



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 №6
по дисциплине «Языки программирования для статистической обработки
данных»

Студент группы *ИМБО-11-23, Журавлев Ф.А.*

(подпись)

Преподаватель *Трушин СМ*

(подпись)

Москва 2025 г.

1) ЦЕЛЬ И ЗАДАЧИ

Цель практической работы:

Освоить построение моделей множественной регрессии, интерпретацию их результатов и визуализацию метрик качества в Python, R.

Задачи практической работы:

1. Построить модели множественной регрессии:
 - Выявить влияние нескольких факторов на целевую переменную.
 - Python: использование statsmodels и sklearn для построения моделей.
 - R: использование функции `lm()`.
2. Оценить качество моделей:
 - Рассчитать метрики R^2 и MSE для оценки качества модели.
 - Python: инструменты `sklearn.metrics`.
 - R: функции `summary()` и пакеты для анализа ошибок.
3. Визуализировать результаты:
 - Построение графиков зависимости предсказанных значений от реальных.
 - Сравнить удобство визуализации в Python и R.
4. Провести сравнительный анализ результатов, полученных в трёх инструментах.

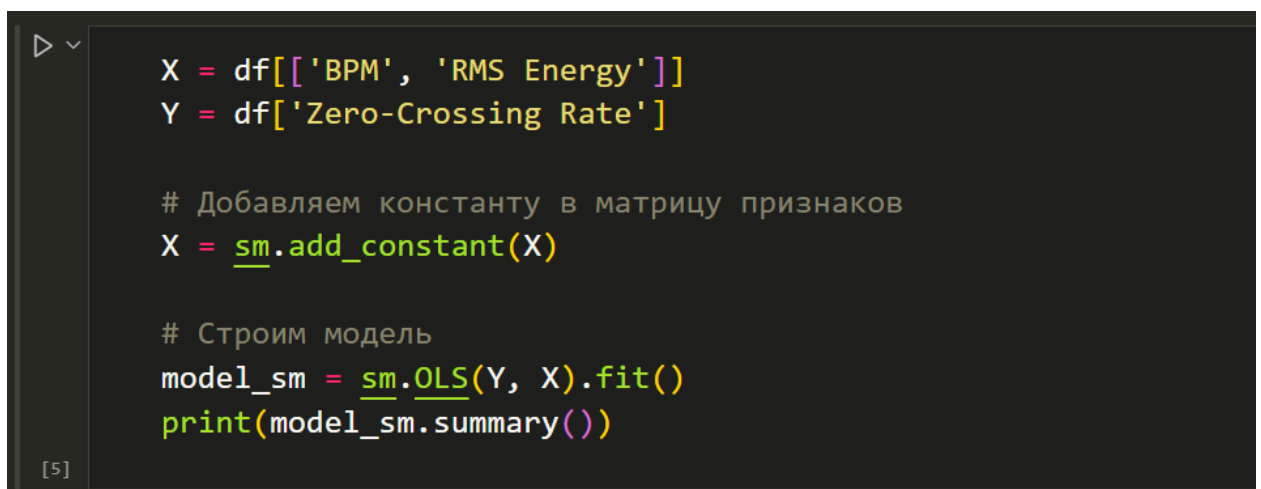
2) РЕЗУЛЬТАТЫ ПРАКТИКИ

Шаг 1) Множественная линейная регрессия в Python

1.1) Построение множественной линейной регрессии.

После загрузки исходной таблицы данных в формате .csv, следует написать код для построения множественной линейной регрессии. Воспользуемся двумя способами.

Рисунок 1.1 — Построение множественной линейной регрессии



```
X = df[['BPM', 'RMS Energy']]
Y = df['Zero-Crossing Rate']

# Добавляем константу в матрицу признаков
X = sm.add_constant(X)

# Строим модель
model_sm = sm.OLS(Y, X).fit()
print(model_sm.summary())
```

[5]

Построим модель чуть-чуть другим способом. Ниже приведен код:

Рисунок 1.2 — Построение множественной линейной регрессии

```
# Построение модели с sklearn
model_sk = LinearRegression()
model_sk.fit(X, Y)

# Коэффициенты
print("Intercept:", model_sk.intercept_)
print("Coefficients:", model_sk.coef_)

[6]
... Intercept: 0.03216947906516615
Coefficients: [0.          0.000437  0.17988613]
```

В данном случае мы сразу получаем коэффициенты модели, а рассматривая код 1.1, то вот что он выводит:

Рисунок 1.3 — Вывод из кода 1.1.

```
=====
Dep. Variable:      Zero-Crossing Rate   R-squared:                0.441
Model:              OLS                  Adj. R-squared:           0.436
Method:             Least Squares        F-statistic:             77.86
Date:               Mon, 28 Apr 2025     Prob (F-statistic):      1.21e-25
Time:               14:11:22             Log-Likelihood:          335.89
No. Observations:   200                  AIC:                     -665.8
Df Residuals:       197                  BIC:                     -655.9
Df Model:           2
Covariance Type:    nonrobust
=====
              coef    std err          t      P>|t|      [0.025    0.975]
-----
const         0.0322     0.017      1.900     0.059     -0.001     0.066
BPM           0.0004     0.000      2.418     0.017     8.06e-05     0.001
RMS Energy    0.1799     0.021      8.440     0.000      0.138     0.222
=====
Omnibus:                 3.820   Durbin-Watson:           2.039
Prob(Omnibus):            0.148   Jarque-Bera (JB):         2.794
Skew:                     0.139   Prob(JB):                 0.247
Kurtosis:                 2.492   Cond. No.                  777.
=====

Notes:
```

Далее стоит рассчитать такие метрические показатели, как R^2 и MSE, то есть коэффициент детерминации и среднеквадратичную ошибку. Код для расчета:

Рисунок 1.4 — Код для MSE & R^2

```
# Предсказание
y_pred = model_sk.predict(X)

# Расчёт метрик качества
r2 = r2_score(Y, y_pred)
mse = mean_squared_error(Y, y_pred)

print("R^2:", r2)
print("MSE:", mse)
```

... R^2: 0.44149749454500364
MSE: 0.002035953549456323

Далее по практической работе нужно построить два графика — это график остатков и график, благодаря которому можно сравнивать реальные и предсказанные значения:

Рисунок 1.4 — График сравнений.

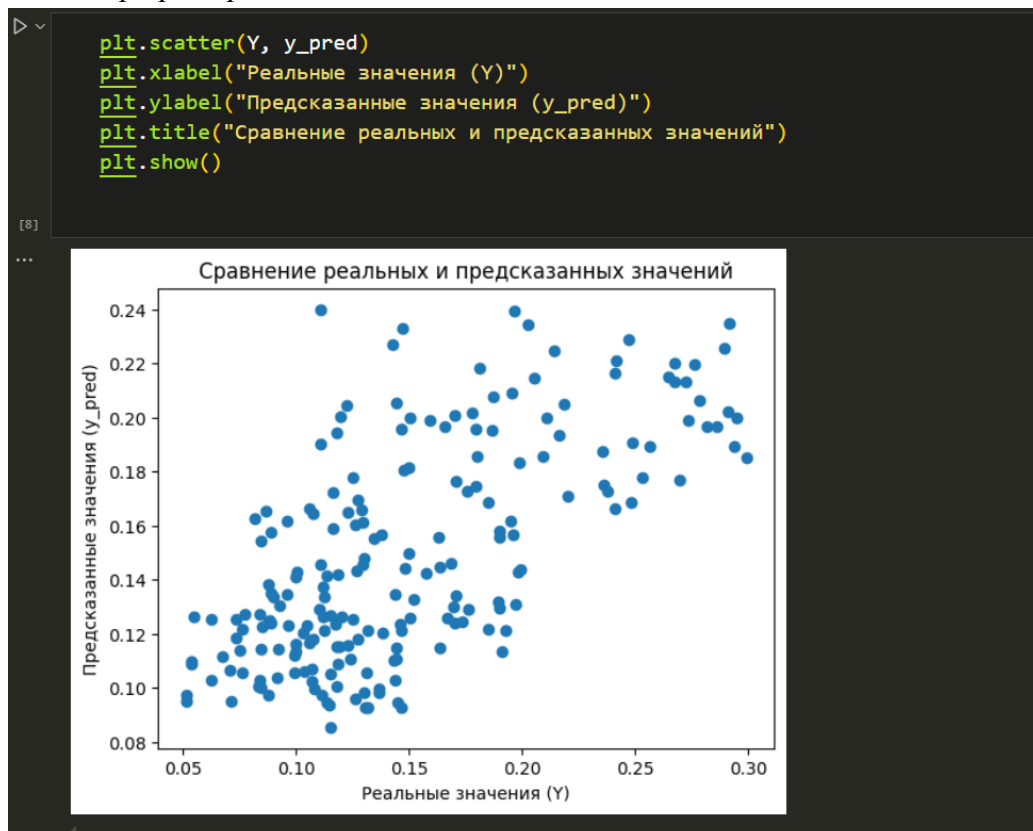
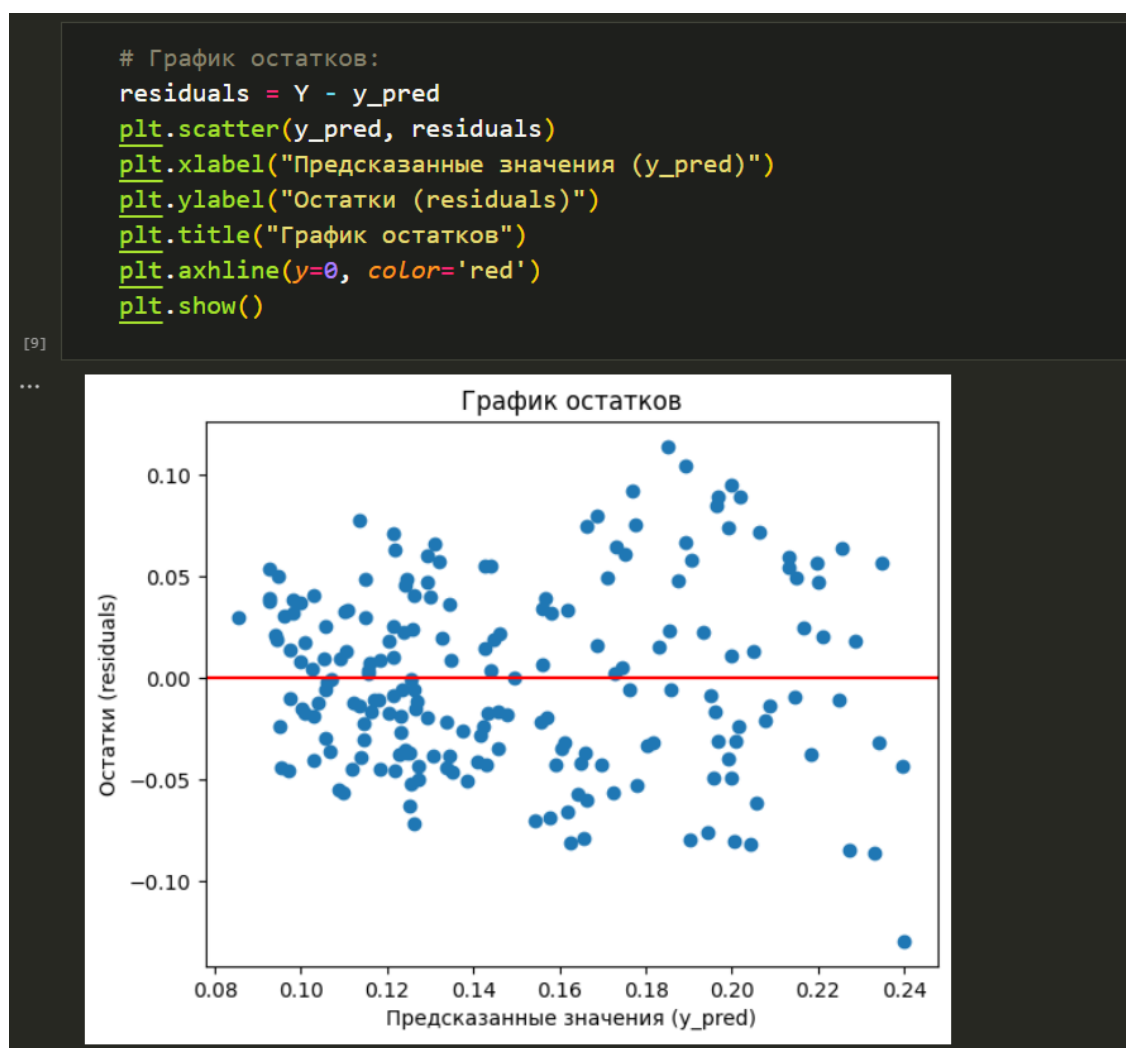


Рисунок 1.5 — График остатков.



Шаг 2) Множественная линейная регрессия в R.

2.1) Построение множественной линейной регрессии.

Далее сделаем все то же самое, но с помощью языка программирования R.

Рисунок 2.1 – Код на языке R.

```
1 library(readr)
2 df <- read.csv("D:/Documents/Learning/3/R/5/df_without_genre.csv")
3
4 model <- lm(Zero.Crossing.Rate ~ BPM + RMS.Energy, data = df)
5 print(summary(model))
6
7 # Дополнительные метрики
8 predictions <- predict(model, df)
9 residuals <- df$Zero.Crossing.Rate - predictions
10 mse <- mean(residuals^2)
11 print(mse)
12
13 library(Metrics)
14 mse_val <- mse(df$Zero.Crossing.Rate, predictions)
15 r2_val <- cor(df$Zero.Crossing.Rate, predictions) ^ 2
16 print(paste("MSE =", mse_val))
17 print(paste("R^2 =", r2_val))
18
19 print(plot(df$Zero.Crossing.Rate, predictions, xlab = "Реальные значение (Y) ",
20 ylab = "Предсказанные значения (Y_pred) ",
21 main = "Реальные vs. Предсказанные"))
22 abline(a=0, b=1, col="yellow")
23
24 print(plot(model))
```

В данном коде мы написали сначала код множественной линейной регрессии, потом код для расчета предсказаний, остатков, MSE & R².

Рисунок 2.2 – Метрические показатели.

```
[1] 0.002035954  
[1] "MSE = 0.00203595354945632"  
[1] "R^2 = 0.441497494545004"
```


3 СРАВНЕНИЕ РЕЗУЛЬТАТОВ

И там, и там результаты получились одинаковые. Понятное дело, с чуть-чуть разным округлением. Работа в Python куда удобнее чем в R. Если мы рассмотрим график сравнения и остатков, то можно сделать такие выводы:

- 1) Модель в целом улавливает общую тенденцию, но имеет проблемы с точностью предсказаний.
- 2) Данные имеют нелинейную структуру, и модель не может адекватно её учитывать.
- 3) Есть две разные группы данных, которые модель обрабатывает по-разному.

4 ВЫВОДЫ

В 6 практической работе приведены реализации множественной линейной регрессии и вычислением метрик регрессии. Также построены графики сравнений и остатков на двух языках программирования.