

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ  
Федеральное государственное автономное образовательное учреждение  
высшего образования  
«ТЮМЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ НАУК  
Кафедра программного обеспечения

РЕКОМЕНДОВАНО К ЗАЩИТЕ В ГЭК

Заведующий кафедрой

К.т.н., доцент



М. С. Воробьева

25 июня 2022 г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА  
магистерская диссертация

СИСТЕМА КОМПЬЮТЕРНОГО МОДЕЛИРОВАНИЯ ДИНАМИКИ  
ИЗМЕНЕНИЯ ТЕМПЕРАТУРЫ ВОЗДУХА В ПОМЕЩЕНИЯХ УЧЕБНО-  
ЛАБОРАТОРНОГО КОРПУСА ПОД ВЛИЯНИЕМ ТЕПЛОВОГО РЕЖИМА

02.04.03 Математическое обеспечение и администрирование информационных систем

Магистерская программа «Разработка технологий Интернета вещей и больших данных»

Выполнили работу  
(групповой проект)  
студенты 2 курса  
очной формы обучения



Мурашкина  
Елизавета Александровна

Таут Павел Владимирович

Научный руководитель  
Старший преподаватель  
Рецензент  
Директор  
ООО "Энергосбережение  
Западной Сибири"



Ромазанов  
Артур Ринатович



Журавлев Сергей Юрьевич

Тюмень  
2022 год

## ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ.....	4
ГЛАВА 1. ТЕОРЕТИЧЕСКИЙ АНАЛИЗ.....	6
1.1. АНАЛИЗ ФАКТОРОВ ВЛИЯНИЯ НА ТЕПЛОВОЙ РЕЖИМ ПОМЕЩЕНИЯ .....	6
1.1.1. ПОНЯТИЕ ТЕПЛООВОГО РЕЖИМА.....	6
1.1.2. ВЛИЯНИЕ ВЛАЖНОСТИ НА ТЕПЛОВОЙ РЕЖИМ .....	8
1.1.3. ВЛИЯНИЕ ВОЗДУШНЫХ ПОТОКОВ НА ТЕПЛОВОЙ РЕЖИМ .....	8
1.1.4. ВЛИЯНИЕ ИНСОЛЯЦИИ НА ТЕПЛОВОЙ РЕЖИМ .....	11
1.2. СРАВНИТЕЛЬНЫЙ АНАЛИЗ И ВЫБОР ТЕХНОЛОГИЙ.....	12
1.2.1. СРАВНЕНИЕ ЯЗЫКОВ ПРОГРАММИРОВАНИЯ .....	12
1.2.2. СРАВНЕНИЕ FRONTEND FRAMEWORKS.....	15
1.2.3. СРАВНЕНИЕ БИБЛИОТЕК ДЛЯ ОБРАБОТКИ ЗАПРОСОВ .....	16
1.3.3. ИСПОЛЬЗУЕМЫЕ БИБЛИОТЕКИ.....	17
1.3. КЛАСТЕРИЗАЦИЯ ВРЕМЕННЫХ РЯДОВ .....	18
1.4. ПОИСК АНОМАЛИЙ ВО ВРЕМЕННЫХ РЯДАХ .....	20
1.4.1. МЕТОДЫ ТРАНСФОРМАЦИИ ДАННЫХ.....	20
1.4.2. МЕТОДЫ ОБНАРУЖЕНИЯ .....	21
ВЫВОДЫ К ГЛАВЕ 1 .....	24
ГЛАВА 2. ПРОГРАММНАЯ РЕАЛИЗАЦИЯ .....	26
2.1. ИНТЕРПРЕТАЦИЯ ДАННЫХ.....	26
2.1.1. ИСТОЧНИКИ ДАННЫХ .....	26
2.1.2. ОПИСАНИЕ ДОСТУПНЫХ ДАННЫХ .....	27

2.1.3. ПРЕДВАРИТЕЛЬНАЯ ОБРАБОТКА ДАННЫХ.....	29
2.1.4. ПЕРВИЧНЫЙ АНАЛИЗ ДАННЫХ .....	30
2.2. КЛАСТЕРИЗАЦИЯ ПОМЕЩЕНИЙ .....	52
2.3. ПОИСК АНОМАЛИЙ .....	57
2.3.1 AGGREGATION ПОДХОД ПОИСКА АНОМАЛИЙ.....	57
2.3.2 PREDICION-BASED ПОДХОД ПОИСКА АНОМАЛИЙ.....	60
2.4. ПОСТРОЕНИЕ МОДЕЛЕЙ ДЛЯ КЛАСТЕРОВ .....	65
2.5. СРАВНЕНИЕ МОДЕЛЕЙ.....	66
2.6. ПОИСК АНОМАЛИЙ ДЛЯ КЛАСТЕРНЫХ МОДЕЛЕЙ.....	72
2.7 ПРИЧИНЫ АНОМАЛИЙ .....	73
2.7.1 ОТКЛЮЧЕНИЕ БАТАРЕЙ .....	73
2.7.2 ПРОВЕТРИВАНИЯ.....	75
2.8. СИСТЕМА МОНИТОРИНГА ТЕМПЕРАТУР.....	79
2.8.1. РАЗРАБОТКА КЛИЕНТСКОЙ ЧАСТИ СИСТЕМЫ.....	79
2.8.2. РАЗРАБОТКА СЕРВЕРНОЙ ЧАСТИ СИСТЕМЫ.....	84
ВЫВОДЫ К ГЛАВЕ 2 .....	87
ЗАКЛЮЧЕНИЕ .....	90
СПИСОК ЛИТЕРАТУРЫ .....	92
ПРИЛОЖЕНИЯ 1-5 .....	95

## ВВЕДЕНИЕ

Согласно нормативу [ГОСТ 30494–96] оптимальная температура в помещениях для отдыха и учебных занятий колеблется от 20 до 22 градусов, а допустимая температура – от 18 до 24. На создание комфортной температуры в общественном здании требуется большое количество ресурсов. На данный момент расписание большинства систем отоплений не учитывает влияние внешних погодных факторов и внутренних факторов. Исследование влияния этих факторов позволит настроить режим системы отопления, позволяющий использовать тепловые ресурсы более эффективно.

Таким образом, обращение к **проблеме** повышения эффективности расхода теплоэнергии определило тему данной работы «Разработка системы мониторинга температуры воздуха в помещениях». Эта проблема наиболее значительна для климатических условий России, где продолжительность отопительного периода составляет для большинства регионов более 200 суток. На выработку тепловой энергии расходуется более 60% добываемого топлива, а на отопление зданий уходит более трети тепловой энергии [Ю. И. Толстова, Р. Н. Шумилов 5–6 с.]. Решение данной проблемы позволит тратить общественным организациям меньше ресурсов на отопление зданий.

**Целью** данной работы является исследование таких внешних климатических факторов влияния, как инсоляция и ветер, и таких внутренних факторов, как проветривания и количество людей, на тепловой режим общественного здания, что позволит скорректировать режим работы отопительных систем с учетом неучтенных ранее факторов.

Для реализации системы мониторинга необходимо решить следующие задачи:

Проект был разделен на **4 этапа**, для каждого из этапов были поставлены следующие **задачи**:

1. Интерпретация данных:

- 1.1. Провести анализ климатических факторов, влияющих на температуру в помещениях.

- 1.2. Провести сравнительный анализ технологий, используемых для анализа данных, и выбрать подходящие.
- 1.3. Определить источники данных.
- 1.4. Собрать и предобработать данные из источников.
- 1.5. Провести анализ данных.
2. Построение модели кластеризации кабинетов.
  - 2.1. Изучить подходящие алгоритмы кластеризации временных рядов.
  - 2.2. Построить модель кластеризации помещений.
  - 2.3. Проанализировать результат.
3. Разработка алгоритма поиска аномалий.
  - 3.1. Изучить подходы поиска аномалий во временных рядах.
  - 3.2. Написать алгоритмы поиска аномалий температур.
  - 3.3. Проанализировать результат.
  - 3.4. Написать алгоритм определения причин аномалий.
4. Разработать систему мониторинга.
  - 4.1. Провести сравнительный анализ технологий, используемых для клиентской и серверной частей приложения, и выбрать подходящие.
  - 4.2. Разработать клиентскую часть приложения.
  - 4.3. Разработать серверную часть приложения.



## ГЛАВА 1. ТЕОРЕТИЧЕСКИЙ АНАЛИЗ

### 1.1. АНАЛИЗ ФАКТОРОВ ВЛИЯНИЯ НА ТЕПЛОВОЙ РЕЖИМ ПОМЕЩЕНИЯ

#### 1.1.1. ПОНЯТИЕ ТЕПЛООВОГО РЕЖИМА

**Тепловым режимом здания** называется совокупность всех факторов и процессов, определяющих тепловую обстановку в его помещениях. Ограждающие конструкции изолируют помещения здания от внешней среды, за счет чего в них создается собственный микроклимат. Наружные ограждения защищают помещения от прямого воздействия атмосферных явлений. Помимо этого, системы кондиционирования микроклимата поддерживают некоторые параметры внутреннего климата [П. Н. Каменев, А. Н. Сканави и др., 8 с.].

Таким образом, климат внутри помещения формируется в результате влияния наружной среды (внешних климатических условий), других помещений здания, систем обеспечения микроклимата: систем отопления-охлаждения, вентиляции и кондиционирования, а также внутренних технологических процессов (людей, оборудования и т. д.) (Рисунок 1).



Рисунок 1. Факторы влияния на микроклимат помещения

Наружная среда воздействует через ограждающие конструкции посредством теплопередачи, воздухопроницаемости и внутренние связи между помещениями (перемещение потоков воздуха, теплообмен) [Ю. А. Кувшинков]. Под действием таких факторов, как солнечная радиация, ветра и разности внутренней и внешней температур помещение перегревается летом и теряет тепло зимой (Рисунок 2). Перепады давлений приводят к передвижению воздуха между сообщающимися помещениями. Атмосферные осадки, влаговыделения в помещениях, разность влажности внутреннего и наружного воздуха приводят к влагообмену через ограждения [Ю. И. Толстова, Р. Н. Шумилов 4 с.].

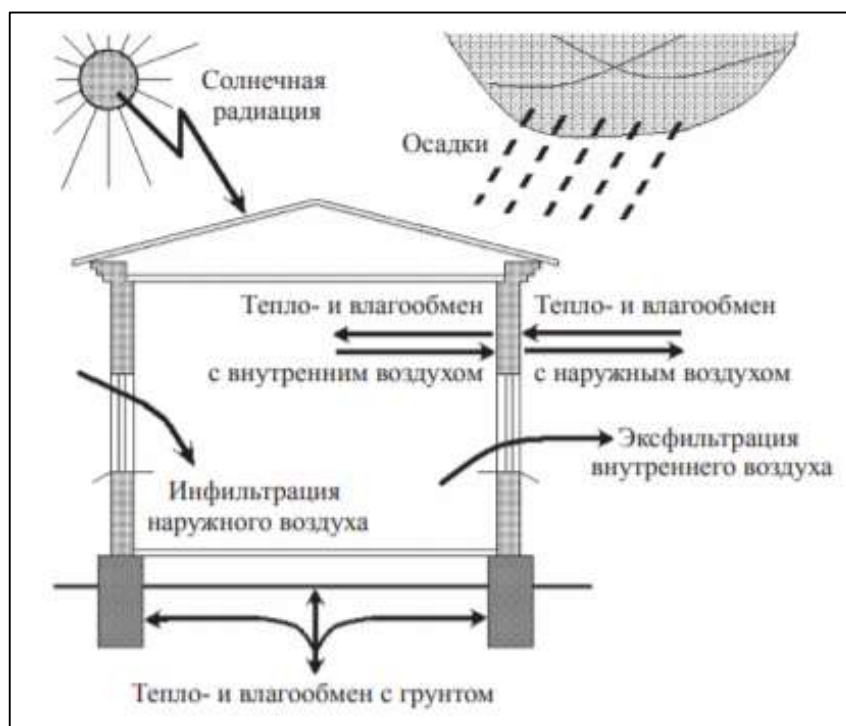


Рисунок 2. Схема внутренних и внешних воздействий на здание

На основании имеющихся данных в рамках научно-исследовательской работы рассматриваются следующие факторы влияния: внешняя среда и системы обеспечения микроклимата. В связи с пандемией 2020–2021 гг. имеются данные, на которые такой фактор как внутренний технологический процесс не оказывал существенного влияния. Рассмотрим внешние факторы более подробно.

### **1.1.2. ВЛИЯНИЕ ВЛАЖНОСТИ НА ТЕПЛОВОЙ РЕЖИМ**

Теплозащитные качества наружных ограждающих конструкций ухудшаются с повышением влажности, что может вызвать значительное увеличение эксплуатационных расходов на дополнительное отопление. Атмосферная влага попадает в ограждение с атмосферными осадками через стыки конструкций и в материал при косом направлении осадков. Влага из воздуха может конденсироваться как на внутренней поверхности ограждения, так и в его толще, что приводит к увлажнению материалов [Ю. И. Толстова, Р. Н. Шумилов 43с.].

При понижении температуры воздуха ниже температуры точки росы избыточное количество влаги будет конденсироваться, в положении точки росы. Оно может блуждать по толщине стены и зависит от толщины и типа материалов самой стены и утеплителя, от показателей относительной влажности и температуры на улице, и в самом помещении.

В настоящее время можно считать установленным, что увеличение весовой влажности строительных материалов на один процент приводит к повышению коэффициентов их теплопроводности на 4—5%. Наличие сырости в наружных ограждающих конструкциях зданий, и в первую очередь в стенах, может привести к значительному перерасходу топлива для отопления зданий [Н. М. Сенченко, 9 с.].

Таким образом, можно сделать вывод, что влажность имеет долгосрочное влияние на теплозащитные свойства ограждающих конструкций здания, но в краткосрочной перспективе его влияние пренебрежительно мало. Расчёт точки росы должен производиться при строительстве и проектировании зданий чтобы влияние влажности было минимизировано в процессе эксплуатации.

### **1.1.3. ВЛИЯНИЕ ВОЗДУШНЫХ ПОТОКОВ НА ТЕПЛОВОЙ РЕЖИМ**

Поступление наружного воздуха увеличивает теплопотери помещений и затраты на отопление. Количество проходящего воздуха зависит от перепада



давлений и воздухопроницаемости наружных ограждений, которая различна для заполнений оконных и дверных проемов, стыков конструкций и наружных стен.

Причинами возникновения перепада давлений с наружной и внутренней стороны наружного ограждения являются различие температур внутреннего и наружного воздуха (тепловое давление), воздействие ветра на здание (ветровое давление) и дисбаланс производительности систем приточной и вытяжной вентиляции. При этом ветровые давления с наветренной и заветренной стороны здания различны, что объясняется аэродинамикой обтекания здания.

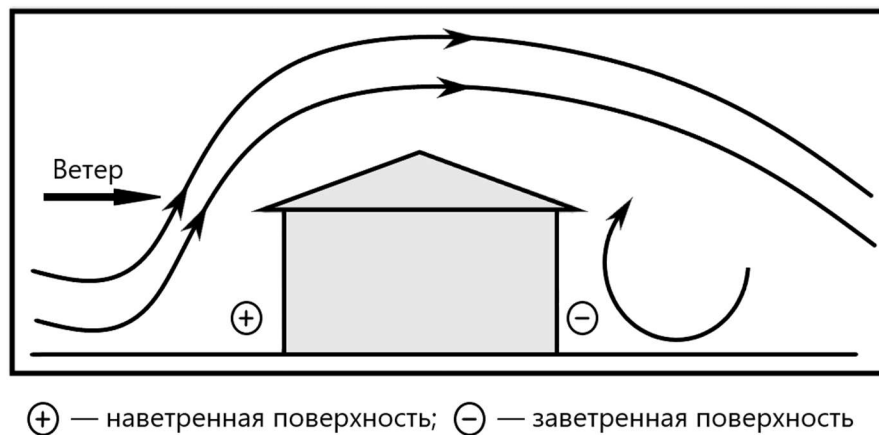


Рисунок 3. Ветровые давления

Дисбаланс производительности приточных и вытяжных систем вентиляции приводит к возникновению дополнительного перепада давлений. Как правило, это явление характерно для промышленных зданий, где воздух, забираемый из помещений на технологические нужды, зачастую не компенсируется соответствующим количеством приточного воздуха. Для жилых и общественных зданий это дополнительное давление не учитывается [Ю. И. Толстова, Р. Н. Шумилов 59 с.].

Влияние ветра на дома и жилую застройку сказывается довольно сильно. При приближении ветрового потока к зданию он начинает оказывать давление на ту часть фасада, которая обращена к нему (наветренная часть здания). В

результате с этой стороны здания образуется зона повышенного давления или ветровой подпор, при котором холодный воздух более интенсивно начинает проникать через стены, окна, стыки, щели внутрь жилых помещений, сильно их охлаждая (данное явление называется инфильтрацией) [Д. В. Дрозд, Ю. В. Елистратова и др] .

Среди трех факторов (ветровое давление, гравитационное давление, давление под действием системы вентиляции), определяющих инфильтрационный перепад давлений, ветровой напор наиболее значим. При равенстве температур внутреннего и наружного воздуха, если направление ветра перпендикулярно одной из стен, через наветренную стену происходит инфильтрация воздуха, а через три других и крышу – эксфильтрация. Если ветер направлен под углом – давление обращено на две стены, а эксфильтрация происходит через две других стены и крышу. Объем воздухообмена при инфильтрации и эксфильтрация зависят от перепада давлений и степени воздухопроницаемости ограждающей конструкции. В свою очередь, давление с внешней стороны стены зависит от силы и направления ветра [W. A. Anis].

Обогнув здание, ветровой поток продолжает свое движение, образуя с противоположной стороны (заветренная или подветренная часть здания) – зону пониженного давления или ветровой отсос. В результате этого возникает значительный перепад давлений с двух противоположных сторон дома, что способствует проникновению холодного воздуха в помещение, более интенсивному движению воздуха внутри дома от наветренной стороны к противоположной, сильные сквозняки, выветривающие тепло из комнат, понижение температуры внутреннего воздуха и резкое увеличение тепловых потерь зимой.

Увеличение скорости ветра при неизменной температуре наружного воздуха вызывает увеличение давления на наветренный фасад здания, в результате чего увеличивается теплопотери помещения, связанные с нагревом поступающего воздуха. Следует отметить, что скорость и направление ветра

оказывают более сильное воздействие на распределение воздушных потоков в системе вентиляции и на расходы инфильтрации, чем температура наружного воздуха. Изменение температуры наружного воздуха от  $-15^{\circ}\text{C}$  до  $-30^{\circ}\text{C}$  приводит к такому увеличению воздухообмена в квартире, как и увеличение скорости ветра от 3 до 3,6 м/с [Д. В. Дрозд, Ю. В. Елистратова и др].

Из всего вышесказанного можно сделать вывод, что на ветер и давление воздуха имеет непосредственное влияние на воздухообмен между помещением и внешней атмосферой, что приводит к изменению теплового режима внутри здания.

#### **1.1.4. ВЛИЯНИЕ ИНСОЛЯЦИИ НА ТЕПЛОВОЙ РЕЖИМ**

Инсоляцией (от латинского *in solo* – выставляю на солнце) называют облучение поверхности, пространства параллельным пучком лучей, поступающих с направления, в котором виден в данный момент времени центр солнечного диска. Инсоляционные процессы зависят от следующих факторов: географическая широта района расположения объекта, текущее состояние облачности в районе наблюдения, пространственная ориентация и высота над уровнем моря облучаемых поверхностей ограждающих конструкций, текущее положение солнца. Изменение вышеуказанных факторов приводит к значительной неравномерности распределения тепlopоступлений от солнечного излучения в течение суток в разные времена года [П. А. Стрижак, М. Н. Морозов].

Поглощение солнечного излучения ограждающим перекрытием зависит от цвета самой стены, которая постоянно отражает какую-то часть приходящего тепла. За счёт выбора цвета наружных стен можно ограничить или усилить теплоприток. Обычно при расчетах учитывается определенный коэффициент поглощения тепла, достигающий 0,9 для стены темного цвета, 0,7 для цвета серого и 0,5 для стены светлого цвета. Помимо этого, поступление тепла от солнечной радиации зависит от угла, под которым солнечные лучи падают на поверхность, ориентации поверхности по странам света и т. д.

В результате воздействия солнечного излучения повышается температура наружной поверхности, избыточное тепло поступает в помещение на протяжении некоторого времени, его количество зависит от характеристик ограждения. Чем больше массивность стены (вес на квадратный метр площади), тем больше времени необходимо для передачи тепла в помещение. Соответственно, очень легкая стена передает полученное от солнечного излучения тепло в помещение почти сразу. Такое явление очень важно при определении тепловых нагрузок в помещении.

Из вышесказанного можно сделать вывод, что: в зданиях из легких конструкций эффективные тепловые нагрузки являются повышенными и быстро изменяющимися; в зданиях с тяжелыми стенами тепловые нагрузки ниже по величине и изменяются на протяжении длительного времени.

Солнечное излучение также оказывает тепловое воздействие через остекление. Показатель получаемого тепла солнечного излучения может достигать 90%. Этот показатель зависит от типа остекления, оставшееся тепло от излучения при этом отражается. Доля тепловой нагрузки от солнечного излучения в общем балансе тепlopоступления чаще всего может достигать 50% для общественных и административных зданий. Обычно максимальная тепловая нагрузка достигается при максимальном уровне излучения. Интенсивность солнечного излучения зависит от широты местности, времени года и времени суток [В. А. Ананьев, Л. Н. Балужева и др.].

Согласно исследованиям [П. А. Стрижак, М. Н. Морозов] максимальный потенциал энергосбережения за счёт снижения потребления отопительных систем зафиксирован в помещении южной ориентации и составляет около 14,3%. В помещениях западной и восточной ориентации это число составляет 3,67% за счет компенсации влияния солнечного излучения.

## **1.2. СРАВНИТЕЛЬНЫЙ АНАЛИЗ И ВЫБОР ТЕХНОЛОГИЙ**

### **1.2.1. СРАВНЕНИЕ ЯЗЫКОВ ПРОГРАММИРОВАНИЯ**

В настоящее время объем данных растет экспоненциально, и чем больше информации, тем сложнее ее обрабатывать, поэтому одной из важнейших

сегодняшних задач является анализ данных. Многие языки программирования адаптируются для ее решения. Вот примеры самых популярных из них.

#### **1.2.1.1. PYTHON**

На сегодняшний день Python считается самым популярным языком для сферы науки о данных благодаря его обширным библиотекам и фреймворкам [C. D. Costa]. Данный язык имеет широкий инструментарий для визуализации данных, что позволяет наглядно представить информацию. Для проектов в сфере науки о данных и машинного обучения Python может предложить широкий выбор полезных библиотек – SciPy, NumPy, Matplotlib, Pandas. Для более сложных проектов в области глубокого обучения можно использовать такие Python-библиотеки, как Keras, Pytorch и TensorFlow [V. S. Khatri].

Для решения сложных задач на Python требуется писать сравнительно небольшие объёмы кода. Так как этот язык довольно гибкий и интерпретируемый, он позволяет вносить изменения в код "на ходу" и проверять результат исполнения кода в интерактивном режиме. Помимо всего вышесказанного стоит отметить универсальность языка программирования Python, он имеет функционал для web-разработки, разработки простых игр, СУБД, языков программирования и т. д.

#### **1.2.1.2. R**

Еще одним из самых распространенных языков программирования в области анализа данных является R, мультипарадигмальный интерпретируемый язык программирования для статистической обработки данных и работы с графикой [C. D. Costa]. Данный язык поддерживает широкий спектр статистических и численных методов, постоянно расширяется за счет библиотек. Имеются библиотеки для визуализации данных и анализа текста.

Базовые статистические методы реализованы в качестве стандартных функций, что значительно повышает скорость разработки. Но так как R является узкоспециализированным языком программирования, направленным

на работу со статистикой, то он больше подходит для анализа и визуализации данных, чем для их сбора, в особенности с применением web-технологий [B. Karakan].

### **1.2.1.3. JAVA SCRIPT**

Популярный многопарадигмальный и управляемый событиями язык сценариев JavaScript - один из лучших языков программирования для веб-разработки. С помощью данного языка разработчики могут создавать насыщенные и интерактивные веб-страницы, и именно это свойство JavaScript делает его отличным выбором для создания красивых визуализаций. Другие варианты использования JavaScript для науки о данных включают управление асинхронными задачами и обработку данных в реальном времени. Но количество и доступность библиотек анализа данных уступают библиотекам Python. JavaScript поддерживает различные современные библиотеки машинного обучения, такие как TensorFlow.js, Keras.js, Brain.js и DeepLearn.js. [C. D. Costa]. Недостатком является то, что наиболее популярный Node.js плохо подходит для обработки большого количества данных из-за слабой оптимизации потоков. Тем не менее у него есть альтернатива – Napa.js, которая помогает Node.js с многопоточностью. [C. Shrestha]

### **1.2.1.4. SCALA**

Высокую популярность имеет язык Scala. Это типизированный язык программирования, который включает функциональное и объектно-ориентированное программирование. Он в первую очередь нацелен на платформу JVM. Scala можно эффективно использовать со Spark для обработки больших объемов разрозненных данных. Базовая поддержка параллелизма делает этот язык идеальным выбором для создания высокопроизводительных сред Data Science [C. D. Costa]. Одним из недостатков Scala является высокий порог входа, т. е. этот язык довольно сложен в освоении.



## **1.2.2. СРАВНЕНИЕ FRONTEND FRAMEWORKS**

### **1.2.2.1. REACT**

React – UI библиотека с открытым исходным кодом. Подходит для несложных приложений с широким пользовательским интерфейсом. Данный фреймворк имеет богатую экосистему для управления DOM объектами, маршрутизации, создания шаблонов [Jeel Patel]. Имеет структуру React Native для мобильной разработки. Преимуществами являются возможность повторного использования компонентов, что уменьшает время разработки, декларативность, что сокращает код и делает его более понятным, широкий инструментарий, стабильный код обеспечивается движением данных в одном направлении. Но данный фреймворк не имеет документации, сам по себе прост в освоении, но могут возникнуть сложности с освоением расширения для синтаксиса языка JSX.

### **1.2.2.2. ANGULAR**

Angular прежде всего нацелен на разработку SPA-решений. Он также имеет свою экосистему, такую как маршрутизация и управление состоянием. Также имеет очень понятную архитектуру, все разбито на 3 компонента: шаблон, стили и логика [Jeel Patel]. В отличие от React имеет двухстороннюю привязку данных, что позволяет синхронизировать представление и модель данных и немедленно обновлять интерфейс при изменениях в модели, и наоборот. Angular намного сложнее в изучении, несмотря на наличие большого количества документации, а также тяжелее из-за своей сложности, что может негативно сказаться на динамических приложениях и потребуются оптимизация. Преимуществами данного языка являются упрощение программирования благодаря сервисам рефакторинга и навигации, большая экосистема, высокая производительность, возможность повторного использования компонентов, независимость компонентов упрощает тестирование.

### 1.2.2.3. VUE

На настоящее время Vue является одним из самых простых фреймворков, представляет собой смесь концепций React и Angular. Он универсален, имеет небольшой размер, вследствие чего имеет довольно высокую производительность, легко интегрируется с другими библиотеками и проектами [Jeel Patel]. Также как Angular использует двухкомпонентную привязку данных и компонентную архитектуру, также как React использует виртуальный DOM. Преимуществами также являются повторное использование компонентов, собственная экосистема, наличие обширной и подробной документации, простота изучения. Недостатками являются – небольшое сообщество, так как эта среда разработки относительно молода, из-за гибкости данного фреймворка могут возникать проблемы при интеграции в очень большие проекты.

## 1.2.3. СРАВНЕНИЕ БИБЛИОТЕК ДЛЯ ОБРАБОТКИ ЗАПРОСОВ

### 1.2.3.1 DJANGO

Django — это популярный, мощный фреймворк на языке Python. Django является высокоуровневым Python веб-фреймворк, который позволяет быстро создавать безопасные и поддерживаемые веб-сайты.

К его преимуществам относятся следующие качества:

- хорошая безопасность;
- быстрый;
- уходит немного времени на разработку крупных проектов;
- относительно прост в освоении.

Недостатки:

- необходимо точное понимание структуры приложения;
- не подходит для небольших приложений.

### 1.2.3.2 FLASK

Flask – фреймворк для создания веб-приложений на языке программирования Python. Он относится к категории так называемых

микрофреймворков – минималистичных каркасов веб-приложений, предоставляющих лишь самые базовые возможности [M. Grinberg].

Преимуществами данного фреймворка являются:

- простота и гибкость;
- хорошо подходит для небольших и средних проектов (также отлично подходит и для прототипирования);
- можно вносить изменения почти на ходу.

### 1.3.3. ИСПОЛЬЗУЕМЫЕ БИБЛИОТЕКИ

Для исследования влияния климатических факторов на тепловой режим общественного здания были использованы следующие библиотеки:

- `pandas` – это высокоуровневая Python библиотека для анализа и обработки данных. Пакет прежде всего предназначен для очистки и первичной оценки данных по общим показателям, например среднему значению, квантилям и так далее.
  - хранение данных о количестве людей в помещении;
  - обработка табличных данных;
- `requests` – простая HTTP-библиотека для Python.
  - получение данных о расписании занятий в помещениях;
- `json` – стандартная Python библиотека для взаимодействия с JavaScript Object Notation (JSON) объектами.
  - отправление запроса на получение данных о расписании;
  - парсинг полученных о расписании занятий данных;
- `matplotlib` – библиотека на языке программирования Python для визуализации данных двумерной графикой (3D графика также поддерживается); `plotly` – библиотека для создания интерактивных графиков.
  - визуализация обучения модели;
  - визуализация результатов прогнозирования;
  - визуализация дендрограмм;

- отображение температур в помещениях;
- отображение аномальности температур.
- `scipy` — это библиотека для языка Python, основанная на расширении NumPy, но для более глубоких и сложных научных вычислений, анализа данных и построения графиков.
  - кластеризация помещений по температуре;
- `sklearn` – библиотека машинного обучения, которая содержит функции и алгоритмы классификации, прогнозирования, нормализации, разбивки данных на группы.
  - нормализация данных;
  - расчет средней квадратичной ошибки прогнозирования.
- `keras` – библиотека для Python, которая предназначена для глубокого машинного обучения, позволяет создавать и настраивать модели.
  - построение модели прогнозирования температуры в помещении;
  - сохранение и загрузка модели прогнозирования.
- `flask` – фреймворк для создания веб-приложений на языке программирования Python;
- `threading` – библиотека для распараллеливания процессов;
- `sqlite3` – библиотека для работы с системой управления базой данных SQLite;
- `primeVue` – библиотека компонентов для Vue.js;
- `vue-chart` – библиотека для построения графиков.

### 1.3. КЛАСТЕРИЗАЦИЯ ВРЕМЕННЫХ РЯДОВ

Существует несколько типов схожести временных рядов:

- Ряды похожие по времени – точки и интервалы возрастания и убывания точно или почти точно соответствуют друг другу во времени;
- ряды похожие по форме – ряды имеют одинаковые характерные особенности;

- ряды похожие по структуре – имеют последовательности с одинаковыми законами изменения.

Один из способов кластеризации временных рядов – преобразовать ряды в данные фиксированного размера, и кластеризовать уже трансформированные данные. Данный способ хорошо работает, если тип схожести данных соответствует рядам, похожим по структуре [S. Aghabozorgi, A. S. Shirkhorshidi и др. 15–25 с.].

Существует несколько метрик для измерения схожести:

- Евклидово расстояние (ряды похожие во времени);
- L1 расстояние (ряды похожие во времени);
- минимальная прыжковая стоимость (ряды похожие во времени);
- кросс-корреляция (ряды похожие по структуре);
- динамическая трансформация временной шкалы (ряды похожие по форме).

Данные метрики применяются при кластеризации, наиболее зарекомендовавший себя вид кластеризации – иерархический. Существует два типа кластеризации, делительная и агломеративная [U. Malik]. В делительной иерархической кластеризации используется нисходящий подход, т. е. все точки данных рассматриваются как один большой кластер, и при кластеризации, данный кластер делится в зависимости от расстояний на меньшие кластеры, пока в каждом кластере не останется по одному элементу или не будет достигнуто некоторое барьерное значение. В то же время в агломеративной кластеризации используется восходящий подход, т. е. изначально все точки данных находятся в отдельных кластерах, и в процессе кластеризации, кластеры объединяются между собой до тех пор, пока не останется один кластер, либо не будет достигнуто заданное число кластеров и/или не будет достигнуто барьерное значение расстояния между кластерами.

## 1.4. ПОИСК АНОМАЛИЙ ВО ВРЕМЕННЫХ РЯДАХ

Обнаружение аномалий — это задача выявления редких элементов, событий или наблюдений, которые являются подозрительными и кажутся отличными от большинства данных. Существует большое количество методов для поиска объектов, которые отличаются от обычных, но данные методы не учитывают аспект последовательности данных. Следовательно, если данные обрабатываются просто как набор значений, игнорируя их временной аспект, аномалия может быть не обнаружена [D. Cheboli. 6–11 с.].

Существует три типа аномальности:

**Аномальное наблюдение** – отдельный экземпляр временного ряда, который является аномальным в определенном контексте. Например, температура в этом месте может быть аномальна зимой, но нормальной летом.

**Аномальная подпоследовательность** – подпоследовательность временного ряда, которая сильно отличается от остальной части последовательности. Например, одно и то же значение существует аномально долгое время, хотя само по себе значение не является аномалией, так как встречается в нескольких местах.

**Аномальный временной ряд** – тестовый временной ряд, который является аномальным по отношению к базе обучающих временных рядов. Предполагается, что обучающий временной ряд описывает нормальное течение исследуемого процесса.

### 1.4.1. МЕТОДЫ ТРАНСФОРМАЦИИ ДАННЫХ

#### Методы агрегирования

Методы агрегирования агрегирует временной ряд, заменяя набор последовательных значений на их репрезентативное значение (чаще всего среднее). Это уменьшает размерность данных, маскирует шум и пропущенные значения, но данный подход также может маскировать некоторые особенности данных, что может затруднить поиск аномалий.

#### Методы дискретизации



Цель дискретизации состоит в преобразовании временного ряда в дискретную последовательность конечных алфавитов. Основная идея – использовать алгоритмы обнаружения аномалий в символьной последовательности. Однако дискретизация также может привести к потере информации.

#### **1.4.2. МЕТОДЫ ОБНАРУЖЕНИЯ**

##### **Алгоритмы на основе скользящих окон**

Алгоритмы на основе скользящих окон делят временной ряд на подпоследовательности фиксированного размера (окна). Данный метод основан на том, что аномалия во временном ряду может быть вызвана наличием одной или несколько аномальных подпоследовательностей.

Недостатком данного подхода является то, что размер окна должен быть выбран тщательно, чтобы можно было зафиксировать аномалию. Оптимальный размер окна зависит от длины аномальной области. Другим недостатком является большое количество ресурсов, необходимых для анализа. Сложность данного алгоритма  $O((nl)^2)$ , где  $l$  – средняя длина временного ряда,  $n$  – количество тестовых и обучающих рядов [D. Cheboli. 19–21 с.].

Данные алгоритмы могут найти различные типы аномалий: аномальное наблюдение, аномальная подпоследовательность, аномальный временной ряд в целом. Поскольку весь временной ряд разбивается на мелкие подпоследовательности, можно легко определить, если наблюдение или подпоследовательность аномальна. Если весь временной ряд аномален, то все подпоследовательности также будут аномальны, поэтому методы на основе окон их хорошо обнаруживают.

##### **Метрические алгоритмы**

Метрические алгоритмы рассчитывают схожесть тестовой и обучающей выборки для вычисления аномальности тестового ряда. Данный подход основывается на предположении, что аномальные временные ряды

отличаются от обычных, и это различие можно зафиксировать с помощью меры близости [D. Cheboli. 21–23 с.].

Оценка аномалии тестового ряда по отношению к обучающей выборке рассчитывается по следующим методикам:

1. K-NN: расстояние тестового временного ряда до его  $k$  ближайших соседей из обучающей выборки является мерой аномалии.
2. Кластеризация: обучающие временные ряды кластеризуются, вычисляются центроиды полученных кластеров, оценкой аномалии является расстояние тестового ряда до ближайшего центроида.

Метрические методы отличаются выбором мер подобия для оценки аномальности. Могут быть использованы такие метрики, как евклидово или косинусное расстояние, корреляция, DTW и т. д. Так как временные ряды могут иметь разные характеристики (длина, амплитуда, условия времени и т. д.), это ограничивает использование тех или иных метрик.

Еще одним недостатком данного подхода является то, что метрические алгоритмы смогут определить является ли временной ряд аномальным, но не смогут определить точное местонахождение аномальности, для этого необходимо выполнить постобработку ряда. Данный алгоритм также может упустить аномальное наблюдение, поскольку его эффект может быть незаметен, когда рассматривается сразу весь временной ряд. Таким образом, метрические методы позволяют найти аномальные последовательности или целый аномальный ряд.

### **Алгоритмы на основе прогнозирования**

Данный подход основывается на предположении, что нормальные значения временного ряда генерируются из статистического процесса, а аномальные – не соответствуют этому процессу. Таким образом, на основе исторических данных строится некоторая модель процесса, затем строится предсказание и на основе отклонения фактического значения от предсказания вычисляется мера аномальности (разницы между предсказанным и фактическим значением) [D. Cheboli. 23–28 с.].

Базовый метод обнаружения аномалий, основанный на модели прогнозирования, состоит из следующих шагов:

1. Построение предсказательной модели для обучающего временного ряда, которая использует  $m$  наблюдений для прогнозирования  $m+1$  наблюдения, следующего за ними.
2. Выполнить прогноз для тестового временного ряда на каждом временном шаге, используя  $m$  предыдущих.

Ошибка прогноза, соответствующая наблюдению, является функцией разницы между прогнозируемым значением и фактическим наблюдением.

Сложность данного подхода состоит в построении прогнозной модели достаточно высокой точности. Также методы, основанные на прогнозировании, используют исторические данные фиксированной длины. Для одного и того же временного ряда иногда для предсказания достаточно небольшой истории, но в других случаях может понадобиться более длинная история. Преимуществом данного подхода является то, что данный алгоритм позволяет обнаружить все типы аномалий.

## ВЫВОДЫ К ГЛАВЕ 1

В данной главе были решены следующие задачи:

1. Анализ климатических факторов, влияющих на температуру в помещениях.
2. Сравнительный анализ технологий и выбор подходящих.
3. Изучение подходящих алгоритмов кластеризации временных рядов.
4. Изучение подходов поиска аномалий во временных рядах.

В результате анализ климатических факторов был сделан следующий вывод. На климат внутри помещения оказывают влияние внешняя и внутренняя среды, технологические процессы, а также системы обеспечения микроклимата. Такие внешние факторы как инсоляция и воздушные потоки могут оказывать достаточно существенное воздействие на микроклимат помещения. Влажность же оказывает заметное влияние только в долгосрочной перспективе, поэтому данным фактором можно пренебречь.

В ходе сравнения языков программирования, подходящих для анализа, обработки и визуализации данных, был выбран Python. Так как R имеет большое количество инструментов для обработки данных, но недостаточно удобен в сборе данных, JavaScript имеет широкий спектр инструментов визуализации, но низкую скорость обработки данных и меньшее количество библиотек анализа данных, Scala подходит для работы с данными, но сложен для освоения и развертывания. Python подходит больше всего, так как его библиотеки легко доступны, подключаемы и удобны, также их большое количество; реализованы удобные инструменты для визуализации, сбора и обработки данных; имеется удобный бесплатный облачный сервис для совместной разработки на языке Python.

В ходе сравнения алгоритмов кластеризации и анализа имеющихся данных, был сделан вывод о том, что имеющиеся временные ряды относятся к структурно схожим. В таком случае было решено использовать иерархическую кластеризацию с подходом, подразумевающим

преобразование рядов в данные фиксированного размера и их дальнейшую кластеризацию.

По итогам анализа методов поиска аномалий были выбраны алгоритмы на основе прогнозирования, так как данные методы могут быть довольно точными при правильном обучении и достаточном количестве данных, позволяют точно локализовать время аномалии без обратного преобразования, а также данный подход может найти закономерности, пропущенные в ходе ручного анализа, без использования методов машинного обучения.

Разрабатываемая система имеет сложную MVC архитектуру с небольшим пользовательским интерфейсом. Поэтому для данного проекта в качестве frontend framework был выбран Vue, так как он имеет высокую производительность, обладает компонентной архитектурой как Angular и виртуальным DOM как React. Для серверной части приложения, ввиду проведения исследования с использованием языка программирования python, также использовался данный язык с микрофреймворком flask, для обработки веб-запросов.

## ГЛАВА 2. ПРОГРАММНАЯ РЕАЛИЗАЦИЯ

### 2.1. ИНТЕРПРЕТАЦИЯ ДАННЫХ

#### 2.1.1. ИСТОЧНИКИ ДАННЫХ

Для поиска аномалий и кластеризации помещений необходимы следующие данные:

- температура с датчиков;
- архив погоды;
- данные о положении солнца;
- данные о расписании занятий.

Было решено проводить исследования по помещениям в здании института математики и компьютерных наук (ИМИКН) Тюменского государственного университета (ТюмГУ). В некоторых кабинетах имеются температурные датчики (на стене и на батарее). Информация о температурах хранится на веб ресурсе в удобном для парсинга виде.

Исходя их достаточно малого количества погодных сервисов, которые хранят подробные данные за предыдущие периоды времени, а также информацию о состоянии облачности, был выбран сервис `meteocenter`. Данный сервис содержит архив погоды по разным городам, информацию можно получить с запрошенной даты в количестве 100 штук на запрос. Периодичность между данными составляет от 30 минут до часа, таким образом, за один запрос приходит информация о погоде за 2–4 дня. В данном архиве содержится вся необходимая для исследования погодная информация.

Для получения данных о положении солнца используется библиотека `ruerphem-sunpath`, с помощью которой в зависимости от времени суток и долготы и широты объекта исследования можно рассчитать азимут солнца и его высоту над горизонтом.

Данные о проведении учебных занятий кабинетах были получены с информационной системы для студентов и преподавателей `MODEUS`. Данный ресурс содержит информацию о расписании занятий в ТЮМГУ ИМИКН:



список мероприятий, места проведения (здание и кабинет), дата и время (начало и конец занятия), информация о студенческой группе и преподавательском составе.

### **2.1.2. ОПИСАНИЕ ДОСТУПНЫХ ДАННЫХ**

Информация с датчиков хранится в виде отдельных таблиц для каждого датчика (на стене и на батареях) и кабинета (210, 316, 412а, 420, 219). Источник данных – web-ресурс, показания хранятся в формате html. Количество записей постоянно растет, так как данные продолжают поступать с марта 2020 года (на данный момент в каждой таблице около 28 тыс. записей). Каждая таблица состоит из трех столбцов:

- date (интервальная шкала) – дата снятия показаний датчиков в формате YYYY-MM-DD;
- time (интервальная шкала) – время снятия показаний датчиков в формате HH:MM:SS;
- temp (интервальная шкала) – температура с датчика в °С.

Архив погоды хранится в виде таблицы. Источник данных – web-ресурс, для получения информации необходим парсинг. Данные снимаются с периодичностью в 30 минут. Таблица состоит из следующих столбцов:

- дата (интервальная шкала) – дата в формате DD.MM.YY;
- время (интервальная шкала) – всемирное время в формате hhmm;
- ветер (направление – номинальная шкала, скорость - относительная) – направление ветра в градусах горизонта и скорость в м/с на высоте 10-12м над земной поверхностью;
- облачность (набор данных, необходимо разбить, шкала не определена) –общее количество облачности в баллах (1 балл = 10% неба), количество нижней (ниже 2000 м) облачности в баллах, высота нижней границы в метрах. Термин "нет сущ. обл." означает отсутствие облаков ниже 1500 м, термин "ясно" – безоблачное небо, "?/?" – небо не видно из-за тумана или других явлений;

- T (интервальная шкала) – температура воздуха на высоте 2 м над земной поверхностью в °C;

Данные о местоположении солнца получены с помощью библиотеки python, рассчитываются на основе координат здания ТюмГУ, даты и времени.

- azimuth – расположение солнца в градусах;
- altitude – высота солнца над горизонтом в градусах.

Для получения информации о проведении занятий в помещении формируется запрос на web-ресурс Modeus со следующими параметрами:

- timeMin (string в формате Y-m-dTH:M:SZ) – начало запрашиваемого периода;
- timeMax (string в формате Y-m-dTH:M:SZ) – конец запрашиваемого периода;
- room\_ID (string) – идентификатор помещения;
- token – временный токен авторизованного пользователя.

Результатом запроса возвращается JSON объект, который содержит информацию о проводимых в этот период занятиях:

- events – список мероприятий:
  - name – название мероприятия;
  - endsAtLocal – дата и местное время окончания мероприятия;
  - startsAtLocal – дата и местное время начала мероприятия;
  - id – идентификатор мероприятия.
- event-teams – список участников мероприятия:
  - eventID – идентификатор мероприятия;
  - size – количество участников мероприятия.

Результат парсится с помощью библиотеки json, достается информация о времени мероприятия, и количестве его участников. Далее формируется датасет, который содержит следующие параметры:

- start\_data\_time (datetime) - дата и время начала занятия;
- end\_data\_time (datetime) - дата и время окончания занятия;

- event\_id (string) - идентификатор помещения;
- event\_name (string) - название мероприятия;
- people\_count (int) количество людей – количество людей на мероприятии.

### 2.1.3. ПРЕДВАРИТЕЛЬНАЯ ОБРАБОТКА ДАННЫХ

Для исследования зависимостей теплового режима здания от интересующих климатических факторов необходимо собрать полученные данные в одну таблицу, объединив данные по дате и времени с часовым интервалом.

Полученные данные были предобработаны следующим образом:

1. Сформирован столбец date\_time на основе столбцов date и time.
2. Проведена переиндексация таблицы данных температурных датчиков на основе столбца date\_time с шагом в час.
3. Все строковые числовые данные переведены в int или float, округлены до сотых.
4. Погодные данные об облачности разбиты на 4 столбца: overall\_amount (COA), under\_lower\_amount (CULA), lower\_bound (CLB), type (CT).
5. Километры переведены в метры в столбце visibility таблицы погоды.
6. Заменены строковые значения в столбцах wind\_speed, wind\_direction в таблице погоды.
7. Посчитаны значения azimuth и altitude на основе date\_time и координатах исследуемого здания;

Данные погоды, температуры в помещении и положении солнца объединены в одну таблицу по столбцу date\_frame. Алгоритм сбора и предобработки данных представлен на рисунке 4.

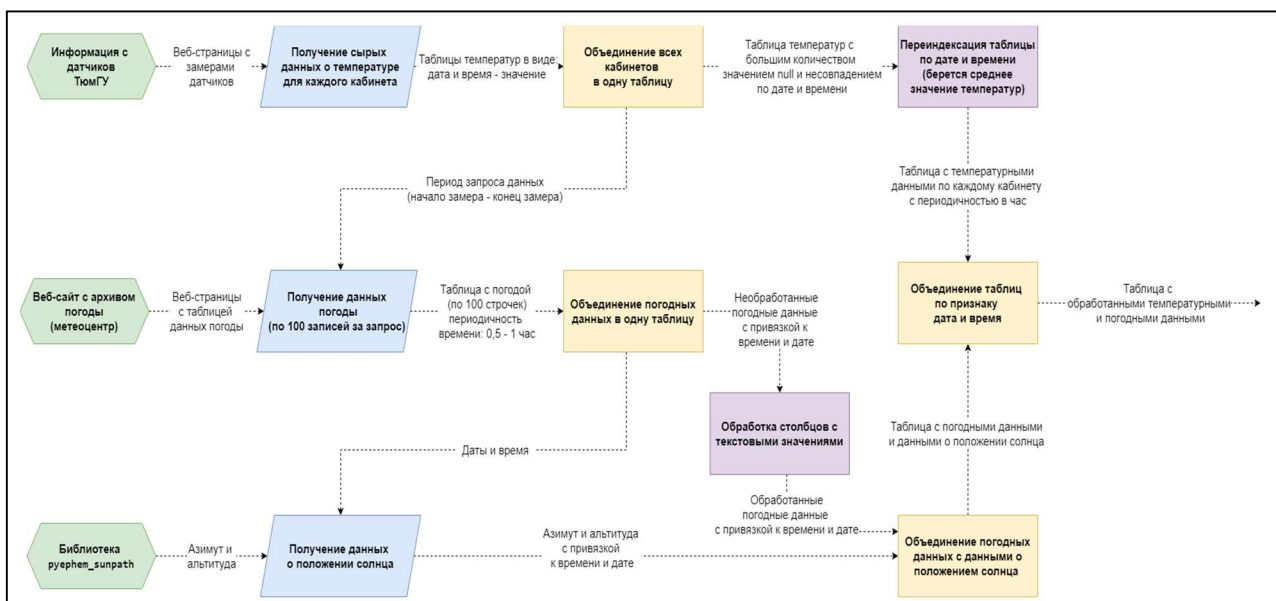


Рисунок 4. Архитектура блока предобработки данных

Итоговая таблица хранятся в csv файле, новые данные дозагружаются. На данный момент в таблице имеются данные с марта по декабрь 2020 года (около 7000 записей).

Также была собрана часть данных из системы «Модеус», а именно расписание занятий для кабинетов и количество человек в аудиториях. Эта информация выгружалась единоразово и использовалась в ходе исследования, но автоматизировать выгрузку данных не удалось ввиду технических ограничений.

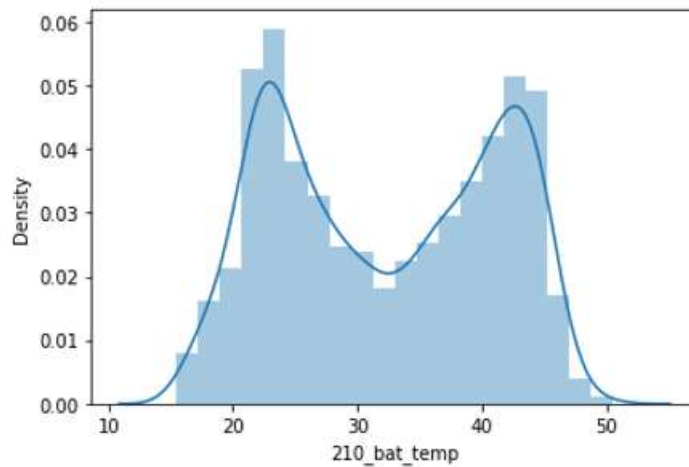
#### 2.1.4. ПЕРВИЧНЫЙ АНАЛИЗ ДАННЫХ

После получения и предобработки данных, был проведён первичный анализ. Данные были визуализированы для просмотра распределений, а также выборочно, для некоторых наборов была проведена проверка на нормальность с помощью теста Шапиро-Уилка.

Итоговая таблица с данными погоды и температур после предобработки состоит из следующих столбцов:

- **date\_time**: datetime (интервальная шкала) – индексы, дата и время снятия данных температурных датчиков и погодных условий с интервалом в час в формате YYYY-MM-DD HH:MM:SS;

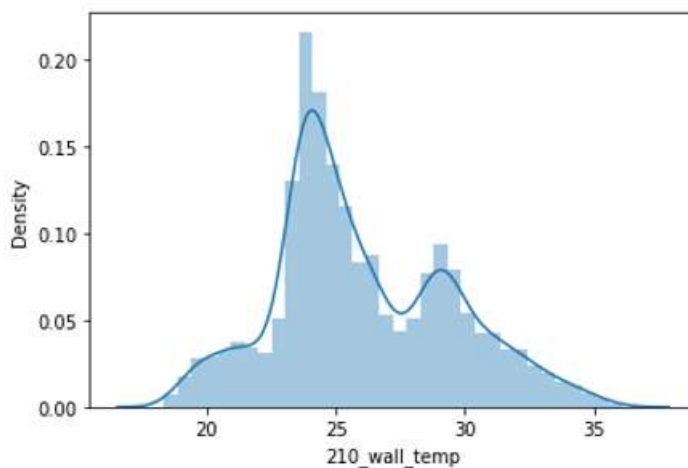
- **210\_bat\_temp**: float (интервальная шкала) – температура с датчика на батарее 210 кабинета в °C (Рисунок 5);



NaN: 313  
 mean: 32.3  
 std: 8.94  
 min: 15.45  
 max: 50.5

Рисунок 5. Распределение температур батарей 210 кабинета

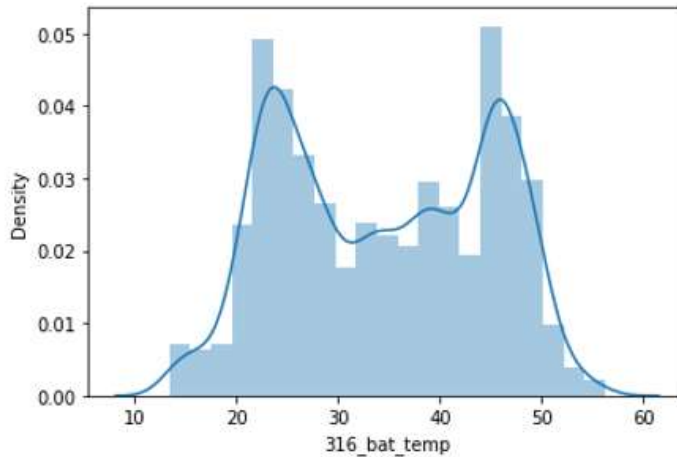
- **210\_wall\_temp**: float (интервальная шкала) – температура с датчика на стене 210 кабинета в °C (Рисунок 6);



NaN: 703  
 mean: 26  
 std: 3.44  
 min: 18.33  
 max: 36.11

Рисунок 6. Распределение температур 210 кабинета

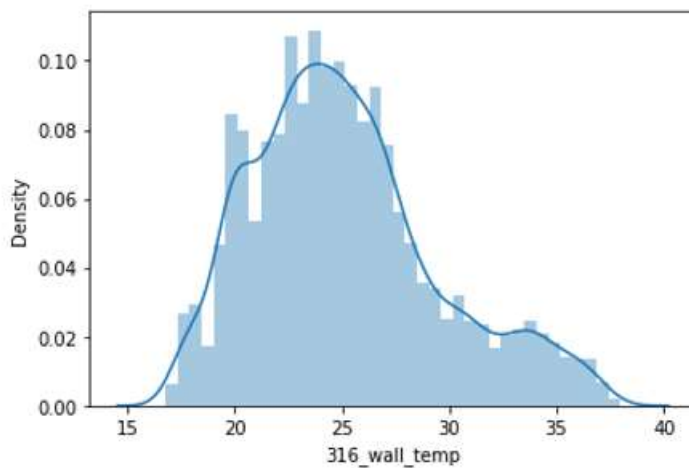
- **316\_bat\_temp**: float (интервальная шкала) – температура с датчика на батарее 316 кабинета в °C (Рисунок 7);



NaN: 218  
 mean: 34.55  
 std: 10.29  
 min: 13.41  
 max: 56.21

Рисунок 7. Распределение температур батарей 316 кабинета

- **316\_wall\_temp:** float (интервальная шкала) – температура с датчика на стене 316 кабинета в °C (Рисунок 8);

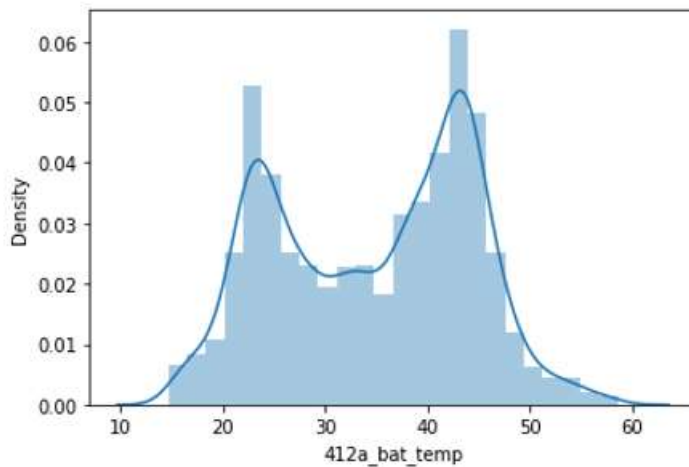


NaN: 235  
 mean: 25.14  
 std: 4.37  
 min: 16.79  
 max: 37.97

Рисунок 8. Распределение температур 316 кабинета

Распределение данного признака отдалённо напоминает нормальное распределение, но гипотеза о нормальном распределении была отвергнута по критерию Шапиро-Уилка.

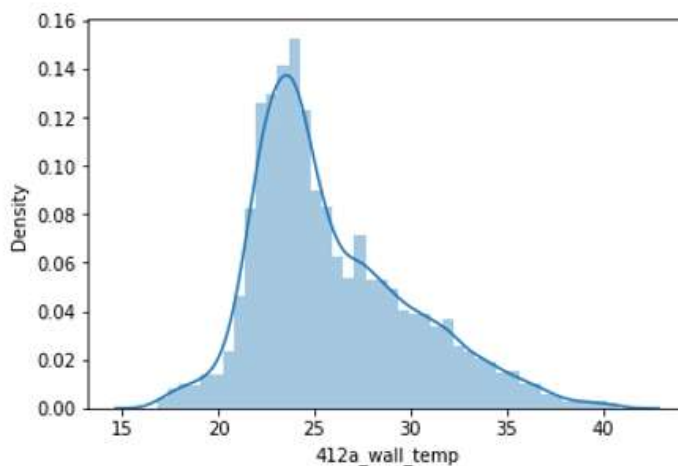
- **412a\_bat\_temp**: float (интервальная шкала) – температура с датчика на батарее 412а кабинета в °С (Рисунок 9);



NaN: 478  
 mean: 34.79  
 std: 9.62  
 min: 14.65  
 max: 58.59

Рисунок 9. Распределение температур батарей 412а кабинета

- **412a\_wall\_temp**: float (интервальная шкала) – температура с датчика на стене 412а кабинета в °С (Рисунок 10);

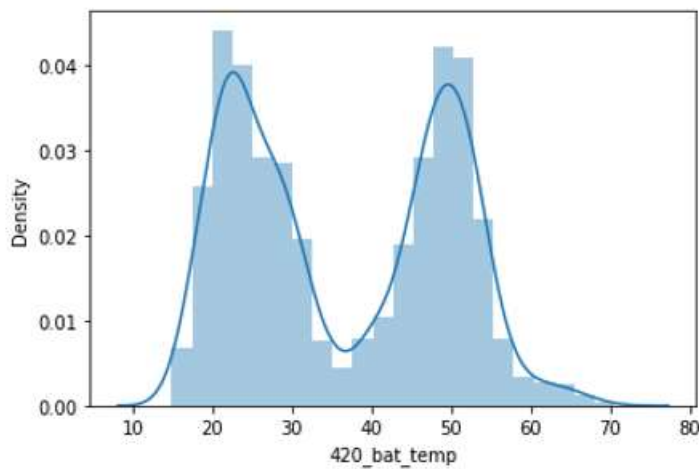


NaN: 468  
 mean: 25.91  
 std: 4.15  
 min: 16.8  
 max: 40.78

Рисунок 10. Распределение температур 412а кабинета

Гипотеза о нормальном распределении данного признака также была отвергнута по критерию Шапиро-Уилка.

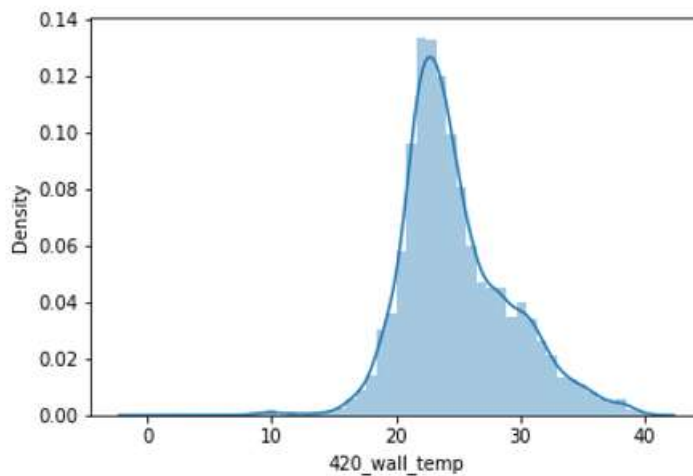
- **420\_bat\_temp**: float (интервальная шкала) – температура с датчика на батарее 420 кабинета в °C (Рисунок 11);



NaN: 46  
 mean: 36.57  
 std: 13.32  
 min: 14.9  
 max: 70.53

Рисунок 11. Распределение температур батарей 420 кабинета

- **420\_wall\_temp**: float (интервальная шкала) – температура с датчика на стене 420 кабинета в °C (Рисунок 12);



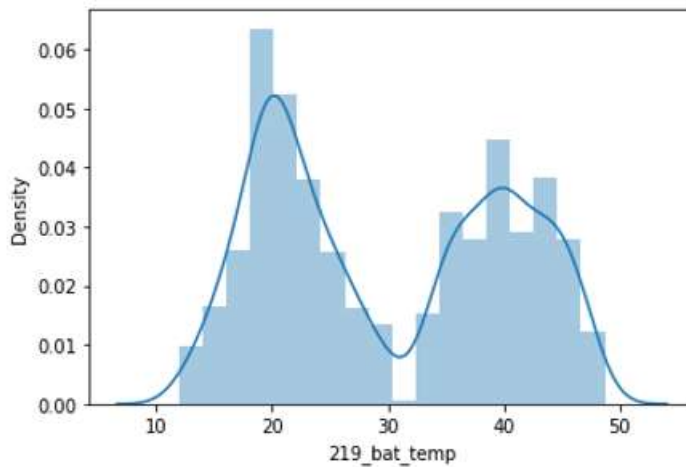
NaN: 47  
 mean: 24.8  
 std: 4.32  
 min: -0.06  
 max: 40.1

Рисунок 12. Распределение температур 420 кабинета

Гипотеза о нормальном распределении данного признака также была отвергнута по критерию Шапиро-Уилка.



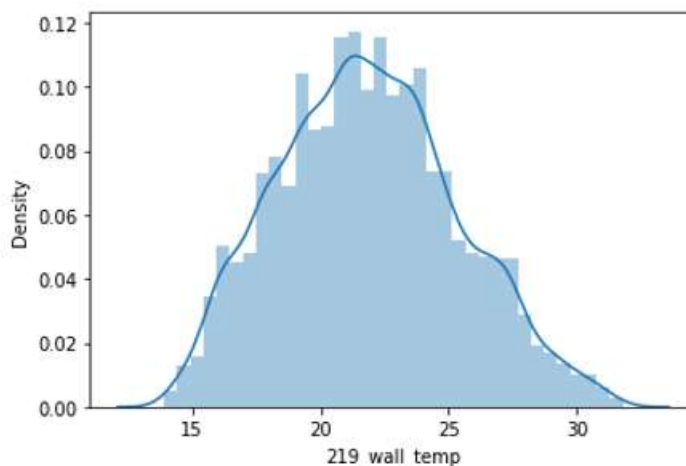
- **219\_bat\_temp**: float (интервальная шкала) – температура с датчика на батарее 219 кабинета в °C (Рисунок 13);



NaN: 238  
 mean: 30.06  
 std: 10.04  
 min: 11.98  
 max: 48.7

Рисунок 13. Распределение температур батарей 219 кабинета

- **219\_wall\_temp**: float (интервальная шкала) – температура с датчика на стене 219 кабинета в °C (Рисунок 14);



NaN: 238  
 mean: 21.87  
 std: 3.5  
 min: 13.91  
 max: 31.78

Рисунок 14. Распределение температур 219 кабинета

Гипотеза о нормальном распределении данного признака также была отвергнута по критерию Шапиро-Уилка.

На всех графиках распределения температуры батарей отопления можно увидеть 2 наиболее частых интервала значений — это распределение температуры в отопительный и в неотопительный сезоны. Для всех кабинетов значение температуры в неотопительный сезон – от 15 до 30, в отопительный – примерно от 35 до 55.

- **wind\_dir**: int (относительная шкала) – направление ветра на высоте 10-12м над земной поверхностью в градусах горизонта (Рисунок 15);

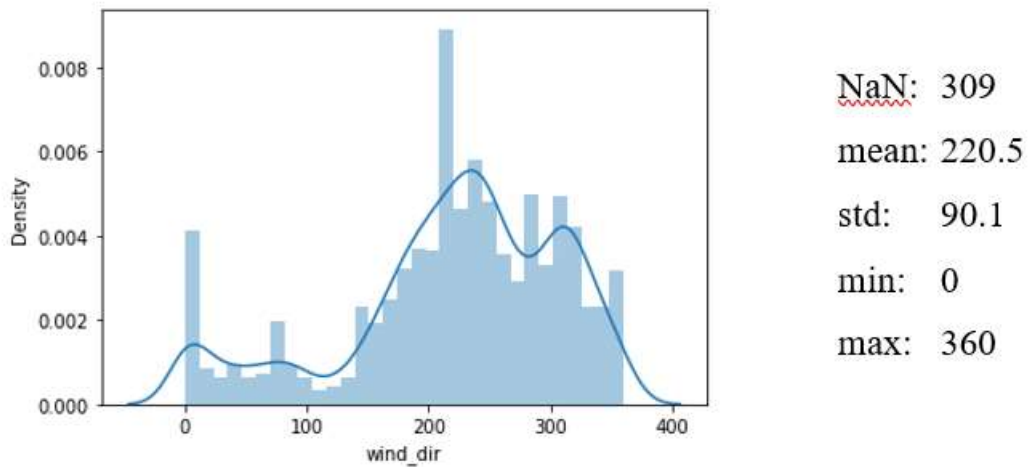


Рисунок 15. Распределение направления ветра

На графике представлено распределение направлений ветра, как можно заметить, направление ветра не распределено равномерно на наших данных. Наиболее частое значение направление ветра - 220 градусов (юго-юго-западное направление).

- **wind\_speed**: float (относительная шкала) – скорость ветра на высоте 10-12м над земной поверхностью в м/с (Рисунок 16);

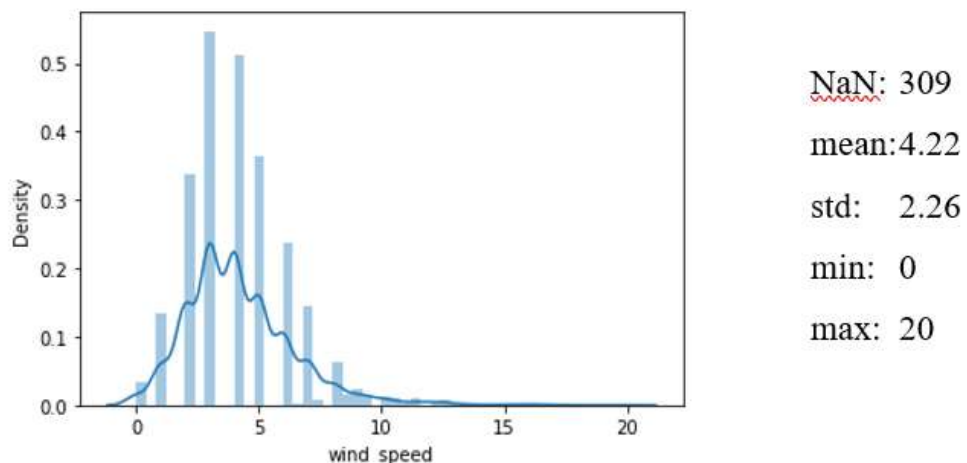


Рисунок 16. Распределение скорости ветра

Глядя на данный график, можно предположить, что величина имеет нормальное распределение, но данная гипотеза была опровергнута по критерию Шапиро-Уилка.

- **visibility:** int (относительная шкала) – горизонтальная дальность видимости (на высоте 2м над земной поверхностью) в км (Рисунок 17);

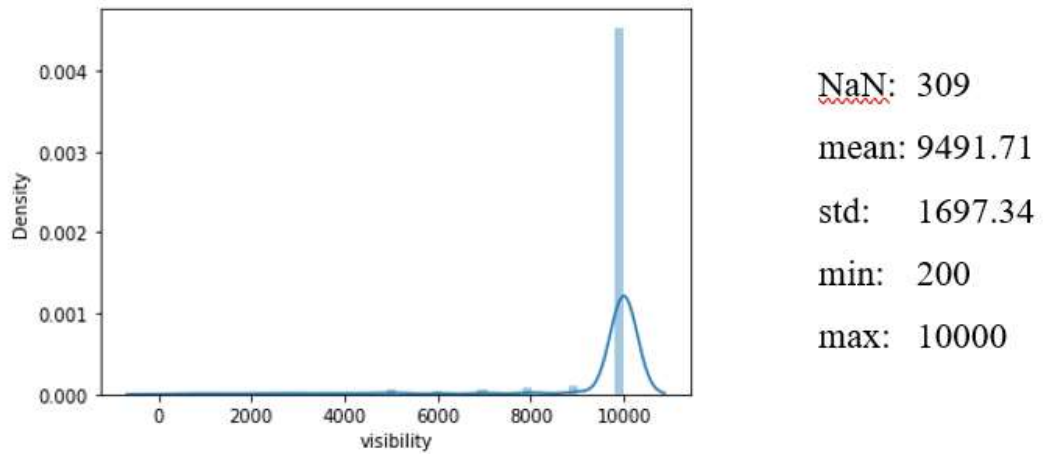


Рисунок 17. Распределение дальности видимости

На данном графике видно, что преобладающее число значений (более 95%) - равно 10000, остальных же значений несущественное количество.

- **atm\_phenomena:** string (номинальная шкала) – атмосферные явления (осадки, грозы, туманы и т. д.)(Рисунок 18);

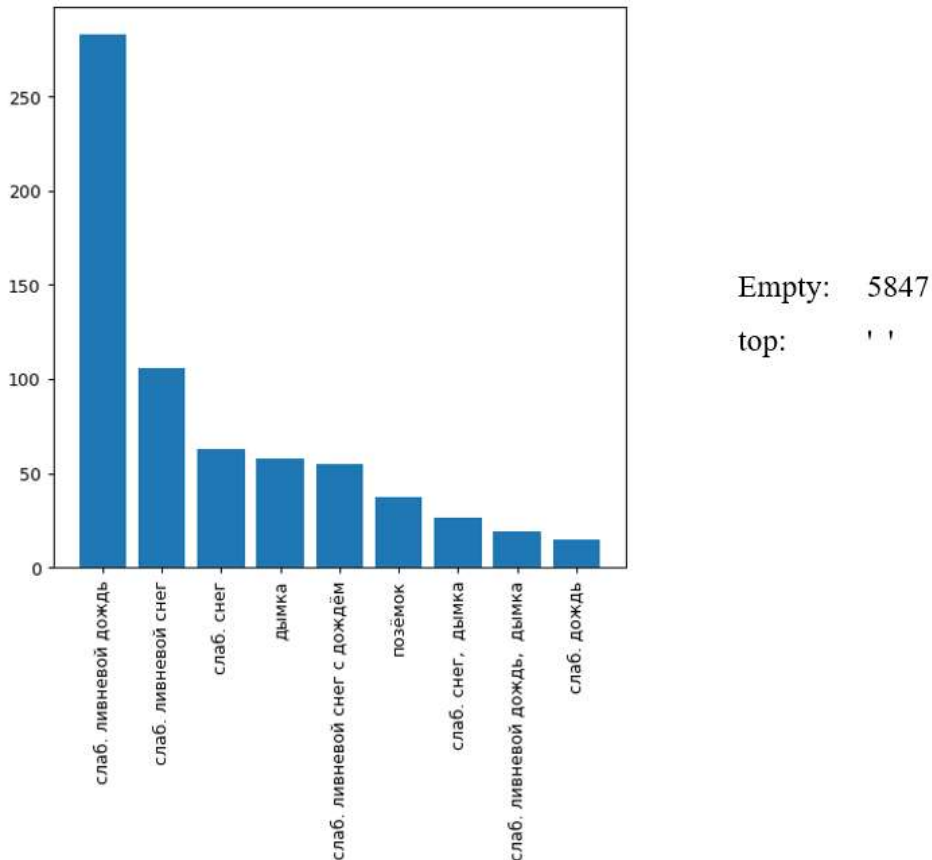


Рисунок 18. Распределение атмосферных явлений

Для наглядности визуализации распределения данного признака были убраны пустые значения. Пустых значений 5847, что составляет более 80% данных.

- **T**: int (интервальная шкала) – температура воздуха на высоте 2м над земной поверхностью в °C (Рисунок 19);

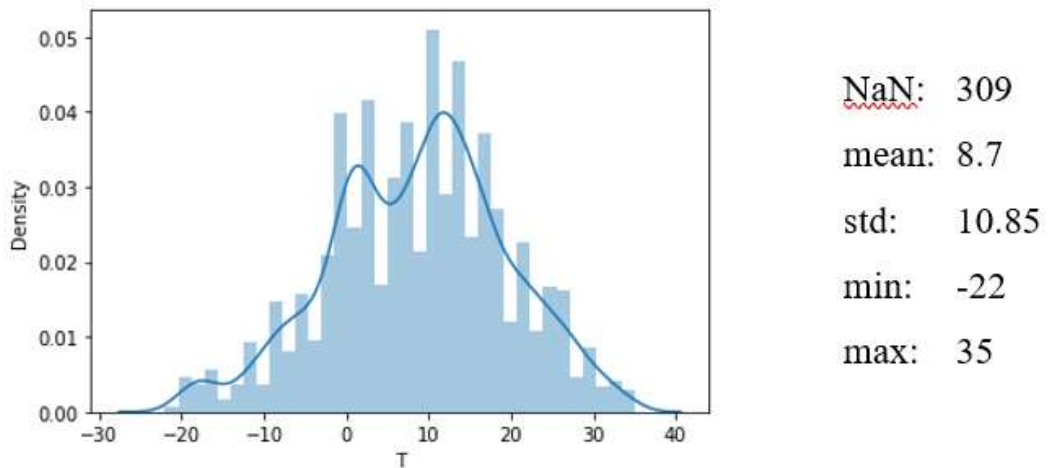


Рисунок 19. Распределение температуры воздуха на улице

Гипотеза о нормальном распределении данного признака была отвергнута по критерию Шапиро-Уилка.

- **Td**: int (интервальная шкала) – точка росы на высоте 2м над земной поверхностью в °C (Рисунок 20);

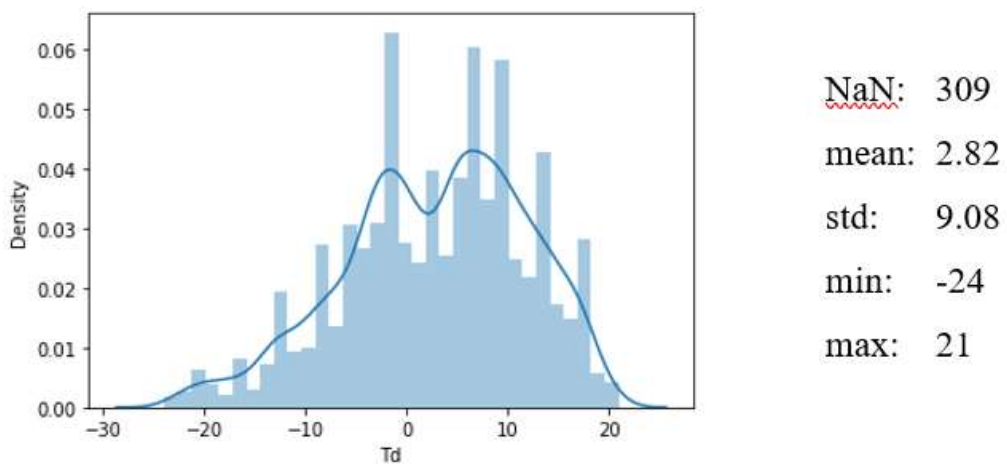
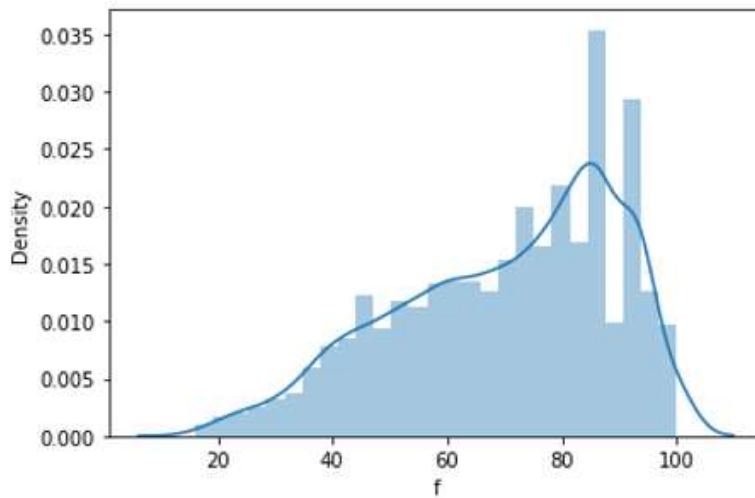


Рисунок 20. Распределение точки росы

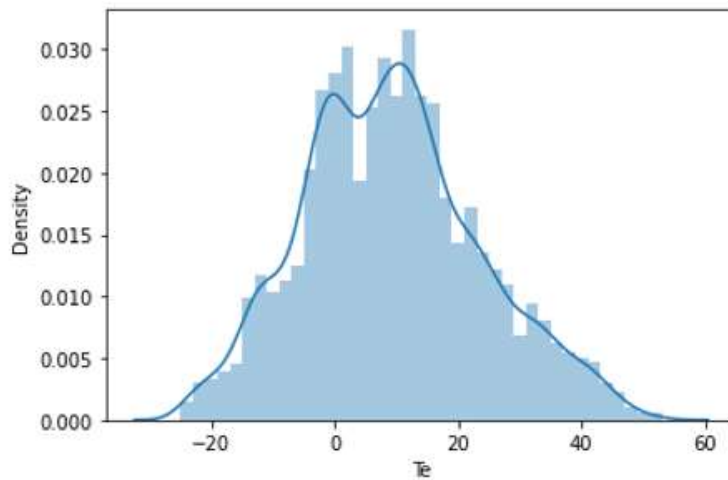
- **f: int** (относительная шкала) – относительная влажность воздуха на высоте 2м над земной поверхностью в % (Рисунок 21);



NaN: 309  
 mean: 70.2  
 std: 19.21  
 min: 16  
 max: 100

Рисунок 21. Распределение относительной влажности воздуха

- **Te: int** (интервальная шкала) – эффективная температура в °C (Рисунок 22);

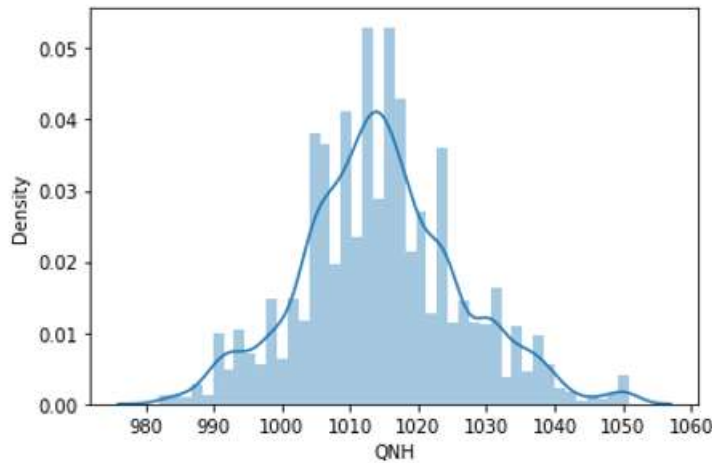


NaN: 309  
 mean: 9.15  
 std: 14.5  
 min: -25  
 max: 53

Рисунок 22. Распределение эффективной температуры

Гипотеза о нормальном распределении для данной величины была опровергнута по критерию Шапиро-Уилка.

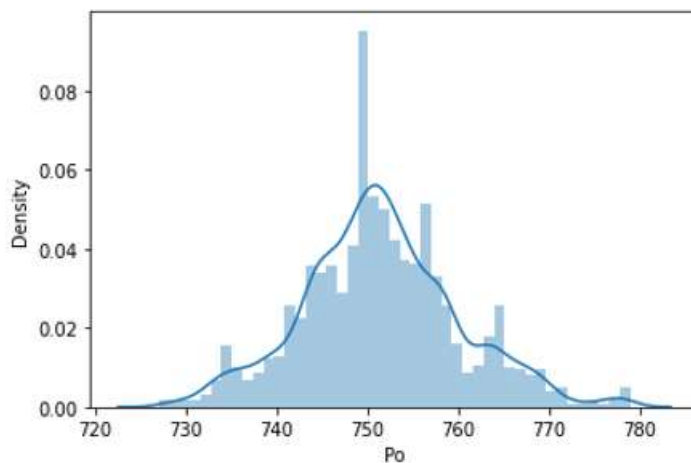
- **QNH: int** (интервальная шкала) – атмосферное давление на уровне моря в гПа (Рисунок 23);



NaN: 309  
 mean: 1014.55  
 std: 11.72  
 min: 982  
 max: 1051

Рисунок 23. Распределение атмосферного давления (гПА)

- **Po**: int (интервальная шкала) – атмосферное давление на уровне метеостанции в мм (Рисунок 24);



NaN: 309  
 mean: 751.48  
 std: 8.65  
 min: 727  
 max: 779

Рисунок 24. Распределение атмосферного давления (мм рт. столба)

Распределение идентично предыдущему параметру, разница в графиках объясняется разными единицами измерения.

- **COA**: int (относительная шкала) – общее количество облачности в баллах (1 балл = 10% неба) (Рисунок 25);

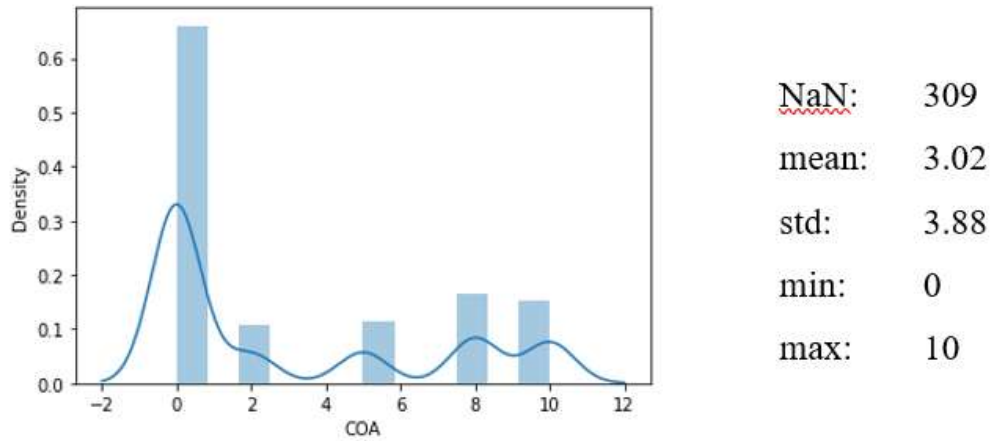


Рисунок 25. Распределение общего количества облачности  
Преобладающее большинство значений - 0 (безоблачное небо).

- **CULA:** int (относительная шкала) – количество нижней (ниже 2000м) облачности в баллах (Рисунок 26);

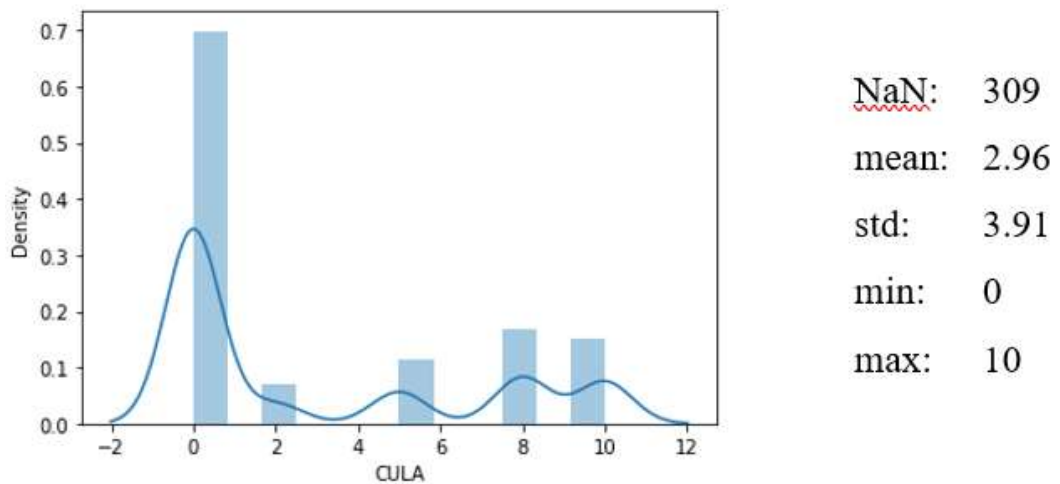


Рисунок 26. Распределение нижней облачности

- **B:** int (относительная шкала) – высота нижней границы в метрах (Рисунок 27);

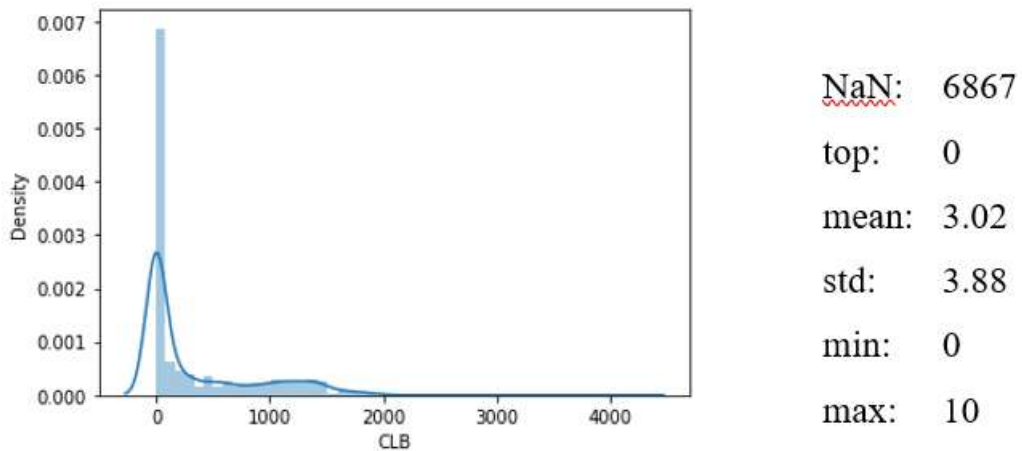


Рисунок 27. Распределение высоты нижней границы облачности  
Данные специфичные для аэропорта.

- **CT:** string (номинальная шкала) – тип облаков (Рисунок 28);

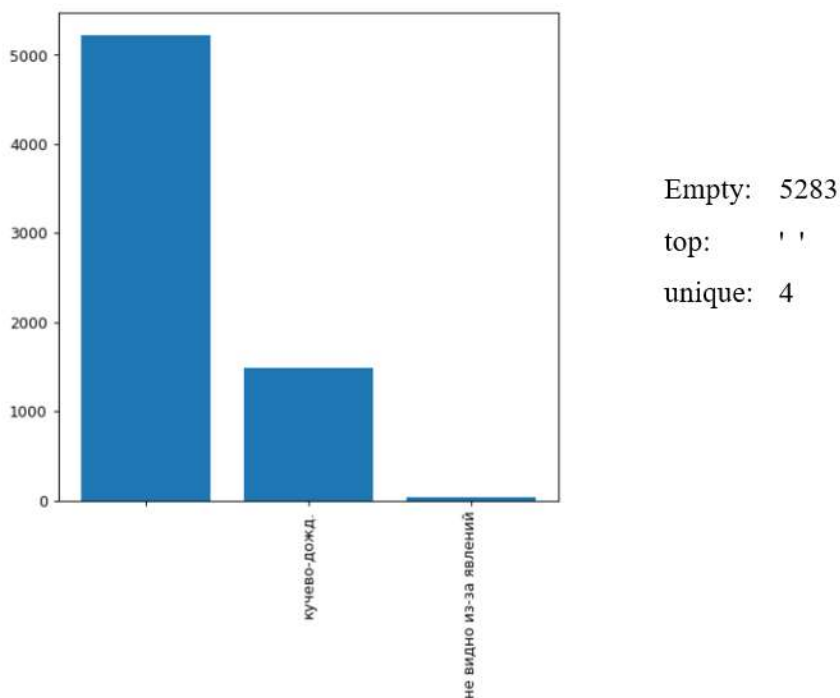
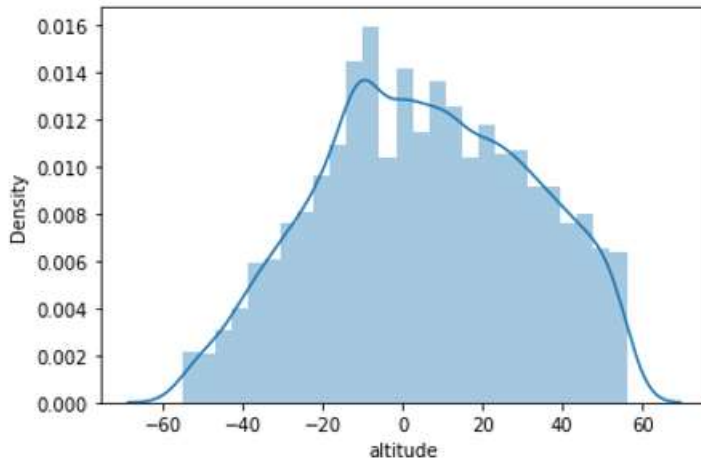


Рисунок 28. Распределение типов облаков

Данный признак содержит преимущественно пустые значения (более 80%).

- **altitude:** float (относительная шкала) – высота солнца над горизонтом в градусах (Рисунок 29);

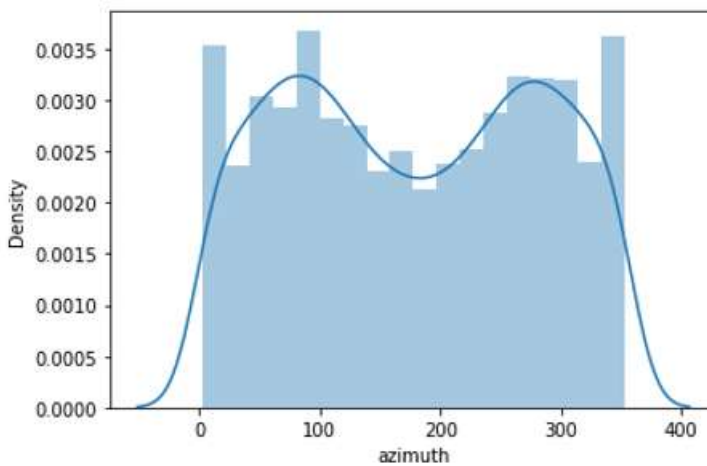




NaN: 309  
 mean: 5.33  
 std: 26.31  
 min: -55.2  
 max: 56.09

Рисунок 29. Распределение высоты солнца

- **azimuth**: float (относительная шкала) – расположение солнца в градусах (Рисунок 30).



NaN: 309  
 mean: 178  
 std: 104.71  
 min: 2.82  
 max: 352.4

Рисунок 30. Распределение расположения солнца

В рамках первичного анализа была проанализирована корреляция между признаками (коэффициент корреляции Спирмена). График корреляции приведен на рисунке 31.

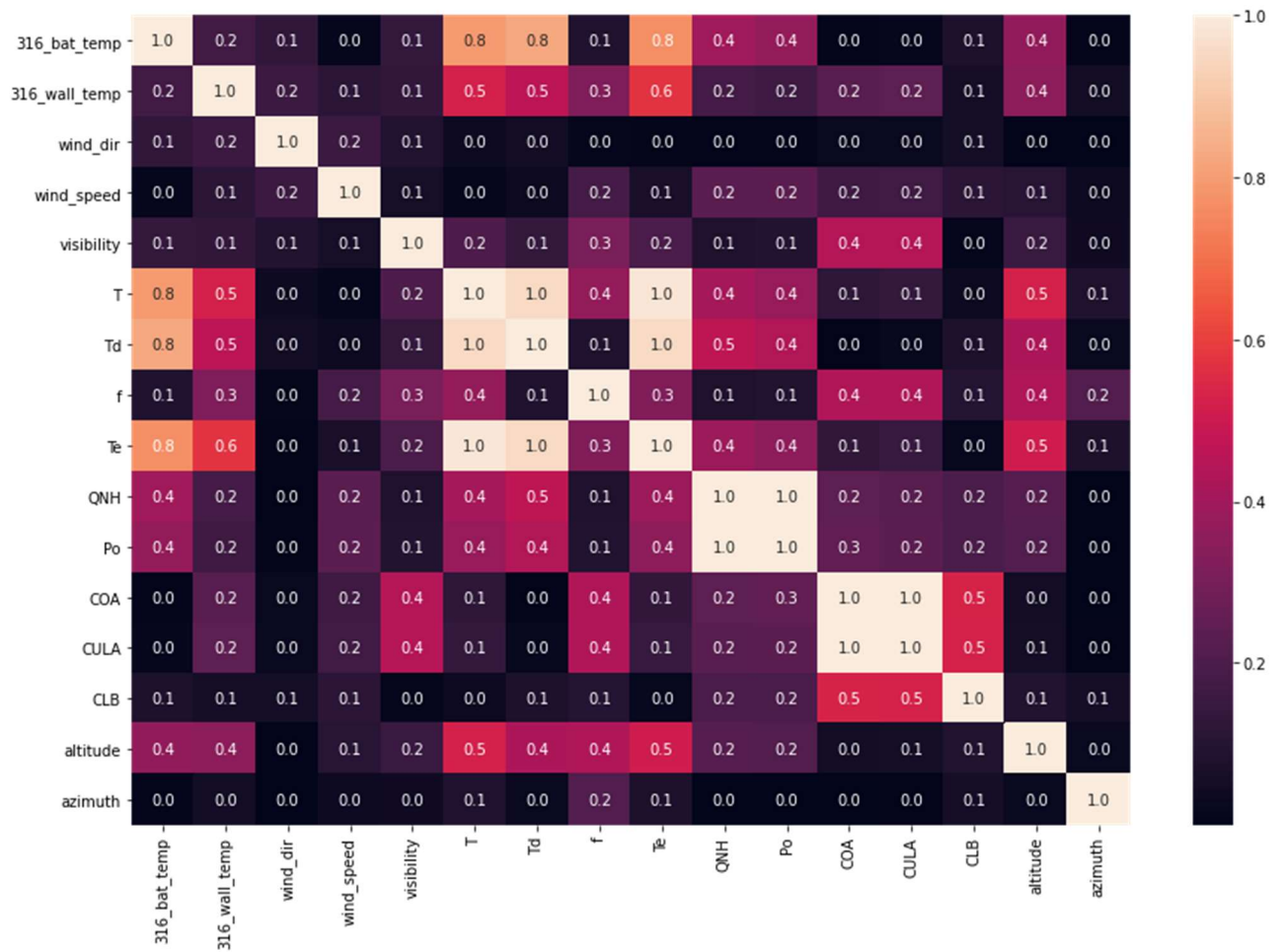


Рисунок 31 Корреляция признаков

Исходя из данного графика можно сказать, что следующие признаки сильно коррелируют:

- температура батарей в помещении и температура наружного воздуха  $\sim 0.8$  (вероятно, потому что система отопления настраивается исходя из температуры внешней среды);
- температура воздуха внутри помещения и температура наружного воздуха  $\sim 0.5$  (так как происходит теплообмен);
- температура воздуха внутри помещения и альтитуда солнца  $\sim 0.4$  (помещение нагревается от солнечных лучей).

Также видна сильная корреляция между параметрами внешней среды:

- облачности COA, CULA (они практически совпадают);
- QNH и Po (совпадают, но в разных единицах измерения);
- температура воздуха и эффективная температура (почти совпадают);

- альтитуда и температура наружного воздуха  $\sim 0.5$  (нагревается от солнца, циклы дня и ночи совпадают с циклами изменения температур).

### 2.1.3.1. АНАЛИЗ ФАКТОРА ИНСОЛЯЦИИ

Для учёта такого фактора влияния как инсоляция, был сконструирован признак «показатель влияния солнца». Данный признак необходим для того, чтобы оценить относительное количество прямых солнечных лучей, попадающих внутрь помещения с учётом положения солнца, направления окон и текущей облачности. Показатель рассчитывается следующим образом:

$$(10 - COA) \times \cos(\text{altitude}) \times \cos(\text{azimuth} - \alpha), \quad (1)$$

где COA - показатель облачности в баллах (10 баллов = 100 %),

altitude - высота солнца над горизонтом (рассчитывается в зависимости от координат помещения),

azimuth - азимут солнца (рассчитывается в зависимости от координат помещения),

alpha - азимут стены с окнами.

Таким образом, для высоты солнца над горизонтом меньшей 0 (altitude < 0) и разницей между азимутом солнца и стены меньше – 90 или больше 90 ((azimuth – alpha) < -90 или > 90) с окнами данный показатель равен нулю, так как в этих случаях солнечные лучи не попадают в помещение.

Для анализа корреляции был построен температурный график с добавлением сконструированного признака “показатель влияния солнца”, на котором анализировалась зависимость температур кабинетов от показателя влияния солнца. Для того, чтобы проверить были ли изменения температур кабинета вызваны изменениями температур наружного воздуха или отопительных систем, на графике также присутствует информация об изменении таких показателей, как температура внешней среды и температура батарей (Рисунки 32-35).

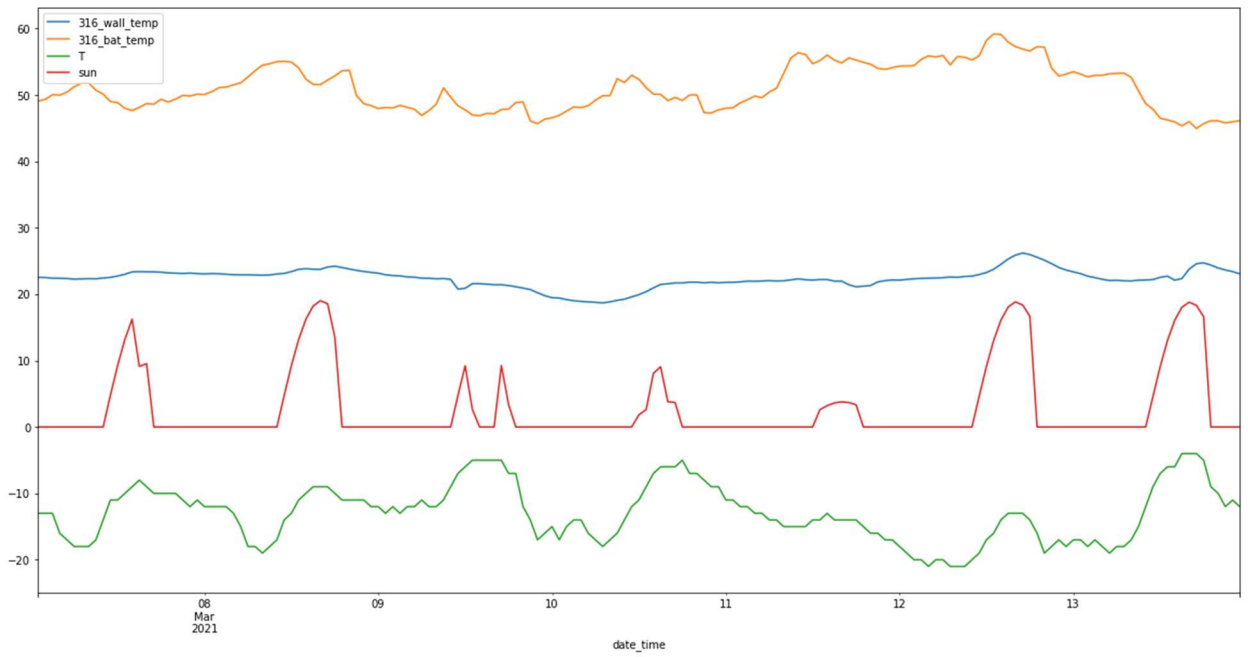


Рисунок 32. График зависимости температур от показателя солнца  
(316 кабинет с 07.03.2021 по 14.03.2021)

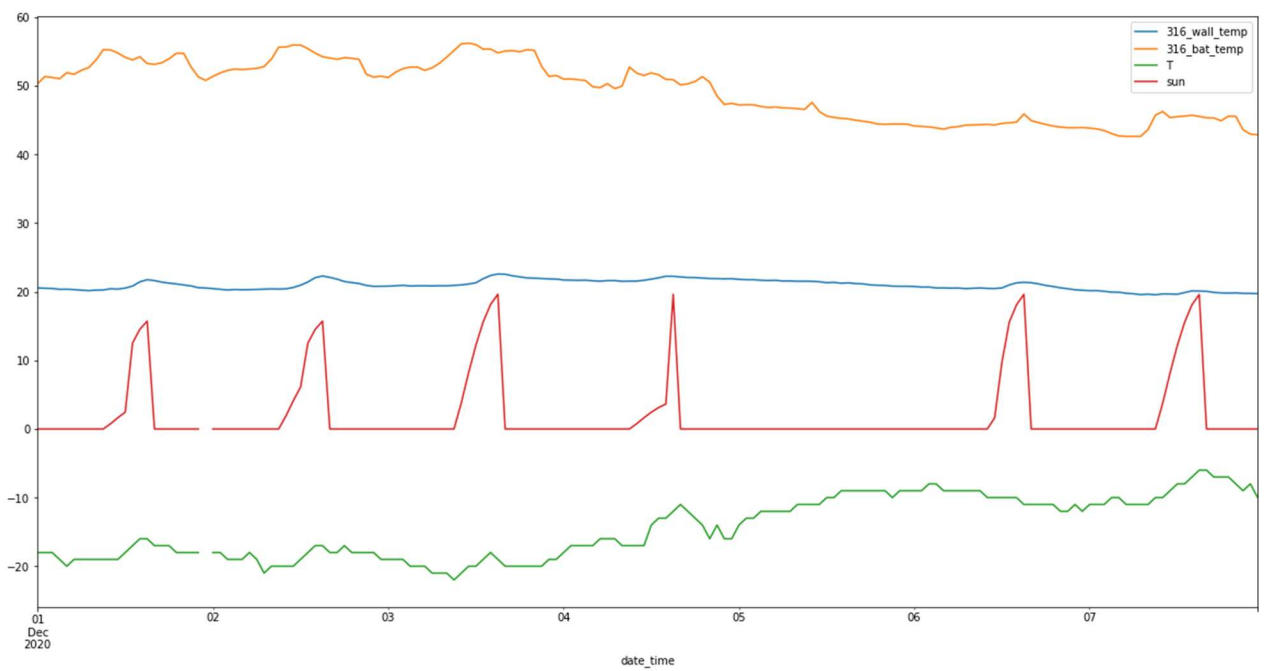


Рисунок 33. График зависимости температур от показателя солнца  
(316 кабинет с 01.12.2020 по 08.12.2020)

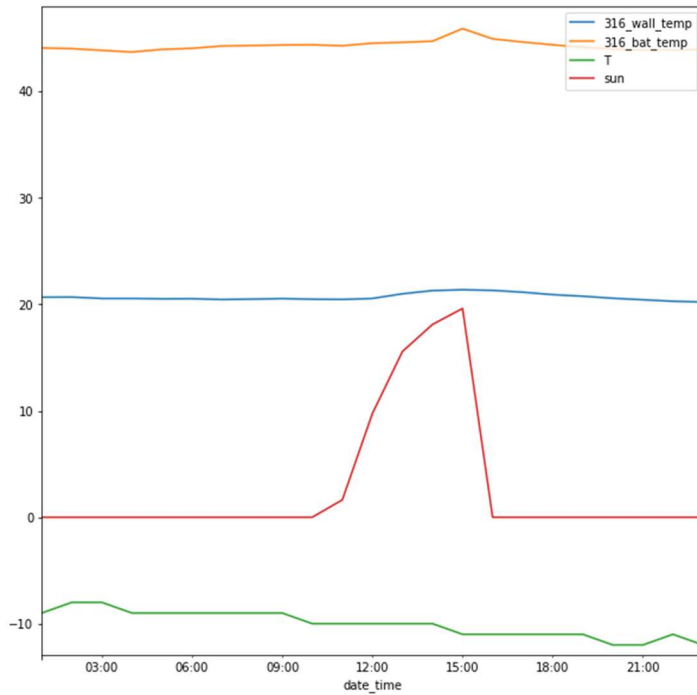


Рисунок 34. График зависимости температур от показателя солнца  
(316 кабинет 06.12.2020)

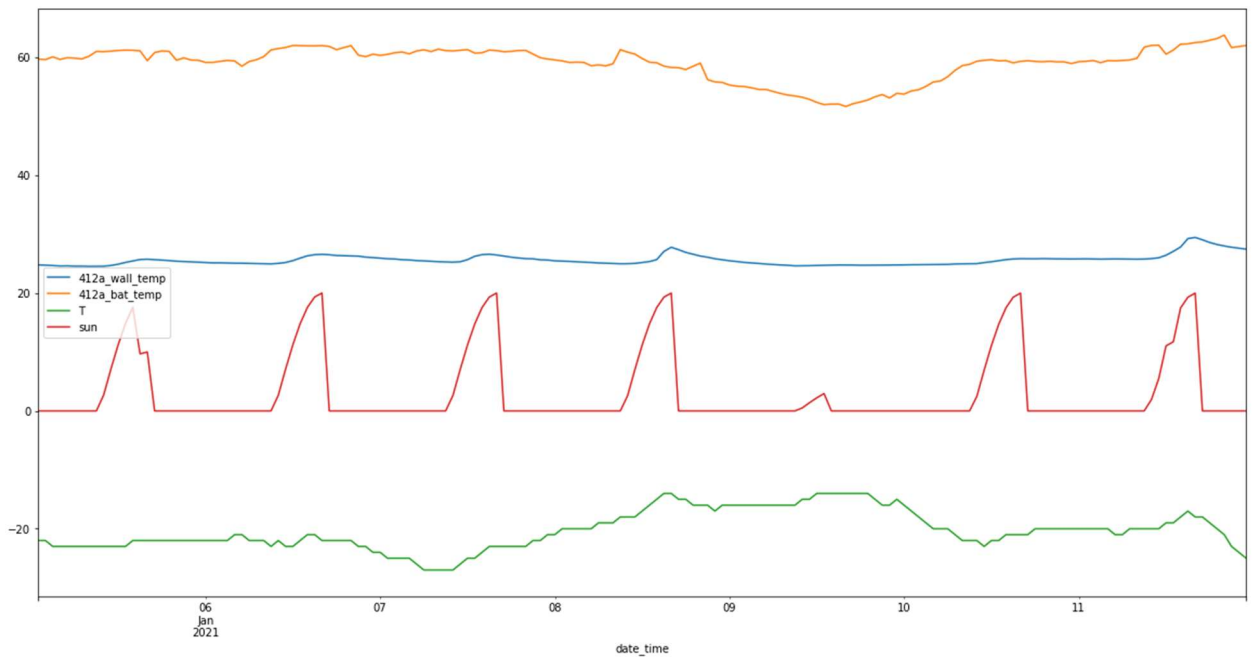


Рисунок 35. График зависимости температур от показателя солнца  
(412а кабинет 05.12.2021 – 12.01.2021)

На данных графиках заметно, что при слабом изменении температуры внешней среды и батарей и при высоком росте показателя влияния солнца температура внутри кабинета также возростала. На рисунке 34 наблюдается скачок температуры батарей. Но возрастание температуры помещения и

изменения “показателя влияния солнца” произошло раньше. Следовательно, в данном случае, повышение температуры внутри кабинета не было спровоцировано изменением температуры отопительных систем.

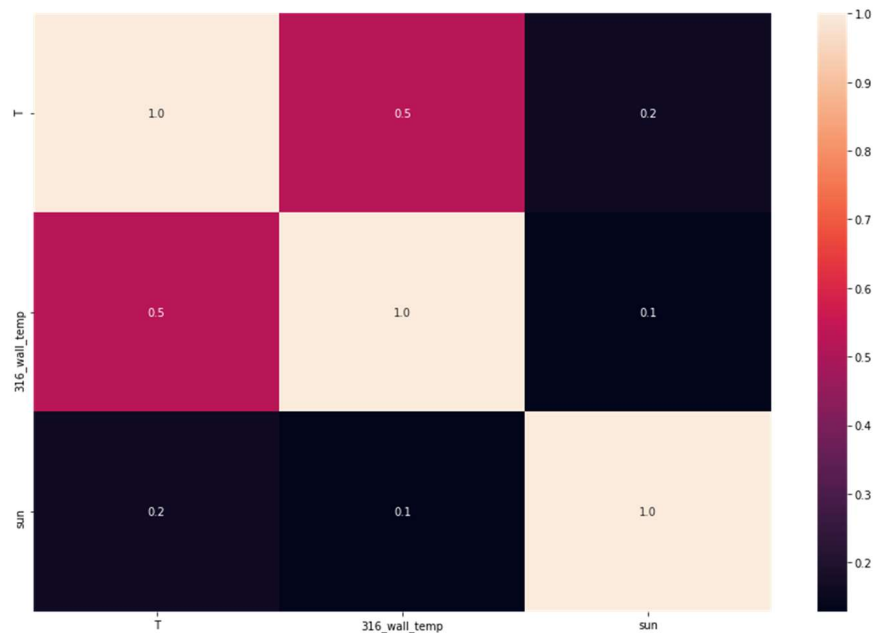


Рисунок 36. Матрица корреляции показателя солнца

Несмотря на то, что на графиках показатель влияния солнца хорошо себя показывает, на матрице корреляций (Рисунок 36) отображается весьма низкое значение коэффициента корреляции Спирмана.

### 2.1.3.2. АНАЛИЗ ВЛИЯНИЯ ВЕТРА

По аналогии с показателем влияния солнца был сконструирован показатель влияния ветра. Данный показатель рассчитывался следующим образом:

$$\text{wind\_speed} \times \cos((\text{wind\_dir} + 180) \bmod 360 - \alpha), \quad (2)$$

где  $\text{wind\_speed}$  – текущая скорость ветра,

$\text{wind\_dir}$  – текущее направление ветра,

$\alpha$  - азимут стены с окнами.

В отличие от азимута солнца, с которого оно светит, направление ветра действует в противоположную сторону, следовательно, его необходимо развернуть.

Была выдвинута гипотеза, что если ветер дует на стену с остеклением, то он должен повышать давление со стороны улицы, тем самым усиливать процесс инфильтрации за счёт увеличения разницы давлений. То есть при усилении ветра, температура падает. Чтобы исследовать эту гипотезу были построены графики зависимости температур от показателя ветра в холодные месяцы.

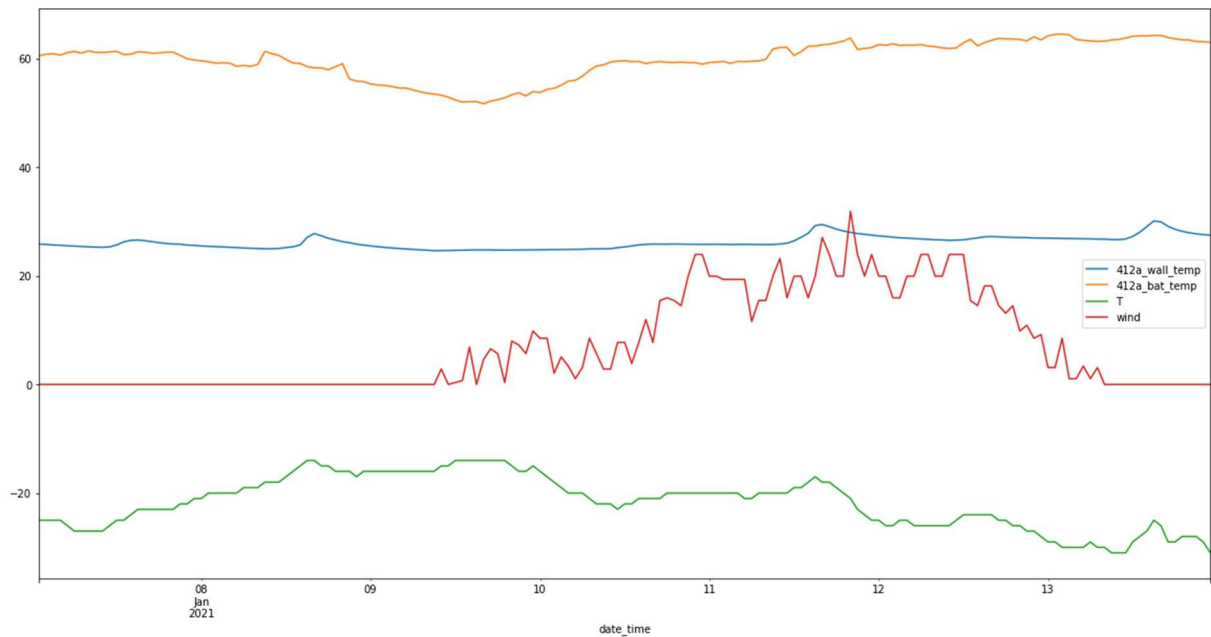


Рисунок 37. График зависимости температур от показателя ветра (412а кабинет 07.01.2021 – 14.01.2021)

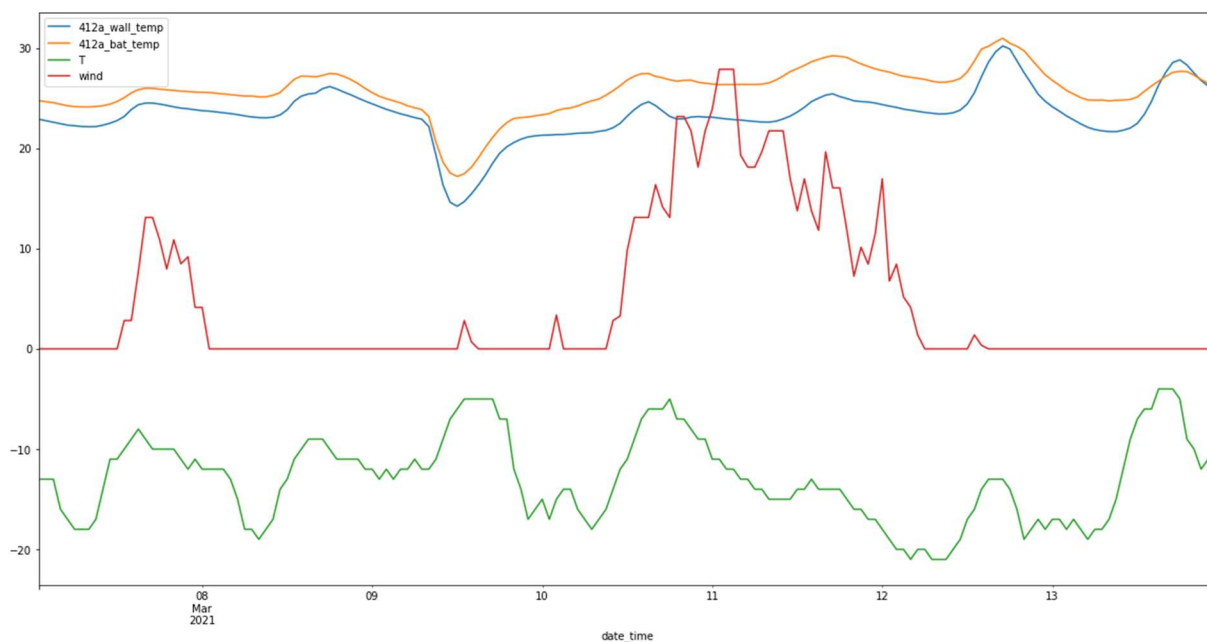


Рисунок 38. График зависимости температур от показателя ветра  
(412а кабинет 07.03.2021 – 14.03.2021)

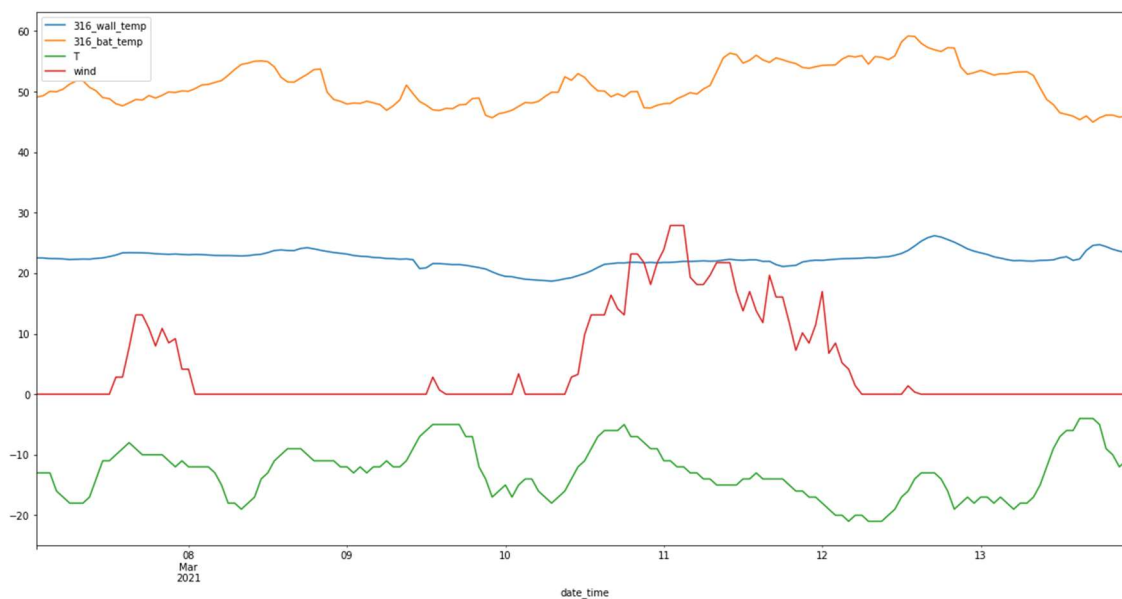


Рисунок 39. График зависимости температур от показателя ветра  
(316а кабинет 07.03.2021 – 14.03.2021)

На графиках видно, что при изменении показателя влияния ветра никаких существенных изменений температуры помещения не последовало (Рисунки 37–39). На основании данного подхода нельзя сказать, что ветер оказывал существенное влияния на температуру внутри кабинета.



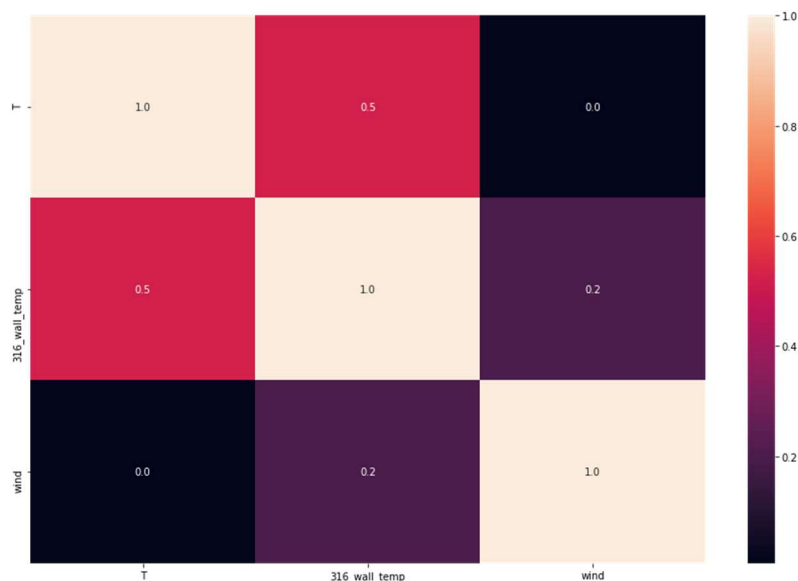


Рисунок 40. Матрица корреляции показателя ветра

Был построен график корреляции (по коэффициенту корреляции Спирмена) данного сконструированного признака с температурами помещения (Рисунок 40). Коэффициент корреляции довольно низкий ( $\sim 0.2$ ), что также указывает на слабое влияние ветра на температуру в помещении.

Был испробован еще один подход. Так как кабинеты ориентированы на разные стороны света, ветер может оказывать достаточное влияние на одни кабинеты при одном направлении ветра, и на другие при другом. Таким образом можно выдвинуть гипотезу, что при разном направлении ветра отзывались кабинеты разной пространственной ориентации. Принцип проверки данной гипотезы состоял в следующем. Рассматривались данные по кабинетам для низких температур наружного воздуха (например, от  $-21^{\circ}\text{C}$  до  $-27^{\circ}\text{C}$ ). Температуры кабинетов и скорость ветра группировались по направлениям ветра, после чего бралось их среднее значение по направлению. На основании информации о направлении наружных стен, анализировались изменения в температурных графиках. Скорость ветра была инвертирована для удобства просмотра, то есть при усилении ветра, график ветра должен падать, а температура в кабинетах, на стены которых дует ветер, предположительно, должна становиться ниже, чем у других кабинетов. Чтобы

исключить влияние солнца, брались данные только за ночные периоды времени.

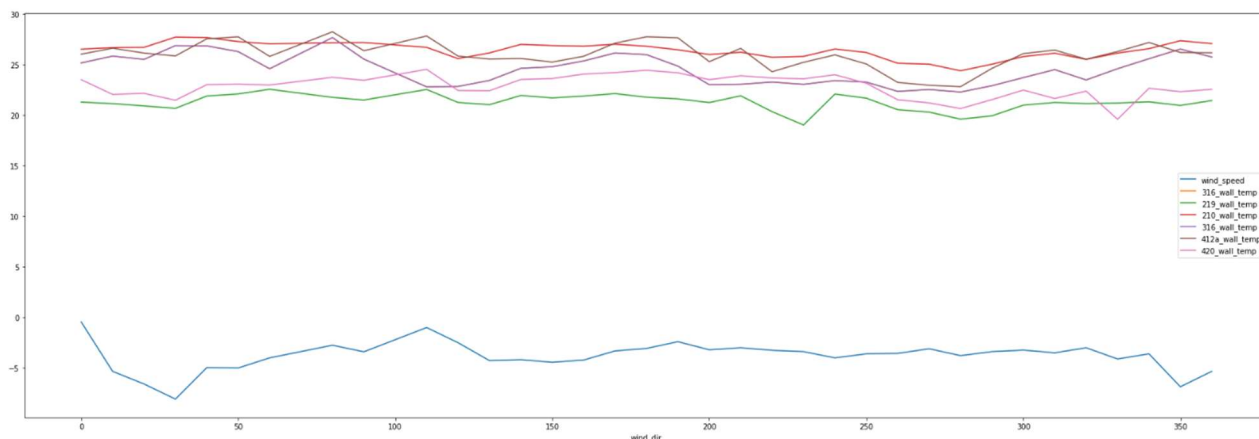


Рисунок 41. График зависимости показателя ветра от направления

На данном графике (Рисунок 41) заметно, что никакого серьёзного падения показателей температур ни у одного из кабинетов не наблюдалось, даже при высокой средней скорости ветра. По итогам исследования данный способ также не принёс результатов.

## 2.2. КЛАСТЕРИЗАЦИЯ ПОМЕЩЕНИЙ

Для кластеризации помещений была использована иерархическая кластеризация. В качестве алгоритма был выбран метод одиночной связи, то есть в качестве расстояния между двумя кластерами используется минимальное расстояние между двумя элементами разных кластеров. Кластеризация осуществлялась на основании близости температурных данных. В качестве метрики была использована корреляционная метрика.

Была выполнена группировка данных по месяцам и по отопительным/неотопительным сезонам. По марту 2020 года ближе всего оказались 316 и 412 кабинеты, к ним же в группу попал 210 кабинет, и 219 и 420 кабинеты определились в другой кластер (Рисунок 42). На температурных графиках (Рисунок 43) видно, что 412 и 316 кабинеты имеют схожие периоды возрастания и убывания. Также 412 кабинет схож по динамике температур с 219 кабинетом, но амплитуда изменений схожа меньше. 420 кабинет схож с 219 и 412, так как графики почти не пересекают друг друга, имеют схожие

периоды изменения, но разница температур меньше с 219 кабинетом. 210 кабинет по динамике температур ближе всего находится к 316 кабинету, не периоды так явно схожи, как в предыдущих случаях, в некоторых периодах разница между температур возрастала.

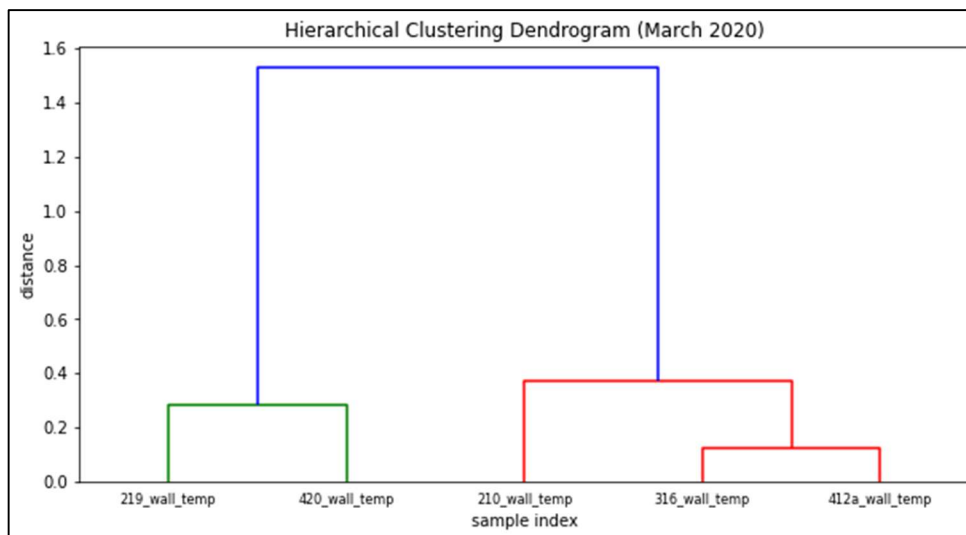


Рисунок 42. Кластеризация кабинетов по температурам за Март 2020

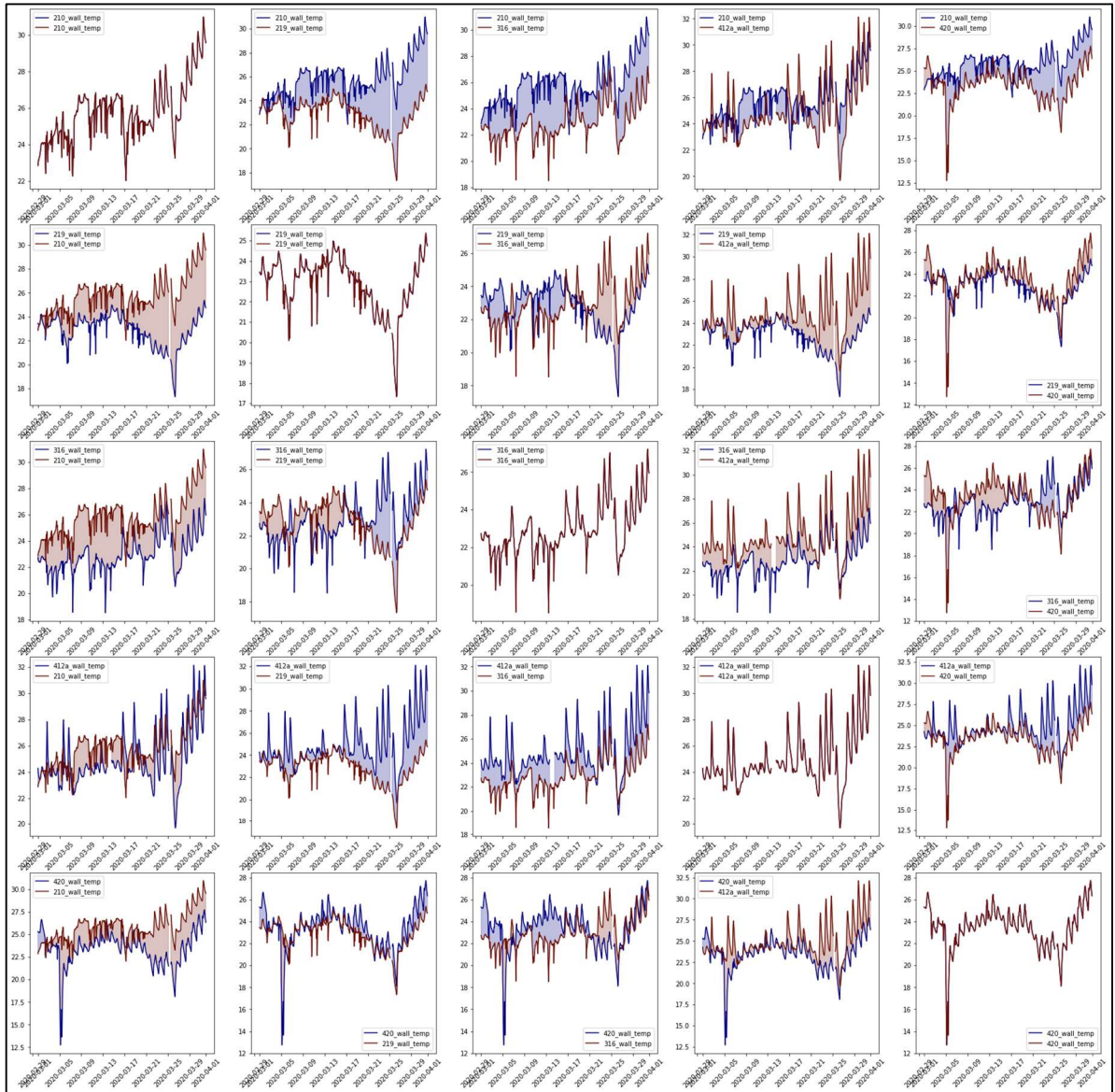


Рисунок 43. Сравнительная таблица температур кабинетов за Март 2020

По температурам февраля в результате кластеризации получились 3 группы: 219 и 420 кабинеты, 210 и 316, 412 кабинет (Рисунок 44). На температурных графиках периоды возрастания и убывания схожи у большинства графиков (Рисунок 45), 420 кабинет по амплитуде изменений и разнице температур больше схож с 219, а 316 кабинет имеет почти совпадающий по изменениям период, а также меньшую разницу температур. 412 кабинет имеет несколько пересекающихся промежутков с 210 и 316 кабинетами, по разнице температур и амплитуде изменений ближе всего к 420 кабинету.

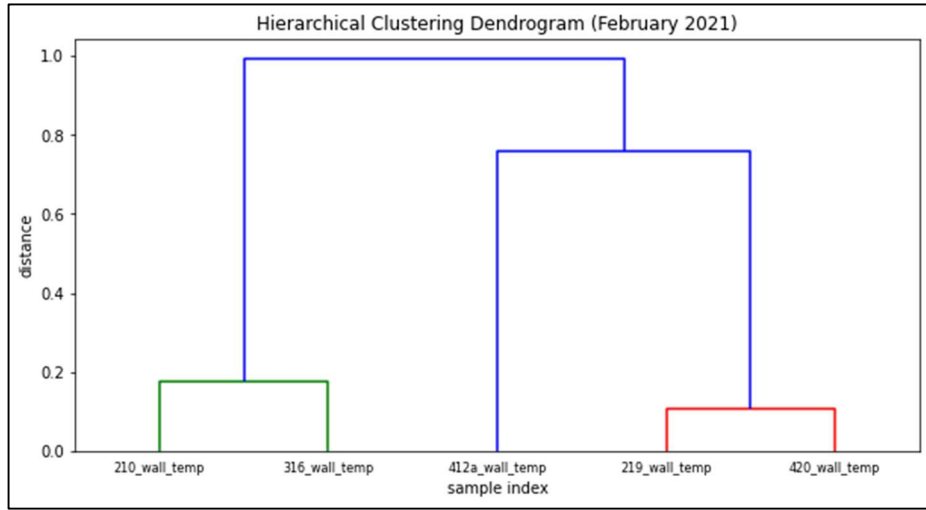


Рисунок 44. Кластеризация кабинетов по температурам за Февраль 2021

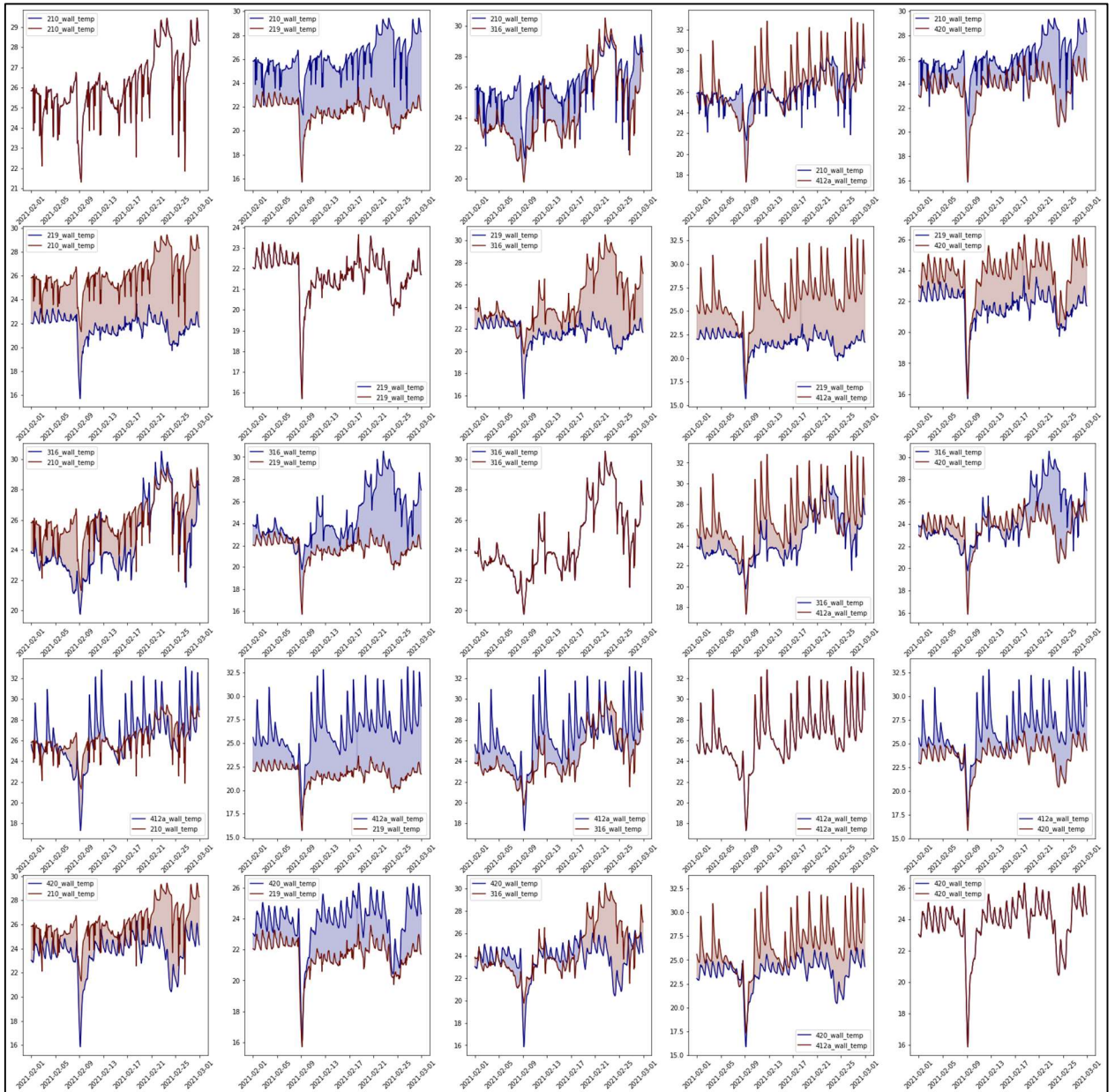


Рисунок 45. Сравнительная таблица температур кабинетов за Сентябрь 2020



Таким образом по разным временным периодам получились разные кластеры. Чтобы определить наиболее схожие кабинеты, была проведена кластеризация по каждому из имеющихся месяцев, а после посчитано количество попаданий каждого кабинета в один кластер с другим.

В результате за период с февраля 2020 года по ноябрь 2021 получились следующие кластеры (Рисунок 46): 219 кабинет чаще всего в одном кластере с 420 (40% случаев), 316 – 412 (50%), 210 чаще всего определялся в кластер с 316 и 210 кабинетами (40%), 420 кабинет чаще всего был один в кластере (50%), и из других кабинетов ближе всего к 219 кабинету.

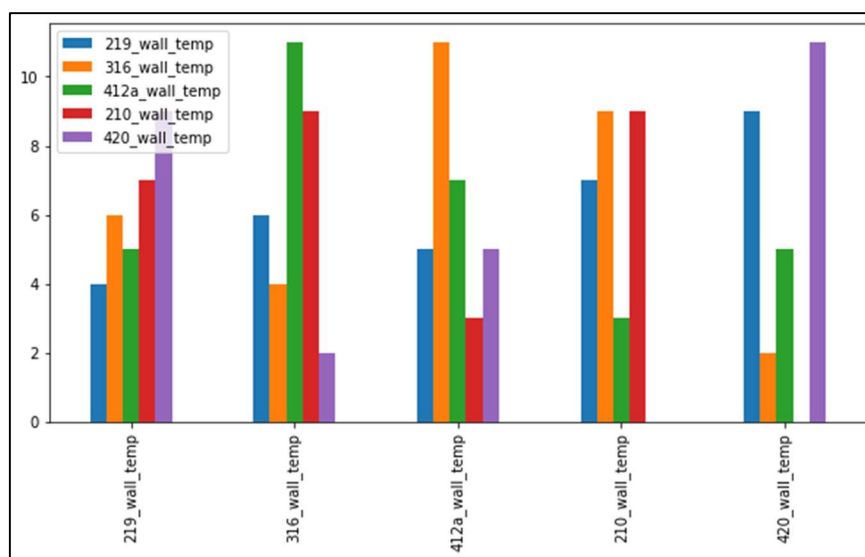


Рисунок 46. Число попаданий кабинетов в один кластер за период с февраля 2020 по ноябрь 2021

Также был проведен подсчет по отопительным и неотопительным сезонам за этот же период. В результате при выключенных батареях получились следующие кластеры (Рисунок 47): 219 и 420 кабинеты (55%), 316 и 412 (45%), 210 кабинет (55%). При включенных батареях (Рисунок 48): 420 кабинет (64%), 412, 316 и 210 (55%), 219 кабинет (36% случаев - с 210 кабинетом, 27% - с 316 и 420). На этих графиках можно заметить, что при кластеризации кабинетов по данным за неотопительный сезон кластеры определяются достаточно явно, для каждого помещения есть значение с преобладающим количеством попаданий в один кластер.

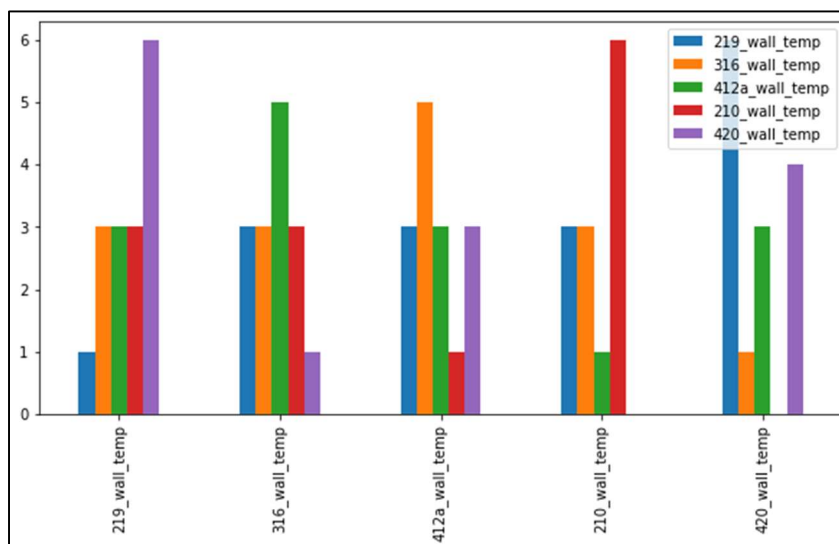


Рисунок 47. Число попаданий кабинетов в один кластер за неотапительные сезоны с февраля 2020 по ноябрь 2021

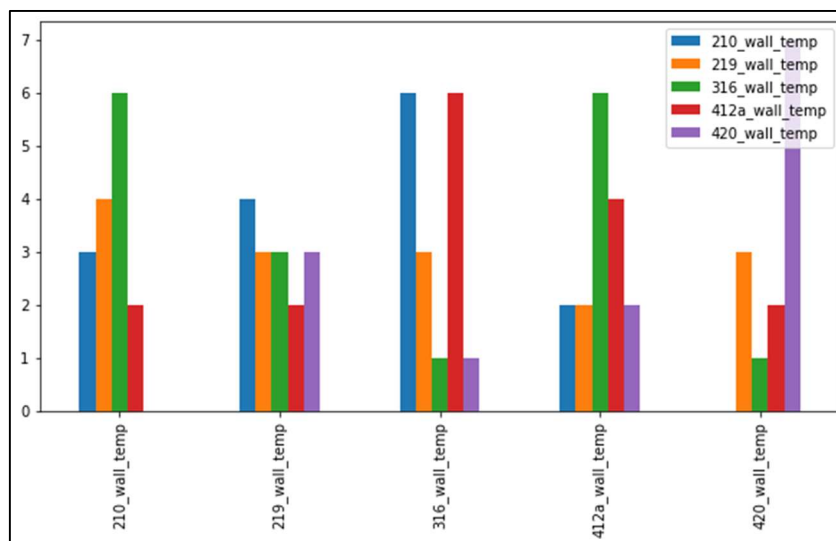


Рисунок 48. Число попаданий кабинетов в один кластер

### 2.3. ПОИСК АНОМАЛИЙ

Так как данная система мониторинга должна определять сильное отклонение температуры, возникает проблема поиска аномалий во временных рядах. Были рассмотрены aggregation и predicted-based подходы поиска аномалий во временных рядах.

#### 2.3.1 AGGREGATION ПОДХОД ПОИСКА АНОМАЛИЙ

В качестве aggregation алгоритма был использован фильтр нижних частот, суть данного метода заключается в следующем:

1. Расчёт скользящего среднего временного ряда.

2. Расчёт  $Z$  - показателя каждой точки временного ряда.
3. Поиск данных временного ряда, которые находятся на расстоянии более некоторого порогового значения от скользящего среднего.

Был выполнен расчет центрированного скользящего среднего разной длины,  $Z$  – показатель был рассчитан как стандартное отклонение в каждой точке, в качестве порогового значения рассматривалось некоторое количество стандартных отклонений. Исследуемый временной ряд содержит данные по температурам 412 кабинета за период с февраля 2020 года по ноябрь 2021 года.

При расчёте центрированного скользящего среднего с длиной в 24 часа и пороговым значением равным 2.5 стандартных отклонений (Рисунок 49) были найдены аномалии только в период с марта по июнь 2021 года (Рисунок 50), в вершинах графика, причем преобладающее большинство значений – аномально высокая температура, аномально низких температур найдено не было.

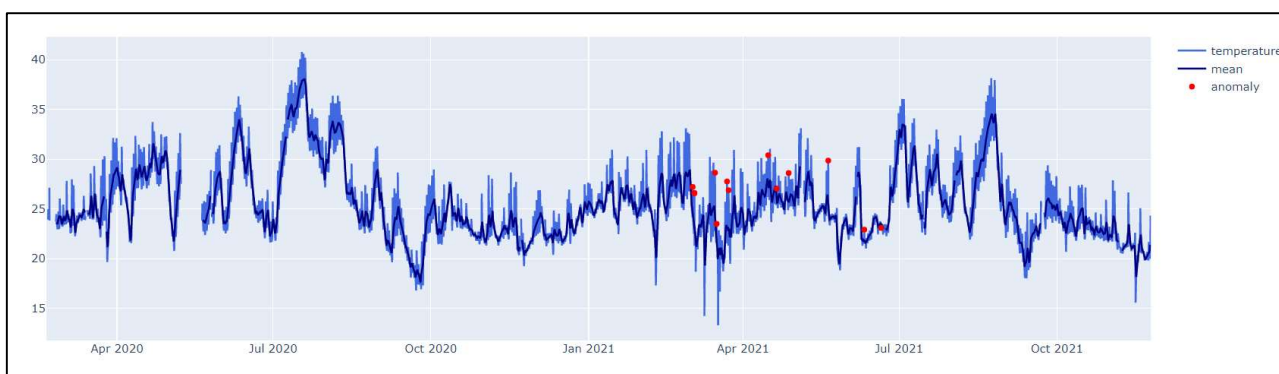


Рисунок 49. Поиск аномальных температур 412 кабинета (Aggregation подход, length – 24 час, std – 2.5)



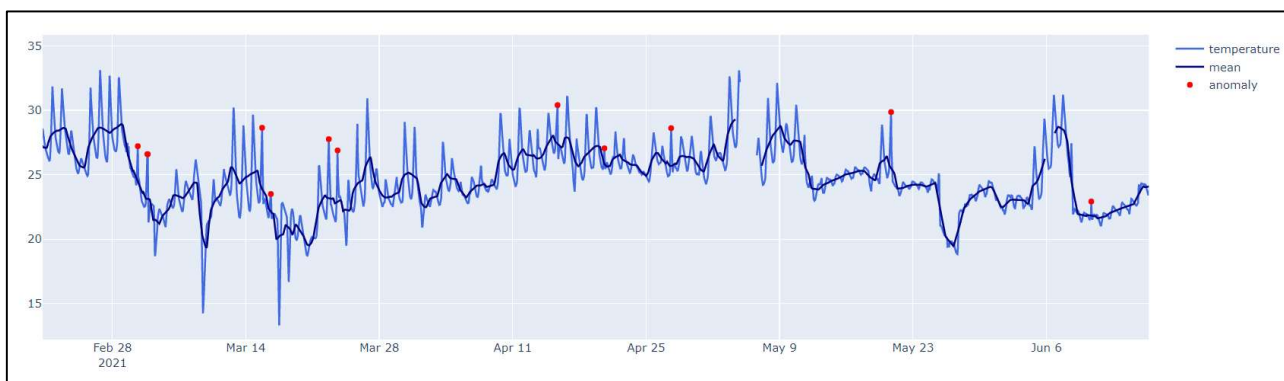


Рисунок 50. Аномальные температурные данные 412 кабинет (март — июнь 2021) (Aggregation подход, length – 24 часа, std – 2.5)

При увеличении длины центрированного среднего до 48 часов (Рисунок 51), количество найденных аномалий увеличилось, преобладающее большинство аномалий – также высокая температура, была обнаружена 1 аномально низкая температура (17 марта), хотя на графике имеются еще 3-4 сильных спадов температур.

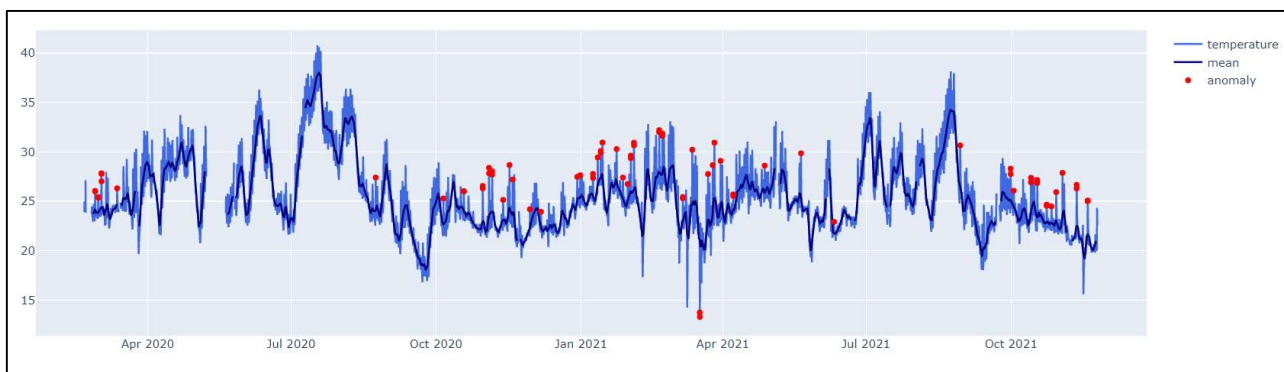


Рисунок 51. Аномальные температурные данные 412 кабинет (Aggregation подход, length – 48 часов, std – 2.5)

При увеличении длины центрированного среднего до 96 часов, и порогового значения до 3 стандартных отклонений (Рисунок 52), скользящее среднее заметно сгладилось, были обнаружены сильные понижения температур.

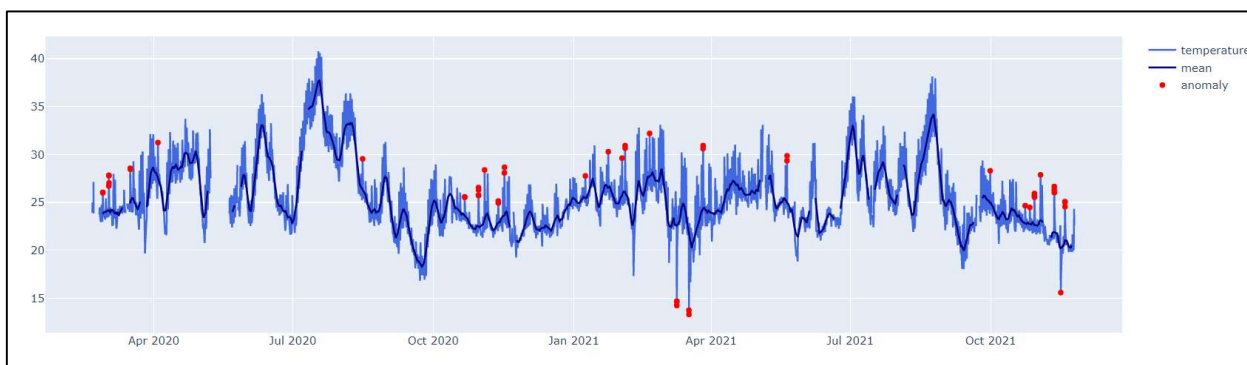


Рисунок 52. Аномальные температурные данные 412 кабинет  
(Aggregation подход, length – 96 часов, std – 3)

Минус данного алгоритма заключается в том, что из-за пропусков в данных некоторые подпоследовательности временного ряда пропускаются, поэтому при увеличении длины центрированного среднего, уменьшится количество найденных аномалий, некоторые будут просто пропущены. Помимо этого, в данном подходе не учитываются определенные особенности данных, например, повышение температуры в помещении к полудню летом, что является нормальным явлением для данного сезона.

### 2.3.2 PREDICION-BASED ПОДХОД ПОИСКА АНОМАЛИЙ

Predicion-based подход состоит из следующих шагов:

1. Построить модель предсказания температуры в помещении.
2. Определить порог значений отклонений действительной температуры от спрогнозированной.
3. Найти наблюдения, отклонение от прогноза которых больше порогового значения.

#### 1 набор признаков

Была построена рекуррентная LSTM нейросеть для прогнозирования температуры в 412 кабинете. Для обучения и тестирования модели был использован датасет, содержащий информацию о погодных условиях внешней среды, температур кабинета (воздуха и батарей) за период с февраля 2020 по ноябрь 2021.

Был сформирован следующий набор признаков:

- temp – температура в 412 кабинете (double);
- hour – час измерения (int);
- is\_daylight – измерение выполнено днем или ночью (bool);
- weekday – день недели измерения (int);
- is\_weekend – является ли день недели выходным днем (bool).

Далее датасет был нормализован и были сформированы обучающий и тестовый выборки, которые для каждого значения температуры содержат 50 предыдущих значений признаков. Таким образом каждый датасет — 3-мерный массив размерности  $n \times h \times k$ , где  $n$  – количество элементов выборки,  $h$  – количество исторических данных,  $k$  – количество признаков.

Для прогнозирования температуры в 412 помещении была построена следующая рекуррентная LSTM нейросеть:

1. входной LSTM рекуррентный слой с dropout, размерность входных данных равна количеству признаков ( $k$ ), размерность выходных – количество исторических данных ( $h$ );
2. скрытый LSTM слой со 100 внутренними узлами, dropout;
3. скрытый плотно связанный слой с 1 выходным значением и линейной функцией активации.

Модель была обучена обучающей выборки размером 4750 наборов данных, на 10 эпохах (Рисунок 53). Итоговое значение RMSE - 0.0267.

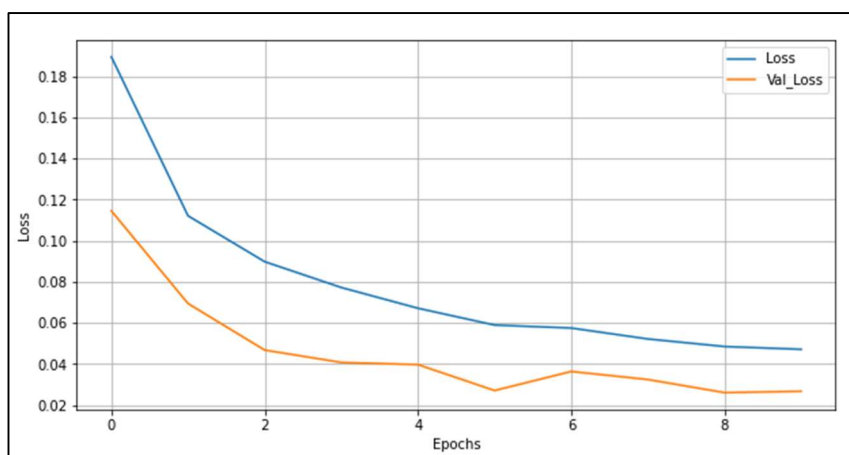


Рисунок 53. Обучение LSTM модели (1 набор признаков)

На тестовой выборке значение  $RMSE = 0.162$ , модель оказалась довольно точной, но есть некоторое различие между фактическим и предсказанным значением в вершинах графика (Рисунок 54). Скорее всего амплитуду данных скачков температуры помещения определяло влияние внутренних или внешних факторов.

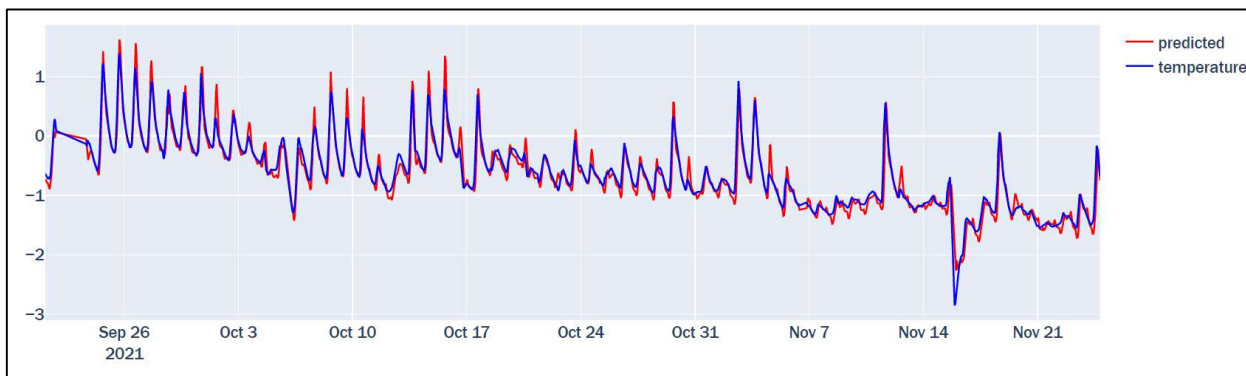


Рисунок 54. Тестирование LSTM модели (1 набор признаков)

## 2 набор признаков

Для формирования 2 набора признаков к 1 набору были добавлены данные о температурах батарей 412 кабинета (признак `bat_temp (double)`). Модель также была обучена обучающей выборки размером 4750 наборов данных, на 10 эпохах (Рисунок 55). Итоговое значение  $RMSE = 0.0435$ .

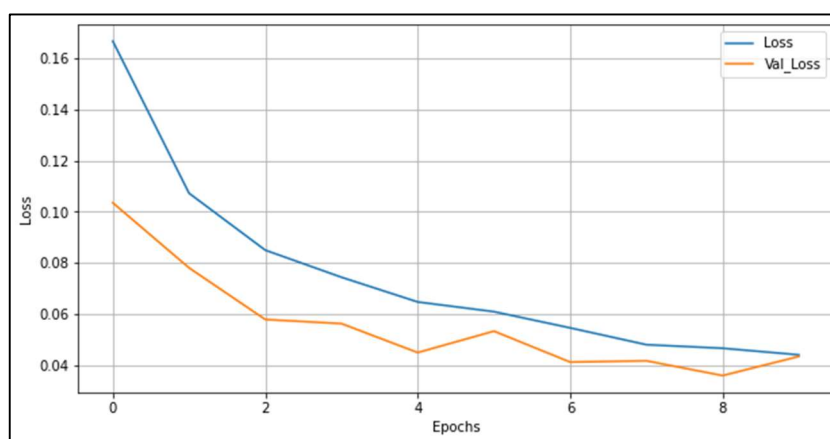


Рисунок 55. Обучение LSTM модели (2 набор признаков)

На тестовой выборке значение  $RMSE = 0.161$ , удалось уменьшить разницу между предсказанным и фактическим значениями в некоторых вершинах графика (Рисунок 56), но значительно точность модели не изменилась.

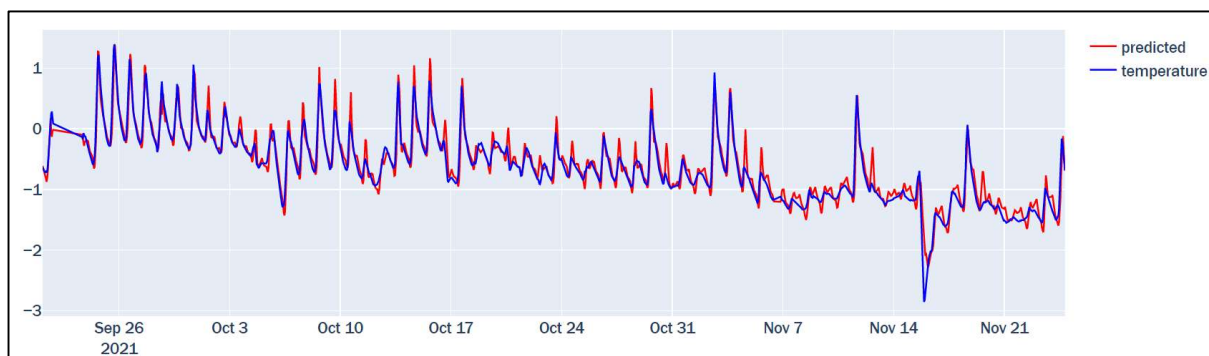


Рисунок 56. Тестирование LSTM модели (2 набор признаков)

### 3 набор признаков

Для формирования 3 набора признаков ко 2 набору были добавлены данные по климатическим условиям внешней среды:

- T – температура наружного воздуха (double);
- altitude – высота солнца (double);
- azimuth – азимут солнца (double).

Так как были добавлены признаки, которые описывают положение солнца, признак `is_daylight` был удален. Таким образом 3 набор содержит 8 признаков. Обучение проводилось выборке того же размера, на 10 эпохах (Рисунок 58). Итоговое значение  $RMSE = 0.0412$ .

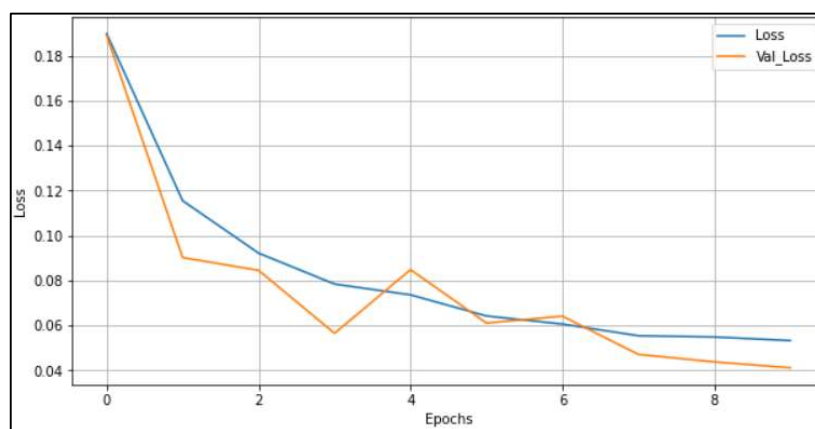


Рисунок 57. Обучение LSTM модели (3 набор признаков)

На тестовой выборке значение  $RMSE = 0.185$ , удалось значительно уменьшить погрешность прогнозирования в вершинах графика, но в период примерно с 7 по 25 ноября предсказанная температуры была немного выше фактической (Рисунок 58). Возможная причина в том, что в эти дни была высокая облачность и солнце не оказывало сильное влияние на температуру,

так как амплитуды колебаний фактической температуры в этот период значительно меньше.

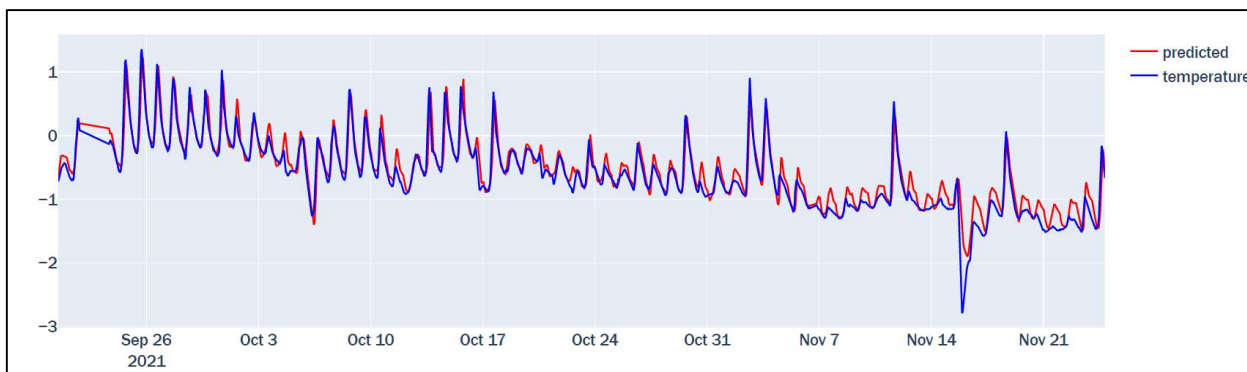


Рисунок 58. Тестирование LSTM модели (3 набор признаков)

### Поиск аномальных данных

Следующий шаг – определение порогового значения, было решено считать аномальными температурами – наиболее отдаленные от прогноза наблюдения. Для каждой точки данных был выполнен расчет Z-показателя, в данном случае это разница между фактическим и предсказанным значением. Данные показатели сортируются по возрастанию и в качестве порогового значения берется  $k$  по счету элемент. Номер элемента  $k$  определяется количеством наблюдений в датасете  $n$  и коэффициентом  $c$  ( $0 < c < 1$ ), который задается в качестве параметра функции, ( $k = c \cdot n$ ).

При коэффициенте  $c$  равным 0.005 и количестве тестируемых данных в 6250, было найдено около 20 аномалий (Рисунок 59). Некоторые точки похожи на аномальные значения, но в некоторых местах были найдены аномалии, похожие на нормальные значения температур (Рисунок 60). Возможная причина в погрешности модели прогнозирования.

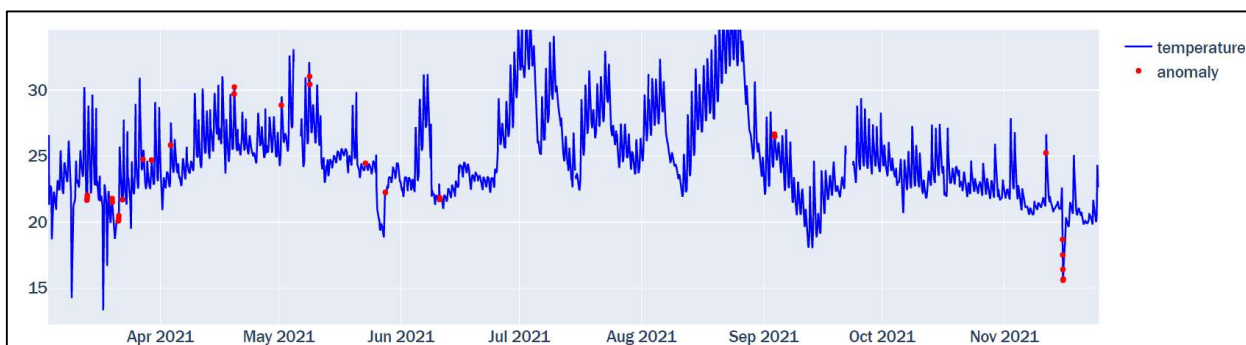


Рисунок 59. Аномальные температурные данные 412 кабинет  
(Prediction-based подход,  $n = 6250$ ,  $c = 0.005$ )

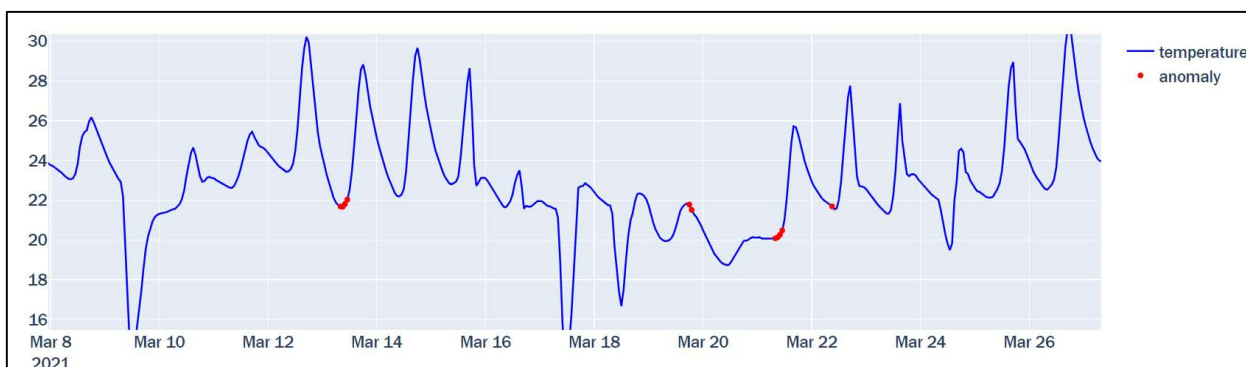


Рисунок 60. Аномальные температурные данные 412 кабинет за март 2021  
(Prediction-based подход,  $n = 6250$ ,  $c = 0.005$ )

## 2.4. ПОСТРОЕНИЕ МОДЕЛЕЙ ДЛЯ КЛАСТЕРОВ

Были определены следующие возможные причины аномалий:

- изменение температуры батарей в помещении
- проветривание

Так как изменение температуры батарей может служить причиной аномалии, было решено убрать эти данные для обучения модели. Таким образом для обучения моделей для поиска аномалий был сформирован следующий набор признаков:

- temp – температура в кабинете (double);
- hour – час измерения (int);
- weekday – день недели измерения (int);
- is\_weekend – является ли день недели выходным днем (bool);
- T – температура наружного воздуха (double);
- altitude – высота солнца (double);



- azimuth – азимут солнца (double).

Для каждого найденного кластера кабинетов была построена модель предсказания температур. Обучение производилось на данных каждого кабинета, входящих в один кластер на 10 эпохах для каждого помещения.

Также было проведено сравнение результатов предсказаний для моделей, построенных на данных одного кабинета, кластеров и модели, обученной на всех кабинетах.

## 2.5. СРАВНЕНИЕ МОДЕЛЕЙ

В сравнении моделей приведены результаты сравнений для тёплого и для холодного сезонов для удобства исследования, при этом данные на которых обучались модели, на сезоны не делились. Также, в данные для обучения моделей были включены только те периоды времени, когда здание активно эксплуатировалось, то есть в них не было включено время закрытия корпуса на карантин в связи с COVID-19.

На рисунках приведены графики температуры в помещении (синим), предсказанная с помощью модели температура в помещении (красным), а также графики, вероятно влияющих на температуру в помещении величин, таких как температура батарей (коричневый) и температура на улице (бирюзовый).

### Теплый сезон

Для начала были рассмотрены различия в моделях на примере тёплых сезонов.

#### Аудитория 412

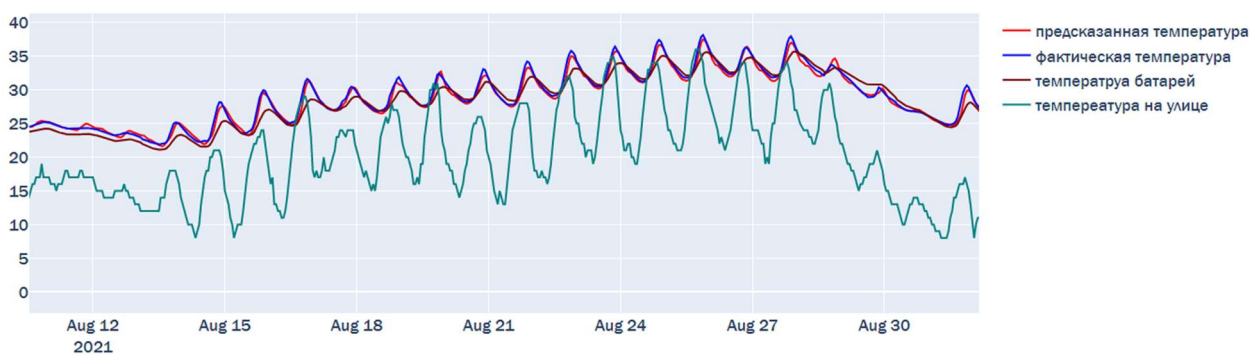


Рисунок 61. Каб. 412а и модель, обученная на 412а



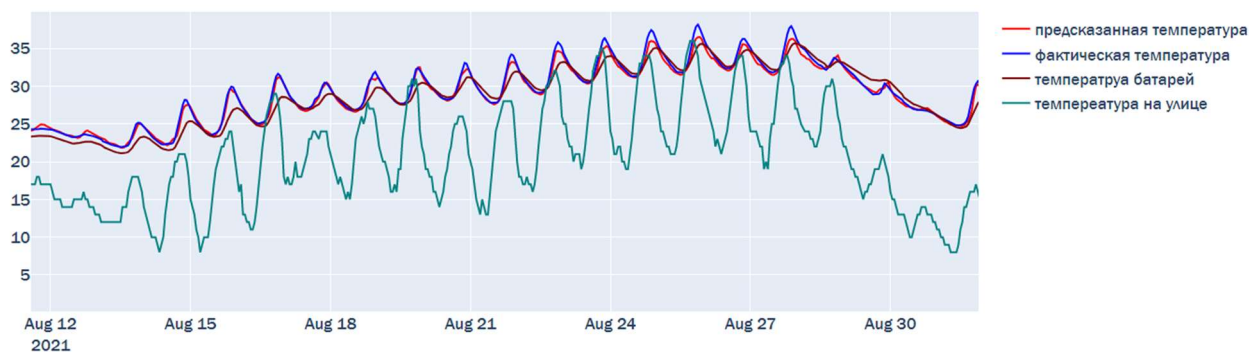


Рисунок 62. Каб. 412a и модель, обученная на кластере (412 и 316)

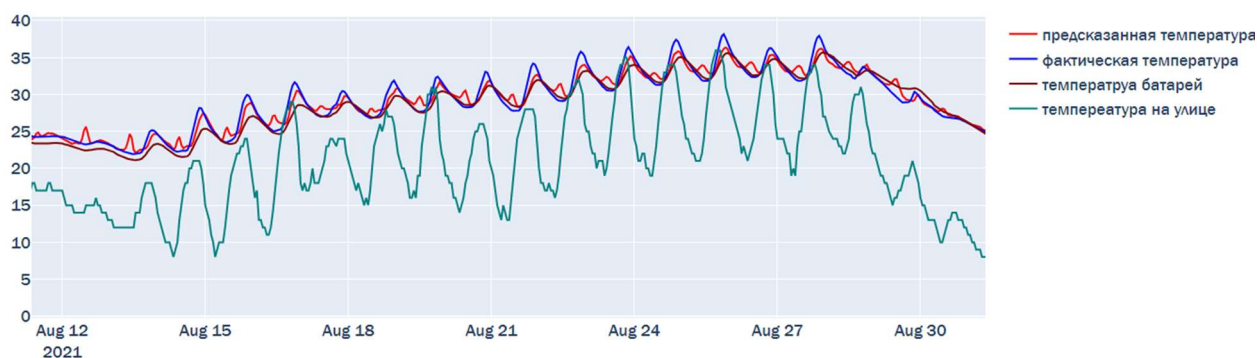


Рисунок 63. Каб. 412a и модель, обученная на всех кабинетах

На рисунках 61-63 приведены результаты предсказания температуры в 412 аудитории, для моделей, обученных на данных по 412 кабинету, на данных по кластеру (412 и 316) и на данных по всем кабинетам. Можно заметить, что нет существенных различий в предсказаниях модели для 412 кабинета и для кластера. Но при этом в предсказаниях, сделанными моделью, обученной на всех кабинетах, есть существенные ошибки. Модель пыталась предсказать ещё некий скачок температур, которого не было в данном кабинете.

### Аудитория 316

Почти аналогичную ситуацию можно наблюдать и в предсказаниях температуры в 316 аудитории, за тем исключением, что модель, построенная для кластеров ожидала более сильных скачков температуры, чем было фактически и чем предсказала модель, построенная на данных 316 кабинета.



Рисунок 64. Каб. 316 и модель, обученная на 316

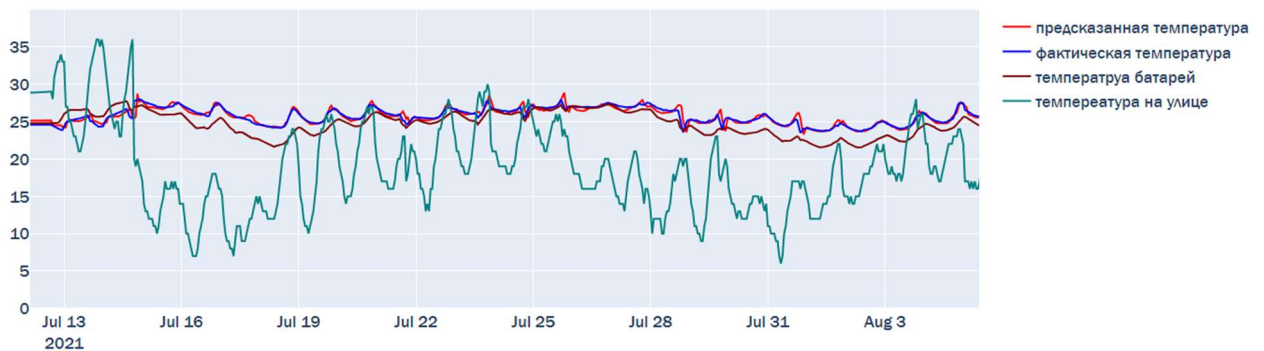


Рисунок 65. Каб. 316 и модель, обученная на кластере (412 и 316)



Рисунок 66. Каб. 316 и модель, обученная на всех кабинетах

### Аудитория 210

Так как 210 кабинет один в своём кластере, то для него была построена модель только на данных от аудитории 210, а также модель, обученная на данных по всем кабинетам. Можно заметить, что предсказанная амплитуда возрастания температуры на рисунке 68 немного выше, чем фактическая и чем предсказанная первой моделью.

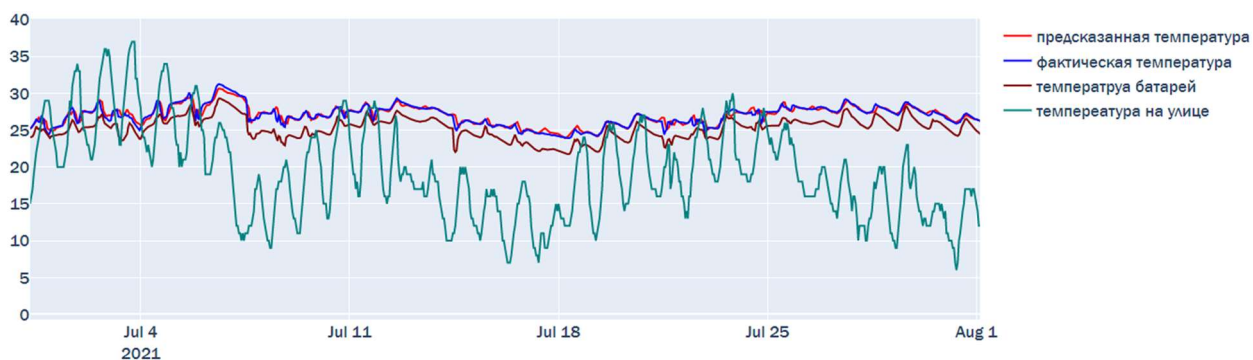


Рисунок 67. Каб. 210 и модель, обученная на 210

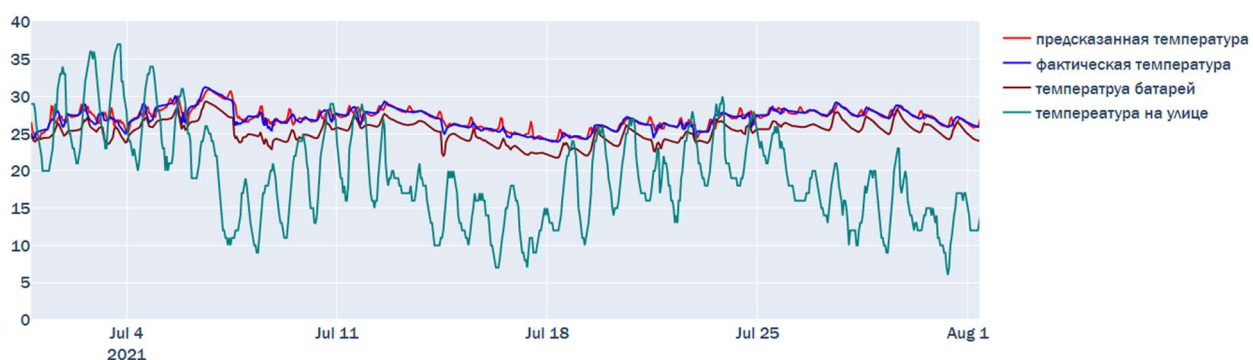


Рисунок 68. Каб. 210 и модель, обученная на всех кабинетах

### Аудитория 219

На рисунках 69-71 выведены графики для 219 кабинета, для данного кабинета картина противоположная. Хуже всех себя показывает обученная на кластере (219 и 420) модель, чуть лучше модель, обученная на кластере, и лучше всех модель, обученная на данных по всем кабинетам. Возможно это происходит из-за не совсем точного подбора по кластерам и/или недостаточного количества данных для обучения (модель, обученная на всех кабинетах, имеет в 5 раз больше данных). Результаты прогноза различных моделей для 420 аудитории аналогичные.

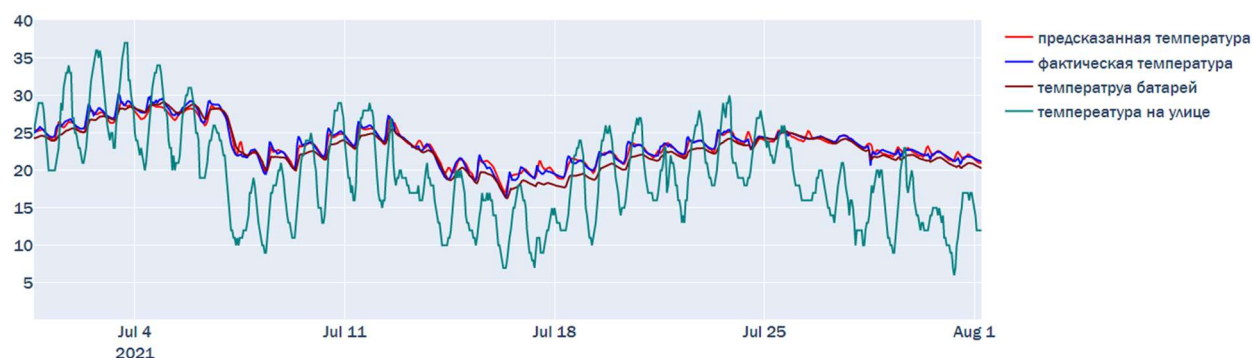


Рисунок 69. Каб. 219 и модель, обученная на 219

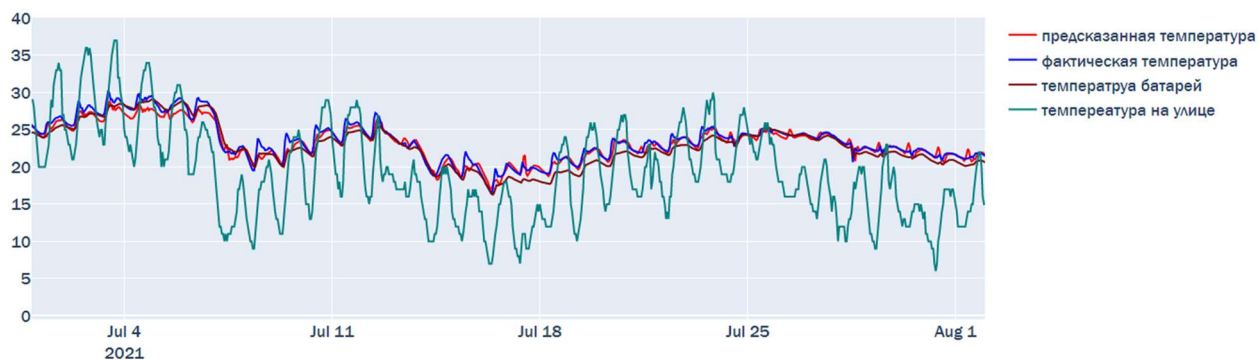


Рисунок 70. Каб. 219 и модель, обученная на кластере (420 и 219)



Рисунок 71. Каб. 219 и модель, обученная на всех кабинетах

### Холодный сезон

Также были рассмотрены различия в моделях на примере холодного сезона. В данном случае было не так много данных для обучения в связи с тем, что не были использованы данные, собранные за время закрытия корпуса на карантин в связи с COVID-19.

### Аудитория 412

На рисунках 72-74 приведены результаты предсказания температуры в 412 аудитории, для моделей, обученных на данных по 412 кабинету, на данных по кластеру (412 и 316) и на данных по всем кабинетам. В целом, предсказание модели, обученной на данных по всем кабинетам, оказалось чуть более точным, но в моменты значительных изменений температур батарей кластерная модель показала себя немного лучше.



Рисунок 72. Каб. 412 и модель, обученная на 412



Рисунок 73. Каб. 412 и модель, обученная на кластере (412 и 316)



Рисунок 74. Каб. 412 и модель, обученная на всех кабинетах

### Остальные аудитории

Для остальных аудиторий модель, обученная на всём массиве данных, показала себя лучше, предположительно, из-за недостаточного количества данных за отопительный сезон. К примеру, на рисунках 75 и 76 приведены результаты прогнозирования для 210 кабинета.



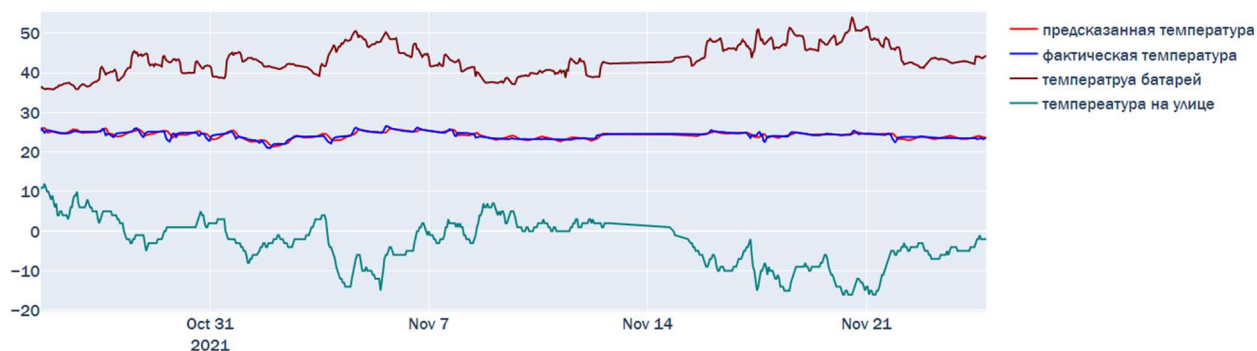


Рисунок 75. Каб. 210 и модель, обученная на 210

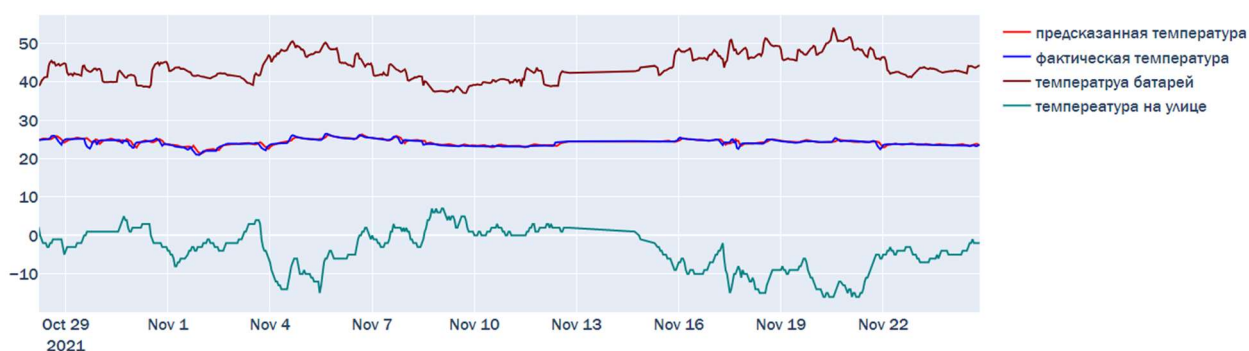


Рисунок 76. Каб. 210 и модель, обученная на всех кабинетах

В результате сравнения моделей было принято решение использовать для системы мониторинга, так как они лучше показали себя на теплом сезоне, чем модель с общими данными, но при этом не были хуже, чем модели, обученные индивидуально. Следовательно, можно уменьшить количество моделей не сильно проигрывая в качестве, используя модели для кластеров. Что касается холодного периода времени, то модель нуждается в дообучении, по соответствующим данным. Также в дальнейшем, в качестве эксперимента, можно обучить модели на имеющихся данных в условия отсутствия людей (карантинное время).

## 2.6. ПОИСК АНОМАЛИЙ ДЛЯ КЛАСТЕРНЫХ МОДЕЛЕЙ

Был выполнен поиск аномалий для всех аудиторий с использованием кластерных моделей. При коэффициенте с равным 0.01 и количестве тестируемых данных в 6250, было найдено около 50 аномалий (Рисунки 66, 67) Большинство точек были найдены в периоды резких возрастаний и резких убываний температуры.

### Кластер 2: 316 и 412 кабинеты

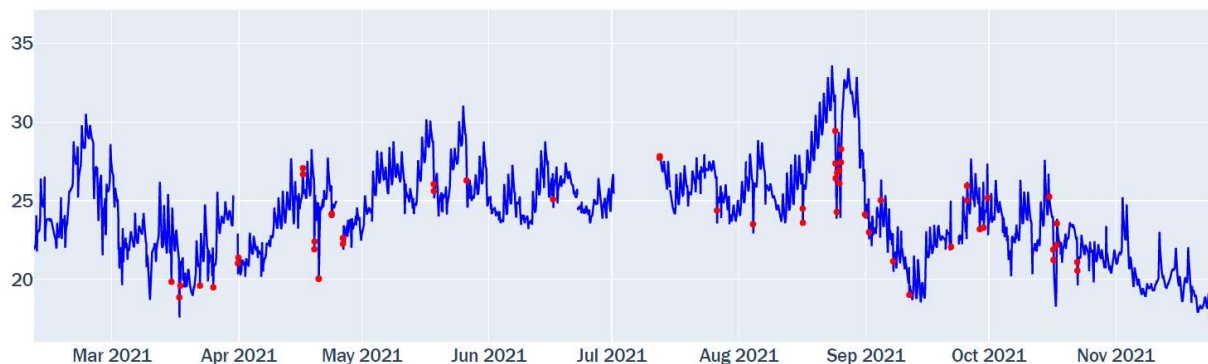


Рисунок 77. Найденные аномалии по 316 кабинету (кластерная модель)

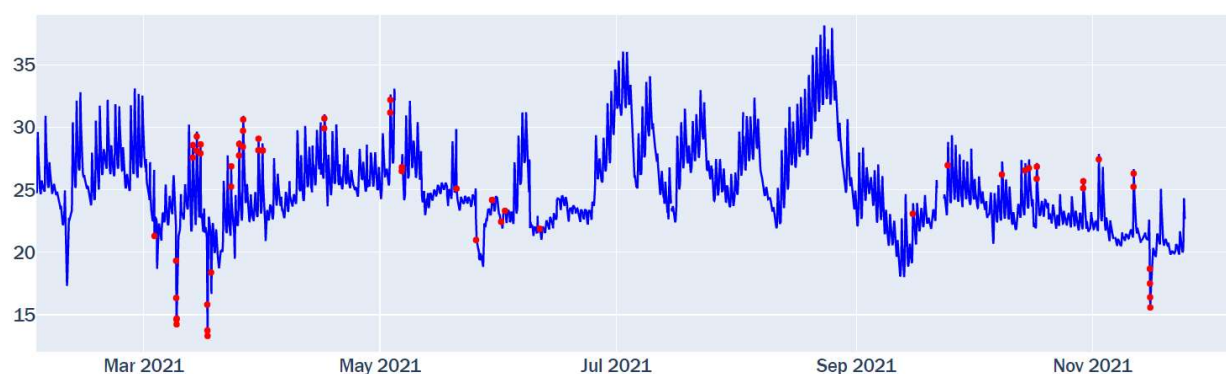


Рисунок 78. Найденные аномалии по 412 кабинету (кластерная модель)

## 2.7 ПРИЧИНЫ АНОМАЛИЙ

Используя информацию о найденных моделью аномалиях, был произведён поиск их причин. Данный поиск осуществлялся по следующему принципу, отдельно искались промежутки времени, в которых могли бы произойти перепады температур для каждой причины (в случае батарей, это отключение отопления), а затем, данные причины пересекались с найденными вышеуказанными моделями аномалиями.

### 2.7.1 ОТКЛЮЧЕНИЕ БАТАРЕЙ

Одной из причин падения температуры была определено отключение отопления в аудитории. Соответственно происходило падение температуры на соответствующих датчиках температур. Сложность заключалась в том, что температура в разных кабинетах поддерживается на разном уровне. В некоторых кабинетах температура батарей держится на уровне в 60-70 °С, в то время как в других кабинетах на уровне 30-40 °С при одинаковых

температурах в кабинете. Так что задать температуру барьерными значениями не представлялось возможным.

Поиск аномалии происходил следующим образом. Производился поиск разницы в температуре батарей текущего измерения с предшествующим, таким образом вычисляя колебания в температуре батарей. Далее выполнялся поиск среднего квадратичного отклонения. За аномалии считаются отклонения, превышающие среднее квадратичное отклонение в 3 раза. На рисунке 79, можно увидеть, как реагировал график отклонений температур в момент отключения и включения батарей.

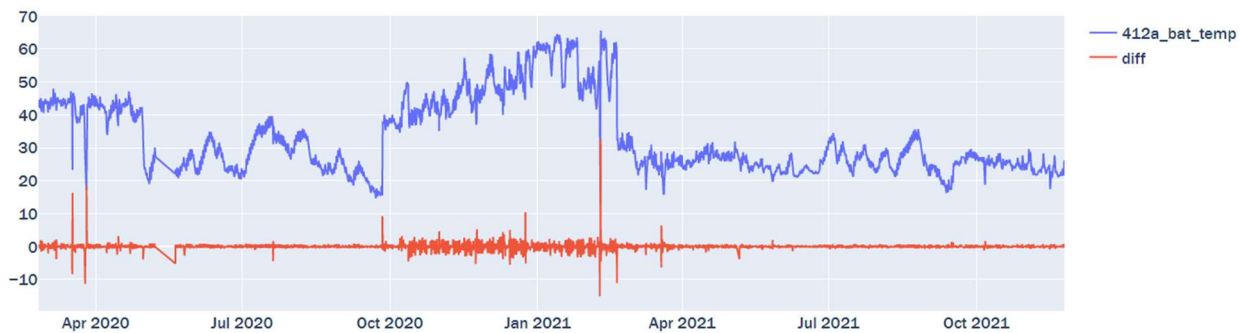


Рисунок 79. 412 кабинет. Температура батарей и график колебаний температуры

В конечном результате мы получаем набор данных с пометкой о том, что в данное время для кабинета могла быть аномалия, связанная с отключением отопления в помещении. (Рисунок 80)

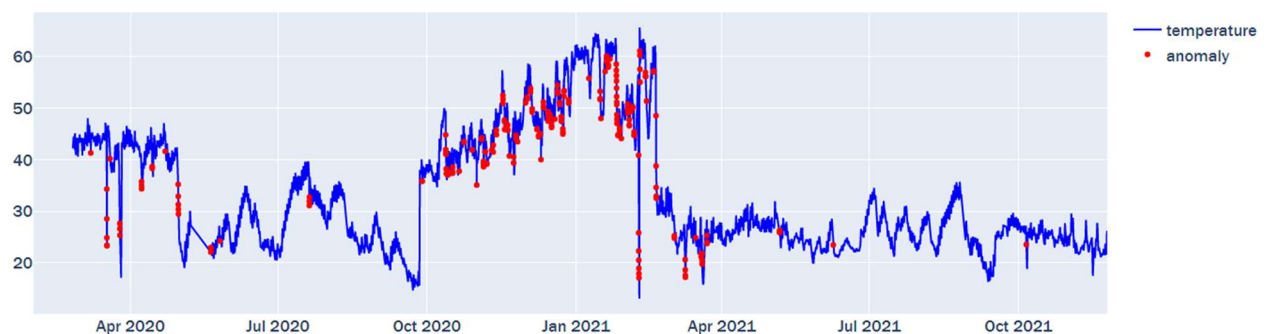


Рисунок 80. Температура батарей в 412 кабинете. Аномалии связанные с отоплением.



## 2.7.2 ПРОВЕТРИВАНИЯ

### 2.7.2.1 Эксперимент

Проветривание кабинетов, также было определено одной из возможных причин падения температуры воздуха в помещении. С целью выяснить, как разные кабинеты реагируют на открытие окон при низкой внешней температуре, был поставлен эксперимент по выстуживанию кабинетов с замером показаний температуры воздуха. Данный эксперимент проводился в двух кабинетах, в аудитории номер 412 и на кафедре университета в 210 кабинете. Эксперимент проводился 29 января. На момент его проведения температура наружного воздуха была -15 градусов Цельсия. Для замеров температуры использовались датчики температуры в кабинете, а также для примерной оценки был взят неточный термометр, температуру на котором можно было наблюдать без подключения к датчикам ТюмГУ.

#### 210 аудитория

В 210 кабинете открытие окон произошло в 15:09 на термометре была температура 25 градусов Цельсия. Датчик температуры давал показания 25.31 градуса Цельсия. В 15:35 температура на термометре была 25 градуса Цельсия, в то время как датчик показывал температуру 23.44 градуса Цельсия. В 16:31 окна были закрыты, температура в кабинете, по показаниям датчика, составляла 21.26 градусов Цельсия. График температуры указан на рисунке 81.

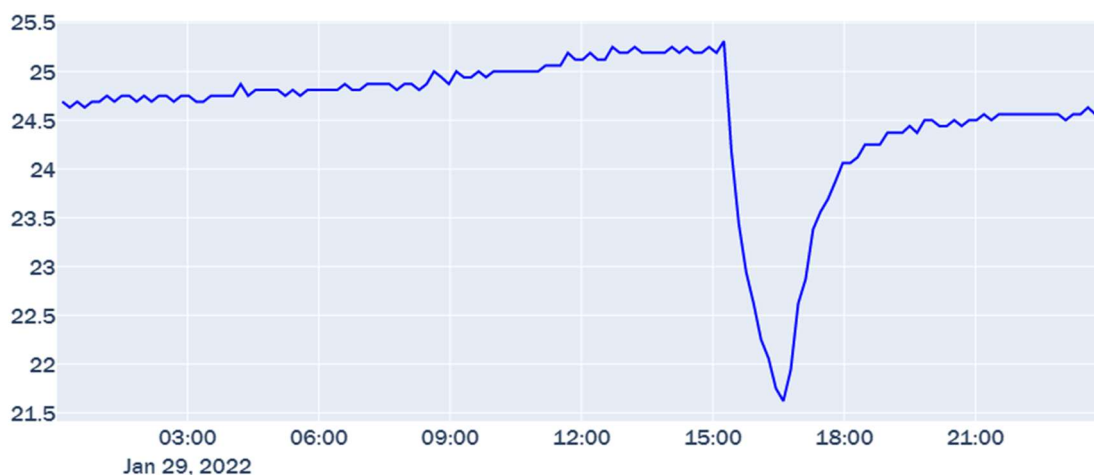


Рисунок 81. Показания датчиков при проветривании 210 аудитории

## 412 аудитория

В 412 кабинете окна открывались дважды с 14:21 до 14:51 и с 15:50 до 16:27. Это было обусловлено тем, что во время первого проветривания, температура воздуха в помещении не уменьшалась, а наоборот увеличивалась. Так что во второй раз было исключено нахождение людей в кабинете. Тем не менее температура в кабинете также снизилась, из чего был сделан вывод о том, что температурный режим данного кабинета слабо реагирует на кратковременное открытие окон. На рисунке 82 можно увидеть показания датчиков в 412 кабинете.

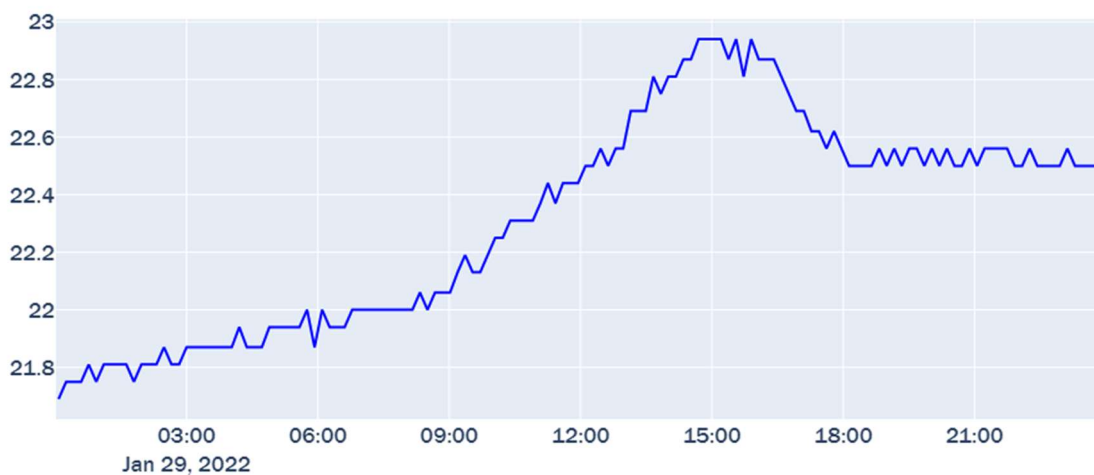


Рисунок 82. Показания датчиков при проветривании 412 аудитории

### Вывод

После проведения эксперимента и анализа полученных данных был сделан вывод о том, что разные кабинеты по-разному реагируют на открытие окон. Некоторые кабинеты реагируют так, как и предполагалось, то есть температура падает при низкой температуре за окном, а некоторые кабинеты не реагируют на проветривание, либо реагируют незначительно.

### 2.7.2.2 Поиск через аппроксимирующую функцию

В качестве одного из способов поиска проветриваний был попробован способ с построением аппроксимирующей функции по результатам эксперимента, и поиск похожих ситуаций через данную функцию. Был взят отрезок с 15:09 до 16:41 29 января.). За значения оси абсцисс были взяты

количество минут, прошедших от начала проветривания, за значения оси ординат – температура в помещении. Были рассчитаны различные аппроксимации графика, но функцией с наименьшей погрешностью стала квадратичная функция, которая была взята за основу для поиска похожих ситуаций. Рассчитанная аппроксимирующая функция выглядела следующим образом:

$$y = 8.3114 \times x^2 - 407.4983 \times x + 5004.3451 \quad (3)$$

Так как коэффициент С отвечает за смещение графика по оси ординат, то, изменяя данное значение, можно смещать начальную температуру и производить дальнейшее сравнение. Других экспериментов не проводилось, поэтому не было возможности сравнить, как изменяется коэффициент А, отвечающий за скорость возрастания/убывания функции на ветвях при других климатических условиях. Тем не менее, с учётом погрешности, удалось найти показания, засчитанные как проветривания. Для того, чтобы каждое падение температуры не считалось проветриванием, было ограничено нахождение как минимум трёх точек совпадения. Из-за этого часть проветриваний была отсеяна, так как период измерений у графиков составляет 10 минут, следовательно, проветривание должно длиться не менее 20 минут, чтобы быть засчитанным алгоритмом, но такой подход позволил избавиться от ошибочно принятых за проветривания значений. На рисунке 83 нарисован график с результатами поиска проветриваний. Синей линией нарисована температура в кабинете, а красными точками – найденные проветривания.

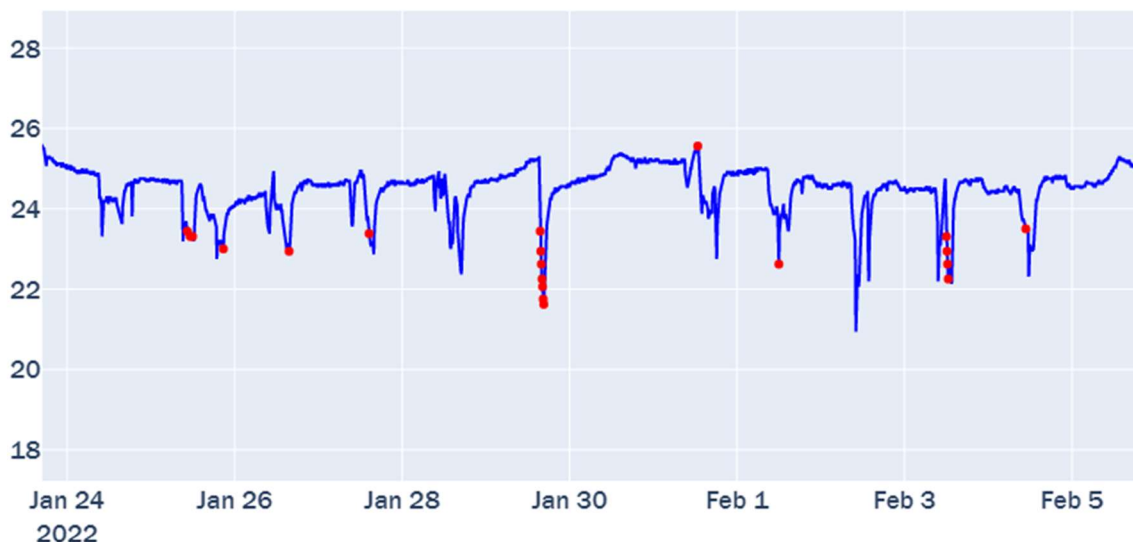


Рисунок 83. Поиск проветриваний с помощью аппроксимирующей функции в 210 аудитории

Данный способ имеет место быть, но при температурах, существенно отличающихся от  $-15^{\circ}\text{C}$ , данный способ неработоспособен. Для его доработки необходимы дополнительные эксперименты для каждого из кабинетов, чтобы найти зависимость между температурой внешнего воздуха и коэффициентом А. Так что данный способ был признан недействительным для всего массива данных

### 2.7.2.3 Поиск по расписанию и температуре.

Вторым способом по поиску проветриваний было использование текущего расписания проветриваний. Разработанный алгоритм поиска ищет совпадения снижения температуры воздуха с расписанием проветриваний с небольшой погрешностью, для уменьшения влияния человеческого фактора. Далее алгоритм проверяет действительно ли температура воздуха была ниже чем температура в кабинете, чтобы быть причиной снижения температуры. Проводится проверка, что в данный момент не было отключений батарей, для того чтобы ошибочно не засчитать падение температуры воздуха от отключения отопления за проветривания.

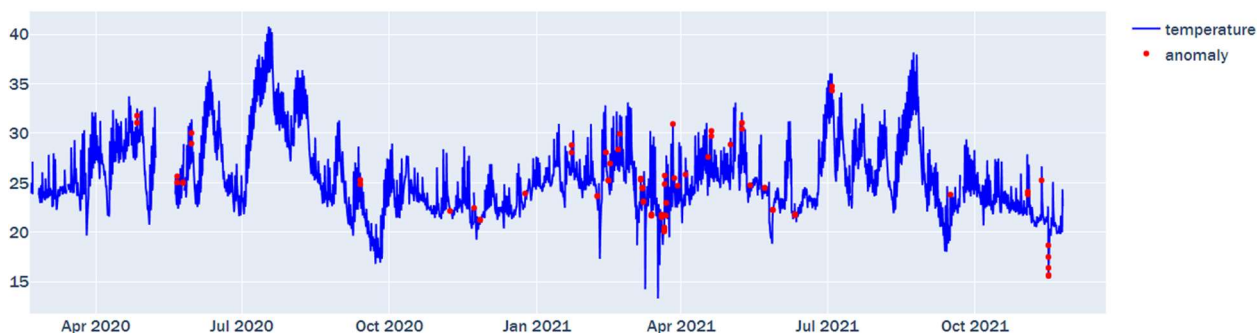


Рисунок 84. Поиск проветриваний в 412 аудитории.

Данный способ более универсален, чем через аппроксимирующую функцию, но также имеет ряд недостатков. Вследствие малого количества размеченных данных с проветриваниями довольно трудно определить точность данных способов. Данные способы нуждаются в доработке, но для этого потребуется больше экспериментов, либо необходимо попробовать принципиально другой подход к поиску проветриваний.

## 2.8. СИСТЕМА МОНИТОРИНГА ТЕМПЕРАТУР

### 2.8.1. РАЗРАБОТКА КЛИЕНТСКОЙ ЧАСТИ СИСТЕМЫ

Было решено, что система мониторинга будет представлять собой клиент-серверное web приложение. Был продуман интерфейс системы мониторинга, система должна содержать следующие страницы:

1. страница просмотра текущих температур кабинетов;
2. температурная карта помещений;
3. история температур по кабинетам.

Страница просмотра текущих температур кабинетов содержит список кабинетов, сгруппированный по этажам здания. Для каждого кабинета выводится его изображение, график текущих температур (за 1–2 часа) и текущие показания температурных графиков. В случае, если выявлена аномалия по температуре выводится предупреждение.

Для страницы мониторинга текущих температур (Рисунок 85) были разработаны следующие компоненты (Таблица 1):

- RoomInfo – компонент отображения информации о помещении;
- TempChart – график температур в помещении и на батареях;

- ThermometerSvg – термометр.

RoomInfo – основной компонент отрисовки информации о текущих температурах в помещении. В качестве параметров данный компонент получает наименование кабинета и данные о температурах кабинета (в помещении и на батареях) за последние 2 часа.

Последние температурные данные передаются в компонент TempChart для вывода графика изменений температур. Этот элемент содержит методы для преобразования исходных показаний в словарь данных для отрисовки графика, а также сам график.

ThermometerSvg компонент – иконка термометра в формате svg, который получает в качестве параметра информацию о том, является ли температура кабинета нормальной, или нет, а также текущее показание температур. При аномальной температуре данный компонент окрашивается в красный цвет.

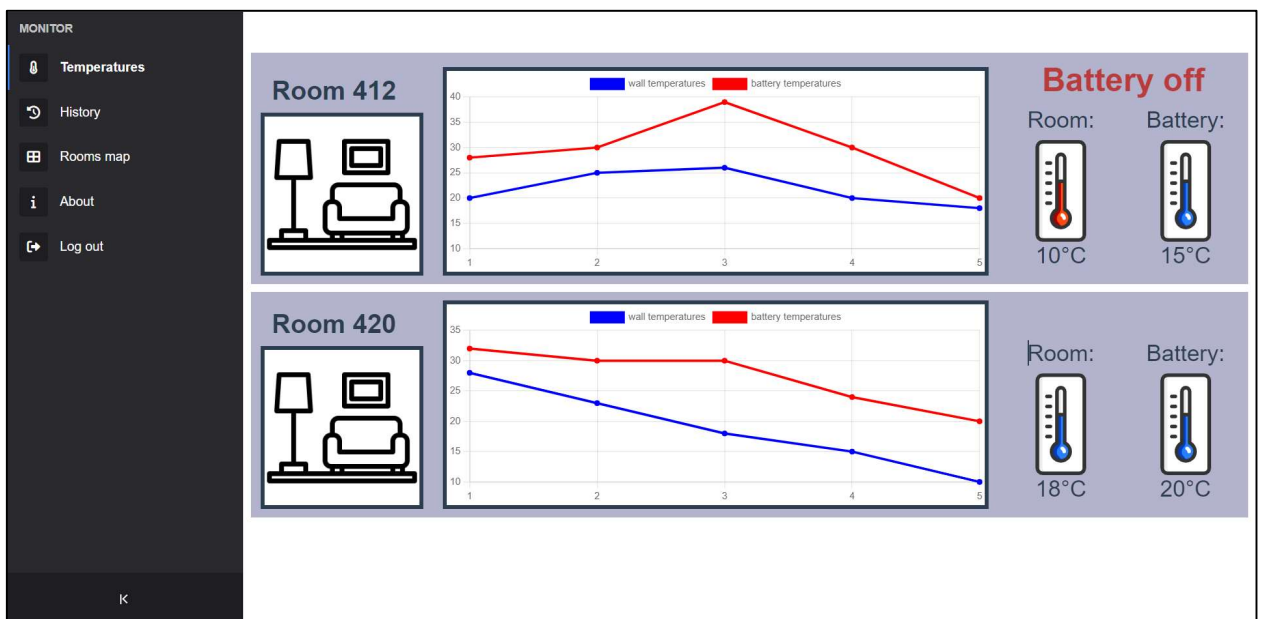


Рисунок 85. Страница текущих температур

Температурная карта представляет собой план здания, на котором для имеющихся кабинетов выводятся показания текущих температур. Также при наличии аномальной температуры, появляется предупреждение.

Для страницы температурной карты помещений (Рисунок 86) планы этажей были перенесены в svg формат, были добавлены компоненты для динамического обновления изображения, данный компонент получает данные о текущих температурах помещения, причинах аномалии, если аномалия имеется, а также помещение окрашивается в красный цвет при большом отклонении температуры от нормы. Для выбора отображаемого этажа используется Dropdown компонент из библиотеки PrimeVue.

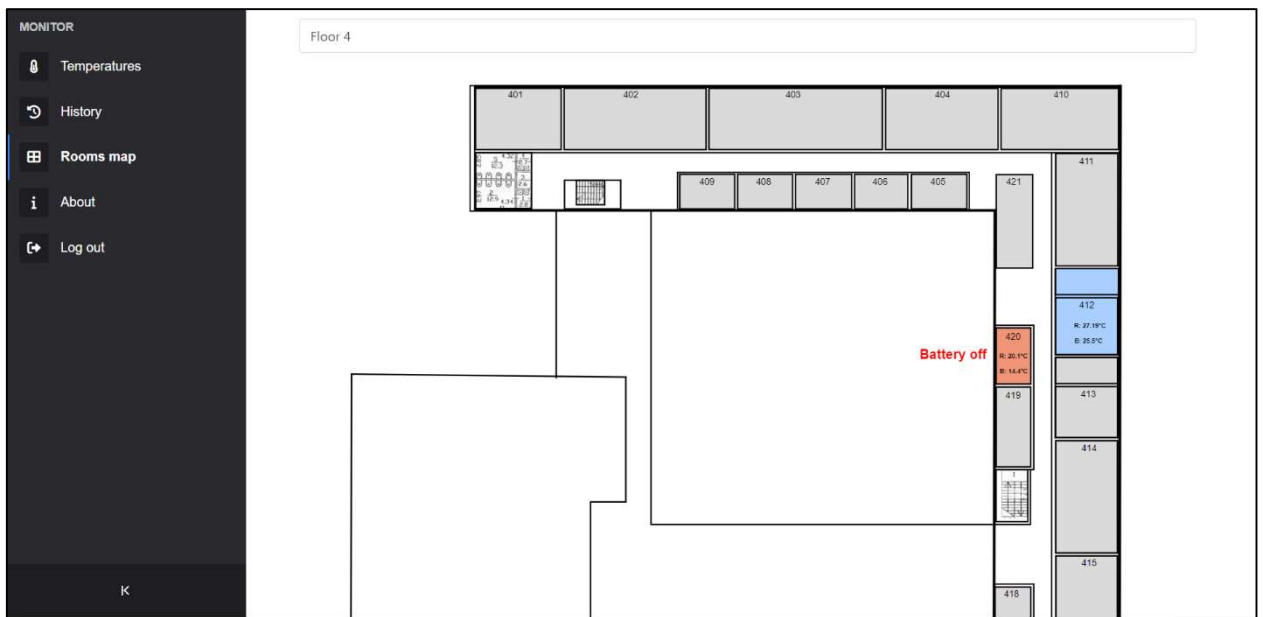


Рисунок 86. Температурная карта помещений

Страница с историей температур по кабинетам содержит выпадающий список с имеющимися кабинетами. При выборе кабинета выводится список с показаниями температур за заданный период, а также список найденных за этот период аномалий (Рисунок 87). Для выбора временного промежутка был использован компонент Calendar, а для вывода таблицы температур – компонент DataTable из библиотеки PrimeVue. Для вывода температур был использован компонент ChartTemp, описанный ранее.

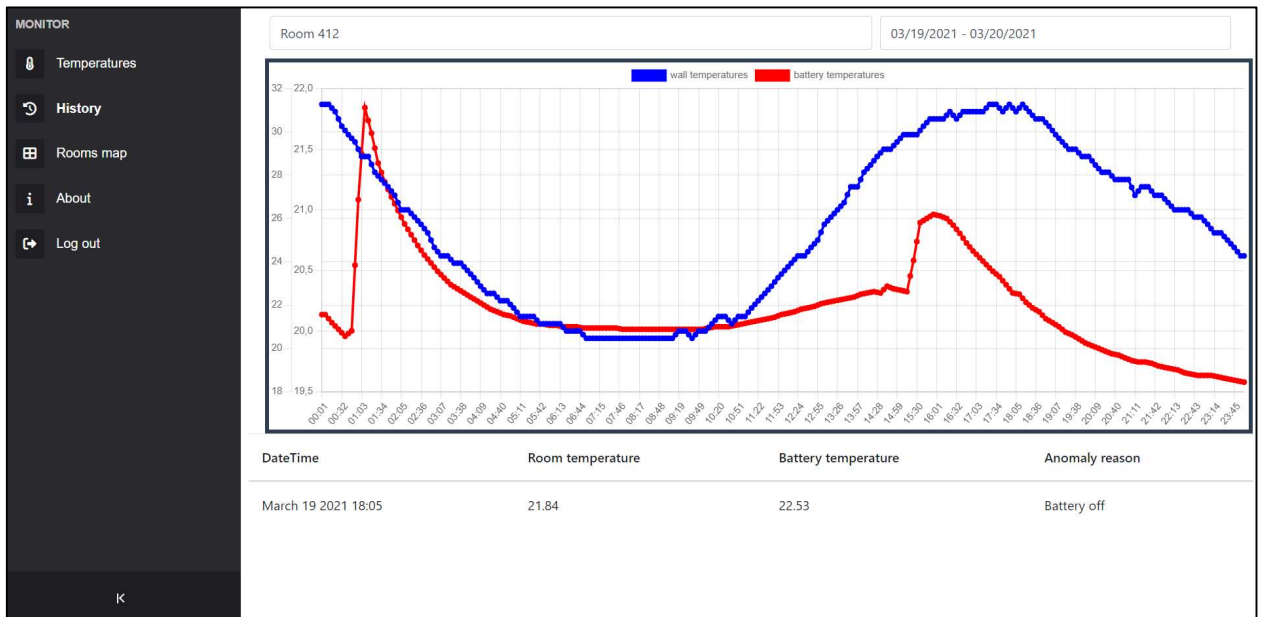


Рисунок 87. История температур по кабинетам.

Для получения информации с сервера был разработан класс RoomTempService, который содержит методы отправляющие запросы на получение последних актуальных данных по всем кабинетам, также температуры и аномалии за определенный период времени по выбранному кабинету (Таблица 1).

Таблица 1

### Основные компоненты интерфейса

Наименование	Входные параметры	Данные	Методы
RoomInfo	roomName (string, required) – наименование помещения. tempsData (object, required) – температурные данные.	isNormal (bool) – является ли температура помещения нормальной. currentWallTemp (number) – текущая температура в помещении. currentBatteryTemp (number) – текущая температура на батареях.	-



TempChart	width, height (number) – ширина и высота графика. tempsData (object, required) – температурные данные.	chartData (object) – словарь данных для отрисовки графика.	getTemps – метод формирования словаря данных со списком точек измерений, массивами данных. convertToChartData – метод преобразования данных по температуре для отрисовки.
ThermometerSvg	normalColor (string) – цвет при нормальной температуре. warnColor (string) – цвет при аномальной температуре.	isNormal (bool) – является ли температура нормальной. temp (number) – показание температуры.	-
Floor4	room_412_wall_temp (double) – температура в помещении room_412_bat_temp (double) – температура на батареях 412_reason (string) – причина аномалии	-	-
RoomsMapView	-	tempsData – полученные температурные данные selectedFloor – выбранный этаж	tempsData (handler) – обновление информации по температурам getData() – получение с сервера данных о температурах getLastData() – получение текущих данных

## 2.8.2. РАЗРАБОТКА СЕРВЕРНОЙ ЧАСТИ СИСТЕМЫ

Серверная часть системы разрабатывалась таким образом, чтобы соответствовать архитектурному стилю взаимодействия компонентов распределённого приложения REST. Разработка велась на языке программирования Python. Он использовался ранее для написания исследовательской части ВКР, поэтому для того, чтобы не переписывать функционал на другой язык программирования или не создавать дополнительных сложностей с вызовом программы с другого языка программирования, было решено продолжить использовать Python. Для обработки web-запросов на сервер использовался фреймворк Flask. Flask – фреймворк для создания веб-приложений на языке программирования Python. Для быстрой отправки данных клиенту необходимо хранить данные в базе данных на компьютере с сервером. В качестве СУБД на начальных этапах разработки была выбрана СУБД SQLite. Данная СУБД является наиболее простой в установке и переносе баз данных с места на место, что позволяет легко переносить базу с компьютера на компьютер и тестировать на разных рабочих местах, что очень важно в условиях совместной разработки.

Серверная часть системы состоит из четырёх основных модулей: модуль, обрабатывающий web-запросы, модуль, отвечающий за работу с базой данных, модуль поиска аномалий и модуль загрузки новых данных с источников.

Модуль обработки web-запросов отвечает за получение и обработку запросов от клиента, а также определяет дальнейшее поведение системы. Если для ответа на запрос модулю необходима дополнительная информация, он обращается к другому модулю, отвечающему за работу с базой данных. На данный момент реализована обработка запроса на получение последних показаний датчиков в JSON-формате и для вывода графиков на главной web-странице пользователя.

Модуль работы с базой данных отвечает за получение и внесение данных из БД. Данный модуль устанавливает соединение с базой данных, форматирует и передаёт необходимые при запросе данные модулю обработки web-запросов, предаёт модулю поиска аномалий хранящиеся модели в json-формате, а также вносит необходимые изменения в базу данных при получении их из модуля загрузки новых данных.

Модуль поиска аномалий, запрашивает у базы данных обученные готовые модели и необходимые для поиска аномалий данные, производит поиск аномалий и возвращает данные по найденным аномалиям обратно в базу данных.

Модуль загрузки новых данных отвечает за проверку наличия новой информации в источниках. Данный модуль периодически опрашивает известные источники информации и, в случае получения новой информации, передаёт эти данные в модуль работы с базой данных.

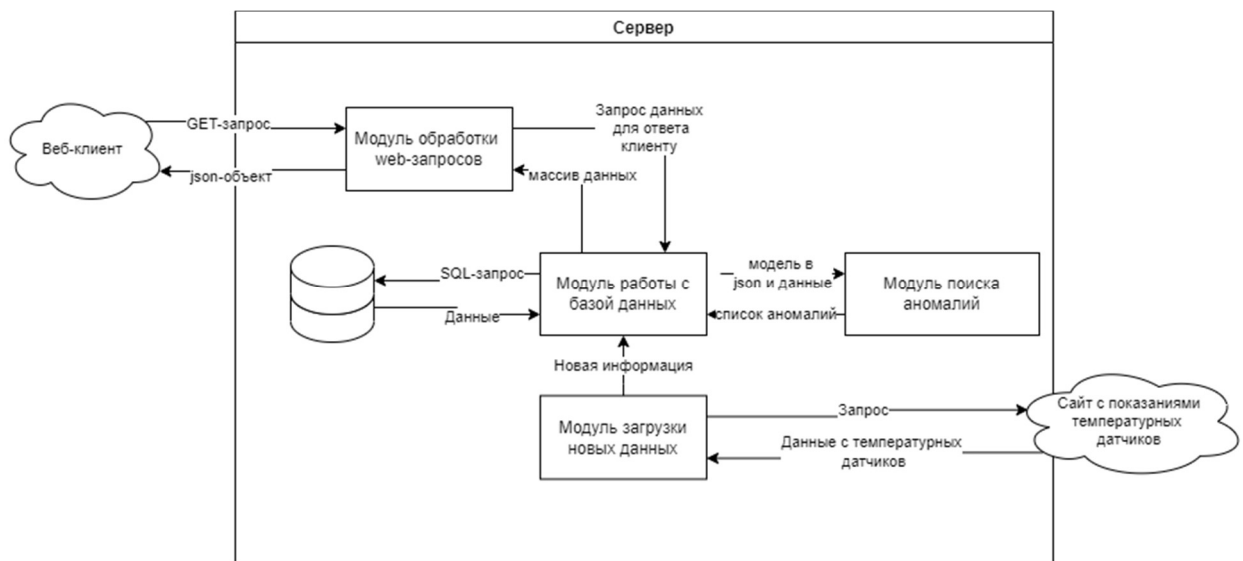
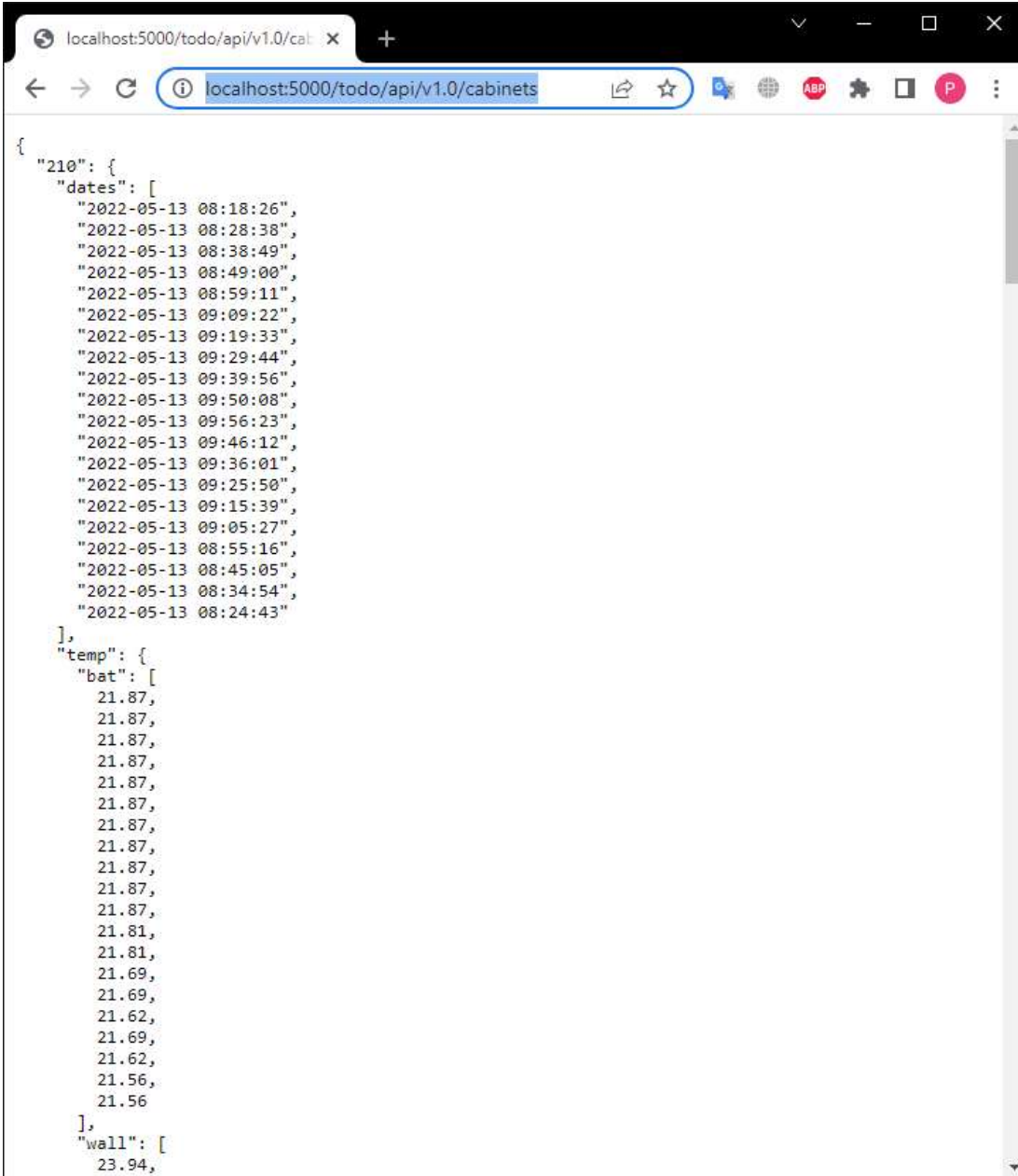


Рисунок 88. Страница текущих температур

Взаимодействие с web-клиентом происходит в соответствии с архитектурным стилем взаимодействия компонентов распределённого приложения REST. Получив GET-запрос, модуль обработки запросов его обрабатывает, запрашивает данные у модуля работы с базой данных, который, в свою очередь, получает данные из базы, преобразовывает их в JSON-формат и возвращает обратно в модуль обработки запросов. Так как сервер построен

в соответствии с REST, то ему нет необходимости запоминать информацию о клиентах, запрашивающих данные - взаимодействие происходит исключительно в ключе “вопрос-ответ”. Пример получаемой информации с сервера на рисунке 89.



```
{
  "210": {
    "dates": [
      "2022-05-13 08:18:26",
      "2022-05-13 08:28:38",
      "2022-05-13 08:38:49",
      "2022-05-13 08:49:00",
      "2022-05-13 08:59:11",
      "2022-05-13 09:09:22",
      "2022-05-13 09:19:33",
      "2022-05-13 09:29:44",
      "2022-05-13 09:39:56",
      "2022-05-13 09:50:08",
      "2022-05-13 09:56:23",
      "2022-05-13 09:46:12",
      "2022-05-13 09:36:01",
      "2022-05-13 09:25:50",
      "2022-05-13 09:15:39",
      "2022-05-13 09:05:27",
      "2022-05-13 08:55:16",
      "2022-05-13 08:45:05",
      "2022-05-13 08:34:54",
      "2022-05-13 08:24:43"
    ],
    "temp": {
      "bat": [
        21.87,
        21.87,
        21.87,
        21.87,
        21.87,
        21.87,
        21.87,
        21.87,
        21.87,
        21.87,
        21.87,
        21.87,
        21.87,
        21.87,
        21.81,
        21.81,
        21.69,
        21.69,
        21.62,
        21.69,
        21.62,
        21.56,
        21.56
      ]
    },
    "wall": [
      23.94,
    ]
  }
}
```

Рисунок 89. Пример возвращаемых данных

Модуль загрузки новых данных отвечает за получение новой информации из источников данных. Данный модуль выгружает с сайта показания датчиков с имеющейся там информацией о температуре в помещении.

## ВЫВОДЫ К ГЛАВЕ 2

В данной главе были решены следующие задачи:

1. Определение источников данных.
2. Сбор и предобработка данных.
3. Предварительный анализ данных.
4. Построение моделей кластеризации и анализ результатов.
5. Реализация алгоритмов поиска аномалий, анализ результатов.
6. Разработка алгоритма определения причин аномалий.
7. Разработка системы мониторинга.

В результате сбора данных были выбраны источники данных и получены необходимые сведения: архив погоды, показания с температурных датчиков и информация о положении солнца. Разработаны алгоритмы постраничных запросов, парсинга полученных страниц и предобработки данных. Результаты объединены в одну таблицу и выгружены в файл.

В процессе предобработки для каждой сущности был построен график распределения. На основании анализа полученных частотных диаграмм были исключены наименее информативные признаки:

- видимость, так как преобладающее большинство значений равно 10км;
- явления, поскольку около 85% значений не заполнены;
- давление на уровне метеостанции, по причине идентичности распределения давлений на уровне метеостанций и на уровне моря;
- количество нижней облачности, поскольку этот признак идентичен общему количеству облачности;
- высота нижней границы в метрах, потому что около 99% пустые или NaN;
- тип облаков, потому как типов всего три возможных значения, а преобладающее большинство данных – пустые.

Помимо этого, был проведён анализ таких факторов как инсоляция и ветер. Были сконструированы новые признаки: показатель влияния ветра и

показатель влияния солнца. Для данных признаков были построены графики изменений этих показателей и температур (внутри помещений, на батареях и внешней среды). Был сделан вывод, что имеется корреляция между показателем инсоляции и тепловым режимом помещения. Корреляция между температурой и показателем ветра довольно слабая, поэтому нельзя сказать, что ветер оказывает значительное влияние на температуру внутри помещения.

Была проведена кластеризация 210, 219, 310, 412 и 420 кабинетов здания ИМИКН. Были определены следующие кластеры: за неотапительный сезон – 1 кластер: 219 и 420, 2 кластер: 316 и 412, 3 кластер: 210; за отопительный сезон – 1 кластер: 412, 316, 210, 2 кластер: 219, 3 кластер: 420.

Были реализованы *aggregation* и *prediction-based* подходы поиска аномалий. В случае использования *aggregation* подхода, были найдены аномалии в вершинах температурных графиков. Данный подход не учитывает специфичность данных о температуре. Для *prediction-based* подхода была спроектирована и обучена LSTM рекуррентная нейросеть. Данная модель показала довольно высокую точность (RMSE на обучающей  $\sim 0.04$ , на тестовой  $\sim 0.2$ ).

Для системы мониторинга было решено использовать *prediction-based* подход. Было выполнено построение прогнозных моделей на данных каждого кабинета, на данных кластера, а также на данных всех исследуемых кабинетов. По результатам сравнения построенных моделей был сделан вывод, что для уменьшения количества моделей можно использовать кластеры, так как данные модели оказались достаточно точными, а модель, обученная на всех кабинетах, обнаружила закономерности, которые являются верными не для всех помещений.

Была разработана система мониторинга температур в помещениях. Серверная часть, была написана в соответствии с архитектурным стилем взаимодействия компонентов распределённого приложения REST. Для обработки веб-запросов использовался микрофреймворк Flask. Клиентская часть разработана с помощью frontend framework Vue.js. Сервис предоставляет

возможность просмотра текущих температур в помещениях, а также отображать наличие аномальных данных с их вероятными причинами.

## ЗАКЛЮЧЕНИЕ

В процессе выполнения выпускной квалификационной работы по теме «Система компьютерного моделирования динамики изменения температуры воздуха в помещениях учебно-лабораторного корпуса под влиянием теплового режима» были выполнены все поставленные задачи. Был проведён теоретический анализ климатических факторов и сделаны следующие выводы:

- на климат внутри помещения оказывают влияние внешняя и внутренняя среды, технологические процессы, а также системы обеспечения микроклимата;
- такие внешние факторы, как инсоляция и воздушные потоки, могут оказывать достаточно существенное воздействие на микроклимат помещения;
- влажность же оказывает заметное влияние только в долгосрочной перспективе, поэтому данным фактором можно пренебречь.

Также был проведён сравнительный анализ технологий и алгоритмов, необходимых для выполнения исследовательской части и программной реализации проекта. В результате сравнения языков программирования был выбран язык программирования Python, потому что в нём имеются удобные инструменты для сбора и обработки данных, визуализации полученной информации, работы с моделями машинного обучения, а также создания и обучения нейронных сетей. В ходе сравнения алгоритмов для кластеризации имеющихся в данных временных рядов была выбрана иерархическая кластеризация, подразумевающая преобразование рядов в данные фиксированного размера, их дальнейшая кластеризация. В качестве алгоритмов для поиска аномальных значений был выбран подход на основе прогнозирования, которые подразумевают использование алгоритмов МО или нейросетей.



Помимо этого, были выбраны источники данных, написаны алгоритмы для загрузки информации из выбранных источников и сформирован датасет, для проведения исследовательской части работы. Проведён анализ полученных данных на наличие корреляций и проведена кластеризация кабинетов. Реализованы aggregation и prediction-based подходы поиска аномалий. Разработана система мониторинга температур в помещениях, представляющая собой клиент-серверное web-приложение.

## СПИСОК ЛИТЕРАТУРЫ

1. ГОСТ 30494-96 Здания жилые и общественные. Параметры микроклимата в помещениях. Дата введения 1999-03-01. – URL: <http://docs.cntd.ru/document/1200003003> (дата обращения: 07.11.2020) – Текст: электронный.
2. Толстова, Ю. И. Основы строительной теплофизики: учебное пособие / Ю. И. Толстова, Р. Н. Шумилов. – Екатеринбург: Издательство Уральского университета, 2014. – 106 с. – URL: [https://elar.urfu.ru/bitstream/10995/29012/1/978-5-7996-1131-6\\_2014.pdf](https://elar.urfu.ru/bitstream/10995/29012/1/978-5-7996-1131-6_2014.pdf) (дата обращения: 14.12.2020).
3. Кувшинов, Ю. А. Общее представление о формировании микроклимата / Ю. А. Кувшинов. Текст: электронный // Холодильщик.RU: Интернет-газета – URL: [http://www.holodilshchik.ru/index\\_holodilshchik\\_issue\\_11\\_2007\\_Microkli mat.htm](http://www.holodilshchik.ru/index_holodilshchik_issue_11_2007_Microkli mat.htm) (Дата обращения: 14.12.2020).
4. Каменев, П. Н. Отопление и вентиляция: учебник для вузов/ П. Н. Каменев, А. Н. Сканави, В. Н. Богословский [и др.]. 3-е изд. – Москва: Стройиздат, 1975. – 483 с. URL: <http://www.bibliotekar.ru/spravochnik-138-otoplenie/index.htm> (Дата обращения: 14.12.2020).
5. Сенченко, Н. М. Сырость в жилых зданиях, ее источники и борьба с ней: книга / Н. М. Сенченко – Москва: Стройиздат, 1967. – 257 с. URL: <http://www.bibliotekar.ru/5-syrost-v-dome/9.htm> (Дата обращения: 14.12.2020).
6. Дрозд, Д. В. Влияние ветра на микроклимат в помещении: материалы V Международной студенческой научной конференции «Студенческий научный форум» / Д. В. Дрозд, Ю. В. Елистратова, А. С. Семиненко. – 2013. – URL: <https://scienceforum.ru/2013/article/2013008381> (Дата обращения: 18.12.2020).
7. Anis, W. A. Влияние воздухопроницаемости на проектирование систем климатизации. / W. A. Anis Текст: электронный // «АВОК» – 2003. – №2

- URL: [https://www.abok.ru/for\\_spec/articles.php?nid=1987](https://www.abok.ru/for_spec/articles.php?nid=1987) (Дата обращения: 18.12.2020).
8. Ананьев В. А. Системы вентиляции и кондиционирования. Теория и практика. / В. А. Ананьев, Л. Н. Балугева, А. Д. Гальперин [и др.]. – Москва: ЕВРОКЛИМАТ, 2000. – 416 с. (Дата обращения: 18.12.2020).
9. Стрижак, П. А. Математическое моделирование теплового режима здания с учетом инсоляционных теплопоступлений. / П. А. Стрижак, М. Н. Морозов. Текст: электронный // Известия Томского политехнического университета. Инжиниринг георесурсов: журнал – URL: [http://earchive.tpu.ru/bitstream/11683/5533/1/bulletin\\_tpu-2015-326-8-05.pdf](http://earchive.tpu.ru/bitstream/11683/5533/1/bulletin_tpu-2015-326-8-05.pdf) (Дата обращения: 20.12.2020).
10. Costa, C. D. Top Programming Languages for Data Science in 2020. / C. D. Costa. Текст: электронный // towards data science: интернет-портал – URL: <https://towardsdatascience.com/top-programming-languages-for-data-science-in-2020-3425d756e2a7> (Дата обращения. 20.02.2022).
11. Khatri, V. S. Python Packages for Data Science. / V. S. Khatri. Текст: электронный // DZone: интернет-портал – URL: <https://dzone.com/articles/python-packages-for-data-science> (Дата обращения. 20.02.2022).
12. Karakan, V. Python vs R for Data Science. / V. Karakan. Текст: электронный // towards data science: интернет-портал – URL: <https://towardsdatascience.com/python-vs-r-for-data-science-6a83e4541000> (Дата обращения. 20.02.2022).
13. Shrestha, C. Can you use JavaScript for Data Science? / C. Shrestha. Текст: электронный // Medium: интернет-портал – URL: <https://medium.com/javascript-in-plain-english/how-about-data-science-and-javascript-lets-take-a-look-c123c6981afa> (Дата обращения. 20.02.2022).
14. Patel Jeel, List of 10 Best Front end Frameworks to Use For Web Development / Jeel Patel. Текст: электронный // Monocubed: интернет-

- портал – URL: <https://www.monocubed.com/blog/best-front-end-frameworks/> (Дата обращения. 23.03.2022).
15. Grinberg M. Flask Web Development. Developing web applications with Python: учебное пособие / M. Grinberg. – O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, 258 с. (Дата обращения. 11.04.2022).
16. Usman Malik – Иерархическая кластеризация с помощью Python и Scikit-Learn / U. Malik. Текст: электронный // Python по байтам интернет-портал – URL: <https://pythobyte.com/hierarchical-clustering-with-python-and-scikit-learn-768953fe/>
17. Saeed Aghabozorgi, Ali Seyed Shirkhorshidi, Teh Ying Wah — Time-series clustering – a Decade Review / S. Aghabozorgi, A. S. Shirkhorshidi, T. Y. Wah. Текст: электронный // Information Systems: интернет-портал – URL: <https://www.sciencedirect.com/science/article/abs/pii/S0306437915000733> (Дата обращения. 13.01.2022).
18. Deepthi Cheboli – Anomaly Detection of Time Series: учебное пособие / D. Cheboli. – University of Minnesota M.S. May 2010, 75с (Дата обращения. 10.02.2022)

## ПРИЛОЖЕНИЯ 1-5

## ПРИЛОЖЕНИЕ 1

## ЗАПРОС ДАННЫХ О ПОГОДЕ

```
def get_weather_info_date(date):
    # get html from the weather source
    url = 'http://meteocenter.net/forecast/all.php'
    html = requests.post(url, data=date_to_object(date)).text
    # parse html and get weather info
    soup = BeautifulSoup(html, 'html.parser')
    weather_info = soup.find('pre').text
    splitted_data = weather_info.split('\r\n')
    # get columns name
    header = splitted_data[1].split(';')
    # get data
    weather_info = splitted_data[2:]
    # to dataframe
    weather =
        pd.DataFrame([x.split(';') for x in weather_info], columns=header)
    weather = format_weather_info(weather)
    weather['date_time'] =
        weather['date_time'].apply(lambda x: x + timedelta(hours=5))
    # add sun position
    weather = df_set_sun_position(weather)
    weather = weather.set_index('date_time')
    return weather
```

## ПРЕДОБРАБОТКА ДАННЫХ О ПОГОДЕ

```
# convert all columns of weather info to appropriate type
def weather_columns_to_digit(df):
    # wind to int
    df['wind_dir'] = df['wind_dir'].apply(lambda x: direction_to_int(x))
    df['wind_speed'] = df['wind_speed'].apply(lambda x: speed_to_int(x))

    # visibility to int
    df['visibility'] = df['visibility'].apply(lambda x: km_to_m(x))

    # cloudiness to int
    cloudiness = df['cloudiness'].apply(lambda x: cloudiness_split(x))
    # cloudiness_overall_amount
    df["COA"] = cloudiness.apply(lambda x: x[0])
    # cloudiness_under_lower_amount
    df["CULA"] = cloudiness.apply(lambda x: x[1])
    # cloudiness_lower_bound
    df["CLB"] = cloudiness.apply(lambda x: x[2])
    # cloud_type
    df["CT"] = cloudiness.apply(lambda x: x[3])
    df = df.drop('cloudiness', axis=1)

    # all digit columns to int
    columns_to_int(df, ['T', 'Td', 'f', 'Te', 'QNH', 'Po'])

    return df
```

## ПОЛУЧЕНИЕ ДАННЫХ С ДАТЧИКОВ

```
# get all sensor data
def get_sensor_info(sensor_info, first_date_time=None, sep=" "):
    sensor_info_tmp = dict()
    # data_count needs to check that all data was added
    data_count = 0
    # get DataFrames from the URL
    for sensor in sensor_info.keys():
        sensor_info_tmp[sensor] =
            get_sensor_info_from_url(sensor_info[sensor], sep)
        data_count += sensor_info_tmp[sensor].shape[0]
        # rename temp column for every sensor
        sensor_info_tmp[sensor] =
            sensor_info_tmp[sensor].rename(columns={"temp": sensor + "_temp"})
    # create summary df
    sensor_info_val = sensor_info_tmp.values()
    # convert values of time column to time
    df = pd.concat(sensor_info_val, sort=True)
    dc.df_add_datetime(df)
    # delete loaded data
    if first_date_time is not None:
        df = df[df['date_time'] >= first_date_time]
    # set index and resample
    df = df.set_index('date_time')
    df = df.resample("1H").mean()
    return df.round(2)
```

## КЛАСС СОЗДАНИЯ МОДЕЛИ (LSTM)

```
class PreprocessingSettings(object):
    cleaning_type = None
    columns = []
    preprocessing_type = None
    start_date = None
    normalize = True

    def __init__(self):
        self.cleaning_type = None
        self.columns = []
        self.preprocessing_type = None
        self.start_date = None
        self.normalize = True

    def __init__(self, preprocessing_type, cleaning_type = None, columns = [],
start_date=None, normalize=True):
        self.cleaning_type = cleaning_type
        self.columns = columns
        self.preprocessing_type = preprocessing_type
        self.start_date = start_date
        self.normalize = True

    def preprocess_data(self, dataset, rooms_columns = []):
        return generate_dataset_for_model(dataset, self.cleaning_type,
rooms_columns + self.columns, self.preprocessing_type,
start_date=self.start_date, normalize=self.normalize)

class ModelData(object):
    input_dataset = None
    scaler = None
    x_train = None
    y_train = None
    x_test = None
    y_test = None
```



```

train_index = None
test_index = None
test_predicted = None
target_column_wall = None
target_column_bat = None

def __init__(self, dataset):
    self.input_dataset = dataset

#fixed room_number 07.06.22
class ModelGeneration(object):
    data_info = []
    result_model = None

    def __init__(self):
        self.result_model = None
        self.data_info = []

    def set_preprocessing(settings):
        ModelGeneration.preprocessing_settings = settings

    def fit_model(self, dataset, target_room, test_size = 1500, drop_out=0.2,
activation='linear', optimizer='rmsprop', epochs_num = 20, all_values_df =
None, columns_to_plot = []):
        target_column_wall = target_room + "_wall_temp"
        target_column_bat = target_room + "_bat_temp"

        print("preprocessing data.....")
        preprocessed, scaler =
ModelGeneration.preprocessing_settings.preprocess_data(dataset,
[target_column_wall])
        print(preprocessed)

        info = ModelData(preprocessed)
        info.scaler = scaler

        info.target_column_wall = target_column_wall

```

```

info.target_column_bat = target_column_bat

print("Splitting dataset.....")
info.x_train, info.y_train, info.x_test, info.y_test, info.train_index,
info.test_index = get_train_test(info.input_dataset, info.target_column_wall,
test_data_size=test_size)

if self.result_model is None:
    print("\nCreating model.....")
    self.result_model = create_model(info.x_train, drop_out, activation,
optimizer=optimizer)

print("\nModel training.....")
fit_model(self.result_model, info.x_train, info.y_train,
epochs=epochs_num)
print("\nModel has been created")
plot_model(self.result_model)
print("\nPredicting values.....")
info.test_predicted = model_predict(self.result_model, info.x_test)
print("\nPrediction finished:")
print_error(info.y_test, info.test_predicted.flatten())
plot_temps_series_predicted(info.test_index, info.y_test,
info.test_predicted.flatten())
if len(columns_to_plot) > 0:
    plot_dataset(all_values_df[info.test_index[0]:info.test_index[-1]],
[info.target_column_wall, info.target_column_bat], info.target_column_wall)

self.data_info.append(info)

def predict(self, dataset, target_room, needPreprocess = True, show_bat =
True, show_T = True):
    target_column_wall = target_room + "_wall_temp"
    target_column_bat = target_room + "_bat_temp"

    if (needPreprocess):

```

```

        preprocessed, scaler =
ModelGeneration.preprocessing_settings.preprocess_data(dataset,
[target_column_wall])
        x_train, y_train, x_test, y_test, train_index, test_index =
get_train_test(preprocessed, target_column_wall,
test_data_size=len(preprocessed))
        res_dataset = preprocessed
    else:
        x_train, y_train, x_test, y_test, train_index, test_index =
get_train_test(dataset, target_column_wall, test_data_size=len(dataset))
        res_dataset = dataset

    predicted = model_predict(self.result_model, x_test)
    print("\nPrediction finished:")
    print_error(y_test, predicted.flatten())

    denorm = denorm_predicted(res_dataset, scaler, predicted.flatten(),
test_index, target_column_wall)
    if show_bat:
        denorm[target_column_bat] = dataset[target_column_bat]
        bat_col = denorm[target_column_bat]
    else:
        bat_col = None

    if show_T:
        denorm["T"] = dataset["T"]
        T_col = denorm["T"]
    else:
        T_col = None

    print(denorm)
    plot_temps_series_predicted(test_index, denorm[target_column_wall],
denorm[target_column_wall + "_pred"], bat_col, T_col)

    return predicted, res_dataset, x_train, y_train, x_test, y_test,
train_index, test_index, scaler, denorm

```

```

def calculate_anomalies(self, dataset, target_room, anomalies_coef,
needPreprocess = True, show_bat_P = False, show_T_P = False, show_bat = True,
show_T = True):
    target_column_wall = target_room + "_wall_temp"
    target_column_bat = target_room + "_bat_temp"
    predicted, res_dataset, x_train, y_train, x_test, y_test, train_index,
test_index, scaler, denorm = self.predict(dataset, target_room,
needPreprocess, show_bat_P, show_T_P)
    print(len(predicted))
    diff = get_differences(predicted, y_test)
    threshold = get_threshold(diff, anomalies_coef)
    df = dataset.copy()
    df["anomaly"] = False
    i = 0
    for index in test_index:
        df.at[index,"anomaly"]= (diff[i] >= threshold)
        i+=1

    print(df['anomaly'].value_counts())
    if show_bat:
        battery_col = target_column_bat
    if show_T:
        T_col = "T"

    plot_anomalies(df, target_column_wall, battery_col, T_col)
    return predicted, res_dataset, df

def save_model(self, name):
    save_model(self.result_model, name)

```

## МЕТОД СОЗДАНИЯ МОДЕЛИ (LSTM)

```
def create_model(x_train, drop_out=0.2, activation='linear',
optimizer='rmsprop'):
    model = Sequential()

    model.add(LSTM(input_dim=x_train.shape[-1], units=50,
return_sequences=True))
    model.add(Dropout(drop_out))

    model.add(LSTM(100, return_sequences=False))
    model.add(Dropout(drop_out))

    model.add(Dense(units=1))
    model.add(Activation(activation))

    start = datetime.now()
    model.compile(loss='mse', optimizer=optimizer)
    print('  Compilation time : {}'.format(datetime.now() - start))

    return model
```