MULTIPLE REGRESSION

June 3, 2024

Student Performance (Multiple Linear Regression)!

This is to examine the impact of Hours Studied,Previous Scores,Extracurricular Activities,Sleep Hours, and Sample Question Papers Practiced on Performance Index

This is my Jupyter Notebook. You can find my professional profile on My LinkedIn Profile.

1 About Dataset

2 Description:

Hours Studied: The total number of hours spent studying by each student

Previous Scores: The scores obtained by students in previous tests.

Extracurricular Activities: Whether the student participates in extracurricular activities (Yes or No).

Sleep Hours: The average number of hours of sleep the student had per day.

Sample Question Papers Practiced: The number of sample question papers the student practiced.

3 Target Variable:

Performance Index: A measure of the overall performance of each student. The performance index represents the student's academic

performance and has been rounded to the nearest integer. The index ranges from 10 to 100, with higher values indicating better

performance.

Note: flee Free to interprete.

Link: https://www.kaggle.com/datasets/nikhil7280/student-performance-multiple-linear-regression

This is my Jupyter Notebook. You can find my professional profile on My LinkedIn Profile.

[66]:
```python
import numpy as np
import pandas as pd
```

[46]:
```python
df = pd.read_csv("C:\\Users\\obaze.samuel\\Downloads\\Student_Performance.csv")
df.head()
```

[46]:

	Hours Studied	Previous Scores	Extracurricular Activities	Sleep Hours	\
0	7	99	Yes	9	
1	4	82	No	4	
2	8	51	Yes	7	
3	5	52	Yes	5	
4	7	75	No	8	

	Sample Question Papers Practiced	Performance Index
0	1	91.0
1	2	65.0
2	2	45.0
3	2	36.0
4	5	66.0

[47]:
```python
df["Extracurricular Activities"] = df["Extracurricular Activities"].
↪replace({"Yes" : 1, "No" : 0})
df.head()
```

[47]:

	Hours Studied	Previous Scores	Extracurricular Activities	Sleep Hours	\
0	7	99	1	9	
1	4	82	0	4	
2	8	51	1	7	
3	5	52	1	5	
4	7	75	0	8	

	Sample Question Papers Practiced	Performance Index
0	1	91.0
1	2	65.0
2	2	45.0
3	2	36.0
4	5	66.0

[48]:
```python
df.describe().T
```

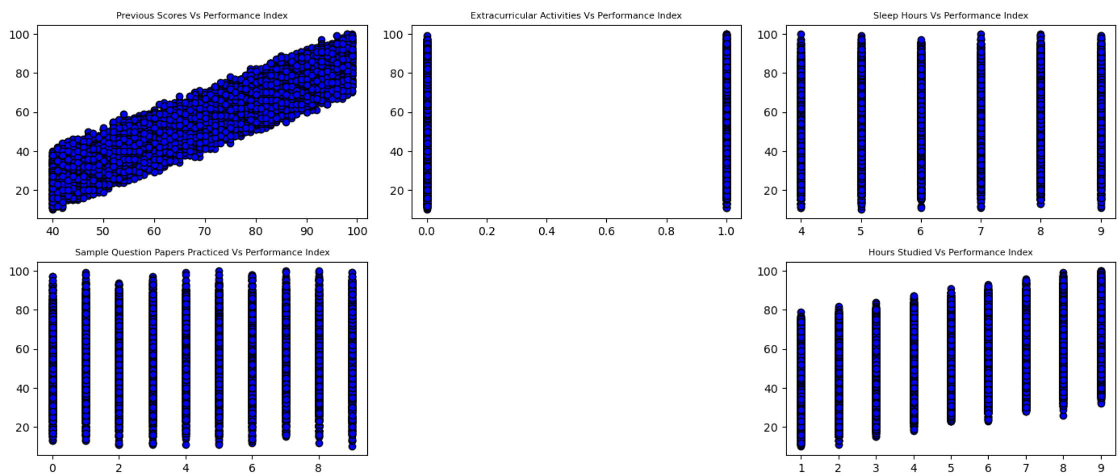
[48]:

	count	mean	std	min	25%	\
Hours Studied	10000.0	4.9929	2.589309	1.0	3.0	
Previous Scores	10000.0	69.4457	17.343152	40.0	54.0	
Extracurricular Activities	10000.0	0.4948	0.499998	0.0	0.0	
Sleep Hours	10000.0	6.5306	1.695863	4.0	5.0	
Sample Question Papers Practiced	10000.0	4.5833	2.867348	0.0	2.0	
Performance Index	10000.0	55.2248	19.212558	10.0	40.0	

	50%	75%	max
Hours Studied	5.0	7.0	9.0
Previous Scores	69.0	85.0	99.0
Extracurricular Activities	0.0	1.0	1.0
Sleep Hours	7.0	8.0	9.0
Sample Question Papers Practiced	5.0	7.0	9.0
Performance Index	55.0	71.0	100.0

4 The Test For Linearity

[49]:
```python
import matplotlib.pyplot as plt

fig,axs = plt.subplots(2,3, figsize = (14,6))
fig.subplots_adjust(hspace = 0.5, wspace = 0.5)
axs = axs.ravel()


for index, column in enumerate(df.columns):
    axs[index-1].set_title("{} Vs Performance Index".format(column), fontsize =
 ↪8)
    axs[index-1].scatter( x = df[column], y = df["Performance Index"], color =
 ↪"blue", edgecolor = "k")

fig.delaxes(axs[4])

fig.tight_layout(pad = 1)
```

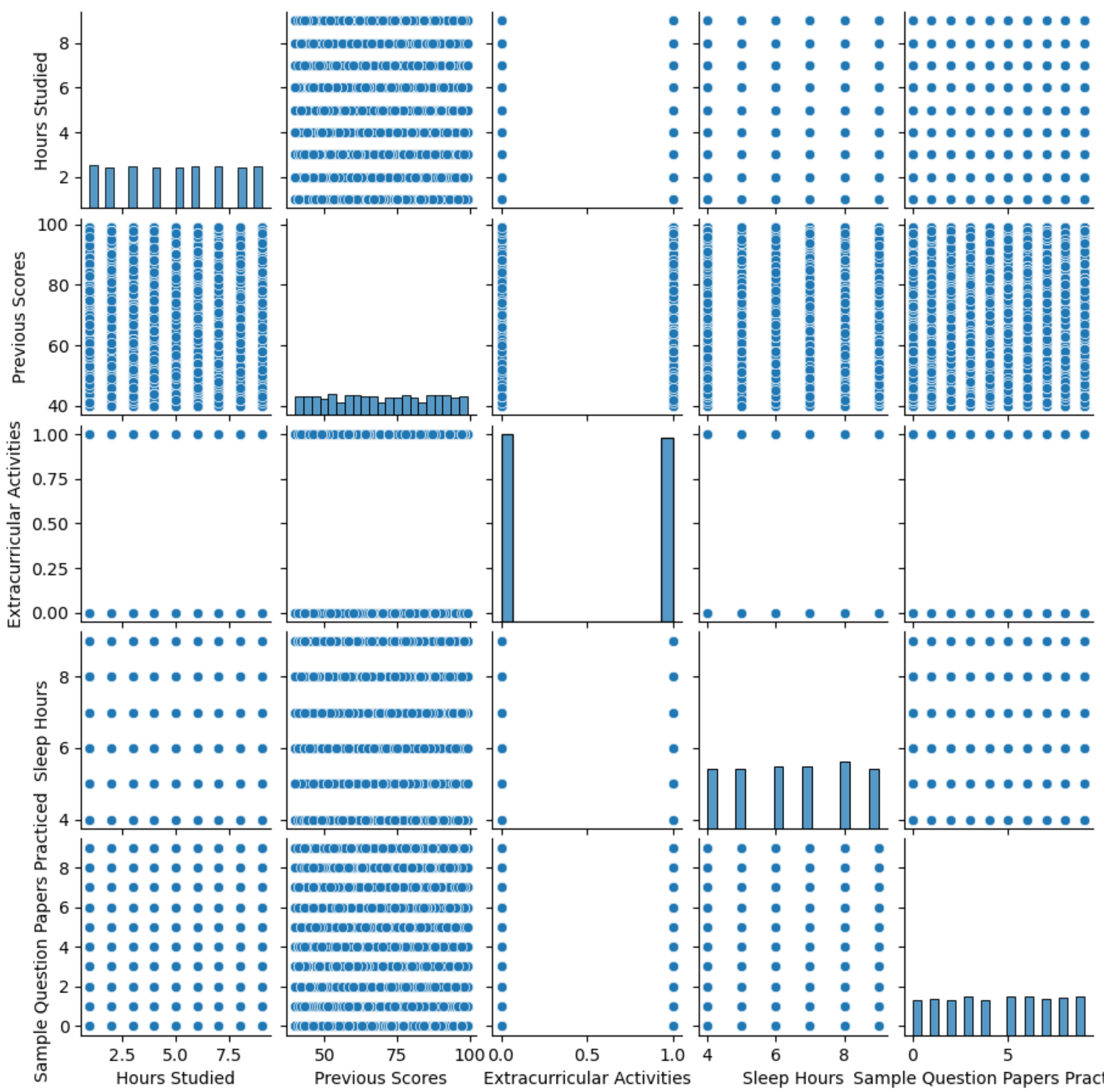


It appears that a significant amount of Hours studied data and Previous Scores has linear relationship with Performance Index

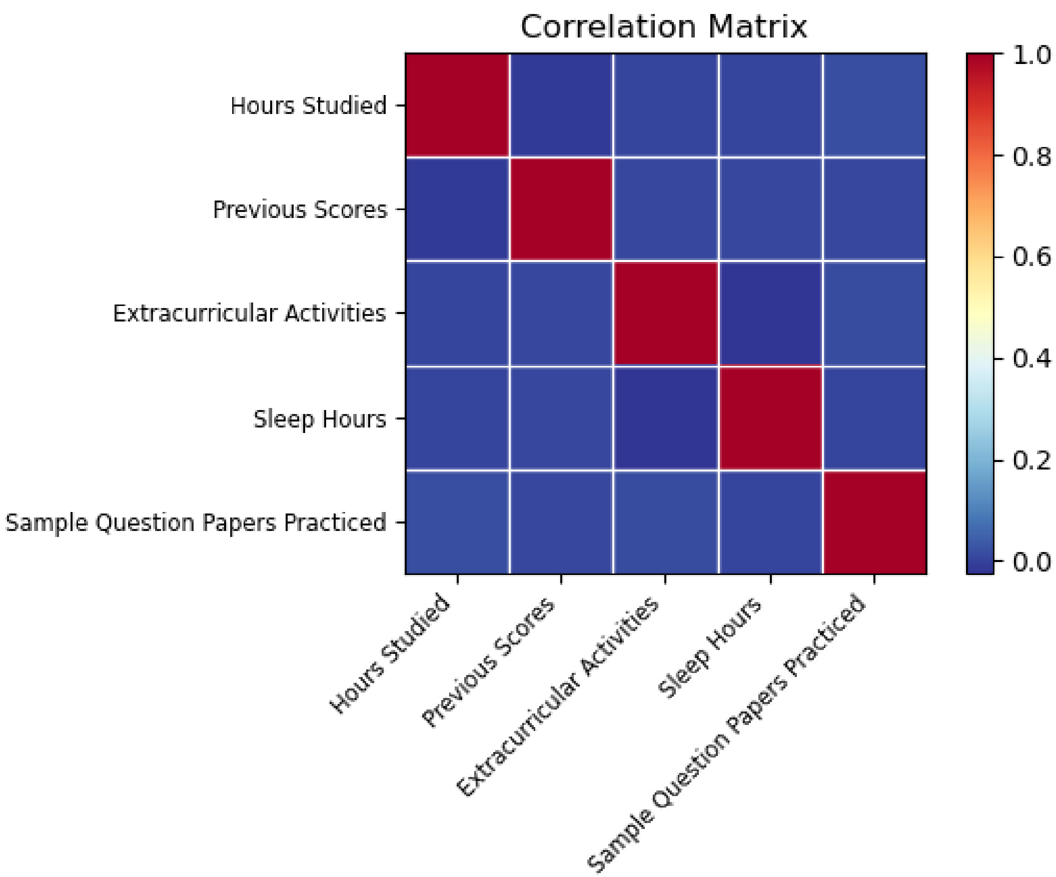
Based on these findings, is there enough linearity present to apply a linear regression model?

5 Testing For Multicollinearity

[50]:
```python
# Pairplot

from seaborn import pairplot

Mp = pairplot(data = df.drop("Performance Index", axis = "columns"))
Mp.fig.set_size_inches(9,9)
```



[51]:
```
# using correlation Heatmap
from statsmodels.graphics.correlation import plot_corr

heatmap = df.drop("Performance Index", axis = 1).corr()

fig = plot_corr(heatmap , xnames = heatmap.columns)
```



[53]:
```
df.rename(columns = {"Performance Index" : "Performance_Index","Extracurricular␣
↪Activities":"Extracurricular_Activities","Hours Studied":
↪"Hours_Studied","Previous Scores":"Previous_Scores", "Sleep Hours":
↪"Sleep_Hours","Sample Question Papers Practiced":
↪"Sample_Question_Papers_Practiced"},inplace = True)
```

[54]:
```
import statsmodels.formula.api as sm

formula1 = df.columns[-1]+ " ~ " + " + ".join(df.columns[:-1])
formula1
```

[54]: 'Performance_Index ~ Hours_Studied + Previous_Scores + Extracurricular_Activities + Sleep_Hours + Sample_Question_Papers_Practiced'

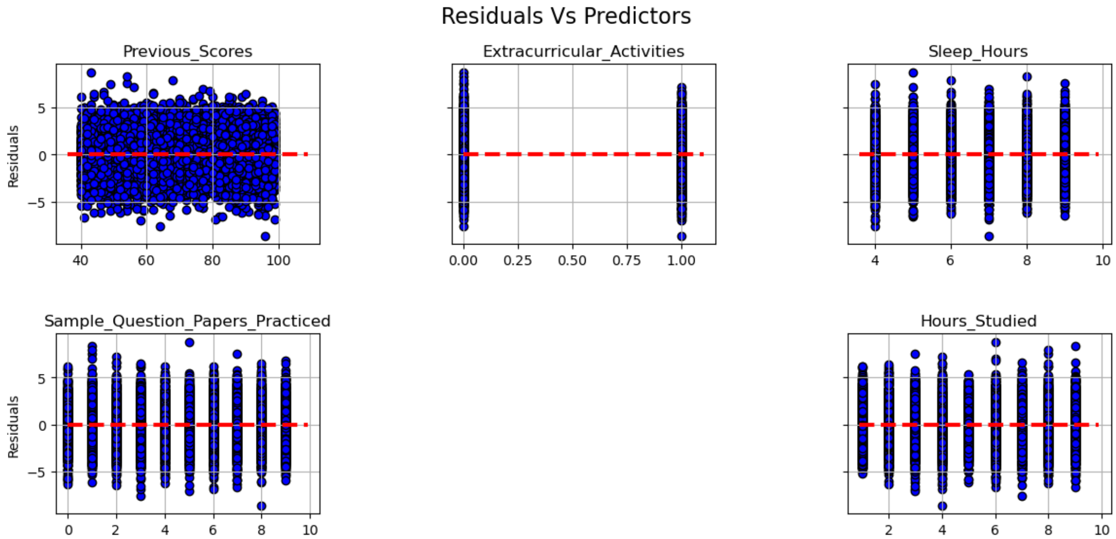
[55]:
```python
model = sm.ols(formula = formula1,data = df)
fitted = model.fit()
print(fitted.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:      Performance_Index   R-squared:                       0.989
Model:                            OLS   Adj. R-squared:                  0.989
Method:                 Least Squares   F-statistic:                 1.757e+05
Date:                Fri, 31 May 2024   Prob (F-statistic):               0.00
Time:                        16:38:53   Log-Likelihood:                -21307.
No. Observations:               10000   AIC:                         4.263e+04
Df Residuals:                    9994   BIC:                         4.267e+04
Df Model:                           5
Covariance Type:            nonrobust
================================================================================
====================
                                       coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------------
--------------------
Intercept                          -34.0756      0.127   -268.010      0.000
-34.325     -33.826
Hours_Studied                        2.8530      0.008    362.353      0.000
2.838       2.868
Previous_Scores                      1.0184      0.001    866.450      0.000
1.016       1.021
Extracurricular_Activities           0.6129      0.041     15.029      0.000
0.533       0.693
Sleep_Hours                          0.4806      0.012     39.972      0.000
0.457       0.504
Sample_Question_Papers_Practiced     0.1938      0.007     27.257      0.000
0.180       0.208
==============================================================================
Omnibus:                        3.851   Durbin-Watson:                   2.001
Prob(Omnibus):                  0.146   Jarque-Bera (JB):                4.036
Skew:                           0.013   Prob(JB):                        0.133
Kurtosis:                       3.095   Cond. No.                         452.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

6 TESTING FOR INDEPENDENCE

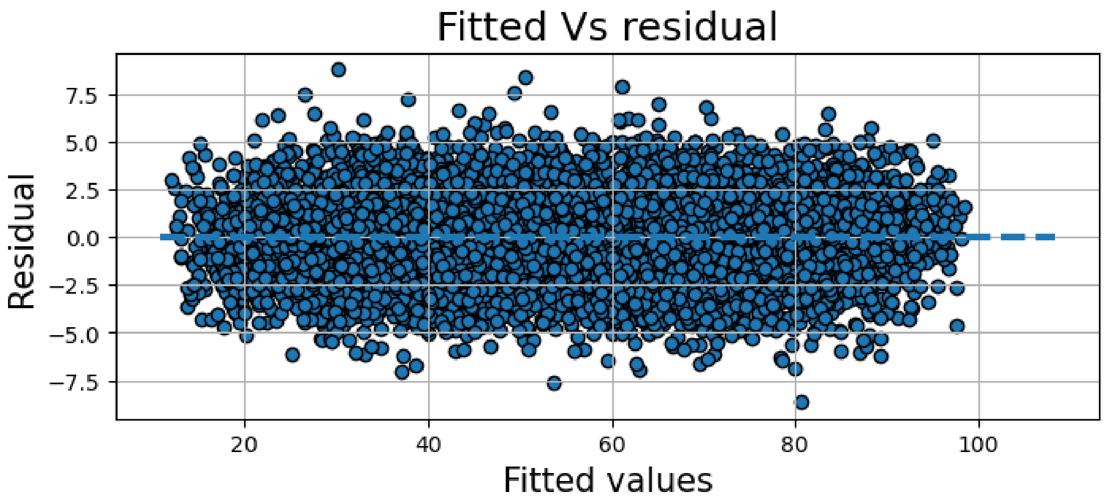
This is my Jupyter Notebook. You can find my professional profile on My LinkedIn Profile.

[56]:
```
fig, axs = plt.subplots(2,3, figsize = (14,6), sharey = True)
fig.subplots_adjust(hspace = 0.5, wspace = .5)
fig.suptitle("Residuals Vs Predictors", fontsize = 16)
axs = axs.ravel()

for index,column in enumerate(df.columns):
    axs[index-1].set_title("{}".format(column), fontsize = 12)
    axs[index-1].scatter(x =df[column], y = fitted.resid, color = "blue",␣
 ↪edgecolor = "k")
    axs[index-1].grid(True)

    xmin = min(df[column])
    xmax = max(df[column])
    axs[index-1].hlines(y = 0, xmax = xmax*1.1, xmin = xmin *.9, color = "red",␣
 ↪linestyle = "--", lw = 3)
    if index == 1 or index ==4:
        axs[index-1].set_ylabel("Residuals")
fig.delaxes(axs[4])
```



[57]:
```
plt.figure(figsize = (8,3))
p = plt.scatter(x = fitted.fittedvalues,y= fitted.resid, edgecolor = "k")
xmin = min(fitted.fittedvalues)
xmax = max(fitted.fittedvalues)
plt.hlines(y=0, xmin = xmin*.9, xmax = xmax * 1.1, linestyle = "--", lw = 3)
```

```
plt.xlabel("Fitted values", fontsize = 15)
plt.ylabel("Residual", fontsize = 15)
plt.title("Fitted Vs residual", fontsize = 18)
plt.grid(True)
plt.show()
```



[58]:

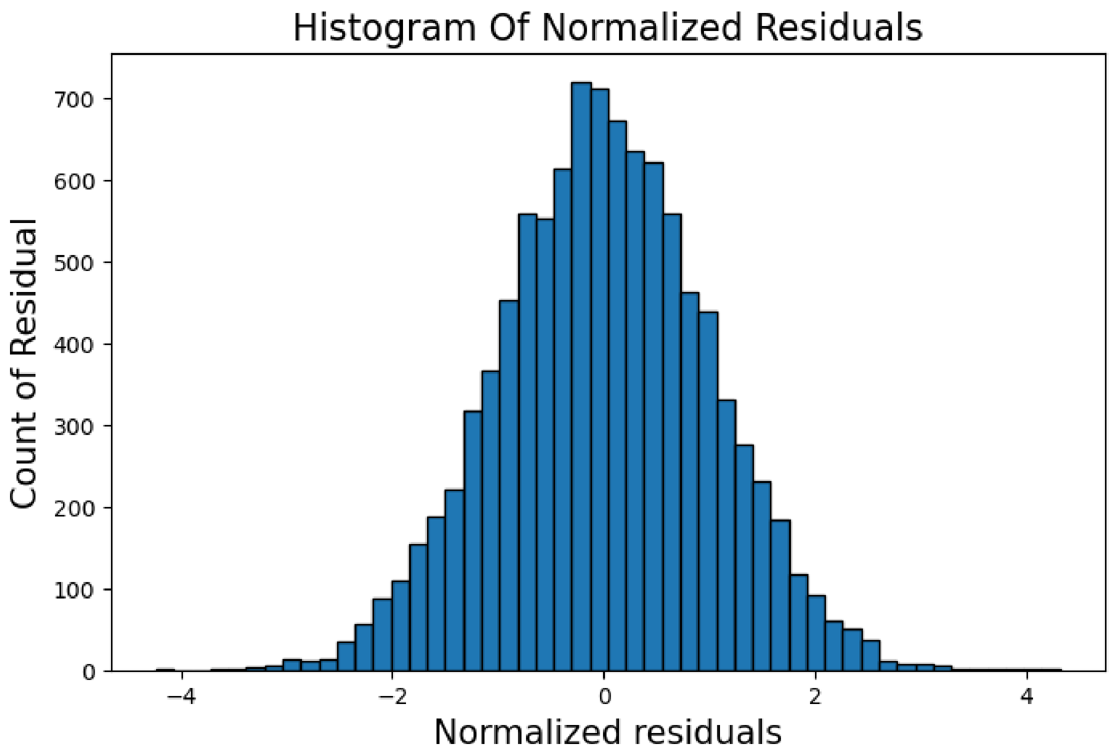
```
import statsmodels.stats.api as sms

model = sm.ols(formula = formula1,data = df)

fitted = model.fit()

residual = fitted.resid

bp_test_result = sms.het_breuschpagan(residual, fitted.model.exog)

print("Breusch Pagan Test")

print("LM statistics",bp_test_result[0])
print("LM Test p-value",bp_test_result[1])
print("F-Statistics", bp_test_result[2])
print("F-Test", bp_test_result[3])
```

```
Breusch Pagan Test
LM statistics 2.1594259364510204
LM Test p-value 0.8266745765789328
F-Statistics 0.4317192827595611
F-Test 0.8267804553852909
```

7 TESTING FOR NORMALITY

HISTOGRAM OF NORMALIZED RESIDUALS

[59]:
```
plt.figure(figsize = (8,5))
plt.hist(fitted.resid_pearson, bins = 50 , edgecolor ="k")
plt.ylabel("Count of Residual", fontsize = 15)
plt.xlabel("Normalized residuals", fontsize = 15)
plt.title("Histogram Of Normalized Residuals", fontsize = 16)
plt.show()
```

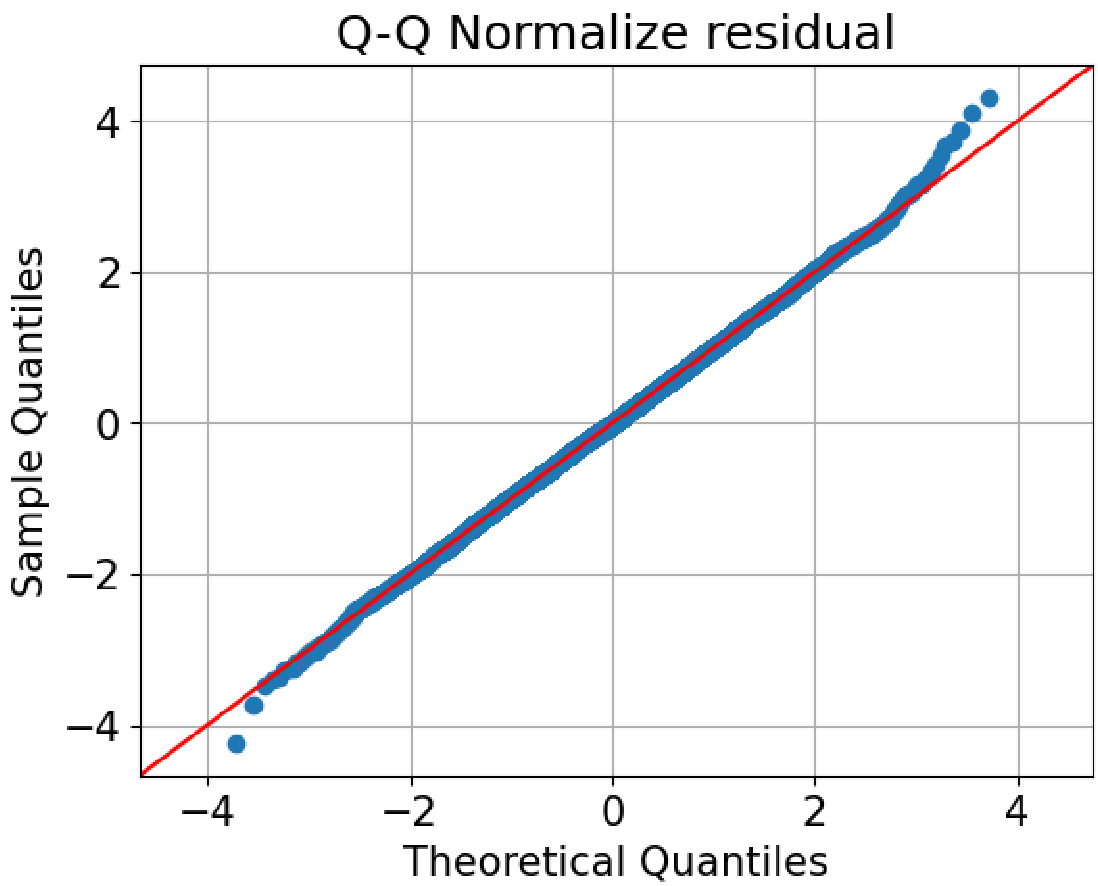


Q-Q PLOT OF THE RESIDUALS

[60]:
```
from statsmodels.graphics.gofplots import qqplot
```

[61]:
```
plt.figure(figsize = (8,5))
fig = qqplot(fitted.resid_pearson, line = "45", fit = "True")
plt.xticks(fontsize = 15)
plt.yticks(fontsize = 15)
plt.xlabel("Theoretical Quantiles", fontsize = 15)
plt.ylabel("Sample Quantiles", fontsize= 15 )
plt.title("Q-Q Normalize residual", fontsize = 18 )
plt.grid(True)
plt.show()
```

```
<Figure size 800x500 with 0 Axes>
```

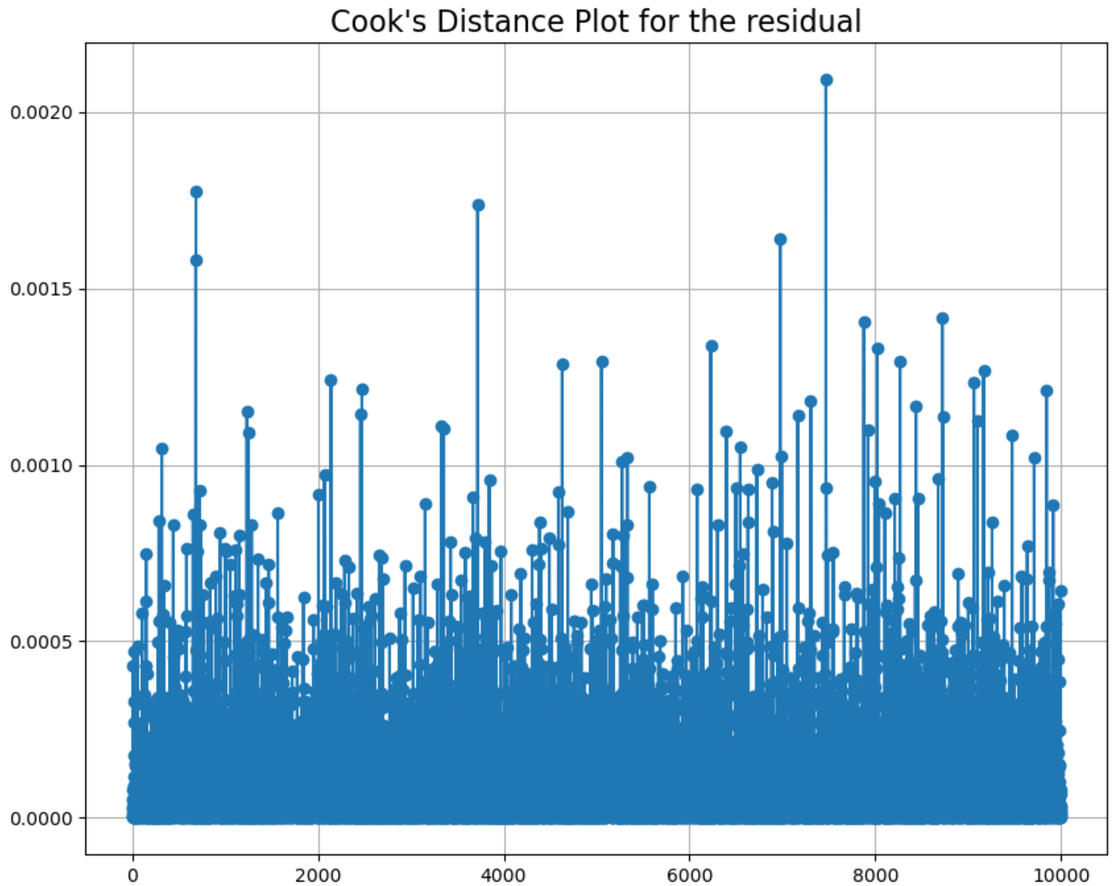


8 CHECKING FOR OUTLIERS IN RESIDUALS

[62]:
```
from statsmodels.stats.outliers_influence import OLSInfluence as influence
```

[63]:
```
inf=influence(fitted)
```

[64]:
```
(c, p) = inf.cooks_distance
plt.figure(figsize = (10,8))
plt.title("Cook's Distance Plot for the residual",fontsize = 16)
plt.plot(np.arange(len(c)), c , marker= "o", linestyle="-")
plt.grid(True)
plt.show()
```



This is my Jupyter Notebook. You can find my professional profile on My LinkedIn Profile.

```
[ ]:
```