CAR FEATURE OPTIMIZATION FOR PROFITABILITY

PRESENTED BY *MEGHNA HALDER*

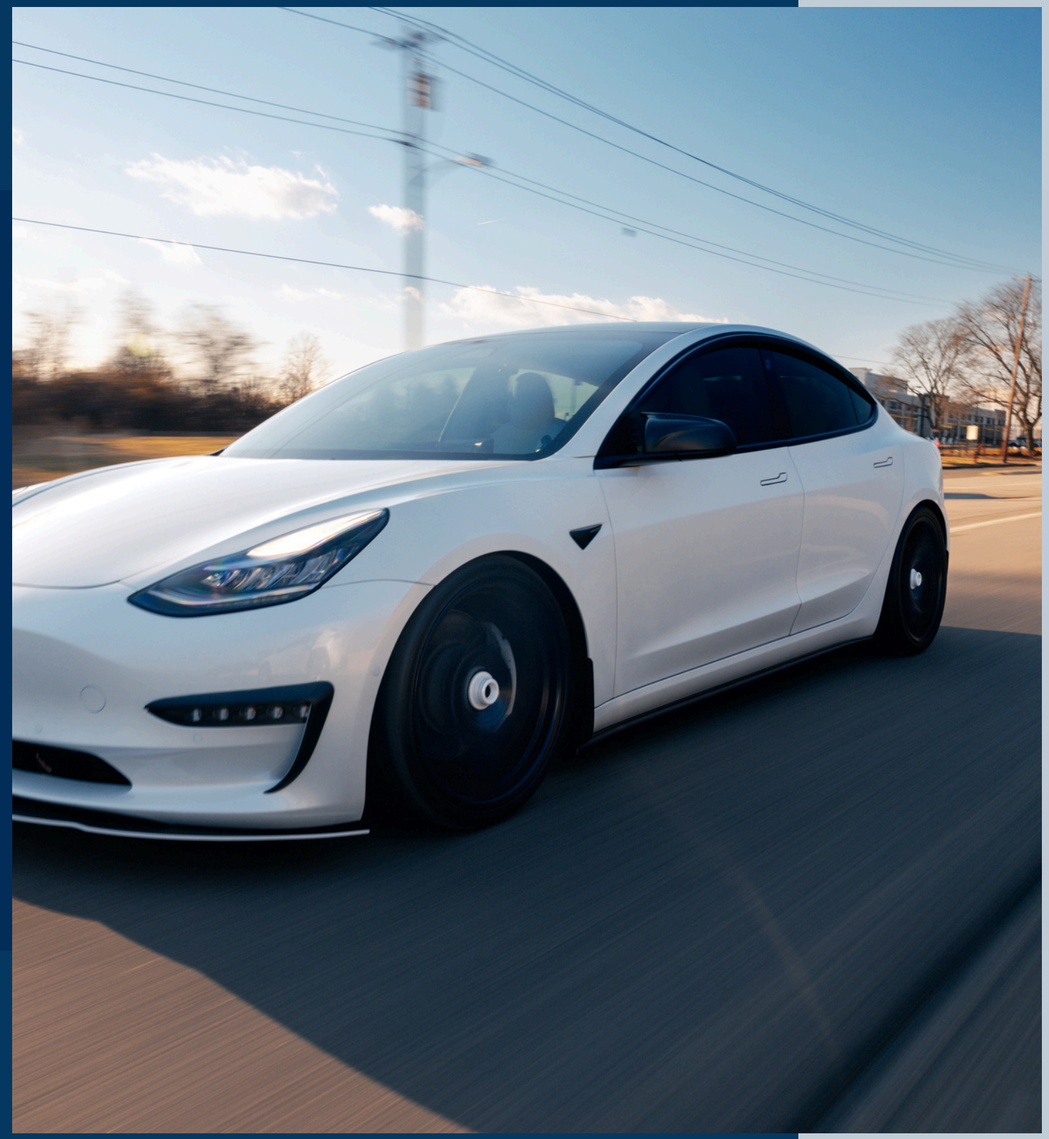
PROJECT DESCRIPTION

Overview:

This project aims to analyze how car features influence pricing, consumer demand, and profitability for manufacturers. The goal is to provide actionable insights to optimize pricing and guide product development.

Key Focus:

Understanding the factors driving car prices, consumer demand, and overall profitability by analyzing various car features and market categories.



BUSINESS PROBLEM

Key Question

HOW CAN A CAR MANUFACTURER OPTIMIZE PRICING AND PRODUCT DEVELOPMENT TO MAXIMIZE PROFITABILITY WHILE MEETING CONSUMER DEMAND?

Focus Areas

- CAR FEATURES: UNDERSTANDING WHICH FEATURES IMPACT PRICING AND DEMAND.
- MARKET CATEGORY: ANALYZING HOW DIFFERENT CATEGORIES AFFECT PROFITABILITY.
- PRICING RELATIONSHIPS: EXPLORING HOW CAR FEATURES AND MARKET CATEGORIES RELATE TO PRICING.

Objectives

- DEVELOP PRICING STRATEGIES: CREATE STRATEGIES TO BALANCE PRICING WITH CONSUMER DEMAND.
- IDENTIFY PROFITABLE FEATURES: DETERMINE WHICH FEATURES CONTRIBUTE MOST TO PROFITABILITY.
- OPTIMIZE PRODUCT DEVELOPMENT: GUIDE FUTURE PRODUCT DEVELOPMENT BASED ON FEATURE IMPORTANCE AND MARKET TRENDS.

APPROACH

METHODS USED

1

Descriptive statistics, visualizations, and regression analysis for trend identification and feature correlation with pricing.

REASONING

2

Chose these methods to summarize data, visualize relationships, and determine the impact of key features on price.

CHALLENGES

3

Handling missing data and ensuring consistency in data formats; some market variables affecting price couldn't be fully captured.

TECH-STACK USED

PYTHON: EMPLOYED FOR DATA MANIPULATION, CLEANING, AND EXPLORATORY DATA ANALYSIS

JUPYTER NOTEBOOKS: USED FOR DATA CLEANING AND PREPROCESSING

WPS EXCEL: USED FOR ANALYSIS TASKS, INCLUDING PIVOT TABLES AND REGRESSION ANALYSIS

TABLEAU PUBLIC 2024.2: CREATED VISUALIZATIONS AND INTERACTIVE DASHBOARDS FOR DEEPER INSIGHTS AND TREND ANALYSIS

DATASET OVERVIEW

Before data preprocessing the dataset had **11914 rows** and **16 columns**.

```
df.head()
```

	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
	BMW	1 Series M	2011	premium unleaded (required)	335.0	6.0	MANUAL	rear wheel drive	2.0	Factory Tuner,Luxury,High-Performance	Compact	Coupe	26	19	3916	46135
	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact	Convertible	28	19	3916	40650
	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,High-Performance	Compact	Coupe	28	20	3916	36350
	BMW	1 Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact	Coupe	28	18	3916	29450
	BMW	1 Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury	Compact	Convertible	28	18	3916	34500

```
df.shape
```

```
(11914, 16)
```

DATA PREPROCESSING

1. Checked for duplicates.

```
[5]: df[df.duplicated(keep=False)]
```

[5]:

	Make	Model	Year	Engine Fuel Type	Engine HP	Engine Cylinders	Transmission Type	Driven_Wheels	Number of Doors	Market Category	Vehicle Size
11	BMW	1 Series	2013	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact
14	BMW	1 Series	2013	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact
17	Audi	100	1992	regular unleaded	172.0	6.0	MANUAL	front wheel drive	4.0	Luxury	Midsize
18	Audi	100	1992	regular unleaded	172.0	6.0	MANUAL	front wheel drive	4.0	Luxury	Midsize
				regular				front wheel			

Then, dropped all the duplicates except the first one.

```
[6]: df.drop_duplicates(inplace=True)
```

DATA PREPROCESSING

2. Found out the total number of null values.

[8]:
```
dft_1=pd.DataFrame()
dft_1['Null']=df.isna().sum().sort_values(ascending=False)
dft_1[dft_1['Null']>0]
```

[8]:

	Null
Market Category	3376
Engine HP	69
Engine Cylinders	30
Number of Doors	6
Engine Fuel Type	3

DATA PREPROCESSING

3. Getting value of Engine HP by searching for rows with same Make, Model and Year. If found then replaced null value with the mode of Engine HP value of all the matching rows.

[11]:
```
for i in dft_2.index:
    mk=dft_2['Make'].loc[i]
    mdl=dft_2['Model'].loc[i]
    yr=dft_2['Year'].loc[i]

    d=dft_3[(dft_3['Make']==mk) & (dft_3['Model']==mdl) & (dft_3['Year']==yr)]['Engine HP'].values
    if len(d)==0:
        d=dft_3[(dft_3['Make']==mk) & (dft_3['Model']==mdl)]['Engine HP'].values

    try:
        md=st.mode(d[d!=None])
    except:
        continue

    df['Engine HP'].loc[i]=md
```

[12]:
```
df[df['Engine HP'].isna()].shape
```

[12]:
```
(38, 16)
```

DATA PREPROCESSING

Some of the values are still NULL, and they turned out to be electric cars.

```
[13]: dft_2=df[df['Engine HP'].isna()]
      dft_2
```

	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
541	FIAT	500e	2017	electric	NaN	0.0	DIRECT_DRIVE	front wheel drive	2.0	Hatchback	Compact	2dr Hatchback	103	121	819
4705	Honda	Fit EV	2013	electric	NaN	0.0	DIRECT_DRIVE	front wheel drive	4.0	Hatchback	Compact	4dr Hatchback	105	132	2202
4706	Honda	Fit EV	2014	electric	NaN	0.0	DIRECT_DRIVE	front wheel drive	4.0	Hatchback	Compact	4dr Hatchback	105	132	2202
6385	Nissan	Leaf	2014	electric	NaN	0.0	DIRECT_DRIVE	front wheel drive	4.0	Hatchback	Compact	4dr Hatchback	101	126	2009
6386	Nissan	Leaf	2014	electric	NaN	0.0	DIRECT_DRIVE	front wheel drive	4.0	Hatchback	Compact	4dr Hatchback	101	126	2009
6387	Nissan	Leaf	2014	electric	NaN	0.0	DIRECT_DRIVE	front wheel drive	4.0	Hatchback	Compact	4dr Hatchback	101	126	2009
6388	Nissan	Leaf	2015	electric	NaN	0.0	DIRECT_DRIVE	front wheel	4.0	Hatchback	Compact	4dr	101	126	2009

DATA PREPROCESSING

Dealt with the NULL values of electric car by replacing them with median value of Engine HP value of all the matching rows.

[14]:
```
# Check the number of missing values in Engine HP before imputation
missing_before = df['Engine HP'].isna().sum()
print(f"Missing values in 'Engine HP' before imputation: {missing_before}")

# Perform Median Imputation
df['Engine HP'].fillna(df['Engine HP'].median(), inplace=True)

# Check the number of missing values in Engine HP after imputation
missing_after = df['Engine HP'].isna().sum()
print(f"Missing values in 'Engine HP' after imputation: {missing_after}")
```

```
Missing values in 'Engine HP' before imputation: 38
Missing values in 'Engine HP' after imputation: 0
```

Finally we fill up all the NULL values in Engine HP column.

```
df[df['Engine HP'].isna()].shape
```

```
(0, 16)
```

DATA PREPROCESSING

4 a. Dealing with Null values in Engine Cylinders column where Engine Fuel Type is electric

```
dft_3=df[~(df['Engine Cylinders'].isna())]
dft_3.head()
```

	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	MSRI
0	BMW	1 Series M	2011	premium unleaded (required)	335.0	6.0	MANUAL	rear wheel drive	2.0	Factory Tuner,Luxury,High-Performance	Compact	Coupe	26	19	3916	4613
1	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact	Convertible	28	19	3916	4065
2	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,High-Performance	Compact	Coupe	28	20	3916	3635
3	BMW	1 Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact	Coupe	28	18	3916	2945
4	BMW	1 Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury	Compact	Convertible	28	18	3916	3450

```
dft_3[dft_3['Engine Fuel Type']=='electric']['Engine Cylinders'].unique()
```

```
array([0.])
```

```
df['Engine Cylinders'].loc[dft_2[dft_2['Engine Fuel Type']=='electric'].index]=0.0
```

Action: We found that electric Cars have 0 Engine Cylinders which is logical. So replaced null Engine Cylinders row values whose Engine Fuel Type is electric with 0.

DATA PREPROCESSING

4 b. Dealing with Null values in Engine Cylinders column where Engine Fuel Type is other than electric.

[21]:
```
dft_2=df[df['Engine Cylinders'].isna()]
dft_2
```

[21]:

	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
695	Mazda	RX-7	1993	regular unleaded	255.0	NaN	MANUAL	rear wheel drive	2.0	Factory Tuner,Performance	Compact	Coupe	23	15	586	7523
696	Mazda	RX-7	1994	regular unleaded	255.0	NaN	MANUAL	rear wheel drive	2.0	Factory Tuner,Performance	Compact	Coupe	23	15	586	8147
697	Mazda	RX-7	1995	regular unleaded	255.0	NaN	MANUAL	rear wheel drive	2.0	Factory Tuner,Performance	Compact	Coupe	23	15	586	8839
698	Mazda	RX-8	2009	premium unleaded (required)	232.0	NaN	MANUAL	rear wheel drive	4.0	Performance	Compact	Coupe	22	16	586	31930
699	Mazda	RX-8	2009	premium unleaded (required)	212.0	NaN	AUTOMATIC	rear wheel drive	4.0	Performance	Compact	Coupe	23	16	586	26435

[22]:
```
#https://www.cars-data.com/en/mazda-rx-7/cylinders
df['Engine Cylinders'].loc[dft_2[dft_2['Model']=='RX-7'].index]=2.0

#https://www.cardekho.com/mazda/mazda-rx-8-specifications.htm
df['Engine Cylinders'].loc[dft_2[dft_2['Model']=='RX-8'].index]=4.0
```

Action: Only two Car models had null values of Engine Cylinders. So searched for the information online and replaced them with the correct values.

DATA PREPROCESSING

5. Dealing with Null values in Number of Doors column.

```
dft_2=df[df['Number of Doors'].isna()]
dft_2
```

	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
4666	Ferrari	FF	2013	premium unleaded (required)	651.0	12.0	AUTOMATED_MANUAL	all wheel drive	NaN	Exotic,High-Performance	Large	Coupe	16	11	2774
6930	Tesla	Model S	2016	electric	239.0	0.0	DIRECT_DRIVE	all wheel drive	NaN	Exotic,Performance	Large	Sedan	105	102	1391
6931	Tesla	Model S	2016	electric	239.0	0.0	DIRECT_DRIVE	all wheel drive	NaN	Exotic,Performance	Large	Sedan	101	98	1391
6932	Tesla	Model S	2016	electric	239.0	0.0	DIRECT_DRIVE	all wheel drive	NaN	Exotic,High-Performance	Large	Sedan	105	92	1391
6933	Tesla	Model S	2016	electric	239.0	0.0	DIRECT_DRIVE	rear wheel drive	NaN	Exotic,Performance	Large	Sedan	100	97	1391
6934	Tesla	Model S	2016	electric	239.0	0.0	DIRECT_DRIVE	all wheel drive	NaN	Exotic,Performance	Large	Sedan	107	101	1391

```
#https://www.cars.com/research/ferrari-ff-2013/
df['Number of Doors'].loc[dft_2[dft_2['Model']=='FF'].index]=2.0

#https://www.carexpert.com.au/tesla/model-s/2016-70-20f0b307
df['Number of Doors'].loc[dft_2[dft_2['Model']=='Model S'].index]=4.0
```

Action: Only two Car models had null values of Number of Doors. So searched for the information online and replaced them with the correct values.

DATA PREPROCESSING

6. Dealing with Null values in Engine Fuel Type column.

```
dft_2=df[df['Engine Fuel Type'].isna()]
dft_2
```

	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
11321	Suzuki	Verona	2004	NaN	155.0	6.0	AUTOMATIC	front wheel drive	4.0	NaN	Midsize	Sedan	25	17	481	17199
11322	Suzuki	Verona	2004	NaN	155.0	6.0	AUTOMATIC	front wheel drive	4.0	NaN	Midsize	Sedan	25	17	481	20199
11323	Suzuki	Verona	2004	NaN	155.0	6.0	AUTOMATIC	front wheel drive	4.0	NaN	Midsize	Sedan	25	17	481	18499

```
#https://www.fueleconomy.gov/feg/noframes/19808.shtml
df['Engine Fuel Type'].fillna('regular unleaded', inplace=True)
```

<u>Action</u>: Only one Car model had null values of Engine Fuel Type. So searched for the information online and replaced them with the correct values.

DATA PREPROCESSING

7 a. Dealing with Null values in Market Category column.

```
[30]: df=pd.concat([df, df['Market Category'].str.split(',', expand=True)], axis=1)
```

```
[31]: df.rename(columns={0: 'Mkt_Cat_0', 1: 'Mkt_Cat_1', 2: 'Mkt_Cat_2', 3: 'Mkt_Cat_3', 4: 'Mkt_Cat_4'}, inplace=True)
```

```
[32]: df.head()
```

[32]:

	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	MSRI
0	BMW	1 Series M	2011	premium unleaded (required)	335.0	6.0	MANUAL	rear wheel drive	2.0	Factory Tuner,Luxury,High-Performance	Compact	Coupe	26	19	3916	4613
1	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact	Convertible	28	19	3916	4065
2	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,High-Performance	Compact	Coupe	28	20	3916	3635
3	BMW	1 Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact	Coupe	28	18	3916	2945
4	BMW	1 Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury	Compact	Convertible	28	18	3916	3450

```
dft_2=df[df['Market Category'].isna()]
```

```
dft_3=df[df['Market Category'].notna()]
```

```
for i in dft_2.index:
    mk=dft_2['Make'].loc[i]
    mdl=dft_2['Model'].loc[i]

    d=dft_3[(dft_3['Make']==mk) & (dft_3['Model']==mdl)][['Mkt_Cat_0', 'Mkt_Cat_1', 'Mkt_Cat_2', 'Mkt_Cat_3', 'Mkt_Cat_4']].to_numpy().flatten()
    try:
        md=st.mode(d[d!=None])
    except:
        continue

    df['Market Category'].loc[i]=md
```

Action:

- Separated the Categories into different columns.
- Searched for Market Categories of all rows of same Car Make and Model in non null values dataframe.
- Found mode of the Market Categories and replaced null value with the mode.

DATA PREPROCESSING

7 b. Dealing with Null values in Market Category column which were not affected by the previous process.

```
param={'C':list(np.arange(0.0,1.0,0.1))}

lr=LogisticRegression(random_state=42, n_jobs=-1)

model = GridSearchCV(lr, param, scoring='accuracy', cv=5, return_train_score=True, verbose=4)
model.fit(X_tr_enc, y_tr_enc)
```

```
Fitting 5 folds for each of 10 candidates, totalling 50 fits
[CV 1/5] END .............C=0.0;, score=(train=nan, test=nan) total time=   0.0s
[CV 2/5] END .............C=0.0;, score=(train=nan, test=nan) total time=   0.0s
```

```
lr=LogisticRegression(random_state=42, n_jobs=-1, C=model.best_params_['C'])
lr.fit(X_tr_enc, y_tr_enc)
```

```
print("Train Data Accuracy: ", accuracy_score(y_tr_enc, pred_y_tr))
print("Test Data Accuracy: ", accuracy_score(y_te_enc, pred_y_te))
```

```
Train Data Accuracy:  0.6534196840349328
Test Data Accuracy:  0.6099499558433912
```

```
pred_y_pr = lr.predict_proba(X_pr_enc)
```

```
for idx,i in enumerate(dft_2.index):
    mk=pred_y_pr[idx]
    mk_5=mk[np.argsort(mk)[::-1][:5]]
    l=[]
    for j in range(0,4):
        l.append(mk_5[j])
        if (mk_5[j]-mk_5[j+1])>0.1:
            break

    mark_cat=', '.join(list(le.inverse_transform(np.sort(np.nonzero(np.in1d(mk,l))[0])[::-1])))
    df['Market Category'].loc[i]=mark_cat
```

<u>Action</u>: Performed Multinomial Logistic Regression to get the top 4 Market Categories for the rows with null values of Market Category.

DATA PREPROCESSING

8 a. Checking of errors in Categorical columns.

```
for i in df:
    if (df[i].dtype=='object'):
        print(i, ' - ', df[i].unique())
        print('\n')
```

```
Make  -  ['BMW' 'Audi' 'FIAT' 'Mercedes-Benz' 'Chrysler' 'Nissan' 'Volvo' 'Mazda'
 'Mitsubishi' 'Ferrari' 'Alfa Romeo' 'Toyota' 'McLaren' 'Maybach'
 'Pontiac' 'Porsche' 'Saab' 'GMC' 'Hyundai' 'Plymouth' 'Honda'
 'Oldsmobile' 'Suzuki' 'Ford' 'Cadillac' 'Kia' 'Bentley' 'Chevrolet'
 'Dodge' 'Lamborghini' 'Lincoln' 'Subaru' 'Volkswagen' 'Spyker' 'Buick'
 'Acura' 'Rolls-Royce' 'Maserati' 'Lexus' 'Aston Martin' 'Land Rover'
 'Lotus' 'Infiniti' 'Scion' 'Genesis' 'HUMMER' 'Tesla' 'Bugatti']


Model  -  ['1 Series M' '1 Series' 100 '124 Spider' '190-Class' '2 Series' 200
 '200SX' '240SX' 240 2 '3 Series Gran Turismo' '3 Series' '300-Class'
 '3000GT' 300 '300M' '300ZX' 323 '350-Class' '350Z' 360 '370Z' 3
 '4 Series Gran Coupe' '4 Series' '400-Class' '420-Class' '456M'
 '458 Italia' '4C' '4Runner' '5 Series Gran Turismo' '5 Series'
 '500-Class' '500e' 500 '500L' '500X' 550 '560-Class' '570S' '575M' 57 599
 5 '6 Series Gran Coupe' '6 Series' '600-Class' 6000 '612 Scaglietti' 626
 62 '650S Coupe' '650S Spider' 6 '7 Series' '718 Cayman' 740 760 780
 '8 Series' 80 850 86 '9-2X' '9-3 Griffin'
 datetime.datetime(2023, 3, 9, 0, 0) '9-4X'
 datetime.datetime(2023, 5, 9, 0, 0) '9-7X' 9000 900 90 911 928 929 940
 944 960 968 'A3' 'A4 allroad' 'A4' 'A5' 'A6' 'A7' 'A8' 'Acadia Limited'
 'Acadia' 'Accent' 'Acclaim' 'Accord Crosstour' 'Accord Hybrid'
 'Accord Plug-In Hybrid' 'Accord' 'Achieva' 'ActiveHybrid 5'
 'ActiveHybrid 7' 'ActiveHybrid X6' 'Aerio' 'Aerostar' 'Alero' 'Allante'
 'allroad quattro' 'allroad' 'ALPINA B6 Gran Coupe' 'ALPINA B7' 'Alpina'
```

DATA PREPROCESSING

8 b. Dealing with errors in Transmission Type column.

```
df[df['Transmission Type']=='UNKNOWN']
```

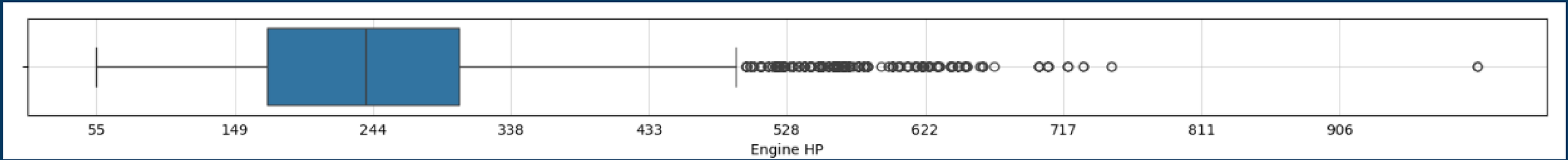
	Make	Model	Year	Engine Fuel Type	Engine HP	Engine Cylinders	Transmission Type	Driven_Wheels	Number of Doors	Market Category	Vehicle Size	Vehicle Style	highway MPG	city mpg	Pop
1289	Oldsmobile	Achieva	1997	regular unleaded	150	4	UNKNOWN	front wheel drive	2	Performance	Midsize	Coupe	29	19	
1290	Oldsmobile	Achieva	1997	regular unleaded	150	4	UNKNOWN	front wheel drive	4	Performance	Midsize	Sedan	29	19	
4691	Pontiac	Firebird	2000	regular unleaded	305	8	UNKNOWN	rear wheel drive	2	Hatchback,Performance	Midsize	2dr Hatchback	23	15	
4692	Pontiac	Firebird	2000	regular unleaded	305	8	UNKNOWN	rear wheel drive	2	Hatchback,Factory Tuner,Performance	Midsize	2dr Hatchback	23	15	
4693	Pontiac	Firebird	2000	regular unleaded	305	8	UNKNOWN	rear wheel drive	2	Factory Tuner,Performance	Midsize	Convertible	23	15	
6158	GMC	Jimmy	1999	regular unleaded	190	6	UNKNOWN	rear wheel drive	2	Performance	Compact	2dr SUV	19	14	

```
#https://www.auto123.com/en/new-cars/technical-specs/oldsmobile/achieva/1997/2-dr/sc/
df['Transmission Type'].loc[df[(df['Make']=='Oldsmobile') & (df['Model']=='Achieva') & (df['Year']==1997) &(df['Transmission Type']=='UNKNOWN')].index]=

#https://www.newcartestdrive.com/reviews/2000-pontiac-firebird/
df['Transmission Type'].loc[df[(df['Make']=='Pontiac') & (df['Model']=='Firebird') & (df['Year']==2000) &(df['Transmission Type']=='UNKNOWN')].index]='A

#https://www.autoblog.com/buy/1999-GMC-Jimmy-SL__4dr_4x4/equipment/
df['Transmission Type'].loc[df[(df['Make']=='GMC') & (df['Model']=='Jimmy') & (df['Year']==1999) &(df['Transmission Type']=='UNKNOWN')].index]='AUTOMATI

#https://www.autoblog.com/buy/2000-GMC-Jimmy/specs/
df['Transmission Type'].loc[df[(df['Make']=='GMC') & (df['Model']=='Jimmy') & (df['Year']==2000) &(df['Transmission Type']=='UNKNOWN')].index]='AUTOMATI

#https://www.edmunds.com/chrysler/le-baron/1993/st-13462/features-specs/
df['Transmission Type'].loc[df[(df['Make']=='Chrysler') & (df['Model']=='Le Baron') & (df['Year']==1993) &(df['Transmission Type']=='UNKNOWN')].index]='

#https://www.autodetective.com/directory/1991/dodge/ram-150/
df['Transmission Type'].loc[df[(df['Make']=='Dodge') & (df['Model']=='RAM 150') & (df['Year']==1991) &(df['Transmission Type']=='UNKNOWN')].index]='AUTO
```

Action: Searched for the information online and replaced the value ‘UNKNOWN’ with correct ones.

DATA PREPROCESSING

9. Dealing with Outliers in Engine HP column.




```
df[df['Engine HP']>520]['Market Category'].value_counts()
```

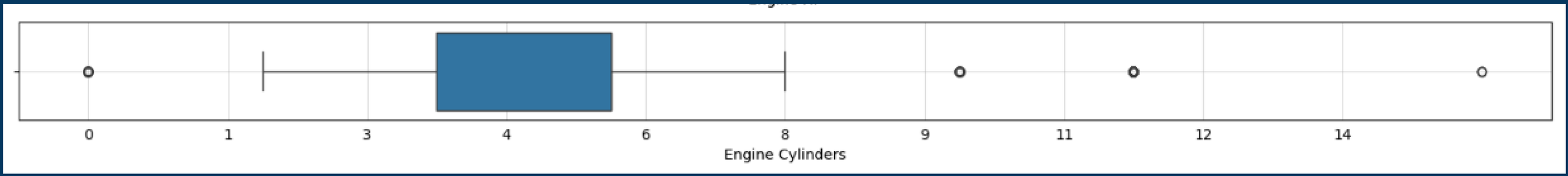
Market Category	
Exotic,High-Performance	109
Factory Tuner,Luxury,High-Performance	73
Factory Tuner,High-Performance	41
Luxury,High-Performance	33
Exotic,Factory Tuner,Luxury,High-Performance	29
Exotic,Luxury,High-Performance	21
Crossover,Factory Tuner,Luxury,High-Performance	16
Exotic,Luxury,Performance	13
Exotic,Flex Fuel,Factory Tuner,Luxury,High-Performance	13
Exotic,Factory Tuner,High-Performance	11
Exotic,Flex Fuel,Luxury,High-Performance	11
Exotic,Luxury	6
High-Performance	6
Crossover,Luxury,High-Performance	2
Factory Tuner,Luxury	2
Exotic,Luxury,High-Performance,Hybrid	1
Flex Fuel,Factory Tuner,Luxury,High-Performance	1

Name: count, dtype: int64

Action: We checked Market Category column for all rows with Null values of Engine HP. All cars are either Exotic or High-Performance or Luxury vechiles. So didn’t change anything.

DATA PREPROCESSING

10. Dealing with Outliers in Engine Cylinders column.




```
df[df['Engine Cylinders']<1]['Engine Fuel Type'].unique()
```

```
array(['electric'], dtype=object)
```

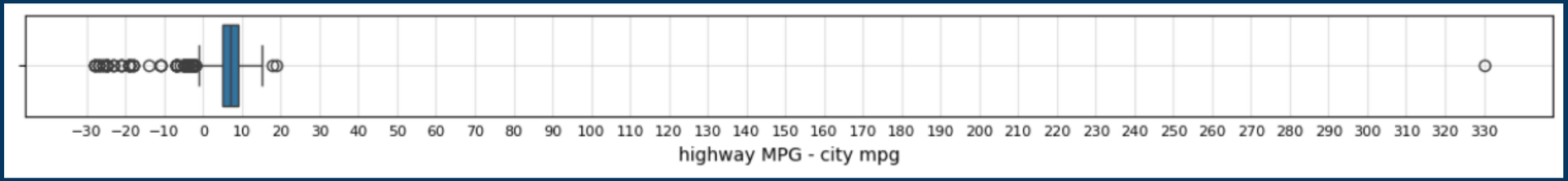
```
df[df['Engine Cylinders']>8]['Market Category'].value_counts()
```

```
Market Category
Exotic,High-Performance                                  141
Exotic,Luxury,Performance                                 25
Luxury,High-Performance                                   25
Exotic,Luxury,High-Performance                            19
Exotic,Factory Tuner,High-Performance                     19
Factory Tuner,Luxury,High-Performance                     13
Exotic,Flex Fuel,Factory Tuner,Luxury,High-Performance    13
Exotic,Flex Fuel,Luxury,High-Performance                  11
Exotic,Luxury                                             10
Luxury,Performance                                        10
Exotic,Factory Tuner,Luxury,High-Performance               5
Luxury                                                     3
Factory Tuner,Luxury,Performance                           1
Crossover,Luxury,Diesel                                    1
Name: count, dtype: int64
```

Action: We checked Engine Fuel Type for rows with 0 Engine Cylinders which are all are electric which is logical. For Engine Cylinders greater than 8, we checked the Market Category and all cars were either Exotic or High-Performance or Luxury. So didn't change anything.

DATA PREPROCESSING

11. Dealing with Outliers in highway MPG and city mpg column.



Action: We can observe that there are few negative value of differences. Generally Mileage in Highway is more than that of City. So we replaced the highway MPG of such rows with the corresponding city mpg.

```
df['highway MPG'].loc[df[d<0].index]=df['city mpg'].loc[df[d<0].index]
```

```
d=df['highway MPG']-df['city mpg']
df[d>15]
```

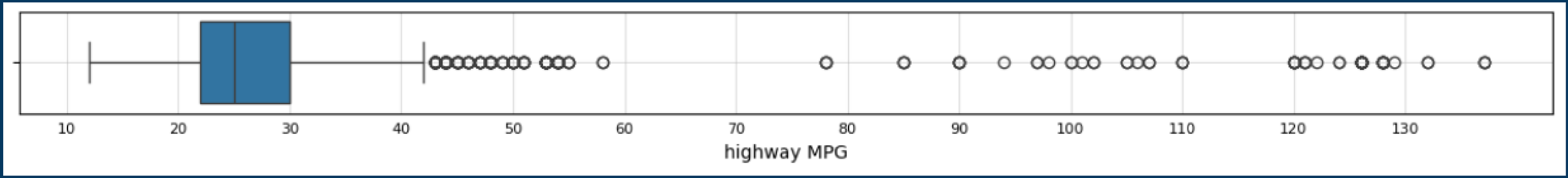
	Make	Model	Year	Engine Fuel Type	Engine HP	Engine Cylinders	Transmission Type	Driven_Wheels	Number of Doors	Market Category	Vehicle Size	Vehicle Style	highway MPG	city mpg	Popu
1119	Audi	A6	2017	premium unleaded (recommended)	252	4	AUTOMATED_MANUAL	front wheel drive	4	Luxury	Midsize	Sedan	354	24	
3157	Chevrolet	Cruze	2015	diesel	148	4	AUTOMATIC	front wheel drive	4	Diesel	Midsize	Sedan	46	27	
4257	Lotus	Evora 400	2017	premium unleaded (required)	400	6	MANUAL	rear wheel drive	2	Exotic,High-Performance	Compact	Coupe	39	21	

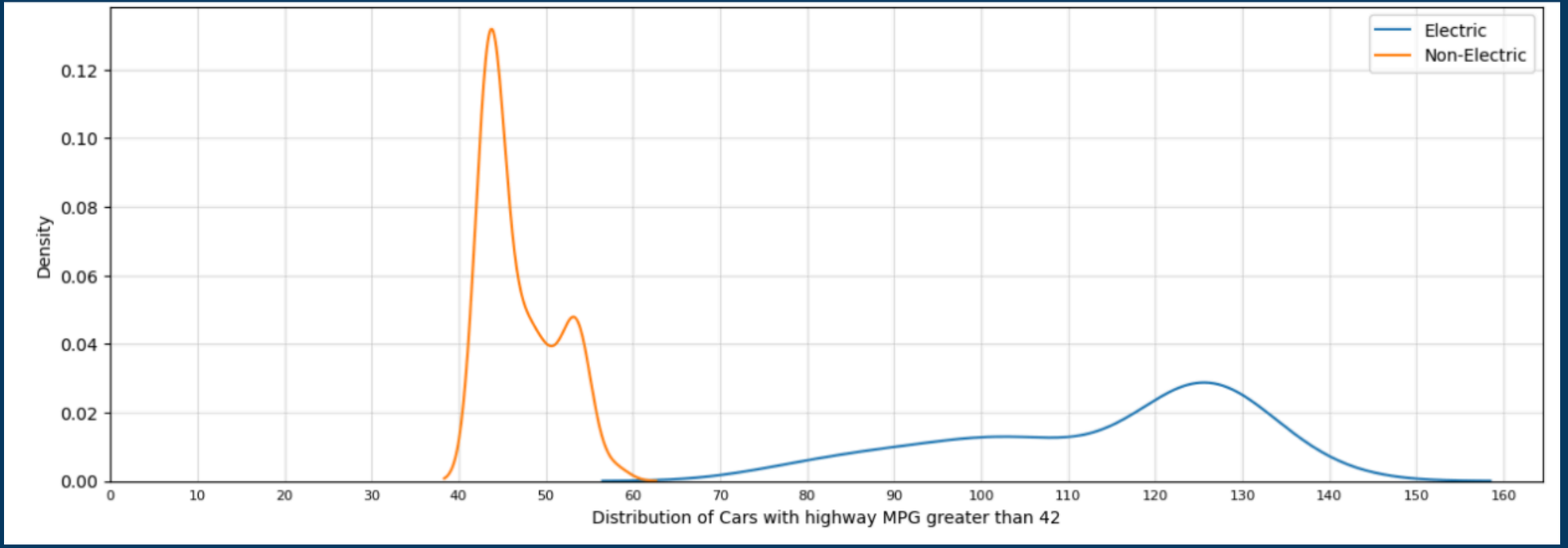
```
#https://www.cars.com/research/audi-a6-2017/
df['highway MPG'].loc[1119]=35

#https://media.chevrolet.com/media/us/en/chevrolet/vehicles/cruze/2015.html
df['highway MPG'].loc[3157]=40

#https://www.fueleconomy.gov/feg/bymodel/2017_Lotus_Evora.shtml
df['highway MPG'].loc[4257]=24
df['city mpg'].loc[4257]=16
```

Action: Also, there are few rows where highway MPG is a lot more than city mpg. On further investigation, we found that these are not correct values. So we searched online for the correct Highway and City mileages and replaced them with the correct values.

DATA PREPROCESSING

12. Dealing with Outliers in highway MPG column.

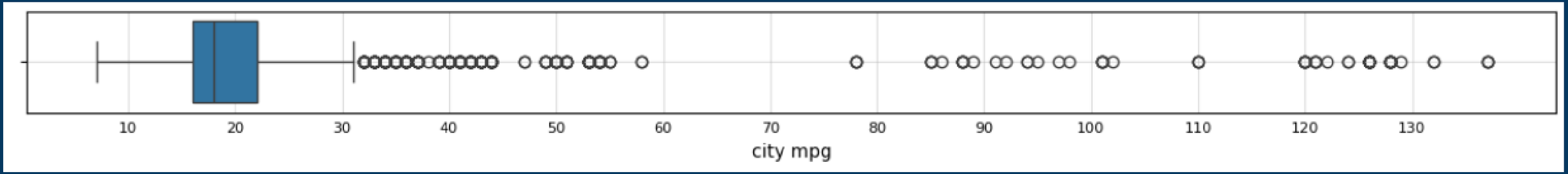


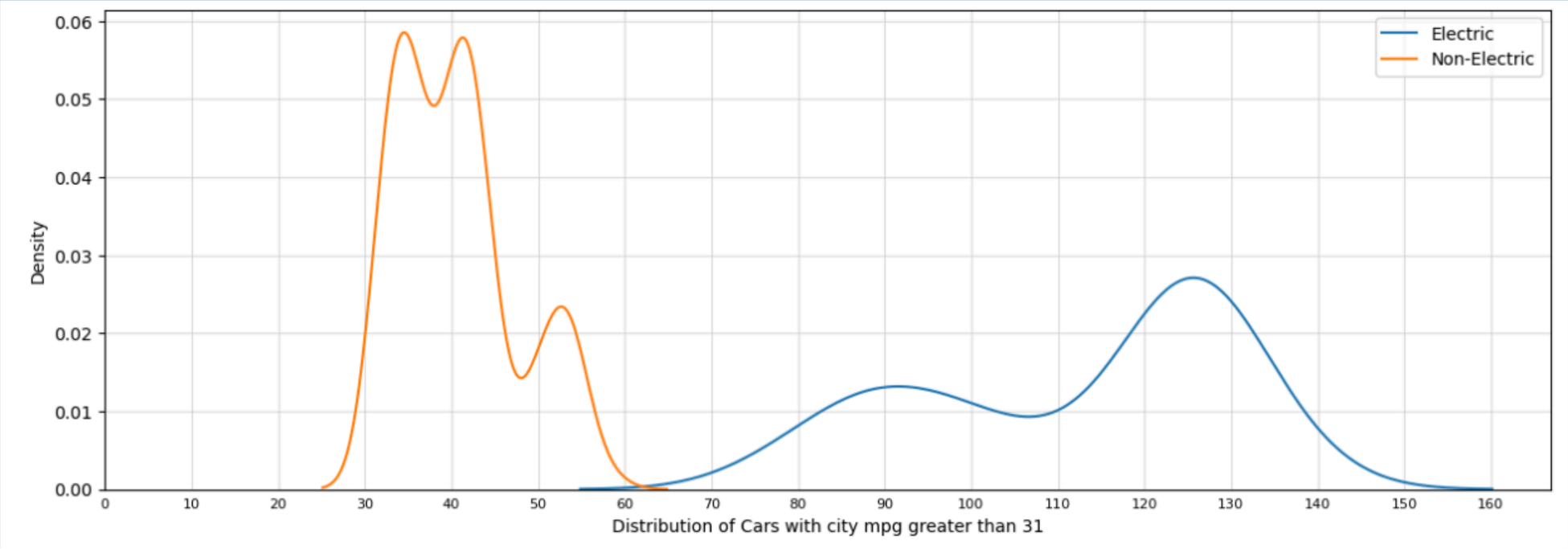


<u>Action</u>:
Considering 42 as the threshold, we can observe that a large percentage of vehicles with very high mileage are electric vehicles which is very logical. So didn't change anything.

DATA PREPROCESSING

13. Dealing with Outliers in city mpg column.



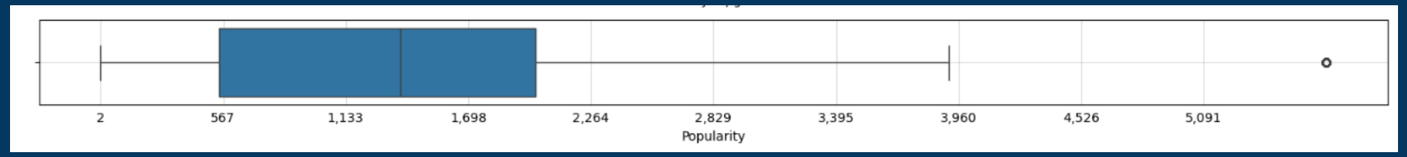


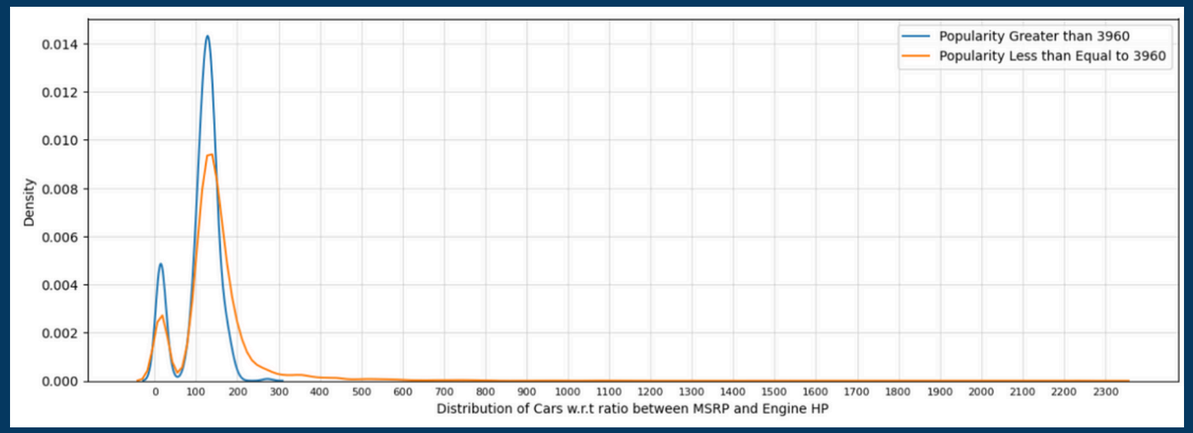
Action:

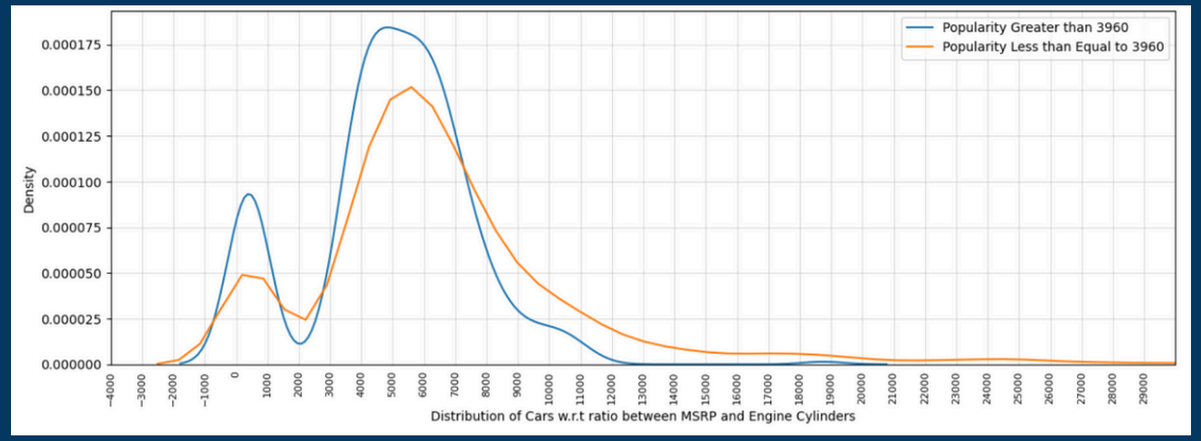
Considering 31 as the threshold, we can observe that a large percentage of vehicles with very high mileage are electric vehicles which is very logical. So didn't change anything.

DATA PREPROCESSING

14. Dealing with Outliers in Popularity column.








```
d=df[df['Popularity']>3960]
```

```
d['Make'].unique()
```

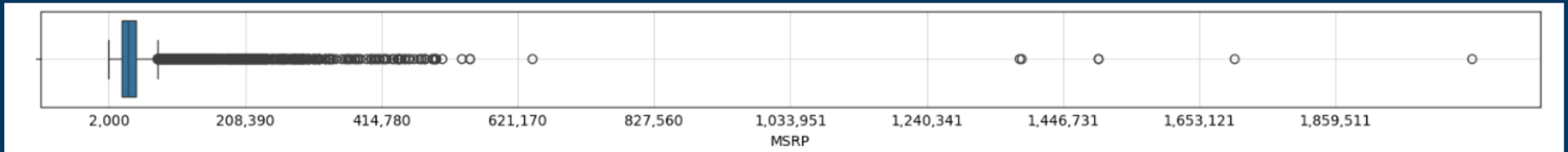
```
array(['Ford'], dtype=object)
```

Action:

- Considering 3960 as the threshold, we can observe that the distribution of ratio of MSRP (Car Price) and Engine HP for cars whose popularity is above and below 3960 is almost the same.
- Also with 3960 as threshold value, we can observe that the distribution of MSRP (Car Price) and Engine Cylinders for cars whose popularity is above and below 3960 is almost the same.
- Also the cars whose popularity is above 3960 are all from Ford which implies that Ford cars are very popular in the region. So didn't change anything.

DATA PREPROCESSING

15. Dealing with Outliers in MSRP column.




```
df[df['MSRP']>100000]['Market Category'].value_counts()
```

```
Market Category
Exotic,High-Performance                                  213
Luxury,High-Performance                                  105
Factory Tuner,Luxury,High-Performance                     69
Exotic,Luxury,High-Performance                            52
Exotic,Factory Tuner,Luxury,High-Performance              48
Exotic,Luxury,Performance                                 36
Exotic,Factory Tuner,High-Performance                     21
Exotic,Flex Fuel,Factory Tuner,Luxury,High-Performance    13
Flex Fuel,Luxury,High-Performance                         13
Exotic,Luxury                                             12
Crossover,Factory Tuner,Luxury,High-Performance           12
Exotic,Flex Fuel,Luxury,High-Performance                  11
High-Performance                                           6
Crossover,Luxury,Performance                               3
Exotic,Factory Tuner,Luxury,Performance                    3
Luxury,High-Performance,Hybrid                             3
Crossover,Luxury,High-Performance                          2
Luxury                                                     2
Factory Tuner,Luxury                                       2
Factory Tuner,High-Performance                             2
Luxury,Performance                                         2
Exotic,Luxury,High-Performance,Hybrid                      1
Flex Fuel,Factory Tuner,Luxury,High-Performance            1
Name: count, dtype: int64
```

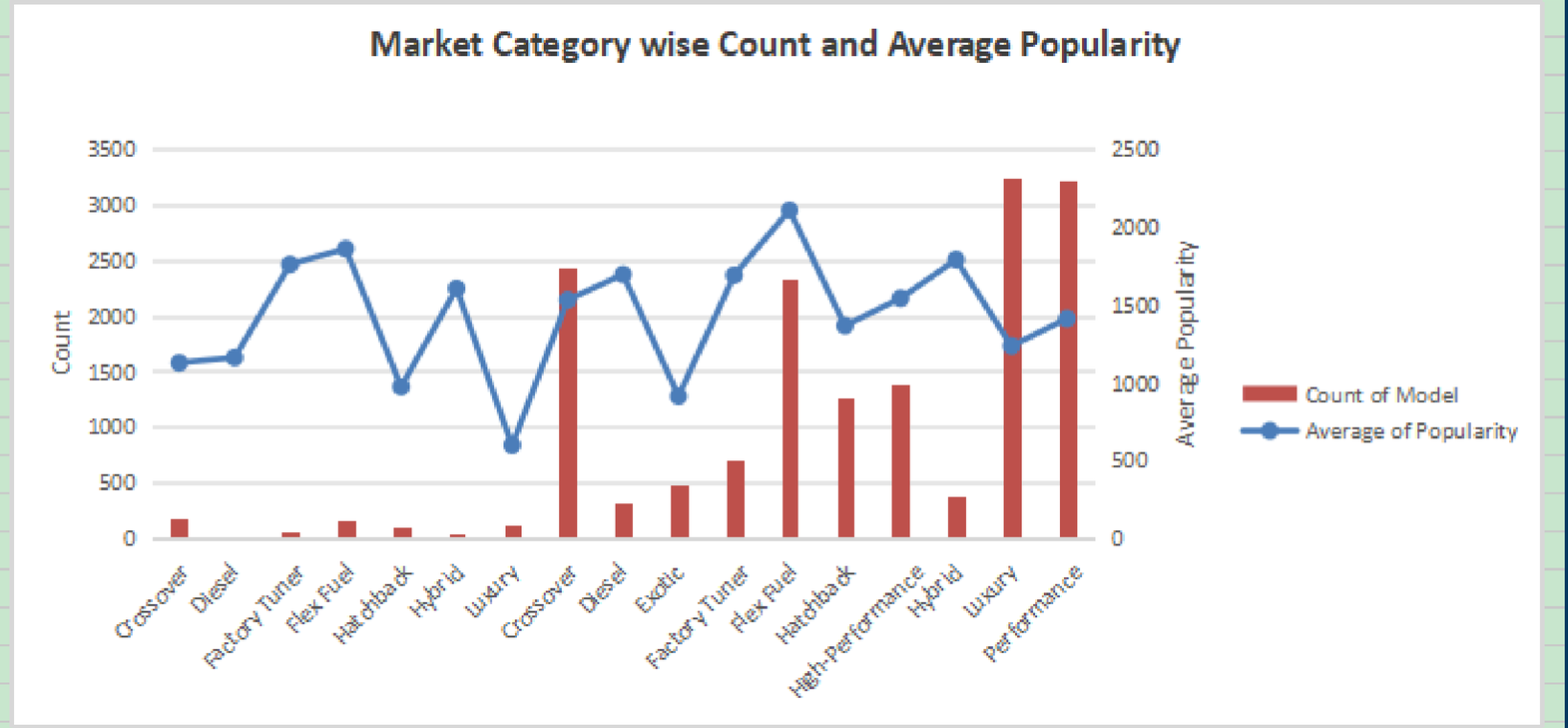
Action: Considering 100000 as the threshold, we can observe that the cars with price above 100000 are all Exotic or Performance or Luxury cars. So didn't change anything.

ANALYSIS

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

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

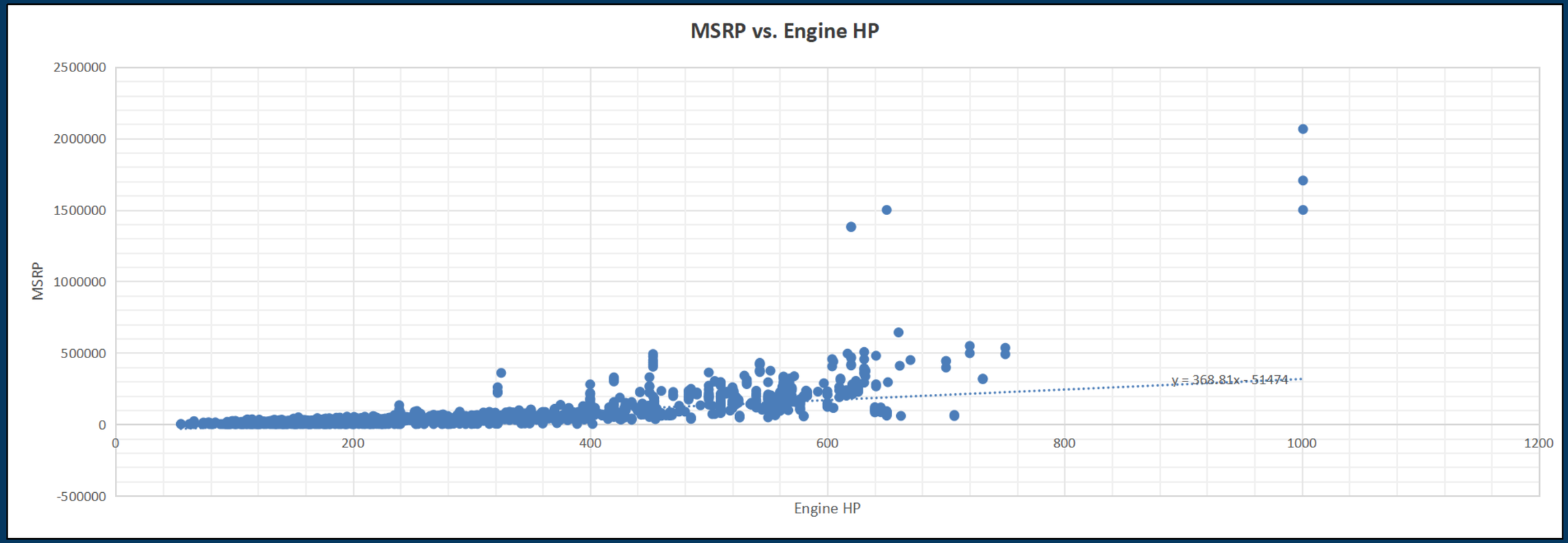
Mkt_Cat	Average of Popularity	Count of Model
Crossover	1126.278409	176
Diesel	1159.666667	6
Factory Tuner	1758.796296	54
Flex Fuel	1858.455696	158
Hatchback	972.7981651	109
Hybrid	1604.272727	44
Luxury	597.304	125
Crossover	1530.461255	2439
Diesel	1694.823708	329
Exotic	912.4195519	491
Factory Tuner	1689.745429	711
Flex Fuel	2106.363559	2338
Hatchback	1364.802362	1270
High-Performance	1541.116162	1386
Hybrid	1789.226316	380
Luxury	1233.849907	3238
Performance	1410.148033	3229
Grand Total	**1505.024753**	**16483**



- The Factory Tuner category, despite having fewer models, has the highest average popularity, indicating a strong consumer preference for these models.
- Performance and Crossover categories have the highest number of models, but their popularity is more evenly distributed, leading to moderate average popularity scores.

ANALYSIS

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.



- There is a positive correlation between engine power and price—higher engine power generally leads to higher prices.
- The relationship is moderate, with some outliers where price does not increase proportionally with engine power.

ANALYSIS

Task 3: Use regression analysis to identify the variables that have the strongest relationship with a car's price. Then create a bar chart that shows the coefficient values for each variable to visualize their relative importance.

Final Regression Analysis

SUMMARY OUTPUT

Regression Statistics	
Multiple R	0.836939232
R Square	0.700467278
Adjusted R Square	0.700038685
Standard Error	33701.96062
Observations	11199

ANOVA

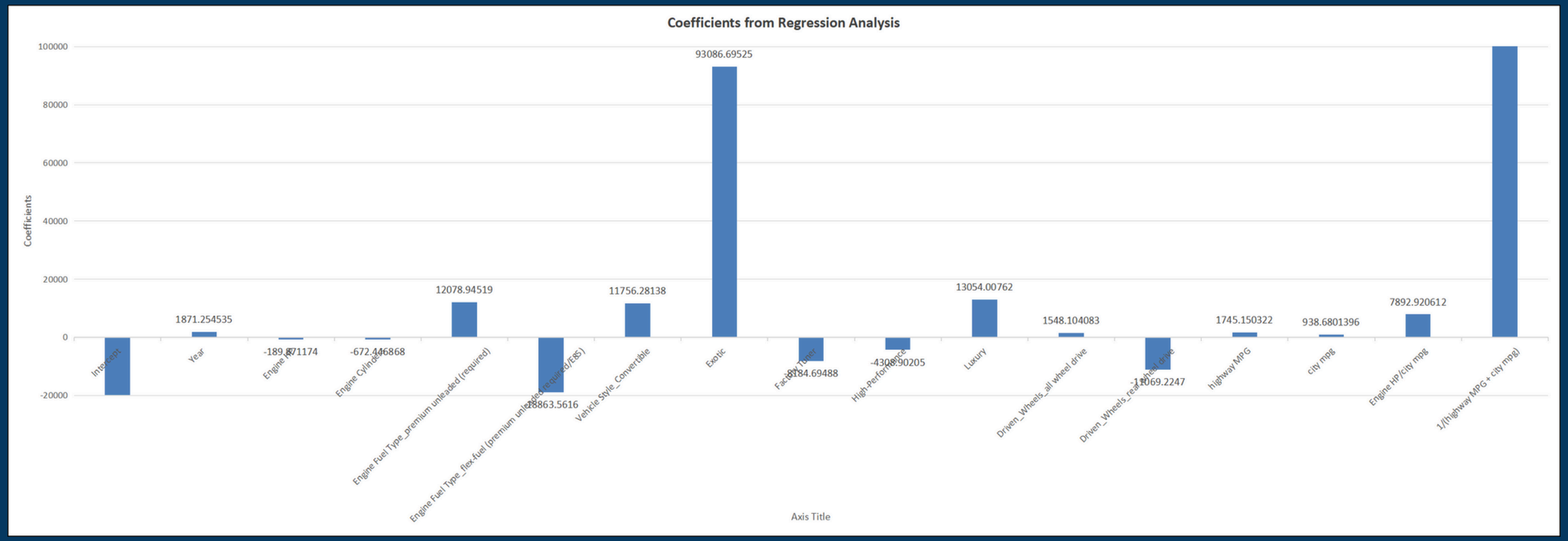
	df	*SS*	*MS*	*F*	*Significance F*
Regression	16	2.97012E+13	1.85632E+12	1634.342536	0
Residual	11182	1.27008E+13	1135822150		
Total	11198	4.24019E+13			

	Coefficients	*Standard Error*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*	*Lower 95.0%*	*Upper 95.0%*
Intercept	-3988458.386	145137.6152	-27.4805286	6.1866E-161	-4272953.679	-3703963.093	-4272953.679	-3703963.093
Year	1871.254535	69.62081994	26.87780088	2.9785E-154	1734.785464	2007.723606	1734.785464	2007.723606
Engine HP	-189.8711738	9.712740602	-19.54867133	1.04842E-83	-208.9098563	-170.8324912	-208.9098563	-170.8324912
Engine Cylinders	-672.4468678	427.7919571	-1.571901614	0.116001663	-1510.994463	166.1007271	-1510.994463	166.1007271
Engine Fuel Type_premium unleaded (required)	12078.94519	1138.046179	10.61375664	3.42634E-26	9848.174203	14309.71618	9848.174203	14309.71618
Engine Fuel Type_flex-fuel (premium unleaded required/E85)	-18863.56155	4866.117474	-3.876511747	0.000106571	-28402.00901	-9325.114098	-28402.00901	-9325.114098
Vehicle Style_Convertible	11756.28138	1360.151319	8.643362851	6.20069E-18	9090.145194	14422.41757	9090.145194	14422.41757
Exotic	93086.69525	2138.261713	43.53381753	0	88895.32562	97278.06488	88895.32562	97278.06488
Factory Tuner	-8184.694881	1420.939036	-5.760060546	8.63008E-09	-10969.9857	-5399.40406	-10969.9857	-5399.40406
High-Performance	-4308.902055	1467.806607	-2.935606118	0.003335744	-7186.06157	-1431.74254	-7186.06157	-1431.74254
Luxury	13054.00762	834.3417307	15.64587643	1.34197E-54	11418.55085	14689.46439	11418.55085	14689.46439
Driven_Wheels_all wheel drive	1548.104083	940.5028344	1.646038721	0.099783891	-295.4471498	3391.655315	-295.4471498	3391.655315
Driven_Wheels_rear wheel drive	-11069.22472	862.209357	-12.83820992	1.84113E-37	-12759.30694	-9379.142492	-12759.30694	-9379.142492
highway MPG	1745.150322	154.1490377	11.32118856	1.49236E-29	1442.991054	2047.309591	1442.991054	2047.309591
city mpg	938.6801396	164.5406492	5.704852536	1.194E-08	616.1514821	1261.208797	616.1514821	1261.208797
Engine HP/city mpg	7892.920612	136.7549606	57.71579016	0	7624.856798	8160.984425	7624.856798	8160.984425
1/(highway MPG + city mpg)	2142054.351	101698.2025	21.0628536	1.29369E-96	1942707.959	2341400.743	1942707.959	2341400.743

- Using regression analysis, we found the top columns. This also include two new columns which were Feature Engineered **(Engine HP/city mpg)** and **(1/(highway MPG + city mpg))**.
- We can observe that the R-Squared score is 0.7 which can be counted as a good score.

ANALYSIS

Task 3: Use regression analysis to identify the variables that have the strongest relationship with a car's price. Then create a bar chart that shows the coefficient values for each variable to visualize their relative importance.



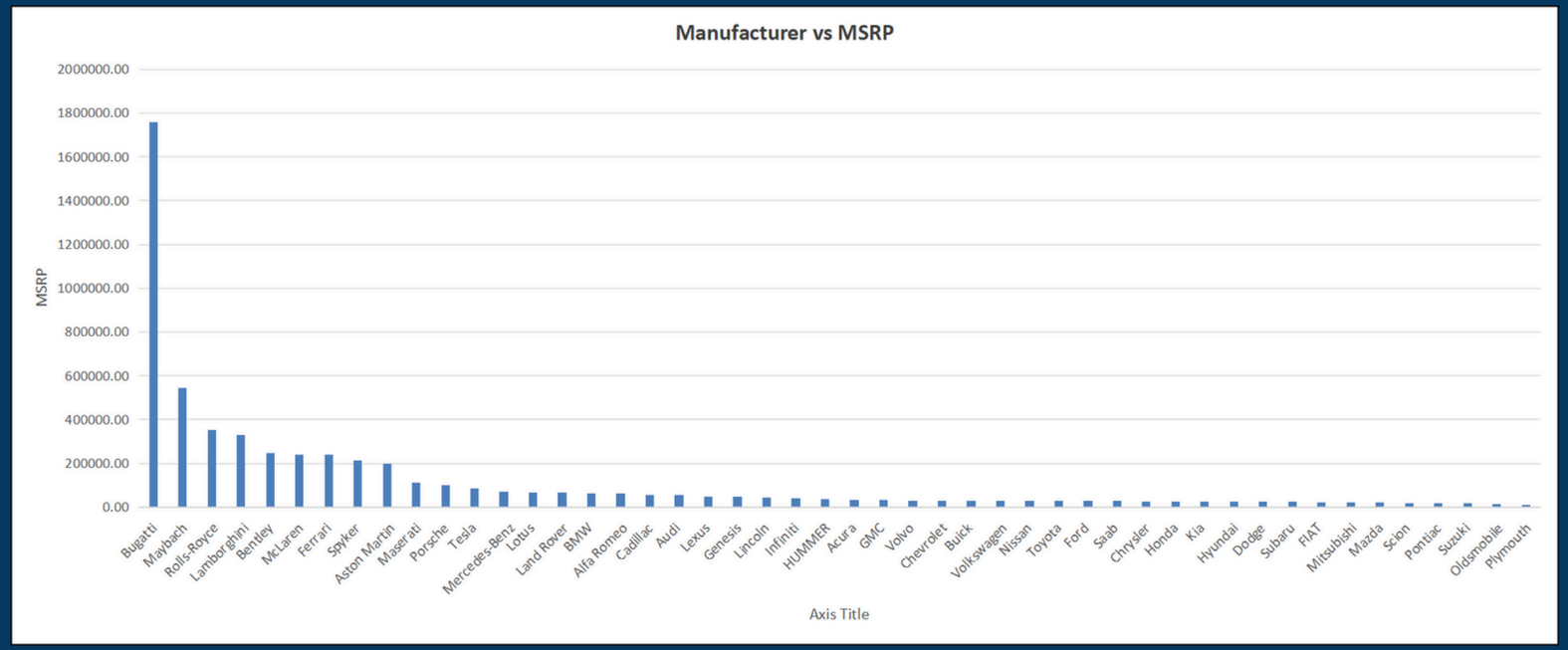
- We can observe that the highest coefficient value is that of Engineered Feature, **1/(highway MPG + city mpg)**.
- This shows that the Engineered Feature is very important relationship with Car's price.

ANALYSIS

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

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

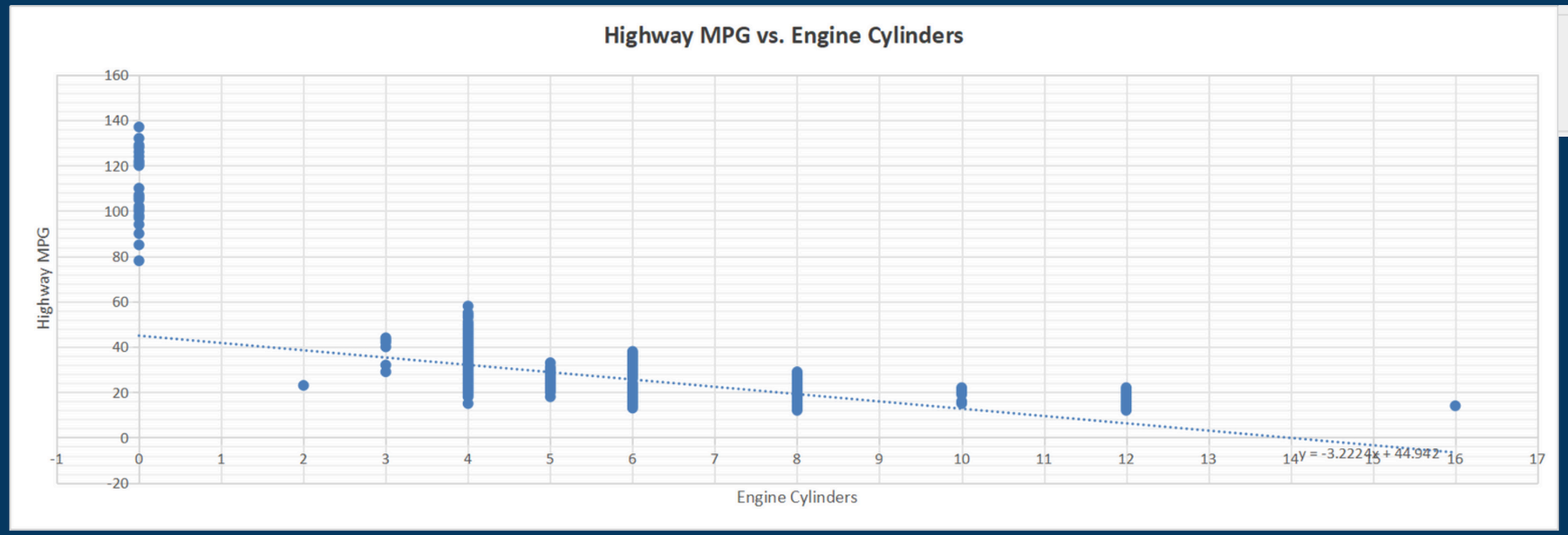
Make	Average of MSRP
Bugatti	1757223.67
Maybach	546221.88
Rolls-Royce	351130.65
Lamborghini	331567.31
Bentley	247169.32
McLaren	239805.00
Ferrari	238218.84
Spyker	214990.00
Aston Martin	198123.46
Maserati	113684.49
Porsche	101622.40
Tesla	85255.56
Mercedes-Benz	72069.53
Lotus	68377.14
Land Rover	68067.09
BMW	62162.56
Alfa Romeo	61600.00
Cadillac	56368.27
Audi	54574.12
Lexus	47549.07
Genesis	46616.67
Lincoln	43860.83
Infiniti	42640.27
HUMMER	36464.41
Acura	35087.49
GMC	32444.09
Volvo	29724.68
Chevrolet	29074.73
Buick	29034.19
Volkswagen	28978.52
Nissan	28921.15
Toyota	28846.56
Ford	28511.31
Saab	27879.81
Chrysler	26722.96
Honda	26655.15
Kia	25513.76
Hyundai	24926.26
Dodge	24857.05
Subaru	24240.67
FIAT	22670.24
Mitsubishi	21340.56
Mazda	20416.62
Scion	19932.50
Pontiac	19800.04
Suzuki	18026.42
Oldsmobile	12843.80
Plymouth	3296.87
Grand Total	**41925.93**



- Luxury Brands like Bugatti, Maybach, and Rolls-Royce have the highest average prices, indicating they focus on high-end, premium vehicles.
- Mass-Market Brands like Honda, Ford, and Toyota have significantly lower average prices, suggesting they cater to a broader consumer base with more affordable vehicles.

ANALYSIS

Task 5.A: Create a scatter plot with the number of cylinders on the x-axis and highway MPG on the y-axis. Then create a trendline on the scatter plot to visually estimate the slope of the relationship and assess its significance.
Task 5.B: Calculate the correlation coefficient between the number of cylinders and highway MPG to quantify the strength and direction of the relationship.

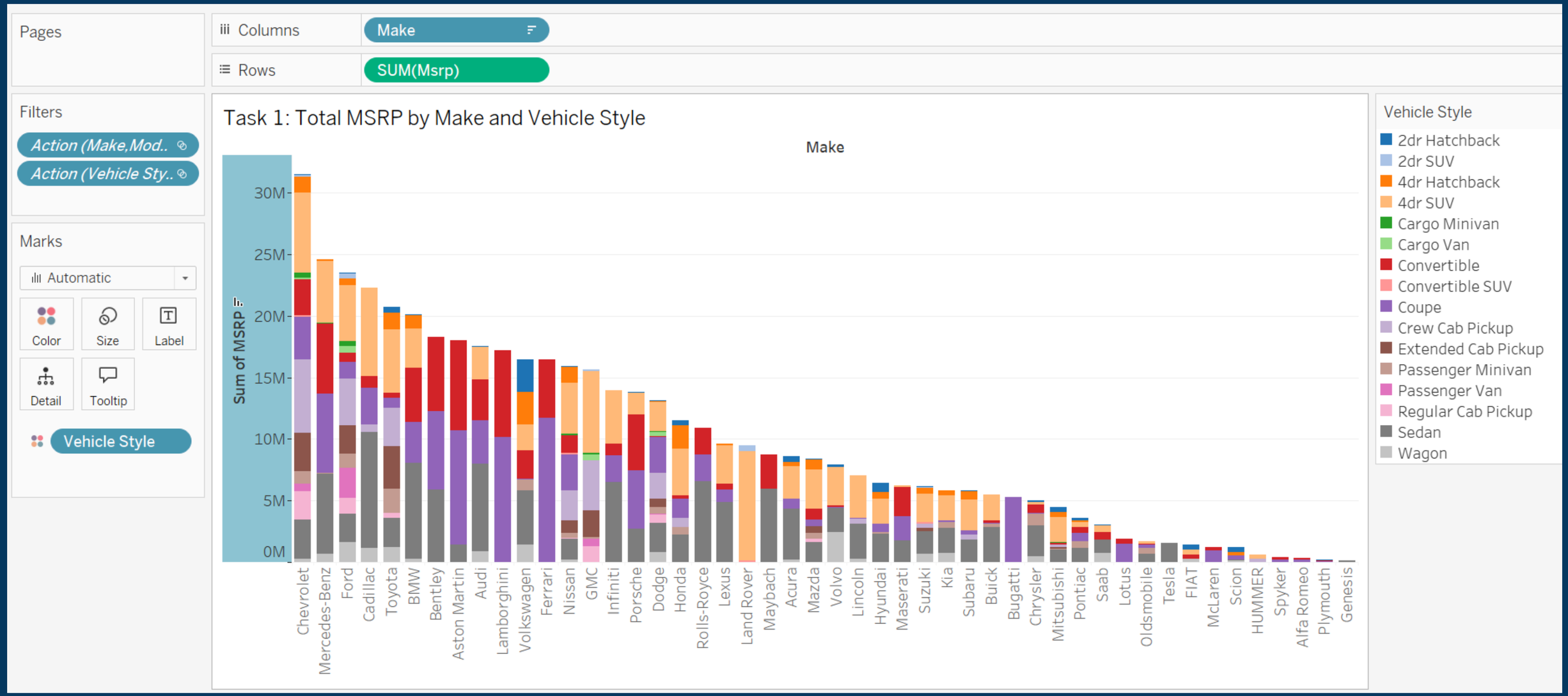


Correlation between **Engine Cylinders** and **highway MPG:**	-0.62293632

- We can observe that the plot between highway MPG and Engine Cylinders has a negative slope with a value of -3.2224.
- The correlation coefficient is also Negative with a value of -0.62294.
- This is logical because as number of Engine Cylinders increases, the amount of fuel to be burnt also increasing, thus decreasing the mileage (highway MPG)

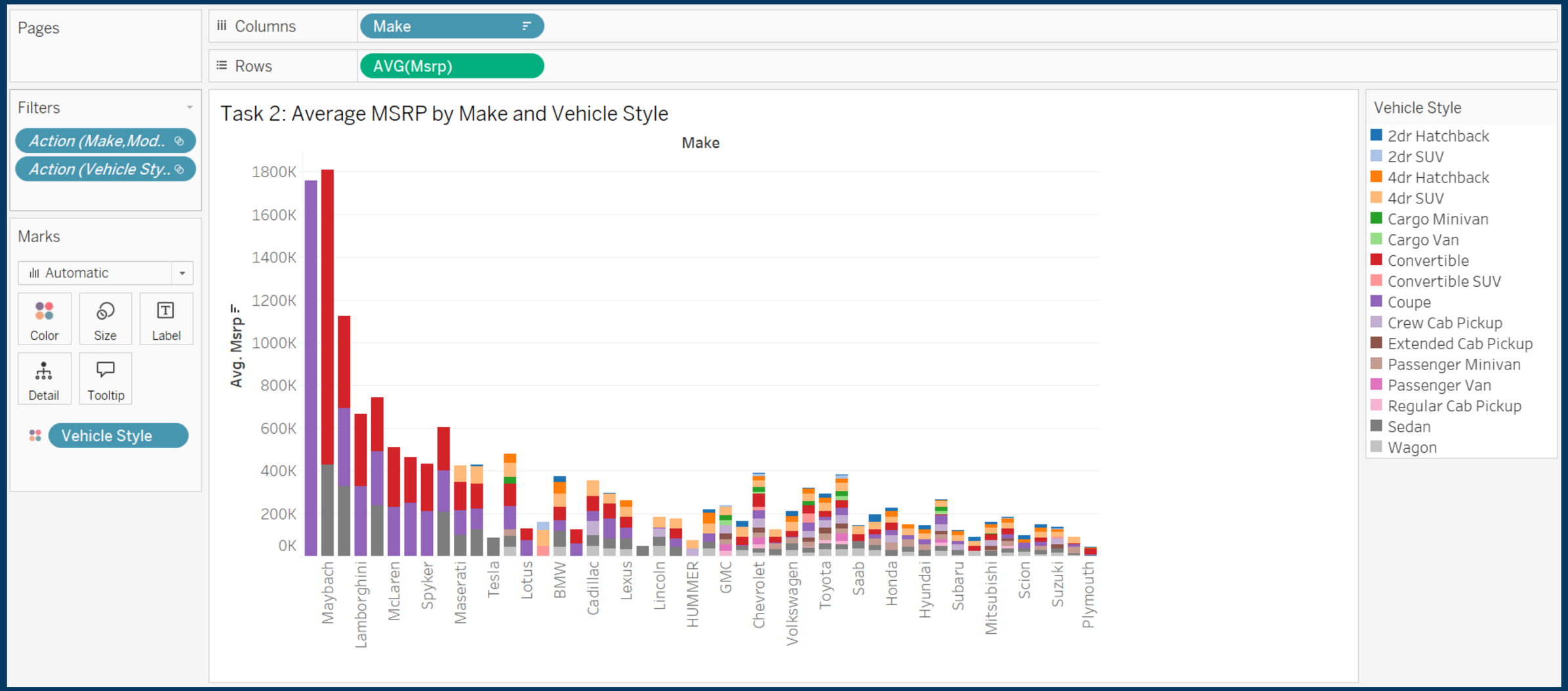
DASHBOARD

Task 1: How does the distribution of car prices vary by brand and body style?



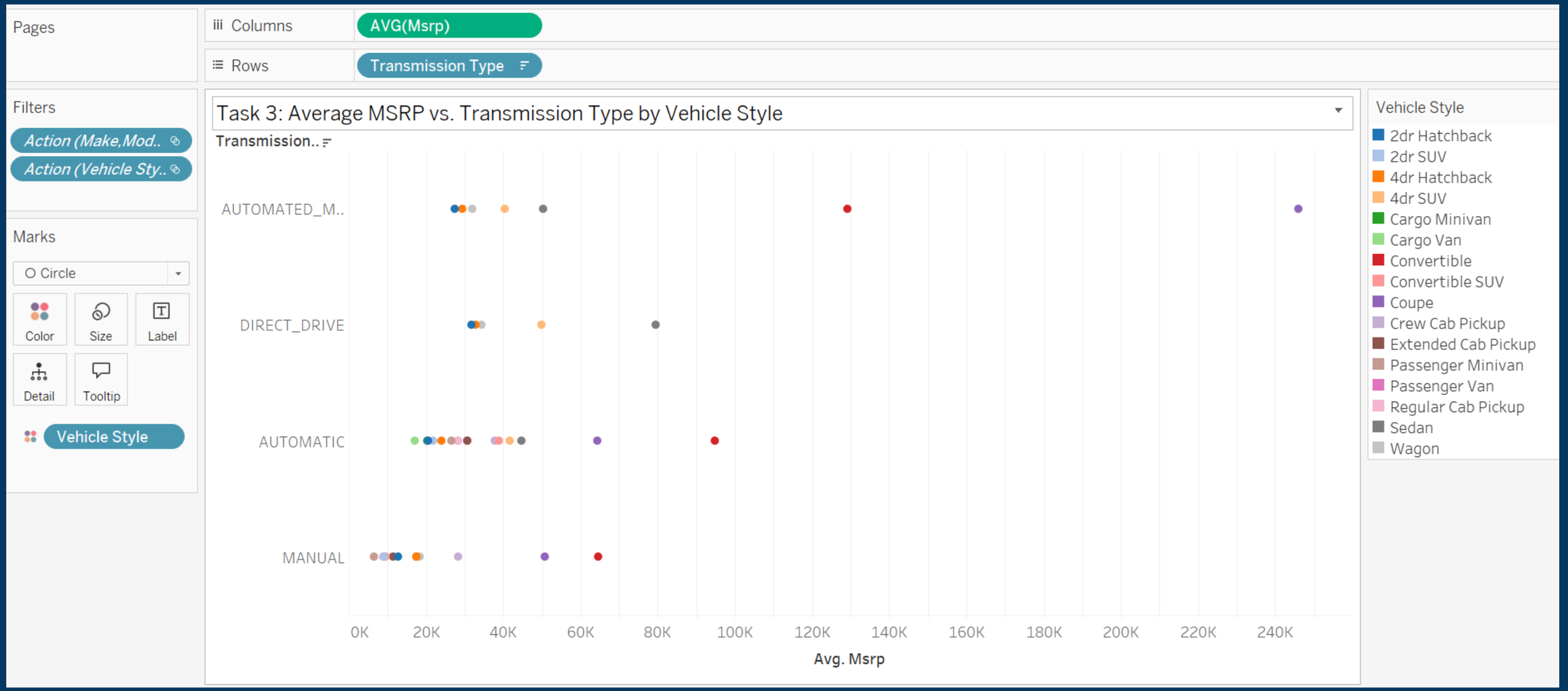
DASHBOARD

Task 2: Which car brands have the highest and lowest average MSRPs, and how does this vary by body style?



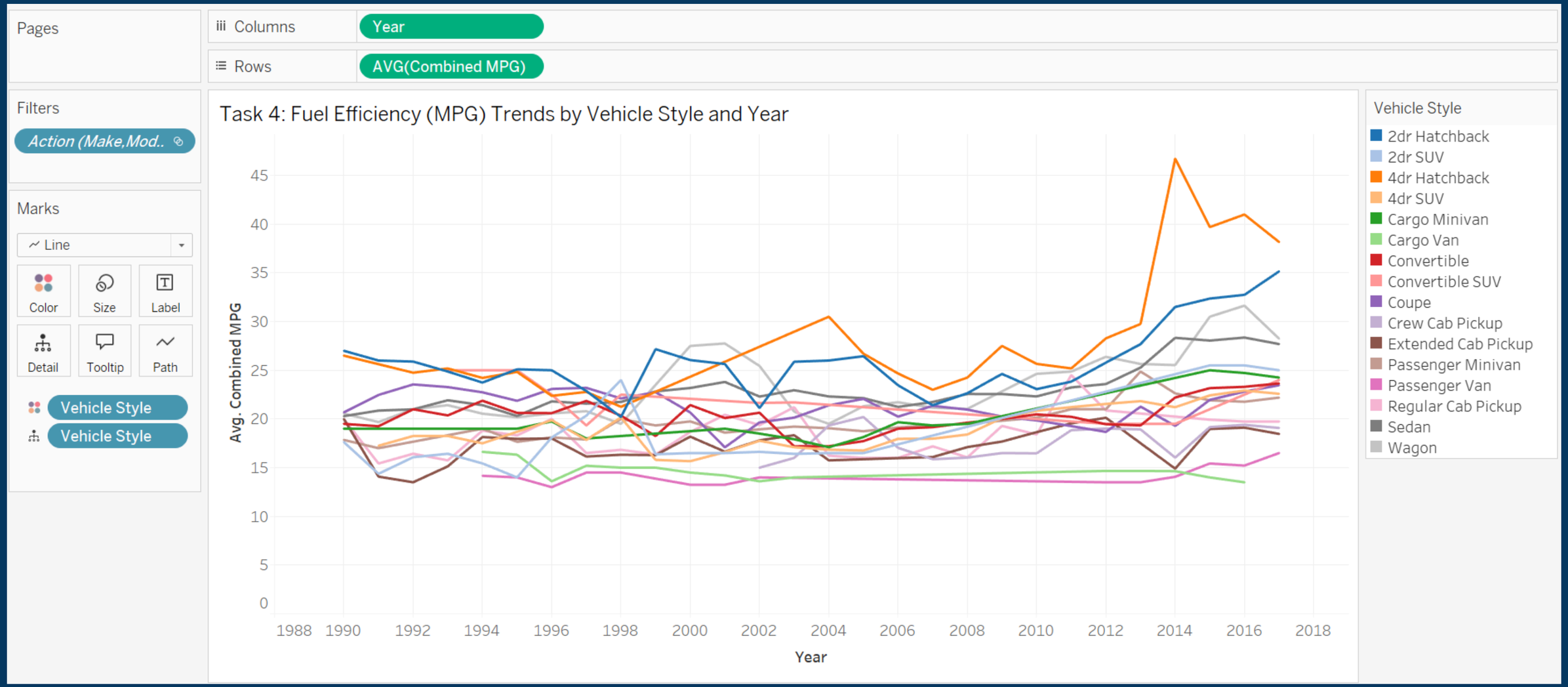
DASHBOARD

Task 3: How do the different feature such as transmission type affect the MSRP, and how does this vary by body style?



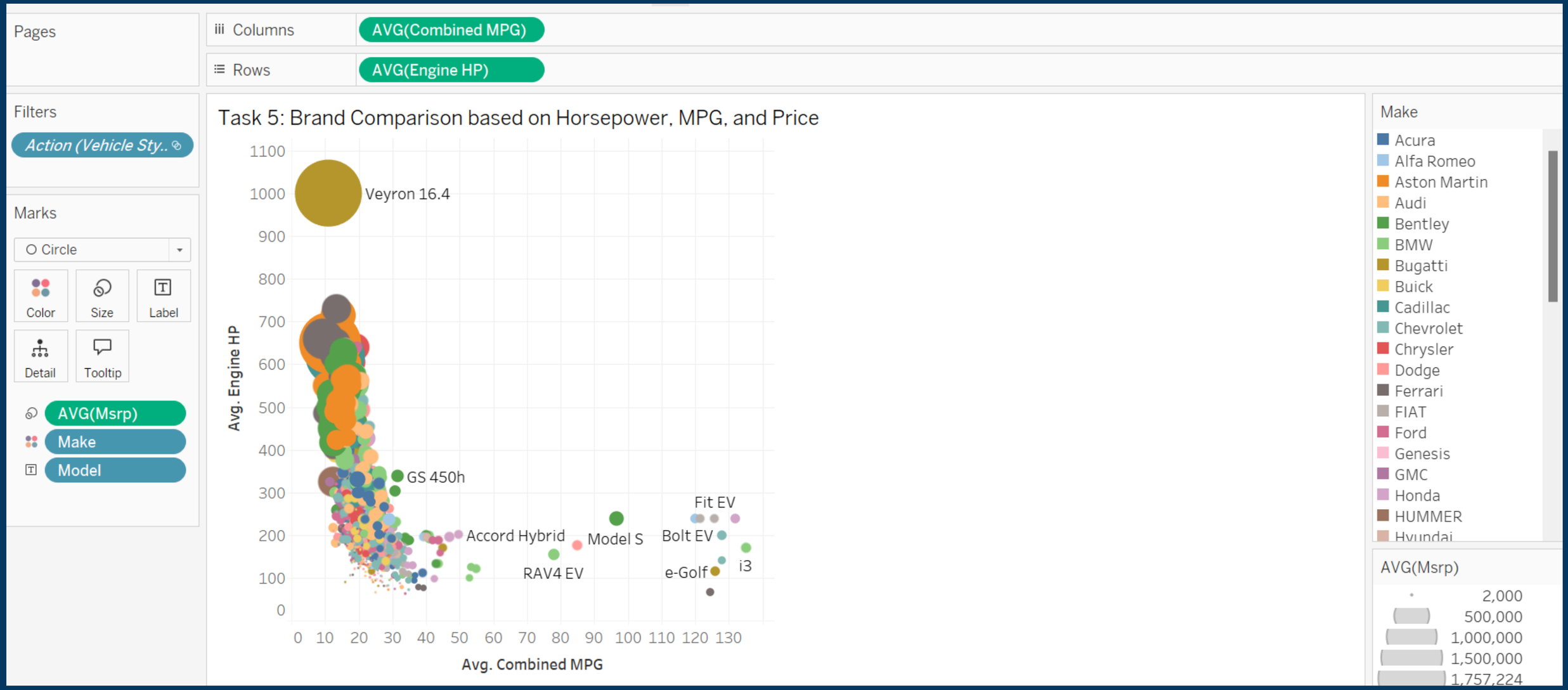
DASHBOARD

Task 4: How does the fuel efficiency of cars vary across different body styles and model years?



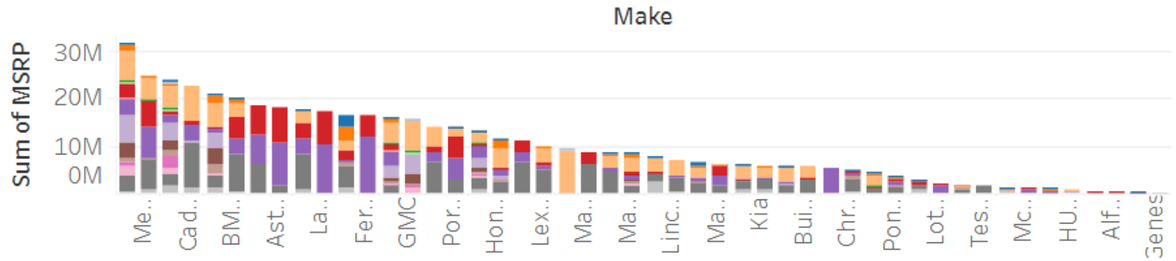
DASHBOARD

Task 5: How does the car's horsepower, MPG, and price vary across different Brands?

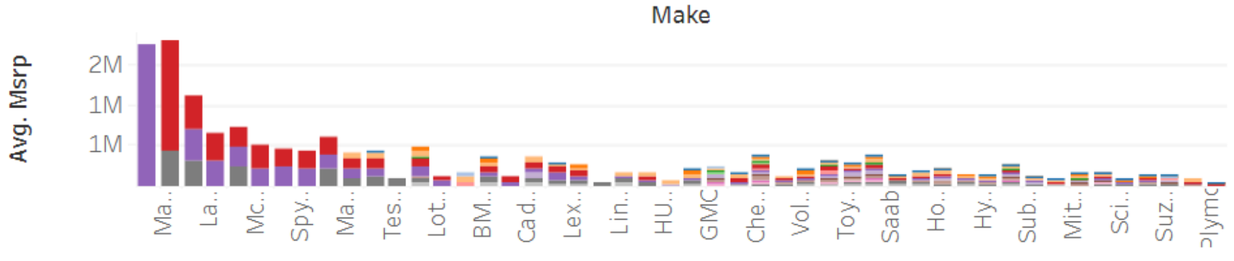


COMPLETE DASHBOARD

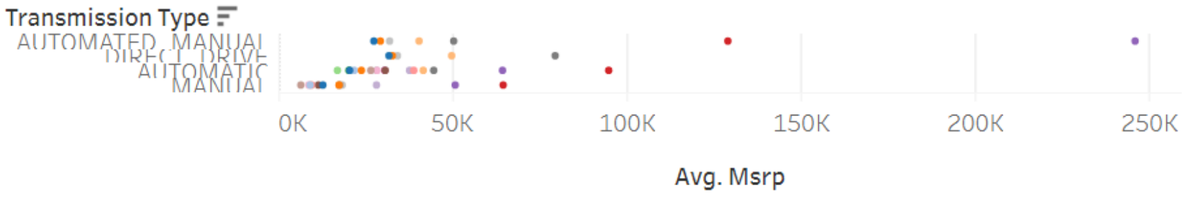
Task 1: Total MSRP by Make and Vehicle Style



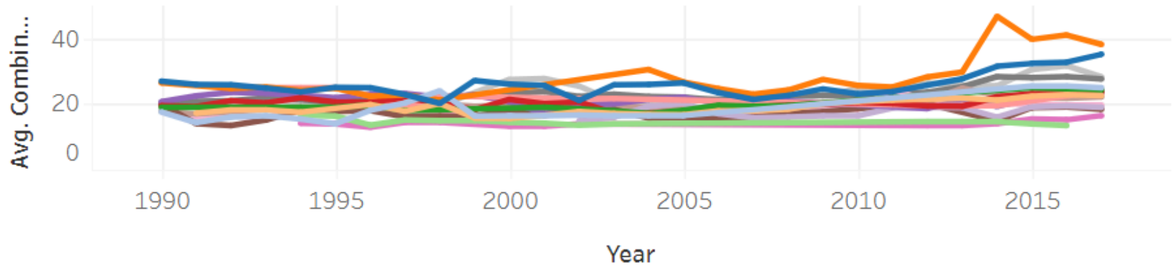
Task 2: Average MSRP by Make and Vehicle Style



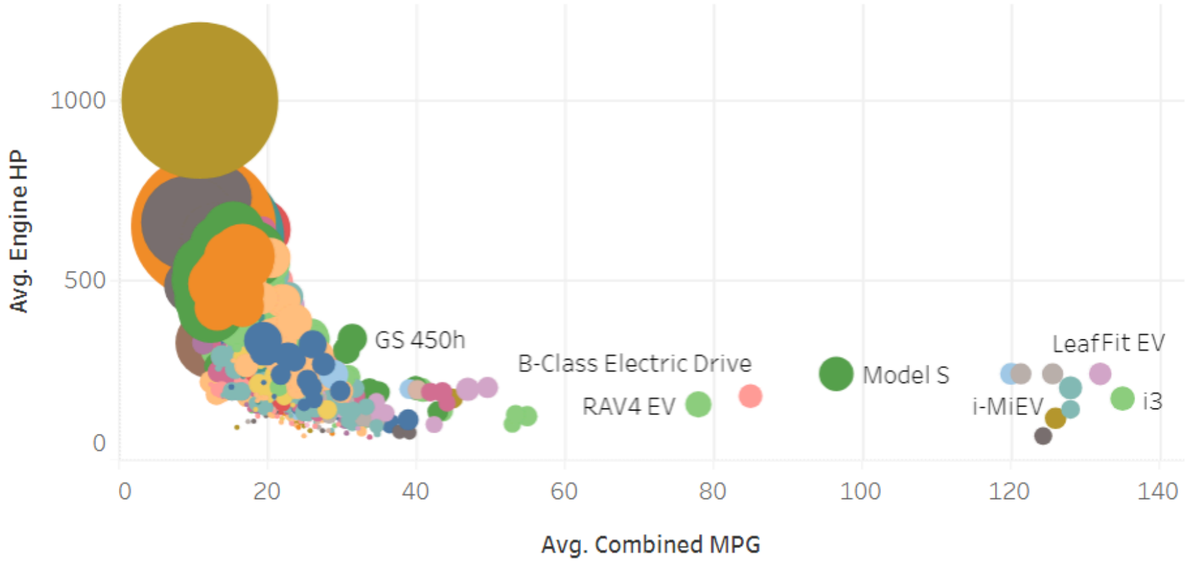
Task 3: Average MSRP vs. Transmission Type by Vehicle Style



Task 4: Fuel Efficiency (MPG) Trends by Vehicle Style and Year



Task 5: Brand Comparison based on Horsepower, MPG, and Price



RESULTS

VISUALIZATION

Results presented through dashboards, scatter plots, and pivot tables (e.g., price variation, key features' impact on price).

DISCUSSION

Engine power, fuel type, and market category significantly influence pricing; manufacturers' pricing strategies vary widely.

LIMITATIONS

Some features may have been underrepresented due to data gaps; limited insights into certain market categories.

FUTURE DIRECTIONS

Deeper analysis into additional features, market trends, or integration of external data for a broader market outlook.

LINKS

Hyperlink to the Excel Sheet: Final Sheet

Hyperlink to the Notebook: Notebook

Hyperlink to the Tableau Dashboard: Dashboard