МИНОБРНАУКИ РО ССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра информационных систем

ОТЧЕТ

по лабораторной работе №1
по дисциплине «Машинное обучение»
Тема: Исследование набора данных

Студентка гр. 2373 _____ Шавлохова А.А.

Преподаватель _____ Татчина Я.А.

Санкт-Петербург
2024

ЗАДАНИЕ

НА ИССЛЕДОВАНИЕ НАБОРА ДАННЫХ ЛАБОРАТОРНУЮ

Студентка Шавлохова А.А.

Группа 2373

Тема лабораторной: Исследование набора данных

Задание на лабораторную:

1. Создать Jupyter Notebook, переименовать его «Lab 1, № Группы, ФИО»
1. Выбрать исследуемый датасет.

2. Для каждого датасета представить краткое описание в Jupyter Notebook:
    - предметная область, источник данных, характер данных (реальные или имитационные)
    - какие атрибуты представлены в датасете, их тип (числовой, строковый (категории)), что они обозначают
    - есть ли описание задачи анализа, если есть - представить
3. Для каждого атрибута определить:
    - среднее значение, ско
    - построить гистограмму распределения значений, определить есть ли выбросы
    - есть ли пропущенные значение, сколько
    - предложить вариант обработки пропущенных значений
4. Определить корреляцию между параметрами
    - какие атрибуты высококоррелированы, определить характер корреляции
    - какие атрибуты не имеют корреляцию
    - построить графики рассеивания
    - проанализировать полученные результаты.

Написать отчет о проделанной работе.

1 Часть

Для выполнения данной практической работы был выбран датасет с сайта: https://www.kaggle.com/datasets, который был предложен в Задании 1.

Из списка датасетов был выбран «Most Popular Programming Languages 2004-2024» (https://www.kaggle.com/datasets/muhammadroshaanriaz/most-popular-programming-languages-2004-2024). Каждая строка датасета представляет данные за определенный месяц, начиная с января 2004 года и заканчивая сентябрем 2024 года, отслеживая тенденции популярности представленных языков программирования во всем мире. В выбранном датасете представлены исключительно числовые данные (данные представлены в процентах).

Этот набор данных содержит следующие атрибуты:

- *Месяц*: дата (в формате год-месяц), когда данные были записаны.
- *Python Worldwide(%)*: процент глобальной популярности Python за этот месяц.
- *JavaScript Worldwide(%)*: процент глобальной популярности JavaScript.
- *Java Worldwide(%):* процент глобальной популярности Java.
- *C# Worldwide(%):* процент глобальной популярности C#.
- *PhP Worldwide(%):* процент глобальной популярности PhP.
- *Flutter Worldwide(%):* процент глобальной популярности Flutter.
- *React Worldwide(%):* процент глобальной популярности React.
- *Swift Worldwide(%):* процент глобальной популярности Swift.
- *TypeScript Worldwide(%):* процент глобальной популярности TypeScript.
- *Matlab Worldwide(%):* процент глобальной популярности Matlab.

2 Часть

1. Средние значения и СКО для атрибутов были определены с помощью функций mean() и std().

```
AvgPython = np.array(df['Python Worldwide(%)'])
AvgPythonMean = AvgPython.mean()
AvgPythonSko = np.std(AvgPython)
```

Рис. 1.1 – Пример кода для определения среднего значения и СКО для Python Worldwide(%)

Средние значения для атрибутов:

- Python Worldwide(%): 41.68
- JavaScript Worldwide(%): 43.96
- Java Worldwide(%): 35.996
- C# Worldwide(%): 59.63
- PhP Worldwide(%): 37.48
- Flutter Worldwide(%): 22.64
- React Worldwide(%): 25.88
- Swift Worldwide(%): 30.68
- TypeScript Worldwide(%): 23.41
- Matlab Worldwide(%): 59.29

СКО для атрибутов:

- Python Worldwide(%): 23.06
- JavaScript Worldwide(%): 15.99
- Java Worldwide(%): 21.85
- C# Worldwide(%): 20.29
- PhP Worldwide(%): 22.95
- Flutter Worldwide(%): 28.31
- React Worldwide(%): 32.45
- Swift Worldwide(%): 15.298
- TypeScript Worldwide(%): 28.55
- Matlab Worldwide(%): 13.26

2. Гистограммы распределения значений были построены с помощью функции .hist() и изменена с помощью функций: .grid(True); .title()

```
plt.hist(df['Python Worldwide(%)'], bins=20)
plt.grid(True)
plt.title('Гистограмма распределения значений Python')
```

Рис. 1.2 – Пример кода для построения гистограммы распределения значений для Python Worldwide(%)

Наличие выбросов на гистограммах распределения значений определялось с помощью вызова функции detect_outliers_iqr(). Функция искала выбросы по алгоритму Межквартильного размаха:

1. Определение квартилей

Квартиль — это значение, которое разделяет набор данных на четыре равные части.

Первый квартиль(Q1) – это значение, ниже которого находится 25% данных.

Третий квартиль (Q3) – это значение, ниже которого находится 75% данных.

2. Вычисление IQR

Межквартильный размах (IQR) – это разница между третьим и первым квартилями.

IQR = Q3 - Q1

3. Определение границ для выявления выбросов

LowerBound = Q1 - 1.5 * IQR

UpperBound = Q3 + 1.5 * IQR

4. Поиск выбросов

Значения, которые меньше нижней границы или больше верхней границы.

Гистограммы для каждого атрибута и наличие или отсутствие выбросов:

```
Выбросов не обнаружено.
Количество выбросов, обнаруженных по методу IQR:  0
Количество пропущенных значений: 0
```

Рис. 1.3 – Гистограмма распределения значений и отсутствие выбросов для Python Worldwide(%)

```
Выбросы:
0      98
1      98
2     100
3      98
4      91
5      94
6      93
7      91
8      91
9      78
10     83
11     82
12     75
13     81
14     75
15     75
16     77
17     80
18     78
19     74
20     71
21     74
22     71
26     71
Name: JavaScript Worldwide(%), dtype: int64
Количество выбросов, обнаруженных по методу IQR:  24
```

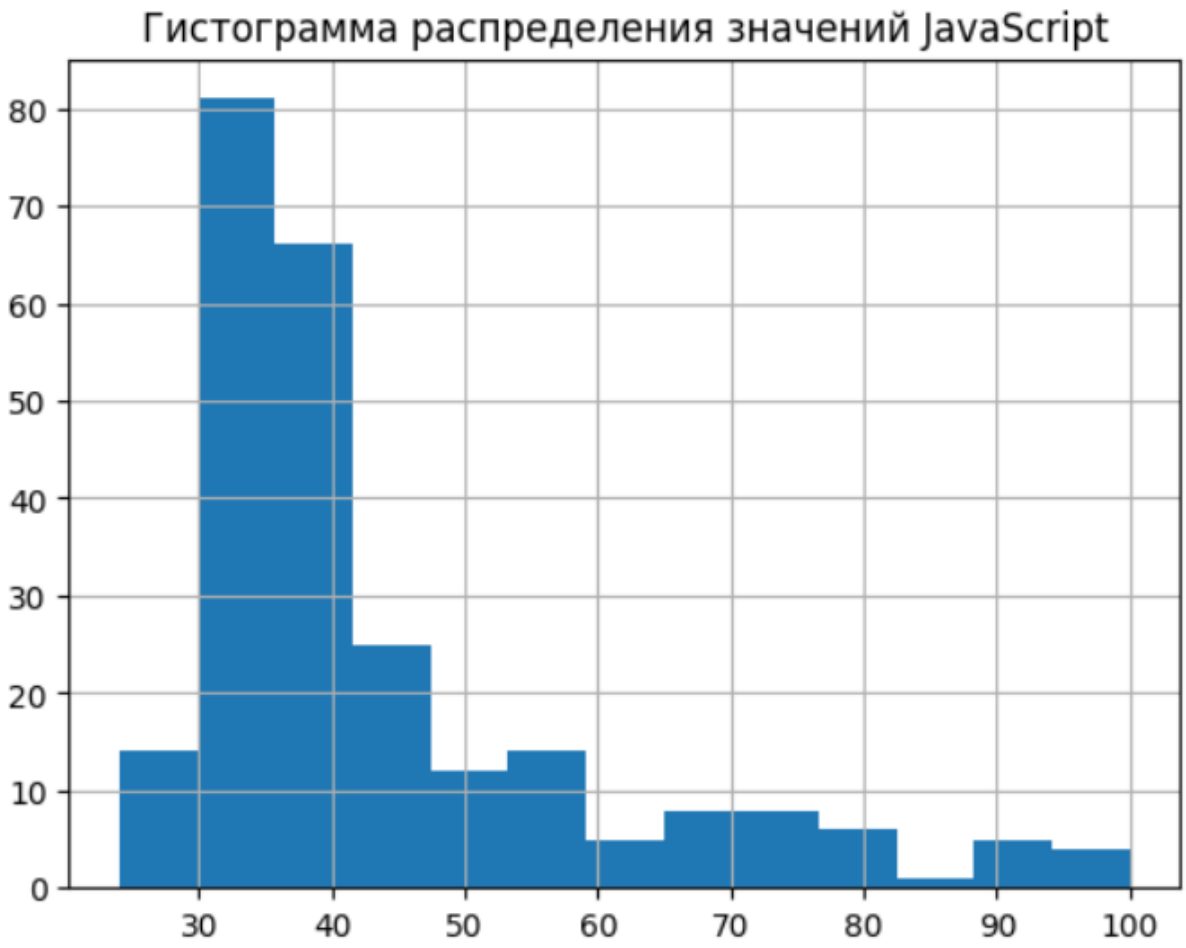

Рис. 1.4-5 – Гистограмма распределения значений и наличие выбросов для JavaScript Worldwide(%)

```
Выбросы:
0      96
1      97
2     100
3      97
4      99
5      89
6      89
7      89
8      91
9      93
10     90
13     89
Name: Java Worldwide(%), dtype: int64
Количество выбросов, обнаруженных по методу IQR:  12
Количество пропущенных значений: 0
```

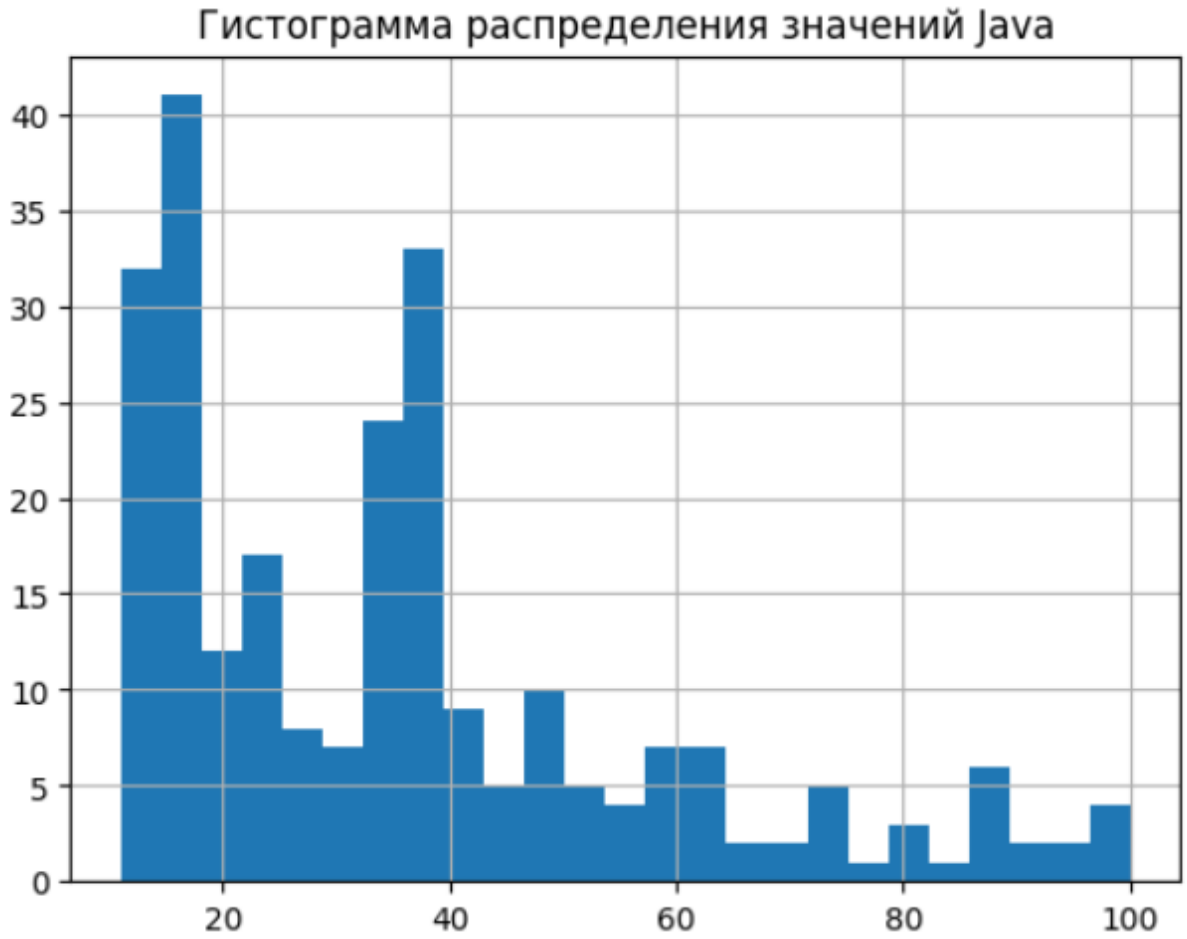

Рис. 1.6 – Гистограмма распределения значений и наличие выбросов для Java Worldwide(%)

```
Выбросов не обнаружено.
Количество выбросов, обнаруженных по методу IQR:  0
Количество пропущенных значений: 0
```

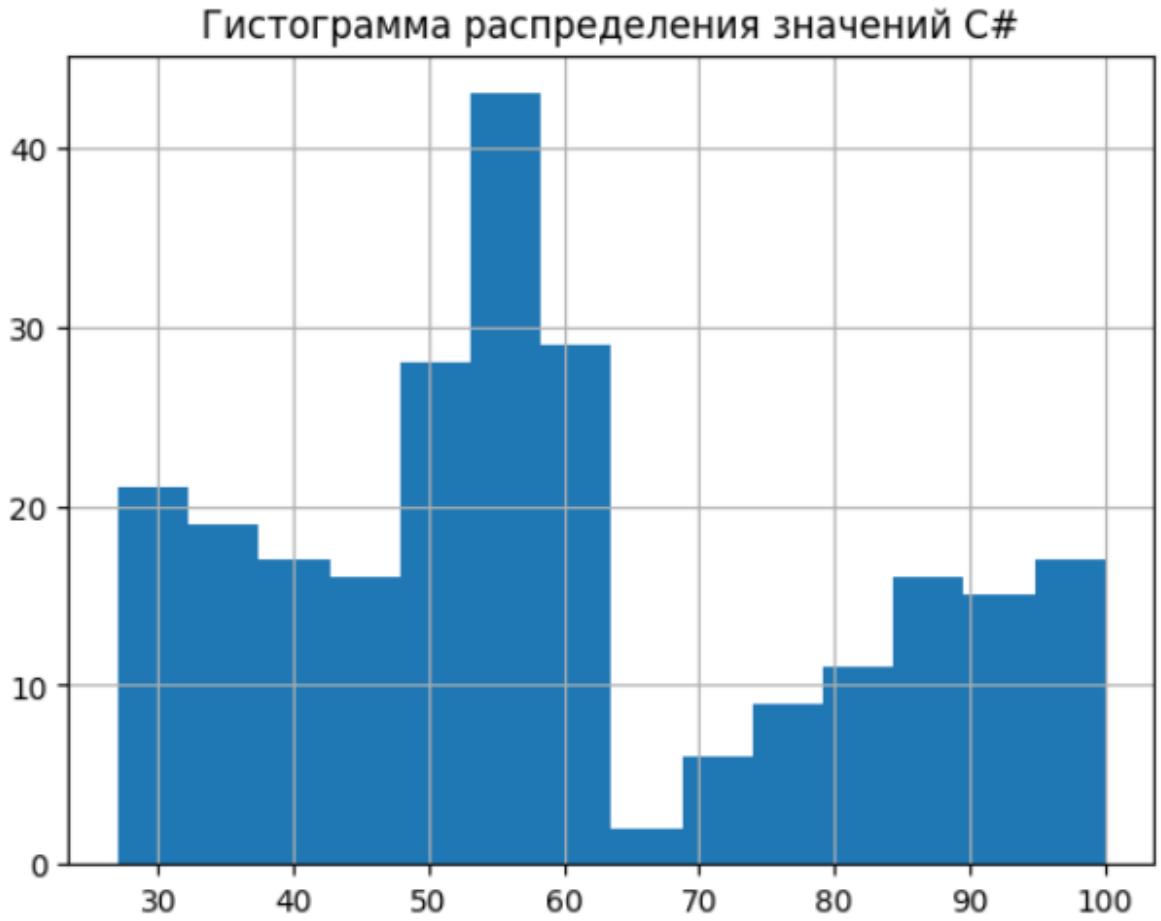

Рис. 1.7 – Гистограмма распределения значений и отсутствие выбросов для C# Worldwide(%)

```
Выбросы:
0     100
1      99
2      97
3     100
4      92
5      97
6      99
7      94
8      93
12     92
13     91
Name: PhP Worldwide(%), dtype: int64
Количество выбросов, обнаруженных по методу IQR:  11
Количество пропущенных значений: 0
```

Рис. 1.8 – Гистограмма распределения значений и наличие выбросов для PhP Worldwide(%)

```
Выбросы:
216     77
217     91
218     94
219     90
220     88
221     92
222     88
223     89
224     89
225     82
226     89
227     91
228    100
229     97
230     91
231     84
232     92
233     87
234     89
235     87
236     86
237     81
238     83
239     81
240     86
241     96
242     90
243     86
244     92
245     84
246     84
247     82
248     79
Name: Flutter Worldwide(%), dtype: int64
```

Рис. 1.9-10 – Гистограмма распределения значений и наличие выбросов для Flutter Worldwide(%)

```
Выбросов не обнаружено.
Количество выбросов, обнаруженных по методу IQR:  0
Количество пропущенных значений: 0
```

Рис. 1.11 – Гистограмма распределения значений и отсутствие выбросов для React Worldwide(%)

```
Выбросы:
68      61
130     63
233     66
234     70
235     61
236     74
237     78
238     77
239     66
240     65
241    100
242     59
243     77
244     61
245     61
248     59
Name: Swift Worldwide(%), dtype: int64
Количество выбросов, обнаруженных по методу IQR:  16
Количество пропущенных значений: 0
```

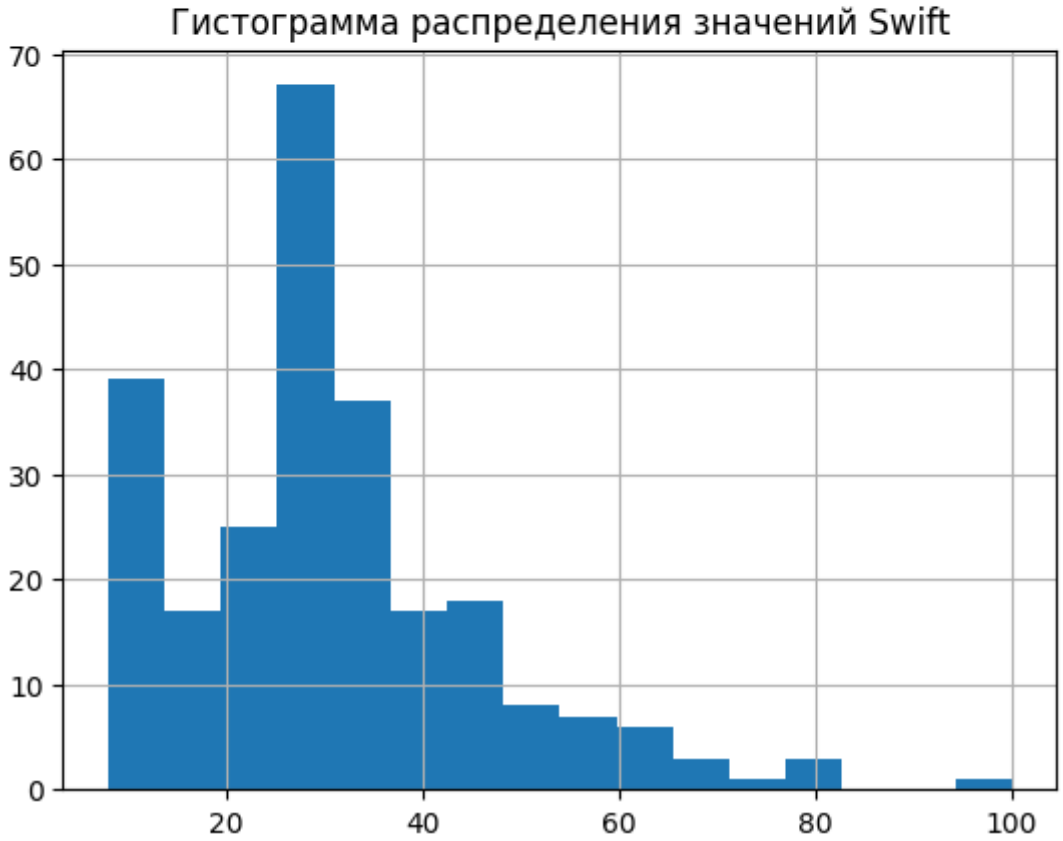

Рис. 1.12 – Гистограмма распределения значений и наличие выбросов для Swift Worldwide(%)

```
Выбросы:
219     92
222     97
223     91
226     91
228     91
229    100
230     97
Name: TypeScript Worldwide(%), dtype: int64
Количество выбросов, обнаруженных по методу IQR:  7
Количество пропущенных значений: 0
```

Рис. 1.13 – Гистограмма распределения значений и наличие выбросов для TypeScript Worldwide(%)

```
Выбросы:
1       91
2       99
3       95
9       94
10      92
219    100
225     93
226     91
Name: Matlab Worldwide(%), dtype: int64
Количество выбросов, обнаруженных по методу IQR:  8
Количество пропущенных значений: 0
```

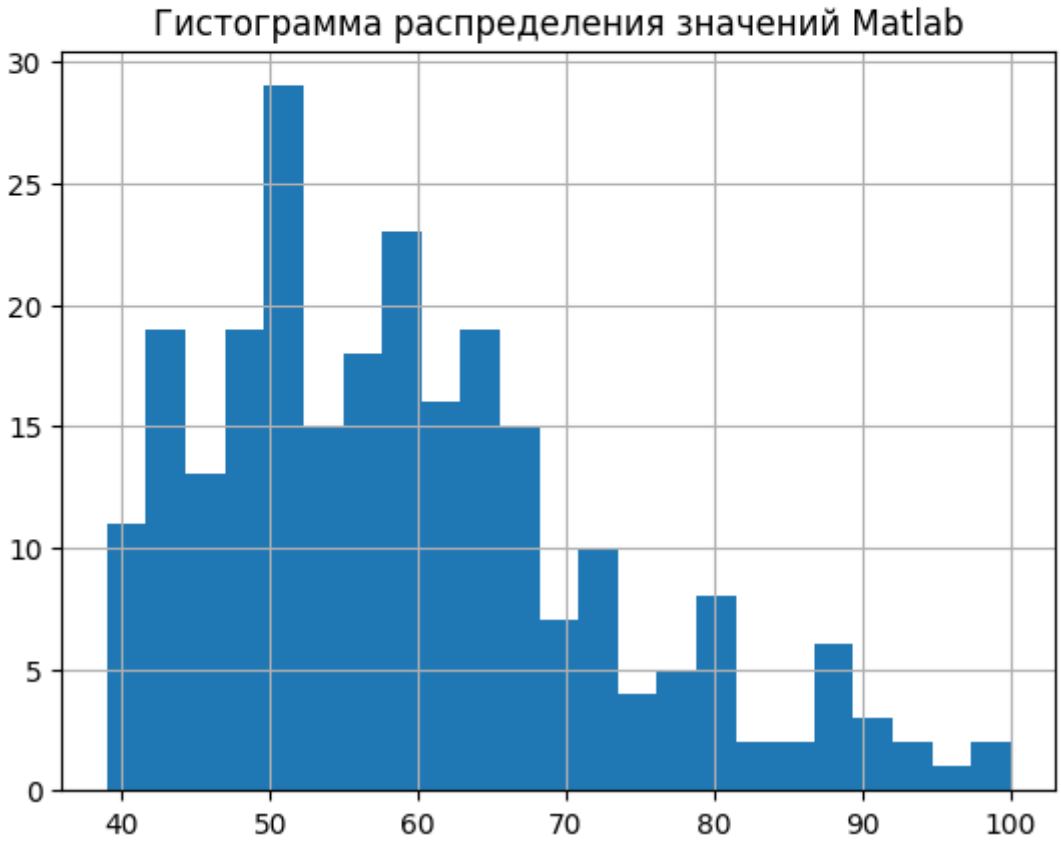

Рис. 1.14 – Гистограмма распределения значений и наличие выбросов для Matlab Worldwide(%)

Количество пропущенных значений определялось с помощью функций .isnull().sum())

```
print('Количество пропущенных значений:',df['Python Worldwide(%)'].isnull().sum())
```

Рис. 1.15 –Пример кода для поиска пропущенных значений для Python Worldwide(%)

Пропущенные значения для каждого атрибута:

- Python Worldwide(%): 0
- JavaScript Worldwide(%): 0
- Java Worldwide(%): 0
- C# Worldwide(%): 0
- PhP Worldwide(%): 0
- Flutter Worldwide(%): 0
- React Worldwide(%): 0
- Swift Worldwide(%): 0
- TypeScript Worldwide(%): 0
- Matlab Worldwide(%): 0

В данном варианте датасета нет пропущенных значений, следовательно нечего обрабатывать. Однако при наличии пропущенных значений их можно было бы удалить с помощью функции .dropna().

3 Часть

1. Корреляция между параметрами была определена с помощью функции correlation = df[].corr(df[]). Для упрощения понимания была создана и выведена в графическом виде матрица корреляции выбранных атрибутов.

```
correlation = df['Python Worldwide(%)'].corr(df['JavaScript Worldwide(%)'])
subset = df[['Python Worldwide(%)', 'JavaScript Worldwide(%)']]
correlation_matrix = subset.corr()
print(correlation_matrix)

sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt='.2f', square=True, linewidths=.5, cbar_kws={"shrink": .8})
plt.title('Корреляционная матрица')
plt.show()
```

Рис. 2.1 – Пример кода для расчета корреляции и вывода в матричном виде для атрибутов Python Worldwide(%) и JavaScript Worldwide(%)

Для данной практической работы были рассмотрены корреляции между следующими атрибутами: Python Worldwide(%) и JavaScript Worldwide(%); Java Worldwide(%) и C# Worldwide(%); PhP Worldwide(%) и Flutter Worldwide(%); React Worldwide(%) и Swift Worldwide(%); TypeScript Worldwide(%) и Matlab Worldwide(%).

Сильная положительная корреляция наблюдается у следующих атрибутов:

	Java Worldwide(%)	C# Worldwide(%)
Java Worldwide(%)	1.000000	0.913002
C# Worldwide(%)	0.913002	1.000000

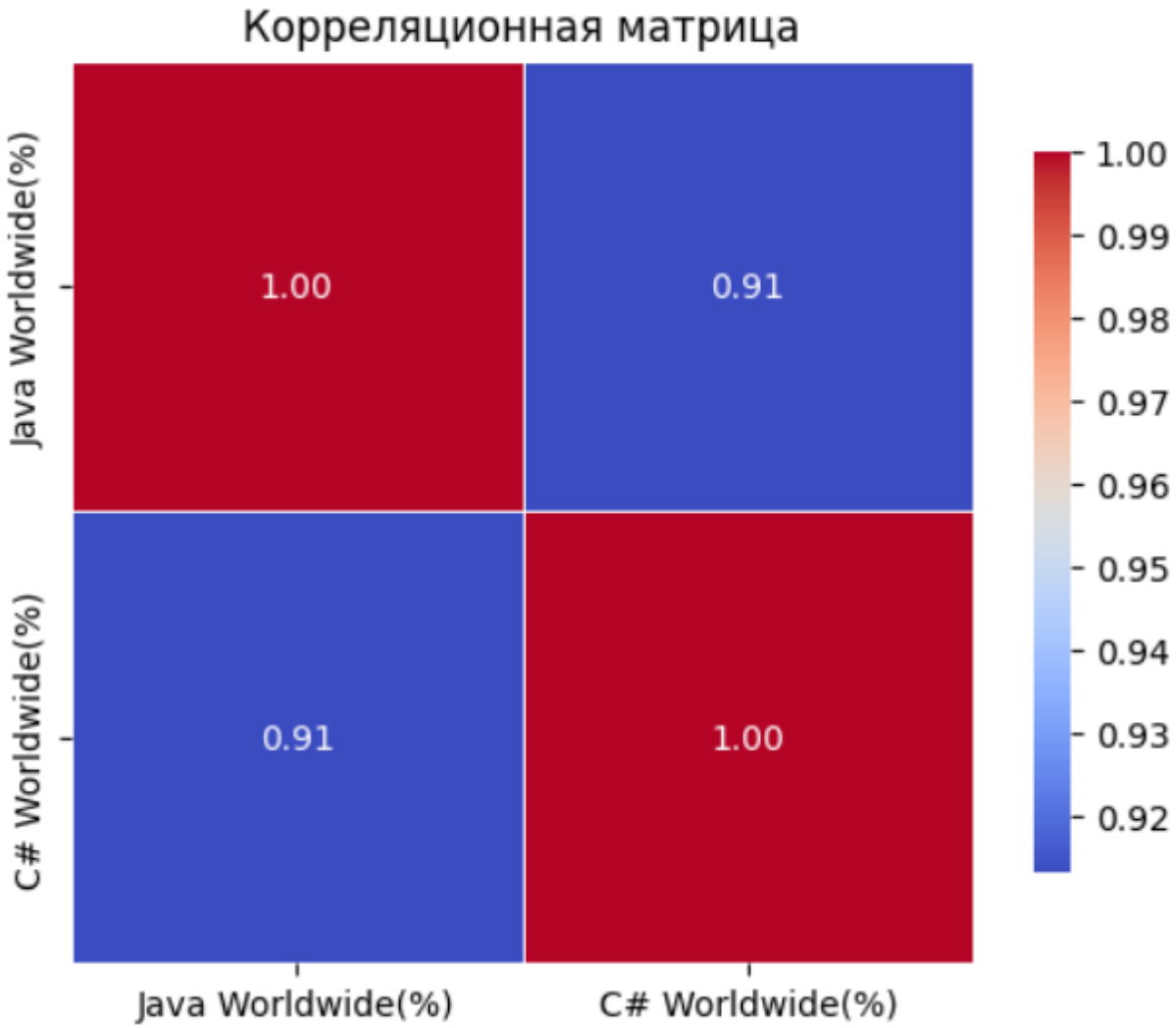

Рис. 2.2 – Корреляция между атрибутами: Java Worldwide(%) и C# Worldwide(%)

Представленная корреляция может говорить о сильной связи между этими языками программирования. Java и C# имеют много сходств и используются в разных сферах разработки, их популярность часто коррелирует в областях бизнес-программирования, веб-разработки и мобильных приложений.

Средняя положительная и отрицательная корреляция наблюдается у следующих атрибутов:

	PhP Worldwide(%)	Flutter Worldwide(%)
PhP Worldwide(%)	1.000000	-0.476221
Flutter Worldwide(%)	-0.476221	1.000000

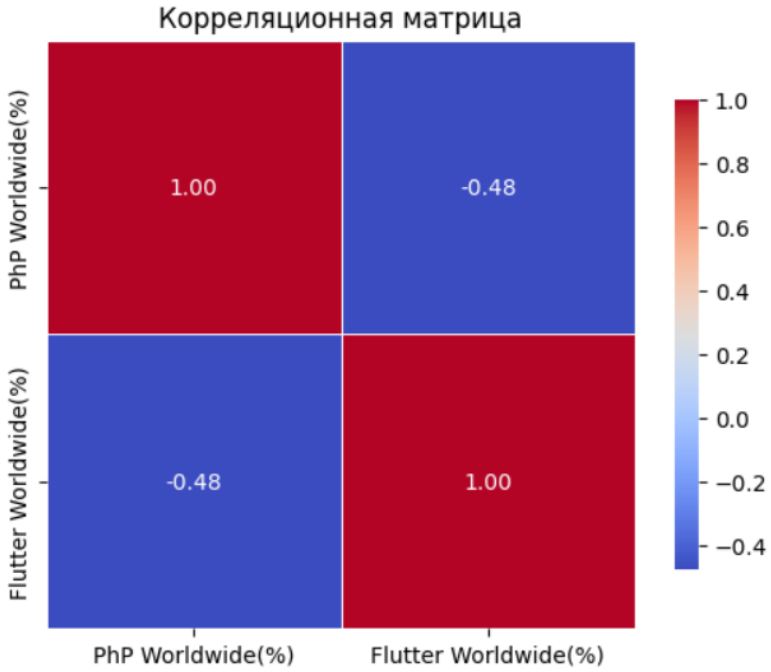

Рис. 2.3 – Корреляция между атрибутами: PhP Worldwide(%) и Flutter Worldwide(%)

	React Worldwide(%)	Swift Worldwide(%)
React Worldwide(%)	1.00000	0.45027
Swift Worldwide(%)	0.45027	1.00000

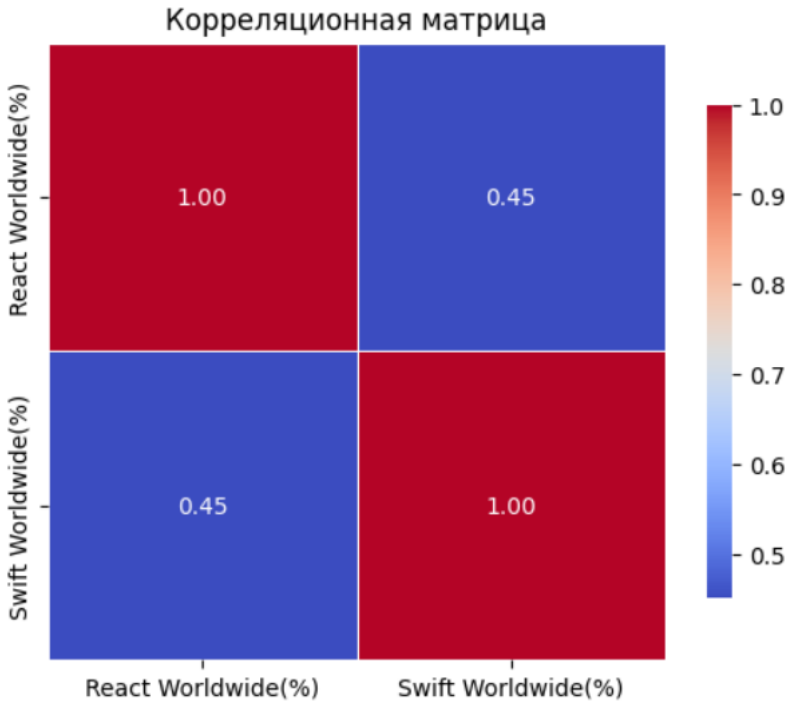

Рис. 2.4 – Корреляция между атрибутами: React Worldwide(%) и Swift Worldwide(%)

Представленная корреляция может говорить о средней связи между этими языками программирования. Их влияние друг на друга невелико.

Низкая положительная и отрицательная корреляция наблюдается у следующих атрибутов:

	TypeScript Worldwide(%)	Matlab Worldwide(%)
TypeScript Worldwide(%)	1.000000	0.306701
Matlab Worldwide(%)	0.306701	1.000000

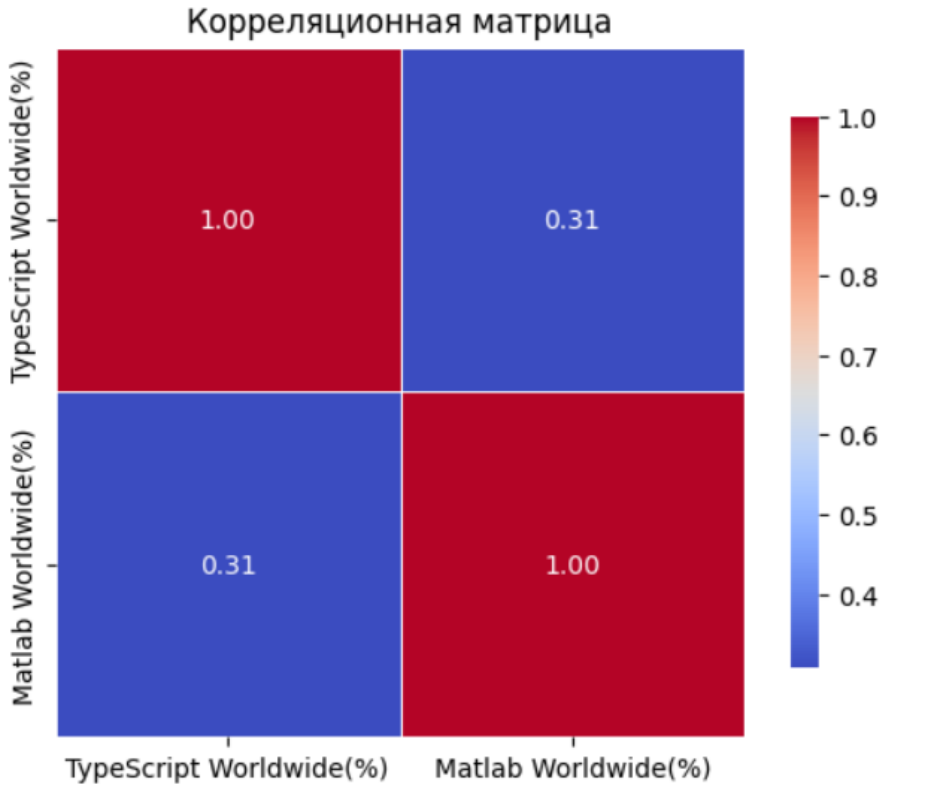

Рис. 2.5 – Корреляция между атрибутами: TypeScript Worldwide(%) и Matlab Worldwide(%)

	Python Worldwide(%)	JavaScript Worldwide(%)
Python Worldwide(%)	1.000000	-0.308851
JavaScript Worldwide(%)	-0.308851	1.000000

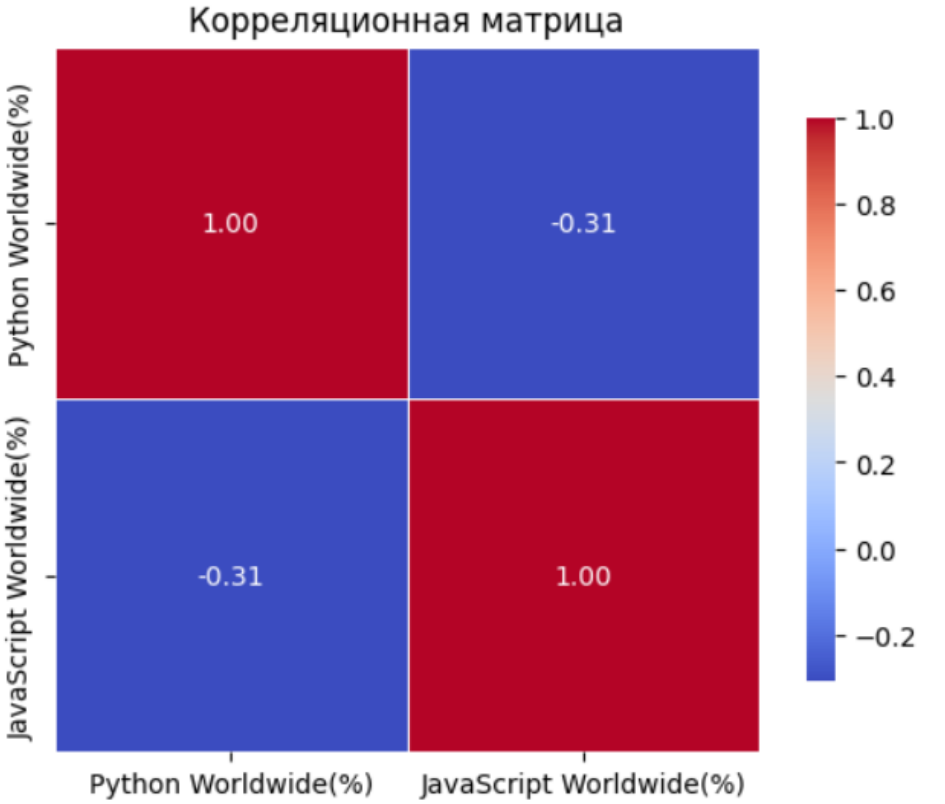

Рис. 2.6 – Корреляция между атрибутами: Python Worldwide(%) и JavaScript Worldwide(%)

Представленная корреляция может говорить о низкой связи между этими языками программирования. Их влияние друг на друга незначительно.

2. По матрицам корреляции нетрудно заметить, что все представленные атрибуты имеют определенную корреляцию.

3. Были построены графики рассеивания с помощью функции .scatter(). Они были изменены с помощью функций: .xlabel(); .ylabel(); .grid(True); .title().

```
plt.scatter(df['Python Worldwide(%)'],df['JavaScript Worldwide(%)']);
plt.xlabel('Python Worldwide(%)')
plt.ylabel('JavaScript Worldwide(%)')
plt.grid(True)
plt.title('График рассеяния: Python Worldwide(%) vs JavaScript Worldwide(%)');
```

Рис. 2.7 – Пример кода для построения графика рассеивания для атрибутов: Python Worldwide(%) и JavaScript Worldwide(%)

Графики рассеивания между следующими атрибутами: Python Worldwide(%) и JavaScript Worldwide(%); Java Worldwide(%) и C# Worldwide(%); PhP Worldwide(%) и Flutter Worldwide(%); React Worldwide(%) и Swift Worldwide(%); TypeScript Worldwide(%) и Matlab Worldwide(%):

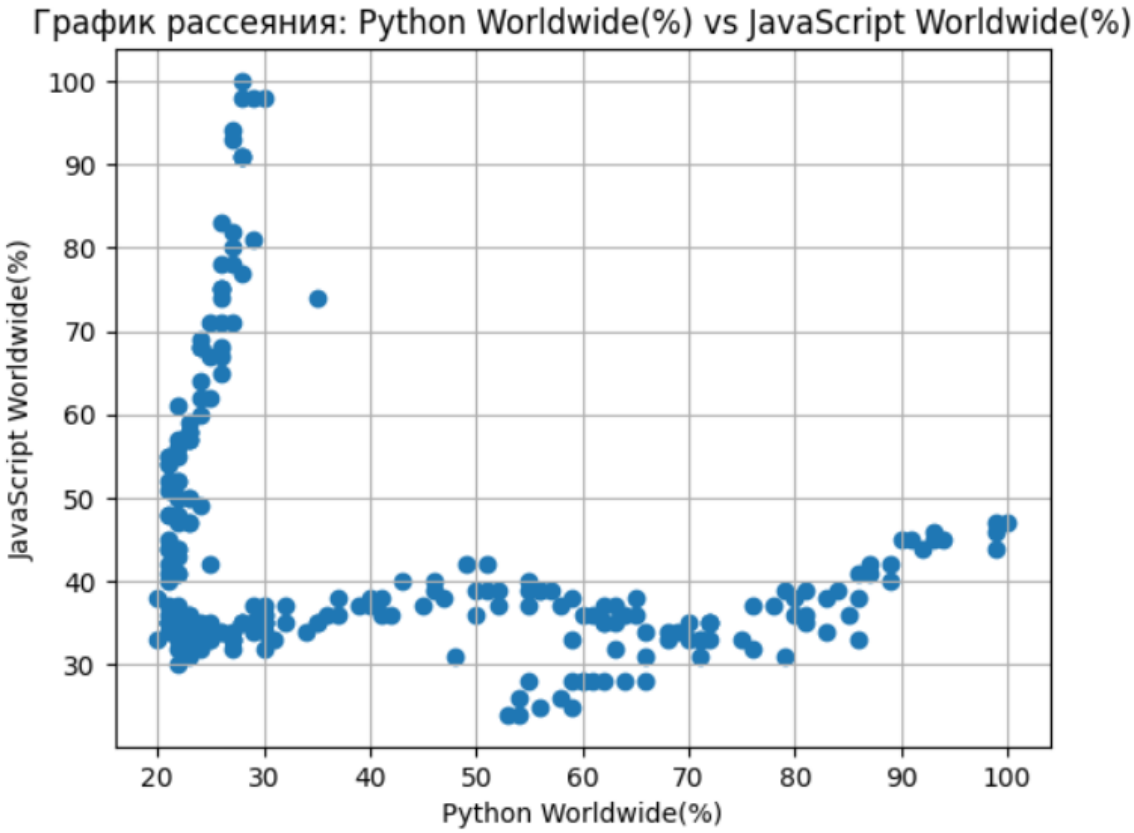

Рис. 2.8 – График рассеяния для атрибутов: Python Worldwide(%) и JavaScript Worldwide(%)

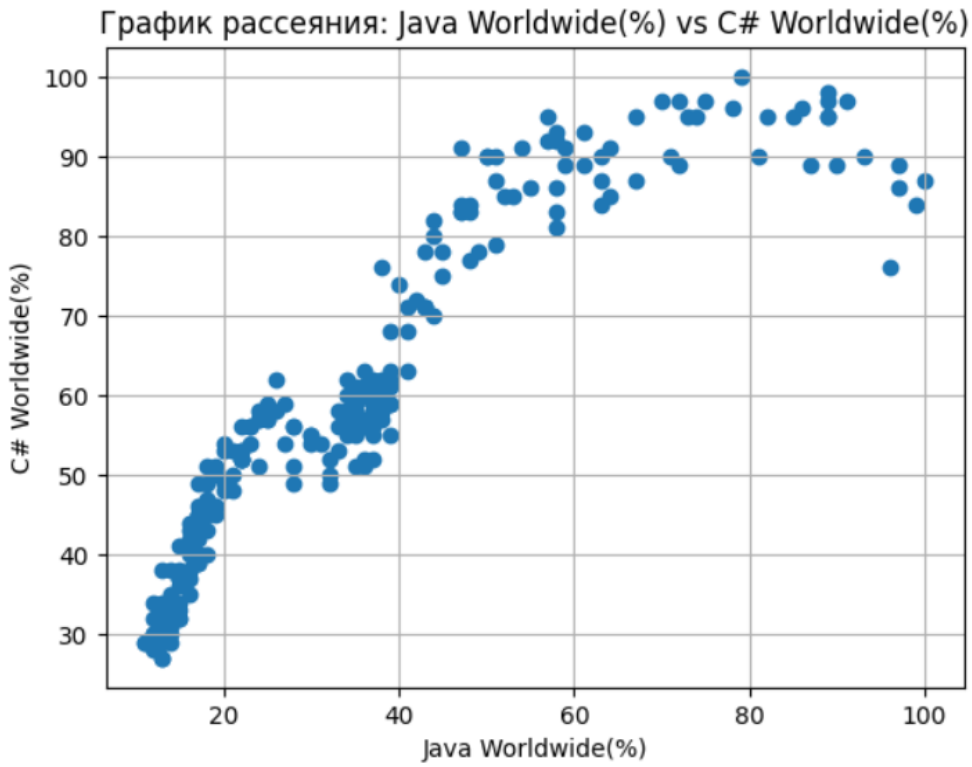

Рис. 2.9 – График рассеяния для атрибутов: Java Worldwide(%) и C# Worldwide(%)

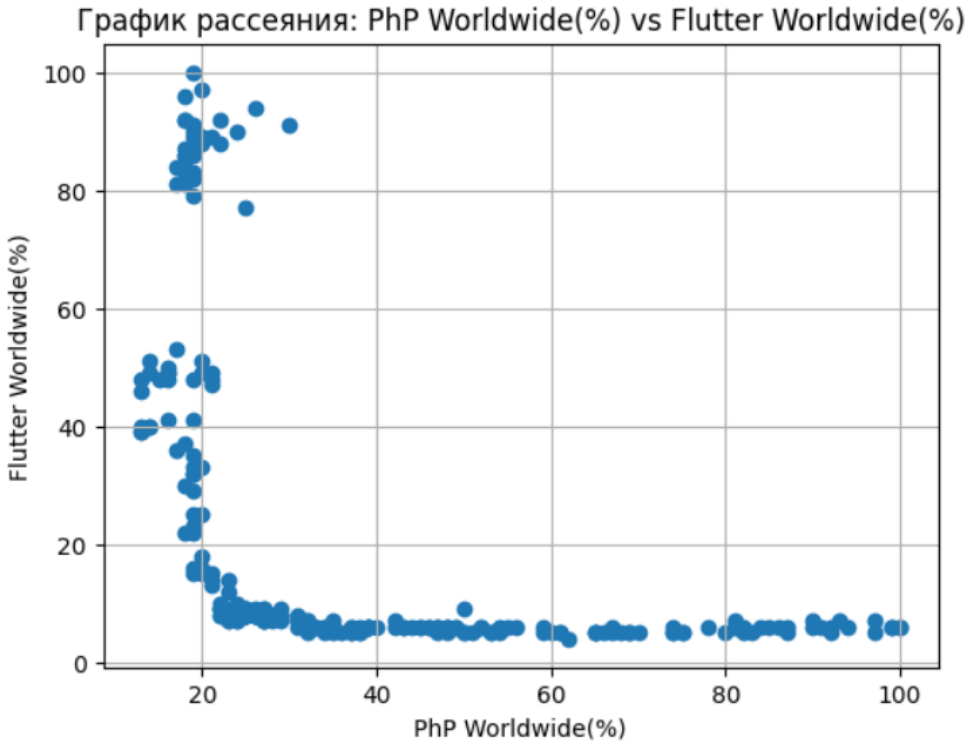

Рис. 2.10 – График рассеяния для атрибутов: PhP Worldwide(%) и Flutter Worldwide(%)

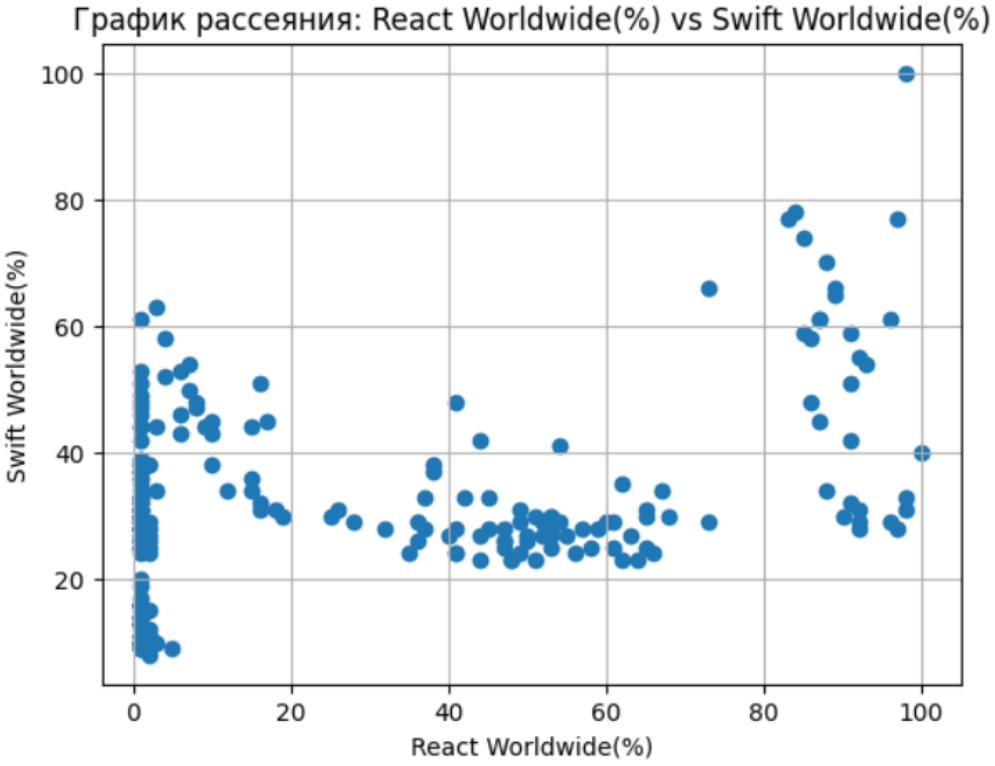

Рис. 2.11 – График рассеяния для атрибутов: React Worldwide(%) и Swift Worldwide(%)

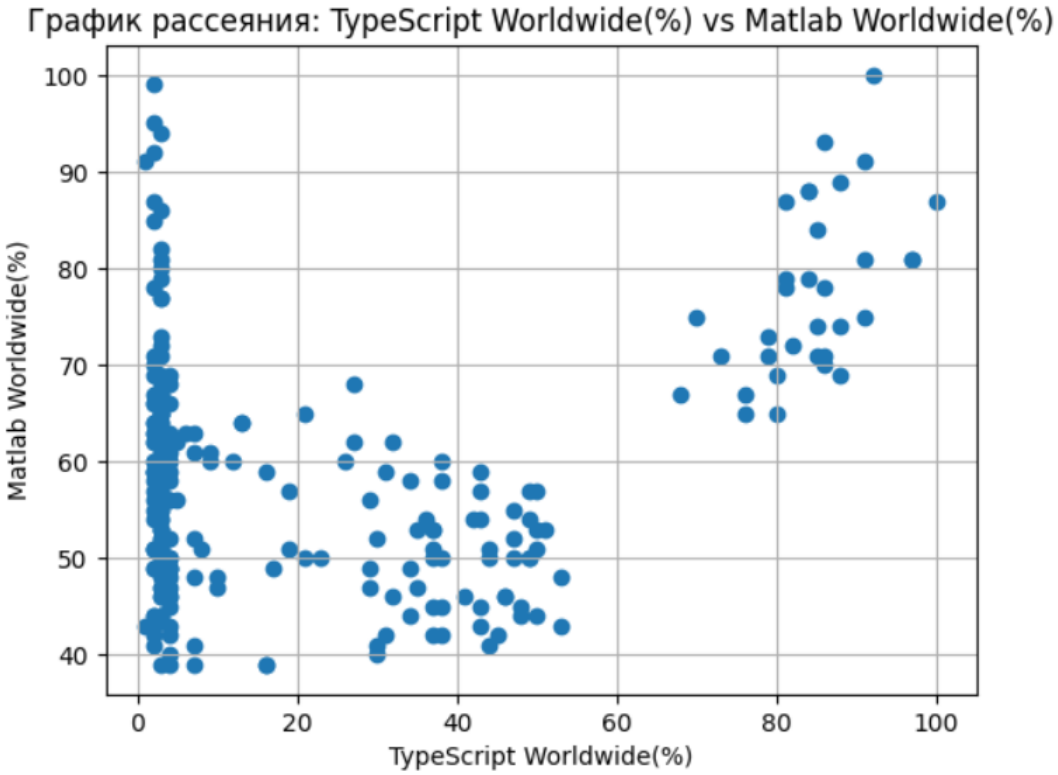

Рис. 2.12 – График рассеяния для атрибутов: TypeScript Worldwide(%) и Matlab Worldwide(%)

ЗАКЛЮЧЕНИЕ

В данной работе были изучены:

- Наборы данных;
- Jupyter Notebook;
- Построение гистограмм на Python;
- Поиск пропущенных значений на Python;
- Определение корреляции на Python;
- Построение графиков рассеивания на Python;

Были построены и вычислены:

- среднее значение, ско
- гистограмма распределения значений
- наличие выбросов
- пропущенные значения, сколько

По построенным графикам, гистограммам и матрицам были сделаны и записаны соответствующие выводы.

С помощью написания лабораторной работы стало ясно: как выявить, проанализировать, оформить зависимость и графическое представление данной зависимости между атрибутами. По выбранному датасету можно сказать об определенной степени зависимости популярности языков программирования, основанной на возможности совместного использования нескольких языков для одного проекта или насколько выбранный язык программирования подходит как для конкретной нишевой задачи, так и для разнообразных задач. Именно эти факторы и определяют популярность языка программирования.

Для выполнения практической работы была использована программа Jupyter-ноутбук; язык программирования Python; библиотеки: matplotlib, numpy, pandas.

Были изучены ссылки, представленные в методических указаниях к лабораторной работе.

Написание практической работы помогло: усвоить информацию о наборах данных и взаимодействиями с ними, разобраться с необходимыми библиотеками для построения графиков, гистограмм и матриц на Python.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. https://www.analyticsvidhya.com/blog/2021/02/an-intuitive-guide-to-visualization-in-python/

2. https://stepik.org/lesson/8086/step/1?unit=1365

3. https://medium.com/swlh/identify-outliers-with-pandas-statsmodels-and-seaborn-2766103bf67c