МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт информационных технологий (ИИТ)
Кафедра прикладной математики (ПМ)

ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ №3

по дисциплине «Языки программирования для статистической обработки данных»

Студент группы *ИМБО-11-23 Журавлев Ф.А.* _____
(подпись)

Преподаватель *Трушин С.М.*

_____
(подпись)

Москва 2025 г.

1 ЦЕЛЬ И ЗАДАЧИ

Цель практической работы:

Освоить методы построения базовых графиков в Python, R, а также научиться визуализировать данные с использованием различных инструментов.

Задачи практической работы:

1. Построить базовые графики:

- Гистограммы, линейные графики, боксплоты в Python (matplotlib, seaborn).
- Те же графики в R (ggplot2).

2. Сравнить визуализации, созданные в Python, R.

3. Проанализировать, в каких ситуациях каждый инструмент наиболее эффективен.

2. РЕЗУЛЬТАТЫ ПРАКТИЧЕСКОЙ РАБОТЫ

2.1 Графическое представление в Python.

После загрузки данных в Python, первым делом необходимо построить гистограмму по исходному набору данных, построим гистограмму по столбцу BPM. Ниже представлен код для реализации графика:

рисунок 2.1.1 – Код гистограммы

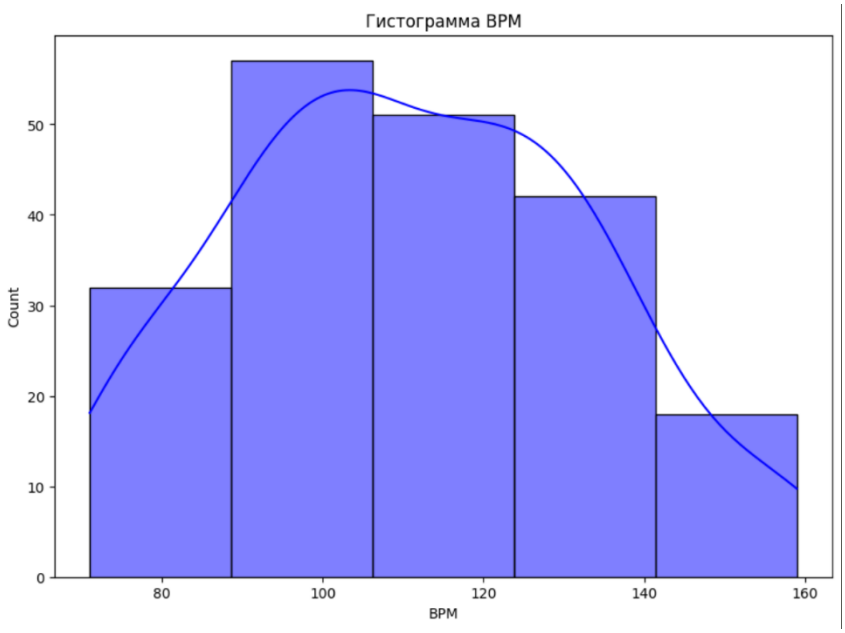
```
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize = (10,7))
sns.histplot(df['BPM'], bins = 5, kde = True, color = 'red',)
plt.title("Гистограмма BPM")
plt.show()
```

[6] ✓ 2.2s

Далее посмотрим на получившийся график:

рисунок 2.1.2 – Гистограмма BPM



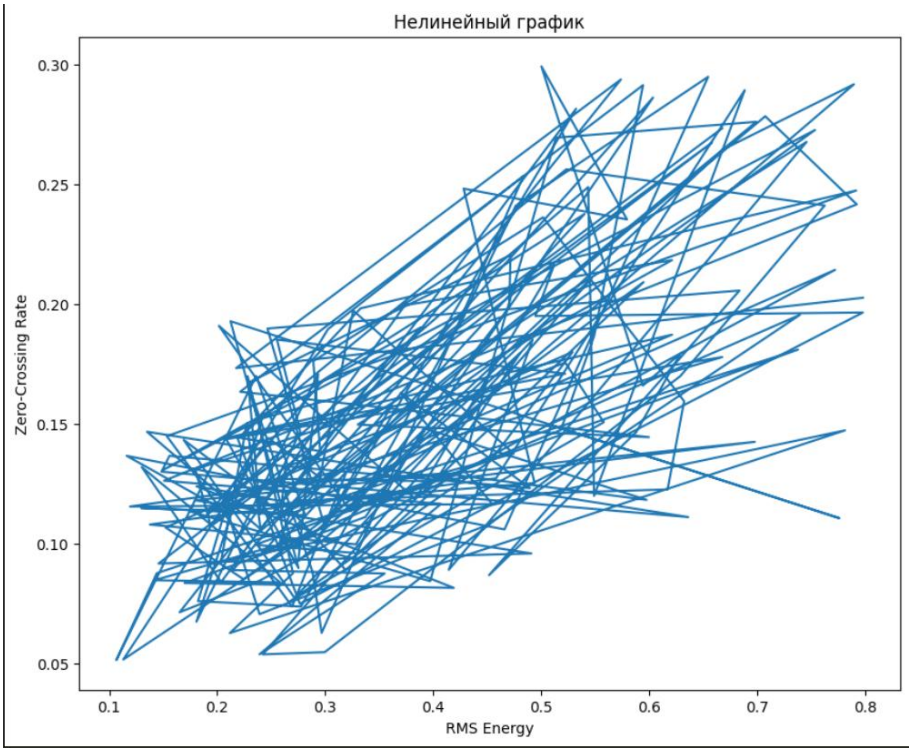
Далее построим график линейной зависимости в Python и напишем код:

рисунок 2.1.3 – Код графика нелинейной зависимости

```
plt.figure(figsize = (10,8))
plt.plot(df['RMS Energy'],df['Zero-Crossing Rate'])
plt.title('Нелинейный график')
plt.xlabel('RMS Energy')
plt.ylabel("Zero-Crossing Rate")
plt.show()
```

Вот график, который нам выводит Python:

рисунок 2.1.4 – График нелинейной зависимости



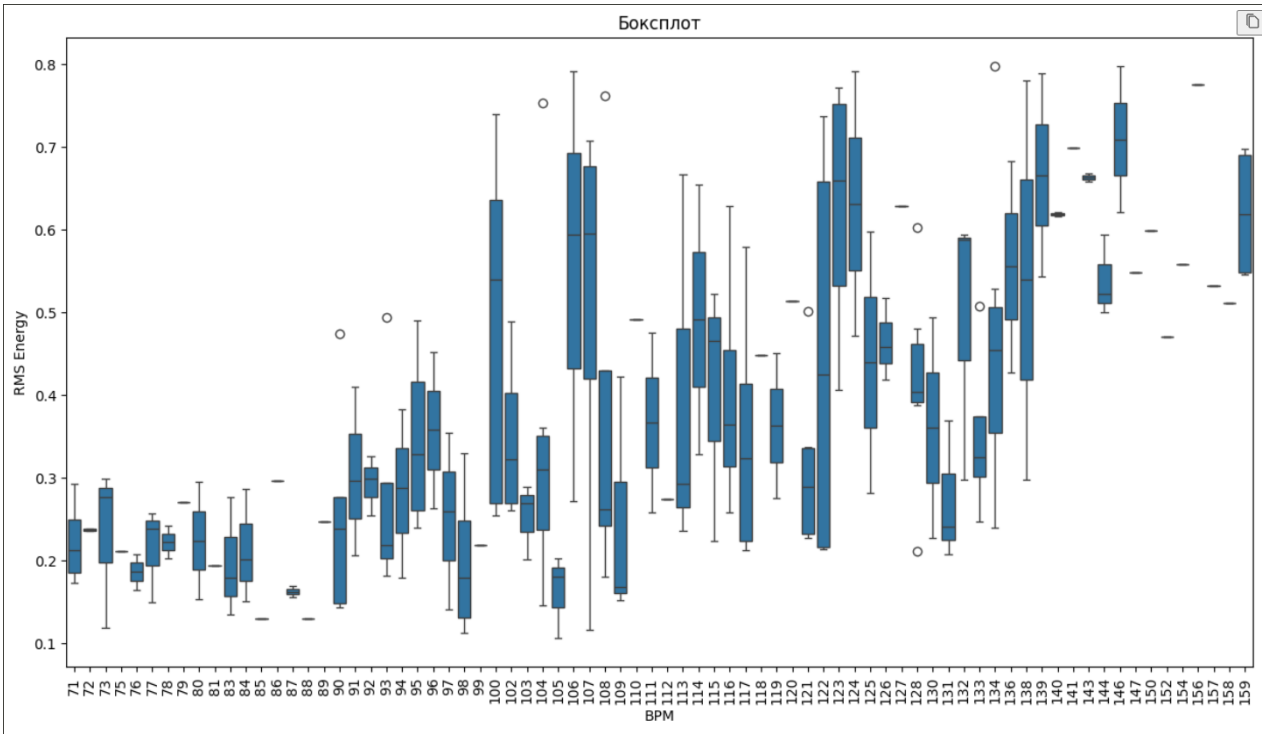
Далее рассмотрим последний график из практической работы «бокс-плот». Напишем код его реализации:

рисунок 2.1.5 – Код реализации боксплота

```
plt.figure(figsize = (15,8))
sns.boxplot(x = 'BPM', y = 'RMS Energy', data = df)
plt.title('Боксплот')
plt.xticks(rotation=90)
plt.show()
```
✓ 0.7s

Посмотрим, что он нам выводит:

рисунок 2.1.6 – График боксплота



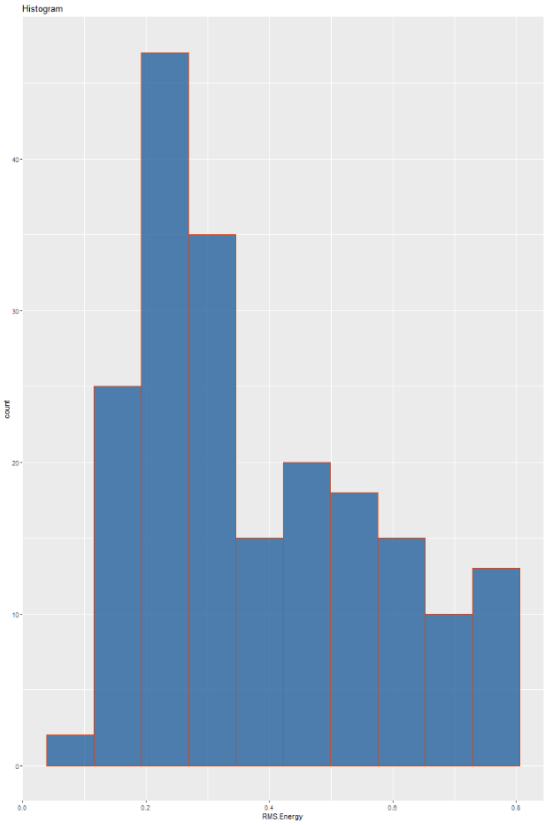
2.2 Графическое представление в R

Теперь рассмотрим реализацию тех же график с помощью языка R.

Рисунок 2.2.1 – Код Гистограммы.

```
library(ggplot2)
library(readr)
df <- read.csv("music_genre_dataset.csv")
p <- ggplot(df, aes(x = RMS.Energy)) +
geom_histogram(bins = 10, fill = '#094e92', color = '#e43f08', alpha = 0.7) +
labs(title = "Histogram")
print(p)
```

Рисунок 2.2.2 – Гистограмма.

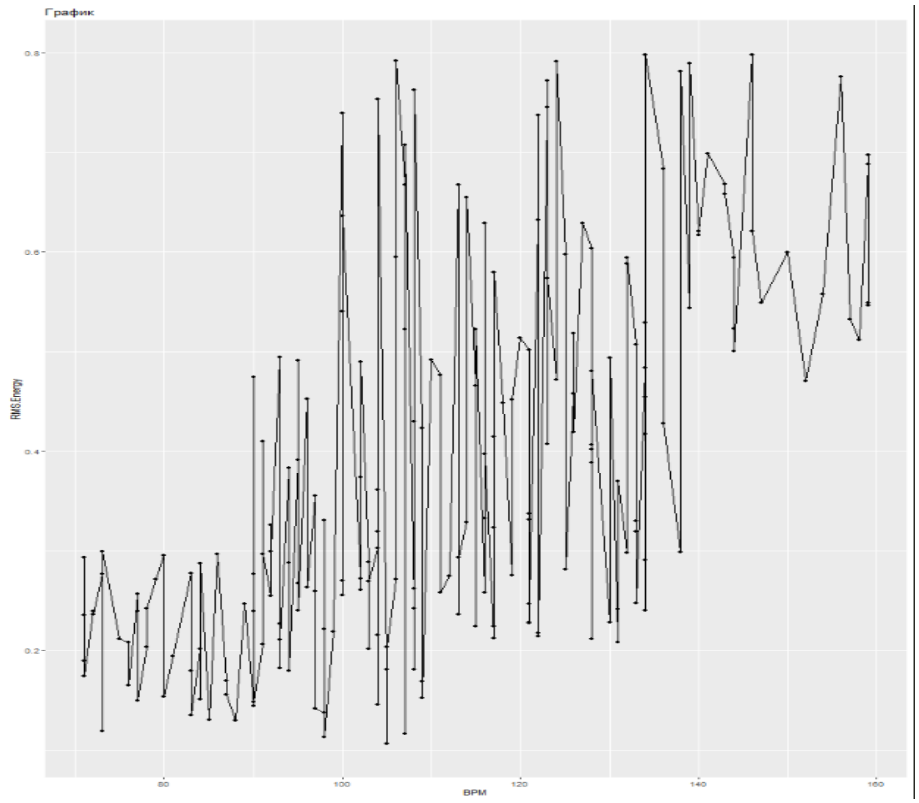


Теперь посмотрим на реализацию графика линейной зависимости:

Рисунок 2.2.3 – код линейной зависимости.

```
library(ggplot2)
library(readr)
df <- read.csv("music_genre_dataset.csv")
p <- ggplot(df, aes(x = BPM,y = RMS.Energy)) +
geom_line() + geom_point() + labs(title = 'График')
print(p)
```

Рисунок 2.2.4 – график линейной зависимости

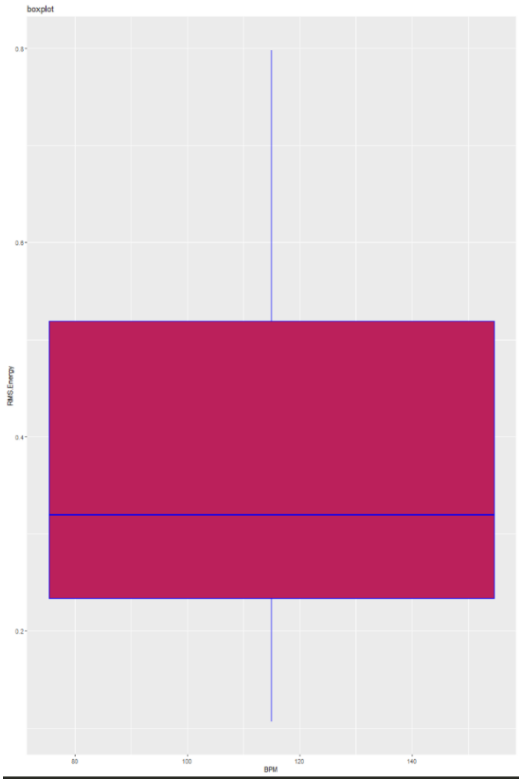


Рассмотрим «боксплот» на языке R:

Рисунок 2.2.5 – код боксплота

```
library(ggplot2)
library(readr)
df <- read.csv("music_genre_dataset.csv")
p <- ggplot(df, aes(x = BPM,y = RMS.Energy)) +
geom_boxplot(fill = '#bb205b',color = 'blue') +
labs(title = 'boxplot')
print(p)
```

Рисунок 2.2.6 – Боксплот



ИТОГИ И ВЫВОДЫ:

Рассмотрев различные способы построения графиков и диаграмм с помощью двух языков программирования и программы для анализа данных, можно сделать пару выводов. Jupyter по мне очень удобная вещь, ведь она позволяет написать часть кода, потом визуализировать результаты, затем продолжить написании программы, R более строг, но тоже достаточно удобный язык для анализа данных.