

# Used Car Price Prediction

## Life cycle of Machine learning Project

- Understanding the Problem Statement
- Data Collection
- Exploratory data analysis
- Data Cleaning
- Data Pre-Processing
- Model Training
- Choose best model ## 1) Problem statement.
- This dataset comprises used cars sold on cardehko.com in India as well as important features of these cars.
- If user can predict the price of the car based on input features. Prediction results can be used to give new seller the price suggestion based on market condition. ## 2) Data Collection.
- The Dataset is collected from scrapping from cardheko website
- The data consists of 13 column and 15411 rows. ### 2.1 Import Data and Required Packages ### Importing Pandas, Numpy, Matplotlib, Seaborn and Warings Library.

```
In [70]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import warnings
from six.moves import urllib
warnings.filterwarnings("ignore")
%matplotlib inline
```

## Download and Import the CSV Data as Pandas DataFrame

```
In [71]: df = pd.read_csv("D:/I neuron/car_dekho.csv")
```

```
In [72]: df.head()
```

	car_name	brand	model	vehicle_age	km_driven	seller_type	fuel_type	transmission_type	mil
0	Maruti Alto	Maruti	Alto	9	120000	Individual	Petrol	Manual	
1	Hyundai Grand	Hyundai	Grand	5	20000	Individual	Petrol	Manual	
2	Hyundai i20	Hyundai	i20	11	60000	Individual	Petrol	Manual	
3	Maruti Alto	Maruti	Alto	9	37000	Individual	Petrol	Manual	
4	Ford Ecosport	Ford	Ecosport	6	30000	Dealer	Diesel	Manual	

In [8]: `df.shape`

Out[8]: (15411, 13)

## summary of the given dataset

In [9]: `df.describe()`

	vehicle_age	km_driven	mileage	engine	max_power	seats	selling_price
count	15411.000000	1.541100e+04	15411.000000	15411.000000	15411.000000	15411.000000	1.541100e+06
mean	6.036338	5.561648e+04	19.701151	1486.057751	100.588254	5.325482	7.749711e+06
std	3.013291	5.161855e+04	4.171265	521.106696	42.972979	0.807628	8.941284e+06
min	0.000000	1.000000e+02	4.000000	793.000000	38.400000	0.000000	4.000000e+06
25%	4.000000	3.000000e+04	17.000000	1197.000000	74.000000	5.000000	3.850000e+06
50%	6.000000	5.000000e+04	19.670000	1248.000000	88.500000	5.000000	5.560000e+06
75%	8.000000	7.000000e+04	22.700000	1582.000000	117.300000	5.000000	8.250000e+06
max	29.000000	3.800000e+06	33.540000	6592.000000	626.000000	9.000000	3.950000e+07

## checking of data types

In [10]: `df.info()`

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15411 entries, 0 to 15410
Data columns (total 13 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   car_name         15411 non-null   object  
 1   brand            15411 non-null   object  
 2   model            15411 non-null   object  
 3   vehicle_age      15411 non-null   int64  
 4   km_driven        15411 non-null   int64  
 5   seller_type       15411 non-null   object  
 6   fuel_type         15411 non-null   object  
 7   transmission_type 15411 non-null   object  
 8   mileage           15411 non-null   float64 
 9   engine            15411 non-null   int64  
 10  max_power         15411 non-null   float64 
 11  seats             15411 non-null   int64  
 12  selling_price     15411 non-null   int64  
dtypes: float64(2), int64(5), object(6)
memory usage: 1.5+ MB
```

### 3. EXPLORING DATA

```
In [17]: # define numerical & categorical columns
numeric_features = [feature for feature in df.columns if df[feature].dtype != 'O']
categorical_features = [feature for feature in df.columns if df[feature].dtype == 'O']

# print columns
print('We have {} numerical features : {}'.format(len(numeric_features), numeric_features))
print('\nWe have {} categorical features : {}'.format(len(categorical_features), categorical_features))

We have 7 numerical features : ['vehicle_age', 'km_driven', 'mileage', 'engine', 'max_power', 'seats', 'selling_price']

We have 6 categorical features : ['car_name', 'brand', 'model', 'seller_type', 'fuel_type', 'transmission_type']
```

### Feature Information

car\_name: Car's Full name, which includes brand and specific model name.

brand: Brand Name of the particular car.

model: Exact model name of the car of a particular brand.

seller\_type: Which Type of seller is selling the used car

fuel\_type: Fuel used in the used car, which was put up on sale.

transmission\_type: Transmission used in the used car, which was put on sale.

vehicle\_age: The count of years since car was bought.

mileage: It is the number of kilometer the car runs per litre.

engine: It is the engine capacity in cc(cubic centimeters)

max\_power: Max power it produces in BHP.

seats: Total number of seats in car.

selling\_price: The sale price which was put up on website.

```
In [18]: # proportion of count data on categorical columns
for col in categorical_features:
    print(df[col].value_counts(normalize=True) * 100)
    print('-----')
```

```
Hyundai i20      5.878918
Maruti Swift Dzire 5.775096
Maruti Swift      5.067809
Maruti Alto       5.048342
Honda City        4.912076
...
Mercedes-AMG C   0.006489
Tata Altroz       0.006489
Ferrari GTC4Lusso 0.006489
Hyundai Aura      0.006489
Force Gurkha     0.006489
Name: car_name, Length: 121, dtype: float64
```

```
-----
Maruti          32.392447
Hyundai         19.349815
Honda           9.635974
Mahindra        6.560249
Toyota          5.145675
Ford             5.126209
Volkswagen      4.023100
Renault          3.478035
BMW              2.848615
Tata              2.790215
Mercedes-Benz    2.186750
Skoda            2.167283
Audi              1.245863
Datsun           1.103108
Jaguar           0.382843
Land Rover       0.330932
Jeep              0.266044
Kia              0.207644
Porsche          0.136266
Volvo            0.129777
MG                0.123289
Mini              0.110311
Nissan           0.071378
Lexus             0.064889
Isuzu            0.051911
Bentley          0.019467
Maserati          0.012978
ISUZU            0.012978
Ferrari          0.006489
Mercedes-AMG     0.006489
Rolls-Royce      0.006489
Force             0.006489
Name: brand, dtype: float64
```

```
-----
i20              5.878918
Swift Dzire     5.775096
Swift            5.067809
Alto              5.048342
City              4.912076
...
Ghibli           0.006489
Altroz           0.006489
GTC4Lusso        0.006489
Aura              0.006489
Gurkha           0.006489
Name: model, Length: 120, dtype: float64
```

```
Dealer           61.897346
Individual       36.980079
Trustmark Dealer 1.122575
Name: seller_type, dtype: float64
-----
Petrol          49.594446
Diesel          48.140938
CNG             1.953150
LPG              0.285510
Electric         0.025955
Name: fuel_type, dtype: float64
-----
Manual          79.326455
Automatic        20.673545
Name: transmission_type, dtype: float64
-----
```

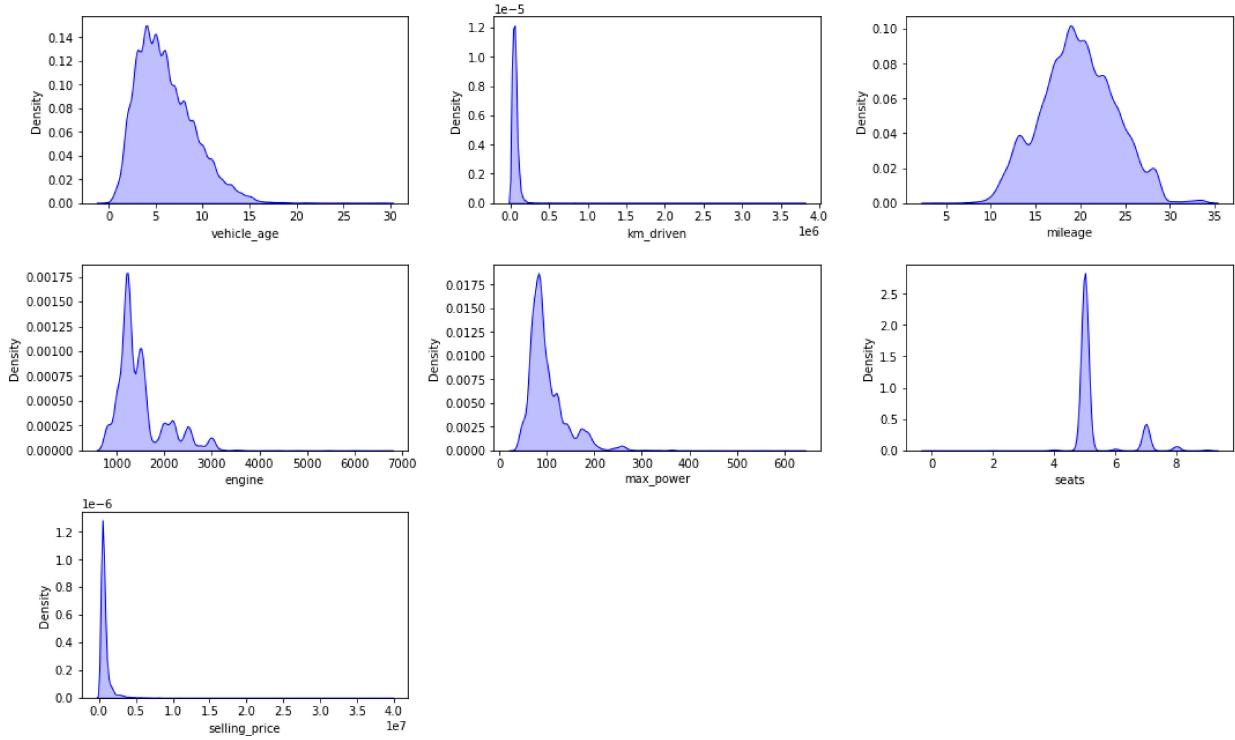
## Univariate Analysis

The term univariate analysis refers to the analysis of one variable prefix "uni" means "one." The purpose of univariate analysis is to understand the distribution of values for a single variable.

## Numerical features

```
In [19]: plt.figure(figsize=(15, 15))
plt.suptitle('Univariate Analysis of Numerical Features', fontsize=20, fontweight='bold')

for i in range(0, len(numeric_features)):
    plt.subplot(5, 3, i+1)
    sns.kdeplot(x=df[numeric_features[i]], shade=True, color='blue')
    plt.xlabel(numeric_features[i])
    plt.tight_layout()
```

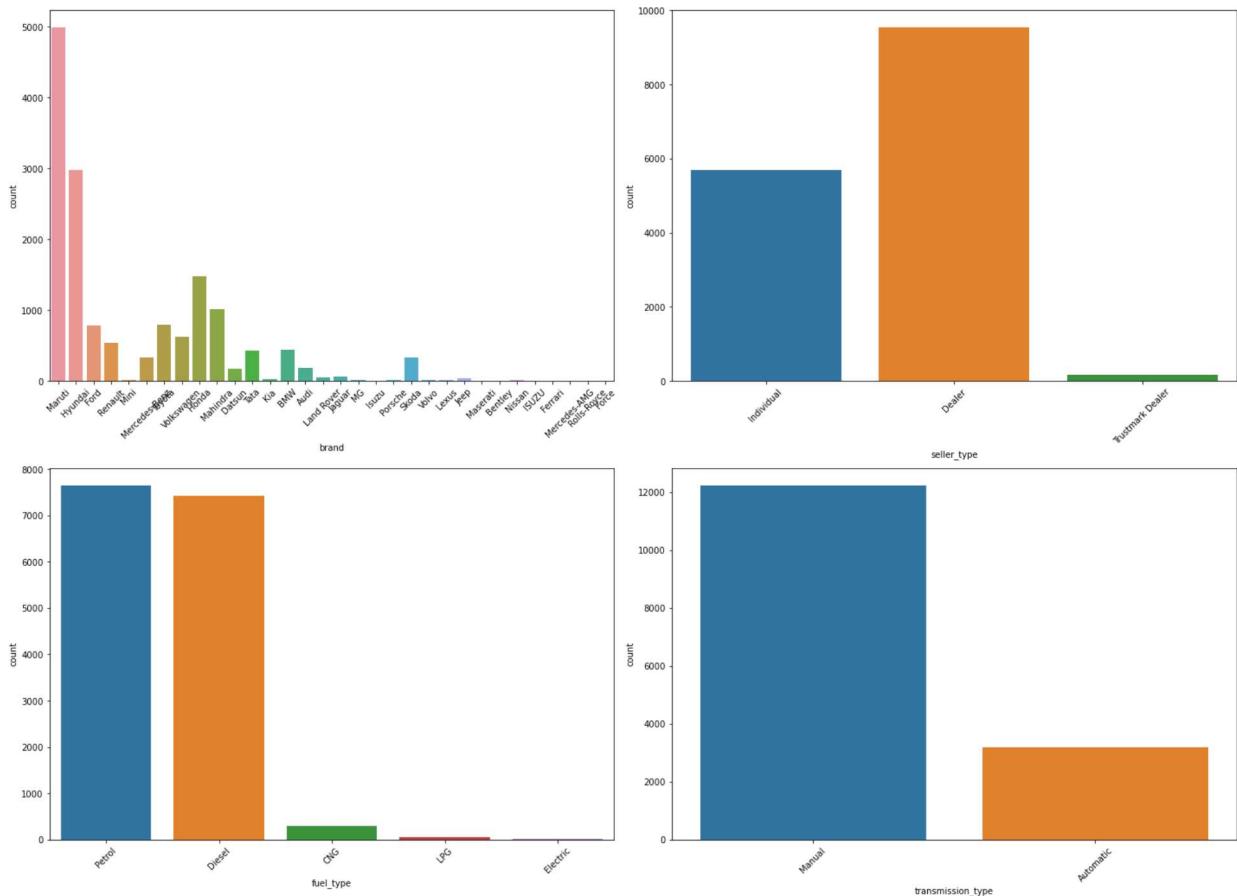
**Univariate Analysis of Numerical Features****Points Taken**

- Km\_driven, max\_power, selling\_price, and engine are right skewed and positively skewed.
- Outliers are found in Km\_driven, max\_power, selling\_price, and engine.

**Categorical Features**

In [21]:

```
# categorical columns
plt.figure(figsize=(20, 15))
plt.suptitle('Univariate Analysis of Categorical Features', fontsize=20, fontweight='bold')
cat1 = [ 'brand', 'seller_type', 'fuel_type', 'transmission_type']
for i in range(0, len(cat1)):
    plt.subplot(2, 2, i+1)
    sns.countplot(x=df[cat1[i]])
    plt.xlabel(cat1[i])
    plt.xticks(rotation=45)
    plt.tight_layout()
```

**Univariate Analysis of Categorical Features****Points taken**

- Maruti and hyundai are highest sellers
- selling at dealers end is highest
- Manual transmission vehicle is more

**Multivariate Analysis**

- Multivariate analysis is the analysis of more than one variable.

**Check Multicollinearity in Numerical features**

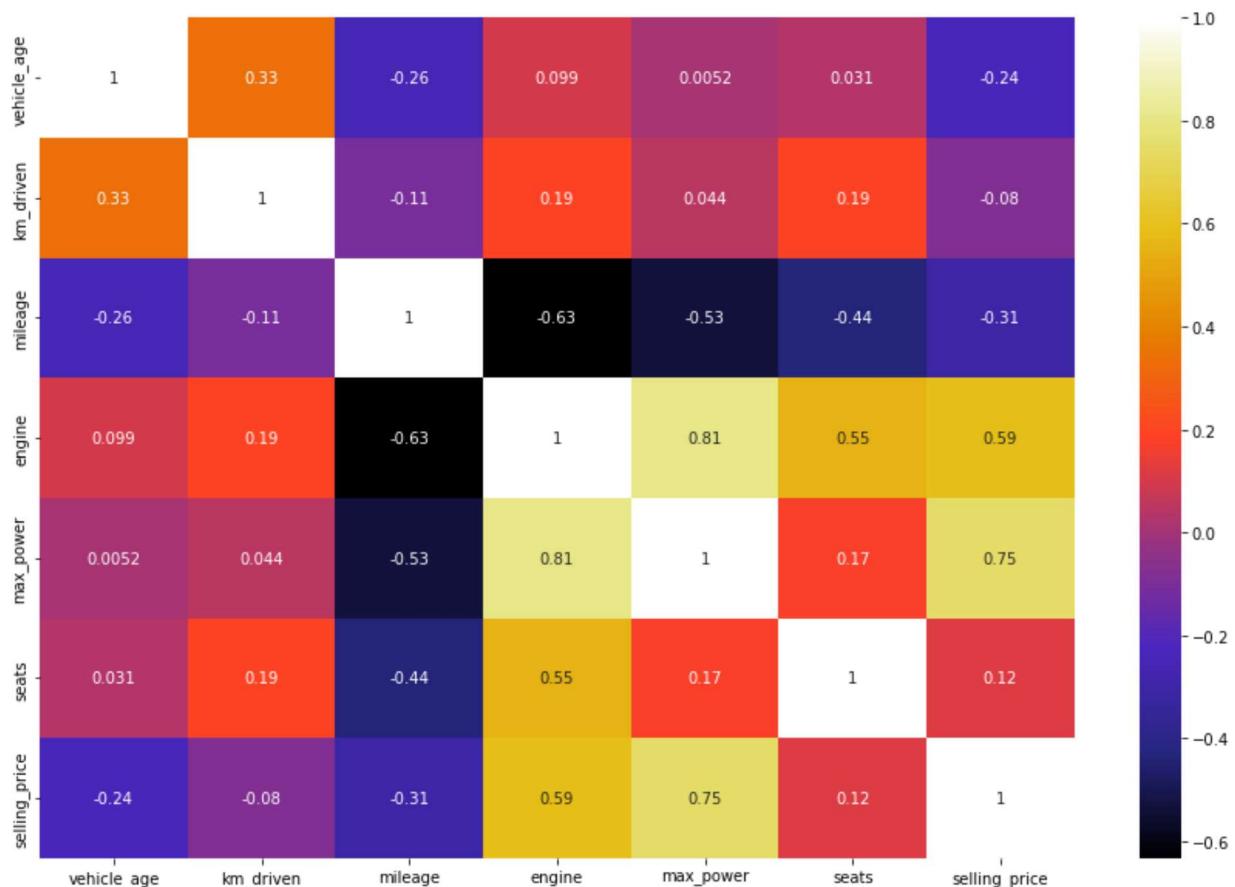
```
In [24]: df[(list(df.columns)[1:])].corr()
```

Out[24]:

	vehicle_age	km_driven	mileage	engine	max_power	seats	selling_price
vehicle_age	1.000000	0.333891	-0.257394	0.098965	0.005208	0.030791	-0.241851
km_driven	0.333891	1.000000	-0.105239	0.192885	0.044421	0.192830	-0.080030
mileage	-0.257394	-0.105239	1.000000	-0.632987	-0.533128	-0.440280	-0.305549
engine	0.098965	0.192885	-0.632987	1.000000	0.807368	0.551236	0.585844
max_power	0.005208	0.044421	-0.533128	0.807368	1.000000	0.172257	0.750236
seats	0.030791	0.192830	-0.440280	0.551236	0.172257	1.000000	0.115033
selling_price	-0.241851	-0.080030	-0.305549	0.585844	0.750236	0.115033	1.000000

In [25]:

```
plt.figure(figsize = (15,10))
sns.heatmap(df.corr(), cmap="CMRmap", annot=True)
plt.show()
```



## Points taken

- Our target column mileage has a weak negative correlation with engine.
- The max\_power has a strong positive correlation with engine.

## Check Multicollinearity for Categorical features

- A chi-squared test (also chi-square or  $\chi^2$  test) is a statistical hypothesis test that is valid to perform when the test statistic is chi-squared distributed under the null hypothesis,

specifically Pearson's chi-squared test

- A chi-square statistic is one way to show a relationship between two categorical variables.
- Here we test correlation of Categorical columns with Target column i.e Selling Price

```
In [26]: from scipy.stats import chi2_contingency
chi2_test = []
for feature in categorical_features:
    if chi2_contingency(pd.crosstab(df['selling_price'], df[feature]))[1] < 0.05:
        chi2_test.append('Reject Null Hypothesis')
    else:
        chi2_test.append('Fail to Reject Null Hypothesis')
result = pd.DataFrame(data=[categorical_features, chi2_test]).T
result.columns = ['Column', 'Hypothesis Result']
result
```

	Column	Hypothesis Result
0	car_name	Reject Null Hypothesis
1	brand	Reject Null Hypothesis
2	model	Reject Null Hypothesis
3	seller_type	Reject Null Hypothesis
4	fuel_type	Reject Null Hypothesis
5	transmission_type	Reject Null Hypothesis

## Checking Null Values

```
In [27]: df.isnull().sum()
```

```
Out[27]: car_name      0
brand          0
model          0
vehicle_age    0
km_driven      0
seller_type    0
fuel_type       0
transmission_type 0
mileage         0
engine          0
max_power       0
seats           0
selling_price   0
dtype: int64
```

```
In [28]: continues_features=[feature for feature in numeric_features if len(df[feature].unique())
print('Num of continues features :',continues_features)
```

Num of continues features : ['vehicle\_age', 'km\_driven', 'mileage', 'engine', 'max\_power', 'selling\_price']

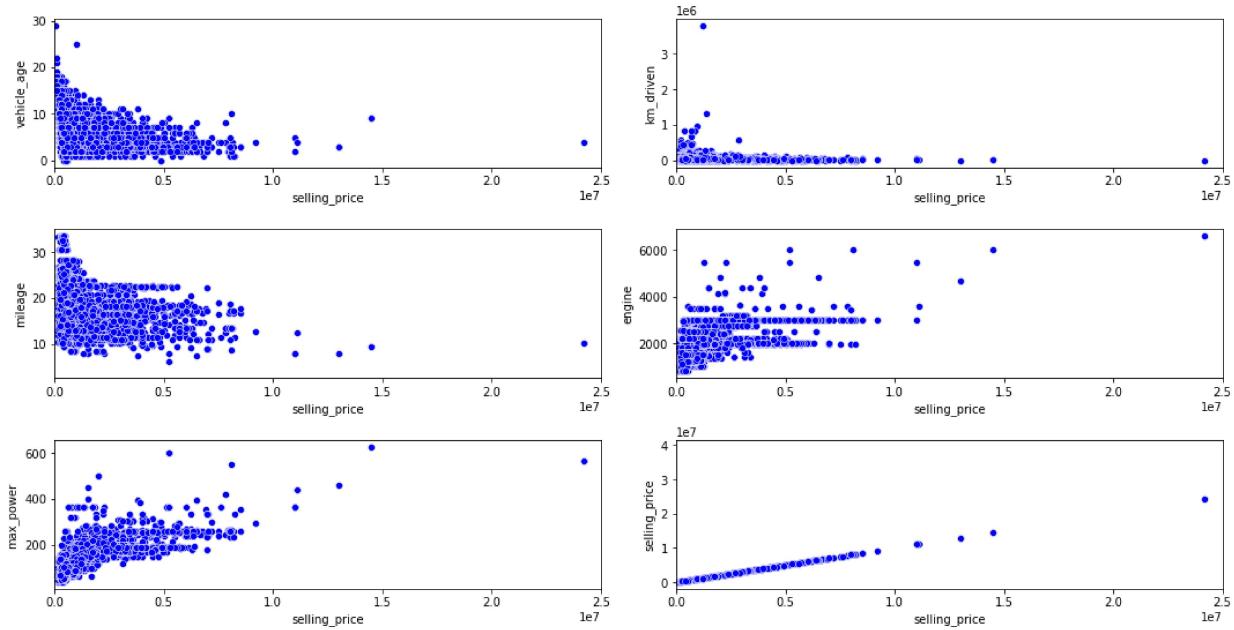
```
In [30]: fig = plt.figure(figsize=(15, 20))
```

```

for i in range(0, len(continues_features)):
    ax = plt.subplot(8, 2, i+1)

    sns.scatterplot(data= df ,x='selling_price', y=continues_features[i], color='b')
    plt.xlim(0,25000000) # Limit to 20 Lakhs Rupees to view clean
    plt.tight_layout()

```



## Initial Analysis Report

### Report

- Lower Vehicle age has more selling price than Vehicle with more age.
- Engine CC has positive effect on price, Vehicle with 2000 cc and below are mostly priced below 5Lakh.
- Kms Driven has negative effect on selling price.

## 4. Visualization

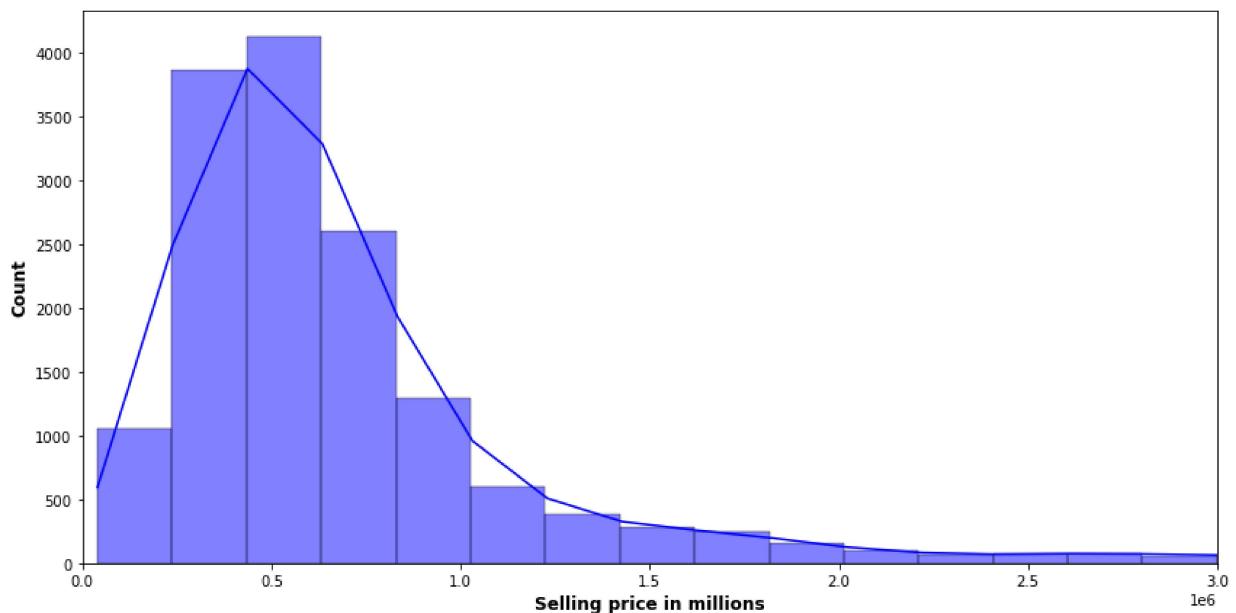
### 4.1 Visualize the Target Feature

```

In [31]: plt.subplots(figsize=(14,7))
sns.histplot(df.selling_price, bins=200, kde=True, color = 'b')
plt.title("Selling Price Distribution", weight="bold", fontsize=20, pad=20)
plt.ylabel("Count", weight="bold", fontsize=12)
plt.xlabel("Selling price in millions", weight="bold", fontsize=12)
plt.xlim(0,3000000)
plt.show()

```

## Selling Price Distribution



- From the chart it is clear that the Target Variable Skewed
- ### 4.2 Most Selling car in Used car website?

```
In [33]: df.car_name.value_counts()[0:10]
```

```
Out[33]:
```

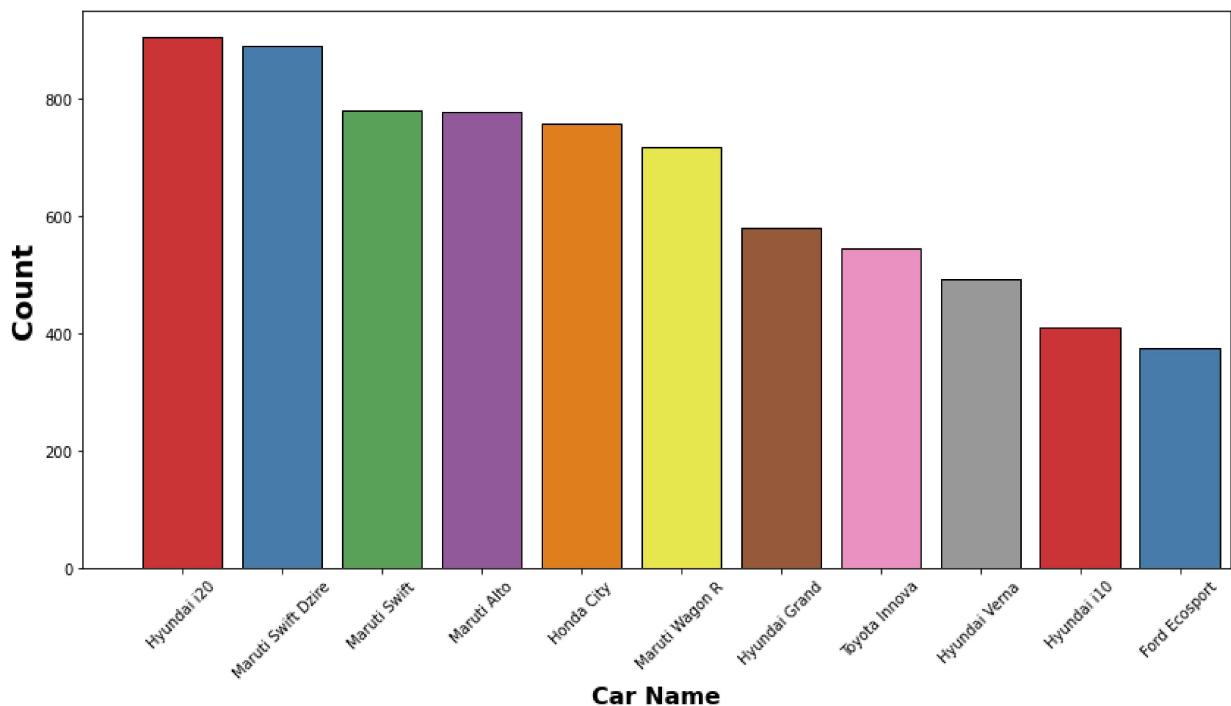
Hyundai i20	906
Maruti Swift Dzire	890
Maruti Swift	781
Maruti Alto	778
Honda City	757
Maruti Wagon R	717
Hyundai Grand	580
Toyota Innova	545
Hyundai Verna	492
Hyundai i10	410

Name: car\_name, dtype: int64

## Point Taken

- Most selling car is Hyundai i20

```
In [35]: plt.subplots(figsize=(14,7))
sns.countplot(x="car_name", data=df, ec = "black", palette="Set1", order = df['car_name']
plt.title("Top 10 Most Sold Car", weight="bold", fontsize=20, pad=20)
plt.ylabel("Count", weight="bold", fontsize=20)
plt.xlabel("Car Name", weight="bold", fontsize=16)
plt.xticks(rotation= 45)
plt.xlim(-1,10.5)
plt.show()
```

**Top 10 Most Sold Car**

**Check mean price of Hyundai i20 which is most sold**

```
In [36]: i20 = df[df['car_name'] == 'Hyundai i20']['selling_price'].mean()
print(f'The mean price of Hyundai i20 is {i20:.2f} Rupees')
```

The mean price of Hyundai i20 is 543603.75 Rupees

### Point Taken:

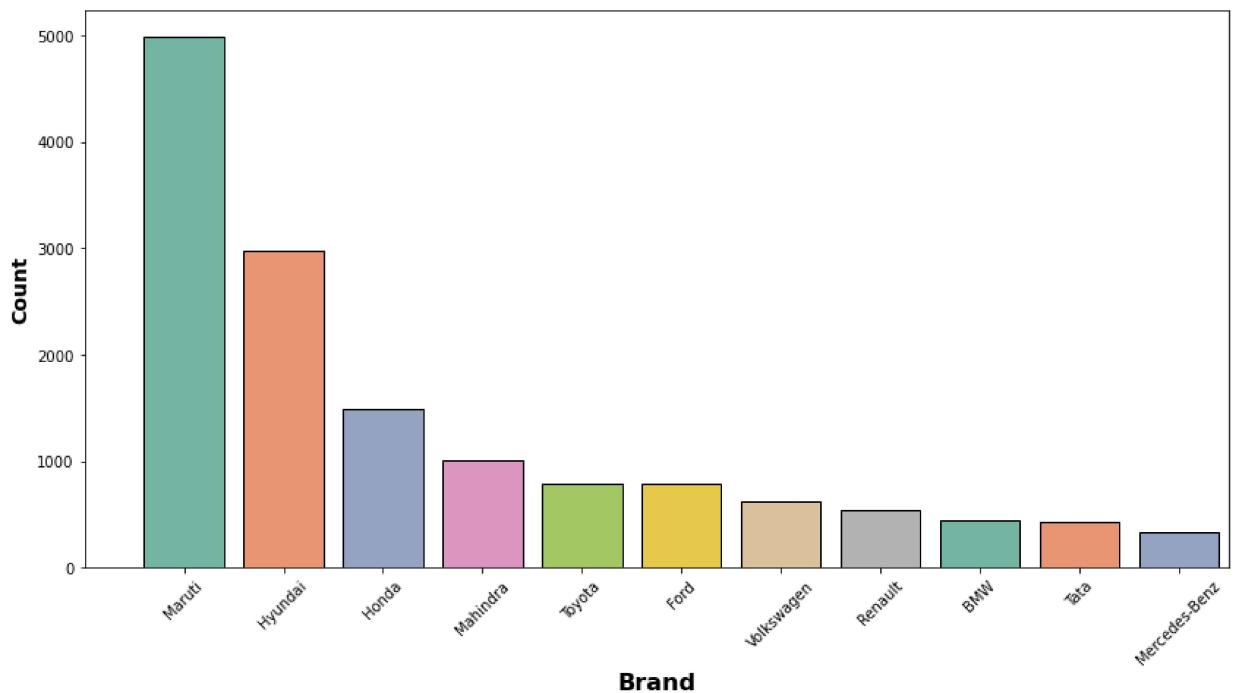
- As per the Chart these are top 10 most selling cars in used car website.
- Of the total cars sold Hyundai i20 shares 5.8% of total ads posted and followed by Maruti Swift Dzire.
- Mean Price of Most Sold Car is 5.4 lakhs.
- This Feature has impact on the Target Variable.

### Most selling brand

```
In [38]: df.brand.value_counts()[0:10]
```

```
Out[38]: Maruti      4992
          Hyundai     2982
          Honda       1485
          Mahindra    1011
          Toyota      793
          Ford        790
          Volkswagen  620
          Renault     536
          BMW         439
          Tata        430
Name: brand, dtype: int64
```

```
In [39]: plt.subplots(figsize=(14,7))
sns.countplot(x="brand", data=df, ec = "black", palette="Set2", order = df['brand'].value_counts().index)
plt.title("Top 10 Most Sold Brand", weight="bold", fontsize=20, pad=20)
plt.ylabel("Count", weight="bold", fontsize=14)
plt.xlabel("Brand", weight="bold", fontsize=16)
plt.xticks(rotation= 45)
plt.xlim(-1,10.5)
plt.show()
```

**Top 10 Most Sold Brand**

**Check the Mean price of Maruti brand which is most sold**

```
In [41]: maruti = df[df['brand'] == 'Maruti']['selling_price'].mean()
print(f'The mean price of Maruti is {maruti:.3f} Rupees')
```

The mean price of Maruti is 487089.318 Rupees

### Report:

- As per the Chart Maruti has the most share of Ads in Used car website and Maruti is the most sold brand.
- Following Maruti we have Hyundai and Honda.
- Mean Price of Maruti Brand is 4.8 lakhs.

### Costliest Brand and Costliest Car

```
In [44]: brand = df.groupby('brand').selling_price.max()
brand_df = brand.to_frame().sort_values('selling_price', ascending=False)[0:10]
brand_df
```

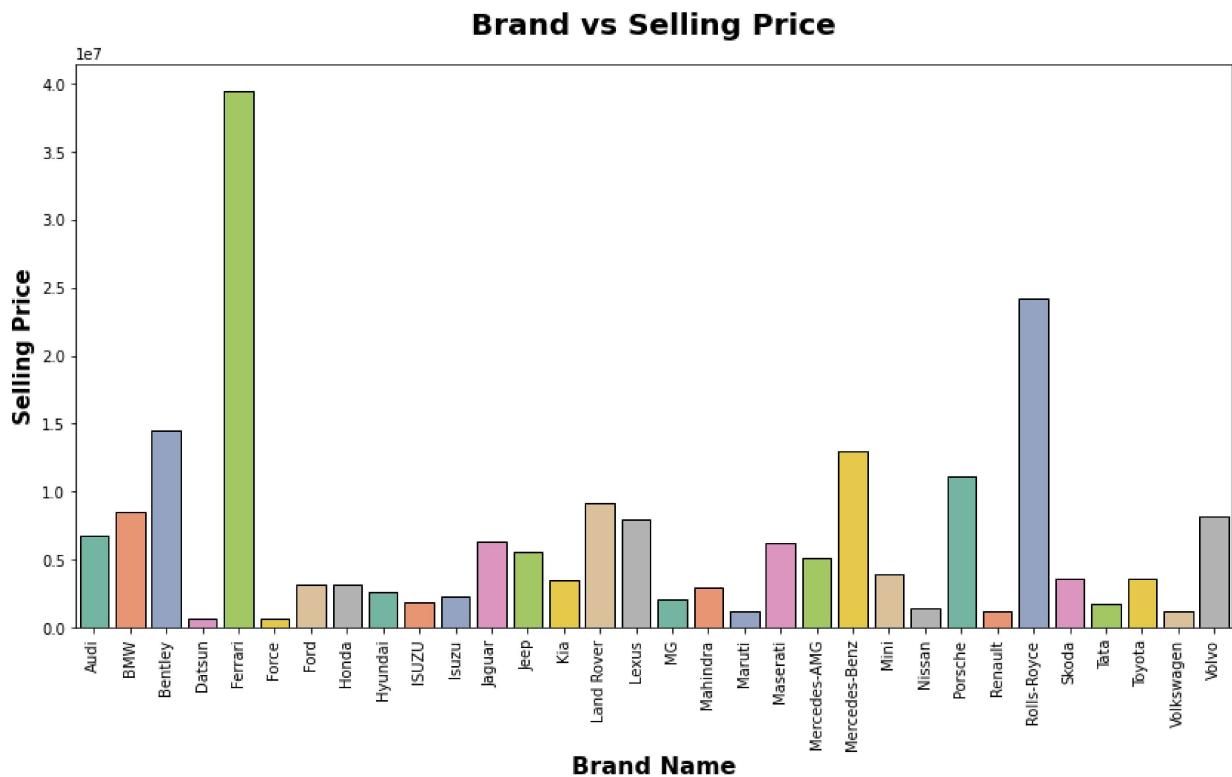
Out[44]:

**selling\_price**

brand	selling_price
Ferrari	39500000
Rolls-Royce	24200000
Bentley	14500000
Mercedes-Benz	13000000
Porsche	11100000
Land Rover	9200000
BMW	8500000
Volvo	8195000
Lexus	8000000
Audi	6800000

In [45]:

```
plt.subplots(figsize=(14,7))
sns.barplot(x=brand.index, y=brand.values, ec = "black", palette="Set2")
plt.title("Brand vs Selling Price", weight="bold", fontsize=20, pad=20)
plt.ylabel("Selling Price", weight="bold", fontsize=15)
plt.xlabel("Brand Name", weight="bold", fontsize=16)
plt.xticks(rotation=90)
plt.show()
```

**Point Taken:**

- Costliest Brand sold is Ferrari at 3.95 Crores.

- Second most costliest car Brand is Rolls-Royce as 2.42 Crores.
- Brand name has very clear impact on selling price.

## Costliest Car

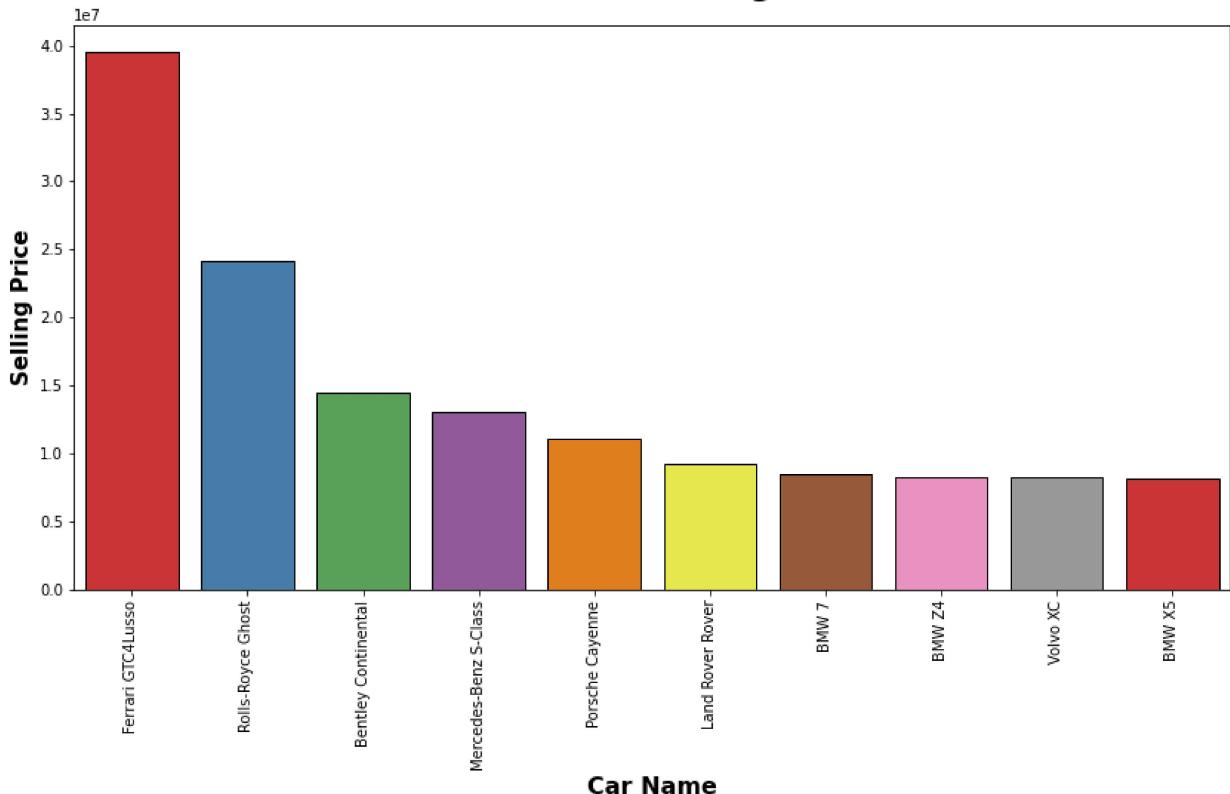
```
In [46]: car= df.groupby('car_name').selling_price.max()
car =car.to_frame().sort_values('selling_price',ascending=False)[0:10]
car
```

Out[46]:

car_name	selling_price
Ferrari GTC4Lusso	39500000
Rolls-Royce Ghost	24200000
Bentley Continental	14500000
Mercedes-Benz S-Class	13000000
Porsche Cayenne	11100000
Land Rover Rover	9200000
BMW 7	8500000
BMW Z4	8250000
Volvo XC	8195000
BMW X5	8100000

```
In [47]: plt.subplots(figsize=(14,7))
sns.barplot(x=car.index, y=car.selling_price,ec = "black",palette="Set1")
plt.title("Car Name vs Selling Price", weight="bold", fontsize=20, pad=20)
plt.ylabel("Selling Price", weight="bold", fontsize=15)
plt.xlabel("Car Name", weight="bold", fontsize=16)
plt.xticks(rotation=90)
plt.show()
```

### Car Name vs Selling Price



### Point Taken

- Costliest Car sold is Ferrari GTC4 Lusso followed by Rolls Royce Ghost.
- Ferrari selling price is 3.95 Crs.
- Other than Ferrari other car has priced below 1.5cr.

### Most Mileage Brand and Car Name

```
In [49]: mileage = df.groupby('brand')['mileage'].mean().sort_values(ascending=False).head(15)
mileage.to_frame()
```

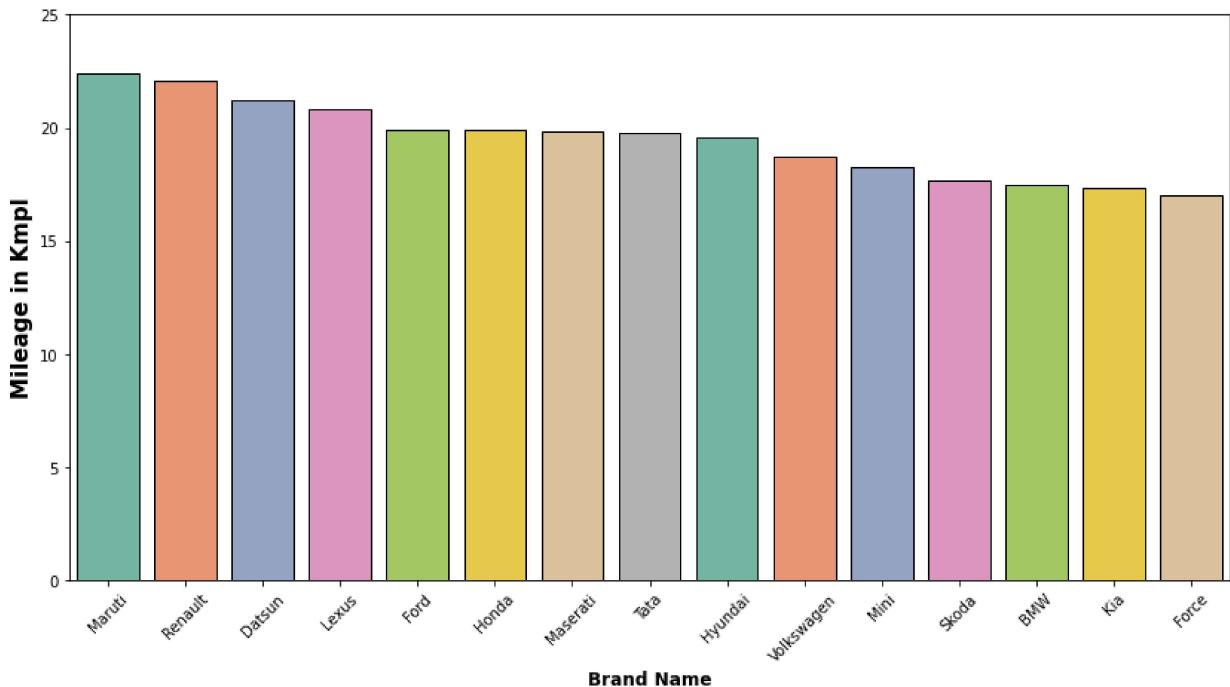
Out[49]:

**mileage**

brand
<b>Maruti</b> 22.430980
<b>Renault</b> 22.099142
<b>Datsun</b> 21.215647
<b>Lexus</b> 20.846000
<b>Ford</b> 19.922620
<b>Honda</b> 19.908795
<b>Maserati</b> 19.820000
<b>Tata</b> 19.755279
<b>Hyundai</b> 19.588776
<b>Volkswagen</b> 18.689774
<b>Mini</b> 18.287647
<b>Skoda</b> 17.667006
<b>BMW</b> 17.440182
<b>Kia</b> 17.323125
<b>Force</b> 17.000000

```
In [50]: plt.subplots(figsize=(14,7))
sns.barplot(x= mileage.index, y=mileage.values, ec = "black", palette="Set2")
plt.title("Brand vs Mileage", weight="bold", fontsize=20, pad=20)
plt.ylabel("Mileage in Kmpl", weight="bold", fontsize=15)
plt.xlabel("Brand Name", weight="bold", fontsize=12)
plt.ylim(0,25)
plt.xticks(rotation=45)
plt.show()
```

### Brand vs Mileage



### Car with Highest Mileage

```
In [51]: mileage_C = df.groupby('car_name')['mileage'].mean().sort_values(ascending=False).head()
mileage_C.to_frame()
```

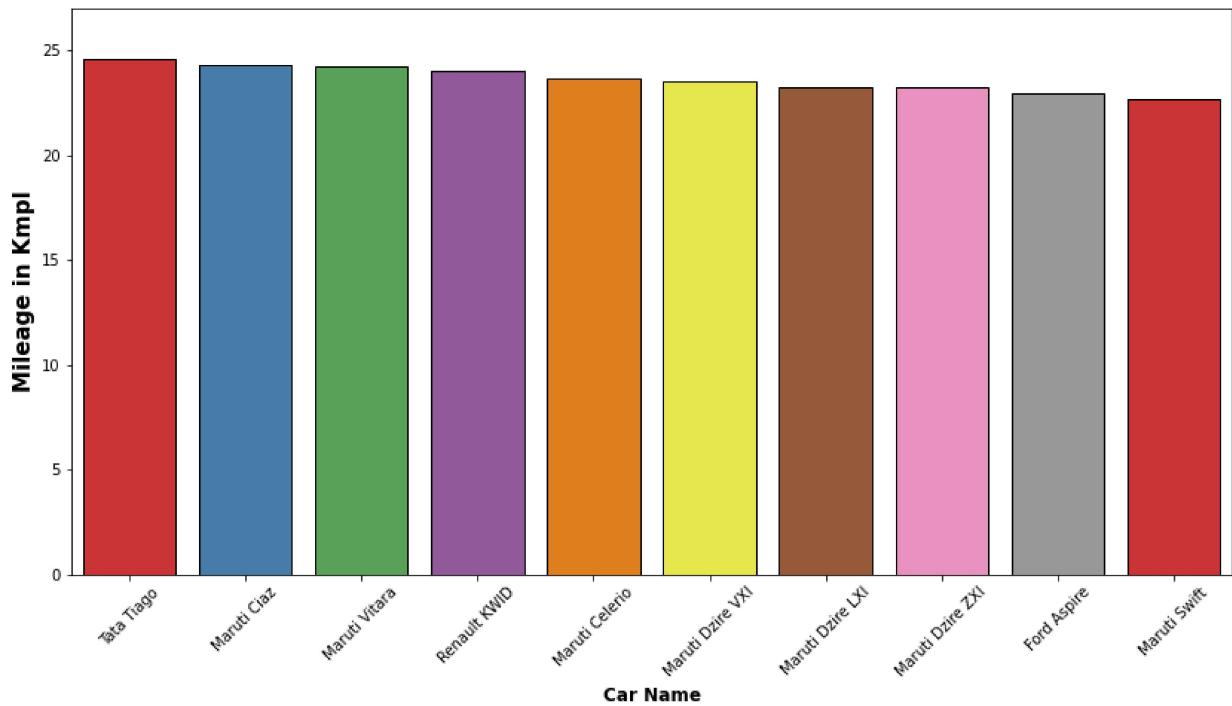
Out[51]:

#### mileage

car_name	
<b>Tata Tiago</b>	24.625103
<b>Maruti Ciaz</b>	24.289046
<b>Maruti Vitara</b>	24.231932
<b>Renault KWID</b>	24.037810
<b>Maruti Celerio</b>	23.703502
<b>Maruti Dzire VXI</b>	23.512941
<b>Maruti Dzire LXI</b>	23.260000
<b>Maruti Dzire ZXI</b>	23.260000
<b>Ford Aspire</b>	22.993846
<b>Maruti Swift</b>	22.719910

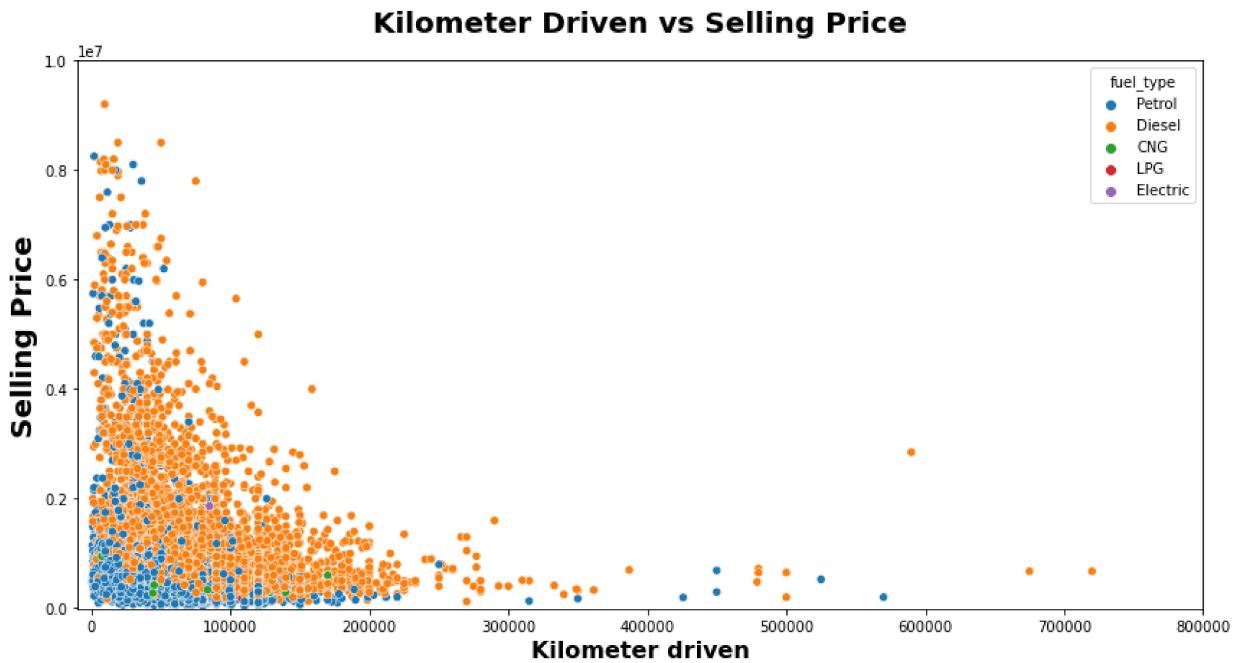
```
In [52]: plt.subplots(figsize=(14,7))
sns.barplot(x=mileage_C.index, y=mileage_C.values, ec = "black", palette="Set1")
plt.title("Car Name vs Mileage", weight="bold", fontsize=20, pad=20)
plt.ylabel("Mileage in Kmpl", weight="bold", fontsize=15)
plt.xlabel("Car Name", weight="bold", fontsize=12)
plt.ylim(0,27)
```

```
plt.xticks(rotation=45)  
plt.show()
```

**Car Name vs Mileage**

## Kilometer driven vs Selling Price

```
In [53]: plt.subplots(figsize=(14,7))  
sns.scatterplot(x="km_driven", y='selling_price', data=df, ec = "white", color='b', hue=  
plt.title("Kilometer Driven vs Selling Price", weight="bold", fontsize=20, pad=20)  
plt.ylabel("Selling Price", weight="bold", fontsize=20)  
plt.xlim(-10000,800000) #used limit for better visualization  
plt.ylim(-10000,10000000)  
plt.xlabel("Kilometer driven", weight="bold", fontsize=16)  
plt.show()
```



## Points taken

- Many Cars were sold with kms between 0 to 20k Kilometers
- Low Kms driven cars had more selling price compared to cars which had more kms driven.

## Fuel Type Selling Price

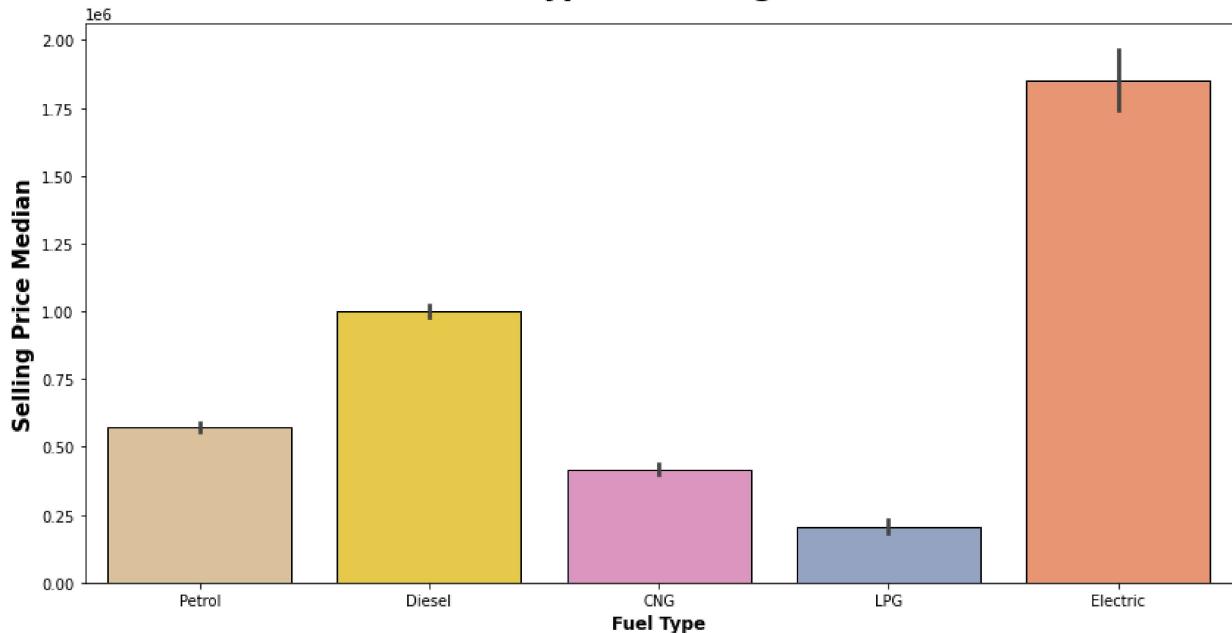
```
In [54]: fuel = df.groupby('fuel_type')['selling_price'].median().sort_values(ascending=False)
fuel.to_frame()
```

```
Out[54]:      selling_price
```

<b>fuel_type</b>	
<b>Electric</b>	1857500.0
<b>Diesel</b>	700000.0
<b>Petrol</b>	460000.0
<b>CNG</b>	370000.0
<b>LPG</b>	182500.0

```
In [55]: plt.subplots(figsize=(14,7))
sns.barplot(x=df.fuel_type, y=df.selling_price, ec = "black", palette="Set2_r")
plt.title("Fuel type vs Selling Price", weight="bold", fontsize=20, pad=20)
plt.ylabel("Selling Price Median", weight="bold", fontsize=15)
plt.xlabel("Fuel Type", weight="bold", fontsize=12)
plt.show()
```

### Fuel type vs Selling Price



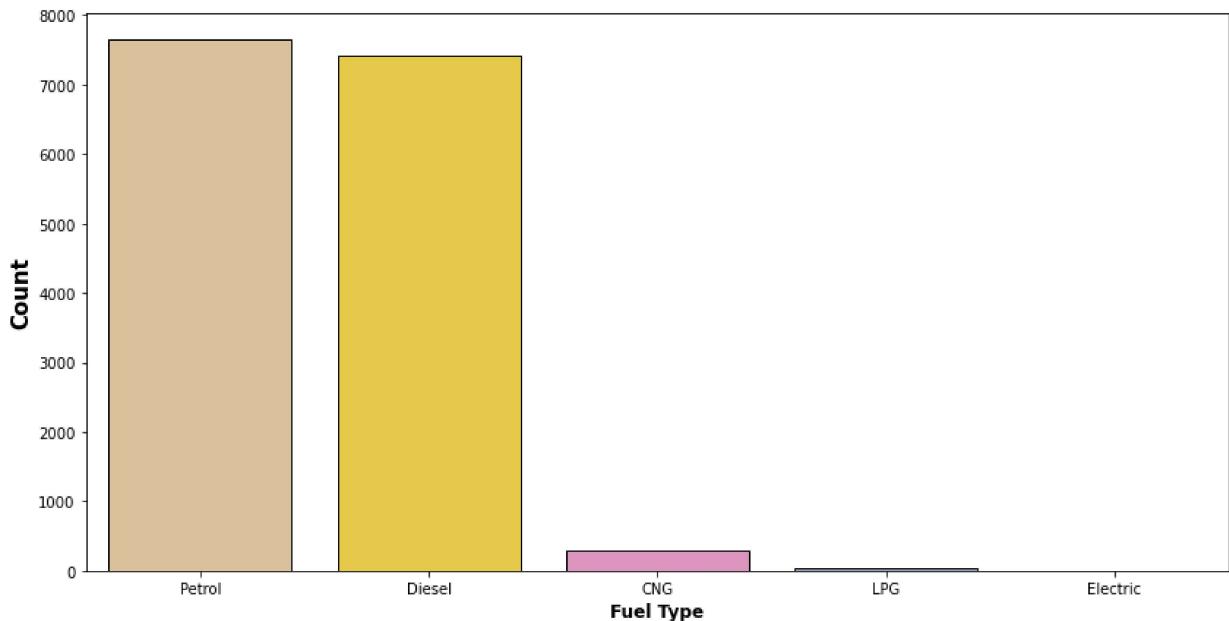
### Points taken

- Electric cars have higher selling average price.
- Followed by Diesel and Petrol.
- Fuel Type is also important feature for the Target variable

### Most sold Fuel type

```
In [56]: plt.subplots(figsize=(14,7))
sns.countplot(x=df.fuel_type, ec = "black", palette="Set2_r")
plt.title("Fuel Type Count", weight="bold", fontsize=20, pad=20)
plt.ylabel("Count", weight="bold", fontsize=15)
plt.xlabel("Fuel Type", weight="bold", fontsize=12)
plt.show()
```

### Fuel Type Count



### Points taken

- Petrol and Diesel dominate the used car market in the website.
- The most sold fuel type Vechicle is Petrol.
- Followed by diesel and CNG and least sold is Electric

### Fuel types available and mileage given

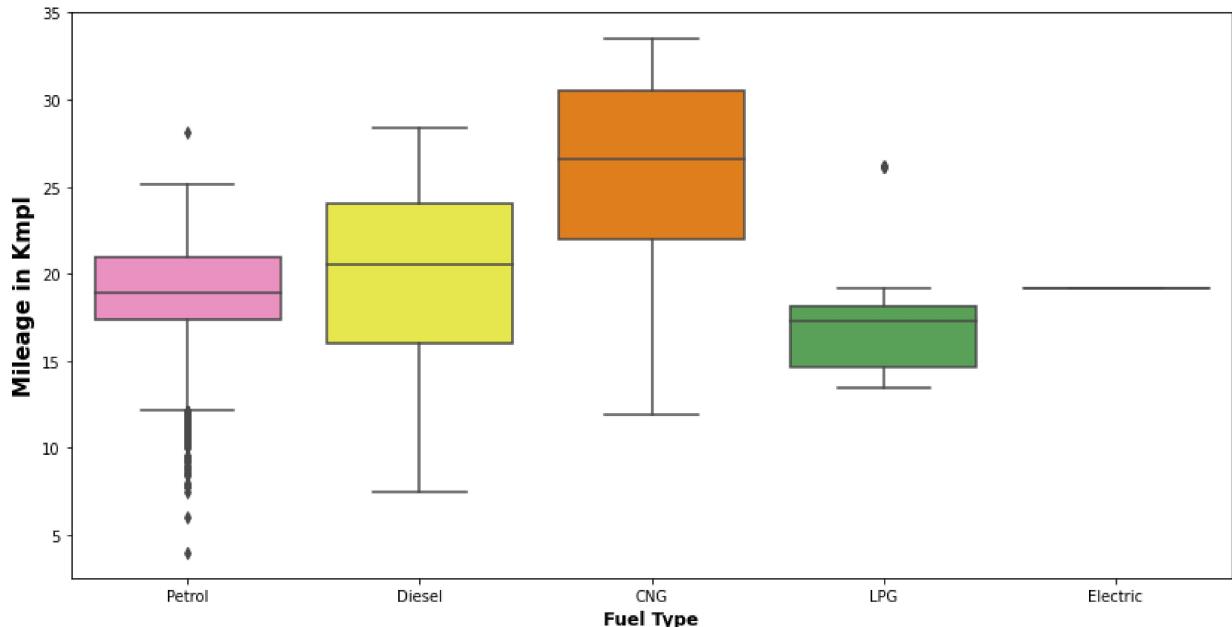
```
In [57]: fuel_mileage = df.groupby('fuel_type')['mileage'].mean().sort_values(ascending=False)
fuel_mileage.to_frame()
```

Out[57]: mileage

<b>fuel_type</b>	
<b>CNG</b>	25.814651
<b>Diesel</b>	20.060030
<b>Electric</b>	19.160000
<b>Petrol</b>	19.123045
<b>LPG</b>	17.836364

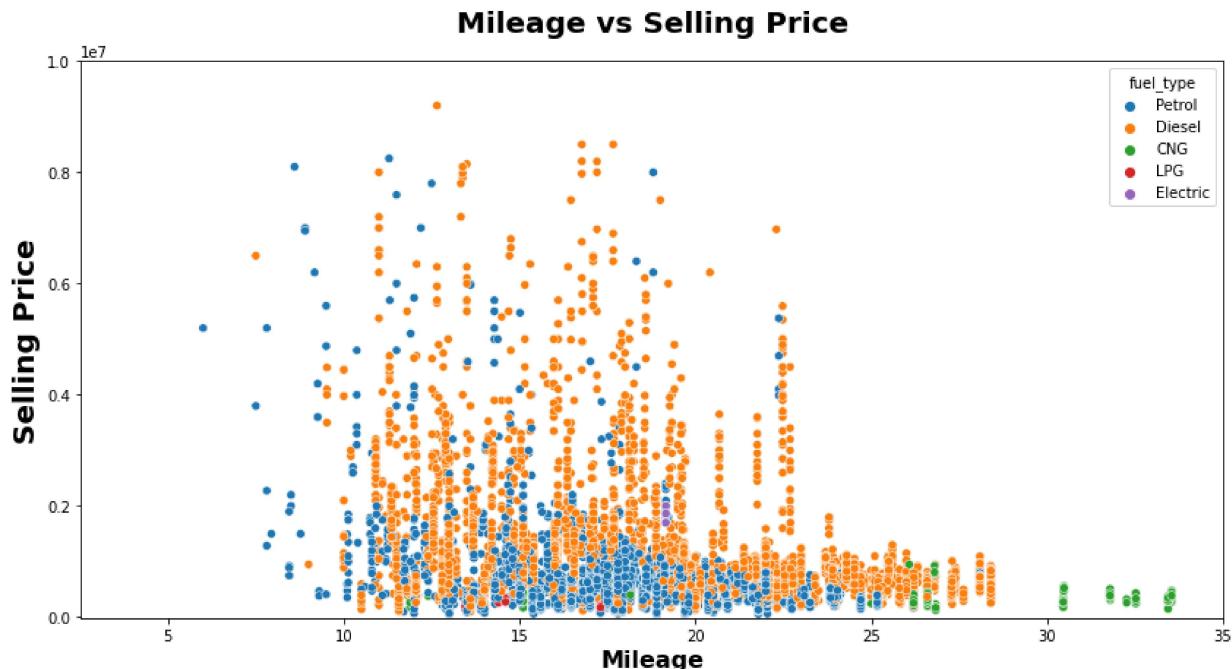
```
In [58]: plt.subplots(figsize=(14,7))
sns.boxplot(x='fuel_type', y='mileage', data=df, palette="Set1_r")
plt.title("Fuel type vs Mileage", weight="bold", fontsize=20, pad=20)
plt.ylabel("Mileage in Kmpl", weight="bold", fontsize=15)
plt.xlabel("Fuel Type", weight="bold", fontsize=12)
plt.show()
```

### Fuel type vs Mileage



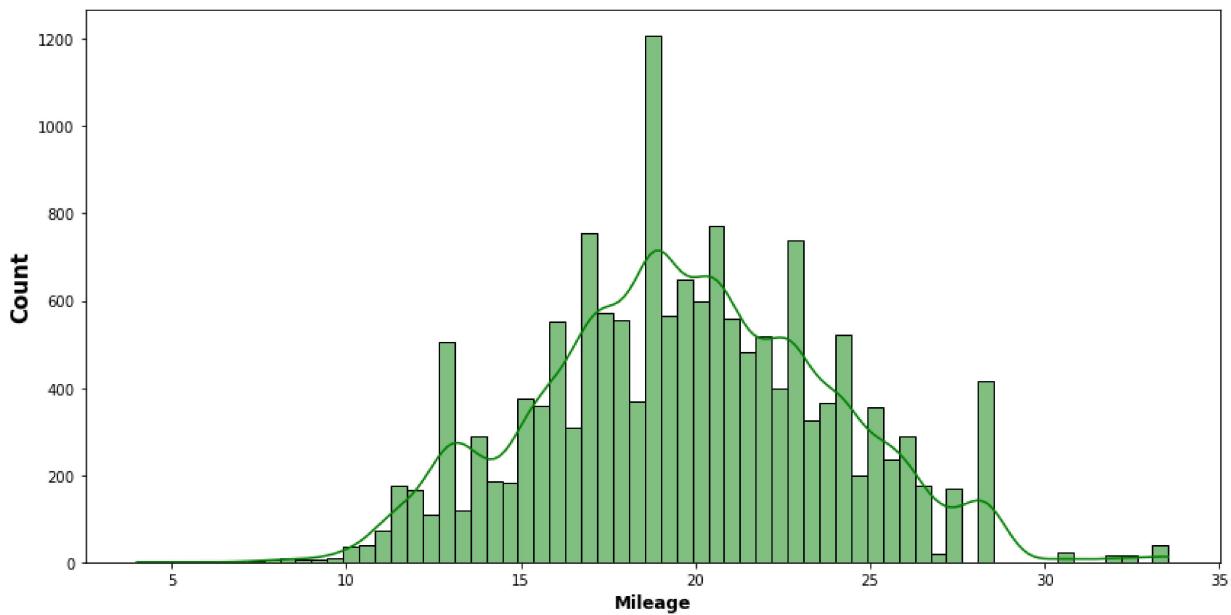
### Mileage vs Selling Price

```
In [59]: plt.subplots(figsize=(14,7))
sns.scatterplot(x="mileage", y='selling_price', data=df, ec = "white", color='b', hue=fuel_type)
plt.title("Mileage vs Selling Price", weight="bold", fontsize=20, pad=20)
plt.ylabel("Selling Price", weight="bold", fontsize=20)
plt.ylim(-10000,10000000)
plt.xlabel("Mileage", weight="bold", fontsize=16)
plt.show()
```



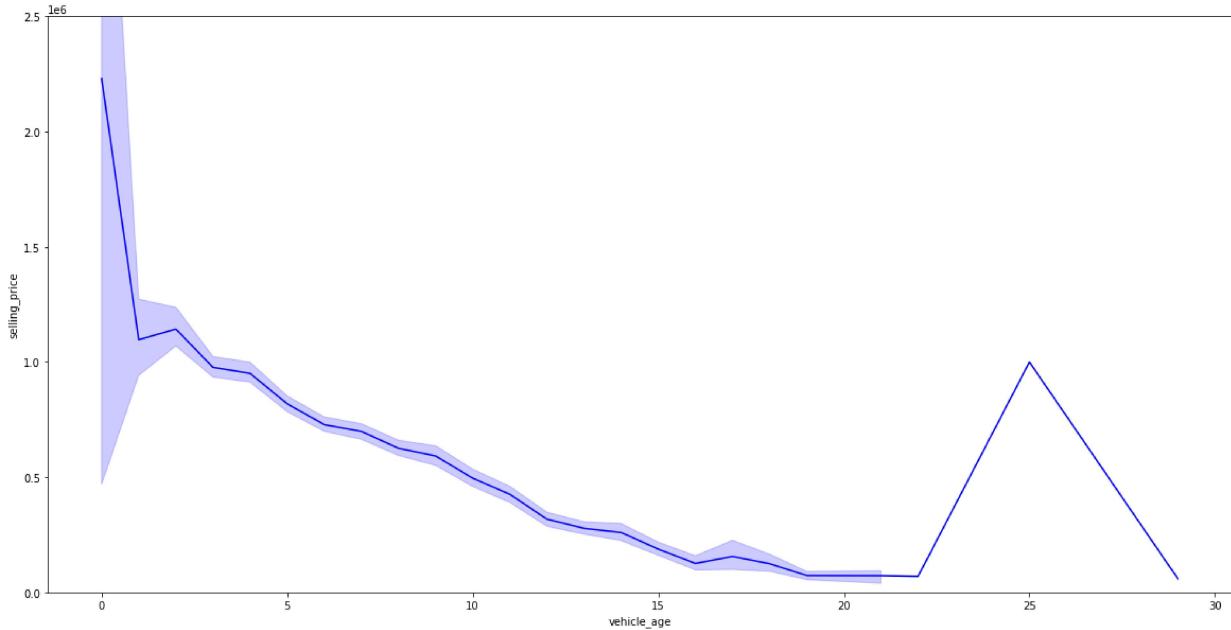
```
In [60]: plt.subplots(figsize=(14,7))
sns.histplot(x=df.mileage, ec = "black", color='g', kde=True)
plt.title("Mileage Distribution", weight="bold", fontsize=20, pad=20)
plt.ylabel("Count", weight="bold", fontsize=15)
```

```
plt.xlabel("Mileage", weight="bold", fontsize=12)
plt.show()
```

**Mileage Distribution**

## Vehicle age vs Selling Price

```
In [61]: plt.subplots(figsize=(20,10))
sns.lineplot(x='vehicle_age',y='selling_price',data=df,color='b')
plt.ylim(0,2500000)
plt.show()
```



## Points taken

- As the Vehicle age increases the price also get reduced.
- Vehicle age has Negative impact on selling price

## Vehicle age vs Mileage

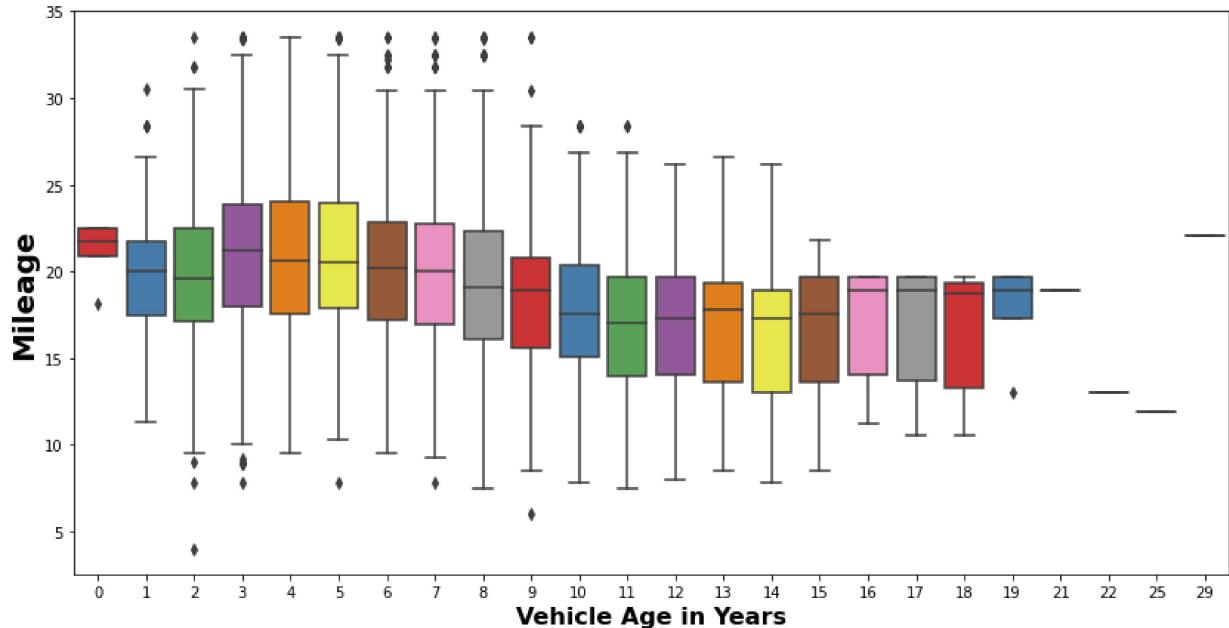
```
In [62]: vehicle_age = df.groupby('vehicle_age')['mileage'].median().sort_values(ascending=False).to_frame().head(5)
```

Out[62]:

**mileage**

<b>vehicle_age</b>	<b>mileage</b>
<b>29</b>	22.05
<b>0</b>	21.70
<b>3</b>	21.21
<b>4</b>	20.63
<b>5</b>	20.51

```
In [63]: plt.subplots(figsize=(14,7))
sns.boxplot(x=df.vehicle_age, y= df.mileage, palette="Set1")
plt.title("Vehicle Age vs Mileage", weight="bold", fontsize=20, pad=20)
plt.ylabel("Mileage", weight="bold", fontsize=20)
plt.xlabel("Vehicle Age in Years", weight="bold", fontsize=16)
plt.show()
```

**Vehicle Age vs Mileage**

### Point taken

- As the Age of vehicle increases the median of mileage drops.
- Newer Vehicles have more mileage median older vehicle.

```
In [65]: oldest = df.groupby('car_name')['vehicle_age'].max().sort_values(ascending=False).head(5)
```

Out[65]:

**vehicle\_age**

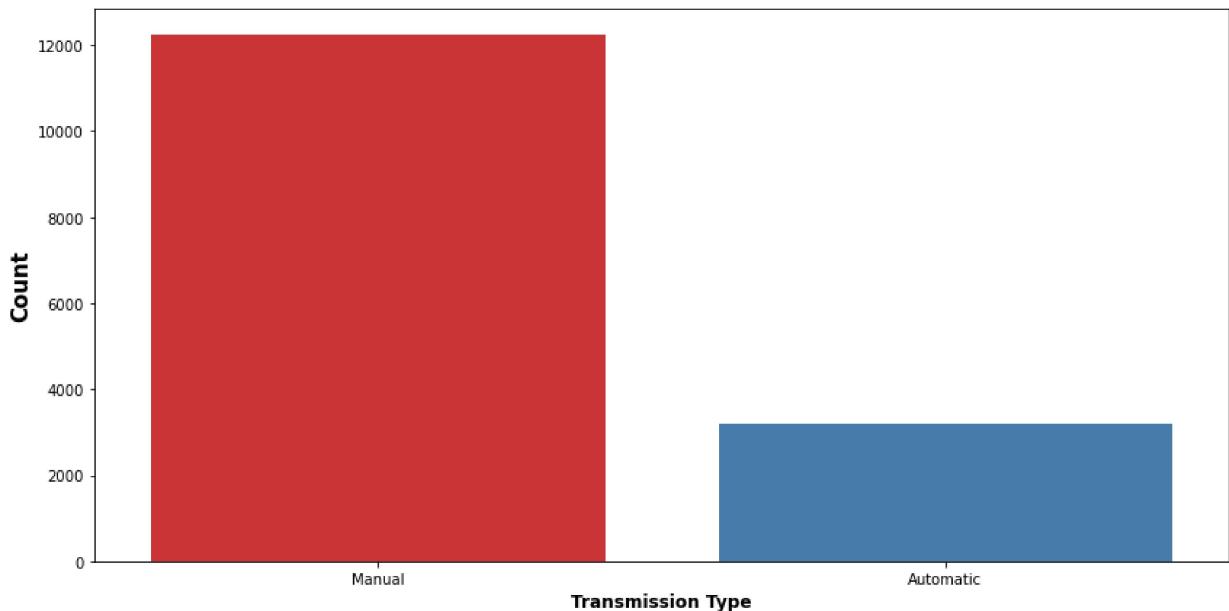
<b>car_name</b>	
<b>Maruti Alto</b>	29
<b>BMW 3</b>	25
<b>Honda City</b>	22
<b>Maruti Wagon R</b>	21
<b>Mahindra Bolero</b>	18
<b>Mahindra Scorpio</b>	18
<b>Skoda Octavia</b>	18
<b>Honda CR-V</b>	17
<b>Mercedes-Benz E-Class</b>	17
<b>Honda Civic</b>	15

**Point taken**

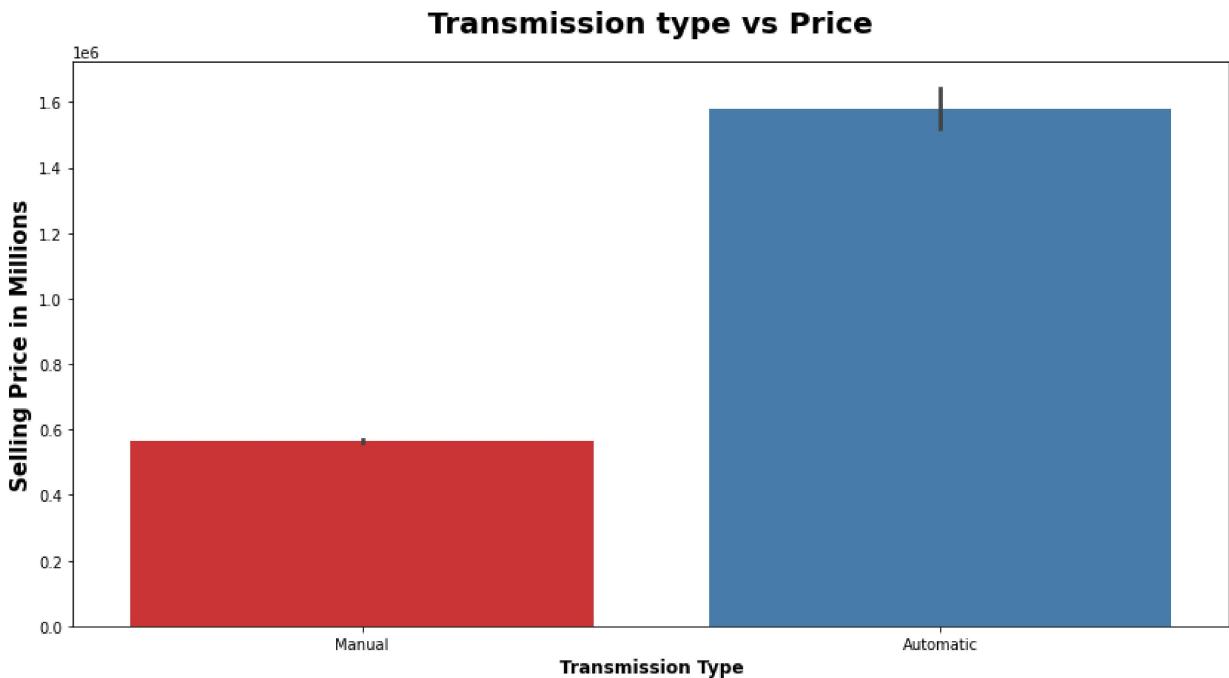
- Maruti Alto is the Oldest car available 29 years old in the used car website followed by BMW 3 for 25 years old.

**Transmission Type**

```
In [66]: plt.subplots(figsize=(14,7))
sns.countplot(x='transmission_type', data=df, palette="Set1")
plt.title("Transmission type Count", weight="bold", fontsize=20, pad=20)
plt.ylabel("Count", weight="bold", fontsize=15)
plt.xlabel("Transmission Type", weight="bold", fontsize=12)
plt.show()
```

**Transmission type Count**

```
In [67]: plt.subplots(figsize=(14,7))
sns.barplot(x='transmission_type', y='selling_price', data=df, palette="Set1")
plt.title("Transmission type vs Price", weight="bold", fontsize=20, pad=20)
plt.ylabel("Selling Price in Millions", weight="bold", fontsize=15)
plt.xlabel("Transmission Type", weight="bold", fontsize=12)
plt.show()
```

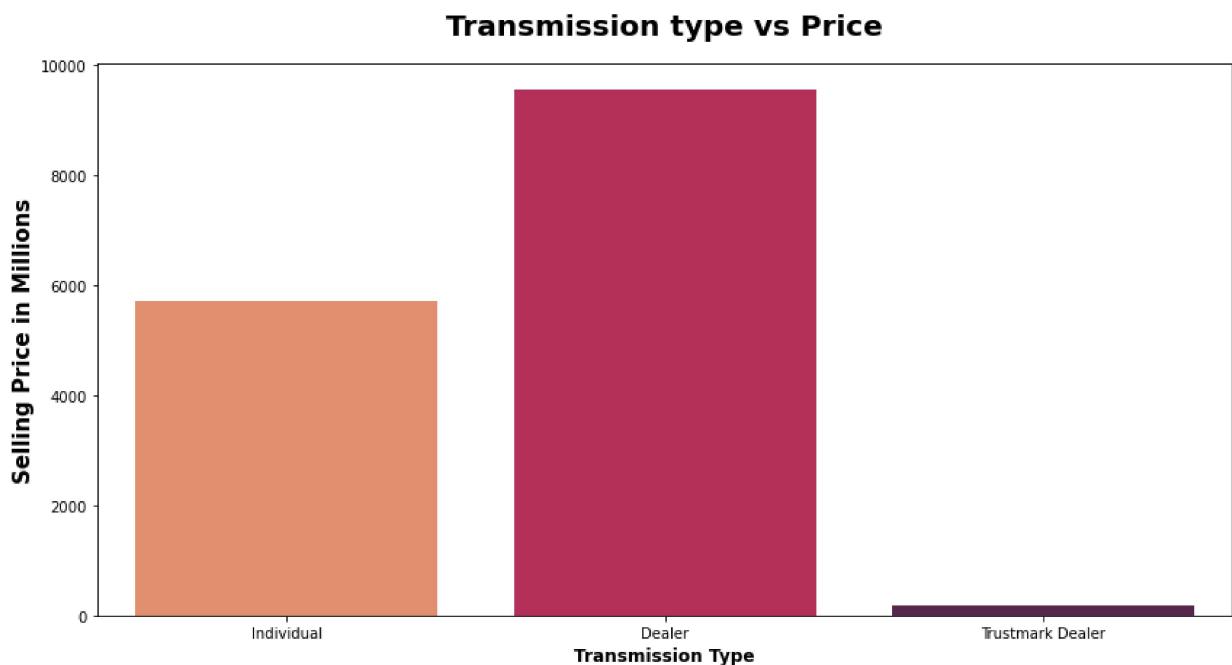


## Point taken

- Manual Transmission was found in most of the cars which was sold.
- Automatic cars have more selling price than manual cars.

## Seller Type

```
In [68]: plt.subplots(figsize=(14,7))
sns.countplot(x='seller_type', data=df, palette="rocket_r")
plt.title("Transmission type vs Price", weight="bold", fontsize=20, pad=20)
plt.ylabel("Selling Price in Millions", weight="bold", fontsize=15)
plt.xlabel("Transmission Type", weight="bold", fontsize=12)
plt.show()
```



```
In [69]: dealer = df.groupby('seller_type')['selling_price'].median().sort_values(ascending=False)
dealer.to_frame()
```

```
Out[69]:      selling_price
seller_type
Dealer      591000.0
Trustmark Dealer  540000.0
Individual    507000.0
```

## Points Taken

- Dealers have put more ads on used car website.
- Dealers have put 9539 ads with median selling price of 5.91 Lakhs.
- Followed by Individual with 5699 ads with median selling price of 5.4 Lakhs.

- Dealers have more median selling price than Individual.

## Final Findings

- The datatypes and Column names were right and there was 15411 rows and 13 columns
- The selling\_price column is the target to predict. i.e Regression Problem.
- There are outliers in the km\_driven, enginer, selling\_price, and max power.
- Dealers are the highest sellers of the used cars.
- Skewness is found in few of the columns will check it after handling outliers.
- Vehicle age has negative impact on the price.
- Manual cars are mostly sold and automatic has higher selling average than manual cars.
- Petrol is the most preffered choice of fuel in used car website, followed by diesel and LPG.
- We just need less data cleaning for this dataset.

In [ ]: