My professional journey includes working as Lab Manager at Texas A & M University for 3+ years and earning two masters' in science one on of them in Computational Biology from CMU. After completing my second master, I took a break due to medical reasons. During that time, I completed a Certification in Statistics from Texas A&M University, and several data science Coursera courses. Also, because I am trilingual, I started working as interpreter. As I am returning to my fields of study and restarting my career, I thought it would be helpful to share my data science skills by researching and completing this project. If interested in learning more, please contact me: bsanchez@alumni.cmu.edu.

Image from shutterstock by tynyuk

Introduction

This paper shows how do Linear Regression Analysis using three different algorithms: Normal Equations, Ridge, and Lasso. It includes a step by step description, an analysis and interpretation of results. The python code is included. These approaches are useful when one wants to know which features are important for the results and data is limited.

To know the data set an initial analysis should be done. Going directly to a higher-level analysis such as Machine Learning is not always appropriate. One such popular Machine Learning algorithm is Deep Learning.

Deep learning is usually used for speech recognition, image identification and to make predictions from large datasets to find intricate relationships. Interpretability of the results is not always easy because we don't know always know which features were used to obtain the final model and the results.

Not all data sets that a researcher works on are large and have intricate relationships between the dependent variables. Important information can be obtained from simple experiments.

This project is based on a KDnuggets post on 14 projects that can be done to improve and strengthen data science skills. I choose the World Health Organization (WHO) project. The dataset is in Kaggle, it is a subset from the WHO dataset and has 22 variables. The objective on the KDnuggets project was to examine factors that have a positive or negative effect on life expectancy. Thanks to the courageous person who posted the initial analysis. I will be expanding on the project to see which factors best describe a linear model in predicting life expectancy by utilizing regression methods.

The python packages used were pandas, NumPy, Scikit-Learn, statsmodels, AWOC, seaborn and yellowbrick.

Data Analysis

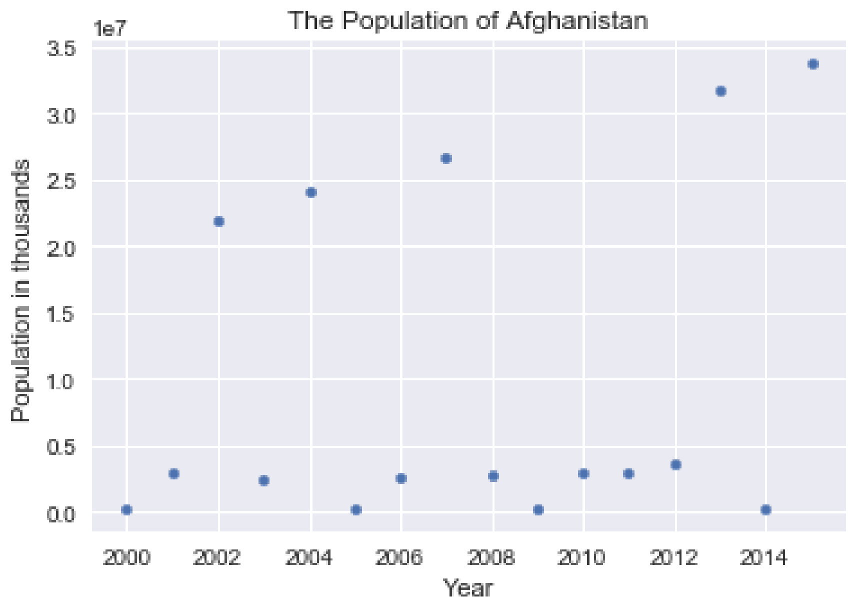
The data file was read into a data frame, it had 2983 rows and 22 variables. The explanation of the variables from Kaggle was incomplete or wrong for some variables.

Specifically, there are four variables that need a better definition: *Adult mortality, Under-five deaths*, *alcohol* and *infant deaths*. *Adult mortality* is the number of deaths in a population of 1000 between the ages of 15-60. *Under-five deaths* are deaths occurring under age 5 in a population of 1000. *Alcohol* is how much of it is consumed per capita (15+ years) over a calendar year in a country, in liters of pure alcohol. *Infant deaths* are how many infants died between birth and age 1 per 1000 live births.

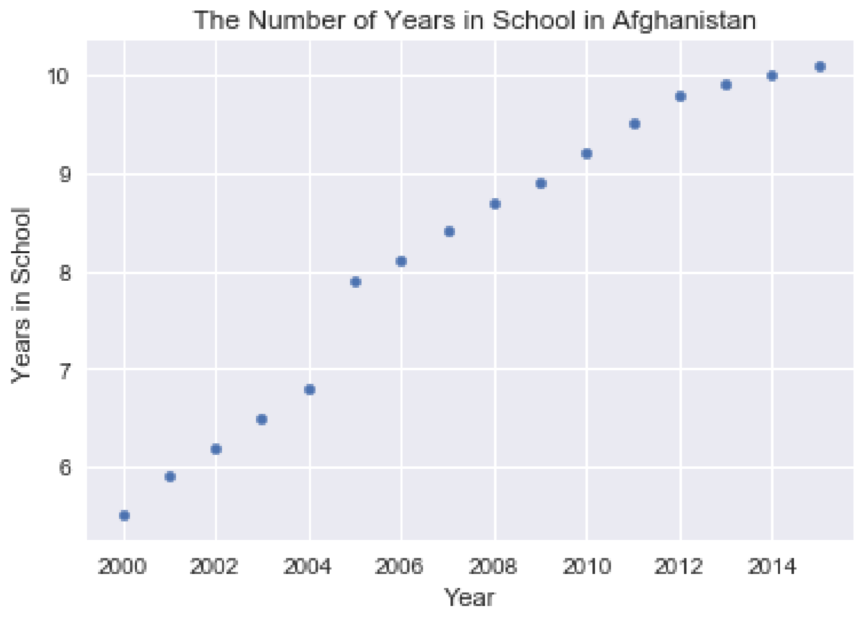
The information found in the Kaggle histograms for the initial project is not easy to decipher because the information for many of the variables is compressed together. It is hard to see the underlying distribution of the variables. There are better plots that convey the information in this data frame such as scatter plots and box plots.

To get an idea of any trends in the dataset, I randomly selected three developing countries and three developed countries: Afghanistan, Romania, Mexico, France, Belgium, and United States, and made scatter plots on the variables*: Adult mortality*, *Infant deaths*, *Alcohol consumption*, and *Schooling*.

Some of the scatter plots showed that the data was inaccurate. For example, the scatter plot below shows that the population in Afghanistan widely fluctuates from year to year. *Population* cannot fluctuate that widely from year to year.



The scatter plot below shows that the number of years in school has risen over the years in Afghanistan



In the original dataset, some country names were long, and I simplified them to shorter names. For example, Venezuela (Bolivarian Republic of) was changed to Venezuela. Also, the variable *status*, was changed from a character variable to a numeric value: *Developed* became a value of 1 and *Developing* a value of 0.

To find trends in the data, I decided to add the variable *Continent* to the original data frame. I used the Python AWOC package that uses the data/world.json file to create a data frame that had the name of the *Country* with its accompanying *Continent*. The data frame created was called df. In the code below, the line `world.get_countries_list(), columns=['Country'])` obtains a list of all the *Countries*, and data. `world.get_country_data()` gets *Continent*. The second data frame had information on 240 countries, more than the ones in the original Kaggle dataset. This data frame was then merged with the original data frame (lifeexp) to add the column *Continent*. The shape command was used to verify the number of rows and columns in the data frame.

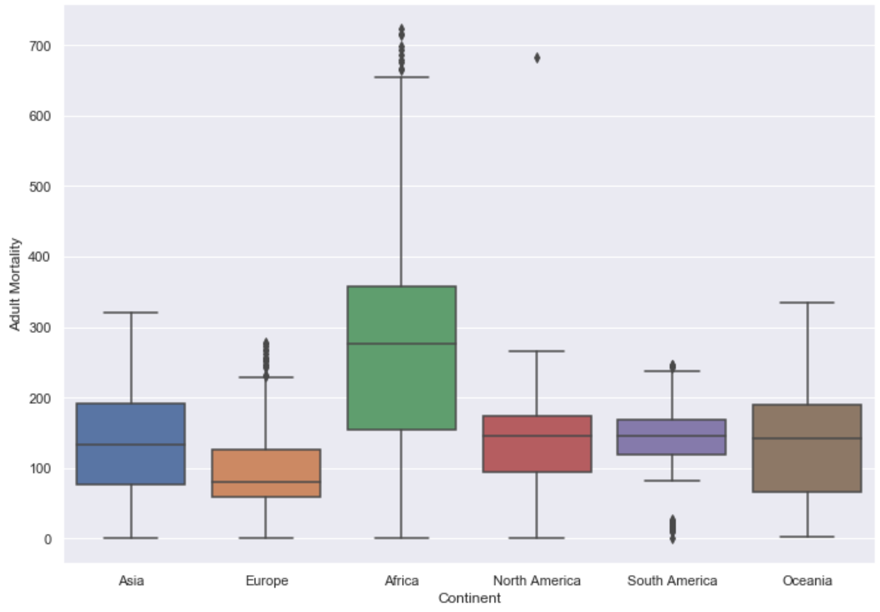
Code for adding continent:

```
import awoc
import pandas as pd
world = awoc.AWOC()


#Create the dataframe

df = pd.DataFrame(world.get_countries_list(), columns=['Country'])
continent = []
for r in df.iterrows():
        country = r[1]['Country']
        continent.append(world.get_country_data(country)['Continent
Name'])

df['Continent']=continent

#Merging the dataframes
lifeexp_continent = pd.merge(lifeexp, df, on = 'Country', how = 'left')
lifeexp_continent.shape
```

The data set had NaN values for some of the variables. Since life expectancy was the dependent variable, any missing countries that had life expectancy missing were removed from the data set. Other variable's NaN missing values were replaced by their mean value.

With the seaborn package, I created boxplots using the mean values of *Adult Mortality, Infant Deaths, Schooling, Population*, and *Alcohol* consumption by *Continent*.

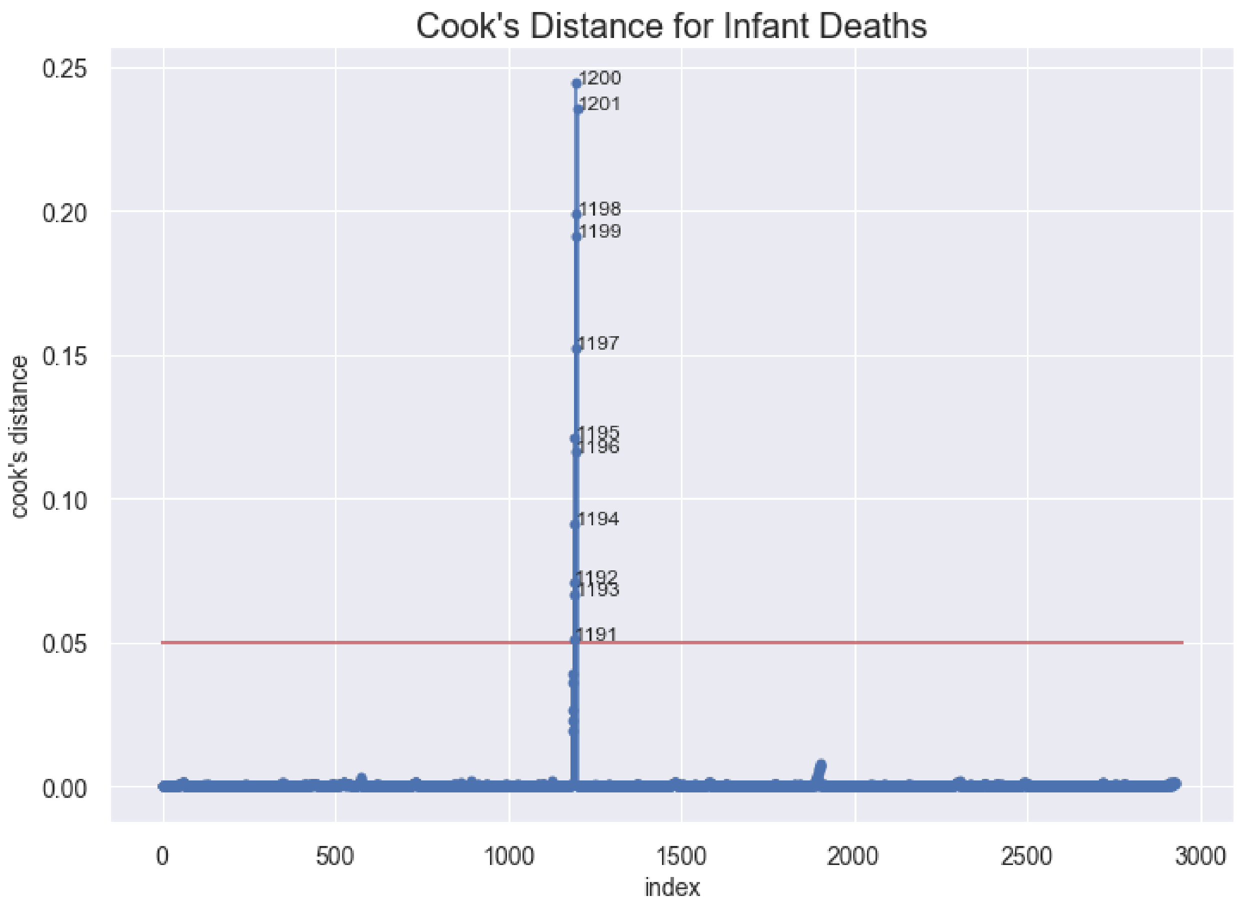
This boxplot shows the *Adult Mortality* by *Continent*.



Europe had the lowest *Adult Mortality* of all the continents and Africa had the highest. All other continents had around the same mean. The *Continent* with the *lowest infant mortality rate* was Europe followed by Oceania, the highest rate was in Asia followed by Africa. The *Continent* with the lowest *Schooling* was Africa and Europe had the highest. Asia and South America had the highest *Population* of all the *Continents*. The most *Alcohol* was drunk in Europe and in Asia the least.

Cook's distance analyzes any data points that are beyond a certain threshold, I used alpha = 0.05. Seventeen variables were tested to see if they had outliers. Some variables had outliers, for example: *Infant deaths* in India; *Percentage expenditure* in Luxemburg, Sweden and Switzerland; *Measles* in China; *Under-five deaths* in India; *HIV/AIDS* in Swaziland and Zimbabwe; *GDP* in Lithuania, and Luxemburg; and *Schooling* in Antigua and Barbuda, Bosnia and Herzegovina and Montenegro. Outliers were studied to see if they influenced the analysis of the data.

This plot shows that there were several outliers in *Infant Deaths*, the rest of the data points were below 0.05. All the outliers came from India.



Supervised vs unsupervised learning

Supervised learning uses data where the dependent variables are labeled. I used supervised learning to build a model that predicted life expectancy. Unsupervised learning uses data where the dependent variables do not have labels. Unsupervised learning searches for possible patterns/relationships in the data. In the future, I will use unsupervised learning to see if any significant clusters are formed.

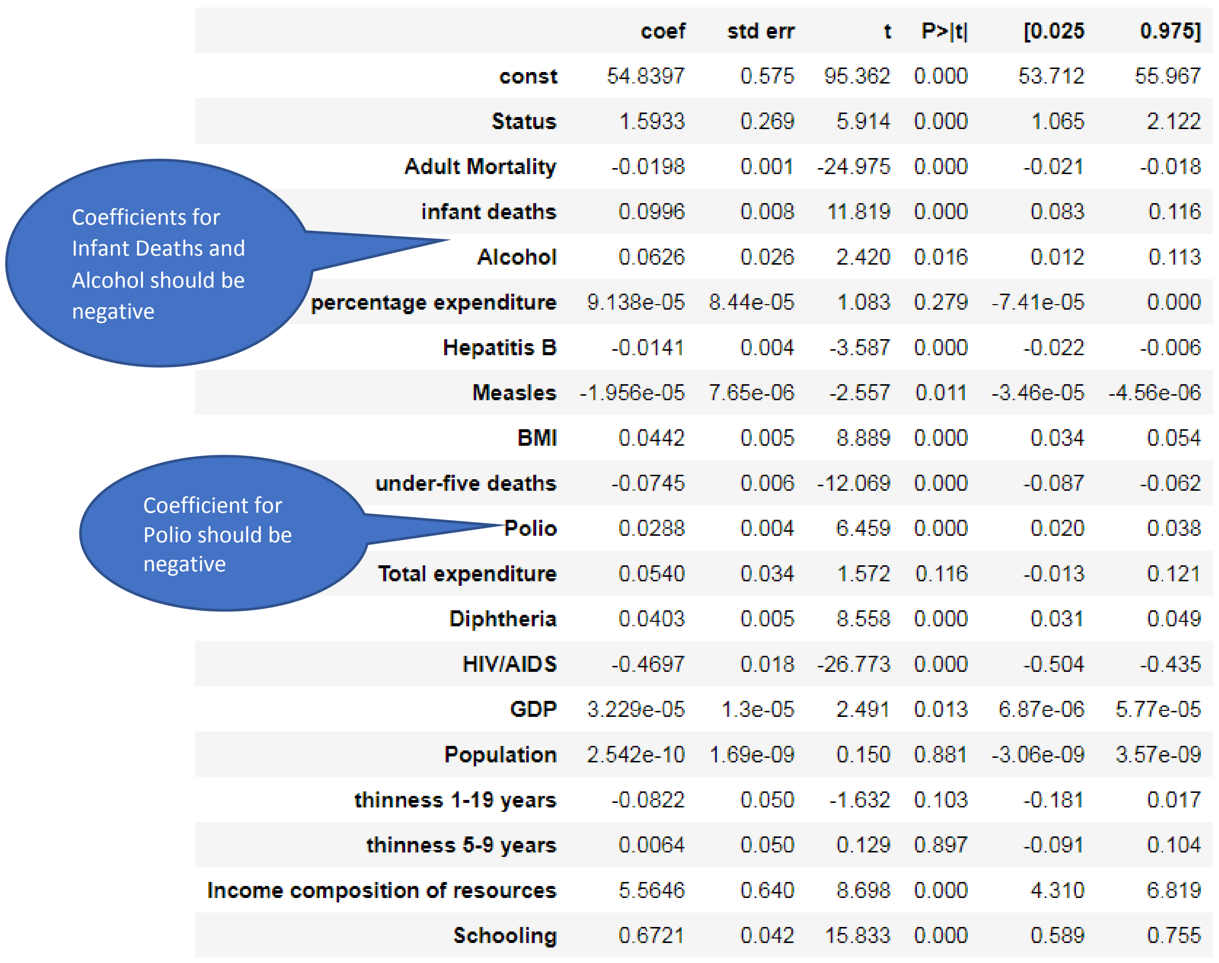
Regression Analysis

Regression analysis is done to study the relationship between independent variables and the dependent variable. The goal is to predict the dependent variable from the independent variables. In this instance, which variables best predict *Life expectancy*. I will start with linear regression the simplest type of regression and then use ridge and lasso regression to find the most important variables that predict *Life expectancy.* These methods go about choosing the independent variables in different manners. They both use a cost function to estimate the variables.

Linear Regression

A linear model must meet three assumptions: 1) there is a linear relationship between the independent variable and the dependent variable; 2) the independent variables are normally distributed and 3) the independent variables are independent from each other.

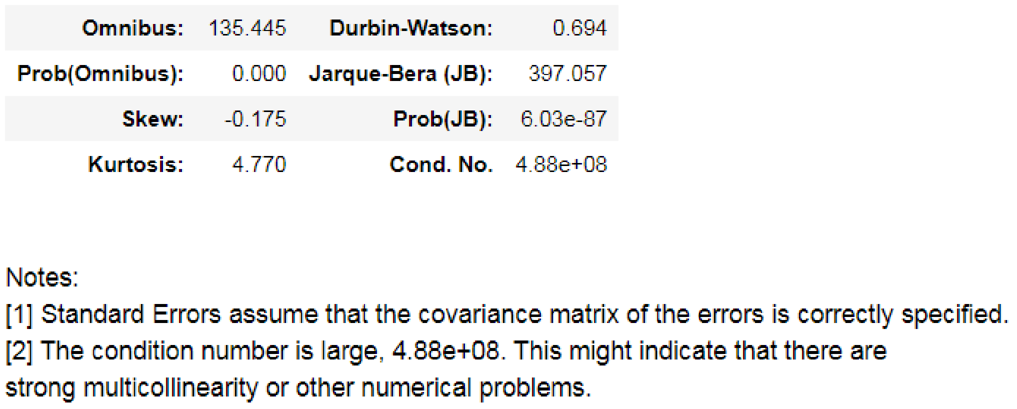
An initial regression analysis showed evidence of multicollinearity. Therefore, at least one of the above assumptions was violated. Some of the coefficients that were questionable were, for example, *Infant deaths*, *Alcohol* and *Polio*:



Coefficients for Infant Deaths and Alcohol should be negative

Coefficient for Polio should be negative

	coef	std err	t	P>|t|	[0.025	0.975]
const	54.8397	0.575	95.362	0.000	53.712	55.967
Status	1.5933	0.269	5.914	0.000	1.065	2.122
Adult Mortality	-0.0198	0.001	-24.975	0.000	-0.021	-0.018
infant deaths	0.0996	0.008	11.819	0.000	0.083	0.116
Alcohol	0.0626	0.026	2.420	0.016	0.012	0.113
percentage expenditure	9.138e-05	8.44e-05	1.083	0.279	-7.41e-05	0.000
Hepatitis B	-0.0141	0.004	-3.587	0.000	-0.022	-0.006
Measles	-1.956e-05	7.65e-06	-2.557	0.011	-3.46e-05	-4.56e-06
BMI	0.0442	0.005	8.889	0.000	0.034	0.054
under-five deaths	-0.0745	0.006	-12.069	0.000	-0.087	-0.062
Polio	0.0288	0.004	6.459	0.000	0.020	0.038
Total expenditure	0.0540	0.034	1.572	0.116	-0.013	0.121
Diphtheria	0.0403	0.005	8.558	0.000	0.031	0.049
HIV/AIDS	-0.4697	0.018	-26.773	0.000	-0.504	-0.435
GDP	3.229e-05	1.3e-05	2.491	0.013	6.87e-06	5.77e-05
Population	2.542e-10	1.69e-09	0.150	0.881	-3.06e-09	3.57e-09
thinness 1-19 years	-0.0822	0.050	-1.632	0.103	-0.181	0.017
thinness 5-9 years	0.0064	0.050	0.129	0.897	-0.091	0.104
Income composition of resources	5.5646	0.640	8.698	0.000	4.310	6.819
Schooling	0.6721	0.042	15.833	0.000	0.589	0.755



Omnibus:	135.445	Durbin-Watson:	0.694
Prob(Omnibus):	0.000	Jarque-Bera (JB):	397.057
Skew:	-0.175	Prob(JB):	6.03e-87
Kurtosis:	4.770	Cond. No.	4.88e+08

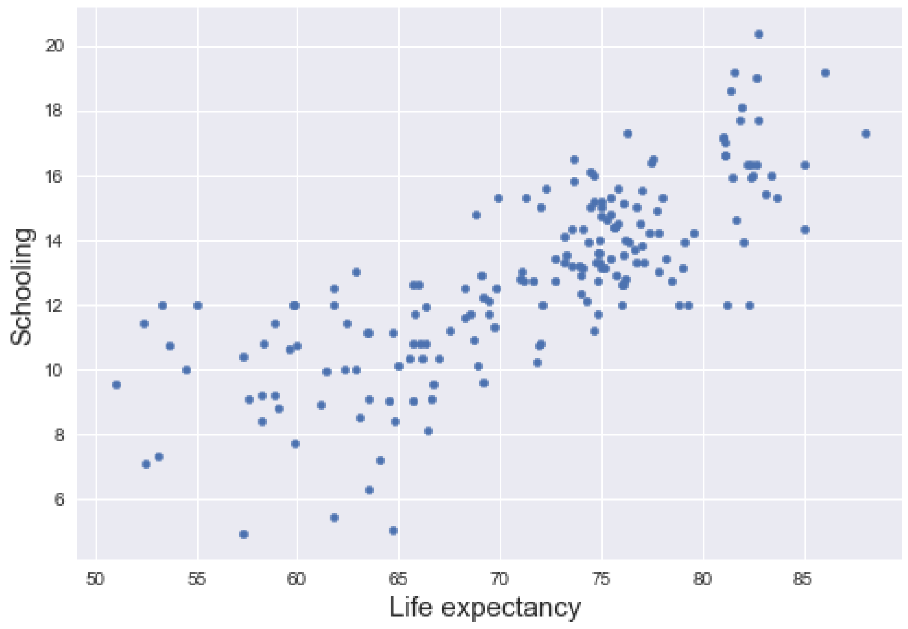
Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 4.88e+08. This might indicate that there are
strong multicollinearity or other numerical problems.

The tables above show that the coefficients for *Alcohol, Infant deaths and Polio are* positive when they should be negative and there are indications of multicollinearity or other numerical problems. These two things indicate that the p-values may not be accurate. Moreover, there are many variables that had zero within their confidence intervals. A VIF analysis to test for multicollinearity confirmed that there was moderate collinearity occurring between the variables. *Infant deaths* and *Under-five deaths* had severe multicollinearity. In a VIF analysis, values under 1 indicate that there is no multicollinearity between the variables. Values between 2 and 5 that there is moderate correlation. A value greater than 5, indicates severe correlation. *Adult mortality, Measles, HIV/AIDS*, and *Population* had values between 2 and 5. The rest of the variables had high VIF values but not as high as *Infant deaths* and *Under-five deaths*. I removed these two variables and repeated the VIF analysis, the results were that all the variables had moderate correlation.

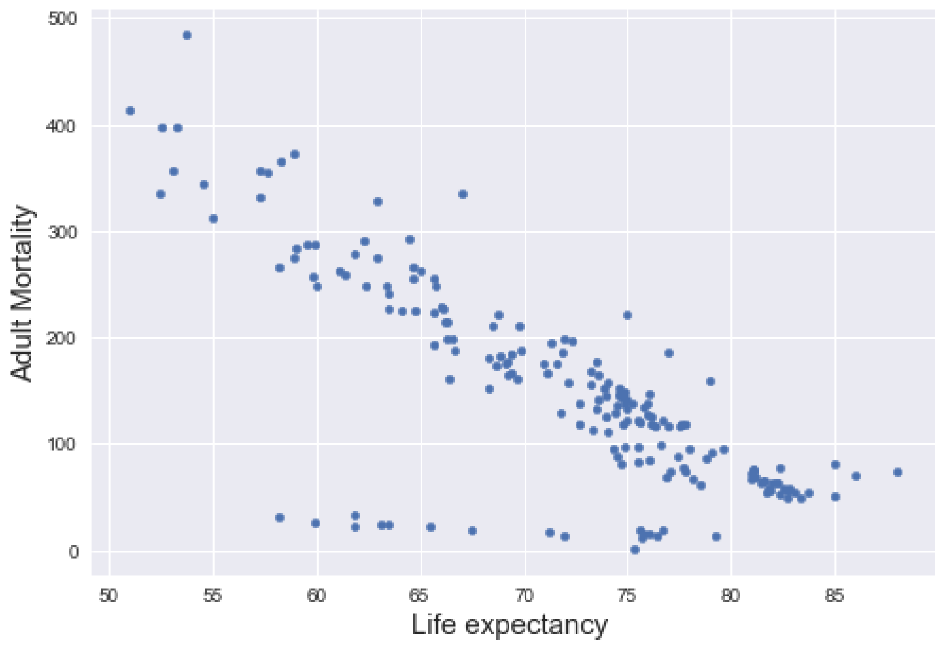
A correlation matrix was generated. Variables that had correlation coefficients greater than 0.5 (positive or negative) with *Life expectancy* were taken into consideration. These correlation coefficients were then seen if they were highly correlated with each other. If they were not, then the variables can be included in our first linear model. *Schooling* and *Adult mortality* were highly correlated with *life expectancy*. The two variables were not highly correlated with each other.

I used only the data for 2015 to search for trends. *Schooling* and *Income composition of resources* were found to have positive correlation with *life expectancy*. *Adult mortality* had a negative correlation with life expectancy.

This scatter plot shows a positive correlation between *Schooling* and *Life expectancy*



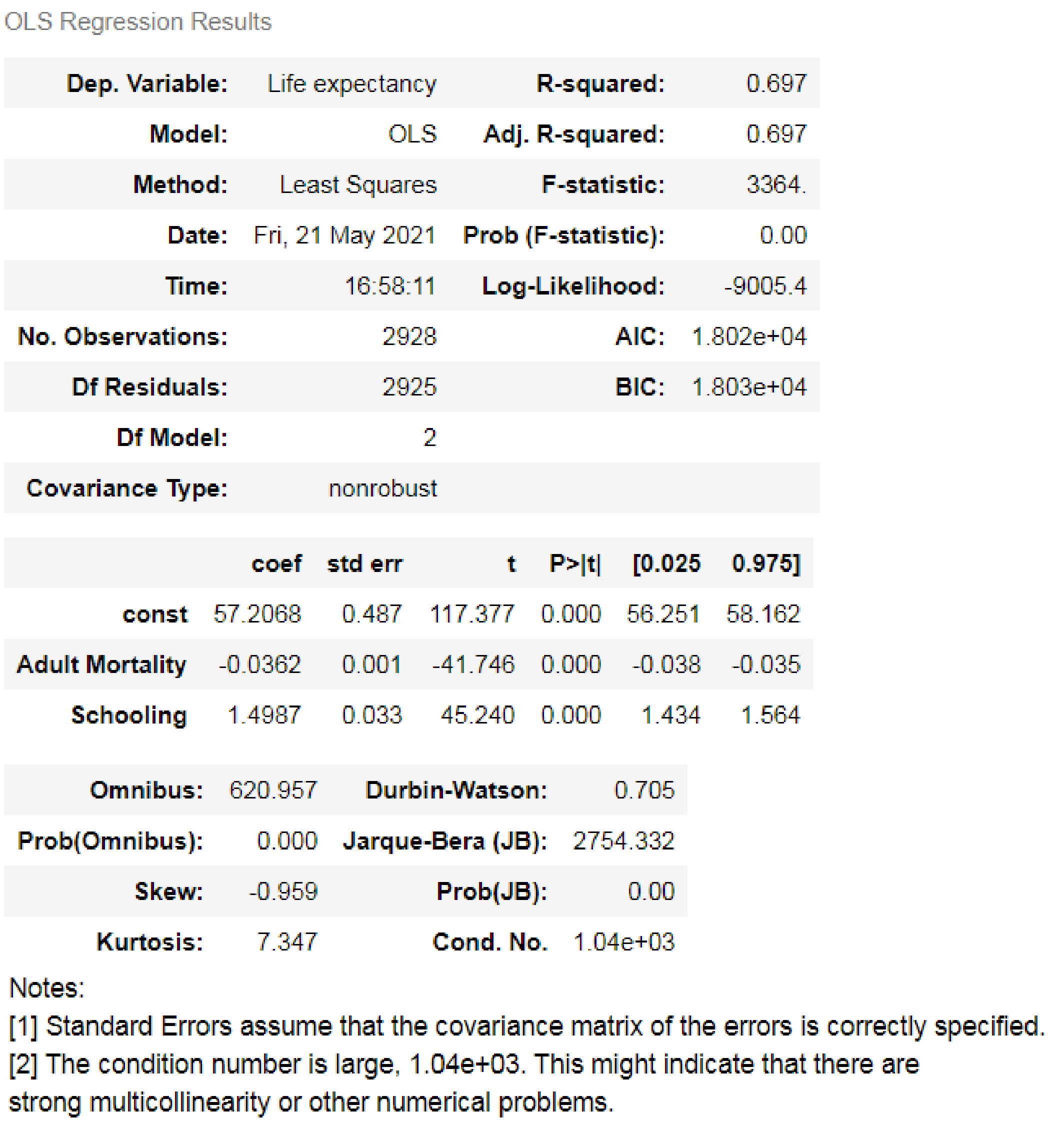
This scatter plot shows a negative correlation between *Life expectancy* and *Adult mortality*. Some of the values for scatter plot do not follow a negative correlation. Cook's distance determined that these values are not outliers.



Linear regression was done with only two variables, *Schooling* and *Adult mortality*. The R^2 value was 0.697 and the adjusted R^2 value was 0.697. The coefficient for *Adult mortality* was negative while positive for *Schooling*. The regression results state that there was multicollinearity occurring with the data.

```
#Linear regression on schooling and adult mortality

import statsmodels.api as sm
from statsmodels.formula.api import ols

# Analysis of Variance (ANOVA) on linear models


from statsmodels.stats.anova import anova_lm

X = lifeexp_continent[['Adult Mortality','Schooling']]
y = lifeexp_continent[['Life expectancy ']]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33)



## fit an OLS model with X and Y

X = sm.add_constant(X)
est=sm.OLS(y,X).fit()
est.summary()
```

The python package statsmodel, was used to obtain coefficients, p-values and other OLS regression values.



OLS Regression Results

Dep. Variable:	Life expectancy	**R-squared:**	0.697
Model:	OLS	**Adj. R-squared:**	0.697
Method:	Least Squares	**F-statistic:**	3364.
Date:	Fri, 21 May 2021	**Prob (F-statistic):**	0.00
Time:	16:58:11	**Log-Likelihood:**	-9005.4
No. Observations:	2928	**AIC:**	1.802e+04
Df Residuals:	2925	**BIC:**	1.803e+04
Df Model:	2		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
const	57.2068	0.487	117.377	0.000	56.251	58.162
Adult Mortality	-0.0362	0.001	-41.746	0.000	-0.038	-0.035
Schooling	1.4987	0.033	45.240	0.000	1.434	1.564

Omnibus:	620.957	**Durbin-Watson:**	0.705
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	2754.332
Skew:	-0.959	**Prob(JB):**	0.00
Kurtosis:	7.347	**Cond. No.**	1.04e+03

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.04e+03. This might indicate that there are
strong multicollinearity or other numerical problems.

The VIF values when only two variables, *Adult mortality* and *Schooling* were used for linear regression stated that there was moderate correlation. Therefore, the data may have been suffering from other numerical problems.

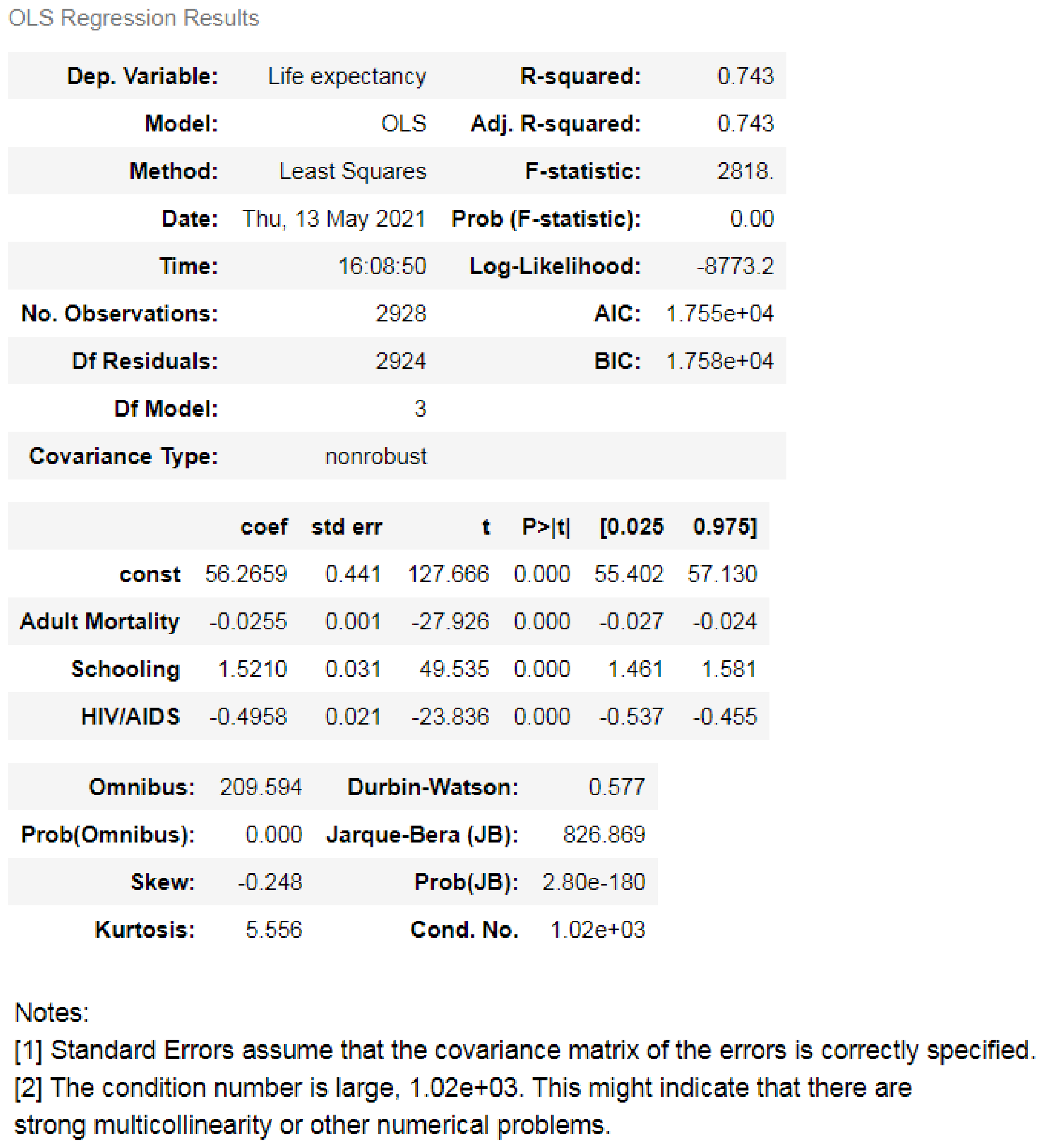
Using the information above the multiple regression equation was:

Life expectancy = 57.2068 - 0.0362 * Adult mortality + 1.4987 * Schooling. The R^2 was 0.697 not bad, it could be better by adding more variables. To see which other variables to add I did Forward and Backward regression.

I first did forward regression where variables were added one at a time. The first three variables were *Schooling*, *Adult Mortality* and *HIV/AIDS*. Doing backwards regression, the last three variables were *HIV/AIDS*, *GDP*, and *Schooling*. Forward regression had better R-squared and Adj. R-squared values.

Based on these results, the model I propose is the one obtained by the Forward regression which has a R^2/adjusted R^2 value of 0.743. The final formula is Life Expectancy = 56.2659 -0.0255 * Adult Mortality + 1.521 * Schooling - 0.4958 * HIV/AIDS.

The coefficients for *Adult mortality* and *HIV/AIDS* are negative while positive for *Schooling*. The regression results state that there was multicollinearity occurring with the data. From the VIF analysis we know that the multicollinearity was moderate. The data could be suffering from other numerical problems.



OLS Regression Results

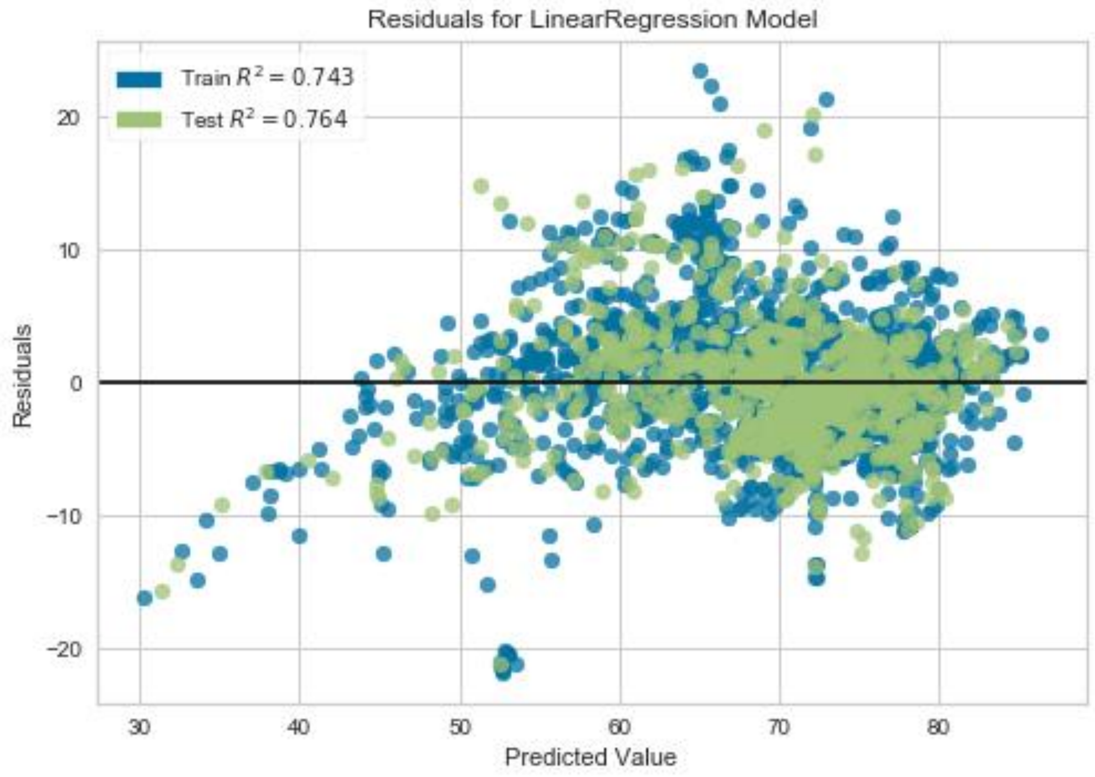
Dep. Variable:	Life expectancy	**R-squared:**	0.743
Model:	OLS	**Adj. R-squared:**	0.743
Method:	Least Squares	**F-statistic:**	2818.
Date:	Thu, 13 May 2021	**Prob (F-statistic):**	0.00
Time:	16:08:50	**Log-Likelihood:**	-8773.2
No. Observations:	2928	**AIC:**	1.755e+04
Df Residuals:	2924	**BIC:**	1.758e+04
Df Model:	3		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
const	56.2659	0.441	127.666	0.000	55.402	57.130
Adult Mortality	-0.0255	0.001	-27.926	0.000	-0.027	-0.024
Schooling	1.5210	0.031	49.535	0.000	1.461	1.581
HIV/AIDS	-0.4958	0.021	-23.836	0.000	-0.537	-0.455

Omnibus:	209.594	**Durbin-Watson:**	0.577
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	826.869
Skew:	-0.248	**Prob(JB):**	2.80e-180
Kurtosis:	5.556	**Cond. No.**	1.02e+03

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.02e+03. This might indicate that there are
strong multicollinearity or other numerical problems.

A residual analysis was done on the three variables using the package yellowbrick. Most of the points are around zero and seem to be random. The R^2 for training data was 0.743 and the R^2 for testing data was 0.764. The model chosen did a good job of predicting life expectancy.



Ridge Regression

I am taking this table from Geron's book where he shows that when using the Normal Equation, the data does not need to be standardized. With Ridge and Lasso the data does have to be standardized.

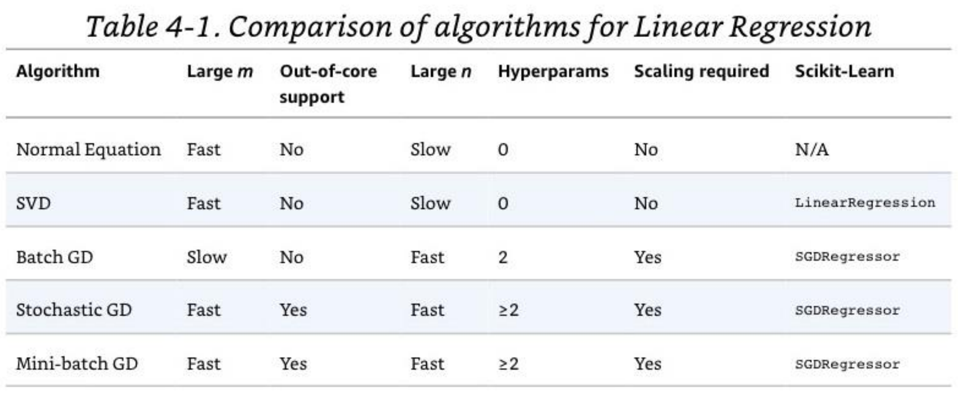


Table 4-1. Comparison of algorithms for Linear Regression

Algorithm	Large *m*	Out-of-core support	Large *n*	Hyperparams	Scaling required	Scikit-Learn
Normal Equation	Fast	No	Slow	0	No	N/A
SVD	Fast	No	Slow	0	No	`LinearRegression`
Batch GD	Slow	No	Fast	2	Yes	`SGDRegressor`
Stochastic GD	Fast	Yes	Fast	≥2	Yes	`SGDRegressor`
Mini-batch GD	Fast	Yes	Fast	≥2	Yes	`SGDRegressor`

In Géron's book we can see the cost function equation for Ridge regression. One must to find the optimal alpha in the cost function that reduces the models' complexity.

Equation 4-8. Ridge Regression cost function

$$J(\boldsymbol{\theta}) = \text{MSE}(\boldsymbol{\theta}) + \alpha\frac{1}{2}\sum_{i=1}^{n}\theta_i^{\,2}$$

Using Scikit-Learn: (In the figure below, you can see the code, this is a summary of what the code does)

- The original data was standardized using StandardScaler
- The data was split into testing and training.
- A grid search to find the best alpha was done. The best alpha was 55.
- Using that alpha ridge regression was done on the training data.
- The mean squared error was 0.1788 and the R^2 0.81
- The variables with the highest coefficients were *Adult Mortality*, *HIV/AIDS*, and *Schooling*. The new model was: Y = -0.007308.-0.263784* Adult Mortality – 0.253712* HIV/AIDS + 0.223867*Schooling.

Code for Ridge Regression

```
from sklearn.preprocessing import StandardScaler
# define standard scaler
scaler = StandardScaler()

# transform data
scaled = scaler.fit_transform(lifeexp_continent[['Life expectancy ','Adult Mortality',
       'Alcohol', 'percentage expenditure','Hepatitis B','Measles ',' BMI ',
       'Polio','Total expenditure','Diphtheria ',' HIV/AIDS','GDP','Population',
       ' thinness  1-19 years',' thinness 5-9 years','Income composition of
       resources','Schooling']])

#transform back to dataset
lifeexp_scaled = pd.DataFrame(scaled)

#Define X and y
X = lifeexp_scaled.iloc[:,1:17]
y = lifeexp_scaled.iloc[:,0]

#ridge regression
from sklearn.linear_model import Ridge
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import train_test_split
from sklearn.linear_model import *

#create train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33)

#Getting a variety of alpha values
```

```
parameters={'alpha':[1e-15,1e-10,1e-8,1e-3,1e-2,1,5,10,20,30,35,40,45,50,55,100]}

#instantiate ridge
ridge=Ridge()

# Tune alpha
ridge_regressor=GridSearchCV(ridge,parameters,scoring='neg_mean_squared_error',cv=5)

# Fit the ridge regression and find the best alpha
ridge_regressor.fit(X_train,y_train)
print("Tuned Ridge Regressor: {}".format(ridge_regressor.best_params_))

#Now using alpha value to do ridge regression and find the coefficients
from pandas import Series, DataFrame

#Alpha comes from grid search
ridge2 = Ridge(alpha = 55)
ridge2.fit(X_train, y_train)
pred2 = ridge2.predict(X_test)

mse = mean_squared_error(y_test, pred2)
print("Tuned Ridge Regressor MSE: {}".format(mse))
print("R squared from ridge method: {}".format(ridge2.score(X_test, y_test)))
print(pd.Series(ridge2.coef_, index = X.columns)) # Print coefficients
print(ridge2.intercept_)
```

Lasso Regression

In Géron's book we can see the cost function. It has the alpha parameter.

Equation 4-10. Lasso Regression cost function

$$J(\boldsymbol{\theta}) = \mathrm{MSE}(\boldsymbol{\theta}) + \alpha \sum_{i=1}^{n} | \theta_i |$$

Lasso Regression tends to minimize coefficients. In this manner it does feature selection. Features that are not part of the model are close to zero or zero.

Using Scikit-Learn: (In the figure below, you can see the code, this is a summary of what the code does)

- The original data was standardized using StandardScaler
- The data was split into testing and training.
- A grid search to find the best alpha was done. The best alpha had a value of 0.001
- Then Lasso Regression was done on the training data.
- The mean squared error was 0.1791 and the R^2 0.81
- The variables with the highest coefficients were *Adult Mortality*, *HIV/AIDS*, and *Schooling*. The new model was: Y =-0.00721 -0.267971* Adult Mortality -0.258168* HIV/AIDS + 0.233620*Schooling.

Code for Lasso Regression

```
from sklearn.linear_model import Lasso
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import mean_squared_error

#Instantiate Lasso
lasso=Lasso(max_iter =1000)

#The data was previously split into training and testing

#We are looking for the best alpha.
parameters={'alpha':[1e-15,1e-10,1e-8,1e-3,1e-2,1,5,10,20,30,35,40,45,50,55,100]}
lasso_regressor=GridSearchCV(lasso,parameters,scoring='neg_mean_squared_error',cv=5)

lasso_regressor.fit(X_train,y_train)
#Best alpha value it finds 0.001
print(lasso_regressor.best_params_)

from pandas import Series, DataFrame
lasso2 = Lasso(alpha =.001)
lasso2.fit(X_train, y_train)
pred2_lasso = lasso2.predict(X_test)

print(pd.Series(lasso2.coef_, index = X.columns)) # Print coefficients
print(mean_squared_error(y_test, pred2_lasso)) #mean squared error
print("R squared from lasso method: {}".format(lasso2.score(X_test, y_test)))
```

Final Notes

This article shows the main steps on how to build a linear regression model to predict *Life expectancy*: 1) How to do an initial inspection of the data and clean it; 2) How to use pandas to add new columns and to merge them; 3) How to use statsmodel to do linear regression and VIF analysis. 4) How to use Scikit-learn to split the data; 5) How to analyze residuals with yellowbrick; 6) How to use the matplotlib and seaborn packages to make a variety of plots and graphs; 7) How to use Scikit-learn to standardize the data and 8) Ridge and Lasso regression.

Initially, I thought that *Percentage expenditure*, *GDP* and *Populations* would be important variables in a linear regression model predicting *Life expectancy*. Since there have been studies that show that schooling improves life quality. I was not surprised that *Schooling* was in the final linear model.

Many of the Kaggle variables in the data set could have been used to predict *Life expectancy*. However, VIF showed they were highly correlated to each other. Three independent variables: *Schooling*, *Adult mortality* and *HIV/AIDS* where shown to predict *Life expectancy.* The simple model has an R^2 of 0.743, which indicates that these three variables do a good job of explaining the model.

The data was then scaled to perform Ridge and Lasso regression. The Ridge and Lasso regression found the same three variables from linear regression: *Adult mortality*, *HIV/AIDS*, and *Schooling.* Ridge and lasso regression had smaller coefficients and intercept because the data was scaled. The R^2 for ridge and lasso is 0.80 for the same three variables which indicates that the variables do a good job of predicting *Life expectancy*.

Of the three types of regression studied here, Lasso regression is the best. It reduces the coefficients of not needed variables to zero. Thus, performing feature selection on the most important features in the data set.

The python ecosystem has a variety of packages that can work together: I used AWOC to add continent information to the countries; Statsmodel for OLS regression; Scikit-learn for splitting the data, Ridge regression, Lasso regression and grid search of the parameter; and Yellowbrick for residual analysis

References

DataSklr. (2021, Jan 11). *Feature Selection with Python.* https://www.datasklr.com/ols-least-squares-regression/variable-selection

Geron, A. (2019). Hands on Machine Learning with Scikit-Learn, Keras, amd TensorFlow. (2nd Edition). O'REILLY.

Grandicelli, L. (2021, Jan 6). *a-world-of-countries 1.0.0.* https://pypi.org/project/a-world-of-countries/

Honda, S. (2021, Jan 5). *Meet Pandas: Grouping and Boxplot.* https://hippocampus-garden.com/pandas_boxplot/

Koleva, Nancy. (May 1, 2020). When and When Not to use Deep Learning. https://blog.dataiku.com/when-and-when-not-to-use-deep-learning

Rajarsi, K. (2021, Jan 1). *Life Expectancy (WHO) Statistical Analysis on factors influencing Life Expectancy.* https://www.kaggle.com/kumarajarshi/life-expectancy-who

Shin, T. (2021, Jan 1). *14 Data Science projects to improve your skills*. https://www.kdnuggets.com/2020/12/14-data-science-projects-improve-skills.html

Statsmodels by developers. (2021, Jan 18). *Ordinary Least Squares.* https://www.statsmodels.org/dev/examples/notebooks/generated/ols.html

Scikit-by developers. (2021, Jan 25). *Residuals Plot.* https://www.scikit-yb.org/en/latest/api/regressor/residuals.html

Scikit-learn developers. (2021, Jan 3). *sklearn.model_selection.train_test_split.* https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

Cosine1509. (2021, Jan 2). *Detecting Multicollinearity with VIF-Python.* https://www.geeksforgeeks.org/detecting-multicollinearity-with-vif-python/

Cite the paper from Terence shin