Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

|  |
| --- |
| Институт информационных технологий  и анализа данных |
| наименование института |

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2**

по дисциплине «Технологии обработки, анализа и визуализации данных»

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Выполнил студент |  | ИИТм-23-1 |  |  |  | Солопов Д. Д. |
|  |  | шифр группы |  | подпись |  | И.О. Фамилия |
| Проверил |  |  |  |  |  | Григорьев С.В. |
|  |  |  |  | подпись |  | И.О. Фамилия |

Отчет защищен с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Иркутск 2024 г

Содержание

[Цель работы 3](#_Toc166285824)

[Загрузка набора данных 4](#_Toc166285825)

[Одномерный анализ данных 18](#_Toc166285826)

[Двумерный анализ данных 22](#_Toc166285827)

[Выводы 27](#_Toc166285828)

[Список использованных источников 28](#_Toc166285829)

Цель работы

Изучить основные этапы разведочного анализа данных с применением различных методов из различных библиотек для языка программирования (на выбор R или Python) для данных по выбранному варианту.

**Задачи:**

1. Загрузить и провести первичный осмотр данных, описать предметную область и предложить назначение данных.
2. Обработать пропущенные значения в наборе данных.
3. Выявить выбросы и аномалии в наборе данных, по возможности использовать методы визуализации.
4. Провести анализ распределения переменных в наборе данных, по возможности использовать методы визуализации.
5. Провести одномерный анализ для изучения категориальных переменных, по возможности использовать методы визуализации.
6. Провести двумерный анализ между переменными в наборе данных, по возможности использовать методы визуализации.

Загрузка набора данных

В качестве платформы, с помощью которой будет производиться разведочный анализ данных, была выбрана платформа Google Colab, поскольку на ней есть заранее предустановленные зависимости и можно легко выводить данные любого формата (таблица, графики и т.д.).

В качестве набора данных используется консолидированная финансовая отчётность за квартал июнь 2023 года условной компании “NAISAL”.

Начнём с импорта необходимых библиотек и чтения данных.

В рамках проведенния разведочного анализа данных (EDA) будут использованы следующие библиотеки:

import numpy as np # numpy

import pandas as pd # pandas

import matplotlib.pyplot as plt # pyplot

import seaborn as sns # seaborn

import matplotlib as mpl # matplotlib

Загрузка данных производится следующим образом:

data = pd.read\_csv("/content/na-isal-june-2023-quarter-consolidated-accounts.csv")

В переменной data будут находится все считанные данные. На рисунке 1 представлен вывод 5-ти первых строк загруженной таблицы.

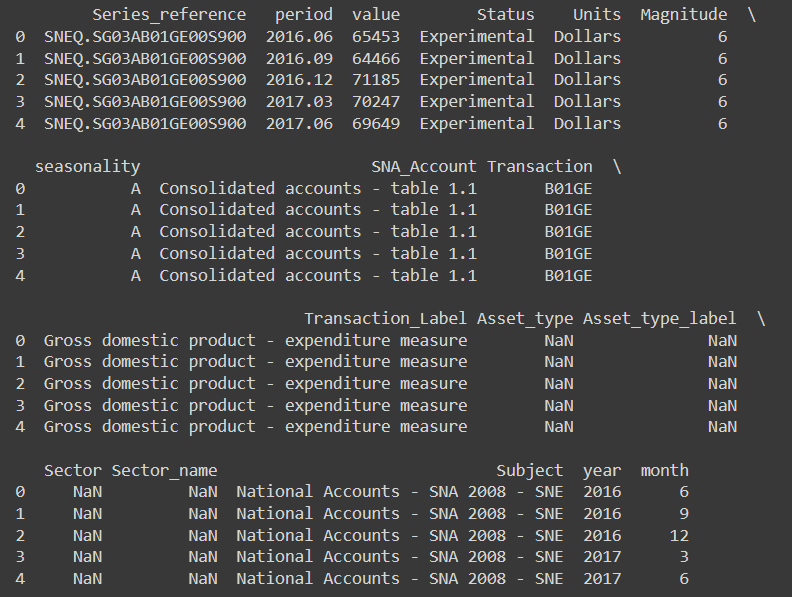


Рисунок 1 – Вывод первых 5-ти строк

Наименование всех столбцов представлено на английском языке. Чтобы было удобно демонстрировать графики и выводить информацию о столбцах переименуем все столбцы на русский язык. Это можно сделать с помощью следующего программного кода:

columnsName = {

'Series\_reference':' Ссылка на серию',

'period':'Период',

'value': 'Значение',

'Status': 'Статус',

'Units': 'Единицы',

'Magnitude': 'Величина',

'seasonality': 'Сезонность',

'SNA\_Account': 'Счёт SNA',

'Transaction': 'Транзакция',

'Transaction\_Label': 'Метка транзакции',

'Asset\_type': 'Тип актива',

'Asset\_type\_label': 'Метка типа актива',

'Sector': 'Сектор',

'Sector\_name': 'Название сектора',

'Subject': 'Субъект',

'year': 'Год',

'month': 'Месяц'

}

data.rename(columns = columnsName, inplace = True )

Теперь можно вывести более подробную информацию о столбцах. Более подробная информация о столбцах, типах в столбцах и занимаемой памяти представлена на рисунке 2.

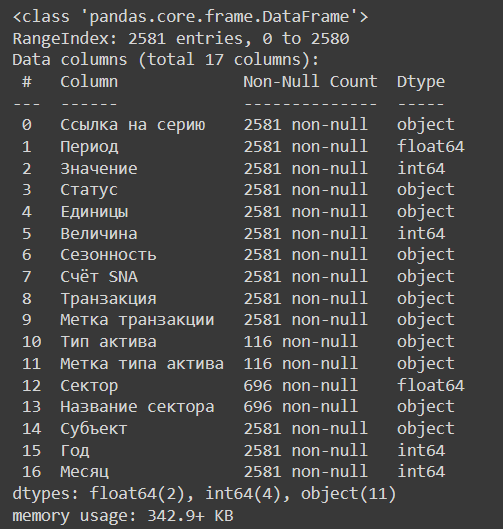


Рисунок 2 – Подробная информация о столбцах таблицы

Исходя из рисунка 2 получаем, что общий объём памяти, занимаемый таблицей, составляет 342,9 KB.

Количество столбцов - 17, количество всех строк - 2581. В столбцах “Тип актива”, “Метка типа актива”, “Сектор” и “Название сектора” присутствуют null-значения.

Всего в столбцах встречается 3 вида типа данных: object (11 столбцов), float64 (2 столбца) и int64 (4 столбца).

В столбцах с типом данных object хранятся строки.

Для более подробной информации о хранении данных с null-значениями в таблице необходимо использовать следующую команду:

data.isnull().sum()



Рисунок 3 – Подробная статистика по null-значениям в столбцах

Также выведем процент отсутствующих значений по столбцам, выполнив следующий программный код:

for col in data.columns:

print(f'{col}: {data[col].isna().sum() / data.shape[0] \* 100:.2f}%')

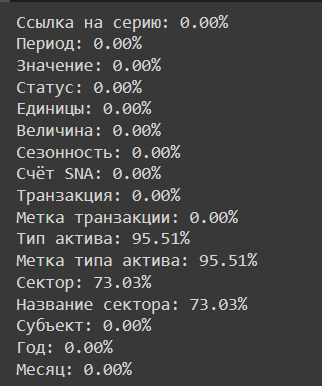


Рисунок 4 – Статистика (в %) распределения отсутствующих значений по столбцам набора данных

Из полученной статистики (см. рис. 4) видно, что в столбцах “Тип актива” и “Метка типа актива” наибольший процент пропущенных значений – 95.51%. В столбце “Сектор” и “Название сектора” процент пропущенных значений чуть меньше – 73.03%.

Пропущенные значения в столбцах “Тип актива” и “Метка типа актива” восстановить не удастся, поскольку их процент очень большой – 95.51%.

Можно также визуализировать пропуски данных с помощью тепловой карты (heatmap):

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

sns.heatmap(data.isna().transpose())

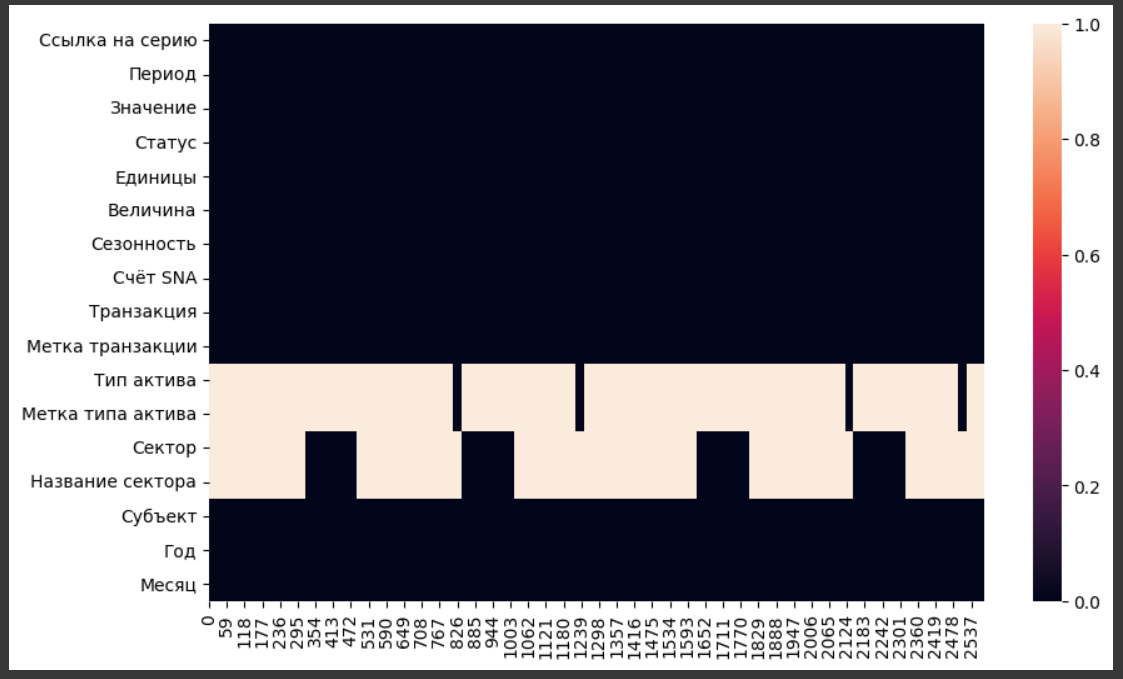


Рисунок 5 – Визуализация пропусков в данных с помощью тепловой карты

Светлые полосы, из полученной тепловой карты, это пропущенные значения в наборе данных. Тут также визуально можно заметить, что столбец “Тип актива” и “Метка типа актива” практически не содержат никаких полезных данных, поэтому эти столбцы могут быть удалены из набора данных.

Чтобы убедиться, что удалять эти столбцы действительно имеет смысл, посмотрим сколько уникальных значений есть в данных столбцах:



Рисунок 6 – Уникальные значения в столбцах “Тип актива” и “Метка типа актива” соответственно

Т.к. в каждом столбце значение всего одно и нет никакого разнообразия, то нельзя даже сформулировать зависимость данных столбцов от других и восстановить данные. Поэтому данные столбцы удаляем с помощью следующего программного кода:

data.drop('Тип актива', axis=1 , inplace=True)

data.drop('Метка типа актива', axis=1, inplace=True)

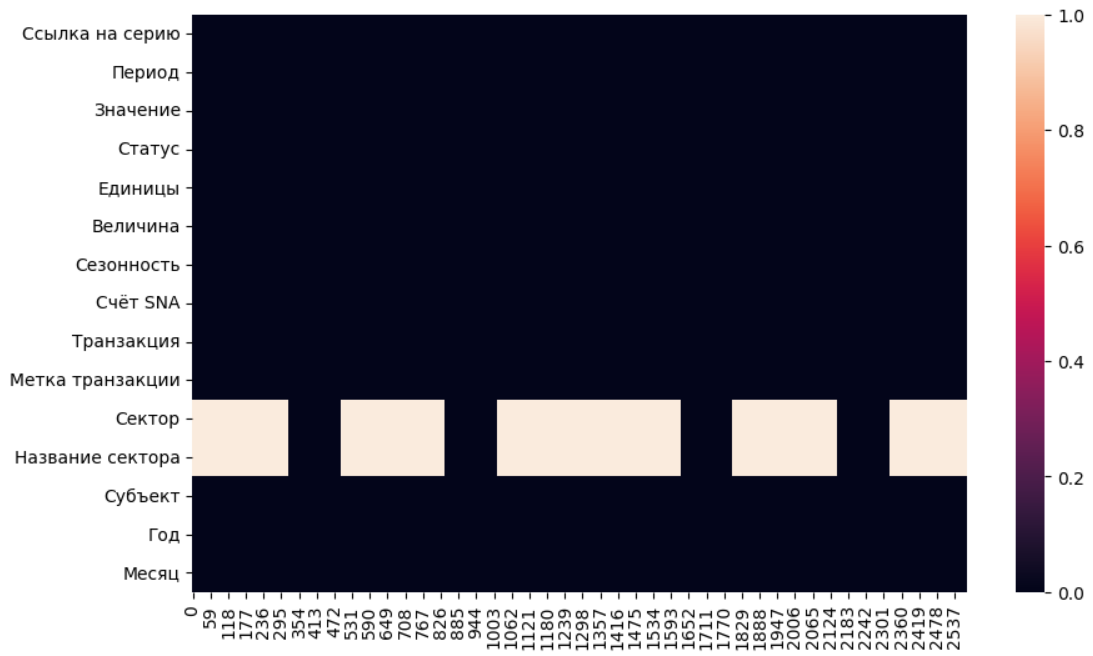


Рисунок 7 – Тепловая карта без удалённых столбцов

Теперь нужно решить, можно ли как-то восстановить недостающие значения у столбцов “Сектор” и “Название сектора”.

Прежде всего, определим уникальные значения в этих столбцах:

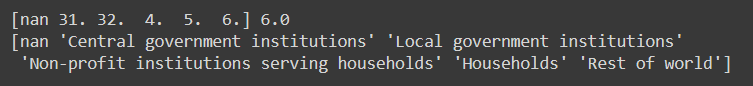


Рисунок 8 – Уникальные значения в столбцах “Сектор” и “Название сектора”

В каждом столбце ровно 5 уникальных значений. Одно уникальное значение столбца “Сектор” соответствует одному уникальному значению столбца “Название сектора”.

Это можно наблюдать, если вывести медиану группировки столбца “Название сектора” по столбцу “Сектор”:

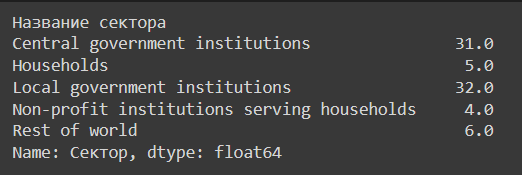


Рисунок 9 – Медиана группировки значений двух столбцов

В целом, на рисунке 9 представлено соответствие значений из столбца “Название сектора” и “Сектор”:

1. Central government institutions – 31 сектор
2. Households – 5 сектор
3. Local government instructions – 32 сектор
4. Non-profit institutions serving households – 4 сектор
5. Rest of world – 6 сектор

Однако данной связи недостаточно, чтобы целиком и полностью восстановить недостающие данные о секторах и их именах, поскольку там, где в столбце “Сектор” есть пропуски, пропуски имеет и столбец “Название сектора”. А это единственный столбец, по которому мы можем восстановить данные в столбце “Сектор”, поскольку с другими столбцами данные столбцы представляется сложным связать однозначно.

Например, можно предположить, что по столбцу “Транзакция” можно восстановить значения столбца “Сектор”, т.к. по транзакциям можно узнать из какого они приходят сектора.

Однако, медианные значения группировки транзакций по секторам не дадут никакой полезной информации (см. рис. 10).

По этой причине столбец “Сектор” и “Название сектора” могут быть удалены, т.к. не хватает данных для их восстановления. Это лишь один из вариантов.

Помимо удаления есть ещё один вариант – в столбце “Сектор” заполнить все отсутствующие значения медианным значением этого столбца, а в столбце “Название сектора” можно поставить всем отсутствующим значениям значение “Rest of world”, которое соответствует сектору 6.

Будем считать, что “экономическая активность сектора” зависит от частоты его встречаемости в столбцах “Значение” и “Название сектора”, а также от общей суммы в столбце “Значение” (чем больше сумма – тем больше экономическая активность).

Rest of world – данный сектор переводится как “Остальной мир”, т.е. в принципе, мы можем сделать так, чтобы данный сектор проявлял наибольшую экономическу активность в нашем наборе данных. Однако, это может значительно повлиять на наше экономическое представление о текущем наборе данных. Могут быть сделаны не правильные выводы по типу “… весь остальной мир проявил наибольшую финансовую активность …”. По такому принципу можно подставить и 5, и 4 и 31 сектор, но суть будет одна и таже – экономическое представление о данном наборе значительно исказится.

Для того, чтобы узнать, исказится ли наше экономическое представление о данном наборе данных, узнаем распределение значений сектора (см. рис. 11).

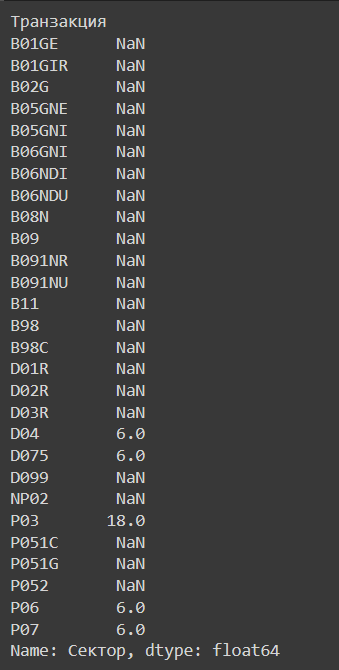


Рисунок 10 – Медианные значения группировки транзакций по секторам

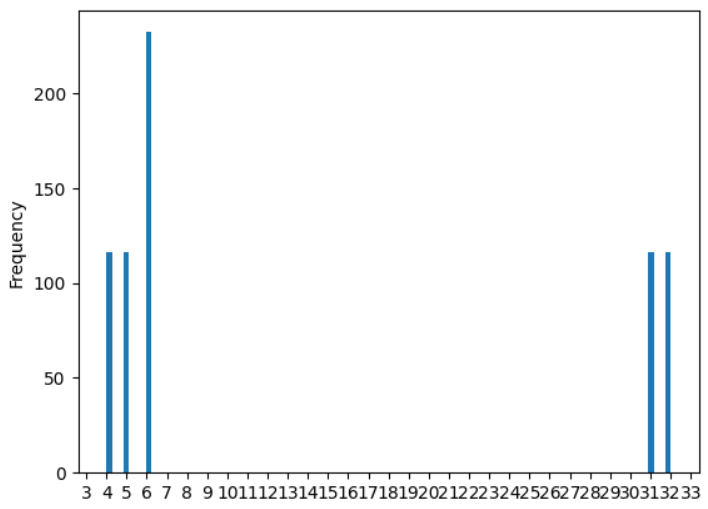


Рисунок 11 – Распределение значений столбца “Сектор”

Как видно из рисунка 11, сектор 6 (весь остальной мир) проявляет наибольшую экономическую активность. То есть, так или иначе в существующем наборе данных мы можем сказать, что весь остальной мир проявляет наибольшую экономическую активность, поскольку данное значение чаще всего встречается в таблице.

Рассмотрим сумму значений группировки столбца “Название сектора” по столбцу “Значение”, чтобы убедиться действительно ли остальной мир проявляет наибольшую экономическую активность.

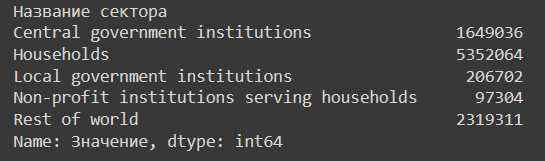


Рисунок 12 – Сумма значений для каждого отдельного сектора

Весь остальной мир (rest of world) проявляет хорошую экономическую активность в данном наборе данных, поскольку общая сумма значений из столбца “Значение” для данного сектора составляет 2319311. Однако, это не лучший показатель.

Лучший показатель будет у сектора с названием Households, чья сумма равна 5352064.

Таким образом нельзя сказать, что весь остальной мир проявляет наибольшую экономическую активность, поэтому столбец “Сектор” и “Наименование сектора” должны быть удалены.

data.drop('Сектор', axis=1 , inplace=True)

data.drop('Наименование сектора', axis=1, inplace=True)

Теперь определим распределение данных по всей таблице. Поскольку в таблице присутствуют разные типы данных, то будем разделять распределение данных по числовым и категориальным (для строк) признакам.

На рисунке 13 представлено распределение по числовым признакам.

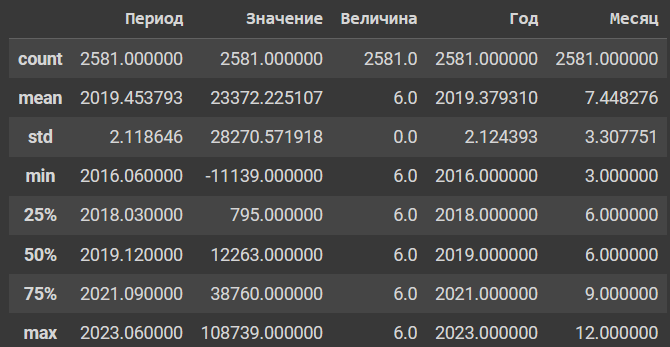


Рисунок 13 – Распределение по числовым признакам

Значение count всегда будет равно числу строк таблицы, поэтому и здесь оно равно 2581.

По значениям min и max можно определить, что датасет охватывает период с 2016 по 2023 год.

По столбцу “Значение” можно понять, что наибольшее значение было получени в 2021 году и оно составляло 38760, а наименьшее – в 2016, которое перерасло в убыток -11139. Таким образом можно увидеть рост экономической активности условной компании.

А вот столбец “Величина” можно удалить из рассматриваемой таблицы, поскольку в данном столбце все значения одинаковы, на что указывает и среднеквадратическое отклонение (std), которое равно 0 (нет никаких отклонений). Действительно, все значения в столбце “Величина” равны 6.

Теперь рассмотрим распределение данных по категориальным признакам (см. рис. 14).

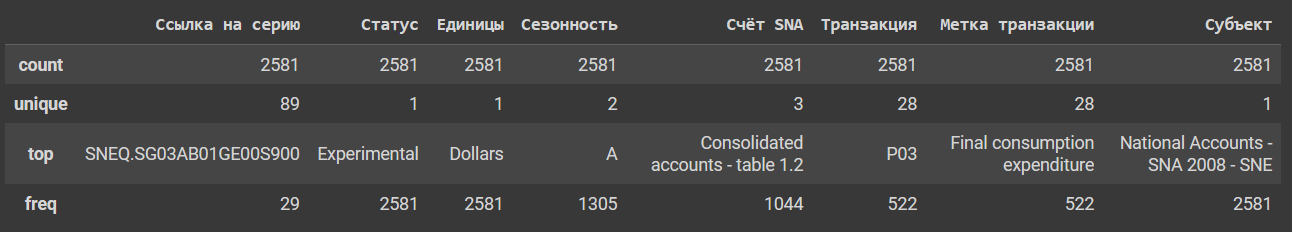


Рисунок 14 – Распределение по категориальным признакам

По категориальным признакам можно судить о бесполезных, с точки зрения анализа, столбцах.

Например, в столбце “Статуса” во всей 2581-ой строке всегда будет одно и тоже значение – Experimental. Аналогично и столбцу “Единицы” и “Субъект” – в этих столбцах находится только одно значение.

Именно поэтому данные столбцы можно убрать из дальнейшего анализа, т.к. они просто бесполезны и не могут повлиять на какие-либо выводы.

Для удаления всех столбцов, в которых все данные одинаковые, можно использовать следующий программный код:

# Подсчёт количества уникальных элементов в столбцах

nunique = data.nunique()

# Определение индексов столбцов, которые имеют только 1 уникальный элемент

cols\_to\_drop = nunique[nunique == 1].index

# Удаление столбцов, в которых все значения одинаковые

data = data.drop(cols\_to\_drop, axis=1)

Таким образом, у нас имеется следующий обработанный набор данных:

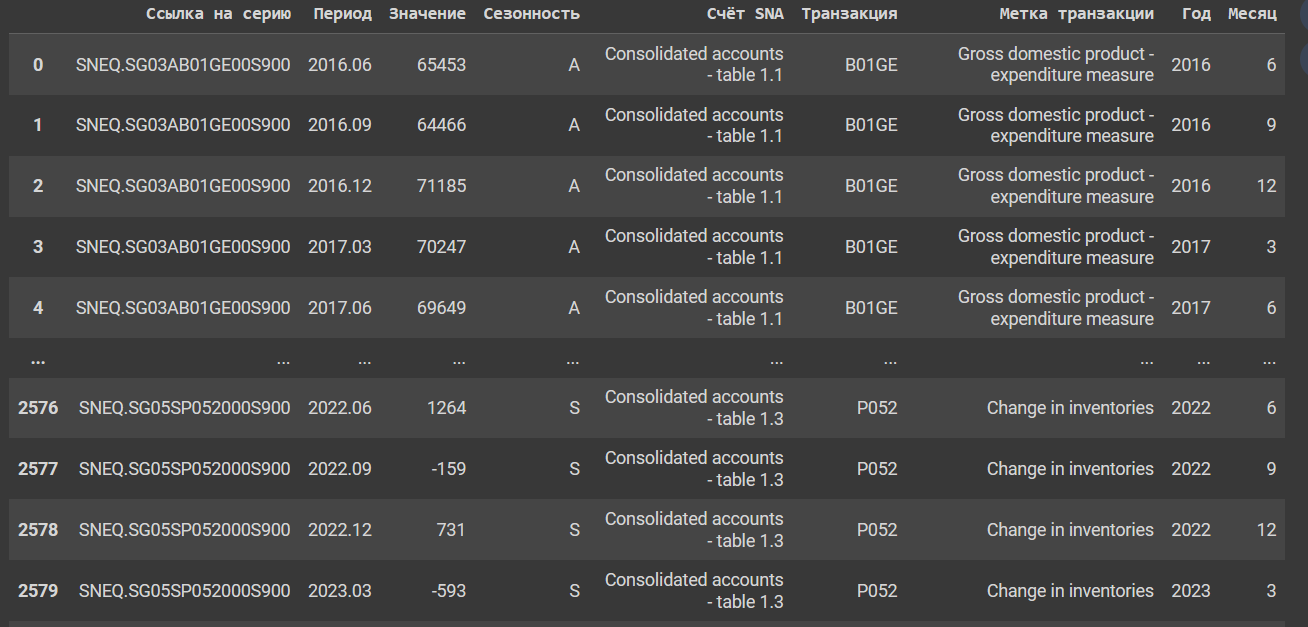


Рисунок 15 – Обработанный набор данных

Теперь можно выяснить, есть ли в данном наборе данных выбросы (для числовых признаков).

Для начала сгенерируем диаграмму размаха (“ящик с усами”), с помощью следующего программного кода:

numeric\_col = ['Период', 'Значение', 'Год', 'Месяц']

data.boxplot(numeric\_col)

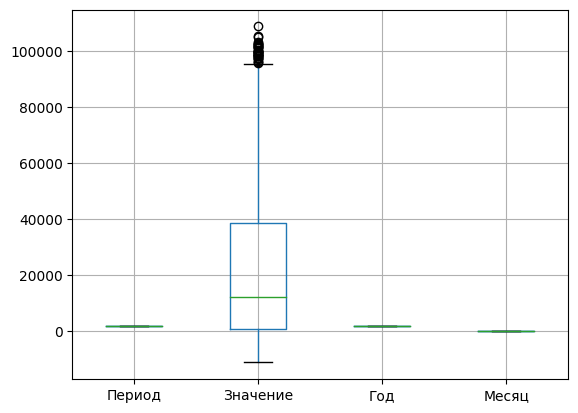


Рисунок 16 – Диаграмма размаха для обработанного набора данных

Как видно выше (см. рис. 16), столбец “Значение” содержит выбросы, которые лежат над нижней границей. В других рассматриваемых столбцах выбросов нет, поэтому необходимо убрать выбросы только в столбце “Значение”.

Для лечения выбросов будем использовать фильтр Хэмпеля.

Опишем функцию, которая заменяет на nan все значения, у которых разница с медианой больше, чем три медианных абсолютных отклонения:

# Фильтр Хэмпеля

def hampel(vals\_orig, x):

# Копирование набора данных

vals = vals\_orig.copy()

# Вычисление абсолютной разницы между медианными значениями и фактическими

difference = np.abs(vals[x].median() - vals[x])

# Формирование медианного абсолютного отклонения

median\_abs\_deviation = difference.median()

# Определение условия "больше 3-х медианных абсолютных отклонения"

threshold = 3 \* median\_abs\_deviation

# Формирование условия для индексации в наборе данных

outlier\_idx = difference > threshold

# Заполнение всех удовлетворяющих условию значений на nan

vals[x][outlier\_idx] = np.nan

return(vals)

Далее, необходимо использовать фильтр Хэмпеля следующим образом:

# Применение фильтра Хэмпеля

copy = hampel(data, "Значение")

# Замена всех значений nan в наборе данных на умноженное на 8 среднее значение

copy["Значение"] = copy["Значение"].fillna(8 \* copy["Значение"].mean())

Теперь визуализируем диаграмму размаха с новыми данными:

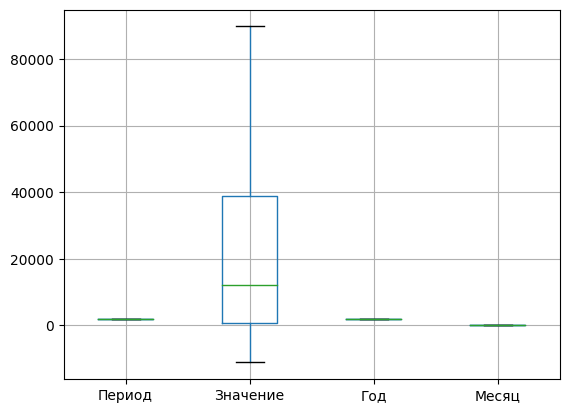


Рисунок 17 – Диаграмма размаха без выбросов

Таким образом с помощью фильтра Хэмпеля были вылечены все выбросы на обработанном наборе данных. Теперь можно приступать к одномерному и двумерному анализу данных.

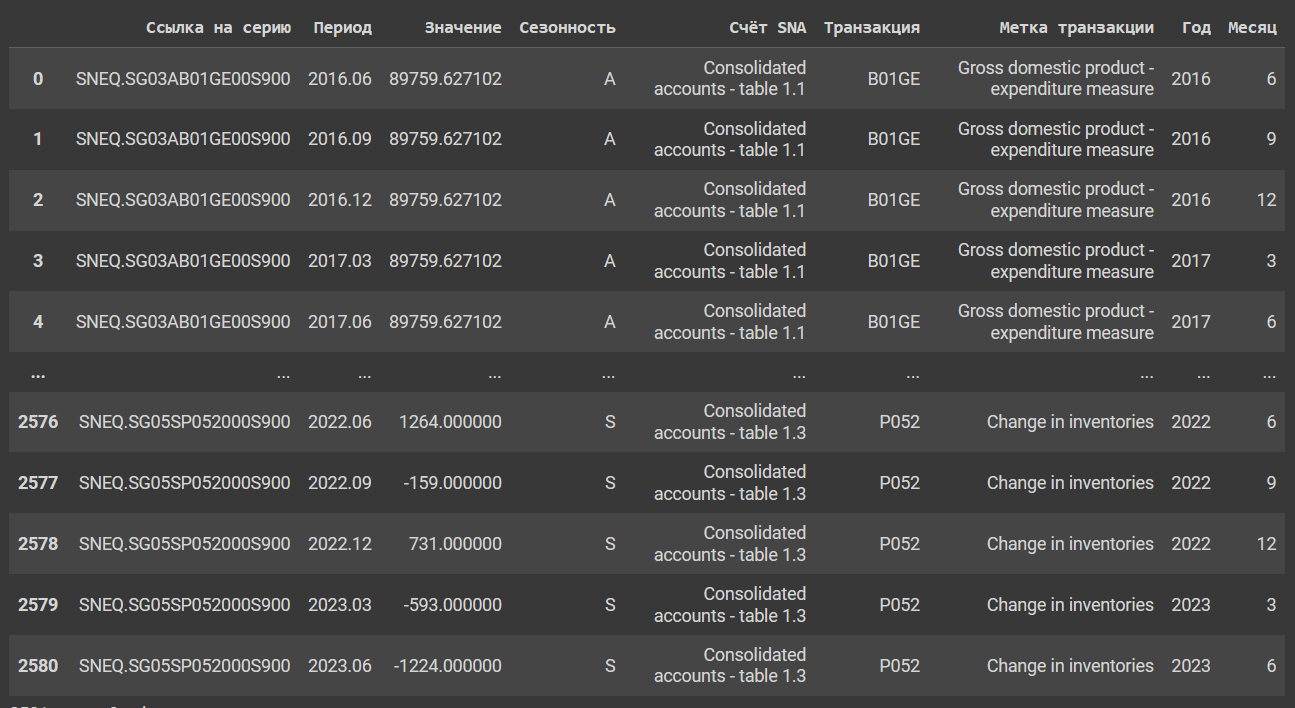


Рисунок 18 – Обработанный набор данных без выбросов

Одномерный анализ данных

Сначала исследуем соотношение сезонов A (autumn – осень) и S (spring – весна) в наборе данных:

sns.set\_style('darkgrid')

sns.countplot(y='Сезонность', data=data, palette='colorblind')

plt.xlabel('Количество')

plt.ylabel('Сезонность')

plt.show()

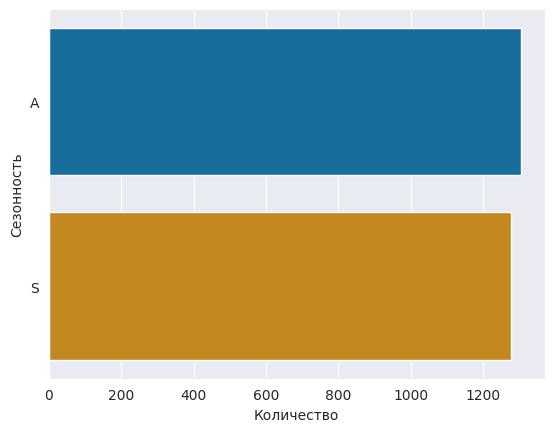


Рисунок 19 – Соотношение сезона осени и весны

Судя по графику seaborn, наиболее активным сезоном, с точки зрения осуществления финансовых операций, является осень (A).

Для более точной оценки воспользуемся следующим программным кодом:

A = len(data[data['Сезонность'] == 'A'])

S = len(data[data['Сезонность'] == 'S'])

print("Всего A: ", A, '\nВсего S: ', S)



Рисунок 20 – Соотношение сезонов

Более точными данными (см. рис. 20) подтверждаем, что наиболее активным сезоном является осень.

Теперь посмотрим на распределение финансовых операций по годам. График строится с помощью следующего программного кода:

sns.set\_style('darkgrid')

sns.countplot(x='Год', data=data, palette='colorblind')

plt.xlabel('Год')

plt.ylabel('Количество')

plt.show()

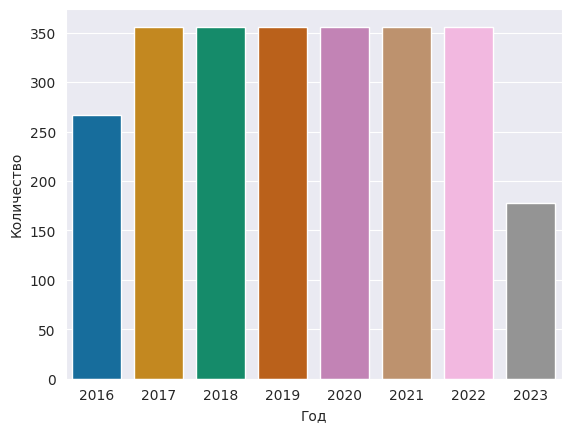


Рисунок 21 – Распределение количества финансовых операций по годам

Это распределение говорит нам о том, что в 2016 году количество финансовых операций было меньше, чем в 2017 году, а с 2017 по 2022 год количество всех финансовых операций было одинаковым, а вот в 2023 году финансовых операций было меньше всего за рассматриваемый период.

Рассмотрим также распределение финансовой активности отдельных счётов SNA:

sns.set\_style('darkgrid')

sns.countplot(y='Счёт SNA', data=data, palette='colorblind')

plt.xlabel('Количество')

plt.ylabel('Счёт SNA')

plt.show()

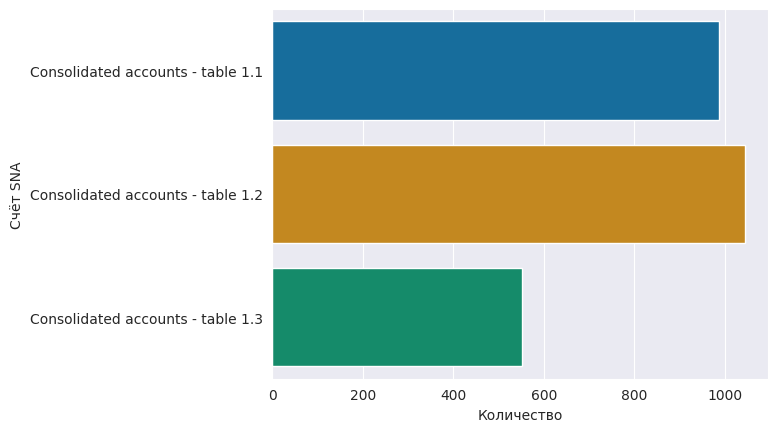


Рисунок 22 – Распределение финансовой активности счётов SNA

Данное распределение говорит нам о том, что наиболее активным счётом SNA является “Consolidated accounts – table 1.2”, а наименее активным – “Consolidated accounts – table 1.3”.

И, наконец, рассмотрим распределения количества транзакций, чтобы понять, какая из них наиболее часто используется, а какая наименее часто:

sns.set\_style('darkgrid')

sns.countplot(y='Транзакция', data=data, palette='colorblind')

plt.xlabel('Количество')

plt.ylabel('Транзакция')

plt.show()

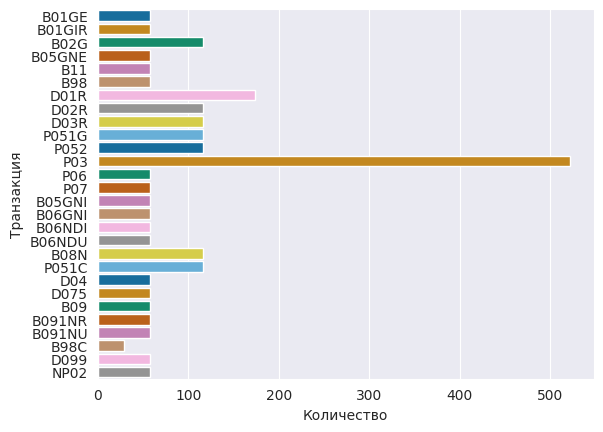


Рисунок 23 – Распределение по количеству транзакций

Исходя из диаграммы на рисунке 23 получаем, что самая популярная транзакция – P03, а самая не популярная – B98C.

Двумерный анализ данных

Теперь необходимо узнать, есть ли какая-либо корреляция (зависимость) между отдельными функциями (столбцами), которые нам необходимо учитывать.

Построим зависимость между отдельными годами и значениями, которые в эти года были получены, учитывая сезонность.

sns.set\_style('darkgrid')

plt.title('Зависимость между годами и значениями', size=16)

plt.xlabel('Год', size=12)

plt.ylabel('Значение', size=12)

sns.scatterplot(x='Год', y='Значение', data=data,

hue='Сезонность', hue\_order=['A', 'S'], edgecolor='black', palette='cubehelix')

plt.show()

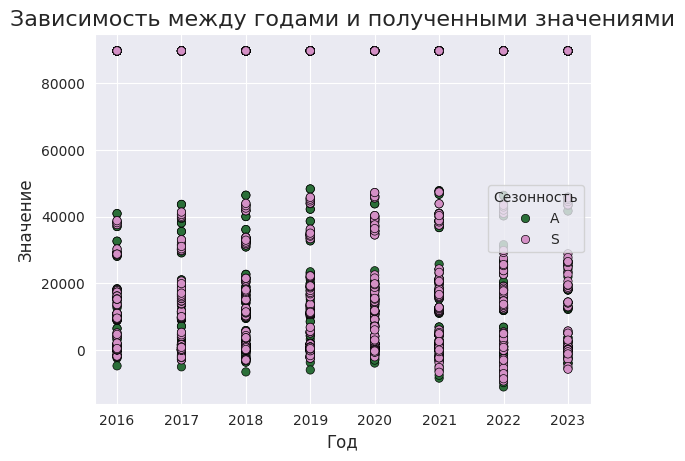


Рисунок 24 – Зависимость между годами и полученными значениями

На диаграмме отчётливо видна дискретная природа распределения значений по годам. Для каждого года сформирована своя линия значений, которые в эти года были получены. На диаграмме также видно распределение сезонности и результат лечения выбросов (все аномальные значения “приближены” к нормальным, для данного набора данных, максимальным значениям).

Рассеяние предполагает высокую степень корреляции между значениями по разным годам. Судя по полученному графику значения по всем годам довольно хорошо разбросаны, но в 2022 году разброс достигает своего пика. В этом можно дополнительно убедиться с помощью следующего программного кода:

years = [2016, 2017, 2018, 2019, 2020, 2021, 2022, 2023]

for year in years:

std = data[data["Год"] == year]["Значение"].std()

print("Year: ", year, " - ", "Std: ", std)

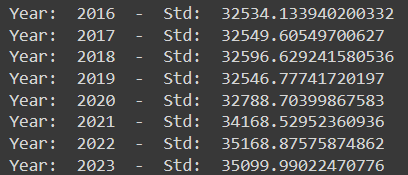


Рисунок 25 – Стандартные отклонения по годам

На рисунке 25 представлены стандартные отклонения по годам. Из него можно увидеть, что в 2022 году отклонение наиболее максимальное и равно 35168 (округлённо).

Теперь рассмотрим зависимость между конкретными транзакциями и полученными значениями:

sns.set\_style('darkgrid')

plt.title('Зависимость между транзакциями и полученными значениями', size=16)

plt.xlabel('Значение', size=12)

plt.ylabel('Транзакция', size=12)

sns.scatterplot(x='Значение', y='Транзакция', data=data,

hue='Сезонность', hue\_order=['A', 'S'], edgecolor='black', palette='cubehelix')

plt.show()

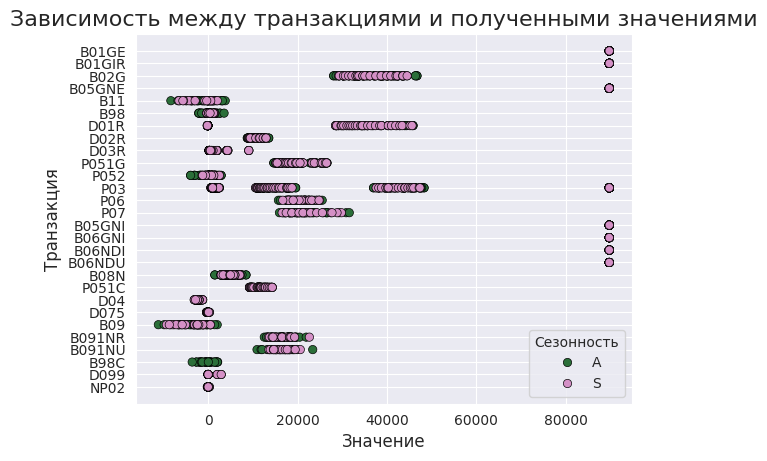


Рисунок 26 – Зависимость между транзакциями и полученными значениями

На рисунке 26 видно, что у транзакции P03 разброс значений больше, чем у остальных транзакций.

Для подтверждения визуальных результатов воспользуемся следующим программным кодом:

transactions = data["Транзакция"].unique()

for transaction in transactions:

std = data[data["Транзакция"] == transaction]["Значение"].std()

print("Transaction: ", transaction, " - ", "Std: ", std)

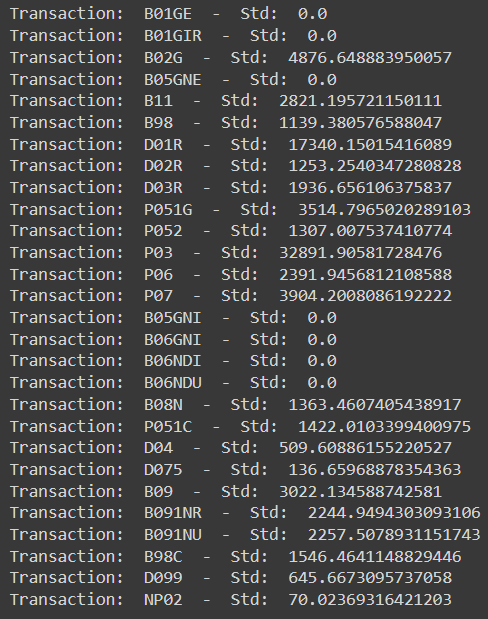


Рисунок 27 – Стандартные отклонения значений по конкретным транзакциям

Можно отметить, что в результате лечения выбросов в наборе данных встретилась одна особенность – некоторые стандартные отклонения по значениям транзакций равны 0. Это такие транзакции, как B01GE, B01GIR, B05GNE, B05GNI, B06GNI, B06NDI и B06NDU.

Интегральная оценка – это обобщённый показатель, рассчитанный на основе значений измерений и характеризующий конкретный набор данных.

Определим, какое из измерений больше всего влияет на её значение.

total\_marks = ((data['Год'] + data['Значение'])/200)\*100

data['Интегральная оценка'] = total\_marks

kde\_data = data[ ['Год','Значение','Интегральная оценка'] ]

sns.set\_style("darkgrid")

sns.kdeplot(data=kde\_data,shade=True, palette='colorblind')

plt.show()

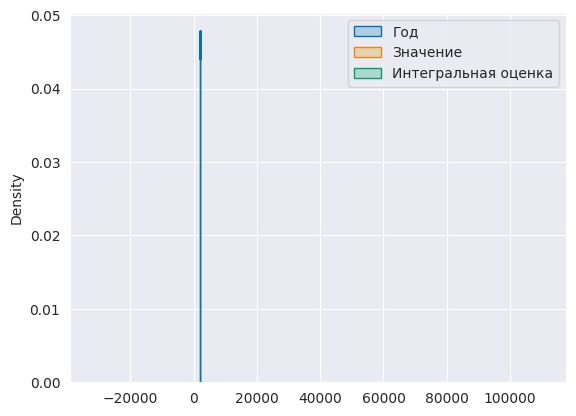


Рисунок 28 – Интегральная оценка

Из рисунка 28 получаем, что в принципе на интегральную оценку в большей степени влияет год, в рамках которого получено какое-либо значение. К сожалению, структура рассматриваемого набор данных не позволяет более детально оценить влияние других столбцов на интегральную оценку, т.к. численных “полезных” столбцов немного и есть ограничение в составлении корреляции между двумя столбцами.

Выводы

В рамках решения данной лабораторной работы были изучены основные этапы разведочного анализа данных (EDA) с применением различных методов из различных библиотек для языка программирования Python для набора данных по выбранному варианту.

В ходе решения задач, был загружен набор данных с заполнением всех пустых значений и удалением не информативных строк таблицы. Также была осуществлена фильтрация данных и произведено “лечение” набора данных от выбросов с использованием фильтра Хэмпеля. Был проведён анализ распределения переменных в наборе данных с визуализацией в виде тепловых карт.

Был проведён одномерный анализ для изучения категориальных переменных, а также двумерный анализ между переменными в наборе данных с использованием методов визуализации библиотеки seaborn.

Ссылка на блокнот: <https://github.com/DanSoW/INRTU/blob/main/data-processing-analysis-and-visualization-technologies/lab2/lab2.ipynb>

**Список использованных источников**

1. Погружение в аналитику данных. Маунт Джордж: Пер. с англ. — СПб.: БХВПетербург, 2023.

2. Python — это просто. Пошаговое руководство по программированию и анализу данных. Нисчал Н.: Пер. с англ. — СПб.: БХВ-Петербург, 2022.

3. Наглядная статистика. Используем R! Шипунов А. Б., Балдин E. М., Волкова П. А., Коробейников А. И., Назарова С. А., Петров С. В., Суфиянов В. Г. - М.: ДМК Пресс, 2012.

4. Разведочный анализ (EDA) (URL: https://habr.com/ru/companies/otus/articles/752434/).

5. Разведочный анализ данных в Python: руководство для новичков на 2021 год (URL: https://waksoft.susu.ru/2021/07/23/issledovatelskij-analiz-dannyh-v-python-rukovodstvo-dlya-novichkov-na-2021-god/).