# Final Project: Life Expectancy

Karla Cornejo Argueta, Satoshi Shinkawa, Sarah Hazan, Sean Huang

2024-12-10

# Introduction

Life expectancy is increasing due to modern advancements in medicine, technology, and standards of living worldwide. Between 2000 and 2015, life expectancy increased globally by 5 years (World Health Organization, 2016). However, major disparities in life expectancy persist globally with some countries projecting into the 50's while others project into the 80's.
In this report, we investigate the global differences in life expectancy through the following research questions in hopes of understanding why and how countries differ in life expectancy and what impacts it: (1) what factors impact life expectancy, (2) do demographic or health-related factors contribute more and how, (3) how can life expectancy inform us about a country, and (4) what model best predicts life expectancy and a country's status.

## Data Description

We used the World Health Organization's data set on life expectancy found on kaggle, containing 22 features (country, year, status (developed vs. developing), life expectancy, adult mortality, infant deaths, average alcohol consumption per person, percent expenditure, hepatitis B, measles, BMI, deaths (under 5 y.o., polio), total expenditure, diphtheria, HIV/AIDS, global domestic product (GDP), population, thinness (10-19 y.o.), thinness (5-9 y.o.), income composition, schooling) and 2,938 observations for 193 countries spanning 16 years (2000-2015) (Rajarshi, 2018). Of these, only Status and Country are categorical.

## Data Preprocessing

### NA Removal

We found 2,563 NA values spanning 1,289 observations, meaning 43.9% of our observations contain NAs. Removing those observations would remove nearly half of our observations, remove entire countries, most of which are developing, skew our data significantly, and ultimately lead to inaccurate results. Instead, we investigated which variables contained mostly NA values. Those were hepatitis B, GDP, population, income composition, and schooling. We removed those variables, resulting in only 262 observations containing NA's. Upon removing those NA's, we had 2,676 observations where no country was removed completely.

### Standardization

We then standardized our data. Standardization involves centering each observation by subtracting the variable's mean from each of its respective observations, and dividing each observation by their corresponding feature's standard deviation. This removes features' false importance from a difference in scales. For example, without standardization, a variable with a range in the thousands will have more weight than a variable with a range from 1-10, but with standardization, this false weight is removed. We did not standardize Country, Year, and Status because they are categorical or time-dependent, thus standardization is not appropriate.

### Codification

Since Country and Status contain character values, we codified them by replacing their respective values with corresponding numbers, thus allowing for further analysis. Country ranges from 1 to 193 with each value relating to a certain country. Status has 2 values: "1" represents "developed" countries and "2" represents "developing" countries.

# Exploratory Data Analysis

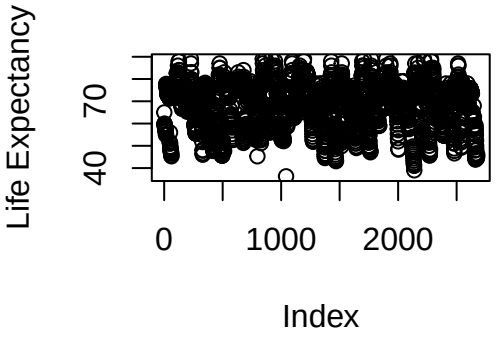
## Summary Statistics

We calculated summary statistics of the mean, standard deviation, median, minimum, maximum, range, skew, and standard error of each variable to obtain an overview of variable distributions. Life expectancy has a mean of 69.28 years, standard deviation of 9.54 years, minimum of 36.3 years, maximum of 89 years, range of 52.7 years, skew of -0.67, and standard error of 0.18, all of which are expected. Adult mortality, infant death, percent expenditure, measles, and deaths (under 5 y.o.) had high standard deviations in the hundreds, thousands, and ten thousands. Measles had the highest standard deviation at 11,822.58, indicating a wide variability in measles rates globally. Such variability can be attributed to vaccine availability and healthcare differences between countries.

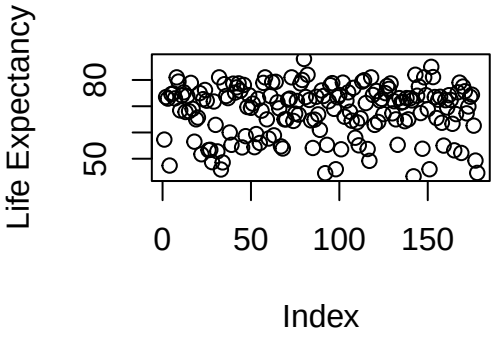
## Scatterplots

We plotted scatterplots of all variables and provide a scatterplot of life expectancy below for simplicity in Figure 1. As with all of our plots, we can see groupings of points since all years were utilized for this plot.

**Figure 1: Life Expectancy**




**Life Expectancy in 2005**

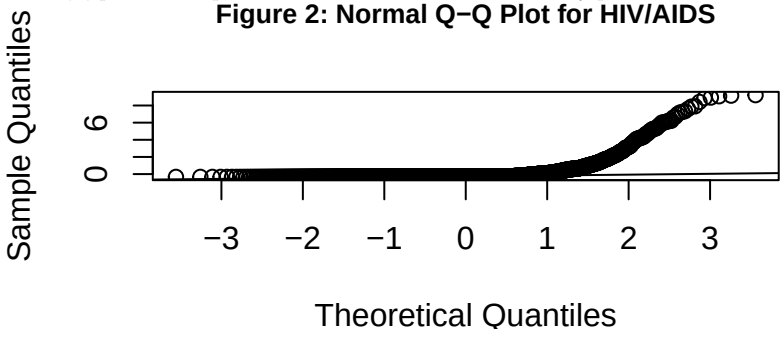


When plotting only one year, such as in the second plot in figure 1, we see the points lose their groupings, indicating randomness.

## Quantile-Quantile Plots

We graphed quantile-quantile plots (QQ-plots). QQ-plots depict each variable's values in ascending order against their theoretical quantile. In other words, it plots the actual value vs. its theoretical predicted relative. We plotted QQ-plots for all features and found that none of the variables follow the projected line, indicating a lack of normality among all features. Figure 2 is one such QQ-plot and depicts the most common non-normality pattern in the data.

**Figure 2: Normal Q−Q Plot for HIV/AIDS**




However, even though none of the features are normally distributed, we are still able to proceed with our analysis since the non-normality only lowers our accuracy rate rather than rendering our prediction methods useless.

## Correlation

We calculated and displayed the correlation coefficients for each pair of variables in Figure 3.

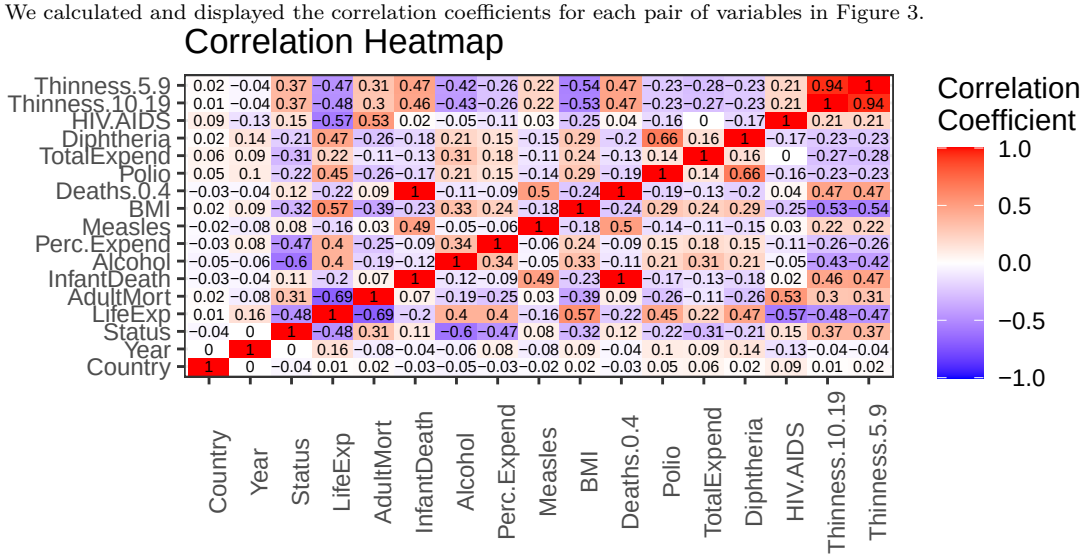



Figure 3. Correlation heatmap of all combinations of all variables and their correlation coefficients

Overall, we see stronger relations between disease-related factors with each other and life expectancy. Many associations are close to 0, especially for Country. For life expectancy, we see Adult Mortality and Status have strong negative correlations with it at -0.69 and -0.48 respectively. Since the Status correlation is much stronger compared to the correlation between Country and life expectancy, we infer that a country's status matters significantly more than the country itself. Many disease-related variables, such as polio, have moderate impacts on life expectancy with correlations around -0.5, indicating stronger impacts on life expectancy compared to other factors.

# Methodology

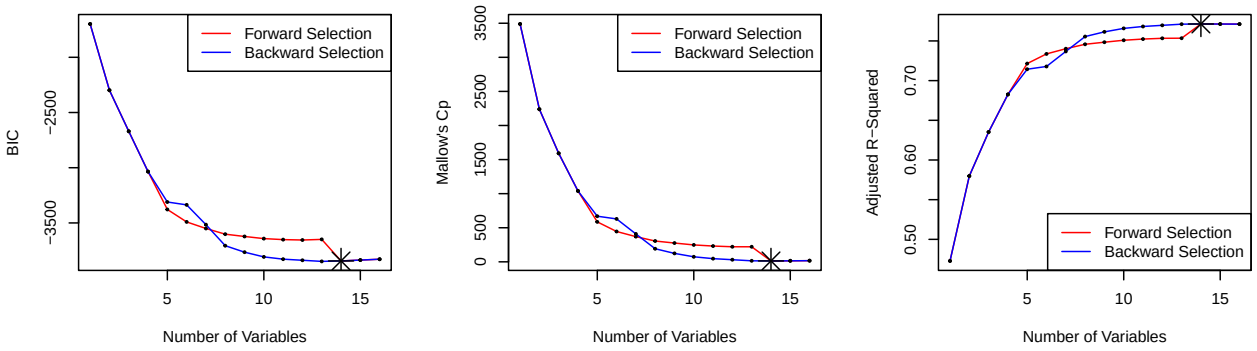
## Feature Selection

### Stepwise Forward and Backward Feature Selection

The first method we utilize to analyze our data is stepwise forward and backward feature selection evaluated using BIC, Mallow's Cp, and Adjusted-R2 criteria. Forward stepwise selection starts with an empty model, and adds and fits each variable based on maximized importance until all features are fitted in the model. Conversely, backward stepwise selection starts with the full model with all features fitted, and removes the least important feature until the model is empty. Both methods are evaluated using their corresponding BIC, Mallow's Cp, and Adjusted-R2 values.

The Bayesian information criterion (BIC) value measures a model's likelihood to predict accurately. The model with the lowest BIC is favorable as this implies low error of the model when predicting. Mallow's Cp acts similarly to BIC in that low Mallow's Cp implies low prediction error. The Adjusted-R2 value measures a model's accuracy while penalizing models for including unneeded features. The model with the highest Adjusted-R2 value is favorable as it implies high prediction accuracy.
We computed and graphed the BIC, Mallow's Cp, and Adjusted-R2 values for each model using both forward and backward stepwise selection in Figure 4. According to this figure, 14 variables yields the best subset for model prediction across all criterion and selection methods.

**Figure 4. Feature Selection**




After fitting this model, we find the following variables are the most impactful and estimate their model coefficients in Table 1. The coefficient estimates are all small with the largest magnitude at -1.79 for Deaths (under 5 y.o.).

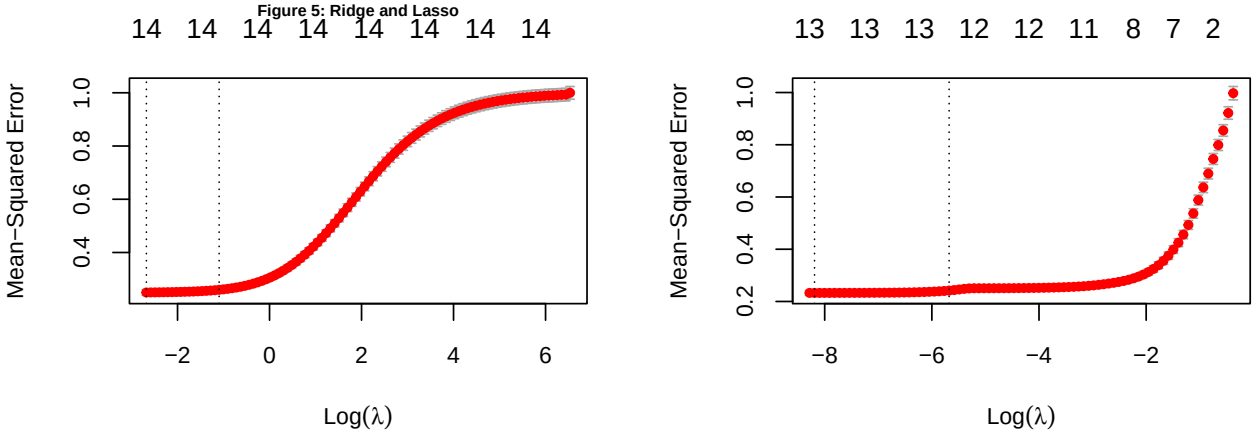
| | Coefficient Estimate |
|---|---|
| (Intercept) | -22.8310909 |
| Country | 0.0007754 |
| Year | 0.0115660 |
| Status | -0.2482066 |
| AdultMort | -0.3008693 |
| InfantDeath | 1.7676845 |
| Alcohol | 0.1170027 |
| Perc.Expend | 0.1162227 |
| Measles | -0.0207271 |
| BMI | 0.1676813 |
| Deaths.0.4 | -1.7925762 |
| Polio | 0.0881411 |
| Diphtheria | 0.1119056 |
| HIV.AIDS | -0.2634233 |
| Thinness.10.19 | -0.0559885 |

## Ridge Regression and Lasso

After applying AIC/BIC model selection to filter the best model for the regression, we use the selected variables for a ridge/lasso test to avoid overfitting and check whether the result comes along the model selection result. There is a difference between the two regression models. Ridge regression adds a penalty equivalent to the square of the magnitude of coefficients to the loss function. Lasso regression adds a penalty equivalent to the absolute value of the magnitude of coefficients to the loss function. By minimizing the objective function, the result is the best-fit model. Because of this difference, both models shrink coefficients but LASSO shrinks coefficients to zero, and ridge does not. This helps perform feature selection. This causes the main difference. Ridge was used when all the variables were predicted to be impactful on the result, LASSO was used to simplify the model when there are many features but only a few are expected to be important.

The more impactful variable has a larger absolute value for the result coefficient of the variables acquired from the Ridge model. From the table, Deaths.0.4, Status, AdultMort, Alcohol, Measles, HIV.AIDS are the more relevant variables when estimating life expectancy.

Figure 5: Ridge and Lasso




The 14s above the plot indicate all variables are included. The Y-axis indicates the MSE and the X-axis is the logarithm for lambda which is the regulation parameter, it indicates the shrinkage of the coefficients. From the plot, with low values of Log lambda (closer to -2), the MSE is at its lowest. This suggests that with minimal regularization, the model fits the data very well, capturing most of the variance. As Log($\lambda$) continues to increase (moving towards 6), the MSE rises sharply. This indicates that too much regularization causes the model to underfit, failing to capture important relationships in the data. The vertical dotted lines indicate two values of lambdas, the one on the left is the point where the MSE is the lowest. The one on the right is lambda.1se. It is the largest value of lambda such that the cross-validated error is within one standard error of the minimum MSE. This often results in a simpler model that is more generalizable.

From the table for LASSO Deaths.0.4, InfantDeath, AdultMort, HIV.AIDS and Status are the variables with the higher absolute value. It indicates these variables are relevant for predicting life expectancy.

The 13 to 1 above the plot represents the range of the number of non-zero coefficients included in the model as the lambda value changes across the range of values displayed in the plot. The X-axis and Y-axis are the same indicators for the ridge plot. At lower values of Log($\lambda$) (closer to -8), the MSE is low. This suggests that a lower level of regularization (smaller $\lambda$) results in a better fit. As Log($\lambda$) increases, the MSE remains relatively flat and low over a range of values, indicating a stable $\lambda$ values. However, as Log($\lambda$) continues to increase beyond a certain point (around -2 to 1), the MSE starts to rise sharply. This indicates that higher levels of regularization (larger $\lambda$) lead to worse model performance, likely due to the model becoming too simple and underfitting the data. The vertical dotted line indicates the lambda value selected by the cross-validation process where the MSE is minimized, balancing the trade-off between bias and variance.

## Classification

We used classification to explain what life expectancy can tell us about a country, which factors can we predict using life expectancy as a predictor, and if the accuracy of these methods is reasonable. Classification methods seek to calculate the probabilities that a factor belongs to one specific category and use this probability to assign the data into predicted categories.

### Logistic Regression:

To answer what life expectancy can tell us about a country, we wanted to use logistic regression to predict some of our categorical binary variables such as Status. Logistic regression is a method that looks at the data and spits back probabilities where we then choose a threshold in which those probabilities belong to one category or the other. We chose a 0.5 threshold, meaning, there is a 50% chance a country is developed. We fit four different logistic regression models: (1) Using life expectancy to predict country development status; (2) Using all of the data to predict country development status; (3) Using only the previously selected best variables to predict country development status; (4) Using the previously selected best variables, excluding life expectancy, to predict country development status.

### Linear Discriminant Analysis (LDA):

Since the data does not perfectly follow the logistic regression multicollinearity assumptions, meaning that some of the factors are correlated. Instead, LDA uses an assumption of normality to separate the data through a linear decision boundary. Using this, it predicts the categories of new data based on what this boundary says about the seen data. We used a different method since, though, assumptions are violated for all 3 methods, comparing the accuracy will lead to the best model. We fitted the same four models as above and compared the accuracy of each.

### Quadratic Discriminant Analysis (QDA):

Same as with LDA, we used QDA to assess accuracy under different assumptions and used the same models as above. QDA is similar to LDA, except some assumptions are more relaxed since the decision boundary for this method is curved instead of linear. So, it allows for more flexibility in the data. For both LDA and QDA, we omitted highly non-normal variables such as HIV and Polio since those violated the assumptions of LDA and QDA to an extreme.

### Multinomial Regression:

Multinomial regression uses the same logic as logistic regression, except it deals with more than two categories. Multinomial regression uses likelihood to trace back probabilities that the data belongs to each category. We used multinomial regression to predict country-given life expectancy, we found this to be highly inaccurate. We then split the life expectancy into categories

so that life expectancy can be related among countries. We fit two multinomial models predicting life expectancy categories. One used all the predictors in the data, while the second used only the best-selected categories. We then compared accuracy percentages.

## Prediction

We selected the methods for prediction by keeping in mind that our data is non-normal. Thus, non-parametric methods are more ideal in successfully predicting for our dataset. We randomly sampled our data into training and testing sets with a ratio of 50% to 50%.

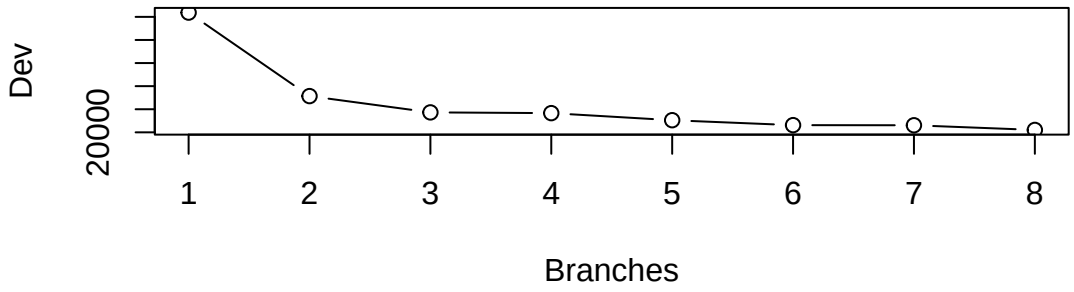
### Regression Trees

Our first method we used was regression trees. Regression trees are a type of decision trees that allow the response variable to be continuous. Decision trees essentially break apart data in order to simplify it to be able to observe the relationships between the predictor variables and the response variable. Each split, or branch, represents a decision rule that optimally separates data to eventually minimize within group variability.

### Pruning and Cross Validation with Regression Trees

Pruning is a method employed on decision trees to determine whether adding extra branches ends up causing the model to become worse. By applying this method and plotting the number of branches by the deviance, you can determine the ideal number of branches that will result in a minimized mean squared error. The idea is that regression trees are made to capture complex and non-linear relationships in data that aren't restricted by assumptions. In our study, we concluded that including all of the branches resulted in the best model as seen in the plot. The lowest deviance is found when all 8 branches are used.

**Figure 6: Regression Tree Branch vs Dev**




### Bagging

Bagging, or also known as Bootstrap Aggregation, is a machine learning method used to improve stability and accuracy of algorithms, This method reduces variance and prevents overfitting making it a strong method to use. Multiple models are combined after being trained on different subsets of data in order to attain improved model performance.

### Random Forest

Random forest essentially takes several decision trees and combines them in order to improve predictive performance and control overfitting. This method is an extension of bagging. When using this approach, the weaknesses of decision trees are mitigated making it a strong method to use on non-linear data

### Boosting

Boosting is a machine learning method that combines several weaker models to reach a stronger predictive model. This is similar to bagging except that boosting creates sequential models that improve upon the errors of the past models. This method is very effective at improving overall model performance.

# Main Results

## Feature Selection

From our feature selection, we found Country, Year, Status, Adult Mortality, Infant Deaths, Average Alcohol Consumption, Percent Expenditure, Measles, BMI, Deaths (under 5 y.o.), Polio, Diphtheria, HIV/AIDS, and Thinness (10-19 y.o.) to be the best subset of variables yielding the best model. One possibility for this many significant features is that their interactions may have a strong impact, thus removal of one would drastically reduce the accuracy of the model. Interestingly, Country has the smallest coefficient estimate in terms of magnitude (0.0008) while Status's estimate was much larger (-0.2482), further substantiating that a country's status is more impactful than the country itself. Finally, 10 of these impactful variables are disease-related, with only 4 demographic variables remaining significant, thus implying health-related factors impact life expectancy more.

## Classification

### Predicting Country Development Status:

After examining the different models, we found that under logistic regression the models with more than one predictor failed, which means the assumptions of the data do not fit the assumptions of logistic regression. Life expectancy as the sole predictor

had a high accuracy of approximately 90% under logistic regression. Under LDA, the models with more variables had higher accuracy, but found that leaving life expectancy out will decrease accuracy, though, slightly. Under QDA, we found that using life expectancy as a sole predictor yields the highest accuracy amongst the four, of approximately 86%. Using this we can conclude that for simplicity life expectancy is the best predictor for the a country's development status which further establishes the relationship.

### Predicting Countries and Life Expectancy:

We concluded that countries and life expectancy are not highly related, and using life expectancy to predict countries is highly inaccurate with an accuracy of 1.8%.

On the other hand, predicting life expectancy under categories is highly accurate with an accuracy of 78%. We found that reducing the number of predictors to the best predictors found under feature selection decreases the accuracy by 2%. Using the first model -the one with the higher accuracy- we created a world map that showcases the original life expectancy category, its respective category code (from 1-10), and the predicted category code under the model. Here is the world map.

From the map we gathered that life expectancy tends to remain within the same category by regions, and it showcases a highly accurate prediction under the model. Thus, we can conclude that factors from each country affect life expectancy, and these remain similar through regions.

## Prediction

### Predicting Life Expectancy Using the Lowest MSE:

Our group used the mean squared error (MSE) to determine which of our prediction methods were performing the best on our test dataset. Mean squared error is the sum of the observed values subtracted from the expected value, squared and then divided by the sample size. Using this metric, we observe the MSE and then compare them. For MSE, we want to choose the lowest value which means that the given method makes the least amount of mistakes. Regression trees had an MSE of 16.05763 with the most important factors being HIV/AIDS, Adult Mortality, Thinness.10.19, BMI and Alcohol. When we tried pruning the regression tree, we determined that the best number of branches ended up being the entire tree so we did not remove any branches. For bagging, we observed an MSE of 5.128707 with the noteworthy factors being HIV/AIDS and Adult Mortality. Next, for random forest, we observed an MSE of 4.986598 with the most important factors being HIV/AIDs, Adult Mortality, and BMI. Lastly, for boosting, our group observed an MSE of 4.450079 with HIV/AIDs and Adult Mortality being the highest relative influence.

# Discussion

Our analysis retained many strengths. We removed NA's based on variables, which avoided skewness and inaccurate data conclusions, rigorously investigated subsets through use of multiple methods, and created multiple prediction models to deeply investigate the methods producing the best predictions. Additionally, the methods we chose were inherently robust. The use of multiple criterion for the stepwise selection methods ensured penalization of large models and reduced as much error as possible. Stepwise selection methods also remove chances of overfitting through its restrictive process. Our tree-based methods are easily interpretable and yield low error values. With the additional applications of pruning and boosting, we were able to make our trees more accurate as well. Also, most of our regression and classification methods had relatively high accuracy rates.

Our analysis was limited, however. Primarily, since our data was not normal, we violated the normality assumptions, resulting in less accurate results from our methods relying on such normality, such as LDA, QDA, etc. Additionally, for our logistic regression, LDA, and QDA methods, we removed Polio and HIV/AIDS, which may have skewed our results. In this logistic regression, Infant Deaths, Deaths (under 5 y.o.), Thinness (10-19 y.o.), and Thinness (5-9 y.o.) all presented high multicollinearity, leading to a less efficient model.
Future work and areas for improvement include: (1) investigating the interaction effects between variables, (2) analyzing life expectancy due to the effects of the COVID-19 pandemic, (3) conducting PCA analysis to create more accurate models and provide a more comprehensive analysis, (4) conduct a time series analysis on life expectancy and its significant factors, and (5) address the non-normality assumptions through Box-Cox feature transformation.

# Conclusion

We conducted our research using various subset selection techniques, regression analyses, and classification methods. We found these factors impact life expectancy: Country, Year, Status, Adult Mortality, Infant Death, Alcohol Consumption, Percent Expenditure, Measles, BMI, Deaths (under 5 y.o.), Polio, Diphtheria, HIV/AIDS, and Thinness (10-19 y.o.).

Additionally, we conclude that health-related factors contribute more to life expectancy compared to demographic factors due to their higher significance and correlative impacts on life expectancy. Of these health-related features, disease-related features are the most impactful, like Polio, Diphtheria, HIV/AIDS, and Measles, indicating a need for disease prevention to increase life expectancy compared to other factors.

As for the differences between countries, we deduce that life expectancy and distinct countries are not good predictors of each other. Instead, life expectancy and a country's development status are highly interdependent as they have high correlation coefficients and both features have high coefficient estimates in models predicting the other. This implies that it does not particularly matter which country you reside in. What matters more is if the country is "developed" or "developing" when estimating and predicting life expectancy.

Finally, the model best-predicting life expectancy is the boosting model with the lowest error (4.450079). Since our data is not normally distributed, a nonparametric method is needed to develop a predictive method. If our data was more noisy, we would

expect boosting to overfit our data and random forest to instead be superior. Boosting captures the complex patterns that our data contains, resulting in the best prediction model.

# References

Mooney, Paul. (2020, March 13). Latitude and Longitude for Every Country and State. Kaggle https://www.kaggle.com/datasets/paultimothymooney/latitude-and-longitude-for-every-country-and-state/data

Rajarshi, Kumar. (2018, February 10). Life expectancy (WHO). Kaggle. https://www.kaggle.com/datasets/kumarajarshi/life-expectancy-who/data

World Health Organization. (2016, May 19). Life expectancy increases by 5 years, but inequalities persist. World Health Organization. https://www.who.int/news/item/19-05-2016-life-expectancy-increased-by-5-years-since-2000-but-health-inequalities-persist

```r
## SET UP

## Reading the data
life = read.csv("/Users/karlacornejoargueta/Downloads/Life Expectancy Data 2.csv")
## loading necessary packages
library(ggplot2)
library(dplyr)
library(knitr)
library(ggpubr)
library(psych)
library(leaps)
library(car)
library(corrplot)
library(gridExtra)
library(MASS)
library(tidyverse)
library(caret)
library(nnet)
library(glmnet)
library(tree)
library(randomForest)
library(gbm)
library(reshape2)
library(reticulate)


## DATA MANIPULATION

# check data
table(is.na(life))
sapply(life, class)
colnames(life)
nrow(life)
ncol(life)

# rename columns
colnames(life) = c("Country", "Year", "Status", "LifeExp", "AdultMort",
                   "InfantDeath", "Alcohol", "Perc.Expend", 'Hepatitis.B',
                   "Measles", "BMI", "Deaths.0.4", "Polio", "TotalExpend",
                   "Diphtheria", "HIV.AIDS", 'GDP', 'Population', "Thinness.10.19",
                   "Thinness.5.9", 'Income', 'Schooling')
# change integer columns to numeric (R is particular about class types)
life$Year = as.numeric(life$Year)
life$AdultMort = as.numeric(life$AdultMort)
life$InfantDeath = as.numeric(life$InfantDeath)
life$Hepatitis.B = as.numeric(life$Hepatitis.B)
life$Measles = as.numeric(life$Measles)
life$Deaths.0.4 = as.numeric(life$Deaths.0.4)
life$Polio = as.numeric(life$Polio)
life$Diphtheria = as.numeric(life$Diphtheria)

# table of how many NAs per column
for (i in 1:ncol(life)){
  print(colnames(life)[i])
  print(table(is.na(life[,i])))
}

# remove NA columns: hepatitis B (19%) (9), GDP (15%) (17), population (22%) (18), income composition of
## resources (6% but removed entire countries) (21), schooling (5% but removed entire countries) (22)
# did this to remove most NAs without removing entire countries
rem = life[,-c(9, 17, 18, 21, 22)]
rem[!complete.cases(rem), ] # only removes a few cases and not entire countries now
crem = rem[complete.cases(rem), ]

# standardize
stan = crem
stan[, 4:17] = sapply(stan[, 4:17], scale)
```

```r
# codify country and status
codified = crem
codified$Country = as.numeric(as.factor(codified$Country))
codified$Status = as.numeric(as.factor(codified$Status))

codified_stan = stan
codified_stan$Country = as.numeric(as.factor(codified_stan$Country))
codified_stan$Status = as.numeric(as.factor(codified_stan$Status))
## EDA

## summary stats
kable(describe(crem[, 4:17]), fast = T)

## correlation matrix (include as figure 4)
c <- (ggplot(melt(round(cor(codified), 2)), aes(x = Var1, y = Var2, fill = value)) +
        geom_tile()+
        labs(x = '', y = '', fill = 'Correlation\nCoefficient', title = 'Correlation Heatmap',
             caption = 'Figure 4. Correlation heatmap of all combinations of all variables and
       their correlation coefficients')+
        theme_bw()+
        scale_fill_gradient2(low = "blue", high = "red", mid = "white", midpoint = 0, limit = c(-1,1),
                             space = "Lab")+
        geom_text(aes(Var2, Var1, label = value), color = "black", size = 2))

c + theme(axis.text.x = element_text(angle = 90))

## all qqplots (standardized)
for (i in 4:ncol(stan)){
  qqnorm(stan[,i], main = colnames(stan)[i])
  qqline(stan[,i])
}

## example of bad QQplots: Polio and HIV/AIDS (include as Figure 3)
qqnorm(stan[, 'HIV.AIDS'], main = 'Normal Q-Q Plot for HIV/AIDS')
qqline(stan[, 'HIV.AIDS'])

## all scatterplots
for (i in 1:ncol(codified)){
  plot(codified[,i], main = colnames(codified)[i], ylab = colnames(codified)[i])
}


## scatterplot examples: life expectancy, adult mortality, total expend, measles (include as figure 1)
par(mfrow=c(1,2))
plot(codified[,'LifeExp'], main = 'Figure 1: Life Expectancy', ylab = 'Life Expectancy',
     cex.main = 0.8)

## scatterplot examples for one year (include as figure 1)
plot(codified[codified$Year == 2005, 'LifeExp'], main = 'Life Expectancy in 2005',
     ylab = 'Life Expectancy', cex.main = 0.8)
## example of bad QQplots: Polio and HIV/AIDS (include as Figure 2)
qqnorm(stan[, 'HIV.AIDS'], main = 'Figure 2: Normal Q-Q Plot for HIV/AIDS', cex.main = 0.8)
qqline(stan[, 'HIV.AIDS'])
## correlation matrix (include as figure 3)
c <- (ggplot(melt(round(cor(codified), 2)), aes(x = Var1, y = Var2, fill = value)) +
        geom_tile()+
        labs(x = '', y = '', fill = 'Correlation\nCoefficient', title = 'Correlation Heatmap',
             caption = 'Figure 3. Correlation heatmap of all combinations of all variables and
       their correlation coefficients')+
        theme_bw()+
        scale_fill_gradient2(low = "blue", high = "red", mid = "white", midpoint = 0, limit = c(-1,1),
                             space = "Lab")+
        geom_text(aes(Var2, Var1, label = value), color = "black", size = 2))

c + theme(axis.text.x = element_text(angle = 90))
## VARIABLE SELECTION: BIC, ADJUSTED R-SQUARE, CP
```

```r
## AIC/BIC selection (include in report as figure 5)
## forward
forward = regsubsets(LifeExp ~ ., data = codified_stan, method = "forward", nvmax = 16)
forwardsum = summary(forward)
which.max(forwardsum$adjr2)
which.min(forwardsum$bic)
which.min(forwardsum$cp)

## backward
backward = regsubsets(LifeExp ~ ., data = codified_stan, method = "backward", nvmax = 16)
backwardsum = summary(backward)
which.max(backwardsum$adjr2)
which.min(backwardsum$bic)
which.min(backwardsum$cp)
## plot function
crit_plot = function(fwdcr, bwdcr, yl, legpos, m){
  plot(fwdcr, type = 'l', xlab = 'Number of Variables',
       ylab = yl, col = 'red', main = m)
  lines(bwdcr, type = 'l', col = 'blue')
  points(1:13, fwdcr[1:13], col = "black", cex = 0.5, pch = 20)
  points(15:16, fwdcr[15:16], col = "black", cex = 0.5, pch = 20)
  points(1:13, bwdcr[1:13], col = "black", cex = 0.5, pch = 20)
  points(15:16, bwdcr[15:16], col = "black", cex = 0.5, pch = 20)
  points(14, fwdcr[14], col = "black", cex = 2.5, pch = 8)
  legend(legpos, legend = c("Forward Selection", "Backward Selection"),
         col = c("red", "blue"), lty = 1)
}
par(mfrow=c(1,3))

## BIC plot
crit_plot(fwdcr = forwardsum$bic, bwdcr = backwardsum$bic, yl = 'BIC',
          legpos = 'topright', m = '')

## cp plot
crit_plot(fwdcr = forwardsum$cp, bwdcr = backwardsum$cp, yl = "Mallow's Cp",
          legpos = 'topright', m = "Figure 4. Feature Selection")

## adjusted r squared plot
crit_plot(fwdcr = forwardsum$adjr2, bwdcr = backwardsum$adjr2, yl = 'Adjusted R-Squared',
          legpos = 'bottomright', m = '')
## best model (include in report as table 1)
kable(coef(backward, 14), col.names = 'Coefficient Estimate')

## dataframes based on the variables selected by backward selection
bestdf_st = stan[, -c(13, 17)]
bestdf = crem[, -c(13, 17)]
bestdf_cod = codified[, -c(13, 17)]
bestdf_st_cod = codified_stan[, -c(13, 17)]
## RIDGE AND LASSO

x = as.matrix(codified_stan[, -c(1, 3)]) # Exclude Country and Status columns
y = codified_stan$LifeExp

## Ridge Regression
ridge_model = cv.glmnet(x, y, alpha = 0, standardize = TRUE)

## Best lambda value for ridge regression
best_lambda_ridge = ridge_model$lambda.min
best_lambda_ridge

## Coefficients for the best model
ridge_coef = coef(ridge_model, s = best_lambda_ridge)
print(ridge_coef)

## Plotting the cross-validation curve for ridge regression
plot(ridge_model)
```

```r
summary(ridge_model)

## Lasso Regression
lasso_model = cv.glmnet(x, y, alpha = 1, standardize = TRUE)

## Best lambda value for lasso regression
best_lambda_lasso = lasso_model$lambda.min
best_lambda_lasso

## Coefficients for the best model
lasso_coef = coef(lasso_model, s = best_lambda_lasso)
print(lasso_coef)

## Plotting the cross-validation curve for lasso regression
plot(lasso_model)
summary(lasso_model)

## best-fit variables selected by backward selection
best_vars = c("Country", "Year", "Status", "AdultMort", "InfantDeath", "Alcohol",
              "Perc.Expend", "Measles", "BMI", "Deaths.0.4", "Polio", "TotalExpend",
              "Diphtheria", "HIV.AIDS")

x_best = as.matrix(codified_stan[, best_vars])
y_best = codified_stan$LifeExp

## Ridge Regression using the best-fit variables
ridge_model_best = cv.glmnet(x_best, y_best, alpha = 0, standardize = TRUE)

## Best lambda value for ridge regression
best_lambda_ridge = ridge_model_best$lambda.min
best_lambda_ridge

## Coefficients for the best model
ridge_coef_best = coef(ridge_model_best, s = best_lambda_ridge)
print(ridge_coef_best)

## Plotting the cross-validation curve for ridge regression
plot(ridge_model_best)
summary(ridge_model_best)

## Lasso Regression using the best-fit variables
lasso_model_best = cv.glmnet(x_best, y_best, alpha = 1, standardize = TRUE)

## Best lambda value for lasso regression
best_lambda_lasso = lasso_model_best$lambda.min
best_lambda_lasso

## Coefficients for the best model
lasso_coef_best = coef(lasso_model_best, s = best_lambda_lasso)
print(lasso_coef_best)

## Plotting the cross-validation curve for lasso regression
plot(lasso_model_best)

summary(lasso_model_best)
## arranging the plots so they can be displayed
par(mfrow=c(1,2))
plot(ridge_model_best, main = "Figure 5: Ridge and Lasso", cex.main = 0.6,cex.axis = 0.8,
     cex.lab = 0.8, cex.sub = 0.8)

plot(lasso_model_best, cex.main = 0.6,cex.axis = 0.8, cex.lab = 0.8, cex.sub = 0.8)
## CLASSIFICATION

## Redoing the previous standardization because of the addition of categories

## sets up sequence by which life expectancy is categorized, each by 10
breaks = seq(0,100, by = 10)
```

```
crem$LifeExpCat = cut(crem$LifeExp, breaks = breaks,
                      labels = c("0-10", "10-20", "20-30", "30-40", "40-50", "50-60", "60-70",
                                 "70-80", "80-90", "90-100"))

## standardizes data
stan = crem
stan[, 4:17] = sapply(stan[, 4:17], scale)
codified = crem
codified$Country = as.numeric(as.factor(codified$Country))
codified$Status = as.numeric(as.factor(codified$Status))

## codifies all the categorical values into factors
codified_stan = stan
codified_stan$Country = as.numeric(as.factor(codified_stan$Country))
codified_stan$Status = as.numeric(as.factor(codified_stan$Status))
codified_stan$LifeExpCat = as.numeric(as.factor(codified_stan$LifeExpCat))

## separates data into a training set and validation test
## training data is from the years 2000-2011
## test data is from 2012-2015
train = stan$Year<2012
life_test = stan[!train,]
life_train = stan[train,]
life_train$StatusBinary = ifelse(life_train$Status == "Developed", 1,0)
life_test$StatusBinary = ifelse(life_test$Status == "Developed", 1,0)
## logistic regression
## using glm function to fit logistic regression model 1
## then testing accuracy using the test data
le_fit1 = glm(StatusBinary ~ LifeExp, data = life_train, family = 'binomial')

le_fit1_prob = predict(le_fit1, life_test, type = "response")
le_fit1_pred = rep("Developing", length(le_fit1_prob))
le_fit1_pred[le_fit1_prob > 0.5] = "Developed"

## creates table of accuracy
kable(table(le_fit1_pred, life_test$Status))
mean(le_fit1_pred == life_test$Status)

##creates plots of the glm method
p2 = ggplot(life_test, aes(x=LifeExp, y = StatusBinary)) + geom_point(color = 'darkgrey') +
  stat_smooth(method="glm", color="skyblue", se=FALSE,method.args = list(family=binomial)) +
  ggtitle("Test Data") + xlab('Life Expectancy') + ylab('Status')
p1 = ggplot(life_train, aes(x=LifeExp, y=StatusBinary)) + geom_point(color = 'darkgrey') +
  stat_smooth(method="glm", color="skyblue", se=FALSE, method.args = list(family=binomial)) +
  ggtitle("Training Data") + xlab('Life Expectancy') + ylab('Status')

## arrange plots into one visualization
grid.arrange(p1,p2)

## using the glm function to fit logistic model 2
le_fit2 = glm(StatusBinary ~ LifeExp + AdultMort + InfantDeath + Alcohol + Perc.Expend + Measles +
                BMI + Deaths.0.4 + Polio + TotalExpend + Diphtheria + HIV.AIDS + Thinness.10.19 +
                Thinness.5.9, data = life_train, family = 'binomial')

le_fit2_prob = predict(le_fit2, life_test, type = "response")
le_fit2_pred = rep("Developing", length(le_fit2_prob))
le_fit2_pred[le_fit2_prob > 0.5] = "Developed"
mean(le_fit2_pred == life_test$Status)

## using the glm function to fit logistic model 3
le_fit3 = glm(StatusBinary ~ LifeExp + AdultMort + InfantDeath + Alcohol + Perc.Expend + Measles +
                BMI + Polio  + Diphtheria + HIV.AIDS + Thinness.10.19, data = life_train, family = 'binomial')

le_fit3_prob = predict(le_fit3, life_test, type = "response")
le_fit3_pred = rep("Developing", length(le_fit3_prob))
le_fit3_pred[le_fit3_prob > 0.5] = "Developed"
mean(le_fit3_pred == life_test$Status)
```

```r
## using the glm function to fit logistic model 4
le_fit4 = glm(StatusBinary ~  AdultMort + InfantDeath + Alcohol + Perc.Expend + Measles + BMI +
                Polio + Diphtheria + HIV.AIDS + Thinness.10.19, data = life_train, family = 'binomial')

le_fit4_prob = predict(le_fit4, life_test, type = "response")
le_fit4_pred = rep("Developing", length(le_fit4_prob))
le_fit4_pred[le_fit4_prob > 0.5] = "Developed"
mean(le_fit4_pred == life_test$Status)

## looks at the multicollinearity of the model
vif(le_fit2)
vif(le_fit3)
vif(le_fit4)
## LDA
## fitting linear discriminant analysis model 1
lda_fit1 = lda(Status ~ LifeExp, data = life_train, family = 'binomial')
lda_fit1_pred = predict(lda_fit1, life_test)
kable(table(lda_fit1_pred$class, life_test$Status), caption = 'Model 1')
mean(lda_fit1_pred$class == life_test$Status)

## fitting linear discriminant analysis model 2
lda_fit2 = lda(Status ~ LifeExp + AdultMort + InfantDeath + Alcohol + Perc.Expend + Measles +
                 BMI + Deaths.0.4 + TotalExpend + Diphtheria  + Thinness.10.19 + Thinness.5.9,
               data = life_train, family = 'binomial')

lda_fit2_pred = predict(lda_fit2, life_test)
kable(table(lda_fit2_pred$class, life_test$Status), caption = 'Model 2')
mean(lda_fit2_pred$class == life_test$Status)

## fitting linear discriminant analysis model 3
lda_fit3 = lda(Status ~ LifeExp + AdultMort + InfantDeath + Alcohol + Perc.Expend + Measles +
                 BMI  + Diphtheria  + Thinness.10.19, data = life_train)

lda_fit3_pred = predict(lda_fit3, life_test)
kable(table(lda_fit3_pred$class, life_test$Status), caption = 'Model 3')
mean(lda_fit3_pred$class == life_test$Status)

## fitting linear discriminant analysis model 4
lda_fit4 = lda(Status ~ AdultMort + InfantDeath + Alcohol + Perc.Expend + Measles + BMI  +
                 Diphtheria + Thinness.10.19, data = life_train, family = 'binomial')

lda_fit4_pred = predict(lda_fit4, life_test)
kable(table(lda_fit4_pred$class, life_test$Status), caption = 'Model 4')
mean(lda_fit4_pred$class == life_test$Status)
## QDA

## fitting quadratic discriminant analysis model 1
qda_fit1 = qda(Status ~ LifeExp, data = life_train)
qda_fit1_pred = predict(qda_fit1, life_test)
kable(table(qda_fit1_pred$class, life_test$Status), caption = "Model 1")
mean(qda_fit1_pred$class == life_test$Status)

## fitting quadratic discriminant analysis model 2
qda_fit2 = qda(Status ~ LifeExp + AdultMort + InfantDeath + Alcohol + Perc.Expend + Measles +
                 BMI + Deaths.0.4 + TotalExpend + Diphtheria + Thinness.10.19 + Thinness.5.9,
               data = life_train)

qda_fit2_pred = predict(qda_fit2, life_test)
kable(table(qda_fit2_pred$class, life_test$Status), caption = 'Model 2')
mean(qda_fit2_pred$class == life_test$Status)

## fitting quadratic discriminant analysis model 3
qda_fit3 = qda(Status ~ LifeExp + AdultMort + InfantDeath + Alcohol + Perc.Expend + Measles +
                 BMI  + Diphtheria  + Thinness.10.19, data = life_train)

qda_fit3_pred = predict(qda_fit3, life_test)
```

```r
kable(table(qda_fit3_pred$class, life_test$Status), caption = 'Model 3')
mean(qda_fit3_pred$class == life_test$Status)

## fitting quadratic discriminant analysis model 4

qda_fit4 = qda(Status ~ AdultMort + InfantDeath + Alcohol + Perc.Expend + Measles + BMI  +
                 Diphtheria  + Thinness.10.19, data = life_train)

qda_fit4_pred = predict(qda_fit4, life_test)
kable(table(qda_fit4_pred$class, life_test$Status), caption = 'Model 4')
mean(qda_fit4_pred$class == life_test$Status)
## Separating the codified data into training and test sets
## for multinomial regression
train = codified_stan$Year<2012
lifecod_test = codified_stan[!train,]
lifecod_train = codified_stan[train,]

## Multinomial Regression

## using multinomial regression to predict countries given life expectancy
multinom_co = multinom(Country ~ LifeExp, data = lifecod_train, limit = 50)
multinom_co_prob = predict(multinom_co, lifecod_test)

mean(multinom_co_prob == lifecod_test$Country)

## splitting data into train and test sets excluding life expectancy
minus_LE_cod = codified_stan[,-4]
train = minus_LE_cod$Year<2012
lifecod_test = minus_LE_cod[!train,]
lifecod_train = minus_LE_cod[train,]

## using multinomial regression to predict life expectancy categories
## using all variables
multinom_le1 = multinom(LifeExpCat ~ ., data = lifecod_train)
multinom_le1_prob = predict(multinom_le1, lifecod_test)

kable(table(multinom_le1_prob, lifecod_test$LifeExpCat), caption = 'Model 1')
mean(multinom_le1_prob == lifecod_test$LifeExpCat)

## training the data using only the best variables and excluding
## life expectancy
minus_LE_cod = codified_stan[,-c(4,13,17)]
train = minus_LE_cod$Year<2012
lifecod_test = minus_LE_cod[!train,]
lifecod_train = minus_LE_cod[train,]

## fitting multinomial regression on life expectancy with selected variables
multinom_le2 = multinom(LifeExpCat ~ ., data = lifecod_train)
multinom_le2_prob = predict(multinom_le2, lifecod_test)

kable(table(multinom_le2_prob, lifecod_test$LifeExpCat), caption = 'Model 2')
mean(multinom_le2_prob == lifecod_test$LifeExpCat)

## making the predictions into a csv file for interactive map
prediction_df = crem
prediction_df$LifeExpMMod = predict(multinom_le1, codified_stan)
prediction_df$CodLifeExpCat = codified_stan$LifeExpCat
write.csv(prediction_df,"~/Downloads/life.csv", row.names = FALSE)
## the packages necessary to run the python code ahead
reticulate::py_install("folium")
reticulate::py_install("pandas")

## Creating world map using python
## importing necessary modules

import folium
import pandas as pd
```

```
## importing the the two csv files, one with the data the other with the coordinates
life = pd.read_csv('/Users/karlacornejoargueta/Downloads/life.csv')
locations = pd.read_csv('/Users/karlacornejoargueta/Downloads/\
world_country_and_usa_states_latitude_and_longitude_values.csv')

## processing data so it only contains coordinates, and the columns can be read
life['Country'] = life['Country'].str.strip()
locations = locations.drop('usa_state_code', axis =1 )
locations = locations.drop('usa_state', axis =1 )
locations = locations.drop('usa_state_latitude', axis =1 )
locations = locations.drop('usa_state_longitude', axis =1 )

## joining the data using pd by the countries
## adds the two data sets so all the data is connected to the coordinates of each country
complete = life.join(locations.set_index('country'), on='Country')
complete = complete.dropna(subset=['latitude', 'longitude'])
## using only year 2014 for the predictions
complete_2014 = complete[complete['Year'] == 2014]

## initializes map
m = folium.Map(zoom_start=2)
## creating for loop so that for each row in the data set, it adds a marker to the map
## with each marker including the true category, the true category code, and the predicted
## category code
for _, row in complete_2014.iterrows():
    popup_text = (f"Life Expectancy Category: {row['LifeExpCat']}<br>"
                  f"Life Expectancy Category Codified: {row['CodLifeExpCat']}<br>"
                  f"Predicted Life Expectancy Category Codified: {row['LifeExpMMod']}")
    folium.Marker(
        location=[row['latitude'], row['longitude']],
        popup=folium.Popup(popup_text, max_width=200)
    ).add_to(m)


## saves map into an html file
m.save('life_map.html')
## PREDICTIVE MODELS FOR LIFE EXPECTANCY

## splitting standardized data into training and test set
set.seed(1)
life_train_T = sample(1:nrow(bestdf_cod), nrow(bestdf_cod)/2)
life_test_T = bestdf_cod[-life_train_T, ]
#Regression Trees
set.seed(1)
tree_life = tree(LifeExp ~ ., data = bestdf_cod, subset = life_train_T)
plot(tree_life, main = "Regression Tree")
text(tree_life, pretty = 0)
summary(tree_life)

yhat_tree_life = predict(tree_life, newdata = life_test_T)
test_life_true = bestdf_cod[-life_train_T, "LifeExp"]
test_life_true = as.data.frame(test_life_true)
plot(yhat_tree_life, test_life_true$LifeExp, xlab = "Yhat", ylab = "True Test Life Expectancy",
     main = "Regression Tree Predicted vs. True Test Life Expectancy")
abline(0,1)

mean((yhat_tree_life - test_life_true$LifeExp)^2)
#Pruning Regression Trees and Cross validation
cv_tree_life = cv.tree(tree_life)
plot(cv_tree_life$size, cv_tree_life$dev, type = "b", xlab = "Branches", ylab = "Dev",
     main = "Figure 6: Regression Tree Branch vs Dev")
pruned_tree_life = prune.tree(tree_life, best = 8)
cv_yhat_tree_life = predict(pruned_tree_life, newdata = life_test_T)
plot(cv_yhat_tree_life, test_life_true$LifeExp)
abline(0,1)
mean((cv_yhat_tree_life - test_life_true$LifeExp)^2)
```

```r
#Bagging
set.seed(1)
life_bag = randomForest(LifeExp ~., data = bestdf_cod, subset = life_train_T, importance = TRUE)
#,mtry = 12, ntree = 25)#recall default is 500
yhat_bag = predict(life_bag, newdata = life_test_T)
plot(yhat_bag, test_life_true$LifeExp, xlab = "Yhat", ylab = "Test True Life Expectancy",
     main = "Bagging Predicted vs Actual Scatterplot")
abline(0, 1)
mean((yhat_bag - test_life_true$LifeExp)^2)
importance(life_bag)
#Random Forest
set.seed(1)
rf_life = randomForest(LifeExp ~., data = bestdf_cod, subset = life_train_T,
                       mtry = 6, importance = TRUE)
yhat_rf = predict(rf_life, newdata = life_test_T)
plot(yhat_rf, test_life_true$LifeExp, xlab = "Yhat", ylab = "Test True Life Expectancy",
     main = "Random Forest Predicted vs Actual Scatterplot")
mean((yhat_rf - test_life_true$LifeExp)^2)
importance(rf_life)
#Boosting
set.seed(1)
life_train_T2 = data.frame(life_train_T)
boost_life = gbm(LifeExp ~ ., data = bestdf_cod[life_train_T ,],#bestdf_cod[life_train_T ,] life_train_T
                 distribution = "gaussian", n.trees = 1000,
                 interaction.depth = 4)
summary(boost_life)
lifeTestTrue = life_test_T$LifeExp
yhat_boost = predict(boost_life,
    newdata = life_test_T, n.trees = 1000)
mean((yhat_boost - lifeTestTrue)^2)
```