Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
(ФГБОУ ВО «АмГУ»)

Институт компьютерных и инженерных наук
Кафедра математического анализа и моделирования
Направление подготовки – 01.03.02 Прикладная математика и информатика
Направленность (профиль) образовательной программы «Прикладная математика и информатика»

ДОПУСТИТЬ К ЗАЩИТЕ
И.о. зав. кафедрой
_____ Н.Н. Максимова
«_____»_____ 2025 г.

БАКАЛАВРСКАЯ РАБОТА

на тему: Модели машинного обучения для прогнозирования стоимости жилья (по набору данных за 2021 год)

Исполнитель
студент группы 1101об _____ Е.А. Фартусова
(подпись, дата)

Руководитель
доцент, канд. физ.-мат. наук _____ Н.Н. Максимова
(подпись, дата)

Нормоконтроль
Ассистент _____ В.О. Салмиянов
(подпись, дата)

Благовещенск 2025

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
(ФГБОУ ВО «АмГУ»)

Факультет математики и информатики
Кафедра математического анализа и моделирования

УТВЕРЖДАЮ
И.о. зав. кафедрой
_____ Н.Н. Максимова
«_____»_____2025 г.

З А Д А Н И Е

К бакалаврской работе студента Фартусовой Елизаветы Алексеевны

1. Тема бакалаврской работы: Модели машинного обучения для прогнозирования стоимости жилья (по набору данных за 2021 год) (утверждена приказом от 20.04.2025 № 951-уч).

2. Срок сдачи студентом законченной работы: 25.06.2025 г.

3. Исходные данные к бакалаврской работе: данные сайта Kaggle, учебные и учебно-методические работы, отчет по производственной практике (научно-исследовательской работе), отчет по преддипломной практике, среда разработки – Python.

4. Содержание бакалаврской работы (перечень подлежащих разработке вопросов): методы машинного обучения, прогнозирование стоимости жилья с помощью модели линейной регрессии, прогнозирование стоимости жилья с помощью многослойной нейронной сети, результаты прогнозирования.

5. Перечень материалов приложения: листинги вычислительных программ, диплом победителя в научной конференции.

6. Консультанты по бакалаврской работе: нормоконтроль – Салмиянов В.О., ассистент.

7. Дата выдачи задания: 14.04.2025 г.

Руководитель бакалаврской работы: Максимова Надежда Николаевна, доцент, канд. физ.-мат. наук.

Задание принял к исполнению (14.04.2025): _____ Фартусова Е.А.

РЕФЕРАТ

Бакалаврская работа содержит 78 с., 20 рисунков, 6 таблиц, 4 приложения, 27 источников.

МАШИННОЕ ОБУЧЕНИЕ, МЕТРИКИ КАЧЕСТВА, PYTHON, ПРОГНОЗИРОВАНИЕ СТОИМОСТИ ЖИЛЬЯ, МОДЕЛЬ ЛИНЕЙНОЙ РЕГРЕСИИ, НЕЙРОННАЯ СЕТЬ.

В данной работе представлено исследование модели машинного обучения для прогнозирования стоимости недвижимости, основанной на методах регрессионного анализа и нейронных сетей, а также проведены вычислительные эксперименты.

Цель работы – разработка и сравнение эффективности моделей машинного обучения (линейной регрессии и нейронной сети) для прогнозирования рыночной стоимости объектов недвижимости на основе набора данных с учетом ключевых параметров.

Основу методологии исследования составляют: теория машинного обучения и регрессионного анализа, Методы предобработки и анализа данных, алгоритмы обучения моделей и оценки их качества, принципы построения нейронных сетей для задач регрессии.

Результатом работы является:

– Реализация модели линейной регрессии в среде Python с использованием библиотек Scikit-learn, Pandas, Matplotlib и NumPy.

– Разработка архитектуры нейронной сети с использованием TensorFlow/Keras.

– Сравнительный анализ эффективности моделей по метрикам качества.

– Интерпретация влияния различных факторов (площадь, местоположение, этажность и др.) на стоимость недвижимости.

– Практические рекомендации по применению моделей для участников рынка недвижимости.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Машинное обучение (МО) представляет собой раздел искусственного интеллекта, ориентированный на разработку алгоритмических моделей, способных извлекать закономерности из данных. В отличие от традиционного программирования, где логика работы системы задаётся явно, в машинном обучении модель самостоятельно формирует правила на основе обучающей выборки. Это позволяет системам совершенствовать свои функциональные возможности с увеличением объёма входных данных и повышает их адаптивность к изменяющимся условиям.

Рынок недвижимости играет ключевую роль в экономической системе государства, являясь важным индикатором её стабильности и развития. Современные реалии характеризуются высокой степенью неопределённости макроэкономической среды: колебаниями процентных ставок по ипотечным кредитам, урбанизационными процессами, миграцией населения и влиянием инфляции. Эти факторы оказывают прямое воздействие на динамику цен на недвижимость, что делает актуальным разработку эффективных методов прогнозирования.

Точная оценка рыночной стоимости объектов недвижимости позволяет участникам рынка – инвесторам, застройщикам и покупателям – принимать более обоснованные управленческие решения, снижать риск ошибочных вложений и повышать общую прозрачность рынка. Традиционные подходы к оценке недвижимости зачастую ограничены в своей аналитической мощности, особенно при работе с большими и многомерными массивами информации.

Использование методов машинного обучения открывает широкие перспективы для автоматизации и повышения точности прогнозов в сфере недвижимости. Современные алгоритмы, такие как градиентный бустинг, случайный лес и нейронные сети, позволяют учитывать сложные нелинейные зависимости между множеством факторов – от локализации объекта до социально-экономических характеристик региона. Это даёт возможность строить более корректные и адап-

тивные модели ценообразования, которые могут применяться как в коммерческих, так и в аналитических целях.

Целью исследования является разработка модели машинного обучения для прогнозирования стоимости недвижимости на основе данных по Российской Федерации за 2021 год. Модель должна учитывать различные признаки, такие как географическое расположение, характеристики объекта, тип здания и другие параметры, чтобы обеспечить высокую точность прогнозирования.

Задачи исследования:

– Сбор и предварительная обработка данных о недвижимости, включая очистку данных, обработку пропусков и кодирование категориальных признаков.

– Анализ данных для выявления ключевых факторов, влияющих на стоимость недвижимости.

– Разработка и обучение модели линейной регрессии для прогнозирования стоимости недвижимости.

– Оценка качества модели с использованием метрик, таких как средняя абсолютная ошибка (MAE) и среднеквадратичная ошибка (MSE).

– Сравнение эффективности линейной регрессии с другими методами машинного обучения (при необходимости).

– Формулирование выводов и рекомендаций по использованию модели на практике.

Научная новизна работы заключается в применении методов машинного обучения для анализа данных о недвижимости в Российской Федерации с учетом географических и категориальных признаков. Особое внимание уделяется исследованию влияния таких факторов, как местоположение объекта, тип здания и этажность, на стоимость недвижимости. Также в работе рассматривается возможность улучшения качества прогнозирования за счет комбинирования различных методов машинного обучения.

Практическая значимость работы заключается в том, что разработанная

модель может быть использована риелторскими компаниями, банками, застройщиками и частными лицами для оценки рыночной стоимости объектов недвижимости. Это позволит повысить точность оценки, минимизировать риски, связанные с переоценкой или недооценкой объектов, и повысить прозрачность сделок на рынке недвижимости. Кроме того, результаты работы могут быть полезны для государственных органов при разработке стратегий развития жилищного строительства и регулирования рынка недвижимости.

Бакалаврская работа состоит из введения, трех основных разделов, заключения, библиографического списка и четырех приложений.

В первой главе рассматриваются основные понятия и определения, история развития машинного обучения, основные методы машинного обучения, формализация задачи регрессии, метрики качества регрессионных моделей.

Во второй главе рассмотрен обзор программных средств (прикладных пакетов) для Machine Learning и Neural Networks, библиотеки языка Python, такие как Pandas, NumPy, Matplotlib, Scikit-learn, Statsmodels, TensorFlow, Keras.

В третьей главе представлены концептуальная и математическая постановки задачи, описание набора данных, предобработка данных, построение модели с помощью функции LinearRegression и оценка качества этой модели, а также нейронная сеть, описание архитектуры и настройка гиперпараметров и оценка качества нейронной сети, обсуждение результатов.

В приложениях приведены листинги вычислительных программ, сертификат участника XXXV научной конференции «День науки».

Результаты исследования докладывались и обсуждались:

– на XXXV научной конференции «День науки» (Амурский государственный университет, г. Благовещенск, 13 марта 2025 года);

По результатам работы опубликованы одни тезисы доклада [27].

1 ВВЕДЕНИЕ В МЕТОДЫ МАШИННОГО ОБУЧЕНИЯ

Современное информационное пространство характеризуется стремительным ростом объема данных, которые могут содержать в себе значимые закономерности, полезные для анализа и прогнозирования. Однако эффективное использование такого рода информации требует применения специализированных методов, способных не только обрабатывать большие массивы данных, но и выявлять скрытые зависимости, а также предсказывать возможные исходы событий. Именно здесь на первый план выходит машинное обучение (Machine Learning), как один из ключевых инструментов аналитики нового поколения.

Основная цель машинного обучения заключается в создании моделей и алгоритмов, которые могут самостоятельно извлекать знания из данных, решать задачи или формировать прогнозы без явного программирования действий. Такие подходы позволяют автоматизировать сложные процессы и принимать более точные решения в различных областях – от бизнеса до науки.

1.1 Основные понятия и определения

Машинное обучение представляет собой область искусственного интеллекта, направленную на разработку систем, способных к самообучению. В отличие от традиционного программирования, где логика работы системы задаётся разработчиком заранее, в машинном обучении модель учится на основе предоставленных ей данных. Задача специалиста — подготовить обучающий набор, определить целевую переменную и выбрать подходящий алгоритм, который сможет найти взаимосвязи и сделать выводы [15, 16].

Примерами таких систем являются нейронные сети, которые имитируют работу человеческого мозга и способны распознавать образы, обрабатывать язык, принимать решения и даже генерировать новые данные. Применяются методы машинного обучения во многих сферах: от создания рекомендательных сервисов и беспилотных автомобилей до медицинской диагностики и финансового анализа. Широко известны такие языковые модели, как ChatGPT, YaGPT и

другие, которые стали доступны широкой аудитории и активно используются как в профессиональной деятельности, так и в повседневной жизни.

1.2 История развития машинного обучения

История машинного обучения начинается с середины XX века, когда возник интерес к возможности создания машин, способных мыслить подобно человеку. Одним из первых шагов в этом направлении стало появление «Тьюринг-теста», предложенного Аланом Тьюрингом в 1950 году. Этот тест был призван определить, может ли машина демонстрировать поведение, неотличимое от человеческого, и стал важной вехой в развитии искусственного интеллекта [2].

В 1952 году Артур Самуэль разработал одну из первых программ, способных к самообучению – шашечную игру, которая улучшала свою производительность через повторяющиеся попытки и ошибки. Это можно считать одним из первых примеров практического применения принципов машинного обучения.

Однако настоящим прорывом стало появление перцептрона, разработанного Фрэнком Розенблаттом в 1957 году. Эта модель стала прообразом современных нейронных сетей и могла выполнять базовые операции классификации, например, распознавание графических символов. Несмотря на ограниченные возможности, перцептрон заложил основу для дальнейшего развития теории искусственных нейросетей.

В 1960-е годы были созданы базовые алгоритмы, многие из которых актуальны и сегодня – это, например, метод ближайших соседей и линейная регрессия. Однако после выхода книги Марвина Минского и Сеймура Пейперта «Perceptrons» в 1969 году, в которой были указаны ограничения однослойных нейронных сетей, интерес к этой области временно снизился.

Восстановление интереса к нейронным сетям произошло в 1980-е годы благодаря появлению алгоритма обратного распространения ошибки. Он позволил эффективно обучать многослойные нейронные сети, что открыло возможность их использования для решения сложных задач, таких как распознавание речи и изображений.

1990-е годы стали периодом становления статистических методов машинного обучения. Были разработаны такие мощные алгоритмы, как метод опорных векторов (SVM) и случайный лес, которые получили широкое применение благодаря своей устойчивости и высокой точности.

На рубеже веков началось активное внедрение машинного обучения в такие области, как компьютерное зрение, биоинформатика и обработка естественного языка. В 2006 году Джеффри Хинтон и его коллеги представили концепцию глубокого обучения, которая позволила строить многослойные нейронные сети с высокой производительностью. Глубокое обучение стало настоящей революцией в области распознавания изображений и голоса.

Переломным моментом стало выступление команды из Университета Торонто на конкурсе ImageNet в 2012 году, где они показали превосходство глубоких нейронных сетей над традиционными методами. Этот успех дал мощный толчок дальнейшим исследованиям и популяризации методов глубокого обучения [3].

Сегодня машинное обучение активно используется в различных отраслях:

Медицина: применение ML-моделей позволяет анализировать изображения, выявлять патологии на ранних стадиях и разрабатывать индивидуальные схемы лечения.

Финансы: алгоритмы помогают прогнозировать рыночные тренды, оценивать кредитные риски, автоматизировать торговлю и обнаруживать мошенничество.

Транспорт: машинное обучение применяется при создании автономных транспортных средств, планировании маршрутов и анализе дорожных ситуаций.

Современные исследования сосредоточены на создании интерпретируемых моделей, снижении энергопотребления и разработке методов, способных работать с малым объемом данных. Эти направления особенно важны в тех случаях, когда сбор информации затруднён или связан с высокими затратами [13].

Будущее машинного обучения связано с созданием более справедливых, понятных и устойчивых систем, которые будут играть всё большую роль в нашей

жизни. Его развитие зависит от качества данных, корректного выбора методов и постоянного совершенствования алгоритмов.

1.3 Основные виды машинного обучения

Машинное обучение охватывает широкий спектр подходов, которые могут быть условно разделены на несколько ключевых категорий: классические методы, обучение с подключением (semi-supervised learning), ансамблевые модели, нейронные сети и глубокое обучение. Каждый из этих подходов имеет свою область применения, зависящую от характера данных и поставленных задач.

Так, классические методы машинного обучения наиболее эффективны при работе с простыми наборами данных, где признаки хорошо определены и интерпретируемы. Обучение с подключением используется в условиях ограниченной разметки, когда модель может частично обучаться на неструктурированных данных. Ансамбли применяются тогда, когда требуется повысить качество прогнозирования за счёт комбинирования нескольких моделей. Наконец, нейронные сети и глубокое обучение становятся основным инструментом при работе с большими и сложными данными, когда заранее сложно выделить значимые признаки [6].

1.3.1 Основные типы задач машинного обучения

В зависимости от целей анализа выделяют следующие категории задач:

– Регрессия – заключается в прогнозировании количественного значения на основе входных факторов. Примеры: предсказание стоимости недвижимости, оценка доходности предприятия, определение качества продукта на основе характеристик сырья.

– Классификация – направлена на отнесение объекта к одному из заданных классов. Задачи такого типа встречаются в распознавании образов (например, определение наличия лица на изображении), медицинской диагностике (выявление заболевания) или анализе текста (определение тематики документа).

– Кластеризация – представляет собой процесс группировки объектов по схожести их характеристик без предварительной разметки. Применяется для сегментации клиентов, анализа астрономических данных, выявления закономерностей в больших массивах информации.

– Уменьшение размерности – позволяет упростить данные, сохранив при этом их информативность. Используется при подготовке данных к последующему анализу и визуализации, особенно в случае высокой размерности исходных признаков.

– Обнаружение аномалий – направлено на выявление редких событий или выбросов, которые отличаются от нормального поведения системы. Этот тип задач важен при обнаружении мошеннических транзакций, технических сбоев или нестандартных ситуаций в реальном времени. Отличительной особенностью является отсутствие достаточного числа примеров аномалий, что делает невозможным применение стандартных методов классификации.

1.3.2 Методы машинного обучения

Существует несколько парадигм обучения моделей, отличающихся степенью участия специалиста в процессе подготовки данных и обучения системы [23, 24]:

1. Обучение с учителем (Supervised Learning)

Данный подход предполагает наличие размеченных данных, то есть таких, где каждому входному объекту соответствует известный результат. Модель обучается на этих примерах, после чего может делать выводы на новых данных. Применяется для решения задач регрессии и классификации.

Примеры алгоритмов:

– Наивный Байес.

– Метод опорных векторов (SVM).

– Деревья решений.

– k-ближайших соседей (kNN).

– Линейная и логистическая регрессия.

Использование:

– Фильтрация спама.

– Компьютерное зрение.

– Анализ документов.

2. Обучение без учителя (Unsupervised Learning)

В данном случае модель работает с данными, не имеющими меток. Её задача – найти внутреннюю структуру или скрытые паттерны в информации. Часто используется при анализе больших объёмов данных, где ручная разметка невозможна.

Основные задачи: кластеризация, выявление аномалий.

3. Полуобучение (Semi-supervised Learning)

Этот подход занимает промежуточное положение между supervised и unsupervised learning. При этом часть данных размечена, а другая – нет. Такой метод полезен, когда получение размеченных данных требует значительных затрат времени и ресурсов, например, при работе с изображениями или звукозаписями.

4. Обучение с подкреплением (Reinforcement Learning)

В отличие от других методов, здесь модель взаимодействует со средой и получает обратную связь в виде награды или штрафа за свои действия. Это позволяет ей формировать стратегию поведения, которая максимизирует суммарную награду.

5. Глубокое обучение (Deep Learning)

Глубокое обучение – это подраздел машинного обучения, основанный на использовании многослойных нейронных сетей. Оно позволяет автоматически извлекать признаки из исходных данных, что делает его особенно эффективным при работе с изображениями, аудио и текстами.

Глубокие модели активно применяются в компьютерном зрении, обработке естественного языка, медицинской диагностике и других сложных задачах, где традиционные методы оказываются недостаточно мощными.

1.4 Формализация задачи регрессии

Регрессия в машинном обучении представляет собой задачу прогнозирования числовых значений на основе имеющихся данных. Основной целью регрессионного анализа является построение математической зависимости, отражающей связь между входными (независимыми) и выходной (зависимой) переменными [4, 5].

Этот метод широко применяется для оценки степени влияния определённых факторов на конечный результат. Например, с помощью регрессионного анализа можно изучить, как возраст, уровень дохода и образование потребителя влияют на вероятность приобретения автомобиля.

Регрессионные модели делятся на линейные и нелинейные. Линейные модели предполагают наличие прямой пропорциональной связи между переменными и чаще всего строятся на основе уравнения прямой линии. Нелинейные модели, напротив, позволяют описывать более сложные, неоднородные взаимосвязи, которые не могут быть корректно отражены с помощью простой линейной функции.

Формальная постановка:

Задача регрессии в машинном обучении может быть сформулирована следующим образом:

Дано:

Набор данных, состоящий из пар входных значений (фичей) и соответствующих им целевых значений (ответов).

В задаче регрессии предполагается, что существует неизвестная целевая зависимость $y: X \to Y$, чьи значения известны только на объектах обучающей выборки $XY = \left\{(x^{(1)}, y_1), (x^{(2)}, y_2), ..., (x^{(n)}, y_n)\right\}$, $x^{(i)} \in X \subset R^m$, $y_i \in Y \subset R$. Необходимо получить алгоритм *a*: $X \to Y$, приближающий целевую зависимость как на множестве *XY*, так и на *X*. То есть решить задачу регрессии – значит найти алгоритм, обладающий способностью к обобщению эмпирических фактов (способностью к выводу общих знаний из частных наблюдений, прецедентов).

Подбор зависимости в виде семейства параметрических функций в машинном обучении заключается в том, чтобы представить функционал в виде выбранной параметрической функции с настраиваемым набором параметров.

Вид функции может быть сколь угодно сложным и в общем случае состоять из композиции других, более простых функций. То есть вид функции должен отражать характер изменения данных между входом и выходом, а параметры подгоняют её под конкретный набор данных.

Для нахождения подходящих параметров вводится функция потерь, которая вычисляет меру потерь (несоответствия) между моделью и обучающей выборкой. Затем определяется средний эмпирический риск – показатель качества, который нужно минимизировать, подбирая значения вектора параметров.

Функции потерь являются важнейшим компонентом машинного обучения, поскольку они количественно оценивают ошибку между прогнозами модели и фактическими целевыми значениями. Существует несколько типов функций потерь, которые можно условно разделить на две группы: функции потерь при регрессии и функции потерь при классификации. Так как нас интересует только регрессия, рассмотрим функции потерь при регрессии.

Функции потерь при регрессии используются, когда целевая переменная является непрерывной. Некоторые распространённые функции потерь при регрессии:

Среднеквадратическая ошибка (MSE): измеряет среднюю квадратичную разницу между прогнозируемыми и фактическими значениями и вычисляется по формуле

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - y_i\right)^2,$$

здесь y_i – точное значение таргетной переменной, y_i – выход модели для таргетной переменной.

Преимущества:

– Подходит для задач регрессии, где конечной целью является минимизация квадратичных разностей между прогнозируемыми и фактическими значениями.

– Дифференцируемый и выпуклый.

Недостатки:

– Чувствительность к выбросам из-за операции возведения в квадрат, которая отклоняет результаты в процессе оптимизации.

– Не подходит для задач, где ошибки должны наказываться по-другому.

Средняя абсолютная ошибка (MAE): измеряет среднюю абсолютную разницу между прогнозируемыми и фактическими значениями и вычисляется по формуле

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - y_i\right|.$$

Преимущество:

– MAE более устойчив к выбросам по сравнению со средней квадратичной ошибкой (MSE), поскольку учитывает абсолютную разницу, снижая влияние чрезвычайно больших ошибок.

– Потери MAE легко интерпретировать, поскольку они представляют собой среднюю величину ошибок, что упрощает информирование заинтересованных сторон о результатах работы модели.

Недостаток:

– MAE обрабатывает все ошибки одинаково, независимо от их величины. Это может быть недостатком в случаях, когда важно различать мелкие и крупные ошибки.

Регрессионный анализ широко используется в различных областях, таких как финансы, маркетинг, медицина и др. Он позволяет определить влияние различных факторов на зависимую переменную и сделать прогнозы на основе этих факторов.

Линейная регрессия представляет собой зависимость одной величины от другой величины. Самая простая модель линейной регрессии имеет следующие допущения:

1. Все значения зависимой переменной определяются без ошибки.

2. У модели задано конечное число параметров.

3. Ошибка распределения стремится к нулю и имеет постоянное отклонение.

4. Значения параметров заранее не известны, но они подбираются по обучающей выборке.

Достоинства линейной регрессии:

– Скорость и простота получения модели.

– Интерпретируемость модели.

– Линейная модель является прозрачной и понятной для аналитика.

– По полученным коэффициентам регрессии можно судить о том, как тот или иной фактор влияет на результат, сделать на этой основе дополнительные полезные выводы.

– Широкая применимость.

Недостатки:

– В случае нелинейных данных полиномиальную регрессию трудно спроектировать.

– Необходимо иметь информацию о структуре данных и взаимосвязи между переменными.

Основываясь на изложенных выше фактах, линейная регрессия неэффективна, когда речь идёт об очень сложных данных и больших объёмах

Если данные подбираются экспериментально, то чаще всего применяются программные продукты. Они предназначены специально для обработки статистических данных. Но существуют специальные формулы для расчета параметров, поэтому есть возможность рассчитать их вручную.

1.5 Метрики качества регрессионных моделей

Чтобы оценить, насколько повышение сложности модели значимо увеличивает её точность, необходимо использовать аппарат оценки качества регрессионных моделей. Он включает в себя следующие меры:

– Среднеквадратичная ошибка (MSE).
– Корень из среднеквадратичной ошибки ($RMSE$).
– Среднеквадратичная ошибка в процентах ($MSPE$).
– Средняя абсолютная ошибка (MAE).
– Средняя абсолютная ошибка в процентах ($MAPE$).
– Средняя абсолютная масштабированная ошибка ($MASE$)
– Коэффициент детерминации R^2-квадрат.

Среднеквадратичная ошибка (Mean Squared Error) применяется в случаях, когда требуется подчеркнуть большие ошибки и выбрать модель, которая дает меньше именно больших ошибок. Большие значения ошибок становятся заметнее за счет квадратичной зависимости. Среднеквадратичная ошибка вычисляется следующим образом:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - y_i\right)^2 .$$

Корень из среднеквадратичной ошибки (Root Mean Squared Error) вычисляется просто как квадратный корень из MSE.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - y_i\right)^2} .$$

Среднеквадратичная ошибка в процентах (Mean Squared Percentage Error) представляет собой относительную ошибку, где разность между наблюдаемым и фактическим значениями делится на наблюдаемое значение и выражается в процентах.

$$MSPE = \frac{100}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{y_i}\right)^2 .$$

Средняя абсолютная ошибка (Mean Absolute Error) вычисляется следующим образом:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|.$$

MAE рассчитывается как среднее абсолютных разностей между наблюдаемым и предсказанным значениями.

Средняя абсолютная процентная ошибка (Mean Absolute Percentage Error) вычисляется следующим образом:

$$MAPE = \frac{100}{n}\sum_{i=1}^{n}\frac{|y_i - \hat{y}_i|}{|y_i|}.$$

Эта ошибка не имеет размерности и очень проста в интерпретации. Её можно выражать как в долях, так и в процентах.

Средняя абсолютная масштабированная ошибка (Mean Absolute Scaled Error) – это показатель, который позволяет сравнивать две модели. Если поместить MAE для новой модели в числитель, а MAE для исходной модели в знаменатель, то полученное отношение и будет равно $MASE$. Если значение $MASE$ меньше 1, то новая модель работает лучше, если $MASE$ равно 1, то модели работают одинаково, а если значение $MASE$ больше 1, то исходная модель работает лучше, чем новая модель. Формула для расчета $MASE$ имеет вид:

$$MASE = \frac{MAE_i}{MAE_j}.$$

Так же R^2 или же как его называют по другому коэффициент детерминации (Coefficient of determination), который показывает долю дисперсии зависимой переменной, объяснённой с помощью регрессионной модели. Наиболее общей формулой для вычисления коэффициента детерминации является следующая:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \overline{y}_i)^2}.$$

Коэффициент детерминации (R^2) является одной из ключевых метрик оценки качества регрессионных моделей. Формально он определяется как отношение объяснённой дисперсии к общей дисперсии и рассчитывается по формуле,

где в числителе указана среднеквадратичная ошибка модели, а в знаменателе – ошибка базовой модели, учитывающей только константное значение.

Одним из главных достоинств R^2 перед другими метриками, основанными на абсолютных значениях ошибок, является его независимость от масштаба целевой переменной. Это позволяет сравнивать модели, обученные на разных наборах данных или с различными диапазонами значений зависимой переменной. Диапазон изменения коэффициента теоретически не ограничен снизу и лежит в пределах от минус бесконечности до единицы. При этом значения, близкие к 1, свидетельствуют о высоком уровне соответствия модели реальным данным. Такое происходит тогда, когда отношение ошибок стремится к нулю, то есть модель с предикторами значительно превосходит простую константную модель по точности прогноза.

Если значение R^2 равно 0, это говорит о том, что добавление независимых переменных в модель не улучшило качество предсказания относительно среднего значения. Иными словами, между входными и выходной переменными отсутствует статистическая связь, или она выражена крайне слабо.

Снижение коэффициента до значений, близких к нулю, интерпретируется как низкая эффективность модели. В этом случае модель с предикторами демонстрирует результаты, не отличающиеся от модели, которая всегда предсказывает среднее значение.

Интересным случаем является появление отрицательных значений R^2. Это возможно, если ошибка модели с предикторами превышает ошибку константной модели. Отрицательный коэффициент указывает на то, что включение переменных в модель ухудшило её способность к предсказанию, и, следовательно, модель требует пересмотра – например, необходимо исключить нерелевантные признаки или изменить тип используемой регрессионной зависимости.

На практике часто применяют следующие эмпирические критерии оценки:

– Значение R^2 выше 0.5 считается удовлетворительным, что говорит о приемлемом соответствии модели данным.

– Если коэффициент превышает 0.8 , модель признаётся качественной и может быть рекомендована к использованию в аналитических или прогнозных задачах.

– Показатель ниже 0.5 свидетельствует о неудовлетворительном качестве модели, что делает её использование малоприемлемым без дополнительной доработки.

Таким образом, коэффициент детерминации предоставляет удобную и информативную меру для сравнения различных моделей и позволяет сделать вывод о практической ценности построенной регрессионной зависимости.

2 ПРОГРАММНЫЕ СРЕДСТВА МАШИННОГО ОБУЧЕНИЯ

2.1 Обзор программных средств (прикладных пакетов) для Machine Learning и Neural Networks

При решении задач прогнозирования стоимости недвижимости доступен широкий спектр программных средств, различающихся по функциональности, уровню сложности и сфере применения. Ниже приведён обзор наиболее популярных и эффективных инструментов, начиная от простых решений и заканчивая профессиональными аналитическими платформами.

На начальных этапах работы с данными, особенно при проведении предварительного анализа и построении базовых моделей прогноза, часто применяются табличные процессоры, такие как Microsoft Excel. Они позволяют выполнять элементарный статистический анализ, строить графики, рассчитывать тренды и использовать регрессионные методы. Несмотря на удобство и доступность, возможности Excel ограничены при работе с большими объемами информации и сложными моделями машинного обучения.

Для более детального исследования данных могут использоваться специализированные пакеты статистического анализа. Например, IBM SPSS Statistics сочетает в себе понятный интерфейс и богатый набор инструментов: множественная регрессия, кластеризация, факторный анализ – всё это делает его востребованным среди исследователей. Stata активно используется в научной среде благодаря продвинутым возможностям работы с панельными данными и эконометрическому функционалу. SAS – мощное решение для корпоративной аналитики, поддерживающее обработку больших массивов и сложные статистические вычисления. Statistica также заслуживает внимания за счет интуитивно понятного интерфейса и разнообразия встроенных алгоритмов.

Однако для реализации гибких и масштабируемых моделей машинного обучения и нейронных сетей чаще всего используются языки программирования. Python считается одним из самых популярных инструментов в этой области. С его библиотеками, такими как Pandas (для работы с данными), Scikit-learn (для

построения моделей машинного обучения), Statsmodels (для статистического моделирования) и TensorFlow/Keras (для работы с глубоким обучением), можно создавать сложные прогнозные системы.

Не менее значимую роль играет язык R, который был изначально разработан для статистических вычислений. Он предлагает огромное количество пакетов для анализа данных, включая обработку временных рядов и пространственной информации, что особенно важно при оценке недвижимости [1].

Ещё одним мощным инструментом является MATLAB – вычислительная среда, ориентированная на математическое моделирование и численные методы. Благодаря своей матричной архитектуре и встроенным функциям он широко используется в инженерных и научных расчётах. В MATLAB реализованы средства для статистического анализа, обработки сигналов, решения систем уравнений и оптимизации. Также доступны инструменты для построения графиков и создания пользовательских интерфейсов. Для задач машинного обучения предусмотрен специальный пакет – Statistics and Machine Learning Toolbox, включающий алгоритмы регрессии, классификации и кластеризации.

С развитием облачных технологий появились современные платформы, предоставляющие возможность удалённой разработки и развертывания моделей. Такие сервисы, как Google Colab, Microsoft Azure ML и Amazon SageMaker, предлагают высокопроизводительные ресурсы и готовые окружения для работы с данными. Кроме того, AutoML-платформы, например H2O.ai и DataRobot, автоматизируют выбор и настройку моделей, что позволяет значительно сократить время на получение результатов.

2.2 Библиотеки языка Python

2.2.1 Pandas, NumPy, Seaborn, Matplotlib

Pandas – это мощная библиотека, предназначенная для анализа и обработки структурированных данных. Её основным объектом является DataFrame – аналог таблицы, который позволяет выполнять операции фильтрации, группировки, агрегирования и очистки данных. Pandas поддерживает загрузку и сохранение информации в различных форматах (CSV, Excel, SQL), что делает её незаменимой

на этапе предварительной подготовки данных перед построением моделей машинного обучения [9, 21].

NumPy служит базовой библиотекой для численных вычислений в Python. Она предоставляет удобные средства работы с многомерными массивами и математическими функциями. Основное преимущество NumPy заключается в высокой производительности: реализация части кода на языках C и C++ позволяет достичь скорости, сравнимой с компилируемыми языками. В отличие от стандартных структур Python, массивы NumPy состоят из элементов одного типа, что ускоряет вычисления и снижает риск ошибок при обработке данных.

Matplotlib – одна из первых и наиболее популярных библиотек для построения графиков в Python. С её помощью можно создавать широкий спектр диаграмм и визуализаций: линейные графики, гистограммы, круговые диаграммы, трёхмерные поверхности и другие. Библиотека поддерживает два подхода к работе с графиками: процедурный через модуль pyplot и объектно-ориентированный, где каждая часть графика представлена отдельным объектом. Matplotlib часто используется как основа для других инструментов визуализации, таких как Seaborn.

Seaborn строится на основе Matplotlib и предоставляет высокоуровневый интерфейс для создания статистических графиков. Эта библиотека оптимизирована для работы с табличными данными, напрямую поддерживая объекты Pandas. Seaborn предлагает широкие возможности настройки внешнего вида графиков, включая цветовые палитры, стили и аннотации. Также она автоматически выполняет агрегирование данных, что упрощает построение сложных диаграмм и исследование зависимостей между переменными.

2.2.2 Scikit-learn

Scikit-learn – одна из самых известных библиотек машинного обучения на Python. Разработка началась ещё в 2007 году, и с тех пор она стала стандартом в области Data Science благодаря своей простоте, универсальности и богатому набору алгоритмов.

К основным преимуществам библиотеки относятся:

– Широкий выбор методов: включая регрессию, классификацию, кластеризацию, понижение размерности и оценку модели.

– Интеграция с другими библиотеками: легко сочетается с NumPy, Pandas и Matplotlib, что обеспечивает полный цикл обработки, анализа и моделирования данных.

– Простота использования: понятный API и документация позволяют быстро освоить инструмент даже новичкам.

– Открытость и доступность: Scikit-learn распространяется бесплатно под лицензией BSD, что допускает использование в коммерческих проектах.

– Платформенная совместимость: работает на всех популярных ОС (Windows, Linux, macOS) и поддерживает различные среды разработки.

– Активное сообщество: наличие учебных материалов, примеров и поддержки пользователей делает библиотеку особенно популярной среди исследователей и разработчиков.

Однако у Scikit-learn есть и ограничения. Например, он не предусматривает развитых возможностей для построения глубоких нейронных сетей, а также требует хорошего понимания математических принципов машинного обучения.

2.2.3 Statsmodels

Библиотека Statsmodels предназначена для выполнения статистического анализа и проверки гипотез. Она дополняет возможности Python в тех задачах, где требуется строгое статистическое обоснование моделей [18].

Statsmodels объединяет в себе мощные вычислительные функции NumPy и SciPy, средства визуализации Matplotlib и удобство работы с данными из Pandas. Благодаря этому библиотека активно используется в эконометрике, финансовых исследованиях и научном анализе.

Основные возможности:

– Построение регрессионных моделей (в том числе линейных, логистических и обобщённых).

– Проверка статистических гипотез.

– Вычисление корреляций.

– Анализ временных рядов.

– Поддержка одномерных и двумерных моделей.

Statsmodels особенно ценна для специалистов, имеющих опыт работы с R, так как воспроизводит многие из его статистических методов. При этом она остаётся полностью интегрированной в экосистему Python, что делает её удобной для комплексного анализа данных.

2.2.4 TensorFlow и Keras

TensorFlow – это мощная библиотека машинного и глубокого обучения, разработанная компанией Google. Она позволяет строить сложные модели, включая нейронные сети, для решения задач распознавания образов, обработки естественного языка, прогнозирования и других.

Работа с TensorFlow осуществляется через вычислительные графы – структуры, представляющие собой последовательность операций над данными. Это даёт возможность гибкой настройки моделей и их оптимизации [19,20].

Преимущества:

– Высокая степень абстракции, позволяющая сосредоточиться на логике модели, а не на технической реализации.

– Гибкость: поддержка CPU и GPU, работа в облаке и на мобильных устройствах.

– Кроссплатформенность: поддерживает Windows, Linux, macOS, Android и iOS.

– Интерактивная разработка: возможность тестировать отдельные части модели.

Недостатки:

– Сложность освоения для новичков.

– Высокое потребление памяти при использовании GPU.

– Зависимость от внутренних стандартов Google, усложняющих отладку кода.

Keras – это высокоуровневая библиотека для построения моделей глубокого обучения, которая использует в качестве бэкенда такие фреймворки, как

TensorFlow, Theano или CNTK. Keras создаёт удобный интерфейс для быстрого прототипирования и обучения нейронных сетей [10,22].

Основные достоинства:

– Простота и интуитивность API.

– Модульность: модель собирается из готовых блоков (слоёв, функций активации, оптимизаторов).

– Расширяемость: позволяет добавлять собственные модули.

– Широкое сообщество и хорошая документация.

К недостаткам можно отнести:

– Ограниченную самостоятельность – Keras не может работать без стороннего бэкенда.

– Отсутствие обратной совместимости между версиями.

– Невысокую универсальность для нетиповых задач.

3 ПРОГНОЗИРОВАНИЕ СТОИМОСТИ ЖИЛЬЯ НА НАБОРЕ ДАННЫХ

3.1 Концептуальная постановка задачи

Задача: Прогнозирование стоимости недвижимости на основе данных по Российской Федерации за 2021 год.

Цель работы: Разработка модели машинного обучения, способной предсказывать стоимость недвижимости на основе набора признаков, включающих как количественные, так и категориальные характеристики объектов недвижимости.

Данные: В качестве исходных данных используются сведения о продажах недвижимости на территории Российской Федерации за 2021 год.

Модель: В качестве базовой модели для прогнозирования стоимости недвижимости предлагается использовать линейную регрессию. Данный метод выбран благодаря его интерпретируемости, простоте реализации и возможности эффективно работать с числовыми признаками. В случае необходимости, модель может быть усложнена или заменена на более сложные алгоритмы, такие как деревья решений, ансамбли моделей или нейронные сети.

Оценка качества модели:

Для оценки качества модели будут использоваться следующие метрики:

Средняя абсолютная ошибка (MAE) – среднее значение абсолютных отклонений предсказанных значений от фактических;

Среднеквадратичная ошибка (MSE) – среднее значение квадратов отклонений предсказанных значений от фактических.

Коэффициент детерминации (R^2) – доля дисперсии зависимой переменной.

Эти метрики позволят оценить точность модели и сравнить её с альтернативными подходами.

3.2 Математическая постановка задачи

Входные данные: $X = \left\{ (x^{(1)}, x^{(2)}, ..., x^{(n)} \right\}$, $x^{(i)}$ – i-й объект выборки.

Целевая переменная: Стоимость недвижимости *y*.

Оценки качества модели: $MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$,

где n – количество наблюдений, y_i – фактическое значение целевой переменной.

Оценка качества модели: $MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$,

где n – количество наблюдений, y_i – фактическое значение целевой переменной.

3.3 Описание набора данных

Объявления о недвижимости в России публикуются на сайтах avito.ru, realty.yandex.ru, cian.ru, sob.ru, youla.ru, n1.ru, moyareklama.ru. Сервис ads-api.ru позволяет загружать объявления о недвижимости за плату. Парсер сервиса работает странно и дублирует объявления о недвижимости в базе, если авторы продлили их через какое-то время. Также на российском рынке много перекупов (плохих риелторов), которые воруют объявления и публикуют их от своего

Российские сервисы недвижимости позволяют авторам объявлений вручную писать данные о квартире или доме. Поэтому часто бывает так, что пользователь может опубликовать объявление с ошибками или опечатками. Также пользователь может не знать, например, тип стен у своего дома.

В качестве датасета для построения модели был выбран набор Russian Real Estate 2021 с платформы Kaggle [25]. Все переменные и их описание представлено в таблице 1.

Таблица 1 – Описание параметров модели

Переменная	Тип данных	Описание	Значимость
date	object – текстовая переменная	Дата	Не значимо
price	int64 – целое число	Стоимость жилья	Целевая переменная
level	int64 – целое число	Этаж, на которой расположена квартира	Значимо

levels	int64 – целое число	Этажность дома	Значимо
rooms	int64 – целое число	Количество комнат	Значимо
area	float64 – число с плавающей точкой	Площадь квартиры	Значимо
kitchen_area	float64 – число с плавающей точкой	Площадь кухни	Значимо
geo_lat	float64 – число с плавающей точкой	Широта	Значимо
geo_lon	float64 – число с плавающей точкой	Долгота	Значимо
building_type	int64 – целое число	Тип фасада	Значимо
object_type	int64 – целое число	Тип объекта	Значимо
postal_code	int64 – целое число	Почтовый код	Не значимо
street_id	int64 – целое число	Код улицы	Не значимо
id_region	int64 – целое число	Код региона	Значимо
house_id	int64 – целое число	Номер дома	Не значимо

Рынок недвижимости в России бывает двух типов, в наборе данных он используется как тип объекта 0 – Вторичный рынок недвижимости; 2 – Новостройка.

Также есть номер региона России. Например, номер Амурской области – 28. Дополнительно есть номер дома, который синхронизируется через федеральную публичную базу данных Федеральной налоговой службы «ФИАС». Поскольку данные получены через платный сторонний сервис, я не могу публиковать результаты, однако могу их анонимизировать и публиковать такие параметры, как Street ID и House ID. В основном все дома построены из блоков, таких как кирпич, дерево, панель и другие. Тип здания обозначены цифрами: 0 – Не знаю, 1 – Другое, 2 – панельное, 3 – монолитное, 4 – кирпичное, 5 – блочное, 6 – деревянное.

Количество комнат может быть 1, 2 или больше. Однако есть тип квартиры, который называется квартира-студия. Обозначено как «-1».

3.4 Предобработка данных

Предобработка данных представляет собой этап подготовки информационного набора перед его использованием в модели машинного обучения. В исходном виде данные часто содержат различные артефакты — такие как шум,

пропущенные значения или дубликаты, которые затрудняют их анализ и снижают эффективность применяемых алгоритмов [7, 8].

Причины появления подобных артефактов могут быть различными:

– Человеческий фактор – ошибки при ручном вводе информации, включая опечатки, пропуски значений или некорректные данные.

– Недостаток информации – отсутствие определённых полей, например, неуказанный адрес проживания в анкете.

– Сбои систем сбора данных – технические проблемы, такие как кратковременное отключение сети, что может привести к потере части информации.

– Объединение источников – разница в форматах или структуре данных при интеграции нескольких баз данных.

– Технические ограничения – например, система принимает только положительные числа, но при этом получает отрицательное значение.

– Устаревшие данные – информация не обновляется своевременно и перестаёт отражать текущую ситуацию.

– Ошибка миграции – повреждение или потеря данных при переносе из одной системы в другую.

Автоматизация процесса предобработки позволяет ускорить выполнение типовых задач и минимизировать вероятность ошибок. Для этих целей могут использоваться следующие подходы и программные средства:

– Библиотеки для предобработки, такие как Scikit-Learn, предоставляют готовые функции: SimpleImputer – для заполнения пропусков, StandardScaler – для нормализации признаков, LabelEncoder – для кодирования категориальных переменных.

– Scikit-Learn также позволяет объединить несколько этапов обработки в единую цепочку (pipeline), что делает повторное применение одних и тех же действий к новым данным более удобным и структурированным.

– Pandas – мощный инструмент для работы с табличными данными, который может использоваться как для ручной, так и для автоматизированной обра-

ботки. Например, можно создать собственные функции для автоматического заполнения пропусков, масштабирования признаков и кодирования категориальных данных.

– Feature-engine – библиотека, позволяющая автоматизировать операции по обработке признаков, включая создание новых переменных, кодирование категорий и работу с выбросами.

3.4.1 Этапы проведения предобработки

Подготовка данных является ключевым шагом в подготовке выборки для последующего обучения моделей машинного обучения. На каждом этапе возможна как ручная корректировка, так и использование автоматизированных методов, либо их комбинация.

Для примера возьмём набор данных о стоимости жилья за 2021 год. При его обработке были задействованы следующие инструменты:

Pandas – используется для работы с таблицами (DataFrame), очистки, фильтрации, удаления дубликатов и заполнения пропусков.

Scikit-Learn – предоставляет широкий спектр инструментов для предварительной обработки: заполнение пропущенных значений, масштабирование признаков, работа с выбросами.

Matplotlib и Seaborn – применяются для визуализации данных, в том числе для выявления аномалий и выбросов.

NumPy – служит для быстрой работы с числовыми массивами и выполнения сложных математических преобразований.

3.4.2 Этап 1 – Сбор и первичный анализ данных

Первым этапом является сбор данных и анализ их качества, поскольку именно от этого зависит успешность дальнейшей модели. На данном этапе можно понять структуру набора, выявить существующие проблемы (пропуски, дубликаты, выбросы) и наметить пути их устранения.

Данные по недвижимости были загружены с платформы Kaggle через API. Для этого был установлен пакет kaggle, настроена авторизация и скачан архив в формате ZIP, который затем был распакован для последующей обработки.

После загрузки проводится анализ структуры данных с целью выявления проблем, требующих предобработки. Для этой задачи могут быть использованы следующие функции из библиотеки Pandas [14,17]:

– head() — вывод первых строк таблицы для ознакомления с данными;

– info() — отображение количества непустых значений и типа данных в каждом столбце;

– describe() — предоставление сводной статистики: среднее, стандартное отклонение, минимум и максимум для числовых признаков; количество уникальных значений и частота наиболее распространённой категории для категориальных.

На основании анализа были выявлены следующие проблемы:

– Пропущенные значения в столбцах: «price», «rooms», «area», «level», «levels», «kitchen_area», «id_region», «geo_lat», «geo_lon», «building_type», «object_type».

– Некорректные данные: в таких столбцах, как «price», «area», «level», «levels», «kitchen_area», «id_region», «building_type» и «object_type», встречались отрицательные значения, что логически невозможно.

– Выбросы: например, площадь квартиры в 300 кв. м при наличии одной комнаты, а также цена в размере 43 млн рублей, которая значительно превышает средние показатели.

3.4.3 Этап 2 – Очистка данных

После анализа следует этап очистки, направленный на устранение найденных проблем. Его цель – повысить качество набора данных, тем самым улучшая точность прогноза модели.

Очистка может выполняться вручную. Для удаления дубликатов или некорректных записей применяются методы из Pandas, а для работы с отсутствующими значениями – функции из NumPy.

1. Автоматизация процесса очистки

Для автоматической обработки пропущенных данных можно использовать библиотеку Scikit-Learn, например, класс SimpleImputer. Если некоторые значения легче обрабатывать как пропуски, то их предварительно заменяют на NaN.

Пропуски могут возникнуть по различным причинам: отсутствие информации, сбой сбора данных и другие.

2. Удаление строк или столбцов с пропусками

Если пропуски встречаются редко и не несут критически важной информации, соответствующие строки можно просто удалить. Если же пропуски затрагивают значительную часть одного столбца, его имеет смысл исключить полностью.

В нашем случае в наборе данных было обнаружено большое количество пропусков в столбцах postal_code, street_id, house_id.

Кроме того, была проведена проверка на наличие невозможных значений, например, количества комнат, равного нулю. Результат показал, что таких записей в данных нет. На рисунке 1 представлено количество строк после завершения этапа очистки.

3. Обработка выбросов и шума

Выбросы представляют собой значения, сильно отличающиеся от большинства данных. Они могут быть вызваны как ошибками ввода, так и редкими, но реальными событиями.

Шум – это случайные отклонения значений, не несущие полезной информации. Возникает вследствие ошибок измерения, технических сбоев, влияния внешних условий или просто случайных факторов, не связанных с решаемой задачей.

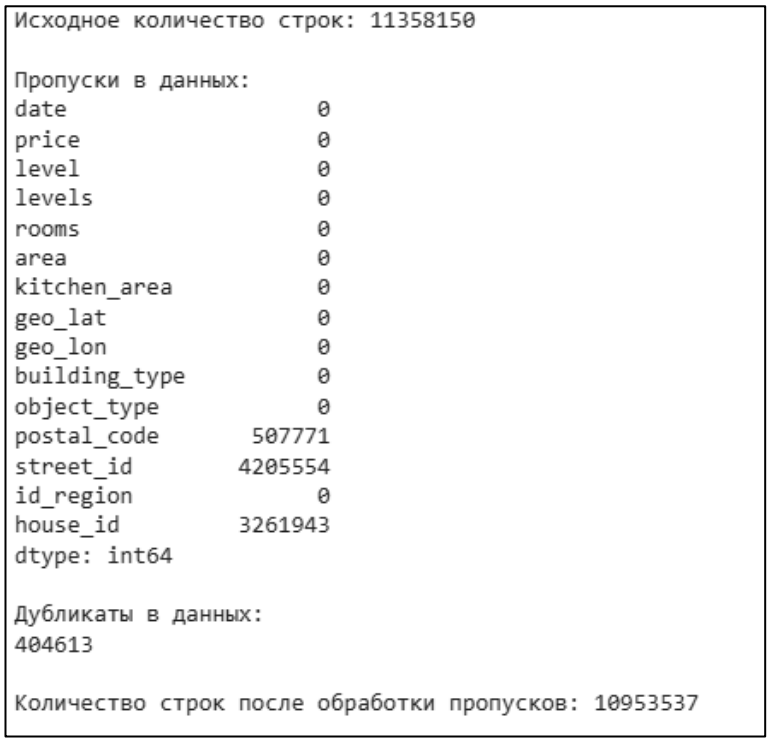

Рисунок 1 – Очистка данных

Шумы и выбросы могут привести к неверным выводам модели, затрудняя выделение значимых признаков. Например, если использовать алгоритм линейной регрессии – он пытается найти линию, которая минимизирует ошибки между прогнозом и фактом. Выбросы, которые сильно отличаются от остальных данных, могут «тянуть» эту линию и искажать результат.

Метод работы с выбросами:

Обнаружение выбросов с помощью Boxplot («Ящик с усами»): используется для визуализации выбросов. Выбросы отображаются как точки, находящиеся за пределами «усов» графика. На рисунке 2 представлено количество строк после обработки выбросов.

```
Удалено 0 строк с отрицательными значениями в столбце 'level'
Удалено 0 строк с отрицательными значениями в столбце 'levels'
Удалено 1079847 строк с отрицательными значениями в столбце 'kitchen_area'

Удалено 0 строк с некорректным количеством комнат (оставлены только студии (-1) и квартиры с 1+ комнатами)

Удалено 8770 строк с аномально большой площадью
Удалено 35668 строк с аномально маленькой площадью

Удалено 96985 строк с ценами выше 40,000,000 руб. (99-й перцентиль)
Удалено 17924 строк с ценами ниже 400,000 руб.

Количество строк после обработки выбросов: 9714343
```

Рисунок 2 – Обработка выбросов

После обработки и очистки данных мы делаем фильтрацию только по 28 региону, что помогло отобрать только нужные 30911 строк.

3.4.4 Этап 3 – Стандартизация и нормализация данных

Этот этап помогает оценить, как хорошо модель будет справляться с новыми данными, а также помогает вовремя обнаружить переобучение.

Стандартизация (Standardization)

Приведение признаков к нулевому среднему и стандартному отклонению, равному единице. Стандартизация делает признаки сопоставимыми по масштабу, но не обязательно приводит их к нормальному распределению.

В результате применения StandardScaler признаки, такие как «Площадь (кв. м)» и «Цена (млн руб.)», будут иметь среднее значение около 0 и стандартное отклонение 1. Но это не означает, что их распределение станет нормальным.

Нормализация (Min-Max Scaling)

Нормализация приводит значения признака к диапазону от 0 до 1, чтобы сделать их сопоставимыми. Она полезна, если у данных разные масштабы, что может мешать работе модели.

Нормализация важна для алгоритмов, чувствительных к масштабу признаков, таких как KNN или SVM. Но если в данных есть выбросы, Min-Max-нормализация может сжать остальные значения, потому что учитывает экстремальные.

После применения MinMaxScaler все числовые признаки будут находиться в диапазоне [0, 1]. На рисунке 3 представлены данные до применения MinMaxScaler, а на рисунке 4 уже с применением MinMaxScaler.

price (до)	level (до)	levels (до)	rooms (до)	area (до)	kitchen_area (до)
5900000.0	5.0	5.0	2.0	47.3	0.0
9800000.0	6.0	10.0	3.0	73.0	11.8
3350000.0	3.0	3.0	1.0	32.3	0.0
3021100.0	7.0	10.0	-1.0	30.3	0.0
3250000.0	1.0	4.0	-1.0	34.0	5.0

geo_lat (до)	geo_lon (до)	building_type (до)	object_type (до)
50.262087	127.543289	0.0	0.0
50.25894	127.56764	0.0	0.0
50.320724	127.508551	0.0	0.0
50.268029	127.5006467	0.0	0.0
50.2710109	127.4767121	4.0	0.0

Рисунок 3 – Данные до применения MinMaxScaler

price (после)	level (после)	levels (после)	rooms (после)	area (после)
0.13866853949615499	0.2272727272727273	0.13513513513513514	0.42857142857142855	0.12716535433070864
0.23719419153386756	0.2727272727272727	0.2702702702702703	0.5714285714285714	0.22834645669291337
0.07424792085611212	0.13636363636363635	0.08108108108108109	0.2857142857142857	0.06811023622047241
0.0659389242009317	0.3181818181818182	0.2702702702702703	0.0	0.06023622047244095
0.07172162208591437	0.045454545454545456	0.10810810810810811	0.0	0.07480314960629919

kitchen_area (после)	geo_lat (после)	geo_lon (после)	building_type (после)	object_type (после)
0.0	0.3799527710845867	0.8667310822204247	0.0	0.0
0.14460784313725492	0.37978999658364954	0.8669446808845082	0.0	0.0
0.0	0.382985693998938	0.8664263723361109	0.0	0.0
0.0	0.3802601133421559	0.8663570385132487	0.0	0.0
0.06127450980392157	0.3804143482593294	0.8661470923679008	0.6666666666666666	0.0

Рисунок 4 – Данные после применения MinMaxScaler

3.5 Визуализация данных

Для визуализации данных были построены гистограмма распределения цен и график зависимости цены от площади, они представлены на рисунке 5 и рисунке 6 соответственно. Дополнительно была построена гистограмма распределения типов зданий, она представлена н рисунке 7.

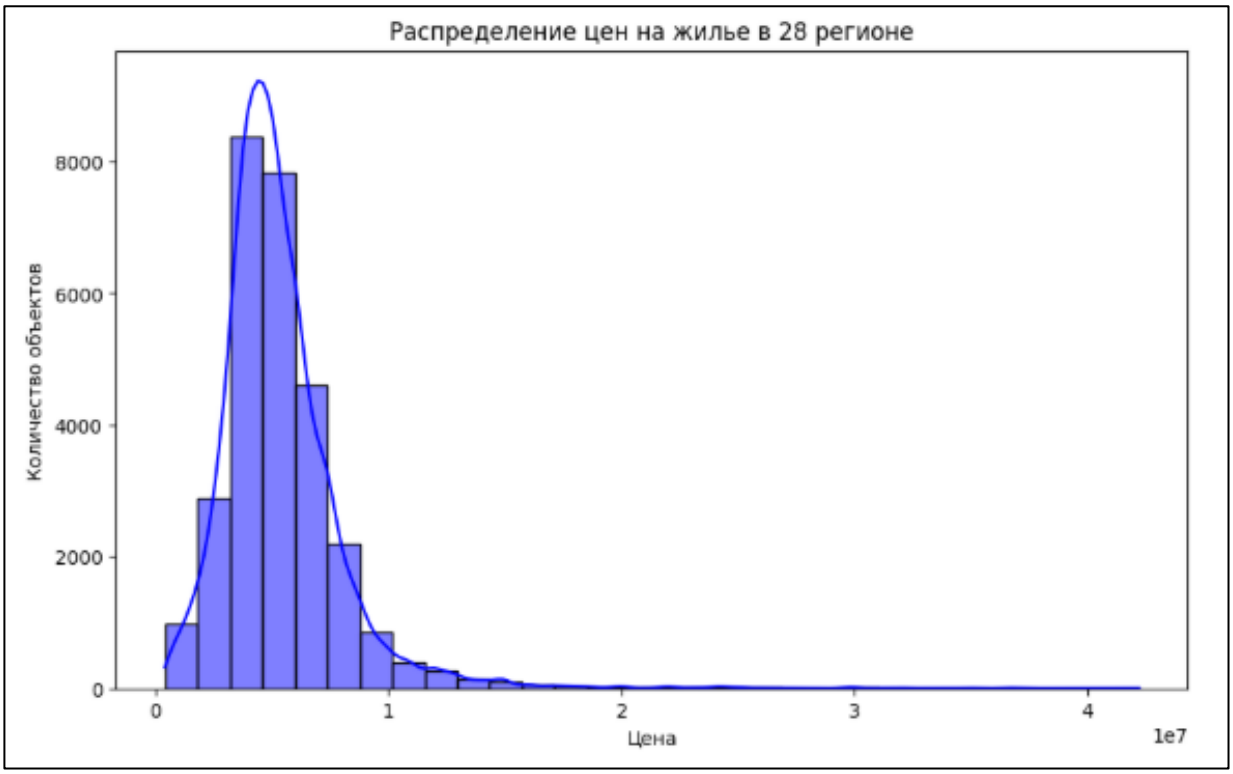

Рисунок 5 – Гистограмма распределения цен

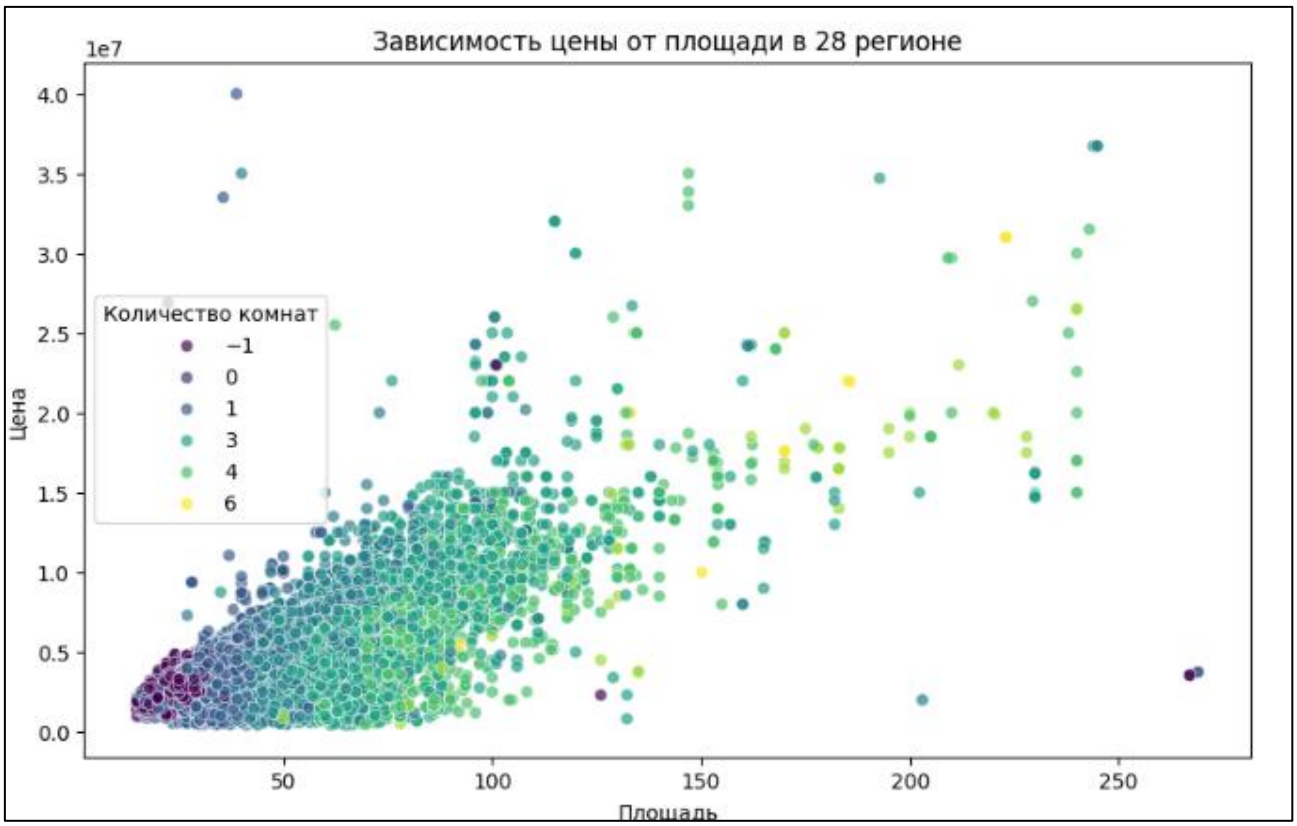

Рисунок 6 – График зависимости цены от площади

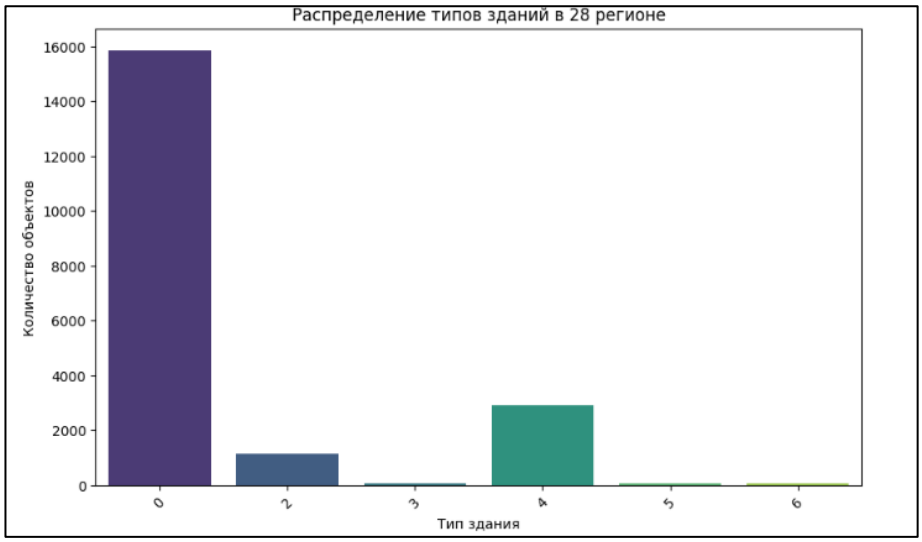

Рисунок 7 – Гистограмма распределения типов зданий

Для анализа взаимосвязи между признаками была построена матрица коэффициентов корреляции, она представлена на рисунке 8. Это позволило выявить наиболее значимые зависимости между признаками, такими как level и levels, а также rooms и area. Коэффициент корреляции между признаками level и levels равен 0.6108066371897789, что указывает на умеренную прямую связь**,** а коэффициент корреляции между признаками rooms и area равен 0.7253578693258891, что говорит о более сильной прямой связи.

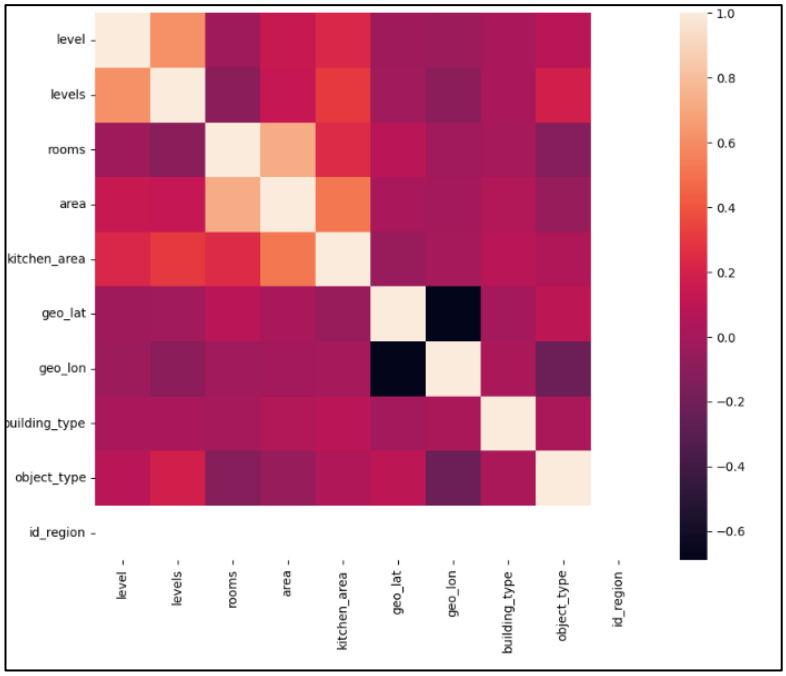

Рисунок 8 – Матрица коэффициентов корреляции признаков

3.6 Модель линейной регрессии

3.6.1 Построение модели с помощью функции LinearRegression

Когда мы сделали предобработку данных и отобрали для работы только 28 регион, сохраняем его в отдельный файл и с ним работаем.

Для наглядности представления данных и их структуры приведен фрагмент таблицы с первыми несколькими строками набора данных на рисунке 9.

	date	price	level	levels	rooms	area	kitchen_area	geo_lat	geo_lon	building_type	object_type	id_region
1090	2021-01-01	8000000	5	5	3	89.1	9.3	50.279400	127.521980	4	0	28
1372	2021-01-01	1150000	8	9	1	29.0	8.0	55.146713	124.739809	4	0	28
2081	2021-01-01	2400000	4	5	1	29.0	0.0	50.261460	127.573460	0	0	28
2338	2021-01-01	440000	4	5	3	66.5	0.0	49.801231	129.401535	0	0	28
2694	2021-01-01	2500000	1	5	3	69.2	69.2	51.990504	127.688268	2	0	28
...	...	...	...	...	...	...	...	...	...	...	...	...
11355820	2021-12-31	4850000	4	9	1	39.0	12.0	50.258556	127.561920	0	0	28
11356581	2021-12-31	3990000	4	9	1	27.0	5.0	50.267200	127.519180	0	0	28
11356713	2021-12-31	2050000	4	11	1	39.0	10.0	50.265592	127.538290	0	0	28
11357671	2021-12-31	4499000	6	9	1	38.9	10.0	50.259403	127.562030	0	0	28
11358079	2021-12-31	7700000	6	9	2	60.0	8.2	50.316358	127.506593	0	0	28

30911 rows × 12 columns

Рисунок 9 – Фрагмент набора данных

Данные были разделены на обучающую и тестовую выборки в соотношении 80:20.

Для прогнозирования стоимости недвижимости применялась модель LinearRegression из библиотеки sklearn.linear_model [26]. Работа проводилась только с 28 регионом. Модель была обучена на тренировочной выборке, а затем была проведена оценка ее качества на тестовой выборке. После обучения получаем следующее уравнение регрессии:

$$\hat{y} = 589122x_1 + 5674984x_2 - 1315735x_3 + 24590998x_4 + 1392492x_5 -$$
$$-9735188x_6 - 3083408x_7 - 611206x_8 - 552928x_9 + 0x_{10},$$

где x_1 – level, x_2 – levels, x_3 – rooms, x_4 – area, x_5 – kitchen_area, x_6 – geo_lat, x_7 – geo_lon, x_8 – building_type, x_9 – object_type, x_{10} – id_region.

Обучение модели линейной регрессии производилось на масштабированных с использованием MinMaxScaler данных. Визуализация коэффициентов регрессии представлена на рисунке 10. Поскольку все признаки находятся в одном диапазоне, то коэффициенты можно сравнивать между собой. Результат моделирования показал очевидный факт – наибольший вклад в прогноз приносит переменная area, отвечающая за площадь квартиры.

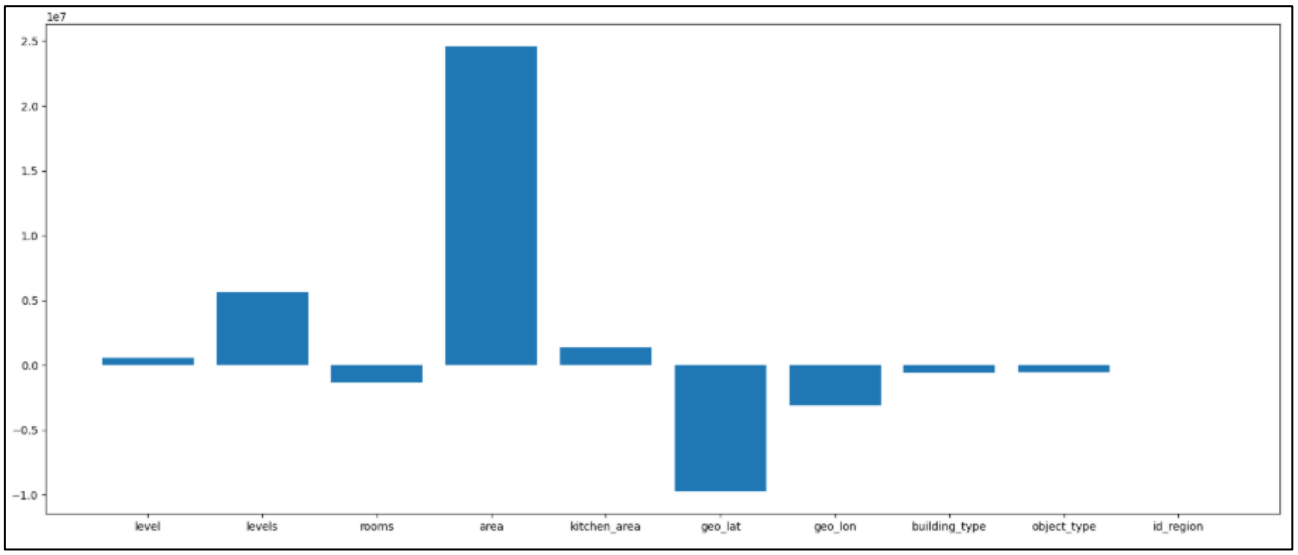

Рисунок 10 – Коэффициенты регрессии (на нормированных данных с помощью MinMaxScaler)

3.6.2 Оценка качества модели

Для оценки качества модели были использованы следующие метрики:

– MAE (Mean Absolute Error): Средняя абсолютная ошибка.

– MSE (Mean Squared Error): Средняя квадратичная ошибка.

– R^2 (R-squared): Коэффициент детерминации.

В рамках оценки качества модели были получены следующие результаты, которые представлены в таблице 2.

MAE означает, на сколько в среднем предсказания модели отклоняются от реальных значений. Чем меньше это значение, тем точнее модель.

Средняя квадратичная ошибка показывает, насколько сильно отклоняются предсказания от реальных значений. Высокое значение MSE указывает на наличие значительных ошибок в прогнозах.

Коэффициент детерминации R^2 в первом случае равен 0.669, что означает, что модель объясняет около 66.9% дисперсии данных. Это свидетельствует о том, что модель достаточно хорошо описывает зависимость между признаками и целевой переменной, но есть возможности для улучшения. В остальных случаях коэффициент детерминации показывает меньший результат. Это достаточно хороший результат для такого большого набора данных и для такой простой модели.

Также проводилось исследование качества при обучении без некоторых признаков. Результат представлен в таблице 2. Как и следовало ожидать, качество модели сильно ухудшается, если не включить в набор данных такие важные признаки как 'level', 'rooms', 'area'.

Таблица 2 – Метрики качества

Метрики	Результаты со всеми признаками	Результаты без признаков 'geo_lat', 'geo_lon'	Результаты без признаков 'kitchen_area', 'building_type', 'object_type'	Результаты без признаков 'kitchen_area', 'building_type'	Результаты без признаков 'level', 'rooms', 'area'
MAE	893979	1023754	1043158	937809	0.039857
MSE	1888886145879	2493267126391	2543138882560	2239217091784	0.00348879
RMSE	1374367	1579008	1594722	1496401	0.0590660290
R^2	0.703	0.6339	0.6266	0.6713	0.189160522

Результаты прогнозирования с использованием линейной регрессии представлены на рисунке 11 и 12. На рисунке 13 представлена ошибок прогнозирования в процентах для первых 50 объектов.

```
Объект 1: Предсказано = 6783337.56, Реально = 5200000.00, Ошибка = 1583337.56
Объект 2: Предсказано = 4107771.44, Реально = 4699000.00, Ошибка = 591228.56
Объект 3: Предсказано = 4533439.37, Реально = 4350000.00, Ошибка = 183439.37
Объект 4: Предсказано = 10043311.20, Реально = 11000000.00, Ошибка = 956688.80
Объект 5: Предсказано = 4954346.45, Реально = 5651500.00, Ошибка = 697153.55
Объект 6: Предсказано = 7740262.31, Реально = 6800000.00, Ошибка = 940262.31
Объект 7: Предсказано = 3642172.69, Реально = 3256000.00, Ошибка = 386172.69
Объект 8: Предсказано = 3865355.20, Реально = 1530000.00, Ошибка = 2335355.20
Объект 9: Предсказано = 3951319.30, Реально = 5450000.00, Ошибка = 1498680.70
Объект 10: Предсказано = 4475678.01, Реально = 5620000.00, Ошибка = 1144321.99
Объект 11: Предсказано = 5756384.65, Реально = 7176000.00, Ошибка = 1419615.35
Объект 12: Предсказано = 4354794.48, Реально = 3800000.00, Ошибка = 554794.48
Объект 13: Предсказано = 2098390.37, Реально = 2200000.00, Ошибка = 101609.63
Объект 14: Предсказано = 6291170.92, Реально = 7222400.00, Ошибка = 931229.08
Объект 15: Предсказано = 4356684.49, Реально = 4300000.00, Ошибка = 56684.49
Объект 16: Предсказано = 5160339.96, Реально = 5875000.00, Ошибка = 714660.04
Объект 17: Предсказано = 2810198.88, Реально = 3550000.00, Ошибка = 739801.12
Объект 18: Предсказано = 6330185.46, Реально = 7500000.00, Ошибка = 1169814.54
Объект 19: Предсказано = 4357079.94, Реально = 4100000.00, Ошибка = 257079.94
Объект 20: Предсказано = 4452394.33, Реально = 4700000.00, Ошибка = 247605.67
Объект 21: Предсказано = 6376918.05, Реально = 6100000.00, Ошибка = 276918.05
Объект 22: Предсказано = 4799932.18, Реально = 5400000.00, Ошибка = 600067.82
Объект 23: Предсказано = 5441640.84, Реально = 4000000.00, Ошибка = 1441640.84
Объект 24: Предсказано = 3650420.41, Реально = 4150000.00, Ошибка = 499579.59
Объект 25: Предсказано = 3404541.41, Реально = 3780000.00, Ошибка = 375458.59
```

Рисунок 11 – Результаты прогнозирования

```
Объект 26: Предсказано = 3669670.73, Реально = 3400000.00, Ошибка = 269670.73
Объект 27: Предсказано = 4906539.84, Реально = 4500000.00, Ошибка = 406539.84
Объект 28: Предсказано = 3002133.50, Реально = 2450000.00, Ошибка = 552133.50
Объект 29: Предсказано = 4276072.43, Реально = 5500000.00, Ошибка = 1223927.57
Объект 30: Предсказано = 3082740.15, Реально = 3550000.00, Ошибка = 467259.85
Объект 31: Предсказано = 6437925.53, Реально = 6800000.00, Ошибка = 362074.47
Объект 32: Предсказано = 5305574.47, Реально = 3200000.00, Ошибка = 2105574.47
Объект 33: Предсказано = 2837718.38, Реально = 2900000.00, Ошибка = 62281.62
Объект 34: Предсказано = 4208118.00, Реально = 4400000.00, Ошибка = 191882.00
Объект 35: Предсказано = 3681314.82, Реально = 2554500.00, Ошибка = 1126814.82
Объект 36: Предсказано = 5973032.76, Реально = 5000000.00, Ошибка = 973032.76
Объект 37: Предсказано = 5658538.96, Реально = 4900000.00, Ошибка = 758538.96
Объект 38: Предсказано = 5486098.02, Реально = 5100000.00, Ошибка = 386098.02
Объект 39: Предсказано = 5489255.78, Реально = 6104700.00, Ошибка = 615444.22
Объект 40: Предсказано = 5169269.58, Реально = 6200000.00, Ошибка = 1030730.42
Объект 41: Предсказано = 4871793.93, Реально = 7100000.00, Ошибка = 2228206.07
Объект 42: Предсказано = 6497794.21, Реально = 9750000.00, Ошибка = 3252205.79
Объект 43: Предсказано = 4508629.68, Реально = 6500000.00, Ошибка = 1991370.32
Объект 44: Предсказано = 4413405.60, Реально = 4990000.00, Ошибка = 576594.40
Объект 45: Предсказано = 3254680.76, Реально = 3425500.00, Ошибка = 170819.24
Объект 46: Предсказано = 4979114.96, Реально = 1050000.00, Ошибка = 3929114.96
Объект 47: Предсказано = 4708850.84, Реально = 4950000.00, Ошибка = 241149.16
Объект 48: Предсказано = 3804823.43, Реально = 3500000.00, Ошибка = 304823.43
Объект 49: Предсказано = 4904902.50, Реально = 5300000.00, Ошибка = 395097.50
Объект 50: Предсказано = 4707858.82, Реально = 5100000.00, Ошибка = 392141.18
```

Рисунок 12 – Результаты прогнозирования

```
Объект 1: Ошибка = 30.45%
Объект 2: Ошибка = 12.58%
Объект 3: Ошибка = 4.22%
Объект 4: Ошибка = 8.70%
Объект 5: Ошибка = 12.34%
Объект 6: Ошибка = 13.83%
Объект 7: Ошибка = 11.86%
Объект 8: Ошибка = 152.64%
Объект 9: Ошибка = 27.50%
Объект 10: Ошибка = 20.36%
Объект 11: Ошибка = 19.78%
Объект 12: Ошибка = 14.60%
Объект 13: Ошибка = 4.62%
Объект 14: Ошибка = 12.89%
Объект 15: Ошибка = 1.32%
Объект 16: Ошибка = 12.16%
Объект 17: Ошибка = 20.84%
Объект 18: Ошибка = 15.60%
Объект 19: Ошибка = 6.27%
Объект 20: Ошибка = 5.27%
Объект 21: Ошибка = 4.54%
Объект 22: Ошибка = 11.11%
Объект 23: Ошибка = 36.04%
Объект 24: Ошибка = 12.04%
Объект 25: Ошибка = 9.93%
Объект 26: Ошибка = 7.93%
Объект 27: Ошибка = 9.03%
Объект 28: Ошибка = 22.54%
Объект 29: Ошибка = 22.25%
Объект 30: Ошибка = 13.16%
Объект 31: Ошибка = 5.32%
Объект 32: Ошибка = 65.80%
Объект 33: Ошибка = 2.15%
Объект 34: Ошибка = 4.36%
Объект 35: Ошибка = 44.11%
Объект 36: Ошибка = 19.46%
Объект 37: Ошибка = 15.48%
Объект 38: Ошибка = 7.57%
Объект 39: Ошибка = 10.08%
Объект 40: Ошибка = 16.62%
Объект 41: Ошибка = 31.38%
Объект 42: Ошибка = 33.36%
Объект 43: Ошибка = 30.64%
Объект 44: Ошибка = 11.55%
Объект 45: Ошибка = 4.99%
Объект 46: Ошибка = 374.20%
Объект 47: Ошибка = 4.87%
Объект 48: Ошибка = 8.71%
Объект 49: Ошибка = 7.45%
Объект 50: Ошибка = 7.69%
```

Рисунок 13 – Ошибка прогнозирования в процентах

3.7 Нейронная сеть

Нейронные сети – это мощный метод машинного обучения, который используется для моделирования сложных нелинейных зависимостей. Нейронные сети вдохновлены биологическими нейронными сетями и состоят из слоев нейронов, которые обрабатывают входные данные и передают их через последовательные слои для получения выходного значения. Нейронные сети могут быть использованы для задач классификации, регрессии, обработки изображений, текстов и многих других [9].

Преимущества:

– Способность моделировать сложные зависимости: Нейронные сети могут моделировать сложные нелинейные зависимости, что делает их мощным инструментом для решения различных задач.

– Гибкость: Нейронные сети могут быть адаптированы для различных задач, таких как регрессия, классификация, обработка изображений и текстов.

– Высокая точность: На больших данных нейронные сети часто показывают высокую точность, особенно в задачах с большим количеством признаков.

– Автоматическое извлечение признаков: В глубоких нейронных сетях скрытые слои могут автоматически извлекать полезные признаки из данных.

Ограничения:

– Высокая вычислительная сложность: Обучение нейронных сетей требует значительных вычислительных ресурсов, особенно на больших данных. Для ускорения обучения часто используются GPU или TPU.

– Необходимость большого объема данных: Нейронные сети требуют большого количества данных для обучения, чтобы избежать переобучения.

– Сложность интерпретации: Нейронные сети сложнее интерпретировать по сравнению с другими методами, такими как линейная регрессия или деревья решений.

– Чувствительность к гиперпараметрам: Эффективность нейронных сетей сильно зависит от выбора гиперпараметров, таких как количество слоев, количество нейронов в слоях, функция активации и скорость обучения.

Существуют однослойные нейронные сети и многослойные нейронные сети.

3.7.1 Виды нейронных сетей

Однослойная нейронная сеть

Один нейрон может выполнять простейшие вычисления, но основные функции нейросети обеспечиваются не отдельными нейронами, а 19 соединениями между ними. Однослойный перцептрон представляет собой простейшую сеть, которая состоит из группы нейронов, образующих слой. Входные данные кодируются вектором значений, каждый элемент подается на соответствующий вход каждого нейрона в слое. В свою очередь, нейроны вычисляют выход независимо друг от друга. Размерность выхода (то есть количество элементов) равна количеству нейронов, а количество синапсов у всех нейронов должно быть одинаково и совпадать с размерностью входного сигнала, это можно посмотреть на рисунке 14.

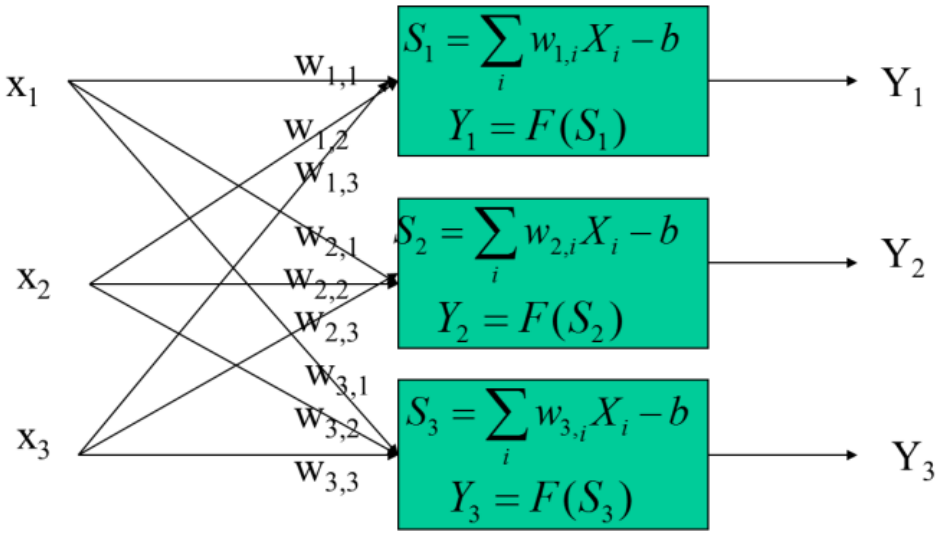

Рисунок 14 – Однослойная нейронная сеть

Здесь X_1, X_2, X_3 – называется входной паттерн, Y_1, Y_2, Y_3 – выходной паттерн, а $w_{i,j}$ – это j-ый весовой коэффициент i-го нейрона.

Многослойная нейронная сеть

Многослойная нейронная сеть (персептрон) — это нейронная сеть, состоящая из входного, выходного и расположенных между ними одного (или нескольких) скрытых слоев нейронов. Многослойная нейронная сеть представлена на рисунке 15.

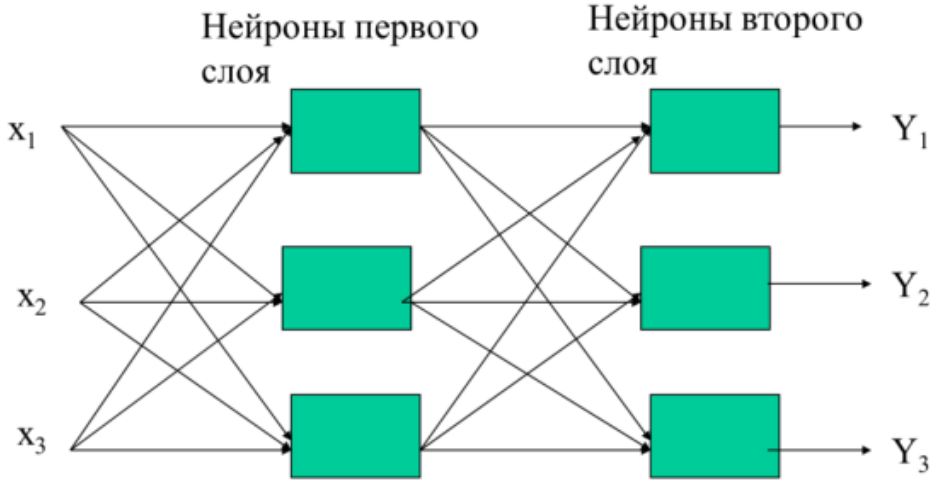

Рисунок 15 – Многослойная нейронная сеть

Чтобы построить многослойный персептрон, необходимо выбрать его параметры по следующему алгоритму:

– Определить, какой смысл вкладывается в компоненты входного вектора *X*. Входной вектор должен содержать формализованное условие задачи, то есть всю информацию, необходимую для того, чтобы получить ответ.

– Выбрать выходной вектор *Y* таким образом, чтобы его компоненты содержали полный ответ для поставленной задачи.

– Выбрать вид функции активации нейронов. При этом желательно учесть специфику задачи, так как удачный выбор увеличит скорость обучения.

– Выбрать количество слоев и нейронов в слое.

– Задать диапазон изменения входов, выходов, весов и пороговых уровней на основе выбранной функции активации.

– Присвоить начальные значения весам и пороговым уровням. Начальные значения не должны быть большими, чтобы нейроны не оказались в насыщении (на горизонтальном участке функции активации), иначе обучение будет очень медленным. Начальные значения не должны быть и слишком малыми, чтобы выходы большей части нейронов не были равны нулю, иначе обучение тоже замедлится.

– Провести обучение, то есть подобрать параметры сети так, чтобы задача решалась наилучшим образом. По окончании обучения сеть сможет решать задачи того типа, которым она обучена.

– Подать на вход сети условия задачи в виде вектора *X*. Рассчитать выходной вектор *Y*, который и даст формализованное решение задачи [11,12].

3.7.2 Описание архитектуры и настройка гиперпараметров

Для нейронной сети мы работаем с файлом «preprocessed_data_28_region.csv», там у нас будут находиться записи только по 28 региону, которые прошли полную предобработку.

Дальнейшее наше действие повторяется как и в случае с линейной регрессией, надо разбить наши данные на обучающую и тестовую.

Дальше уже начинаем работать с нейронной сетью. Создаем многослойную нейронную сеть с полносвязными слоями (64, 128, 256, 512, 1024, 2048 нейронов) и активацией ReLU, представленную на рисунке 16. Выходной слой с одним нейроном и ReLU предсказывает положительное число, в нашем случае цену.

Model: "sequential"

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 64)	704
dense_1 (Dense)	(None, 128)	8,320
dense_2 (Dense)	(None, 256)	33,024
dense_3 (Dense)	(None, 512)	131,584
dense_4 (Dense)	(None, 1024)	525,312
dense_5 (Dense)	(None, 2048)	2,099,200
dense_6 (Dense)	(None, 1)	2,049

Рисунок 16 – Структура нейронной сети

Дальше компилируем и обучаем нейронную сеть. Настройка нейронной сети представлено в таблице 3.

Таблица 3 – Настройка сети

Оптимизатор	Метрики качества	Количество эпох	Количество примеров	% данных для валидации
Adam	MAE, MSE	280	1500	20

Сеть последовательно улучшает свои предсказания, уменьшая ошибку на тренировочных данных и проверяя качество на валидационной выборке. Если разница между ошибками на обучении и валидации растёт – это признак переобучения. Выведем график, где посмотрим, как ведет себя средняя абсолютная ошибка на обучающем наборе и средняя абсолютная ошибка на проверочном наборе на протяжении 280 эпох, это представлено на рисунке 17.

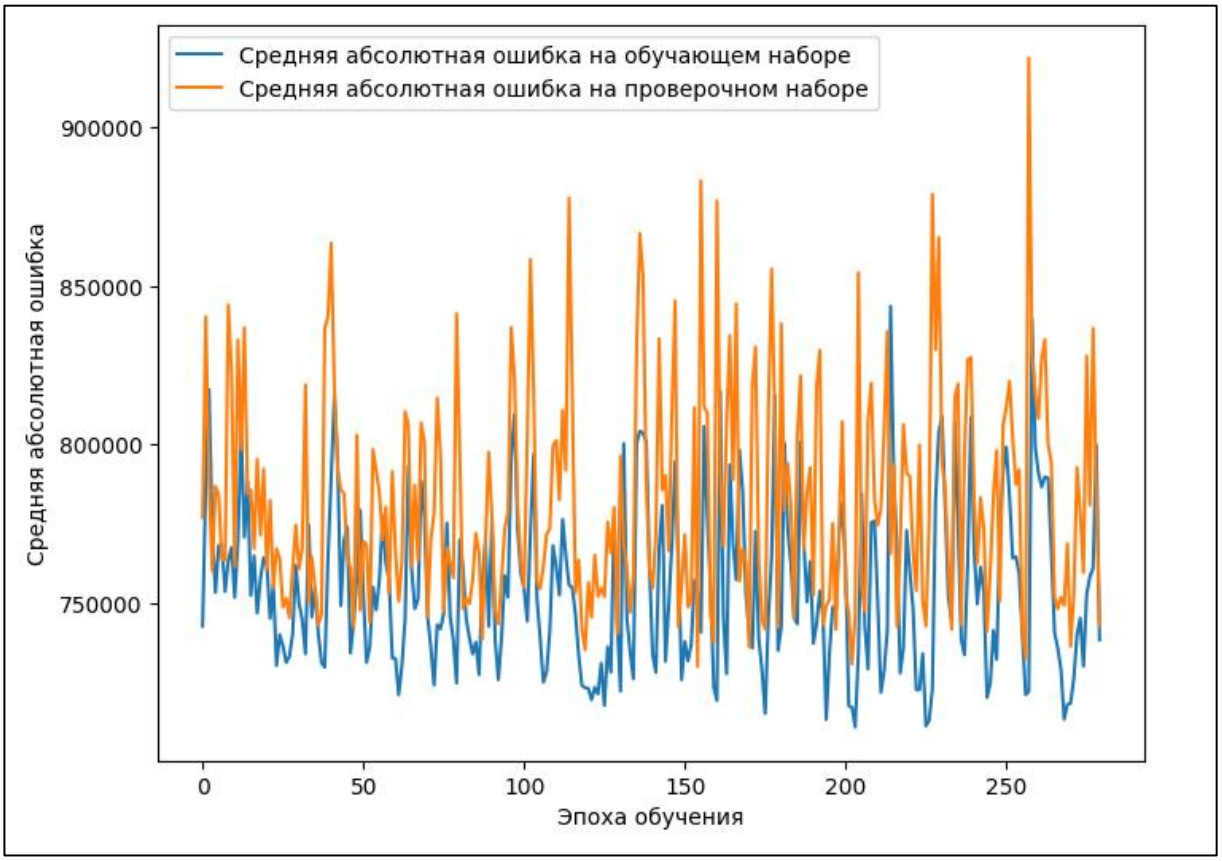

Рисунок 17 – График средних абсолютных ошибок

Дальше просмотрим какая средняя абсолютная ошибка на тестовых данных. Она равна 733348 рубля.

3.7.3 Оценка качества модели

Оценим качество обученной модели на тестовых данных, используя три ключевые метрики. Результаты представлены в таблице 4. В таблице представлены данные, когда только началось обучение модели и данные, когда модель подвергалась постоянным доработкам.

Таблица 4 – Метрики качества нейронной сети

Метрики качества	Данные на начале испытаний	На данное время
MAE	996622.6875	733348.1875
MSE	2351659745280.0	1276468396032.0
R^2	0.6547660827636719	0.799

По этим данным можем сказать, что если допускается округление, то мы получаем модель хорошего качества.

Также представим значения минимальных относительных ошибок, максимальных и средних относительных ошибок, на начале нашего обучения и какие показатели сейчас в таблице 5.

Таблица 5 – Значения относительных ошибок

Время проведения обучения	Минимальная относительная ошибка	Максимальная относительная ошибка	Средняя относительная ошибка
Начало обучения	2.7522935e-05 %.	1706.4005 %.	31.08195178 %.
На данное время	0.0022924901 %.	887.1378333 %.	19.245729637 %

На рисунке 18 представлен график предсказанных значений к правильным.

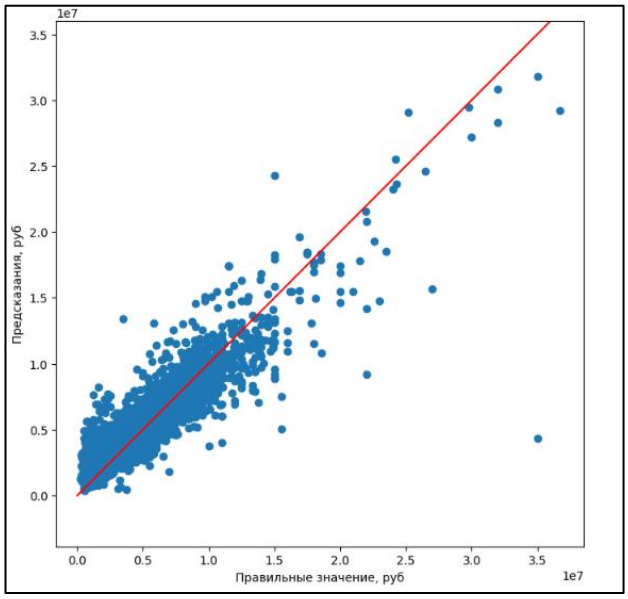

Рисунок 18 – График предсказанных значений к правильным

Результаты прогнозирования с использованием нейронных сетей представлены на рисунке 19 и на рисунке 20 представлены относительные ошибки.

Примеры предсказаний:

	Реальное значение	Предсказанное значение	Абсолютная ошибка
0	5900000	4423161.00	1476839.00
1	9800000	7292220.00	2507780.00
2	3350000	2947222.00	402778.00
3	3021100	3392159.75	371059.75
4	3250000	3357285.00	107285.00
5	10700000	10165764.00	534236.00
6	8500000	6254734.50	2245265.50
7	3100000	3527977.25	427977.25
8	2025000	2260722.50	235722.50
9	6500000	5306493.00	1193507.00
10	6600000	6045150.00	554850.00
11	7632000	6680017.50	951982.50
12	3350000	2892703.25	457296.75
13	3600000	4726724.50	1126724.50
14	3600000	2822652.50	777347.50
15	6300000	6236584.00	63416.00
16	5100000	2885820.25	2214179.75
17	3300000	4365718.50	1065718.50
18	5000000	5104433.50	104433.50
19	5900000	4514351.50	1385648.50
20	6500000	5639260.00	860740.00
21	5100000	5725460.00	625460.00
22	3920000	3626887.00	293113.00
23	4500000	3792236.00	707764.00
24	3600000	3773788.75	173788.75
25	3916500	4281956.00	365456.00
26	3300000	3331970.00	31970.00
27	2800000	2501953.75	298046.25
28	4450000	4010166.50	439833.50
29	1750000	2486248.00	736248.00

Рисунок 19 – Результаты прогнозирования

	Относительная ошибка (%)
0	25.031169
1	25.589592
2	12.023224
3	12.282273
4	3.301077
5	4.992860
6	26.414888
7	13.805718
8	11.640617
9	18.361646
10	8.406818
11	12.473565
12	13.650649
13	31.297903
14	21.592986
15	1.006603
16	43.415289
17	32.294500
18	2.088670
19	23.485568
20	13.242154
21	12.263922
22	7.477372
23	15.728089
24	4.827465
25	9.331189
26	0.968788
27	10.644509
28	9.883899
29	42.071314

Рисунок 20 – Относительные ошибки

3.8 Обсуждение результатов

В ходе исследования были применены два метода машинного обучения для прогнозирования стоимости жилья в 28-м регионе:

Линейная регрессия (простая и интерпретируемая модель). Нейронная сеть (более сложная модель, способная улавливать нелинейные зависимости).

Сравнение качества моделей представлено в таблице 6.

Для оценки эффективности моделей использовались метрики:

MAE (Mean Absolute Error) – средняя абсолютная ошибка

MSE (Mean Squared Error) – средняя квадратичная ошибка

R^2 (Коэффициент детерминации) – доля объясненной дисперсии

Таблица 6 – Сравнение качества моделей

Метрика	Линейная регрессия	Нейронная сеть
MAE	893979	733348
MSE	1888886145879	1276468396032
R^2	0.703	0.799

Нейронная сеть показала лучшие результаты по всем метрикам. Линейная регрессия дала приемлемый результат, но хуже справляется с нелинейными зависимостями. Линейная регрессия оказалась менее точной, но ее преимущество – простота интерпретации коэффициентов. Она подходит для базового анализа, когда важна прозрачность модели. Нейронная сеть продемонстрировала наилучшие результаты, так как смогла уловить сложные нелинейные зависимости в данных, которые линейная модель игнорирует.

3.8.1 Ограничения и возможные улучшения

Ограничения моделей.

– Линейная регрессия

Не учитывает нелинейные зависимости: Например, цена может расти экспоненциально с увеличением площади или резко меняться в зависимости от района. Линейная модель не способна уловить такие сложные взаимосвязи.

Чувствительность к выбросам: Даже после предобработки оставшиеся аномалии могут искажать коэффициенты регрессии.

Предположение о линейной связи: Модель предполагает, что все признаки влияют на цену аддитивно, что не всегда соответствует реальности.

– Нейронная сеть

Требует больших данных: Для устойчивого обучения глубоких сетей необходимо значительно больше данных, чем для линейных моделей.

Сложность настройки: Выбор архитектуры (число слоев, нейронов), функции активации, скорости обучения требует экспериментов и может быть трудоемким.

Риск переобучения: Без регуляризации (например, Dropout, L2-нормализация) сеть может "запомнить" обучающие данные вместо выявления общих закономерностей.

Возможные улучшения.

Добавление новых признаков.

– Геоданные: Удаленность от центра, ближайшего метро, парков, школ.

– Инфраструктура: Наличие торговых центров, больниц, транспортной доступности.

– Временные факторы: Динамика цен по кварталам/годам, если данные временные.

Глубокая настройка нейронной сети

– Оптимизация гиперпараметров: Использование GridSearch или Bayesian Optimization для подбора оптимальных параметров.

– Расширение архитектуры: Добавление сверточных (CNN) или рекуррентных слоев (LSTM), если данные имеют пространственную/временную структуру.

– Регуляризация: Применение Dropout, Batch Normalization для борьбы с переобучением.

ЗАКЛЮЧЕНИЕ

В ходе выполнения бакалаврской работы было проведено комплексное исследование методов прогнозирования цен на жилье с использованием линейной регрессии и нейронных сетей. Особое внимание уделено анализу влияния состава признаков на качество линейной регрессионной модели.

Для оценки устойчивости модели проводилась серия экспериментов с последовательным исключением различных групп признаков. Результаты показали, что линейная регрессия демонстрирует разную степень чувствительности к удалению тех или иных характеристик. Наибольшее влияние на точность прогнозирования оказывают такие параметры, как площадь помещения, количество комнат и этаж расположения – их исключение приводит к существенному снижению коэффициента детерминации R^2. В то же время модель оказалась менее чувствительной к удалению второстепенных признаков, таких как тип фасада и тип объекта.

Полученные результаты имеют важное практическое значение для разработки систем автоматизированной оценки недвижимости. Они позволяют оптимизировать процесс сбора данных, сосредоточившись на наиболее значимых характеристиках объектов, и создавать более эффективные модели прогнозирования. Дальнейшие исследования в этом направлении могут быть связаны с разработкой методики автоматического отбора наиболее информативных признаков для задач оценки жилой недвижимости.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1 Андропова О.Ю. Искусственный интеллект и язык программирования Python / О. Ю. Андропова, И. И. Васильева, Н. А. Гнездилова. – Елец: ЕГУ им. И.А. Бунина, 2024 – 106 с.

2 Баланов, А.Н. Машинное обучение и искусственный интеллект: учебное пособие для вузов / А.Н. Баланов. – 2-е изд., стер. – Санкт-Петербург: Лань, 2025. – 172 с.

3 Барский, А.Б. Введение в нейронные сети: учебное пособие / А.Б. Барский. – 4-е изд. – Москва: Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2024. – 357 с.

4 Воронина, В.В. Теория и практика машинного обучения: учебное пособие / В.В. Воронина, А.В. Михеев, Н.Г. Ярушкина, К.В. Святов. – Ульяновск: УлГТУ, 2017. – 290 с

5 Воронова, Л.И. Machine Learning: регрессионные методы интеллектуального анализа данных: учебное пособие / Л.И. Воронова, В.И. Воронов. – Москва: Московский технический университет связи и информатики, 2018. – 82 с.

6 Гладилин, П.Е. Технологии машинного обучения: учебное пособие / П. Е. Гладилин – СПб.: Университет ИТМО, 2020. – 75 с.

7 Горбаченко, В.И. Машинное обучение: настраиваем ПО, готовим данные, анализируем / В.И. Горбаченко, К.Е. Савенков, М.А. Малахов. – Москва, Алматы: Ай Пи Ар Медиа, EDPHub (ИдипиХаб), 2024. – 248 с.

8 Горбаченко, В.И. Машинное обучение: учебное пособие / В.И. Горбаченко, К.Е. Савенков, М.А. Малахов. – Москва: Ай Пи Ар Медиа, 2023. – 217 с.

9 Груздев, А.В. Изучаем Pandas / А. В. Груздев, М. Хейдт; перевод с английского А. В. Груздева. – 2-ое изд., испр. и доп. – Москва: ДМК Пресс, 2019. – 700 с.

10 Джулли, А. Библиотека Keras – инструмент глубокого обучения. Реализация нейронных сетей с помощью библиотек Theano и TensorFlow / А. Джулли, С. Пал: перевод А.А. Слинкин. – Москва: ДМК Пресс, 2018. – 294 с.

11 Казанцев Т. Искусственный интеллект и Машинное обучение. Основы программирования на Python / Т. Казанцев – «ЛитРес: Самиздат», 2020

12 Картер, Д. Машинное обучение [Электронный ресурс]: учебное пособие / Д. Картер – Режим доступа: https://www.litres.ru/book/dzheyd-karter/mashinnoe-obuchenie-69356998/chitat-onlayn/

13 Косарев, В. С. Нейронные сети в экономике и финансах / В. С. Косарев. – Москва: Дело, 2021. – 118 с.

14 Котельников, Е.В. Введение в машинное обучение и анализ данных: учебное пособие / Е. В. Котельников, А. В. Котельникова. – Киров: ВятГУ, 2023. – 68 с.

15 Митяков, Е.С. Искусственный интеллект и машинное обучение: учебное пособие для вузов / Е.С. Митяков, А.Г. Шмелева, А.И. Ладынин. – Санкт-Петербург: Лань, 2025. – 252 с.

16 Мухаммедиев, Р.И. Введение в машинное обучение [Электронный ресурс]: учебное пособие / Р.И. Мухаммедиев – Алматы, 2022. – 252 с. – Режим доступа: https://geoml.info/wp-conten/uploads/2022/04/MLF_Theory_Practice_short_v.3.36.pdf

17 Назаров, Д.М. Data Science и интеллектуальный анализ данных: учебное пособие / Д.М. Назаров, С.В. Бегичева, Д.Б. Ковтун, А.Д. Назаров. – Москва: Ай Пи Ар Медиа, 2023. – 304 с.

18 Наумов, В.Н. Анализ данных и машинное обучение. Методы и инструментальные средства: учебное пособие / В.Н. Наумов. – Москва: Дело РАНХиГС, 2020. – 260 с.

19 Рамсундар, Б. TensorFlow для глубокого обучения: Пер. с англ. / Б. Рамсундар, Р. Б. Заде. – СПб.: БХВ-Петербург, 2019. – 256 с.

20 Рашка, С. Python и машинное обучение: крайне необходимое пособие по новейшей предсказательной аналитике, обязательное для более глубокого понимания методологии машинного обучения: руководство / С. Рашка; перевод с английского А. В. Логунова. – Москва: ДМК Пресс, 2017. – 418 с.

21 Титов, А.Н. Обработка данных в Python. Основы работы с библиотекой Pandas: учебно-методическое пособие / А.Н. Титов, Р. Ф. Тазиева. – Казань: Издательство КНИТУ, 2022. – 116 с.

22 Титов, А.Н. Интерактивная визуализация данных. Работа с библиотекой Plotly : учебно-методическое пособие / А. Н. Титов, Р. Ф. Тазиева. – Казань: КНИТУ, 2023. – 136 с.

23 Целых, А.Н. Извлечение знаний методами машинного обучения: учебное пособие по курсам «Модели и методы инженерии знаний», «Методы машинного обучения» / А.Н. Целых, Э.М. Котов. – Ростов-на-Дону, Таганрог: Издательство Южного федерального университета, 2022. – 105 с.

24 Шарден, Б. Крупномасштабное машинное обучение вместе с Python / Б. Шарден, Л. Массарон, А. Боскетти; перевод А.В. Логунов. – Москва: ДМК Пресс, 2018. – 358 с.

25 Agniashvili, D. Russian Real Estate 2021 [Электронный ресурс]: набор данных / D. Agniashvili – Режим доступа: https://www.kaggle.com/datasets/mrdaniilak/russia-real-estate-2021

26 Scikit-learn documentation [Электронный ресурс] – Режим доступа: https://scikit-learn.org/stable/modules/linear_model.html

Список работ, опубликованных автором по теме исследования

27 Фарутсова, Е.А. Прогнозирование стоимости жилья в рф (по набору данных за 2021 год)/ Е.А. Фартусова // День науки: Материалы XXXIV научной конференции Амурского государственного университета, Благовещенск, 13 марта 2025 года. – Благовещенск: Амурский государственный университет, 2025. – *в печати*

ПРИЛОЖЕНИЕ А

Листинг программы для предобработки и визуализации данных

```
From google.colab import user data

! pip install kaggle

! mkdir ~/.kaggle

! cp kaggle.json ~/.kaggle/

! chmod 600 ~/.kaggle/kaggle.json

! kaggle datasets download mrdaniilak/russia-real-estate-2021

import kagglehub

path = kagglehub.dataset_download("mrdaniilak/russia-real-estate-2021")

print("Path to dataset files:", path)

! unzip /content/russia-real-estate-2021.zip

# 1. Загрузка данных

df = pd.read_csv('input_data.csv', delimiter=';')

print(df.shape)

df.head()

# Проверка исходного количества строк

print(f"Исходное количество строк: {len(df)}")

print("\nПропуски в данных:")

print(df.isnull().sum())

# Удаление столбцов 'house_id', 'postal_code', 'street_id'

df.drop(columns=['house_id', 'postal_code', 'street_id'], inplace=True)

print("\nДубликаты в данных:")

print(df.duplicated().sum())

df.drop_duplicates(inplace=True)
```

```
    # Проверка количества строк после обработки пропусков
    print(f"\nКоличество строк после обработки пропусков: {len(df)}")
    # Удаление отрицательных значений в указанных столбцах
    negative_cols = ['level', 'levels', 'area', 'kitchen_area']
    for col in negative_cols:
        if col in df.columns:
            initial_count = len(df)
            df = df[df[col] >= 0]
            removed_count = initial_count - len(df)
            print(f"Удалено {removed_count} строк с отрицательными значениями в столбце '{col}'")
            # Особый случай для столбца 'rooms' (значение -1 означает студию)
    if 'rooms' in df.columns:
        initial_count = len(df)
        # Сохраняем студии (-1) и квартиры с 1+ комнатами
        df = df[(df['rooms'] == -1) | (df['rooms'] >= 1)]
        removed_count = initial_count - len(df)
        print(f"\nУдалено {removed_count} строк с некорректным количеством комнат (оставлены только студии (-1) и квартиры с 1+ комнатами)")
     #Фильтрация аномальных значений площади
    if 'area' in df.columns and 'rooms' in df.columns:
        # Максимальная разумная площадь (300 кв.м)
        max_area = 300
        initial_count = len(df)
        # Для студий (-1) считаем как 1 комнату при проверке площади
```

```
        df = df[(df['area'] <= max_area) & ((df['rooms'] == -1) | (df['area'] <= df['rooms'] * max_area))]

        removed_count = initial_count - len(df)

        print(f"\nУдалено {removed_count} строк с аномально большой площадью")    # Фильтрация очень маленьких площадей (менее 15 кв.м)

        initial_count = len(df)

        df = df[df['area'] >= 15]

        removed_count = initial_count - len(df)

        print(f"Удалено {removed_count} строк с аномально маленькой площадью")

    # Обработка выбросов цены

    if 'price' in df.columns:

        # Верхний предел цены (99-й перцентиль)

        price_upper_limit = df['price'].quantile(0.99)

        initial_count = len(df)

        df = df[df['price'] <= price_upper_limit]

        removed_count = initial_count - len(df)

        print(f"\nУдалено {removed_count} строк с ценами выше {price_upper_limit:,.0f} руб. (99-й перцентиль)")

        # Нижний предел цены

        price_lower_limit = 400000

        initial_count = len(df)

        df = df[df['price'] > price_lower_limit]

        removed_count = initial_count - len(df)

        print(f"Удалено {removed_count} строк с ценами ниже {price_lower_limit:,.0f} руб.")

    # Проверка количества строк после удаления выбросов

    print(f"\nКоличество строк после обработки выбросов: {len(df)}")

    # Фильтрация данных по 28 региону
```

```
    df = df[df['id_region'] == 28]

    # Итоговые данные

    print(f"\nИтоговое количество строк: {len(df)}")

    print(f"Итоговое количество столбцов: {len(df.columns)}")

    # Сохранение результатов

    df.to_csv('preprocessed_data_28_region.csv', index=False)

    print("\nПредобработанные данные сохранены в 'preprocessed_data_28_region.csv'")

    from sklearn.preprocessing import MinMaxScaler

    from tabulate import tabulate


    # Выбираем числовые колонки для нормализации

    numeric_cols = df.select_dtypes(include=['int64', 'float64']).columns

    numeric_cols = [col for col in numeric_cols if col not in ['id_region']]

    # Сохраняем копию данных до нормализации

    df_before = df[numeric_cols].copy()

    # Применяем MinMaxScaler

    scaler = MinMaxScaler()

    df[numeric_cols] = scaler.fit_transform(df[numeric_cols])

    # Выбираем 5 случайных строк для сравнения

    sample_rows = df_before.sample(5, random_state=42)

    sample_rows_scaled = df.loc[sample_rows.index, numeric_cols]

    # Создаем таблицы для сравнения

    comparison = []

    for idx in sample_rows.index:
```

```
        original_values = sample_rows.loc[idx].values
        scaled_values = sample_rows_scaled.loc[idx].values
        comparison.append([idx] + list(original_values) + list(scaled_values))
    # Заголовки для таблицы
    headers = ["Index"] + [f"{col} (до)" for col in numeric_cols] + [f"{col} (после)" for col in numeric_cols]
    # Выводим сравнение в рамочке
    print("\n╔═══════════════════════════════════════════════╗")
    print("║    СРАВНЕНИЕ ДО И ПОСЛЕ НОРМАЛИЗАЦИИ ║")
    print("╠═══════════════════════════════════════════════╣")
    print(tabulate(comparison, headers=headers, tablefmt="pretty", floatfmt=".4f"))
    print("╚═══════════════════════════════════════════════╝")
    # Сохранение предобработанных данных
    df.to_csv('preprocessed_data_28_region.csv', index=False)
    # Итоговое количество строк и столбцов
    print(f"\nИтоговое количество строк после фильтрации по 28 региону: {len(df)}")
    print(f"Итоговое количество столбцов: {len(df.columns)}")
    print("\nПредобработанные данные по 28 региону сохранены в файл 'preprocessed_data_28_region.csv'")# Итоговое количество строк и столбцов
    print(f"\nИтоговое количество строк после фильтрации по 28 региону: {len(df)}")
    print(f"Итоговое количество столбцов: {len(df.columns)}")
    print("\nПредобработанные данные по 28 региону сохранены в файл 'preprocessed_data_28_region.csv'")
    # Фильтрация данных по 28 региону
```

```
df_28 = df[df['id_region'] == 28]
# 1. Гистограмма цен
plt.figure(figsize=(10, 6))
sns.histplot(df_28['price'], bins=30, kde=True, color='blue')
plt.title('Распределение цен на жилье в 28 регионе')
plt.xlabel('Цена')
plt.ylabel('Количество объектов')
plt.show()
#  Зависимость цены от количества комнат
plt.figure(figsize=(12, 8))  # Размер графика
sns.scatterplot(x='rooms', y='price', data=df_28, color='blue', alpha=0.6)  # alpha - прозрачность точек
plt.title('Зависимость цены от количества комнат в 28 регионе', fontsize=16)
plt.xlabel('Количество комнат', fontsize=14)
plt.ylabel('Цена', fontsize=14)
plt.grid(True)  # Добавляем сетку для удобства
plt.show()
# 3. График цены и площади
plt.figure(figsize=(10, 6))
sns.scatterplot(x='area', y='price', data=df_28, hue='rooms', palette='viridis', alpha=0.7)
plt.title('Зависимость цены от площади в 28 регионе')
plt.xlabel('Площадь')
plt.ylabel('Цена')
plt.legend(title='Количество комнат')
plt.show()
# 4. Распределение типов зданий
plt.figure(figsize=(10, 6))
```

```
    sns.countplot(x='building_type',   data=df_28,   pal-
ette='viridis')

    plt.title('Распределение типов зданий в 28 регионе')

    plt.xlabel('Тип здания')

    plt.ylabel('Количество объектов')

    plt.xticks(rotation=45)

    plt.show()

    # 5. График цены и уровня этажа

    plt.figure(figsize=(10, 6))

    sns.scatterplot(x='level',   y='price',   data=df_28,
hue='rooms', palette='viridis', alpha=0.7)

    plt.title('Зависимость цены от уровня этажа в 28 ре-
гионе')

    plt.xlabel('Уровень этажа')

    plt.ylabel('Цена')

    plt.legend(title='Количество комнат')

    plt.show()

    import seaborn as sns   # Seaborn — это библиотека для
создания статистических графиков на Python

    XX   =   df[['level',   'levels',   'rooms',   'area',
'kitchen_area',  'geo_lat',  'geo_lon',  'building_type',
'object_type', 'id_region']]

    plt.figure(figsize = (10,8))

    sns.heatmap(XX.corr())

    print('Коэффициент корреляции между признаками level
и levels', XX.corr().loc['level', 'levels'])

    print('Коэффициент   корреляции   между   признака-
миroomsиarea', XX.corr().loc['rooms', 'area'])
```

ПРИЛОЖЕНИЕ Б

Листинг программы для модели линейной регрессии

```
import pandas as pd
from sklearn.model_selection import train_test_split
import numpy as np
# 1. Загрузка данных
# data = pd.read_csv('preprocessed_data.csv')
# 2. Разделение данных на обучающую и тестовую выборки
train, test = train_test_split(df, test_size=0.2, ran-
dom_state=42)
# 3. Определение признаков и целевой переменной
features = ['level', 'levels', 'rooms', 'area',
'kitchen_area', 'geo_lat', 'geo_lon', 'building_type',
'object_type', 'id_region']
target = 'price'
# 4. Выделение данных для обучения и преобразование в
массивы numpy
x_train = train[features].values
y_train = train[target].values
x_test = test[features].values
y_test = test[target].values
# 5. Проверка размеров данных
print(f"\nРазмерx_train: {x_train.shape}")
print(f"Размеру_train: {y_train.shape}")
print(f"Размерx_test: {x_test.shape}")
print(f"Размеру_test: {y_test.shape}")
from sklearn.metrics import mean_absolute_error,
mean_squared_error, r2_score    # импорт метрик
# масштабирование признаков
```

```
    from sklearn.preprocessing import StandardScaler,
MinMaxScaler

    scaler = MinMaxScaler()

    XX_train = scaler.fit_transform(x_train)

    XX_test = scaler.transform(x_test)# вычисляем предска-
зание

    regressor_M = LinearRegression()

    regressor_M.fit(XX_train, y_train)# обучение модели

    test_predictions_M          =          regressor_M.pre-
dict(XX_test)#вычисляем предсказание

    print('test mae score: ', mean_absolute_error(y_test,
test_predictions_M))

    print('test    mse:    ',    mean_squared_error(y_test,
test_predictions_M))

    print('test r2 score: ', r2_score(y_test, test_predic-
tions_M))

    plt.figure(figsize=(20, 8))

    plt.bar(XX.columns, regressor_M.coef_)# bar() - для
построения вертикальной диаграммы

    for i in range(len(XX.columns)):

        print((XX.columns[i], regressor_M.coef_[i]))

    error = test_predictions_M  -y_test

    plt.hist(error, bins = 50)

    plt.xlabel("Значениео шибки, руб")

    plt.ylabel("Количество")

    plt.show()
```

ПРИЛОЖЕНИЕ В

Листинг программы для модели линейной регрессии

```
from tensorflow.keras.models import Sequential    # Импорт
последовательной модели из библиотеки моделей
from tensorflow.keras.layers import Dense, Dropout
model = Sequential()
model.add(Dense(64,         activation='relu',         in-
put_shape=(x_train.shape[1],)))
model.add(Dense(128, activation='relu'))
model.add(Dense(256, activation='relu'))
model.add(Dense(512, activation='relu'))
model.add(Dense(1024, activation='relu'))
model.add(Dense(2048, activation='relu'))
model.add(Dense(1, activation='relu'))
print(model.summary())
model.compile(optimizer='adam',      loss='mse',      met-
rics=['mae'])
history = model.fit(x_train,
                    y_train,
                    batch_size = 1500,
                    epochs=280,
                    validation_split=0.2,
                    verbose=1)
plt.figure(figsize=(8, 6))
plt.plot(history.history['mae'],
         label='Средняя абсолютная ошибка на обучающем
наборе')
plt.plot(history.history['val_mae'],
         label='Средняя абсолютная ошибка на проверочном
наборе')
```

```
plt.xlabel('Эпоха обучения')
plt.ylabel('Средняя абсолютная ошибка')
plt.legend()
plt.show()
scores = model.evaluate(x_test, y_test, verbose=1)
print("Средняя абсолютная ошибка на тестовых данных:",
round(scores[1], 4))
pred = model.predict(x_test).flatten()
# Фильтрация y_test и pred, чтобы y_test > 0
valid_indices = y_test > 0  # Булев массив, где y_test
положительные
y_test_filtered = y_test[valid_indices]
pred_filtered = pred[valid_indices]
# Вычисление относительной ошибки
relative_errors = abs(y_test_filtered - pred_filtered) /
y_test_filtered * 100
# Вывод результатов
print('Минимальная относительная ошибка -',
np.min(relative_errors), '%.')
print('Максимальная относительная ошибка -',
np.max(relative_errors), '%.')
print('Средняя относительная ошибка -', np.mean(rela-
tive_errors), '%.')
print('test mae score: ', mean_absolute_error(y_test,
pred))
print('test mse: ', mean_squared_error(y_test, pred))
print('test r2 score: ', r2_score(y_test, pred))
# Получаем предсказания для тестовой выборки
pred = model.predict(x_test).flatten()
# Выводим метрики качества
```

```
print('\nОценка качества модели на тесте:')
print('Test MAE:', mean_absolute_error(y_test, pred))
print('Test MSE:', mean_squared_error(y_test, pred))
print('Test R2 score:', r2_score(y_test, pred))
# Создаем DataFrame для наглядного сравнения
results = pd.DataFrame({
    'Реальное значение': y_test,
    'Предсказанное значение': pred,
    'Абсолютная ошибка': abs(y_test - pred),
    'Относительная ошибка (%)': abs(y_test - pred) / y_test
* 100
})
# Выводим первые 26 строк для примера
print('\nПримеры предсказаний:')
print(results.head(26))
plt.figure(figsize=(8, 8))
plt.scatter(y_test, pred)
plt.xlabel('Правильные значение, руб')
plt.ylabel('Предсказания, руб')
plt.axis('equal')
plt.xlim(plt.xlim())
plt.ylim(plt.ylim())
plt.plot([0, 60000000], [0, 60000000], 'r')
plt.show()
```

ПРИЛОЖЕНИЕ Г

Диплом II степени XXXIV научной конференции «День науки»

Рисунок Г.1 – Диплом II степени