Regression Analysis Project

Subject Code : MTH416A
Semester II
April 2021

Title

A Regression Analysis to Study the Effects of Major Economical Variables on the Change in GDP (At Current Prices) of India (1980-2020)

By:
Rachita Mondal - 201374
Souvik Bhattacharyya - 201433
Shreya Pramanik - 201415
Arkaprova Saha -201278
Bimal Roy - 201292

Under Supervision of
Dr. Sharmishtha Mitra

Department: Statistics
IIT Kanpur

ACKNOWLEDGEMENT

We would like to express a deep sense of thanks and gratitude to Dr. Sharmishtha Mitra for providing us the opportunity to prepare this project and constantly motivating us with constructive advices. It has been a great learning experience building practical insights of the theoretical knowledge gathered during course lectures.

Last, but not the least, our parents provided us with continuous encouragement and extensive support throughout the session. So , with due regards we express our gratitude to them for completion of the project within the stipulated time-period.

THANK YOU

Table of Contents

15.BIBLIOGRAPHY

1. Introduction & Objective

1.1. Introduction

Gross Domestic Product:

Gross Domestic Product is an essential concept of the economy of any country. It measures the growth of a country in terms of its production. By definition Gross Domestic Product is the total monetary value of all finished goods and services produced in a country within a specified period of time. There are various uses of the figure of GDP of a country in determining the overall growth of that country. Specially, the growth rate of the economy is measured by this quantity.

There are various ways for calculating the GDP in a country. Mainly three methods are adopted to calculate it. The measures are: The method of Expenditure, The method of Production and The method of Incomes.

The Expenditure Method: in this method the Gross Domestic product is calculated by adding the total Consumption Expenditure of a country, the Government spending, the investment and the net exports of a country. The net export is calculated by taking the difference between the export and the imports.

Production Approach: this approach can be considered as the opposite of the expenditure method. In this method the monetary value of the total output of a country is measured. The total output is adjusted by deducting the cost of intermediate goods in the process of production.

Income Approach: the income approach can be thought of as an intermediate approach that lies between the production approach and the expenditure approach. It basically calculates the income earned by all the factors of production that are essential in conducting a production process in an economy. It includes the wages paid to the labour, the rent earned by the land, the return to the capitals, and the profit earned by the entrepreneurs.

In India the method of Expenditure approach is used mainly to calculate the GDP. India's Central statistics office calculates the Gross Domestic Product.

GDP= Private Consumption(C) + Government Spending (G)+ Investment (I)+ Net Export (N)

1.2. Objective

The elements considered in the Gross Domestic Product calculation can be affected by several sectors of the Indian economy. The production and the corresponding prices of product can directly influence the GDP. Not only that the components present in the Money and banking sector and also that of the financial markets can have an impact upon the Indian Gross Domestic Product. The sectors of public finance and the factor of trading and balance of payment should also be considered by discussing the factors that influence the gross domestic product. In this particular study we tried to select a sufficient number of variables from all possible sectors so that the reason behind the change in Gross Domestic Product can be accounted for. Among all the selected variables some factors can have severe impact while some can have in direct impact upon the GDP. In this study our objective is to detect those variables that are important

in the concept of GDP of India and to use an appropriate regression model to express the GDP in terms of those variables.

2. Description of Data

Apart from the Gross Domestic Product we have selected 15 more important economic variables. Brief Descriptions of all these variables are given below:

Agricultural Production of Foodgrains:
Agriculture plays an important role in the formation of Indian economy. The production of food grains constitutes a major part of India's total agricultural production. The major food grains that are produced in India are rice, wheat, Maize, Bajra, Pulses, oil millets etc.

Agricultural Production of Commercial Products:
Commercial products are also an important type of Agricultural production. Apart from food grains the products like tea, coffee, tobacco, cotton, rubber, oil seeds are generally grown for commercial purposes. Also there are some food grains that are also used as commercial goods for example with maize, oil seeds, sugarcane etc.

.

Production and Import of Crude oil and Petroleum:
Indian economy and Indian market are strongly affected by the prices of crude oil and petroleum. Therefore the production of these commodities are very much important in the Indian context. The overall economical cycle can be affected by the price of oil.

Export of Principal Commodities:
India's major Exports are mainly the petroleum products, Gems, Jewelleries, machineries, tea, coffee, tobacco, iron steel etc. The total income from exporting affects the Indian economy to a remarkable amount.

Import of Principal Commodities:
The most important products that are imported to India are crude oil, gold, solid oil,diamonds etc. Not only that some major factors of production like machineries are imported so that a good quality production can be possible**.**

Direct and Indirect Tax Revenue:
Direct and indirect tax revenue is a principal source of government's income. Direct tax includes Income Tax, commercial property tax, personal property tax, taxes on assets etc. Whereas indirect taxes are those taxes that are imposed on the goods and services like sales tax, consumption tax, Goods and Service tax (GST), tax collected by the intermediaries.

Total Saving Deposits in Commercial Banks:
The savings account in a commercial bank includes the feature that only a pre-specified number of withdrawals can be taken within a specified period of time. This money plays an important role in building the Indian economy when the government invests this money for loan purposes.

Gross Fiscal Deficit:
Fiscal deficit is the difference between the total income of the Government and its total expenditure. It is an important concept in the context of Indian Economy. The government needs to take necessary measures for financing this deficit and that in turn can lead to the changes of major aspects of Indian economic cycle.

Average price of Gold in Domestic Market:
The gold reserve of a country affects the supply of currency within the economy. If the central bank imports gold then it can result to an inflation in the economy. Therefore, the price of gold affects the demand and supply of gold and alternatively it affects the economic cycle.

Liabilities of RBI:
The liability of Reserve Bank of India mainly consists of the issued notes, the deposits by the commercial banks that are held by RBI, the cash reserve ratio and the provisions that is again the combination of the currency and the gold reserves etc.

Total Market Borrowing of Government:
In many cases the government needs to raise money from the market to meet its necessary expenses. These expenses can include the financing of Fiscal deficit and repaying loans etc. The government borrowing affects the private investment of a country.

Total Currency with Public:
The total currency with the public is the difference between the total value of the currency including the coins and the paper currency issued by the Reserve Bank of India and the amount of that currency withdrawn by the Reserve Bank of India. The currency with the public may affect the production of the total investment expenditure of the country.

Government's Developmental and Non-Developmental expenditure:
The developer expenditure includes those expenditures that it helps in increasing the production and in turn the national income of the country. The expenditures incurred by the government that do not directly help in economic development or production can be termed as the non developmental expenditures. It includes the cost of tax collection, the cost of printing notes, the expenses for maintaining the law and order of a country, the expenditure on Defence etc.

Investment by LIC:
This factor plays an important role in increase in Indian Development.

Minimum Support Price for foodgrains:
The minimum support price is an agricultural product price that is set by the government of India to purchase products directly from the producers. Basically this is provided to safeguard the farmers interest so that the farmer can get a minimum amount of profit over the total production.

3. Description of Multiple Linear Regression

3.1. Model

Given a dataset of n observations involving p regressors the MLR model takes the form:

$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta p X_P + \epsilon \quad \forall i=1,2,\ldots..,n$

ε = error term of the model

3.2. Assumption

$E[\varepsilon_i] = 0, \forall i = 1(1)n$

$Var[\varepsilon_i] = \sigma^2, \forall i = 1(1)n$

$Cov[\varepsilon_i, \varepsilon_j] = 0, \quad \forall i \neq j$

Referring to the MLR equation above, in our example:

3.3. Response Variable

Y= Gross Domestic Product at Current Price

3.4. Explanatory Variables

X1=Agricultural production of food grains.

X2= Agricultural production of commercial products.

X3 = Production and import of crude oil and Petroleum.

X4 = Export of principal commodities.

X5= Import of principal commodities.

X6=Direct and Indirect tax revenue.

X7=Total saving deposits in commercial banks.

X8=Gross Fiscal Deficit.

X9=Total borrowing of Government

X10=Liabilities of RBI.

X11=Government's Developmental and Non-Developmental expenditure.

X12=Total currency with public.

X13=Investment by LIC.

X14=Minimum Support Price for food grains

X15=Price of Gold(Spread)

3.5. Normal equations

In matrix notation our model can be written as,

$$Y=X\beta + \epsilon$$

Here ,

$$Y = \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ . \\ y_n \end{bmatrix}, X = \begin{bmatrix} 1 & x_{11} & x_{12} & \dots & x_{1p} \\ 1 & x_{21} & x_{22} & \dots & x_{2p} \\ & & . & & \\ & & . & & \\ & & . & & \\ 1 & x_{n1} & x_{n2} & \dots & x_{np} \end{bmatrix} \text{ and } \beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ . \\ . \\ . \\ \beta_p \end{bmatrix} \text{ also } \epsilon = \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ . \\ . \\ . \\ \epsilon_n \end{bmatrix}$$

We will find the estimate of β by minimizing the Sum of Squares due to Error with respect to β.

Sum of Square due to Error = $S(\beta) = \varepsilon^T\varepsilon = (Y-X\beta)^T(Y-X\beta)$

Differentiating S(β) with respect to β we can obtain the Least Square Normal Equation,

$$(X^TX)\beta = X^TY.$$

The Least Square Estimate of β is given by, $\hat{\beta} = (X^TX)^{-1}X^TY$, provided $(X^TX)^{-1}$ exists.

4. Data Cleaning

In [1]:
```
import pandas as pd
import numpy as np
import xlsxwriter
import matplotlib.pyplot as plt
%matplotlib inline
```

Data cleaning is the process of ensuring data is correct, consistent and usable. In order to create a reliable dataset we need to adopt Data Cleaning method so that we can increase the quality of our data set. In our study we will go through following steps for cleaning our data:

i) Missing value Treatment

ii) Duplicate data Treatment

4.1. Missing value Treatment, Removal of Duplicate Values GDP data

In [9]:
```
gdp_data=pd.read_csv(r"GDP Data.csv")
gdp_data.head(2)
```

Out[9]:

	Year	GDP_at_Current_Prices
0	1980-81	1496.42
1	1981-82	1758.05

4.1.1. Missing Value Treatment:

In [12]:
```
#No of missing values in the data
gdp_data.isnull().sum()
```

Out[12]:
```
Year                    0
GDP_at_Current_Prices   0
dtype: int64
```

Our data on GDP at Current Price does not contain any missing value. So, we don't need to take any remedial measure for it.

4.1.2. Detection and Removal of Duplicate data:

In [13]:
```
#No of duplicate rows in year column
gdp_data['Year'].duplicated().sum()
```

Out[13]: 1

In [14]:
```
#Duplicate data in dataset
gdp_data.loc[gdp_data['Year'].duplicated(keep=False), :]
```

Out[14]:

	Year	GDP_at_Current_Prices
31	2011-12	90097.21892
32	2011-12	87360.39000

It can be observed from the above analysis that the original data on GDP at Current Price contains duplicated rows. Here both the 31st and 32nd datapoints contain the data on GDP for 2011-12. In this case we will drop the 31st row and keep the 32nd row in our dataset.

In [15]:
```
#Removal of the duplicate data saving changes to original dataset
gdp_data.drop_duplicates(subset=['Year'],keep='last',inplace= True)
```

In [16]:
```
print(gdp_data.tail(10))
```

```
       Year  GDP_at_Current_Prices
30  2010-11               77841.16
32  2011-12               87360.39
33  2012-13               99513.44
34  2013-14              112727.64
35  2014-15              124882.05
36  2015-16              135760.86
37  2016-17              153916.69
38  2017-18              170983.04
39  2018-19              189712.37
40  2019-20              203398.49
```

In [17]:
```
gdp_data.shape
```

Out[17]:
```
(40, 2)
```

Thus, we have succesfully got rid of the problem of duplicated row from the data on GDP at Current Price.

4.2. Missing Value Treatment, Removal of Duplicate Data of the Regressor Variables

4.2.1. Missing Value Treatment:

In [2]:
```
regressor_data=pd.read_csv(r"Regressor Data.csv")
regressor_data.head(2)
```

Out[2]:

	Year	Total_Foodgrains	Commercial_Crops	CrudeOil_POL_Prod_Imp	Total_Exports	Total_Imports
0	1980-81	1295.9	8752.7	57	6710.71	125.4915
1	1981-82	1333.0	9376.6	72	7805.90	136.0755

In [3]:
```
#No of missing values in the data
regressor_data.isnull().sum()
```

Out[3]:
```
Year                      0
Total_Foodgrains          0
Commercial_Crops          2
CrudeOil_POL_Prod_Imp     0
Total_Exports             0
Total_Imports             0
Tax_Reveneues             0
Total_Savings_Deposits    0
Fiscal_Deficit            0
Gross_Market_Borrowing    0
Liabilities_of_RBI        0
Total_Expenditures        0
Currenc_with_the_Public   0
```

```
Investments_by_LIC              0
Support_Price_for_Foodgrains    1
Price_of_Gold                   0
dtype: int64
```

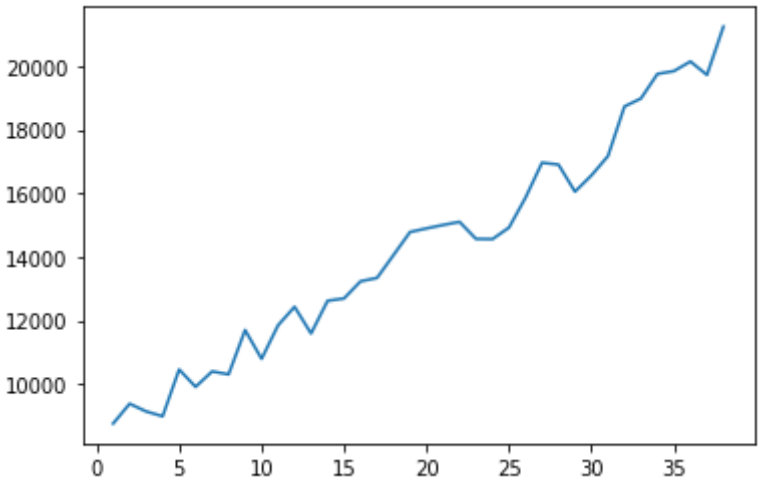
From the above analysis it can be realised that the data on Commercial_Crops contains two missing observations and the data on Support_Price_for_Foodgrains contain one missing observation.

4.2.1.a. Extrapolation of the values of Commercial_Crops

In [4]:
```
x=[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,
```

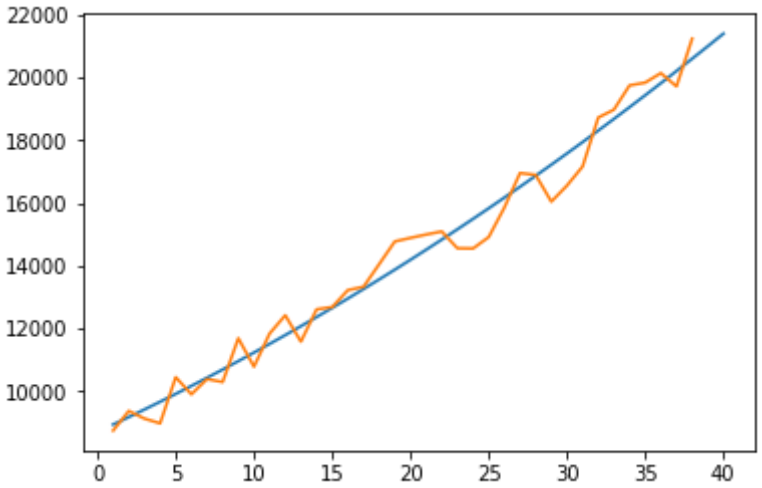
In [5]:
```
y=regressor_data['Commercial_Crops'].dropna()
plt.plot(x[0:38],y)
```

Out[5]:
```
[<matplotlib.lines.Line2D at 0x183cc57d7c0>]
```



In [6]:
```
x_arg=x[0:38]
poly_coeff=np.polyfit(x_arg,y,2)
poly_coeff
y_fit=np.poly1d(poly_coeff)
x_new=x
plt.plot(x_new,y_fit(x_new))
plt.plot(x[0:38],y)
```

Out[6]:
```
[<matplotlib.lines.Line2D at 0x183ce6653a0>]
```



We have plotted the available observations on Commercial_Crops againts their respective time points. From the plot it can be realised that a second degree polynomial will be suitable to the

data. So, we fitted a second degree polynomial to the data.

In [7]:
```
y_fit(38)
```

Out[7]:
```
20611.37955465587
```

In [8]:
```
y_fit(39)
```

Out[8]:
```
21008.542247510668
```

In [9]:
```
y=np.append(y,20611.379554655876)
y=np.append(y,21008.54224751067)
y
```

Out[9]:
```
array([ 8752.7       ,  9376.6       ,  9134.2       ,  8982.1       ,
       10455.3       ,  9907.7       , 10394.1       , 10301.3       ,
       11695.7       , 10789.4       , 11847.        , 12428.1       ,
       11588.1       , 12617.2       , 12691.7       , 13231.2       ,
       13337.4       , 14058.4       , 14782.5       , 14896.1       ,
       15003.3       , 15101.8       , 14565.4       , 14558.9       ,
       14922.7       , 15863.2       , 16964.        , 16904.8       ,
       16053.2       , 16564.9       , 17172.8       , 18734.2       ,
       18986.2       , 19759.5       , 19848.7       , 20150.8       ,
       19727.3       , 21248.2       , 20611.37955466, 21008.54224751])
```

In [10]:
```
regressor_data['Commercial_Crops']=y
regressor_data.tail(5)
```

Out[10]:

	Year	Total_Foodgrains	Commercial_Crops	CrudeOil_POL_Prod_Imp	Total_Exports	Total_Imports
35	2015-16	2515.7	20150.800000	501	1716384.0	24903.06
36	2016-17	2751.1	19727.300000	530	1849434.0	25776.75
37	2017-18	2850.1	21248.200000	545	1956515.0	30010.33
38	2018-19	2852.1	20611.379555	555	2307726.0	35946.75
39	2019-20	2966.5	21008.542248	565	2218233.0	33557.62

So, there is no missing observation any more in the Commercial_Crops.

4.2.1.b. Interpolation of the Values of Support_Price_of_Foodgrains

The second observation on Support_Price_of_Foodgrains i.e. the observation corresponding to the year 1981-1982 are missing. As the observations before and after the missing observations are available we can use the interpolation technique to estimate the misssing value in this case.

In [11]:
```
#Using Interpolation for Support_Price_for_Foodgrains
regressor_data['Support_Price_for_Foodgrains']=regressor_data['Support_Price_for_Foo
regressor_data['Support_Price_for_Foodgrains'].head(5)
```

Out[11]:
```
0     930.0
1    1120.5
2    1311.0
3    1388.0
```

```
4    1249.0
Name: Support_Price_for_Foodgrains, dtype: float64
```

Hence, the estimated value corresponding to 1981-1982 is 1120.5

In [12]:
```
regressor_data.isnull().sum()
```

Out[12]:
```
Year                          0
Total_Foodgrains              0
Commercial_Crops              0
CrudeOil_POL_Prod_Imp         0
Total_Exports                 0
Total_Imports                 0
Tax_Reveneues                 0
Total_Savings_Deposits        0
Fiscal_Deficit                0
Gross_Market_Borrowing        0
Liabilities_of_RBI            0
Total_Expenditures            0
Currenc_with_the_Public       0
Investments_by_LIC            0
Support_Price_for_Foodgrains  0
Price_of_Gold                 0
dtype: int64
```

Thus, the above reveals that our data does not contain any missing value anymore.

4.2.2. Removal and Detection of Duplicate data:

The original data on regressors does not contain any duplicated row. So, we don't need to ake any remedial measure for it.

In [ ]:
```
#Detection of Duplicate data
gdp_data['Year'].duplicated().sum()
```

0

Finally we can claim that we have succesfully cleaned the data sets on Regressors and the dataset on GDP at Current Prices by estimating the missing values and by removing the duplicated observations and the outliers.

In [172…
```
Regression_Clean_data=regressor_data
```

In [175…
```
Regression_Clean_Data['GDP_at_Current_Prices']=gdp_data['GDP_at_Current_Prices']
```

In [179…
```
Regression_Clean_Data.shape
```

Out[179… (40, 17)

In [183…
```
import xlsxwriter
```

In [187…
```
file = pd.ExcelWriter('Regression_Clean_Data.xlsx')
Regression_Clean_Data.to_excel(file)
file.save()
```

5. OLS Fitting and Preliminary Analysis of the Cleaned Data

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from statsmodels.graphics.gofplots import ProbPlot
from sklearn.linear_model import LinearRegression
import statsmodels.api as sm
import scipy.stats as stats
from scipy.stats import norm
from scipy.stats import shapiro
import pylab
```

In [2]:
```
regression_data=pd.read_csv(r"Regression_Clean_Data.csv")
regression_data.head(2)
```

Out[2]:

	Year	Total_Foodgrains	Commercial_Crops	CrudeOil_POL_Prod_Imp	Total_Exports	Total_Imports
0	1980-81	1295.9	8752.7	57	31332.8	447.202
1	1981-82	1333.0	9376.6	72	31332.8	447.202

In [3]:
```
X=regression_data[['Total_Foodgrains','CrudeOil_POL_Prod_Imp','Total_Exports','Total
Y=regression_data['GDP_at_Current_Prices']
```

In [4]:
```
X1=sm.add_constant(X)#to add constant value in the model for the intercept term
```

Here we have fitted an MLR model using Ordinary Least Square method.

In [5]:
```
model=sm.OLS(Y,X1).fit()
```

In [6]:
```
predictions= model.summary()
predictions
```

Out[6]:

OLS Regression Results

Dep. Variable:	GDP_at_Current_Prices	**R-squared:**	1.000
Model:	OLS	**Adj. R-squared:**	1.000
Method:	Least Squares	**F-statistic:**	2.885e+04
Date:	Tue, 20 Apr 2021	**Prob (F-statistic):**	2.62e-47
Time:	09:19:19	**Log-Likelihood:**	-266.97
No. Observations:	40	**AIC:**	565.9
Df Residuals:	24	**BIC:**	593.0
Df Model:	15		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-2376.4158	612.710	-3.879	0.001	-3640.988	-1111.844
Total_Foodgrains	-0.0118	0.582	-0.020	0.984	-1.212	1.189
CrudeOil_POL_Prod_Imp	-18.4927	5.180	-3.570	0.002	-29.184	-7.802
Total_Exports	-0.0102	0.007	-1.509	0.144	-0.024	0.004

Total_Imports	-0.0153	0.322	-0.048	0.962	-0.679	0.649
Tax_Reveneues	0.0061	0.004	1.444	0.162	-0.003	0.015
Total_Savings_Deposits	-0.0033	0.003	-1.251	0.223	-0.009	0.002
Fiscal_Deficit	-0.0020	0.002	-0.901	0.376	-0.006	0.003
Gross_Market_Borrowing	-0.0088	0.004	-2.025	0.054	-0.018	0.000
Liabilities_of_RBI	0.0680	0.012	5.563	0.000	0.043	0.093
Total_Expenditures	0.0147	0.004	3.479	0.002	0.006	0.023
Currenc_with_the_Public	-0.0007	0.001	-0.504	0.619	-0.004	0.002
Investments_by_LIC	0.0009	0.001	1.126	0.271	-0.001	0.003
Support_Price_for_Foodgrains	0.1823	0.118	1.540	0.137	-0.062	0.427
Price_of_Gold	-0.3246	0.154	-2.103	0.046	-0.643	-0.006
Commercial_Crops	0.4362	0.084	5.219	0.000	0.264	0.609

Omnibus:	3.484	**Durbin-Watson:**	1.880
Prob(Omnibus):	0.175	**Jarque-Bera (JB):**	1.778
Skew:	-0.198	**Prob(JB):**	0.411
Kurtosis:	2.046	**Cond. No.**	3.15e+07

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 3.15e+07. This might indicate that there are
strong multicollinearity or other numerical problems.

Test for significance of the regressors:

H_0:β_1=.....=β15=0 against H_1:at least βj≠0 ; at least one j

F statistics = 2.885e+04 P_value = 2.62e-47 < 0.05,we reject the null hypothesis at 5% level of significance and conclude on the basis of the given data that all the variables are not insignificant in explaining the GDP data.

In [7]:
```
#Renaming for ease of graph plotting
dataframe = pd.concat([X, Y], axis=1)
model_fitted_Y = model.fittedvalues
model_residuals = model.resid    # model residuals
model_norm_residuals = model.get_influence().resid_studentized_internal  # normalize
```

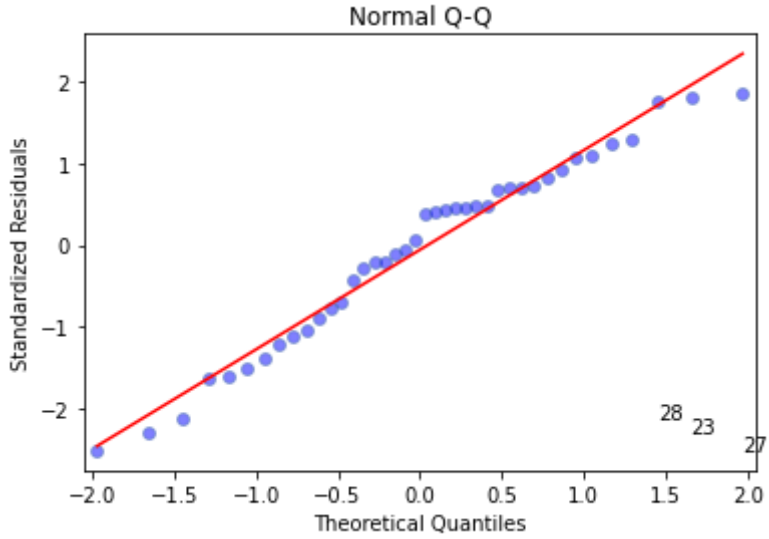
5.1.Inspection of the Normality Assumption of Errors

5.1.a. Q-Q Plot:

In this method we would plot the the ordered residuals $e_{(i)}$ against $\Phi^{-1}(\frac{i-0.5}{n})$, for i=1,2,...,n.If the errors are truely from Normal Distribution then the plot will be nearly a straight line.

In [8]:
```
#Using Normal Q-Q Plot
```

```
QQ = ProbPlot(model_norm_residuals)
plot_2 = QQ.qqplot(line='r', alpha=0.5, color='#4C72B0', lw=1)
plot_2.axes[0].set_title('Normal Q-Q')
plot_2.axes[0].set_xlabel('Theoretical Quantiles')
plot_2.axes[0].set_ylabel('Standardized Residuals');
# annotations
abs_norm_resid = np.flip(np.argsort(np.abs(model_norm_residuals)), 0)
abs_norm_resid_top_3 = abs_norm_resid[:3]
for r, i in enumerate(abs_norm_resid_top_3):
    plot_2.axes[0].annotate(i,
                                xy=(np.flip(QQ.theoretical_quantiles, 0)[r],
                                    model_norm_residuals[i]))
```

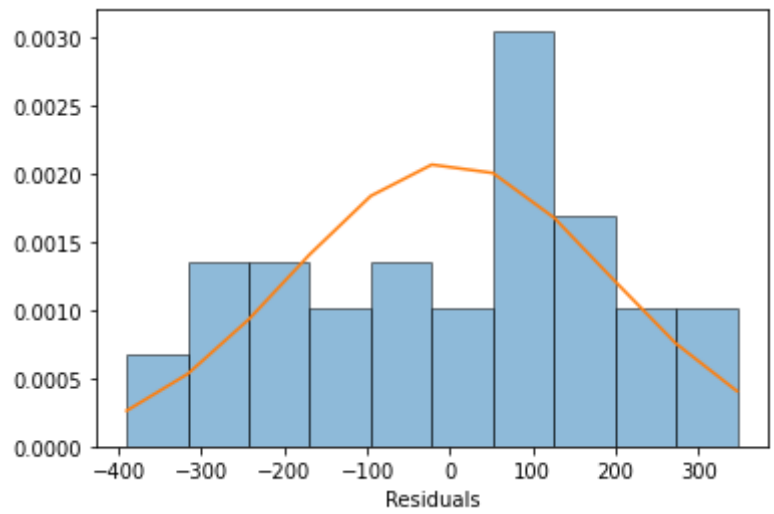


The above Q-Q plot yields almost a straight line.So, it can be concluded that the residuals can be assumed to follow a Normal Distribution which supports our assumption.
But we will use others methods to get check our assumption.

5.1.b. Histogram Approach:

In [9]:
```
#Histogram approach of normality checking
_, bins, _ = plt.hist(model.resid,density=1,alpha=0.5,ec='k')
plt.xlabel('Residuals')
mu, sigma = stats.norm.fit(model.resid)
best_fit_line = stats.norm.pdf(bins, mu, sigma)
plt.plot(bins, best_fit_line)
```

Out[9]:
```
[<matplotlib.lines.Line2D at 0x22c792df8e0>]
```



The histogram of Residuals is not significantly different from a Normal Curve. From here we could have concluded that our normality assumption for errors hold, but we will apply Shaipro-Wilk Test for Normality to get the final conclusion.

5.1.c. Shapiro-Wilk Test for Normality:

Here the null hypothesis is,
H0:Errors are normally distributed

The Alternative hypothesis is
H1:H0 is not true
The test Statistic is:

$$W = \frac{\sum_{i=1}^{n} a_i \hat{e_{(i)}}}{\sum_{i=1}^{n} (\hat{e_i} - \bar{e})^2}$$

Here, $\hat{e_i}$ are the ith fitted residual
$\hat{e_{(i)}}$ is the ith order statistic
$\bar{e}$ is the sample mean
$(a_1, \ldots, a_n) = \frac{m^T V^{-1}}{C}$
And , $C = (m^T V^{-1} V^{-1} m)^{\frac{1}{2}}$
Here m is made of the expected values of the order statistics of independent and identically distributed random variables sampled from the standard normal distribution; finally, V is the covariance matrix of those normal order statistics.If p-value is greater than chosen level of significance, null hypothesis is accepted(i.e.distribution of error is not significantly different from a normal population).

In [10]:
```
#Shapiro Wilk Test for normality
stat, p = shapiro(model.resid)
```

In [11]:
```
stat
```

Out[11]:
```
0.9673185348510742
```

In [12]:
```
p
```

Out[12]:
```
0.29499444365501404
```

Test statistic, W = 0.967, p-value = 0.294>0.05(α)
So we fail to reject the null hypothesis at 5% level of significance and conclude on the basis of the given data that the distribution of errors is not significantly different from Normal Distribution. So, our assumption is true.

5.2. Inspection of Homoscedastic Assumptions of Errors

5.2.a. Residual vs Fitted Plot:

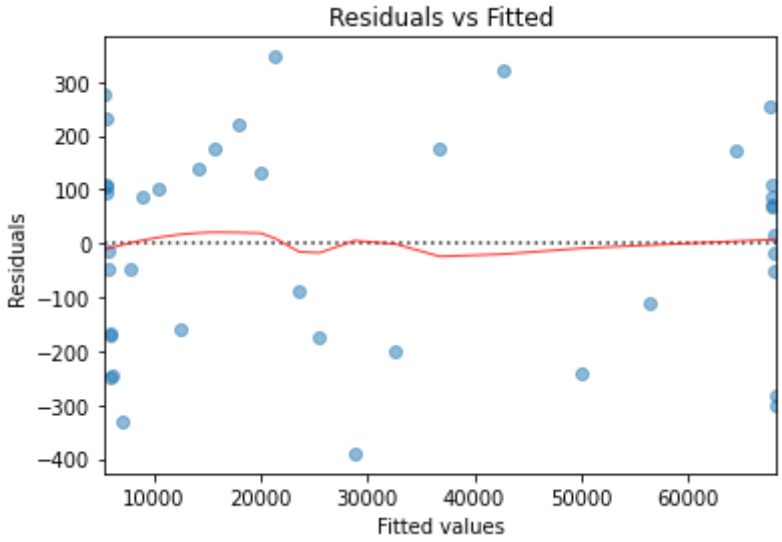
Here we plot the residuals against the fitted responses. If the errors are homoscedastic then we would expect a horizontal band and completely random pattern around $\hat{e_i} = 0$ line. If any

pattern is detected this will indicate that the variances may be non constant.

In [12]:
```
#Residual vs Fitted Plot
plot_1 = plt.figure()
plot_1.axes[0] = sns.residplot(model_fitted_Y,dataframe.columns[-1],data=dataframe,
                           lowess=True,
                           scatter_kws={'alpha': 0.5},
                           line_kws={'color': 'red', 'lw': 1, 'alpha': 0.8})

plot_1.axes[0].set_title('Residuals vs Fitted')
plot_1.axes[0].set_xlabel('Fitted values')
plot_1.axes[0].set_ylabel('Residuals')
```

```
c:\users\s\appdata\local\programs\python\python38\lib\site-packages\seaborn\_decorat
ors.py:36: FutureWarning: Pass the following variables as keyword args: x, y. From v
ersion 0.12, the only valid positional argument will be `data`, and passing other ar
guments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

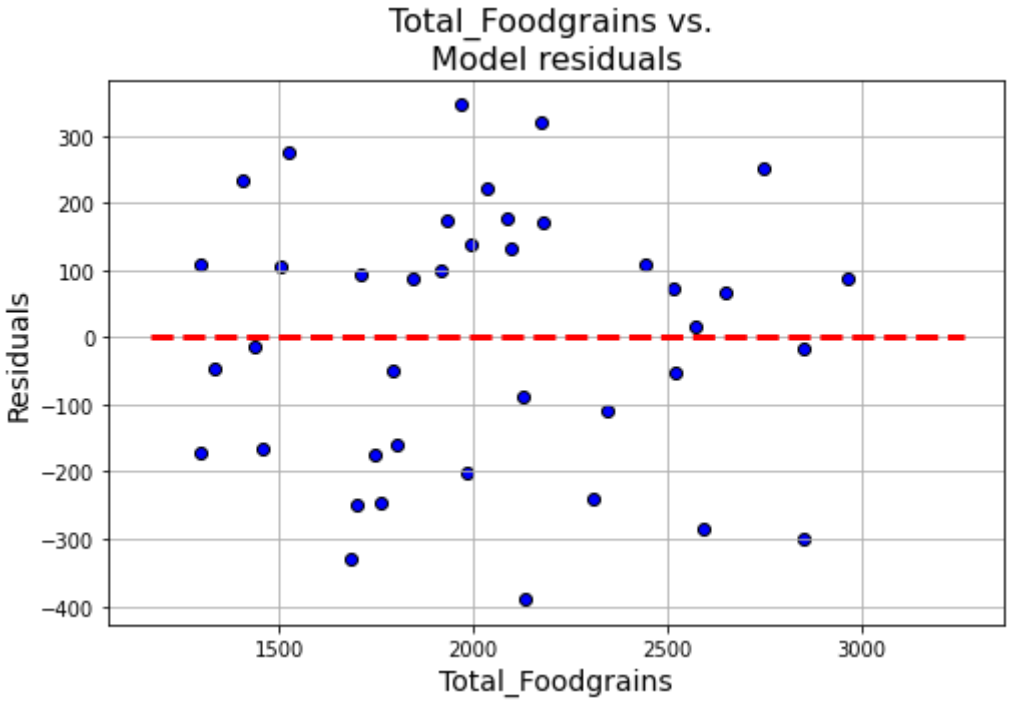
Out[12]:
```
Text(0, 0.5, 'Residuals')
```

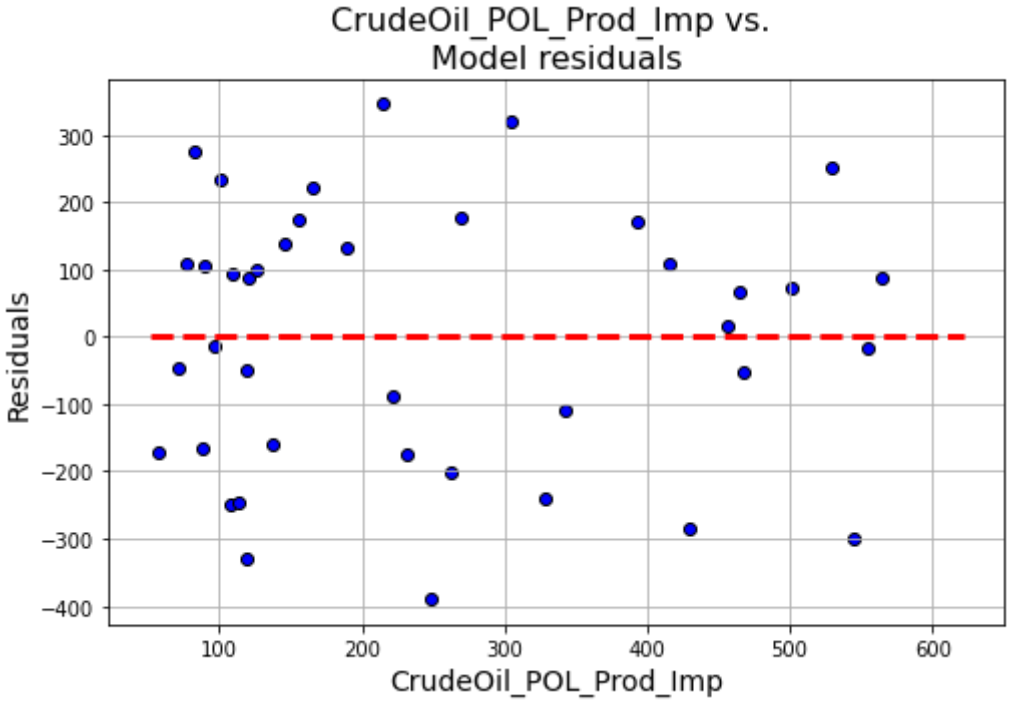


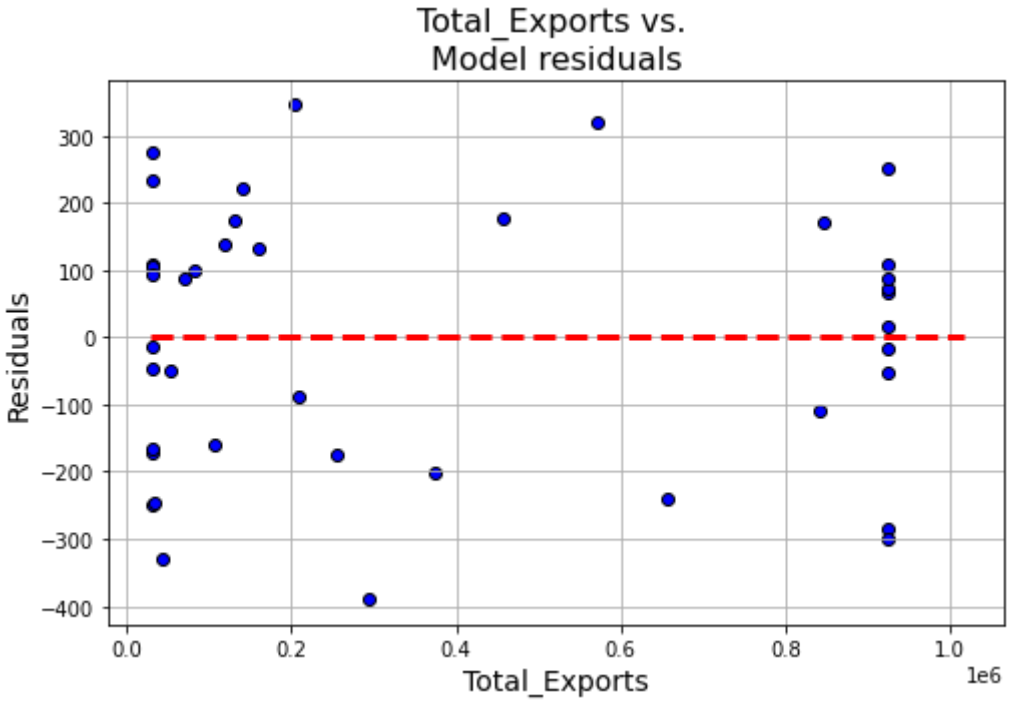
We have plotted the Residuals aginst the Fitted responses. We obtained a more or less random pattern among the residuals about the horizontal band. So we can conclude that, the assumption based on homoscedasticity is true in our Model.

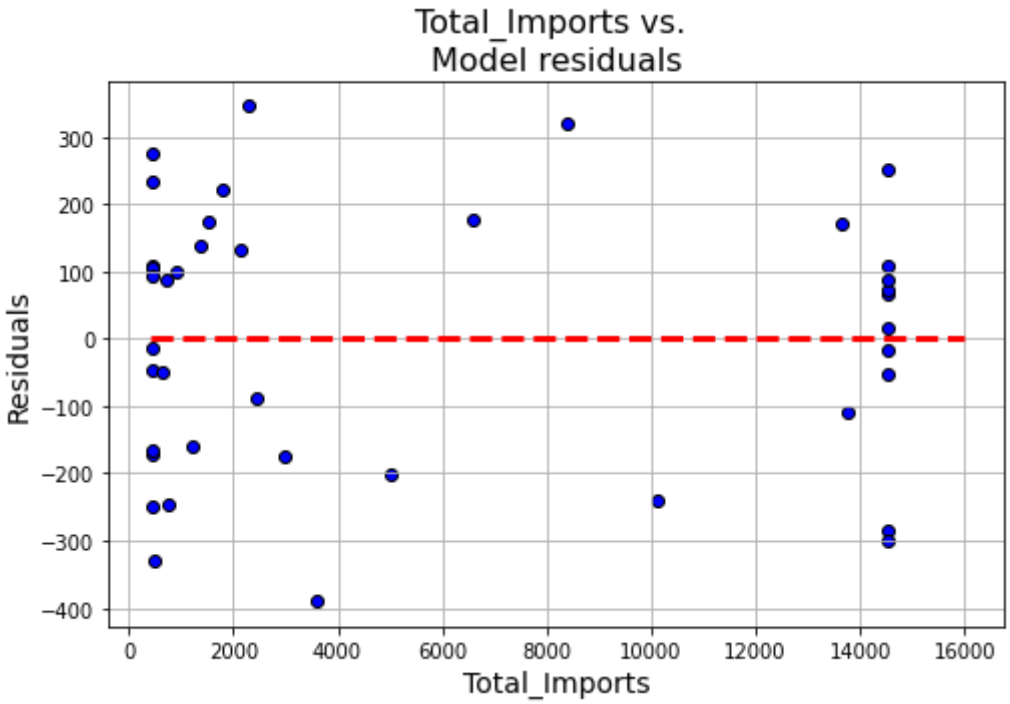
5.2.b. Residual VS Each Regressor Plot

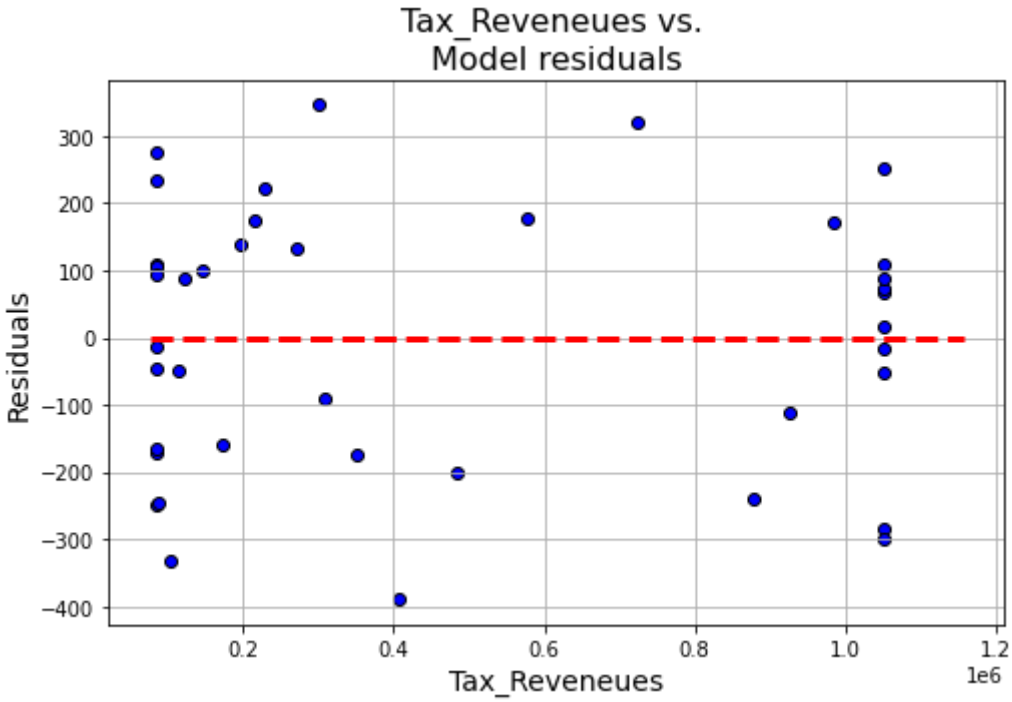
In [14]:
```
for c in dataframe.columns[:-1]:
    plt.figure(figsize=(8,5))
    plt.title("{} vs. \nModel residuals".format(c),fontsize=16)
    plt.scatter(x=dataframe[c],y=model.resid,color='blue',edgecolor='k')
    plt.grid(True)
    xmin=min(dataframe[c])
    xmax = max(dataframe[c])
    plt.hlines(y=0,xmin=xmin*0.9,xmax=xmax*1.1,color='red',linestyle='--',lw=3)
    plt.xlabel(c,fontsize=14)
    plt.ylabel('Residuals',fontsize=14)
    plt.show()
```

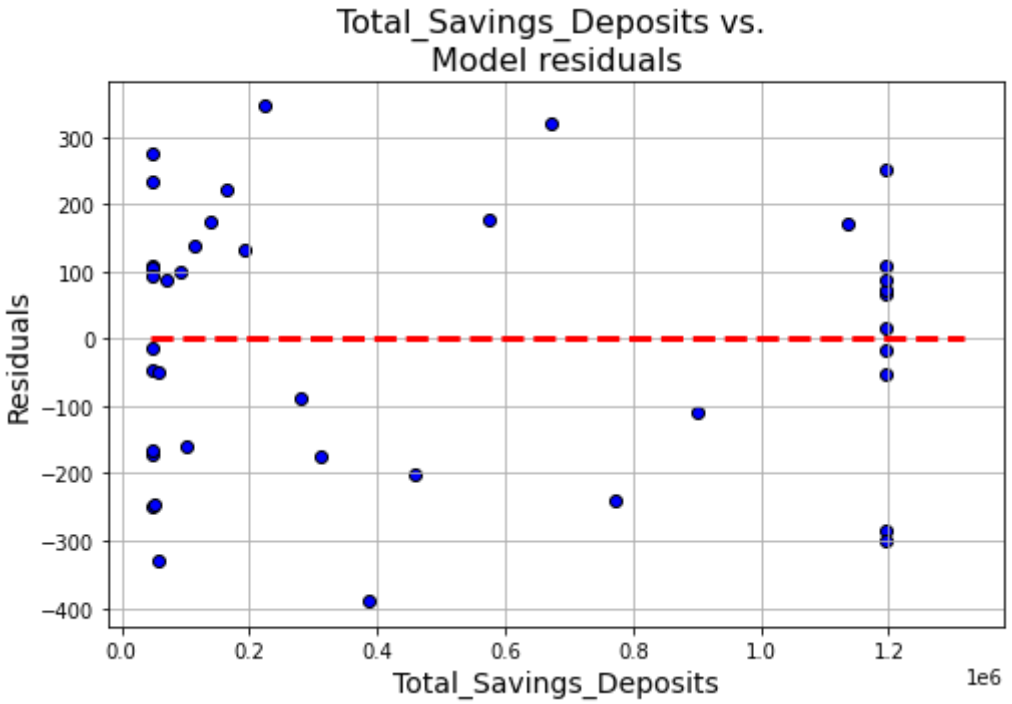

Total_Foodgrains vs.
Model residuals
Residuals
Total_Foodgrains
300
200
100
0
−100
−200
−300
−400
1500
2000
2500
3000

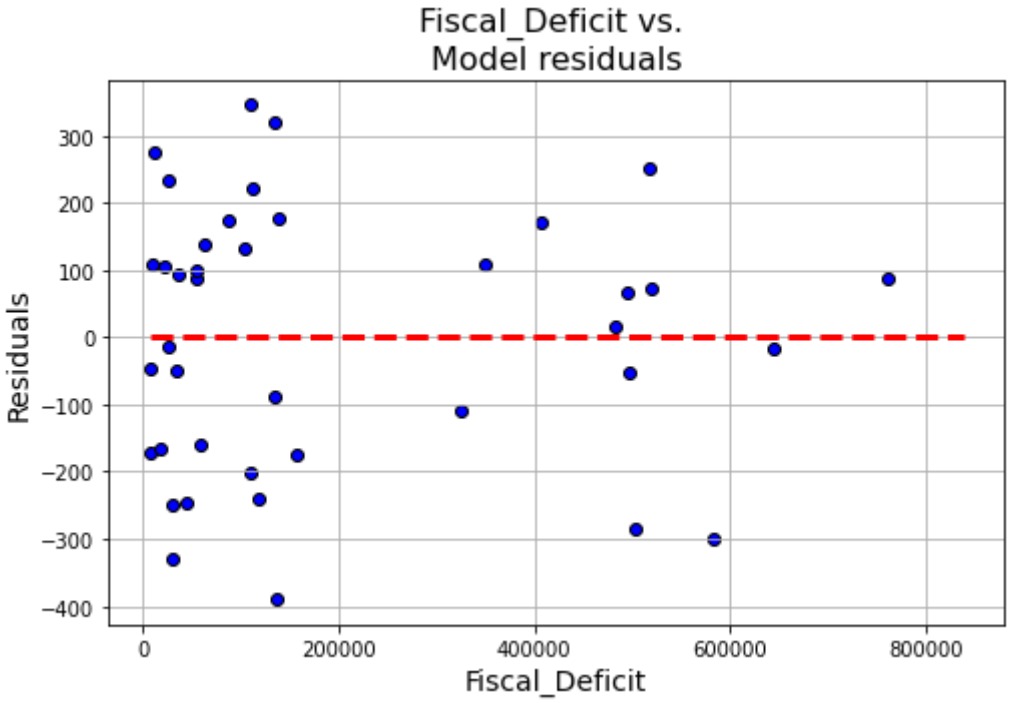

CrudeOil_POL_Prod_Imp vs.
Model residuals
Residuals
CrudeOil_POL_Prod_Imp
300
200
100
0
−100
−200
−300
−400
100
200
300
400
500
600

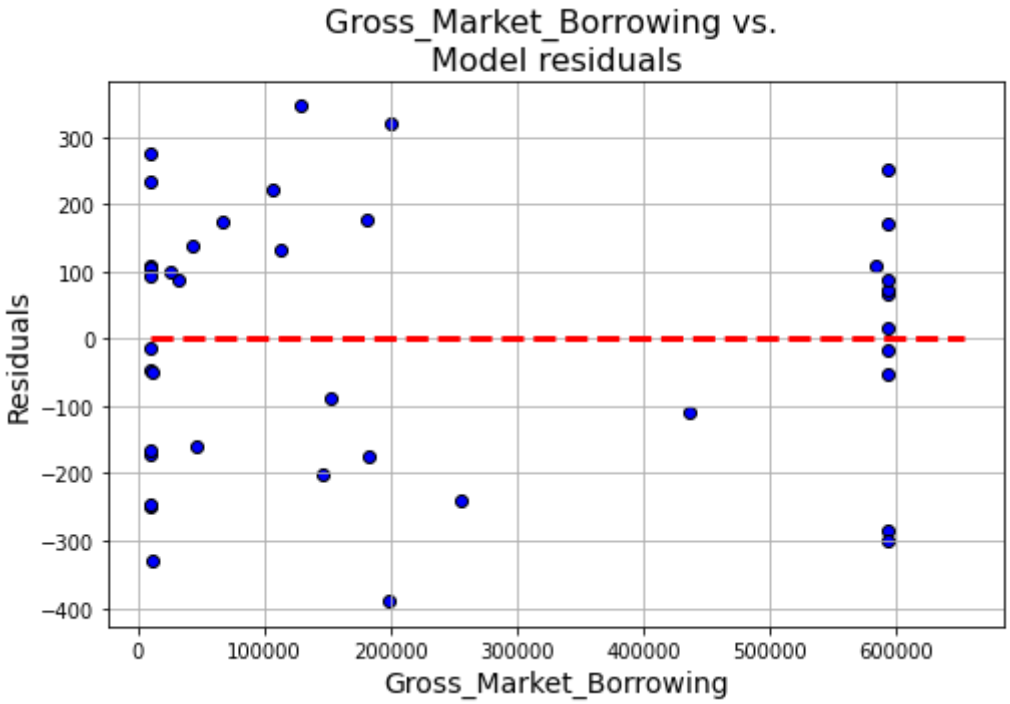

Total_Exports vs.
Model residuals
Residuals
Total_Exports
1e6

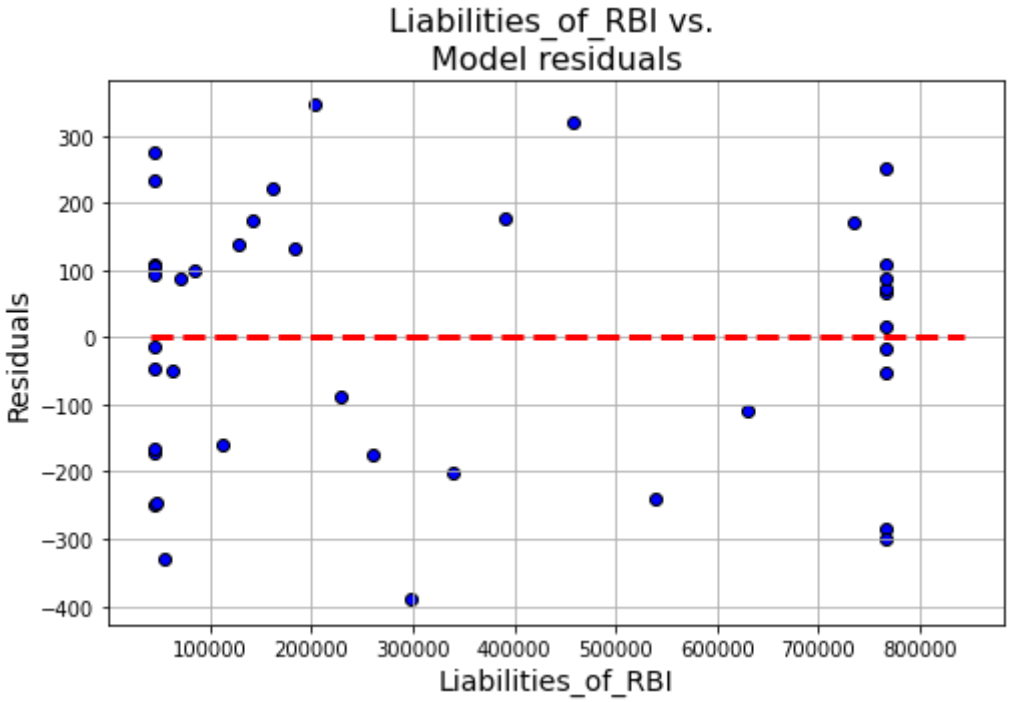

Total_Imports vs.
Model residuals
Residuals
Total_Imports

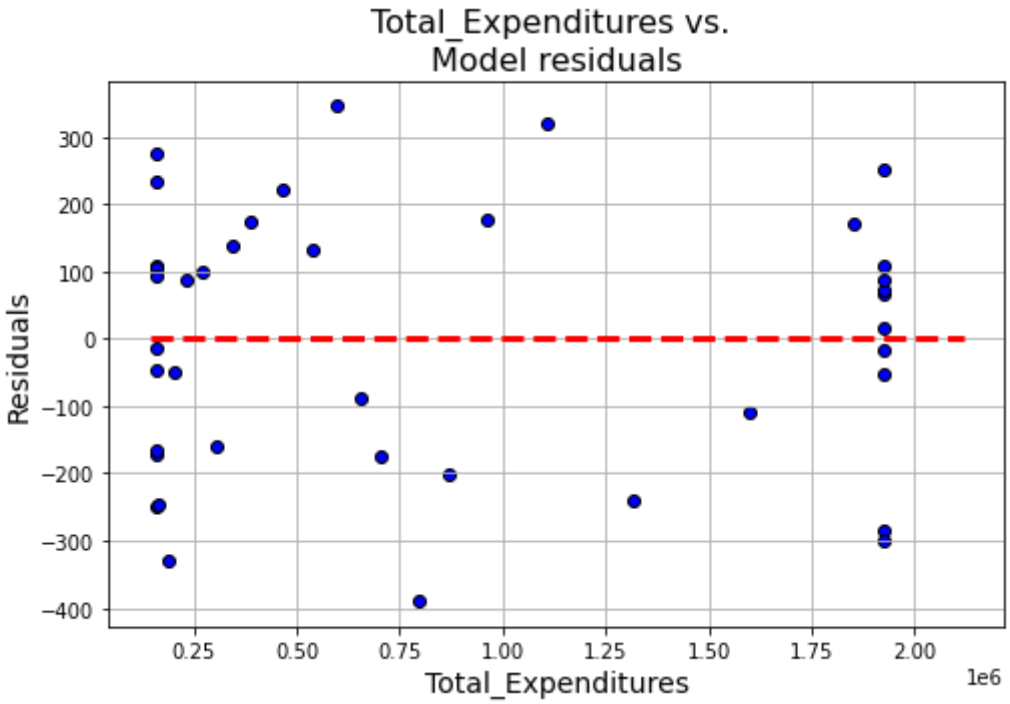

Tax_Reveneues vs.
Model residuals
Residuals
Tax_Reveneues
1e6

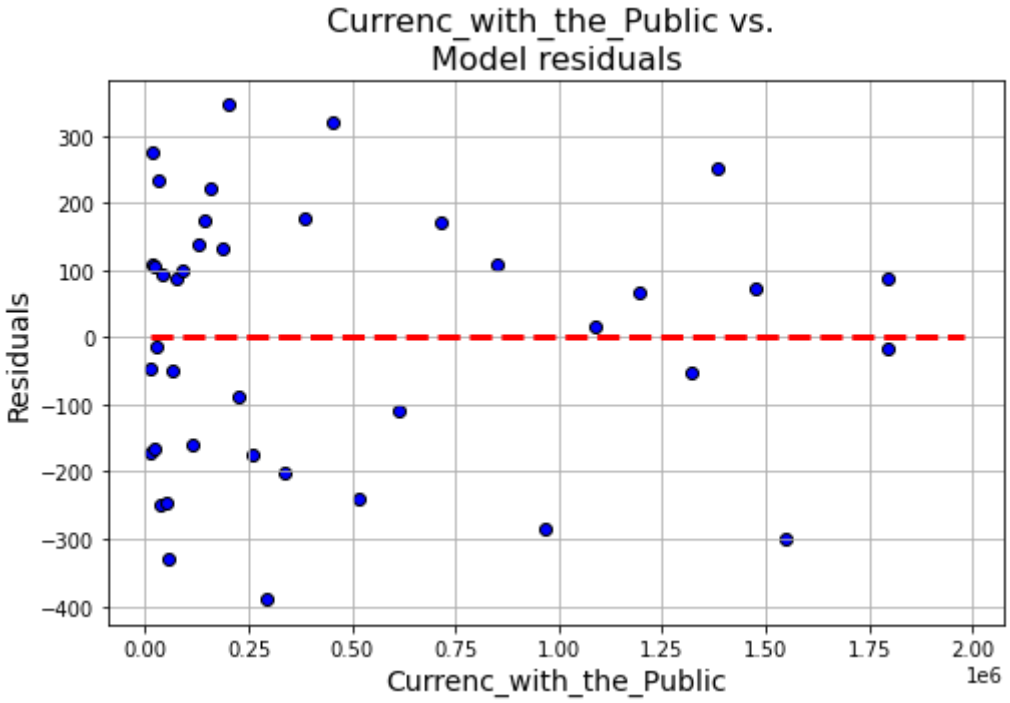

Total_Savings_Deposits vs.
Model residuals
Residuals
Total_Savings_Deposits
1e6

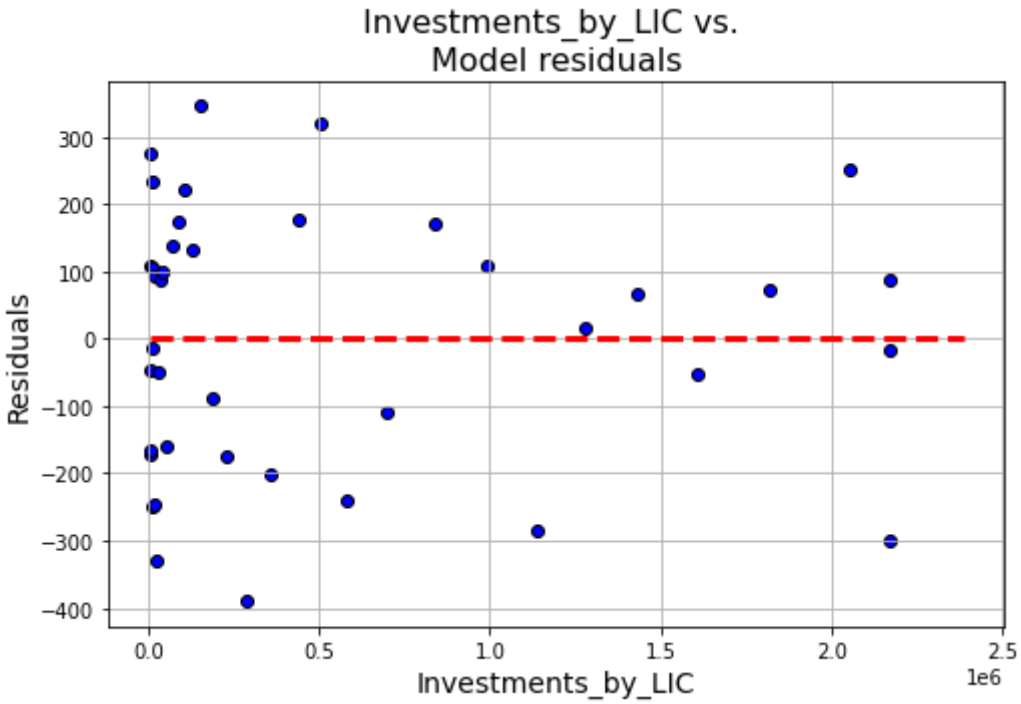

Fiscal_Deficit vs.
Model residuals
Residuals
Fiscal_Deficit
300
200
100
0
−100
−200
−300
−400
0
200000
400000
600000
800000

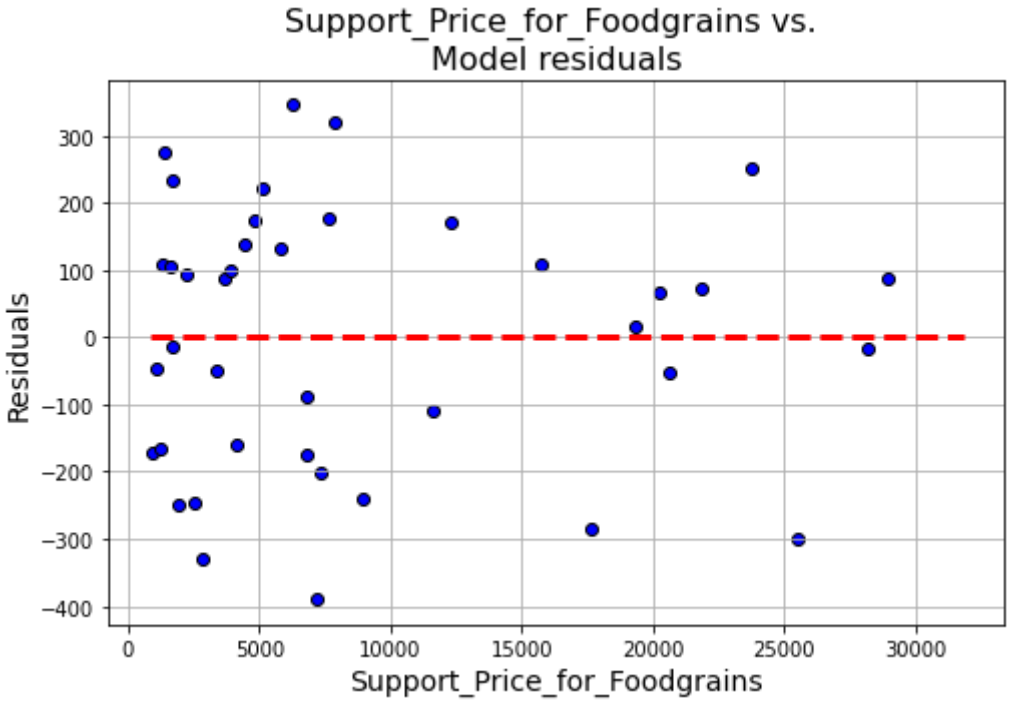

Gross_Market_Borrowing vs.
Model residuals
Residuals
Gross_Market_Borrowing
300
200
100
0
−100
−200
−300
−400
0
100000
200000
300000
400000
500000
600000

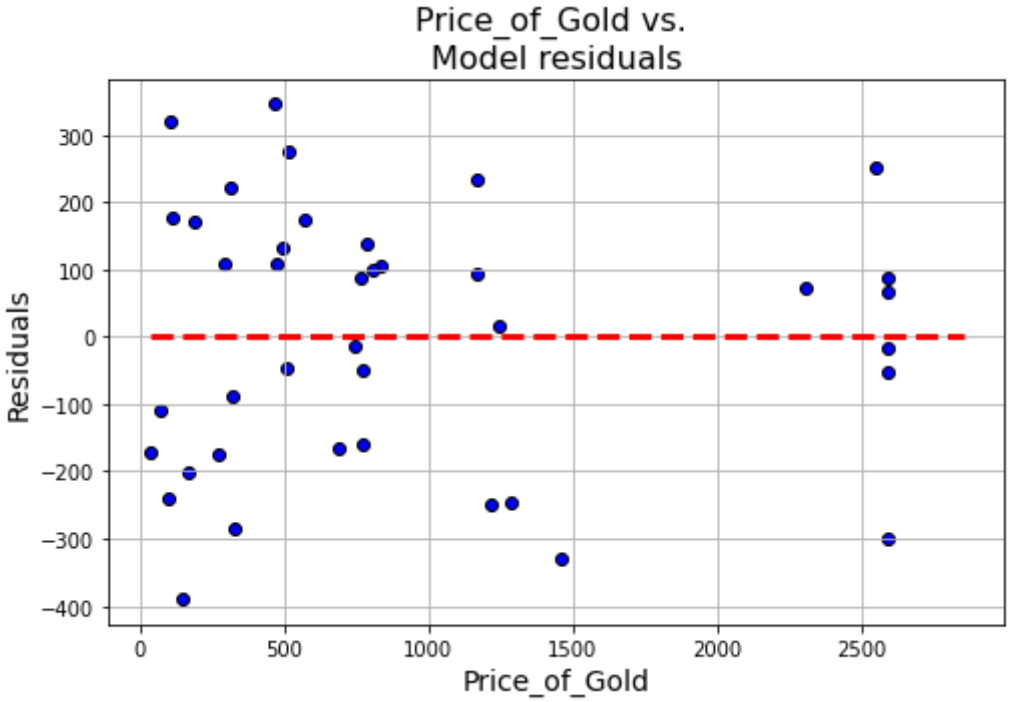

Liabilities_of_RBI vs.
Model residuals
Residuals
Liabilities_of_RBI
300
200
100
0
−100
−200
−300
−400
100000
200000
300000
400000
500000
600000
700000
800000

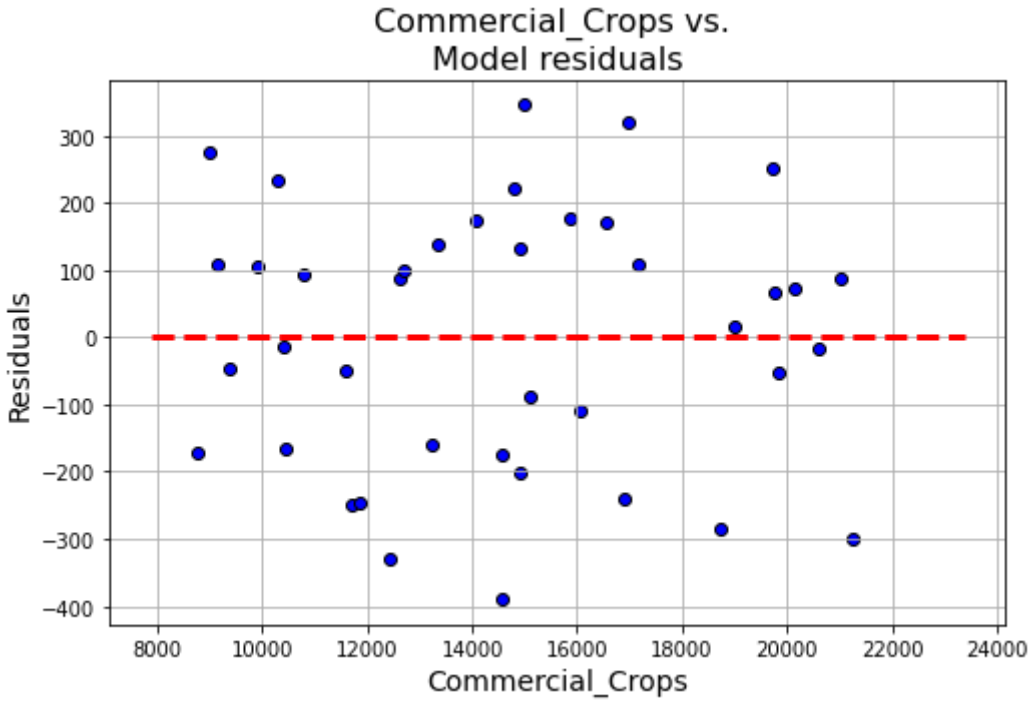

Total_Expenditures vs.
Model residuals
Residuals
Total_Expenditures
1e6
300
200
100
0
−100
−200
−300
−400
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00


Currenc_with_the_Public vs.
Model residuals
Residuals
Currenc_with_the_Public
1e6


Investments_by_LIC vs.
Model residuals
Residuals
Investments_by_LIC
1e6


Support_Price_for_Foodgrains vs.
Model residuals
Residuals
Support_Price_for_Foodgrains
300
200
100
0
−100
−200
−300
−400
0
5000
10000
15000
20000
25000
30000


Price_of_Gold vs.
Model residuals
Residuals
Price_of_Gold
300
200
100
0
−100
−200
−300
−400
0
500
1000
1500
2000
2500



Hence we can observe that the residual v/s regressor plot for each regressor exhibits random behaviour which supports our previous conlusion about homoscedasticity of errors.

5.2.c. Breusch-Pagan Test for Heteroscedasticity

To rationalize the test for heteroscedasticity we first note that the homoscedasticity assumption in OLS implies

$$Var(\varepsilon|x_1, x_2, ..., x_p) = \sigma^2 \; \forall \; i = 1, 2, ..., n$$

If we want to test for heteroscedasticity, our assumption is that the errors are actually homoscedastic i.e. our null hypothesis is:

$$H_0 : Var(\varepsilon|x_1, x_2, ..., x_p) = \sigma^2$$

Now, the expected value of the errors being zero

$$Var(\varepsilon|x_1, x_2, ..., x_p) = E(\varepsilon^2|x_1, x_2, ..., x_p) = \sigma^2$$

So, we can rewrite the null hypothesis as

$$H_0 : E(\varepsilon^2|x_1, x_2, ..., x_p) = \sigma^2$$

If we assume a simple linear relationship between ε with respect to the independent variables, we could then test the hypothesis. To see this, a model can be examined by regressing the squared residuals on the independent variables . Let e be the error term in the linear relationship, and assume that it is normally distributed with mean 0 given the independent variables .Then

$$\varepsilon^2 = \delta_0 + \delta_1 x_1 + \delta_2 x_2 + ... + \delta_p x_p + e$$

If homoscedasticity holds then we would have,

$$H_0 : \delta_1 = \delta_2 = ...= \delta_p = 0$$

$$\& H_1 : \delta_i \neq 0 \text{ for some } i$$

Of course we do not observe the true population error term, nor could we get a sample. However, we could use the residuals from the original OLS regression of y against $x_1, x_2, ..., x_p$, let that be $\hat{\varepsilon}^2$.The test statistic is dependent on the goodness of fit measure from the above regression. Let that be $R^2_{\hat{\varepsilon}^2}$.

Hence, Lagrange Multiplier statistic (LM) for Breusch-Pagan Test for Heteroscedasticity is given by,

$$LM = n \times R^2_{\hat{\varepsilon}^2}$$

The LM statistic is distributed asymptodically as χ^2_p .

We calculate the LM statistic, and the p-value using the χ^2_p distribution. If the p-value is smaller than the level of significance 0.05, we reject H_0 i.e. homoscedasticity.

In [17]:
```
from statsmodels.compat import lzip
import statsmodels.stats.api as sms
```

In [18]:
```
names = ['Lagrange multiplier statistic', 'p-value',
        'f-value', 'f p-value']
test = sms.het_breuschpagan(model.resid, model.model.exog)
lzip(names, test)##Model Homoscedastic
```

Out[18]:
```
[('Lagrange multiplier statistic', 18.09688221778359),
 ('p-value', 0.257606439712869),
 ('f-value', 1.3219584461150513),
 ('f p-value', 0.2632552221153009)]
```

p-value = 0.257>0.05(α)

So we fail to reject the null hypothesis at 5% level of significance and conclude on the basis of the given data that the distribution of errors is not heteroscedastic. So, our assumption is true.

5.3. Inspection of Autocorrelation among the Errors

In our model the value of Durbin-Watson Statistic is d=1.880. It indicates that there may exist a positive correlation among the errors as 0<1.88<2. Hence we would like to test,
H_0 : ρ = 0 ag. H_1 : ρ > 0 For n=40, p=15, α=0.05, d_L=0.678, d_U=2.557. So, d_L<d<d_U
Since the exact Durbin Watson test becomes inconclusive, we use the modified Durbin Watson test. Here d < $d_{U}, so H0 is rejected. i.e. there exists a positive autocorrelation among the errors.

Remedy:

We will fit the model with usual error assumption
$Y_i = \beta_0 + \beta_1 X_{i1} + \ldots + \beta_p X_{ip} + \gamma Y_{i-1} + \epsilon_i$, i=2(1)n

In [16]:
```
data=regression_data.drop(index=0)
X_new=data[['Total_Foodgrains','CrudeOil_POL_Prod_Imp','Total_Exports','Total_Import
data.head(2)
Y_new=data['GDP_at_Current_Prices']
Y1=regression_data['GDP_at_Current_Prices'].drop(index=39)
index=[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29
ind=pd.DataFrame(index)
Y1=pd.concat([ind,Y1],axis=1)
Y1=Y1.set_index(Y1[0])
Y1=Y1['GDP_at_Current_Prices']
X=pd.concat([X_new,Y1],axis=1)
X1=sm.add_constant(X)
model= sm.OLS(Y_new,X1).fit()
predictions= model.summary()
predictions
```

Out[16]:

OLS Regression Results

Dep. Variable:	GDP_at_Current_Prices	**R-squared:**	1.000
Model:	OLS	**Adj. R-squared:**	1.000
Method:	Least Squares	**F-statistic:**	2.505e+04
Date:	Wed, 14 Apr 2021	**Prob (F-statistic):**	4.34e-43
Time:	19:26:15	**Log-Likelihood:**	-260.10
No. Observations:	39	**AIC:**	554.2
Df Residuals:	22	**BIC:**	582.5
Df Model:	16		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
const	-2240.5376	651.414	-3.440	0.002	-3591.487	-889.589
Total_Foodgrains	-0.0150	0.608	-0.025	0.981	-1.275	1.245

CrudeOil_POL_Prod_Imp	-20.1193	7.065	-2.848	0.009	-34.770	-5.468
Total_Exports	-0.0105	0.007	-1.469	0.156	-0.025	0.004
Total_Imports	-0.0266	0.355	-0.075	0.941	-0.764	0.711
Tax_Reveneues	0.0063	0.004	1.408	0.173	-0.003	0.015
Total_Savings_Deposits	-0.0035	0.003	-1.192	0.246	-0.010	0.003
Fiscal_Deficit	-0.0018	0.002	-0.787	0.440	-0.006	0.003
Gross_Market_Borrowing	-0.0092	0.004	-2.064	0.051	-0.019	4.49e-05
Liabilities_of_RBI	0.0678	0.013	5.333	0.000	0.041	0.094
Total_Expenditures	0.0151	0.005	3.217	0.004	0.005	0.025
Currenc_with_the_Public	-0.0007	0.002	-0.446	0.660	-0.004	0.002
Investments_by_LIC	0.0011	0.001	1.269	0.218	-0.001	0.003
Support_Price_for_Foodgrains	0.1758	0.137	1.283	0.213	-0.108	0.460
Price_of_Gold	-0.3533	0.164	-2.160	0.042	-0.693	-0.014
Commercial_Crops	0.4344	0.091	4.759	0.000	0.245	0.624
GDP_at_Current_Prices	0.0082	0.049	0.169	0.867	-0.092	0.109

Omnibus:	2.869	**Durbin-Watson:**	2.010
Prob(Omnibus):	0.238	**Jarque-Bera (JB):**	1.587
Skew:	-0.184	**Prob(JB):**	0.452
Kurtosis:	2.083	**Cond. No.**	3.26e+07

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 3.26e+07. This might indicate that there are strong multicollinearity or other numerical problems.

In this case to detect the presence of autocorrelation we need to use Durbin's h Test.
$H_0 : \rho = 0$ vs $H_1 : NotH_0$
The test statistic is given by

$$h = \hat{\rho}\sqrt{\frac{40}{1-(40*V(\hat{\gamma}))}}$$

We will reject the Null hypothesis at 5% level of significnce if the observed h<-1.9596 or h>1.95996
Now, $\hat{\rho} = 1 - \frac{d}{2} = 1 - \frac{2.010}{2} = -0.005$
So, h=-0.033
Hence the absolute value of h is less that 1.95996, So we accept the Null Hypothesis at 5% level of Significance and conclude on the bais of the given data that the errors in our new model are independent.

5.4. Detection of Leverage Points

A Leverage point is an outlier in the x_space but it lies almost on the regression line passing through the rest of the sample points. But Leverage points do not affect the estimates of the Regression Coefficients. But these points can affect the standard errors, predicted values, and model summary statistics.

Let us denote,

$$H = X(X^TX)^{-1}X^T,$$

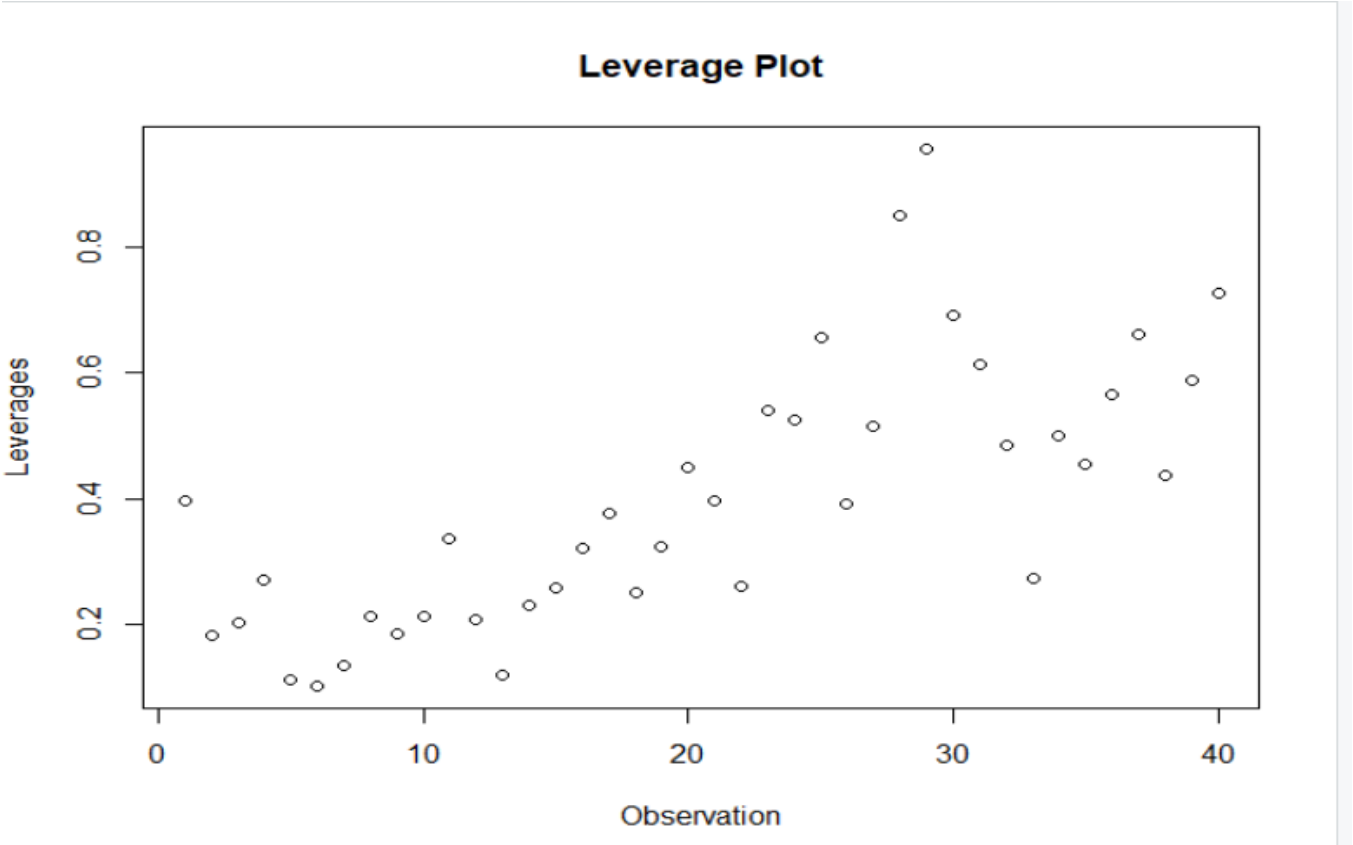
An observation will be considered as a Leverage point if $h_{ii} > \frac{2(p+1)}{n}$,

Here ,

h_{ii} is the ith diagonal element of matrix H

p is the number of regressors in the model

n is the total number of observations in the model.



In our data p=15, n= 40. Hence an observation will be considered to be a Leverage point if $h_{ii} > \frac{32}{40}$.

Here, it can be observed that,

```
> h_ii[h_ii>(32/40)]
       28        29
0.8512543 0.9562771
```

the 28th and the 29th observation are leverage Points in our data.

5.5. Detection of Influential Observations

An influential observation is an outlier in the y_space which is also moderately remote in x_space.. It forces the regression model to be moved towards its direction. It has a remarkable impact on the estimates of the model coefficients.

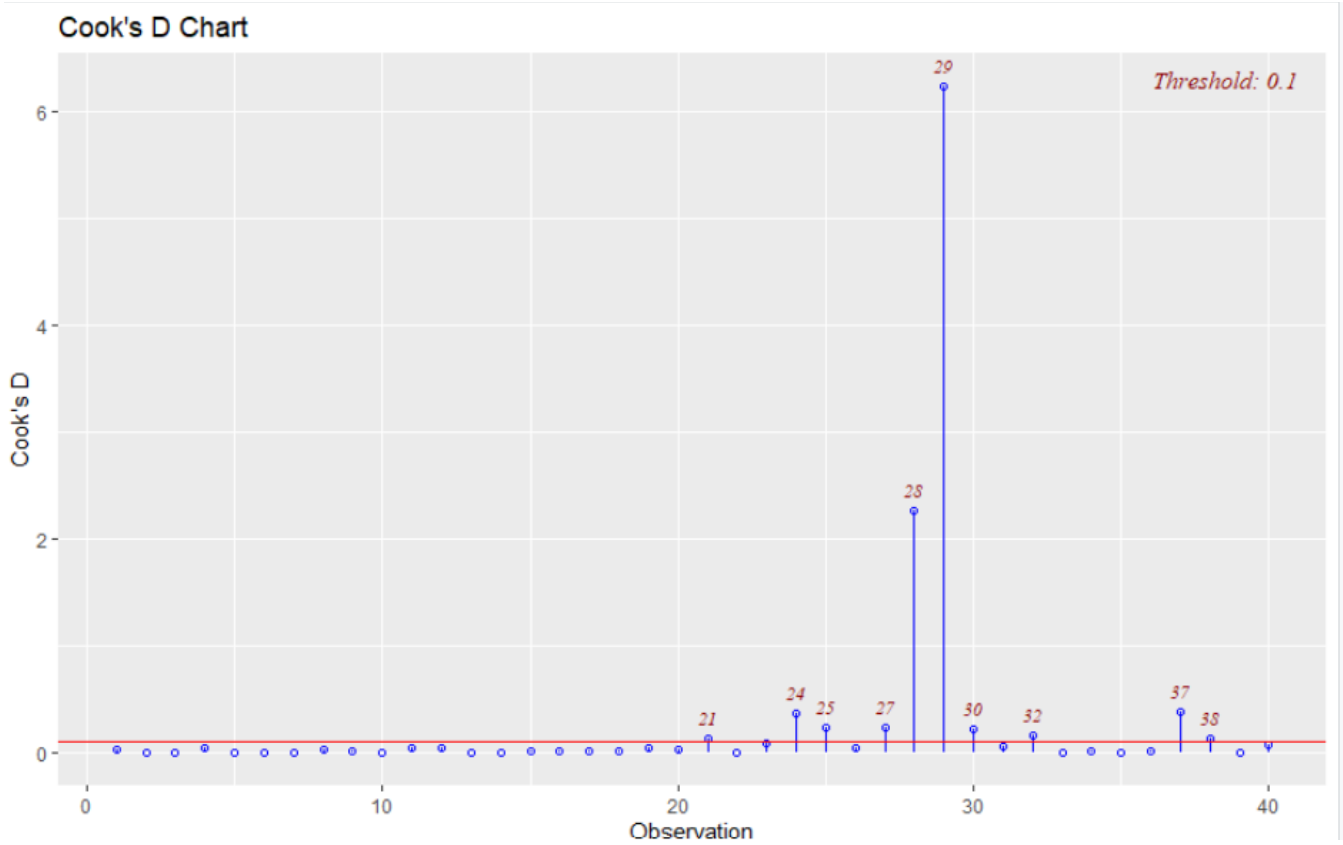
Cook's D Method:
Cook's Distance for the ith observation is calculated as,(i=1,2,...,n)

$$D_i = \frac{(\hat{\beta}_{(i)} - \hat{\beta})^T X^T X (\hat{\beta}_{(i)} - \hat{\beta})}{(p+1) MSRes}$$

Where , $\hat{\beta}_{(i)}$ is the estimate the estimate obtained by deleting the i th point

---

Any point having $D_i > 1$ i.e in our case for $D_i > 1$ is considered to be an influential point .



In this case the 28th and the 29th observations are Influential points. Previously we have observed these observations as Leverage Points also. So, in our data the Leverage points and the Influential points are the same which is not the case in general.

```
> v=cooks.distance(olsreg)
> v[v>1]
      28       29
2.263432 6.224324
>
```

But, as we have a time series data we will not remove any observation from our data because it will disturb the ordering of our data set.

6. Multicollinearity

Multicollinearity refers to a situation in which more than two explanatory variables in a multiple regression model are highly linearly related. There can be more than one reason behind multicollinearity, such as:

- The data collection method employed
- Model specification using too many regressors
- An over-defined model etc.

The consequences of multicollinearity being present in the model can be severe. When one or more regressors are linearly related with each other, the design matrix becomes ill-conditioned producing regression coefficients with large standard errors which can potentially damage the prediction capability of the model. There can be other problems like significant variable becoming insignificant one or regression coefficients appearing with wrong signs from what is expected.

6.1. Detection

There are several methods for knowing the presence of multicollinearity in the model. One such method is to calculate the VIFs of the model.

VIF or Variance Inflation Factor for the j-th regressor is defined as:

$\text{VIF}_j = \frac{1}{1-R_j^2}$, j = 1, 2, ..., p;

Where R_j^2 is the multiple R^2 obtained from regressing X_j on other regressors. The VIF value of 5 or more is an indicator of multicollinearity. Large values of VIF indicate multicollinearity leading to poor estimates of associated regression coefficients.

We started our initial analysis with 15 regressors. So there is a high likelihood of multicollinearity being present the preliminary model.

```
> vif(olsreg)
       x1         x2         x3         x4         x5         x6         x7         x8
 46.81389   62.64781  460.86270 4233.34613 2457.44631 1932.12707 1044.39448  142.64630
       x9        x10        x11        x12        x13        x14        x15
717.26553 8813.28123 6228.41098  405.41339  210.38359  634.03168   10.75055
```

As we can see from the above R snippets, all the VIFs are unusually high indicating the presence of multicollinearity in the model.

6.2. Multicollinearity Diagnostics with Variance Decomposition

After knowing the presence of multicollinearity in our model, we would like to know the group(s) of variables responsible for it. For doing this we can use Variance Decomposition Method.

Variance Decomposition Method is a method to identify subsets that are involved in multi-collinearity. Variance decomposition proportions, defined as

$$\pi_{kj} = \frac{\frac{v_{kj}^2}{l_k}}{\sum_{k=1}^{p} \frac{v_{kj}^2}{l_k}}, \forall k, j = 1(1)p$$

where, $l_1, l_2, \dots, l_p$ are eigen values of $X^T X$ and $v_1, v_2, \dots, v_p$ are corresponding orthonormal eigen vectors and $v_j = \left(v_{j1}, v_{j2}, \dots, v_{jp}\right)^T, j = 1(1)p$.

Now a variance decomposition table is formed with the π_{kj} values along with a column containing the corresponding condition indices arranged in ascending order. So, large proportion in a row corresponding to the maximum condition index indicates the presence of multicollinearity among the corresponding regressors.

Step 1:

```
> eigprop(olsreg)

Call:
eigprop(mod = olsreg)

   Eigenvalues       CI (Intercept)     x1     x2     x3     x4     x5     x6     x7     x8     x9    x10    x11
1      14.5143   1.0000      0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
2       0.9743   3.8596      0.0015 0.0002 0.0002 0.0000 0.0000 0.0000 0.0000 0.0000 0.0001 0.0000 0.0000 0.0000
3       0.4204   5.8759      0.0001 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0002 0.0000 0.0000 0.0000
4       0.0464  17.6872      0.0000 0.0005 0.0004 0.0003 0.0001 0.0004 0.0001 0.0004 0.0116 0.0002 0.0000 0.0000
5       0.0230  25.1175      0.0018 0.0000 0.0003 0.0009 0.0001 0.0000 0.0005 0.0000 0.0598 0.0102 0.0000 0.0000
6       0.0090  40.2179      0.1890 0.0028 0.0102 0.0067 0.0000 0.0014 0.0002 0.0004 0.0018 0.0000 0.0001 0.0001
7       0.0058  50.1006      0.0074 0.0049 0.0001 0.0050 0.0013 0.0067 0.0009 0.0011 0.1156 0.0366 0.0001 0.0004
8       0.0024  77.2889      0.0724 0.1129 0.0080 0.0025 0.0003 0.0070 0.0011 0.0073 0.0053 0.0061 0.0004 0.0002
9       0.0017  92.7298      0.0044 0.0449 0.0239 0.0659 0.0000 0.0002 0.0003 0.0000 0.3701 0.0287 0.0001 0.0000
10      0.0009 125.4711      0.0465 0.0442 0.3205 0.0405 0.0012 0.0101 0.0005 0.1732 0.0228 0.0671 0.0004 0.0000
11      0.0008 137.4815      0.4066 0.1612 0.1174 0.1352 0.0093 0.0039 0.0000 0.1647 0.0241 0.0000 0.0002 0.0015
12      0.0005 167.5131      0.0505 0.2945 0.2719 0.1857 0.0114 0.0026 0.0086 0.0694 0.0003 0.0321 0.0002 0.0008
13      0.0002 242.4584      0.0827 0.0027 0.1045 0.3637 0.0148 0.1006 0.3914 0.1299 0.2698 0.0186 0.0110 0.0128
14      0.0001 316.9815      0.0061 0.1936 0.0687 0.1342 0.1481 0.1025 0.3652 0.0924 0.0520 0.1962 0.0488 0.0867
15      0.0001 477.5934      0.0079 0.1360 0.0591 0.0291 0.7207 0.6629 0.1577 0.0009 0.0115 0.5580 0.0163 0.2807
16      0.0000 657.0986      0.1231 0.0016 0.0149 0.0303 0.0926 0.1017 0.0734 0.3603 0.0551 0.0460 0.9225 0.6167
      x12    x13    x14    x15
1  0.0000 0.0000 0.0000 0.0001
2  0.0001 0.0002 0.0000 0.0014
3  0.0002 0.0006 0.0000 0.0589
4  0.0025 0.0031 0.0021 0.2566
5  0.0029 0.0192 0.0001 0.0083
6  0.0028 0.0571 0.0041 0.1060
7  0.0041 0.0614 0.0000 0.0119
8  0.1702 0.1922 0.0102 0.0133
9  0.1652 0.1742 0.0780 0.0115
10 0.0572 0.0552 0.0370 0.0663
11 0.0386 0.1104 0.1288 0.0573
12 0.2971 0.0152 0.3626 0.0024
13 0.1613 0.1652 0.0139 0.3030
14 0.0564 0.1079 0.1838 0.1019
15 0.0414 0.0028 0.1491 0.0011
16 0.0000 0.0353 0.0303 0.0000

==============================
Row 15==> x4, proportion 0.720738 >= 0.50
Row 15==> x5, proportion 0.662897 >= 0.50
Row 15==> x9, proportion 0.558011 >= 0.50
Row 16==> x10, proportion 0.922514 >= 0.50
Row 16==> x11, proportion 0.616668 >= 0.50
```

•So the subsets (x4,x5,x9) and (x10,x11) are involved in Multicollinearity.

• In the first subset VIF of x4 is the highest and in the second subset the VIF of x10 is highest.

• We drop the variables x4 and x10, and again fit a model.

Step 2:

```
> eigprop(olsreg_1)

Call:
eigprop(mod = olsreg_1)

   Eigenvalues       CI (Intercept)     x1     x2     x3     x5     x6     x7     x8     x9    x11    x12    x13    x14    x15
1      12.5821   1.0000      0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0002
2       0.9461   3.6467      0.0018 0.0002 0.0002 0.0000 0.0000 0.0000 0.0000 0.0001 0.0000 0.0000 0.0001 0.0002 0.0000 0.0009
3       0.3856   5.7120      0.0001 0.0000 0.0000 0.0000 0.0001 0.0000 0.0001 0.0001 0.0001 0.0000 0.0001 0.0005 0.0000 0.0708
4       0.0430  17.1007      0.0000 0.0007 0.0004 0.0003 0.0025 0.0003 0.0015 0.0107 0.0008 0.0001 0.0029 0.0035 0.0029 0.2506
5       0.0225  23.6347      0.0020 0.0000 0.0002 0.0010 0.0004 0.0008 0.0001 0.0691 0.0111 0.0000 0.0029 0.0203 0.0000 0.0054
6       0.0088  37.7811      0.2252 0.0039 0.0119 0.0071 0.0040 0.0004 0.0004 0.0021 0.0000 0.0003 0.0031 0.0606 0.0054 0.1080
7       0.0053  48.5316      0.0078 0.0057 0.0002 0.0055 0.0375 0.0021 0.0006 0.1224 0.0509 0.0006 0.0071 0.0674 0.0001 0.0135
8       0.0024  72.9735      0.0832 0.1480 0.0115 0.0040 0.0331 0.0013 0.0141 0.0014 0.0033 0.0006 0.1882 0.2051 0.0113 0.0107
9       0.0017  86.4463      0.0055 0.0581 0.0289 0.0693 0.0004 0.0005 0.0003 0.3799 0.0332 0.0000 0.1667 0.1705 0.1025 0.0112
10      0.0009 117.6117      0.0975 0.0254 0.3804 0.0639 0.0444 0.0009 0.2746 0.0158 0.0749 0.0000 0.0704 0.0812 0.0704 0.0773
11      0.0007 132.5513      0.3680 0.1514 0.1373 0.1516 0.0763 0.0042 0.4729 0.0249 0.0018 0.0044 0.0151 0.1036 0.0976 0.0526
12      0.0005 161.9748      0.1202 0.5546 0.2864 0.1396 0.0107 0.0001 0.0672 0.0022 0.0395 0.0002 0.4168 0.0032 0.6732 0.0004
13      0.0002 235.9963      0.0605 0.0236 0.1389 0.3421 0.7872 0.5546 0.1412 0.3235 0.0056 0.0303 0.1238 0.1887 0.0002 0.2999
14      0.0001 362.6236      0.0283 0.0285 0.0037 0.2156 0.0033 0.4347 0.0270 0.0477 0.7787 0.9634 0.0027 0.0952 0.0363 0.0985

==============================
Row 12==> x1, proportion 0.554590 >= 0.50
Row 13==> x5, proportion 0.787193 >= 0.50
Row 13==> x6, proportion 0.554642 >= 0.50
Row 14==> x9, proportion 0.778674 >= 0.50
Row 14==> x11, proportion 0.963377 >= 0.50
Row 12==> x14, proportion 0.673214 >= 0.50
```

•Here the subsets (x1,x14), (x5,x6), (x9,x11) are involved in Multicollinearity.

•

```
> vif(olsreg_1)
      x1       x2       x3       x5       x6       x7       x8       x9      x11      x12      x13      x14      x15
38.61694 56.48854 460.85626 621.85164 1652.31401 565.79407 137.52537 591.53293 3228.08915 380.93840 204.23134 495.56974 10.43363
```

x14, x6 and x11 have highest VIFs in their respective subsets.

• So, we remove these variables and fit the model again.

Step 3:

```
eigprop(mod = olsreg_2)

   Eigenvalues       CI (Intercept)     x1     x2     x3     x5     x7     x8     x9    x12    x13    x15
1       9.6746   1.0000      0.0001 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0006
2       0.9204   3.2421      0.0028 0.0004 0.0002 0.0000 0.0001 0.0000 0.0003 0.0002 0.0002 0.0004 0.0009
3       0.3329   5.3908      0.0002 0.0001 0.0000 0.0000 0.0007 0.0002 0.0000 0.0006 0.0002 0.0005 0.1688
4       0.0339  16.8898      0.0001 0.0012 0.0006 0.0010 0.0168 0.0038 0.0227 0.0118 0.0112 0.0137 0.4354
5       0.0204  21.7878      0.0014 0.0000 0.0003 0.0024 0.0081 0.0017 0.1622 0.0397 0.0041 0.0324 0.0062
6       0.0078  35.2310      0.4663 0.0110 0.0248 0.0291 0.0078 0.0000 0.0026 0.0008 0.0014 0.0583 0.1461
7       0.0049  44.2409      0.0190 0.0125 0.0002 0.0107 0.1933 0.0002 0.1649 0.2952 0.0150 0.0941 0.0230
8       0.0022  65.7433      0.1450 0.2030 0.0106 0.0032 0.0633 0.0423 0.0218 0.0037 0.4106 0.4328 0.0089
9       0.0014  84.1036      0.0982 0.3480 0.0059 0.1921 0.0000 0.0327 0.5104 0.3897 0.4862 0.0402 0.0782
10      0.0008 107.3002      0.0018 0.4238 0.5130 0.0339 0.4458 0.4329 0.0209 0.1546 0.0010 0.0498 0.0790
11      0.0006 123.1490      0.2652 0.0000 0.4444 0.7276 0.2641 0.4860 0.0941 0.1037 0.0702 0.2778 0.0528

==============================
Row 10==> x2, proportion 0.512999 >= 0.50
Row 11==> x3, proportion 0.727643 >= 0.50
Row 9==> x8, proportion 0.510404 >= 0.50
>
```

• We can see (x2,x3) and x8 are involved in Multicollinearity.

•

```
> vif(olsreg_2)
       x1        x2        x3        x5        x7        x8        x9       x12       x13       x15
 24.429281 47.004906 232.064254 168.507669 410.857145 80.023514 127.646846 218.335755 149.228620  5.609401
>
```

●The VIFs of x3 ,x2 and x8 is very much higher than 5.

• We drop those variables and again fit the model.

Step 4:

```
eigprop(mod = olsreg_3)

  Eigenvalues      CI (Intercept)     x1     x5     x7     x9    x12    x13    x15
1      6.8916  1.0000      0.0001 0.0001 0.0001 0.0000 0.0002 0.0001 0.0001 0.0015
2      0.7409  3.0499      0.0064 0.0021 0.0002 0.0001 0.0007 0.0004 0.0007 0.0045
3      0.3208  4.6349      0.0011 0.0007 0.0010 0.0005 0.0015 0.0003 0.0008 0.2002
4      0.0291 15.3989      0.0031 0.0121 0.0122 0.0035 0.0419 0.0380 0.0497 0.6223
5      0.0089 27.7576      0.0000 0.0015 0.1411 0.0105 0.6736 0.0043 0.0032 0.0333
6      0.0048 37.8963      0.7652 0.6668 0.0012 0.0001 0.0082 0.0409 0.1612 0.0777
7      0.0025 52.5806      0.2144 0.3162 0.0032 0.0004 0.0237 0.9124 0.7790 0.0023
8      0.0014 71.1480      0.0095 0.0005 0.8412 0.9848 0.2502 0.0036 0.0052 0.0581

================================
Row 6==> x1, proportion 0.666773 >= 0.50
Row 8==> x5, proportion 0.841159 >= 0.50
Row 8==> x7, proportion 0.984826 >= 0.50
Row 5==> x9, proportion 0.673586 >= 0.50
Row 7==> x12, proportion 0.912364 >= 0.50
Row 7==> x13, proportion 0.778982 >= 0.50
Row 4==> x15, proportion 0.622316 >= 0.50
```

• Here x1,x9 and x15 has proportions higher than 0.50 and the subsets (x5,x7) , (x12,x13) are involved in multicollinearity.

•

```
> vif(olsreg_3)
       x1        x5        x7        x9       x12       x13       x15
 7.010616 139.075096 246.162504  53.632205 125.136346 110.232012  4.599277
```

• Here x7, x12 have highest VIFs in their corresponding subsets and the VIFs of x1 and x9 are higher than 5.

• We remove x1, x7,x12 and x9 and again fit the model .

Step 5:

```
eigprop(mod = olsreg_4)

  Eigenvalues      CI (Intercept)     x5    x13    x15
1      3.2878  1.0000      0.0124 0.0036 0.0027 0.0084
2      0.4543  2.6902      0.2699 0.0117 0.0168 0.0043
3      0.2415  3.6896      0.0899 0.0495 0.0003 0.2753
4      0.0164 14.1651      0.6278 0.9352 0.9803 0.7121

================================
Row 4==> x5, proportion 0.935183 >= 0.50
Row 4==> x13, proportion 0.980265 >= 0.50
Row 4==> x15, proportion 0.712051 >= 0.50
```

• Although the highest condition index is 14.165, but the proportion of variability of x5,x13 and x15 contributed by the Eigen value 0.0164 are much higher than 0.5. We will check for the VIFs to decide whether to keep or drop any variable from the subset (x5,x13,x15).

•

```
> vif(olsreg_4)
       x5       x13       x15
11.776451 19.516452  4.148475
```

• The VIFs of x13 is highest in its corresponding subset.we remove it and again fit the model.

Step_6:

```
eigprop(mod = olsreg_5)

  Eigenvalues     CI (Intercept)     x5    x15
1      2.4508 1.0000      0.0541 0.0544 0.0488
2      0.3095 2.8141      0.6342 0.6754 0.0004
3      0.2397 3.1977      0.3117 0.2702 0.9509

================================
Row 2==> x5, proportion 0.675385 >= 0.50
Row 3==> x15, proportion 0.950891 >= 0.50
```

• Here the highest condition index is 3.1977 <15. Now, as we can see after 6 steps, variance decomposition method is not able to provide us a valid set of regressors, almost all the important regressors are going out of our model. So we resort to Variable Selection Technique as the next step.

7. Variable Selection

When we fit a MLR model, we use the p-value in the ANOVA table to determine whether the model, as a whole, is significant. A natural question arises which regressors, among a larger set of all potential regressors, are important. We could use the individual p-values of the regressors and refit the model with only significant terms. But the p-values of the regressors are adjusted for the other terms in the model. So, picking out the subset of significant regressors can be somewhat challenging. This procedure of identifying the best subset of regressors to include in the model, among all possible subsets of regressors, is referred to as *variable selection*.

One approach is to start with a model containing only the intercept. Then using some chosen model fit criterion we slowly add terms to the model, one at a time, whose inclusion gives the most statistically significant improvement of the the model, and repeat this process until none improves the model to a statistically significant extent. This procedure is referred to as *forward selection*.

Another alternative is *backward elimination*. Here we start with the full model, then based on some model fit criterion we slowly remove variables one at a time, whose deletion gives the most statistically insignificant deterioration of the model fit, and repeat this process until no further variables can be deleted without a statistically insignificant loss of fit.

A third classical approach is *stepwise selection*. This is a combination of forward selection (FS) and backward elimination (BE). We start with FS, but at each step we recheck all regressors already entered, for possible deletion by BE method, this is because of the fact that regressor added at an earlier step may now be unnecessary in presence of new regressor.

Here we use stepwise selection method based on partial F-test & AIC criterions to determine the best subset model.

7.1. On The Basis Of Partial F-Test

```
> library(olsrr)
> Step <- ols_step_both_p(olsreg,pent=0.05,prem=0.05)
> Step
```

Stepwise Selection Summary

Step	Variable	Added/ Removed	R-Square	Adj. R-Square	C(p)	AIC	RMSE
1	x10	addition	0.999	0.999	368.0030	652.8739	806.4422
2	x7	addition	1.000	1.000	111.7600	614.0954	490.8973
3	x2	addition	1.000	1.000	57.8640	596.7490	390.7640
4	x15	addition	1.000	1.000	28.3010	581.4421	319.2114
5	x7	removal	1.000	1.000	32.5700	583.5270	331.2353

As we can see from the above stepwise selection summary we are losing most of our important variables, hence we go for stepwise selection based on *Information Theoretic Criterion* to obtain a better model.

7.2. On The Basis Of Information Theoretic Criterion

Our MLR model is

$$Y = X\beta + \varepsilon$$

Where we assume that $\varepsilon \sim N(0, \sigma^2)$ and $Y \sim N_n(X\beta, \sigma^2 I_n)$

The likelihood function given by,

$$L(\beta, \sigma^2 | y) = (2n)^{-n/2}(\sigma^2)^{-n/2} exp\{-\frac{1}{2}(y - X\beta)^T(y - X\beta)\}$$

So, the general form of the penalized likelihood function is given by,

$$-2\ln \hat{L} + \text{penalty term}$$
$$= nln(SSRes) + \text{penalty term}$$

. Where,

$$\hat{L} = max_{\beta,\sigma^2} L(\beta, \sigma^2 | y) = L(\hat{\beta}_{mle}, \widehat{\sigma^2}_{mle})$$

Akaike Information Criterion

The Akaike information criterion (AIC) is a measure of the relative quality of statistical models for a given dataset. Given a collection of models for the data, AIC estimates the quality of each model, relative to each of the other models. Hence, AIC provides a means for model selection.

AIC is founded on information theory: it offers a relative estimate of the information lost when a given model is used to represent the process that generates the data. In doing so, it deals with the trade-off between the goodness of fit of the model and the complexity of the model.

AIC does not provide a test of a model in the sense of testing a null hypothesis, so it can tell nothing about the absolute quality of the model.If all the candidate models fit poorly, AIC will not give any warning of that.

Definition

Suppose that we have a statistical model of some n data. Then the AIC value of the model is the given by,

$AIC = -2 \ln(\hat{L}) + 2k$

Where

K = The number of estimated parameters in the model

$\hat{L}$ = The maximized value of the likelihood function for the model

At first we consider all the subset models excluding one regressor at a time, and calculate the AIC value for each of those subset models. Then we discard the variable for which the subset model has the minimum AIC value.

```
> AIC<-stepAIC(olsreg,direction="both")
Start:  AIC=452.42
Y ~ x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9 + x10 + x11 +
    x12 + x13 + x14 + x15

       Df Sum of Sq     RSS    AIC
- x1    1        25 1468130 450.42
- x5    1       139 1468244 450.43
- x12   1     15543 1483648 450.85
- x8    1     49701 1517806 451.76
<none>              1468105 452.42
- x13   1     77606 1545711 452.48
- x7    1     95781 1563886 452.95
- x6    1    127497 1595602 453.76
- x4    1    139213 1607319 454.05
- x14   1    145003 1613108 454.19
- x9    1    250854 1718959 456.73
- x15   1    270498 1738603 457.19
- x11   1    740378 2208484 466.76
- x3    1    779621 2247726 467.46
- x2    1   1666482 3134587 480.77
- x10   1   1892813 3360918 483.55
```

The method considered the Full 15 parameter model in the first step.
The AIC corresponding to the Full Model is 452.42.
In this step this method compares the AICs by discarding each variable from the full model with the AIC of the full model.
From the table it can be observed that, the AIC corresponding to the model with 14 regressors after discarding the X1 variable is lower than the full model and also it is minimum among all 14 regressor model.

```
Step:  AIC=450.42
Y ~ x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9 + x10 + x11 + x12 +
    x13 + x14 + x15

       Df Sum of Sq     RSS    AIC
- x5    1       250 1468380 448.43
- x12   1     18486 1486616 448.93
- x8    1     50495 1518625 449.78
<none>              1468130 450.42
- x13   1     83068 1551198 450.63
- x7    1     96103 1564233 450.96
- x6    1    127586 1595716 451.76
+ x1    1        25 1468105 452.42
- x4    1    162747 1630877 452.63
- x14   1    255196 1723326 454.84
- x9    1    256234 1724364 454.86
- x15   1    272320 1740450 455.23
- x11   1    740980 2209110 464.77
- x3    1    785241 2253371 465.56
- x10   1   1893115 3361245 481.56
- x2    1   2337930 3806060 486.53
```

This step considers the subset model by discarding X1 from the full model. The AIC corresponding to that model is 450.42.
AIC will be calculated after discarding each of the variable from the current subset model. The AIC corresponding to the model after discarding X5 from the current subset model is minimum, and in the next step this variable will be deleted from the model.

```
Step:  AIC=448.43
Y ~ x2 + x3 + x4 + x6 + x7 + x8 + x9 + x10 + x11 + x12 + x13 +
    x14 + x15

       Df Sum of Sq     RSS    AIC
- x12   1     18237 1486617 446.93
- x8    1     62875 1531255 448.11
<none>              1468380 448.43
- x13   1     86024 1554403 448.71
- x7    1     98957 1567336 449.04
+ x5    1       250 1468130 450.42
+ x1    1       136 1468244 450.43
- x6    1    159034 1627414 450.54
- x14   1    272410 1740790 453.24
- x9    1    285095 1753475 453.53
- x15   1    294668 1763047 453.75
- x4    1    580056 2048436 459.75
- x11   1    743331 2211711 462.82
- x3    1    819618 2287998 464.17
- x10   1   2160486 3628865 482.62
- x2    1   2382497 3850877 485.00
```

This step considers the subset model by discarding X5 and X1 from the full model. The AIC corresponding to that model is 448.43.
AIC will be calculated after discarding each of the variable from the current subset model. The AIC corresponding to the model after discarding X12 from the current subset model is minimum, and in the next step this variable will be deleted from the model.

```
Step:  AIC=446.93
Y ~ x2 + x3 + x4 + x6 + x7 + x8 + x9 + x10 + x11 + x13 + x14 +
    x15

       Df Sum of Sq     RSS    AIC
- x13   1     68253 1554870 446.72
<none>              1486617 446.93
- x7    1    100741 1587357 447.55
- x8    1    117027 1603644 447.96
+ x12   1     18237 1468380 448.43
- x6    1    142968 1629585 448.60
+ x1    1      2375 1484242 448.86
+ x5    1         1 1486616 448.93
- x9    1    267206 1753823 451.54
- x14   1    277337 1763954 451.77
- x15   1    384969 1871585 454.14
- x4    1    579626 2066243 458.09
- x11   1    732026 2218643 460.94
- x3    1    811637 2298254 462.35
- x10   1   2142466 3629083 480.62
- x2    1   2530283 4016900 484.69
```

This step considers the subset model by discarding X12,X5 and X1 from the full model. The AIC corresponding to that model is 446.93. AIC will be calculated after discarding each of the variable from the current subset model.The AIC corresponding to the model after discarding X13 from the current subset model is minimum, and in the next step this variable will be deleted from the model.

```
Step:  AIC=446.72
Y ~ x2 + x3 + x4 + x6 + x7 + x8 + x9 + x10 + x11 + x14 + x15

       Df Sum of Sq     RSS    AIC
<none>              1554870 446.72
- x7    1     79764 1634634 446.72
+ x13   1     68253 1486617 446.93
+ x5    1      3233 1551637 448.64
+ x1    1      2394 1552476 448.66
+ x12   1       466 1554403 448.71
- x8    1    169883 1724753 448.87
- x6    1    175518 1730388 449.00
- x9    1    225939 1780808 450.15
- x15   1    325555 1880424 452.33
- x4    1    557702 2112572 456.98
- x11   1    664033 2218902 458.95
- x14   1    737505 2292374 460.25
- x3    1    764862 2319732 460.72
- x10   1   2183122 3737992 479.81
- x2    1   2867182 4422051 486.53
```

This step considers the subset model by discarding X13,X12, X5 and X1 from the full model. The AIC corresponding to that model is 446.72. AIC will be calculated after discarding each of the variable from the current subset model.

If any one of the variables is discarded from the current subset model the AIC is higher than the current model. So , no variable will be discarded any more, the current model is our final model.
So, our best subset model chosen by AIC is given by,

```
lm(formula = Y ~ x2 + x3 + x4 + x6 + x7 + x8 + x9 + x10 + x11 +
    x14 + x15, data = my_data)
```

Now we inspect the estimate of parameters and adjusted R-squared of the fitted model.

```
> summary(AIC)

Call:
lm(formula = Y ~ x2 + x3 + x4 + x6 + x7 + x8 + x9 + x10 + x11 +
    x14 + x15, data = my_data)

Residuals:
    Min      1Q  Median      3Q     Max
-430.54 -145.50    6.35  139.17  378.36

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -2.237e+03  4.767e+02  -4.692 6.44e-05 ***
x2           4.055e-01  5.643e-02   7.186 8.05e-08 ***
x3          -1.548e+01  4.171e+00  -3.711 0.000906 ***
x4          -9.661e-03  3.048e-03  -3.169 0.003682 **
x6           6.055e-03  3.406e-03   1.778 0.086293 .
x7          -2.932e-03  2.446e-03  -1.198 0.240770
x8          -2.898e-03  1.657e-03  -1.749 0.091235 .
x9          -7.379e-03  3.658e-03  -2.017 0.053365 .
x10          6.836e-02  1.090e-02   6.270 8.88e-07 ***
x11          1.328e-02  3.839e-03   3.458 0.001758 **
x14          2.053e-01  5.634e-02   3.644 0.001081 **
x15         -3.122e-01  1.290e-01  -2.421 0.022197 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 235.7 on 28 degrees of freedom
Multiple R-squared:  0.9999,    Adjusted R-squared:  0.9999
F-statistic: 4.333e+04 on 11 and 28 DF,  p-value: < 2.2e-16
```

As we can observe from the above summary, the adjusted R-squared of the model is 0.999. Now, we want to keep all the regressors in our selected model but their may be presence of multicollinearity.
So, we check for the presence of Multicollinearity by looking at the corresponding VIFs.

7.3. Multicollinearity Detection After AIC

Variable	VIF
X2	31.463143
X3	329.083696
X4	943.456362
X6	1379.285097
X7	1003.416278
X8	91.394157
X9	566.061453
X10	7713.605173
X11	5677.063718
X14	158.120383
X15	8.264733

As all the VIFs are higher than 5, we can say that the selected Subset model is also suffering from Multicollinearity.
Now as we obtained the best subset by AIC, we will keep all the variables in our model. For removal of multicollinearity we will use *Ridge Regression*.

8. Ridge Regression

Ridge regression is a model tuning method that is used to analyze any data that suffers from multicollinearity. This method performs L2 regularization. L2 regularization adds an L2 penalty, which equals the square of the magnitude of coefficients. coefficients are shrunk by the same factor (so none are eliminated).

A tuning parameter (λ) controls the strength of the penalty term. When λ = 0, ridge regression equals least squares regression. If λ = ∞, all coefficients are shrunk to zero. The ideal penalty is therefore somewhere in between 0 and ∞.

Ridge estimators theoretically produce new estimators that are shrunk closer to the "true" population parameters.

The ridge function fitting the ridge regression is given by,

$R(\beta) = min_{\beta}\ (Y-X\beta)^T(Y-X\beta) + \lambda\ \beta^T\beta$

OLS regression uses the following formula to estimate coefficients:

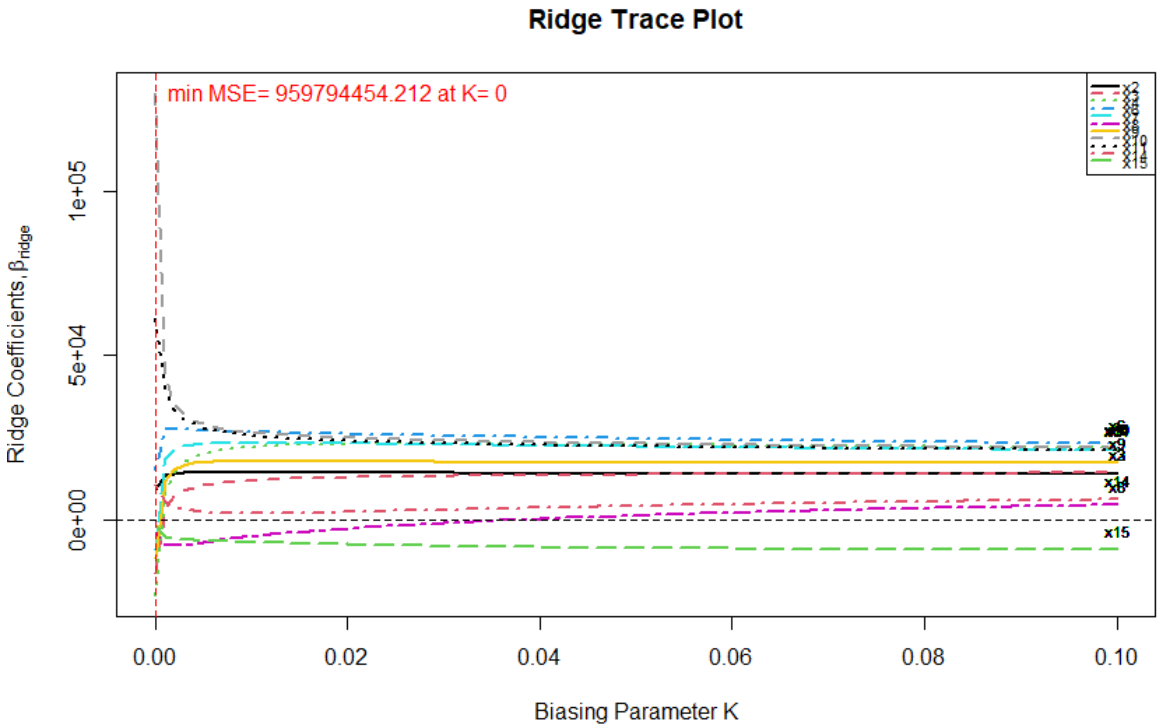
$\hat{\beta} = (X^TX)^{-1}X^TY$

Ridge regression adds a product of *ridge parameter* & the identity matrix to the cross product matrix (X^TX), forming a new matrix ($X^TX + \lambda I$). The new formula is used to find the coefficients:

$\tilde{\beta} = (X^TX+\lambda I)^{-1}X^TY$

To choose the value of λ we've used a graphical method called *ridge trace plot,* a plot of estimated coefficients against a shrinkage parameter, to determine a favorable trade-off of bias against precision(inverse variance) of the estimates.

RIDGE TRACE PLOT



From the above plot it seems that the estimates of coefficients stabilizes for some value of λ between 0.02 and 0.04.

From the *ridge trace plot* we choose that value of λ for which VIFs all get stabilized (i.e.<5). The estimate of λ obtained by this method is 0.035. Hence we fit a new model with this value of λ and inspect its adjusted R-squared value.

```
Coefficients: for Ridge parameter K= 0.035
             Estimate Estimate (Sc) StdErr (Sc) t-value (Sc) Pr(>|t|)
Intercept -3.4707e+03   -6.3794e+10  8.2773e+08     -77.0717   <2e-16 *
x2         6.0940e-01    1.4275e+04  1.3597e+03      10.4984   <2e-16 *
x3         1.3252e+01    1.3584e+04  1.0826e+03      12.5468   <2e-16 *
x4         9.8000e-03    2.3250e+04  8.2210e+02      28.2815   <2e-16 *
x6         9.9000e-03    2.5452e+04  7.9735e+02      31.9202   <2e-16 *
x7         7.5000e-03    2.2816e+04  7.2750e+02      31.3621   <2e-16 *
x8        -2.0000e-04   -2.1583e+02  1.3668e+03      -0.1579   0.8754
x9         1.1500e-02    1.7660e+04  1.2922e+03      13.6669   <2e-16 *
x10        1.2600e-02    2.3932e+04  3.8415e+02      62.2984   <2e-16 *
x11        5.0000e-03    2.3099e+04  4.4678e+02      51.7006   <2e-16 *
x14        6.6600e-02    3.5010e+03  1.2390e+03       2.8257   0.0078
x15       -1.5265e+00   -8.0193e+03  8.7001e+02      -9.2174   <2e-16 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Ridge Summary
        R2     adj-R2   DF ridge          F        AIC        BIC
   0.99000    0.98650    4.08678 5906.78547  519.49407  673.95133

P-value for F-test ( 4.08678 , 34.96489 ) = 4.655804e-49
```

VIFs for the new fitted model are :

```
             x2      x3      x4      x6      x7      x8      x9     x10      k=0.035
4.53822 2.87705 1.65893 1.56054 1.29912 4.58522 4.09836 0.36222
             x11       x14     x15
           0.48996 3.76807 1.85794
```

Observation:

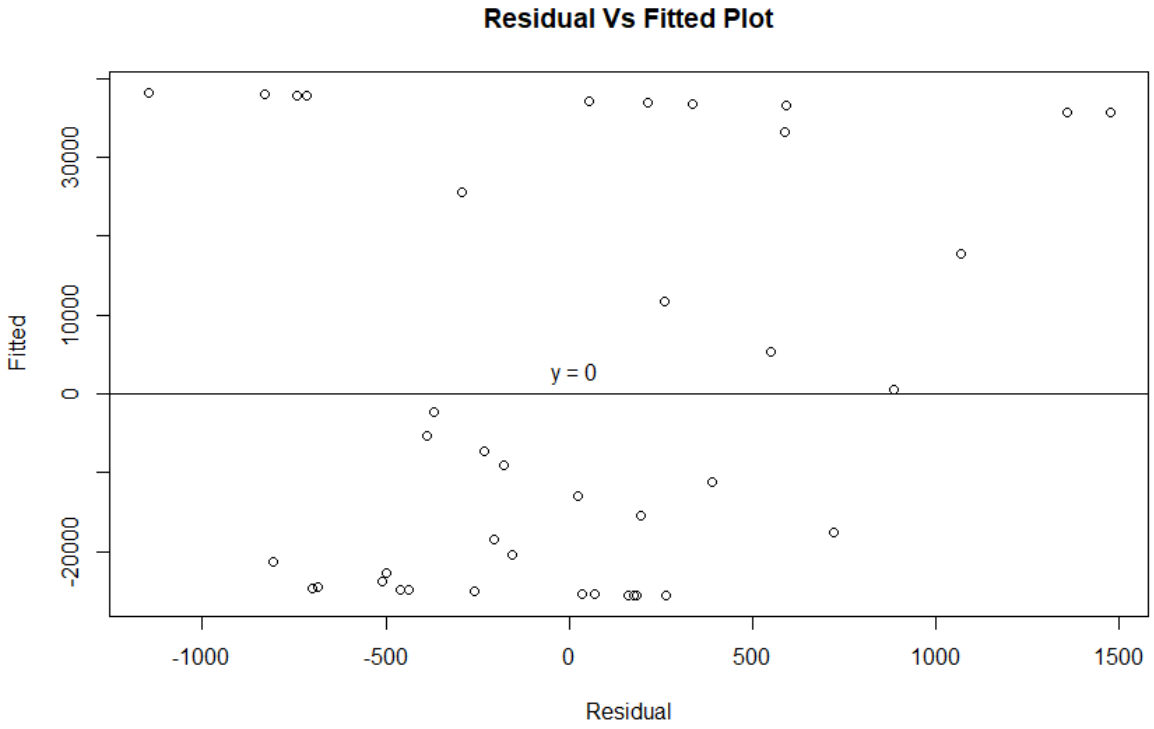
- We observe that after fitting ridge regression model the VIFs have decreased significantly.
- The adjusted R-square is 98.65%.

Now we perform residual analysis on our newly fitted model.

9. Inspection of Properties of Fitted Model After Ridge Regression

9.1. Check for Homoscedasticity Assumption of Errors

The correlation between fitted values and residuals is `0.2057644`.

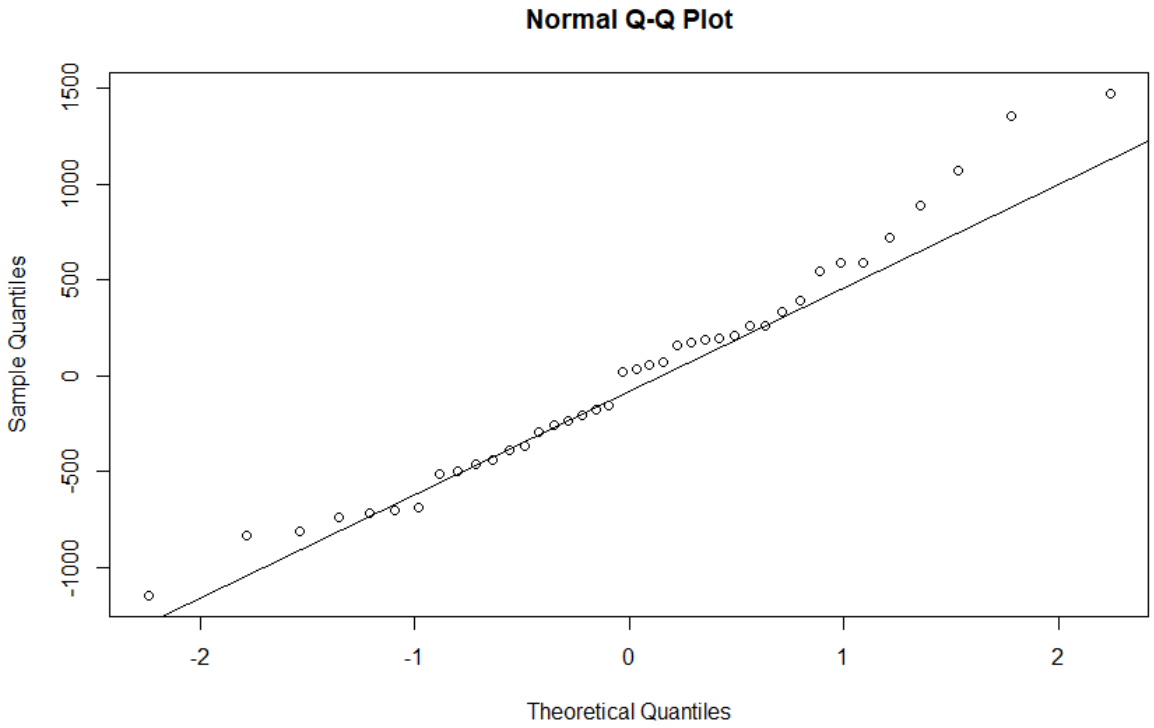


From the plot we cannot find any systematic behavior and the correlation between fitted values and residuals is nearly 0. Hence our assumption of homoscedasticity holds true. For more concrete evidence we perform *Breusch-Pagan Test* for heteroscedasticity.

```
studentized Breusch-Pagan test

data:  model1
BP = 14.646, df = 11, p-value = 0.1993
```

As p-value=0.1993>0.05. Hence we conclude that there is no violation of homoscedasticity assumption in our model.

9.2. Test for Normality Assumption of Errors



As we can see majority of points lies on the straight line. Hence no evidence of violation of normality assumption is found. To strengthen our judgement we further perform *Shapiro-Wilk Test* for normality.

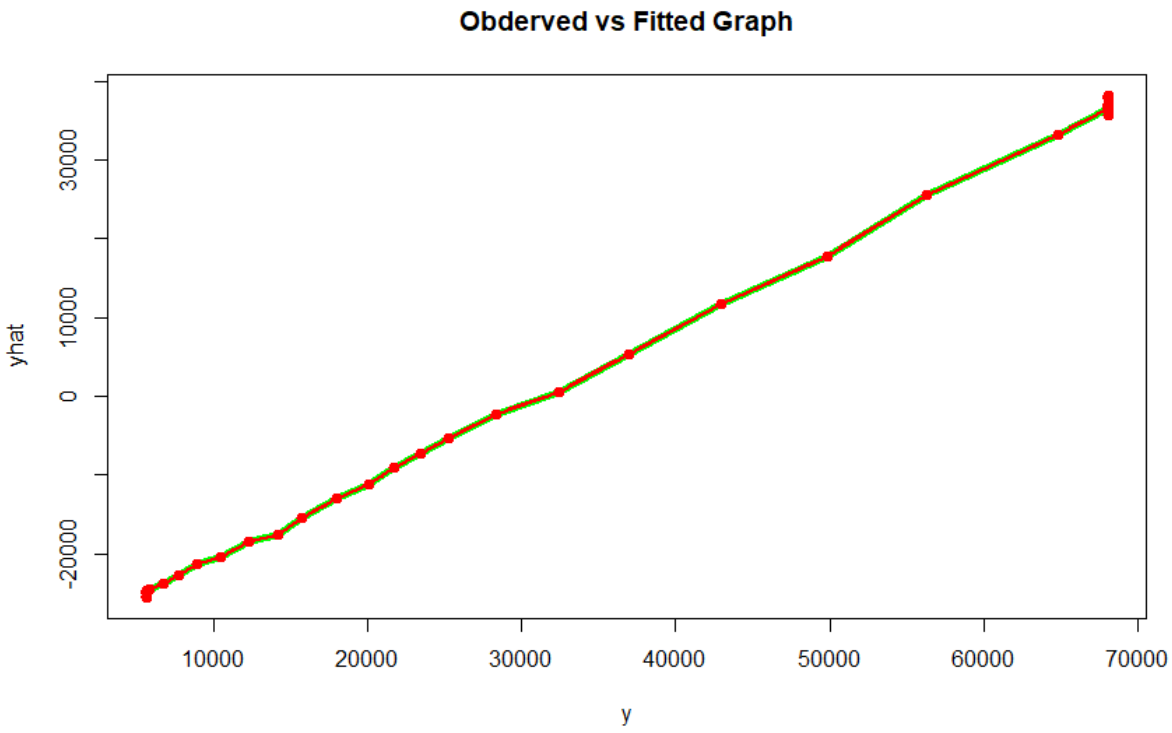
```
Shapiro-Wilk normality test

data:  res
W = 0.97385, p-value = 0.4721
```

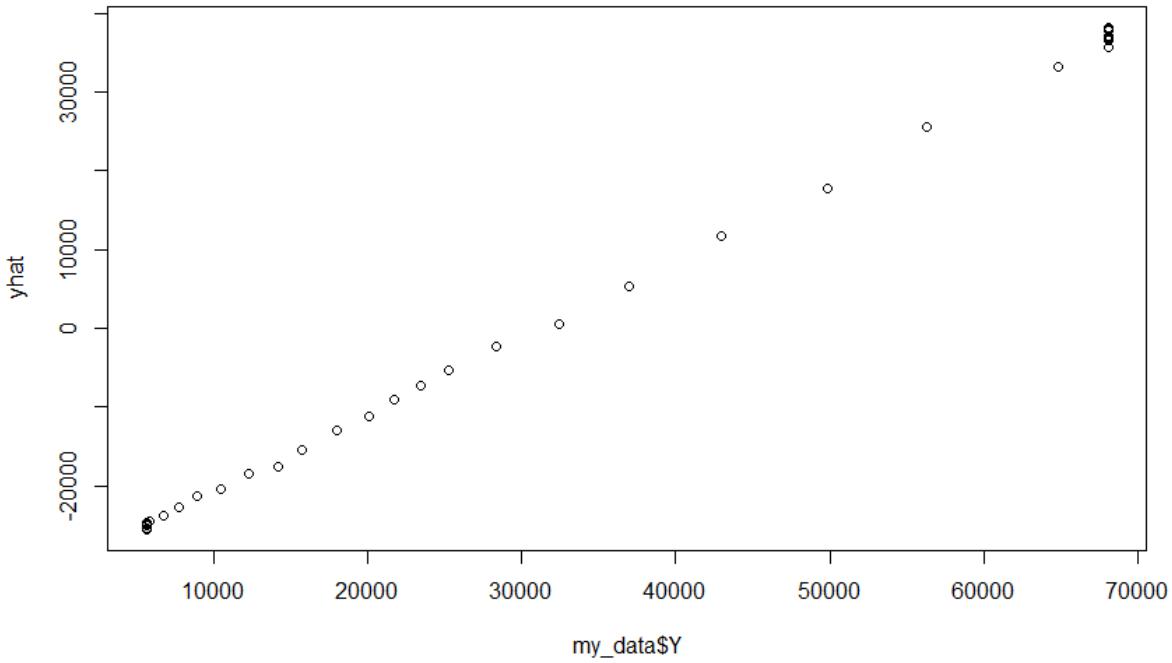
As we can see p-value=0.4721>0.05, hence Normality Assumption of error holds.

Now we compare between *observed and fitted responses*.

The correlation between fitted and observed response is `0.9997423` which indicates a good fit of the observed responses.

9.3.GRAPH BETWEEN OBSERVED AND FITTED RESPONSE





From the above graph, we conclude that our fitted values are approximately equal to observed values of response variable (GDP at Current Prices).

10. Final Fitted Model

Our final model after Ridge regression is given by,

$\widehat{Y}$ = (-3.4707e+03) + (6.0940e-01)*x2 + (1.3252e+01)*x3 + (9.8000e-03)*x4 + (9.9000e-03)*x6 + (7.5000e-03)*x7 + (-2.0000e-04)*x8 + (1.1500e-02)*x9 + (1.2600e-02)*x10 + (5.0000e-03)*x11 + (6.6600e-02)*x14 + (-1.5265e+00)*x15

11. CONCLUSION

R^2 and Adjusted R^2 are used to explain the overall adequacy of the model, where,

$$R^2 = 1 - \frac{SS_{Res}}{SS_{Tot}}$$

$$R^2_{Adj} = 1 - \frac{SS_{Res}/(n-p-1)}{SS_{Tot}/(n-1)}$$

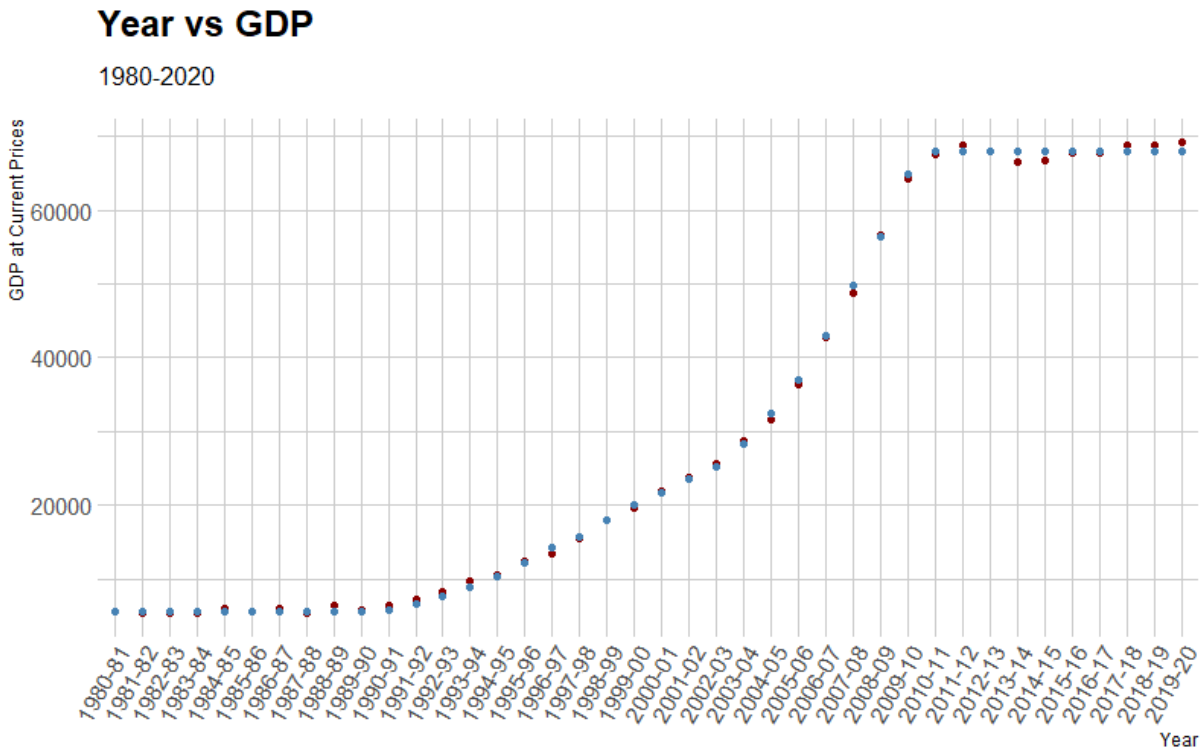
As adjusted R-squared value is 0.9865, we can conclude that 98.65% variability of our response variable (GDP at current prices) can be explained by the regressors we included in the model.

Finally, from our analysis we come to conclude that agricultural production of commercial products , production and import of crude oil and petroleum , export of principal commodities, direct and indirect tax revenues, total savings deposit in commercial banks, gross fiscal deficit, gross market borrowing of government, liabilities of RBI, total developmental and non-developmental expenditures of government, minimum Support price for foodgrains , average price of gold in domestic markets these economical variables have effects on the change of Indian GDP at current prices. By optimizing these variables we can optimize the Indian GDP at current prices. We also see that gross fiscal deficits & average price of gold in domestic markets have negative impacts on GDP.

Now, we visualize our fitted and observed responses for the time period 1980-2020.

12. GRAPHICAL OVERVIEW OF THE MODEL

●Fitted Response Variables ●Observed Response Variables



We can see from the figure that our model is satisfactorily efficient in explaining the change in Indian GDP at current prices.

13. APPENDIX A

Data Source:

We collected the data on GDP (at current prices) & on 15 other economic variables for past 40 years (1980-2020) from handbook of statistics on Indian Economy available at https://www.rbi.org.in.

14. APPENDIX B(R-Code)

```
##import dataset and fit olsreg model
my_data<-read.csv("E:/data/Regression_Clean_Data.csv")
names(my_data)
 <-c("Year","x1","x2","x3","x4","x5","x6","x7","x8","x9","x10","x11","x12","x13","x14","x15","Y")
olsreg<-lm(Y~x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11+x12+x13+x14+x15,my_data)
# Leverage Plot
h_ii=lm.influence(olsreg)$hat
plot(seq(1,40,1),h_ii,pch=,xlab="Observation",ylab="Leverages",
     main="Leverage Plot")
h_ii
h_ii[h_ii>(32/40)]
# Detection of Influential points by Cook's D Method
library(olsrr)
ols_plot_cooksd_chart(olsreg) #
v=cooks.distance(olsreg)
v[v>1]
##detection of multicollinearity
library(car)
vif(olsreg)
library(mctest)
##Approach with variance Decomposition
eigprop(olsreg)
olsreg_1<-lm(Y~x1+x2+x3+x5+x6+x7+x8+x9+x11+x12+x13+x14+x15,my_data)
summary(olsreg_1)
eigprop(olsreg_1)
olsreg_2<-lm(Y~x1+x2+x3+x5+x7+x8+x9+x12+x13+x15,my_data)
summary(olsreg_2)
eigprop(olsreg_2)
olsreg_3<-lm(Y~ x1+x5+x7+x9+x12+x13+x15,my_data)
summary(olsreg_3)
eigprop(olsreg_3)
olsreg_4<-lm(Y~x5+x13+x15,my_data)
```

```
summary(olsreg_4)
eigprop(olsreg_4)
olsreg_5<-lm(Y~x5+x15,my_data)
summary(olsreg_5)
eigprop(olsreg_5)
##stepwise selection
library(olsrr)
Step <- ols_step_both_p(olsreg,pent=0.05,prem=0.05)
Step
library(MASS)
AIC<-stepAIC(olsreg,direction="both")
AIC
summary(AIC)
##creation of new data frame
df <- data.frame(my_data[,3:5],my_data[,7:12],my_data[,15:16])
head(df)
#ridge regression
library(lmridge)
ridge_mod = lmridge(my_data$Y~.,df, K = seq(0, 0.1, 0.001),scaling="sc")
ridge_mod
## Ridge trace
plot(ridge_mod, type = "ridge")
model1=lmridge(my_data$Y~.,df, K=0.035)
summary(model1)
summary(model1)$summaries[[1]]$stats
vif(model1)
plot(ridge_mod, type = "vif")
##Residual Analysis of fitted model after Ridge regression
res<-residuals(model1)
yhat<-fitted(model1)
#covariance between fitted and residual
cor(res,yhat)
#Residual vs Fitted plot
plot(res,yhat,xlab = "Residual",ylab = "Fitted",main = "Residual Vs Fitted Plot")
abline(h=0)
text(17,0,"y = 0",pos=3)
#Breusch-Pagan test for Heteroscedasticity
library(lmtest)
bptest(model1)
#Test for Normality Assumption of residuals
shapiro.test(res)
```

```
#p-value=0.4721>0.05.Hence Normality Assumption of error holds
#Q-Q plot
qqnorm(res)
qqline(res)
##Comparison between Fitted and Observed response
cor(my_data$Y,yhat)
#Fitted vs Observed Graph
plot(my_data$Y,yhat)
plot(my_data$Y,yhat, type="l", col="green", lwd=10, xlab="y", ylab="yhat", main="Obderved vs Fitted Graph")
lines(my_data$Y, yhat, col="red", lwd=2)
lines(my_data$Y, yhat, type="b", col="red", lwd=2, pch=19)
##fit and visualization of final model
yfit=(-3.4707e+03)+(6.0940e-01)*df$x2+(1.3252e+01)*df$x3+(9.8000e-03)*df$x4+(9.9000e-03)*df$x6+(7.5000e-03 )*df$x7+(-2.0000e-04)*df$x8+(1.1500e-02)*df$x9+(1.2600e-02)*df$x10+(5.0000e-03)*df$x11+(6.6600e-02)*df$x14+(-1.5265e+00)*df$x15
yfit
library(ggplot2)
library(dplyr)
library(hrbrthemes)
#observation of fitted vs observed data w.r.t. year
ggplot(df, aes(x=my_data$Year)) +
  geom_point(aes(y = yfit), color = "darkred") +
  geom_point(aes(y =my_data$Y), color="steelblue", linetype="twodash") +
  labs(x = "Year",
       y = "GDP at Current Prices",
       title = "Year vs GDP",
       subtitle = "1980-2020")+
  theme_ipsum()+
  theme(axis.text.x=element_text(angle=60, hjust=1))
```

15. BIBLIOGRAPHY

1.Lecture notes of Dr.Sharmishtha Mitra, Associate Professor, Department of Mathematics & Statistics, IIT Kanpur.

2.Introduction to Linear Regression Analysis -Montgomery, Peck, Vining.

3.Wikipedia.