Выполнил:

студент группы УВП-412

Рогов К.Д.

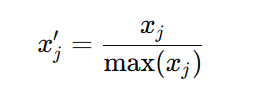
**Задание № 4**

**Нормировка**

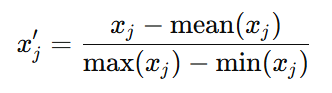
**Задание**

Выполнить нормировку признаков тремя способами. Визуализировать исходные признаки и нормированные. Среднее знач и СКО вычислять явно по определению и используя стандартные ф-и питона.

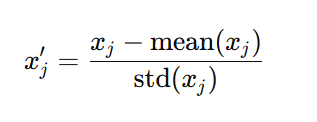
**Теоретическая часть**

1. Нормализация (max)  
   Формула:

Используется максимальное значение в столбце для масштабирования значений в интервал [0,1] .

1. Нормализация (max-min)  
   Формула:

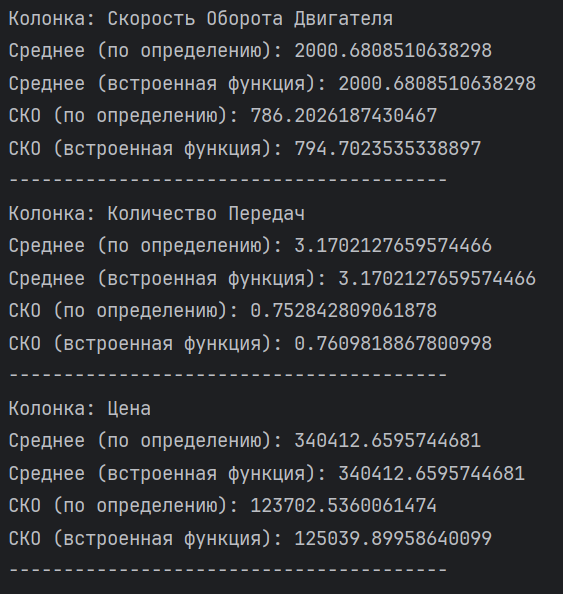
Эта нормализация центрирует данные вокруг среднего, затем масштабирует их в зависимости от диапазона (max−min\text{max} - \text{min}max−min).

1. Нормализация (стандартное отклонение)  
   Формула:

Стандартизация делает среднее значение равным 000 и приводит данные к нормированному масштабу с единичным стандартным отклонением.

**Реализация кода**

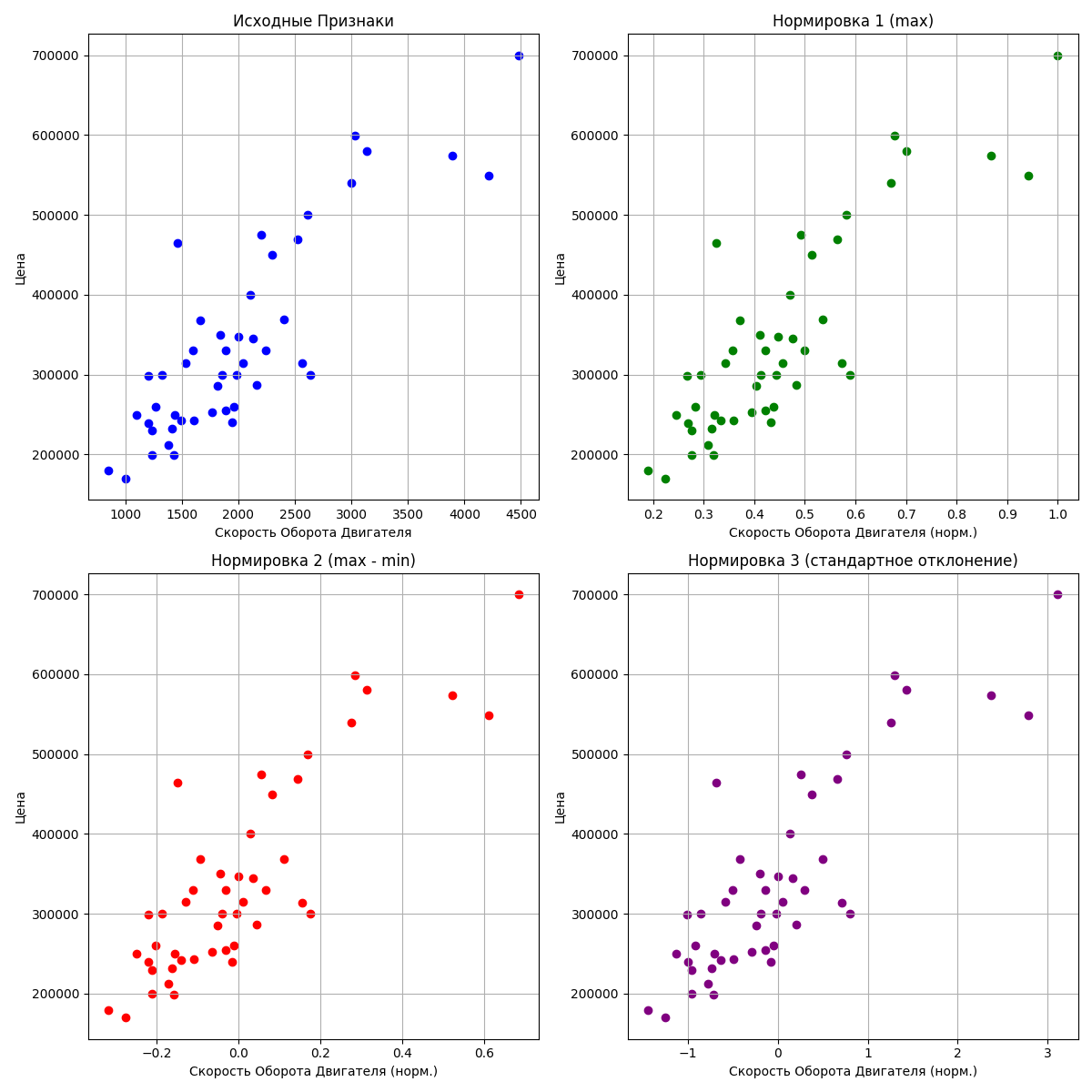
1. Ввод данных:  
   Загружается CSV-файл с признаками и целевой переменной (цена).
2. Реализация формул:  
   Каждая функция нормализации (normalize\_max, normalize\_range, normalize\_std) реализована отдельно.
   * Вычисление среднего (calculate\_mean) и стандартного отклонения (calculate\_std) реализовано двумя способами:
     + По определению (суммирование и деление на количество элементов).
     + Встроенные методы pandas (mean(), std()).



1. Нормализация данных:  
   Нормализация применяется к каждому признаку с использованием .apply().
2. Визуализация:  
   Исходные данные и нормализованные данные визуализируются на графиках:
   * Первый график: Исходные данные.
   * Второй график: Нормализация с использованием max.
   * Третий график: Нормализация с использованием max-min.
   * Четвертый график: Нормализация с использованием стандартного отклонения.
3. Сохранение графиков:  
   Итоговые графики сохраняются в файл normalized\_features.png.

4. Анализ графиков

Графики идентичны, так как в исходных данных есть ярко выраженная линейная зависимость между скоростью оборота двигателя и ценой. Это означает, что изменение признаков пропорционально отражается на целевой переменной. После нормализации, эти пропорциональные отношения сохраняются, потому что все методы нормализации, которые использовались в этом коде (max, max-min, и стандартизация), масштабируют или изменяют масштаб данных, но не меняют их взаимную зависимость.



5. Заключение

1. Все три метода нормализации успешно применены, данные визуализированы.
2. Для задач машинного обучения рекомендуется выбирать метод в зависимости от требований модели:
   * Max-нормализация: подходит для данных с фиксированными ограничениями (например, проценты).
   * Max-min нормализация: хорошо работает при равномерном распределении данных.
   * Стандартизация: предпочтительна для алгоритмов, чувствительных к масштабу (например, линейная регрессия, SVM).

import pandas as pd  
import matplotlib.pyplot as plt  
  
file\_path = "ex1data2.txt"  
data = pd.read\_csv(file\_path, header=None, names=["Скорость Оборота Двигателя", "Количество Передач", "Цена"])  
  
  
def normalize\_max(data):  
 *"""  
 Формула: x'\_j = x\_j / max(x\_j)  
 """* return data / data.max()  
  
  
def normalize\_range(data):  
 *"""  
 Формула: x'\_j = (x\_j - mean(x\_j)) / (max(x\_j) - min(x\_j))  
 """* mean = data.mean()  
 range\_ = data.max() - data.min()  
 return (data - mean) / range\_  
  
  
def normalize\_std(data):  
 *"""  
 Формула: x'\_j = (x\_j - mean(x\_j)) / std(x\_j)  
 """* mean = data.mean()  
 std = data.std()  
 return (data - mean) / std  
  
  
# Вычисление среднего и стандартного отклонения явно по определению  
def calculate\_mean(data):  
 *"""  
 Среднее значение: mean = sum(x\_i) / N  
 """* return sum(data) / len(data)  
  
  
def calculate\_std(data, mean):  
 *"""  
 Стандартное отклонение: std = sqrt(sum((x\_i - mean)^2) / N)  
 """* variance = sum((x - mean) \*\* 2 for x in data) / len(data)  
 return variance \*\* 0.5  
  
  
# Признаки и целевая переменная  
features = data.iloc[:, :-1]  
price = data.iloc[:, -1]  
  
# Добавление целевой переменной в список для обработки  
columns\_to\_process = features.join(price).columns  
  
# Вычисление среднего и СКО по каждой колонке  
for column in columns\_to\_process:  
 column\_data = data[column]  
 mean\_by\_def = calculate\_mean(column\_data)  
 std\_by\_def = calculate\_std(column\_data, mean\_by\_def)  
  
 # Использование встроенных функций pandas  
 # mean\_builtin = column\_data.mean()  
 # std\_builtin = column\_data.std()  
  
 print(f"Колонка: {column}")  
 print(f"Среднее (по определению): {mean\_by\_def}")  
 # print(f"Среднее (встроенная функция): {mean\_builtin}")  
 print(f"СКО (по определению): {std\_by\_def}")  
 # print(f"СКО (встроенная функция): {std\_builtin}")  
 print("-" \* 40)  
  
# Нормализация  
norm\_max = features.apply(normalize\_max, axis=0)  
norm\_range = features.apply(normalize\_range, axis=0)  
norm\_std = features.apply(normalize\_std, axis=0)  
  
# Построение графиков  
plt.figure(figsize=(12, 12))  
  
plt.subplot(2, 2, 1)  
plt.scatter(features.iloc[:, 0], price, color="blue")  
plt.xlabel("Скорость Оборота Двигателя")  
plt.ylabel("Цена")  
plt.title("Исходные Признаки")  
plt.grid()  
  
plt.subplot(2, 2, 2)  
plt.scatter(norm\_max.iloc[:, 0], price, color="green")  
plt.xlabel("Скорость Оборота Двигателя (норм.)")  
plt.ylabel("Цена")  
plt.title("Нормировка 1 (max)")  
plt.grid()  
  
plt.subplot(2, 2, 3)  
plt.scatter(norm\_range.iloc[:, 0], price, color="red")  
plt.xlabel("Скорость Оборота Двигателя (норм.)")  
plt.ylabel("Цена")  
plt.title("Нормировка 2 (max - min)")  
plt.grid()  
  
plt.subplot(2, 2, 4)  
plt.scatter(norm\_std.iloc[:, 0], price, color="purple")  
plt.xlabel("Скорость Оборота Двигателя (норм.)")  
plt.ylabel("Цена")  
plt.title("Нормировка 3 (стандартное отклонение)")  
plt.grid()  
  
plt.tight\_layout()  
plt.savefig("normalized\_features.png")