Data Visualization with Plotly & Dash

Agenda

- What is Data Visualization
- Plotly and Dash
- Plotly Themes
- Distribution Plots
- Categorical Plots
- Matrix Plots
- Customize Plots
- Dash

Agenda

- **What is Data Visualization**
- Plotly and Dash
- Plotly Themes
- Distribution Plots
- Categorical Plots
- Matrix Plots
- Customize Plots
- Dash

What is Data Visualization

Data visualization is the graphical representation of information and data. By using visual elements like charts, graphs, and maps, data visualization tools provide an accessible way to see and understand trends, outliers, and patterns in data.

Our eyes are drawn to colors and patterns. We can quickly identify red from blue, square from circle. Our culture is visual, including everything from art and advertisements to TV and movies.

Data visualization is another form of visual art that grabs our interest and keeps our eyes on the message.

Agenda

- What is Data Visualization
- **Plotly and Dash**
- Plotly Themes
- Distribution Plots
- Categorical Plots
- Matrix Plots
- Customize Plots
- Dash

Plotly and Dash

Plotly and Dash work together. Dash can build interactive dashboards with multiple Plotly graphs.

Dash is a Open Source Python framework based on flask framework for creating reactive, Web-based applications.

https://dash.plotly.com/

https://plotly.com/python/

Plotly and Dash

Install

```
pip install plotly
pip install dash
pip install dash-bootstrap-components
```

Use

```
from dash.dependencies import Input, Output, State
from dash import dcc
from dash import html
import dash_bootstrap_components as dbc
import plotly.express as px
import plotly.figure_factory as ff
```

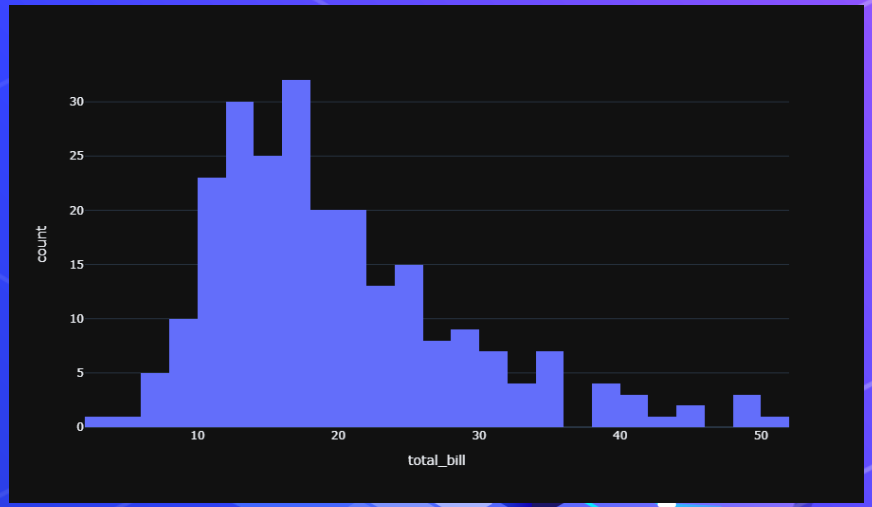
Agenda

- What is Data Visualization
- Plotly and Dash
- **Plotly Themes**
- Distribution Plots
- Categorical Plots
- Matrix Plots
- Customize Plots
- Dash

Plotly Themes

- ggplot2
- seaborn
- simple_white
- plotly
- plotly_white
- plotly_dark
- presentation
- xgridoff
- ygridoff
- gridon

https://plotly.com/python/templates/




```
from numpy import histogram
import plotly.io as pio
pio.templates.default = "plotly_dark"
fig = px.histogram(df, x="total_bill", hover_data=df.columns)
fig.show()
```

Agenda

- What is Data Visualization
- Plotly and Dash
- Plotly Themes
- **Distribution Plots**
- Categorical Plots
- Matrix Plots
- Customize Plots
- Dash

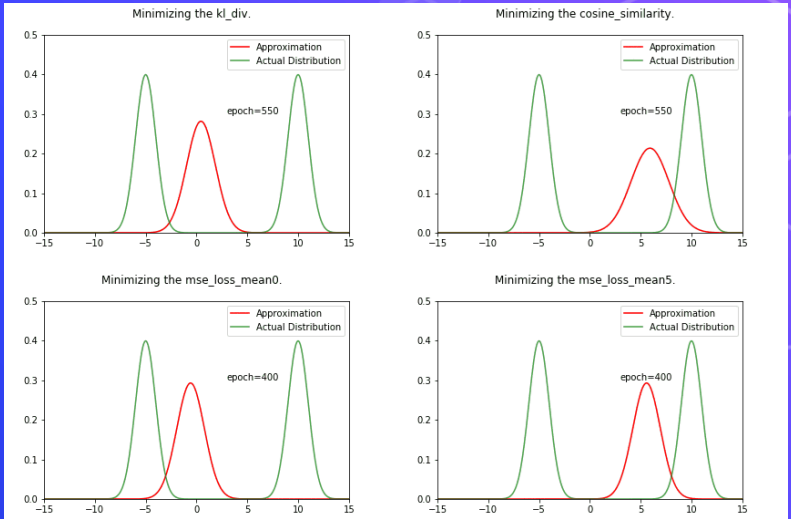
Distribution Plots

We somehow know that there're many types of plots but we're going to discuss about the distribution plot, so **What** is it?

It's a type of plots that shows us the distribution of the numerical values exist in our data

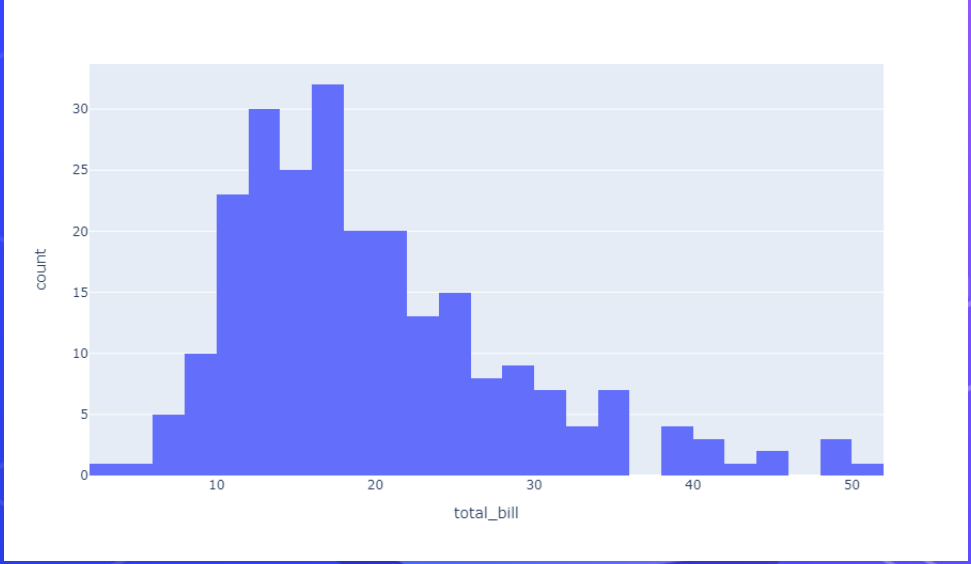
But **WHY !!**

Distribution plots visually assess the distribution of sample data by comparing the empirical distribution of the data with the theoretical values expected from a specified distribution



Distribution Plots

- Univariate Plots
  - **histogram**
  - distplot
- Bivariate Plots
  - scatter
  - scatter_matrix
  - line

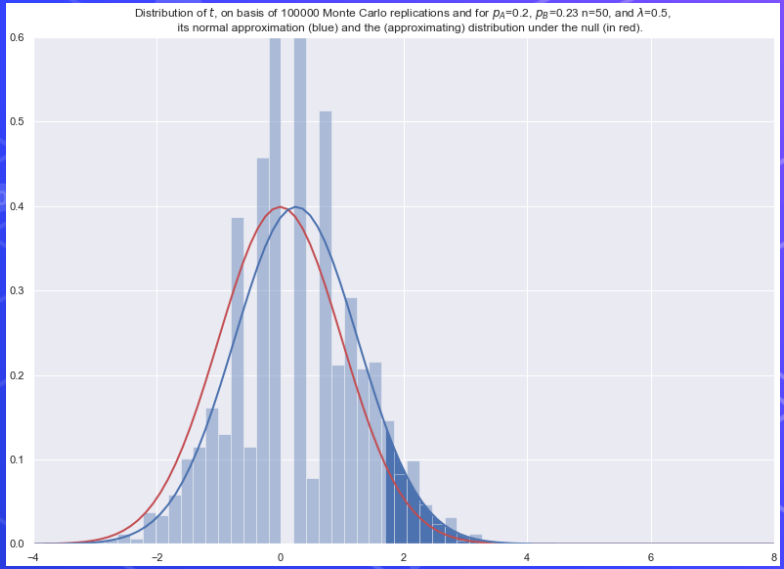



```
fig = px.histogram(df, x="total_bill", hover_data=df.columns)
fig.show()
```

Distribution Plots

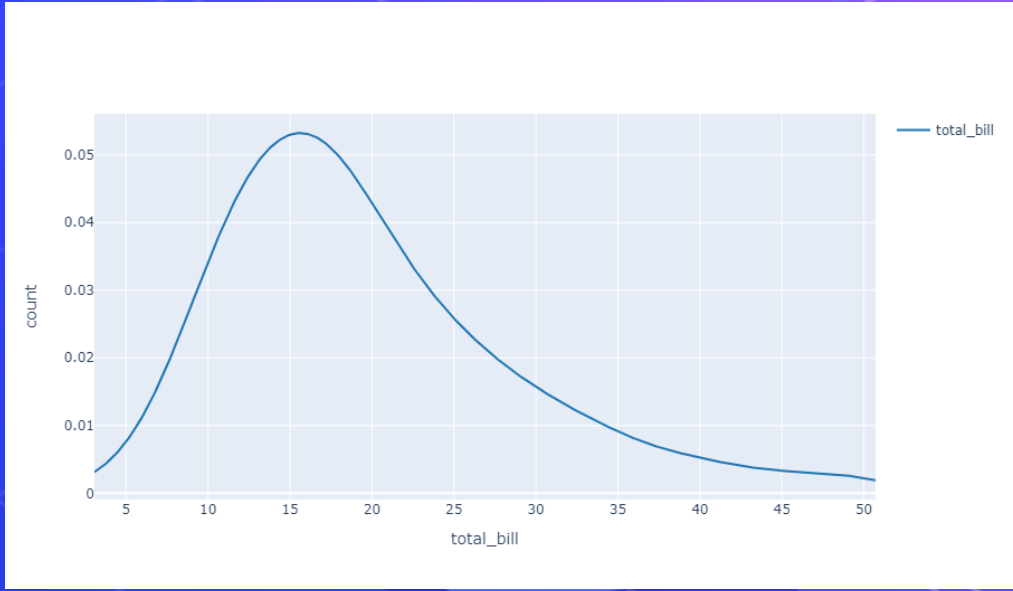
Distplot

Distplot helps us to determine the distribution of the data in the form of histogram bins



Distribution Plots

- Univariate Plots
  - histogram
  - **distplot**
- Bivariate Plots
  - scatter
  - scatter_matrix
  - line



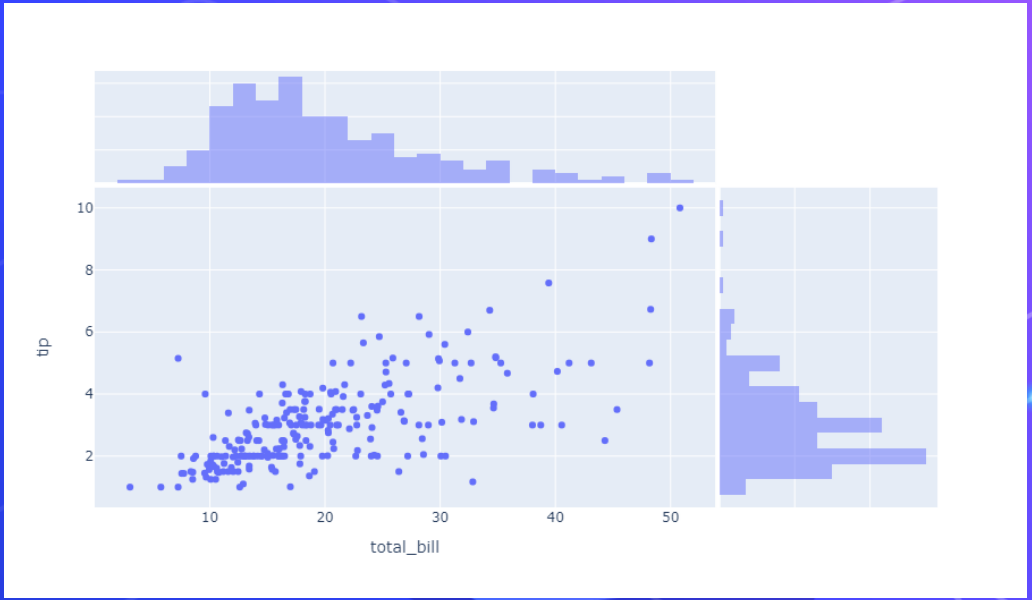

```
hist_data = [df['total_bill'].to_list()]
fig = ff.create_distplot(hist_data, ['total_bill'],
                         show_rug=None, show_hist=False)

fig.update_xaxes(title_text = "total_bill")
fig.update_yaxes(title_text = "count")
fig.show()
```

Task 1

- Read the given dataset “diamond.csv”, perform univariate analysis on the numerical features ‘carat , ‘price’ using the histogram and the density plot

Distribution Plots

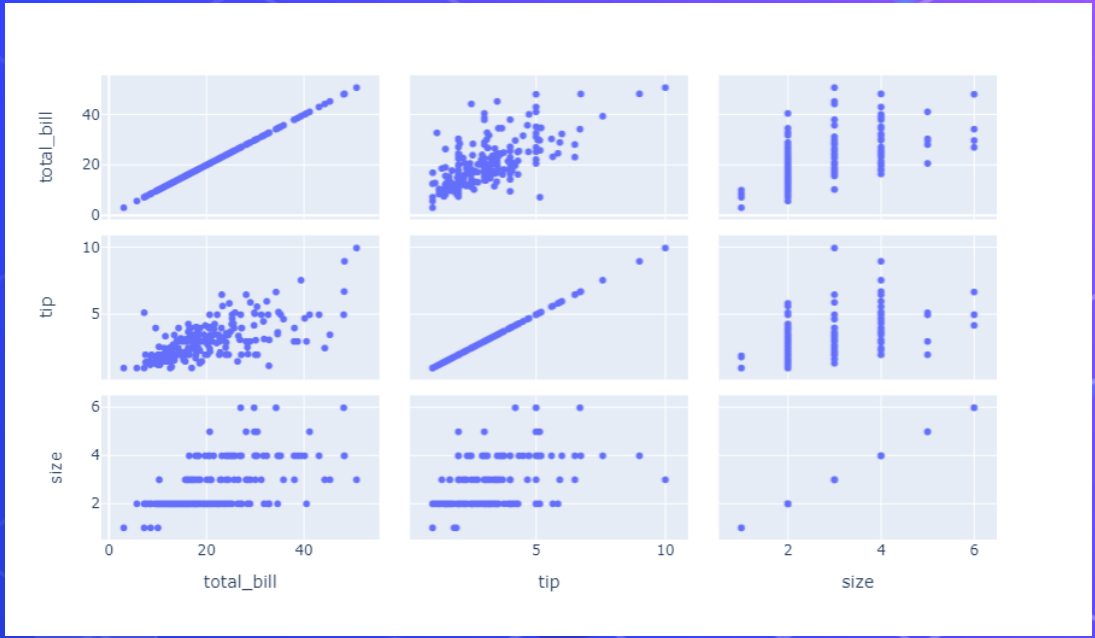
- Univariate Plots
  - histogram
  - distplot
- Bivariate Plots
  - **scatter**
  - scatter_matrix
  - line




```
fig = px.scatter(df, x="total_bill", y="tip", marginal_x="histogram", marginal_y="histogram")
fig.show()
```

Distribution Plots

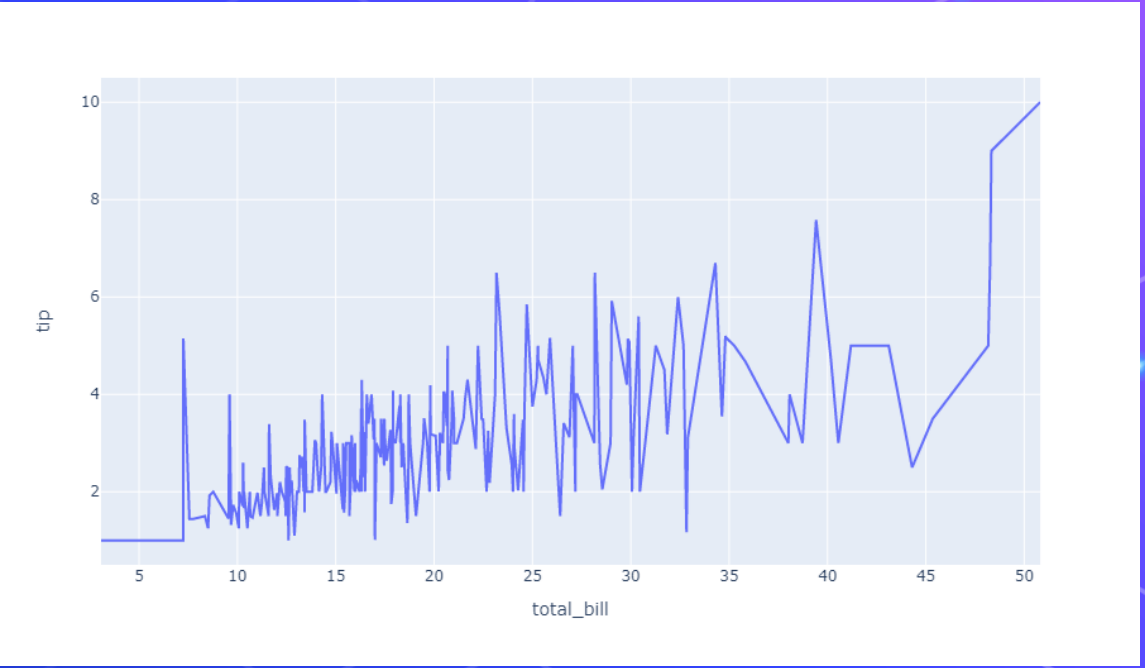
- Univariate Plots
  - `histogram`
  - `distplot`
- Bivariate Plots
  - `scatter`
  - **`scatter_matrix`**
  - `line`




```
fig = px.scatter_matrix(df,
                        dimensions=["total_bill", "tip", "size"])
fig.show()
```

Distribution Plots

- Univariate Plots
  - histogram
  - distplot
- Bivariate Plots
  - scatter
  - scatter_matrix
  - **line**




```
df = df.sort_values(by="total_bill")
fig = px.line(df, x="total_bill", y="tip")
fig.show()
```

Task 2

- Read the given dataset “diamond.csv”, check the relationship between ‘carat and ‘price’ of the diamonds

Quiz !!

What is the best plot(s) for distribution of a univariate numerical value ?

A) distplot

b) kdeplot

c) lineplot

d) scatterplot

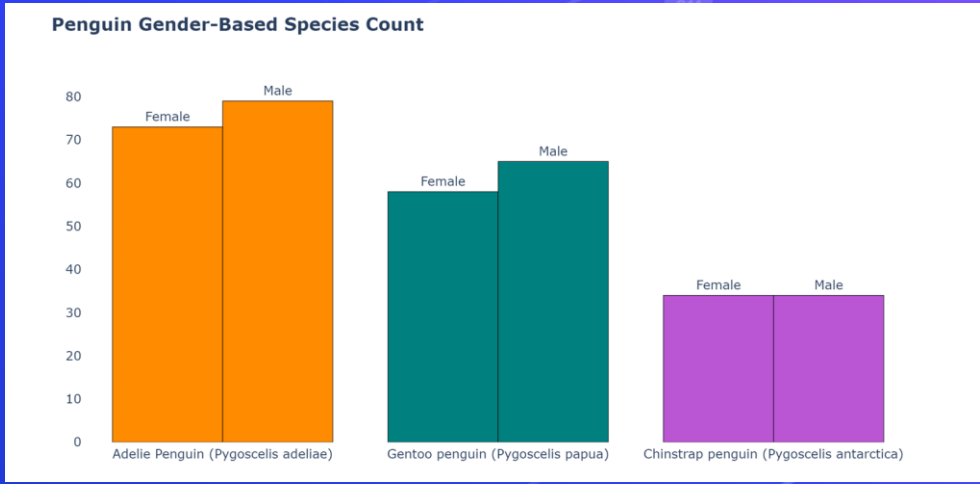
Multiple Choice

Agenda

- What is Data Visualization
- Plotly and Dash
- Plotly Themes
- Distribution Plots
- **Categorical Plots**
- Matrix Plots
- Customize Plots
- Dash

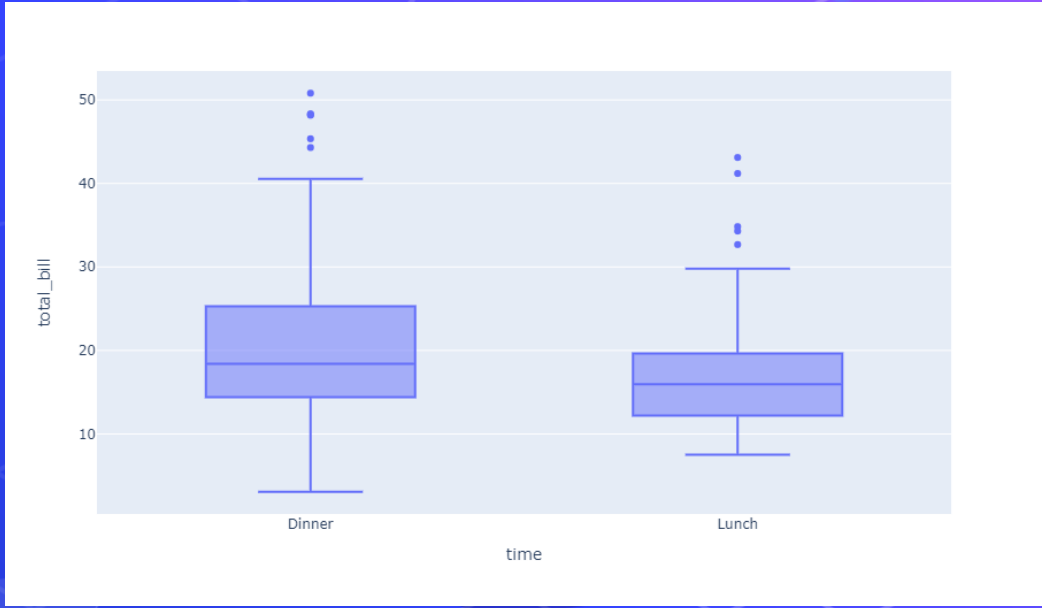
Categorical Plots

- After dealing with numerical values by different types of plots, we need some plots to represent the categorical values to have a complete background about everything in our data, but **What** is that thing ?!
- As a data scientist you'll need to use what is called Categorical Plots



Categorical Plots

- Categorical Distribution Plots
  - **box**
  - violin
- Categorical Scatter Plots
  - strip
- Categorical Estimate Plots
  - bar
  - histogram

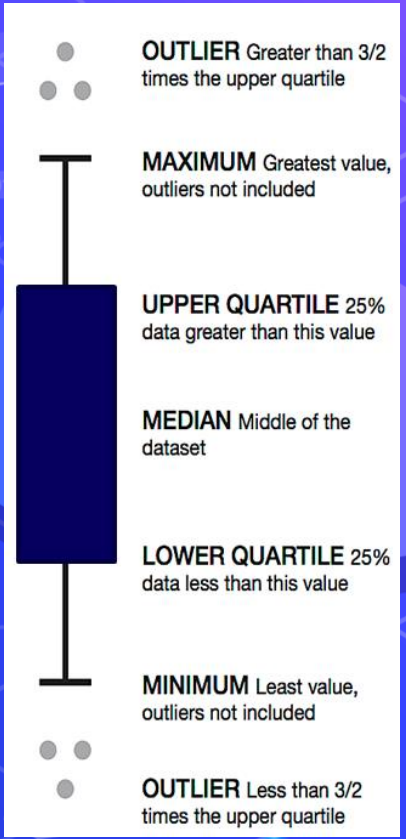



```
fig = px.box(df, x="time", y="total_bill")
fig.show()
```

Categorical Plots

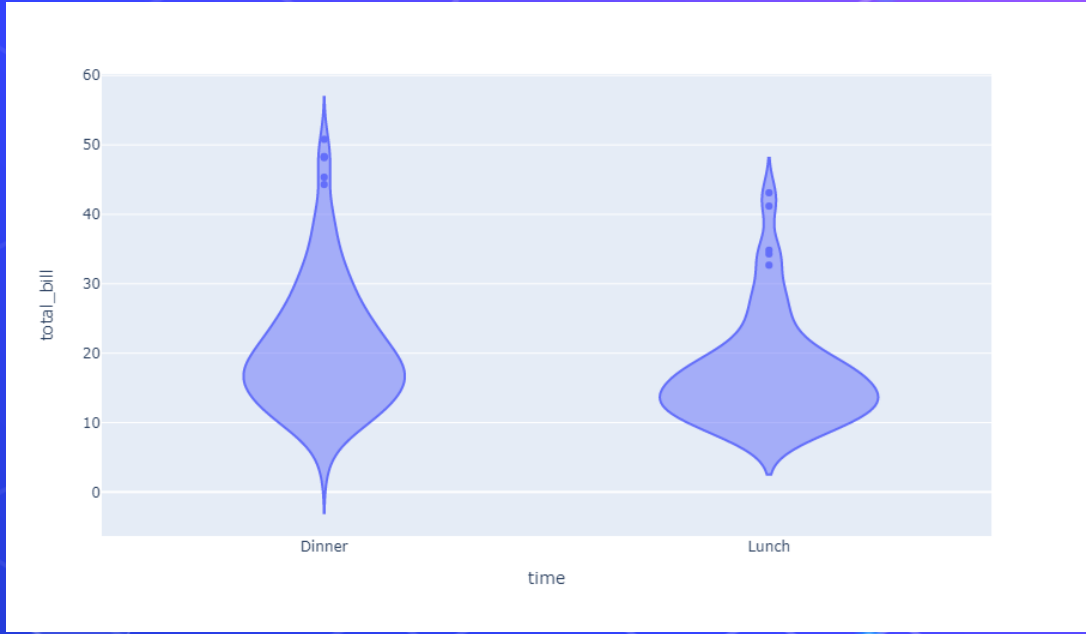
Boxplot

A boxplot is a standardized way of displaying the distribution of data based on a five number summary (“minimum”, first quartile (Q1), median, third quartile (Q3), and “maximum”). ... It can also tell you if your data is symmetrical, how tightly your data is grouped, if and how your data is skewed, and if there’re outliers exist in your data



Categorical Plots

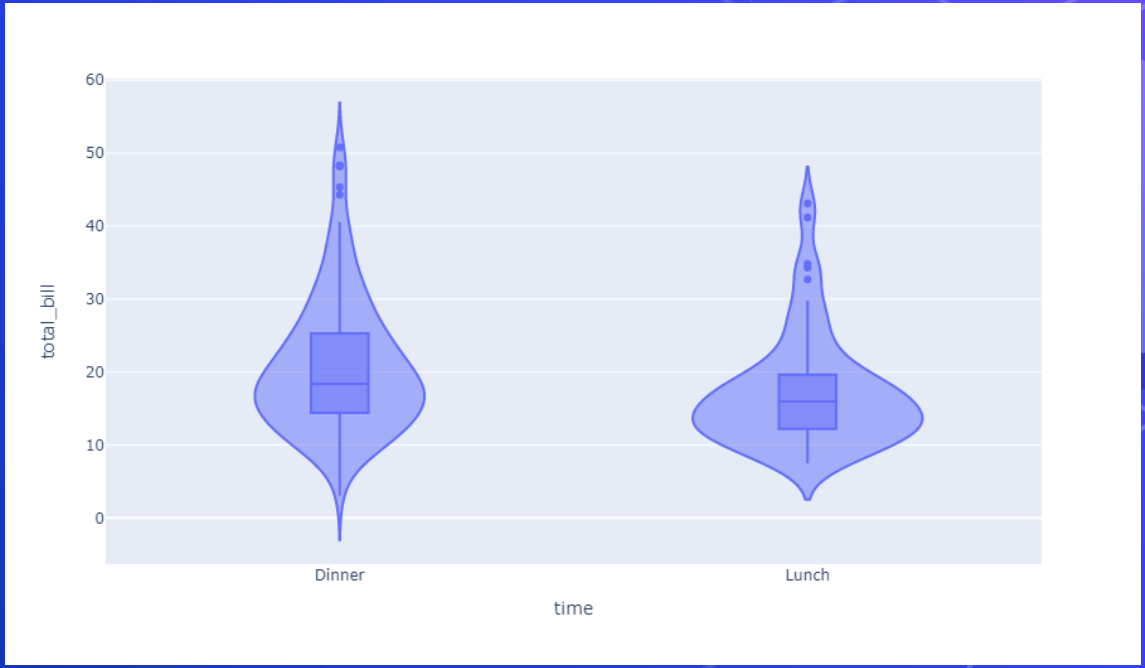
- Categorical Distribution Plots
  - box
  - **violin**
- Categorical Scatter Plots
  - strip
- Categorical Estimate Plots
  - bar
  - histogram




```
fig = px.violin(df, x="time", y="total_bill")
fig.show()
```

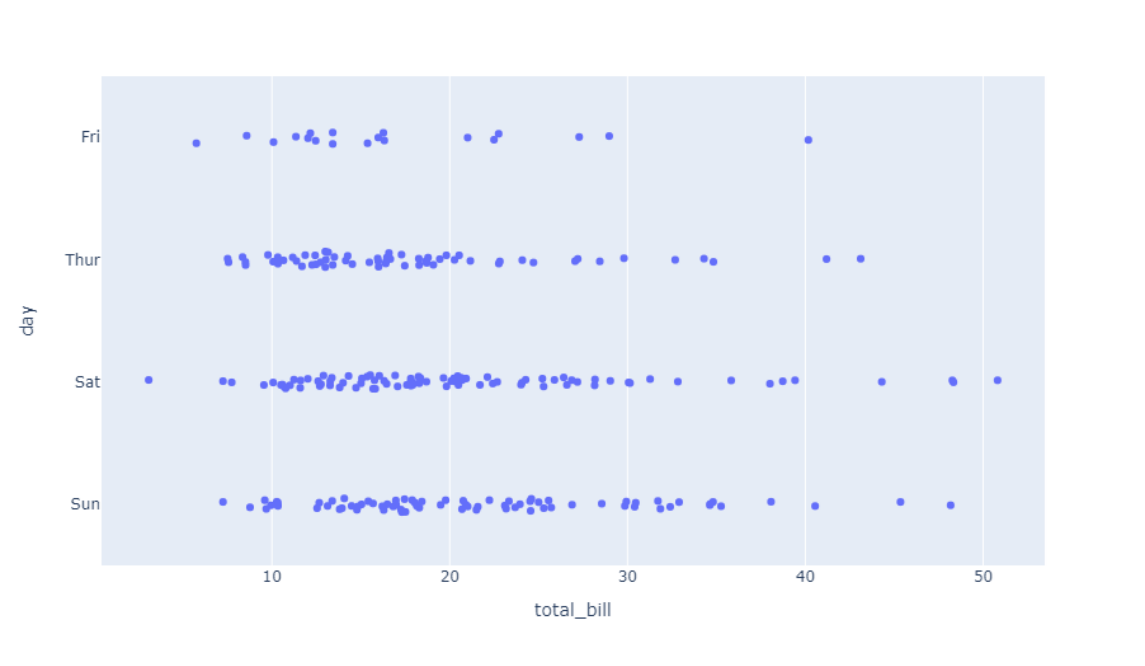
Question !!

Do you notice a difference or match between boxplot and violinplot ?



Categorical Plots

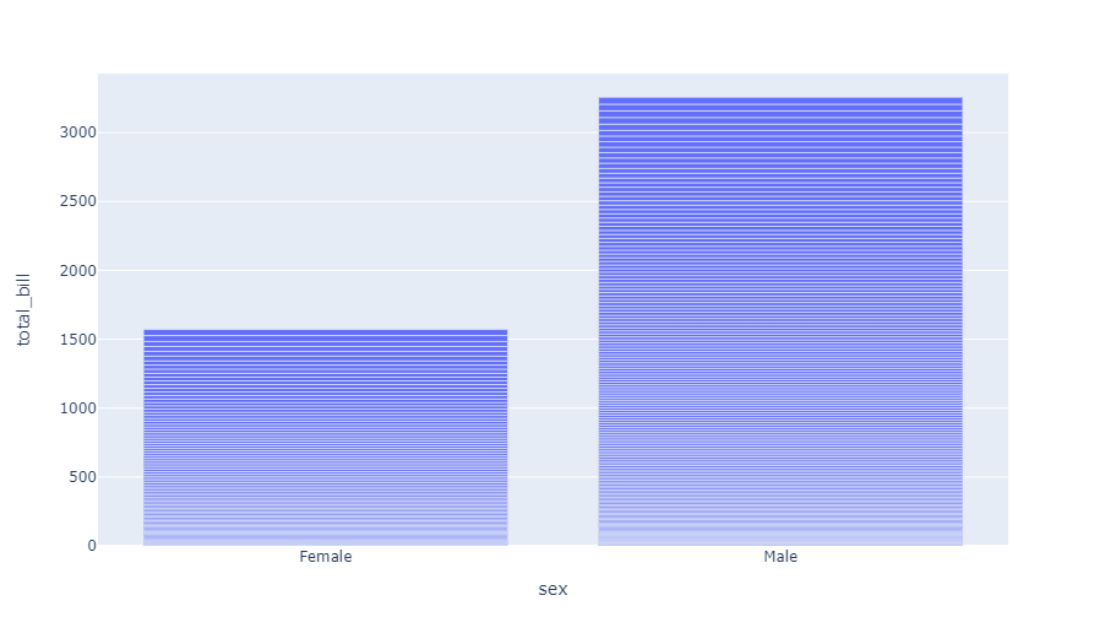
- Categorical Distribution Plots
  - box
  - violin
- Categorical Scatter Plots
  - **strip**
- Categorical Estimate Plots
  - bar
  - histogram




```
fig = px.strip(df, x="total_bill", y="day")
fig.show()
```

Categorical Plots

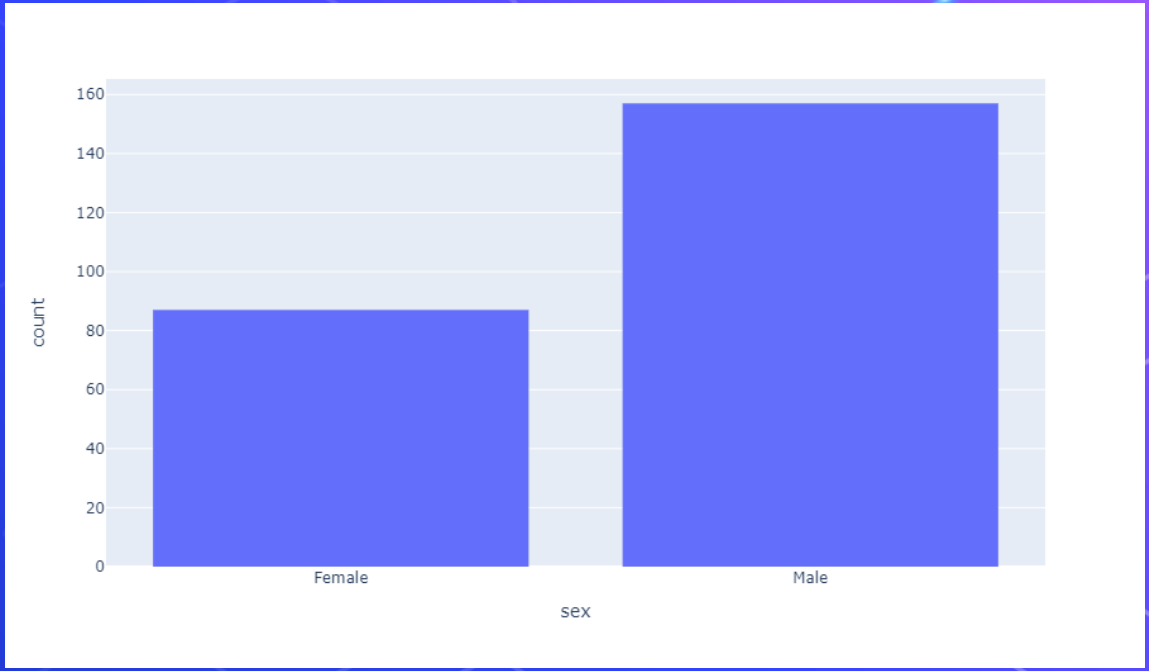
- Categorical Distribution Plots
  - box
  - violin
- Categorical Scatter Plots
  - strip
- Categorical Estimate Plots
  - **bar**
  - histogram




```
px.bar(tips, x="sex", y="total_bill")
```

Categorical Plots

- Categorical Distribution Plots
  - box
  - violin
- Categorical Scatter Plots
  - strip
- Categorical Estimate Plots
  - bar
  - **histogram**




```
fig = px.histogram(df, x="sex")
fig.show()
```

Quiz !!

If you have categorical value, what is the best plot for showing the count of each record ?

A) Violinplot

B) Stripplot

C) Boxplot

D) Histogram

Multiple Choice

Agenda

- What is Data Visualization
- Plotly and Dash
- Plotly Themes
- Distribution Plots
- Categorical Plots
- **Matrix Plots**
- Customize Plots
- Dash

Matrix Plots

heatmap

What is heatmap ?!
Heatmap is a data visualization technique that shows magnitude of a phenomenon as color in two dimensions. The variation in color may be by hue or intensity, giving obvious visual cues to the reader about how the phenomenon is clustered or varies over space

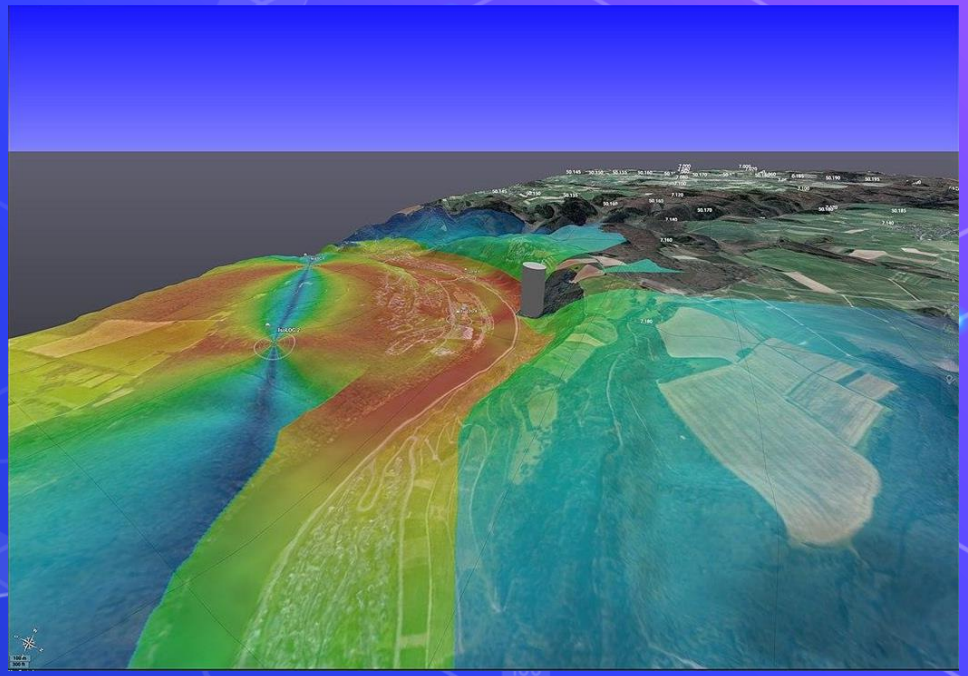
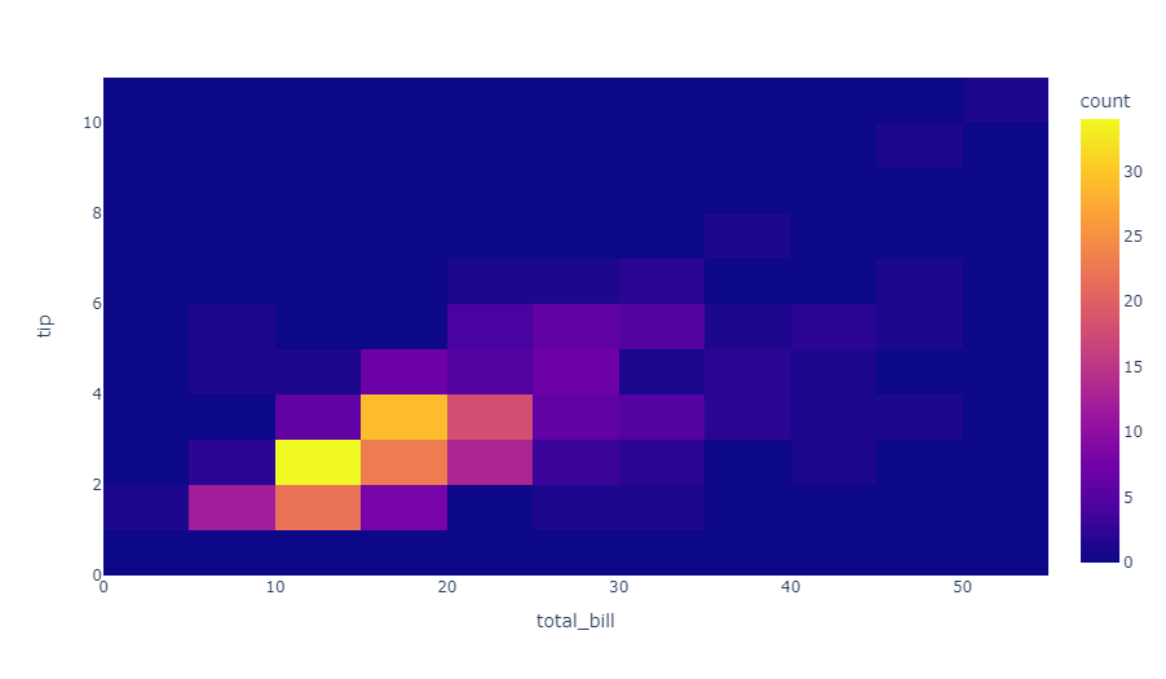


Figure :
A heatmap showing the RF coverage of a drone detection system

Matrix Plots

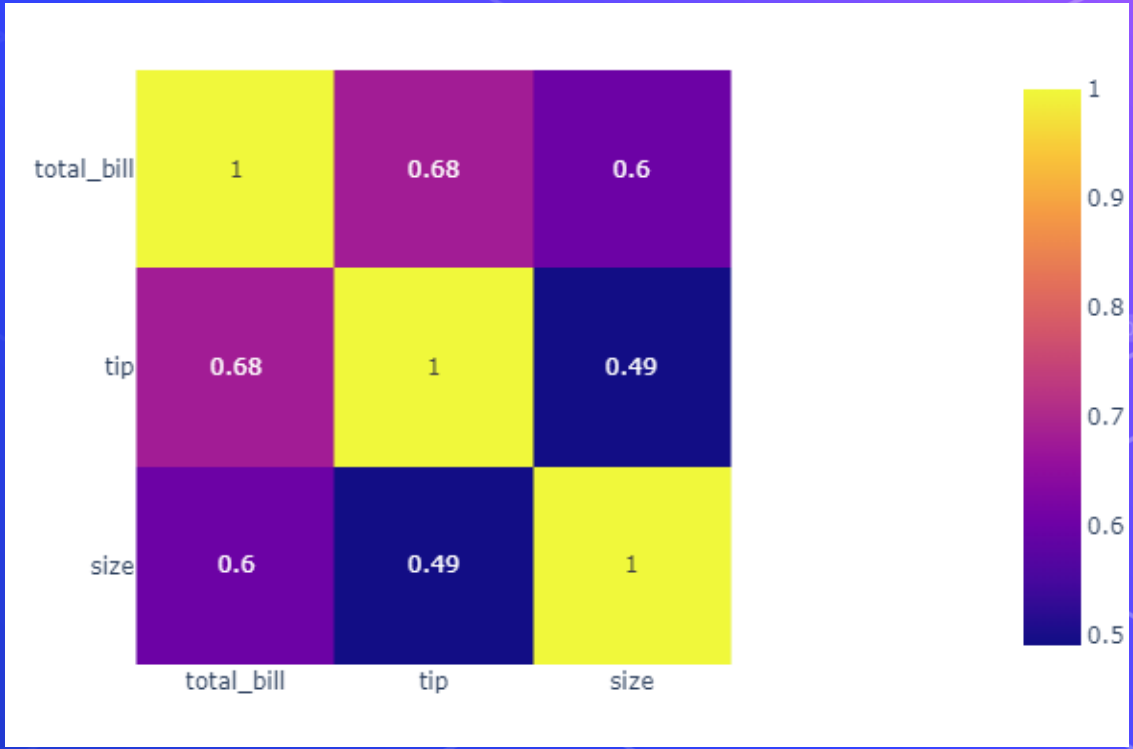
heatmap




```
fig = px.density_heatmap(df, x="total_bill", y="tip")
fig.show()
```

Matrix Plots

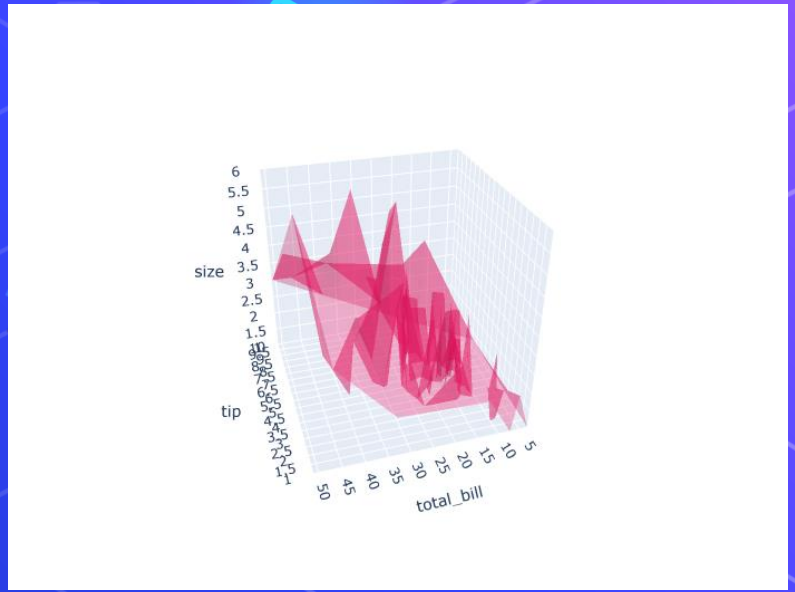
heatmap




```
px.imshow(tips.corr().round(2), text_auto=True)
```

3D Plots

```
fig = go.Figure(data=[go.Mesh3d(x=df['tip'],
                    y=df['total_bill'],
                    z=df['size'],
                    opacity=0.5,
                    color='rgba(244,22,100,0.6)'
                   )])

fig.update_layout(
    scene = dict(
        xaxis = dict(nticks=20, range=[1,10],),
                     yaxis = dict(nticks=20, range=[3,51],),
                     zaxis = dict(nticks=20, range=[1,6],),),
    width=700,
    margin=dict(r=20, l=10, b=10, t=10))

fig.update_layout(scene = dict(
                    xaxis_title='tip',
                    yaxis_title='total_bill',
                    zaxis_title='size'))


fig.show()
```



Agenda

- What is Data Visualization
- Plotly and Dash
- Plotly Themes
- Distribution Plots
- Categorical Plots
- Matrix Plots
- **Customize Plots**
- Dash

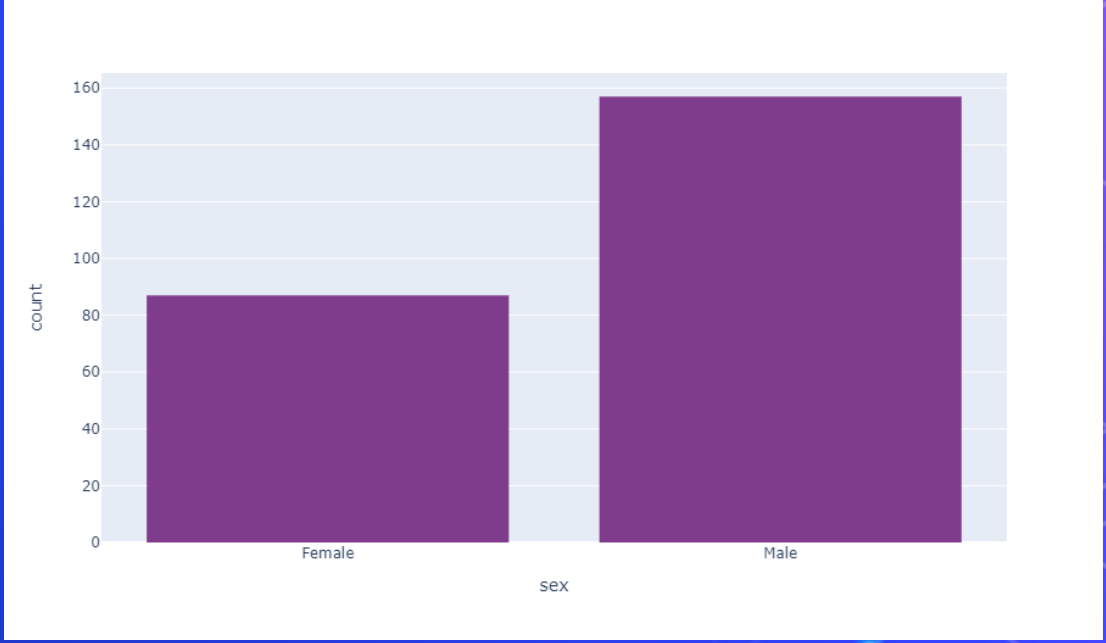
Customize Plots

Change Palette

```
fig = px.colors.qualitative.swatches()
fig.show()
```

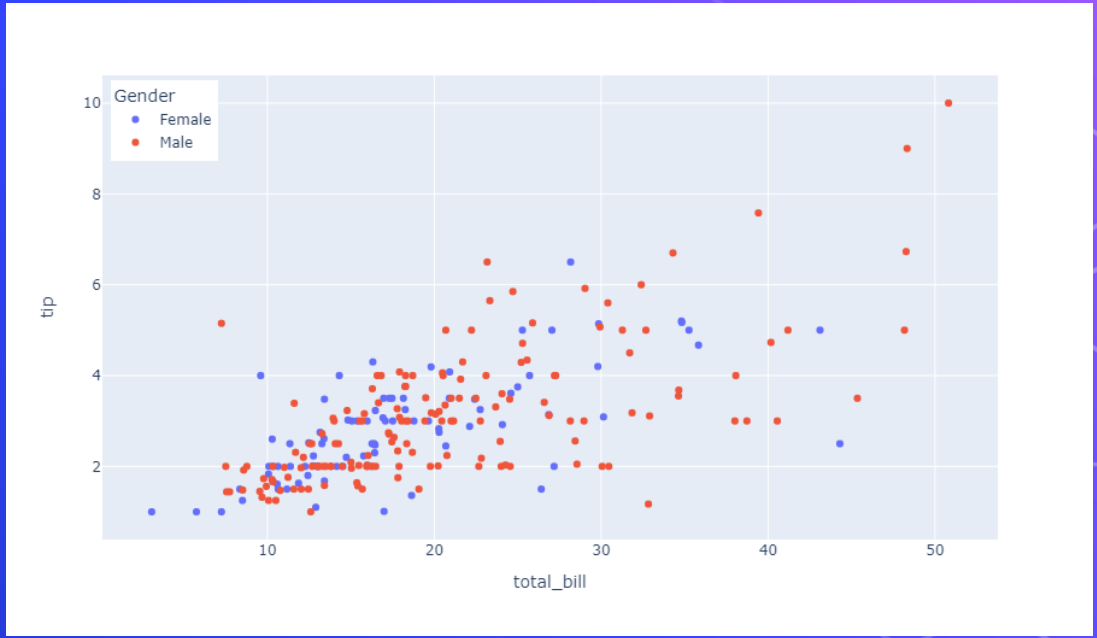
Customize Plots

Change Palette




```
fig = px.histogram(df, x="sex", color_discrete_sequence=px.colors.qualitative.Bold)
fig.show()
```

Customize Plots



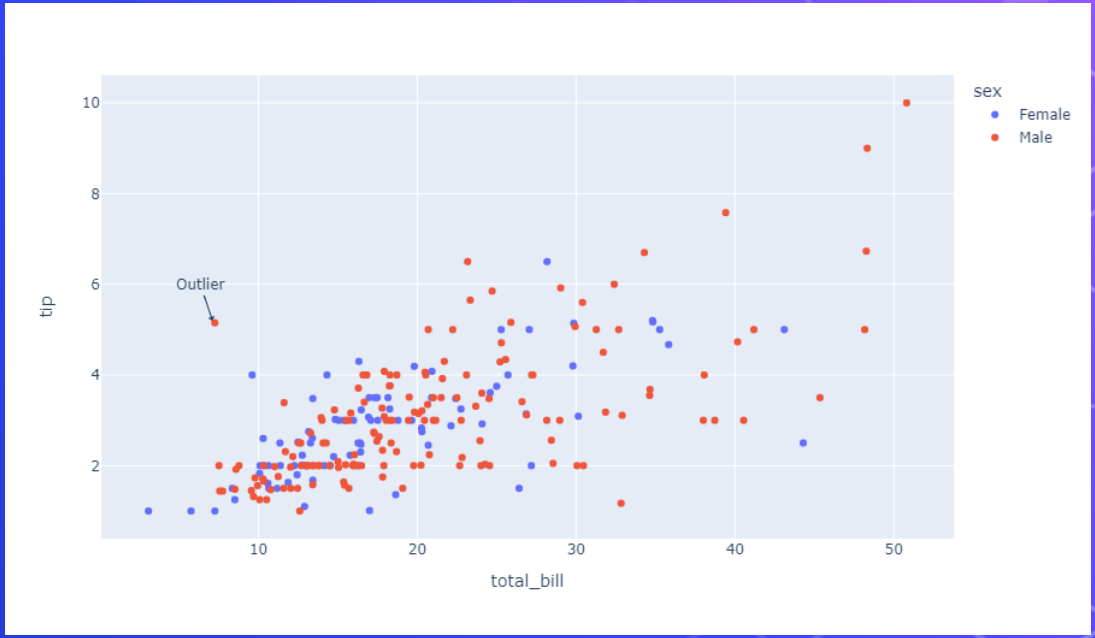
Legend

```
fig = px.scatter(df, x="total_bill", y="tip", color="sex")
fig.update_layout(legend=dict(
    title='Gender',
    yanchor="top",
    y=0.99,
    xanchor="left",
    x=0.01
))
fig.show()
```

Customize Plots

Annotate

- **Text Annotating**




```
fig = px.scatter(df, x="total_bill", y="tip", color="sex")
fig.add_annotation(x=7.1, y=5.2,
            text="Outlier",
            showarrow=True,
            arrowhead=1)
fig.show()
```

Agenda

- What is Data Visualization
- Plotly and Dash
- Plotly Themes
- Distribution Plots
- Categorical Plots
- Matrix Plots
- Customize Plots
- Dash

Agenda

- What is Data Visualization
- Plotly and Dash
- Plotly Themes
- Distribution Plots
- Categorical Plots
- Matrix Plots
- Customize Plots
- **Dash**

Dash App

```python
app = dash.Dash()
app.layout = dcc.Graph(id='examplegraph',figure=bar_fig)
if __name__== '__main__':
  app.run_server(debug=True)
```

```
Dash is running on http://127.0.0.1:8050/

 * Serving Flask app "simple_app" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
```

Dash App in the Browser

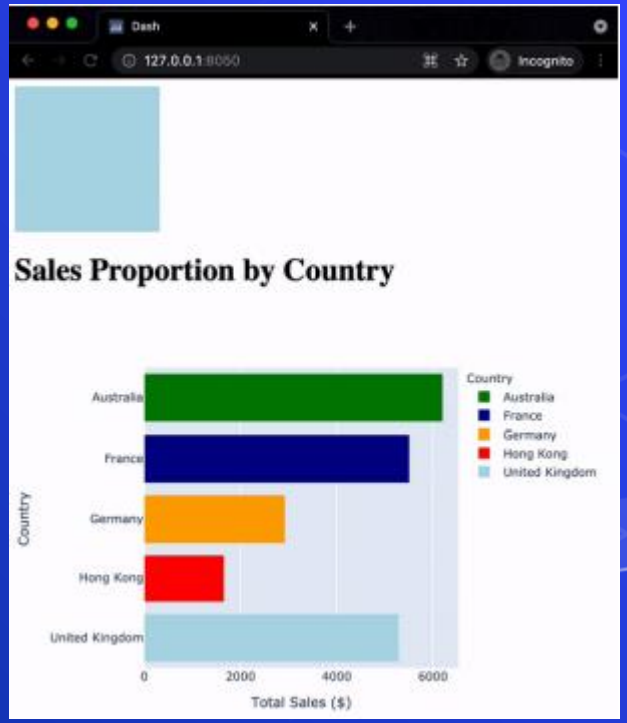
HTML

Dash uses dash _ html _ components to interface between HTML and Python. Two important HTML structures ('tags’):

- Div tags:
  - Important for structuring websites
  - Can have many different-sized divs with different things inside
- H tags:
  - Different sized titles (H1 > H6)

```
<div>
  <div style="background-color: red;
              width:250; height:250;">
  </div>
  <div style="background-color: lightblue;
              width:250; height:250;">
    <h1>This box</h1>
    <h2>Another Title</h2>
  </div>
</div>
```

HTML




```
app.layout = html.Div(
children=[
 html.Div( style={
        'width':150,
        'height':150,
        'background-color':'lightblue'}),
 html.H1("Sales Proportion by Country"),
 dcc.Graph(id='bar_graph',figure=bar_fig_country)
    ]
 )
```

Callbacks in Dash

Functionality triggered by interaction

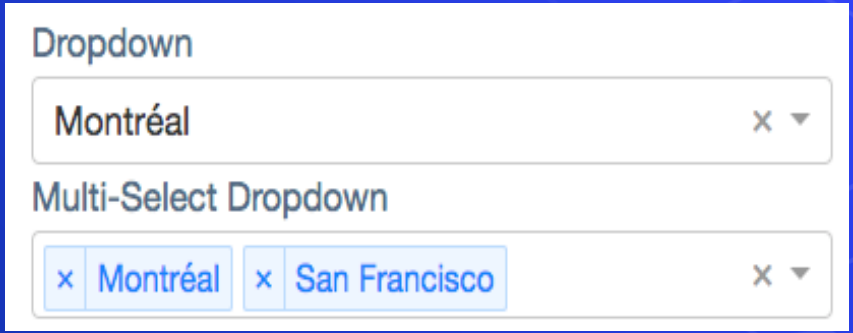
- A user interacts with an element
- A Python function is triggered
- Something is changed

Why? Enhances interactivity

```python
from dash.dependencies import Input, Output
@app.callback(
    Output(component_id='my_plot',
           component_property='figure'),
    Input(component_id='my_input',
          component_property='value'))
def some_function(data):
    # Subset Data
    # Recreate Figure
    return fig
```

Dropdowns in Dash

List of label-value dictionaries




```
dcc.Dropdown(id='title_dd',
             options=[{'label':'Title 1'
                      ,'value':'Title 1'},
                      {'label':'Title 2'
                      ,'value':'Title 2'}]),
```

Reminder: Can link to callback Update plots or components

Sliders in Dash

Slider: drag and move for a single value

Range Slider: drag and move for two values


```
dcc.Slider(
        min=10,
        max=50,
        value=45,
        step=5,
        vertical=False
)
```

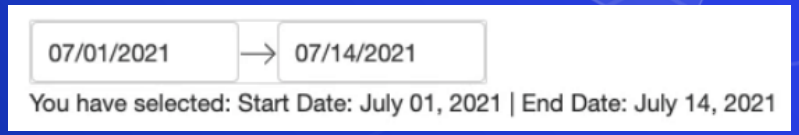
Reminder: Can link to callback Update plots or components

Date Pickers in Dash

DatePickerSingle: Select a single date


```
dcc.DatePickerSingle(
    date=date(2021, 7, 1),
    initial_visible_month=datetime.now(),
)
```

DatePickerRange: Set an initial start _ date and end _ date



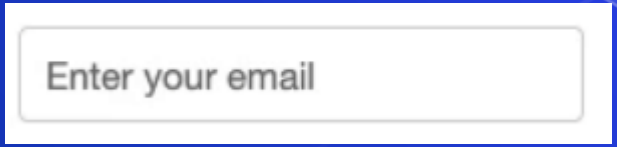

```
dcc.DatePickerRange(
    initial_visible_month=datetime.now(),
    start_date=date(2021, 7, 1),
    end_date=date(2021, 7, 14),
)
```

User Input in Dash

A user input is a dash _ core _ components
Input type (dcc.Input)


```
dcc.Input(
    id='my_input',
    type='text',
    placeholder="Enter your text"
)
```

Dash offers useful input types: 'text' , 'number' , 'password' , 'email'

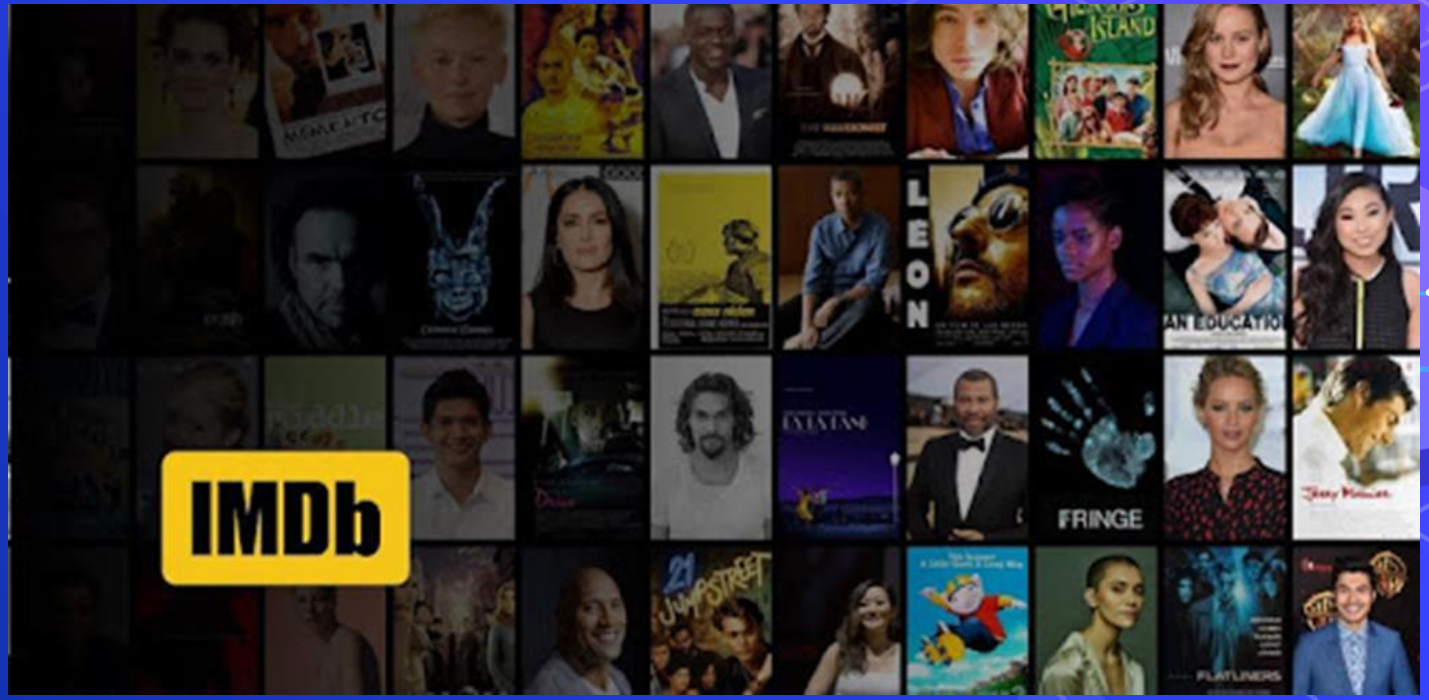



```
dcc.Input(
    id='my_input',
    type='email',
    placeholder="Enter your email"
)
```

Task 3

- Build interactive graphs using dash and plotly on tips dataset

Project 14 - Analyze IMDB Movies dataset cont..



Project 15 - Analyze Shopping cart dataset cont…

Project 16 - Analyze FIFA dataset cont….

Questions ?!

Thanks!

>_ Live long and prosper