CSCE 5320 Scientific Data Visualization
ICE-9
Using Color and Size in Visualization

1. Encoding Data using Color and Size

1.1 Please show part of your dataset

Firstly, I have chosen Car Sales Dataset for this task, and I have loaded my data into the GitHub Gist. Now I will be using the raw data path of my data to access it via Jupyter Notebook.

GitHub Gist:
https://gist.githubusercontent.com/nehabaddam/1f47243bf7cd359b25e88d9c100b8248/raw/cc814c12211b47b99233144a642dd8ada0fb52a5/car_sales.csv

```
Manufacturer,Model,Sales_in_thousands,__year_resale_value,Vehicle_type,Price_in_thousands,Engine_size,Horsepower,Wheelbase,Width,Length,Curb_weight,Fuel_capacity,Fuel_efficiency,Latest_Launch,Power_perf_factor
Acura,Integra,16.919,16.36,Passenger,21.5,1.8,140,101.2,67.3,172.4,2.639,13.2,28,2/2/2012,58.28014952
Acura,TL,39.384,19.875,Passenger,28.4,3.2,225,108.1,70.3,192.9,3.517,17.2,25,6/3/2011,91.37077766
Acura,CL,14.114,18.225,Passenger,,3.2,225,106.9,70.6,192,3.47,17.2,26,1/4/2012,
Acura,RL,8.588,29.725,Passenger,42,3.5,210,114.6,71.4,196.6,3.85,18,22,3/10/2011,91.38977933
Audi,A4,20.397,22.255,Passenger,23.99,1.8,150,102.6,68.2,178,2.998,16.4,27,10/8/2011,62.7776392
Audi,A6,18.78,23.555,Passenger,33.95,2.8,200,108.7,76.1,192,3.561,18.5,22,8/9/2011,84.56510502
Audi,A8,1.38,39,Passenger,62,4.2,310,113,74,198.2,3.902,23.7,21,2/27/2012,134.6568582
BMW,323i,19.747,,Passenger,26.99,2.5,170,107.3,68.4,176,3.179,16.6,26,6/28/2011,71.19120671
BMW,328i,9.231,28.675,Passenger,33.4,2.8,193,107.3,68.5,176,3.197,16.6,24,1/29/2012,81.87706856
BMW,528i,17.527,36.125,Passenger,38.9,2.8,193,111.4,70.9,188,3.472,18.5,25,4/4/2011,83.9987238
Buick,Century,91.561,12.475,Passenger,21.975,3.1,175,109,72.7,194.6,3.368,17.5,25,11/2/2011,71.18145132
Buick,Regal,39.35,13.74,Passenger,25.3,3.8,240,109,72.7,196.2,3.543,17.5,23,9/3/2011,95.63670253
Buick,Park Avenue,27.851,20.19,Passenger,31.965,3.8,205,113.8,74.7,206.8,3.778,18.5,24,3/23/2012,85.82840825
Buick,LeSabre,83.257,13.36,Passenger,27.885,3.8,205,112.2,73.5,200,3.591,17.5,25,7/23/2011,84.25452581
Cadillac,DeVille,63.729,22.525,Passenger,39.895,4.6,275,115.3,74.5,207.2,3.978,18.5,22,2/23/2012,113.8545976
Cadillac,Seville,15.943,27.1,Passenger,44.475,4.6,275,112.2,75,201,,18.5,22,4/29/2011,115.6213578
Cadillac,Eldorado,6.536,25.725,Passenger,39.665,4.6,275,108,75.5,200.6,3.843,19,22,11/27/2011,113.7658739
Cadillac,Catera,11.185,18.225,Passenger,31.01,3,200,107.4,70.3,194.8,3.77,18,22,9/28/2011,83.48309358
Cadillac,Escalade,14.785,,Car,46.225,5.7,255,117.5,77,201.2,5.572,30,15,4/17/2012,109.5091165
Chevrolet,Cavalier,145.519,9.25,Passenger,13.26,2.2,115,104.1,67.9,180.9,2.676,14.3,27,8/17/2011,46.36334747
Chevrolet,Malibu,135.126,11.225,Passenger,16.535,3.1,170,107,69.4,190.4,3.051,15,25,3/19/2012,67.31446216
Chevrolet,Lumina,24.629,10.31,Passenger,18.89,3.1,175,107.5,72.5,200.9,3.33,16.6,25,5/24/2011,69.9913956
Chevrolet,Monte Carlo,42.593,11.525,Passenger,19.39,3.4,180,110.5,72.7,197.9,3.34,17,27,12/22/2011,72.03091719
Chevrolet,Camaro,26.402,13.025,Passenger,24.34,3.8,200,101.1,74.1,193.2,3.5,16.8,25,10/23/2011,81.11854333
Chevrolet,Corvette,17.947,36.225,Passenger,45.705,5.7,345,104.5,73.6,179.7,3.21,19.1,22,5/12/2012,141.14115
Chevrolet,Prizm,32.299,9.125,Passenger,13.96,1.8,120,97.1,66.7,174.3,2.398,13.2,33,9/11/2011,48.2976361
Chevrolet,Metro,21.855,5.16,Passenger,9.235,1,55,93.1,62.6,149.4,1.895,10.3,45,4/13/2012,23.27627233
Chevrolet,Impala,107.995,,Passenger,18.89,3.4,180,110.5,73,200,3.389,17,27,6/18/2011,71.83803944
Chrysler,Sebring Coupe,7.854,12.36,Passenger,19.84,2.5,163,103.7,69.7,190.9,2.967,15.9,24,1/16/2012,65.95718396
Chrysler,Sebring Conv.,32.775,14.18,Passenger,24.495,2.5,168,106,69.2,193,3.332,16,24,11/17/2011,69.52135505
Chrysler,Concorde,31.148,13.725,Passenger,22.245,2.7,200,113,74.4,209.1,3.452,17,26,6/6/2012,80.02378204
Chrysler,Cirrus,32.306,12.64,Passenger,16.48,2,132,108,71,186,2.911,16,27,10/6/2011,53.56619987
Chrysler,LHS,13.462,17.325,Passenger,28.34,3.5,253,113,74.4,207.7,3.564,17,23,5/8/2012,101.3292807
Chrysler,Town & Country,53.48,19.54,Car,,,,,,,,,7/13/2011,
Chrysler,300M,30.696,,Passenger,29.185,3.5,253,113,74.4,197.8,3.567,17,23,2/10/2012,101.6552441
Dodge,Neon,76.034,7.75,Passenger,12.64,2,132,105,74.4,174.4,2.567,12.5,29,12/12/2011,52.08489875
Dodge,Avenger,4.734,12.545,Passenger,19.045,2.5,163,103.7,69.1,190.2,2.879,15.9,24,7/1/2012,65.65050834
Dodge,Stratus,71.186,10.185,Passenger,20.23,2.5,168,108,71,186,3.058,16,24,10/31/2011,67.87610784
Dodge,Intrepid,88.028,12.275,Passenger,22.505,2.7,202,113,74.7,203.7,3.489,17,,6/2/2012,80.83147017
Dodge,Viper,0.916,58.47,Passenger,69.725,8,450,96.2,75.7,176.7,3.375,19,16,8/7/2011,188.144323
Dodge,Ram Pickup,227.061,15.06,Car,19.46,5.2,230,138.7,79.3,224.2,4.47,26,17,3/6/2012,90.21170005
Dodge,Ram Wagon,16.767,15.51,Car,21.315,3.9,175,109.6,78.8,192.6,4.245,32,15,1/6/2012,71.13529161
Dodge,Ram Van,31.038,13.425,Car,18.575,3.9,175,127.2,78.8,208.5,4.298,32,16,7/26/2012,70.07832154
Dodge,Dakota,111.313,11.26,Car,16.98,2.5,120,131,71.5,215,3.557,22,19,11/25/2011,49.64500177
Dodge,Durango,101.323,,Car,26.31,5.2,230,115.7,71.7,193.5,4.394,25,17,6/27/2012,92.85412522
```

I have used the read_csv command in the pandas library to get the data and load it into a pandas data frame df. Now I am just printing the dataset using the head function to display the first five rows of the dataset.

```
# Load the data
carsales = pd.read_csv('https://gist.githubusercontent.com/nehabaddam/1f47243bf7cd359b25e88d9c100b8248/raw/cc814c12211b47b9923314
```

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.

```
print(carsales.shape)
carsales.head()
```

```
(157, 16)
```

	Manufacturer	Model	Sales_in_thousands	__year_resale_value	Vehicle_type	Price_in_thousands	Engine_size	Horsepower	Wheelbase	Width	Length	Curb
0	Acura	Integra	16.919	16.360	Passenger	21.50	1.8	140.0	101.2	67.3	172.4	
1	Acura	TL	39.384	19.875	Passenger	28.40	3.2	225.0	108.1	70.3	192.9	
2	Acura	CL	14.114	18.225	Passenger	NaN	3.2	225.0	106.9	70.6	192.0	
3	Acura	RL	8.588	29.725	Passenger	42.00	3.5	210.0	114.6	71.4	196.6	
4	Audi	A4	20.397	22.255	Passenger	23.99	1.8	150.0	102.6	68.2	178.0	

This is the Car sales data set including information about different cars. I have taken this dataset from Kaggle, it is basically car sales information that has many attributes related to a car. It basically has records of car sales of different manufacturing companies, the car models, sales information, price information, and car features altogether. It has around 157 records. This data can be used to visualize how Car sales are dependent on different variables like Horsepower, Price, Car features, etc.

The Car Sales Dataset consists of 16 attributes that define the different features of Car Sales.

1) **Manufacturer**: the name of the car manufacturer. Ex: Audi, Toyota, Ford.
2) **Model**: the name of the car model. Ex: Caravan.
3) **Sales_in_thousands**: the number of cars sold.
4) **__year_resale_value**: the resale value of the car after one year of launch.
5) **Vehicle_type**: the type of vehicle. Ex: Passenger, Car, etc.
6) **Price_in_thousands**: the price of the car.
7) **Engine_size**: the size of the engine (in liters).
8) **Horsepower**: the horsepower rating.
9) **Wheelbase**: the distance between the front and rear axles (in inches).
10) **Width**: the width of the car (in inches).
11) **Length**: the length of the car (in inches).
12) **Curb_weight**: the weight of the car(in pounds).
13) **Fuel_capacity**: the capacity of the fuel tank (in gallons).
14) **Fuel_efficiency**: the fuel efficiency (in miles per gallon).
15) **Latest_Launch**: the date when the car was launched.
16) **Power_perf_factor**: a measure of the car's performance, calculated using horsepower, curb weight, and other factors.

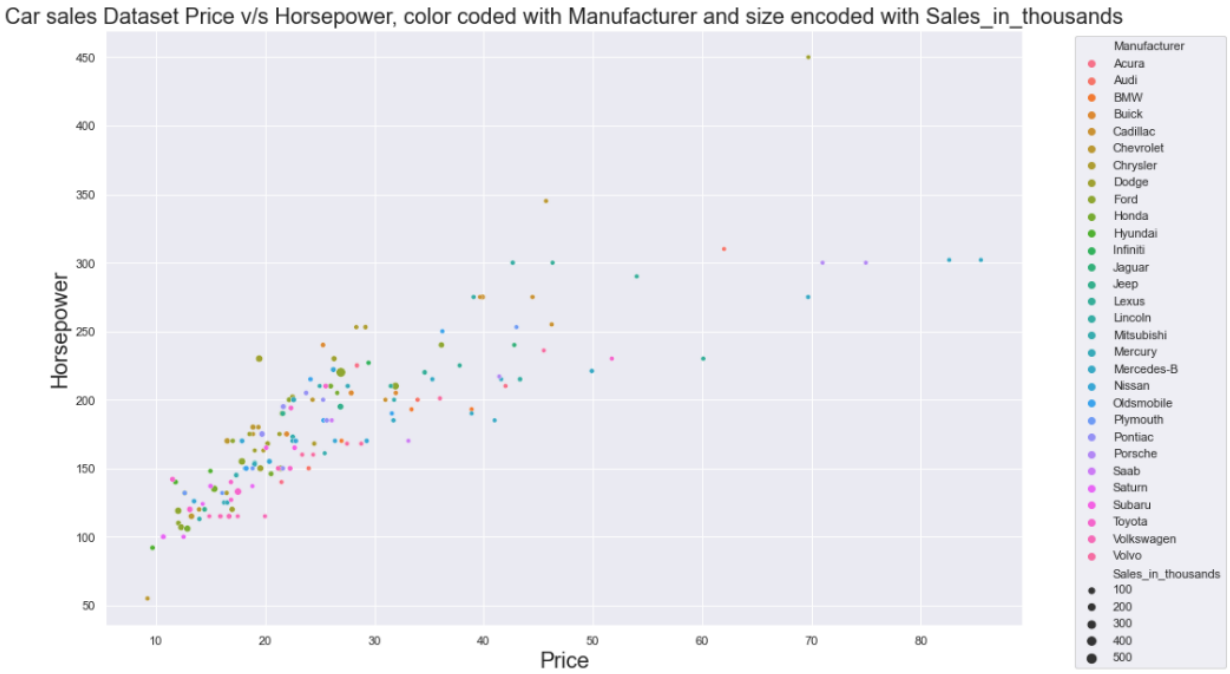
1.2 Encoding the data with x-y channels, add both color and size to your graph, different colors and size should represent different attributes of the data.

Firstly, I am using a seaborn library for visualizing my data. I am using only columns Manufacturer, Sales_in_thousands, Horsepower and Price_in_thousands

Secondly, I am plotting a scatter plot using my Car Sales dataset. I am using the Sales_in_thousands and Horsepower as x and y channels respectively. Now I am using the Manufacturer attribute to color code the plot using the hue attribute of the scatterplot function in Seaborn, each manufacturer represents a different color dynamically selected in the scatterplot function in Seaborn. To represent the size of the data points, I am using the Sales_in_thousands attribute, which defines the size of the data based on the number of sales. All these attributes are passed to the scatterplot function.

Finally, we are using the matplotlib library to define the figure size of the plot, the x and y labels, and the title of the plot, we display the legend with encoding information and finally we use the show function to display the chart.

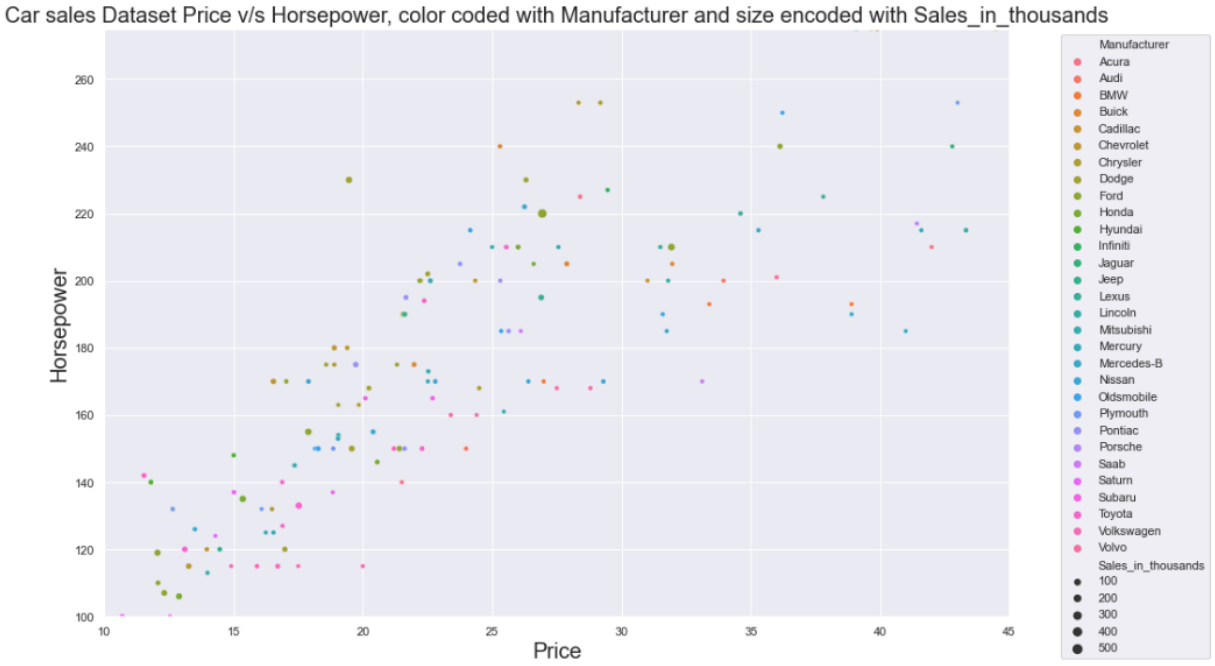
```
# create scatter plot

# 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='Manufacturer', size='Sales_in_thousands')

# 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 Manufacturer and size encoded with Sales_in_thousands',
          fontsize=20)

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



1.3 Try to Optimize your graph and explain why and how you optimize it.

Now I am optimizing the code, by ignoring all the outliers. As we know, it is always better to ignore the outliers, because they represent extremities in the data. I am optimizing the code by ignoring all the less dense areas of the plot by changing the scale of my plot. To do that I am setting axis limits on both the x and y axis. I am considering the limit of (10, 45) for the x-axis and (100, 275) for the y-axis. Below is the optimized code. Now we have a denser, optimized plot with the axes limit of (10, 45) for the x-axis and (100, 275) for the y-axis. This will improve the way we visualize by removing all the unnecessary data points from the chart.

```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='Manufacturer', size='Sales_in_thousands')

# 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 Manufacturer and size encoded with Sales_in_thousands',
          fontsize=20)

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



2. Stacked & Grouped Bar Chart

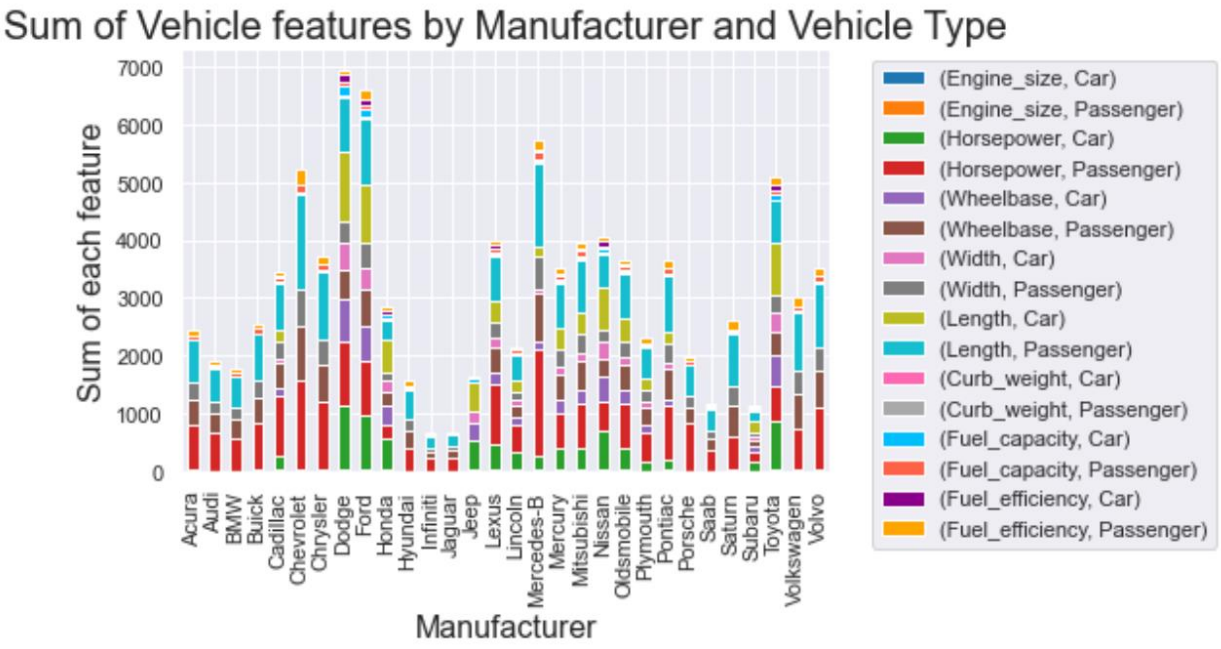
2.1 Create a stacked & Grouped Bar Chart for your data.

For creating stacked and grouped bar charts, I am first creating grouped data. This grouped data is formed by using Manufacturer as an index and Vehicle type as a main column. I am using other columns 'Engine_size', 'Horsepower', 'Wheelbase', 'Width', 'Length', 'Curb_weight',

'Fuel_capacity', 'Fuel_efficiency' to find their sums with respect to the Manufacturer and Vehicle type. So, this Stacked and Grouped bar chart displays how the sums of the selected columns vary with Vehicle types in each manufacturer. The graph displays the same. We are using a color palette to display each feature for each vehicle type using a different color. We plot the Manufacturers on the x-axis and Sums of features on the y-axis.

Finally, we are using the matplotlib library to define the x and y labels, and the title of the plot, we display the legend with encoding information and finally we use the show function to display the chart.

```
# Group the data by Manufacturer and Vehicle_type and get the sum of Sales_in_thousands
grouped_stack = carsales.groupby(['Manufacturer', 'Vehicle_type'])['Engine_size', 'Horsepower', 'Wheelbase' , 'Width' ,'Length',

# Define a custom color palette
mycolors = ['#1f77b4', '#ff7f0e', '#2ca02c', '#d62728', '#9467bd', '#8c564b', '#e377c2', '#7f7f7f', '#bcbd22', '#17becf', '#ff69b

# 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=20)

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



2.2Analysis of the data based on the bar chart.

From the above chart, we can observe that the variation of columns 'Engine_size', 'Horsepower', 'Wheelbase', 'Width', 'Length', 'Curb_weight', 'Fuel_capacity', 'Fuel_efficiency' with respect to the Vehicle Type for each manufacturer.

For Example, If the manufacturer is Dodge, we can observe that the Horsepower for both Car and Passenger Vehicles is almost the same. Whereas the Wheelbase for Cars is greater than the Wheelbase for Passenger Vehicles. The width of the car is greater than the passenger Vehicle. The length of the car is greater than the passenger Vehicle. The Fuel capacity for Cars is greater than the Fuel capacity for Passenger Vehicles. The Fuel efficiency for Cars is greater than the Fuel efficiency for Passenger Vehicles.

For Example, If the manufacturer is Mercedes Benz, we can observe that the Horsepower for a Car is less than for a Passenger vehicle. The Wheelbase for Cars is also lesser than the Wheelbase for Passenger Vehicles. The width of the car is much lesser than the passenger Vehicle. The length of the car is much lesser than the passenger Vehicle. The Fuel capacity for Cars is much lesser than the Fuel capacity for Passenger Vehicles. The Fuel efficiency for Cars is much lesser than the Fuel efficiency for Passenger Vehicles.

In conclusion, we can use this chart to compare the features of cars like engine power, horsepower, length, width, wheelbase, curb weight, fuel capacity, and fuel efficiency for Vehicle types Car and Passengers for each Manufacturer.

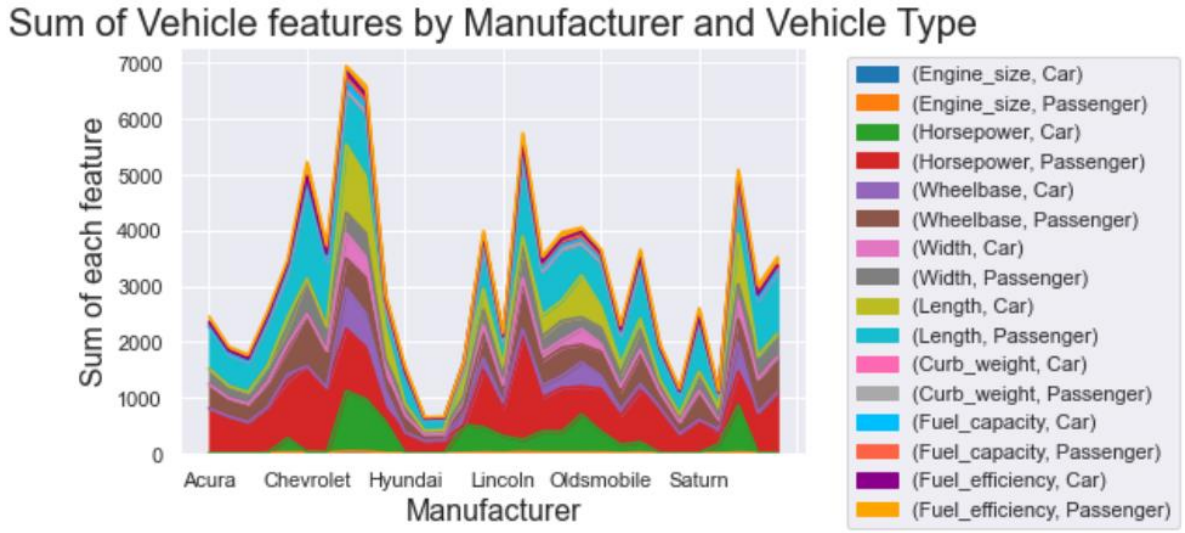
3. Stacked Area Chart

3.1Create a stacked area chart for your data.

For creating stacked area charts, I am first creating grouped data. This grouped data is formed by using Manufacturer as an index and Vehicle type as a main column. I am using other columns 'Engine_size', 'Horsepower', 'Wheelbase', 'Width', 'Length', 'Curb_weight', 'Fuel_capacity', 'Fuel_efficiency' to find their sums with respect to the Manufacturer and Vehicle type. So, this area chart displays how the sums of the selected columns vary with Vehicle types in each manufacturer. We are using a color palette to display each feature for each vehicle type using a different color. We plot the Manufacturers on the x-axis and Sums of features on the y-axis.

Finally, we are using the matplotlib library to define the x and y labels, and the title of the plot, we display the legend with encoding information and finally we use the show function to display the chart.

```
# Group the data by Manufacturer and Vehicle_type and get the sum of Sales_in_thousands
area_chart = carsales.groupby(['Manufacturer', 'Vehicle_type'])['Engine_size', 'Horsepower', 'Wheelbase' , 'Width' ,'Length', 'Cu

# Define a custom color palette
mycolors = ['#1f77b4', '#ff7f0e', '#2ca02c', '#d62728', '#9467bd', '#8c564b', '#e377c2', '#7f7f7f', '#bcbd22', '#17becf', '#ff69b

# Create an area chart
area.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=20)

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



3.2Analysis of the data based on the area chart. Explain the trends or changes.

From the above chart, we can observe that the variation of columns 'Engine_size', 'Horsepower', 'Wheelbase', 'Width', 'Length', 'Curb_weight', 'Fuel_capacity', 'Fuel_efficiency' with respect to the Vehicle Type for each manufacturer. Unlike a bar chart, this area chart shows the data in the form of area waves one over the other.

For example, we can see that most of the area is covered by Horsepower, indicating that no matter what the manufacturer is horsepower is an important feature among all Vehicles. We can also differentiate the horsepower of a Car and a Passenger Vehicle.

We can also observe that the Vehicles from Hyundai Manufacturers have fewer features compared to other manufacturers. And Chevrolet has the highest sum of features.

We can observe the trends in Car and Passenger Vehicle types, for features Horsepower, Wheelbase, Length, and Width clearly.

In conclusion, we can use this chart to compare and study the high-level trends in the features of cars like engine power, horsepower, length, width, wheelbase, curb weight, fuel capacity, and fuel efficiency for Vehicle types of Car and Passengers for each Manufacturer.

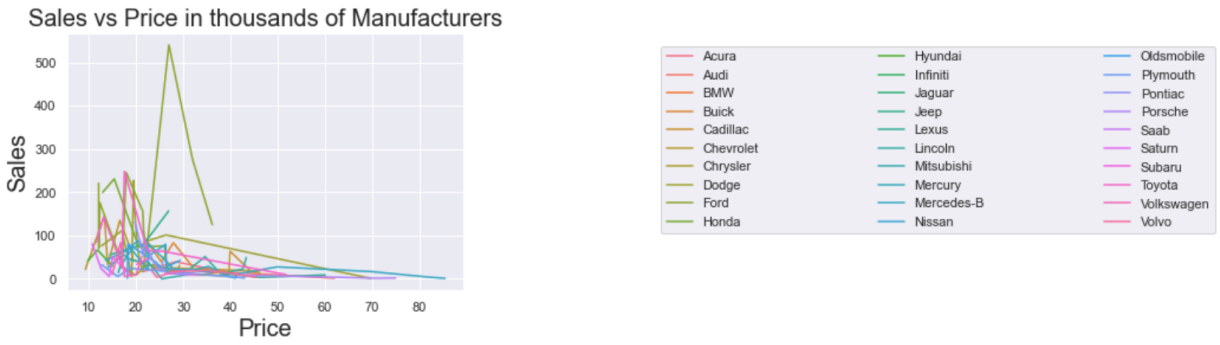
4. Line Chart with Multiple Lines

4.1Create a line chart for your data.

For creating a line chart, I am using columns Price_in_thousands and Sales_in_thousands for all the Manufacturers. I am plotting Price_in_thousands on the x-axis and Sales_in_thousands on the y-axis. I am passing Manufacturer to hue, displaying each line in a different color representing each manufacturer. This will plot a line chart to display Sales data against the Price with each manufacturer represented using a different color.

Finally, we are using the matplotlib library to define the x and y labels, and the title of the plot, we display the legend with encoding information and finally we use the show function to display the chart.

```
# Create a line chart
sns.lineplot(data=carsales, x='Price_in_thousands', y='Sales_in_thousands', hue='Manufacturer')

# Set the legend
plt.legend(bbox_to_anchor=(1.5, 0.9, 1.4, .05), loc='upper left', ncol=3, mode="expand", borderaxespad=0.)

# 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()
```

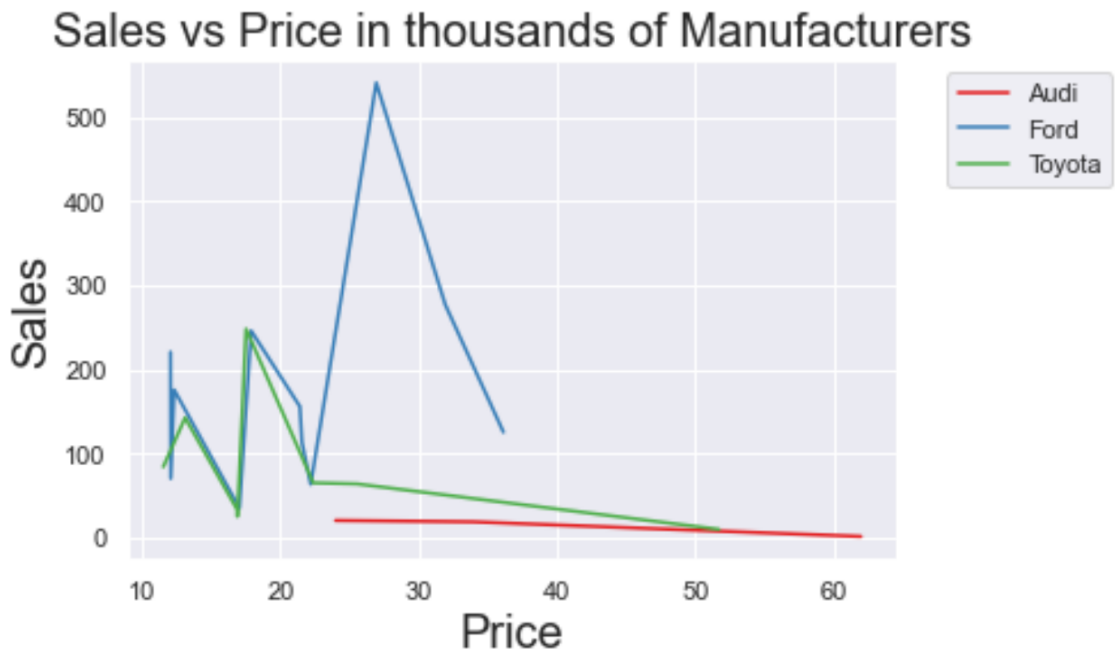


4.2Create another line chart that is more comparative.

For creating another line chart, I am filtering the car data set to only get the data for car Manufacturers Audi, Ford, and Toyota. I am plotting Price_in_thousands on the x-axis and Sales_in_thousands on the y-axis. I am passing Manufacturer to hue, displaying each line in a different color representing each manufacturer. This will plot a line chart to display Sales data against the Price with each of the three manufacturers represented using a different color.

Finally, we are using the matplotlib library to define the x and y labels, and the title of the plot, we display the legend with encoding information and finally we use the show function to display the chart.

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

# Create a line chart with colors
sns.lineplot(data=line_chart_optimized, x='Price_in_thousands', y='Sales_in_thousands', hue='Manufacturer')

# 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()
```



4.3Analysis of the data based on both line charts. Explain the trends or changes.

In the First Line Chart, we can see how the Sales of the vehicles changed with the Price of the Vehicle for all Manufacturers. Each Manufacturer is represented using a different color line. We

can observe the chart and realize that the sales are high when the Price is low, to be specific, the sales of vehicles of most of the Manufactures are more between the Price range of 10,000 to 30,000 dollars. We can also observe that Ford has the highest Sales around 500,000 compared to all the Manufacturers, especially among the Vehicles between the price range of 20,000 and 30,000 dollars. We can also observe that, as the price range increases, the sales decrease drastically for most of the Manufacturers. Also, this chart has lines overlapping with each other, making it hard to analyze the data for every Manufacturer. So, we can develop a new chart that displays only a few manufacturers to better understand the trends of those specific manufacturers.

In the Second Line Chart, we can see how the Sales of the vehicles changed with the Price of the Vehicle for 3 specific Manufacturers i.e., Audi, Ford, and Toyota. For Audi, we can see that the Sales are always under 50,000 dollars and are almost constant between the price is between 20,000 and 70,000 dollars, with a slight decrease in sales as the price increases. For Ford, we can see both an increase and decrease in sales between the price range of 10,000 and 20,000 and a sudden rise in sales after that. We can also observe that Ford has the highest Sales around 500,000, especially among the Vehicles between the price range of 20,000 and 30,000 dollars, the sales decrease when the price range increases. For Toyota, we can see that the Sales are both increasing and decreasing under the price range of 20,000 but gradually decreasing when the price is above 20,000, it has peak sales under the 20,000 dollar price range. Using this chart, we can clearly analyze the sales of each manufacturer without any overlapping lines.

5. Interactive Chart

5.1 Create any chart of your choice for your data and make it interactive.

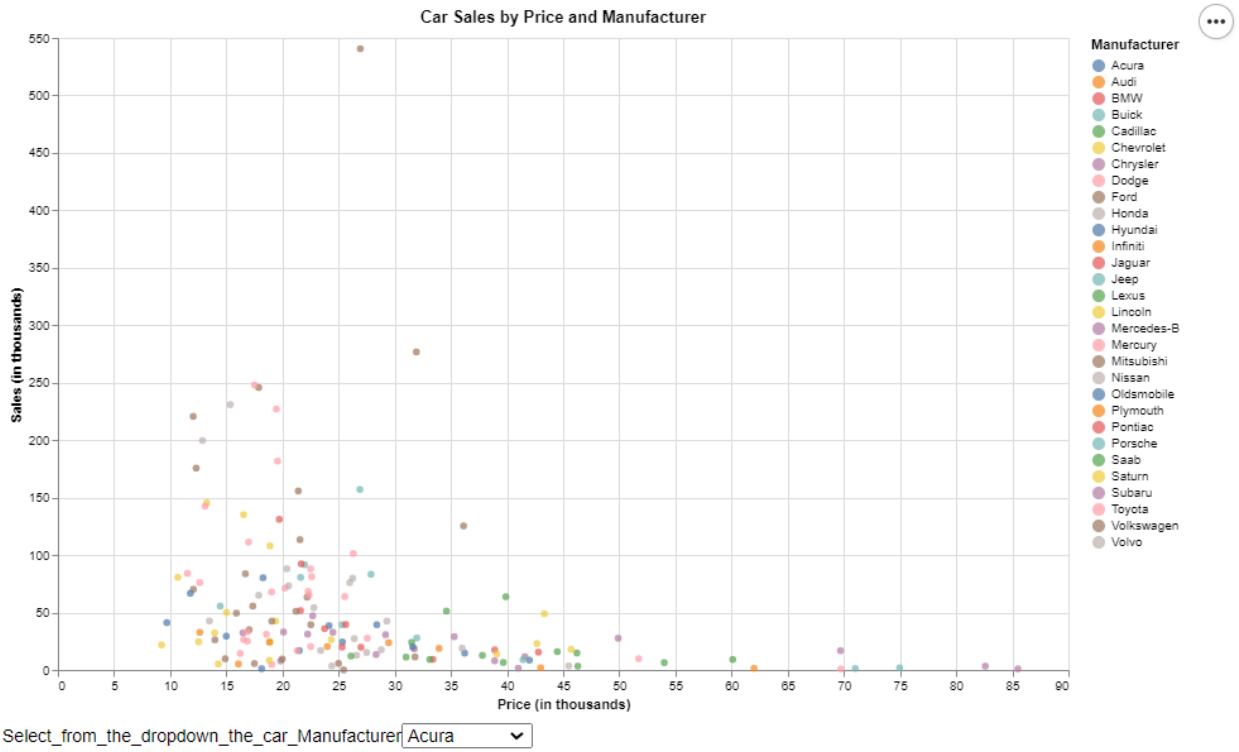
Firstly, I am using an Altair library for visualizing my data and making the chart interactive. I am using only columns Manufacturer, Sales_in_thousands, and Price_in_thousands. The selection function displays the dropdown to select the manufacturer. I have also added an interval function to zoom in and out of the chart interactively, We can even move the chart scales.

Secondly, I am plotting a scatter plot using my Car Sales dataset. I am using the Price_in_thousands and Sales_in_thousands as x and y channels respectively with respective labels. Now I am using the Manufacturer attribute to color code the plot using the selection function. The tooltip displays the Model, Latest Launch, Price_in_thousands, and Sales_in_thousands for each data point on the chart. We also define the Height and Width of the chart including the title.

Finally, we are using the scatter to display the chart.

```
# Create the dropdown selection
dropdown = alt.binding_select(options=list(carsales['Manufacturer'].unique()))
selection = alt.selection_single(fields=['Manufacturer'], bind=dropdown, name='Select from the dropdown the car')
# 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_thousands:Q']
).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
```

The Plot looks like below. It displays Sales against the Price of Vehicles for each manufacturer. The different colors represent different Manufacturers. The legend shows the colors of each manufacturer. The labels and title are in place.

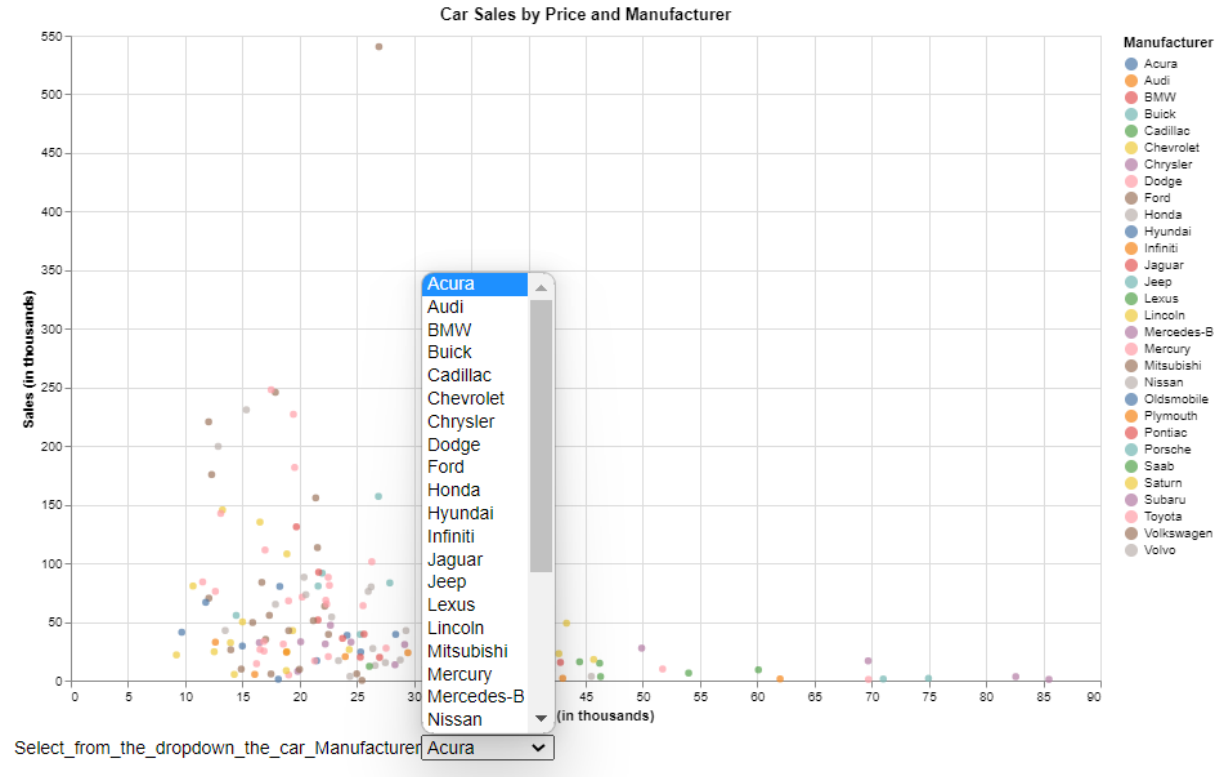


5.2 Give an explanation about your interactive chart.

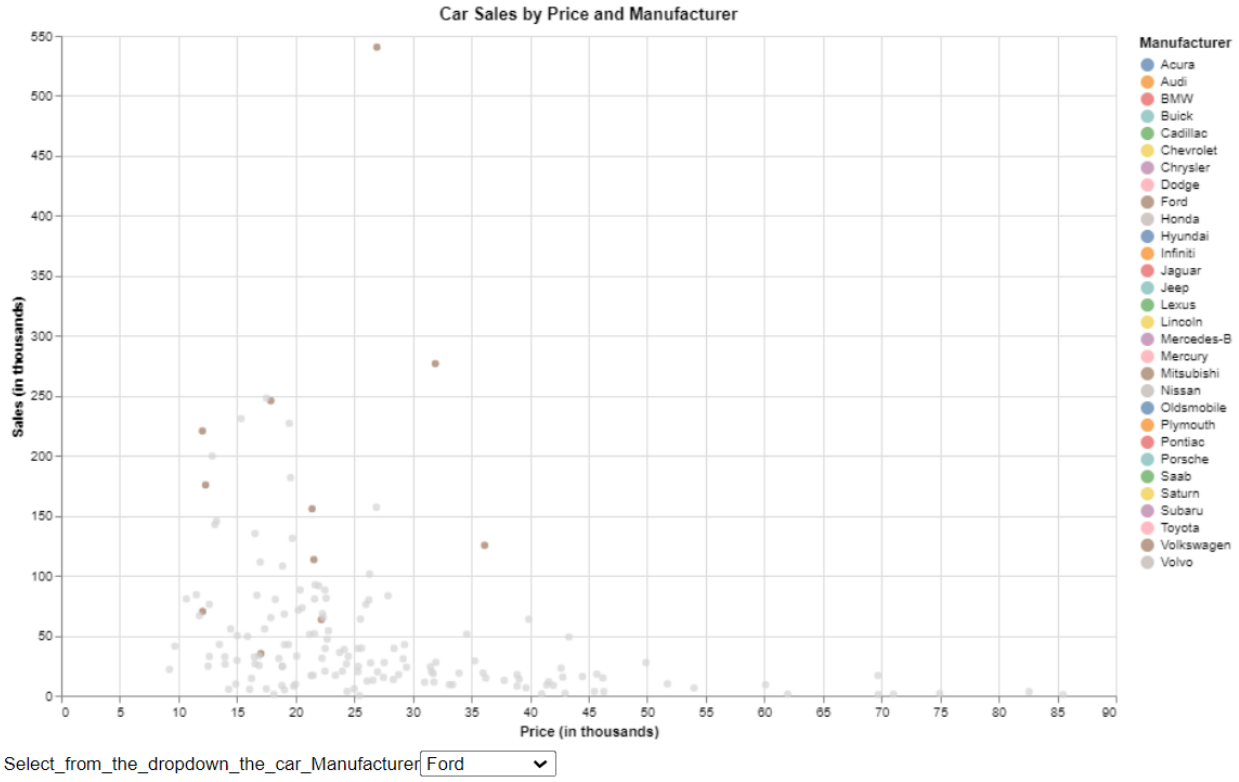
Firstly, the chart displays the Sales and Price information of the Vehicles of different Manufacturers. We can just hover over the data points to display the tooltip with Model, Latest

Launch, Price_in_thousands, and Sales_in_thousands information. This information can be viewed for all the data points on the chart.

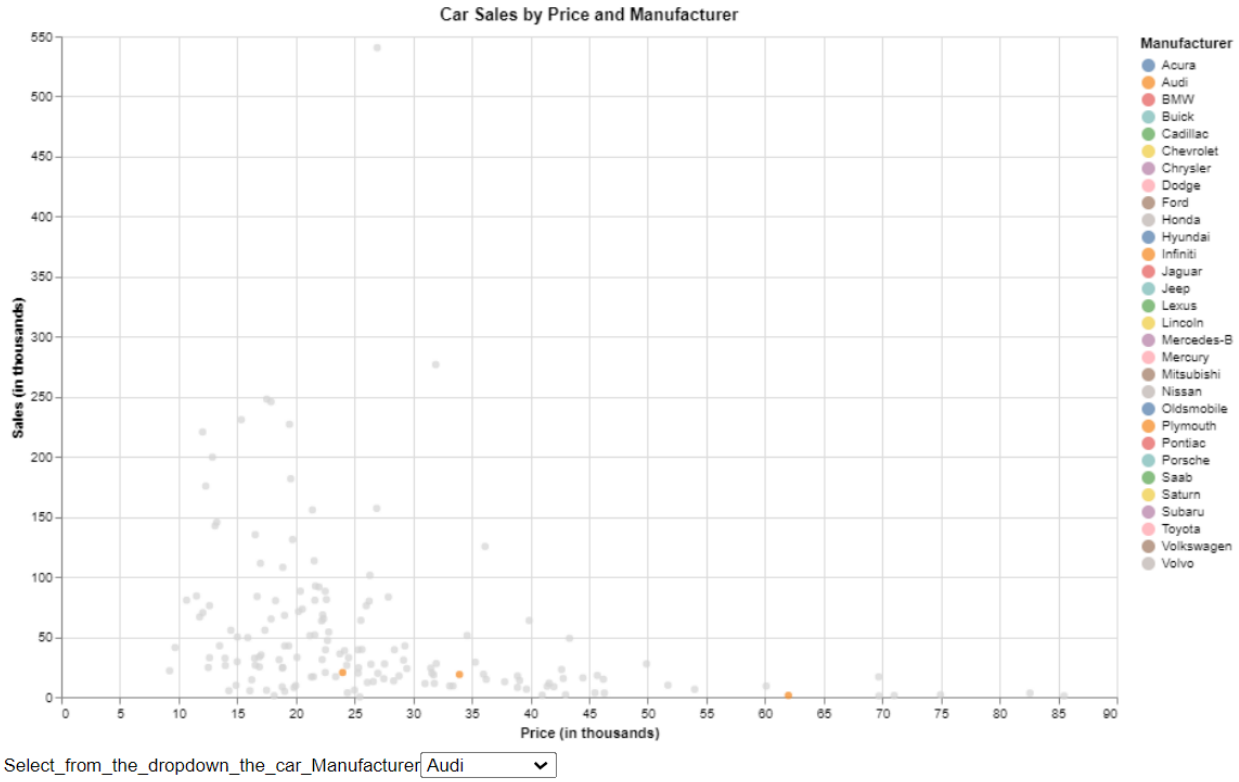
Secondly, We can choose the Manufacturer from the dropdown menu. All the manufacturers are displayed in the dropdown. Whenever we select a manufacturer, only the data points specific to the selected manufacturer are displayed with color, the rest of the data points are grayed out.



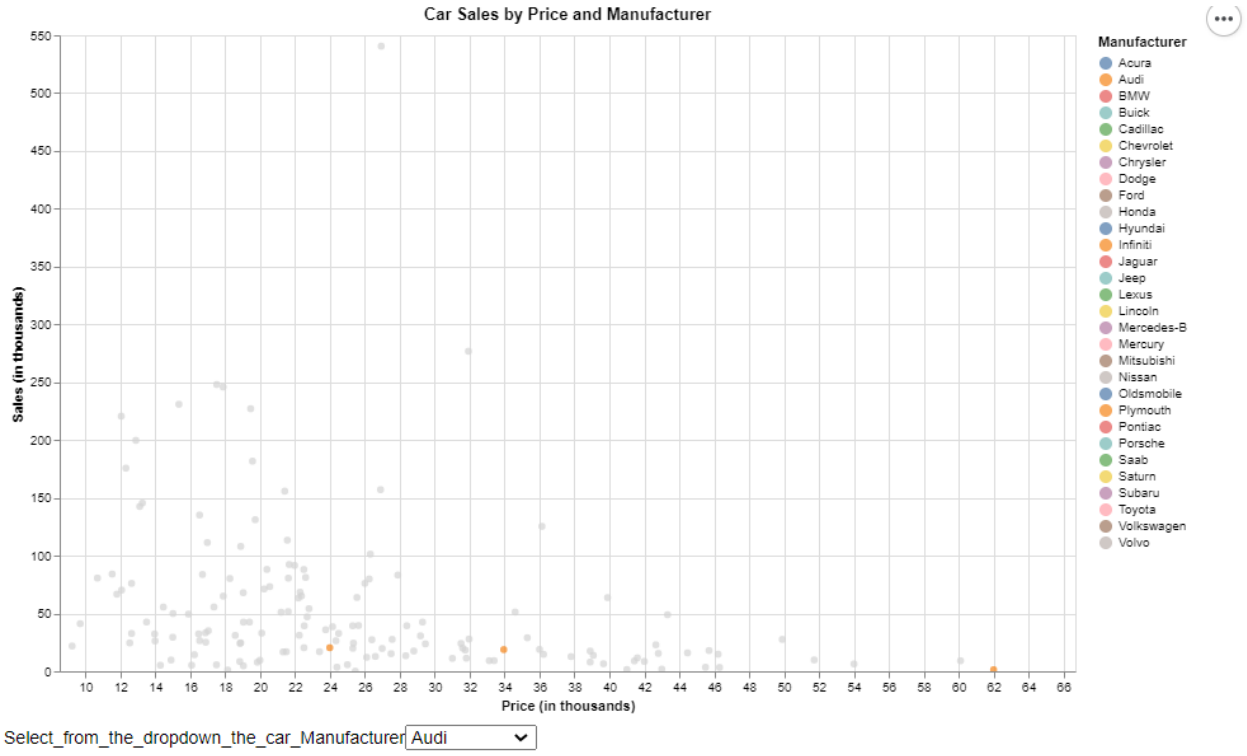
I have selected a Manufacturer Ford as shown below. Now we can see that all data points have turned gray but the data points for Ford are shown in tan color.



I have selected a Manufacturer Audi as shown below. Now we can see that all data points have turned gray but the data points for Audi are shown in yellow color.



To the above chart, we can even zoom in and scale the chart as we want. We can see below how the scale on the x-axis is changed. Before it was shown as a 5-digit increment. Now, the scale is 2 digits incremented.



In conclusion, we can use this chart to display the Sales and Prices of Vehicles for selected Manufacturers from the dropdown and by interactively changing the scale of the chart.