Architecture Design
Amazon Sales Data Analysis

Revision Number: 1.0 Last
Date of revision: 10/2/2024

Bodapatla Vinay Kumar Reddy

Document Version Control

Date Issued	Version	Description	Author
10th Feb 2024	1.0	First Version	Vinay Kumar Reddy Bodapatla

Contents

1. Introduction

1.1 What is an Architecture design document?

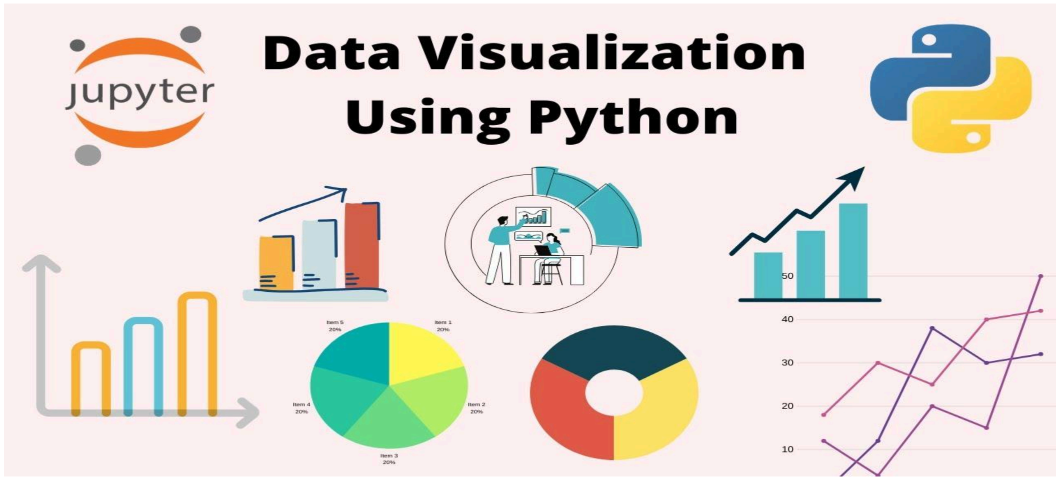
It is a document that provides a comprehensive description of the software architecture of a system. It defines the high-level design decisions, choices, and trade-offs made during the development of a software system, and serves as a reference for developers, stakeholders, and other members of the project team. Each style will describe a system category that consists of:

- A set of components (e.g.: a database, computational modules) that will perform a function required by the system.
- The set of connectors will help in coordination, communication, and cooperation between the components.
- Conditions that how components can be integrated to form the system.
- Semantic models that help the designer to understand the overall properties of the system.

1.2 Scope

Architecture Design Document is an architecture design process that follows a step-by-step refinement process. The process can be used for designing data structures, required software architecture, source code and ultimately, performance algorithms. Overall, the design principles may be defined during requirement analysis and then refined during architectural design work

2. Architecture of Pandas, NumPy, MatPlotlib, SeaBorn



NumPy Architecture Design Document:

Introduction:
Overview of NumPy as a numerical computing library for Python.

Architectural Goals and Objectives:
Provide efficient and flexible array operations for numerical computing. Facilitate vectorized operations and broadcasting.

Architectural Overview:
Description of N-dimensional array as the fundamental data structure. Explanation of universal functions (ufunc) for element-wise operations.

Key Design Decisions:
Adoption of N-dimensional array for efficient array manipulation. Implementation of universal functions for vectorized operations.

System Components:
N-dimensional array as the core component for storing and manipulating arrays. Universal functions for element-wise operations.

Data Design:
Discussion on how NumPy efficiently manages memory layout. Support for homogeneous, fixed-size arrays.

Performance Considerations:
Emphasis on vectorized operations for improved performance. Memory layout optimizations.

Scalability and Extensibility:
Discuss how NumPy supports scalability through efficient array operations. Extensibility through integration with other scientific libraries.

Matplotlib Architecture Design Document:

Introduction:
Overview of Matplotlib as a 2D plotting library for Python.

Architectural Goals and Objectives:
Provide a versatile and customizable plotting framework. Support diverse plot types and visualizations.

Architectural Overview:
Description of Figure and Axes structure. Explanation of backend rendering.

Key Design Decisions:
Adoption of a layered design for customization. Support for multiple backends for rendering.

System Components:
Figure as the top-level container. Axes representing individual plots within a Figure.

Data Design:
Handling and plotting of data using high-level functions. Data visualization through a variety of plot types.

Performance Considerations:
Backend rendering optimizations. Layered design for efficient customization.

Scalability and Extensibility:
Support for creating multiple plots within a single Figure. Customization and extensibility through high-level and low-level components.

Seaborn Architecture Design Document:

Introduction:
Overview of Seaborn as a statistical data visualization library for Python.

Architectural Goals and Objectives:
Simplify the creation of informative statistical plots. Enhance aesthetics and styling compared to Matplotlib.

Architectural Overview:
Built on top of Matplotlib. Integration with Pandas DataFrames.

Key Design Decisions:
Focus on statistical plotting functions. Aesthetic enhancements and built-in themes.
System Components:
Utilization of Matplotlib components for plotting. Statistical plotting functions specific to Seaborn.
Data Design:
Seamless integration with Pandas DataFrames. Statistical plotting for deeper insights.

Performance Considerations:
Leveraging Matplotlib backend rendering optimizations. Aesthetic enhancements without sacrificing performance.

Scalability and Extensibility:
Integration with Pandas for handling tabular data. Customization and extensibility through Matplotlib's capabilities.

3. Data Visualization:

Data visualization is the discipline of trying to understand data by placing it in a visual context so that patterns, trends and correlations that might not otherwise be detected can be exposed.Python offers multiple great graphing libraries that come packed with lots of different features. No matter if you want to create interactive, live or highly customized plots python has an excellent library for you.

To get a little overview here are a few popular plotting libraries:

- Matplotlib: low level, provides lots of freedom
- Pandas Visualization: easy to use interface, built on Matplotlib
- Seaborn: high-level interface, great default styles
- ggplot: based on R's ggplot2

Matplotlib

Matplotlib is the most popular python plotting library. It is a low-level library with a Matlab-like interface which offers lots of freedom at the cost of having to write more code.Matplotlib is specifically good for creating basic graphs like line charts, bar charts, histograms and many more.

Scatter Plot

To create a scatter plot in Matplotlib we can use the scatter method. We will also create a figure and an axis using plt.subplots so we can give our plot a title and labels.We can give the graph more meaning by coloring in each data-point by its class. This can be done by creating a dictionary which maps from class to color and then scattering each point on its own using a for-loop and passing the respective color.

Line Chart

In Matplotlib we can create a line chart by calling the plot method. We can also plot multiple columns in one graph, by looping through the columns we want and plotting each column on the same axis.

Histogram

In Matplotlib we can create a Histogram using the hist method. If we pass categorical data like the points column from the wine-review dataset it will automatically calculate how often each class occurs.

Bar Chart

A bar chart can be created using the bar method. The bar-chart isn't automatically calculating the frequency of a category so we are going to use pandas value_counts function to do this. The bar-chart is useful for categorical data that doesn't have a lot of different categories (less than 30) because else it can get quite messy.

Pandas Visualization

Pandas is an open source high-performance, easy-to-use library providing data structures, such as dataframes, and data analysis tools like the visualization tools we will use in this article.

Seaborn

Seaborn is a Python data visualization library based on Matplotlib. It provides a high-level interface for creating attractive graphs.Seaborn has a lot to offer. You can create graphs in one line that would take you multiple tens of lines in Matplotlib. Its standard designs are awesome and it also has a nice interface for working with pandas dataframes.

Scatter plot

We can use the .scatterplot method for creating a scatterplot, and just as in Pandas we need to pass it the column names of the x and y data, but now we also need to pass the data as an additional argument because we aren’t calling the function on the data directly as we did in Pandas.

Line chart

To create a line-chart the sns.lineplot method can be used. The only required argument is the data, which in our case are the four numeric columns from the Iris dataset. We could also use the sns.kdeplot method which rounds of the edges of the curves and therefore is cleaner if you have a lot of outliers in your dataset.

Histogram

To create a histogram in Seaborn we use the sns.distplot method. We need to pass it the column we want to plot and it will calculate the occurrences itself. We can also pass it the number of bins, and if we want to plot a gaussian kernel density estimate inside the graph.
Bar chart
In Seaborn a bar-chart can be created using the sns.countplot method and passing it the data.