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

In [2]:
```
df=pd.read_csv("Desktop/Studies/Datasets/Quikr_car.csv")
df
```

Out[2]:

	Unnamed: 0	Name	Label	Location	Price	Kms_driven	Fuel_type	Owner	Year	Company
0	0	Ford Figo Duratec Petrol EXI 1.2 - 2015	PLATINUM	Bangalore	₹3,80,000	35,056 kms	Petrol	NaN	2015	Ford
1	1	Maruti Suzuki Wagon R VXI BS IV - 2016	PLATINUM	Bangalore	₹4,65,000	44,000 kms	Petrol	NaN	2016	Maruti
2	2	Hyundai Creta 1.6 SX PLUS AUTO PETROL - 2018	PLATINUM	Bangalore	₹13,50,000	42,917 kms	Petrol	NaN	2018	Hyundai
3	3	Hyundai Venue - 2019	PLATINUM	Chennai	₹10,19,699	16,112 kms	Petrol	2nd Owner	2019	Hyundai
4	4	Honda Jazz - 2017	PLATINUM	Pune	₹7,13,499	30,988 kms	Petrol	2nd Owner	2017	Honda
...	...	...	...	...	...	...	...	...	...	...
1027	1027	Hyundai i10 Magna 1.2 - 2014	GOLD	Bangalore	₹2,29,000	65,000 kms	Petrol	1st Owner	2014	Hyundai
1028	1028	Maruti Suzuki Alto K10 LXi CNG - 2014	GOLD	Bangalore	₹2,75,000	60,000 kms	Petrol	NaN	2014	Maruti
1029	1029	Jeep Compass Limited 2.0 Diesel - 2017	GOLD	Mahasamund	₹17,50,000	31,000 kms	Diesel	1st Owner	2017	Jeep
1030	1030	Jeep Compass Limited 2.0 Diesel - 2017	GOLD	Bilaspur	₹17,50,000	31,000 kms	Diesel	1st Owner	2017	Jeep
1031	1031	Jeep Compass Limited 2.0 Diesel - 2017	GOLD	Jagdalpur	₹17,50,000	31,000 kms	Diesel	1st Owner	2017	Jeep

1032 rows × 10 columns

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1032 entries, 0 to 1031
Data columns (total 10 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   Unnamed: 0  1032 non-null   int64 
 1   Name        1032 non-null   object
 2   Label       1032 non-null   object
 3   Location    1030 non-null   object
 4   Price       1032 non-null   object
 5   Kms_driven  1032 non-null   object
 6   Fuel_type   1032 non-null   object
 7   Owner       740 non-null    object
 8   Year        1032 non-null   int64 
 9   Company     1032 non-null   object
dtypes: int64(2), object(8)
memory usage: 80.8+ KB
```

In [4]:
```
df.describe()
```

Out[4]:

	Unnamed: 0	Year
count	1032.000000	1032.000000
mean	515.500000	2015.810078
std	298.057042	3.396920
min	0.000000	2000.000000
25%	257.750000	2014.000000
50%	515.500000	2016.000000
75%	773.250000	2018.000000
max	1031.000000	2022.000000

In [5]:
```
df.drop(['Unnamed: 0'],axis=1,inplace=True)
```

In [6]:
```
df
```

Out[6]:

	Name	Label	Location	Price	Kms_driven	Fuel_type	Owner	Year	Company
0	Ford Figo Duratec Petrol EXI 1.2 - 2015	PLATINUM	Bangalore	₹3,80,000	35,056 kms	Petrol	NaN	2015	Ford
1	Maruti Suzuki Wagon R VXI BS IV - 2016	PLATINUM	Bangalore	₹4,65,000	44,000 kms	Petrol	NaN	2016	Maruti
2	Hyundai Creta 1.6 SX PLUS AUTO PETROL - 2018	PLATINUM	Bangalore	₹13,50,000	42,917 kms	Petrol	NaN	2018	Hyundai
3	Hyundai Venue - 2019	PLATINUM	Chennai	₹10,19,699	16,112 kms	Petrol	2nd Owner	2019	Hyundai
4	Honda Jazz - 2017	PLATINUM	Pune	₹7,13,499	30,988 kms	Petrol	2nd Owner	2017	Honda
...	...	...	...	...	...	...	...	...	...
1027	Hyundai i10 Magna 1.2 - 2014	GOLD	Bangalore	₹2,29,000	65,000 kms	Petrol	1st Owner	2014	Hyundai
1028	Maruti Suzuki Alto K10 LXi CNG - 2014	GOLD	Bangalore	₹2,75,000	60,000 kms	Petrol	NaN	2014	Maruti
1029	Jeep Compass Limited 2.0 Diesel - 2017	GOLD	Mahasamund	₹17,50,000	31,000 kms	Diesel	1st Owner	2017	Jeep
1030	Jeep Compass Limited 2.0 Diesel - 2017	GOLD	Bilaspur	₹17,50,000	31,000 kms	Diesel	1st Owner	2017	Jeep
1031	Jeep Compass Limited 2.0 Diesel - 2017	GOLD	Jagdalpur	₹17,50,000	31,000 kms	Diesel	1st Owner	2017	Jeep

1032 rows × 9 columns

In [7]:
```
len(df['Name'].unique())
```

Out[7]:
```
691
```

In [8]:
```
df['Name']=df['Name'].str.split (" ")
df['Name']
```

Out[8]:
```
0        [Ford, Figo, Duratec, Petrol, EXI, 1.2, -, 2015]
1       [Maruti, Suzuki, Wagon, R, VXI, BS, IV, -, 2016]
2       [Hyundai, Creta, 1.6, SX, PLUS, AUTO, PETROL, ...
3                          [Hyundai, Venue, -, 2019]
4                             [Honda, Jazz, -, 2017]
                              ...
1027                [Hyundai, i10, Magna, 1.2, -, 2014]
1028       [Maruti, Suzuki, Alto, K10, LXi, CNG, -, 2014]
1029       [Jeep, Compass, Limited, 2.0, Diesel, -, 2017]
1030       [Jeep, Compass, Limited, 2.0, Diesel, -, 2017]
1031       [Jeep, Compass, Limited, 2.0, Diesel, -, 2017]
Name: Name, Length: 1032, dtype: object
```

In [9]:
```
df['Name']=df['Name'].str.slice(0,3).str.join(' ')
df['Name']
```

Out[9]:
```
0          Ford Figo Duratec
1        Maruti Suzuki Wagon
2          Hyundai Creta 1.6
3            Hyundai Venue -
4               Honda Jazz -
                ...
1027       Hyundai i10 Magna
1028      Maruti Suzuki Alto
1029    Jeep Compass Limited
1030    Jeep Compass Limited
1031    Jeep Compass Limited
Name: Name, Length: 1032, dtype: object
```

In [10]:
```
hyphenReplace=lambda x:x.replace('-','')
df['Name']=df['Name'].apply(hyphenReplace)
df['Name']
```

Out[10]:
```
0          Ford Figo Duratec
1        Maruti Suzuki Wagon
2          Hyundai Creta 1.6
3             Hyundai Venue
4                Honda Jazz
```

```
              ...
1027       Hyundai i10 Magna
1028      Maruti Suzuki Alto
1029    Jeep Compass Limited
1030    Jeep Compass Limited
1031    Jeep Compass Limited
Name: Name, Length: 1032, dtype: object
```

In [11]:
```
for i in range(9):
    print("The number of null values in column",i,"are:")
    nullCheck=df.iloc[:,i].isnull().sum()
    print(nullCheck)
```

```
The number of null values in column 0 are:
0
The number of null values in column 1 are:
0
The number of null values in column 2 are:
2
The number of null values in column 3 are:
0
The number of null values in column 4 are:
0
The number of null values in column 5 are:
0
The number of null values in column 6 are:
292
The number of null values in column 7 are:
0
The number of null values in column 8 are:
0
```

In [12]:
```
df['Location'].unique()
```

Out[12]:
```
array(['Bangalore', 'Chennai', 'Pune', 'Hyderabad', 'Faridabad',
       'Kanchipuram', 'Kozhikode', 'Kolkata', 'Mumbai', 'Lucknow',
       'Coimbatore', 'Delhi', 'Nanded', 'Malappuram', 'Chandigarh',
       'Madurai', 'BolpurSantiniketan', 'Bhubaneswar', nan, 'Pondicherry',
       'Udaipur', 'NaviMumbai', 'Gurgaon', 'Kurnool', 'Surat',
       'Ahmedabad', 'Jaipur', 'Thane', 'Nagpur', 'Dwarka', 'GirSomnath',
       'Anand', 'Muzaffarnagar', 'Trichy', 'Uttarpara', 'Kochi',
       'Dhanbad', 'Mahasamund', 'Bilaspur', 'Jagdalpur'], dtype=object)
```

In [13]:
```
df[df['Location'].isnull()]
```

Out[13]:

	Name	Label	Location	Price	Kms_driven	Fuel_type	Owner	Year	Company
283	Maruti Suzuki Maruti	GOLD	NaN	₹65,000	1,08,252 kms	Petrol	NaN	2005	Maruti
887	Honda City ZX	GOLD	NaN	₹10,69,999	54,000 kms	CNG	1st Owner	2018	Honda

In [14]:
```
nullLocation=df[df['Location'].isnull()].index
nullLocation
```

Out[14]:
```
Int64Index([283, 887], dtype='int64')
```

In [15]:
```
df=df.drop(nullLocation)
df
```

Out[15]:

	Name	Label	Location	Price	Kms_driven	Fuel_type	Owner	Year	Company
0	Ford Figo Duratec	PLATINUM	Bangalore	₹3,80,000	35,056 kms	Petrol	NaN	2015	Ford
1	Maruti Suzuki Wagon	PLATINUM	Bangalore	₹4,65,000	44,000 kms	Petrol	NaN	2016	Maruti
2	Hyundai Creta 1.6	PLATINUM	Bangalore	₹13,50,000	42,917 kms	Petrol	NaN	2018	Hyundai
3	Hyundai Venue	PLATINUM	Chennai	₹10,19,699	16,112 kms	Petrol	2nd Owner	2019	Hyundai
4	Honda Jazz	PLATINUM	Pune	₹7,13,499	30,988 kms	Petrol	2nd Owner	2017	Honda
...	...	...	...	...	...	...	...	...	...
1027	Hyundai i10 Magna	GOLD	Bangalore	₹2,29,000	65,000 kms	Petrol	1st Owner	2014	Hyundai

1028	Maruti Suzuki Alto	GOLD	Bangalore	₹2,75,000	60,000 kms	Petrol	NaN	2014	Maruti
1029	Jeep Compass Limited	GOLD	Mahasamund	₹17,50,000	31,000 kms	Diesel	1st Owner	2017	Jeep
1030	Jeep Compass Limited	GOLD	Bilaspur	₹17,50,000	31,000 kms	Diesel	1st Owner	2017	Jeep
1031	Jeep Compass Limited	GOLD	Jagdalpur	₹17,50,000	31,000 kms	Diesel	1st Owner	2017	Jeep

1030 rows × 9 columns

In [16]:
```
for i in range(9):
    print("The number of null values in column",i,"are:")
    nullCheck=df.iloc[:,i].isnull().sum()
    print(nullCheck)
```

```
The number of null values in column 0 are:
0
The number of null values in column 1 are:
0
The number of null values in column 2 are:
0
The number of null values in column 3 are:
0
The number of null values in column 4 are:
0
The number of null values in column 5 are:
0
The number of null values in column 6 are:
291
The number of null values in column 7 are:
0
The number of null values in column 8 are:
0
```

In [17]:
```
len(df['Price'].unique())
```

Out[17]:
```
802
```

In [18]:
```
df['Price']=df['Price'].str.replace(',','')
df['Price']
```

Out[18]:
```
0        ₹380000
1        ₹465000
2       ₹1350000
3       ₹1019699
4        ₹713499
          ...
1027     ₹229000
1028     ₹275000
1029    ₹1750000
1030    ₹1750000
1031    ₹1750000
Name: Price, Length: 1030, dtype: object
```

In [19]:
```
df['Price']=df['Price'].str.replace('₹','')
df['Price']
```

Out[19]:
```
0        380000
1        465000
2       1350000
3       1019699
4        713499
          ...
1027     229000
1028     275000
1029    1750000
1030    1750000
1031    1750000
Name: Price, Length: 1030, dtype: object
```

In [20]:
```
df['Price'].min()
```

Out[20]:
```
'1000000'
```

In [21]:
```
df[df['Price']=='Ask For Price'].index
```

Out[21]:
```
Int64Index([260, 316, 356, 546, 644, 934], dtype='int64')
```

In [22]:
```
askForPrice=df[df['Price']=='Ask For Price']
```

In [23]:
```
askForPrice=df[df['Price']=='Ask For Price'].index
askForPrice
```

Out[23]:
```
Int64Index([260, 316, 356, 546, 644, 934], dtype='int64')
```

In [24]:
```
df=df.drop(askForPrice)
df
```

Out[24]:

	Name	Label	Location	Price	Kms_driven	Fuel_type	Owner	Year	Company
0	Ford Figo Duratec	PLATINUM	Bangalore	380000	35,056 kms	Petrol	NaN	2015	Ford
1	Maruti Suzuki Wagon	PLATINUM	Bangalore	465000	44,000 kms	Petrol	NaN	2016	Maruti
2	Hyundai Creta 1.6	PLATINUM	Bangalore	1350000	42,917 kms	Petrol	NaN	2018	Hyundai
3	Hyundai Venue	PLATINUM	Chennai	1019699	16,112 kms	Petrol	2nd Owner	2019	Hyundai
4	Honda Jazz	PLATINUM	Pune	713499	30,988 kms	Petrol	2nd Owner	2017	Honda
...	...	...	...	...	...	...	...	...	...
1027	Hyundai i10 Magna	GOLD	Bangalore	229000	65,000 kms	Petrol	1st Owner	2014	Hyundai
1028	Maruti Suzuki Alto	GOLD	Bangalore	275000	60,000 kms	Petrol	NaN	2014	Maruti
1029	Jeep Compass Limited	GOLD	Mahasamund	1750000	31,000 kms	Diesel	1st Owner	2017	Jeep
1030	Jeep Compass Limited	GOLD	Bilaspur	1750000	31,000 kms	Diesel	1st Owner	2017	Jeep
1031	Jeep Compass Limited	GOLD	Jagdalpur	1750000	31,000 kms	Diesel	1st Owner	2017	Jeep

1024 rows × 9 columns

In [25]:
```
askForPrice=df[df['Price']=='Ask For Price'].index
askForPrice
```

Out[25]:
```
Int64Index([], dtype='int64')
```

In [26]:
```
df['Price'].min()
```

Out[26]:
```
'1000000'
```

In [27]:
```
df['Price']=pd.to_numeric(df['Price'])
df['Price']
```

Out[27]:
```
0        380000
1        465000
2       1350000
3       1019699
4        713499
         ...
1027     229000
1028     275000
1029    1750000
1030    1750000
1031    1750000
Name: Price, Length: 1024, dtype: int64
```

In [28]:
```
df['Price'].min()
```

Out[28]: 39000

In [29]:
```
df['Price'].max()
```

Out[29]: 7500000

In [30]:
```
df['Kms_driven'].unique()
```

Out[30]:
```
array(['35,056 kms ', '44,000 kms ', '42,917 kms ', '16,112 kms ',
       '30,988 kms ', '69,163 kms ', '42,859 kms ', '34,919 kms ',
       '44,940 kms ', '13,687 kms ', '29,848 kms ', '21,426 kms ',
       '68,177 kms ', '70,650 kms ', '25,730 kms ', '47,096 kms ',
       '1,31,000 kms ', '26,000 kms ', '58,460 kms ', '48,000 kms ',
       '26,600 kms ', '36,000 kms ', '39,500 kms ', '76,000 kms ',
       '23,698 kms ', '37,860 kms ', '91,830 kms ', '31,172 kms ',
       '3,243 kms ', '9,616 kms ', '30,249 kms ', '60,875 kms ',
       '25,740 kms ', '38,587 kms ', '3,108 kms ', '35,353 kms ',
       '34,748 kms ', '2,359 kms ', '17,814 kms ', '68,934 kms ',
       '72,600 kms ', '81,000 kms ', '80,000 kms ', '12,000 kms ',
       '6,000 kms ', '56,000 kms ', '15,840 kms ', '32,482 kms ',
       '1,13,779 kms ', '26,150 kms ', '34,555 kms ', '30,880 kms ',
       '70,423 kms ', '30,232 kms ', '50,077 kms ', '18,482 kms ',
       '10,246 kms ', '25,021 kms ', '70,646 kms ', '45,490 kms ',
       '53,306 kms ', '29,000 kms ', '1,10,000 kms ', '95,000 kms ',
       '1,70,000 kms ', '58,000 kms ', '22,000 kms ', '69,215 kms ',
       '33,340 kms ', '9,783 kms ', '15,110 kms ', '3,399 kms ',
       '89,310 kms ', '53,148 kms ', '42,020 kms ', '74,594 kms ',
       '6,110 kms ', '24,346 kms ', '65,668 kms ', '48,644 kms ',
       '36,877 kms ', '79,673 kms ', '54,352 kms ', '64,000 kms ',
       '54,000 kms ', '40,000 kms ', '1,07,000 kms ', '38,000 kms ',
       '86,000 kms ', '10,000 kms ', '48,658 kms ', '11,998 kms ',
       '32,513 kms ', '21,372 kms ', '34,178 kms ', '20,547 kms ',
       '34,731 kms ', '30,536 kms ', '10,205 kms ', '25,483 kms ',
       '39,113 kms ', '43,152 kms ', '5,325 kms ', '44,705 kms ',
       '70,022 kms ', '47,435 kms ', '89,000 kms ', '9,826 kms ',
       '83,000 kms ', '1,95,000 kms ', '35,000 kms ', '13,000 kms ',
       '46,000 kms ', '72,000 kms ', '29,026 kms ', '59,258 kms ',
       '1,20,399 kms ', '28,853 kms ', '9,811 kms ', '43,538 kms ',
       '66,400 kms ', '59,341 kms ', '56,246 kms ', '35,944 kms ',
       '17,699 kms ', '41,065 kms ', '2,311 kms ', '2,991 kms ',
       '29,762 kms ', '91,917 kms ', '0 kms ', '85,000 kms ',
       '37,000 kms ', '67,000 kms ', '1,20,000 kms ', '71,000 kms ',
       '1,05,000 kms ', '30,771 kms ', '29,488 kms ', '36,559 kms ',
       '38,242 kms ', '66,426 kms ', '16,950 kms ', '18,866 kms ',
       '53,245 kms ', '44,543 kms ', '50,128 kms ', '56,349 kms ',
       '69,524 kms ', '41,218 kms ', '9,085 kms ', '11,292 kms ',
       '75,000 kms ', '28,000 kms ', '51,000 kms ', '24,990 kms ',
       '48,600 kms ', '3,600 kms ', '1,15,000 kms ', '29,594 kms ',
       '55,953 kms ', '16,980 kms ', '41,828 kms ', '64,803 kms ',
       '44,460 kms ', '14,276 kms ', '23,425 kms ', '23,056 kms ',
       '16,102 kms ', '56,615 kms ', '70,317 kms ', '6,760 kms ',
       '8,870 kms ', '78,209 kms ', '64,292 kms ', '57,000 kms ',
       '42,000 kms ', '1,41,000 kms ', '82,000 kms ', '1,17,000 kms ',
       '70,000 kms ', '89,450 kms ', '58,771 kms ', '53,266 kms ',
       '9,200 kms ', '61,721 kms ', '18,333 kms ', '51,780 kms ',
       '30,854 kms ', '24,680 kms ', '85,663 kms ', '20,541 kms ',
       '64,182 kms ', '58,819 kms ', '32,057 kms ', '18,403 kms ',
       '7,877 kms ', '15,509 kms ', '15,000 kms ', '62,000 kms ',
       '1,34,500 kms ', '63,896 kms ', '90,000 kms ', '48,329 kms ',
       '21,301 kms ', '34,566 kms ', '34,297 kms ', '1,19,571 kms ',
       '27,957 kms ', '27,210 kms ', '21,610 kms ', '40,902 kms ',
       '11,057 kms ', '72,505 kms ', '31,338 kms ', '28,674 kms ',
       '25,416 kms ', '14,765 kms ', '60,380 kms ', '73,000 kms ',
       '55,000 kms ', '79,950 kms ', '19,775 kms ', '28,495 kms ',
       '28,344 kms ', '10,801 kms ', '4,365 kms ', '50,866 kms ',
       '24,230 kms ', '64,275 kms ', '8,924 kms ', '12,208 kms ',
       '26,956 kms ', '12,223 kms ', '23,960 kms ', '33,687 kms ',
       '24,887 kms ', '74,000 kms ', '45,000 kms ', '41,000 kms ',
       '61,000 kms ', '20,000 kms ', '71,423 kms ', '34,095 kms ',
       '32,658 kms ', '33,166 kms ', '31,284 kms ', '66,090 kms ',
       '51,295 kms ', '48,693 kms ', '36,626 kms ', '55,752 kms ',
       '49,341 kms ', '60,251 kms ', '22,269 kms ', '38,999 kms ',
       '29,110 kms ', '14,506 kms ', '60,000 kms ', '88,000 kms ',
       '29,115 kms ', '34,763 kms ', '12,500 kms ', '1,24,000 kms ',
       '78,000 kms ', '44,696 kms ', '33,004 kms ', '46,034 kms ',
       '60,307 kms ', '8,825 kms ', '5,743 kms ', '87,000 kms ',
       '19,500 kms ', '65,000 kms ', '35,623 kms ', '1,60,000 kms ',
       '98,500 kms ', '25,735 kms ', '37,522 kms ', '38,955 kms ',
       '65,308 kms ', '30,467 kms ', '1,954 kms ', '35,220 kms ',
       '75,876 kms ', '27,020 kms ', '13,739 kms ', '21,544 kms ',
       '13,816 kms ', '23,817 kms ', '60,985 kms ', '30,123 kms ',
       '26,123 kms ', '60,123 kms ', '52,123 kms ', '11,123 kms ',
```

'41,233 kms ', '50,141 kms ', '86,033 kms ', '30,000 kms ',
'86,924 kms ', '68,870 kms ', '22,292 kms ', '44,013 kms ',
'96,250 kms ', '59,839 kms ', '49,693 kms ', '22,223 kms ',
'8,030 kms ', '38,780 kms ', '31,919 kms ', '47,564 kms ',
'42,836 kms ', '27,000 kms ', '6,90,000 kms ', '1,35,000 kms ',
'50,488 kms ', '12,916 kms ', '30,731 kms ', '21,627 kms ',
'5,580 kms ', '41,371 kms ', '42,313 kms ', '74,913 kms ',
'74,760 kms ', '34,045 kms ', '1,43,147 kms ', '28,149 kms ',
'38,049 kms ', '78,895 kms ', '27,027 kms ', '66,000 kms ',
'47,000 kms ', '82,650 kms ', '23,940 kms ', '26,678 kms ',
'41,342 kms ', '59,614 kms ', '39,550 kms ', '28,621 kms ',
'60,437 kms ', '26,598 kms ', '62,723 kms ', '21,145 kms ',
'48,699 kms ', '10,102 kms ', '34,881 kms ', '53,997 kms ',
'49,076 kms ', '65,951 kms ', '1,16,000 kms ', '1,36,000 kms ',
'63,000 kms ', '15,032 kms ', '15,575 kms ', '1,47,819 kms ',
'35,203 kms ', '7,597 kms ', '34,883 kms ', '59,580 kms ',
'49,647 kms ', '67,577 kms ', '10,890 kms ', '57,444 kms ',
'33,598 kms ', '17,804 kms ', '62,613 kms ', '56,974 kms ',
'1,405 kms ', '8,000 kms ', '1,18,000 kms ', '23,000 kms ',
'34,000 kms ', '36,965 kms ', '25,526 kms ', '20,235 kms ',
'61,064 kms ', '12,558 kms ', '14,179 kms ', '20,379 kms ',
'42,837 kms ', '67,473 kms ', '27,576 kms ', '80,900 kms ',
'1,00,934 kms ', '29,060 kms ', '36,129 kms ', '1,56,442 kms ',
'48,710 kms ', '97,000 kms ', '77,000 kms ', '20,619 kms ',
'51,149 kms ', '12,978 kms ', '40,025 kms ', '63,075 kms ',
'78,724 kms ', '12,257 kms ', '26,890 kms ', '20,485 kms ',
'69,220 kms ', '41,900 kms ', '11,944 kms ', '66,143 kms ',
'16,563 kms ', '14,460 kms ', '47,120 kms ', '79,200 kms ',
'48,877 kms ', '32,000 kms ', '14,000 kms ', '79,000 kms ',
'1,47,960 kms ', '38,766 kms ', '8,409 kms ', '20,718 kms ',
'51,125 kms ', '64,459 kms ', '29,467 kms ', '18,019 kms ',
'40,710 kms ', '1,956 kms ', '21,531 kms ', '20,989 kms ',
'33,901 kms ', '54,696 kms ', '50,000 kms ', '53,000 kms ',
'52,000 kms ', '1,49,000 kms ', '27,317 kms ', '79,784 kms ',
'73,656 kms ', '27,664 kms ', '41,898 kms ', '4,889 kms ',
'39,836 kms ', '16,108 kms ', '46,111 kms ', '23,761 kms ',
'1,46,782 kms ', '26,597 kms ', '12,323 kms ', '3,230 kms ',
'62,059 kms ', '28,188 kms ', '49,000 kms ', '39,226 kms ',
'32,877 kms ', '17,283 kms ', '17,419 kms ', '24,416 kms ',
'26,700 kms ', '28,979 kms ', '27,150 kms ', '3,195 kms ',
'26,761 kms ', '71,992 kms ', '27,239 kms ', '36,244 kms ',
'69,807 kms ', '31,803 kms ', '92,373 kms ', '2,00,000 kms ',
'1,79,000 kms ', '59,000 kms ', '59,612 kms ', '30,212 kms ',
'18,113 kms ', '48,435 kms ', '29,297 kms ', '53,525 kms ',
'15,890 kms ', '31,694 kms ', '36,279 kms ', '33,349 kms ',
'1,03,182 kms ', '63,048 kms ', '26,358 kms ', '18,932 kms ',
'20,369 kms ', '58,769 kms ', '1,55,000 kms ', '1,23,920 kms ',
'17,900 kms ', '31,041 kms ', '61,875 kms ', '29,210 kms ',
'35,826 kms ', '29,364 kms ', '16,304 kms ', '95,731 kms ',
'31,011 kms ', '30,601 kms ', '14,568 kms ', '89,559 kms ',
'48,834 kms ', '74,190 kms ', '43,077 kms ', '99,000 kms ',
'88,500 kms ', '16,000 kms ', '17,981 kms ', '20,780 kms ',
'56,719 kms ', '31,108 kms ', '33,611 kms ', '43,700 kms ',
'49,920 kms ', '57,338 kms ', '23,096 kms ', '48,732 kms ',
'67,924 kms ', '30,418 kms ', '50,593 kms ', '1,36,381 kms ',
'46,943 kms ', '36,277 kms ', '5,600 kms ', '1,50,000 kms ',
'70,359 kms ', '42,205 kms ', '60,539 kms ', '16,533 kms ',
'26,596 kms ', '30,912 kms ', '17,951 kms ', '36,928 kms ',
'22,854 kms ', '89,156 kms ', '41,999 kms ', '66,145 kms ',
'29,157 kms ', '33,882 kms ', '29,379 kms ', '17,352 kms ',
'16,189 kms ', '41,341 kms ', '19,000 kms ', '91,000 kms ',
'13,486 kms ', '33,562 kms ', '13,691 kms ', '31,980 kms ',
'69,526 kms ', '21,492 kms ', '47,425 kms ', '40,988 kms ',
'51,983 kms ', '69,376 kms ', '33,094 kms ', '26,237 kms ',
'42,415 kms ', '1,12,308 kms ', '61,427 kms ', '59,888 kms ',
'58,492 kms ', '75,691 kms ', '84,301 kms ', '38,346 kms ',
'41,530 kms ', '23,049 kms ', '74,510 kms ', '20,141 kms ',
'85,860 kms ', '63,133 kms ', '71,894 kms ', '95,627 kms ',
'75,324 kms ', '64,611 kms ', '77,128 kms ', '36,306 kms ',
'83,803 kms ', '1,08,000 kms ', '700 kms ', '25 kms ', '258 kms ',
'81,152 kms ', '75,398 kms ', '42,317 kms ', '60,214 kms ',
'58,861 kms ', '27,995 kms ', '61,018 kms ', '67,109 kms ',
'81,811 kms ', '27,357 kms ', '53,293 kms ', '45,260 kms ',
'3,139 kms ', '44,690 kms ', '13,858 kms ', '25,256 kms ',
'94,000 kms ', '44,100 kms ', '1,26,000 kms ', '15,227 kms ',
'87,375 kms ', '7,679 kms ', '48,378 kms ', '54,645 kms ',
'7,211 kms ', '37,217 kms ', '32,372 kms ', '37,947 kms ',
'51,886 kms ', '43,187 kms ', '33,616 kms ', '91,087 kms ',
'26,372 kms ', '58,762 kms ', '1,01,920 kms ', '63,088 kms ',
'54,123 kms ', '35,140 kms ', '50,842 kms ', '43,412 kms ',
'46,201 kms ', '63,626 kms ', '18,251 kms ', '41,019 kms ',
'46,463 kms ', '20,110 kms ', '17,364 kms ', '35,109 kms ',
'44,266 kms ', '39,156 kms ', '57,993 kms ', '51,573 kms ',
'39,000 kms ', '87,500 kms ', '19,900 kms ', '78,329 kms ',
'61,977 kms ', '58,640 kms ', '40,650 kms ', '19,121 kms ',
'85,893 kms ', '9,874 kms ', '25,202 kms ', '95,999 kms ',
'49,074 kms ', '52,439 kms ', '10,813 kms ', '1,27,971 kms ',
'1,14,108 kms ', '52,030 kms ', '44,721 kms ', '17,000 kms ',
'52,500 kms ', '81,500 kms ', '22,667 kms ', '68,519 kms ',

```
       '64,011 kms ', '13,598 kms ', '47,037 kms ', '69,711 kms ',
       '49,189 kms ', '18,368 kms ', '16,874 kms ', '24,484 kms ',
       '20,688 kms ', '24,065 kms ', '22,609 kms ', '56,523 kms ',
       '48,031 kms ', '45,578 kms ', '70,070 kms ', '30,030 kms ',
       '25,000 kms ', '68,000 kms ', '28,648 kms ', '59,819 kms ',
       '19,574 kms ', '46,337 kms ', '73,563 kms ', '51,428 kms ',
       '95,754 kms ', '34,533 kms ', '50,154 kms ', '23,492 kms ',
       '37,739 kms ', '46,178 kms ', '42,893 kms ', '57,875 kms ',
       '4,832 kms ', '27,805 kms ', '1,40,000 kms ', '68,844 kms ',
       '37,070 kms ', '20,393 kms ', '22,244 kms ', '47,024 kms ',
       '64,707 kms ', '52,149 kms ', '11,203 kms ', '18,071 kms ',
       '81,412 kms ', '41,912 kms ', '32,516 kms ', '41,552 kms ',
       '21,790 kms ', '34,010 kms ', '50,050 kms ', '2,000 kms ',
       '1,25,000 kms ', '75,822 kms ', '28,589 kms ', '15,883 kms ',
       '63,648 kms ', '1,56,931 kms ', '13,829 kms ', '28,361 kms ',
       '77,960 kms ', '33,473 kms ', '29,052 kms ', '68,041 kms ',
       '61,361 kms ', '32,218 kms ', '28,579 kms ', '51,343 kms ',
       '86,868 kms ', '35,500 kms ', '19,942 kms ', '17,357 kms ',
       '14,517 kms ', '13,517 kms ', '42,302 kms ', '65,783 kms ',
       '1,38,716 kms ', '43,339 kms ', '19,526 kms ', '25,673 kms ',
       '93,304 kms ', '94,409 kms ', '66,799 kms ', '27,339 kms ',
       '9,095 kms ', '21,934 kms ', '51,706 kms ', '31,281 kms ',
       '46,312 kms ', '32,989 kms ', '50,599 kms ', '26,127 kms ',
       '58,118 kms ', '23,534 kms ', '58,978 kms ', '33,103 kms ',
       '14,600 kms ', '26,049 kms ', '33,895 kms ', '32,893 kms ',
       '61,717 kms ', '82,334 kms ', '87,381 kms ', '2,10,000 kms ',
       '24,200 kms ', '34,642 kms ', '1,54,104 kms ', '61,826 kms ',
       '44,751 kms ', '5,183 kms ', '1,36,082 kms ', '5,817 kms ',
       '99,138 kms ', '27,936 kms ', '39,845 kms ', '15,615 kms ',
       '53,158 kms ', '16,002 kms ', '31,133 kms ', '20,172 kms ',
       '69,653 kms ', '1,32,000 kms ', '84,000 kms ', '1,47,000 kms ',
       '26,023 kms ', '19,805 kms ', '41,365 kms ', '90,001 kms ',
       '45,285 kms ', '42,441 kms ', '24,342 kms ', '14,308 kms ',
       '32,456 kms ', '58,448 kms ', '86,576 kms ', '50,421 kms ',
       '26,035 kms ', '23,499 kms ', '25,636 kms ', '15,023 kms ',
       '250 kms ', '22,584 kms ', '22,394 kms ', '93,563 kms ',
       '10,537 kms ', '81,515 kms ', '24,676 kms ', '11,917 kms ',
       '69,654 kms ', '16,226 kms ', '8,925 kms ', '27,819 kms ',
       '26,543 kms ', '90,088 kms ', '34,258 kms ', '1,01,744 kms ',
       '19,830 kms ', '13,636 kms ', '21,393 kms ', '76,915 kms ',
       '16,300 kms ', '40,106 kms ', '66,461 kms ', '48,182 kms ',
       '41,816 kms ', '73,841 kms ', '19,908 kms ', '26,226 kms ',
       '43,287 kms ', '26,527 kms ', '37,801 kms ', '54,044 kms ',
       '29,632 kms ', '31,000 kms '], dtype=object)
```

In [31]:
```
df['Kms_driven']=df['Kms_driven'].str.replace(',','')
df['Kms_driven']
```

Out[31]:
```
0       35056 kms
1       44000 kms
2       42917 kms
3       16112 kms
4       30988 kms
          ...
1027    65000 kms
1028    60000 kms
1029    31000 kms
1030    31000 kms
1031    31000 kms
Name: Kms_driven, Length: 1024, dtype: object
```

In [32]:
```
df['Kms_driven']=df['Kms_driven'].str.replace('kms','')
df['Kms_driven']
```

Out[32]:
```
0       35056
1       44000
2       42917
3       16112
4       30988
         ...
1027    65000
1028    60000
1029    31000
1030    31000
1031    31000
Name: Kms_driven, Length: 1024, dtype: object
```

In [33]:
```
df['Kms_driven']=pd.to_numeric(df['Kms_driven'])
df['Kms_driven']
```

Out[33]:
```
0       35056
1       44000
2       42917
3       16112
4       30988
        ...
1027    65000
1028    60000
1029    31000
1030    31000
1031    31000
Name: Kms_driven, Length: 1024, dtype: int64
```

In [34]:
```
df['Kms_driven'].max()
```

Out[34]:
```
690000
```

In [35]:
```
df['Kms_driven'].min()
```

Out[35]:
```
0
```

In [36]:
```
df['Owner'].unique()
```

Out[36]:
```
array([nan, ' 2nd Owner', ' 1st Owner', ' 3rd Owner'], dtype=object)
```

In [37]:
```
 df=df.drop(['Owner'],axis=1)
```

In [38]:
```
for i in range(8):
    print("The number of null values in column",i,"are:")
    nullCheck=df.iloc[:,i].isnull().sum()
    print(nullCheck)
```

```
The number of null values in column 0 are:
0
The number of null values in column 1 are:
0
The number of null values in column 2 are:
0
The number of null values in column 3 are:
0
The number of null values in column 4 are:
0
The number of null values in column 5 are:
0
The number of null values in column 6 are:
0
The number of null values in column 7 are:
0
```

In [39]:
```
df['Label'].unique()
```

Out[39]:
```
array(['PLATINUM', 'GOLD'], dtype=object)
```

In [40]:
```
Labels={'PLATINUM':2,'GOLD':1}
df=df.replace(Labels)
df
```

Out[40]:

	Name	Label	Location	Price	Kms_driven	Fuel_type	Year	Company
0	Ford Figo Duratec	2	Bangalore	380000	35056	Petrol	2015	Ford
1	Maruti Suzuki Wagon	2	Bangalore	465000	44000	Petrol	2016	Maruti
2	Hyundai Creta 1.6	2	Bangalore	1350000	42917	Petrol	2018	Hyundai
3	Hyundai Venue	2	Chennai	1019699	16112	Petrol	2019	Hyundai
4	Honda Jazz	2	Pune	713499	30988	Petrol	2017	Honda

...	...	...	...	...	...	...	...	...
1027	Hyundai i10 Magna	1	Bangalore	229000	65000	Petrol	2014	Hyundai
1028	Maruti Suzuki Alto	1	Bangalore	275000	60000	Petrol	2014	Maruti
1029	Jeep Compass Limited	1	Mahasamund	1750000	31000	Diesel	2017	Jeep
1030	Jeep Compass Limited	1	Bilaspur	1750000	31000	Diesel	2017	Jeep
1031	Jeep Compass Limited	1	Jagdalpur	1750000	31000	Diesel	2017	Jeep

1024 rows × 8 columns

In [41]:
```
len(df['Fuel_type'].unique())
```

Out[41]:
```
10
```

In [42]:
```
len(df['Location'].unique())
```

Out[42]:
```
39
```

In [43]:
```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
le.fit_transform(df['Location'])
```

Out[43]:
```
array([ 2,  2,  2, ..., 25,  4, 16])
```

In [44]:
```
dummies1=pd.get_dummies(df['Location'])
dummies1
```

Out[44]:

	Ahmedabad	Anand	Bangalore	Bhubaneswar	Bilaspur	BolpurSantiniketan	Chandigarh	Chennai	Coimbatore	Delhi	...	Nagpur	Nandec
0	0	0	1	0	0	0	0	0	0	0	...	0	
1	0	0	1	0	0	0	0	0	0	0	...	0	
2	0	0	1	0	0	0	0	0	0	0	...	0	
3	0	0	0	0	0	0	0	1	0	0	...	0	
4	0	0	0	0	0	0	0	0	0	0	...	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...
1027	0	0	1	0	0	0	0	0	0	0	...	0	
1028	0	0	1	0	0	0	0	0	0	0	...	0	
1029	0	0	0	0	0	0	0	0	0	0	...	0	
1030	0	0	0	0	1	0	0	0	0	0	...	0	
1031	0	0	0	0	0	0	0	0	0	0	...	0	

1024 rows × 39 columns

In [45]:
```
df=df.drop(['Location'],axis=1)
df
```

Out[45]:

	Name	Label	Price	Kms_driven	Fuel_type	Year	Company
0	Ford Figo Duratec	2	380000	35056	Petrol	2015	Ford
1	Maruti Suzuki Wagon	2	465000	44000	Petrol	2016	Maruti
2	Hyundai Creta 1.6	2	1350000	42917	Petrol	2018	Hyundai
3	Hyundai Venue	2	1019699	16112	Petrol	2019	Hyundai
4	Honda Jazz	2	713499	30988	Petrol	2017	Honda
...	...	...	...	...	...	...	...
1027	Hyundai i10 Magna	1	229000	65000	Petrol	2014	Hyundai
1028	Maruti Suzuki Alto	1	275000	60000	Petrol	2014	Maruti

1029	Jeep Compass Limited	1	1750000	31000	Diesel	2017	Jeep
1030	Jeep Compass Limited	1	1750000	31000	Diesel	2017	Jeep
1031	Jeep Compass Limited	1	1750000	31000	Diesel	2017	Jeep

1024 rows × 7 columns

In [46]:
```
df=pd.concat([df,dummies1],axis=1)
df
```

Out[46]:

	Name	Label	Price	Kms_driven	Fuel_type	Year	Company	Ahmedabad	Anand	Bangalore	...	Nagpur	Nanded	NaviMumbai	Pc
0	Ford Figo Duratec	2	380000	35056	Petrol	2015	Ford	0	0	1	...	0	0	0	
1	Maruti Suzuki Wagon	2	465000	44000	Petrol	2016	Maruti	0	0	1	...	0	0	0	
2	Hyundai Creta 1.6	2	1350000	42917	Petrol	2018	Hyundai	0	0	1	...	0	0	0	
3	Hyundai Venue	2	1019699	16112	Petrol	2019	Hyundai	0	0	0	...	0	0	0	
4	Honda Jazz	2	713499	30988	Petrol	2017	Honda	0	0	0	...	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1027	Hyundai i10 Magna	1	229000	65000	Petrol	2014	Hyundai	0	0	1	...	0	0	0	
1028	Maruti Suzuki Alto	1	275000	60000	Petrol	2014	Maruti	0	0	1	...	0	0	0	
1029	Jeep Compass Limited	1	1750000	31000	Diesel	2017	Jeep	0	0	0	...	0	0	0	
1030	Jeep Compass Limited	1	1750000	31000	Diesel	2017	Jeep	0	0	0	...	0	0	0	
1031	Jeep Compass Limited	1	1750000	31000	Diesel	2017	Jeep	0	0	0	...	0	0	0	

1024 rows × 46 columns

In [47]:
```
df=df.drop(df.columns[45],axis=1)
df
```

Out[47]:

	Name	Label	Price	Kms_driven	Fuel_type	Year	Company	Ahmedabad	Anand	Bangalore	...	Muzaffarnagar	Nagpur	Nanded
0	Ford Figo Duratec	2	380000	35056	Petrol	2015	Ford	0	0	1	...	0	0	0
1	Maruti Suzuki Wagon	2	465000	44000	Petrol	2016	Maruti	0	0	1	...	0	0	0
2	Hyundai Creta 1.6	2	1350000	42917	Petrol	2018	Hyundai	0	0	1	...	0	0	0
3	Hyundai Venue	2	1019699	16112	Petrol	2019	Hyundai	0	0	0	...	0	0	0
4	Honda Jazz	2	713499	30988	Petrol	2017	Honda	0	0	0	...	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1027	Hyundai i10 Magna	1	229000	65000	Petrol	2014	Hyundai	0	0	1	...	0	0	0
1028	Maruti Suzuki Alto	1	275000	60000	Petrol	2014	Maruti	0	0	1	...	0	0	0
1029	Jeep Compass Limited	1	1750000	31000	Diesel	2017	Jeep	0	0	0	...	0	0	0
1030	Jeep Compass Limited	1	1750000	31000	Diesel	2017	Jeep	0	0	0	...	0	0	0

1031	Jeep Compass Limited	1	1750000	31000	Diesel	2017	Jeep	0	0	0	...	0	0	0	

1024 rows × 45 columns

In [48]:
```
dummies2=pd.get_dummies(df['Fuel_type'])
dummies2
```

Out[48]:

	CNG	CNG	Diesel	Diesel	Electric	Hybrid	LPG	Petrol	Petrol	Petrol + CNG
0	0	0	0	0	0	0	0	1	0	0
1	0	0	0	0	0	0	0	1	0	0
2	0	0	0	0	0	0	0	1	0	0
3	0	0	0	0	0	0	0	0	1	0
4	0	0	0	0	0	0	0	0	1	0
...	...	...	...	...	...	...	...	...	...	...
1027	0	0	0	0	0	0	0	0	1	0
1028	0	0	0	0	0	0	0	1	0	0
1029	0	0	0	1	0	0	0	0	0	0
1030	0	0	0	1	0	0	0	0	0	0
1031	0	0	0	1	0	0	0	0	0	0

1024 rows × 10 columns

In [49]:
```
df=df.drop(['Fuel_type'],axis=1)
df
```

Out[49]:

	Name	Label	Price	Kms_driven	Year	Company	Ahmedabad	Anand	Bangalore	Bhubaneswar	...	Muzaffarnagar	Nagpur	Nande
0	Ford Figo Duratec	2	380000	35056	2015	Ford	0	0	1	0	...	0	0	
1	Maruti Suzuki Wagon	2	465000	44000	2016	Maruti	0	0	1	0	...	0	0	
2	Hyundai Creta 1.6	2	1350000	42917	2018	Hyundai	0	0	1	0	...	0	0	
3	Hyundai Venue	2	1019699	16112	2019	Hyundai	0	0	0	0	...	0	0	
4	Honda Jazz	2	713499	30988	2017	Honda	0	0	0	0	...	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1027	Hyundai i10 Magna	1	229000	65000	2014	Hyundai	0	0	1	0	...	0	0	
1028	Maruti Suzuki Alto	1	275000	60000	2014	Maruti	0	0	1	0	...	0	0	
1029	Jeep Compass Limited	1	1750000	31000	2017	Jeep	0	0	0	0	...	0	0	
1030	Jeep Compass Limited	1	1750000	31000	2017	Jeep	0	0	0	0	...	0	0	
1031	Jeep Compass Limited	1	1750000	31000	2017	Jeep	0	0	0	0	...	0	0	

1024 rows × 44 columns

In [50]:
```
df=pd.concat([df,dummies2],axis=1)
df
```

Out[50]:

	Name	Label	Price	Kms_driven	Year	Company	Ahmedabad	Anand	Bangalore	Bhubaneswar	...	CNG	CNG	Diesel	Diesel	El

0	Ford Figo Duratec	2	380000	35056	2015	Ford	0	0	1	0	...	0	0	0	0
1	Maruti Suzuki Wagon	2	465000	44000	2016	Maruti	0	0	1	0	...	0	0	0	0
2	Hyundai Creta 1.6	2	1350000	42917	2018	Hyundai	0	0	1	0	...	0	0	0	0
3	Hyundai Venue	2	1019699	16112	2019	Hyundai	0	0	0	0	...	0	0	0	0
4	Honda Jazz	2	713499	30988	2017	Honda	0	0	0	0	...	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1027	Hyundai i10 Magna	1	229000	65000	2014	Hyundai	0	0	1	0	...	0	0	0	0
1028	Maruti Suzuki Alto	1	275000	60000	2014	Maruti	0	0	1	0	...	0	0	0	0
1029	Jeep Compass Limited	1	1750000	31000	2017	Jeep	0	0	0	0	...	0	0	0	1
1030	Jeep Compass Limited	1	1750000	31000	2017	Jeep	0	0	0	0	...	0	0	0	1
1031	Jeep Compass Limited	1	1750000	31000	2017	Jeep	0	0	0	0	...	0	0	0	1

1024 rows × 54 columns

In [51]:
```
df=df.drop(df.columns[53],axis=1)
df
```

Out[51]:

	Name	Label	Price	Kms_driven	Year	Company	Ahmedabad	Anand	Bangalore	Bhubaneswar	...	Udaipur	CNG	CNG	Diesel
0	Ford Figo Duratec	2	380000	35056	2015	Ford	0	0	1	0	...	0	0	0	0
1	Maruti Suzuki Wagon	2	465000	44000	2016	Maruti	0	0	1	0	...	0	0	0	0
2	Hyundai Creta 1.6	2	1350000	42917	2018	Hyundai	0	0	1	0	...	0	0	0	0
3	Hyundai Venue	2	1019699	16112	2019	Hyundai	0	0	0	0	...	0	0	0	0
4	Honda Jazz	2	713499	30988	2017	Honda	0	0	0	0	...	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1027	Hyundai i10 Magna	1	229000	65000	2014	Hyundai	0	0	1	0	...	0	0	0	0
1028	Maruti Suzuki Alto	1	275000	60000	2014	Maruti	0	0	1	0	...	0	0	0	0
1029	Jeep Compass Limited	1	1750000	31000	2017	Jeep	0	0	0	0	...	0	0	0	0
1030	Jeep Compass Limited	1	1750000	31000	2017	Jeep	0	0	0	0	...	0	0	0	0
1031	Jeep Compass Limited	1	1750000	31000	2017	Jeep	0	0	0	0	...	0	0	0	0

1024 rows × 53 columns

In [52]:
```
df=df.reset_index(drop=True)
df
```

Out[52]:

	Name	Label	Price	Kms_driven	Year	Company	Ahmedabad	Anand	Bangalore	Bhubaneswar	...	Udaipur	CNG	CNG	Diesel
	Ford														

0	Figo Duratec	2	380000	35056	2015	Ford	0	0	1	0	...	0	0	0	0
1	Maruti Suzuki Wagon	2	465000	44000	2016	Maruti	0	0	1	0	...	0	0	0	0
2	Hyundai Creta 1.6	2	1350000	42917	2018	Hyundai	0	0	1	0	...	0	0	0	0
3	Hyundai Venue	2	1019699	16112	2019	Hyundai	0	0	0	0	...	0	0	0	0
4	Honda Jazz	2	713499	30988	2017	Honda	0	0	0	0	...	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1019	Hyundai i10 Magna	1	229000	65000	2014	Hyundai	0	0	1	0	...	0	0	0	0
1020	Maruti Suzuki Alto	1	275000	60000	2014	Maruti	0	0	1	0	...	0	0	0	0
1021	Jeep Compass Limited	1	1750000	31000	2017	Jeep	0	0	0	0	...	0	0	0	0
1022	Jeep Compass Limited	1	1750000	31000	2017	Jeep	0	0	0	0	...	0	0	0	0
1023	Jeep Compass Limited	1	1750000	31000	2017	Jeep	0	0	0	0	...	0	0	0	0

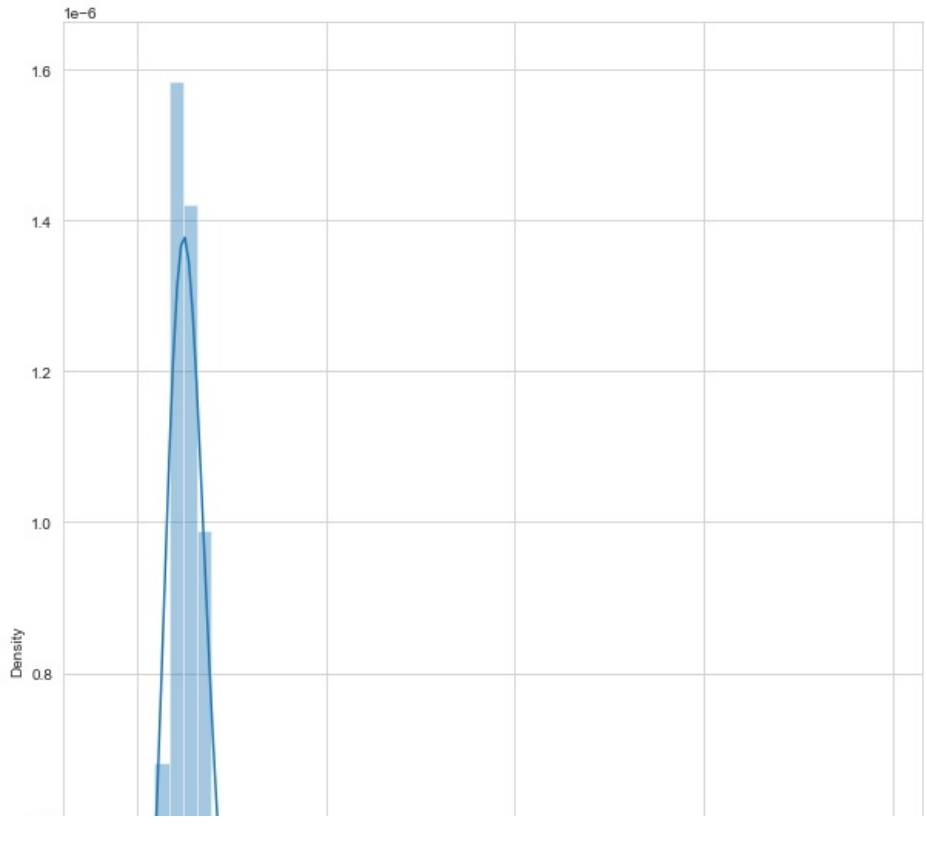
1024 rows × 53 columns

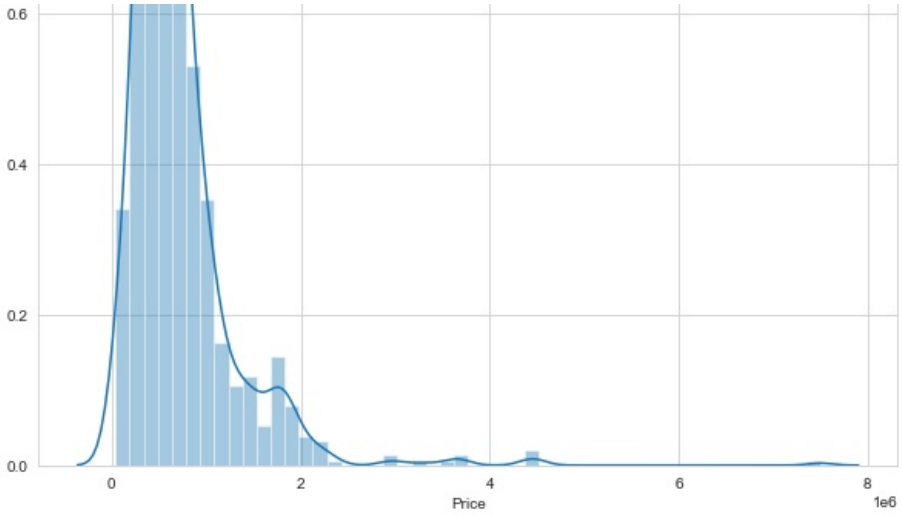
EXPLORATIVE DATA ANALYSIS

In [53]:
```
sns.set_style('whitegrid')
```

In [54]:
```
plt.figure(figsize=(10,15))
sns.distplot(df['Price'])
plt.show()
```

```
C:\Users\revan\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```





In [55]:
```
df.corr()
```

Out[55]:

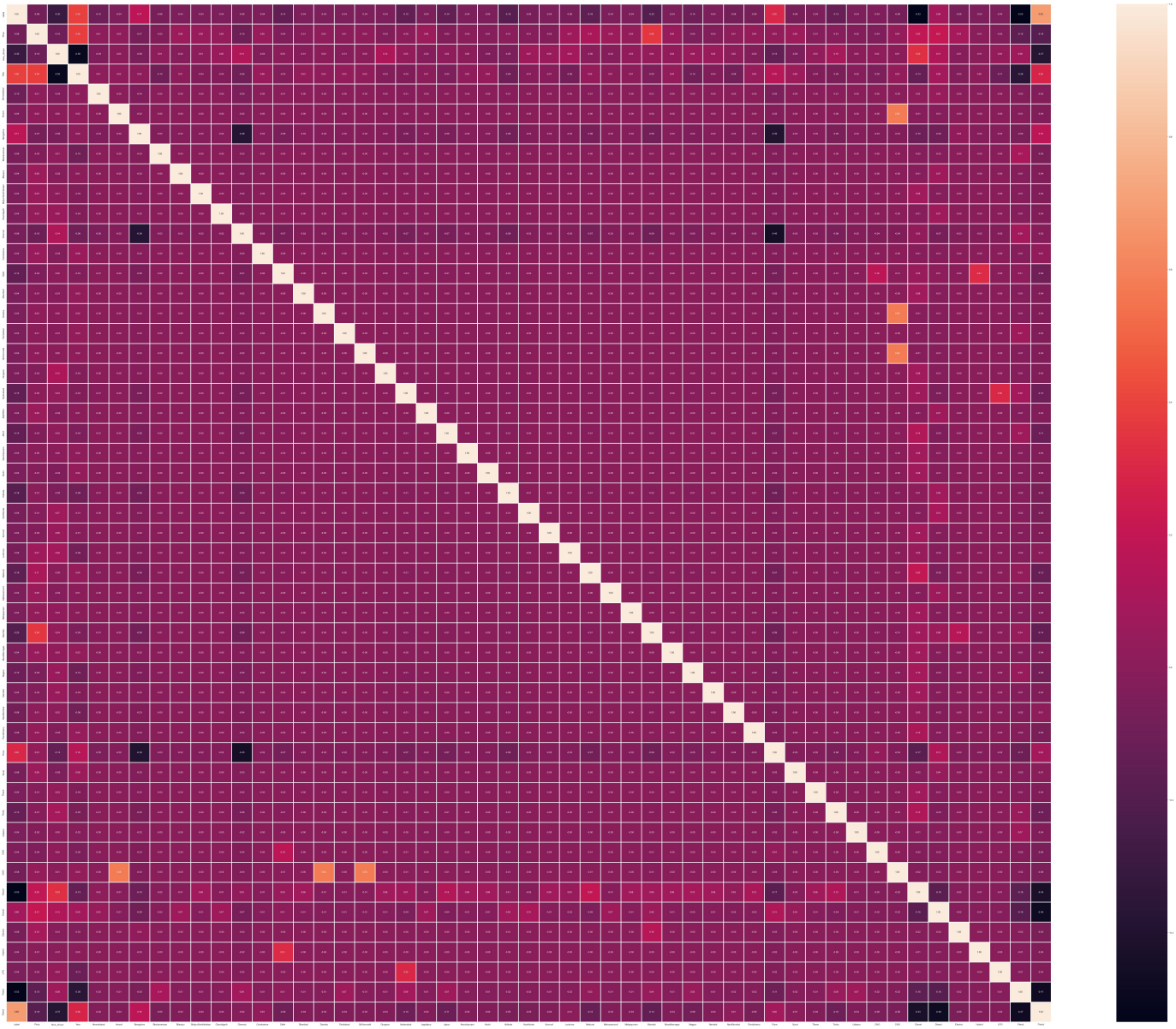
	Label	Price	Kms_driven	Year	Ahmedabad	Anand	Bangalore	Bhubaneswar	Bilaspur	BolpurSantiniketan
Label	1.000000	-0.082353	-0.297303	0.387619	-0.100014	-0.044640	0.173986	-0.063162	-0.044640	-0.044640
Price	-0.082353	1.000000	-0.097850	0.381180	0.012315	0.022350	-0.070198	-0.034907	0.061922	0.047374
Kms_driven	-0.297303	-0.097850	1.000000	-0.501249	-0.041741	0.003085	-0.058897	0.014809	-0.015780	0.011201
Year	0.387619	0.381180	-0.501249	1.000000	0.012028	0.020136	0.016311	-0.095572	0.010906	-0.035242
Ahmedabad	-0.100014	0.012315	-0.041741	0.012028	1.000000	-0.002190	-0.040442	-0.003099	-0.002190	-0.002190
Anand	-0.044640	0.022350	0.003085	0.020136	-0.002190	1.000000	-0.018051	-0.001383	-0.000978	-0.000978
Bangalore	0.173986	-0.070198	-0.058897	0.016311	-0.040442	-0.018051	1.000000	-0.025540	-0.018051	-0.018051
Bhubaneswar	-0.063162	-0.034907	0.014809	-0.095572	-0.003099	-0.001383	-0.025540	1.000000	-0.001383	-0.001383
Bilaspur	-0.044640	0.061922	-0.015780	0.010906	-0.002190	-0.000978	-0.018051	-0.001383	1.000000	-0.000978
BolpurSantiniketan	-0.044640	0.047374	0.011201	-0.035242	-0.002190	-0.000978	-0.018051	-0.001383	-0.000978	1.000000
Chandigarh	-0.044640	0.031719	0.054758	-0.035242	-0.002190	-0.000978	-0.018051	-0.001383	-0.000978	-0.000978
Chennai	-0.064661	-0.103024	0.136429	-0.086014	-0.046370	-0.020697	-0.382188	-0.029284	-0.020697	-0.020697
Coimbatore	-0.044640	0.050283	-0.030544	0.057055	-0.002190	-0.000978	-0.018051	-0.001383	-0.000978	-0.000978
Delhi	-0.141790	-0.041099	0.000236	-0.026923	-0.006956	-0.003105	-0.057335	-0.004393	-0.003105	-0.003105
Dhanbad	-0.044640	-0.012276	-0.010039	0.029366	-0.002190	-0.000978	-0.018051	-0.001383	-0.000978	-0.000978
Dwarka	-0.044640	0.022350	0.003085	0.020136	-0.002190	-0.000978	-0.018051	-0.001383	-0.000978	-0.000978
Faridabad	-0.044640	0.007801	-0.011679	0.029366	-0.002190	-0.000978	-0.018051	-0.001383	-0.000978	-0.000978
GirSomnath	-0.044640	0.022350	0.003085	0.020136	-0.002190	-0.000978	-0.018051	-0.001383	-0.000978	-0.000978
Gurgaon	-0.044640	-0.025952	0.122836	-0.026013	-0.002190	-0.000978	-0.018051	-0.001383	-0.000978	-0.000978
Hyderabad	-0.148784	-0.000377	0.032674	-0.033564	-0.007299	-0.003258	-0.060163	-0.004610	-0.003258	-0.003258
Jagdalpur	-0.044640	0.061922	-0.015780	0.010906	-0.002190	-0.000978	-0.018051	-0.001383	-0.000978	-0.000978
Jaipur	-0.141790	-0.031320	0.020241	-0.044513	-0.006956	-0.003105	-0.057335	-0.004393	-0.003105	-0.003105
Kanchipuram	-0.044640	-0.002092	0.003085	0.020136	-0.002190	-0.000978	-0.018051	-0.001383	-0.000978	-0.000978
Kochi	-0.044640	-0.014313	-0.041002	0.038595	-0.002190	-0.000978	-0.018051	-0.001383	-0.000978	-0.000978
Kolkata	-0.180916	-0.006830	-0.046844	-0.086625	-0.009631	-0.004299	-0.079384	-0.006083	-0.004299	-0.004299
Kozhikode	-0.063162	-0.013252	0.065872	-0.010687	-0.003099	-0.001383	-0.025540	-0.001957	-0.001383	-0.001383
Kurnool	-0.044640	-0.020714	0.057219	-0.007553	-0.002190	-0.000978	-0.018051	-0.001383	-0.000978	-0.000978
Lucknow	-0.063162	0.067194	0.086761	-0.075983	-0.003099	-0.001383	-0.025540	-0.001957	-0.001383	-0.001383
Madurai	-0.155476	0.112805	-0.004493	0.040664	-0.007628	-0.003405	-0.062869	-0.004817	-0.003405	-0.003405
Mahasamund	-0.044640	0.061922	-0.015780	0.010906	-0.002190	-0.000978	-0.018051	-0.001383	-0.000978	-0.000978
Malappuram	-0.044640	0.031719	0.026050	0.010906	-0.002190	-0.000978	-0.018051	-0.001383	-0.000978	-0.000978
Mumbai	-0.196317	0.357772	0.035018	-0.046035	-0.009631	-0.004299	-0.079384	-0.006083	-0.004299	-0.004299
Muzaffarnagar	-0.044640	0.047374	0.028511	0.029366	-0.002190	-0.000978	-0.018051	-0.001383	-0.000978	-0.000978
Nagpur	-0.100014	-0.047556	0.056160	-0.095502	-0.004907	-0.002190	-0.040442	-0.003099	-0.002190	-0.002190
Nanded	-0.044640	-0.028279	0.053117	-0.035242	-0.002190	-0.000978	-0.018051	-0.001383	-0.000978	-0.000978
NaviMumbai	-0.077395	0.014837	0.016487	-0.082437	-0.003797	-0.001695	-0.031296	-0.002398	-0.001695	-0.001695
Pondicherry	-0.063162	0.036975	-0.025229	0.041550	-0.003099	-0.001383	-0.025540	-0.001957	-0.001383	-0.001383

Pune	0.296523	0.029129	-0.136752	0.158616	-0.047549	-0.021223	-0.391906	-0.030028	-0.021223	-0.021223
Surat	-0.063162	0.062995	-0.045770	0.061138	-0.003099	-0.001383	-0.025540	-0.001957	-0.001383	-0.001383
Thane	-0.044640	-0.013731	0.030562	-0.035242	-0.002190	-0.000978	-0.018051	-0.001383	-0.000978	-0.000978
Trichy	-0.126696	-0.009653	0.095177	-0.054182	-0.006216	-0.002774	-0.051232	-0.003925	-0.002774	-0.002774
Udaipur	-0.044640	-0.022751	0.004725	-0.016783	-0.002190	-0.000978	-0.018051	-0.001383	-0.000978	-0.000978
CNG	-0.038941	-0.039780	0.020990	-0.045099	-0.003797	-0.001695	-0.031296	-0.002398	-0.001695	-0.001695
CNG	-0.077395	0.038749	0.005348	0.034911	-0.003797	0.576786	-0.031296	-0.002398	-0.001695	-0.001695
Diesel	-0.534412	0.195467	0.329016	-0.130506	0.015114	-0.011975	-0.099593	-0.016943	-0.011975	0.081630
Diesel	0.092042	0.212030	0.127492	0.094730	0.047645	-0.013355	-0.090527	-0.018896	0.073197	-0.013355
Electric	-0.063162	0.096672	-0.009562	-0.030276	-0.003099	-0.001383	0.025540	-0.001957	-0.001383	-0.001383
Hybrid	-0.044640	-0.005583	-0.005036	0.001676	-0.002190	-0.000978	-0.018051	-0.001383	-0.000978	-0.000978
LPG	-0.044640	-0.025952	0.038354	-0.109080	-0.002190	-0.000978	-0.018051	-0.001383	-0.000978	-0.000978
Petrol	-0.525964	-0.152371	0.075966	-0.355474	-0.029246	-0.013053	-0.050747	0.105956	-0.013053	-0.013053
Petrol	0.690089	-0.181379	-0.372923	0.280770	-0.022230	-0.035102	0.178226	-0.049666	-0.035102	-0.035102

51 rows × 51 columns

In [56]:

```
plt.figure(figsize=(100,80))
sns.heatmap(df.corr(),linewidth=3,annot=True,fmt='.2f')
plt.show()
```



In [57]:

```
label1=df[df['Label']==1]
label1
```

Out[57]:

	Name	Label	Price	Kms_driven	Year	Company	Ahmedabad	Anand	Bangalore	Bhubaneswar	...	Udaipur	CNG	CNG	Diesel
16	Toyota Innova 2.5	1	1025000	131000	2012	Toyota	0	0	1	0	...	0	0	0	1
17	Hyundai Elite i20	1	735000	26000	2016	Hyundai	0	0	1	0	...	0	0	0	0
18	Hyundai Creta 1.6	1	600000	58460	2019	Hyundai	0	0	0	0	...	0	0	0	0
19	Kia Seltos	1	800000	48000	2020	Kia	0	0	0	0	...	0	0	0	0
20	Maruti Suzuki Ciaz	1	700000	26600	2017	Maruti	0	0	0	0	...	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1019	Hyundai i10 Magna	1	229000	65000	2014	Hyundai	0	0	1	0	...	0	0	0	0
1020	Maruti Suzuki Alto	1	275000	60000	2014	Maruti	0	0	1	0	...	0	0	0	0
1021	Jeep Compass Limited	1	1750000	31000	2017	Jeep	0	0	0	0	...	0	0	0	0
1022	Jeep Compass Limited	1	1750000	31000	2017	Jeep	0	0	0	0	...	0	0	0	0
1023	Jeep Compass Limited	1	1750000	31000	2017	Jeep	0	0	0	0	...	0	0	0	0

337 rows × 53 columns

In [58]:
```
label2=df[df['Label']==2]
label2
```

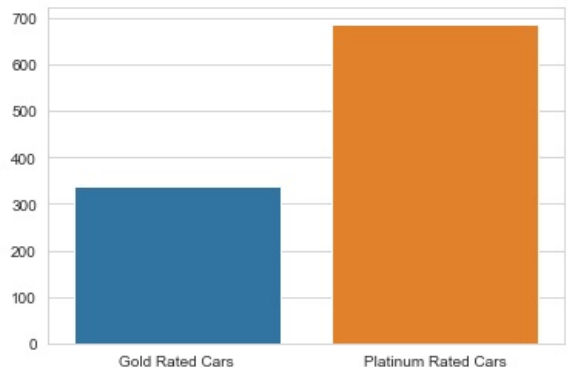
Out[58]:

	Name	Label	Price	Kms_driven	Year	Company	Ahmedabad	Anand	Bangalore	Bhubaneswar	...	Udaipur	CNG	CNG	Diesel	D
0	Ford Figo Duratec	2	380000	35056	2015	Ford	0	0	1	0	...	0	0	0	0	
1	Maruti Suzuki Wagon	2	465000	44000	2016	Maruti	0	0	1	0	...	0	0	0	0	
2	Hyundai Creta 1.6	2	1350000	42917	2018	Hyundai	0	0	1	0	...	0	0	0	0	
3	Hyundai Venue	2	1019699	16112	2019	Hyundai	0	0	0	0	...	0	0	0	0	
4	Honda Jazz	2	713499	30988	2017	Honda	0	0	0	0	...	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1011	Maruti Suzuki Baleno	2	681399	43287	2016	Maruti	0	0	0	0	...	0	0	0	0	
1012	Datsun Redi GO	2	415699	26527	2018	Datsun	0	0	1	0	...	0	0	0	0	
1013	Maruti Suzuki Alto	2	300199	37801	2017	Maruti	0	0	1	0	...	0	0	0	0	
1014	Maruti Suzuki Vitara	2	856699	54044	2017	Maruti	0	0	0	0	...	0	0	0	0	
1015	Ford Figo Aspire	2	481399	29632	2017	Ford	0	0	0	0	...	0	0	0	0	

687 rows × 53 columns

In [59]:
```
sns.barplot(x=['Gold Rated Cars','Platinum Rated Cars'],y=[len(label1),len(label2)])
```

Out[59]:
```
<AxesSubplot:>
```



In [60]:
```
df2=df.groupby(['Company']).count()
df2
```

Out[60]:

Company	Name	Label	Price	Kms_driven	Year	Ahmedabad	Anand	Bangalore	Bhubaneswar	Bilaspur	...	Udaipur	CNG	CNG	Diesel
Audi	5	5	5	5	5	5	5	5	5	5	...	5	5	5	5
BMW	7	7	7	7	7	7	7	7	7	7	...	7	7	7	7
Chevrolet	11	11	11	11	11	11	11	11	11	11	...	11	11	11	11
Datsun	4	4	4	4	4	4	4	4	4	4	...	4	4	4	4
Fiat	3	3	3	3	3	3	3	3	3	3	...	3	3	3	3
Ford	51	51	51	51	51	51	51	51	51	51	...	51	51	51	51
Honda	78	78	78	78	78	78	78	78	78	78	...	78	78	78	78
Hyundai	226	226	226	226	226	226	226	226	226	226	...	226	226	226	226
Jaguar	2	2	2	2	2	2	2	2	2	2	...	2	2	2	2
Jeep	10	10	10	10	10	10	10	10	10	10	...	10	10	10	10
Kia	10	10	10	10	10	10	10	10	10	10	...	10	10	10	10
Land	1	1	1	1	1	1	1	1	1	1	...	1	1	1	1
MG	17	17	17	17	17	17	17	17	17	17	...	17	17	17	17
Mahindra	41	41	41	41	41	41	41	41	41	41	...	41	41	41	41
Maruti	382	382	382	382	382	382	382	382	382	382	...	382	382	382	382
Mercedes	4	4	4	4	4	4	4	4	4	4	...	4	4	4	4
Nissan	5	5	5	5	5	5	5	5	5	5	...	5	5	5	5
Porsche	1	1	1	1	1	1	1	1	1	1	...	1	1	1	1
Renault	43	43	43	43	43	43	43	43	43	43	...	43	43	43	43
Skoda	6	6	6	6	6	6	6	6	6	6	...	6	6	6	6
Ssangyong	1	1	1	1	1	1	1	1	1	1	...	1	1	1	1
Tata	44	44	44	44	44	44	44	44	44	44	...	44	44	44	44
Toyota	41	41	41	41	41	41	41	41	41	41	...	41	41	41	41
Volkswagen	29	29	29	29	29	29	29	29	29	29	...	29	29	29	29
Volvo	2	2	2	2	2	2	2	2	2	2	...	2	2	2	2

25 rows × 52 columns

In [61]:
```
df2.sort_values(by=['Name'],ascending=False)
```

Out[61]:

Company	Name	Label	Price	Kms_driven	Year	Ahmedabad	Anand	Bangalore	Bhubaneswar	Bilaspur	...	Udaipur	CNG	CNG	Diesel
Maruti	382	382	382	382	382	382	382	382	382	382	...	382	382	382	382
Hyundai	226	226	226	226	226	226	226	226	226	226	...	226	226	226	226
Honda	78	78	78	78	78	78	78	78	78	78	...	78	78	78	78
Ford	51	51	51	51	51	51	51	51	51	51	...	51	51	51	51
Tata	44	44	44	44	44	44	44	44	44	44	...	44	44	44	44
Renault	43	43	43	43	43	43	43	43	43	43	...	43	43	43	43

Toyota	41	41	41	41	41	41	41	41	41	41	...	41	41	41	41
Mahindra	41	41	41	41	41	41	41	41	41	41	...	41	41	41	41
Volkswagen	29	29	29	29	29	29	29	29	29	29	...	29	29	29	29
MG	17	17	17	17	17	17	17	17	17	17	...	17	17	17	17
Chevrolet	11	11	11	11	11	11	11	11	11	11	...	11	11	11	11
Jeep	10	10	10	10	10	10	10	10	10	10	...	10	10	10	10
Kia	10	10	10	10	10	10	10	10	10	10	...	10	10	10	10
BMW	7	7	7	7	7	7	7	7	7	7	...	7	7	7	7
Skoda	6	6	6	6	6	6	6	6	6	6	...	6	6	6	6
Nissan	5	5	5	5	5	5	5	5	5	5	...	5	5	5	5
Audi	5	5	5	5	5	5	5	5	5	5	...	5	5	5	5
Mercedes	4	4	4	4	4	4	4	4	4	4	...	4	4	4	4
Datsun	4	4	4	4	4	4	4	4	4	4	...	4	4	4	4
Fiat	3	3	3	3	3	3	3	3	3	3	...	3	3	3	3
Jaguar	2	2	2	2	2	2	2	2	2	2	...	2	2	2	2
Volvo	2	2	2	2	2	2	2	2	2	2	...	2	2	2	2
Porsche	1	1	1	1	1	1	1	1	1	1	...	1	1	1	1
Land	1	1	1	1	1	1	1	1	1	1	...	1	1	1	1
Ssangyong	1	1	1	1	1	1	1	1	1	1	...	1	1	1	1

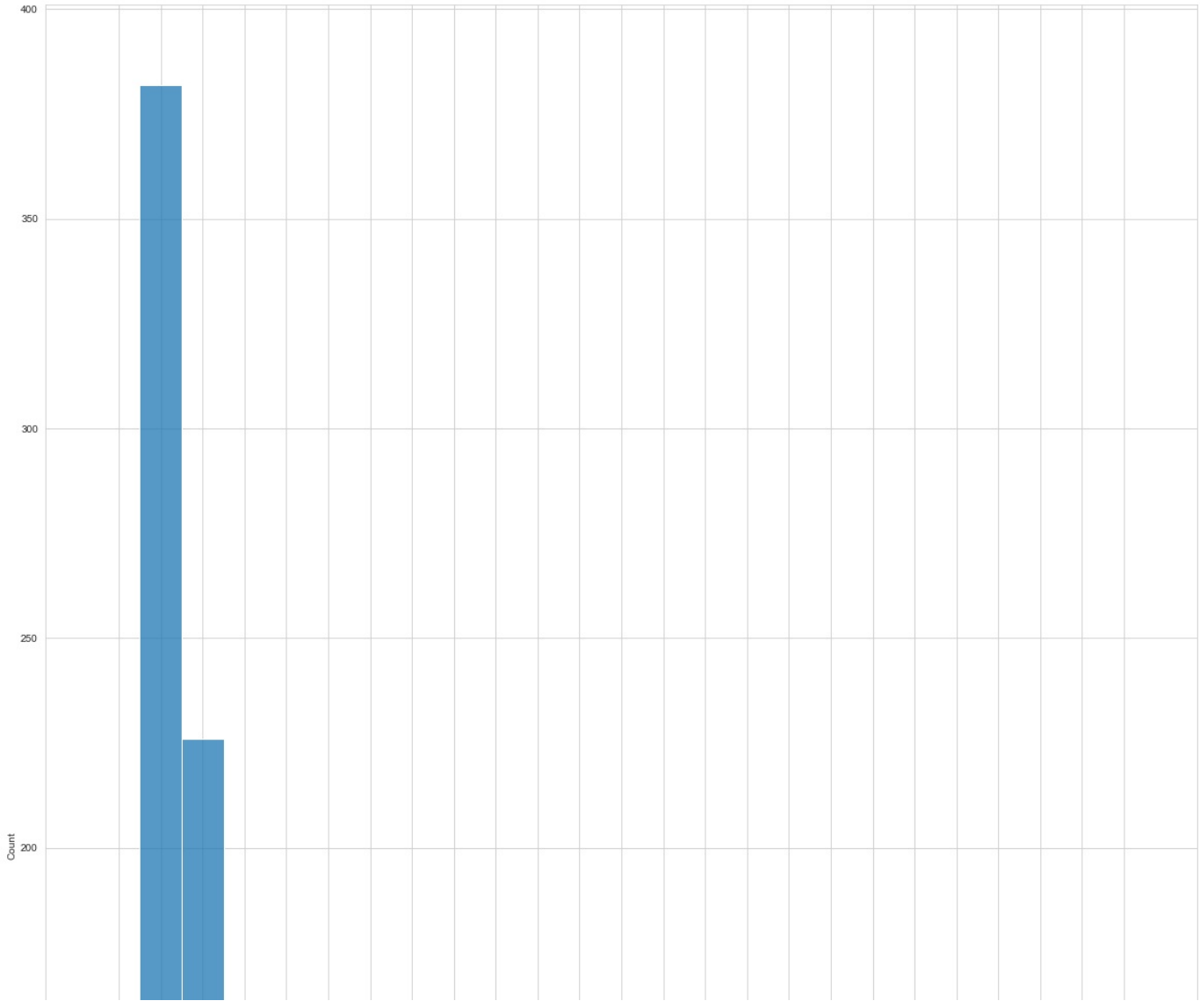
25 rows × 52 columns

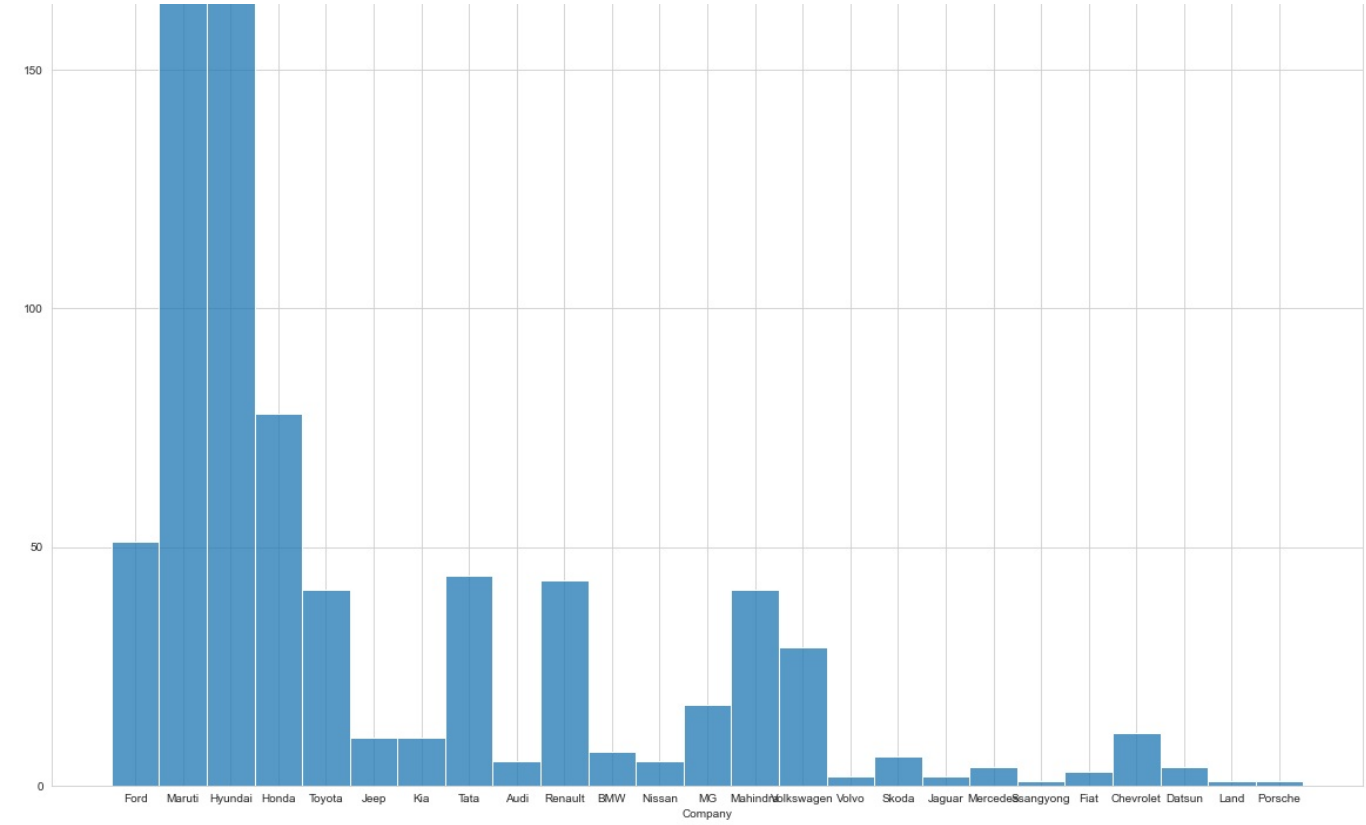
In [62]:

```
plt.figure(figsize=(20,30))
#plt.xticks(rotation=90)
sns.histplot(x=df['Company'])
```

Out[62]:

```
<AxesSubplot:xlabel='Company', ylabel='Count'>
```

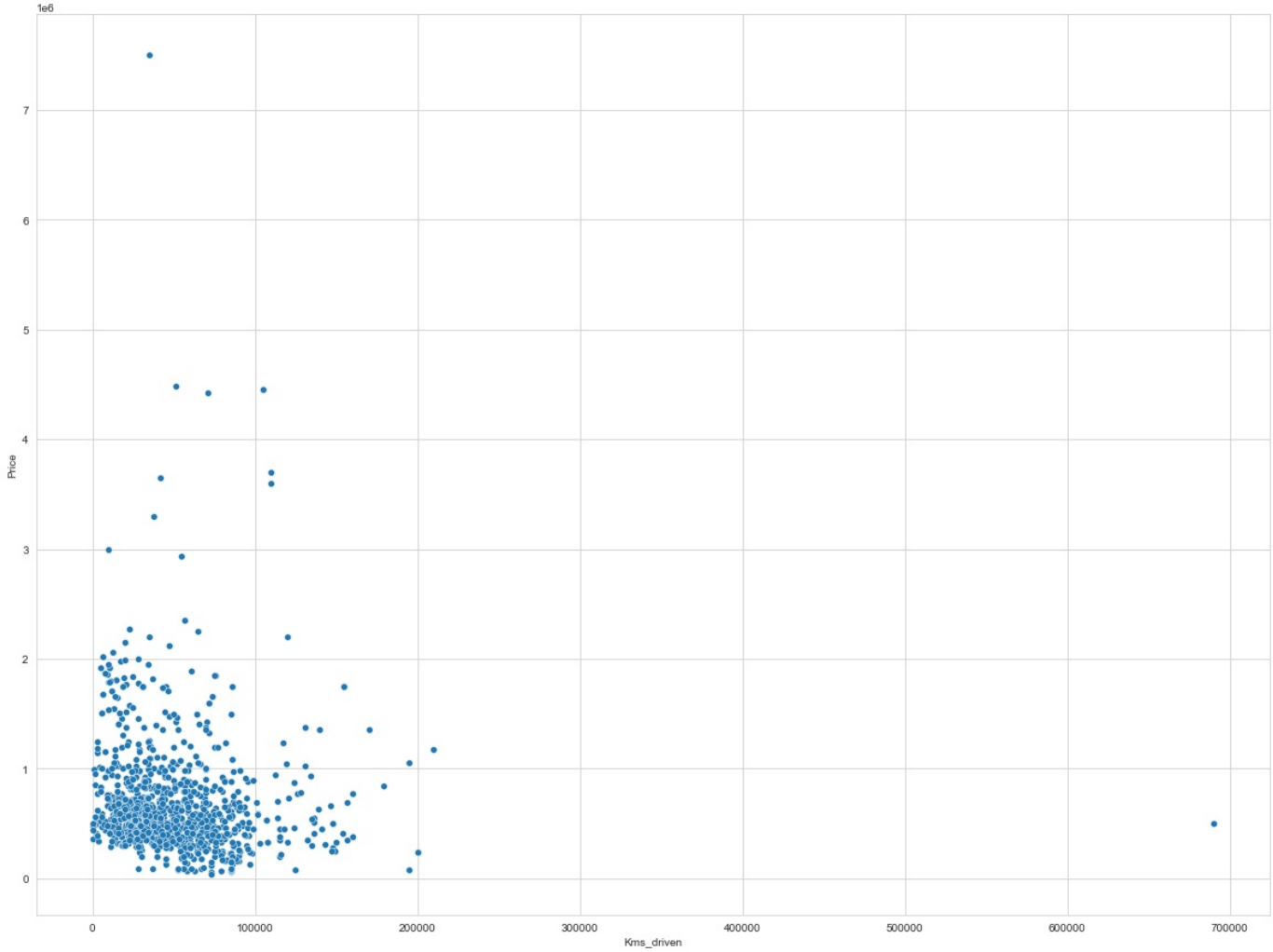




In [63]:

```
plt.figure(figsize=(20,15))
sns.scatterplot(x=df['Kms_driven'],y=df['Price'])
```

Out[63]: <AxesSubplot:xlabel='Kms_driven', ylabel='Price'>



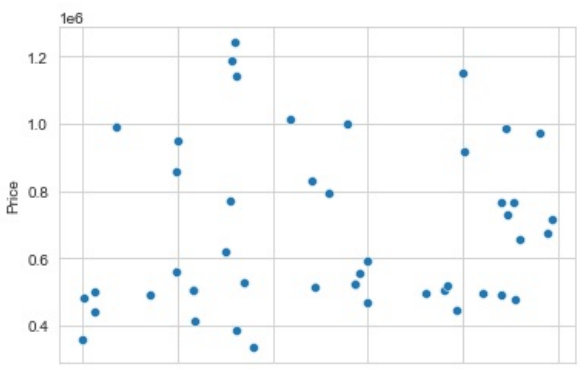
In [64]:

In [64]:
```
lessKms=df[df['Kms_driven']<10000]
lessKms=lessKms['Kms_driven']
lessKms
```

Out[64]:
```
28     3243
29     9616
34     3108
37     2359
44     6000
74     9783
76     3399
81     6110
108    5325
113    9826
124    9811
132    2311
133    2991
136       0
144    2311
158    9085
166    3600
180    6760
181    8870
194    9200
206    7877
244    4365
249    8924
300    8825
301    5743
310    5743
316    1954
344    8030
349    6000
359    8825
361    5580
408    7597
419    1405
421    8000
478    8409
485    1956
505    4889
513    3230
532    3195
612    5600
615    5600
685     700
686      25
687     258
702    3139
716    7679
719    7211
748    6760
768    9874
806    8000
824    4832
850    2000
894    9095
932    5183
934    5817
968     250
985    8925
Name: Kms_driven, dtype: int64
```

In [65]:
```
lessPrice=df[df['Price']<=1500000]
lessPrice=lessPrice['Price']
lessPrice
sns.scatterplot(x=lessKms,y=lessPrice)
```

Out[65]:
```
<AxesSubplot:xlabel='Kms_driven', ylabel='Price'>
```

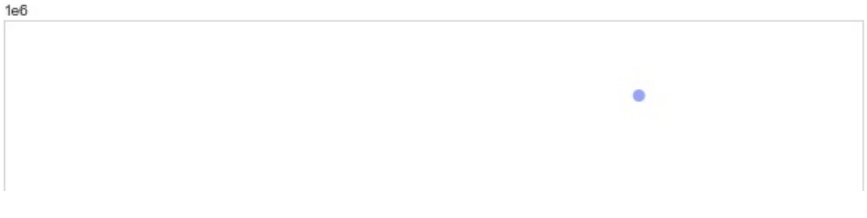


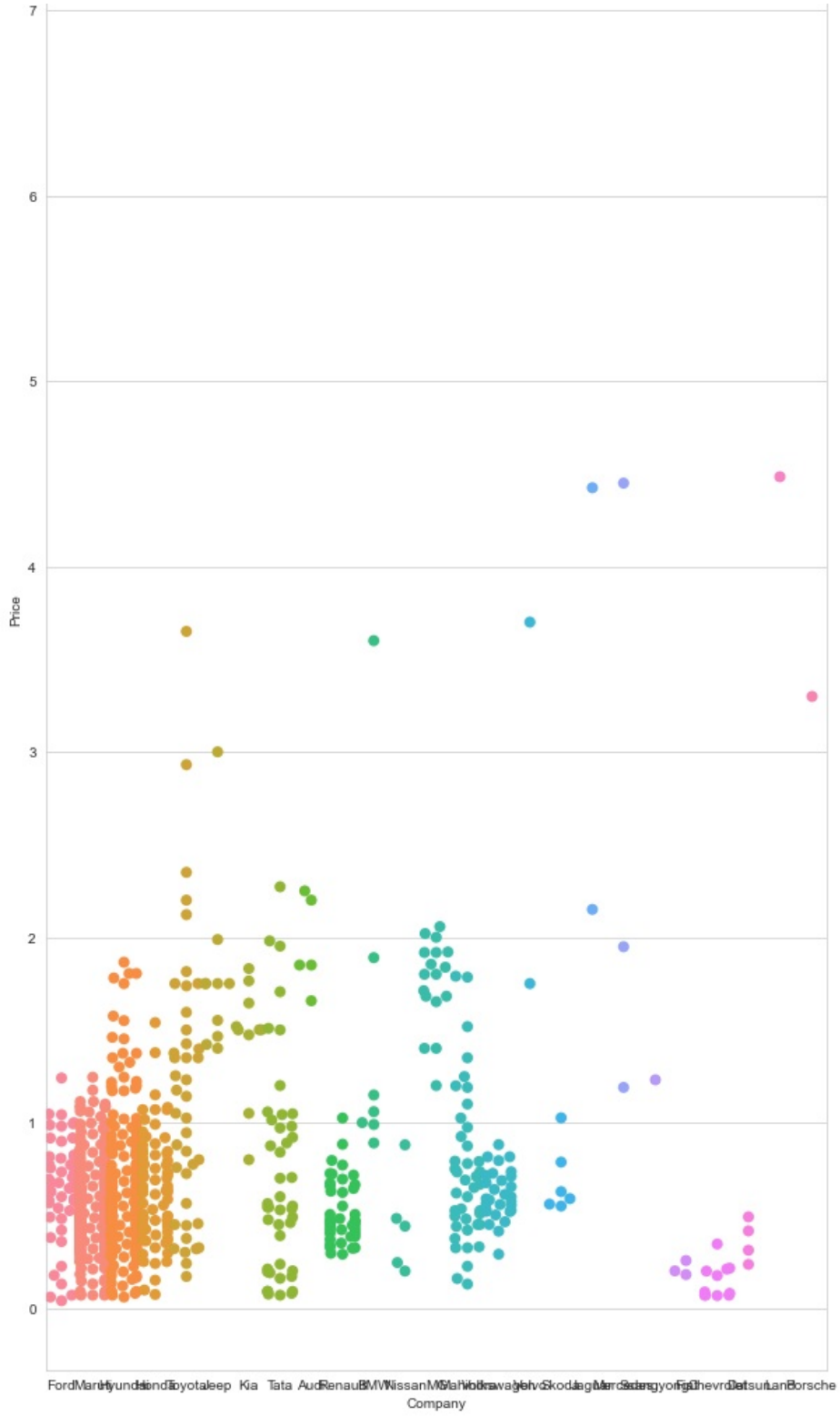
In [66]:

```
plt.figure(figsize=(10,20))
sns.swarmplot(x=df['Company'],y=df['Price'],size=8)
plt.xticks(rotaion=90)
plt.show()
```

```
C:\Users\revan\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 39.2% of the points cannot b
e placed; you may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\revan\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 89.3% of the points cannot b
e placed; you may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\revan\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 77.0% of the points cannot b
e placed; you may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\revan\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 52.6% of the points cannot b
e placed; you may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\revan\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 9.8% of the points cannot be
placed; you may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\revan\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 10.0% of the points cannot b
e placed; you may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\revan\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 20.0% of the points cannot b
e placed; you may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\revan\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 29.5% of the points cannot b
e placed; you may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\revan\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 51.2% of the points cannot b
e placed; you may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\revan\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 5.9% of the points cannot be
placed; you may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\revan\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 24.4% of the points cannot b
e placed; you may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\revan\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 44.8% of the points cannot b
e placed; you may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\revan\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 36.4% of the points cannot b
e placed; you may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_25604/3119442742.py in <module>
      1 plt.figure(figsize=(10,20))
      2 sns.swarmplot(x=df['Company'],y=df['Price'],size=8)
----> 3 plt.xticks(rotaion=90)
      4 plt.show()

~\anaconda3\lib\site-packages\matplotlib\pyplot.py in xticks(ticks, labels, **kwargs)
   1814         labels = ax.set_xticklabels(labels, **kwargs)
   1815     for l in labels:
-> 1816         l.update(kwargs)
   1817
   1818     return locs, labels

~\anaconda3\lib\site-packages\matplotlib\text.py in update(self, kwargs)
    182         # Update bbox last, as it depends on font properties.
    183         bbox = kwargs.pop("bbox", sentinel)
--> 184         super().update(kwargs)
    185         if bbox is not sentinel:
    186             self.set_bbox(bbox)

~\anaconda3\lib\site-packages\matplotlib\artist.py in update(self, props)
   1060                     func = getattr(self, f"set_{k}", None)
   1061                     if not callable(func):
-> 1062                         raise AttributeError(f"{type(self).__name__!r} object "
   1063                                              f"has no property {k!r}")
   1064                     ret.append(func(v))

AttributeError: 'Text' object has no property 'rotaion'
```





In [67]:

```
df3=df.sort_values(by='Price',ascending=False)
df3
```

Out[67]:

	Name	Label	Price	Kms_driven	Year	Company	Ahmedabad	Anand	Bangalore	Bhubaneswar	...	Udaipur	CNG	CNG	Diesel	D
712	Mercedes Benz GLE	1	7500000	35000	2019	Mercedes	0	0	0	0	...	0	0	0	0	
921	Land Rover Range	1	4490000	51000	2017	Land	0	0	0	0	...	0	0	0	1	
143	Mercedes Benz GL	1	4450000	105000	2016	Mercedes	0	0	0	0	...	0	0	0	1	
142	Jaguar XJ L	1	4425000	71000	2016	Jaguar	0	0	0	0	...	0	0	0	1	
66	Volvo XC90 D5	1	3700000	110000	2016	Volvo	0	0	0	0	...	0	0	0	1	

	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
138	Maruti Suzuki 800	1	69000	85000	2000	Maruti	0	0	0	0	...	0	0	0	0
401	Chevrolet Aveo	1	67000	63000	2007	Chevrolet	0	0	0	0	...	0	0	0	0
402	Ford Ikon	1	59000	73000	2009	Ford	0	0	0	0	...	0	0	0	0
232	Hyundai Santro	1	59000	85000	2001	Hyundai	0	0	0	0	...	0	0	0	0
992	Ford Ikon	1	39000	73000	2007	Ford	0	0	0	0	...	0	0	0	0

1024 rows × 53 columns

In [68]:
```
a=df3.iloc[0:10]
a
```

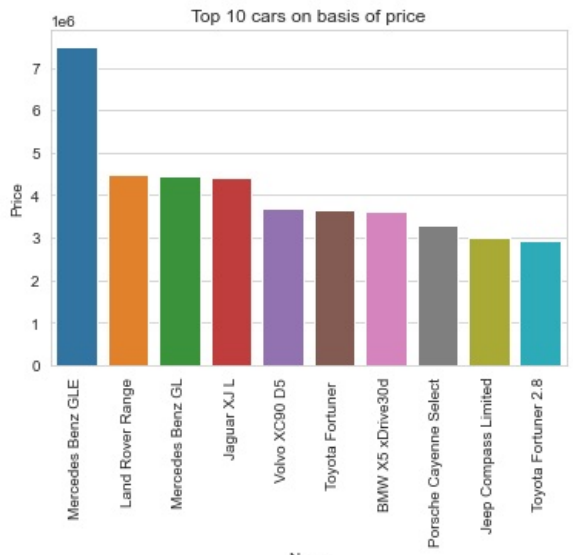
Out[68]:

	Name	Label	Price	Kms_driven	Year	Company	Ahmedabad	Anand	Bangalore	Bhubaneswar	...	Udaipur	CNG	CNG	Diesel	D
712	Mercedes Benz GLE	1	7500000	35000	2019	Mercedes	0	0	0	0	...	0	0	0	0	
921	Land Rover Range	1	4490000	51000	2017	Land	0	0	0	0	...	0	0	0	1	
143	Mercedes Benz GL	1	4450000	105000	2016	Mercedes	0	0	0	0	...	0	0	0	1	
142	Jaguar XJ L	1	4425000	71000	2016	Jaguar	0	0	0	0	...	0	0	0	1	
66	Volvo XC90 D5	1	3700000	110000	2016	Volvo	0	0	0	0	...	0	0	0	1	
185	Toyota Fortuner	1	3650000	42000	2019	Toyota	0	0	0	0	...	0	0	0	1	
611	BMW X5 xDrive30d	1	3600000	110000	2014	BMW	0	0	0	0	...	0	0	0	1	
923	Porsche Cayenne Select	1	3300000	38000	2011	Porsche	0	0	0	0	...	0	0	0	0	
95	Jeep Compass Limited	1	3000000	10000	2021	Jeep	0	0	0	0	...	0	0	0	1	
237	Toyota Fortuner 2.8	1	2931000	55000	2019	Toyota	0	0	0	0	...	0	0	0	1	

10 rows × 53 columns

In [69]:
```
sns.barplot(x=a['Name'],y=a['Price'])
plt.xticks(rotation=90)
plt.title("Top 10 cars on basis of price")
```

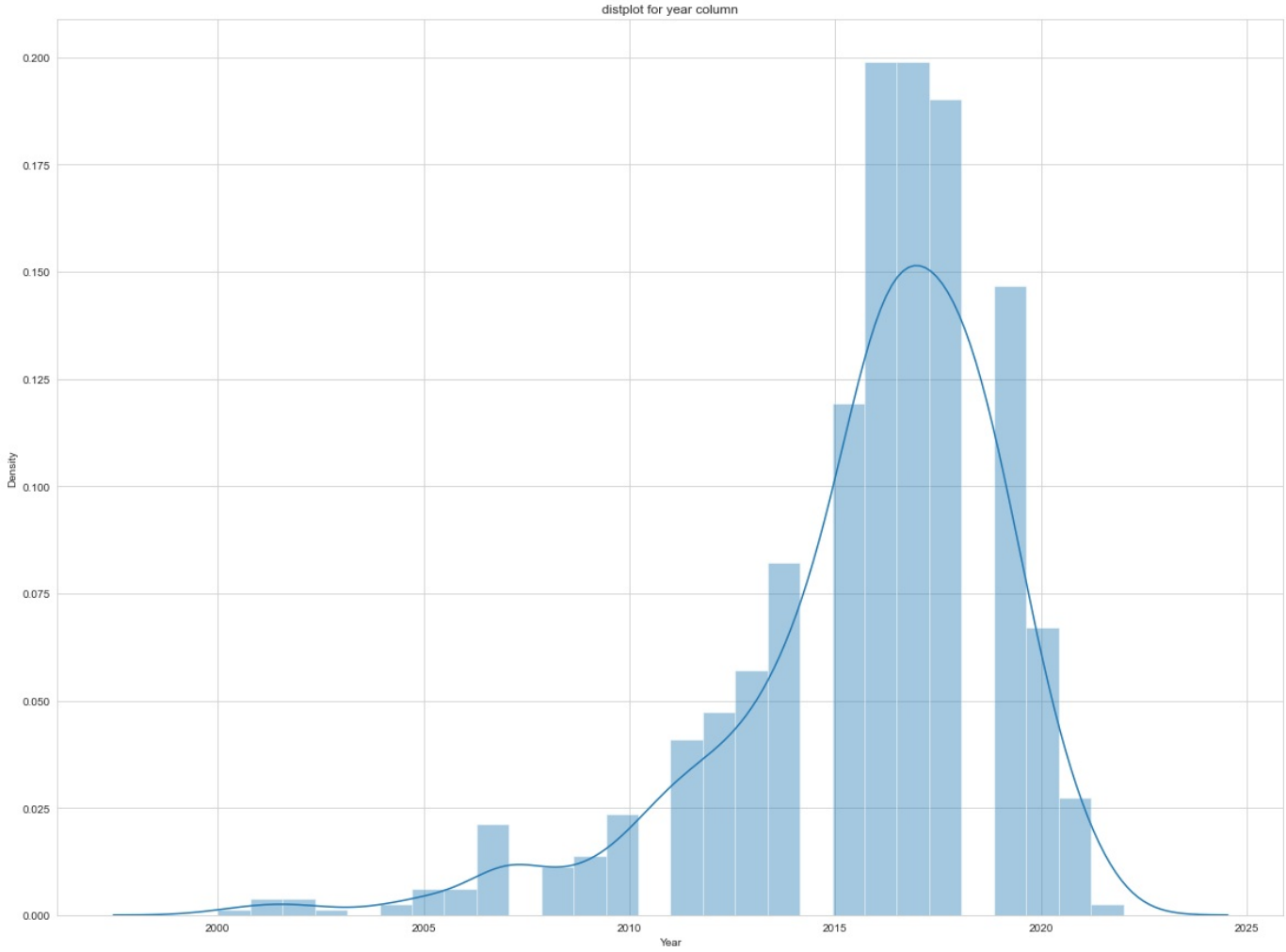
Out[69]:
```
Text(0.5, 1.0, 'Top 10 cars on basis of price')
```



Name

In [70]:
```
plt.figure(figsize=(20,15))
plt.title("distplot for year column")
sns.distplot(df['Year'])
plt.show()
```

```
C:\Users\revan\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecat
ed function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-lev
el function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



Feature Engineering and Data Preprocessing

In [71]:
```
dummies3=pd.get_dummies(df['Name'])
dummies3
```

Out[71]:

	Audi Q3	Audi Q5 2.0	Audi Q7 3.0	Audi Q7 35	BMW 3 Series	BMW 5 Series	BMW X5 xDrive30d	Chevrolet Aveo	Chevrolet Beat	Chevrolet Beat LS	...	Volkswagen Polo	Volkswagen Polo COMFORTLINE	Volkswagen Polo Comfortline	Volksw Pol
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1019	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
1020	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
1021	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
1022	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
1023	0	0	0	0	0	0	0	0	0	0	...	0	0	0	

1024 rows × 229 columns

In [72]:
```
dummies3=dummies3.drop(dummies3.columns[228],axis=1)
dummies3
```

Out[72]:

	Audi Q3	Audi Q5 2.0	Audi Q7 3.0	Audi Q7 35	BMW 3 Series	BMW 5 Series	BMW X5 xDrive30d	Chevrolet Aveo	Chevrolet Beat	Chevrolet Beat LS	...	Volkswagen Ameo Comfortline	Volkswagen Polo	Volkswagen Polo COMFORTLINE	Volksw Comfo
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1019	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
1020	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
1021	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
1022	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
1023	0	0	0	0	0	0	0	0	0	0	...	0	0	0	

1024 rows × 228 columns

In [73]:
```
dummies4=pd.get_dummies(df['Company'])
dummies4
```

Out[73]:

	Audi	BMW	Chevrolet	Datsun	Fiat	Ford	Honda	Hyundai	Jaguar	Jeep	...	Mercedes	Nissan	Porsche	Renault	Skoda	Ssangyong
0	0	0	0	0	0	1	0	0	0	0	...	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0
2	0	0	0	0	0	0	0	1	0	0	...	0	0	0	0	0	0
3	0	0	0	0	0	0	0	1	0	0	...	0	0	0	0	0	0
4	0	0	0	0	0	0	1	0	0	0	...	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1019	0	0	0	0	0	0	0	1	0	0	...	0	0	0	0	0	0
1020	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0
1021	0	0	0	0	0	0	0	0	0	1	...	0	0	0	0	0	0
1022	0	0	0	0	0	0	0	0	0	1	...	0	0	0	0	0	0
1023	0	0	0	0	0	0	0	0	0	1	...	0	0	0	0	0	0

1024 rows × 25 columns

In [74]:
```
dummies4=dummies4.drop(dummies4.columns[24],axis=1)
dummies4
```

Out[74]:

	Audi	BMW	Chevrolet	Datsun	Fiat	Ford	Honda	Hyundai	Jaguar	Jeep	...	Maruti	Mercedes	Nissan	Porsche	Renault	Skoda	Ssa
0	0	0	0	0	0	1	0	0	0	0	...	0	0	0	0	0	0	
1	0	0	0	0	0	0	0	0	0	0	...	1	0	0	0	0	0	
2	0	0	0	0	0	0	0	1	0	0	...	0	0	0	0	0	0	
3	0	0	0	0	0	0	0	1	0	0	...	0	0	0	0	0	0	
4	0	0	0	0	0	0	1	0	0	0	...	0	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1019	0	0	0	0	0	0	0	1	0	0	...	0	0	0	0	0	0	
1020	0	0	0	0	0	0	0	0	0	0	...	1	0	0	0	0	0	
1021	0	0	0	0	0	0	0	0	0	1	...	0	0	0	0	0	0	
1022	0	0	0	0	0	0	0	0	0	1	...	0	0	0	0	0	0	
1023	0	0	0	0	0	0	0	0	0	1	...	0	0	0	0	0	0	

1024 rows × 24 columns

In [75]:
```
df=df.drop(['Name','Company'],axis=1)
df
```

Out[75]:

	Label	Price	Kms_driven	Year	Ahmedabad	Anand	Bangalore	Bhubaneswar	Bilaspur	BolpurSantiniketan	...	Udaipur	CNG	CNG
0	2	380000	35056	2015	0	0	1	0	0	0	...	0	0	0
1	2	465000	44000	2016	0	0	1	0	0	0	...	0	0	0
2	2	1350000	42917	2018	0	0	1	0	0	0	...	0	0	0
3	2	1019699	16112	2019	0	0	0	0	0	0	...	0	0	0
4	2	713499	30988	2017	0	0	0	0	0	0	...	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1019	1	229000	65000	2014	0	0	1	0	0	0	...	0	0	0
1020	1	275000	60000	2014	0	0	1	0	0	0	...	0	0	0
1021	1	1750000	31000	2017	0	0	0	0	0	0	...	0	0	0
1022	1	1750000	31000	2017	0	0	0	0	1	0	...	0	0	0
1023	1	1750000	31000	2017	0	0	0	0	0	0	...	0	0	0

1024 rows × 51 columns

In [76]:
```
df=pd.concat([df,dummies3,dummies4],axis=1)
df
```

Out[76]:

	Label	Price	Kms_driven	Year	Ahmedabad	Anand	Bangalore	Bhubaneswar	Bilaspur	BolpurSantiniketan	...	Maruti	Mercedes	Nis
0	2	380000	35056	2015	0	0	1	0	0	0	...	0	0	
1	2	465000	44000	2016	0	0	1	0	0	0	...	1	0	
2	2	1350000	42917	2018	0	0	1	0	0	0	...	0	0	
3	2	1019699	16112	2019	0	0	0	0	0	0	...	0	0	
4	2	713499	30988	2017	0	0	0	0	0	0	...	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1019	1	229000	65000	2014	0	0	1	0	0	0	...	0	0	
1020	1	275000	60000	2014	0	0	1	0	0	0	...	1	0	
1021	1	1750000	31000	2017	0	0	0	0	0	0	...	0	0	
1022	1	1750000	31000	2017	0	0	0	0	1	0	...	0	0	
1023	1	1750000	31000	2017	0	0	0	0	0	0	...	0	0	

1024 rows × 303 columns

In [77]:
```
from sklearn.preprocessing import StandardScaler
scaler=StandardScaler()
```

In [78]:
```
Kms_driven=np.array(df['Kms_driven'])
Kms_driven=Kms_driven.reshape(-1,1)
Kms_driven
```

Out[78]:
```
array([[35056],
       [44000],
       [42917],
       ...,
       [31000],
       [31000],
       [31000]], dtype=int64)
```

In [79]:
```
df['Kms_driven']=scaler.fit_transform(Kms_driven)
df['Kms_driven']
```

Out[79]:
```
0      -0.398319
1      -0.163683
```

```
2      -0.192094
3      -0.895296
4      -0.505039
          ...
1019    0.387231
1020    0.256061
1021   -0.504724
1022   -0.504724
1023   -0.504724
Name: Kms_driven, Length: 1024, dtype: float64
```

In [80]:
```
Year=np.array(df['Year'])
Year=Year.reshape(-1,1)
Year
```

Out[80]:
```
array([[2015],
       [2016],
       [2018],
       ...,
       [2017],
       [2017],
       [2017]], dtype=int64)
```

In [81]:
```
df['Year']=scaler.fit_transform(Year)
df['Year']
```

Out[81]:
```
0      -0.241586
1       0.053622
2       0.644036
3       0.939243
4       0.348829
          ...
1019   -0.536793
1020   -0.536793
1021    0.348829
1022    0.348829
1023    0.348829
Name: Year, Length: 1024, dtype: float64
```

In [82]:
```
Y=df['Year']
Y
```

Out[82]:
```
0      -0.241586
1       0.053622
2       0.644036
3       0.939243
4       0.348829
          ...
1019   -0.536793
1020   -0.536793
1021    0.348829
1022    0.348829
1023    0.348829
Name: Year, Length: 1024, dtype: float64
```

In [83]:
```
X=df.drop(['Price'],axis=1)
X
```

Out[83]:

	Label	Kms_driven	Year	Ahmedabad	Anand	Bangalore	Bhubaneswar	Bilaspur	BolpurSantiniketan	Chandigarh	...	Maruti	Merce
0	2	-0.398319	-0.241586	0	0	1	0	0	0	0	...	0	
1	2	-0.163683	0.053622	0	0	1	0	0	0	0	...	1	
2	2	-0.192094	0.644036	0	0	1	0	0	0	0	...	0	
3	2	-0.895296	0.939243	0	0	0	0	0	0	0	...	0	
4	2	-0.505039	0.348829	0	0	0	0	0	0	0	...	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...
1019	1	0.387231	-0.536793	0	0	1	0	0	0	0	...	0	
1020	1	0.256061	-0.536793	0	0	1	0	0	0	0	...	1	
1021	1	-0.504724	0.348829	0	0	0	0	0	0	0	...	0	
1022	1	-0.504724	0.348829	0	0	0	0	1	0	0	...	0	

1023	1	-0.504724	0.348829	0	0	0	0	0	0	0	...	0

1024 rows × 302 columns

In [84]:
```
df
```

Out[84]:

	Label	Price	Kms_driven	Year	Ahmedabad	Anand	Bangalore	Bhubaneswar	Bilaspur	BolpurSantiniketan	...	Maruti	Mercedes
0	2	380000	-0.398319	-0.241586	0	0	1	0	0	0	...	0	0
1	2	465000	-0.163683	0.053622	0	0	1	0	0	0	...	1	0
2	2	1350000	-0.192094	0.644036	0	0	1	0	0	0	...	0	0
3	2	1019699	-0.895296	0.939243	0	0	0	0	0	0	...	0	0
4	2	713499	-0.505039	0.348829	0	0	0	0	0	0	...	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
1019	1	229000	0.387231	-0.536793	0	0	1	0	0	0	...	0	0
1020	1	275000	0.256061	-0.536793	0	0	1	0	0	0	...	1	0
1021	1	1750000	-0.504724	0.348829	0	0	0	0	0	0	...	0	0
1022	1	1750000	-0.504724	0.348829	0	0	0	0	1	0	...	0	0
1023	1	1750000	-0.504724	0.348829	0	0	0	0	0	0	...	0	0

1024 rows × 303 columns

In [85]:
```
from sklearn.model_selection import train_test_split
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,train_size=0.8,random_state=1)
```

In [86]:
```
X_train
```

Out[86]:

	Label	Kms_driven	Year	Ahmedabad	Anand	Bangalore	Bhubaneswar	Bilaspur	BolpurSantiniketan	Chandigarh	...	Maruti	Merced
878	1	-0.386671	0.348829	0	0	0	0	0	0	0	...	0	
404	2	-0.923628	0.348829	0	0	1	0	0	0	0	...	1	
718	2	0.115578	-0.536793	0	0	0	0	0	0	0	...	1	
846	2	-0.227903	-0.241586	0	0	1	0	0	0	0	...	0	
928	2	-0.409180	0.939243	0	0	0	0	0	0	0	...	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	
767	2	0.935337	0.053622	0	0	0	0	0	0	0	...	0	
72	2	0.497807	0.348829	0	0	0	0	0	0	0	...	0	
908	2	0.009435	0.644036	0	0	1	0	0	0	0	...	0	
235	1	0.911910	-4.079280	0	0	0	0	0	0	0	...	0	
37	2	-1.256091	1.234451	0	0	0	0	0	0	0	...	1	

819 rows × 302 columns

In [87]:
```
X_test
```

Out[87]:

	Label	Kms_driven	Year	Ahmedabad	Anand	Bangalore	Bhubaneswar	Bilaspur	BolpurSantiniketan	Chandigarh	...	Maruti	Merce
830	1	0.177359	-2.012829	0	0	0	0	0	0	0	...	1	
795	2	-0.675665	0.939243	0	0	1	0	0	0	0	...	1	
495	1	-0.006279	-0.536793	0	0	0	0	0	0	0	...	0	
822	2	-0.192724	0.348829	0	0	0	0	0	0	0	...	1	
859	2	0.671135	-1.422415	0	0	0	0	0	0	0	...	1	
...	...	...	...	...	...	...	...	...	...	...	...	...	
770	2	1.200458	0.053622	0	0	0	0	0	0	0	...	0	
582	2	-0.935801	0.939243	0	0	1	0	0	0	0	...	1	
1009	2	-0.795711	0.053622	0	0	0	0	0	0	0	...	0	
267	2	-0.497274	-0.536793	0	0	0	0	0	0	0	...	1	

974	1	0.072423	-1.717622	0	0	0	0	0	0	0	...	1

205 rows × 302 columns

In [88]:
```
Y_test
```

Out[88]:
```
830    -2.012829
795     0.939243
495    -0.536793
822     0.348829
859    -1.422415
          ...
770     0.053622
582     0.939243
1009    0.053622
267    -0.536793
974    -1.717622
Name: Year, Length: 205, dtype: float64
```

In [89]:
```
Y_train
```

Out[89]:
```
878    0.348829
404    0.348829
718   -0.536793
846   -0.241586
928    0.939243
         ...
767    0.053622
72     0.348829
908    0.644036
235   -4.079280
37     1.234451
Name: Year, Length: 819, dtype: float64
```

Model Building

In [90]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(X_train,Y_train)


Y_predicted=lr.predict(X_test)
print(Y_predicted[0:5])
Y_test[0:5]
```

Out[90]:
```
[-2.01282938  0.9392435  -0.53679294  0.34882892 -1.4224148 ]
830   -2.012829
795    0.939243
495   -0.536793
822    0.348829
859   -1.422415
Name: Year, dtype: float64
```

In [91]:
```
from sklearn.metrics import r2_score
r2=r2_score(Y_test,Y_predicted)
r2
```

Out[91]:
```
0.999999999696002
```

In [92]:
```
from sklearn.model_selection import cross_val_score
cross_val_score(lr,X,Y,cv=10)
np.average(cross_val_score(lr,X,Y,cv=10))
```

Out[92]:
```
0.9999556054888558
```

In [93]:
```
from sklearn.linear_model import Lasso
lasso=Lasso()
lasso.fit(X_train,Y_train)
#lasso.fit(X_train,Y_train)


Y_predicted=lasso.predict(X_test)
print(Y_predicted[0:5])
Y_test[0:5]
```

Out[93]:
```
[-0.17118578  0.0310127  -0.07008654 -0.009427   -0.13074608]
830   -2.012829
795    0.939243
495   -0.536793
822    0.348829
859   -1.422415
Name: Year, dtype: float64
```

In [ ]:
```
from sklearn.metrics import r2_score
r2=r2_score(Y_test,Y_predicted)
r2
```

In [ ]:
```
cross_val_score(lr,X,Y,cv=10)
np.average(cross_val_score(lr,X,Y,cv=10))
```

In [ ]:
```
from sklearn.linear_model import Ridge
rr=Ridge()
rr.fit(X_train,Y_train)


Y_predicted=rr.predict(X_test)
print(Y_predicted[0:5])
Y_test[0:5]
```

In [ ]:
```
r2=r2_score(Y_test,Y_predicted)
r2
```

In [ ]:
```
cross_val_score(lr,X,Y,cv=10)
np.average(cross_val_score(lr,X,Y,cv=10))
```

In [ ]:
```
from sklearn.tree import DecisionTreeRegressor
dtr=DecisionTreeRegressor()
dtr.fit(X_train,Y_train)

Y_predicted=dtr.predict(X_test)
print(Y_predicted[0:5])
Y_test[0:5]
```

In [ ]:
```
r2=r2_score(Y_test,Y_predicted)
r2
```

In [ ]:
```
cross_val_score(lr,X,Y,cv=10)
np.average(cross_val_score(lr,X,Y,cv=10))
```

In [ ]:
```
from sklearn.model_selection import GridSearchCV
#linearRegression
parameters={'normalize':[True,False]}
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: