

Project Report



Problem Statement :

Regression analysis for establishing a relation between response and regressor variables.

We are given a life expectancy dataset. We need to find whether there is a positive or negative correlation between alcohol and life expectancy. Next, we need to report this relational analysis using linear and non-linear regression. Finally, we need to calculate R^2 to measure the quality of fit.

(A) Understanding the theory to solve the project problem :

Part-A Correlational Analysis :

The correlation coefficient (ρ) is a measure that determines the degree to which the movement of two different variables is associated. The most common correlation coefficient, generated by the Pearson product-moment correlation, is used to measure the linear relationship between two variables.

Correlation coefficients indicate the strength of the linear relationship between two different variables, x , and y . A linear correlation coefficient greater than zero indicates a positive relationship. The value that is less than zero signifies a negative relationship. Finally, a value of zero indicates no relationship between the two variables x and y .

The possible range of values for the correlation coefficient is -1.0 to 1.0. In other words, the values cannot exceed 1.0 or be less than -1.0. A correlation of -1.0 indicates a perfect negative, and a correlation of 1.0 indicates a perfect positive correlation.

Formula

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

r = correlation coefficient

x_i = values of the x-variable in a sample

\bar{x} = mean of the values of the x-variable

y_i = values of the y-variable in a sample

\bar{y} = mean of the values of the y-variable

Part-B (i) Linear Regression analysis:

Linear regression is a basic and commonly used type of predictive analysis.

It is a machine learning algorithm based on supervised learning. It performs the task to predict a dependent variable value (Y) based on a given independent variable (x).

This regression technique finds out a linear relationship between x (input) and y(output). Hence, the name is Linear Regression. These regression estimates are used to explain the relationship between one dependent variable and one or more independent variables.

In simple linear regression, we have only two variables:

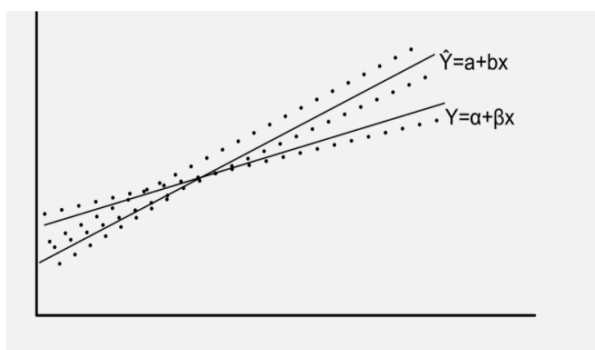
- ☐ Dependent variable (also called Response), usually denoted as Y.
- ☐ Independent variable (alternatively called Regressor), usually denoted as x.

The linear relationship between the dependent variable(Y) and the independent variable(x) is represented in the form of $Y = \alpha + \beta x$.

The concept of regression analysis deals with finding the best relationship between Y and x.

The best-fitted values of a & b quantify the strength of the relationship between them. Here, α and β are called regression coefficients. These values are to be estimated from the data.

Suppose, we denote the estimates a for α and b for β . Then the fitted regression line becomes $Y' = a + bx$. Then, the plotted lines for these two equations are as shown below.



$$b = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}$$
$$a = \bar{y} - b\bar{x}$$

Now, these two equations can be solved to determine the values of a and b.

Part-B (ii) Non-linear regression analysis:

When the regression equation is in terms of r-degree, $r > 1$, then it is called a nonlinear regression model. When more than one independent variable is there, then it is called the Multiple Non-linear Regression model. It is also alternatively termed as a polynomial regression model. In general, it takes the form as shown

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \dots + \beta_r x^r + \epsilon$$

The polynomial model can be transformed into a general linear regression model setting $x_1 = x, x_2 = x^2, \dots, x_r = x^r$. Thus, the equation assumes the form:

$$y_i = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_r x_r + \epsilon_i$$

$$\hat{y}_i = b_0 + b_1 x_1 + b_2 x_2 + \dots + b_r x_r + e_i$$

This model then can be solved using the procedure followed for multiple linear regression model.

Nonlinear regression uses logarithmic functions, trigonometric functions, exponential functions, power functions, Gaussian functions, and other fitting methods.

In simple terms, **Non-linear regression** is a method of finding a nonlinear model of the relationship between the dependent variable and a set of independent variables.

$$\text{We have } SSE = \sum_{i=1}^n (y_i - \hat{y})^2$$

The goal of the model is to make the **Sum of the Squares (SSE)** as small as possible. The sum of squares is a measure that tracks how far the Y observations vary from the nonlinear (curved) function that is used to predict Y.

Coefficient of Determination (R-Square)

Calculation of R2 for both linear and nonlinear regression quantity R2 is called the coefficient of determination which is used to measure the proportion of the variability of the fitted model.

let us define the **total corrected sum of squares**, defined as

$$SST = \sum_{i=1}^n (y_i - \bar{y})^2$$

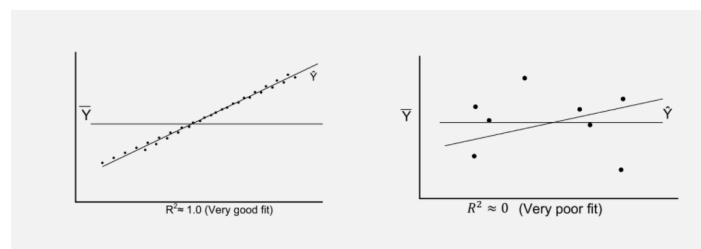
- SST represents the variation in the response values. The R^2 is

$$R^2 = 1 - \frac{SSE}{SST}$$

Note:

- If fit is perfect, all residuals are zero and thus $R^2 = 1.0$ (very good fit)
- If SSE is only slightly smaller than SST, then $R^2 \approx 0$ (very poor fit)


The R2 value represents the variance of the data. The value of R2=0.90 means that 0.10 of the variance can not be explained by the model. In the logical case when R2=1, the model completely fits and explains all variance.



Code Implementation

Step 1: Installing Required Packages

```
install.packages("ggplot2")  
install.packages("tidyverse")
```



```
install.packages("dplyr")
install.packages("GGally")
install.packages("caTools")
install.packages("GGally")
install.packages("MLmetrics")
```

```
#Using the installed packages
library(ggplot2)
library(tidyverse)
library(dplyr)
library(GGally)
library(caTools)
library(MLmetrics)
library(caret)
```

Step 2: Importing data

```
# Choose file
dataset <- read.csv(file.choose(),header=TRUE)
summary(dataset)    # Overview of the dataset
str(dataset)
```

Step 3: Data Preprocessing

```
# Dropping the NaN values
dataset <- dataset %>% drop_na()
range(dataset$Life.expectancy) # new dataset size
```

```
# Considering only Life Expectancy and Alcohol rows in the data
life_selected <- dataset %>% select(Life.expectancy, Alcohol)
```

Step 4: Finding Correlation Analysis

```
# Correlation between variables without using In-built
x <- dataset$Alcohol
y <- dataset$Life.expectancy

m1 <- mean(x)
m2 <- mean(y)
size <- dim(dataset)
temp1 <- sum((x-m1) * (y-m2))/size[1]
s1<-sd(x)
s2<-sd(y)
```

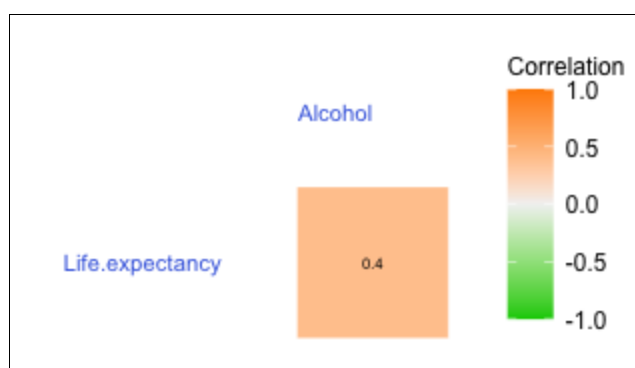
```
temp2 <- s1*s2
coff <- temp1/temp2
sprintf("Coefficient of Correlation is %f",coff)
```

Our Output using the Formula:

"Coefficient of Correlation is **0.402474**"

```
# Correlation between variables using In-built
data_num <- life_selected %>% select_if(is.numeric)
ggcorr(data_num,
  label = T,
  label_size = 2,
  label_round = 2,
  hjust = 1,
  size = 3,
  color = "royalblue",
  layout.exp = 5,
  low = "green3",
  mid = "gray95",
  high = "darkorange",
  name = "Correlation")
```

Correlation between Life Expectancy and Alcohol



Hence, Life Expectancy has a positive relationship with drinking alcohol.

Step 5: Simple Linear Regression

```
# Generate synthetic data with a clear linear relationship
x <- dataset$Alcohol
y <- dataset$Life.expectancy

# Convert to dataframe
```



```

simple_lr_data <- data.frame(x, y)

# Calculate coefficients
b1 <- (sum((x - mean(x)) * (y - mean(y)))) / (sum((x - mean(x))^2))
b0 <- mean(y) - b1 * mean(x)

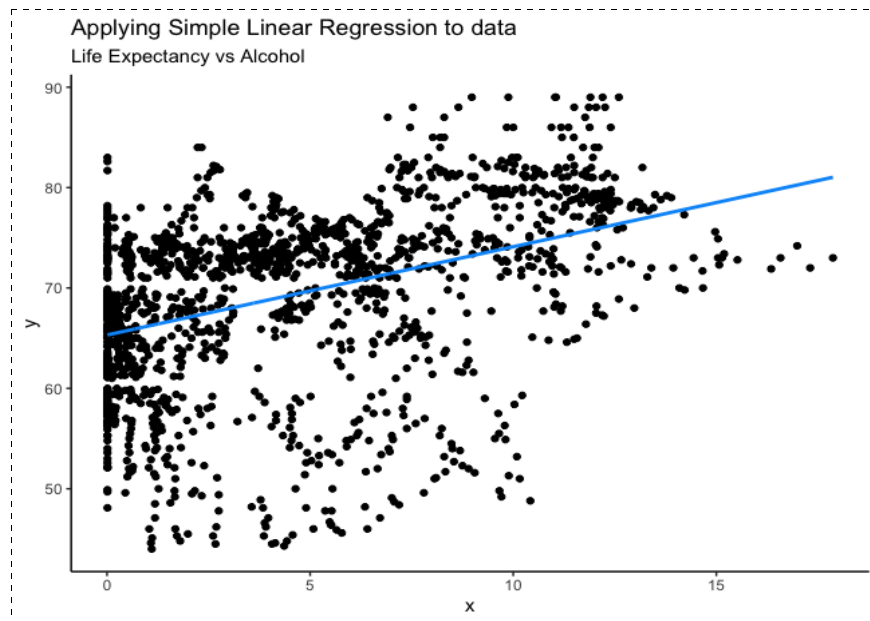
# Function for generating predictions
simple_lr_predict <- function(x) {
  return(b0 + b1 * x)
}

# Apply simple_lr_predict() to input data
simple_lr_predictions <- sapply(x, simple_lr_predict)
simple_lr_data$yhat <- simple_lr_predictions

# Visualize input data and the best fit line
ggplot(data = simple_lr_data, aes(x = x, y = y)) + geom_point(size = 3, color = "#0099f9") +
  geom_line(aes(x = x, y = yhat), size = 2) + theme_classic() +
  labs(
    title = "Applying Simple Linear Regression to data",
    subtitle = "Life Expectancy vs Alcohol"
  )

```

Linear Regression Output using Formula



```

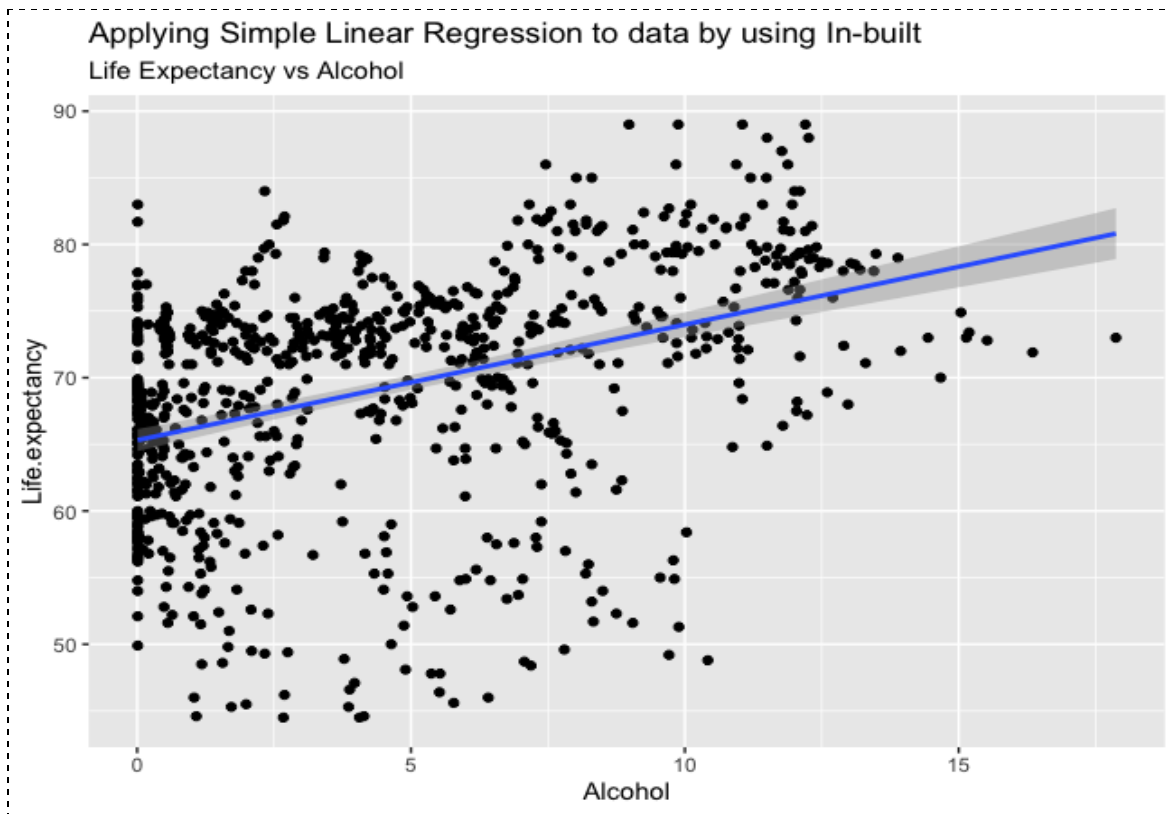
# Visualizing best fit line using In-built
ggplot(train, aes(Alcohol, Life.expectancy)) + geom_point() + stat_smooth(method = lm,
formula = y ~ x) +
  labs(
    title = "Applying Simple Linear Regression to data by using In-built",

```

```
subtitle = "Life Expectancy vs Alcohol"
```

```
)
```

Linear Regression Output using In-built Library



Non-Linear Regression without using In-built:

```
non_lr_data <- data.frame(x, y)
```

```
non_lr_predict <- function(x) {  
  return(8 + b0 + (0.4-b0)*exp(-b1*(x+2.3)))  
}
```

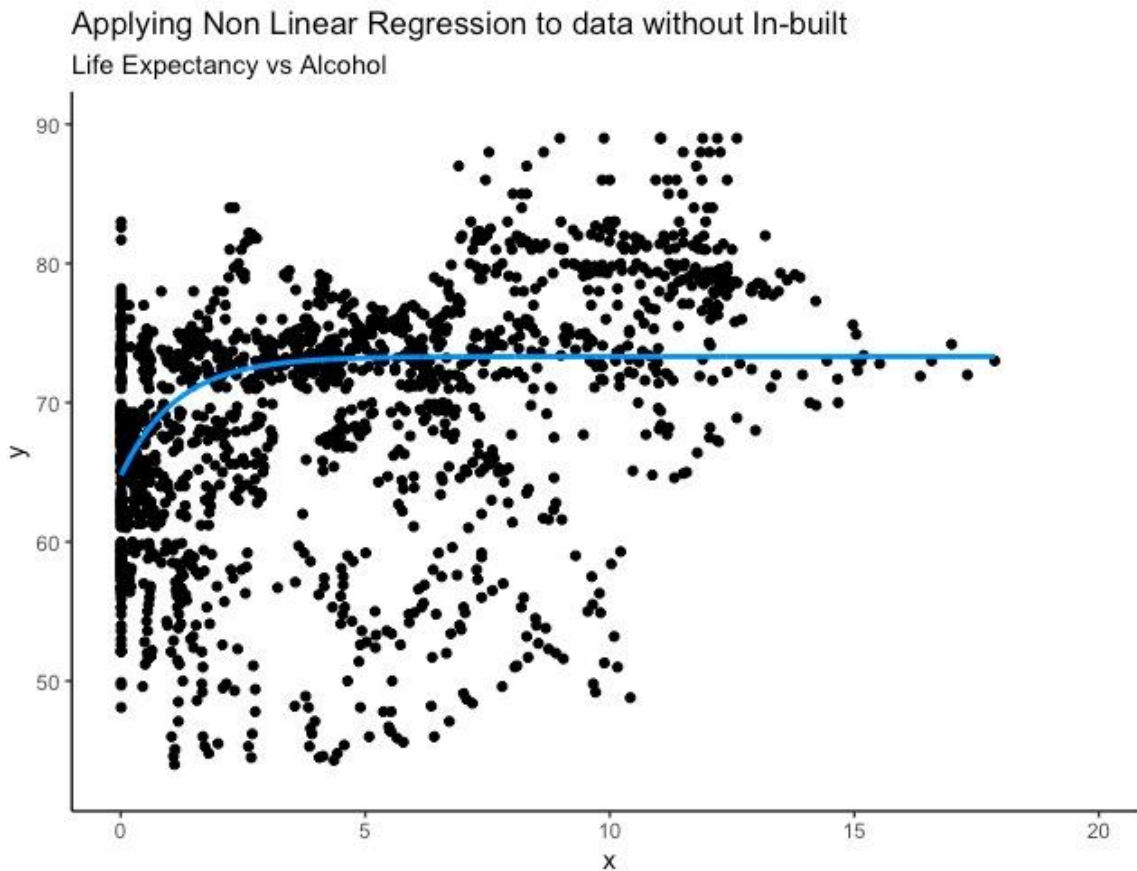
```
non_lr_predictions <- sapply(x, non_lr_predict)  
non_lr_data$yhat <- non_lr_predictions
```

```
ggplot(data = non_lr_data, aes(x = x, y = y)) +  
  geom_point() +  
  geom_line(aes(x = x, y = yhat), size = 1, color = "#0099f9") +  
  theme_classic() +  
  labs(
```

```

title = "Applying Non Linear Regression to data without In-built",
subtitle = "Life Expectancy vs Alcohol"
) + xlim(0, 20) + ylim(43, 90)

```



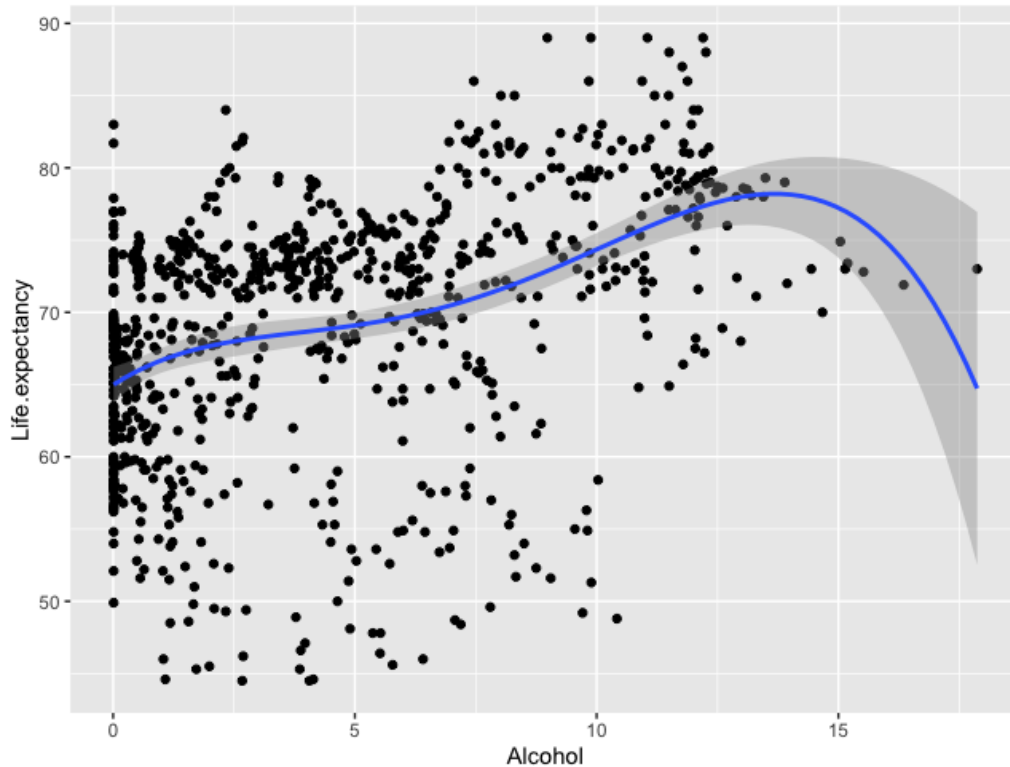
Non-Linear Regression using In-built:

```

# Non Linear Regression using In-built
lm(Life.expectancy ~ poly(Alcohol, 4, raw = TRUE), data = train) %>% # Taking bi-quadratic equation
summary()
model <- lm(Life.expectancy ~ Alcohol, data = train)

# Visualize input data and the best fit line
ggplot(train, aes(Alcohol, Life.expectancy)) +
  geom_point() + stat_smooth(method = lm, formula = y ~ poly(x, 4, raw = TRUE))

```



R Square for Simple Linear Regression for the given dataset:

```
predictions <- sapply(test$Alcohol, simple_lr_predict)
```

```
# Finding R Square value using formula
```

```
rss <- sum((predictions - test$Life.expectancy) ^ 2) ## residual sum of squares
```

```
tss <- sum((test$Life.expectancy - mean(test$Life.expectancy)) ^ 2) ## total sum of squares
```

```
rsq <- 1 - rss/tss
```

```
sprintf("R Square is %f",rsq)
```

Our output using the formula:

"R Square is **0.166774**"

```
# Finding R Square value using In-built
```

```
data.frame(
```


```
  RMSE = RMSE(predictions, test$Life.expectancy),
```

```
  R2 = R2(predictions, test$Life.expectancy)
```

```
)
```

Output using In-built package:

R2: **0.1668495**



```
# Finding R Square value using inbuilt correlation
RSQUARE = function(y_actual,y_predict){
  cor(y_actual,y_predict)^2
}

LR_R = RSQUARE(test$Life.expectancy,predictions)
sprintf("R Square using In-built is %f",LR_R)
```

Output using above function using correlation:

" R Square using In-built is **0.166850** "

R Square value of Non-Linear Regression for the given dataset

```
predictions2 <- model %>% predict(test) # Predicting o/p values for testing data

# Finding R Square value using Formula
rss <- sum((predictions2 - test$Life.expectancy) ^ 2) ## residual sum of squares
tss <- sum((test$Life.expectancy - mean(test$Life.expectancy)) ^ 2) ## total sum of squares
rsq <- 1 - rss/tss
sprintf("R Square is %f",rsq)
```

Our output using the formula:

"R Square is **0.166551**"

```
# Finding R Square value using In-built
data.frame(
  RMSE = RMSE(predictions, test$Life.expectancy),
  R2 = R2(predictions, test$Life.expectancy)
)
```

Output using In-built package:
0.1668495

```
# Finding R Square value using In-built Correlation
RSQUARE = function(y_actual,y_predict){
  cor(y_actual,y_predict)^2
}

LR_R = RSQUARE(test$Life.expectancy,predictions2)
sprintf("R Square using In-built is %f",LR_R)
```

Output using above function using correlation:

"R Square using In-built is **0.1668495**"



Experimental results :

In this experiment, we used a Life expectancy dataset downloaded from Kaggle.

1. After carefully observing the Life Expectancy dataset and calculating the correlation between the dataset\$Life.Expectancy and dataset\$Alcohol, we observed that these both have a positive relationship. The measured value was found out to be 0.4. Since the range of correlation coefficient ranges between -1.0 to 1.0 and as also explained in part A of Understanding the theory to solve the project problem part, the value 0.4 means they are positively correlated. Therefore, Life Expectancy has a positive relationship to Drinking alcohol.
2. a. For simple linear regression analysis we have performed it using both the formula, and then calculated the coefficients and also done by inbuilt methods. It was observed that both the methods resulted in the same plots. Firstly, we have split the data into train and test. Then, In the case where the formula was used, the coefficients b_0 and b_1 were computed separately using the formulas as discussed earlier mentioned in the theory part. Now, the equation $Y=b_0+b_1x$ is computed and then the graph is plotted. Also, in the in-built case, the model was trained on the training dataset using the `lm()` function. The plots were displayed earlier in the code implementation and they were observed to be the same.

b. For simple non-linear regression also, we have done it using both the methods including formula calculation and built-in functions. In the case of the formula method, the degree of the polynomial was taken to be 2 and the parameters such as $x_1=x$ and $x_2=x^2$ were taken into consideration, and the equations transformed from

$$y_i = \beta_0 + \beta_1 x + \beta_2 x^2$$
$$\hat{y}_i = b_0 + b_1 x + b_2 x^2$$

To

$$y_i = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$
$$\hat{y}_i = b_0 + b_1 x_1 + b_2 x_2$$

Now, this model is turned to linear and can be solved using the procedure followed for the multiple linear regression model by first calculating the coefficients b_0, b_1, b_2 appropriately by partial derivative method and then substituting back into the equation and then plotting the graph.

In the case of using an in-built function, the model was trained on the training dataset using the `lm()` function and substituting the parameters as using a polynomial function with degree 4. The higher degree was taken for our training set to perform better and give us the best fit while plotting the graph.

3. a. The value of R square for the Simple Linear Regression model using the formula was found out to be 0.166774 and the R square value for the same Simple Linear Regression was found out to be 0.1668495 (by using In-built Correlation and Not In-Built R2 methods).
- b. The value of R square for Non-Linear Regression using formula was found out to be 0.166551 and the R square value for the same Nonlinear Regression was found out to be 0.1668495 (by using In-built Correlation and Not In-Built R2)