Silver: 11064   Class :character  
##  Mode  :character   Gold  : 11302   Mode  :character  
##                     NA's  :187189                     
##                                                       
##                                                       
```

Country Statistics Dataset

We then imported the GDP, Population, and per capita GDP data from: https://www.rug.nl/ggdc/historical development/maddison/releases/maddison-project-database-2018

About dataset

The Maddison Project Database provides information on comparative economic growth and income levels over the very long run. The 2018 version of this database covers 169 countries and the period up to 2016.

We used the Real GDP per Capita in 2011 USD and the Population in thousands.

The first step we took to prepare this data for analysis was to change the column names to match the athlete columns. We also selected to keep data later than the year 1896. We also multiplied Population by 1000 because it was in 1000's and we wanted the actual Population of each country.

The next step we took was to find the whole world's GDP by year, by using that we were able to calculate the share of GDP per country per year. We also examined the structure and summary of this dataset.

```
## tibble [14,617 x 7] (S3: tbl_df/tbl/data.frame)
##  $ NOC       : chr [1:14617] "AFG" "AFG" "AFG" "AFG" ...
##  $ Country   : chr [1:14617] "Afghanistan" "Afghanistan" "Afghanistan" "Afghanistan" ...
##  $ Year      : num [1:14617] 1913 1950 1951 1952 1953 ...
##  $ GDPPC     : num [1:14617] NA 2392 2422 2462 2568 ...
##  $ Population: num [1:14617] 5730000 8150000 8284000 8425000 8573000 ...
##  $ GDP       : num [1:14617] NA 1.95e+10 2.01e+10 2.07e+10 2.20e+10 ...
##  $ GDPShare  : num [1:14617] NA 0.00236 0.00229 0.00225 0.00226 ...
```

```
##      NOC              Country               Year          GDPPC
##  Length:14617       Length:14617       Min.   :1896   Min.   :   134
##  Class :character   Class :character   1st Qu.:1952   1st Qu.:  1755
##  Mode  :character   Mode  :character   Median :1973   Median :  4018
##                                        Mean   :1970   Mean   :  8366
##                                        3rd Qu.:1995   3rd Qu.:  9914
##                                        Max.   :2016   Max.   :220717
##                                                       NA's   :1346
##    Population            GDP               GDPShare
##  Min.   :1.100e+04   Min.   :6.378e+07   Min.   :0.0000
##  1st Qu.:2.293e+06   1st Qu.:7.630e+09   1st Qu.:0.0003
##  Median :6.110e+06   Median :2.697e+10   Median :0.0013
##  Mean   :2.929e+07   Mean   :2.408e+11   Mean   :0.0092
##  3rd Qu.:1.790e+07   3rd Qu.:1.210e+11   3rd Qu.:0.0048
##  Max.   :1.373e+09   Max.   :1.721e+13   Max.   :0.4357
##  NA's   :114         NA's   :1458        NA's   :1458
```

Model Building Preparation

To prepare for the model building process we first created a dataframe called medal_counts, this consisted of the medal counts, the country code. We also added a column that stated whether the country is the host country or not.

```
## `summarise()` has grouped output by 'Year', 'Host_NOC'. You can override using the `.groups` argumen
```

We specified that it would be countries that have earned five or more medals in the year 2012. The time range we selected is from 1988 to 2016.

Next we created a column that stated is the country has a planned (communist) economy, and another that states if a country is or has been a part of the Soviet Union.

We dropped the following two country codes: "EUN", "SCG".

Our final data frame that we used is a merge of the medal counts data frame and the GDP and Population data.

Finally we created a column with the proportion of medals. We added another column that states the medal proportions for each country from previous Olympic games.

Model Building

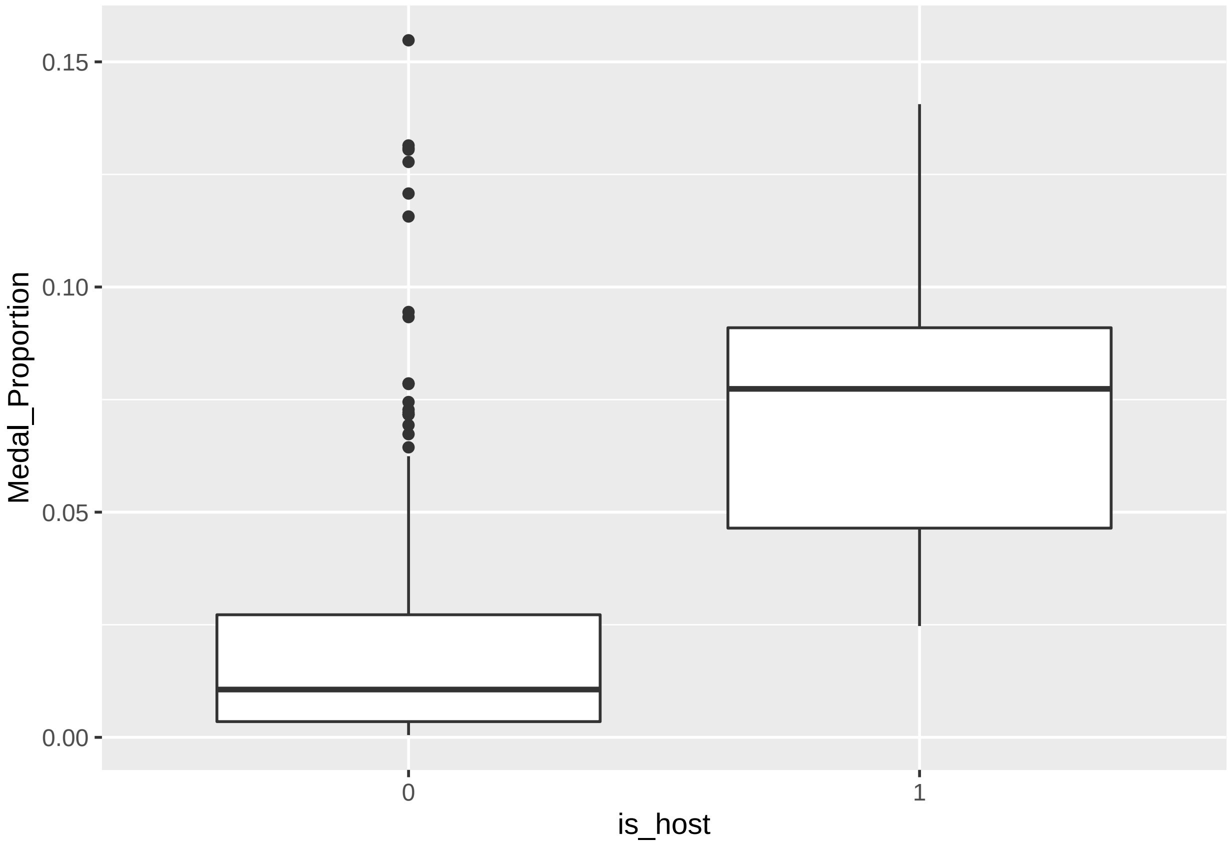
Model Development

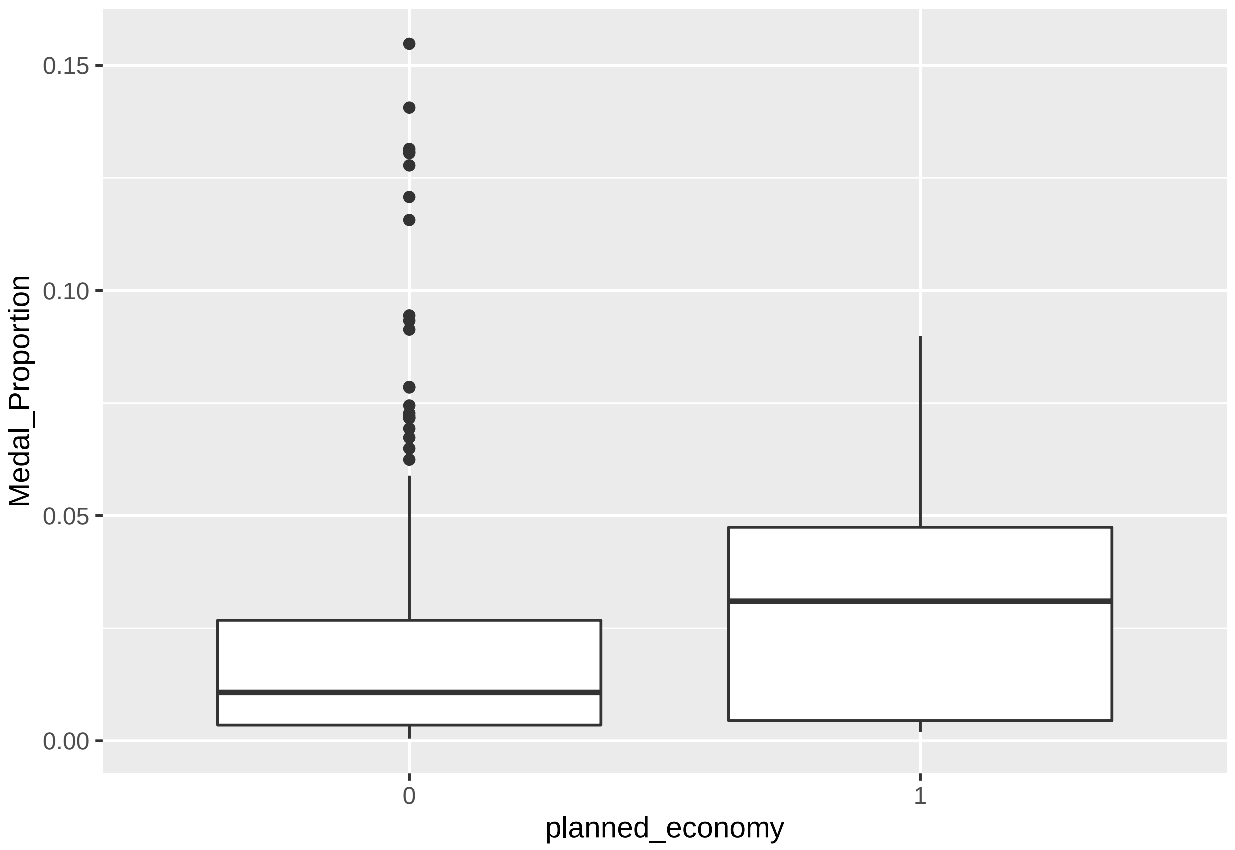
To develop our model we began by dividing the data into training and test sets. The training data was from 1988 to 2012 and the test data was the year 2016.

Plots

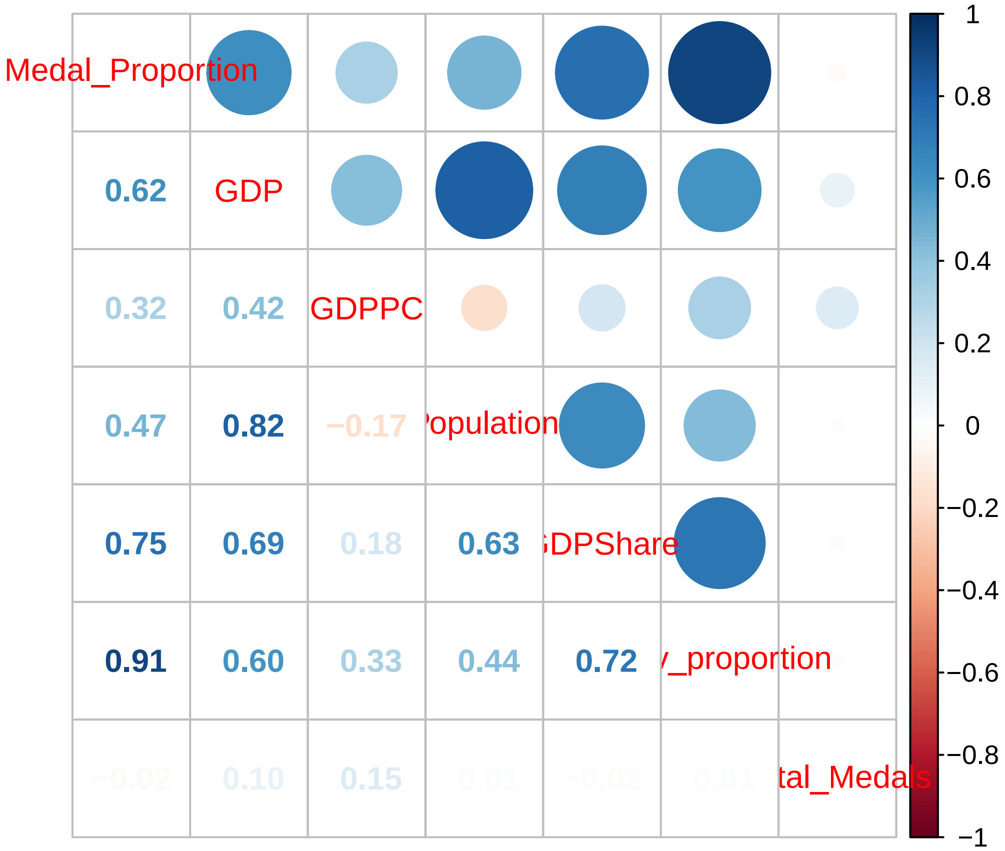
The variables we wanted to take into consideration where: is_host, planned_economy, is_soviet, log(GDPPC), log(Population), log(GDP), GDPShare, pre_proportion, Total_Medals.

We began by looking at box plots of is_host and planned_economy to makes sure the distributions were different.

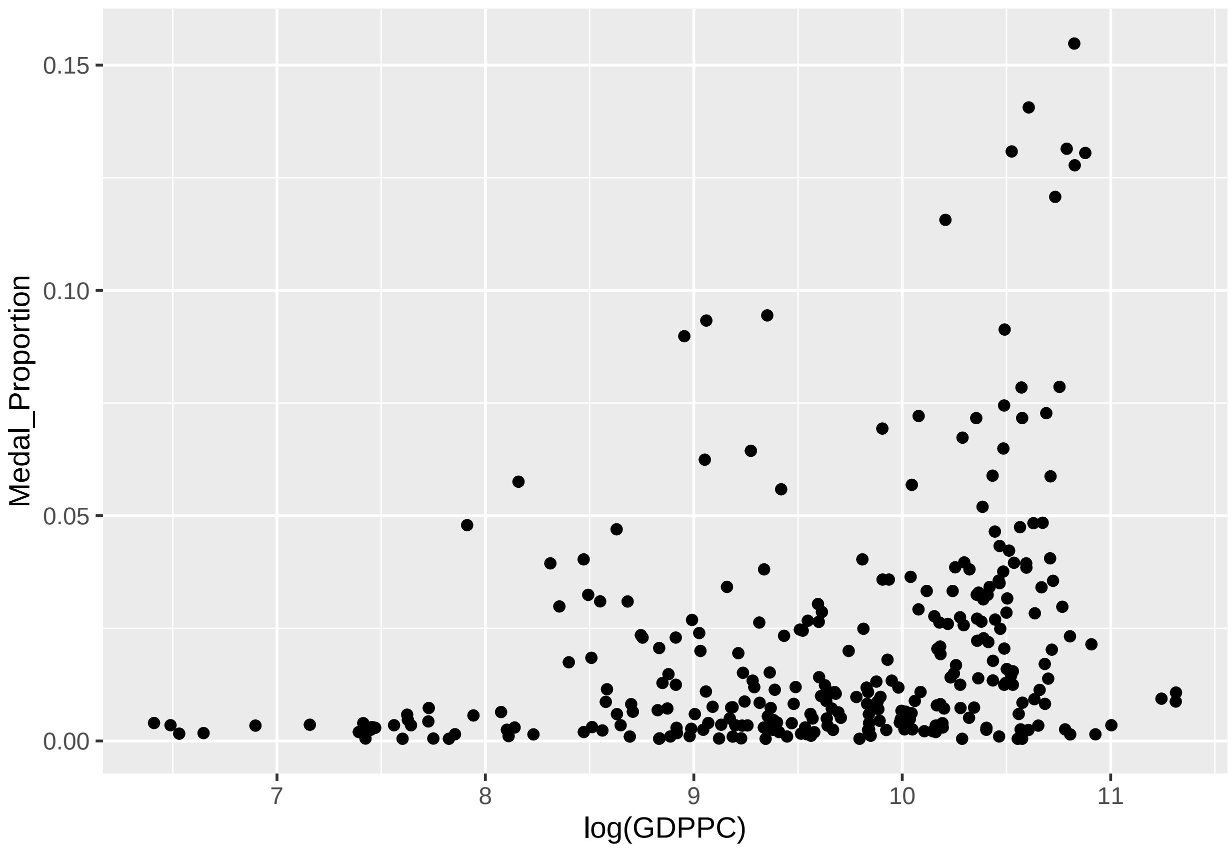




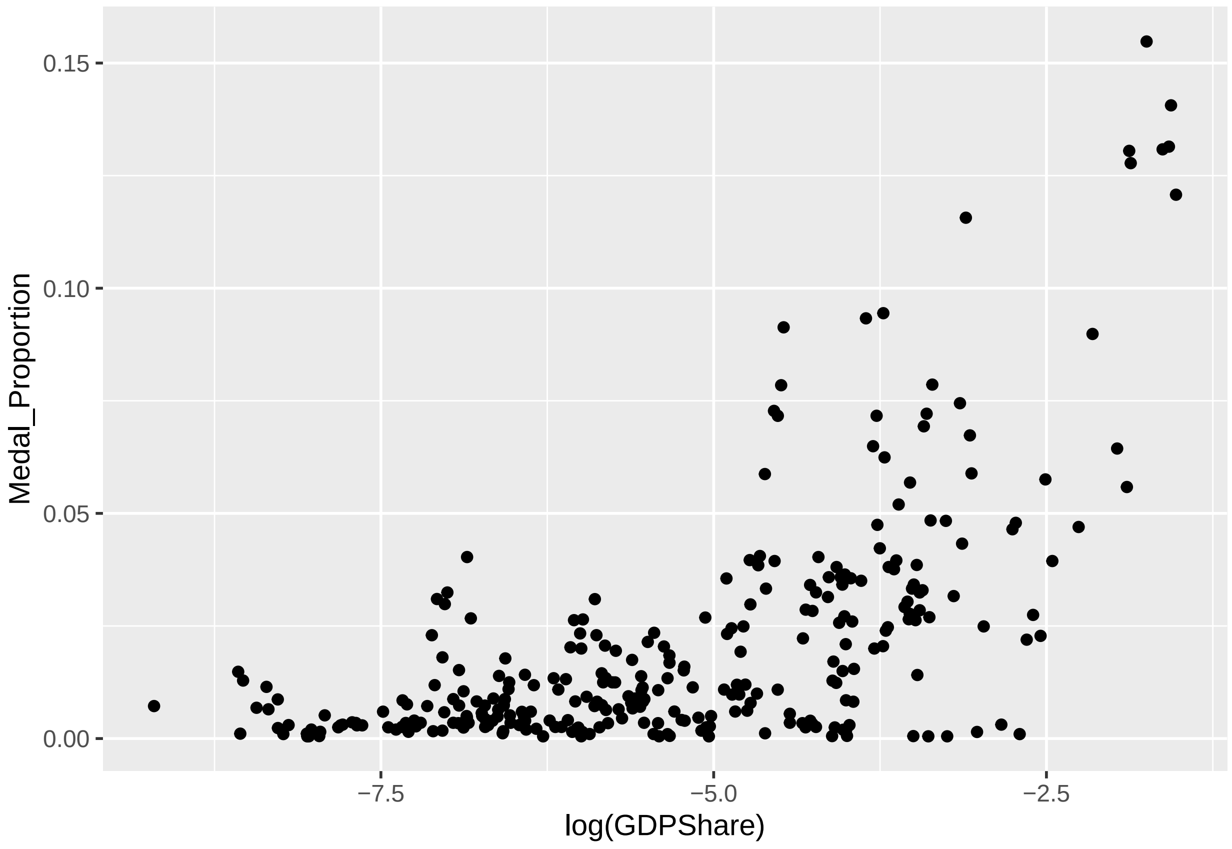
After examining the box plots, we can see that means appear visually different between both is_host and planned_economy variables. Next we created a correlation plot between the numerical variables. By doing so we can see that the variables that correlate with Medal_Proportion are: log(GDP), log(GDPPC), log(Population), GDPShare, prev_proportion.



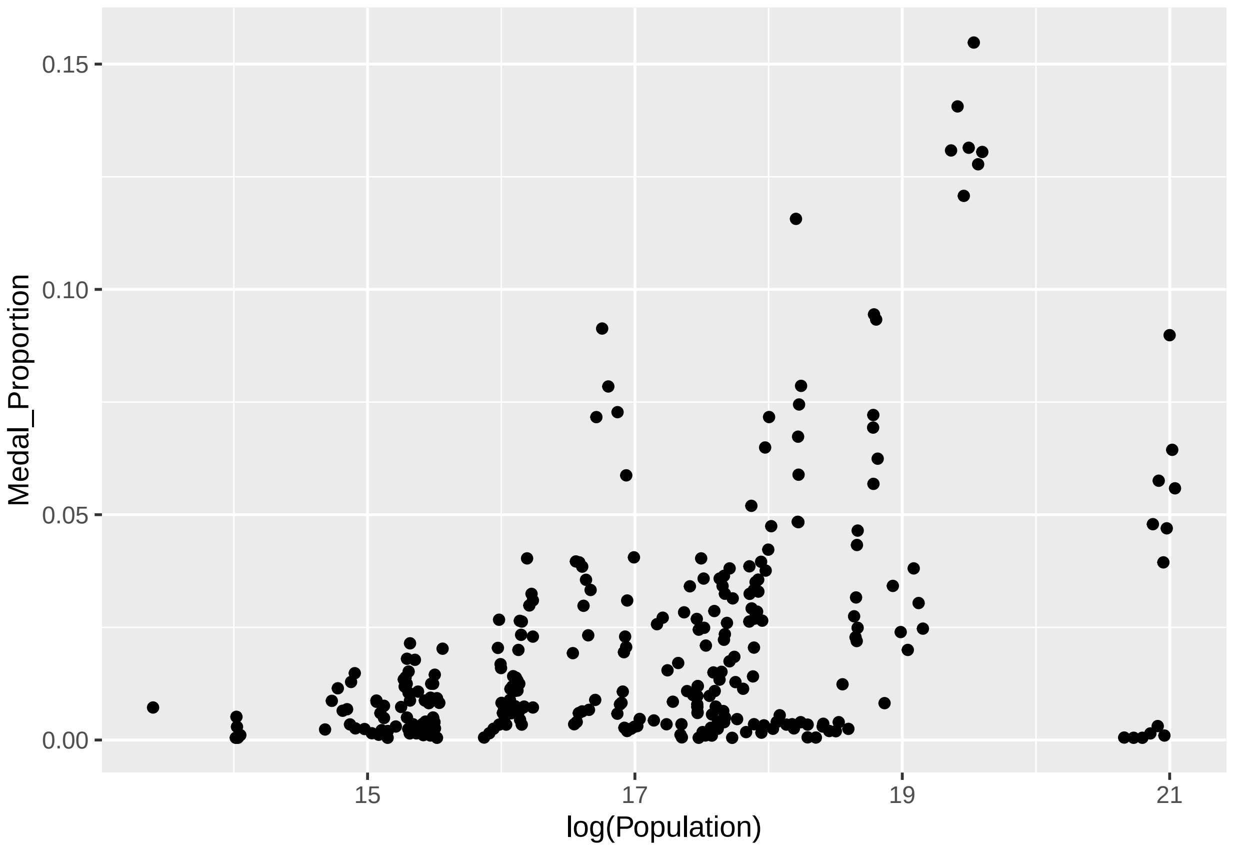
To create a clearer picture we created scatter plots between log(GDPPC) and Medal_Proportion and another between log(GDP) and Medal_Proportion. We created a third scatter plot between log(Population) and Medal_proportion. We can see there is a some correlation that is positively increasing.



Medal_Proportion
0.15
0.10
0.05
0.00
7
8
9
10
11
log(GDPPC)

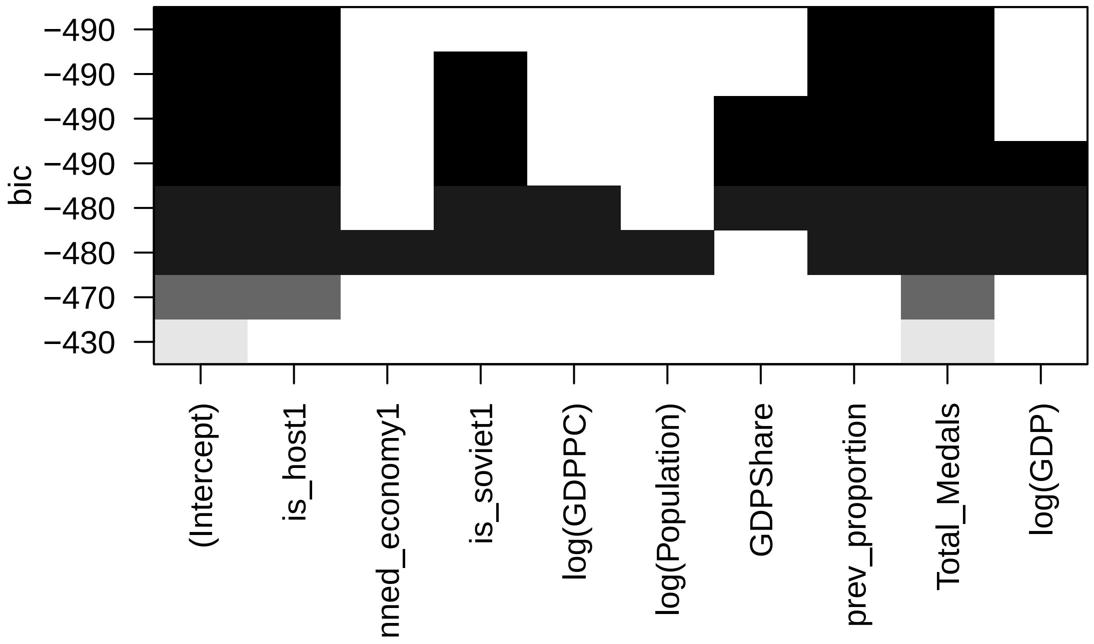


Medal_Proportion
0.15
0.10
0.05
0.00
−7.5
−5.0
−2.5
log(GDPShare)



We can see there is a some correlation that is positively increasing.

The next step we took in selecting our models variables was feature selection. This showed us that the best variables are prev_proportion, GDPShare, is_host, and Total_Medals. After some trial and error we found that Total_Medals is not a good variable to use in our model. Below you can see the BIC plot that we used for the medal selection.



The Model

Multiple Linear Regression model

The variables that we found created the best model (as stated earlier) are prev_proportion, GDPShare, and is_host.

We incorporated these variables into the linear model and trained the model, using the training dataset. The model can be seen below.

```
#Prediction model
medal_model <- lm(Medal_Proportion ~ prev_proportion + GDPShare + is_host, data = training_data)
summary(medal_model)
```

```
##
## Call:
## lm(formula = Medal_Proportion ~ prev_proportion + GDPShare +
##     is_host, data = training_data)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.025255 -0.004826 -0.002236  0.003704  0.054830
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     0.0038527  0.0007591   5.076 7.53e-07 ***
## prev_proportion 0.6955991  0.0311435  22.335  < 2e-16 ***
## GDPShare        0.1541476  0.0268227   5.747 2.63e-08 ***
## is_host1        0.0293705  0.0046179   6.360 9.47e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.009922 on 251 degrees of freedom
## Multiple R-squared:  0.8677, Adjusted R-squared:  0.8661
## F-statistic: 548.5 on 3 and 251 DF,  p-value: < 2.2e-16
```

The VIF values for these variables are all around 2 and 1, indicating that there is not much multicollinearity between these variables.

Variance inflation factor (VIF) is a measure of the amount of multicollinearity in a set of multiple regression variables.

```
vif(medal_model)
```

```
## prev_proportion        GDPShare        is_host1
##        2.079466        2.129171        1.061938
```

The small p-values that these variables have indicate that they are statistically significant in our model. The lower the p-value, the greater the statistical significance of the observed difference.

Our adjusted R squared is 86.61 percent, this accounts for much of the variance in the data

Ridge regression

```
#define response variable
y <- training_data$Medal_Proportion
```

```r
#define matrix of predictor variables
x <-data.matrix(training_data[, c('prev_proportion', 'GDPShare', 'is_host')])
```

we'll use the glmnet() function to fit the ridge regression model and specify alpha=0. Alpha equal to 1 is equivalent to using Lasso Regression and setting alpha to some value between 0 and 1 is equivalent to using an elastic net.

glmnet() automatically performs this standardization for you. If we happened to already standardize the variables, we can specify standardize=False.

```r
library(glmnet)
```

```
## Loading required package: Matrix

## Loaded glmnet 4.1-3
```

```r
#fit ridge regression model
model <- glmnet(x, y, alpha = 0)

#view summary of model
summary(model)
```

```
##           Length Class     Mode   
## a0        100    -none-    numeric
## beta      300    dgCMatrix S4     
## df        100    -none-    numeric
## dim         2    -none-    numeric
## lambda    100    -none-    numeric
## dev.ratio 100    -none-    numeric
## nulldev     1    -none-    numeric
## npasses     1    -none-    numeric
## jerr        1    -none-    numeric
## offset      1    -none-    logical
## call        4    -none-    call   
## nobs        1    -none-    numeric
```

Choosing the optimal value for lambda

Identify the lambda value that produces the lowest test mean squared error (MSE) by using k-fold cross-validation.

glmnet has the function cv.glmnet() that automatically performs k-fold cross validation using k = 10 folds.

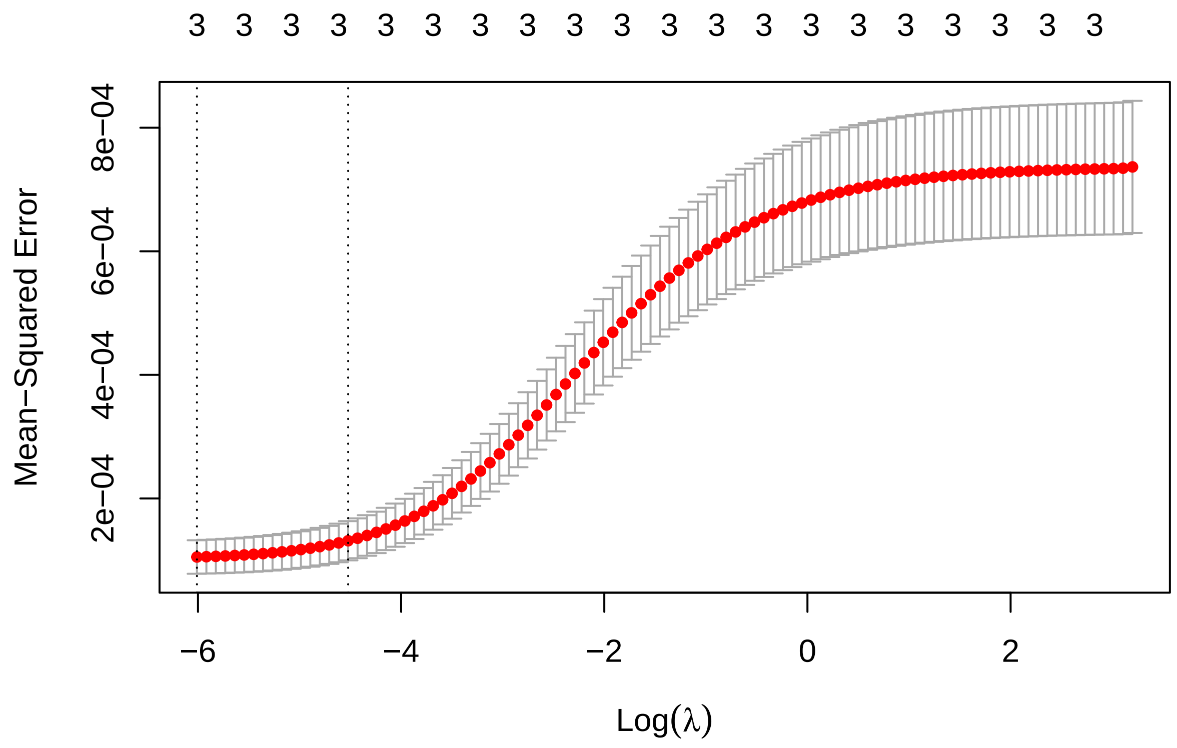
```r
#perform k-fold cross-validation to find optimal lambda value
cv_model <- cv.glmnet(x, y, alpha = 0)

#find optimal lambda value that minimizes test MSE
best_lambda <- cv_model$lambda.min
best_lambda
```

```
## [1] 0.002453023
```

The lambda value that minimizes the test MSE turns out to be 0.002453023.

```r
#produce plot of test MSE by lambda value
plot(cv_model)
```



Final Ridge Regression model with optimal lambda value

```
#find coefficients of best model
best_model <- glmnet(x, y, alpha = 0, lambda = best_lambda)
coef(best_model)
```

```
## 4 x 1 sparse Matrix of class "dgCMatrix"
##                           s0
## (Intercept)     -0.02305580
## prev_proportion  0.61105302
## GDPShare         0.19008317
## is_host          0.02802123
```

Calculate the R squared value of ridge regrression model on training data.

```
#use fitted best model to make predictions
y_predicted <- predict(model, s = best_lambda, newx = x)

#find SST and SSE
sst <- sum((y - mean(y))^2)
sse <- sum((y_predicted - y)^2)

#find R-Squared
rsq <- 1 - sse/sst
rsq
```

```
## [1] 0.8632302
```

R square is 0.8632302. The best model was able to explain 86.32 % of variation in the response value of the training data.

The multiple linear regression model gives better results so we are going to use that for further work.

Reliability of results

As seen above in the model summary, we can see that the adjusted R squared is 0.8661, indicating a good model.

We can see below that we made a prediction for Japan in the 2020 Olympic Games, which is in Tokyo. The values that we used were GDPShare and the prev_proportion. The GDPShare was a predicted value that we obtained from tradingeconomics.com. Our model predicted that Japan will win 6.37% of the Proportion of Medals.

```
## 0.06374145
```

Next we wanted to see the reliability of our results, we tested out model on the USA in the year 2016. Our model predicted that the US would have 11.6% of the Medals (235). The actual was 13.0% (264 medals.)

```
## 0.1162911
```

Below we can see the mean error in the prediction, which is .12 (2.5 medals) and the standard deviation is 1.08 (21.81 medals). This indicated that the model error could essentially have been 0 because it falls in the 95% confidence interval.

```
## [1] 0.1236875
```

```
## [1] 1.078295
```

Predictions we can make

By using this model we can predict the proportion of medals that each country will win in future Summer Olympic games. Our model is already quite good, able to explain 86.61% of the variance.

But we could improve the model accuracy by somehow including the athletes' information for each country, since some athletes perform much better that others.

Conclusion

In conclusion the best variables we found to use when predicting Olympic Medal Proportions are: prev_proportion, GDPShare, is_host. The process we used was to clean the Olympic dataset and then we cleaned the GDP and Population dataset. Next we looked at the boxplots, correlation plots and scatter plots to determine which variables were best to use. The next step we took was feature selection and we used a BIC plot to determine the best plots. By using all these methods we were able to create our model.We created a multiple linear regression model and ridge regression model.The Multiple Linear Regression gives better parameters.By looking at the adjusted R squared of 86.62% and the VIF values we were able to determine that our model is a good model.