

A Study of Python Applications in Data Visualization

Yuan Yang^{1*}

¹ Media and Information Engineering College, Yunnan Open University, Kunming, Yunnan, China

Email Address

yangyuan@ynou.edu.cn (Yuan Yang)

*Correspondence: yangyuan@ynou.edu.cn

Received: 23 October 2023; Accepted: 13 November 2023; Published: 24 December 2023

Abstract:

This study explores the application of Python in data visualization, delving into popular visualization libraries and both static and dynamic visualization techniques. With the use of the Python programming language and a plethora of visualization libraries, we can effortlessly create various types of charts that showcase trends, correlations, distributions, and features within the data. In terms of static data visualization, widely utilized techniques include line charts, bar charts, scatter plots, histograms, box plots, heatmaps, and radar charts for data analysis and visualization. As for dynamic data visualization, techniques such as dynamic time series plots, real-time data monitoring graphs, interactive visualization, and animations excel in showcasing the dynamic changes and real-time conditions of the data. The application of Python in the field of data visualization provides robust tools and technical support for data scientists and data visualization professionals, while also opening up broader opportunities for future visualization research and innovation.

Keywords:

Python, Data visualization, Applications

1. Introduction

Data visualization is the practice of presenting data in a visual way to enhance understanding and uncover patterns, trends, and relationships within the data. As the volume of data and complexity of analytical tasks continue to increase, data visualization has become an integral and indispensable tool in the fields of data science and data analysis. Through our research, we aim to provide professionals with practical insights and technical guidance on using Python for data visualization. Additionally, we seek to foster the development of Python in the field of data visualization, providing strong support and reference for future research and innovative applications in data visualization. In this ever-changing era of data, data visualization will continue to play a crucial role in decision-making, problem-solving, and communication of information, with Python serving as a powerful tool to achieve these objectives.

Python is an advanced and versatile programming language, developed by the

Dutch programmer Guido van Rossum in the early 1990s. Python has gained widespread popularity for its simplicity, readability, and maintainability, making it a preferred language in fields such as data science, machine learning, and artificial intelligence. Python boasts numerous features that give it a unique advantage in the programming realm. Firstly, Python has a straightforward and concise syntax that is easy to learn and comprehend. It utilizes indentation to represent code blocks instead of relying on braces or keywords, resulting in code that is clearer and more structured. Secondly, Python is an interpreted language, meaning that code is executed line by line. This makes the development process more efficient, as the execution results can be immediately observed without the need to compile the entire program [1]. Additionally, Python benefits from a vast and active community that offers a wide range of third-party libraries and tools to cater to various domains and applications. For data visualization, libraries such as Matplotlib, Seaborn, and Plotly provide powerful plotting and visualization capabilities to users. Python also supports multiple programming paradigms, including object-oriented programming (OOP) and functional programming (FP), enabling flexible and scalable program design. In summary, Python's simplicity, powerful functionality, and robust community support have made it one of the primary programming languages in the fields of data visualization and data science. Its widespread adoption and growing user base contribute to a vibrant and innovative development landscape in the realm of data visualization.

2. Common Visualization Libraries

2.1. Pandas

Pandas, one of the most popular data processing and analysis libraries in Python, is also frequently used in data visualization. It provides powerful data structures and analysis tools, making data manipulation and visualization simpler and more efficient. The core data structure in pandas is the DataFrame, resembling a table in Excel, which allows for easy storage and manipulation of data. The DataFrame effortlessly handles structured and semi-structured data, offering various methods for slicing, filtering, merging, and grouping data. The pandas library also offers extensive functionalities for data manipulation and transformation, including data cleaning, handling missing values, data standardization, sorting, and reshaping. These features provide robust support for data preprocessing and analysis, enhancing the accuracy and credibility of data visualization. In terms of data visualization, combining pandas with other visualization libraries enables the convenient generation of various types of charts, such as line plots, bar plots, scatter plots, and box plots. Pandas provides a user-friendly plotting interface, allowing users to quickly create and customize their desired charts, showcasing trends, correlations, and distributions in the data. Besides basic chart plotting, pandas also supports further processing and optimization of data visualizations. Users can add titles, axis labels, and legends, set the range and style of axes, adjust the size and color of charts, etc. These flexible customization options enable users to create charts with a professional appearance and high readability [2]. To summarize, pandas serves as a powerful tool for data processing and analysis, offering rich functionality and flexibility for data visualization. Its concise syntax and wide range of applications make it a preferred data visualization library among data scientists and analysts. Through the application of pandas, users can conveniently explore and showcase data, providing deeper insights for data analysis and decision-making.

2.2. *Pyecharts*

Pyecharts is a Python-based data visualization library that provides rich chart types and interactive visualization capabilities, making data display more vivid and attractive. Based on the famous JavaScript visualization library ECharts, Pyecharts has been encapsulated and extended, enabling its integration into the Python environment. This makes it possible for users to generate and customize various types of charts through Python code without the need to delve into JavaScript. The chart types supported by Pyecharts include line charts, bar charts, scatter plots, geographic maps, heat maps, radar charts, etc. These chart types cover many common data visualization needs, and users can choose the appropriate chart type based on their data characteristics and display purposes. Pyecharts provides rich options for setting and style control, enabling users to customize chart titles, axis labels, legends, adjust colors, line styles, and sizes, etc. Users can also set the chart's interactive functions, such as zoom, drag, and add toolbars, to make the chart more interactive and operable. A significant feature is that Pyecharts supports generating static HTML files and dynamic Jupyter Notebook outputs. This allows users to save the generated charts as HTML files for easy display and sharing on websites. Meanwhile, using Pyecharts to generate charts in Jupyter Notebook can display charts and data in real-time feedback, facilitating interactive data exploration and analysis. In summary, Pyecharts provides Python users with a variety of chart types and interactive visualization functions, enabling the quick and flexible generation of attractive and effective data visualization charts. Its simplicity, ease of use, strong customizability, and good compatibility with the Python ecosystem make Pyecharts one of the preferred tools for data scientists, data analysts, and data visualization professionals.

2.3. *Spatial Visualization*

Spatial visualization is a technique that represents spatial data in the form of graphics or images. It aids in better understanding and analyzing geographical and spatial relationships, revealing patterns, trends, and anomalies within the data. Maps are one of the most common means of expression in spatial visualization. They display boundaries, topography, landmarks, and other relevant information related to geographic regions. Different attributes or data values of various regions are represented through color, labeling, and symbols. The interactivity and operability of maps allow users to further explore and analyze the data. Apart from maps, spatial visualization also encompasses other forms of charts and images, such as heat maps, grids, and 3D visualizations. Heat maps intuitively display the distribution and density of hotspots within an area, helping us discover phenomena like population clustering and event occurrence. Grids display continuous data such as terrain height and climate changes, aiding our understanding of geographic environment characteristics. 3D visualization, on the other hand, provides a more realistic and immersive spatial experience, enhancing our cognitive abilities to interpret and understand three-dimensional spatial data. Spatial visualization is widely applied not only in the field of geographic information systems (GIS) but also in urban planning, environmental science, transportation analysis, and logistics management, among many other domains [3]. Through spatial visualization, we can uncover trends and hotspots in urban development, analyze traffic flow and route selection, evaluate environmental quality and climate change, among others. These visualization techniques provide powerful support for decision-making and problem-solving, facilitating a deeper understanding and application of spatial data. In summary, spatial visualization is a

powerful tool that transforms complex spatial data into intuitive and easily comprehensible graphics and images. Through spatial visualization, we can better understand and analyze spatial relationships from a visual and perceptual standpoint, offering greater possibilities for the comprehension and application of geographical information.

3. Python in Static Data Visualization

3.1. *Line, Bar and Scatter Charts*

Line charts, bar charts, and scatter plots are some of the most commonly used types of charts in data visualization. They provide clear representations of trends and relationships within the data. A line chart is composed of lines connecting data points, often used to illustrate trends over time. It can showcase relationships between different datasets and easily incorporate elements such as titles, axis labels, legends, and annotations to enhance the information and readability of the chart. On the other hand, a bar chart presents data in the form of rectangular bars, typically used to compare differences between different datasets. It intuitively displays data distribution and comparisons in size, allowing for differentiation between datasets using colors, shading, and patterns. Similar to a line chart, a bar chart can be customized with titles, axis labels, legends, and other elements, enabling users to generate professional-looking charts quickly. Furthermore, a scatter plot is used to display the relationship between two variables. Each data point is represented as a point in a coordinate system and can be differentiated based on attributes such as color, size, and shape. Scatter plots can be used to explore correlations and trends within the data, such as relationships between units, concentration, and trends. Similarly, scatter plots can be customized with titles, axis labels, legends, annotations, and styles, enhancing clarity and readability. These types of charts have corresponding visualization libraries and functions in Python, such as matplotlib, seaborn, and pandas. Due to their ease of use and flexibility, Python has become one of the preferred tools for data scientists, data analysts, and developers. By utilizing these visualization libraries in Python, it is possible to quickly generate line charts, bar charts, and scatter plots with professional aesthetics and high readability, facilitating better understanding and analysis of data.

3.2. *Histograms and box plots*

Histograms and boxplots are commonly used types of charts in static data visualization. They are used to display the distribution and statistical features of data, helping us understand the central tendency, variability, and presence of outliers. Histograms represent the frequency distribution of numerical data using bars. By dividing the data into intervals (referred to as "bins"), histograms show the number or proportion of data within each interval. Not only do histograms show the distribution of data, but they also reveal characteristics such as symmetry, skewness, and kurtosis. In Python, libraries such as matplotlib and pandas can be used to create histograms. By adjusting parameters like the number of bins and the width of the bars, we can customize the appearance and representation of histograms. Boxplots, on the other hand, are common charts used to display data distribution. They are based on the five-number summary of data (minimum, first quartile, median, third quartile, and maximum) and consist of a box and two "whiskers" (or "worms") that represent the central tendency and variability of the data. The upper and lower boundaries of the box represent the third quartile and first quartile, respectively, while the line within

the box represents the median. The whiskers above and below the box represent the range of data. Boxplots can also display outliers by marking them separately. Libraries like seaborn and matplotlib in Python can be used to create boxplots. By adjusting styles, colors, and outlier display methods, we can customize clear and aesthetically pleasing boxplots to analyze and compare the distribution of multiple datasets. Both histograms and boxplots are important tools in data analysis and statistics. They help us understand the distribution characteristics and identify anomalies in our data. With the visualization libraries available in Python, we can conveniently create histograms and boxplots and utilize their rich range of parameters and options to customize the appearance and features of our charts, enabling better data representation and analysis [4].

3.3. Heatmaps and radar charts

Heatmaps and radar charts are commonly used chart types in static data visualization. They are used to represent data correlations and compare multidimensional data, respectively. A heatmap is a chart that uses color coding to display a data matrix. It represents the value of each position in the data matrix by coloring each cell, displaying differences in values through variations in color intensity and hue. Heatmaps are often used to visualize relationships and patterns in two-dimensional data, making them particularly suitable for exploring and analyzing large datasets. In Python, libraries such as seaborn and matplotlib can be used to create heatmaps. By adjusting parameters such as color mapping, cell size, and labels, we can customize heatmaps to have both readability and informative value, enhancing our understanding of data distribution and relationships. A radar chart (also known as a spider chart or polar chart) is a chart used to represent multidimensional data. It uses vertices and edges of a polygon to represent data features, with each vertex corresponding to a data dimension and the length of each edge representing the magnitude of that dimension's value. By connecting the vertices and filling the area enclosed by the polygon, a radar chart intuitively compares multidimensional data across different datasets, highlighting differences and trends. Radar charts are commonly used for visualizing and evaluating multiple indicators. In Python, the matplotlib library can be utilized to create radar charts. By adjusting parameters such as angles, labels, and styles, we can customize radar charts to achieve clarity and aesthetics, effectively showcasing and comparing features and indicators across multiple datasets. Both heatmaps and radar charts are valuable tools in data analysis and visualization. They help us discover patterns, correlations, and differences within data. Leveraging the visualization libraries provided by Python, we can easily create, customize, and effectively showcase heatmaps and radar charts to analyze and depict data features and trends according to our specific needs.

4. Applications of Python in Dynamic Data Visualization

The application of Python in dynamic data visualization is becoming increasingly widespread, primarily due to its powerful data processing and visualization libraries. Dynamic data visualization not only presents the static features of data but also showcases the trends and dynamic changes over time. Furthermore, it allows for interactive manipulation and exploration of data, enabling users to gain a deeper understanding of the characteristics and patterns within the data. One of the most popular dynamic visualization libraries in Python is matplotlib, which provides APIs such as animation, FuncAnimation, and ArtistAnimation for real-time drawing of

animations. These APIs can be utilized to create dynamic charts, including line charts, scatter plots, bar charts, and heat maps, among others. By continuously plotting the various components of a chart, using these matplotlib APIs, we can effectively showcase the changes and trends in various data sets. Compared to static charts, dynamic charts are more effective in visualizing the dynamic variations in data, such as the trends in time series, relationships between multiple variables, and distributions. In addition to matplotlib, Python also has other popular dynamic visualization libraries, such as bokeh and Plotly. Bokeh is an interactive visualization library based on Python, which can generate dynamic charts using the JavaScript language. It can be utilized for graphing, presentations, reports, and more. Plotly, on the other hand, is an online data visualization tool based on Python. It allows users to upload datasets to their website and share them with others. In Python environments, Plotly supports high-end visualization applications, including dynamic visualization, interactive visualization, and 3D charts. Python's dynamic data visualization tools provide robust support for data analysis and interactive data exploration. They assist data scientists, analysts, and researchers in better understanding the dynamic changes and trends within data sets, as well as discovering new patterns and insights in real-time data processing and analysis. From the monitoring of time series data and industrial process data analysis to the processing and presentation of scientific experimental data, Python's dynamic data visualization libraries offer rich functionality and flexible customization options to meet various requirements in data analysis and visualization [5].

5. Conclusions

Research on the application of Python in data visualization is a vibrant and rapidly developing field. With the rise of data analysis and data science, Python has become the preferred programming language for many data professionals and researchers, driving the continuous development and improvement of data visualization tools and libraries. The application research of Python in data visualization can help people better understand and interpret data, revealing patterns, correlations, and trends within the data. Python provides a wealth of data processing and visualization libraries, such as matplotlib, seaborn, plotly, and bokeh, which are flexible and scalable, and can meet various data visualization needs. Using these tools, we can generate various types of images, charts, and dynamic charts, and display data in an intuitive and interactive way. Data visualization not only helps us better understand data but also conveys complex information to audiences and decision-makers in a visual way. Through the application research of Python in data visualization, we can combine data analysis and visualization, discover the stories and insights behind the data. These insights can be applied to various fields such as business decision-making, scientific research, and social analysis, and play an important role in helping to solve practical problems and make more informed decisions. In the future, there are still many challenges and opportunities for the application research of Python in data visualization. With the increase in data size and diversity, we need more powerful and efficient tools and algorithms to process and visualize data. At the same time, with the development of technologies such as deep learning and artificial intelligence, Python's data visualization can also be combined with these fields to explore more new methods and applications. In summary, the application research of Python in data visualization provides us with powerful tools and methods to understand and display data. Through continuous exploration, innovation, and research, we can further improve the effectiveness and quality of data visualization and provide more accurate,

intuitive, and effective visual solutions for data analysis and decision-making.

Conflicts of Interest

The author declares that there is no conflict of interest regarding the publication of this article.

Funding

This work was supported by the Education Science Planning Research Fund of Yunnan Province in 2023 “Research and Practice of Constructing the Talent Training Model of ‘Post Course Competition Certificate Integration’ Based on 1+X Certificate Standard” (No.: BE22019).

References

- [1] D. A. Kapustin.; V. V. Shvyrov.; T. I. Shulika. Static Analysis of Corpus of Source Codes of Python Applications. *Programming and Computer Software*, 2023, 49(4), 11.
- [2] Manolis, K.; Lazaros, P.; Mario, K, et al. A memory footprint optimization framework for Python applications targeting edge devices. *Journal of Systems Architecture*, 2023, 1, 142.
- [3] Themistoklis, K.; Georgios, X.; Konstantinos, K , et al. In-Cognitive: A web-based Python application for fuzzy cognitive map design, simulation, and uncertainty analysis based on the Monte Carlo method. *Software X*, 2023, 2, 23.
- [4] Zona, K.; Catherine, D.; Sarah, P, et al. Exploring Mid-Air Hand Interaction in Data Visualization. *IEEE transactions on visualization and computer graphics*, 2023, 1, 11-13.
- [5] Henrique, M.; Nuno, L.; Jorge, B. Injecting software faults in Python applications. *Empirical Software Engineering*, 2021, 27(1), 22-24.



© 2023 by the author(s); licensee International Technology and Science Publications (ITS), this work for open access publication is under the Creative Commons Attribution International License (CC BY 4.0). (<http://creativecommons.org/licenses/by/4.0/>)