|  |  |  |
| --- | --- | --- |
|  | Изображение выглядит как зарисовка, рисунок, символ, эмблема  Автоматически созданное описание |  |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |
| **Институт комплексной безопасности и специального приборостроения**  Кафедра КБ-14 «Технологии искусственного интеллекта и анализ данных»  Дисциплина «Введение в машинное обучение» | | |

**Отчёт**

«Практическая работа №2»

Выполнил:

Студент 3 курса

Группа БСБО-06-22

Шифр 22Б1240

Белявцева Екатерина Александровна

Москва, 2024

# Задание

Построить по многорядному полиномиальному алгоритму метода группового учета аргументов модель предметной области, заданной ретроспективным паттерном. В качестве опорной функции использовать функцию: y = a0 + a1 xi + a2 xj . В качестве обучающей выборки взять первые 20 значений паттерна, в качестве тестовой выборки, оставшиеся 5 паттернов исходной таблицы. Обучающую выборку поделить на две в соотношении: 60% и 40% (непосредственно обучающая выборка и проверочная выборка для отбора по МГУА). Провести сравнение значений исходной модели и модели, построенной по МГУА. Результат сравнения представить в таблице. Построить график значений исходной модели и модели, построенной по МГУА. Просчитать среднюю ошибку аппроксимации и сделать вывод о качестве обученной модели по методы МГУА.

# Ход работы

Для выполнения первой части был использован язык программирования Python в приложении PyCharm. Для корректной работы кода была создана таблица значений на основании приведённой в задании (см. Рисунок 1).

Изображение выглядит как текст, число, меню, Шрифт

Автоматически созданное описание

Рисунок 1. Таблица «Исходные данные»

Для выполнения всех дальнейших требований задания был написан единый код (см. Листинг 1), который выводит среднюю ошибку аппроксимации, таблицу сравнения (фактические значения, предсказания модели МГУА, ошибка) и оценку качества модели. Помимо этого, реализован вывод графика значений исходной модели и модели, построенной по МГУА.

Листинг 1. Код для выполнения задания

|  |
| --- |
| import pandas as pd  import numpy as np  from sklearn.model\_selection import train\_test\_split  from sklearn.linear\_model import LinearRegression  from sklearn.metrics import mean\_squared\_error  import matplotlib.pyplot as plt  # Загрузка данных из файла  data = pd.read\_excel('Исходные данные.xlsx')  # Разделение данных на обучающую (первые 20 строк) и тестовую выборки (последние 5 строк)  train\_data = data.iloc[:20]  test\_data = data.iloc[20:]  # Разделение обучающей выборки на 60% и 40%  X\_train\_full = train\_data[['x1', 'x2', 'x3', 'x4']]  y\_train\_full = train\_data['y']  X\_train, X\_val, y\_train, y\_val = train\_test\_split(X\_train\_full, y\_train\_full, test\_size=0.4, random\_state=42)  # Используем лучшие признаки  best\_i, best\_j = 0, 1 # Это можно улучшить  X\_train\_best = X\_train.iloc[:, [best\_i, best\_j]].to\_numpy()  X\_val\_best = X\_val.iloc[:, [best\_i, best\_j]].to\_numpy()  X\_test\_best = test\_data[['x1', 'x2', 'x3', 'x4']].iloc[:, [best\_i, best\_j]].to\_numpy()  # Обучение модели на выбранных признаках  model\_mgua = LinearRegression()  model\_mgua.fit(X\_train\_best, y\_train)  # Предсказание на валидационной выборке  y\_val\_pred = model\_mgua.predict(X\_val\_best)  val\_mse = mean\_squared\_error(y\_val, y\_val\_pred)  # Предсказание на тестовой выборке для модели по МГУА  y\_test\_pred\_mgua = model\_mgua.predict(X\_test\_best)  # Формируем таблицу для сравнения  comparison\_table = pd.DataFrame({  'Фактические значения': test\_data['y'],  'Предсказания модели МГУА': y\_test\_pred\_mgua  })  # Подсчет средней ошибки аппроксимации  comparison\_table['Ошибка'] = np.abs((comparison\_table['Фактические значения'] - comparison\_table['Предсказания модели МГУА']) / comparison\_table['Фактические значения'])  mean\_approx\_error = comparison\_table['Ошибка'].mean() \* 100  # Вывод средней ошибки аппроксимации  print(f'Средняя ошибка аппроксимации: {mean\_approx\_error:.2f}%')  # Вывод таблицы  print(comparison\_table)  # Оценка качества модели  if mean\_approx\_error < 10:  print("Модель имеет высокое качество (низкая ошибка аппроксимации).")  elif mean\_approx\_error < 20:  print("Модель имеет удовлетворительное качество.")  else:  print("Модель имеет низкое качество (высокая ошибка аппроксимации).")  # Построение графика  plt.figure(figsize=(10, 6))  # Фактические значения  plt.plot(comparison\_table.index, comparison\_table['Фактические значения'], label='Фактические значения', marker='o')  # Предсказания модели МГУА  plt.plot(comparison\_table.index, comparison\_table['Предсказания модели МГУА'], label='Предсказания модели МГУА', marker='x')  # Настройка графика  plt.title('Сравнение фактических значений и предсказаний модели МГУА')  plt.xlabel('Индекс')  plt.ylabel('Значение')  plt.legend()  plt.grid(True)  # Отображение графика  plt.show() |

Далее будут представлены выводы, которые были получены в результате работы данного кода. Первый вывод представлен на рисунке 2. Здесь выводится средняя ошибка аппроксимации, таблица сравнения (фактические значения, предсказания модели МГУА, ошибка) и оценка качества модели.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 2. Вывод результата работы программы

Далее выводится график значений исходной модели и модели, построенной по МГУА. Результат представлен на рисунке 3.

Изображение выглядит как линия, диаграмма, График, текст

Автоматически созданное описание

Рисунок 3. График значений исходной и обученной модели