Consider only the below columns and prepare a prediction model for predicting Price.

Corolla<-Corolla[c("Price","Age\_08\_04","KM","HP","cc","Doors","Gears","Quarterly\_Tax","Weight")]

Model -- model of the car

Price -- Offer Price in EUROs

Age\_08\_04 -- Age in months as in August 2004

Mfg\_Month -- Manufacturing month (1-12)

Mfg\_Year -- Manufacturing Year

KM -- Accumulated Kilometers on odometer

Fuel\_Type -- Fuel Type (Petrol, Diesel, CNG)

HP -- Horse Power

Met\_Color -- Metallic Color? (Yes=1, No=0)

Color -- Color (Blue, Red, Grey, Silver, Black, etc.)

Automatic -- Automatic ( (Yes=1, No=0)

cc -- Cylinder Volume in cubic centimeters

Doors -- Number of doors

Cylinders -- Number of cylinders

Gears -- Number of gear positions

Quarterly\_Tax -- Quarterly road tax in EUROs

Weight -- Weight in Kilograms

Mfr\_Guarantee -- Within Manufacturer's Guarantee period (Yes=1, No=0)

BOVAG\_Guarantee -- BOVAG (Dutch dealer network) Guarantee (Yes=1, No=0)

Guarantee\_Period -- Guarantee period in months

ABS -- Anti-Lock Brake System (Yes=1, No=0)

Airbag\_1 -- Driver\_Airbag (Yes=1, No=0)

Airbag\_2 -- Passenger Airbag (Yes=1, No=0)

Airco -- Airconditioning (Yes=1, No=0)

Automatic\_airco -- Automatic Airconditioning (Yes=1, No=0)

Boardcomputer -- Boardcomputer (Yes=1, No=0)

CD\_Player -- CD Player (Yes=1, No=0)

Central\_Lock -- Central Lock (Yes=1, No=0)

Powered\_Windows -- Powered Windows (Yes=1, No=0)

Power\_Steering -- Power Steering (Yes=1, No=0)

Radio -- Radio (Yes=1, No=0)

Mistlamps -- Mistlamps (Yes=1, No=0)

Sport\_Model -- Sport Model (Yes=1, No=0)

Backseat\_Divider -- Backseat Divider (Yes=1, No=0)

Metallic\_Rim --Metallic Rim (Yes=1, No=0)

Radio\_cassette -- Radio Cassette (Yes=1, No=0)

Tow\_Bar -- Tow Bar (Yes=1, No=0)

ANS: import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import statsmodels.formula.api as smf

import statsmodels.api as sm

from statsmodels.graphics.regressionplots import influence\_plot

toyo=pd.read\_csv('E:\\ToyotaCorolla.csv',encoding='latin1')

toyo.head()

toyo.info()

toyo2=pd.concat([toyo.iloc[:,2:4],toyo.iloc[:,6:7],toyo.iloc[:,8:9],toyo.iloc[:,12:14],toyo.iloc[:,15:18]],axis=1)

toyo2

toyo3=toyo2.rename({'Age\_08\_04':'Age','cc':'CC','Quarterly\_Tax':'QT'},axis=1)

toyo3.head()

toyo3[toyo3.duplicated()]

toyo4=toyo3.drop\_duplicates().reset\_index(drop=True)

toyo4

toyo4.describe()

toyo4.corr()

sns.set\_style(style='white')

sns.pairplot(toyo4)

model=smf.ols('Price~Age+KM+HP+CC+Doors+Gears+QT+Weight',data=toyo4).fit()

model.params

model.tvalues , np.round(model.pvalues,5)

model.rsquared , model.rsquared\_adj

slr\_c=smf.ols('Price~CC',data=toyo4).fit()

slr\_c.tvalues , slr\_c.pvalues

slr\_d=smf.ols('Price~Doors',data=toyo4).fit()

slr\_d.tvalues , slr\_d.pvalues

slr\_d=smf.ols('Price~CC+Doors',data=toyo4).fit()

slr\_d.tvalues , slr\_d.pvalues

rsq\_age=smf.ols('Age~KM+HP+CC+Doors+Gears+QT+Weight',data=toyo4).fit().rsquared

vif\_age=1/(1-rsq\_age)

rsq\_KM=smf.ols('KM~Age+HP+CC+Doors+Gears+QT+Weight',data=toyo4).fit().rsquared

vif\_KM=1/(1-rsq\_KM)

rsq\_HP=smf.ols('HP~Age+KM+CC+Doors+Gears+QT+Weight',data=toyo4).fit().rsquared

vif\_HP=1/(1-rsq\_HP)

rsq\_CC=smf.ols('CC~Age+KM+HP+Doors+Gears+QT+Weight',data=toyo4).fit().rsquared

vif\_CC=1/(1-rsq\_CC)

rsq\_DR=smf.ols('Doors~Age+KM+HP+CC+Gears+QT+Weight',data=toyo4).fit().rsquared

vif\_DR=1/(1-rsq\_DR)

rsq\_GR=smf.ols('Gears~Age+KM+HP+CC+Doors+QT+Weight',data=toyo4).fit().rsquared

vif\_GR=1/(1-rsq\_GR)

rsq\_QT=smf.ols('QT~Age+KM+HP+CC+Doors+Gears+Weight',data=toyo4).fit().rsquared

vif\_QT=1/(1-rsq\_QT)

rsq\_WT=smf.ols('Weight~Age+KM+HP+CC+Doors+Gears+QT',data=toyo4).fit().rsquared

vif\_WT=1/(1-rsq\_WT)

d1={'Variables':['Age','KM','HP','CC','Doors','Gears','QT','Weight'],

'Vif':[vif\_age,vif\_KM,vif\_HP,vif\_CC,vif\_DR,vif\_GR,vif\_QT,vif\_WT]}

Vif\_df=pd.DataFrame(d1)

Vif\_df

sm.qqplot(model.resid,line='q')

plt.title("Normal Q-Q plot of residuals")

plt.show()

list(np.where(model.resid>6000))

list(np.where(model.resid<-6000))

def standard\_values(vals) : return (vals-vals.mean())/vals.std()

plt.scatter(standard\_values(model.fittedvalues),standard\_values(model.resid))

plt.title('Residual Plot')

plt.xlabel('standardized fitted values')

plt.ylabel('standardized residual values')

plt.show()

fig=plt.figure(figsize=(15,8))

sm.graphics.plot\_regress\_exog(model,'Age',fig=fig)

plt.show()

fig=plt.figure(figsize=(15,8))

sm.graphics.plot\_regress\_exog(model,'KM',fig=fig)

plt.show()

fig=plt.figure(figsize=(15,8))

sm.graphics.plot\_regress\_exog(model,'HP',fig=fig)

plt.show()

fig=plt.figure(figsize=(15,8))

sm.graphics.plot\_regress\_exog(model,'CC',fig=fig)

plt.show()

fig=plt.figure(figsize=(15,8))

sm.graphics.plot\_regress\_exog(model,'Doors',fig=fig)

plt.show()

fig=plt.figure(figsize=(15,8))

sm.graphics.plot\_regress\_exog(model,'Gears',fig=fig)

plt.show()

fig=plt.figure(figsize=(15,8))

sm.graphics.plot\_regress\_exog(model,'QT',fig=fig)

plt.show()

fig=plt.figure(figsize=(15,8))

sm.graphics.plot\_regress\_exog(model,'Weight',fig=fig)

plt.show()

(c,\_)=model.get\_influence().cooks\_distance

c

fig=plt.figure(figsize=(20,7))

plt.stem(np.arange(len(toyo4)),np.round(c,3))

plt.xlabel('Row Index')

plt.ylabel('Cooks Distance')

plt.show()

np.argmax(c) , np.max(c)

fig,ax=plt.subplots(figsize=(20,20))

fig=influence\_plot(model,ax = ax)

k=toyo4.shape[1]

n=toyo4.shape[0]

leverage\_cutoff = (3\*(k+1))/n

leverage\_cutoff

toyo4[toyo4.index.isin([80])]

toyo\_new=toyo4.copy()

toyo\_new.head()

toyo5=toyo\_new.drop(toyo\_new.index[[80]],axis=0).reset\_index(drop=True)

toyo5

new\_data=pd.DataFrame({'Age':12,"KM":40000,"HP":80,"CC":1300,"Doors":4,"Gears":5,"QT":69,"Weight":1012},index=[0])

new\_data