**Python Project**

**Interactive Data Visualization Dashboard**

**By:** S. Harsshita

**Offer No:** U242103274

**Task Submission No:** UT240321487

**Introduction**:

The Interactive Data Visualization Dashboard project aims to provide a user-friendly interface for exploring and analysing data from multiple sources. The dashboard allows users to interactively select data sources, attributes for visualization, time periods, and visualize the data in various chart types such as scatter plots, bar charts, and pie charts. The project leverages the Dash framework in Python along with Plotly for interactive plotting capabilities.

**Problem Statement:**

A company has collected data from various sources and needs an interactive data visualization dashboard for better decision-making. The current method of using static reports is limiting their ability to gain insights.

**Objectives**:

1. Develop a data visualization dashboard using Python.
2. Integrate multiple data sources into the dashboard.
3. Provide interactive features for exploration.

**Project Overview:**

The project consists of the following key components:

* **Data Generation:** Sample data for three different data sources is generated using NumPy and Pandas libraries. Each data source represents a fictional business entity with attributes such as sales, profit, expenses, revenue, orders, and customers.
* **Dashboard Interface:** The dashboard interface is built using Dash, a Python framework for building analytical web applications. It includes tabs for selecting data sources, dropdown menus for selecting attributes, a range slider for selecting time periods, and graph components for displaying visualizations.
* **Visualizations**: The dashboard supports three types of visualizations: scatter plots, bar charts, and pie charts. These visualizations are dynamically updated based on user selections.
* **Callbacks**: Callback functions are implemented to update the visualizations in real-time based on user inputs. These callbacks handle data retrieval, filtering, and updating the graph components.

**Features:**

* **Multiple Data Sources:** The dashboard supports visualization of data from three different sources, allowing users to compare metrics across different business entities.
* **Interactive Selection:** Users can interactively select attributes for visualization, time periods, and data sources using dropdown menus and sliders.
* **Dynamic Visualization:** Visualizations are dynamically updated based on user selections, providing real-time insights into the data.
* **Chart Types:** The dashboard supports scatter plots, bar charts, and pie charts, offering flexibility in data representation.

**Implementation:**

**Data Generation:**

* Sample data for three different data sources (Starbucks, Lassi Corner, Village Rose Milk) is generated using NumPy and Pandas libraries.
* The `np.random.seed(42)` ensures reproducibility of random data generation.
* Each data source consists of attributes such as 'Year', 'Sales', 'Profit', 'Expenses', 'Revenue', 'Orders', and 'Customers'.

**Dashboard Interface:**

* The dashboard interface is built using the Dash framework, which provides a Python interface for creating web applications.
* Tabs are created using `dcc.Tabs`, allowing users to switch between different data sources.
* Dropdown menus (`dcc.Dropdown`) are used for selecting X-axis and Y-axis attributes.
* A range slider (`dcc.RangeSlider`) enables users to select a time period for data visualization.
* Graph components (`dcc.Graph`) are used to display visualizations such as scatter plots, bar charts, and pie charts.

**Visualization:**

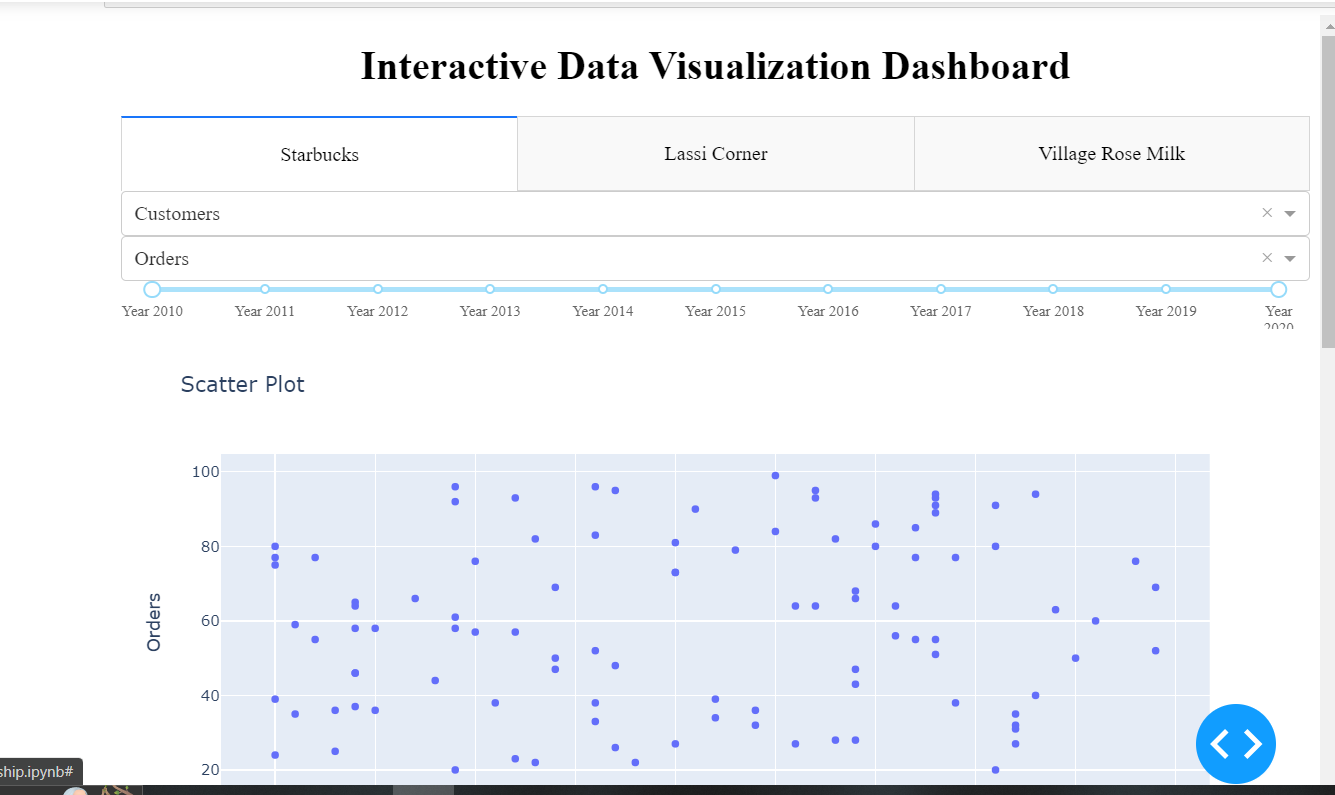
* Plotly graph objects (`go.Scatter`, `go.Bar`, `go.Pie`) are used to create interactive visualizations.
* For the scatter plot, `go.Scatter` is used to plot data points, while for bar charts, `go.Bar` is used to create bar plots.
* Pie charts are created using `go.Pie` to represent data as slices of a pie.

**Callbacks:**

* Callback functions are defined using `@app.callback` decorator to update the visualizations based on user inputs.
* The `Output` parameter specifies the component to be updated, while the `Input` parameters define the components triggering the callback.
* Callback functions retrieve data from the selected data source, filter it based on user selections, and update the graph components accordingly.

**Result:**

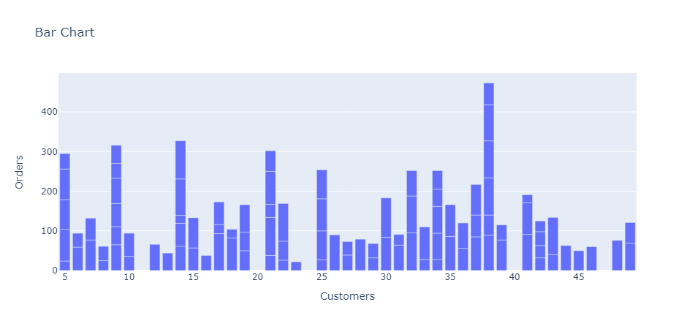
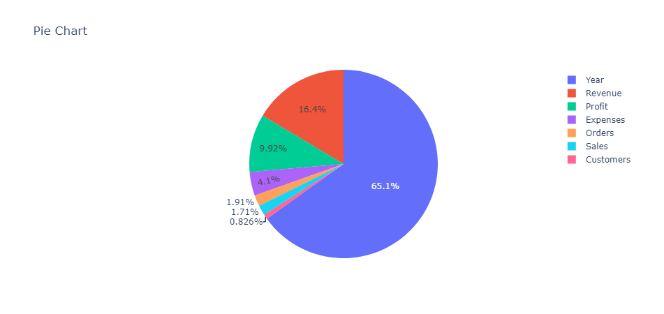
The implemented data visualization dashboard provides users with an interactive platform for exploring and analyzing data from multiple sources. Figure 1 illustrates the dashboard interface, where users can seamlessly switch between different data sources using tabs. From Figure 1, we can see that the interface allows users to select specific X-axis and Y-axis attributes for visualization, facilitating customized analysis based on their preferences.



**Figure 1**

The range slider feature enables users to specify a time period of interest, allowing them to focus on specific data subsets for analysis. By interacting with the dropdown menus and sliders, users can dynamically update the visualizations displayed on the dashboard, gaining insights into various metrics such as sales, profit, expenses, revenue, orders, and customers.

The visualizations include scatter plots, bar charts, and pie charts, providing different perspectives on the data. For instance, scatter plots are useful for visualizing relationships between two numerical variables, while bar charts help in comparing categorical variables across different categories. Figure 2 showcases a sample scatter plot and bar chart and pie chart generated from the dashboard.





**Figure 2**

Overall, the data visualization dashboard offers a user-friendly and intuitive interface for exploring and interpreting complex datasets, empowering users to make informed decisions based on data-driven insights.

**Conclusion:**

The Interactive Data Visualization Dashboard project provides a powerful tool for exploring and analysing data from multiple sources in an interactive manner. By leveraging Python libraries such as Dash and Plotly, users can gain valuable insights into business metrics and make informed decisions. The project demonstrates the potential of interactive data visualization in facilitating data-driven decision-making processes.

**References:**

* + Dash Documentation: https://dash.plotly.com/
  + Plotly Documentation: https://plotly.com/python/