# **Data visualization for data analysis**

Data visualization is a fundamental tool in data analysis. Data can often be difficult to understand simply by looking at a table of numbers, but when represented graphically, patterns and trends can be more apparent and easier to interpret. Additionally, visualizations are an effective way to communicate results and ideas to others.

When creating charts and visualizations, it's important to keep some best practices in mind, such as how to select the right visualization for the type of data being represented, make sure the visualization is easy to read, and avoid exaggerating or distorting the data. It is also important to clearly label the axes and provide informative titles for the visualizations.

Common chart types include pie charts, box plots, scatter charts, bar charts, area charts, and histograms.

Pie charts are useful for showing the proportion of different categories in a data set, while boxplots are useful for visualizing the distribution and outliers in a numerical data set. Scatter charts are great for showing the relationship between two variables, while bar and area charts are useful for comparing values ​​between different categories. Histograms are useful for showing the distribution of numerical data.

In Python, there are several data visualization libraries available, such as Seaborn, Matplotlib, and Plotly.Seaborn is a data visualization library based on Matplotlib, which provides an easier to use syntax and more attractive graphics. Matplotlib is an older and more established data visualization library, and it provides a wide range of customizable plots. Plotly is an interactive data visualization library that allows the creation of interactive visualizations online.

## chart types

### pastel graphics

They are useful for showing the proportion of different categories in a data set. They are best used when the number of categories is small and when the proportions are significantly different from each other. For example, a pie chart can be used to show the proportion of a company's revenue that comes from different revenue sources, such as online sales, physical store sales, services, and others.

### box plots

Also known as boxplots, they are useful for showing the distribution of a data set and for identifying possible outliers or outliers. They are best used when you have numerical data and when you want to compare multiple data distributions. For example, a boxplot can be used to compare the distribution of scores for different groups of students on a test.

### scatter plots

They are useful for showing the relationship between two numeric variables. They are best used when you want to identify patterns or trends in data and when you want to determine if there is a relationship between two variables. For example, a scatter plot can be used to show the relationship between the number of study hours and students' test scores.

### Bar graphs

They are useful for showing the frequency or number of different categories in a data set. They are best used when you want to compare the frequency or number of different categories and when you have categorical data. For example, a bar chart can be used to show the number of sales of different products in a store during a given period of time.

### Area charts

They are useful for showing the cumulative distribution of a data set and for comparing the number or frequency of different categories over time. They are best used when you have a data set that is distributed over time or when you want to show the evolution of the frequency of different categories. For example, an area chart can be used to show the number of sales of different products in a store during a year.

### histograms

They are useful for showing the distribution of a set of numerical data and for identifying possible outliers or outliers. It is best used when you have a large data set and want to identify patterns in the distribution of the data. For example, a histogram can be used to show the distribution of test scores.

## visualization tools

Data visualization is a key part of data analysis, as it helps to understand and communicate the patterns and trends found in the data. There are several data visualization tools in Python, but three of the most popular are seaborn, matplotlib, and plotly.

### Seaborn

It is a matplotlib based data visualization library used to create attractive and efficient statistical plots in Python. This library is ideal for creating complex multi-variable charts and allows for easy customization of charts using color palettes, hatch styles, and style elements.

### Matplotlib

It is a very powerful and versatile data visualization library used to create high-quality charts in Python. This library allows you to create a wide variety of charts, from simple line charts to more complex charts such as maps and stacked bar charts. Matplotlib is also highly customizable and has a large community of users who share different solutions for creating custom plots.

### Plotly

It is an online interactive data visualization library that allows you to create interactive charts, diagrams, and dashboards. It is an open source library and can be used in both Python and R. Plotly is a very powerful tool for creating interactive visualizations, and is especially useful for sharing real-time charts and data online.

## Best practices when graphing data

Some of the best practices to consider when creating charts and visualizations for data analysis are:

### keep it simple

Too much information in a single graph can make interpretation difficult. It's best to split your data into multiple visualizations or use techniques like data pooling or aggregation to reduce the amount of information. Make sure the charts are easy to understand and are not overloaded with information. Use only the elements necessary to tell the story.

### Use the right color palette

Colors must be used carefully and with a specific purpose. Bright colors can distract from important information, while subtler colors can be hard to distinguish. It is important to choose colors that highlight the information you want to communicate. It is also important to note that some colors may have cultural connotations and that your choice may affect the perception of the data.

### Use titles and descriptive tags

Labels must be clear and descriptive, and the axes must be scaled appropriately to accurately represent the data. Legends should be concise and descriptive to facilitate understanding of the information.

### Choosing the right chart type

It is important to choose the type of graph that best represents the data and allows for easy interpretation. Select the type of graph that best suits the data and the story you want to tell. Some chart types work better for certain types of data, such as histograms for the frequency distribution. For example, a bar chart is great for comparing numerical values, while a line chart is better for showing trends over time.

### Avoid distortion and lack of context

Charts must be accurate and must not distort information to make the data appear more favorable. It is important to avoid techniques such as rescaling the axes or manipulating data values ​​to make the results appear more impressive. Additionally, it is important to provide context to help readers understand the importance of the data.

### try different options

There is no one correct way to create a chart. It is important to experiment with different options and adjust the graph as necessary to present the information clearly and concisely.

Other good practices include proper use of axis titles and labels, appropriate choice of chart type based on the nature of the data and purpose of the display, and attention to aesthetic details such as colors, readability, and clarity. simplicity. Also when choosing colors for a chart, it is important to select colors that are easily distinguishable from one another and are not too bright or garish. In addition, it is important to avoid using overly complicated three-dimensional graphics that can make the data difficult to understand.

For example, suppose we want to graph the amount of sales of a store per month in the last year. Using the Matplotlib library, we can create a line plot to represent this information. First, we import Matplotlib and create two lists, one for the months and one for the corresponding sales:

| import matplotlib.pyplot as plt  months = ["January", "February", "March", "April", "Mayo", "June",  "July", "August", "September", "October", "November", "December"] sales = [10000, 12000, 15000, 18000, 20000, 22000,  25000, 28000, 30000, 32000, 35000, 38000] |
| --- |

Next, we create the line chart and add appropriate labels and legends:

| plt.figure(figsize=(15,5)) plt.plot(months, sales) plt.title("Sales per month in the last year") plt.xlabel("months") plt.ylabel("Sales") plt.show() |
| --- |
|  |

This code will produce a line chart showing the amount of sales per month in the last year. The choice of a line chart is appropriate for showing trends over time, and the clear labels and legends make the information easy to interpret.

| plt.pie(sales, labels=months, autopct='%1.1f%%') plt.title('Monthly Sales') plt.show() |
| --- |
|  |

This code will generate a pie chart showing the ratio of sales for each month to total sales. The labels on the graph indicate the corresponding month and the percentage of sales it represents in relation to the total.

Another example would be, suppose we have a data set that represents the number of monthly sales for two different stores. We want to compare the sales of each store and see which one is more successful.

First, we create two data lists, one for each store:

| sales\_store1 = [12000, 15000, 18000, 20000, 21000, 23000, 25000, 27000, 28000, 30000, 32000, 34000] store\_sales2 = [10000, 13000, 14000, 17000, 19000, 21000, 24000, 25000, 27000, 28000, 29000, 31000] |
| --- |

We can then use the Python Matplotlib library to create a bar chart showing the monthly sales for each store.

| import matplotlib.pyplot as plt  months = range(1, 13)  plt.bar(months, sales\_store1, color='green', label='Shop 1') plt.bar(months, sales\_store2, color='blue', label='Shop 2')  plt.title('Monthly Sales of Store 1 and Store 2') plt.xlabel('we') plt.ylabel('Sales') plt.legend()  plt.show() |
| --- |
|  |

In this example, we create a list of numbers that represent the months of the year (1 through 12) and then use Matplotlib's bar function to create bars that represent the monthly sales for each store. We've also added a title, axis labels, and a legend to make the graph easier to understand.

The resulting graph will allow us to clearly visualize the monthly sales of each store and compare them easily.

In conclusion, data visualization is a fundamental part of data analysis, since it allows us to better understand the patterns and trends present in the data. With a good selection of charts and the right tools, we can present our findings clearly and effectively, making it easier to make data-driven decisions.