МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

Федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

**«Вятский государственный университет»**

**(ФГБОУ ВПО «ВятГУ»)**

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

Допущено к защите

Руководитель проекта

\_\_\_\_\_\_\_\_\_\_\_\_ (Долженкова М. Л.)

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_2024г.

«**Разработка программного модуля для предсказания интересов**

**посетителей парковки»**

Пояснительная записка курсового проекта по дисциплине

«Курс знаний бакалавра»

ТПЖА 090301.387 ПЗ

Разработал студент группы ИВТ-41 \_\_\_\_\_\_\_\_\_\_\_\_/Жеребцов К.А./

Руководитель \_\_\_\_\_\_\_\_\_\_/Долженкова М. Л./

Консультант \_\_\_\_\_\_\_\_\_\_/Долженкова М.Л./

Работа защищена с оценкой «\_\_\_\_\_\_\_\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(оценка) (дата)*

Члены комиссии \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_\_\_\_\_\_\_\_\_\_\_\_/

(подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_\_\_\_\_\_\_\_\_\_\_\_\_/

(подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_\_\_\_\_\_\_\_\_\_\_\_\_/

(подпись)

Киров 2024

**Реферат**

Жеребцов К. А. «Разработка программного модуля для сбора информации об интересах посетителей парковки». ТПЖА. 090301.387 ПЗ: Курс. проект / ВятГУ, каф. ЭВМ; рук. Долженкова М. Л. - Киров, 2024. Графическая часть 4 л. – ф. А2, ПЗ 62 с, 20рис., 7 источников, 5 приложений.

Разработка программного модуля для предсказания интересов посетителей парковки.

Объект исследования и разработки – для предсказания интересов посетителей парковки.

Цель курсового проекта – автоматизация процесса предсказания интересов посетителей парковки.

Результат работы – готовый прототип программного модуля.

# Содержание

[Содержание 3](#_Toc166695504)

[Введение 5](#_Toc166695505)

[1 Анализ предметной области 6](#_Toc166695506)

[1.1 Ключевые аспекты 6](#_Toc166695507)

[1.1.1 Тенденции рынка автомобилей и потребительских предпочтений: 6](#_Toc166695508)

[1.1.2 Методы сбора данных о владельцах и их автомобилях: 7](#_Toc166695509)

[1.1.3 Обработка и анализ данных 9](#_Toc166695510)

[1.1.3.1 Методы предобработки данных 9](#_Toc166695511)

[1.1.3.2 Методы визуализации данных 11](#_Toc166695512)

[1.1.3.3 Методы машинного обучения 12](#_Toc166695513)

[1.1.3.4 Методы статистического анализа 17](#_Toc166695514)

[1.1.4 Персонализация услуг и предложений 17](#_Toc166695515)

[1.2 Актуальность темы 18](#_Toc166695516)

[1.3 Постановка задачи 19](#_Toc166695517)

[1.3.1 Цели и задачи разработки 19](#_Toc166695518)

[1.3.2 Определение ключевых функций 19](#_Toc166695519)

[2 Выбор метода 22](#_Toc166695520)

[2.1 Описание искусственных нейронных сетей 22](#_Toc166695521)

[2.2 Наивный байесовский классификатор 26](#_Toc166695522)

[2.3 Логистическая регрессия 27](#_Toc166695523)

[2.4 Обоснование выбора 30](#_Toc166695524)

[3 Проектирование 31](#_Toc166695525)

[3.1 Структура модуля 31](#_Toc166695526)

[3.2 Алгоритм обучения нейронной сети 34](#_Toc166695527)

[3.3 Описание структуры нейронной сети 36](#_Toc166695528)

[3.4 Разработка модуля 40](#_Toc166695529)

[4 Программная реализация 42](#_Toc166695530)

[4.1 Обучение нейросети 42](#_Toc166695531)

[4.2 Реализация модуля 45](#_Toc166695532)

[4.3 Результат 47](#_Toc166695533)

[5 Оценка качества 48](#_Toc166695534)

[Заключение 51](#_Toc166695535)

[Список литературы 52](#_Toc166695536)

[Приложение А 53](#_Toc166695537)

[Приложение Б 54](#_Toc166695538)

[Приложение В 55](#_Toc166695539)

[Приложение Г 56](#_Toc166695540)

[Приложение Д 57](#_Toc166695541)

Введение

В современном мире обработка данных и использование искусственного интеллекта становятся все более значимыми в различных областях. Особенно в контексте рынка автомобилей и сервисов для владельцев, понимание и прогнозирование интересов владельцев автомобилей может существенно повлиять на улучшение услуг и персонализацию предложений. В данном исследовании представлена разработка и обучение нейронной сети для предсказания интересов владельцев автомобилей на основе их модели автомобиля и возраста.

Целью данного проекта является создание инструмента, который способен автоматически определять предпочтения владельцев автомобилей, чтобы предоставлять им персонализированные предложения и услуги. Для достижения этой цели были поставлены следующие задачи:

* Сбор и подготовка данных: Исследование доступных источников данных о владельцах автомобилей, их моделях и интересах.
* Создание и обучение модели: Разработка нейронной сети для анализа данных о модели автомобиля и возрасте владельца с целью предсказания их интересов в различных областях.
* Оценка результатов: Проведение тестирования и оценка точности и эффективности предсказаний модели.

Этот проект не только позволяет оптимизировать услуги и предложения для владельцев автомобилей, но и демонстрирует потенциал искусственного интеллекта в сегменте персонализации услуг и улучшения взаимодействия с клиентами.

1. Анализ предметной области

В данном разделе проводится анализ предметной области, который позволит обосновать актуальность разработки модуля, приводятся ключевые требования и особенности, которые должны быть реализованы.

* 1. Ключевые аспекты

В современном мире автомобили играют ключевую роль в повседневной жизни людей, а данные об их владельцах и предпочтениях становятся все более ценными для бизнеса. В контексте проекта, цель которого - предсказать интересы владельцев автомобилей, важно рассмотреть несколько ключевых аспектов предметной области.

* + 1. Тенденции рынка автомобилей и потребительских предпочтений:

Понимание текущих тенденций на рынке автомобилей и изменений в потребительских предпочтениях является достаточно важным.

В современном автомобильном секторе наблюдается постоянное развитие и изменение, под влиянием различных факторов, включая технологические инновации, экологические требования и социокультурные изменения. Одним из ключевых аспектов этого развития является постоянное изменение потребительских предпочтений в отношении моделей автомобилей и их характеристик.

В последние годы отмечается растущий интерес к экологически чистым технологиям, таким как электромобили и гибридные автомобили, вследствие повышенного внимания к экологической устойчивости и сокращению выбросов вредных веществ. Помимо этого, наблюдается увеличение спроса на автомобили с продвинутыми технологиями безопасности и удобства, такими как системы автопилота, умные системы информации и развлечений, а также функции связи и дистанционного управления.

Важно также отметить, что предпочтения в выборе автомобиля могут сильно различаться в зависимости от возрастной группы потребителей. Например, молодые водители могут проявлять больший интерес к стильным и технологичным моделям, в то время как старшие покупатели могут уделять большее внимание комфорту, безопасности и эффективности использования.

Таким образом, понимание текущих тенденций рынка автомобилей и предпочтений потребителей в различных возрастных группах является важным аспектом проекта, поскольку это позволяет точнее выявить интересы владельцев автомобилей и предложить им более релевантные услуги и продукты.

* + 1. Методы сбора данных о владельцах и их автомобилях:

Важным этапом является сбор данных о моделях автомобилей и интересах их владельцев. Данный вопрос подробно рассматривался в предыдущем курсовом проекте, поэтому в текущей работе описывается только главная суть.

Сбор данных - важный аспект экономики, поскольку он помогает систематически принимать обоснованные решения. Данные - бесценный инструмент для принятия взвешенных решений, который экономит как время, так и ресурсы. Для сбора данных экономисты и статистики, среди прочих, используют некоторые соответствующие методы, которые помогают им получать информацию, относящуюся к интересующим их предметам.

Сбор данных - это процесс измерения и сбора информации о желаемых переменных таким образом, чтобы можно было находить вопросы, связанные с данными, и использовать их в исследованиях различных типов. Сбор данных является общей чертой обучения по различным дисциплинам, таким как маркетинг, статистика, экономика, естественные науки и т.д. Методы сбора данных могут варьироваться в зависимости от предмета, но конечная цель исследования и честность при сборе данных имеют одинаковое значение во всех вопросах обучения.

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

* Первичный метод сбора данных
* Вторичный метод сбора данных.

Первичные данные собираются исследователями самостоятельно и впервые в рамках исследования. Существуют различные способы сбора первичных данных, некоторые из которых следующие:

* Интервью: Интервью являются наиболее часто используемым методом первичного сбора данных. При проведении интервью для сбора данных используется анкета или исследователь может задавать вопросы непосредственно интервьюируемому. Идея заключается в поиске информации по волнующим темам из ответов респондента. Используемые анкеты можно отправить по электронной почте или уточнить подробности в ходе телефонного интервью.
* Метод Дельфи: В этом методе исследователь запрашивает информацию у группы экспертов. Исследователь может выбрать очное исследование или отправить анкеты по электронной почте. В конце применения метода Дельфи все данные собираются в соответствии с потребностями исследования.
* Проективные методы: Проективные методы используются в исследованиях, которые являются частными или конфиденциальными таким образом, что исследователь считает, что респонденты не раскроют информацию, если будут заданы прямые вопросы. Существует множество типов проективных методик, таких как тематические тесты оценки (TAT), ролевые игры, завершение мультфильма, словесная ассоциация и завершение предложения.
* Интервью в фокус-группе: Здесь собираются несколько человек, чтобы обсудить насущную проблему. В таких интервью обычно участвует от шести до двенадцати человек. Каждый участник выражает свое собственное мнение, и принимается коллективное единогласное решение.
* Метод анкетирования: Здесь используется вопросник для сбора данных от различных групп населения. Для соответствующего исследования используется набор вопросов, и респонденты отвечают на вопросы, прямо или косвенно связанные с вопросником. Этот метод может быть, как открытым, так и закрытым.

Про вторичные методы сбора данных говорить не имеет смысла, так как процесс реализации этих методов значительно усложняет работу, а результат получается такой же.

* + 1. Обработка и анализ данных

Обработка и анализ данных - это процесс преобразования и интерпретации данных с целью выявления закономерностей, паттернов и тенденций. В современном мире данные становятся все более объемными и разнообразными, и для их полноценного анализа необходимо использовать различные методы и инструменты.

* + - 1. Методы предобработки данных

Предобработка данных - это этап обработки данных, нацеленный на подготовку данных для последующего анализа. Она включает в себя такие процессы, как очистка данных от выбросов и ошибок, заполнение пропущенных значений, масштабирование признаков и преобразование категориальных признаков в числовые.

Предварительная обработка данных включает выполнение следующих этапов:

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:

* Дублирующихся записей, несоответствий и ошибок.
* Необычных или аномальных наблюдений.
* Пропущенных значений.

Очистка данных включает:

* Удаление повторяющихся записей, несоответствий и ошибок.
* Обработку необычных или аномальных наблюдений.
* Заполнение пропущенных значений.

При работе с данными важно правильно обрабатывать пропущенные значения. Их можно удалить, пропустить или заполнить. Однако необходимо проявлять осторожность при удалении пропущенных значений, особенно если их доля велика по сравнению с общим объемом данных.

Пропущенные значения можно заполнить:

* Нулевыми значениями.
* Модой, медианой или средним значением.
* Использованием индикаторных переменных.

Преобразование данных включает в себя:

* 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:

* Сокращение размерности.
* Идентификацию и исключение незначительных признаков.
  + - 1. Методы визуализации данных

Визуализация данных - это важный инструмент для исследования данных и выявления визуальных закономерностей. Она позволяет представить данные в понятной и наглядной форме с помощью графиков, диаграмм и карт.

Существует несколько видов визуализации:

* Обычные графические представления: Это тип визуализации, который использует стандартные диаграммы и графики для отображения количественной информации. К ним относятся круговые диаграммы, которые показывают доли целого, линейные диаграммы, отображающие изменение величин с течением времени, гистограммы, которые представляют распределение данных по интервалам, и точечные графики, используемые для отображения взаимосвязи между двумя переменными.
* Преобразование данных: Этот тип визуализации позволяет изменить формат данных для более наглядного восприятия. Например, это могут быть карты, которые помогают визуализировать пространственные данные, или полярные графики, которые отображают данные в полярной координатной системе. Также сюда относятся временные линии, которые представляют изменение данных с течением времени, и графики с параллельными осями, используемые для сравнения нескольких категорий данных.
* Концептуальная визуализация: Этот тип визуализации используется для представления сложных концепций, идей и планов. К ним относятся концептуальные карты, которые помогают организовать и визуализировать связанные идеи, и диаграммы Ганта, используемые для планирования и отслеживания проектов с учетом временных рамок.
* Стратегическая визуализация: Этот тип визуализации используется для отображения данных о работе организаций и их производительности. Сюда входят различные диаграммы, например, графики жизненного цикла продукта или структурные графики организаций, которые показывают их иерархию и взаимосвязи.
* Метафорическая визуализация: Этот тип визуализации использует метафоры для организации и структурирования данных. Например, карта метро может использоваться для отображения связей между различными категориями информации, а пирамиды или деревья - для иллюстрации иерархических структур.
* Комбинированная визуализация: Этот тип визуализации объединяет несколько видов графиков и диаграмм в одной схеме для более полного представления данных. Например, это может быть карта с прогнозом погоды, которая включает в себя данные о температуре, осадках и ветре.Начало формы
  + - 1. Методы машинного обучения

Технология машинного обучения, основанная на анализе данных, начала свое развитие еще в 1950-х годах, когда были созданы первые программы для игры в шашки. За последующие десятилетия общий принцип остался неизменным, но благодаря взрывному росту вычислительной мощности компьютеров значительно усложнились методы анализа данных и прогнозирования, а также расширился спектр задач, которые можно решить с использованием машинного обучения.

Для запуска процесса машинного обучения сначала необходимо загрузить некоторое количество исходных данных, называемых датасетом. Например, это могут быть фотографии собак и кошек, уже помеченные для обозначения к какому классу они относятся. После обучения на этих данных алгоритм может распознавать собак и кошек на новых изображениях без меток. Обучение продолжается и после выдачи прогнозов: чем больше данных проанализировано, тем точнее модель сможет распознавать нужные изображения.

Машинное обучение позволяет компьютерам распознавать не только лица на фотографиях, но и пейзажи, объекты, текст и цифры. Для анализа текста также применяют методы машинного обучения: функция проверки грамматики уже встроена во многие текстовые редакторы и мобильные приложения. Эта функция учитывает не только правильное написание слов, но и контекст, оттенки смысла и другие лингвистические аспекты. Также уже существует программное обеспечение, способное автоматически создавать новостные статьи (например, о событиях в экономике и спорте) без участия человека.

1. Типы задач машинного обучения:
2. Задача регрессии: предсказание числового значения на основе набора признаков. Например, прогноз цены квартиры или ожидаемого дохода магазина.
3. Задача классификации: присвоение объекту одной из категорий на основе его признаков. Например, определение, есть ли на фотографии кошка или собака, или является ли письмо спамом.
4. Задача кластеризации: разделение данных на группы по их сходству. Например, разделение клиентов по уровню платежеспособности.
5. Задача уменьшения размерности: сокращение количества признаков для удобства визуализации данных. Например, сжатие данных для отображения на графиках.
6. Задача выявления аномалий: выявление нестандартных случаев среди данных. Например, выявление мошеннических операций с банковскими картами.

Основные виды машинного обучения:

Основные задачи, решаемые с помощью методов машинного обучения, можно разделить на две категории: обучение с учителем и обучение без учителя. При обучении с учителем модели предоставляются правильные ответы, а при обучении без учителя модели самостоятельно выявляют закономерности в данных.

* Машинное обучение с учителем: модель обучается на данных, для которых известны правильные ответы (например, стоимость квартиры). Модель должна научиться предсказывать эти ответы для новых данных.
* Машинное обучение без учителя: модель обучается на данных, не имея заранее определенных правильных ответов. Здесь модель должна самостоятельно выявлять закономерности в данных, например, группировать их по схожести (кластеризация).

Основные алгоритмы моделей машинного обучения:

1. Дерево принятия решений: Дерево принятия решений - это метод, который помогает принимать решения, используя древовидную структуру, учитывая потенциальные последствия, эффективность и затраты ресурсов. В бизнесе оно строится из набора вопросов с ответами "да" или "нет". После последовательного ответа на эти вопросы можно сделать правильный выбор. Преимущество данного метода в том, что он структурирует проблему и приводит к логическим выводам.
2. Наивная байесовская классификация: Наивная байесовская классификация - это метод классификации, основанный на теореме Байеса, предполагающий независимость признаков. Применяется в областях, таких как определение спама, автоматическая категоризация новостных статей, анализ эмоциональной окраски текста, распознавание лиц и образов на изображениях.
3. Метод наименьших квадратов: Метод наименьших квадратов используется для линейной регрессии, когда нужно подобрать прямую, проходящую через множество точек с минимальной суммой квадратов расстояний от точек до прямой. Этот метод используется для минимизации ошибок в машинном обучении при подгонке данных.
4. Логистическая регрессия: Логистическая регрессия - это метод определения зависимости между переменными, где одна переменная является категориальной, а другие - независимыми. Он используется для предсказания событий в таких областях, как кредитный скоринг, оценка успешности рекламных кампаний, прогнозирование прибыли от товаров и др.
5. Метод опорных векторов (SVM): Метод опорных векторов - это набор алгоритмов для классификации и регрессионного анализа, который строит гиперплоскость для разделения точек двух классов в N-мерном пространстве. SVM помогает решать сложные задачи, такие как сплайсинг ДНК, определение пола по фотографии и вывод рекламных баннеров на сайты.
6. Метод ансамблей: Метод ансамблей в машинном обучении базируется на идее создания множества классификаторов, которые затем объединяются, чтобы принять решение на основе усреднения предсказаний или голосования. Изначально он был представлен как байесовское усреднение, но позже стал более сложным с использованием других алгоритмов:

* Бустинг превращает слабые модели в сильные, создавая ансамбль классификаторов.
* Бэггинг собирает разнообразные классификаторы, обучая их параллельно.
* Метод корректирования ошибок выходного кодирования.

Метод ансамблей мощнее по сравнению с отдельными моделями, так как минимизирует случайные ошибки, уменьшает дисперсию и исключает выход за пределы множества гипотез.

1. Алгоритмы кластеризации: Кластеризация - это процесс группировки объектов таким образом, чтобы объекты внутри одной группы были максимально похожи между собой. Существуют различные алгоритмы кластеризации:

* На основе центра тяжести треугольника.
* На базе подключения.
* Сокращение размерности.
* Основанные на плотности.
* Вероятностные.
* Машинное обучение, включая нейронные сети.

Алгоритмы кластеризации используются в различных областях, таких как биология, социология и информационные технологии.

1. Метод главных компонент (PCA): PCA - это статистический метод, который используется для ортогонального преобразования данных с целью уменьшения размерности и визуализации. Он полезен для сокращения размерности данных и упрощения процесса обучения.
2. Сингулярное разложение: Сингулярное разложение (SVD) - это метод линейной алгебры, который разлагает матрицу на три составляющие: левую сингулярную матрицу, правую сингулярную матрицу и диагональную матрицу сингулярных значений. Он используется для сжатия данных и уменьшения размерности.
3. Анализ независимых компонент (ICA): ICA - это статистический метод, который находит скрытые факторы, влияющие на набор данных, и выделяет их в независимые компоненты. Он эффективен в выявлении скрытых причин явлений и находит применение в различных областях, таких как астрономия, медицина и финансы.
   * + 1. Методы статистического анализа

Статистический анализ - это методы анализа данных, основанные на статистических принципах и методах. Он включает в себя такие процессы, как описательная статистика, проверка гипотез, анализ дисперсии, корреляционный анализ и регрессионный анализ.

Эти методы являются основой для анализа и интерпретации данных в различных областях, включая науку о данных, бизнес-аналитику, медицину, финансы и многие другие. Комбинация различных методов позволяет получить более полное представление о данных и принимать обоснованные решения на основе их анализа.

* + 1. Персонализация услуг и предложений

Персонализация услуг и предложений играет ключевую роль в современном бизнесе, позволяя компаниям лучше удовлетворять потребности и предпочтения клиентов. Этот подход направлен на создание индивидуального опыта для каждого клиента на основе его уникальных характеристик, интересов и поведенческих паттернов.

Одной из основных целей персонализации является предложение клиентам более релевантных и ценных продуктов и услуг. Путем анализа данных о клиентах, их предпочтениях, истории покупок и поведения можно определить индивидуальные потребности и предложить решения, соответствующие конкретному клиенту.

Например, компания может использовать данные о предпочтениях клиента для персонализированных рекомендаций продуктов или услуг. Если клиент интересуется определенными темами или категориями товаров, компания может предложить ему соответствующие продукты или дополнительные услуги.

Персонализация также может применяться в области обслуживания клиентов, где компании стремятся предоставить клиентам индивидуальный и персонализированный опыт взаимодействия. Это может включать в себя персонализированные рекомендации, обратную связь, поддержку и информацию о продуктах или услугах.

Важным аспектом персонализации является также использование данных для оптимизации маркетинговых усилий и ресурсов компании. Путем точного нацеливания коммуникации на конкретные потребности и интересы клиентов компании могут повысить эффективность своих маркетинговых кампаний и улучшить взаимодействие с клиентами.

* 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.

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

* 1. Постановка задачи
     1. Цели и задачи разработки

Целью разработки является автоматизация процесса определения интересов посетителей автостоянки.

Задачи:

* предварительная подготовка данных
* выбор наиболее оптимального метода для анализа и прогнозирования интересов
* реализация выбранного метода
  + 1. Определение ключевых функций

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

* Искусство
* Спорт
* Книги/фильмы
* Наука
* Путешествия
* Кулинария
* Политика и др.

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.

В данном проекте для примера используется распределение из предыдущего курсового проекта, который был направлен на сбор данных. Ниже приведен список вопросов, которые там применялись, а также структура бланка на рисунке 1:

* 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.

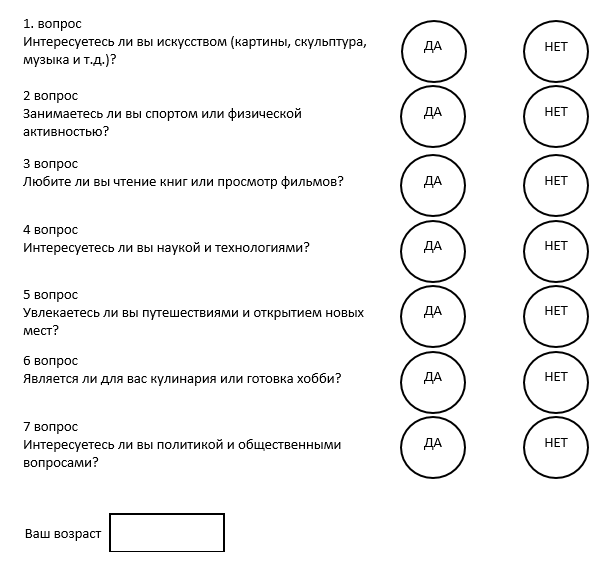


Рисунок 1 – Опрос

Точность предсказания должна быть не ниже 80%.

Вывод

В данном разделе был проведен анализ предметной области, подчеркнута актуальность разработки модуля для анализа интересов посетителей парковки. Исходя из рассмотренных факторов, определены ключевые требования и функции, которые должны быть реализованы в разрабатываемом программном продукте.

Модуль представляет собой средство для более точного и автоматизированного предсказания интересов посетителей парковки. Определение интересов клиентов позволяет оптимизировать уровень сервиса, а также разрабатывать персонализированные предложения для посетителей.

Таким образом, разработка данного модуля является актуальной и перспективной в контексте современных тенденций в сфере общественных мест и сервисной индустрии.

1. Выбор метода

В этом разделе будут рассмотрены основные методы, а также определен наиболее подходящий.

* 1. Описание искусственных нейронных сетей

Человеческий мозг проявляет выдающиеся способности в решении задачи распознавания образов, превосходя в этом аспекте даже самые мощные цифровые компьютеры. Это обусловлено особенностями информационной обработки мозга, которые существенно отличаются от применяемых в обычных цифровых компьютерах методов. Мозг человека представляет собой сложную, нелинейную, параллельную систему обработки информации. Он способен так организовать свои структурные компоненты, чтобы успешно выполнять конкретные задачи, такие как распознавание образов, с многократным преимуществом в скорости по сравнению с самыми быстрыми компьютерами.

В данной работе предлагается использовать искусственные нейронные сети (ИНС). Согласно, нейронную сеть можно определить следующим образом:

Искусственная нейронная сеть представляет собой масштабный распределенный параллельный процессор, состоящий из элементарных блоков для обработки информации, называемых нейронами. Эти нейроны накапливают экспериментальные знания и предоставляют их для последующей обработки.

Перечислим 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.

Процесс обучения сети, или настройки, представляет собой процесс модификации свободных параметров нейронной сети, осуществляемый путем моделирования её взаимодействия с окружающей средой. Тип обучения определяется методом выбора 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 реальными выходными значениями нейронной сети (ANN) и желаемыми выходами. Его детальное описание можно найти в источниках.

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.

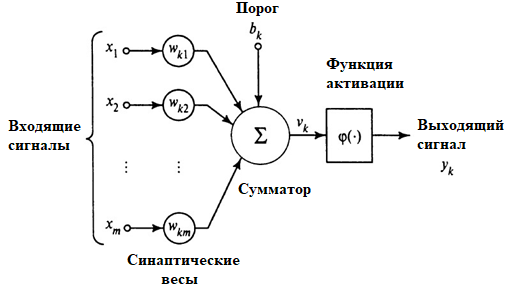


Рисунок 4 – Нелинейная модель нейрона

Нейрон, в контексте нейронных сетей, представляет собой ключевую единицу обработки информации. На иллюстрации 4 мы можем увидеть модель нейрона, которая лежит в основе искусственных нейронных сетей. Эта модель включает в себя три основных компонента:

Ансамбль синапсов (или связей), каждый из которых имеет свой собственный вес или силу.

Сумматор, который складывает взвешенные входные сигналы, учитывая их веса, связанные с соответствующими синапсами нейрона.

Функция активации, которая ограничивает амплитуду выходного сигнала нейрона.

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

Математически, работу нейрона можно описать следующими уравнениями:





где , , … ,  – входные сигналы; , , … ,  — синаптические весы нейрона k;  - линейная комбинация входных действий (выход линейного комбинатора);  - порог;  – функция активации;  является выходным сигналом нейрона.

Постсинаптический потенциал рассчитывается следующим образом:



Функции активации – это специальные инструменты, которые формируют выход сигнала нейрона, в зависимости от воздействия на него локального поля. Давайте рассмотрим несколько разнообразных функций активации. Существует множество других функций активации.

Одной из таких функций является сигмовидная функция. Она выделяется своей способностью стремительно переходить от линейного к нелинейному поведению. Логистическая функция является примером сигмовидной функции и может быть представлена следующим образом:



Другой функцией активации является линейный выпрямитель (ReLU), заданный следующим выражением:

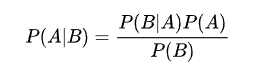


* 1. Наивный байесовский классификатор

Наивный байесовский классификатор (Naive Bayes classifier) — вероятностный классификатор на основе формулы Байеса со строгим (наивным) предположением о независимости признаков между собой при заданном классе, что сильно упрощает задачу классификации из-за оценки одномерных вероятностных плотностей вместо одной многомерной.

В данном случае, одномерная вероятностная плотность — это оценка вероятности каждого признака отдельно при условии их независимости, а многомерная — оценка вероятности комбинации всех признаков, что вытекает из случая их зависимости. Именно по этой причине данный классификатор называется наивным, поскольку позволяет сильно упростить вычисления и повысить эффективность алгоритма. Однако такое предположение не всегда является верным на практике и в ряде случаев может привести к значительному ухудшению качества прогнозов.

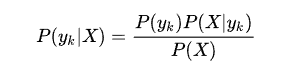
Сама же формула Байеса выглядит следующим образом:



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.

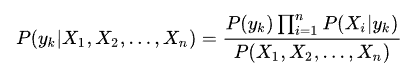
А в контексте машинного обучения формула Байеса приобретает следующий вид:



где:

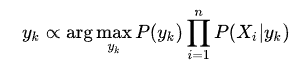
* P(yk|X) — апостериорная вероятность принадлежности образца к классу yk с учётом его признаков X;
* P(X|yk) — правдоподобие, то есть вероятность признаков X при заданном классе yk;
* P(yk) — априорная вероятность принадлежности случайно выбранного наблюдения к классу yk;
* P(X) — априорная вероятность признаков X.

Если объект описывается не одним, а несколькими признаками X1, X2,...,Xn, то формула принимает вид:



На практике числитель данной формулы представляет наибольший интерес, поскольку знаменатель зависит только от признаков, а не от класса, и поэтому часто он опускается при сравнении вероятностей разных классов. В конечном счёте правило классификации будет пропорционально выбору

класса с максимальной апостериорной вероятностью:



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

* 1. Логистическая регрессия

Логистическая регрессия — это метод анализа данных, который использует математику для поиска взаимосвязей между двумя факторами данных. Затем эта взаимосвязь используется для прогнозирования значения одного из этих факторов на основе другого. Предсказание обычно имеет конечное количество результатов, например, «да» или «нет».

Модель логистической регрессии имеет несколько составляющих:

* Уравнения

В математике уравнения дают связь между двумя переменными: *x* и *y*. Эти уравнения или функции можно использовать для построения графика вдоль осей x и y, введя разные значения *x* и *y*. Например если построить график для функции *y* = 2\* *x*, получится прямая линия, как показано ниже. Поэтому эту функцию также называют линейной функцией.

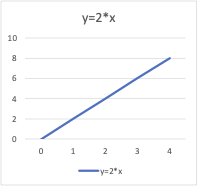


Рисунок 2 – Функция

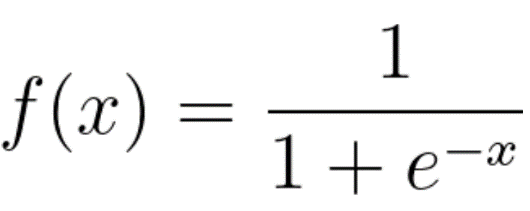
* Переменные

В статистике переменные – это факторы данных или атрибуты, значения которых различаются. Для любого анализа определенные переменные являются независимыми или объясняющими переменными. Эти атрибуты являются причиной результата. Другие переменные являются зависимыми или переменными ответа; их значения зависят от независимых переменных. В целом логистическая регрессия исследует, как независимые переменные влияют на одну зависимую переменную, рассматривая исторические значения обеих переменных.

В приведенном выше примере x называется независимой переменной, предикторной переменной или объясняющей переменной, потому что она имеет известное значение. Y называется зависимой переменной, переменной результата или переменной отклика, потому что ее значение неизвестно.

* Функция логистической переменной

Логистическая регрессия – это статистическая модель, которая использует логистическую или логитную функцию в математике в качестве уравнения между x и y. Логитная функция отображает y как сигмовидную функцию от x.



Если построить это уравнение логистической регрессии, получится S-образная кривая, как показано ниже.

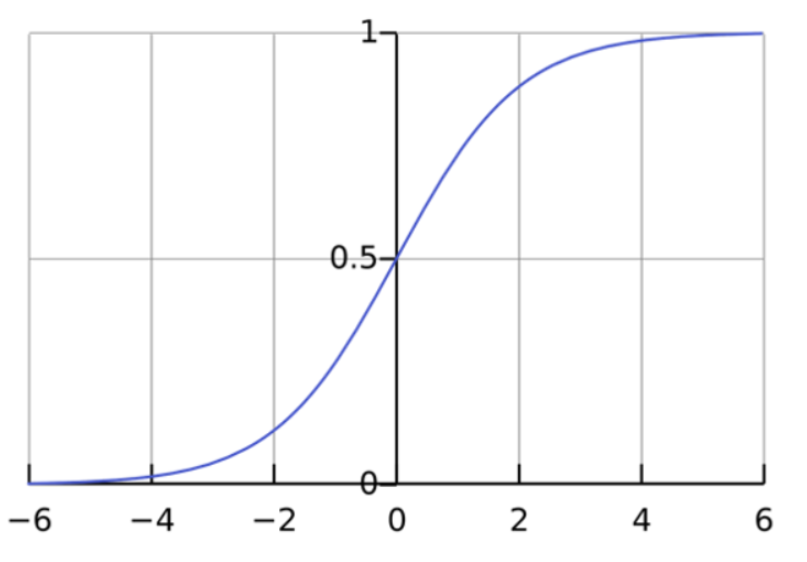


Рисунок 3 – Функция сигмоиды

Как можно увидеть, логитная функция возвращает только значения от 0 до 1 для зависимой переменной, независимо от значений независимой переменной. Так логистическая регрессия оценивает значение зависимой переменной. Методы логистической регрессии также моделируют уравнения между несколькими независимыми переменными и одной зависимой переменной.

* 1. Обоснование выбора

Ниже приведены достоинства и недостатки рассмотренных методов.

**Логистическая регрессия:**

*Плюсы:*

* Простота реализации и интерпретации.
* Эффективность при линейной разделяющей гиперплоскости.
* Меньшая склонность к переобучению на небольших наборах данных.
* Хорошо работает с линейно разделимыми данными.

*Минусы:*

* Ограничение на моделирование нелинейных зависимостей.
* Ограниченная способность к обработке больших объемов данных.
* Не учитывает взаимосвязи между признаками.

**Наивный байесовский классификатор:**

*Плюсы:*

* Простота и скорость обучения.
* Эффективность на небольших наборах данных.
* Меньшая склонность к переобучению.

*Минусы:*

* Предположение о независимости признаков может быть слишком сильным и нереалистичным.
* Не способен улавливать сложные взаимосвязи между признаками.
* Требует хорошо подготовленных данных.

**Нейронные сети:**

*Плюсы:*

* 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:*

* Требуют большого объема данных для обучения.
* Сложность интерпретации модели.
* Большое количество настраиваемых параметров, требующих подбора.

**Заключение:**

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

Вывод

В данном разделе были рассмотрены основные методы и также определен наиболее подходящий.

Нейронные сети обеспечивают лучшую производительность на сложных задачах, хотя и требуют больше вычислительных ресурсов и данных для обучения. Нейронные сети представляют собой мощный инструмент для решения разнообразных задач машинного обучения и обладают большим потенциалом для применения в различных областях.

1. Проектирование

В этом разделе будет рассмотрен процесс проектирования разрабатываемого модуля. Будет определен функционал и особенности.

* 1. 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 4.

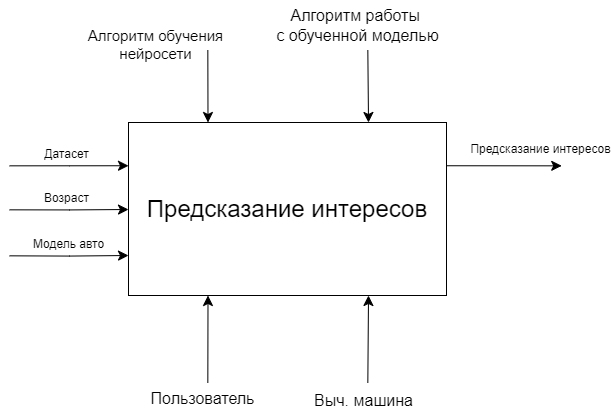


Рисунок 4 – Контекстная диаграмма IDEF0

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:

* обучение нейросети. На вход поступает датасет, а на выходе получается обученная модель
* определение интересов. На вход поступает обученная модель, а на выход предсказанные интересы

Детализированная контекстная диаграмма представлена на рисунке 5.



Рисунок 5 – Детализированная контекстная диаграмма IDEF0

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

Так же стоит определить формат датасета, который используется для обучения нейросети. В него входит вся информация, которую можно получить с помощью разработанного в прошлом курсовом проекте. Следовательно, датасет имеет 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
  1. 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 6.



Рисунок 6 – 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 оптимизатора: 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.
    1. Описание 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.

1. 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.

1. 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.

1. 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 используют 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
* epochs – количество поколений

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

В таблице 1 представлены результаты лучших моделей.

Таблица 1 – Список лучших моделей

|  |  |  |
| --- | --- | --- |
| Номер модели | Точность | Ошибка |
| 1 | 0,896 | 0,117 |
| 2 | 0,823 | 0,296 |
| 3 | 0,796 | 0,244 |
| 4 | 0.780 | 0,305 |
| 5 | 0,775 | 0,398 |

На рисунке 7 представлена структура лучшей модели.

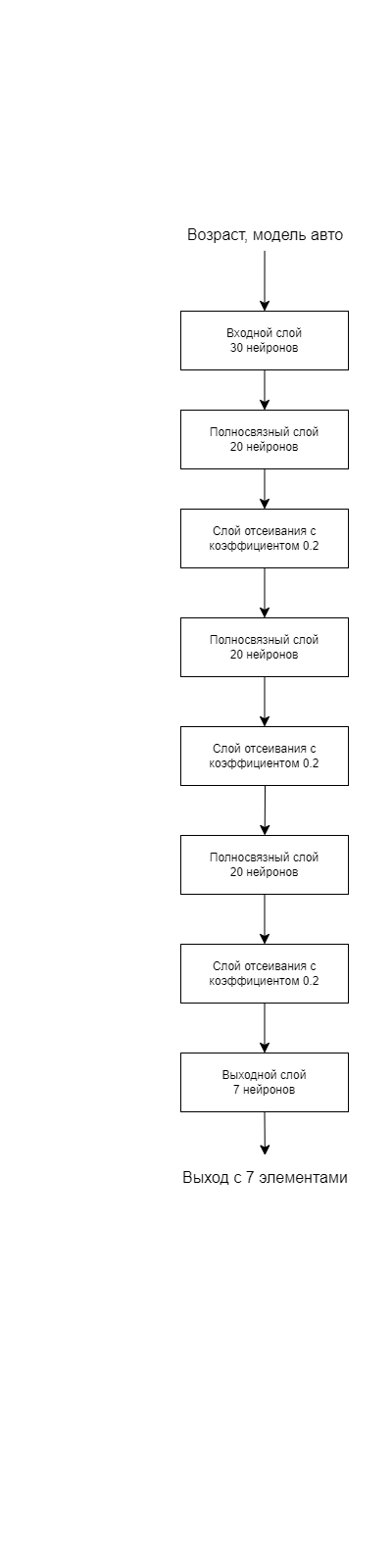


Рисунок 7 – Структура нейронной сети

Из рисунка 7 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.

* 1. 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 таблице 2.

Таблица 2 – Пример форматирования результата

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Искусство | Спорт | Книги/фильмы | Наука | Путешествия | Кулинария | Политика |
| 93.0% | 97.3% | 68.61% | 71.92% | 0.34% | 3.82% | 92.60% |

Исходя из необходимых компонентов, которые должны быть на экране, можно создать макет экранной формы. Он представлен на рисунке 8.



Рисунок 8 – Макет экранной формы

Рассмотрим алгоритм работы модуля. Схема алгоритма представлена на рисунке 9.



Рисунок 9 – Алгоритм модуля

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.
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.

Вывод

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.

1. Программная реализация

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

* 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:

1. Библиотека pandas: Используется для работы с данными в формате DataFrame, включая загрузку данных из CSV-файла и их предварительную обработку.
2. Библиотека numpy: Применяется для работы с числовыми данными и выполнения математических операций, таких как нормализация числовых данных.
3. 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.
4. 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.

Датасет имеет следующие поля, определенные в предыдущем курсовом проекте:

* 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

Входными параметрами являются поля Model и Age, а остальные поля выходными.

1. 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.
2. Разделение данных: Данные разделяются на обучающий и тестовый наборы с использованием функции train\_test\_split из библиотеки sklearn.
3. 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.
4. Создание модели KerasClassifier: Создается модель KerasClassifier, обертка над моделью Keras, для использования с методом кросс-валидации GridSearchCV из библиотеки sklearn.
5. Задание сетки параметров для подбора: Задаются параметры для поиска лучших комбинаций параметров модели с помощью GridSearchCV.
6. Создание объекта GridSearchCV: Создается объект GridSearchCV с указанными параметрами для поиска оптимальных гиперпараметров модели.
7. Определение обратного вызова EarlyStopping: Обратный вызов EarlyStopping определяется для автоматической остановки обучения, если происходит переобучение модели.
8. Подгонка объекта GridSearchCV к данным: Объект GridSearchCV подгоняется к обучающим данным с использованием метода fit, включая обратный вызов EarlyStopping.
9. Вывод лучших параметров: Выводятся лучшие параметры модели, найденные с помощью GridSearchCV.
10. Оценка модели с лучшими параметрами: Лучшая модель из GridSearchCV оценивается на тестовом наборе данных, и выводится точность предсказания.
    1. Реализация модуля

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

1. PyQt6: PyQt6 является набором библиотек для Python, предоставляющим интерфейс для работы с графическими приложениями на основе Qt, кросс-платформенного 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.
2. TensorFlow: TensorFlow в модуле используется для загрузки обученной модели нейронной сети, выполнения предсказаний и обработки данных.
3. NumPy: В модуле NumPy используется для предобработки данных и работы с массивами.
4. Pandas: В модуле Pandas используется для загрузки данных из CSV-файла и предварительной обработки данных перед подачей их на вход нейронной сети.
5. scikit-learn: В модуле scikit-learn используется для предобработки данных и выполнения OneHotEncoding для категориальных переменных.

Рассмотрим первый скрипт более подробно:

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 модуля: 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()).
   1. Результат

В ходе выполнения курсового проекта был разработан модуль, который выполняет поставленную задачу прогнозирования интересов посетителей автостоянки.

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.

Ниже, на рисунках 10-11, приведены экранные формы разработанного модуля.

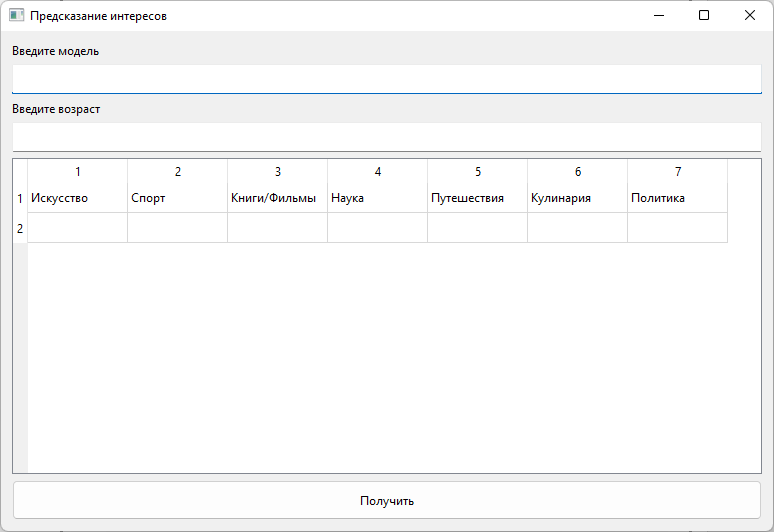


Рисунок 10 – Окно модуля

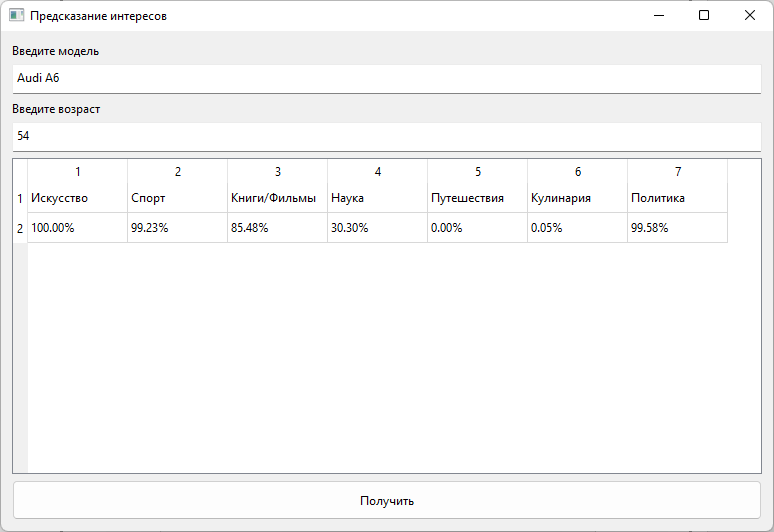


Рисунок 11 – Результат работы модуля

Вывод

В данном разделе была описана программная реализация разрабатываемого модуля. Также были описаны средства, которые использовались при создании и основные функции, и полученный результат.

1. Оценка качества

В данном разделе будет проведено тестирование разработанного программного модуля, а также проведена оценка качества его работы.

Для тестирования было подготовлено 100 тестовых строк, примеры которых приведены в таблице 3.

Таблица 3 - Примеры

|  |  |  |
| --- | --- | --- |
| Модель | Возраст | Коэффициенты |
| Acura ILX | 52 | 0,0,0,0,1,1,0 |
| Audi A6 | 54 | 1,1,1,0,0,0,1 |
| BMW X6 xDrive35i | 38 | 1,1,0,1,0,0,1 |

Так как изначально в датасете, который использовался для обучения сети, интересы людей обозначены не вероятностями, а четкими коэффициентами, а модель выдает вероятность присутствия интереса у человека, то процесс оценки качества затрудняется. По вероятности нельзя точно сказать, интересуется ли человек одной из обозначенных областей или нет. Для упрощения будем считать, что вероятности больше либо равны 0.5 являются знаком, что интерес присутствует. В противном случае же наоборот – интерес отсутствует. В датасете же присутствие интереса обозначено цифрой «1», а отсутствие «0».

Результат прогона тестовых строк показал, что точность модели составляет примерно 84%. Не самый высокий результат может быть обусловлен тем, что датасет, использующийся при обучении сгенерирован и скорее всего имеет некоторые противоречия и неверное отображение действительности. Предполагается, что при сборе достаточного количества реальных данных, точность поднимется.

Примеры тестовых прогонов приведены на рисунках 12-14

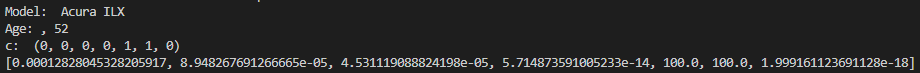


Рисунок 12 – Тест 1

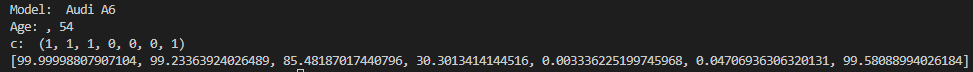


Рисунок 13 – Тест 2

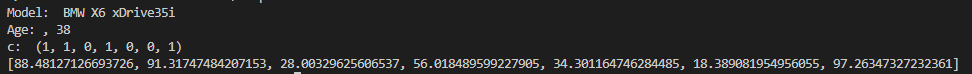


Рисунок 13 – Тест 3

## Заключение

В ходе выполнения проекта был разработан программный модуль, который позволяет прогнозировать интересы посетителей парковки. Для этого было использовано машинное обучение.

Основной целью разработки является автоматизация процесса определения интересов посетителей автостоянки.

Задачи:

* предварительная подготовка данных
* выбор наиболее оптимального метода для анализа и прогнозирования интересов
* реализация выбранного метода

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

Для реализации функционала модуля были разработаны 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.

В результате работы был разработан готовый прототип программного модуля, который может быть использован для прогнозирования интересов посетителей парковки.

Таким образом, проект "Разработка программного модуля для прогнозирования интересов посетителей парковки" был успешно выполнен. Разработанный модуль предоставляет возможность автоматизировать процесс прогнозирования интересов, что позволяет повысить эффективность работы парковки и улучшить уровень сервиса.

## Список литературы

1. API Documentation [Электронный ресурс]:

<https://www.tensorflow.org/api_docs>

1. Машинное обучение и анализ данных. [Электронный ресурс]:

<https://habr.com/ru/companies/yandex/articles/206058/>

1. Нейронная сеть [Электронный ресурс]:

<https://ru.wikipedia.org/wiki/Нейронная_сеть>

1. Нейросети и глубокое обучение, глава 3, ч.3: как выбрать гиперпараметры нейросети? [Электронный ресурс]:

<https://habr.com/ru/articles/460711/>

1. scikit-learn [Электронный ресурс]:

<https://scikit-learn.org/stable/index.html>

1. PyQt6 — полное руководство [Электронный ресурс]:

https://habr.com/ru/companies/skillfactory/articles/599599/

1. GridSearchCV [Электронный ресурс]:

https://scikitlearn.org/stable/modules/generated/sklearn.model\_selection.GridSearchCV.html

Приложение А

(Обязательное)

Схемы алгоритмов



Рисунок 14 – Схема алгоритма модуля



Рисунок 15 – Схема обучние нейросети

Приложение Б

(Обязательное)

Экранные формы

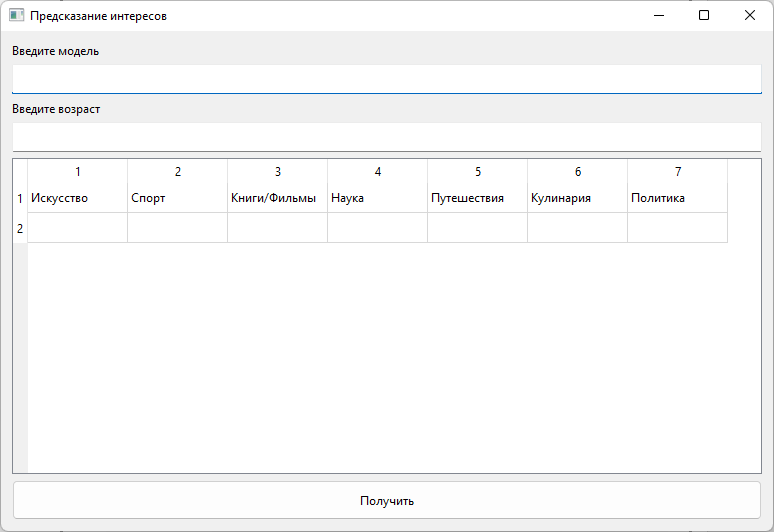


Рисунок6– Окно модуля

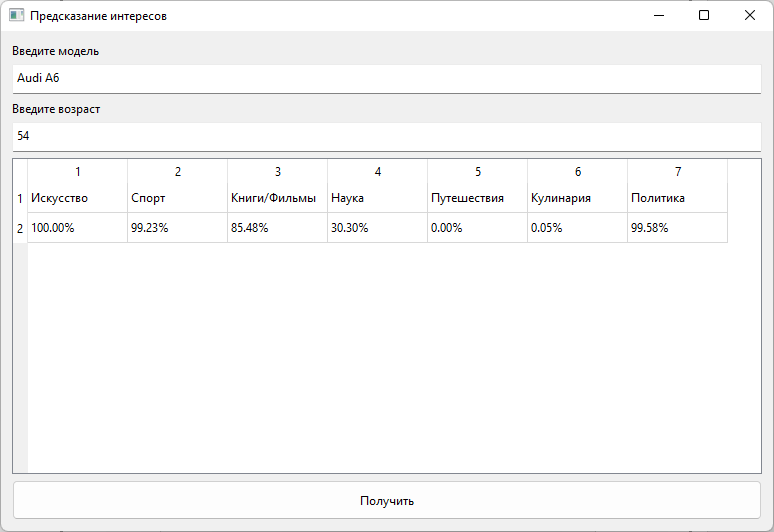


Рисунок 17 – Результат работы модуля

Приложение В

(Обязательное)

Структура программы

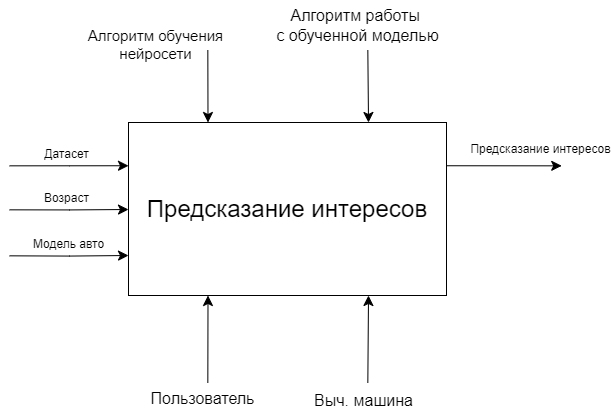


Рисунок 18 – Контекстная диаграмма IDEF0



Рисунок 19 – Детализированная контекстная диаграмма IDEF0

Приложение Г

(Обязательное)

Структура нейросети

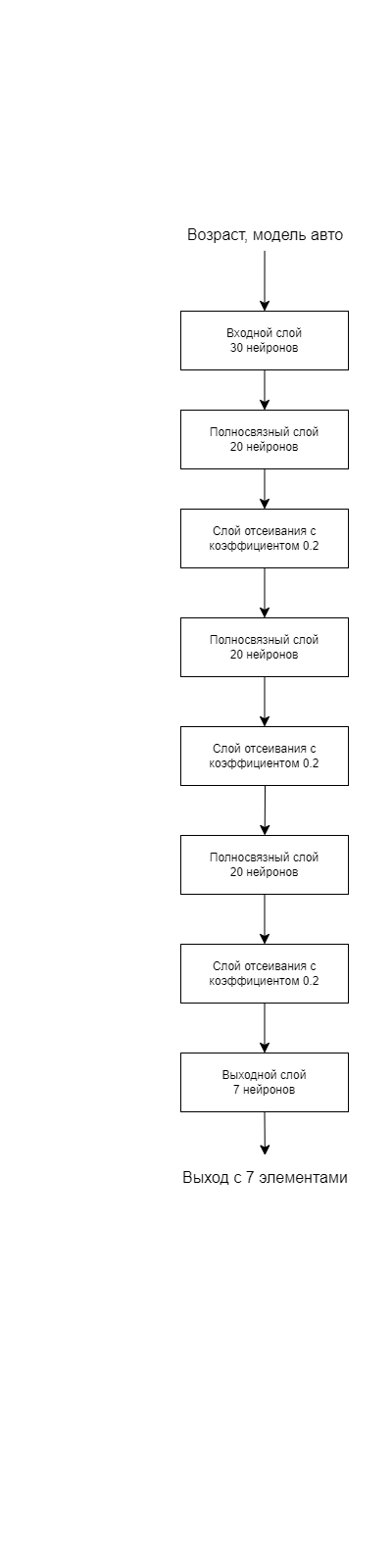


Рисунок 20 – Структура нейросети

Приложение Д

(Обязательное)

Программный код

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

# Загрузка данных

data = pd.read\_csv('data/datanew.csv')

# Преобразование категориального столбца "Model" в числовые значения с помощью OneHotEncoding

encoder = OneHotEncoder()

model\_encoded = encoder.fit\_transform(data[['Model']]).toarray()

# Нормализация числовых данных (возраст)

age = data[['Age']].values

age\_normalized = (age - np.mean(age)) / np.std(age)

# Объединение преобразованных данных

X = np.concatenate([age\_normalized, model\_encoded], axis=1)

# Выходные данные (интересы)

y = data[['Art', 'Sport', 'Book/Films', 'Science', 'Travel', 'Cooking', 'Politics']].values

# Разделение данных на обучающий и тестовый наборы

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Функция для создания модели с указанными параметрами

def create\_model(optimizer='adam', num\_hidden\_layers=2, num\_neurons=20, activation='relu', dropout\_rate=0.2, learning\_rate=0.001, epochs=10):

    model = tf.keras.Sequential()

    model.add(tf.keras.layers.Dense(num\_neurons, activation=activation, input\_shape=(X.shape[1],)))

    model.add(tf.keras.layers.Dropout(dropout\_rate))

    for \_ in range(num\_hidden\_layers - 1):

        model.add(tf.keras.layers.Dense(num\_neurons, activation=activation))

        model.add(tf.keras.layers.Dropout(dropout\_rate))

    model.add(tf.keras.layers.Dense(7, activation='sigmoid'))

    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)

    model.compile(optimizer=optimizer, loss='binary\_crossentropy', metrics=['accuracy'])

    return model

# Создание модели KerasClassifier

model = KerasClassifier(build\_fn=create\_model, verbose=0)

# Задание сетки параметров для подбора

param\_grid = {

    'num\_hidden\_layers': [2, 3, 4, 5],

    'num\_neurons': [20, 30, 40],

    'activation': ['relu', 'tanh'],

    'dropout\_rate': [0.1, 0.2],

    'optimizer': ['adam', 'rmsprop', 'sgd'],

    'learning\_rate': [0.001, 0.01, 0.1],

    'epochs': [10, 20, 30, 40, 50, 100, 150]

}

# Создание объекта GridSearchCV

grid\_search = GridSearchCV(estimator=model, param\_grid=param\_grid, cv=3, verbose=1)

# Определение обратного вызова EarlyStopping

early\_stopping = EarlyStopping(monitor='val\_loss', patience=5, restore\_best\_weights=True)

# Подгонка объекта GridSearchCV к данным с использованием обратного вызова EarlyStopping

grid\_result = grid\_search.fit(X\_train, y\_train, callbacks=[early\_stopping])

# Вывод наилучших параметров

print("Best: %f using %s" % (grid\_result.best\_score\_, grid\_result.best\_params\_))

# Оценка модели с лучшими параметрами

best\_model = grid\_search.best\_estimator\_

loss, accuracy = best\_model.evaluate(X\_test, y\_test)

print(f'Test Accuracy: {accuracy}')

import numpy as np

import pandas as pd

import tensorflow as tf

from sklearn.preprocessing import OneHotEncoder

# Загрузка обученной модели

model = tf.keras.models.load\_model("modelnew2.keras")

# Преобразование новых данных

def preprocess\_new\_data(new\_data, full\_data):

    # Преобразование категориального столбца "Model" в числовые значения с помощью OneHotEncoding

    encoder = OneHotEncoder()

    encoder.fit(full\_data[['Model']])

    model\_encoded = encoder.transform(new\_data[['Model']]).toarray()

    # Нормализация числовых данных (возраст)

    age = new\_data[['Age']].values

    age\_normalized = (age - np.mean(full\_data['Age'])) / np.std(full\_data['Age'])

    # Объединение преобразованных данных

    X\_new = np.concatenate([age\_normalized, model\_encoded], axis=1)

    return X\_new

# Получение предсказаний

def get\_predictions(X\_new):

    predictions = model.predict(X\_new)

    # binary\_predictions = np.round(predictions)

    return predictions

# Вывод результатов

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):

    # Пример новых данных для тестирования

    new\_data = pd.DataFrame({'Model': [model], 'Age': [int(age)]})

    # Преобразование новых данных

    X\_new = preprocess\_new\_data(new\_data, full\_data)

    # Получение предсказаний только для одного примера

    binary\_predictions = get\_predictions(X\_new)

    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]

    # Запись в массив

    predictions\_array = [art\_prediction, sport\_prediction, book\_films\_prediction, science\_prediction, travel\_prediction, cooking\_prediction, politics\_prediction]

    return predictions\_array

if \_\_name\_\_ == "\_\_main\_\_":

    # model\_auto = "Acura ILX"

    # age = "52"

    # cc = 0,0,0,0,1,1,0

    # model\_auto = "Audi A6"

    # age = "54"

    # cc = 1,1,1,0,0,0,1

    model\_auto = "BMW X6 xDrive35i"

    age = "38"

    cc = 1,1,0,1,0,0,1

    predictions = main(model\_auto, age)

    print('Model: ', model\_auto)

    print('Age: ,', age)

    print('c: ', cc)

    for n in range(7):

        predictions[n] = float(predictions[n]) \* 100

    print(predictions)

import sys

from PyQt6.QtWidgets import QApplication, QWidget, QPushButton, QLabel, QVBoxLayout, QTextEdit, QTableWidget, QTableWidgetItem

import test\_single

list = ["Искусство", "Спорт", "Книги/Фильмы", "Наука", "Путешествия", "Кулинария", "Политика"]

class MyWindow(QWidget):

    def \_\_init\_\_(self):

        super().\_\_init\_\_()

        self.initUI()

    def initUI(self):

        self.setWindowTitle('Предсказание интересов')

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

        self.label\_model = QLabel('Введите модель', self)

        self.label\_age = QLabel('Введите возраст', self)

        self.text\_edit\_model = QTextEdit(None, self)

        self.text\_edit\_model.setFixedSize(750, 30)

        self.text\_edit\_age = QTextEdit(None, self)

        self.text\_edit\_age.setFixedSize(750, 30)

        self.table = QTableWidget(self)

        self.table.setColumnCount(7)

        self.table.setRowCount(2)

        for col in range(7):

            item = QTableWidgetItem(list[col])

            self.table.setItem(0, col, item)

        self.button = QPushButton('Получить', self)

        self.button.setFixedSize(750, 40)

        self.button.clicked.connect(self.proc)

        vbox = QVBoxLayout()

        vbox.addWidget(self.label\_model)

        vbox.addWidget(self.text\_edit\_model)

        vbox.addWidget(self.label\_age)

        vbox.addWidget(self.text\_edit\_age)

        vbox.addWidget(self.table)

        vbox.addWidget(self.button)

        self.setLayout(vbox)

    def proc(self):

        model = self.text\_edit\_model.toPlainText()

        age = self.text\_edit\_age.toPlainText()

        if (model != "") & (age != ""):

            predictions = test\_single.main(model, age)

            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 = MyWindow()

    window.show()

    sys.exit(app.exec())