In [16]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [17]:
```
df=pd.read_csv('car_price_dataset_missing.csv')
df
```

Out[17]:

	Brand	Model	Year	Engine_Size	Fuel_Type	Transmission	Mileage	Doors	Owner_Count	Price
0	Kia	Rio	2020.0	4.2	Diesel	Manual	289944.0	3.0	5.0	8501.0
1	Chevrolet	Malibu	2012.0	2.0	Hybrid	Automatic	5356.0	2.0	3.0	12092.0
2	Mercedes	GLA	2020.0	NaN	Diesel	Automatic	231440.0	4.0	2.0	11171.0
3	Audi	Q5	2023.0	2.0	Electric	Manual	160971.0	2.0	1.0	11780.0
4	Volkswagen	Golf	2003.0	2.6	Hybrid	Semi-Automatic	286618.0	3.0	3.0	2867.0
...	...	...	...	...	...	...	...	...	...	...
9995	Kia	Optima	2004.0	3.7	NaN	Semi-Automatic	5794.0	2.0	4.0	8884.0
9996	Chevrolet	Impala	2002.0	1.4	Electric	Automatic	168000.0	2.0	1.0	6240.0
9997	BMW	NaN	2010.0	3.0	Petrol	Automatic	86664.0	5.0	1.0	9866.0
9998	Ford	Explorer	2002.0	1.4	Hybrid	Automatic	225772.0	4.0	1.0	4084.0
9999	Volkswagen	NaN	2001.0	NaN	Diesel	Manual	157882.0	3.0	3.0	3342.0

10000 rows × 10 columns

In [18]:
```
df.shape
```

Out[18]:
```
(10000, 10)
```

In [19]:
```
df.describe()
```

Out[19]:

	Year	Engine_Size	Mileage	Doors	Owner_Count	Price
count	8975.000000	8994.000000	9062.000000	8968.000000	9039.000000	8925.000000
mean	2011.563565	2.997465	148950.282719	3.495540	2.987167	8852.541737
std	6.908484	1.148929	86391.746111	1.111284	1.422891	3117.021705
min	2000.000000	1.000000	25.000000	2.000000	1.000000	2000.000000
25%	2006.000000	2.000000	74484.750000	3.000000	2.000000	6657.000000
50%	2012.000000	3.000000	149461.500000	3.000000	3.000000	8853.000000
75%	2018.000000	4.000000	223338.250000	4.000000	4.000000	11091.000000
max	2023.000000	5.000000	299947.000000	5.000000	5.000000	18301.000000

In [20]:
```
df.isnull().sum()
```

Out[20]:
```
Brand           1041
Model            959
Year            1025
Engine_Size     1006
Fuel_Type        974
Transmission     989
Mileage          938
Doors           1032
Owner_Count      961
Price           1075
dtype: int64
```

In [21]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 10 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Brand         8959 non-null   object 
 1   Model         9041 non-null   object 
 2   Year          8975 non-null   float64
 3   Engine_Size   8994 non-null   float64
 4   Fuel_Type     9026 non-null   object 
 5   Transmission  9011 non-null   object 
 6   Mileage       9062 non-null   float64
 7   Doors         8968 non-null   float64
 8   Owner_Count   9039 non-null   float64
 9   Price         8925 non-null   float64
dtypes: float64(6), object(4)
memory usage: 781.4+ KB
```

In [29]:
```
df.dtypes
```

Out[29]:
```
Brand           object
Model           object
Year           float64
Engine_Size    float64
Fuel_Type       object
Transmission    object
Mileage        float64
Doors          float64
Owner_Count    float64
Price          float64
dtype: object
```

In [31]:
```
df.Brand.value_counts()
```

Out[31]:
```
Audi          929
Ford          925
Volkswagen    908
Honda         907
BMW           902
Chevrolet     892
Hyundai       891
Kia           890
Toyota        873
Mercedes      842
Name: Brand, dtype: int64
```

In [33]:
```
df.Model.value_counts()
```

Out[33]:
```
Accord      332
Impala      328
Fiesta      324
Elantra     320
Tiguan      320
Q5          318
Focus       316
A4          315
Golf        308
Optima      308
5 Series    307
A3          305
Malibu      305
3 Series    305
Explorer    304
Civic       300
E-Class     297
RAV4        296
Corolla     295
Camry       294
Tucson      293
Passat      292
Rio         292
Equinox     290
GLA         283
X5          283
Sonata      282
Sportage    280
CR-V        279
C-Class     270
Name: Model, dtype: int64
```

In [35]:
```
df.Year.value_counts()
```

Out[35]:
```
2002.0    411
2011.0    406
2023.0    402
2018.0    392
2015.0    391
2012.0    390
2005.0    382
2010.0    380
2007.0    379
2013.0    378
2017.0    378
2019.0    375
2006.0    374
2020.0    371
2014.0    370
2022.0    365
2009.0    364
2001.0    363
2000.0    360
2021.0    359
2004.0    349
2016.0    348
2008.0    348
2003.0    340
Name: Year, dtype: int64
```

In [37]:
```
df.Engine_Size.value_counts()
```

Out[37]:

2.5	246
1.5	243
1.4	242
3.7	241
3.3	240
4.4	240
4.0	239
3.6	239
3.5	238
3.1	237
1.6	233
2.0	233
1.3	233
3.2	232
4.7	231
2.6	230
4.6	229
2.8	229
1.8	228
4.5	227
4.1	225
3.4	224
4.9	224
1.7	222
2.7	222
1.2	222
2.2	221
3.8	220
2.9	218
2.4	217
4.3	214
3.0	212
1.9	211
3.9	211
1.1	210
2.1	208
4.2	207
4.8	206
2.3	199
1.0	99

```
5.0     92
Name: Engine_Size, dtype: int64
```

In [39]:
```
df.Fuel_Type.value_counts()
```

Out[39]:
```
Electric    2360
Diesel      2278
Hybrid      2219
Petrol      2169
Name: Fuel_Type, dtype: int64
```

In [41]:
```
df.Transmission.value_counts()
```

Out[41]:
```
Manual            3029
Automatic         2995
Semi-Automatic    2987
Name: Transmission, dtype: int64
```

In [45]:
```
df.Mileage.value_counts()
```

Out[45]:
```
230974.0    3
82111.0     2
143095.0    2
79442.0     2
74760.0     2
           ..
208070.0    1
81504.0     1
182254.0    1
94922.0     1
157882.0    1
Name: Mileage, Length: 8946, dtype: int64
```

In [47]:
```
df.Doors.value_counts()
```

Out[47]:
```
3.0    2295
4.0    2257
2.0    2215
5.0    2201
Name: Doors, dtype: int64
```

In [49]:
```
df.Owner_Count.value_counts()
```

Out[49]:
```
1.0    1849
5.0    1834
2.0    1827
3.0    1788
4.0    1741
Name: Owner_Count, dtype: int64
```

In [51]:
```
df.Price.value_counts()
```

Out[51]:
```
2000.0     80
9189.0      7
8217.0      7
9754.0      6
10267.0     5
           ..
14186.0     1
7180.0      1
5351.0      1
10181.0     1
3342.0      1
Name: Price, Length: 6180, dtype: int64
```

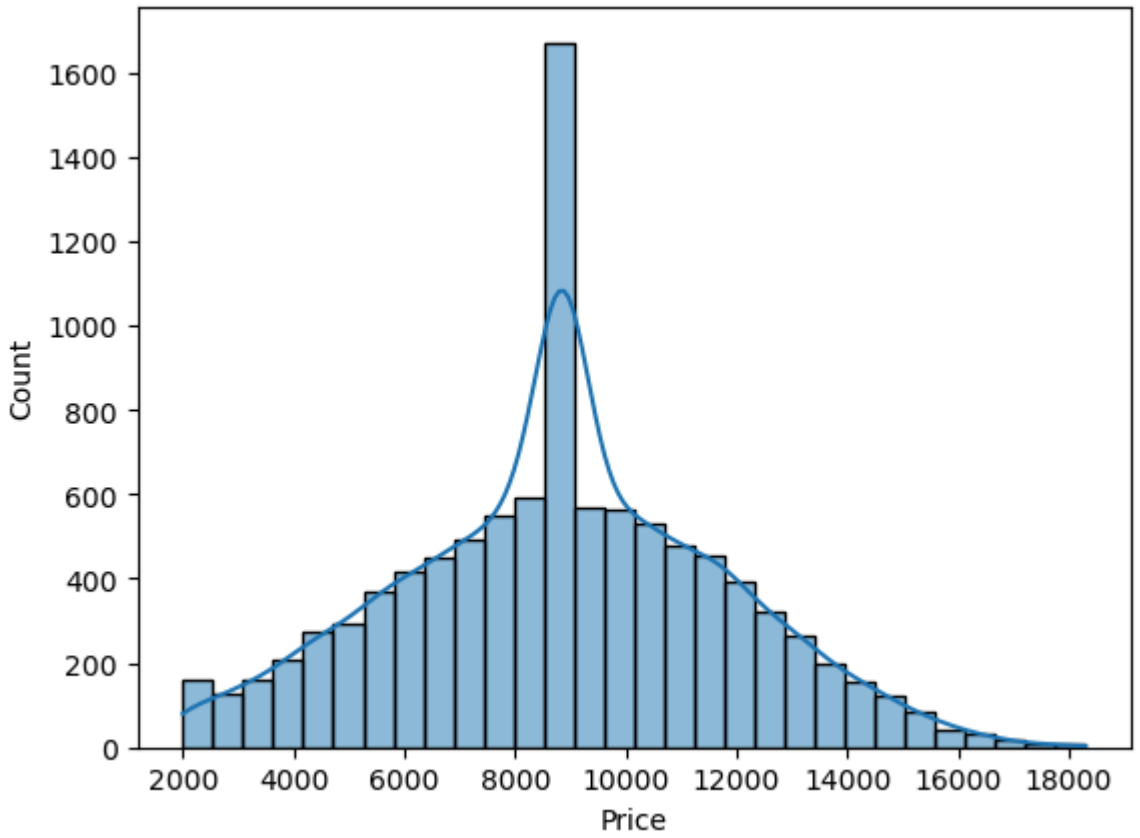
In [53]:
```
categorical_cols = ['Brand', 'Model', 'Fuel_Type', 'Transmission']
for col in categorical_cols:
    df[col].fillna(df[col].mode()[0], inplace=True)
```

In [55]:
```
numerical_cols = ['Year', 'Engine_Size', 'Mileage', 'Doors', 'Owner_Count', 'Price']
for col in numerical_cols:
    df[col].fillna(df[col].mean(), inplace=True)
```

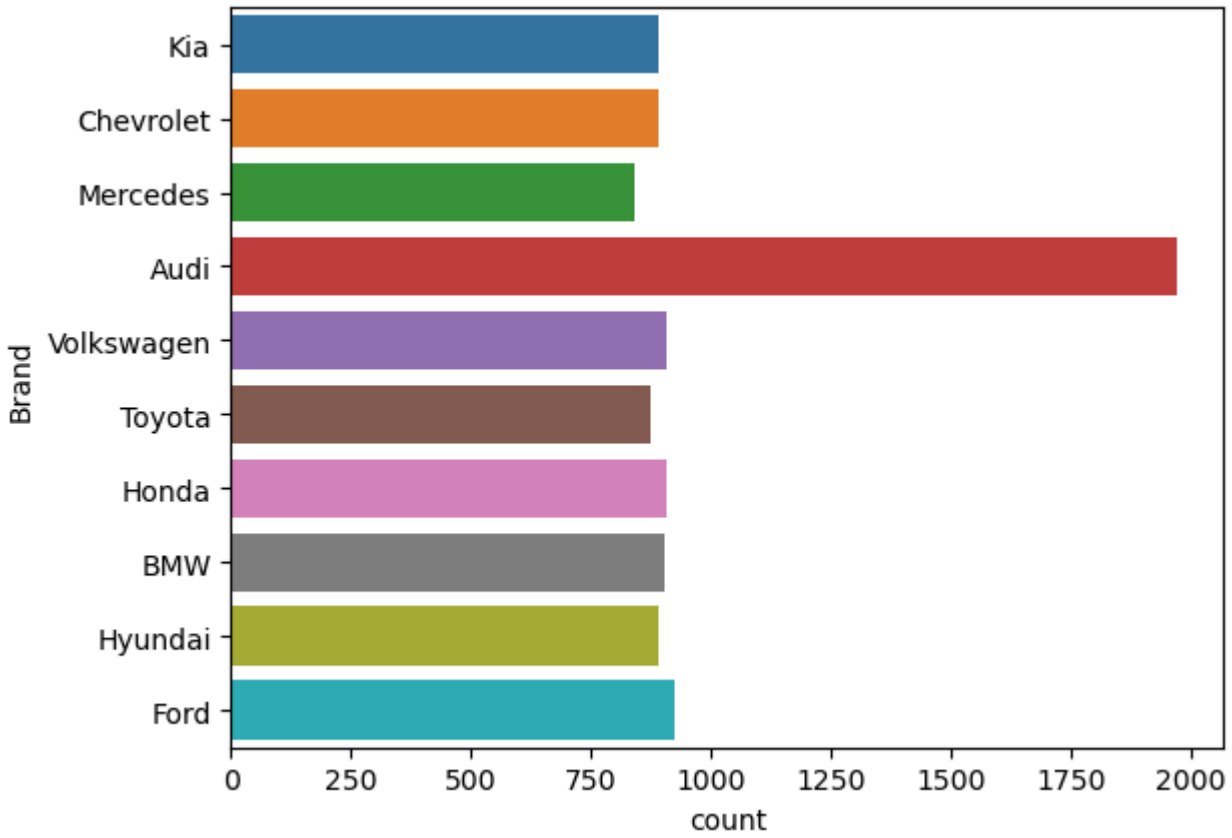
In [57]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 10 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Brand         10000 non-null  object 
 1   Model         10000 non-null  object 
 2   Year          10000 non-null  float64
 3   Engine_Size   10000 non-null  float64
 4   Fuel_Type     10000 non-null  object 
 5   Transmission  10000 non-null  object 
 6   Mileage       10000 non-null  float64
 7   Doors         10000 non-null  float64
 8   Owner_Count   10000 non-null  float64
 9   Price         10000 non-null  float64
dtypes: float64(6), object(4)
memory usage: 781.4+ KB
```

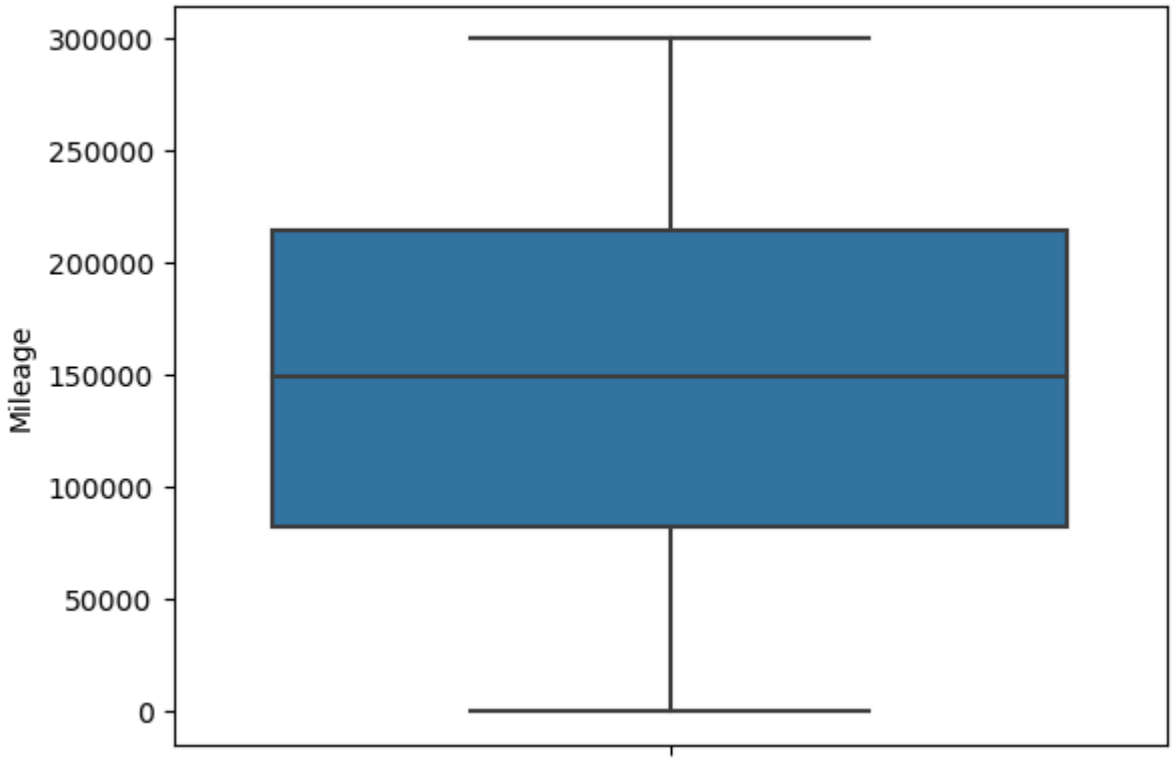
In [59]:
```
sns.histplot(df['Price'], bins=30, kde=True)
plt.show()
```



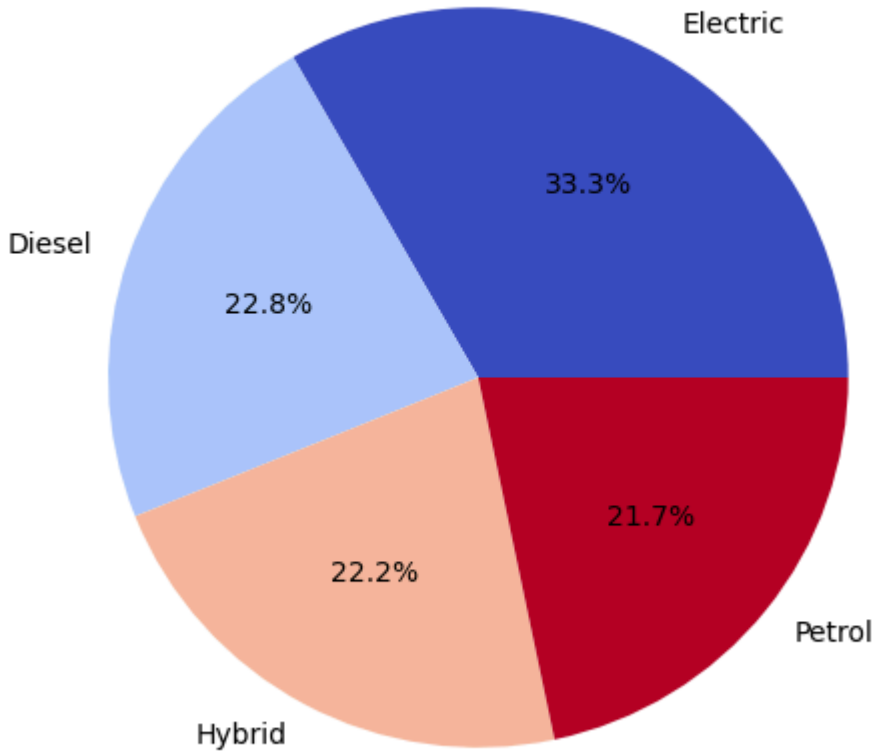
In [67]:
```
sns.countplot(y=df['Brand'])
plt.show()
```



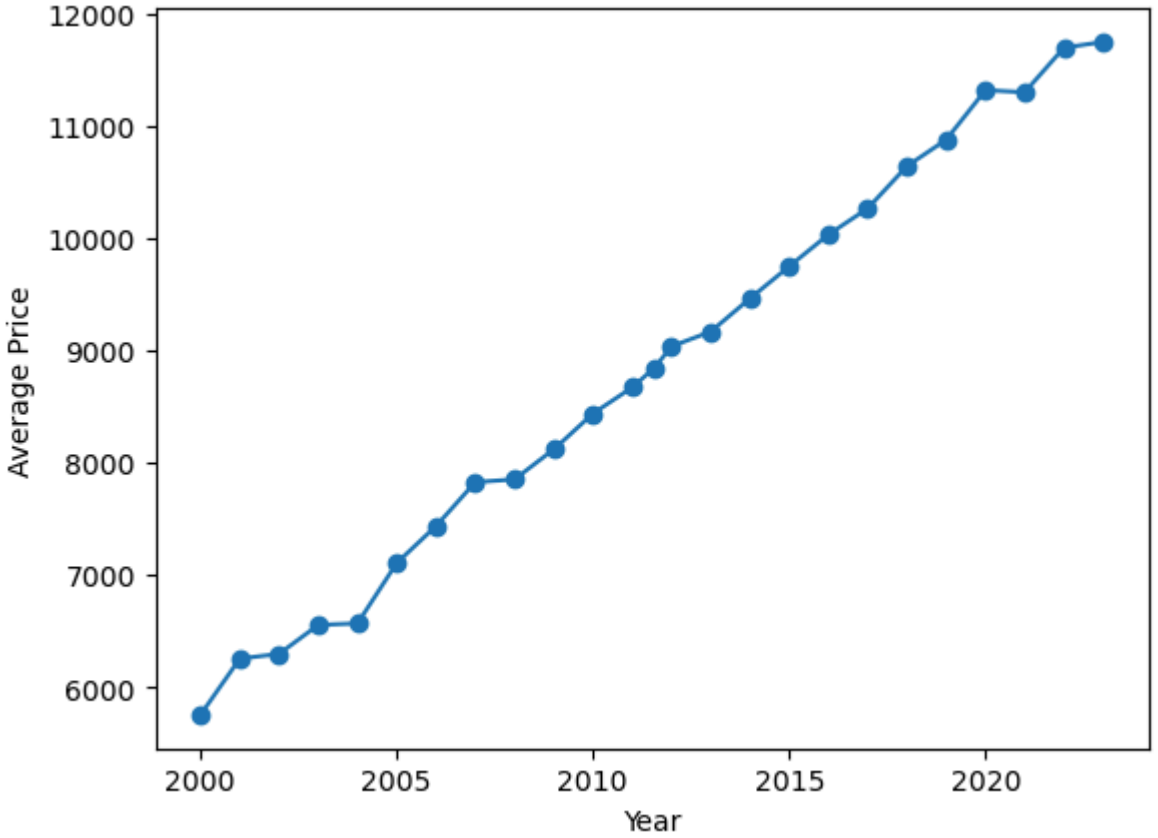
In [73]:
```
sns.boxplot(y=df['Mileage'])
plt.show()
```



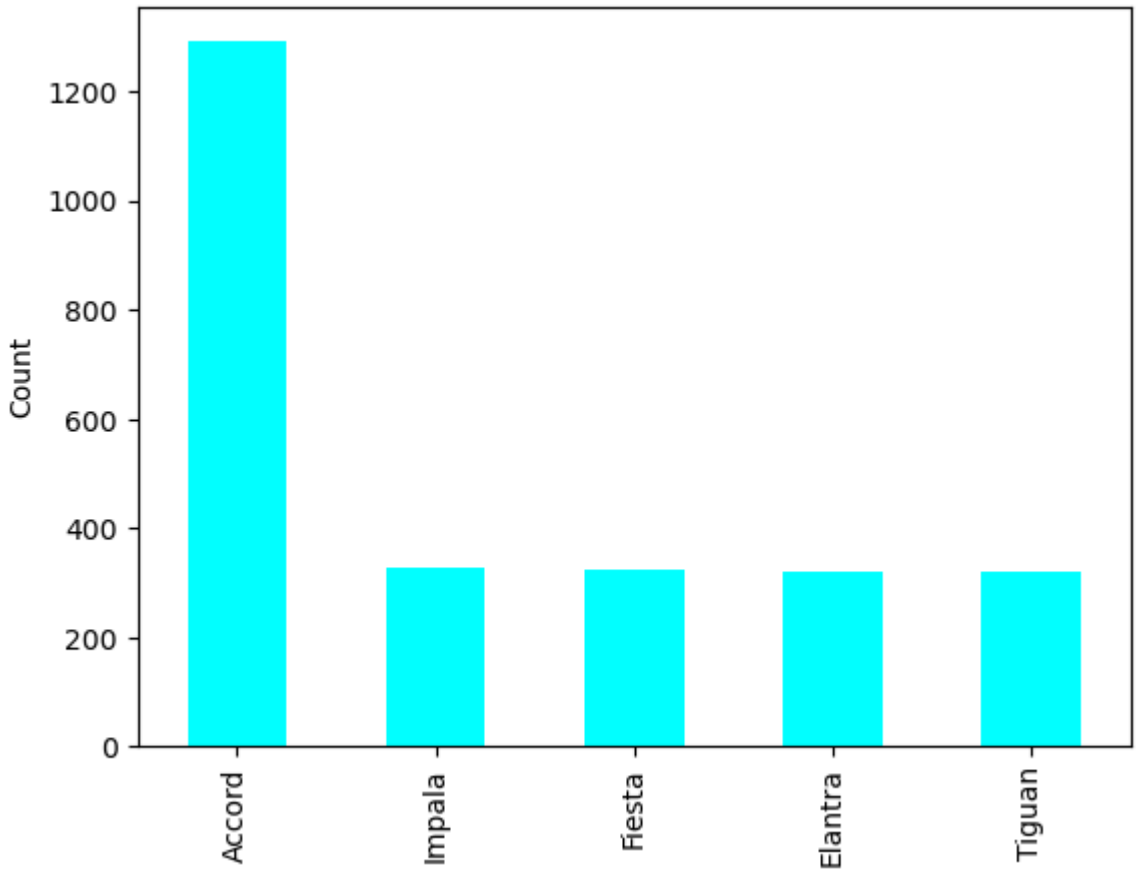
In [75]:
```
df['Fuel_Type'].value_counts().plot.pie(autopct='%1.1f%%', cmap='coolwarm', figsize=(6, 6))
plt.ylabel('')
plt.show()
```



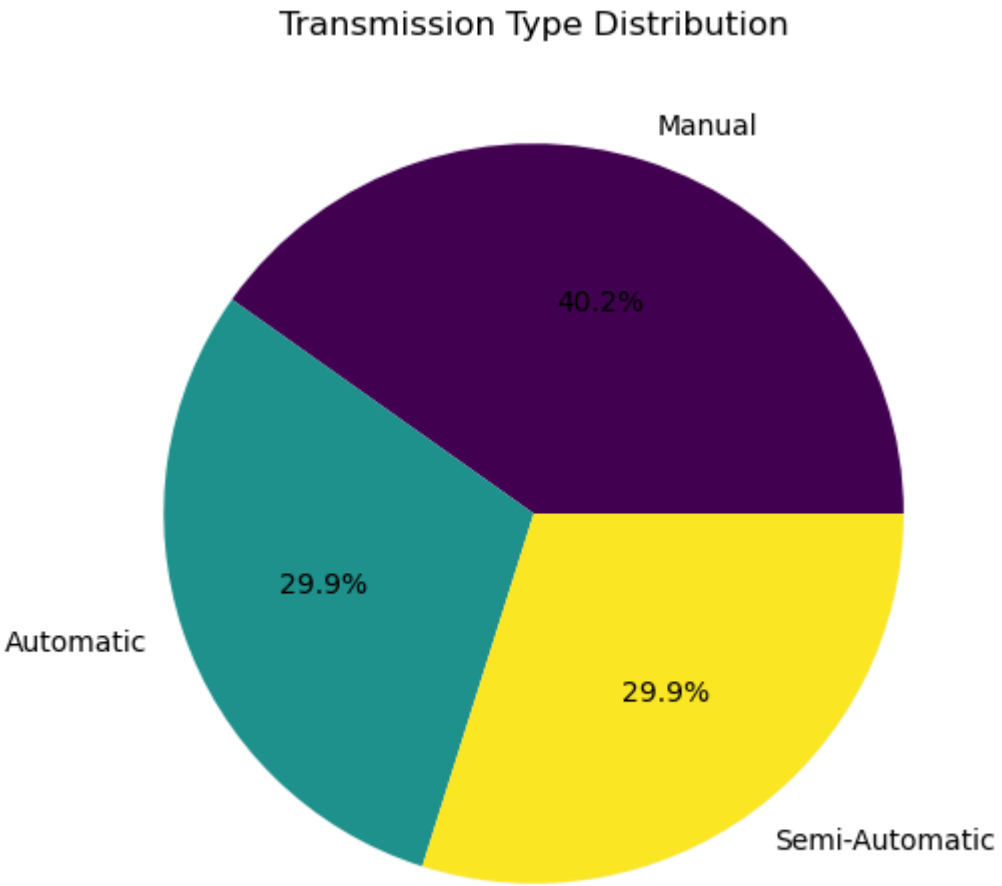
In [77]:
```
df.groupby('Year')['Price'].mean().plot(kind='line', marker='o')
plt.ylabel('Average Price')
plt.show()
```



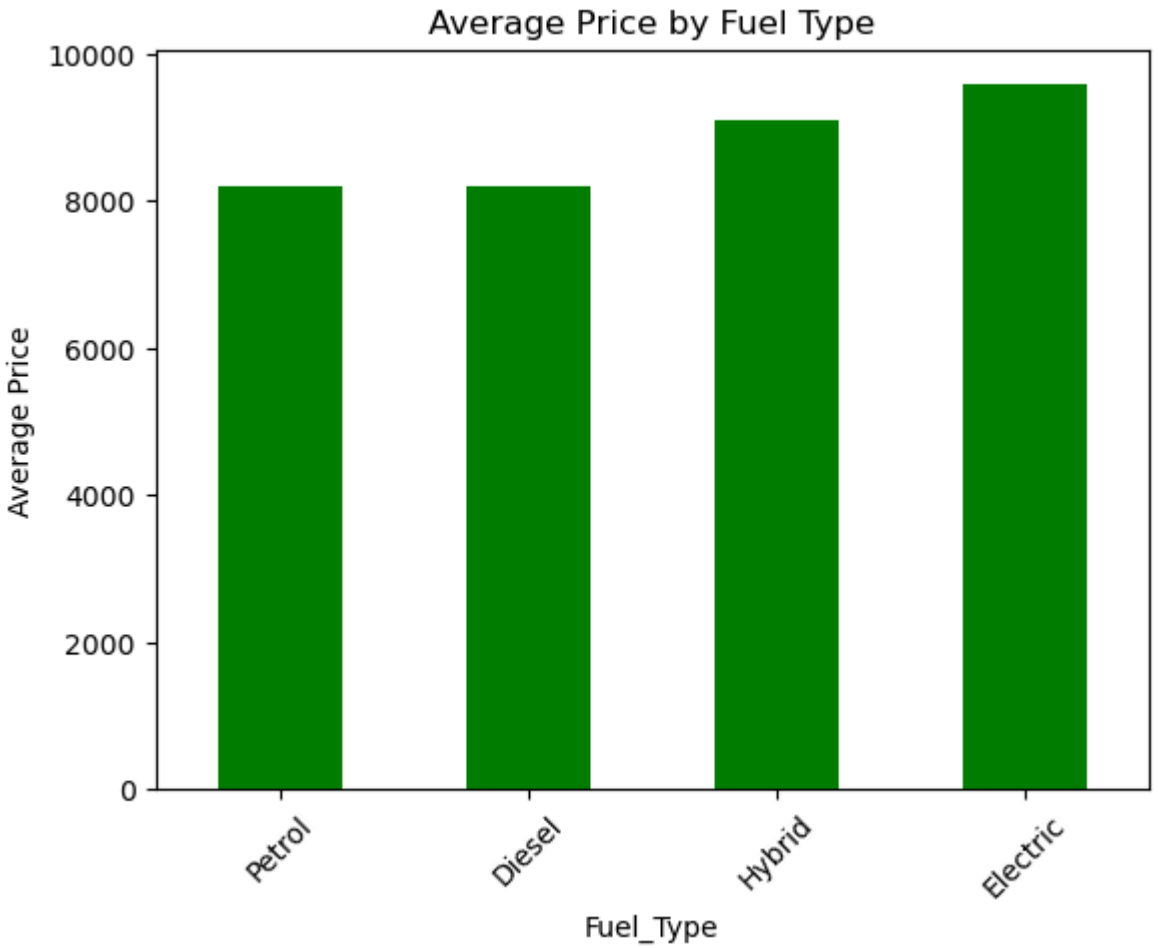
In [85]:
```
df['Model'].value_counts()[:5].plot(kind='bar', color='cyan')
plt.ylabel('Count')
plt.show()
```



In [95]:
```
df['Transmission'].value_counts().plot.pie(autopct='%1.1f%%', cmap='viridis', figsize=(6, 6))
plt.ylabel('')
plt.title("Transmission Type Distribution")
plt.show()
```

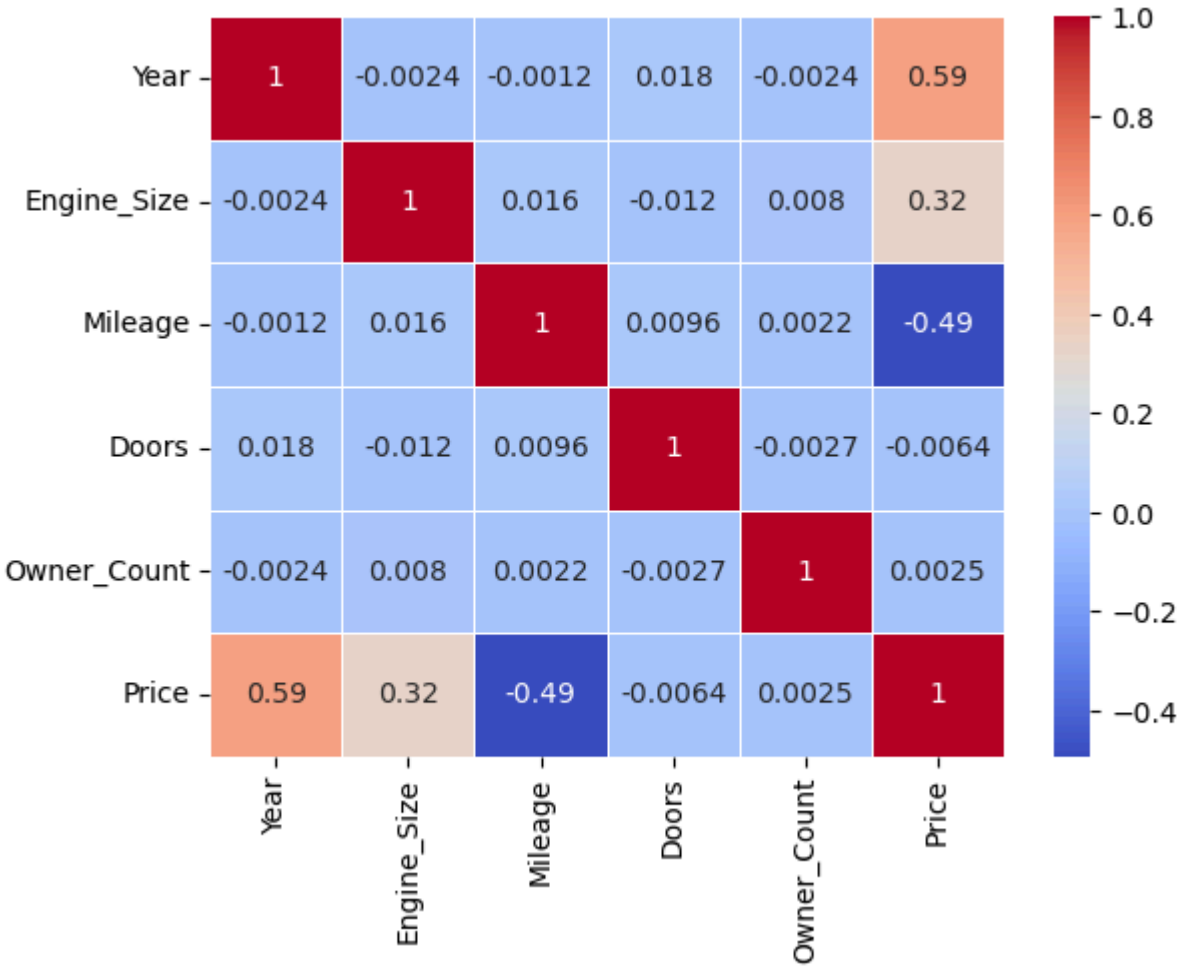


In [99]:
```
df.groupby('Fuel_Type')['Price'].mean().sort_values().plot(kind='bar', color='green')
plt.ylabel("Average Price")
plt.title("Average Price by Fuel Type")
plt.xticks(rotation=45)
plt.show()
```



In [101]:
```
sns.heatmap(df.corr(), annot=True, cmap='coolwarm', linewidths=0.5)
plt.show()
```

```
C:\Users\subha\AppData\Local\Temp\ipykernel_22788\1760633978.py:1: FutureWarning: The default value of numeric_only in
DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the v
alue of numeric_only to silence this warning.
  sns.heatmap(df.corr(), annot=True, cmap='coolwarm', linewidths=0.5)
```



In [103]:
```
brand=pd.get_dummies(df['Brand'],prefix="Brand")
model=pd.get_dummies(df['Model'],prefix="Model")
fuel_type=pd.get_dummies(df['Fuel_Type'],prefix="Fuel_Type")
transmission=pd.get_dummies(df['Transmission'],prefix="Transmission")
```

In [105]:
```
print(brand)
```

```
      Brand_Audi  Brand_BMW  Brand_Chevrolet  Brand_Ford  Brand_Honda  \
0              0          0                0           0            0
1              0          0                1           0            0
2              0          0                0           0            0
3              1          0                0           0            0
4              0          0                0           0            0
...          ...        ...              ...         ...          ...
9995           0          0                0           0            0
9996           0          0                1           0            0
9997           0          1                0           0            0
9998           0          0                0           1            0
9999           0          0                0           0            0

      Brand_Hyundai  Brand_Kia  Brand_Mercedes  Brand_Toyota  Brand_Volkswagen
0                 0          1               0             0                 0
1                 0          0               0             0                 0
2                 0          0               1             0                 0
3                 0          0               0             0                 0
4                 0          0               0             0                 1
...             ...        ...             ...           ...               ...
9995              0          1               0             0                 0
9996              0          0               0             0                 0
9997              0          0               0             0                 0
9998              0          0               0             0                 0
9999              0          0               0             0                 1

[10000 rows x 10 columns]
```

In [107]:
```
print(model)
```

```
      Model_3 Series  Model_5 Series  Model_A3  Model_A4  Model_Accord  \
0                  0               0         0         0             0
1                  0               0         0         0             0
2                  0               0         0         0             0
3                  0               0         0         0             0
4                  0               0         0         0             0
...              ...             ...       ...       ...           ...
9995               0               0         0         0             0
9996               0               0         0         0             0
9997               0               0         0         0             1
9998               0               0         0         0             0
9999               0               0         0         0             1

      Model_C-Class  Model_CR-V  Model_Camry  Model_Civic  Model_Corolla  ...  \
0                 0           0            0            0              0  ...
1                 0           0            0            0              0  ...
2                 0           0            0            0              0  ...
3                 0           0            0            0              0  ...
4                 0           0            0            0              0  ...
...             ...         ...          ...          ...            ...  ...
9995              0           0            0            0              0  ...
9996              0           0            0            0              0  ...
9997              0           0            0            0              0  ...
9998              0           0            0            0              0  ...
9999              0           0            0            0              0  ...

      Model_Optima  Model_Passat  Model_Q5  Model_RAV4  Model_Rio  \
0                0             0         0           0          1
1                0             0         0           0          0
2                0             0         0           0          0
3                0             0         1           0          0
4                0             0         0           0          0
...            ...           ...       ...         ...        ...
9995             1             0         0           0          0
9996             0             0         0           0          0
9997             0             0         0           0          0
9998             0             0         0           0          0
9999             0             0         0           0          0

      Model_Sonata  Model_Sportage  Model_Tiguan  Model_Tucson  Model_X5
0                0               0             0             0         0
```

```
1                0              0              0              0         0
2                0              0              0              0         0
3                0              0              0              0         0
4                0              0              0              0         0
...            ...            ...            ...            ...       ...
9995             0              0              0              0         0
9996             0              0              0              0         0
9997             0              0              0              0         0
9998             0              0              0              0         0
9999             0              0              0              0         0

[10000 rows x 30 columns]
```

In [109]:
```
print(fuel_type)
```

```
      Fuel_Type_Diesel  Fuel_Type_Electric  Fuel_Type_Hybrid  Fuel_Type_Petrol
0                    1                   0                 0                 0
1                    0                   0                 1                 0
2                    1                   0                 0                 0
3                    0                   1                 0                 0
4                    0                   0                 1                 0
...                ...                 ...               ...               ...
9995                 0                   1                 0                 0
9996                 0                   1                 0                 0
9997                 0                   0                 0                 1
9998                 0                   0                 1                 0
9999                 1                   0                 0                 0

[10000 rows x 4 columns]
```

In [111]:
```
print(transmission)
```

```
      Transmission_Automatic  Transmission_Manual  Transmission_Semi-Automatic
0                          0                    1                            0
1                          1                    0                            0
2                          1                    0                            0
3                          0                    1                            0
4                          0                    0                            1
...                      ...                  ...                          ...
9995                       0                    0                            1
9996                       1                    0                            0
9997                       1                    0                            0
9998                       1                    0                            0
9999                       0                    1                            0

[10000 rows x 3 columns]
```

In [113]:
```
df.drop(["Brand","Model","Fuel_Type","Transmission"],axis=1,inplace=True)
df=pd.concat([df,brand,model,fuel_type,transmission],axis=1)
df
```

Out[113]:

	Year	Engine_Size	Mileage	Doors	Owner_Count	Price	Brand_Audi	Brand_BMW	Brand_Chevrolet	Brand_Ford	...	Model_Tiguan	Mode
0	2020.0	4.200000	289944.0	3.0	5.0	8501.0	0	0	0	0	...	0	
1	2012.0	2.000000	5356.0	2.0	3.0	12092.0	0	0	1	0	...	0	
2	2020.0	2.997465	231440.0	4.0	2.0	11171.0	0	0	0	0	...	0	
3	2023.0	2.000000	160971.0	2.0	1.0	11780.0	1	0	0	0	...	0	
4	2003.0	2.600000	286618.0	3.0	3.0	2867.0	0	0	0	0	...	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	
9995	2004.0	3.700000	5794.0	2.0	4.0	8884.0	0	0	0	0	...	0	
9996	2002.0	1.400000	168000.0	2.0	1.0	6240.0	0	0	1	0	...	0	
9997	2010.0	3.000000	86664.0	5.0	1.0	9866.0	0	1	0	0	...	0	
9998	2002.0	1.400000	225772.0	4.0	1.0	4084.0	0	0	0	1	...	0	
9999	2001.0	2.997465	157882.0	3.0	3.0	3342.0	0	0	0	0	...	0	

10000 rows × 53 columns

In [137]:
```
from sklearn.linear_model import LinearRegression
from sklearn import linear_model
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score, mean_squared_error
```

In [117]:
```
X = df.drop("Price", axis=1)
y = df["Price"]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

In [119]:
```
reg = linear_model.LinearRegression()
reg.fit(X_train, y_train)
```

Out[119]:
```
▾ LinearRegression

LinearRegression()
```

In [129]:
```
X_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 8000 entries, 9254 to 7270
Data columns (total 52 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   Year                   8000 non-null   float64
 1   Engine_Size            8000 non-null   float64
 2   Mileage                8000 non-null   float64
 3   Doors                  8000 non-null   float64
 4   Owner_Count            8000 non-null   float64
 5   Brand_Audi             8000 non-null   uint8  
 6   Brand_BMW              8000 non-null   uint8  
 7   Brand_Chevrolet        8000 non-null   uint8  
 8   Brand_Ford             8000 non-null   uint8  
 9   Brand_Honda            8000 non-null   uint8  
 10  Brand_Hyundai          8000 non-null   uint8  
 11  Brand_Kia              8000 non-null   uint8  
 12  Brand_Mercedes         8000 non-null   uint8  
 13  Brand_Toyota           8000 non-null   uint8  
 14  Brand_Volkswagen       8000 non-null   uint8  
 15  Model_3 Series         8000 non-null   uint8  
 16  Model_5 Series         8000 non-null   uint8  
 17  Model_A3               8000 non-null   uint8  
 18  Model_A4               8000 non-null   uint8  
 19  Model_Accord           8000 non-null   uint8  
 20  Model_C-Class          8000 non-null   uint8  
 21  Model_CR-V             8000 non-null   uint8  
 22  Model_Camry            8000 non-null   uint8  
 23  Model_Civic            8000 non-null   uint8  
 24  Model_Corolla          8000 non-null   uint8  
 25  Model_E-Class          8000 non-null   uint8  
 26  Model_Elantra          8000 non-null   uint8  
 27  Model_Equinox          8000 non-null   uint8  
 28  Model_Explorer         8000 non-null   uint8  
 29  Model_Fiesta           8000 non-null   uint8  
 30  Model_Focus            8000 non-null   uint8  
 31  Model_GLA              8000 non-null   uint8  
 32  Model_Golf             8000 non-null   uint8  
 33  Model_Impala           8000 non-null   uint8  
 34  Model_Malibu           8000 non-null   uint8  
 35  Model_Optima           8000 non-null   uint8  
```

```
 36  Model_Passat                 8000 non-null   uint8
 37  Model_Q5                     8000 non-null   uint8
 38  Model_RAV4                   8000 non-null   uint8
 39  Model_Rio                    8000 non-null   uint8
 40  Model_Sonata                 8000 non-null   uint8
 41  Model_Sportage               8000 non-null   uint8
 42  Model_Tiguan                 8000 non-null   uint8
 43  Model_Tucson                 8000 non-null   uint8
 44  Model_X5                     8000 non-null   uint8
 45  Fuel_Type_Diesel             8000 non-null   uint8
 46  Fuel_Type_Electric           8000 non-null   uint8
 47  Fuel_Type_Hybrid             8000 non-null   uint8
 48  Fuel_Type_Petrol             8000 non-null   uint8
 49  Transmission_Automatic       8000 non-null   uint8
 50  Transmission_Manual          8000 non-null   uint8
 51  Transmission_Semi-Automatic  8000 non-null   uint8
dtypes: float64(5), uint8(47)
memory usage: 742.2 KB
```

In [131]:
```
y_train.info()
```

```
<class 'pandas.core.series.Series'>
Int64Index: 8000 entries, 9254 to 7270
Series name: Price
Non-Null Count  Dtype
--------------  -----
8000 non-null   float64
dtypes: float64(1)
memory usage: 125.0 KB
```

In [125]:
```
X_test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2000 entries, 6252 to 6929
Data columns (total 52 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   Year                   2000 non-null   float64
 1   Engine_Size            2000 non-null   float64
 2   Mileage                2000 non-null   float64
 3   Doors                  2000 non-null   float64
 4   Owner_Count            2000 non-null   float64
 5   Brand_Audi             2000 non-null   uint8  
 6   Brand_BMW              2000 non-null   uint8  
 7   Brand_Chevrolet        2000 non-null   uint8  
 8   Brand_Ford             2000 non-null   uint8  
 9   Brand_Honda            2000 non-null   uint8  
 10  Brand_Hyundai          2000 non-null   uint8  
 11  Brand_Kia              2000 non-null   uint8  
 12  Brand_Mercedes         2000 non-null   uint8  
 13  Brand_Toyota           2000 non-null   uint8  
 14  Brand_Volkswagen       2000 non-null   uint8  
 15  Model_3 Series         2000 non-null   uint8  
 16  Model_5 Series         2000 non-null   uint8  
 17  Model_A3               2000 non-null   uint8  
 18  Model_A4               2000 non-null   uint8  
 19  Model_Accord           2000 non-null   uint8  
 20  Model_C-Class          2000 non-null   uint8  
 21  Model_CR-V             2000 non-null   uint8  
 22  Model_Camry            2000 non-null   uint8  
 23  Model_Civic            2000 non-null   uint8  
 24  Model_Corolla          2000 non-null   uint8  
 25  Model_E-Class          2000 non-null   uint8  
 26  Model_Elantra          2000 non-null   uint8  
 27  Model_Equinox          2000 non-null   uint8  
 28  Model_Explorer         2000 non-null   uint8  
 29  Model_Fiesta           2000 non-null   uint8  
 30  Model_Focus            2000 non-null   uint8  
 31  Model_GLA              2000 non-null   uint8  
 32  Model_Golf             2000 non-null   uint8  
 33  Model_Impala           2000 non-null   uint8  
 34  Model_Malibu           2000 non-null   uint8  
 35  Model_Optima           2000 non-null   uint8
```

```
 36  Model_Passat                 2000 non-null   uint8
 37  Model_Q5                     2000 non-null   uint8
 38  Model_RAV4                   2000 non-null   uint8
 39  Model_Rio                    2000 non-null   uint8
 40  Model_Sonata                 2000 non-null   uint8
 41  Model_Sportage               2000 non-null   uint8
 42  Model_Tiguan                 2000 non-null   uint8
 43  Model_Tucson                 2000 non-null   uint8
 44  Model_X5                     2000 non-null   uint8
 45  Fuel_Type_Diesel             2000 non-null   uint8
 46  Fuel_Type_Electric           2000 non-null   uint8
 47  Fuel_Type_Hybrid             2000 non-null   uint8
 48  Fuel_Type_Petrol             2000 non-null   uint8
 49  Transmission_Automatic       2000 non-null   uint8
 50  Transmission_Manual          2000 non-null   uint8
 51  Transmission_Semi-Automatic  2000 non-null   uint8
dtypes: float64(5), uint8(47)
memory usage: 185.5 KB
```

In [127]:
```
y_test.info()
```

```
<class 'pandas.core.series.Series'>
Int64Index: 2000 entries, 6252 to 6929
Series name: Price
Non-Null Count  Dtype
--------------  -----
2000 non-null   float64
dtypes: float64(1)
memory usage: 31.2 KB
```

In [139]:
```
Predictions = reg.predict(X_test)
```

In [147]:
```
r2 = r2_score(y_test, Predictions)
mse = mean_squared_error(y_test, Predictions)

print(f"R² Score: {r2}")
print(f"Mean Squared Error: {mse}")
```

```
R² Score: 0.7876564555519954
Mean Squared Error: 1748252.8376176425
```

In [149]:
```
Predictions
```

Out[149]:
```
array([ 2626.20022967, 11060.98936863, 12769.15888373, ...,
        8347.11623903,  6005.33867948, 11581.9397512 ])
```