**INFOSYS MILESTONE-2 DOCUMENTATION**

**About dataset:**

tips Dataset Description

The tips dataset is a built-in dataset available within the plotly.express library. It is commonly used for demonstrations and learning purposes in data visualization.

**Data Source:** The dataset is a sample of restaurant bills, originally from a study published in the Journal of Statistics Education.

**Contents:**

The dataset contains information about 244 restaurant bills, with the following columns:

* **total\_bill:** The total bill amount in US dollars.
* **tip:** The tip amount in US dollars.
* **sex:** The gender of the bill payer ('Male' or 'Female').
* **smoker:** Whether the party included smokers ('Yes' or 'No').
* **day:** The day of the week the bill was paid ('Thur', 'Fri', 'Sat', 'Sun').
* **time:** The time of day the bill was paid ('Lunch' or 'Dinner').
* **size:** The number of people in the dining party.

**Data Types:**

* **total\_bill:** Numeric (float64)
* **tip:** Numeric (float64)
* **sex:** Categorical (object)
* **smoker:** Categorical (object)
* **day:** Categorical (object)
* **time:** Categorical (object)
* **size:** Numeric (int64)

**Usage:**

This dataset is often used to explore relationships between different variables, such as:

* The relationship between the total bill and tip amount.
* The effect of gender or smoking status on tipping behavior.
* The distribution of bills and tips on different days of the week or times of day.

**SOME QUERIES ON DATASET.**

**Code:**

import plotly.express as px

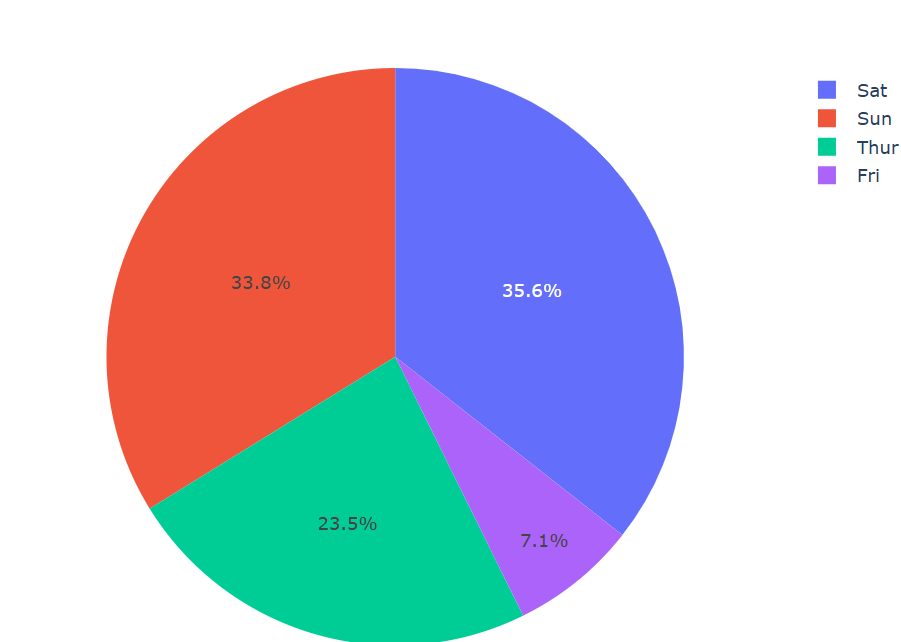
fig = px.pie(df1, values='tip', names='day')

fig.show()

**Explanation:**

This Python code generates a pie chart using the plotly.express library to visualize the distribution of tip amounts across different days of the week based on data in a DataFrame named df1.

**Output:**

****

**Code:**

import plotly.express as px

fig = px.histogram(df1, x="total\_bill", color="day", marginal="rug",

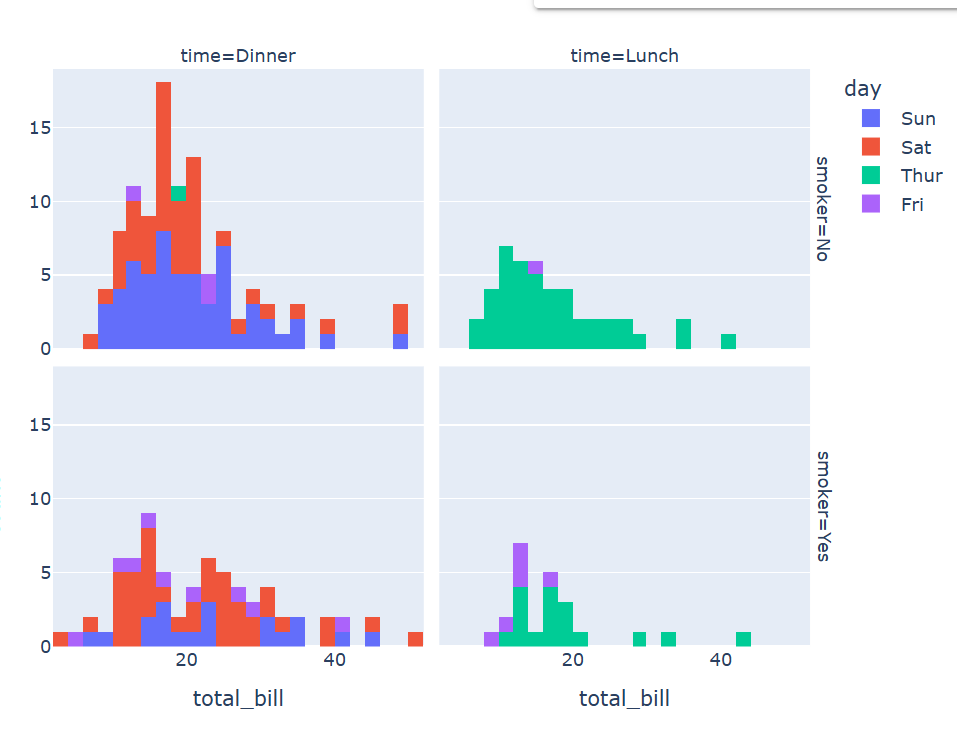
                   nbins=30, facet\_col="time", facet\_row="smoker")

fig.show()

**Explanation:**

This code creates a histogram using the plotly.express library. It visualizes the distribution of total bill amounts across different days and times, categorized by smoking status. The marginal="rug" argument adds a rug plot to show the individual data points, and nbins=30 sets the number of bins for the histogram. The facet\_col="time" and facet\_row="smoker" arguments create subplots based on time and smoking status, respectively.

**Output:**

****

**Code:**

import plotly.express as px

# Assuming your DataFrame is named 'df'

fig = px.bar(df1, x="day", y="total\_bill", color="sex",

             barmode="stack", facet\_col="time", facet\_row="smoker",

             category\_orders={"day": ["Thur", "Fri", "Sat", "Sun"],

                             "time": ["Lunch", "Dinner"]})

fig.show()

**Explanation:**

This code creates a stacked bar chart using the plotly.express library. It visualizes the total bill amount by day, categorized by sex. The chart is further segmented into subplots based on time of day and smoking status.

**Output:**



**Code:**

import plotly.express as px

# Group by day and time, calculate average total\_bill

df\_grouped = df1.groupby(['day', 'time'])['total\_bill'].mean().reset\_index()

# Create polar bar plot

fig = px.bar\_polar(df\_grouped,

                   r="total\_bill",

                   theta="day",

                   color="time",

                   template="plotly\_dark",

                   color\_discrete\_sequence=px.colors.sequential.Plasma\_r,

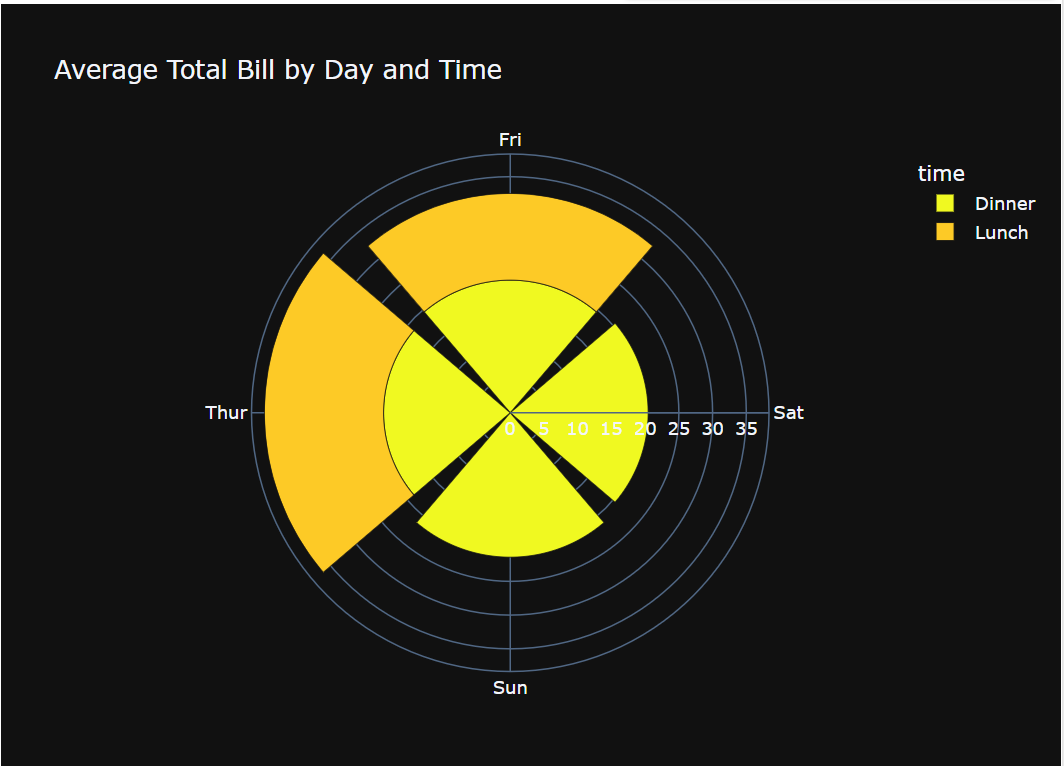
                   title="Average Total Bill by Day and Time")

fig.show()

**Explanation:**

This code visualizes how the average total bill varies across different days of the week and between lunch and dinner times using a circular bar chart. The dark theme and color palette enhance the visual appeal of the plot.

**Output:**

****

**Code:**

import plotly.express as px

# Calculate tip percentage

df1['tip\_percentage'] = (df1['tip'] / df1['total\_bill']) \* 100

# Create polar bar plot

fig = px.bar\_polar(df1,

                   r="tip\_percentage",

                   theta="day",

                   color="sex",

                   template="plotly\_dark",

                   color\_discrete\_sequence=px.colors.sequential.Plasma\_r,

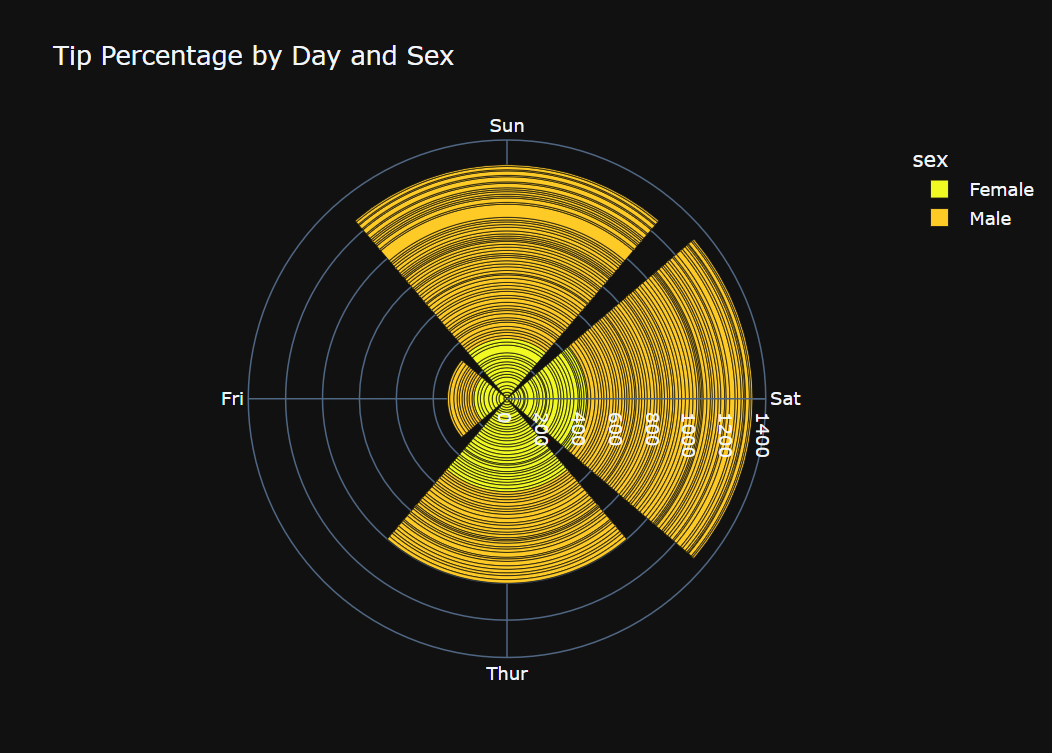
                   title="Tip Percentage by Day and Sex")

fig.show()

**Explanation:**

This code visualizes how tip percentages vary across different days of the week and between genders using a circular bar chart. The dark theme and color palette enhance the visual appeal of the plot.

**Output:**

****

**Code:**

import plotly.express as px

fig = px.bar\_polar(df1,

                   r="total\_bill",

                   theta="day",

                   color="sex",

                   template="plotly\_dark",

                   color\_discrete\_sequence=px.colors.diverging.RdYlGn,  # Diverging color scale

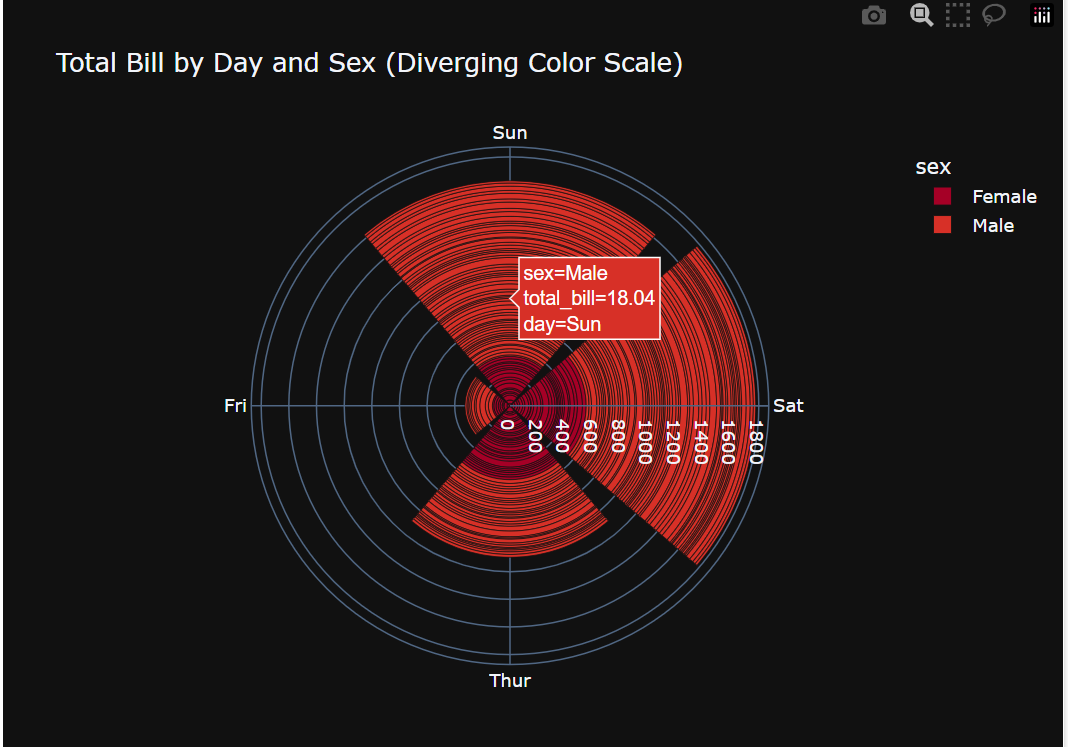
                   title="Total Bill by Day and Sex (Diverging Color Scale)")

fig.show()

**Explanation:**

This code visualizes how the total bill amount varies across different days of the week and between genders using a circular bar chart with a diverging color scale. The diverging color scale helps to highlight differences between the genders in terms of total bill amounts. The dark theme enhances the visual appeal of the plot.

**Output:**

****

**Code:**

import plotly.express as px

# Create the polar bar plot

fig = px.bar\_polar(df1,

                   r="total\_bill",

                   theta="day",

                   color="sex",

                   template="plotly\_dark",

                   color\_discrete\_sequence=px.colors.sequential.Plasma\_r,

                   title="Total Bill by Day and Sex (with Hole)")

# Update the polar subplot to add a hole

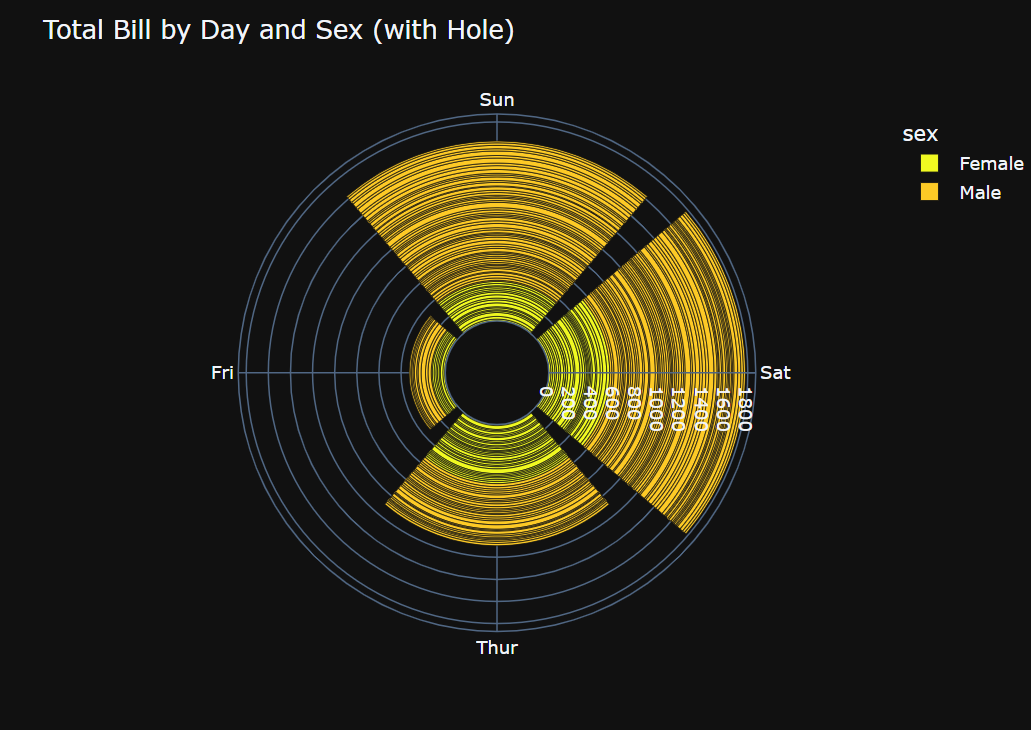
fig.update\_polars(hole=0.2)

fig.show()

**Explanation:**

This code visualizes how the total bill amount varies across different days of the week and between genders using a circular bar chart with a hole in the center. The hole is a stylistic choice that can make the plot more visually appealing and can also be used to display additional information in the center if desired.

**Output:**

****

**Code:**

import plotly.express as px

# Create the polar bar plot

fig = px.bar\_polar(df1,

                   r="total\_bill",

                   theta="day",

                   color="sex",

                   template="plotly\_dark",

                   color\_discrete\_sequence=px.colors.sequential.Plasma\_r,

                   title="Total Bill by Day and Sex (with Sector Highlighting)")

# Add an annotation to highlight the desired sector (e.g., Friday)

fig.add\_annotation(

    x=0.5,  # Adjust x-coordinate for annotation position

    y=0.5,  # Adjust y-coordinate for annotation position

    xref="paper",  # Refer to the plot area for positioning

    yref="paper",

    text="Friday",  # Text to display in the annotation

    showarrow=False,  # Hide arrow

    font=dict(size=16, color="white"),  # Customize font

    bgcolor="rgba(255, 0, 0, 0.5)",  # Set background color with transparency

    bordercolor="red",  # Set border color

    borderwidth=2,  # Set border width

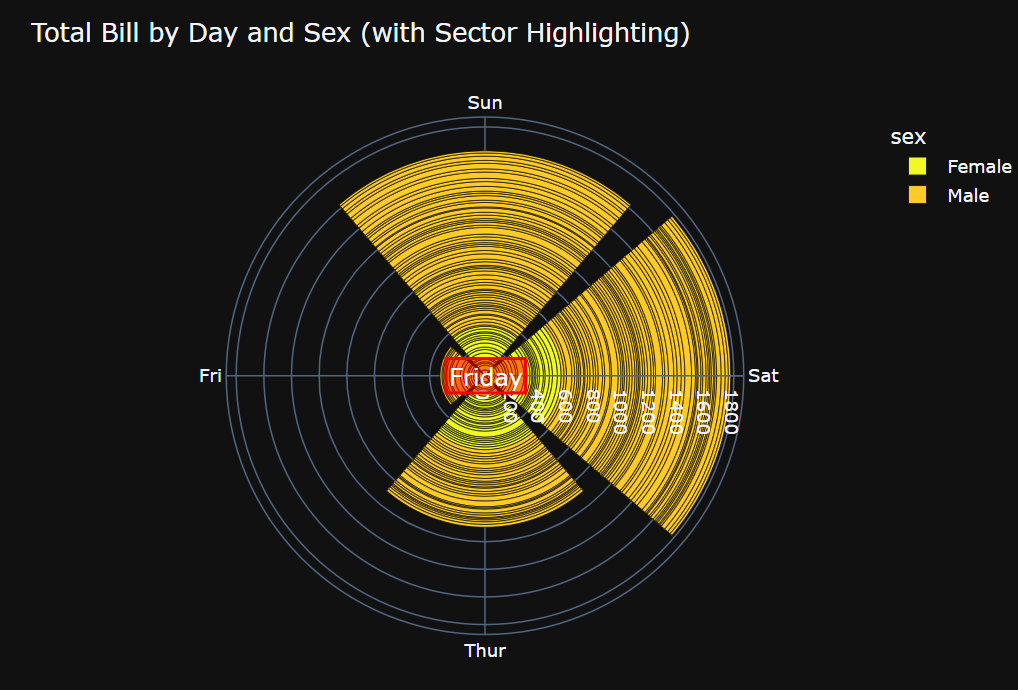
)

fig.show()

**Explanation:**

This code visualizes how the total bill amount varies across different days of the week and between genders using a circular bar chart with a specific sector highlighted. The annotation is used to draw attention to a particular day, in this case, Friday. The dark theme and color palette enhance the visual appeal of the plot.

**Output:**

****

**Code:**

import plotly.express as px

import plotly.graph\_objects as go

from plotly.subplots import make\_subplots

# Numerical columns to include

num\_cols = ['total\_bill', 'tip', 'size']

# Create subplots

fig = make\_subplots(rows=1, cols=len(num\_cols), specs=[[{'type': 'polar'}] \* len(num\_cols)])

# Add traces for each numerical column

for i, col in enumerate(num\_cols):

    trace = px.bar\_polar(df1, r=col, theta="day", color="sex",

                        template="plotly\_dark",

                        color\_discrete\_sequence=px.colors.sequential.Plasma\_r).data[0]

    fig.add\_trace(trace, row=1, col=i + 1)

# Update layout

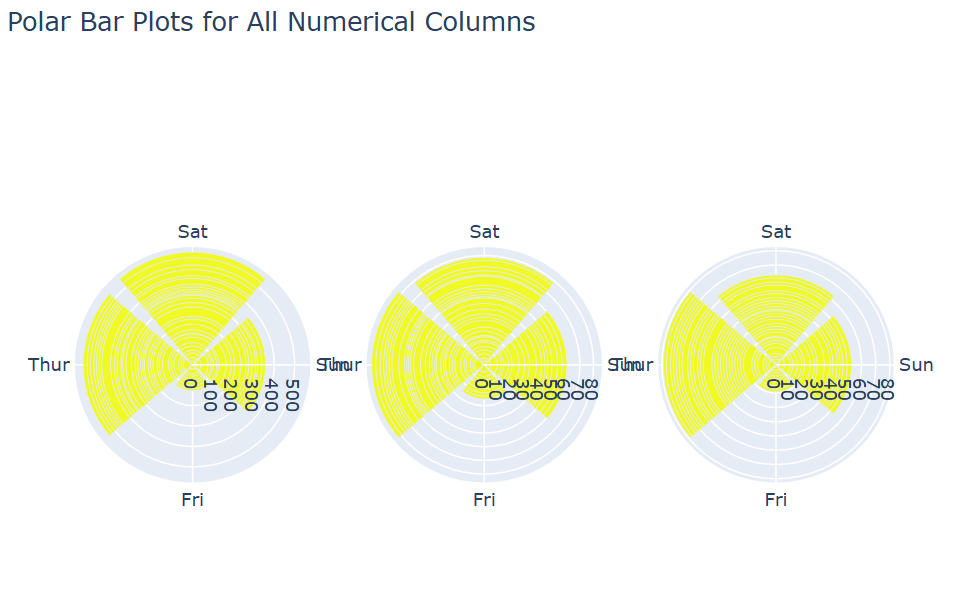
fig.update\_layout(title\_text="Polar Bar Plots for All Numerical Columns", showlegend=False)

fig.show()

**Explanation:**

This code generates a series of polar bar plots, each representing a different numerical variable ('total\_bill', 'tip', 'size') against the categorical variables 'day' and 'sex'. This allows for comparison of these numerical values across different days and genders in a circular format. The dark theme and sequential color palette enhance the visual appeal of the plots. The subplots arrange these plots side-by-side for a comprehensive view.

**Output:**



**Code:**

import plotly.express as px

import pandas as pd

# Melt the DataFrame to combine numerical columns

df\_melted = pd.melt(df1, id\_vars=['sex', 'smoker', 'day', 'time'],

                    value\_vars=['total\_bill', 'tip', 'size'],

                    var\_name='bill\_type', value\_name='amount')

# Create a categorical column combining all categorical variables

df\_melted['all\_categories'] = df\_melted['sex'] + ', ' + df\_melted['smoker'] + ', ' + df\_melted['day'] + ', ' + df\_melted['time']

# Create the polar bar plot

fig = px.bar\_polar(df\_melted,

                   r="amount",

                   theta="all\_categories",

                   color="bill\_type",

                   template="plotly\_dark",

                   color\_discrete\_sequence=px.colors.sequential.Plasma\_r)

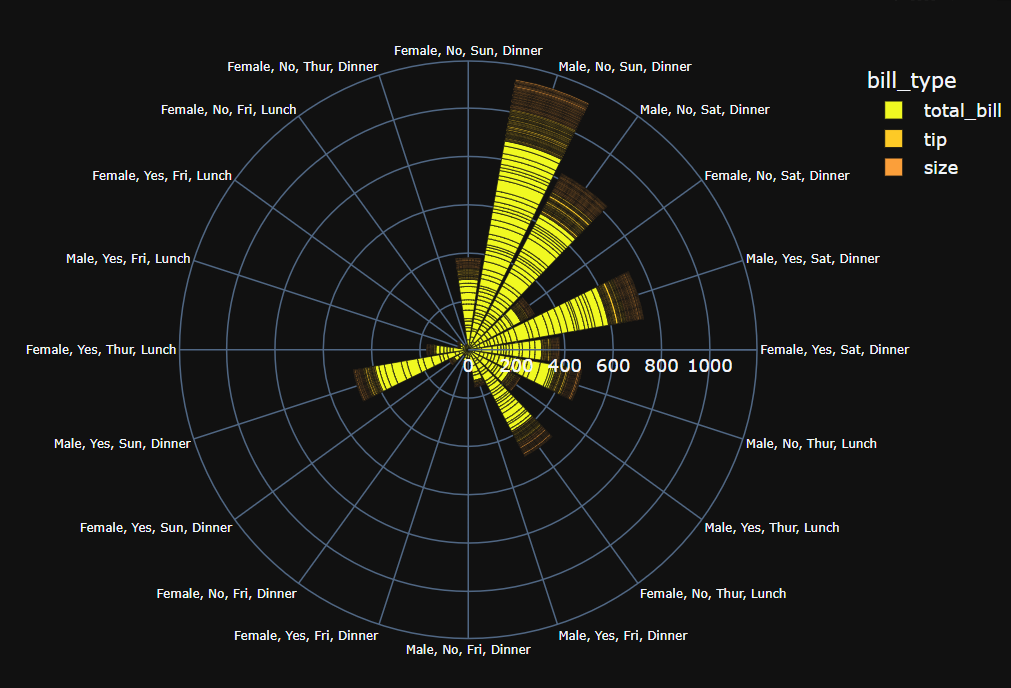
fig.update\_layout(polar=dict(angularaxis=dict(tickfont=dict(size=8)))) # Adjust font size for theta labels

fig.show()

**Explanation:**

This code visualizes the relationship between the melted numerical values ('amount') and the combined categorical information ('all\_categories') using a circular bar chart. This allows for a comprehensive view of how different bill types ('bill\_type') vary across different combinations of gender, smoking status, day, and time. The dark theme and sequential color palette enhance the visual appeal of the plot. The reduced font size for theta labels helps to avoid overcrowding when there are many categories.

**Output:**



**Code:**

import plotly.express as px

import pandas as pd

# Assuming your DataFrame is named 'df1'

# 1. Calculate tip percentage

df1['tip\_percentage'] = (df1['tip'] / df1['total\_bill']) \* 100

# 2. Define a categorical column combining all categorical variables

df1['all\_categories'] = df1['sex'] + ', ' + df1['day']  # Combine gender and day

# 3. Create the polar bar plot

fig = px.bar\_polar(df1,

                   r="total\_bill",  # Represents sales or most selling

                   theta="all\_categories",  # Gender and day combined

                   color="tip\_percentage",  # Represents tip percentage

                   template="plotly\_dark",

                   color\_continuous\_scale=px.colors.sequential.Plasma\_r,

                   title="Sales, Gender, and Tip Percentage")

# 4. Update layout for better readability

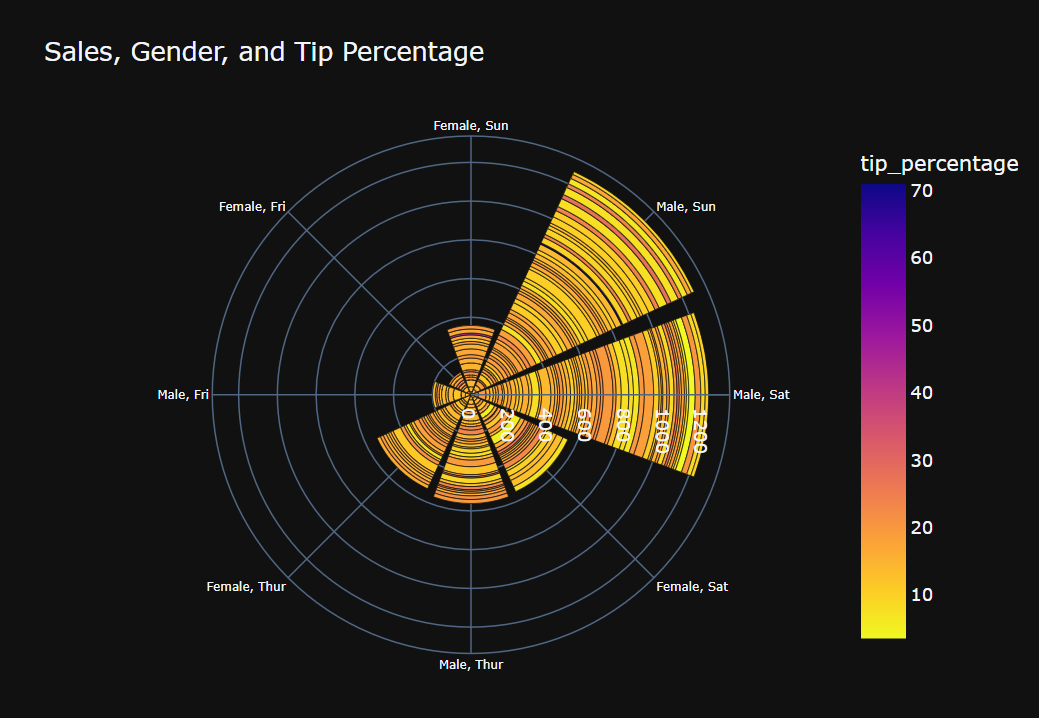
fig.update\_layout(polar=dict(angularaxis=dict(tickfont=dict(size=8))))  # Adjust font size if needed

fig.show()

**Explanation:**

This code visualizes how sales (total bill amount) vary across different combinations of gender and day of the week, while also using color to represent the tip percentage for each combination. This allows for a comprehensive understanding of sales patterns and tipping behavior based on these factors. The dark theme and sequential color scale enhance the visual appeal of the plot. The reduced font size for theta labels helps to avoid overcrowding when there are many categories.

**Output:**

****

**Collab\_Link-** **https://colab.research.google.com/drive/1hfMgC5RhxkRRWNpdA9D7lDwoyEjKf7vk?usp=sharing**