Creating Different plots :- Histograms, Pie Chart

Topics Covered:

- Introduction to Histogram
- Creating and Customizing Histogram
- Introduction to Pie Chart
- Creating and Customizing Pie Chart

Introduction to Histogram

What is Histogram ? Why do we use Histogram?

A histogram is basically used to **represent data provided in a form of some groups.** It is an accurate method for the graphical representation of numerical data distribution. It is a type of bar plot where the X-axis represents the bin ranges while Y-axis gives information about frequency.
The purpose of a histogram is to **graphically summarize the distribution of a univariate data set**.
For more information on Histogram please refer to the official Matplotlib Documentation for Histogram.

Creating and Customizing Histogram

To create a histogram the first step is to create bin of the ranges, then distribute the whole range of the values into a series of intervals, and count the values which fall into each of the intervals.Bins are clearly identified as consecutive, non-overlapping intervals of variables.The matplotlib.pyplot.hist() function is used to compute and create histogram of x.
We can create Histograms in python using Matplotlib and the syntax for the same is given below.

Syntax:-

```
plt.hist(x,color,bins,edgecolor)
```

Where,

- plt is the alias name for pyplot
- The hist() function takes arguments that describe the layout of the histogram.
- x - data(column) to be represented on the x-axis
- color - Used to specify color of the line
- bins - Used to specify the buckets for which the frequency will be computed.
- edgecolor - Used to specify the edgecolor of the bars of histogram.

Some more Customization parameters for Histogram are shown below

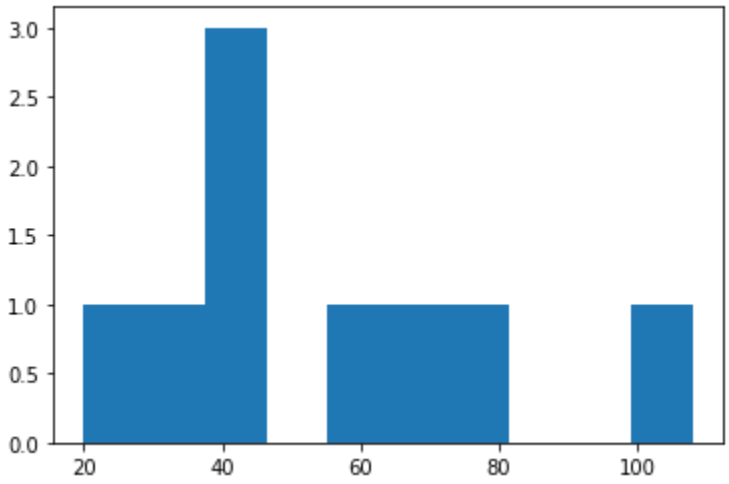
Attribute	Parameter
density	contains boolean values
range	represents upper and lower range of bins
histtype	used to create type of histogram [bar, barstacked, step, stepfilled], default is “bar”
align	controls the plotting of histogram [left, right, mid]
weights	contains array of weights having same dimensions as x
bottom	basline of each bin
rwidth	relative width of the bars with respect to bin width
label	string or sequence of string to match with multiple datasets
log	used to set histogram axis on log scale

Plotting a normal Histogram

Code example:-

```
# Plotting a simple Histogram
plt.hist(df_budget['Budget'])
plt.show()# Showing the plot
```

Output:-



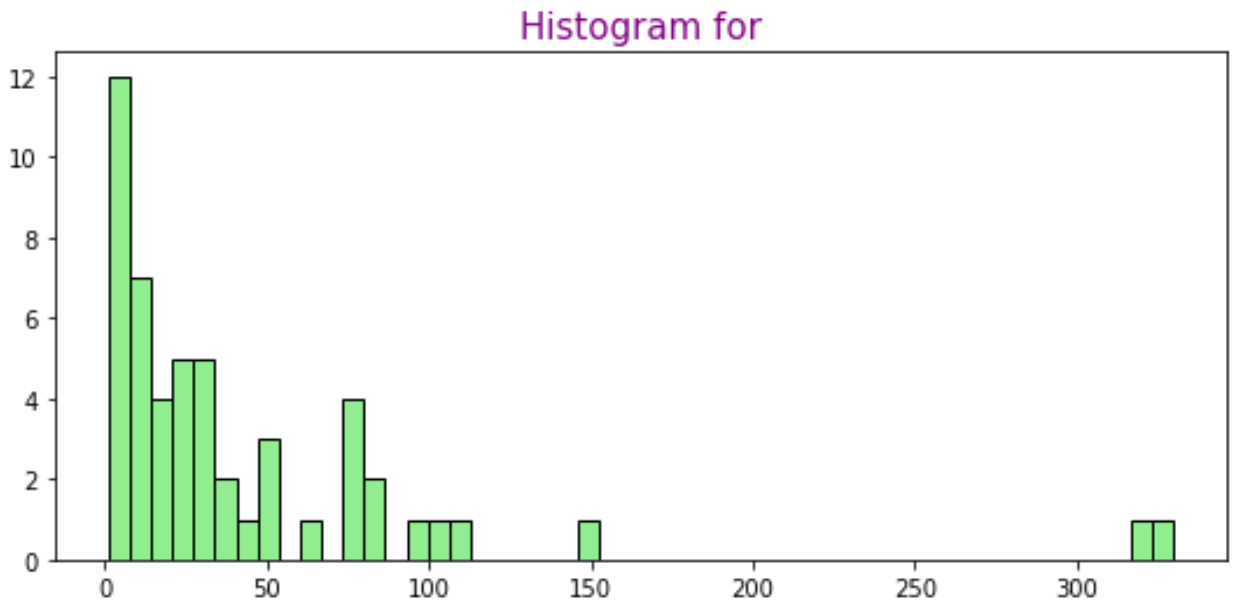
Customizing plot with color, edgecolor and bins

We can add color to the histogram along with edge color. We can also define bins for the histogram along with figure size and other customizations. The code example for the same is given below.

Code example:-

```
plt.figure(figsize=(9,4)) # setting plt figure
# Customizing plot with histogram color, edgecolor and bins
plt.hist(df['BO_Collection'],color='lightgreen',edgecolor='black',bins=50)
# Setting title for the plot
plt.title('Histogram for',fontsize=15,color='purple')
plt.show() # Showing the plot
```

Output:-



Introduction to Pie Chart

What is a Pie Chart ? Why do we use Pie Chart?

A Pie Chart is a **circular statistical plot** that can display only one series of data. The area of the chart is the total percentage of the given data. **The area of slices of the pie represents the percentage of the parts of the data.** The slices of pie are called wedges. The area of the wedge is determined by the length of the arc of the wedge. The area of a wedge represents the relative percentage of that part with respect to the whole data. Pie charts are commonly used in business presentations like sales, operations, survey results, resources, etc as they provide a quick summary
For more information on Pie Chart please refer to the official Matplotlib Documentation for Pie Chart.

Creating and Customizing Pie Chart

We can create a Pie Chart in python using Matplotlib, To plot pie chart, we need a continuous column and its corresponding labels (the categories that this continuous column highlights), the syntax for the same is given below.

Syntax:-

```
plt.pie(x,labels,color,explode,autopct,shadow)
```

Where,

- plt is the alias name for pyplot
- The pie() function takes arguments that describe the layout of the pie chart.

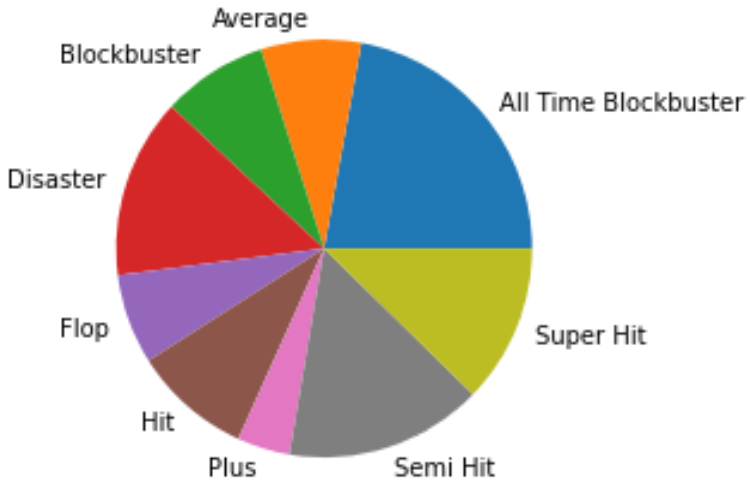
- x - data(continuous column) for which percentage values will be computed
- Label - Used to define the categorical description for the continuous column x. It will be represented as legend
- color - Used to specify the color of the different pies
- autopct - Used to display the percentage values in a particular format
- shadow - Used to set the shadow on or off
- counterclock - Used to display the values anticlockwise or clockwise

Plotting a normal Pie Chart

Code example:-

```
# Plotting a simple Pie Chart
plt.pie(x = df_budget['Budget'],labels=df_budget['Box_Office_Verdict'])
plt.show()# Showing the plot
```

Output:-



Customizing the Pie chart

A pie chart can be customized on the basis of several aspects. There are various customization parameters to customize a pie chart and they are mentioned below.

- explode - Used to make one of the wedges of the pie chart to stand out
- textprops- Used to define the color for all the labels in the pie chart.
- startangle - startangle attribute rotates the plot by the specified degrees in counter clockwise direction performed on the x-axis of the pie chart.
- wedgeprop - Wedges of the pie can be customized using wedgeprop which takes a Python dictionary as parameter with name values pairs denoting the wedge properties like linewidth, edgecolor, etc.
- frame - By setting frame=True axes frame is drawn around the pie chart.

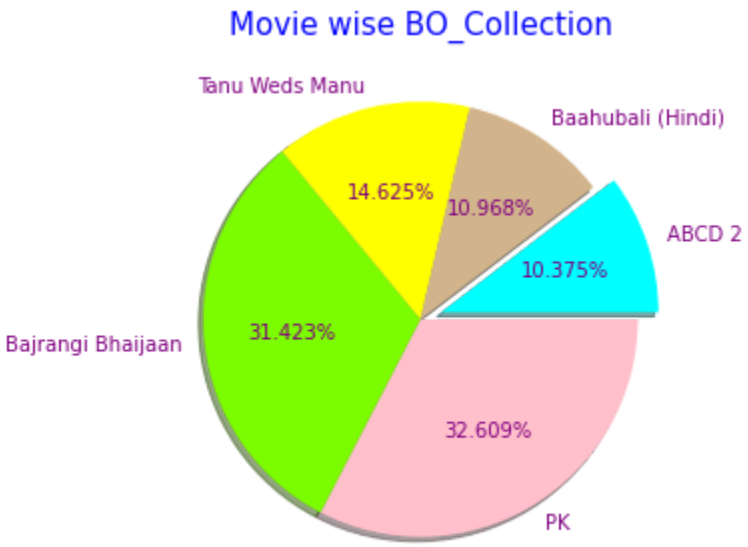
For more information on Pie Chart customization please refer to the official Pie Chart Customization Guide.

The code example and the syntax and example for that is given below.

Code example:-

```
plt.figure(figsize=(5,5))  # setting plt figure
# Customizing plot by setting colors, autopact, explode, shadow, counterclock, textprops properties
plt.pie(x = df_top_movies['BO_Collection'],labels=df_top_movies.index,
      colors=['pink','lawngreen','yellow','tan','cyan'],autopct='%.3f%%',
     explode=(0,0,0,0,0.1),shadow=True,counterclock=False,textprops=dict(color="Purple"))
# Setting title for the plot
plt.title('Movie wise BO_Collection',fontsize=15,color='blue')
plt.show() # Showing the plot
```

Output:-



Customizing the Pie chart with legend

We can add legend to the pie chart by using legend() function. A legend is an area describing the elements of the graph. We can customize the legend by using following properties:-

- title - title is used to display title of the legend,
- loc - loc is used to specify the location of the legend.Default value of loc is loc="best" (upper left). The strings 'upper left', 'upper right', 'lower left', 'lower right' place the legend at the corresponding corner of the axes/figure.
- bbox_to_anchor - bbox_to_anchor=(x, y) of legend() function is used to specify the coordinates of the legend,
- ncol- ncol represents the number of columns that the legend has. Its default value is 1.

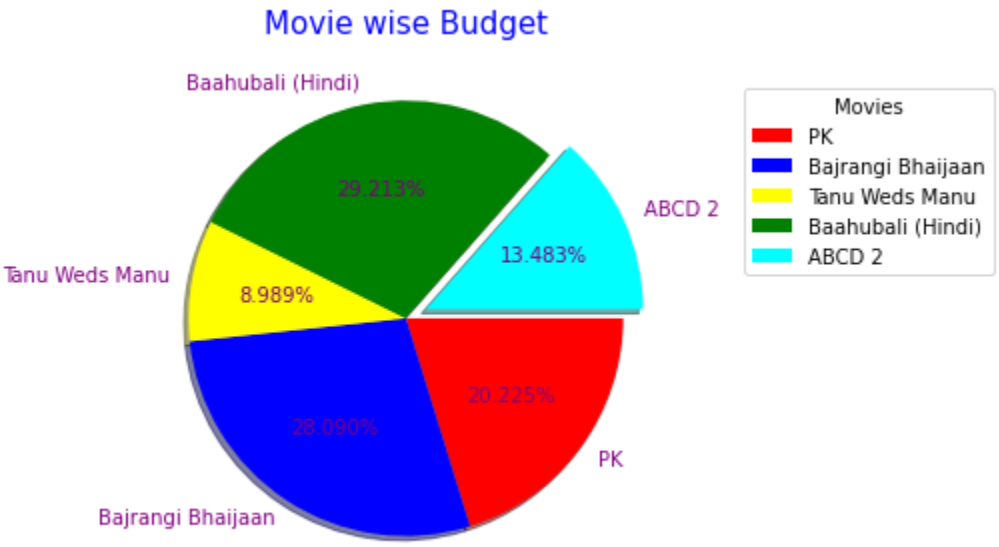
For more information on legend() customization please refer to the official Matplotlib Documentation for legend() customization,

The code example and the syntax and example for that is given below.

Code example:-

```
plt.figure(figsize=(5,5)) # setting plt figure
# Customizing plot by setting Label, colors, autopact, explode, shadow and counterclock properties
plt.pie(x = df_top_movies['Budget'],labels=df_top_movies.index,
      colors=['red','blue','yellow','green','cyan'],autopct='%.3f%%',
     explode=(0,0,0,0,0.1),shadow=True,counterclock=False)
# Setting title for the plot
plt.title('Movie wise Budget',fontsize=15,color='blue')
# Setting legend for the plot with Legend title, location and bbox_to_anchor properties
plt.legend(df_top_movies.index,  title="Movies",loc='right', bbox_to_anchor=(1, 1,0.6,-0.5))
plt.show() # Showing the plot
```

Output:-



Reference Notebook & Dataset

- Jupyter Notebook Download Link
- Bollywood Movies Dataset Download Link