**Python Plotly tutorial**

**Python Plotly** Library is an open-source library that can be used for data visualization and understanding data simply and easily. Plotly supports various types of plots like line charts, scatter plots, histograms, cox plots, etc. So you all must be wondering why Plotly over other visualization tools or libraries? Here’s the answer –

* Plotly has hover tool capabilities that allow us to detect any outliers or anomalies in a large number of data points.
* It is visually attractive that can be accepted by a wide range of audiences.
* It allows us for the endless customization of our graphs that makes our plot more meaningful and understandable for others.

This tutorial aims at providing you the insight about Plotly with the help of the huge dataset explaining the Plotly from basics to advance and covering all the popularly used charts.

**Installation**

Plotly does not come built-in with Python. To install it type the below command in the terminal.

pip install plotly

This may take some time as it will install the dependencies as well.

**Package Structure of Plotly**

There are three main modules in Plotly. They are:

* plotly.plotly
* plotly.graph.objects
* plotly.tools

**plotly.plotly** acts as the interface between the local machine and Plotly. It contains functions that require a response from Plotly’s server.

**plotly.graph\_objects** module contains the objects (Figure, layout, data, and the definition of the plots like scatter plot, line chart) that are responsible for creating the plots.  The Figure can be represented either as dict or instances of **plotly.graph\_objects.Figure**and these are serialized as JSON before it gets passed to plotly.js. Consider the below example for better understanding.

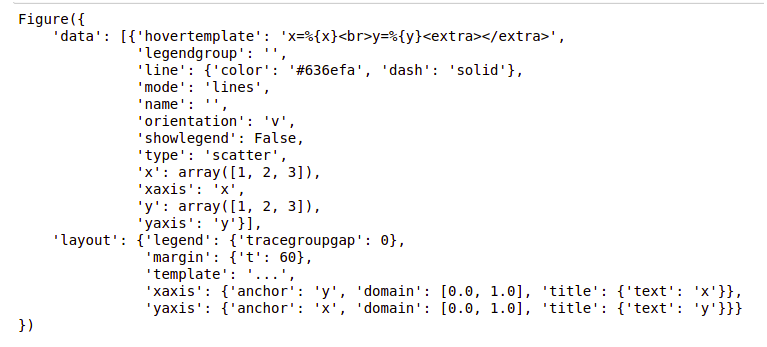
**Note:**plotly.express module can create the entire Figure at once. It uses the graph\_objects internally and returns the graph\_objects.Figure instance.

**Example:**

* Python3

|  |
| --- |
| import plotly.express as px      # Creating the Figure instance  fig = px.line(x=[1,2, 3], y=[1, 2, 3])    # printing the figure instance  print(fig) |

**Output:**

****

Figures are represented as trees where the root node has three top layer attributes – **data, layout, and frames**and the named nodes called ‘attributes’. Consider the above example, **layout.legend**is a nested dictionary where the legend is the key inside the dictionary whose value is also a dictionary.

**plotly.tools**module contains various tools in the forms of the functions that can enhance the Plotly experience.

**Getting Started**

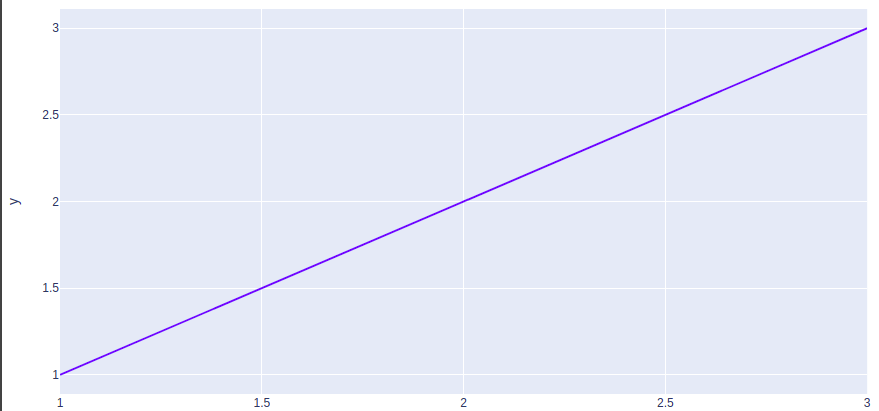
After learning the installation and basic structure of the Plotly, let’s create a simple plot using the pre-defined data sets defined by the plotly.

**Example:**

* Python3

|  |
| --- |
| import plotly.express as px      # Creating the Figure instance  fig = px.line(x=[1, 2, 3], y=[1, 2, 3])    # showing the plot  fig.show() |

**Output:**



In the above example, the plotly.express module is imported which returns the Figure instance. We have created a simple line chart by passing the x, y coordinates of the points to be plotted.

Creating Different Types of Charts

With plotly we can create more than 40 charts and every plot can be created using the plotly.express and plotly.graph\_objects class. Let’s see some commonly used charts with the help of Plotly.

**Line Chart**

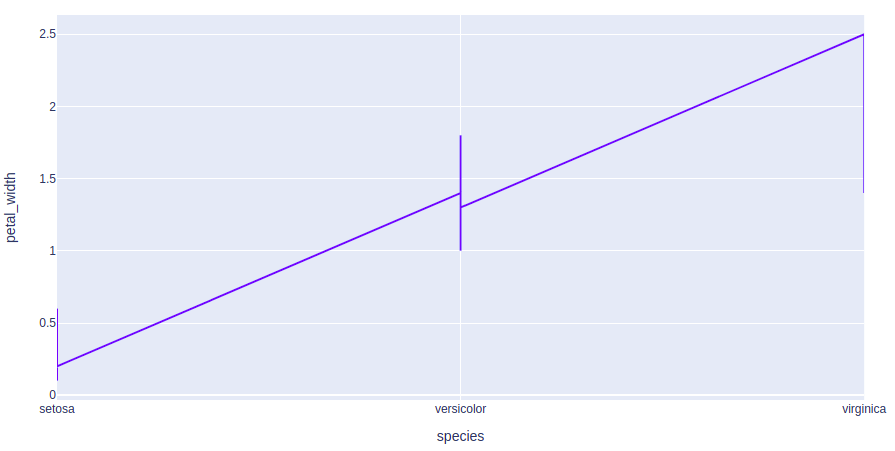
[**Line plot**](https://www.geeksforgeeks.org/line-chart-using-plotly-in-python/)in Plotly is much accessible and illustrious annexation to plotly which manage a variety of types of data and assemble easy-to-style statistic. With **px.line** each data position is represented as a vertex  (which location is given by the x and y columns) of a polyline mark in 2D space.

**Example:**

* Python3

|  |
| --- |
| import plotly.express as px    # using the iris dataset  df = px.data.iris()    # plotting the line chart  fig = px.line(df, x="species", y="petal\_width")    # showing the plot  fig.show() |

**Output:**



**Bar Chart**

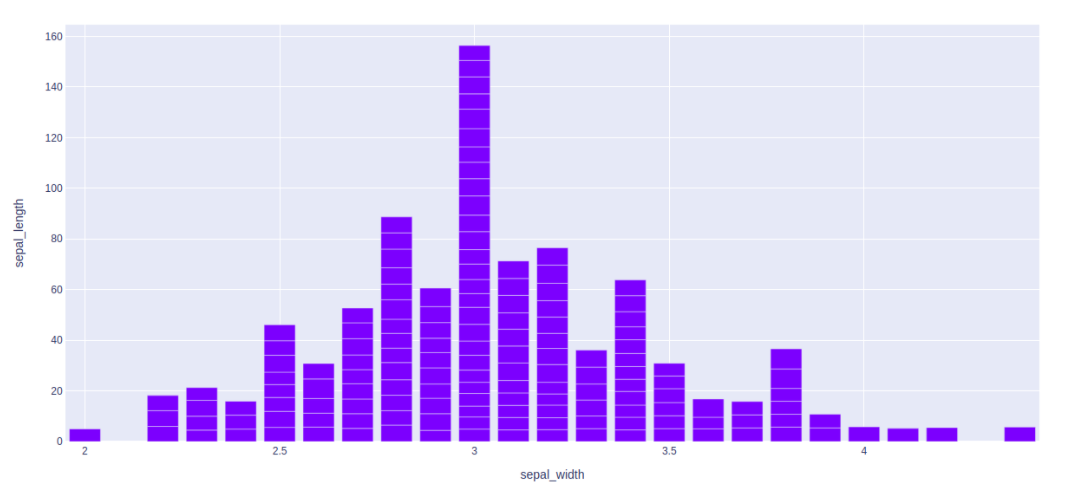
A [**bar chart**](https://www.geeksforgeeks.org/bar-chart-using-plotly-in-python/) is a pictorial representation of data that presents categorical data with rectangular bars with heights or lengths proportional to the values that they represent. In other words, it is the pictorial representation of dataset. These data sets contain the numerical values of variables that represent the length or height.

**Example:**

* Python3

|  |
| --- |
| import plotly.express as px    # using the iris dataset  df = px.data.iris()    # plotting the bar chart  fig = px.bar(df, x="sepal\_width", y="sepal\_length")    # showing the plot  fig.show() |

**Output:**



**Scatter Plot and Bubble charts**

A [**scatter plot**](https://www.geeksforgeeks.org/scatter-plot-using-plotly-in-python/) is a set of dotted points to represent individual pieces of data in the horizontal and vertical axis. A graph in which the values of two variables are plotted along X-axis and Y-axis, the pattern of the resulting points reveals a correlation between them.

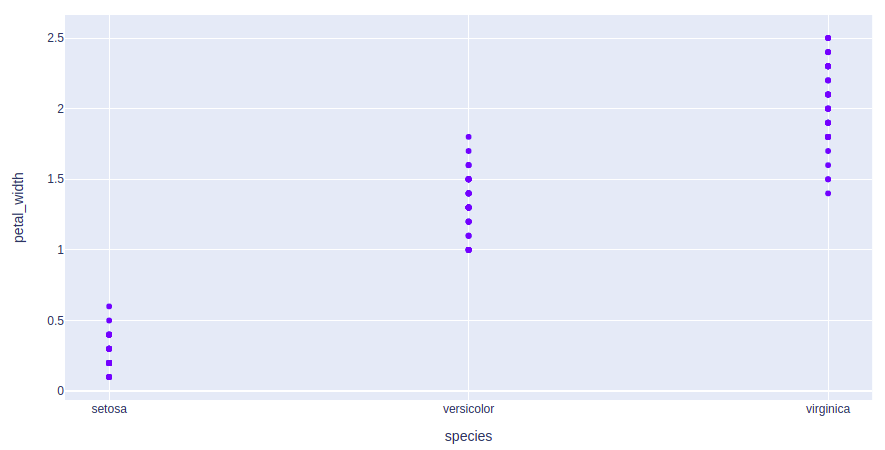
A [**bubble plot**](https://www.geeksforgeeks.org/bubble-chart-using-plotly-in-python/) is a scatter plot with bubbles (color-filled circles). Bubbles have various sizes dependent on another variable in the data. It can be created using the scatter() method of plotly.express.

**Example 1:**Scatter Plot

* Python3

|  |
| --- |
| import plotly.express as px    # using the iris dataset  df = px.data.iris()    # plotting the scatter chart  fig = px.scatter(df, x="species", y="petal\_width")    # showing the plot  fig.show() |

**Output:**



**Example 2:**Bubble Plot

* Python3

|  |
| --- |
| import plotly.express as px    # using the iris dataset  df = px.data.iris()    # plotting the bubble chart  fig = px.scatter(df, x="species", y="petal\_width",                   size="petal\_length", color="species")    # showing the plot  fig.show() |

**Output:**



**Pie Charts**

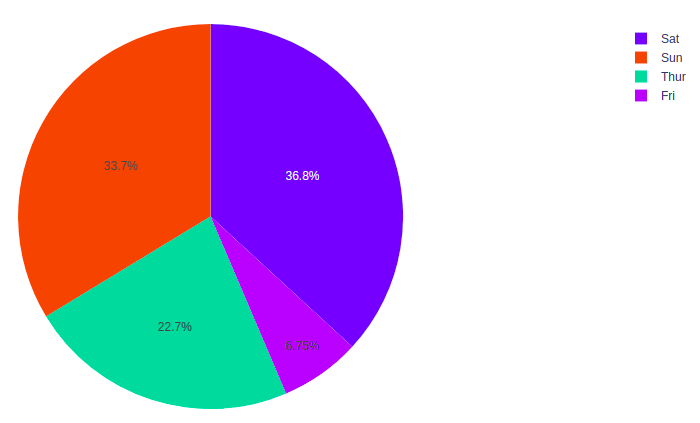
A [**pie chart**](https://www.geeksforgeeks.org/pie-plot-using-plotly-in-python/) is a circular statistical graphic, which is divided into slices to illustrate numerical proportions. It depicts a special chart that uses “pie slices”, where each sector shows the relative sizes of data. A circular chart cuts in a form of radii into segments describing relative frequencies or magnitude also known as circle graph.

**Example:**

* Python3

|  |
| --- |
| import plotly.express as px    # using the tips dataset  df = px.data.tips()    # plotting the pie chart  fig = px.pie(df, values="total\_bill", names="day")    # showing the plot  fig.show() |

**Output:**

****