Geekbrains

**Разработка и применение методов веб-скрапинга для автоматизированного извлечения данных с веб-ресурсов и их последующего анализа для принятия бизнес-решений**

Программа: Разработчик - Аналитик

Специализация: Data Engineer

ФИО: Кожемякова Елена Олеговна

Роквилл

2024

**Содержание**

[Введение 4](#_Toc175916719)

[Обоснование актуальности темы 4](#_Toc175916720)

[Цели и задачи исследования 4](#_Toc175916721)

[Для выполнения проекта будут использоваться следующие инструменты: 5](#_Toc175916722)

[В проекте будут использованы следующие технологии: 5](#_Toc175916723)

[Глава 1. Основы веб-скрапинга 6](#_Toc175916724)

[1.1. Что такое веб-скрапинг и его особенности 6](#_Toc175916725)

[Определение веб-скрапинга 6](#_Toc175916726)

[История и развитие веб-скрапинга 6](#_Toc175916727)

[Современные тенденции в веб-скрапинге 7](#_Toc175916728)

[Примеры использования веб-скрапинга 8](#_Toc175916729)

[Методы и подходы к веб-скрапингу 9](#_Toc175916730)

[Технические и организационные аспекты веб-скрапинга 9](#_Toc175916731)

[1.2 Анализ данных с использованием pandas и scikit-learn 10](#_Toc175916732)

[Статистический анализ данных 10](#_Toc175916733)

[Применение методов машинного обучения 13](#_Toc175916734)

[1.3 Юридические и этические аспекты Веб-скрапинга 17](#_Toc175916735)

[Правовые ограничения и авторское право 17](#_Toc175916736)

[Этические вопросы и лучшие практики 18](#_Toc175916737)

[Глава 2. Подготовка к веб-скрапингу 20](#_Toc175916738)

[2.1 Сбор требований и выбор целевых веб-ресурсов 20](#_Toc175916739)

[Определение цели сбора данных 20](#_Toc175916740)

[Анализ потребностей и определение целей 20](#_Toc175916741)

[Определение ключевых метрик успеха 21](#_Toc175916742)

[Критерии выбора веб-ресурсов 21](#_Toc175916743)

[Анализ доступных ресурсов 21](#_Toc175916744)

[Примеры подходящих веб-ресурсов 22](#_Toc175916745)

[Оценка и отбор ресурсов 22](#_Toc175916746)

[2.2 Разработка архитектуры скрапера 23](#_Toc175916747)

[Определение структуры и логики скрапера 23](#_Toc175916748)

[Потоковая обработка данных 24](#_Toc175916749)

[Проектирование схемы базы данных 25](#_Toc175916750)

[Пример проектирования схемы базы данных 26](#_Toc175916751)

[Нормализация данных 27](#_Toc175916752)

[Денормализация данных 27](#_Toc175916753)

[Индексация данных 28](#_Toc175916754)

[2.3. Настройка окружения для веб-скрапинга 28](#_Toc175916755)

[Установка и настройка инструментов 28](#_Toc175916756)

[Работа с виртуальными окружениями 30](#_Toc175916757)

[Глава 3. Реализация веб-скрапинга для анализа данных в электронной коммерции. 34](#_Toc175916758)

[3.1 Извлечение данных с помощью BeautifulSoup 34](#_Toc175916759)

[Установка BeautifulSoup и зависимостей 34](#_Toc175916760)

[Основы работы с BeautifulSoup 34](#_Toc175916761)

[Извлечение данных 35](#_Toc175916762)

[Обработка данных 36](#_Toc175916763)

[Сохранение данных 36](#_Toc175916764)

[Примеры использования 37](#_Toc175916765)

[3.2 Извлечение данных с помощью Scrapy 37](#_Toc175916766)

[Основы работы с Scrapy 38](#_Toc175916767)

[Установка Scrapy 38](#_Toc175916768)

[Создание проекта Scrapy 38](#_Toc175916769)

[Создание паука 38](#_Toc175916770)

[Примеры кода и пояснения 39](#_Toc175916771)

[Обработка данных 40](#_Toc175916772)

[Пример обработки данных: Извлечение изображений 40](#_Toc175916773)

[Сохранение данных 42](#_Toc175916774)

[3.3 Извлечение данных с помощью Selenium 42](#_Toc175916775)

[Основы работы с Selenium 43](#_Toc175916776)

[Глава 4. Анализ и обработка данных 45](#_Toc175916777)

[4.1 Предварительная обработка данных 45](#_Toc175916778)

[Очистка данных 45](#_Toc175916779)

[Обработка пропущенных значений 46](#_Toc175916780)

[4.2 Анализ данных с использованием pandas и scikit-learn 48](#_Toc175916781)

[Применение методов машинного обучения 49](#_Toc175916782)

[4.3 Визуализация данных 51](#_Toc175916783)

[Инструменты для визуализации 51](#_Toc175916784)

[Глава 5. Применение результатов анализа для принятия бизнес-решений 55](#_Toc175916785)

[5.1 Прогнозирование спроса и оптимизация запасов 55](#_Toc175916786)

[Применение моделей машинного обучения 55](#_Toc175916787)

[Линейная регрессия 55](#_Toc175916788)

[Модели временных рядов 57](#_Toc175916789)

[Пример использования ARIMA 57](#_Toc175916790)

[Интерпретация результатов 58](#_Toc175916791)

[Интерпретация коэффициентов регрессии 58](#_Toc175916792)

[Визуализация прогнозов 58](#_Toc175916793)

[Оптимизация запасов 60](#_Toc175916794)

[5.2 Анализ конкурентной среды 60](#_Toc175916795)

[Сравнительный анализ данных 60](#_Toc175916796)

[Сбор и обработка данных 60](#_Toc175916797)

[Анализ позиций на рынке 62](#_Toc175916798)

[Разработка стратегий на основе полученных данных 62](#_Toc175916799)

[5.3 Рекомендации по улучшению бизнес-процессов 63](#_Toc175916800)

[Внедрение предложений на основе анализа данных 64](#_Toc175916801)

[Пример внедрения предложений 64](#_Toc175916802)

[Оценка эффективности внедренных изменений 65](#_Toc175916803)

[Пример оценки эффективности 65](#_Toc175916804)

[Заключение 67](#_Toc175916805)

[Выводы по проделанной работе 67](#_Toc175916806)

[Достижение поставленных целей 68](#_Toc175916807)

[Перспективы дальнейших исследований 68](#_Toc175916808)

[Список используемой литературы 70](#_Toc175916809)

[Книги 70](#_Toc175916810)

[Статьи 70](#_Toc175916811)

[Веб-ресурсы 70](#_Toc175916812)

[Приложения 71](#_Toc175916813)

[Исходные коды: 71](#_Toc175916814)

[Дополнительные материалы: 71](#_Toc175916815)

# Введение

### Обоснование актуальности темы

Современный мир стремительно движется в направлении цифровизации, и огромное количество информации, ежедневно создаваемой в интернете, требует автоматизированных методов сбора и обработки. Веб-скрапинг — это одна из ключевых технологий, позволяющих извлекать данные из веб-источников и превращать их в структурированный формат для дальнейшего анализа. С учетом возросшего объема данных, веб-скрапинг становится неотъемлемым инструментом для аналитиков, исследователей и разработчиков, работающих в различных областях — от маркетинга и социологии до финансов и медицины.

Актуальность веб-скрапинга также обусловлена его широким спектром применения. Компании могут использовать эту технологию для мониторинга цен у конкурентов, анализа отзывов потребителей, сбора данных для маркетинговых исследований и многого другого. В условиях жесткой конкурентной борьбы и стремительного развития интернет-технологий, способность оперативно получать и обрабатывать информацию становится важным преимуществом.

Однако, несмотря на популярность и широкое использование веб-скрапинга, существует множество технических и этических проблем, связанных с его применением. Это включает в себя вопросы правового регулирования, обхода CAPTCHA и защиты от блокировки IP-адресов, а также необходимость соблюдения правил использования данных, установленных владельцами сайтов. Данная дипломная работа направлена на исследование и решение некоторых из этих проблем, что подчеркивает ее актуальность и практическую значимость.

### Цели и задачи исследования

Целью данной дипломной работы является исследование методов и инструментов веб-скрапинга, а также разработка и реализация проекта по автоматическому сбору данных с веб-сайтов для последующего анализа. Для достижения этой цели необходимо решить следующие задачи:

1. Изучить литературу и современные подходы к веб-скрапингу.
2. Описать основные инструменты и библиотеки, используемые для веб-скрапинга, такие как BeautifulSoup, Scrapy и Selenium.
3. Исследовать этические и правовые аспекты использования веб-скрапинга.
4. Разработать план и архитектуру проекта по сбору данных с конкретного веб-ресурса.
5. Реализовать проект, используя выбранные инструменты и библиотеки.
6. Оценить эффективность и производительность разработанного решения.
7. Сформулировать рекомендации по улучшению и дальнейшему развитию проекта.

Данная работа охватывает как теоретические, так и практические аспекты веб-скрапинга, предлагая полезные результаты для специалистов по данным и компаний, стремящихся улучшить свои аналитические возможности и конкурентные преимущества. Введение закладывает основу для всестороннего изучения и применения технологий веб-скрапинга. В последующих главах будут подробно рассмотрены методы и инструменты автоматического сбора данных, а также разработан и реализован проект, демонстрирующий практическую значимость веб-скрапинга в современном цифровом мире.

### Для выполнения проекта будут использоваться следующие инструменты:

* BeautifulSoup и Scrapy для веб-скрапинга
* Selenium для автоматизации взаимодействия с веб-страницами
* Pandas и Scikit-learn для анализа данных
* Matplotlib и Seaborn для визуализации данных

### В проекте будут использованы следующие технологии:

* Python как основной язык программирования
* Базы данных для хранения собранных данных
* Git для контроля версий кода(https://github.com/yokozhem/graduation-project-GB-data-engineer)

**Проект выполнялся индивидуально. В рамках проекта были выполнены следующие роли:**

* Исследователь: анализ инструментов и технологий
* Разработчик: создание и настройка скрапер-сценариев
* Аналитик данных: предварительная обработка и анализ данных
* Документатор: подготовка документации и отчета по результатам работы

# Глава 1. Основы веб-скрапинга

## 1.1. Что такое веб-скрапинг и его особенности

### Определение веб-скрапинга

Веб-скрапинг, или веб-парсинг, представляет собой процесс автоматизированного извлечения данных из веб-страниц. С помощью специальных программных средств, называемых скрейперами, можно автоматически собирать информацию с различных сайтов и превращать её в структурированный формат, пригодный для дальнейшего анализа. Скрейперы имитируют действия пользователя, посещая веб-страницы, извлекая необходимую информацию и сохраняют её в виде таблиц, баз данных или других форматов.

Основные компоненты веб-скрапинга включают:

1. **URL-парсинг:** процесс извлечения и обработки URL-адресов, по которым будет производиться сбор данных.
2. **Загрузка страницы:** скрейпер отправляет HTTP-запросы на сервер и получает HTML-код веб-страницы.
3. **Анализ HTML:** с помощью парсеров, таких как BeautifulSoup или lxml, скрейпер разбирает HTML-код и извлекает интересующие данные.
4. **Сохранение данных:** извлеченные данные сохраняются в удобном для анализа формате, например, в виде CSV-файлов, JSON или баз данных.

Веб-скрапинг используется в различных областях, включая маркетинг, аналитику, научные исследования, разработку программного обеспечения и многое другое.

### История и развитие веб-скрапинга

#### Начало веб-скрапинга

История веб-скрапинга берет своё начало с зарождения Всемирной паутины. В первые годы существования интернета, разработчики вручную копировали данные с веб-страниц. С ростом объема информации и числа веб-сайтов, такой подход стал неэффективным и трудоемким. Появилась необходимость в автоматизированных методах сбора данных.

В 1990-е годы начали появляться первые инструменты для автоматизации сбора данных из интернета. Одним из таких инструментов стал **Perl**, который использовался для написания скриптов, имитирующих действия пользователя и извлекающих данные с веб-страниц. На этом этапе веб-скрапинг был простым и ограниченным из-за невысокой сложности веб-страниц того времени.

#### Развитие веб-скрапинга

С развитием веб-технологий, таких как JavaScript, AJAX и CSS, веб-страницы стали более сложными и динамичными. Это потребовало создания более продвинутых инструментов для веб-скрапинга. В начале 2000-х годов появились первые специализированные библиотеки и фреймворки для парсинга данных, такие как **BeautifulSoup** для языка Python. BeautifulSoup позволял легко разбирать и анализировать HTML-код, что значительно упростило процесс извлечения данных.

Параллельно с развитием парсеров, начали развиваться инструменты для автоматизации взаимодействия с веб-страницами. Одним из таких инструментов стал **Selenium**, который изначально был разработан для автоматизированного тестирования веб-приложений. Selenium позволяет имитировать действия пользователя, такие как клики, ввод текста и навигация по страницам, что делает его мощным инструментом для веб-скрапинга, особенно в случае динамических веб-сайтов.

### Современные тенденции в веб-скрапинге

Сегодня веб-скрапинг является важным инструментом для многих областей, и его развитие продолжается с появлением новых технологий и методов. Современные инструменты для веб-скрапинга включают в себя мощные фреймворки, такие как **Scrapy** и **Puppeteer**, которые предоставляют множество возможностей для автоматизации сбора данных.

**Scrapy** является одним из самых популярных фреймворков для веб-скрапинга. Он позволяет легко создавать скрейперы, управлять процессом сбора данных, обрабатывать и сохранять данные. Scrapy поддерживает сложные сценарии сбора данных, такие как обработка форм, работа с JavaScript и управление сессиями.

**Puppeteer** — это инструмент от Google, который позволяет управлять браузером Chrome или Chromium через DevTools протокол. Puppeteer особенно полезен для работы с динамическими веб-сайтами, где данные загружаются с помощью JavaScript. Puppeteer предоставляет удобный интерфейс для взаимодействия с веб-страницами и позволяет делать скриншоты, генерировать PDF-документы и собирать данные.

Другие современные инструменты включают **Playwright**, который предоставляет аналогичные возможности для управления браузером, и **Octoparse**, который является облачным сервисом для веб-скрапинга, не требующим программирования.

### Примеры использования веб-скрапинга

Веб-скрапинг применяется в самых различных областях, где требуется автоматический сбор и анализ данных:

1. **Маркетинговые исследования:** компании используют веб-скрапинг для мониторинга цен конкурентов, анализа отзывов потребителей и изучения рыночных тенденций. Например, интернет-магазины могут собирать данные о ценах на аналогичные товары у конкурентов и на основе этой информации корректировать свои цены.
2. **Научные исследования:** исследователи используют веб-скрапинг для сбора больших объемов данных из научных публикаций, социальных сетей и других источников для проведения анализа и разработки моделей. Например, анализ данных социальных сетей позволяет изучать поведение пользователей и выявлять тренды.
3. **Финансовый анализ:** в финансовой сфере веб-скрапинг используется для сбора данных о котировках акций, новостях компаний и других финансовых показателях. Эти данные помогают аналитикам делать прогнозы и принимать обоснованные инвестиционные решения.
4. **Анализ текстов:** веб-скрапинг применяется для сбора текстовых данных из блогов, форумов и новостных сайтов. Эти данные используются для анализа мнений, проведения исследований в области лингвистики и разработки систем автоматического анализа текстов.
5. **Разработка чат-ботов:** данные, собранные с помощью веб-скрапинга, могут использоваться для обучения чат-ботов. Например, чат-боты для поддержки клиентов могут использовать данные из FAQ и форумов для автоматического ответа на вопросы пользователей.
6. **Мониторинг новостей и социальных медиа:** компании и исследовательские организации используют веб-скрапинг для мониторинга упоминаний брендов, продуктов и ключевых событий в новостях и социальных медиа. Это помогает оперативно реагировать на изменения и принимать стратегические решения.

### Методы и подходы к веб-скрапингу

Существуют различные методы и подходы к веб-скрапингу, каждый из которых имеет свои особенности и области применения:

1. **Парсинг HTML:** это наиболее распространенный метод веб-скрапинга, который заключается в анализе и извлечении данных из HTML-кода веб-страниц. Используются библиотеки и инструменты, такие как BeautifulSoup, lxml и XPath.
2. **Парсинг API:** некоторые веб-сайты предоставляют программные интерфейсы (API), через которые можно получать данные в структурированном формате (например, JSON или XML). Этот метод удобен и легален, но требует наличия открытого API.
3. **Парсинг динамических страниц:** современные веб-страницы часто используют JavaScript для загрузки данных. Для работы с такими страницами применяются инструменты, такие как Selenium, Puppeteer и Playwright, которые позволяют имитировать действия пользователя и получать данные после выполнения скриптов.
4. **Обход ограничений:** некоторые веб-сайты ограничивают частоту запросов или блокируют скрейперов. Для обхода таких ограничений используются прокси-серверы, рандомизация заголовков запросов и другие методы маскировки.

### Технические и организационные аспекты веб-скрапинга

Организация процесса веб-скрапинга включает в себя несколько важных этапов:

1. **Выбор инструмента:** в зависимости от задачи и сложности веб-сайта выбирается подходящий инструмент или библиотека для веб-скрапинга. Например, для парсинга статических страниц подойдет BeautifulSoup, а для динамических — Selenium или Puppeteer.
2. **Разработка скрейпера:** создается скрипт или программа, которая будет выполнять сбор данных. Важно учитывать особенности структуры веб-страницы и правильно настроить парсер для извлечения нужной информации.
3. **Тестирование:** перед началом массового сбора данных скрейпер тестируется на небольшом объеме данных, чтобы убедиться в корректности работы и отсутствии ошибок.
4. **Сбор и обработка данных:** запускается процесс сбора данных. Полученные данные сохраняются в структурированном виде и проходят этап предварительной обработки для удаления дубликатов и исправления ошибок.
5. **Анализ и визуализация:** собранные данные анализируются и визуализируются с помощью инструментов для анализа данных, таких как pandas, NumPy и matplotlib.
6. **Обновление данных:** для поддержания актуальности данных процесс веб-скрапинга может быть автоматизирован и запускаться периодически для обновления информации.

Веб-скрапинг представляет собой мощный инструмент для автоматизации сбора данных из интернета. С его помощью можно извлекать большие объемы информации и использовать их для анализа, принятия решений и разработки новых технологий. Несмотря на правовые и этические вызовы, веб-скрапинг продолжает развиваться, предоставляя новые возможности и инструменты для работы с данными.

## 1.2 Анализ данных с использованием pandas и scikit-learn

Анализ данных играет ключевую роль в современных исследованиях и проектах, связанных с обработкой больших объемов информации. Для проведения анализа данных часто используются такие библиотеки Python, как Pandas и Scikit-learn. В данном разделе мы рассмотрим, как с их помощью можно выполнять статистический анализ данных и применять методы машинного обучения.

### Статистический анализ данных

#### Основы работы с Pandas

Pandas — это мощная библиотека для работы с данными, которая предоставляет удобные структуры данных и функции для их манипуляции и анализа. Основные структуры данных Pandas — это DataFrame и Series. DataFrame представляет собой двумерную таблицу, аналогичную таблице в базе данных или электронных таблицах, в то время как Series — это одномерный массив данных.

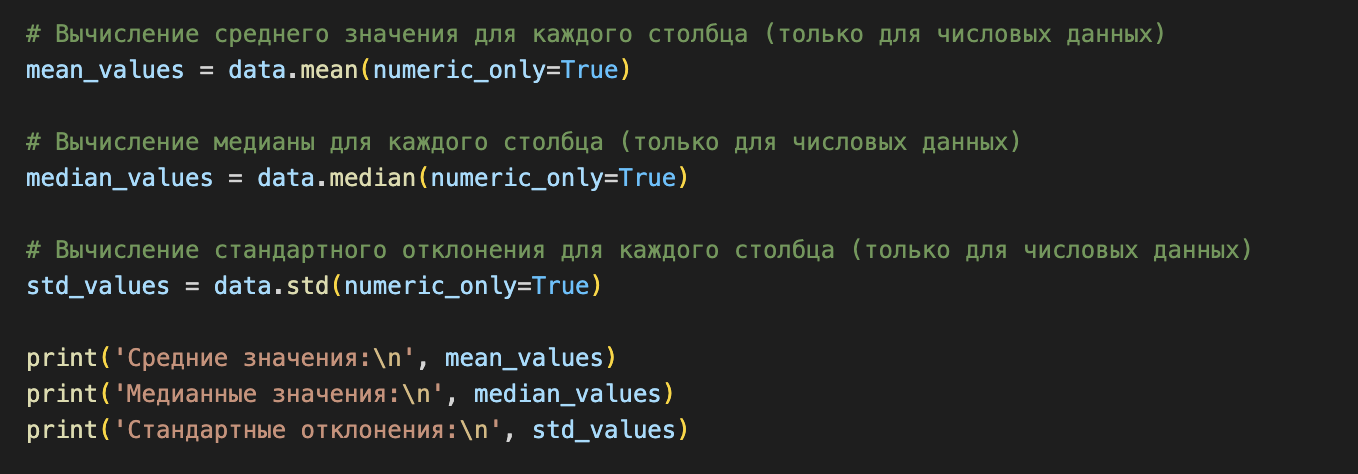
Для начала работы с Pandas необходимо импортировать библиотеку и загрузить данные:



Приложение 1.

#### Описательная статистика

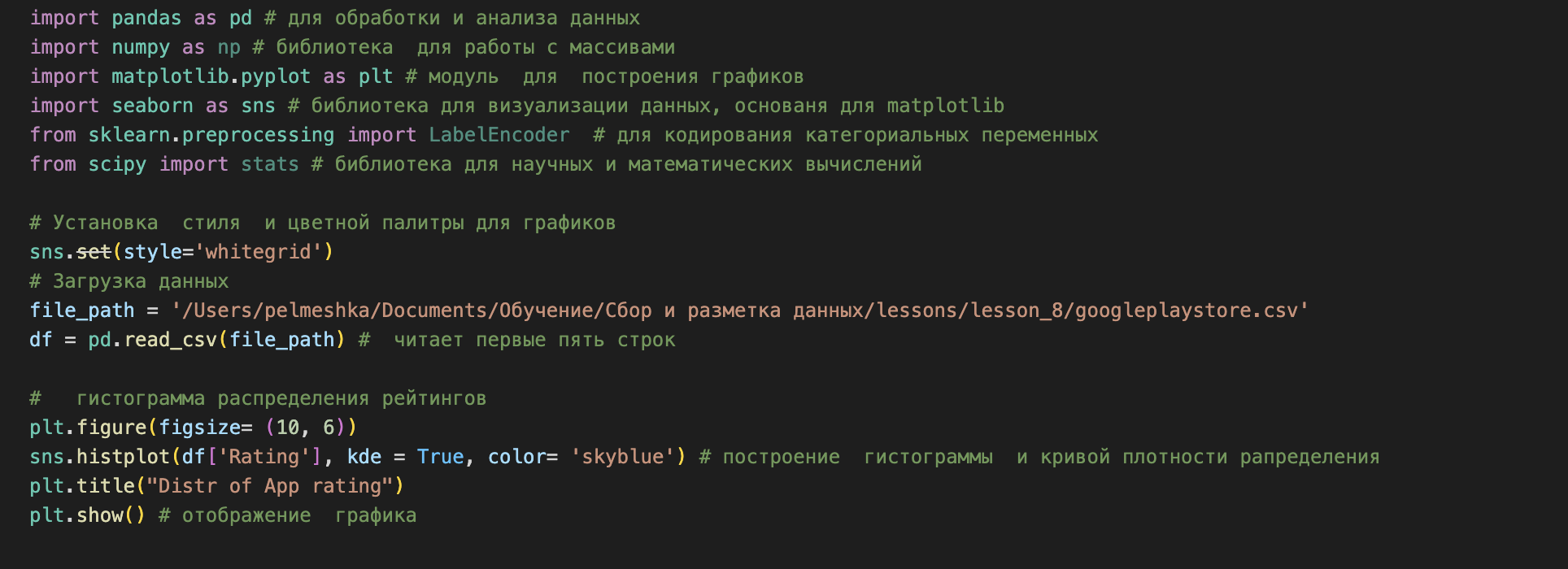
Pandas предоставляет множество функций для выполнения описательной статистики, таких как mean, median, mode, std, var и других. Эти функции позволяют получить общее представление о данных и их распределении:



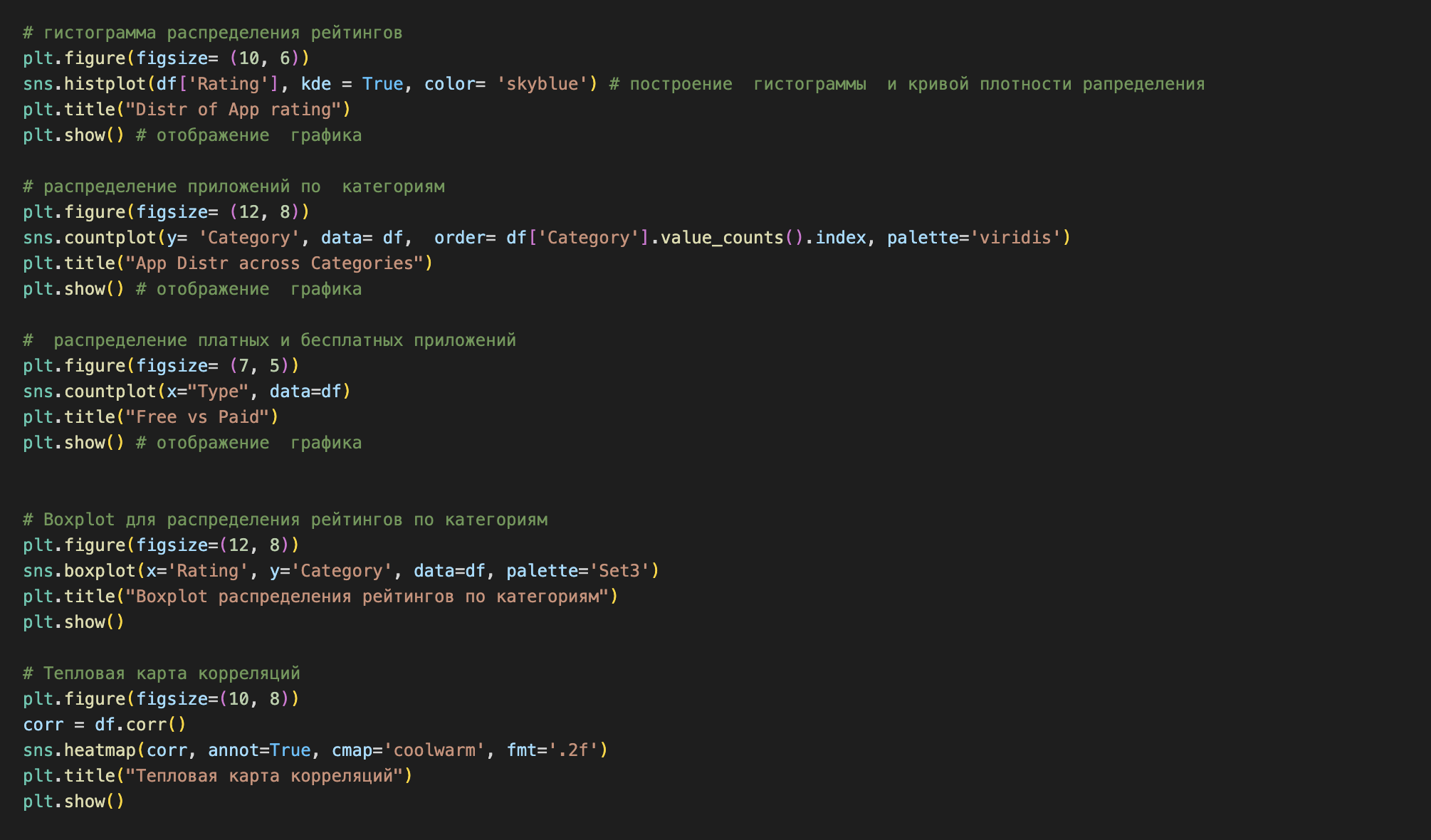
Приложение 1.

#### Визуализация данных

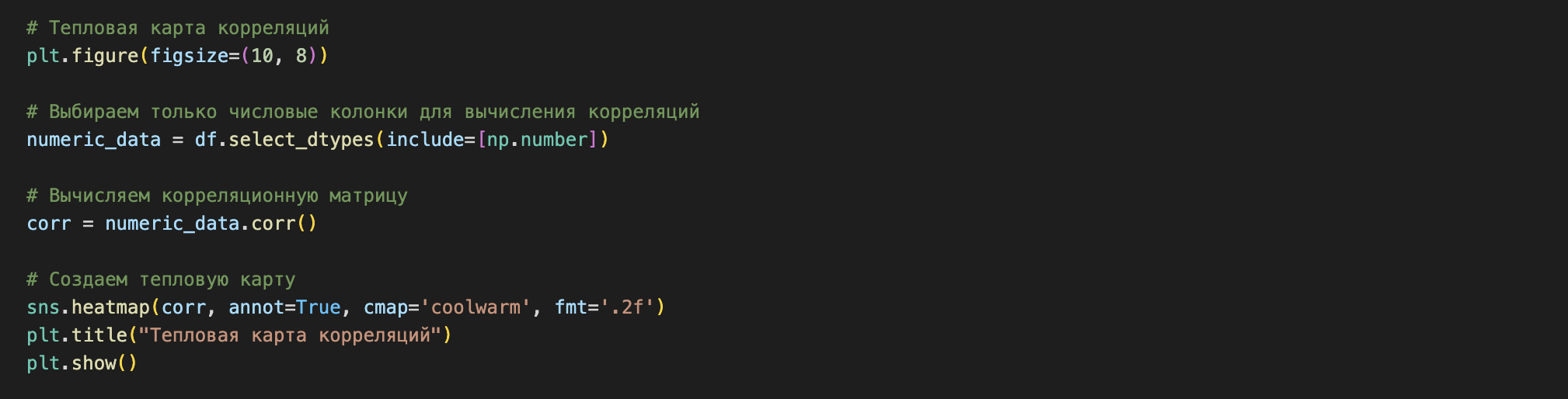
Для визуализации данных можно использовать такие библиотеки, как Matplotlib и Seaborn. Они позволяют создавать различные графики и диаграммы, которые помогают лучше понять данные и выявить закономерности:



Пример 2.



Пример 2.



Пример 2.

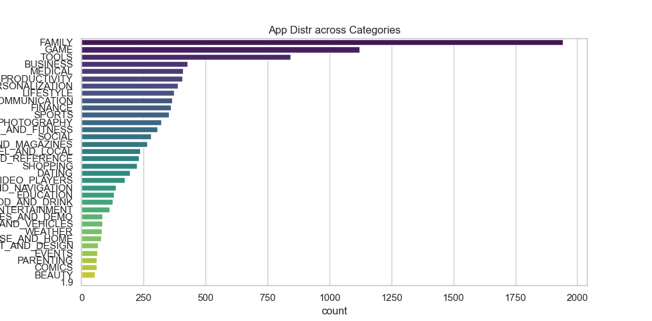
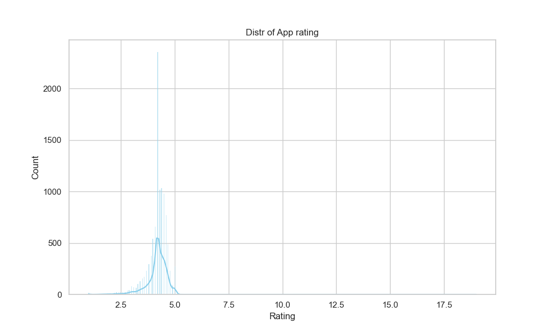


Рисунок 1 и 2.

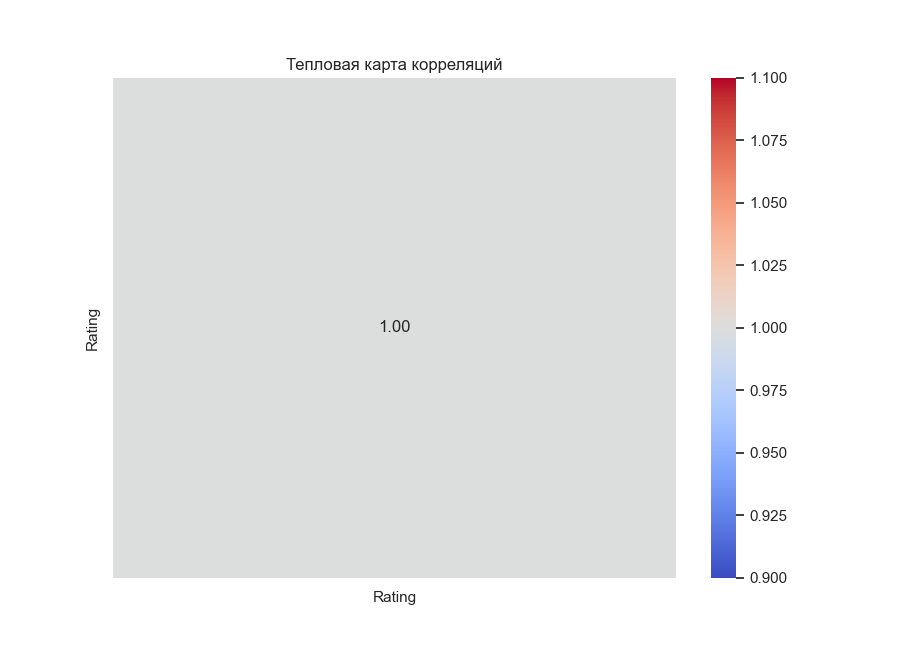
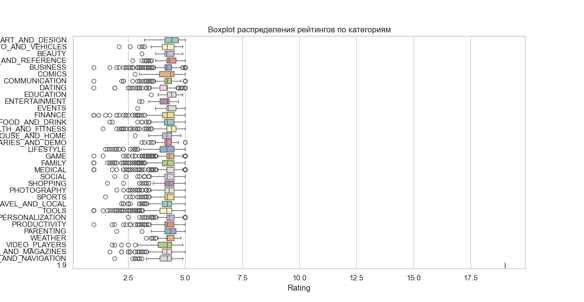


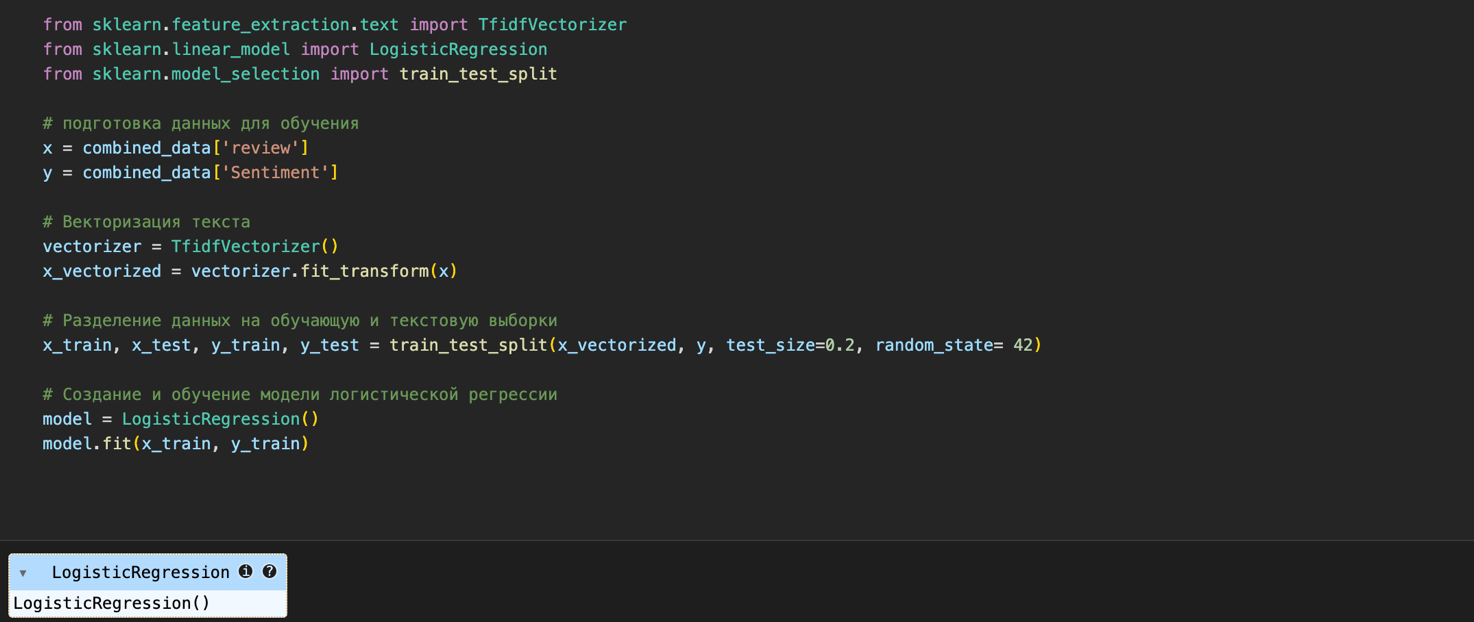
Рисунок 3 и 4.

### Применение методов машинного обучения

#### Основы работы с Scikit -learn

Scikit-learn — это библиотека для машинного обучения в Python, которая предоставляет простые и эффективные инструменты для анализа данных и моделирования. Она включает в себя множество алгоритмов для классификации, регрессии, кластеризации и снижения размерности.

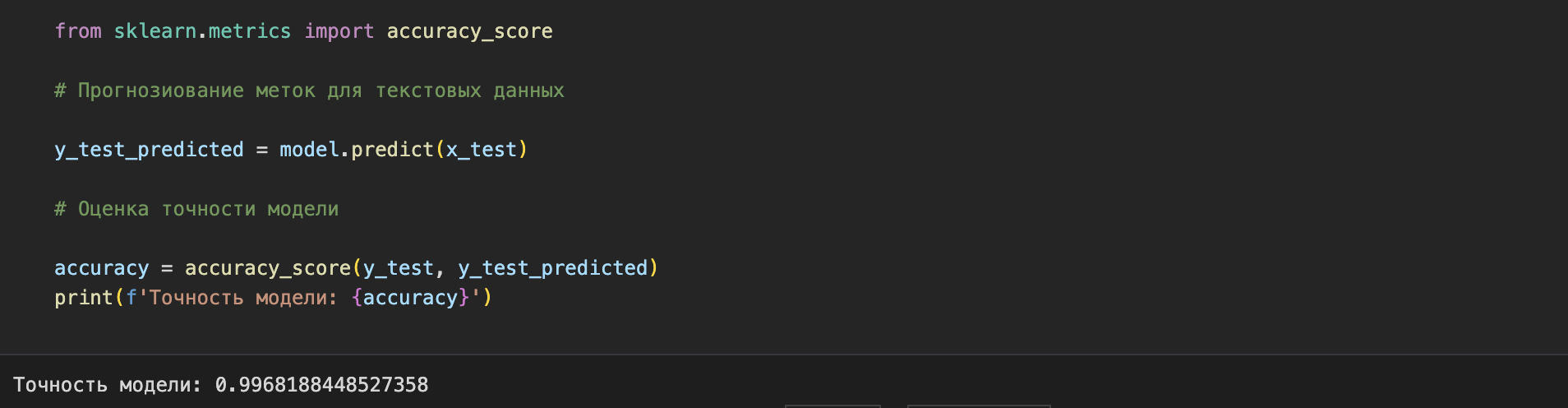
Для начала работы с Scikit -learn необходимо импортировать необходимые модули и разделить данные на обучающую и тестовую выборки:



Пример 3.

#### Классификация

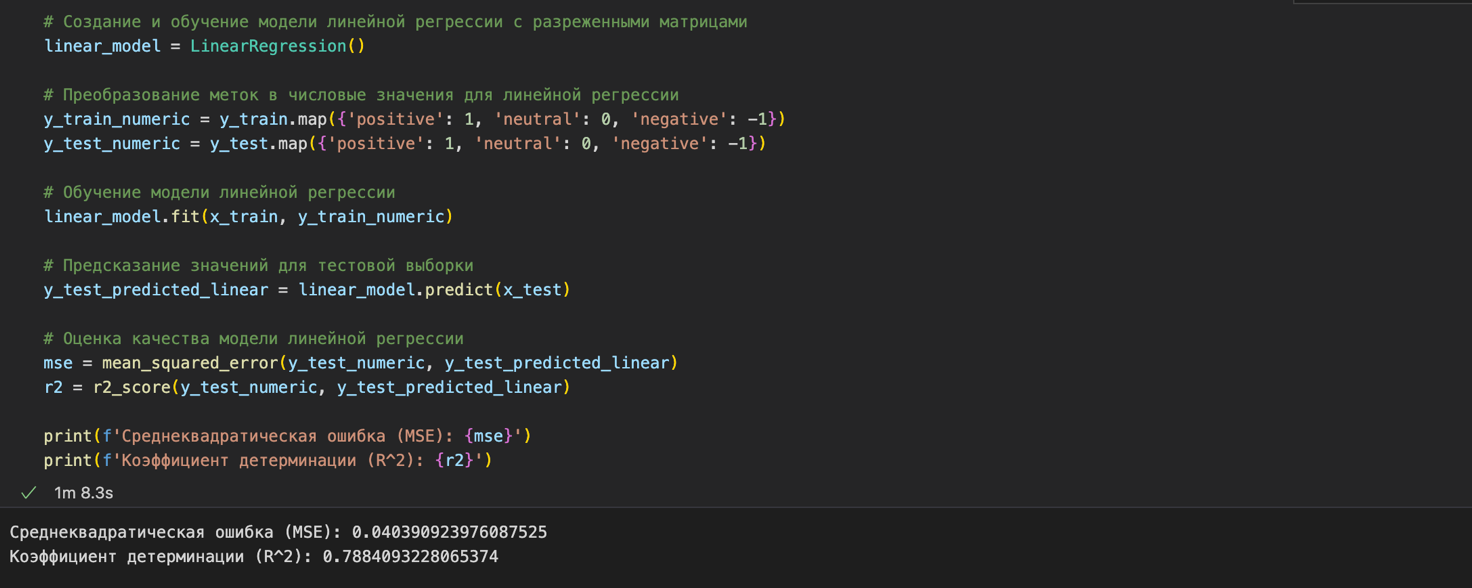
Классификация — это задача машинного обучения, в которой модель обучается на основе обучающей выборки и предсказывает метки классов для новых данных. Рассмотрим пример использования алгоритма логистической регрессии для классификации данных:



Пример 3.

#### Регрессия

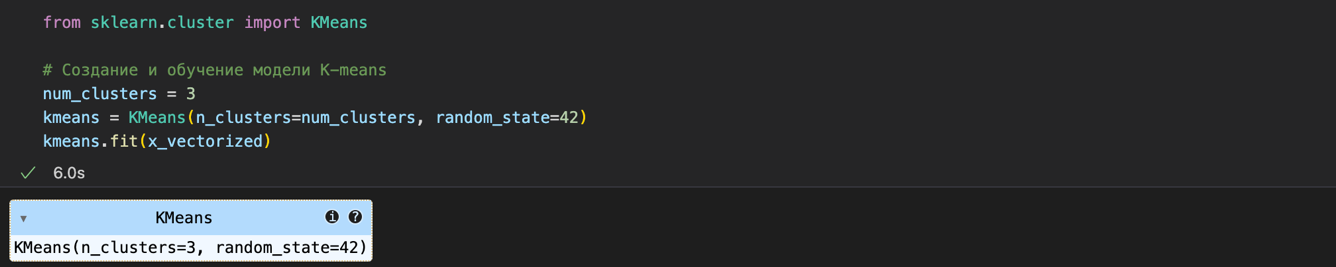
Регрессия — это задача машинного обучения, в которой модель обучается предсказывать непрерывные значения. Рассмотрим пример использования алгоритма линейной регрессии для предсказания значений:



Пример 3.

#### Кластеризация

Кластеризация — это задача машинного обучения, в которой модель группирует данные на основе их сходства. Рассмотрим пример использования алгоритма k-means для кластеризации данных:



Пример 3.



Пример 3.

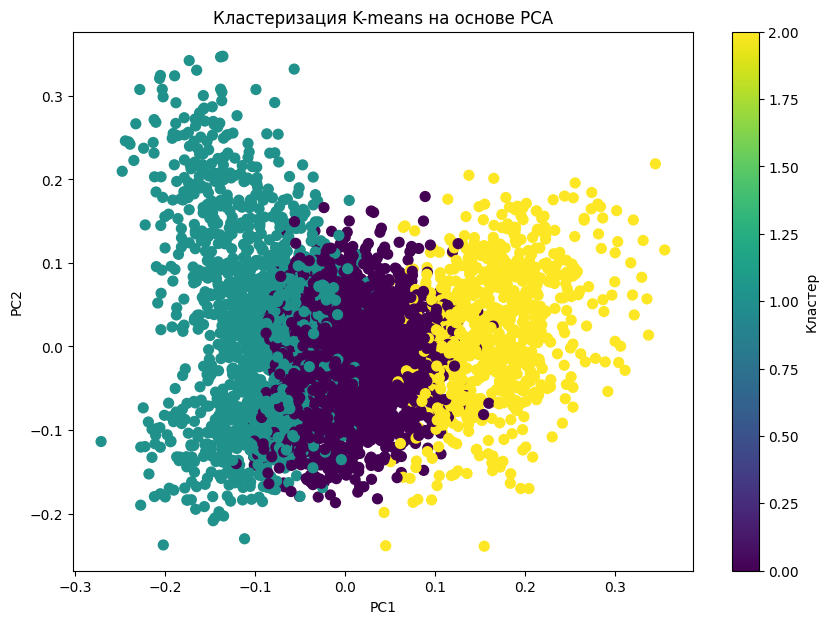


Рисунок 5.

Использование pandas и scikit-learn позволяет эффективно проводить анализ данных и применять методы машинного обучения. Pandas предоставляет мощные инструменты для манипуляции и визуализации данных, в то время как scikit-learn обеспечивает широкий выбор алгоритмов для моделирования и анализа. Совместное использование этих библиотек помогает решать различные задачи, такие как классификация, регрессия и кластеризация, и получать ценные инсайты из данных.

## 1.3 Юридические и этические аспекты Веб-скрапинга

При использовании Веб-скрапинга возникают важные юридические и этические вопросы. Понимание правовых ограничений и соблюдение этических стандартов крайне важно для всех, кто занимается веб-скрапингом, чтобы избежать возможных юридических проблем и сохранить репутацию.

### Правовые ограничения и авторское право

#### Законы об авторском праве

Законы об авторском праве защищают оригинальные произведения, включая текст, изображения, и другие материалы, размещенные на веб-сайтах. Веб-скрапинг может нарушать эти права, если данные используются без разрешения владельца контента. В разных странах законы об авторском праве могут существенно различаться, поэтому важно понимать местные правовые нормы.

В Соединенных Штатах, например, Закон об авторском праве 1976 года и Закон о защите авторских прав в цифровую эпоху (DMCA) регулируют использование и защиту цифрового контента. В Европейском Союзе Директива о защите авторских прав на едином цифровом рынке (CDSM Directive) также предусматривает строгие правила защиты авторских прав.

#### Право на использование данных

Многие веб-сайты включают в свои условия использования (Terms of Service, TOS) положения, запрещающие автоматизированное извлечение данных. Нарушение этих условий может привести к юридическим санкциям, включая иски о нарушении авторских прав или незаконном доступе к данным. Например, в случае Facebook против Power Ventures, суд постановил, что доступ к веб-сайту в обход технических мер защиты, установленных сайтом, является незаконным.

#### Закон о компьютерном мошенничестве и злоупотреблениях (CFAA)

В США Закон о компьютерном мошенничестве и злоупотреблениях (CFAA) запрещает несанкционированный доступ к компьютерным системам. Веб-скрапинг, который нарушает условия использования веб-сайта или обход технических мер защиты, может быть квалифицирован как нарушение CFAA.

#### Доступность данных и лицензии

Некоторые веб-сайты предоставляют доступ к своим данным через API (интерфейсы программирования приложений), которые позволяют получать данные законным и контролируемым способом. Использование API часто требует соблюдения условий лицензии, которые могут ограничивать объем и способ использования данных.

### Этические вопросы и лучшие практики

#### Этические принципы веб-скрапинга

Этические аспекты веб-скрапинга включают в себя уважение к владельцам контента, соблюдение конфиденциальности пользователей и минимизацию воздействия на веб-сайты. Основные принципы, которых следует придерживаться:

1. **Прозрачность**: Уведомление владельцев сайтов о намерении собирать данные, если это возможно и уместно.
2. **Соблюдение условий использования**: Чтение и соблюдение условий использования веб-сайтов, чтобы избежать нарушений.
3. **Соблюдение конфиденциальности**: Избегание сбора персональных данных без явного согласия пользователей.
4. **Минимизация нагрузки**: Настройка скрапера таким образом, чтобы минимизировать нагрузку на серверы, например, использование тайм-аутов между запросами и уважение к файлу robots.txt.

#### Роботика и этика

Файл robots.txt — это стандарт, который позволяет владельцам веб-сайтов управлять поведением автоматизированных систем (роботов), таких как веб-скраперы. Уважение к указаниям в файле robots.txt является важной частью этических практик веб-скрапинга. Например, если файл robots.txt запрещает доступ к определенным разделам сайта, скрапер должен уважать эти ограничения.

#### Примеры хороших практик

1. **Использование API**: если веб-сайт предоставляет API для доступа к своим данным, предпочтительно использовать его вместо веб-скрапинга. API обычно предоставляют более стабильный и законный способ доступа к данным.
2. **Ограничение частоты запросов**: установка разумных ограничений на частоту запросов к веб-сайту, чтобы избежать перегрузки серверов и потенциальных отказов в обслуживании.
3. **Обработка ошибок и исключений**: реализация обработки ошибок и исключений, чтобы избежать повторных запросов в случае временных проблем с доступом к сайту.
4. **Документирование и аудит**: ведение документации о скраперских активностях и проведение регулярных аудитов для обеспечения соблюдения правовых и этических норм.

Юридические и этические аспекты веб-скрапинга играют критически важную роль в обеспечении законности и уважительности этого процесса. Понимание правовых рамок, таких как законы об авторском праве и условия использования веб-сайтов, помогает избежать юридических проблем. Соблюдение этических принципов, таких как прозрачность, уважение конфиденциальности и минимизация нагрузки на серверы, способствует созданию ответственной и устойчивой практики веб-скрапинга.

В конечном итоге, успех веб-скрапинга зависит не только от технических навыков, но и от осознания и соблюдения юридических и этических стандартов. Это не только помогает избежать проблем с законом, но и способствует развитию доверия и уважения в сообществе разработчиков и владельцев контента.

# Глава 2. Подготовка к веб-скрапингу

## 2.1 Сбор требований и выбор целевых веб-ресурсов

### Определение цели сбора данных

Веб-скрапинг начинается с четкого понимания целей, для которых будут собираться данные. Эти цели определяют, какие данные будут извлечены, как они будут использованы, и какие методы и инструменты будут применены в процессе. Понимание целей также помогает формировать критерии для выбора целевых веб-ресурсов и определять объем и формат данных, необходимых для выполнения поставленных задач.

### Анализ потребностей и определение целей

Первым шагом в любом проекте веб-скрапинга является анализ потребностей бизнеса или исследовательского проекта. Это включает в себя обсуждение с заинтересованными сторонами для понимания их требований и ожиданий от данных. Вопросы, которые нужно рассмотреть на этом этапе, включают:

* **Какие данные нужны?** Определите типы данных, которые будут наиболее полезны для проекта. Это могут быть текстовые данные, изображения, видео, метаданные, рейтинги, отзывы и т.д.
* **Как будут использованы данные?** Определите, как собранные данные будут применены. Это может включать аналитические отчеты, машинное обучение, маркетинговые исследования, улучшение пользовательского опыта и т.д.
* **Какие форматы данных необходимы?** Определите, в каком формате должны быть предоставлены данные. Это могут быть таблицы CSV, базы данных SQL, форматы JSON или XML, изображения и т.д.
* **Как часто должны обновляться данные?** Определите частоту обновления данных. Некоторые данные могут быть актуальными только при регулярном обновлении, например, ежедневно или еженедельно.
* **Какие ограничения по времени и ресурсам существуют?** Определите временные рамки и ресурсы, доступные для проекта. Это поможет установить реалистичные цели и планировать процесс веб-скрапинга.

### Определение ключевых метрик успеха

Ключевые метрики успеха (Key Performance Indicators, KPI) помогают оценить эффективность процесса сбора данных. Примеры таких метрик включают:

1. **Полнота данных**: доля извлеченных данных по отношению к общему объему доступных данных на веб-ресурсах.
2. **Точность данных**: степень соответствия извлеченных данных реальным данным на веб-ресурсах.
3. **Время извлечения**: время, затраченное на сбор и обработку данных.
4. **Стабильность и надежность**: частота ошибок и сбоев в процессе веб-скрапинга.
5. **Соответствие требованиям**: насколько извлеченные данные соответствуют потребностям бизнеса или исследовательского проекта.

### Критерии выбора веб-ресурсов

Выбор целевых веб-ресурсов — это важный этап, который определяет качество и релевантность данных для проекта. Правильный выбор ресурсов позволяет минимизировать затраты времени и усилий на сбор данных, а также избежать юридических и этических проблем.

### Анализ доступных ресурсов

Первым шагом является проведение анализа доступных веб-ресурсов, которые могут содержать нужные данные. Это может включать:

1. **Обзор рынка**: Изучение доступных веб-ресурсов, которые предоставляют данные по интересующей тематике. Например, для анализа рыночных трендов можно рассматривать интернет-магазины, новостные сайты, социальные сети и форумы.
2. **Исследование конкурентов**: Изучение веб-ресурсов конкурентов, чтобы понять, какие данные они публикуют и как они могут быть полезны для анализа.
3. **Использование поисковых систем**: Поиск релевантных веб-ресурсов с помощью поисковых систем, таких как Google, Bing или Yahoo.

При выборе целевых веб-ресурсов следует учитывать несколько ключевых критериев:

1. **Релевантность данных**: Веб-ресурс должен содержать данные, которые соответствуют целям проекта. Например, если цель проекта — анализ цен на товары, то веб-ресурс должен предоставлять информацию о ценах и доступности товаров.
2. **Объем и полнота данных**: Веб-ресурс должен содержать достаточное количество данных для проведения анализа. Это важно для обеспечения статистической значимости результатов.
3. **Актуальность данных**: Данные на веб-ресурсе должны быть актуальными и регулярно обновляться. Это особенно важно для анализа динамических рынков и трендов.
4. **Доступность данных**: Веб-ресурс должен быть доступен для автоматизированного извлечения данных. Некоторые сайты могут иметь технические ограничения, такие как блокировки IP-адресов или капчи, которые затрудняют процесс веб-скрапинга.
5. **Легальность и этичность**: Веб-ресурс должен позволять автоматизированное извлечение данных в соответствии с его условиями использования и местными законами. Нарушение этих условий может привести к юридическим последствиям.

### Примеры подходящих веб-ресурсов

В зависимости от целей проекта могут быть выбраны различные веб-ресурсы:

1. **Интернет-магазины**: подходят для анализа цен, ассортимента товаров, отзывов и рейтингов. Примеры: Amazon, eBay, AliExpress.
2. **Социальные сети**: подходят для анализа пользовательских мнений, трендов, популярности брендов и продуктов. Примеры: Facebook, Twitter, Instagram.
3. **Новостные сайты**: подходят для анализа новостных трендов, событий и общественного мнения. Примеры: BBC, CNN, Reuters.
4. **Форумы и блоги**: подходят для анализа пользовательских дискуссий, мнений и опыта. Примеры: Reddit, Medium, Stack Overflow.
5. **Открытые данные**: подходят для получения официальной статистики и информации. Примеры: data.gov, Европейский портал данных.

### Оценка и отбор ресурсов

После первоначального анализа и выбора потенциальных веб-ресурсов следует провести более детальную оценку каждого ресурса. Это включает:

1. **Тестирование доступности данных**: Проверка возможности автоматизированного извлечения данных с веб-ресурса. Это может включать проверку на наличие капч, блокировок IP-адресов и ограничений по частоте запросов.
2. **Анализ структуры данных**: Изучение HTML-структуры веб-страниц для определения методов извлечения данных. Это может включать использование инструментов, таких как BeautifulSoup, Scrapy или Selenium.
3. **Оценка качества данных**: Проверка точности и полноты данных, представленных на веб-ресурсе. Это может включать сравнение данных с другими источниками и проверку на наличие ошибок или неконсистентности.
4. **Соблюдение юридических и этических норм**: Проверка соответствия извлечения данных условиям использования веб-ресурса и местным законам.

Сбор требований и выбор целевых веб-ресурсов являются критически важными этапами в проекте веб-скрапинга. Определение цели сбора данных позволяет формировать четкие критерии для выбора веб-ресурсов и обеспечивает направленность всего процесса. Критерии выбора веб-ресурсов помогают выбрать наиболее релевантные, доступные и легальные источники данных, что в конечном итоге способствует достижению поставленных целей проекта.

## 2.2 Разработка архитектуры скрапера

### Определение структуры и логики скрапера

Создание скрапера — это многокомпонентный процесс, который требует четко определенной архитектуры для эффективного и корректного извлечения данных. Важно определить структуру и логику работы скрапера, чтобы он мог адаптироваться к различным типам веб-ресурсов и обеспечивать стабильное выполнение задач по извлечению данных.

#### Общая структура скрапера

Общая структура скрапера включает несколько ключевых компонентов, каждый из которых выполняет свою уникальную функцию. Ниже приведены основные компоненты и их описание:

#### Логика работы скрапера

Логика работы скрапера включает в себя последовательность шагов, которые он выполняет для извлечения данных. Ниже приведен общий алгоритм:

1. **Инициализация**:
   * Загрузка конфигурационных файлов.
   * Настройка параметров сети (например, использование прокси).
   * Загрузка библиотек и модулей.
2. **Загрузка веб-страниц**:
   * Отправка HTTP-запроса к целевому веб-ресурсу.
   * Обработка ответа (например, проверка кода состояния HTTP).
   * Повторные попытки при ошибках сети или серверных ошибках.
3. **Парсинг данных**:
   * Анализ HTML-кода с использованием регулярных выражений или библиотек для парсинга.
   * Извлечение нужной информации по заданным критериям (например, CSS-селекторы или XPath).
4. **Обработка данных**:
   * Очистка и нормализация данных (например, удаление лишних пробелов или символов).
   * Форматирование данных в соответствии с требованиями базы данных или файла.
5. **Сохранение данных**:
   * Подготовка данных для сохранения (например, преобразование в JSON или CSV).
   * Сохранение данных в базу данных (например, SQL или NoSQL) или файл.
6. **Логирование и мониторинг**:
   * Запись информации о ходе выполнения скрапера (например, успешные и неудачные попытки).
   * Мониторинг состояния скрапера в реальном времени.
   * Отправка уведомлений при возникновении ошибок или проблем.

### Потоковая обработка данных

Для повышения эффективности работы скрапера можно использовать потоковую обработку данных. Это позволяет скраперу обрабатывать данные по мере их поступления, что снижает нагрузку на память и увеличивает скорость работы. Потоковая обработка включает:

1. **Очереди задач**: Использование очередей для управления заданиями по загрузке и обработке веб-страниц.
2. **Асинхронная обработка**: Применение асинхронных библиотек, таких как asyncio, для параллельного выполнения задач.
3. **Параллелизация**: Разделение задач на несколько потоков или процессов для повышения производительности.

### Проектирование схемы базы данных

Проектирование схемы базы данных является ключевым этапом разработки скрапера. Правильная схема базы данных обеспечивает эффективное хранение, поиск и обработку данных. При проектировании схемы базы данных необходимо учитывать типы данных, которые будут извлекаться, и способы их использования.

#### Определение требований к базе данных

Перед проектированием схемы базы данных необходимо определить требования, которые она должна удовлетворять. Эти требования могут включать:

1. **Типы данных**: Определение типов данных, которые будут храниться в базе данных (например, текст, числа, даты, изображения).
2. **Объем данных**: Оценка предполагаемого объема данных для выбора подходящей структуры базы данных (например, реляционная или NoSQL база данных).
3. **Частота обновления данных**: Определение частоты обновления данных для выбора оптимальной стратегии индексации и кэширования.
4. **Требования к доступу**: Определение требований к скорости доступа и поиска данных для выбора подходящей структуры индексов и оптимизации запросов.
5. **Резервное копирование и восстановление данных**: Планирование стратегий резервного копирования и восстановления данных для обеспечения надежности и безопасности.

#### Выбор типа базы данных

В зависимости от требований проекта можно выбрать различные типы баз данных:

1. **Реляционные базы данных (SQL)**: подходят для структурированных данных с четкими связями между таблицами. Примеры: MySQL, PostgreSQL, SQLite.
2. **Документоориентированные базы данных (NoSQL)**: подходят для неструктурированных или слабо структурированных данных. Примеры: MongoDB, CouchDB.
3. **Колонноориентированные базы данных:** подходят для аналитических задач и хранения больших объемов данных. Примеры: Apache Cassandra, Google Bigtable.
4. **Графовые базы данных**: подходят для данных с сложными взаимосвязями. Примеры: Neo4j, ArangoDB.

#### Проектирование схемы данных

При проектировании схемы данных необходимо учитывать следующие аспекты:

1. **Таблицы и коллекции**: Определение основных сущностей и их атрибутов. Например, для интернет-магазина это могут быть таблицы продуктов, категорий, отзывов и т.д.
2. **Связи между сущностями**: Определение связей между таблицами (например, связи один-к-одному, один-ко-многим, многие-ко-многим). Это включает создание внешних ключей и индексов.
3. **Индексы и оптимизация запросов**: Определение индексов для ускорения поиска и обработки данных. Это может включать создание составных индексов и индексов по часто используемым атрибутам.
4. **Нормализация данных**: Разделение данных на логически связанные таблицы для минимизации избыточности и обеспечения целостности данных. Это включает применение нормальных форм (1NF, 2NF, 3NF и т.д.).
5. **Денормализация данных**: В некоторых случаях, для повышения производительности, может потребоваться денормализация данных, то есть объединение связанных данных в одну таблицу.

### Пример проектирования схемы базы данных

Рассмотрим пример проектирования схемы базы данных для интернет-магазина, который занимается продажей товаров и хранением отзывов клиентов. Схема базы данных может включать следующие таблицы:

1. **Таблица products (товары)**:

* id (INTEGER, PRIMARY KEY, UNIQUE) — Уникальный идентификатор товара.
* name (VARCHAR) — Название товара.
* description (TEXT) — Описание товара.
* price (DECIMAL) — Цена товара.
* category\_id (INTEGER) — Идентификатор категории.
* created\_at (TIMESTAMP) — Дата создания.
* updated\_at (TIMESTAMP) — Дата обновления.

1. **Таблица categories (категории)**:

* id (INTEGER, PRIMARY KEY, UNIQUE) — Уникальный идентификатор категории.
* name (VARCHAR) — Название категории.
* created\_at (TIMESTAMP) — Дата создания.
* updated\_at (TIMESTAMP) — Дата обновления.

1. **Таблица reviews (отзывы)**:

* id (INTEGER, PRIMARY KEY, UNIQUE) — Уникальный идентификатор отзыва.
* product\_id (INTEGER) — Идентификатор товара, к которому относится отзыв.
* user\_id (INTEGER) — Идентификатор пользователя, оставившего отзыв.
* rating (INTEGER) — Оценка отзыва.
* comment (TEXT) — Комментарий отзыва.
* created\_at (TIMESTAMP) — Дата создания отзыва.
* updated\_at (TIMESTAMP) — Дата обновления отзыва.

1. **Таблица users (пользователи)**:

* id (INTEGER, PRIMARY KEY, UNIQUE) — Уникальный идентификатор пользователя.
* name (VARCHAR) — Имя пользователя.
* email (VARCHAR, UNIQUE) — Электронная почта пользователя.
* password (VARCHAR) — Пароль пользователя.
* created\_at (TIMESTAMP) — Дата создания учетной записи.
* updated\_at (TIMESTAMP) — Дата обновления учетной записи.

### Нормализация данных

Для обеспечения целостности и минимизации избыточности данных, применим нормализацию:

1. **Первая нормальная форма (1NF)**: Все атрибуты таблиц должны быть атомарными, то есть не должны содержать повторяющиеся группы или множественные значения.
2. **Вторая нормальная форма (2NF)**: Все неключевые атрибуты должны зависеть от первичного ключа.
3. **Третья нормальная форма (3NF)**: Все неключевые атрибуты должны зависеть только от первичного ключа и не должны зависеть от других неключевых атрибутов.

### Денормализация данных

Для повышения производительности можно применить денормализацию, если это необходимо. Например, можно создать кэшированные таблицы для часто используемых запросов или объединить таблицы для уменьшения количества соединений.

### Индексация данных

Индексация является важным аспектом проектирования базы данных, который позволяет ускорить поиск и обработку данных. Примеры индексов:

1. **Простые индексы**: Индексы по одному атрибуту, например, индекс по name в таблице products.
2. **Составные индексы**: Индексы по нескольким атрибутам, например, индекс по category\_id и price в таблице products.
3. **Уникальные индексы**: Индексы, обеспечивающие уникальность значений, например, уникальный индекс по email в таблице users.

Разработка архитектуры скрапера включает определение структуры и логики его работы, а также проектирование схемы базы данных. Правильное проектирование этих аспектов обеспечивает эффективное извлечение, обработку и хранение данных, что в конечном итоге способствует успешной реализации проекта веб-скрапинга.

## 2.3. Настройка окружения для веб-скрапинга

Настройка окружения для веб-скрапинга является критически важным этапом, который включает установку и настройку необходимых инструментов и библиотек. Этот процесс обеспечивает корректную работу скрапера и позволяет избежать различных проблем, связанных с несовместимостью версий или отсутствием необходимых компонентов. В данном разделе будут рассмотрены основные шаги по установке и настройке инструментов, а также работа с виртуальными окружениями.

### Установка и настройка инструментов

##### Шаг 1: Установка Python

Python является основным языком программирования для веб-скрапинга благодаря своей простоте и широкому набору библиотек. Скачать Python можно с официального сайта [python.org](https://www.python.org/). Следуя инструкциям для установки Python на операционную систему (Windows, macOS, Linux).

##### Шаг 2: Установка менеджера пакетов pip

pip является стандартным менеджером пакетов для Python. Он позволяет устанавливать и управлять библиотеками и зависимостями, необходимыми для веб-скрапинга. В большинстве случаев pip уже установлен вместе с Python. Чтобы проверить это, выполняем команду:

****

Если pip не установлен, его можно установить, следуя инструкциям на официальном сайте [pip.pypa.io](https://pip.pypa.io/en/stable/installation/).

##### Шаг 3: Установка библиотек для веб-скрапинга

Для веб-скрапинга понадобятся несколько библиотек, таких как requests, BeautifulSoup, Scrapy, Selenium и другие. Ниже приведены команды для их установки:

* Установка requests:

****

Установка BeautifulSoup:



Установка Scrapy:

****

Установка Selenium:



Установка lxml для парсинга HTML:



##### Шаг 4: Установка драйверов для Selenium

Если планируется использовать Selenium для автоматизации браузера, то потребуется установить соответствующие драйверы для браузера. Например, для Chrome это будет ChromeDriver, для Firefox — GeckoDriver.

### Работа с виртуальными окружениями

Виртуальные окружения позволяют создать изолированные среды для проектов, где каждый проект может иметь свои собственные зависимости, не конфликтующие с другими проектами. Это особенно полезно при работе с различными версиями библиотек.

##### Шаг 1: Установка virtualenv

Для создания виртуальных окружений можно использовать virtualenv или встроенный в Python модуль venv. Рассмотрим установку и использование virtualenv:

Установка virtualenv:



##### Шаг 2: Создание виртуального окружения

Создаем виртуальное окружение в каталоге проекта. Это создаст изолированное окружение для проекта.

Создание виртуального окружения:



Также можно указать версию Python, которая будет использоваться в виртуальном окружении:



##### Шаг 3: Активация виртуального окружения

После создания виртуального окружения его необходимо активировать, чтобы устанавливаемые пакеты сохранялись в этом окружении.

Активация виртуального окружения (Windows):



Активация виртуального окружения (macOS и Linux):



После активации виртуального окружения в командной строке будет отображаться префикс (venv), указывающий на активное виртуальное окружение.

##### Шаг 4: Установка зависимостей в виртуальном окружении

После активации виртуального окружения устанавливаем все необходимые зависимости, используя pip:

Мы также можем создать файл requirements.txt, в котором перечислить все зависимости проекта, и затем установить их одной командой:

Создание файла requirements.txt:



Установка зависимостей из requirements.txt:



##### Шаг 5: Деактивация виртуального окружения

После завершения работы с проектом деактивируем виртуальное окружение, чтобы вернуться в глобальное окружение Python:

Деактивация виртуального окружения:



Настройка окружения для веб-скрапинга включает в себя установку и настройку необходимых инструментов и библиотек, а также работу с виртуальными окружениями для изоляции проектов и их зависимостей. Правильная настройка окружения обеспечивает стабильную и эффективную работу скрапера, а также упрощает управление зависимостями и предотвращает конфликты между различными проектами.

# Глава 3. Реализация веб-скрапинга для анализа данных в электронной коммерции.

## 3.1 Извлечение данных с помощью BeautifulSoup

BeautifulSoup — это популярная библиотека для парсинга HTML и XML документов в Python. Она позволяет легко извлекать данные из веб-страниц и обрабатывать их в удобном для вас формате.

### Установка BeautifulSoup и зависимостей

Перед тем как начать, проверяем, что установлены необходимые библиотеки:



requests используется для выполнения HTTP-запросов, а beautifulsoup4 — для парсинга HTML.

### Основы работы с BeautifulSoup

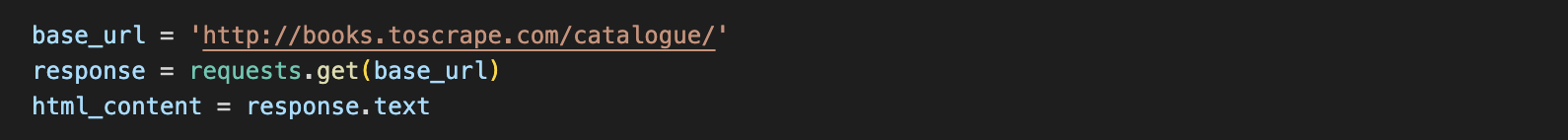
1. Импорт библиотек



Пример 4.

1. Получение HTML-кода страницы

Сначала нужно получить HTML-код страницы с помощью библиотеки requests.



Пример 4.

1. Создание объекта BeautifulSoup

Теперь создаем объект BeautifulSoup для парсинга HTML-кода.



Пример 4.

### Извлечение данных

1. Извлечение тегов

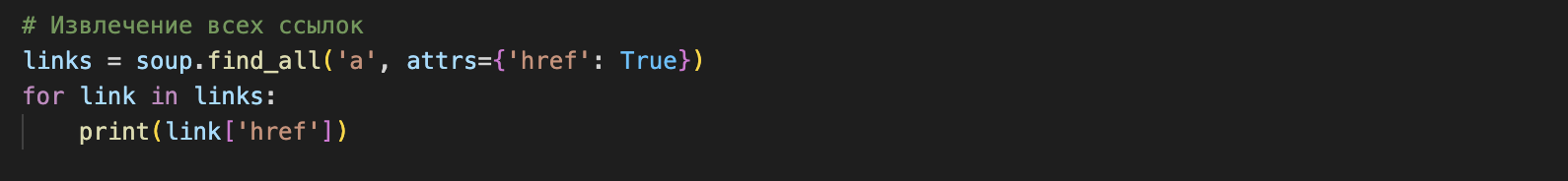
Мы можем извлечь все теги определенного типа с помощью метода find\_all.



Пример 4.

1. Извлечение по атрибутам

Для поиска элементов по атрибутам используем метод find\_all с аргументом attrs.



Пример 4.

1. Извлечение с помощью CSS-классов

Если нужно извлечь элементы с определенным классом, используем параметр class\_.



Пример 4.

### Обработка данных

После извлечения данных их можно обработать. Например, если мы извлекли заголовки статей, мы можем сохранить их в файл.



Пример 4.

### Сохранение данных

1. В файл CSV

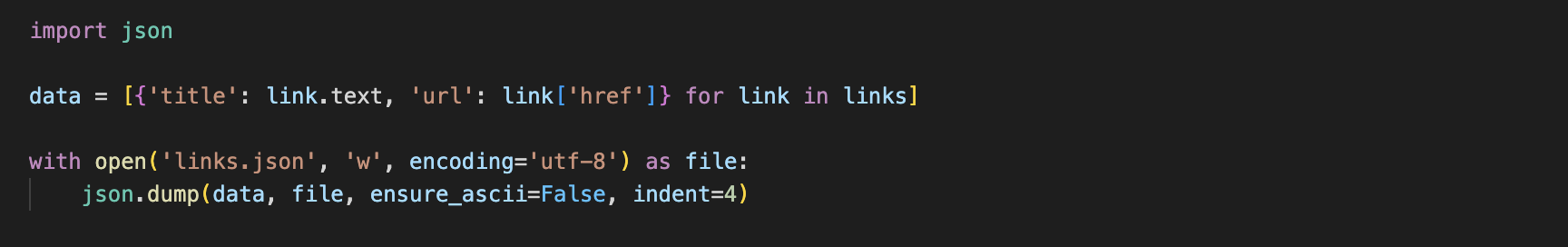
Для сохранения данных в формате CSV используем библиотеку csv.



Пример 4.

1. В файл JSON

Для сохранения данных в формате JSON используем библиотеку json.



Пример 4.

### Примеры использования

Извлечение заголовков статей и ссылок



Пример 5.

Извлечение и сохранение изображений



Пример 6.

## 3.2 Извлечение данных с помощью Scrapy

Scrapy — это фреймворк для веб-скрейпинга, который позволяет легко извлекать данные из веб-сайтов и обрабатывать их в удобном формате. В этом разделе рассматриваются основные принципы работы с Scrapy, примеры кода и методы обработки и сохранения данных.

### Основы работы с Scrapy

Scrapy предоставляет удобные инструменты для создания пауков (spiders), которые автоматически обрабатывают веб-страницы, извлекают данные и сохраняют их в различных форматах. Начнем с создания простого проекта и паука.

### Установка Scrapy

Для начала работы необходимо установить Scrapy, используя pip:



### Создание проекта Scrapy

Создаем новый проект Scrapy с помощью команды:

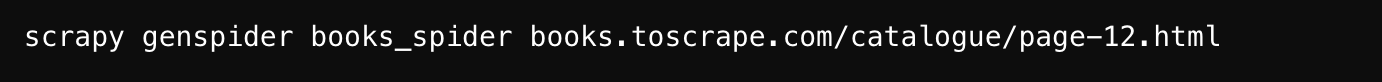


Переходим в каталог проекта:



### Создание паука

Создаем нового паука (spider) внутри каталога spiders. Например, для создания паука для сайта books.toscrape.com/catalogue/page-12.html, выполняем:

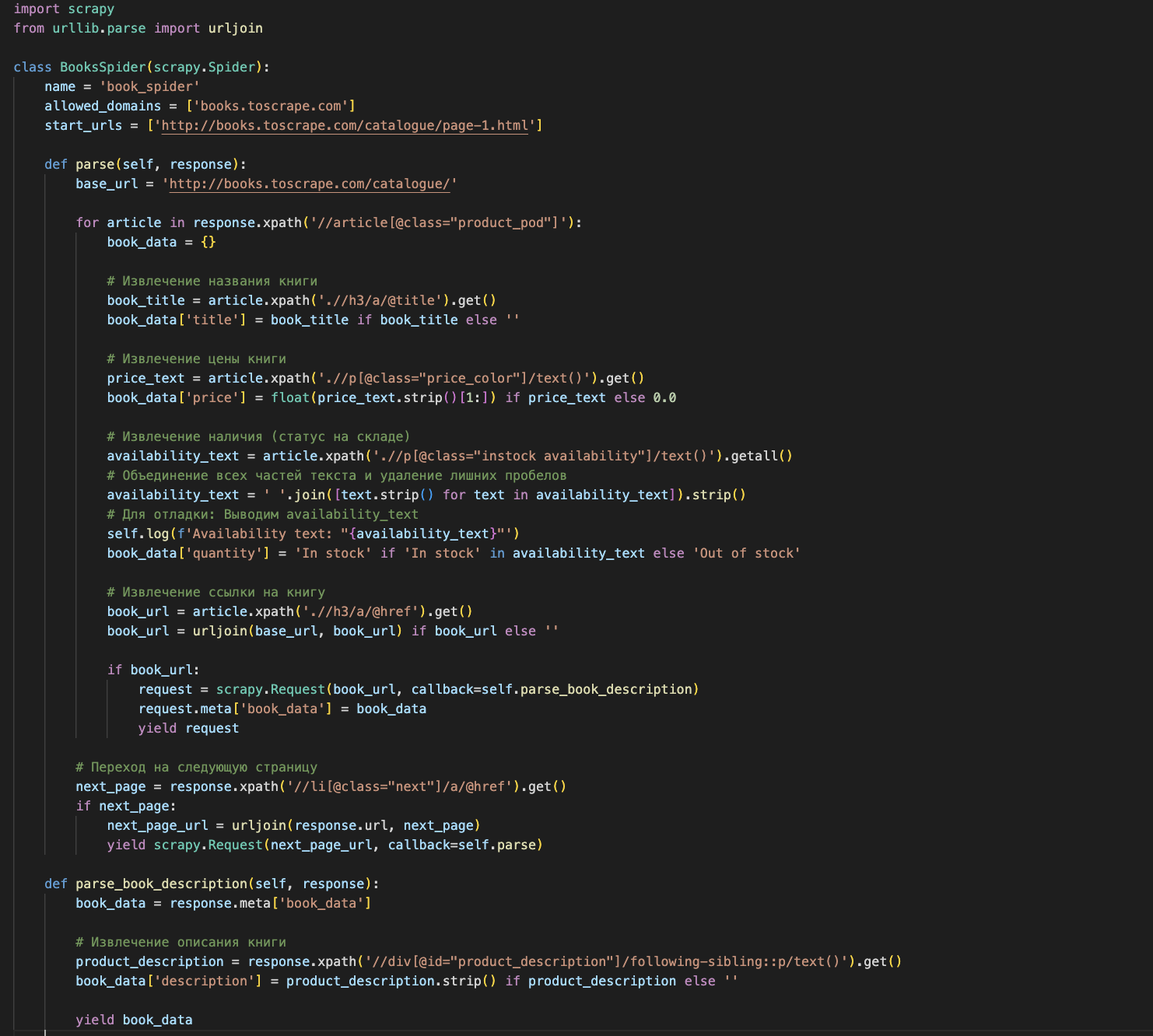


Этот код создаст файл book\_spider.py в каталоге spiders, который будет содержать начальную реализацию паука.

### Примеры кода и пояснения

#### Извлечение данных

Рассмотрим пример паука, который извлекает и собирает информацию о книгах (название, цена, наличие, ссылка на детальную страницу, описание):



Пример 7.

**Пояснение:**

* **name**: Имя паука, уникальное для проекта. В данном случае, паук называется «book\_spider».
* **allowed\_domains**: Список доменов, с которых паук может извлекать данные. В данном случае, это «books.toscrape.com».
* **start\_urls**: Начальные URL-адреса для сканирования. В данном случае, паук начинает с «http://books.toscrape.com/catalogue/page-1.html».
* **parse()**: Метод для обработки ответа. В этом методе используется XPath для:
  + Извлечения названия книги, цены, наличия на складе, и ссылки на детальную страницу каждой книги.
  + Перехода на следующую страницу каталога, если она существует.
* **parse\_book\_description()**: Метод для обработки ответа на детальной странице книги. В этом методе извлекается описание книги и сохраняется вместе с остальными данными.

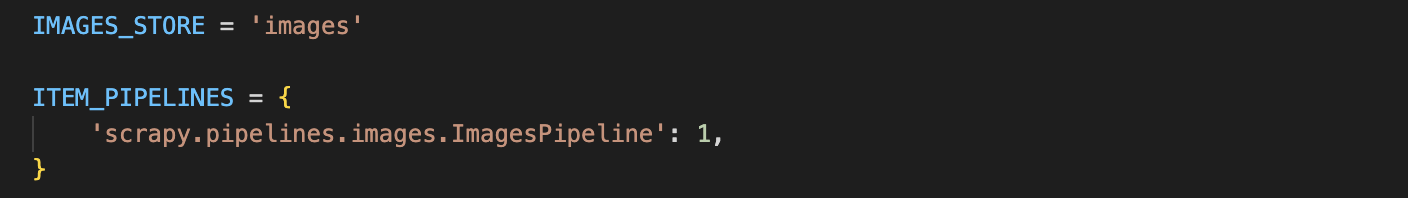
### Обработка данных

Scrapy позволяет обрабатывать данные непосредственно в процессе извлечения с помощью функций-парсеров. Мы можем использовать различные методы для обработки и форматирования данных в зависимости от требований.

### Пример обработки данных: Извлечение изображений

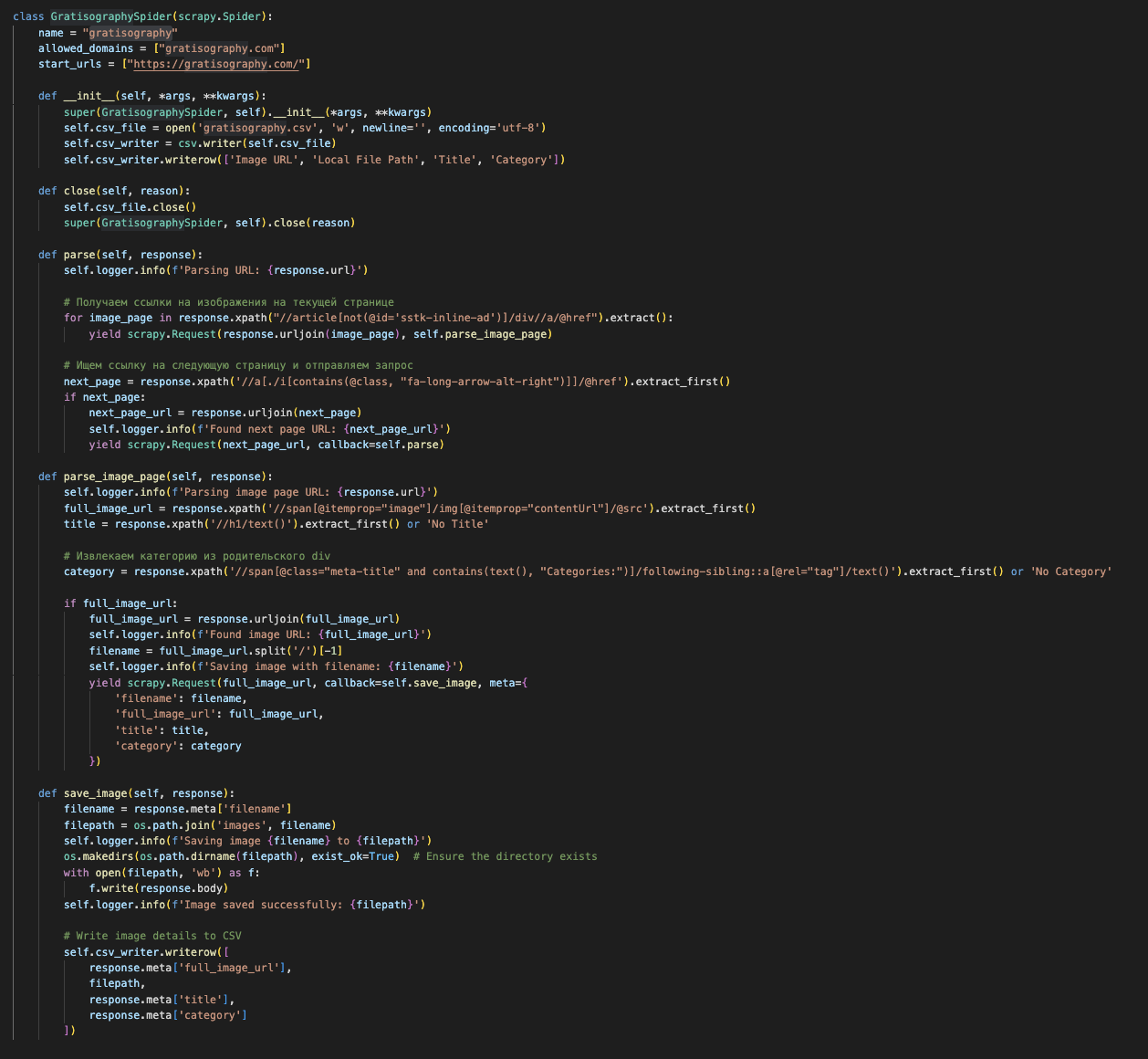
Для извлечения изображений и их сохранения на диск используем scrapy.pipelines.images.ImagesPipeline.

##### Настройки (settings.py):



Пример 8.

Паук:



Пример 8.

**Пояснение:**

В этом примере паук извлекает URL-адреса изображений и их описания. ImagesPipeline автоматически загружает и сохраняет изображения в указанной директории.

### Сохранение данных

Scrapy поддерживает различные форматы для сохранения данных, включая JSON, CSV и XML.

#### Сохранение данных в JSON

Запустите паука и сохраните результаты в файл JSON:



#### Сохранение данных в CSV

Для сохранения данных в CSV используйте следующий параметр:



**Пояснение:**

Аргумент -o указывает формат выходного файла и его имя. Scrapy автоматически преобразует данные в выбранный формат и сохраняет их в файл.

Scrapy предоставляет эффективные и гибкие инструменты для извлечения, обработки и сохранения данных с веб-сайтов. Используя примеры кода, приведенные выше, можно легко адаптировать Scrapy под ваши задачи и эффективно обрабатывать данные, извлеченные из различных источников.

## 3.3 Извлечение данных с помощью Selenium

Selenium — это инструмент для автоматизации веб-браузеров, который позволяет взаимодействовать с веб-страницами так, как это делает пользователь. Он полезен для извлечения данных с веб-страниц, особенно когда необходимо взаимодействовать с динамически загружаемым контентом или сложными элементами страницы.

### Основы работы с Selenium

Selenium поддерживает взаимодействие с различными браузерами и предоставляет API для выполнения действий на веб-страницах, таких как клики, ввод текста и извлечение данных.

#### Установка Selenium

Для начала работы с Selenium необходимо установить библиотеку и драйвер для браузера. Например, для Google Chrome используем следующий код:

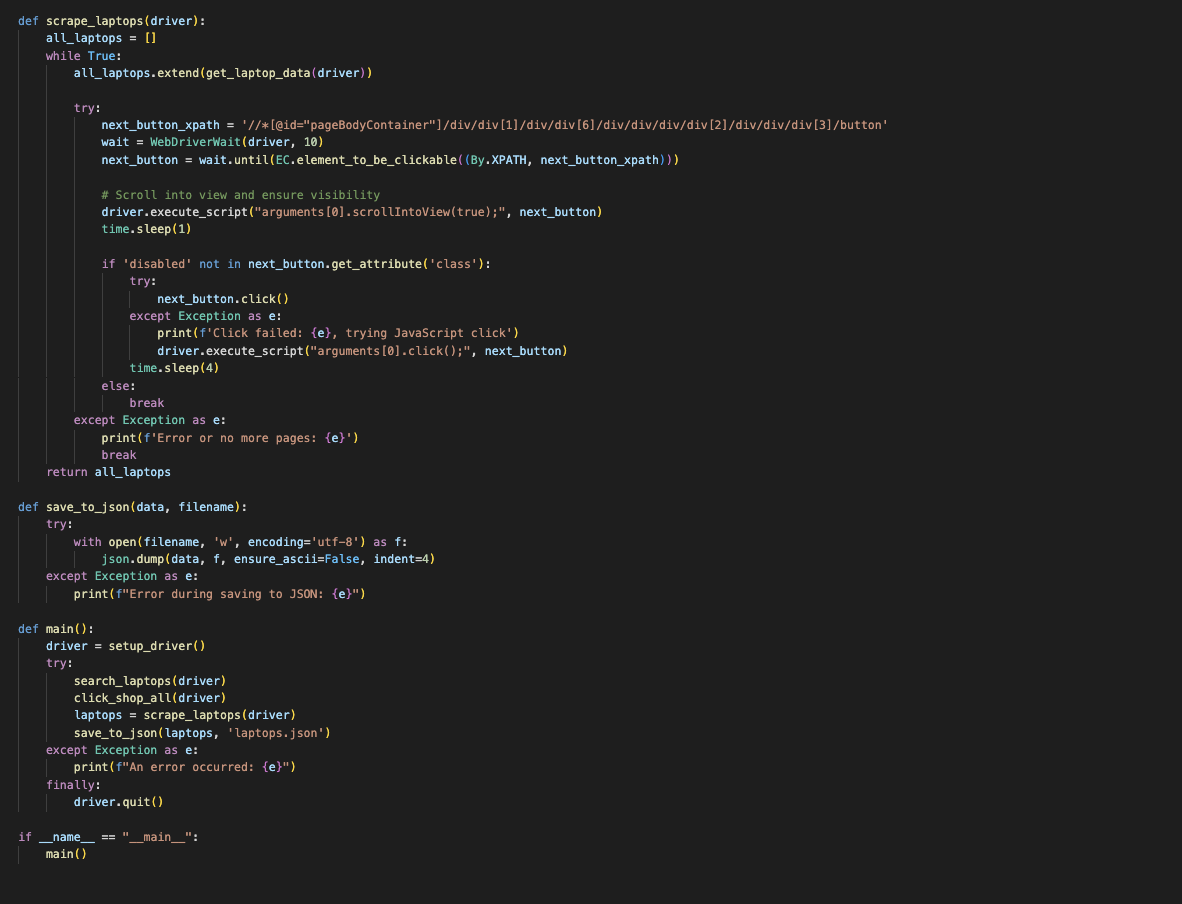


#### Пример скрипта

Рассмотрим пример скрипта на Python, который извлекает данных о ноутбуках с веб-сайта Target.



Пример 9.



Пример 9.

# Глава 4. Анализ и обработка данных

Анализ и обработка данных играют ключевую роль в любой исследовательской работе, особенно когда речь идет о больших и разнообразных данных. Эффективная предварительная обработка данных позволяет улучшить качество моделей, ускорить процесс анализа и повысить достоверность полученных результатов. В данной главе рассматриваются основные этапы предварительной обработки данных, включая очистку данных и обработку пропущенных значений.

## 4.1 Предварительная обработка данных

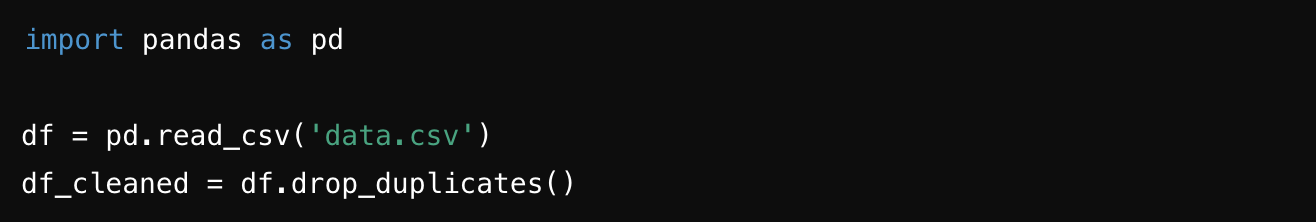
Предварительная обработка данных — это начальный этап в процессе анализа данных, на котором проводится подготовка данных для дальнейшего использования в моделях машинного обучения или статистическом анализе. Этот этап включает очистку данных, обработку пропущенных значений, нормализацию данных, преобразование категориальных переменных и другие методы подготовки данных.

### Очистка данных

Очистка данных — это процесс удаления или исправления ошибочных, дублирующихся или неактуальных данных из набора данных. Этот этап является важным, так как качество данных напрямую влияет на точность и надежность последующего анализа. Основные шаги по очистке данных включают:

1. **Удаление дубликатов:** Дублирующиеся записи могут возникать в результате ошибок при сборе данных или объединении нескольких источников данных. Эти дубликаты могут исказить результаты анализа, поэтому их необходимо удалять.

Пример на Python с использованием библиотеки pandas:



В этом примере все дублирующиеся строки будут удалены из набора данных df.

1. **Обнаружение и исправление ошибок:** Ошибки могут включать опечатки, неверные значения и другие отклонения от ожидаемых данных. Для их обнаружения и исправления используются различные методы, такие как визуализация, проверка статистических закономерностей и использование правил логики.

Например, если в столбце с возрастом обнаружены отрицательные значения, их следует скорректировать или удалить:

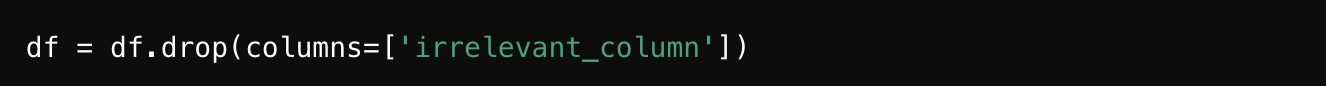


1. **Стандартизация данных**: Разные источники данных могут использовать различные форматы для представления одной и той же информации. Например, даты могут быть представлены в различных форматах (DD/MM/YYYY, MM-DD-YYYY и т.д.). Стандартизация этих данных позволяет избежать ошибок и несоответствий.

Пример стандартизации даты:



1. **Удаление нерелевантных данных:** Некоторые переменные или записи могут быть неактуальными для текущего анализа. Удаление таких данных позволяет сократить объем обрабатываемой информации и сфокусироваться на наиболее значимых характеристиках.



### Обработка пропущенных значений

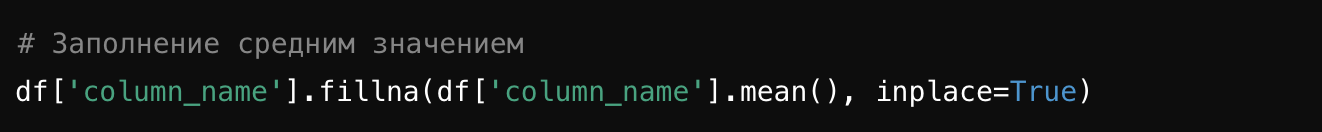
Пропущенные значения представляют собой одну из наиболее распространенных проблем в анализе данных. Игнорирование или неправильное обращение с пропущенными данными может привести к серьезным искажениям в результатах анализа. Основные стратегии обработки пропущенных данных включают:

1. **Удаление пропущенных данных**: если пропущенные значения составляют небольшой процент от общего объема данных, возможно целесообразным будет удаление этих записей или столбцов.



В этом примере будут удалены все строки, содержащие хотя бы одно пропущенное значение.

1. **Заполнение пропущенных данных**: в некоторых случаях лучше не удалять данные, а заполнить пропущенные значения. Это можно сделать с помощью различных методов, таких как заполнение средним, медианой, модой или с использованием более сложных моделей.



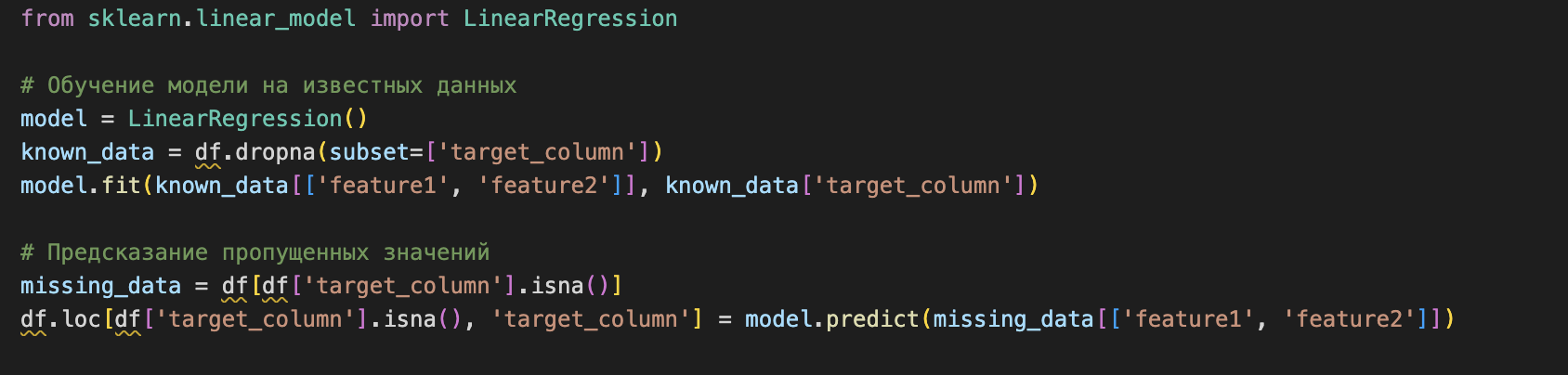
В этом примере пропущенные значения в столбце column\_name будут заменены средним значением данного столбца.

1. **Интерполяция**: Интерполяция — это метод заполнения пропущенных значений на основе существующих данных. Он может быть особенно полезен для временных рядов, где пропущенные данные могут быть восстановлены на основе предыдущих и последующих значений.



Интерполяция использует линейный метод по умолчанию, но можно применять и другие методы, такие как полиномиальная или сплайновая интерполяция.

1. **Использование предсказательных моделей**: в некоторых случаях для заполнения пропущенных значений могут быть использованы модели машинного обучения. Например, можно обучить модель на известных данных и использовать ее для предсказания пропущенных значений.



Пример 3.

Этот подход позволяет использовать всю доступную информацию для более точного заполнения пропущенных значений.

Предварительная обработка данных является важнейшим этапом подготовки данных к анализу. Очистка данных и обработка пропущенных значений обеспечивают высокое качество данных и, как следствие, точность и надежность последующего анализа. В следующем разделе будет рассмотрен процесс нормализации данных и преобразования категориальных переменных, что также играет важную роль в подготовке данных для машинного обучения.

## 4.2 Анализ данных с использованием pandas и scikit-learn

В данном разделе будет представлен процесс анализа данных, полученных в результате предварительной обработки. Основное внимание будет уделено статистическому анализу и применению методов машинного обучения для выявления закономерностей и получения значимых выводов.

После предварительной обработки данных следующим шагом является проведение статистического анализа. Этот процесс включает в себя описание основных статистических характеристик данных, таких как среднее значение, медиана, стандартное отклонение и другие метрики. Использование библиотеки pandas позволяет эффективно выполнять эти операции.

Например, с помощью pandas можно рассчитать основные статистики для числовых признаков следующим образом:



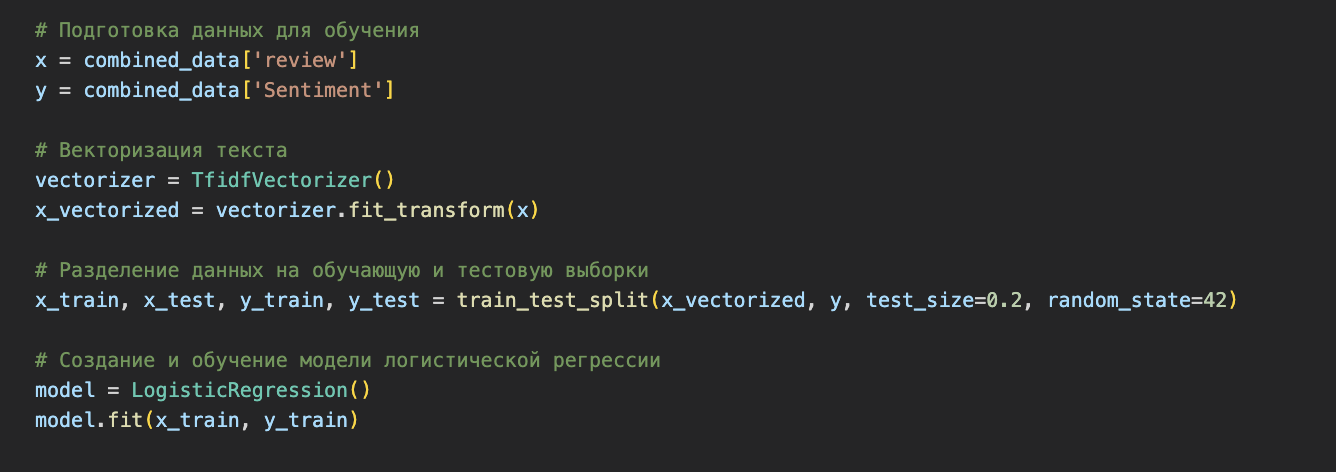
Пример 10.

Метод describe() предоставляет информацию о распределении значений в столбцах, включая среднее значение, стандартное отклонение, минимальные и максимальные значения. Это позволяет получить представление о том, как распределены данные и выявить возможные аномалии или выбросы.

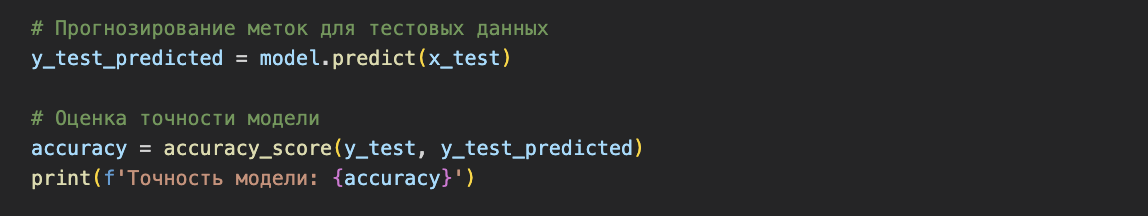
### ****Применение методов машинного обучения****

После анализа данных и их предварительной обработки можно переходить к применению методов машинного обучения. В scikit-learn представлены различные алгоритмы, которые можно использовать для решения задач классификации, регрессии и кластеризации.

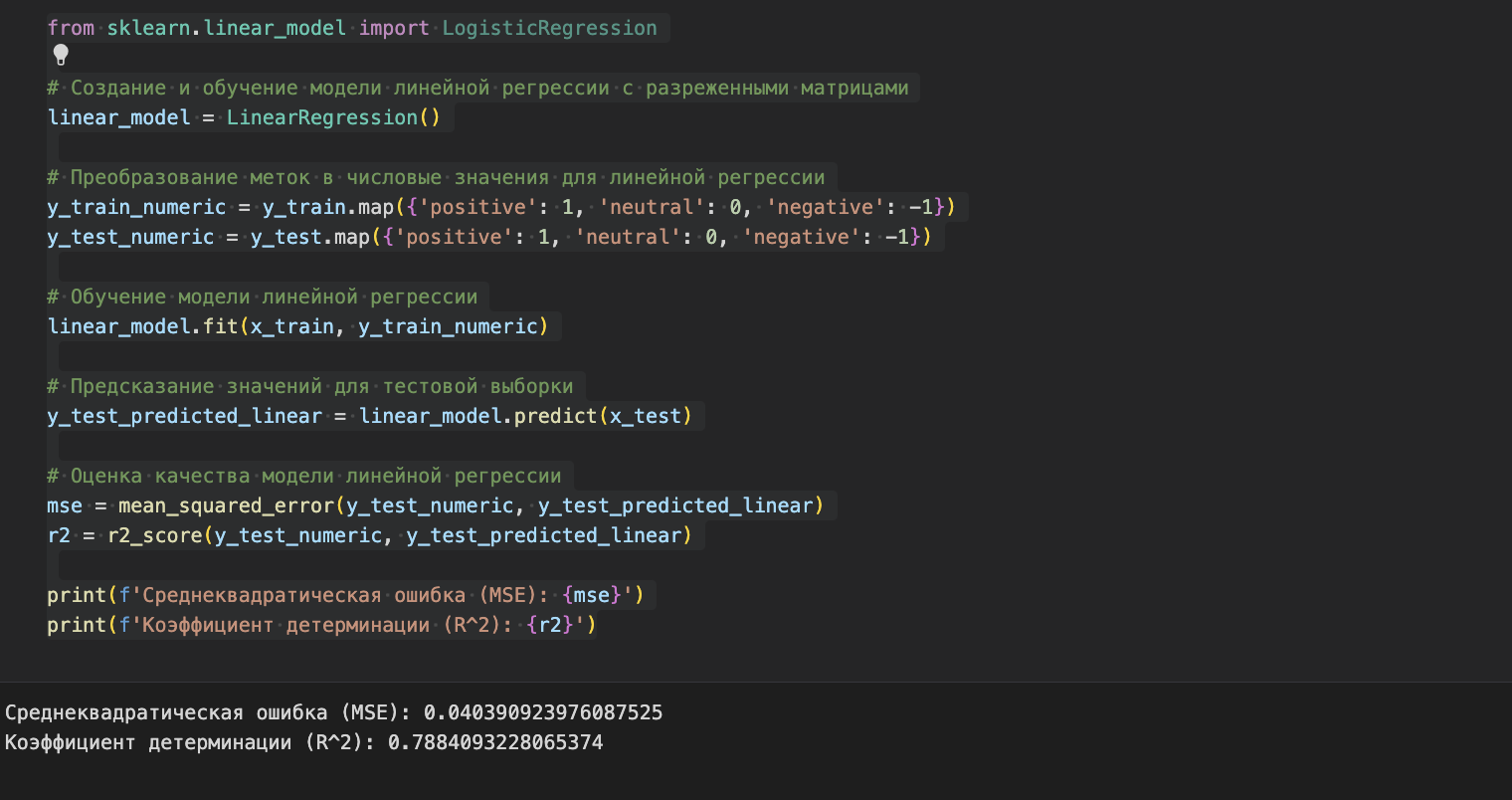
Для начала, необходимо разделить данные на обучающую и тестовую выборки. Это позволяет оценить производительность модели на независимых данных. Пример кода для разделения данных и обучения модели логистической регрессии:



Пример 3.



Пример 3.



Пример 3.

В этом примере используется логистическая регрессия для классификации. Оценка модели производится с помощью точности (accuracy), которая показывает долю правильных предсказаний.

## 4.3 Визуализация данных

### ****Инструменты для визуализации****

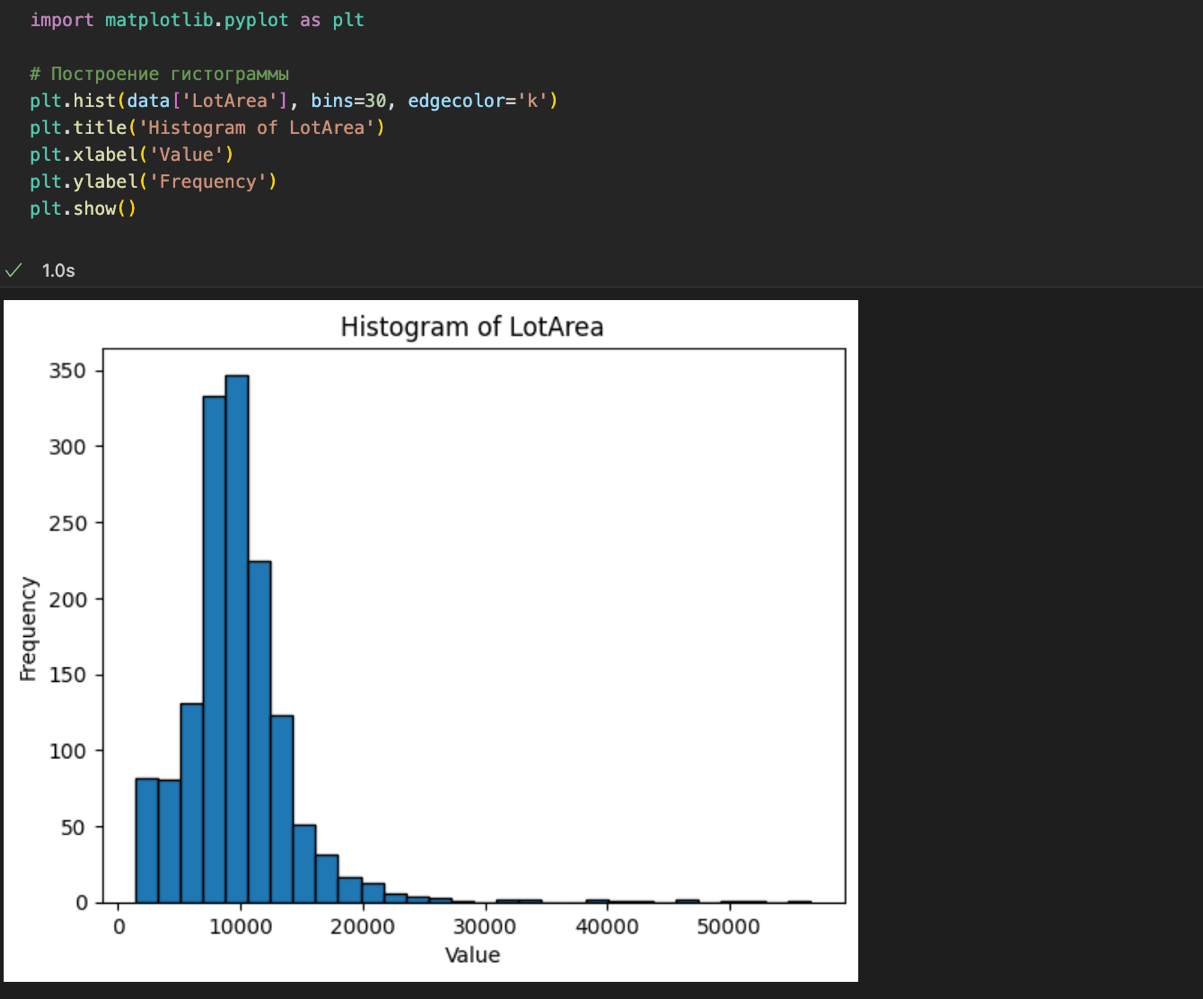
Визуализация данных играет ключевую роль в анализе данных, так как позволяет легко интерпретировать результаты и выявлять закономерности. В Python для визуализации данных можно использовать библиотеки Matplotlib и Seaborn.

* **Matplotlib** – это базовая библиотека для создания графиков и диаграмм. Она предоставляет широкий спектр возможностей для настройки внешнего вида графиков.
* **Seaborn** – это библиотека, построенная на основе Matplotlib, которая упрощает создание более сложных визуализаций и предоставляет дополнительные возможности для анализа данных.

#### ****Примеры визуализации****

1. **Гистограммы и распределения**

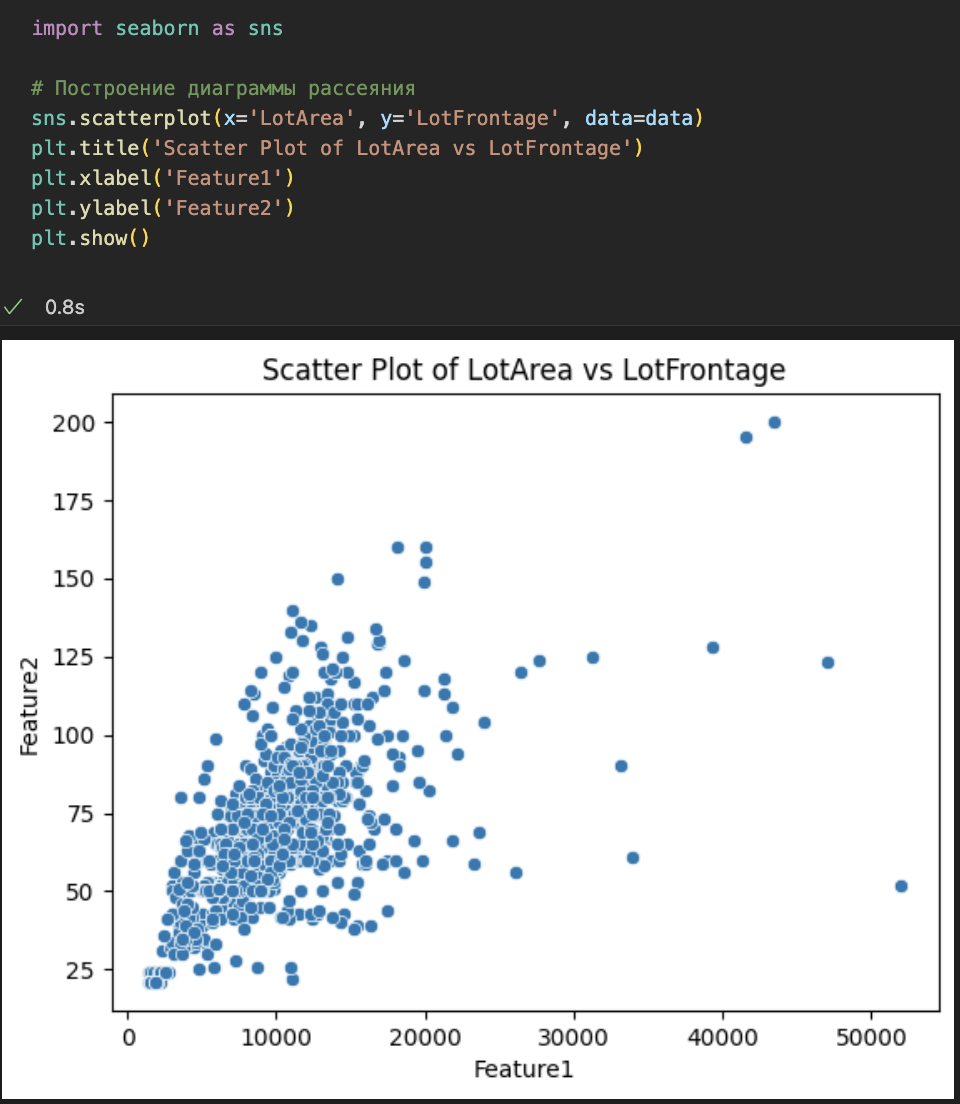
Гистограммы позволяют оценить распределение данных по интервалам. Пример использования библиотеки Matplotlib для построения гистограммы:

****

**Приложение 10.**

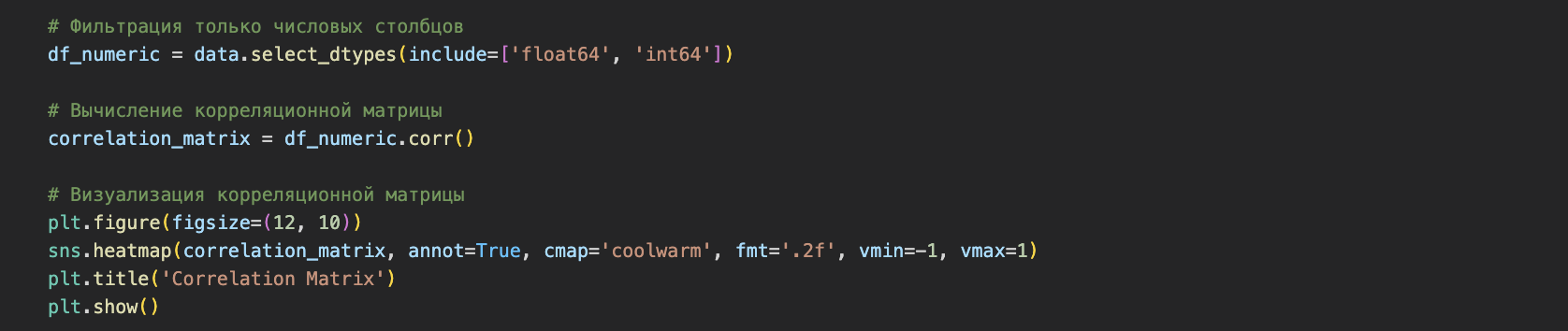
1. **Диаграммы рассеяния**

Диаграммы рассеяния помогают визуализировать зависимость между двумя переменными. Пример с использованием Seaborn:



**Приложение 10.**

1. **Корреляционные матрицы**

Корреляционные матрицы позволяют увидеть, как сильно связаны между собой разные признаки. Пример визуализации с использованием Seaborn: 

**Приложение 10.**

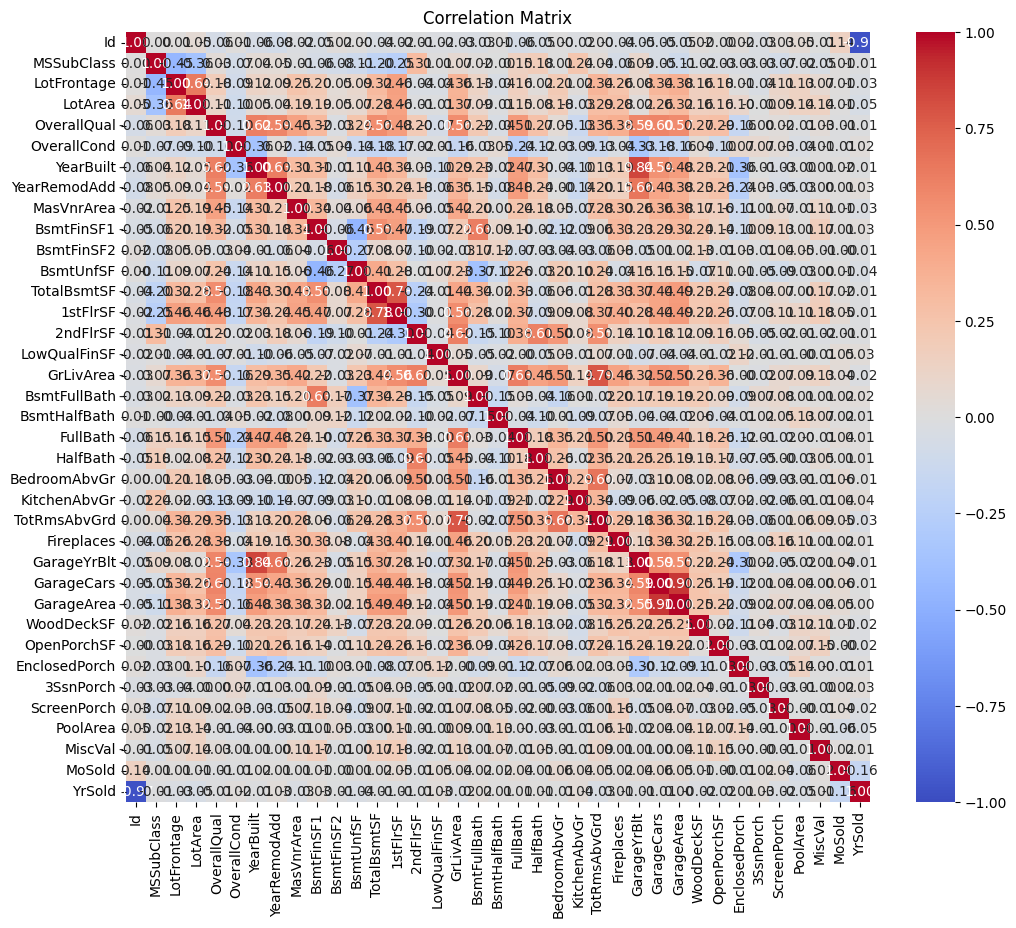


Рисунок 6.

Эти методы визуализации помогают глубже понять структуру данных и выявить важные взаимосвязи между переменными.

# Глава 5. Применение результатов анализа для принятия бизнес-решений

## 5.1 Прогнозирование спроса и оптимизация запасов

Прогнозирование спроса и оптимизация запасов — важные задачи, стоящие перед предприятиями, особенно в условиях конкурентного рынка и неопределенности спроса. Эти задачи можно решать с использованием моделей машинного обучения, которые позволяют анализировать исторические данные, выявлять паттерны и тренды, и на основе этого делать прогнозы на будущее.

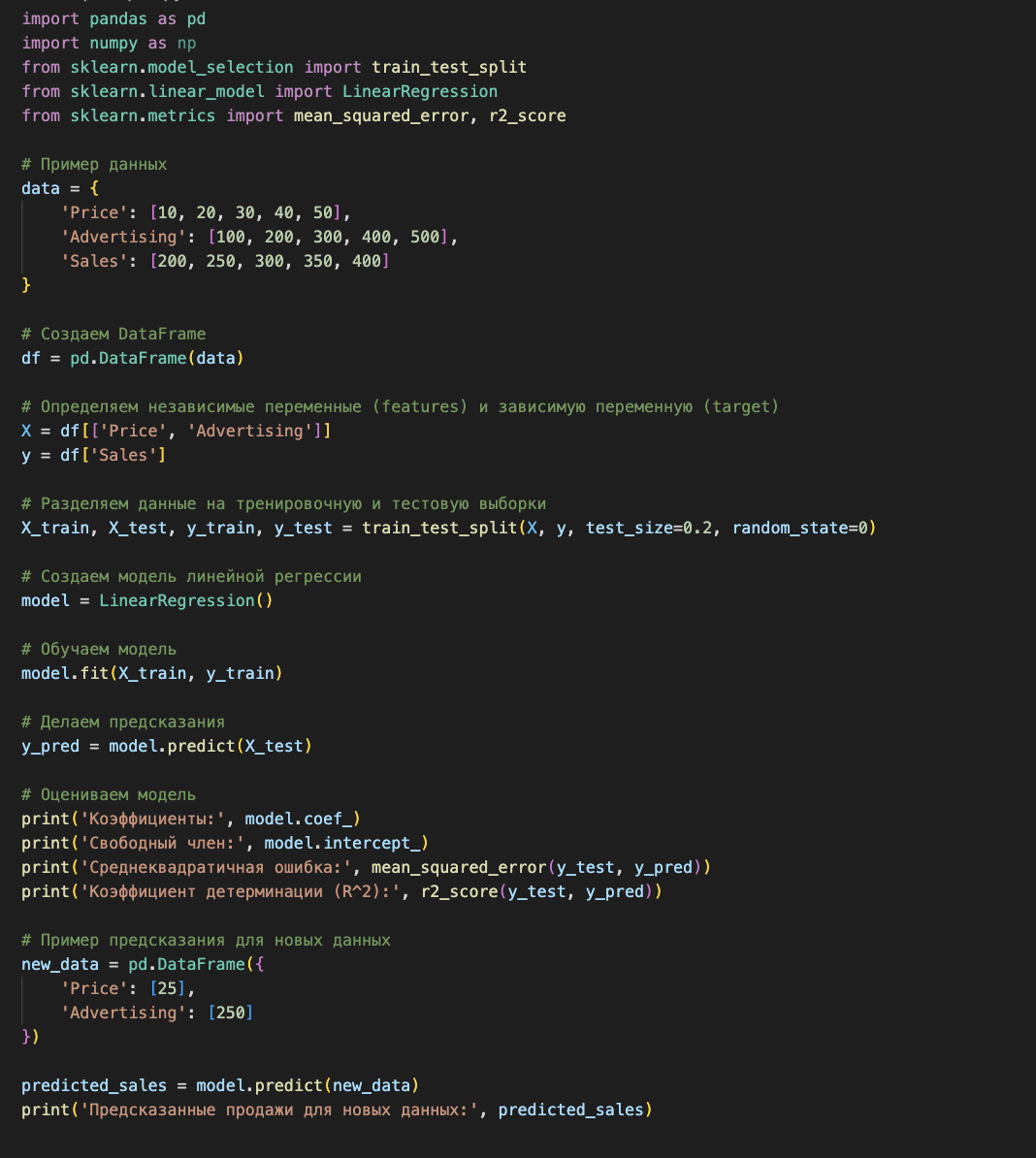
### Применение моделей машинного обучения

Для прогнозирования спроса на продукцию можно использовать различные модели машинного обучения, такие как линейная регрессия, модели временных рядов (например, ARIMA или Prophet), случайные леса и градиентный бустинг. Эти модели позволяют учитывать множество факторов, влияющих на спрос, включая сезонные колебания, тренды, акции, праздники и другие внешние переменные.

### Линейная регрессия

Линейная регрессия — это простая, но мощная модель, которая может быть использована для прогнозирования спроса. Она предполагает линейную зависимость между целевой переменной (например, объемом продаж) и независимыми переменными (например, ценой, рекламными затратами, сезоном).

Пример использования линейной регрессии:



Пример 11.

**Пояснение:**

* **LinearRegression():** Модель линейной регрессии.
* **mean\_squared\_error():** Среднеквадратичная ошибка, которая измеряет среднее отклонение прогнозируемых значений от фактических.

### Модели временных рядов

Модели временных рядов используются для анализа данных, собранных в хронологическом порядке. Эти модели помогают выявлять тренды, сезонные колебания и другие временные зависимости в данных.

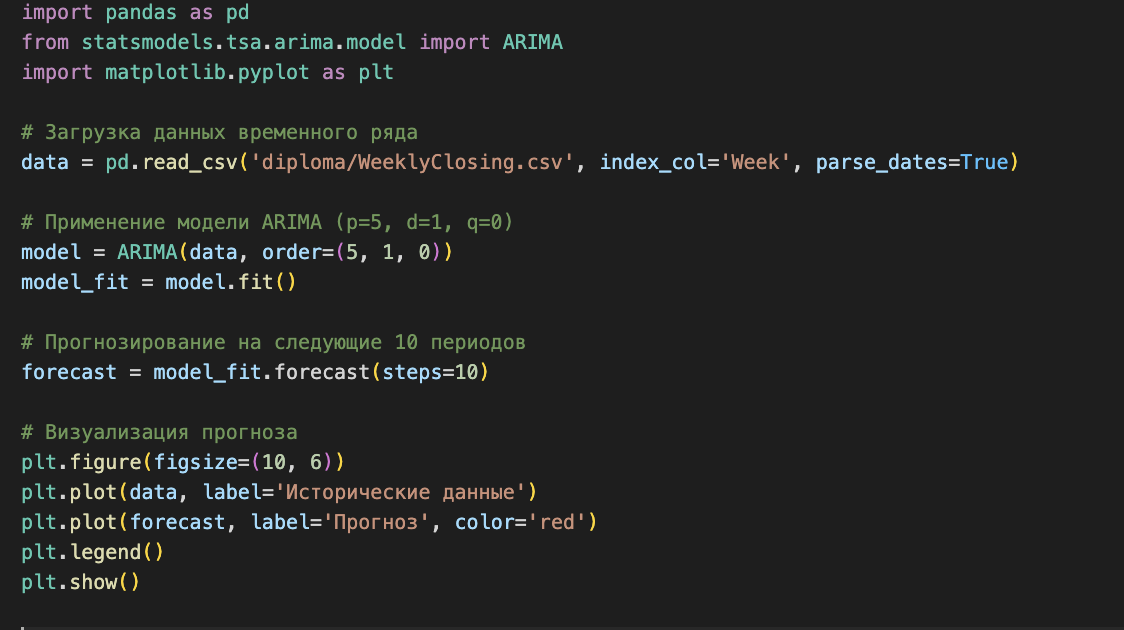
Одной из наиболее популярных моделей для анализа временных рядов является **ARIMA** (AutoRegressive Integrated Moving Average). Эта модель учитывает автокорреляцию, интеграцию (преобразование данных для их стационарности) и скользящее среднее.

### Пример использования ARIMA

ARIMA моделируется с использованием трех параметров: (p, d, q), где:

* **p** — порядок автокорреляции (AR),
* **d** — порядок интеграции (I), указывающий, сколько раз необходимо дифференцировать данные для их стационарности,
* **q** — порядок скользящего среднего (MA).

Пример использования ARIMA для прогнозирования временного ряда:



Пример 12.

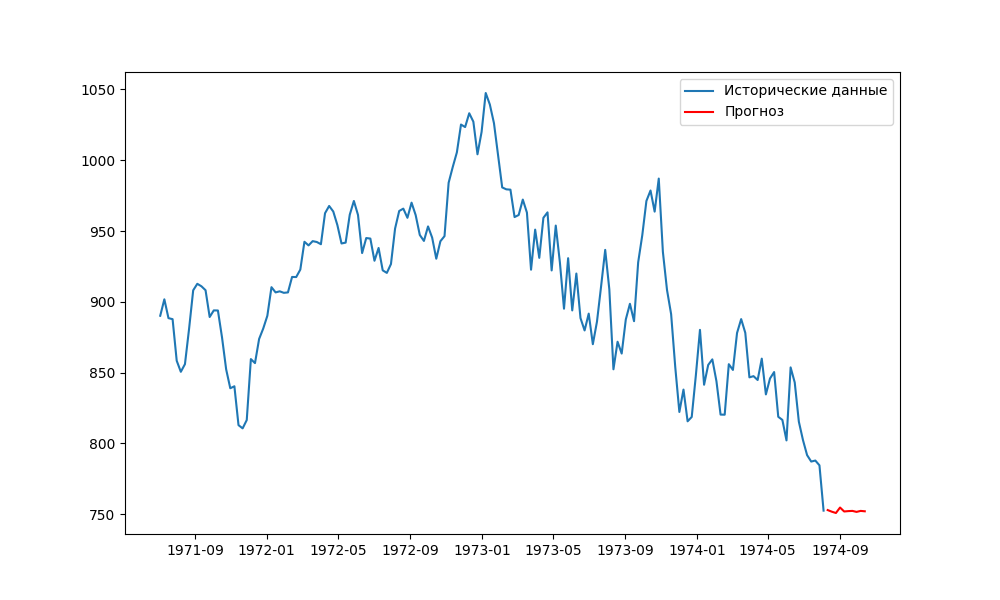


Рисунок 7.

### Интерпретация результатов

Интерпретация результатов моделей машинного обучения в контексте прогнозирования спроса и оптимизации запасов играет ключевую роль. Важно не только получить прогнозы, но и понять, какие факторы оказывают наибольшее влияние на спрос и как эти прогнозы могут быть использованы для принятия решений.

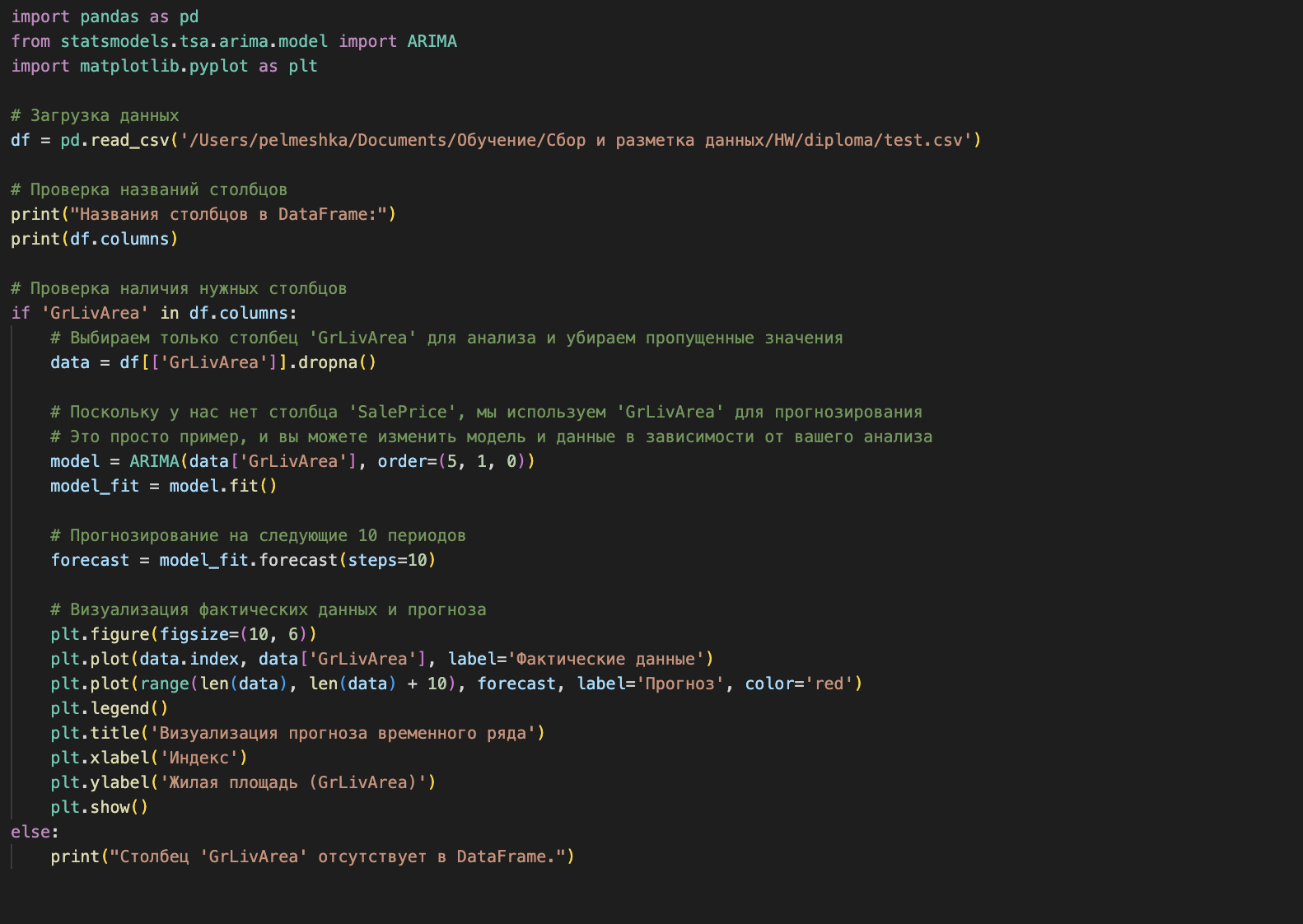
### Интерпретация коэффициентов регрессии

В случае использования линейной регрессии важно проанализировать коэффициенты модели, чтобы понять, как изменение каждой независимой переменной влияет на объем продаж. Например, если коэффициент при цене отрицательный, это означает, что увеличение цены приводит к снижению спроса, что логично и соответствует ожиданиям.

### Визуализация прогнозов

Визуализация прогнозов помогает лучше понять, насколько хорошо модель предсказывает спрос. Это может быть сделано с помощью графиков, на которых показаны фактические данные и прогнозы модели.

Пример визуализации прогнозов:



Пример 13.



Рисунок 8.

### Оптимизация запасов

На основе прогнозов спроса можно оптимизировать запасы, чтобы минимизировать издержки, связанные с хранением и отсутствием товара. Оптимизация запасов включает определение оптимального уровня запасов, заказа и размера партии.

Для этого могут использоваться методы оптимизации, такие как экономический размер заказа (EOQ) или модели, учитывающие риск и неопределенность спроса.

Применение моделей машинного обучения для прогнозирования спроса и оптимизации запасов предоставляет предприятиям современные инструменты для повышения эффективности и снижения издержек. Интерпретация результатов и их использование для принятия обоснованных решений являются ключевыми этапами в этом процессе.

## 5.2 Анализ конкурентной среды

Анализ конкурентной среды является важным элементом стратегического планирования и позволяет предприятиям оценить свои позиции на рынке относительно конкурентов. В этом разделе будут рассмотрены методы сравнительного анализа данных, а также процесс разработки стратегий на основе полученных результатов.

### Сравнительный анализ данных

Сравнительный анализ данных (бенчмаркинг) предполагает сравнение ключевых показателей деятельности компании с аналогичными показателями конкурентов. Это позволяет выявить сильные и слабые стороны, определить возможности для улучшения и определить направления развития.

### Сбор и обработка данных

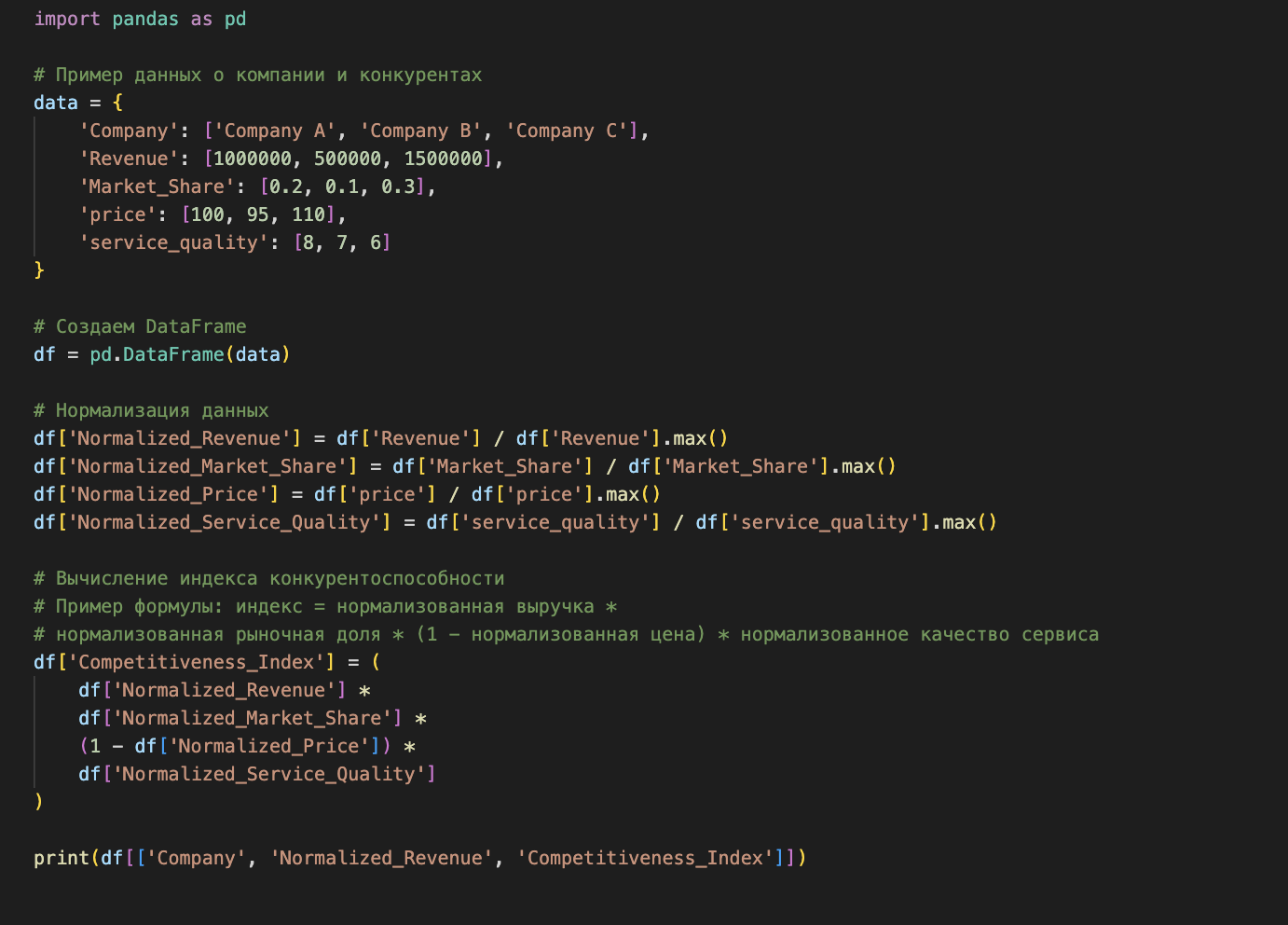
Для проведения сравнительного анализа необходимо собрать данные о конкурентах, такие как:

* Финансовые показатели (выручка, прибыль, маржинальность)
* Доля рынка
* Ценовая политика
* Ассортимент продукции
* Уровень сервиса

Эти данные могут быть получены из различных источников, включая открытые базы данных, отчеты аналитических агентств, исследования рынка и собственные наблюдения.

После сбора данных их необходимо структурировать и обработать, чтобы они были сопоставимы. Это может включать нормализацию данных, удаление выбросов и пропущенных значений, а также расчет дополнительных показателей, таких как индекс конкурентоспособности.

Пример обработки данных:



Пример 14.

**Пояснение:**

* data: Данные о компании и конкурентах.
* Normalized\_Revenue: Нормализованный показатель выручки, позволяющий сравнивать компании разного размера.
* Competitiveness\_Index: Индекс конкурентоспособности, который учитывает несколько показателей.

### Анализ позиций на рынке

Анализ конкурентных позиций может включать такие методы, как SWOT-анализ (анализ сильных и слабых сторон, возможностей и угроз) и PEST-анализ (анализ политических, экономических, социальных и технологических факторов). Эти методы позволяют определить внешние и внутренние факторы, влияющие на конкурентоспособность компании.

Пример SWOT-анализа:

|  |  |
| --- | --- |
| **Сильные стороны** | **Слабые стороны** |
| Высокое качество продукции | Высокие издержки |
| Лояльная клиентская база | Ограниченный ассортимент |

|  |  |
| --- | --- |
| **Возможности** | **Угрозы** |
| Выход на новые рынки | Усиление конкуренции |
| Введение новых технологий | Изменение потребительских предпочтений |

### Разработка стратегий на основе полученных данных

На основе проведенного сравнительного анализа и анализа конкурентных позиций разрабатываются стратегии, направленные на усиление конкурентоспособности компании. Эти стратегии могут включать:

1. **Дифференциация продукции:** Разработка уникальных предложений, которые выделят компанию на фоне конкурентов. Это может включать улучшение качества, добавление новых функций, улучшение дизайна или расширение ассортимента.
2. **Ценовая стратегия:** В зависимости от позиций компании на рынке, можно выбрать стратегию ценового лидерства или премиум-ценообразования. Первая предполагает установление конкурентоспособных цен для привлечения массового потребителя, вторая — высокие цены, соответствующие уникальности предложения.
3. **Улучшение клиентского сервиса:** Ориентация на потребности клиента и улучшение сервиса может стать важным фактором для удержания существующих клиентов и привлечения новых.
4. **Инновации и технологии:** Внедрение новых технологий и инноваций позволяет компании оставаться впереди конкурентов. Это может включать автоматизацию процессов, использование искусственного интеллекта, улучшение онлайн-платформы и другие технологические решения.

#### Пример разработки стратегии

Допустим, компания A выявила, что её основные конкуренты предлагают продукцию по более низким ценам, но не обладают такой лояльной клиентской базой. В этом случае можно предложить следующие стратегии:

* **Ценовая стратегия:** введение специальных предложений и скидок для текущих клиентов, чтобы сохранить их лояльность, а также разработка гибкой системы ценообразования, которая позволит предложить конкурентоспособные цены новым клиентам.
* **Улучшение сервиса:** введение программы обучения персонала, направленной на улучшение качества обслуживания, и внедрение системы оценки клиентского опыта для постоянного мониторинга уровня удовлетворенности клиентов.
* **Инновации:** разработка нового функционала для продукции, который выделит компанию среди конкурентов, или внедрение онлайн-сервисов для упрощения взаимодействия с клиентами.

Анализ конкурентной среды и разработка стратегий на его основе являются ключевыми этапами в процессе стратегического планирования. Сравнительный анализ данных позволяет объективно оценить позиции компании на рынке, а грамотная разработка стратегий — усилить её конкурентоспособность и создать условия для устойчивого роста в долгосрочной перспективе.

## 5.3 Рекомендации по улучшению бизнес-процессов

Эффективность работы компании во многом зависит от оптимизации бизнес-процессов. Улучшение бизнес-процессов направлено на повышение производительности, сокращение затрат и улучшение качества продукции или услуг. В этом разделе будут рассмотрены рекомендации по внедрению предложений на основе анализа данных, а также методы оценки эффективности внедренных изменений.

### Внедрение предложений на основе анализа данных

Рекомендации по улучшению бизнес-процессов должны основываться на детальном анализе данных, полученных в ходе предыдущих этапов исследования. Анализ данных позволяет выявить слабые места и узкие места в существующих процессах, а также определить направления для их оптимизации. Основные шаги по внедрению предложений включают:

1. **Анализ текущих бизнес-процессов:** Оценка существующих процессов с точки зрения их эффективности, затрат и времени выполнения. Это может включать использование таких инструментов, как диаграммы процессов, карты ценности и анализ временных затрат.
2. **Определение целей и показателей эффективности:** установление четких целей для улучшения процессов и определение ключевых показателей эффективности (KPI), которые будут использоваться для оценки результатов.
3. **Разработка и тестирование решений:** на основе анализа данных разрабатываются конкретные решения для улучшения процессов. Перед широким внедрением рекомендуется провести пилотное тестирование, чтобы оценить влияние изменений в небольшом масштабе.
4. **Внедрение изменений:** после успешного тестирования решения внедряются в масштабах всей компании. Это может потребовать обучения сотрудников, перенастройки систем и процессов, а также внесения изменений в организационную структуру.
5. **Мониторинг и корректировка:** важно постоянно отслеживать результаты внедренных изменений и при необходимости вносить корректировки. Это позволяет быстро реагировать на возможные проблемы и оптимизировать процессы на основе полученных данных.

### Пример внедрения предложений

Предположим, компания выявила, что процесс обработки заказов занимает больше времени, чем у конкурентов. В результате анализа данных были предложены следующие улучшения:

* Внедрение автоматизированной системы управления заказами, которая сокращает время обработки на 30%.
* Оптимизация логистики с использованием анализа данных о транспортировке, что позволяет сократить время доставки на 20%.
* Улучшение системы уведомлений клиентов для повышения удовлетворенности и снижения числа обращений в службу поддержки.

### Оценка эффективности внедренных изменений

После внедрения изменений необходимо оценить их эффективность. Это можно сделать с использованием следующих методов:

1. **Сравнительный анализ до и после изменений:** Оценка ключевых показателей эффективности (KPI) до и после внедрения изменений. Например, сравнение среднего времени обработки заказа до внедрения новой системы и после.
2. **Обратная связь от сотрудников и клиентов:** Сбор мнений и предложений от сотрудников, участвующих в процессе, а также от клиентов, которые непосредственно сталкиваются с изменениями. Это помогает выявить возможные скрытые проблемы или дополнительные возможности для улучшения.
3. **Анализ затрат и экономии:** Оценка того, насколько изменения позволили сократить затраты и увеличить прибыль. Например, расчет возврата инвестиций (ROI) от внедрения новой технологии.
4. **Мониторинг в реальном времени:** Использование систем мониторинга для отслеживания эффективности процессов в реальном времени. Это позволяет оперативно реагировать на изменения и поддерживать высокий уровень эффективности.

### Пример оценки эффективности

После внедрения автоматизированной системы управления заказами в компании был проведен анализ результатов:

* Среднее время обработки заказа сократилось с 24 до 16 часов.
* Уровень удовлетворенности клиентов вырос на 15% за счет быстрого получения информации о заказе.
* Снижение затрат на обработку заказов на 10% благодаря автоматизации и сокращению ручного труда.

Внедрение изменений на основе анализа данных и регулярная оценка их эффективности — это ключевые элементы для обеспечения устойчивого роста и конкурентоспособности компании. Постоянное совершенствование бизнес-процессов позволяет компании адаптироваться к изменениям на рынке, удовлетворять потребности клиентов и повышать свою эффективность.

# Заключение

### Выводы по проделанной работе

В ходе выполнения данной дипломной работы были достигнуты важные результаты в области анализа данных и их использования для улучшения бизнес-процессов. Основное внимание было уделено изучению методов сбора, обработки и анализа данных с использованием современных инструментов, таких как BeautifulSoup, Scrapy и Selenium. Эти технологии были выбраны не случайно, так как каждая из них предоставляет уникальные возможности для эффективного извлечения и анализа данных из веб-источников.

В первой части работы был проведен анализ методов извлечения данных с использованием библиотеки BeautifulSoup. Были рассмотрены основные принципы работы с HTML-документами, примеры кода и подходы к обработке и сохранению данных. BeautifulSoup зарекомендовала себя как мощный инструмент для парсинга веб-страниц, особенно в тех случаях, когда требуется точное извлечение информации из сложных HTML-структур.

Во второй части работы основное внимание было уделено фреймворку Scrapy, который предоставляет более комплексный подход к веб-скрапингу. Примеры кода показали, как можно эффективно использовать Scrapy для массового сбора данных, обработки и их сохранения. Scrapy особенно полезен в ситуациях, требующих работы с большими объемами данных и автоматизации процесса сбора информации.

В третьей части работы были рассмотрены возможности Selenium для автоматизации взаимодействия с веб-страницами. Этот инструмент оказался незаменимым для работы с динамическими веб-страницами, которые требуют выполнения скриптов JavaScript для полной загрузки контента. Примеры кода и пояснения продемонстрировали, как Selenium позволяет эффективно управлять браузером, автоматически извлекая необходимую информацию.

Таким образом, в ходе работы была выполнена всесторонняя оценка различных методов извлечения данных из веб-источников, были выявлены их сильные и слабые стороны, а также рассмотрены возможные сценарии их использования.

### Достижение поставленных целей

Основной целью дипломной работы было изучение современных методов извлечения данных из веб-источников, их анализа и использования для улучшения бизнес-процессов. В ходе выполнения работы все поставленные цели были достигнуты.

1. **Исследование инструментов для извлечения данных:** были изучены и продемонстрированы возможности трех основных инструментов — BeautifulSoup, Scrapy и Selenium. Каждый из них был проанализирован с точки зрения его применения в реальных задачах веб-скрапинга.
2. **Разработка примеров кода:** для каждого из рассмотренных инструментов были разработаны примеры кода, которые иллюстрируют основные приемы работы с данными, их обработки и сохранения. Эти примеры могут служить основой для дальнейших разработок и адаптации под конкретные задачи.
3. **Анализ конкурентной среды:** в работе был рассмотрен процесс анализа конкурентной среды, включая сравнительный анализ данных и разработку стратегий на основе полученных результатов. Были предложены конкретные методы для улучшения бизнес-процессов, основанные на анализе данных.
4. **Рекомендации по улучшению бизнес-процессов:** на основе проведенного анализа были предложены рекомендации по оптимизации бизнес-процессов, включая внедрение новых технологий и оценку их эффективности.

Все эти результаты свидетельствуют о том, что поставленные в начале работы цели были достигнуты в полном объеме.

### Перспективы дальнейших исследований

Несмотря на то, что в данной работе были достигнуты значительные результаты, остаются области, которые требуют дальнейшего изучения и разработки.

1. **Оптимизация инструментов веб-скрапинга:** Совершенствование методов извлечения данных с учетом специфики различных веб-ресурсов и их динамической природы. Это включает в себя более глубокое изучение использования Selenium для работы с динамическим контентом и исследование новых библиотек и фреймворков для веб-скрапинга.
2. **Обработка больших данных:** Разработка методов и подходов для обработки и анализа больших объемов данных, полученных в результате веб-скрапинга. Это может включать использование облачных технологий и распределенных систем для хранения и обработки данных.
3. **Машинное обучение и анализ данных:** Включение методов машинного обучения в процесс анализа данных, полученных с помощью веб-скрапинга. Это позволит более точно и эффективно анализировать данные, выявлять скрытые закономерности и делать прогнозы на основе полученной информации.
4. **Этические и правовые аспекты веб-скрапинга:** Исследование этических и правовых вопросов, связанных с использованием методов веб-скрапинга. Это особенно важно в свете растущего внимания к вопросам конфиденциальности данных и прав на интеллектуальную собственность.
5. **Интеграция с бизнес-процессами:** Дальнейшее изучение того, как результаты анализа данных могут быть интегрированы в бизнес-процессы для повышения их эффективности и конкурентоспособности. Это может включать разработку стратегий, ориентированных на клиента, улучшение качества обслуживания и оптимизацию производственных процессов.

Заключение подчеркивает важность непрерывного совершенствования методов извлечения и анализа данных, а также их применения в реальных бизнес-задачах. Дальнейшие исследования и разработки в этой области будут способствовать улучшению бизнес-процессов, повышению конкурентоспособности компаний и более эффективному использованию ресурсов.

# Список используемой литературы

### Книги

1. **Mitchell, R. (2018).** *Web Scraping with Python: Collecting More Data from the Modern Web. 2nd Edition.* O'Reilly Media.
2. **Chapagain, A. (2023).** *Hands-On Web Scraping with Python - Second Edition: Extract quality data from the web using effective Python techniques* (2nd ed.). Packt Publishing. ISBN 978-1837636211.
3. **Patel, J. M. (2020).** *Getting structured data from the Internet: Running web crawlers/scrapers on a big data production scale*. Apress. [Berkeley, CA].
4. **Russell, M. (2013).** *Mining the Social Web: Data Mining Facebook, Twitter, LinkedIn, Instagram, GitHub, and More.* O'Reilly Media, Inc.
5. **Barone, A. (2023). *Web scraping practical with Python: Crawling with Scrapy, Beautiful Soup, Selenium and more* (Kindle ed.). [Amazon Kindle]. ASIN B0C2CWH1TY.**
6. **Obekpa, A. (2024).** *Python for web scraping and crawling: A comprehensive guide to data extraction for business and research* (Kindle ed.). [Amazon Kindle]. ASIN B0DCT79YP6.

### Статьи

1. **Kahlon, N. K., & Singh, W. (2024).** Comparative analysis of web scraping tools for low-resource language text. *International Journal of Engineering Trends and Technology, 72*(1), 284-299. Источник: https://ijettjournal.org/Volume-72/Issue-1/IJETT-V72I1P128.pdf

### Веб-ресурсы

1. Densmore, J.(2017, July 23). Ethics in Web Scraping. Источник: <https://towardsdatascience.com/ethics-in-web-scraping-b96b18136f01>
2. Moz. (2019). Robots.txt. Источник: <https://moz.com/learn/seo/robotstxt>
3. Koshy, J. (2016, April 11). *Is Data scraping an Ethical practice?* Источник: <https://www.promptcloud.com/blog/is-data-scraping-ethical/>
4. Kansal, S. (2019, January 23). *Advanced Python Web Scraping: Best Practices & Workarounds.*Источник: <https://www.codementor.io/blog/python-web-scraping-63l2v9sf2q>
5. ScrapeHero (2019, October 10). *How to prevent getting blocked while scraping*. Источник: <https://www.scrapehero.com/how-to-prevent-getting-blacklisted-while-scraping/>

# Приложения

## Исходные коды:

Папка: source\_code

* + Пример 1.py — Пример кода для демонстрации импорт библиотеки Pandas и загрузка данных, демонстрация получения описательной статистики.
  + Пример 2.py — Пример кода для демонстрации визуализации данных.
  + Пример 3.ipynb — Пример кода для демонтсрации машинного обучения.
  + Пример 4.py — Пример кода для демонстрации работы BeautigulSoup.
  + Пример 5.py — Пример кода для извлечение заголовков статей и ссылок.
  + Пример 6.py — Пример кода для извлечение и сохранение изображений.
  + Папка Пример 7: books\_spider - Содержит файлы паука для веб-скрапинга данных о книгах.
  + Папка: gratisography\_spyder - Содержит файлы паука для веб-скрапинга изображений с сайта Gratisography.
  + Пример 9.py — Пример кода для извлечения данных о ноутбуках с веб-сайта Target.
  + Пример 10.ipynb — Пример кода расчета основной статистики для числовых признаков.
  + Пример 11.py — Пример кода использования линейной регрессии.
  + Пример 12.py — Пример кода использования ARIMA для прогнозирования временного ряда.
  + Пример 13.py — Пример кода для визуализации прогнозов.
  + Пример 14.py — Пример кода обработки данных.

## Дополнительные материалы:

Папка: images содержит рисунки, использованные в дипломной работе:

* + - Рисунок 1.png – гистограмма распределения рейтингов.
    - Рисунок 2.png - распределение приложений по категориям.
    - Рисунок 3.png - Boxplot для распределения рейтингов по категориям.
    - Рисунок 4.png – тепловая карта корреляций.
    - Рисунок 5.png - Кластеризация K-means на основе PCA.
    - Рисунок 6.png - Корреляционная матрица.
    - Рисунок 7.png - Визуализация прогноза.
    - Рисунок 8.png – Визуализация прогноза временного ряда.

Файлы данных:

* + books\_data.csv — Файл с данными о книгах, собранными в результате процедуры веб-скрапинга и последующего сохранения данных.
  + googleplaystore.csv — Файл с данными из Google Play Store.
  + test.csv — Файл с тестовыми данными, содержащий информацию по продаже недвижимости, собранную для анализа и проверки моделей.
  + WeeklyClosing.csv — Файл с данными о недельных закрытиях акций, используемыми для демонстрации и анализа временных рядов с помощью модели ARIMA..