УДК 004.8

БИБЛИОТЕКИ PYTHON ДЛЯ АНАЛИЗА ДАННЫХ: ПРЕДОБРАБОТКА И ПОДГОТОВКА ДАННЫХ

Ларин С.Э.

студент,

ФГБОУ ВО «Калужский государственный университет

им. К.Э. Циолковского»

Калуга, Россия

Белаш В.Ю.

к.пед.н., доцент,

ФГБОУ ВО «Калужский государственный университет

им. К.Э. Циолковского»

Калуга, Россия

Аннотация: Статья посвящена актуальной теме анализа данных с использованием языка программирования Python. В современном информационном обществе, где данные играют ключевую роль в различных сферах, от науки до бизнеса, процесс анализа данных становится неотъемлемой частью принятия важных решений. Статья охватывает основные этапы анализа данных, начиная от предобработки и подготовки данных с использованием модулей Python, таких как pandas и scikit-learn, до детального анализа и визуализации данных.

Ключевые слова: анализ данных, библиотеки python, подготовка данных, машинное обучение, предобработка данных, информационные технологии.

PYTHON LIBRARIES FOR DATA ANALYSIS: DATA PREPROCESSING AND PREPARATION

Larin S.E.

student,

Kaluga State University named after K. E. Tsiolkovsky
Kaluga, Russia
Belash V.Yu.
Ph.D., Associate Professor,
Kaluga State University named after K. E. Tsiolkovsky
Kaluga, Russia

Abstract: The article is devoted to the topical topic of data analysis using the Python programming language. In the modern information society, where data plays a key role in various fields, from science to business, the process of data analysis becomes an integral part of making important decisions. The article covers the main stages of data analysis, starting from data preprocessing and preparation using Python modules such as pandas and scikit-learn, to detailed data analysis and visualization.
Keywords: data analysis, python libraries, data preparation, machine learning, data preprocessing, information technology.

Анализ данных является неотъемлемой частью современного информационного общества. В различных сферах жизни человека, таких как: наука, медицина, бизнес, финансы, маркетинг и многие другие, ежедневно генерируется колоссальный объем данных. Анализ проводится в несколько этапов, включая сбор данных, их обработку и очистку, анализ и интерпретацию результатов, а также представление полученной информации в понятном виде.

Исходной точкой для извлечения ценной информации из этих данных является процесс анализа. В результате этого процесса выделяются ключевые паттерны, тренды и важные закономерности, что, в свою очередь, обеспечивает базу для принятия основных решений в конкретной предметной области.

На сегодняшний день Python является одним из наиболее перспективных языков программирования для проведения анализа данных. У Python есть ряд преимуществ, которые позволяют считать его совершенным инструментом, имеющим следующие достоинства: универсальность, обширная библиотека функций и легкость освоения. Также с развитием технологий машинного обучения анализ данных с Python получил широкое распространение.

Актуальность темы данной научной работы обусловлена тем, что анализ данных подходит для повышения эффективности бизнес-процессов, научных исследований и образовательных решений. Это делает тему ключевой в контексте современного информационного общества, в котором данные играют все более важную роль.

Прежде чем приступить к предобработке данных и их анализу, необходимо подробно рассмотреть ключевые модули (библиотеки) Python, представленные в настоящее время, которые могут быть задействованы для эффективного решения поставленной задачи.

Перейдем к рассмотрению каждого из этих модулей, чтобы получить полное представление об их функциональности.

В начале рассмотрим алгоритмические библиотеки, обеспечивающие разнообразные многочисленные инструменты обработки, анализа и моделирования данных.

- scikit-learn: универсальный инструментарий для машинного обучения, который позволяет решать задачи любой сложности. Он включает в себя все необходимое для подготовки данных, обучения моделей и оценки их эффективности. Основное назначение – решение задач машинного обучения и статистического анализа данных.
- statsmodels: библиотека предназначена для оценки статистических моделей и обеспечивает инструменты, необходимые для

проведения разнообразных статистических тестов, оценки регрессионных моделей и анализа временных рядов.

Теперь рассмотрим библиотеки визуализации. Они обеспечивают возможности для наглядного представления данных, что является ключевым аспектом в процессе анализа.

Библиотеки визуализации:

- matplotlib: помогает создавать статические, интерактивные и анимационные графики в Python. Обеспечивает обширный арсенал средств для визуализации данных в различных форматах, раскрывая широкий спектр возможностей для наглядного представления информации.
- seaborn: библиотека визуализации данных, основанная на matplotlib. Предоставляет высокоуровневый интерфейс для создания красочных и информативных статистических графиков.

И последний модуль, который осталось рассмотреть – библиотеки научных вычислений. Они предоставляют мощные инструменты для математических и научных расчетов, а также обработки данных.

Библиотеки научных вычислений:

- pandas: фреймворк, позволяющий обрабатывать и анализировать данные. Предоставляет мощные инструменты, включая структуры данных, например, DataFrame, который упрощает работу с табличными данными. Библиотека находит применение в процессах загрузки, очистки, анализа и подготовки данных перед интеграцией в алгоритмы машинного обучения.
- numpy: библиотека, предназначенная для манипулирования многомерными массивами и матрицами, а также возможность выполнять математические операции над ними. Предлагает разнообразие функций, оптимизированных под эффективную обработку числовых данных.
- scipy: высокоэффективная библиотека для выполнения научных и технических вычислений, которая расширяет дополнительную

функциональность для оптимизации, интеграции, интерполяции и многих других научных задач.

На рисунке 1 изображены все три модуля с описанием своих уникальных функций:

Рис. 1. Модули Python[1]

В качестве примера организации предобработки данных воспользуемся учебным датасетом "Титаник", ориентированным на начинающих в области машинного обучения. Цель исследования – проведение анализа данных и нахождение выживаемости пассажиров на основе доступных атрибутов.

Для реализации предобработки данных следует выполнить чтение данных из CSV-файла и загрузить их в объект DataFrame, используя библиотеку pandas. Этот этап включает в себя обнаружение и обработку пропущенных значений, выбор стратегии заполнения пропусков для столбцов, а также удаление несущественных признаков.

Перейдем к анализу датасета Титаник, подготовим все релевантные данные и применим рассмотренные модули python на практике.

Датасет содержит информацию о 891 пассажире. Данные включают в себя такие характеристики, как пол, возраст, класс билета и другие.

Первый этап – извлечение данных из CSV-файла. В области анализа данных, эффективное хранение и обработка табличных данных являются критическими аспектами. Один из наиболее распространенных форматов для представления табличной информации – CSV (Comma-Separated Values) –

[1] составлено авторами

простой и универсальный формат, который используется для хранения данных в виде текстовых файлов, основанный на разделении значений запятыми.

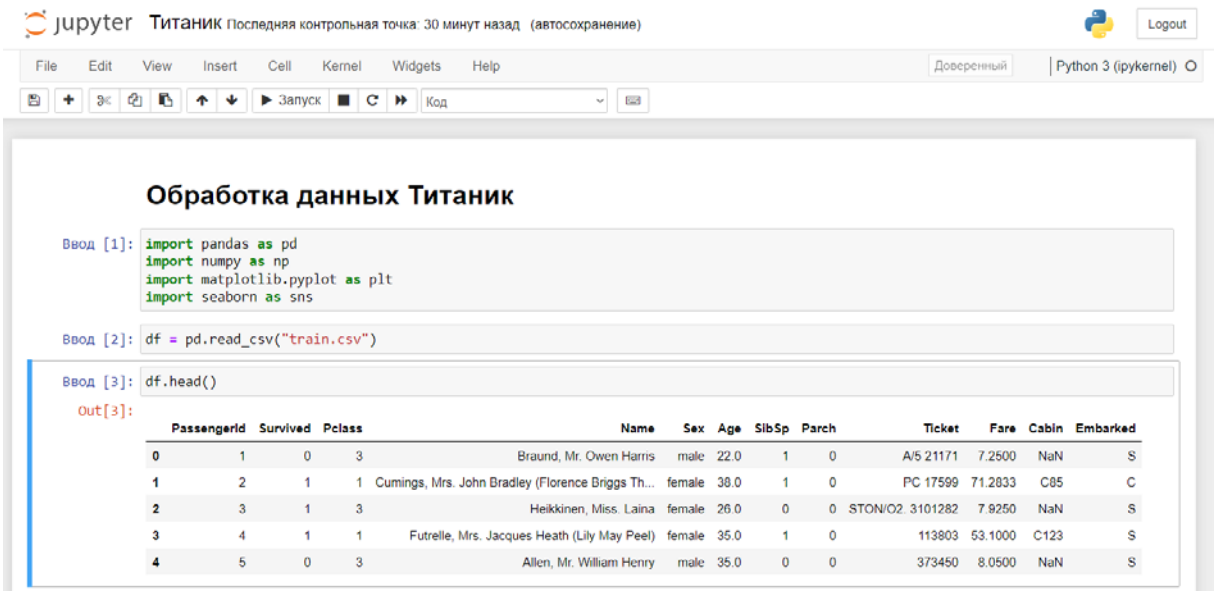
CSV предоставляет простой и удобный способ представления табличных данных. Он основан на идее разделения значений столбцов с использованием запятых, что обеспечивает легкость в чтении и интерпретации данных.

Загрузка в объект DataFrame осуществляется с помощью команды:

df = pd.read_csv("train.csv")

Данная строка кода выполняет чтение данных из CSV-файла "train.csv" и сохраняет их в переменной df в виде DataFrame. После выполнения этой строки, переменная df будет содержать табличные данные из файла, что позволяет легко выполнять различные операции и анализ этих данных с использованием богатого функционала Pandas.

Метод df.head() применяется для того, чтобы оценить структуру и содержание данных в DataFrame. Он предоставляет возможность просмотра первых пяти записей: их значения, наличие столбцов и данных (рис. 2).



	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

Рис. 2. Предобработка данных[2]

[2] составлено авторами

С использованием метода count(), можно проанализировать количество записей в каждом из столбцов данных. Этот подход позволяет сделать вывод о том, что общее количество пассажиров на борту Титаника составляет 891 (рис. 3).

```
Ввод [5]: df.count()

Out[5]: PassengerId    891
        Survived       891
        Pclass         891
        Name           891
        Sex            891
        Age            714
        SibSp          891
        Parch          891
        Ticket         891
        Fare           891
        Cabin          204
        Embarked       889
        dtype: int64
```

Рис. 3. Вывод данных со значениями[3]

Однако следует отметить, что не во всех столбцах имеются полные данные. Например, в столбцах "Age" (возраст), "Cabin" (каюта) и "Embarked" (порт посадки) наблюдаются пропущенные значения. Для корректного анализа и представления данных необходимо провести определенные манипуляции.

Для столбца "Age" рациональным решением является заполнение пропущенных данных медианным значением этого столбца, что позволит уменьшить влияние неполных данных на общий анализ.

В случае "Embarked" рекомендуется заполнить пропуски наиболее часто встречающимся портом посадки, что обеспечит логическую и статистическую достоверность данных.

В отношении столбца "Cabin" предлагается удалить его из датасета, поскольку в постановке задачи отсутствует явная необходимость в анализе данных о каютах пассажиров. Удаление этого столбца сделает датасет более

[3] составлено авторами

легким и удобным для последующего анализа, сосредотачивая внимание на более значимых переменных. На рисунке 4 представлена полная предобработка неполных данных для различных столбцов датасета.

Рис. 4. Предобработка неполных данных[4]

Теперь необходимо с помощью словаря закодировать категориальные переменные, поскольку при работе с большинством алгоритмов машинного обучения, ожидаются числовые значения в данных. Значения порта посадки "S", "C" и "Q" были закодированы соответственно числовыми значениями 0, 1 и 2. Этот процесс облегчит последующий анализ, позволяя использовать числовые значения вместо текстовых меток. Далее с помощью метода value_counts () осуществляется подсчет уникальных значений в столбце "Embarked". Дополнительно с помощью кода df["Embarked"].fillna(df["Embarked"].value_counts().index[0], inplace = True) пропущенные значения заполняются наиболее часто встречающимся значением в столбце. Благодаря методу df.count() можно увидеть, что теперь в каждом столбце нет пропущенных значений (рис.5).

[4] составлено авторами

```
Ввод [11]: # с помощью словаря закодируем Embarked
           d = {"S": 0, "C": 1, "Q": 2}
           df["Embarked"] = df["Embarked"].map(d)

Ввод [12]: df["Embarked"].value_counts()

Out[12]: 0.0    644
         1.0    168
         2.0     77
         Name: Embarked, dtype: int64

Ввод [13]: df["Embarked"].fillna(df["Embarked"].value_counts().index[0], inplace = True)

Ввод [14]: df.count()

Out[14]: PassengerId    891
         Survived       891
         Pclass         891
         Name           891
         Sex            891
         Age            891
         SibSp          891
         Parch          891
         Ticket         891
         Fare           891
         Embarked       891
         dtype: int64
```

Рис. 5. Обработанные данные[5]

В итоге был выполнен важный процесс – предобработка данных. Ее основная цель – подготовить и очистить данные для дальнейшего анализа или обучения модели.

Анализ данных с Python является востребованным навыком в современном мире. Этот язык отличается относительной простотой в изучении, что обеспечивает его доступность для начинающих.

В этой статье были рассмотрены основные этапы анализа данных с использованием библиотек Python: предобработка и подготовка данных. На примере датасета "Титаник" были продемонстрированы основные приемы предобработки данных, такие как обнаружение и обработка пропущенных значений, выбор стратегии заполнения пропусков для столбцов, а также удаление несущественных атрибутов.

Обработка и предобработка данных с Python является важным этапом анализа данных, который позволяет подготовить данные к дальнейшей обработке и реализации модели машинного обучения.

[5] составлено авторами

Библиографический список

1. Data Visualization with Titanic / [Электронный ресурс] // Kaggle: [сайт]. — URL: https://www.kaggle.com/code/enessasmaz/data-visualization-with-titanic (дата обращения: 28.11.2023).
2. Анализ данных: основные этапы процесса правильной обработки информации / [Электронный ресурс] // apipython: [сайт]. — URL: https://apipython.ru/analiz-dannyh-osnovnye-etapy-proczessa-pravilnoj-obrabotki-informaczii/ (дата обращения: 28.11.2023).
3. Введение в анализ данных / [Электронный ресурс] // miptstats.github.io: [сайт]. — URL: https://mipt-stats.gitlab.io/courses/ad_fivt/titanik.html (дата обращения: 28.11.2023).
4. Разбор задачи Титаник / [Электронный ресурс] // habr: [сайт]. — URL: https://habr.com/ru/articles/655955/ (дата обращения: 24.11.2023).

Оригинальность 86%