**1. Введение**

**1.1 Обоснование выбора темы**

Визуализация данных является важным инструментом в аналитике, позволяющим исследователям и специалистам в различных областях эффективно представлять и интерпретировать информацию. Существует множество библиотек для визуализации данных на языке программирования Python, среди которых наиболее популярными являются Matplotlib, Seaborn и Plotly. Цель данной дипломной работы заключается в сравнительном анализе этих трех библиотек с точки зрения их функциональности, удобства использования и возможностей визуализации данных.

**Задачи работы:**

1. Провести обзор каждой библиотеки, описав ее основные функции и возможности.
2. Выбрать подходящий датасет для визуализации.
3. Разработать набор визуализаций, используя Matplotlib, Seaborn и Plotly.
4. Сравнить полученные результаты по различным критериям: функциональность, удобство использования, интерактивность и производительность.
5. Сделать выводы о том, какая библиотека лучше подходит для различных типов задач визуализации данных.

**1.2 Актуальность темы**

С увеличением объемов данных, с которыми работают компании и исследователи, возрастает необходимость в эффективных методах их анализа и представления. Визуализация данных помогает выявить скрытые закономерности, тренды и аномалии, что делает ее неотъемлемой частью процесса принятия решений. Выбор подходящей библиотеки для визуализации имеет ключевое значение для успешного анализа данных. Matplotlib, Seaborn и Plotly предлагают разные подходы к визуализации: от простых статических графиков до сложных интерактивных визуализаций. Понимание сильных и слабых сторон каждой из этих библиотек позволит пользователям более эффективно использовать их в своих проектах. Таким образом, исследование функциональности и удобства использования Matplotlib, Seaborn и Plotly не только актуально с точки зрения теории визуализации данных, но и практично для специалистов, работающих с данными в различных областях.

**2. Обзор литературы**

**2.1 Общие сведения о библиотеках**

**Matplotlib**  
Matplotlib — это одна из самых популярных библиотек для визуализации данных в Python, разработанная Джоном Д. Хунтером в 2003 году. Она предоставляет широкий спектр инструментов для создания статических, анимационных и интерактивных графиков.**Основные характеристики Matplotlib:**

* **Гибкость:** Позволяет создавать практически любые виды графиков, включая линейные графики, гистограммы, диаграммы рассеяния и трехмерные графики.
* **Настраиваемость:** Пользователи могут изменять практически все аспекты графиков, включая цвета, шрифты, размеры и стили линий.
* **Интеграция:** Легко интегрируется с другими библиотеками Python, такими как NumPy и Pandas.

Однако Matplotlib может быть сложной в использовании для новичков из-за своей низкоуровневой природы и необходимости ручной настройки многих параметров.**Seaborn**  
Seaborn — это библиотека, построенная на основе Matplotlib, которая упрощает создание красивых и информативных статистических графиков. Разработанная Виллом Д. Эвансом, Seaborn предлагает более высокоуровневый интерфейс для создания визуализаций.**Основные характеристики Seaborn:**

* **Статистическая визуализация:** Предоставляет функции для создания сложных статистических графиков с минимальными усилиями.
* **Эстетика:** По умолчанию использует более привлекательные цветовые палитры и стили оформления по сравнению с Matplotlib.
* **Интеграция с Pandas:** Легко работает с данными в формате DataFrame из библиотеки Pandas.

Seaborn отлично подходит для быстрого создания визуализаций, но может быть менее гибким по сравнению с Matplotlib в плане настройки.**Plotly**  
Plotly — это библиотека для создания интерактивных графиков и визуализаций данных. Она поддерживает как статические, так и динамические графики и позволяет пользователям взаимодействовать с визуализациями через веб-браузер.**Основные характеристики Plotly:**

* **Интерактивность:** Позволяет создавать интерактивные графики с возможностью масштабирования, наведения курсора и других действий.
* **Поддержка различных форматов:** Может экспортировать графики в различные форматы, такие как HTML и PNG.
* **Совместимость с Dash:** Позволяет создавать веб-приложения для визуализации данных с использованием Dash — фреймворка на основе Flask.

Plotly может быть более сложным в установке и использовании по сравнению с Matplotlib и Seaborn, но его возможности интерактивности делают его мощным инструментом для анализа данных.

**2.2 Сравнительный анализ**

Сравнение Matplotlib, Seaborn и Plotly можно провести по нескольким критериям:

* **Функциональность:** Все три библиотеки предлагают широкий спектр возможностей для визуализации данных. Matplotlib является наиболее универсальным инструментом; Seaborn ориентирован на статистическую визуализацию; Plotly выделяется интерактивностью.
* **Удобство использования:** Seaborn часто считается более удобным для новичков благодаря высокоуровневым функциям и простоте создания красивых графиков. Matplotlib требует больше времени на изучение из-за своей гибкости и низкоуровневого подхода; Plotly требует определенных знаний о веб-технологиях для эффективного использования интерактивных функций.
* **Эстетика:** Seaborn предлагает более привлекательные по умолчанию стили оформления по сравнению с Matplotlib; Plotly также обеспечивает высококачественные визуализации с возможностью настройки внешнего вида.
* **Интерактивность:** Plotly предоставляет наибольшую интерактивность среди трех библиотек, что делает его идеальным выбором для веб-приложений и презентаций данных.

Таким образом, каждая из библиотек имеет свои сильные и слабые стороны, что делает их подходящими для различных задач в области визуализации данных. В следующем разделе будет рассмотрен выбор данных для визуализации и типы графиков, которые будут использоваться в рамках исследования.

**3. Методология**

**3.2 Выбор данных для визуализации:**

Для анализа и визуализации данных в рамках данной работы можно выбрать следующие источники:

* **Gapminder:** Датасет, содержащий информацию о различных показателях развития стран (например, ВВП на душу населения).
* **Formula 1 Race Data:** Данные о гонках Формулы 1 (результаты гонок).

**Типы визуализаций:**  
Определить типы графиков (гистограммы, диаграммы рассеяния).

**3.2 В рамках исследования будут созданы следующие типы графиков:**

* **Гистограммы:** Для отображения распределения числовых данных (например, ВВП на душу населения).
* **Диаграммы рассеяния:** Для изучения взаимосвязи между двумя переменными (например, между ВВП на душу населения и ожидаемой продолжительностью жизни).
* **Линейные графики:** Для отображения изменений во времени (например, динамики ожидаемой продолжительности жизни по годам).

**4. Реализация**

* 1. **Создание визуализаций с использованием Matplotlib**:

Рис.1 Создание визуализаций с использованием Matplotlib

Описание результатов выполнения кода:

При выполнении кода, который создает различные графики с использованием библиотеки Matplotlib для визуализации данных о ВВП на душу населения, вы получите четыре различных типа графиков. Каждый из них предоставляет уникальную информацию о состоянии экономики стран в указанные годы. Ниже приведено описание каждого графика и его результатов.

1. Линейный график

Название: ВВП на душу населения за несколько лет

Описание: Этот график отображает изменения ВВП на душу населения для трех выбранных годов (2000, 2001, 2002) по различным странам.

Тенденции: Линейные графики позволяют увидеть, как изменялся ВВП на душу населения в течение этих трех лет. Вы можете заметить, какие страны демонстрируют рост, а какие — снижение.

Читаемость: Поворот названий стран на 90 градусов улучшает читаемость, особенно для стран с длинными названиями.

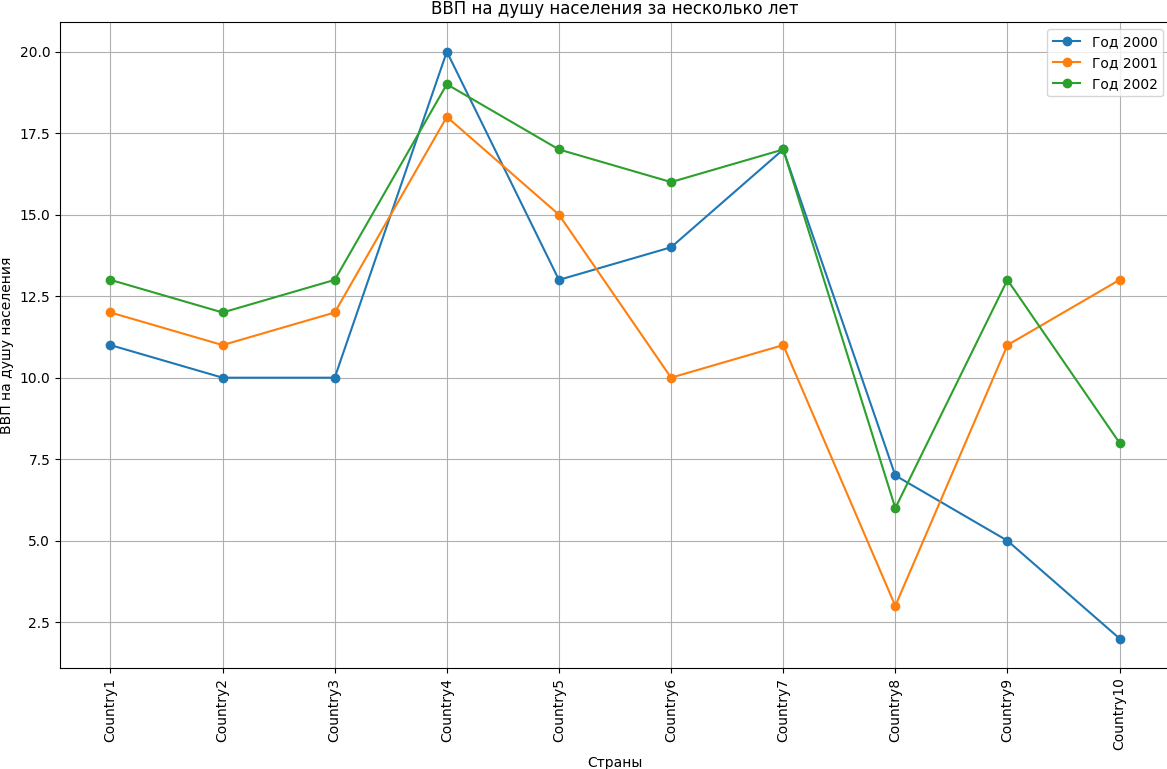


Рис.2 Линейный график с использованием Matplotlib

2. Гистограмма

Название: Гистограмма ВВП на душу населения в 2002 году

Описание: Этот график представляет распределение ВВП на душу населения по странам в 2002 году.

Сравнение: Гистограмма позволяет легко сравнить значения между странами. Вы можете быстро увидеть, какие страны имеют высокий или низкий ВВП на душу населения.

Визуализация: Столбцы отображают значения ВВП, а цветовая схема (например, светло-голубой) делает график более привлекательным.

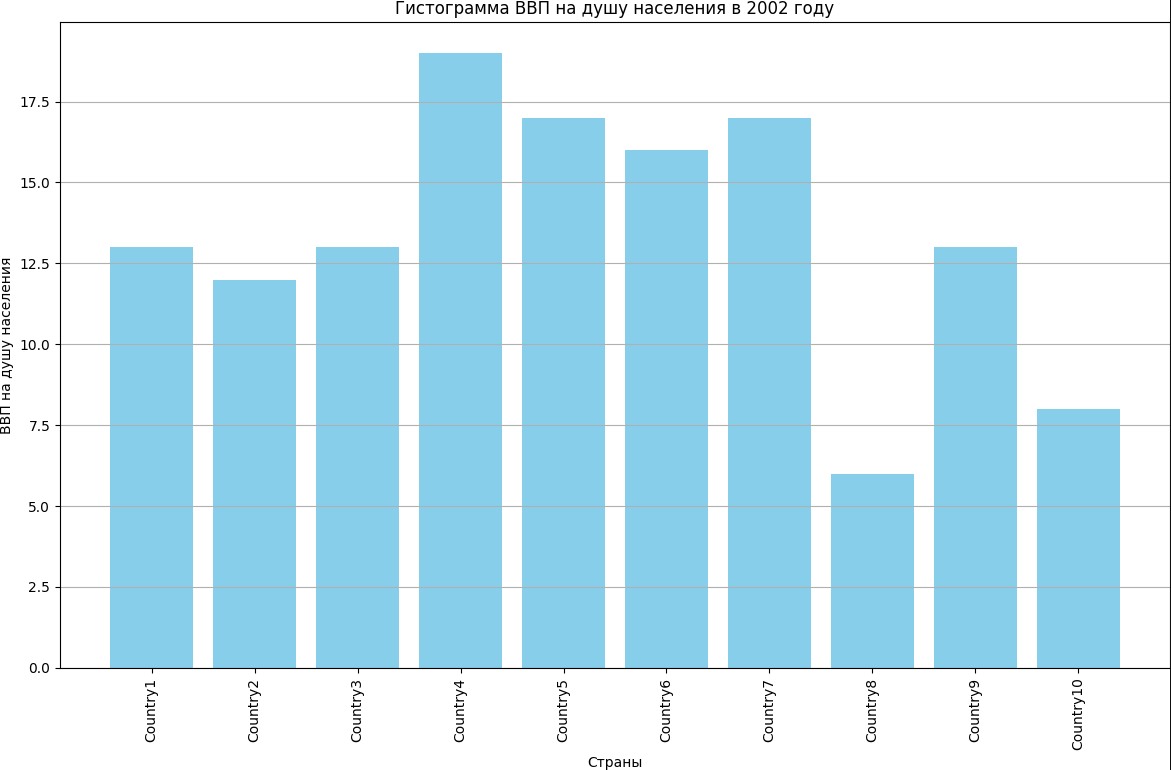


Рис.3 Гистограмма с использованием Matplotlib

3. Диаграмма рассеяния

Название: Диаграмма рассеяния: ВВП на душу населения (2000 vs 2002)

Описание: Этот график показывает взаимосвязь между ВВП на душу населения в 2000 и 2002 годах.

Анализ зависимости: Каждая точка представляет одну страну, и вы можете видеть, как изменился ВВП за два года. Если точки располагаются близко к диагонали, это указывает на стабильность значений.

Аннотации: Аннотации для каждой точки помогают идентифицировать страны и делают анализ более информативным.

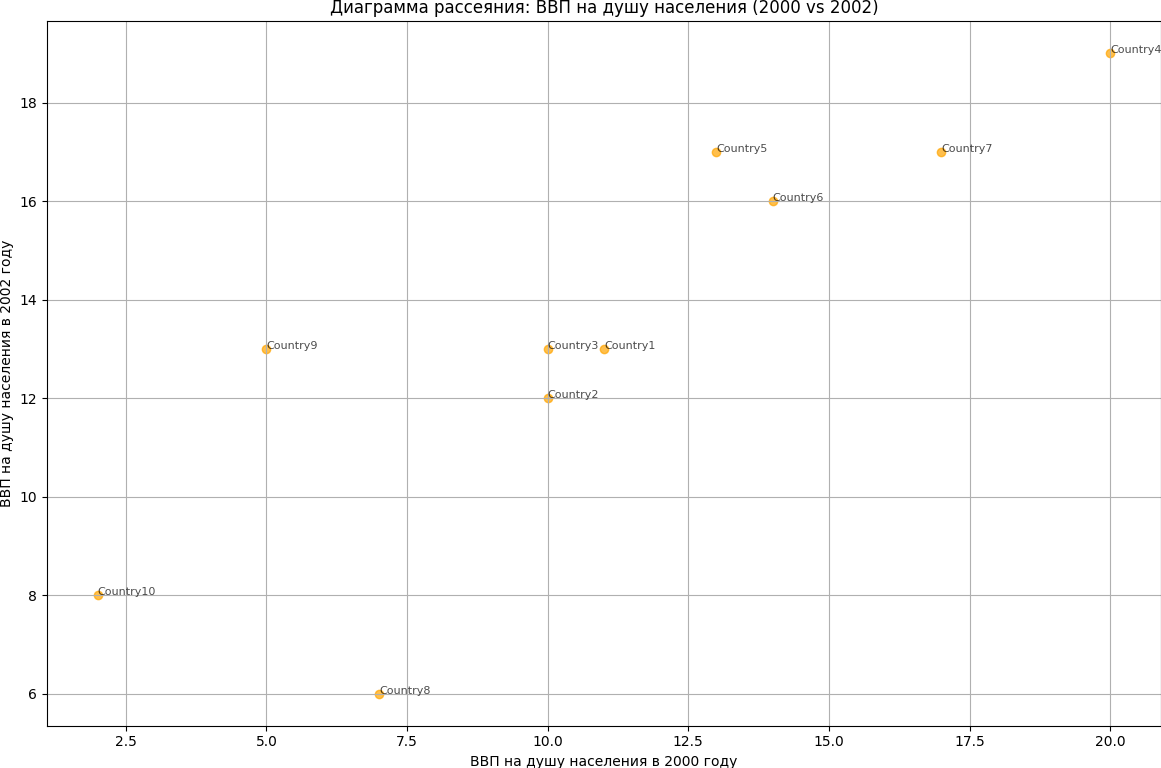


Рис.4 Диаграмма рассеяния с использованием Matplotlib

4. Круговая диаграмма

Название: Средний ВВП на душу населения по странам в 2002 году

Описание: Этот график визуализирует средний ВВП на душу населения по странам в 2002 году.

Доля стран: Круговая диаграмма показывает процентное соотношение среднего ВВП для каждой страны относительно общего среднего. Это позволяет увидеть, какие страны занимают большую долю в общем ВВП.

Эстетика: Использование цветов и автоматического отображения процентов делает диаграмму более понятной и привлекательной.

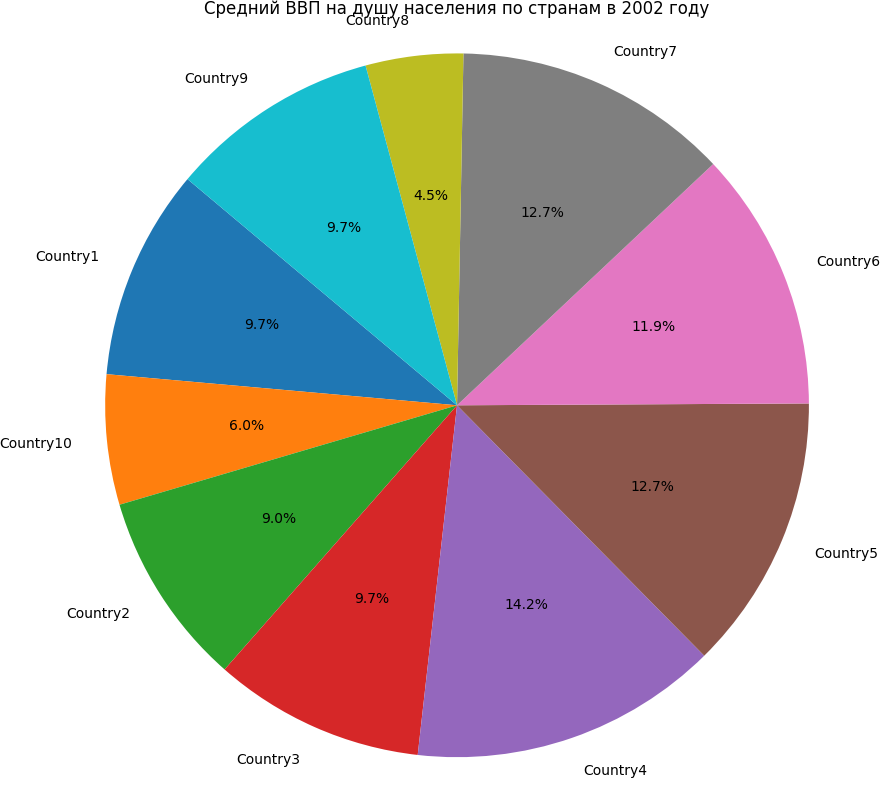


Рис.5 Круговая диаграмма с использованием Matplotlib

Заключение

Эти графики предоставляют разнообразные способы анализа данных о ВВП на душу населения в разных странах за указанные годы. Они помогают выявить тенденции и различия между странами, что может быть полезно для экономического анализа и принятия решений. Каждый тип графика имеет свои преимущества и может быть использован для различных целей анализа данных.

* 1. **Создание визуализаций с использованием Seaborn**:

Рис.6 Создание визуализаций с использованием Seaborn

Описание результатов выполнения кода:

1. Линейный график

Общий вид:

Линейный график показывает ожидаемую продолжительность жизни (life expectancy) по годам для трех стран: Соединенных Штатов, Китая и Индии. Каждая страна представлена своей линией, различающейся по цвету.

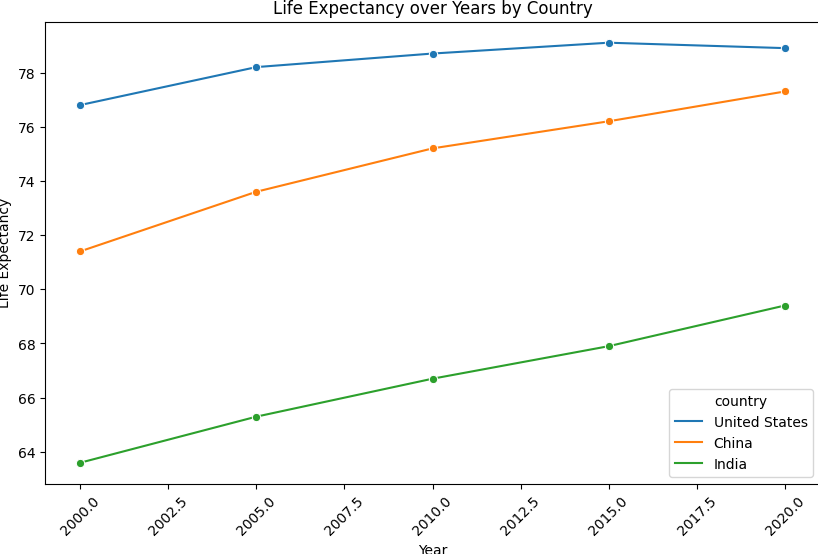


Рис.7 Линейный график с использованием Seaborn

Тенденции:

Соединенные Штаты: Ожидаемая продолжительность жизни постепенно увеличивается с 76.8 лет в 2000 году до 79.1 лет в 2015 году, затем немного снижается до 78.9 лет в 2020 году.

Китай: Ожидаемая продолжительность жизни также демонстрирует устойчивый рост — с 71.4 лет в 2000 году до 77.3 лет в 2020 году, что указывает на значительные улучшения в здравоохранении и условиях жизни.

Индия: Ожидаемая продолжительность жизни увеличивается с 63.6 лет в 2000 году до 69.4 лет в 2020 году, что также свидетельствует о положительных изменениях, хотя уровень остается ниже, чем у США и Китая.

Выводы:

Линейный график наглядно демонстрирует разницу в ожидаемой продолжительности жизни между тремя странами и показывает, что Китай и Индия значительно улучшили свои показатели за рассматриваемый период.

1. Столбчатая диаграмма

Общий вид:

Столбчатая диаграмма представляет те же данные, но в виде столбцов для каждого года и страны. Это позволяет лучше сравнивать значения между разными странами за один год.

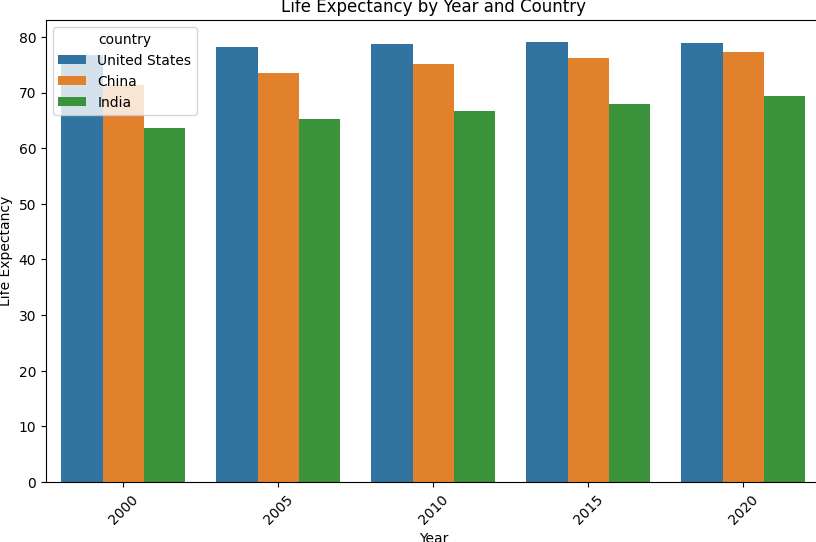


Рис.8 Столбчатая диаграмма с использованием Seaborn

Сравнение по годам:

Визуально видно, что Соединенные Штаты имеют наивысший уровень ожидаемой продолжительности жизни среди трех стран, хотя их темпы роста замедлились после 2015 года.

Китай показывает наиболее резкий рост, особенно заметный в последние два десятилетия.

Индия, хотя и демонстрирует улучшение, все еще отстает от двух других стран по ожидаемой продолжительности жизни.

Выводы:

Столбчатая диаграмма позволяет легко увидеть изменения по годам и сравнить их между странами. Она подчеркивает значительные достижения Китая в области здоровья населения за последние два десятилетия.

Заключение

Оба графика эффективно иллюстрируют данные о продолжительности жизни в разных странах и показывают важные тенденции и различия. Линейный график акцентирует внимание на динамике изменений во времени, а столбчатая диаграмма помогает быстро оценить значения для конкретных годов. Эти визуализации могут быть полезны для анализа политики здравоохранения и социальных изменений в этих странах.

* 1. **Создание визуализаций с использованием Plotly**:

Рис.9 Создание визуализаций с использованием Plotly

Описание результатов выполнения кода:

При выполнении кода, который создает и отображает графики с использованием библиотеки Plotly, будут получены четыре различных визуализации, каждая из которых предоставляет уникальную информацию о ожидаемой продолжительности жизни в трех странах: Соединенных Штатах, Китае и Индии. Ниже приведено описание каждого графика:

1. Линейный график

Название: Изменение ожидаемой продолжительности жизни по странам

Описание: Этот график отображает изменения ожидаемой продолжительности жизни в Соединенных Штатах, Китае и Индии с 2000 по 2020 год. Каждая линия представляет одну из стран, и вы можете наблюдать за трендами:

Соединенные Штаты: Линейный график показывает небольшое увеличение ожидаемой продолжительности жизни с 76.8 лет в 2000 году до 79.1 лет в 2015 году, но с небольшим снижением до 78.9 лет в 2020 году.

Китай: Ожидаемая продолжительность жизни значительно увеличилась с 71.4 лет в 2000 году до 77.3 лет в 2020 году, что указывает на улучшение здоровья населения.

Индия: Ожидаемая продолжительность жизни также увеличилась, но менее резко, от 63.6 лет в 2000 году до 69.4 лет в 2020 году.

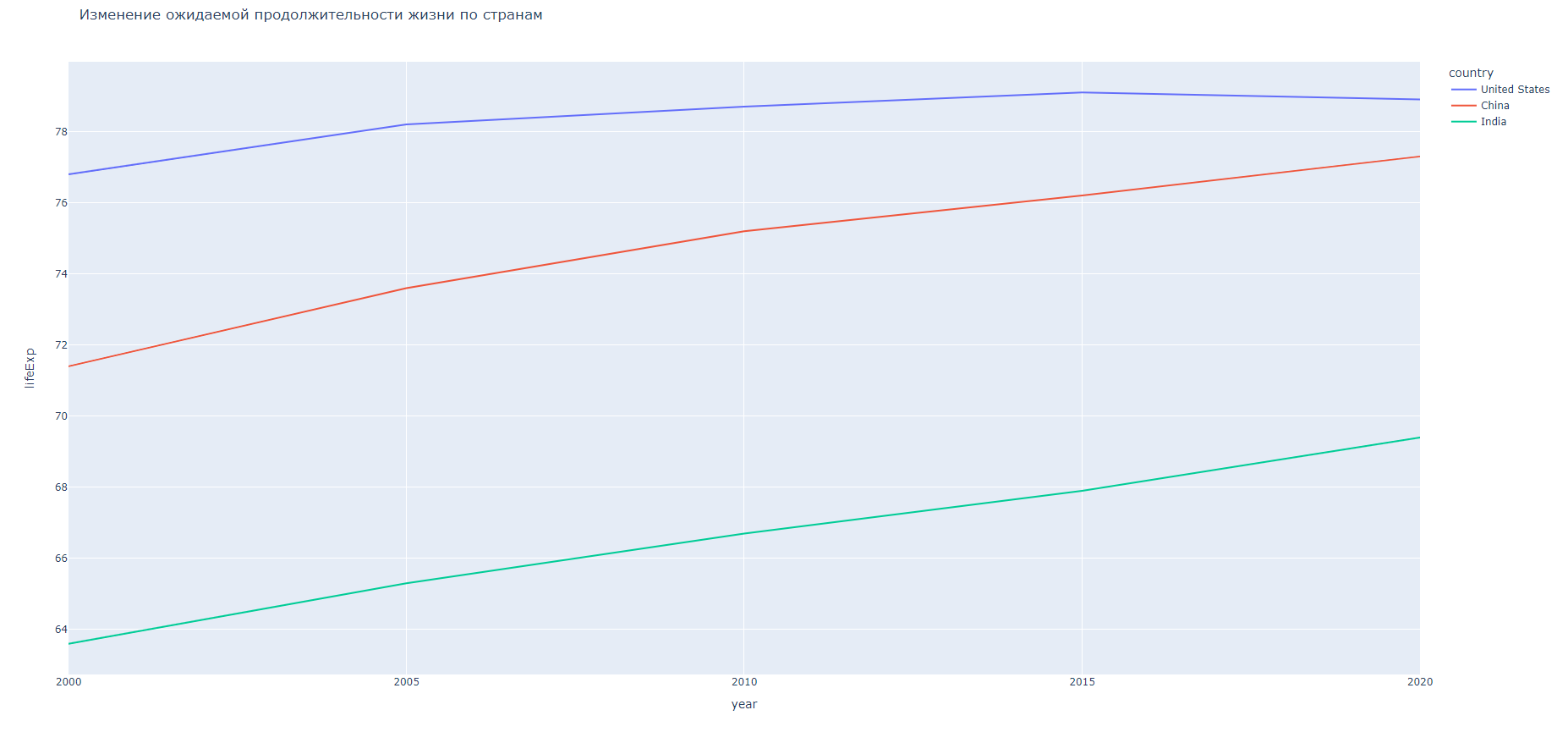


Рис.10 Линейный график с использованием Plotly

2. Гистограмма

Название: Ожидаемая продолжительность жизни по странам

Описание: Этот график показывает среднюю ожидаемую продолжительность жизни для каждой страны на момент последнего года (2020). Гистограмма позволяет легко сравнить значения:

Соединенные Штаты имеют наивысшую ожидаемую продолжительность жизни (78.9 лет).

Китай следует за ними с ожидаемой продолжительностью жизни (77.3 лет).

Индия имеет наименьшее значение (69.4 лет), что подчеркивает различия в уровне здоровья и доступе к медицинским услугам между этими странами.

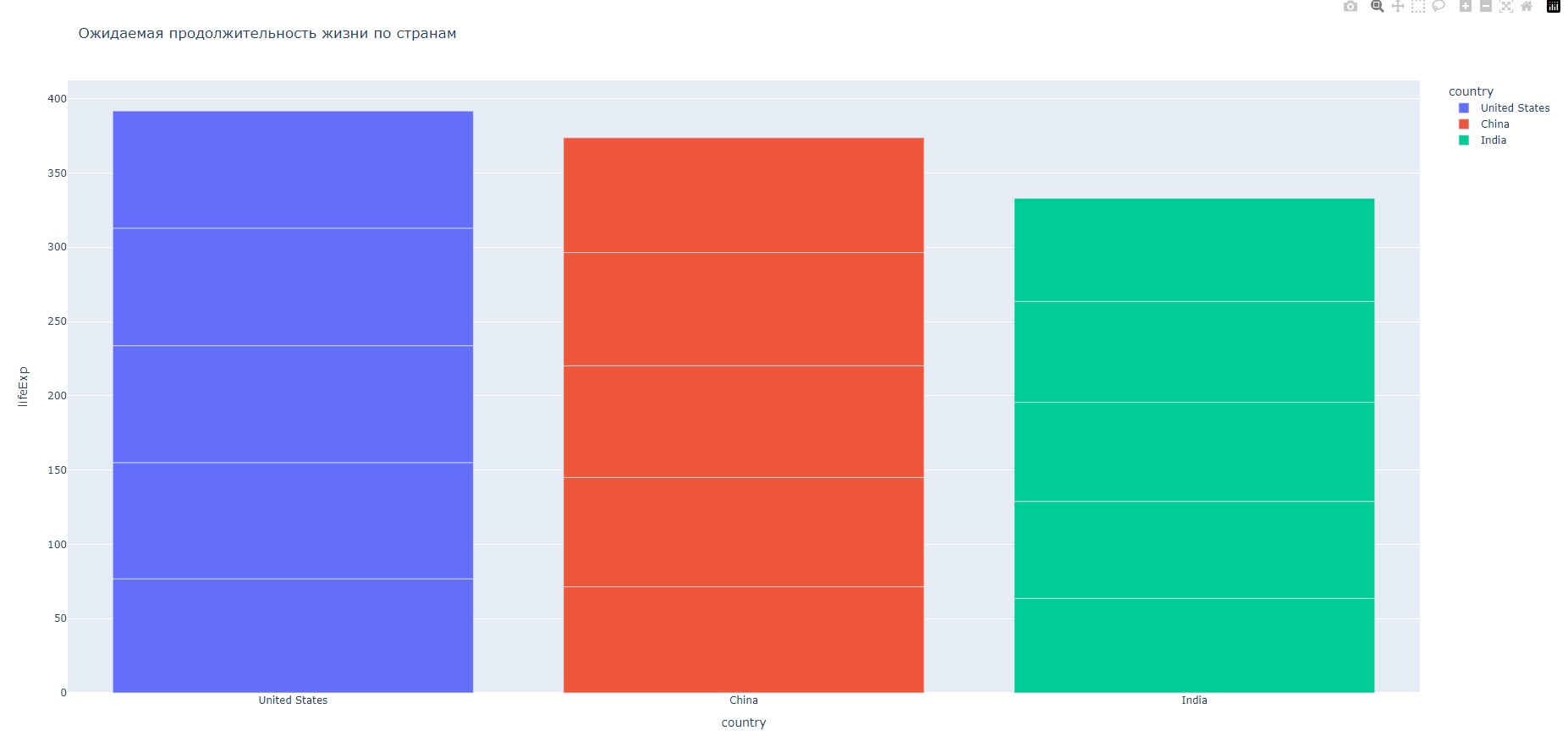


Рис.11 Гистограмма с использованием Plotly

3. Диаграмма рассеяния

Название: Диаграмма рассеяния: Год vs Ожидаемая продолжительность жизни

Описание: Этот график отображает взаимосвязь между годом и ожидаемой продолжительностью жизни для каждой страны. Каждая точка на графике представляет данные за определенный год:

Вы можете увидеть, как со временем увеличивалась ожидаемая продолжительность жизни для всех трех стран.

Диаграмма позволяет выявить тренды и аномалии, например, замедление роста ожидаемой продолжительности жизни в Соединенных Штатах после 2015 года.

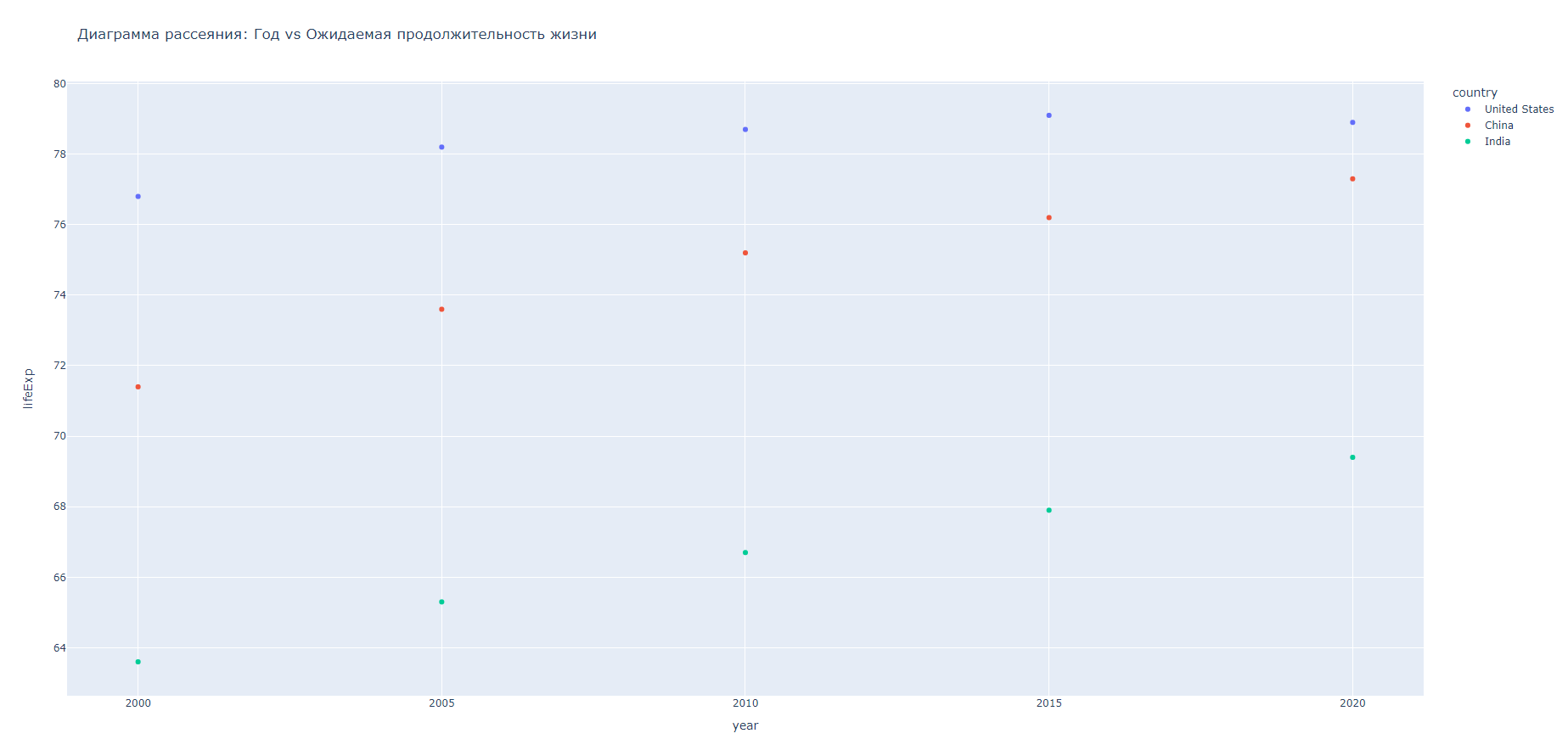


Рис.12 Диаграмма рассеяния с использованием Plotly

4. Круговая диаграмма

Название: Средняя ожидаемая продолжительность жизни по странам

Описание: Этот график визуализирует долю средней ожидаемой продолжительности жизни для каждой страны на основе средних значений:

Круговая диаграмма показывает процентное соотношение между тремя странами.

Соединенные Штаты занимают наибольшую долю, за ними следует Китай, а Индия занимает меньшую долю.

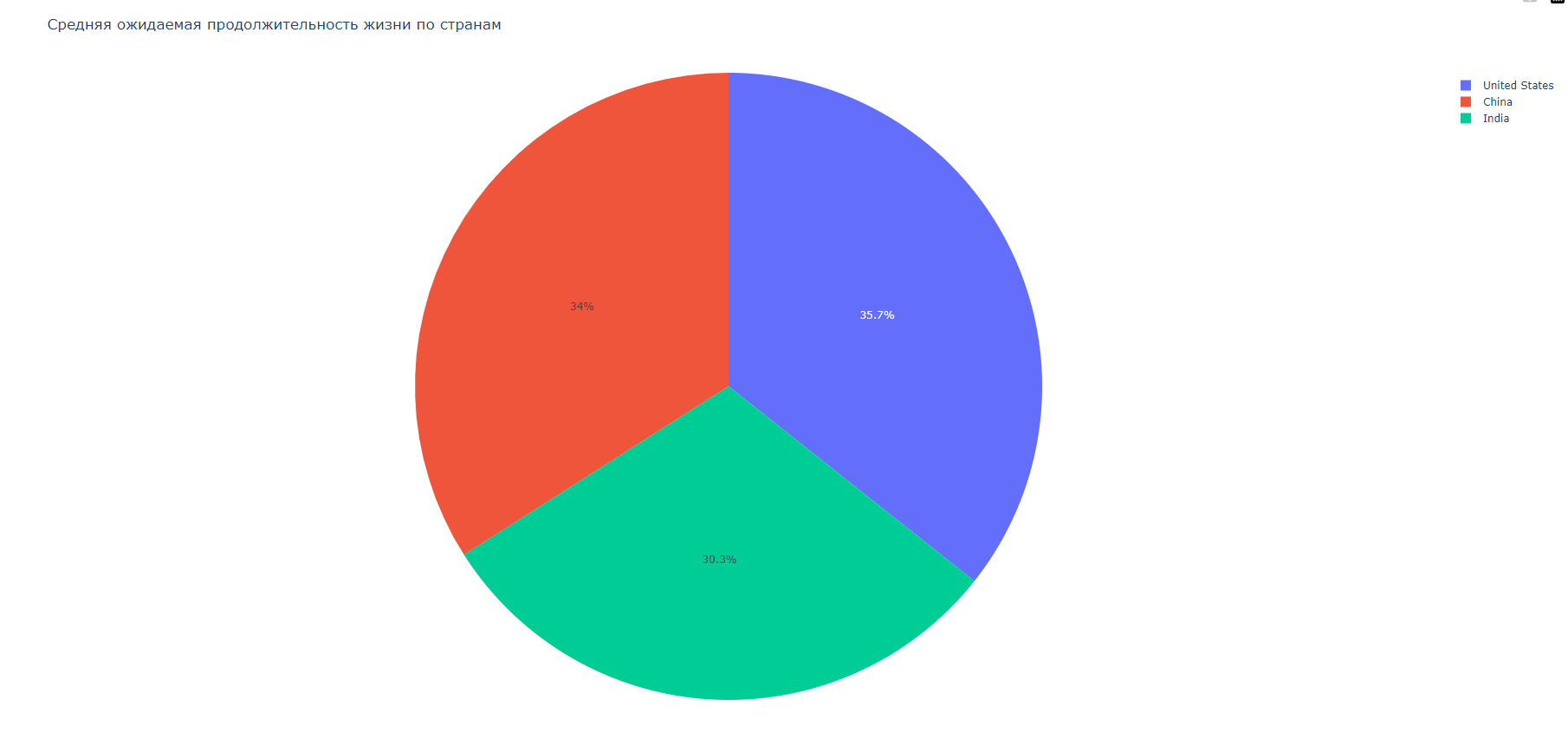


Рис.13 Круговая диаграмма с использованием Plotly

Заключение

Эти графики предоставляют наглядное представление о том, как изменялась ожидаемая продолжительность жизни в различных странах за последние два десятилетия. Они могут быть полезны для анализа тенденций в области здравоохранения и для принятия решений на основе данных о здоровье населения.

**5. Сравнительный анализ**

В данном разделе будет проведен сравнительный анализ библиотек Matplotlib, Seaborn и Plotly по нескольким критериям: функциональность, удобство использования, интерактивность и производительность. Это позволит выявить сильные и слабые стороны каждой библиотеки и определить, какая из них наиболее подходит для различных задач визуализации данных.

Функциональность

* **Matplotlib:**
  + **Плюсы:** Обеспечивает широкий спектр возможностей для создания различных типов графиков, включая статические, анимационные и интерактивные. Позволяет детально настраивать графики, что делает её универсальным инструментом для визуализации.
  + **Минусы:** Для создания сложных визуализаций может потребоваться значительное количество кода и времени на настройку.
* **Seaborn:**
  + **Плюсы:** Построен на основе Matplotlib и предлагает более высокоуровневые функции для создания статистических графиков. Упрощает создание красивых визуализаций с минимальными усилиями.
  + **Минусы:** Менее гибок в плане настройки по сравнению с Matplotlib. Некоторые специфические графики могут быть труднее реализовать.
* **Plotly:**
  + **Плюсы:** Специализируется на создании интерактивных визуализаций. Поддерживает создание сложных графиков с возможностью взаимодействия (масштабирование, наведение курсора и т.д.).
  + **Минусы:** Может быть сложнее в использовании для новичков из-за необходимости понимания веб-технологий.

Удобство использования

* **Matplotlib:**
  + Требует больше времени на изучение из-за своей низкоуровневой природы. Новичкам может быть сложно начать работу из-за большого количества параметров настройки.
* **Seaborn:**
  + Считается более удобным для новичков благодаря высокоуровневым функциям. Позволяет быстро создавать графики с помощью простого синтаксиса.
* **Plotly:**
  + Имеет интуитивно понятный интерфейс для создания интерактивных графиков, однако требует знаний о веб-технологиях для полного использования всех возможностей библиотеки.

Интерактивность

* **Matplotlib:**
  + Основные возможности интерактивности ограничены. Подходит в основном для статических графиков, хотя есть возможность создания интерактивных графиков с использованием дополнительных библиотек (например, mpld3).
* **Seaborn:**
  + Не предоставляет встроенной интерактивности. Графики создаются статическими, однако их можно экспортировать в Matplotlib для добавления интерактивных функций.
* **Plotly:**
  + Обеспечивает высокий уровень интерактивности. Пользователи могут взаимодействовать с графиками через веб-браузер, что делает его идеальным выбором для презентаций и веб-приложений.

Производительность

* **Matplotlib:**
  + Обычно показывает хорошую производительность при создании сложных визуализаций на больших наборах данных. Время выполнения может увеличиваться при добавлении большого количества элементов на график.
* **Seaborn:**
  + Производительность аналогична Matplotlib, но может быть немного ниже при создании очень сложных графиков из-за дополнительного уровня абстракции.
* **Plotly:**
  + Может замедляться при работе с большими объемами данных из-за своей интерактивности. Время загрузки может увеличиваться при создании сложных визуализаций с множеством точек данных.

Выводы

Каждая из библиотек имеет свои сильные и слабые стороны:

* **Matplotlib** является наиболее универсальным инструментом, подходящим для создания статических и анимационных графиков с высокой степенью настройки.
* **Seaborn** идеально подходит для быстрого создания статистических графиков с привлекательным дизайном и простым синтаксисом.
* **Plotly** выделяется своей интерактивностью и возможностями веб-визуализации, что делает его отличным выбором для динамических приложений и презентаций данных.

**6. Заключение**

Выводы по результатам исследования

В ходе исследования были проанализированы три популярные библиотеки для визуализации данных в Python: Matplotlib, Seaborn и Plotly. Каждая из этих библиотек имеет свои уникальные особенности, которые делают их подходящими для различных задач визуализации.

1. **Matplotlib**:
   * Обеспечивает максимальную гибкость и контроль над графиками. Это идеальный инструмент для пользователей, которым необходимо создавать сложные и детализированные визуализации. Однако, его низкоуровневая природа может затруднить процесс создания графиков для новичков.
2. **Seaborn**:
   * Предлагает более высокоуровневый интерфейс, что делает его удобным для быстрого создания статистических графиков с эстетически привлекательным дизайном. Он отлично подходит для исследователей и аналитиков, которые хотят быстро визуализировать данные без необходимости глубоко погружаться в детали настройки.
3. **Plotly**:
   * Выделяется своей интерактивностью и возможностями веб-визуализации. Это делает его идеальным выбором для приложений, где требуется взаимодействие с пользователем, таких как дашборды и веб-презентации. Тем не менее, он может быть более сложным в установке и использовании по сравнению с другими библиотеками.

Рекомендации

При выборе библиотеки для визуализации данных важно учитывать конкретные требования проекта:

* **Для статических графиков с высокой степенью настройки**: рекомендуется использовать Matplotlib, особенно если необходимо создать сложные визуализации с индивидуальными настройками.
* **Для быстрого создания красивых статистических графиков**: Seaborn будет лучшим выбором благодаря своей простоте и удобству использования.
* **Для интерактивных визуализаций и веб-приложений**: Plotly предоставляет мощные инструменты для создания динамических графиков и дашбордов, что делает его идеальным для проектов, требующих взаимодействия с пользователем.

Таким образом, понимание сильных и слабых сторон каждой из библиотек позволяет пользователям более эффективно использовать их в своих проектах по визуализации данных. Важно выбирать библиотеку в зависимости от специфики задачи, чтобы добиться наилучших результатов в анализе и представлении данных.

**7. Приложения**

* **Кодовые примеры**: Полные примеры кода для каждой библиотеки.

**Визуализация с использованием Matplotlib**:

import matplotlib.pyplot as plt

import pandas as pd

# Загрузка данных

df = pd.read\_csv('gapminder.csv')

# Проверка наличия нужных столбцов

print("Столбцы в DataFrame:", df.columns)

# Проверка наличия данных за 2000 и 2002 годы

if '2000' not in df.columns or '2002' not in df.columns:

print("Ошибка: Не найдены данные для годов 2000 и 2002.")

else:

print("Данные за 2000 и 2002 годы найдены.")

# Выбор годов для отображения

years = ['2000', '2001', '2002']

# Линейный график ВВП на душу населения за несколько лет

plt.figure(figsize=(12, 8))

for year in years:

plt.plot(df['Country Name'], df[year], marker='o', label=f'Год {year}')

plt.title('ВВП на душу населения за несколько лет')

plt.xlabel('Страны')

plt.ylabel('ВВП на душу населения')

plt.xticks(rotation=90) # Поворачиваем названия стран для лучшей читаемости

plt.legend()

plt.grid()

plt.tight\_layout() # Улучшает размещение элементов

plt.show()

# Гистограмма ВВП на душу населения в 2002 году

plt.figure(figsize=(12, 8))

plt.bar(df['Country Name'], df['2002'], color='skyblue')

plt.title('Гистограмма ВВП на душу населения в 2002 году')

plt.xlabel('Страны')

plt.ylabel('ВВП на душу населения')

plt.xticks(rotation=90)

plt.grid(axis='y')

plt.tight\_layout()

plt.show()

# Диаграмма рассеяния: ВВП на душу населения в 2000 и 2002 годах

if '2000' in df.columns and '2002' in df.columns:

plt.figure(figsize=(12, 8))

plt.scatter(df['2000'], df['2002'], color='orange', alpha=0.7)

plt.title('Диаграмма рассеяния: ВВП на душу населения (2000 vs 2002)')

plt.xlabel('ВВП на душу населения в 2000 году')

plt.ylabel('ВВП на душу населения в 2002 году')

# Добавление аннотаций для каждой точки

for i in range(len(df)):

plt.annotate(df['Country Name'][i], (df['2000'][i], df['2002'][i]), fontsize=8, alpha=0.7)

plt.grid()

plt.tight\_layout()

plt.show()

else:

print("Ошибка: Не найдены данные для годов 2000 и 2002.")

# Круговая диаграмма: средний ВВП на душу населения по странам в 2002 году

average\_gdp = df.groupby('Country Name')['2002'].mean()

plt.figure(figsize=(10, 10))

plt.pie(average\_gdp, labels=average\_gdp.index, autopct='%1.1f%%', startangle=140)

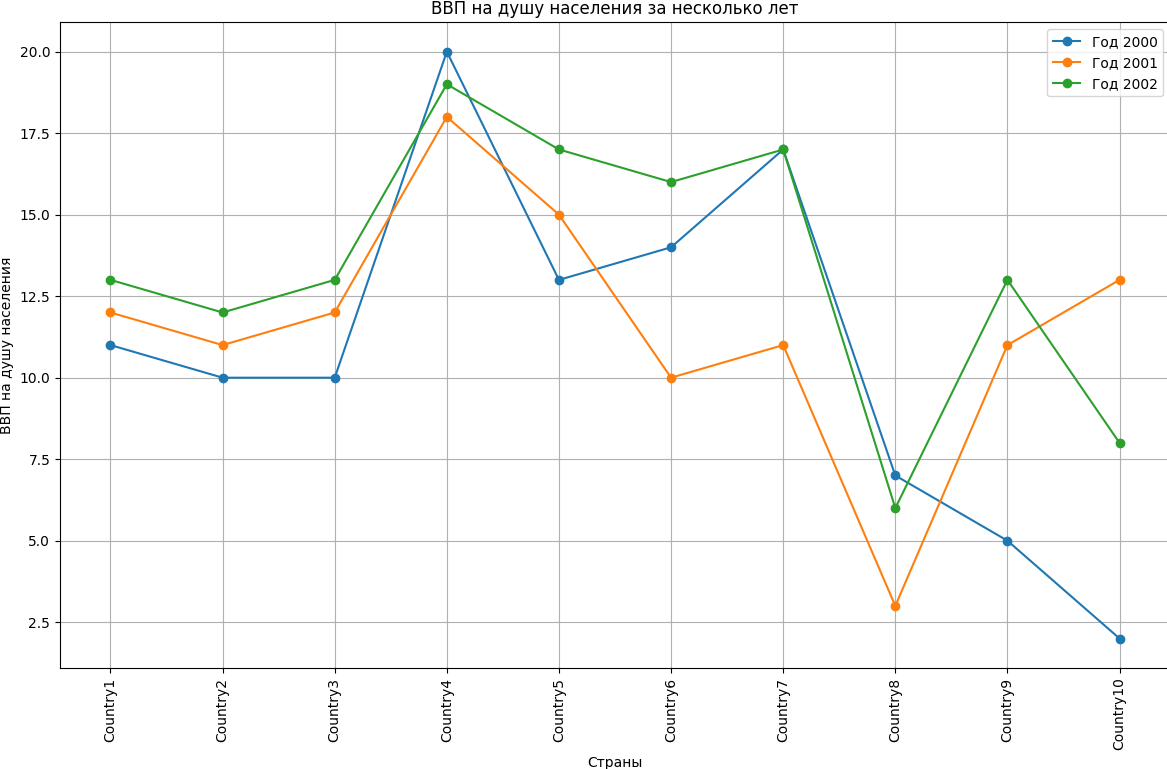
plt.title('Средний ВВП на душу населения по странам в 2002 году')

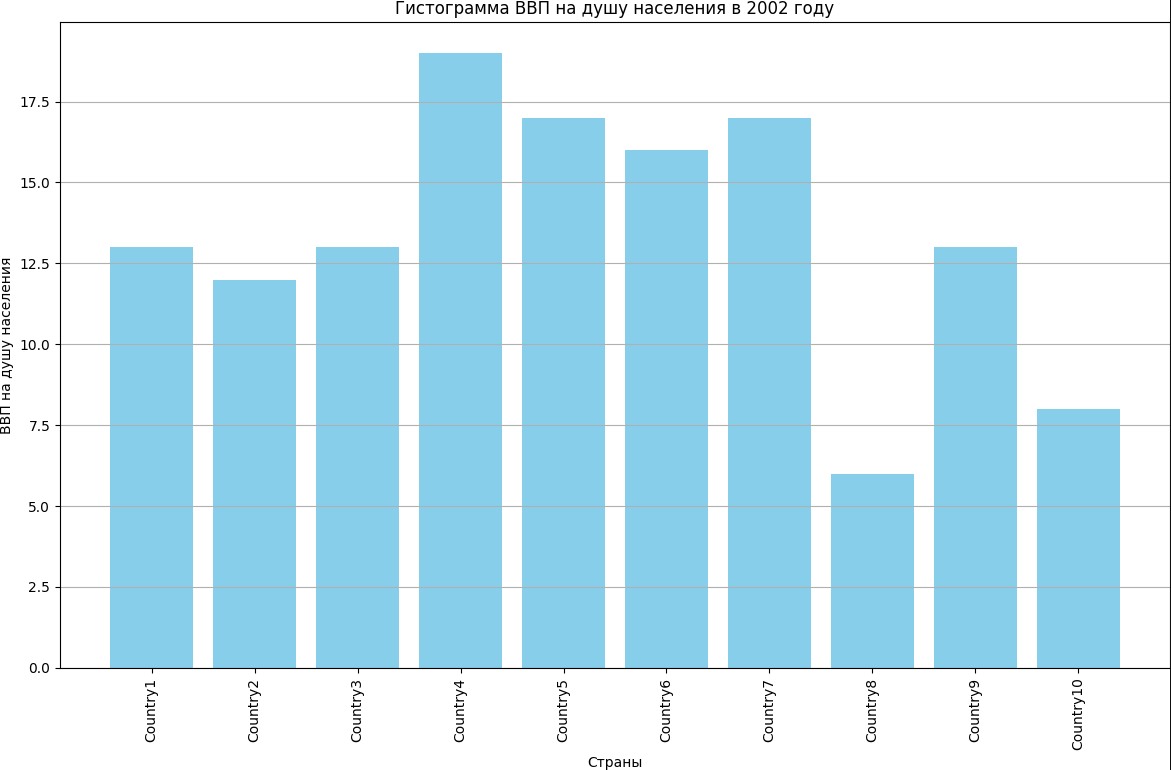
plt.axis('equal') # Чтобы круговая диаграмма была круглой

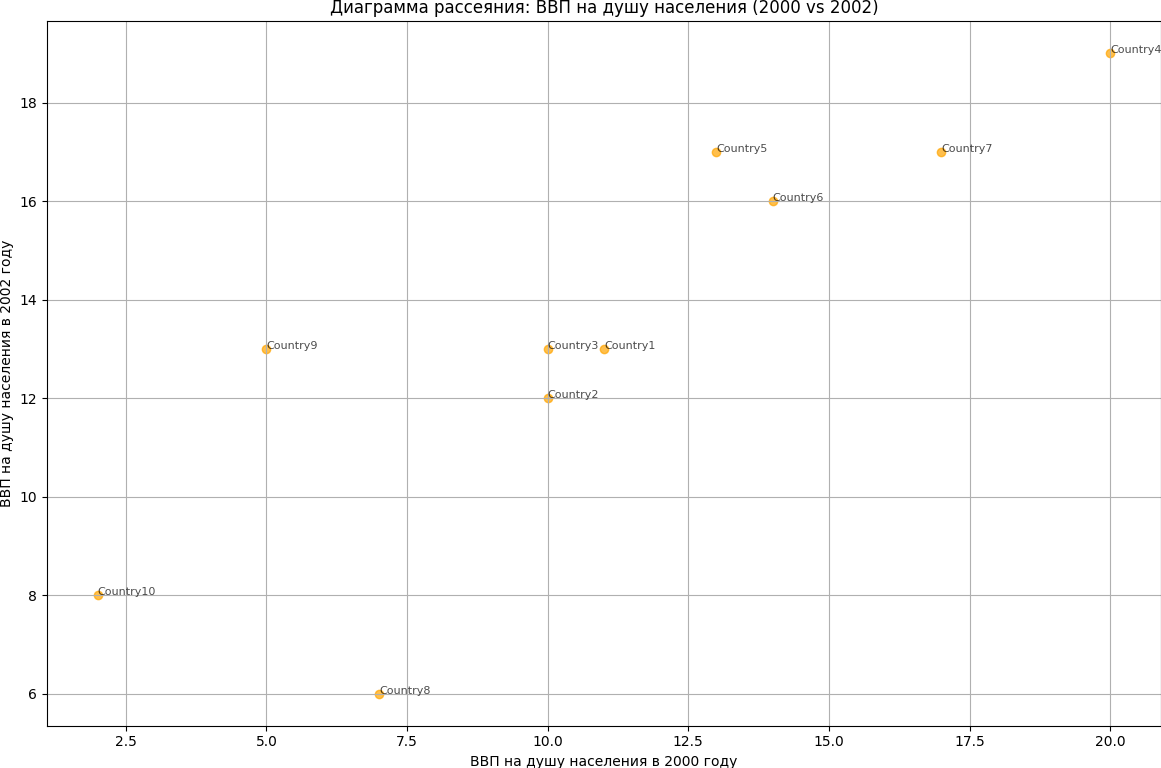
plt.show()

DataFrame:

Country Name,Code,2000,2001,2002,2003,2004,2005  
Country1,ABC,11,12,13,4,5,6  
Country2,GFI,10,11,12,13,14,15  
Country3,GFL,10,12,13,13,14,15  
Country4,GAI,20,18,19,13,14,15  
Country5,GJI,13,15,17,13,14,15  
Country6,RHI,14,10,16,13,14,15  
Country7,THI,17,11,17,13,14,15  
Country8,YHI,7,3,6,13,14,15  
Country9,UHI 9,5,11,13,14,15  
Country10,IHI,2,13,8,13,14,15







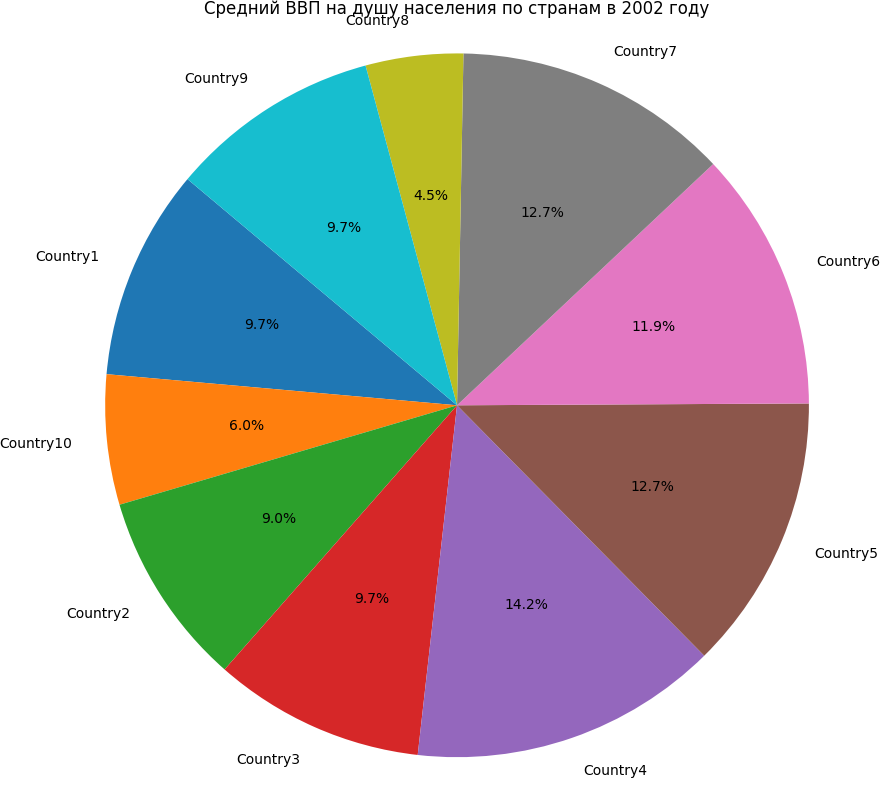


Рис.14,15,16,17 Результат выполнения с использованием Matplotlib.

**Визуализация с использованием Seaborn**:

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

# Создание данных для DataFrame

data = {

'country': ['United States', 'United States', 'United States', 'United States', 'United States',

'China', 'China', 'China', 'China', 'China',

'India', 'India', 'India', 'India', 'India'],

'year': [2000, 2005, 2010, 2015, 2020,

2000, 2005, 2010, 2015, 2020,

2000, 2005, 2010, 2015, 2020],

'lifeExp': [76.8, 78.2, 78.7, 79.1, 78.9,

71.4, 73.6, 75.2, 76.2, 77.3,

63.6, 65.3, 66.7, 67.9, 69.4]

}

# Создание DataFrame

df = pd.DataFrame(data)

# Линейный график

plt.figure(figsize=(10, 6))

sns.lineplot(data=df, x='year', y='lifeExp', hue='country', marker='o')

plt.title('Life Expectancy over Years by Country')

plt.xlabel('Year')

plt.ylabel('Life Expectancy')

plt.xticks(rotation=45)

plt.show()

# Столбчатая диаграмма

plt.figure(figsize=(10, 6))

sns.barplot(data=df, x='year', y='lifeExp', hue='country', ci=None)

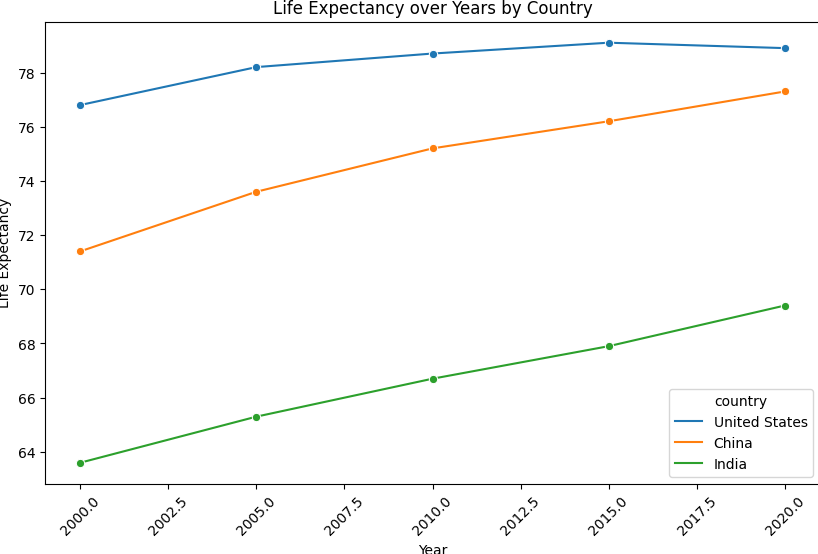
plt.title('Life Expectancy by Year and Country')

plt.xlabel('Year')

plt.ylabel('Life Expectancy')

plt.xticks(rotation=45)

plt.show()



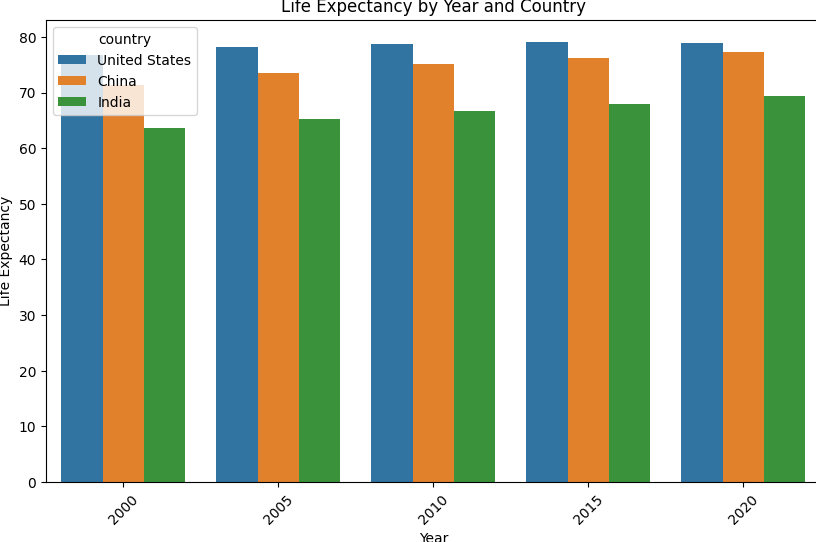


Рис.18,19 Результат выполнения с использованием Seaborn.

**Визуализация с использованием Plotly**:

import pandas as pd

import plotly.express as px

# Создание данных для DataFrame

data = {

'country': ['United States', 'United States', 'United States', 'United States', 'United States',

'China', 'China', 'China', 'China', 'China',

'India', 'India', 'India', 'India', 'India'],

'year': [2000, 2005, 2010, 2015, 2020,

2000, 2005, 2010, 2015, 2020,

2000, 2005, 2010, 2015, 2020],

'lifeExp': [76.8, 78.2, 78.7, 79.1, 78.9,

71.4, 73.6, 75.2, 76.2, 77.3,

63.6, 65.3, 66.7, 67.9, 69.4]

}

# Создание DataFrame

df = pd.DataFrame(data)

# Линейный график

fig\_line = px.line(df[df['country'].isin(['United States', 'China', 'India'])],

x='year', y='lifeExp', color='country',

title='Изменение ожидаемой продолжительности жизни по странам')

fig\_line.write\_html("line\_plot.html")

# Гистограмма

fig\_bar = px.bar(df[df['country'].isin(['United States', 'China', 'India'])],

x='country', y='lifeExp',

title='Ожидаемая продолжительность жизни по странам',

color='country')

fig\_bar.write\_html("bar\_plot.html")

# Диаграмма рассеяния

fig\_scatter = px.scatter(df,

x='year', y='lifeExp',

color='country',

title='Диаграмма рассеяния: Год vs Ожидаемая продолжительность жизни')

fig\_scatter.write\_html("scatter\_plot.html")

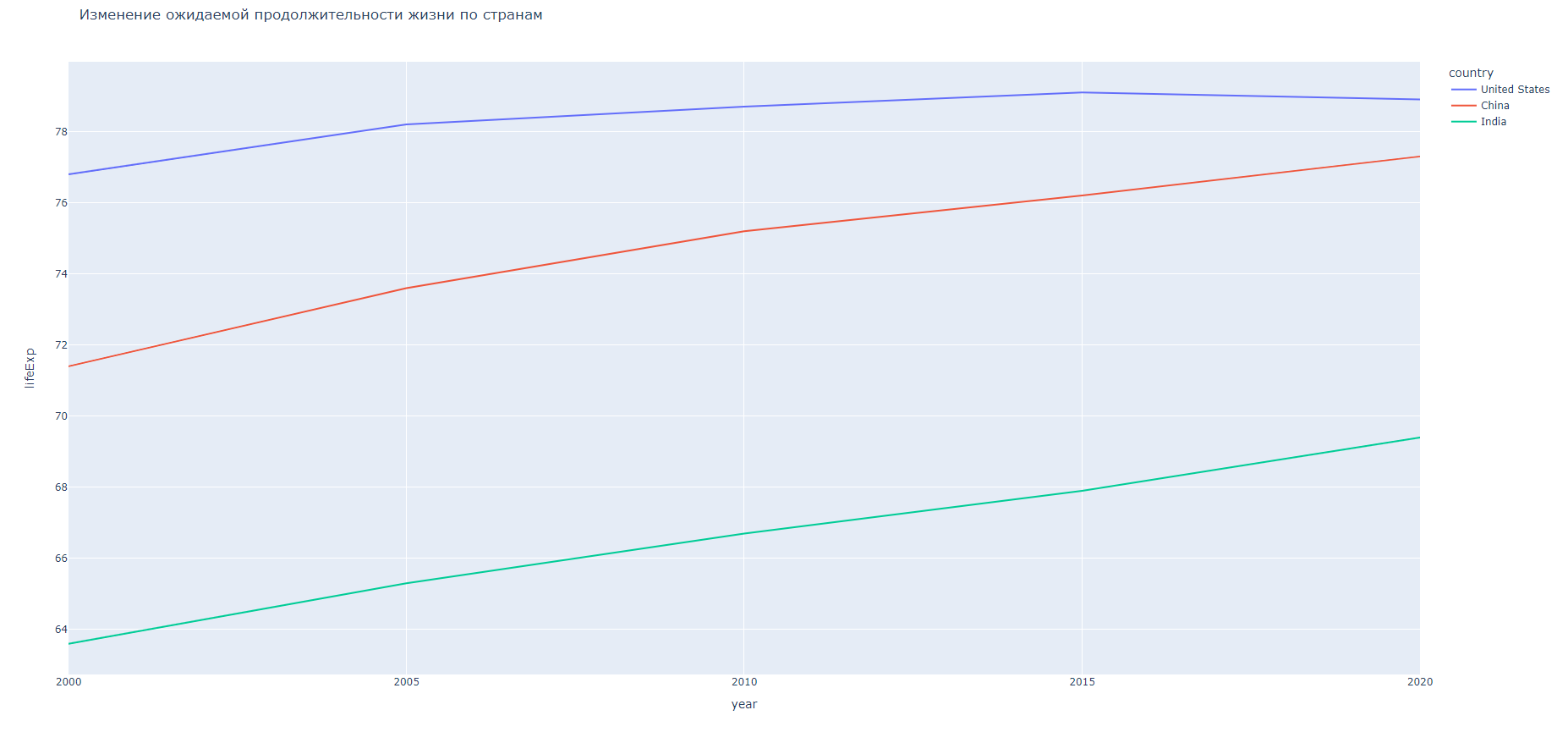
# Круговая диаграмма

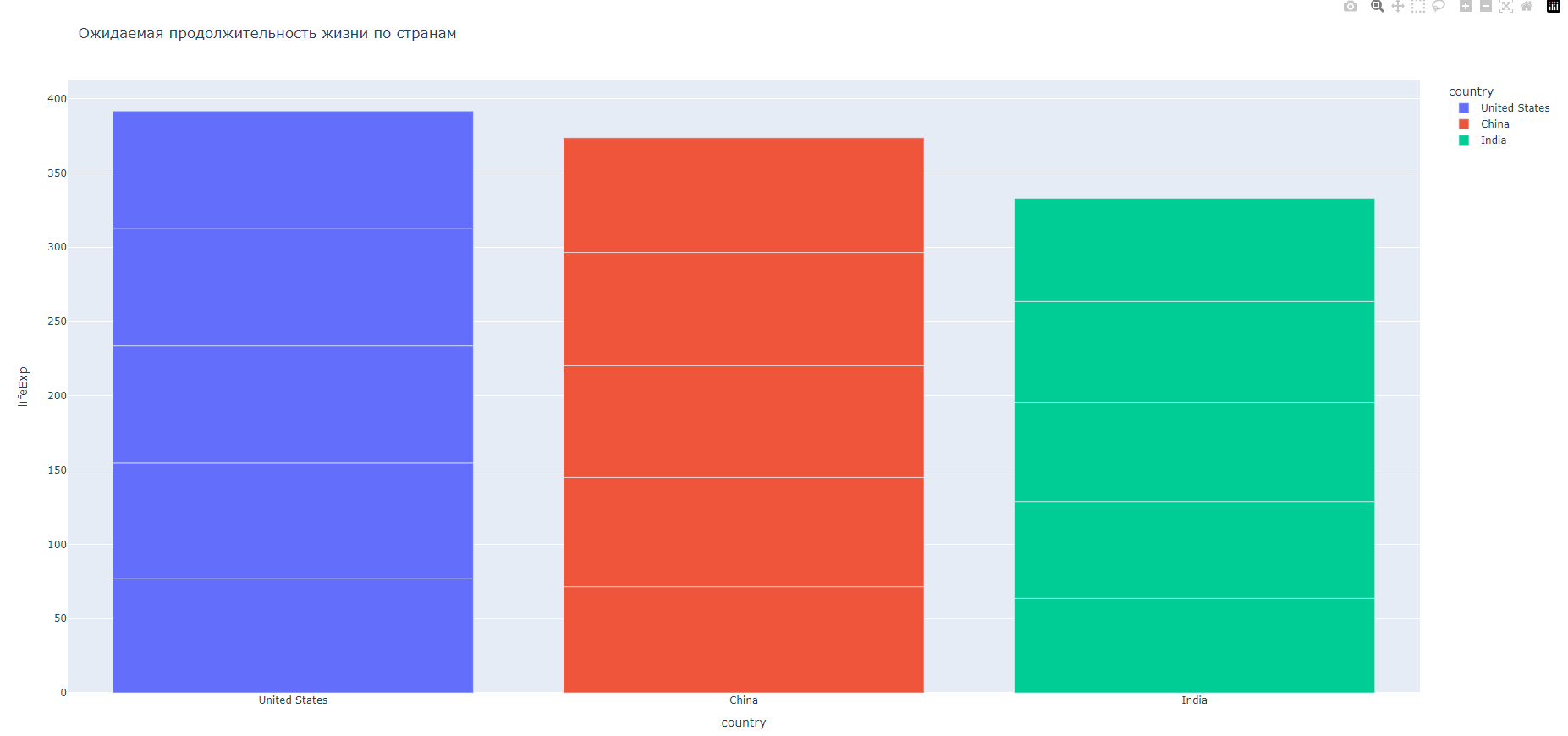
fig\_pie = px.pie(df.groupby('country')['lifeExp'].mean().reset\_index(),

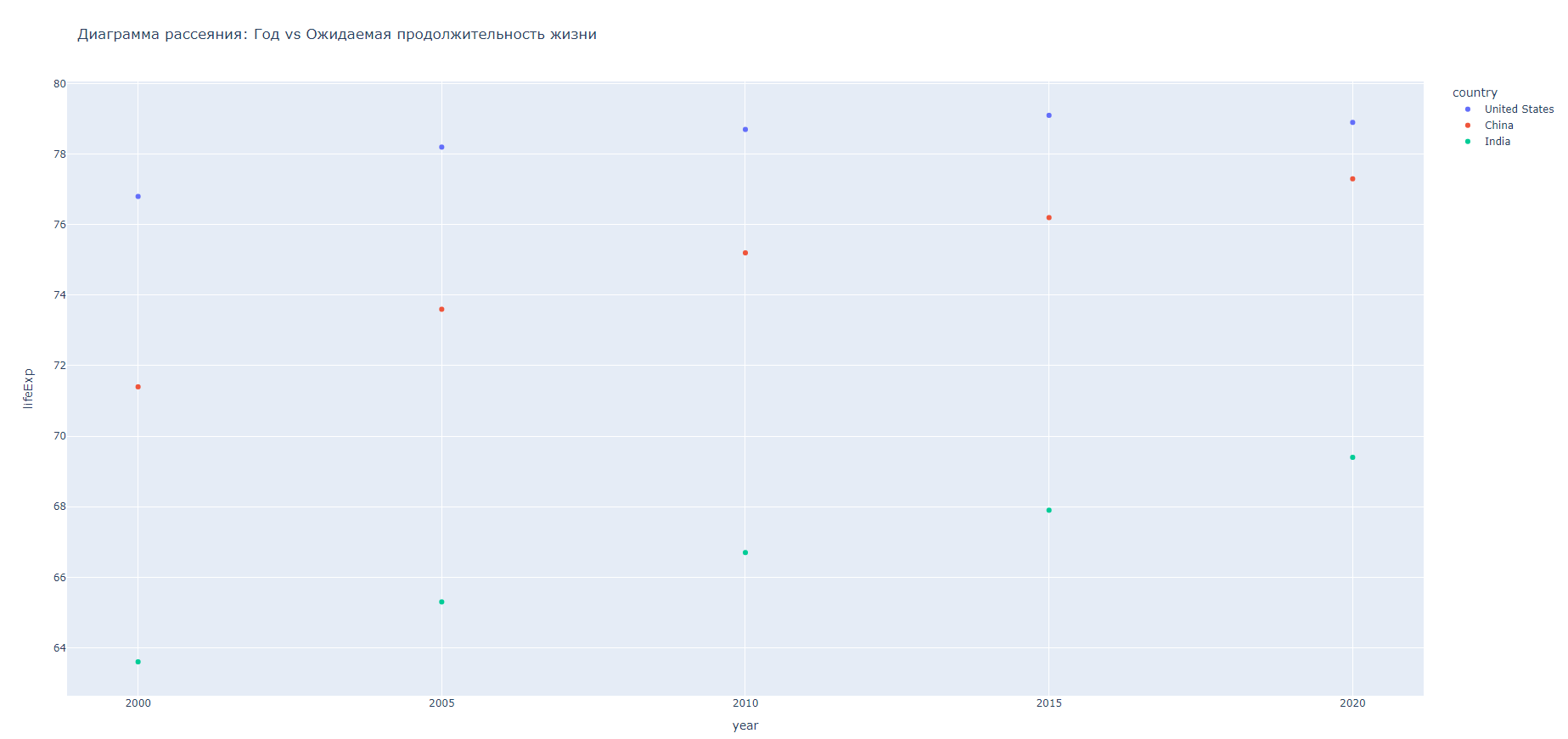
values='lifeExp', names='country',

title='Средняя ожидаемая продолжительность жизни по странам')

fig\_pie.write\_html("pie\_chart.html")







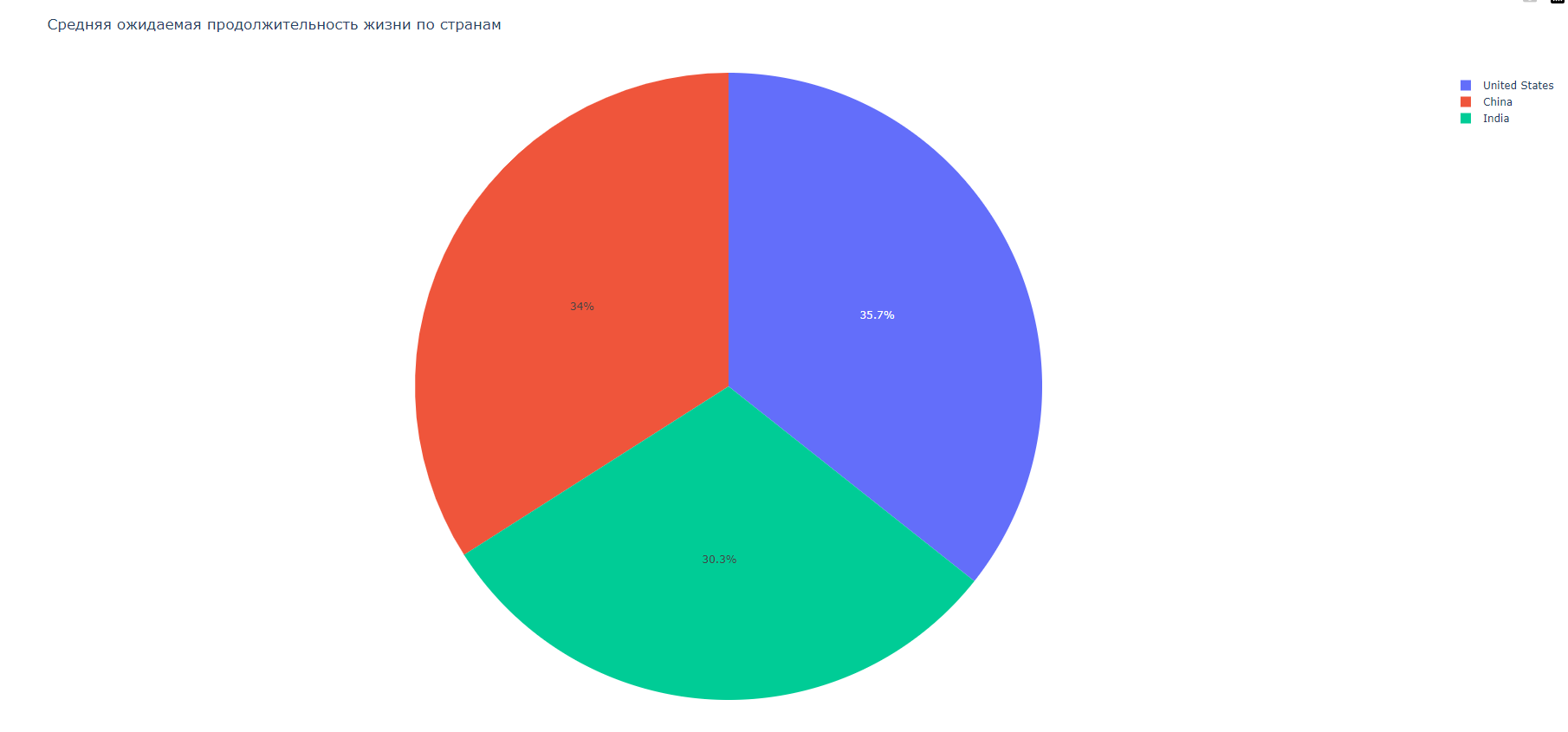


Рис.20,21,22,23 Результат выполнения с использованием Plotly.