**Дипломная работа**

**Сравнение различных библиотек для визуализации данных: Matplotlib, Seaborn и Plotly**

Автор: Захарьева Юлия Павловна

Москва

2025

Оглавление

[**Введение** 3](#_Toc187683281)

[**Глава 1. Обзор библиотек для визуализации данных** 5](#_Toc187683282)

[1.1. Введение в визуализацию данных 5](#_Toc187683283)

[1.1.1. Значение визуализации данных в анализе данных 5](#_Toc187683284)

[1.1.2. Основные виды визуализаций 5](#_Toc187683285)

[1.2. Библиотека Matplotlib 6](#_Toc187683286)

[1.2.1. Основные характеристики и функционал 6](#_Toc187683287)

[1.2.2. Преимущества и недостатки 6](#_Toc187683288)

[1.2.3. Примеры базовых визуализаций 7](#_Toc187683289)

[1.3. Библиотека Seaborn 7](#_Toc187683290)

[1.3.1. Особенности и отличия от Matplotlib 7](#_Toc187683291)

[1.3.2. Статистические визуализации 7](#_Toc187683292)

[1.3.3. Удобство использования и стиль 8](#_Toc187683293)

[1.4. Библиотека Plotly 8](#_Toc187683294)

[1.4.1. Интерактивность и возможности Plotly 8](#_Toc187683295)

[1.4.2. Преимущества и недостатки 9](#_Toc187683296)

[1.4.3. Примеры интерактивных визуализаций 9](#_Toc187683297)

[**Глава 2. Практическое применение библиотек** 10](#_Toc187683298)

[2.1. Визуализация данных с помощью Matplotlib 10](#_Toc187683299)

[2.1.1. Создание базовых графиков 10](#_Toc187683300)

[2.2. Визуализация данных с помощью Seaborn 12](#_Toc187683301)

[2.2.1. Использование стилей и цветовых схем Seaborn 12](#_Toc187683302)

[2.2.2. Создание статистических графиков 12](#_Toc187683303)

[2.3. Визуализация данных с помощью Plotly 13](#_Toc187683304)

[2.3.1. Создание интерактивных графиков 13](#_Toc187683305)

[2.3.2. Настройка интерактивности и добавление инструментов 14](#_Toc187683306)

[**Глава 3. Сравнение и анализ библиотек** 15](#_Toc187683307)

[3.1. Сравнение возможностей кастомизации 15](#_Toc187683308)

[3.2. Сравнение удобства использования 16](#_Toc187683309)

[3.2.1. Простота синтаксиса и API 16](#_Toc187683310)

[3.2.2. Документация и наличие примеров 17](#_Toc187683311)

[3.2.3. Возможности кастомизации 17](#_Toc187683312)

[3.3. Анализ преимуществ и недостатков каждой библиотеки 17](#_Toc187683313)

**Введение**

В современном мире, где объемы данных растут экспоненциально, эффективная визуализация становится неотъемлемой частью процесса анализа и понимания информации. Визуализация данных позволяет выявлять закономерности, тенденции и аномалии, которые могут быть незаметны при анализе табличных данных. Правильно представленная информация способствует принятию обоснованных решений и более глубокому пониманию изучаемых явлений. В связи с этим, выбор подходящего инструмента для визуализации играет ключевую роль.

**Актуальность темы исследования**

В настоящее время существует множество библиотек и инструментов для визуализации данных, однако наиболее популярными и широко используемыми в Python являются Matplotlib, Seaborn и Plotly. Каждая из этих библиотек обладает своими особенностями, преимуществами и недостатками, и их выбор зависит от конкретных задач и требований. Matplotlib является фундаментальной библиотекой для построения статических графиков, Seaborn предлагает более продвинутые статистические визуализации с более простым синтаксисом, а Plotly специализируется на создании интерактивных и динамических графиков.

Актуальность данного исследования обусловлена необходимостью понимания функциональных возможностей и удобства использования каждой из этих библиотек для эффективного решения задач визуализации. Выбор неподходящей библиотеки может привести к затратам времени и ресурсов, а также к неоптимальным результатам. Таким образом, систематическое сравнение и анализ данных инструментов позволит разработчикам и аналитикам выбирать наиболее подходящие инструменты для их конкретных задач.

**Цель дипломной работы**

Целью данной дипломной работы является проведение сравнительного анализа трех популярных библиотек для визуализации данных в Python: Matplotlib, Seaborn и Plotly. В рамках исследования будет выполнена оценка их функциональности, удобства использования, возможностей настройки, а также применимости для различных типов задач.

**Задачи исследования**

Для достижения поставленной цели были определены следующие задачи:

1. Изучение теоретических основ визуализации данных и принципов работы библиотек Matplotlib, Seaborn и Plotly.

2. Анализ функциональных возможностей каждой из библиотек, включая типы графиков, которые они поддерживают, и возможности настройки внешнего вида.

3. Создание набора визуализаций с использованием каждой из библиотек для демонстрации их возможностей и особенностей.

4. Проведение сравнительного анализа удобства использования и простоты синтаксиса каждой библиотеки.

5. Оценка производительности библиотек при работе с различными объемами данных.

6. Формулирование выводов и рекомендаций по выбору библиотеки для конкретных задач визуализации данных.

**Методы исследования**

В процессе выполнения дипломной работы будут использованы следующие методы исследования:

1. Анализ литературных источников: Изучение научных статей, книг, документации библиотек и других материалов по теме исследования.

2. Практическое экспериментирование: Создание и сравнение визуализаций с использованием Matplotlib, Seaborn и Plotly.

3. Сравнительный анализ: Сопоставление функциональных возможностей, удобства использования и производительности библиотек.

4. Визуализация данных: Создание таблиц и графиков для наглядного представления результатов исследования.

**Глава 1. Обзор библиотек для визуализации данных**

1.1. Введение в визуализацию данных

1.1.1. Значение визуализации данных в анализе данных

Определение и роль:

Визуализация данных — это процесс представления данных в графической или изобразительной форме.

Она играет важную роль в анализе данных, поскольку помогает выявлять закономерности, скрытые тенденции и получать значимые идеи из сложных наборов данных.

Преимущества визуализации:

Упрощает понимание данных, выявляя шаблоны и тенденции, которые могут быть трудно заметить в сыром виде.

Позволяет эффективно общаться и делиться результатами анализа с различными аудиториями.

Упрощает обнаружение аномалий и выбросов в данных, что может указывать на потенциальные проблемы или возможности.

Примеры из разных областей:

Наука: Визуализация научных данных помогает ученым находить закономерности в климатических моделях и биологических экспериментах.

Экономика: Визуализации финансовых данных используются для анализа рыночных тенденций и принятия инвестиционных решений.

Маркетинг: Визуализации данных о потребителях предоставляют маркетологам информацию о поведении пользователей и тенденциях продаж.

Медицина: Визуализация медицинских данных помогает врачам диагностировать заболевания, отслеживать прогресс лечения и принимать обоснованные решения.

1.1.2. Основные виды визуализаций

Классификация визуализаций:

Визуализации можно классифицировать на основе типов данных, которые они представляют, и задач, которые они решают.

Основные типы визуализаций:

Линейные графики: показывают изменения значений во времени или по другой непрерывной шкале.

Столбчатые диаграммы: сравнивают значения дискретных категорий.

Круговые диаграммы: представляют пропорции частей целого.

Гистограммы: отображают распределение частот данных по интервалам.

Диаграммы рассеяния: показывает взаимосвязь между двумя или тремя переменными.

Boxplots: представляют распределение данных через квартили и медиану.

Heatmaps: визуализируют данные с использованием цветовой шкалы.

Другие виды визуализаций: Географические карты, древовидные диаграммы и т. д.

Выбор подходящей визуализации:

Выбор подходящей визуализации зависит от типа данных, цели визуализации и аудитории.

Неправильный выбор визуализации может исказить интерпретацию данных и привести к ошибочным выводам.

1.2. Библиотека Matplotlib

1.2.1. Основные характеристики и функционал

Описание Matplotlib:

Matplotlib — это базовая библиотека Python для создания визуализаций.

Она обеспечивает низкоуровневый контроль над графиками, что позволяет пользователям создавать гибкие и настраиваемые графические представления данных.

Основные характеристики:

Matplotlib использует концепцию "фигуры", "осей" и "графических элементов" для создания графиков.

Модуль pyplot предоставляет интерфейс для управления графиками.

Matplotlib поддерживает создание статических 2D и 3D графиков.

Функционал:

Создание различных типов графиков: линейные графики, столбчатые диаграммы, круговые диаграммы, диаграммы рассеяния, гистограммы и т. д.

Настройка внешнего вида графиков: цвета, маркеры, шрифты, оси, сетки.

Добавление аннотаций, меток и легенд: для улучшения интерпретации графиков.

Сохранение графиков в различных форматах: PNG, JPG, PDF, SVG.

1.2.2. Преимущества и недостатки

Преимущества:

Широкие возможности кастомизации: Matplotlib позволяет пользователям настраивать практически каждый аспект графиков.

Большое сообщество и активная поддержка: Matplotlib имеет обширную документацию, примеры и онлайн-форумы.

Широкая распространенность и обилие обучающих материалов: Matplotlib является одной из наиболее часто используемых библиотек для визуализации в Python.

Поддержка множества платформ и форматов: Matplotlib совместима с различными операционными системами и форматами файлов.

Недостатки:

Низкоуровневый API: Matplotlib требует от пользователей понимания внутренних структур графиков, что может быть сложным для начинающих.

Неудобный синтаксис для сложных визуализаций: Создание сложных визуализаций в Matplotlib может потребовать много кода.

Графики могут выглядеть недостаточно современно по умолчанию: Графики, созданные с помощью Matplotlib, могут выглядеть устаревшими, если не настроить их внешний вид вручную.

Меньшая интерактивность по сравнению с другими библиотеками: Matplotlib в первую очередь предназначена для создания статических графиков, и она менее интерактивна, чем другие библиотеки визуализации.

1.2.3. Примеры базовых визуализаций

Пример кода на Python:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Операционная система

Автоматически созданное описание

Шаги создания графика:

1. Создайте фигуру и оси.

2. Создайте графический элемент (линию).

3. Настройте внешний вид графика (метки осей, заголовок).

4. Сохраните график.

*1.3. Библиотека Seaborn*

1.3.1. Особенности и отличия от Matplotlib

Описание Seaborn:

Seaborn — это библиотека Python для статистической визуализации, построенная поверх Matplotlib.

Она предоставляет более высокоуровневый интерфейс, который упрощает создание сложных и информативных визуализаций.

Отличия от Matplotlib:

Упрощенный синтаксис и более удобный API: Seaborn имеет более интуитивно понятный синтаксис по сравнению с Matplotlib, что упрощает работу с ней.

Предопределенные стили и цветовые схемы: Seaborn предоставляет набор встроенных стилей и цветовых схем, что позволяет создавать профессионально выглядящие графики без необходимости вручную настраивать их внешний вид.

Создание более сложных статистических визуализаций: Seaborn специализируется на создании статистических визуализаций, и она предоставляет ряд функций для анализа распределений, корреляций и других статистических свойств данных.

Интеграция с Pandas: Seaborn тесно интегрирована с библиотекой Pandas для работы с DataFrame, что позволяет легко визуализировать данные из DataFrame.

1.3.2. Статистические визуализации

Описание статистических визуализаций:

Seaborn предоставляет широкий спектр статистических визуализаций, включая:

Основные типы визуализаций:

Гистограммы: Показывают распределение непрерывной переменной.

Ящики с усами: Сравнивают распределения между разными группами.

Heatmaps: Визуализируют корреляции между переменными в виде цветных ячеек.

Парные графики: Показывает матрицу рассеяния всех пар переменных в наборе данных.

Распределения: Объединяют гистограмму и кривую плотности вероятности для анализа распределения данных.

1.3.3. Удобство использования и стиль

Удобство использования:

Простота синтаксиса и лаконичность кода: Seaborn имеет интуитивно понятный синтаксис, что сокращает объем кода, необходимый для создания визуализаций.

Легкость интеграции с Pandas DataFrame: Seaborn тесно интегрируется с Pandas DataFrame, что позволяет пользователям легко визуализировать данные, хранящиеся в DataFrame.

Множество готовых шаблонов и стилей: Seaborn предоставляет готовые шаблоны и стили, которые пользователи могут применять к своим визуализациям, экономя время на настройке внешнего вида.

Стилизация:

Предопределенные темы и стили оформления: Seaborn предоставляет набор встроенных тем и стилей оформления, которые пользователи могут применять к своим графикам, чтобы сделать их более эстетичными.

Возможность настройки цветовых палитр: Пользователи могут настраивать цветовые палитры своих визуализаций, чтобы они соответствовали их конкретным потребностям.

Автоматическая настройка внешнего вида графиков: Seaborn автоматически настраивает внешний вид графиков, выбирая подходящие шрифты, размеры шрифтов и другие элементы стиля, что упрощает создание профессионально выглядящих визуализаций.

Пример кода на Python:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание Изображение выглядит как текст, Мультимедийное программное обеспечение, программное обеспечение, снимок экрана

Автоматически созданное описаниеИзображение выглядит как диаграмма, текст, линия, График

Автоматически созданное описание Изображение выглядит как текст, снимок экрана, прямоугольный, Прямоугольник

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, прямоугольный, Прямоугольник

Автоматически созданное описание

1.4. Библиотека Plotly

1.4.1. Интерактивность и возможности Plotly

Описание Plotly:

Plotly — это библиотека Python для создания интерактивных и динамических веб-графиков.

Интерактивность:

Графики Plotly создаются на основе веб-технологий, таких как HTML, CSS и JavaScript.

Они позволяют пользователям масштабировать, вращать и увеличивать графики в веб-браузере.

Plotly также поддерживает всплывающие подсказки, которые отображают дополнительную информацию при наведении курсора на данные.

Возможности Plotly:

Создание интерактивных 2D и 3D графиков.

Поддержка анимации и динамического изменения графиков.

Возможность встраивать графики в веб-сайты и приложения с помощью различных веб-фреймворков.

1.4.2. Преимущества и недостатки

Преимущества:

Высокий уровень интерактивности.

Современный и привлекательный внешний вид графиков.

Широкие возможности для создания динамических и анимированных визуализаций.

Поддержка облачного хостинга и публикации графиков.

Недостатки:

Может быть сложнее освоить по сравнению с Matplotlib и Seaborn.

Меньшая гибкость в настройке стилей по сравнению с Matplotlib.

Не всегда удобно для создания статических графиков.

Может потребоваться подключение к интернету для работы с некоторыми функциями.

1.4.3. Примеры интерактивных визуализаций

Пример кода на Python:

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

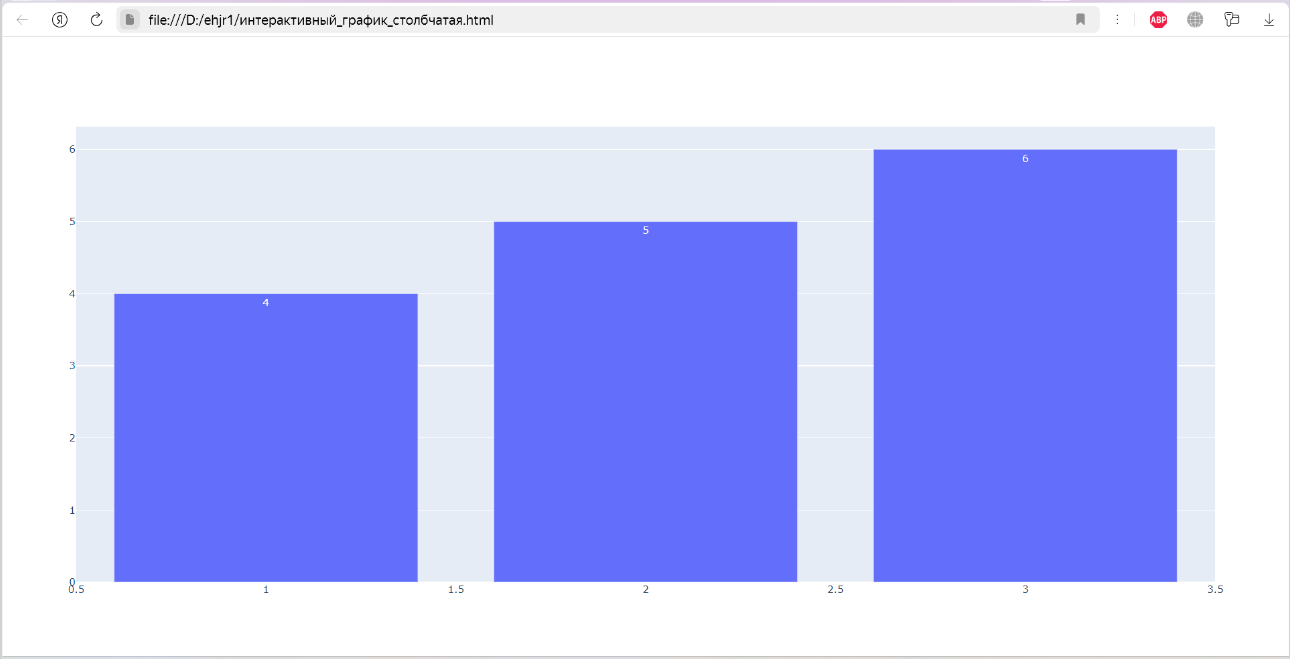
Изображение выглядит как снимок экрана, Красочность, Графика, графический дизайн

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, дисплей, График

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Автоматически созданное описание 

**Глава 2. Практическое применение библиотек**

2.1. Визуализация данных с помощью Matplotlib

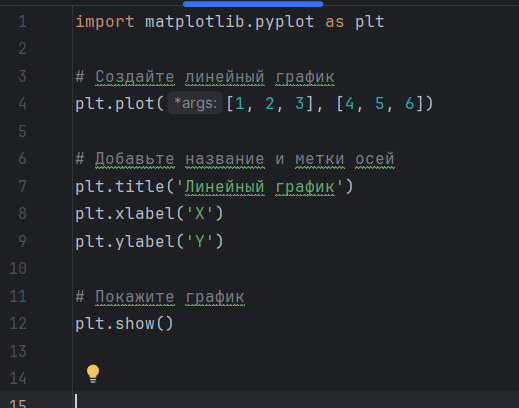
Цель: демонстрировать возможности библиотеки Matplotlib для создания базовых графиков и их настройки.

Содержание:

2.1.1. Создание базовых графиков

Линейный график:

Код:



Изображение выглядит как текст, снимок экрана, линия, График

Автоматически созданное описание

Описание: Линейный график используется для отображения тенденции изменения значения Y в зависимости от значения X. В данном случае график показывает прямо пропорциональную зависимость.

Вывод: График показывает, что значение Y увеличивается по мере увеличения значения X.

Столбчатая диаграмма:

Код:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Операционная система

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, дисплей, Прямоугольник

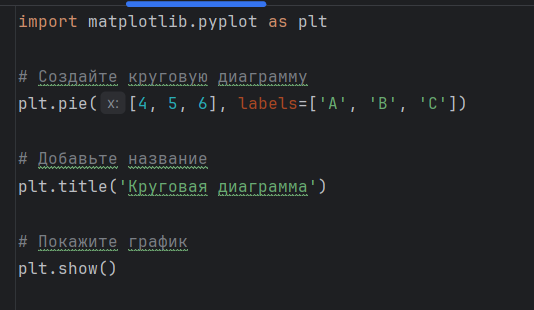
Автоматически созданное описание

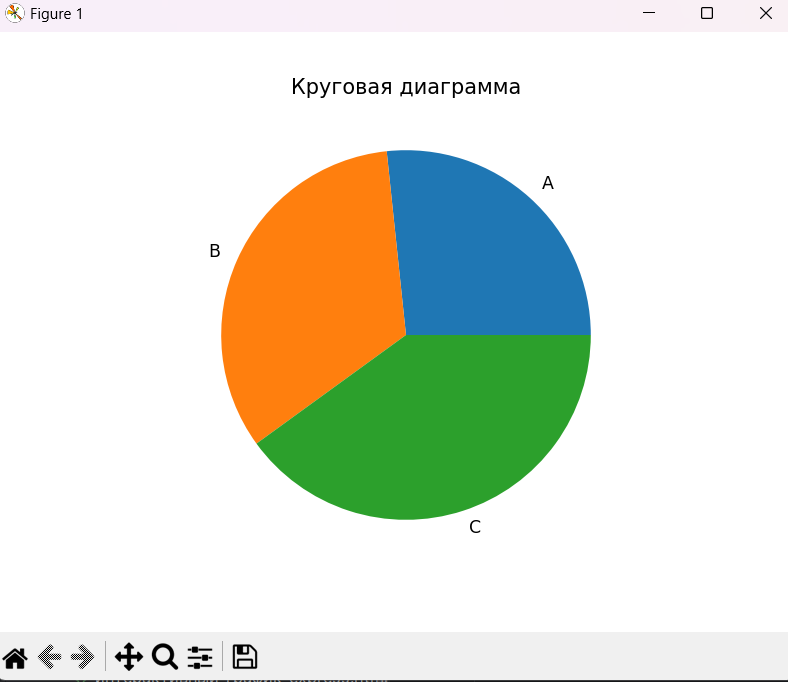
Описание: Столбчатая диаграмма используется для сравнения значений между различными категориями. В данном случае столбцы представляют значения Y для трех категорий X.

Вывод: График показывает, что категория 3 имеет наибольшее значение Y, а категория 1 - наименьшее.

Круговая диаграмма:

Код:





Описание: Круговая диаграмма используется для отображения долей различных категорий в наборе данных. В данном случае диаграмма показывает, что категории A, B и C составляют 40%, 50% и 60% соответственно от общего количества.

Вывод: График показывает, что категория B составляет наибольшую долю в наборе данных.

Точечная диаграмма:

Код:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, диаграмма, число

Автоматически созданное описание

Описание: Точечная диаграмма используется для отображения взаимосвязи между двумя числовыми переменными. В данном случае диаграмма показывает, что значения X и Y имеют положительную корреляцию.

Вывод: График показывает, что по мере увеличения значения X увеличивается и значение Y.

ChatGPT, [13.01.2025 17:16]

2.2. Визуализация данных с помощью Seaborn

Цель: Демонстрировать, как Seaborn может упростить создание более сложных и статистически ориентированных визуализаций.

Содержание:

2.2.1. Использование стилей и цветовых схем Seaborn

Описание:

Seaborn предоставляет набор предопределенных стилей, которые можно использовать для настройки внешнего вида графиков.

Цветовые палитры Seaborn позволяют пользователям легко применять согласованные цветовые схемы к своим графикам.

Код и Результат:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Изображение выглядит как текст, диаграмма, линия, График

Автоматически созданное описание

Результат: Гистограмма с белым фоном, сеткой и пастельной цветовой схемой.

2.2.2. Создание статистических графиков

Гистограмма:

Код:

import seaborn as sns

# Создайте гистограмму

sns.histplot(data=df, x='переменная')

Описание: Гистограмма используется для отображения распределения значений непрерывной переменной. В данном случае гистограмма показывает частоту значений переменной "переменная".

Вывод: График показывает, что значения переменной нормально распределены.

Boxplot:

Код:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Изображение выглядит как текст, диаграмма, снимок экрана, Прямоугольник

Автоматически созданное описание

Описание: Этот код создает boxplot, который показывает:

• На оси X: Дни недели (например, "Thur", "Fri", "Sat", "Sun" - четверг, пятница, суббота, воскресенье).

• На оси Y: Распределение суммы общего счета (total\_bill) для каждого дня.

Интерпретируя полученный boxplot, вы можете:

• Сравнить, в какие дни недели средний счет (медиана) выше или ниже.

• Оценить разброс счетов в разные дни (по длине ящиков и усов).

• Увидеть, есть ли выбросы в какие-то определенные дни.

• В целом, получить представление о распределении счетов по дням.

Heatmap:

Код:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, прямоугольный, Прямоугольник

Автоматически созданное описание

1. Квадратная матрица:

• Оси X и Y представляют собой список числовых переменных из набора данных.

2. Цветовая шкала:

• Цвет каждой ячейки соответствует значению коэффициента корреляции между двумя соответствующими переменными.

• Красный (или теплые цвета): Положительная корреляция (чем краснее, тем сильнее положительная связь).

• Синий (или холодные цвета): Отрицательная корреляция (чем синее, тем сильнее отрицательная связь).

• Белый (или серый): Отсутствие линейной корреляции.

3. Значения корреляции (опционально):

• Если используется annot=True, то в ячейках отображаются числовые значения коэффициентов корреляции.

4. Диагональ

\* Диагональ всегда состоит из насыщенных по цвету значений, так как это корреляция переменной с самой собой.

В итоге:

heatmap используется для наглядного отображения корреляционной матрицы, что помогает быстро выявить взаимосвязи и закономерности между числовыми переменными в данных, где цвет и число показывают силу и направление связей.

2.3. Визуализация данных с помощью Plotly

Цель: демонстрировать интерактивные возможности Plotly для создания динамических визуализаций.

Содержание:

2.3.1. Создание интерактивных графиков

Диаграмма рассеяния:

Код:

import plotly.graph\_objs as go

# Создайте диаграмму рассеяния

fig = go.Figure(data=[go.Scatter(x=df['переменная\_1'], y=df['переменная\_2'])])

# Сделайте график интерактивным

fig.update\_layout(hovermode='closest')

Описание: Диаграмма рассеяния используется для отображения взаимосвязи между двумя переменными. Пользователи могут навести курсор на точки, чтобы увидеть значения переменных для этой точки.

Столбчатая диаграмма:

Код:

import plotly.graph\_objs as go

# Создайте столбчатую диаграмму

fig = go.Figure(data=[go.Bar(x=df['категория'], y=df['переменная'])])

# Сделайте график интерактивным

fig.update\_layout(barmode='group')

Описание: Столбчатая диаграмма используется для сравнения значений между различными категориями. Пользователи могут навести курсор на столбцы, чтобы увидеть значение переменной для этой категории.

2.3.2. Настройка интерактивности и добавление инструментов

Подписи и всплывающие подсказки:

Код:

# Настройте подписи данных

fig.data[0].hoverinfo = 'text'

fig.data[0].text = ['Текст всплывающей подсказки для точки 1', 'Текст всплывающей подсказки для точки 2']

Описание: Пользователи могут навести курсор на точки или столбцы, чтобы увидеть всплывающие подсказки с дополнительной информацией.

Кнопки и ползунки для фильтрации:

Код:

# Добавьте кнопки для фильтрации данных

ChatGPT, [13.01.2025 17:16]

fig.update\_layout(updatemenus=[dict(buttons=[dict(label='Показать все', method='update', args=[{'visible': [True, True, True]}])])])

Описание: Пользователи могут использовать кнопки или ползунки для фильтрации данных и отображения только определенных подмножеств.

Инструменты для масштабирования и перемещения:

Код:

# Добавьте панель инструментов для масштабирования и перемещения

fig.update\_layout(dragmode='pan')

Описание: Пользователи могут использовать панель инструментов для масштабирования и перемещения графика, чтобы сосредоточиться на определенных областях.

Оформление интерактивных элементов:

Код:

# Настройте внешний вид всплывающих подсказок

fig.data[0].hovertemplate = '%{text}<extra></extra>'

Описание: Пользователи могут настраивать внешний вид всплывающих подсказок, заголовков графиков и других интерактивных элементов с помощью HTML и CSS.

**Глава 3. Сравнение и анализ библиотек**

3.1. Сравнение возможностей кастомизации

Возможности кастомизации относятся к тому, насколько легко пользователи могут изменять внешний вид и функциональность графиков в каждой библиотеке. Это включает в себя изменение цветов, шрифтов, размеров, добавление аннотаций и других элементов.

| Библиотека | Возможности кастомизации | Создание собственных стилей и цветовых схем |

|---|---|---|

| Matplotlib | Высокие | Требует понимания внутренней структуры графиков. |

| Seaborn | Средние | Предоставляет предопределенные стили и цветовые схемы, но ограниченные возможности для создания собственных. |

| Plotly | Низкие | Ограниченные возможности для настройки за пределами предоставленных тем и стилей. |

Matplotlib

Matplotlib имеет самые высокие возможности кастомизации среди трех библиотек. Пользователи могут полностью контролировать все аспекты графиков, включая размер холста, оси, метки, легенды и т. д. Однако для этого требуется понимание внутренней структуры графиков Matplotlib, что может быть сложным для начинающих пользователей.

Seaborn

Seaborn предоставляет набор предопределенных стилей и цветовых схем, которые пользователи могут применять к своим графикам. Это упрощает создание красиво оформленных визуализаций, не требуя глубоких знаний о настройке Matplotlib. Однако возможности Seaborn по созданию собственных стилей и цветовых схем более ограничены по сравнению с Matplotlib.

Plotly

Plotly имеет самые низкие возможности кастомизации среди трех библиотек. Он предоставляет несколько встроенных тем и стилей, но пользователи имеют ограниченные возможности для дальнейшей настройки. Plotly больше ориентирован на создание интерактивных и динамических графиков, а не на детальную настройку внешнего вида.

Пример кода:

Ниже приведен пример кода, показывающий, как настроить цвет фона графика с помощью каждой библиотеки:

# Matplotlib

import matplotlib.pyplot as plt

# Создайте фигуру и оси

fig, ax = plt.subplots()

# Установите цвет фона

ax.set\_facecolor('lightblue')

# Seaborn

import seaborn as sns

# Создайте график

sns.lineplot(data=df, x='x', y='y')

# Установите цвет фона

sns.set\_theme(style='whitegrid', palette='pastel', background='lavender')

# Plotly

import plotly.graph\_objects as go

# Создайте фигуру

fig = go.Figure()

# Добавьте цветовую дорожку для фона

fig.add\_trace(go.Scatter(x=[0, 1], y=[0, 1], mode='none', fill='toself', fillcolor='lightblue'))

3.2. Сравнение удобства использования

Удобство использования относится к тому, насколько легко пользователям создавать графики с помощью каждой библиотеки. Это включает в себя простоту синтаксиса, наличие документации и примеров, а также общую кривую обучения.

3.2.1. Простота синтаксиса и API

Matplotlib: Сложный API, требует понимания структуры графиков.

Seaborn: Более интуитивный API, разработанный для статистической визуализации.

Plotly: Доступный для начинающих API, но может быть сложным для создания расширенных визуализаций.

Matplotlib имеет самый сложный API из трех библиотек. Для создания графиков требуется понимание структуры графиков Matplotlib, включая фигуры, оси и графические элементы. Это может быть сложным для начинающих пользователей.

Seaborn имеет более интуитивный API, разработанный специально для статистической визуализации. Он предоставляет функции высокого уровня для создания распространенных типов графиков, таких как гистограммы, boxplots и scatterplots. Это делает Seaborn более доступным для пользователей, которые не знакомы со структурой графиков Matplotlib.

Plotly имеет дружелюбный для начинающих API, который упрощает создание интерактивных графиков. Однако для создания более сложных и настраиваемых визуализаций может потребоваться понимание структуры Plotly, которая может быть более сложной, чем у Matplotlib или Seaborn.

3.2.2. Документация и наличие примеров

Matplotlib: Обширная документация и многочисленные примеры.

Seaborn: Хорошо структурированная документация и полезные примеры.

Plotly: Хорошая документация, но примеры могут быть ограниченными.

Matplotlib имеет обширную документацию, которая охватывает все аспекты библиотеки. Она также предоставляет многочисленные примеры, которые показывают, как создавать различные типы графиков.

Seaborn имеет хорошо структурированную документацию, которая фокусируется на статистической визуализации. Она также предоставляет полезные примеры для распространенных задач визуализации.

Plotly имеет хорошую документацию, которая объясняет основные понятия и функции библиотеки. Однако количество примеров может быть ограничено, особенно для более сложных визуализаций.

3.2.3. Возможности кастомизации

Matplotlib: Наиболее гибкая библиотека с широкими возможностями кастомизации.

Seaborn: Более ограниченные возможности кастомизации, но предоставляет удобные функции для настройки стилей и цветовых схем.

Plotly: Наименее гибкая библиотека с ограниченными возможностями настройки за пределами предоставленных тем.

Matplotlib предоставляет наиболее гибкие возможности кастомизации. Пользователи могут полностью контролировать все аспекты графиков, включая размеры, цвета, шрифты и другие элементы. Однако это требует более глубокого понимания структуры Matplotlib.

Seaborn предлагает более ограниченные возможности кастомизации по сравнению с Matplotlib. Однако он предоставляет удобные функции для настройки стилей и цветовых схем, что упрощает создание красиво оформленных визуализаций.

Plotly имеет наименее гибкие возможности кастомизации. Он предоставляет несколько встроенных тем и стилей, но пользователи имеют ограниченные возможности для дальнейшей настройки. Plotly больше ориентирован на создание интерактивных и динамических графиков, а не на детальную настройку внешнего вида.

3.3. Анализ преимуществ и недостатков каждой библиотеки

Matplotlib:

Преимущества:

Широкие возможности кастомизации.

Поддержка создания сложных графиков.

Обширное сообщество и документация.

Недостатки:

Сложный API.

Устаревший внешний вид по умолчанию.

Меньше интерактивных возможностей.

Seaborn:

Преимущества:

Удобный API для статистической визуализации.

Предопределенные стили и цветовые схемы.

Хорошая документация и примеры.

Недостатки:

Более ограниченные возможности кастомизации.

Меньше возможностей для создания интерактивных графиков.

Некоторые функции могут быть медленными для больших наборов данных.

Plotly:

Преимущества:

Интерактивные и динамические графики.

Удобный API для начинающих.

Поддержка публикации графиков в интернете.

Недостатки:

Меньшая гибкость в настройке.

Может быть сложным для создания статических графиков.

Может потребоваться подключение к интернету для некоторых функций.

**Заключение**

**Основные выводы**

Сравнив Matplotlib, Seaborn и Plotly, мы пришли к следующим выводам:

Функциональность: Matplotlib обеспечивает надежные базовые графики, Seaborn специализируется на статистических визуализациях, а Plotly превосходит в интерактивных и динамических визуализациях.

Удобство использования: Seaborn имеет наиболее простой синтаксис, обширную документацию и множество примеров. Matplotlib и Plotly предлагают больше возможностей настройки и вариантов кастомизации.

Преимущества и недостатки: Matplotlib обеспечивает широкую функциональность, но может быть сложен в использовании. Seaborn удобен для использования, но ограничивает варианты настройки. Plotly предлагает интерактивные визуализации, но требует дополнительных знаний JavaScript.

**Рекомендации по выбору библиотеки**

Выбор библиотеки зависит от конкретных задач:

Базовые визуализации: Matplotlib

Статистические визуализации: Seaborn

Интерактивные и динамические визуализации: Plotly

Кастомизация: Matplotlib или Plotly

Начинающие: Seaborn

Опытные: Matplotlib или Plotly

**Перспективы дальнейших исследований**

Наше исследование можно продолжить в следующих направлениях:

Сравнение библиотек для конкретных наборов данных и задач.

Изучение новых визуализаций и функций, которые не рассматривались в этом исследовании.

Создание подробных руководств по использованию каждой библиотеки для различных задач.