In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm
from statsmodels.graphics.regressionplots import influence_plot
import statsmodels.formula.api as smf
import numpy as np
```

In [2]:
```
toyota=pd.read_csv('ToyotaCorolla.csv', encoding= 'unicode_escape')
toyota
```

Out[2]:

	Id	Model	Price	Age_08_04	Mfg_Month	Mfg_Year	KM	Fuel_Type	HP	Met_Color	...	Central_
0	1	TOYOTA Corolla 2.0 D4D HATCHB TERRA 2/3-Doors	13500	23	10	2002	46986	Diesel	90	1	...	
1	2	TOYOTA Corolla 2.0 D4D HATCHB TERRA 2/3-Doors	13750	23	10	2002	72937	Diesel	90	1	...	
2	3	TOYOTA Corolla 2.0 D4D HATCHB TERRA 2/3-Doors	13950	24	9	2002	41711	Diesel	90	1	...	
3	4	TOYOTA Corolla 2.0 D4D HATCHB TERRA 2/3-Doors	14950	26	7	2002	48000	Diesel	90	0	...	
4	5	TOYOTA Corolla 2.0 D4D HATCHB SOL 2/3-Doors	13750	30	3	2002	38500	Diesel	90	0	...	
...	...	...	...	...	...	...	...	...	...	...	...	
1431	1438	TOYOTA Corolla 1.3 16V HATCHB G6 2/3-Doors	7500	69	12	1998	20544	Petrol	86	1	...	
1432	1439	TOYOTA Corolla 1.3 16V HATCHB LINEA TERRA 2/3-...	10845	72	9	1998	19000	Petrol	86	0	...	
1433	1440	TOYOTA Corolla 1.3 16V HATCHB LINEA TERRA 2/3-...	8500	71	10	1998	17016	Petrol	86	0	...	
1434	1441	TOYOTA Corolla 1.3 16V HATCHB LINEA TERRA 2/3-...	7250	70	11	1998	16916	Petrol	86	1	...	
1435	1442	TOYOTA orolla	6950	76	5	1998	1	Petrol	110	0	...	

	Id	Model	Price	Age_08_04	Mfg_Month	Mfg_Year	KM	Fuel_Type	HP	Met_Color	...	Central_
		1.6 LB LINEA TERRA 4/5-Doors										

1436 rows × 38 columns

In [3]:
```
toyota.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1436 entries, 0 to 1435
Data columns (total 38 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Id                1436 non-null   int64
 1   Model             1436 non-null   object
 2   Price             1436 non-null   int64
 3   Age_08_04         1436 non-null   int64
 4   Mfg_Month         1436 non-null   int64
 5   Mfg_Year          1436 non-null   int64
 6   KM                1436 non-null   int64
 7   Fuel_Type         1436 non-null   object
 8   HP                1436 non-null   int64
 9   Met_Color         1436 non-null   int64
 10  Color             1436 non-null   object
 11  Automatic         1436 non-null   int64
 12  cc                1436 non-null   int64
 13  Doors             1436 non-null   int64
 14  Cylinders         1436 non-null   int64
 15  Gears             1436 non-null   int64
 16  Quarterly_Tax     1436 non-null   int64
 17  Weight            1436 non-null   int64
 18  Mfr_Guarantee     1436 non-null   int64
 19  BOVAG_Guarantee   1436 non-null   int64
 20  Guarantee_Period  1436 non-null   int64
 21  ABS               1436 non-null   int64
 22  Airbag_1          1436 non-null   int64
 23  Airbag_2          1436 non-null   int64
 24  Airco             1436 non-null   int64
 25  Automatic_airco   1436 non-null   int64
 26  Boardcomputer     1436 non-null   int64
 27  CD_Player         1436 non-null   int64
 28  Central_Lock      1436 non-null   int64
 29  Powered_Windows   1436 non-null   int64
 30  Power_Steering    1436 non-null   int64
 31  Radio             1436 non-null   int64
 32  Mistlamps         1436 non-null   int64
 33  Sport_Model       1436 non-null   int64
 34  Backseat_Divider  1436 non-null   int64
 35  Metallic_Rim      1436 non-null   int64
 36  Radio_cassette    1436 non-null   int64
 37  Tow_Bar           1436 non-null   int64
dtypes: int64(35), object(3)
memory usage: 426.4+ KB
```

In [4]:
```
toyota.isna().sum()
```

Out[4]:
```
Id                  0
Model               0
Price               0
Age_08_04           0
Mfg_Month           0
Mfg_Year            0
KM                  0
Fuel_Type           0
HP                  0
Met_Color           0
Color               0
Automatic           0
cc                  0
Doors               0
Cylinders           0
Gears               0
Quarterly_Tax       0
Weight              0
Mfr_Guarantee       0
BOVAG_Guarantee     0
Guarantee_Period    0
ABS                 0
Airbag_1            0
Airbag_2            0
Airco               0
Automatic_airco     0
Boardcomputer       0
CD_Player           0
Central_Lock        0
Powered_Windows     0
Power_Steering      0
Radio               0
Mistlamps           0
Sport_Model         0
Backseat_Divider    0
Metallic_Rim        0
Radio_cassette      0
Tow_Bar             0
dtype: int64
```

In [5]:
```
toyota.corr()
```

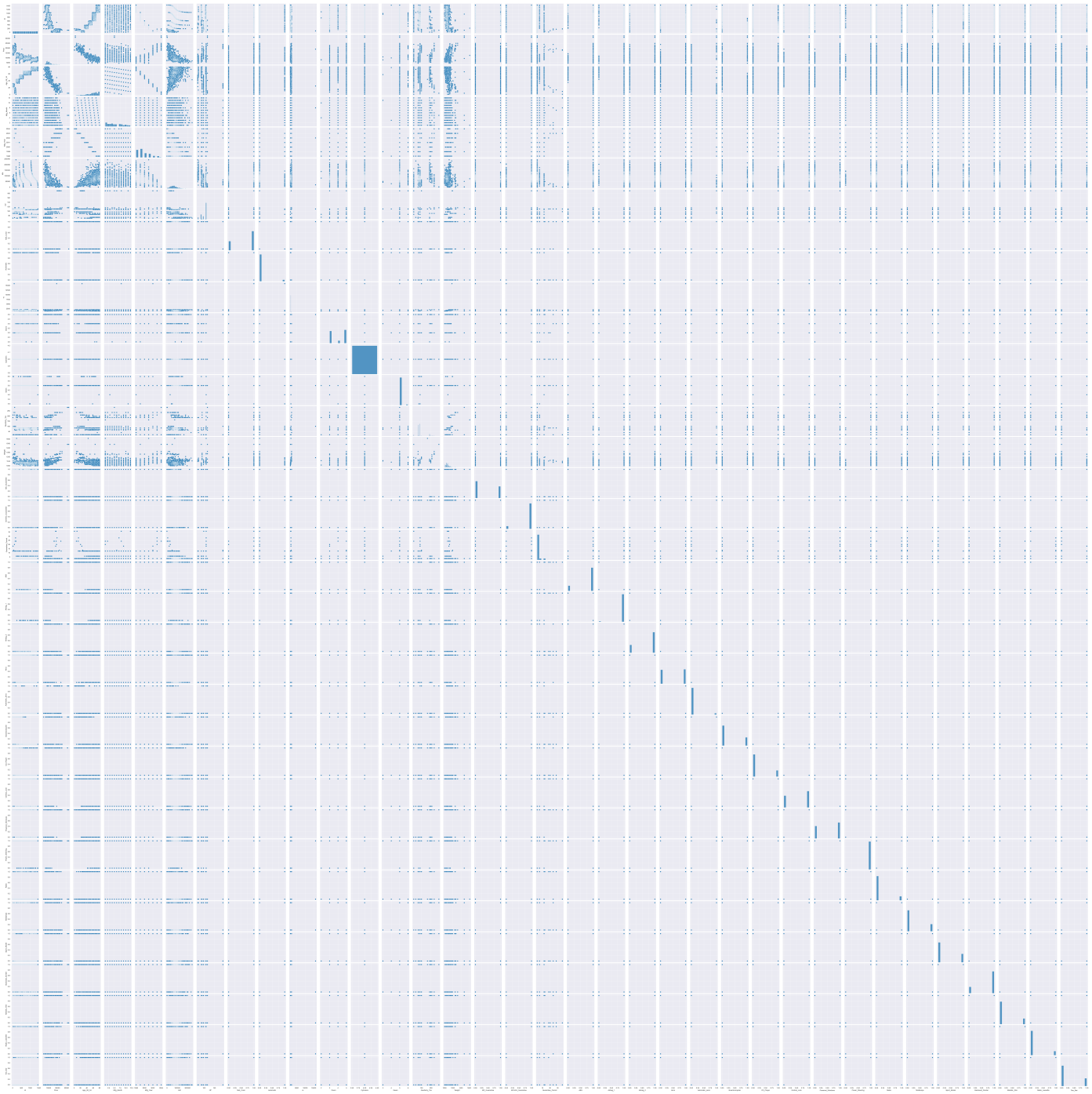
Out[5]:

	Id	Price	Age_08_04	Mfg_Month	Mfg_Year	KM	HP	Met_Color	Au
Id	1.000000	-0.738250	0.906132	0.043742	-0.919523	0.273298	-0.109375	-0.079713	0
Price	-0.738250	1.000000	-0.876590	-0.018138	0.885159	-0.569960	0.314990	0.108905	0
Age_08_04	0.906132	-0.876590	1.000000	-0.123255	-0.983661	0.505672	-0.156622	-0.108150	0
Mfg_Month	0.043742	-0.018138	-0.123255	1.000000	-0.057416	-0.020630	-0.039312	0.030266	0
Mfg_Year	-0.919523	0.885159	-0.983661	-0.057416	1.000000	-0.504974	0.164697	0.103310	-0
KM	0.273298	-0.569960	0.505672	-0.020630	-0.504974	1.000000	-0.333538	-0.080503	-0
HP	-0.109375	0.314990	-0.156622	-0.039312	0.164697	-0.333538	1.000000	0.058712	0
Met_Color	-0.079713	0.108905	-0.108150	0.030266	0.103310	-0.080503	0.058712	1.000000	-0
Automatic	0.066265	0.033081	0.031717	0.009146	-0.033567	-0.081854	0.013144	-0.019335	1
cc	-0.117704	0.126389	-0.098084	0.037387	0.091892	0.102683	0.035856	0.031812	0
Doors	-0.130207	0.185326	-0.148359	-0.012069	0.151442	-0.036197	0.092424	0.085243	-0
Cylinders	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
Gears	-0.043343	0.063104	-0.005364	-0.013063	0.007766	0.015023	0.209477	0.018601	-0
Quarterly_Tax	-0.240821	0.219197	-0.198431	0.031373	0.193934	0.278165	-0.298432	0.011326	-0
Weight	-0.414500	0.581198	-0.470253	-0.002167	0.473478	-0.028598	0.089614	0.057929	0
Mfr_Guarantee	-0.162006	0.197802	-0.164658	-0.005771	0.166697	-0.212851	0.140026	0.154850	0
BOVAG_Guarantee	-0.015065	0.028133	0.006865	-0.003863	-0.006206	0.001438	0.022701	0.010783	0
Guarantee_Period	-0.086256	0.146627	-0.152563	0.029010	0.148218	-0.138942	0.076163	0.009295	-0
ABS	-0.461437	0.306138	-0.412887	0.072532	0.402215	-0.177203	0.057832	0.022298	-0
Airbag_1	-0.123465	0.093588	-0.105406	0.003756	0.105359	-0.018012	0.025137	0.100055	-0
Airbag_2	-0.358316	0.248974	-0.329017	0.076749	0.317075	-0.139275	0.017644	0.038416	0
Airco	-0.386207	0.429259	-0.403600	0.057088	0.395674	-0.133057	0.241134	0.114190	-0
Automatic_airco	-0.327468	0.588262	-0.426259	-0.049017	0.437718	-0.258221	0.244957	0.027977	0
Boardcomputer	-0.695207	0.601292	-0.719449	0.017715	0.720567	-0.353862	0.129715	0.089886	-0
CD_Player	-0.464520	0.481374	-0.510895	-0.016736	0.517008	-0.266826	0.102300	0.198220	-0
Central_Lock	-0.238940	0.343458	-0.279631	0.010055	0.279490	-0.125177	0.250122	0.153307	-0
Powered_Windows	-0.236723	0.356518	-0.283856	0.025185	0.280996	-0.156242	0.265593	0.145147	-0
Power_Steering	-0.091587	0.064275	-0.069192	-0.055495	0.079676	0.007397	0.048850	0.086544	-0
Radio	-0.010971	-0.041887	0.013791	0.031601	-0.019607	0.013661	0.020998	0.072756	-0
Mistlamps	-0.139708	0.222083	-0.126895	-0.033504	0.133737	-0.074327	0.210571	0.023821	0
Sport_Model	-0.028704	0.164121	-0.110988	0.052789	0.102080	-0.044784	-0.006027	0.003779	0
Backseat_Divider	-0.136398	0.102569	-0.116751	0.023245	0.113237	-0.045658	0.010908	0.037741	-0
Metallic_Rim	-0.022232	0.108564	-0.040045	0.023506	0.036022	-0.013599	0.206784	0.053829	-0
Radio_cassette	-0.011611	-0.043179	0.012857	0.032576	-0.018844	0.015770	0.019919	0.071530	-0
Tow_Bar	0.159171	-0.172369	0.188720	-0.042170	-0.182206	0.084153	0.068271	0.148536	0

35 rows × 35 columns

In [6]:
```
sns.set_style(style='darkgrid')
sns.pairplot(toyota)
```

Out[6]:
```
<seaborn.axisgrid.PairGrid at 0x1953e8389a0>
```



In [7]:
```
model = smf.ols('Price~Age_08_04+KM+HP+cc+Doors+Gears+Quarterly_Tax+Weight',data=toyota)
```

In [8]:
```
model.params
```

Out[8]:
```
Intercept       -5573.106358
Age_08_04        -121.658402
KM                 -0.020817
HP                 31.680906
cc                 -0.121100
Doors              -1.616641
Gears             594.319936
Quarterly_Tax       3.949081
Weight             16.958632
dtype: float64
```

In [9]:
```
print(model.tvalues, '\n', model.pvalues)
```

```
Intercept        -3.948666
Age_08_04       -46.511852
KM              -16.621622
HP               11.241018
cc               -1.344222
Doors            -0.040410
Gears             3.016007
Quarterly_Tax     3.014535
Weight           15.879803
dtype: float64
 Intercept         8.241949e-05
Age_08_04        3.354724e-288
KM                7.538439e-57
HP                3.757218e-28
cc                1.790902e-01
Doors             9.677716e-01
Gears             2.606549e-03
Quarterly_Tax     2.619148e-03
Weight            2.048576e-52
dtype: float64
```

In [10]:
```
(model.rsquared,model.rsquared_adj)
```

Out[10]:
```
(0.8637627463428192, 0.8629989775766963)
```

In [11]:
```
model.summary()
```

Out[11]:

OLS Regression Results

Dep. Variable:	Price	**R-squared:**	0.864
Model:	OLS	**Adj. R-squared:**	0.863
Method:	Least Squares	**F-statistic:**	1131.
Date:	Sun, 28 Jan 2024	**Prob (F-statistic):**	0.00
Time:	21:27:48	**Log-Likelihood:**	-12376.
No. Observations:	1436	**AIC:**	2.477e+04
Df Residuals:	1427	**BIC:**	2.482e+04
Df Model:	8		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	-5573.1064	1411.390	-3.949	0.000	-8341.728	-2804.485
Age_08_04	-121.6584	2.616	-46.512	0.000	-126.789	-116.527
KM	-0.0208	0.001	-16.622	0.000	-0.023	-0.018
HP	31.6809	2.818	11.241	0.000	26.152	37.209
cc	-0.1211	0.090	-1.344	0.179	-0.298	0.056
Doors	-1.6166	40.006	-0.040	0.968	-80.093	76.859
Gears	594.3199	197.055	3.016	0.003	207.771	980.869
Quarterly_Tax	3.9491	1.310	3.015	0.003	1.379	6.519
Weight	16.9586	1.068	15.880	0.000	14.864	19.054

Omnibus:	151.719	**Durbin-Watson:**	1.543
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	1011.853
Skew:	-0.219	**Prob(JB):**	1.90e-220
Kurtosis:	7.089	**Cond. No.**	3.13e+06

Notes:

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

[2] The condition number is large, 3.13e+06. This might indicate that there are strong multicollinearity or other numerical problems.

In [12]:
```
ml_cc=smf.ols('Price~cc',data = toyota).fit()
print(ml_cc.tvalues, '\n', ml_cc.pvalues)
```

```
Intercept    24.694090
cc            4.824822
dtype: float64
 Intercept    1.766912e-112
cc            1.550808e-06
dtype: float64
```

In [13]:
```
ml_cc.summary()
```

Out[13]:

OLS Regression Results

Dep. Variable:	Price	**R-squared:**	0.016
Model:	OLS	**Adj. R-squared:**	0.015
Method:	Least Squares	**F-statistic:**	23.28
Date:	Sun, 28 Jan 2024	**Prob (F-statistic):**	1.55e-06
Time:	21:27:49	**Log-Likelihood:**	-13795.
No. Observations:	1436	**AIC:**	2.759e+04
Df Residuals:	1434	**BIC:**	2.760e+04
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	9027.5548	365.576	24.694	0.000	8310.435	9744.675
cc	1.0802	0.224	4.825	0.000	0.641	1.519

Omnibus:	465.181	**Durbin-Watson:**	0.267
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	1390.401
Skew:	1.649	**Prob(JB):**	1.20e-302
Kurtosis:	6.516	**Cond. No.**	6.29e+03

Notes:

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

[2] The condition number is large, 6.29e+03. This might indicate that there are strong multicollinearity or other numerical problems.

In [14]:
```
ml_d=smf.ols('Price~Doors',data = toyota).fit()
print(ml_d.tvalues, '\n', ml_d.pvalues)
```

```
Intercept    19.258097
Doors         7.141657
dtype: float64
 Intercept    1.094732e-73
Doors        1.461237e-12
dtype: float64
```

In [15]:
```
ml_d.summary()
```

Out[15]:

OLS Regression Results

Dep. Variable:	Price	**R-squared:**	0.034
Model:	OLS	**Adj. R-squared:**	0.034
Method:	Least Squares	**F-statistic:**	51.00
Date:	Sun, 28 Jan 2024	**Prob (F-statistic):**	1.46e-12
Time:	21:27:49	**Log-Likelihood:**	-13782.
No. Observations:	1436	**AIC:**	2.757e+04
Df Residuals:	1434	**BIC:**	2.758e+04
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	7885.0058	409.438	19.258	0.000	7081.843	8688.168
Doors	705.5586	98.795	7.142	0.000	511.761	899.356

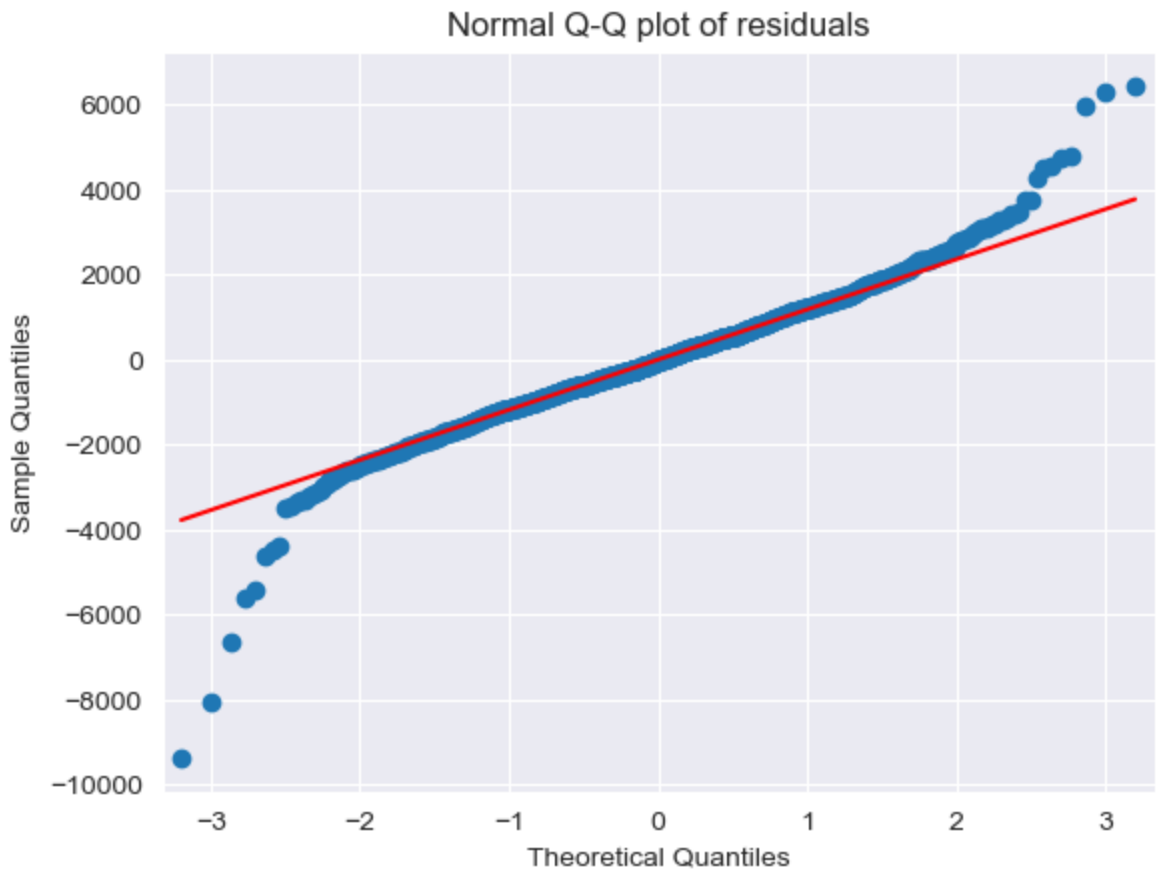
Omnibus:	466.779	**Durbin-Watson:**	0.287
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	1406.209
Skew:	1.651	**Prob(JB):**	4.42e-306
Kurtosis:	6.549	**Cond. No.**	19.0

Notes:

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

In [16]:

```
qqplot=sm.qqplot(model.resid,line='q')
plt.title("Normal Q-Q plot of residuals")
plt.show()
```

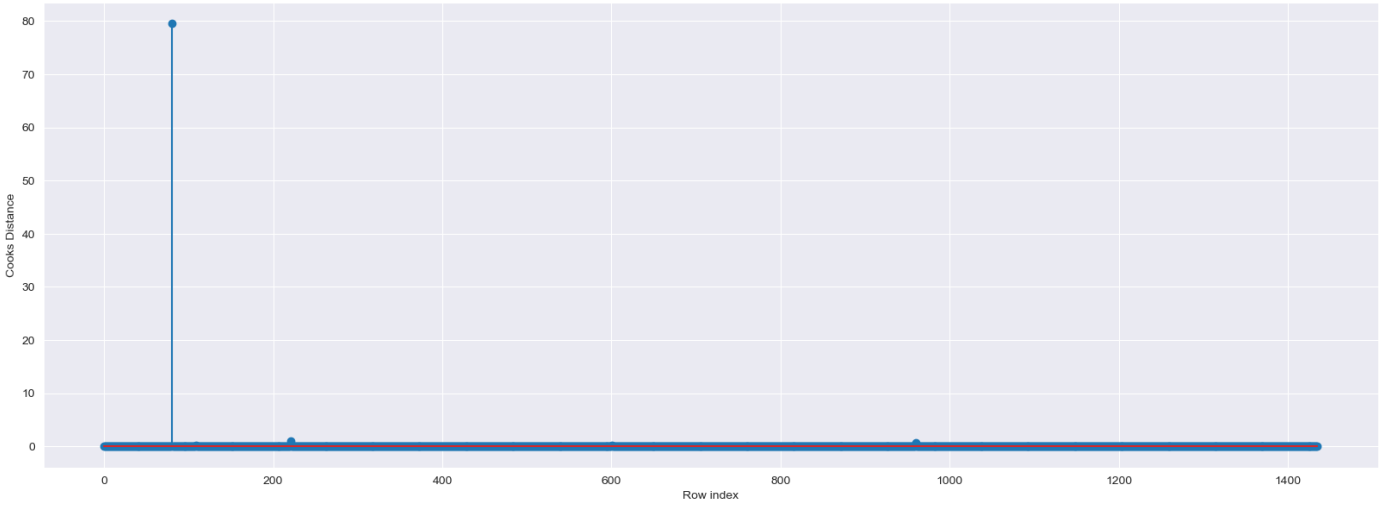


In [18]:
```
list(np.where(model.resid>2100))
```

Out[18]:
```
[array([  14,   16,   19,   49,   52,   53,   62,   63,   64,   66,   68,
          72,   74,   76,   77,   80,   89,   91,  109,  110,  111,  115,
         119,  125,  139,  141,  146,  147,  149,  151,  154,  161,  167,
         171,  174,  178,  179,  223,  468,  523,  557,  656,  693,  696,
         796,  840,  913, 1054, 1058, 1059, 1062, 1079, 1081, 1090, 1131,
        1133, 1142, 1175, 1196, 1211, 1214, 1240, 1250, 1280, 1327, 1378,
        1383, 1402, 1432], dtype=int64)]
```

In [19]:
```
model_influence = model.get_influence()
(c, _) = model_influence.cooks_distance
```

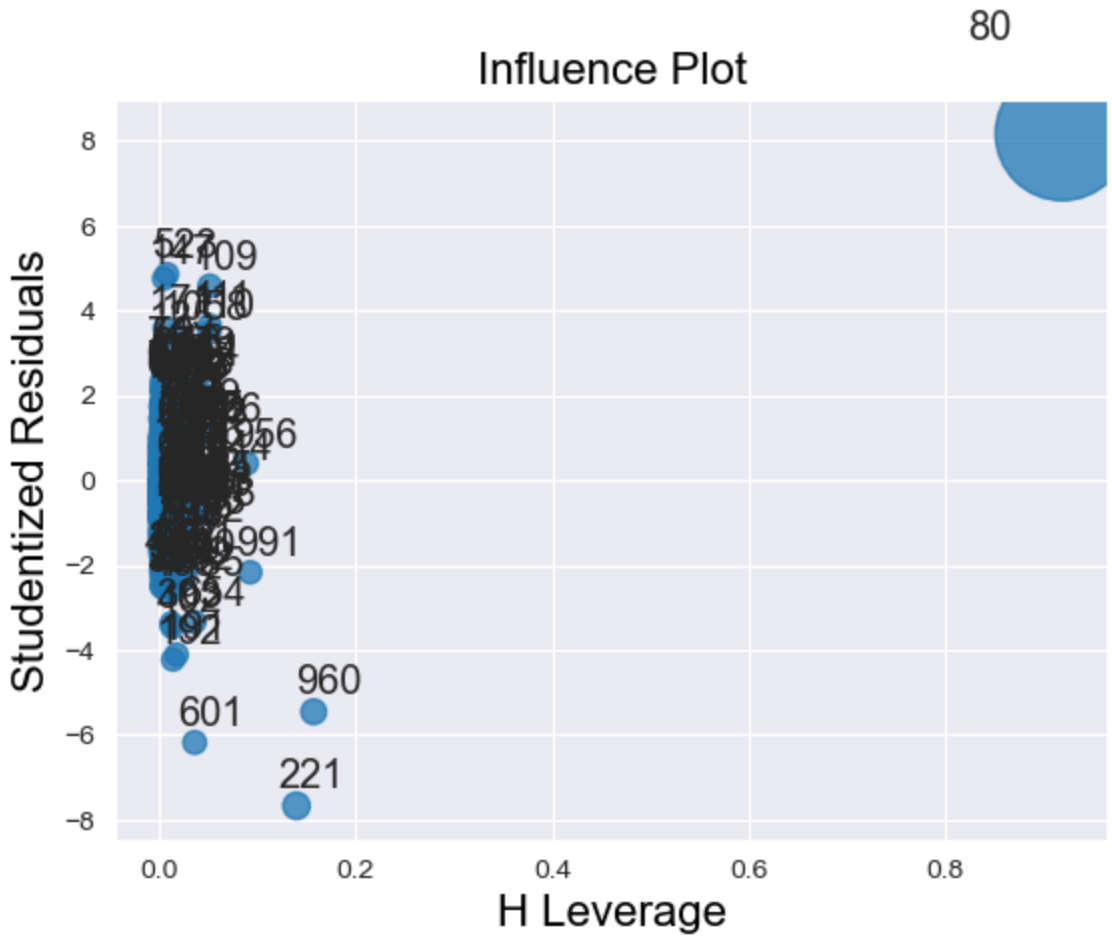
In [20]:
```
fig = plt.subplots(figsize=(20, 7))
plt.stem(np.arange(len(toyota)), np.round(c, 3))
plt.xlabel('Row index')
plt.ylabel('Cooks Distance')
plt.show()
```



In [21]:
```
(np.argmax(c),np.max(c))
```

Out[21]:
```
(80, 79.52010624138717)
```

In [22]:
```
influence_plot(model)
plt.show()
```



In [23]:
```
k = toyota.shape[1]
n = toyota.shape[0]
leverage_cutoff = 3*((k + 1)/n)
leverage_cutoff
```

Out[23]:
```
0.08147632311977715
```

In [24]:
```
toyota[toyota.index.isin([80,221,960])]
```

Out[24]:

	Id	Model	Price	Age_08_04	Mfg_Month	Mfg_Year	KM	Fuel_Type	HP	Met_Color	...	Central_Lo
80	81	TOYOTA Corolla 1.6 5drs 1 4/5-Doors	18950	25	8	2002	20019	Petrol	110	1	...	
221	223	TOYOTA Corolla 1.6 HB LINEA SOL 4/5-Doors	12450	44	1	2001	74172	Petrol	110	1	...	
960	964	TOYOTA Corolla	9390	66	3	1999	50806	Petrol	86	0	...	

3 rows × 38 columns

In [25]:
```
toyota.head()
```

Out[25]:

	Id	Model	Price	Age_08_04	Mfg_Month	Mfg_Year	KM	Fuel_Type	HP	Met_Color	...	Central_Lock
0	1	TOYOTA Corolla 2.0 D4D HATCHB TERRA 2/3-Doors	13500	23	10	2002	46986	Diesel	90	1	...	1
1	2	TOYOTA Corolla 2.0 D4D HATCHB TERRA 2/3-Doors	13750	23	10	2002	72937	Diesel	90	1	...	1
2	3	TOYOTA Corolla 2.0 D4D HATCHB TERRA 2/3-Doors	13950	24	9	2002	41711	Diesel	90	1	...	0
3	4	TOYOTA Corolla 2.0 D4D HATCHB TERRA 2/3-Doors	14950	26	7	2002	48000	Diesel	90	0	...	0
4	5	TOYOTA Corolla 2.0 D4D HATCHB SOL 2/3-Doors	13750	30	3	2002	38500	Diesel	90	0	...	1

5 rows × 38 columns

In [26]:
```
toyota_new=pd.read_csv('ToyotaCorolla.csv', encoding= 'unicode_escape')
```

In [27]:
```
toyota1=toyota_new.drop(toyota_new.index[[80,221,960]],axis=0).reset_index()
toyota1.shape
```

Out[27]:
```
(1433, 39)
```

In [28]:
```
toyota2=toyota1.drop(['index'],axis=1)
toyota2.shape
```

Out[28]:
```
(1433, 38)
```

In [29]:
```
final_ml_V= smf.ols('Price~Age_08_04+KM+HP+cc+Gears+Doors+Quarterly_Tax+Weight',data = t
```

In [30]:
```
model_influence_V = final_ml_V.get_influence()
(c_V, _) = model_influence_V.cooks_distance
```

In [31]:
```
fig= plt.subplots(figsize=(20,7))
plt.stem(np.arange(len(toyota2)),np.round(c_V,3));
plt.xlabel('Row index')
plt.ylabel('Cooks Distance');
```

In [32]:
```
(np.argmax(c_V),np.max(c_V))
```

Out[32]:
```
(599, 0.31661315281441693)
```

In [33]:
```
final_ml_V= smf.ols('Price~Age_08_04+KM+HP+cc+Gears+Doors+Quarterly_Tax+Weight',data =to
```

In [34]:
```
(final_ml_V.rsquared,final_ml_V.aic)
```

Out[34]:
```
(0.8851845904421739, 24469.715205158594)
```

In [35]:
```
new_data=pd.DataFrame({'Age_08_04':50,"KM":160,"HP":1100,"cc":225,"Gears":7,"Weight":250
```

In [36]:
```
final_ml_V.predict(new_data)
```

Out[36]:
```
1    31317.703254
dtype: float64
```

In [ ]:
```

```