

**«Санкт-Петербургский государственный электротехнический университет
«ЛЭТИ» им. В. И. Ульянова (Ленина)»
(СПбГЭТУ «ЛЭТИ»)**

Направление	09.04.02 - Информационные системы и технологии
Профиль	Распределительные вычислительные системы и комплексы реального времени
Факультет	КТИ
Кафедра	ИС

К защите допустить

Зав. кафедрой

Цехановский В.В.

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
МАГИСТРА**

Тема: ВЫЯВЛЕНИЕ ПАТТЕРНОВ ФУНКЦИОНИРОВАНИЯ ДАТЧИКОВ ДЛЯ ДИАГНОСТИКИ СОСТОЯНИЯ ЗДАНИЯ

Студент		<hr/>	Уруков С. Д.
		<i>подпись</i>	
Руководитель	к.т.н., доцент (Уч. степень, уч. звание)	<hr/>	Новикова Е. С.
		<i>подпись</i>	
Антиплагиат	к.т.н., доцент (Уч. степень, уч. звание)	<hr/>	Назаренко Н. А.
		<i>подпись</i>	
Нормоконтроль	к.т.н., доцент (Уч. степень, уч. звание)	<hr/>	Егоров С. С.
		<i>подпись</i>	
Консультант	к.э.н., доцент (Уч. степень, уч. звание)	<hr/>	Жукова Т. Н.
		<i>подпись</i>	

Санкт-Петербург

2021

ЗАДАНИЕ НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ

Утверждаю

Зав. кафедрой ИС

_____ Цехановский В.В.

«___» _____ 20__ г.

Студент Уруков С.Д.

Группа 5374

Тема работы: Выявление Паттернов Функционирования Датчиков Для Диагностики Состояния Здания

Место выполнения ВКР: Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В. И. Ульянова (Ленина)

Исходные данные (технические требования): Требуется разработать инструмент, который позволил бы выявлять паттерны функционирования датчиков и автоматически аннотировать данные, с целью мониторинга показателей SCADA систем.

Содержание ВКР:

Анализ предметной области, разработка подхода по выявлению паттернов и аннотированию данных, реализация подходов и описание результатов, технико-экономическое обоснование.

Перечень отчетных материалов: пояснительная записка и презентация

Дополнительные разделы: Техничко-экономическое обоснование

Дата выдачи задания

«01» февраля 2021 г.

Дата представления ВКР к защите

«25» мая 2021 г.

Студент

Уруков С. Д.

Руководитель к.т.н, доцент
(Уч. степень, уч. звание)

Новикова Е. С.

КАЛЕНДАРНЫЙ ПЛАН ВЫПОЛНЕНИЯ ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ

Утверждаю

Зав. кафедрой ИС

_____ Цехановский В.В.

« ____ » _____ 20 ____ г.

Студент Уруков С.Д.

Группа 5374

Тема работы: Исследование паттернов поведения показателей сенсоров
внутри здания для диагностики состояния здания.

№ п/п	Наименование работ	Срок выполнения
1	Обзор литературы по теме работы	08.02 – 22.02
2	Сбор, преобразование и очистка данных	22.02 – 08.03
3	Исследование данных, применение методов интеллектуального анализа данных	08.03 – 22.03
4	Составление бизнес-плана по коммерциализации результатов НИР магистранта	22.03 – 05.04
5	Оформление пояснительной записки	05.04 – 19.04
6	Оформление иллюстративного материала	19.04 – 03.05

Студент _____

Уруков С. Д.

Руководитель к.т.н. доцент
(Уч. степень, уч. звание)

Новикова Е. С.

РЕФЕРАТ

Пояснительная записка 91 стр., 26 рис., 27 табл., 7 ист., 1 прил.

КЛЮЧЕВЫЕ СЛОВА И СЛОВСОЧЕТАНИЯ: БОЛЬШИЕ ДАННЫЕ, АНАЛИТИКА, МОНИТОРИНГ, JSON, PYTHON, ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ, СЕРВЕР, ВРЕМЕННОЙ РЯД, NUMPY, PANDAS, JUPYTER LAB.

Предметом работы является набор пакетных программ и модулей, реализованных на языке программирования Python, которые упростят процесс мониторинга данных в информационной системе автоматизированных зданий и позволят контролировать состояние здания, на основе больших данных.

Цель: в рамках выполнения задания выпускной квалификационной работы необходимо исследовать принципы хранения time-series данных, получаемых от контроллеров, обработать большие данные, выявить паттерны поведения показателей температур.

Содержание: для достижения цели была изучена документация следующих программных продуктов:

- математических пакетов NumPy, pandas, SciPy, scikit;
- облачной инфраструктуры Yandex Cloud;
- интерактивной IDE для математических расчетов JupyterLab.

По итогам изучения был разработан метод, позволяющий выявлять проблемы в показаниях сенсоров на основе построенных паттернов поведения данных.

ABSTRACT

The subject of the work is a set of package programs and modules, implemented in the Python programming language, which will simplify the process of monitoring data in an automated building system and allow you to monitor the state of a building based on large dimensions.

Purpose: as part of the assignment of the final qualifying work, it is necessary to investigate the basics of storing time series of data received from controllers, process big data, and identify patterns of behavior of temperature indicators.

Content: to achieve the goal, the documentation of the following software products was studied:

- mathematical packages NumPy, pandas, SciPy, scikit;
- cloud infrastructure Yandex Cloud;
- interactive IDE for mathematical calculations JupyterLab.

Based on the results of the study, a method was developed that allows one to identify problems in the readings of sensors based on the constructed patterns of data behavior.

СОДЕРЖАНИЕ

Определения, обозначения и сокращения	7
Введение	8
1. Анализ предметной области	9
1.1. Экономия электричества	10
1.2. Архитектура хранилища данных	11
1.3. Проблема Больших Данных	13
1.4. Алгоритм РСА	13
1.5. Триангуляция Делоне	15
1.6. Используемые библиотеки и математические пакеты	18
2. Разработка подхода по выявлению паттернов и аннотированию данных ..	20
2.1. Описание исследуемого здания	20
2.2. Описание подхода по построению паттернов	21
2.3. Геометрический смысл	21
2.4. Выявление паттернов	21
2.5. Аннотирование паттернов	24
3. Реализация подхода и описание результатов	26
3.1. Получение данных InfluxDB	26
3.2. Данные	27
3.3. Преобразование и очистка данных	28
3.4. Хранение данных	30
3.5. Развертывание лаборатории в облачной инфраструктуре	30
3.6. Расчет декомпозиции РСА	33
3.7. Расчет площадей фигур, сформированных триангуляцией Делоне	38
3.8. Автоматическое аннотирование данных	45
4. Техничко-экономическое обоснование	53
4.1. Описание концепции проекта	53
4.2. Описание продукции (результатов исследований)	54
4.3. Анализ рынка сбыта продукции	55
4.4. Анализ конкурентов	59
4.5. План маркетинга	63
4.6. Коммуникационная политика	64
4.7. Ценовая политика	65
4.8. План Продаж	66
4.9. План производства	66
4.10. Организационный план	71
4.11. Финансовый план	74
4.12. Стратегия финансирования	77
4.13. Оценка риска проекта	77
Заключение	80
Список использованных источников	81
ПРИЛОЖЕНИЕ А	82

ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

Ndarray — это многомерный контейнер (обычно фиксированного размера) для элементов одного типа и размера. Количество измерений и элементов в массиве определяется его формой, которая представляет собой кортеж из N неотрицательных целых чисел, определяющих размеры каждого измерения. Тип элементов в массиве определяется отдельным объектом типа данных (dtype), один из которых связан с каждым ndarray.

Серия — Одномерный ndarray с метками осей (включая временные ряды).

Датафрейм — Двумерные потенциально неоднородные табличные данные с изменяемым размером. Структура данных также содержит помеченные оси (строки и столбцы). Арифметические операции выравниваются по меткам строк и столбцов. Может рассматриваться как контейнер типа dict для объектов Series. Основная структура данных pandas.

ВВЕДЕНИЕ

Человечество достигло небывалых высот в области компьютерных наук. За последнее десятилетие появилось много профессий, которые решают задачи по сбору, анализу и исследованию больших данных. Есть профессия Data Scientist – люди, которые занимают эту должность, занимаются исследованием данных и построением гипотез. Выявление паттернов в анализе данных часто используемая методика. Благодаря ней работают системы распознавания лица, автоматическое исправление в клавиатуре при наборе текста.

Большой поток данных сложно контролировать строгим алгоритмом, потому что в действительности за слегка изменяющимися показателями датчиков может скрываться закономерность, едва уловимая детерминированными алгоритмами.

В настоящей работе целью является исследование набора данных показателей сенсоров здания, поступающих с датчиков и контроллеров, для дальнейшего анализа и контролю больших данных в режиме реального времени.

Данная разработка в дальнейшем способствует автоматизации масштабирования системы, повышению отказоустойчивости и контролю качества модулей, устанавливаемых в зданиях.

В первом разделе приводится анализ деятельности компании K3D и проводится обзор предметной области.

Во втором разделе описывается подход к исследованию задач по построению паттернов и автоматическому аннотированию. Выполняется обзор литературы и способы применения исследуемых методов.

Третий раздел посвящен реализации подхода и исследованию полученных результатов, сопоставлению результатов с исходным набором данных.

В четвертом разделе производится оценка исследуемого решения с точки зрения экономической составляющей. Рассчитывается и проводится анализ технико-экономического обоснования работы.

1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

К3D стремится внедрять решения для всего жилого дома с целью экономии энергоресурсов. Решение - комплексный аудит здания с последующим оснащением компьютеризированными модулями. В частности, например, при аудите учитываются следующие факторы: скорость ветра, температура наружного воздуха, температура внутри, положение здания относительно солнца, движение воздушных масс по коридорам и лестницам, температура в квартире. В здании установлен программируемый логический контроллер. Программное обеспечение контроллера управляет всеми обогревателями в здании, самостоятельно принимает решение о включении отопления на основании данных о состоянии системы и способно обогреть любую часть здания по запросу. Эта система позволяет владельцу здания сэкономить от 28% до 45% на счетах за отопление.

Сердцем HCS является центральная цифровая система управления, обеспечивающая контроль и мониторинг климата. Система полностью настраивается и позволяет установить желаемый температурный диапазон для каждой квартиры и управлять отоплением каждой секции дома индивидуально. HCS отлично подходит для сокращения неэффективного использования тепла, когда встроенные термостаты настроены на высокую температуру, а окна необходимо открывать, чтобы предотвратить перегрев. На рис. 1.1 представлена трехмерная модель здания, оборудованного системой ЖКХ.

Если жилец решает оставить окна или балкон открытыми, то он вынужден мириться с низкой температурой и неважно, как на термостате выставлен обогрев. Если он хочет открыть окно, то он должен учитывать, что могут быть потери тепла.



Рис. 1.1 – 3D-модель автоматизированного здания.

1.1. Экономия электричества

На рис. 1.2 показан результат анализа накопленных энергоресурсов в здании по адресу 3170 Donnelly St. в Виндзоре, провинция Онтарио. Система была установлена летом 2008 года, протестирована во время отопительного сезона 2008-2009 годов и запущена в производство 1 июня 2009 года. Данные, использованные для анализа, основаны на счетах за электроэнергию, полученных от местных служб. Производительность системы оценивается в сравнении с потреблением энергии за период 2009-2010 гг. И 2010-2011 гг. С пятью предыдущими сезонами.

Предлагаемая система предъявляет очень высокие требования к качеству, и для их выполнения необходимо обеспечить надежное хранение и бережную обработку данных.

Далее будет описан процесс сбора данных и схема развертывания системы.

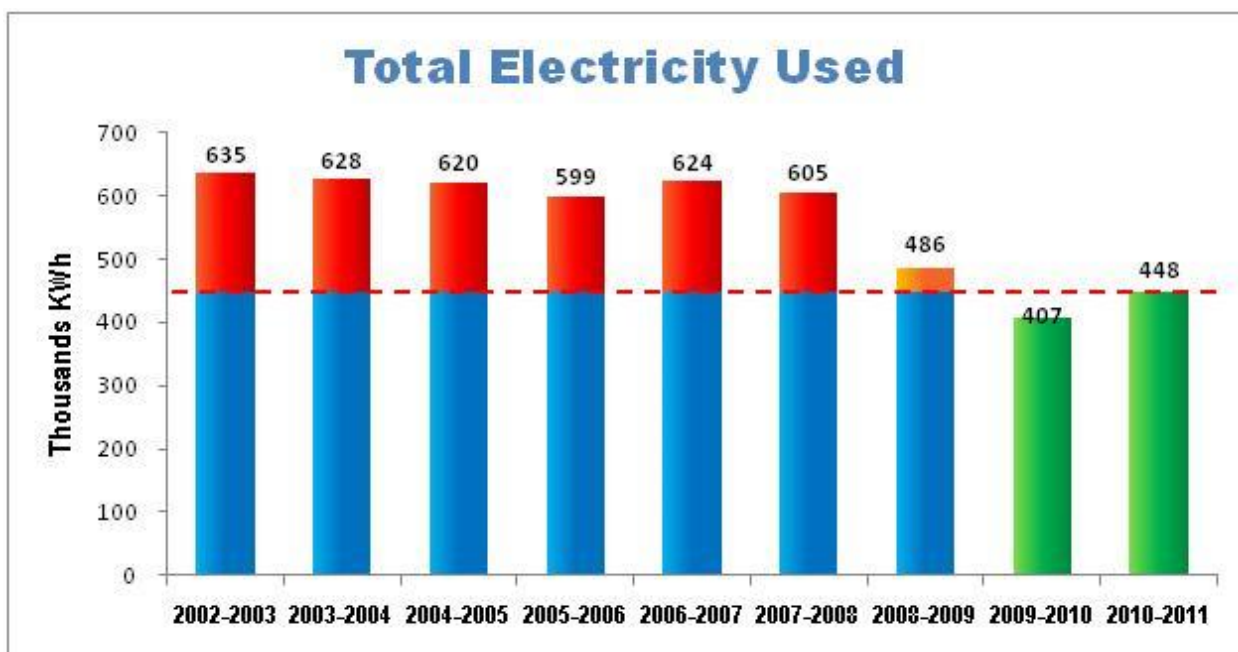


Рис. 1.2 – Гистограмма использования электричества в здании.

1.2. Архитектура хранилища данных

Информация от датчиков собирается на ПЛК и впоследствии обрабатывается сервером OPC, где впоследствии полученные данные сохраняются в InfluxDB.

Причина выбора InfluxDB связана с тем, что, в отличие от классической реляционной структуры, которая обычно используется при описании предметной области, такая база данных имеет свойства, которые обеспечивают лучшую производительность и более оптимальное хранение большого потока данных. Данные, собранные с контроллеров, имеют специфические особенности: приоритет операции записи, данные привязаны к метке времени. InfluxDB распространяется под лицензией с открытым исходным кодом. База данных написана на языке Go и оптимизирована для быстрого использования. Он отлично подходит для хранения данных о работе программного обеспечения, данных датчиков, относящихся к категории IoT (Интернет вещей), для сбора статистики логов и трассировок, данных бизнес-процессов.

Компания InfluxData, помимо разработки вышеупомянутой базы данных, предлагает еще три продукта для ИТ-рынка: Chronograf, Telegraf и Kapacitor.

Chronograf - это веб-приложение, используемое для визуализации и мониторинга данных. Набор инструментов веб-приложений позволяет создавать гибкие веб-страницы, состоящие из компонентов (графиков, диаграмм, индикаторов и т. Д.). Хорошее решение для визуализации данных и аналитики, но причины выбора подобной системы Grafana будут указаны ниже.

Если у компании есть потребность в сборе данных об использовании серверных ресурсов, то, несомненно, одним из хороших решений будет установка программного обеспечения Telegraf. Программа работает в системе как демон, работает в фоновом режиме и регулярно записывает данные о загрузке системы в InfluxDB.

Наверняка рано или поздно возникнет ситуация, требующая реакции на значения полученных данных. Karasitor справляется с этой задачей. Программа, которую можно поставить на сервер, подключить к базе данных InfluxDB, а затем повесить триггеры на возникающие события. Однако, в K3D данные собираются не только в InfluxDB, но еще и в Prometheus. Prometheus обладает рядом сильных качеств:

- многомерная модель данных;
- гибкий язык запросов;
- не полагается на распределенное хранилище; каждый узел-сервер автономный;
- данные могут поступать по HTTP протоколу;
- специализированные инструменты экспорта в другие сервисы;
- встроенный менеджер «тревожных» ситуаций, которые требуют внимания;

Prometheus был выбран как временное решение для сбора данных по протоколу MQTT с микроконтроллеров. Данные из MQTT собираются на Raspberry Pi и впоследствии записываются в эту базу данных.

Выбранные базы данных успешно справляются с задачей хранения информации о состоянии всех датчиков из разных зданий, однако другая база данных, MSSQL, в подзаголовке не упоминалась.

MSSQL используется для хранения метаданных о зданиях, подключенных к информационной системе компании. Использование реляционной модели данных позволяет восполнить недостатки баз данных временных рядов. В частности, в нем хранятся данные о строительных конструкциях и их элементах.

1.3. Проблема Больших Данных

Компания сталкивается с серьезной проблемой в управлении большими данными с разных датчиков. Это требуется не только для обеспечения бесперебойной работы термостатической системы, но и для распознавания нестандартных ситуаций до их возникновения.

В работу закладывается идея разработки программного продукта, который бы позволил в режиме реального времени выявлять закономерности в поведении данных и автоматическом аннотировании. Это позволит бизнесу развиваться шире, обеспечивая качественный сервис и бесперебойную работу.

К рассмотрению предлагается использовать популярные инструменты и алгоритмы обработки больших данных, которые в последствии в совокупном использовании покажут результат, который в конечном счете можно будет использовать как готовый программный продукт в виде библиотеки, веб-сервиса или плагина для программ по аналитике данных.

Анализ данных хорошо себя показывает на практике. Его ключевая особенность в том, что он хорошо подстраивается под любую предметную область. Например, с помощью метода главных компонент было произведено исследование в определении аномалий в беспроводных ячеистых сетях [1].

Далее будут описаны основные инструменты и библиотеки, которые используются в настоящей работе.

1.4. Алгоритм PCA

Метод главных компонент — это технология многомерного статистического анализа, используемая для сокращения размерности пространства признаков с минимальной потерей полезной информации. Предложен К.

Пирсоном в 1901 г., а затем детально разработан американским экономистом и статистиком Г. Хоттелингом [2].

С математической точки зрения метод главных компонент представляет собой ортогональное линейное преобразование, которое отображает данные из исходного пространства признаков в новое пространство меньшей размерности.

При этом первая ось новой системы координат строится таким образом, чтобы дисперсия данных вдоль неё была бы максимальна. Вторая ось строится ортогонально первой так, чтобы дисперсия данных вдоль неё, была бы максимальной из оставшихся возможных и т.д. Первая ось называется первой главной компонентой, вторая — второй и т.д.

Идея подхода заключается в том, что с каждой главной компонентой связана конкретная доля общей дисперсии исходного набора данных (иными словами нагрузка). Дисперсия, в свою очередь, может отражать уровень их информативности и является мерой изменчивости данных.

Метод главных компонент решает задачу построения нового пространства признаков меньшей размерности, дисперсия между осями которой будет перераспределена так, чтобы максимизировать дисперсию по каждой из них. Для этого выполняется по порядку следующие действия:

- Вычисляется общая дисперсия исходного пространства признаков. Это нельзя сделать простым суммированием дисперсий по каждой переменной, поскольку они, в большинстве случаев, не являются независимыми. Поэтому суммировать нужно взаимные дисперсии переменных, которые определяются из ковариационной матрицы.
- Вычисляются собственные векторы и собственные значения ковариационной матрицы, определяющие направления главных компонент и величину связанной с ними дисперсии.

Выполняется снижение размерности. Диагональные элементы ковариационной матрицы показывают дисперсию по исходной системе координат, а её собственные значения — по новой. Тогда разделив дисперсию, связанную

с каждой главной компонентой на сумму дисперсий по всем компонентам, получаем долю дисперсии, связанную с каждой компонентой. После этого отбрасывается столько главных компонент, чтобы доля оставшихся составляла 80-90%.

Алгоритм PCA очень часто находит свое применение в задачах по кластеризации и снижению размерности многомерных разнородных данных.

1.5. Триангуляция Делоне

Если некоторые пары точек соединены ребром, любая конечная грань в получившемся графе образует треугольник, ребра не пересекаются, и граф максимален по количеству ребер, то такое множество точек задают триангуляцией.

Триангуляцией Делоне называется такая триангуляция, в которой для любого треугольника верно, что внутри описанной около него окружности не находится точек из исходного множества.

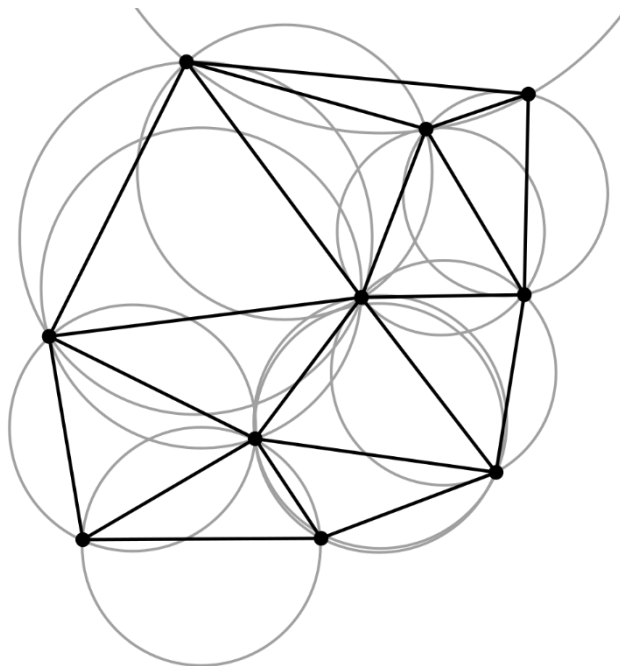


Рис. 1.3 – Пример триангуляции Делоне.

Условие Делоне

Пусть на множестве точек задана триангуляция. Будем говорить, что некоторое подмножество точек удовлетворяет условию Делоне, если

триангуляция, ограниченная на это подмножество, является триангуляцией Делоне для него [3].

Свойства триангуляции Делоне

Триангуляция Делоне часто используется на практике и представляет собой особый вид триангуляции, отвечающий следующим свойствам:

- Триангуляция Делоне уникальна, в триангуляции Делоне не будет других точек в пределах описанной окружности любого треугольника.
- Треугольник образован тремя ближайшими точками, и каждый отрезок линии не пересекается.
- Независимо от того, где начинается область, конечный результат будет стабильным.
- Если диагонали выпуклого четырехугольника, образованного любыми двумя соседними треугольниками, взаимозаменяемы, то наименьший угол среди шести внутренних углов двух треугольников не станет больше.
- если наименьший угол каждого треугольника в триангуляции расположен в порядке возрастания, расположение триангуляции Делоне получит наибольшее значение.
- Добавление, удаление или перемещение вершины повлияет только на соседний треугольник.
- Самая внешняя граница треугольной сетки образует выпуклую многоугольную оболочку.

Применение триангуляции в задачах анализа данных

Построение триангуляции хорошо проявляет себя в задачах по поиску паттернов поведения некоторых данных.

В статье «Combined Approach to Anomaly Detection in Wireless Sensor Networks on Example of Water Management System»[5] представлен подход к обнаружению аномалий в беспроводных сенсорных сетях (WSN). Он основан

на сочетании методов машинного обучения и визуального анализа данных. Подход был протестирован на примере системы управления водными ресурсами, представленной программно-аппаратным прототипом. Результаты экспериментов показали высокие значения показателей качества обнаружения аномалий, что свидетельствует о целесообразности применения на практике комбинированного подхода к обнаружению аномалий для беспроводных сенсорных сетей.

В главе книги «Digital Terrain Analysis in Soil Science and Geology»[6] объясняется топография почвенной системы. В ней также демонстрируются возможности анализа, моделирования и прогнозного картирования пространственного распределения свойств почвы на основе триангуляции Делоне с использованием следующих подходов: временная изменчивость влияния топографии на динамические свойства почвы, изменения. Корректность моделирования на основе триангуляции Делоне и прогнозного картирования свойств почвы зависит не только от этой изменчивости. Эта изменчивость подразделяется на два метода: временная изменчивость и изменчивость по глубине. Глава также исследует изменчивость во взаимосвязях между почвой и морфометрическими переменными.

В статье китайских ученых «The Application of Delaunay Triangulation to Face Recognition»[6] предлагается новый подход, который отличается от традиционных методов распознавания лиц разделением изображения лица на треугольные характерные области. Триангуляция Делоне, основанная на барицентрах, заменяет треугольники последовательно меньшими треугольниками до тех пор, пока не будет удовлетворен критерий остановки. Процесс слияния объединяет смежные треугольники схожих средних и отклонений. Гистограмма последнего набора областей используется в качестве дискриминанта для идентификации лиц. Результат использования этого метода триангуляции Делоне инвариантен к различиям в масштабе и ориентации тестируемого изображения лица и устойчив к помехам.

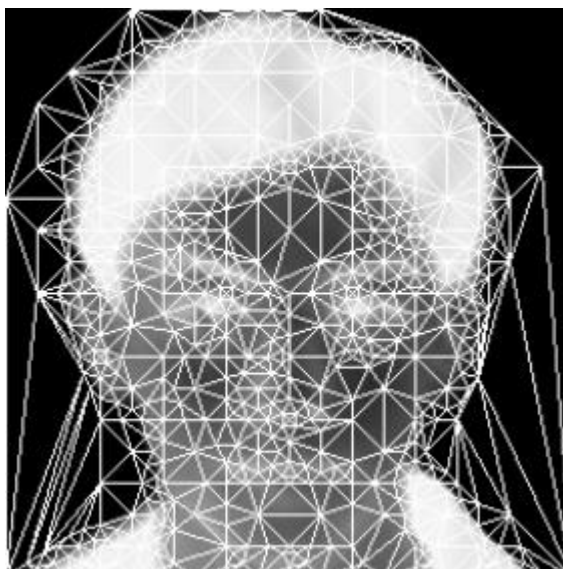


Рис. 1.4 – Фрагмент из статьи по распознаванию лиц.

Таким образом, рассмотренные материалы показывают, что триангуляция Делоне хорошо проявляет в себя в задачах по выявлению паттернов.

Роль визуализации в построении триангуляции Делоне

Визуализация полученных результатов способствует повышению восприятия данных. Интерпретация больших массивов данных в виде графиков позволит лучше понимать, как работает «живет» система контроля отопления в здании.

1.6. Используемые библиотеки и математические пакеты

Исследовательская работа с практической точки зрения включает в себя программный продукт, написанный на языке программирования Python. Этот язык очень популярен среди программистов и программисты Python пользуются большим спросом на рынке труда. Python позиционируется как язык общего назначения, однако имеет обширное сообщество в анализе данных. Специалисты выбирают Python, потому что с ним просто работать, имеется большое количество библиотек и модулей, которые легко встраиваются в проект. Для анализа данных применяется ещё язык R. Несмотря на то, что в R так же много возможностей, как и в Python, его все же выбирают исключительно для расчетов, потому что собирать, преобразовывать и подготавливать данные на языке Python проще из-за его широкого применения.

Поэтому, в соответствии с выбором языка были выбраны следующие библиотеки для вычислений:

- Numpy и Pandas – для сбора, хранения и обработки данных в виде векторов и датафреймов.
- SciKit – для применения алгоритмов машинного обучения. Содержит в себе широкий спектр функций по задачам классификации, регрессионного анализа, кластеризации, уменьшения размерности. В работе используется алгоритм PCA из представленной библиотеки.
- SciPy – для применения пространственных алгоритмов. В работе используется функция, которая считает триангуляцию Делоне.

2. РАЗРАБОТКА ПОДХОДА ПО ВЫЯВЛЕНИЮ ПАТТЕРНОВ И АННОТИРОВАНИЮ ДАННЫХ

2.1. Описание исследуемого здания

В исследовании принимает участие здание, которое располагается в городе Гамильтон, провинция Онтарио, Канада.



Рис. 2.1 – Фотография здания, данные которого подвергнуты исследованию.

В здании располагаются 375 квартир на 23 этажах. Отопление внутри здания поделено на зоны. Каждая зоны может содержать от 6 до 36 квартир.

Отопительный сезон в здании начинается с середины сентября и заканчивается в апреле. Показатели датчиков собираются круглогодично, однако владельцы здания выключают систему управления нагревом в летний период времени. Это незначительно отражается на данных в исследовании.

2.2. Описание подхода по построению паттернов

Исходными данными для исследования являются температурные показания, собранные по времени. Каждый момент времени характеризуется многомерным вектором с температурами.

Для категоризации расчетов и уменьшения времени выполнения расчетов данные были разделены по зонам. И каждые вычисления производились только по нескольким зонам. Это ускорило время получения результата.

Для каждой зоны многомерные данные были приведены к двухкомпонентному виду по алгоритму PCA. Получилось множество точек, характеризующее вектор данных о температуре в момент времени в виде точки на двумерной плоскости.

Полученный массив данных по точкам в дальнейшем был поделен по временным промежуткам. На каждую группу точек был применен алгоритм триангуляции Делоне. Итоговым результатом данной процедуры стала разметка триплетов, каждый содержит информацию о трех точках, образующих треугольник.

Впоследствии рассчитывается площадь всех треугольников. Так в итоге получается фигура, у которой площадь является параметром, характеризующим зону в определенный промежуток времени.

2.3. Геометрический смысл

Взаимное расположение точек на плоскости PCA говорит о схожести данных. Если в данных имеются выбросы или данные неоднородные, то это скажется на их расположении на координатной плоскости, что в свою очередь повлияет на площадь фигуры, образованной треугольниками триангуляции Делоне.

2.4. Выявление паттернов

Предполагается, что паттерном в работе будет принята совокупность результирующих последовательных площадей фигур, построенных на триангуляции Делоне.

Чтобы сравнить статистические данные, будет использован метод расчета индекса структурного сходства.

Измерение индекса структурного сходства (SSIM) - это метод прогнозирования воспринимаемого качества цифрового телевидения и кинематографических изображений, а также других видов цифровых изображений и видео. SSIM используется для измерения сходства между двумя изображениями. Индекс SSIM — это полная справочная метрика; другими словами, измерение или прогноз качества изображения основывается на исходном несжатом или свободном от искажений изображении в качестве эталона.

SSIM - это модель, основанная на восприятии, которая рассматривает деградацию изображения как воспринимаемое изменение структурной информации, а также включает важные явления восприятия, включая как маскирование яркости, так и условия маскирования контраста. Отличие от других методов, таких как MSE или PSNR, заключается в том, что эти подходы оценивают абсолютные ошибки. Структурная информация — это идея о том, что пиксели сильно взаимозависимы, особенно когда они пространственно близки. Эти зависимости несут важную информацию о структуре объектов визуальной сцены. Маскирование яркости — это явление, при котором искажения изображения (в данном контексте) имеют тенденцию быть менее заметными в ярких областях, в то время как маскирование контраста — это явление, при котором искажения становятся менее заметными там, где есть значительная активность или «текстура» в изображении.

На рис. 2.2 представлено сравнение SSIM индекса по различным изображениям

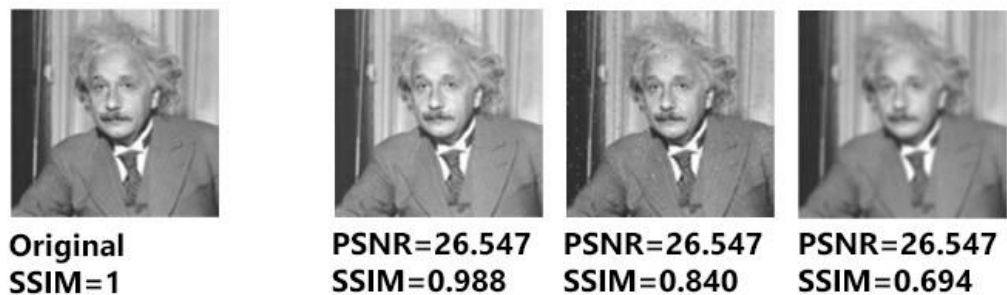


Рис. 2.2 – Изображение и его SSIM индексы.

SSIM был впервые представлен в документе IEEE 2004 г. «Оценка качества изображения: от видимости ошибок к структурному подобию». Реферат дает хорошее представление об идее предлагаемой системы.

Объективные методы оценки качества воспринимаемого изображения традиционно пытались количественно оценить видимость ошибок (различий) между искаженным изображением и эталонным изображением с использованием множества известных свойств зрительной системы человека. Исходя из предположения, что зрительное восприятие человека хорошо приспособлено для извлечения структурной информации из сцены, мы вводим альтернативную дополнительную структуру для оценки качества, основанную на деградации структурной информации.

Авторы отмечают 2 важных момента:

Большинство методов оценки качества изображения основаны на количественной оценке ошибок между эталонным и образцом изображения. Общий показатель — это количественная оценка разницы в значениях каждого из соответствующих пикселей между образцом и эталонными изображениями (с использованием, например, среднеквадратичной ошибки).

Система визуального восприятия человека обладает высокой способностью идентифицировать структурную информацию из сцены и, следовательно, определять различия между информацией, извлеченной из эталонной и эталонной сцены. Следовательно, метрика, которая воспроизводит это

поведение, будет лучше работать с задачами, которые включают различие между образцом и эталонным изображением.

Алгоритмы расчета индекса SSIM очень хорошо срабатывают на векторных цифровых данных, что дает возможность использования не только в сфере компьютерного зрения и анализа изображений, но и в построении и выявлении паттернов.

2.5. Аннотирование паттернов

В работе рассматриваемый метод поиска паттернов позволяет запрограммировать автоматическую аннотацию данных. В 3 главе подробно описывается выявление паттерна и его повторное применение.

Выполняется проверка данных, которые подходят по этим паттернам – приводится статистический анализ и график «ящик с усами». Сравниваются показатели температур в совокупности для каждой квартиры, их минимальные и максимальные значения. Ставится задача определения жизнеспособности метода. Если выбранный паттерн найдет методом скользящего окна похожие паттерны, и похожие паттерны окажутся сравнимы по статистическим показателям, то метод, исследуемый в настоящей выпускной квалификационной работе, окажется оправданным и сможет использоваться для выявления сбоящих датчиков и внештатных ситуаций.

Метод скользящего окна — алгоритм трансформации, позволяющий сформировать из членов временного ряда набор данных, который может служить обучающим множеством для построения модели прогнозирования.

Под окном в данном случае понимается временной интервал, содержащий набор значений, которые используются для формирования обучающего примера. В процессе работы алгоритма окно смещается по временной последовательности на единицу наблюдения, и каждое положение окна образует один пример.

Например, если еженедельно поступают данные о продажах в течение 50 недель и установлено окно в 5 недель, то в первом примере используются данные с 1 по 5 неделю, а целевым значением будут данные за 6-ю неделю. Во

втором случае используются данные со 2 по 6 неделю, а в качестве целевого значения берутся данные за 7-ю и так далее.

Если требуется построить прогноз не на одну единицу наблюдения, а на несколько, то в качестве целевых выбирается соответствующее число значений, которое называется горизонтом прогноза. Количество наблюдений ряда, которые берутся в качестве входных значений, называется глубиной прогноза.

3. РЕАЛИЗАЦИЯ ПОДХОДА И ОПИСАНИЕ РЕЗУЛЬТАТОВ

3.1. Получение данных InfluxDB

Взаимодействовать с данными, расположенными на InfluxDB узле, не удобно, потому что сбор, преобразование и очистка данных подразумевают под собой выполнение множественных дисковых операций. Сервер располагается географически очень далеко от места проведения исследований, поэтому возникла потребность в загрузке данных с узла на локальную машину.

Было принято решение выгрузить данные с базы данных InfluxDB в формат, удобный для использования на локальной машине. Для этого был написан скрипт на языке программирования Python, который подключался к базе данных, отправлял запрос на получение данных за период равный двум последним годам и экспортировал данные в текстовый формат CSV. Файлы CSV представляют собой дампы базы данных, содержат две колонки *time* и *values*. Время представлено в виде строки, формата ISO, содержащей в себе информацию о дате, времени и часовом поясе. Value – значение, представлено в виде числа с плавающей точкой, либо целого числа. В таблице XX представлен пример данных, которые содержатся в типовом CSV файле дампа измерения базы данных.

Таблица 3.1 – Пример исходных данных.

	time	value
1	2018-12-31T21:00:22.762852391Z	23.7999992
2	2018-12-31T21:02:45.905678882Z	23.7000008
3	2018-12-31T21:04:46.743163102Z	23.7000008
4	2018-12-31T21:09:39.793465629Z	23.7000008
5	2018-12-31T21:10:07.983209692Z	23.7000008
6	2018-12-31T21:13:15.766897119Z	23.6000004
7	2018-12-31T21:15:14.269799795Z	23.6000004

Произведена загрузка данных трех типовых зданий, которые географически располагаются в разных городах.

3.2. Данные

Компания оснащает дом системой управления теплом (Heating Management System). Она представляет собой совокупность аппаратных средств и программного обеспечения для него. В здание устанавливается единая панель управления, с которой связаны различные сенсоры. Каждый элемент системы, который посылает данные сохраняет в регистры программируемого логического контроллера данные о какой-либо измерении.

Собираемые данные впоследствии обрабатываются орс-сервером, который в свою очередь считывает данные с регистров plc и по защищенному каналу vpn передает данные в базу данных InfluxDB. Орс-сервер публикует значения на регистрах с разным интервалом. Интервал устанавливается специалистом, который генерирует конфигурационный файл здания. Существует 4 класса частоты сканирования: immediate (100 мс), fast (1000 мс), slow (120 с), default (10 с). Большинство измерений сканируются с частотой 10 секунд. Если значение параметра не изменилось, данные в базу данных не сохраняются.

Система управления теплом собирает с сенсоров здания следующие параметры:

- Температура юнита (градусы Цельсия);
- Установленная температура юнита (градусы Цельсия);
- Температура здания (градусы Цельсия);
- Установленная температура здания (градусы Цельсия);
- Температура снаружи здания (градусы Цельсия);
- Фактор холода (градусы Цельсия) (функция от скорости ветра и внешней температуры);
- Скорость ветра (км/ч);
- Направление ветра (целочисленное значение);
- Нагрев зоны (0-100);
- Температура зоны (градусы Цельсия);
- Нагрев квартиры (0-100);

В базе данных здания содержатся сведения о количестве этажей, количестве квартир на этаже и о принадлежности квартир к зонам. Эти данные содержатся в иерархическом виде в формате json-файла.

Перечисленные параметры содержат численное значение, округленное до 1 знака.

Данные по каждому зданию сохранены в специальную папку, при вызове команды «du -sh» из директории с исходными данными, в консоль выдается следующий результат:

2.1G buckley

2.7M json

1.3G paris

3.4G park

1.5G portage

Каждая папка содержит в себе около 559 csv файлов.

3.3. Преобразование и очистка данных

Каждый показатель температуры изменяет свое значение примерно 1 раз в 3 минуты. В базе данных каждая запись значения сохраняется с точностью до миллисекунды. Но ввиду сложности обработки данных временные показатели округляются до одной минуты. Таким образом, например, события с разницей в 20 секунд считаются событиями, произошедшими одновременно.

Данные могут содержать ошибки. Ошибка обычно представлена как неадекватное значение для показателя. Например, встречаются температурные значения равные 800.1°C.

Строковое значение даты и времени преобразуется в timestamp объект. Объект сохраняет часовой пояс и контролирует «daily savings time» - переход на летнее время. Летнее время — это практика перевода часов (обычно на один час). в теплые месяцы, чтобы темнота наступила позже. Типичная реализация DST — это перевод часов на один час вперед весной («весна вперед») и перевод часов на один час осенью («возврат назад»), чтобы вернуться к

стандартному времени. В результате, в конце зимы или в начале весны есть один 23-часовой рабочий день и один 25-часовой день осенью.

Для учета времени используется параметр `nonexistent – shift_forward`. Несуществующее время не существует в конкретном часовом поясе, где часы переместились вперед из-за перехода на летнее время. Допустимые значения:

- «*shift_forward*» сдвинет несуществующее время вперед на ближайшее существующее время;
- «*shift_backward*» сдвинет несуществующее время назад к ближайшему существующему времени;
- «*NaT*» вернет `NaT`, если нет времени;
- объекты *timedelta* будут сдвигать несуществующие времена на *timedelta*;
- «*raise*» вызовет исключение `NonExistentTimeError`, если нет несуществующих значений времени.

Числа с плавающей точкой округлены до одного знака после запятой. В базе данных сохраняются и значения с большим количеством знаков, однако эти данные не соответствуют действительность – точность приборов, собирающих данные один знак.

Для объединения разных значений по времени в один датафрейм, поле *время* было задано как ключевое, и по ключевому полю соединялись данные через «внешнее соединение». Оно использует объединение ключей из обоих фреймов, аналогично полному внешнему соединению SQL; сортирует ключи лексикографически. После объединения остаются поля, которые содержат `NA` (неопределенные) значения. Эти значения заполняются по правилу `forward fill`. Последующее значение заполняется по значению предыдущего, если его значение не указано. Правило справедливо, потому что в OPC сервер посылает данные на сохранение только при изменении. Если значение параметра не изменилось во время очередного опроса, то оно не отправляется в базу данных.

3.4. Хранение данных

В таблице представлен срез данных, которые хранятся в датафрейме. Для исследования были выбраны температурные значения, разделенные по зонам. Здание отапливается по зонам. Зоны содержат в себе температуры разных квартир. В одной зоне могут быть квартиры с разных этажей.

Каждый датафрейм имеет размерность примерно 493045 строк \times 36 столбцов.

Таблица 3.2 – Фрагмент сводной таблицы

time	unit_1802	unit_1803	unit_1804	unit_1817	unit_1818	unit_1819	unit_1902	unit_1903	unit_1904	unit_1917
2019-01-01 00:00:00- 05:00	22.6	24.1	24.0	24.00	23.60	22.20	24.0	24.2	21.7	23.1
2019-01-01 00:03:00- 05:00	22.6	24.2	24.0	24.20	23.60	22.20	24.1	24.2	21.8	23.2
2019-01-01 00:11:00- 05:00	22.6	24.2	23.9	23.95	23.55	22.25	24.1	24.1	21.8	23.1
2019-01-01 00:15:00- 05:00	22.7	24.2	24.0	24.10	23.60	22.30	24.2	24.1	21.8	23.2
2019-01-01 00:17:00- 05:00	22.6	24.2	24.0	24.00	23.60	22.30	24.2	24.1	21.8	23.3

3.5. Развертывание лаборатории в облачной инфраструктуре

Выполнение исследовательской работы требует большие вычислительные ресурсы для комфортной работы. Для этого был арендована виртуальная машина в облачной инфраструктуре Yandex.Cloud.

Yandex.Cloud — это набор связанных сервисов, которые помогут вам быстро и безопасно взять в аренду вычислительные мощности в тех объемах,

в которых это необходимо. При этом доступ к вычислительным мощностям вы получаете через интернет. Такой подход к потреблению вычислительных ресурсов называется облачные вычисления.

Yandex.Cloud — это публичная облачная инфраструктурная платформа Яндекса, на которой клиенты могут создавать и развивать свои цифровые продукты и решения. Фокус платформы — это инструменты разработки, аналитики и управления данными, искусственного интеллекта и машинного обучения. В основе лежат уникальные и трудновоспроизводимые технологии, которые развернуты в трех собственных дата-центрах Яндекса.

Облачные вычисления заменяют и дополняют традиционные дата-центры, расположенные на территории потребителя. Yandex.Cloud берет на себя задачи по поддержанию работоспособности и производительности аппаратного и программного обеспечения облачной платформы.

Yandex.Cloud предлагает вам различные категории облачных ресурсов: например, виртуальные машины, диски, базы данных. Управлять ресурсами каждой категории можно с помощью соответствующего сервиса.

Инфраструктура Yandex.Cloud защищена в соответствии с Федеральным законом Российской Федерации «О персональных данных» № 152-ФЗ.

Платформа Yandex.Cloud на первом этапе размещается в трех дата-центрах Яндекса, расположенных во Владимирской, Рязанской и Московской областях. Инфраструктура Yandex.Cloud в каждом из дата-центров называется зоной доступности. Каждая зона изолирована от аппаратных и программных сбоев в других зонах доступности.

Yandex Data Science Virtual Machine (DSVM) — это виртуальная машина с предустановленными популярными библиотеками для анализа данных и машинного обучения. DSVM можно использовать как среду для обучения моделей и экспериментов с данными.

Предустановленное программное обеспечение

Операционная система: Ubuntu 18.04

Установленные пакеты:

- Менеджер пакетов conda с Python 2.7 и Python 3.6.
- Инструменты для интерактивных и воспроизводимых вычислений Jupyter Notebook и JupyterLab.
- Библиотеки Machine Learning:
- CatBoost;
- LightGBM;
- XGBoost;
- TensorFlow;
- PyTorch.
- Система управления контейнерами Docker.
- Консольные клиенты систем контроля версий: SVN, Git, Mercurial.
- Библиотеки NumPy, scikit-learn, SciPy, оптимизированные с помощью Intel Math Kernel Library и Data Analytics Acceleration Library.
- Оптимизированные библиотеки для работы с изображениями: libjpeg-turbo, Pillow-SIMD.

Для расчетов были выбраны следующие параметры виртуальной машины:

Таблица 3.3 – Конфигурация сервера.

Конфигурация сервера	
Инфраструктура	Yandex Cloud
Зона доступности	ru-central1-c
Платформа	Intel Cascade Lake
Гарантированная доля vCPU	50%
vCPU	2
RAM	4 Гб
Объём дискового пространства	32 Гб

3.6. Расчет декомпозиции РСА

Для каждой зоны была произведена декомпозиция многомерных РСА данных. Выполнение расчетов происходит в многопоточном режиме, результаты вычислений сохраняются в специальную структуру – датафрейм. Датафрейм содержит 3 поля – время, первая компонента РСА, вторая компонента РСА. Компоненты РСА рассматриваются как точки координатной плоскости. Так, если проиллюстрировать данные температур в декомпозированном виде, накопленные за 2 года на одном графике для зоны, то можно получить изображение как на рис. 3.1.

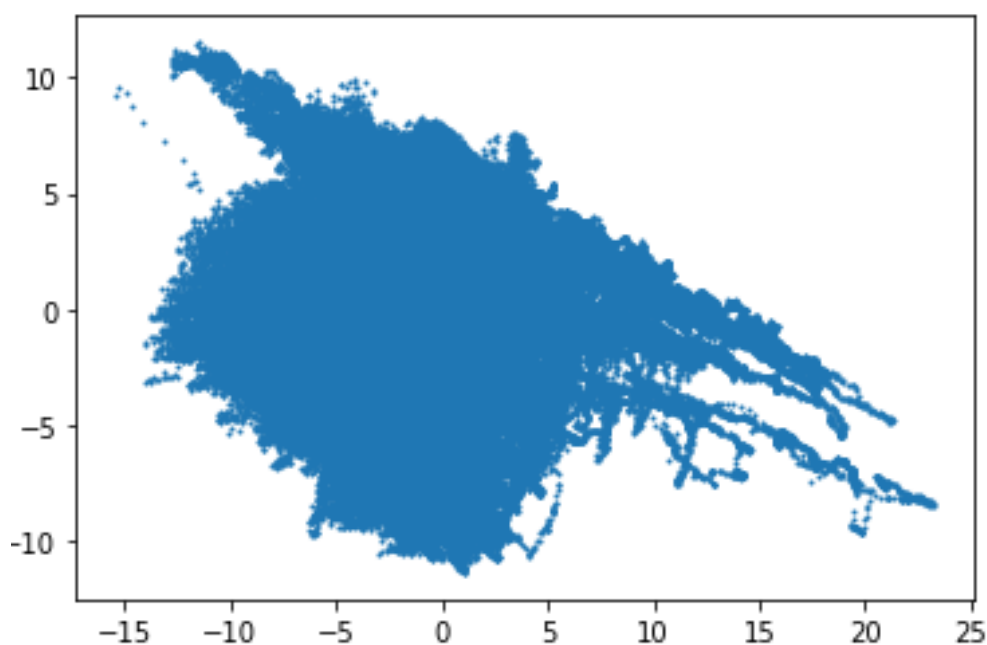


Рис. 3.1 – Визуальное представление температурных векторов на двумерной плоскости для зоны №22.

Эти данные визуализируют около полумиллиона точек на одном изображении, что в итоге принимает вид пятна. Это «пятно» для каждой зоны свое. Например, визуализация для другой зоны представлена на рис 3.2.

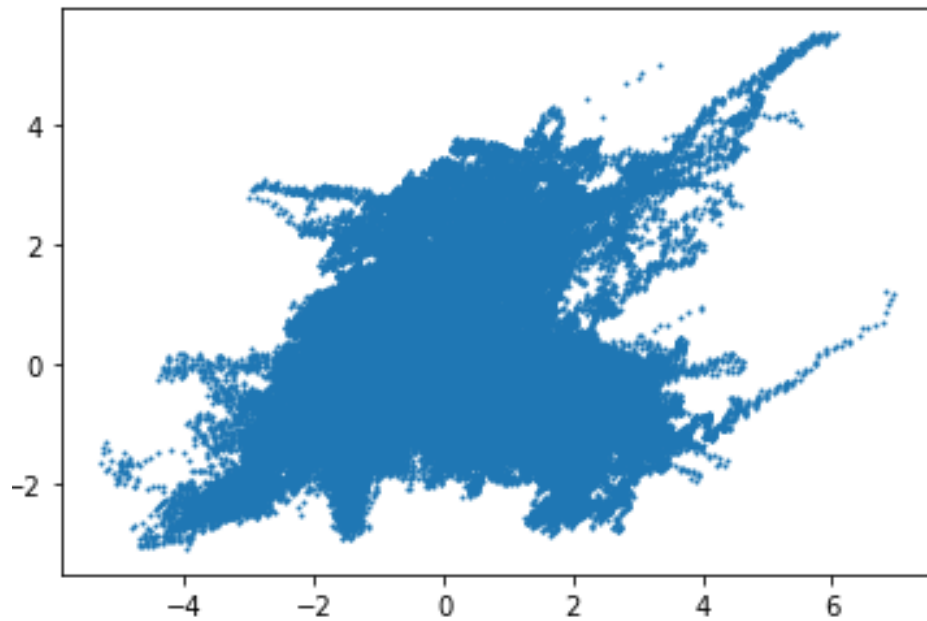


Рис. 3.2 – Визуальное представление температурных векторов на двумерной плоскости для зоны №11.

Промежуточные расчеты декомпозиции PCA кэшируются программой. Результаты сохраняются на жестком диске. В следующий раз, когда программа запустится, данные загрузятся из файлов. Это позволяет быстрее производить эксперименты и не ждать вычисления заново, когда программа перезапускается.

В ходе исследования кучности точек было замечено, что, если визуальное представление рассмотреть по месяцам, то будут наблюдаться фигуры совершенно разных форм.

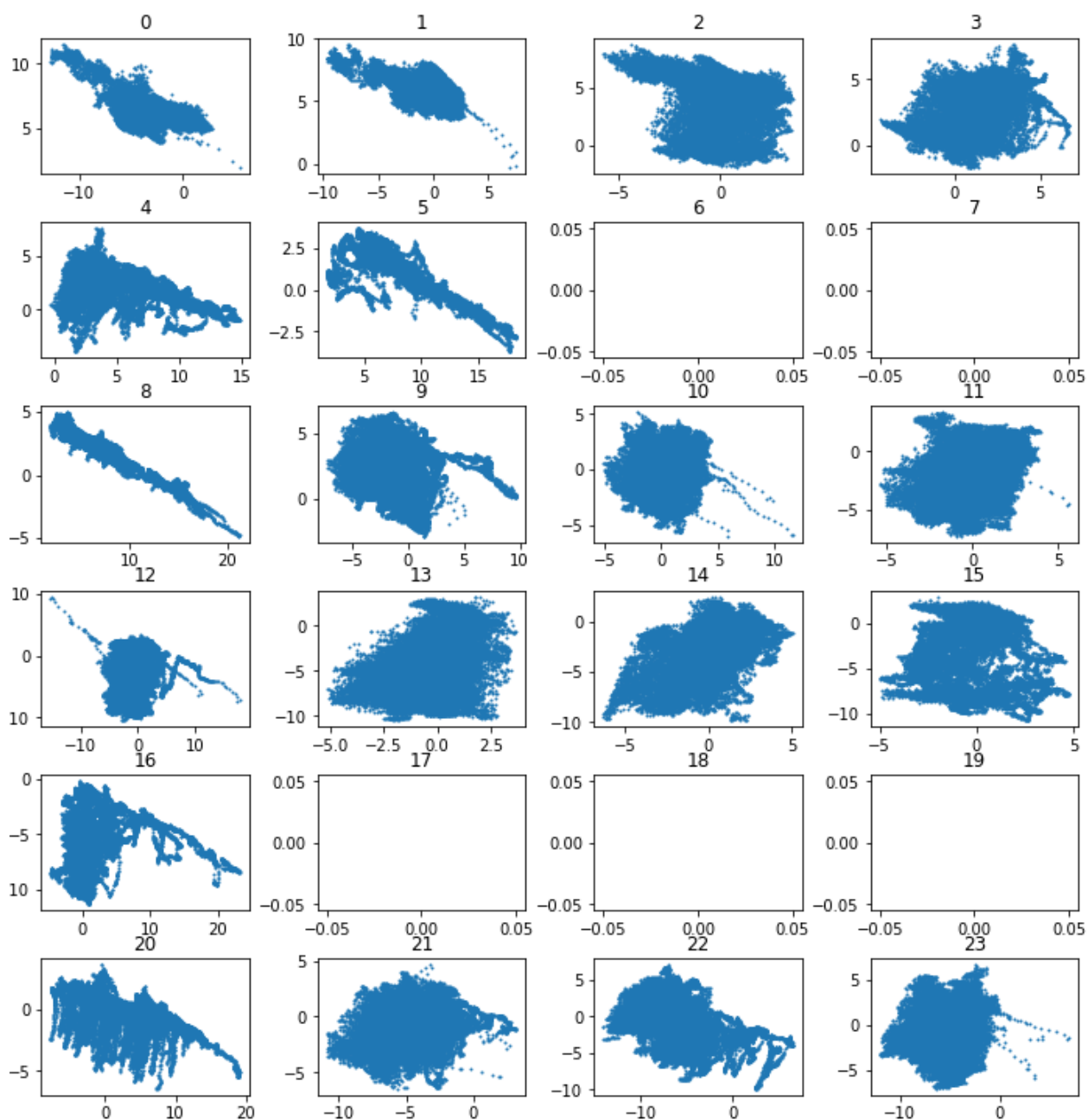


Рис. 3.3 – Визуализация PCA декомпозиции по месяцам.

В областях, где нет графика, система обогрева здания не работала. Нумерация месяцев на графике идет от 0 до 23, что соответствует номеру месяца в рамках заданной выборки. Наблюдается, что совокупность точек меняет свою форму и её силуэт совпадает с силуэтом того же месяца. Например, под номером 4 – диаграмма рассеяния температурных показателей за май 2019 года, а под номером 16 температурные показатели за май 2020 года. На рис. XX представлено сравнение двух графиков на одной диаграмме рассеяния. Стоит обратить внимание на то, что хотя графики и имеют разные размеры, пропорции очень хорошо сохраняются.

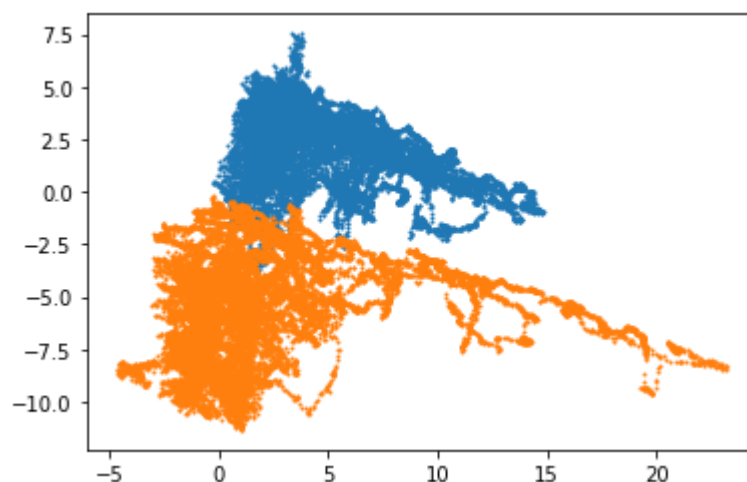


Рис. 3.4 – Сравнение двух майских PCA представлений на одном графике.

Затем было проведено такое же исследование, визуально отображающее другую зону здания. Выбрана зона №11. Так как расчеты декомпозиции PCA строились на другом наборе данных с другой размерностью, то и результаты получились не похожи.

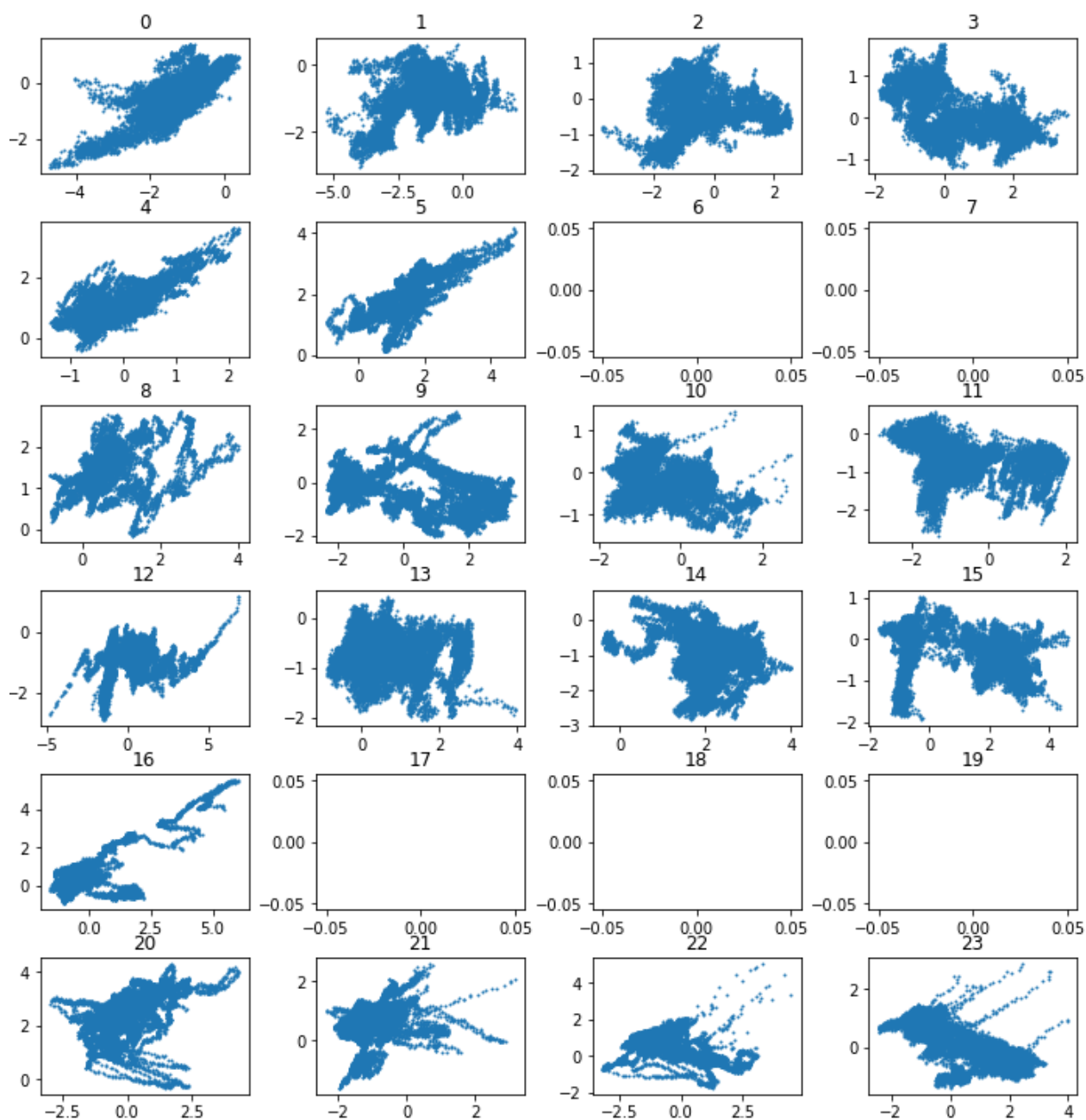


Рис. 3.5 – Зона №11 PCA декомпозиция.

На получившемся рисунке едва можно заметить похожие фигуры. Однако, проведенные испытания показали, что фигуры, которые получаются после декомпозиции PCA, существенно отличаются, что подводит к идее исследования триангуляции Делоне и площади получившихся фигур.

Далее были произведены расчеты площадей фигур по определенным периодам времени.

3.7. Расчет площадей фигур, сформированных триангуляцией Делоне

Идея, которая закладывается в смысл, заключается в том, чтобы сравнить и проверить насколько будет сильно различаться поведение температурных датчиков в перспективе времени и выявить, существуют ли варианты с одинаковыми паттернами.

Для расчета площади фигур по триангуляции Делоне были взяты РСА результаты, полученные из прошлых опытов. Каждый датасет зоны был поделен по месяцам и по нему строилась триангуляция. Вычисления заняли 12 часов, результаты расчетов сохранились на сервере в бинарный файл. Расчеты запускались на многопоточной основе ThreadPool.

Если вершины треугольника заданы, как точки в прямоугольной декартовой системе координат: $A_1(x_1, y_1)$, $A_2(x_2, y_2)$, $A_3(x_3, y_3)$, то площадь такого треугольника можно вычислить по формуле определителя второго порядка:

$$S = \pm \frac{1}{2} \begin{vmatrix} x_1 - x_3 & y_1 - y_3 \\ x_2 - x_3 & y_2 - y_3 \end{vmatrix}$$

В таблице 3.4 представлен результат вычисления результирующей площади фигуры по триангуляции Делоне по месяцам.

Таблица 3.4.

	0	1	2	min_max
0	2019-01-01 00:00:00-05:00	2019-01-31 23:58:00-05:00	68.736567	0.099984
1	2019-02-01 00:00:00-05:00	2019-02-28 23:44:00-05:00	60.942684	0.068376
2	2019-03-03 11:11:00-05:00	2019-03-31 23:59:00-04:00	71.04418	0.109342
3	2019-04-01 00:01:00-04:00	2019-04-30 23:59:00-04:00	71.192314	0.109943
4	2019-05-01 00:01:00-04:00	2019-05-31 23:59:00-04:00	100.16574	0.227443
5	2019-06-01 00:01:00-04:00	2019-06-20 09:57:00-04:00	60.199614	0.065363
8	2019-09-13 10:50:00-04:00	2019-09-30 23:59:00-04:00	44.082372	0
9	2019-10-01 00:01:00-04:00	2019-10-31 23:59:00-04:00	100.73837	0.229765
10	2019-11-01 00:01:00-04:00	2019-11-30 23:59:00-05:00	116.74783	0.294691

	0	1	2	min_max
11	2019-12-01 00:01:00-05:00	2019-12-31 23:59:00-05:00	85.653172	0.168588
12	2020-01-01 00:04:00-05:00	2020-01-31 23:59:00-05:00	290.66449	1
13	2020-02-01 00:04:00-05:00	2020-02-29 23:59:00-05:00	94.79583	0.205666
14	2020-03-01 00:04:00-05:00	2020-03-31 23:59:00-04:00	98.461131	0.22053
15	2020-04-01 00:01:00-04:00	2020-04-30 20:02:00-04:00	99.413062	0.224391
16	2020-05-01 06:03:00-04:00	2020-05-26 09:07:00-04:00	187.97278	0.58354
20	2020-09-03 14:52:00-04:00	2020-09-30 23:59:00-04:00	163.66449	0.484959
21	2020-10-01 00:01:00-04:00	2020-10-31 23:59:00-04:00	109.97559	0.267226
22	2020-11-01 00:01:00-04:00	2020-11-30 23:59:00-05:00	213.68169	0.687801
23	2020-12-01 00:00:00-05:00	2020-12-30 23:59:00-05:00	176.37012	0.536486

Для того, чтобы результаты площадей фигур, построенные по разным декомпозициям РСА, можно было сравнивать, к площадям фигур дополнительно было применено MinMax шкалирование.

Для всех зон были построены графики, где на оси абсцисс отмечались месяца, а на оси ординат площади фигур. В такой градации 1 – максимальная площадь фигуры, 0 – минимальная площадь фигуры.

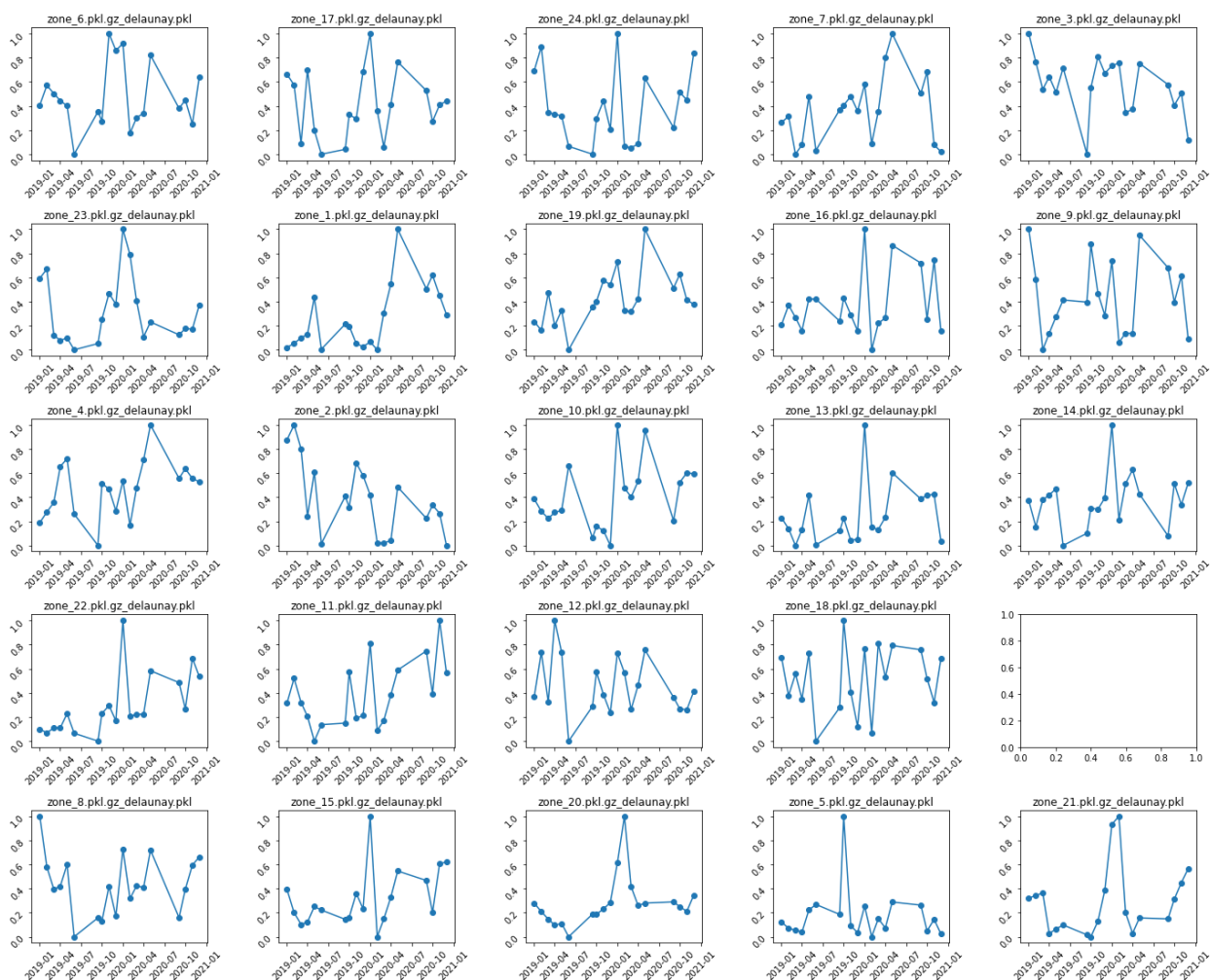


Рис. 3.6 – Нормализованная площадь фигур всех зон за весь период, разбитый по месяцам.

Несмотря на то, что каждая зона отапливается по-своему, РСА декомпозиция построена своя, и кучности точек, выглядят разными, в них все же наблюдается подобие.

К рассмотрению были приведены все кривые на один общий график, чтобы всецело скомпоновать результат и пронаблюдать, проявляются ли некоторые общие тенденции.

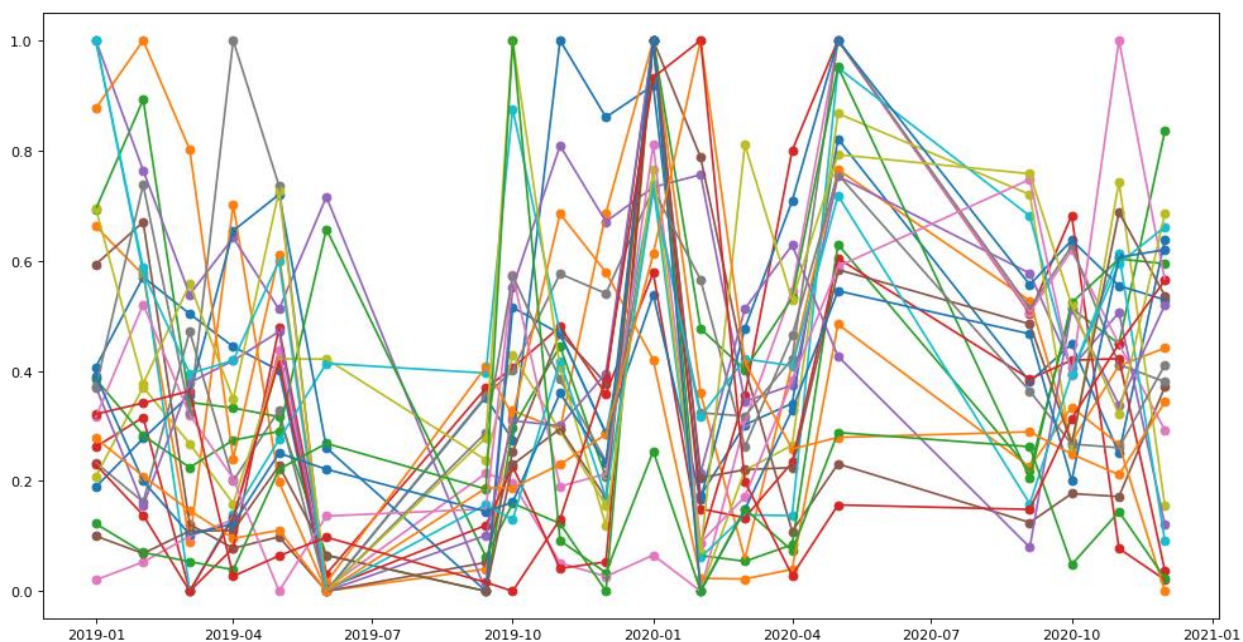


Рис. 3.7 – нормализованная площадь всех фигур, сведенная на один график.

Как видно из графика, наблюдается общая тенденция роста и упадка в определенные месяцы. На зимние месяцы в период активного отопления площади фигур показывают большие значения. В мае отопление выключается. Отопление включается в сентябре.

Для более детального рассмотрения опыт был повторен. Был выбран период 1 месяц и площади фигур разделились по дням.

На рис. 3.8 представлены графики всех зон. Для более детального рассмотрения далее будут выбраны другие зоны – 10, 9, 11, 12.

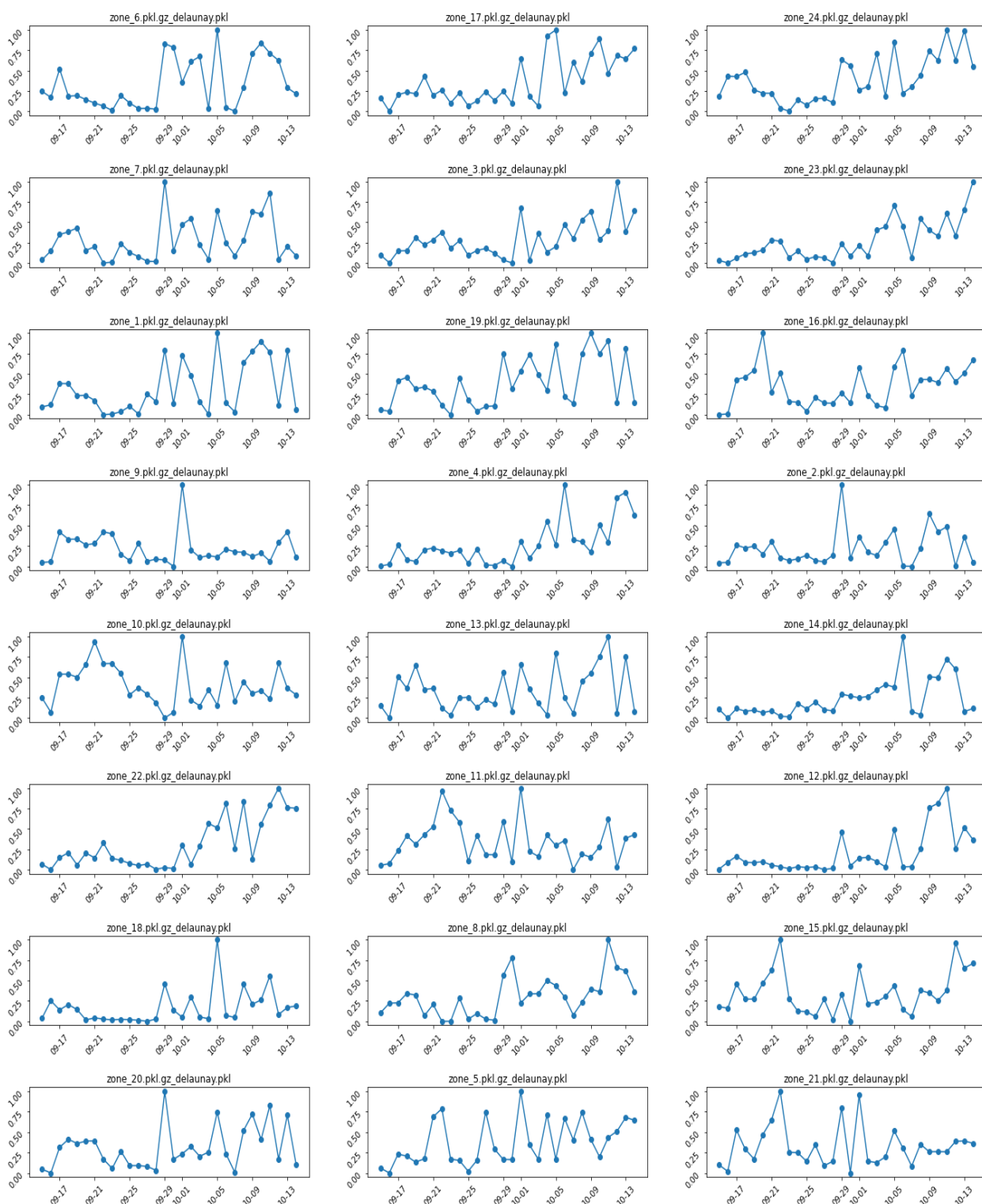


Рис. 3.8 – Графики площадей фигур по триангуляции Делоне за период одного месяца.

На рис. 3.9 представлены графики площадей фигур для четырех выбранных зон. В отчете представлены для более детального изучения в масштабе.

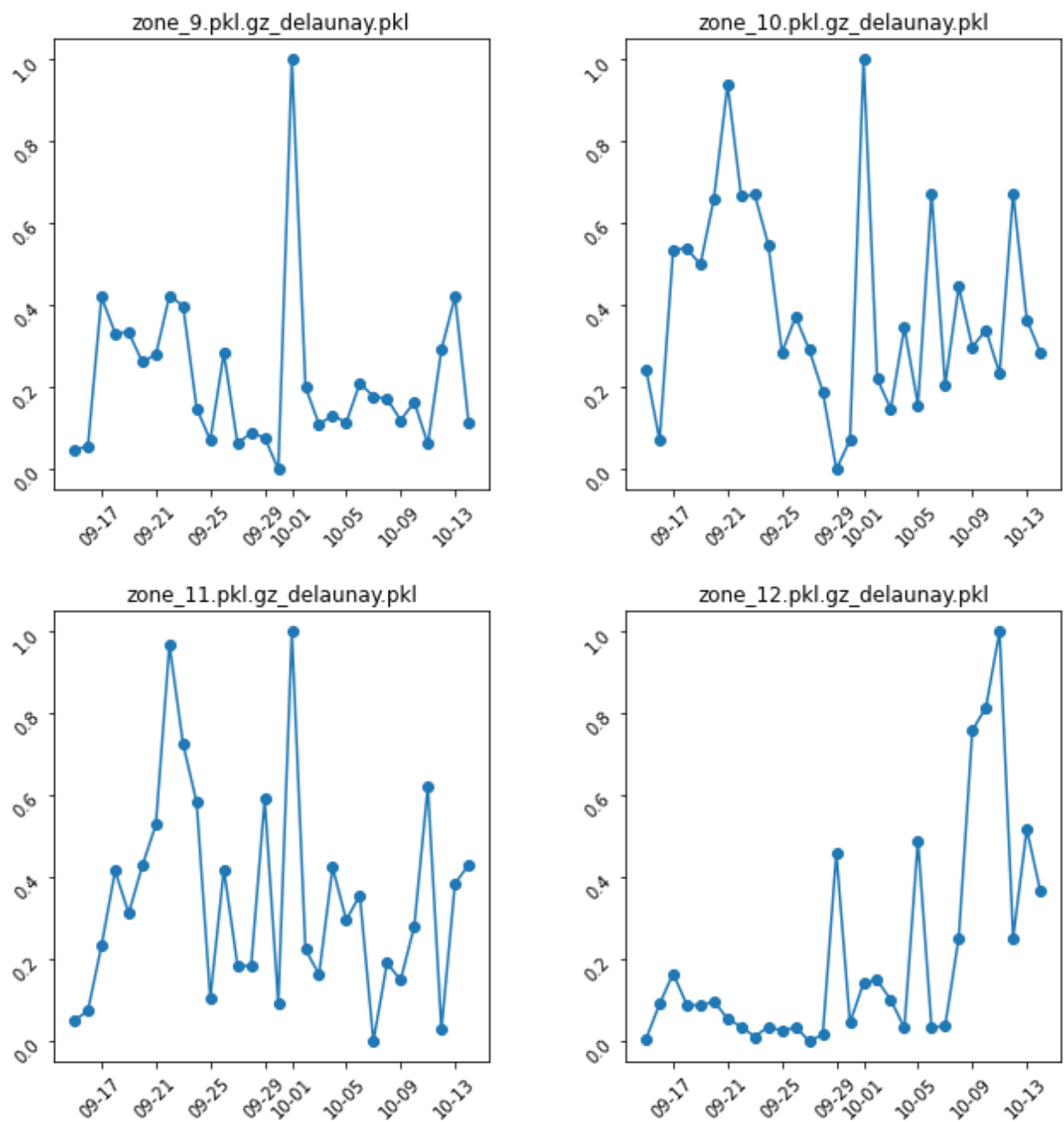


Рис 3.9 – график площади фигур четырех зон по триангуляции Делоне.

Свойства зон перенесены в отдельную таблицу.

Таблица 3.5

Номер зоны	Количество квартир	Количество этажей
9	5	12-17
10	35	12-17
11	5	12-17
12	5	12-17

Графики из предыдущего рисунка были совмещены на один график и приведены к общему масштабу. Получившиеся данные свидетельствуют о наличии некоторых особенностей.

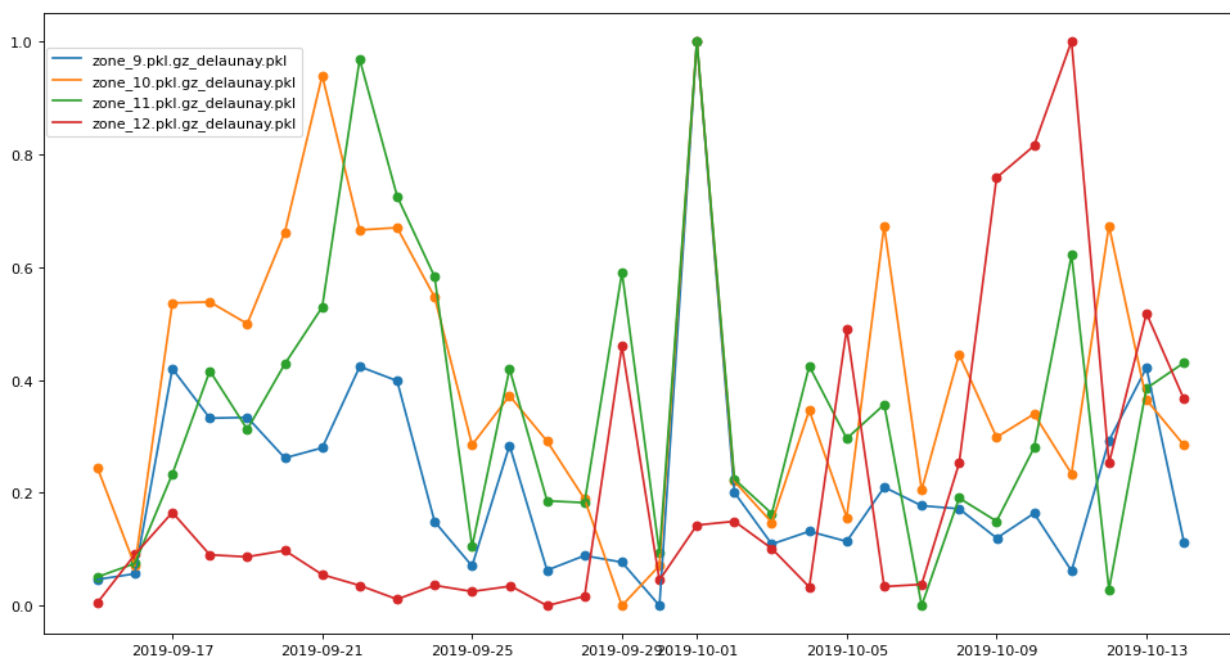


Рис. 3.10 – График температур в пиковый период.

Очень отчетливо акцентируется внимание на пиковом значении 1 октября 2019 года. По данным сайта world-weather.info в этот день наблюдалась сильная жара – днем температура достигала $+28^{\circ}\text{C}$ (рис 3.11).

Конечно, такая высокая температура может косвенно сказаться на температурных показателях внутри здания. Солнце под воздействием прямых лучей нагревает здание, повышая температуру внутри квартир.

В дополнении к этому максимальные значения достигаются и 11 октября. В этот день температура была хоть и не высокой, однако температурная амплитуда составляла 9 градусов. Погода была ясная, поэтому здание под воздействием солнечных лучей нагревалось сильнее, чем в обычные дни.

Стоит отметить, что количество квартир в зоне не влияет на то, какие величины площади фигур в конечном итоге получаются.

Weather in Hamilton in October 2019

Hamilton Weather Forecast for October 2019 is based on statistical data.

2015 2016 2017 2018 **2019** 2020 2021 2022





















Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Mon		Tue	Wed	Thu	Fri	Sat	Sun				
	1	2	3	4	5	6					
	 +28° night +17°	 +15° night +19°	 +12° night +9°	 +10° night +12°	 +12° night +4°	 +19° night +12°					
7	8	9	10	11	12	13					
 +16° night +12°	 +15° night +10°	 +15° night +7°	 +15° night +8°	 +18° night +9°	 +11° night +14°	 +14° night +5°					
14	15	16	17	18	19	20					
 +10° night +7°	 +13° night +4°	 +9° night +11°	 +9° night +6°	 +9° night +4°	 +12° night +2°	 +17° night +6°					

Рис. 3.11 – Фрагмент сайта о погоде в период исследования.

3.8. Автоматическое аннотирование данных

Опыт повторен и увеличен временной интервал вдвое. Для поиска паттернов выделен некоторый интервал, который считается исходным паттерном. На графике был выделен интервал из 9 зоны с периодом в одну неделю со 2 по 8 октября. На рис. 3.12 изображена кривая, которая характеризует состояние системы внутри зоны за период 7 дней.

Поиск похожих паттернов производился с помощью расчета индекса структурной похожести.

Выбранные данные на интервале становятся референсными данными (V_{ref}). Далее последовательно начиная с 0 происходит выбор вектора данных для сравнения (V_i), размер вектора такой же как V_{ref} . Для векторов V_i и V_{ref} производится расчет $ssim_i$. Далее окно выбора вектора смещается на заданный шаг K . Повтор действий, пока не достигнем конца данных. Получаем вектор $ssim$, который показывает схожесть интервалов данных с выделенным

интервалом. Все данные на интервале, который имеет значение схожести, большее чем заданный порог T , помечаются меткой D.

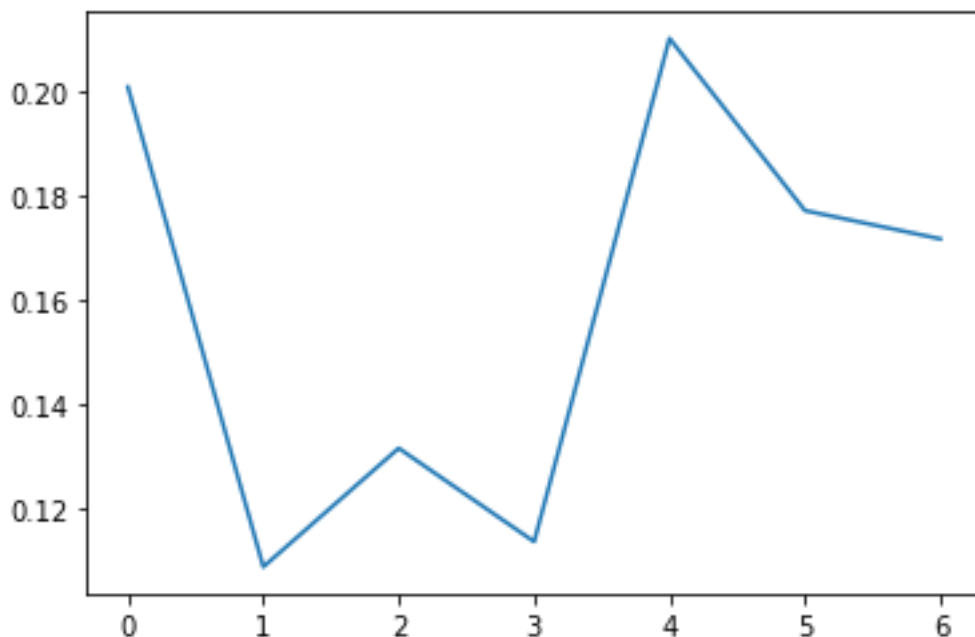


Рис 3.12 – исследуемый паттерн 9 зоны.

Программа проходит по циклу с шагом 1 и сравнивает референсный паттерн одновременно со всеми четырьмя wybranными для исследования зонами. В программе установлено условие – если значение индекса структурной схожести превышает 0.5 (его максимальное значение равно 1), тогда следует отпечатать этот промежуток в консоль.

Был получен следующий результат:

zone= 9 ssim= 1.0 indexes[17:24]

zone= 10 ssim= 0.5113971502360586 indexes[32:39]

zone= 10 ssim= 0.6592758089998122 indexes[52:59]

На полученных результатах отмечено, что с индексом 1 встречается тот самый паттерн, который был взят за основу.

Два других паттерна были рассмотрены более детально – они проявились в других временных промежутках зоны №10 (рис 3.13 и рис. 3.14)

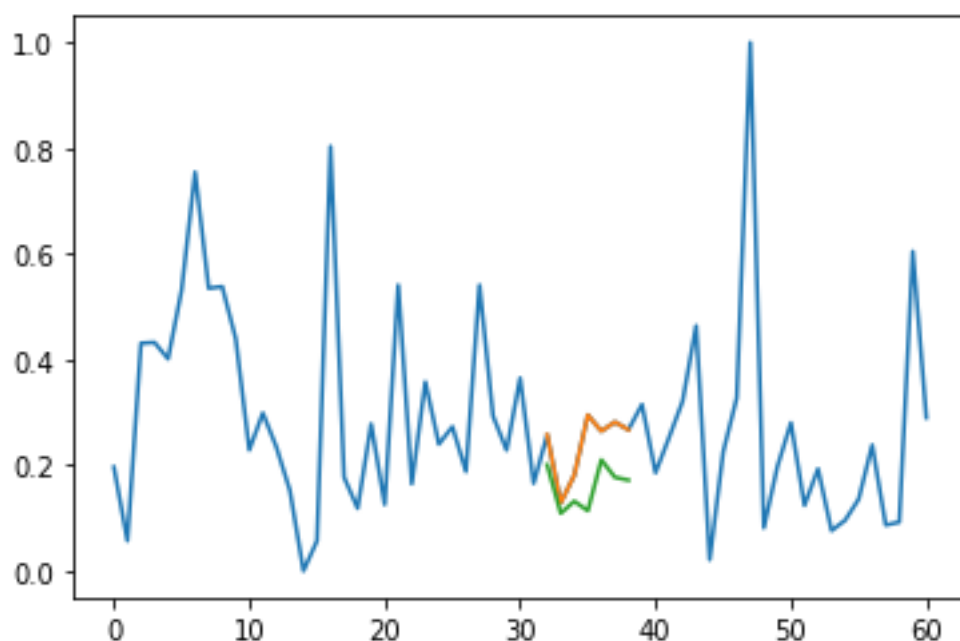


Рис. 3.13 – Обнаружение паттерна с индексом структурной похожести 0.51.

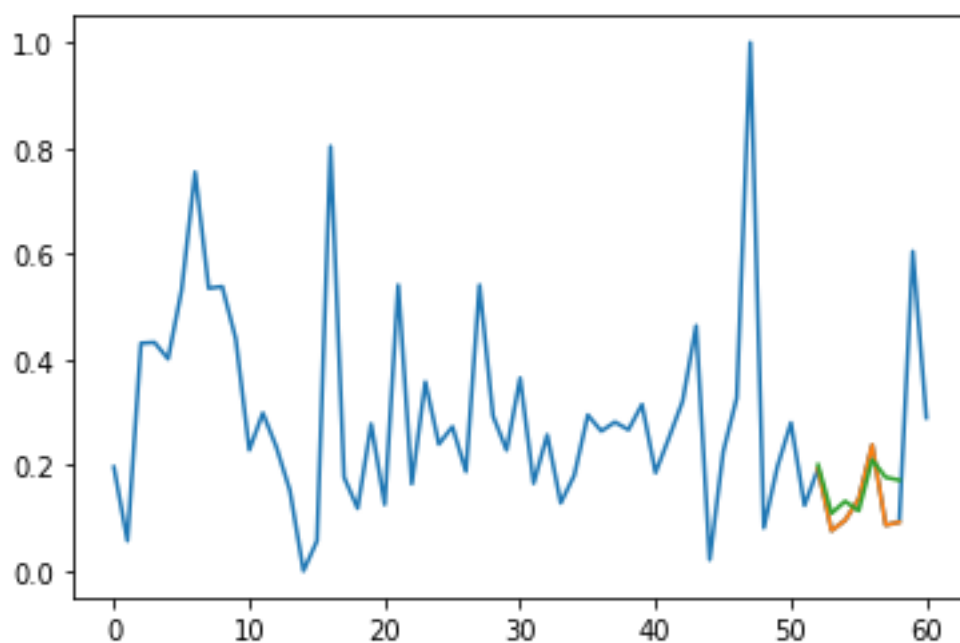


Рис. 3.14 – Обнаружение паттерна с индексом структурной похожести 0.65.

На графиках, представленных выше, зеленым цветом отмечается референсный паттерн, оранжевый цвет – цвет участка, который совпадает с паттерном.

Для того, чтобы оценить, с другой стороны, то, как паттерны себя проявляют на исходных данных, принято решение посмотреть какие параметры температур были зафиксированы в этот интервал времени.

Построены график, на котором по оси абсцисс размечены квартиры, а по оси ординат величины агрегированных значений. Проиллюстрировано минимальное, максимальное и среднее значение температур в указанный промежуток времени.

На рис. 3.15 проиллюстрировано состояние 10-й зоны в период времени с 17 октября по 24 октября.

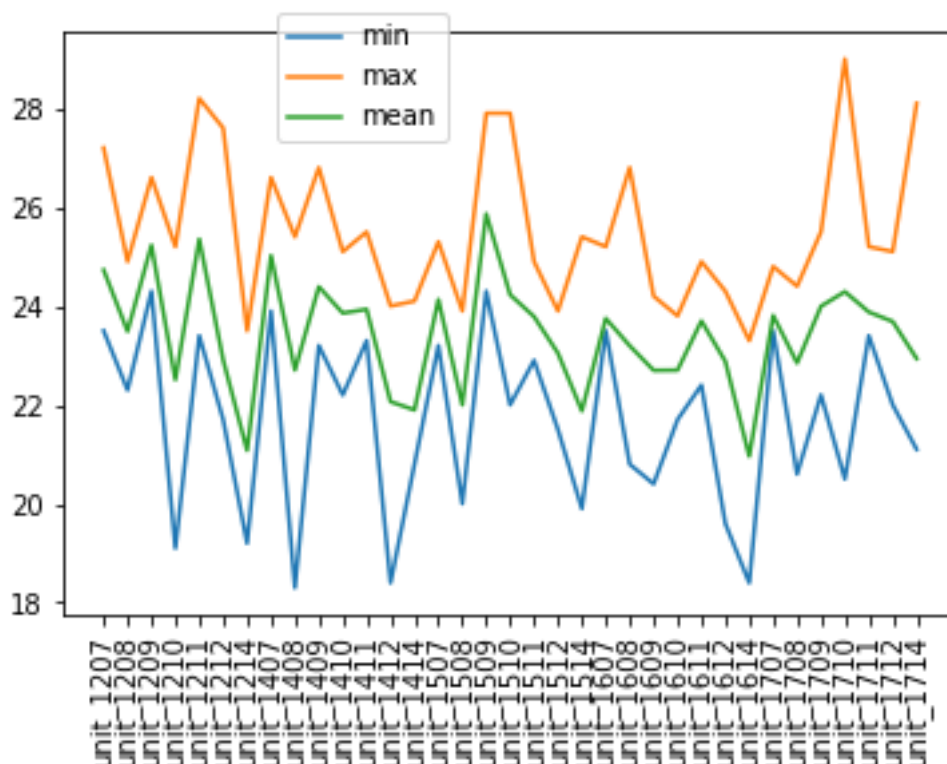


Рис. 3.15 – график агрегирующих показателей всех квартир по температурам с 17 по 24 октября.

Что немаловажно, был построен и другой график, который индекс структурной похожести ещё больший, чем предыдущий интервал времени. Итак, полученные значения построены и проиллюстрированы (рис. 3.16) тем же самым образом. Период времени указывается с 6 ноября по 13 ноября.

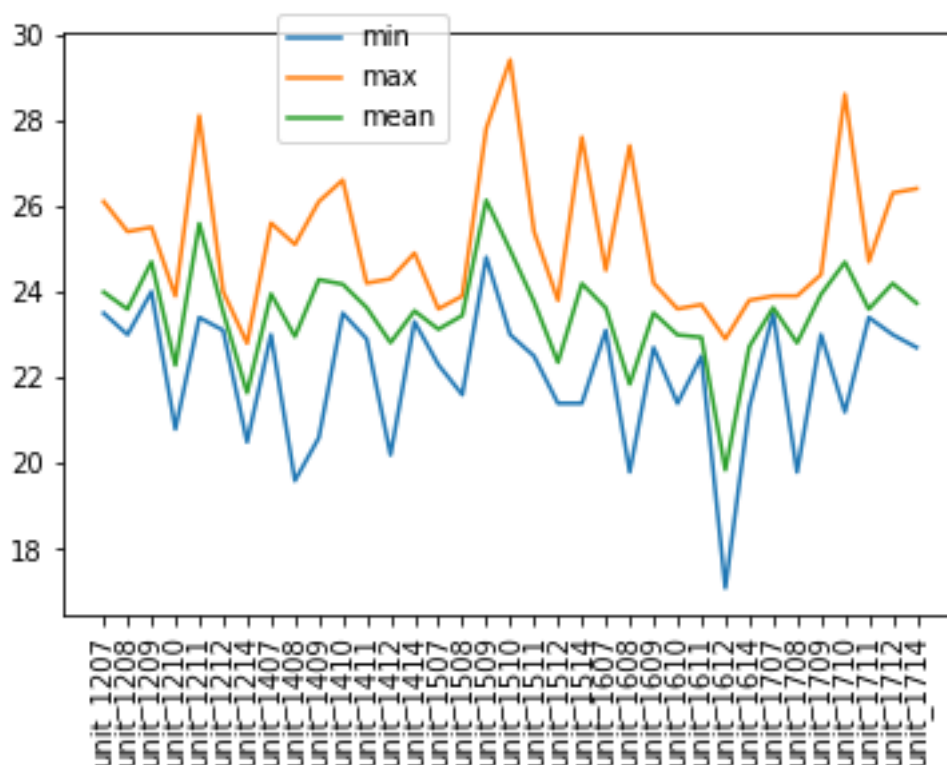


Рис. 3.16 - график агрегирующих показателей всех квартир по температурам с 6 по 13 ноября.

Действительно, состояние системы похоже, в этих интервалах времени самой холодной была квартира 1614. В нескольких квартирах наблюдался перегрев. Были зафиксированы температуры, превышающие 28°C.

Кривая средних температур имеет похожую линию.

Для исходного паттерна был так же построен график агрегирующих показателей, он представлен на рис. 3.17.

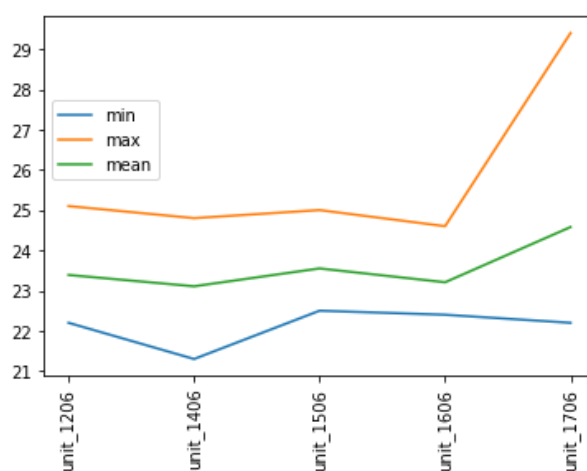


Рис. 3.17 – график агрегируемых показателей за период времени со 2 по 9 октября.

На исходном паттерне тоже наблюдается перегрев – в квартире на 17 этаже. За интервальный промежуток времени неделю видно, что температура имела схожие значения минимального, среднего и максимального показателей.

На рис. 3.18 отмечены температуры всех показателей за определенный промежуток времени.

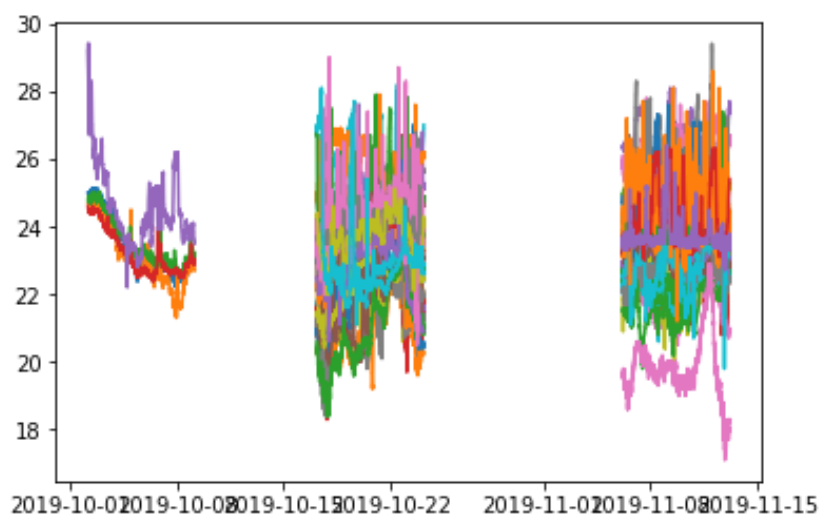


Рис. 3.18 – показания всех температур за выбранные промежутки времени.

На приведенных диаграммах размаха представлены два выделенных участка.

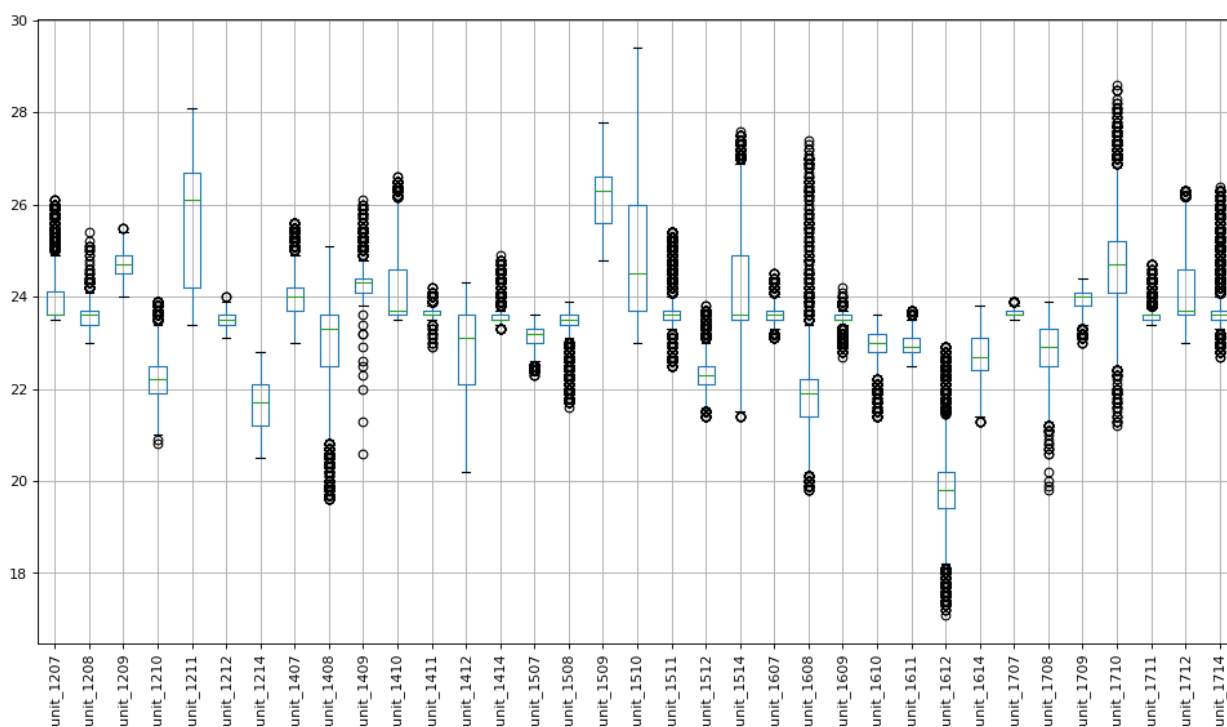


Рис. 3.19 – Диаграмма размаха для участка 10 зоны с 17 по 24 октября.

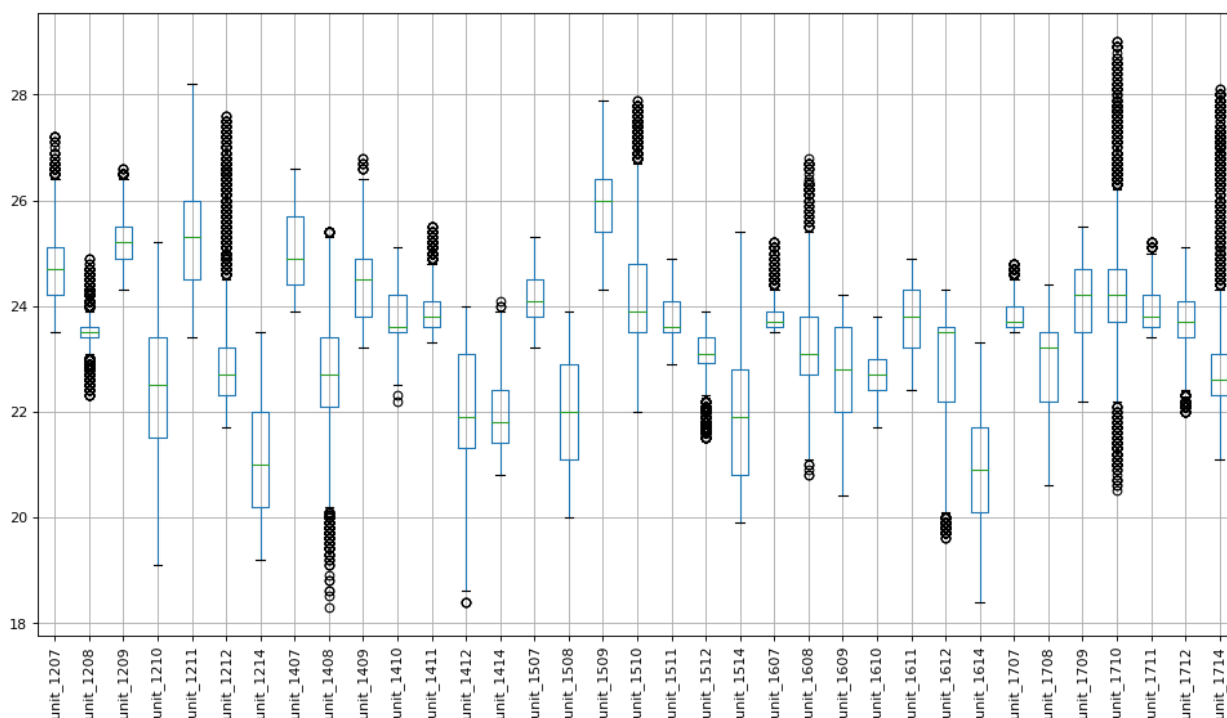


Рис. 3.20 – Диаграмма размаха для участка 10 зоны с 6 по 13 ноября.

График «ящик с усами», или «ящичковая диаграмма», был разработан Джоном Тьюки в 1970-х годах. По сути, ящик с усами — это быстрый способ изучения одного или нескольких наборов данных в графическом виде. Этот график может показаться более примитивным, чем, например, гистограммы, но он имеет некоторые преимущества. Он занимает меньше места и поэтому особенно полезен для сравнения распределений между несколькими группами или наборами данных. Кроме того, ящик с усами в своей первоначальной форме прост для построения.

Ящик с усами компактный и по нему легко можно оценить медианы, квантили, дисперсию и асимметрию в данных, а также выявить выбросы. Асимметрию данных можно увидеть не только по медиане, смещённой к какому-либо концу ящика, но и по разной длине усов, выходящих из ящика.

График «ящик с усами» очень прост для понимания и именно поэтому часто используется в различных публикациях для визуализации данных.

Наблюдается, что для каждой величины, выбросы и форма ящиков очень похожа, что говорит в пользу того, что паттерны схожи.

Таким образом, разработанный метод определения паттернов и автоматической аннотации хорошо себя проявил для многомерных данных. Может использоваться для выявления закономерностей на сложных данных и позволит выявлять ошибки в данных.

4. ТЕХНИКО-ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ

4.1. Описание концепции проекта

Название проекта

Программа по автоматическому аннотированию данных.

Сущность проекта

Проект включает в себя программный код и набор бинарных файлов конфигурации.

Деловая необходимость

Требуется разработать инструмент, который бы позволил автоматически размечать данные и выявлять паттерны поведения данных любого рода.

Цель проекта

Разработать программный продукт, который можно было бы продавать как готовое решение в виде платной библиотеки либо как готовая программа.

Исключения (границы проекта)

В проект не включается обучение персонала по работе с библиотекой, тестирование продукта на данных покупателя. Проект не рассматривает процесс развертывания системы на серверной инфраструктуре конечного пользователя.

Допущения

Полагается, что люди, которые заинтересованы в продукте имеют достаточную квалификацию, чтобы работать с данными.

Заинтересованные стороны проекта

Организации, которые имеют дело с большим потоком данных реального времени. Для них этот проект сможет позволить рассмотреть бизнес-процессы компании с иной стороны.

Риски проекта

Продукт может не показать свою жизнеспособность в конкуренции с другими методами аннотирования. Возможно, что ручной труд в некоторых ситуациях получится дешевле. Есть риск отсутствия спроса на рынке.

Ориентировочные сроки проекта

Предполагается, что релиз программного обеспечения состоится в конце 2021 года. К этому времени будут проверена работа на датасетах, будут выявлены ошибки, подготовятся необходимые документы и лицензии.

Первоначальная организация проекта

Для реализации проекта достаточно одного отдела – отдела разработки программного обеспечения. Юридические и прочие обязанности можно отправить на аутсорсинг.

Ориентировочный бюджет проекта

Предполагается, что бюджет проекта может составить около 2 000 000 рублей.

4.2. Описание продукции (результатов исследований)

Описание продукции охватывает ключевые вопросы о продукте и предлагает дополнительные комментарии. Описание подготовлено в таблице 4.1.

Таблица 4.1 – Описание продукции

Ключевые вопросы	Комментарии
1. Наименование продукции (результатов исследований)	Программа для построения паттернов поведения на основе больших данных методами триангуляции
2. Назначение продукта (результатов исследований)	Программный продукт автоматизирует процессы, которые могут вовлекать человеческий труд. Это позволит уменьшить потребность в найме дата-акселераторов и сократить количество потраченных часов дата саентистов. Продукт предназначен для использования дата саентистами и дата аналитиками в своих собственных проектах.

Ключевые вопросы	Комментарии
3. Основные характеристики продукта (результатов исследований)	Программный продукт написан на языке Python. Поддерживаются операционные системы Windows, Linux, MacOS. Минимальные технические характеристики: двухъядерный процессор, 4 Гб оперативной памяти, 16 Гб свободного дискового пространства. Продукт на стадии разработки, подготавливаются программные решения, калибруются алгоритмы на тестовых данных.
4. Потребительские свойства продукции (результатов исследований)	Потребители могут сэкономить трудовые затраты на разметку и выявление паттернов.
5. Основные конкурентные преимущества продукции (результатов исследований)	Программа не требует «обучения с учителем» и сама способна точно разметить данные и обнаружить выбросы или неустойчивое поведение системы.
6. Основные потребители и направления использования продукции (результатов исследований)	Предполагается, что продукция будет использоваться как готовая библиотека для языка программирования Python. Потребители продукта – специалисты в области анализа данных.
7. Юридическая защищенность продукции (результатов исследований)	На исследование планируется получить патент, а программное обеспечение распространять под лицензией.
8. Дополнительные сервисные услуги	Продукт поставляется через Интернет.

4.3. Анализ рынка сбыта продукции

Положение дел в отрасли

На сегодняшний день отрасль больших данных не перестает расти. Компании нуждаются в специализированных кадрах и готовы платить им большую зарплату.

Специалистов на рынке ещё не много, а опытные стоят дорого. Целесообразно брать на работу специалистов младших позиций с расчетом на то, что они «вырастут». Для профессии дата саентистов ещё не сформулирован

четкий набор задач, потому что все их проекты носят исследовательский характер.

Тем не менее, это не убавляет и не останавливает темпы роста рынка больших данных. На рис. 4.1 представлен прогноз выручки от объема больших данных. Ожидается, что рынок будет расти из-за ежегодного числа создаваемых данных человечеством.

Характеристика внешней среды проекта

Таблица 4.2 - SWOT-анализ

<p>Сильные стороны</p> <ol style="list-style-type: none"> 1. Программный продукт легко распространять через интернет. 2. Продукт доступен для любого компьютера. 3. Продукт имеет не высокие системные требования. 	<p>Слабые стороны</p> <ol style="list-style-type: none"> 1. Низкая устойчивость ко взлому и нелегальному распространению продукта. 2. Недостаточно физического контроля данных. 3. Сложно поддерживать стабильную работу на всех устройствах и конфигурациях.
<p>Возможности</p> <ol style="list-style-type: none"> 1. Развить масштабно продажи в онлайн. 2. Можно выбирать пути развития в соответствии с рынком. 3. Предлагать передовые технологии рынка доступно. 	<p>Угрозы</p> <ol style="list-style-type: none"> 1. Продукт может оказаться невостребованным. 2. Функционал программы может быть скопирован другими компаниями, которые могут сделать его доступнее. 3. Полученное решение может оказаться не оптимизированным для решения задач по автоматической аннотации.

Анализ рынка

По прогнозам, к 2027 году глобальный рынок больших данных вырастет до 103 миллиардов долларов США, что более чем вдвое превышает ожидаемый размер рынка в 2018 году. С долей в 45 процентов сегмент программного обеспечения станет крупным сегментом рынка больших данных к 2027 году[7].

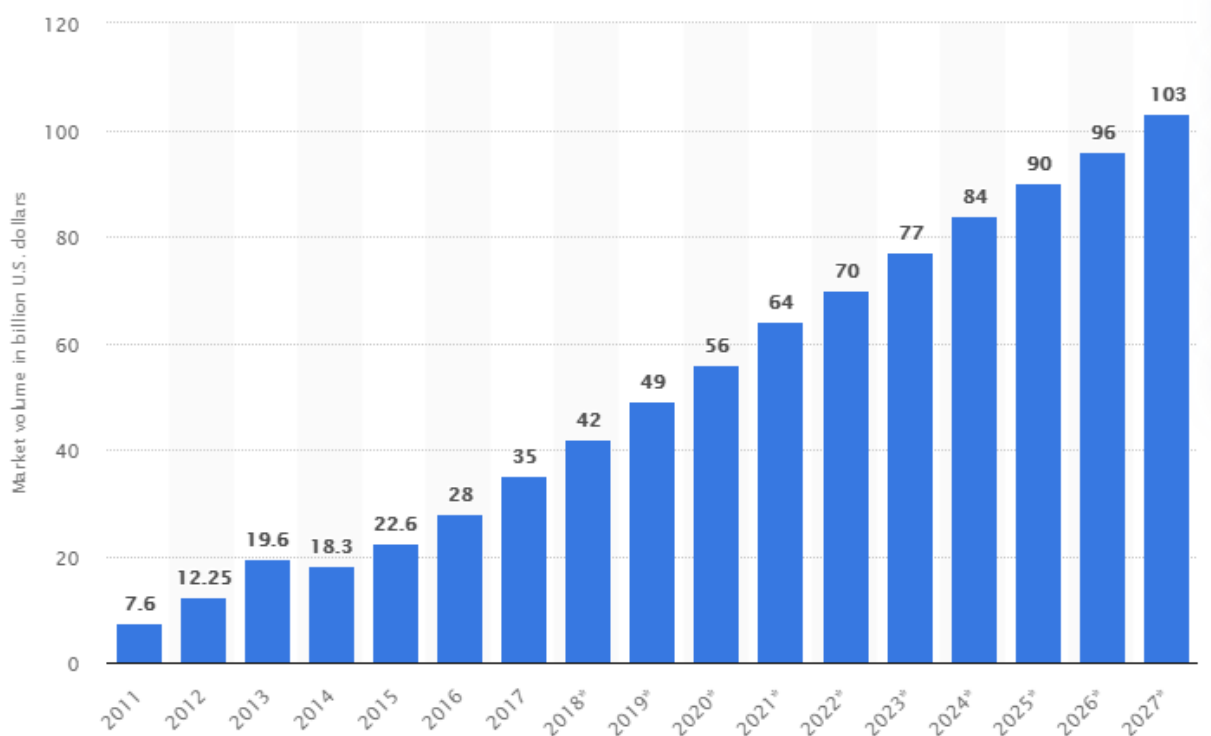


Рис. 4.1 – Прогноз выручки от объема рынка больших данных во всем мире с 2011 по 2027 год.

Потребность в анализе больших данных имеет высокий спрос на текущий момент, и он не будет убавляться в будущем. На выбор программного продукта могут влиять ряд факторов: цена – основополагающий, в нем подразумевается стоимость лицензии на использование программы; качество – продукт должен оправдывать свою стоимость и не проявлять ошибок. В качестве факторов, влияющих на формирование потребности можно выделить недостаток квалификации кадров и наличие на рынке альтернативных решений. На выбор программы может влиять и сервис – техническая поддержка. Под этим термином подразумевается исправление ошибок и помощь в настройке программы и оборудования. Компании, предлагающие свое программное обеспечение по анализу данных – крупные игроки на рынке. Microsoft и Apple покупают небольшие стартапы для воплощения своих проектов. Потребители продукта обращаются обычно к готовым библиотекам, которые можно найти в поисковой выдаче.

Сегментирование рынка и выбор целевых сегментов

В приложенной ниже таблице (4.3) сведены все ключевые вопросы и ожидаемые результаты процедур по вопросу сегментирования рынка и выбора целевых сегментов.

Таблица 4.3 – Сегментирование рынка

Ключевые вопросы	Комментарии к процедурам	Ожидаемый результат
1. Кто потребитель (заказчик) продукции/услуг?	Сегментация рынка	IT-организации, имеющие дело с хранением и обработкой большого потока данных
2. Каковы основные сегменты рынка? Какие потребители или группы являются наиболее привлекательными в финансовом отношении? В каких сегментах рынка имеется устойчивый спрос на товар/услугу?	Выбор целевых сегментов	Основные сегменты рынка информационных технологий: информационные сервисы, программное обеспечение, телекоммуникационные сервисы, аппаратное обеспечение. Наиболее привлекательная группа в финансовом отношении – информационные сервисы. Устойчивый спрос наблюдается во всех сегментах.
3. В чем заключаются товарные особенности целевых сегментов и их ассортиментное наполнение? Как компания будет позиционировать свой товар?	Позиционирование	Компания будет позиционировать товар как готовое программное обеспечение, которое легко внедрить в серверную инфраструктуру или запустить локально на компьютере.
4. Каков спрос и оценка потенциала рынка в целом и по сегментам? Какие цены на рынке? Какую долю компания собирается занимать?	Анализ и оценка привлекательности рынка	В целом спрос на исследование больших данных проявляется во всех сегментах рынка. В частности, наиболее активный спрос наблюдается в сегменте рынка информационных сервисов. На рынке лицензия на программное обеспечение для анализа данных может стоить от 20 долларов в месяц (Microsoft BI Premium). Компания собирается занять 1% мирового рынка.

4.4. Анализ конкурентов

Инструменты аналитика данных — это термин, используемый для описания программного обеспечения и приложений, которые аналитики данных используют для разработки и выполнения аналитических процессов, которые помогают компаниям принимать более обоснованные бизнес-решения при одновременном снижении затрат и увеличении прибыли.

Таблица 4.4 – Анализ конкурентов

Название конкурента / конкурирующего проекта	Применение	Сильные стороны	Слабые стороны
Excel	Обработка данных выполняется в соответствии с общими офисными требованиями. Управление данными и хранение малых и средних компаний. Простой статистический анализ для студентов или преподавателей (например, дисперсионный анализ, регрессионный анализ и т. Д.). Объедините Word и PowerPoint для создания отчетов об анализе данных. Помощник аналитика данных. Изготовление диаграмм для некоторых деловых журналов и газет (визуализация данных).	1. Начать работу с Excel легко. 2. Учебные ресурсы очень насыщены. 3. В Excel можно множество вещей: моделирование, визуализацию, отчеты, динамические диаграммы и т. д. 4. Это может помочь понять значение многих операций, прежде чем изучать другие инструменты (например, Python и R).	Чтобы полностью освоить Excel, вам необходимо изучить VBA, поэтому сложность по-прежнему очень высока. Когда объем данных большой, возникает ситуация заикания. Сам файл данных Excel может содержать только 1,08 миллиона строк без помощи других инструментов, и он не подходит для обработки крупномасштабных наборов данных. Встроенный статистический анализ слишком прост и не имеет практического значения. В отличие от Python, R и другого программного обеспечения с открытым исходным кодом, подлинный Excel является платным.

Название конкурента / конкурирующего проекта	Применение	Сильные стороны	Слабые стороны
Язык программирования R	<p>Функции R охватывают практически любую область, где необходимы данные. Что касается нашего общего анализа данных или академической работы по анализу данных, то, что может делать R, в основном включает следующие аспекты.</p> <p>Очистка и сокращение данных.</p> <p>Веб-сканирование.</p> <p>Визуализация данных.</p> <p>Статистическая проверка гипотез (t-критерий, дисперсионный анализ, критерий хи-квадрат и т. д.).</p> <p>Статистическое моделирование (линейная регрессия, логистическая регрессия, древовидная модель, нейронная сеть и т. д.).</p> <p>Вывод отчета об анализе данных (разметка R).</p>	<p>Легко освоить основы использования, базовую структуру данных, импорт и экспорт данных, а также простую визуализацию данных. Зная основы, при встрече с реальными проблемами, можно найти пакет R, который нужен. Читая файлы справки R и информацию в сети, можно относительно быстро решать определенные проблемы.</p>	<p>Требуется понимания работы языков программирования. Не предлагает готовых решений для многокритериальных данных «из коробки», все можно реализовать только своими усилиями, подключая сторонние библиотеки.</p>

Название конкурента / конкурирующего проекта	Применение	Сильные стороны	Слабые стороны
Язык программирования Python	<p>Сканирование данных.</p> <p>Очистка данных;</p> <p>Моделирование данных;</p> <p>Создание алгоритмов анализа данных на основе бизнес-сценариев и актуальных проблем.</p> <p>Визуализация данных.</p> <p>Расширенные области интеллектуального анализа данных и анализа, такие как машинное обучение и интеллектуальный анализ текста.</p>	<p>1. Обширные библиотеки</p> <p>2. Расширяемый</p> <p>3. Встраиваемый</p> <p>4. Повышенная производительность.</p> <p>5. Возможности Интернета вещей</p> <p>6. Просто и легко</p> <p>7. Читаемый</p> <p>8. Объектно-ориентированный</p> <p>9. Бесплатно и с открытым исходным кодом</p> <p>10. Портативный</p> <p>11. Интерпретируемый</p>	<p>1. Ограничения скорости</p> <p>2. Слабость в мобильных вычислениях и браузерах</p> <p>3. Ограничения дизайна</p> <p>4. Слабо развитые уровни доступа к базе данных</p> <p>5. Простой</p>
Microsoft Power BI	<p>Подключение к данным</p> <p>Преобразование и моделирование данных</p> <p>Создание диаграмм и графиков</p> <p>Создание отчетов и информационных панелей, которые представляют собой коллекции визуальных элементов.</p> <p>Возможность делиться отчетами с другими с помощью службы Power BI.</p>	<p>Стоимость</p> <p>Кривая обучения</p> <p>Постоянные обновления и инновации</p> <p>Источники данных</p> <p>Интеграция с Excel</p> <p>Пользовательские визуализации</p>	<p>Пользовательский интерфейс</p> <p>Жесткие формулы</p> <p>Емкость обработки данных для бесплатных версий</p> <p>Возможность настройки визуализации</p> <p>Отношения между таблицами</p>

Название конкурента / конкурирующего проекта	Применение	Сильные стороны	Слабые стороны
Tableau	<p>ИТ - инвентаризация аппаратного и программного обеспечения, количество обращений в службу поддержки / время решения, распределение ресурсов, соответствие исправлений безопасности</p> <p>Финансы - планирование бюджета и расходов, кредиторская задолженность, командировочные расходы</p> <p>Маркетинг - участие в кампании, участие в сети, лиды</p> <p>Человеческие ресурсы - текучесть кадров, открытая численность персонала, удержание новых сотрудников, удовлетворенность сотрудников</p> <p>Продажи - отслеживание продаж / квот, охват воронки продаж, средний размер сделки, коэффициент выигрыша / проигрыша.</p> <p>Операции с оборудованием - Физические местоположения, объем колл-центра / распределение рабочей нагрузки, объем рабочих запросов / время решения</p>	<p>1. Вариативные возможности визуализации</p> <p>2. Простота использования</p> <p>3. Высокая производительность</p> <p>4. Подключение к нескольким источникам данных.</p> <p>5. Процветающее сообщество и форум</p> <p>6. Удобство для мобильных устройств</p>	<p>1. Высокая стоимость</p> <p>2. Негибкое ценообразование.</p> <p>3. Плохая послепродажная поддержка.</p> <p>4. Проблемы безопасности</p> <p>5. ИТ-помощь для правильного использования</p> <p>6. Слабые возможности бизнес-аналитики</p> <p>7. Плохое управление версиями</p> <p>8. Проблемы с внедрением</p> <p>9. Обучение персонала, требующее больших затрат времени и ресурсов</p>

Данный проект примет во внимание все недостатки имеющихся готовых решений и предложит доступные цены и позволит преодолеть порог

вхождения для новичков. Ожидается, что продукт внесет новаторские методы, которые не используются в готовых программах по аналитике данных.

4.5. План маркетинга

Товарная политика

1. Характеристики продукта

Язык программирования – Python. Имеет возможность подключаться модульно в любой проект. Поддерживает все операционные системы. Требуется версия Python 3.9+. Для работы требует 4 гигабайта оперативной памяти и 16 гигабайт дискового пространства. Использует многоядерную обработку, поэтому минимальное рекомендуемое число ядер – 2.

2. Стратегия развития ассортимента (в сторону более сложных и дорогих или упрощенных и дешевых моделей)

Ассортимент функционала развивается в сторону более сложных моделей.

3. Содержание инструкций к товару

Программный продукт будет сопровожден технической документацией к нему и набором базовых уроков по обработке данных.

4. Дополнительные услуги (монтаж, обучение, гарантии).

Дополнительные услуги не подразумеваются.

Распределительная политика

1. Планируемая география сбыта – мировой рынок (преимущественно Европа, Северная Америка)

2. Характеристики каналов распределения (виды каналов, длина каналов), стратегия распределения – интенсивный, селективный или эксклюзивный сбыт

3. Способы поиска посредников – через заказные статьи в Интернете.

4. Организация доставки товара – дистрибуция лицензионных ключей через электронную почту.

5. План продаж по годам.

5.1. Количество лицензий в первый год – 250 шт.

5.2. Увеличение числа продаж в последующие годы в 10 раз.

4.6. Коммуникационная политика

Концепции и инструменты продвижения

Для продвижения будет использоваться в основном только каналы интернет-рекламы. Во-первых, потому что можно настроить гибкую ценовую политику. Во-вторых, можно точно настроить таргетинг именно на целевых потребителей.

Таблица 4.5 – Коммуникационная политика.

Инструмент	Канал	Концепция
Реклама	Текстовая реклама для ВКонakte	Gif-анимация с движущимся текстом, который одним слоганом объясняет какие задачи решает продукт.
	Ролик на Youtube	Короткая 15-секундная видео реклама, где бородатый мужчина произносит одну фразу. На дисплее монитора большие таблицы и данные.
	Реклама в Instagram	Вертикальное видео, такое же, которое смонтировано для YouTube ролика, но адаптированное под формат мобильных устройств.
	Акция	Предлагать скидки на общеизвестные дни распродаж – черная пятница, киберпонедельник, день холостяка. Giveaway в рождественские праздники.
	Обзорные статьи в научных журналах.	Продемонстрировать метод и привести кий пример.
Стимулирование сбыта	Благотворительность	Перечисление части выручки с каждой проданной лицензии в фонды по защите окружающей среды.
Связи с общественностью	Участие в выставках и онлайн вебинарах	Подготовка презентаций и печать необходимого материала.
	На сайте компании	Скриншоты товара, его характеристики.
Интернет-представительство	Аккаунты на GitHub, Instagram, YouTube.	На GitHub будут размещаться репозитории плагинов для программы, Instagram как основной блог компании, YouTube для видеозаписей семинаров и презентаций.

Таблица 4.6 – Расчет бюджета продвижения

Инструмент	Наименование мероприятия	Период мероприятия	Место размещения	Расчет затрат	Итого сумма, руб.
Реклама ВКонтакте	Таргетированная реклама	2021 год	Социальная сеть «ВКонтакте»	30000 руб. в месяц	360000 руб.
Реклама в Instagram Stories	Таргетированная реклама	2021 год	Социальная сеть Instagram	30000 руб. в месяц	360000 руб.
Участие в конференциях с докладом	Data Science Con	2021 год	Онлайн-конференция	15000 руб. за участие	15000 руб.

Таблица 4.7 – Расходы на рекламную кампанию за 2021 год

Показатель	Май	Июнь	Июль	Август	Сентябрь	Октябрь	Ноябрь	Декабрь	Январь	Февраль	Март	Апрель	Итого за 2021 год
Расходы на рекламную кампанию с НДС, тыс. руб.	30	30	30	30	30	30	30	30	30	30	30	30	360
НДС, тыс. руб.	24	24	24	24	24	24	24	24	24	24	24	24	288
Расходы на рекламную кампанию без НДС, тыс. руб.	6	6	6	6	6	6	6	6	6	6	6	6	72

Таблица 4.8 – График мероприятий

Мероприятия	1 кв	2 кв	3 кв	4 кв
Реклама ВКонтакте				
Реклама в Instagram				
Участие в онлайн-семинаре				

4.7. Ценовая политика

Цена на лицензию должна покрывать стоимость расходов, и доля выручки за неё должно идти на благотворительность. Лицензия выдается на одну ЭВМ, завязывается на мак-адрес устройства. Лицензионный ключ

синхронизируется с сервером, который контролирует срок действия лицензий. Срок действия лицензии ограничен и составляет три месяца.

Предполагается, что установочная цена лицензии будет не высокой – 2999 рублей. Это дешевле, чем предлагают компании, продающие программы для аналитики данных (Microsoft Power BI Pro).

Себестоимость продукта 814540 руб.

Планируется предлагать скидки на общеизвестные дни распродаж – черная пятница, киберпонедельник, день холостяка. Организация giveaway (бесплатная раздача со 100% скидкой) в рождественские праздники.

4.8. План Продаж

План продаж рассчитан на два года – это максимальный рекомендуемый срок, на котором можно оценить реальные возможные продажи. Связано это с тем, что рынок очень динамичный.

Таблица 4.9 – План продаж

Показатели	Кварталы								Всего
	I	II	III	IV	I	II	III	IV	
Ожидаемый объем продаж, ед.	250	300	350	500	500	800	1000	1200	4900
Цена с НДС, тыс руб	3	3	3	3	3	3	3	3	
Выручка с НДС, тыс руб	750	900	1050	1500	1500	2400	3000	3600	
Нетто-выручка (без НДС), тыс руб	600	720	840	1200	1200	1920	2400	2880	11760
Сумма НДС, тыс руб	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	
ВСЕГО	150000	216000	294000	600000	600000	1536000	2400000	3456000	9252000

4.9. План производства

Деятельность будет осуществляться на действующем предприятии в отделе разработки компании K3D. Для производства цифрового продукта не

требуется аренда помещений. Производство налажено в режиме удаленной работы. Все рабочие процессы фиксируются в онлайн.

Таблица 4.9 – План производства

№ п/п	Наименование	Количество	Цена за единицу с НДС, руб.	Цена за единицу без НДС, руб.	Стоимость без НДС, руб.	Сумма НДС, руб	Поставщик
1	Yandex.Cloud виртуальная машина	2	1500	1200	2400	300	Yandex LLC
	ИТОГО	2	1500	1200	2400	300	

Потребность в производственном персонале

Основные требования к персоналу по разработке продукта следующие:

- - Наличие опыта командной работы в коммерческих проектах более 5 лет.
- Знание Python 3 сильных и слабых сторон языка.
- Умение работать с системой контроля версий Git.
- Умение писать объектно-ориентированный, структурированный и легко читаемый код с грамотными комментариями на английском языке.
- Способность грамотно формулировать свои мысли в письменной форме на русском и английском языках.
- Высшее техническое образование.

Приветствуется:

- Навыки работы с nix среде (centos), консоль
- Опыт работы с Angular, TypeScript
- Опыт разработки Single-page приложений
- Опыт развёртывания Web-приложений (Docker)

Трудоемкость работ установлена в человеко-часах. Для исполнителей была определена часовая ставка. Базируясь на сведениях о трудоемкости и ставке, был произведен расчет расходов на основную заработную плату исполнителей, отчисление страховых взносов. Принимая во внимание факт, что в

месяце 21 рабочий день, где каждый рабочий день – 8 часов, устанавливается, что всего в месяце 168 рабочих часов.

Согласно данным сайта trud.com[8] средняя заработная плата джуниор разработчика 80000 рублей. На основе данных о трудоемкости и часовой ставки соответствующих исполнителей произведены расчеты на заработную плату и отчисления на обязательное социально, пенсионное и медицинское страхование.

Расходы на основную заработную плату рассчитываются согласно формуле:

$$Z_{\text{осн.з./пл}} = \sum_{i=1}^k T_i C_i$$

где $Z_{\text{осн.з./пл}}$ - расходы на основную заработную плату исполнителей (руб.);

k – количество исполнителей;

T_i - время, затраченное i -м исполнителем на проведение исследований (дни);

C_i - ставка i -го исполнителя (руб./день)

$$Z_{\text{осн.зп}} = (165 \cdot 238) = 46053 \text{ (руб.)}.$$

Таблица 4.10 – Расчет стоимости выполнения ВКР на основе часов

№	Наименование работ	Исполнитель	Трудоем- кость, t_0 , ч	Ставка, руб./ч	Затраты, руб.
1	Разработка темы ВКР	Студент	4	476	1904
2	Обзор литературы по теме работы и её анализ	Студент	24	476	11424
3	Составление плана работы	Студент	2	476	952
4	Изучение документации библиотек по Python	Студент	36	476	17136
5	Изучение документации по алгоритмам	Студент	16	476	7616

6	Аудит имеющейся серверной архитектуры	Студент	12	476	5712
7	Решение поставленной задачи по поиску паттернов и автоматическому аннотированию	Студент	32	476	15232
8	Оформление пояснительной записки	Студент	40	476	19040
9	Консультации с научным руководителем	Студент	15	476	7140
10	Расчет технико-экономического обоснования	Студент	16	476	7616
	Итого	-	177	-	93772

Таблица 4.11 – Затраты на заработную плату сотрудников.

Должность	Количество человек	Заработная плата, руб	Отчисления на социальные нужды, руб	Итого (З/П + отчисления), руб
Программист Python	1	93000	27900	120900
	Итого в месяц:	93000	27900	120900
	Итого в год:	1,116,000	334,800	1,450,800

Согласно нормативу общепроизводственных и общехозяйственных затрат персонал компании, в которой происходит расчет представлен в таблице.

Таблица 4.12 – Затраты на общепроизводственные расходы

N	Должность	Численность, чел.	Оклад, руб.	Заработная плата в месяц, руб.	Заработная плата за год, руб.
1	Генеральный директор	1	50000	150000	1800000
2	Коммерческий директор	1	50000	150000	1800000
3	Главный бухгалтер	1	50000	150000	1800000
4	Менеджер по маркетингу	1	50000	80000	1200000

Расчет инвестиционных расходов

Инвестиционные расходы включают в себя затраты на запуск проекта – расходы на НИОКР, оформление документов, предварительная разработка проекта и пуско-наладочные работы.

Таблица 4.13 – Инвестиционные расходы

№	Наименование	Сумма, руб.
1.	Расходы на НИОКР	93772
2.	Регистрация предприятия, подготовка производства, оформление международного патента	65000
3.	Затраты на предварительную разработку проекта	80000
4.	Пуско-наладочные работы	12000
	Всего	157000

Расчет себестоимости продукции

Накладные расходы — это затраты, связанные с формированием условий, способствующих процессу работы, а также его организация и управление им. Накладные расходы вычислены согласно формуле (принято, что они составляют 20% от основной заработной платы):

$$З_{\text{Н}} = З_{\text{осн.з./пл}} \cdot 0,2 = 80000 \cdot 0,2 = 16000 \text{ (руб.)}.$$

В результате объем накладных расходов составил 16000 рублей.

Дополнительная заработная плата рассчитывается по формуле:

$$З_{\text{доп.з./пл}} = З_{\text{осн.з./пл}} \cdot \frac{H_{\text{доп}}}{100},$$

где $З_{\text{доп.з./пл}}$ - расходы на дополнительную заработную плату исполнителей в рублях; $З_{\text{осн.з./пл}}$ - расходы на основную заработную плату в рублях; $H_{\text{доп}}$ - норматив дополнительной заработной платы в процентах (был принят равным 8.3%).

Совершены расчеты, согласно формуле:

$$З_{\text{доп.з./пл}} = 80000 \cdot \frac{8.3}{100} = 6640 \text{ (руб.)}.$$

Следовательно, затраты на дополнительную заработную плату — 6640 рублей.

Смета затрат на производство вынесена в отдельную таблицу.

Таблица 4.14 – Смета затрат на производство

1. Материалы и комплектующие изделия	0
2. Основная зарплата производственных рабочих	93000
3. Дополнительная зарплата произв. рабочих	6640
4. Страховые взносы на социальные нужды (30%)	27900
5. Общепроизводственные расходы	0
6. Общехозяйственные и управленческие расходы	600000
7. Коммерческие расходы	87000
Итого полная себестоимость продукции	814540

4.10. Организационный план

Характеристика организации, реализующей проект

Организационно-правовая форма компании – корпорация (inc.) Компания основана в 2010 году. Опыт работы в отрасли 10 лет, основные этапы развития:

- 2010 год – 3 здания подключены к системе;
- 2015 год – 20 зданий подключены к системе;
- 2020 год – 50 зданий подключены к системе.

Руководитель – Князев Д., основатель компании русский эмигрант.

Нормативно-правовое регулирование

Для осуществления проектов не требуется получать особых разрешений, кроме разрешения руководителя компании. Разрешение от клиентов бизнеса получать не требуется, потому что при оформлении контракта они дают согласие на использование и обработку ПД.

Данные о разработчиках проекта

В разработке принимает участие один человек, он проверяет гипотезу и работоспособность получаемого продукта.

Таблица 4.15 – Данные о разработчиках проекта

ФИО	Должность	Роль в проекте	Стаж	Образование, специальность	Опыт осуществления проектов	Примечание: ученая степень, звание, профессиональные награды, повышение квалификации и проч.
Уруков Серафим Дмитриевич	Программист	Разработчик-исследователь	5 лет	Информационные системы и технологии, СПбГЭТУ «ЛЭТИ», 2019 г.	Разрабатывал программное обеспечение для серверов компании	Отсутствуют

Организационная структура

В работе из персонала по организации принимают участие следующие лица

Таблица 4.16 – Организационная структура проекта

ФИО	Должность	Роль в проекте	Стаж	Образование, специальность	Опыт осуществления проектов	Примечание: ученая степень, звание, профессиональные награды, повышение квалификации и проч.
Князев Д.	Гендиректор, инженер	Генеральный директор, коммерческий директор	10 лет	МГУ, 1995	Оформлял контракты с владельцами зданий	Отсутствуют
Пипов С.	Менеджер по управлению проектами	Менеджер по маркетингу	10 лет	МГИМО, 1993	Исследовал рынок, принимал участие в оформлении контрактов	Отсутствуют
Князева Ю.	Главный бухгалтер	Бухгалтер	10 лет	ИТМО, 2003	Руководила бухгалтером	Отсутствуют

Календарный план проекта

Продолжительность этапов проекта (до запуска в эксплуатацию) рассчитана на основе времени, потраченного на разработку и исследование в рамках выполнения выпускной квалификационной работы.

Таблица 4.17 – Работы по выполнению проекта

№ работы	Наименование работы	Предыдущие работы	Продолжительность, дни
1	Разработка технического задания	-	3
2	Расчет требуемых ресурсов	1	3
3	Проектирование	2	3
4	Создание и испытание опытного образца	3	3
5	Получение необходимых разрешений	3	1
6	Разработка программного обеспечения	5	7
7	Тестирование программного обеспечения	6	5
8	Настройка развертывания системы	7	6

График проекта

На рис. 4.2 представлена диаграмма Ганта, построенная в онлайн сервисе по планированию проекта. Она визуализирует последовательность работ и сроки их выполнения на одном изображении.

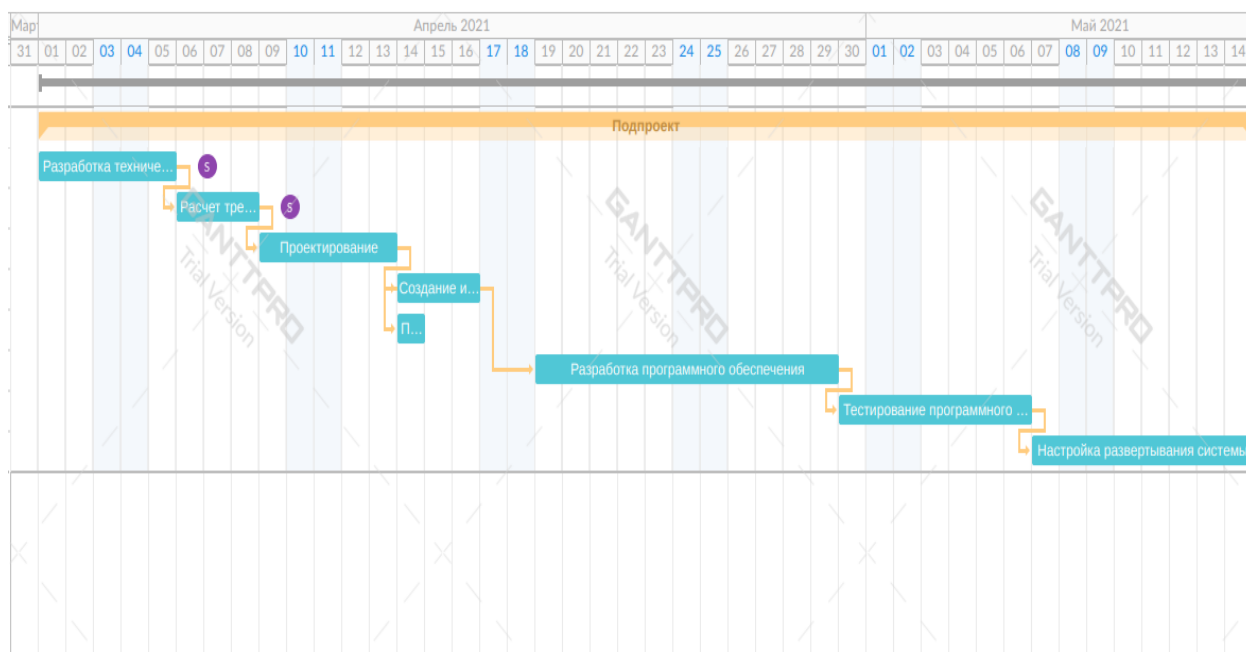


Рис. 4.2 – диаграмма Ганта.

4.11. Финансовый план

Приводится план прибылей и убытков, данные берутся из плана продаж и разделов производственного плана. План составляется на весь период проекта (3 года).

Таблица 4.18 – План прибылей и убытков за 3 года

Показатели	Год			Всего
	I	II	III	
1. Выручка-нетто (без учета НДС) от реализации	9252000	9252000	9252000	27756000
2. Прямые производственные затраты	1116000	1116000	1116000	3348000
2.1. Прямые материальные затраты	0	0	0	0
2.2. Прямые затраты на оплату труда	1116000	1116000	1116000	3348000
3. Общепроизводственные затраты	0	0	0	0
4. Общехозяйственные, управленческие и коммерческие затраты	7287000	7287000	7287000	21861000
5. Прибыль от продаж	849000	849000	849000	2547000
6. Налог на прибыль (20%)	169800	169800	169800	509400
7. Чистая (нераспределенная) прибыль	679200	679200	679200	2037600

По полученным данным видно, что очень много расходов идет на общехозяйственные, управленческие и коммерческие затраты. Они являются

ключевым цен образовательным фактором стоимости лицензии. Ожидается, что в год будет продаваться примерно около 5000 копий. Чтобы получать большую прибыль, требуется сократить затраты, либо выпускать больше копий. Альтернативным вариантом послужит введение новой услуги, которая бы позволила уменьшить стоимость главного продукта – лицензии. Например, можно предоставлять услуги по настройке и размещению ПО. Другой вариант – уменьшить срок действия лицензии. Это позволит снизить цену и вынудит покупателя продлевать её чаще.

Показатели экономического эффекта и эффективности инвестиций

Простой срок окупаемости рассчитывается по формуле:

$$PP = IC / CF,$$

где

PP (Pay-Back Period) – простой срок окупаемости, выраженный в годах/месяцах;

IC (Invest Capital) – сумма первоначальных инвестиций;

CF (Cash Flow) – ожидаемый среднегодовой (среднемесячный) денежный поток.

Для текущего проекта срок окупаемости рассчитывается срок окупаемости:

$$PP = \frac{157000}{70000} = 2,24 \approx 3 \text{ мес.}$$

Рентабельность капитала дает информацию о том, эффективно ли работают инвестиции.

$$R_{СК} = \text{Пр} / \text{СК} * 100,$$

где:

$R_{СК}$ — рентабельность собственного капитала;

Пр — чистая прибыль (рентабельность собственного капитала считают только по чистой прибыли);

СК — средняя величина собственного капитала за расчетный период.

$$R_{СК} = \frac{679200}{9252000} = 0.073$$

Годовая рентабельность у продукта крайне мала.

NPV показатель рассчитывается по формуле:

$$NPV = -IC + \sum_{t=0}^N CF^t / (1 + i)^t$$

Таблица 4.19 – Расчет NPV показателя

Величина инвестиций, тыс.руб.	157000
Ставка дисконтирования,	17%
Величина поступлений в 1-й год, тыс.руб.	679200
Величина поступлений во 2-й год, тыс.руб.	679200
Величина поступлений во 3-й год, тыс.руб.	679200
Величина поступлений во 4-й год, тыс.руб.	679200
Величина поступлений во 5-й год, тыс.руб.	679200

Проект получился дорогой и на самом деле окупит себя только в том случае, если запланированный объем продаж лицензий будет выполняться.

Отчет о движении денежных средств

Производится расчет данных о поступлении и оттоке средств в проект. Выполняется проверка на неотрицательность средств в конце каждого периода.

Предполагается, что кредит на выполнение проекта использоваться не будет, оплата материалов в этот проект не включена, оборудование не продается по окончании проекта, потому что выполняется только на имеющихся ресурсах.

Таблица 4.20 – Отчет о движении денежных средств

Показатели	Годы		
	1	2	3
Внесение собственных средств	0	0	0
Получение кредита	0	0	0
Поступление выручки от продаж от продаж (с НДС)	9252000	9252000	9252000
Инвестиционные затраты:	-157000		
оплата стоимости оборудования			
Оплата труда работающих	-8403000	-8403000	-8403000
Оплата материалов	0	0	0
Проч. издержки	-16000	-16000	-16000
Оплата налогов (взносы в фонды, НДС, налог на прибыль)	-169800	-169800	-169800
Возврат кредита	0	0	0
Продажа оборудования по окончании проекта	0	0	0
Итого наличие денег	506200	663200	663200

4.12. Стратегия финансирования

Финансирование проекта будет выполняться за счет заработанного капитала компании. Часть прибыли от основного бизнеса уходит в фонд развития и поддержки новых проектов компании. Кредит на развитие этого проекта не предусмотрен.

4.13. Оценка риска проекта

Для выявления основных факторов риска был произведен анализ потенциальных проблем. Риск проекта — это неопределенное событие или условие, которое положительно или отрицательно влияет на цели проекта. Риски могут вызывать финансовые потери.

Таблица 4.21 – Риски проекта

Рисковая ситуация	Вероятность возникновения	Влияние на проект	Возможные способы реагирования на риск (меры снижения риска)
Возникновение на рынке аналогичного решения, которое оказывается доступнее	Высокая	Делает проект не окупаемым. Снижает спрос на покупку лицензии	Предоставить дополнительные услуги, которые бы могли способствовать сделать выбор в пользу разрабатываемого продукта
Используемые библиотеки языка программирования изменят лицензию	Низкая	Это не позволит использовать библиотеку в коммерческих целях	Производить обзор библиотек
Продукт проявит неэффективность в работе с данными	Высокая	Проект потеряет свою значимость и не окупится.	Произвести более точные исследования. Повторить опыты на данных разных зданий. Обеспечить надежность и достоверность проводимых результатов.
Продукт будет сложно реализовать на международном уровне	Средняя	Снизится число продаж, что повлияет на прибыль.	Произвести дополнительное исследование в области поддержки продукта. Рассмотреть варианты продажи программы на различных языках.
Разработка программного продукта займет больше времени, чем ожидается.	Средняя	Увеличится время выхода продукта на рынок, уменьшится число продаваемых копий	Произвести аудит и построить план создания минимально жизнеспособного продукта.

Сценарии развития проекта:

- Оптимистический. Продукт внесет вклад в развитие науки о данных. Появятся новые возможности по анализу и предотвращению внештатных ситуаций. Программа окупится и будет востребована на рынке.
- Пессимистический. Разработанное решение окажется не эффективным – либо по качеству результата, либо по сложности выполнения расчетов. Продукт окажется дорогим и может уступать

примитивным методам оценки данных. Фактическое количество проданных лицензий станет меньше, чем ожидаемое.

- Наиболее вероятный сценарий. Проект найдет свое применение внутри компании и будет использоваться как проприетарное программное обеспечение вдобавок к системе контроля обогрева здания. Это улучшит функционал и увеличит безопасность предоставляемых услуг.

Программный продукт несмотря на свою инновационность имеет большие риски оказаться не продаваемым. Выявлено много тонкостей и особенностей на счет его продажи и стоимости. Однако, вклад, который вносит его реализация, очень хорошо сказывается на делах компании.

ЗАКЛЮЧЕНИЕ

В рамках данной выпускной квалификационной работы достигнуты все поставленные цели, а именно: исследованы принципы хранения time-series данных, получаемых от контроллеров, обработаны большие данные, выявлены паттерны поведения показателей температур, апробирован подход к аннотированию данных.

Были получены практические навыки обработки больших данных и взаимодействия с Linux серверами, разработки программного обеспечения. Благодаря модульной архитектуре, спроектированный набор программ в дальнейшем будет перенесен на отдельный сервер с целью полной автоматизации.

Внедрение разработанного программного обеспечения позволит компании K3D сделать свой продукт надежнее и поспособствует расширению бизнеса. Как следствие этому, выявление паттернов позволит определять характер изменения данных и идентифицировать проблемы до их возникновения в режиме реального времени.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Detection and identification of anomalies in wireless mesh networks using Principal Component Analysis (PCA) / сост.: Zainab R. Zaidi, Sara Hakami, Bjorn Landfeldt, and Tim Moors. Изд-во: World Scientific Journal of Interconnection Networks, 24 с.
2. Метод главных компонент (Principal component analysis) [Электронный ресурс]. URL: <https://wiki.loginom.ru/articles/principal-component-analysis.html> (дата обращения 15.04.2021)
3. Алгоритм триангуляции Делоне методом заметающей прямой [Электронный ресурс]. URL: <https://habr.com/ru/post/445048/> (дата обращения 15.04.2021)
4. Скворцов А.В. Триангуляция Делоне и её применение. – Томск: Изд-во Том. ун-та, 2002. – 128 с. ISBN 5-7511-1501-5
5. Combined Approach to Anomaly Detection in Wireless Sensor Networks on Example of Water Management System / сост.: Alexey Meleshko, Vasily Desnitsky, Igor Kotenko, Evgenia Novikova, Anton Shulepov. Изд-во: Saint Petersburg Electrotechnical University ("LETI"), St. Petersburg, Russia, 4 с.
6. The Application of Delaunay Triangulation to Face Recognition / сост.: John Y. Chiang, R. C. Wang, Yun-Lung Chang. Изд-во: Department of Applied Mathematics National Sun Yat-Sen University Kaohsiung, Taiwan 80424.
7. Global Big data market forecast (Электронный ресурс). URL: <https://www.statista.com/statistics/254266/global-big-data-market-forecast/> (дата обращения 16.04.2021)

ПРИЛОЖЕНИЕ А

Листинг кода, выполняющего расчеты

```
from multiprocessing import Pool
from multiprocessing.pool import ThreadPool
from .main import *
from enum import IntEnum
import hashlib
import pickle
import time
from scipy.spatial import Delaunay

# graphics
import matplotlib.pyplot as plt

def triangle_area(points):
    """Calculates area of triangle by 3 points.

    Args:
        points (list): List of 2d points: [point1, point2, point3]

    Returns:
        float: Area of triangle.
    """

    x1 = points[0][0]
    x2 = points[1][0]
    x3 = points[2][0]
    y1 = points[0][1]
    y2 = points[1][1]
    y3 = points[2][1]
    a = np.array([[x1 - x3, y1 - y3], [x2 - x3, y2 - y3]])
    return np.abs(0.5 * np.linalg.det(a))

class CalcMethod(IntEnum):
    DELAUNAY = 0 # Метод Делоне.
    SUBSEQUENT = 1 # Метод последовательной триангуляции.

class Graphy:
    """It used to visualise PCA data."""

    @staticmethod
    def scatter_areas(areas):
        return plt.scatter(areas.index, areas)

class PCAMetric:
    """Custom class encapsulates dataframe with pca and adds some features."""
```

```

def __init__(self):
    pass

def __init__(self, df):
    self.df = df

def by_hours(self):
    return [PCAMetric(g) for n, g in self.df.groupby(pd.Grouper(freq="H"))]

def by_quadro_hours(self):
    return [PCAMetric(g) for n, g in self.df.groupby(pd.Grouper(freq="4H"))]

def by_days(self):
    return [PCAMetric(g) for n, g in self.df.groupby(pd.Grouper(freq="4H"))]

def by_weeks(self):
    """Returns list of dataframes, grouped by week.

    Returns:
        list: each is pd.DataFrame.
    """
    return [PCAMetric(g) for n, g in self.df.groupby(pd.Grouper(freq="W"))]

def by_months(self):
    return [PCAMetric(g) for n, g in self.df.groupby(pd.Grouper(freq="M"))]

def total_delaunay_area(self):
    """Splits data into delaunay triangles. Calculates area of each, return total
    area of the resulting figure."""
    # Делоне объект
    tri = Delaunay(self.df)
    # tri.simplices - это такой массив, который содержит 3 индекса, исходного массива, которые формируют треугольник.
    areas_list = []

    # Точки треугольников. Каждый треугольник - ((x1, y1), (x2, y2), (x3, y3))
    triangle_points = [
        (
            self.df.iloc[simplice[0]][0], self.df.iloc[simplice[0]][1],
            self.df.iloc[simplice[1]][0], self.df.iloc[simplice[1]][1],
            self.df.iloc[simplice[2]][0], self.df.iloc[simplice[2]][1],
        )
    ]

```

```

        )
        for simplice in tri.simplices
    ]

    # Треды нифига не ускоряют, ну ладно
    with ThreadPool(4) as pool:
        areas_list = pool.map(triangle_area, trinangle_points)

    # Площадь всех треугольников в одной серии
    return sum(areas_list)

class Driver:
    HOME_DIR = pathlib.Path("/home/urukov") # Home directory.
    CACHE_DIR = HOME_DIR / ".utils_cache"

    def __init__(self):
        self._method: CalcMethod = None
        self._import_dir: pathlib.Path = None
        self._export_dir: pathlib.Path = None
        self._data_grid = self.setup_data_grid()
        # Содержит массив PCAMetric.
        self.pca_results: list = []

        # сюда накидаем кэш. Все вычисления будем сохранять, чтоб быст-
        # рее перезапускать
        os.makedirs(self.CACHE_DIR, exist_ok=True)

        print("🔌 Shulepov driver connected!")

    def main(self):
        """
        Calls main() function from this file.

        """
        assert self._method is not None, "⊗ Метод не указан!"
        print(f"🔌 Выбран метод {self._method}")
        main(
            {
                "method": self._method,
                "data_grid": self._data_grid,
                "import_dir": self._import_dir,
                "export_dir": self._export_dir,
            }
        )

    @staticmethod
    def _remove_outliers(df):
        """Remove outliers from dataset. It can be commented if need."""
        return df.loc[((df > 5) & (df < 35)).all(axis=1)]

    def pca_job(self, fname):
        """Загружает 1 датафрейм и вычисляет PCA."""

```

```

start = time.monotonic()

# Эта переменная станет True, если найдется кэш
from_cache = False
# Сперва возьмем хеш файла и тут же откроем датасет
with open(fname["path"], "rb") as f:
    file_bytes = f.read() # read file as bytes
    hashsum = hashlib.md5(file_bytes).hexdigest()

if os.path.exists(self.CACHE_DIR / hashsum):
    from_cache = True
    pca_df = pd.read_pickle(self.CACHE_DIR / hashsum)
else:
    df = pd.read_pickle(fname["path"])

    # Здесь убираются крайние значения. Можно закомментиро-
    вать, если надо.
    df = self._remove_outliers(df)

    pca_result = PCA(n_components=2).fit_transform(df)
    pca_df = pd.DataFrame(pca_result, index=df.index.copy())

    pca_df.to_pickle(self.CACHE_DIR / hashsum)

finish = time.monotonic() - start
print(f'🚩 [{fin-
ish:.2f}s] Готово! Из кэша = {from_cache}, {fname["name"]}')
return {fname["name"]: pca_df}

def prepare_pca(self):
    """Подготовим двумерные данные, полученные по PCA.
    Загоним их в датафреймы и закешируем."""

    # Зачитаем файлы
    fnames = [
        {"path": f.path, "name": f.name} for f in os.scandir(self._im-
port_dir)
    ]
    start = time.monotonic()

    # self.pca_job(fnames[0])
    with Pool(processes=4) as pool:
        self.pca_results = pool.map(self.pca_job, fnames)

    # Конвертим [{zone:pca}, {zone:pca}] в {zone:pca, zone:pca}
    tmp = dict()
    for i in self.pca_results:
        k, v = [(k, v) for k, v in i.items()][0]
        tmp[k] = PCAMetric(v)
    self.pca_results = tmp

    finish = time.monotonic() - start
    print(

```

```

        f"🚧🚧🚧 [{finish:.2f}s] PCA рассчитано для {len(self.pca_results)} датафреймов."
    )

    @property
    def import_dir(self):
        return self._import_dir

    @import_dir.setter
    def import_dir(self, value: pathlib.Path):
        self._import_dir = value

    @property
    def method(self):
        return self._method

    @method.setter
    def method(self, value: CalcMethod):
        self._method = value

    @property
    def export_dir(self):
        return self._export_dir

    @export_dir.setter
    def export_dir(self, value: pathlib.Path):
        self._export_dir = value

    def setup_data_grid(self):
        return [
            ["All", [(1, 13)], [(0, 7200)]]
        ]

```

Datasets.py

```

import os
import pandas as pd
import time

from .building import Building
from .constants import *

def funny_apt_str(units):
    s = ""
    for unit in units:
        s += f"🏠 {unit['unit_number']}, "
    return s[:-2] + "."

class Datasets:
    def __init__(self, building: Building):
        self.building = building

    def _merge_temps(self, unit_refs, time_from: str, time_to: str):

```

```

# Обрабатываем первый датасет и к нему начинаем прилеплять другие
first_unit = unit_refs[0]
# Ограничиваем по временным промежуткам, потому что биг дата все-
таки
time_from = pd.Timestamp(time_from, tz="America/New_York")
time_to = pd.Timestamp(time_to, tz="America/New_York")
time_filter = lambda index: (time_from <= index) & (in-
dex <= time_to)

# Открываем файл, устанавливаем ключевое поле, мерджим средним дуб-
ликаты, обрезаем по времени.
filepath = BIN_DIR / self.building.name / "%s.csv.pkl.gz"
column_name = "unit_%s"
ds = pd.read_pickle(str(filepath) % first_unit["name_opc"])
ds.rename(
    columns={"value": column_name % first_unit["hmi_unit_num-
ber"]}, inplace=True
)
ds.set_index("time", inplace=True)
ds = ds[time_filter(ds.index)]
ds = ds.groupby(ds.index).mean()

for i, ur in enumerate(unit_refs[1:]):
    # Открываем файл, устанавливаем ключевое поле, мерджим сред-
    ним дубликаты, обрезаем по времени.
    try:
        iter_ds = pd.read_pickle(str(filepath) % ur["name_opc"])
    except FileNotFoundError:
        print(f"☹ Файл не найден - {ur['name_opc']}.")
        continue
    iter_ds.rename(
        columns={"value": column_name % ur["hmi_unit_num-
ber"]}, inplace=True
    )
    iter_ds.set_index("time", inplace=True)
    iter_ds = iter_ds[time_filter(iter_ds.index)]
    iter_ds = iter_ds.groupby(iter_ds.index).mean()

    # Склеиваем с основным датасетом.
    ds = pd.merge(ds, iter_ds, how="outer", left_in-
dex=True, right_index=True)
    progress = (i + 1) * 100 / len(unit_refs[1:])
    print(f"Progress: {progress:.2f}% ", end="\x1b[1K\r")

# У датасета остаются пустые поля от OUTER объединения, заполним их.
ds.fillna(method="ffill", inplace=True)
ds.fillna(method="bfill", inplace=True)
return ds

def create_temperature_da-
taset_by_zones(self, time_from: str, time_to: str):

```

```

        """Создает несколько датасетов на основе building, сгруппирован-
ных по
        зонам."""
        zones: set = self.building.zones()
        output_dir = PREPARED_DIR / self.build-
ing.name / f"{time_from}_{time_to}"
        os.makedirs(output_dir, exist_ok=True)

        print("🌀 Запущен процесс создания датасетов с температурными пока-
зателями.")
        print(
            f"Здание поделено на {len(zones)} зон(ы). Выделяются следую-
щие зоны {zones}"
        )
        print(f"Будет создано столько же файлов в директории. {output_dir}")

        for zone in zones:
            dataset_name = "zone_%s.pkl.gz" % zone
            units = self.building.units(zone_n=zone)
            print(f"📄 Создаем датасет {dataset_name}", end="\x1b[1K\r")
            start_time = time.monotonic()
            ds = self._merge_temps(units, time_from, time_to)
            ds.to_pickle(output_dir / dataset_name)
            print(
                f"🕒 Создали и сохранили {dataset_name} за {time.моно-
tonic()-start_time:.2f} сек"
            )
        print("🏁 Все, завершили.")

```

Delaunay_zones_triangle_sum.py

```

import pandas as pd
import os
import utils # Подключаем наши python скрипты-ути-
литы, в том числе и код Антона
from multiprocessing.pool import ThreadPool

DELAUNAY_DIR_NAME = "delaunay_autumn"

def selector(func):
    def wrapper(*args, **kwargs):
        print("Функция-обертка")
        # args это такой дикт, который содержит нужные поля для выполне-
ния функции
        args = kwargs.get("args")

        # print(args)
        # print(kwargs)

        # print(args["pca_result"])
        # time_from = pd.Timestamp("2019-09-01", tz="America/New_York")

```



```

        # time_to = pd.Timestamp("2019-10-01", tz="America/New_York")
        # time_filter = lambda index: (time_from <= index) & (in-
dex <= time_to)

        # args["pca_result"].df = args["pca_result"].df[time_filter(ds.in-
dex)]
        func(*args, **kwargs)

    return wrapper

# @selector
def f(args):
    code = args["code"]

    # хотел написать вранпер не получилось
    pca_result = args["pca_result"]

    time_from = pd.Timestamp("2019-09-15", tz="America/New_York")
    time_to = pd.Timestamp("2019-11-15", tz="America/New_York")
    time_filter = lambda index: (time_from <= index) & (index <= time_to)
    pca_result.df = pca_result.df[time_filter(pca_result.df.index)]

    print(f"🔍 {code} Началась обработка.")
    areas_by_time = []
    result_period = pca_result.by_days()
    n_of_time_intervals = len(result_period)
    done = 0
    for pca_metric in result_period:
        if pca_metric.df.empty:
            from_date = None
            to_date = None
            area = 0
        else:
            from_date = pca_metric.df.index[0]
            to_date = pca_metric.df.index[-1]
            area = pca_metric.total_delaunay_area()
            areas_by_time.append([from_date, to_date, area])
            done += 1
        print(
            f"🌀 [{done}/{n_of_time_inter-
vals}] ({code}) Площадь фигуры для периода {from_date} рассчитана."
        )
    print(f"💾 [{done}/{n_of_time_intervals}] ({code}) Сохранение файла.")
    delaunay_areas = pd.DataFrame(areas_by_time)
    # Создадим папку
    os.makedirs(driver.export_dir / DELAUNAY_DIR_NAME, exist_ok=True)
    delaunay_areas.to_pickle(
        driver.export_dir / DELAUNAY_DIR_NAME / f"{code}_delaunay.pkl"
    )
    print(f"✅ [{done}/{n_of_time_intervals}] ({code}) Готово.")

driver = utils.shulepov.Driver()

```

```

# Укажем метод здесь
driver.method = utils.shulepov.CalcMethod.DELAUNAY
# Возьмем список файлов здесь
driver.import_dir = utils.PREPARED_DIR / "park" / "2019-01-01_2020-12-31"
driver.export_dir = utils.PROCESSED_DIR / "park" / "2019-01-01_2020-12-31"
driver.method, driver.import_dir, driver.export_dir

# Подготовим PCA
driver.prepare_pca()

with ThreadPool(8) as pool:
    # вот тут надо было выбрать отдельные зоны
    params = [{"code": k, "pca_result": v} for k, v in driver.pca_re-
sults.items()]
    less_params = filter(
        lambda x: x["code"]
        in ["zone_9.pkl.gz", "zone_10.pkl.gz", "zone_11.pkl.gz", "zone_12.pk
l.gz"],
        params,
    )
    pool.map(f, less_params)

building.py
import json

from .constants import *

class Building:
    """Building."""

    def __init__(self, filename):
        self.filename = filename
        self.name = filename.replace(".json", "")
        self.working_dir = SRC_DIR / self.name
        self.dict = {}
        self.variables = []
        self.references = []
        self.load()

    def load(self):

        with open(SRC_DIR / JSON_DIR / self.filename) as f:
            self.dict = json.load(f)

            global_variables = self.dict["building"]["heating_management_sys-
tem"]["plc"][
                "global_variables"
            ]
            for k, v in global_variables.items():
                self.variables += v["variable"]
            for va in self.variables:
                for ref in va["refs"]["ref"]:

```

```

        self.references.append(ref)

def floors(self):
    """Returns set of available floors."""
    units = self.dict["building"]["configuration"]["units"]["unit"]
    return set([u["floor_number"] for u in units])

def zones(self):
    """Returns set of available zones."""
    units = self.dict["building"]["configuration"]["units"]["unit"]
    return set([u["zone_number"] for u in units])

def units(self, floor_n=None, zone_n=None):
    """Returns list of references related to unit temps.

    NOTE: you can only use floor_n or zone_n. Using both params is not
    implemented.
    """
    res = [i for i in self.references if i["flag"] == "UNIT_TEMP"]

    if floor_n is None and zone_n is None:
        return res
    elif floor_n is not None:
        unit_ids_on_floor = list(
            filter(
                lambda x: x["floor_number"] == floor_n,
                self.dict["building"]["configuration"]["units"]["unit"],
            )
        )
        unit_ids_on_floor = [i["unit_id"] for i in unit_ids_on_floor]
        res = list(filter(
            lambda x: x["unit_id"] in unit_ids_on_floor, res))
        return res
    elif zone_n is not None:
        unit_ids_on_zone = list(
            filter(
                lambda x: x["zone_number"] == zone_n,
                self.dict["building"]["configuration"]["units"]["unit"],
            )
        )
        unit_ids_on_zone = [i["unit_id"] for i in unit_ids_on_zone]
        res = list(filter(
            lambda x: x["unit_id"] in unit_ids_on_zone, res))
        return res

def get_csv_path(self, name_opc):
    return self.working_dir / f"{name_opc}.csv"

```