ORES
Научно-практический электронный журнал
Оригинальные исследования (ОРИС), *ISSN 2222-0402*
www.ores.su
original-research.ru

УДК 004.896

ИСПОЛЬЗОВАНИЕ БИБЛИОТЕКИ NYMPY ДЛЯ ОПРЕДЕЛЕНИЯ НАИХУДШЕГО МОТОРА ДЛЯ КОНВЕЙЕРА

Чеснокова Мария Николаевна
Студент группы ИУК5-42Б Калужского филиала федерального государственного бюджетного образовательного учреждения высшего образования «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)», 248000, Россия, г. Калуга, ул. Баженова, д.2.
chesnokovamn@student.bmstu.ru

Федоров Виктор Олегович
Кандидат технических наук, доцент Калужского филиала федерального государственного бюджетного образовательного учреждения высшего образования «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)», 248000, Россия, г. Калуга, ул. Баженова, д.2.
fedorov_vo@bmstu.ru

Аннотация

В статьи рассматривается решение проблемы определения наихудшего мотора в предприятиях, использующих конвейер. Было предложено использовать библиотеку NumPy.

Ключевые слова: Python, NumPy, определение наихудшего случая

USING THE NUMPY LIBRARY TO DETERMINE THE WORST MOTOR FOR A CONVEYOR BELT

Maria N. Chesnokova
Student of group IUK5-42B of Bauman Moscow State Technical University (Kaluga Branch), 248000, Russia, Kaluga, Bazhenova st., 2.
chesnokovamn@student.bmstu.ru

Fyodorov Viktor Olegovich
Candidate of Technical Sciences, Docent of of Bauman Moscow State Technical University (Kaluga Branch), 248000, Russia, Kaluga, Bazhenova st., 2.
fedorov_vo@bmstu.ru

ABSTRACT

The article discusses the solution to the problem of determining the worst motor in enterprises using a conveyor belt. It was proposed to use the NumPy library.

Keywords: Python, NumPy, determination of the worst case

ВВЕДЕНИЕ

В условиях санкций, наложенных на отечественную промышленность иностранными государствами, которые делают затруднительным, а зачастую и невозможным покупку интересующей продукции или программного обеспечения за рубежом, становится всё более актуальной задача выпуска таких изделий на территории Российской Федерации.

Вместе с тем, роскошь выпускать некачественную продукцию, которая рассчитана на своего покупателя на внутреннем рынке, в современном мире является непозволительной. Отечественные предприятия должны выпускать качественную, конкурентоспособную продукцию, пользуясь современными достижениями в области искусственного интеллекта, программной инженерии, а также средств и способов обработки и анализа данных.

Воздействие компьютерных технологий на производство прямо пропорционально тому, насколько они позволяют воспроизводить в цифровой среде весь поток информации о продукте: от этапа исследования до этапа производства, постпродажного сопровождения и утилизации. Владение такими технологиями радикально меняет правила конкурентной борьбы, в которых такой набор лучших практик, которые уделяют внимание качеству продукта, безусловно, является одним из основополагающих аспектов производства.

Инвестирование в качество означает инвестирование в новые источники экономических преимуществ. Среди наиболее значимых примеров – процессы управления отклонениями в качестве продукции, и снижение или устранение экономических потерь, связанных с низким качеством продукции.

Вместе с этим для того, чтобы выпускать качественную продукцию на предприятиях, использующих конвейер, необходимо также озаботится тем, чтобы моторы на протяжении всей линии работали исправно.

В данной статье рассмотрим предприятие, выпускающее определенные изделия при помощи конвейера. На примере задачи определения наихудшего мотора, продемонстрируем возможность использования библиотеки NumPy для анализа качества конвейерной линии. Анализ данных в таком контексте может помочь выявить источники проблем и принять меры для их устранения, а решение данной задачи позволяет наглядно продемонстрировать процесс анализа данных на примере реальной бизнес-ситуации.

Используемые инструменты

Библиотека, которую мы будем использовать, называется NumPy, которая используется в области науки о данных и машинного обучения. NumPy означает Numeric Python и представляет собой [3] основной пакет для научных расчетов с использованием Python. Это, безусловно, одна из самых больших библиотек для математических и научных расчетов на Python. В целом, NumPy очень полезен для выполнения сложных математических операций в основном на векторах (массивах), но не только. Среди функций NumPy есть эффективные инструменты для выполнения логико-математических операций, операций сортировки, манипуляций с формой и статических операций. В рамках классификации изображений библиотека представляет собой мощный инструмент для представления и предварительной обработки изображений.

Задача определения качественного изделия предполагает использование различных методов анализа данных, таких как моделирование синусоидальных значений, добавление

шума и вычисление среднеквадратичной ошибки. Этот процесс отражает реальные шаги, которые аналитик может предпринять для решения подобных проблем.

Данные включают в себя информацию о множестве моторов для конвейеров и временных отсчетов, что позволяет продемонстрировать эффективность работы с большими объемами данных при помощи библиотеки NumPy.

Использование метода среднеквадратичной ошибки для определения наихудшего мотора для конвейера является важным инструментом анализа данных. Этот метод помогает аналитику выявить искажения в данных и выделить аномалии.

Библиотека NumPy была выбрана для решения этой задачи по нескольким причинам [3]:

Массивы и векторные операции: NumPy предоставляет мощные средства для работы с многомерными массивами данных, что делает его идеальным выбором для обработки данных о моторах.

Удобство математических операций: Библиотека NumPy имеет встроенную поддержку математических функций, таких как синус, которые широко используются в данной задаче для моделирования ожидаемых значений моторов.

Эффективность вычислений: NumPy оптимизирована для выполнения вычислений на многомерных массивах, что позволяет эффективно обрабатывать большие объемы данных, как в данном случае с множеством моторов и временных отсчетов.

Генерация случайных данных [2]: Библиотека NumPy также предоставляет инструменты для генерации случайных данных, которые используются здесь для добавления шума к исходным данным моторов.

Таким образом, использование NumPy упрощает и ускоряет процесс анализа данных [4] и вычисления среднеквадратичной ошибки для определения наиболее отклоняющегося мотора от ожидаемого значения.

Постановка задачи

Пусть существует некое предприятие, завод, выпускающий некую продукцию. Было замечено, что с каждым разом качество произведённой на конвейере продукции падает, и работники завода подозревают, что какой-то из моторов сломался. Всего моторов - AMOUNT_OF_MOTORS штук.

В систему с датчиков поступают показания от всех моторов, причём известно, что если мотор работает правильно, то значения его напряжения должны быть очень близки к sin(t * 10), где t – это время в секундах с момента запуска завода. Всего завод проработал AMOUNT_OF_SECONDS секунд. Однако всё осложняется тем, что датчики несовершенны и искажают данные, так что нельзя просто взять и проверить значения_моторов==sin(t * 10).

Задача состоит в том, чтобы корректно определить, какой из моторов работает хуже всего, то есть наиболее сильно отличается от значения sin(t * 10).

Для решения поставленной задачи, импортируем библиотеку numpy под псевдонимом np. Затем задаем переменные, определяющие количество моторов (AMOUNT_OF_MOTORS), количество секунд (AMOUNT_OF_SECONDS), значение ошибки (ERROR) и масштабный коэффициент индекса (INDEX_SCALE).

Затем создаем массив data, содержащий данные о каждом моторе. Для этого используем синусоидальную функцию np.sin, сдвигаемую на различные интервалы времени для каждого мотора. Для добавления реалистичного шума используем случайную генерацию значений в пределах заданной ошибки. Массив data затем масштабируется с использованием масштабного коэффициента и объединяется с модифицированной версией индексов моторов.

Каждая строчка - данные об одном из моторов. Например, такая таблица [2]:

0 1 2 3
3 1 3 1
2 2 4 4

означала бы, что всего у нас имеется три мотора, показания с которых снимались в течение 4-х секунд. Для идеального мотора строчка должна выглядеть следующим образом: sin(0), sin(10), sin(20), sin(30).

Далее, данные случайным образом перемешиваются для обеспечения случайного порядка. Затем генерируется синусоидальная волна p, предполагаемо представляющая собой идеальное значение для каждого момента времени. Эта волна повторяется для каждого мотора, чтобы сформировать матрицу предсказаний pred.

Используемый метод

Для поиска наиболее/наименее похожегомотора проще всего использовать метод M SE -метод среднеквадратичной ошибки. Среднеквадратическая ошибка (MSE) [1]— это распространенный способ измерения точности предсказания модели. Он рассчитывается как:

$$MSE = \left(\frac{1}{n}\right) * \sum (\text{фактическое} - \text{прогноз})^2$$

где:

n – размер выборки;

фактическое – фактическое значение данных;

прогноз – прогнозируемое значение данных.

Чем ниже значение MSE, тем лучше модель способна точно предсказывать значения.

Данная функция определена для вычисления среднеквадратичной ошибки между исходными данными и предсказаниями. После этого вычисляется среднеквадратичная ошибка для каждого мотора конвейера, а затем находится индекс мотора с наибольшей ошибкой.

В конце кода происходит проверка на то, совпадает ли индекс наихудшего мотора с правильным индексом, и выводится сообщение о выполнении задачи. Этот алгоритм представляет собой метод анализа данных для выявления моторов с наибольшими отклонениями от ожидаемой производительности на основе среднеквадратичной ошибки

Решение задачи

Листинг программы:

```
# Дано
import numpy as np # Импорт библиотеки numpy под псевдонимом np
# На вход подается большая таблица данных
AMOUNT_OF_MOTORS = 100 # Задание количества моторов как 100
AMOUNT_OF_SECONDS = 500 # Задание количества секунд как 500
ERROR = 0.1 # Установка значения ошибки
INDEX_SCALE = 0.0001 # Установка масштабного коэффициента
# Создание массива 'data' с синусоидальными значениями для каждого мотора, используя numpy
data = np.array([np.sin(np.arange(1, AMOUNT_OF_SECONDS))] + [
                np.sin(np.arange(1, AMOUNT_OF_SECONDS) * 10) + np.random.random(AMOUNT_OF_SECONDS - 1) * ERROR for _ in range(AMOUNT_OF_MOTORS - 1)
])
```

```
# использовать data вместо _data - с подчёркиванием массив технический, он нужен для проверки решения потом. А с массивом data мы работаем.
# Добавление масштабирования индексов к массиву 'data' путем объединения с модифицированной версией 'np.arange(AMOUNT_OF_MOTORS)'
data = np.hstack((np.array([np.arange(AMOUNT_OF_MOTORS)]).T * INDEX_SCALE, data))
# Перемешивание массива 'data'
np.random.shuffle(data)
_data = data.copy()# Создание копии массива 'data' и присвоение ее переменной '_data'
print(data) # Вывод массива 'data'

# Решение
# Генерация синусоиды 'p' и создание матрицы предсказаний 'pred', повторяя 'p' для 100 моторов
p = np.sin(np.arange(0, 5000, 10))
pred = np.tile(p, (100, 1))
print("pred",pred) # Вывод матрицы 'pred'
# Определение функции MSE, которая вычисляет среднеквадратичную ошибку между 'data' и 'pred' по оси 1
def MSE(data,pred):
   return np.mean((data - pred)**2,axis=1)
# Расчет среднеквадратичных ошибок для каждого мотора и сохранение их в 'motors'
motors =  MSE(data,pred)
print(motors)
# Нахождение индекса наихудшего по производительности мотора
worst = np.argmax(motors)
print(worst) # Вывод индекса наихудшего мотора

#Проверка
# Поиск правильных индексов, где первый столбец '_data' равен 0
correct = np.where(_data[:, 0] == 0)
print(correct) # Вывод правильных индексов
# Проверка, что индекс наихудшего мотора совпадает с правильным индексом и вывод сообщения об ошибке, если они не совпадают
assert worst == correct, f"Правильный ответ был {correct}, ваш ответ был {wost}"
print("Задача выполнена!") # Вывод "Задача выполнена!"
```

Проверка решения задачи на тестовых моторах

Листинг программы:

```
import numpy as np
AMOUNT_OF_MOTORS = 3 # Задание количества моторов как 10
AMOUNT_OF_SECONDS = 4 # Задание количества секунд как 4
# Создание массива с данными о моторах
data = np.array([
   [0.0021,-0.52705146,0.96975833,0.05379639],
   [0.0083,-0.53482104,0.98779271,0.03854533],
   [0.004,-0.47064159,0.94622065,0.04270474]
])
#print(data)
```

```
 p = np.sin(np.arange(0, 40, 10))
pred = np.tile(p, (data.shape[0], 1))
print(pred)
def MSE(data,pred):
  return np.mean((data - pred)**2,axis=1)
 motors =  MSE(data,pred)
print("motor",motors)
worst = np.argmax(motors)
print(worst)
```

ЗАКЛЮЧЕНИЕ:

В данной статье, основанной на представленной задаче анализа данных с производства, был предложен способ решения задачи, которая заключается в определении неисправного мотора конвейера на основе показаний с датчиков, при условии наличия зашумленности в данных.

С помощью библиотеки NumPy и с использованием метода среднеквадратичной ошибки (MSE) аналитик может эффективно выявить наиболее отклоняющийся от желаемых значений мотор конвейера и сделать вывод о его неисправности. Представленные данные о значениях напряжения с разных моторов за определенное время обеспечивают возможность провести анализ и выявить техническую проблему.

Структурированный подход к анализу данных, включая моделирование синусоидальных значений, добавление шума и вычисление ошибки, демонстрирует важность применения методов анализа данных в решении реальных проблем производства.

Список литературы:

1. Как рассчитать среднеквадратичную ошибку (MSE) в Python – [Электронный ресурс]. – URL: https://www.codecamp.ru/blog/mean-squared-error-python/ (дата обращения: 05.04.2024)
2. Самоучитель по Python для начинающих. Часть 24: Основы работы с NumPy – [Электронный ресурс].– URL: https://proglib.io/p/samouchitel-po-python-dlya-nachinayushchih-chast-24-osnovy-raboty-s-numpy-2023-07-10 (дата обращения: 05.04.2024)
3. Яворски Михал, Зиаде Тарек. - Python. Лучшие практики и инструменты - [Электронный ресурс]. – URL: https://ibooks.ru/bookshelf/376831 (дата обращения: 05.04.2024)
4. NumPy-STL. - [Электронный ресурс]. - URL: https://github.com/WoLpH/numpystl/ (дата обращения: 05.04.2024)

References:

1. How to calculate mean squared error (MSE) in Python – [Electronic resource]. – URL: https://www.codecamp.ru/blog/mean-squared-error-python/ (accessed on April 5, 2024)

2. Self-study guide to Python for beginners. Part 24: Basics of working with NumPy – [Electronic resource]. – URL: https://proglib.io/p/samouchitel-po-python-dlya-nachinayushchih-chast-24-osnovy-raboty-s-numpy-2023-07-10 (accessed on April 5, 2024)
3. Jaworski Michal, Ziade Tarek. Python. Best practices and tools – [Electronic resource]. – URL: https://ibooks.ru/bookshelf/376831 (accessed on April 5, 2024)
4. NumPy-STL. – [Electronic resource]. – URL: https://github.com/WoLpH/numpy-stl/ (accessed on April 5, 2024).