МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ВЯТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
ИНСТИТУТ МАТЕМАТИКИ И ИНФОРМАЦИОННЫХ СИСТЕМ
ФАКУЛЬТЕТ АВТОМАТИКИ И ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ
КАФЕДРА ЭЛЕКТРОННЫХ ВЫЧИСЛИТЕЛЬНЫХ МАШИН

Направление 09.03.01 - Информатика и вычислительная техника
(код и наименование направления)
Профиль – Программное и аппаратное обеспечение вычислительной техники

Допускаю к защите
Заведующий кафедрой ЭВМ
_____ / Долженкова М. Л. /
(подпись) *(Ф.И.О.)*

Разработка системы оценки и прогнозирования интересов клиентов автостоянки

Пояснительная записка выпускной квалификационной работы
ТПЖА 09.03.01.387 ПЗ

Разработал: студент гр.ИВТб-4301-04-00 _____ / Жеребцов К. А. / _____

Руководитель: к.т.н., доцент, зав. кафедры _____ / Долженкова М. Л. / _____

Нормоконтролер: к.т.н., доцент _____ / Скворцов А. А. / _____
(подпись) *(Ф.И.О)* *(дата)*

Киров 2024

Реферат

Жеребцов К. А. Разработка системы оценки и прогнозирования интересов клиентов автостоянки: ТПЖА 09.03.01.387 ПЗ ВКР / ВятГУ, каф. ЭВМ; рук. Долженкова М. Л.– Киров, 2024. – Гр.ч. 8л. ф.А1; ПЗ 102 с., 40 рис., 2 табл., 8 форм., 12 источников, 3 прил.

ИНТЕРЕСЫ, ПАРКОВКА, ПОСЕТИТЕЛИ, ПРЕДСКАЗАНИЕ, СИСТЕМА, PYTHON, TENSORFLOW.

Объект - маркетинговое исследование для улучшения качества обслуживания клиентов.

Предмет - прогнозирования интересов клиентов автостоянки.

Целью дипломного проекта является помощь маркетологам в понимании предпочтений владельцев автомобилей через автоматизированный сбор и анализ данных, что позволит разрабатывать более точные и персонализированные маркетинговые стратегии.

Задачи:

– автоматизация процесса сбора данных;
– распознавание интересов посетителей парковки;
– прогнозирование интересов посетителей парковки;

Результат - разработанная программная система для сбора информации о интересах клиентов и их прогнозироваения.

Содержание

Введение

В современном мире сбор и анализ данных играют ключевую роль в принятии решений во многих сферах деятельности. Благодаря развитию технологий, стало возможным эффективно использовать огромные объемы данных для улучшения различных процессов и повышения эффективности работы. В особенности, методы компьютерного зрения и искусственного интеллекта открывают новые возможности для автоматизации и оптимизации задач, которые ранее выполнялись вручную.

С развитием технологий обработки данных и машинного обучения, особенно нейронных сетей, стало возможным решать сложные задачи, такие как распознавание образов, анализ поведения, предсказания исходов каких-любо событи и так далее. Эти технологии находят широкое применение в самых разных областях, от маркетинга и розничной торговли до здравоохранения и транспорта.

Современные маркетологи сталкиваются с необходимостью глубоко понимать интересы и предпочтения своих клиентов, чтобы создавать эффективные стратегии и кампании. Проект по автоматизации процесса сбора и анализа интересов клиентов автостоянки предоставляет маркетологам мощный инструмент для достижения этой цели.

Благодаря полученным данным, маркетологи смогут более точно сегментировать аудиторию, разрабатывать персонализированные предложения и эффективно проводить рекламные кампании. Это приведет к улучшению взаимодействия с клиентами, увеличению их лояльности и, в конечном итоге, к росту продаж и прибыли компании.

Исходя из выше сказанного, было принято решение о создании системы, которая смогла бы автоматизировать процесс сбора и анализа интересов клиентов автостоянки, для того чтобы предоставить пользователям информацию о посетителях.

1 Анализ предметной области

На данном этапе работы будет обоснована актуальность темы. Также необходимо рассмотреть основные области, которые затрагивает разрабатываемая система. Сюда входят такие темы, как методы сбора данных, а также методы их анализа и обработки. Кроме того, нужно описать, почему важна персонализация услуг и предложений, на основе интересов клиентов.

1.1 Актуальность темы

Сбор и анализ данных о предпочтениях владельцев автомобилей представляет собой важный аспект для улучшения качества обслуживания и удовлетворения их потребностей. Традиционные методы исследований, такие как опросы и анкетирование, часто требуют значительных усилий и временных затрат для обработки и анализа полученных данных. Однако, использование современных методов машинного обучения и анализа данных открывает новые возможности для автоматизации этого процесса.

Данный проект направлен на создание модуля, способного автоматически анализировать предпочтения владельцев автомобилей на основе данных о модели автомобиля и возрасте. Этот подход позволяет существенно сократить время, затрачиваемое на сбор и обработку данных, а также предоставляет более точную и полную информацию о предпочтениях владельцев.

В условиях растущей конкуренции в автомобильной индустрии, эффективное использование данных о предпочтениях клиентов становится ключевым конкурентным преимуществом. Автоматизированный анализ интересов владельцев автомобилей позволит нам лучше понять их потребности, оптимизировать наши услуги и предложения, и разработать персонализированные подходы для каждого клиента.

Ключевым аспектом проекта является определение наиболее подходящих методов сбора данных. Это включает в себя исследование различных источников информации, таких как базы данных регистраций автомобилей, данные из соци-

альных сетей, онлайн-опросы и другие методы. Важно выбрать те методы, которые обеспечат наибольшую точность и актуальность получаемых данных, а также будут соответствовать требованиям конфиденциальности и защиты информации.

После сбора данных необходимо обеспечить их качественную обработку и анализ. Это включает очистку данных, устранение пропусков и ошибок, а также применение методов машинного обучения и аналитических инструментов для выявления закономерностей и трендов. Важным аспектом является также визуализация данных, которая позволит легко интерпретировать результаты анализа и принимать обоснованные решения на их основе.

Таким образом, разработка модуля для анализа предпочтений владельцев автомобилей на основе современных методов машинного обучения представляет собой актуальное и перспективное направление в области обслуживания клиентов.

1.2 Методы сбора данных

Сбор данных является процессом, включающим измерение и накопление информации о важных переменных, что позволяет формулировать и решать различные исследовательские вопросы. Этот процесс занимает ключевое место в образовательных программах по маркетингу, статистике, экономике, естественным наукам и другим дисциплинам. Методы сбора информации могут варьироваться в зависимости от особенностей предмета, однако цель исследования и добросовестность в этом процессе остаются одинаково значимыми во всех областях знаний. Сбор данных имеет решающее значение, так как позволяет принимать обоснованные и систематические решения. Данные служат незаменимым инструментом для принятия обдуманных решений, экономя время и ресурсы [1].

В зависимости от характера сбора данных их можно разделить на два основных типа, а именно [1]:

– первичный метод;
– вторичный метод.

Первичные данные исследователи получают самостоятельно и впервые в

контексте своего исследования. Существуют несколько способов сбора первичных данных [1]:

– интервью. Интервью являются наиболее распространенным методом первичного сбора данных. В ходе интервью исследователь использует анкету или задает вопросы непосредственно респонденту. Основная цель состоит в том, чтобы получить информацию по интересующим темам из ответов участников. Анкеты могут быть отправлены по электронной почте, либо детали могут быть уточнены в ходе телефонного интервью;

– метод Дельфи. В этом методе исследователь обращается за информацией к группе экспертов. Он может провести исследование очно или отправить анкеты по электронной почте. По завершении применения метода Дельфи все данные собираются и обрабатываются в соответствии с потребностями исследования;

– проективные методы. Проективные методы применяются в исследованиях, где вопросы касаются частной или конфиденциальной информации, и исследователь полагает, что респонденты не раскроют данные при прямом опросе. Существует множество типов проективных методик, таких как тематические апперцептивные тесты, ролевые игры, завершение мультфильмов, словесные ассоциации и завершение предложений;

– интервью в фокус-группе. Для обсуждения насущной проблемы собирается группа из нескольких человек. В таких интервью обычно участвует от шести до двенадцати человек. Каждый участник высказывает свое мнение, и в итоге принимается коллективное единогласное решение;

– метод анкетирования. Здесь используется вопросник для сбора данных от различных групп населения. Для сбора данных от различных групп населения используется вопросник. В соответствующем исследовании применяется набор вопросов, на которые респонденты отвечают, прямо или косвенно связанные с темой вопросника.

Эти методы позволяют анализировать большие объемы информации и выявлять статистически значимые закономерности, что способствует более обоснованным выводам и рекомендациям.

Вторичные данные означают данные, которые исследователь собирает не самостоятельно и в раз. Фактически, вторичные данные уже доступны и их необ-

ходимо собирать из различных источников [1].

Некоторые источники данных, которые могут быть использованы для вторичного сбора данных, включают [1]:

– газеты;
– журналы;
– публичные записи;
– деловые документы;
– исторические и статистические документы и т.д.

Помимо упомянутых выше опубликованных источников данных, для вторичного сбора информации могут использоваться неопубликованные данные, такие как письма, дневники и неопубликованные биографии [1].

Количественные методы первичного сбора данных могут быть различных типов, некоторые из которых следующие [1]:

– вероятностная выборка. В этом методе в целевой совокупности используется форма случайного отбора. Затем на основе данных, полученных от случайной выборки, делаются вероятностные утверждения о генеральной совокупности. Преимущество случайной выборки заключается в том, что исследователи могут собирать данные от представителей, а не от всей популяции, что позволяет получить в основном беспристрастные данные;

– интервью. Интервью также являются отличным способом сбора первичных данных. Собеседования могут проводиться тремя способами: по телефону, очно и с использованием компьютера. Цель интервью состоит в быстром и эффективном получении объективных персональных данных о респондентах;

– опросы. Анкетирование также представляет собой отличный метод сбора первичных данных. Анкеты могут быть распространены по электронной почте или через интернет. Основная цель опросов - быстрое получение данных в форме, которую легко проверить. Это также очень простой способ собрать первичные данные от населения;

– наблюдения. Наблюдения также представляют собой очень простой способ сбора количественных данных. В процессе наблюдения за группой населения в определенной демографической зоне собираются данные, связанные с первичным исследованием.

Теперь поговорим про методы сбора качественных данных. Многие методы сбора данных могут быть использованы как для сбора количественной, так и для сбора качественной информации. Интервью, например, могут быть структурированными, чтобы собирать количественные данные, или полуструктурированными или неструктурированными, чтобы собирать более глубокие качественные данные. То же самое относится и к наблюдениям: они могут быть использованы для сбора как количественных, так и качественных данных в зависимости от методологии и целей исследования [1].

Опросы, групповые обсуждения, наблюдения и интервью являются часто используемыми методами сбора качественных данных. Кроме того, такие методы, как методика Дельфи и проективные методики, также могут быть использованы для сбора качественных данных в зависимости от специфики исследования и целей исследователя.

Качественные данные, часто представленные в виде текстовой информации, играют важную роль в исследованиях и анализе. Поддержание допустимой погрешности в качественных данных существенно для получения достоверной картины результатов исследования. В то же время количественные данные должны быть максимально точными, поскольку они непосредственно влияют на результаты исследования. Такое разделение помогает исследователям адекватно оценивать и интерпретировать данные в соответствии с целями и характером исследования [1].

1.3 Обработка и анализ данных

Обработка и анализ данных - это критически важный этап, включающий преобразование и толкование информации с целью выявления закономерностей, паттернов и тенденций. С увеличением объема и разнообразия данных в современном мире становится необходимым использование разнообразных методов и инструментов для их полноценного анализа. Это включает в себя как классические статистические методы, так и современные техники машинного обучения и анализа больших данных. Грамотный анализ данных позволяет выявить скрытые

закономерности и важные тренды, что в свою очередь помогает принимать обоснованные решения и формулировать стратегии на основе фактических данных [2].

1.3.1 Методы предобработки данных

Предобработка данных играет важную роль в процессе обработки информации, нацеленной на подготовку данных для дальнейшего анализа. Она включает в себя процессы, такие как очистка данных от выбросов и ошибок, заполнение пропущенных значений, масштабирование признаков и преобразование категориальных признаков в числовые. Эти шаги направлены на создание более чистого и пригодного для анализа набора данных, что способствует более точному и надежному анализу и интерпретации результатов [2].

Предварительная обработка данных включает выполнение следующих этапов [2]:

1) проверка целостности данных;
2) очистка данных от ошибок и аномалий;
3) преобразование данных в удобный формат;
4) добавление новых данных для повышения информативности;
5) улучшение эффективности обработки данных.

Проверка данных включает в себя обнаружение [2]:

– дублирующихся записей, несоответствий и ошибок;
– необычных или аномальных наблюдений;
– пропущенных значений.

Очистка данных включает [2]:

– удаление повторяющихся записей, несоответствий и ошибок;
– обработку необычных или аномальных наблюдений;
– заполнение пропущенных значений.

При работе с данными важно правильно обрабатывать пропущенные значения. Этот процесс является важным этапом работы с данными. Удаление пропусков может быть необходимым, но такой подход требует внимательного анализа и оценки влияния на результаты исследования. В некоторых случаях удаление может привести к искажению данных или потере важной информации. Пропускание пропущенных значений - еще один способ обработки, который может быть приме-

нен, если пропуски не существенно влияют на анализ. Заполнение пропущенных значений - еще одна распространенная стратегия, которая позволяет сохранить данные и избежать искажений. Все эти методы требуют осторожности и обоснованного подхода в зависимости от конкретной ситуации и характера данных. Пропущенные значения можно заполнить [2]:

– нулевыми значениями;
– модой, медианой или средним значением;
– использованием индикаторных переменных.

Преобразование данных включает в себя [2]:

– изменение названий признаков;
– сортировку и группировку данных;
– кодирование переменных;
– нормализацию данных.

Добавление данных предполагает создание новых признаков и объединение существующих. Улучшение эффективности обработки данных включает [2]:

– сокращение размерности;
– идентификацию и исключение незначительных признаков.

1.3.2 Методы визуализации данных

Визуализация данных играет ключевую роль в исследовании и анализе данных, помогая выявить визуальные закономерности и тренды. Она позволяет представить данные в понятной и наглядной форме с использованием различных графических элементов, таких как графики, диаграммы, и карты. Визуализация делает данные более доступными для понимания и интерпретации как для специалистов, так и для широкой аудитории. Этот инструмент позволяет исследователям и аналитикам обнаруживать важные закономерности и взаимосвязи в данных, что в свою очередь помогает принимать обоснованные решения и формулировать стратегии на основе данных [3].

Существует несколько видов визуализации [3]:

– обычные графические представления. Он включает в себя различные виды диаграмм и графиков, используемых для отображения количественной информации. Круговые диаграммы, например, используются для представления долей целого, линейные диаграммы отображают изменение величин с течением вре-

мени, гистограммы представляют распределение данных по интервалам, а точечные графики позволяют визуализировать взаимосвязи между двумя переменными. Эти стандартные методы визуализации являются основой для анализа данных и широко используются в научных и бизнес-исследованиях для визуального представления количественной информации;

– преобразование данных. Он позволяет изменить формат данных для более наглядного восприятия. Например, карты используются для визуализации пространственных данных, а полярные графики отображают данные в полярной координатной системе. Также сюда относятся временные линии, которые представляют изменение данных во времени, и графики с параллельными осями, используемые для сравнения нескольких категорий данных. Эти методы визуализации помогают улучшить понимание данных и выявить закономерности, которые могут быть невидимы при простом представлении данных;

– концептуальная визуализация. Она используется для представления сложных концепций, идей и планов. К этому типу визуализации относятся концептуальные карты, которые помогают организовать и визуализировать связанные идеи, и диаграммы Ганта, используемые для планирования и отслеживания проектов с учетом временных рамок. Эти инструменты помогают упростить восприятие и понимание сложных идей и процессов, делая их более доступными для анализа и интерпретации;

– стратегическая визуализация. Этот тип визуализации включает в себя различные диаграммы, такие как графики жизненного цикла продукта, которые отображают этапы развития продукта, и структурные графики организаций, показывающие их иерархию и взаимосвязи. Эти инструменты помогают руководителям и аналитикам лучше понимать внутренние процессы, оценивать эффективность и разрабатывать стратегические планы;

– метафорическая визуализация. Это способ представления данных с помощью метафор, чтобы сделать их более понятными. Например, карта метро может показать, как разные категории информации связаны друг с другом, а пирамиды или деревья могут продемонстрировать иерархические структуры. Этот подход помогает легче воспринимать сложные данные, используя знакомые образы;

– комбинированная визуализация. Этот тип визуализации объединяет

разные виды графиков и диаграмм в одном изображении, чтобы более полно представить данные. Например, карта прогноза погоды может показывать одновременно температуру, осадки и ветер. Такой способ помогает лучше понять данные, поскольку вся информация представлена вместе и легко доступна для анализа.

Визуализация помогает быстрее и эффективнее донести результаты анализа до заинтересованных сторон, улучшая понимание и принятие решений.

1.3.3 Методы машинного обучения

Корни технологии машинного обучения, основанной на анализе данных, уходят еще в 1950-е годы, когда появились первые программы для игры в шашки. С течением времени, при сохранении общего принципа, эта область претерпела значительные изменения благодаря росту вычислительной мощности компьютеров. Это привело к усложнению методов анализа данных и прогнозирования, а также к расширению спектра задач, которые можно решить с использованием машинного обучения [4].

Процесс машинного обучения начинается с загрузки исходных данных, известных как датасет. Например, это могут быть помеченные фотографии собак и кошек. После обучения на этих данных алгоритм способен распознавать собак и кошек на новых изображениях без пометок. Процесс обучения не заканчивается после первых прогнозов: чем больше данных анализируется, тем выше точность модели в распознавании нужных изображений [4].

Машинное обучение позволяет компьютерам не только распознавать лица на фотографиях, но и анализировать пейзажи, объекты, текст и цифры. Для анализа текста также используются методы машинного обучения. Функция проверки грамматики, доступная во многих текстовых редакторах и мобильных приложениях, учитывает не только правильность написания слов, но и контекст, смысловые оттенки и другие лингвистические аспекты. Кроме того, существует программное обеспечение, способное автоматически создавать новостные статьи (например, о событиях в экономике и спорте) без участия человека [4].

Типы задач машинного обучения [4]:

– задача регрессии. Суть заключается в предсказании числового значения на основе определенного набора признаков. Например, это может быть прогноз цены квартиры на основе таких факторов, как площадь, количество комнат, рас-

положение и другие характеристики. Также задача регрессии может применяться для предсказания ожидаемого дохода магазина на основе таких факторов, как расположение, количество посетителей, временной период и прочее. Основная цель регрессионного анализа - построить модель, которая бы максимально точно предсказывала зависимую переменную на основе доступных признаков;

– задача классификации. Суть заключается в присвоении объекту одной из категорий на основе его признаков. Например, это может быть определение, есть ли на фотографии кошка или собака, или является ли письмо спамом или не спамом. Для решения задачи классификации используются алгоритмы машинного обучения, которые обучаются на размеченных данных и способны классифицировать новые объекты на основе изученных характеристик. Основная цель классификационного анализа - построить модель, которая бы максимально точно разделяла объекты на заданные классы;

– задача кластеризации. Суть заключается в разделении данных на группы или кластеры на основе их сходства. Например, в контексте разделения клиентов по уровню платежеспособности, алгоритм кластеризации может автоматически выделить группы клиентов с похожими характеристиками или поведением в отношении платежей. Это позволяет более эффективно анализировать и работать с данными, выявляя закономерности и тренды внутри каждой группы;

– задача уменьшения размерности. Суть заключается в сокращении количества признаков в данных с сохранением максимально возможной информации. Например, это может быть сжатие данных для удобства визуализации на графиках. При использовании методов уменьшения размерности данные проецируются из исходного пространства высокой размерности в пространство меньшей размерности, при этом стараясь сохранить как можно больше вариативности и структуры данных. Это позволяет сократить количество признаков для анализа данных, уменьшить их размер и облегчить визуализацию без значительной потери информации;

– задача выявления аномалий. Суть заключается в обнаружении нестандартных или нетипичных случаев среди данных. Например, это может быть выявление мошеннических операций с банковскими картами среди обычных финансовых транзакций. Для решения этой задачи используются методы машинного

обучения, которые строят модели на основе нормального поведения данных и выявляют аномалии как объекты, сильно отклоняющиеся от этого нормального шаблона. Выявление аномалий имеет важное значение в областях безопасности, финансов, здравоохранения и других, где даже небольшие отклонения могут иметь серьезные последствия.

Методы машинного обучения можно разделить на два основных класса: обучение с учителем и обучение без учителя [4].

В случае обучения с учителем модели предоставляются правильные ответы для обучающих данных. Эти данные состоят из пар ”входные признаки - правильные ответы”. Модель обучается на таких данных с целью научиться предсказывать правильные ответы для новых, ранее не виданных данных [4].

В случае обучения без учителя модели не предоставляются правильные ответы. Модель обучается только на основе входных данных, без информации о правильных ответах. Задача модели здесь состоит в том, чтобы самостоятельно выявить закономерности в данных, например, кластеризовать их или находить структуры и шаблоны, которые могут быть полезны для дальнейшего анализа или решения задач [4].

Основные алгоритмы моделей машинного обучения [4]:

1) дерево принятия решений. Дерево принятия решений - это инструмент, который помогает систематизировать процесс принятия решений. В его основе лежит древовидная структура, где каждый узел представляет собой вопрос, а каждая ветвь - возможный ответ на этот вопрос. Этот метод особенно полезен в бизнесе, где решения часто принимаются на основе конкретных критериев и оценки потенциальных последствий. Пройдя через последовательность вопросов и ответов, можно прийти к логически обоснованному решению;
2) наивная байесовская классификация. Наивная байесовская классификация - это метод, который используется для разделения данных на категории, основываясь на теореме Байеса и предполагая, что признаки независимы друг от друга. Он применяется в различных областях, включая фильтрацию спама, автоматическую категоризацию текстов, анализ эмоциональной окраски текста, а также в распознавании объектов

и лиц на изображениях. Основная идея заключается в том, что каждый признак вносит свой вклад в классификацию независимо от других признаков, делая метод простым и эффективным в некоторых случаях;

3) метод наименьших квадратов. Метод наименьших квадратов - это способ, используемый в линейной регрессии, чтобы найти прямую, которая наилучшим образом соответствует набору данных. Он основан на минимизации суммы квадратов расстояний от каждой точки данных до этой прямой. В контексте машинного обучения, этот метод помогает уменьшить ошибку при аппроксимации данных и настройке моделей под имеющиеся данные;
4) логистическая регрессия. Логистическая регрессия - это метод анализа, который используется для определения связи между независимыми переменными и категориальной зависимой переменной. В отличие от линейной регрессии, где зависимая переменная является непрерывной, в логистической регрессии зависимая переменная обычно представляет собой бинарную или категориальную переменную. Этот метод широко применяется в различных областях, таких как кредитный скоринг, анализ результатов маркетинговых кампаний, прогнозирование успешности продуктов и других событий. Он помогает предсказывать вероятность наступления определенного события на основе значений независимых переменных;
5) метод опорных векторов (SVM). Метод опорных векторов - это набор алгоритмов машинного обучения, который используется для классификации и регрессионного анализа данных. Он работает путем построения гиперплоскости в многомерном пространстве, которая наилучшим образом разделяет точки между двумя классами. SVM пытается максимизировать расстояние между этой гиперплоскостью и ближайшими точками обучающего набора, называемыми опорными векторами.
6) метод ансамблей. Метод ансамблей в машинном обучении основан на идее создания нескольких классификаторов, которые затем объединяются для принятия решения. Этот подход позволяет уменьшить случайные ошибки и дисперсию модели, а также избежать выхода за пределы

предполагаемых гипотез. Изначально метод представлял собой байесовское усреднение, но с течением времени стал более сложным благодаря использованию различных алгоритмов. Он может использовать разнообразные техники, такие как бэггинг, случайный лес, градиентный бустинг и другие, чтобы повысить качество прогнозов;

7) алгоритмы кластеризации. Кластеризация - это метод, позволяющий объединить объекты в группы таким образом, чтобы объекты внутри одной группы были максимально похожи друг на друга. Алгоритмы кластеризации находят применение в различных областях, включая биологию, социологию и информационные технологии. Например, в биологии они используются для классификации видов или для анализа геномных данных, в социологии - для выявления социальных группировок, а в информационных технологиях - для сегментации пользователей или анализа текстовой информации;
8) метод главных компонент (PCA). PCA представляет собой статистический метод, который используется для уменьшения размерности данных и визуализации. Он работает путем ортогонального преобразования исходных признаков в новый набор признаков, называемых главными компонентами. Главная цель PCA - сохранить как можно больше информации, содержащейся в исходных данных, при этом снижая количество признаков. Это помогает упростить процесс анализа данных, улучшить производительность моделей машинного обучения и сократить вычислительные затраты;
9) сингулярное разложение. Сингулярное разложение - это метод линейной алгебры, который разлагает матрицу на три составляющие: левую сингулярную матрицу, правую сингулярную матрицу и диагональную матрицу сингулярных значений. В контексте анализа данных, SVD используется для сжатия данных и уменьшения размерности, что позволяет выделить основные компоненты информации и убрать шумы или ненужные детали, сохраняя при этом основные характеристики данных. Этот метод является основой для таких техник, как метод главных компонент (PCA), и находит широкое применение в различных об-

ластях, включая обработку изображений, анализ текстов и рекомендательные системы;

10) анализ независимых компонент (ICA). Анализ независимых компонент - это метод, который помогает выявить скрытые причины или источники влияния на данные и разделить их на независимые компоненты. Он ищет такие независимые составляющие в наборе данных, которые могут быть полезны для понимания его структуры и свойств. Этот метод особенно полезен в областях, где данные могут быть представлены в виде комбинации различных воздействий или источников.

Методы машинного обучения рассматриваются в проекте для выбора оптимального подхода к анализу данных, что позволяет максимально эффективно и точно решать поставленные задачи.

1.3.4 Персонализация услуг и предложений

В современном мире персонализация услуг и предложений стала краеугольным камнем для бизнеса. Это не просто стратегия, а неотъемлемая часть успешной работы компаний, которая позволяет им лучше понимать и удовлетворять потребности своих клиентов. Главное в этом подходе - создание индивидуального опыта для каждого клиента, учитывая его уникальные характеристики и предпочтения [5].

Конечная цель персонализации - предоставление клиентам наиболее релевантных и ценных продуктов и услуг. Для этого компании активно анализируют данные о своих клиентах, стремясь выявить их индивидуальные потребности и предложить соответствующие решения [5].

Примером может служить использование данных о предпочтениях клиентов для создания персонализированных рекомендаций продуктов или услуг. Если клиент интересуется определенными темами или категориями товаров, компания предлагает ему соответствующие продукты, что может сильно повысить удовлетворенность клиента и вероятность его повторных покупок [5].

В области обслуживания клиентов персонализация также играет решающую роль. Здесь компании стремятся предоставить клиентам индивидуализированный опыт взаимодействия, что включает в себя персонализированные рекомендации, обратную связь и поддержку. Этот подход позволяет укрепить связь с

клиентом и повысить его лояльность к бренду [5].

Однако важно помнить, что успешная персонализация невозможна без использования данных для оптимизации маркетинговых усилий и ресурсов компании. Точное нацеливание коммуникации на потребности и интересы клиентов помогает повысить эффективность маркетинговых кампаний и улучшить общее взаимодействие с ними [5].

1.4 Расширенное техническое задание

Для нагляного отображения доступного пользователю функционала используются диаграмма вариантов использования, которая изображена на рисунке 1.

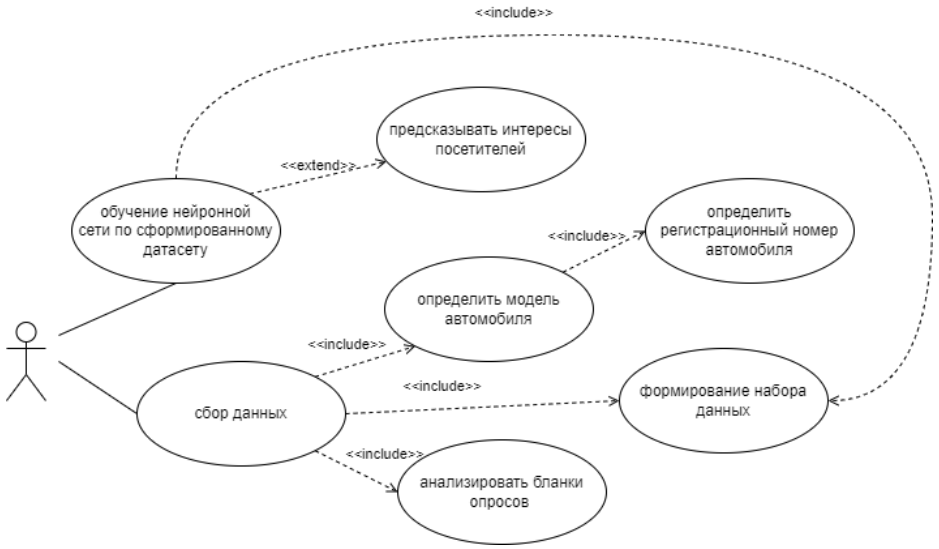

Рисунок 1 – Диаграмма вариантов использования

1.4.1 Основание для разработки

Программа разрабатывается на основе учебного плана кафедры “Электронные вычислительные машины” по направлению 09.03.01.

1.4.2 Цель и задача

Название темы: «Разработка системы оценки и прогнозирования интересов клиентов автостоянки».

Объект - маркетинговое исследование для улучшения качества обслужива-

ния клиентов.

Предмет - прогнозирования интересов клиентов автостоянки.

Целью дипломного проекта является помощь маркетологам в понимании предпочтений владельцев автомобилей через автоматизированный сбор и анализ данных, что позволит разрабатывать более точные и персонализированные маркетинговые стратегии.

Задачи:

– автоматизация процесса сбора данных;
– распознавание интересов посетителей парковки;
– прогнозирование интересов посетителей парковки;

Результат - разработанная программная система для сбора информации о интересах клиентов и их прогнозироваения.

1.4.3 Наименование и область применения

Наименование разрабатываемого программного обеспечения: система оценки и прогнозирования интересов клиентов автостоянки.

Область применения: прогнозирование интересов посетителей парковки для улучшения сервиса.

1.4.4 Требования к функциональности

Программа должна обеспечивать возможность выполнения следующих функций:

– определять регистрационный номер автомобиля;
– определять модель автомобиля
– анализировать бланки опросов определенного формата;
– формирование набора данных в файл типа CSV;
– обучение нейронной сети по сформированному датасету;
– предсказывать интересы посетителей с использование обученной модели.

1.4.5 Требования к нейронной сети

Нейронная сеть должна быть реализована на базе языка программирования Python с использованием библиотеки Tensorflow, Keras.

1.4.6 Требования к бланку

Бланк должен содержать вопросы и варианты ответа на вопросы (ДА/НЕТ), а также поле для ввода возраста. Пример представлен на рисунке 2.

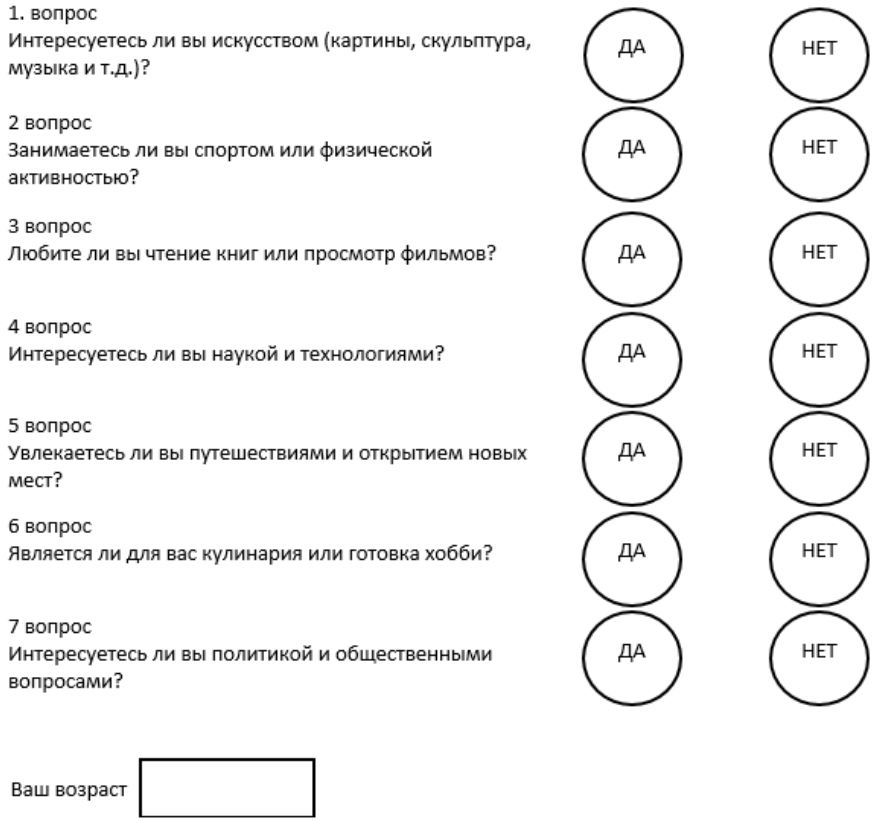

Рисунок 2 – Бланк опроса

1.4.7 Требования к программной документации

Состав программной документации должен включать в себя:

– техническое задание;

– пояснительная записка, содержащая описание разработки;

– руководство пользователя.

Разрабатываемые программные модули должны быть самодокументированы, т.е. тексты программ должны содержать все необходимые комментарии.

1.4.8 Технико-экономические характеристики

Реализованное программное обеспечение использует бесплатную модель распространения и разрабатывается с помощью бесплатных решений.

Выводы

В данном разделе был проведен анализ предметной области, подчеркнута актуальность разработки модуля для анализа интересов посетителей парковки. Система представляет собой средство сбора информации об интересах и более точного и автоматизированного предсказания интересов посетителей парковки. Определение интересов клиентов позволяет оптимизировать уровень сервиса, а также разрабатывать персонализированные предложения для посетителей. Также составлено техническое задание. Были составлены требования к разрабатываемой системе, а также определены основные функции.

2 Разработка структуры системы

В данном разделе описана структура разрабатываемой системы, а также модули из которых она состоит. Для кажого модуля определяются входные и выходные данные, так же описываестя взаимосвязь моудлей. Кроме того, в разделе рассказывается про готовые технологии, которые используются модулями в процессе своего функционирования.

Разрабатываемая система состоит из 2 модулей, каждый из которых может работать обособленно, при наличии необходимых ресурсов:

– модуль для сбора информации;
– модуль для обучения нейросети и предсказания интересов.

Необходимо сформировать наиболее точное описание разрабатываемого программного обеспечения. Для этого было принято решение о рассмотрении функциональной диаграммы верхнего уровня. В данном случае в качестве отображения взаимосвязей была выбрана нотация IDEF0. В качестве входных данные для сбора информации (бланк опроса, изображение автомобиля с номером) и данные для предсказания интересов (возраст владельца и модель авто). В качестве субъекта выступает пользователь и вычислительная машина. Управление задается алгоритмами для сбора данных, алгоритм обучения нейросети, алгоритм работы с обученной моделью. К выходным данным относятся датасет, обученная модель и предсказанные интересы.

Контекстная диаграмма IDEF0 представлена на рисунке 3.

На детализирующей функциональной диаграмме показаны следующие этапы:

– сбор информации;
– предсказание интересов.

Детализированная контекстная диаграмма представлена на рисунке 4

Как видно из схемы, разрабатываемая система состоит из 2 модулей: сбора информации и предсказания интересов. Далее рассмотрим каждый из них подробнее.

Рисунок 3 – Контекстная диаграмма IDEF0

2.1 Разработка структуры компонента для сбора информации

В данном разделе будет детально рассмотрена структура модуля, его функции и алгорритмы функционирования. В разделе описываются компоненты, из которых состоих сам модуль, а так же связь между ними.

2.1.1 Определение ключевых функций

Разрабатываемый модуль состоит из трех компонентов. Рассмотрим функционал каждого подробнее.

2.1.1.1 Сбор и обработка данных

Модуль должен быть способен автоматически собирать опросные данные с заполненных анкет посетителей парковки.

Был определен ряд вопросов, который позволяют определить некоторые основные интересы:

– интересуетесь ли вы искусством (картины, скульптура, музыка и т.д.);
– занимаетесь ли вы спортом или физической активностью;
– любите ли вы чтение книг или просмотр фильмов;
– интересуетесь ли вы наукой и технологиями;
– увлекаетесь ли вы путешествиями и открытием новых мест;
– является ли для вас кулинария или готовка хобби;

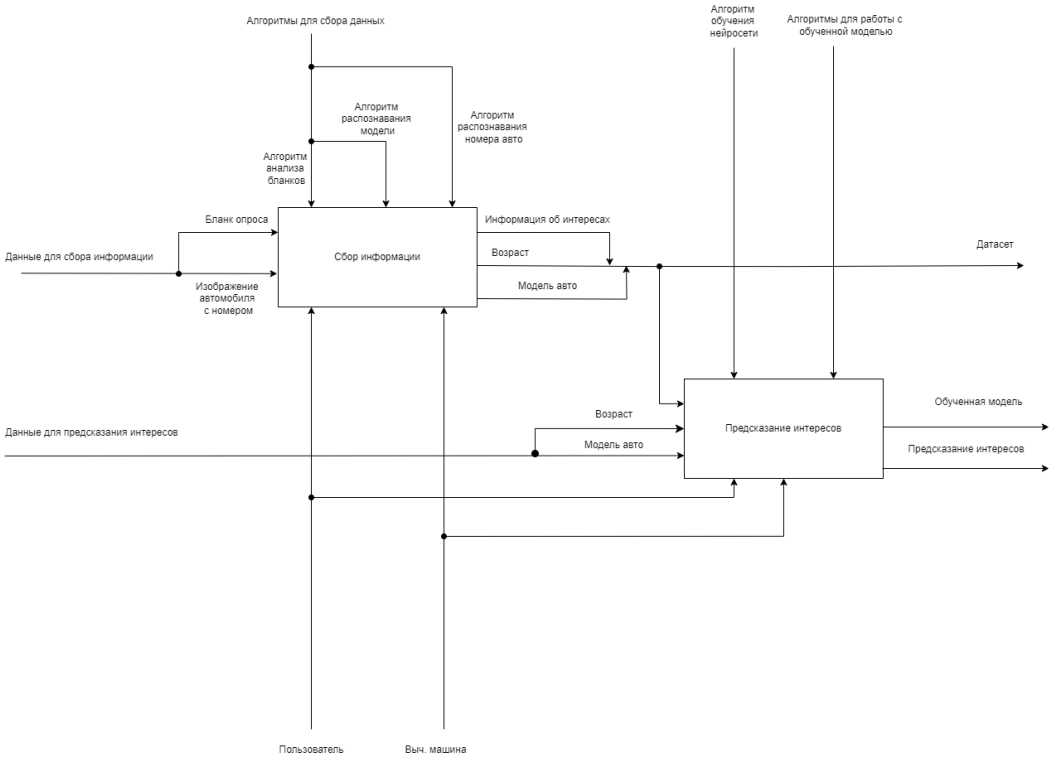

Рисунок 4 – Детализированная контекстная диаграмма IDEF0

– интересуетесь ли вы политикой и общественными вопросами.

Так же на бланке (представлен на рисунке 2 выше) присутствует поле, в котором опрашиваемый должен указать свой возраст.

Опрашиваемый должен закрасить кружки, соответствующие его ответу на каждый вопрос, а также указать свой возраст.

Бланк на вход должен подаваться в виде фотографии или скан копии.

Присутсвует возможность изменить блнак. Вопросы пользователь может выбрать произвольно, главное, чтобы они были представленны в определенном в примере формате. Для изменения областей интересов, с которыми работает система, необходимо внести их списком в текстовый файл. Далее система будет подгружать их в таблицу. Сам файл хранится в папке конфигурации по пути \cfg от корневой папки и выглядит следущим образом (пример файла привен на рисунке 5).

После сбора данных необходимо провести их предварительную обработку, включая распознавание текста и извлечение ответов на вопросы опроса.

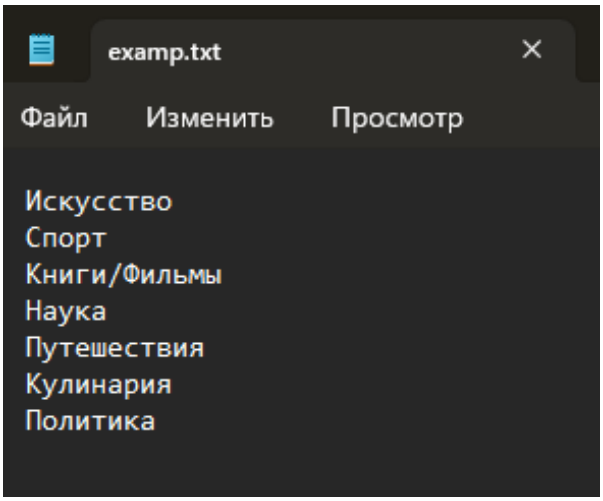

Рисунок 5 – Пример файла с конфигурацией областей интересов

2.1.1.2 Распознавание номеров автомобилей

Модуль должен обеспечивать распознавание номеров автомобилей с изображений, полученных с камер видеонаблюдения на парковке.

Разрешение входного изображения должно быть не ниже 1280x720. Кроме того, распознаваемый номер должен быть читаем, то есть быть чистым, не выцветшим и не закрыт посторонними предметами. Это позволит повысить точность распознавания и сократить количество ошибок.

После распознавания номеров необходимо извлечь информацию о модели автомобиля и ассоциировать её с данными о посетителе.

2.1.1.3 Поиск модели автомобиля

Данный модуль должен по распознанному номеру найти модель автомобиля. Данная процедура должна выполнятся с помощью сторонних сервисов, например ”Номерограмм.ру которые позволяют получить базовую информации об автомобиле, на основе WIN-номера или гос. регистрационного знака.

2.1.2 Структура модуля

Необходимо сформировать наиболее точное описание разрабатываемого программного обеспечения. Для этого было принято решение о рассмотрении функциональной диаграммы верхнего уровня.

В данном случае в качестве отображения взаимосвязей была выбрана нота-

ция IDEF0. В качестве входных бланк опроса и изображение автомобиля. В качестве субъекта выступает пользователь и вычислительная машина. Управление задается алгоритмами анализа бланков, распознавания номера и получения модели автомобиля. К выходным данным будут относиться информация об интересах посетителя, его возраст и модель машины. Контекстная диаграмма IDEF0 представлена на рисунке 6.

Рисунок 6 – Контекстная диаграмма IDEF0

Детализирующая функциональная диаграмма более подробно раскрывает функциональную диаграмму верхнего уровня: описывает взаимодействия и связи процессов, происходящих внутри системы. На ней можно увидеть, какие процессы взаимосвязаны и что между ними общего. На детализирующей функциональной диаграмме показаны следующие этапы:

– анализ бланков. Входными данными являются бланки опросов. На выход поступает информация об интересах и возраст опрашиваемого;

– распознавание номеров. Входными данными являются изображения автомобилей. На выход поступает государственный номер автомобиля;

– определение модели. На вход поступает номер автомобиля, а на выход идет модель машины.

Детализированная контекстная диаграмма представлена на рисунке 7.

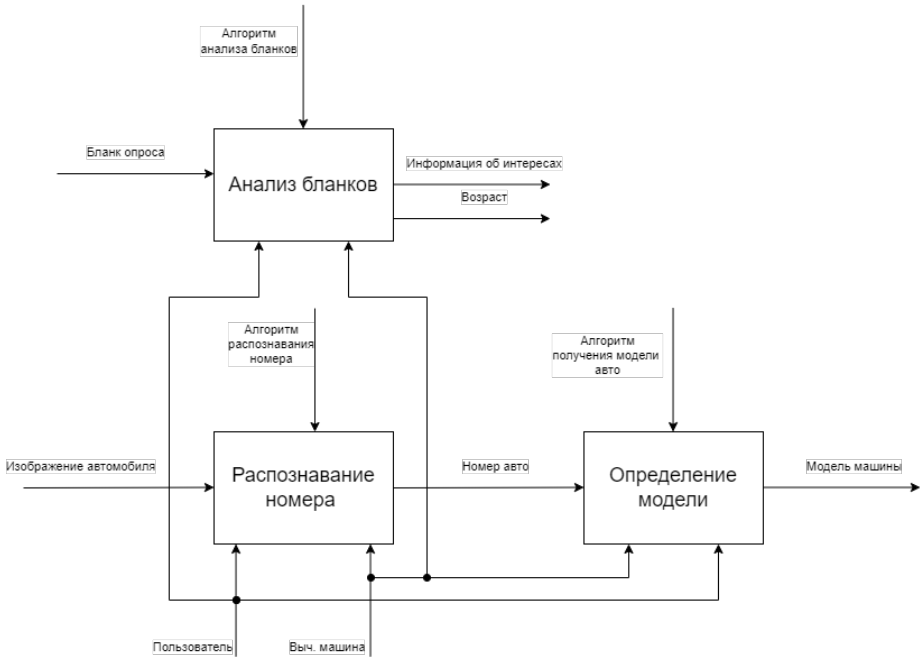

Рисунок 7 – Детализированная контекстная диаграмма IDEF0

Таким образом, разрабатываемый модуль будет состоять из трех основных компонентов: «Анализ бланков», «Определение модели», «Распознавание номера».

Так же стоит определить формат создаваемого датасета. В него будет входить вся получаемая информация. Следовательно, датасет будет иметь следующие поля:

– model: модель автомобиля;
– age: возраст;
– art: интерес к искусству;
– sport: интерес к спорту;
– book/film: интерес к фильмам и книгам;
– science: интерес к науке и технологиям;
– travel: интерес к путешествиям;
– cooking: интерес к готовке;
– politics: интерес к политике.

2.1.3 Анализатор бланков

Основными функциями компонента являются функции для анализа анкет, сбора информации об ответах и возрасте, указанном на бланке.

Задача компонента заключается в анализе анкет. Он должен автоматически обрабатывать изображения анкет, извлекать информацию об ответах на вопросы и возрасте, указанном на бланке.

Для выполнения задачи выбраны следующие инструменты:

– OpenCV: для обработки изображений, выделения контуров и преобразований перспективы;

– imutils: для удобной работы с изображениями, сортировки контуров и других операций;

– pytesseract: для распознавания текста на изображениях с использованием технологии OCR.

На вход компоненту подается изображение анкеты. Это может быть фотография сверху или скан копия бланка. После своей работы результаты будут сохраняются в CSV-файл.

Рассмотрим алгоритм работы компонента:

1) начало: вход в компонент;
2) загрузка изображения: компонент получает на вход изображение анкеты для анализа;
3) препроцессинг изображения;
    3.1) преобразование в оттенки серого;
    3.2) сглаживание с помощью фильтра Гаусса;
    3.3) применение оператора Canny для обнаружения границ;
4) поиск области анкеты;
    4.1) поиск контуров на изображении;
    4.2) определение контура анкеты;
5) выделение области анкеты;
    5.1) применение преобразования перспективы для выделения области анкеты;
    5.2) получение и сохранение выделенной области;
6) обработка анкеты;

6.1) применение пороговой обработки к выделенной области для получения четкого изображения;

6.2) обнаружение контуров на обработанном изображении;

6.3) определение контуров блоков с ответами на анкете;

7) анализ ответов;

7.1) идентификация блоков с ответами на анкете;

7.2) определение заполненных ответов в каждом блоке;

7.3) оценка правильности ответов согласно ключу ответов;

7.4) формирование списка результатов анализа;

8) извлечение возраста;

8.1) определение области, содержащей возраст на анкете;

8.2) применение преобразования перспективы для выделения области с возрастом;

8.3) применение пороговой обработки и распознавание текста для извлечения возраста;

9) запись результатов: сохранение результатов анализа и возраста в файл;

10) конец: завершенеи работы компонента.

Алгоритм работы компонента представлен на рисунке 8.

Данный модуль в принципе является самостоятельным. Он может использоваться в различных областях, где требуется автоматизированный анализ анкет. Также при желании, его можно расширить и настроить для анализа других типов анкет.

Также не стоит забывать об обработке ошибок. При реализации данного компонента необходимо предусмотреть решение ошибочных ситуаций (неправильное распознавание, неверный формат изображения), которые могут возникать в процессе работы, чтобы обеспечить корректную работу.

2.1.4 Поиск модели автомобиля

Разработку компонента стоит начать с определения основной функции: обеспечение автоматизированного доступа к информации о моделях автомобилей. Информацию о модели можно взять с различных сайтов, таких как Nomerogram.ru, Avtocod.ru и другие. Важно учесть, что работа с сайтом будет вестись программно, а значит стоит выбрать сайт не только с самой большой базой

Рисунок 8 – Алгоритм работы анализатора

автомобилей, но и тот, у которого нет «проверки на робота». Из представленных вариантов, а именно Nomerogram.ru, Avtocod.ru, Autoteka.ru, Avtoproverka.ru, имеющих примерно одинаковую базу автомобилей, все кроме Autoteka.ru имеют так называемую «проверку на робота» (рисунок 9).

Рисунок 9 – Проверка на робота

Следовательно, работа будет происходить именно с Autoteka.ru (рисунок 10).

Autoteka.ru – это веб-сервис, который собирает и предоставляет информацию о различных транспортных средствах на территории России. Он пользует-

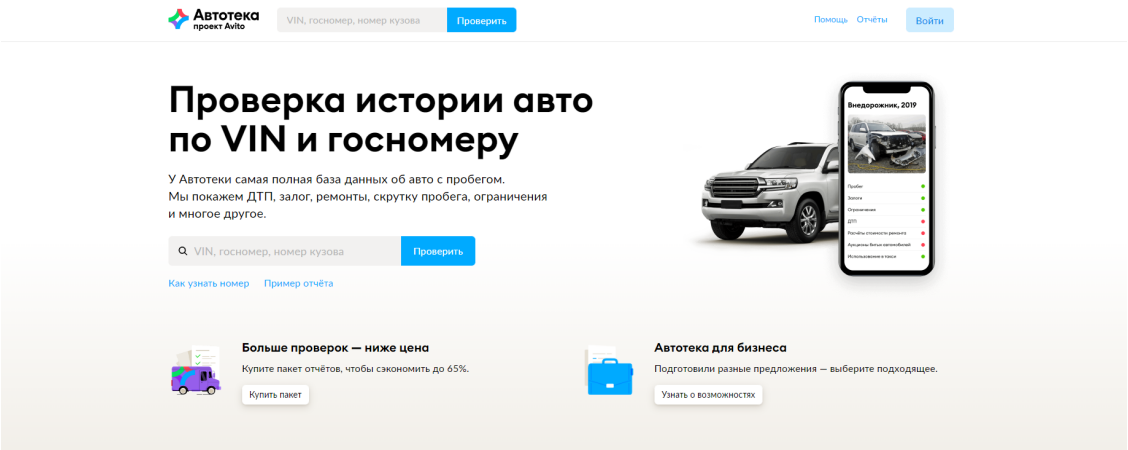

Рисунок 10 – Главный экран Autoteka.ru

ся популярностью среди водителей, автодилеров, страховых компаний и других участников автомобильного сектора. Вот основные характеристики и особенности сайта autoteka.ru:

– проверка истории автомобиля. На сайте autoteka.ru пользователи могут проверить историю определенного автомобиля, используя его номер регистрации. Эта функция предоставляет информацию о предыдущих владельцах, результатах технического осмотра, наличии ограничений на регистрацию, участии в ДТП и других значимых данных;

– получение технических данных. Люди могут просматривать информацию о технических характеристиках машины, таких как бренд, модель, год выпуска, тип кузова, мощность двигателя и других деталях. Это может быть полезно при принятии решения о покупке или продаже автомобиля;

– проверка наличия ограничений. Пользователи имеют возможность проверить, есть ли какие-либо ограничения на регистрацию автомобиля, такие как залоги, аресты или запреты на регистрацию по различным причинам. Это важно для тех, кто планирует приобрести автомобиль, и желает удостовериться в его юридической чистоте перед сделкой;

– получение отчетов по автомобилю. Пользователи имеют возможность запросить подробные отчеты об истории автомобиля, включая информацию о предыдущих сделках, количестве владельцев, технических характеристиках, уча-

стии в ДТП и прочих данных. Это способствует осознанному выбору при покупке подержанного транспортного средства;

– онлайн-консультации. Кроме того, на avtoteka.ru доступны онлайн-консультации и помощь пользователям по вопросам автомобильной истории, юридическим аспектам регистрации транспортных средств и другим темам.

Avtoteka.ru – это востребованный ресурс среди пользователей, источник полезной информации о транспортных средствах. С его помощью можно узнать всё, что важно о машине, просто введя VIN-код, госномер или номер кузова. Процесс простой: вводите данные и жмёте ”Поиск после чего получаете страницу с информацией о транспортном средстве (рисунок 11).

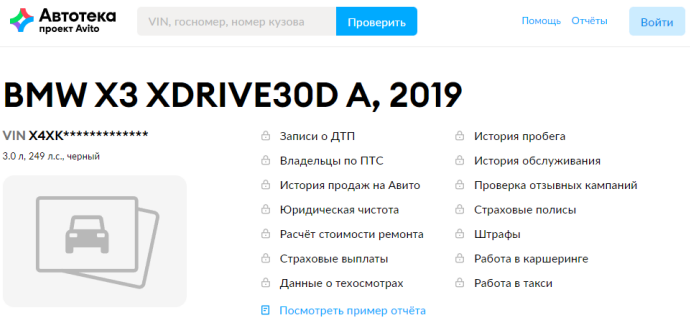

Рисунок 11 – Информация об автомобиле

Полная информация предоставляется по платной подписке или за разовый платеж, но для выполнения поставленной задачи необходимо получить только модель, что можно сделать бесплатно.

Функциональные требования: компонент должен обеспечивать ввод номера, отправку запроса на сайт Autoteka, получение результатов и извлечение необходимой информации.

Выбор инструментов и технологий:

– для автоматизации взаимодействия с веб-страницами можно использовать библиотеку Selenium для Python, так как она обладает мощными возможностями веб-автоматизации;

– для парсинга полученных данных удобно применять инструменты для работы с текстом и регулярные выражения, например, встроенные функции

Python.

Составление алгоритма работы:

– компонент должен открывать веб-браузер, переходить на сайт Autoteka и вводить указанный регистрационный номер;

– после получения результата, необходимо обработать страницу с результатами и извлечь нужные данные, такие как модель автомобиля;

– важно учесть возможные сценарии ошибок и способы их обработки, например, отсутствие данных по указанному номеру или проблемы с доступом к сайту Autoteka.

Алгоритм работы представлен на рисунке 12.

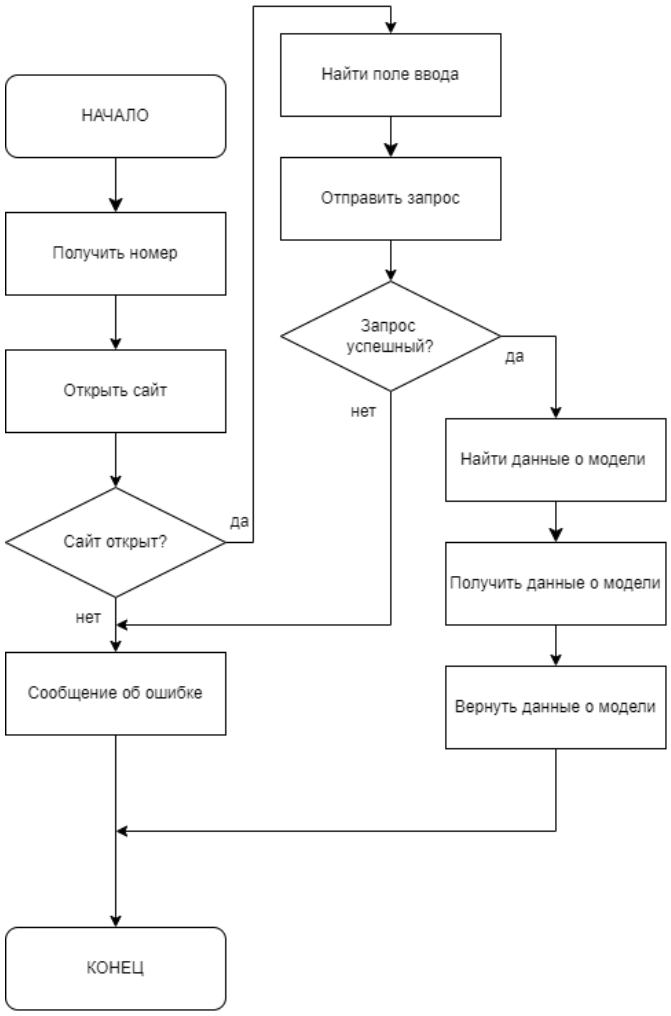

Рисунок 12 – Алгоритм работы компонента для поиска модели

2.1.5 Распознование номера автомобиля

В этом разделе будет описан компонент, который используется для распознавания номерных знаков автомобиля, его алгоритм и функции. Рвссматриваются модели, которые он использует в процессе своей работы.

2.1.5.1 Обнаружение положения номерного знака

Для обнаружения положения номерного знака используется предобученная модель model-resnet.tflite с архитектурой ResNet-50. Она предназначена для выявления границ номерного знака и выделение его области. ResNet-50 выбрана благодаря ее способности эффективно извлекать признаки из изображений и обучаться на больших объемах данных [6].

ResNet50 состоит из четырех ключевых модулей: сверточные слои, блок идентификации, сверточный блок и полностью связанные слои. Сверточные слои занимаются извлечением признаков из изображений, таких как края, текстуры и формы. Блок идентификации и сверточный блок обрабатывают и преобразовывают эти признаки. В завершение, полностью связанные слои отвечают за финальную классификацию [6].

Сверточные слои ResNet50 состоят из нескольких сверточных блоков, сопровождаемых пакетной нормализацией и активацией. Они извлекают особенности из входных изображений и передают их дальше для анализа. Далее следуют слои объединения, которые уменьшают размеры карт признаков, сохраняя важные детали [6].

Блок идентификации и сверточный блок - это основные строительные блоки ResNet50. Он передает входные данные через ряд сверточных слоев и добавляет к ним остаточные данные. Это позволяет сети изучать функции, оставшиеся после извлечения признаков. Сверточный блок аналогичен блоку идентификации, но использует сверточные слои 1x1 для уменьшения размерности перед основным сверточным слоем 3x3 [6].

Полностью связанные слои выполняют окончательную классификацию. Выходные данные последнего полностью связанного слоя поступают на функцию активации softmax для получения вероятностей классов.

Графическое излбражение структуры представлено на рисунке 13.

50-уровневая архитектура ResNet включает в себя следующие элементы:

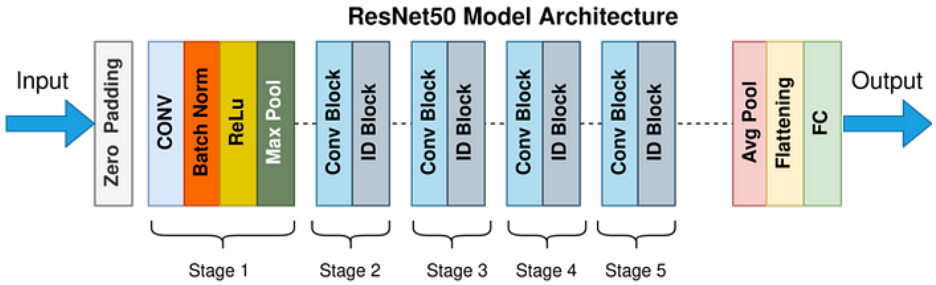

Рисунок 13 – Архитектура ResNet-50

– свертка ядра 7 × 7 наряду с 64 другими ядрами с шагом в 2 размера;

– максимальный уровень объединения с шагом в 2 размера;

– 9 слоев- свертка ядра размером 3 × 3,64, другой с 1 × 1,64 ядрами и третий с 1 × 1256 ядрами. Эти 3 слоя повторяются 3 раза;

– 12 слоев с 1 × 1128 ядрами, 3 × 3128 ядрами и 1 × 1512 ядрами, повторяется 4 раза;

– 18 слоев с ядрами 1 × 1256 и 2 ядрами 3 × 3,256 и 1 × 1,1024, повторяется 6 раз;

– 9 слоев с ядрами 1 × 1512, 3 × 3512 и 1 × 12048 повторяются 3 раза;

– создание среднего пула, за которым следует полностью подключенный уровень с 1000 узлами, с использованием функции активации softmax.

Тестирование модели показало достаточно хороший уровень точности. Результаты тестирования представлены на рисунке 14.

2.1.5.2 Определение номера автомобиля

Для определения номера автомобиля используется предобученная модель model-nomer.tflite. Данная модель предназначена для определения текста на номерном знаке. фрагмент изображения с номерным знаком поступает от модели, описанной выше. Модель представляет собой комбинацю сверточной нейронной сети (CNN), рекуррентной нейронной сети долгой краткосрочной памяти (LSTM) и функции свертки по времени (CTC) для распознавания текста на изображениях. CNN используется для извлечения признаков из изображения номерного знака, LSTM преобразует эти признаки в последовательность символов, а CTC используется для выравнивания последовательности символов с фактическим текстом

Рисунок 14 – Тестирование модели

номера [7].

Структура модели представлена на рисунке 15.

Результаты тестирования представлены на рисунке 16.

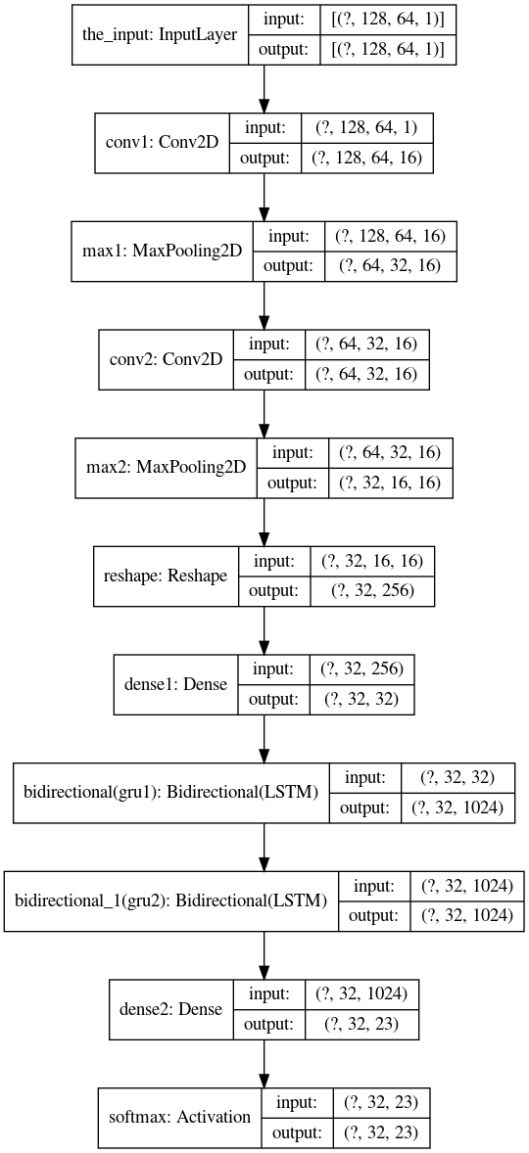

Рисунок 15 – Структура модели

2.1.5.3 Алгоритм компонента

К функциям компонента относятся следующие пункты:

– загрузка и разбиение входного видео на кадры;

– выбор отдельного кадра;

– распознавание номера автомобиля на получившимся изображении.

Алгоритмы работы представлены на рисунках 17- 21.

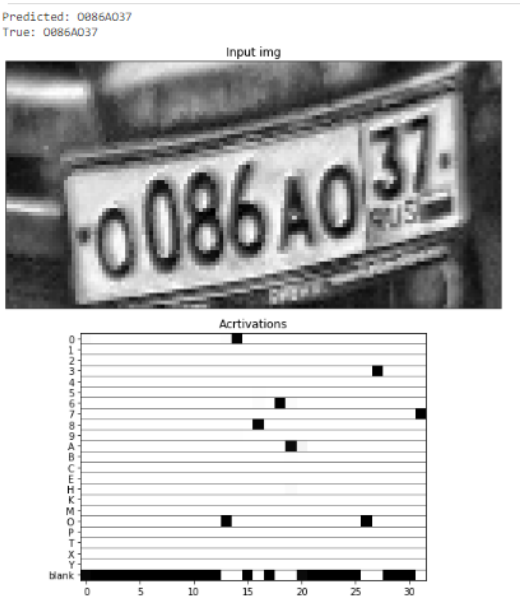

Рисунок 16 – Результаты тестирования

Выводы

В этом разделе было рассмотрено описание модуля для сбора данных, был рассмотрен процесс разработки компонентов, входящих в модуль, а также их функции и особенности.

2.2 Разработка структуры компонента для обучения нейросети и предсказания интересов

В данном разделе будет детально рассмотрена структура модуля, его функции и алгоритмы функционирования. Кроме того, будут рассмотрены наиболее популярные методы предсказания, будет обоснован выбор наиболее

2.2.1 Определение ключевых функций

Разрабатываемый модуль должен предсказывать интересы людей на основе их автомобиля и возраста. Интересы разбиваются на различные группы, например:

– искусство;
– спорт;

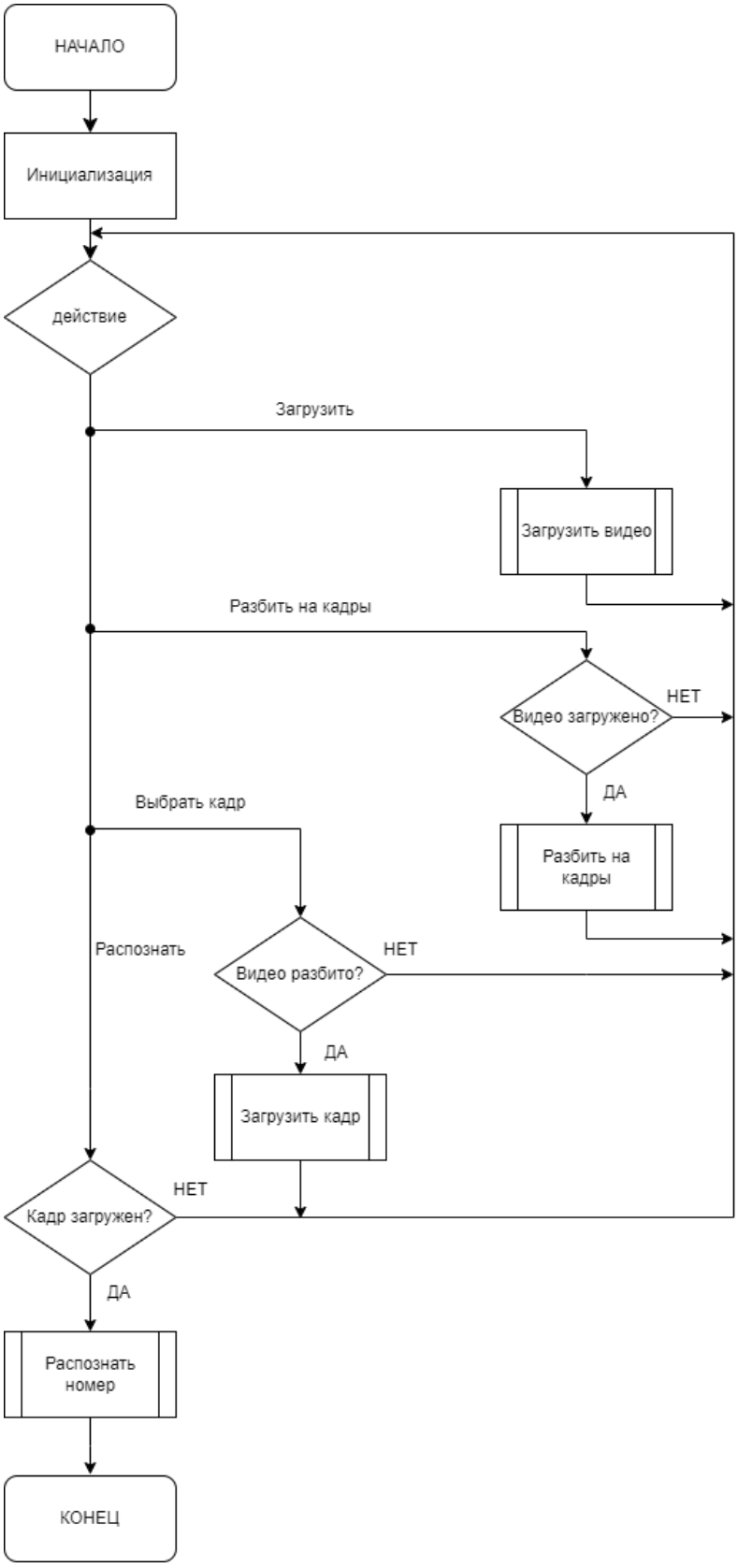

Рисунок 17 – Схема алгоритма компонента распознавания номера

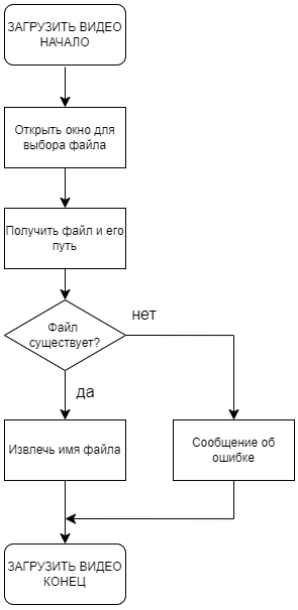

Рисунок 18 – Схема алгоритма загрузки видео

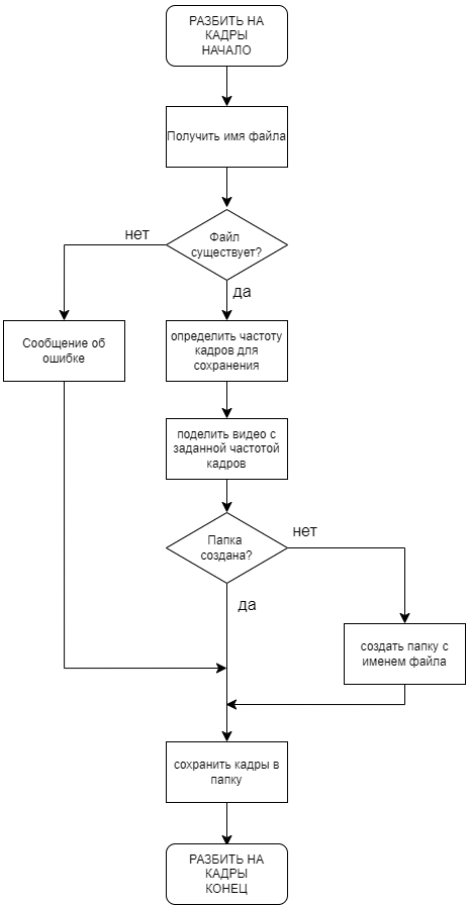

Рисунок 19 – Схема алгоритма разбиения видео на кадры

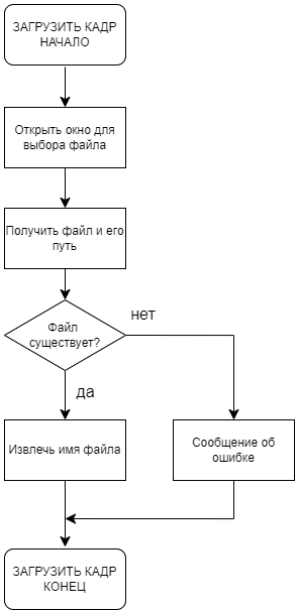

Рисунок 20 – Схема алгоритма выбора одного из кадров

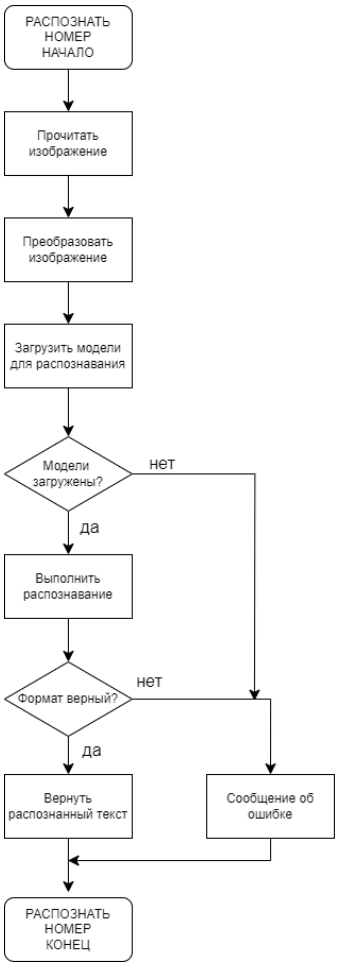

Рисунок 21 – Схема алгоритма распознавания номера

– книги/фильмы;
– наука;
– путешествия;
– кулинария;
– политика.

Модуль должен предсказать вероятность того, есть ли у клиента интересы в этих областях, например, 10% - искусство, 78% - спорт и так далее.

Для определения разделения на области интересов могут использоваться любые вопросы, то есть можно учитывать интересы людей, относящиеся к другим областям, например, музыка, животные и т.п.

При проектировании модуля для примера используется распределение, который было определено при проектировании модуля, направленног на сбор данных. Ниже приведен список вопросов, которые там применялись:

– интересуетесь ли вы искусством (картины, скульптура, музыка и т.д.);
– занимаетесь ли вы спортом или физической активностью;
– любите ли вы чтение книг или просмотр фильмов;
– интересуетесь ли вы наукой и технологиями;
– увлекаетесь ли вы путешествиями и открытием новых мест;
– является ли для вас кулинария или готовка хобби;
– интересуетесь ли вы политикой и общественными вопросами;
– так же на бланке присутствует поле, в котором опрашиваемый должен указать свой возраст.

Точность предсказания должна быть не ниже 80%.

2.2.2 Выбор метода

В этом разделе будут описаны наиболее популярные методы, которые используются для предсказания чего-либо, а также будет обоснован выбор наиболее подходящего.

2.2.2.1 Описание искусственных нейронных сетей

Человеческий мозг великолепно справляется с задачами по распознаванию образов, опережая в этом даже самые передовые компьютеры. Это объясняется особенностями обработки информации в мозге, которые значительно отличаются от методов, используемых в цифровых устройствах. Мозг человека - это сложная,

нелинейная и параллельная система обработки данных. Он умело организовывает свои структурные компоненты для успешного решения разнообразных задач, таких как распознавание образов, превосходя при этом даже самые мощные компьютеры в скорости обработки информации [8].

Искусственная нейронная сеть – это сложный параллельный процессор, который состоит из множества элементарных блоков, называемых нейронами, предназначенных для обработки информации. Эти нейроны накапливают опыт и передают его для дальнейшей обработки [8].

Ниже приведены некоторые преимущества нейронных сетей:

– нелинейность, что позволяет моделировать сложные отношения в данных;

– преобразование входящей информации в выходные результаты;

– адаптивность, позволяющая сети настраиваться под различные задачи;

– понятность ответов, делая выводы из сети более интерпретируемыми;

– учёт контекстной информации для более точных прогнозов;

– способность к отказоустойчивости, что позволяет сохранять работоспособность даже при некоторых повреждениях;

– единообразный подход к анализу и проектированию.

Обучение нейронной сети, или ее настройка, представляет собой процесс изменения свободных параметров сети через моделирование ее взаимодействия с окружающей средой. Тип обучения определяется выбором метода для определения характеристик.

Этот процесс настройки можно описать следующей последовательностью действий:

1) сеть получает стимулы извне;
2) свободные параметры сети подвергаются изменениям;
3) после изменений во внутренней структуре нейронная сеть реагирует на входные стимулы по-новому.

Алгоритм обратного распространения, известный как один из основных методов обучения, находит свое место среди разнообразных алгоритмов обучения. Наиболее известным и широко применяемым из них является алгоритм обратного распространения. Этот метод нацелен на минимизацию разницы между реальны-

ми выходными значениями нейронной сети (ANN) и желаемыми выходами. Его детальное описание можно найти в источниках [9].

Суть обучения через алгоритм обратного распространения заключается в переводе задачи отображения от входных данных к выходным значениям (путем использования набора примеров) в установку оптимальных синаптических весов и порогов для многослойного персептрона. Процесс настройки сети можно рассматривать как выбор наилучшей модели из набора ”кандидатов”структур, основанный на определенных критериях [9].

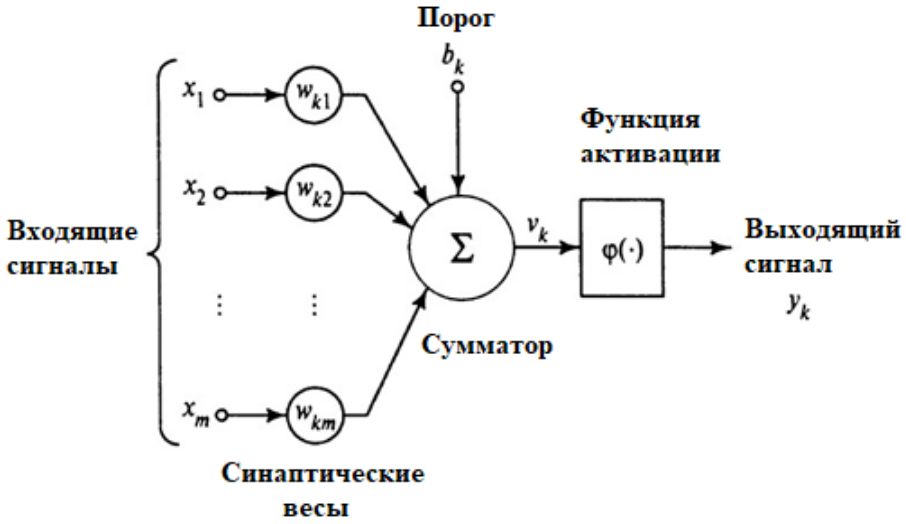

Рисунок 22 – Нелинейная модель нейрона

Нейрон, в контексте нейронных сетей, представляет собой ключевую единицу обработки информации [10]. На рисунке 22 представлена модель нейрона, которая лежит в основе искусственных нейронных сетей. Эта модель включает в себя три основных компонента.

Ансамбль синапсов состоит из множества связей, каждая из которых имеет свой собственный вес или силу. Сумматор суммирует входные сигналы, учитывая их веса, которые связаны с соответствующими синапсами нейрона. Функция активации контролирует выходной сигнал нейрона, ограничивая его амплитуду [10].

Модель нейрона также может включать пороговый элемент, обозначаемый символом b_k, который влияет на функцию активации, регулируя входной сигнал.

Математически, работу нейрона можно описать следующими уравнениями

$$u_k = \sum_{j=1}^{m} w_{kj} * x_j, \tag{1}$$

$$y_k = \phi * (u_k + b_k) \tag{2}$$

где $x_1, x_2, ..., x_m$ - входные сигналы

$w_{k1}, w_{k2}, ..., w_{km}$ - синаптические весы нейрона k

u_k - линейная комбинация входных действий (выход линейного комбинатора)

b_k- порог; $\phi()$ – функция активации

y_k является выходным сигналом нейрона.

Постсинаптический потенциал рассчитывается следующим образом

$$v_k = u_k + b_k \tag{3}$$

Функции активации - это особые инструменты, которые определяют, какой будет выходной сигнал нейрона в ответ на воздействие на него окружающей области. Давайте рассмотрим несколько различных функций активации. Кроме перечисленных, существует еще множество других функций активации [10].

Одной из таких функций является сигмовидная функция. Она примечательна тем, что быстро переходит от линейного к нелинейному поведению. Логистическая функция является примером сигмовидной функции и может быть представлена следующим образом

$$f(x) = \frac{1}{1 + e^{-x}} \tag{4}$$

2.2.2.2 Наивный байесовский классификатор

Наивный байесовский классификатор (Naive Bayes classifier) – это вероятностный алгоритм, основанный на теореме Байеса. Он делает строгое (наивное) предположение о том, что признаки независимы друг от друга для каждого заданного класса. Это предположение значительно упрощает процесс классификации, так как позволяет оценивать одномерные вероятностные плотности вместо мно-

гомерных [11].

В этом контексте одномерная вероятностная плотность означает оценку вероятности каждого отдельного признака при условии их независимости, тогда как многомерная плотность оценивает вероятность комбинации всех признаков, предполагая их взаимозависимость. Классификатор называется наивным именно потому, что такое предположение значительно упрощает вычисления и улучшает эффективность алгоритма. Однако, на практике, предположение о независимости признаков часто не соответствует действительности, что может существенно снизить качество прогнозов в некоторых случаях [11].

Сама же формула Байеса выглядит следующим образом

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)} \tag{5}$$

где $P(A|B)$ — апостериорная вероятность события A при условии выполнения события B

$P(B|A)$ — условная вероятность события B при условии выполнения события A

$P(A)$ и $P(B)$ — априорные вероятности событий A и B соответственно.

А в контексте машинного обучения формула Байеса приобретает следующий вид

$$P(y_k|X) = \frac{P(X|y_k) * P(y_k)}{P(X)} \tag{6}$$

где $P(yk|X)$ — апостериорная вероятность принадлежности образца к классу y_k с учётом его признаков X

$P(X|yk)$ — правдоподобие, то есть вероятность признаков X при заданном классе y_k

$P(yk)$ — априорная вероятность принадлежности случайно выбранного наблюдения к классу y_k

$P(X)$ — априорная вероятность признаков X.

Если объект описывается не одним, а несколькими признаками $X_1, X_2, ..., X_n$, то формула принимает вид

$$P(y_k|X_1, X_2, ..., X_n) = \frac{P(y_k) * \prod_{i=1}^{n} P(X_i|y_k)}{P(X_1, X_2, ..., X_n)} \quad (7)$$

На практике основное внимание уделяется числителю формулы, так как знаменатель зависит только от признаков и не влияет на класс. Поэтому его часто опускают при сравнении вероятностей различных классов. В результате правило классификации сводится к выбору класса с максимальной апостериорной вероятностью.

$$y_k \propto argmax P(y_k) \prod_{i=1}^{n} P(X_i|y_k) \quad (8)$$

Для оценки параметров модели, то есть вероятностей $P(y_k)$ и $P(X_i|y_k)$, обычно применяется метод максимального правдоподобия, который в данном случае основан на частотах встречаемости классов и признаков в обучающей выборке.

2.2.2.3 Логистическая регрессия

Логистическая регрессия – это метод анализа данных, который применяет математические методы для выявления взаимосвязей между двумя переменными. Эти взаимосвязи затем используются для прогнозирования значения одной переменной на основе другой. Обычно результаты прогноза имеют ограниченное количество возможных исходов, например, ”да”или ”нет”[12].

Модель логистической регрессии имеет несколько составляющих:

– уравнения. В математике уравнения выражают зависимость между двумя переменными, например, x и y. Эти уравнения позволяют построить график по осям x и y, подставляя различные значения x и y. Например, если построить график для уравнения $y = 2 * x$, получится прямая линия. Поэтому такие уравнения называют линейными [12];

– переменные. В статистике переменные – это факторы или атрибуты данных, значения которых могут изменяться. В анализе некоторые переменные являются независимыми или объясняющими и служат причиной результата. Дру-

гие переменные зависят от первых и называются зависимыми переменными или переменными отклика. Логистическая регрессия исследует, как независимые переменные влияют на зависимую, анализируя их исторические значения. В нашем примере x является независимой переменной, предиктором или объясняющей переменной, потому что ее значение известно. А y является зависимой переменной, результатом или переменной отклика, так как ее значение необходимо предсказать [12];

– функция логистической переменной. Логистическая регрессия – это статистическая модель, использующая логистическую или логит-функцию в качестве уравнения между x и y. Логит-функция отображает y как сигмоидальную функцию от x [12].

2.2.2.4 Обоснование выбора метода

Ниже приведены достоинства и недостатки рассмотренных методов.

Плюсы логистической регрессии:

– простота реализации и интерпретации;
– эффективность при линейной разделяющей гиперплоскости;
– меньшая склонность к переобучению на небольших наборах данных;
– хорошо работает с линейно разделимыми данными.

Минусы логистической регрессии:

– ограничение на моделирование нелинейных зависимостей;
– ограниченная способность к обработке больших объемов данных;
– не учитывает взаимосвязи между признаками.

Плюсы наивного байесовского классификатора:

– простота и скорость обучения;
– эффективность на небольших наборах данных;
– меньшая склонность к переобучению.

Минусы наивного байесовского классификатора:

– предположение о независимости признаков может быть слишком сильным и нереалистичным;
– не способен улавливать сложные взаимосвязи между признаками;
– требует хорошо подготовленных данных.

Плюсы нейронных сетей:

– способность моделировать сложные нелинейные зависимости;
– автоматическое извлечение признаков из данных;
– гибкость и адаптивность к различным типам данных и задачам;
– эффективность на больших объемах данных;
– улучшенная производительность на сложных задачах.

Минусы нейронных сетей:

– требуют большого объема данных для обучения;
– сложность интерпретации модели;
– большое количество настраиваемых параметров, требующих подбора.

Исходя из приведенного выше анализа, нейронные сети выделяются среди других методов машинного обучения благодаря их способности моделировать сложные зависимости и обрабатывать большие объемы данных.

2.2.3 Структура модуля

Необходимо сформировать наиболее точное описание разрабатываемого программного обеспечения. Для этого было принято решение о рассмотрении функциональной диаграммы верхнего уровня.

В данном случае в качестве отображения взаимосвязей была выбрана нотация IDEF0. В качестве входов датасет для обучения нейросети, возраст посетителя, модель автомобиля посетителя. В качестве субъекта выступает пользователь и вычислительная машина. Управление задается алгоритмами обучения нейронной сети и алгоритмом работы с обученной моделью. К выходным данным будут относиться предсказанные интересы посетителя.

Контекстная диаграмма IDEF0 представлена на рисунке 23

Детализирующая функциональная диаграмма более подробно раскрывает функциональную диаграмму верхнего уровня: описывает взаимодействия и связи процессов, происходящих внутри системы. На ней можно увидеть, какие процессы взаимосвязаны и что между ними общего.

На детализирующей функциональной диаграмме показаны следующие этапы:

– обучение нейросети. На вход поступает датасет, а на выходе получается обученная модель;
– определение интересов. На вход поступает обученная модель, а на вы-

Рисунок 23 – Контекстная диаграмма IDEF0

ход предсказанные интересы.

Детализированная контекстная диаграмма представлена на рисунке 24

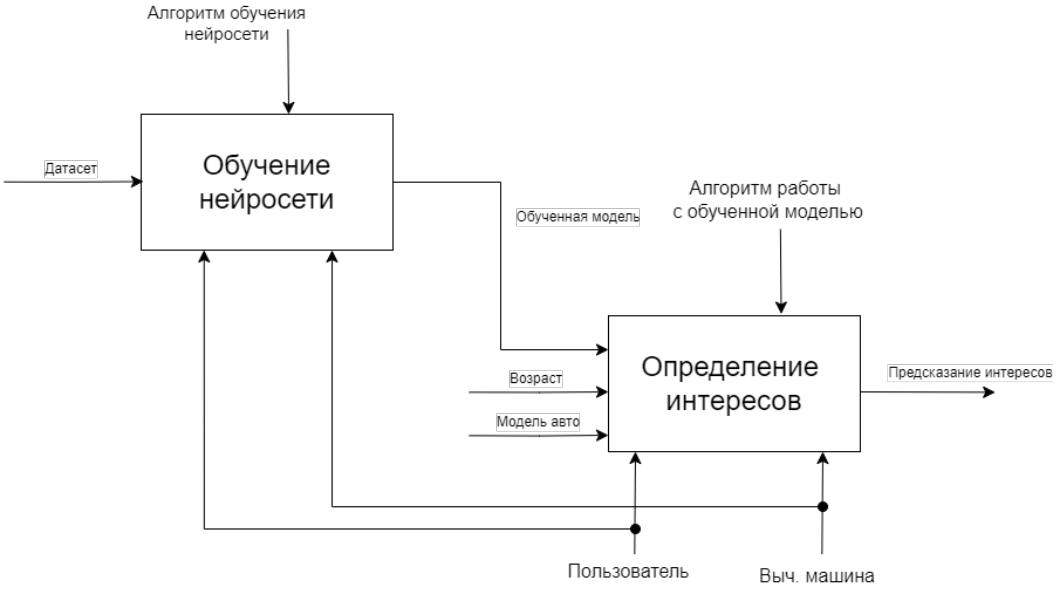

Рисунок 24 – Детализированная контекстная диаграмма IDEF0

Таким образом, разрабатываемый модуль будет состоять из двух основных компонентов: «Обучения нейросети», «Определение интересов».

2.2.4 Алгоритм обучения нейронной сети

В этом разделе рассматривается общий алгоритм обучения нейронных сетей, который включает в себя загрузку, предварительную обработку и разделение данных, определение архитектуры модели, обучение, оценку производительности и настройку гиперпараметров. Понимание этого алгоритма позволит лучше понять процесс создания и настройки нейронных сетей для решения конкретных задач машинного обучения. Сам алгоритм представлен на рисунке 25

Рисунок 25 – Алгоритм обучения сети

Рассмотрим алгоритм подробнее:

1) загрузка данных. Начинаем с загрузки данных из источника, чаще всего из файлов CSV, баз данных или других источников данных;

2) предварительная обработка данных. Перед тем как данные попадут в модель, их необходимо подготовить. Это может включать в себя удаление или заполнение отсутствующих значений, преобразование категориальных переменных в числовые (например, с помощью кодирования One-Hot), нормализацию числовых данных и т.д.;
3) разделение данных. Для оценки модели данные обычно разделяются на обучающий и тестовый наборы. Обучающий набор используется для обучения модели, а тестовый - для оценки ее производительности;
4) определение архитектуры модели. Решается, какая будет архитектура нейронной сети: сколько слоев и скрытых нейронов в каждом слое, какие функции активации использовать, будет ли применяться метод регуляризации (например, Dropout), и т.д.;
5) определение функции потерь и оптимизатор. Выбирается функция потерь (например, кросс-энтропия для классификации) и оптимизатор (например, стохастический градиентный спуск, Adam и т.д.), которые будут использоваться в процессе обучения;
6) обучение модели. Модель обучается на обучающем наборе данных. Обычно это включает в себя несколько эпох обучения, где каждая эпоха представляет собой один проход по всем обучающим данным;
7) оценка производительности модели. После завершения обучения модель оценивается на тестовом наборе данных, чтобы определить ее производительность и обобщающую способность;
8) настройка гиперпараметров. Для улучшения производительности модели может быть выполнен подбор оптимальных гиперпараметров, таких как количество слоев, количество нейронов, скорость обучения и т.д.;
9) выбор лучшей модели. Выбирается модель с лучшей производительностью на основе метрик оценки, таких как точность (accuracy), F1-мера, и т.д.;
10) интеграция с обратной связью. Для улучшения обучения и производительности модели могут быть применены различные методы обратной связи, такие как ранняя остановка (Early Stopping), адаптивная скорость обучения и т.д.

2.2.5 Описаение структуры нейронной сети

Определение структуры нейронной сети включает в себя выбор архитектуры сети, составление слоев и их параметров. Обычно структура нейронной сети определяется в соответствии с характеристиками входных данных, задачей, которую необходимо решить, и требованиями к производительности и точности.

В данном случае, для решения задачи многоклассовой классификации, используется последовательная модель нейронной сети, что означает, что слои нейронов последовательно соединены друг с другом. Вот общая структура нейронной сети:

1) входной слой. Входной слой представляет собой первый слой нейронной сети, который принимает входные данные. В данном случае, размерность входного слоя определяется количеством признаков в данных;
2) скрытые слои. Скрытые слои представляют собой слои нейронов, которые выполняют преобразование входных данных. Каждый скрытый слой состоит из нескольких нейронов (узлов), количество которых определяется архитектурой сети. Для каждого нейрона в скрытом слое применяется активационная функция (например, ReLU) для введения нелинейности и извлечения признаков из входных данных;
3) выходной слой. Выходной слой представляет собой последний слой нейронной сети, который генерирует выходные данные. В данном случае, выходной слой имеет 7 нейронов, по одному для каждого класса (интереса). Каждый нейрон выходного слоя возвращает вероятность принадлежности к соответствующему классу, что достигается использованием активационной функции сигмоида;
4) регуляризация. Не стоит забывать о слоях «отсеивания» Dropout, которые помогают предотвратить переобучение путем случайного «выключения» нейронов во время обучения.

Таким образом, структура нейронной сети определяется количеством слоев, количеством нейронов в каждом слое, выбором активационной функции и применением регуляризации для обеспечения лучшей обобщающей способности модели.

Для получения множества моделей с различной архитектурой использу-

ют специальные библиотеки-тюнеры, которые по заданным гиперпараметрам для каждого варианта обучают получившуюся сеть, после чего можно выбрать архитектуру сети, показавшую наибольшую точность.

Были выделены следующие параметры, которые учитывались при создании структуры нейронной сети:

– num-hidden-layers – число скрытых слоев;
– num-neurons – число нейронов в слоях;
– activation – функция активации;
– dropout-rate – значение слоя Dropout;
– optimizer – определяет оптимизатор;
– learning-rate – скорость обучения.

Разработка нейронной сети производилась на языке Python с использованием библиотеки Keras. Для автоматического подбора параметров использовалась библиотека GridSearchCV.

Лучшие модели имеюют сходные параметры. После проведения эксперементов, было выявлено, что все они имеют по 3 скрытых слоя, одинаковую функцию активации ReLU, оптимизвтор и скорость обучения. Отличными параметрами являются число нейронов, которое у всех моделей разное, и значение слоя отсеивания, которое одинаково и равно 0,2 у 1 и 3 модели, а у 2 модели оно принимает значение 0,1.

В таблице 1 представлен результат моделей.

Таблица 1 – Результат моделей

Метрики \Модели	Модель 1	Модель 2	Модель 3
Параметры	1-20, 2-20, 3-20	1-35, 2-30, 3-20	1-30, 2-20, 3-25
Точность	0,869	0,834	0,823
Ошибка	0,117	0,130	0,174

На рисунке 26 представлена структура лучшей модели.

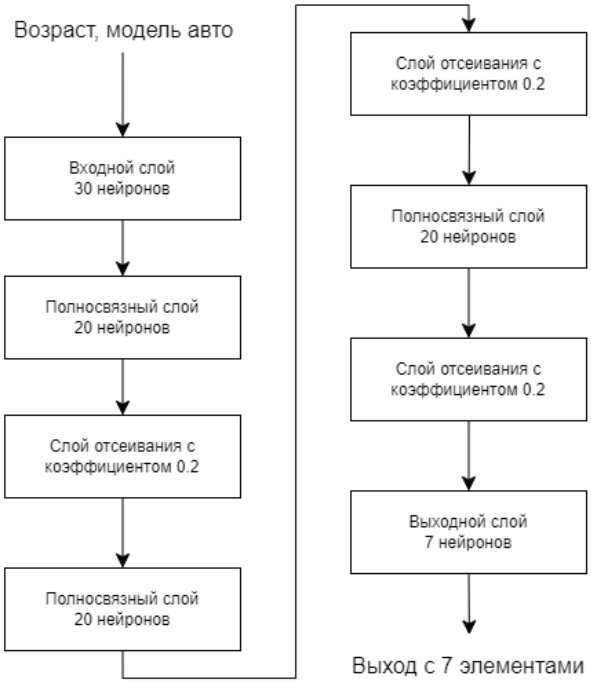

Рисунок 26 – Структура нейронной сети

Из рисунка 26 видно, что полученная модель состоит из:

– входного слоя с 30 нейронами;
– 3-х полносвязных слоев с 20 нейронами;
– 3 слоев Dropout с коэффициентом 0.2;
– выходного слоя с 7 нейронами.

Полносвязные и Dropout слои чередуются.

2.2.6 Разработка модуля

Основной модуль должен получить от пользователя информацию о модели автомобиля и возрасте владельца. На основе этих данных, использую обученную модель, должен предсказать возможные интересы человека. Для считывания модели и возраста на окне необходимо расположить 2 поля для ввода. Результат работы проще воспринимать в таблице, пример которой представлен в таблице2.

Исходя из необходимых компонентов, которые должны быть на экране, можно создать макет экранной формы. Он представлен на рисунке 27.

Рассмотрим алгоритм работы модуля. Схема алгоритма представлена на рисунке 28.

Таблица 2 – Пример форматирования результата

Искусство	Спорт	Книги/ Фильмы	Наука	Путешест-вия	Кулинария	Политика
93.00%	97.30%	68.61%	71.92%	0.34%	3.82%	92.60%

Рисунок 27 – Макет экранной формы

Рассмотрим алгоритм подробнее:

1) инициализация интерфейса. Модуль создает графический интерфейс. Он состоит из окна, содержащего поля для ввода модели и возраста, кнопки для выполнения предсказания и таблицы для отображения результатов;
2) считывание данных. Пользователь вводит данные в поля ”Модель”и ”Возраст”;
3) обработка введенных данных. Данные из полей ввода предварительно обрабатываются таким же образом, как и обрабатывались данные, использующиеся для обучения нейронной сети;
4) получение предсказаний. Введенные данные (модель и возраст) передаются в обученную модель для выполнения предсказания интересов. Результаты предсказаний возвращаются в виде вероятностей принадлежности к каждой из категорий интересов;

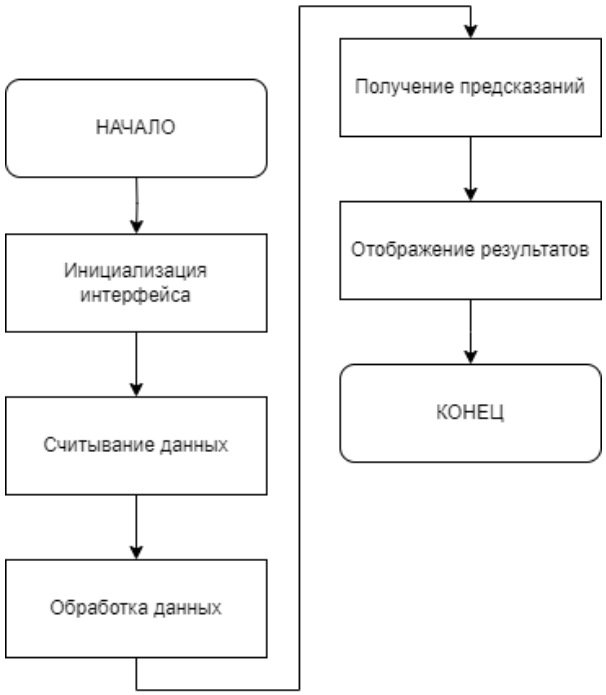

Рисунок 28 – Алгоритм модуля

5) отображение результатов. После получения результатов предсказаний, проценты принадлежности к каждой категории интересов отображаются в таблице. Каждый столбец таблицы соответствует одной из категорий интересов;
6) завершение работы интерфейса. После завершения работы с интерфейсом пользователь может закрыть окно, нажав на кнопку закрытия или продолжить работу;

Выводы

В данном разделе был рассмотрен процесс проектирования разрабатываемого модуля. Будет определен функционал и особенности. Кроме того, были разработаны алгоритмы функционирования, а также структура нейросети и самого модуля в целом.

3 Программная реализация

В данном разделе будут рассмотрены средства, методы и библиотеки, которые использовались при программной реализации. Кроме того, тут будут описаны основные функции, которые реализованы в модулях, и то как они работают. Так же будет рассмотрен процесс работы системы на заданном примере.

Для наглядного описания взаимодействия разработанных компонентов ниже приведена диаграмма вариантов использования (рисунок 29).

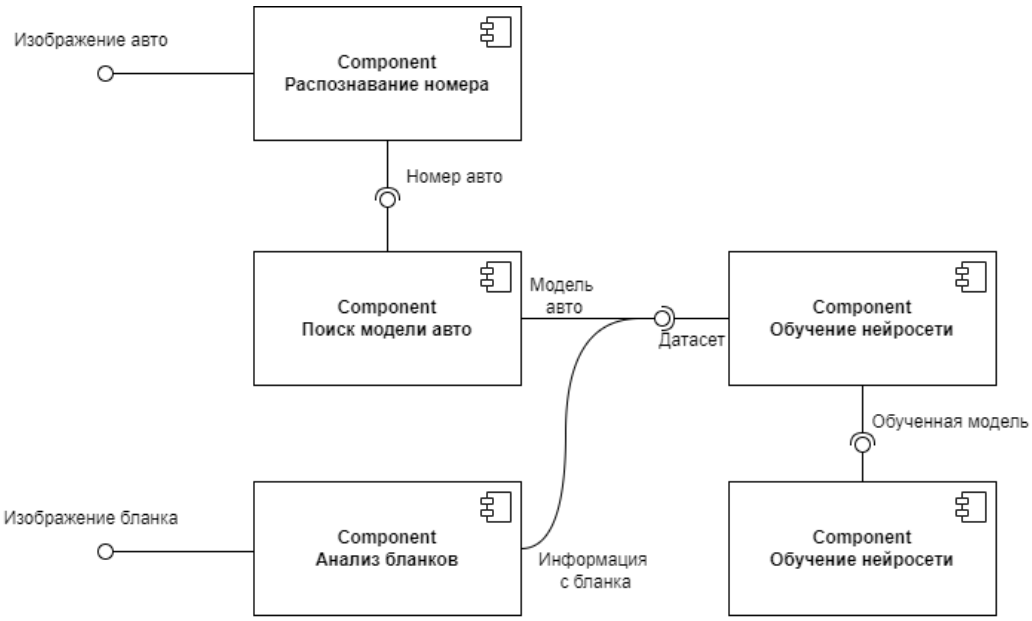

Рисунок 29 – Диаграмма компонентов

3.1 Программная реализация модуля для сбора данных

В данном разделе рассматриваются технологии, используемые при реализации, и описывается сама программная реализация всех компонентов, которые выходят в модуль сбора данных.

3.1.1 Анализ бланков

В коде компонента используются библиотеки «cv2» (OpenCV) для обработки изображений, «numpy» для работы с массивами, «argparse» для обработки аргументов командной строки, «imutils» для удобных функций обработки изображений, «csv» для записи результатов в CSV-файл и «pytesseract» для распознавания текста на изображении. Сам компонент имеет следующие основные функции:

1) process_exam(image_path). Функция принимает путь к изображению листа. Используя библиотеку «cv2», выполняет следующие шаги обработки изображения:
   1.1) преобразует изображение в оттенки серого, размывает его и применяет пороговое преобразование для выделения контуров;
   1.2) находит контуры на изображении и выбирает наибольший контур, который предположительно является контуром экзаменационного листа;
   1.3) используя функцию «four_point_transform» из «imutils», выполняет перспективное преобразование изображения, чтобы получить прямоугольный экзаменационный лист;
   1.4) применяет пороговое преобразование к преобразованному изображению и находит контуры вопросов на экзаменационном листе;
   1.5) для каждого вопроса находит наибольший контур, который предположительно является контуром выбранного ответа;
   1.6) определяет выбранные ответы и возвращает результаты в виде списка;
2) get_age(image_path). Функция принимает путь к изображению документа, содержащего возраст. Используя библиотеку «cv2», выполняет следующие шаги обработки изображения:
   2.1) находит контуры на изображении и выбирает наибольший контур, который предположительно является контуром документа;
   2.2) используя функцию «four_point_transform» из «imutils», выполняет перспективное преобразование изображения, чтобы получить прямоугольный документ;
   2.3) применяет пороговое преобразование к преобразованному изоб-

ражению и находит контуры чисел на документе;

2.4) используя библиотеку «pytesseract», выполняет распознавание текста на изображении и находит числа, представляющие возраст;

2.5) возвращает возраст в виде строки;

3) write_to_csv(filename, results):

3.1) функция принимает путь к CSV-файлу и результаты обработки;

3.2) используя библиотеку «csv», открывает файл в режиме добавления и записывает результаты в виде строки в CSV-файл;

4) blank(path_in, path_out, model):

4.1) функция принимает путь к входному изображению, путь к выходному CSV-файлу и модель (номер экзаменационного листа);

4.2) вызывает функции «process_exam» и «get_age» для обработки изображения и определения возраста;

4.3) добавляет модель и возраст в начало списка результатов;

4.4) вызывает функцию «write_to_csv» для записи результатов в CSV-файл;

5) main():

5.1) основная функция, которая принимает путь к входному изображению и путь к выходному CSV-файлу с помощью аргументов командной строки;

5.2) вызывает функции «process_exam» и «get_age» для обработки изображения и определения возраста;

5.3) вызывает функцию «write_to_csv» для записи результатов в CSV-файл;

3.1.2 Поиск модели авто

При программной реализации данного компонента использовалась библиотеку Selenium для автоматизации веб-браузера. Она включает в себя следующие функции и шаги:

1) импортируются необходимые модули: selenium, time, и другие модули, необходимые для работы с Selenium;

2) указывается путь к драйверу браузера (chromedriver.exe) в переменной

«driver_path»;

3) указывается URL-адрес веб-сайта, на котором будет выполняться поиск, в переменной «website_url»;
4) указывается номер, который будет использоваться в качестве вводного параметра, в переменной «num»;
5) создается экземпляр браузера Chrome с использованием указанного драйвера в переменной «driver»;
6) определяется функция «site», которая принимает в качестве аргумента номер и выполняет следующие действия:
    6.1) загружается веб-страница с указанным URL-адресом;
    6.2) на странице вводится номер в поле с именем ”identifier”;
    6.3) номер отправляется в поле ввода с помощью метода «send_keys» и клавиши Enter;
    6.4) ожидается изменение URL-адреса на веб-странице с использованием «WebDriverWait»;
    6.5) ожидается 3 секунды для загрузки страницы;
    6.6) мна странице находится элемент с классом ”pit4K который содержит текст, связанный с номером;
    6.7) текст из элемента извлекается и разбивается на строки;
    6.8) результирующий текст объединяется в две строки с помощью «join(lines[:2])»;
    6.9) возвращается первая строка текста, разделенная запятой и приводится к нижнему регистру;
    6.10) завершается работа браузера с помощью «driver.quit()»;
7) в блоке «if __name__ == ”__main__”:» вызывается функция «site» с передачей номера в качестве аргумента и выводится результат в консоль.

3.1.3 Распознавание номера

Для работы модуля используются следующие библиотеки и модули:

– PyQt6. Для создания графического интерфейса пользователя и обработки событий;

– OpenCV. для обработки изображений и видео, включая разделение видео на кадры и обнаружение номерных знаков;

– TensorFlow Lite. Для загрузки и использования моделей машинного обучения для распознавания номеров автомобилей;

– NumPy. Для работы с массивами и математических операций;

– Matplotlib. Для визуализации результатов обработки изображений;

– SciPy. Для преобразования изображений и обнаружения линий;

– autoteka и test_blank. Пользовательские модули для работы с сайтом и создания пустых бланков.

Функции компонента:

1) load():
   1.1) функция загружает видеофайл из диалогового окна выбора файлов;
   1.2) использует метод QFileDialog.getOpenFileName() для открытия диалогового окна выбора файлов;
   1.3) путь к выбранному видеофайлу сохраняется в переменной video_file в формате Path(video_file[0]);
2) split():
   2.1) функция разделяет видеофайл на отдельные кадры;
   2.2) создает папку с именем, соответствующим имени исходного видеофайла, если такой папки еще нет;
   2.3) использует библиотеку moviepy для разделения видео на кадры с заданной частотой кадров (SAVING_FRAMES_PER_SECOND);
   2.4) каждый кадр сохраняется в папке с именем, соответствующим времени его появления в видео;
3) choose():
   3.1) функция распознает номер автомобиля на изображении;
   3.2) использует модели машинного обучения для обнаружения номерного знака и его распознавания;
   3.3) производит предварительную обработку изображения, включая изменение размера, преобразование в оттенки серого, применение фильтра Canny для обнаружения границ и преобразование изображения в бинарное;
   3.4) использует модель TensorFlow Lite для распознавания символов

на номерном знаке;

3.5) возвращает распознанный номер автомобиля в виде строки;

4) recognize():

4.1) функция вызывается при нажатии на кнопку ”Распознать”;

4.2) вызывает функцию ‘carplate_text()‘ для распознавания номера автомобиля на выбранном изображении;

4.3) отображает результат распознавания на метке;

4.4) обновляет статистику по количеству распознанных номеров и их вероятности;

4.5) проверяет формат распознанного номера и выводит сообщение об ошибке, если формат неверный;

4.6) если формат верный, сохраняет результат в массив ‘arr‘ и обновляет статистику на метках;

4.7) передает полученный номер в компонент определения модели машины;

3.2 Программная реализация модуля для обучения нейросети и предсказания интересов

В данном разделе будет описана программная реализация всех компонентов. Модуль разрабатывался на языке Python в среде разработки Visual Studio Code.

3.2.1 Обучение нейросети

Этот скрипт реализует процесс обучения нейронной сети для прогнозирования интересов на основе их возраста и модели авто. При его создании использовались следующие средства:

– библиотека pandas. Используется для работы с данными в формате DataFrame, включая загрузку данных из CSV-файла и их предварительную обработку;

– библиотека numpy. Применяется для работы с числовыми данными и выполнения математических операций, таких как нормализация числовых данных;

– библиотека scikit-learn. Используется для разделения данных на обучающий и тестовый наборы, преобразования категориальных переменных в числовые с помощью OneHotEncoding, подбора оптимальных параметров модели с помощью GridSearchCV и других методов машинного обучения;

– библиотека TensorFlow. Применяется для создания и обучения нейронных сетей. В данном скрипте используется Keras API для построения и компиляции модели нейронной сети.

Теперь более подробно рассмотрим процесс реализации:

1) загрузка данных. Из CSV-файла ’datanew.csv’ данные загружаются в объект DataFrame библиотеки pandas. Входными параметрами являются поля Model и Age, а поля Sport, book/films, Science, Travel, Cooking, Politics выходными.
2) предобработка данных. Категориальный столбец ”Model”преобразуется в числовые значения с помощью метода OneHotEncoding из библиотеки sklearn. Нормализация числовых данных о возрасте также выполняется для лучшей работы нейронной сети;
3) разделение данных. Данные разделяются на обучающий и тестовый наборы с использованием функции train_test_split из библиотеки sklearn;
4) определение функции создания модели. Функция create_model определяет архитектуру нейронной сети с возможностью настройки различных параметров, таких как количество слоев, количество нейронов, функция активации, dropout rate, оптимизатор и т.д.;
5) создание модели KerasClassifier. Создается модель KerasClassifier, обертка над моделью Keras, для использования с методом кросс-валидации GridSearchCV из библиотеки sklearn;
6) задание сетки параметров для подбора. Задаются параметры для поиска лучших комбинаций параметров модели с помощью GridSearchCV;
7) создание объекта GridSearchCV. Создается объект GridSearchCV с указанными параметрами для поиска оптимальных гиперпараметров модели;
8) определение обратного вызова EarlyStopping. Обратный вызов

EarlyStopping определяется для автоматической остановки обучения, если происходит переобучение модели;

9) подгонка объекта GridSearchCV к данным. Объект GridSearchCV подгоняется к обучающим данным с использованием метода fit, включая обратный вызов EarlyStopping;
10) вывод лучших параметров. Выводятся лучшие параметры модели, найденные с помощью GridSearchCV;
11) оценка модели с лучшими параметрами. Лучшая модель из GridSearchCV оценивается на тестовом наборе данных, и выводится точность предсказания.

3.2.2 Реализация модуля

При реализации модуля были написаны 2 скрипта. Один для выполнения предсказаний, другой реализует интерфейс. Использовались следующие средства:

– PyQt6. PyQt6 является набором библиотек для Python, предоставляющим интерфейс для работы с графическими приложениями на основе Qt, кроссплатформенного фреймворка для разработки приложений с графическим интерфейсом. В модуле с GUI используется PyQt6 для создания оконного приложения и визуализации элементов пользовательского интерфейса, таких как кнопки, поля ввода и таблица;

– TensorFlow. TensorFlow в модуле используется для загрузки обученной модели нейронной сети, выполнения предсказаний и обработки данных;

– NumPy. В модуле NumPy используется для предобработки данных и работы с массивами;

– Pandas. В модуле Pandas используется для загрузки данных из CSV-файла и предварительной обработки данных перед подачей их на вход нейронной сети;

– scikit-learn. В модуле scikit-learn используется для предобработки данных и выполнения OneHotEncoding для категориальных переменных.

Рассмотрим первый скрипт более подробно. Ниже представлен его функционал:

1) загрузка обученной модели. В начале модуля загружается обученная

модель нейронной сети из файла, созданная и сохраненная в этапе обучения;

2) преобразование новых данных. Функция preprocess_new_data принимает новые данные (модель и возраст) и преобразует их в формат, совместимый с обученной моделью. В частности, она преобразует категориальный столбец ”Model”в числовые значения с помощью OneHotEncoding и нормализует числовые данные о возрасте;
3) получение предсказаний. Функция get_predictions принимает преобразованные данные и использует обученную модель для выполнения предсказаний. Она возвращает вероятности принадлежности к каждой из категорий интересов;
4) вывод результатов. Функция print_results выводит результаты предсказаний в консоль для отладки;
5) главная функция main. Главная функция main принимает модель и возраст, вызывает описанные выше функции для выполнения предсказаний и форматирует результаты для дальнейшего использования. Она возвращает массив вероятностей принадлежности к каждой категории интересов;
6) тестирование модули. В блоке if __name__ == ”__main__”: пример вызова функции main с тестовыми данными и вывод результатов предсказаний в консоль для отладки.

Теперь рассмотрим скрипт, который реализует интерфейс:

1) определение графического интерфейса. Создается класс MyWindow, который наследуется от QWidget и представляет собой окно приложения. В методе initUI определяется интерфейс окна, включая надписи, поля ввода, таблицу для отображения результатов и кнопку для выполнения предсказаний;
2) обработка событий. В методе proc происходит обработка события нажатия на кнопку. Данные из полей ввода для модели и возраста извлекаются с помощью метода toPlainText(). Если оба поля не пустые, вызывается функция main(), описанная в предыдущем скрипте, для выполнения предсказаний. Результаты предсказаний отображаются в таблице;

3) запуск приложения. В блоке if __name__ == '__main__': создается экземпляр приложения QApplication, создается окно MyWindow, показывается на экране и запускается выполнение приложения с помощью sys.exit(app.exec()).

3.3 Результат

В ходе выполения проекта была разработана система, которая выполняет поставленную задачу сбора данных и прогнозирования интересов посетителей автостоянки.

Окно для сбора информации представлено на рисунке 30. На экране имеются необходимые кнопки дял выполнения заданых функций.

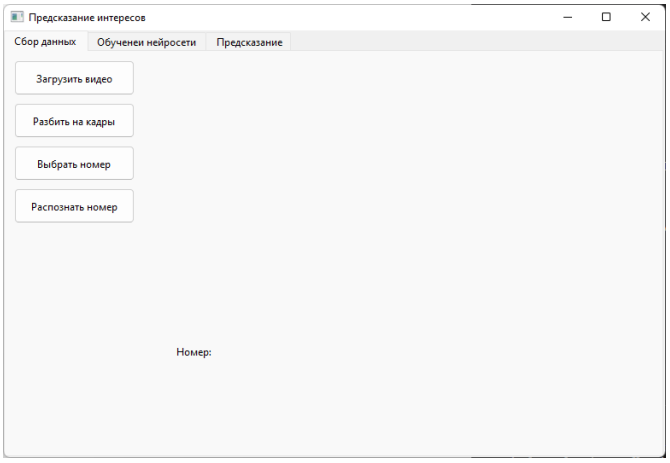

Рисунок 30 – Окно программы

Выбор изображения представлен на рисунке 31. Рядом с кнопками есть место для отрисовкм выбранного изображения.

Распознавание номера представлено на рисунке 32.

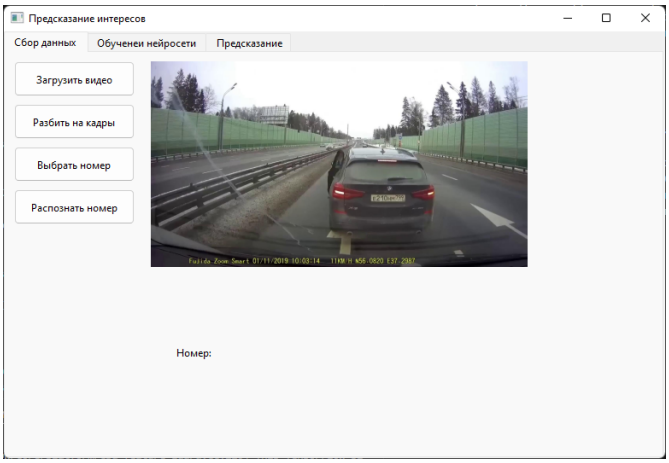

Рисунок 31 – Выбор изображения

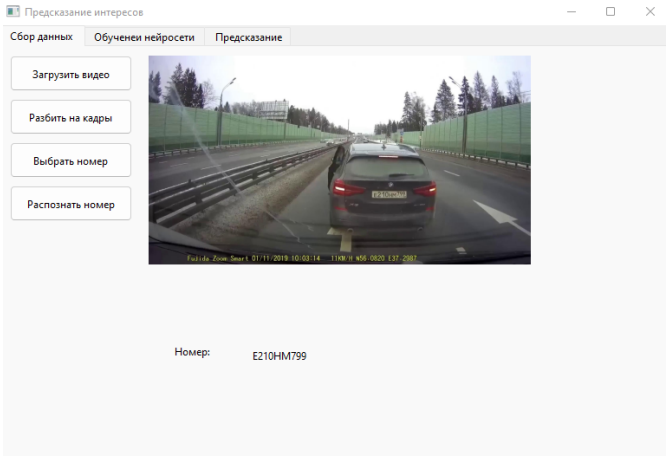

Рисунок 32 – Распознавание номера

Дальше происходит переход на сайт для определения модели автомобиля по распознанному номеру (рисунок 33).

После этого предлагается выбрать изображение с бланком (рисунок 34).

На бланке происходит обводка ответов. красным обводится в случае ответа НЕТ, а зеленым – в случае ответа ДА. Так же происходит считывание возраста, указанного в специальном поле.

Для удобства отладки происходит вывод в консоль (рисунок 35).

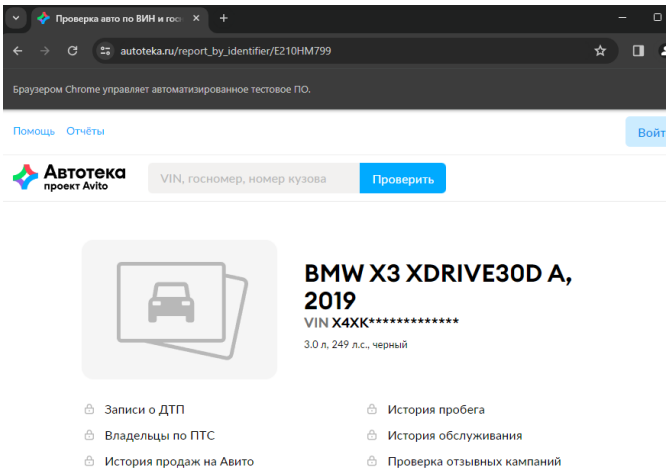

Рисунок 33 – Определение модели авто

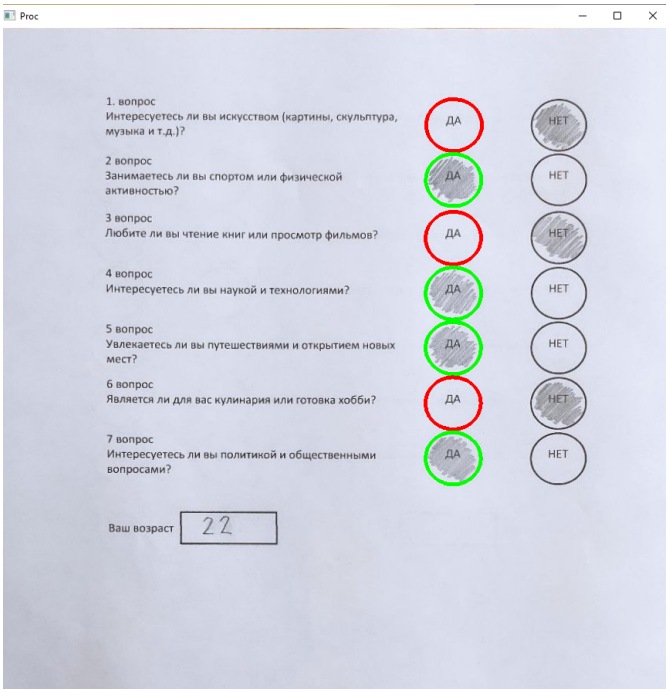

Рисунок 34 – Бланк опроса

Как можно увидеть по рисунку 35, была определена модель автомобиля, так же выведен массив, содержащий ответы на вопросы. «0» означает ответ НЕТ, а «1» - ответ ДА. Кроме того, можно увидеть распознанный возраст «22». В конце все данные объединяются в единое целое, для записи в датасет – файл формата CSV. Результат записи представлен на рисунке 36.

```
BMW X3 XDRIVE30D A
recognize
[0, 1, 0, 1, 1, 0, 1]
22
['BMW X3 XDRIVE30D A', 22, 0, 1, 0, 1, 1, 0, 1]
```

Рисунок 35 – Работа программы

```
1   AUDI A6,35,1,1,0,0,0,0,1
2   BMW X6,28,1,1,0,1,1,1,0
3   BMW X3 XDRIVE30D A,22,0,1,0,1,1,0,1
4
```

Рисунок 36 – Датасет

Окно для обучения нейросети представлено на рисунке 37. Имеются 2 кнопки дял выбора файла и начала обучения.

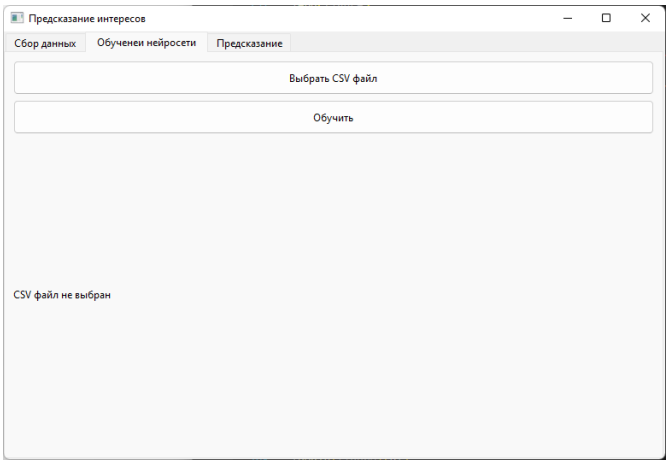

Рисунок 37 – Окно программы

После выбора файла датасета выводится сообщение о выбранном файле (рисунок 38). Дальше можно приступать к обучению. По итогу обучения создается модель с расиширением ”.keras”.

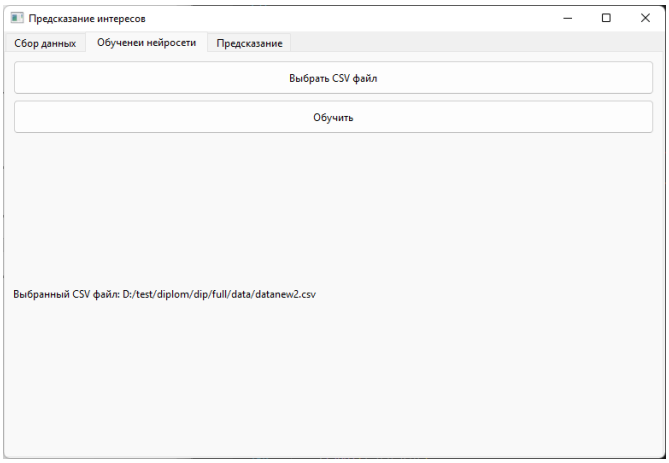

Рисунок 38 – Окно программы

Окно для предсказания интересов придставелно на рисунке 39.

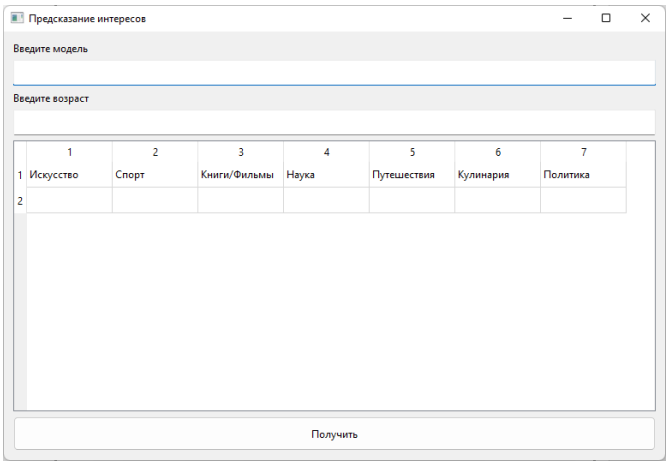

Рисунок 39 – Окно программы

В поля для ввода необходимо ввести данные: название автомобиля и возраст владельца. После этого нажать на кнопку внизу экрана. Дальше модуль проведет работу, используя обученную модель для прогнозирования интересов. Результат работы представлен на рисунке 40.

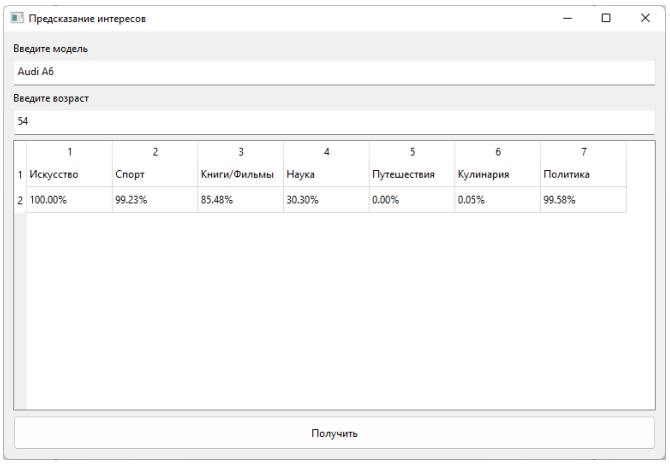

Рисунок 40 – Результат работы модуля

Выводы

В двнном разделе были описаны основные технологии, которые использовались при разработке модулей, из которых состоит система. Было дано пояснение к основным функциям, а также расписан порядок их выполнения.

Заключение

В ходе выполнения проекта был разработан программная система, которая позволяет собирать информацию об интересах посетителей парковки и прогнозировать эти интересы. Для этого было использовано машинное обучение, методы компьютерного зрения и обработки опросных данных.

Основной целью проекта было создать систему, которая поможет маркетологам анализировать интересы клиентов.

В процессе работы был проведен анализ предметной области, выявлены актуальность темы и ключевые требования. Были рассмотрены методы анализа данных, машинного обучениия и принцыпы копьютерного зрения.

Для реализации функционала были разработаны следующие компоненты: анализатор бланков, поиск модели автомобиля и распознавание номера. Кроме того были созданы скрипты для обучения нейронной сети с автоматизированным подбором гиперпараметров, а также для работы с обученной моделью. С помощью библиотеки-тюнера были подобраны наилучшие параметры для обучения и выбора оптимальной структуры нейронной сети.

Анализатор бланков позволяет автоматически обрабатывать изображения анкет, извлекать информацию об ответах на вопросы и возрасте. Для этого используются технологии обработки изображений и распознавания текста.

Компонент поиска модели автомобиля обеспечивает автоматизированный доступ к информации о моделях автомобилей. Для этого используется взаимодействие с сайтом, предоставляющим данные о транспортных средствах.

Компонент распознавания номера автомобиля служит для определения номера на изображении. Для этого используется предобученная модель машинного обучения.

В результате работы была разработана программная система, которая может быть использована в реальных задачах, направленных на анализ интересов посетителей парковок. Разработанная система принесет пользу для маркетологов и других людей, котороые работают с клиентами. Она помогает легко собирать и анализировать информацию о том, что нравится посетителям парковки. Это поз-

воляет более точно нацеливать рекламные компании и предложения, что, в свою очередь, улучшает впечателния клиентов и делает маркетинг более эффективным.

Приложение А

(обязательное)

Авторская справка

Я, Жеребцов Кирилл Андреевич, автор выпускной квалификационной работы «Разработка системы оценки и прогнозирования интересов клиентов автостоянки» сообщаю, что мне известно о персональной ответственности автора за разглашение сведений, подлежащих защите законами РФ о защите объектов интеллектуальной собственности.

Одновременно сообщаю, что:

1. При подготовке к защите выпускной квалификационной работы не использованы источники (документы, отчёты, диссертации, литература и т.п.), имеющие гриф секретности или «Для служебного пользования» ФГБОУ ВО «Вятский государственный университет» или другой организации.

2. Данная работа не связана с незавершёнными исследованиями или уже с завершёнными, но ещё официально не разрешёнными к опубликованию ФГБОУ ВО «Вятский государственный университет» или другими организациями.

3. Данная работа не содержит коммерческую информацию, способную нанести ущерб интеллектуальной собственности ФГБОУ ВО «Вятский государственный университет» или другой организации.

4. Данная работа не является результатом НИР или ОКР, выполняемой по договору с организацией.

5. В предлагаемом к опубликованию тексте нет данных по незащищённым объектам интеллектуальной собственности других авторов.

6. Использование моей дипломной работы в научных исследованиях оформляется в соответствии с законодательством РФ о защите интеллектуальной собственности отдельным договором.

Автор: Жеребцов К. А. «____» _____ 2024 г. _____

подпись

Сведения по авторской справке подтверждаю: «____» _____2024 г.

Заведующий кафедрой ЭВМ: М. Л. Долженкова _____

подпись

Приложение Б

(обязательное)

Листинг кода

```
 # Component for searching car models using the autoteka.ru
website
 from selenium import webdriver
 from selenium.webdriver.common.keys import Keys
 from selenium.webdriver.common.by import By
 from selenium.webdriver.support.ui import WebDriverWait  #
WebDriverWait
 from selenium.webdriver.support import expected_conditions as EC
 import time

 # Path to driver
 driver_path = 'drv/chromedriver-win64/chromedriver.exe'

 # URL link to the page
 website_url = 'https://autoteka.ru/'

 # Initializing the driver
 chrome_service = webdriver.chrome.service.Service(driver_path)
 driver = webdriver.Chrome(service=chrome_service)

 # Function for parsing
 def site(number):
     input = number
     driver.get(website_url)

     input_field = driver.find_element(By.NAME, 'identifier')
     input_field.send_keys(input)
     input_field.send_keys(Keys.ENTER)

     WebDriverWait(driver, 15).until(EC.url_changes(website_url))

     time.sleep(5)

     text_element = driver.find_element(By.CLASS_NAME, 'pit4K')
     text = text_element.text
```

```
        time.sleep(3)

        lines = text.split('\n')
        result = ' '.join(lines[:2])

        driver.quit()

        return result.split(',')[0].strip()

    # Component for analyzing forms
    from imutils.perspective import four_point_transform
    from imutils import contours
    import numpy as np
    import argparse
    import imutils
    import cv2
    import csv
    import re
    import pytesseract

    # Define the correct answers for the exam (placeholder)
    ANSWER_KEY = {0: 0, 1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0}

    # Path to Tesseract executable
    pytesseract.pytesseract.tesseract_cmd = r'C:\\Program Files\\
Tesseract-OCR\\tesseract.exe'

    # Tesseract OCR configuration
    custom_config = r'--oem 3 --psm 6'

    def process_exam(image_path):
        # Read the input image
        image = cv2.imread(image_path)

        # Convert the image to grayscale
        gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

        # Apply GaussianBlur to reduce noise and detail in the image
        blurred = cv2.GaussianBlur(gray, (5, 5), 0)

        # Use Canny edge detector to find edges in the image
```

```
    edged = cv2.Canny(blurred, 75, 200)

    # Find contours in the edged image
    cnts = cv2.findContours(edged.copy(), cv2.RETR_EXTERNAL, cv2.
CHAIN_APPROX_SIMPLE)
    cnts = imutils.grab_contours(cnts)
    docCnt = None

    # Ensure at least one contour was found
    if len(cnts) > 0:
        # Sort the contours by area, keeping only the largest one
        cnts = sorted(cnts, key=cv2.contourArea, reverse=True)
        for c in cnts:
            # Approximate the contour
            peri = cv2.arcLength(c, True)
            approx = cv2.approxPolyDP(c, 0.02 * peri, True)

            # If the approximated contour has four points, assume
 it's the paper
            if len(approx) == 4:
                docCnt = approx
                break

    # Apply a perspective transform to obtain a top-down view of
the paper
    paper = four_point_transform(image, docCnt.reshape(4, 2))
    warped = four_point_transform(gray, docCnt.reshape(4, 2))

    # Apply a binary threshold to the warped image
    thresh = cv2.threshold(warped, 0, 255, cv2.THRESH_BINARY_INV
| cv2.THRESH_OTSU)[1]

    # Find contours in the thresholded image
    cnts = cv2.findContours(thresh.copy(), cv2.RETR_EXTERNAL, cv2
.CHAIN_APPROX_SIMPLE)
    cnts = imutils.grab_contours(cnts)
    questionCnts = []

    # Loop over the contours
    for c in cnts:
        (x, y, w, h) = cv2.boundingRect(c)
```

```
            ar = w / float(h)

            # Filter the contours to find those that correspond to
question bubbles
            if (w >= 50) and (h >= 50) and (ar >= 0.9) and (ar <=
1.1):
                questionCnts.append(c)

        # Sort the question contours top-to-bottom
        questionCnts = contours.sort_contours(questionCnts, method="
top-to-bottom")[0]
        results = []

        # Loop over the question groups
        for (q, i) in enumerate(np.arange(0, len(questionCnts), 2)):
            cnts = contours.sort_contours(questionCnts[i:i + 2])[0]
            bubbled = None
            question_result = 0

            # Loop over the sorted contours
            for (j, c) in enumerate(cnts):
                mask = np.zeros(thresh.shape, dtype="uint8")
                cv2.drawContours(mask, [c], -1, 255, -1)

                mask = cv2.bitwise_and(thresh, thresh, mask=mask)
                total = cv2.countNonZero(mask)
                if bubbled is None or total > bubbled[0]:
                    bubbled = (total, j)

            color = (0, 0, 255)
            k = ANSWER_KEY[q]

            if k == bubbled[1]:
                question_result = 1
                color = (0, 255, 0)

            cv2.drawContours(paper, [cnts[k]], -1, color, 3)
            results.append(question_result)

        print(results)
```

```
        cv2.imshow("Proc", paper)
        cv2.waitKey(0)

        return results

    def get_age(image_path):
        # Read the input image
        image = cv2.imread(image_path)

        # Convert the image to grayscale
        gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

        # Apply GaussianBlur to reduce noise and detail in the image
        blurred = cv2.GaussianBlur(gray, (5, 5), 0)

        # Use Canny edge detector to find edges in the image
        edged = cv2.Canny(blurred, 75, 200)

        # Find contours in the edged image
        cnts = cv2.findContours(edged.copy(), cv2.RETR_EXTERNAL, cv2.
CHAIN_APPROX_SIMPLE)
        cnts = imutils.grab_contours(cnts)
        docCnt = None

        # Ensure at least one contour was found
        if len(cnts) > 0:
            # Sort the contours by area, keeping only the largest one
            cnts = sorted(cnts, key=cv2.contourArea, reverse=True)
            for c in cnts:
                # Approximate the contour
                peri = cv2.arcLength(c, True)
                approx = cv2.approxPolyDP(c, 0.02 * peri, True)

                # If the approximated contour has four points, assume
    it's the paper
                if len(approx) == 4:
                    docCnt = approx
                    break

        # Apply a perspective transform to obtain a top-down view of
the paper
```

```
        paper = four_point_transform(image, docCnt.reshape(4, 2))

        # Further process the transformed image to find the age
        gray = cv2.cvtColor(paper, cv2.COLOR_BGR2GRAY)
        blurred = cv2.GaussianBlur(gray, (5, 5), 0)
        edged = cv2.Canny(blurred, 75, 200)

        cnts = cv2.findContours(edged.copy(), cv2.RETR_EXTERNAL, cv2.
CHAIN_APPROX_SIMPLE)
        cnts = imutils.grab_contours(cnts)
        docCnt = None

        if len(cnts) > 0:
            cnts = sorted(cnts, key=cv2.contourArea, reverse=True)
            for c in cnts:
                peri = cv2.arcLength(c, True)
                approx = cv2.approxPolyDP(c, 0.02 * peri, True)
                if len(approx) == 4:
                    docCnt = approx
                    break

        paper = four_point_transform(paper, docCnt.reshape(4, 2))

        gray = cv2.cvtColor(paper, cv2.COLOR_BGR2GRAY)
        thresh = cv2.threshold(gray, 0, 255, cv2.THRESH_BINARY_INV |
cv2.THRESH_OTSU)[1]

        # Use OCR to extract text from the thresholded image
        text = pytesseract.image_to_string(thresh, config=
custom_config)

        # Find all two-digit numbers in the extracted text
        number = re.findall(r'\b\d+\b', text)
        two_digit_numbers = ' '.join(num for num in number if len(num
) == 2)
        print(two_digit_numbers)
        return two_digit_numbers

    def write_to_csv(filename, results):
        # Append the results to a CSV file
        with open(filename, mode='a', newline='') as file:
```

```
            writer = csv.writer(file)
            writer.writerow(results)

    def blank(path_in, path_out, model):
        print(path_in)
        print(path_out)

        # Process the exam and get the age
        results = process_exam(path_in)
        age = get_age(path_in)

        # Insert model and age to the results
        results.insert(0, int(age))
        results.insert(0, model)

        print(results)
        write_to_csv(path_out, results)
        print("done")

    def main():
        # Set up argument parser
        ap = argparse.ArgumentParser()
        ap.add_argument("-i", "--image", required=True, help="path to
    the input image")
        ap.add_argument("-o", "--output", required=True, help="path
    to save the output CSV file")
        args = vars(ap.parse_args())

        # Process the exam and get the age
        results = process_exam(args["image"])
        age = get_age(args["image"])

        # Insert age to the results
        results.insert(0, int(age))
        print(results)

        # Write results to the CSV file
        write_to_csv(args["output"], results)

    if __name__ == "__main__":
        main()
```

```
 # Component for working with a trained model
 import numpy as np
 import pandas as pd
 import tensorflow as tf
 from sklearn.preprocessing import OneHotEncoder
 import os

 # Transforming new data
 def preprocess_new_data(new_data, full_data):
     # Convert categorical column "Model" to numeric values using
OneHotEncoding
     encoder = OneHotEncoder()
     encoder.fit(full_data[['Model']])
     model_encoded = encoder.transform(new_data[['Model']]).
toarray()

     # Normalization of numerical data (age)
     age = new_data[['Age']].values
     age_normalized = (age - np.mean(full_data['Age'])) / np.std(
full_data['Age'])

     # Merging the transformed data
     X_new = np.concatenate([age_normalized, model_encoded], axis
=1)

     return X_new

 # Receiving Predictions
 def get_predictions(X_new, model_p):
     predictions = model_p.predict(X_new)
     # binary_predictions = np.round(predictions)
     return predictions

 # Output results
 def print_results(binary_predictions):
     # print("New data:")
     # print(new_data)
     print("\nPredictions:")
     np.set_printoptions(threshold=np.inf)
     print(binary_predictions)
```

```
 def main(model, age, model_pred):

     # Example of new data for testing
     new_data = pd.DataFrame({'Model': [model], 'Age': [int(age)
]})

     # Transforming new data
     X_new = preprocess_new_data(new_data, full_data)

     if not os.path.isfile(model_pred):
         raise FileNotFoundError(f  " {model_pred}      ")

     # Check file extension
     if not model_pred.endswith('.keras'):
         raise ValueError      ("        .                  .keras")

     model_p = tf.keras.models.load_model(model_pred)

     # Getting predictions for just one example
     binary_predictions = get_predictions(X_new, model_p)

     art_prediction = binary_predictions[0][0]
     sport_prediction = binary_predictions[0][1]
     book_films_prediction = binary_predictions[0][2]
     science_prediction = binary_predictions[0][3]
     travel_prediction = binary_predictions[0][4]
     cooking_prediction = binary_predictions[0][5]
     politics_prediction = binary_predictions[0][6]

     # Writing to an array
     predictions_array = [art_prediction, sport_prediction,
book_films_prediction, science_prediction, travel_prediction,
cooking_prediction, politics_prediction]

     return predictions_array

 # Component for training neural networks
 import pandas as pd
 import numpy as np
 import tensorflow as tf
 from tensorflow import keras
```

```
 from sklearn.model_selection import train_test_split,
GridSearchCV
 from sklearn.preprocessing import OneHotEncoder
 from scikeras.wrappers import KerasClassifier
 from tensorflow.python.keras.callbacks import EarlyStopping,
ModelCheckpoint
 from tqdm.keras import TqdmCallback

 # Load the dataset
 data = pd.read_csv('datanew2.csv')

 # Convert the categorical "Model" column to numerical values
using OneHotEncoder
 encoder = OneHotEncoder()
 model_encoded = encoder.fit_transform(data[['Model']]).toarray()

 # Normalize the numerical "Age" column
 age = data[['Age']].values
 age_normalized = (age - np.mean(age)) / np.std(age)

 # Combine the normalized age data and the encoded model data
 X = np.concatenate([age_normalized, model_encoded], axis=1)

 # Extract the target variables (interests)
 y = data[['Art', 'Sport', 'Book/Films', 'Science', 'Travel', '
Cooking', 'Politics']].values

 # Split the data into training and test sets
 X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

 # Function to create the model with specified parameters
 def create_model(optimizer='adam', num_hidden_layers=2,
num_neurons=20, activation='relu', dropout_rate=0.2, learning_rate
=0.001):
     # Input layer
     inputs = tf.keras.Input(shape=(X.shape[1],))

     # First hidden layer
     x = tf.keras.layers.Dense(num_neurons, activation=activation)
(inputs)
```

```
    x = tf.keras.layers.Dropout(dropout_rate)(x)

    # Additional hidden layers based on the parameter
num_hidden_layers
    for _ in range(num_hidden_layers - 1):
        x = tf.keras.layers.Dense(num_neurons, activation=
activation)(x)
        x = tf.keras.layers.Dropout(dropout_rate)(x)

    # Output layer with sigmoid activation for multi-label
classification
    outputs = tf.keras.layers.Dense(7, activation='sigmoid')(x)

    # Create the model
    model = tf.keras.Model(inputs=inputs, outputs=outputs)

    # Set the optimizer based on the specified parameter
    if optimizer == 'adam':
        optimizer = tf.keras.optimizers.Adam(learning_rate=
learning_rate)
    elif optimizer == 'rmsprop':
        optimizer = tf.keras.optimizers.RMSprop(learning_rate=
learning_rate)
    elif optimizer == 'sgd':
        optimizer = tf.keras.optimizers.SGD(learning_rate=
learning_rate)

    # Compile the model
    model.compile(optimizer=optimizer, loss='binary_crossentropy
', metrics=['accuracy'])
    return model

# Create a KerasClassifier with the create_model function
model = KerasClassifier(model=create_model, verbose=0, epochs
=100)

# Define the grid of parameters for GridSearchCV
param_grid = {
    'model__num_hidden_layers': [2, 3, 4, 5],
    'model__num_neurons': [20, 30, 40],
    'model__activation': ['relu', 'tanh'],
```

```
    'model__dropout_rate': [0.1, 0.2],
    'model__optimizer': ['adam', 'rmsprop', 'sgd'],
    'model__learning_rate': [0.001, 0.01, 0.1]
}

# Create a GridSearchCV object
grid_search = GridSearchCV(estimator=model, param_grid=param_grid
, cv=3, verbose=1, n_jobs=-1)

# Fit the GridSearchCV object to the training data
grid_search.fit(X_train, y_train)

# Print the best parameters found by GridSearchCV
print("Best: %f using %s" % (grid_search.best_score_, grid_search
.best_params_))

# Get the best parameters and create a model with them
best_params = grid_search.best_params_
best_model = create_model(
    optimizer=best_params['model__optimizer'],
    num_hidden_layers=best_params['model__num_hidden_layers'],
    num_neurons=best_params['model__num_neurons'],
    activation=best_params['model__activation'],
    dropout_rate=best_params['model__dropout_rate'],
    learning_rate=best_params['model__learning_rate']
)

# Define callbacks for early stopping, model checkpointing, and
TQDM progress bar
early_stopping = EarlyStopping(monitor='val_loss', patience=5,
restore_best_weights=True)
model_checkpoint = ModelCheckpoint('model.keras', save_best_only=
True, monitor='val_loss', mode='min')
tqdm_callback = TqdmCallback(verbose=1)

# Train the best model with the callbacks
history = best_model.fit(
    X_train, y_train,
    validation_split=0.2,
    epochs=100,
    callbacks=[early_stopping, model_checkpoint, tqdm_callback],
```

```
    verbose=0
)

# Evaluate the model on the test set
loss, accuracy = best_model.evaluate(X_test, y_test)
print(f'Test Accuracy: {accuracy}')

# Vehicle Number Plate Recognition Component
import os
from datetime import timedelta
from pathlib import Path
import matplotlib.pyplot as plt
import cv2
import numpy as np
import re
import imutils
import pytesseract
import pytesseract as tess
tess.pytesseract.tesseract_cmd = r'Tesseract-OCR\tesseract.exe'
from PyQt6 import uic
from PyQt6.QtGui import QPixmap
from PyQt6.QtWidgets import QApplication, QFileDialog,
QMessageBox
from imutils import contours
from moviepy.editor import VideoFileClip
import json
import sqlite3
import itertools
import tensorflow as tf
from skimage.feature import canny
from skimage.transform import hough_line, hough_line_peaks,
rotate
from skimage.color import rgb2gray

import matplotlib.gridspec as gridspec
import cv2

import autoteka
import test_blank

# Constants for saving frames per second from video
```

```
SAVING_FRAMES_PER_SECOND = 1

# Load the UI design file
Form, Window = uic.loadUiType("Parking.ui")

# Initialize the application and load the UI window
app = QApplication([])
window = Window()
form = Form()
form.setupUi(window)
window.show()

# Global variables to store the paths of video, image, and the
result
video_file = ''
image_file = ''
result = ''
arr = []

# Function to format the time delta for naming frames
def format_timedelta(td):
    result = str(td)
    try:
        result, ms = result.split(".")
    except ValueError:
        return result + ".00".replace(":", "-")

    ms = round(int(ms) / 10000)
    return f"{result}.{ms:02}".replace(":", "-")

# Function to load the video file
def load():
    global video_file
    video_file = QFileDialog.getOpenFileName()
    path = Path(video_file[0])
    video_file = path.name
    print("load")

# Function to split the video into frames
def split():
    video_clip = VideoFileClip(video_file)
```

```
    filename, _ = os.path.splitext(video_file)

    if not os.path.isdir(filename):
        os.mkdir(filename)

    saving_frames_per_second = min(video_clip.fps,
SAVING_FRAMES_PER_SECOND)
    step = 1 / video_clip.fps if saving_frames_per_second == 0
else 1 / saving_frames_per_second

    for current_duration in np.arange(0, video_clip.duration,
step):
        frame_duration_formatted = format_timedelta(timedelta(
seconds=current_duration)).replace(":", "-")
        frame_filename = os.path.join(filename, f"frame{
frame_duration_formatted}.jpg")

        video_clip.save_frame(frame_filename, current_duration)
    print("split")

 # Function to check the format of the recognized license plate
 def check_format(variable):
    pattern = r'^[A-Z]\d{3}[A-Z]{2}\d{3}$'
    pattern2 = r'^[A-Z]\d{3}[A-Z]{2}\d{2}$'
    if re.match(pattern, variable) or re.match(pattern2, variable
):
        return True
    else:
        return False

 # Function to choose an image file for recognition
 def choose():
    global image_file
    image_file = QFileDialog.getOpenFileName()
    image_file = image_file[0]
    form.label_6.setPixmap(QPixmap(image_file))
    form.label_6.setScaledContents(True)
    print("choose")

 # Function to recognize text from the car plate
 def carplate_text():
```

```
    global image_file

    image0 = cv2.imread(image_file)
    image_height, image_width, _ = image0.shape
    image = cv2.resize(image0, (1024, 1024))
    image = image.astype(np.float32)
    paths = 'model/recognize/model_resnet.tflite'
    interpreter = tf.lite.Interpreter(model_path=paths)
    interpreter.allocate_tensors()
    input_details = interpreter.get_input_details()
    output_details = interpreter.get_output_details()
    X_data1 = np.float32(image.reshape(1, 1024, 1024, 3))

    interpreter.set_tensor(input_details[0]['index'], X_data1)
    interpreter.invoke()
    detection = interpreter.get_tensor(output_details[0]['index
'])
    net_out_value2 = interpreter.get_tensor(output_details[1]['
index'])
    net_out_value3 = interpreter.get_tensor(output_details[2]['
index'])
    net_out_value4 = interpreter.get_tensor(output_details[3]['
index'])

    img = image0
    razmer = img.shape

    img2 = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    img3 = img[:, :, :]

    number = 0
    while number < len(detection[0][number]) and detection[0,
number, 0] > 0.9:
        number = number + 1

    box_x = int(detection[0, number, 0] * image_height)
    box_y = int(detection[0, number, 1] * image_width)
    box_width = int(detection[0, number, 2] * image_height)
    box_height = int(detection[0, number, 3] * image_width)

    cv2.rectangle(img2, (box_y, box_x), (box_height, box_width),
```

```
(230, 230, 21), thickness=5)

    net_out_value3

    image = image0[box_x:box_width, box_y:box_height, :]
    img2 = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

    grayscale = rgb2gray(image)
    edges = canny(grayscale, sigma=3.0)
    out, angles, distances = hough_line(edges)
    h, theta, d = out, angles, distances
    angle_step = 0.5 * np.diff(theta).mean()
    d_step = 0.5 * np.diff(d).mean()
    bounds = [np.rad2deg(theta[0] - angle_step),
            np.rad2deg(theta[-1] + angle_step),
            d[-1] + d_step, d[0] - d_step]

    _, angles_peaks, _ = hough_line_peaks(out, angles, distances,
 num_peaks=20)
    angle = np.mean(np.rad2deg(angles_peaks))

    if 0 <= angle <= 90:
        rot_angle = angle - 90
    elif -45 <= angle < 0:
        rot_angle = angle - 90
    elif -90 <= angle < -45:
        rot_angle = 90 + angle
    if abs(rot_angle) > 20:
        rot_angle = 0

    rotated = rotate(image, rot_angle, resize=True) * 255
    rotated = rotated

    rotated1 = rotated[:, :, :]
    if rotated.shape[1] / rotated.shape[0] < 2:
        minus = np.abs(int(np.sin(np.radians(rot_angle)) *
rotated.shape[0]))
        rotated1 = rotated[minus:-minus, :, :]
        print(minus)

    lab = cv2.cvtColor(rotated1, cv2.COLOR_BGR2LAB)
```

```
        l, a, b = cv2.split(lab)
        clahe = cv2.createCLAHE(clipLimit=3.0, tileGridSize=(8, 8))
        cl = clahe.apply(l)
        limg = cv2.merge((cl, a, b))
        final = cv2.cvtColor(limg, cv2.COLOR_LAB2BGR)

        letters = ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9',
'A', 'B', 'C', 'E', 'H', 'K', 'M', 'O', 'P', 'T', 'X',
                'Y']

        def decode_batch(out):
            ret = []
            for j in range(out.shape[0]):
                out_best = list(np.argmax(out[j, 2:], 1))
                out_best = [k for k, g in itertools.groupby(out_best)
]
                outstr = ''
                for c in out_best:
                    if c < len(letters):
                        outstr += letters[c]
                ret.append(outstr)
            return ret

        paths = 'model/recognize/model1_nomer.tflite'
        interpreter = tf.lite.Interpreter(model_path=paths)
        interpreter.allocate_tensors()

        input_details = interpreter.get_input_details()
        output_details = interpreter.get_output_details()
        img = final
        img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        img = cv2.resize(img, (128, 64))
        img = img.astype(np.float32)
        img /= 255

        img1 = img.T
        img1.shape
        X_data1 = np.float32(img1.reshape(1, 128, 64, 1))
        input_index = (interpreter.get_input_details()[0]['index'])
        interpreter.set_tensor(input_details[0]['index'], X_data1)
```

```
    interpreter.invoke()

    net_out_value = interpreter.get_tensor(output_details[0]['
index'])
    pred_texts = decode_batch(net_out_value)
    pred_texts

    fig = plt.figure(figsize=(10, 10))
    outer = gridspec.GridSpec(2, 1, wspace=10, hspace=0.1)
    ax1 = plt.Subplot(fig, outer[0])
    fig.add_subplot(ax1)
    ax2 = plt.Subplot(fig, outer[1])
    fig.add_subplot(ax2)
    return pred_texts[0]

 # Function to recognize the license plate and display information
 def recognize():
    global result
    result = carplate_text().upper()
    print(result)
    if result == '':
        QMessageBox.information(None,    "      ",  "      ")
    elif check_format(result) != True:
        QMessageBox.information(None,    "    ",     ":" +
result)
    else:
        form.label_7.setText(result)

        count = 0
        for i in range(len(arr)):
            if arr[i][0] == result:
                count += 1

        if count == 0:
            arr.append([result, 1])
        else:
            for i in range(len(arr)):
                if arr[i][0] == result:
                    arr[i][1] += 1

        for i in range(len(arr)):
```

```
            if arr[i][0] == result:
                form.label_2.setText(str(arr[i][1]))
                if arr[i][1] < 5:
                    form.label_5.setText('0%')
                elif 5 <= arr[i][1] < 10:
                    form.label_5.setText('5%')
                elif 10 <= arr[i][1] < 20:
                    form.label_5.setText('10%')
                else:
                    form.label_5.setText('15%')

    model = autoteka.site(result)
    print(model)
    print("recognize")

    file_path, _ = QFileDialog.getOpenFileName()
    test_blank.blank(file_path, 'data/data.csv', model)

# Connect UI buttons to their respective functions
form.pushButton.clicked.connect(load)
form.pushButton_2.clicked.connect(split)
form.pushButton_3.clicked.connect(choose)
form.pushButton_4.clicked.connect(recognize)

# Start the application event loop
app.exec()

#Interface
# Importing necessary modules and libraries
from PyQt6.QtWidgets import (
    QApplication,
    QWidget,
    QPushButton,
    QLabel,
    QVBoxLayout,
    QTextEdit,
    QTableWidget,
    QTableWidgetItem,
    QTabWidget,
    QMainWindow,
    QFileDialog,
```

```
    QMessageBox
)
from PyQt6 import QtCore

# Constant for saving frames per second from a video
SAVING_FRAMES_PER_SECOND = 1

# List of interest categories
list_int =     ["",    "",      "/",    "",      "",     "",
   ""]

# Global variables for storing file paths and results
video_file = ''
image_file = ''
result = ''
arr = []

# Main window class inheriting from QMainWindow
class MainWindow(QMainWindow):
    def __init__(self):
        super().__init__()

        self.setWindowTitle      ("      ")
        self.setGeometry(400, 250, 750, 500)

        # Creating a tab widget and setting it as the central
widget
        self.tabs = QTabWidget()
        self.setCentralWidget(self.tabs)

        # Creating tabs
        self.create_tabs()

        self.model_pred = ""

    # Method to create tabs
    def create_tabs(self):
        # Creating the first tab for data collection
        tab1 = QWidget()

        self.pushButton = QPushButton(tab1)
```

```
        self.pushButton.setGeometry(QtCore.QRect(10, 10, 141, 41)
)
        self.pushButton.setObjectName("pushButton")
        self.pushButton.clicked.connect(self.load)
        self.pushButton.setText    ("    ")

        self.pushButton_2 = QPushButton(tab1)
        self.pushButton_2.setGeometry(QtCore.QRect(10, 60, 141,
41))
        self.pushButton_2.setObjectName("pushButton_2")
        self.pushButton_2.clicked.connect(self.split)
        self.pushButton_2.setText    ("      ")

        self.pushButton_3 = QPushButton(tab1)
        self.pushButton_3.setGeometry(QtCore.QRect(10, 110, 141,
41))
        self.pushButton_3.setObjectName("pushButton_3")
        self.pushButton_3.clicked.connect(self.choose)
        self.pushButton_3.setText    ("    ")

        self.pushButton_4 = QPushButton(tab1)
        self.pushButton_4.setGeometry(QtCore.QRect(10, 160, 141,
41))
        self.pushButton_4.setObjectName("pushButton_4")
        self.pushButton_4.clicked.connect(self.recognize)
        self.pushButton_4.setText     ("    ")

        self.label_6 = QLabel(tab1)
        self.label_6.setGeometry(QtCore.QRect(170, 10, 441, 241))
        self.label_6.setText("")
        self.label_6.setAlignment(QtCore.Qt.AlignmentFlag.
AlignCenter)
        self.label_6.setObjectName("label_6")

        self.label_3 = QLabel(tab1)
        self.label_3.setGeometry(QtCore.QRect(200, 340, 71, 21))
        self.label_3.setText   (":")
        self.label_3.setObjectName("label_3")

        self.label_7 = QLabel(tab1)
        self.label_7.setGeometry(QtCore.QRect(290, 340, 161, 31))
```

```
self.label_7.setText("")
self.label_7.setObjectName("label_7")

# Creating the second tab for neural network training
tab2 = QWidget()
tab2_layout = QVBoxLayout()

self.pushButton_csv = QPushButton(tab2)
self.pushButton_csv.setText   (" CSV   ")
self.pushButton_csv.setFixedSize(750, 40)
self.pushButton_csv.clicked.connect(self.choose_csv_file)
tab2_layout.addWidget(self.pushButton_csv)

self.pushButton_learn = QPushButton(tab2)
self.pushButton_learn.setText   ("")
self.pushButton_learn.setFixedSize(750, 40)
self.pushButton_learn.clicked.connect(self.learn)
tab2_layout.addWidget(self.pushButton_learn)

self.label_csv = QLabel(tab2)
self.label_csv.setText("CSV         ")
tab2_layout.addWidget(self.label_csv)

tab2.setLayout(tab2_layout)

# Creating the third tab for prediction
tab3 = QWidget()
tab3_layout = QVBoxLayout()

self.label_model = QLabel   ('    ', tab3)
self.label_age = QLabel   ('     ', tab3)

self.text_edit_model = QTextEdit(tab3)
self.text_edit_model.setFixedSize(750, 30)

self.text_edit_age = QTextEdit(tab3)
self.text_edit_age.setFixedSize(750, 30)

self.table = QTableWidget(tab3)
self.table.setColumnCount(7)
self.table.setRowCount(2)
```

```
            for col in range(7):
                item = QTableWidgetItem(list_int[col])
                self.table.setItem(0, col, item)

            self.button_predict = QPushButton    ('', tab3)
            self.button_predict.setFixedSize(750, 40)
            self.button_predict.clicked.connect(self.proc)

            self.button_choose_file = QPushButton   ('       ', tab3)
            self.button_choose_file.setFixedSize(750, 40)
            self.button_choose_file.clicked.connect(self.
choose_model_file)

            # Adding widgets to the third tab
            tab3_layout.addWidget(self.label_model)
            tab3_layout.addWidget(self.text_edit_model)
            tab3_layout.addWidget(self.label_age)
            tab3_layout.addWidget(self.text_edit_age)
            tab3_layout.addWidget(self.button_choose_file)
            tab3_layout.addWidget(self.table)
            tab3_layout.addWidget(self.button_predict)

            tab3.setLayout(tab3_layout)

            # Adding tabs to the QTabWidget
            self.tabs.addTab(tab1,   "    ")
            self.tabs.addTab(tab2,     "      ")
            self.tabs.addTab(tab3,       "")

        # Placeholder method for learning
        def learn(path):
            pass

        # Method to choose a CSV file
        def choose_csv_file(self):
            file_dialog = QFileDialog()
            options = file_dialog.options()
            file_name, _ = QFileDialog.getOpenFileName(self,     "
CSV   ", "", "CSV Files (*.csv);;All Files (*)", options=options)
            if file_name:
```

```
            self.csv_file = file_name
            self.label_csv.setText(f    " CSV   : {file_name}")
        print(self.csv_file)

    # Method to choose a model file
    def choose_model_file(self):
        file_dialog = QFileDialog()
        options = file_dialog.options()
        file_name, _ = QFileDialog.getOpenFileName(self,     "
     ", "", "Model Files (*.keras);;All Files (*)", options=
options)
        if file_name:
            self.model_pred = file_name
        print(self.model_pred)

    # Method to process the model and age input and get
predictions
    def proc(self):
        model = self.text_edit_model.toPlainText()
        age = self.text_edit_age.toPlainText()
        if (model != "") & (age != "") & (self.model_pred != ""):
            predictions = test_single.main(model, age, self.
model_pred)
            print(predictions)
            for col in range(7):
                predictions[col] = float(predictions[col])
                item = QTableWidgetItem(str("{:2.2f}".format(
predictions[col] * 100)) + "%")
                self.table.setItem(1, col, item)

if __name__ == "__main__":
    app = QApplication(sys.argv)

    window = MainWindow()
    window.show()

    sys.exit(app.exec())
```

Приложение В

(Справочное)

Библиографический список

1. Data Collection [Электронный ресурс] – Режим доступа: https://www.tutorialspoint.com/data-collection, свободный. – Загл. с экрана.

2. Анализ данных [Электронный ресурс] – Режим доступа: https://en.wikipedia.org/wiki/Data_analysis, свободный. – Загл. с экрана.

3. Визуализация данных [Электронный ресурс] – Режим доступа: https://practicum.yandex.ru/blog/vizualizaciya-dannyh/, свободный. – Загл. с экрана.

4. Машинное обучение: методы и способы [Электронный ресурс] – Режим доступа: https://www.osp.ru/cio/2018/05/13054535, свободный. – Загл. с экрана.

5. Медведев П.М. Организация маркетинговой службы с нуля [Текст] ЗАО Издательский дом «Питер». 2005. – 224 с.: ил.

6. Exploring ResNet50: An In-Depth Look at the Model Architecture and Code Implementation [Электронный ресурс] – Режим доступа: https://medium.com/@nitishkundu1993/exploring-resnet50-an-in-depth-look-at-the-model-architecture-and-code-implementation-d8d8fa67e46f, свободный. – Загл. с экрана.

7. CNN-LSTM Architecture and Image Captioning [Электронный ресурс] – Режим доступа: https://medium.com/analytics-vidhya/cnn-lstm-architecture-and-image-captioning-2351fc18e8d7, свободный. – Загл. с экрана.

8. Дюк В.А., Флегонтов А.В., Фомина И.К. Применение технологий интеллектуального анализа данных в естественнонаучных, технических и гуманитарных областях [Текст] Известия Российского государственного педагогического университета им. А.И. Герцена. 2011. No 138.

9. Метод обратного распространения ошибки [Электронный ресурс] – Режим доступа: https://education.yandex.ru/handbook/ml/article/metod-obratnogo-rasprostraneniya-oshibki, свободный. – Загл. с экрана.

10. Neurons in Neural Networks [Электронный ресурс] – Режим доступа: https://www.baeldung.com/cs/neural-networks-neurons, свободный. – Загл. с экрана.

11. Наивный байесовский классификатор [Электронный ресурс] – Режим

доступа: https://ru.wikipedia.org/wiki/Наивный_байесовский_классификатор, свободный. – Загл. с экрана.
12. Пампел Фред, Цвиркун Дмитрий, Груздев Артем. Логистическая регрессия [Текст] ДМК-Пресс, 2023 г. – 218 с.: ил.