# РЕФЕРАТ

Магистерская диссертация содержит 42 страницы, 17 рисунков, 2 таблицы, 25 использованных источников.

МАШИННОЕ ОБУЧЕНИЕ, АНАЛИЗ ДАННЫХ, ПОИСК АНОМАЛИЙ, ЯЗЫК SCALA, HDFS, ЯЗЫК PYTHON, SPARK.

В магистерской диссертации собраны и представлены требования к системе автоматического поиска аномалий; спроектированы: архитектура системы, её компоненты и средства интеграции компонентов; представлена реализация модели.

Реализованная модель ускоряет процесс анализа и поиска аномалий человеком, сокращая объем данных путем применения модели: после подготовки данных и применения модели, результаты записываются в таблицу.

СОДЕРЖАНИЕ

# ОПРЕДЕЛЕНИЯ И СОКРАЩЕНИЯ

Перечень определений, условных обозначений, символов, единиц, сокращений и терминов

| | |
|---|---|
| РПО | Регистрируемое почтовое отправление |
| ОПС | Объект почтовой связи |
| Плечо | Путь между двумя объектами почтовой связи |
| Трассировка | Маршрут следования регистрируемого почтового отправления в логистической сети |
| БД | База данных |
| ПО | Программное обеспечение |
| Фреймворк | Программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта. |
| HDFS | Hadoop Distributed File System |
| SQL | Structured query language |
| Датасет | Набор данных |
| Прецедент | Случай (объект выборки), служащий примером |
| Скрипт | Последовательность действий, описанных с помощью скриптового языка программирования |

# ВВЕДЕНИЕ

На данный момент в организации заказчика нет автоматических подходов обнаружения аномалий. Данные анализируются в результате выполнения SQL-запросов. Но учитывая тот факт, что размер данных равен нескольким терабайтам - даже результат агрегирования не позволяет полностью оценить и проанализировать весь объем данных. Была поставлена задача разработать модель, позволяющую выявлять подозрительные, аномальные случаи на разных объектах почтовой связи. Модель должна быть универсальной и применимой к объектам почтовой связи разного типа и назначения. При этом, результатом работы модели должна быть витрина данных. Например, задав индекс объекта и временной промежуток, в течение которого должен быть проведен анализ, эксперт получает некоторое количество подозрительных временных интервалов. Таким образом, анализ сокращается с 20 гигабайт данных до нескольких мегабайт (нескольких подозрительных временных промежутков). Выполненная работа может помочь в анализе, так как предлагаемое решение определяет аномалии разного рода, которые выявляются после агрегирования и представления данных разными способами.

Задача была решена путем использования фреймворка Apache Spark, который позволяет быстро обрабатывать большие потоки данных и содержит библиотеки для машинного обучения, предоставляющие различные алгоритмы. В работе было опробовано несколько подходов и библиотек для анализа и поиска аномалий и в качестве итоговой модели были выбраны градиентный бустинг и нейронная сеть, которые в дальнейшем были реализованы на Scala + Spark.

# ОСНОВНАЯ ЧАСТЬ

# 1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

## 1.1 Цели и задачи работы

Целью работы является реализация автоматического поиска аномалий в объектах почтовой связи. Для достижения цели были поставлены следующие задачи:

- Анализ структуры данных
- Обзор существующих методов, алгоритмов и метрик
- Выбор технологии обработки данных с учетом требований к большому объему и скорости поступления данных
- Реализация прототипа модели определения аномалий
- Тестирование работы прототипа

## 1.2 Проблема детектирования аномалий

Аномалия - отклонение от нормы, от общей закономерности, неправильность [1].

Аномалии отличаются от обычных статистических выбросов тем, что, зачастую, проявляются не редким событием, а образуют вспышку активности, которая может повлечь за собой последующее изменение обычного поведения. Поэтому основные методы обнаружения статистических выбросов не помогают решать подобные задачи без дополнительных преобразований данных путем подходящей группировки.

Существуют три основных вида аномалий: точечные, контекстуальные и коллективные [2]. Точечной аномалией называют событие, которое по поведению ярко выделяется из основной массы данных. Точечные аномалии считаются наиболее легко-определяемыми, так как значения признаков таких событий, как правило, значительно отличается от всех остальных. Такие аномалии можно считать выбросами (Рис. 1.1а). Контекстуальные аномалии похожи на точечные, но при этом они считаются аномальными лишь в определенной контексте. То есть

значения признаков допустимы и могут не выбиваться из значений общей массы, но при этом, при дополнительном атрибуте, таком, например, как время, которое задает положение экземпляра - они (значения признаков) не вписываются в контекст. Пример - временной ряд Рис. 1.1б). Коллективные аномалии возникают тогда, когда несколько связанных объектов, например, последовательная часть данных временного ряда, приобретает нетипичное поведение и становится аномальной по отношению ко всему набору данных (Рис. 1.1в). Некоторые аномалии могут относиться сразу к нескольким группам.


Рис.1.1 Виды аномалий.

Детектирование аномалий - процесс распознавания объектов, случаев, или других данных, которые являются подозрительными на фоне поведения основной массы данных. Наиболее понятными и известными аномалиями считаются такие события как подозрительные операции в банках, нетипичные показания счетчиков оборудования, повышенная активность в интернете и другие, как правило, порожденные мошенническими действиями события, игнорирование которых может повлечь за собой убытки для предприятия, поломку системы, утечку данных и другие неблагоприятные последствия. Таким образом, своевременное выявление аномальных ситуаций предотвращает нанесение негативных последствий разного характера.

## 1.4 Методы решения задач логистики

Научную базу логистики составляет широкий спектр методов, разработанных в рамках различных дисциплин. Наиболее распространенные из них базируются на следующих подходах:

- Математические подходы
- Методы исследования операций
- Кибернетические методы
- Прогностические методы

В математических подходах применяется теория вероятностей; математическая статистика; теория случайных процессов; теория матриц; факторный анализ, математическая логика; теория нечетких множеств и др.

Исследование операций представляет собой линейное, нелинейное и динамическое программирование; теорию игр; теорию статистических решений; теорию массового обслуживания; теорию управления запасами; метод имитационного моделирования; метод сетевого планирования и управления; теория эффективности и др.

Кибернетические подходы разделяются на техническую и экономическую кибернетику. Техническая кибернетика включает в себя теорию больших систем; теорию прогнозирования; общую теорию управления; теорию автоматического регулирования; теорию графов; теорию информации; теорию расписаний и др.

Экономическая кибернетика: теория оптимального планирования; теория эффективности; теория квалиметрии; функционально-стоимостной анализ; методы маркетинговых исследований; менеджмент; теория принятия решений; производственный менеджмент; стратегическое и оперативное планирование; ценообразование; управление качеством; управление персоналом; управление проектами; управление инвестициями; социальная психология; экономика и организация транспорта, складского хозяйства, торговли и др.

Прогностика - это методы экономического прогнозирования; прогнозирование временных рядов; регрессионный и корреляционный анализ; методы логического прогнозирования; экспертные методы и др. Поставленной заказчиком задачей было использование прогностического или математического методов.

## 1.5 Подходы к детектированию аномалий

Существуют три группы задач обнаружения аномалий. Задача выявления “без учителя” подразумевает наличие тестовых данных, в которых отсутствует отметки об аномальности. Предполагается, что большинство экземпляров в наборе данных является нормальным, и претендентами на аномалии становятся экземпляры, которые меньше всего соответствуют остальной части набора данных. Другой класс задач обнаружения аномалий - это обучение “с учителем”. То есть исходный набор данных имеет дополнительный признак, информирующий нас о том, является ли экземпляр “нормальным” или “аномальным” и чаще всего задачи с такими исходными данными решаются путем построения и обучения классификатора - алгоритма машинного обучения, ключевым отличием которого от многих других проблем статистической классификации является несбалансированный характер классов, более подробно о котором будет рассказано в следующем разделе. В обучении с частичным привлечением учителя исходных набор данных содержит информацию лишь о нормальных объектах и не содержит примеров аномальных.

Существуют разные способы определения аномалий, которые можно разбить на несколько различных групп. Это выявление аномалий на основании метрических показателей, кластерный анализ, отклонение от ассоциативных правил, применение алгоритмов машинного обучения и нейронные сети. Рассмотрим наиболее распространенные способы.

### k-NN (метод k-ближайших соседей)

Частым в использовании методом обнаружения аномалий является метод k-NN (метод ближайших соседей) [3]. Алгоритм k-NN - это непараметрический метод, который для тестового примера позволяет выявить k ближайших прецедентов, содержащихся в обучающей выборке (Рис. 1.2). Аномалиями могут быть классифицированы потенциально любые изолированные точки данных. Для работы алгоритма необходимо задать метрику, по которой будет определяться расстояние между объектами выборки. Пример метрик, используемых этим алгоритмом: расстояния Евклида, косинусная мера, расстояние Махаланобиса и

др. Важным замечанием при использовании метода k-NN является необходимость размеченных данных, то есть объекты должны иметь маркер “нормальности/аномальности”.

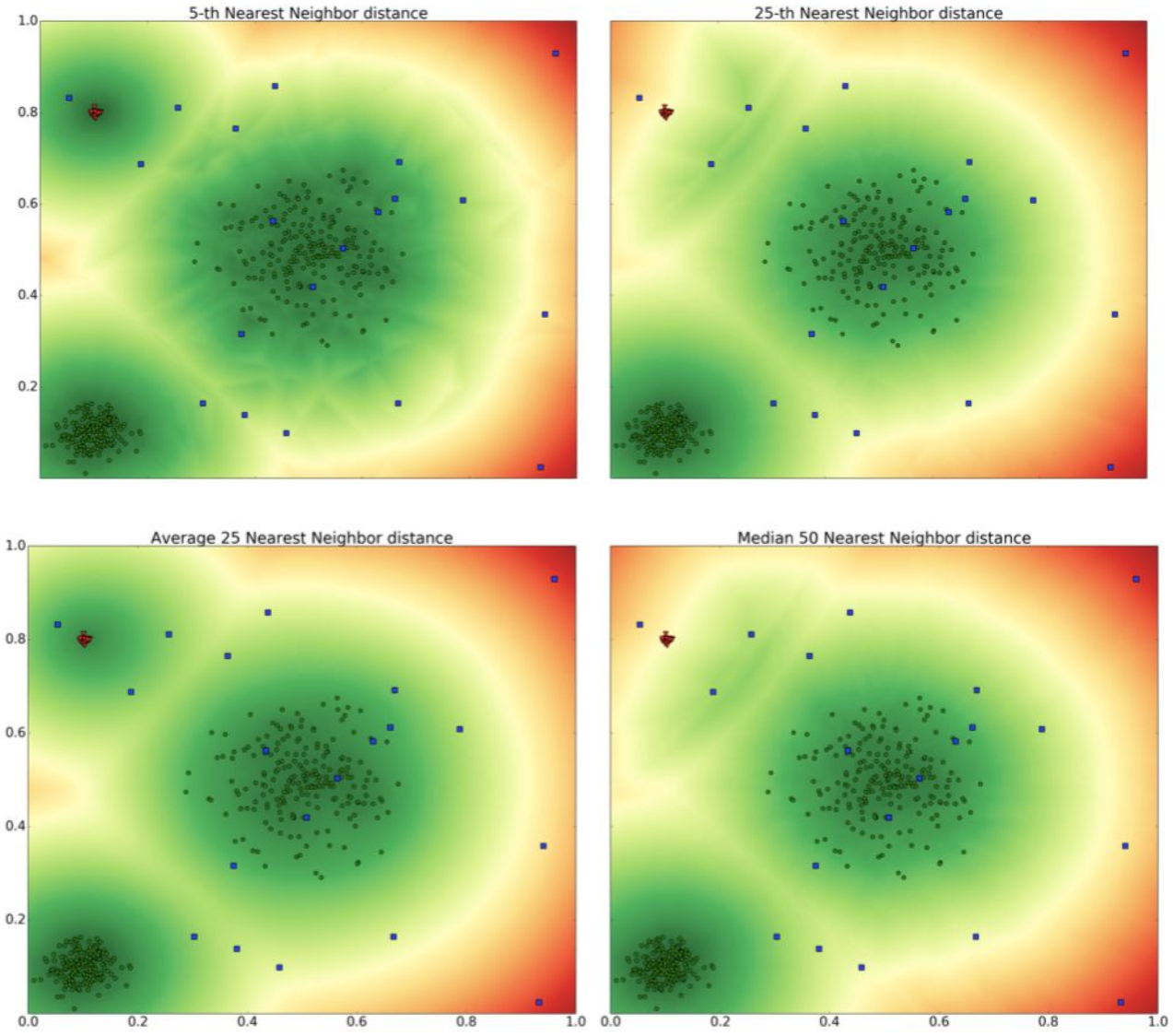


Рис.1.2 Визуализация примера работы k-NN с разными параметрами.

Изолирующий лес

Изолирующий лес устроен по принципу Монте-Карло [4]. В пространстве объектов путем случайного выбора определяется объект, и затем случайным образом происходит выбор порога разбиения, сэмплированного из равномерного распределения на отрезке от минимального значения случайно выбранного признака, до максимального. Поскольку рекурсивное разбиение может быть структурировано в виде дерева, количество разбиений, необходимых для

выделения выборки, эквивалентно длине пути от корневого узла до конечного узла. Эта длина пути, усредненная по лесу таких случайных деревьев, является мерой нормальности и нашей решающей функцией. Случайное разбиение создает более короткие пути для аномалий. Следовательно, когда лес случайных деревьев в совокупности дает более короткие длины пути для конкретных выборок, они с большой вероятностью могут быть аномалиями.

SVM (Support Vector Machine/Метод опорных векторов)[5]

Цель метода опорных векторов состоит в том, чтобы найти такую гиперплоскость в N-мерном пространстве (N - число признаков), которая бы с разделяла точки (объекты выборки) на отдельные классы. Для разделения точек на два класса существует множество возможных гиперплоскостей. Цель метода - найти плоскость, которая имеет максимальное расстояние между точками обоих классов.

Байесовские сети[6]

Метод, использующие байесовские сети применяется в задачах классификации. Основной идеей метода является переход от априорных вероятностей к апостериорным, при этом все признаки и величины в этом подходе считаются случайными. Байесовскую сеть можно представить в виде ациклического графа, в качестве вершин которого выступают признаки, а ребрами - вероятностные зависимости между ними. Элементарной реализацией байесовской сети является наивный байесовский классификатор, который показывает себя как один из самых эффективных способов обработки данных [7].

Кластерный анализ

Кластерный анализ или кластеризация - это задача группирования набора объектов таким образом, чтобы объекты в одной и той же группе (называемой кластером) были более похожи (в некотором смысле) друг на друга, чем объекты в других группах (кластерах). Это основная задача интеллектуального анализа данных и общая методика статистического анализа данных, используемая во

многих областях, включая распознавание образов, анализ изображений, поиск информации, биоинформатику, сжатие данных, компьютерную графику и машинное обучение. Выявление аномалий в подобной группировке происходит в случае выполнения гипотезы о том, что нормальные экземпляры данных располагаются ближе к центрам кластеров, в то время как аномальные находятся на дальнем расстоянии и могут не принадлежать ни одному из кластеров, либо образовывать свой, отдельный кластер. Кластеризация входит в группу обучения “без учителя”. Наиболее распространенные алгоритмы, применяемые в кластерном анализе аномалий - k-средних, k-медиан, DBSCAN и другие.

### Скрытые марковские модели

Скрытые марковские модели основаны на цепи Маркова. Цепь Маркова - это модель, которая сообщает нам о вероятностях последовательностей случайных величин, состояний, каждое из которых может принимать значения из некоторого множества [8]. Эти наборы могут быть множествами слов, тегов или символов, и другими состояниями. Цепь Маркова основана на предположении о том, что если мы хотим предсказать следующее состояние последовательности после текущего, все, что необходимо для этого и имеет значение - это информация о текущем состоянии и все предыдущие состояния никак не влияют на результат прогноза.

### Другие способы нахождения аномалий в задаче классификации

### Нейронные сети, бустинг, случайный лес

В некоторых случаях для поиска аномалий можно использовать классические алгоритмы машинного обучения. Для решения задачи таким способом применяется подход обучения с учителем, необходимым условием которого является наличие разметки исходных данных. Более детальное рассмотрение алгоритмов и частный случай их применения будет рассмотрен в следующей главе, так как в экспериментальной части работы такой подход частично используется.

Спектральные методы

Спектральные методы находят аппроксимацию данных, используя комбинацию атрибутов, которые передают большую часть вариативности в данных. Эта методика основана на следующем предположении: данные могут быть вложены в подпространство меньшей размерности, в котором нормальное состояние и аномалии проявляются иначе. Спектральные методы часто применяются совместно с другими алгоритмами для предобработки данных [9].

Сравнительный анализ методов приведен в таблице.

Таблица 1. Сравнение подходов

| **Метод** | **Результат** | **Режим распознавания** | **Определение класса аномалий** | **Работа без предварительного обучения** |
|---|---|---|---|---|
| **Классификация** | Метка | Supervised, semi- supervised | да | нет |
| **Кластеризация** | Метка | Unsupervised, semi- supervised | нет | нет |
| **Статистический анализ** | Степень | Semi-supervised | нет | нет |
| **Спектральные методы** | Метка | Unsupervised, Semi-supervised | нет | да |

## 1.6 Стек используемых технологий

Для хранения данных компания-заказчик использует HDFS (Hadoop distributed file system) - распределенную файловую систему, предназначенная для хранения файлов больших размеров с возможностью потокового доступа к информации, поблочно распределенной по узлам вычислительного кластера, который может состоять из произвольного аппаратного обеспечения [10]. Hadoop

Distributed File System, как и любая файловая система - это иерархия каталогов с вложенными в них подкаталогами и файлами.

Подготовка данных реализована в СУБД на основе платформы Hadoop - Hive. Hive предоставляет SQL-подобный интерфейс для запроса данных, хранящихся в различных базах данных и файловых системах, которые интегрируются с Hadoop.

Для распределенной работы со слабо структурированными и неструктурированными данными чаще всего используется фреймворк с открытым исходным кодом Apache Spark. Проект, являющийся частью экосистемы Hadoop, написан на языках программирования Scala[11] и Java[12] и обладает отличительной чертой - реализацией работы в парадигме резидентных вычислений, то есть обработка данных происходит в оперативной памяти, что добавляет Spark преимущество по скорости вычислений и делает его привлекательным для работы с алгоритмами машинного обучения[13]. Spark предоставляет программные интерфейсы для работы с языками Scala, Java, R и Python [14].

В ходе выполнения работы были использованы 2 основных языка - Scala и Python. Python использовался для подбора моделей, определения лучших метрик и реализации визуализации. Python - это интерпретируемый объектно - ориентированный язык программирования высокого уровня с динамической семантикой. Его встроенные структуры данных высокого уровня в сочетании с динамической типизацией и динамическим связыванием делают его очень привлекательным для быстрой разработки приложений, а также для использования в качестве скриптового или связующего языка для соединения существующих компонентов. Простой, легкий в освоении синтаксис Python подчеркивает удобочитаемость и, следовательно, снижает стоимость обслуживания программы. Python поддерживает модули и пакеты, что способствует модульности программы и повторному использованию кода.

Интерпретатор Python и обширная стандартная библиотека доступны в исходном или двоичном виде бесплатно для всех основных платформ и могут свободно распространяться. Python является привлекательным языком для реализации алгоритмов машинного обучения и удобным для визуализации результатов. Рассмотрим наиболее популярные библиотеки:

- Numpy
- SciPy
- Scikit-learn
- Theano
- TensorFlow
- Keras
- PyTorch
- Pandas
- Matplotlib

NumPy - это хорошо известный универсальный пакет для обработки массивов. Обширный набор математических функций высокой сложности делает NumPy мощным средством обработки больших многомерных массивов и матриц. NumPy полезен в задачах обработки линейной алгебры, преобразований Фурье и случайных чисел. Другие библиотеки, такие как TensorFlow [15], используют NumPy в бэкэнде для манипулирования тензорами. С NumPy можно определять произвольные типы данных и легко интегрироваться с большинством баз данных. NumPy также может служить эффективным многомерным контейнером для любых общих данных любого типа данных.

Библиотека SciPy [16] предлагает модули для линейной алгебры, оптимизации изображений, интеграционной интерполяции, специальных функций, быстрого преобразования Фурье, обработки сигналов и изображений, решения обыкновенных дифференциальных уравнений и других вычислительных задач в науке и аналитике. Базовая структура данных, используемая SciPy,

представляет собой многомерный массив, предоставленный модулем NumPy. Библиотека SciPy была создана для работы с массивами NumPy и предоставления удобных и эффективных числовых функций.

Scikit-learn имеет широкий спектр supervised и unsupervised алгоритмов обучения. Библиотека также может быть использована для анализа данных. Основными функциями машинного обучения, с которыми может работать библиотека Scikit-learn, являются классификация, регрессия, кластеризация, уменьшение размерности, выбор модели и предварительная обработка.

Theano - это библиотека машинного обучения Python, которая может выступать в качестве оптимизирующего компилятора для работы с математическими выражениями и матричными вычислениями. Построенный на NumPy, Theano демонстрирует тесную интеграцию с NumPy и имеет очень похожий интерфейс. Theano может работать на графическом процессоре (GPU) и центральном процессоре (CPU). Работа на архитектуре GPU дает более быстрые результаты. Theano может выполнять интенсивные вычисления до 140 раз быстрее на GPU, чем на CPU. Theano может автоматически избегать ошибок при работе с логарифмическими и экспоненциальными функциями. Также, имеет встроенные инструменты для модульного тестирования и проверки, что позволяет избежать различных ошибок.

TensorFlow был разработан для внутреннего использования Google командой Google Brain. Это популярная вычислительная среда для создания моделей машинного обучения и нейронных сетей. TensorFlow поддерживает множество различных наборов инструментов для построения моделей на разных уровнях абстракции. Также TensorFlow предоставляет API-интерфейсы для Python и C ++. TensorFlow имеет гибкую архитектуру, с помощью которой он может работать на различных вычислительных платформах: CPU, GPU и TPU.

Keras [17] - это библиотека с открытым исходным кодом, используемая для нейронных сетей и машинного обучения. Keras может работать поверх

TensorFlow, Theano, Microsoft Cognitive Toolkit, R или PlaidML. Keras также может эффективно работать на CPU и GPU. Keras работает с элементами нейронной сети, такими как слои, функции активации и оптимизаторы. Помимо стандартной нейронной сети, Keras поддерживает сверточные и рекуррентные нейронные сети.

PyTorch [18] имеет ряд инструментов и пакетов, которые позволяют решать задачи компьютерного зрения, машинного обучения и обработку естественного языка. Библиотека PyTorch имеет открытый исходный код и основана на библиотеке Torch. Самым значительным преимуществом библиотеки PyTorch может интегрироваться со стеком данных Python, включая NumPy. PyTorch также позволяет разработчикам выполнять вычисления на тензорах. PyTorch имеет надежную структуру для построения вычислительных графиков и даже изменения их во время выполнения. Также PyTorch можно запускать на GPU.

Pandas является самой популярной библиотекой Python, которая используется для анализа данных с поддержкой быстрых и гибких структур данных, предназначенных для работы как с «реляционными», так и с «размеченными» данными. Pandas обладает высокой стабильностью, обеспечивая высокую производительность. Код бэкэнда написан исключительно на C или Python.

Matplotlib - это библиотека визуализации данных, которая используется для построения и создания 2D-графиков и рисунков. Библиотека помогает генерировать гистограммы, графики, диаграммы ошибок, точечные диаграммы всего за несколько строк кода. Она обеспечивает интерфейс, похожий на MATLAB и работает с использованием стандартных инструментов GUI [19].

Заказчиком был установлен основной язык реализации задачи - Scala. Scala, (от Scalable Language - масштабируемый язык) - высокоуровневый язык программирования общего назначения, который статически типизирован. Scala сочетает как функциональное, так и объектно-ориентированное

программирование и написан для работы JVM. Программы Scala компилируются в байт-коды JVM, и их производительность во время исполнения в основном соответствует программам на Java, а в некоторых случаях код Scala компилируется быстрее благодаря усовершенствованному компилятору scalac. Scala позволяет вызывать методы Java, наследовать от классов Java и реализовывать интерфейсы в коде без дополнительного какого-либо специального синтаксиса. Хотя схожесть между Java и Scala очевидна, Scala отличается во многих отношениях тем, что ее код гораздо более краткий и имеет свои преимущества.

Выбор библиотек Scala. Scala допускает использование алгоритмов машинного обучения и нейронной сети посредством обращения к нескольким библиотекам.

Библиотеки для машинного обучения и нейронных сетей для Scala

MLlib - это библиотека машинного обучения Spark (ML)[20]. Ее цель - сделать практическое машинное обучение масштабируемым и легким. На высоком уровне она предоставляет такие Алгоритмы ML, как: общие алгоритмы обучения, такие как классификация, регрессия, кластеризация и коллаборативная фильтрация

Особенности: извлечение, преобразование, уменьшение размерности и выбор функций. Из недостатков можно отметить отсутствие возможности реализации нейронных сетей.

Sparkling Water [21] позволяет комбинировать быстрые, масштабируемые алгоритмы машинного обучения H2O с возможностями Spark. H2O является платформой для машинного обучения в оперативной памяти. Интеграция этих двух сред с открытым исходным кодом обеспечивает беспроблемную работу для пользователей, которые хотят сделать запрос с использованием Spark SQL, передать результаты в H2O, чтобы построить модель и сделать прогнозы, а затем снова использовать результаты в Spark.

Deeplearning4j[22] - это первая коммерческая распределенная библиотека глубокого обучения с открытым исходным кодом, написанная для Java и Scala. DL4J, интегрированный с Hadoop и Apache Spark, позволяет использовать AI в бизнес-средах для использования на распределенных графических процессорах и процессорах CPU.

Краткое сравнение библиотек приведено в таблице ниже (табл. 2). Для работы была выбрана связка Spark.mlib + deepLearning4Java.

Таблица 2. Сравнение библиотек

| | Плюсы | Минусы |
|---|---|---|
| Spark.ML / **Spark.mlib** | алгоритмы машинного обучения | отсутствие нейронных сетей |
| Sparkling Water H2O | поддерживает работу с нейронными сетями, размер библиотеки | новая библиотека, много недоработок и несовместимостей версий |
| **deepLearning4Java (dl4j)** | поддерживает работу с нейронными сетями | размер библиотеки |

Далее осуществляется сборка проекта в jar-файл и его запуск в HDFS. Более подробно полная архитектура решения будет описана в следующем разделе.

# 2. ПРАКТИЧЕСКАЯ ЧАСТЬ

## 2.1 Архитектура решения

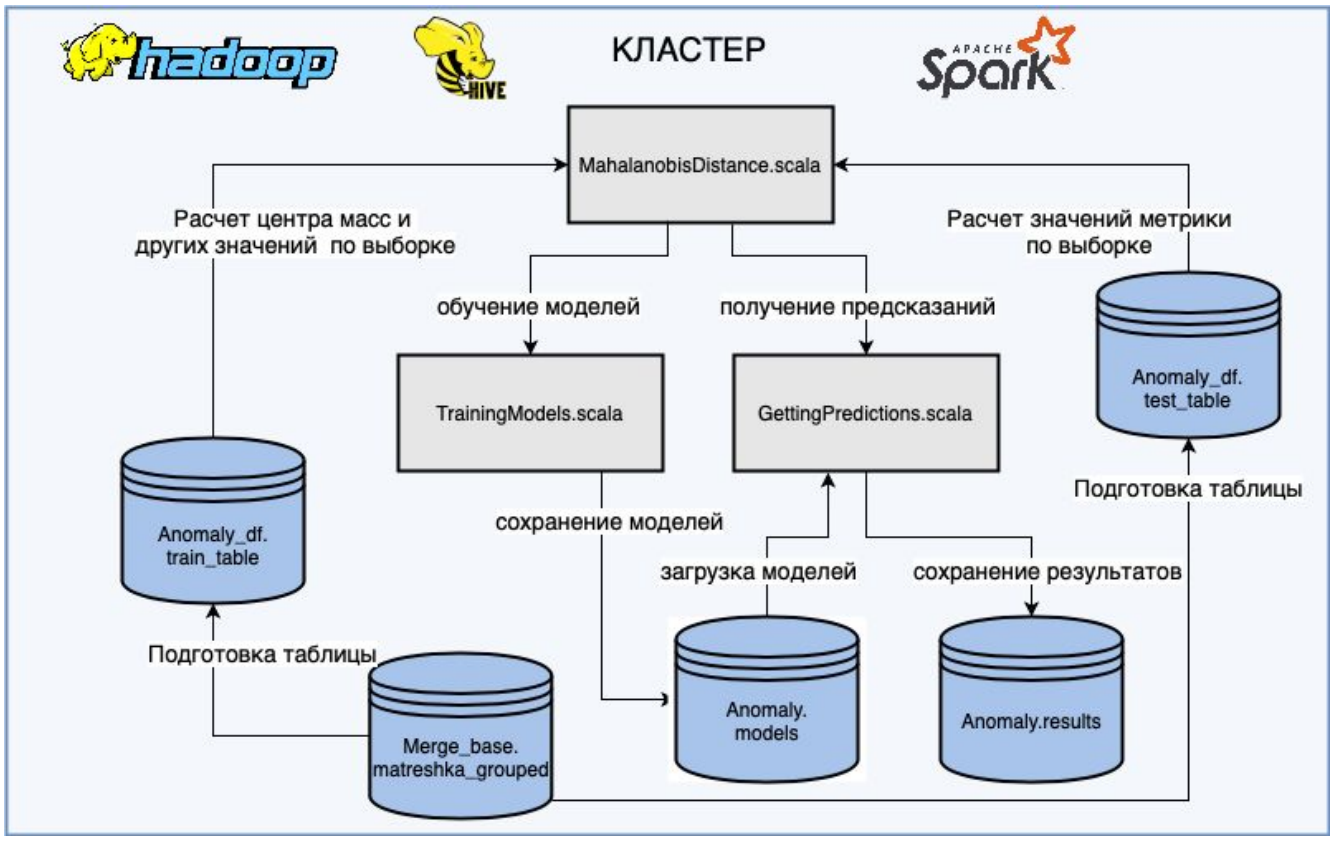


Рис.2.1 Схема архитектуры решения.

На схеме визуализирована архитектура решения (Рис. 2.1). Исходные данные хранятся в HDFS. Сперва в Hive (система управления базами данных на основе платформы Hadoop) на языке запросов SQL происходит создание подготовительных таблиц. Таким образом из первоначального представления данные агрегируются в обучающую и тестовую выборки со структурой временного ряда, строками которой являются временные интервалы с признаками. Следующим шагом данные выгружаются из hdfs, и выполняется один из двух сценариев: обучение и сохранение модели или предсказание. В первом случае сначала рассчитывается центр масс выборки, затем расстояния от каждого объекта до центра. После чего выполняется настройка и обучение модели градиентного бустинга и нейронной сети. Модели сохраняются до дальнейшего востребования. Во втором сценарии выполняется загрузка моделей и

предсказание для тестовых данных. Претенденты на аномальные интервалы записываются в результирующий файл в hdfs.

## 2.2 Описание данных и их подготовка

Мы имеем неразмеченные на аномальные и неаномальные объекты - записи с полями:

- идентификатор отправления
- дата и время операции
- локальные дата и время
- тип операции
- атрибут операции
- индекс ОПС, в котором произошла операция
- индекс следующего ОПС
- код города
- данные о ПО, с которого была совершена отметка о совершении операции
- объект, над которым была совершена операция (РПО, емкость, документ)
- конечное место назначения (часто пустое или неправильно указанное)
- место отправления
- масса
- и др.

Необходимо, чтобы при запросе анализа поведения определенного ОПС в заданный промежуток времени определялись аномальные интервальные промежутки, равные 4-12 часам (задается параметрически). Поэтому нам необходимо привести данные к виду, удобному для совершения обучения и построения прогнозов. Таким образом, данные были агрегированы и имеют следующие поля:

- кол-во РПО, находящихся на пути к объекту
- кол-во принятых РПО
- кол-во покинувших ОПС РПО

- общее количество посещенных РПО объектов (a*)
- среднее количество посещенных (кем?) РПО (чего?) объектов (b*)
- общее количество обменов (чьих?) РПО м/у объектами (c*)
- среднее количество обменов (чьих?) РПО м/у объектами (d*)
- количество РПО с наличием правильного порядка операций "8",'1018' (обработка)
- количество операций '1' (прием)
- количество операций '8' над типом объекта "1" (обработка РПО)
- количество операций '8' над типом объекта "2" (обработка емкости)
- средняя квадратичная ошибка (b*) и (d*)
- средняя квадратичная ошибка (a*) и (c*)
- час окончания интервала
- день недели

Оптимизированная версия запросов состоит из 10 подготовительных таблиц и составляет 350 строк кода.

## 2.3 Подбор моделей и метрик

Анализ аномальной работы объекта (анализ пропускной способности) можно рассматривать, используя разные подходы. Первым подходом была работа с кумулятивными суммами временного ряда. Подготовлены данные по работе объекта за полгода с признаками: количество принятых, обработанных и покинувших ОПС РПО. Дополнительными признаками были скорость обработки (пропускная способность ОПС) и среднее время обработки одного РПО. Проведен анализ и пробный поиск аномалий, который не дал интересных результатов. Следующим шагом было добавление данных по РПО, находящихся на пути к объекту. Выгружены данные по всем операциям на объекте и РПО отправления с предыдущих объектов, находящиеся в пути (рис. 2.2). Для временного ряда с шагом в четыре часа подсчитаны следующие признаки: кол-во обработанных РПО к концу периода, кол-во принятых РПО к концу периода, кол-во РПО на пути к

объекту в период, кол-во РПО в остатках на начало периода, среднее время обработки РПО. Построены графики работы объекта. На основании метрики Евклида были построены доверительные интервалы и выявлены некоторые аномалии. Следующим шагом было добавление информации не только по ОПС, а по всем объектам, имеющим один и тот же border_index (входящих в границы одного ОПС). Так как нам необходимо видеть в том числе не принятые остатки и их возможное накапливание - для РПО учитывается не только прием в пределы границы ОПС, а приемы и обработки в каждый ОПС в пределах одной границы. Это сделано по причине того, что если смотреть только первый прием в границы ОПС и последний выход из границы, все промежуточное время будет выглядеть как одна большая (по времени) обработка, хотя в некоторое время РПО может не обрабатываться и быть не принятым каким-либо ОПС.

На графике представлена кумулятивная сумма признаков:

- зеленый график  - РПО в пути (не принятые)
- синий график - принятые РПО
- желтый график - обработанные РПО

Что может быть аномалией?

- по какой то причине не обрабатывают (низкая скорость обработки)
- не принимают (РПО висит в пути)

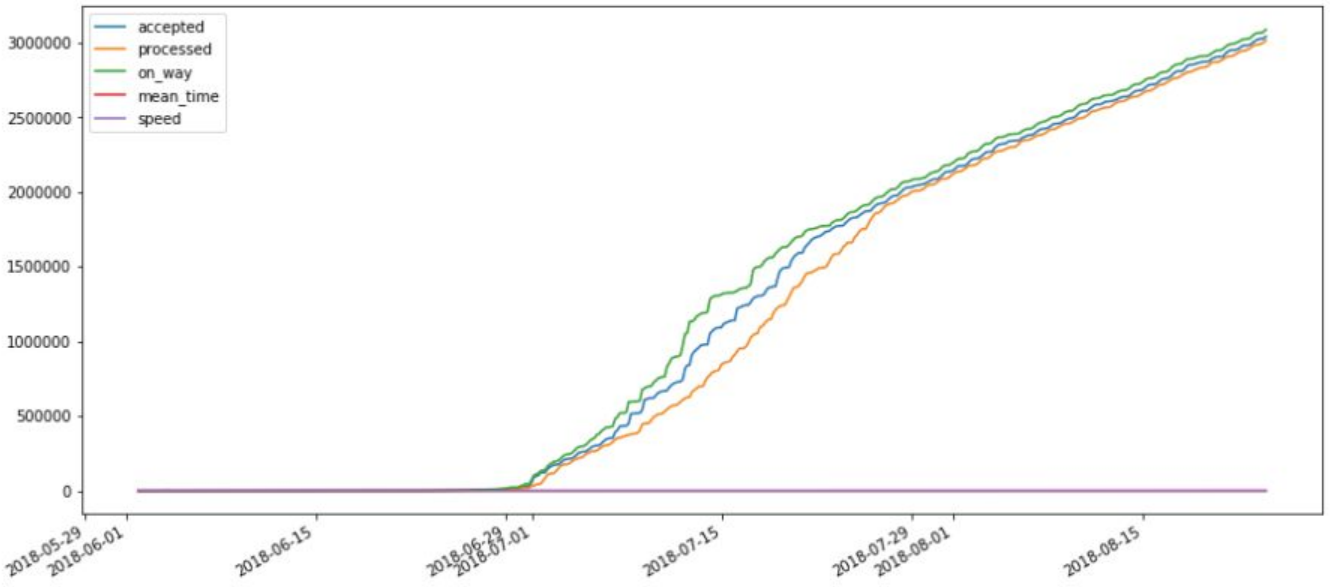

Рис.2.2 Кумулятивная сумма по пяти признакам.

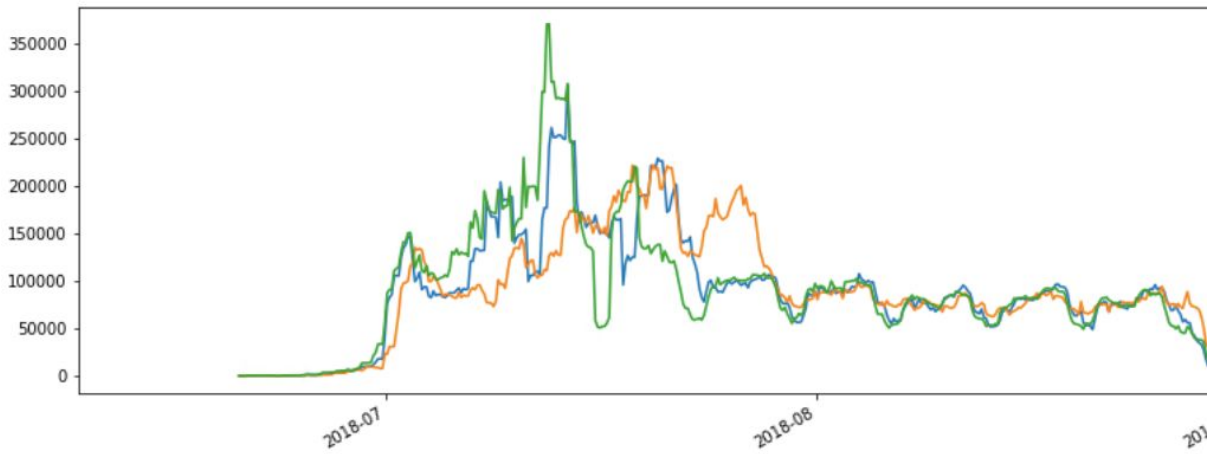


Рис.2.3 Скользящее среднее с окном 1 день.

Распространенные методы поиска аномалий предлагают анализировать одномерные временные ряды. Поставленная в нашей работе задача подразумевает анализ как нескольких временных рядов в отдельности, так и значения в совокупности. Так как, например, случай всплеска активности признака “прием” при анализе отдельно взятого временного ряда может выделить данных случай как аномальный, в то время как рассмотрение двух признаков “прием” и “обработка” в совокупности, при своевременной обработке и соответствующем количестве делает случай вполне обычным и на класс аномальных не претендует. Таким образом задача представления данных сводится к свертке нескольких признаков в одно значение, путем подбора некоторой функции.

Функция, основанная на Евклидовом расстоянии.

$$f(x) = \sqrt{\sum_{j=1}^{m} (x_j - \frac{1}{n} \sum_{i=1}^{n} x_i)^2},$$

где m - количество признаков, n - количество объектов в выборке

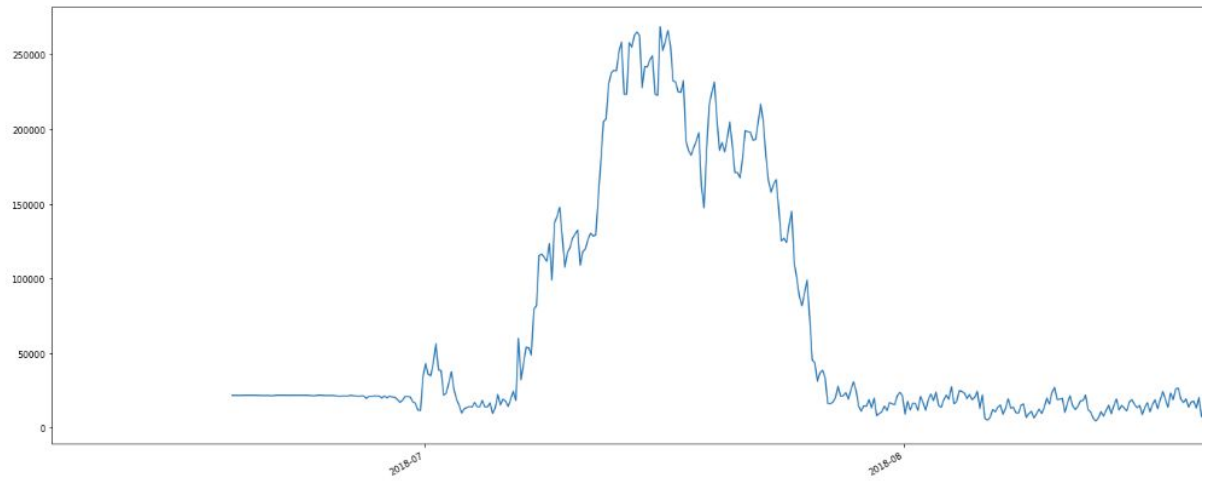


Рис.2.4 Евклидово расстояние между вектором признаков и вектором со средними значениями.

После построения доверительного интервала выявились следующие аномалии (Рис. 2.5).

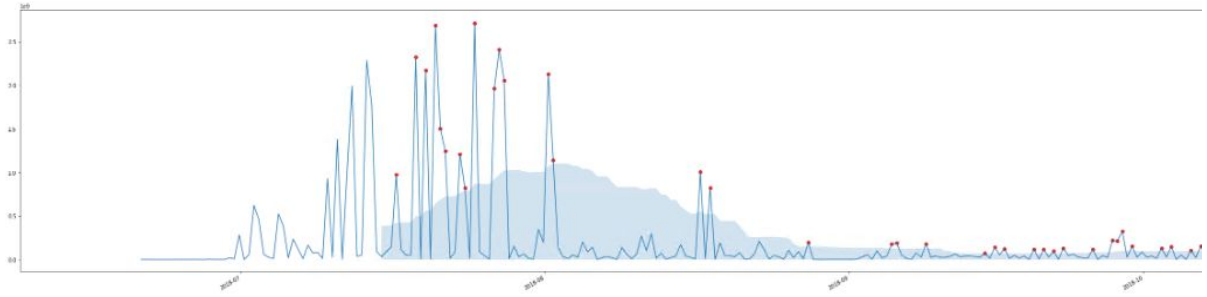

Рис.2.5 Аномалии, вышедшие за пределы доверительного интервала.

Такой способ позволяет выделить разнородные аномалии, но, к сожалению, он также ложно выделяет интервалы, когда разница значений признаков большая. То есть под аномалии попадают не только такие случаи как: "много принято - мало обработано", но и:  "много обработано- мало принято", "много принято, много обработано - мало в пути", что с учетом контекста условий задачи аномалией не является.

Следующий подход: смотреть возрастание и убывание функций и определять интервалы, где остатки не убывают (то есть неизменны или

возрастают), а скорость обработки убывает. На графике (Рис. 2.6) изображены случаи, когда остатки возрастают, а скорость обработки падает:

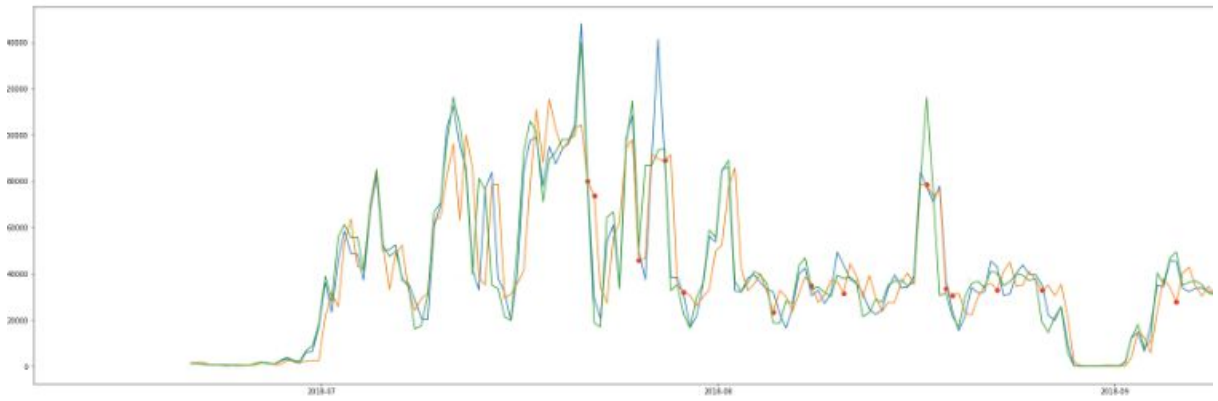

Рис.2.6 Показатели и найденные аномалии.

Одной из попыток поиска аномалий было применение скользящего среднего и подбор функции. Идея со сглаживанием и последующей аппроксимацией не принесла результатов, так как не удалось подобрать функцию, наиболее точно описывающую поведение временного ряда.

Кластеризация

Один из примененных подходов - кластеризация. Была проведена кластеризация объектов методом к-средних (с пересчетом центров масс).

Дополнительно были добавлены несколько новых признаков: количество совершенных операций '8', '1018', в интервале, 'ostatki_8', 'ostatki_1018','displaced_ost_8', 'displaced_ost_1018', 'day', 'hour', 'DayOfTheWeek','WeekDay', 'categories'. Затем, для выделения наиболее значимых признаков и снижения размерности (сохраняем 95% данных) был применен метод главных компонент. Метод главных компонент (principal component analysis, PCA) — один из основных способов уменьшить размерность данных, потеряв наименьшее количество информации. Вычисление главных компонент может быть сведено к вычислению сингулярного разложения матрицы данных или к вычислению собственных векторов и собственных значений ковариационной матрицы исходных данных. Иногда метод главных компонент называют

преобразованием Кархунена - Лоэва или преобразованием Хотеллинга [23]. Для K-means находим оптимальное кол-во кластеров (методом локтя). Разбиваем на кластеры и считаем расстояние точек от центров масс. Затем задаем порог, либо долю в процентах (подозрительных точек) ~5 %. Ниже представлены графики комбинаций разных признаков с разным количеством кластеров (Рис. 2.7-2.10).

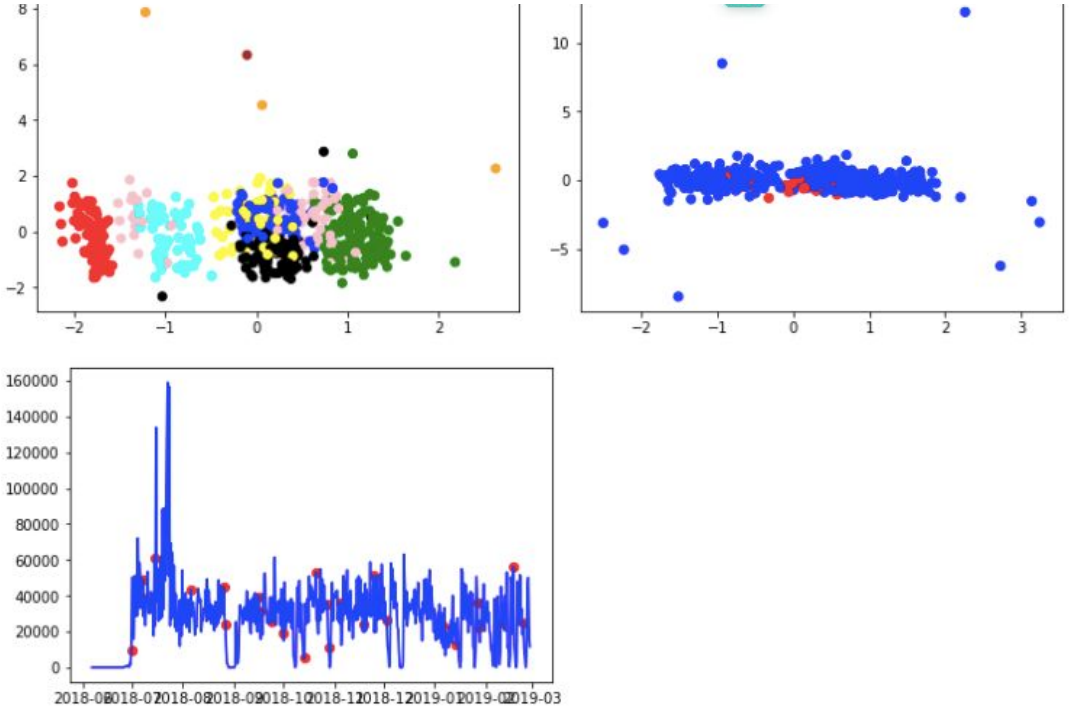

Рис.2.7 Разбиение на 8 кластеров.

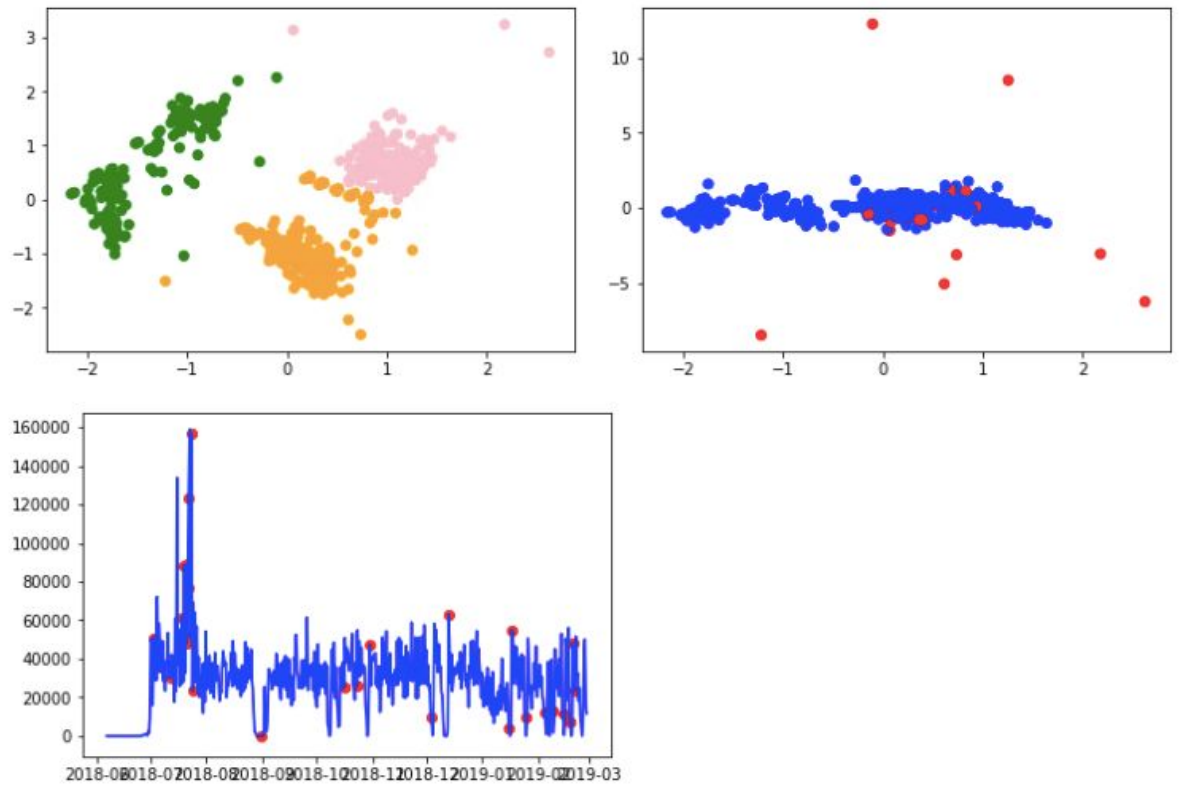

Рис.2.8 Разбиение на 3 кластера.

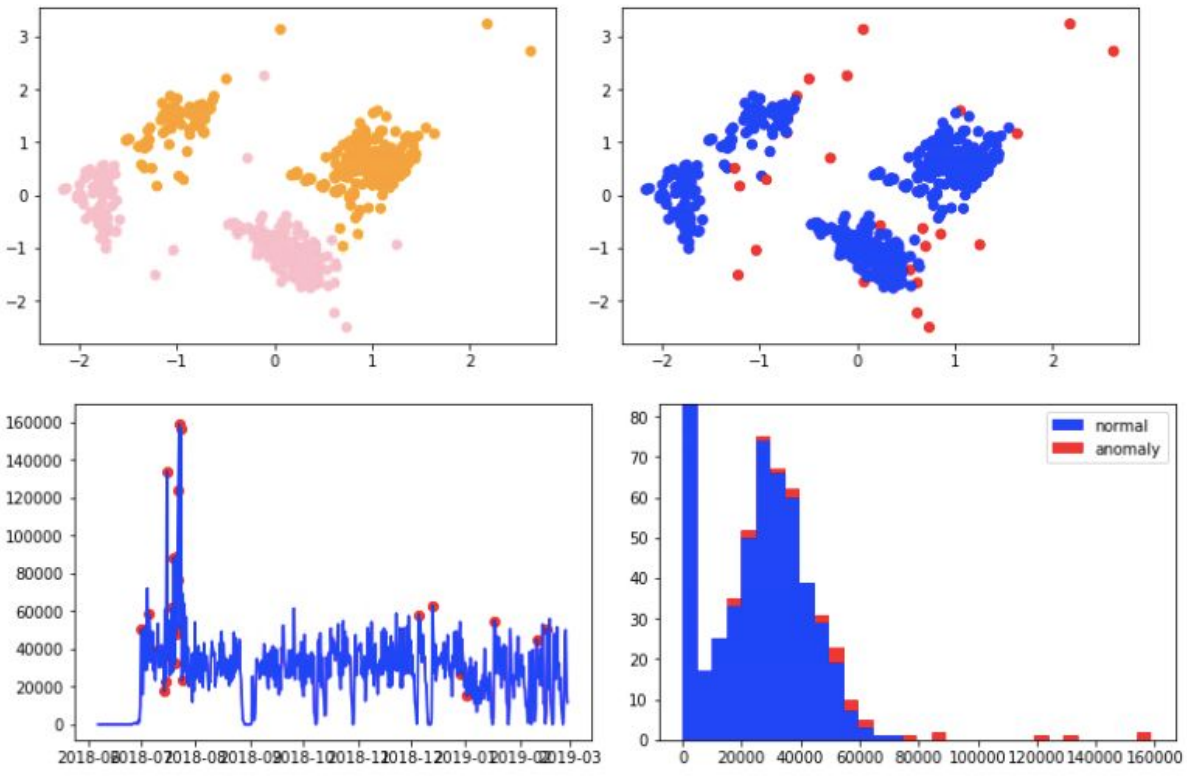


Рис.2.9 Разбиение на 2 кластера.

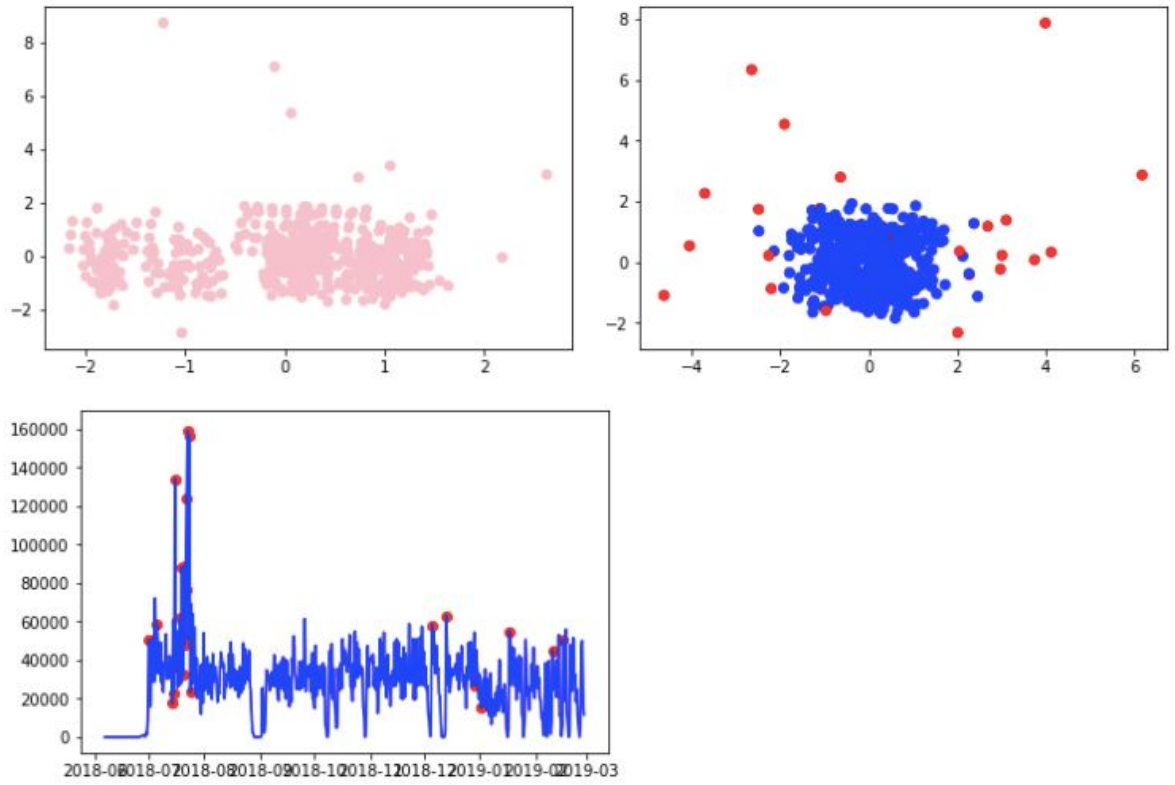


Рис.2.10 Разбиение на 1 кластер.

Подход с кластеризацией позволяет находить аномалии, но имеет существенный минус в контексте нашей задачи. Необходимо реализовать автоматизированную модель поиска аномалий, а кластеризация требует анализа визуализированных результатов или донастройки модели под объект вручную. Также, такой подход не является универсальным, в том смысле, что итоговая обученная модель должна делать предсказания по любому ОПС, идентификатор которого передан параметрически. Поэтому следующим этапом исследований был переход от кластеризации к регрессии и классификации.

Регрессия

Одной из идей для регрессионного подхода было предсказание значения каждого признака по отдельности, основываясь на остальных признаках. Ниже на графиках представлены результаты работы моделей линейной регрессии и градиентного бустинга с данными, целевой переменной в которых был признак “обработанное количество РПО на интервале”. Характеристиками признакового пространства были значения:

- количество принятых РПО
- количество РПО в пути
- количество РПО, покинувших ОПС
- остатки ОПС
- день недели
- ½ половина дня.

На графике синяя ломаная - истинные значения, оранжевая - предсказанные значения признака. Измеряя ошибку метрикой MSE (Mean Squared Error (MSE) - измеряет среднюю сумму квадратной разности между фактическим значением и прогнозируемым значением для всех точек данных. Выполняется возведение во вторую степень, поэтому отрицательные значения не компенсируют положительными) получили медиану ошибки: 4073, среднее: 5091 (Рис. 2.11).

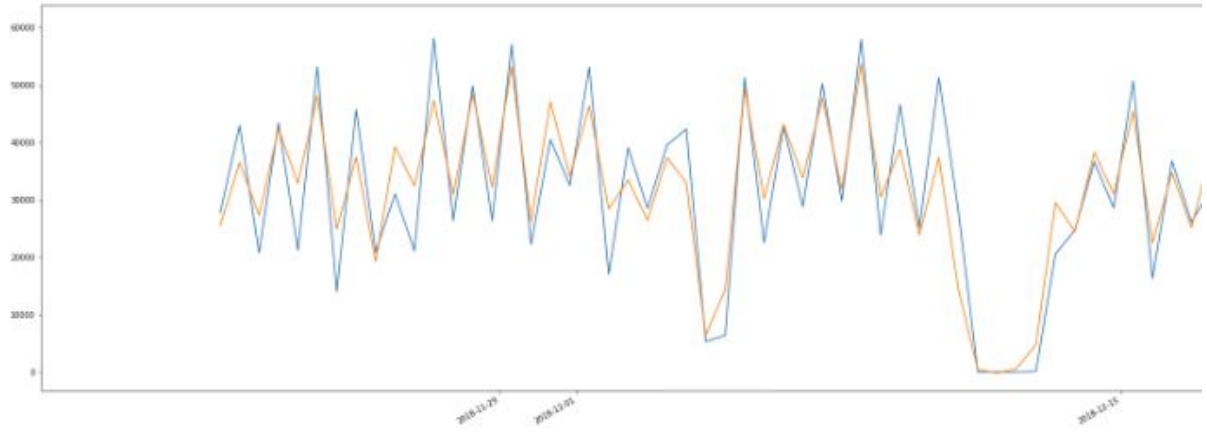

Рис.2.11 График истинных значений и предсказаний модели линейной регрессии.

Для ошибки результатов модели градиентного бустинга получили медиану ошибки: 590, Среднее: 1087. На графике (Рис. 2.12) также заметна большая, чем на предыдущей модели, точность попадания

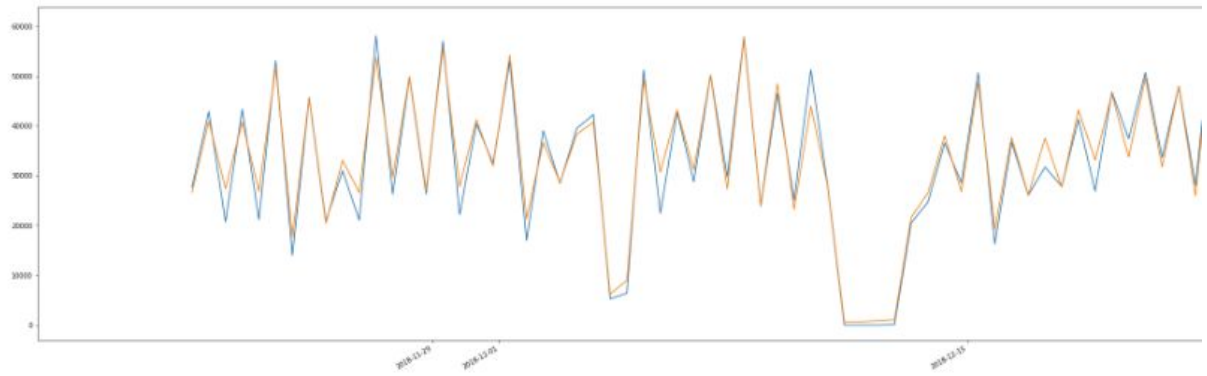

Рис.2.12 График истинных значений и предсказаний модели градиентного бустинга.

Подход с обучением моделей и предсказанием одного признака по остальным показывает неплохие результаты, но также, является не универсальным, поскольку подобное решение подразумевает обучение индивидуальных моделей для каждого ОПС, а это, с учетом общего количества ОПС, - примерно 50 тысяч моделей.

Метрика Махаланобиса и ее предсказание. Одной из мер близости объектов в n-мерном пространстве является расстояние Махаланобиса. Расстояние Махаланобиса - это мера расстояния между точками, или точкой P и распределением множества D, введенное П. Махаланобисом. Формально, расстояние Махаланобиса от многомерного вектора $x = (x_1, x_2, x_3, ..., x_N)^T$ до множества со средним значением $\mu = (\mu_1, \mu_2, \mu_3, ..., \mu_N)^T$ и матрицей ковариации $S$ определяется следующим образом:

$$D_M(x) = \sqrt{(x-\mu)^T S^{-1}(x-\mu)}$$

Расстояние Махаланобиса также можно определить как меру несходства между двумя случайными векторами $\bar{x}$ и $\bar{y}$ из одного распределения вероятностей с матрицей ковариации $S$:

$$d(\bar{x}, \bar{y}) = \sqrt{(\bar{x}-\bar{y})^T S^{-1}(\bar{x}-\bar{y})}$$

Если матрица ковариации является единичной матрицей, то расстояние Махаланобиса становится равным расстоянию Евклида. Если матрица ковариации диагональная (но необязательно единичная), то получившаяся мера расстояния носит название нормализованное расстояние Евклида:

$$d(\bar{x}, \bar{y}) = \sqrt{\sum_{i=1}^{N} \frac{(x_i - y_i)^2}{\sigma_i^2}}$$

Здесь $\sigma$ — среднеквадратичное отклонение $x_i$ от $y_i$ в выборке.

Это многомерное обобщение идеи измерения того, сколько стандартных отклонений от P равно среднему значению D. Это расстояние равно нулю, если P находится в среднем значении D, и увеличивается по мере удаления P от среднего значения вдоль каждой основной составляющая ось. Если каждая из этих осей масштабируется, чтобы иметь единичную дисперсию, тогда расстояние Махаланобиса соответствует стандартному евклидову расстоянию в преобразованном пространстве [24]. Таким образом, расстояние Махаланобиса не имеет единиц измерения и не зависит от масштаба и учитывает корреляции набора данных. Посмотрим на распределения метрики Махаланобиса на разных объектах (Рис. 2.13). На первом графике показано как распределяются значения метрики на объекте, где присутствует превышение остатков в пути и на объекте перегрузило мощности объекта, сортировка работала штатно, с объемом не справлялась - аномалия, определенная экспертом. В ОПС на 2-ом и 4-ом графиках присутствовало замедление сортировки, связанное с производственными причинами (не выход людей, поломка сортировочной машины), что тоже спровоцировало аномальное поведение. В 3-м случае - ОПС работал штатно, объемы поступлений были больше среднего, но в пределах нормы. Таким образом экспертной оценкой был выявлен порог, превышение которого можно было считать подозрением на аномалию.

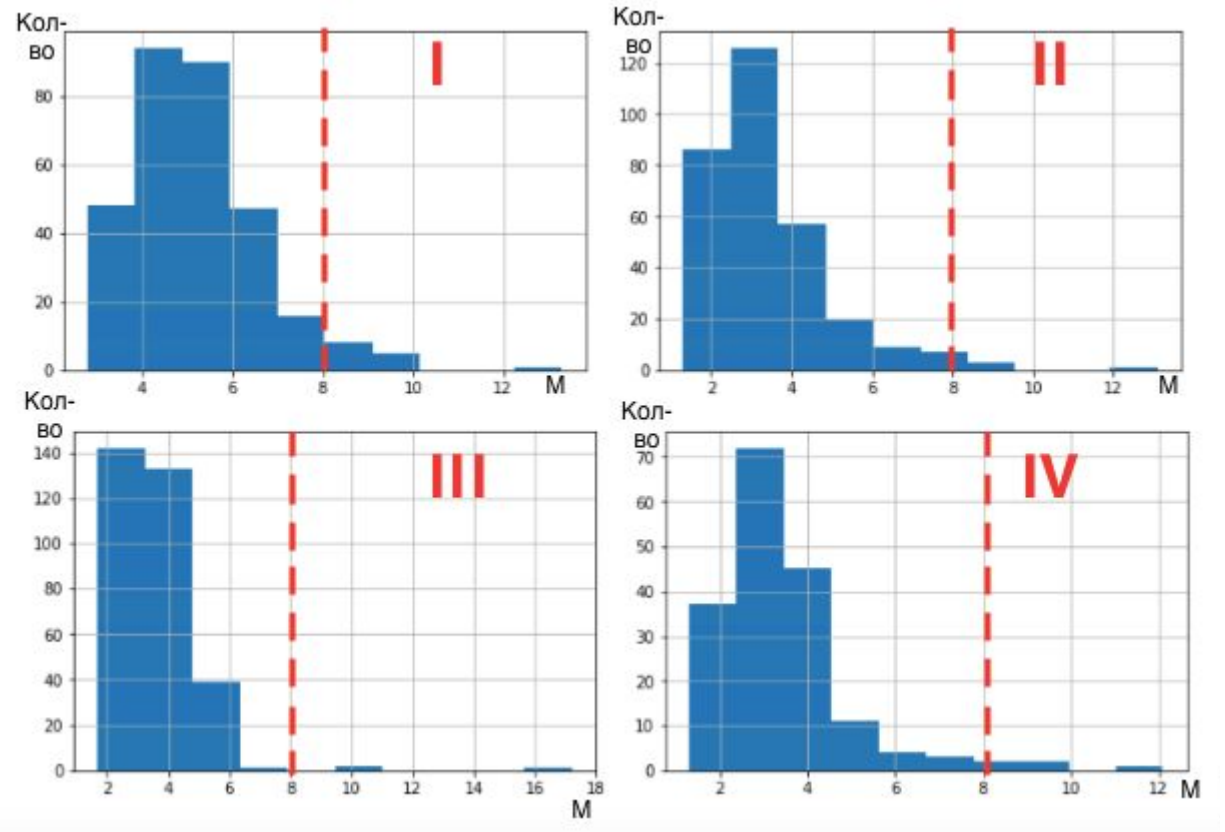


Рис. 2.13 Распределения метрики Махаланобиса на разных ОПС.

Градиентный бустинг - это техника машинного обучения для задач классификации и регрессии, которая строит модель предсказания в форме ансамбля слабых предсказывающих моделей, обычно деревьев решений [25]. Итоговый алгоритм классификации находится в виде композиции:

$$F_M(x) = \sum_{m=1}^{M} b_m h(x; a_m),\ b_m \in R,\ a_m \in A.$$

Однако подбор оптимального набора параметров $\{a_m, b_m\}_{m=1}^{M}$ – очень трудоемкая задача. Поэтому мы будем пытаться построить такую композицию путем жадного наращивания, каждый раз добавляя в сумму слагаемое, являющееся наиболее опти- мальным алгоритмом из возможных. Будем считать, что нами уже построен классификатор $F_{m-1}$ длины m − 1. Таким образом задача сводится к поиску пары наиболее оптимальных параметров $\{a_m, b_m\}$ для классификатора длины m:

$$F_m(x) = F_{m-1}(x) + b_m h(x; a_m),\ b_m \in R,\ a_m \in A.$$

Настроим модель градиентного бустинга и LSTM на признаках со значением метрики Махаланобиса в качестве целевой переменной. Результаты предсказания изображены на рисунке (Рис. 2.14, 2.15).

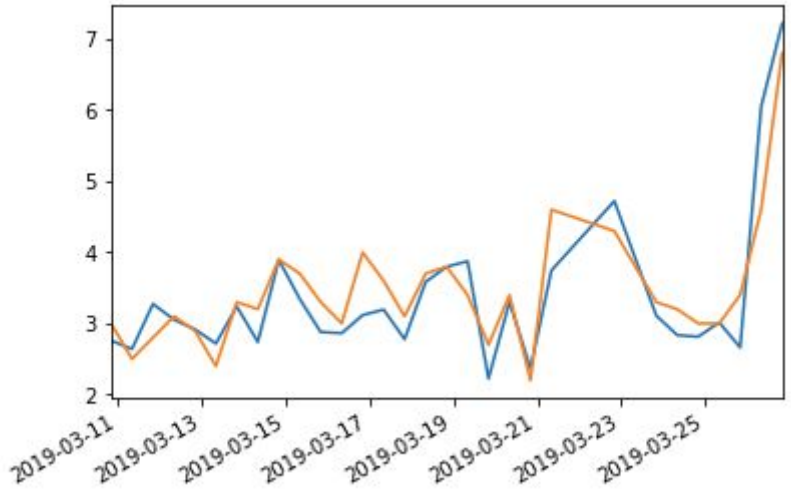


Рис. 2.14 График истинных значений и предсказаний модели градиентного бустинга.

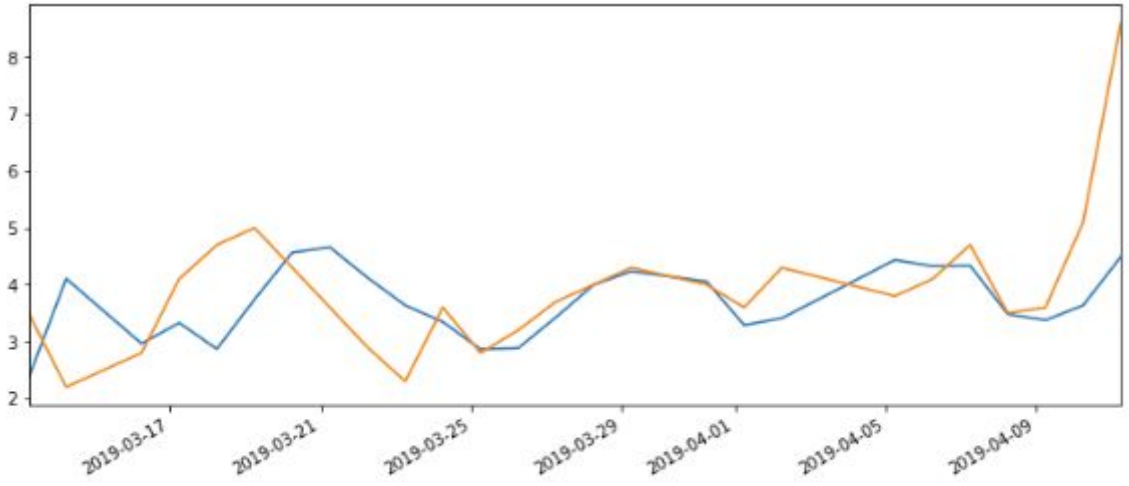


Рис. 2.15 График истинных значений и предсказаний LSTM.

Было сделано предположение, что, при большом отклонении предсказания по признакам и истинного значения метрики, интервал можно также рассматривать как аномальный. Далее, предположение было подтверждено

экспертно. Результирующими аномалиями стали случаи, проявившиеся во всех трех моделях.

## 2.4 Реализация детектирования аномалий на объектах почтовой связи

В итоговой версии рабочего решения были реализованы следующие модули:

- Класс с имплементацией алгоритма подсчета расстояния Махаланобиса между точками, или точкой и распределением множества;
- Класс, содержащий настроенные ранее модели с подобранными параметрами, реализующий считывание данных и обучение, сохранение моделей;
- Класс, в котором реализовано считывание данных для прогноза, считывание сохраненных моделей, совершение предсказаний и блендинг моделей; запись результата в Hive;
- Классы-настройки передачи параметров (названия БД, таблиц, объектов, временных интервалов) в модели;
- sql-скрипт создания подготовительных таблиц для преобразования, агрегирования данных к необходимому для моделей формату;

# ЗАКЛЮЧЕНИЕ

Целью работы являлась реализация автоматического поиска аномалий в объектах почтовой связи. Для достижения цели были выполнены все поставленные задачи. Сперва была проанализирована структура данных. Затем был проведен обзор подходов к решению задач логистики, и изучен набор использующихся методов, алгоритмов и метрик. Далее, после согласования плана работы с заказчиком, была выбрана технология обработки данных с учетом требований к большому объему и скорости поступления данных. После чего была проведена обширная работа по выбору лучшей модели на языке Python. Реализация наиболее подходящей модели была переписана на язык Scala.

Задача была решена путем использования фреймворка Apache Spark, который позволяет быстро обрабатывать большие потоки данных и содержит библиотеки для машинного обучения, предоставляющие различные алгоритмы. В работе было опробовано несколько подходов и библиотек для анализа и поиска аномалий и в качестве итоговой модели были выбраны градиентный бустинг и нейронная сеть, которые в дальнейшем были реализованы на Scala + Spark. Было реализовано несколько модулей с использованием языков Python, SQL, Scala.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ


1. Аномалия, Википедия [Электронный ресурс], – режим доступа: https://ru.wikipedia.org/wiki/Аномалия.
2. В.П. Шкодырев, К.И. Ягафаров, В.А. Баштовенко, Е.Э. Ильина, «Обзор методов обнаружения аномалий в потоках данных», Second Conference on Software Engineering and Information Management, 2017.
3. KNN, официальный сайт с документацией по реализации [Электронный ресурс], – режим доступа: https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html
4. Изолирующий лес, официальный сайт с документацией по реализации [Электронный ресурс], – режим доступа: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.IsolationForest.html
5. Метод опорных векторов, [Электронный ресурс], – режим доступа: http://www.machinelearning.ru/wiki/index.php?title=SVM
6. Д. П. Ветров, Д. А. Кропотов, А. А. Осокин, “Байесовские сети”, [Электронный ресурс], – режим доступа: http://www.machinelearning.ru/wiki/images/5/5b/Lecture1_GM.pdf
7. Е. В. Зубков, В. М. Белов, «Методы интеллектуального анализа данных и обнаружение вторжений», Вестник СибГУТИ No 1, 2016.
8. Цепь маркова, описание подхода, [Электронный ресурс], – режим доступа: https://ru.wikipedia.org/wiki/Цепь_Маркова
9. В.П. Шкодырев, К.И. Ягафаров, В.А. Баштовенко, Е.Э. Ильина, «Обзор методов обнаружения аномалий в потоках данных», Second Conference on Software Engineering and Information Management, 2017.
10. Hadoop, HDFS, официальный сайт [Электронный ресурс], – режим доступа: https://hadoop.apache.org.

11. Scala, официальный сайт [Электронный ресурс], – режим доступа: https://www.scala-lang.org
12. Java, официальный сайт [Электронный ресурс], – режим доступа: https://www.java.com/ru/
13. Apache Spark, официальный сайт [Электронный ресурс], – режим доступа: https://spark.apache.org.
14. Python, официальный сайт [Электронный ресурс], – режим доступа: https://www.python.org
15. TensorFlow, официальный сайт [Электронный ресурс], – режим доступа: http://tensorflow.org/
16. Scipy, официальный сайт [Электронный ресурс], – режим доступа: https://www.scipy.org
17. Keras, официальный сайт [Электронный ресурс], – режим доступа: https://keras.io
18. PyTorch, официальный сайт [Электронный ресурс], – режим доступа: https://pytorch.org
19. upGrad, «Top 9 Python Libraries for Machine Learning in 2020» [Электронный ресурс], – режим доступа: https://www.upgrad.com/blog/top-python-libraries-for-machine-learning/.
20. Библиотека Mlib от Spark, официальный ресурс [Электронный ресурс], – режим доступа: https://spark.apache.org/mllib/
21. Sparkling Water, официальная документация, [Электронный ресурс], – режим доступа: https://www.h2o.ai/products/h2o-sparkling-water/
22. Deep learning for Java, официальная документация, [Электронный ресурс], – режим доступа: https://deeplearning4j.org
23. Метод главных компонент, Википедия [Электронный ресурс], – режим доступа: https://ru.wikipedia.org/wiki/Метод_главных_компонент.

24. Метрика Махалонобиса, Википедия [Электронный ресурс], – режим доступа: https://en.wikipedia.org/wiki/Mahalanobis_distance.
25. Фонарев А.Ю., “Обзор алгоритмов бустинга”, МГУ, 2012