

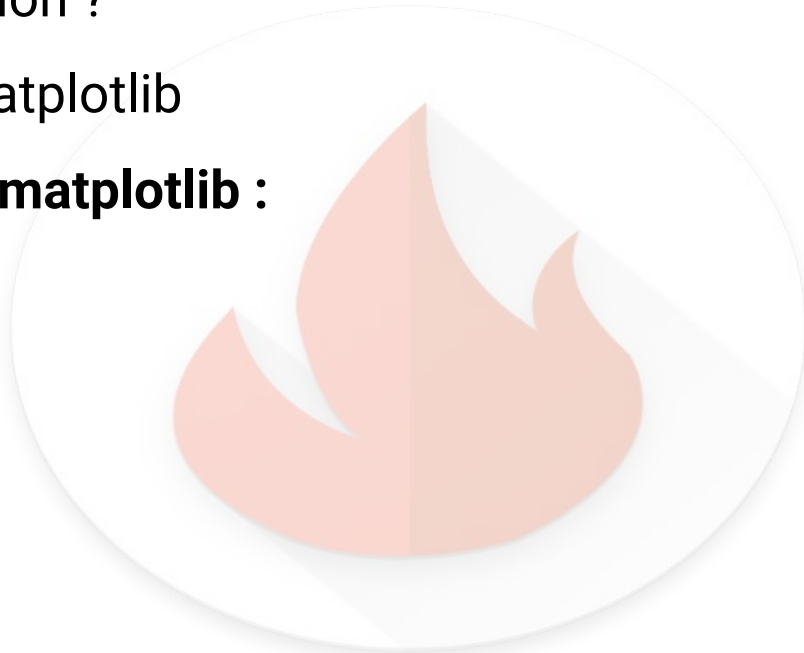


Matplotlib

By Fireblaze AI School

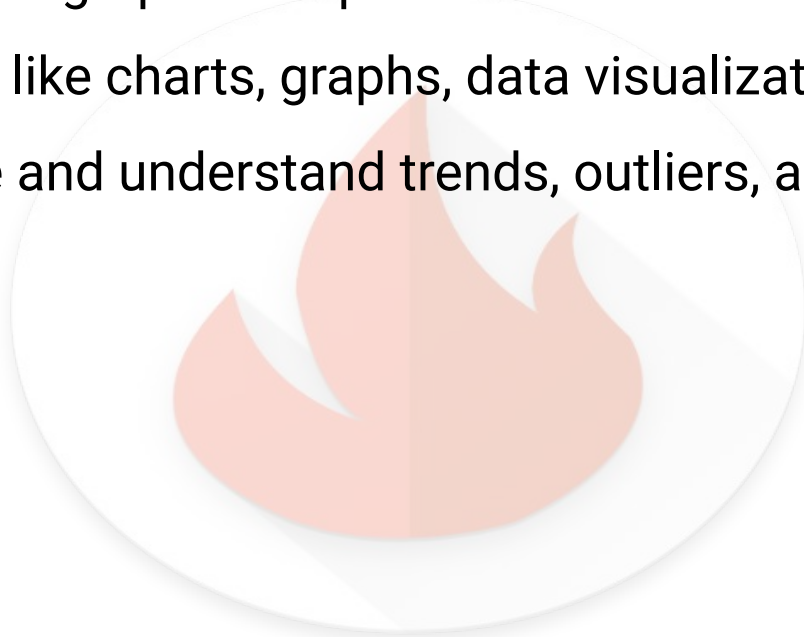
Index

- What is visualization ?
- Introduction to Matplotlib
- **Different plots in matplotlib :**
 - bar()
 - pie()
 - scatter()
 - boxplot()
 - line # plot()
 - hist()



Visualization

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



Introduction to Matplotlib

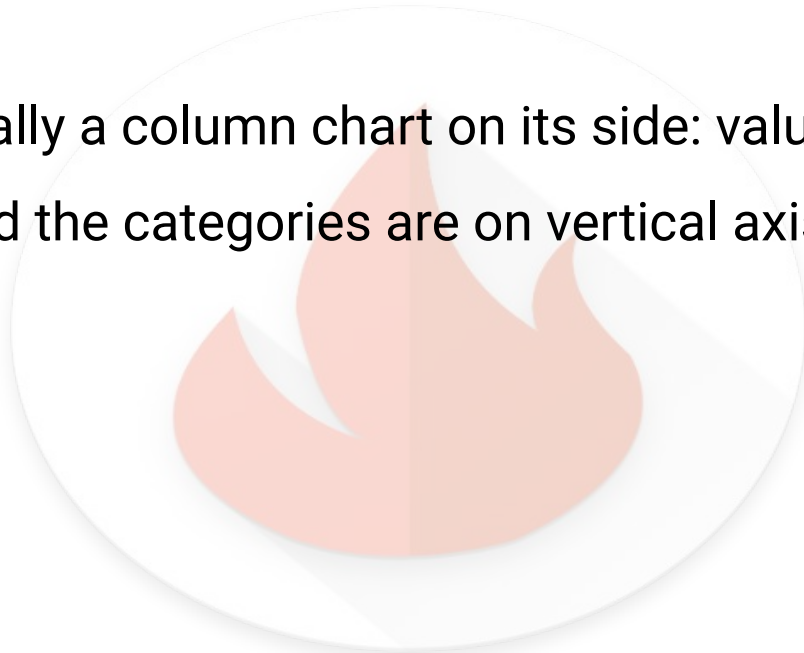
Matplotlib is one of the most popular Python packages used for data visualization. It is a cross-platform library for making 2D plots from data in arrays. Matplotlib is written in Python and makes use of NumPy, the numerical mathematics extension of Python.



Bar plot

What is It?

A bar chart is essentially a column chart on its side: values are presented on the horizontal axis and the categories are on vertical axis, on the left.



Bar plot

What Does It Visualize?

Bar charts are more commonly used to compare different values, items and categories of data. From a purely practical perspective, they're also used over column charts when the names of the categories are too long to comfortably read on their side! They are not usually used to show trends over time

Bar plot

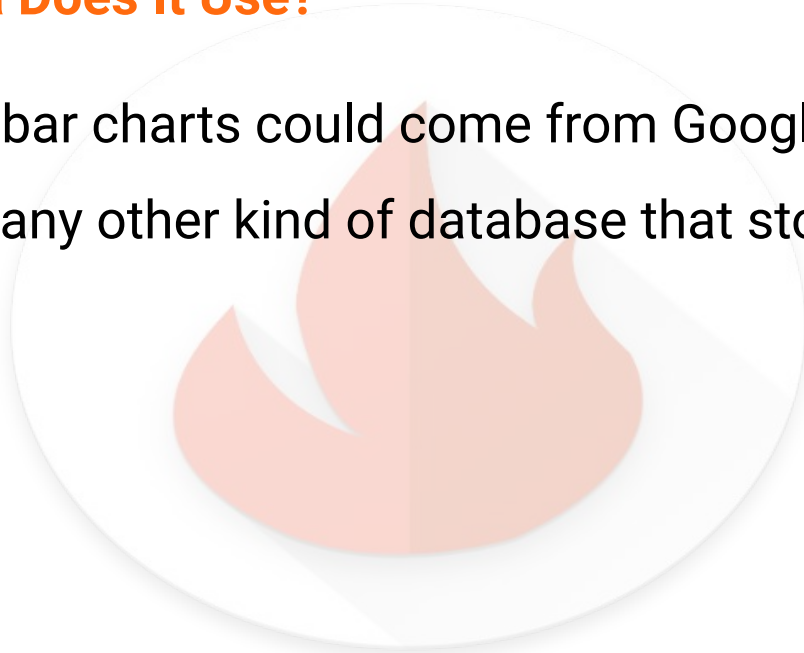
What Does It Measure?

Like column charts, bar charts are frequently used to compare the total number of items within a category, for example total sales or the number of respondents that selected a particular answer. However, they're also handy for visualizing sub-categories using colour coding.

Bar plot

What Sources of Data Does It Use?

Data used to compile bar charts could come from Google Analytics, your CRM, sales figures or any other kind of database that stores data numerically.



Bar plot

importing matplotlib module

`from matplotlib.pyplot as plt`

x-axis values

`x = [5, 2, 9, 4, 7]`

Y-axis values

`y = [10, 5, 8, 4, 2]`

Function to plot the bar

`plt.bar(x,y)`

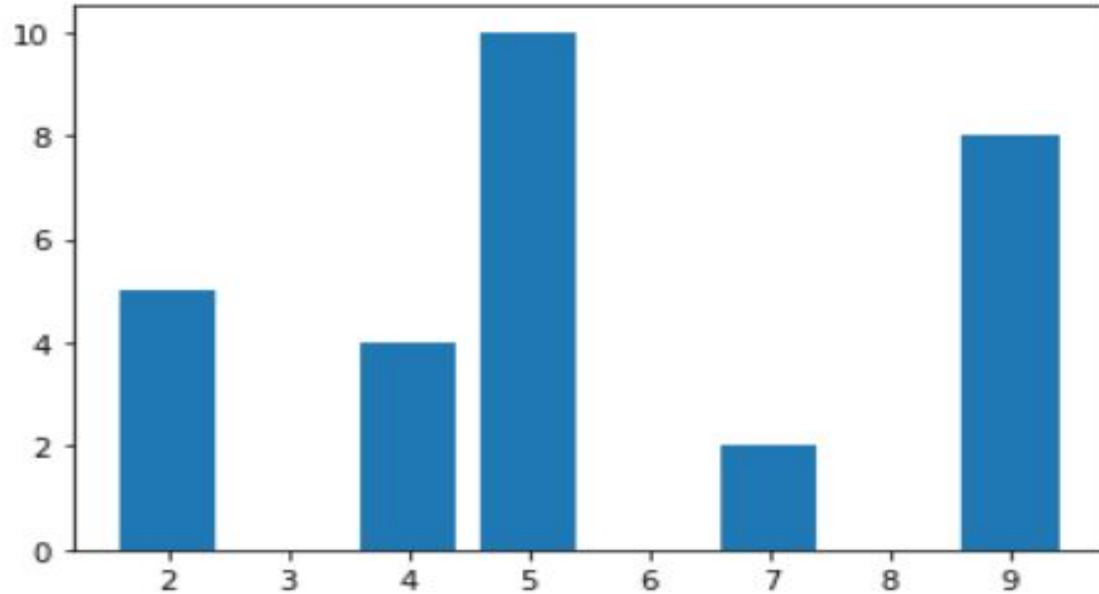
function to show the plot

`plt.show()`



Bar plot

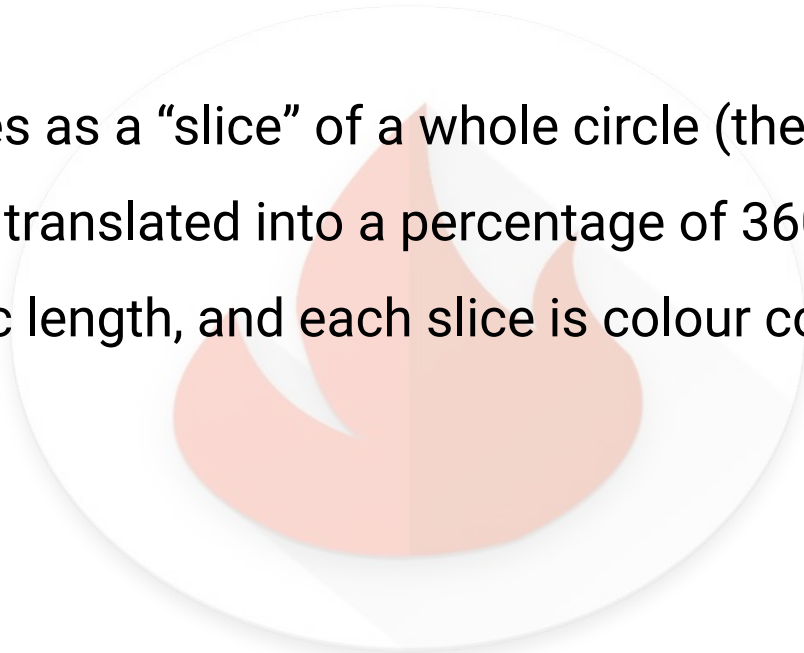
Output :



Pie Chart

What is It?

Pie charts show values as a “slice” of a whole circle (the whole pie). Numerical Values are translated into a percentage of 360 degrees, represented by the arc length, and each slice is colour coded accordingly



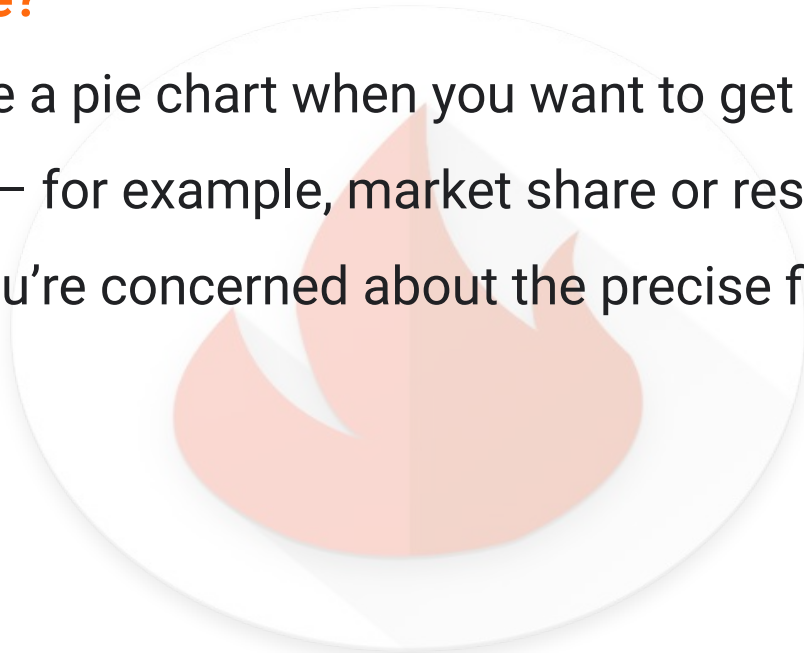
Pie Chart

What Does It Visualize?

Pie charts show what percentage of the whole is made up of each category. That means they deal with total numbers, and trends in overall responses, rather than changes over time. That means it's a good idea to use a pie chart when displaying proportional data and/or percentages. Remember that the point is to represent the size relationship between the parts and the entire entity, so these parts need to add up to a meaningful whole.

What Does It Measure?

It makes sense to use a pie chart when you want to get a rapid, overall idea of the spread of data – for example, market share or responses to a survey – rather than when you're concerned about the precise figures they represent.



What Sources of Data Does It Use?

Survey and questionnaire responses, data from social media sources or Google analytics, total sales figures and so on will all work. Keep it fairly simple though – if you have more than 6 categories, your pie chart won't give you much information at a glance, especially if there's no clear “winning” answer

Pie Chart

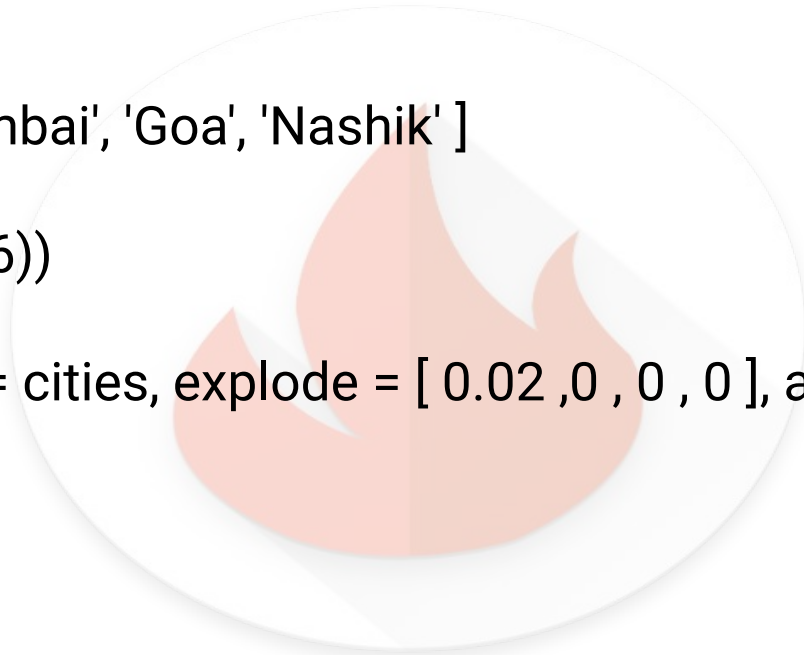
```
values=[12,34,56,98]
```

```
cities=[ 'Nagpur', 'Mumbai', 'Goa', 'Nashik' ]
```

```
plt.figure(figsize=(12,6))
```

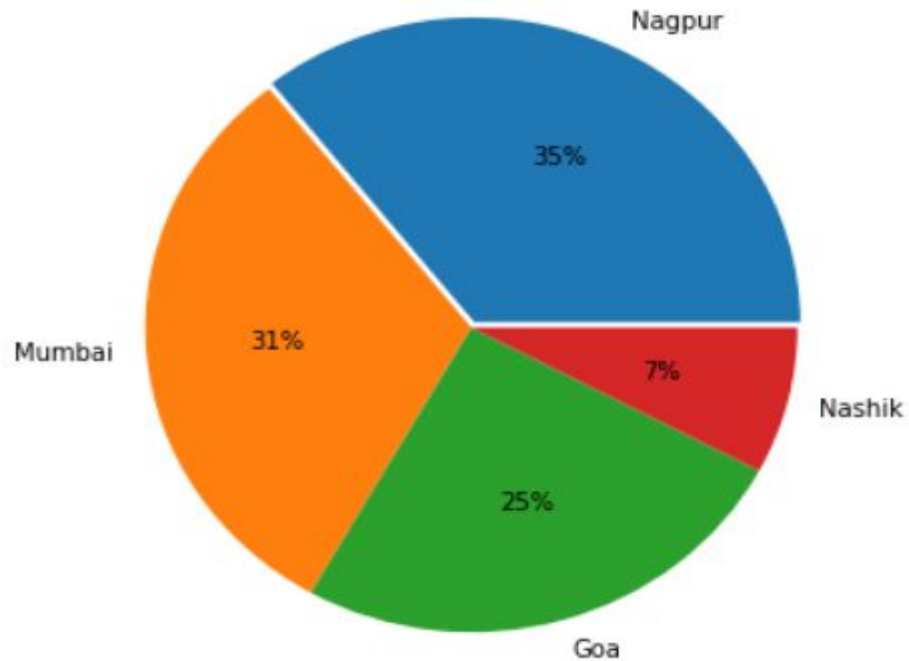
```
plt.pie(values, labels = cities, explode = [ 0.02 ,0 , 0 , 0 ], autopct = '%i%%' )
```

```
plt.show()
```



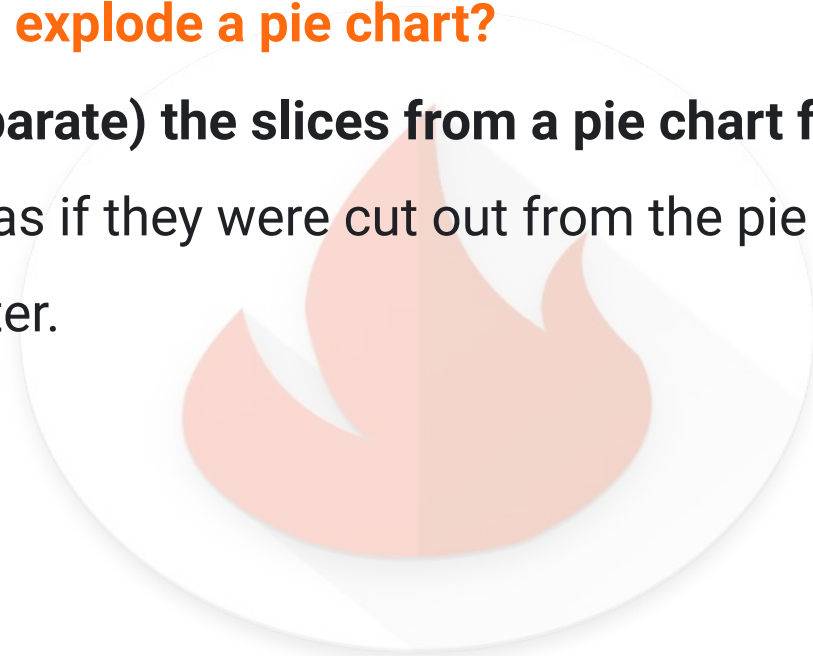
Pie Chart

Output :



What does it mean to explode a pie chart?

You can **explode (separate) the slices from a pie chart for emphasis**. The exploded slices look as if they were cut out from the pie and moved slightly outward from its center.



What is Autopct in pie chart?

autopct **enables you to display the percent value using Python string formatting**. For example, if autopct='%. 2f' , then for each pie wedge, the format string is '%. 2f' and the numerical percent value for that wedge is pct , so the wedge label is set to the string '%. 2f'.

Scatter plot

What is It?

Scatter charts are a more unusual way to visualize data than the examples above. These are mathematical diagrams or plots that rely on Cartesian coordinates. If you're using one color in the graph, this means you can display two values for two variables relating to a data set, but you can also use two colors to incorporate an additional variable.

What Does It Visualize?

In this type of graph, the circles on the chart represent the categories being compared (demonstrated by circle color), and the numeric volume of the data (indicated by the circle size).

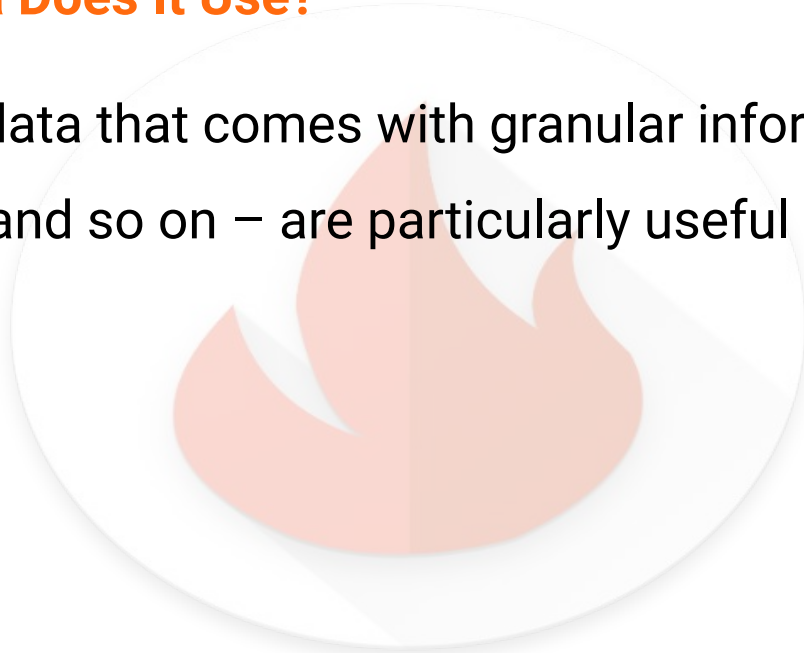
What Does It Measure?

Scatter charts are great in scenarios where you want to display both distribution and the relationship between two variables.

Scatter plot

What Sources of Data Does It Use?

CRM, sales and lead data that comes with granular information on buyers – age, gender, location and so on – are particularly useful for this kind of graph.



Scatter plot

```
import matplotlib.pyplot as plt
```

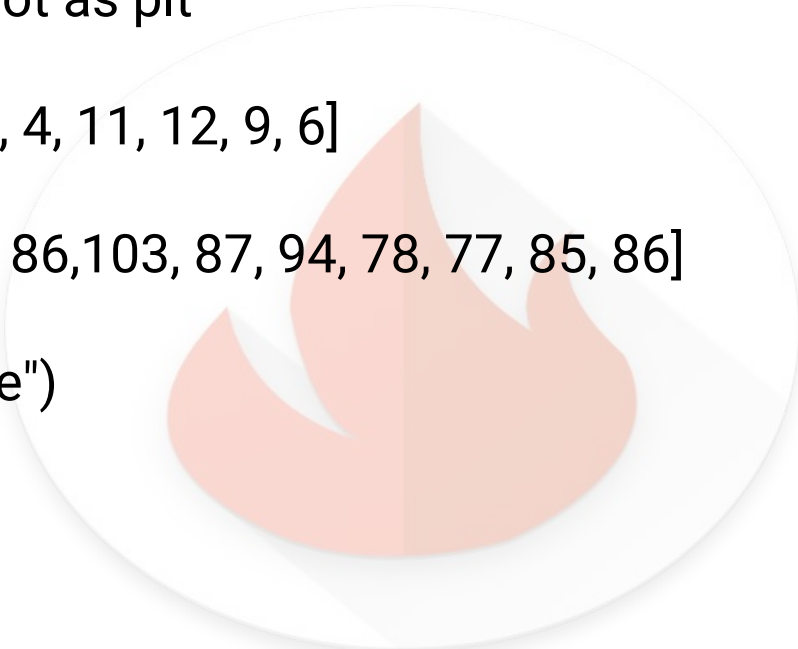
```
x =[5, 7, 8, 7, 2, 17, 2, 9, 4, 11, 12, 9, 6]
```

```
y =[99, 86, 87, 88, 100, 86, 103, 87, 94, 78, 77, 85, 86]
```

```
plt.scatter(x, y, c ="blue")
```

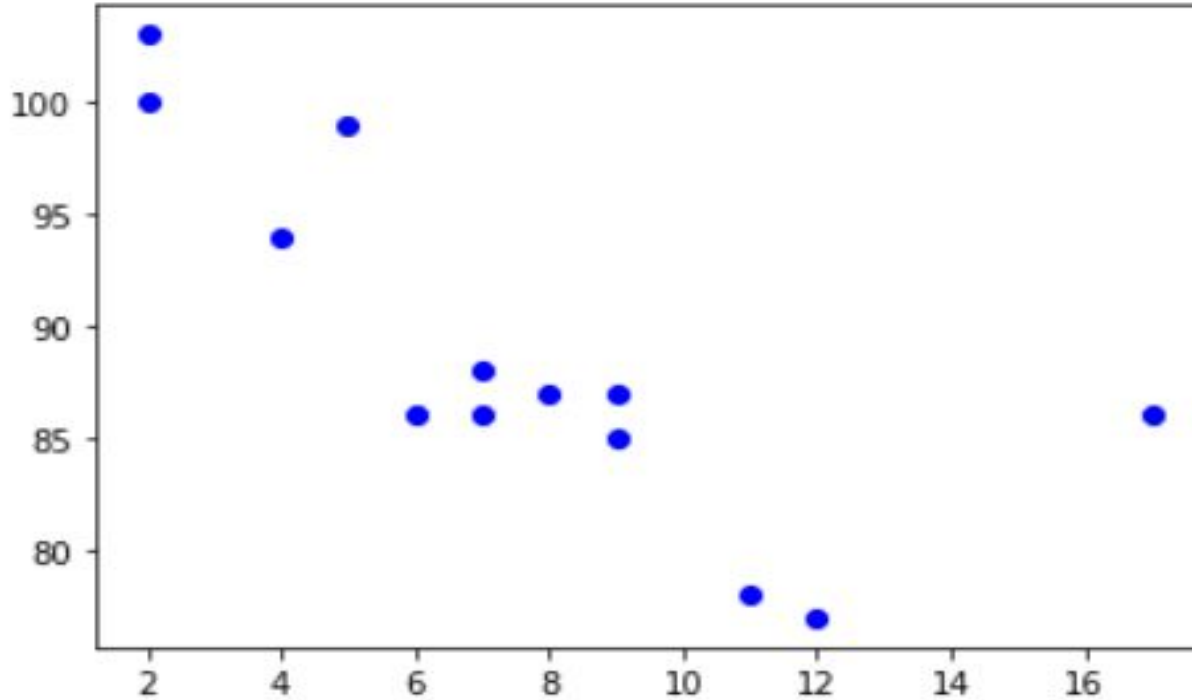
```
# To show the plot
```

```
plt.show()
```



Scatter plot

Output :



Boxplot

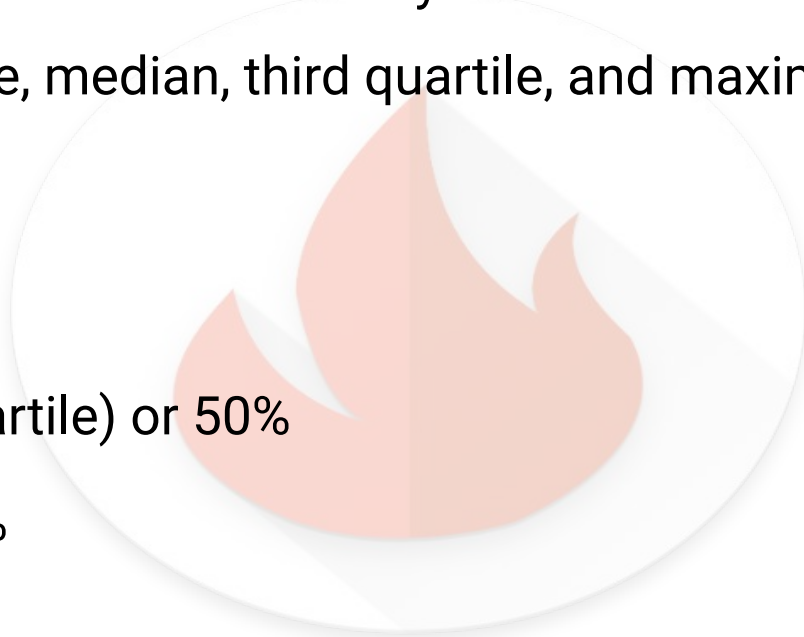
What is It?

1. The box plot, also called the box and whisker diagram is used for depicting groups of numerical data through the quartiles. It is known as the box and whisker diagram because it is composed of a box and whiskers. Boxplot is also used for detecting the outlier in a data set.

Boxplot

2. A box plot is composed of a summary of 5 different data points: the minimum, first quartile, median, third quartile, and maximum.-

- Minimum
- First Quartile or 25%
- Median (Second Quartile) or 50%
- Third Quartile or 75%
- Maximum



Boxplot

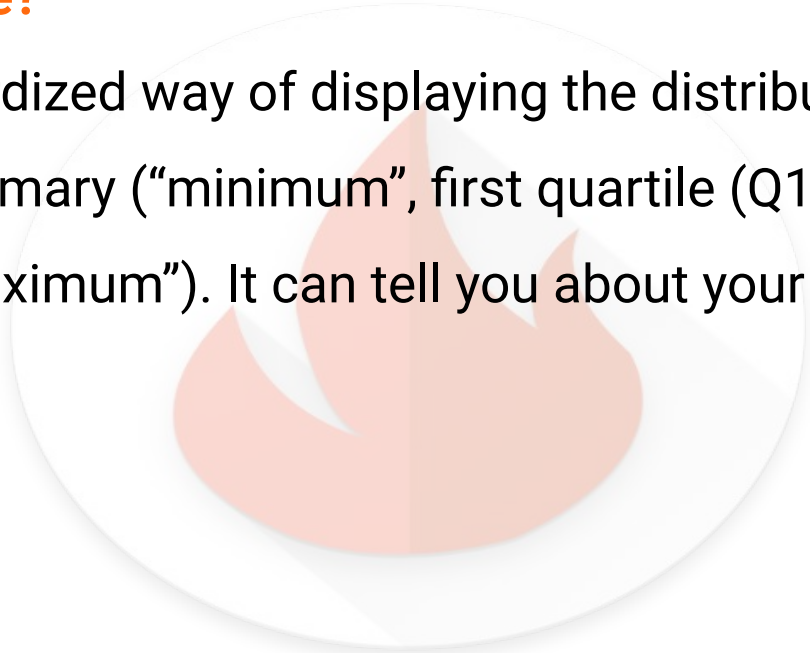
What Does It Visualize?

A Box and Whisker Plot (or Box Plot) is a convenient way of visually displaying **the data distribution through their quartiles**. The lines extending parallel from the boxes are known as the “whiskers”, which are used to indicate variability outside the upper and lower quartiles.

Boxplot

What Does It Measure?

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 tell you about your outliers and what their values are.



Boxplot

What type of data does a box plot use?

In descriptive statistics, a box plot or boxplot (also known as box and whisker plot) is a type of chart often used in explanatory data analysis. Box plots visually show the distribution of **numerical data** and skewness through displaying the data quartiles (or percentiles) and averages.

Boxplot

Import libraries

```
import matplotlib.pyplot as plt  
import numpy as np
```

Creating dataset

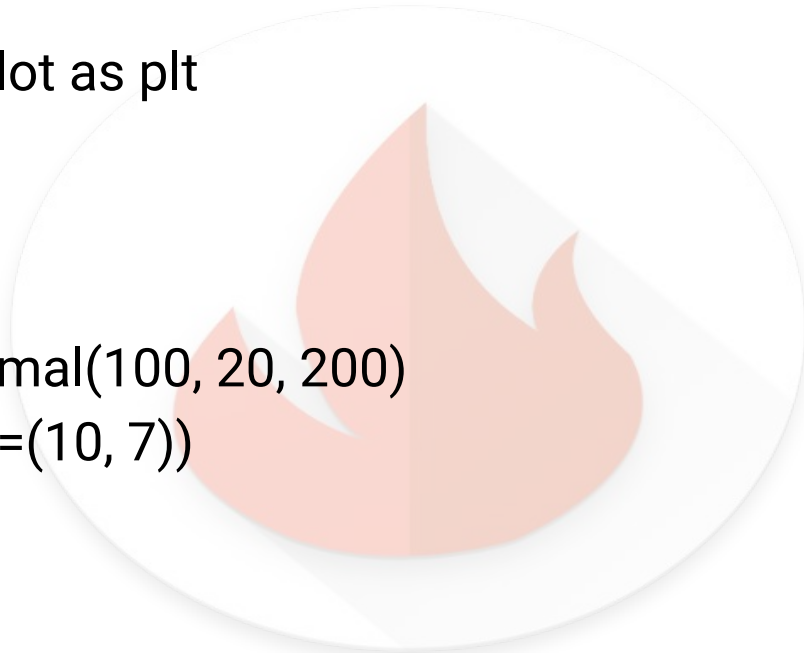
```
np.random.seed(10)  
data = np.random.normal(100, 20, 200)  
fig = plt.figure(figsize =(10, 7))
```

Creating plot

```
plt.boxplot(data)
```

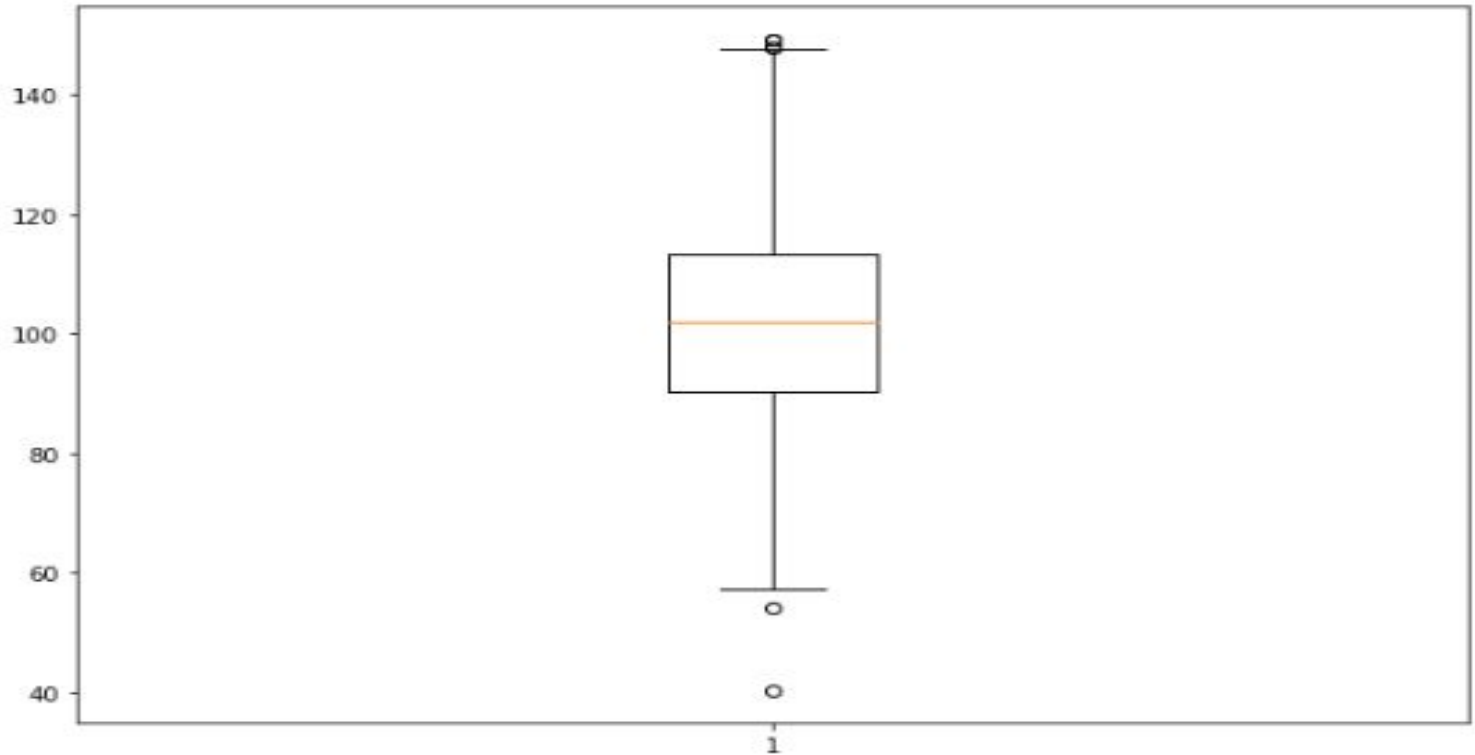
show plot

```
plt.show()
```



Boxplot

Output :



Lineplot

What is It?

Line charts plot data points on a graph and then join them up with a single line that zigzags from each point to the next.

What Does It Visualize?

These are super simple and very popular, because they give you an immediate idea of how a trend emerged over time. You can see when peaks and troughs hit, whether the overall values are going up or down, and when there's a sharp spike or drop in numbers.

Lineplot

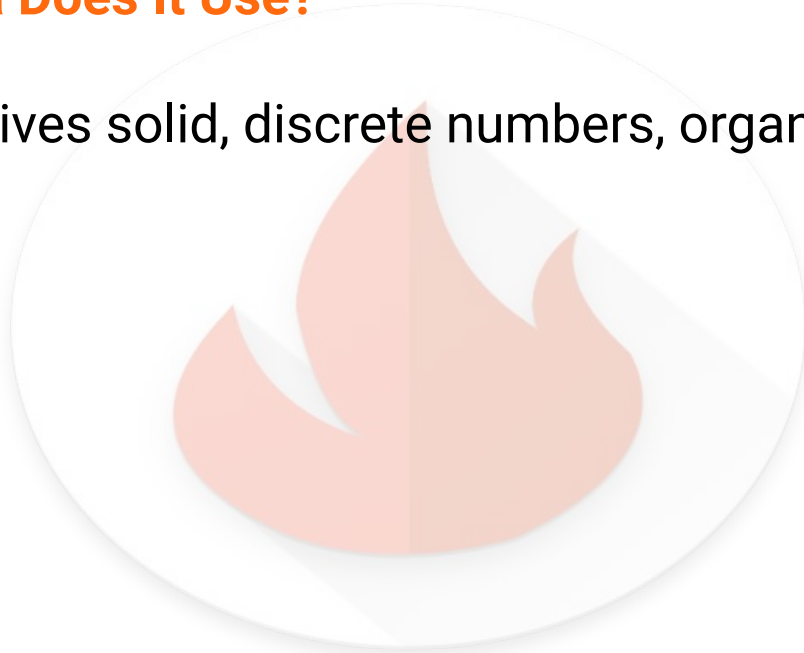
What Does It Measure?

There are many different business cases that work well with line charts. Pretty much anything that compares data, or shows changes, over time is well suited to this type of visualization. Again, it's all about visualizing a trend. You can also compare changes over the same period of time for more than one group or category very easily, by adding a “break by” category

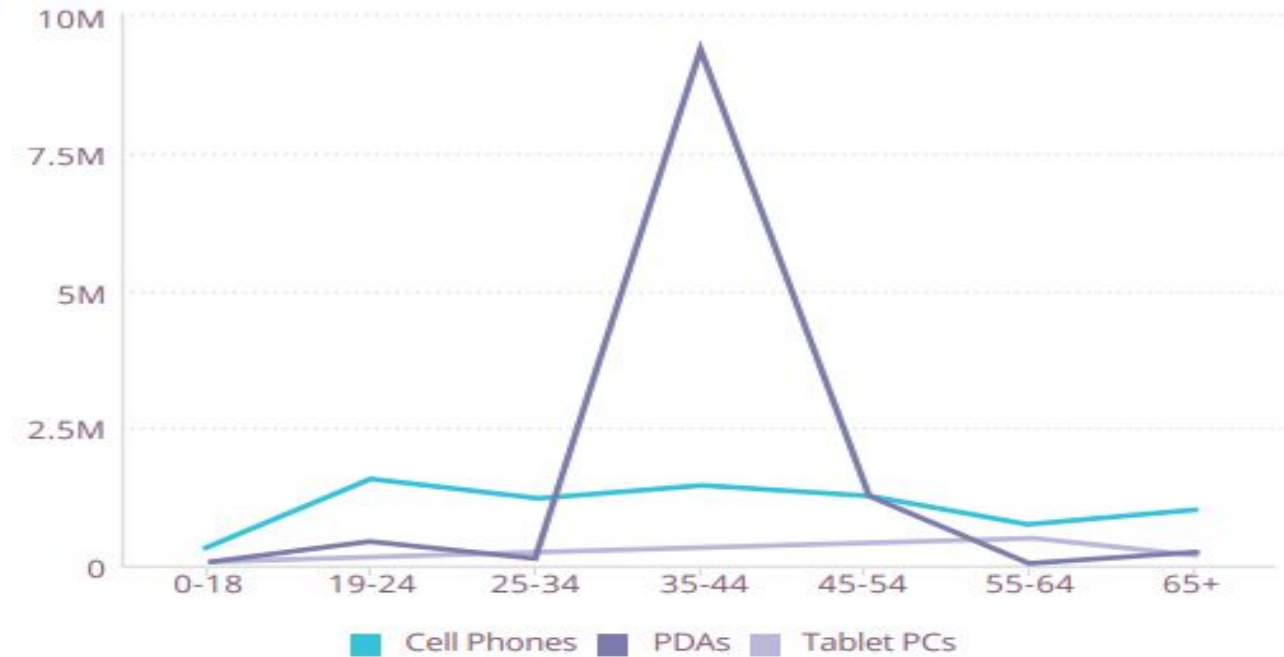
Lineplot

What Sources of Data Does It Use?

Again, anything that gives solid, discrete numbers, organized by time.

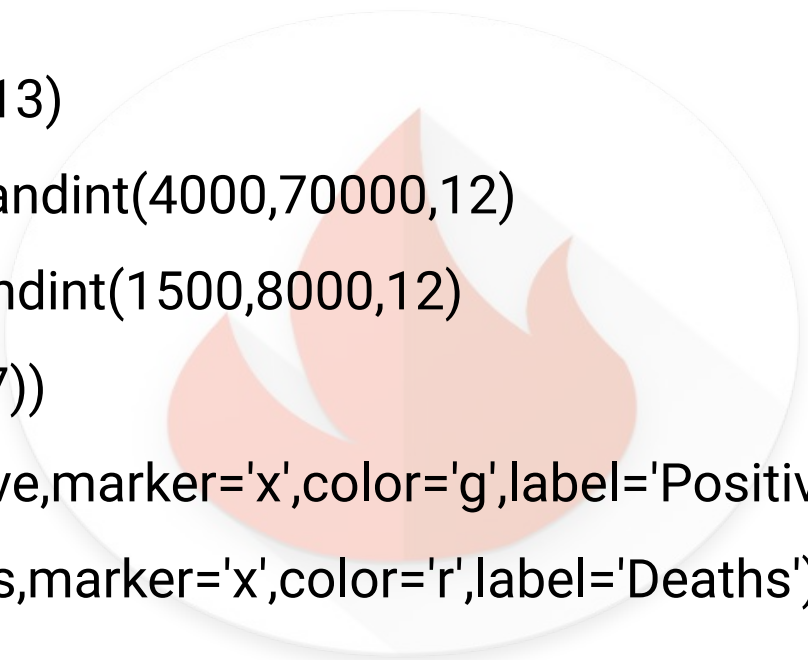


Lineplot



Lineplot

```
import numpy as np
months=np.arange(1,13)
positive=np.random.randint(4000,70000,12)
deaths=np.random.randint(1500,8000,12)
plt.figure(figsize=(10,7))
plt.plot(months,positive,marker='x',color='g',label='Positive')
plt.plot(months,deaths,marker='x',color='r',label='Deaths')
plt.xlabel('Months')
```



Lineplot

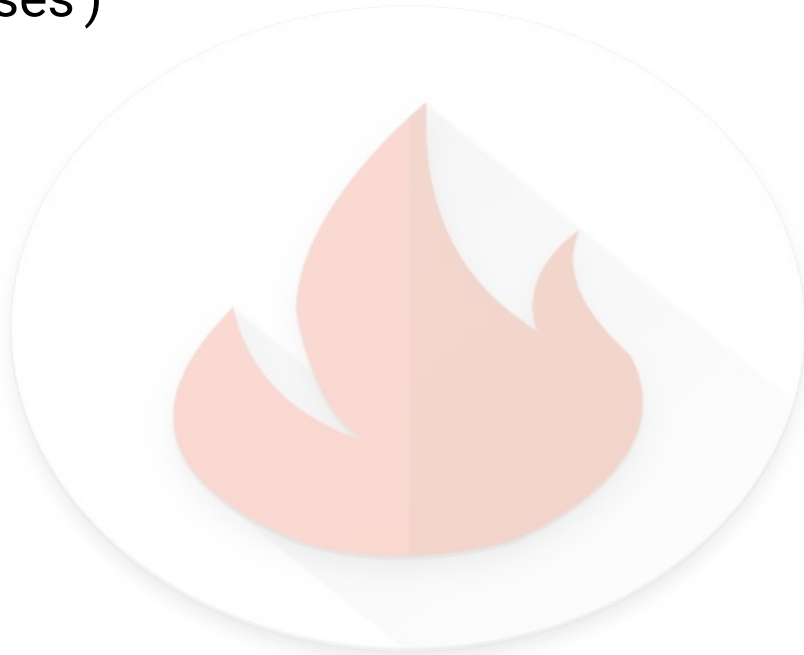
```
plt.ylabel('Positive Cases')
```

```
plt.legend()
```

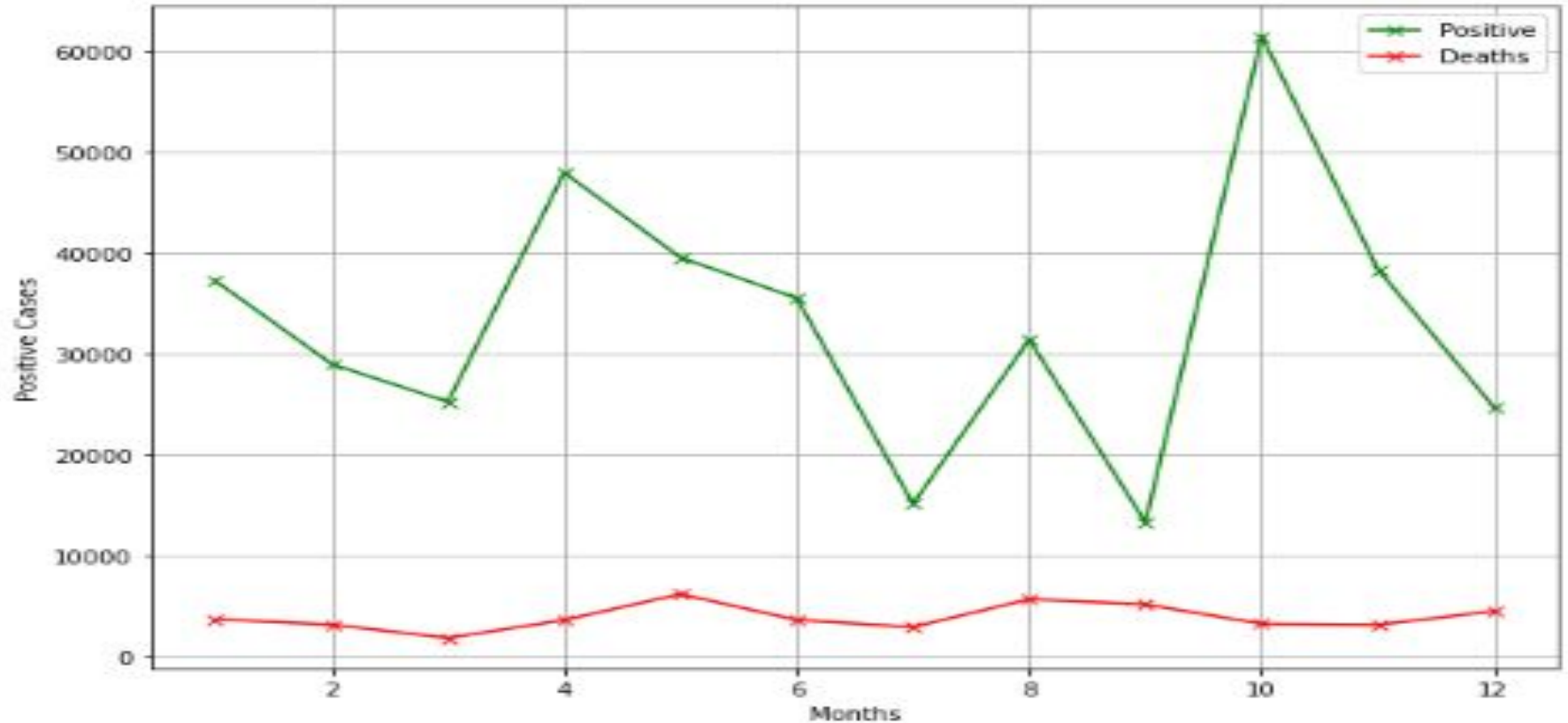
```
plt.grid()
```

```
plt.show()
```

Output:



Lineplot



Histogram

What is it ?

- A histogram is a graph showing *frequency* distributions.
- It is a graph showing the number of observations within each given interval.
- In Matplotlib, we use the **hist ()** function to create histograms. The hist () function will use an array of numbers to create a histogram, the array is sent into the function as an argument.

Histogram

What does it visualize ?

One of the most common statistical charts, histograms visualize **the underlying frequency distribution of a given set of continuous data**. You can easily summarize a large range of values by grouping/splitting the entire data set into defined intervals or classes – commonly known as bins.

Histogram

What does it measure ?

The histogram is a popular graphing tool. It is used to summarize **discrete or continuous data** that are measured on an interval scale. It is often used to illustrate the major features of the distribution of the data in a convenient form.

Histogram

```
from matplotlib import pyplot as plt    # import matplotlib
import numpy as np                      # import numpy library

fig,ax = plt.subplots(1,1)
a = np.array([22,87,5,43,56,73,55,54,11,20,51,5,79,31,27]) #Create Array
ax.hist(a, bins = [0,25,50,75,100])
ax.set_title("histogram of result")
ax.set_xticks([0,25,50,75,100])
ax.set_xlabel('marks')
ax.set_ylabel('no. of students')
plt.show()
```

Histogram

Output :

