Car Price Pridiction

Load the Dataset

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

data = pd.read_csv('/content/car data.csv')
print(data.head())
```

```
  Car_Name  Year  Selling_Price  Present_Price  Driven_kms Fuel_Type  \
0     ritz  2014           3.35           5.59       27000    Petrol
1      sx4  2013           4.75           9.54       43000    Diesel
2     ciaz  2017           7.25           9.85        6900    Petrol
3  wagon r  2011           2.85           4.15        5200    Petrol
4    swift  2014           4.60           6.87       42450    Diesel

  Selling_type Transmission  Owner
0       Dealer       Manual      0
1       Dealer       Manual      0
2       Dealer       Manual      0
3       Dealer       Manual      0
4       Dealer       Manual      0
```

Display basic information

```
print(data.info())
print(data.describe())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 301 entries, 0 to 300
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Car_Name              301 non-null    object
 1   Year                  301 non-null    int64
 2   Selling_Price         301 non-null    float64
 3   Present_Price         301 non-null    float64
 4   Driven_kms            301 non-null    int64
 5   Fuel_Type             301 non-null    object
 6   Selling_type          301 non-null    object
 7   Transmission          301 non-null    object
 8   Owner                 301 non-null    int64
 9   Car_Age               301 non-null    int64
 10  Price_per_km          301 non-null    float64
 11  Price_Age_Interaction  301 non-null    float64
dtypes: float64(4), int64(4), object(4)
memory usage: 28.3+ KB
None
              Year  Selling_Price  Present_Price     Driven_kms       Owner  \
```

```
count   301.000000     301.000000     301.000000     301.000000  301.000000
mean   2013.627907       4.322591       7.628472   36947.205980    0.043189
std       2.891554       3.867603       8.642584   38886.883882    0.247915
min    2003.000000       0.250000       0.320000     500.000000    0.000000
25%    2012.000000       0.900000       1.200000   15000.000000    0.000000
50%    2014.000000       3.600000       6.400000   32000.000000    0.000000
75%    2016.000000       6.000000       9.900000   48767.000000    0.000000
max    2018.000000      14.500000      92.600000  500000.000000    3.000000

         Car_Age  Price_per_km  Price_Age_Interaction
count 301.000000    301.000000             301.000000
mean    9.372093      0.000355              72.670193
std     2.891554      0.000589              97.689287
min     5.000000      0.000001               2.560000
25%     7.000000      0.000086              11.250000
50%     9.000000      0.000193              55.200000
75%    11.000000      0.000352              95.200000
max    20.000000      0.006038            1203.800000
/usr/local/lib/python3.10/dist-packages/numpy/lib/function_base.py:4824: UserWarning: Warning: 'partition' will ignore the 'mask' of the MaskedArray.
  arr.partition(
/usr/local/lib/python3.10/dist-packages/numpy/lib/function_base.py:4824: UserWarning: Warning: 'partition' will ignore the 'mask' of the MaskedArray.
  arr.partition(
/usr/local/lib/python3.10/dist-packages/numpy/lib/function_base.py:4824: UserWarning: Warning: 'partition' will ignore the 'mask' of the MaskedArray.
  arr.partition(
/usr/local/lib/python3.10/dist-packages/numpy/lib/function_base.py:4824: UserWarning: Warning: 'partition' will ignore the 'mask' of the MaskedArray.
  arr.partition(
```

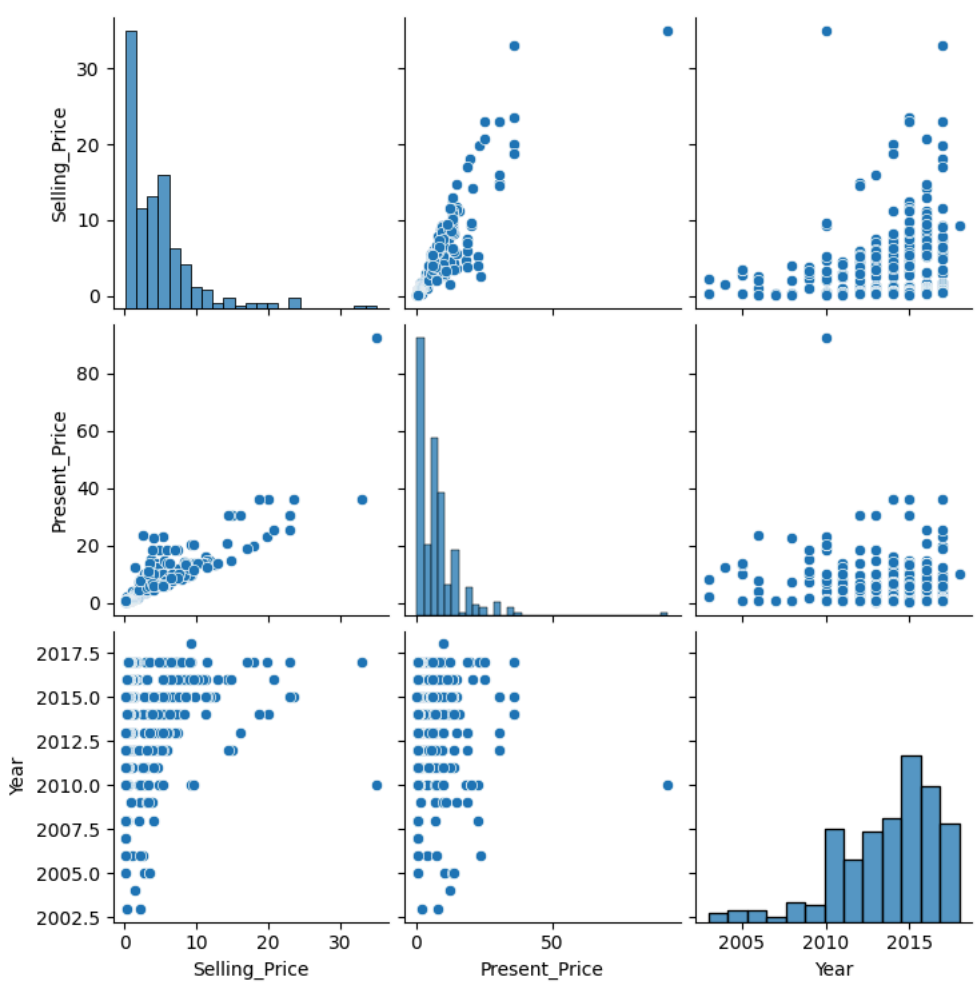
Check for missing values

```
print(data.isnull().sum())
```

```
Car_Name         0
Year             0
Selling_Price    0
Present_Price    0
Driven_kms       0
Fuel_Type        0
Selling_type     0
Transmission     0
Owner            0
dtype: int64
```

Explore relationships between variables

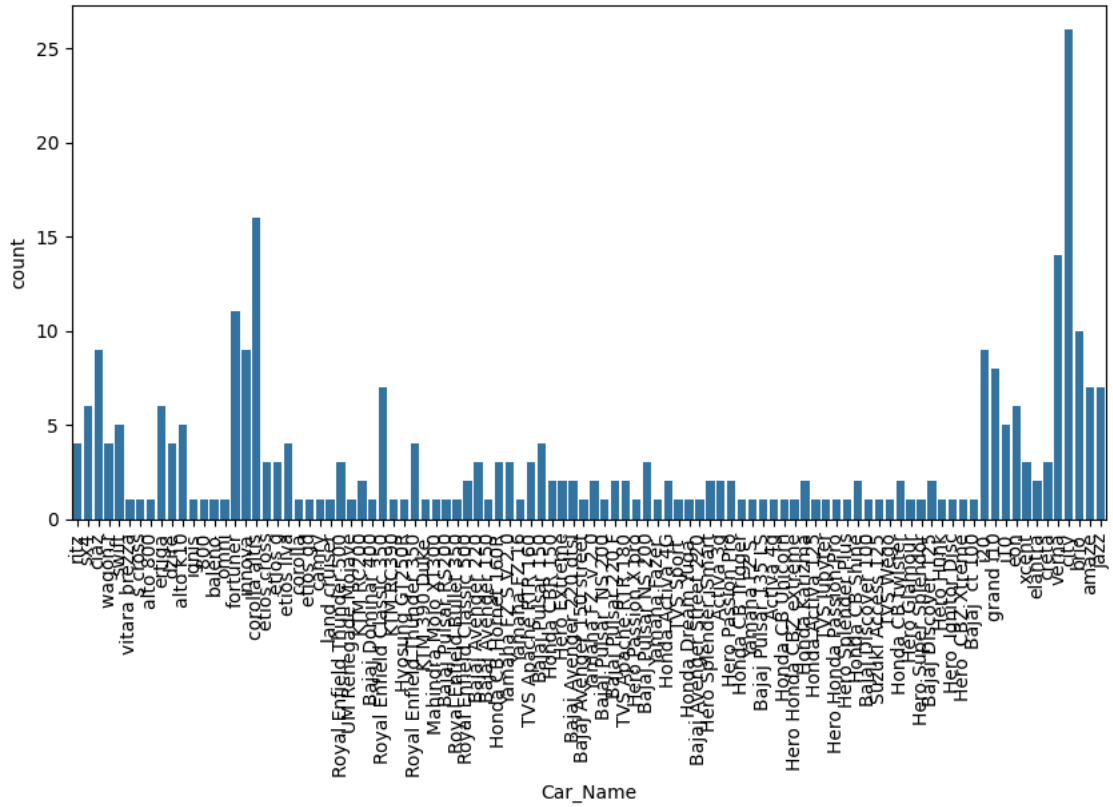
```
sns.pairplot(data[['Selling_Price', 'Present_Price', 'Year']])
plt.show()
```



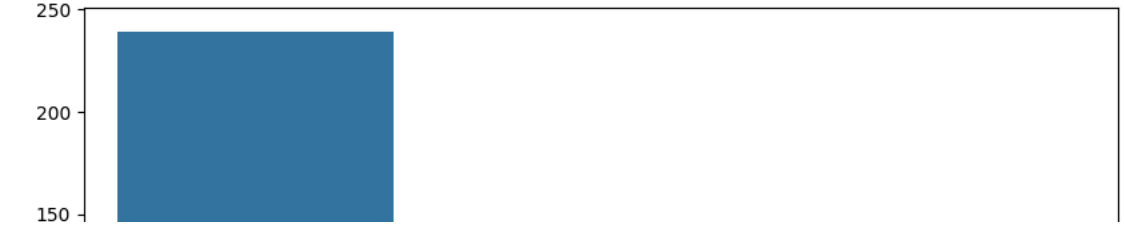
Analyze categorical features

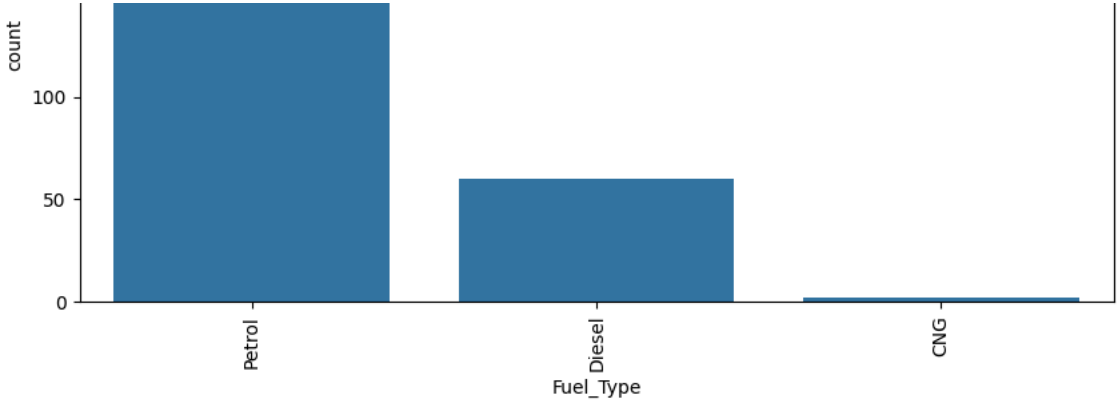
```
for col in ['Car_Name', 'Fuel_Type', 'Selling_type', 'Transmission']:
  print(data[col].value_counts())
  plt.figure(figsize=(10, 5))
  sns.countplot(x=col, data=data)
  plt.xticks(rotation=90)
  plt.show()
```

```
Car_Name
city                         26
corolla altis                16
verna                        14
fortuner                     11
brio                         10
                             ..
Honda CB Trigger              1
Yamaha FZ S                   1
Bajaj Pulsar 135 LS           1
Activa 4g                     1
Bajaj Avenger Street 220      1
Name: count, Length: 98, dtype: int64
```

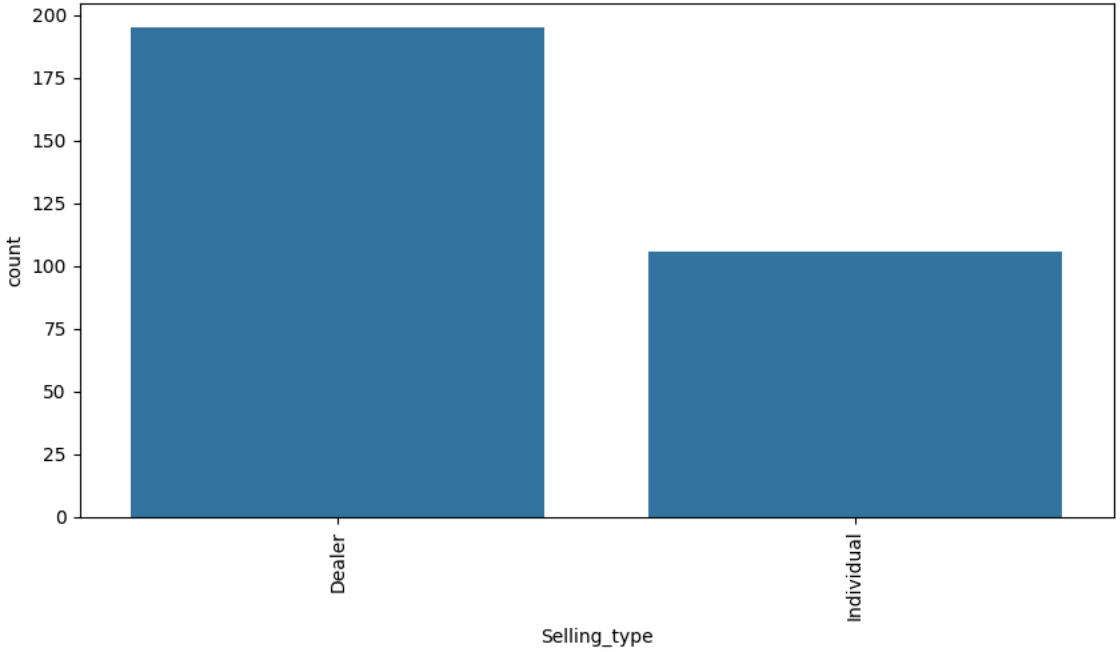



```
Fuel_Type
Petrol    239
Diesel     60
CNG         2
Name: count, dtype: int64
```

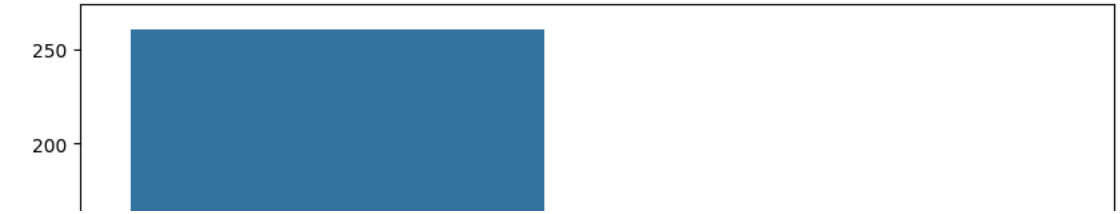


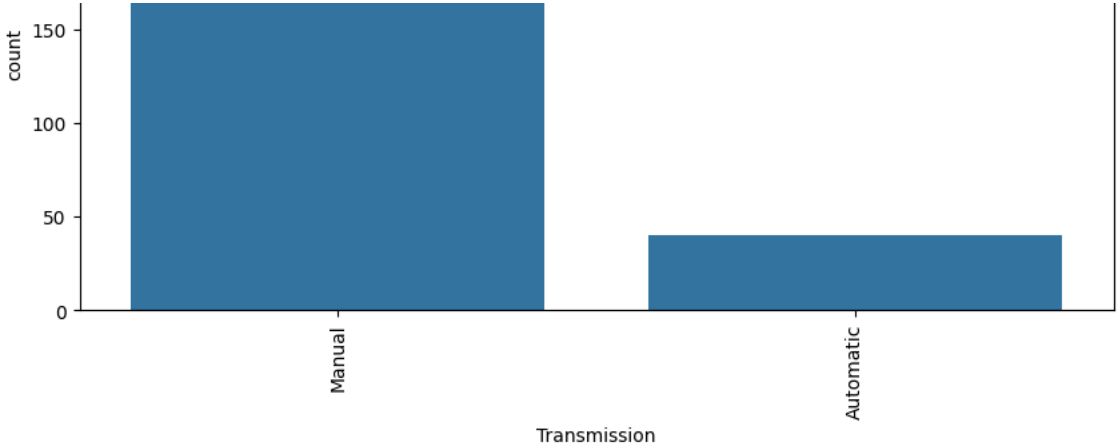



```
Selling_type
Dealer        195
Individual    106
Name: count, dtype: int64
```



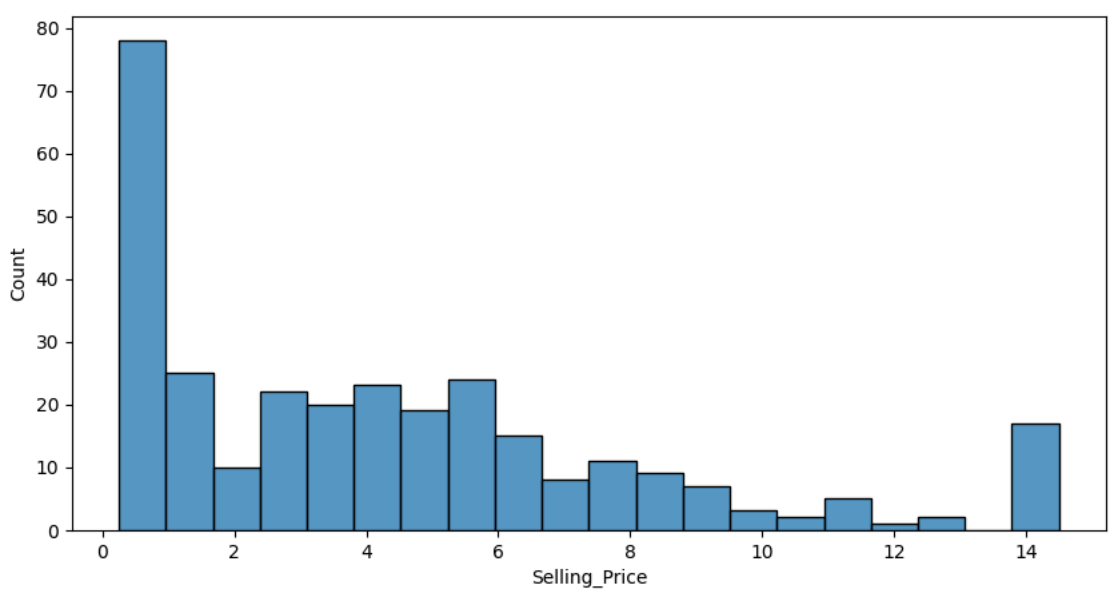

```
Transmission
Manual       261
Automatic     40
Name: count, dtype: int64
```




count
150
100
50
0
Manual
Automatic
Transmission

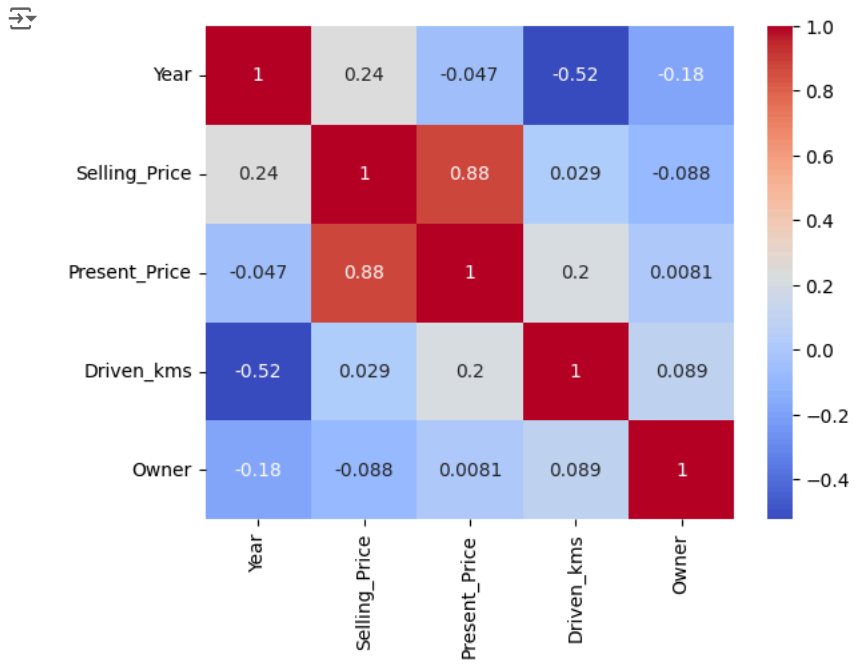
Analyze numerical features

```
plt.figure(figsize=(10, 5))
sns.histplot(data['Selling_Price'], bins=20)
plt.show()
```



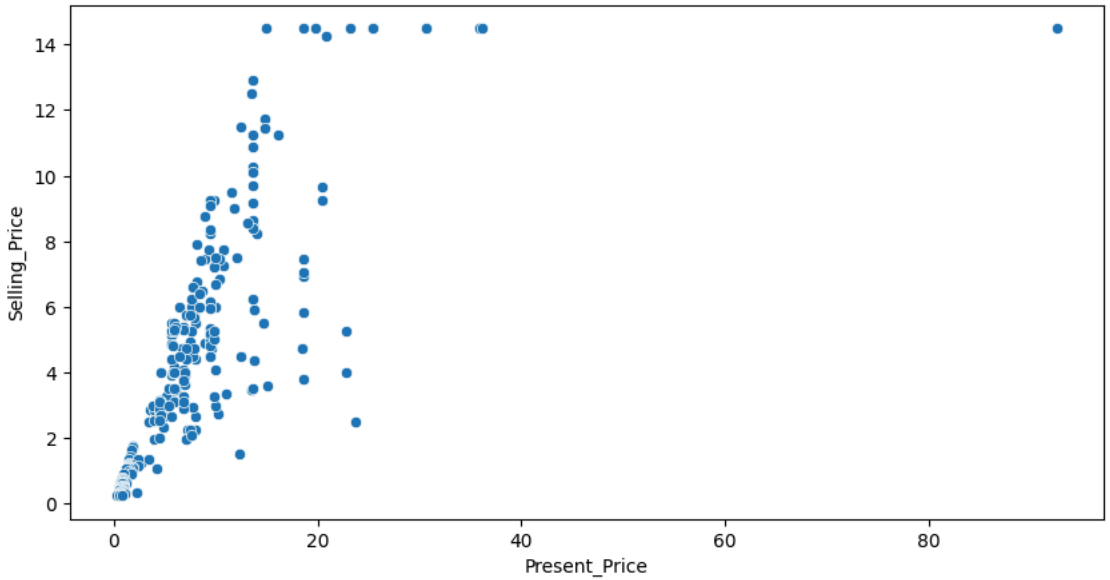
Correlation matrix

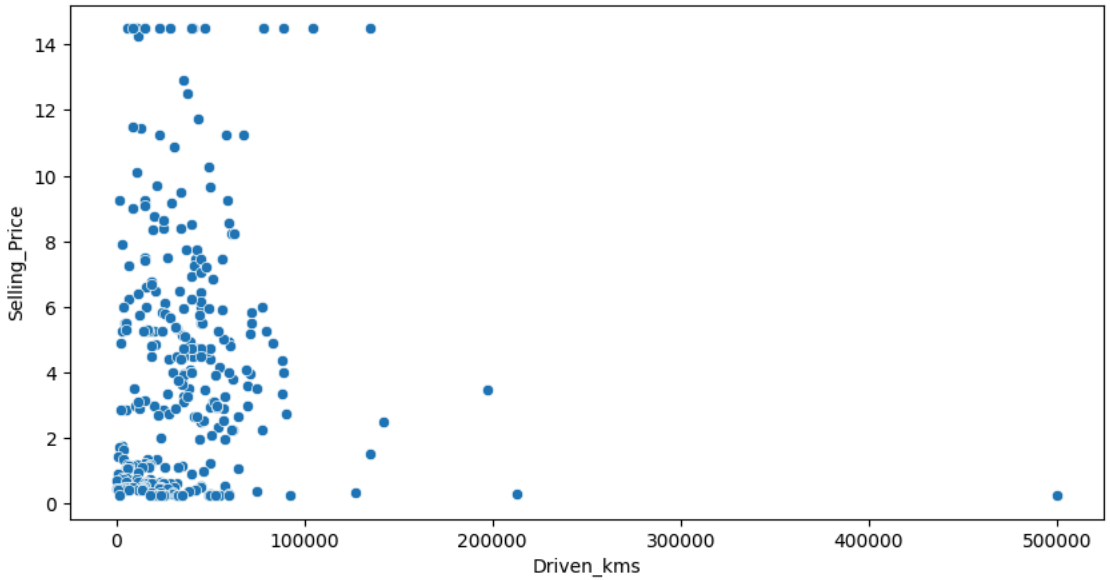
```
import numpy as np

correlation_matrix = data.select_dtypes(include=np.number).corr()
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.show()
```



Analyze the relationship between selling price and other features

```
plt.figure(figsize=(10, 5))
sns.scatterplot(x='Present_Price', y='Selling_Price', data=data)
plt.show()

plt.figure(figsize=(10, 5))
sns.scatterplot(x='Driven_kms', y='Selling_Price', data=data)
plt.show()
```



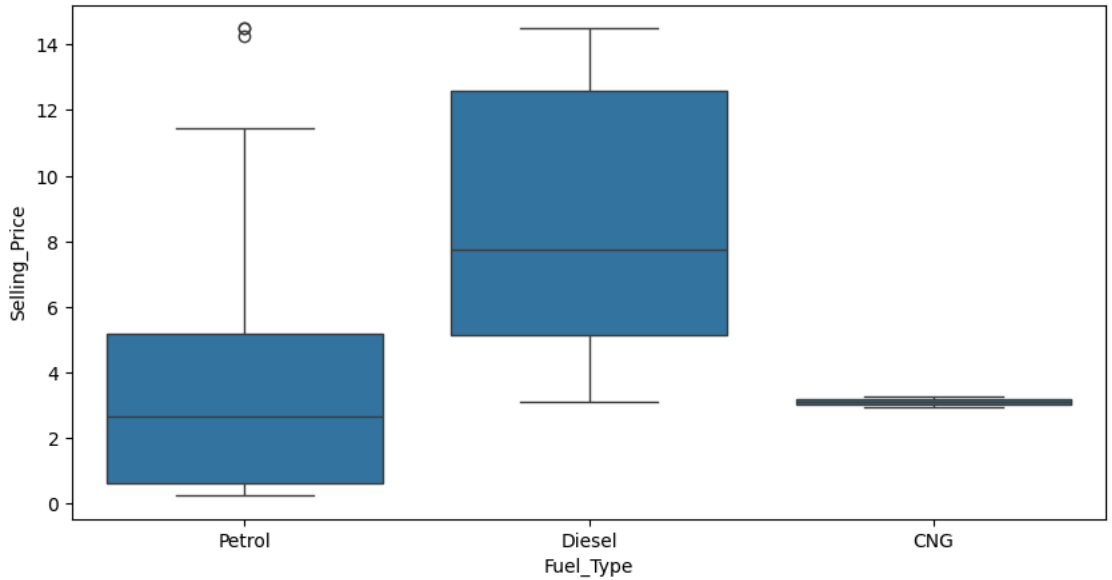


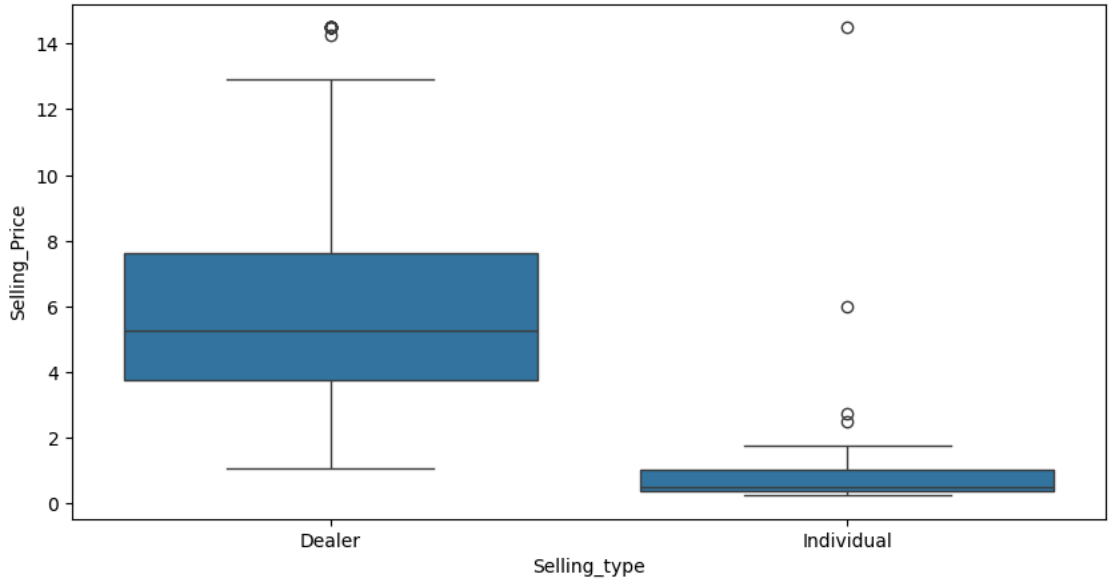
Analyze the impact of categorical features on selling price

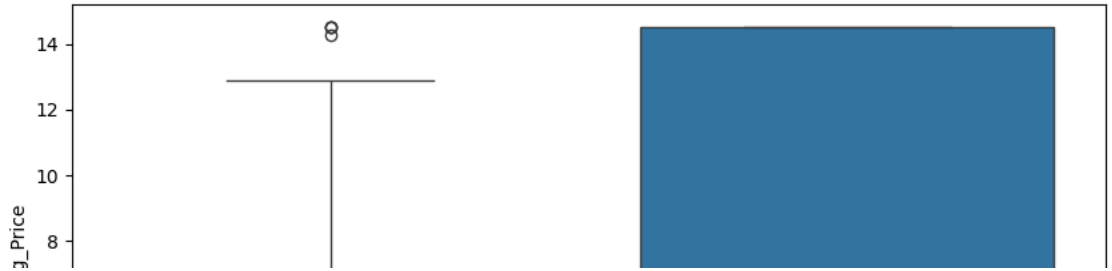
```
plt.figure(figsize=(10, 5))
sns.boxplot(x='Fuel_Type', y='Selling_Price', data=data)
plt.show()

plt.figure(figsize=(10, 5))
```

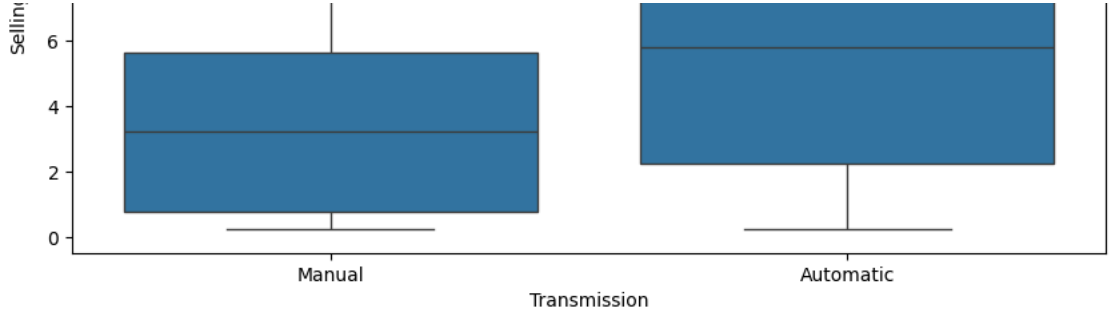
```
sns.boxplot(x='Selling_type', y='Selling_Price', data=data)
plt.show()

plt.figure(figsize=(10, 5))
sns.boxplot(x='Transmission', y='Selling_Price', data=data)
plt.show()
```

[4]


Selling_Price
0
2
4
6
8
10
12
14
Petrol
Diesel
CNG
Fuel_Type

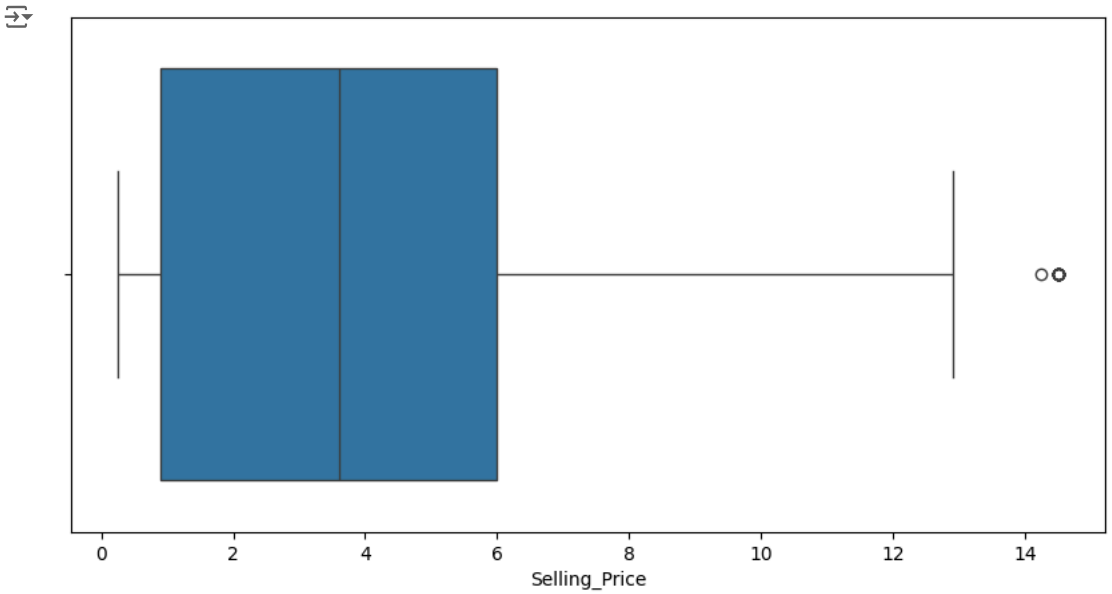

Selling_Price
0
2
4
6
8
10
12
14
Dealer
Individual
Selling_type


8
10
12
14



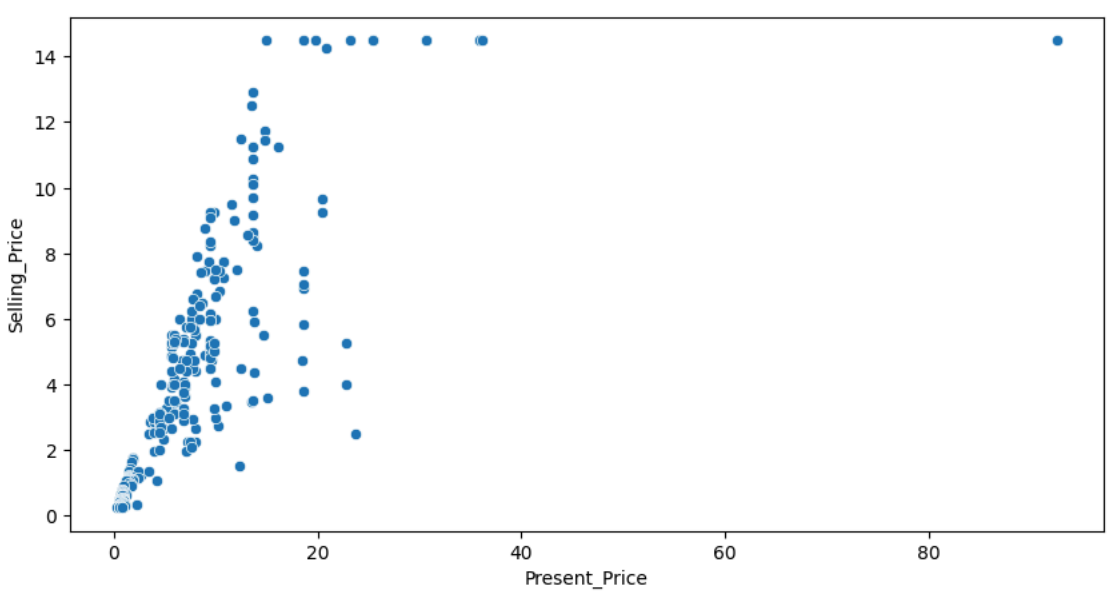
Identify outliers using box plots

```
plt.figure(figsize=(10, 5))
sns.boxplot(x='Selling_Price', data=data)
plt.show()
```



Identify outliers using scatter plots

```
plt.figure(figsize=(10, 5))
sns.scatterplot(x='Present_Price', y='Selling_Price', data=data)
plt.show()
```



Handle outliers

```
threshold = 3

# Calculate the mean and standard deviation

mean = data['Selling_Price'].mean()
std_dev = data['Selling_Price'].std()

# Identify outliers

outliers = data[(data['Selling_Price'] > mean + threshold * std_dev) |
                (data['Selling_Price'] < mean - threshold * std_dev)]
# Remove outliers
data_cleaned = data[~data.isin(outliers)].dropna()
```

Cap outliers

```
# Define upper and lower limits for outliers
upper_limit = data['Selling_Price'].mean() + 3 * data['Selling_Price'].std()
lower_limit = data['Selling_Price'].mean() - 3 * data['Selling_Price'].std()
# Cap outliers
data['Selling_Price'] = data['Selling_Price'].clip(lower=lower_limit, upper=upper_limit)
```

Winsorizing

```
!pip install scipy
from scipy.stats.mstats import winsorize
# Winsorize the data
data['Selling_Price'] = winsorize(data['Selling_Price'], limits=[0.05, 0.05])
```

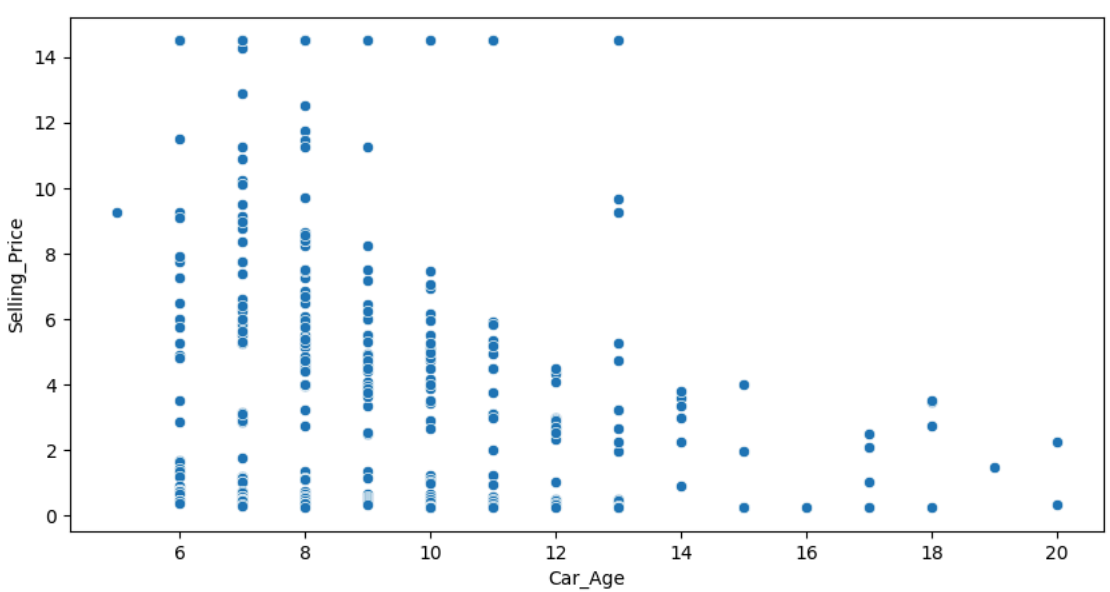
```
Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages (1.13.1)
Requirement already satisfied: numpy<2.3,>=1.22.4 in /usr/local/lib/python3.10/dist-packages (from scipy) (1.26.4)
```

Feature Engeniering

```
# Create a new feature for car age
data['Car_Age'] = 2023 - data['Year']

# Create a feature for price per kilometer driven
data['Price_per_km'] = data['Present_Price'] / data['Driven_kms']

# Create an interaction term between present price and car age
data['Price_Age_Interaction'] = data['Present_Price'] * data['Car_Age']
```

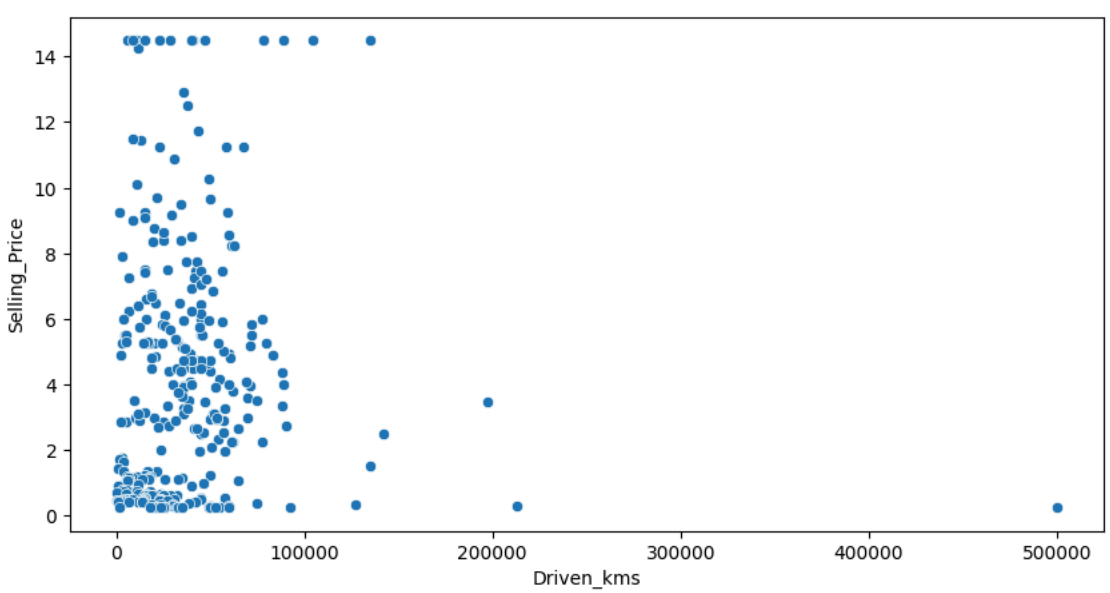
Analyze the relationship between car age and selling price

```
plt.figure(figsize=(10, 5))
sns.scatterplot(x='Car_Age', y='Selling_Price', data=data)
plt.show()
```



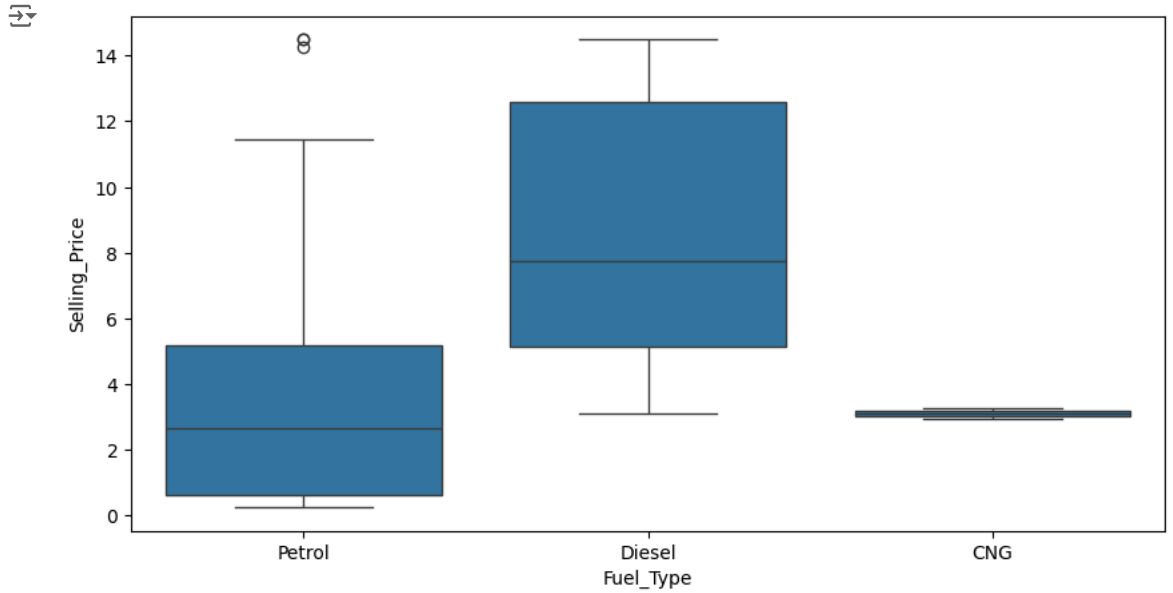
Analyze the relationship between kilometers driven and selling price

```
plt.figure(figsize=(10, 5))
sns.scatterplot(x='Driven_kms', y='Selling_Price', data=data)
plt.show()
```



Analyze the impact of fuel type on selling price

```
plt.figure(figsize=(10, 5))
sns.boxplot(x='Fuel_Type', y='Selling_Price', data=data)
plt.show()
```



Analyze the impact of transmission type on selling price

```
plt.figure(figsize=(10, 5))
sns.boxplot(x='Transmission', y='Selling_Price', data=data)
plt.show()
```