Importing Libraries and Dataset

In [2]:
```
# importing the required libraries/Modules

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

In [3]:
```

car_data = pd.read_csv("car_data.csv")
```

Dataset Explore

In [4]:
```
car_data.head()
```

Out[4]:

	Make	Model	Year	Engine Fuel Type	Engine HP	Engine Cylinders	Transmission Type	Driven_Wheels	Number of Doors	M
0	BMW	1 Series M	2011	premium unleaded (required)	335.0	6.0	MANUAL	rear wheel drive	2.0	Tu
1	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Lux
2	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	
3	BMW	1 Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Lux
4	BMW	1 Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	

In [5]:
```
1 car_data.tail()
```

Out[5]:

	Make	Model	Year	Engine Fuel Type	Engine HP	Engine Cylinders	Transmission Type	Driven_Wheels	N
11909	Acura	ZDX	2012	premium unleaded (required)	300.0	6.0	AUTOMATIC	all wheel drive	
11910	Acura	ZDX	2012	premium unleaded (required)	300.0	6.0	AUTOMATIC	all wheel drive	
11911	Acura	ZDX	2012	premium unleaded (required)	300.0	6.0	AUTOMATIC	all wheel drive	
11912	Acura	ZDX	2013	premium unleaded (recommended)	300.0	6.0	AUTOMATIC	all wheel drive	
11913	Lincoln	Zephyr	2006	regular unleaded	221.0	6.0	AUTOMATIC	front wheel drive	

In [6]:
```
1 car_data.columns
```

Out[6]:
```
Index(['Make', 'Model', 'Year', 'Engine Fuel Type', 'Engine HP',
       'Engine Cylinders', 'Transmission Type', 'Driven_Wheels',
       'Number of Doors', 'Market Category', 'Vehicle Size', 'Vehicle Style',
       'highway MPG', 'city mpg', 'Popularity', 'MSRP'],
      dtype='object')
```

In [7]:
```
1 car_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11914 entries, 0 to 11913
Data columns (total 16 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   Make               11914 non-null  object 
 1   Model              11914 non-null  object 
 2   Year               11914 non-null  int64  
 3   Engine Fuel Type   11911 non-null  object 
 4   Engine HP          11845 non-null  float64
 5   Engine Cylinders   11884 non-null  float64
 6   Transmission Type  11914 non-null  object 
 7   Driven_Wheels      11914 non-null  object 
 8   Number of Doors    11908 non-null  float64
 9   Market Category    8172 non-null   object 
 10  Vehicle Size       11914 non-null  object 
 11  Vehicle Style      11914 non-null  object 
 12  highway MPG        11914 non-null  int64  
 13  city mpg           11914 non-null  int64  
 14  Popularity         11914 non-null  int64  
 15  MSRP               11914 non-null  int64  
dtypes: float64(3), int64(5), object(8)
memory usage: 1.5+ MB
```

In [8]:
```
1 car_data.isnull().sum()
```

Out[8]:
```
Make                  0
Model                 0
Year                  0
Engine Fuel Type      3
Engine HP            69
Engine Cylinders     30
Transmission Type     0
Driven_Wheels         0
Number of Doors       6
Market Category    3742
Vehicle Size          0
Vehicle Style         0
highway MPG           0
city mpg              0
Popularity            0
MSRP                  0
dtype: int64
```

In [9]:
```
# Fill null values in the "Engine HP" column with the mean value of the co
car_data['Engine HP'] = car_data['Engine HP'].fillna(car_data['Engine HP'

# Fill null values in the "Engine Cylinders" column with the median value
car_data['Engine Cylinders'] = car_data['Engine Cylinders'].fillna(car_da

# Fill null values in the "Engine Cylinders" column with the mode value oj
car_data['Engine Fuel Type'] = car_data['Engine Fuel Type'].fillna(car_da


```

In [10]:
```
car_data.describe()
```

Out[10]:

	Year	Engine HP	Engine Cylinders	Number of Doors	highway MPG	city mpg	F
count	11914.000000	11914.000000	11914.000000	11908.000000	11914.000000	11914.000000	119
mean	2010.384338	249.386070	5.629763	3.436093	26.637485	19.733255	15
std	7.579740	108.875192	1.778413	0.881315	8.863001	8.987798	14
min	1990.000000	55.000000	0.000000	2.000000	12.000000	7.000000	
25%	2007.000000	170.000000	4.000000	2.000000	22.000000	16.000000	5
50%	2015.000000	227.000000	6.000000	4.000000	26.000000	18.000000	13
75%	2016.000000	300.000000	6.000000	4.000000	30.000000	22.000000	20
max	2017.000000	1001.000000	16.000000	4.000000	354.000000	137.000000	56

In [11]:
```
1 print(car_data)
```

	Make	Model	Year	Engine Fuel Type	Engine HP
0	BMW	1 Series M	2011	premium unleaded (required)	335.0
1	BMW	1 Series	2011	premium unleaded (required)	300.0
2	BMW	1 Series	2011	premium unleaded (required)	300.0
3	BMW	1 Series	2011	premium unleaded (required)	230.0
4	BMW	1 Series	2011	premium unleaded (required)	230.0
...	...	...	...	...	...
11909	Acura	ZDX	2012	premium unleaded (required)	300.0
11910	Acura	ZDX	2012	premium unleaded (required)	300.0
11911	Acura	ZDX	2012	premium unleaded (required)	300.0
11912	Acura	ZDX	2013	premium unleaded (recommended)	300.0
11913	Lincoln	Zephyr	2006	regular unleaded	221.0

	Engine Cylinders	Transmission Type	Driven_Wheels	Number of Doors
0	6.0	MANUAL	rear wheel drive	2.0
1	6.0	MANUAL	rear wheel drive	2.0
2	6.0	MANUAL	rear wheel drive	2.0
3	6.0	MANUAL	rear wheel drive	2.0
4	6.0	MANUAL	rear wheel drive	2.0
...	...	...	...	...
11909	6.0	AUTOMATIC	all wheel drive	4.0
11910	6.0	AUTOMATIC	all wheel drive	4.0
11911	6.0	AUTOMATIC	all wheel drive	4.0
11912	6.0	AUTOMATIC	all wheel drive	4.0
11913	6.0	AUTOMATIC	front wheel drive	4.0

	Market Category	Vehicle Size	Vehicle Style
0	Factory Tuner,Luxury,High-Performance	Compact	Coupe
1	Luxury,Performance	Compact	Convertible
2	Luxury,High-Performance	Compact	Coupe
3	Luxury,Performance	Compact	Coupe
4	Luxury	Compact	Convertible
...	...	...	...
11909	Crossover,Hatchback,Luxury	Midsize	4dr Hatchback
11910	Crossover,Hatchback,Luxury	Midsize	4dr Hatchback
11911	Crossover,Hatchback,Luxury	Midsize	4dr Hatchback
11912	Crossover,Hatchback,Luxury	Midsize	4dr Hatchback
11913	Luxury	Midsize	Sedan

	highway MPG	city mpg	Popularity	MSRP
0	26	19	3916	46135
1	28	19	3916	40650
2	28	20	3916	36350
3	28	18	3916	29450
4	28	18	3916	34500
...	...	...	...	...
11909	23	16	204	46120
11910	23	16	204	56670
11911	23	16	204	50620
11912	23	16	204	50920
11913	26	17	61	28995

[11914 rows x 16 columns]

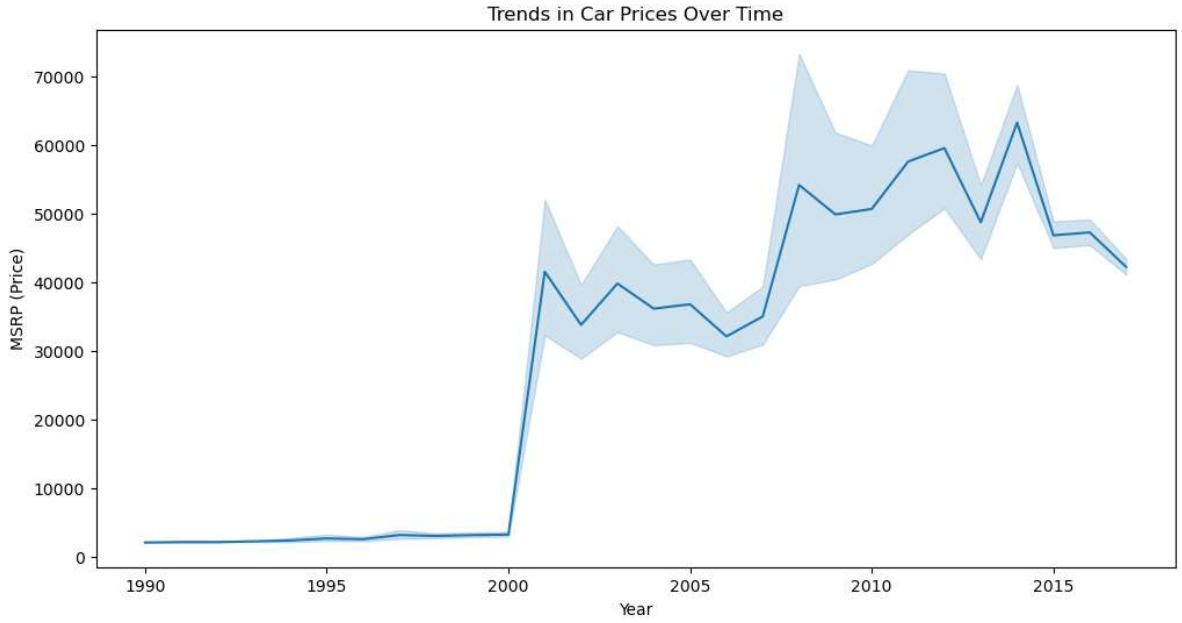
In [12]:
```
1 car_data.duplicated()
```

Out[12]:
```
0        False
1        False
2        False
3        False
4        False
         ...
11909    False
11910    False
11911    False
11912    False
11913    False
Length: 11914, dtype: bool
```

Understanding the Dataset

Visualize trends in car features and pricing over time

In [13]:
```
plt.figure(figsize=(12, 6))
sns.lineplot(x='Year', y='MSRP', data=car_data)
plt.title('Trends in Car Prices Over Time')
plt.xlabel('Year')
plt.ylabel('MSRP (Price)')
plt.show()
```

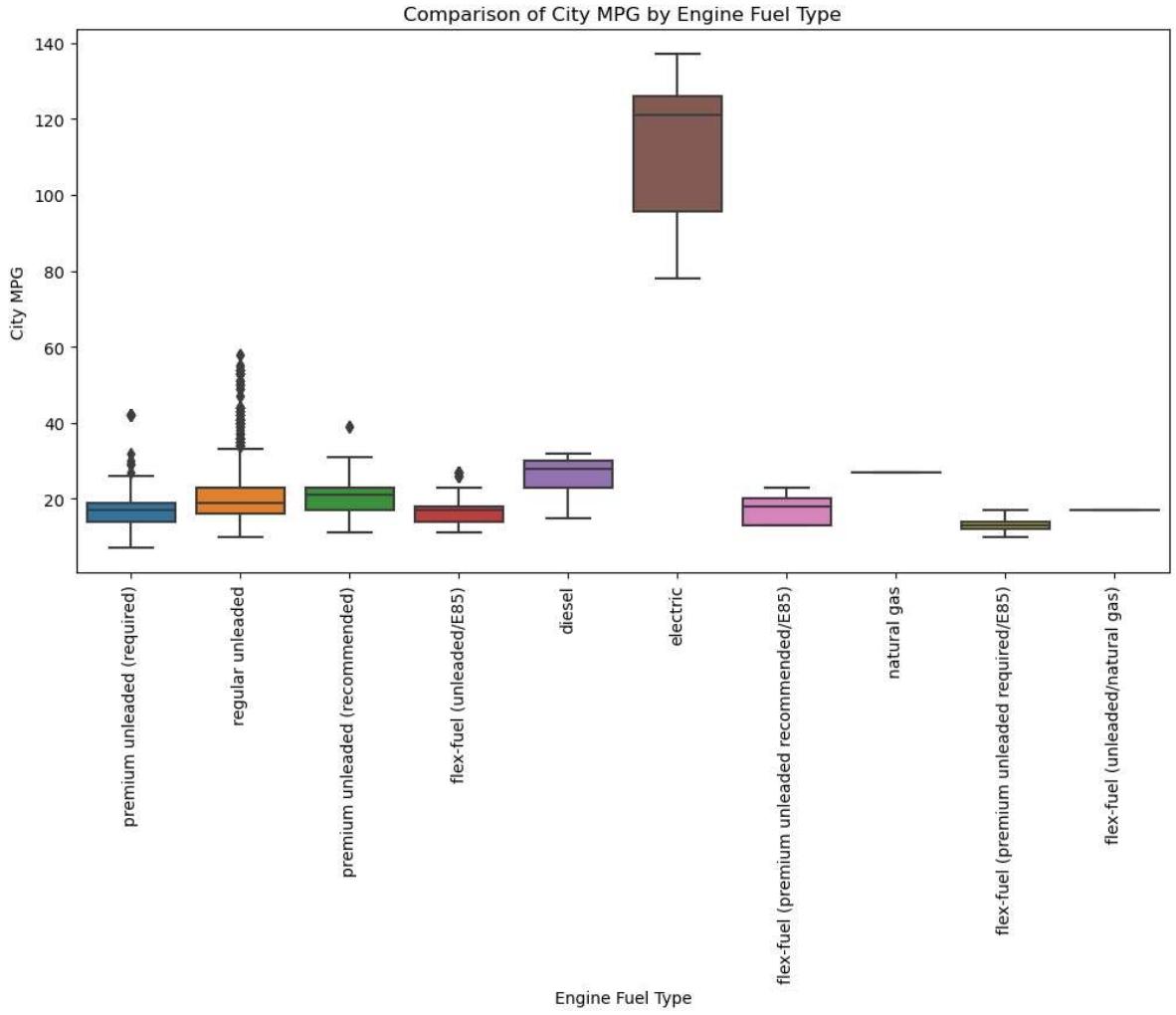


Compare fuel efficiency of different types of cars

In [14]:
```
car_data.columns
```

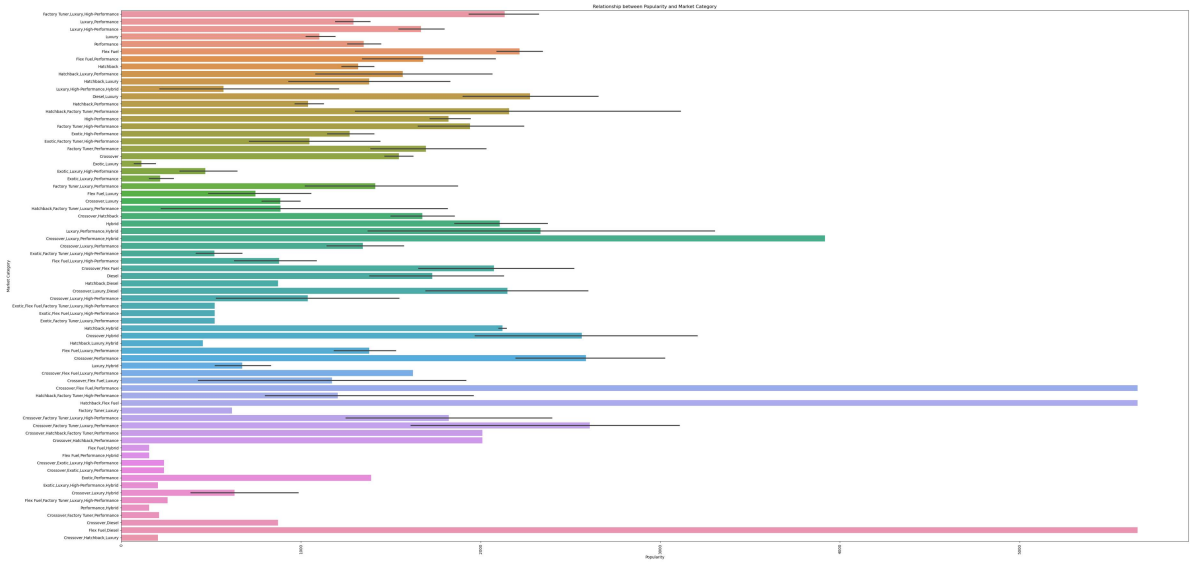
Out[14]:
```
Index(['Make', 'Model', 'Year', 'Engine Fuel Type', 'Engine HP',
       'Engine Cylinders', 'Transmission Type', 'Driven_Wheels',
       'Number of Doors', 'Market Category', 'Vehicle Size', 'Vehicle Style',
       'highway MPG', 'city mpg', 'Popularity', 'MSRP'],
      dtype='object')
```

In [15]:
```
plt.figure(figsize=(12, 6))
sns.boxplot(x='Engine Fuel Type', y='city mpg', data=car_data)
plt.title('Comparison of City MPG by Engine Fuel Type')
plt.xlabel('Engine Fuel Type')
plt.ylabel('City MPG')
plt.xticks(rotation=90)
plt.show()
```



Investigating the relationship between a car's features and its popularity

In [16]:
```
plt.figure(figsize=(50, 25))
sns.barplot(x='Popularity', y='Market Category', data=car_data)
plt.title('Relationship between Popularity and Market Category')
plt.xlabel('Popularity')
plt.ylabel('Market Category')
plt.xticks(rotation=90)
plt.show()
```

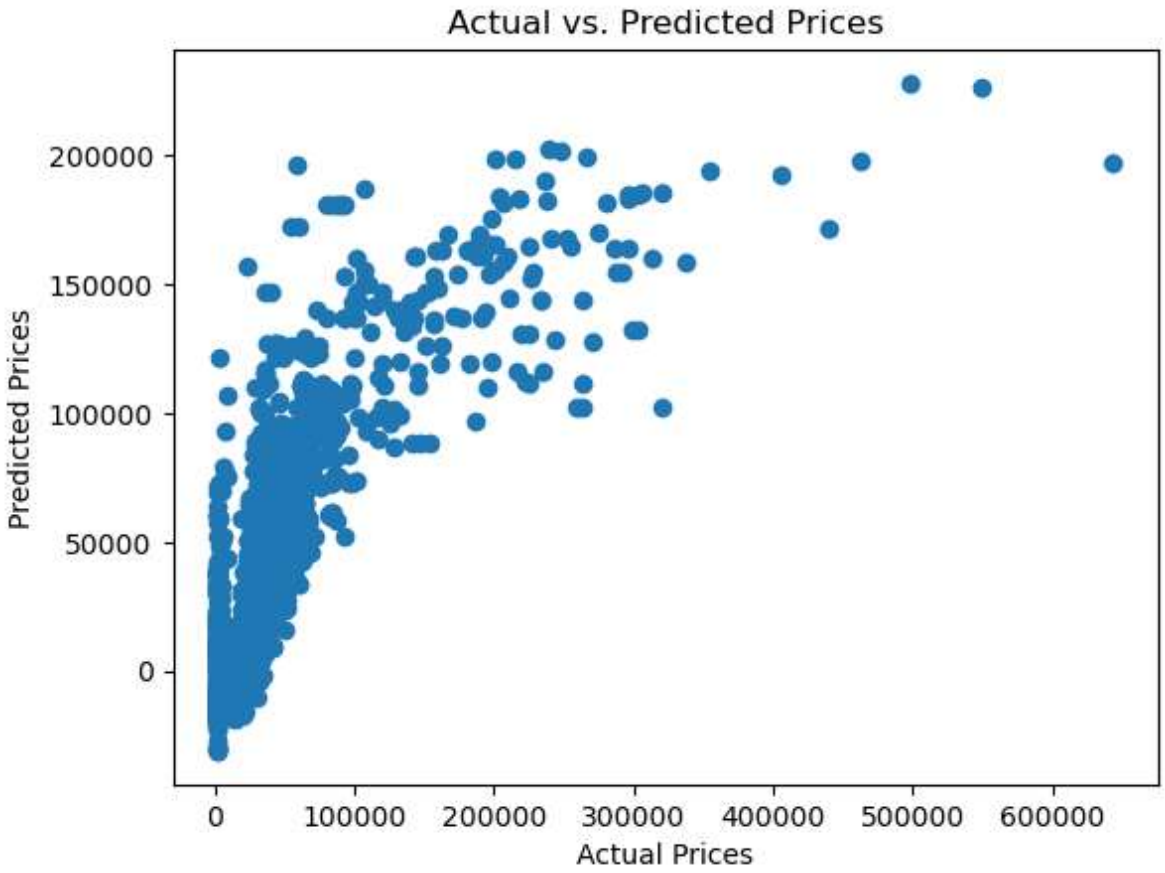


Predicting the price of a car based on its numeric features

In [17]:

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

# Assuming you've preprocessed your data and have a feature matrix X and t
X = car_data[['Engine HP', 'Engine Cylinders', 'highway MPG', 'city mpg',
y = car_data['MSRP']  # Dependent variable (car prices)
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, r

# Create a Linear Regression model
model = LinearRegression()

# Train the model
model.fit(X_train, y_train)

# Make predictions on the test data
y_pred = model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Squared Error: {mse}")
print(f"R-squared: {r2}")

# Visualize actual vs. predicted prices
plt.scatter(y_test, y_pred)
plt.xlabel("Actual Prices")
plt.ylabel("Predicted Prices")
plt.title("Actual vs. Predicted Prices")
plt.show()

```

```
Mean Squared Error: 1070196857.0707968
R-squared: 0.5510109345873742
```



Analysis of Dataset

Analysis of Dataset

How does the popularity of a car model vary across different market categories?

Task 1.A: : Create a pivot table that shows the number of car models in each market category and their corresponding popularity scores.

In [25]:
```
pivot_table = car_data.groupby('Market Category').agg({'Model': 'count',

# Rename the columns for clarity
pivot_table = pivot_table.rename(columns={'Model': 'Number of Car Models'

# Sort the pivot table by the number of car models in descending order
pivot_table = pivot_table.sort_values(by='Number of Car Models', ascendin

# Display the pivot table
pivot_table
```

Out[25]:

Market Category	Number of Car Models	Average Popularity Score
Crossover	1110	1545.263063
Flex Fuel	872	2217.302752
Luxury	855	1102.657310
Luxury,Performance	673	1292.615156
Hatchback	641	1318.865835
...	...	...
Exotic,Luxury,High-Performance,Hybrid	1	204.000000
Flex Fuel,Factory Tuner,Luxury,High-Performance	1	258.000000
Crossover,Exotic,Luxury,Performance	1	238.000000
Crossover,Exotic,Luxury,High-Performance	1	238.000000
Performance,Hybrid	1	155.000000

71 rows × 2 columns

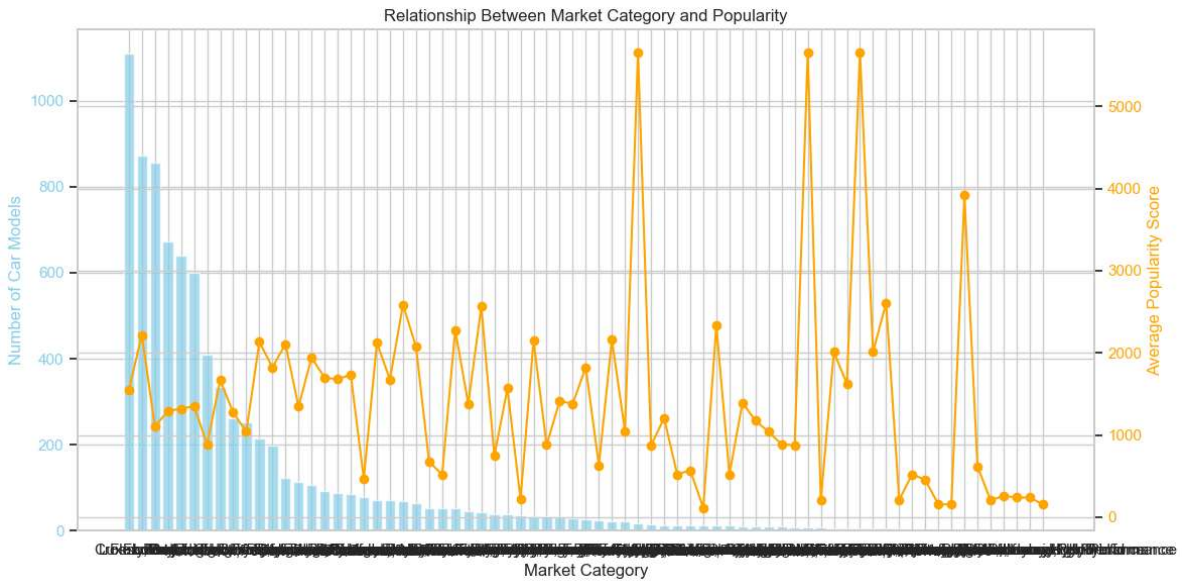
Task 1.B: Create a combo chart that visualizes the relationship between market category and popularity.

In [22]:
```
car_data.columns
```

Out[22]:
```
Index(['Make', 'Model', 'Year', 'Engine Fuel Type', 'Engine HP',
       'Engine Cylinders', 'Transmission Type', 'Driven_Wheels',
       'Number of Doors', 'Market Category', 'Vehicle Size', 'Vehicle Style',
       'highway MPG', 'city mpg', 'Popularity', 'MSRP'],
      dtype='object')
```

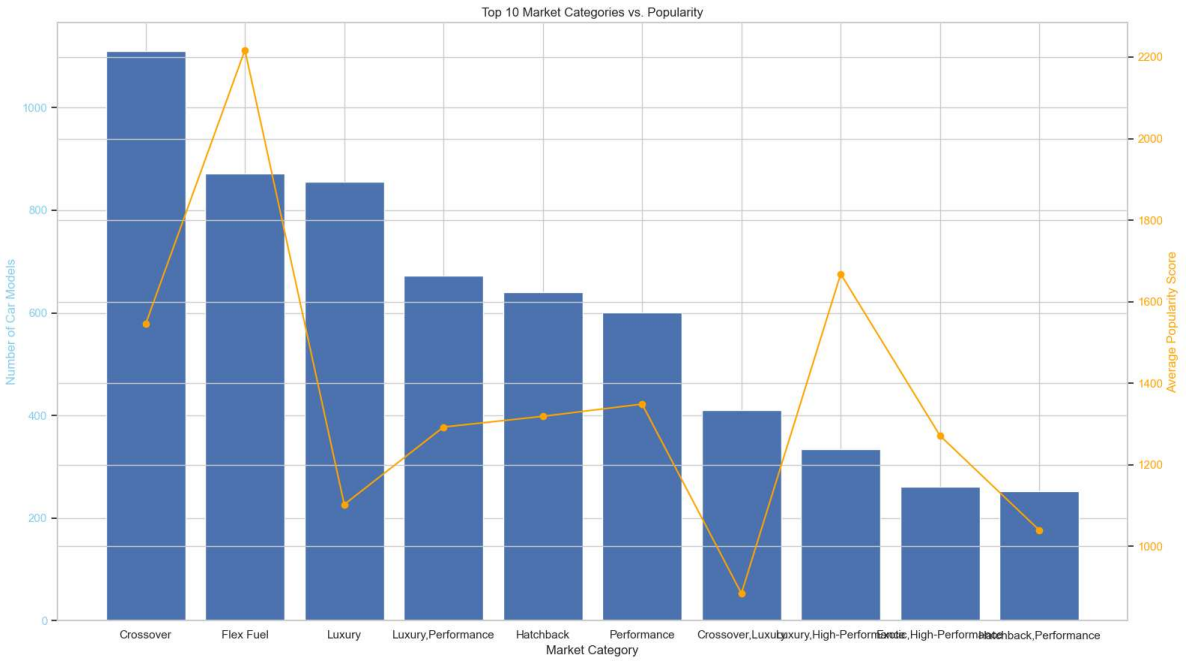
In [28]:

```
pivot_table = car_data.groupby('Market Category').agg({'Model': 'count',

# Sort the pivot table by the number of car models in descending order
pivot_table = pivot_table.sort_values(by='Model', ascending=False)

# Create a combo chart
fig, ax1 = plt.subplots(figsize=(12, 6))

# Bar chart for the number of car models
ax1.bar(pivot_table['Market Category'], pivot_table['Model'], color='skybl
ax1.set_xlabel("Market Category")
ax1.set_ylabel("Number of Car Models", color='skyblue')
ax1.tick_params(axis='y', labelcolor='skyblue')

# Create a secondary y-axis for the line chart
ax2 = ax1.twinx()

# Line chart for the average popularity score
ax2.plot(pivot_table['Market Category'], pivot_table['Popularity'], marke
ax2.set_ylabel("Average Popularity Score", color='orange')
ax2.tick_params(axis='y', labelcolor='orange')

# Rotate x-axis labels for better readability
plt.xticks(rotation=90)
plt.title("Relationship Between Market Category and Popularity")

# Show the combo chart
plt.tight_layout()
plt.show()
```



In [32]:

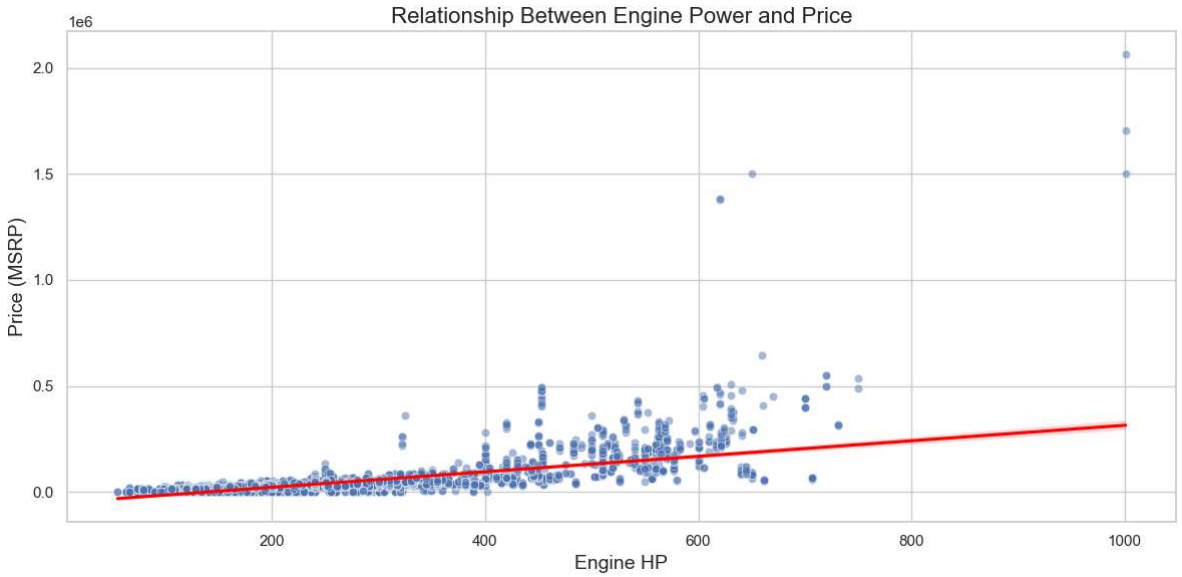
```
# Group the data by Market Category and calculate the count of car models
pivot_table = car_data.groupby('Market Category').agg({'Model': 'count',

# Sort the pivot table by the number of car models in descending order
pivot_table = pivot_table.sort_values(by='Model', ascending=False)

# Display only the top N categories
top_n = 10  # Change this value to the desired number of top categories
pivot_table = pivot_table.head(top_n)

# Create a combo chart
fig, ax1 = plt.subplots(figsize=(16, 9))

# Bar chart for the number of car models
ax1.bar(pivot_table['Market Category'], pivot_table['Model'])
ax1.set_xlabel("Market Category")
ax1.set_ylabel("Number of Car Models", color='skyblue')
ax1.tick_params(axis='y', labelcolor='skyblue')

# Create a secondary y-axis for the line chart
ax2 = ax1.twinx()

# Line chart for the average popularity score
ax2.plot(pivot_table['Market Category'], pivot_table['Popularity'], marke
ax2.set_ylabel("Average Popularity Score", color='orange')
ax2.tick_params(axis='y', labelcolor='orange')

# Giving title to the chart
plt.title(f"Top {top_n} Market Categories vs. Popularity")

# Show the combo chart
plt.tight_layout()
plt.show()
```



What is the relationship between a car's engine power and its price?

Task 2: Create a scatter chart that plots engine power on the x-axis and price on the y-axis. Add a trendline to the chart to visualize the relationship between these variables.

In [48]:
```
# Create a scatter plot
plt.figure(figsize=(12, 6))
sns.scatterplot(x='Engine HP', y='MSRP', data=car_data, alpha=0.5)

# Add a trendline
sns.regplot(x='Engine HP', y='MSRP', data=car_data, scatter=False, color=

# Customize labels and title
plt.xlabel("Engine HP", fontsize=14)
plt.ylabel("Price (MSRP)", fontsize=14)
plt.title("Relationship Between Engine Power and Price", fontsize=16)

# Show the plot
plt.grid(True)
plt.tight_layout()
plt.show()
```

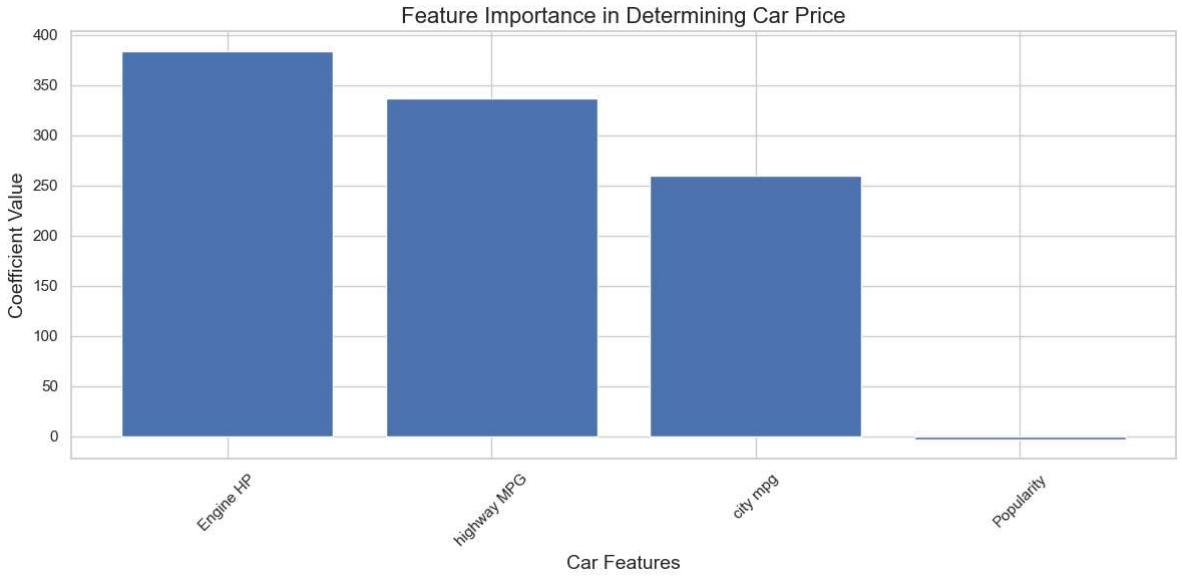


Which car features are most important in determining a car's price?

Task 3: Using regression analysis to identify the variables that have the strongest relationship with a car's price.

In [49]:
```
# Select relevant features and target variable
X = car_data[['Engine HP', 'highway MPG', 'city mpg', 'Popularity']]
y = car_data['MSRP']

# Create and fit a linear regression model
model = LinearRegression()
model.fit(X, y)

# Get the coefficient values for each feature
coefficients = model.coef_
print(coefficients)

# Create a bar chart to visualize feature importance
plt.figure(figsize=(12, 6))
plt.bar(X.columns, coefficients)
plt.xlabel("Car Features", fontsize=14)
plt.ylabel("Coefficient Value", fontsize=14)
plt.title("Feature Importance in Determining Car Price", fontsize=16)
plt.xticks(rotation=45)  # Rotate x-axis labels for readability
plt.grid(True)

# Show the bar chart
plt.tight_layout()
plt.show()
```

```
[383.9162872  336.83794293 259.89510263  -3.05399185]
```



How does the average price of a car vary across different manufacturers?

Task 4.A: Create a pivot table that shows the average price of cars for each manufacturer

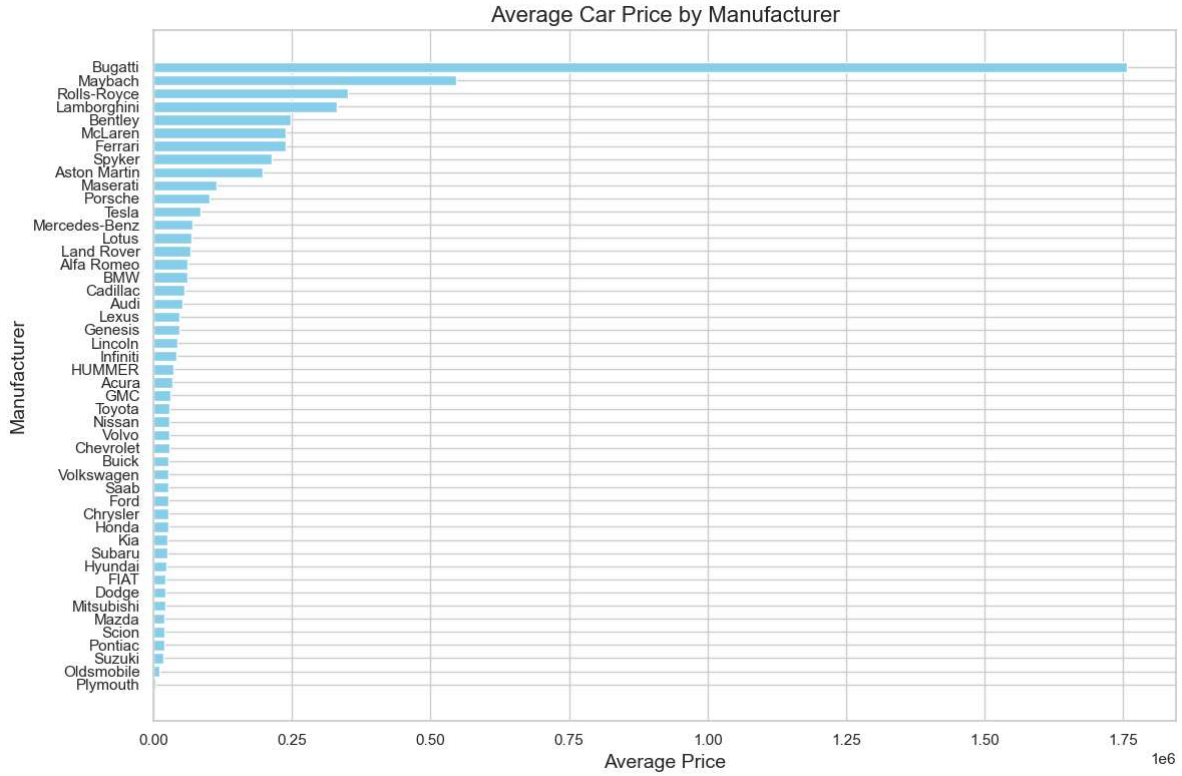
In [51]:

```
# Create a pivot table to show the average price for each manufacturer
pivot_table = car_data.pivot_table(values='MSRP', index='Make', aggfunc='
# Sort the pivot table by average price in descending order
pivot_table = pivot_table.sort_values(by='MSRP')
# Display the pivot table
print(pivot_table)
```

Make	MSRP
Plymouth	3.122902e+03
Oldsmobile	1.154254e+04
Suzuki	1.790721e+04
Pontiac	1.932155e+04
Scion	1.993250e+04
Mazda	2.003938e+04
Mitsubishi	2.124054e+04
Dodge	2.239006e+04
FIAT	2.267024e+04
Hyundai	2.459704e+04
Subaru	2.482750e+04
Kia	2.531017e+04
Honda	2.667434e+04
Chrysler	2.672296e+04
Ford	2.739927e+04
Saab	2.741350e+04
Volkswagen	2.810238e+04
Buick	2.820661e+04
Chevrolet	2.835039e+04
Volvo	2.854116e+04
Nissan	2.858343e+04
Toyota	2.903002e+04
GMC	3.049330e+04
Acura	3.488759e+04
HUMMER	3.646441e+04
Infiniti	4.239421e+04
Lincoln	4.283983e+04
Genesis	4.661667e+04
Lexus	4.754907e+04
Audi	5.345211e+04
Cadillac	5.623132e+04
BMW	6.154676e+04
Alfa Romeo	6.160000e+04
Land Rover	6.782322e+04
Lotus	6.918828e+04
Mercedes-Benz	7.147623e+04
Tesla	8.525556e+04
Porsche	1.016224e+05
Maserati	1.142077e+05
Aston Martin	1.979104e+05
Spyker	2.133233e+05
Ferrari	2.382188e+05
McLaren	2.398050e+05
Bentley	2.471693e+05
Lamborghini	3.315673e+05
Rolls-Royce	3.511306e+05
Maybach	5.462219e+05
Bugatti	1.757224e+06

Task 4.B: Create a bar chart or a horizontal stacked bar chart that visualizes the relationship between manufacturer and average price.

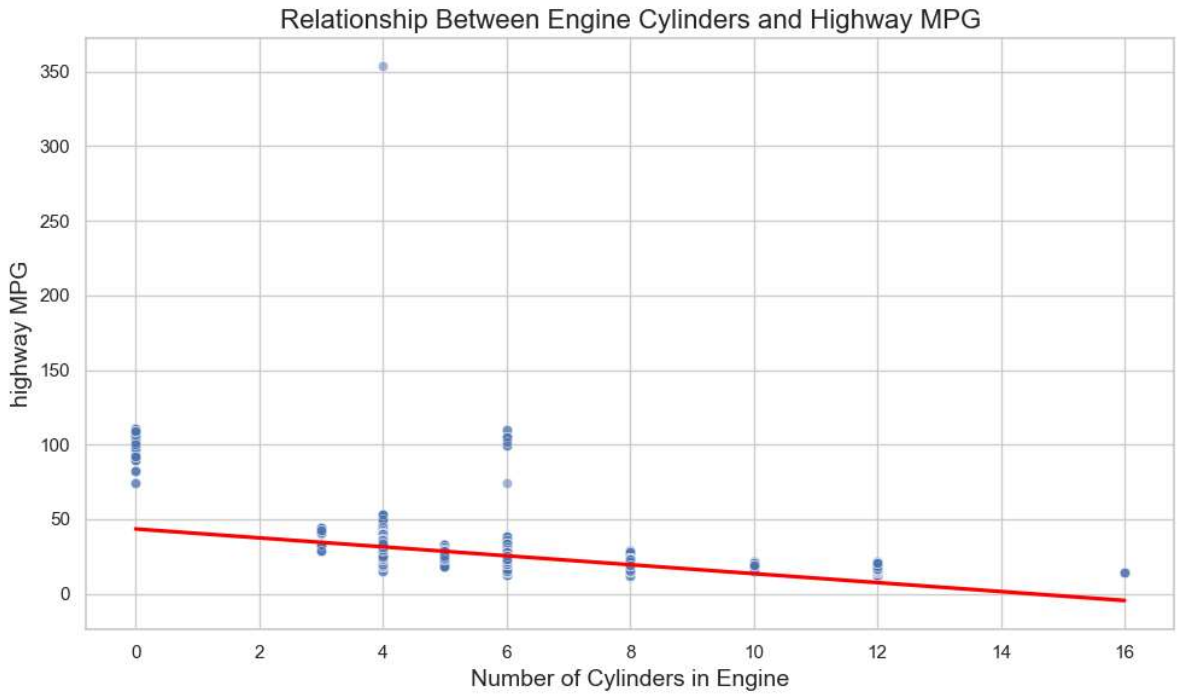
In [54]:
```
# Create a horizontal bar chart
plt.figure(figsize=(12, 8))
plt.barh(pivot_table.index, pivot_table['MSRP'], color='skyblue')
plt.xlabel("Average Price", fontsize=14)
plt.ylabel("Manufacturer", fontsize=14)
plt.title("Average Car Price by Manufacturer", fontsize=16)


# Show the bar chart
plt.tight_layout()
plt.show()
```



What is the relationship between fuel efficiency and the number of cylinders in a car's engine?

Task 5.A: Create a scatter plot & trend line with the number of cylinders on the x-axis and highway MPG on the y-axis.

In [64]:
```
# Create a scatter plot with trendline
plt.figure(figsize=(10, 6))
sns.scatterplot(x='Engine Cylinders', y='highway MPG', data=car_data, alpl

# Add a trendline
sns.regplot(x='Engine Cylinders', y='highway MPG', data=car_data, scatter=

# Customize labels and title
plt.xlabel("Number of Cylinders in Engine", fontsize=14)
plt.ylabel("highway MPG", fontsize=14)
plt.title("Relationship Between Engine Cylinders and Highway MPG", fontsi:

# Show the plot
plt.tight_layout()
plt.show()
```



Task 5.B: Calculate the correlation coefficient between the number of cylinders and highway MPG

In [69]:
```
# Calculate the correlation coefficient
correlation = car_data['Engine Cylinders'].corr(car_data['highway MPG'])

# Print the correlation coefficient
print(f"Correlation coefficient between Engine Cylinders and Highway MPG:
```

```
Correlation coefficient between Engine Cylinders and Highway MPG: -0.60
```

Inference :- A negative value indicates a negative correlation. i.e. as number of cylinders increases, highway MPG decreases