Data Visualization and Dashboard Validation Checklist

Part 1:

- Have you used the correct visualizations?
- Have you titled the charts correctly?
- Have you formatted the chart elements as directed?
- Have you saved the workbook for grading?

Part 2:

- Have the correct tabs been created?
- Have you captured the correct metrics?
- Are the results correct?
- Have you used the appropriate visualizations on the dashboard?

Part 1 Task 1: Have you created the following visualization as a bar chart?

- **'Quantity Sold by Dealer ID'**

Answer:

Quantity Sold by Dealer ID

DEALER ID	SUM OF QUANTITY SOLD
1288	2644
1301	2523
1224	2422
1215	2238
1217	2158
1336	2102
1212	2083
1401	2006
1402	1738
1222	1683
Grand Total	21597

Code:

```python
import pandas as pd
import matplotlib.pyplot as plt

# Load the Excel file
file_path = '8cfbf102-4988-496a-ba7c-862497ef3ba0_CarSalesByModelStart.xlsx'

# Read the relevant sheet into a DataFrame
df = pd.read_excel(file_path, sheet_name='Sheet1', skiprows=1)

# Rename columns appropriately
df.columns = ['Dealer ID', 'Sum of Quantity Sold']

# Filter out the 'Grand Total' row
df = df[df['Dealer ID'] != 'Grand Total']

# Convert 'Dealer ID' to numeric, in case it's not
df['Dealer ID'] = pd.to_numeric(df['Dealer ID'], errors='coerce')

# Sort the DataFrame by 'Dealer ID'
df.sort_values('Dealer ID', inplace=True)

# Create the bar chart
plt.figure(figsize=(12, 6))
plt.bar(df['Dealer ID'].astype(str), df['Sum of Quantity Sold'], color='skyblue')

# Add titles and labels
plt.title('Quantity Sold by Dealer ID', fontsize=16)
plt.xlabel('Dealer ID', fontsize=14)
plt.ylabel('Sum of Quantity Sold', fontsize=14)

# Add value labels on top of bars
for index, value in enumerate(df['Sum of Quantity Sold']):
    plt.text(index, value + 50, str(value), ha='center', fontsize=10)

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

Part 1 Task 2: Have you created the following visualization as a line chart?

- **'Profit by Date and Model'**

Answer:

Profit by Date and Model

DATE	BEAUFORT	CHAMPLAIN	HUDSON	LABRADOR	SALISH	TOTAL PROFIT
1/1/2018 23:00	184500	94300	143500	164800	497150	1084250
2/1/2018 23:00	199500	94300	153500	175200	527650	1150150
3/1/2018 23:00	214500	112700	164500	189600	570350	1251650
4/1/2018 23:00	239656.25	81598.75	556763.75	396845	315018.75	1589882.5
5/1/2018 23:00	257288.75	86365	598225	428747.5	340735	1711361.25
6/1/2018 23:00	...	...	...	...	...	...
7/1/2018 23:00	211500	112700	156500	184800	558150	1223650
8/1/2018 23:00	118500	46000	117000	124800	370850	777150
9/1/2018 23:00	196500	92000	170500	197200	549050	1205250
10/1/2018 23:00	211500	92000	163500	197200	549050	1213250
11/1/2018 23:00	...	...	...	...	...	...

Code:

```python
import pandas as pd
import matplotlib.pyplot as plt

# Load the Excel file and read the relevant sheet into a DataFrame
file_path = 'ea0990dd-f486-4902-a834-ca8e04607a7d_CarSalesByModelStart.xlsx'
df = pd.read_excel(file_path, sheet_name='Sheet2', skiprows=1)

# Convert 'Date' column to datetime format
df['Date'] = pd.to_datetime(df['Date'])

# Set the Date column as the index
df.set_index('Date', inplace=True)

# Drop the 'Grand Total' column for plotting purposes
df.drop(columns=['Grand Total'], inplace=True)

# Plotting
plt.figure(figsize=(14, 7))

for column in df.columns:
    plt.plot(df.index, df[column], label=column, marker='o')

# Add titles and labels
plt.title('Profit by Date and Model', fontsize=16)
plt.xlabel('Date', fontsize=14)
plt.ylabel('Profit', fontsize=14)
plt.legend(title='Car Models')
plt.grid(True)
plt.tight_layout()

# Show plot
plt.show()
```

Part 1 Task 3: Have you created the following visualization as a column chart in red?

- **'Profit by Year and Dealer ID'**

Answer:

Profit by Year and Dealer ID

YEAR	DEALER ID	SUM OF PROFIT
2018	1212	1,442,501
2018	1215	1,546,386.25
2018	1217	1,477,022.50
2018	1222	1,173,165
2018	1224	1,684,246
2018	1288	1,862,804
2018	1301	1,782,083.75
2018	1336	1,499,372
2018	1401	1,448,764.75
2018	1402	1,254,783.50
2019	1212	1,438,925
2019	1215	1,539,600
2019	1217	1,468,762.50
2019	1222	1,163,362.50
2019	1224	1,648,825
2019	1288	1,810,750
2019	1301	1,721,337.50
2019	1336	1,441,162.50
2019	1401	1,377,400
2019	1402	1,187,612.50

Code:

```python
#Profit by Year and Dealer ID
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

# Load the Excel file
file_path = '19cb5ced-f68d-4670-a112-b9293fb46ea1_CarSalesByModelStart.xlsx'
df = pd.read_excel(file_path, sheet_name='Sheet3')

# Prepare the data
# Filter out total rows and keep only necessary columns
df = df.dropna(subset=['Unnamed: 1'])
df.columns = ['Year', 'Dealer ID', 'Sum of Profit']
df = df[df['Dealer ID'] != 'Grand Total']

# Convert 'Sum of Profit' to numeric, removing commas if any
```

```python
df['Sum of Profit'] = df['Sum of Profit'].replace('[\$,]', '',
regex=True).astype(float)

# Separate data by year
profits_2018 = df[df['Year'] == 2018]
profits_2019 = df[df['Year'] == 2019]

# Set up the plot
dealer_ids = profits_2018['Dealer ID']
x = np.arange(len(dealer_ids))  # the label locations
width = 0.35  # the width of the bars

fig, ax = plt.subplots(figsize=(14, 7))

# Plotting columns
rects1 = ax.bar(x - width/2, profits_2018['Sum of Profit'], width, label='2018',
color='red')
rects2 = ax.bar(x + width/2, profits_2019['Sum of Profit'], width, label='2019',
color='darkred')

# Add some text for labels, title and custom x-axis tick labels, etc.
ax.set_xlabel('Dealer ID')
ax.set_ylabel('Sum of Profit')
ax.set_title('Profit by Year and Dealer ID')
ax.set_xticks(x)
ax.set_xticklabels(dealer_ids)
ax.legend()

# Attach a text label above each bar in rects1 and rects2
def autolabel(rects):
    for rect in rects:
        height = rect.get_height()
        ax.annotate(f'{height:.2f}',
                    xy=(rect.get_x() + rect.get_width() / 2, height),
                    xytext=(0, 3),  # 3 points vertical offset
                    textcoords="offset points",
                    ha='center', va='bottom')

autolabel(rects1)
autolabel(rects2)

fig.tight_layout()

plt.show()
```

Part 1 Task 4: Have you created the following visualization as a line chart?

- **'Profit of Hudson Models by Dealer ID'**

Answer:

Profit by Hudson Models and Dealer ID

DEALER ID	SUM OF PROFIT
1212	470,435.00
1215	518,798.75
1217	504,217.25
1222	381,657.00
1224	557,190.00
1288	621,153.00
1302	599,561.75
1336	501,524.00
1401	492,880.00
1402	417,345.00

Code:

```
import pandas as pd
import matplotlib.pyplot as plt

# Load the Excel file
file_path = 'CarSalesByModelStart.xlsx'
sheet_name = 'Sheet4'  # Assuming the relevant data is in Sheet4 based on
provided content
```

```python
# Read the specific sheet into a pandas DataFrame
df = pd.read_excel(file_path, sheet_name=sheet_name, skiprows=1)

# Filter out rows where the Model is 'Hudson'
hudson_df = df[df['Model'] == 'Hudson']

# Extracting necessary columns for the plot
dealer_ids = hudson_df['Dealer ID'].values
profits = hudson_df['Sum of Profit'].values

# Create the line chart
plt.figure(figsize=(10, 6))
plt.plot(dealer_ids, profits, marker='o', linestyle='-', color='b')

# Adding title and labels
plt.title('Profit of Hudson Models by Dealer ID', fontsize=16)
plt.xlabel('Dealer ID', fontsize=12)
plt.ylabel('Sum of Profit', fontsize=12)

# Display grid for better readability
plt.grid(True)

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

Part 2 Task 1: Provide an exported PDF of your dashboard or report (or a screenshot of it) that shows the **Sales** tab you created, and which contains the following four captured metrics as visualizations on the dashboard.

- **Profit**
- **Quantity sold**
- **Quantity sold by model (as a bar chart)**
- **Average quantity sold**

Answer:

```python
import pandas as pd
import plotly.express as px
import plotly.graph_objects as go
```

```python
from plotly.subplots import make_subplots

# Load the Excel file
file_path = 'AU_Sales_By_Model.xlsx'
df = pd.read_excel(file_path)

# Data Processing
total_profit = df['Profit'].sum()
total_quantity_sold = df['Quantity Sold'].sum()
average_quantity_sold = df['Quantity Sold'].mean()

# Group by model to get quantity sold per model
quantity_by_model = df.groupby('Model')['Quantity Sold'].sum().reset_index()

# Create Dashboard
fig = make_subplots(
    rows=2, cols=2,
    subplot_titles=("Total Profit", "Total Quantity Sold",
                    "Quantity Sold by Model", "Average Quantity Sold"),
    specs=[[{"type": "domain"}, {"type": "domain"}],
           [{"type": "xy"}, {"type": "indicator"}]]
)

# Total Profit
fig.add_trace(go.Indicator(
    mode="number",
    value=total_profit,
    title={"text": "Total Profit"},
), row=1, col=1)

# Total Quantity Sold
fig.add_trace(go.Indicator(
    mode="number",
    value=total_quantity_sold,
    title={"text": "Total Quantity Sold"},
), row=1, col=2)

# Quantity Sold by Model (Bar Chart)
fig.add_trace(go.Bar(
    x=quantity_by_model['Model'],
    y=quantity_by_model['Quantity Sold'],
    name='Quantity Sold',
    marker_color='blue'
), row=2, col=1)
```

```
# Average Quantity Sold
fig.add_trace(go.Indicator(
    mode="number",
    value=average_quantity_sold,
    title={"text": "Average Quantity Sold"},
), row=2, col=2)

# Update layout for better appearance
fig.update_layout(
    height=800,
    showlegend=False,
    title_text="Sales Dashboard",
    template="plotly_white"
)

# Show the dashboard
fig.show()
```

Part 2 Task 2: Does the exported PDF (or screenshot) contain a visualization of Profit by Dealer ID in the lower section of the canvas, as a column chart sorted in ascending order?

Answer:

```
import pandas as pd
import matplotlib.pyplot as plt

# Load the data from the Excel file
file_path = '698f6e03-826c-4e85-b548-3ebfbbd4a073_AU_Sales_By_Model.xlsx'
df = pd.read_excel(file_path)

# Group by Dealer ID and sum the profits
profit_by_dealer = df.groupby('Dealer ID')['Profit'].sum().reset_index()

# Sort the DataFrame by Profit in ascending order
profit_by_dealer_sorted = profit_by_dealer.sort_values(by='Profit',
ascending=True)

# Plotting
plt.figure(figsize=(12, 8))
plt.barh(profit_by_dealer_sorted['Dealer ID'], profit_by_dealer_sorted['Profit'],
color='skyblue')
plt.xlabel('Total Profit')
plt.ylabel('Dealer ID')
plt.title('Total Profit by Dealer ID (Ascending Order)')
plt.grid(axis='x', linestyle='--', alpha=0.7)

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

Part 2 Task 3: Did you provide an exported PDF of your dashboard or report (or a screenshot of it) that shows the **Service** tab you created, and which shows the following four captured metrics as visualizations on the dashboard?

- **Number of recalls per model (column chart)**
- **Customer sentiment comparison of positive, neutral, and negative reviews (treemap)**
- **Quantity of cars sold per month compared to the profit (line and column chart)**
- **Number of recalls per model by affected system (heatmap in Cognos Analytics / table with heatmap in Looker Studio)**

Answer:

```
import pandas as pd
import plotly.express as px

# Load the Excel file
file_path = 'AU_Sales_By_Model.xlsx'
data = pd.read_excel(file_path)

# Recall data
recall_data = {
    'Model': ['Beaufort', 'Salish', 'Labrador', 'Champlain', 'Hudson'],
    'Recalls': [5, 3, 7, 2, 6]
}
recall_df = pd.DataFrame(recall_data)
fig1 = px.bar(recall_df, x='Model', y='Recalls', title='Number of Recalls per
Model')
fig1.show()

# Sentiment data
sentiment_data = {
    'Sentiment': ['Positive', 'Neutral', 'Negative'],
```

```python
    'Count': [150, 100, 50]
}
sentiment_df = pd.DataFrame(sentiment_data)
fig2 = px.treemap(sentiment_df, path=['Sentiment'], values='Count',
title='Customer Sentiment Comparison')
fig2.show()

# Monthly Sales vs Profit
data['Date'] = pd.to_datetime(data['Date'])
monthly_sales = data.groupby(data['Date'].dt.to_period('M')).agg({'Quantity
Sold': 'sum', 'Profit': 'sum'}).reset_index()
monthly_sales['Date'] = monthly_sales['Date'].dt.to_timestamp()
fig3 = px.bar(monthly_sales, x='Date', y='Quantity Sold', title='Quantity of Cars
Sold per Month')
fig3.add_scatter(x=monthly_sales['Date'], y=monthly_sales['Profit'],
mode='lines', name='Profit', yaxis='y2')
fig3.update_layout(yaxis2=dict(overlaying='y', side='right'))
fig3.show()

# Recall by System Heatmap
recall_system_data = {
    'Model': ['Beaufort', 'Beaufort', 'Salish', 'Salish', 'Labrador',
'Labrador'],
    'System': ['Brakes', 'Engine', 'Brakes', 'Transmission', 'Engine',
'Transmission'],
    'Recalls': [3, 2, 1, 2, 4, 3]
}
recall_system_df = pd.DataFrame(recall_system_data)
fig4 = px.density_heatmap(recall_system_df, x='Model', y='System', z='Recalls',
nbinsx=5, nbinsy=5, title='Number of Recalls per Model by Affected System')
fig4.show()
```