МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное автономное образовательное учреждение
высшего образования «Самарский национальный исследовательский
университет имени академика С.П. Королева»
(Самарский университет)

Институт информатики и кибернетики

Кафедра программных систем

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

«ВЕБ-ПРИЛОЖЕНИЕ ПРОГНОЗИРОВАНИЯ СТОИМОСТИ ЛЕГКОВОГО АВТОМОБИЛЯ»

по направлению подготовки 02.03.02

Фундаментальная информатика и информационные технологии

(уровень бакалавриата)

направленность (профиль) «Информационные технологии»

Обучающийся Д.О. Колбанов
(подпись, дата)

Руководитель ВКР
к.т.н., доцент О.А. Гордеева
(подпись, дата)

Нормоконтролер Е.В. Сопченко
(подпись, дата)

Самара 2025

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное
образовательное учреждение высшего образования
«Самарский национальный исследовательский университет имени
академика С.П. Королева»

Кафедра программных систем

УТВЕРЖДАЮ
Заведующий кафедрой

_____С.В. Востокин
«____» _____2025 г.

задание на выпускную квалификационную работу (ВКР)

обучающемуся Колбанову Дмитрию Олеговичу
группы 6401-020302D

1. Тема работы: Веб-приложение прогнозирования стоимости легкового автомобиля

утверждена приказом по университету от «24» апреля 2025 г. № 223-Т

2. Перечень вопросов, подлежащих разработке в ВКР:
1) Провести анализ предметной области: прогнозирование стоимости автомобиля, регрессионный анализ, машинное обучение, регрессия, градиентный бустинг
2) Сделать обзор систем-аналогов в области систем прогнозирования стоимости автомобилей
3) Разработать проект системы с использованием методологии UML
4) Провести сравнительный анализ примененных алгоритмов прогнозирования стоимости легкового автомобиля
5) Разработать и реализовать информационное и программное обеспечение
6) Провести тестирование и отладку разработанного веб-приложения
7) Провести исследование влияния наличия повреждений у автомобиля на результат прогноза

3. Дата выдачи задания: «24» апреля 2025г.
4. Срок представления на кафедру законченной ВКР: «5» июня 2025г.

Руководитель ВКР
доцент кафедры программных систем,
к.т.н., доцент _____О.А. Гордеева
«_24_» ____04____ 2025 г.

Задание принял к исполнению _____Д.О. Колбанов
«_24_» ____04____ 2025 г.

РЕФЕРАТ

Пояснительная записка 56 с, 12 рисунков, 4 таблицы, 21 источник, 2 приложения.

Графическая часть: 24 слайда презентации PowerPoint.

ПРОГНОЗИРОВАНИЕ СТОИМОСТИ АВТОМОБИЛЯ, РЕГРЕССИОННЫЙ АНАЛИЗ, МАШИННОЕ ОБУЧЕНИЕ, РЕГРЕССИЯ, ГРАДИЕНТНЫЙ БУСТИНГ.

Цель работы – разработать веб-приложение для прогнозирования стоимости легкового автомобиля с использованием нескольких алгоритмов машинного обучения и провести сравнительный анализ их эффективности.

В процессе работы были разработаны алгоритмы и соответствующая программа, позволяющая пользователю получить прогноз рыночной стоимости легкового автомобиля на основе входных параметров. Система выполняет предобработку данных, рассчитывает стоимость, предоставляет результаты в текстовом виде.

Клиентская часть системы разработана на языке JavaScript с использованием библиотеки Vue.js. Серверная часть системы разработана на языке Python с использованием фреймворка Flask.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В настоящее время рынок подержанных и новых легковых автомобилей характеризуется высокой динамичностью и большим разнообразием моделей, что усложняет процесс объективной оценки их стоимости [1]. Одновременно с этим широкое распространение современных веб-технологий и методов машинного обучения открывает новые возможности для автоматизации процесса прогнозирования цен, повышая прозрачность рынка и помогая как продавцам, так и покупателям принимать более обоснованные решения.

Прогнозирование стоимости автомобиля представляет собой задачу регрессионного анализа, где в качестве входных признаков используются технические и эксплуатационные характеристики (марка, модель, год выпуска, пробег, тип двигателя и др.). Традиционные методы ручной оценки требуют глубоких экспертных знаний и существенно зависят от субъективного опыта оценщика. Автоматизированная система на основе машинного обучения позволяет построить модель, учитывающую многомерность данных и выявляющую сложные взаимосвязи между признаками, что повышает точность прогноза и скорость обработки запросов.

Практическая значимость работы заключается в создании доступного и наглядного инструмента, который может быть использован автодилерами, сервисами частных объявлений и конечными пользователями для быстрой оценки рыночной стоимости автомобиля. Полученные результаты могут быть внедрены в информационно-аналитические системы автокомпаний и онлайн-площадок для повышения конкурентоспособности и доверия клиентов.

В процессе работы необходимо разработать алгоритмы и соответствующую программу, позволяющую пользователю получить прогноз рыночной стоимости легкового автомобиля на основе входных параметров. Система должна выполнять предобработку данных, рассчитывать стоимость, предоставлять результаты в текстовом виде.

1 Описание и анализ предметной области

1.1 Основные понятия и определения

Прогнозирование рыночной стоимости автомобиля — процесс определения будущей или текущей стоимости транспортного средства на основе анализа исторических данных, рыночных тенденций и технических характеристик [2]. Прогноз строится с использованием математических моделей и алгоритмов машинного обучения.

Регрессионный анализ — это статистический метод, позволяющий исследовать связь переменных. Основная цель регрессионного анализа — предсказать значение одной переменной на основе одной или нескольких других переменных. При этом регрессионный анализ помогает выявить, насколько тесно эти переменные связаны и как одна из них влияет на другую [3]. В задачах прогнозирования стоимости автомобилей регрессионный анализ позволяет установить связь между характеристиками автомобиля (например, пробег, год выпуска, марка) и его ценой.

Машинное обучение — это область знаний и исследований в области искусственного интеллекта, которая занимается разработкой алгоритмов и статистических моделей, которые могут аппроксимировать данные, обучаться на них, обобщать их на невидимые зависимости и, таким образом, выполнять задачи без явных инструкций [4].

Модель машинного обучения — это объект (хранящийся локально в файле), который был обучен для распознавания определенных типов шаблонов. Модель обучается на основе набора данных по предоставленному ей алгоритму, который она может использовать для анализа и обучения на основе этих данных [5].

Регрессия в машинном обучении представляет собой тип задач обучения с учителем, при котором модель обучается на основе обучающего набора данных, а целью является прогнозирование непрерывного числового значения на основе одной или нескольких независимых характеристик [6]. Прогнозирование стоимости легкового автомобиля является задачей регрессии.

1.2 Актуальность задачи

Прогнозирование рыночной стоимости автомобиля является одной из ключевых задач в современной автомобильной индустрии и секторах, связанных с продажей, страхованием и кредитованием автотранспорта. Рынок подержанных автомобилей отличается высокой динамичностью: на цену влияет широкий спектр факторов — возраст и пробег машины, марка и модель, географическое расположение, сезонность, техническое состояние, наличие ДТП, а также колебания спроса и предложения. В условиях растущей конкуренции и всё более требовательных потребителей своевременная и точная оценка стоимости позволяет продавцам и покупателям принимать обоснованные решения, минимизировать финансовые риски и повышать прозрачность сделок на вторичном рынке.

В России покупка автомобиля на вторичном рынке приобрела особую значимость из-за политической и экономической нестабильности, санкций и ограниченного предложения новых моделей у официальных дилеров. Недостаток выбора на первичном рынке вынуждает потребителей обращаться к б/у автомобилям, что делает задачу автоматизированного и объективного прогнозирования цен ещё более востребованной.

Существуют различные подходы к оценке стоимости автомобилей: от экспертных систем и аналитики исторических объявлений до машинного обучения и нейронных сетей, обрабатывающих большие объёмы данных. Однако многие существующие решения либо требуют финансовых вложений, либо ограничены в удобстве использования. В связи с этим разработка веб-приложения, объединяющего в себе современные модели прогнозирования, автоматизированный сбор и предобработку данных, а также удобный интерфейс для различных категорий пользователей, является актуальной и востребованной задачей как в научно-исследовательской, так и в практической плоскости.

1.3 Описание систем-аналогов

В настоящее время существуют различные сервисы для прогнозирования стоимости автомобиля, предоставляющие различный функционал. Каждая

система имеет свои достоинства и недостатки. Рассмотрим некоторые из них.

1.3.1 Сервис Avito Оценка

Сервис Avito Оценка [7] предоставляет пользователю встроенный в площадку Avito онлайн-инструмент для быстрой и бесплатной оценки рыночной стоимости автомобиля. При переходе в карточку любого объявления система автоматически анализирует параметры конкретного лота — марку, модель, год выпуска, пробег, регион и комплектующие опции — и сравнивает их с тысячами совершённых сделок на той же площадке. Благодаря интеграции непосредственно в интерфейс Avito, пользователь получает рекомендации по справедливой цене без необходимости вручную вводить все данные и переключаться на сторонние сайты.

Результат оценки выводится в виде цветовой метки и текста непосредственно в списке объявлений и на странице лота: пользователь видит, попадает ли цена в «зону справедливой стоимости» или нет, а при клике может посмотреть график средней и разброса цен по аналогам. Это позволяет сразу ориентироваться на выгодные предложения.

Результат прогнозирования стоимости автомобиля, полученный данным сервисом представлен на рисунке 1.

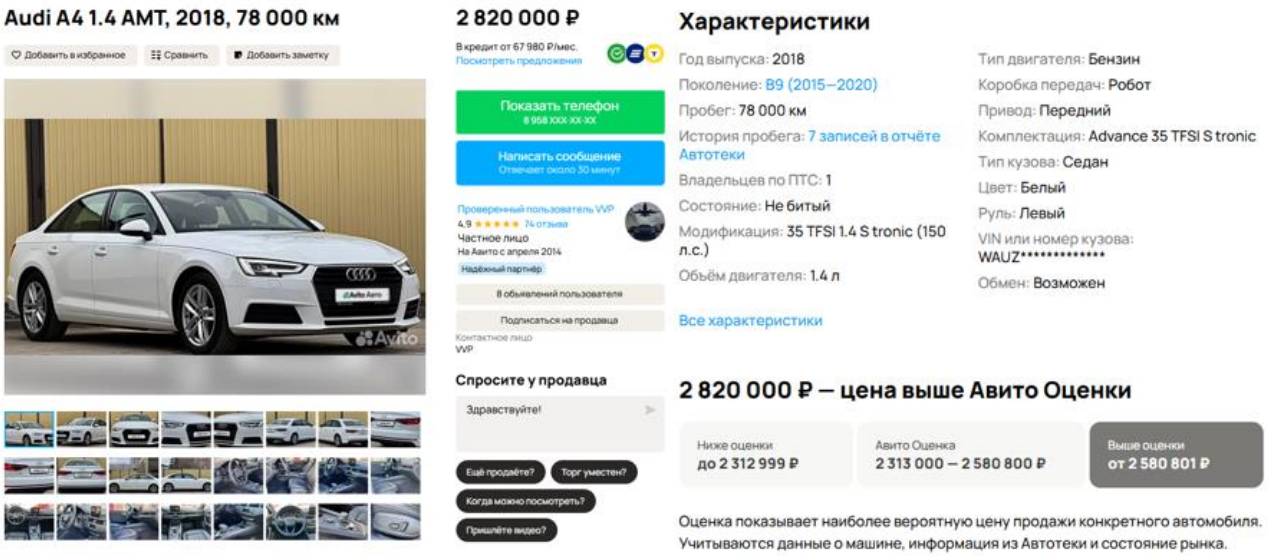

Рисунок 1 – Пример результата прогнозирования стоимости автомобиля сервисом Avito Оценка

1.3.2 Сервис Auto.ru «Оценка»

Сервис Auto.ru «Оценка» [8] — это бесплатный веб-калькулятор, доступный на портале Auto.ru, предназначенный для моментальной оценки б/у автомобилей. Пользователь вводит либо госномер автомобиля, либо вручную основные параметры (марка, модель, год, пробег, тип кузова и комплектация), после чего система автоматически подбирает аналоги по базе объявлений и сделок площадки. Для удобства доступен быстрый ввод VIN-кода, что позволяет получить более точные данные по комплектации.

Результат работы выводится в виде конкретного диапазона рыночной цены и «справедливой стоимости» в процентах от среднего значения по аналогам. Пользователь может сформировать и скачать PDF-отчёт с подробной разбивкой: график динамики цен, таблицу сравнения аналогов и выводы экспертов Auto.ru.

Недостатком данного сервиса является платный доступ к расширенным функциям анализа.

На рисунке 2 представлен пример прогнозирования стоимости автомобиля, полученный данным сервисом.

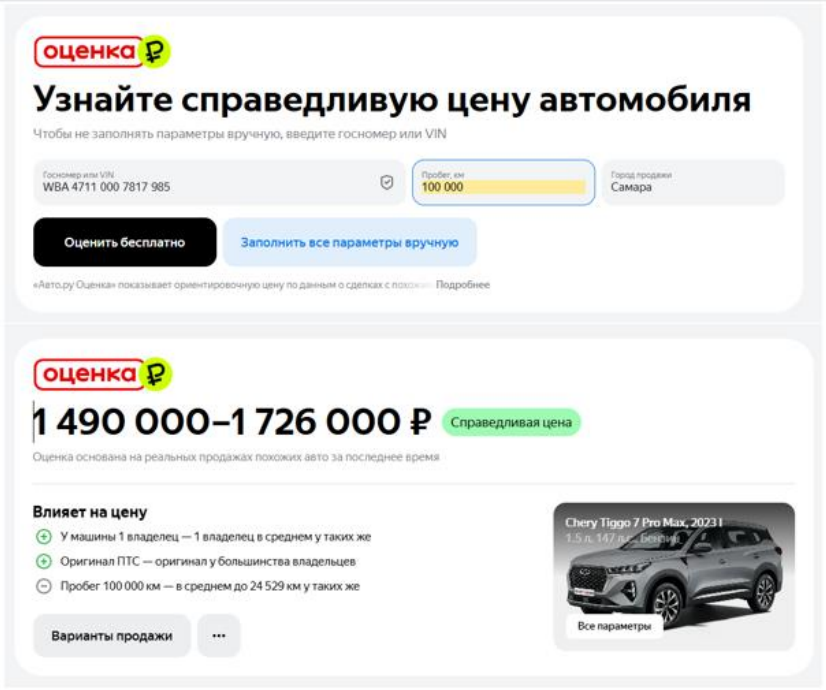

Рисунок 2 – Пример результата прогнозирования стоимости автомобиля сервисом Auto.ru «Оценка»

1.3.3 Конкурентный анализ систем-аналогов

Проведем сравнительный анализ для вышеописанных сервисов прогнозирования стоимости автомобиля, выделив их основные отличительные особенности, см. таблицу 1.

Таблица 1 – Сравнительная характеристика систем-аналогов

Показатель	Система		Требования к разрабатываемой системе
	Avito Оценка	Auto.ru «Оценка»	
Полностью бесплатное использование	+	–	+
Использование машинного обучения	–	+	+
Удобный интерфейс	+	+	+
Ввод характеристик автомобиля	–	+	+
Экспорт данных	–	+	–

1.4 Описание автоматизируемого процесса

В рамках выпускной квалификационной работы необходимо разработать веб-приложение прогнозирования стоимости легкового автомобиля.

К основным этапам разработки веб-приложений с использованием алгоритмов машинного обучения относятся следующие: разработка модели прогнозирования, проектирование баз данных, разработка клиентской части, разработка серверной части.

Разработка модели прогнозирования включает в себя сбор, анализ и

обработку данных для обучения, а также непосредственно обучение нескольких различных моделей машинного обучения, подбор гипер-параметров и выбор наилучшего алгоритма.

Разработка клиентской части приложения включает в себя такие задачи как разработка интерфейса пользователя и его прототипов. Необходимо учитывать, что интерфейс пользователя является важной частью системы, так как именно через работу с интерфейсом пользователь может взаимодействовать с системой.

Разработка серверной части приложения включает в себя следующие подзадачи: реализация основной бизнес-логики системы, разработка способов взаимодействия сервера с клиентом и базой данных. В рамках автоматизируемого процесса прогнозирования стоимости легкового автомобиля, необходимо реализовать предварительную обработку данных, а также расчет прогноза стоимости.

Для решения задачи прогнозирования стоимости легкового автомобиля и последующего исследования были выбраны линейная регрессия, k-ближайших соседей, случайный лес и градиентный бустинг. Этап исследования включает в себя вычисление метрики, отражающей точность прогнозирования стоимости – MAPE.

1.5 Постановка задачи

Цель работы: во время выпускной квалификационной работы необходимо разработать веб-приложение прогнозирования стоимости легкового автомобиля.

Задачи:

- провести анализ предметной области;
- сделать обзор систем-аналогов в области прогнозирования стоимости автомобилей;
- собрать данные для обучения моделей машинного обучения;
- провести сравнительный анализ примененных алгоритмов прогнозирования стоимости легкового автомобиля;

– разработать и реализовать информационное и программное обеспечение;

– провести тестирование и отладку разработанного веб-приложения.

Разрабатываемое веб-приложение должно выполнять следующие функции:

1) серверная часть системы:
   – предобработка данных, полученных с клиентской части;
   – прогнозирование стоимости автомобиля моделью машинного обучения;
2) клиентская часть системы:
   – ввод характеристик автомобиля;
   – валидация параметров, введенных пользователем;
   – просмотр результатов прогноза.

1.6 Выводы по главе

В данной главе был произведен анализ предметной области: изучены и описаны основные определения в области прогнозирования стоимости легкового автомобиля, актуальность исследования.

Произведен обзор существующих систем-аналогов, сформулирована постановка задачи ВКР и определены основные функции разрабатываемой системы.

2 Проектирование системы

2.1 Описание методов прогнозирования

Существует большое количество методов машинного обучения, решающих задачу регрессии. Были взяты несколько различных моделей, обучены на подготовленных данных и оценены по метрике MAPE (Mean Absolute Percentage Error). Ниже рассмотрим каждый алгоритм:

2.1.1 Линейная регрессия

Линейная регрессия (Linear regression) — один из простейший алгоритмов машинного обучения, описывающий зависимость целевой переменной от признака в виде линейной функции y = kx + b [9]. В данном случае была представлена простая или парная линейная регрессия, а уравнение вида

$$f_{w,b}(x) = w_0x_0 + w_1x_1 + \cdots + w_nx_n + b = w \cdot x + b$$

называется множественной линейной регрессией, где b — смещение модели, w — вектор её весов, а x — вектор признаков одного обучающего образца.

Выбор регрессионной линии (плоскости), описывающей взаимосвязь данных наилучшим образом, заключается в минимизации функции потерь, представленной в виде среднеквадратичной ошибки. Проще говоря, линия должна проходить через данные таким образом, чтобы в среднем разница квадратов ожидаемых и реальных значений была минимальна. Данный метод называется методом наименьших квадратов.

Линейная регрессия обучается с помощью итеративной оптимизации с постепенным снижением ошибки модели на основе градиентного спуска и его разновидностей.

2.1.2 Градиентный спуск

Градиентный спуск — это итеративный алгоритм оптимизации, используемый для минимизации функции, чаще всего функции потерь в контексте машинного обучения. Он работает путем нахождения направления, в котором функция потерь уменьшается наиболее быстро, и делает шаги в этом направлении для постепенного уменьшения значения функции потерь [10].

Градиент функции — это вектор, состоящий из частных производных, который указывает направление наискорейшего роста функции. В контексте оптимизации, мы интересуемся направлением наискорейшего убывания, то есть движемся в противоположном направлении градиента. Частные производные вычисляются для каждого параметра модели.

На каждой итерации алгоритма параметры модели обновляются в направлении, противоположном градиенту функции потерь. Размер шага, который делает алгоритм в этом направлении, определяется скоростью обучения (learning rate). Оптимальная скорость обучения — ключевой параметр, поскольку слишком большой шаг может привести к пропуску минимума, а слишком маленький делает процесс оптимизации медленным.

Формула для обновления параметра θ на каждой итерации выглядит следующим образом:

$$\theta = \theta - \eta \cdot \nabla_\theta \cdot J(\theta),$$

где η - скорость обучения, а $\nabla_\theta J(\theta)$ - градиент функции потерь J по параметру θ.

2.1.3 Регуляризация линейной регрессии

Регуляризация в статистике, машинном обучении, теории обратных задач — метод добавления некоторых дополнительных ограничений к условию с целью решить некорректно поставленную задачу или предотвратить переобучение. Чаще всего эта информация имеет вид штрафа за сложность модели [11].

В линейной регрессии методы регуляризации работают путем добавления штрафных коэффициентов к исходной функции потерь модели таким образом, что высокие значения коэффициентов снижаются. А признаки с очень низкими значениями коэффициентов (после штрафования) могут быть вообще отброшены. Это помогает уменьшать сложность модели. Рассмотрим Ridge и Lasso регуляризации.

Гребневая регрессия (Ridge regression) или регуляризация Тихонова применяется в случае мультиколлинеарности через добавление L2-

регуляризации к функции потерь во время обучения и сильнее всего занижает веса для признаков с высокой корреляцией: их значения будут приближаться к нулю, но никогда его не достигнут.

$$L2 - \text{регуляризация} = \lambda \sum_{i=1}^{n} w_i^2.$$

Лассо-регрессия (Lasso regression или Least Absolute Shrinkage & Selection Operator) обычно используется для отбора признаков через добавление L1-регуляризации к функции потерь во время обучения. Проще говоря, лассо-регрессия стремится уменьшить число параметров модели путем зануления весов для неинформативных и избыточных признаков, что на выходе даст разреженную модель (с небольшим числом ненулевых весов признаков).

$$L1 - \text{регуляризация} = \lambda \sum_{i=1}^{n} |w_i|,$$

где λ (лямбда) — это гиперпараметр, который контролирует силу регуляризации. Он выбирается заранее и может быть настроен в процессе обучения модели, n - количество признаков в модели, w_i - вес (коэффициент) i-го признака.

2.1.4 Метод ближайших соседей

Метод ближайших соседей (KNN) в задаче регрессии — это непараметрический метод, используемый для прогнозирования непрерывных значений [12].

Основная идея заключается в прогнозировании целевого значения для новой точки данных путём усреднения целевых значений K ближайших соседей в пространстве объектов. Расстояние между точками данных обычно измеряется с использованием евклидова расстояния, хотя могут использоваться и другие показатели расстояния.

Работа KNN-регрессии включает несколько шагов:

– выбор количества соседей (K). Этот выбор сильно влияет на производительность модели. Меньшее значение K делает модель более

подверженной шуму, в то время как большее значение K приводит к более плавным прогнозам;

– вычисление расстояний. Для новой точки данных вычисляют расстояние между этой точкой и всеми точками в обучающем наборе;

– поиск K ближайших соседей. Определяют K точек в обучающем наборе, которые находятся ближе всего к новой точке данных;

– прогнозирование целевого значения. Вычисляют среднее значение целевых значений K ближайших соседей и используют это в качестве прогнозируемого значения для новой точки данных.

2.1.5 Случайный лес

Решающие деревья являются хорошим семейством базовых классификаторов для бэггинга, поскольку они достаточно сложны и могут достигать нулевой ошибки на любой выборке. Метод случайных подпространств позволяет снизить коррелированность между деревьями и избежать переобучения. Базовые алгоритмы обучаются на различных подмножествах признакового описания, которые также выделяются случайным образом [13].

Алгоритм построения случайного леса, состоящего из N деревьев, выглядит следующим образом: для каждого n = 1, …, N:

1) сгенерировать выборку X_n с помощью бутстрэпа;
2) построить решающее дерево b_n по выборке X_n:

– по заданному критерию мы выбираем лучший признак, делаем разбиение в дереве по нему и так до исчерпания выборки;

– дерево строится, пока в каждом листе не более n_{min} объектов или пока не достигнем определенной высоты дерева;

– при каждом разбиении сначала выбирается m случайных признаков из n исходных, и оптимальное разделение выборки ищется только среди них.

$$a(x) = \frac{1}{N}\sum_{i=1}^{N} b_i(x).$$

2.1.6 Градиентный бустинг

Бустинг – это ансамблевый метод машинного обучения, целью которого является объединение нескольких слабых моделей предсказания для создания одной сильной. Слабая модель – это такая, которая выполняет предсказания немного лучше, чем наугад, в то время как сильная модель обладает высокой предсказательной способностью. Цель бустинга – улучшить точность предсказаний [14].

Бустинг работает путём последовательного добавления моделей в ансамбль. Каждая следующая модель строится таким образом, чтобы исправлять ошибки, сделанные предыдущими моделями. Это достигается путём фокусировки на наиболее проблемных данных, которые были неверно классифицированы или предсказаны ранее.

Одной из основных особенностей бустинга является динамическое взвешивание обучающих данных. После каждого этапа обучения модели в ансамбле, данные, на которых были допущены ошибки, получают больший вес. Это означает, что последующие модели уделяют больше внимания именно этим трудным случаям.

Когда используются решающие деревья, каждое последующее дерево строится с учетом ошибок, сделанных предыдущими деревьями. Новые деревья учатся на ошибках, улучшая общую точность ансамбля.

Несмотря на свою эффективность, бустинг может быть склонен к переобучению, особенно если в ансамбле слишком много моделей или они слишком сложные. Для контроля переобучения используется ранняя остановка (early stopping).

2.1.7 Принцип построения ансамбля

Ансамбль в градиентном бустинге обычно состоит из последовательности слабых предсказательных моделей. Чаще всего используются решающие деревья из-за их способности моделировать нелинейные зависимости и взаимодействия между признаками. Каждое новое дерево в ансамбле строится так, чтобы уменьшить оставшуюся ошибку предыдущих деревьев [15].

В градиентном бустинге каждая следующая модель обучается с учетом ошибок, допущенных всеми предыдущими моделями в ансамбле. Это достигается путем фокусировки на самых трудных для предсказания случаях, которые были неправильно классифицированы или предсказаны ранее.

Суть метода заключается в том, что веса для каждого наблюдения в обучающем наборе данных корректируются на каждом шаге. Наблюдения, которые были неправильно предсказаны предыдущей моделью, получают больший вес, тем самым увеличивая вероятность их правильного предсказания последующими моделями.

2.1.8 Реализация градиентного бустинга в CatBoost

CatBoost использует решающие деревья глубины 1 или 2 в качестве базовых моделей [16]. Эти неглубокие деревья имеют следующие характеристики:

– каждый узел дерева делает бинарное разбиение на основе значения одной из признаков;

– эти короткие деревья обладают небольшой глубиной, что делает их более устойчивыми к переобучению.

CatBoost реализует механизм регуляризации для предотвращения переобучения модели. В частности, используется L2-регуляризация, модифицирующая функцию потерь добавлением штрафного члена. Это способствует повышению обобщающей способности модели и снижению риска переобучения, особенно при большом количестве признаков или их высокой корреляции.

Кроме того, CatBoost автоматически осуществляет отбор признаков на основе оценки их важности для модели. Это позволяет модели сосредоточиться на наиболее информативных признаках и уменьшить шум от менее значимых. В CatBoost используется градиентный бустинг для обучения ансамбля решающих деревьев. Этот метод минимизирует функцию потерь с помощью градиентного спуска, постепенно улучшая точность модели.

На каждой итерации градиентного бустинга к ансамблю добавляется новое решающее дерево, предназначенное для коррекции ошибок, допущенных предыдущими моделями.

2.1.9 Метрика качества

В качестве метрики для сравнения использованных алгоритмов использовалась MAPE, выражающая среднее абсолютное отклонение прогнозируемых значений от фактических значений в процентах, что делает эту метрику очень наглядной для интерпретации результатов [17].

Формула MAPE определяется как:

$$MAPE(y^{true}, y^{pred}) = \frac{1}{N}\sum_{i=1}^{N}\frac{|y_i - f(x_i)|}{|y_i|} \times 100\%$$

где:

N – количество наблюдений,

y_i – фактическое значение целевой переменной,

x_i – вектор признаков,

$f(x_i)$ – прогнозируемое значение, полученное моделью,

$|y_i - f(x_i)|$ – абсолютная ошибка прогноза для (i)-го наблюдения,

$|y_i|$ – абсолютное значение фактического значения целевой переменной для нормализации ошибки.

В случае прогнозирования стоимости легкового автомобиля целевой переменной является цена, а вектор признаков содержит следующие параметры автомобиля:

- марка;
- модель;
- пробег;
- количество лет в эксплуатации;
- наличие повреждений;
- тип кузова;
- тип топлива;

– тип привода;
– тип трансмиссии;
– объем двигателя;
– мощность двигателя.

2.2 Проект системы

2.2.1 Структурная схема системы

Структурный подход при разработке системы подразумевает разбиение системы на функциональные подсистемы, тем самым система представляет из себя совокупность отдельных компонентов, взаимодействующих между собой [18]. Таким образом, сохраняется целостность представления о системе, в которой все выделенные подсистемы связаны.

Структурная схема системы представляет из себя диаграмму, отражающую взаимосвязь компонентов системы для понимания принципов работы разрабатываемой системы.

На рисунке 3 приведена структурная схема разрабатываемой системы, в ее состав входят клиентская и серверная части, которые взаимодействуют между собой с помощью протокола HTTP.

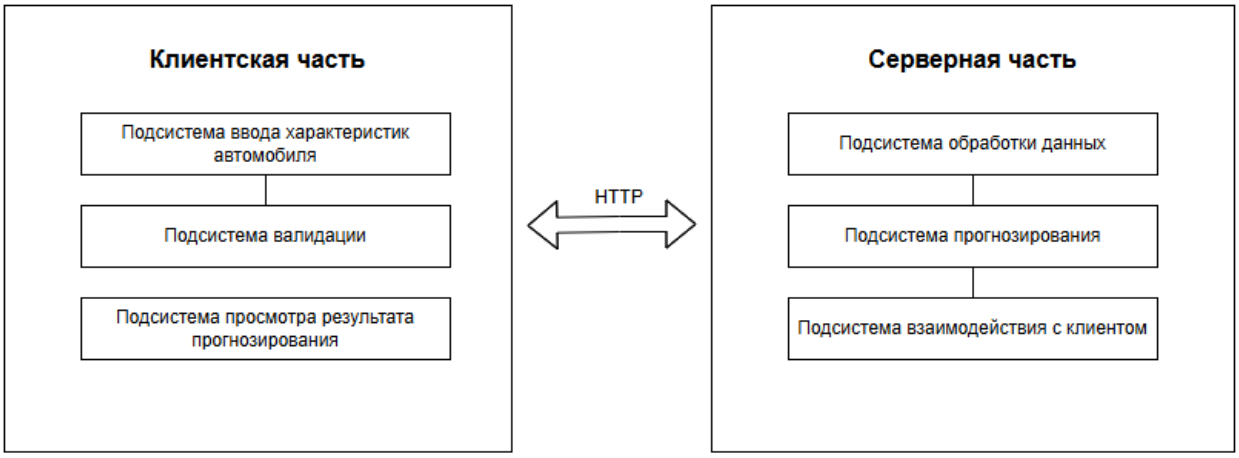

Рисунок 3 – Структурная схема системы

В состав клиентской части входят следующие подсистемы:

1) подсистема ввода характеристик автомобиля, позволяющая пользователю вводить данные об автомобиле;

2) подсистема валидации, отвечающая за проверку корректности введенных значений;

3) подсистема просмотра результатов прогнозирования, которая позволяет пользователю просмотреть результат работы модели.

В состав серверной части входят следующие подсистемы:

1) подсистема обработки данных, которая подготавливает данные, полученные от клиента, в формат, пригодный для работы модели;

2) подсистема прогнозирования, рассчитывающая стоимость автомобиля по введенным данным, используя модель машинного обучения;

3) подсистема взаимодействия с клиентом, которая отвечает за взаимодействие с клиентом.

2.2.2 Диаграмма вариантов использования

Диаграмма вариантов использования – диаграмма, отражающая взаимодействие между актерами и вариантами использования, позволяющая описать концептуальную модель системы [19].

Данная диаграмма широко используется на начальных этапах проектирования программных систем для определения и анализа требований, описания функционала и поведения системы, не погружаясь в особенности реализации функционала.

В качестве актеров выступает некоторое множество ролей, взаимодействующих с системой различными вариантами использования.

Вариант использования представляет из себя спецификацию последовательности действий, которые система выполняет при взаимодействии с актерами, таким образом выделяется набор действий, которые совершает систем, взаимодействуя с актером. Это не просто набор функций, а именно сценарий поведения, который приводит к конкретному результату.

На рисунке 4 приведена диаграмма вариантов использования для пользователя в разрабатываемой системе.

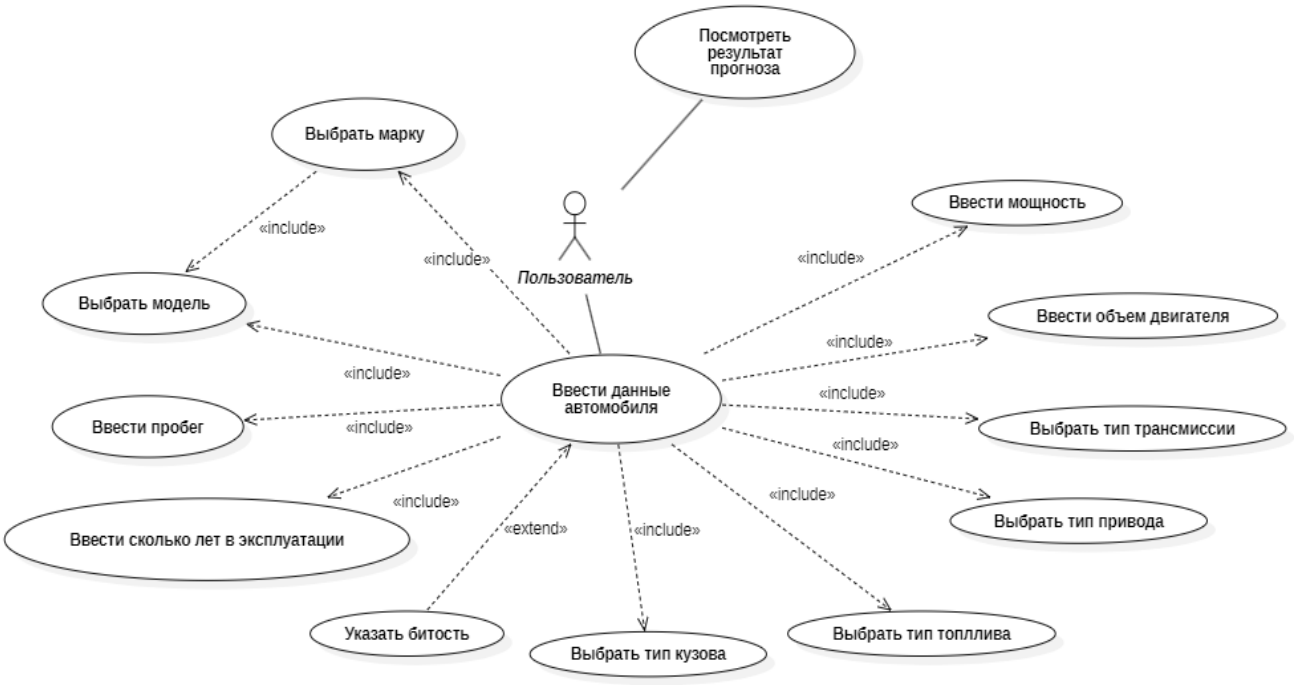

Рисунок 4 – Диаграмма вариантов использования

Основным актором выступает Пользователь, который инициирует основной вариант использования «Ввести данные автомобиля». Этот вариант включает в себя ввод обязательных параметров автомобиля для последующего прогноза.

Действия «Выбрать марку» и «Выбрать модель» тесно связаны между собой, так как модель выбирается из списка, соответсвующего выбранной марке. Действие «Указать битость» является опциональным.

После вворда всех необходимых данных пользователль может инициировать вариант использования «Посмотреть результат прогноза», в котором отображается рассчитанный системой прогноз стоимости.

2.2.3 Диаграмма деятельности

Диаграмма деятельности – это графическое представление рабочих процессов поэтапных действий с поддержкой выбора, итерации и параллелизма. Она отображает динамические аспекты поведения системы и показывает, как поток управления переходит от одной деятельности к другой. Диаграммы деятельности используются при моделировании бизнес-процессов, технологических процессов, последовательных и параллельных вычислений [20].

На рисунке 5 представлена диаграмма деятельности системы.

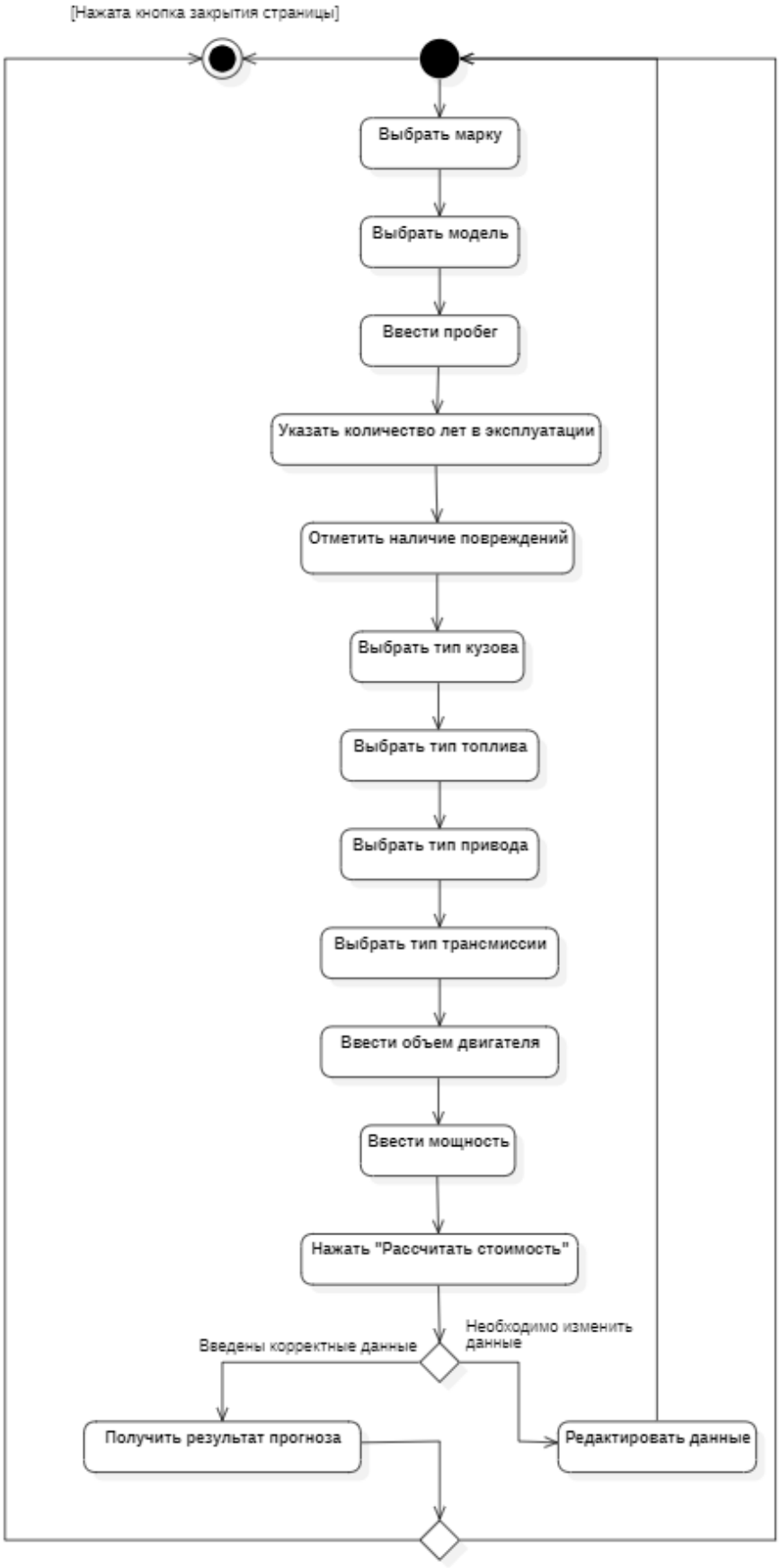

Рисунок 5 – Диаграмма деятельности

Процесс начинается с ввода всех необходимых параметров, после чего пользователь нажимает кнопку «Рассчитать стоимость». Далее происходит проверка введенной информации. Если данные корректны, система отображает результат прогноза, если имеются ошибки или введены не все данные, система сообщит о необходимости редактирования.

2.2.4 Диаграмма последовательности для варианта использования «Посмотреть результат прогноза»

Диаграмма последовательности – UML-диаграмма, отображающая жизненный цикл взаимодействия объектов и акторов на временной оси.

Диаграмма последовательности состоит из следующих элементов: объектов, вертикальных «линий жизни», отображающих временной отрезок, прямоугольников, которые отображают деятельность объекта или исполнение им определенной функции (прямоугольники на пунктирной «линии жизни»), и стрелок, показывающих взаимодействия между объектами [21].

Диаграмма последовательности облегчает разработку автоматизированной системы, так как предоставляет информацию о взаимодействии элементов системы во времени, отражая последовательность проводимых операций.

На рисунке 6 отображена диаграмма последовательности для варианта использования «Посмотреть результат прогноза».

Диаграмма последовательностей описывает процесс получения прогноза стоимости автомобиля пользователем. Пользователь вводит данные о транспортном средстве через форму на странице, затем инициирует расчёт, нажимая кнопку «Рассчитать стоимость». Это действие запускает передачу данных на сервер, где они обрабатываются и используются для прогнозирования цены. Результат прогноза затем отображается пользователю на странице. В завершение пользователь закрывает страницу. Диаграмма визуализирует последовательность действий и передачу сообщений между пользователем, клиентской частью приложения и сервером, позволяя понять, как система работает в реальном времени.

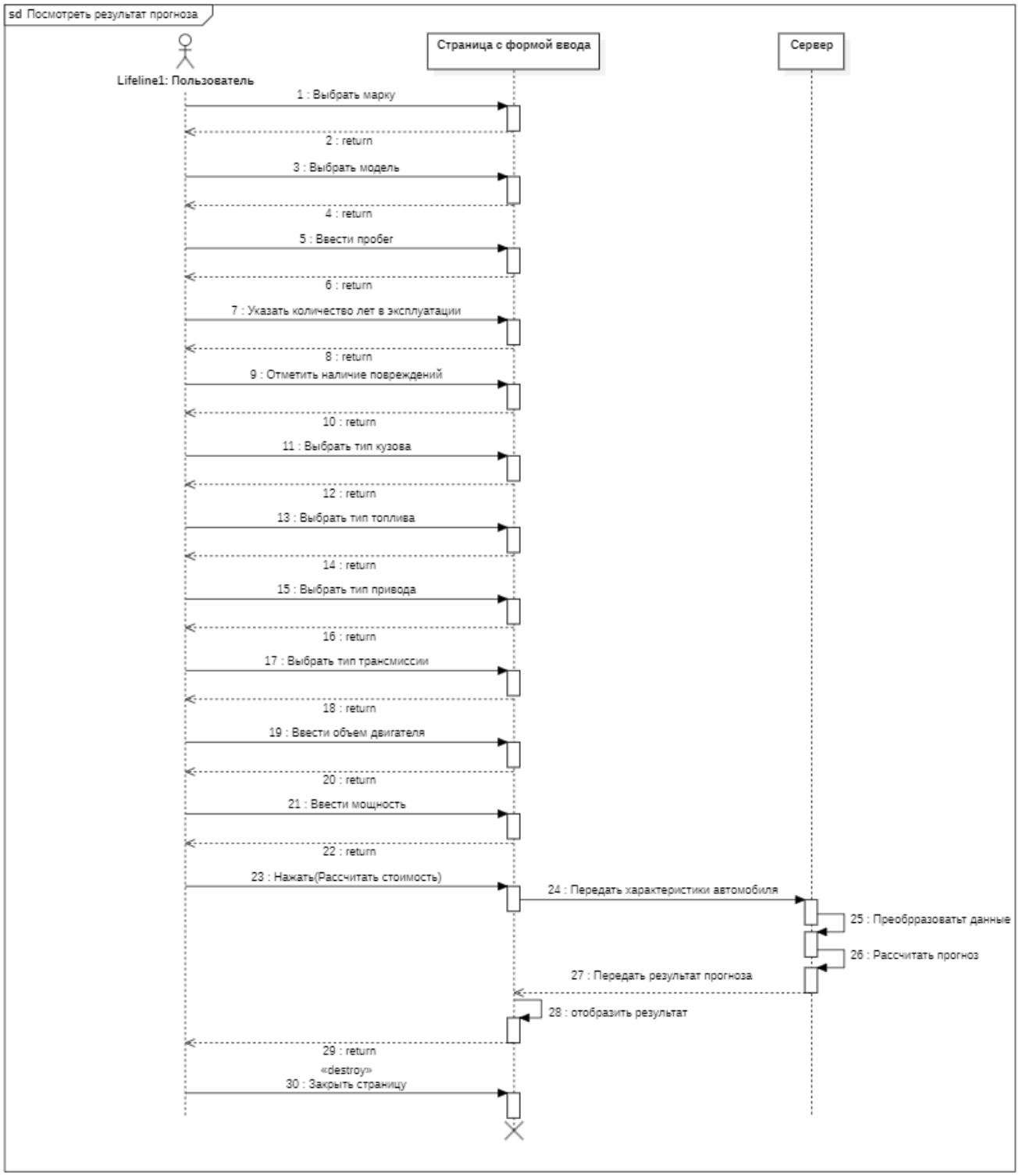

Рисунок 6 – Диаграмма последовательности для варианта использования «Посмотреть результат прогноза»

2.3 Описание средств реализации

В настоящее время существует огромное количество программных продуктов, позволяющих в эффективно и качественно разработать программный комплекс для различных предметных областей. Для правильного

и обоснованного выбора во внимание принимались различные критерии для оценки качества программного продукта.

2.3.1 Описание операционной системы

В качестве операционной системы (ОС) для клиентской части выбрана Windows 10 – операционная система, разработанная компанией Microsoft в 2015 году. Имеет следующие преимущества:

– обладает удобным интерфейсом для облегчения установки и поддержки любого программного обеспечения;

– справляется с перепадами напряжения в сети и обеспечивает отказоустойчивость;

– практически любое программное обеспечение выпускается помимо других ОС для ОС Windows 10.

2.3.2 Описание языка программирования

Для реализации клиентской части системы выбран язык программирования JavaScript с использованием фреймворка Vue.js.

JavaScript – это интерпретируемый язык программирования, который широко используется для создания динамических веб-сайтов и веб-приложений. Он обеспечивает взаимодействие пользователя с содержимым страницы, обновление данных без перезагрузки страницы и дополнительные функциональные возможности.

Vue.js — JavaScript фреймворк для создания пользовательских интерфейсов. Он создан на стандартах HTML, CSS и JavaScript и предоставляет декларативную и компонентную модель программирования, которая помогает эффективно разрабатывать пользовательские интерфейсы любой сложности.

JavaScript был выбран для реализации фронтенда проекта из-за его широкого использования в веб-разработке и гибкости. Vue.js, был выбран из-за своей простоты, производительности и поддержки различных функциональных возможностей для веб-приложений.

Для реализации серверной части системы был выбран язык программирования Python и фреймворк Flask. Python — мультипарадигмальный

высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ. Язык является полностью объектно-ориентированным. Синтаксис ядра языка минималистичен, за счёт чего на практике редко возникает необходимость обращаться к документации. Сам же язык известен как интерпретируемый и используется в том числе для написания скриптов. Также Python имеет большое количество библиотек и фреймворков для анализа данных и машинного обучения.

2.3.3 Описание среды разработки

В качестве среды программирования была выбрана Visual Studio Code. Visual Studio Code (VS Code) — это редактор исходного кода. Его разработал Microsoft для всех популярных операционных систем: Windows, Linux и macOS. Визуальный редактор кода позволяет:

- работать с IntelliSense — автоматическим дописыванием функций при вводе первых букв;
- выполнять отладку — искать и устранять ошибки в написанном коде;
- удобно писать код — автоматически заполнять нужную информацию, подсвечивать элементы синтаксиса в зависимости от выбранного языка, расставлять нужные отступы;
- контролировать версии кода, в том числе с помощью системы управления версиями Git;
- рефакторить код для улучшения его работы и читабельности.

2.3.4 Описание используемой библиотеки

Для написания парсера, собирающего об автомобилях, использовалась библиотека Selenium. Selenium WebDriver — это инструмент для автоматизации тестирования веб-приложений. Он позволяет напрямую взаимодействовать с браузером через API, предоставляя разработчикам возможность автоматизировать задачи, которые раньше занимали много времени [22].

По сути своей использование такого веб-драйвера сводится к созданию бота, выполняющего всю ручную работу с браузером автоматизировано.

Библиотеки WebDriver доступны на языках Java, .Net (C#), Python, Ruby, JavaScript, драйверы реализованы для браузеров Firefox, InternetExplorer, Safari, а также Chrome и Opera.

Для обработки данных использовалась библиотека pandas. Pandas — это инструмент с открытым исходным кодом, предназначенный для обработки организованных данных, представленных в табличном формате. Название библиотеки образовано от термина «панель данных», который обозначает структурированную информацию, полученную в ходе исследований. Работа pandas с данными строится поверх библиотеки NumPy, являющейся инструментом более низкого уровня. Предоставляет специальные структуры данных и операции для манипулирования числовыми таблицами и временны́ми рядами. Pandas имеет хорошие показатели скорости за счет оптимизации кода, интуитивно понятный интерфейс, интеграцию с другими Python-библиотеками [23].

Для реализации метода прогнозирования использовались библиотеки Scikit-learn и CatBoost.

Scikit-learn — Scikit-learn — один из наиболее широко используемых пакетов Python для Data Science и Machine Learning. Он позволяет выполнять множество операций и предоставляет множество алгоритмов. В её состав входят различные алгоритмы, в том числе предназначенные для задач классификации, регрессионного и кластерного анализа данных, включая метод опорных векторов, метод случайного леса, алгоритм усиления градиента, метод k-средних и DBSCAN. Библиотека была разработана для взаимодействия с численными и научными библиотеками языка программирования Python NumPy и SciPy [24].

CatBoost — открытая программная библиотека, разработанная компанией Яндекс и реализующая уникальный патентованный алгоритм построения моделей машинного обучения, использующий одну из оригинальных схем градиентного бустинга. CatBoost обладает рядом преимуществ, таких как

автоматическая обработка категориальных признаков, встроенная регуляризация, автоматический отбор признаков и высокая производительность [25].

2.4 Выводы по главе

В данной главе было выполнено проектирование разрабатываемого веб-приложения прогнозирования стоимости автомобиля. Были исследованы и протестированы различные методы прогнозирования и выбран наилучший по представленным метрикам. Был выбран и описан комплекс программных средств для реализации.

3 Реализация системы

3.1 Подготовка данных

Сбор данных является ключевым этапом разработки модели машинного обучения, так как качество прогнозов напрямую зависит от объема и репрезентативности данных. На основе собранных характеристик автомобилей модель обучается выявлять закономерности между входными параметрами и стоимостью, что позволяет ей делать точные предсказания для новых данных.

Для обучения модели прогнозирования стоимости автомобилей использовались данные, собранные с платформы Avito — одного из крупнейших российских сайтов объявлений о продаже автомобилей.

С помощью библиотеки Selenium был разработан парсер, который автоматически собирает информацию из объявлений о продаже автомобилей.

В результате парсинга были собраны данные о 19994 автомобилях в Самарской области.

Из карточки каждого автомобиля были извлечены ключевые характеристики:

– бренд и модель;
– эксплуатационные характеристики (год выпуска, пробег, состояние);
– технические характеристики (мощность, объем двигателя и т.д.);
– цена – таргет.

Пример карточки приведен на рисунке 7.

Рисунок 7 – Карточка объявления автомобиля

Процесс обработки данных включал следующие этапы:

– очистка данных;

– преобразование категориальных признаков;

– анализ значимости признаков;

– разбиение на обучающую и тестовую выборки.

3.2 Описание экранных форм разработанного веб-приложения

Пользовательский интерфейс – одна из разновидностей интерфейсов, который является совокупностью средств и методов взаимодействия пользователя с вычислительными устройствами (персональным компьютером).

Дружественный интерфейс предоставляет пользователю наиболее удобный способ взаимодействия с программным обеспечением путем обеспечения логичности и простоты в расположении элементов управления.

Разработанная система представляет собой веб-приложение, которое может работать на компьютере под управлением любой операционной системы при наличии браузера.

При запуске приложения открывается страница с формой для ввода характеристик автомобиля (см. рисунок 8).

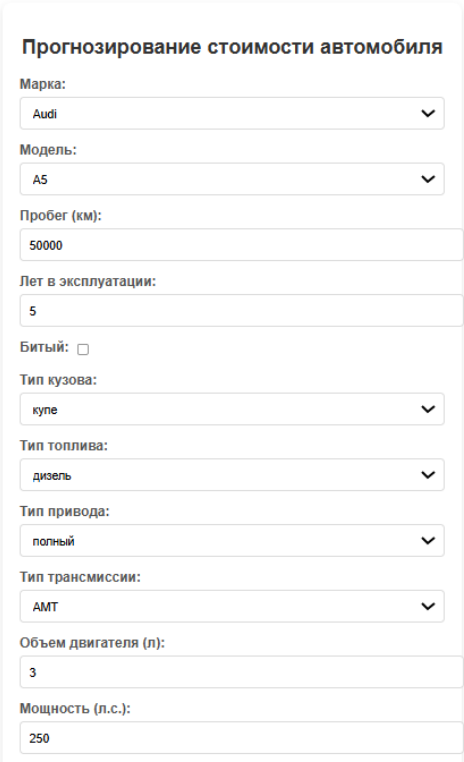

Рисунок 8 – Форма ввода характеристик

Здесь пользователь сможет ввести параметры интересующего автомобиля, после чего нажать кнопку «Рассчитать стоимость» и получить прогноз (см. рисунок 9).

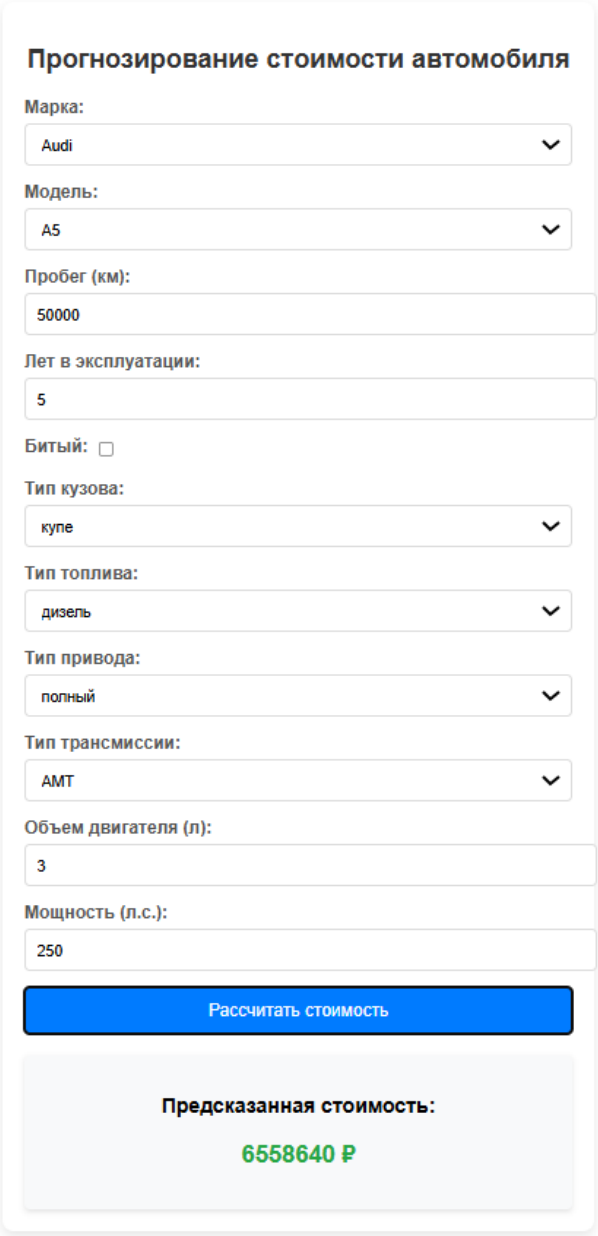

Рисунок 9 – Результат прогноза

3.3 Апробация системы

3.3.1 Ввод данных

При запуске приложения пользователю предоставляется возможность ввести данные транспортного средства и при нажатии кнопки «Рассчитать стоимость» получить прогноз. На рисунке 10 представлена форма для ввода характеристик автомобиля.

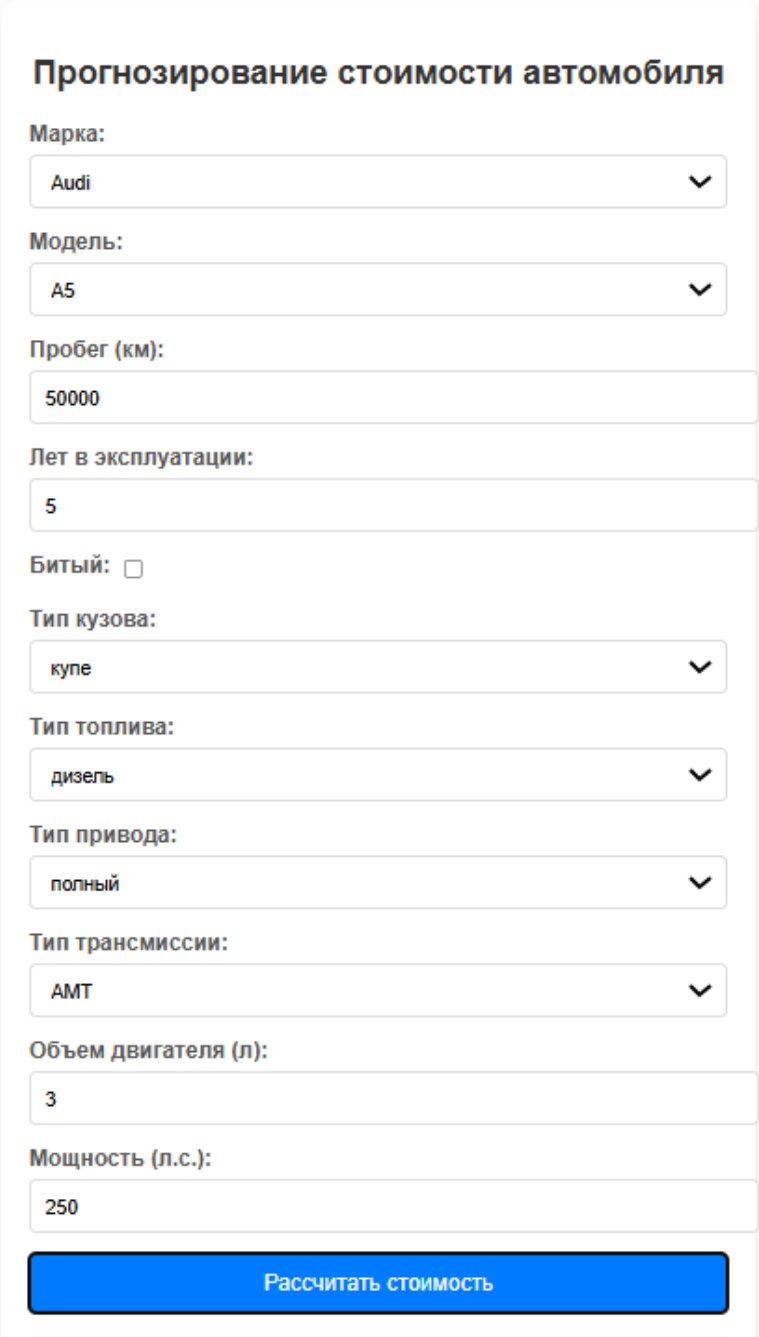

Рисунок 10 – Форма ввода характеристик

После заполнения всех полей и нажатия на кнопку «Рассчитать стоимость» данные проходят валидацию. Система выводит сообщение об ошибке в следующих случаях:

– не заполнены обязательные поля;

– введены некорректные данные.

Также в системы заложены ограничения на поля для предотвращения ошибок. Ограничения представлены в таблице 2.

Таблица 2 – Ограничения на поля в форме ввода характеристик

Поле	Ограничение
Модель	не может быть заполнено, пока не выбрана марка
Пробег	от 0 до 500000
Лет в эксплуатации	от 0 до 50
Объем двигателя	от 0,5 до 6
Мощность	от 30 до 600

3.3.2 Точность прогнозирования на различных входных данных

Было проведено тестирование точности прогнозирования на различных данных для определения ценовых сегментов автомобилей, в которых наблюдается наибольшая ошибка. Результаты тестирования приведены в таблице 3.

Таблица 3 – Примеры прогнозирования в различных ценовых сегментах

Модель	Фактическая стоимость, рублей	Спрогнозированная стоимость, рублей	Абсолютная ошибка, рублей	Абсолютная ошибка в процентах, %
Lada Granta	465 000	446 782	28 218	6,06
Kia Rio	940 000	987 462	47 462	5,04
Skoda Octavia	1 500 000	1 344 871	155 129	10,34
Audi A6	5 000 000	5 862 159	862 159	17,24
Porsche 911	16 500 000	20 252 487	3 752 487	22,74

3.3.3 Интерпретация результатов

Выбор алгоритма прогнозирования основывался на результатах прогнозирования по метрике MAPE, выражающей среднее абсолютное отклонение прогнозируемых значений от фактических значений в процентах.

Результаты оценки моделей на тестовых данных – случайной выборки из 3400 автомобилей представлены на рисунке 11.

Model	MAPE(Test)
LinearRegression	0.511
Ridge	0.510
Lasso	0.511
KNN	0.229
RandomForestRegressor	0.226
CatBoostRegressor	0.167

Рисунок 11 – Результаты оценки моделей

Наилучший результат показала модель CatBoostRegressor, которая представляет собой реализацию градиентного бустинга над решающими деревьями библиотеки CatBoost. В итоговой реализации веб-приложения используется именно эта модель.

Проведённое тестирование веб-приложения подтвердило стабильную и точную работу модели прогнозирования стоимости, особенно в бюджетном и среднем ценовом сегментах. Ошибка увеличивается при оценке премиальных автомобилей, например, Audi A6 и Porsche 911 из-за ограниченного количества данных и высокой чувствительности их цены к различным факторам

Модель хорошо справляется с типичными случаями, но менее точна при наличии нестандартных характеристик. Встроенная валидация данных минимизирует ошибки ввода. В целом, система показала свою работоспособность, но нуждается в дальнейшем развитии, особенно в части

повышения точности прогнозирования для автомобилей премиум-класса и включения в модель дополнительных факторов, влияющих на ценообразование

3.4 Описание проведенных исследований

3.4.1 Предмет исследования

В рамках выпускной квалификационной работы предметом исследования является значимость признаков в прогнозировании стоимости автомобиля. Рассмотрим исследование признака is_crashed («битый / небитый»), используя сравнительную статистику.

3.4.2 Результаты исследования

Для каждого битого автомобиля из выборки подбирался аналогичный не битый автомобиль той же модели и года выпуска, с максимально близким пробегом (допустимое отклонение ±5%). Таким образом, сравнивались пары автомобилей, различающиеся только по признаку аварийности, при прочих равных характеристиках. На основании анализа 89 пар автомобилей была построена сводная таблица с основными статистическими характеристиками разницы в стоимости между битым и не битым автомобилем (см. таблицу 4).

Таблица 4 – Статистические характеристики разницы в стоимости между битым и не битым автомобилем

Показатель	Значение
Количество пар автомобилей	89
Средняя разница в цене	270 328 ₽
Минимальная разница в цене	10 000 ₽
Максимальная разница в цене	1 249 000 ₽
Медианная разница в цене	190 000 ₽
Средний пробег битого авто	142 785 км
Средний пробег не битого авто	139 212 км

Как видно из данных таблицы 4, средняя разница составила 270 328 ₽, при этом максимальная разница достигала 1 249 000 ₽, а медианная разница — 190

000 ₽. Это подтверждает, что битость оказывает существенное влияние на рыночную цену автомобилей. Также важно отметить, что различия в пробеге между сравниваемыми автомобилями были минимальными (в среднем около 3 500 км), что делает результаты сравнения более достоверными.

Как видно на рисунке 12, распределение разницы в стоимости между битым и небитым автомобилем характеризуется концентрацией значений в диапазоне от 150 000 ₽ до 300 000 ₽.

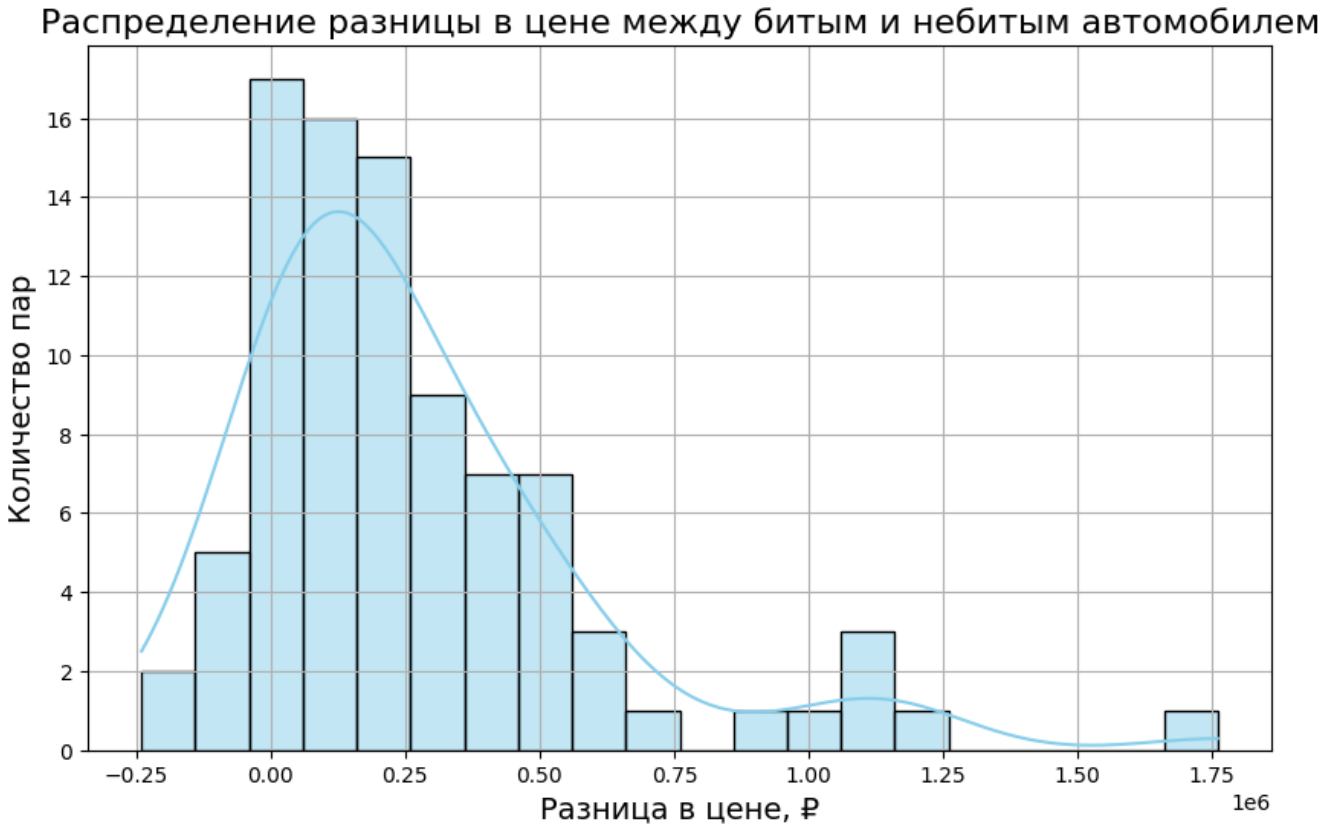

Рисунок 12 – График распределения разницы в ценах

Анализ плотности распределения показывает, что наибольшее количество пар автомобилей имеет разницу в цене, близкую к средней величине, которая составляет 270 328 ₽.

Распределение имеет асимметричную форму с выраженным правосторонним хвостом, что свидетельствует о наличии автомобилей с экстремально высокой разницей в стоимости. Такие случаи, как правило, соответствуют дорогостоящим моделям автомобилей, для которых битость существенно влияет на потерю рыночной стоимости.

На основании полученных данных можно сделать вывод, что признак битости автомобиля является значимым фактором при прогнозировании его рыночной стоимости. Наличие повреждений приводит к заметному снижению стоимости транспортного средства, что необходимо учитывать при разработке моделей оценки цены автомобиля.

3.5 Выводы по главе

В данной главе был рассмотрен процесс подготовки данных для построения модели прогнозирования стоимости автомобиля. С использованием парсинга объявлений с платформы Avito был сформирован объемный и репрезентативный набор данных, содержащий ключевые технические и эксплуатационные характеристики автомобилей, а также информацию об их состоянии.

Также было проведено исследование значимости признаков, в частности бинарного признака состояния автомобиля "битый/небитый". Проведенный сравнительный анализ пар автомобилей с одинаковыми характеристиками (модель, год выпуска, пробег) показал, что наличие повреждений оказывает существенное влияние на рыночную стоимость автомобиля.

Таким образом, проведенное исследование позволило обоснованно утверждать, что признак битости является одним из наиболее важных факторов, влияющих на стоимость автомобиля, и должен обязательно учитываться при построении моделей машинного обучения для оценки рыночной цены транспортных средств.

ЗАКЛЮЧЕНИЕ

В процессе выполнения выпускной работы было разработано веб-приложение прогнозирования стоимости легкового автомобиля.

В первом разделе были приведены основные понятия и определения предметной области прогнозирования стоимости легкового автомобиля, актуальность исследования, приведены характеристики систем-аналогов, на основании этого была сформулирована постановка задачи и основные требования к системе.

Во втором разделе было выполнено проектирование разрабатываемого веб-приложения прогнозирования стоимости автомобиля. Были исследованы и протестированы различные методы прогнозирования и выбран наилучший по представленным метрикам, а также был выбран и обоснован комплекс программных средств.

В третьем разделе описан процесс сбора и подготовки данных, интерфейс пользователя, а также приведены результаты проведенных исследований, проведен анализ значимости признаков на прогнозирование стоимости автомобиля.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 В России бум продаж подержанных машин. [Электронный ресурс] // Газета.Ru – URL: https://www.gazeta.ru/auto/2025/02/20/20585066.shtml (дата обращения: 22.04.2025).

2 Обзор методов прогнозирования. [Электронный ресурс] // Ivan Shamaev – URL: https://ivan-shamaev.ru/overview-forecast-methods/ (дата обращения: 24.04.2025).

3 Что такое регрессионный анализ [Электронный ресурс] // Яндекс Практикум – URL: https://practicum.yandex.ru/blog/chto-takoe-regressionnyj-analiz/ (дата обращения: 22.04.2025).

4 Машинное обучение: общие принципы и концепции [Электронный ресурс] // Хабр: [сайт]. – URL: https://habr.com/ru/articles/862704/ (дата обращения: 22.04.2025).

5 Что такое модель машинного обучения? [Электронный ресурс]. // Microsoft Learn: [сайт]. – URL: https://learn.microsoft.com/ru-ru/windows/ai/windows-ml/what-is-a-machine-learning-model (дата обращения: 22.04.2025).

6 Regression in machine learning [Электронный ресурс]. // GeeksforGeeks: [сайт]. – URL: https://www.geeksforgeeks.org/regression-in-machine-learning/ (дата обращения: 22.04.2025).

7 Авито Оценка [Электронный ресурс]. – URL: https://www.avito.ru/evaluation/cars (дата обращения: 26.04.2025).

8 Auto.ru «Оценка» [Электронный ресурс]. – URL: https://miratext.ru/seo_analiz_text (дата обращения: 26.04.2025).

9 Линейная регрессия. Основная идея, модификации и реализация с нуля на Python [Электронный ресурс]. URL: https://habr.com/ru/articles/804135/ (дата обращения: 20.12.2024).

10 Алгоритм градиентного спуска в машинном обучении [Электронный ресурс]. URL: https://www.geeksforgeeks.org/gradient-descent-algorithm-and-its-variants/ (дата обращения: 20.12.2024).

11 Регуляризация – Викиконспекты [Электронный ресурс]. URL: https://neerc.ifmo.ru/wiki/index.php?title=Регуляризация (дата обращения: 20.12.2024).

12 Регрессия методом k-ближайших соседей (KNN) с помощью Scikit-Learn [Электронный ресурс]. URL: https://www.geeksforgeeks.org/k-nearest-neighbors-knn-regression-with-scikit-learn/ (дата обращения: 20.12.2024).

13 Открытый курс машинного обучения. Тема 5. Композиции: бэггинг, случайный лес [Электронный ресурс]. URL: https://habr.com/ru/companies/ods/articles/324402/ (дата обращения: 20.12.2024).

14 Что такое бустинг? [Электронный ресурс]. URL: https://aws.amazon.com/ru/what-is/boosting/ (дата обращения: 21.12.2024).

15 Градиентный бустинг [Электронный ресурс]. URL: https://education.yandex.ru/handbook/ml/article/gradientnyj-busting (дата обращения: 21.12.2024).

16 CatBoost [Электронный ресурс]. URL: https://habr.com/ru/companies/otus/articles/778714/ (дата обращения: 21.12.2024).

17 Метрики в машинном обучении: понимание, применение и интерпретация [Электронный ресурс]. URL: https://shakhbanov.org/metriki-v-mashinnom-obuchenii/ (дата обращения: 21.12.2024).

18 Composit structure diagram [Электронный ресурс]. URL: https://en.wikipedia.org/wiki/Composite_structure_diagram (дата обращения: 1.05.2025).

19 Диаграмма вариантов использования: Обзор [Электронный ресурс]. URL: https://habr.com/ru/articles/566218/ (дата обращения: 3.05.2025).

20 Диаграмма деятельности: Обзор [Электронный ресурс]. URL: https://ru.wikipedia.org/wiki/Диаграмма_деятельности/ (дата обращения: 3.05.2025).

21 Буч Г. Язык UML. Руководство пользователя: Пер. с англ. / Г. Буч, Д. Рамбо, Б. Джекобсон. М.: ДМК-Пресс, 2001. 432 с.

22 Что такое Selenium WebDriver [Электронный ресурс]. URL: https://ru.hexlet.io/blog/posts/chto-takoe-selenium-webdriver?ysclid=mb3ionxn5j136782631 (дата обращения: 1.05.2025).

23 Библиотека Pandas в Python: что это и как работает [Электронный ресурс] – URL: https://simulative.ru/blog/pandas-kak-eto-rabotaet (дата обращения: 3.05.2025).

24 Что такое Scikit Learn — гайд по популярной библиотеке Python для начинающих [Электронный ресурс]. URL: https://datastart.ru/blog/chto-takoe-scikit-learn-gayd-po-populyarnoy-biblioteke-python-dlya-nachinayuschih/ (дата обращения: 3.05.2025).

25 Introduction to CatBoost [Электронный ресурс]. URL: https://www.geeksforgeeks.org/introduction-to-catboost/?ysclid=mb3iureuzy125449732 (дата обращения: 3.05.2025).

ПРИЛОЖЕНИЕ А. Руководство пользователя

А.1 Назначение системы

Данная система предназначена для прогнозирования стоимости легкового автомобиля по введенным данным.

А.2 Условия работы системы

Для корректной работы системы необходимо наличие соответствующих программных и аппаратных средств.

1) Требования к техническому обеспечению:
   - ЭВМ типа IBM PC;
   - процессор типа x86 или x64 тактовой частоты 1400 МГц и выше;
   - клавиатура или иное устройство ввода;
   - мышь или иное манипулирующее ввода;
   - дисплей с разрешением не менее 1280 × 768 пикселей;
   - широкополосное подключение к сети Интернет, не менее 1 Мб/сек.
2) Требования к программному обеспечению:
   - операционная система Windows 10 и выше.

А.3 Установка системы

Перед запуском системы необходимо установить Node.js и Python. Система поставляется в виде zip-архива. Данный файл необходимо распаковать в любую директорию на жестком диске.

В директории «backend» необходимо:

- установить зависимости, выполнив команду pip install requirements.txt;
- запустить файл app.py.

В директории «frontend» необходимо:

- установить зависимости, выполнив команду npm install;
- выполнить команду npm run serve.

При успешной настройке система будет доступна в браузере по ссылке http://localhost:8080/.

А.4 Работа с системой

При начале работы с системой пользователю открывается страница с формой ввода характеристик, представленная на рисунке А.1.

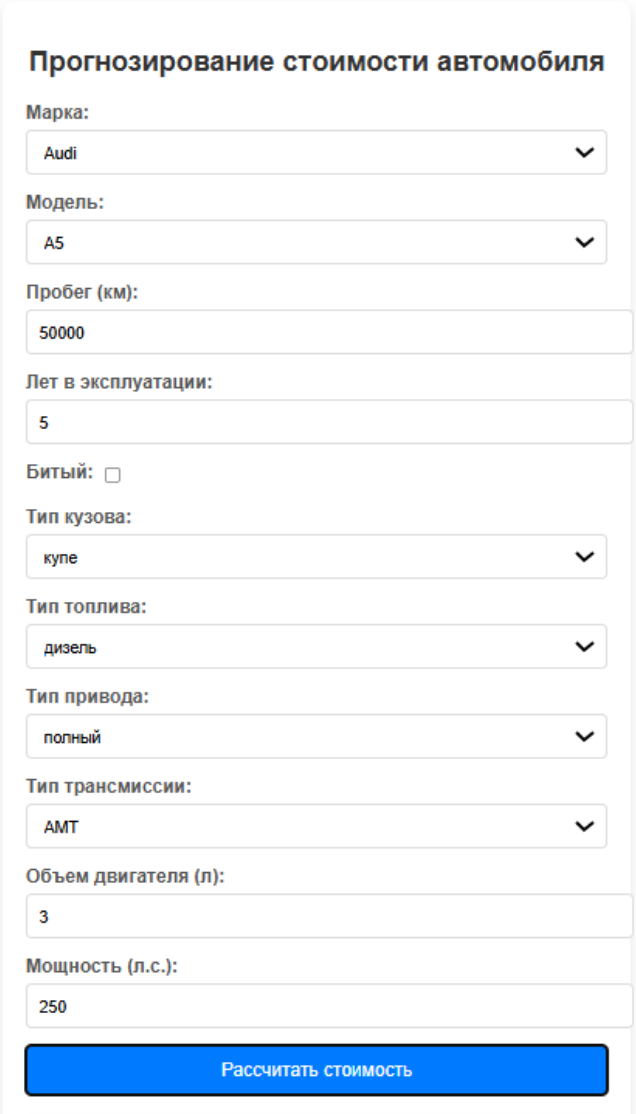

Рисунок А.1 – Форма ввода характеристик

Чтобы получить прогноз, необходимо заполнить все поля и нажать кнопку «Рассчитать стоимость». Пример результата прогноза приведен на рисунке А.2.

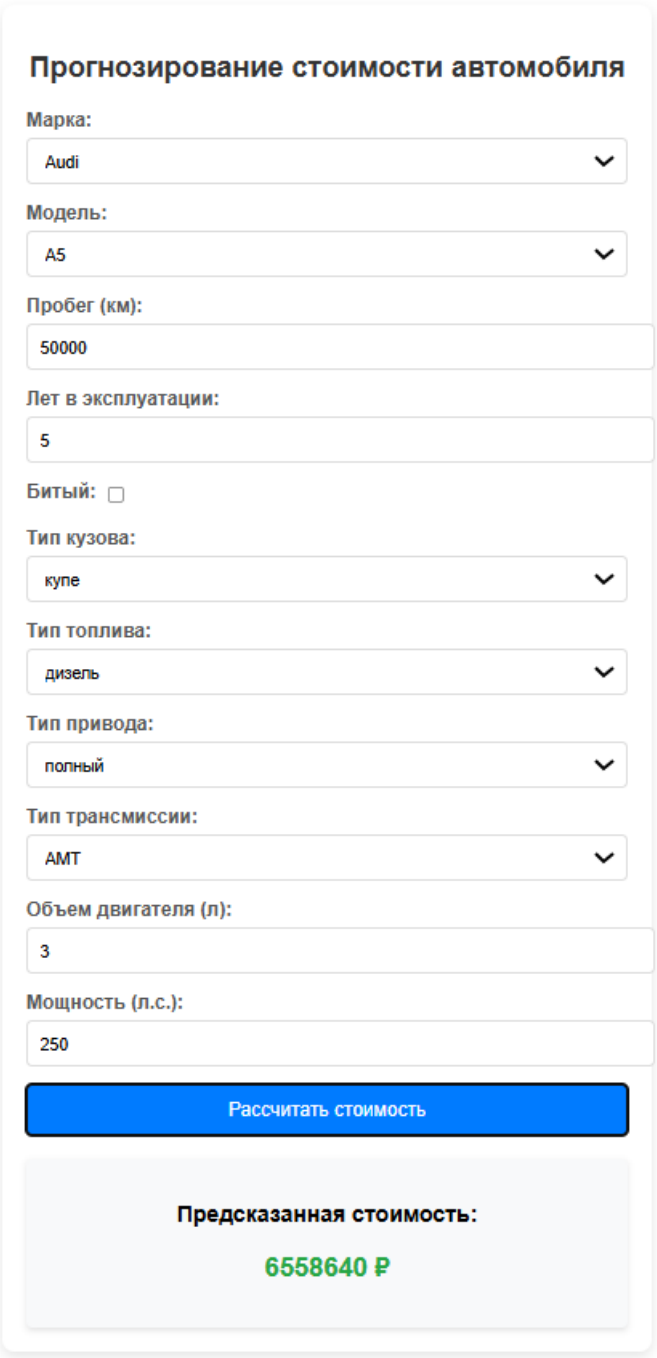

Рисунок А.2 – Пример результата прогноза

```
from flask import Flask, request, jsonify

from flask_cors import CORS

import pickle

import pandas as pd


app = Flask(__name__)

CORS(app)



path_to_utils =
'C:/Users/Dmitry/Documents/GitHub/data-science-
projects/car_price/backend/utils/'

scaler = pickle.load(open(path_to_utils + 'scaler.pkl',
'rb'))

model = pickle.load(open(path_to_utils +
'catboost_model.pkl', 'rb'))



def translate_data(car):

    rename_dict = pickle.load(open(path_to_utils +
'rename_dict.pkl', 'rb'))

    car['drive_type'] = car['drive_type'].apply(lambda
x: rename_dict[x])

    car['fuel_type'] = car['fuel_type'].apply(lambda x:
```

```
rename_dict[x])


def one_hot_encoding(car):
    one_hot_cols = ['drive_type', 'fuel_type',
'transmission_type']
    result = car.drop(columns=one_hot_cols)


    for col in one_hot_cols:
        encoder = pickle.load(
            open(path_to_utils + f'{col}_one_hot.pkl',
'rb'))


        col_encoded =
pd.DataFrame(encoder.transform(car[[col]]).toarray(
        ),
columns=encoder.get_feature_names_out([col]))


        result = pd.concat((result, col_encoded),
axis=1)


    return result


def target_encoding(car):
    target_encoded_cols = ['body_type', 'brand',
'model']
```

```
    for col in target_encoded_cols:
        encoder = pickle.load(
            open(path_to_utils + f'{col}_target.pkl', 'rb'))

        car[col] = encoder[car[col].values[0]]

def prepare_data(data):
    car = make_df(data)
    car.reset_index(drop=True, inplace=True)
    translate_data(car)
    target_encoding(car)
    return one_hot_encoding(car)

def make_df(data):
    prepared_data = [{
        'brand': data['brand'],
        'years_in_operation': data['yearsInUse'],
        'is_crashed': 1 if data['damaged'] else 0,
        'model': data['brand'] + ' ' + data['model'],
        'mileage': data['mileage'],
        'body_type': data['bodyType'],
        'drive_type': data['driveType'],
```

```
        'fuel_type': data['fuelType'],
        'engine_volume': data['engineCapacity'],
        'transmission_type': data['transmissionType'],
        'power': data['horsePower'],
    }]
    return pd.DataFrame(prepared_data)


@app.route('/api/predict', methods=['POST'])
def predict():
    data = request.json

    ready_data = prepare_data(data)

    data_scaled = scaler.transform(ready_data)
    price =int(model.predict(data_scaled)[0])

    return jsonify({'price': price}), 200


if __name__ == '__main__':
    app.run(debug=True)
```

<h3>Парсинг объявлений о продаже

автомобилей с Авито</h3>

```
# импорт библиотек
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import
WebDriverWait
from selenium.webdriver.support import
expected_conditions as EC
from selenium.common.exceptions import
TimeoutException

import time
from random import randint

import pandas as pd
# начальная страница парсинга
URL =
'https://www.avito.ru/samarskaya_oblast/transport'
```

Напишем функцию для решения Captcha

```
def kill_captcha(driver):
    wait = WebDriverWait(driver, 5)
    try:
        # Проверка на наличие капчи
        captcha_element =
```

```
wait.until(EC.presence_of_element_located(
                (By.XPATH, "//h2[@class='firewall-
title']")), message="Капча обнаружена")


    if captcha_element:
        print("Капча обнаружена. Решите ее
вручную.")
        input("Нажмите Enter, когда решите
капчу...")


  except TimeoutException:
    pass
```

Напишем функцию получения данных с объявления

```
def get_car_data(ad, mark):
  data = {}


  data['id'] = ad.get_attribute("id")
  data['mark'] = mark
  data['name'] = ad.find_element(
                by=By.CSS_SELECTOR,
value='h3[itemprop="name"]'
              ).text
```

```
    data['price'] = ad.find_element(
                    by=By.CSS_SELECTOR,
value='meta[itemprop="price"]'
                ).get_attribute("content")

    data['params'] = ad.find_element(
                    by=By.CSS_SELECTOR,
value='p[data-marker="item-specific-params"]'
                ).text

    return  data
```

Напишем функцию парсинга для всех объявлений

```
def parse_cars(url):
    # Настраиваем параметры браузера
    options = webdriver.ChromeOptions()
    # Отключает функцию, указывающую сайтам,
что браузер управляется автоматизацией
    options.add_argument("--disable-blink-
features=AutomationControlled")
    driver = webdriver.Chrome(options=options)
    driver.implicitly_wait(10)

    driver.get(url)
```

```
    kill_captcha(driver)
    # парсим популярные модели
    popular_models = driver.find_element(
        by=By.CSS_SELECTOR, value='div[data-
marker="popular-rubricator/links"]'
    ).find_elements(
        by=By.CSS_SELECTOR, value='a[data-
marker="popular-rubricator/link"]'
    )

    models = [{'name': x.text, 'link':
x.get_attribute('href')} for x in popular_models]

    all_cars = []
    for model in models:
        # перейдем к объявлениям марки
        driver.get(model['link'])

        kill_captcha(driver)

        cars = []
        while True:
            # делаем небольшую задержку
```

```
            rand_sleep = randint(25, 49)
            time.sleep(rand_sleep / 10)

            ads = driver.find_elements(
                by=By.CSS_SELECTOR,
value='div[data-marker="item"]'
            )

            for ad in ads:
                car_data = get_car__data(ad,
model['name'])
                cars.append(car_data)

            try:
                # Ожидаем, пока кнопка станет
видимой и доступной
                button = WebDriverWait(driver, 10).until(

EC.presence_of_element_located((By.CSS_SELEC
TOR, 'a[data-marker="pagination-
button/nextPage"]'))
                )

                if button.is_displayed() and
button.is_enabled():
```

```
                button.click()
            else:
                break

        except Exception:
            break

    all_cars += cars

pd.DataFrame(cars).to_csv(f"../data/brands/cars_dat
a_{model['name']}.csv")

driver.quit()
return all_cars
```

Авито отображает максимум 100 страниц по 50 объявлений, поэтому мы проходимся по популярным брендам и парсим их по очереди, чтобы суммарно было больше объявлений.

```
# Парсинг
cars = parse_cars(URL)
# сохраняем данные
cars = pd.DataFrame(cars)
cars.to_csv('../data/cars_data.csv')cars.sample(n=5)
```