**Introduction**

The purpose of this project is to use data obtained from Kaggle to create models that will be used to help predict life expectancy. It will be broken up into sections. The first is EDA (Exploratory Data Analysis) where we will take an initial look at the data and perform any cleaning, modification, etc. that is required. Then addressing two objectives. The first of which is aimed at finding relationships and interpreting these relationships. This also includes variable selection and creating our first model. The second objective adds a little more complexity to the model we made in objective 1 to develop multiple models that we can compare to each other. Then we use the best model as the model to predict for future data. At the end will be the Final Summary with a quick recap and findings.

**Data Description**

The data can be found here: <https://www.kaggle.com/kumarajarshi/life-expectancy-who>. The data was collected by the WHO (World Health Organization) over the years of 2000 – 2015. From the data source, “In a nutshell, this study will focus on immunization factors, mortality factors, economic factors, social factors and other health related factors as well”. It is important to note that the data obtained excludes these countries, Vanuatu, Tonga, Togo, Cabo Verda, and other small fewer known countries. The data set consists of 22 Columns, 2,938 rows which gave us 20 predicting variables.

**EDA (Exploratory Data Analysis)**

Note: Throughout the EDA multiple versions of the training data is made to keep track of changes made.   
 The first step when conducting our EDA was to find where the missing values were and how to deal with them. Just counting the number of rows with “NA’s” gave us a count of 1,289. So, we know at least 44% of rows in our data have some sort of missing data. Let’s investigate which columns have missing data. Results below:  
*Columns with # of 'NA' values*

Life.expectancy: 10, Adult.Mortality: 10

Alcohol: 194, **Hepatitis.B: 553**

BMI: 34, Polio: 19

Total.expenditure: 226, Diphtheria: 19

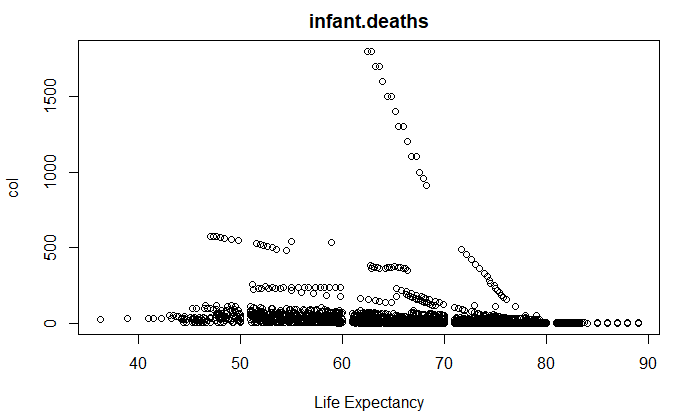
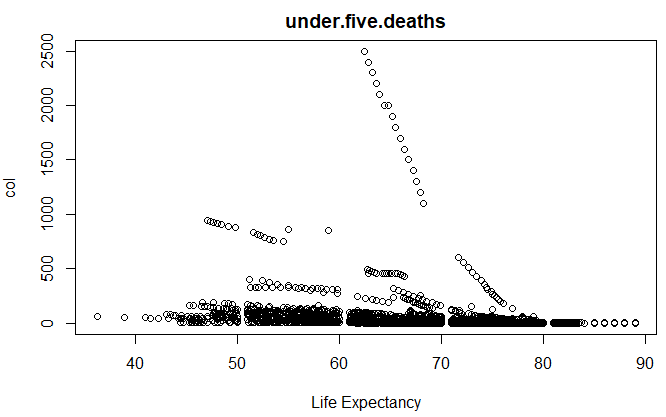
**GDP: 448**, **Population: 652**

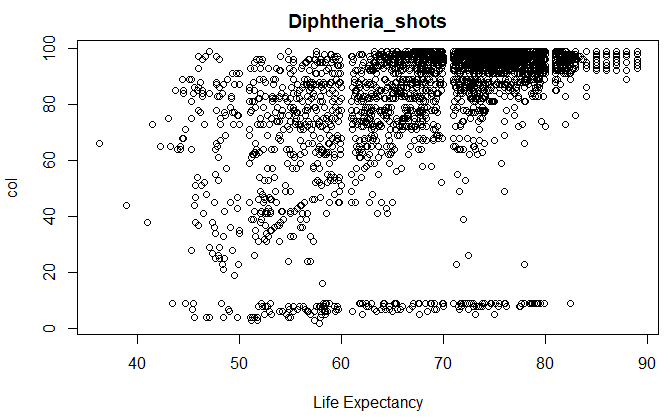
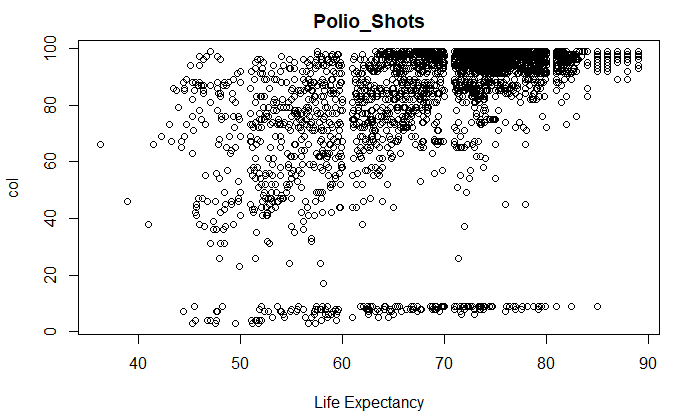
thiness1-19years: 34, thiness5-9years: 34

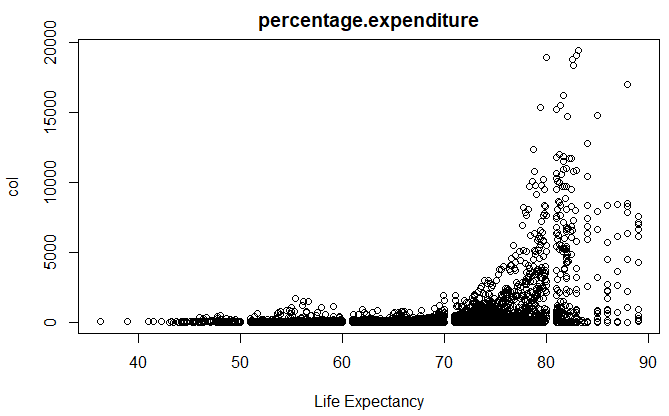
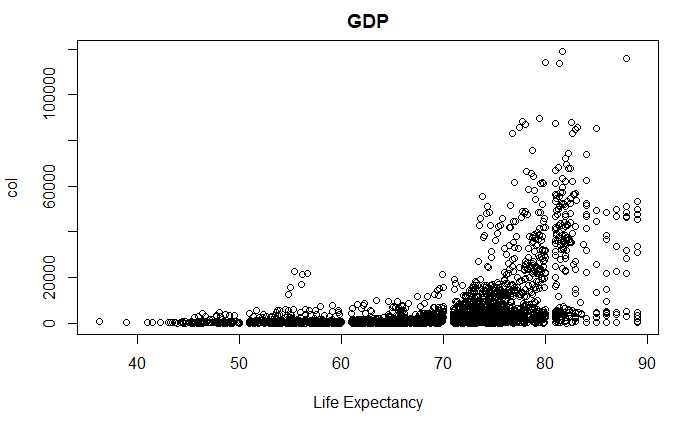
Income.comp.of.resc: 167, Schooling: 163

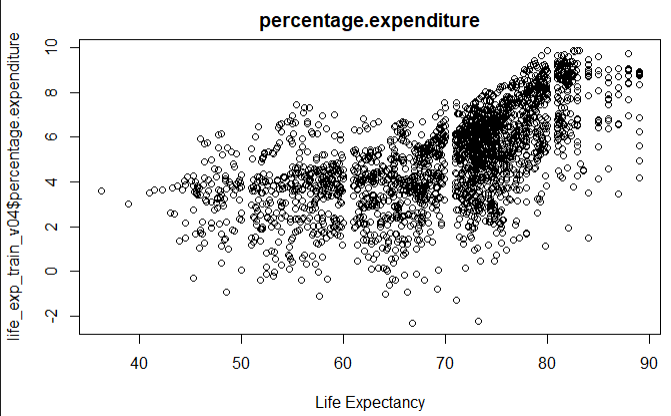
The first notable instance is that Life.expectancy has 10 missing data values. Since our models will be built trying to predict Life.expectancy, we cannot include those in our training model. There are also 3 columns that have a high amount of NA values (Hepatitis.B, GDP, and Population). Other notes include a lot of 0 values in the infant death’s column with no missing data values. Need to keep an eye on those columns.

Second part includes renaming some of the columns because the names of the columns can be misleading. Hepatitis.B, Polio, and Diphtheria is actually a representation of percentage vaccinations among 1 year old. As a result, “\_shot” was appended to the name. The measles had the “\_cases” appended and it’s per 1,000. Total.expenditure changed to “Total\_Health\_Spending”. Lastly HIV.AIDS is actually based on the number of deaths of infants that died at birth based on HIV/AIDs. That was changed to HIV\_AIDS\_birth\_deaths. After the name changes the columns Country, Year, and Status was changed to a Factor. Country and status were self-explanatory as to why we did this, but year was decided to be a factor because we want to see the effect of each factor having on an individual year instead of averaging it as a continuous variable.

Next was plotting all columns against life expectancy. These plot were noted:  
  
**Infant.Deaths vs. Under.five.deaths –** Similar categories and look highly correlated.   
 

**Diphtheria vs. Polio** – Very similar results   
 

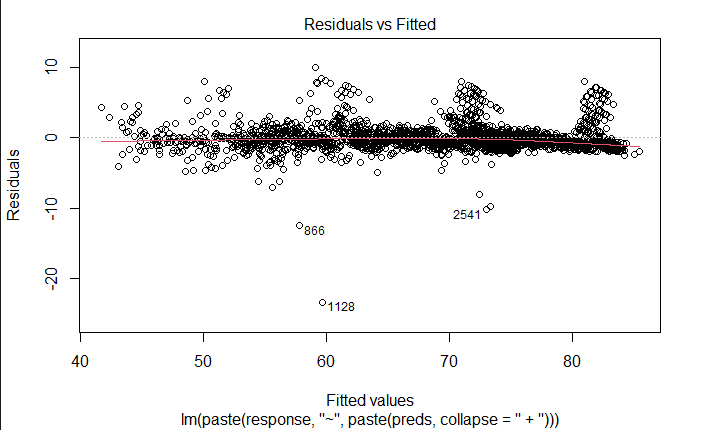
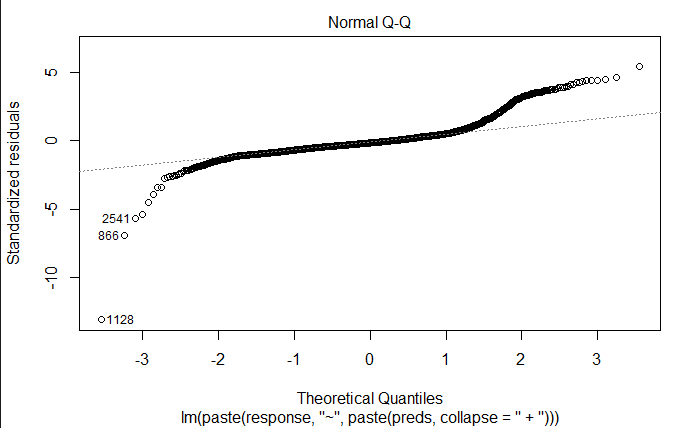
**Percentage.expenditure And GDP** – Both look exponential and could benefit from a log transformation.   
 

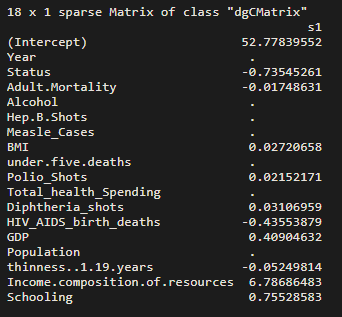
The next decisions after looking at this data was to first log trasform the GDP and Percentage.expendature data to get a better fit against the life expectancy data. The resulting graphs look like this.   
   
It’s not perfect, but it is better than the exponential chart we had before.

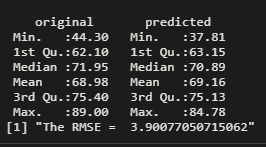
Next was creating a correlation plot to find variables that we can remove immediately before moving on to the variable selection portion. Based on the results we had 3 highly correlated values. Note at this point the thinness.5.9.years has already been removed, but I previous test was done that showed a correlation of .99. The other three were (GDP vs. Percentage.expenditure) at .94, (under.five.deaths vs infant deaths) at .99, and (Hep.B\_Shots vs Diptheria\_shots) at .61. The decision is to drop Percentage.expendature and infant.deaths. The shots can remain because there is not enough evidence to suggest it is not co-linnear just yet.

**Objective 1**

The objective is trying to find the life expectancy given the predictors from the data set and the modified data set already created. We will start with a couple basic variables selection methods and then move on to a LASSO selection method.

For the basic variable selection, we decided to stick with the stepwise selection and go from there. The first was a stepwise selection that used the P values and the second used an AIC value. The Residuals and Q-Q plot were not fantastic as noted below.   
   
  
The choice here is to not use either of the stepwise models. They could have been a little useful, but knowing the amount of NA data that was used in the data and knowing that the LASSO model outperformed the stepwise models, we decided to sidestep these choices.

The LASSO model was a little tricky to get going. The problem with LASSO is that it will not handle NA values. Also important to note at this point is that the Country column is no longer used. To handle the NA values we hired the **“mice”** library to help populate the missing data. The parameters for mice was setting a seed to 1234 and using method=”cart”. From here we split the data 85/15 and trained the data using “cv.glmnet(lasso\_x, lasso\_y, alpha=1). We used alpha = 1 to simulate a LASSO model. The cutout below shows the results of the variables selection from the model.  
   
The variables that have a dot instead of a number next are ones that are not included in the model. Any variables that has a negative sign means there is a negative impact to Life Expectancy and positive numbers have a positive impact to life excpectancy. Residual plots where attempted, but very hard to obtain for a model using the glmnet function. With more time a residual plot can be produced.

When using the model to create predictions we created a fairly useful model that gave us very close numbers in the summary. It also gave an RMSE of 3.90.   


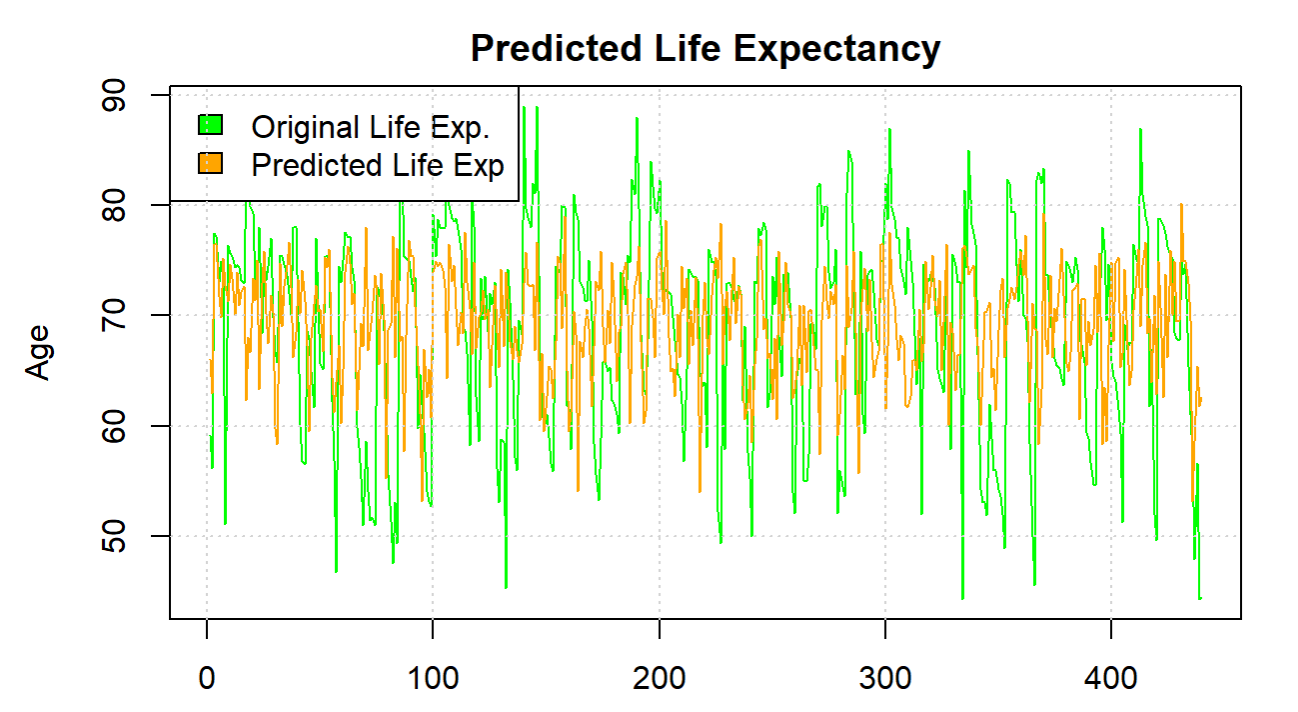
**Objective 2**

To predict life expectency of individuals from historical data, we’ve taken steps to clean the data set and complete missing data points within the set. We’ve evaluated the data provided and determined which columns would be unlikely to contribute to a valuable prediction. In this section, we have fitted a K Nearest Neighbors model to predit the life expectency of people.

K Nearest Neigbors attempts to determine where a value you seek to predict lies within values of trained data, based on the variables it recieves to predit upon. It then uses those nearest values, or neighbors, to determine the predicted value. Think of this as saying “Among all of these previous values I have seen, which ones does it look most like? Let’s make an estimate based on those known values.”

A potential downfall of this type of model is over fitting to your training data set. If your training set has seen very specific cases under a certain set of circumstances, this could create a bias in prediction that fails to account for changes in the predictor variables due to structural or environmental changes of that data set. In our life expectancy data, this may be something as simple as a new, low cost medicine to treat a disease that kills many in lower wealth nations. Because the training data would not have seen that shift in the predictor variables yet, it would fail entirely to predict because of a bias towards a world where that medicine never existed. Eliminating biases such as these would require monitoring and retraining when some event occurs that changes the dynamics of the model.

In our case, the KNN model performed with a root mean square error of 9.435 years. The chart below represents the prediction vs the actual values among the testing split of the data.



Overall, the model appears to be slightly overfit to the training data. As there were numerous points of data which we missing and needed to be imputed, I suspect that we’ve weakened slightly the ability of the KNN Regression model from being as powerful as it could be. I think exploring other methods to impute the missing data, even possibly by grouping by status and/or Country, could improve the values imputed and ultimately improve the performance of the KNN and other non-paramteric models.

**Summary**

Quick Recap: From objective one we discovered a method of using the mice library to fill in missing data using that algorithm. From there we were able to create a LASSO model that had an RSME value of 3.9. From objective 2 we discovered the KNN model had a RMSE of 9.435. The theory is that the KNN model suffered from overfitting the data. Variables that were removed [Country, thinness.5.9.years, infant.deaths, percentage. Expenditure]. Caution should be taken as we had to impute some data using the mice library. Also, good to note is a reminder that this data set does not contain smaller lesser known countries and data generated may not be applicable for those countries. While very powerful, it’s not 100% factual. I feel confident this data could be used to predict life expectancy from future values. If we had more time we would like to explore more into comparing different models using ANOVA tables. If we also had more time I would have liked to replace the countries and put regions or continents instead.

**Appendix**

All code can be found from the main branch at: <https://github.com/Abillelatus/MSDS-6373-Project-1>  
There is an included html file that has a knitted version of code.