Regresi Linear Berganda Car Dekho

Ika Lulus Yuliatin

Car Dekho is an Indian auto portal that helps its users with car research, finance, insurance, used cars, and any other aspect of car buying and selling. The company has tie-ups with many auto manufacturers, car dealers, and numerous financial institutions to facilitate the purchase of vehicles.

In this report, we will do data visualization analysis from 2 kinds of variable continue and 6 kinds of variable discrete The details of variables included in the dataset are :

1) Car Name 2) Year 3) Selling Price 4) Kms driven 5) Fuel 6) Seller type 7) Transmission 8) Owner

This report is a continuation of the previous detailed report which carried out further visual analysis of the fuel in the selling cars

Source of Data : https://www.kaggle.com/datasets/akshaydattatraykhare/car-details-dataset/code

In [1]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import os

import statsmodels.api as sm
```

In [2]:
```
pd.set_option('display.max_rows', 500)
pd.set_option('display.max_columns', 500)
pd.set_option('display.width', 1000)

pd.reset_option('display.float_format')
pd.options.display.float_format = '{:.5f}'.format
```

In [4]:
```
car=pd.read_csv('CAR DETAILS FROM CAR DEKHO.csv')
```

In [5]:
```
car.head()
```

Out[5]:

	name	year	selling_price	km_driven	fuel	seller_type	transmission	owner
0	Maruti 800 AC	2007	60000	70000	Petrol	Individual	Manual	First Owner
1	Maruti Wagon R LXI Minor	2007	135000	50000	Petrol	Individual	Manual	First Owner
2	Hyundai Verna 1.6 SX	2012	600000	100000	Diesel	Individual	Manual	First Owner
3	Datsun RediGO T Option	2017	250000	46000	Petrol	Individual	Manual	First Owner
4	Honda Amaze VX i-DTEC	2014	450000	141000	Diesel	Individual	Manual	Second Owner

In [6]:
```
car.isnull().sum()
```

Out[6]:
```
name             0
year             0
selling_price    0
km_driven        0
fuel             0
seller_type      0
transmission     0
owner            0
dtype: int64
```

In [7]:
```
car.describe(include = 'all')
```

Out[7]:

	name	year	selling_price	km_driven	fuel	seller_type	transmission	owner
count	4340	4340.00000	4340.00000	4340.00000	4340	4340	4340	4340
unique	1491	NaN	NaN	NaN	5	3	2	5
top	Maruti Swift Dzire VDI	NaN	NaN	NaN	Diesel	Individual	Manual	First Owner
freq	69	NaN	NaN	NaN	2153	3244	3892	2832
mean	NaN	2013.09078	504127.31175	66215.77742	NaN	NaN	NaN	NaN
std	NaN	4.21534	578548.73614	46644.10219	NaN	NaN	NaN	NaN
min	NaN	1992.00000	20000.00000	1.00000	NaN	NaN	NaN	NaN
25%	NaN	2011.00000	208749.75000	35000.00000	NaN	NaN	NaN	NaN
50%	NaN	2014.00000	350000.00000	60000.00000	NaN	NaN	NaN	NaN
75%	NaN	2016.00000	600000.00000	90000.00000	NaN	NaN	NaN	NaN
max	NaN	2020.00000	8900000.00000	806599.00000	NaN	NaN	NaN	NaN

In [8]:
```
# creating model column
car['manufacturer']=car.name.str.split().str.get(0)
```

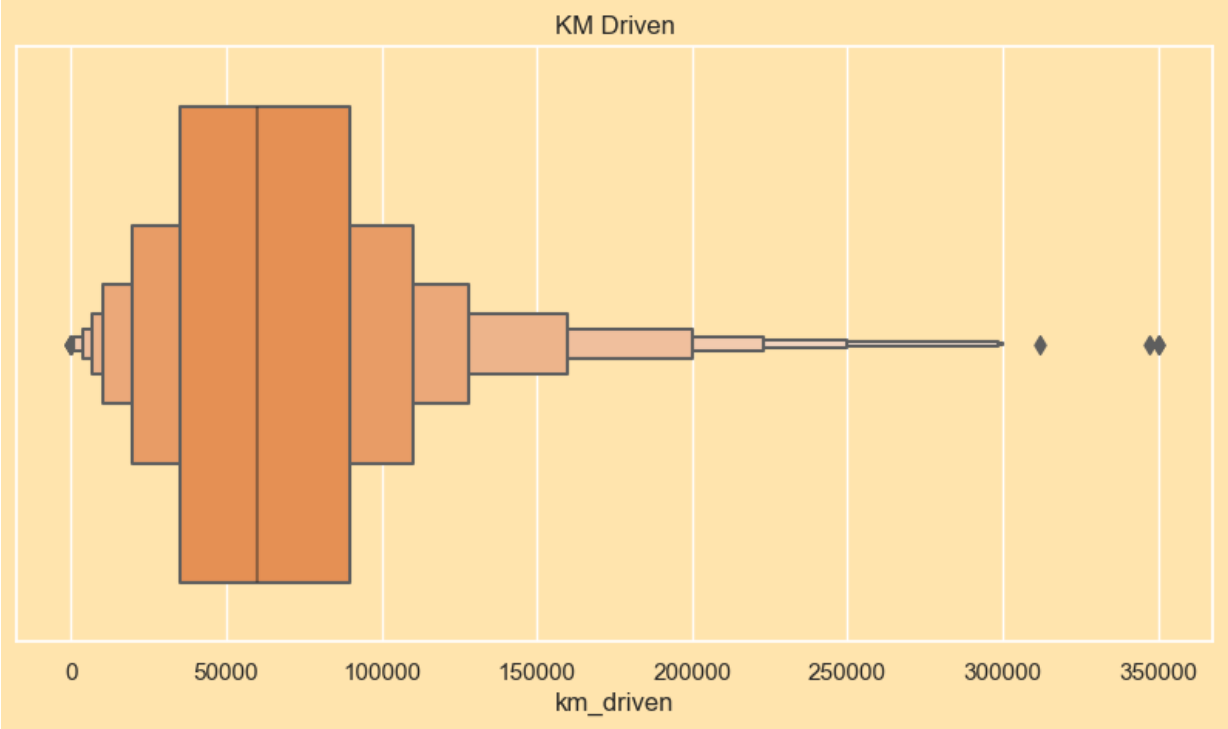
In [9]:
```
car.head()
```

Out[9]:

	name	year	selling_price	km_driven	fuel	seller_type	transmission	owner	manufacturer
0	Maruti 800 AC	2007	60000	70000	Petrol	Individual	Manual	First Owner	Maruti
1	Maruti Wagon R LXI Minor	2007	135000	50000	Petrol	Individual	Manual	First Owner	Maruti
2	Hyundai Verna 1.6 SX	2012	600000	100000	Diesel	Individual	Manual	First Owner	Hyundai
3	Datsun RediGO T Option	2017	250000	46000	Petrol	Individual	Manual	First Owner	Datsun
4	Honda Amaze VX i-DTEC	2014	450000	141000	Diesel	Individual	Manual	Second Owner	Honda

In [16]:
```
sns.set(rc={"axes.facecolor":"#ffe4ad","figure.facecolor":"#ffe4ad"})
pallet = ["#682F2F", "#9E726F", "#D6B2B1", "#B9C0C9", "#9F8A78", "#F3AB60"]
f, ax = plt.subplots(figsize=(10, 5))
ax.ticklabel_format(style='plain', axis='both')
sns.boxenplot(car.km_driven, ax=ax, palette='Oranges').set_title('KM Driven')
plt.show()
```

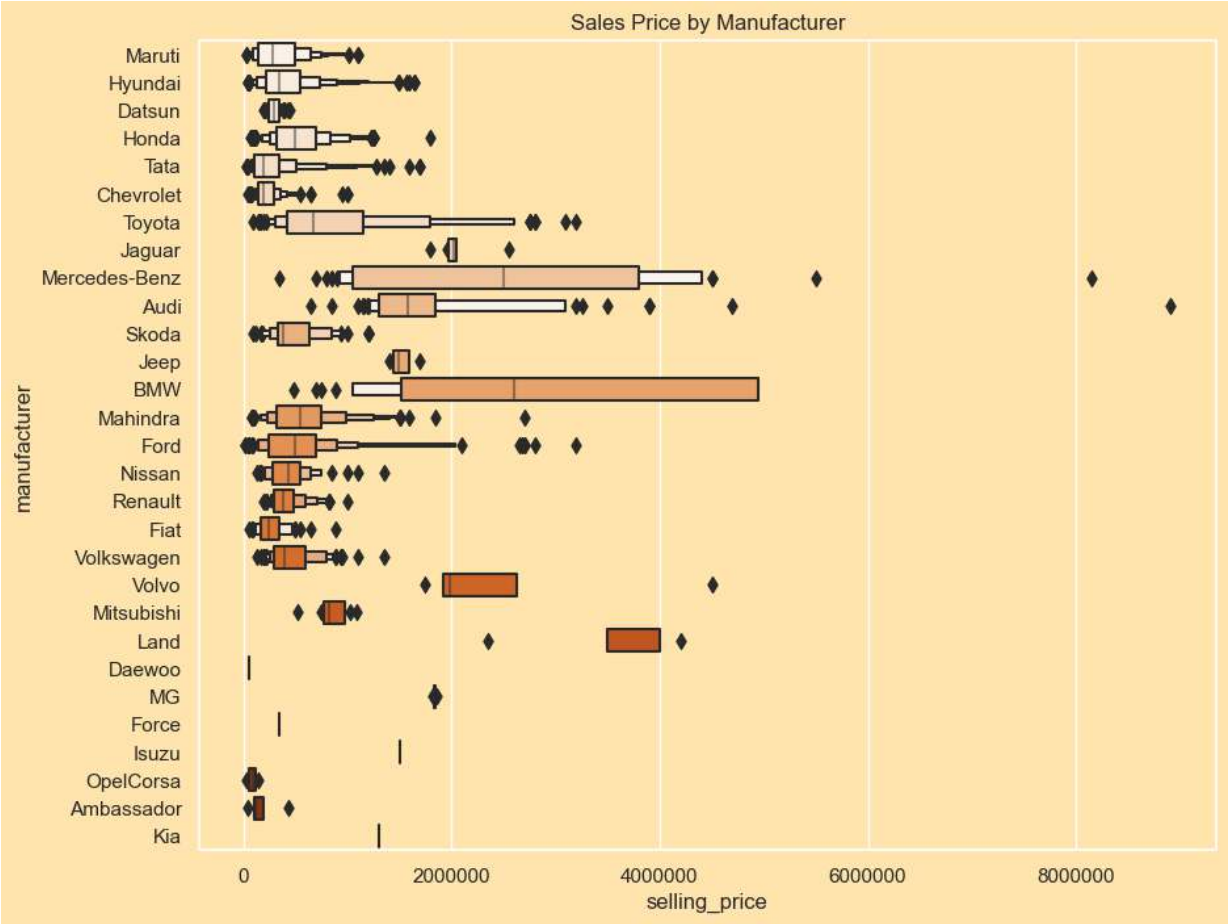
```
C:\Users\User\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarn
ing: Pass the following variable as a keyword arg: x. From version 0.12, the on
ly valid positional argument will be `data`, and passing other arguments withou
t an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```



Removing outliers

In [11]:
```
# removing outliers
car=car[car.km_driven<400000]
```

In [19]:
```
sns.set(rc={"axes.facecolor":"#ffe4ad","figure.facecolor":"#ffe4ad"})
pallet = ["#682F2F", "#9E726F", "#D6B2B1", "#B9C0C9", "#9F8A78", "#F3AB60"]
f, ax = plt.subplots(figsize=(10, 8))
ax.ticklabel_format(style='plain', axis='both')
sns.boxenplot(data= car, x='selling_price', y='manufacturer', ax=ax, palette='Ora
plt.show()
```



In [20]:
```
car.manufacturer.value_counts()
```

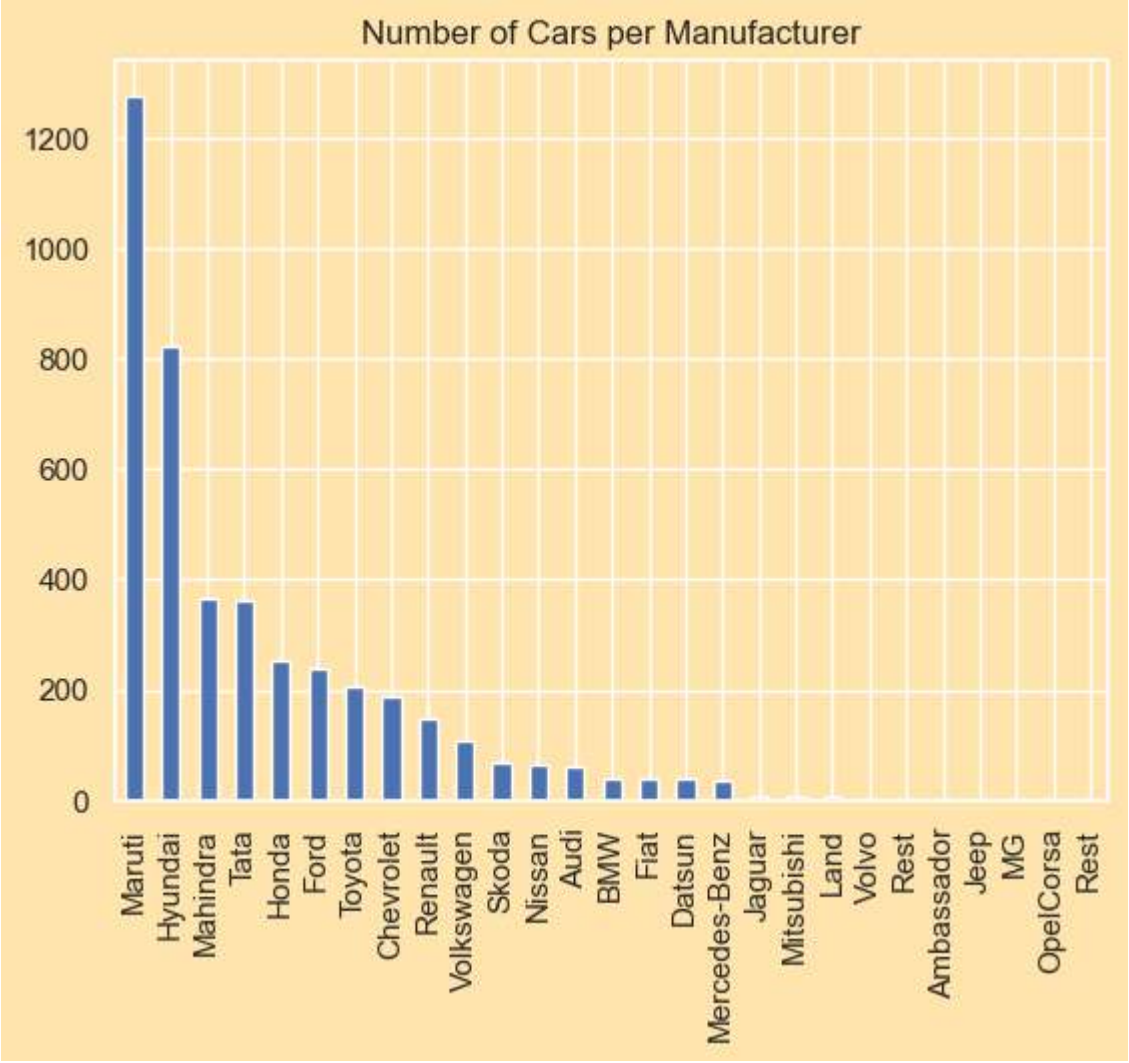
Out[20]:
```
Maruti           1277
Hyundai           821
Mahindra          365
Tata              361
Honda             252
Ford              238
Toyota            205
Chevrolet         188
Renault           146
Volkswagen        107
Skoda              68
Nissan             64
Audi               60
BMW                39
Fiat               37
Datsun             37
Mercedes-Benz      35
Jaguar              6
Mitsubishi          6
Land                5
Volvo               4
Ambassador          4
Jeep                3
MG                  2
OpelCorsa           2
Daewoo              1
Force               1
Isuzu               1
Kia                 1
Name: manufacturer, dtype: int64
```

if number of records per manufacturer is 1 then 'Rest'

In [21]:
```
cars_rest_filter=car.manufacturer.value_counts()[car.manufacturer.value_counts()=
```

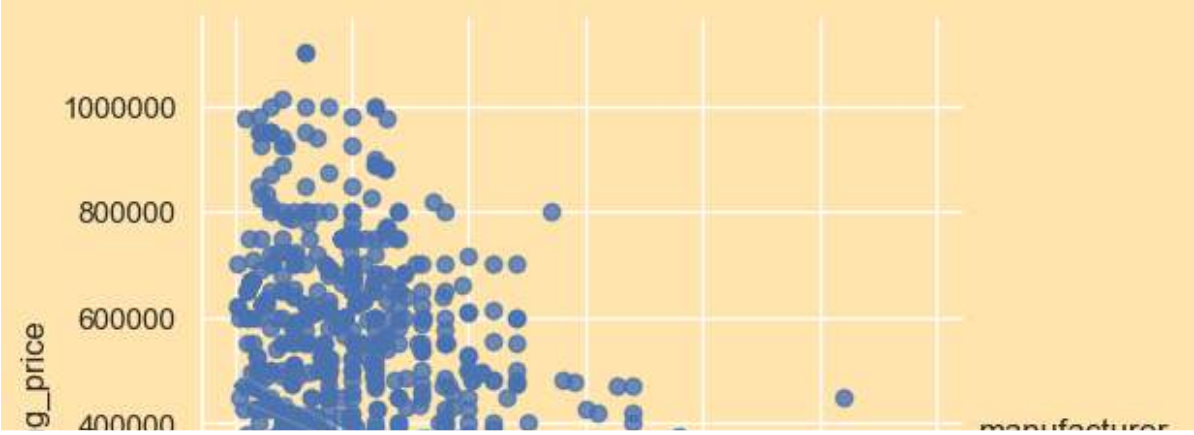
In [22]:
```
car.loc[(car.manufacturer.isin(cars_rest_filter)), 'manuf']='Rest'
car.loc[~(car.manufacturer.isin(cars_rest_filter)), 'manuf']=car['manufacturer']
```

In [25]:
```
pd.concat([car.manuf.value_counts(), pd.Series(car.manuf.value_counts(), index=['
plt.title('Number of Cars per Manufacturer')
plt.show()
```



In [26]:
```
for i in car.manuf.unique():
    sns.lmplot(x='km_driven', y='selling_price',data=car[(car.manuf==i)], hue='ma
    plt.ticklabel_format(style='plain', axis='y')
```

```
C:\Users\User\anaconda3\lib\site-packages\seaborn\axisgrid.py:409: RuntimeWar
ning: More than 20 figures have been opened. Figures created through the pypl
ot interface (`matplotlib.pyplot.figure`) are retained until explicitly close
d and may consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).
  fig = plt.figure(figsize=figsize)
```



Resgression

In [27]:
```
car=car.join(pd.get_dummies(car.fuel))
```

In [28]:
```
car=car.join(pd.get_dummies(car.seller_type))
```

In [29]:
```
car=car.join(pd.get_dummies(car.transmission))
```

In [30]:
```
car=car.join(pd.get_dummies(car.owner))
```

In [31]:
```
car=car.join(pd.get_dummies(car.manuf))
```

In [32]:
```
y=car.selling_price
```

In [33]:
```
car.head().T
```

Out[33]:

	0	1	2	3	4
name	Maruti 800 AC	Maruti Wagon R LXI Minor	Hyundai Verna 1.6 SX	Datsun RediGO T Option	Honda Amaze VX i-DTEC
year	2007	2007	2012	2017	2014
selling_price	60000	135000	600000	250000	450000
km_driven	70000	50000	100000	46000	141000
fuel	Petrol	Petrol	Diesel	Petrol	Diesel
seller_type	Individual	Individual	Individual	Individual	Individual
transmission	Manual	Manual	Manual	Manual	Manual
owner	First Owner	First Owner	First Owner	First Owner	Second Owner
manufacturer	Maruti	Maruti	Hyundai	Datsun	Honda
manuf	Maruti	Maruti	Hyundai	Datsun	Honda
CNG	0	0	0	0	0
Diesel	0	0	1	0	1
Electric	0	0	0	0	0
LPG	0	0	0	0	0
Petrol	1	1	0	1	0
Dealer	0	0	0	0	0
Individual	1	1	1	1	1
Trustmark Dealer	0	0	0	0	0
Automatic	0	0	0	0	0
Manual	1	1	1	1	1
First Owner	1	1	1	1	0
Fourth & Above Owner	0	0	0	0	0
Second Owner	0	0	0	0	1
Test Drive Car	0	0	0	0	0
Third Owner	0	0	0	0	0
Ambassador	0	0	0	0	0
Audi	0	0	0	0	0
BMW	0	0	0	0	0
Chevrolet	0	0	0	0	0
Datsun	0	0	0	1	0
Fiat	0	0	0	0	0
Ford	0	0	0	0	0

	0	1	2	3	4
Honda	0	0	0	0	1
Hyundai	0	0	1	0	0
Jaguar	0	0	0	0	0
Jeep	0	0	0	0	0
Land	0	0	0	0	0
MG	0	0	0	0	0
Mahindra	0	0	0	0	0
Maruti	1	1	0	0	0
Mercedes-Benz	0	0	0	0	0
Mitsubishi	0	0	0	0	0
Nissan	0	0	0	0	0
OpelCorsa	0	0	0	0	0
Renault	0	0	0	0	0
Rest	0	0	0	0	0
Skoda	0	0	0	0	0
Tata	0	0	0	0	0
Toyota	0	0	0	0	0
Volkswagen	0	0	0	0	0
Volvo	0	0	0	0	0

In [34]:
```
X=car.drop(['name','selling_price', 'fuel', 'seller_type', 'transmission', 'owner
```

In [35]:
```
X = sm.add_constant(X)
```

In [36]:
```
model = sm.OLS(y, X)
results = model.fit()
print(results.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:          selling_price   R-squared:                       0.689
Model:                            OLS   Adj. R-squared:                  0.686
Method:                 Least Squares   F-statistic:                     250.4
Date:                Sat, 17 Dec 2022   Prob (F-statistic):               0.00
Time:                        20:30:16   Log-Likelihood:                -61153.
No. Observations:                4336   AIC:                         1.224e+05
Df Residuals:                    4297   BIC:                         1.226e+05
Df Model:                          38
Covariance Type:            nonrobust
==============================================================================
=========
                         coef    std err          t      P>|t|      [0.025
0.975]
------------------------------------------------------------------------------
---------
const                -3.366e+07   1.33e+06    -25.228      0.000   -3.63e+07
-3.1e+07
year                  3.855e+04   1504.516     25.624      0.000    3.56e+04
4.15e+04
km_driven               -1.2718      0.144     -8.818      0.000      -1.555
-0.989
CNG                  -6.694e+06   2.81e+05    -23.847      0.000   -7.24e+06
-6.14e+06
Diesel               -6.495e+06   2.78e+05    -23.402      0.000   -7.04e+06
-5.95e+06
Electric             -7.147e+06   3.71e+05    -19.262      0.000   -7.87e+06
-6.42e+06
LPG                  -6.644e+06   2.79e+05    -23.780      0.000   -7.19e+06
-6.1e+06
Petrol               -6.676e+06   2.74e+05    -24.328      0.000   -7.21e+06
-6.14e+06
Dealer                -1.13e+07   4.44e+05    -25.446      0.000   -1.22e+07
-1.04e+07
Individual           -1.131e+07   4.45e+05    -25.432      0.000   -1.22e+07
-1.04e+07
Trustmark Dealer     -1.105e+07   4.46e+05    -24.756      0.000   -1.19e+07
-1.02e+07
Automatic            -1.667e+07   6.68e+05    -24.950      0.000    -1.8e+07
-1.54e+07
Manual               -1.699e+07   6.66e+05    -25.502      0.000   -1.83e+07
-1.57e+07
First Owner          -6.757e+06    2.7e+05    -25.061      0.000   -7.29e+06
-6.23e+06
Fourth & Above Owner -6.725e+06   2.64e+05    -25.507      0.000   -7.24e+06
-6.21e+06
Second Owner         -6.791e+06   2.66e+05    -25.521      0.000   -7.31e+06
-6.27e+06
Test Drive Car       -6.602e+06   2.82e+05    -23.397      0.000   -7.15e+06
-6.05e+06
Third Owner          -6.781e+06   2.65e+05    -25.601      0.000    -7.3e+06
-6.26e+06
```

```
Ambassador          -1.803e+06   1.64e+05    -11.013      0.000   -2.12e+06
-1.48e+06
Audi                -7.021e+05   6.85e+04    -10.254      0.000   -8.36e+05
-5.68e+05
BMW                  2.039e+05   7.51e+04      2.714      0.007    5.66e+04
3.51e+05
Chevrolet           -1.949e+06   5.98e+04    -32.578      0.000   -2.07e+06
-1.83e+06
Datsun              -2.029e+06   7.96e+04    -25.504      0.000   -2.19e+06
-1.87e+06
Fiat                -1.918e+06   7.56e+04    -25.364      0.000   -2.07e+06
-1.77e+06
Ford                -1.752e+06   6.02e+04    -29.112      0.000   -1.87e+06
-1.63e+06
Honda               -1.749e+06      6e+04    -29.133      0.000   -1.87e+06
-1.63e+06
Hyundai             -1.822e+06   5.75e+04    -31.705      0.000   -1.93e+06
-1.71e+06
Jaguar              -4.324e+05   1.39e+05     -3.110      0.002   -7.05e+05
-1.6e+05
Jeep                -9.685e+05    1.9e+05     -5.106      0.000   -1.34e+06
-5.97e+05
Land                 9.766e+05   1.51e+05      6.466      0.000    6.81e+05
1.27e+06
MG                  -8.436e+05   2.29e+05     -3.690      0.000   -1.29e+06
-3.95e+05
Mahindra            -1.725e+06   5.87e+04    -29.376      0.000   -1.84e+06
-1.61e+06
Maruti               -1.85e+06   5.67e+04    -32.620      0.000   -1.96e+06
-1.74e+06
Mercedes-Benz        1.597e+05   7.53e+04      2.122      0.034    1.21e+04
3.07e+05
Mitsubishi           -1.31e+06   1.39e+05     -9.412      0.000   -1.58e+06
-1.04e+06
Nissan              -1.841e+06    6.9e+04    -26.678      0.000   -1.98e+06
-1.71e+06
OpelCorsa            -1.64e+06   2.27e+05     -7.228      0.000   -2.08e+06
-1.19e+06
Renault             -1.941e+06   6.35e+04    -30.553      0.000   -2.07e+06
-1.82e+06
Rest                -1.599e+06   1.64e+05     -9.718      0.000   -1.92e+06
-1.28e+06
Skoda               -1.836e+06   6.62e+04    -27.734      0.000   -1.97e+06
-1.71e+06
Tata                -1.928e+06    5.8e+04    -33.213      0.000   -2.04e+06
-1.81e+06
Toyota              -1.378e+06   6.11e+04    -22.577      0.000    -1.5e+06
-1.26e+06
Volkswagen          -1.848e+06   6.36e+04    -29.034      0.000   -1.97e+06
-1.72e+06
Volvo                -1.33e+05   1.67e+05     -0.798      0.425    -4.6e+05
1.94e+05
==============================================================================
Omnibus:                     4224.341   Durbin-Watson:                   2.016
Prob(Omnibus):                  0.000   Jarque-Bera (JB):          1006308.930
Skew:                           4.124   Prob(JB):                         0.00
Kurtosis:                      77.175   Cond. No.                     1.20e+16
```

```
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctl
y specified.
[2] The smallest eigenvalue is 1.87e-19. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
```

In [ ]: