In [139]:
```
#importing the libraries that are required
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [140]:
```
#loading the used cars data
df = pd.read_csv('used_cars.csv')
#displaying the first 5 rows of dataset
df.head()
```

Out[140]:

	Unnamed: 0	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Seats	Ne
0	0	Maruti Wagon R LXI CNG	Mumbai	2010	72000	CNG	Manual	First	26.6 km/kg	998 CC	58.16 bhp	5.0	
1	1	Hyundai Creta 1.6 CRDi SX Option	Pune	2015	41000	Diesel	Manual	First	19.67 kmpl	1582 CC	126.2 bhp	5.0	
2	2	Honda Jazz V	Chennai	2011	46000	Petrol	Manual	First	18.2 kmpl	1199 CC	88.7 bhp	5.0	8
3	3	Maruti Ertiga VDI	Chennai	2012	87000	Diesel	Manual	First	20.77 kmpl	1248 CC	88.76 bhp	7.0	
4	4	Audi A4 New 2.0 TDI Multitronic	Coimbatore	2013	40670	Diesel	Automatic	Second	15.2 kmpl	1968 CC	140.8 bhp	5.0	

In [141]:
```
df['Price'].isnull().sum()
```

Out[141]:
```
0
```

In [142]:
```
df['Seats'].isnull().sum()
```

Out[142]:
```
42
```

In [143]:
```
df['Seats'].fillna(value = df['Seats'].mode()[0],inplace = True)
df['Seats'].isnull().sum()
```

Out[143]:
```
0
```

In [144]:
```
#removing km/kg and kmpl from values
df['Mileage']=df['Mileage'].astype(str);
df['Mileage']= df['Mileage'].str[:-5]

#removing bhp from values
df['Power']=df['Power'].astype(str);
pow = list(map(lambda str : str.split(" ")[0],df['Power']))
df['Modified_Power'] = pow
df.drop(df[df['Modified_Power']=='null'].index,inplace=True)
df[df['Modified_Power']=='null']
df['Modified_Power']=df['Modified_Power'].astype('float')

df['Engine']=df['Engine'].astype(str);
eng = list(map(lambda str : str.split(" ")[0],df['Engine']))
df['Modified_Engine'] = eng
df['Modified_Engine']=df['Modified_Engine'].astype('float')
```

In [145]:
```
df.head()
```

Out[145]:

	Unnamed: 0	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Seats	Ne
0	0	Maruti Wagon R LXI CNG	Mumbai	2010	72000	CNG	Manual	First	26.6	998 CC	58.16 bhp	5.0	
1	1	Hyundai Creta 1.6 CRDi SX Option	Pune	2015	41000	Diesel	Manual	First	19.67	1582 CC	126.2 bhp	5.0	
2	2	Honda Jazz V	Chennai	2011	46000	Petrol	Manual	First	18.2	1199 CC	88.7 bhp	5.0	8
3	3	Maruti Ertiga VDI	Chennai	2012	87000	Diesel	Manual	First	20.77	1248 CC	88.76 bhp	7.0	
4	4	Audi A4 New 2.0 TDI Multitronic	Coimbatore	2013	40670	Diesel	Automatic	Second	15.2	1968 CC	140.8 bhp	5.0	

In [146]:
```
index = df.index
number_of_rows = len(index)
print(number_of_rows)
```

```
5912
```

Euclidean Distance

Column taken are Kilometers_Driven, Price and Seats.

In [147]:
```
import math
matrix = np.zeros(shape=(500,500))
cols=df[['Kilometers_Driven','Price','Seats']]

for i in range(0,500):
    for j in range(0,i):
        res=int(0)
        for k in range(0,3):
            res+=(cols.iloc[i,k]-cols.iloc[j,k])**2
        res=math.sqrt(res)
        matrix[i][j]=res
for i in range(0,500):
    for j in range(0,500):
        matrix[i][j]=matrix[j][i]
print(matrix)
```

```
[[    0.         31000.00186391 26000.00014543 ... 63413.00109974
  42000.00136012  7814.00025852]
 [31000.00186391     0.          5000.0064     ... 32413.00001733
  11000.00018466 23186.00164728]
 [26000.00014543  5000.0064         0.         ... 37413.00109699
  16000.00200195 18186.00001506]
 ...
 [63413.00109974 32413.00001733 37413.00109699 ...     0.
  21413.00013347 55599.00086368]
 [42000.00136012 11000.00018466 16000.00200195 ... 21413.00013347
      0.         34186.00111274]
 [ 7814.00025852 23186.00164728 18186.00001506 ... 55599.00086368
  34186.00111274     0.        ]]
```

Manhattan Distance

Column taken are Kilometers_Driven, Price and Seats.

In [148]:
```
matrix = np.zeros(shape=(500,500))
cols=df[['Kilometers_Driven','Price','Seats']]

for i in range(0,500):
    for j in range(0,i):
        res=int(0)
        for k in range(0,3):
            res+=abs(cal.iloc[i,k]-cal.iloc[j,k])

        matrix[i][j]=res
for i in range(0,500):
    for j in range(0,500):
        matrix[i][j]=matrix[j][i]
print(matrix)
```

```
[[    0.   31010.75 26002.75 ... 63424.81 42012.5   7816.01]
 [31010.75     0.    5008.   ... 32414.06 11002.25 23194.74]
 [26002.75  5008.       0.   ... 37422.06 16009.75 18186.74]
 ...
 [63424.81 32414.06 37422.06 ...     0.   21416.31 55608.8 ]
 [42012.5  11002.25 16009.75 ... 21416.31     0.   34196.49]
 [ 7816.01 23194.74 18186.74 ... 55608.8  34196.49     0.  ]]
```

Supremum Distance

Column taken are Kilometers_Driven, Price and Seats.

In [149]:
```
matrix = np.zeros(shape=(500,500))
cols=df[['Kilometers_Driven','Price','Seats']]

for i in range(0,500):
    for j in range(0,i):
        res=int(0)
        max=int(0)
        for k in range(0,3):
            res+=(cal.iloc[i,k]-cal.iloc[j,k])
            res=abs(res)
            if(res>max):
                max=ans
        matrix[i][j]=res
for i in range(0,500):
    for j in range(0,500):
        matrix[i][j]=matrix[j][i]
print(matrix)
```

```
[[    0.   31010.75 26002.75 ... 63424.81 42012.5   7816.01]
 [31010.75     0.    4992.   ... 32414.06 11001.75 23177.26]
 [26002.75  4992.       0.   ... 37422.06 16009.75 18185.26]
 ...
 [63424.81 32414.06 37422.06 ...     0.   21413.69 55589.2 ]
 [42012.5  11001.75 16009.75 ... 21413.69     0.   34175.51]
 [ 7816.01 23177.26 18185.26 ... 55589.2  34175.51     0.  ]]
```

Mahalanobis Distance

Column taken are Kilometers_Driven, Price and Seats.

In [150]:
```
Kilometers_Driven_mean=df['Kilometers_Driven'].mean()
Price_mean=df['Price'].mean()
Seats_mean=df['Seats'].mean()
```

In [151]:
```
count=df['Kilometers_Driven'].count()
count=count-1
```

In [152]:
```
cols=df[['Kilometers_Driven','Price','Seats']]
Kilometers_Driven=[]
Price=[]
Seats=[]
for i in range(number_of_rows):
    Kilometers_Driven.append(cols.iloc[i][0]-Kilometers_Driven_mean)
    Price.append(cols.iloc[i][1]-Price_mean)
    Seats.append(cols.iloc[i][1]-Seats_mean)
print(Kilometers_Driven)
```

```
89, -18055.655277401893, -5801.655277401893, -3377.655277401893, -11377.655277401893, -35214.65527740189, -8
377.655277401893, -46477.65527740189, -21377.655277401893, -28377.655277401893, -45546.65527740189, 84639.34
47225981, -45275.65527740189, -33427.65527740189, -545.6552774018928, -24110.655277401893, -25377.6552774018
93, -3277.655277401893, -18377.655277401893, -26536.655277401893, -43377.65527740189, 24378.344722598107, -3
7777.65527740189, -28425.655277401893, -43377.65527740189, 3560.344722598107, 8622.344722598107, 3294.344722
598107, -32192.655277401893, -20377.655277401893, 8622.344722598107, 7622.344722598107, 22505.344722598107,
21841.344722598107, -45454.65527740189, 28264.344722598107, -39142.65527740189, -7375.655277401893, -51577.6
5527740189, 24371.344722598107, -7377.655277401893, -26912.655277401893, 18098.344722598107, -44377.65527740
189, -11377.655277401893, -9911.655277401893, -12377.655277401893, 104285.3447225981, -33235.65527740189, 10
622.344722598107, -31972.655277401893, -13377.655277401893, 5906.344722598107, -31377.655277401893, -54699.6
5527740189, -37377.65527740189, 8622.344722598107, -34377.65527740189, 16622.344722598107, 120622.344722598
1, -11179.655277401893, -6377.655277401893, 32622.344722598107, 1622.3447225981072, -20504.655277401893, -27
377.655277401893, 70622.3447225981, 19622.344722598107, -26377.655277401893, 622.3447225981072, -22940.65527
7401893, 39622.34472259811, 13622.344722598107, 20622.344722598107, -29377.655277401893, 7634.344722598107,
-23377.655277401893, 16622.344722598107, 21622.344722598107, -25660.655277401893, -33377.65527740189, 20622.
344722598107, 27622.344722598107, 78622.3447225981, -33377.65527740189, -49877.65527740189, -24377.655277401
893, -16748.655277401893, 21849.344722598107, -49790.65527740189, -28377.655277401893, 5808.344722598107, 18
049.344722598107, 16622.344722598107, -31302.655277401893, -28377.655277401893, 11056.344722598107, -2377.65
5277401893, -9102.655277401893, -49677.65527740189, -40377.65527740189, -17336.655277401893, 22086.344722598
```

In [153]:
```
def cov(x,y):
    sum=int(0)
    count=len(x)
    for i in range(count):
        sum+=x[i]*y[i]
    return sum/(count-1)
cov(Kilometers_Driven,Kilometers_Driven)
```

Out[153]: 8445331047.230176

In [154]:
```
matrix=np.zeros([3,3])
matrix
```

Out[154]:
```
array([[0., 0., 0.],
       [0., 0., 0.],
       [0., 0., 0.]])
```

In [155]:
```
matrix[0][0]=cov(Kilometers_Driven,Kilometers_Driven)
matrix[1][1]=cov(Price,Price)
matrix[2][2]=cov(Seats,Seats)

matrix[0][1]=cov(Kilometers_Driven,Price)
matrix[0][2]=cov(Kilometers_Driven,Seats)
matrix[1][0]=cov(Price,Kilometers_Driven)
matrix[1][2]=cov(Price,Seats)
matrix[2][0]=cov(Seats,Kilometers_Driven)
matrix[2][1]=cov(Seats,Price)
```

In [156]:
```
matrix
inverse=np.linalg.inv(matrix)
inverse
```

Out[156]:
```
array([[ 1.18417191e-10,  8.24111981e-09, -8.11918889e-22],
       [ 8.24111981e-09,  6.19205405e-02, -5.39957782e-02],
       [-8.08268543e-22, -5.39957782e-02,  5.39957782e-02]])
```

In [157]:
```
mahalanobis=[]
for i in range(count):
    values=cols.iloc[i]
    values.iloc[0]=values.iloc[0]-Kilometers_Driven_mean
    values.iloc[1]=values.iloc[1]-Price_mean
    values.iloc[2]=values.iloc[2]-Seats_mean
    values2=values.transpose()
    res = np.dot(values,inverse)
    mahalanobis.append(np.dot(res,values2))
print(mahalanobis)
```

```
195589, 5.137168797259135, 2.4453055586233856, 0.5343777484051249, 28.344313176663228, 0.1318862807324257,
0.6396234177340909, 3.6989667944617253, 1.2274106890799403, 0.7631238623450951, 2.927543112976714, 0.2047541
308828299, 1.1704602071364418, 0.7811736474409334, 0.9410653587766902, 0.4860781088093421, 6.34903815171380

8, 2.393791514542668, 1.6083385952324245, 1.863058412452842, 0.36586243013952185, 3.766178426069202, 0.31999
1994868422, 2.3170011512081885, 1.231915004828083, 19.317662148516206, 0.2513721929913856, 0.497414067608996
87, 0.6866082331812151, 1.5896316361741398, 3.350356562381849, 3.0673750055840188, 3.6300971279155068, 2.587
821279037542, 61.61960675127371, 19.33119008737668, 0.4303265412523043, 3.9938244896258044, 56.3485233808672
85, 2.2739959185731844, 0.17693459811113274, 0.4060060830981931, 1.187766909901752, 3.536469775069794, 4.887
991793811247, 2.0606038616932816, 3.321489857413657, 2.8858582335817684, 0.22233346857818234, 0.907604584277
0795, 9.83726923131865, 9.954590444549307, 2.7784838636429483, 1.056724194981031, 2.936570677183617, 2.26237
0162581431, 0.44902880483896584, 4.645198687378954, 0.3313267905583271, 0.34686135400762697, 1.0637332121977
288, 2.584704216828344, 2.774295758444042, 1.7659846874428495, 0.6203703466285699, 2.375362486781725, 1.7806
367422871383, 1.4986708386575391, 0.776181879387869, 1.4774602780698123, 151.21619083744943, 4.2173511173886
13, 3.39343810021474, 2.5418581970381684, 0.9446574526066901, 2.339125073267179, 1.6882466288076305, 0.24460
170152431282, 0.5360204496839455, 476.6614938519664, 3.20183402132675, 2.6436877713762645, 23.45713019075604
7, 0.9700701276968695, 1.821186327665626, 2.885165273116876, 1.6458883580684758, 39.20006985177712, 3.330698
768764117, 0.2938280991771579, 27.404484089579483, 0.6699279590058128, 0.9973849342003381, 0.110636809775141
18, 1.6063372064303827, 2.3623318412485506, 1.2037562911262212, 0.9423950205826761, 0.30490098624907574, 0.2
6864393865261116, 0.3149054354751377, 2.5002733240516966, 10.822182372644985, 0.48376361236481125, 1.6051568
```