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

Направление	09.03.01 – Информатика и вычислительная техника
Профиль	Организация и программирование вычислительных и информационных систем
Факультет	КТИ
Кафедра	ВТ

К защите допустить
Зав. кафедрой ВТ, д. т. н., проф. _____ М. С. Куприянов

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА БАКАЛАВРА

Тема: АНАЛИЗ ДАННЫХ И ПРОГНОЗИРОВАНИЕ ПОТРЕБЛЕНИЯ ЭЛЕКТРОЭНЕРГИИ В ЗАВИСИМОСТИ ОТ ПОГОДНЫХ УСЛОВИЙ

Студентка _____ Е. А. Сабурова

Руководитель к.т.н., доц. _____ Я. А. Бекенева

Консультанты к.э.н., доц. _____ Г. В. Голигузова

_____ М. Н. Гречухин

Санкт-Петербург
2024

ЗАДАНИЕ

НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ

Утверждаю

Зав. кафедрой ВТ, д.т.н., проф.

_____ М. С. Куприянов

«___»_____20___ г.

Студентка Е.А. Сабурова Группа 0308

Тема работы: Анализ данных и прогнозирование потребления электроэнергии в зависимости от погодных условий

Место выполнения ВКР: кафедра ВТ

Технические требования: Набор данных, в котором содержатся данные о погодных условиях, а также о потребленном электричестве в трех зонах. Существует потребность в анализе этих данных для дальнейшего прогнозирования будущего потребления электроэнергии.

Содержание ВКР: Изучение технической документации по теме работы, анализ данных потребления электричества, подготовка данных к обучению моделей, сравнение алгоритмов машинного обучения и выбор лучшего для конкретного набора данных, программная реализация предсказания будущего потребления электричества, описание выполненной работы.

Перечень отчетных материалов: пояснительная записка, иллюстративный материал, реферат, аннотация, презентация.

Дополнительные разделы: Экономическое обоснование ВКР

Дата выдачи задания	Дата представления ВКР к защите
«22» февраля 2024 г.	«18» июня 2024 г.

Студентка _____ Е. А. Сабурова

Руководитель к.т.н., доц. _____ Я. А. Бекенева

КАЛЕНДАРНЫЙ ПЛАН ВЫПОЛНЕНИЯ

ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ

Утверждаю

Зав. кафедрой ВТ, д.т.н., проф.

_____ М. С. Куприянов

«___»_____20___ г.

Студентка Е. А. Сабурова Группа 0308

Тема работы: Анализ данных и прогнозирование потребления электроэнергии в зависимости от погодных условий

№ этапа	Наименование работ	Срок выполнения
1	Изучение технической документации по теме работы	22.02 – 10.03
2	Анализ данных потребления электричества	11.03 – 19.03
3	Подготовка данных к обучению моделей	20.03 – 24.03
4	Сравнение алгоритмов машинного обучения и выбор лучшего для конкретного набора данных	25.03 – 15.04
5	Программная реализация предсказания будущего потребления электричества	16.04 – 01.05
6	Экономическое обоснование	01.05 – 10.05
7	Оформление пояснительной записки	11.05 – 31.05
8	Оформление демонстрационного материала, подготовка доклада	01.06 – 10.06
9	Представление работы к защите	15.06

Студентка _____ Е. А. Сабурова

Руководитель к. т. н., доц. _____ Я. А. Бекенева

РЕФЕРАТ

Пояснительная записка содержит: 54 с., 24 рис., 3 табл., 1 приложение, 19 источников литературы.

Ключевые слова: Анализ данных, прогнозирование, приложение, Python

Объектом исследования является прогнозная модель потребления электроэнергии.

Целью работы являлись анализ данных, обучение моделей машинного обучения, сравнение полученных результатов и выбор лучшей модели для дальнейшего прогнозирования, создание приложения для удобного использования. Разработанное приложение позволяет получать данные о погодных условиях, а затем прогнозирует потребляемую электроэнергию.

В ходе выполнения дипломного проекта были использованы библиотеки для интеллектуального анализа данных и язык программирования Python. Был выполнен обзор различных моделей машинного обучения. В результате обзора, а также сравнения работы этих моделей на используемом наборе данных было принято решение прогнозировать будущие данные с использованием модели XGBoost.

Дополнительным разделом выпускной квалификационной работы является экономическое обоснование. В этом разделе были произведены расчеты полных затрат на разработку проекта.

ABSTRACT

The graduation work included data analysis, training machine-learning models, comparing the results obtained and choosing the best model for further forecasting, creating an application for easy use. The developed application allows you to obtain data on weather conditions and then predicts the electricity consumption.

During the research, libraries for data mining and the Python programming language were used. A review of various machine-learning models was performed. As a result, of the review, as well as comparison of the performance of these models on the used data set, it was decided to predict future data using the XGBoost model.

СОДЕРЖАНИЕ

ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

В настоящей пояснительной записке применяются следующие определения, обозначения и сокращения:

API (англ. Application Programming Interface) – описание способов, с помощью которых одна программа может взаимодействовать с другой.

CNN (Convolutional Neural Network) – сверточная нейронная сеть

LSTM (Long short-term memory) – долгая краткосрочная память

MAE (Mean Absolute Error) – средняя абсолютная ошибка

MLP (Multilayer perceptron) – многослойный перцептрон

MSE (Mean Square Error) – средняя квадратичная ошибка

XGB (eXtreme Gradient Boosting) – экстримальный градиентный бустинг

ВВЕДЕНИЕ

В наше время невозможно представить жизнь без электричества. Ежедневно люди во всем мире потребляют электроэнергию. Именно поэтому получение точных прогнозов по электропотреблению крайне важно для различных категорий людей, так как они позволяют лучше спланировать вероятные затраты предприятий, семей и других групп лиц. Погодные условия имеют большое влияние на жизнь человека, в том числе и на количество расходуемой электроэнергии. Возникает необходимость в разработке метода для прогнозирования потребления электроэнергии в зависимости от погодных условий, который позволял бы предсказывать электропотребление с высокой точностью.

Объектом исследования является прогнозная модель потребления электроэнергии.

Предметом исследования дипломной работы являются методы и подходы прогнозирования потребления электроэнергии.

Целью работы являлись анализ данных, обучение моделей машинного обучения, сравнение полученных результатов и выбор лучшей модели для дальнейшего прогнозирования, создание приложения для удобного использования. Разработанное приложение позволяет получать данные о погодных условиях, а затем прогнозирует потребляемую электроэнергию.

Для достижения поставленной цели в дипломной работе решаются следующие задачи:

- Анализ данных потребления электричества и подготовка этих данных к обучению моделей.
- Исследование различных алгоритмов машинного обучения, применимых для достижения поставленной цели, по выбранным параметрам и сравнение по определенным метрикам для оценки.
- Программная реализация и обучение выбранного алгоритма по результатам исследования.

- Разработка приложения для предсказания будущего потребления электричества.

1 Анализ и прогнозирование электропотребления

В этой главе проводится обзор существующих решений для осуществления анализа электропотребления в зависимости от погодных условий. Для того, чтобы понимать, что необходимо делать, требуется понимать, что подразумевает под собой анализ данных и их прогнозирование.

Интеллектуальный анализ данных – это процесс выявления закономерностей и поиска аномалий и взаимосвязей в больших наборах данных, которые можно использовать для прогнозирования будущих тенденций. По сути это совокупность методов и приложений, связанных с алгоритмами обработки данных и не имеющих четко зафиксированного ответа на каждый входящий объект. Основная цель интеллектуального анализа данных – извлечение ценной информации из доступных данных.

В большинстве случаев такие закономерности невозможно обнаружить при классическом просмотре данных, так как объем данных очень велик, а связи слишком сложны. Данные закономерности можно собрать вместе и определить как модель интеллектуального анализа данных. Модели интеллектуального анализа данных могут применяться к конкретным сценариям: прогнозированию, поиску последовательностей, определению рисков и вероятностей, принятию решений.

Прогнозирование потребления электроэнергии – сложная задача, которая зависит от множества аспектов, не всегда понятных и очевидных на первый взгляд. Эксперты, когда создают прогнозы, часто обращают внимание на какой день строится прогноз (рабочий, выходной, праздничный), в каких климатических условиях находится объект (температура, осадки) и на время суток.

Предсказание электропотребления на длительные сроки и с достаточной заблаговременностью может быть использовано при планировании денежных затрат.

Термин «машинное обучение» обозначает множество математических, статистических и вычислительных методов для разработки алгоритмов, способных решить задачу не прямым способом, а на основе поиска закономерностей в разнообразных входных данных. Решение вычисляется не по четкой формуле, а по установленной зависимости результатов от конкретного набора признаков и их значений. Поэтому машинное обучение применяется для диагностики, предсказанию, распознавания и принятия решений в различных прикладных сферах.

1.1 Анализ и подготовка данных

Основная задача этапа анализа и подготовки данных состоит в том, чтобы получить обработанный, высококачественный набор данных, который подчиняется некоторой закономерности [1]. Этот этап состоит из 4 стадий, показанных на рисунке 1.

Рисунок 1 – Этапы анализа и подготовки данных

Анализ данных: задача этого шага – понять слабые и сильные стороны в имеющихся данных, определить их достаточность, предложить идеи, как их

использовать. Если собственных данных не хватает, тогда необходимо купить данные у третьих лиц или организовать сбор новых данных. Данные могут быть: собственными, сторонними и «потенциальными» данными (нужно организовать сбор, чтобы их получить). С помощью таблиц и графиков смотрим на данные, чтобы сформулировать гипотезы о том, как данные помогут решить поставленную задачу. Обязательно до моделирования требуется оценить, насколько качественные нужны данные, так как любые ошибки на данном шаге могут негативно повлиять на ход проекта. Типичные проблемы, которые могут быть в данных: пропущенные значения, ошибки в данных, опечатки.

Сбор данных – это процесс сбора информации по интересующим переменным в установленной систематической форме, которая позволяет отвечать на поставленные вопросы исследования, проверять гипотезы и оценивать результаты. Правильный сбор данных имеет важное значение для обеспечения целостности исследований. Как выбор подходящих инструментов сбора данных, так и четко разграниченные инструкции по их правильному использованию снижают вероятность возникновения ошибок. Прогнозирующие модели хороши только для данных, из которых они построены. Данные не должны содержать ошибок и должны быть релевантными.

Следующий шаг в процессе подготовки – очистка и нормализация "сырых" данных. В широком смысле нормализация необходима для приведения к определённому формату и представлению, которые обеспечивают их корректное применение в многомерном анализе, совместных исследованиях, сложных технологиях аналитической обработки. Принятие решения насчет того, что делать с отсутствующими или неполными данными, а также с выбросами, также является частью нормализации данных.

Моделирование данных является следующим этапом подготовки данных, которые будут использоваться для прогнозирования. Моделирование данных — это сложный процесс создания логического представления

структуры данных. Правильно сконструированная модель данных должна соответствовать всем пользовательским представлениям данных. Моделирование также включает в себя смешивание и агрегирование веб данных, данных из мобильных приложений, оффлайн данных и др.

1.2 Моделирование

Модель машинного обучения — это приложение искусственного интеллекта, которое дает возможность автоматически учиться и совершенствоваться на основе собственного опыта без явного участия человека. Модель представляет собой алгоритм или математическую функцию, которая преобразует входные данные в выходные. Для разных задач и типов данных пробуются различные модели, перебираются гиперпараметры, сравниваются значения выбранной метрики и выбирается лучшая комбинация. На рисунке 2 показаны все этапы моделирования.

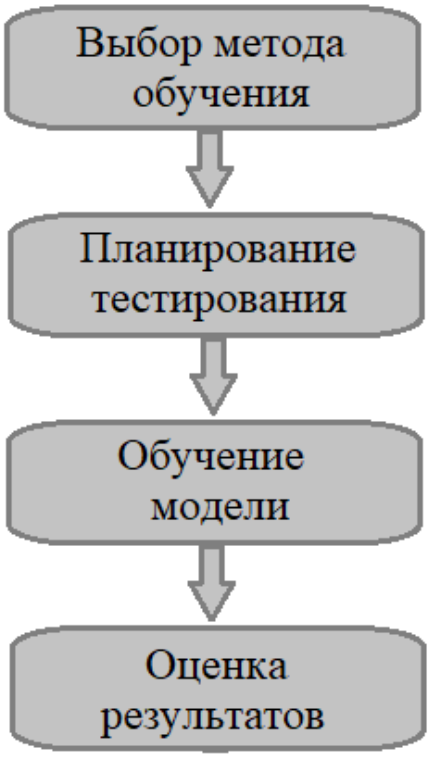

Рисунок 2 – Этапы моделирования

Существует множество методов машинного обучения, но рассмотрим 2 классических вида:

- С учителем, когда необходимо найти функциональную зависимость результатов от входов и построить алгоритм, на входе принимающий описание объекта и на выходе выдающий ответ [2]. Функционал качества, как правило, определяется через среднюю ошибку ответов алгоритма по всем объектам выборки. К обучению с учителем относятся задачи классификации, регрессии, ранжирования и прогнозирования.
- Без учителя, когда ответы не задаются, и нужно искать зависимости между объектами. Сюда входят задачи кластеризации, поиска ассоциативных правил, фильтрации выбросов, построения доверительной области, сокращения размерности и заполнения пропущенных значений.

Далее необходимо определить, на каких данных будет обучаться модель, а на каких тестироваться. Традиционный подход – это разделение набора данных на 3 части (обучение, валидация и тестирование. В данном случае обучающая выборка используется для обучения модели, а валидация и тестирование для получения значения метрики без эффекта переобучения. Также на данном шаге требуется определить, как будет происходить оптимизация гиперпараметров моделей, сколько потребуется итераций для каждого алгоритма, будет ли использоваться grid-search или random-search.

На данном шаге начинается цикл обучения модели. После каждой итерации записывается результат модели. На выходе получаем результаты для каждой модели и использованных в ней гиперпараметров. Кроме того, для моделей, у которых значение выбранной метрики превышает минимально допустимое, нужно обратить внимание на следующие особенности:

- Необычные закономерности. Например, точность предсказания модели на 95% объясняется всего лишь одним признаком.

- Скорость обучения модели. Если модель долго обучается, то стоит использовать более эффективный алгоритм или уменьшить обучающую выборку.
- Проблемы с данными. Например, в тестовую выборку попали объекты с пропущенными значениями, и, как следствие, значение метрики было посчитано не полностью, и она не позволяет целиком оценить модель.

После формирования списка из подходящих моделей, нужно еще раз их детально проанализировать и выбрать лучшие модели. На выходе необходимо иметь список моделей, отсортированный по объективному и/или субъективному критерию. Задачи шага: провести технический анализ качества модели, достигаются ли заданные критерии качества, проанализировать результаты с точки зрения достижения запланированных целей. Если критерий успешности не достигнут, то необходимо или улучшить текущую модель, или использовать другую. Прежде чем переходить к внедрению нужно убедиться, что результат моделирования понятен и логичен. Необходимо также следить за слишком хорошим результатом, если он близок к 100% это повод проверить модель еще раз.

1.3 Прогнозирование в интеллектуальном анализе данных

Прогнозирование – один из основных аспектов интеллектуального анализа данных, который представляет собой одно из четырех направлений аналитики. Прогнозная аналитика использует шаблоны, в текущих или исторических данных, чтобы распространить их на будущее. Таким образом, это дает организациям представление о том, какие тенденции будут происходить в их данных в будущем. Некоторые из наиболее продвинутых подходов аналитики включают аспекты машинного обучения и искусственного интеллекта.

Прогнозные методы могут отличаться в зависимости от срока прогнозирования. Как правило используют следующую классификацию прогнозов:

– краткосрочные прогнозы;

– среднесрочные прогнозы;

– долгосрочные прогнозы.

Термины «кратко-», «средне-» и «долго-» не имеют чёткого определения, но должны зависеть от инерционности объекта исследования.

Так краткосрочным можно назвать прогноз, осуществляемый на срок, не превышающий период инерционности объекта исследования. Главное, что характеризует этот тип прогноза – это то, что исследуемый объект сохраняет свою устойчивость. Он может быть достаточно точно спрогнозирован. Прогнозные методы в случае с краткосрочным прогнозированием должны в большей степени учитывать последние полученные данные.

Среднесрочный прогноз – это прогноз на срок, незначительно превышающий период инерционности объекта исследования. В этом случае в динамике показателя может наметиться какая-нибудь тенденция, которую можно выловить и спрогнозировать с помощью математических методов. Прогнозные методы в этом случае должны брать в расчёт не только последние полученные данные, но и более старые данные, однако предпочтение всё же должно отдаваться первым.

Долгосрочный прогноз – это прогноз на срок, значительно превышающий период инерционности. Здесь уже динамику показателя спрогнозировать становится практически невозможно: за этот срок слишком многое может произойти. Поэтому для прогнозирования на долгий срок нужно обращаться к различным сценариям и использовать экспертные методы для выбора оптимистичного, пессимистичного и наиболее вероятного из них.

Период инерционности очевидным образом меняется от одного объекта исследования к другому, поэтому подобный метод прогнозирования не всегда бывает хорош [3].

1.4 Используемые алгоритмы моделей машинного обучения

В работе было рассмотрено несколько алгоритмов моделей для анализа данных.

1.4.1 Модель Multilayer Perceptron

Multilayer Perceptron (MLP) — это контролируемый алгоритм обучения, который изучает функцию $f(\cdot):R^m \rightarrow R^o$ путем обучения на наборе данных, где m – количество измерений для ввода, и o – количество измерений для вывода. Учитывая набор функций $X=x_1,x_2,...,x_m$ и цель y, он может изучить аппроксиматор нелинейной функции для классификации или регрессии. Он отличается от логистической регрессии тем, что между входным и выходным слоем может быть один или несколько нелинейных слоев, называемых скрытыми слоями. На рисунке 3 показана модель MLP.

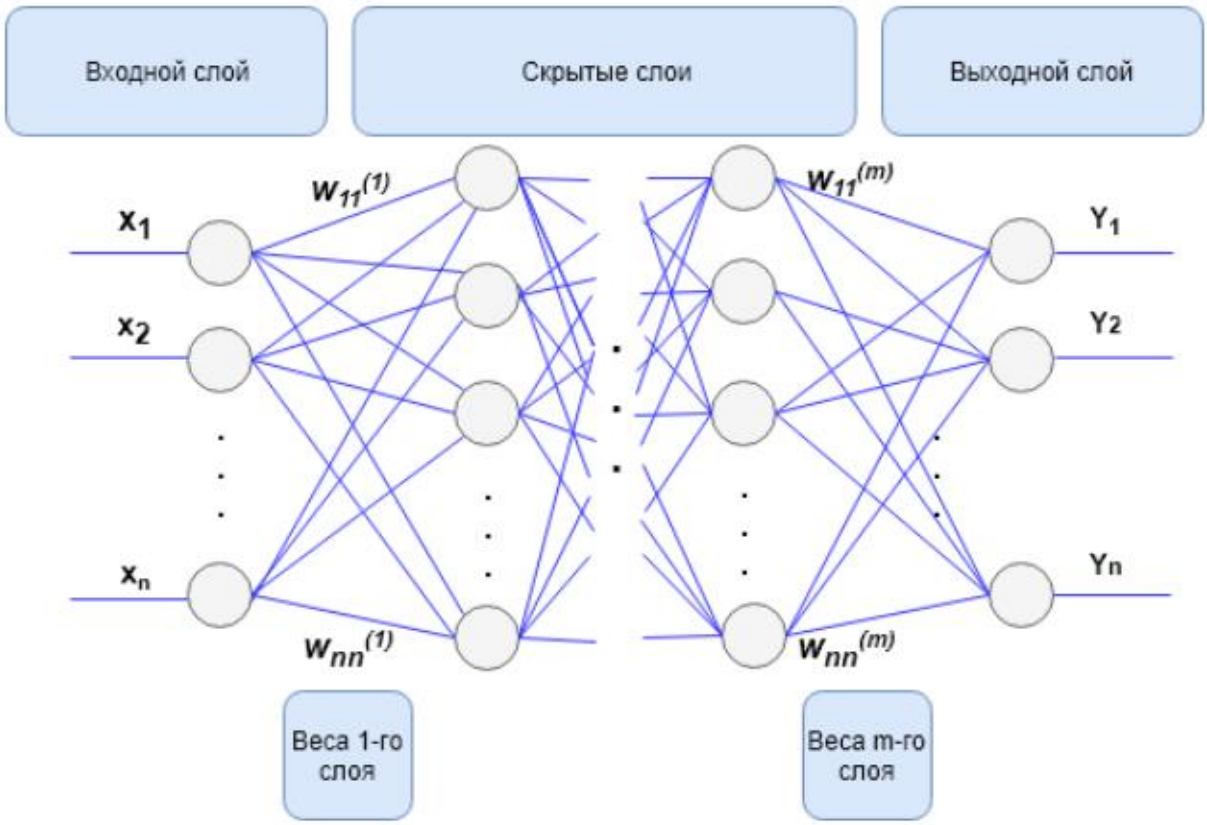

Рисунок 3 – Архитектура MLP [4]

Самый левый слой, известный как входной слой, который принимает исходный вектор данных $(x_1, x_2, ..., x_n)$. $(w_1, w_2, ..., w_n)$ – синаптические веса.

Каждый нейрон скрытого слоя преобразует значения предыдущего слоя с помощью взвешенного линейного суммирования. Выходной слой (правый) получает значения из последнего скрытого слоя и преобразует их в результат работы сети.

Алгоритм обучается по шагам, т. е. веса нейронов корректируются после подачи на вход обучающего примера. В каждом шаге происходит 2 прохода: первый проход называют прямым, второй — обратным. При прямом проходе синаптические веса не изменяются, а функциональные сигналы вычисляются он нейрона к нейрону, на выходе формируется выходной вектор, который соответствуют текущему состоянию весов. После этого вычисляется ошибка нейронной сети как разность фактическим и целевым значениями. При обратном проходе ошибка распространяется от выхода ко входам сети, и происходит корректировка весов нейроном [4].

Преимущества MLP:

- Возможность изучения нелинейных моделей.
- Возможность изучения моделей в режиме реального времени (онлайн-обучение).

К недостаткам MLP относятся:

- MLP со скрытыми слоями имеет невыпуклую функцию потерь, где существует более одного локального минимума. Поэтому разные инициализации случайных весов могут привести к разной точности проверки.
- MLP требует настройки ряда гиперпараметров, таких как количество скрытых нейронов, слоев и итераций.
- MLP чувствителен к масштабированию функций.

1.4.2 Модель CNN

Сверточная нейронная сеть (CNN) — это вид модели машинного обучения, а именно тип алгоритма глубокого обучения, хорошо подходящий для анализа визуальных данных. CNN использует принципы линейной алгебры, в частности операции свертки, для извлечения признаков и выявления закономерностей в изображениях. Хотя CNN в основном используются для обработки изображений, их также можно адаптировать для работы со звуком и другими сигнальными данными [5].

Архитектура CNN основана на схемах взаимодействия человеческого мозга, в частности, зрительной коры головного мозга, которая играет важную роль в восприятии и обработке зрительных стимулов. Искусственные нейроны в CNN устроены таким образом, чтобы эффективно интерпретировать визуальную информацию, что позволяет этим моделям обрабатывать целые изображения. Поскольку CNN настолько эффективны при идентификации объектов, они часто используются для задач компьютерного зрения. CNN также хорошо подходят для трансферного обучения, при котором предварительно обученная модель точно настраивается для новых задач. CNN используют серию слоев, каждый из которых обнаруживает различные особенности входного изображения. В зависимости от сложности предполагаемого назначения CNN может содержать десятки, сотни или даже тысячи слоев, каждый из которых основан на результатах предыдущих слоев для распознавания детальных закономерностей. Пример работы CNN можно увидеть на рисунке 4.

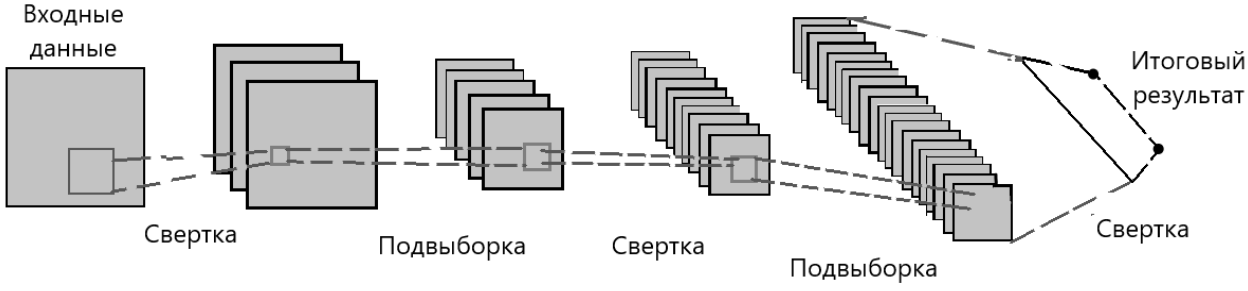

Рисунок 4 – Работа модели CNN

Преимущества CNN:

- Возможность повторного использования делает CNN универсальными и эффективными, особенно для задач с ограниченными обучающими данными.
- По сравнению с полносвязной нейронной сетью (типа перцептрона) — гораздо меньшее количество настраиваемых весов.
- Возможность развертывать на широком спектре устройств, включая мобильные устройства, такие как смартфоны, а также в сценариях периферийных вычислений.

Недостатки CNN:

- Слишком много варьируемых параметров сети; непонятно, для какой задачи и вычислительной мощности какие нужны настройки. Так, к варьируемым параметрам можно отнести: количество слоёв, размерность ядра свёртки для каждого из слоёв, количество ядер для каждого из слоёв, шаг сдвига ядра при обработке слоя и тд. Все эти параметры существенно влияют на результат, но выбираются исследователями эмпирически [6].

1.4.3 Модель LSTM

Долгая краткосрочная память (LSTM) – особая разновидность архитектуры рекуррентных нейронных сетей, способная к обучению долговременным зависимостям. Они прекрасно решают целый ряд разнообразных задач и в настоящее время широко используются.

LSTM разработаны специально, чтобы избежать проблемы долговременной зависимости. Запоминание информации на долгие периоды времени – это их обычное поведение, а не что-то, чему они с трудом пытаются обучиться [7]. Таким образом, хранимое значение не размывается во времени и градиент не исчезает при тренировке сети. На рисунке 5 показана работа модели LSTM.

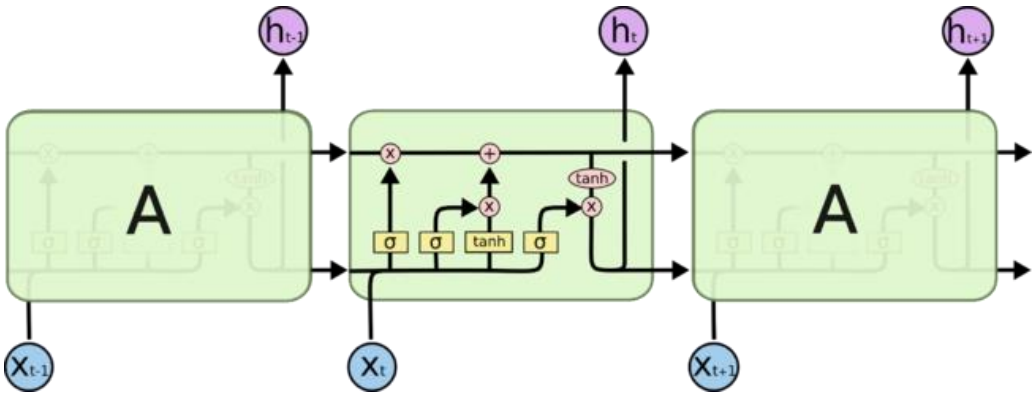

Рисунок 5 – Модель LSTM [6]

Ключевые компоненты модуля LSTM: состояние ячейки и различные фильтры. О состоянии ячейки можно говорить, как о памяти сети, которая передает соответствующую информацию по всей цепочке модулей. Таким образом, даже информация из ранних временных шагов может быть получена на более поздних, нивелируя эффект кратковременной памяти.

Преимущества LSTM:

- Отсутствие проблемы долговременной зависимости, что позволяет существенно повысить качество прогнозирования.
- Отсутствие необходимости проведения ряда тестов, к которым относится тест на стационарность, на наличие фактора сезонности и другие.
- Возможность использования рекуррентной нейронной сети со сверточными слоями в целях повышения точности прогноза.
- На основании информации тестовой выборки сравнительно высокий уровень точности прогноза.

Недостатки LSTM:

- Отсутствие возможности прогнозирования на основании малой выборки данных.
- Сложность подготовки данных к обучению.

1.4.4 Модель XGBoost

XGBoost (XGB) — это оптимизированная распределенная библиотека повышения градиента, разработанная для обеспечения высокой эффективности, гибкости и портативности. [8] Он реализует алгоритмы машинного обучения в рамках платформы градиентного бустинга. Она обеспечивает параллельное усиление деревьев и является ведущей библиотекой машинного обучения для решения задач регрессии, классификации и ранжирования.

Один и тот же код работает в основных распределенных средах и может решать проблемы.

Контролируемое машинное обучение использует алгоритмы для обучения модели поиску шаблонов в наборе данных с метками и функциями, а затем использует обученную модель для прогнозирования меток на объектах нового набора данных.

Деревья решений создают модель, которая предсказывает метку путем оценки дерева с истинными/ложными специальными вопросами и вопросами «если-то-иначе», и путем оценки минимального количества вопросов, необходимых для оценки вероятности принятия правильного решения. Деревья решений можно использовать для классификации для прогнозирования категории или регрессии для прогнозирования непрерывного числового значения.

Деревья решений градиентного бустинга (GBDT) — это алгоритм обучения ансамбля деревьев решений для классификации и регрессии. Алгоритмы ансамблевого обучения объединяют несколько алгоритмов машинного обучения для получения лучшей модели [9]. На рисунке 6 показан принцип работы модели XGB наглядно.

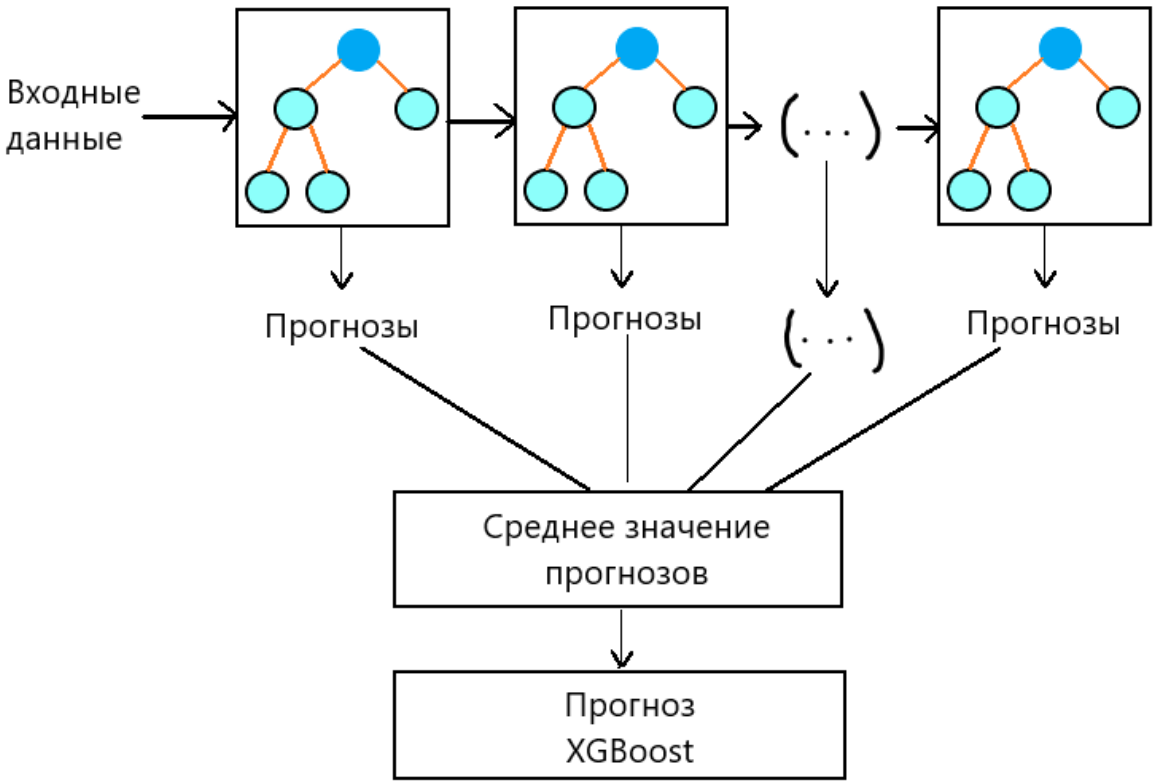

Рисунок 6 – Модель XGB

При использовании XGB деревья строятся параллельно, а не последовательно, как в GBDT. Он следует поуровневой стратегии, сканируя значения градиента и используя эти частичные суммы для оценки качества разделения при каждом возможном разделении в обучающем наборе.

Преимущества XGB:

- Вычисления для очень больших наборов данных, которые не вписываются в память.
- Параллелизация построения дерева с использованием всех доступных ядер процессора во время обучения.

Недостатки XGB:

- XGB имеет ограничения с точки зрения функции потерь и масштабируемости.
- Кроме того, XGB может оказаться неэффективным в вычислительном отношении для обработки больших наборов данных со сложной структурой данных. Эти ограничения могут снизить

производительность и эффективность XGB в определенных сценариях.

Вывод

Приведённые в данном разделе определения – это необходимый минимум для дальнейших размышлений. Чтобы окончательно убедиться, какая модель будет использоваться в дальнейшем для обучения и прогнозирования данных, попробуем обучить все, и выбрать лучшую из них. Для этого выберем языком программирования Python.

2 Моделирование

В этом разделе будут рассмотрены инструменты, использованные для моделирования, а также результаты различных моделей машинного обучения для выбора лучшей.

2.1 Используемые инструменты

Для реализации моделей прогнозирования в работе использовалась библиотека Scikit-learn [10]. Scikit-learn – это библиотека Python, которая специализируется на машинном обучении, например, моделировании данных. Scikit-learn построен на основе нескольких общих библиотек данных и математических библиотек, таких как:

- numpy [11] – библиотека, которая предоставляет функционал для поддержки многомерных массивов и математических функций, предназначенных для работы с массивами;
- pandas [12] – библиотека для обработки и анализа данных;
- и другие.

Это позволяет легко интегрировать их в работу.

Scikit-learn предлагает обширный набор инструментов для решения задач: регрессия, кластеризация, классификации. В данной библиотеке реализованы алгоритмы: нейронных сетей, SVM, деревья решений и др. Кроме в ней есть несколько удобных функций, которые обычно предлагаются не во всех библиотеках: обнаружение выбросов, оценка и проверка модели, управление характеристиками.

В работе также использовалась библиотека Pandas, для обработки и анализа данных. Данная библиотека предоставляет широкий функционал для работы с наборами данных. С ее помощь можно легко считать и сохранять данные из файлов, проводить предобработку данных, а затем использовать для обучения и тестирования моделей.

Разработка этого раздела проводилась в Jupyter Notebook. Jupyter Notebook – это интерактивный блокнот с открытым исходным кодом, используемый для интерактивных вычисления и визуализации результатов. Данный инструмент был выбран, т.к. с его помощью можно легко работать с файлами и наборами данных, визуализировать работу моделей, выполнять отдельные вычисления, а не выполнять всю программу.

Стандартизация набора данных является общим требованием для многих моделей машинного обучения: они могут вести себя некорректно, если отдельные функции не более или менее похожи на стандартные нормально распределенные данные. Стандартизация происходит, удалив среднее значение и масштабируя его до единичной дисперсии. В работе была использована функция StandartSkaler (рисунок 7) для дальнейшего обучения моделей.

```
from sklearn.preprocessing import StandardScaler

# Разделение входных объектов (X) и целевых объектов (y)
X = df.drop(['PowerConsumption_Zone1', 'PowerConsumption_Zone2',
'PowerConsumption_Zone3'], axis=1)
y = df[['PowerConsumption_Zone1', 'PowerConsumption_Zone2',
'PowerConsumption_Zone3']]

# Инициализировать StandardScaler для y
scaler_y = StandardScaler()

# Подгон и трансформация y
y_scaled = scaler_y.fit_transform(y)
X_train, X_test, y_train, y_test = train_test_split(X, y_scaled,
test_size=0.25, shuffle=False)
```

Рисунок 7 – Стандартизация

StandardScaler чувствителен к выбросам, и при наличии выбросов объекты могут масштабироваться по-разному.

2.2 Реализация обучения выбранных моделей

В разделе 1 были описаны 4 алгоритма, которые могут использоваться для задачи прогнозирования потребления электроэнергии: MLP, CNN, LSTM, XGB. Данные алгоритмы реализованы с помощью библиотеки Scikit-learn [10] (рисунки 8 – 11).

```
epochs = 40
batch = 256
lr = 0.0003
adam = optimizers.Adam(lr)
model_mlp = Sequential()
model_mlp.add(Dense(100, activation='relu', input_dim=X_train.shape[1]))
model_mlp.add(Dense(3))
model_mlp.compile(loss='mse', optimizer=adam)
model_mlp.summary()
mlp_history = model_mlp.fit(X_train.values, y_train,
                            validation_data=(X_test.values, y_test),
epochs=epochs, verbose=2)
```

Рисунок 8 – Модель MLP

```
model_cnn = Sequential()
model_cnn.add(Conv1D(filters=64, kernel_size=2, activation='relu',
input_shape=(X_train_series.shape[1], X_train_series.shape[2])))
model_cnn.add(MaxPooling1D(pool_size=2))
model_cnn.add(Flatten())
model_cnn.add(Dense(50, activation='relu'))
model_cnn.add(Dense(3))
model_cnn.compile(loss='mse', optimizer='adam')
model_cnn.summary()
cnn_history = model_cnn.fit(X_train_series, y_train,
validation_data=(X_test_series, y_test), epochs=epochs, verbose=2)
```

Рисунок 9 – Модель CNN

```
model_lstm = Sequential()
model_lstm.add(LSTM(50, activation='relu',
input_shape=(X_train_series.shape[1], X_train_series.shape[2])))
model_lstm.add(Dense(3))
model_lstm.compile(loss='mse', optimizer='adam')
model_lstm.summary()
lstm_history = model_lstm.fit(X_train_series, y_train,
validation_data=(X_test_series, y_test), epochs=epochs, verbose=2)
```

Рисунок 10 – Модель LSTM

```
param_grid = {
    'objective':['reg:squarederror'],
    'max_depth': [2, 5, 7],
    'learning_rate': [0.1, 0.01, 0.001],
    'subsample': [0.5, 0.7],
    'n_estimators': [1000,1500,2000],
    'min_child_weight': [1, 2],
    'booster':['gbtree']
}
xgb_model = xgb.XGBRegressor(tree_method='hist', device = 'cuda')
grid_search = GridSearchCV(xgb_model, param_grid, cv=7,
scoring='neg_mean_squared_error')
grid_search.fit(X_train, y_train)
```

Рисунок 11 – Модель XGB

Обучение моделей произведено при помощи функции fit(X_train, y_train) [10], с использование тренировочного набора данных. После обучения, созданные модели могут предсказывать будущее значение потребления электроэнергии.

2.3 Метрики для оценки результатов

Точность прогнозирования характеризуется степенью соответствия величины электропотребления, полученной в результате прогноза, и фактической величины электропотребления. Одним из важнейших этапов ИАД является оценка обученной модели. Для оценки работы модели необходимо выбрать меры качества.

В общем случае для оценки точности рекомендуется использовать следующие показатели: Средняя абсолютная ошибка прогноза (Mean absolute error, MAE) [13]:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left| f(x_i) - y_i \right|,$$

где x_i – фактическое значение параметра; y_i – прогнозное значение параметра; n – количество точек на оцениваемом интервале.

Средняя абсолютная ошибка показывает среднее значение абсолютных значений ошибки прогноза для каждого экземпляра из тестового набора. Ошибка прогноза – это разность между фактическим и спрогнозированным значением.

Средняя квадратичная ошибка (Mean squared error, MSE) [13]:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(f(x_i) - y_i)^2$$

Средняя квадратичная ошибка измеряет среднее значение квадратов ошибок, т.е. среднеквадратичную разницу между предсказанным значением и фактическим знамением.

Средняя квадратичная ошибка более чувствительна к выбросам, по сравнению со средней абсолютной, поэтому сильнее штрафует за большие отклонения.

3 Разработка приложения

Для одноразового решения задачи предсказания потребления электроэнергии было бы достаточным реализовать модель машинного обучения, выбранной в разделе 3, которая бы работала, как просто программный файл, выполняющий необходимый пользователю сценарий в виде кода. Однако для более эргономичного использования для пользователей и возможного расширения функционала, была разработана программа, решающая поставленную задачу.

3.1 Подготовительный этап

В данном разделе будет рассмотрен набор данных, в котором содержится информация о электропотреблении в трех разных зонах города Тетуан в Марокко [14], а также о погодных условиях в этом городе.

Набор данных содержит 52416 строк и 9 столбцов. В них входят столбцы о погоде в Тетуане, о дате и времени замеров, а также о трех зонах электропотребления (рисунок 12).

	Datetime	Temperature	Humidity	WindSpeed	GeneralDiffuseFlows	DiffuseFlows	PowerConsumption_Zone1	PowerConsumption_Zone2	PowerConsumption_Zone3
0	1/1/2017 0:00	6.559	73.8	0.083	0.051	0.119	34055.69620	16128.87538	20240.96386
1	1/1/2017 0:10	6.414	74.5	0.083	0.070	0.085	29814.68354	19375.07599	20131.08434
2	1/1/2017 0:20	6.313	74.5	0.080	0.062	0.100	29128.10127	19006.68693	19668.43373
3	1/1/2017 0:30	6.121	75.0	0.083	0.091	0.096	28228.86076	18361.09422	18899.27711
4	1/1/2017 0:40	5.921	75.7	0.081	0.048	0.085	27335.69620	17872.34043	18442.40964

Рисунок 12 – Столбцы набора данных.

Названия столбцов на русском языке слева направо: Дата и время (временной интервал в 10 минут); Температура (в градусах Цельсия); Влажность (%); Скорость ветра (км/ч); Общие диффузные потоки (км/ч); Зона электропотребления 1; Зона электропотребления 2; Зона электропотребления 3.

3.2 Нормализация и выбросы

В ходе исследования была проведена проверка на наличие выбросов, а также пустых ячеек. Если ячейки не содержали значений, то они изымались вместе со строкой. В работе не было обнаружено подобных строк (рисунок 13). Также набор данных был проверен на сохранение хронологии и равные промежутки времени между соседними измерениями, проверка также прошла успешно (рисунок 14).

```
In [10]: df.isna().sum()

Out[10]:
Datetime                 0
Temperature              0
Humidity                 0
WindSpeed                0
GeneralDiffuseFlows      0
DiffuseFlows             0
PowerConsumption_Zone1   0
PowerConsumption_Zone2   0
PowerConsumption_Zone3   0
dtype: int64
```

Рисунок 13 – Проверка данных на наличие пустых ячеек

```
In [8]: df['Datetime']=pd.to_datetime(df.Datetime)
        df.sort_values(by='Datetime', ascending=True, inplace=True)

        chronological_order = df['Datetime'].is_monotonic_increasing

        time_diffs = df['Datetime'].diff()
        equidistant_timestamps = time_diffs.nunique() == 1

In [9]: chronological_order, equidistant_timestamps

Out[9]: (True, True)
```

Рисунок 14 – Проверка данных на верную хронологию и равные промежутки времени между данными

Из этих проверок следует, что данные не нуждаются в корректировке и дополнении, следовательно, можно переходить к следующему этапу – нормализации.

Нормализация данных в сформированных наборах следующим образом:

$$Z = \frac{X - \min(X)}{\max(X) - \min(x)},$$

где X — входной вектор данных, Z — нормализованный вектор.

3.3 Сравнение результатов работы выбранных моделей

В результате обучения моделей были получены прогнозы для трех зон в городе Тетуан по потреблению электроэнергии на тестовой выборке. Для сравнения и выбора лучшей модели были посчитаны метрики MAE и MSE, результаты которых представлены в таблице 1 и рисунках 15-16.

Таблица 1 – MAE и MSE прогнозирования потребления электроэнергии на тестовой выборке

	MLP	CNN	LSTM	XGB
MAE	1,6167	0,6962	0,8722	0,4242
MSE	4,0666	0,7888	1,1600	0,2904

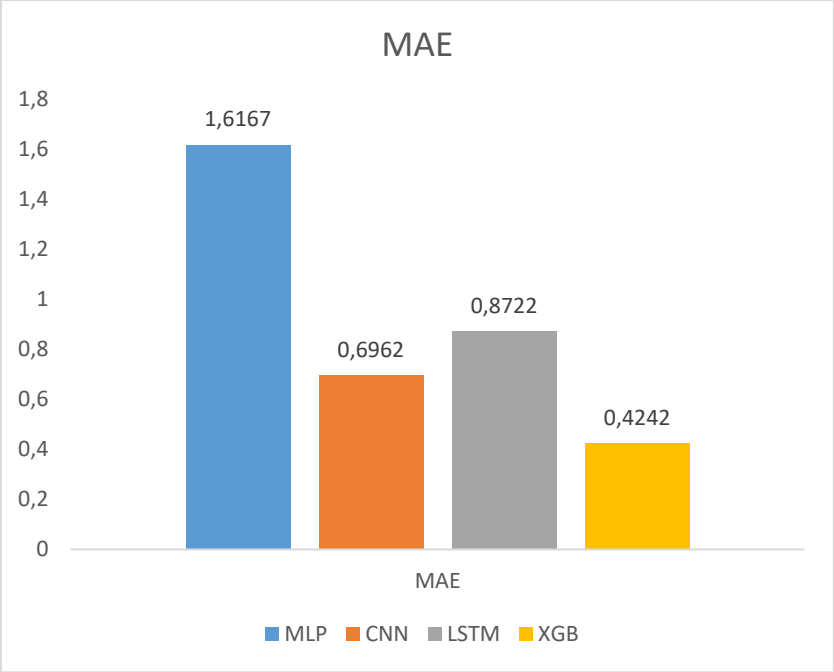

Рисунок 15 – MAE для прогноза электропотребления

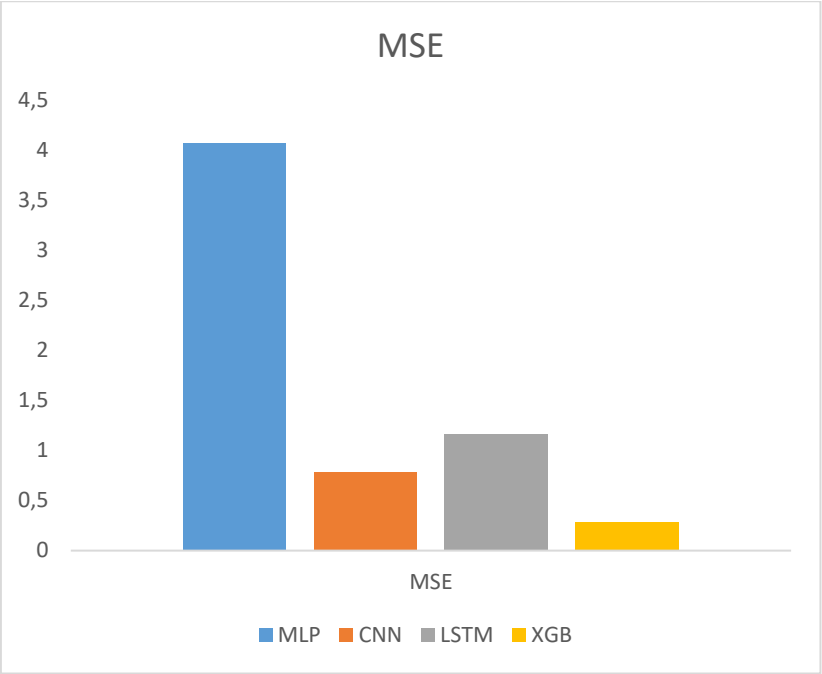

Рисунок 16 – MSE для прогноза электропотребления

В процессе проведенного исследования получены значения среднеквадратических ошибок (MSE) и средних абсолютных ошибок (MAE). Это было необходимо для того, чтобы оценить работу алгоритмов следующих моделей машинного обучения: MLP, CNN, LSTM, XGB. Также были изучены различия между этими моделями.

Наилучший результат по MSE и MAE в обоих случаях получает модель XGB. По данным подраздела 3.2 можно известно, что в исходных данных не были найдены пропущенные значения. Это значит, что ничто не может сказываться плохо на прогностических способностях моделей, следовательно, можно считать полученные результаты в разделе 3 достоверными.

Таким образом, для дальнейшего прогнозирования потребления электроэнергии на основе погодных условий будем использовать модель XGB.

3.4 Используемые инструменты

Для того, чтобы сделать возможным работу программы в виде приложения, была использована библиотека tkinter. Пакет tkinter представляет собой стандартный интерфейс Python для набора инструментов Tcl/Tk GUI,

который добавляет пользовательские команды для создания виджетов графического интерфейса и управления ими [15].

Поддержка Tkinter распределена по нескольким модулям. Для приложения использовались основной tkinter модуль, а также tkinter.ttk модуль, предоставляющий современный набор тематических виджетов и API.

Для создания главного окна была использована функция Tk(), представленная на рисунке 17.

```
# Создание главного окна
root = tk.Tk()
root.title("Прогнозирование потребления электричества")
```

Рисунок 17 – Создание главного окна

Также были использованы функции Frame(), Label(), Button() для создания фреймов, текстовых меток и кнопок соответственно. В результате получилось приложение, представленное на рисунке 18.

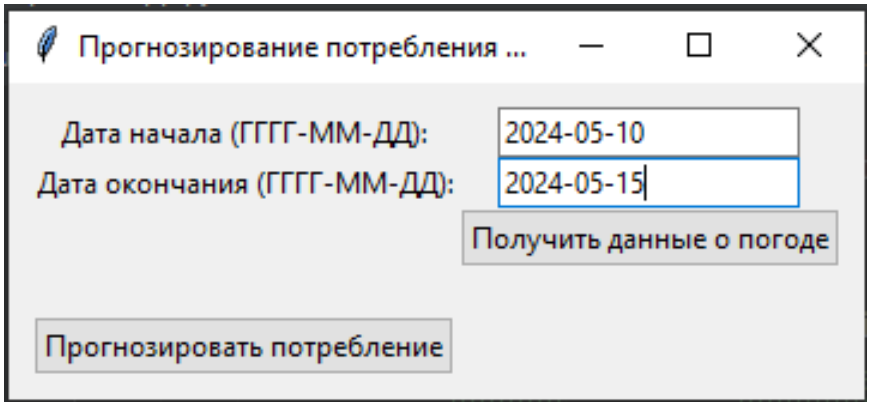

Рисунок 18 – Приложение по прогнозированию потребления электроэнергии

3.5 Алгоритм приложения

Структурная схема приложения представляет собой наглядное представление функциональных возможностей системы приложения, функций и других операций.

На рисунке 19, представленном ниже, показана структурная схема приложения.

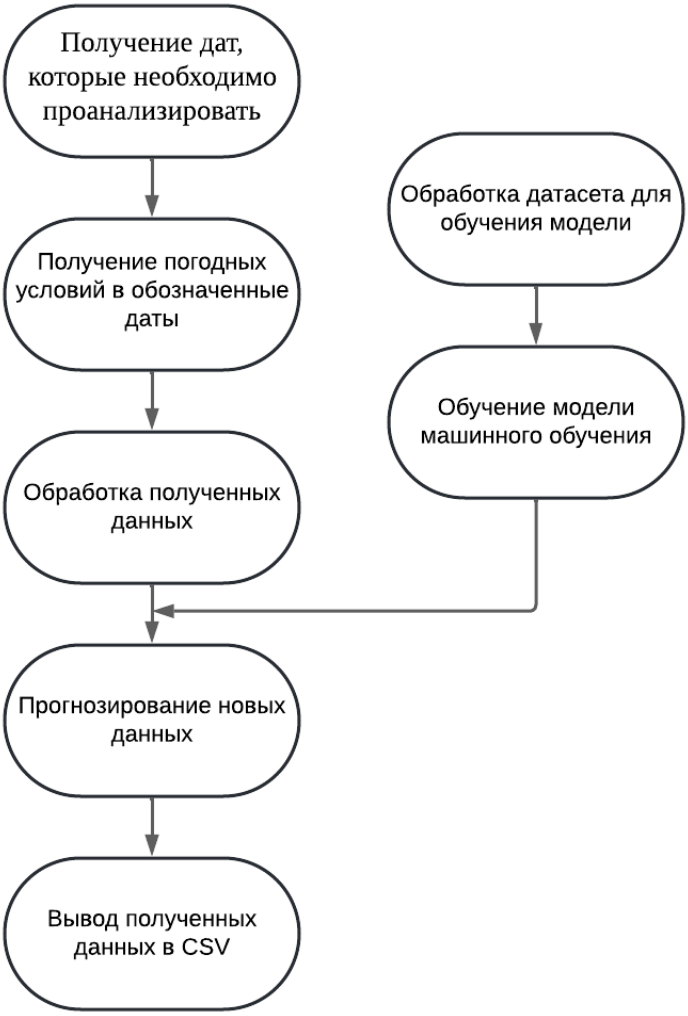

Рисунок 19 – Структурная схема приложения

В целом, существует семь этапов жизненного цикла приложения, как показано выше на рисунке 19. Каждый этап объясняется следующим образом.

- **Получение дат, которые необходимо проанализировать**

Чтобы программа получила даты, которые необходимо проанализировать, необходимо, чтобы пользователь ввел нужные ему даты в окна, показанные на рисунке 20.

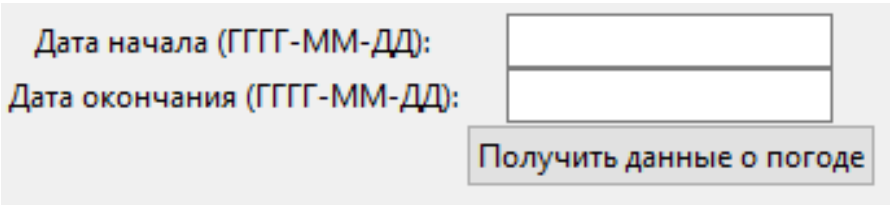

Рисунок 20 – Окна для ввода дат начала прогнозирования и его окончания

- **Получение погодных условий в обозначенные даты**

После ввода необходимых для прогнозирования дат, необходимо нажать на кнопку «Получить данные о погоде», чтобы программа получила данные о погодных условиях почасово, благодаря библиотеке meteostat, и загрузила полученные погодные условия в файл CSV, который будет использоваться в прогнозировании новых данных. После нажатия кнопки появится окно, показанное на рисунке 21.

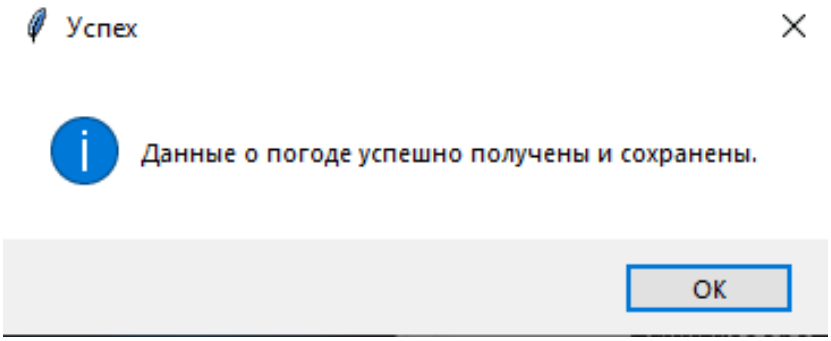

Рисунок 21 – Окно, извещающее об успешном получении погодных условий

Из-за того, что датасет, который используется для обучения, основан на данных города Тетуан, для meteostat были взяты координаты города и введены в программу без возможности их изменения пользователем.

Особенность библиотеки состоит в том, что крайние погодные условия, которые она может выгрузить, ограничиваются двумя следующими сутками от дня обращения (рисунок 22). При введении даты, которая будет, к примеру, через две недели с даты обращения, программа загрузит в файл CSV только столбец с датой и временем указанного промежутка.

lated_app.py × cleaned_weather_data.csv × predicted_power_consumption.csv ×

```
temp,dwpt,rhum,prcp,wdir,wspd,pres,coco,time
17.0,11.1,68.0,0.0,250.0,22.0,1019.0,3.0,2024-05-01 00:00:00
17.0,11.1,68.0,0.0,260.0,19.0,1018.0,3.0,2024-05-01 01:00:00
17.0,11.1,68.0,0.0,250.0,26.0,1017.0,3.0,2024-05-01 02:00:00
17.0,11.1,68.0,0.0,240.0,26.0,1016.0,3.0,2024-05-01 03:00:00
17.0,11.1,68.0,0.0,250.0,30.0,1016.0,3.0,2024-05-01 04:00:00
16.0,12.0,77.0,0.0,240.0,24.0,1015.0,3.0,2024-05-01 05:00:00
17.0,11.9,72.0,0.1,250.0,13.0,1015.0,7.0,2024-05-01 06:00:00
15.0,12.0,82.0,0.1,250.0,26.0,1015.0,17.0,2024-05-01 07:00:00
15.0,13.0,88.0,0.3,230.0,17.0,1015.0,17.0,2024-05-01 08:00:00
16.0,12.9,82.0,0.6,240.0,32.0,1015.0,8.0,2024-05-01 09:00:00
14.0,13.0,94.0,1.4,330.0,15.0,1016.0,9.0,2024-05-01 10:00:00
15.0,12.0,82.0,2.4,310.0,9.0,1017.0,9.0,2024-05-01 11:00:00
16.0,12.9,82.0,0.5,250.0,20.0,1016.0,8.0,2024-05-01 12:00:00
```

Рисунок 22 – CSV файл с погодными условиями

• **Обработка полученных данных**

После того, как набор данных новых погодных условий был получен, этот набор данных обрабатывается системой, чтобы в прогнозировании данных не было проблем с данными для обучения.

В процессе обработки данных были удалены столбцы, в которых отсутствовали значения, а также столбцы, которые не нужны для дальнейшего использования.

• **Обработка датасета для обучения модели**

Для прогнозирования данных необходимо обработать датасет, который позже будет использоваться для обучения модели машинного обучения. В данном случае, был взять датасет [14], в котором содержатся данные о погодных условиях (температура, влажность, скорость ветра, общие диффузные потоки, диффузные потоки), а также о 3 зонах потребления электроэнергии в городе Тетуан, Марокко. Датасет был уже рассмотрен и проанализирован в разделах 2 и 3, поэтому в программе будет проведена только необходимая обработка.

Чтобы датасет согласовывался с новым набором обработанных данных из предыдущего пункта, необходимо преобразовать некоторые столбцы в

нужный формат, а также удалить ненужные данные. Для программы были удалены некоторые столбцы и строки, которые не использовались для прогнозирования.

- **Обучение модели машинного обучения**

В подразделе 3.3 было подробно рассмотрено моделирование и обучение четырех разных моделей машинного обучения. Из них была выбрана лучшая модель на основе значений MSE и MAE – модель XGB.

- **Прогнозирование новых данных**

Для прогнозирования потребления электроэнергии для дат, введенных в самом начале работы программы (получение дат, которые необходимо прогнозировать), необходимо нажать на кнопку «Прогнозировать потребление» (рисунок 23).

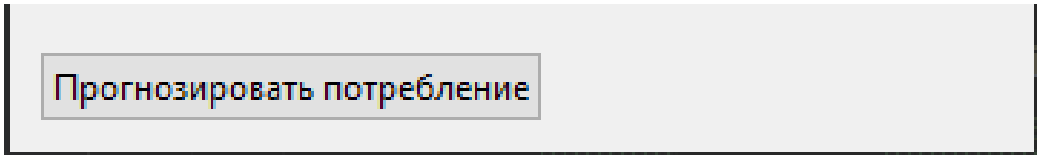

Рисунок 23 – Кнопка прогнозирования потребления

Для прогнозирования использовалась функция predict().

- **Вывод полученных данных в CSV**

Полученные данные из прошлого пункта переносятся в файл CSV для более удобного рассмотрения и дальнейшнего использования. В файле расположено семь столбцов: три столбца для погодных условий (температура, влажность и скорость ветра), три столбца для каждой зоны электропотребления и один столбец для даты и времени. На рисунке 24 представлена часть CSV файла с предсказанным потреблением электроэнергии для трех зон города Тетуан.

	Temp...	Hum...	Win...	time	Power...	Power...	Power...
1	17.2	65.0	14.8	2023-05-01 00:…	31085.682	19355.494	15604.441
2	18.0	64.0	16.6	2023-05-01 01:…	33776.926	25508.967	15131.155
3	17.0	68.0	9.4	2023-05-01 02:…	27678.912	18267.951	13206.802
4	15.5	72.0	18.4	2023-05-01 03:…	29814.879	19461.885	12077.037
5	15.0	72.0	18.4	2023-05-01 04:…	30727.373	19100.951	11140.126
6	14.0	77.0	13.0	2023-05-01 05:…	26938.361	18729.418	10985.0625
7	12.6	78.0	14.8	2023-05-01 06:…	28577.887	17431.592	14072.252
8	16.0	72.0	16.6	2023-05-01 07:…	29936.922	20864.285	9664.776
9	20.0	56.0	9.4	2023-05-01 08:…	39168.504	19912.281	13061.847
10	24.2	48.0	7.6	2023-05-01 09:…	32790.477	20110.783	13560.5625
11	23.0	61.0	13.0	2023-05-01 10:…	36916.363	22397.9	14605.953

Рисунок 24 – Выходной CSV файл

На рисунке 25 показаны полученные оценки для этого прогноза.

```
MSE for on test set: 6.1937 for PowerConsumption_Zone1
MAE for on test set: 4.8762 for PowerConsumption_Zone1
MSE for on test set: 4.5536 for PowerConsumption_Zone2
MAE for on test set: 3.5212 for PowerConsumption_Zone2
MSE for on test set: 5.0355 for PowerConsumption_Zone3
MAE for on test set: 3.8552 for PowerConsumption_Zone3
```

Рисунок 25 – Полученные метрики для каждой из трех зон

Как видно на рисунке, метрики для разных зон варьируются в районе 3,5 – 6,2. Подобный результат обуславливается тем, что для прогнозирования выбрана небольшая выборка данных.

Выводы

В результате выполнения данного раздела были созданы программа и приложение для прогнозирования потребления электроэнергии на основе погодных условий. Приложение удобно в использовании и интуитивно понятно даже при первом использовании.

Программа работает корректно и верно, проверка была осуществлена с помощью открытых источников в Интернете.

4 Экономическое обоснование

В данном разделе производится расчёт себестоимости исследования в соответствии с [16].

4.1 Концепция экономического обоснования

Экономическое обоснование необходимо для оценки себестоимости разработки программного продукта. На основании этого можно сделать выводы об экономической эффективности проекта.

Целью данного проекта является разработка приложения для ознакомления с вероятными затратами на электроэнергию в Марокко. Была создана программа, для прогноза электропотребления на основе погодных условий. С помощью этой информации предприниматель может распределить электроэнергию для обслуживания потребителей. Поскольку потребление электроэнергии имеет решающее значение для страны, идея состоит в том, чтобы изучить влияние на потребление энергии. На рынке невозможно найти готовый продукт, подходящий по требованиям.

Система реализуется с помощью свободно распространяемого в исходном коде дистрибутива Anaconda для языка Python 3.7 на операционной системе Windows 10.

Все продукты обладают открытой лицензией и не требуют затрат на их использование. В данной главе будут рассмотрены и подсчитаны такие экономические показатели, как основная и дополнительная заработные платы, отчисления на социальные нужды, затраты на материалы, затраты на содержание и эксплуатацию оборудования, амортизационные отчисления, накладные расходы и совокупная величина затрат, связанных с выполнением ВКР.

4.2 Расчёт расходов на оплату труда исполнителей

Для каждого участника проекта рассчитывается заработная плата, на основе месячной ставки. Дневная ставка заработной платы руководителя проекта и разработчика определяется отношением оклада за месяц к количеству часов в месяце, а точнее 168 часов. Согласно «Положению об оплате труда работников СПбГЭТУ «ЛЭТИ» ред. с 15.10.2020 г.» [17] зарплата руководителя, как кандидата технических наук и доцента равна 47 500 рублей, для инженера равна 14 000 рублей. Исходя из полученных данных рассчитаем часовую ставку руководителя проекта и инженера:

$$C_{рук} = \frac{44600}{168} = 265 \text{ руб./день (часовая ставка руководителя проекта)}$$

$$C_{инж} = \frac{14500}{168} = 83 \text{ руб./час (часовая ставка инженера)}$$

Для расчета затрат, необходимо определить продолжительность каждой работы. При определении продолжительности учитывается время, фактически затрачиваемое исполнителями на выполнение каждой работы. Длительность каждой работы измеряется в человеко-часах. Все работы выполняет студент, практически на всех этапах его консультирует руководитель. Продолжительность этапов работы представлена в таблице 2.

Таблица 2 – Трудоемкость работ

№ п/п	Наименование работы	Исполнитель	Трудоемкость, чел/часы	Ставка, руб/час
1	Разработка ТЗ	Инженер	15	83
		Руководитель	8	265
2	Анализ ТЗ, работа с источниками	Инженер	20	83
		Руководитель	2	265
3	Изучение существующих систем	Инженер	8	83
		Руководитель	1	265
4	Выбор методов решения задачи, анализ данных	Инженер	40	83
		Руководитель	10	265

Продолжение таблицы 2

№ п/п	Наименование работы	Исполнитель	Трудоемкость, чел/часы	Ставка, руб/час
5	Разработка и написание программы	Инженер	120	83
		Руководитель	-	265
6	Тестирование и отладка программы	Инженер	60	83
		Руководитель	8	265
7	Оформление пояснительной записки	Инженер	40	83
		Руководитель	-	265
8	Оформление иллюстративных материалов	Инженер	10	83
		Руководитель	-	265
9	Сдача проекта	Инженер	8	83
		Руководитель	2	265
ИТОГО		Инженер	321	26 643
		Руководитель	38	12190

Расходы на основную заработную плату исполнителей определяются по формуле (1):

$$З_{осн.з/пл} = \sum_{i=1}^{k} T_i \cdot C_i \,, \qquad (1)$$

где k – количество исполнителей, T_i – время, затраченное исполнителем на проведение разработки, C_i – ставка исполнителя.

На основе формулы 1 рассчитывается основная заработная плата:

$$З_{осн.з/пл} = \sum_{i=1}^{k} T_i \cdot C_i = 321 \cdot 83 + 38 \cdot 265 =$$
$$= 26\ 643 + 10\ 070 = 36\ 713 \text{ руб.}$$

Дополнительная заработная плата исполнителей рассчитывается по формуле (2):

$$З_{доп.з/пл} = З_{осн.з/пл} \cdot \frac{Н_{доп}}{100}, \quad (2)$$

где $З_{доп.з/пл}$ – расходы на дополнительную заработную плату исполнителей, руб.; $З_{осн.з/пл}$ – расходы на основную заработную плату исполнителей, руб.; $Н_{доп}$ – норматив дополнительной заработной платы, %.

Рассчитаны расходы по дополнительной заработной плате исполнителей на основе формулы (2):

$$З_{доп.з/пл} = З_{осн.з/пл} * \frac{Н_{доп}}{100} = 36\,713 * \frac{14}{100} = 5\,139{,}82 \text{ руб.}$$

В итоге расходы на дополнительную заработную плату для исполнителей равны 5 139,82 рублей.

При выполнении расчетов в ВКР норматив дополнительной заработной платы принимаем равным 14 %.

Расходы на оплату труда по основной и дополнительной заработной плате равны 41 852,82 рублей.

4.3 Расчет отчислений на социальные нужды

Отчисления на страховые взносы на обязательное социальное, пенсионное и медицинское страхование с основной и дополнительной заработной платы исполнителей определяются по формуле (3):

$$З_{соц} = (З_{осн.з/пл} + З_{доп.з/пл}) \cdot \frac{Н_{соц}}{100}, \quad (3)$$

где Зсоц – отчисления на социальные нужды с заработной платы, руб; Нсоц – норматив отчислений страховых взносов, он составляет 30%.

Отчисления на социальные нужды рассчитывается по формуле (3):

$$З_{соц} = (З_{осн.з/пл} + З_{доп.з/пл}) * \frac{Н_{соц}}{100} = (36\,713 + 5139{,}82) * 0{,}3 =$$
$$= 12\,555{,}85 \text{ руб.}$$

Итог расходов на социальные нужды 12 555,85 рублей.

4.4. Расчет затрат на содержание и эксплуатацию оборудования

Были определены расходы на содержание и эксплуатацию оборудования из расчета на 1 час работы оборудования с учетом стоимости и производительности по формуле (4):

$$З_{эо} = \sum_{i=1}^{m} C_i^{мч} \cdot t_i^{м},$$

где $З_{эо}$ – затраты на содержание и эксплуатацию оборудования, руб.; $C_i^{мч}$ – расчетная себестоимость одного машино-часа работы оборудования на *i*-й технологической операции, руб./м-ч*м; $t_i^{м}$ – количество машино-часов, затрачиваемых на выполнение *i*-й технологической операции, м-ч.

В течение всего периода работ использовался интернет Ростелеком с тарифом Облачный 990 руб./мес. [17]. Время, в течение которого используется интернет: 321 : 168 = 1,9 месяцев. Итоговые затраты на интернет: 990 * 1,9 = = 1 881 руб.

Также учитывается, что при разработке программы использовался ноутбук Asus K543U. Мощность потребления была рассчитана по параметрам блока зарядки показанный рис. 1: 19V * 2.37A = 45 Вт/час = 0,045 кВт/час.

По причине постоянного использования ноутбука с начала и до конца проекта, то можно взять без изменений времени работы инженера, за исключением сдачи ВКР: 321 - 8 = 313 машинно-часов.

Себестоимость машино-часа равна стоимости по тарифу на электроэнергию (одноставочный тариф 5,7 руб за кВт/ч [19]) умноженная на мощность оборудования: 5,7 * 0,045 = 0,26 руб/час.

Рассчитываем расход на содержание и эксплуатацию оборудования на основе формулы (4):

Зэо = 0,26 * 313 +1 881 = 1 962,4 руб.

4.5 Расчет затрат на амортизацию

Амортизационные отчисления по основному средству за год определяются по формуле (5):

$$А=Ц_{ц.н.}\frac{Н_{а}}{100},$$

где – амортизационные отчисления за год по основному средству (руб.), $Ц_{ц.н.}$ – первоначальная стоимость основного средства (руб.), $Н_а$ – годовая норма амортизации основного средства (%).

Налоговый кодекс четко требует от владельцев правильно распределять оборудования по амортизационным группам в соответствии с утвержденным Постановлением Правительства РФ от 01.01.2002 N 1 (ред. от 28.04.2018) «О Классификации основных средств, включаемых в амортизационные группы». По утвержденной Классификации ноутбук можно отнести ко 2 амортизационной группе, для которой определен полезный срок использования от 2 до 3 лет включительно. Компьютерная техника по ОКОФ имеет код 330.28.23.23 и именуется «Машины офисные прочие». Считая, что оборудование надежное принимаем время эксплуатации техники три года.

Тогда получается:

$$H_{AM}=\frac{100\%}{3}=33,3\%$$

При разработке программы использовался ноутбук Asus K543U стоимостью 45 000 рублей.

А = 45 000 * 33,3 : 100 = 14 985 руб.

Тогда определим величину амортизационных отчислений по основному средству, используемых при работе над ВКР, по формуле (6):

$$A_{ВКР}=A*\frac{T_{ВКР}}{12},$$

где $А_{ВКР}$ – амортизационные отчисления по основному средству, используемому студентом в работе над ВКР (руб.); А – амортизационные

отчисления за год по основному средству (руб.); $Т_{ВКР}$ – время, в течение которого студент использует основное средство (мес.).

$$T_{ВКР} = \frac{321}{168} = 1,9 \text{ мес.}$$

Величина амортизационных отчислений по основному средству рассчитывается по формуле (6):

$$A_{ВКР} = 14985 * \frac{1,9}{12} = 2\ 372,63 \text{ руб.}$$

4.6 Расчет накладных расходов

Далее рассмотрим накладные расходы. К ним относятся обслуживание помещений и оборудования, а также управление процессами. Для их расчёта используется формула (7):

$$С_{нр} = З_{полн.з/п} * 0,4,$$

где $З_{полн.опл.тр}$ – это полная оплата труда. Если подставить в формулу (7) рассчитанное в разделе 1.2 значение, получим:

$$С_{нр} = 41\ 852,82 * 0,4 = 16\ 741,13 \text{ руб.}$$

4.7 Расчет суммарной стоимости выполнения ВКР

Чтобы подвести итоговые траты, рассмотрим суммарные расходы, представленные в таблице 3.

Таблица 3 – итоговые траты

№ п/п	Наименование статьи	Сумма, руб
1	Расходы на оплату труда	41 852,82
2	Отчисления на социальные нужды	12 555,85
3	Расход на содержание и эксплуатацию оборудования	1 962,4
4	Амортизационные отчисления	2 372, 63

Продолжение таблицы 3

№ п/п	Наименование статьи	Сумма, руб
5	Накладные расходы	16 741,13
ИТОГО		75 484,83

4.8 Выводы

В разделе экономическое обоснование была выполнена оценка себестоимости разработки модели и написания ВКР. Выполнение работы заняло 321 рабочих часов. Итоговая стоимость работы равна 75 484,83 рублей.

Разработанная модель прогнозирует электропотребление, основываясь на прогнозе погоды. С помощью этой информации предприниматель может распределить электроэнергию для обслуживания потребителей. Из этого следует, что создание подобной системы является экономически целесообразной.

ЗАКЛЮЧЕНИЕ

В результате дипломной работы был проведен обзор методов интеллектуального анализа данных, которые могут быть использованы для потребления электроэнергии в зависимости от погодных условий. Были рассмотрены 4 модели машинного обучения для прогнозирования электропотребления: модели MLP, CNN, LSTM и XGB.

Были подготовлены данные для обучения моделей. После обучения были получены прогнозы на тестовых данных, которые позволили сравнить модели для решения поставленной задачи. Результаты обучения были сравнены по двум метрикам: MAE и MSE. По итогам сравнения для решения задачи была выбрана модель XGB. Результаты исследования показали, что выбранная модель дает результат с высокой точностью и может быть использована на практике.

После выбора модели для дальнейшего прогнозирования была разработана программа для прогнозирования потребления электроэнергии на выбранных датах, не содержащихся в данных для обучения модели. Также было разработано приложение для более удобного использования пользователями.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Этапы машинного обучения, Викиконспект [Электронный ресурс]. – 2022. – URL: https://neerc.ifmo.ru/wiki/index.php?title=Жизненный_цикл_модели_машинного_обучения (дата обращения 22.02.2024)

2. Методы машинного обучения, Big Data School [Электронный ресурс]. – 2023. – URL: https://bigdataschool.ru/wiki/machine-learning (дата обращения 26.02.2024)

3. Хайндман, Р. Дж., И Атанасопулос, Г. Прогнозирование: принципы и практика – 2-е издание – Мельбурн, Австралия; OTexts. – 2018.

4. Рашка С. Python и машинное обучение. – М.: ДМК Пресс. – 2017. – 848 с.

5. Techtarget CNN [Электронный ресурс]. – 2024. – URL: https://www.techtarget.com/searchenterpriseai/definition/convolutional-neural-network (дата обращения 01.03.2024)

6. Wikipedia сверточная нейронная сеть [Электронный ресурс]. – 2024 – URL:https://ru.wikipedia.org/wiki/Свёрточная_нейронная_сеть (дата обращения 02.03.2024)

7. Хабр, модель LSTM [Электронный ресурс]. – 2017. – URL: https://habr.com/ru/companies/wunderfund/articles/331310/ (дата обращения 04.03.2024)

8. Макаров, Д. А., Использование алгоритма XGBoost для предсказания завершения курса обучающимся / Д. А. Макаров // Научно-образовательный журнал «StudNet» – М., 2021. – № 1. – С. 10.

9. Модель XGBoost [Электронный ресурс]. – 2020. – URL: https://www.nvidia.com/en-us/glossary/xgboost/ (дата обращения 07.03.2024)

10. Scikit–learm documentation. URL: https://scikit-learn.org/ (дата обращения 17.03.2024)

11. Numpy documentation. URL: https://www.numpy.org/ (дата обращения 20.03.2024)

12. Pandas 0.24.2 documentation; [Электронный ресурс]. – 2024. – URL: https://pandas.pydata.org/pandas-docs/stable/index.html (дата обращения 22.03.2024)

13. Курс Евгения Соколова: Семинар по выбору моделей; machinelearning [Электронный ресурс]. – 2024. – URL: http://www.machinelearning.ru/wiki/images/1/1c/Sem06_metrics.pdf (дата обращения: 28.03.2024).

14. Датасет потребления электроэнергии с погодными условиями города Тетуан [Электронный ресурс]. – 2022. – URL: https://www.kaggle.com/datasets/fedesoriano/electric-power-consumption (дата обращения 22.02.2024)

15. Tkinter documentation [Электронный ресурс]. – 2024. – URL: https://docs.python.org/3/library/tkinter.html (дата обращения 01.04.2024)

16. Алексеева О. Г. Методические указания по экономическому обоснованию выпускных квалификационных работ бакалавров: Метод. Указания; О.Г. Алексеева. – СПб.: Изд-во СПбГЭТУ “ЛЭТИ”. – 2013. – 15с.

17. Положение об оплате труда работников СПбГЭТУ «ЛЭТИ» ред. с 15.10.2020.

18. Тариф на интернет [Электронный ресурс]. – 2024. – URL: https://spb.rt.ru/homeinternet/order_internet (дата обращения 07.05.2024)

19. Тарифы на электроэнергию по СПБ. URL: pes.spb.ru. (дата обращения 07.05.2024)

ПРИЛОЖЕНИЕ А
КОД ПРОГРАММЫ

```
import tkinter as tk
from tkinter import ttk, messagebox
from datetime import datetime
from meteostat import Point, Hourly
import pandas as pd
import xgboost as xgb
from sklearn.model_selection import train_test_split,
GridSearchCV
from sklearn.metrics import mean_squared_error,
mean_absolute_error
from sklearn.preprocessing import StandardScaler

def fetch_weather_data():
    try:
        # Преобразование введенных дат в формат datetime
        start = datetime.strptime(start_date.get(), '%Y-%m-%d')
        end = datetime.strptime(end_date.get(), '%Y-%m-%d')

        # Установка локации для Тетуан, Марокко
        location = Point(35.5889, -5.3626)

        # Получение погодных данных
        data = Hourly(location, start, end)
        data = data.fetch()

        if data.empty:
            messagebox.showerror("Ошибка", "Не удалось получить
данные о погоде.")
            return

        # Добавление столбца времени
        data['time'] = data.index

        # Удаление столбцов с отсутствующими значениями
        df_cleaned = data.dropna(axis=1)

        # Сохранение данных в CSV файл
        df_cleaned.to_csv('cleaned_weather_data.csv',
index=False)

        messagebox.showinfo("Успех", "Данные о погоде успешно
получены и сохранены.")
    except Exception as e:
        messagebox.showerror("Ошибка", str(e))

def predict_consumption():
    try:
        # Загрузка данных
```

```
        data = pd.read_csv('powerconsumption.csv')
        new_data = pd.read_csv('cleaned_weather_data.csv')

        # Создание копии new_data для переименования столбцов
        new_data_renamed = new_data.copy()
        new_data_renamed = new_data_renamed.drop(columns =
['dwpt', 'wdir', 'prcp', 'pres', 'coco'], axis = 1)
        new_data_renamed.rename(columns={'temp': 'Temperature',
'rhum': 'Humidity', 'wspd': 'WindSpeed'}, inplace=True)

        # Подготовка данных
        features = data[['Temperature', 'Humidity',
'WindSpeed']]
        targets = data[['PowerConsumption_Zone1',
'PowerConsumption_Zone2', 'PowerConsumption_Zone3']]

        # Разделение на обучающую и тестовую выборки
        X_train, X_test, y_train, y_test =
train_test_split(features, targets, test_size=0.2,
random_state=42)

        # Масштабирование признаков
        scaler = StandardScaler()
        X_train_scaled = scaler.fit_transform(X_train)
        X_test_scaled = scaler.transform(X_test)

        # Обучение моделей XGBoost для каждой зоны отдельно
        models = {}
        predictions = {}

        for zone in targets.columns:
            model =
xgb.XGBRegressor(objective='reg:squarederror', random_state=42)
            model.fit(X_train_scaled, y_train[zone])
            models[zone] = model
            predictions[zone] = model.predict(X_test_scaled)

        # Подготовка новых данных для предсказания
        new_features = new_data_renamed[['Temperature',
'Humidity', 'WindSpeed']]
        new_features_scaled = scaler.transform(new_features)

        new_predictions = {}
        for zone in targets.columns:
            new_predictions[zone] =
models[zone].predict(new_features_scaled)

        # Создание DataFrame с предсказаниями
        new_predictions_df = pd.DataFrame(new_predictions)
        new_predictions_df.columns = ['PowerConsumption_Zone1',
'PowerConsumption_Zone2', 'PowerConsumption_Zone3']

        new_data_with_predictions = pd.concat([new_data_renamed,
```

```
new_predictions_df], axis=1)

        # Сохранение данных с предсказаниями в новый файл

new_data_with_predictions.to_csv('predicted_power_consumption.cs
v', index=False)

        for zone in targets.columns:
            mae = mean_absolute_error(y_test[zone],
predictions[zone])
            mse = mean_squared_error(y_test[zone],
predictions[zone])
            print(f'MAE for {zone}: {mae}', f'MSE for {zone}:
{mse}')


        messagebox.showinfo("Прогнозирование завершено",
"Прогнозы сохранены в файл 'predicted_power_consumption.csv'.")
    except Exception as e:
        messagebox.showerror("Ошибка", str(e))
        print("Произошла ошибка:", e)

# Создание главного окна
root = tk.Tk()
root.title("Прогнозирование потребления электричества")

# Создание фреймов
frame1 = ttk.Frame(root, padding="10")
frame1.grid(row=0, column=0, sticky=(tk.W, tk.E))

frame2 = ttk.Frame(root, padding="10")
frame2.grid(row=1, column=0, sticky=(tk.W, tk.E))

# Виджеты для выбора даты
ttk.Label(frame1, text="Дата начала (ГГГГ-ММ-ДД):").grid(row=0,
column=0)
start_date = ttk.Entry(frame1)
start_date.grid(row=0, column=1)

ttk.Label(frame1, text="Дата окончания (ГГГГ-ММ-
ДД):").grid(row=1, column=0)
end_date = ttk.Entry(frame1)
end_date.grid(row=1, column=1)

# Кнопка для получения данных о погоде
ttk.Button(frame1, text="Получить данные о погоде",
command=fetch_weather_data).grid(row=2, column=1)

# Кнопка для прогнозирования потребления электричества
ttk.Button(frame2, text="Прогнозировать потребление",
command=predict_consumption).grid(row=0, column=1)
# Запуск главного цикла приложения
root.mainloop()
```