Car Sales Exploratory Data Analysis

Import Relevent Library To Perform EDA

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

import warnings
warnings.filterwarnings('ignore')
```

Import Car Sale Csv Dataset

```
df=pd.read_csv('Car Sale.csv')
```

Data exploration

Explore the loaded dataset to understand its characteristics.

- Need to explore the dataset by examining data types, missing values, descriptive statistics, unique values of categorical features, correlation matrix, distribution of the target variable, and the shape of the data.

```
df.head()
```

	Car_id	Date	Customer Name	Gender	Annual Income	Dealer_Name	Company	Model	Engine	Transmission	Color	Price ($)	Deale
0	C_CND_000001	1/2/2022	Geraldine	Male	13500	Buddy Storbeck's Diesel Service Inc	Ford	Expedition	DoubleÂ Overhead Camshaft	Auto	Black	26000	0
1	C_CND_000002	1/2/2022	Gia	Male	1480000	C & M Motors Inc	Dodge	Durango	DoubleÂ Overhead Camshaft	Auto	Black	19000	6
2	C_CND_000003	1/2/2022	Gianna	Male	1035000	Capitol KIA	Cadillac	Eldorado	Overhead Camshaft	Manual	Red	31500	3
3	C_CND_000004	1/2/2022	Giselle	Male	13500	Chrysler of Tri-Cities	Toyota	Celica	Overhead Camshaft	Manual	Pale White	14000	9
4	C_CND_000005	1/2/2022	Grace	Male	1465000	Chrysler Plymouth	Acura	TL	DoubleÂ Overhead Camshaft	Auto	Red	24500	5

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
df.columns
```

```
Index(['Car_id', 'Date', 'Customer Name', 'Gender', 'Annual Income',
       'Dealer_Name', 'Company', 'Model', 'Engine', 'Transmission', 'Color',
       'Price ($)', 'Dealer_No ', 'Body Style', 'Phone', 'Dealer_Region'],
      dtype='object')
```

```
df.duplicated().sum()
```

```
np.int64(0)
```

```
df.isnull().sum()
```

	0
Car_id	0
Date	0
Customer Name	1
Gender	0
Annual Income	0
Dealer_Name	0
Company	0
Model	0
Engine	0
Transmission	0
Color	0
Price ($)	0
Dealer_No	0
Body Style	0
Phone	0
Dealer_Region	0

dtype: int64

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23906 entries, 0 to 23905
Data columns (total 16 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Car_id         23906 non-null  object
 1   Date           23906 non-null  object
 2   Customer Name  23905 non-null  object
 3   Gender         23906 non-null  object
 4   Annual Income  23906 non-null  int64
 5   Dealer_Name    23906 non-null  object
 6   Company        23906 non-null  object
 7   Model          23906 non-null  object
 8   Engine         23906 non-null  object
 9   Transmission   23906 non-null  object
 10  Color          23906 non-null  object
 11  Price ($)      23906 non-null  int64
 12  Dealer_No      23906 non-null  object
 13  Body Style     23906 non-null  object
 14  Phone          23906 non-null  int64
 15  Dealer_Region  23906 non-null  object
dtypes: int64(3), object(13)
memory usage: 2.9+ MB
```

```
df.describe()
```

	Annual Income	Price ($)	Phone
count	2.390600e+04	23906.000000	2.390600e+04
mean	8.308403e+05	28090.247846	7.497741e+06
std	7.200064e+05	14788.687608	8.674920e+05
min	1.008000e+04	1200.000000	6.000101e+06
25%	3.860000e+05	18001.000000	6.746495e+06
50%	7.350000e+05	23000.000000	7.496198e+06
75%	1.175750e+06	34000.000000	8.248146e+06
max	1.120000e+07	85800.000000	8.999579e+06

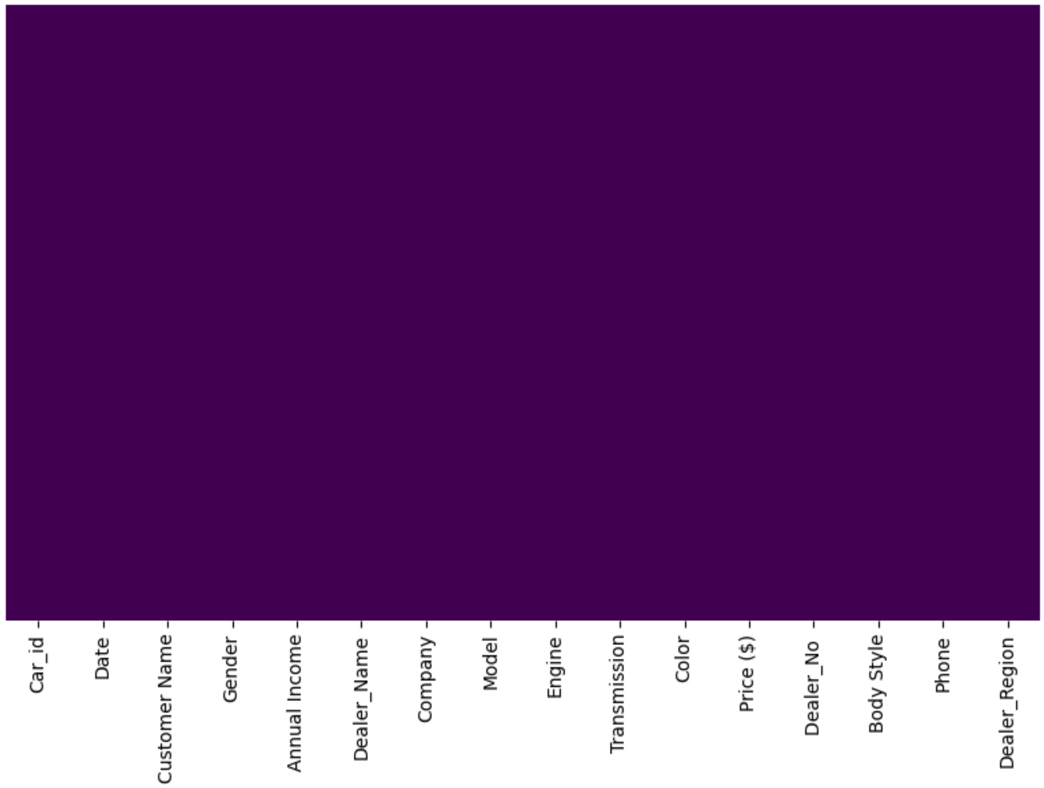
```
df.dropna(inplace=True)
```

```
# Check for missing values
missing_values = df.isnull().sum()
missing_percentage = (missing_values / len(df)) * 100
print("\nMissing Values:\n", missing_values)
print("\nMissing Value Percentage:\n", missing_percentage)
```

```
Missing Values:
 Car_id           0
Date            0
Customer Name   1
Gender          0
Annual Income   0
Dealer_Name     0
Company         0
Model           0
Engine          0
Transmission    0
Color           0
Price ($)       0
Dealer_No       0
Body Style      0
Phone           0
Dealer_Region   0
dtype: int64

Missing Value Percentage:
 Car_id           0.000000
Date            0.000000
Customer Name   0.004183
Gender          0.000000
Annual Income   0.000000
Dealer_Name     0.000000
Company         0.000000
Model           0.000000
Engine          0.000000
Transmission    0.000000
Color           0.000000
Price ($)       0.000000
Dealer_No       0.000000
Body Style      0.000000
Phone           0.000000
Dealer_Region   0.000000
dtype: float64
```

```
plt.figure(figsize=(10,6))
sns.heatmap(df.isnull(),cbar=False,cmap='viridis',yticklabels=False)
```

```
<Axes: >
```



Data cleaning

Clean the data by handling missing values and outliers.

Handle missing values, outliers, convert data types, and remove irrelevant columns as per the instructions.

```
# Convert 'Date' to datetime objects
df['Date'] = pd.to_datetime(df['Date'])

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 23905 entries, 0 to 23905
Data columns (total 16 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Car_id         23905 non-null  object
 1   Date           23905 non-null  datetime64[ns]
 2   Customer Name  23905 non-null  object
 3   Gender         23905 non-null  object
 4   Annual Income  23905 non-null  int64
 5   Dealer_Name    23905 non-null  object
 6   Company        23905 non-null  object
 7   Model          23905 non-null  object
 8   Engine         23905 non-null  object
 9   Transmission   23905 non-null  object
 10  Color          23905 non-null  object
 11  Price ($)      23905 non-null  int64
 12  Dealer_No      23905 non-null  object
 13  Body Style     23905 non-null  object
 14  Phone          23905 non-null  int64
 15  Dealer_Region  23905 non-null  object
dtypes: datetime64[ns](1), int64(3), object(12)
memory usage: 3.1+ MB
```

EDA Analysis With Multiple Step:

1.What is the avarage selling price of cars for each dealer,and how does it compare accross different dealers?

```
df['Dealer_Name'].unique()
```

```
array(["Buddy Storbeck's Diesel Service Inc", 'C & M Motors Inc',
       'Capitol KIA', 'Chrysler of Tri-Cities', 'Chrysler Plymouth',
       'Classic Chevy', 'Clay Johnson Auto Sales', 'U-Haul CO',
       'Rabun Used Car Sales', 'Race Car Help', 'Saab-Belle Dodge',
       'Scrivener Performance Engineering', 'Diehl Motor CO Inc',
       'Star Enterprises Inc', 'Suburban Ford', 'Tri-State Mack Inc',
       'Progressive Shippers Cooperative Association No',
       'Ryder Truck Rental and Leasing', 'Enterprise Rent A Car',
       'Gartner Buick Hyundai Saab', 'Hatfield Volkswagen',
       'Iceberg Rentals', 'McKinney Dodge Chrysler Jeep',
       'Motor Vehicle Branch Office', 'Nebo Chevrolet',
       'New Castle Ford Lincoln Mercury', 'Pars Auto Sales',
       'Pitre Buick-Pontiac-Gmc of Scottsdale'], dtype=object)
```

```
df['Dealer_Name'].value_counts()
```

Dealer_Name	count
Progressive Shippers Cooperative Association No	1318
Rabun Used Car Sales	1313
Race Car Help	1253
Saab-Belle Dodge	1250
Star Enterprises Inc	1249
Tri-State Mack Inc	1249
Ryder Truck Rental and Leasing	1248
U-Haul CO	1247
Scrivener Performance Engineering	1246
Suburban Ford	1243
Nebo Chevrolet	633
Pars Auto Sales	630
New Castle Ford Lincoln Mercury	629
McKinney Dodge Chrysler Jeep	629
Hatfield Volkswagen	629
Gartner Buick Hyundai Saab	628
Pitre Buick-Pontiac-Gmc of Scottsdale	628
Capitol KIA	628
Clay Johnson Auto Sales	627
Iceberg Rentals	627
Buddy Storbeck's Diesel Service Inc	627
Motor Vehicle Branch Office	626
Chrysler of Tri-Cities	626
C & M Motors Inc	625
Enterprise Rent A Car	625
Chrysler Plymouth	625
Diehl Motor CO Inc	624
Classic Chevy	623

dtype: int64

```
df['Price ($)'].value_counts()
```

Price ($)	count
22000	1191
19000	974
21000	873
26000	689
18000	627
...	...
25301	1
39340	1
16300	1
53600	1
13800	1

870 rows × 1 columns

dtype: int64

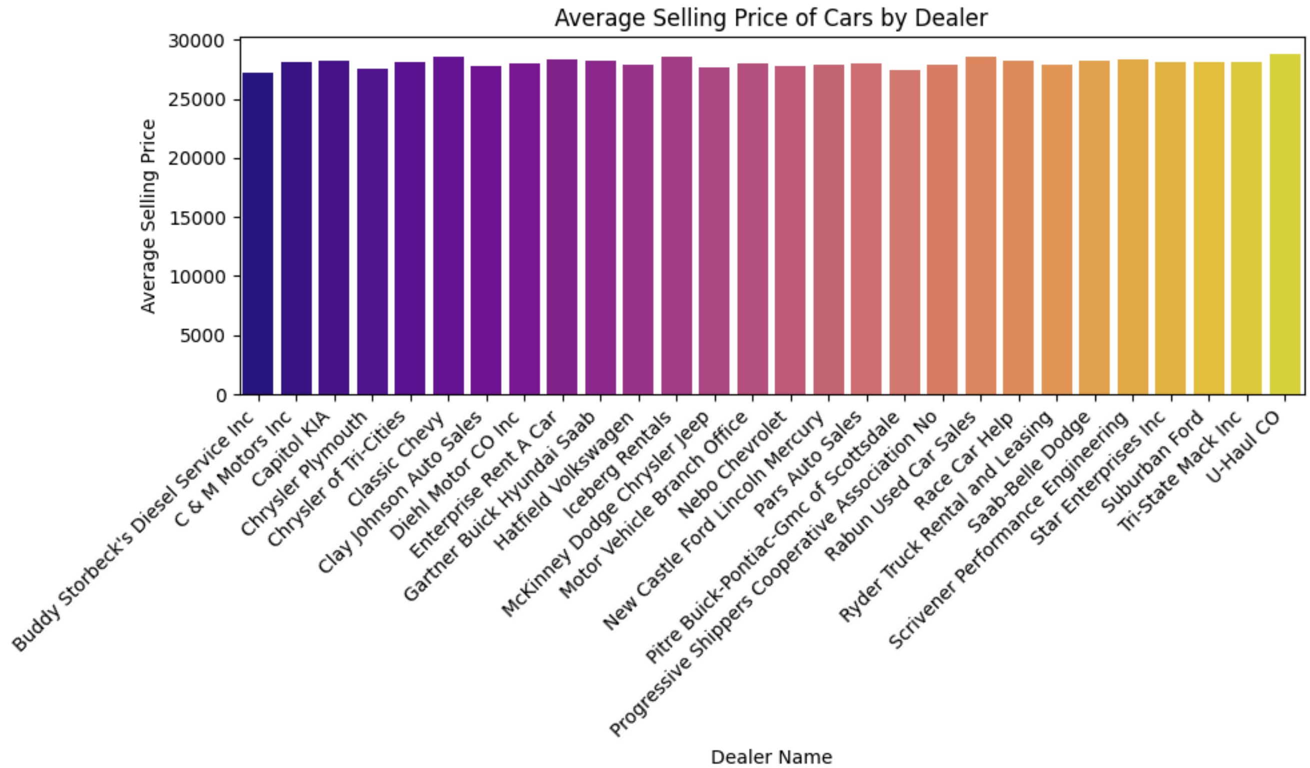
Average selling Price By Dealer

```
avg_selling_price_by_dealer = df.groupby('Dealer_Name')['Price ($)'].mean()
print(avg_selling_price_by_dealer)
```

```
Dealer_Name
Buddy Storbeck's Diesel Service Inc                27217.261563
C & M Motors Inc                                   28111.755200
Capitol KIA                                        28189.703822
Chrysler Plymouth                                  27555.526400
Chrysler of Tri-Cities                             28123.091054
Classic Chevy                                      28602.014446
Clay Johnson Auto Sales                            27816.027113
Diehl Motor CO Inc                                 27993.929487
Enterprise Rent A Car                              28312.580800
Gartner Buick Hyundai Saab                         28247.621019
Hatfield Volkswagen                                27853.712242
Iceberg Rentals                                    28522.958533
McKinney Dodge Chrysler Jeep                       27684.096979
Motor Vehicle Branch Office                        27956.739617
Nebo Chevrolet                                     27818.889415
New Castle Ford Lincoln Mercury                    27867.131955
Pars Auto Sales                                    28013.060317
Pitre Buick-Pontiac-Gmc of Scottsdale              27404.248408
Progressive Shippers Cooperative Association No    27884.264036
Rabun Used Car Sales                               28527.536177
Race Car Help                                      28163.372706
Ryder Truck Rental and Leasing                     27914.988782
Saab-Belle Dodge                                   28176.692000
Scrivener Performance Engineering                  28297.371589
Star Enterprises Inc                               28113.055244
Suburban Ford                                      28112.206758
Tri-State Mack Inc                                 28095.562050
U-Haul CO                                          28769.919006
Name: Price ($), dtype: float64
```

Price Distribution of Cars By Dealer

```
plt.figure(figsize=(10, 6))
sns.barplot(x=avg_selling_price_by_dealer.index, y=avg_selling_price_by_dealer.values, palette='plasma')
plt.title('Average Selling Price of Cars by Dealer')
plt.xlabel('Dealer Name')
plt.ylabel('Average Selling Price')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



2.Which car brand(company) has the highest variation in prices,and what does this tell us about pricing trends?

```
df.head()
```

	Car_id	Date	Customer Name	Gender	Annual Income	Dealer_Name	Company	Model	Engine	Transmission	Color	Price ($)	Deale
0	C_CND_000001	1/2/2022	Geraldine	Male	13500	Buddy Storbeck's Diesel Service Inc	Ford	Expedition	DoubleÂ Overhead Camshaft	Auto	Black	26000	0
1	C_CND_000002	1/2/2022	Gia	Male	1480000	C & M Motors Inc	Dodge	Durango	DoubleÂ Overhead Camshaft	Auto	Black	19000	6
2	C_CND_000003	1/2/2022	Gianna	Male	1035000	Capitol KIA	Cadillac	Eldorado	Overhead Camshaft	Manual	Red	31500	3
3	C_CND_000004	1/2/2022	Giselle	Male	13500	Chrysler of Tri-Cities	Toyota	Celica	Overhead Camshaft	Manual	Pale White	14000	9
4	C_CND_000005	1/2/2022	Grace	Male	1465000	Chrysler Plymouth	Acura	TL	DoubleÂ Overhead Camshaft	Auto	Red	24500	5

```
#Calculate Price Variation for Each Brand:
price_variation_by_brand = df.groupby('Model')['Price ($)'].std()
print(price_variation_by_brand)
```

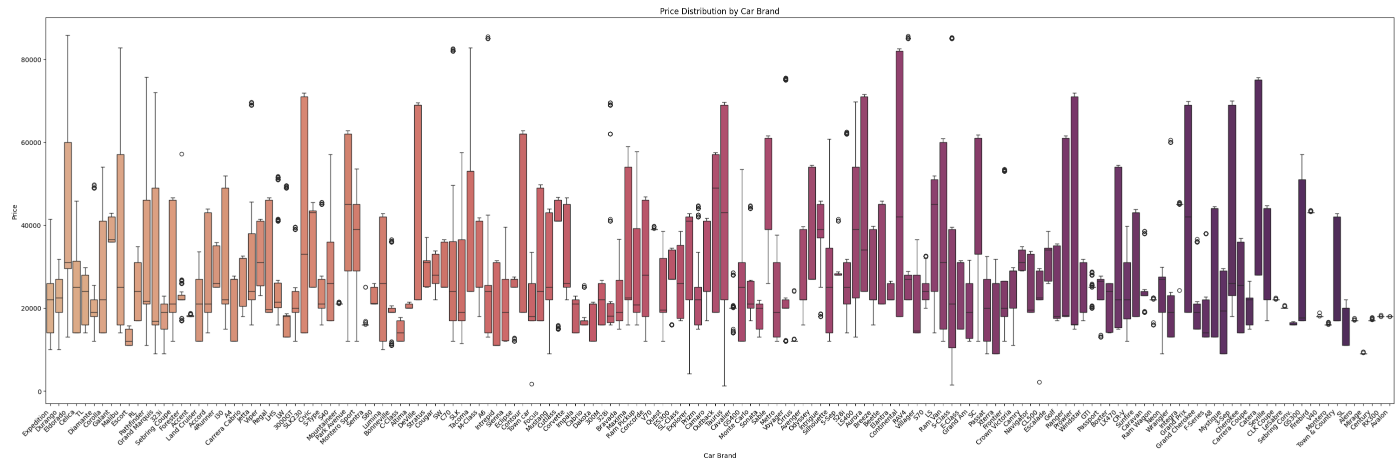
```
Model
3-Sep       21403.933176
3000GT       6238.010306
300M         4229.419059
323i         4750.820214
328i        18800.231598
                ...
Viper        6897.549327
Voyager     23064.019494
Windstar     6307.737514
Wrangler    10409.485641
Xterra       9640.303739
Name: Price ($), Length: 154, dtype: float64
```

```
#Find the Brand with the Highest Variation:
brand_with_highest_variation = price_variation_by_brand.idxmax()
```

```
print(brand_with_highest_variation)
```

```
Continental
```

```
plt.figure(figsize=(30, 10))
sns.boxplot(x='Model', y='Price ($)', data=df,palette='flare')
plt.title('Price Distribution by Car Brand')
plt.xlabel('Car Brand')
plt.ylabel('Price')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



Insights about Pricing Trends:

High price variation for a specific car brand might suggest a few things:

Wide range of models and trims:

The brand might offer a variety of models with different features and price points, leading to larger price differences.

Variable demand:

Popular or in-demand models from that brand could have higher prices, while less popular models might be priced lower.

Negotiation:

There might be more room for negotiation on the prices of cars from that brand, leading to variability in final selling prices.

External factors:

Supply chain issues, economic conditions, or regional differences in pricing strategies could also contribute to price variation.

3.What is the distribution of car prices for each transmission type,and how do the interquartile ranges compare?

```
df.head()
```

	Car_id	Date	Customer Name	Gender	Annual Income	Dealer_Name	Company	Model	Engine	Transmission	Color	Price ($)	Deale
0	C_CND_000001	1/2/2022	Geraldine	Male	13500	Buddy Storbeck's Diesel Service Inc	Ford	Expedition	DoubleÂ Overhead Camshaft	Auto	Black	26000	0
1	C_CND_000002	1/2/2022	Gia	Male	1480000	C & M Motors Inc	Dodge	Durango	DoubleÂ Overhead Camshaft	Auto	Black	19000	6
2	C_CND_000003	1/2/2022	Gianna	Male	1035000	Capitol KIA	Cadillac	Eldorado	Overhead Camshaft	Manual	Red	31500	3
3	C_CND_000004	1/2/2022	Giselle	Male	13500	Chrysler of Tri-Cities	Toyota	Celica	Overhead Camshaft	Manual	Pale White	14000	9
4	C_CND_000005	1/2/2022	Grace	Male	1465000	Chrysler Plymouth	Acura	TL	DoubleÂ Overhead Camshaft	Auto	Red	24500	5

```
df['Transmission'].unique()
```

```
array(['Auto', 'Manual'], dtype=object)
```

```
df['Transmission'].value_counts()
```

Transmission	count
Auto	12570
Manual	11335

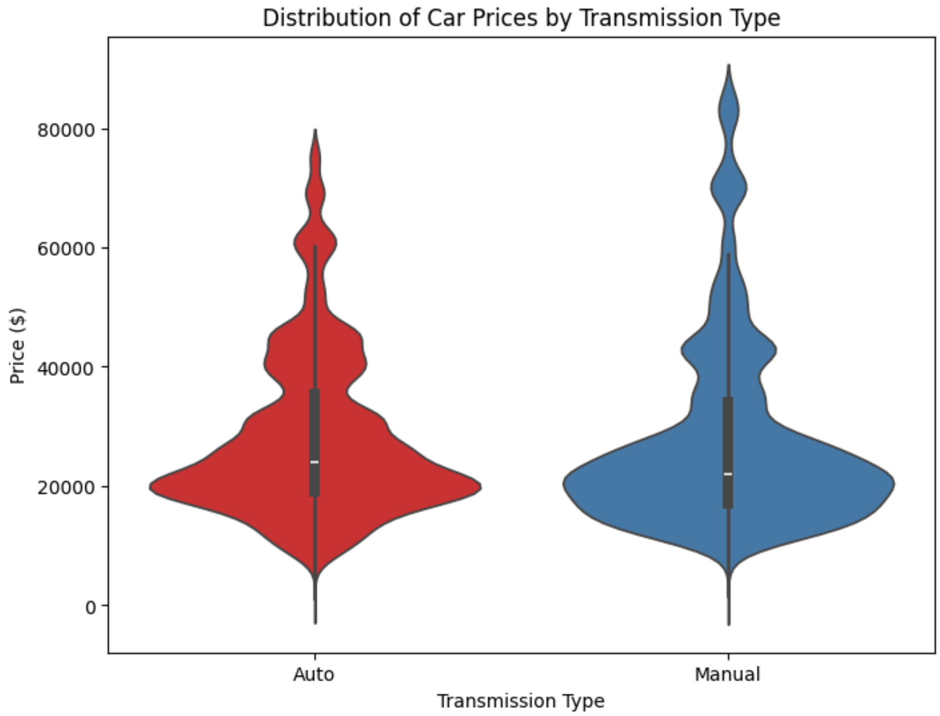
dtype: int64

```
#Group data by transmission type and calculate price statistics:
distribution = df.groupby('Transmission')['Price ($)'].describe()
print(distribution)
```

```
                count          mean           std     min      25%      50%  \
Transmission
Auto          12570.0  28247.193317  13746.805377  1200.0  19000.0  24000.0
Manual        11335.0  27914.710631  15862.871978  1700.0  17000.0  22001.0

                  75%      max
Transmission
Auto          35500.0  75700.0
Manual        34000.0  85800.0
```

Distribution Of Car prices By Transmission Type

```
#Visualize the distribution using violin plots:
plt.figure(figsize=(8, 6))
sns.violinplot(x='Transmission', y='Price ($)', data=df,palette='Set1')
plt.title('Distribution of Car Prices by Transmission Type')
plt.xlabel('Transmission Type')
plt.ylabel('Price ($)')
plt.show()
```




```
#Compare the interquartile ranges (IQRs):
automatic_iqr = distribution.loc['Auto', '75%'] - distribution.loc['Auto', '25%']
manual_iqr = distribution.loc['Manual', '75%'] - distribution.loc['Manual', '25%']
print(f"Automatic IQR: {automatic_iqr}")
print(f"Manual IQR: {manual_iqr}")
```

```
Automatic IQR: 16500.0
Manual IQR: 17000.0
```

4.What is the distribution of car prices accross the regions?

```
df.columns
```

```
Index(['Car_id', 'Date', 'Customer Name', 'Gender', 'Annual Income',
       'Dealer_Name', 'Company', 'Model', 'Engine', 'Transmission', 'Color',
       'Price ($)', 'Dealer_No ', 'Body Style', 'Phone', 'Dealer_Region'],
      dtype='object')
```

```
df["Dealer_Region"].unique()
```

```
array(['Middletown', 'Aurora', 'Greenville', 'Pasco', 'Janesville',
       'Scottsdale', 'Austin'], dtype=object)
```

```
df['Price ($)'].unique()
```

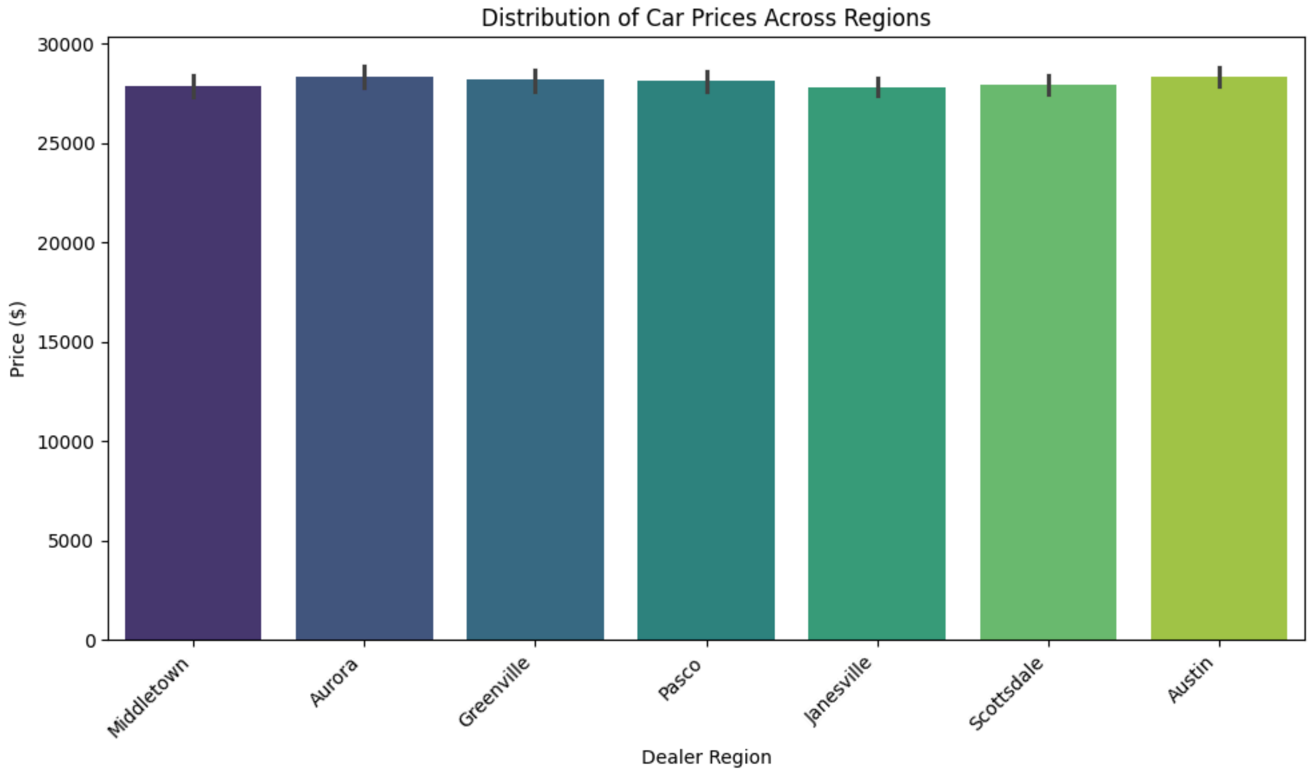
```
array([26000, 19000, 31500, 14000, 24500, 12000, 42000, 82000, 15000,
       31000, 46000,  9000, 17000, 18000, 33000, 21000, 25000, 22000,
       31250, 41000, 13000, 20000, 43000, 16000, 61000, 39000, 42500,
       45001, 36001, 21001, 29000, 27000, 25600, 36000, 31100, 22600,
       45000, 62000, 22700, 49000, 28000, 22001, 25001, 12800, 22500,
       46500, 54000, 16001, 38000, 21200, 71000, 57001, 62001, 69001,
       20001, 26750, 24000, 28501, 15500, 12500, 27250, 11000, 26500,
       69000, 14150, 60500, 44000, 11650, 11800, 27500, 16900, 14200,
       51000, 32000, 39500, 13500,  9250, 29500, 18501, 17001, 23500,
       53000, 60750, 24001, 35000, 18500, 21500, 41100, 20200, 59000,
       57000, 25500, 19100, 10000, 23000, 11501, 24250, 37000, 54500,
       25100, 34000, 21900, 29200, 85000, 43500, 14500, 16500, 85600,
       75000, 71500, 61500, 19500, 29001, 60000, 28001, 60001, 22100,
       21100, 31001, 36600, 53500, 49300, 17500, 26001, 23501,  9200,
       22650, 57500, 39600, 28100,  9500, 16700, 69500, 18001, 45500,
       15601, 16800, 22200, 39501, 19001, 20500, 14100, 12001, 34300,
       51200, 32500, 49500, 20600, 33500, 49001, 39001, 85001, 26501,
       18100, 62500, 15510, 14001, 27501, 16251, 26251, 10500, 29300,
       22250, 19020, 15001, 18250, 12300, 41500, 42200, 51850, 82500,
       15100, 45200, 44001, 20700, 28500, 51500, 28800, 53001, 75500,
```

```
       17250, 12700, 27200, 22150, 15400, 19750, 19200, 13050, 36200,
       69100, 38500, 16100, 29600, 24100, 18180, 31800, 42700, 14600,
       26700, 41001, 27001, 24200, 36500, 23001, 11100, 71200, 28200,
       19300,  9600, 26100, 22300, 42100, 46801, 10100, 21501, 33800,
       12600,  9001, 33001, 31600, 49750, 27300, 42001, 34500, 14900,
       43200, 22601, 33200, 11500, 18200, 31700, 38001, 15600, 31900,
       43001, 35001, 51001, 16501, 51501, 34001, 71501, 17201, 25800,
       62600, 22101, 27201, 75001, 54001, 36700, 15880, 20100, 22800,
       25200, 20250, 42801, 71001, 21701, 19700, 13001, 24501, 46600,
       14800, 71700,  9800, 19250, 12250, 61001, 41501, 24101, 41251,
       75501, 10001, 46001, 19400, 21700, 29080, 44500, 32001, 41400,
       75600, 46100, 45800, 14501, 10700, 13501, 85301, 17350, 13750,
       43300, 24751, 12200, 18101, 18800, 26200, 45100, 69800, 20150,
       22501, 11001, 36501, 43800, 69980, 82001, 19800, 62200, 19501,
       29100,  9700, 36350, 41200, 23700, 17801, 13801, 29400, 20300,
       19600, 17800, 17700, 31201, 21600, 29800, 12501, 31501, 19050,
       21801, 38281, 18601, 33501, 57600, 49101, 71950, 25400, 31301,
       51100, 17501, 71210, 42800, 82101, 16250, 23250, 41250, 26600,
       17600, 44750, 24900, 25251, 69600, 17100, 45101, 75350, 39050,
       22401, 17450, 41900, 29250, 45350, 32600, 61100, 21881, 10701,
       12750, 61501, 15251, 39200, 53101, 35500, 22400, 34501, 18550,
       28600, 43501, 69300, 38200, 46080, 54900, 31801, 44700, 17200,
       24701, 45601, 31400, 69101, 31750, 39100, 82600, 15800, 20501,
       21400, 38100, 23200, 43100, 61050, 26250, 39400, 13100, 13400,
       22350, 45501, 26881, 57201, 22201, 18600, 28700, 45600, 27100,
       26601, 12801, 51150, 43540, 12100, 69200, 71900, 33600, 23100,
       33100, 18400, 46700, 61101, 41450, 21050, 10600, 25250, 71901,
       14750, 21300, 27801, 44501, 49010, 38201, 82300, 60501, 37001,
       17301, 23801,  9380, 12701, 16101, 16200, 22801, 34200, 85250,
       45450, 23050, 45180, 53300, 11900,  9100, 19801, 71800, 13780,
       45880, 29801, 35222, 16151, 24800, 17101, 60600, 54750, 42600,
       18081, 22251, 14400, 17220, 27750, 85800, 25501, 49501, 42501,
       18201, 27251, 16901, 39550, 33601, 26801, 17701, 20601, 21250,
       71580, 18900, 26690, 25300, 57990, 25900, 33700, 26550, 29880,
       13550, 71990, 53900, 18350, 53501, 17750, 21201, 18801, 31200,
       22750, 27800, 37500, 46501, 35100, 25951, 26101, 17050, 18150,
       14901, 44751, 26800, 20140, 16600, 14201, 51800, 13601, 46601,
       28250, 31110, 25650, 18301, 21101, 17751, 29501, 15501, 24601,
```

```
Distribution_car_prices=df.groupby('Dealer_Region')['Price ($)'].describe()
print(Distribution_car_prices)
```

```
                count          mean           std     min      25%      50%  \
Dealer_Region
Aurora         3129.0  28329.300735  15025.653685  9000.0  18001.0  23000.0
Austin         4135.0  28341.603628  14903.884549  9000.0  18001.0  23801.0
Greenville     3128.0  28180.819054  15101.538328  1200.0  18001.0  22500.0
Janesville     3821.0  27833.350955  14344.995638  4300.0  18001.0  23000.0
Middletown     3128.0  27856.338875  14619.842395  1700.0  18000.0  22750.0
Pasco          3131.0  28119.039923  14659.315941  9000.0  18500.5  23000.0
Scottsdale     3433.0  27954.958928  14902.916820  1450.0  18000.0  22600.0

                   75%      max
Dealer_Region
Aurora         35000.0  85800.0
Austin         35001.0  85601.0
Greenville     34500.0  85200.0
Janesville     34000.0  85400.0
Middletown     34000.0  85300.0
Pasco          34000.0  85600.0
Scottsdale     33500.0  85001.0
```

```
#lets visualize  Distribution of car prices accross region using
plt.figure(figsize=(10, 6))
sns.barplot(x='Dealer_Region', y='Price ($)', data=df, palette='viridis')
plt.title('Distribution of Car Prices Across Regions')
plt.xlabel('Dealer Region')
plt.ylabel('Price ($)')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



5.What is the distribution of car prices based on body style?

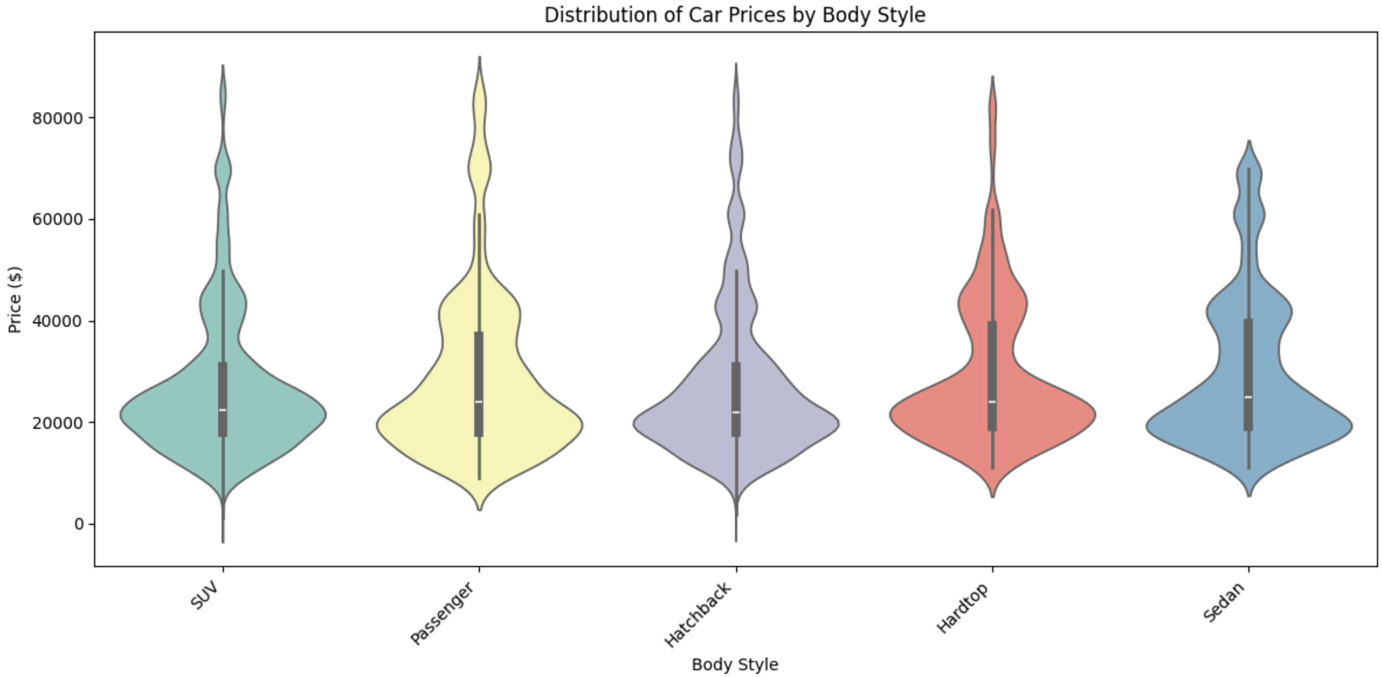
```
df.columns
```

```
Index(['Car_id', 'Date', 'Customer Name', 'Gender', 'Annual Income',
       'Dealer_Name', 'Company', 'Model', 'Engine', 'Transmission', 'Color',
       'Price ($)', 'Dealer_No ', 'Body Style', 'Phone', 'Dealer_Region'],
      dtype='object')
```

```
df['Body Style'].unique()
```

```
array(['SUV', 'Passenger', 'Hatchback', 'Hardtop', 'Sedan'], dtype=object)
```

```
plt.figure(figsize=(12, 6))
sns.violinplot(x='Body Style', y='Price ($)', data=df, palette='Set3')
plt.title('Distribution of Car Prices by Body Style')
plt.xlabel('Body Style')
plt.ylabel('Price ($)')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



6.How does the avarge selling price of cars vary by customer gender and annual income?

```
df.head()
```

	Car_id	Date	Customer Name	Gender	Annual Income	Dealer_Name	Company	Model	Engine	Transmission	Color	Price ($)	Deale
0	C_CND_000001	1/2/2022	Geraldine	Male	13500	Buddy Storbeck's Diesel Service Inc	Ford	Expedition	DoubleÂ Overhead Camshaft	Auto	Black	26000	0
1	C_CND_000002	1/2/2022	Gia	Male	1480000	C & M Motors Inc	Dodge	Durango	DoubleÂ Overhead Camshaft	Auto	Black	19000	6
2	C_CND_000003	1/2/2022	Gianna	Male	1035000	Capitol KIA	Cadillac	Eldorado	Overhead Camshaft	Manual	Red	31500	3
3	C_CND_000004	1/2/2022	Giselle	Male	13500	Chrysler of Tri-Cities	Toyota	Celica	Overhead Camshaft	Manual	Pale White	14000	9
4	C_CND_000005	1/2/2022	Grace	Male	1465000	Chrysler Plymouth	Acura	TL	DoubleÂ Overhead Camshaft	Auto	Red	24500	5

```
df.columns
```

```
Index(['Car_id', 'Date', 'Customer Name', 'Gender', 'Annual Income',
       'Dealer_Name', 'Company', 'Model', 'Engine', 'Transmission', 'Color',
       'Price ($)', 'Dealer_No ', 'Body Style', 'Phone', 'Dealer_Region'],
      dtype='object')
```

```
#avarage selling price of car
avg_selling_price_car = df.groupby('Company')['Price ($)'].mean()
print(avg_selling_price_car)
```

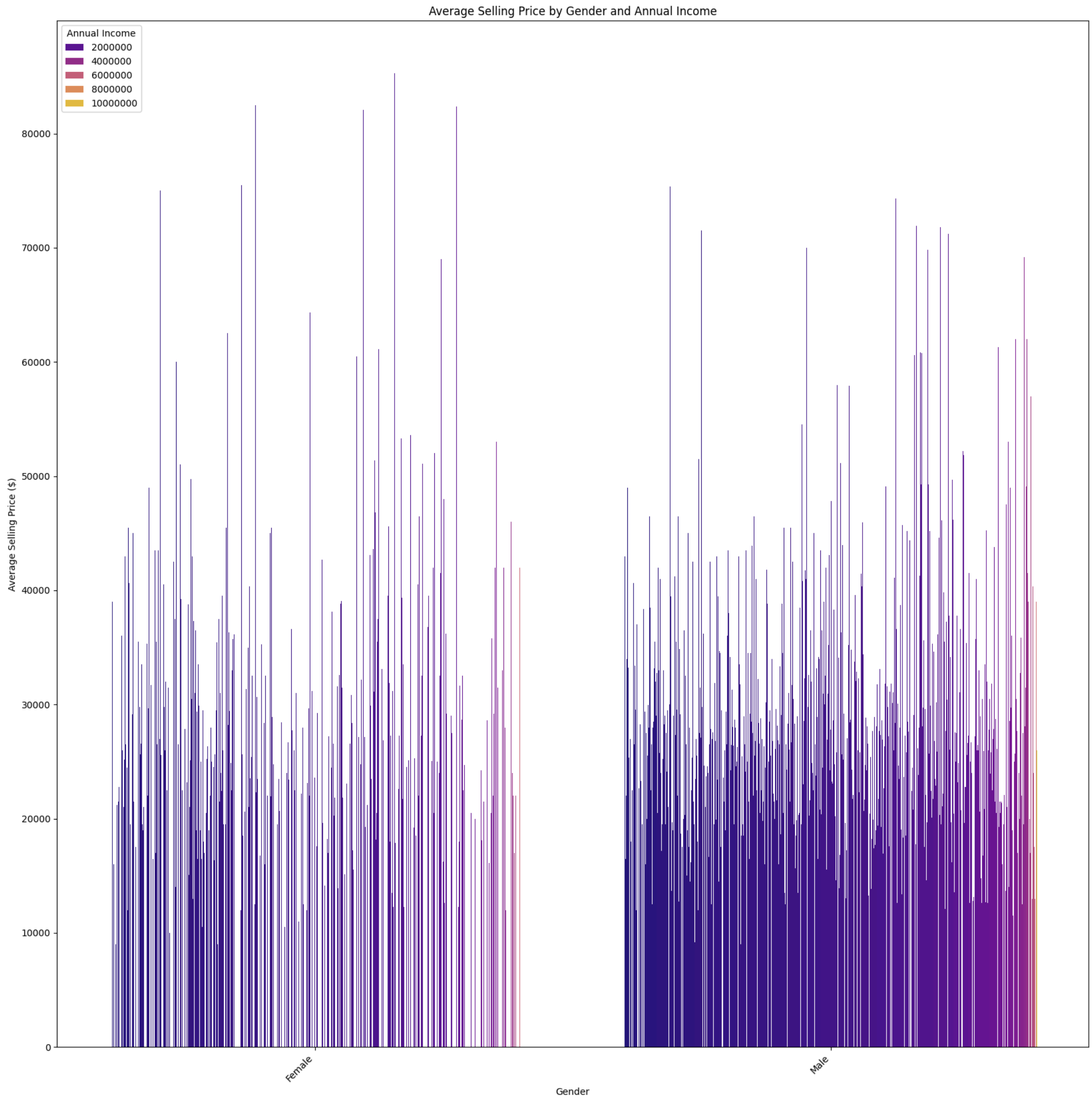
```
Company
Acura         24758.561684
Audi          22851.790598
BMW           25090.622785
Buick         33634.362187
Cadillac      40972.093558
Chevrolet     26198.606377
Chrysler      26019.529464
Dodge         26395.207186
```

```
Ford          29263.682156
Honda         28082.959040
Hyundai       19386.234848
Infiniti      29318.153846
Jaguar        25138.194444
Jeep          21057.338843
Lexus         34024.567332
Lincoln       31407.036585
Mercedes-B    26944.842802
Mercury       28535.163616
Mitsubishi    26673.818324
Nissan        27047.511287
Oldsmobile    31894.250225
Plymouth      29404.980551
Pontiac       29358.300251
Porsche       22674.894737
Saab          36516.338095
Saturn        31092.609215
Subaru        27931.340741
Toyota        29513.120721
Volkswagen    25568.552888
Volvo         27788.593156
Name: Price ($), dtype: float64
```

```
avg_price_by_gender_income = df.groupby(['Gender', 'Annual Income'])['Price ($)'].mean()
print(avg_price_by_gender_income)
```

```
Gender  Annual Income
Female  13500            28132.038732
        106000           46001.000000
        121000           20000.000000
        190000           19001.000000
        211000           51000.000000
                             ...
Male    6600000          39000.000000
        6800000          15000.000000
        7650000          21000.000000
        8000000          85000.000000
        11200000         26001.000000
Name: Price ($), Length: 3442, dtype: float64
```

```
# Calculate the average price by gender and annual income
avg_price_by_gender_income = df.groupby(['Gender', 'Annual Income'])['Price ($)'].mean().reset_index()

plt.figure(figsize=(20,20))
# Use the new DataFrame for plotting
sns.barplot(x='Gender', y='Price ($)', hue='Annual Income', data=avg_price_by_gender_income, palette='plasma')
plt.title('Average Selling Price by Gender and Annual Income')
plt.xlabel('Gender')
plt.ylabel('Average Selling Price ($)')
plt.xticks(rotation=45, ha='right')
plt.legend(title='Annual Income')
plt.show()
```



7.What is the distribution of car prices by region, and how does the number of cars sold vary by region?

```
df.columns
```

```
Index(['Car_id', 'Date', 'Customer Name', 'Gender', 'Annual Income',
       'Dealer_Name', 'Company', 'Model', 'Engine', 'Transmission', 'Color',
       'Price ($)', 'Dealer_No ', 'Body Style', 'Phone', 'Dealer_Region'],
      dtype='object')
```

```
df['Dealer_Region'].unique()
```

```
array(['Middletown', 'Aurora', 'Greenville', 'Pasco', 'Janesville',
       'Scottsdale', 'Austin'], dtype=object)
```

```
df['Dealer_Region'].value_counts()
```

Dealer_Region	count
Austin	4135
Janesville	3821
Scottsdale	3433
Pasco	3131
Aurora	3130
Middletown	3128
Greenville	3128

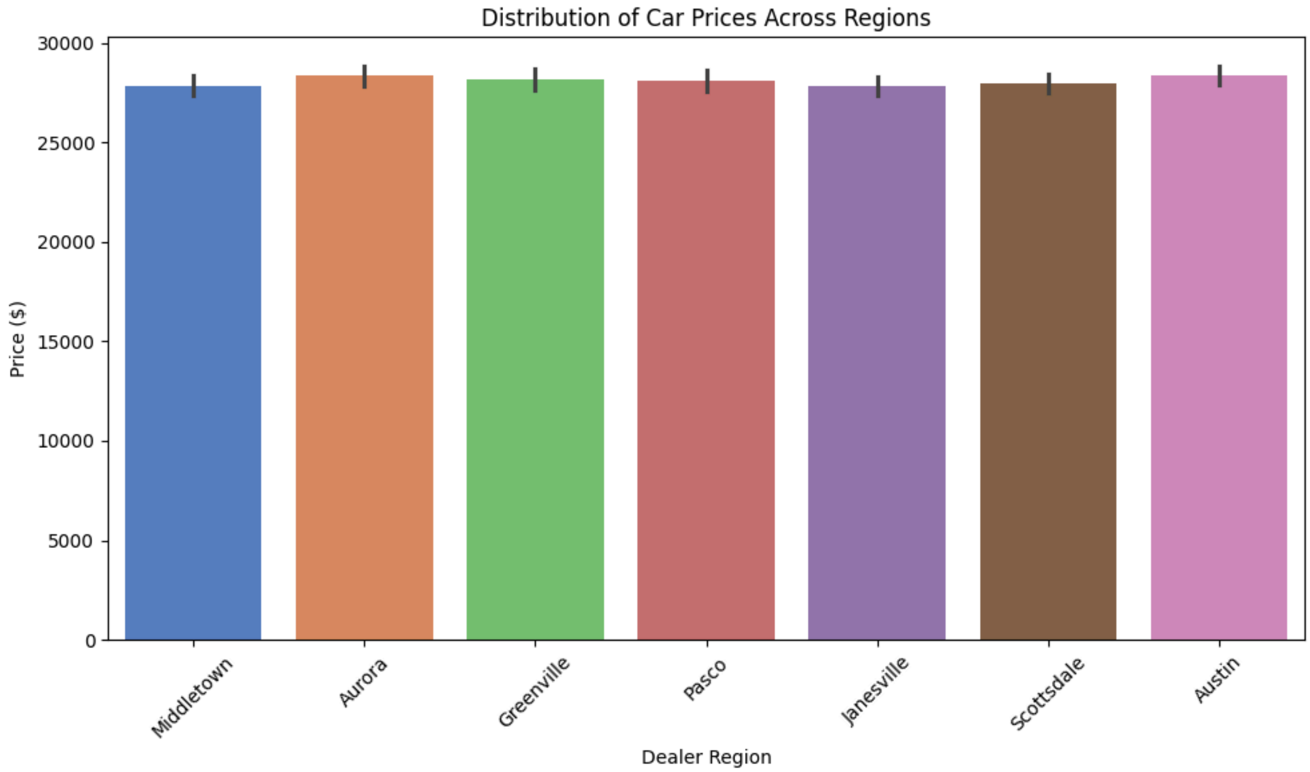
dtype: int64

```
distribution_car_prices_region = df.groupby('Dealer_Region')['Price ($)'].describe()
print(distribution_car_prices_region)
```

```
                count          mean           std     min      25%      50%  \
Dealer_Region
Aurora         3130.0  28334.626837  15026.207252  9000.0  18001.0  23000.0
Austin         4135.0  28341.603628  14903.884549  9000.0  18001.0  23801.0
Greenville     3128.0  28180.819054  15101.538328  1200.0  18001.0  22500.0
Janesville     3821.0  27833.350955  14344.995638  4300.0  18001.0  23000.0
Middletown     3128.0  27856.338875  14619.842395  1700.0  18000.0  22750.0
Pasco          3131.0  28119.039923  14659.315941  9000.0  18500.5  23000.0
Scottsdale     3433.0  27954.958928  14902.916820  1450.0  18000.0  22600.0

                   75%      max
Dealer_Region
Aurora         35000.0  85800.0
Austin         35001.0  85601.0
Greenville     34500.0  85200.0
Janesville     34000.0  85400.0
Middletown     34000.0  85300.0
Pasco          34000.0  85600.0
Scottsdale     33500.0  85001.0
```

Distribution Of Car Pricess across Regions

```
plt.figure(figsize=(10, 6))
sns.barplot(x='Dealer_Region', y='Price ($)',data=df, palette='muted')
plt.title('Distribution of Car Prices Across Regions')
plt.xlabel('Dealer Region')
plt.ylabel('Price ($)')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Insight:

Regional differences in factors like supply and demand, economic conditions, and taxes can also contribute to variations in car prices. we can also assume the standar devaition of car prices accross the region.

```
#number of car sold vary by region
car_sold_by_region_company= df.groupby('Dealer_Region')['Company'].count()
print(car_sold_by_region)
```

```
Dealer_Region  Company
Aurora         Chevrolet     223
               Dodge         214
               Ford          213
               Mercedes-B    191
               Mitsubishi    170
                            ...
Scottsdale     Hyundai        53
               Porsche        40
               Jaguar         35
               Saab           31
               Infiniti       21
Name: count, Length: 210, dtype: int64
```

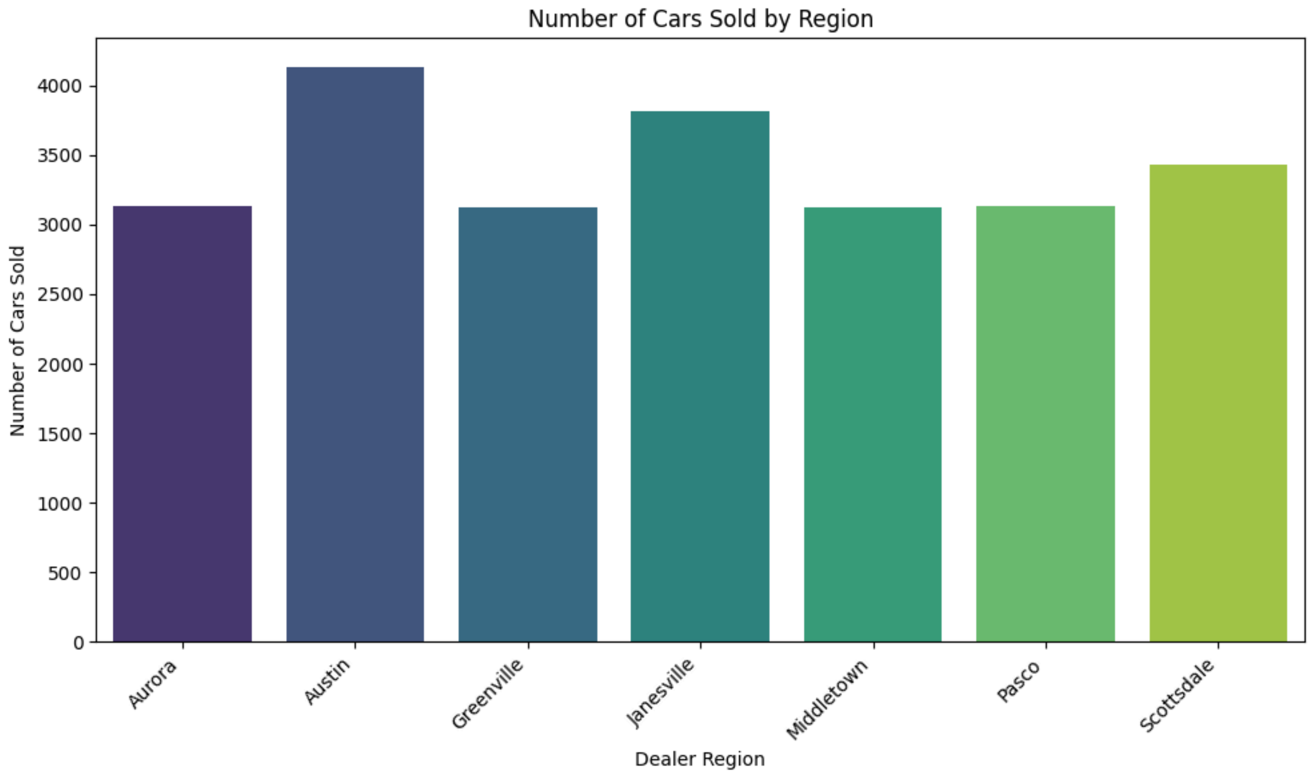
```
cars_sold_by_region = df.groupby('Dealer_Region')['Model'].count()
print(cars_sold_by_region)
```

```
Dealer_Region
Aurora        3130
Austin        4135
Greenville    3128
Janesville    3821
Middletown    3128
Pasco         3131
Scottsdale    3433
Name: Model, dtype: int64
```

Car Sales By Region

```
plt.figure(figsize=(10, 6))
sns.barplot(x=cars_sold_by_region.index, y=cars_sold_by_region.values, palette='viridis')
plt.title('Number of Cars Sold by Region')
plt.xlabel('Dealer Region')
```

```
plt.ylabel('Number of Cars Sold')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



Insights: previous output various type of car model is sold more than other region.this is austin.

8.How does the avarage car price differ between cars with different engine sizes?*

```
df.columns
```

```
Index(['Car_id', 'Date', 'Customer Name', 'Gender', 'Annual Income',
       'Dealer_Name', 'Company', 'Model', 'Engine', 'Transmission', 'Color',
       'Price ($)', 'Dealer_No ', 'Body Style', 'Phone', 'Dealer_Region'],
      dtype='object')
```

```
df['Engine'].unique()
```

```
array(['DoubleÂ\xa0Overhead Camshaft', 'Overhead Camshaft'], dtype=object)
```

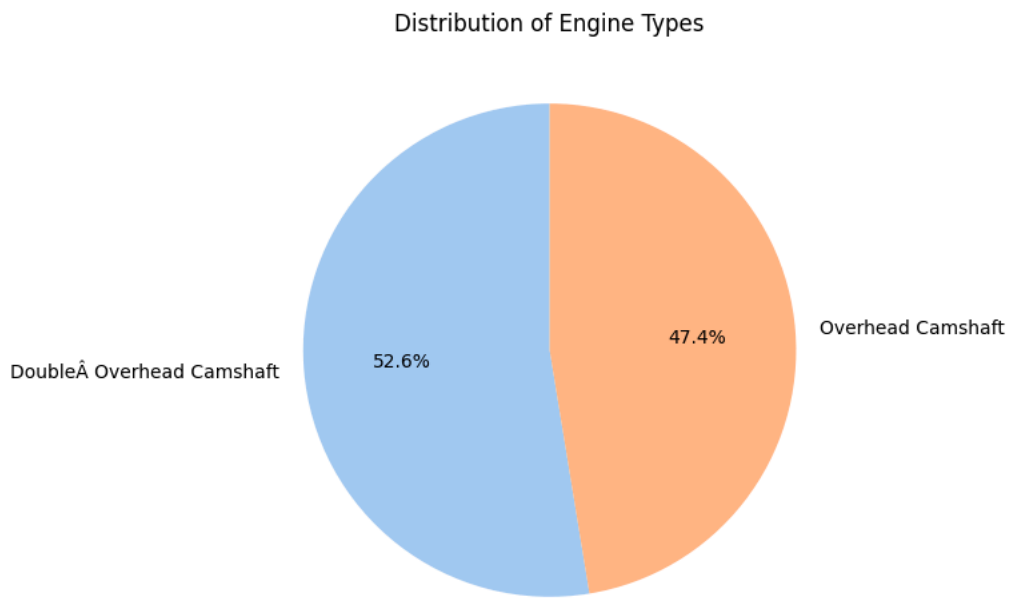
```
df['Engine'].value_counts()
```

	count
Engine	
DoubleÂ Overhead Camshaft	12571
Overhead Camshaft	11335

dtype: int64

Distribution Of Engine Types:

```
plt.figure(figsize=(10, 6))
plt.pie(df['Engine'].value_counts(), labels=df['Engine'].unique(), autopct='%1.1f%%', startangle=90, colors=sns.color_palette('pastel'))
plt.title('Distribution of Engine Types')
plt.show()
```

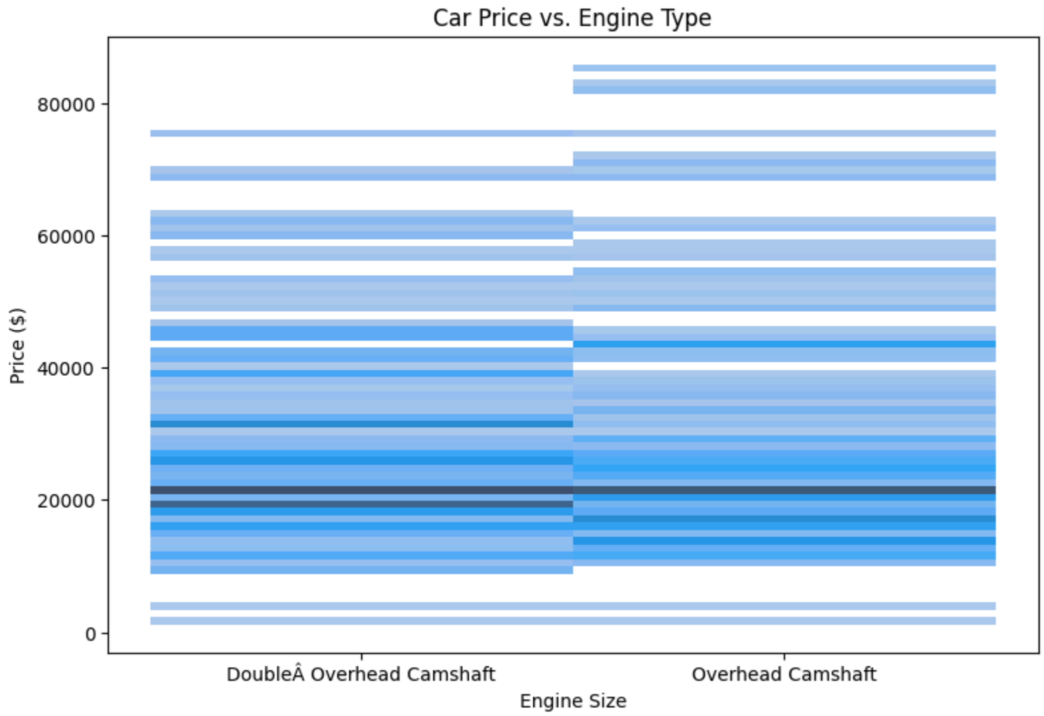



```
difference_car_price=df.groupby('Engine')['Price ($)'].describe()
print(difference_car_price)
```

```
                          count          mean           std     min  \
Engine
DoubleÂ Overhead Camshaft  12571.0  28248.525972  13747.070597  1200.0
Overhead Camshaft         11335.0  27914.710631  15862.871978  1700.0

                             25%      50%      75%      max
Engine
DoubleÂ Overhead Camshaft  19000.0  24000.0  35500.0  75700.0
Overhead Camshaft         17000.0  22001.0  34000.0  85800.0
```

Car Price Variation Vs Engine Type

```
plt.figure(figsize=(9, 6))
sns.histplot(x='Engine', y='Price ($)', data=df,palette="muted")
plt.title('Car Price vs. Engine Type')
plt.xlabel('Engine Size')
plt.ylabel('Price ($)')
plt.show()
```



9.How do car prices vary based on the customer's annual income bracket?

```
df.columns
```

```
Index(['Car_id', 'Date', 'Customer Name', 'Gender', 'Annual Income',
       'Dealer_Name', 'Company', 'Model', 'Engine', 'Transmission', 'Color',
       'Price ($)', 'Dealer_No ', 'Body Style', 'Phone', 'Dealer_Region'],
      dtype='object')
```

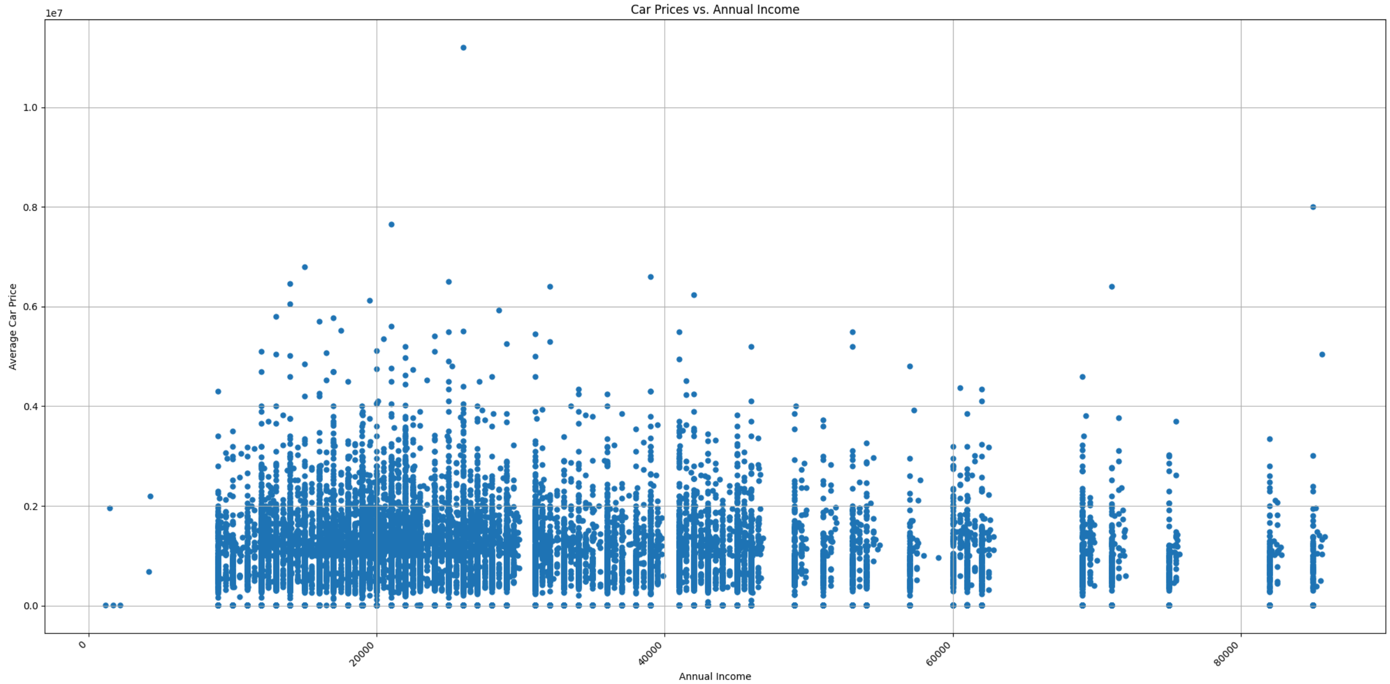
```
df['Annual Income'].unique()
```

```
array([  13500, 1480000, 1035000, ...,  374060, 4111000, 1369000])
```

Car Price vs annual Income

```
avg_price_by_income = df.groupby('Annual Income')['Price ($)'].mean()
```

```
plt.figure(figsize=(20,10))
sns.scatterplot(x=df['Price ($)'],y=df['Annual Income'],linewidth=0,palette="magma")
plt.title('Car Prices vs. Annual Income')
plt.xlabel('Annual Income')
plt.ylabel('Average Car Price')
plt.xticks(rotation=45, ha='right')
plt.grid(True)
plt.tight_layout()
plt.show()
```



Insights:

By examining the output and the visualization, you can draw insights about the relationship. For example:

Positive correlation: You might observe that as annual income increases, the average car price also tends to increase. This suggests a positive correlation between the two variables. Income brackets: You can identify specific income brackets where the average car price differs significantly. This can help understand the purchasing power of different income groups. Pricing strategies: The relationship can also provide insights into pricing strategies, such as targeting specific income brackets with different car models or price points.

10.What are the top 5 models with the highest number of sales and how does their price distribution lock?

```
df.columns
```

```
Index(['Car_id', 'Date', 'Customer Name', 'Gender', 'Annual Income',
       'Dealer_Name', 'Company', 'Model', 'Engine', 'Transmission', 'Color',
       'Price ($)', 'Dealer_No ', 'Body Style', 'Phone', 'Dealer_Region'],
      dtype='object')
```

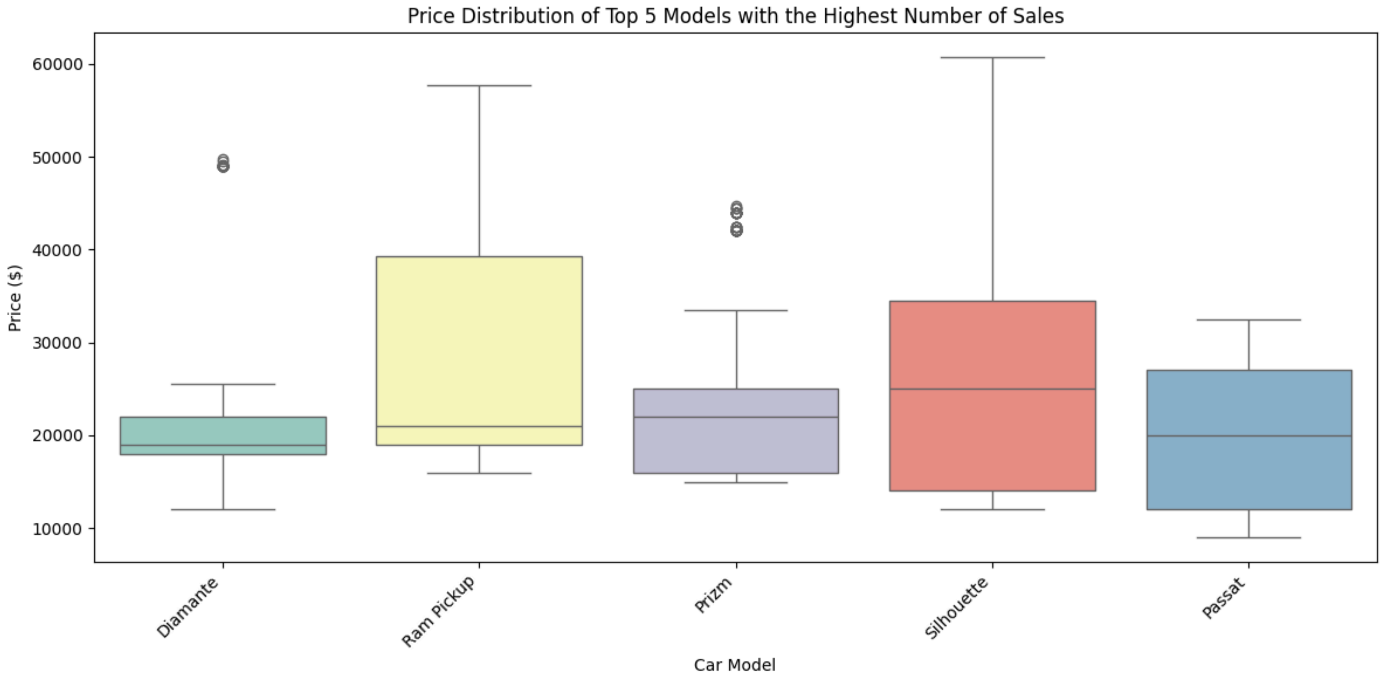
```
model_sales = df['Model'].value_counts().head(5)
print(model_sales)
```

```
Model
Diamante      418
Silhouette    411
Prizm         411
Passat        391
Ram Pickup    383
Name: count, dtype: int64
```

Price Distribution Vs Top 5 Model With Highest No Of Sales

```
#Visualize the price distribution of the top 5 models
plt.figure(figsize=(12, 6))
sns.boxplot(x='Model', y='Price ($)', data=df[df['Model'].isin(model_sales.index)], palette='Set3')
plt.title('Price Distribution of Top 5 Models with the Highest Number of Sales')
plt.xlabel('Car Model')
plt.ylabel('Price ($)')
```

```
plt.ylabel('Price ($)')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



11.How does car price vary with engine size accross different car colors,and which colors have the highest price variation?

```
df.columns
```

```
Index(['Car_id', 'Date', 'Customer Name', 'Gender', 'Annual Income',
       'Dealer_Name', 'Company', 'Model', 'Engine', 'Transmission', 'Color',
       'Price ($)', 'Dealer_No ', 'Body Style', 'Phone', 'Dealer_Region'],
      dtype='object')
```

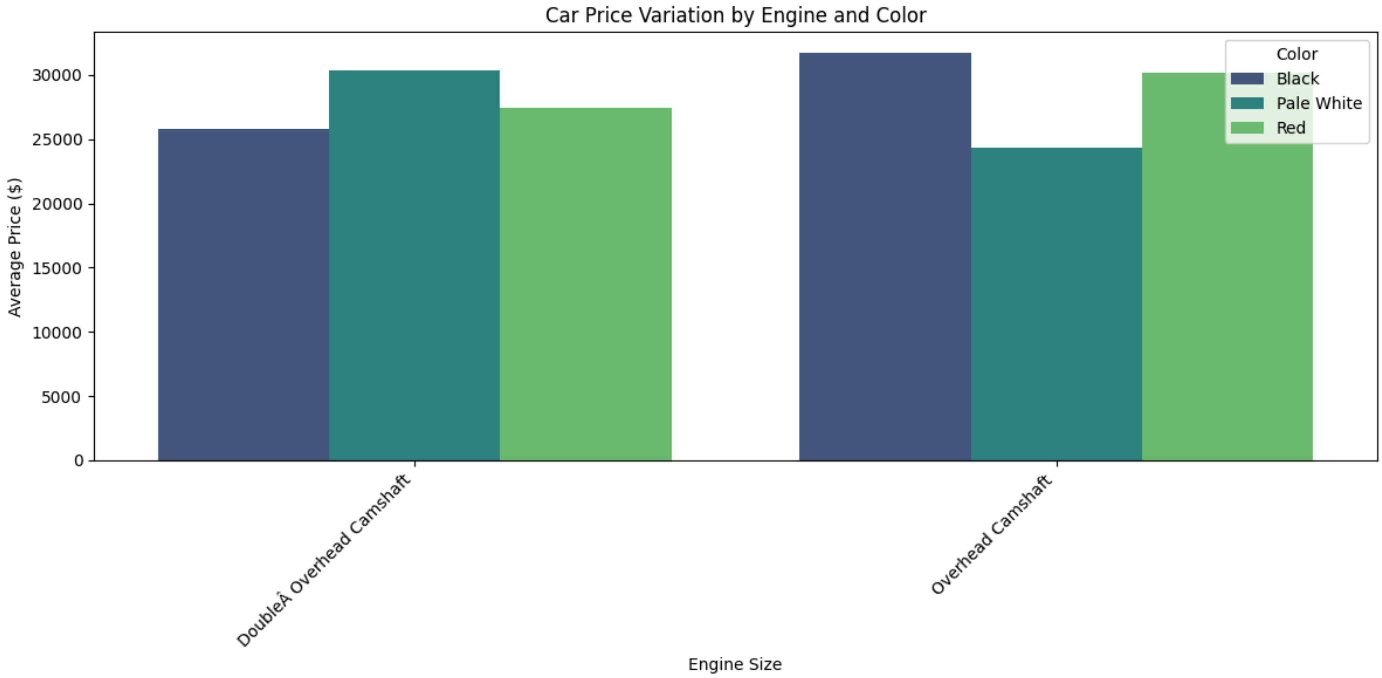
```
df['Color'].unique()
```

```
array(['Black', 'Red', 'Pale White'], dtype=object)
```

```
price_by_engine_color = df.groupby(['Engine', 'Color'])['Price ($)'].mean().reset_index()
```

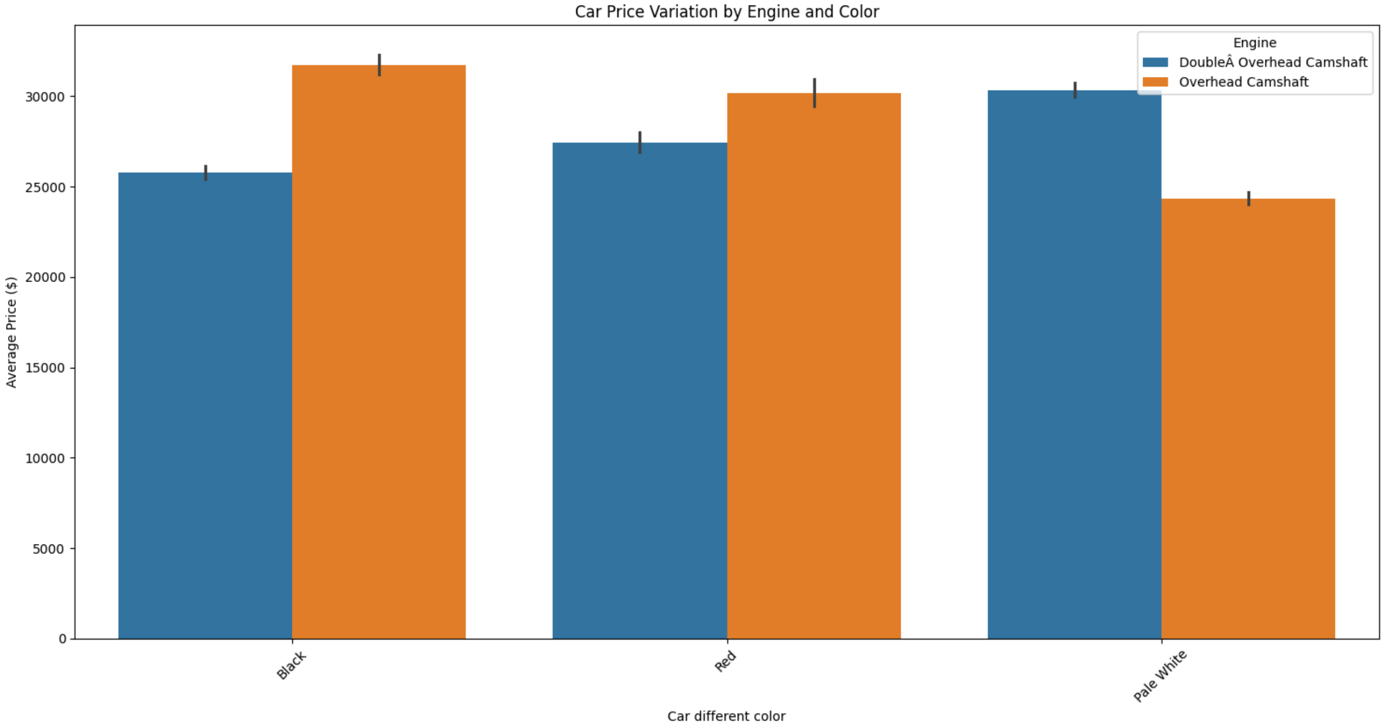
Car Price Variation By engine And Color

```
plt.figure(figsize=(12, 6))
sns.barplot(x='Engine', y='Price ($)', hue='Color', data=price_by_engine_color, palette='viridis')
plt.title('Car Price Variation by Engine and Color')
plt.xlabel('Engine Size')
plt.ylabel('Average Price ($)')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



Price Variation

```
#Visualize Price Variation:
plt.figure(figsize=(15, 8))
sns.barplot(x='Color', y='Price ($)', hue='Engine', data=df,)
plt.title('Car Price Variation by Engine and Color')
plt.xlabel('Car different color')
plt.ylabel('Average Price ($)')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Insight: visulalize this plot we can see different color and engine type car price are different .We can assume this distribution.

12.Is there any seasonal trend in car sales based on the date of sale?

```
df.columns
```

```
Index(['Car_id', 'Date', 'Customer Name', 'Gender', 'Annual Income',
       'Dealer_Name', 'Company', 'Model', 'Engine', 'Transmission', 'Color',
       'Price ($)', 'Dealer_No ', 'Body Style', 'Phone', 'Dealer_Region'],
      dtype='object')
```

```
# Data transformation
df['Date'].value_counts()
```

	count
Date	
9/5/2023	190
11/10/2023	175
12/29/2023	151
12/11/2023	140
11/24/2023	135
...	...
6/21/2022	5
7/12/2023	5
12/9/2022	5
7/8/2022	5
6/29/2023	5

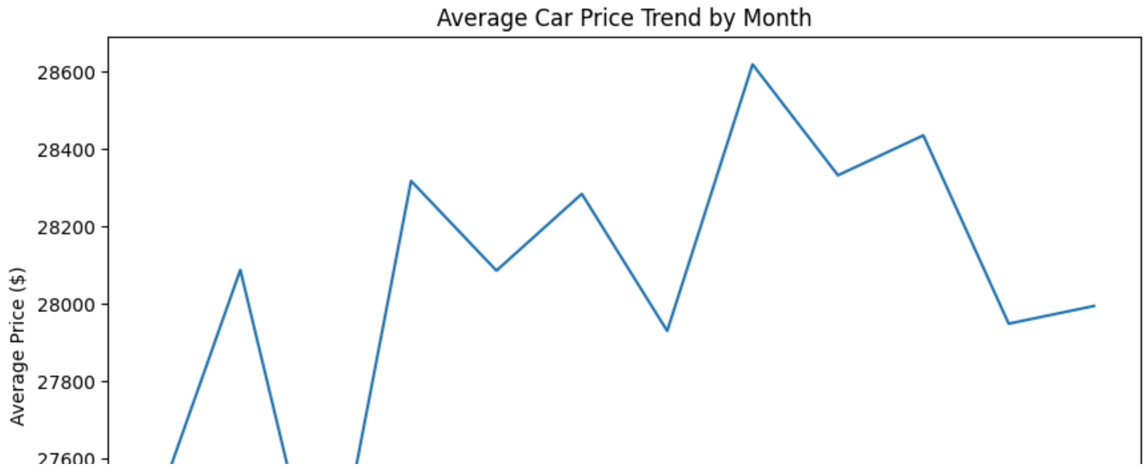
612 rows × 1 columns

dtype: int64

```
df['Date'] = pd.to_datetime(df['Date'])
#Extract Month and Quarter:
df['Month'] = df['Date'].dt.month
df['Quarter'] = df['Date'].dt.quarter

# For average price:
monthly_avg_price = df.groupby('Month')['Price ($)'].mean()

# For number of sales:
monthly_sales_count = df.groupby('Month')['Model'].count()
```

Monthly Sales Trend By Average Car Price

```
# For average price:
plt.figure(figsize=(10, 6))
sns.lineplot(x=monthly_avg_price.index, y=monthly_avg_price.values)
plt.title('Average Car Price Trend by Month')
plt.xlabel('Month')
plt.ylabel('Average Price ($)')
plt.show()

# For number of sales:
plt.figure(figsize=(10, 6))
sns.lineplot(x=monthly_sales_count.index, y=monthly_sales_count.values)
plt.title('Car Sales Trend by Month')
plt.xlabel('Month')
plt.ylabel('Number of Sales')
plt.show()
```


Average Car Price Trend by Month
Average Price ($)
28600
28400
28200
28000
27800
27600