ORES
Научно-практический электронный журнал
Оригинальные исследования (ОРИС), *ISSN 2222-0402*
www.ores.su
original-research.ru

ПРИМЕНЕНИЕ БИБЛИОТЕКИ PYTHON PANDAS ДЛЯ СОЗДАНИЯ ПРОГНОЗНОЙ МОДЕЛИ

Чеснокова Мария Николаевна,
Студент группы ИУК5-52Б Калужского филиала федерального государственного бюджетного образовательного учреждения высшего образования «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)», 248000, Россия, г. Калуга, ул. Баженова, д.2.
chesnokovamn@student.bmstu.ru

Федоров Виктор Олегович,
Кандидат технических наук, доцент Калужского филиала федерального государственного бюджетного образовательного учреждения высшего образования «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет) », 248000, Россия, г. Калуга, ул. Баженова, д.2.
fedorov_vo@bmstu.ru

Аннотация

В статьи рассматривается решение проблемы выбора смартфона: создание прогнозной модели. Было предложено использовать библиотеку Pandas.

Ключевые слова: Python, Pandas, создание прогнозной модели

USING THE PYTHON PANDAS LIBRARY TO CREATE A PREDICTIVE MODEL

Maria N. Chesnokova,
Student of group IUK5-52B of Bauman Moscow State Technical University (Kaluga Branch), 248000, Russia, Kaluga, Bazhenova st., 2.
chesnokovamn@student.bmstu.ru

Fedorov Viktor Olegovich,
Candidate of Technical Sciences, Docent of of Bauman Moscow State Technical University (Kaluga Branch), 248000, Russia, Kaluga, Bazhenova st., 2.

ABSTRACT

The article discusses the solution to the problem of choosing a smartphone: creating a predictive model. It was suggested to use the Pandas library.

Keywords: Python, Pandas, creating a predictive model.

ВВЕДЕНИЕ

Современный человек ежедневно сталкивается с огромным массивом разнообразных данных, представляющих закодированные описания сущностей, явлений, событий. Даже, такое, казалось бы, простое желание пользователя как выбор нового мобильного телефона может оказаться трудноосуществимым. Открывая веб-сайт интернет-магазина с интересующими его товарами, пользователь должен быть морально готов к тому, что на него обрушится целый шквал всевозможных моделей. Для того чтобы найти среди всего многообразия данных нужный ему вариант, пользователь должен дополнительно обработать, переработать и проанализировать полученные данные. Например, он может отсортировать список и выбрать интересующий его товар, воспользовавшись специальными фильтрами.

Однако любому пользователю хочется выбрать оптимальную модель телефона по таким критериям, как: цена, характеристики, бренд, которые не просто были бы актуальными в текущей рыночной ситуации, но и продолжали быть конкурентоспособными на протяжение последующих нескольких лет. Для этого необходимо создать прогнозную модель, которая бы отражала специфику рынка современных мобильных телефонов.

На сегодняшний день операции по извлечению данных, анализу и трансформации наборов данных являются полностью автоматизированными. Для автоматизации обработки структурированных данных используются специализированные программные средства и библиотеки. Такой инструментарий доступен для широкого круга языков программирования и для всех основных операционных систем. Обычно предпочтение отдается языкам, созданным для научной сферы, таким как FORTRAN, MATLAB, JULIA и R, но также могут использоваться и другие высокоуровневые языки, такие как C++, C#, JAVA, JAVASCRIPT, PHP или Python.

В данной статье мы рассмотрим применение библиотеки Python Pandas для создания прогнозной модели с целью сегментации смартфонов по различным критериям (цена, характеристики, бренд), что поможет лучше понимать структуру рынка и целевые аудитории.

Используемые инструменты

Библиотека Python Pandas позволяет легко агрегировать и группировать данные [3], выявляя тенденции в изменении цен, популярности брендов, востребованности технологий (5G, NFC), предпочтениях пользователей к характеристикам устройств. Pandas упрощает для пользователя анализ характеристик смартфонов разных брендов по цене, характеристикам, рейтингу, что позволяет оценивать конкурентоспособность продуктов и выявлять сильные и слабые стороны. Pandas интегрируется с библиотеками визуализации, такими как Matplotlib [4] и Seaborn, что позволяет представлять результаты анализа в виде информативных графиков и диаграмм. Визуализация помогает быстро схватывать тенденции [5], закономерности и выводы, сделанные в ходе анализа.

Pandas — это библиотека, предоставляющая функции анализа и структуры хранения, предназначенные специально для работы с данными реляционного типа (например, с базами данных). Она может принимать в качестве параметров массивы NumPy. С целью сделать Pandas самым мощным инструментом манипулирования с открытым исходным кодом в отрасли [1], его разработчики оптимизировали многие низкоуровневые процедуры путем замены кода Python на предварительно скомпилированный и оптимизированный код C, который работает быстрее. В основном данная библиотека состоит из двух структур хранения [5], которые покрывают практически все потребности в представлении данных, используемых в финансах, статистике, науке и технике.

Постановка задачи

Предположим, что у нас есть набор данных о смартфонах, представленный в табличном виде. Заголовки столбцов включают в себя следующие позиции: Название бренда, цену, рейтинг, has_5g, has_nfc и т.д. Приведенный набор данных может быть использован для любого вида анализа и прогнозной модели. Для нашего примера возьмем уже имеющийся DataSet [2].

Решение задачи

Приступаем к получению информации из DataSet. Для этого получим данные с сайта, чтобы посмотреть в каком виде представлена информация.

```
import pandas as pd

df = pd.read_csv('smartphones_cleaned_v6.csv')
print(df.head(10))
```

Полученную информацию сгруппируем в таблицы 1-4.

Таблица 1.

название бренда	цена	рейтинг	Сеть_5g	NFC	ИК-бластером
бренд О1	19989	81.0	True	False	False
бренд О2	54999	89.0	True	True	False
бренд С	16499	75.0	True	False	False
бренд М	14999	81.0	True	False	False
бренд Р	24999	82.0	True	False	False
бренд С	16999	80.0	True	True	False
бренд А	65999	81.0	True	True	False
бренд Х	29999	86.0	True	False	True
бренд Н	26749	85.0	True	True	False
бренд О3	28999	84.0	True	True	False
….	….	….	….	….	…

Таблица 2.

название бренда	модель процессора	количество ядер	скорость процессора	емкость батареи	быстрая зарядка
бренд О1	snapdragon	8.0	3.20	5000.0	Да
бренд О2	snapdragon	8.0	2.20	5000.0	Нет
бренд С	exynos	8.0	2.40	5000.0	Нет
бренд М	snapdragon	8.0	2.20	5000.0	Да
бренд Р	dimensity	8.0	2.60	5000.0	Да
бренд С	snapdragon	8.0	2.20	5000.0	Да
бренд А	bionic	6.0	3.22	3279.0	Нет
бренд Х	dimensity	8.0	2.60	4980.0	Нет
бренд Н	snapdragon	8.0	2.50	4500.0	Да
бренд О3	dimensity	8.0	3.00	4500.0	Да
…	…	…	…	….	….

Таблица 3.

название бренда	внутренняя память	размер экрана	частота обновления экрана	Операционная система	объем оперативной памяти
бренд О1	256.0	6.70	120	android	12.0
бренд О2	128.0	6.59	120	android	6.0
бренд С	64.0	6.60	90	android	4.0
бренд М	128.0	6.55	120	android	6.0
бренд Р	128.0	6.70	120	android	6.0
бренд С	128.0	6.60	120	android	6.0
бренд А	128.0	6.10	60	ios	6.0
бренд Х	256.0	6.67	120	android	8.0
бренд Н	128.0	6.55	120	android	8.0
бренд О3	128.0	6.43	90	android	8.0
….	….	…	…	….	….

Таблица 4.

название бренда	основная камера	к-во задних камер	фронтальная камера	ширина разрешения	высота расширения
бренд О1	50.0	3	16.0	1440	3216
бренд О2	64.0	3	16.0	1080	2412
бренд С	50.0	3	13.0	1080	2408
бренд М	50.0	3	16.0	1080	2400
бренд Р	108.0	3	16.0	1080	2412
бренд С	50.0	3	8.0	1080	2408
бренд А	12.0	2	12.0	1170	2532
бренд Х	200.0	3	16.0	1080	2400
бренд Н	50.0	2	16.0	1080	2400
бренд О3	50.0	3	32.0	1080	2400
…	…	…	…	…	…

Следующая команда выведет краткую сводку о структуре и содержимом датафрейма, включая количество строк, названия столбцов, количество непустых значений в каждом столбце и их тип данных. В результате выполнения этого вы увидите информацию о вашем датафрейме, которая поможет вам понять его состав и подготовить данные для анализа или обработки.

```
print(df.info())
```

вывод:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 980 entries, 0 to 979
```

```
Data columns (total 25 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   название бренда         980 non-null    object
 1   цена                    980 non-null    int64
 2   рейтинг                 879 non-null    float64
 3   Сеть_5g                 980 non-null    bool
 4   NFC                     980 non-null    bool
 5   ИК-бластером            980 non-null    bool
 6   модель процессора       960 non-null    object
 7   количество ядер         974 non-null    float64
 8   скорость процессора     938 non-null    float64
 9   емкость батареи         969 non-null    float64
 10  доступна быстрая зарядка  980 non-null  int64
 11  быстрая зарядка         769 non-null    float64
 12  объем оперативной памяти  980 non-null  float64
 13  внутренняя память       980 non-null    float64
 14  размер экрана           980 non-null    float64
 15  частота обновления экрана  980 non-null  int64
 16  к-во задних камер       980 non-null    int64
 17  к-во передних камер     976 non-null    float64
 18  Операционная система    966 non-null    object
 19  основная камера         980 non-null    float64
 20  фронтальная камера      975 non-null    float64
 21  расширено до            500 non-null    float64
 22  ширина разрешения       980 non-null    int64
 23  высота расширения       980 non-null    int64
dtypes: bool(3), float64(12), int64(7), object(3)
memory usage: 171.4+ KB
None
Далее перейдём к непосредственному анализу:
print(df[['цена', 'рейтинг']].corr())
```

Интерпретация коэффициента корреляции

После того, как была получена матрица корреляции для столбцов 'цена' и 'рейтинг' из DataFrame, следует её интерпретировать.

Значение 1.000000 на диагонали указывает на идеальную корреляцию каждой переменной с самой собой. Значение 0.283504 вне диагонали представляет собой коэффициент корреляции между 'ценой' и 'рейтингом'.

Коэффициент 0.283504 указывает на положительную, но слабую корреляцию между ценой и рейтингом. Это означает, что с увеличением цены рейтинг склонен увеличиваться незначительно, однако связь не очень сильная.

Отсюда можно сделать вывод, что в данных присутствует слабая положительная корреляция между ценой и рейтингом.

Чтобы посчитать количество смартфонов в каждом интервале рейтинга на гистограмме, используем параметр bins в функции plt.hist. Этот параметр определяет количество интервалов, или корзин, на которые разбивается диапазон значений рейтинга. Каждая корзина представляет собой интервал значений, и высота столбца в гистограмме показывает количество смартфонов, чьи рейтинги попадают в соответствующий интервал.

```
# Вывод основных статистик по колонке 'рейтинг'
```

```
print(df['рейтинг'].describe())

# Построение гистограммы распределения рейтингов смартфонов
plt.figure(figsize=(8, 6))
plt.hist(df['рейтинг'], bins=10, edgecolor='black')
plt.title('Распределение рейтингов смартфонов')
plt.xlabel('Рейтинг')
plt.ylabel('Количество смартфонов')
plt.show()
```

В результате получаем следующие значения:

count 879.000000 – количество смартфонов

mean 78.258248 - среднее значение рейтинга смартфонов.

std 7.402854 - стандартное отклонение, показывающее разброс рейтингов вокруг среднего значения.

min 60.000000 - минимальное значение рейтинга.

25% 74.000000 - значение, ниже которого находится 25% рейтингов (первый квартиль).

50% 80.000000 - медиана, значение, ниже и выше которого находится по 50% рейтингов(второй квартиль).

75% 84.000000 - значение, ниже которого находится 75% рейтингов (третий квартиль).

max 89.000000 - максимальное значение рейтинга.

Распределение рейтингов смартфонов показано на рисунке 1.

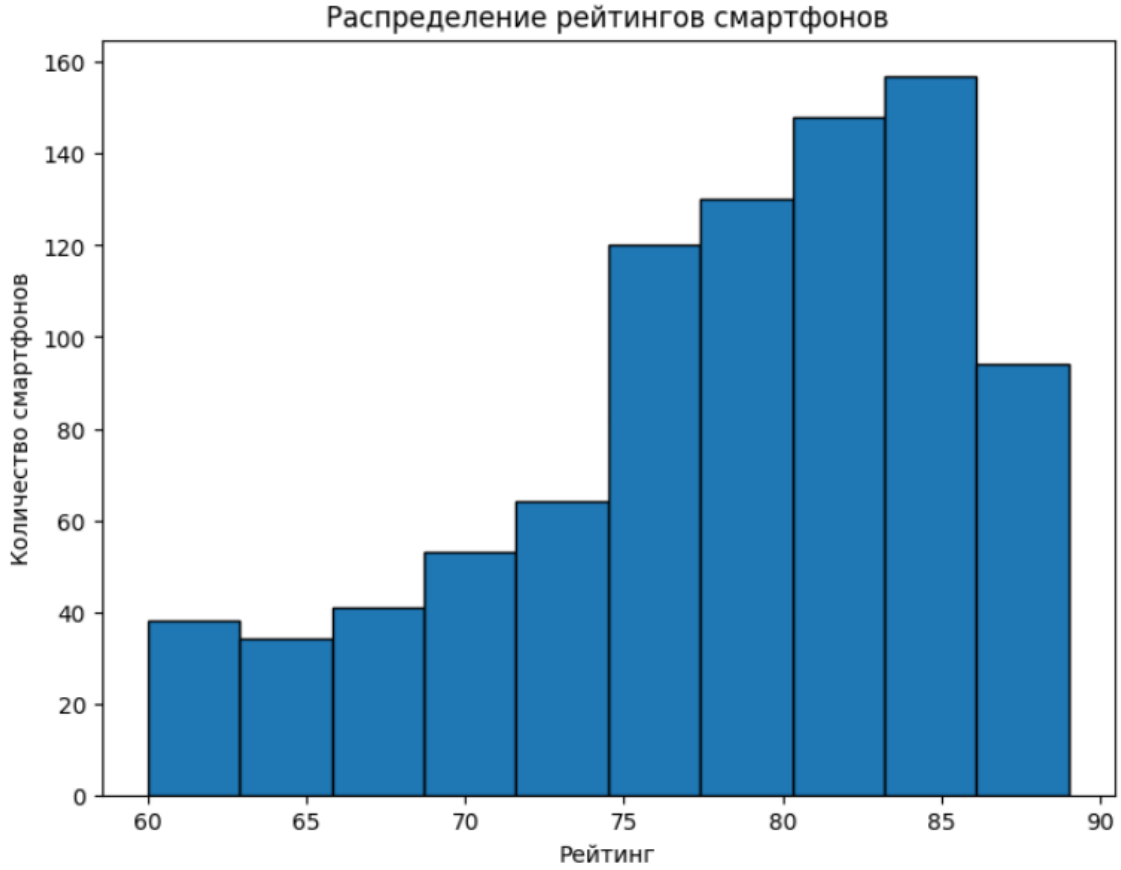

Рисунок 1 – Распределение рейтингов смартфонов.

Гистограмма показывает, что распределение рейтингов смартфонов сосредоточено преимущественно в диапазоне от 60 до 89. Большинство смартфонов имеют рейтинг около 78, но если смотреть только на этот график, то - 85.

Теперь следует узнать, у какого бренда самый высокий рейтинг, а у какого - самый низкий. Для того, чтобы это сделать, воспользуемся следующим кодом:

```
brand_ratings = df.groupby('название бренда')['рейтинг'].mean().sort_values(ascending=False)
print(brand_ratings)

plt.figure(figsize=(10, 6))
brand_ratings.plot(kind='bar')
plt.title('Средний рейтинг по брендам')
plt.xlabel('Бренд')
plt.ylabel('Средний рейтинг')
plt.xticks(rotation=45, ha='right')
plt.show()
```

При помощи этого кода мы можем построить график (см. рисунок 2). Он показывает, какие бренды имеют самый высокий и самый низкий средний рейтинг среди всех брендов в данных.

По графику можно сравнить различные бренды по их средним рейтингам. Это поможет определить популярность или удовлетворенность потребителей товарами от каждого бренда.

График также может показать тенденции или выбросы в оценках, что может быть полезно для дальнейшего анализа причин высоких или низких рейтингов.

Этот код полезен для визуализации и анализа данных о рейтингах брендов, что помогает понять их восприятие на рынке и принимать соответствующие бизнес-решения.

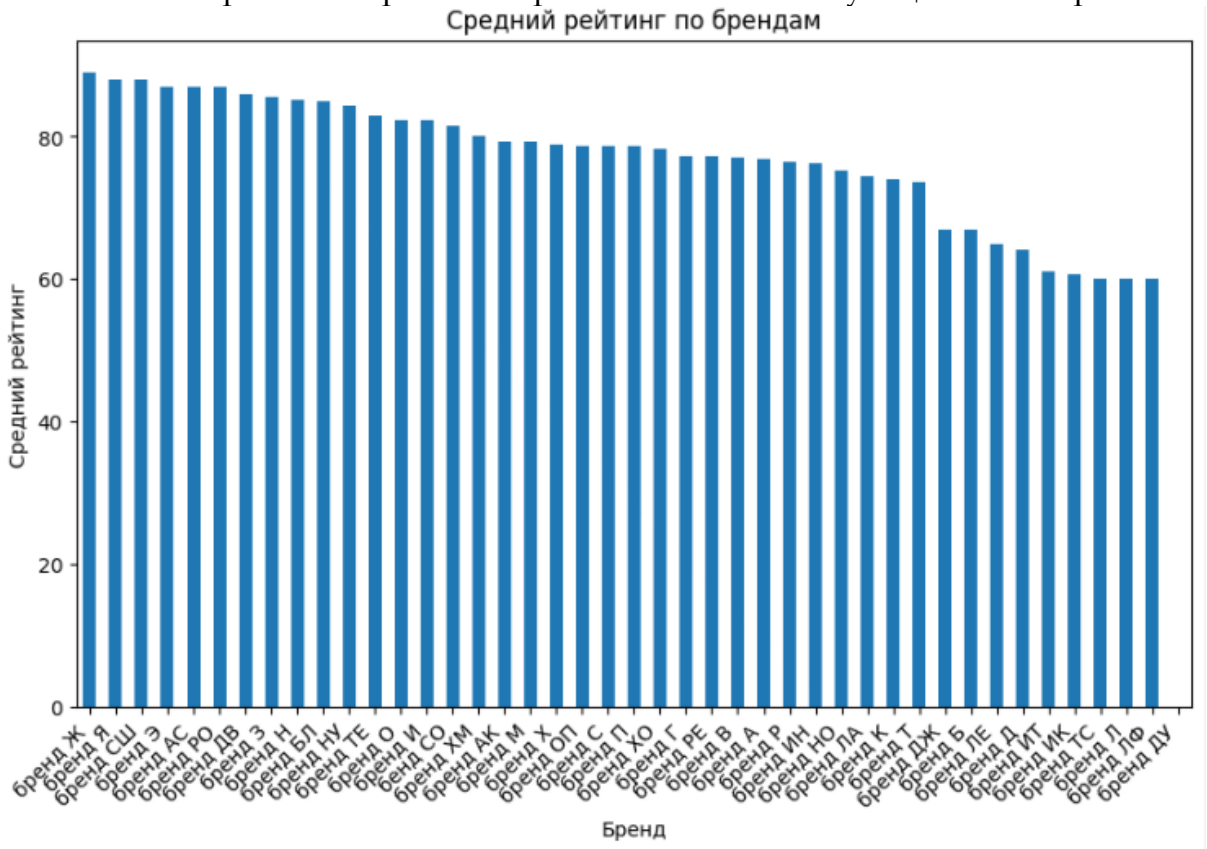

Рисунок 2 – Средний рейтинг смартфонов по брендам

На графике мы видим, что самый высокий рейтинг у бренда Ж, а самый низкий у бренда ДУ, Л, ЛФ, еще мы можем увидеть бренды по середине у бренда ХО, П, С, ОП, Х.

Давайте посмотрим, что у них за различия возьмем 3 бренда максимальный Ж, минимальный Л и средний С.

```
brand1 = 'бренд Ж' # высокий рейтинг
brand2 = 'бренд С'# средний рейтинг
brand3 = 'бренд Л'# низкий рейтинг
brand1_data = df[df['название бренда'] == brand1]
brand2_data = df[df['название бренда'] == brand2]
brand3_data = df[df['название бренда'] == brand3]
# Выбор атрибутов для сравнения
attributes = ['цена', 'рейтинг', 'Сеть_5g', 'NFC','скорость процессора', 'быстрая зарядка', 'объем оперативной памяти',
    'внутренняя память', 'размер экрана', 'частота обновления экрана',
    'к-во задних камер', 'к-во передних камер',
     'основная камера', 'фронтальная камера', 'доступная расширенная память', 'ширина разрешения', 'высота расшерения']

# Создание словаря для хранения сравнительных данных
comparison_data = {}

# Сравнение средних значений для каждого атрибута
for attribute in attributes:
        comparison_data[attribute] = [brand1_data[attribute].mean(), brand2_data[attribute].mean(), brand3_data[attribute].mean()]

# Создание DataFrame для визуализации
comparison_df = pd.DataFrame(comparison_data, index=[brand1, brand2, brand3])

# Вывод DataFrame в виде таблицы
print(comparison_df.to_string())
```

Полученные данные округлим и представим в виде таблиц 5 и 6. Обращаем внимание, что здесь представлена не вся информация.

Таблица 5.

	Цена	Рейтинг	Сеть 5g	NFC	Скорость процессора	Быстрая зарядка	Объём оперативной памяти	Внутренняя память	Размер экрана	Частота обновления экрана
Бренд Ж	124990	89.00	1.00	1.00	3.0	NaN	12.00	256	6.60	240.00
Бренд С	36832	78.72	0.52	0.51	2.4	24.07	6.54	137	6.55	88.36
Бренд Л	6165	60.00	0.00	0.00	1.9	10.00	4.67	43	6.51	60.00

Таблица 6.

	К-во задних камер	К-во передн	Основная камера	Фронтальная камера	Доступная расширен	Ширина разрешения	Высота разрешения

		их камер			ная память		
Бренд Ж	2	1	47.2	12.6	1.0	1260	2730
Бренд С	3	1	54.2	16.7	0.8	1103	2272
Бренд Л	2	1	9. 7	6.0	0.0	720	1560

Если мы посмотрим на параметры, мы можем определить их и выбрать телефон по ним, и по тому бюджету, который нас устраивает. Удобный анализ мы можем сравнивать и по другим атрибутам, или наоборот убрать те, которые нас не интересуют.

Выводы:

- В данных наблюдается слабая положительная корреляция между ценой и рейтингом смартфонов.
- Распределение рейтингов смартфонов сосредоточено в диапазоне от 60 до 89, при этом большинство смартфонов имеют рейтинг около 78.
- Бренды с более высоким рейтингом, как правило, предлагают более высокие технические характеристики, такие как скорость процессора, объем памяти, размер экрана и количество камер.
- Анализ позволяет выбрать смартфоны по конкретным параметрам и бюджету, учитывая технические характеристики и рейтинг бренда.

В целом, проведенный анализ предоставляет ценную информацию для принятия решений о покупке смартфонов, учитывая как технические характеристики, так и рейтинги брендов.

Проведенный анализ смартфонов с использованием библиотеки Pandas предоставляет набор практических навыков и подходов, которые можно применить к анализу других датасетов.

Список литературы:

1. Pandas – [Электронный ресурс]. – URL: http://pandas.pydata.org (дата обращения: 05.06.2024)
2. Smartphones_Dataset – [Электронный ресурс]. – URL: https://www.kaggle.com/datasets/informrohit1/smartphones-dataset/data?select=smartphones_cleaned_v6.csv (дата обращения: 05.06.2024)
3. Ильичев, В.Ю., Юрик, Е.А. Анализ массивов данных с использованием библиотеки Pandas для Python// Научное обозрение. Технические науки. 2020. №4. с.41-45
4. Стычев, С.Н., Краснопевцева, Н.А. Анализ демографической ситуации при помощи библиотеки Pandas языка программирования Python // Инновационные научные исследования. 2021. №3-2(5). с. 221-226
5. Дрянкова, Д.А. Обработка пропущенных значений и дубликатов в данных с помощью библиотеки PANDAS для языка программирования Python// Дневник науки. 2023. 6(78). Порядковый номер: 20

References:

1. Pandas – [Electronic resource]. – URL: http://pandas.pydata.org (date of application: 06/05/2024)
2. Smartphones_Dataset – [Electronic resource]. – URL: https://www.kaggle.com/datasets/informrohit1/smartphones-dataset/data?select=smartphones_cleaned_v6.csv (date of application: 06/05/2024)
3. Ilyichev, V.Yu., Yurik, E.A. Analysis of data arrays using the Pandas library for Python// Scientific Review. Technical sciences. 2020. No.4. pp.41-45
4. Sychev, S.N., Krasnopevtseva, N.A. Demographic situation analysis using the Pandas library of the Python programming language // Innovative scientific research. 2021. No.3-2(5). pp. 221-226
5. Dryankova, D.A. Processing of missing values and duplicates in data using the PANDAS library for the Python programming language// The diary of Science. 2023. 6(78). Serial number: 20.