МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН
Некоммерческое акционерное общество
АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ
имени Гумарбека Даукеева

Кафедра «Телекоммуникационные сети и системы»
Специальность: 6М071900 «Радиотехника, электроника и телекоммуникации»

ДОПУЩЕН К ЗАЩИТЕ
Зав. кафедрой
PhD, доцент Темырканова Э.К.
(ученая степень, звание, ФИО)

_____
(подпись)
«_____»_____2020 г.

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ
пояснительная записка

на тему: «Разработка методологии анализа мобильных данных на примере недельной записи городского трафика»

Магистрант: Кабдюшев И.Е _____ группа МРЭТн-18-2
(Ф.И.О.) (подпись)
Руководитель: к.т.н., профессор АУЭС _____ Туманбаева К.Х.
(ученая степень, звание (подпись) (Ф.И.О.)

Рецензент_____ _____ _____
(ученая степень, звание) (подпись) (Ф.И.О.)

Консультант по ВТ: к.т.н., профессор АУЭС _____ Туманбаева К.Х.
(ученая степень, звание) (подпись) (Ф.И.О.)

Нормоконтроль: к.т.н., профессор АУЭС _____ Туманбаева К.Х.
(ученая степень, звание) (подпись) (Ф.И.О.)

Алматы 2020

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН
Некоммерческое акционерное общество
АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ
имени Гумарбека Даукеева

Институт Космической Инженерии и Телекоммуникаций

Специальность: 6М071900 «Радиотехника, электроника и телекоммуникации»

Кафедра: «Телекоммуникационные сети и системы»

ЗАДАНИЕ
на выполнение магистерской диссертации

Магистранту Кабдюшеву Ильясу Ерликулы
(фамилия, имя, отчество)

Тема диссертации «Разработка методологии анализа мобильных данных на примере недельной записи городского трафика»

Утверждена Ученым советом университета №122 от 25 октября 2018

Срок сдачи законченной диссертации «25» мая 2020г.

Цель исследования состоит в статистической обработке сырых данных с использованием современных инструментов Data Science, разработке шагов для интеллектуального анализа и машинного обучения, разработке рекомендаций к анализу данных.

Перечень подлежащих разработке в магистерской диссертации вопросов или краткое содержание магистерской диссертации:

1. Теория и инструменты анализа данных
2. Первичный анализ данных с Pandas
3. Использование Python для аналитики данных CDR
4 Подготовки данных и машинное обучение
5 Методология анализа данных

Перечень графического материала (с точным указанием обязательных чертежей)

Рисунок 2.5 – Сравнение модели и реальных данных
Рисунок 2.6 – Тепловая карта активных соединений
Рисунок 2.10 – Вариативность по дням
Рисунок 2.12 – Результаты теста
Рисунок 2.13 – Диаграмма рассеяния для всех ячеек
Рисунок 2.15 – Функция автокорреляции для остатков и данных

Рекомендуемая основная литература

1 «Технологии анализа данных: Data Mining. Visual Mining. Text Mining, OLAP» А. А. Барсегян. М. С. Куприянов, В. В. Стенаненко, И. И. Холод. — 2-е изд., перераб. и доп.

2 Saari A. Multi-Class Semi-Supervised and Online Boosting: Ph.D. thesis / Graz University of Technology, Faculty of Computer Science. 2010.

Г Р А Ф И К

подготовки магистерской диссертации

Наименование разделов, перечень разрабатываемых вопросов	Сроки представления научному руководителю	Примечание
1. Информационный обзор согласно теме	05.10.2018	
2. Основные понятия и инструменты анализа, изучение данных телекоммуникаций (теоретическая часть)	14.01.2019	
3. Ознакомление и первичный анализ с библиотеками Python: MatPlotlib, SciPy, Pandas (расчетная часть)	02.02.2019	
4. Анализ полученных данных, применение машинного обучения (расчетная часть), разработка рекомендаций	18.10.2019	
5. Сбор аналитики приложений и сравнение с данными телекоммуникаций	10.12.2019	

Дата выдачи задания 30 сентября 2018г.

Заведующий кафедрой _____ (Темырканова Э.К.)
(подпись) (Ф.И.О.)

Научный
руководитель диссертации _____ (Туманбаева К.Х.)
(подпись) (Ф.И.О.)

Задание принял к исполнению
магистрант _____ (Кабдюшев И.Е.)
(подпись) (Ф.И.О.)

Аңдатпа

Бұл жұмыста мобильді трафикқа арналған деректер жинағы зерттеледі. Деректер жиынтығының бірегей құрамы оны оқиғалар мен аномалияларды анықтау, болжау, қаланың өркендеуі және басқа да көптеген мәселелерді шешуге бағытталған әдіснамалар мен тәсілдер үшін тамаша сынау қондырғысына айналдырады. Негізгі құралдар ретінде Python бағдарламалау тілінің статистикалық құралдары мен кітапханалары пайдаланылатын болады.

Аннотация

В настоящей работе исследуется набор данных мобильного трафика. Уникальный состав набора данных делает его идеальным испытательным стендом для методологий и подходов, направленных на решение широкого спектра проблем, такие как обнаружение событий и аномалий, прогнозирование, городское благополучие и много других. В качестве основных инструментов будут использоваться статистические инструменты и библиотеки языка программирования Python.

Abstract

This paper explores a mobile traffic data set. The data set is ideal test environment for developing methods and approaches that solves wide range of problems, such as detecting events and anomalies, forecasting, urban prosperity detection and many others. The main tools for analyzing are python programming language libraries and known statistical tools.

Содержание

Введение

Огромное количество данных доступно в информационном секторе и его число постоянно растет. Эти данные бесполезны, пока они не преобразуются в информацию, благодаря которой бизнес сможет получить прибыль. Но для этого необходимо проанализировать этот огромный объем данных и извлечь из них пользу.

Термин Data Mining часто переводится как интеллектуальный анализ данных, извлечение информации, средства исследования шаблонов, извлечение знаний, анализ шаблонов, «извлечение зерна знаний из гор данных», «очистка данных». Концепция обнаружения знаний в базе данных может считаться синонимом интеллектуального анализа данных [1].

Концепция Data Mining, появившаяся в 1978 году, приобрела большую популярность в современной интерпретации с первой половины 90-х годов. До этого обработка и анализ данных выполнялись в рамках прикладной статистики.

Актуальность Data Mining также подтверждается тем фактом, что поисковый термин «Data Mining» в поисковой системе Google (по состоянию на сентябрь 2005 года) составляет более 17 миллионов страниц.

Области исследования Data Mining обширны и наиболее применяемы инструменты анализа у телекоммуникационных операторов. Пользователи генерируют большое количество информации при обращении в сеть или при совершении звонков, а учитывая рост цифровизации, многие процессы обновления информации и пользования сетью стали автоматическим фоновым процессом в смартфонах его пользователей.

Целью данной работы является статистическая обработка сырых данных с использованием современных инструментов Data Science, разработки шагов для интеллектуального анализа и машинного обучения, разработка рекомендаций к анализу данных. Для достижения этой цели, понадобится выделить основные задачи:

1) изучить современные средства и языки программирования, такие как Python и его библиотеки pandas, numpy, scipy;

2) ознакомиться с исходными данными, использовать статистические инструменты для анализа и визуализации

3) подготовить данные к машинному обучению и применить алгоритм

Данная работа поделена на три части. В первой части будут рассмотрена основная теоретическая информация об интеллектуальном анализе данных и его инструментах. Вторая часть посвящена непосредственно анализу данных с использованием инструментов языка Python. В третьей части рассмотрена аналитика данных приложений и его корреляция с мобильными данными.

1 Теория и инструменты анализа данных

1.1 Data mining как процесс анализа данных

Процесс интеллектуального анализа данных является одним из видов исследований и состоит из нескольких этапов. Этапы включают в себя элементы сравнения, написания, классификации, обобщения, абстракции и повторения.

Процесс извлечения данных неотделим от процесса принятия решений.

Процесс интеллектуального анализа данных создает модель, которая используется в процессе принятия решений. Традиционный процесс интеллектуального анализа данных включает в себя:

- предметный анализ;
- постановка проблемы;
- обработка данных;
- построение моделей;
- обзор и оценка моделей;
- подбор модели;
- применение модели;
- Коррекция и обновление модели.

В процессе можно выделить три основных уровня

Уровень 1. Анализ домена

Исследование - это процесс определения конкретной области, темы, объекта или явления для специфической цели.

Процесс исследования включает наблюдение свойств объектов для выявления и оценки важных регулярных связей между показателями этих свойств с точки зрения исследователя.

Решение проблемы разработки программного обеспечения должно начаться с изучения предметной или доменной области исследования.

Тематическая область состоит из объектов, которые характеризуются свойствами и тесно связаны друг с другом или каким-либо образом друг с другом взаимодействуют.

Тематическая область является частью реального мира, бесконечна и, с точки зрения исследования, содержит как значимые, так и не относящиеся к цели исследования данные.

Исследователь должен быть в состоянии идентифицировать значительную их часть и значение данных зависит от выбора тематической области.

При изучении темы создается модель. При помощи определенных средств знания из разных источников должны быть однозначно формализованы.

Это могут быть текстовые описания предмета или специальные графические обозначения. Существует множество методов описания доменов: например, метод анализа структуры SADT и основанный на нем IDEF0, диаграммы потоков данных Хейна-Сарсона, методика объектно-

ориентированного анализа UML и другие. Модель домена описывает процессы, запущенные в домене, и данные, используемые в этих процессах.

Это первый шаг в процессе интеллектуального анализа данных. Однако успех дальнейшей разработки приложений для интеллектуального анализа данных зависит от того, насколько правильно происходит моделирование домена. В данной работе предметом или доменом являются телекоммуникации.

Уровень 2. Постановка проблемы/задачи

Следующие шаги включаются в постановку задачи:

- определение проблемы/задачи;
- ее формализация.

Задача также содержит описание статического и динамического поведения тестируемых объектов.

Пример задачи. При рекламе нового продукта на рынке необходимо определить, какая группа клиентов компании наиболее заинтересована в этом продукте.

Описание статики подразумевает описание объектов и их свойств.

Пример. Клиент является субъектом. Свойства объекта клиента: семейное положение, доход за предыдущий год, место жительства.

При описании динамики описываются поведение объектов и причины, которые влияют на их поведение.

Пример. Покупатель покупает продукт А. Когда прибывает новый продукт В, покупатель больше не покупает продукт А, только продукт В. Наличие на рынке продукта В изменило поведение клиента. Динамика поведения объекта часто описывается вместе со статикой.

Технология интеллектуального анализа данных не может заменить анализ и ответы на оставшиеся без ответа вопросы. Поэтому выяснение проблемы является необходимым шагом в процессе интеллектуального анализа данных, поскольку мы определяем, какую задачу следует решить на этом этапе. Иногда фазы анализа предметной области и постановки проблемы объединяются в одну фазу.

Этап 3. Подготовка данных

Цель этапа: разработка базы данных для интеллектуального анализа данных.

Подготовка данных является наиболее важным этапом, качество которого определяет возможность получения качественных результатов от всего процесса интеллектуального анализа данных. По некоторым оценкам, до 80% общего времени, потраченного на проект, может быть потрачено на подготовку данных.

Этот этап составляют следующие пункты

1. Определение и анализ требований к данным

На этом этапе происходит моделирование данных, т.е. определение и анализ требований к данным, необходимых для реализации интеллектуального анализа данных. В то же время изучается распределение

пользователей (географическое, организационное, функциональное). Вопросы доступа к данным, необходимым для анализа, необходимость внешних и / или внутренних источников данных; а также системные аналитические свойства (измерение данных, основные типы выходных документов, порядок преобразования информации и т. д.).

2. Сбор данных

Для простоты и эффективности анализа данных необходимы хранилища данных в вашей организации. С инвестиционной точки зрения их использование дешевле, чем использование отдельных баз данных или центров обработки данных. Однако не все компании оснащены хранилищами данных. В этом случае источником исходных данных являются оперативные, справочные и архивные базы данных, т.е. данные из существующих ИТ-систем.

Кроме того, для интеллектуального анализа данных может потребоваться информация из информационных систем менеджеров, внешних источников, бумажных носителей и специальных знаний или результатов исследований.

Следует отметить, что аналитики и программисты не должны быть привязаны к имеющимся показателям во время подготовки данных и должны описывать максимальное количество факторов и признаков, которые влияют на процесс анализа.

На этом этапе часть данных может кодироваться. Предположим, что одним из атрибутов клиента является уровень дохода, который должен быть представлен в системе одним из следующих значений: очень низкий, низкий, средний, высокий, очень высокий. Необходимо указать уровень дохода. В этом процессе требуется сотрудничество аналитика со специалистом в этой области.

Определение количества необходимых данных

При определении необходимого количества данных важно учитывать, были ли данные упорядочены или нет.

Когда данные упорядочены и во временных рядах, желательно знать, содержит ли такой набор данных сезонный / циклический компонент. Если запись содержит сезонный / циклический компонент, у вас должны быть данные по крайней мере для одного сезона / цикла. Если данные не упорядочены, то есть события из записи данных не связаны со временем, при сборе данных следует соблюдать следующие правила.

Количество записей в данных. Недостаточно записей в датасете может привести к созданию неправильной модели. Со статистической точки зрения точность модели возрастает с увеличением объема данных. Некоторые данные могут быть устаревшими или описывать необычную ситуацию и должны быть исключены из базы данных. Алгоритмы, используемые для построения моделей в больших базах данных, должны быть масштабируемыми.

Отношение количества записей в наборе к количеству входных переменных. При использовании многих алгоритмов требуется желаемое

соотношение входных переменных и количества наблюдений. Количество записей данных (примеров) должно быть намного выше, чем количество признаков.

Набор данных должен быть репрезентативным и представлять как можно больше ситуаций. Пропорции представления различных примеров в наборе данных должны соответствовать реальной ситуации.

3. Первичная обработка данных

Возможно анализировать данные высокого и низкого качества. Результат достигается в обоих случаях. Чтобы обеспечить качественный анализ, следует выполнить начальную обработку данных, что является необходимым шагом в процессе интеллектуального анализа данных.

Оценка качества данных. Данные, полученные из коллекции, должны соответствовать по выбранному критерию качества. Это позволяет нам выделить важный подэтап процесса интеллектуального анализа данных - оценку качества данных.

Качественными данными является данные с критерием полноты, точности, своевременности и способности интерпретировать данные.

Данные могут быть высокого или низкого качества, Низкокачественные данные называют «грязными» или «плохими» данными.

Высококачественные данные - это полные, точные и своевременные данные, которые можно интерпретировать.

Данные такого типа дают качественный результат: знания, способные поддержать процесс принятия решений. Важность этой темы подтверждается тем фактом, что «серьезный подход к качеству данных» является одним из десяти основных трендов, прогнозируемых Knightsbridge Solutions в начале 2005 года в области бизнес-анализа и хранилищ данных. Прогноз был подготовлен в январе 2005 года, а в июне 2005 года Даффи Брансон, одна из ведущих компаний Knightsbridge Solutions, провела анализ соответствия этих прогнозов.

Ниже приведен прогноз и его анализ через полгода, а резюме его анализа показано в [2].

Прогноз. Большинство компаниий начали уделять поышенное внимание качеству данных, поскольку низкое качество приводит к снижению производительности, ошибочным бизнес-решениям и невозможности достичь желаемого результата, а также затрудняет выполнение требований законодательства, что увеличивает трату денежных средств. Поэтому меры для решения проблем качества данных действительно начали приниматься в разных огранизациях.

Реальность. Эта тенденция особенно сохраняется в сфере финансовых услуг. Это особенно верно для компаний, стремящихся соблюдать соглашение Базель II. Низкое качество данных не может быть использовано в системах оценки рисков, которые устанавливают цены по кредитам и рассчитывают требования к капиталу компании. Стоит отметить, что взгляды на то, как решить проблему качества данных, существенно изменились. Первоначально

менеджеры сосредоточились на инструментах оценки качества и считали, что «владелец» данных должен решить проблему на уровне источника, например, удалив данные и переподготовив персонал. Но теперь их взгляды значительно поменялись. Концепция качества данных является гораздо более всеобъемлющей, чем просто подробное введение в систему на первом этапе. Многие люди уже понимают, что качество данных должно обеспечиваться посредством процессов извлечения, преобразования и загрузки (извлечение, преобразование, загрузка - ETL), а также путем загрузки данных из источников, которые подготавливают данные.

Отсутствие данных, неточные или бесполезные данные с точки зрения практичности анализа (например, в неправильном формате, который не соответствует стандарту) называют данными низкого качества или грязными данными. Грязные данные создавались одновременно с системами ввода данных.

Грязные данные могут возникать по разным причинам, например, из-за ошибок при вводе, использования других единиц измерения или стандартов, неудачного удаления дублированных наборов данных и прочие. Наличие грязных данных может фактически привести к финансовым потерям и юридической ответственности, если их невозможно предотвратить или если они не признаны и не удалены [3].

Для более детального понимания «грязных» данных можно изучить [4], в котором представлена таксономия 33 типов «грязных» данных и таксономия для методов предотвращения, распознавания и очистки данных. Типы грязных данных:

- которые могут быть автоматически распознаны и удалены;
- данные, наличие которых можно предотвратить;
- данные, которые не подходят для автоматического детектирования и удаления;
- данные, которые нельзя предотвратить.

Поэтому важно понимать, что специальные «очистители» данных могут не обрабатывать все типы грязных данных.

Наиболее популярные типы грязных данных:

- пропущенные значения;
- дубликаты данных;
- шум и выбросы.

Недостающие значения

Некоторые значения данных могут быть опущены, потому что:

- данные вообще не собирались;
- некоторые признаки не подходят к некоторым предметам

Что мы можем сделать с отсутствующими данными?

- исключить строки с пропущенными значениями из анализа.
- рассчитать новые значения для отсутствующих признаков.
- игнорировать пропущенные значения в процессе обработки.
- заменить пропущенные значения прогнозируемыми значениями.

Дубликаты данных.

Запись данных может содержать дубликаты данных, записи с одинаковыми значениями для всех признаков.

Наличие дубликатов в датасете может быть одним из способов повысить ценность некоторых записей. Иногда вам нужно выделить некоторые записи из датасета. Однако чаще всего дублированные данные - результат ошибок при подготовке выборки.

Что мы можем сделать с дубликатами?

Есть два способа обработки дубликатов. Первый вариант удаляет все строки, с дубликатами. Эта опция используется, когда наличие дубликатов приводит к недоверию к информации и полностью обесценивает ее.

Второй вариант - заменить дублирующуюся группу уникальной записью.

Шумы и выбросы.

Выбросы - это резко отличающиеся наблюдения в датасете.

Шумы и выбросы являются довольно распространенной проблемой при обработке данных. Выбросы могут быть отдельными наблюдениями или могут быть сгруппированы вместе. Задача аналитика - не только распознать их, но и оценить их влияние на результаты дальнейшего анализа.

Практика проведения двухэтапного анализа - с выбросами и без них - и сравнения результатов является распространенной практикой [5].

Различные методы анализа данных имеют разную чувствительность к выбросам. Этот факт следует учесть при выборе способа анализа. Некоторые инструменты анализа данных также имеют встроенные процедуры очистки от шума и выбросов.

Визуализация данных обеспечивает графическое представление данных, включая выбросы. Пример наличия выбросов показан на Scatter Plot на рисунке 1.1 Заметны наблюдения, сильно отличающихся от других (на большом расстоянии от большинства наблюдений).

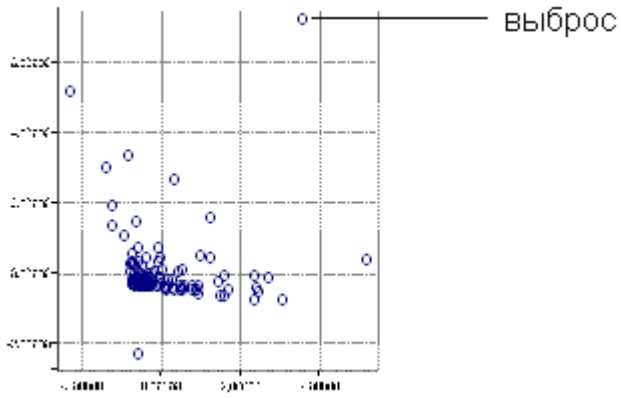

Рисунок 1.1 - Пример набора данных с выбросами

Конечно, результаты интеллектуального анализа данных, основанные на грязных данных, нельзя считать надежными и полезными. Наличие таких данных не обязательно означает, что их следует удалить или предотвратить. Всегда должен быть разумный выбор между грязными данными и стоимостью и / или временем для их удаления.

1.2 Инструменты Data Mining

В этой главе будет предоставлен сравнительный обзор пяти популярных инструментов для анализа данных - Python, R, Weka, Knime и RapidMiner.

Краткие описания

Python - в русскоязычной среде называемый «питон» - по определению является общим языком программирования высокого уровня, целью которого является повышение производительности и удобочитаемости кода. За эти годы Python приобрел много библиотек. В данной работе будут использоваться 6 их них:

- Pandas - отвечает за работу с датасетами;
- Numpy - работа с массивами;
- StatsModels - содержит основные статистические функции и модели;
- Sklearn и Pybrain - алгоритмы машинного обучения;
- Matplotlib отвечает за рендеринг.

В дополнение к библиотекам, Python является гибким и имеет четкий синтаксис - с последним приятно работать.

Язык R, опубликованный в 1993 году, сегодня является стандартом анализа данных. R «заточен» для статистической обработки данных, работы с графикой и алгоритмами машинного обучения. С пакетом ggplot2 становится возможна эффективная визуализация и рендеринг.

Инструмент Weka - это набор инструментов и алгоритмов для анализа и прогнозирования. Преимущества включают в себя:

- удобный UI;
- преобразование данных;
- поддержка многих алгоритмов машинного обучения и возможность их быстрого использования;
- удобный вывод результатов алгоритма;
- подбор признаков;
- визуализация;
- возможность проводить эксперименты (и вы можете запустить несколько алгоритмов для разных задач одновременно и получить общий отчет);
- способность изобразить весь процесс решения проблем в виде диаграммы.

Инструменты *Knime и RapidMiner* схожи по форме и контенту (хотя первый существует совершенно бесплатно в отличие от второго) - поэтому они рассматриваются вместе. Оба инструмента поддерживают множество общих задач - преобразование данных, статистику, машинное обучение и

визуализацию [11]. Весь процесс анализа данных представлен в виде интерактивной диаграммы - последовательности операторов, в то время как операторы Weka и R доступны пользователю.

Сравнительные свойства

Ниже приведены 6 таблиц, которые показывают результаты оценки инструментов. Основываясь на результатах, общая оценка каждого инструмента была рассчитана по отношению к конкретной характеристике, а результаты сведены в таблицу.

Таблица 1.2.1 - Обработка данных

Обработка данных	Python	R	Weka	Knime / RapidMiner
Практические навыки	****	**	*	*
Возможно ли делать сложные преобразования	***	***	*	*
Простые преобразование (нормализация, и т.д.)	****	****	****	****
Сумма	11	9	6	6

Таблица 1.2.2 - Визуализация

Визулизация	Python	R	Weka	Knime / RapidMiner
Гибкость	****	****	**	**
Эстетика	**	****	**	**
Сумма	6	8	4	4

Таблица 1.2.3 - Машинное обучение

Машинное обучение	Python	R	Weka	Knime / RapidMiner
Количество методов	****	****	****	***
Настройка параметров	****	****	**	**
Сумма	8	8	6	5

Таблица 1.2.4 - Представление результатов работы

Представление результатов работы	Python	R	Weka	Knime / RapidMiner
Гибкость	****	****	*	**
2 * Трата времени на настройку хорошего вывода	*	**	****	***
Сумма	6	8	9	8

Таблица 1.2.5 - Скорость получения предварительных результатов

Скорость получения предварительных результатов	Python	R	Weka	Knime / RapidMiner
Затрата времени на написание кода	*	*	****	****
Быстрый вывод результатов	**	***	****	****
Сумма	3	4	8	8

Таблица 1.2.6 - Наглядность процесса анализа данных

Наглядность процесса анализа данных	Python	R	Weka	Knime / RapidMiner
Наглядность	*	*	**	****
Сумма	1	1	2	4

В заключительной таблице – итоги анализа. В каждой из шести «номинаций» были выбраны те программные продукты, которые эффективнее всего решают конкретные задачи.

Таблица 1.2.7 - Итоги

Обработка данных	Python
Визуализация	R, Python
Машинное обучение	все, но Python и R предоставляют больше свободы
Представление результатов работы	Weka
Быстрое получение предварительных результатов	Weka, Knime, RM
Реализация собственных алгоритмов	Python, R
Наглядность процесса анализа данных	Knime, RM

По результатам анализа очевидно, что Python (1), R (2), Weka (3) «объективно» лучше, чем Knime, Rapid Miner (4, 5):

- (1), (2) обладают большей гибкостью;
- с помощью (3) можно проанализировать данные быстрее;
- (3) представляет результаты работы более подробно и с большим удобством;
- в (4, 5, 3), свобода действия ограничена;
- в дополнение к низкому пределу входа и красивому изображению, представление процесса анализа данных в виде графика в (4, 5) не имеет объективных преимуществ.

Сравнивая два последних инструмента, мы можем заключить, что Knime (4) лучше, чем Rapid Miner (5):

- (4), в отличие от (5), предоставляется совершенно бесплатно;
- параметры (4) и (5) выглядят одинаково.

Однако ситуация с работой с базой данных и большими наборами данных остается неопределенной: в этих случаях можно использовать Knime и Rapid Miner. Но в этой работе мы будем использовать инструмент Python.

2 Анализ и предобработка с Pandas

2.1 Первичный анализ данных с Pandas

Pandas - это библиотека Python, которая предоставляет широкие возможности анализа данных. Данные или датасеты, с которыми мы работаем, обычно хранятся в виде файлов - например, в форматах .csv, .tsv или .xlsx. Используя библиотеку Pandas, эти табличные данные очень удобны для загрузки, обработки и анализа. В сочетании с библиотеками Matplotlib и Seaborn, Pandas предлагает широкие возможности для визуального анализа табличных данных.

Series и DataFrame являются основными структурами данных в Pandas. Первая - это одномерная индексированная матрица данных определенного типа. Вторая - это двумерная структура данных, которая представляет собой таблицу, и каждый столбец содержит данные одного типа. Датафрейм можно рассматривать как словарь объектов Series. Структура DataFrame идеально подходит для представления реальных данных: строки соответствуют описаниям категорий для отдельных объектов, а столбцы соответствуют самим категориям (признакам).

Далее покажутся основные методы, проанализировав набор данных об уходе клиентов от оператора связи.

В начале необходимо прочитать данные методом read_csv и просмотреть первые 5 строк можно, используя метод head:

```
df = pd.read_csv('../../data/telecommuniations_churn.csv')
df.head()
```

Описание признаков

Посмотреть на размер данных можно используя метод shape.

```
print(df.shape)
(3333, 20)
```

Видно, что размер таблицы 3333 строки и 20 столбцов.

Вывести названия столбцов используя метод column:

```
print(df.columns)
Index(['StateCode', 'Acc.length', 'Zone code', 'International
package',
       'Voicemail package', 'Vmail messages count', 'Day mins.
sum',
       'Day calls sum', 'Day charge sum', 'Eve mins. sum',
       'Eve calls sum', 'Eve charge sum', 'Night mins. sum',
       'Night calls sum', 'Night charge sum', 'Intl mins. sum',
       'Intl calls sum', 'Intl charge sum', 'Customer service
calls',
       'Churn'],
      dtype='object')
```

Для просмотра общей информации по датафрейму и его признакам, необходимо использовать функцию info:

```
print(df.info())
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3333 entries, 0 to 3332
```

```
Data columns (sum 20 columns):
StateCode                       3333 non-null object
Acc.length                      3333 non-null int64
Zone code                       3333 non-null int64
International package           3333 non-null object
Voicemail package               3333 non-null object
Vmail messages count            3333 non-null int64
Day mins. sum                   3333 non-null float64
Day calls sum                   3333 non-null int64
Day charge sum                  3333 non-null float64
Eve mins. sum                   3333 non-null float64
Eve calls sum                   3333 non-null int64
Eve charge sum                  3333 non-null float64
Night mins. sum                 3333 non-null float64
Night calls sum                 3333 non-null int64
Night charge sum                3333 non-null float64
Intl mins. sum                  3333 non-null float64
Intl calls sum                  3333 non-null int64
Intl charge sum                 3333 non-null float64
Customer service calls          3333 non-null int64
Churn                           3333 non-null bool
dtypes: bool(1), float64(8), int64(8), object(3)
memory usage: 498.1+ KB
None
```

bool, int64, float64 и object являются типами атрибутов. Видно, что 1 атрибут является логическим (bool), 3 атрибута имеют тип объекта и 16 атрибутов являются числовыми. Также удобно быстро исследовать пробелы в данных, используя метод info. В нашем случае их нет, в каждом столбце 3333 значения.

Вы можете изменить тип столбца, используя метод astype. Далее применяется этот метод к атрибуту churn и переводится в int64:

```
df['Churn'] = df['Churn'].astype('int64')
```

Метод описания показывает наиболее важные статистические характеристики данных для каждой числовой характеристики (типы int64 и float64): количество пропущенных значений, среднее значение, стандартное отклонение, диапазон, медиана, квартиль 0,25 и 0,75.

```
df.describe()
```

Для качественных (тип объекта) и логических (тип bool) символов разумно использовать метод value_counts. Далее будет просмотрено распределение данных по целевой переменной - Churn:

```
df['Churn'].value_counts()
0    2850
1     483
Name: Churn, dtype: int64
```

2850 пользователей из 3333 — лояльные, значение Churn у них нулевое.

Для примера можно также посмотреть на распределение пользователей по переменной кода города. Для этого необходимо указать значение параметра normalize = True, чтобы увидеть не абсолютные, а относительные частоты.

```
df['Zone code'].value_counts(normalize=True)
415    0.496550
```

```
510    0.252025
408    0.251425
Name: Zone code, dtype: float64
```

Сортировка

Сортировку датафрейма можно выполнить на основе значения любого из признаков. В нашем случае, например, для всей совокупности дней (возрастание = False для классификации в порядке убывания):

```
df.sort_values(by='Day charge sum',
        ascending=False).head()
```

Можно сортировать используя группу столбцов:

```
df.sort_values(by=['Churn', 'Day charge sum'],
        ascending=[True, False]).head()
```

Индексация и извлечение данных

Датафреймы могут быть проиндексированы различными способами. В связи с этим мы рассматриваем различные способы индексации и извлечения необходимых нам данных из блока данных на примере простых вопросов.

Чтобы получить один столбец, необходимо использовать конструкцию формы DataFrame[“Name”]. Здесь будет использоваться данный метод, для того, чтобы ответить на вопрос: какова доля людей нелояльных пользователей в нашей базе данных?

```
df['Churn'].mean().
 # выведется: 0.14491449144914492
```

14,5% - довольно плохой показатель для бизнеса. С таким высоким процентом оттока есть вероятность обанкротиться.

Логическая индексация DataFrame в одном столбце очень удобна. Это выглядит так: `df[P(df['Name'])]`, где P - это определенное логическое условие, которое проверяется для каждого элемента в столбце Name. Результатом этой индексации является датафрейм, который состоит только из строк, которые соответствуют условию P в столбце Name.

Далее это будет использоваться, чтобы ответить на вопрос: каковы средние значения числовых символов среди нелояльных пользователей?

```
df[df['Churn'] == 1].mean()
Acc.length              102.664596
Day mins. sum           206.914079
Day calls sum             101.335404
Day charge sum             35.175921
Eve mins. sum           212.410145
Eve calls sum             100.561077
Eve charge sum             18.054969
Night mins. sum         205.231677
Night calls sum           100.399586
Night charge sum            9.235528
Customer service calls      2.229814
Churn                       1.000000
dtype: float64
```

Объединив два предыдущих типа индексации, можно ответить на вопрос: сколько в среднем нелояльных пользователей разговаривают по телефону в течение дня?

```
df[df['Churn'] == 1]['Day mins. sum'].mean()
# выводится: 206.91407867494823
```

Какова максимальная продолжительность международных звонков между постоянными пользователями (оборот == 0), которые не пользуются услугой международного роуминга («Международный план» == «Нет»)?

```
df[(df['Churn'] == 0) & (df['International package'] == 'No')]['Intl mins. sum'].max()
# выводитс: 18.899999999999999
```

Таблицы данных могут быть проиндексированы по имени столбца или строки или по серийному номеру. Для индексации по имени используется метод loc, по номеру - iloc.

В первом случае говорится «введите значения для строк от 0 до 5 и для столбцов Код города и код области», а во втором - «введите значения для первых пяти строк в первых трех столбцах».

```
df.iloc[0:5, 0:3]
```

	State Code	Acc.length	Zone code
0	S	128	415
1	H	107	415
2	J	137	415
3	OH	84	408
4	K	75	415

Длч применение функции к каждому столбцу воспользуемся apply

```
df.apply(np.max)
```

```
StateCode                    WY
Acc.length                  243
Zone code                   510
International package       Yes
Voicemail package           Yes
Day mins. sum             350.8
Day calls sum               165
Day charge sum            59.64
Eve mins. sum             363.7
Eve calls sum               170
Eve charge sum            30.91
Night mins. sum             395
Night calls sum             175
Night charge sum          17.77
Customer service calls        9
Churn                      True
dtype: object
```

Для применения функции apply к каждой строке. нужно ввести axis = 1, а применение функции к каждой ячейке в столбце: map

Метод map можно использовать для замены значения в столбце, передав ему аргумент в формате {old_value: new_value} в качестве словаря:

```
d = {'No' : False, 'Yes' : True}
df['International package'] = df['International package'].map(d)
df.head()
```

Аналогичное действие можно провернуть при помощи функции replace:

```
df = df.replace({'Voicemail package': d})
df.head()
```

Группировка данных

Группировка данных в Pandas выглядит следующим образом:

```
df.groupby(by=grouping_columns)[columns_to_show].function()
```

Метод groupby применяется к фрейму данных, который разделяет данные с помощью grouping_columns - элемента или набора элементов.

Можно выбрать нужные нам столбцы (column_to_show).

Одна или несколько функций применяются к полученным группам.

Далее будут сгруппированы данные по значению атрибута Churn и просмотрена статистика по трем столбцам в каждой группе.

```
columns_to_show = ['Day mins. sum', 'Eve mins. sum', 'Night mins. sum']
df.groupby(['Churn'])[columns_to_show].describe(percentiles=[])
```

Сделать то же самое можно передав в agg список функций:

```
columns_to_show = ['Day mins. sum', 'Eve mins. sum', 'Night mins. sum']
df.groupby(['Churn'])[columns_to_show].agg([np.mean,       np.std, np.min, np.max])
```

Сводные таблицы

Предположим, необходимо увидеть, как наблюдения в нашей выборке распределяются в контексте двух категорий - оттока и международного плана. Для этого можно построить таблицу сопряженности, используя метод кросс-таблицы:

```
pd.crosstab(df['Churn'], df['International package'])
International package   No     Yes

Churn

0                       2664   186

1                       346    137

pd.crosstab(df['Churn'], df['Voicemail package'], normalize=True)
Voicemail package   No         Yes

Churn

0                   0.602460   0.252625
```

```
Voicemail package   No         Yes

Churn

1                   0.120912   0.024002
```

Как оказывается, большинство пользователей лояльны и в то же время пользуются дополнительными услугами (международный роуминг / голосовая почта).

Преобразование датафреймов

Как и многие другие операции в Pandas, добавление столбцов в DataFrame может быть сделано разоичными путями.

Например, если необходимо рассчитать общее количество звонков для всех пользователей, нужно создать объект sum_calls типа Series и вставить его в блок данных:

```
sum_calls = df['Day calls sum'] + df['Eve calls sum'] + \
                    df['Night calls sum'] + df['Intl calls sum']
df.insert(loc=len(df.columns),         column='Sum         calls',
value=sum_calls)
# loc - номер столбца, после которого нужно вставить данный Series
# мы указали len(df.columns), чтобы вставить его в самом конце
df.head()
```

Добавить столбец из имеющихся столбцов можно, не создавая промежуточных массивов:

```
df['Sum charge'] = df['Day charge sum'] + df['Eve charge sum'] + df['Night charge sum'] +
df['Intl charge sum']
df.head()
```

Использовать метод drop можно, чтобы удалить столбцы или строки. Нужно передать в аргументе нужные индексы и требуемое значение параметра оси (1, если необходимо удалить столбцы, и ничего или 0, если необходимо удалить строки):

```
# удаляем созданные только что столбцы
df = df.drop(['Sum charge', 'Sum calls'], axis=1)
df.drop([1, 2]).head() # а вот так можно удалить строчки
```

2.2 Использование Python для аналитики данных CDR

Каждый раз, когда пользователь участвует в телекоммуникационном взаимодействии, оператор назначает базовую радиостанцию (RBS) и обеспечивает связь через сеть. Затем создается новый CDR, записывающий время взаимодействия и RBS, который его обработал. Из RBS можно получить указание географического местоположения пользователя благодаря картам покрытия C_{map}, которые связывают каждый RBS с частью территории, которую он обслуживает (рисунок 2.1).

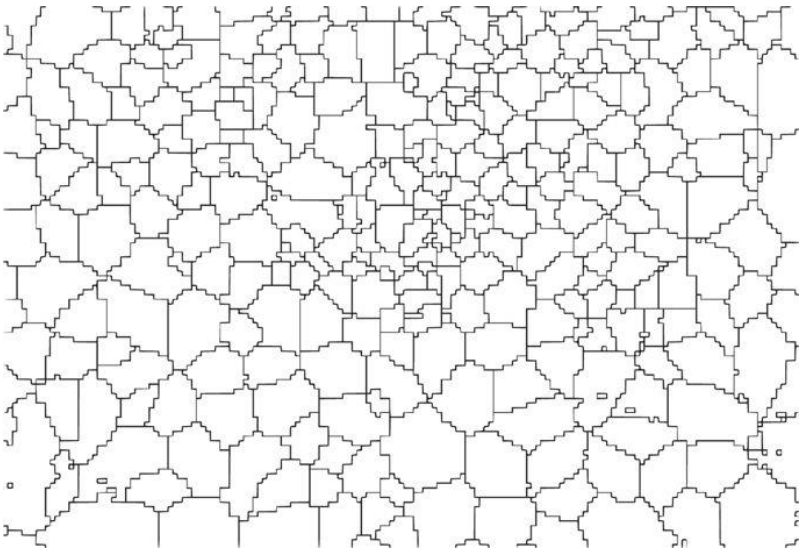

Рисунок 2.1 – Пример зоны покрытия RBS

Чтобы пространственно агрегировать CDR внутри сети, каждое взаимодействие связано с областью покрытия RBS, которая его обработала. Следовательно, количество записей $S_i(t)$ в квадрате сетки i в момент времени t вычисляется следующим образом:

$$S_i(t) = \sum_{v \in C_{map}} R_v(t) \cdot \frac{A_{v \cap \mathrm{i}}}{A_v}$$

где $R_{vj}(t)$ - количество записей в зоне покрытия v в момент времени t, A_v - поверхность зоны покрытия v, а $A_{v \cap i}$ - поверхность пространственного пересечения между v и квадратом i.

Зарегистрированы следующие виды деятельности:

Полученное SMS - CDR генерируется каждый раз, когда пользователь получает SMS

Отправленные SMS - CDR генерируется каждый раз, когда пользователь отправляет SMS

Входящий вызов - CDR генерируется каждый раз, когда пользователь получает вызов

Исходящий вызов - CDR генерируется каждый раз, когда пользователь выполняет вызов

Интернет - CDR генерируется каждый раз, когда пользователь запускает подключение к Интернету или завершает подключение к Интернету. Во время того же соединения создается CDR, если соединение длится более 15 минут или пользователь перевел более 5 МБ.

Общие наборы данных были созданы, объединяя всю эту анонимную информацию с временной агрегацией временных интервалов в десять минут. Количество записей в наборах данных S'i (t) следует правилу:

$S'_i(t) = S_i(t)*k$

где k - это константа, которая скрывает истинное количество вызовов, SMS и соединений.

Были также созданы два типа наборов данных CDR для измерения интенсивности взаимодействия между различными местоположениями: один из определенной области города в любую из соседних и один количественно определяющий взаимодействия в пределах города

Идентификатор площади: идентификационная строка данного квадрата города;

Интервал времени: начальный интервал времени, выраженный в миллисекундах. Время окончания интервала может быть получено путем добавления 600 000 миллисекунд (10 мин) к этому значению;

Активность входящих SMS: активность, пропорциональная количеству полученных SMS-сообщений внутри данного идентификатора площади и в течение определенного интервала времени. SMS-сообщения отправляются из страны, указанной кодом страны;

Активность отправленных SMS: активность, пропорциональная количеству отправленных SMS-сообщений внутри данного идентификатора площади в течение заданного интервала времени. SMS-сообщения принимаются в стране, указанной кодом страны;

Активность входящих вызовов: активность, пропорциональная количеству входящих вызовов внутри идентификатора площади в течение заданного интервала времени. Звонки осуществляются из страны, указанной кодом страны;

Активность исходящих вызовов: активность, пропорциональная количеству исходящих вызовов внутри идентификатора площади в течение заданного интервала времени. Звонки принимаются в стране, указанной кодом страны;

Активность интернет-трафика: количество CDR, созданных внутри заданного идентификатора площади в течение заданного интервала времени. Интернет-трафик инициируется из страны, обозначенной кодом страны;

Код страны: телефонный код страны.

Поскольку наборы данных поступают от разных компаний, которые приняли разные стандарты, их неравномерность в пространственном распределении объединяется в сетку с квадратными ячейками. Это позволяет сравнивать различные области и упрощает географическое управление данными. Таким образом, область города поделена наложением сетки из квадратов (квадраты размером примерно 235×235 метров

Для работы с полученными данными, подходящий инструмент – библиотека Pandas. Данный пакет делает Python мощным инструментом для анализа данных. Пакет дает возможность строить сводные таблицы, выполнять группировки, предоставляет удобный доступ к табличным данным, а при наличии пакета matplotlib дает возможность рисовать графики на полученных наборах данных. При считывании наших csv файлов, в оболочке Jupyter Notebook отображает удобные для визуального восприятия таблицы (Рисунок 2.2).

```
df_cdrs=df_cdrs.fillna(0)
df_cdrs['sms'] = df_cdrs['smsin'] + df_cdrs['smsout']
df_cdrs['calls'] = df_cdrs['callin'] + df_cdrs['callout']
df_cdrs['hour'] = df_cdrs.datetime.dt.hour
df_cdrs.head()
```

Out[56]:

	datetime	CellID	countrycode	smsin	smsout	callin	callout	internet	sms	calls	hour
0	2013-11-01	1	0	0.3521	0.0000	0.0000	0.0273	0.0000	0.3521	0.0273	0
1	2013-11-01	1	33	0.0000	0.0000	0.0000	0.0000	0.0261	0.0000	0.0000	0
2	2013-11-01	1	39	1.7322	1.1047	0.5919	0.4020	57.7729	2.8369	0.9939	0
3	2013-11-01	2	0	0.3581	0.0000	0.0000	0.0273	0.0000	0.3581	0.0273	0
4	2013-11-01	2	33	0.0000	0.0000	0.0000	0.0000	0.0274	0.0000	0.0000	0

Рисунок 2.2 – Отображение таблиц Pandas в Jupyter Notebook

Для визуализации понадобятся библиотеки matplotlib и seaborn – с помощью них, мы сможем рисовать гистограммы, графики и прочие полезные визуальные средства, а также geojson и descartes (Рисунок 2.3)

```
In [64]: df_cdrs['hour'].replace(0,24).replace(1,25).replace(2,26).hist(bins=24)
Out[64]: <matplotlib.axes._subplots.AxesSubplot at 0x16a8a02ee48>
```

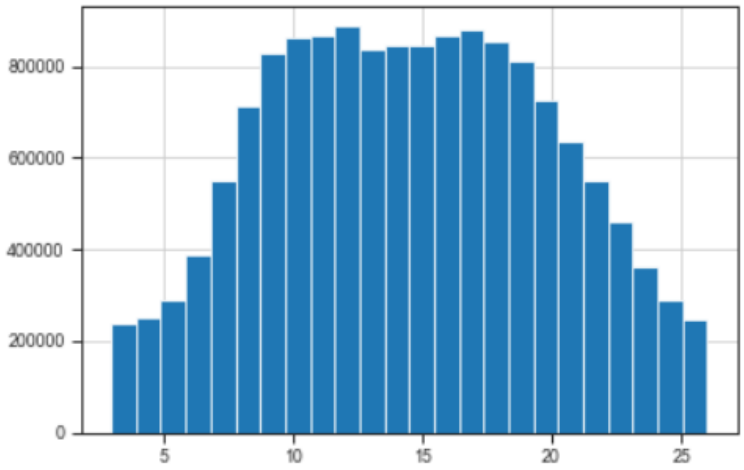

Рисунок 2.3 – Пример визуализации с matplotlib

Пакеты numpy, scipy и scikit-learn необходимы для работы с массивами данных. Numpy – инструмент для оперирования с массивами, scipy позволяет использовать статистические инструменты для анализа наших массивов, а scikit-learn обладает встроенными алгоритмами машинного обучения, для решения различных задач классификации, регрессии и кластеризации.

Обработка данных

В начале объединим несколько записей в один большой датасет. Для удобства исследования просуммируем звонки, смс и сделаем отдельную колонку «час» взаимодействия (Рисунок 2.2), построив гистограмму по часам

взаимодействий (Рисунок 2.3) видно, что большую часть времени люди общаются в период с 10 часов до приблизительно 18 часов вечера, что звучит достаточно правдоподобно.

Выполнив проверку интернет соединений по разным ячейкам (Рисунок 2.4), видим одинаковую природу роста и падения интернет соединений, что подтверждает одинаковое поведение людей в разных частях города. Листинг кода приведен в Приложении А (Рисунок А.1)

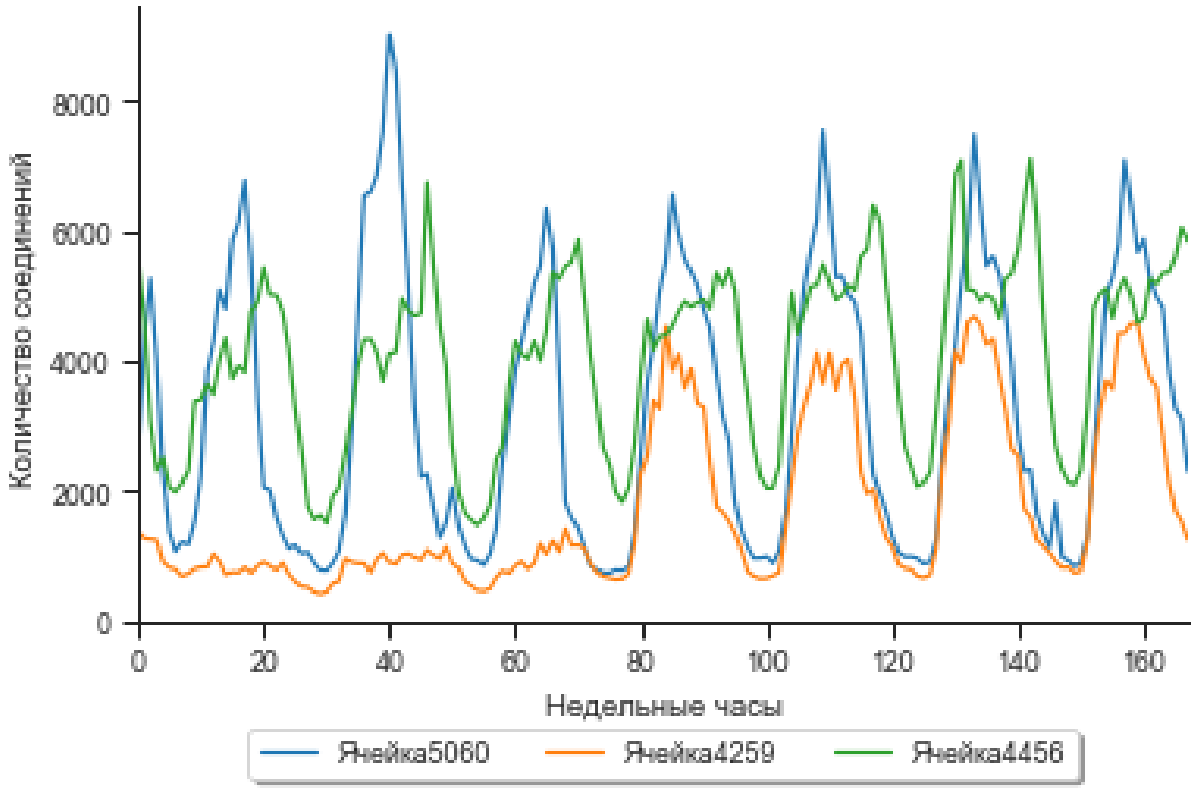

Рисунок 2.4 – Интернет соединения по областям

Взяв к примеру, ячейку5060 и сравнив ее с моделью синусоиды, видим большое совпадение построенной модели синусоиды с реальными данных, что подтверждает синусоидальную «природу» мобильной активности (Рисунок 2.5). Листинг кода приведен в Приложении Б (Рисунок Б.1)

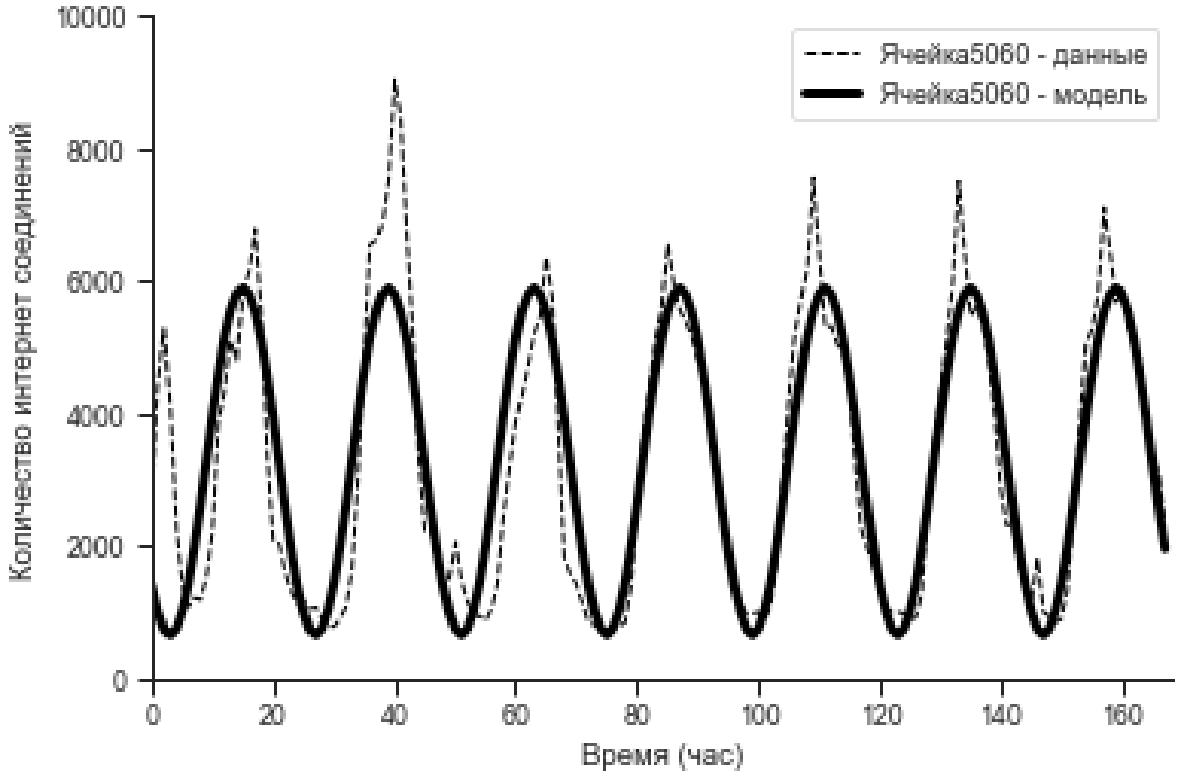

Рисунок 2.5 – Сравнение модели и реальных данных

Если строить тепловую карту по ячейкам, окажется, что большая часть соединений приходится на центр города, а меньшая – на окраины (Рисунок 2.6). Листинг кода приведен в Приложении А (Рисунок А.3)

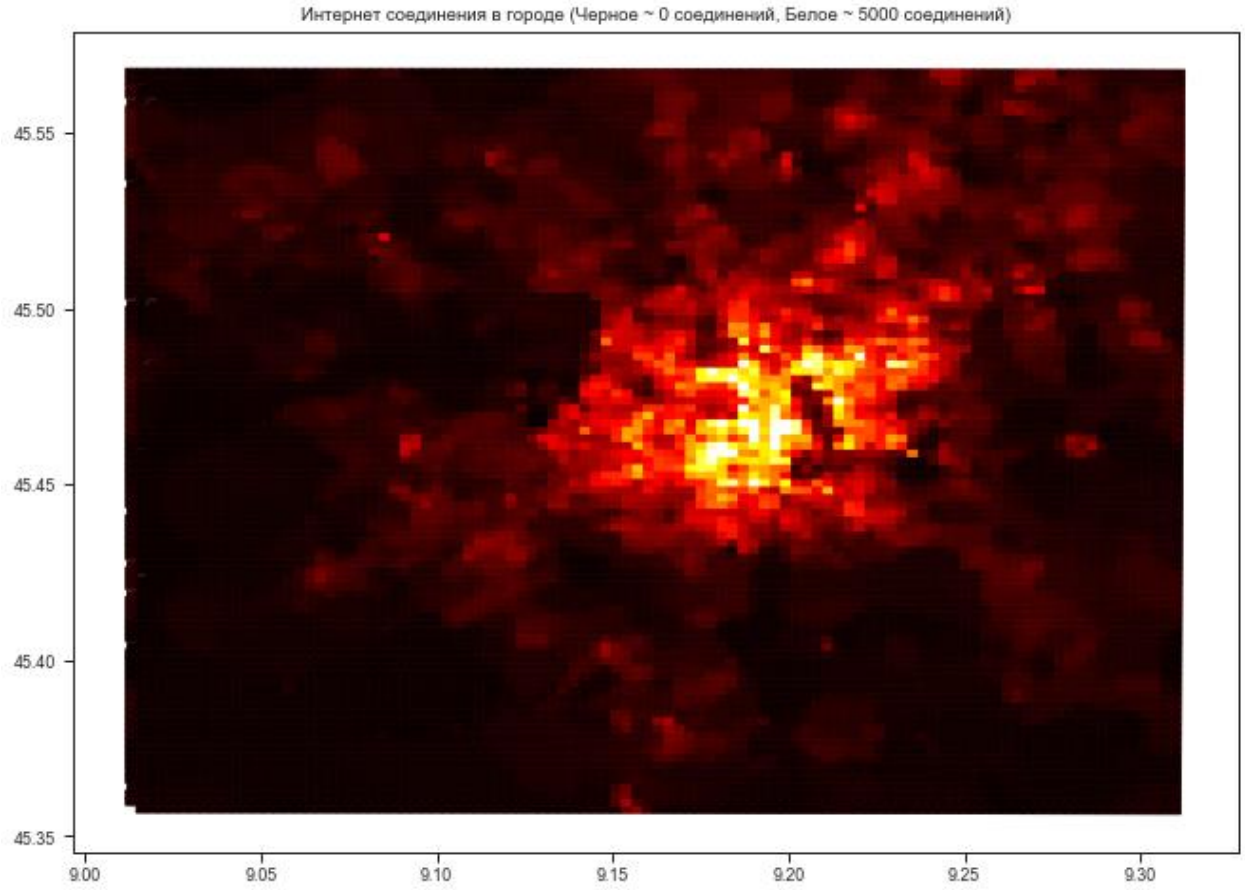

Рисунок 2.6 – Тепловая карта активных соединений

Вариативность в максимальном/минимальном использования данных покажет скачкообразную природу использования данных в этом регионе (Рисунок 2.7). Светлые тона могут говорить о частом изменении количества людей в этих областях.

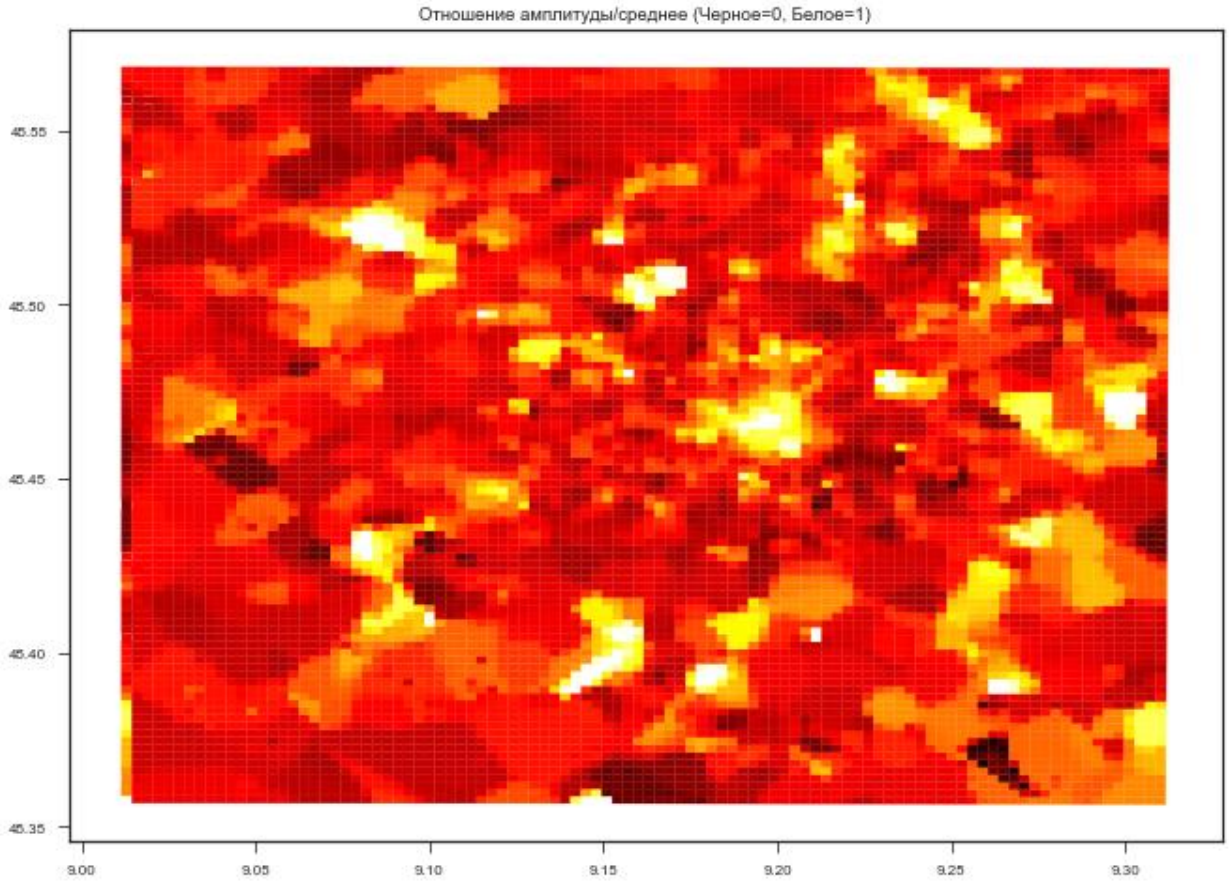

Рисунок 2.7 – Вариативность использования данных

Посмотрев пиковую активность по часам, можно определить преобладание ночных жизни в участках города (где белые участки показывают активность в 10 вечера, а черные – в полдень, Рисунок 2.8). Листинг кода приведен в Приложении Б (Рисунок Б.2)

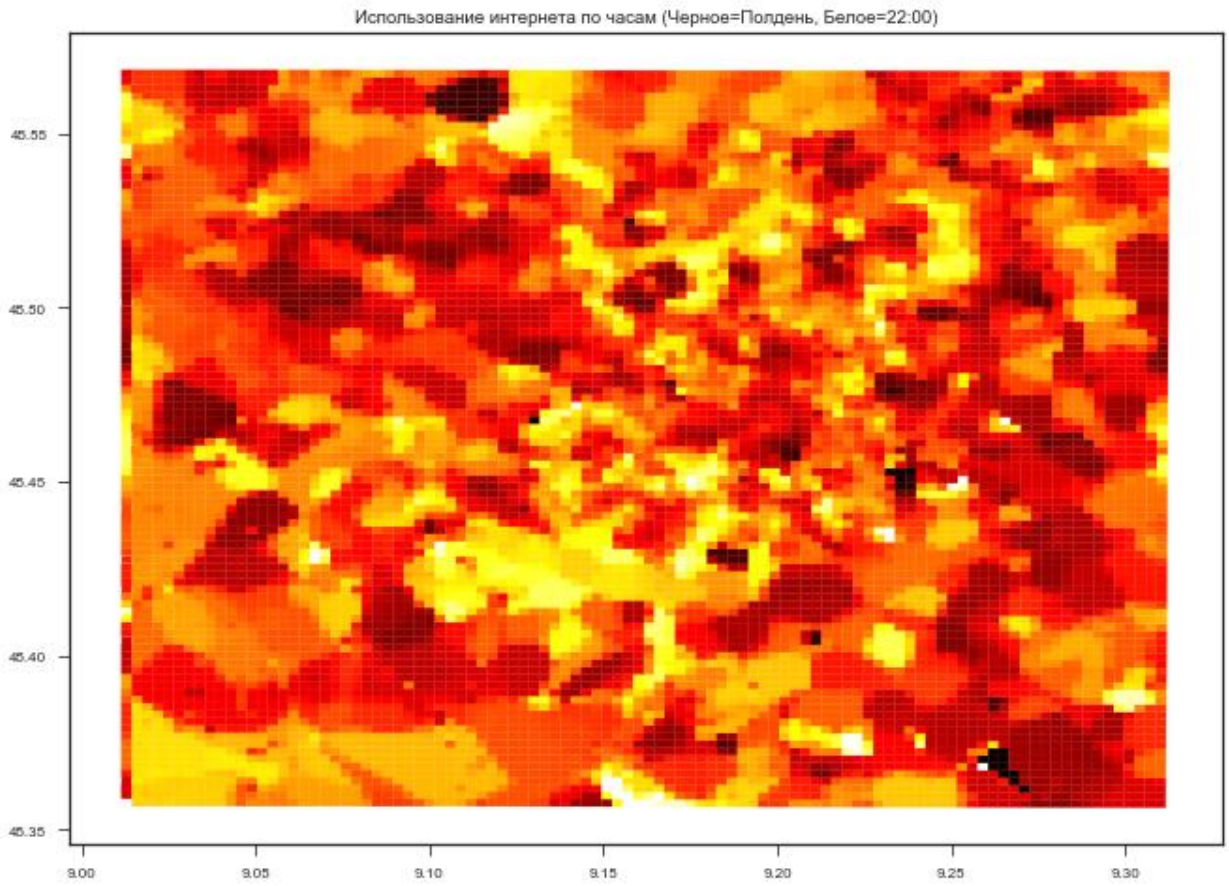

Рисунок 2.8 – Пиковая активность по часам в регионах

Сравнение пиковых значения по ячейкам может подтвердить географическую удаленность регионов (центр города, окраины). На рисунке 2.9 видна разница в 2 часа между двумя ячейками. Листинг кода приведен в Приложении Б (Рисунок Б.3)

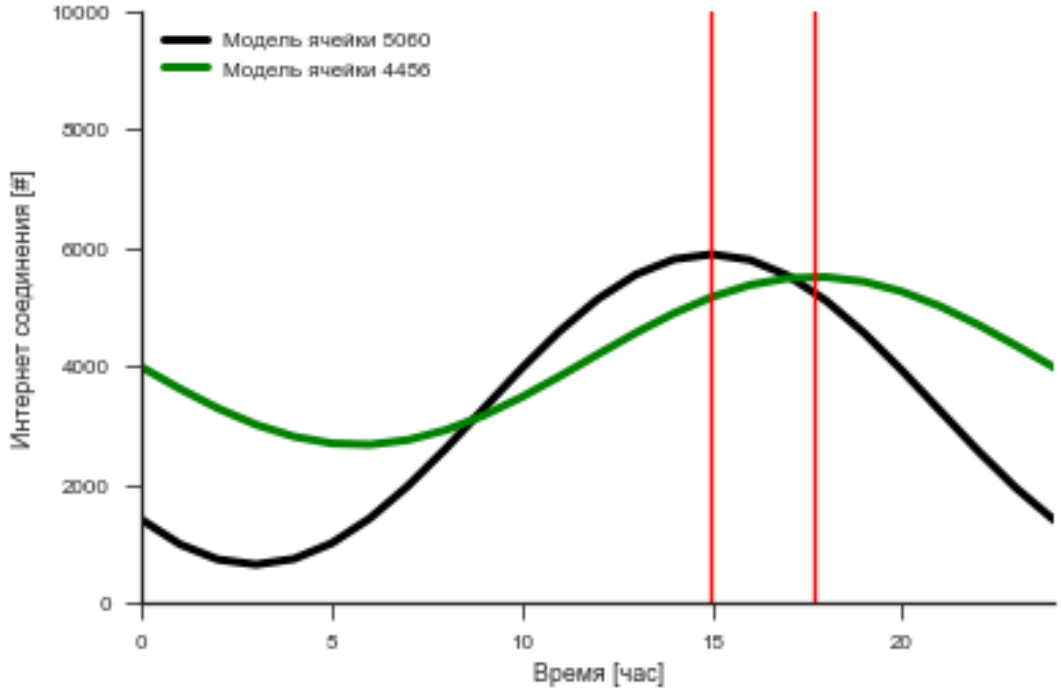

Рисунок 2.9 – Пиковая активность по часам в регионах

Box-plot анализ

Box-plot даёт визуальную сводку вариативности в датасете. Здесь показывается медиана, верхний и нижний квартили, минимальное и максимальное значения, а также все выбросы, которые могут вскрыть необычные случаи в данных. Диаграмма создается с помощью числового поля чисел или поля доля/отношение на оси У. Box-plot используется для ответа на следующего типа вопросы: Какое распределение у данных? Есть ли выбросы в датасете? Каковы вариации в распространении в нескольких сериях в наборе данных?

Построим Box-plot для нашей выбранной ячейки 5060 (Рисунок 2.10). Листинг кода приведен в Приложении А (Рисунок А.2)

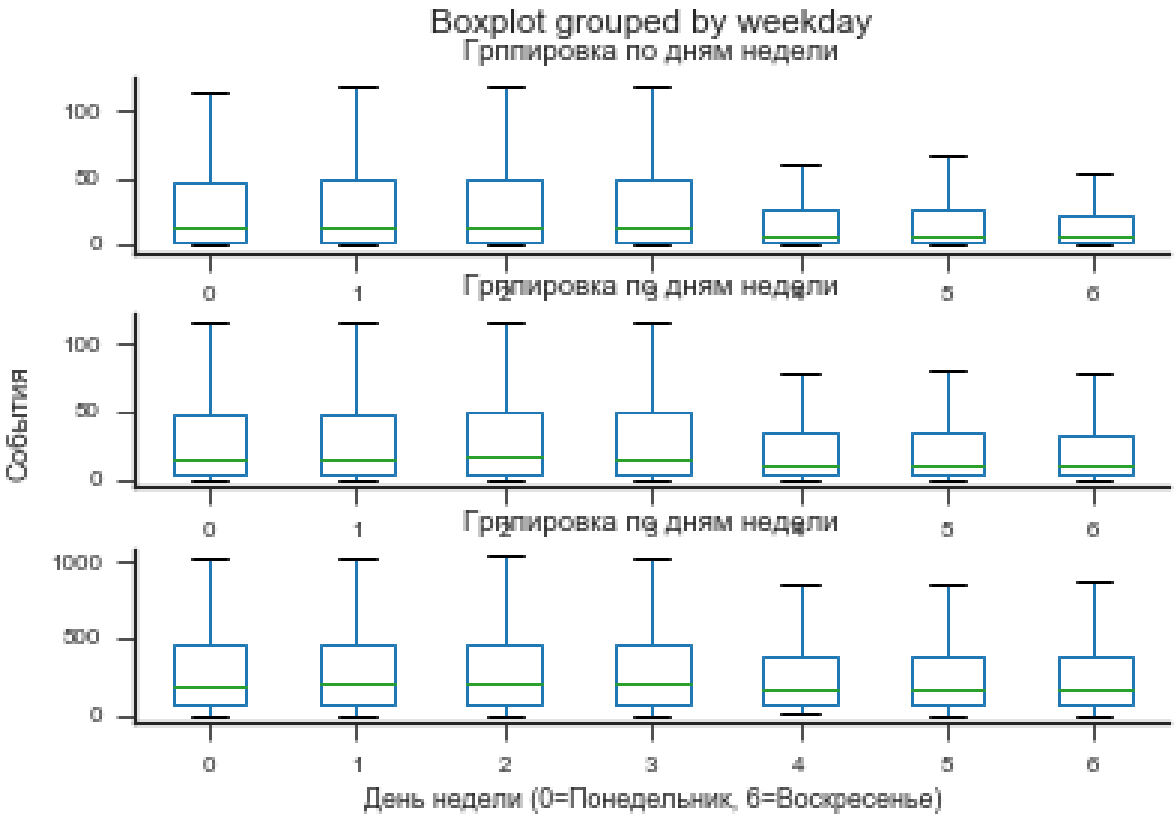

Рисунок 2.10 – Вариативность по дням

Как видно из графика, наименьшей вариативностью данных обладают выходные дни и пятница.

Техническая проверка

Техническая проверка качества наборов данных ограничена из-за отсутствия аналогичных наборов данных для сравнения наших результатов. Следовательно, в этом разделе предлагается статистическая и визуальная характеристика с целью поддержки наивной правильности предоставленной информации.

Анализ остатков

Остатки регрессии - это различия между наблюдаемыми значениями и прогнозными значениями для исследуемой регрессионной модели.

Чем лучше регрессионная модель в соответствии с данными, тем меньше остатки. Остатки (e_i) рассчитываются как (1):

$$e_i=(y_i-\hat{y}_i) \tag{1}$$

где

y_i - наблюдаемое значение;

$\hat{y}_i$ - соответствующее предсказанное значение.

В терминах матриц можно записать также (2,3):

$$e = y - \hat{y} = y - X(X^T X)^{-1} X^T y \quad (2)$$

$$e = y - Hy = (I - H)y \quad (3)$$

где H - проекционная матрица (hat matrix)

Дисперсия остатков (4):

$$Var(e) = \sigma^2 (I - H) \quad (4)$$

где σ^2 - дисперсия ошибок модели

Дисперсия *i-го* остатка (5):

$$Var(e_i) = \sigma^2 (1 - h_{ii}) \quad (5)$$

Стандартное отклонение *i-го* остатка (6):

$$\sigma(e_i) = \sigma \sqrt{(1 - h_{ii})} \quad (6)$$

В качестве анализа остатков можно взять ту же ячейку 5060 и сравнить остатки с моделью синусоиды (Рисунок 2.11). Листинг кода приведен в Приложении Б (Рисунок Б.4)

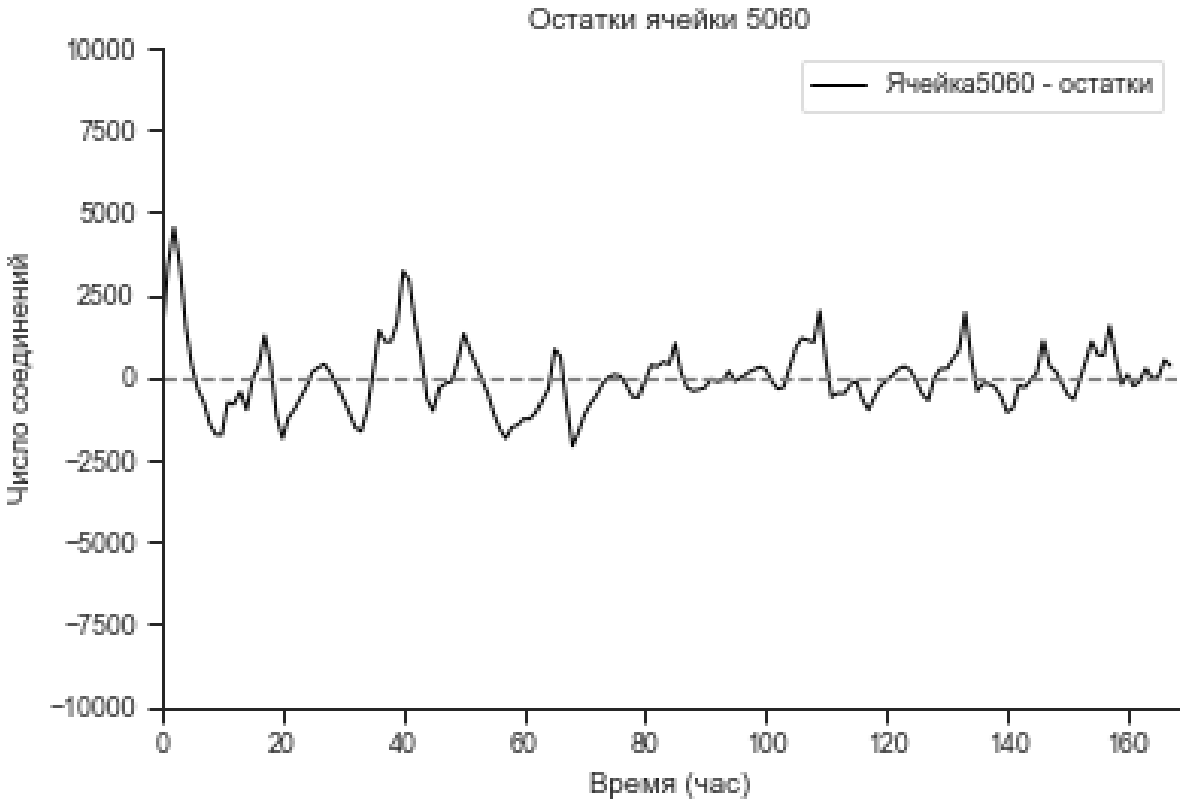

Рисунок 2.11 – Остатки ячейки 5060

В качестве проверки стационарности остатков, можно воспользоваться тестом Дикки-Фуллера [20].

Тест Дики-Фуллера является средством проверки стационарности временного ряда.

Нулевая гипотеза H_0: $g = 0$ (существует единичный корень, ряд нестационарный).

Нулевая гипотеза: H_1: $g < 0$ (единичного корня нет, ряд стационарный).

Отвергаем H_0 на N процентном (N=1%, 5%, 10%) уровне значимости, если $t_1 < t_{\text{крит}}^{N\%}$

В результате теста, можем отвергнуть нулевую гипотезу и считать, что остатки стационарны (Рисунок 2.12)

```
Results of Dickey-Fuller Test:
Test Statistic                -7.405407e+00
p-value                        7.367270e-11
#Lags Used                     1.000000e+00
Number of Observations Used    1.660000e+02
Critical Value (1%)           -3.470370e+00
Critical Value (5%)           -2.879114e+00
Critical Value (10%)          -2.576139e+00
dtype: float64
```

Рисунок 2.12 – Результаты теста

Это дает право сказать, что модель простой синусоиды для ячейки была подобрана верно.

Диаграмма рассеяния

Точечная диаграмма - это метод визуализации для оценки точности регрессионных моделей. Это график, на котором изображены горизонтальные значения (ось) целевых значений (фактически наблюдаемых) обучающих примеров, а вертикальные значения представляют значения, оцененные моделью. Поэтому каждая пара оцененных значений цели и Y может быть представлена на диаграмме в виде точки.

Множество точек, для которых оценочное значение будет равно действительному, составляет так называемую линию идеальных значений для каждой точки, равенство которой Y = Y '. Остальные точки, сформированные примерами, в которых модель допустила ошибку, будут разбросаны по этой линии.

Основываясь на степени диффузии, можно оценить точность модели. Если большинство точек сосредоточено вдоль идеальной линии, а значительные отклонения редки или полностью отсутствуют, модель работает хорошо. Если разброс точек большой, его точность низкая.

Произведем тест на всех ячейках и построим Scatter Plot (Рисунок 2.13)

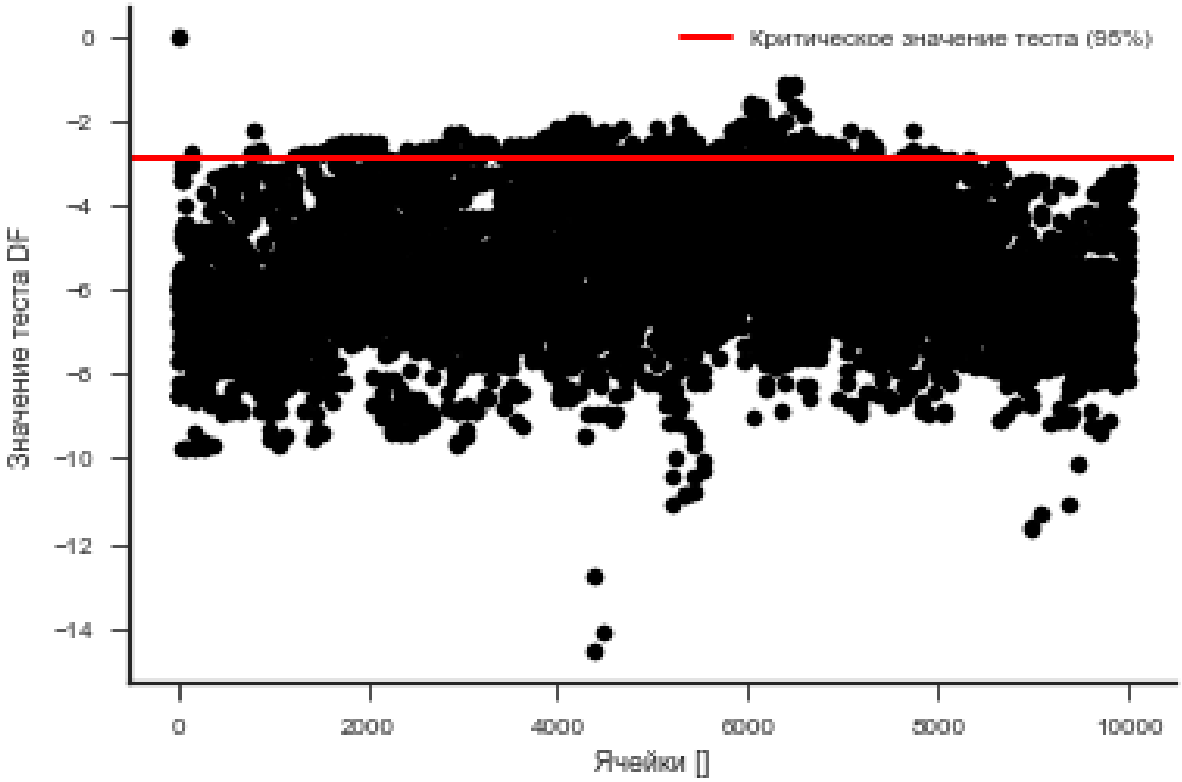

Рисунок 2.13 – Диаграмма рассеяния для всех ячеек

Функция автокорреляции

Функция автокорреляции - это свойство сигнала, которое помогает находить повторяющиеся части сигнала или определять несущую частоту сигнала, которая скрыта на других частотах из-за перекрывающегося шума и вибрации. Автокорреляционная функция часто используется при обработке сигналов и анализе временных рядов.

Выполним анализ ячейки 5060 (Рисунок 2.14)

Рисунок 2.14 – Автокорелляция для ячейки 5060

По графику функции автокорреляции можно увидеть сильную корреляцию на значении временной задержки в 24 часа, что является согласованным с наблюдением по другим графикам.

Функцию автокорреляции можно построить и на остатках, для подтверждения природы данных (Рисунок 2.15). Листинг кода приведен в Приложении Б (Рисунок Б.5)

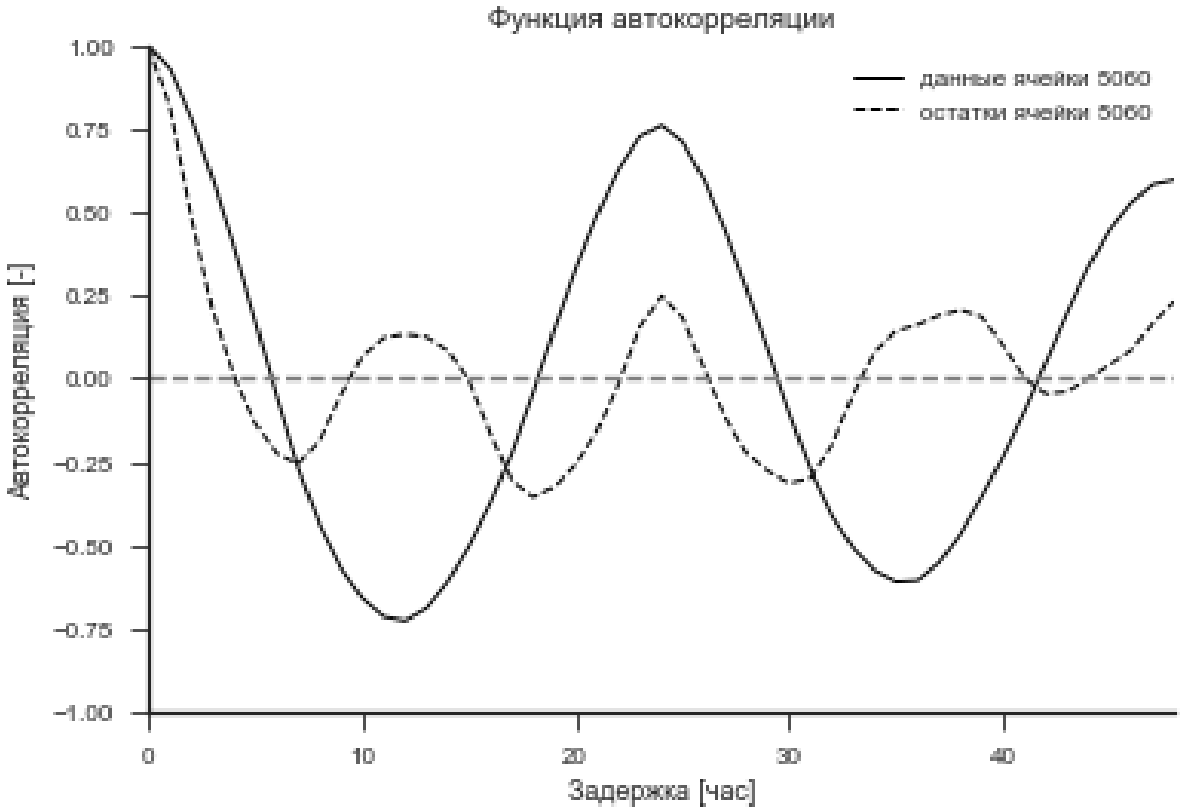

Рисунок 2.15 – Функция автокорреляции для остатков и данных

Как видно из рисунка, автокорреляция для остатков незначительна на участках 12 и 24 часа, что говорит, о близости со статистической стационарностью остатков, что является целью прошлого моделирования.

2.3 Подготовки данных и машинное обучение

Машинное обучение - это класс методов автоматического создания прогнозных моделей на основе данных. Алгоритмы машинного обучения преобразуют набор данных в модель. Какой алгоритм лучше всего работает, зависит от типа решаемой проблемы, доступных вычислительных ресурсов и характера данных.

Алгоритмы обычно говорят компьютеру напрямую, что делать. Например, алгоритмы сортировки преобразуют неупорядоченные данные в данные, отсортированные по некоторым критериям, часто в числовом или алфавитном порядке одного или нескольких полей данных.

Алгоритмы линейной регрессии "сопоставляют" прямую линию с числовыми данными, обычно путем инверсии матрицы, чтобы минимизировать квадратную ошибку между линией и данными. Квадрат ошибки используется в качестве метрики, потому что не имеет значения, находится ли линия регрессии выше или ниже точек данных - важно только расстояние между построенной линией и контрольными точками.

Алгоритмы нелинейной регрессии, которые «адаптируют» кривые (например, полиномы или экспоненциальные) к данным, несколько сложнее: в

отличие от задач линейной регрессии, для них нет детерминированных подходов. Напротив, алгоритмы нелинейной регрессии реализуют определенный процесс итеративной минимизации, часто разновидность метода более крутого спуска.

Наибольший спуск в общем случае включает вычисление квадрата ошибки и ее градиента при текущих значениях параметров, выбор размера шага (это также скорость обучения) в соответствии с направлением градиента «заранее», а затем пересчет квадрата ошибки и его градиента с использованием нового значения параметров. В конце концов, если вам повезет, все пойдет вместе. Изменяя алгоритм наискорейшего спуска, они пытаются улучшить его характеристики сходимости.

Алгоритмы машинного обучения даже более сложны, чем нелинейная регрессия, отчасти потому, что машинное обучение устраняется без ограничения «адаптации» к конкретной математической функции. Есть две основные категории действий, которые часто решаются машинным обучением: регрессия и классификация. Регрессия - для числовых данных (например, каков вероятный доход человека с конкретным адресом и профессией). Классификация - для нечисловых данных (например, если заемщик может погасить кредит).

Прогнозирование (например, какая цена открытия акций Яндекса будет завтра) является подмножеством регрессионных действий для этих временных рядов. Классификационные мероприятия иногда делятся на двоичные (да или нет) и мультикатегории (животные, фрукты или мебель).

При построении прогностических моделей исходные данные обычно делятся на обучающие («учебный комплект») и контрольные образцы («испытательный комплект», «проверочный комплект»). Учебная выборка фактически используется для «обучения» конкретной модели, то есть для построения математических отношений между определенной переменной ответа и предикторами, а контрольная выборка («test», «exam») используется для оценки предсказанных свойств модели на новых данных, данных, которые не использовались для создания модели. Как правило, обучающая выборка составляет 75-80% исходных данных, хотя строгих правил в этом отношении нет.

Нормализация данных является одной из операций преобразования функций, выполняемых при их создании на этапе подготовки данных.

В случае машинного обучения нормализация - это процедура предварительной обработки входной информации (обучающие, тестовые и проверочные образцы, а также фактические данные), в которой значения атрибутов входного вектора сводятся до определенного диапазона, например, [0…1] или [-1…1]

Необходимость стандартизировать выборки данных обусловлена природой алгоритмов и моделей, используемых машинным обучением. Начальные значения характеристик могут варьироваться в очень широком диапазоне и отличаться друг от друга на несколько порядков. Предположим,

что набор данных содержит информацию о концентрации активного вещества, измеряемой в десятых или сотых долях процента, и показателях давления в сотнях тысяч атмосфер. Или, например, во входном векторе есть информация о возрасте и доходе клиента.

Разные по своему физическому значению данные значительно различаются в абсолютном выражении. Работа аналитических моделей машинного обучения (нейронных сетей, карт Кохонена и т. Д.) С такими индикаторами будет некорректной: дисбаланс между значениями атрибутов может привести к нестабильности модели, ухудшению результатов обучения и замедлению процесса моделирования [14]. В частности, параметрические методы машинного обучения (нейронные сети, деревья решений) обычно требуют симметричного и унимодального распределения данных. Популярный метод ближайшего соседа, часто используемый в задачах классификации, а иногда и в регрессионном анализе, также чувствителен к диапазону изменения входных переменных.

После нормализации все числовые значения входных объектов будут уменьшены до той же области их модификации - некоторого узкого диапазона. Это объединит их в одну модель машинного обучения и гарантирует правильное функционирование алгоритмов расчета.

Рассмотрим One-Hot кодирование. Для категориальных переменных, где порядковые отношения не существуют, кодирование целых чисел недостаточно.

Фактически, использование этого кода и возможность естественной сортировки модели между признаками может привести к снижению производительности или неожиданным результатам (предсказания на полпути между категориями).

В этом случае однозначное кодирование может быть применено ко всему представлению. В этом случае целочисленная переменная удаляется, и для каждого уникального целочисленного значения добавляется новая двоичная переменная.

В «цвете, например, есть 3 категории, поэтому необходимы 3 двоичные переменные. Значение «1» помещается в двоичную переменную для цвета, а значение «0» - для других цветов.

Например:

красный, зеленый, синий

1 0 0

0, 1, 0

0, 0, 1

Бинарные переменные часто называют «фиктивными переменными» в других областях, таких как статистика.

Типы машинного обучения

Контролируемое машинное обучение (подготовка учителей)

Контролируемое обучение [17,18] создает модель, которая делает прогнозы на основе фактических данных. Методы регрессии и классификации используются для разработки этих прогностических моделей.

Линейная регрессия: Линейная регрессия выражает линейную зависимость между входом и выходом.

Полиномиальная регрессия: выражает полиномиальную регрессию между входом и выходом. Это дает более точный результат по сравнению с линейной регрессией.

Дерево решений (также называемое деревом классификации или деревом регрессии) является инструментом поддержки принятия решений, используемым в машинном обучении, анализе данных и статистике [13]. Древовидная структура состоит из «листьев» и «ветвей». По краям («ветвям») дерева решений записываются атрибуты, от которых зависит целевая функция, значения целевой функции записываются в «листьях», а атрибуты используются для различения случаев в других узлах. Чтобы классифицировать новый случай, вы должны спуститься с дерева на лист и потратить соответствующее значение. Подобные деревья решений широко используются в интеллектуальном анализе данных. Цель состоит в том, чтобы создать модель, которая прогнозирует значение целевой переменной на основе нескольких входных переменных.

Случайные леса: случайные леса используют много деревьев решений. Они представляют собой набор деревьев решений [15]. Каждое дерево решений создается с использованием подмножества атрибутов. Случайный лес - эффективный алгоритм управляемой классификации.

Неконтролируемое машинное обучение (обучение без учителя)

Обучение без учителей выявляет закономерности данных, делая выводы из безымянного ввода.

Группировка: группировка объектов в кластеры. Алгоритм группировки - это k-means, который определяет лучшие k-кластерные центры в итерационной форме.

Уменьшение размера: распределение по одному значению (SVD) - это метод декомпозиции матрицы, используемый для сокращения матрицы до ее составных частей. Основным алгоритмом является принцип анализа компонентов (PCA).

Анализ ассоциаций: помогает найти наиболее используемые и самые большие наборы элементов в больших наборах данных. Используется для обнаружения зависимостей между переменными в больших базах данных.

Применение алгоритмов машинного обучения

В качестве алгоритма машинного обучения выберем случайный лес. Для того, чтобы алгоритм правильно работал с нашими признаками, сделаем нормализацию данных. Все листинги кода приведены в Приложении В

1) Заполним все пустые признаки
2) Выполним Dummy-кодирование – генерация признаков звонков
3) Нормализация – делим остаток на среднеквадратичное отклонение

4) Делим выборку на обучающую и тестовую
5) Добавляем аномальные значения в обучающую выборку, столбец anomaly со значением 1
6) Находим похожие аномалии в реальных данных (Рисунок 2.16)

```
callin_Denmark              0.350033
internet_Ethiopia           0.925006
CellID                   6137.000000
anomaly                     1.000000
Name: 679, dtype: float64
```

Рисунок 2.16 – Найденные аномалии по обученному алгоритму

2.4 Методология анализа данных

Исходя из предыдущих пунктов, можно разработать свою методологию изучения и анализа данных предметной области.

1) Определить предметную область, собрать информацию о значении признаков для интерпретации результатов

2) Оценить объем выборки, агрегировать источники (в случае если выборка разбита на части)

3) Определить основные типы данных (количественные, категориальные, логические и т.д.), наличие пропущенных значений

4) Сделать первичный анализ при помощи средних, максимальных, минимальных значений, подсчета количества значений – при наличии категорий, выполнить анализ BoxPlot

5) Увеличить количество признаков, путем трансформации существующих. К примеру, из типа данных «Время» можно выделить день недели, месяц, или просуммировать два столбца назвав “Sum”

6) Использовать гистограммы для определения распределений признаков, агрегируя в Pandas строить интересующие зависимости (к примеру количество звонков в час)

7) Для определения модели распределения использовать подстройку данных под модель и построить оба графика на одной оси

8) Для определения корреляции использовать статистические библиотеки по определению уровня корреляции/автокорреляции

9) Использовать статистические тесты для подтверждения гипотез

10) Для пространственных признаков строить HeatMap/GeoJSON отображения

11) Для подготовки к машинному обучению выполнить нормировку данных – избавление от выбросов, заполнение ненулевых значений, приведение категориальных признаков в OneHot Encoded признаки, поделить выборку на обучающую и тестовую

12) Используя таблицу ошибок определить эффективность алгоритма.

3 Аналитика данных приложений

3.1 Архитектура сбора метрик со стороны приложений

Производительность приложений состоит из ряда параметров, которые характеризуют производительность и доступность ИТ-услуг в различных частях распределенной ИТ-инфраструктуры, от отдельных компонентов ИТ до конечных пользователей. Инструменты мониторинга приложений необходимы для мониторинга требуемого качества услуг. Популярным инструментом мониторинга является пакет Prometheus + Grafana.

Prometheus - это временные ряды СУБД с открытым исходным кодом (Apache 2.0), написанные на Go и изначально разработанные SoundCloud. Другими словами, этот инструмент хранит временные данные. Интересной особенностью Prometheus является то, что он извлекает данные из определенного набора сервисов. Из-за этого Прометей не использует очереди или буферы, что означает, что мониторинг никогда не становится узким местом системы

Prometheus имеет центральный компонент, который называется Prometheus Server. Основная задача - сохранить и контролировать некоторые объекты. Объектом может стать все: сервер Linux, сервер Apache, один из процессов, сервер базы данных или другой системный компонент, которым вы хотите управлять. Что касается Прометея, то основная служба наблюдения называется сервером Прометея, а объекты наблюдения называются целями. Целью может быть один сервер или цели для проверки конечных точек по HTTP, HTTPS, DNS, TCP и ICMP (* Black Box Exporter) или простая конечная точка HTTP, вызывающая приложение. Сервер Prometheus проверяет состояние приложения через конечную точку HTTP.

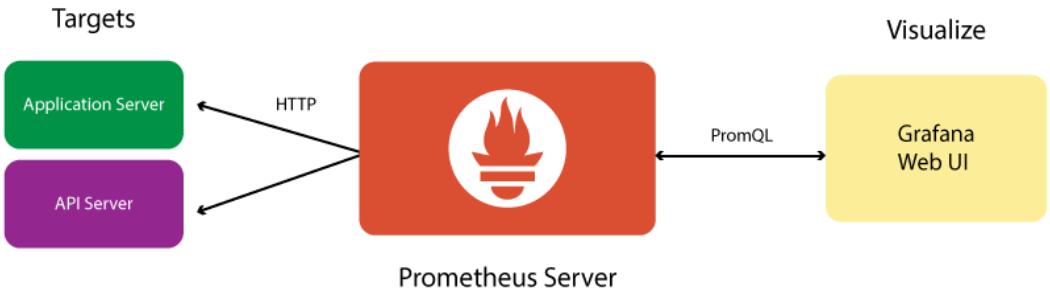

Рисунок 3.1 – Схема работы мониторинга приложений

Каждый элемент цели, которую вы хотите отслеживать (состояние процессора, память или любой другой элемент), называется метрикой. Таким образом, Prometheus собирает целевые объекты с использованием HTTP-метрик, сохраняет их локально или удаленно и отображает их.

Сервер Prometheus считывает точки назначения с интервалом, который вы определяете для сбора метрик, и сохраняет их в базе данных временных рядов.

Grafana - это открытый интерфейс (Apache 2.0) для нескольких СУБД временных рядов, таких как Graphite, InfluxDB и, конечно, Prometheus. В общем, Grafana рисует отличную графику и таблицы, используя информацию от Прометея (Рисунок 3.2).

Рисунок 3.2 – Дашборд в графане

Поскольку Grafana фокусируется исключительно на пользовательском интерфейсе, он должен получать данные где-то снаружи. Он поддерживает Графит, InfluxDB, Prometheus и многие другие агрегаторы по умолчанию (рис. 3.3). Если стандартных недостаточно, можно найти другие типы источников данных среди плагинов.

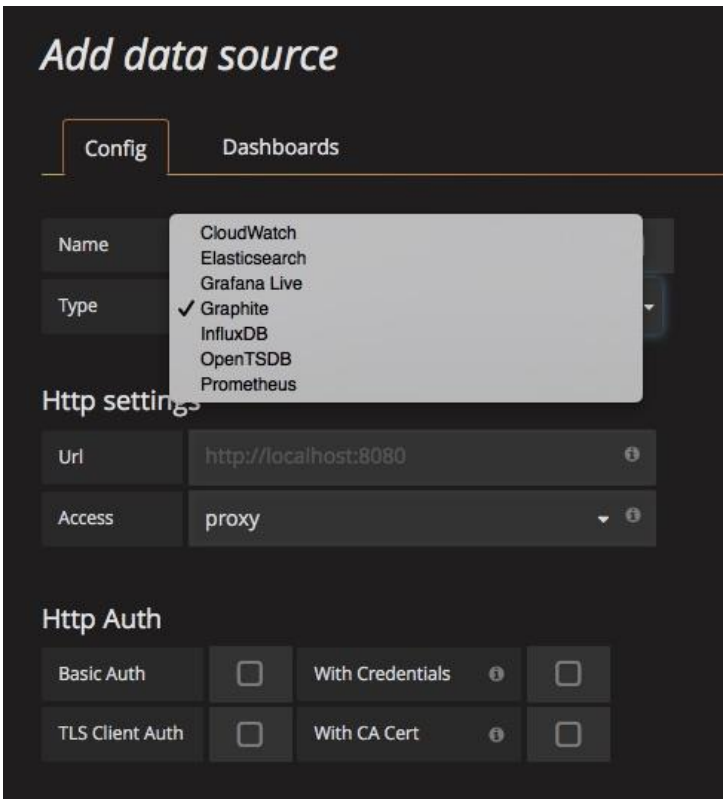

Рисунок 3.3 – Поддерживаемые источники данных в Grafana

Архитектура сбора метрик для приложений включает в себя веб-сервер Nginx, который является точкой входа для пользователей в приложения (рисунок 3.4).

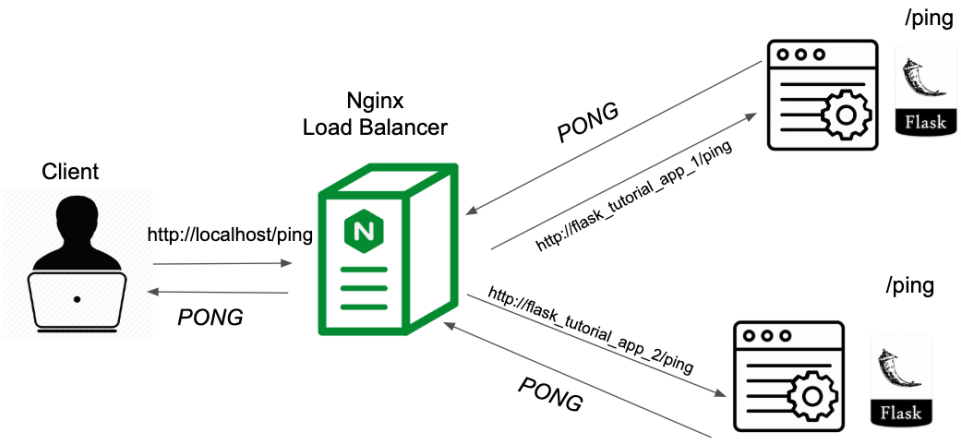

Рисунок 3.4 – Веб сервер Nginx фиксирует пользовательскую активность.

NGINX - это программное обеспечение, написанное для систем UNIX. Основное назначение - это отдельный HTTP-сервер или, как это чаще всего используется, интерфейс для сильно загруженных проектов. NGINX может

использоваться как почтовый сервер SMTP / IMAP / POP3, а также как обратный прокси-сервер TCP.

Некоторые метрики хранятся в базе данных PostgreSQL и отображаются так же в Grafana.

3.2 Сравнение активности приложений с трафиком оборудования

Активные пользователи - это люди, которые связываются с вашей компанией в течение определенного периода времени, который вы можете идентифицировать по идентификатору, адресу электронной почты или логину. Для большинства компаний активный пользователь - это тот, кто создал учетную запись (учетную запись), и для осуществления любого взаимодействия которому необходимо войти на сайт.

Проведем исследование над тремя цифровыми продуктами, где посмотрим в Grafana активность пользователей услугами сети по трем разным направлениям. Визуализируем в Grafana метрики по Nginx (Рисунок 3.5, 3.6, 3.7)

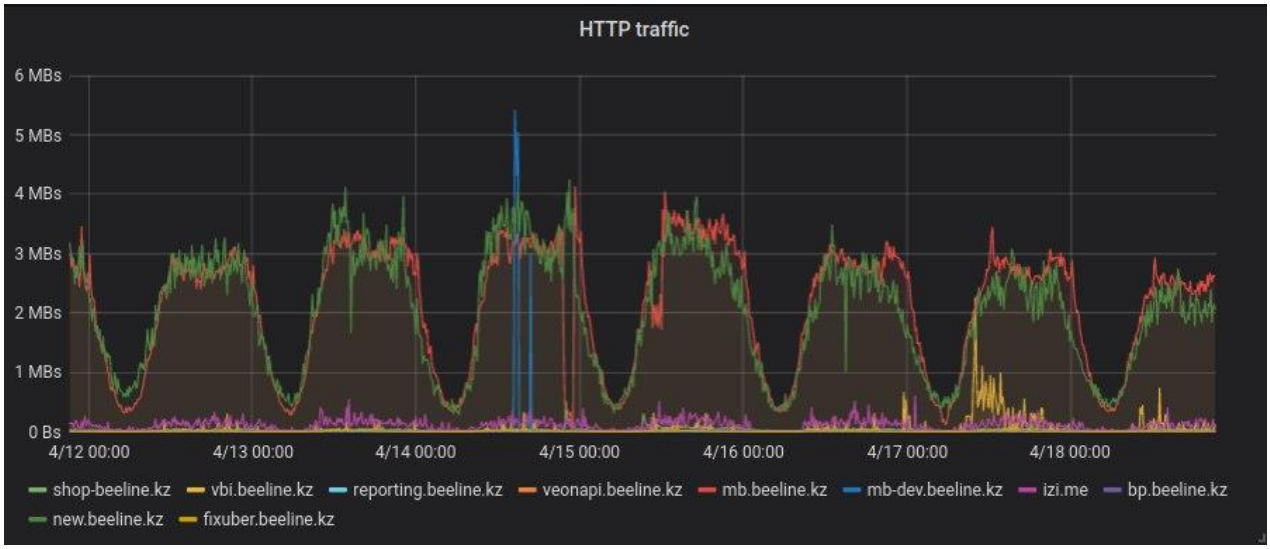

Рисунок 3.5 – Трафик на стороне веб сервера NGINX

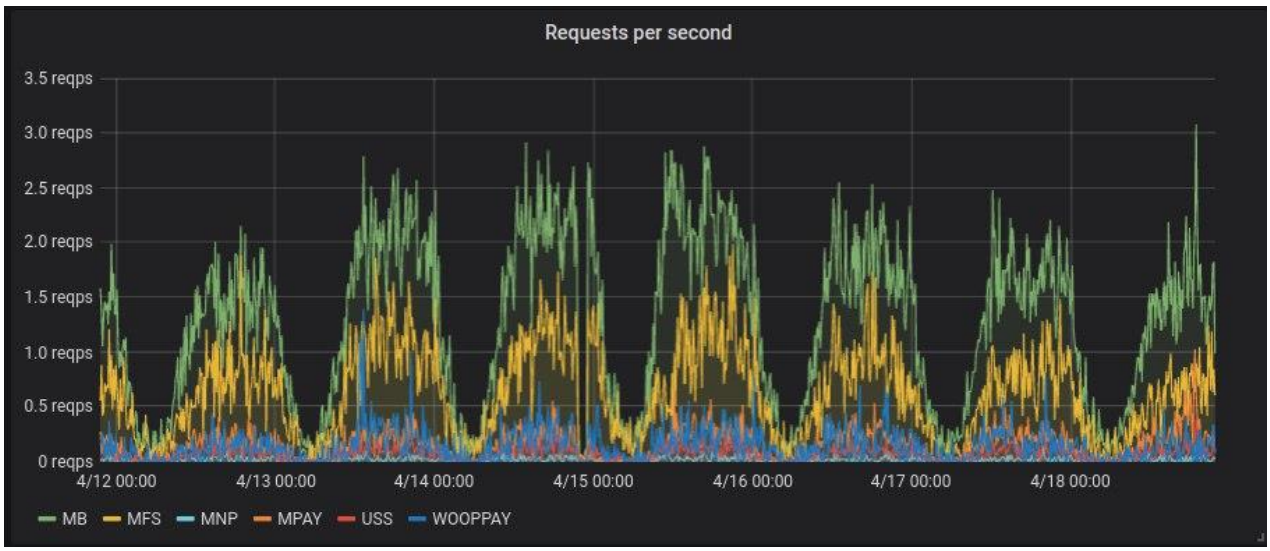

Рисунок 3.6 – Активность пользователей в секунду

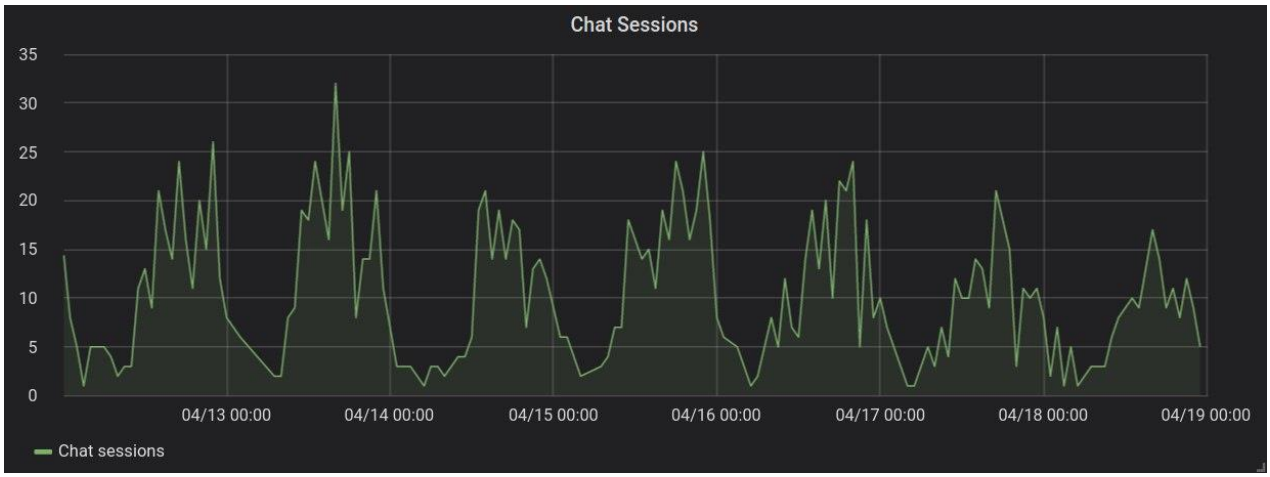

Рисунок 3.7 – Анализ активности по чату в цифровом продукте

При составлении графика по NGINX трафику, видно, что пользовательская активность так же подчиняется закону синусоиды, как в случае с анализом интернет/смс соединений на оборудовании.

В связи со сложившейся ситуацией с пандемией коронавируса, можно наблюдать больший спрос интернет услугами и повышенную активность с апреля (Рисунок 3.8)

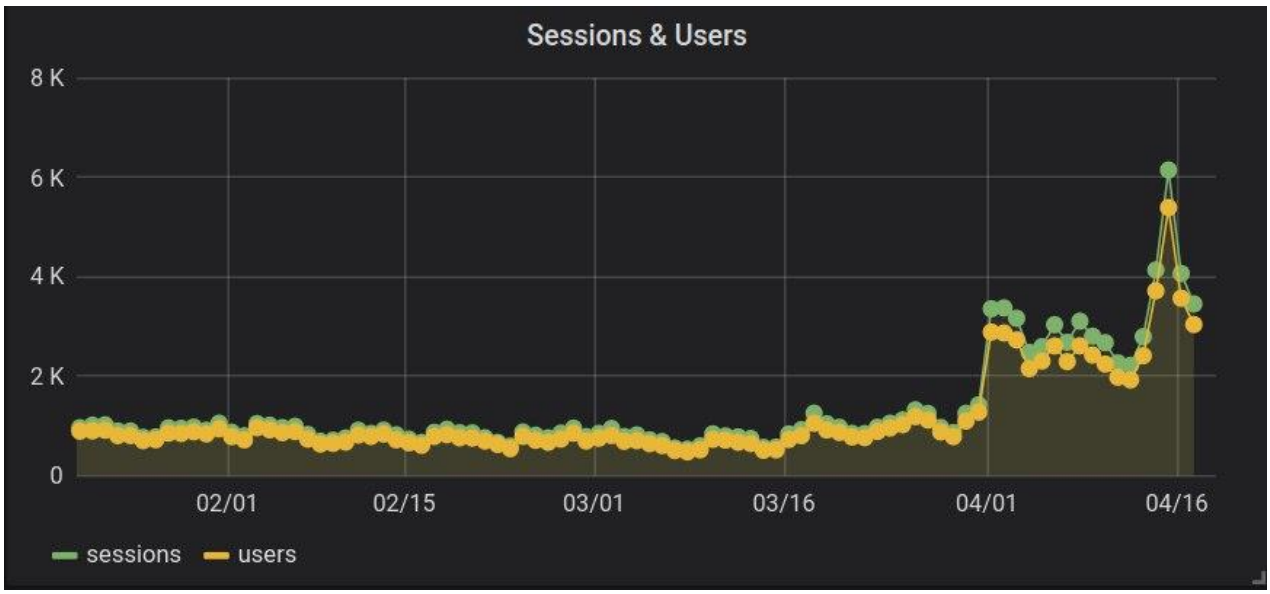

Рисунок 3.8 – Повышение активности из-за карантина

Заключение

Интеллектуальный анализ данных является одной из наиболее актуальных и востребованных областей прикладной математики. Современные бизнес и производственные процессы генерируют большие объемы данных, и людям становится все труднее интерпретировать и реагировать на эти объемы, которые динамически изменяются во время выполнения, не говоря уже о предотвращении критических ситуаций. «Интеллектуальный анализ данных» извлекает максимум полезных знаний из разнородных, неполных, неточных, противоречивых, косвенных данных и это полезно, когда объем данных измеряется в гигабайтах или даже терабайтах. Анализ данных помогает создавать алгоритмы, которые поспобствуют вам обучению принимать решения в различных профессиональных областях. Инструменты интеллектуального анализа данных защищают людей от информационной перегрузки, превращая оперативные данные в полезную информацию, чтобы они могли принимать правильные меры в краткие сроки.

В данной работе была рассмотрена доменная область телекоммуникаций, датасет состоял из нескольких файлов, и получены следующие результаты:

1) Освоены библиотеки Python: Pandas, Matplotlib, SciPy, NumPy

2) Произведена первичная предобработка и анализ данных

3) Подобрана модель использования данных и статистически доказана стационарность остатков – мобильные данные же являются нестационарными по природе

4) Построены карты горячих точек и пиковых значений

5) Приведен пример поиска аномалий с использованием машинного обучения

6) Разработаны рекомендации по анализу данных

Интеллектуальный анализ данных включает в себя такие процессы, как очистка данных, интеграция данных, преобразование данных, представление данных. После завершения всех этих процессов мы можем использовать эту информацию во многих приложениях, таких как обнаружение мошенничества, анализ рынка, контроль производства, исследования и прочие.

Список литературы

1 Uthurusamy, Fayyad, Piatetsky-Shapiro, Smyth. Advances in Knowledge Discovery and Data Mining, (Chapter 1) AAAI/MIT Press 1996

2 Вон Ким. Три основных недостатка современных хранилищ данных. Открытые системы, 2003, №2

3 B. Choi, D. Lee, E. Hong, S. Kim, W. Kim. A Taxonomy of Dirty Data. Journal of Data Mining and Knowledge Discovery, the Kluwer Academic-Publishers, 2003

4 Erhard Ram, Hong Hai Do. Очистка данных: проблемы и актуальные подходы

5 S. Park, M. S. Chen and P. S. Yu Efficient Hash-Based Algorithm for Mining Association Rules Proc. Int'l Conf. Information and Knowledge Management, Baltomore, Md., Nov. 1995

6 A. Savasere, and S. Navathe, E. Omiecinski An Efficient Algorithm for Mining Association Rules in Large Databases Proc. 21st Int'l Conf. Very Large Data Bases, Morgan Kaufmann, San Francisco, 1995, pp. 432-444

7 S. Brin et al Dynamic Itemset Counting and Implication Rules for Market Basket Data Proc. ACM SIGMOD Int'l Conf. Management of Data, ACM Press, New York, 1997, pp. 255-264

8 У. Боумен Графическое представление информации М.: "Мир", 1971

9 «Технологии анализа данных: Data Mining. Visual Mining. Text Mining, OLAP» А. А. Барсегян. М. С. Куприянов, В. В. Стенаненко, И. И. Холод. — 2-е изд., перераб. и доп.

10 Edward. R. Tufte The Visual Display of Quantitative Information Graphics Press. 2001

11 Филип Рассом Тенденции программного обеспечения в области визуализации данных для бизнес-пользователей 01.05.2000

12 Плотинский Ю.М Визуализация информации М., 1994

13 A. Criminisi J. Shotton E. K. Decision forests for classification, regression, density estimation, manifold learning and semi-supervised learning.: Tech. Rep. MSR-TR2011-114: Microsoft Research, 2011. - Oct.

14 Blum A., Mitchell T. Combining labeled and unlabeled data with co-training // Proceedings of the Eleventh Annual Conference on Computational Learning Theory. COLT' 98. - New York, NY, USA: ACM, 1998. Pp. 92 100. http://doi.acm.org/10.1145/279943.279962.

15 Breiman L. Random forests // Machine Learning. - 2001. - Vol. 45, no. 1. - Pp. 5-32. http://www.cs.colorado.edu/~grudic/teaching/CSCI5622_2004/RandomForests_ML_Journal.pdf.

16 M. Zaharia M. Chowdhury T. D. A. D. J. M. M. M. M. J. F. S. S. I. S. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing // Presented as part of the 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI 12). - San Jose, CA: USENIX, 2012.

Pp. 15-28. https://www.usenix.org/conference/nsdi12/technical-sessions/presentation/zaharia.

17 O. Chapelle B. Schlkopf A. Z. Semi-Supervised Learning. 1st edition. - The MITPress, 2010.

18 P. Mallapragada R. Jin A. K. J. Y. L. Semiboost: Boosting for semi-supervised learning // IEEE Trans. Pattern Anal. Mach. Intell. - 2009. Vol. 31, no. 11.-Pp. 2000-2014. http://dx.doi.org/10.1109/TPAMI.2008.235.

19 Saari A. Multi-Class Semi-Supervised and Online Boosting: Ph.D. thesis / Graz University of Technology, Faculty of Computer Science. 2010.

20 Vapnik V. N. Statistical Learning Theory. - Wiley-Interscience, 1998.

21 Chernoff, H The Use of Faces to Represent Points in K-Dimensional Space Graphically Journal of American Statistical Association, 68, 361-368

Приложение А
Листинги кода для визуализации данных

```
f = plt.figure()

ax = df_cdrs_internet[df_cdrs_internet.CellID==5060]['internet'].plot(label='Ячейка5060')
df_cdrs_internet[df_cdrs_internet.CellID==4259]['internet'].plot(ax=ax, label='Ячейка4259')
df_cdrs_internet[df_cdrs_internet.CellID==4456]['internet'].plot(ax=ax, label='Ячейка4456')
plt.xlabel("Недельные часы")
plt.ylabel("Количество соединений")
sns.despine()

# Shrink current axis's height by 10% on the bottom
box = ax.get_position()
ax.set_position([box.x0, box.y0 + box.height * 0.1,
                 box.width, box.height * 0.9])
ax.legend(loc='upper center', bbox_to_anchor=(0.5, -0.15),
          fancybox=True, shadow=True, ncol=5)
```

Рисунок А.1 - Листинг кода для визуализации недельных часов

```
boxplots = {
    'calls': "Calls",
    'sms': "SMS",
    "internet": "Internet CDRs"
}

df_cdrs_internet['weekday'] = df_cdrs_internet.datetime.dt.weekday

f, axs = plt.subplots(len(boxplots.keys()), sharex=True, sharey=False)
f.subplots_adjust(hspace=.35,wspace=0.1)
i = 0
plt.suptitle("")
for k,v in boxplots.items():
    ax = df_cdrs_internet.reset_index().boxplot(column=k, by='weekday', \
                                                grid=False, sym='', ax =axs[i])
    axs[i].set_title("Грппировка по дням недели")
    axs[i].set_xlabel("")
    sns.despine()
    i += 1

plt.xlabel("День недели (0=Понедельник, 6=Воскресенье)")
f.text(0, 0.5, "События", rotation="vertical", va="center")
```

Рисунок А.2 - Листинг кода для визуализации Box Plot

```
import geojson
import matplotlib.colors as colors
import matplotlib.cm as cmx
#import matplotlib as mpl
from descartes import PolygonPatch

num = int(10000+1)
arr_cellID = np.zeros(num)
arr_mean = np.zeros(num)
for i in range(1,num):
    ydata = df_cdrs_internet[df_cdrs_internet.CellID==i]['internet']
    xdata = df_cdrs_internet[df_cdrs_internet.CellID==i]['internet'].index
    mean = np.mean(ydata)
    arr_cellID[i]=i
    arr_mean[i]=mean

arr_mean[arr_mean<=0] = 1 #replacing 0's with 1's for log calc
arr_mean_log = np.log(arr_mean)

#https://gis.stackexchange.com/questions/93136/how-to-plot-geo-data-using-matplotlib-python
with open("../input/grid.geojson") as json_file:
    json_data = geojson.load(json_file)


fig = plt.figure()
ax = fig.gca()

coordlist = json_data.features[1]['geometry']['coordinates'][0]

jet = cm = plt.get_cmap('hot')
#cNorm  = colors.Normalize(vmin=0, vmax=np.max(arr_mean))
cNorm  = colors.Normalize(vmin=0, vmax=5000)
#cNorm  = colors.Normalize(vmin=0, vmax=np.max(arr_mean_log))
scalarMap = cmx.ScalarMappable(norm=cNorm, cmap=jet)
#print(scalarMap.get_clim())

for i in range(1,10000):
    poly = json_data.features[i]['geometry']
    colorVal = scalarMap.to_rgba(arr_mean[i])
    ax.add_patch(PolygonPatch(poly, fc=colorVal, ec=colorVal, alpha=1, zorder=1 ))
ax.axis('scaled')

fig.set_size_inches(11,11)
plt.title("Интернет соединения в городе (Черное ~ 0 соединений, Белое ~ 5000 соединений)")
plt.show()
```

Рисунок А.3 - Листинг кода для рендеринга горячих точек города

Приложение Б
Листинги кода для моделирования данных

```
import scipy
def func(xdata, a,b,c):
    return a*np.sin(2*np.pi*(1/24)*xdata+b)+c
```

```
popt,pcov = scipy.optimize.curve_fit(func, xdata, ydata)

print(popt)

f = plt.figure()
yfit = func(xdata, *popt)
#residual
residual = ydata - yfit
rss = np.sum(residual**2)
mean = np.mean(ydata)
b = popt[1] #phase shift (b) from curve_fit
#print('rss_navigli',rss_navigli,'mean_navigli',mean_navigli,'rss-norm',rss_navigli/mean_navigli)
#stddev = np.std(residual_navigli)
#print(np.std(residual_navigli))

yfit_duomo = yfit
b_duomo = b
ydata_duomo = ydata
xdata_duomo = xdata
T_peak_duomo = 18-(b_duomo%(2*np.pi))*(24/(2*np.pi))

ydata_moving_avg = ydata.rolling(window=24,center=False).mean()
residual_moving_avg = residual.rolling(window=24,center=False).mean()
```

```

f = plt.figure()
plt.plot(xdata, ydata, color='black', linewidth=1, linestyle='--', label='Ячейка5060 - данные')
plt.plot(xdata, yfit, color='black', linewidth=3, label='Ячейка5060 - модель')
plt.xlabel("Время (час)")
plt.ylabel("Количество интернет соединений")
plt.xlim([0,168])
plt.ylim([0,10000])
plt.legend()
sns.despine()
plt.show()
```

Рисунок Б.1 – Листинг кода для составления модели данных и визуализации

```
fig = plt.figure()
ax = fig.gca()

jet = cm = plt.get_cmap('hot')
cNorm  = colors.Normalize(vmin=12, vmax=20)
#cNorm  = colors.Normalize(vmin=0, vmax=np.max(arr_T_peak))
scalarMap = cmx.ScalarMappable(norm=cNorm, cmap=jet)

for i in range(1,10000):
    poly = json_data.features[i]['geometry']
    colorVal = scalarMap.to_rgba(arr_T_peak[i])
    ax.add_patch(PolygonPatch(poly, fc=colorVal, ec=colorVal, alpha=1, zorder=2 ))
ax.axis('scaled')
plt.title("Использование интернета по часам (Черное=Полдень, Белое=22:00)")
fig.set_size_inches(11,11)
plt.show()
```

Рисунок Б.2 – Листинг кода для визуализации пиковых точек на карте

```
f = plt.figure()
#plt.plot(xdata_duomo, ydata_duomo, color='black', linewidth=1, linestyle='--', label='Duomo - data')
plt.plot(xdata_duomo, yfit_duomo, color='black', linewidth=3, label='Модель ячейки 5060')
#plt.plot(xdata_navigli, ydata_navigli, color='green', linewidth=1, linestyle='--', label='Navigli - data')
plt.plot(xdata_navigli, yfit_navigli, color='green', linewidth=3, label='Модель ячейки 4456')
plt.axvline(x=T_peak_duomo, color='red')
plt.axvline(x=T_peak_navigli, color='red')

plt.xlabel("Время [час]")
plt.ylabel("Интернет соединения [#]")
plt.xlim([0,24])
plt.ylim([0,10000])
plt.legend()
sns.despine()
```

Рисунок Б.3 – Листинг кода для визуализации пиковых различий

```
ff = plt.figure()
plt.plot(xdata, residual, color='black', linewidth=1, linestyle='-', label='Ячейка5060 - остатки')
plt.axhline(y=0,linestyle='--',color='gray')
plt.title("Остатки ячейки 5060")
plt.xlabel("Время (час)")
plt.ylabel("Число соединений")
plt.xlim([0,168])
plt.ylim([-10000,10000])
plt.legend()
sns.despine()
plt.show()
```

Рисунок Б.4 – Листинг кода для визуализации остатков

```
#Plot ACF:
from statsmodels.tsa.stattools import acf, pacf
acf_r = acf(residual, nlags=48)

f = plt.figure()

plt.plot(acf_y, color='black', linewidth=1, linestyle='-', label='данные ячейки 5060')
plt.plot(acf_r, color='black', linewidth=1, linestyle='--', label='остатки ячейки 5060')

plt.axhline(y=0,linestyle='--',color='gray')
plt.title('Функция автокорреляции')
plt.xlabel("Задержка [час]")
plt.ylabel("Автокорреляция [-]")
plt.xlim([0,48])
plt.ylim([-1,1])
plt.legend()
sns.despine()
plt.show()
```

Рисунок Б.5 – Листинг кода для визуализации функции автокорреляции

Приложение В
Листинг кода для применения машинного обучения

```
# Add Random Anomalies
def add_anomalies(data, percentage=0.01):
    n_anomalies = int(percentage * len(data)) # 1% of data
    data = data.append(pd.DataFrame(np.zeros((n_anomalies, data.shape[1])), columns=data.columns))
    for index in range(len(data) - n_anomalies, len(data)):
        # Random connections
        n_random_indices = np.random.randint(4, 10)
        random_indices = np.random.randint(0, data.shape[1],  n_random_indices)
        data.iloc[index, random_indices] = np.random.uniform(-1.0, 1.0, size=n_random_indices)
    return data, n_anomalies

data, n_anomalies = add_anomalies(data)
data.tail()
```

```
# Add CellIDs
data['CellID'] = 0
data.loc[data.index.values[:-n_anomalies], 'CellID'] = np.array([cell_index for _, cell_index in df.index.values])
data.loc[data.index.values[-n_anomalies:], 'CellID'] = np.random.randint(0, n_cells, n_anomalies)
```

```
# Add Labels
data['anomaly'] = 0
data.loc[data.index.values[-n_anomalies:], 'anomaly'] = 1
```

```
from sklearn.model_selection import train_test_split
train_data, test_data = train_test_split(data, test_size=0.2)
```

```
test_data.head()
```

```
def batches(data, batch_size=1024):

    n_batches = len(data)//batch_size
    print('Data consists of {} batches'.format(n_batches))
    sys.stdout.flush()

    bar = progressbar.ProgressBar(max_value=n_batches)
    for batch_index in range(0, n_batches):

        bar.update(batch_index)

        batch = data.iloc[batch_index*batch_size:(batch_index+1)*batch_size]

        one_hot = np.zeros((batch_size, n_cells))
        for i in range(len(one_hot)):
            one_hot[i, batch['CellID'].values[i]] = 1
        one_hot = pd.DataFrame(one_hot, columns=['CellID_'+str(i) for i in range(n_cells)], index=batch.index)
        batch = batch.join(one_hot)

        yield batch
```

```
name = 'Random Forest'
clf = RandomForestClassifier()

print('Training {}'.format(name))
for batch in batches(train_data):
    print(batch)
    clf.fit(batch.drop(['CellID', 'anomaly'], axis=1), batch['anomaly'])

print('Evaluating {}'.format(name))
y_true = np.array([])
y_pred = np.array([])
for batch in batches(test_data, batch_size=4096):
    y_true = np.append(y_true, batch['anomaly'].values)
    y_pred = np.append(y_pred, clf.predict(batch.drop(['CellID', 'anomaly'], axis=1)))

acc = accuracy_score(y_true, y_pred)
print('Accuracy of {}: {:7.3f}%'.format(name, acc * 100))
print('Confusion Matrix:\n{}'.format(confusion_matrix(y_true, y_pred)))
```

```
indices = np.where(y_pred == 1)
index = indices[0][0]
anomaly_example = test_data.iloc[index]
cell_id = int(anomaly_example['CellID'])
print(anomaly_example[anomaly_example > 0], end='\n\n')
```