МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по практической работе

по дисциплине «Анализ потоковых данных»

Тема: Имитация потоковых данных

Студент гр. 8304 _____ Зуев Д.В.

Преподаватель _____ Бекенева Я.А.

Санкт-Петербург

2023

Цель работы.

Приобретение навыков работы с потоковыми данными.

Основные теоретические положения.

Потоковые данные – это данные, непрерывно генерируемые тысячами источников данных, которые обычно отправляют записи данных одновременно и небольшими объемами (по несколько килобайтов).

Выполнение работы.

1. Источник данных

В качестве источника данных был выбран набор данных Wine Quality Dataset [1]. Набор данных описывает количество различных химических веществ, присутствующих в вине, и их влияние на его качество. И качество этого вина, определенное экспертами. Набор данных представлен в виде csv-файла и является источником накопленных данных.

Для того, чтобы работать с набором данных, как с источником потоковых данных был написан скрипт [2] на языке программирования Python, который делит данные на пакеты и отправляет в брокер сообщений.

Разбиение данных производилось начиная с 10 записей в одном пакете, заканчивая 510 записями с шагом 20 записей.

2. Передача данных.

Для передачи потоковых данных был выбран протокол AMPQ, так как для работы с ним написано достаточно библиотек на языке программирования Python.

Среди брокеров сообщений, которые работают с протоколом AMPQ, наибольшую популярность, а следовательно, большое сообщество и достаточно подробно описанную документацию имеют Apache Kafka, Amazon SQS и RabbitMQ.

Брокер Apache Kafka были отброшены, так как имеют гораздо больший функционал, чем это необходимо для приобретения навыков работы с потоковыми данными. Помимо базовых операций добавления и доставания сообщения из очереди они предоставляют механизм реплицирования, репликации и хранения сообщений, которые были прочитаны консьюмером.

Для передачи потоковых данных был выбран брокер сообщений RabbitMQ версии 3.10.7. Запуск брокера на локальной машине производится с использованием Docker контейнера командой

```
docker run --name rabbitmq --rm -p 5672:5672 rabbitmq:3.10.7-management
```

Потоковые данные отправляются по порядку в синхронном режиме в очередь `wine-quality`.

3. Алгоритм обработки.

В качестве инструмента для обработки данных используется алгоритм обучения модели линейной регрессии, которая должна предсказывать качество вина по его характеристикам.

Обучение модели выполняется с использованием библиотеки pytorch. Для обучения используется оптимизатор Adam. В качестве метрики ошибки используется средняя абсолютная ошибка:

$$\mathrm{MAE} = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n} = \frac{\sum_{i=1}^{n} |e_i|}{n}$$

Для оценки точности предсказания модели производится округление предсказанного значения до целочисленного и сравнение его с действительным.

Обучение производится по потокам данных, каждый обучает модель отдельно. Экспериментальным путем было выяснено, что наибольшую точность модели можно получить, обучая модель по три эпохи на каждом потоке.

4. Обработка данных.

Потоковые данные считываются из брокера сообщений синхронно, чтобы модель обучалась последовательно на каждом пакете. Далее производится приведение данных к виду (тензору), который принимает модель.

Каждый пакет в потоке данных обучает модель отдельно. После каждого пакета модель можно использовать для предсказания данных, она хранится в состоянии программы. При желании модель можно сохранять на диске и использовать ее в других программах.

Для удобства анализа модели в конце каждого потока посылается сигнал об окончании этого потока.

Код программы представлен в [3].

5. Сравнение полученных моделей.

Для анализа влияния разбиения данных на потоки по количеству данных в потоке были использованы общее время работы алгоритма на потоке, ошибка полученной модели и ее точность.

Время работы алгоритма в зависимости от разбиения представлено на рис. 1.

Рисунок 1 – Время работы алгоритма

Ошибка полученной модели в зависимости от разбиения представлено на рис. 2.

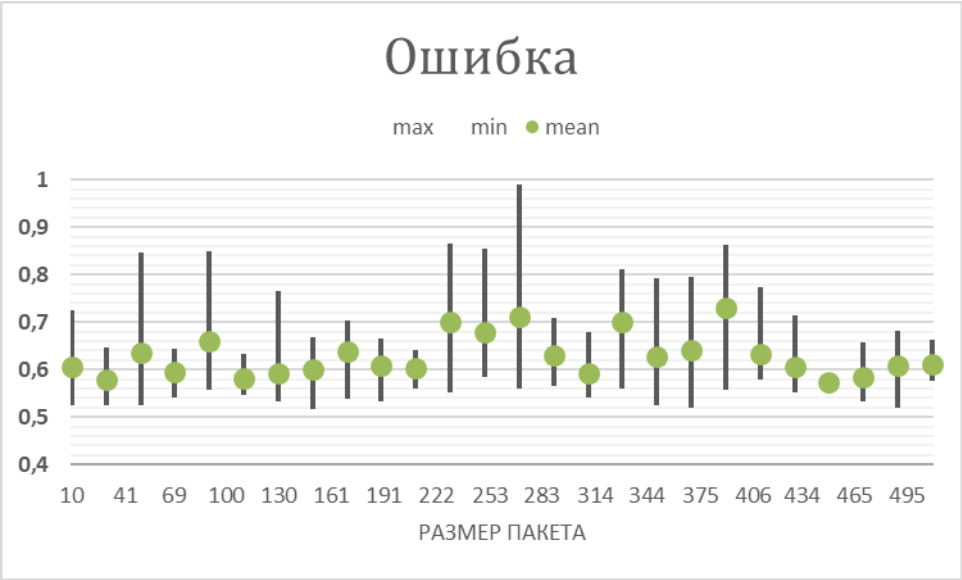

Рисунок 2 – Ошибка модели

Точность полученной модели в зависимости от разбиения представлено на рис. 1.

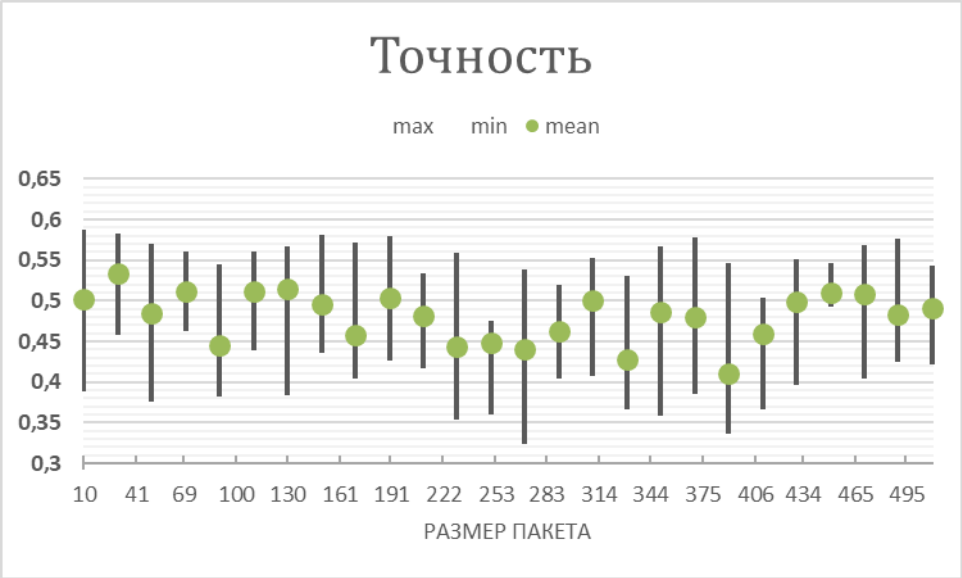

Рисунок 3 – Точность модели

По результатам обработки потоковых данных видно, что при увеличении размера пакета увеличивается время обработки его алгоритмом, причем увеличение времени происходит скачком на количестве записей в пакете равном 170. Это связано с увеличением времени приведения данных к виду, который принимается моделью, так как операции выделения памяти на диске, чтения и записи на него занимают относительно обучения модели достаточно

продолжительное время. Так же, так как увеличение времени произошло скачкообразно, вероятно, внутренние процессы в языке программирования Python изменили алгоритм управления памятью.

С увеличением размера пакета так же произошло падение точности и увеличение ошибки, особенно при размерах пакета, начиная с двухсот и заканчивая четырьмястами. Такое падение может быть обусловлено большей разнородностью в данных. При малом размере пакета данные поступают в алгоритм обработки более равномерно, чем при большом.

Выводы.

В результате выполнения работы были приобретены навыки развертывания на локальной машине работы с брокером сообщений RabbitMQ. Была реализована генерация потоковых данных из набора накопленных данных.

Для обработки потоковых данных был разработан алгоритм, который обучает модель линейной регрессии. Был проведен сравнительный анализ, по результатам которого выявлено преимущество более мелкого разбиения данных на потоки.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Wine Quality Dataset // kaggle. URL: https://www.kaggle.com/datasets/yasserh/wine-quality-dataset/data (дата обращения: 15.12.2023).

2. Файл send.py // github. URL: https://github.com/danilzuev21/streaming-data-analysis/blob/main/send.py (дата обращения: 26.12.2023).

3. Файл receive.py // github. URL: https://github.com/danilzuev21/streaming-data-analysis/blob/main/ receive.py (дата обращения: 26.12.2023).