In [1]:
```
import pandas as pd
```

In [2]:
```
import numpy as np
```

In [3]:
```
from sklearn.linear_model import LinearRegression
```

In [4]:
```
from sklearn.model_selection import train_test_split
```

In [5]:
```
data=pd.read_csv('./quiker_car.csv')
```

In [6]:
```
data
```

Out[6]:

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing XO eRLX Euro III	Hyundai	2007	80,000	45,000 kms	Petrol
1	Mahindra Jeep CL550 MDI	Mahindra	2006	4,25,000	40 kms	Diesel
2	Maruti Suzuki Alto 800 Vxi	Maruti	2018	Ask For Price	22,000 kms	Petrol
3	Hyundai Grand i10 Magna 1.2 Kappa VTVT	Hyundai	2014	3,25,000	28,000 kms	Petrol
4	Ford EcoSport Titanium 1.5L TDCi	Ford	2014	5,75,000	36,000 kms	Diesel
...	...	...	...	...	...	...
887	Ta	Tara	zest	3,10,000	NaN	NaN
888	Tata Zest XM Diesel	Tata	2018	2,60,000	27,000 kms	Diesel
889	Mahindra Quanto C8	Mahindra	2013	3,90,000	40,000 kms	Diesel
890	Honda Amaze 1.2 E i VTEC	Honda	2014	1,80,000	Petrol	NaN
891	Chevrolet Sail 1.2 LT ABS	Chevrolet	2014	1,60,000	Petrol	NaN

892 rows × 6 columns

In [7]:
```
data.name
```

Out[7]:
```
0        Hyundai Santro Xing XO eRLX Euro III
1                      Mahindra Jeep CL550 MDI
2                  Maruti Suzuki Alto 800 Vxi
3      Hyundai Grand i10 Magna 1.2 Kappa VTVT
4            Ford EcoSport Titanium 1.5L TDCi
                          ...
887                                       Ta
888                      Tata Zest XM Diesel
889                       Mahindra Quanto C8
890                  Honda Amaze 1.2 E i VTEC
891                 Chevrolet Sail 1.2 LT ABS
Name: name, Length: 892, dtype: object
```

In [8]:
```
#we removed the unwanted part of price column

data=data[data['Price']!='Ask For Price']
```

In [9]:
```
data
```

Out[9]:

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing XO eRLX Euro III	Hyundai	2007	80,000	45,000 kms	Petrol
1	Mahindra Jeep CL550 MDI	Mahindra	2006	4,25,000	40 kms	Diesel
3	Hyundai Grand i10 Magna 1.2 Kappa VTVT	Hyundai	2014	3,25,000	28,000 kms	Petrol
4	Ford EcoSport Titanium 1.5L TDCi	Ford	2014	5,75,000	36,000 kms	Diesel
6	Ford Figo	Ford	2012	1,75,000	41,000 kms	Diesel
...	...	...	...	...	...	...
887	Ta	Tara	zest	3,10,000	NaN	NaN
888	Tata Zest XM Diesel	Tata	2018	2,60,000	27,000 kms	Diesel
889	Mahindra Quanto C8	Mahindra	2013	3,90,000	40,000 kms	Diesel
890	Honda Amaze 1.2 E i VTEC	Honda	2014	1,80,000	Petrol	NaN
891	Chevrolet Sail 1.2 LT ABS	Chevrolet	2014	1,60,000	Petrol	NaN

857 rows × 6 columns

In [10]:
```
data.isna().sum()
```

Out[10]:
```
name           0
company        0
year           0
Price          0
kms_driven    38
fuel_type     41
dtype: int64
```

In [11]:
```
data['year'].unique()
```

Out[11]:
```
array(['2007', '2006', '2014', '2012', '2013', '2016', '2015', '2010',
       '2017', '2008', '2018', '2011', '2019', '2009', '2005', '2000',
       '150k', 'TOUR', '2003', 'r 15', '2004', 'sale', '1995', 'ara)',
       '2002', 'SELL', '2001', 'tion', 'odel', '2 bs', 'arry', 'o...',
       'Zest', 'ture', 'emi', 'car', 'able', 'd...', 'SALE', 'sell',
       'd Ex', 'n...', 'e...', 'go .', 'k...', 'o c4', 'zire', 'Sumo',
       'cab', 'EV2', 'r...', 'zest'], dtype=object)
```

In [12]:
```
data.dtypes
```

Out[12]:
```
name          object
company       object
year          object
Price         object
kms_driven    object
fuel_type     object
dtype: object
```

In [13]:
```
data=data[data['year'].str.isnumeric()]
```

In [14]:
```
data
```

Out[14]:

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing XO eRLX Euro III	Hyundai	2007	80,000	45,000 kms	Petrol
1	Mahindra Jeep CL550 MDI	Mahindra	2006	4,25,000	40 kms	Diesel
3	Hyundai Grand i10 Magna 1.2 Kappa VTVT	Hyundai	2014	3,25,000	28,000 kms	Petrol
4	Ford EcoSport Titanium 1.5L TDCi	Ford	2014	5,75,000	36,000 kms	Diesel
6	Ford Figo	Ford	2012	1,75,000	41,000 kms	Diesel
...	...	...	...	...	...	...
886	Toyota Corolla Altis	Toyota	2009	3,00,000	1,32,000 kms	Petrol
888	Tata Zest XM Diesel	Tata	2018	2,60,000	27,000 kms	Diesel
889	Mahindra Quanto C8	Mahindra	2013	3,90,000	40,000 kms	Diesel
890	Honda Amaze 1.2 E i VTEC	Honda	2014	1,80,000	Petrol	NaN
891	Chevrolet Sail 1.2 LT ABS	Chevrolet	2014	1,60,000	Petrol	NaN

819 rows × 6 columns

In [15]:
```
data['year'].unique()
```

Out[15]:
```
array(['2007', '2006', '2014', '2012', '2013', '2016', '2015', '2010',
       '2017', '2008', '2018', '2011', '2019', '2009', '2005', '2000',
       '2003', '2004', '1995', '2002', '2001'], dtype=object)
```

In [16]:
```
data.dtypes
```

Out[16]:
```
name          object
company       object
year          object
Price         object
kms_driven    object
fuel_type     object
dtype: object
```

In [17]:
```
data.shape
```

Out[17]:
```
(819, 6)
```

In [18]:
```
data['year']=data['year'].astype(int)
```

```
C:\Users\91855\AppData\Local\Temp/ipykernel_14272/30093866.py:1: SettingWithCopyWarn
ing:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  data['year']=data['year'].astype(int)
```

In [19]:
```
data
```

Out[19]:

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing XO eRLX Euro III	Hyundai	2007	80,000	45,000 kms	Petrol
1	Mahindra Jeep CL550 MDI	Mahindra	2006	4,25,000	40 kms	Diesel
3	Hyundai Grand i10 Magna 1.2 Kappa VTVT	Hyundai	2014	3,25,000	28,000 kms	Petrol
4	Ford EcoSport Titanium 1.5L TDCi	Ford	2014	5,75,000	36,000 kms	Diesel
6	Ford Figo	Ford	2012	1,75,000	41,000 kms	Diesel
...	...	...	...	...	...	...
886	Toyota Corolla Altis	Toyota	2009	3,00,000	1,32,000 kms	Petrol
888	Tata Zest XM Diesel	Tata	2018	2,60,000	27,000 kms	Diesel
889	Mahindra Quanto C8	Mahindra	2013	3,90,000	40,000 kms	Diesel
890	Honda Amaze 1.2 E i VTEC	Honda	2014	1,80,000	Petrol	NaN
891	Chevrolet Sail 1.2 LT ABS	Chevrolet	2014	1,60,000	Petrol	NaN

819 rows × 6 columns

In [20]:
```
data.dtypes
```

Out[20]:
```
name          object
company       object
year           int32
Price         object
kms_driven    object
fuel_type     object
dtype: object
```

In [21]:
```
#we completed data cleaning and typecasting of year column
```

In [22]:
```
data['Price'].unique()
```

Out[22]:
```
array(['80,000', '4,25,000', '3,25,000', '5,75,000', '1,75,000',
       '1,90,000', '8,30,000', '2,50,000', '1,82,000', '3,15,000',
       '4,15,000', '3,20,000', '10,00,000', '5,00,000', '3,50,000',
       '1,60,000', '3,10,000', '75,000', '1,00,000', '2,90,000', '95,000',
       '1,80,000', '3,85,000', '1,05,000', '6,50,000', '6,89,999',
       '4,48,000', '5,49,000', '5,01,000', '4,89,999', '2,80,000',
       '3,49,999', '2,84,999', '3,45,000', '4,99,999', '2,35,000',
       '2,49,999', '14,75,000', '3,95,000', '2,20,000', '1,70,000',
       '85,000', '2,00,000', '5,70,000', '1,10,000', '4,48,999',
       '18,91,111', '1,59,500', '3,44,999', '4,49,999', '8,65,000',
       '6,99,000', '3,75,000', '2,24,999', '12,00,000', '1,95,000',
```

```
       '3,51,000', '2,40,000', '90,000', '1,55,000', '6,00,000',
       '1,89,500', '2,10,000', '3,90,000', '1,35,000', '16,00,000',
       '7,01,000', '2,65,000', '5,25,000', '3,72,000', '6,35,000',
       '5,50,000', '4,85,000', '3,29,500', '2,51,111', '5,69,999',
       '69,999', '2,99,999', '3,99,999', '4,50,000', '2,70,000',
       '1,58,400', '1,79,000', '1,25,000', '2,99,000', '1,50,000',
       '2,75,000', '2,85,000', '3,40,000', '70,000', '2,89,999',
       '8,49,999', '7,49,999', '2,74,999', '9,84,999', '5,99,999',
       '2,44,999', '4,74,999', '2,45,000', '1,69,500', '3,70,000',
       '1,68,000', '1,45,000', '98,500', '2,09,000', '1,85,000',
       '9,00,000', '6,99,999', '1,99,999', '5,44,999', '1,99,000',
       '5,40,000', '49,000', '7,00,000', '55,000', '8,95,000', '3,55,000',
       '5,65,000', '3,65,000', '40,000', '4,00,000', '3,30,000',
       '5,80,000', '3,79,000', '2,19,000', '5,19,000', '7,30,000',
       '20,00,000', '21,00,000', '14,00,000', '3,11,000', '8,55,000',
       '5,35,000', '1,78,000', '3,00,000', '2,55,000', '5,49,999',
       '3,80,000', '57,000', '4,10,000', '2,25,000', '1,20,000', '59,000',
       '5,99,000', '6,75,000', '72,500', '6,10,000', '2,30,000',
       '5,20,000', '5,24,999', '4,24,999', '6,44,999', '5,84,999',
       '7,99,999', '4,44,999', '6,49,999', '9,44,999', '5,74,999',
       '3,74,999', '1,30,000', '4,01,000', '13,50,000', '1,74,999',
       '2,39,999', '99,999', '3,24,999', '10,74,999', '11,30,000',
       '1,49,000', '7,70,000', '30,000', '3,35,000', '3,99,000', '65,000',
       '1,69,999', '1,65,000', '5,60,000', '9,50,000', '7,15,000',
       '45,000', '9,40,000', '1,55,555', '15,00,000', '4,95,000',
       '8,00,000', '12,99,000', '5,30,000', '14,99,000', '32,000',
       '4,05,000', '7,60,000', '7,50,000', '4,19,000', '1,40,000',
       '15,40,000', '1,23,000', '4,98,000', '4,80,000', '4,88,000',
       '15,25,000', '5,48,900', '7,25,000', '99,000', '52,000',
       '28,00,000', '4,99,000', '3,81,000', '2,78,000', '6,90,000',
       '2,60,000', '90,001', '1,15,000', '15,99,000', '1,59,000',
       '51,999', '2,15,000', '35,000', '11,50,000', '2,69,000', '60,000',
       '4,30,000', '85,00,003', '4,01,919', '4,90,000', '4,24,000',
       '2,05,000', '5,49,900', '4,35,000', '1,89,700', '3,89,700',
       '3,60,000', '2,95,000', '1,14,990', '10,65,000', '4,70,000',
       '48,000', '1,88,000', '4,65,000', '1,79,999', '21,90,000',
       '23,90,000', '10,75,000', '4,75,000', '10,25,000', '6,15,000',
       '19,00,000', '14,90,000', '15,10,000', '18,50,000', '7,90,000',
       '17,25,000', '12,25,000', '68,000', '9,70,000', '31,00,000',
       '8,99,000', '88,000', '53,000', '5,68,500', '71,000', '5,90,000',
       '7,95,000', '42,000', '1,89,000', '1,62,000', '35,999',
       '29,00,000', '39,999', '50,500', '5,10,000', '8,60,000',
       '5,00,001'], dtype=object)
```

In [23]:

```
#firstly we removed askforvalue price columns rows
#now we have to remove columns and we have to change object data to int
```

In [24]:

```
data['Price'].str.replace(',','')
```

Out[24]:

```
0       80000
1      425000
3      325000
4      575000
6      175000
        ...
886    300000
888    260000
889    390000
890    180000
891    160000
Name: Price, Length: 819, dtype: object
```

In [25]:
```
data['Price']=data['Price'].str.replace(',','')
```

```
C:\Users\91855\AppData\Local\Temp/ipykernel_14272/239215182.py:1: SettingWithCopyWar
ning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  data['Price']=data['Price'].str.replace(',','')
```

In [26]:
```
data
```

Out[26]:

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing XO eRLX Euro III	Hyundai	2007	80000	45,000 kms	Petrol
1	Mahindra Jeep CL550 MDI	Mahindra	2006	425000	40 kms	Diesel
3	Hyundai Grand i10 Magna 1.2 Kappa VTVT	Hyundai	2014	325000	28,000 kms	Petrol
4	Ford EcoSport Titanium 1.5L TDCi	Ford	2014	575000	36,000 kms	Diesel
6	Ford Figo	Ford	2012	175000	41,000 kms	Diesel
...	...	...	...	...	...	...
886	Toyota Corolla Altis	Toyota	2009	300000	1,32,000 kms	Petrol
888	Tata Zest XM Diesel	Tata	2018	260000	27,000 kms	Diesel
889	Mahindra Quanto C8	Mahindra	2013	390000	40,000 kms	Diesel
890	Honda Amaze 1.2 E i VTEC	Honda	2014	180000	Petrol	NaN
891	Chevrolet Sail 1.2 LT ABS	Chevrolet	2014	160000	Petrol	NaN

819 rows × 6 columns

In [27]:
```
data.dtypes
```

Out[27]:
```
name          object
company       object
year           int32
Price         object
kms_driven    object
fuel_type     object
dtype: object
```

In [28]:
```
data['Price']=data['Price'].astype(int)
```

```
C:\Users\91855\AppData\Local\Temp/ipykernel_14272/1626439248.py:1: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  data['Price']=data['Price'].astype(int)
```

In [29]:
```
data
```

Out[29]:

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing XO eRLX Euro III	Hyundai	2007	80000	45,000 kms	Petrol
1	Mahindra Jeep CL550 MDI	Mahindra	2006	425000	40 kms	Diesel
3	Hyundai Grand i10 Magna 1.2 Kappa VTVT	Hyundai	2014	325000	28,000 kms	Petrol
4	Ford EcoSport Titanium 1.5L TDCi	Ford	2014	575000	36,000 kms	Diesel
6	Ford Figo	Ford	2012	175000	41,000 kms	Diesel
...	...	...	...	...	...	...
886	Toyota Corolla Altis	Toyota	2009	300000	1,32,000 kms	Petrol
888	Tata Zest XM Diesel	Tata	2018	260000	27,000 kms	Diesel
889	Mahindra Quanto C8	Mahindra	2013	390000	40,000 kms	Diesel
890	Honda Amaze 1.2 E i VTEC	Honda	2014	180000	Petrol	NaN
891	Chevrolet Sail 1.2 LT ABS	Chevrolet	2014	160000	Petrol	NaN

819 rows × 6 columns

In [30]:
```
data.dtypes
```

Out[30]:
```
name          object
company       object
year           int32
Price          int32
kms_driven    object
fuel_type     object
dtype: object
```

In [31]:
```
#now we are going to work with fuel_type column
```

In [32]:
```
data['fuel_type'].unique()
```

Out[32]:
```
array(['Petrol', 'Diesel', nan, 'LPG'], dtype=object)
```

In [33]:
```
#we have to remove NaN values or else we have to fill them
```

In [34]:
```
data.isna().sum()
```

Out[34]:
```
name          0
company       0
year          0
Price         0
kms_driven    0
fuel_type     3
dtype: int64
```

In [35]:
```
#there are only 3 NaN values so we are going to filled with petrol
```

In [36]:
```
data['fuel_type']=data['fuel_type'].fillna('Petrol')
```

```
C:\Users\91855\AppData\Local\Temp/ipykernel_14272/1723300110.py:1: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  data['fuel_type']=data['fuel_type'].fillna('Petrol')
```

In [37]:
```
data
```

Out[37]:

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing XO eRLX Euro III	Hyundai	2007	80000	45,000 kms	Petrol
1	Mahindra Jeep CL550 MDI	Mahindra	2006	425000	40 kms	Diesel
3	Hyundai Grand i10 Magna 1.2 Kappa VTVT	Hyundai	2014	325000	28,000 kms	Petrol
4	Ford EcoSport Titanium 1.5L TDCi	Ford	2014	575000	36,000 kms	Diesel
6	Ford Figo	Ford	2012	175000	41,000 kms	Diesel
...	...	...	...	...	...	...
886	Toyota Corolla Altis	Toyota	2009	300000	1,32,000 kms	Petrol
888	Tata Zest XM Diesel	Tata	2018	260000	27,000 kms	Diesel
889	Mahindra Quanto C8	Mahindra	2013	390000	40,000 kms	Diesel
890	Honda Amaze 1.2 E i VTEC	Honda	2014	180000	Petrol	Petrol
891	Chevrolet Sail 1.2 LT ABS	Chevrolet	2014	160000	Petrol	Petrol

819 rows × 6 columns

In [38]:
```
data['fuel_type'].unique()
```

Out[38]:
```
array(['Petrol', 'Diesel', 'LPG'], dtype=object)
```

In [39]:
```
#now we are going to work with kms_driven column
```

In [40]:
```
data['kms_driven'].unique()
```

Out[40]:
```
array(['45,000 kms', '40 kms', '28,000 kms', '36,000 kms', '41,000 kms',
       '25,000 kms', '24,530 kms', '60,000 kms', '30,000 kms',
       '32,000 kms', '48,660 kms', '4,000 kms', '16,934 kms',
       '43,000 kms', '35,550 kms', '39,522 kms', '39,000 kms',
       '55,000 kms', '72,000 kms', '15,975 kms', '70,000 kms',
       '23,452 kms', '35,522 kms', '48,508 kms', '15,487 kms',
       '82,000 kms', '20,000 kms', '68,000 kms', '38,000 kms',
       '27,000 kms', '33,000 kms', '46,000 kms', '16,000 kms',
       '47,000 kms', '35,000 kms', '30,874 kms', '15,000 kms',
       '29,685 kms', '1,30,000 kms', '19,000 kms', '54,000 kms',
       '13,000 kms', '38,200 kms', '22,000 kms', '50,000 kms',
       '13,500 kms', '3,600 kms', '45,863 kms', '60,500 kms',
       '12,500 kms', '18,000 kms', '13,349 kms', '29,000 kms',
       '44,000 kms', '42,000 kms', '14,000 kms', '49,000 kms',
       '36,200 kms', '51,000 kms', '1,04,000 kms', '33,333 kms',
```

```
'33,600 kms', '5,600 kms', '7,500 kms', '26,000 kms', '24,330 kms',
'65,480 kms', '2,00,000 kms', '59,000 kms', '99,000 kms',
'2,800 kms', '21,000 kms', '11,000 kms', '66,000 kms', '3,000 kms',
'7,000 kms', '38,500 kms', '37,200 kms', '43,200 kms',
'24,800 kms', '45,872 kms', '40,000 kms', '11,400 kms',
'97,200 kms', '52,000 kms', '31,000 kms', '1,75,430 kms',
'37,000 kms', '65,000 kms', '3,350 kms', '75,000 kms',
'62,000 kms', '73,000 kms', '2,200 kms', '54,870 kms',
'34,580 kms', '97,000 kms', '60 kms', '80,200 kms', '3,200 kms',
'0,000 kms', '5,000 kms', '588 kms', '71,200 kms', '1,75,400 kms',
'9,300 kms', '56,758 kms', '10,000 kms', '56,450 kms',
'56,000 kms', '32,700 kms', '9,000 kms', '73 kms', '1,60,000 kms',
'58,559 kms', '57,000 kms', '1,70,000 kms', '80,000 kms',
'6,821 kms', '23,000 kms', '34,000 kms', '1,800 kms',
'4,00,000 kms', '48,000 kms', '90,000 kms', '12,000 kms',
'69,900 kms', '1,66,000 kms', '122 kms', '0 kms', '36,469 kms',
'7,800 kms', '24,695 kms', '15,141 kms', '59,910 kms',
'1,00,000 kms', '4,500 kms', '1,29,000 kms', '300 kms',
'1,31,000 kms', '1,11,111 kms', '59,466 kms', '25,500 kms',
'44,005 kms', '2,110 kms', '43,222 kms', '1,00,200 kms', '65 kms',
'1,40,000 kms', '1,03,553 kms', '58,000 kms', '1,20,000 kms',
'49,800 kms', '100 kms', '81,876 kms', '6,020 kms', '55,700 kms',
'18,500 kms', '53,000 kms', '35,500 kms', '22,134 kms',
'1,000 kms', '8,500 kms', '87,000 kms', '6,000 kms', '8,000 kms',
'55,800 kms', '56,400 kms', '72,160 kms', '11,500 kms',
'1,33,000 kms', '2,000 kms', '88,000 kms', '65,422 kms',
'1,17,000 kms', '1,50,000 kms', '10,750 kms', '6,800 kms',
'9,800 kms', '57,923 kms', '30,201 kms', '6,200 kms', '37,518 kms',
'24,652 kms', '383 kms', '95,000 kms', '3,528 kms', '52,500 kms',
'47,900 kms', '52,800 kms', '1,95,000 kms', '48,008 kms',
'48,247 kms', '9,400 kms', '64,000 kms', '2,137 kms', '10,544 kms',
'1,47,000 kms', '90,001 kms', '48,006 kms', '74,000 kms',
'85,000 kms', '29,500 kms', '39,700 kms', '67,000 kms',
'19,336 kms', '60,105 kms', '45,933 kms', '1,02,563 kms',
'28,600 kms', '41,800 kms', '1,16,000 kms', '42,590 kms',
'7,400 kms', '54,500 kms', '76,000 kms', '00 kms', '11,523 kms',
'38,600 kms', '95,500 kms', '37,458 kms', '85,960 kms',
'12,516 kms', '30,600 kms', '2,550 kms', '62,500 kms',
'69,000 kms', '28,400 kms', '68,485 kms', '3,500 kms',
'85,455 kms', '63,000 kms', '1,600 kms', '77,000 kms',
'26,500 kms', '2,875 kms', '13,900 kms', '1,500 kms', '2,450 kms',
'1,625 kms', '33,400 kms', '60,123 kms', '1,37,495 kms',
'91,200 kms', '1,46,000 kms', '1,00,800 kms', '2,100 kms',
'2,500 kms', '1,32,000 kms', 'Petrol'], dtype=object)
```

In [41]:
```
#we have to remove those petrol fields
```

In [42]:
```
data=data[data['kms_driven']!='Petrol']
```

In [43]:
```
data
```

Out[43]:

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing XO eRLX Euro III	Hyundai	2007	80000	45,000 kms	Petrol
1	Mahindra Jeep CL550 MDI	Mahindra	2006	425000	40 kms	Diesel
3	Hyundai Grand i10 Magna 1.2 Kappa VTVT	Hyundai	2014	325000	28,000 kms	Petrol
4	Ford EcoSport Titanium 1.5L TDCi	Ford	2014	575000	36,000 kms	Diesel

	name	company	year	Price	kms_driven	fuel_type
6	Ford Figo	Ford	2012	175000	41,000 kms	Diesel
...	...	...	...	...	...	...
883	Maruti Suzuki Ritz VXI ABS	Maruti	2011	270000	50,000 kms	Petrol
885	Tata Indica V2 DLE BS III	Tata	2009	110000	30,000 kms	Diesel
886	Toyota Corolla Altis	Toyota	2009	300000	1,32,000 kms	Petrol
888	Tata Zest XM Diesel	Tata	2018	260000	27,000 kms	Diesel
889	Mahindra Quanto C8	Mahindra	2013	390000	40,000 kms	Diesel

817 rows × 6 columns

In [44]:
```
data['kms_driven']=data['kms_driven'].str.replace(',','')
```

```
C:\Users\91855\AppData\Local\Temp/ipykernel_14272/1095163534.py:1: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  data['kms_driven']=data['kms_driven'].str.replace(',','')
```

In [45]:
```
data
```

Out[45]:

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing XO eRLX Euro III	Hyundai	2007	80000	45000 kms	Petrol
1	Mahindra Jeep CL550 MDI	Mahindra	2006	425000	40 kms	Diesel
3	Hyundai Grand i10 Magna 1.2 Kappa VTVT	Hyundai	2014	325000	28000 kms	Petrol
4	Ford EcoSport Titanium 1.5L TDCi	Ford	2014	575000	36000 kms	Diesel
6	Ford Figo	Ford	2012	175000	41000 kms	Diesel
...	...	...	...	...	...	...
883	Maruti Suzuki Ritz VXI ABS	Maruti	2011	270000	50000 kms	Petrol
885	Tata Indica V2 DLE BS III	Tata	2009	110000	30000 kms	Diesel
886	Toyota Corolla Altis	Toyota	2009	300000	132000 kms	Petrol
888	Tata Zest XM Diesel	Tata	2018	260000	27000 kms	Diesel
889	Mahindra Quanto C8	Mahindra	2013	390000	40000 kms	Diesel

817 rows × 6 columns

In [46]:
```
data['kms_driven']=data['kms_driven'].str.replace(' kms','')
```

```
C:\Users\91855\AppData\Local\Temp/ipykernel_14272/4280961435.py:1: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  data['kms_driven']=data['kms_driven'].str.replace(' kms','')
```

In [47]:
```
data
```

Out[47]:

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing XO eRLX Euro III	Hyundai	2007	80000	45000	Petrol
1	Mahindra Jeep CL550 MDI	Mahindra	2006	425000	40	Diesel
3	Hyundai Grand i10 Magna 1.2 Kappa VTVT	Hyundai	2014	325000	28000	Petrol
4	Ford EcoSport Titanium 1.5L TDCi	Ford	2014	575000	36000	Diesel
6	Ford Figo	Ford	2012	175000	41000	Diesel
...	...	...	...	...	...	...
883	Maruti Suzuki Ritz VXI ABS	Maruti	2011	270000	50000	Petrol
885	Tata Indica V2 DLE BS III	Tata	2009	110000	30000	Diesel
886	Toyota Corolla Altis	Toyota	2009	300000	132000	Petrol
888	Tata Zest XM Diesel	Tata	2018	260000	27000	Diesel
889	Mahindra Quanto C8	Mahindra	2013	390000	40000	Diesel

817 rows × 6 columns

In [48]:
```
data['kms_driven']=data['kms_driven'].astype(int)
```

```
C:\Users\91855\AppData\Local\Temp/ipykernel_14272/1717504539.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  data['kms_driven']=data['kms_driven'].astype(int)
```

In [49]:
```
data.dtypes
```

Out[49]:
```
name          object
company       object
year           int32
Price          int32
kms_driven     int32
fuel_type     object
dtype: object
```

In [50]:
```
data.isna().sum()
```

Out[50]:
```
name          0
company       0
year          0
Price         0
kms_driven    0
```

```
fuel_type     0
dtype: int64
```

In [51]:
```
#now the final task is to presize name column data to maximum limit of 3 words
```

In [52]:
```
data['name']=data['name'].str.split()
```

```
C:\Users\91855\AppData\Local\Temp/ipykernel_14272/4072474120.py:1: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  data['name']=data['name'].str.split()
```

In [53]:
```
data
```

Out[53]:

	name	company	year	Price	kms_driven	fuel_type
0	[Hyundai, Santro, Xing, XO, eRLX, Euro, III]	Hyundai	2007	80000	45000	Petrol
1	[Mahindra, Jeep, CL550, MDI]	Mahindra	2006	425000	40	Diesel
3	[Hyundai, Grand, i10, Magna, 1.2, Kappa, VTVT]	Hyundai	2014	325000	28000	Petrol
4	[Ford, EcoSport, Titanium, 1.5L, TDCi]	Ford	2014	575000	36000	Diesel
6	[Ford, Figo]	Ford	2012	175000	41000	Diesel
...	...	...	...	...	...	...
883	[Maruti, Suzuki, Ritz, VXI, ABS]	Maruti	2011	270000	50000	Petrol
885	[Tata, Indica, V2, DLE, BS, III]	Tata	2009	110000	30000	Diesel
886	[Toyota, Corolla, Altis]	Toyota	2009	300000	132000	Petrol
888	[Tata, Zest, XM, Diesel]	Tata	2018	260000	27000	Diesel
889	[Mahindra, Quanto, C8]	Mahindra	2013	390000	40000	Diesel

817 rows × 6 columns

In [54]:
```
#we clearly splited data
```

In [55]:
```
data['name'].str.slice(0,3)
```

Out[55]:
```
0         [Hyundai, Santro, Xing]
1         [Mahindra, Jeep, CL550]
3           [Hyundai, Grand, i10]
4      [Ford, EcoSport, Titanium]
6                   [Ford, Figo]
                  ...
883        [Maruti, Suzuki, Ritz]
885            [Tata, Indica, V2]
886      [Toyota, Corolla, Altis]
888              [Tata, Zest, XM]
889        [Mahindra, Quanto, C8]
Name: name, Length: 817, dtype: object
```

In [56]:
```
data['name']=data['name'].str.slice(0,3)
```

```
C:\Users\91855\AppData\Local\Temp/ipykernel_14272/3295127314.py:1: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  data['name']=data['name'].str.slice(0,3)
```

In [57]:
```
data
```

Out[57]:

	name	company	year	Price	kms_driven	fuel_type
0	[Hyundai, Santro, Xing]	Hyundai	2007	80000	45000	Petrol
1	[Mahindra, Jeep, CL550]	Mahindra	2006	425000	40	Diesel
3	[Hyundai, Grand, i10]	Hyundai	2014	325000	28000	Petrol
4	[Ford, EcoSport, Titanium]	Ford	2014	575000	36000	Diesel
6	[Ford, Figo]	Ford	2012	175000	41000	Diesel
...	...	...	...	...	...	...
883	[Maruti, Suzuki, Ritz]	Maruti	2011	270000	50000	Petrol
885	[Tata, Indica, V2]	Tata	2009	110000	30000	Diesel
886	[Toyota, Corolla, Altis]	Toyota	2009	300000	132000	Petrol
888	[Tata, Zest, XM]	Tata	2018	260000	27000	Diesel
889	[Mahindra, Quanto, C8]	Mahindra	2013	390000	40000	Diesel

817 rows × 6 columns

In [58]:
```
data['name']=data['name'].str.join('')
```

```
C:\Users\91855\AppData\Local\Temp/ipykernel_14272/2987050025.py:1: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  data['name']=data['name'].str.join('')
```

In [59]:
```
data
```

Out[59]:

	name	company	year	Price	kms_driven	fuel_type
0	HyundaiSantroXing	Hyundai	2007	80000	45000	Petrol
1	MahindraJeepCL550	Mahindra	2006	425000	40	Diesel
3	HyundaiGrandi10	Hyundai	2014	325000	28000	Petrol
4	FordEcoSportTitanium	Ford	2014	575000	36000	Diesel

	name	company	year	Price	kms_driven	fuel_type
6	FordFigo	Ford	2012	175000	41000	Diesel
...	...	...	...	...	...	...
883	MarutiSuzukiRitz	Maruti	2011	270000	50000	Petrol
885	TataIndicaV2	Tata	2009	110000	30000	Diesel
886	ToyotaCorollaAltis	Toyota	2009	300000	132000	Petrol
888	TataZestXM	Tata	2018	260000	27000	Diesel
889	MahindraQuantoC8	Mahindra	2013	390000	40000	Diesel

817 rows × 6 columns

In [60]:
```
#we finished our data cleaning process
```

In [61]:
```
data.isna().sum()
```

Out[61]:
```
name          0
company       0
year          0
Price         0
kms_driven    0
fuel_type     0
dtype: int64
```

In [62]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 817 entries, 0 to 889
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   name        817 non-null    object
 1   company     817 non-null    object
 2   year        817 non-null    int32 
 3   Price       817 non-null    int32 
 4   kms_driven  817 non-null    int32 
 5   fuel_type   817 non-null    object
dtypes: int32(3), object(3)
memory usage: 35.1+ KB
```

In [ ]:
```

```

In [63]:
```
#now we are going to implement multiple-linear regression
```

In [88]:
```
x=data[['name','company','fuel_type','kms_driven','year']].to_numpy()
```

In [89]:
```
x
```

Out[89]:
```
array([['HyundaiSantroXing', 'Hyundai', 'Petrol', 45000, 2007],
       ['MahindraJeepCL550', 'Mahindra', 'Diesel', 40, 2006],
       ['HyundaiGrandi10', 'Hyundai', 'Petrol', 28000, 2014],
```

```
       ...,
       ['ToyotaCorollaAltis', 'Toyota', 'Petrol', 132000, 2009],
       ['TataZestXM', 'Tata', 'Diesel', 27000, 2018],
       ['MahindraQuantoC8', 'Mahindra', 'Diesel', 40000, 2013]],
      dtype=object)
```

In [90]:
```
y=data['Price'].to_numpy()
```

In [91]:
```
y
```

Out[91]:
```
array([  80000,  425000,  325000,  575000,  175000,  190000,  830000,
         250000,  182000,  315000,  415000,  320000,   80000,  425000,
        1000000,  500000,  350000,  160000,  350000,  310000,   75000,
         100000,  100000,  100000,  190000,  290000,   95000,  180000,
         385000,  250000,  180000,  105000,  105000,  650000,  689999,
         448000,  549000,  501000,  489999,  280000,  250000,  349999,
         284999,  345000,  499999,  235000,  249999, 1475000,  180000,
         385000,  250000,  180000,  105000,  105000,  395000,  220000,
         170000,   85000,  175000,  190000,  200000,  830000,  200000,
         570000,  315000,  182000,  315000,  110000,  501000,  448999,
        1891111,  235000,  159500,  344999,  344999,  449999, 1891111,
         865000,  699000,  375000,  489999,  224999, 1200000,  195000,
         351000,  160000,  240000,   90000,  415000,  155000,  600000,
         189500,  350000,  210000,  390000,  135000, 1600000,  701000,
         265000,  525000,  372000,  635000,  550000,  575000,  485000,
         155000,  345000,  325000,  329500,  195000,  251111,  569999,
          69999,  299999,  220000,  399999,  372000,  450000,  270000,
         350000,  158400,  350000,  179000,  125000,  200000,  299000,
         220000,  150000,  275000,  285000,  830000,  210000,  340000,
          90000,   70000,  289999,  349999,  849999,  749999,  399999,
         274999,  984999,  449999,  344999,  224999,  599999,  244999,
         399999,  489999,  474999,  499999,  310000,   85000,  245000,
         189500,  169500,  159500,  275000,  370000,  168000,  150000,
         145000,   98500,  699000,   85000,  575000,  549000,  209000,
         185000,  900000,  699999,  224999,  274999,  284999,  599999,
         199999,  544999,  199000,  320000,  540000,  340000,   75000,
         159500, 1891111,   49000,  700000,   55000,  448999,  895000,
         355000,  565000,  365000,  145000,  210000,   40000,  125000,
         135000,  135000,  285000,  145000,  135000,  450000,  375000,
         375000,  365000,  500000,  400000,  390000,  501000,  330000,
         580000,  265000,  379000,  219000,  385000,  275000,  330000,
         110000,   80000,  519000,  730000, 1475000,  699000, 2000000,
        2100000,  340000,  390000, 1400000,  245000,  320000,  320000,
         450000,  311000,  284999,  399999,  599999,  344999,  699000,
         580000,  855000,  535000, 1891111,  699000,  375000,  284999,
         178000,  300000,   90000,   95000,  255000,  245000,  329500,
         195000,  251111,  569999,   69999,  299999,  220000,  399999,
         249999,  289999, 1891111,  499999,  489999,  489999,  549999,
         380000,  325000,   57000,  349999,  689999,  349999,  410000,
         225000,  120000,  320000,   59000,  540000,   80000,  340000,
          75000,  220000,  159500,  599000,   80000,  675000, 1891111,
        1891111,  150000, 1891111,   72500,  610000,  230000,  175000,
         855000,  375000,  520000,  524999,  299999,  299999,  284999,
         220000,  424999,  644999,  399999,  199999,  584999,  349999,
         449999,  799999,  444999,  649999,  444999,  689999,  344999,
         944999,  274999,  689999,  574999,  374999,  199999,  549999,
         130000,  210000,  501000,  401000, 1350000,  600000,  610000,
         400000,  375000,  375000,  365000,  500000,  400000,  524999,
         449999,  174999,  244999,  574999,  244999,  239999,   99999,
         489999,  324999, 1074999,  230000,  699000, 1000000,  240000,
         110000,  390000,  501000, 1130000,  250000,  330000,  580000,
         340000,  120000,  265000,  265000,   85000,  379000,  175000,
```

219000, 350000, 149000, 385000, 425000, 150000, 225000,
375000, 770000, 30000, 275000, 330000, 335000, 450000,
225000, 80000, 130000, 245000, 399000, 450000, 65000,
75000, 70000, 190000, 600000, 245000, 240000, 155000,
169999, 450000, 40000, 165000, 270000, 280000, 560000,
950000, 310000, 715000, 340000, 235000, 610000, 95000,
1000000, 220000, 1200000, 230000, 45000, 940000, 155555,
1500000, 210000, 495000, 125000, 195000, 550000, 270000,
500000, 240000, 800000, 1299000, 530000, 1499000, 220000,
900000, 250000, 395000, 130000, 32000, 540000, 540000,
405000, 400000, 760000, 500000, 175000, 900000, 750000,
419000, 90000, 140000, 1540000, 275000, 150000, 230000,
123000, 900000, 900000, 300000, 499999, 165000, 498000,
480000, 488000, 250000, 220000, 290000, 1525000, 548900,
650000, 55000, 550000, 90000, 399000, 730000, 725000,
195000, 130000, 1525000, 190000, 250000, 80000, 120000,
149000, 250000, 120000, 450000, 99999, 135000, 225000,
99000, 370000, 52000, 2800000, 190000, 499000, 90000,
149000, 400000, 120000, 250000, 375000, 381000, 180000,
580000, 278000, 1000000, 690000, 480000, 85000, 40000,
90000, 340000, 260000, 250000, 180000, 350000, 90001,
115000, 1599000, 130000, 159000, 160000, 110000, 425000,
900000, 150000, 110000, 51999, 115000, 215000, 580000,
380000, 350000, 35000, 1150000, 300000, 269000, 60000,
400000, 430000, 140000, 8500003, 1299000, 199000, 90000,
550000, 265000, 100000, 215000, 380000, 401919, 490000,
280000, 650000, 160000, 424000, 225000, 350000, 950000,
485000, 205000, 160000, 310000, 180000, 549900, 150000,
175000, 95000, 230000, 230000, 180000, 400000, 185000,
385000, 90000, 32000, 435000, 225000, 189700, 389700,
365000, 360000, 210000, 170000, 380000, 295000, 185000,
160000, 290000, 100000, 315000, 114990, 120000, 125000,
210000, 855000, 210000, 260000, 95000, 255000, 300000,
340000, 550000, 60000, 750000, 230000, 130000, 270000,
280000, 280000, 280000, 600000, 190000, 500000, 1065000,
350000, 350000, 540000, 470000, 179000, 48000, 650000,
190000, 500000, 270000, 125000, 188000, 380000, 365000,
465000, 240000, 179999, 140000, 2190000, 2390000, 1075000,
475000, 1025000, 615000, 475000, 270000, 475000, 240000,
120000, 1900000, 360000, 450000, 900000, 650000, 275000,
210000, 175000, 85000, 1490000, 800000, 450000, 1000000,
1510000, 1850000, 790000, 1725000, 135000, 1000000, 299999,
1225000, 175000, 200000, 270000, 525000, 180000, 140000,
400000, 499000, 85000, 70000, 550000, 370000, 690000,
250000, 110000, 490000, 320000, 68000, 130000, 970000,
3100000, 280000, 125000, 285000, 165000, 250000, 865000,
390000, 60000, 215000, 475000, 899000, 1499000, 240000,
99000, 260000, 1200000, 115000, 88000, 390000, 135000,
90000, 220000, 424999, 135000, 95000, 430000, 115000,
215000, 53000, 500000, 85000, 165000, 200000, 200000,
425000, 600000, 130000, 430000, 568500, 71000, 560000,
590000, 750000, 125000, 135000, 60000, 120000, 95000,
240000, 115000, 795000, 55000, 300000, 320000, 265000,
160000, 300000, 130000, 250000, 380000, 42000, 400000,
120000, 120000, 130000, 189000, 365000, 170000, 215000,
60000, 599999, 400000, 900000, 299999, 374999, 600000,
70000, 100000, 150000, 225000, 210000, 425000, 162000,
60000, 650000, 750000, 375000, 230000, 35999, 380000,
560000, 285000, 2900000, 39999, 85000, 395000, 175000,
400000, 750000, 250000, 425000, 525000, 130000, 30000,
475000, 300000, 60000, 100000, 260000, 100000, 265000,
115000, 180000, 45000, 50500, 270000, 290000, 325000,
160000, 350000, 290000, 290000, 465000, 325000, 510000,

```
       860000,  450000,  125000,  500001,   95000,  250000,  110000,
       270000,  110000,  300000,  260000,  390000])
```

In [92]:
```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 1/4, random_st
```

In [93]:
```
model=LinearRegression()
```

In [ ]:
```

```

In [95]:
```
data['name'].unique()
```

Out[95]:
```
array(['HyundaiSantroXing', 'MahindraJeepCL550', 'HyundaiGrandi10',
       'FordEcoSportTitanium', 'FordFigo', 'HyundaiEon',
       'FordEcoSportAmbiente', 'MarutiSuzukiAlto', 'SkodaFabiaClassic',
       'MarutiSuzukiStingray', 'HyundaiElitei20', 'MahindraScorpioSLE',
       'AudiA8', 'AudiQ7', 'MahindraScorpioS10', 'Hyundaii20Sportz',
       'MarutiSuzukiVitara', 'MahindraBoleroDI', 'MarutiSuzukiSwift',
       'MarutiSuzukiWagon', 'ToyotaInnova2.0', 'RenaultLodgy85',
       'SkodaYetiAmbition', 'MarutiSuzukiBaleno', 'RenaultDuster110',
       'RenaultDuster85', 'HondaCity1.5', 'MarutiSuzukiDzire',
       'HondaAmaze', 'HondaAmaze1.5', 'HondaCity', 'DatsunRediGO',
       'MarutiSuzukiSX4', 'MitsubishiPajeroSport', 'HondaCityZX',
       'TataIndigoeCS', 'VolkswagenPoloHighline', 'ChevroletSparkLS',
       'RenaultDuster110PS', 'MiniCooperS', 'SkodaFabia1.2L',
       'RenaultDuster', 'MahindraScorpioS4', 'MahindraScorpioVLX',
       'MahindraQuantoC8', 'FordEcoSport', 'HondaBrio',
       'VolkswagenVentoHighline', 'Hyundaii20Magna', 'ToyotaCorollaAltis',
       'HyundaiVernaTransform', 'BMW3Series', 'MarutiSuzukiA',
       'ToyotaEtiosGD', 'FordFigoDiesel', 'ChevroletBeatLT', 'BMW7Series',
       'MahindraXUV500W8', 'Hyundaii10Magna', 'HyundaiVernaFluidic',
       'MarutiSuzukiErtiga', 'HondaAmaze1.2', 'Hyundaii20Asta',
       'MarutiSuzukiEeco', 'MarutiSuzukiEsteem', 'MarutiSuzukiRitz',
       'ToyotaEtiosLiva', 'ChevroletSpark', 'NissanMicraXV',
       'ChevroletBeat', 'ToyotaCorolla', 'FordEcoSportTrend',
       'TataIndicaV2', 'HindustanMotorsAmbassador', 'ToyotaInnova2.5',
       'VolkswagenJettaHighline', 'VolkswagenPoloComfortline',
       'VolkswagenPolo', 'MahindraScorpio', 'NissanSunny', 'RenaultKwid',
       'ChevroletSparkLT', 'FiatPuntoEmotion', 'Hyundaii10Sportz',
       'ChevroletBeatLS', 'TataIndigoCS', 'HyundaiEonEra',
       'MahindraXUV500', 'FordFiesta', 'Hyundaii20',
       'HyundaiFluidicVerna', 'FiatPetraELX', 'MarutiSuzukiCiaz',
       'MarutiSuzukiZen', 'HyundaiCreta1.6', 'MahindraScorpioSLX',
       'TataNanoCx', 'TataSumoVicta', 'VolkswagenPassatDiesel',
       'RenaultScalaRxL', 'Hyundaii20Active', 'MahindraXyloE4',
       'MahindraJeepMM', 'MahindraBoleroSLE', 'ForceMotorsForce',
       'ToyotaEtios', 'HondaCityVX', 'MahindraTharCRDe', 'AudiA41.8',
       'MercedesBenzGLA', 'LandRoverFreelander', 'RenaultKwidRXT',
       'TataAriaPleasure', 'MercedesBenzB', 'DatsunGOT', 'HondaJazzVX',
       'ChevroletTaveraNeo', 'HyundaiEonSportz', 'TataSumoGold',
       'ChevroletEnjoy1.4', 'NissanTerranoXL', 'MarutiSuzukiMaruti',
       'RenaultKwid1.0', 'HyundaiAccentGLX', 'MahindraTUV300T4',
       'HondaAccord', 'MahindraScorpio2.6', 'HondaMobilio', 'SkodaLaura',
       'TataManzaAura', 'ChevroletSailUVA', 'AudiA42.0',
       'HyundaiElantraSX', 'MahindraKUV100K8', 'Hyundaii10',
       'HyundaiAccent', 'HyundaiVerna', 'ToyotaFortuner',
       'MahindraBoleroPower', 'SkodaRapidElegance', 'TataVistaQuadrajet',
       'ChevroletBeatDiesel', 'HyundaiVerna1.4', 'MarutiSuzukiVersa',
       'TataIndigoLX', 'VolkswagenVentoKonekt', 'MercedesBenzC',
       'MarutiSuzukiOmni', 'HyundaiSonataTransform', 'HondaJazzS',
```

```
'MahindraScorpioW', 'HondaBrioV', 'MahindraTUV300T8',
'NissanXTrail', 'FordIkon1.3', 'ToyotaFortuner3.0',
'TataManzaELAN', 'MercedesBenzA', 'TataIndigoLS',
'HyundaiVerna1.6', 'BMW5Series', 'SkodaSuperb1.8', 'AudiQ32.0',
'FordFigoDuratorq', 'MahindraLoganDiesel', 'TataNanoGenX',
'HondaCitySV', 'FordFigoPetrol', 'ToyotaCorollaH2',
'HyundaiXcentBase', 'HyundaiAccentExecutive', 'TataZestXE',
'MahindraXUV500W6', 'TataTigorRevotron', 'MarutiSuzuki800',
'HondaMobilioS', 'TataIndica', 'HondaBrioVX', 'TataNanoLx',
'JaguarXEXE', 'HyundaiEonMagna', 'HyundaiEonD',
'MarutiSuzukiEstilo', 'MahindraScorpioVlx', 'MitsubishiLancer1.8',
'FordFiestaSXi', 'AudiA62.0', 'HyundaiGetzPrime', 'HyundaiSantro',
'ChevroletBeatPS', 'BMWX1xDrive20d', 'TataNano',
'ChevroletCruzeLTZ', 'MahindraXUV500W10', 'HyundaiAccentGLE',
'ForceMotorsOne', 'ChevroletSpark1.0', 'RenaultDuster85PS',
'ChevroletEnjoy', 'JeepWranglerUnlimited', 'HyundaiVernaVGT',
'MarutiSuzukiCelerio', 'TataZestQuadrajet', 'Hyundaii10Era',
'TataIndigoMarina', 'HyundaiXcentSX', 'TataNanoLX',
'MahindraXyloE8', 'TataManzaAqua', 'TataVentureEX',
'SkodaOctaviaClassic', 'FordIkon1.6', 'NissanSunnyXL',
'VolkswagenPoloTrendline', 'HyundaiElantra1.8', 'TataIndicaeV2',
'JaguarXF2.2', 'AudiQ52.0', 'BMWX1sDrive20d', 'MarutiSuzukiS',
'VolkswagenVentoComfortline', 'MahindraKUV100',
'VolkswagenJettaComfortline', 'VolvoS80Summum', 'BMWX1',
'RenaultDusterRxL', 'HondaWRV', 'MahindraScorpioLX',
'AudiA3Cabriolet', 'HyundaiSantroAE', 'MahindraXyloD2',
'HyundaiGetzGLE', 'NissanMicraXL', 'ChevroletTaveraLS',
'TataTiagoRevotron', 'TataTiagoRevotorq', 'FordFusion1.4',
'FiatLineaEmotion', 'TataSumoGrande', 'VolkswagenPoloHighline1.2L',
'HyundaiCreta', 'TataBoltXM', 'DatsunGoPlus', 'FordEndeavor4x4',
'MahindraLogan', 'ChevroletSail1.2', 'TataManza', 'ToyotaEtiosG',
'ToyotaQualis', 'MahindraQuantoC4', 'Hyundaii20Select',
'HyundaiGetz', 'SkodaFabia', 'TataZestXM'], dtype=object)
```

In [97]:

```
data['name']=data['name'].astype(category)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_14272/3720065683.py in <module>
----> 1 data['name']=data['name'].astype(category)

NameError: name 'category' is not defined
```

In [ ]: