Assignment - 4 Key

1. Explain the structure of a Plotly plot in terms of its components?

A Plotly plot consists of various components, including data, layout, and optionally, configuration settings. The key components are:

Data: This includes the actual data points or traces that you want to visualize, such as scatter plots, lines, bars, etc.

Layout: This component defines the overall appearance and structure of the plot, including axis labels, titles, annotations, and other visual elements.

Configuration (optional): Additional settings for customizing the plot's behaviour, such as responsiveness, display mode, and more.

Plotly constructs plots using graph objects, which are Python classes representing different parts of the plot, and dictionaries, which contain key-value pairs specifying plot attributes. Graph objects provide a more structured approach to creating plots, while dictionaries offer more flexibility and control.

Example:

```
python
Copy code
import plotly.graph_objects as go

# Using graph objects
fig = go.Figure(data=go.Scatter(x=[1, 2, 3], y=[4, 1, 2]), layout=go.Layout(title='Plot Title'))

# Using dictionaries
fig_dict = {
```

360 DigiTMG Digital Transformation | Management | Governance

```
'data': [{'x': [1, 2, 3], 'y': [4, 1, 2]}],
'layout': {'title': 'Plot Title'}
}
```

2. Compare and contrast online and offline plotting in Plotly?

Online Plotting: In online plotting, plots are rendered and stored on the Plotly cloud, allowing for easy sharing and collaboration. This approach is suitable when you need to share plots with others or embed them in web applications.

Offline Plotting: Offline plotting generates plots locally without requiring an internet connection. This is useful for creating plots in environments where internet access is limited or when working with sensitive data that shouldn't be shared online.

```
Python Copy code
# Online plotting
import plotly.graph_objects as go
fig = go.Figure(data=go.Scatter(x=[1, 2, 3], y=[4, 1, 2]))
fig.show()

# Offline plotting
import plotly.offline as py
py.init_notebook_mode()  # Run this in Jupyter Notebook
py.iplot(fig)
```

3. Explore the concept of updating plots in Plotly?

Updating plots in Plotly involves adding new traces or updating existing traces within the same figure. This can be achieved by modifying the data attribute of the figure object.

```
Python Copy code
import plotly.graph_objects as go

# Adding a new trace
fig.add_trace(go.Scatter(x=[1, 2, 3], y=[4, 1, 2]))

# Updating an existing trace
fig.update_traces(marker=dict(colour='red'))
```

4. Describe the method of creating subplots in Plotly?

Subplots allow you to display multiple plots within the same figure, enabling comparison and visualization of related data simultaneously. They are beneficial for presenting different aspects of data or comparing multiple datasets.

```
Python
Copy code
from plotly.subplots import make_subplots

fig = make_subplots(rows=2, cols=2)
fig.add_trace(go.Scatter(x=[1, 2, 3], y=[4, 1, 2]), row=1, col=1)
fig.add_trace(go.Bar(x=[1, 2, 3], y=[4, 1, 2]), row=1, col=2)
fig.add_trace(go.Box(y=[1, 2, 3]), row=2, col=1)
```

5. Investigate the implementation of interactive features in Plotly?

Interactive features like dropdown menus and Dash interactivity enhance user engagement and exploration of data. Dropdown menus allow users to dynamically select data or change plot configurations, while Dash interactivity enables building interactive web applications with Plotly plots.

Python Copy code

```
import plotly.graph_objects as go
from dash import Dash, dcc, html

app = Dash(__name__)

app.layout = html.Div([
dcc.Graph(
id='plot',
figure={
'data': [
go.Scatter(x=[1, 2, 3], y=[4, 1, 2], mode='markers', marker={'size': 20})
],
'layout': go.Layout(title='Interactive Plot')
}
),
dcc.Dropdown(
id='dropdown',
options=[
{'label': 'Option 1', 'value': 'option1'},
{'label': 'Option 2', 'value': 'option2'}
],
value='option1'
)
])
```

```
if __name__ == '__main__':
app.run_server(debug=True)
```