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

ОТЧЕТ ПО ПРАКТИКЕ

Вид практики производственная
(учебная, производственная)

Тип практики научно-исследовательская работа
(в соответствии с ОПОП ВО)

Сроки прохождения практики: с 01.10.2024 по 26.12.2024
(в соответствии с календарным учебным графиком)

по направлению подготовки 02.03.02
«Фундаментальная информатика и информационные технологии
(уровень бакалавриата)
направленность (профиль) «Информационные технологии»

Обучающийся группы № 6401-020302D_____ Д.О. Колбанов

Руководитель практики,
Доцент кафедры программных систем,
к.т.н, доцент _____ О.А. Гордеева

Дата сдачи 26.12.2024
Дата защиты 26.12.2024

Оценка _____

Самара 2024

СОДЕРЖАНИЕ

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

Задания по практике для выполнения определенных видов работ, связанных с будущей профессиональной деятельностью
(сбор и анализ данных и материалов, проведение исследований)

Обучающемуся Колбанову Дмитрию Олеговичу группы 6401-020302D

Направлен на практику приказом по университету от 27.09.2024 г. №451-ПР
на кафедру программных систем

Тема: Веб-приложение прогнозирования стоимости легкового автомобиля

Планируемые результаты освоения образовательной программы (компетенции)	Выполнение определенных видов работ, связанных с будущей профессиональной деятельностью (сбор и анализ данных и материалов, проведение исследований)	Результаты практики
ОПК-1. Способен применять фундаментальные знания, полученные в области математических и (или) естественных наук, и использовать их в	Исследовать современные подходы и технологии создания веб-приложений с использованием универсальных программных средств.	Обзор существующих библиотек и программных средств для разработки веб-приложений и

профессиональной деятельности. **ОПК-1.1.** Использует основные положения и концепции в области математических и естественных наук, Базовые теории и истории основного, теории коммуникации; знает основную терминологию. **ОПК-1.2.** Осуществляет первичный сбор и анализ материала, интерпретирует различные математические объекты. **ОПК-1.3.** Применяет опыт решения стандартных математических задач в профессиональной деятельности.	Исследовать существующие методы и библиотеки для машинного обучения.	моделей машинного обучения.
ОПК-2. Способен применять компьютерные/суперкомпьютерные методы, современное программное обеспечение, в том числе отечественного происхождения, для решения задач профессиональной деятельности. **ОПК-2.1.** Использует основные положения и концепции в области программирования, архитектуры языков программирования, теории коммуникации, знает основную терминологию, знаком с содержанием Единого Реестра Российских программ. **ОПК-2.2.** Анализирует код на типовых языках программирования, может составлять программы. **ОПК-2.3.** Применяет опыт решения задач анализа, интеграции различных типов	Проанализировать возможности Scikit-learn и CatBoost для создания моделей машинного обучения	Изучены и освоены следующие программные средства: Scikit-learn, CatBoost

программного обеспечения, анализа типов коммуникаций.		
ОПК-3. Способен к разработке алгоритмических и программных решений в области системного и прикладного программирования, математических, информационных и имитационных моделей, созданию информационных ресурсов глобальных сетей, образовательного контента, прикладных баз данных, тестов и средств тестирования систем и средств на соответствие стандартам и исходным требованиям. **ОПК-3.1.** Понимает методы теории алгоритмов, методы системного и прикладного программирования, основные положения и концепции в области математических, информационных и имитационных моделей. **ОПК-3.2.** Соотносит знания в области программирования, интерпретацию прочитанного, определяет и создает информационные ресурсы глобальных сетей, образовательного контента, средств тестирования систем. **ОПК-3.3.** Имеет практический опыт применения разработки программного обеспечения.	Анализ современных подходов обработки данных и построения моделей машинного обучения.	Подготовлены данные для обучения и протестированы различные алгоритмы.
ОПК-4. Способен участвовать в разработке технической документации программных продуктов и комплексов с использованием стандартов, норм и правил, а также в управлении проектами	Написание письменного отчета по выполненной работе.	Отчет по результатам выполненных задач

создания информационных систем на стадиях жизненного цикла. **ОПК-4.1.** Использует принципы сбора и анализа информации, создания информационных систем на стадиях жизненного цикла. **ОПК-4.2.** Осуществляет управление проектами информационных систем. **ОПК-4.3.** Демонстрирует практический опыт анализа и интерпретации информационных систем.		
ОПК-5. Способен инсталлировать и сопровождать программное обеспечение информационных систем и баз данных, в том числе отечественного происхождения, с учетом информационной безопасности. **ОПК-5.1.** Понимает методику установки и администрирования информационных систем и баз данных. Знаком с содержанием Единого реестра российских программ. **ОПК-5.2.** Реализует техническое сопровождение информационных систем и баз данных. **ОПК-5.3.** Использует практические навыки установки и инсталляции программных комплексов, применения основ сетевых технологий.	Апробация выбранных средств разработки модели прогнозирования стоимости.	Апробированы инструменты Scikit-learn и CatBoost.
ОПК-6. Способен понимать принципы работы современных информационных технологий	Изучение и апробация выбранных фреймворков для	Апробированы фреймворки Vue.js и Flask

и использовать их для решения задач профессиональной деятельности **ОПК-6.1.** Понимает основные положения, концепции и современные методы обработки и хранения данных. **ОПК-6.2.** Осуществляет первичный сбор и анализ данных для организации информационных процессов **ОПК-6.3.** Обладает практическим опытом применения современных информационных технологий для решения задач профессиональной деятельности.	разработки веб-приложения.	

Дата выдачи задания 01.10.2024.
Срок представления на кафедру отчета о практике 26.12.2024.

Руководитель практики,
Доцент кафедры программных систем,
к.т.н, доцент _____ О.А. Гордеева
(подпись)

Задание принял к исполнению
обучающийся группы № 6401-020302D _____ Д.О. Колбанов
(подпись)

ВВЕДЕНИЕ

Современные веб-приложения играют важную роль в жизни людей, предоставляя удобные инструменты для решения различных задач — от общения и покупок до анализа данных и управления бизнес-процессами. Одним из ключевых преимуществ веб-приложений является их доступность и гибкость, что делает их популярным выбором для реализации сложных пользовательских сервисов [1]. С развитием технологий и увеличением объемов данных требования к функциональности, точности и производительности таких приложений стремительно растут. Пользователи ожидают, что сервисы будут не только удобными и интуитивно понятными, но и способными быстро предоставлять достоверные результаты.

Одной из актуальных задач в этой области является разработка инструментов, которые помогают пользователям принимать обоснованные решения, используя алгоритмы машинного обучения и анализа данных. Примером такого инструмента является веб-приложение для прогнозирования рыночной стоимости легкового автомобиля. Оно позволяет пользователям оценить стоимость транспортного средства на основе характеристик, таких как марка, модель, пробег, год выпуска и другие параметры. Такие решения находят применение в автомобильных дилерских центрах, онлайн-площадках для продажи автомобилей и среди частных пользователей, предоставляя им точные и оперативные прогнозы.

Во время практики необходимо решить следующие задачи:

– исследовать современные подходы и технологии создания веб приложений с использованием универсальных программных средств;

– исследовать существующие методы и библиотеки для машинного обучения;

– изучить и освоить инструменты для создания веб-приложений;

– изучить и освоить инструменты для разработки моделей машинного обучения;

– протестировать различные методы прогнозирования и взять наилучший по выбранной метрике;

– подготовить и оформить письменный отчет по выполненной работе.

1 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.

1.1 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.

1.2 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.

1.3 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.

1.4 Описание используемой библиотеки

Для написание парсера, собирающего об автомобилях, использовалась библиотека Selenium. Selenium WebDriver – это программная библиотека для управления браузерами. WebDriver представляет собой драйверы для различных браузеров и клиентские библиотеки на разных языках программирования, предназначенные для управления этими драйверами.

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 — программная библиотека на языке Python для обработки и анализа данных. 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.

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

Scikit-learn — библиотека, предназначенная для машинного обучения, написанная на языке программирования Python и распространяемая в виде свободного программного обеспечения. 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.

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.

2 Описание проекта разрабатываемого приложения

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 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 [2]. 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.

2.1.1 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 [3].

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.

2.1.2 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 [4].

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.

2.2 Метод ближайших соседей

Метод ближайших соседей (KNN) в задаче регрессии — это непараметрический метод, используемый для прогнозирования непрерывных значений [5].

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.

2.3 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 [6].

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.

Итоговая модель выглядит следующим образом:

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

2.4 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 [7].

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).

2.4.1 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 [8].

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.

2.4.2 Реализация градиентного бустинга в CatBoost

CatBoost использует решающие деревья глубины 1 или 2 в качестве базовых моделей [9]. 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 включает механизм регуляризации, чтобы предотвратить переобучение модели. Он использует L2-регуляризацию, представляет собой метод добавления штрафа к функции потерь модели с целью предотвратить переобучение. Этот метод помогает улучшить обобщающую способность модели и снизить риск переобучения, особенно в случаях, когда много признаков или они коррелированы между собой.

CatBoost автоматически выполняет отбор признаков путем оценки их важности для модели. Это позволяет модели сосредотачиваться на наиболее информативных признаках и уменьшить шум от менее значимых.

CatBoost применяет градиентный бустинг для обучения ансамбля решающих деревьев. Градиентный бустинг минимизирует функцию потерь с использованием градиентного спуска, постепенно улучшая качество модели.

На каждой итерации градиентного бустинга добавляется новое решающее дерево, которое исправляет ошибки предыдущих деревьев.

2.5 Метрика качества

MAPE выражает среднее абсолютное отклонение прогнозируемых значений от фактических значений в процентах, что делает эту метрику очень наглядной для интерпретации результатов [10].

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,

$f(x_i)$ – 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 результате оценки моделей на тестовых данных получились следующие значения:

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

Рисунок 1 – результаты оценки моделей

Наилучший результат показала модель CatBoostRegressor, которая представляет собой реализацию градиентного бустинга над решающими деревьями библиотеки CatBoost. Для прогнозирования стоимости автомобиля будет использоваться эта модель.

3 Описание экранных форм разработанного программного приложения

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

Интерфейс характеризуется удобством, эффективностью, понятностью и дружественностью.

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

Разработанная система представляет собой web-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.

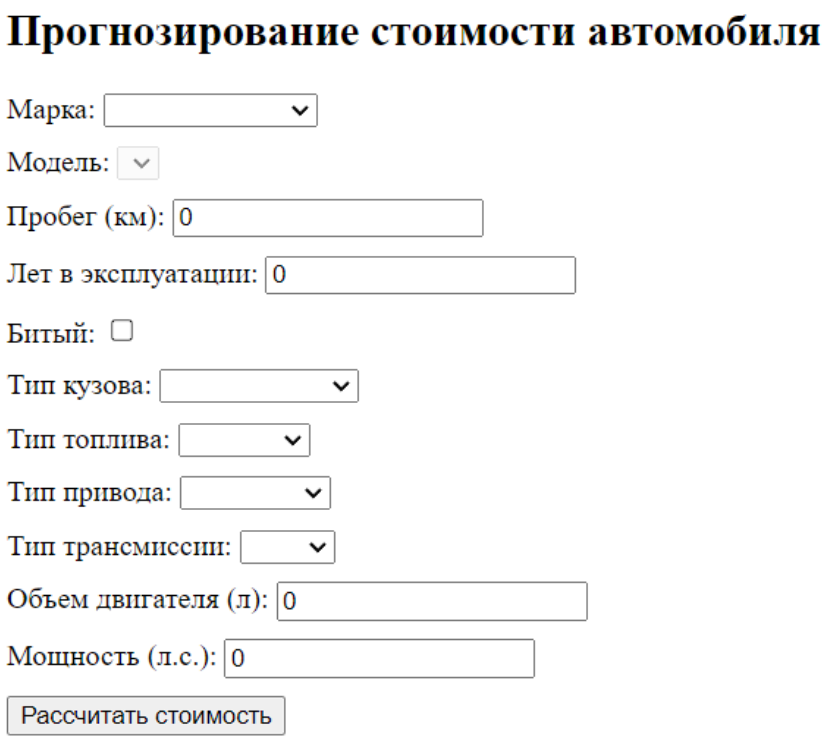

Рисунок 2 – форма ввода характеристик

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

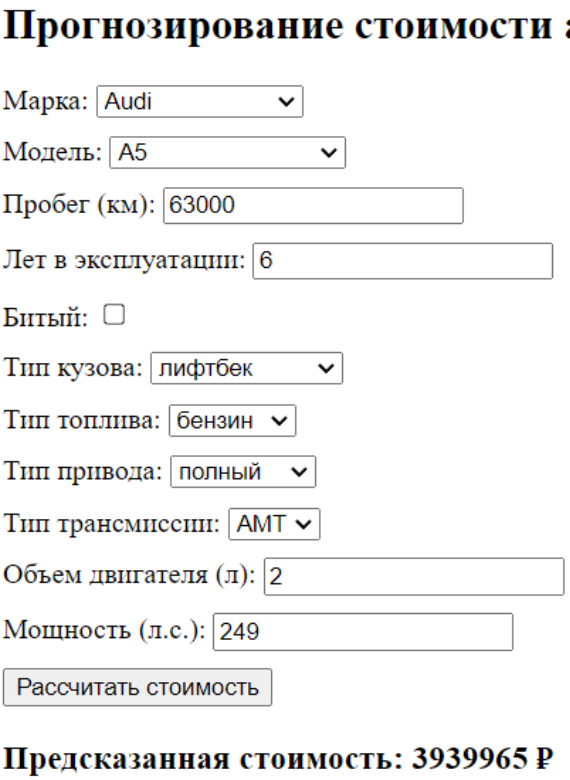

Рисунок 3 – результат прогноза

ЗАКЛЮЧЕНИЕ

В результате выполнения производственной практики (научно-исследовательской работы):

– исследованы современные подходы и технологии создания веб приложений с использованием универсальных программных средств;

– исследованы существующие методы и библиотеки для машинного обучения;

– изучены и освоены инструменты для создания веб-приложений;

– изучены и освоены инструменты для разработки моделей машинного обучения;

– протестированы различные методы прогнозирования и взят наилучший по выбранной метрике;

– подготовлен и оформлен письменный отчет по практике.

Таким образом, в процессе выполнения научно-исследовательской работы были освоены все необходимые индикаторы (ОПК-1.1, ОПК-1.2, ОПК-1.3, ОПК-2.1, ОПК-2.2, ОПК-2.3, ОПК-3.1, ОПК-3.2, ОПК-3.3, ОПК-4.1, ОПК-4.2, ОПК-4.3, ОПК-5.1, ОПК-5.2, ОПК-5.3, ОПК-6.1, ОПК-6.2, ОПК-6.3) компетенций (ОПК-1, ОПК-2, ОПК-3, ОПК-4, ОПК-5, ОПК-6).

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Шпак, А.А. Разработка современных веб-приложений [Текст]: учеб. пособие / А.А. Шпак, М.И. Ковалев. – М.: Изд-во МГУ, 2020. – 342 с.

2 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).

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

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

5 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).

6 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).

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

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

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

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