LifeExpectancy

Douglas Melis

10/18/2020

## Data Preprocessing

Not all of the data sets I used have the standardized country code, so I had to use country names to match up the records. Unfortuantely, a number of countries have multiple names or slight variations in spellings or format. Therefore, the first step was to create a function to clean the country names column in each dataset. The function I created runs through a set of name variations and replaces each one with the variation I selected for the final dataset.

clean\_country\_names <- function(data){  
 data[data == "Bolivia (Plurinational State of)"] <- "Bolivia"  
 data[data == "Democratic People's Republic of Korea"] <- "North Korea"  
 data[data == "Cote d'Ivoire"] <- "Côte d’Ivoire"  
 data[data == "CÃ´te d'Ivoire"] <- "Côte d’Ivoire"  
 data[data == "Czech Republic"] <- "Czechia"  
 data[data == "Egypt, Arab Rep."] <- "Egypt"  
 data[data == "Swaziland"] <- "Eswatini"  
 data[data == "Gambia, The"] <- "Gambia"  
 data[data == "Guinea Bissau"] <- "Guinea-Bissau"  
 data[data == "Iran (Islamic Republic of)"] <- "Iran"  
 data[data == "Iran, Islamic Rep."] <- "Iran"  
 data[data == "Kyrgyz Republic"] <- "Kyrgyzstan"  
 data[data == "Lao People's Democratic Republic"] <- "Laos"  
 data[data == "Lao PDR"] <- "Laos"  
 data[data == "Micronesia, Fed. Sts."] <- "Micronesia"  
 data[data == "Republic of Korea"] <- "South Korea"  
 data[data == "Korea, Rep."] <- "South Korea"  
 data[data == "Republic of Moldova"] <- "Moldova"  
 data[data == "Macedonia (TFYR)"] <- "North Macedonia"  
 data[data == "The former Yugoslav Republic of Macedonia"] <- "North Macedonia"  
 data[data == "Micronesia (Federated States of)"] <- "Micronesia"  
 data[data == "Micronesia (country)"] <- "Micronesia"  
 data[data == "Russian Federation"] <- "Russia"  
 data[data == "St. Lucia"] <- "Saint Lucia"  
 data[data == "St. Vincent and the Grenadines"] <- "Saint Vincent and the Grenadines"  
 data[data == "Slovak Republic"] <- "Slovakia"  
 data[data == "Sudan (until 2011)"] <- "Sudan"  
 data[data == "Syrian Arab Republic"] <- "Syria"  
 data[data == "Timor Leste"] <- "Timor-Leste"  
 data[data == "United Kingdom of Great Britain and Northern Ireland"] <- "United Kingdom"  
 data[data == "United States of America"] <- "United States"  
 data[data == "Tanzania"] <- "United Republic of Tanzania"  
 data[data == "Venezuela (Bolivarian Republic of)"] <- "Venezuela"  
 data[data == "Venezuela, RB"] <- "Venezuela"  
 data[data == "Viet Nam"] <- "Vietnam"  
 data[data == "Yemen, Rep."] <- "Yemen"  
 data[data == "Brunei Darussalam"] <- "Brunei"  
 data[data == "Cape Verde"] <- "Cabo Verde"  
 data[data == "Congo, Rep."] <- "Congo"  
 data[data == "Congo, Dem. Rep."] <- "Democratic Republic of Congo"  
 data[data == "Democratic Republic of the Congo"] <- "Democratic Republic of Congo"  
 data[data == "DR Congo"] <- "Democratic Republic of Congo"  
 data[data == "Macedonia"] <- "North Macedonia"  
 data[data == "Timor"] <- "Timor-Leste"  
 data[data == "Bahamas, The"] <- "Bahamas"  
 data[data == "Korea, Dem. People’s Rep."] <- "North Korea"  
  
 return(data)  
}

Next I read in the data for the life expectancy variable, and removed the unnecessary columns. This data frame will act as the base that I joined to the data frames for the other variables.

life <- read.csv("data/life expectancy.csv", header = T, stringsAsFactors = F)  
  
data <- life[,c(1,2,5)]  
data <- rename(data, Country = Location,  
 Year = Period,  
 Life.Expectancy = First.Tooltip)  
data <- clean\_country\_names(data)  
  
print(head(data))

## Country Year Life.Expectancy  
## 1 Afghanistan 2016 62.69  
## 2 Afghanistan 2015 63.18  
## 3 Afghanistan 2014 62.97  
## 4 Afghanistan 2013 62.74  
## 5 Afghanistan 2012 62.21  
## 6 Afghanistan 2011 61.72

Both BMI and obesity are included in the same dataset; however, they were split up by sex, which scatters each record across two rows. In order to fix that, I had to spread the data. I filtered all of the male records into one table and all of the female ones into another. I renamed the columns and joined them into a single table where each record represented one year for one country, with columns for mean male BMI, mean female BMI, the prevalence of obesity in men, and prevalence of obesity in women. Then I was able to take the average of both BMI columns to create the mean BMI for the total population, and then take the average of both obesity prevalence columns to create the mean obesity prevalence for the total population. This method assumes that the countries have equal male and female populations for each year.

bmi <- read.csv("data/bmi.csv", header = T, stringsAsFactors = F)  
bmi <- bmi[,c(1,3,4,5,8)]  
names(bmi)[5] = "Obesity"  
names(bmi)[1] = "Country"  
  
bmi.men <- filter(bmi, Sex == 'Men')  
bmi.men <- rename(bmi.men, BMI.M = Mean.BMI,  
 Obesity.M = Obesity)  
bmi.men <- bmi.men[-2]  
  
bmi.women <- filter(bmi, Sex == 'Women')  
bmi.women <- rename(bmi.women, BMI.W = Mean.BMI,  
 Obesity.W = Obesity)  
bmi.women <- bmi.women[-2]  
  
bmi.both <- left\_join(bmi.men, bmi.women, by = c("Country", "Year"))  
  
bmi.both[,"BMI"] <- rowMeans(bmi.both[,c('BMI.M', 'BMI.W')], na.rm=TRUE)  
bmi.both[,"Obesity"] <-rowMeans(bmi.both[,c('Obesity.M', 'Obesity.W')], na.rm=TRUE)  
  
bmi.both <- bmi.both[,c(1,2,7,8)]  
  
bmi.both <- clean\_country\_names(bmi.both)  
data <- left\_join(data, bmi.both, by = c("Country", "Year"))  
print(head(data))

## Country Year Life.Expectancy BMI Obesity  
## 1 Afghanistan 2016 62.69 23.45830 0.05681314  
## 2 Afghanistan 2015 63.18 23.35209 0.05396362  
## 3 Afghanistan 2014 62.97 23.24576 0.05124183  
## 4 Afghanistan 2013 62.74 23.13956 0.04864080  
## 5 Afghanistan 2012 62.21 23.03342 0.04615575  
## 6 Afghanistan 2011 61.72 22.92832 0.04378424

The alcohol dataset was split into two files to be joined, the first having columns for the years 2000-2009, and the second having columns for the years 2010-2018. After being combined, they were still in the wrong format, as the columns were not different variables, just different values of the same variable: year. In order to clean that, I had to gather the columns into a new pair of variables, putting all of the years into one variable and the alcohol consumption levels into a second variable.

alcohol.00 <- read.csv("data/alcohol 00-09.csv", header = T, skip = 1, check.names = F, stringsAsFactors = F)  
alcohol.00 <- alcohol.00[,-c(2,3)]  
  
alcohol.00 <- gather(alcohol.00, `2009`, `2008`, `2007`, `2006`, `2005`, `2004`, `2003`, `2002`, `2001`, `2000`, key = "Year", value = "Alcohol")  
  
alcohol.10 <- read.csv("data/alcohol 10-18.csv", header = T, skip = 1, check.names = F, stringsAsFactors = F)  
alcohol.10 <- alcohol.10[,-c(2,3)]  
  
alcohol.10 <- gather(alcohol.10, `2018`, `2017`, `2016`, `2015`, `2014`, `2013`, `2012`, `2011`, `2010`, key = "Year", value = "Alcohol")  
  
alcohol <- rbind(alcohol.00, alcohol.10)  
  
alcohol[,"Year"] <- as.numeric(alcohol[,"Year"])  
  
alcohol <- clean\_country\_names(alcohol)  
data <- left\_join(data, alcohol, by = c("Country", "Year"))  
print(head(data))

## Country Year Life.Expectancy BMI Obesity Alcohol  
## 1 Afghanistan 2016 62.69 23.45830 0.05681314 0.02  
## 2 Afghanistan 2015 63.18 23.35209 0.05396362 0.00  
## 3 Afghanistan 2014 62.97 23.24576 0.05124183 0.01  
## 4 Afghanistan 2013 62.74 23.13956 0.04864080 0.01  
## 5 Afghanistan 2012 62.21 23.03342 0.04615575 0.01  
## 6 Afghanistan 2011 61.72 22.92832 0.04378424 0.01

The GDP dataset had the same structure as the alcohol consumption, so I used the same gather process, creating a year variable and a GDP variable.

gdp <- read.csv("data/GDP per capita.csv", header = T, check.names = F, stringsAsFactors = F)  
gdp <- gdp[,c('Country Name', '2018', '2017', '2016', '2015', '2014', '2013', '2012', '2011', '2010',  
 '2009', '2008', '2007', '2006', '2005', '2004', '2003', '2002', '2001', '2000')]  
  
gdp <- gather(gdp, `2018`, `2017`, `2016`, `2015`, `2014`, `2013`, `2012`, `2011`, `2010`, `2009`,  
 `2008`, `2007`, `2006`, `2005`, `2004`, `2003`, `2002`, `2001`, `2000`, key = "Year", value = "GDP")  
  
gdp[,"Year"] <- as.numeric(gdp[,"Year"])  
names(gdp)[1] <- "Country"  
  
gdp <- clean\_country\_names(gdp)  
data <- left\_join(data, gdp, by = c("Country", "Year"))  
print(head(data))

## Country Year Life.Expectancy BMI Obesity Alcohol GDP  
## 1 Afghanistan 2016 62.69 23.45830 0.05681314 0.02 547.2281  
## 2 Afghanistan 2015 63.18 23.35209 0.05396362 0.00 578.4664  
## 3 Afghanistan 2014 62.97 23.24576 0.05124183 0.01 613.8567  
## 4 Afghanistan 2013 62.74 23.13956 0.04864080 0.01 637.1655  
## 5 Afghanistan 2012 62.21 23.03342 0.04615575 0.01 641.8715  
## 6 Afghanistan 2011 61.72 22.92832 0.04378424 0.01 591.1628

The HIV/AIDS dataset was already in the required format, so I only had to remove the unnecessary columns.

hiv <- read.csv("data/deaths-and-new-cases-of-hiv.csv", header = T, stringsAsFactors = F)  
hiv <- hiv[,c(1,3,5,6)]  
names(hiv) <- c("Country", "Year", "HIV.Prevalence", "HIV.Incidence")  
  
hiv <- clean\_country\_names(hiv)  
data <- left\_join(data, hiv, by = c("Country", "Year"))  
print(head(data))

## Country Year Life.Expectancy BMI Obesity Alcohol GDP  
## 1 Afghanistan 2016 62.69 23.45830 0.05681314 0.02 547.2281  
## 2 Afghanistan 2015 63.18 23.35209 0.05396362 0.00 578.4664  
## 3 Afghanistan 2014 62.97 23.24576 0.05124183 0.01 613.8567  
## 4 Afghanistan 2013 62.74 23.13956 0.04864080 0.01 637.1655  
## 5 Afghanistan 2012 62.21 23.03342 0.04615575 0.01 641.8715  
## 6 Afghanistan 2011 61.72 22.92832 0.04378424 0.01 591.1628  
## HIV.Prevalence HIV.Incidence  
## 1 4133.235 790.0594  
## 2 3630.161 679.0092  
## 3 3200.788 584.7016  
## 4 2788.420 495.3635  
## 5 2429.128 418.4731  
## 6 2164.778 362.4356

The diabetes dataset had the same structure as the BMI and obesity dataset, so I used the same process to restructure the age-standardized diabetes prevalence data.

diabetes <- read.csv("data/diabetes.csv", header = T, stringsAsFactors = F)  
diabetes <- diabetes[,c(1,3,4,5)]  
names(diabetes)[4] = "Diabetes"  
names(diabetes)[1] = "Country"  
  
diabetes.men <- filter(diabetes, Sex == 'Men')  
diabetes.men <- rename(diabetes.men, Diabetes.M = Diabetes)  
diabetes.men <- diabetes.men[-2]  
  
diabetes.women <- filter(diabetes, Sex == 'Women')  
diabetes.women <- rename(diabetes.women, Diabetes.W = Diabetes)  
diabetes.women <- diabetes.women[-2]  
  
diabetes.both <- left\_join(diabetes.men, diabetes.women, by = c("Country", "Year"))  
  
diabetes.both[,"Diabetes"] <- rowMeans(diabetes.both[,c('Diabetes.M', 'Diabetes.W')], na.rm=TRUE)  
  
diabetes.both <- diabetes.both[,c(1,2,5)]  
  
diabetes.both <- clean\_country\_names(diabetes.both)  
data <- left\_join(data, diabetes.both, by = c("Country", "Year"))  
print(head(data))

## Country Year Life.Expectancy BMI Obesity Alcohol GDP  
## 1 Afghanistan 2016 62.69 23.45830 0.05681314 0.02 547.2281  
## 2 Afghanistan 2015 63.18 23.35209 0.05396362 0.00 578.4664  
## 3 Afghanistan 2014 62.97 23.24576 0.05124183 0.01 613.8567  
## 4 Afghanistan 2013 62.74 23.13956 0.04864080 0.01 637.1655  
## 5 Afghanistan 2012 62.21 23.03342 0.04615575 0.01 641.8715  
## 6 Afghanistan 2011 61.72 22.92832 0.04378424 0.01 591.1628  
## HIV.Prevalence HIV.Incidence Diabetes  
## 1 4133.235 790.0594 NA  
## 2 3630.161 679.0092 NA  
## 3 3200.788 584.7016 0.1190786  
## 4 2788.420 495.3635 0.1162422  
## 5 2429.128 418.4731 0.1133741  
## 6 2164.778 362.4356 0.1105226

For the immunizations, I anticipated that using all ten of them would result in serious multicollinearity problems, so I decided to compute an average immunization coverage value for all ten immunizations. In order to do so, I joined the datasets for each vaccine into one immunization table. Then I created a new column that was the mean of the values in all the other columns.

bcg <- read.csv("data/bcg.csv", header = T, stringsAsFactors = F)  
bcg <- bcg[,c(1,3,4)]  
names(bcg) <- c("Country", "Year", "BCG")  
  
bcg <- subset(bcg, Year >= 2000)  
  
immunization <- bcg  
  
  
add\_vaccine <- function(immunization, file.name, vaccine.name){  
 vaccine <- read.csv(file.name, header = T, stringsAsFactors = F)  
 vaccine <- vaccine[,c(1,3,4)]  
 names(vaccine) <- c("Country", "Year", vaccine.name)  
   
 vaccine <- subset(vaccine, Year >= 2000)  
  
 immunization <- full\_join(immunization, vaccine, by = c("Country", "Year"))  
 return(immunization)  
}  
  
immunization <- add\_vaccine(immunization, "data/dtp3.csv", "DTP3")  
immunization <- add\_vaccine(immunization, "data/hepb3.csv", "HepB3")  
immunization <- add\_vaccine(immunization, "data/hib3.csv", "Hib3")  
immunization <- add\_vaccine(immunization, "data/mcv1.csv", "MCV1")  
immunization <- add\_vaccine(immunization, "data/mcv2.csv", "MCV2")  
immunization <- add\_vaccine(immunization, "data/pab.csv", "PAB")  
immunization <- add\_vaccine(immunization, "data/pcv3.csv", "PCV3")  
immunization <- add\_vaccine(immunization, "data/pol3.csv", "Pol3")  
immunization <- add\_vaccine(immunization, "data/rotac.csv", "RotaC")  
  
print(head(immunization))

## Country Year BCG DTP3 HepB3 Hib3 MCV1 MCV2 PAB PCV3 Pol3 RotaC  
## 1 Afghanistan 2019 78 66 66 66 64 39 68 65 73 55  
## 2 Afghanistan 2018 78 66 66 66 64 39 68 65 73 45  
## 3 Afghanistan 2017 78 66 66 66 64 39 70 65 73 NA  
## 4 Afghanistan 2016 78 66 66 66 64 39 70 65 73 NA  
## 5 Afghanistan 2015 76 65 65 65 63 39 70 65 69 NA  
## 6 Afghanistan 2014 74 62 62 62 60 39 70 40 62 NA

immunization.avg <- immunization  
immunization.avg[,"Immunization"] <- rowMeans(immunization.avg[,c('BCG', 'DTP3', 'HepB3', 'Hib3','Pol3', 'MCV1',  
 'MCV2', 'PAB', 'PCV3', 'RotaC')], na.rm=TRUE)  
  
immunization.avg <- immunization.avg[,c("Country", "Year", "Immunization")]  
  
data <- left\_join(data, immunization.avg, by = c("Country", "Year"))  
print(head(data))

## Country Year Life.Expectancy BMI Obesity Alcohol GDP  
## 1 Afghanistan 2016 62.69 23.45830 0.05681314 0.02 547.2281  
## 2 Afghanistan 2015 63.18 23.35209 0.05396362 0.00 578.4664  
## 3 Afghanistan 2014 62.97 23.24576 0.05124183 0.01 613.8567  
## 4 Afghanistan 2013 62.74 23.13956 0.04864080 0.01 637.1655  
## 5 Afghanistan 2012 62.21 23.03342 0.04615575 0.01 641.8715  
## 6 Afghanistan 2011 61.72 22.92832 0.04378424 0.01 591.1628  
## HIV.Prevalence HIV.Incidence Diabetes Immunization  
## 1 4133.235 790.0594 NA 65.22222  
## 2 3630.161 679.0092 NA 64.11111  
## 3 3200.788 584.7016 0.1190786 59.00000  
## 4 2788.420 495.3635 0.1162422 60.62500  
## 5 2429.128 418.4731 0.1133741 62.25000  
## 6 2164.778 362.4356 0.1105226 62.25000

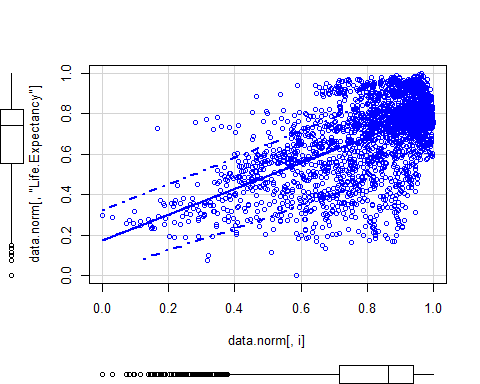
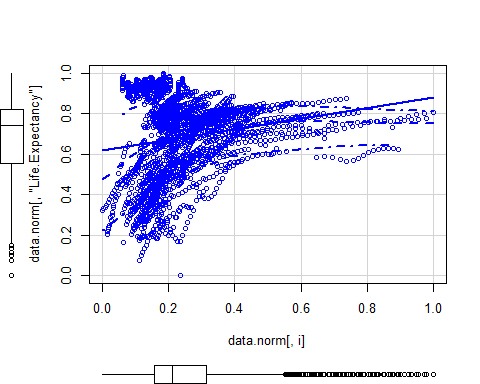
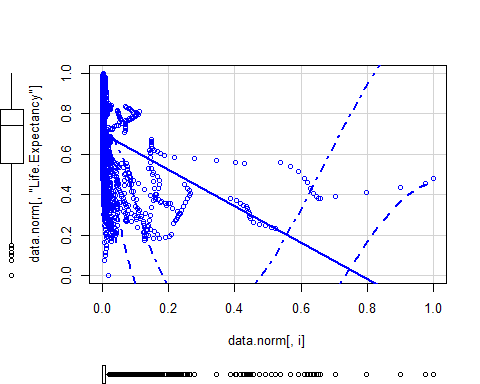
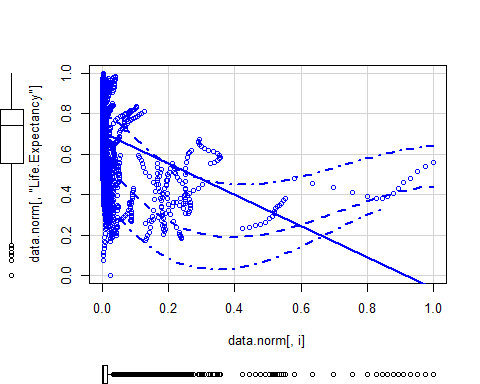
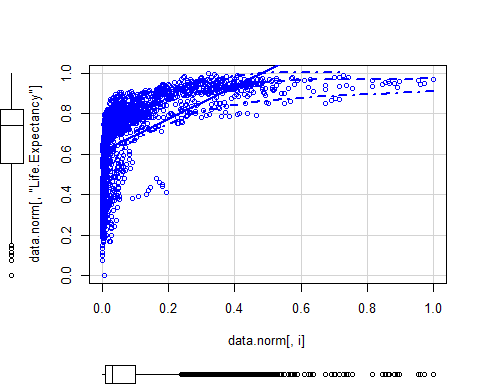
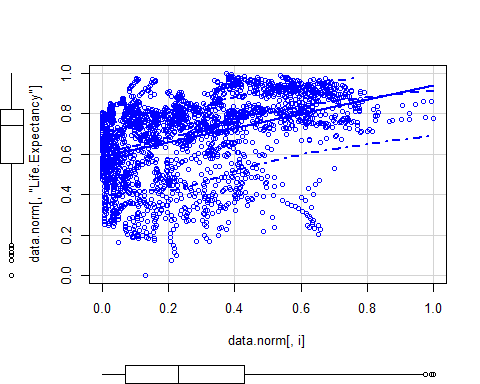
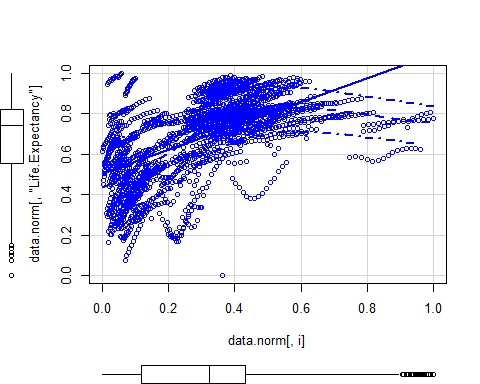
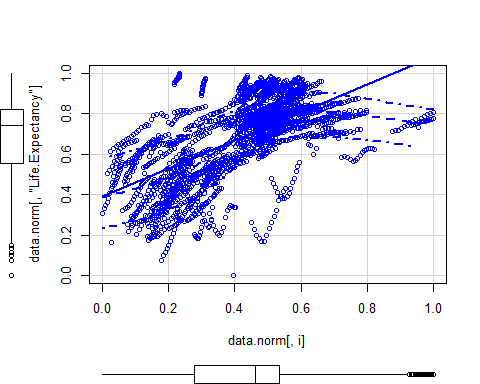
Since linear regression cannot handle missing values, I removed all records with at least one missing value. Next I used min/max normalization on the data since the scales are different for most of the variables.

data <- data[,c("Life.Expectancy", "BMI", "Obesity", "Alcohol", "GDP", "HIV.Prevalence", "HIV.Incidence",   
 "Diabetes", "Immunization")]  
  
  
data <- data[complete.cases(data),]  
  
normalize <- function(x){  
 return((x- min(x)) /(max(x)-min(x)))  
}  
  
data.norm <- lapply(data, normalize)  
  
data.norm <- as.data.frame(data.norm)

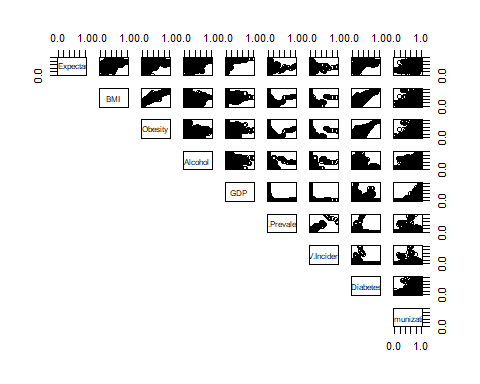
## Exploring the Data

I created scatterplots with life expectancy on the left axis for each of my regressors. I also created a scatterplot matrix of all variables to examine how each variable interacted with each of the others.

for(i in 2:ncol(data.norm)){  
 scatterplot(y=data.norm[,"Life.Expectancy"], x=data.norm[,i])  
}

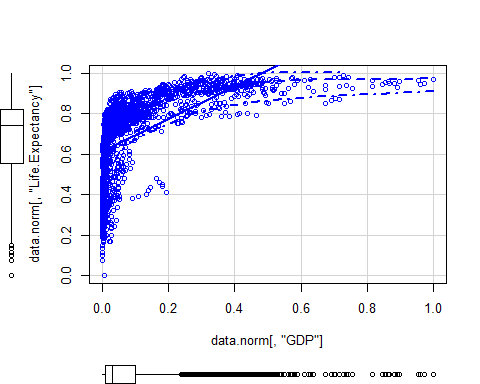


pairs(data.norm, lower.panel = NULL)

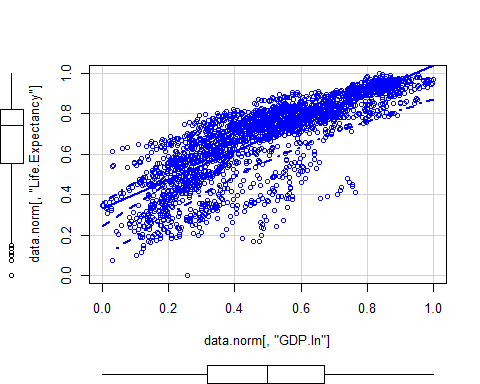


I noticed that the fourth graph (life expectancy versus GDP) had a sharp bend, indicating that the relationship could be exponential. I created a new variable to be the natural log of the original, non-normalized GDP variable. I then normalized this new natural log of GDP variable. I then made a scatterplot with life expectancy on the y-axis and the natural log of GDP on the x-axis and compared it to the original. The graph of life expectancy versus natural log of GDP did not have the bend in it and closely resembled a strong linear relationship.

data.norm[,"GDP.ln"] <- as.data.frame(normalize(log(data[,"GDP"])))  
  
scatterplot(y=data.norm[,"Life.Expectancy"], x=data.norm[,"GDP"])

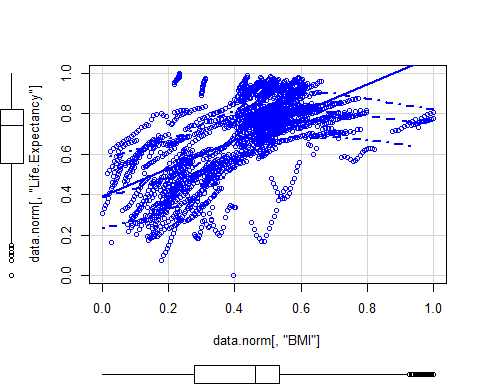


scatterplot(y=data.norm[,"Life.Expectancy"], x=data.norm[,"GDP.ln"])

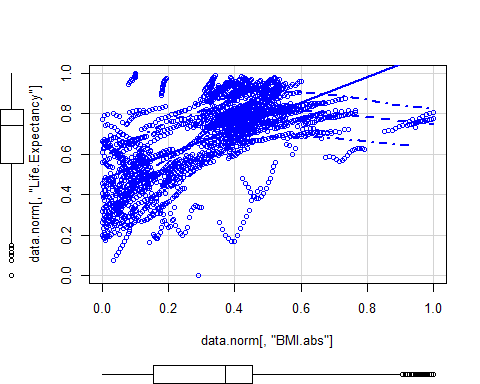


I noticed that the first graph (life expectancy versus BMI) was curved and that the highest values were in the middle, indicating that the relationship may not be linear. I took the median value of the healthy BMI range, 21.7, and subtracted it from the original, non-normalized BMI column, took the absolute value, and normalized it. I then created a scatterplot with life expectancy on the y-axis and the new absolute value BMI on the x-axis and compared it to the original. The graph of life expectancy versus absolute value BMI appeared only slightly more linear than the original BMI graph.

data.norm[,"BMI.abs"] <- as.data.frame(normalize(abs(data[,"BMI"] - 21.7)))  
  
scatterplot(y=data.norm[,"Life.Expectancy"], x=data.norm[,"BMI"])



scatterplot(y=data.norm[,"Life.Expectancy"], x=data.norm[,"BMI.abs"])



I randomly split the data into a test set and a train set.

data.split = sort(sample(nrow(data.norm), nrow(data.norm)\*.8))  
  
data.train = data.norm[data.split,]  
data.test = data.norm[-data.split,]

## Finding the Best Predictors

I created a model named baseline with all regressors. This baseline will be used to compare to the models with the best predictors. For each of the models, I calculated the PRESS statistic, and variance inflation factors for the model. Then I tested for normality. For Model 1, I used stepwise regression to select the regressors.

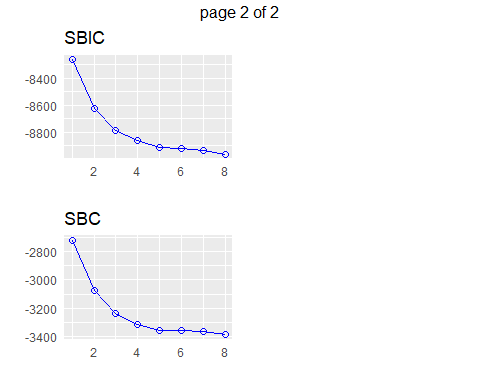
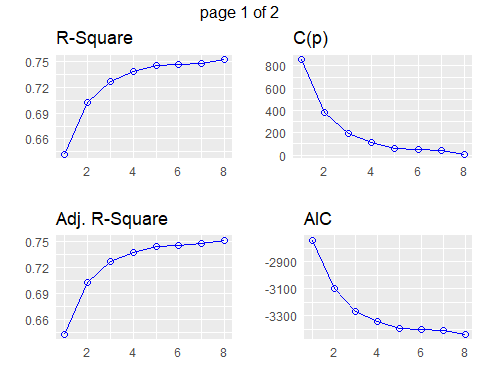
baseline <- lm(Life.Expectancy ~ ., data.train)  
stepwise <- ols\_step\_both\_p(baseline, pent=.04, prem=.05)  
stepwise

##   
## Stepwise Selection Summary   
## ----------------------------------------------------------------------------------------------  
## Added/ Adj.   
## Step Variable Removed R-Square R-Square C(p) AIC RMSE   
## ----------------------------------------------------------------------------------------------  
## 1 GDP.ln addition 0.642 0.642 854.2470 -2741.4411 0.1195   
## 2 Immunization addition 0.703 0.702 383.6730 -3099.6588 0.1089   
## 3 HIV.Prevalence addition 0.727 0.727 192.6520 -3266.4075 0.1043   
## 4 BMI.abs addition 0.738 0.738 109.4520 -3343.8187 0.1023   
## 5 BMI addition 0.745 0.745 57.9220 -3393.4240 0.1009   
## 6 GDP addition 0.747 0.746 49.2440 -3401.8831 0.1007   
## 7 Diabetes addition 0.748 0.747 38.0470 -3412.8893 0.1004   
## 8 Alcohol addition 0.752 0.751 9.1620 -3441.6706 0.0996   
## ----------------------------------------------------------------------------------------------

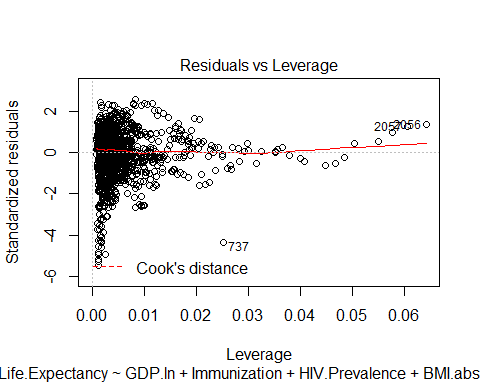
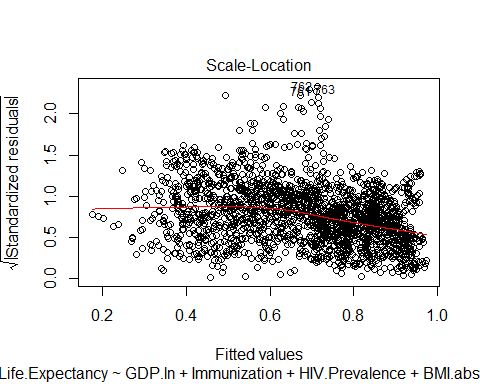
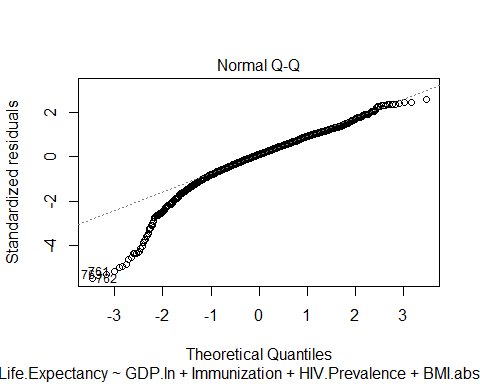
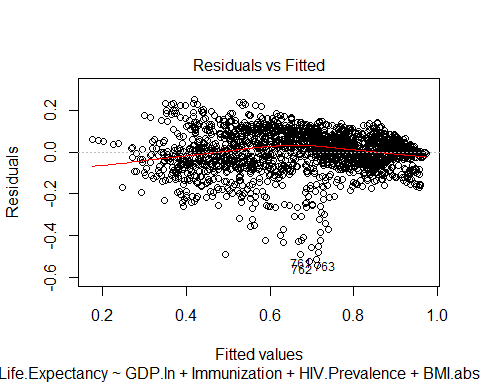
model.1 <- lm(Life.Expectancy ~ GDP.ln + Immunization + HIV.Prevalence + BMI.abs + BMI + Diabetes + Alcohol + GDP, data.train)  
summary(model.1)

##   
## Call:  
## lm(formula = Life.Expectancy ~ GDP.ln + Immunization + HIV.Prevalence +   
## BMI.abs + BMI + Diabetes + Alcohol + GDP, data = data.train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.54279 -0.04854 0.00952 0.06304 0.25306   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.25640 0.01336 19.190 < 0.0000000000000002 \*\*\*  
## GDP.ln 0.63537 0.02432 26.127 < 0.0000000000000002 \*\*\*  
## Immunization 0.23748 0.01484 16.005 < 0.0000000000000002 \*\*\*  
## HIV.Prevalence -0.31718 0.02580 -12.295 < 0.0000000000000002 \*\*\*  
## BMI.abs 0.65346 0.06330 10.324 < 0.0000000000000002 \*\*\*  
## BMI -0.48632 0.07207 -6.747 0.0000000000198 \*\*\*  
## Diabetes -0.18647 0.02937 -6.349 0.0000000002688 \*\*\*  
## Alcohol -0.08830 0.01589 -5.557 0.0000000312085 \*\*\*  
## GDP -0.13752 0.02998 -4.587 0.0000047934137 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.09961 on 1936 degrees of freedom  
## Multiple R-squared: 0.7522, Adjusted R-squared: 0.7512   
## F-statistic: 734.6 on 8 and 1936 DF, p-value: < 0.00000000000000022

plot(stepwise, which=1)



plot(model.1)



#Tests  
PRESS(model.1)

## [1] 19.37254

vif(model.1)

## GDP.ln Immunization HIV.Prevalence BMI.abs BMI   
## 5.751052 1.434195 1.121495 29.563066 32.331604   
## Diabetes Alcohol GDP   
## 3.616086 2.498355 3.322263

ols\_test\_normality(model.1)

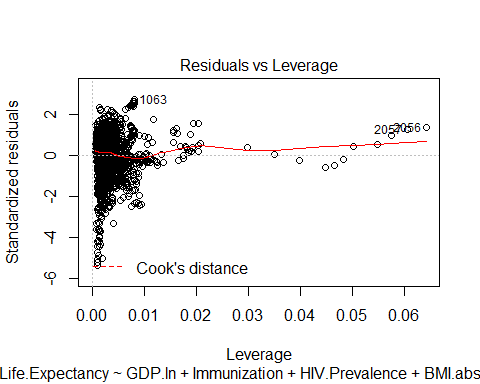
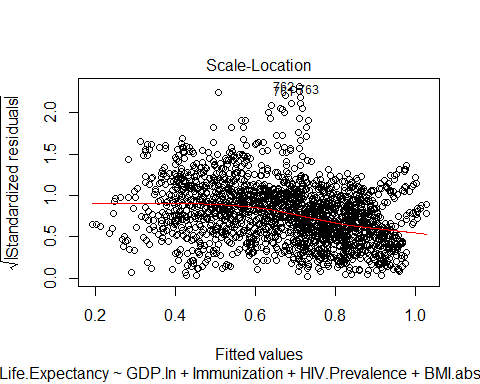
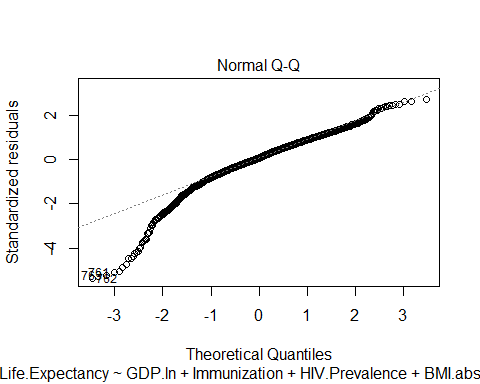
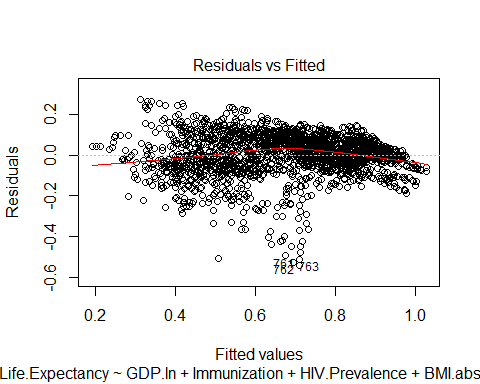
## -----------------------------------------------  
## Test Statistic pvalue   
## -----------------------------------------------  
## Shapiro-Wilk 0.9396 0.0000   
## Kolmogorov-Smirnov 0.0662 0.0000   
## Cramer-von Mises 536.76 0.0000   
## Anderson-Darling 18.6333 0.0000   
## -----------------------------------------------

Unfortunately, using both the mean BMI and the mean BMI’s absolute value will be problematic in the model and cause severe multicollinearity. The same goes for using both the natural log of GDP and the unaltered GDP. Since the absolute value BMI and the natural log of GDP performed better than their unaltered counterparts, I created a second model that used absolute value BMI and the natural log of GDP and the other regressors selected by stepwise regression without the unaltered BMI and GDP.

model.2 <- lm(Life.Expectancy ~ GDP.ln + Immunization + HIV.Prevalence + BMI.abs + Diabetes + Alcohol, data.train)  
summary(model.2)

##   
## Call:  
## lm(formula = Life.Expectancy ~ GDP.ln + Immunization + HIV.Prevalence +   
## BMI.abs + Diabetes + Alcohol, data = data.train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.54173 -0.04917 0.00812 0.06524 0.27258   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.22185 0.01192 18.612 < 0.0000000000000002 \*\*\*  
## GDP.ln 0.53245 0.01559 34.161 < 0.0000000000000002 \*\*\*  
## Immunization 0.23842 0.01487 16.035 < 0.0000000000000002 \*\*\*  
## HIV.Prevalence -0.31196 0.02615 -11.928 < 0.0000000000000002 \*\*\*  
## BMI.abs 0.25841 0.02368 10.912 < 0.0000000000000002 \*\*\*  
## Diabetes -0.19352 0.02860 -6.765 0.0000000000176 \*\*\*  
## Alcohol -0.08846 0.01573 -5.624 0.0000000213965 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.101 on 1938 degrees of freedom  
## Multiple R-squared: 0.7448, Adjusted R-squared: 0.744   
## F-statistic: 942.7 on 6 and 1938 DF, p-value: < 0.00000000000000022

plot(model.2)



#Tests  
PRESS(model.2)

## [1] 19.91083

vif(model.2)

## GDP.ln Immunization HIV.Prevalence BMI.abs Diabetes   
## 2.296377 1.399848 1.120526 4.022371 3.334518   
## Alcohol   
## 2.380157

ols\_test\_normality(model.2)

## -----------------------------------------------  
## Test Statistic pvalue   
## -----------------------------------------------  
## Shapiro-Wilk 0.9426 0.0000   
## Kolmogorov-Smirnov 0.0674 0.0000   
## Cramer-von Mises 534.7593 0.0000   
## Anderson-Darling 18.442 0.0000   
## -----------------------------------------------

## Selecting the Best Model

I performed cross validation on my models.

folds <- cvFolds(nrow(data.train), K = 10, R =100)  
cvfit0 <- cvLm(baseline, cost = rtmspe, folds = folds)  
cvfit1 <- cvLm(model.1, cost = rtmspe, folds = folds)  
cvfit2 <- cvLm(model.2, cost = rtmspe, folds = folds)  
cvFits <- cvSelect(Baseline = cvfit0, Model1 = cvfit1, Model2 = cvfit2)  
cvFits

##   
## 10-fold CV results:  
## Fit CV  
## 1 Baseline 0.05229416  
## 2 Model1 0.05217966  
## 3 Model2 0.05314129  
##   
## Best model:  
## CV   
## "Model1"

test.labels <- data.test[,1]  
test <- data.test[,-1]  
pred <- predict(model.2, test)  
  
mae <- mae(test.labels, pred)  
print(mae)

## [1] 0.07435972

mse <- mse(test.labels, pred)  
print(mse)

## [1] 0.009248624

rmse <- rmse(test.labels, pred)  
print(rmse)

## [1] 0.09616977

mape <- mape(test.labels, pred)  
print(mape)

## [1] 0.1358602

smape <- smape(test.labels, pred)\*100  
print(smape)

## [1] 12.68433

print(100-smape)

## [1] 87.31567

rsq <- function(x, y){  
 return(cor(x, y) ^ 2)  
}  
  
rsq <- rsq(test.labels, pred)  
print(rsq)

## [1] 0.7399126