```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```python
var=pd.read_excel("C:\\Users\\HP\\Documents\\Documents\\CarDekho_Dataset.xlsx")
```

```python
var
```

	Car_Name	Year	Selling_Price	KM_Driven	Fuel	Seller_Type	Transmission	Owner	Mileage(km/ltr/kg)	Engine	Max_power	Seats
0	Maruti Swift Dzire VDI	2014	450000	145500	Diesel	Individual	Manual	First Owner	23	1248	74.0	5
1	Skoda Rapid 1.5 TDI Ambition	2014	370000	120000	Diesel	Individual	Manual	Second Owner	21	1498	104.0	5
2	Honda City 2017-2020 EXi	2006	158000	140000	Petrol	Individual	Manual	Third Owner	18	1497	78.0	5
3	Hyundai i20 Sportz Diesel	2010	225000	127000	Diesel	Individual	Manual	First Owner	23	1396	90.0	5
4	Maruti Swift VXI BSIII	2007	130000	120000	Petrol	Individual	Manual	First Owner	16	1298	88.0	5
...	...	...	...	...	...	...	...	...	...	...	...	...
7902	Hyundai i20 Magna	2013	320000	110000	Petrol	Individual	Manual	First Owner	19	1197	83.0	5
7903	Hyundai Verna CRDi SX	2007	135000	119000	Diesel	Individual	Manual	Fourth & Above Owner	17	1493	110.0	5
7904	Maruti Swift Dzire ZDi	2009	382000	120000	Diesel	Individual	Manual	First Owner	19	1248	74.0	5
7905	Tata Indigo CR4	2013	290000	25000	Diesel	Individual	Manual	First Owner	24	1396	70.0	5
7906	Tata Indigo CR4	2013	290000	25000	Diesel	Individual	Manual	First Owner	24	1396	70.0	5

7907 rows × 12 columns

```python
var.shape
```

```
(7907, 12)
```

```python
var.isnull().sum()
```

```
Car_Name              0
Year                  0
Selling_Price         0
KM_Driven             0
Fuel                  0
Seller_Type           0
Transmission          0
Owner                 0
Mileage(km/ltr/kg)    0
Engine                0
Max_power             1
Seats                 0
dtype: int64
```

```python
var.info
```

```
<bound method DataFrame.info of                          Car_Name  Year  Selling_Price  KM_Driven    Fuel  \
0           Maruti Swift Dzire VDI  2014         450000     145500  Diesel
1     Skoda Rapid 1.5 TDI Ambition  2014         370000     120000  Diesel
2         Honda City 2017-2020 EXi  2006         158000     140000  Petrol
3        Hyundai i20 Sportz Diesel  2010         225000     127000  Diesel
4           Maruti Swift VXI BSIII  2007         130000     120000  Petrol
...                            ...   ...            ...        ...     ...
7902             Hyundai i20 Magna  2013         320000     110000  Petrol
7903         Hyundai Verna CRDi SX  2007         135000     119000  Diesel
7904        Maruti Swift Dzire ZDi  2009         382000     120000  Diesel
7905               Tata Indigo CR4  2013         290000      25000  Diesel
7906               Tata Indigo CR4  2013         290000      25000  Diesel

     Seller_Type Transmission                 Owner  Mileage(km/ltr/kg)  \
0     Individual       Manual           First Owner                  23
1     Individual       Manual          Second Owner                  21
2     Individual       Manual           Third Owner                  18
3     Individual       Manual           First Owner                  23
4     Individual       Manual           First Owner                  16
...          ...          ...                   ...                 ...
7902  Individual       Manual           First Owner                  19
7903  Individual       Manual  Fourth & Above Owner                  17
7904  Individual       Manual           First Owner                  19
7905  Individual       Manual           First Owner                  24
7906  Individual       Manual           First Owner                  24

      Engine  Max_power  Seats
0       1248       74.0      5
1       1498      104.0      5
2       1497       78.0      5
3       1396       90.0      5
4       1298       88.0      5
...      ...        ...    ...
7902    1197       83.0      5
7903    1493      110.0      5
7904    1248       74.0      5
7905    1396       70.0      5
7906    1396       70.0      5

[7907 rows x 12 columns]>
```

```python
var.describe()
```

	Year	Selling_Price	KM_Driven	Mileage(km/ltr/kg)	Engine	Max_power	Seats
count	7907.000000	7.907000e+03	7.907000e+03	7907.000000	7907.000000	7906.000000	7907.000000
mean	2013.982168	6.497417e+05	6.919256e+04	19.446566	1458.625016	91.635214	5.416719
std	3.866650	8.135565e+05	5.678976e+04	4.012947	503.916303	35.781047	0.959588
min	1994.000000	2.999900e+04	1.000000e+00	0.000000	624.000000	33.000000	2.000000
25%	2012.000000	2.700000e+05	3.500000e+04	17.000000	1197.000000	68.000000	5.000000
50%	2015.000000	4.500000e+05	6.000000e+04	19.000000	1248.000000	82.000000	5.000000
75%	2017.000000	6.900000e+05	9.575000e+04	22.000000	1582.000000	102.000000	5.000000
max	2020.000000	1.000000e+07	2.360457e+06	42.000000	3604.000000	400.000000	14.000000

```python
var.head()
```

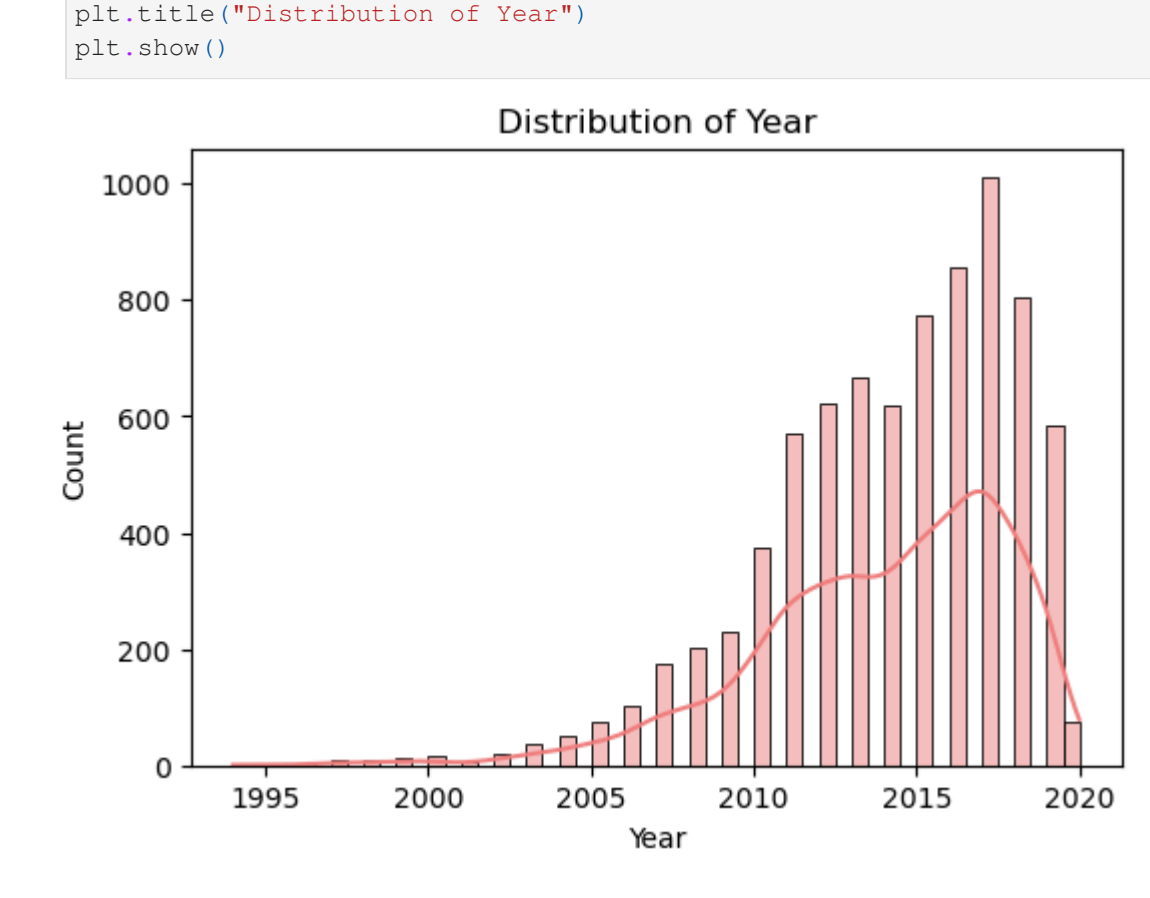
	Car_Name	Year	Selling_Price	KM_Driven	Fuel	Seller_Type	Transmission	Owner	Mileage(km/ltr/kg)	Engine	Max_power	Seats
0	Maruti Swift Dzire VDI	2014	450000	145500	Diesel	Individual	Manual	First Owner	23	1248	74.0	5
1	Skoda Rapid 1.5 TDI Ambition	2014	370000	120000	Diesel	Individual	Manual	Second Owner	21	1498	104.0	5
2	Honda City 2017-2020 EXi	2006	158000	140000	Petrol	Individual	Manual	Third Owner	18	1497	78.0	5
3	Hyundai i20 Sportz Diesel	2010	225000	127000	Diesel	Individual	Manual	First Owner	23	1396	90.0	5
4	Maruti Swift VXI BSIII	2007	130000	120000	Petrol	Individual	Manual	First Owner	16	1298	88.0	5

```python
var.columns
```

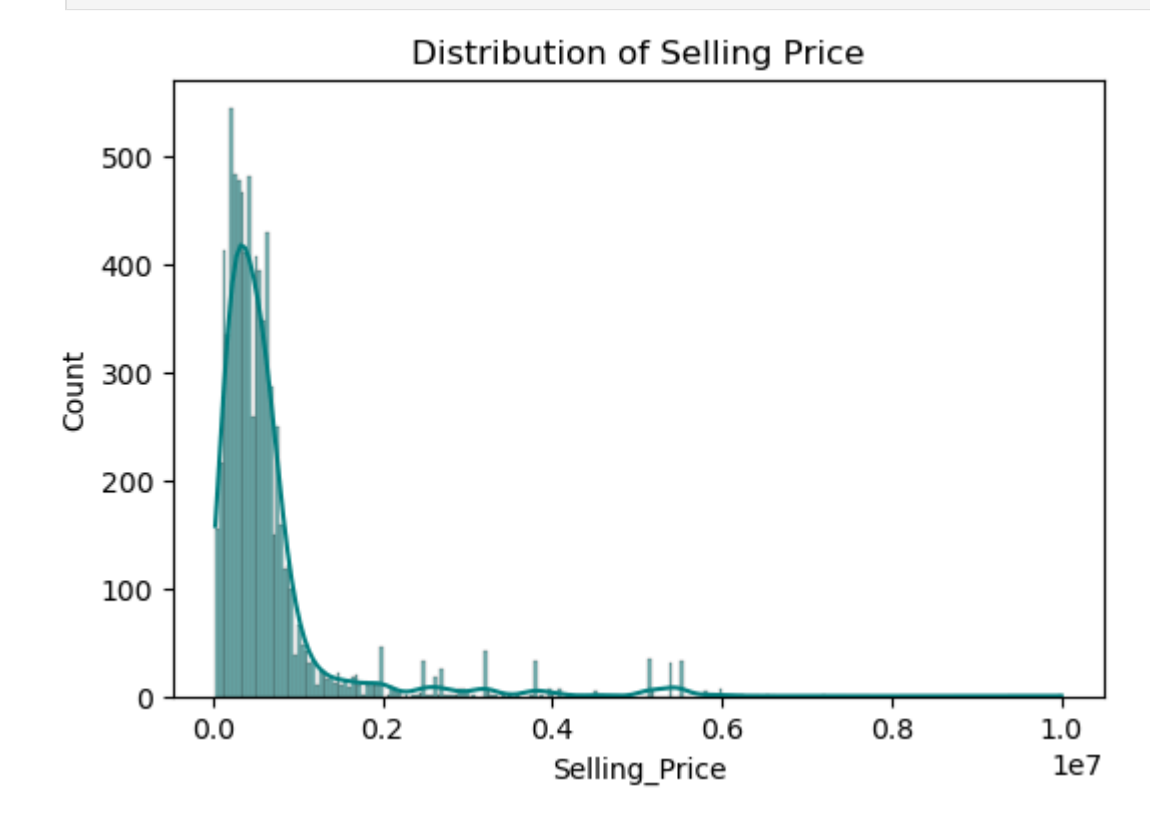
```
Index(['Car_Name', 'Year', 'Selling_Price', 'KM_Driven', 'Fuel', 'Seller_Type',
       'Transmission', 'Owner', 'Mileage(km/ltr/kg)', 'Engine', 'Max_power',
       'Seats'],
      dtype='object')
```

Exploratory Data Analysis:

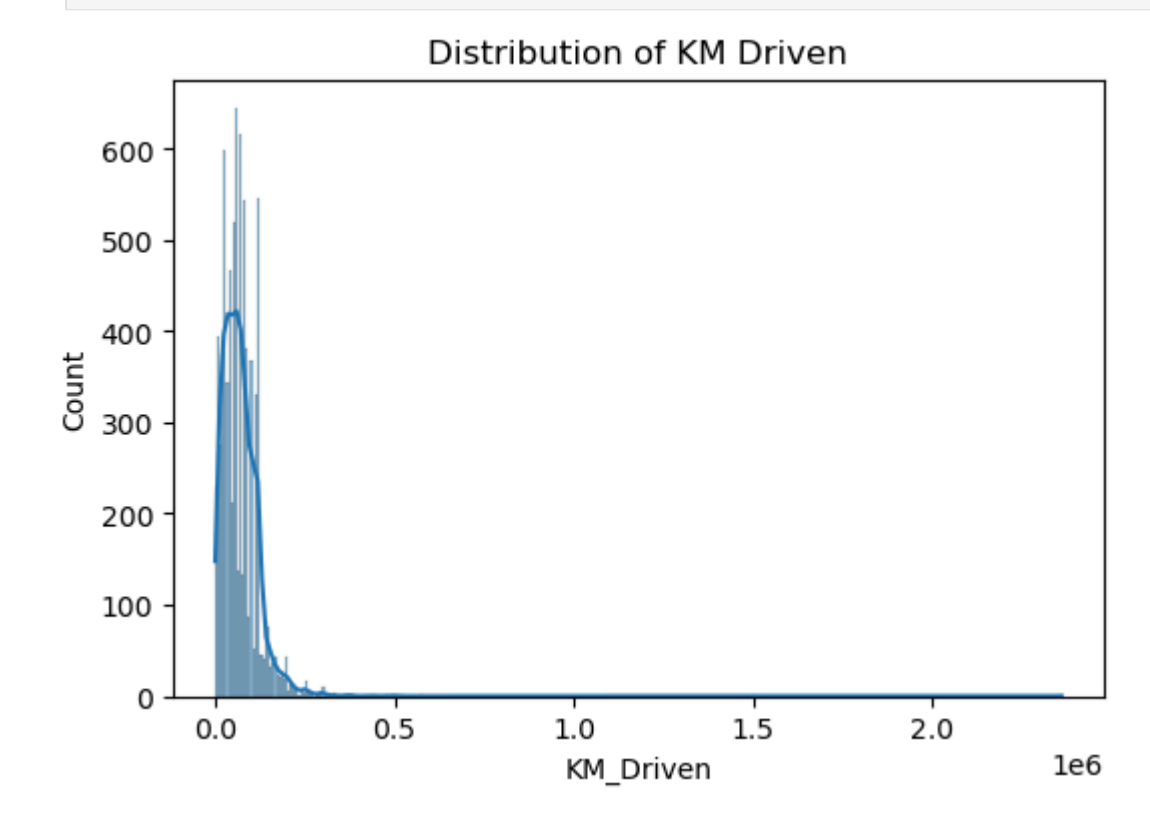
```python
plt.figure(figsize=(6,4))
sns.histplot(var["Year"], kde=True, color="lightcoral")
plt.title("Distribution of Year")
plt.show()
```




```python
plt.figure(figsize=(6,4))
sns.histplot(var["Selling_Price"], kde=True, color="teal")
plt.title("Distribution of Selling Price")
plt.show()
```




```python
plt.figure(figsize=(6,4))
sns.histplot(var["KM_Driven"], kde="blue")
plt.title("Distribution of KM Driven")
plt.show()
```

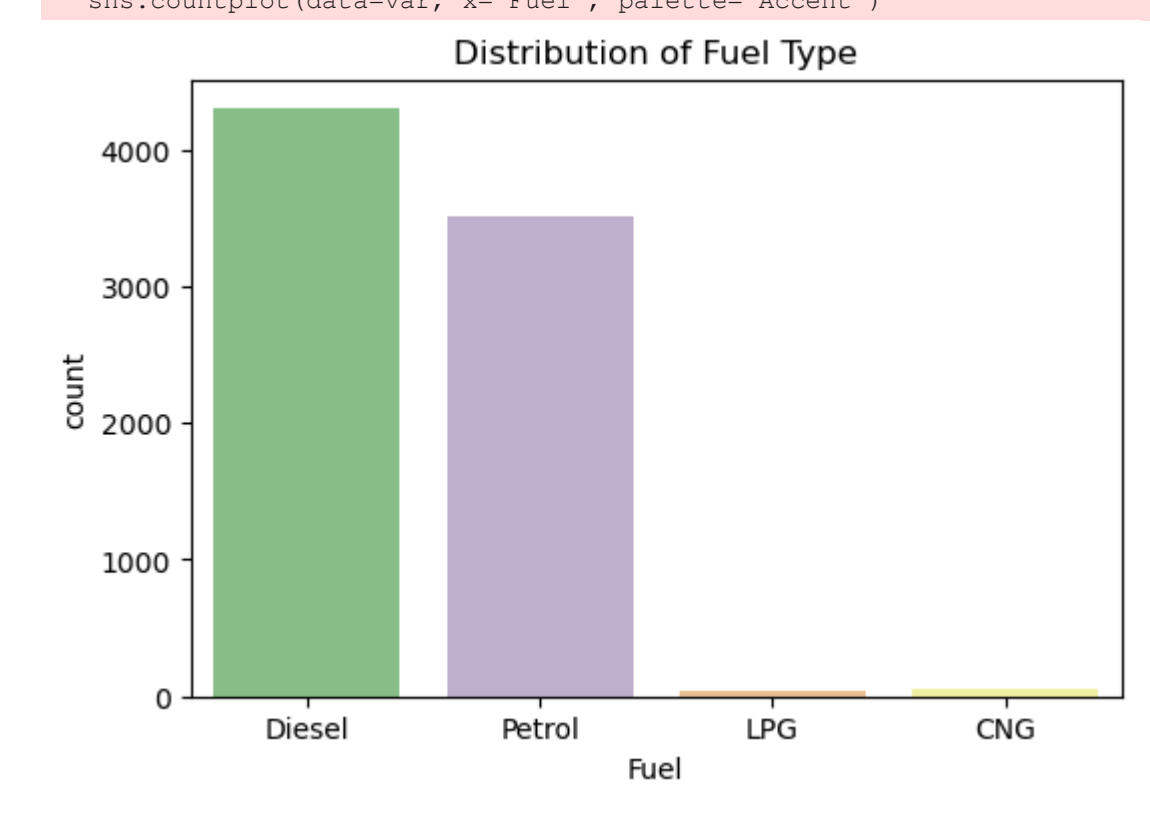



```python
print("Unique Fuel Types:\n", var['Fuel'].value_counts())
print("\nUnique Seller Types:\n", var['Seller_Type'].value_counts())
```

```
Unique Fuel Types:
 Fuel
Diesel    4299
Petrol    3520
CNG         53
LPG         35
Name: count, dtype: int64

Unique Seller Types:
 Seller_Type
Individual          6564
Dealer              1107
Trustmark Dealer     236
Name: count, dtype: int64
```

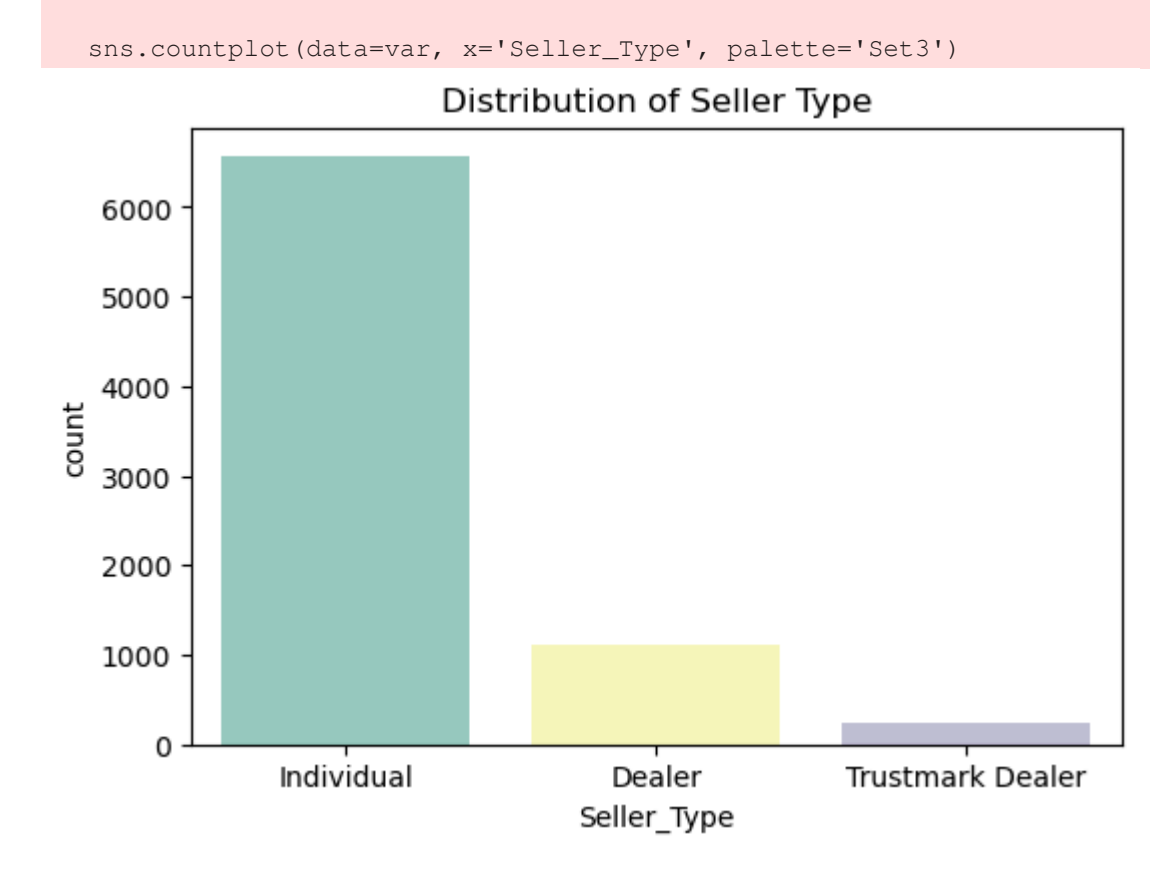
```python
plt.figure(figsize=(6,4))
sns.countplot(data=var, x='Fuel', palette='Accent')
plt.title('Distribution of Fuel Type')
plt.show()
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_11768\608373977.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.countplot(data=var, x='Fuel', palette='Accent')
```



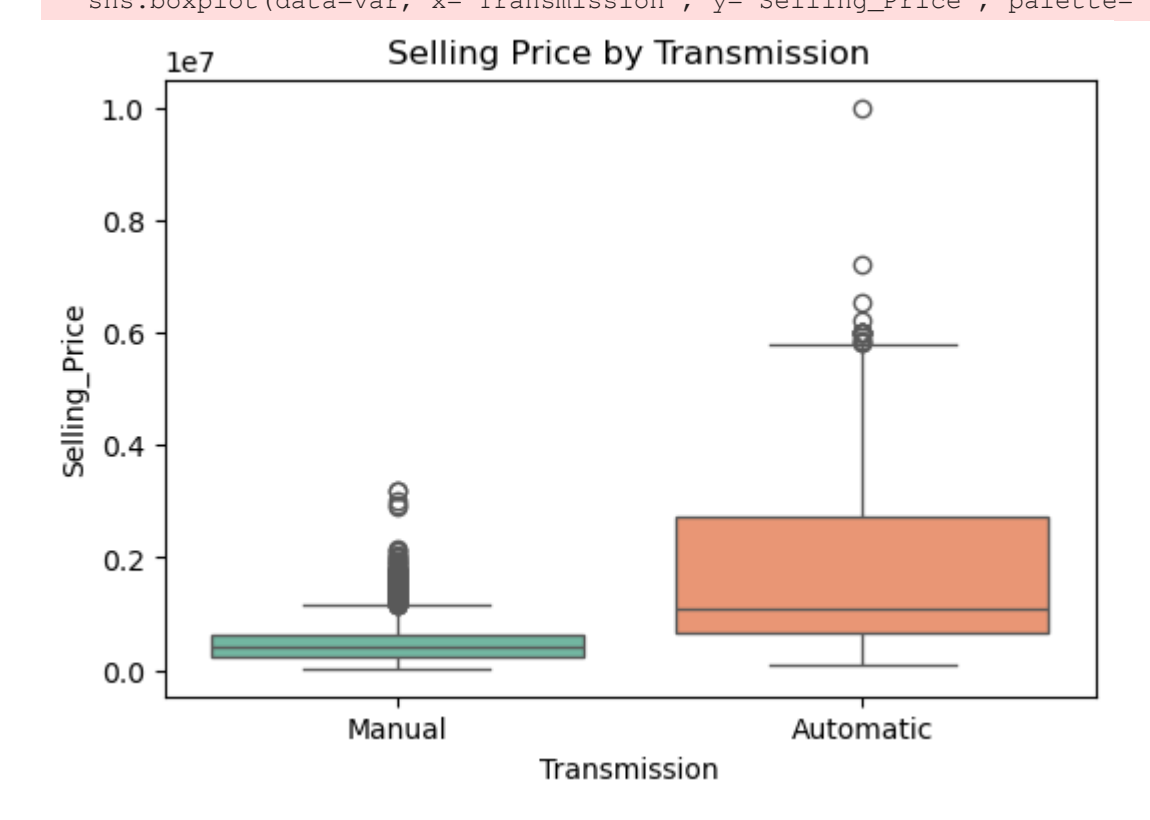

```python
plt.figure(figsize=(6,4))
sns.countplot(data=var, x='Seller_Type', palette='Set3')
plt.title('Distribution of Seller Type')
plt.show()
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_11768\725721029.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.countplot(data=var, x='Seller_Type', palette='Set3')
```

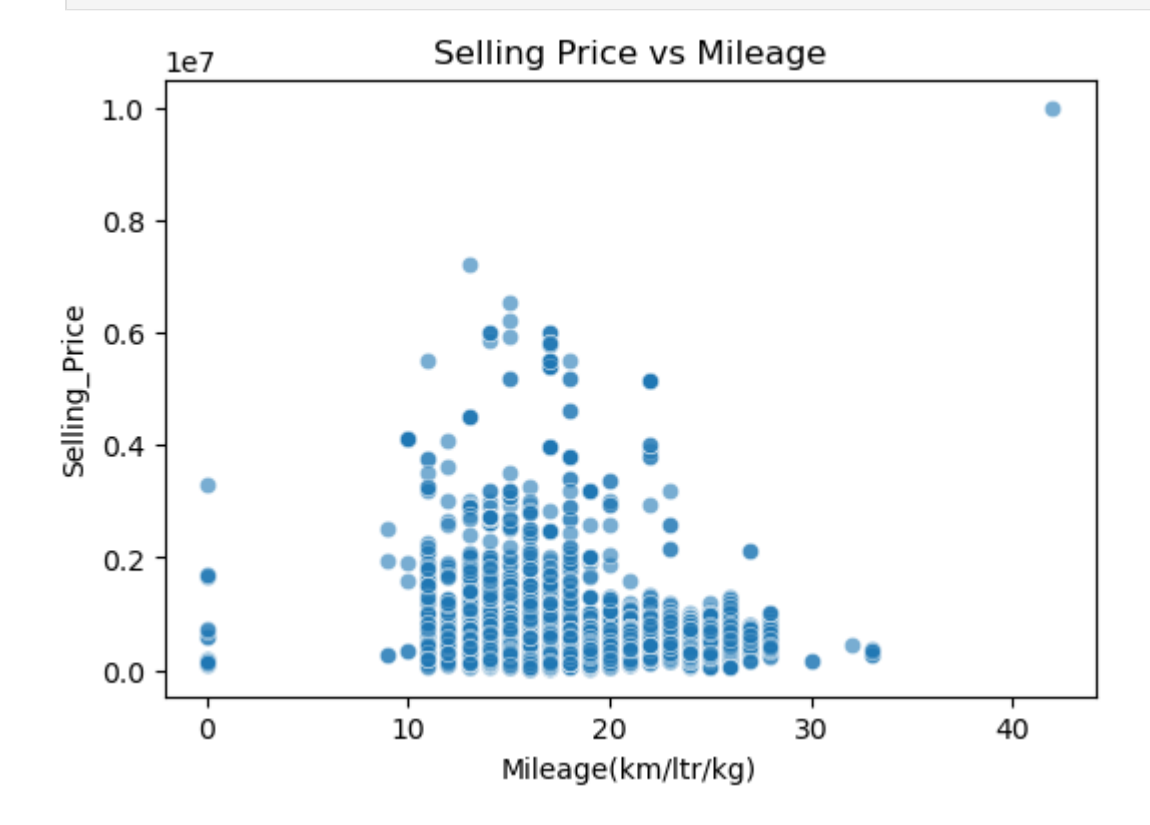



```python
plt.figure(figsize=(6,4))
sns.boxplot(data=var, x='Transmission', y='Selling_Price', palette='Set2')
plt.title('Selling Price by Transmission')
plt.show()
```

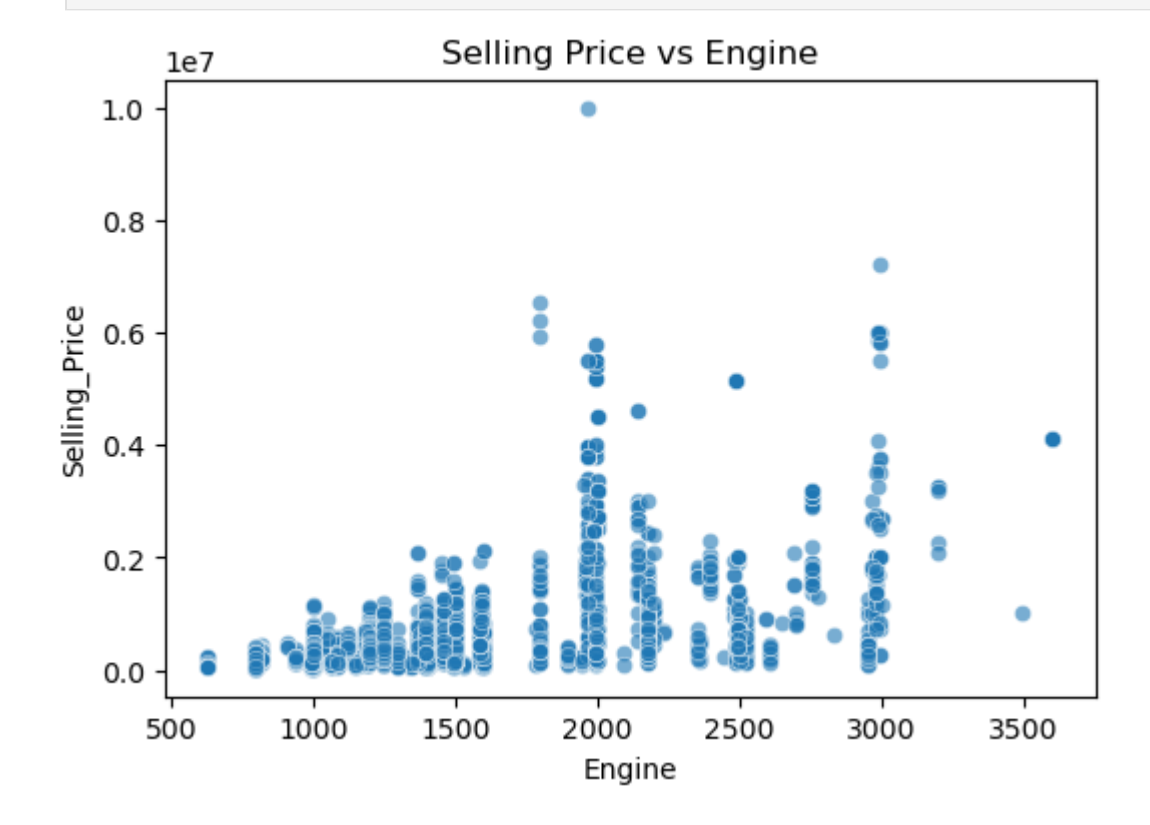
```
C:\Users\HP\AppData\Local\Temp\ipykernel_11768\2792767789.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.boxplot(data=var, x='Transmission', y='Selling_Price', palette='Set2')
```




```python
plt.figure(figsize=(6,4))
sns.scatterplot(x='Mileage(km/ltr/kg)', y='Selling_Price', data=var, alpha=0.6)
plt.title('Selling Price vs Mileage')
plt.show()
```




```python
plt.figure(figsize=(6,4))
sns.scatterplot(x='Engine', y='Selling_Price', data=var, alpha=0.6)
plt.title('Selling Price vs Engine')
plt.show()
```



Regression Analysis:

```python
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
```

```python
var
```

	Car_Name	Year	Selling_Price	KM_Driven	Fuel	Seller_Type	Transmission	Owner	Mileage(km/ltr/kg)	Engine	Max_power	Seats
0	Maruti Swift Dzire VDI	2014	450000	145500	Diesel	Individual	Manual	First Owner	23	1248	74.0	5
1	Skoda Rapid 1.5 TDI Ambition	2014	370000	120000	Diesel	Individual	Manual	Second Owner	21	1498	104.0	5
2	Honda City 2017-2020 EXi	2006	158000	140000	Petrol	Individual	Manual	Third Owner	18	1497	78.0	5
3	Hyundai i20 Sportz Diesel	2010	225000	127000	Diesel	Individual	Manual	First Owner	23	1396	90.0	5
4	Maruti Swift VXI BSIII	2007	130000	120000	Petrol	Individual	Manual	First Owner	16	1298	88.0	5
...	...	...	...	...	...	...	...	...	...	...	...	...
7902	Hyundai i20 Magna	2013	320000	110000	Petrol	Individual	Manual	First Owner	19	1197	83.0	5
7903	Hyundai Verna CRDi SX	2007	135000	119000	Diesel	Individual	Manual	Fourth & Above Owner	17	1493	110.0	5
7904	Maruti Swift Dzire ZDi	2009	382000	120000	Diesel	Individual	Manual	First Owner	19	1248	74.0	5
7905	Tata Indigo CR4	2013	290000	25000	Diesel	Individual	Manual	First Owner	24	1396	70.0	5
7906	Tata Indigo CR4	2013	290000	25000	Diesel	Individual	Manual	First Owner	24	1396	70.0	5

7907 rows × 12 columns

```python
var.isnull().sum()
```

```
Car_Name              0
Year                  0
Selling_Price         0
KM_Driven             0
Fuel                  0
Seller_Type           0
Transmission          0
Owner                 0
Mileage(km/ltr/kg)    0
Engine                0
Max_power             0
Seats                 0
dtype: int64
```

```python
var['Car_Name'] = var['Car_Name'].astype('category')
var['Car_Name'] = var['Car_Name'].cat.codes

var['Fuel'] = var['Fuel'].astype('category')
var['Fuel'] = var['Fuel'].cat.codes

var['Seller_Type'] = var['Seller_Type'].astype('category')
var['Seller_Type'] = var['Seller_Type'].cat.codes

var['Transmission'] = var['Transmission'].astype('category')
var['Transmission'] = var['Transmission'].cat.codes

var['Owner'] = var['Owner'].astype('category')
var['Owner'] = var['Owner'].cat.codes
```

```python
x=var.drop(columns="Selling_Price")
x
```

	Car_Name	Year	KM_Driven	Fuel	Seller_Type	Transmission	Owner	Mileage(km/ltr/kg)	Engine	Max_power	Seats
0	1249	2014	145500	1	1	1	0	23	1248	74.0	5
1	1546	2014	120000	1	1	1	2	21	1498	104.0	5
2	375	2006	140000	3	1	1	4	18	1497	78.0	5
3	743	2010	127000	1	1	1	0	23	1396	90.0	5
4	1290	2007	120000	3	1	1	0	16	1298	88.0	5
...	...	...	...	...	...	...	...	...	...	...	...
7902	733	2013	110000	3	1	1	0	19	1197	83.0	5
7903	618	2007	119000	1	1	1	1	17	1493	110.0	5
7904	1261	2009	120000	1	1	1	0	19	1248	74.0	5
7905	1633	2013	25000	1	1	1	0	24	1396	70.0	5
7906	1633	2013	25000	1	1	1	0	24	1396	70.0	5

7906 rows × 11 columns

```python
y=var["Selling_Price"]
y
```

```
0       450000
1       370000
2       158000
3       225000
4       130000
         ...
7902    320000
7903    135000
7904    382000
7905    290000
7906    290000
Name: Selling_Price, Length: 7906, dtype: int64
```

```python
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=0.25, random_state=0)
```

```python
model=LinearRegression()
```

```python
model.fit(x_train, y_train)
```

```
▾ LinearRegression
LinearRegression()
```

```python
c=model.intercept_
c
```

```
-65846887.13410325
```

```python
m=model.coef_
m
```

```
array([-2.86018367e+01,  3.27392846e+04, -1.00056037e+00, -2.12823215e+04,
       -2.04102904e+05, -4.79568948e+05, -1.19954691e+04,  1.16265723e+04,
        9.53012709e+01,  1.27036516e+04, -4.12373599e+04])
```

```python
predicted_y_test=model.predict(x_test)
predicted_y_test
```

```
array([ 582603.47243907, 1024695.99329194,  135493.22963422, ...,
        407143.53431147,  370562.7418961 , 3691977.20840198])
```

```python
mse=mean_squared_error(y_test, predicted_y_test)
mse
```

```
214920480253.29788
```

```python
r2=r2_score(y_test, predicted_y_test)
r2
```

```
0.6781921744374465
```

```python
print(f" R^2 Value is: {r2*100:.2f}%")
```

```
R^2 Value is: 67.82%
```

Insight: In this modelling scenario, we are getting the R-squared value of 67.82%, it means that 67.82% of times, the independent variables would be able to explain the changes in the depenedent variable.