МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«Донецкий государственный университет»

Кафедра компьютерных технологий

**ЛАБОРАТОРНАЯ РАБОТА №1**

по дисциплине: «Методология и методы научных исследований»

Выполнил:

студент группы ИВТ-2

Лямцев О.А.

Проверил:

проф. Аверин Г.В.

Донецк

2024

**Лабораторная работа № 1**

Применение метода группового учета аргументов к анализу сложных процессов

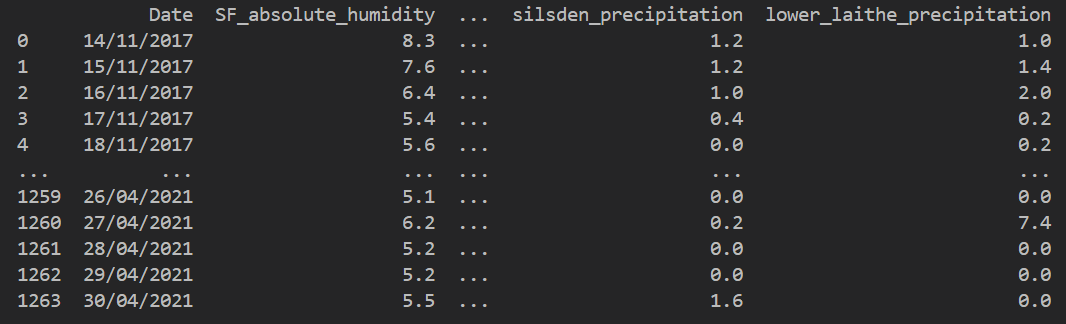
**Цель работы**: изучить метод группового учета аргументов, применить его к построению прогнозных моделей сложной системы (процесса).

**Датасет**

Ссылка на датасет: <https://www.kaggle.com/datasets/thomaswrightanderson/river-aire-discharge-time-series>

Датасет посвящен прогнозированию суточного стока рек в Армли (Лидс, Великобритания)

Выходной столбец: *discharge\_armley.* Все остальные столбцы, кроме *Date* – входные данные.



**Выбор оптимальной модели**

Перебор моделей Multi:

LEVEL 1 [=========================] 100% [00m:00s] (22 combinations) error=0.025575

LEVEL 2 [=========================] 100% [00m:00s] (21 combinations) error=0.022355

LEVEL 3 [=========================] 100% [00m:00s] (20 combinations) error=0.020307

LEVEL 4 [=========================] 100% [00m:00s] (19 combinations) error=0.016295

LEVEL 5 [=========================] 100% [00m:00s] (18 combinations) error=0.014762

LEVEL 6 [=========================] 100% [00m:00s] (17 combinations) error=0.014413

LEVEL 7 [=========================] 100% [00m:00s] (16 combinations) error=0.014399

LEVEL 8 [=========================] 100% [00m:00s] (15 combinations) error=0.014399

LEVEL 9 [=========================] 100% [00m:00s] (14 combinations) error=0.014454

Перебор моделей Ria

LEVEL 1 [=========================] 100% [00m:00s] (231 combinations) error=0.016017

LEVEL 2 [=========================] 100% [00m:00s] (22 combinations) error=0.012083

LEVEL 3 [=========================] 100% [00m:00s] (22 combinations) error=0.010677

LEVEL 4 [=========================] 100% [00m:00s] (22 combinations) error=0.009671

LEVEL 5 [=========================] 100% [00m:00s] (22 combinations) error=0.009179

LEVEL 6 [=========================] 100% [00m:00s] (22 combinations) error=0.008649

LEVEL 7 [=========================] 100% [00m:00s] (22 combinations) error=0.008605

LEVEL 8 [=========================] 100% [00m:00s] (22 combinations) error=0.008573

LEVEL 9 [=========================] 100% [00m:00s] (22 combinations) error=0.008574

Лучшая модель Multi (значение критерия CR = 0.01445, квадратичная погрешность MSE = 0.001239):

y = 0.0003\*x1 - 0.0017\*x3 + 0.0243\*x7 + 0.8444\*x12 - 0.0359\*x13 + 0.0896\*x15 - 0.0913\*x17 + 0.1206\*x19 - 0.0572

Лучшая модель Ria (значение критерия CR = 0.008574, квадратичная погрешность MSE = 0.001358):

f1 = 0.828\*x12 - 0.6442\*x15 - 0.7244\*x12\*x15 + 0.4639\*x12^2 + 1.0305\*x15^2 + 0.0061

f2 = 0.0126\*x7 + 0.718\*f1 + 0.4825\*x7\*f1 - 0.0295\*x7^2 - 0.0879\*f1^2 + 0.0016

f3 = 0.0021\*x2 + 0.8522\*f2 + 0.2069\*x2\*f2 - 0.0011\*x2^2 + 0.0328\*f2^2 - 0.0004

f4 = 0.0394\*x16 + 0.9939\*f3 + 0.2737\*x16\*f3 - 0.0222\*x16^2 - 0.1153\*f3^2 - 0.0014

f5 = 0.0567\*x12 + 0.9453\*f4 - 1.5372\*x12\*f4 + 0.5619\*x12^2 + 0.985\*f4^2 - 0.0002

f6 = - 0.0109\*x17 + 1.0062\*f5 - 0.0656\*x17\*f5 + 0.0264\*x17^2 + 0.0105\*f5^2 + 0.0001

f7 = 0.0139\*x12 + 0.9841\*f6 - 0.987\*x12\*f6 + 0.4373\*x12^2 + 0.5503\*f6^2 + 1.40961e-05

y = - 0.0008\*x17 + 1.0003\*f7 - 0.0046\*x17\*f7 + 0.0019\*x17^2 + 0.001\*f7^2 + 1.30535e-05

**Графики**

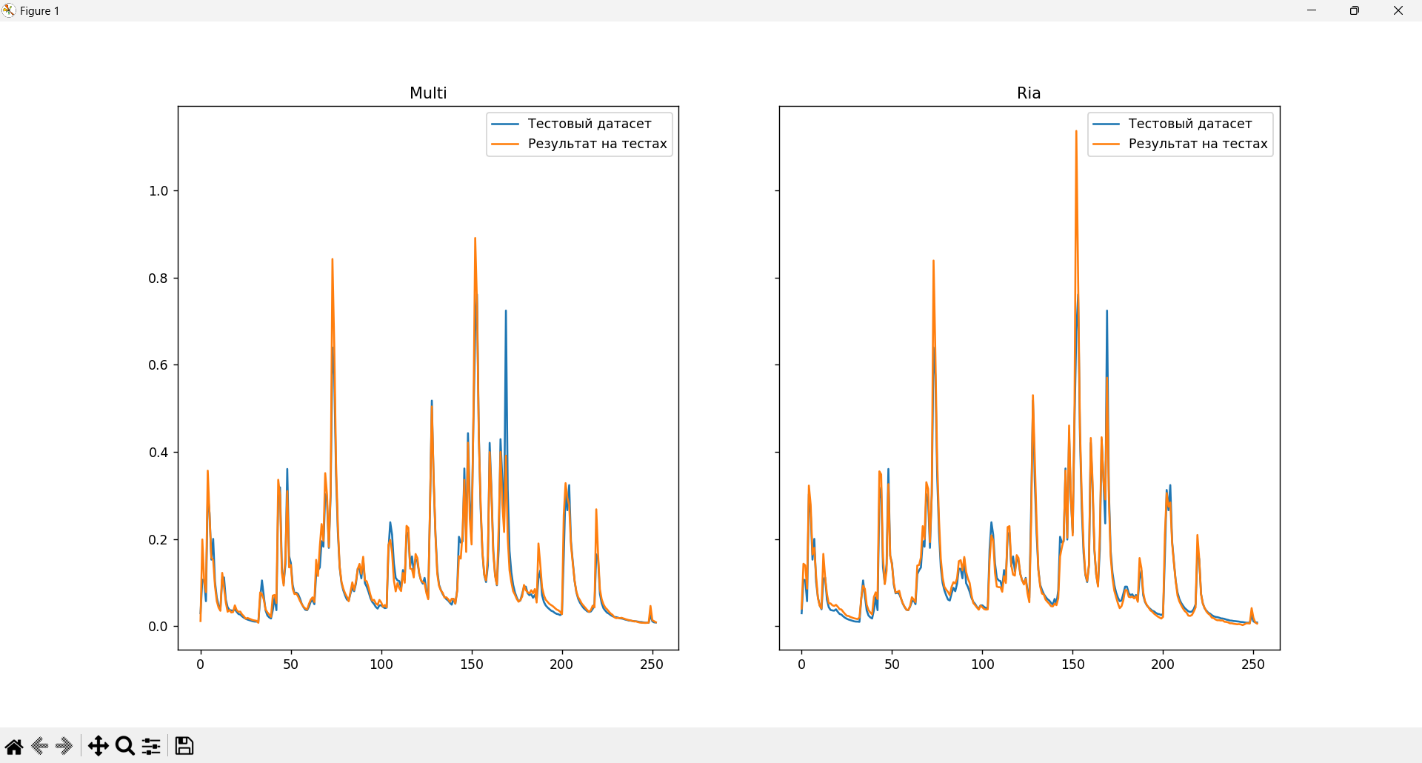


Рисунок 1 - тестирование моделей на экзаменационной выборке

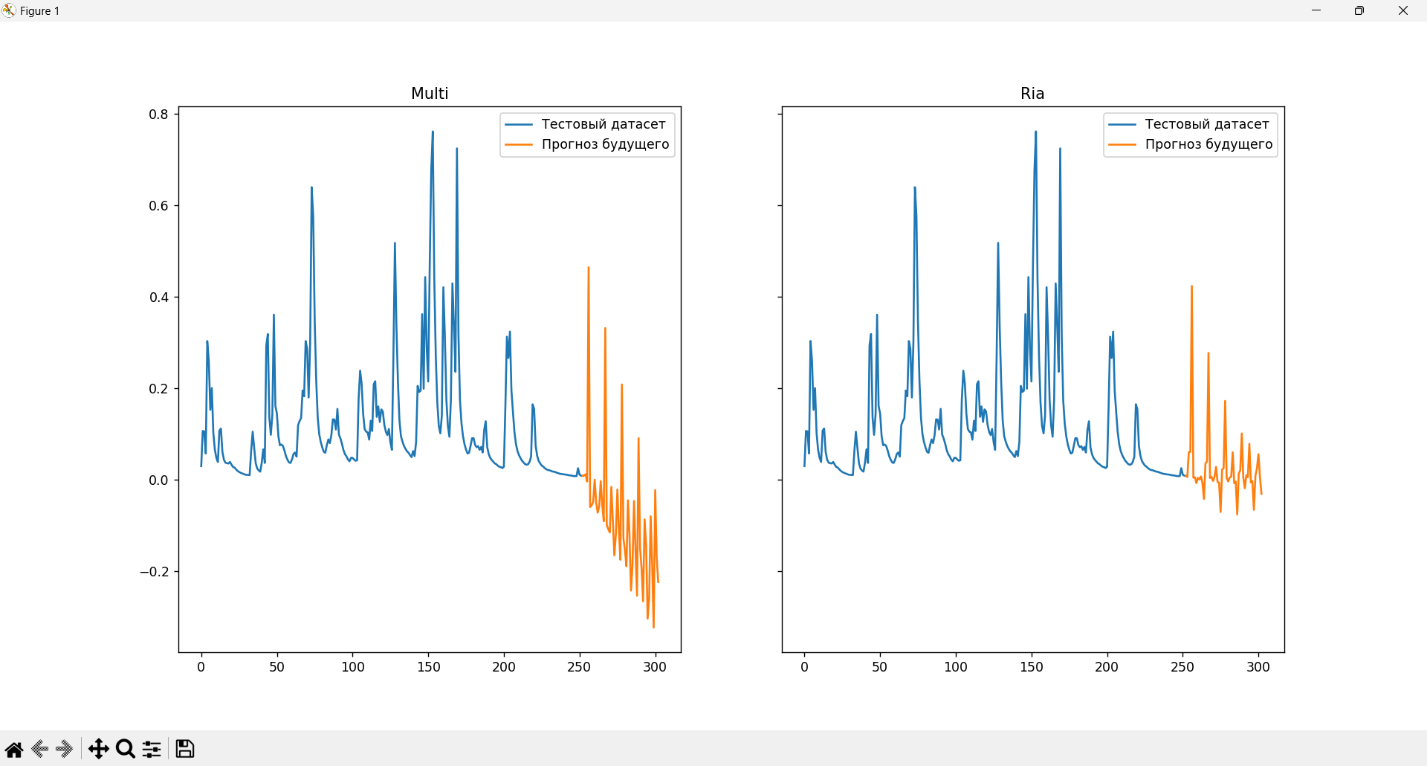


Рисунок 2 - прогнозирование развития процесса на 50 дней вперед

**Листинг программы**

import gmdh

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from gmdh import Combi, Mia, Ria, Multi

from IPython.display import display, HTML

data = pd.read\_csv("river\_aire\_discharge\_timeseries.csv")

data = data.drop(['Date'], axis=1)

data = data.fillna(0)

data=(data-data.min())/(data.max()-data.min())

train\_size = int(len(data) \* 0.8)

train = data[:train\_size]

test = data[train\_size:]

y\_train = train['discharge\_armley'].values.tolist()

x\_train = train.drop(['discharge\_armley'], axis=1).values.tolist()

y\_test = test['discharge\_armley'].values.tolist()

x\_test = test.drop(['discharge\_armley'], axis=1).values.tolist()

modelMulti = Multi()

modelRia = Ria()

modelMulti.fit(x\_train, y\_train, verbose=1, n\_jobs=-1,test\_size=0.1)

modelRia.fit(x\_train, y\_train, verbose=1, n\_jobs=-1,test\_size=0.1)

# Testing subplots

pred\_multi = modelMulti.predict(x\_test)

pred\_ria = modelRia.predict(x\_test)

figure, axis = plt.subplots(1, 2, sharey=True)

print('MSE Multi: ', np.mean((y\_test - pred\_multi)\*\*2))

print('MSE Ria: ', np.mean((y\_test - pred\_ria)\*\*2))

# Multi

axis[0].plot(y\_test, label = 'Тестовый датасет')

axis[0].plot(pred\_multi, label = 'Результат на тестах')

axis[0].set\_title("Multi")

axis[0].legend()

# Ria

axis[1].plot(y\_test,label = 'Тестовый датасет')

axis[1].plot(pred\_ria, label = 'Результат на тестах')

axis[1].set\_title("Ria")

axis[1].legend()

plt.show()

# Forecast subplots

future\_multi = [y\_test[-1]]

future\_ria = [y\_test[-1]]

x = list(range(0, len(y\_test)))

pred\_multi = modelMulti.predict(x\_test)

pred\_ria = modelRia.predict(x\_test)

figure, axis = plt.subplots(1, 2, sharey=True)

future\_multi = future\_multi + modelMulti.predict( x\_test[-1], lags=50).tolist()

future\_ria = future\_ria + modelRia.predict( x\_test[-1], lags=50).tolist()

# Multi

axis[0].plot(x,y\_test, label = 'Тестовый датасет')

axis[0].plot(list(range(x[-1], x[-1] + len(future\_multi))), future\_multi, label='Прогноз будущего')

axis[0].set\_title("Multi")

axis[0].legend()

# Ria

axis[1].plot(x,y\_test,label = 'Тестовый датасет')

axis[1].plot(list(range(x[-1], x[-1] + len(future\_ria))), future\_ria, label='Прогноз будущего')

axis[1].set\_title("Ria")

axis[1].legend()

plt.show()

print('Multi: ', modelMulti.get\_best\_polynomial())

print('Ria: ', modelRia.get\_best\_polynomial())

modelMulti.save('multi.json')

modelRia.save('ria.json')