ICE 9 - Using Color and Size in Visualization

In [395]:
```
import matplotlib.pyplot as plt
import plotly.express as px
import pandas as pd
import numpy as np
import warnings
import seaborn as sns
import altair as alt

warnings.filterwarnings('ignore')
```

1. Encoding Data using Color and Size (25 points)

In [393]:
```
# Load the data
carsales = pd.read_csv('https://gist.githubusercontent.com/nehabaddam/1f47243b
```

1.1) Please show part of your dataset (use python), submit the screenshot of the data, and describe your data including its different attributes/ columns.

In [394]:
```
print(carsales.shape)
carsales.head()
```

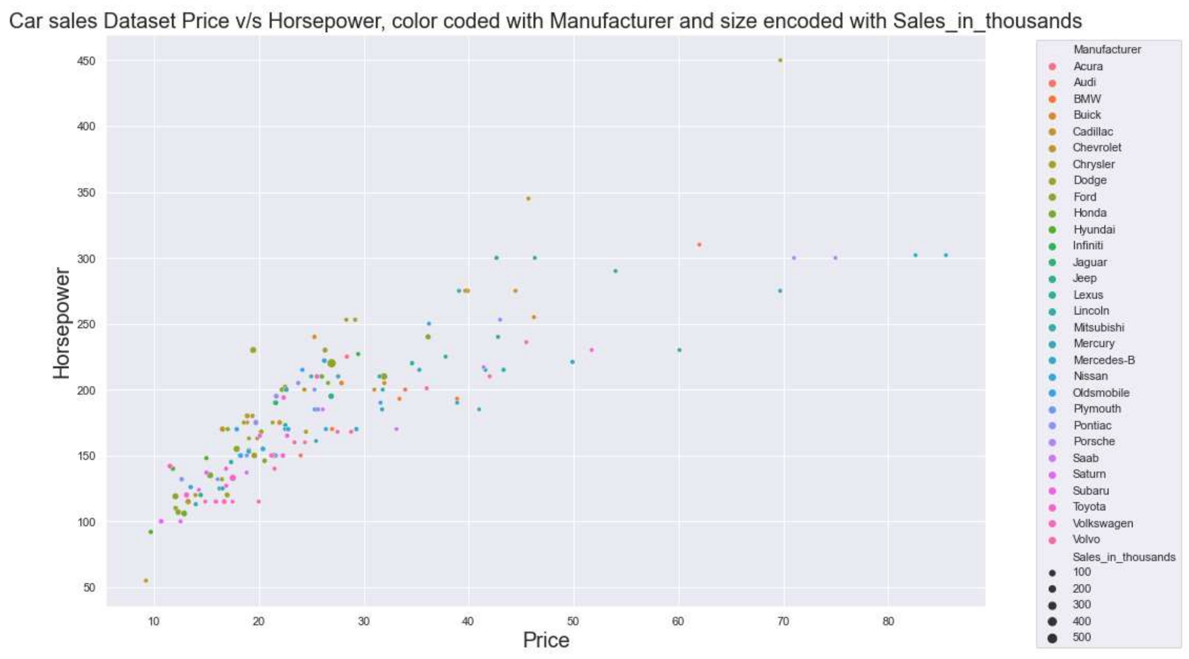
```
(157, 16)
```

Out[394]:

	Manufacturer	Model	Sales_in_thousands	__year_resale_value	Vehicle_type	Price_in_thousa
0	Acura	Integra	16.919	16.360	Passenger	21
1	Acura	TL	39.384	19.875	Passenger	28
2	Acura	CL	14.114	18.225	Passenger	N
3	Acura	RL	8.588	29.725	Passenger	42
4	Audi	A4	20.397	22.255	Passenger	23

1.2) Encoding the data with x-y channels, add both color and size to your graph, different color and size should represent different attributes of the data. Submit a screenshot of the graph and a screenshot of your code (commented properly).

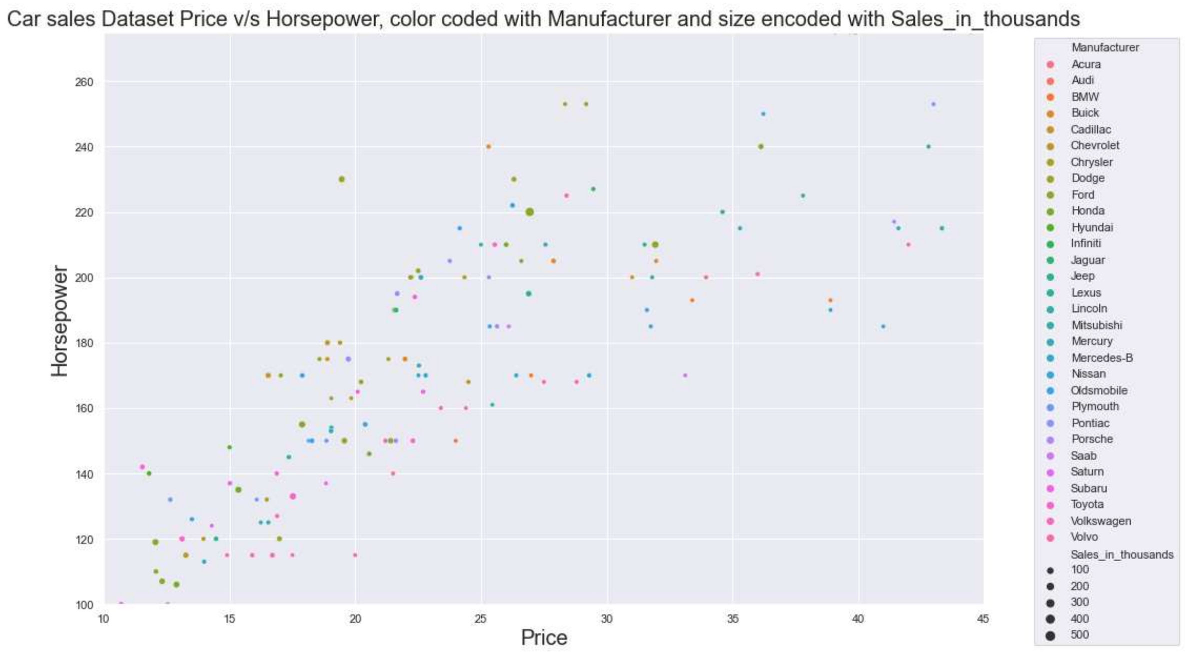
In [400]:
```
# create scatter plot

# set the figure size
plt.figure(figsize=(15, 10))

# Create a scatter plot with colors and size
sns.scatterplot(data=carsales, x='Price_in_thousands', y='Horsepower', hue='Ma

# Set the legend
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')

# set x and y axis labels
plt.xlabel('Price',  fontsize=20)
plt.ylabel('Horsepower',  fontsize=20)

# set plot title
plt.title('Car sales Dataset Price v/s Horsepower, color coded with Manufactur
          fontsize=20)

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



1.3) Try to Optimize your graph and explain why and how you optimize it. Provide a screenshot of your

optimized graph and the code for optimization

In [401]:
```python
# create a optimized scatter plot

# set the figure size
plt.figure(figsize=(15, 10))

# Create a scatter plot with colors and size
sns.scatterplot(data=carsales, x='Price_in_thousands', y='Horsepower', hue='Ma

# Set the legend
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')


# set axis limits
plt.xlim(10, 45)
plt.ylim(100, 275)

# set x and y axis labels
plt.xlabel('Price',  fontsize=20)
plt.ylabel('Horsepower',  fontsize=20)

# set plot title
plt.title('Car sales Dataset Price v/s Horsepower, color coded with Manufactur
          fontsize=20)

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

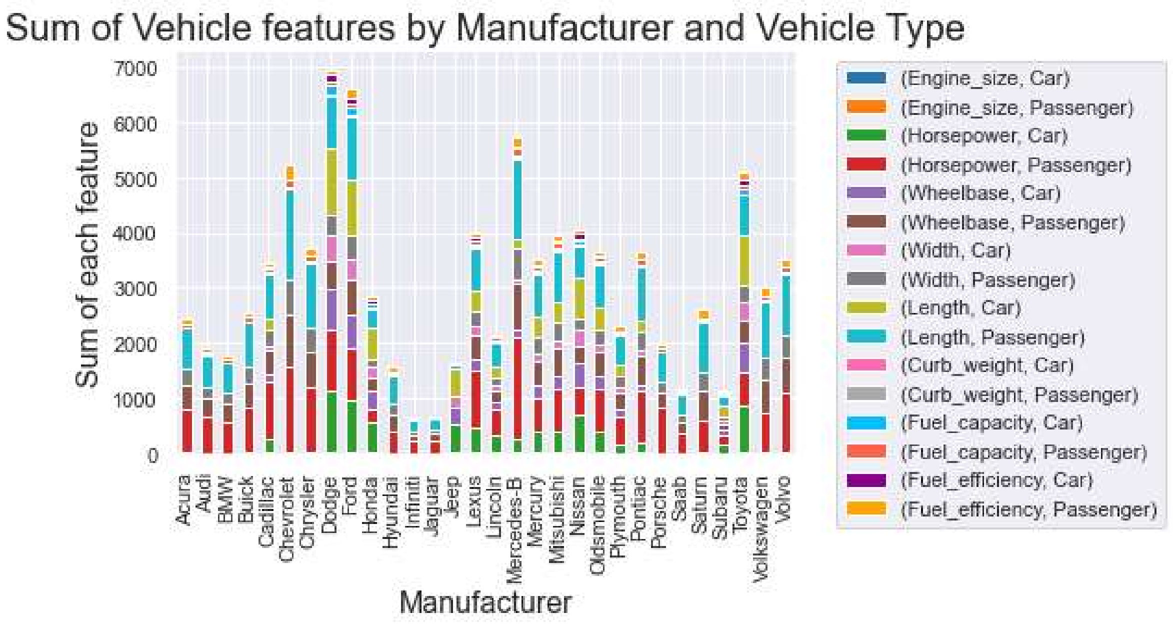


In [ ]:

2. Stacked & Grouped Bar Chart (15 points)

2.1) Create a stacked & Grouped Bar Chart for your data. Submit a screenshot of the graph and a screenshot of your code (commented properly).

In [402]:
```
# Group the data by Manufacturer and Vehicle_type and get the sum of Sales_in_
grouped_stack = carsales.groupby(['Manufacturer', 'Vehicle_type'])['Engine_siz

# Define a custom color palette
mycolors = ['#1f77b4', '#ff7f0e', '#2ca02c', '#d62728', '#9467bd', '#8c564b',

# Create a stacked and grouped bar chart with the custom color palette
grouped_stack.plot(kind='bar', stacked=True, color=mycolors)

# Set the legend
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')

# Set the x and y axis labels
plt.xlabel('Manufacturer', fontsize=16)
plt.ylabel('Sum of each feature', fontsize=16)

# Set the plot title
plt.title('Sum of Vehicle features by Manufacturer and Vehicle Type', fontsize

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

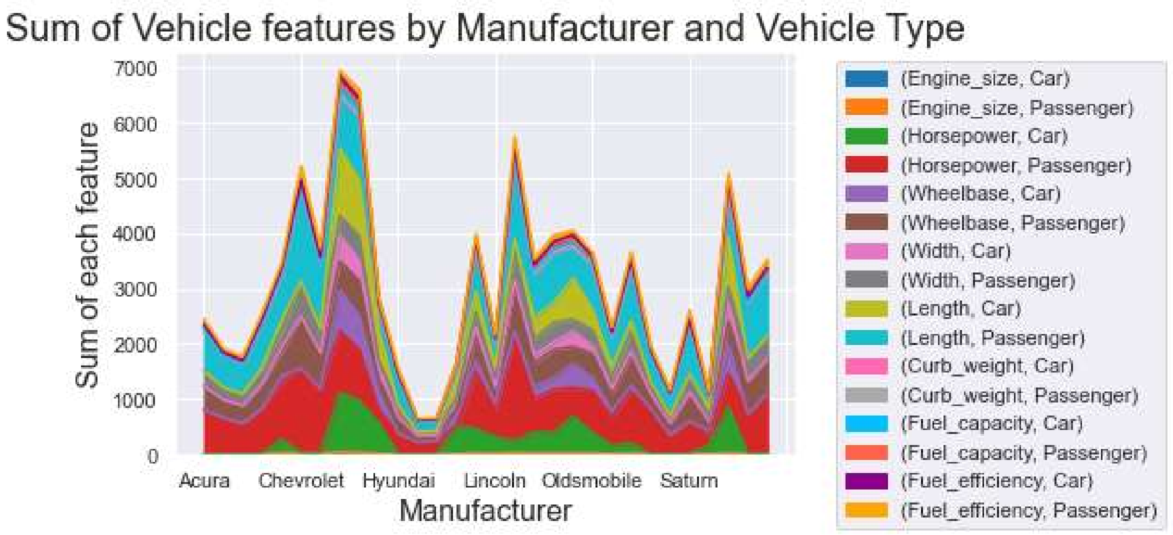


In [ ]:

3. Stacked Area Chart (15 points)

3.1) Create a stacked area chart for your data (or part of your data). Submit a screenshot of the graph and a screenshot of your code

In [403]:
```
# Group the data by Manufacturer and Vehicle_type and get the sum of Sales_in_
area_chart = carsales.groupby(['Manufacturer', 'Vehicle_type'])['Engine_size',

# Define a custom color palette
mycolors = ['#1f77b4', '#ff7f0e', '#2ca02c', '#d62728', '#9467bd', '#8c564b',

# Create an area chart
area_chart.plot.area(color=mycolors)

# Set the legend
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')

# Set the x and y axis labels
plt.xlabel('Manufacturer', fontsize=16)
plt.ylabel('Sum of each feature', fontsize=16)

# Set the plot title
plt.title('Sum of Vehicle features by Manufacturer and Vehicle Type', fontsize

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

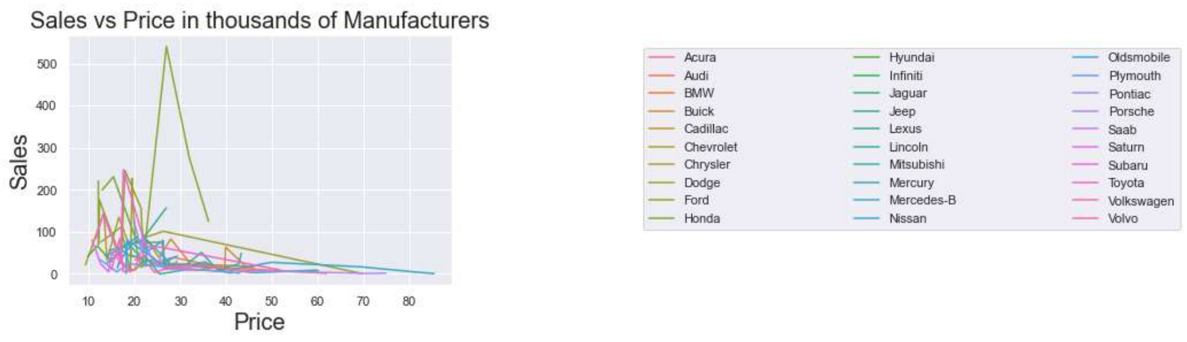


In [ ]:

4. Line Chart with Multiple Lines (25 points)

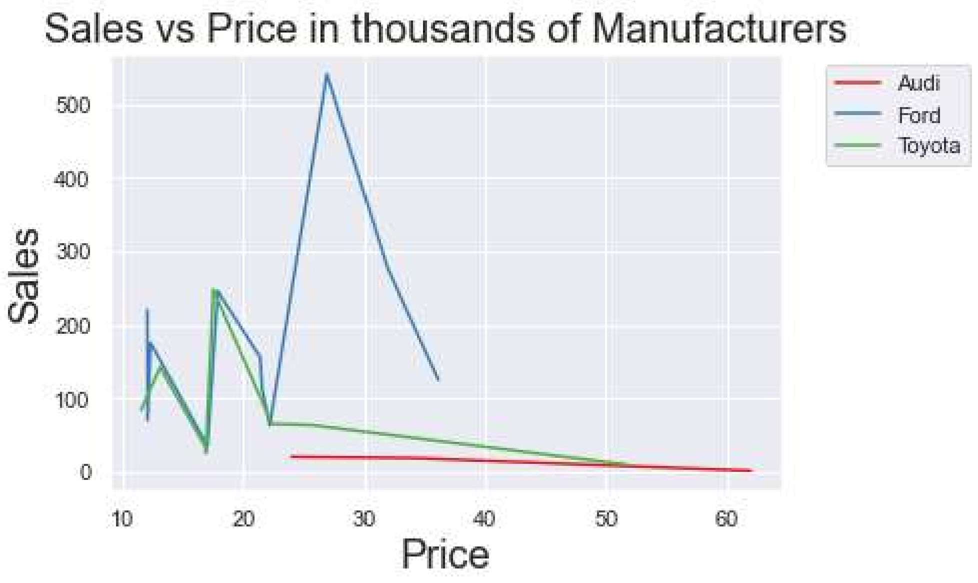
4.1) Create a line chart for your data. Submit a screenshot of the graphand a screenshot of your code.

```
In [404]: # Create a line chart
          sns.lineplot(data=carsales, x='Price_in_thousands', y='Sales_in_thousands', hu

          # Set the legend
          plt.legend(bbox_to_anchor=(1.5, 0.9, 1.4, .05), loc='upper left', ncol=3, mode

          # set x and y axis labels
          plt.xlabel('Price',  fontsize=20)
          plt.ylabel('Sales',  fontsize=20)

          # Set the plot title
          plt.title('Sales vs Price in thousands of Manufacturers', fontsize=20)

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



4.2) Create another line chart which is more comparative

In [410]:
```
# Filter the dataset to include only 'Ford' and 'Audi' and 'Toyota'
line_chart_optimized = carsales.loc[(carsales['Manufacturer'] == 'Ford') | (ca

# Create a line chart with colors
sns.lineplot(data=line_chart_optimized, x='Price_in_thousands', y='Sales_in_th

# Set the legend
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')

# set x and y axis labels
plt.xlabel('Price',  fontsize=20)
plt.ylabel('Sales',  fontsize=20)

# Set the plot title
plt.title('Sales vs Price in thousands of Manufacturers', fontsize=20)

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



In [ ]:

5. Interactive Chart(20 points)

5.1) Create any chart of your choice for your data and make it interactive. Submit a screenshot of the graph and a screenshot of your code

In [411]:

```
# Create the dropdown selection
dropdown = alt.binding_select(options=list(carsales['Manufacturer'].unique()))
selection = alt.selection_single(fields=['Manufacturer'], bind=dropdown, name=
# Create the interactive selection to change scale of chart
interval = alt.selection_interval()
zoom = alt.selection_interval(bind='scales', encodings=['x'])
# Create the scatter plot
scatter = alt.Chart(carsales).mark_circle().encode(
    x=alt.X('Price_in_thousands:Q', title='Price (in thousands)'),
    y=alt.Y('Sales_in_thousands:Q', title='Sales (in thousands)'),
    color=alt.condition(selection, 'Manufacturer:N', alt.value('lightgray')),
    tooltip=['Model:N', 'Latest_Launch:N', 'Price_in_thousands:Q', 'Sales_in_t
).add_selection(selection).properties(
    width=800,
    height=500,
    title='Car Sales by Price and Manufacturer'
).add_selection(
    zoom,interval
).interactive(bind_y=False)

# Show the plot
scatter
```

Out[411]:

In [ ]:

In [ ]: