МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
**«Южно-Уральский государственный университет
(национальный исследовательский университет)»**
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

РАБОТА ПРОВЕРЕНА

Рецензент
Доцент кафедры ВМиИТ
ФГБОУ ВО «ЧелГУ»,
к.ф.-м.н.

_____ А.Ю. Маковецкий

«___»_____ 2024 г.

ДОПУСТИТЬ К ЗАЩИТЕ

Заведующий кафедрой, д.ф.-м.н.,
профессор

_____ Л.Б. Соколинский

«___»_____ 2024 г.

Разработка приложения для классификации литературных произведений по жанрам с помощью методов машинного обучения

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
ЮУрГУ – 02.04.02.2024.308-1393.ВКР

Научный руководитель,
доцент кафедры СП, к.ф.-м.н.
_____ Т.Ю. Маковецкая

Автор работы,
студент группы КЭ-220
_____ Д.С. Курочкин

Ученый секретарь
(нормоконтролер)
_____ И.Д. Володченко
«___»_____ 2024 г.

Челябинск, 2024 г.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

УТВЕРЖДАЮ
Зав. кафедрой СП
_____ Л.Б. Соколинский
29.01.2024 г.

ЗАДАНИЕ
на выполнение выпускной квалификационной работы магистранта
студенту группы КЭ-220
Курочкину Дмитрию Сергеевичу,
обучающемуся по направлению
02.04.02 «Фундаментальная информатика и информационные технологии»
(магистерская программа «Машинное обучение и анализ больших данных»)

1. **Тема работы** (утверждена приказом ректора от 22.04.2024 г. № 764-13/12)
Разработка приложения для классификации литературных произведений по жанрам с помощью методов машинного обучения.
2. **Срок сдачи студентом законченной работы:** 20.05.2024 г.
3. **Исходные данные к работе**

3.1. Python documentation. [Электронный ресурс] URL: https://www.python.org/doc/ (дата обращения: 01.02.2024 г.).
3.2. NLTK documentation. [Электронный ресурс] URL: https://www.nltk.org/index.html (дата обращения: 09.02.2024 г.).
3.3. Рынок электронных книг. [Электронный ресурс] URL: https://www.mordorintelligence.com/ru/industry-reports/e-book-market (дата обращения: 27.02.2024 г.).
3.4. Рынок электронных книг в России. [Электронный ресурс] URL: https://clck.ru/396fki (дата обращения: 27.02.2024 г.).
3.5. Рост рынка электронных книг. [Электронный ресурс] URL: https://www.retail.ru/news/litres-v-iii-kvartale-rynok-tsifrovykh-knig-vyros-pochti-na-tret/ (дата обращения: 27.02.2024 г.).
3.6. Django documentation. [Электронный ресурс] URL: https://docs.djangoproject.com/en/5.0/ (дата обращения: 17.02.2024 г.).

4. Перечень подлежащих разработке вопросов

4.1. Провести обзор аналогов и научной литературы.

4.2. Осуществить сбор и предобработку данных для обучения.

4.3. Разработать модель машинного обучения и оценить результаты работы.

4.4. Реализовать приложение.

4.5. Провести тестирование приложения.

5. Дата выдачи задания: 29.01.2024 г.

Научный руководитель,
доцент кафедры СП, к.ф.-м.н. Т.Ю. Маковецкая

Задание принял к исполнению Д.С. Курочкин

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность

В настоящее время все больше популярности набирают сервисы с использованием нейронных сетей и машинным обучением. Они помогают людям как в повседневной жизни, так и в работе. Также организации стремятся внедрять к себе как готовые сервисы с искусственным интеллектом, так и реализовывать свои модели нейронных сетей для покрытия своих нужд. Не хотят отставать и книжные онлайн магазины.

С каждым годом объем литературных произведений становится все больше. С постоянным увеличением количеств книг и статей часто бывает сложно определить жанр произведения только по его описанию или названию. Сервис для классификации литературных произведений по жанрам мог бы облегчить работу по распределению электронных книг в каталогах.

Постановка задачи

Целью выпускной квалификационной работы является разработка приложения для классификации литературных произведений по жанрам с помощью методов машинного обучения. Для достижения поставленной цели необходимо решить следующие задачи:

1) провести обзор аналогов и научной литературы в данной области;
2) осуществить сбор и предобработку данных для обучения;
3) разработать модель машинного обучения и оценить результаты ее работы;
4) реализовать приложение;
5) провести тестирование приложения.

Структура и содержание работы

Работа состоит из введения, трех глав, заключения и списка литературы. Объем работы составляет 52 страницы, объем списка литературы – 36 источников.

В первой главе описывается анализ предметной области. А именно предмет исследования, в данном случае литература в виде электронном виде.

Сайты и сервисы, которые работают в сфере электронных книг, будь то продажа, подписка или бесплатный доступ. Обзор рынка: прогнозы и ожидания. Далее были проанализированы существующие сервисы по работе с текстом, использующие методы машинного обучения или нейронных сетей, их функции и возможности. В конце первой главы был проведен обзор научной литературы в данной сфере: какие данные использовались, как их обрабатывали и каким способом обучали модели.

Вторая глава посвящена разработке и тестированию модели машинного обучения. Был проведен сбор своего набора данных и его дальнейшая предобработка. После этого использовались несколько подходов машинного обучения для классификации книг по жанрам. В конце был проведен анализ и выбор наилучшего метода.

В третьей главе содержится описание разработки и тестирования приложения с графическим интерфейсом для работы с моделью, а именно определения требований к приложению, его архитектура. Далее была реализована разработка самого приложения с возможностью обработки и классификации текстов из второй главы. В конце было проведено тестирование приложения.

1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1. Описание предметной области

Литература – одно из ключевых искусств, воплощающих искусство слова. Термин «литература» охватывает также все произведения человеческой мысли, зафиксированные в письменной форме и имеющие социальную ценность. Существуют разные виды литературы: художественная, техническая, научная, публицистическая, справочная, эпистолярная и другие. Однако в обычном и более узком смысле «литература» относится к художественным и письменным произведениям [1].

Литература является неотъемлемой частью культуры и искусства и занимает важное место в мире слова. Существуют следующие виды литературы, каждый из которых имеет свое значение и конкретную цель [2].

1. Художественная литература – особый вид искусства, которое создает изображение мира, в качестве материала используя слова и язык. Представителями этого жанра являются эпос, проза, лирика, драма, эссе, литература для детей и переводная литература.

2. Документальная проза – форма литературы, которая строится на реальных событиях с минимальными добавлениями художественного вымысла. Она включает биографические рассказы о выдающихся людях, истории событий, описания страноведческого характера, а также расследования громких преступлений.

3. Научная литература представляет собой разновидность литературы, которая информирует ученых и специалистов о последних научных достижениях. Научная литература создается на основе исследований и теоретических обобщений, выполненных с использованием научного метода. Формы научной литературы включают монографии, обзорные статьи, доклады, авторефераты, рефераты и рецензии.

4. Справочная литература – литературные источники, предназначенные для получения основной информации по различным вопросам. Они

включают в себя словари, справочники и энциклопедии, которые предлагают систематизированное описание знаний в определенной области.

5. Учебная литература – литература, которая предназначена для обучения и изучения определенных предметов и дисциплин. Она включает в себя учебники и сборники задач, которые ставят своей задачей систематическое изложение знаний и формирование необходимых навыков учителями и учащимися.

6. Техническая литература – литература, которая относится к области техники и производства. Она включает каталоги изделий, инструкции по эксплуатации, обслуживанию и ремонту, а также патенты, содержащие информацию о технических решениях и разработках.

С конца XX века стали возникать новые носители и, соответственно, типы литературы. В дополнение к книжному (или как еще говорят физическому) типу появились аудиокниги и электронные книги.

Технологические достижения и разработка устройств для чтения, которые обеспечивают ощущения, аналогичные чтению настоящей книги, являются ключевыми факторами, влияющими на мировой рынок электронных книг. Ожидается, что такие преимущества, как растущее распространение смартфонов и многоязычных электронных книг, повысят глобальный спрос на электронные книги следующим образом.

1. Появление портативных устройств для чтения, таких как смартфоны и планшеты, станет основным драйвером роста мирового рынка электронных книг. Потребители предпочитают электронные книги физическим книгам из-за хрупкости бумажных экземпляров, необходимости надлежащего ухода и их высокой стоимости.

2. Такие системы, как цифровое образование и обучение с погружением, также будут способствовать росту рынка.

3. Растущее влияние технологий в сочетании с тенденцией оцифровки со стороны значительной части населения может оказать положительное влияние на рост рынка электронных книг.

Это также недорогая альтернатива традиционному методу распространения, обеспечивающая легкий доступ к различным библиотекам электронных книг через приложения или онлайн-сервисы. Более того, ресурсы, доступные для электронных книг, такие как электронная подписка, также способствовали росту рынка электронных книг.

На мировом рынке имеются прогнозы об увеличении рынка электронных книг. В частности, примерный рост рынка год за годом до 2028 года будет достигать около 4,78% [3].

Есть тенденции роста и на российском рынке. Во втором квартале 2023 года, по сравнению с тем же периодом годом ранее, рост составил 28% [4], а кварталом позднее рынок цифровых книг вырос на 31% в сравнении с аналогичным периодом 2022 года [5].

В целом рынок электронных книг на российских площадках за последний 2023 год вырос по сравнению с предыдущим 2022 годом на 19%. На это повлияли несколько причин: во-первых, увеличение рынка самиздата на 30% в сравнении с позапрошлым годом. Второй причиной роста рынка электронных книг можно назвать подписочные сервисы, объем выручки которых составил 23% от общего рынка электронных книг против 17% годом ранее. Такие показатели удалось получить и благодаря двум новым «игрокам» в сегменте электронных книг, а именно новый сервис «Строки» от оператора мобильной связи МТС, и сервис «Букмейт», который приобрела компания «Яндекс». Оба эти сервиса входят в их свои подписочные сервисы.

Сервис «Букмейт» за третий квартал 2023 года принес компании «Яндекс» выручку, в два раза превышающую оную за предыдущий квартал того же года. В сервисе «Ridero» отчитались об увеличение выручки за тот же период (3-й квартал 2023-го) более чем на 35% [6].

Вдобавок ко всему прочему на российском рынке появилось новое издательство, под названием «Дом историй». Данное издательство планирует заниматься выпуском в основном художественной литературы. Кроме этого,

«Дом историй» собирается издавать произведения зарубежных и российских авторов в разных форматах: как в бумажном, так и в электронном (обычные электронные книги и аудиокниги). Электронные книги планируется размещать на цифровых платформах и подписочных сервисах. Об этом уже есть договоренности с компаниями «ЛитРес» и «Яндекс» [7].

Одной из причин для роста российского рынка электронных книг называют подорожание физических экземпляров [5, 6]. Другой причиной может стать желание пользователей быстрее получить возможность воспользоваться новой покупкой, не теряя времени и средств [3].

При текущем росте рынка электронных книг и прогнозах роста в будущем предполагается как увеличение числа пользователей цифровых сервисов, так и прирост новых литературных произведений от разных авторов, которых может становиться больше с каждым годом.

Поэтому для увеличения приращения потока новых пользователей на свои сервисы следует внедрить или улучшить уже имеющуюся систему рекомендаций, как это сделано в музыкальных сервисах. Новая система рекомендаций должна быть основана на современных технологиях искусственного интеллекта, таких как нейронные сети и машинное обучение.

1.2. Сравнительный анализ аналогов

На рынке есть сервисы для обработки текста на естественных языках, которые могут находиться в экосистеме какой-либо компании.

Одной из таких является Amazon. У компании Amazon есть сервис под названием Amazon Comprehend.

Amazon Comprehend – сервис обработки естественного языка (NLP), который использует методы машинного обучения для поиска информации в тексте. Amazon Comprehend предоставляет API для распознавания сущностей, включая распознавание пользовательских сущностей, пользовательскую классификацию, извлечение ключевых слов, анализ настроений и многое другое. Это упрощает встраивание механизмов обработки естественного

языка в приложения. Для интеграции достаточно просто вызвать API сервиса Amazon Comprehend в приложении и предоставить информацию о местоположении исходного документа или текста. API будет выводить сущности, ключевые слова, настроения и язык в формате JSON, и эту информацию можно использовать в приложениях.

Amazon Comprehend имеет множество возможностей для работы с текстом, некоторые описаны ниже.

1. Идентификация сущности. Обнаружение объектов возвращает именованные сущности (люди, места, местоположения и т. д.), которые автоматически группируются по категориям на основе предоставленного текста.

2. Моделирование темы. Тематическое моделирование позволяет обнаружить популярные термины или темы в серии документов, хранящихся в Amazon S3. Оно определяет наиболее распространенные темы в массиве и организует их в группы, а затем связывает каждый документ с соответствующей темой.

3. Извлечение ключевых слов. Извлечение ключевых фраз возвращает ключевые слова или основные моменты, а также уровни достоверности для каждой ключевой фразы.

4. Анализ настроений. Анализ тональности возвращает общий эмоциональный тон текста (положительный, отрицательный, нейтральный или смешанный).

5. Обнаружение токсичности. Обнаружение токсичности Comprehend – простое решение для обработки естественного языка (NLP) для обнаружения оскорбительного содержимого в текстовых документах. Данная функция дает возможность модерировать одноранговые коммуникации на онлайн-платформах, а также генерировать входные и выходные данные искусственного интеллекта.

6. Обнаружение событий. Comprehend Events позволяет извлечь структуру событий из документа. Страницы текста разбиваются на легко

обрабатываемые данные, которые могут использоваться приложениями искусственного интеллекта или инструментами визуализации графиков. Оно помогает отвечать на вопросы «кто, что, когда, где» для больших наборов документов в любом масштабе и без предварительного опыта NLP. В основном Comprehend Events используется для извлечения подробной информации о фактических событиях и связанных с ними объектах в неструктурированном тексте.

7. Определение языка. Обнаружение языка автоматически обнаруживает текст, написанный на более чем 100 языках, и возвращает основной язык и уровень уверенности в том, что этот язык основной.

8. Синтаксический анализ. Syntax API в Amazon Comprehend позволяет анализировать текст с использованием токенизации и частей речи (PoS) для определения границ слов и тегов, таких как существительные и прилагательные, в тексте.

Amazon Comprehend может анализировать текст на немецком, английском, испанском, итальянском, португальском, французском, японском, корейском, арабском, китайском (упрощенном и традиционном) языках, а также на хинди, но не на русском. Хотя русский язык и поддерживается в функции определении языка, описанной ранее, однако данная возможность предоставляет просто обыкновенное определение языка текста, а не анализ этого самого текста. Для анализа текста на языке, не попавший в вышеописанный список, Amazon предлагает перевести текст с помощью Amazon Translate в один из поддерживаем языков для уже дальнейшего анализа [8].

Для классификации текстов лучше всего подходит функция моделирования темы. Однако для ее использования требуется наличие Amazon S3. Amazon Simple Storage Service (Amazon S3) – облачное объектное хранилище, предоставляемое возможность хранения и получения любого объема данных в любое время из любой точки сети. Поэтому, чтобы использовать данную функцию потребуются дополнительные расходы на облачное хранилище помимо затрат на доступ к Amazon Comprehend.

Кроме моделирования тем Amazon Comprehend имеет много других полезных функций, которыми можно воспользоваться при наличии доступа к услуге. Тем не менее, требование наличие другого сервиса Amazon для доступа к Comprehend и отсутствие поддержки анализа русского языка делает Amazon Comprehend не очень подходящей услугой.

Другой представитель анализа естественных языков – IBM Watson Natural Language Understanding. Он использует глубокое обучение для извлечения значения и метаданных из неструктурированных текстовых данных. Позволяет изучить данные с помощью текстовой аналитики для извлечения категорий, классификаций, объектов, ключевых слов, настроений, эмоций, отношений и синтаксиса.

Далее приведены некоторые функции, доступные в IBM Watson Natural Language Understanding [9].

1. Возможность научить Watson понимать язык определенного бизнеса и извлекать персонализированную информацию с помощью Watson Knowledge Studio.
2. Возможность получать полезную информацию в режиме реального времени, чтобы предоставить сотрудникам инструменты, необходимые для извлечения метаданных и шаблонов из огромных массивов данных.
3. Возможность развернуть Watson Natural Language Understanding за брандмауэром или в любом облаке.
4. Обнаружение людей, места, события и другие типы объектов, упомянутых в контенте, используя готовые возможности.
5. Классификация текста с помощью пользовательских меток, чтобы автоматизировать рабочие процессы, извлекать ценную информацию и улучшать поиск и обнаружение.
6. Извлечение эмоций (радость, гнев, печаль, страх и другие чувства), передаваемые конкретными целевыми фразами или всем документом в целом.

Несомненным преимуществом IBM Watson Natural Language Understanding является то, что он имеет бесплатную демоверсию, которая, в свою очередь, находится на сайте, ее можно использовать сразу и проверить, как она работает (рисунки 1, 2).

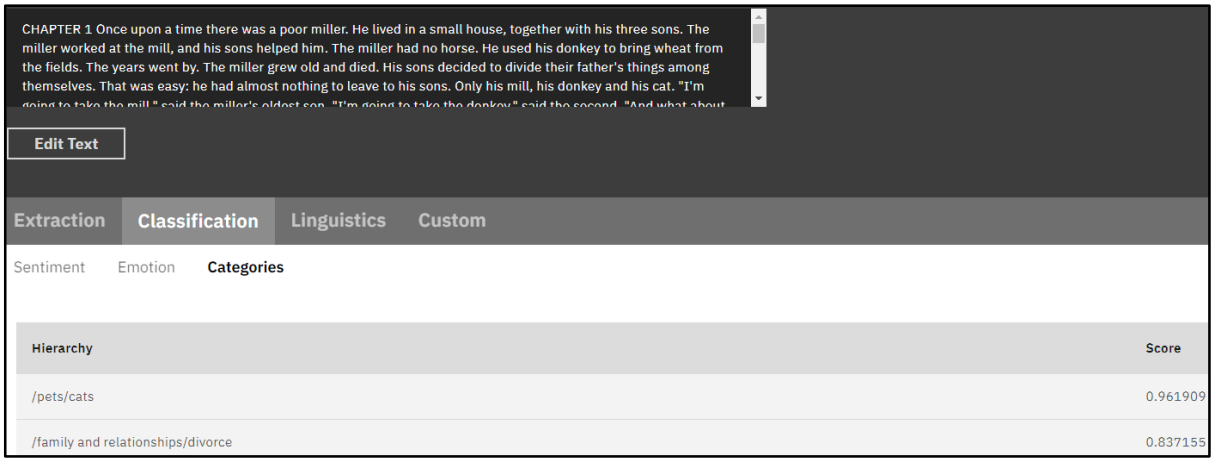

Рисунок 1 – Результат классификации

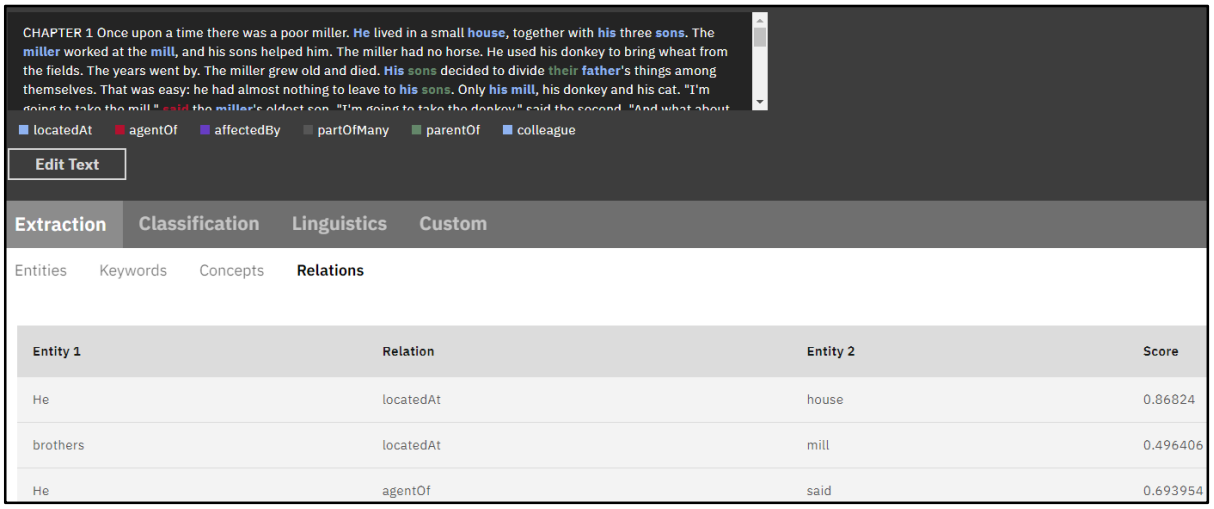

Рисунок 2 – Результат извлечения связей

Однако данный сервис имеет ограниченный функционал для работы с русским языком, что продемонстрировано на рисунках 3 и 4. Для проверки была взята книга из жанра фэнтези.

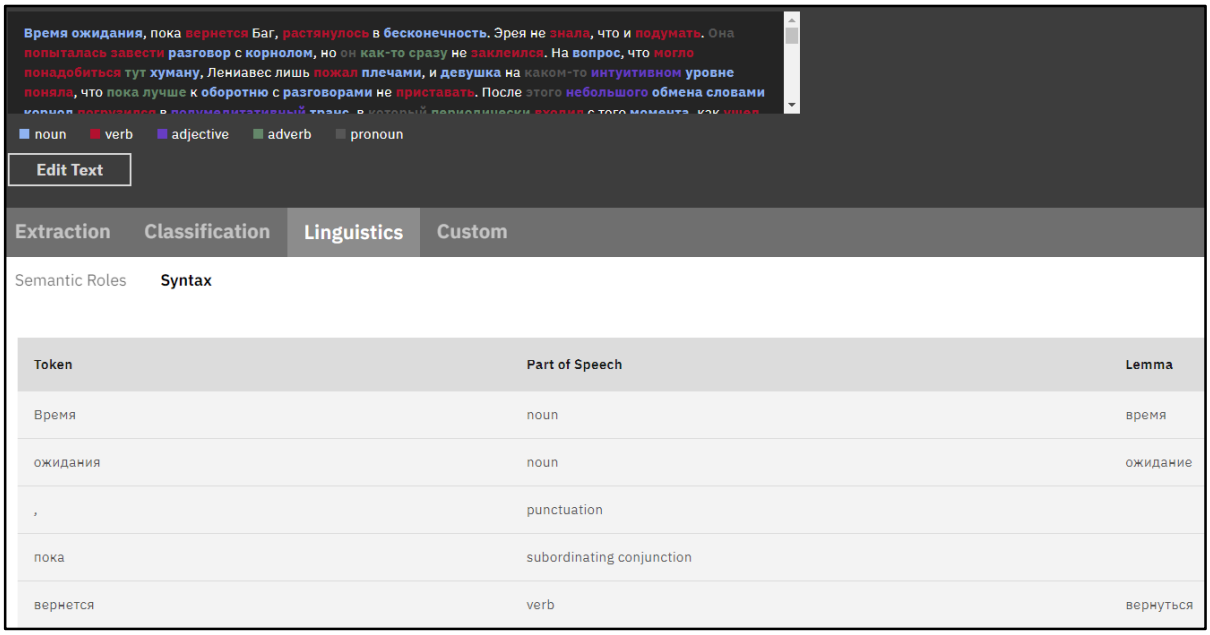

Рисунок 3 – Результат синтаксического анализа с русским текстом

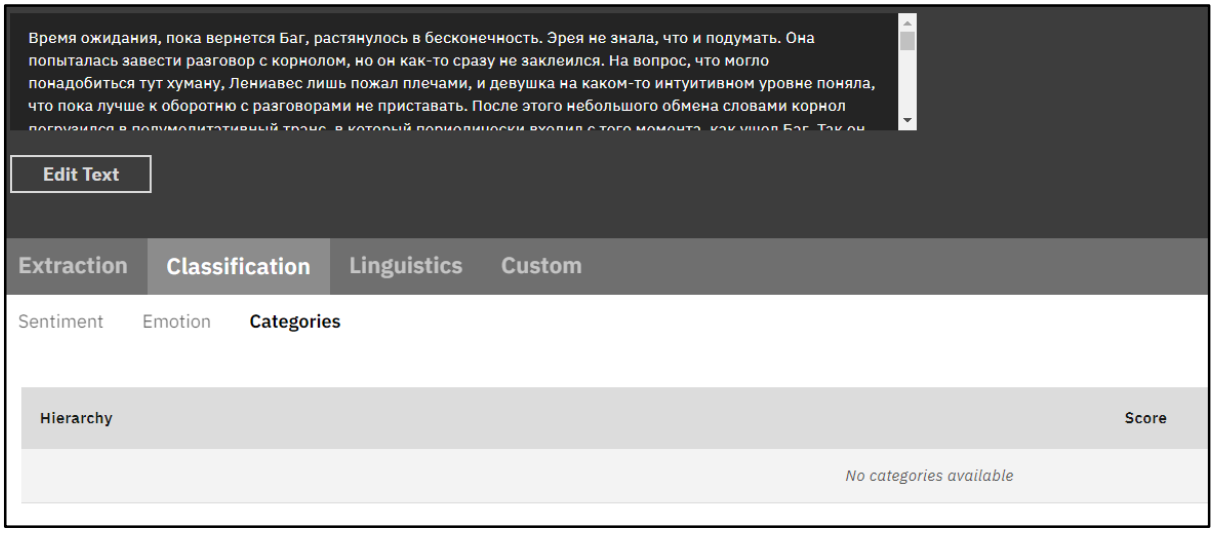

Рисунок 4 – Результат классификации с русским текстом

Из демоверсии видно, что классификация текста на русском языке не поддерживается. Для убедительности была взята другая книга с русским текстом. Как видно на рисунках 5 и 6, работоспособность с русским языком ограничена, так же, как и с первой книгой.

Рисунок 5 – Результат классификации с русским текстом из книги 2

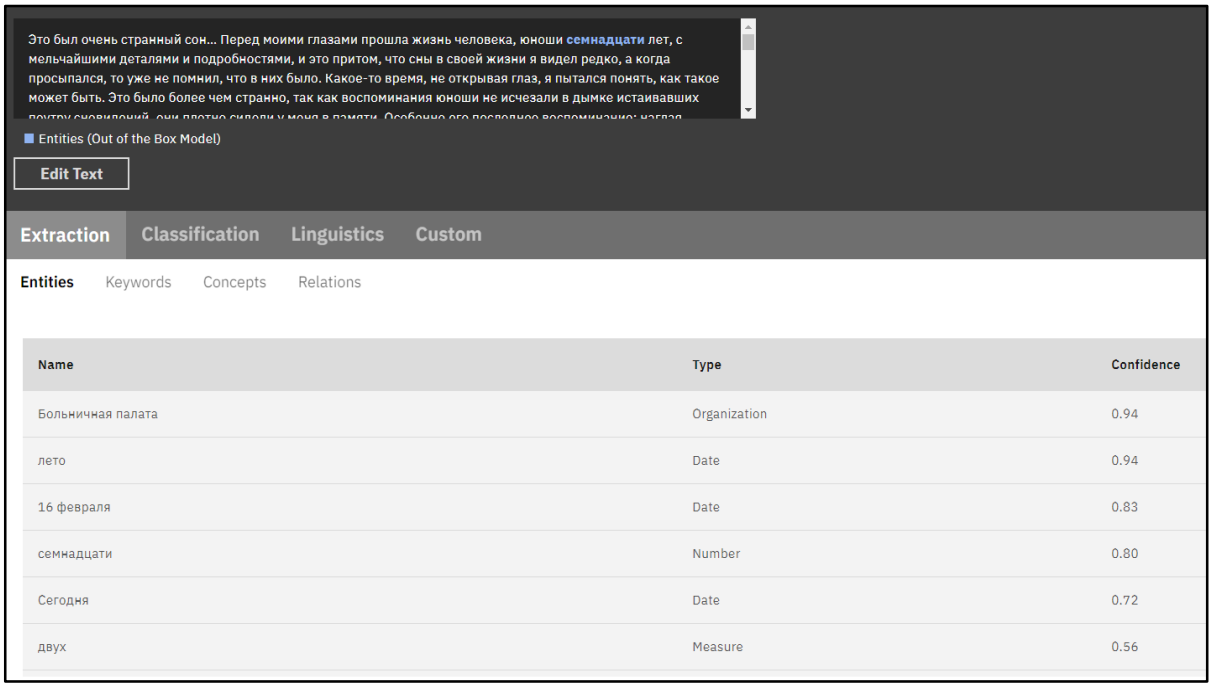

Рисунок 6 – Результат извлечения сущностей с русским текстом из книги 2

Подводя итог, можно сказать, что поддержка русского языка данным сервисом это преимущество в сравнении с Amazon Comprehend, но функция классификации русского языка не доступна.

Следующий сервис по обработке и анализу естественного языка может предоставлять известная компания Google. Этот сервис называется Natural Language AI. Данное решение имеет следующие «ключевые функции».

1. AutoML. Позволяет обучать пользовательские высококачественные модели машинного обучения классификации, извлечению и обнаружению настроений с минимальными усилиями и опытом машинного обучения с помощью Vertex AI для естественного языка на базе AutoML. Можно использовать пользовательский интерфейс AutoML для загрузки данных обучения и тестирования пользовательских моделей без кода.

2. Natural Language API. Готовые шаблоны Natural Language API позволяют применять понимание естественного языка к приложениям с помощью таких функций, как анализ тональности, анализ сущностей, анализ тональности сущностей, классификация контента и синтаксический анализ. Все эти функции поддерживают несколько языков.

3. Healthcare Natural Language AI. Предоставляет анализ концепций, хранящихся в неструктурированном медицинском тексте, в режиме реального времени. Healthcare Natural Language AI позволяет извлекать машиночитаемые концепции здравоохранения из медицинских документов, а AutoML Entity Extraction for Healthcare позволяет создавать модели извлечения знаний для приложений в сфере здравоохранения и медико-биологических наук [10].

Однако эти функции – разные решения, которые могут быть использованы отдельно и независимо друг от друга. Так, например, AutoML это решение, которое позволяет обучать пользовательские модели для машинного обучения и нейронных сетей, что и было ранее сказано. Natural Language API же это сервис для использования уже обученных и готовых компанией Google моделей, что также написано выше.

Что касается Healthcare Natural Language AI, то данная функция схожа с Natural Language AI тем, что имеет обученные модели естественного языка,

но используются они только для области здравоохранения. Однако, поскольку Healthcare Natural Language AI имеет специфичную направленность, она входит в другой сервис под названием Cloud Healthcare API [11].

Возвращаясь к Natural Language AI, встает выбор между использованием AutoML и Natural Language API. На официальной основной странице сайта сервиса от Google имеется таблица сравнений функций данных решений (таблица 1).

Таблица 1 – Сравнение решений

Функции	**AutoML**	**Natural Language API**
Поддержка REST API	+	+
Синтаксический анализ	–	+
Анализ сущностей	–	+
Извлечение пользовательских сущностей	+	–
Анализ настроений	–	+
Пользовательский анализ настроений	+	–
Классификация контента	–	+
Пользовательская классификация контента	+	–
Поддержка нескольких языков	+	+
Пользовательские модели	+	–
Использование AutoML от Google	+	–
Понимание пространственной структуры	+	–
Поддержка больших наборов данных	+	–

Исходя из наличия тех или иных функций у обозреваемых решений, подходящим вариантом является сервис Natural Language API. Несмотря на наличие большего количества функций у AutoML, Natural Language API привлекателен тем, что имеет уже обученные языковые модели, включая требуемую для данной работы функцию классификации контента.

Компания Google, решая не отставать от других участников рынка, сделала свою реализацию демонстрационной версии, как и конкурент в лице компании IBM со своим сервисом Watson Natural Language Understanding, разобранным ранее. Демонстрационная версия предоставляет функции исключительно Natural Language API [12]. В отличии от IBM, в компании Google было принято решение разместить ее на начальной странице сервиса

для более быстрого и удобного способа взаимодействия. Хотя у демонстрационной версии сервиса от компании Google есть те же типы анализа текста, что и у Watson Natural Language Understanding. Однако, хоть сервис Natural Language API и имеет те же виды работы с текстом, они не такие подробные как у решения от компании IBM.

Далее на рисунках 7 и 8 продемонстрирована работа данной демоверсии. В частности, интересующей функцией является функция классификации. Она находится во вкладке под названием Categories, и ее работа представлена на рисунке 8. В дополнение к вышеописанному, данная вкладка предоставляет пользователю ссылку на просмотр всех доступных у Google категорий.

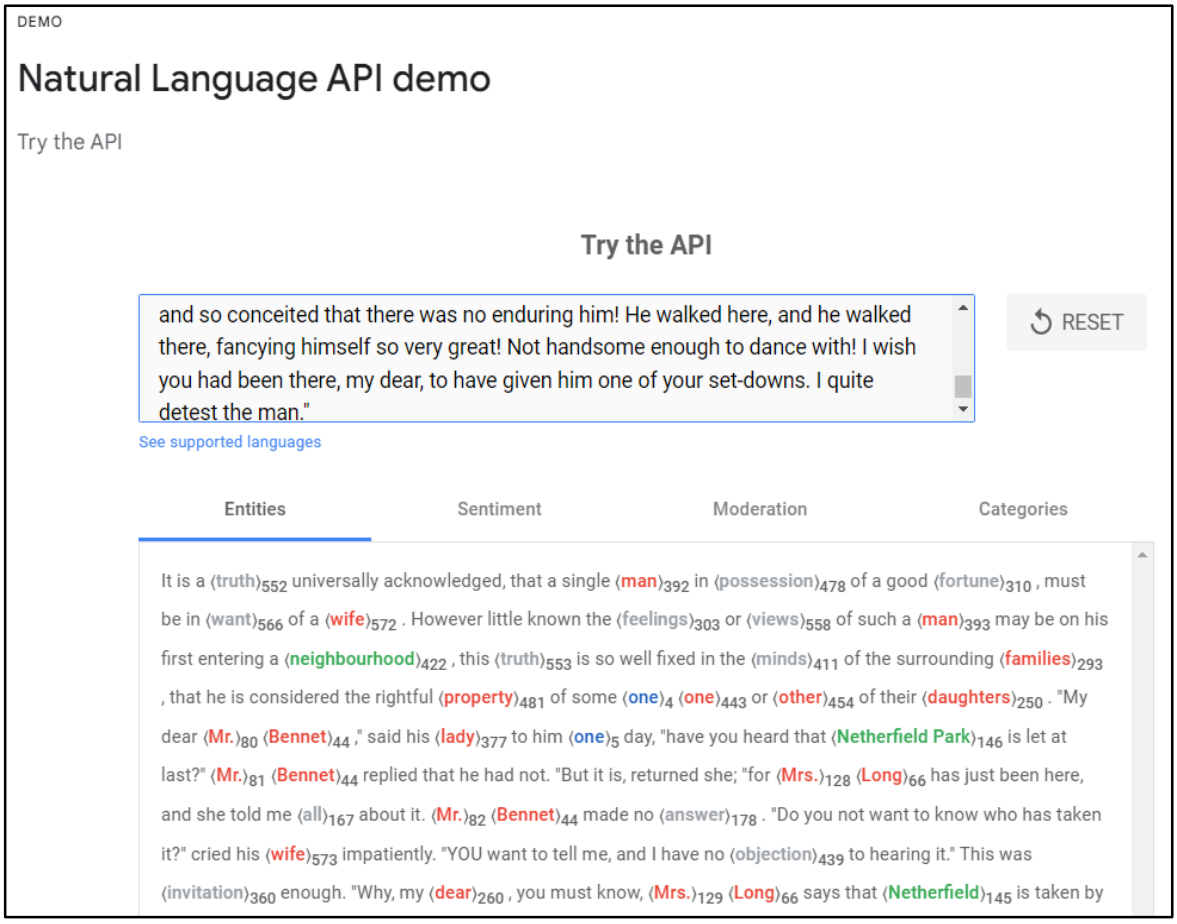

Рисунок 7 – Работа демонстрационной версии

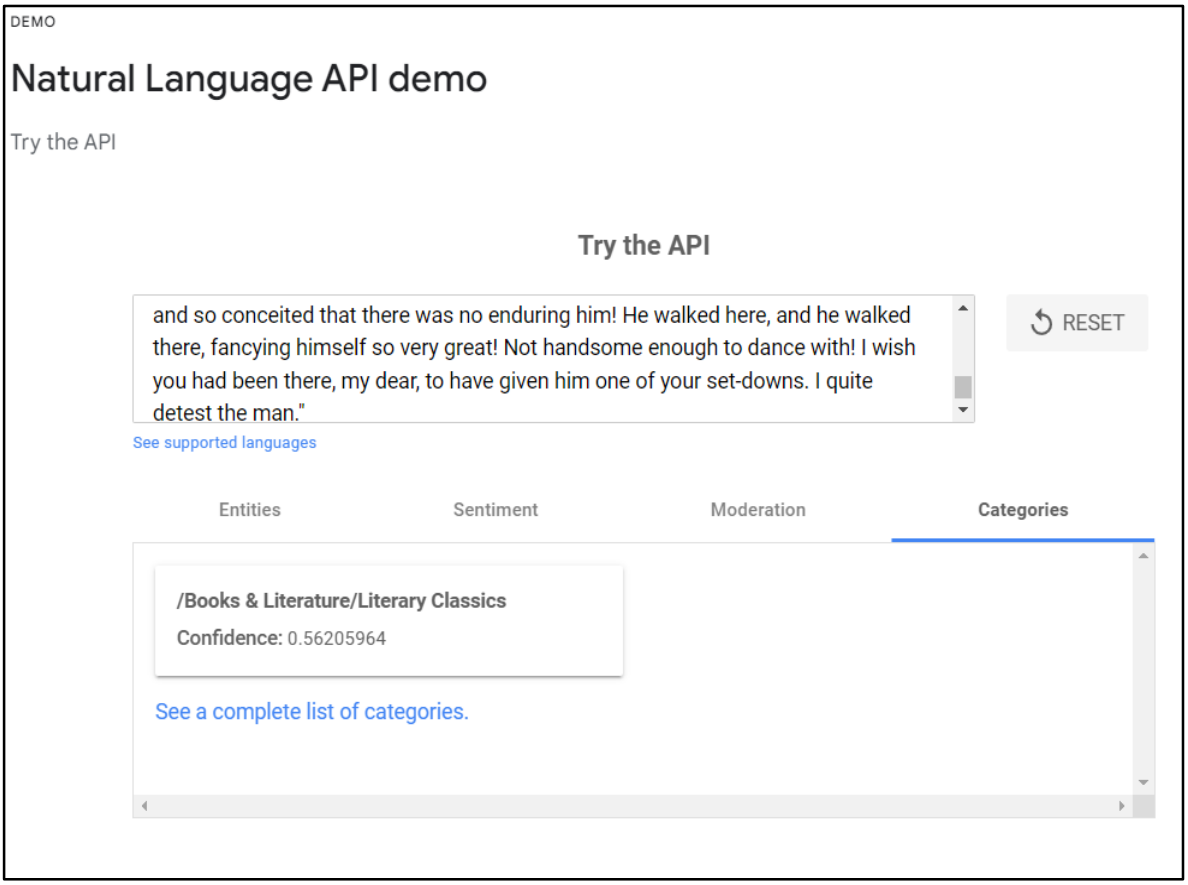

Рисунок 8 – Классификация в демоверсии

В отличии от двух предыдущих сервисов, в решении от компании Google официально заявлена поддержка анализа русского языка, помимо всех остальных иностранных языков. Хоть русский язык и не поддерживается в некоторых функция, он доступен в большинстве всех функций данного сервиса, включая и классификацию [13]. Это является одним из главных преимуществ в сравнении с другими решениями.

Работа с русским языком показана на рисунках 9 и 10. Для проверки работы данного решения в анализе русского языка в целом, и классификации текстов на русском языке в частности, был взят отрывок из электронной книги.

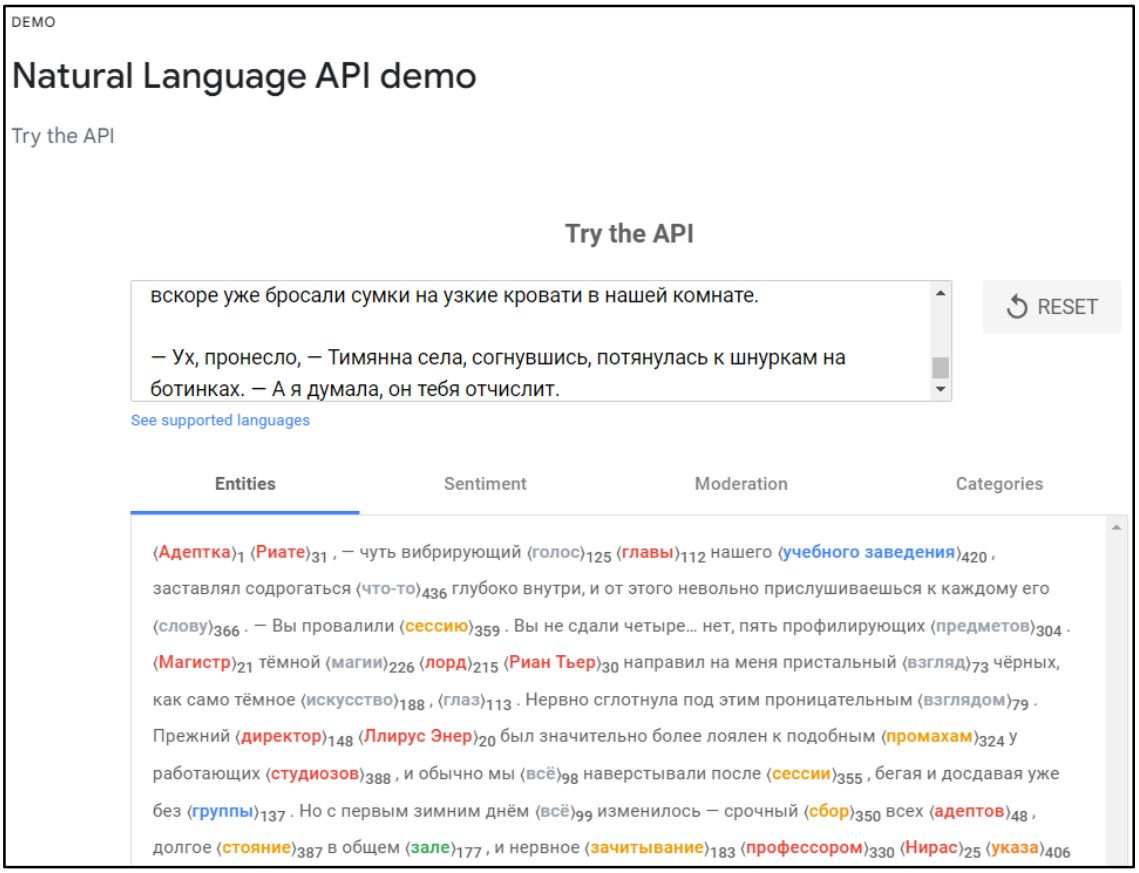

Рисунок 9 – Демоверсия с русским языком

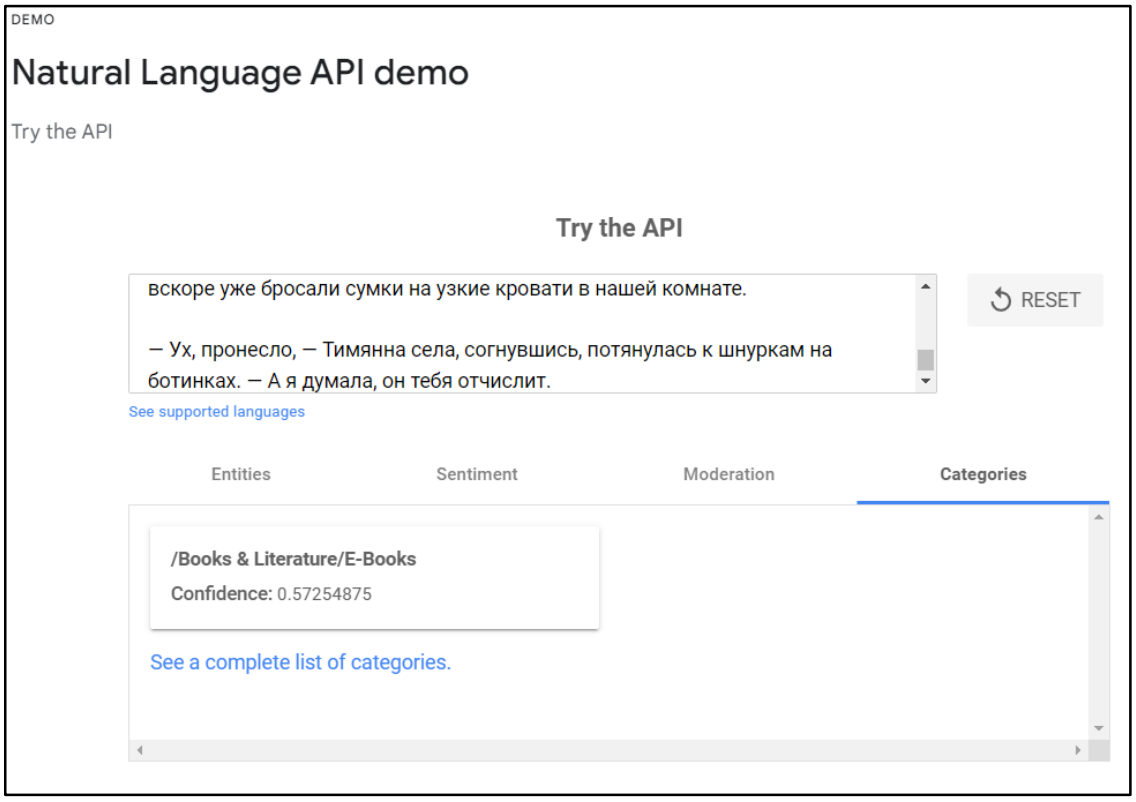

Рисунок 10 – Классификация русского языка

Natural Language API правильно определил, что это литература, и в виде электронной книги. Однако, если посмотреть список всех категорий, которые могут быть определены данным сервисом, то можно заметить довольно скудную классификацию книг (рисунок 11).

```
/Beauty & Fitness/Weight Loss
/Books & Literature/Audiobooks
/Books & Literature/Book Retailers
/Books & Literature/Children's Literature
/Books & Literature/E-Books
/Books & Literature/Fan Fiction
/Books & Literature/Literary Classics
/Books & Literature/Poetry
/Books & Literature/Writers Resources
/Books & Literature/Other
/Business & Industrial/Other
```

Рисунок 11 – Поддерживаемые категории книг

Этот недочет не позволяет рассматривать данное решение для использования классификации литературных произведений, то есть распознавания их по жанрам.

Другой игрок на рынке предоставления услуг с машинным обучением – компания Microsoft. Ее сервис называется ИИ Azure. ИИ Azure представляет собой набор разных служб, перечисленных ниже.

1. Служба Azure OpenAI. Позволяет создавать собственные приложения для помощника и генеративного ИИ с использованием новейших моделей.

2. Поиск с использованием ИИ Azure. С его помощью можно получать наиболее релевантные данные с помощью поиска по ключевым словам, векторного и гибридного поиска.

3. Безопасность содержимого ИИ Azure. Дает возможность отслеживать текст и изображения для обнаружения оскорбительного или неприемлемого контента.

4. Переводчик ИИ Azure. Переводит документы и текст в реальном времени на более чем 100 языков.

5. Речь ИИ Azure. Используется для преобразования речи в текст, преобразование текста в речь, перевод речи и распознавание говорящего.

6. Зрение ИИ Azure. Читает текст, анализируйте изображения и распознавайте лица с помощью оптического распознавания символов (OCR) и машинного обучения.

7. Язык ИИ Azure. Позволяет обобщать документы и анализировать текст с помощью предварительно созданных функций на базе искусственного интеллекта.

8. Интеллектуальный анализ документов ИИ Azure. Применяет расширенное машинное обучение для извлечения текста, пар «ключ-значение», таблиц и структур из документов [14].

Из всех этих служб данной работе может подойти седьмая (язык ИИ Azure) служба.

Язык ИИ Azure – это управляемая служба для разработки приложений обработки естественного языка. Определяет ключевые термины и фразы, анализирует тональность, обобщает текст. Ее можно использовать для добавления заметок, обучения, оценки и развертывания настраиваемых моделей ИИ. Данная служба имеет следующие функции [15].

1. Определение языка. Распознавание языка – предварительно настроенная функция, которая может определять язык, на котором написан документ, и возвращать код языка для широкого спектра языков, вариантов, диалектов и некоторых региональных или культурных языков.

2. Анализ тональности и интеллектуальный анализ мнений. Анализ тональности и интеллектуальный анализ мнений – предварительно настроенные функции, которые помогают узнать, что люди думают о торговой марке или теме, используя интеллектуальный анализ текста, чтобы получить подсказки о положительных или отрицательных тональности, и связать их с конкретными моментами из текста.

3. Извлечение ключевой фразы. Извлечение ключевых фраз – предварительно настроенная функция, которая оценивает и возвращает основные концепции в неструктурированном тексте и возвращает их в виде списка.

4. Анализ текста для сферы здравоохранения. Анализ текста для здравоохранения – предварительно настроенная функция, которая извлекает и помечает соответствующую медицинскую информацию из неструктурированных текстов, таких как заметки врача, выписки, клинические документы и электронные медицинские записи.

5. Распознавание именованных сущностей (NER). Распознавание именованных сущностей – предварительно настроенная функция, которая классифицирует сущности (слова или фразы) в неструктурированном тексте по нескольким предопределенным группам категорий, например, люди, события, места, даты и другое.

6. Пользовательская классификация текстов. Настраиваемая классификация текста позволяет создавать пользовательские модели ИИ для классификации неструктурированных текстовых документов в определяемые пользовательские классы.

7. Ответы на вопросы. Ответы на вопросы это пользовательская функция, которая находит наиболее подходящий ответ на входные данные от пользователей.

Поддержка русского языка есть в половине функций данного решения. В основном есть английский, французский, немецкий, итальянский, японский, испанский, португальский языки.

У Microsoft есть демоверсии некоторых функций своего сервиса. Однако они показывают только заготовленные варианты предпоказа, что не позволяет пользователям использовать свои данные для просмотра работы. Это является минусом в сравнении с последними двумя сервисами.

Однако в этой службе отсутствуют какие-либо готовые модели для решения классификации текстовых документов.

Далее в таблице 2 приведено сравнение всех перечисленных сервисов, предоставляющих услуги с ИИ возможностями.

Таблица 2 – Сравнение сервисов

Функции	**Amazon Comprehend**	**IBM Watson Natural Language Understanding**	**Natural Language AI**	**ИИ Azure**
Классификация текстовых документов	+	+	+	–
Выделение сущностей	+	+	+	+
Поддержка русского языка	–	+/–	+	+/–
Наличие демоверсии	–	+	+	+/–
Выделение ключевых фраз	+	+	+	+
Анализ тональностей	+	+	+	+
Синтаксический анализ	–	+	+	+
Количество дополнительных возможностей (другие функции)	2	5	6	11

Из данных сервисов лучшим оказался Natural Language AI от Google. У него есть почти все необходимое: классификация текстовых документов, поддержка анализа русского языка, наличие демонстрационной версии, и количество дополнительных функций не самое маленькое из рассматриваемых сервисов. У него даже получилось распознать жанр книги, однако количество распознаваемых книжных жанров довольно маленькое и не позволяет использовать его на рынке электронных книг.

По полученным результатам следует вывод, что для классификации книг по жанрам нужно разработать свою программу.

1.3. Обзор научной литературы

В основном работы по рассматриваемой теме посвящены классификации книг на английском языке. В работах [16, 17] рассматривается классификация книг по жанрам исключительно на основе изображений книжных обложек без использования текстов произведений, аннотаций и без указания авторов и названий. В [16] наилучшие результаты показала комбинация

предобученной сверточной нейронной сети с методами обработки естественного языка (Stanford NLP Classifier [18] и word2vec [19]), а в [17] – комбинация предобученной сверточной нейросети и рекуррентной LSTM-сети. В обоих случаях сверточные сети распознают само изображение, а методы обработки естественного языка и рекуррентная сеть используются для распознавания текстов на книжных обложках.

В [20] рассматривается классификация книг по жанрам непосредственно по текстовому содержимому. В данной работе перебираются разные варианты представлений входных данных. Наряду с входными данными исследуются различные модели для классификации именно текстовых данных, так как некоторые модели либо использовались для других задач и не использовались для текста вообще, либо считались лучше для работ, не связанных с текстом. Новые модели были взяты для проверки их работы с текстом при условии разных входных данных. По приведенным в работе итогам, наилучшие результаты показаны сверточной нейронной сетью.

Классификации книг на русском языке посвящена работа [21], в которой в качестве исходных данных выступают полные тексты произведений. В ней также используется сверточная нейронная сеть. В дополнение к ней задействуется дополнительная вспомогательная модель word2vec.

При этом в работах [20] и [21] за основу используемых нейросетей берутся сети, рассмотренные в исследовании, посвященному разбору методов классификации текстов [22].

Следующая работа также рассматривает классификацию текстов естественного языка на русском языке, а именно литературные произведения [23]. В ней использовалась глубокая нейронная сеть, состоявшая из разных видов слоев. Однако, в отличии от предыдущей работы [21], в этой на вход подавались не исходные тексты произведений, а их текстовое описание.

В другой научной литературе [24] используются разные методы машинного обучения для классификации научных работ на русском языке по специальностям и сферам специальностей. Там используются небольшие

отрывки текстов из научных статей. Используется стандартная предобработка текстовых данных для дальнейшего машинного обучения. В качестве методов машинного обучения было взято и протестировано 6 алгоритмов.

Выводы по первой главе

В первой главе было описано несколько аспектов.

Первый аспект – предметная область. В предметной области дано определение «что такое литература» и какие ее виды бывают. Так же был рассмотрен книжный рынок, а именно рынок электронных книг: прогнозы роста рынка, определение «игроков», предоставляющие услуги, связанные с электронными книгами, ожидание роста количества пользователей.

Во втором аспекте проанализированы существующие сервисы по работе с текстовыми данными. Были разобраны их функции, возможности и в конце было проведено их сравнение.

Третий и последний аспект представляет разбор научных работ по данной теме. Какие методы использовались, с какими данными и как проходила обработка этих данных.

2. МАШИННОЕ ОБУЧЕНИЕ

2.1. Алгоритмы машинного обучения

Машинное обучение – один из разделов искусственного интеллекта. Алгоритмы машинного обучения дают возможность компьютерам приходить к решению на основе данных, не используя прямые методы решения задач. Алгоритмы во время обучения тренируются поиску определенных связей и закономерностей на основе наборах данных для последующего анализа данных для определенной задачи какой-либо сферы деятельности.

Машинное обучение подразделяется на два вида моделей машинного обучения и подразумевает собой использование определенных алгоритмов на наборе данных. В зависимости от этого набора данных и специфики определенной задачи используется один из двух видов: обучение с учителем (Supervised learning) и обучение без учителя (Unsupervised learning). Для каждого из этих видов можно применять как один, так и несколько алгоритмических методов. В основном алгоритмы машинного обучения используются для следующих типов задач: классификация, кластеризация, прогнозирование, обнаружение аномалий, регрессия, понижение размерности.

Обучение с учителем – один из видов машинного обучения. Данный вид используется чаще всего [25]. В алгоритмах обучения с учителем обучения происходит на примерах. В эти алгоритмы набор данных представляет собой пары входных и выходных данных, где выходные данные помечены нужным значением. Входные данные это то, что обрабатывается алгоритмами машинного обучения, а выходные – то, что должно получиться для каждого входа. С течением времени система с алгоритмом машинного обучения начинает определять корреляцию сходств и различий между данными [26]. Обычно в данном виде используются задачи классификации, регрессии, прогнозирования.

Второй вид машинного обучения – обучения без учителя. В этом виде обучения подразумевается отсутствие правильного ответа, то есть из пары входных и выходных данных, в наборы данных остаются только входные,

выходных нет. Алгоритмы данного вида обрабатывают входные данные без пометок, так же со временем выявляя корреляцию [26]. В задачи обучения без учителя можно отнести кластеризацию, понижение размерности, обнаружение аномалий.

Данная работа о классификации книг по жанрам, что соответствует задаче классификации. Как было ранее описано, задача классификации относится к виду машинного обучения с учителем, соответственно, в наборе данных для каждой книги в качестве входных данных будет предусмотрена метка, представляющая собой жанр книги. Однако, жанров для книг больше чем два, поэтому в работе будет рассматриваться задача многоклассовой классификации, но в целом это все одна задача классификации.

2.2. Сбор и обработка данных

Для использования алгоритмов машинного обучения обязательно требуется набор данных. Для классификации книг на русском языке не нашлось никаких наборов данных, поэтому было принято решения сделать свой набор данных.

Для набора данных сначала требовалось найти сайт, предоставляющий возможность скачать электронные книги на компьютер/телефон для дальнейшего чтения без сети Интернет.

После того, как, будут найдены сайты, следует выбрать один из них для реализации вспомогательной программы сбора данных с сайтов, поскольку собирать большой набор данных вручную это очень долго. Обычно такие программы известны под названием «парсер».

Сначала следовало определить, какие жанры книг самые распространенные. Для этого было посещено несколько сайтов, занимающиеся электронными книгами. В результате поиска в интернете была найдена страница, обозревающая, какие сайты и сервисы работают в сфере электронных книг [27]. В каждом из них присутствовало меню «жанры», где можно было посмотреть, какие жанры использует конкретный сайт. Жанры с каждого

посещенного сайта были переписаны в табличный процессор Excel по столбцам, где в названиях столбцов находятся названия сайтов, а ниже – используемы этим сайтом жанры.

После переписывания и анализа наличия жанров на тех или иных сайтах были определены следующие жанры:

1) бизнес;
2) боевики;
3) детективы;
4) дом/дача;
5) знания/навыки;
6) история;
7) классика;
8) комедии;
9) компьютерная;
10) культура/искусство;
11) поэзия;
12) приключения;
13) психология/мотивация;
14) религия;
15) романы;
16) спорт/здоровье/красота;
17) ужасы/мистика;
18) фантастика;
19) фэнтези;
20) хобби/досуг.

После определения жанров последовал поиск сайтов, предоставляющих возможность бесплатного скачивания электронных книг. В итоге были найдены два сайта [28, 29].

Далее последовала реализация вспомогательного парсера. При работе парсера на первом сайте [28] возникали проблемы как при скачивании электронных книг, так и к доступу страницы скачивания.

Поэтому парсер в следствие вышеописанных проблем был переписан на второй сайт [29]. В листинге 1 представлена реализация парсера.

Листинг 1 – Функция сбора книг с сайта

```
def downloading_flib(genre_url:str, klass: str, pages: int = 60) -> None:
    main_url = "https://flibusta.one"
    for i in range(1, pages + 1):
        time.sleep(0.1)
        if not os.path.exists(f"booky/{klass}/"):
            os.mkdir(f"booky/{klass}/")
        r = requests.get(genre_url + f"?page={i}", headers=heads)
        print(genre_url+f"?page={i}", r.status_code)
        soup = BeautifulSoup(r.text, "html.parser")
        books = soup.find_all("a", attrs={"class": "th-in with-mask"})
        for book in books:
            print(main_url + book.get("href"), i)
            time.sleep(0.1)
            try:
                book_page = requests.get(main_url + book.get("href"), head-
ers=heads)
            except requests.exceptions.TooManyRedirects:
                continue
            sup = BeautifulSoup(book_page.text, "html.parser")
            lnk1 = sup.find("span", class_="link pdf")
            try:
                lnk = lnk1.get("data-link")
            except AttributeError:
                continue
            print(lnk + "         fglbkflndb")
            time.sleep(0.1)
            file = tempfile.TemporaryFile()
            fzip = zipfile.ZipFile("1.zip")
            if not os.path.exists(f"booky/{klass}/{klass}_books"):
                os.mkdir(f"booky/{klass}/{klass}_books")
            try:
                while True:
                    try:
                        download_zip = requests.get(lnk, headers=heads)
                    except requests.exceptions.ConnectionError:
                        print("Waiting 2 minutes...")
                        time.sleep(120)
                        continue
                    else:
                        break
                file.write(download_zip.content)
                fzip = zipfile.ZipFile(file)
                fzip.extractall(path=f"booky/{klass}/{klass}_books")
            except zipfile.BadZipFile:
                continue
            finally:
                file.close()
                fzip.close()
```

При сборе данных пришлось отказаться от некоторых жанров, так как после исследования второго сайта было обнаружено, что количество книг этих жанров недостаточно - не набиралось и 100 наименований. В итоге жанров осталось 16, были исключены: комедии, компьютерная, культура/искусство и приключения. В общей сложности было скачано 13 228 электронных книг.

Книги с данного сайта представляют собой пробные версии, в следствии этого была написана функция, определяющая минимальное и максимальное количество страниц в этих pdf-фалах (листинг 2).

Листинг 2 – Функция определения количества страниц

```
def scanning(path, expected_list):
    mini, maxi = 100, 0
    for genre_folder in os.listdir(path):
        genre_path = os.path.join(path, genre_folder)
        if os.path.isdir(genre_path):
            for file_name in os.listdir(genre_path):
                file_path = os.path.join(genre_path, file_name)
                if file_path.endswith(".pdf"):
                    with open(file_path, "rb") as filehandle:
                        try:
                            pdf = PdfReader(filehandle)
                        except Exception:
                            continue
                        pages = len(pdf.pages)
                        if pages < mini:
                            mini = pages
                        if pages > maxi:
                            maxi = pages
    expected_list.append(mini)
    expected_list.append(maxi)
    expected_list.append((mini+maxi)/2)
```

Было определено, что минимальное количество страниц из электронных книг равно пяти, а максимальное – 905. Поэтому было принято решение использовать максимум 100 страниц из каждой книги. В листинге 3 представлена функция считывания текста из файлов.

Листинг 3 – Функция извлечения текста из pdf

```
def scanning_text(path, expected_list):
    ident = 0
    for genre_folder in os.listdir(path):
        genre_path = os.path.join(path, genre_folder)
        if os.path.isdir(genre_path):
            for file_name in os.listdir(genre_path):
                file_path = os.path.join(genre_path, file_name)
                if file_path.endswith(".pdf"):
```

```
                    print(genre_folder)
                    ident = ident + 1
                                print(f"a number of a BOOK IS {ident}
**********************************")
                    with open(file_path, "rb") as filehandle:
                        try:
                            pdf = PdfReader(filehandle)
                        except Exception:
                            continue
                        pages = len(pdf.pages)
                        pdfs = ""
                        if pages > 100:
                            pages = 100
                        for i in range(pages):
                            page = pdf.pages[i]
                            pdfs += page.extract_text().strip()
                        expected_list.append([pdfs, genre_folder])
```

Данная функция собирала текст из каждой книги, название папки книги как жанр и сохраняла в списке в виде «[книга, жанр]».

После формирования набора данных была проведена базовая отчистка текста от лишних символов. Далее для упрощения работы набор данных был переведен из списка в DataFrame из библиотеки pandas.

Далее для работы с алгоритмами машинного обучения был использован пакет sklearn. Данный пакет предоставляет не только разные алгоритмы машинного обучения, но и некоторую работу с набором данных.

Для работы с текстовыми данными были из данного пакета были использованы `TfidfVectorizer` и `train_test_split`. В качестве вспомогательного аспекта для работы с текстом была так же использована библиотека NLTK (листинг 4).

Листинг 4 – Подготовка текстового набора данных

```
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from nltk.corpus import stopwords

nltk.download('stopwords')
stops = nltk.corpus.stopwords.words('russian')
X_train, X_test, y_train, y_test = train_test_split(df['text'], df['label'],
test_size=0.2, random_state=42, stratify=df['label'])
vectorizer = TfidfVectorizer(stop_words=stops, min_df=50, max_df=13_000)
```

Функция `train_test_split` разделяет набор данных на тренировочную и тестовую выборки. Класс `TfidfVectorizer` используется для преобразования текстовых данных в числовой формат, так как машинное обучение, как и компьютеры в целом, работают в основном ч числовыми данными. В параметры класса передаются следующее: `min_dif` – минимальное количество документов, в которых встречаются слова; `max_dif` – аналогично `min_dif`, только указывает верхний порог; `stop_words` – «стоп-слов», то есть те слова, которые часто используются и не несут в себе никакой смысловой нагрузки для классификации машинного обучения. В пакете sklearn есть свой набор «стоп-слов», однако он только для английского языка. Однако, с этим может помочь другая библиотека. Библиотека NLTK используется для работы с естественным языком. В данной работе как раз она и предоставляет набор «стоп-слов» русского языка.

2.3. Реализация и тестирование модели машинного обучения

Для реализации использован пакет sklearn. Было проведено некоторое количество экспериментов, в ходе которых варьировались разные методы машинного обучения. Среди них были такие, как MultinomialNB, ComplementNB, SVC, LinearRegression, LogisticRegression, Ridge, RidgeClassifier, SGDClassifier, SGDRegressor, RandomForestClassifier и другие.

Для обучения моделей было использовано 80% всего набора данных, разделенного с помощью функции `train_test_split` из библиотеки sklearn.

В результате экспериментов и с учетом ограничений ресурсов для реализации машинного обучения были найдены методы, точность классификации которых превышает 50%. Этих методов 7: ComplementNB, MultinomialNB, SVC, LogisticRegression, SGDClassifier, RidgeClassifier и RandomForestClassifier.

Для тестирования использовалась оставшаяся доля всех текстов электронных книг в 20%. Данная доля не использовалась в обучении, поэтому

она хорошо подходит для определения точностей моделей и их дальнейшего анализа.

Далее для всех оставшихся методов были построены матрицы несоответствий (confusion matrix) и выведены отчеты классификаций (classification report) для дальнейшего отбора самой лучшей модели из представленных.

Точность модели машинного обучения для классификации основывается, в основном, на трех показателях: аккуратность (accuracy), точность (precision) и полнота (recall). Однако расчет метрик точности модели машинного обучения для многоклассовой классификации более сложен, чем для бинарной классификации. Для многоклассовой классификации есть 2 дополнительные метрики: микроусреднение и макроусреднение. Максроусреднение считается более качественной метрикой, чем микроусреднение [30]. Параметр макроусреднения доступен в отчете классификаций.

Поэтому в качестве основных метрик были взяты аккуратность и макроусреднение. Аккуратность было поставлено в дополнение к макроусреднению потому, что данная метрика используется самим пакетом sklearn для точности модели классификации. Вызов у объекта класса модели машинного обучения метода `model.score` возвращает аккуратность [31].

После анализа отчетов для каждого метода самым успешным был признан метод стохастического градиентного спуска (SGDClassifier). Он показал лучший результат аккуратности среди остальных методов. Его аккуратность равна 0,6818. Ближайшим конкурентом градиентного спуска является логистическая регрессия с аккуратностью 0,6776. По параметрам макроусреднения методы SGDClassifier и LogisticRegression почти равны с небольшим перевесом в сторону градиентного спуска (SGDClassifier). В таблице 3 представлены результаты для всех проанализированных методов.

Таблица 3 – Результаты метрик отобранных методов

Методы	**Accuracy**	**Macro avg (precision)**	**Macro avg (recall)**
ComplementNB	0,6478	0,66	0,63
MultinomialNB	0,5877	0,62	0,53
SVC	0,6523	0,67	0,64

Окончание таблицы 3

LogisticRegression	0,6776	0,69	0,66
SGDClassifier	0,6818	0,68	0,68
RidgeClassifier	0,6723	0,67	0,67
RandomForestClassifier	0,565	0,56	0,54

Далее представлен код для обучения и тестирования метода SGDClassifier в листинге 5, а на рисунке 12 изображен отчет классификаций. В отчете можно посмотреть как показатели макроусреднения, так и точность и полноту для каждого жанра.

	precision	recall	f1-score	support
business	0.67	0.77	0.72	170
detective	0.71	0.71	0.71	182
fantastic_books	0.64	0.57	0.60	160
fantsy_books	0.65	0.67	0.66	154
history	0.75	0.86	0.80	190
hobby	0.49	0.43	0.46	187
home_books	0.60	0.53	0.56	34
horrors	0.58	0.64	0.61	204
klassica	0.79	0.86	0.83	94
loves	0.69	0.62	0.66	153
poez_books	0.89	0.94	0.91	203
psyco_books	0.65	0.75	0.70	198
rel_books	0.79	0.85	0.82	122
skill_books	0.55	0.37	0.44	213
sport_books	0.64	0.63	0.63	199
warfare	0.76	0.73	0.74	183
accuracy			0.68	2646
macro avg	0.68	0.68	0.68	2646
weighted avg	0.68	0.68	0.68	2646

Рисунок 12 – Отчет классификации

Листинг 5 – Обучение и тестирование модели

```
sgd1 = SGDClassifier()
sgd1.fit(X_train_vec, y_train)
print(sgd1.score(X_test_vec, y_test))
y_pred_sgd1 = sgd1.predict(X_test_vec)
accuracy = accuracy_score(y_test, y_pred_sgd1)
print("Accuracy: ", accuracy)
draw_Conf(y_pred_sgd1, sgd1)
print(classification_report(y_test, y_pred_sgd1))
```

Матрица несоответствий показывает ошибки модели, то есть какое количество объектов она определила, как несоответствующий класс (жанр). Матрица представлена на рисунке 13.

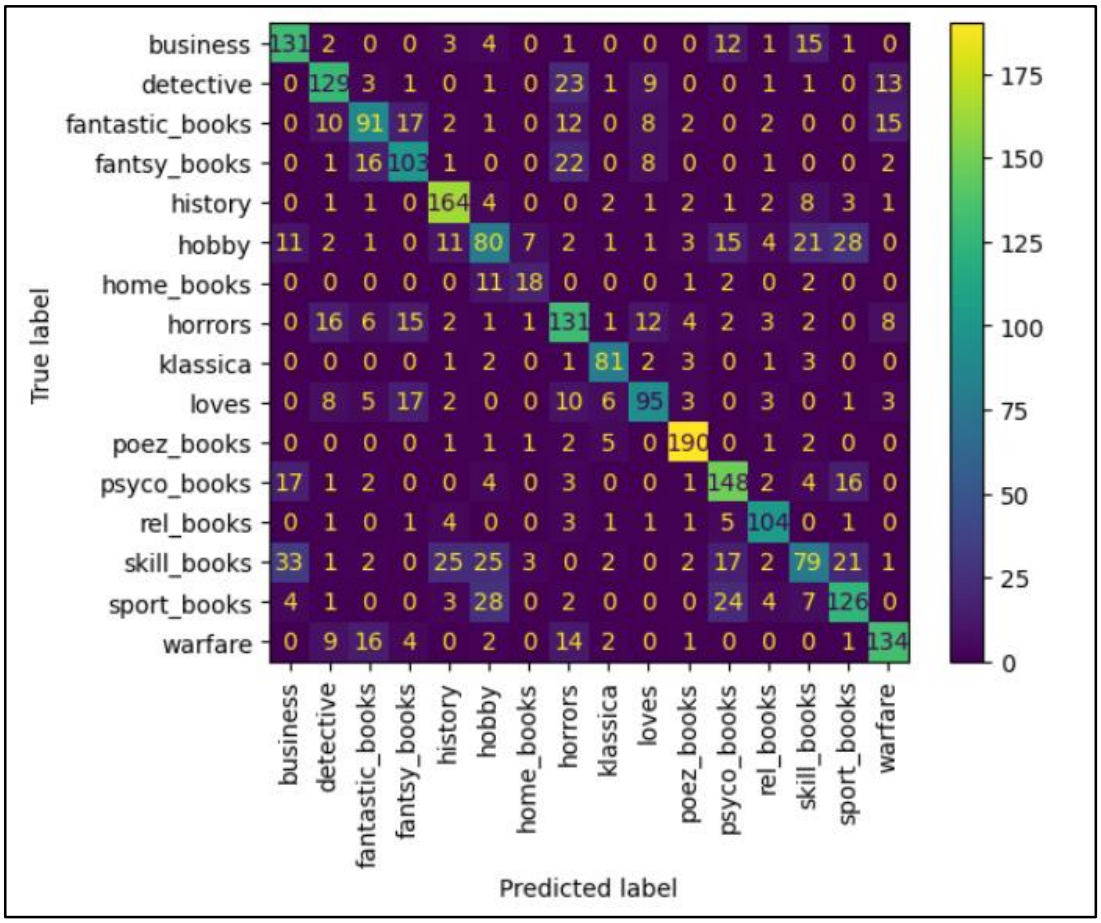

Рисунок 13 – Матрица несоответствий

Выводы по второй главе

В этой главе было дано определение машинному обучению и разобраны его виды. Далее были определены распространенные жанры книг и собран новый набор данных с помощью вспомогательной программы. После сбора книг был извлечен и обработан текст из каждой из них. После обработки текста был проведен анализ результатов методов классификации, в следствие чего была выбрана модель, показывающая лучшие результаты.

3. РАЗРАБОТКА ПРИЛОЖЕНИЯ

3.1. Требования к приложению

Приложение для определения жанра литературных произведений реализовано в виде web-приложения. В процессе проектирования приложения к системе были обозначены функциональные и нефункциональные требования к системе.

Функциональные требования

1. Система должна предоставлять результаты классификации.
2. В системе должна быть возможность регистрации и авторизации пользователей.

Нефункциональные требования

1. Система должна обрабатывать файлы формата pdf.
2. Система должна работать с файлами до 5 мегабайт.

На рисунке 14 представлена диаграмма вариантов использования для приложения классификации книг по жанрам. Главным актором выступает пользователь сайта. В качестве дополнительного актора обозначен администратор.

В рамках данного приложения для пользователя предусмотрены следующие варианты использования.

1. Вариант использования «регистрация». Пользователь может зарегистрироваться в приложении на сайте.
2. Вариант использования «авторизация». После процесса регистрации пользователь может войти в свой личный кабинет, используя данные при регистрации.
3. Вариант использования «загрузка файла». Пользователь может загрузить в приложение файл электронной книги для распознавания класса и дальнейшего просмотра результата.
4. Вариант использования «просмотра результата классификации». Пользователь может посмотреть результаты работы приложения в виде определения жанра загруженной книги.

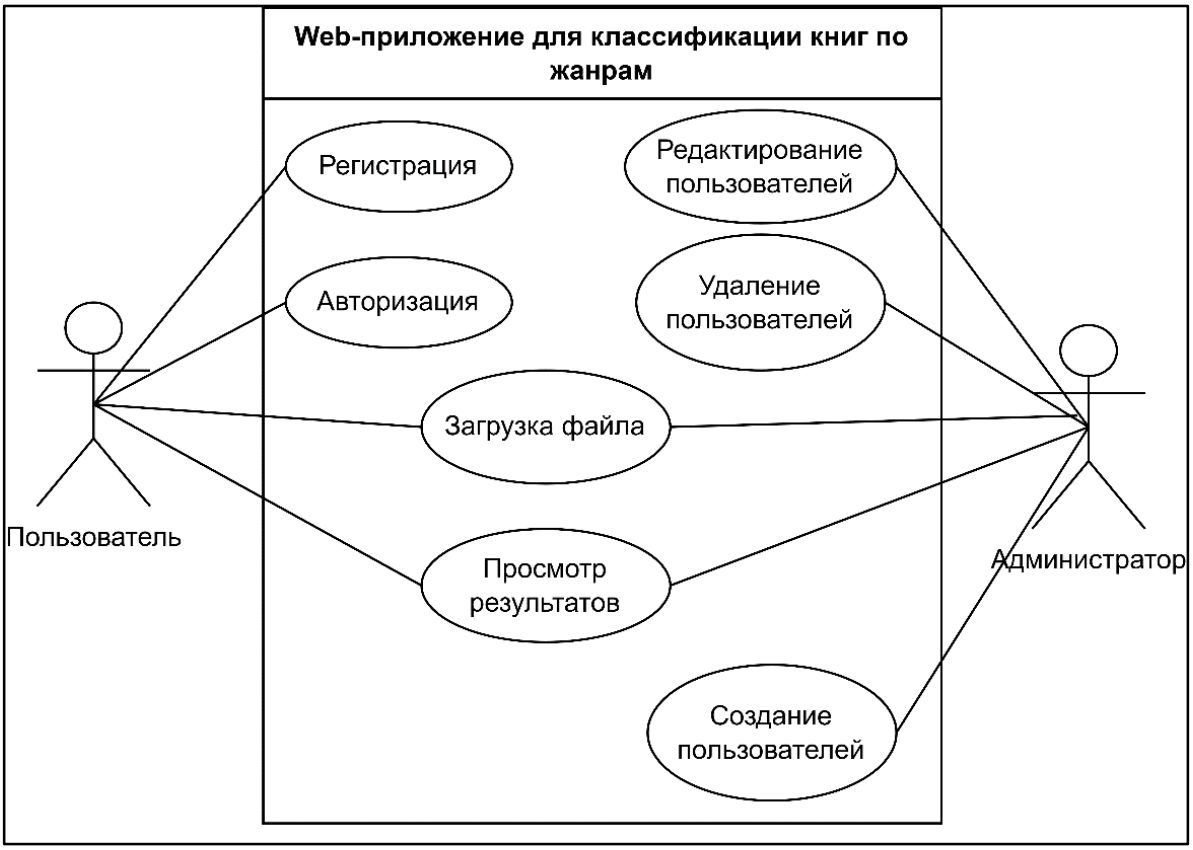

Рисунок 14 – Диаграмма вариантов использования

Для администратор также доступны варианты использования «загрузка файла» и «просмотра результата классификации». Но кроме этого у него есть особые варианты использования для администрирования.

1. Вариант использования «редактирование пользователей». Администратор может редактировать пользователей, добавлять их в группы, давать им права или вручную менять их данные.

2. Вариант использования «создание пользователей». В данном варианте администратор может создавать заранее пользователей, например, для какой-либо компании.

3. Вариант использования «удаление пользователей». Удаление пользователей администратором по тем или иным причинам.

3.2. Архитектура системы

Архитектура приложения разделена на несколько компонентов, взаимодействующих друг с другом. На рисунке 15 представлена диаграмма архитектуры приложения.

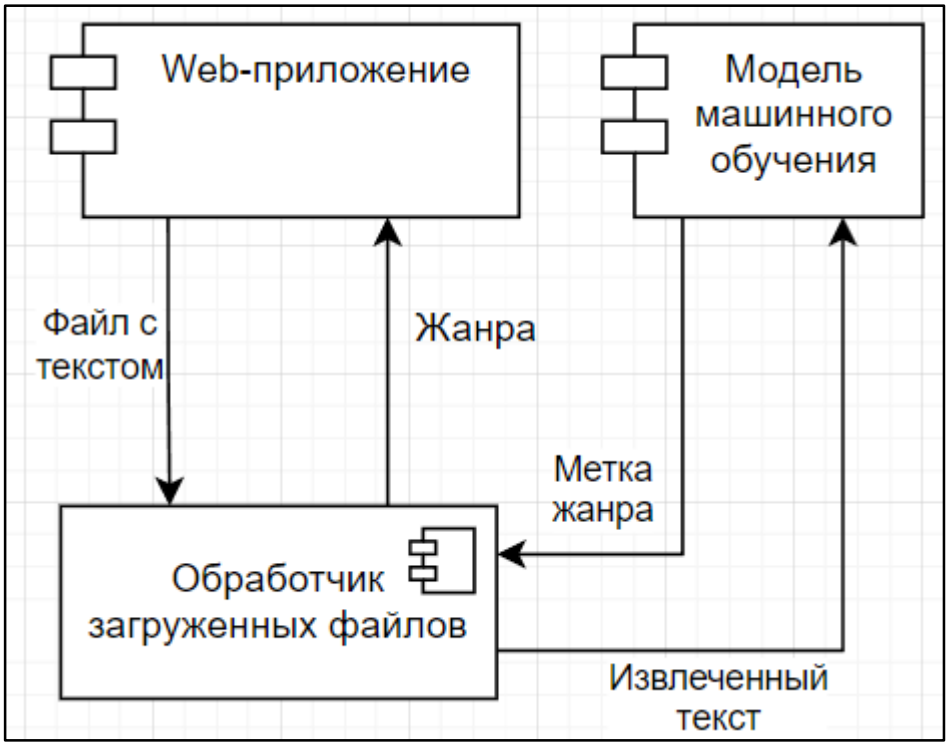

Рисунок 15 – Архитектура приложения

Далее описывается каждый компонент.

1. Web-приложение. Это приложение, получающее и передающее пользователю какие-либо данные, так же предоставляющее пользователям возможность использования классификации книг.

2. Обработчик загруженных файлов. Данный компонент извлекает текст из загруженных пользователем файлов, далее происходит предварительная очистка текста с последующей передачей оного в модель.

3. Модель машинного обучения. Получает предварительно очищенный текст, производит дополнительную обработку и анализирует текст. После анализа возвращает метку класса.

Порядок действий для главной функции приложения изображен на диаграмме деятельности (рисунок 16).

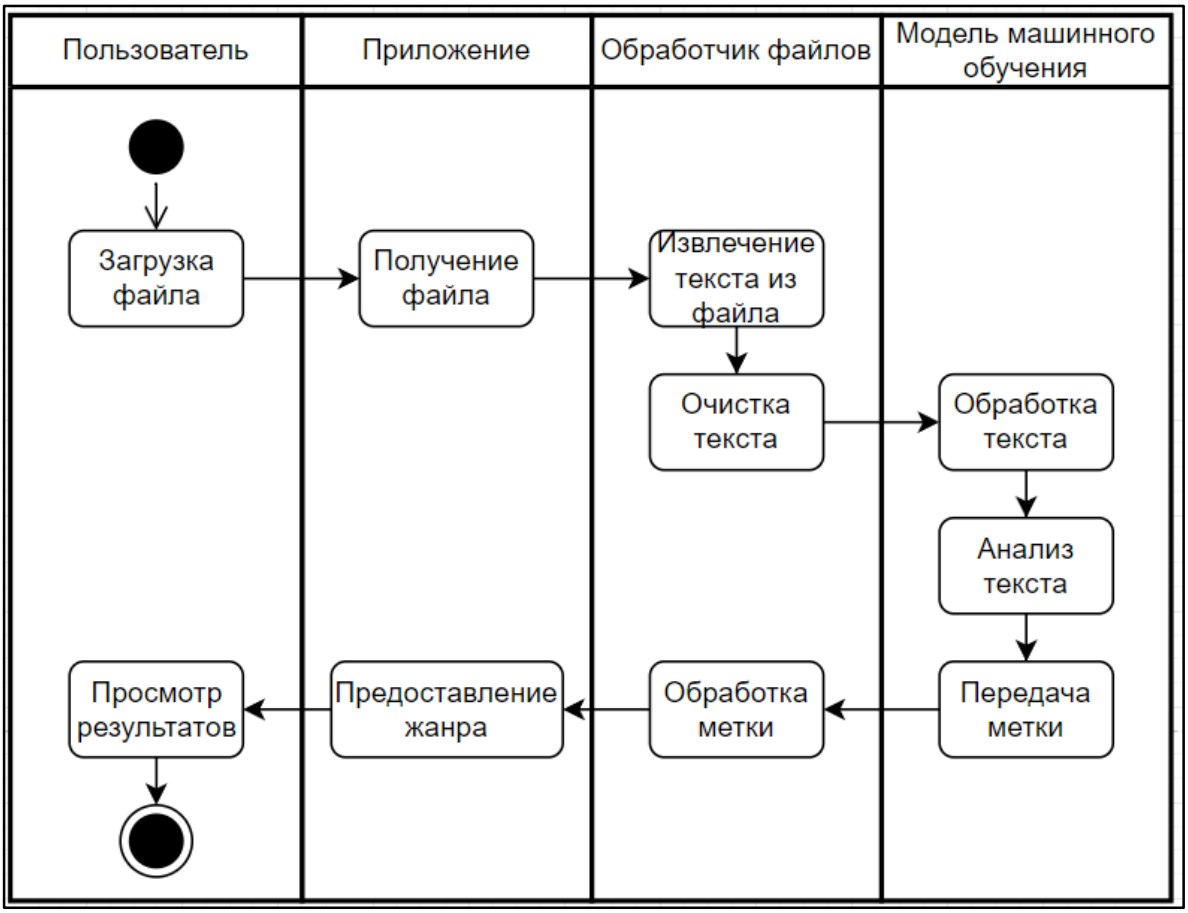

Рисунок 16 – Диаграмма деятельности

Поскольку в приложении есть возможность регистрации, необходимо хранить данные пользователей, такие как логин, пароль, электронная почта и другие. Для хранения данной информации используется база данных пользователей. На рисунке 17 изображена схема базы данных.

user		
PK	**ID**	
	password	VARCHAR
	last_login	DATETIME
	is_superuser	BOOLEAN
	username	VARCHAR
	last_name	VARCHAR
	email	VARCHAR
	is_staff	BOOLEAN
	is_active	BOOLEAN
	date_joined	DATETIME
	first_name	VARCHAR

Рисунок 17 – Схема базы данных пользователей

3.3. Проектирование графического интерфейса

Графический интерфейс состоит из нескольких web-страниц, представленные в виде шаблонов html-файлов.

1. Страница регистрации. На данной странице отображаются поля регистрации на сайте, такие, как электронная почта, логин, пароль, подтверждение пароля и некоторые дополнительные необязательные поля. После нажатия на кнопку регистрации, происходит проверка на корректность вводимых данных.

2. Страница авторизации. На странице авторизации присутствуют поля логина и пароля для входа.

3. Главная страница. На главной странице отображаются правила использования приложением для работы с классификатором. На рисунке 18 представлен макет страницы.

Рисунок 18 – Макет главной страницы

4. Загрузка файла. Здесь пользователь может загрузить свой файл для классификации. Шаблон страницы загрузки изображен на рисунке 19.

5. Результат. На данной странице отображается результат процесса классификации книги в виде жанра.

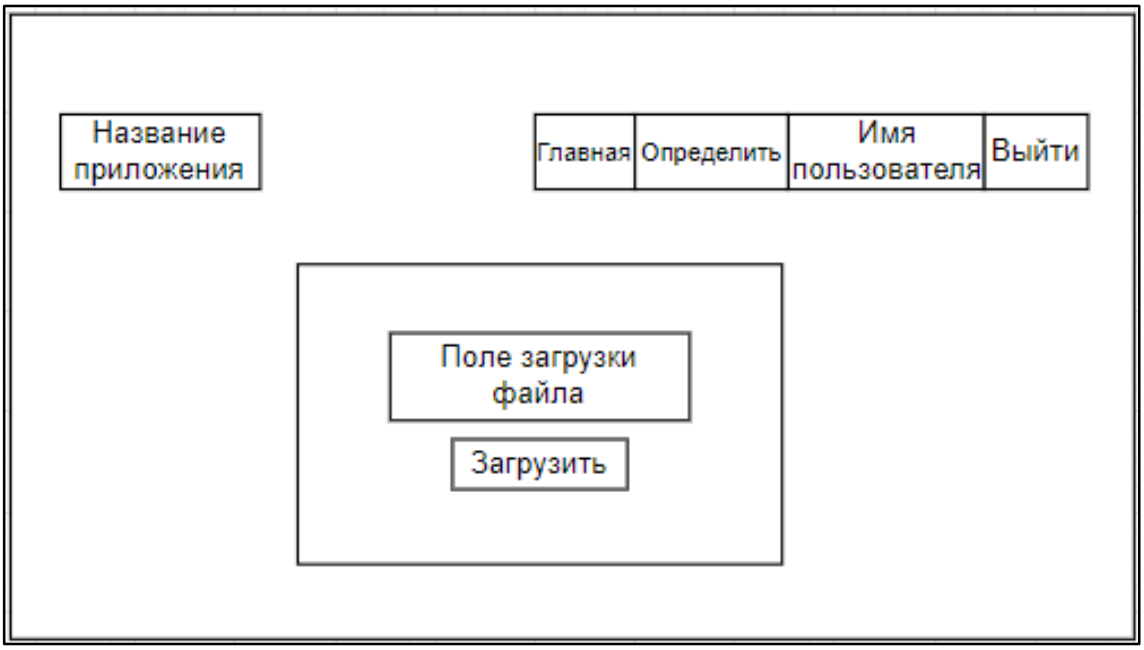

Рисунок 19 – Шаблон страны загрузки файла

3.4. Программная реализация

Для реализации приложения для классификации литературных произведений по жанрам был использован язык программирования Python версии 3.9.13. Графический интерфейс был реализован с помощью языка разметки HTML и языка таблицы каскадных стилей CSS. Для написания кода использовалась среда программирования PyCharm.

В процессе реализации web-приложения были использованы следующие библиотеки и фреймворки.

1. Фреймворк Django (4.2.13) [32]. Это фреймворк с открытым исходным кодом для создания web-приложений на языке Python.

2. Пакет sklearn (1.2.2) [33]. Пакет, предоставляющий возможность работы с машинным обучением и набор данных. Так же с открытым исходным кодом.

3. Библиотека NLTK (3.8.1) [34]. Библиотека для работы с естественными языками. Используется как для ручной токенизации текста, так и для дополнительной обработки.

4. Модуль PyPDF2 (3.0.1) [35]. Модуль для работы с файлами формата pdf. Он позволяет извлекать данные, объединять pdf-документы, обрезать страницы и прочее. С открытым исходным кодом.

5. Фреймворк Bootstrap (5.2.3) [36]. Данный фреймворк предоставляет готовые дизайнерские шаблоны для интеграции в web-приложения. Есть вариант использования через CDN.

Реализация графического интерфейса

Интерфейс приложения состоит из 5 страниц, реализованных с помощью HTML. Для стилей используется вышеупомянутый фреймворк Bootstrap.

Главная страница представляет собой инструкцию для получения результата классификации. На ней также доступны кнопки для перехода на страницы загрузки файла, входа или регистрации. Если пользователь уже авторизовался, то на главной странице вместо «Войти» будет отображаться его логин, а вместо «Регистрация» – «Выйти» (рисунок 20).

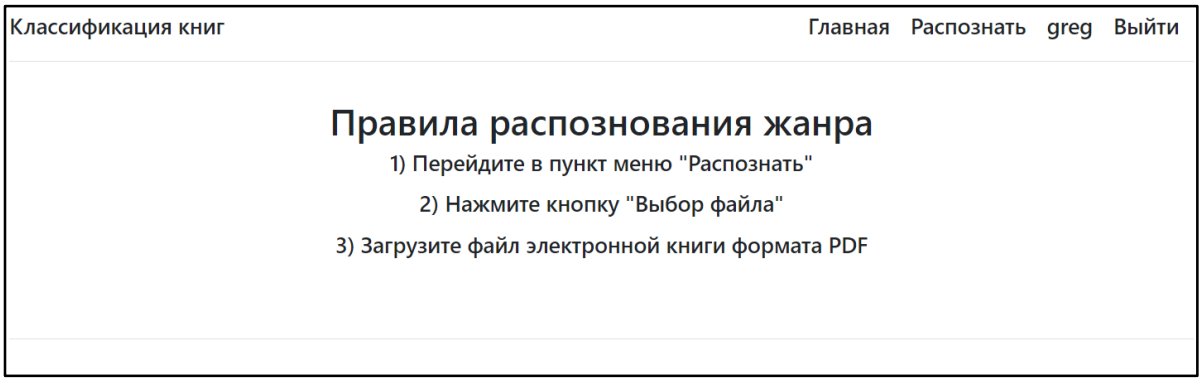

Рисунок 20 – Главная страница

На странице загрузки файла пользователь может загрузить файл электронной книги в виде pdf для дальнейшей ее классификации. На рисунке 21 изображена страница загрузки файла.

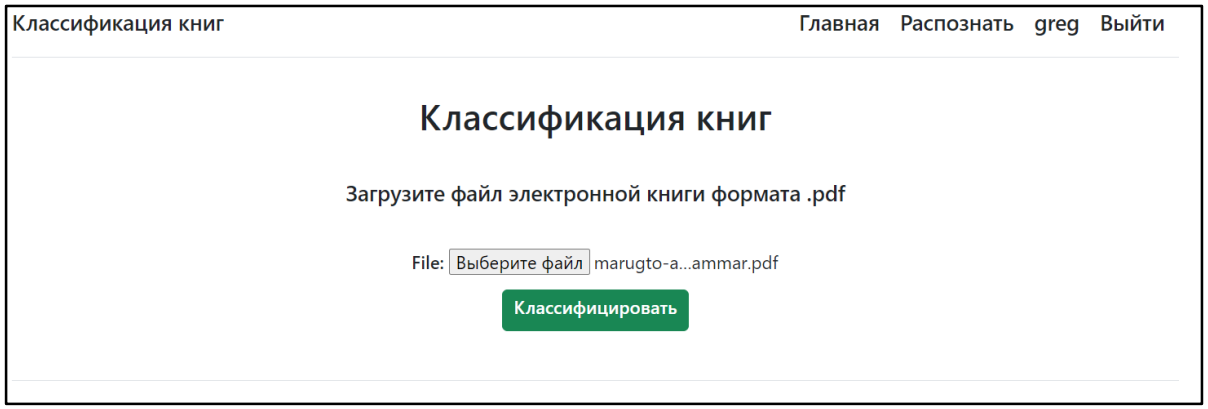

Рисунок 21 – Страница загрузки файла

После нажатия на кнопку идет перенаправление на страницу с результатом. На ней предоставляется ответ в виде определения жанра книги (рисунок 22).

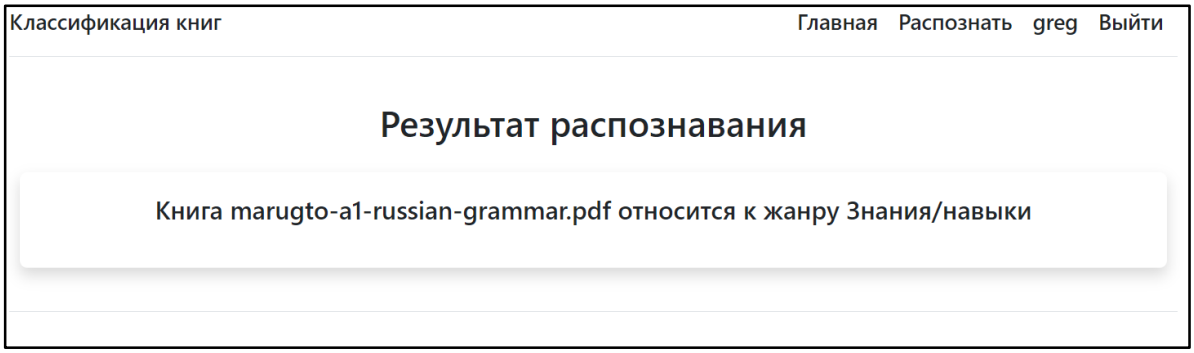

Рисунок 22 – Страница результата классификации

Реализация приложения

В приложении было реализовано четыре функций. Первые две функции приведены далее (листинги 6–7). Они используются для получения файлов электронных книг от пользователей с их дальнейшей загрузкой и сохранением.

Листинг 6 – Получение файла от пользователя

```
def upload_file(request):
    if request.method == 'POST':
        form = FileUploadForm(request.POST, request.FILES)
        if form.is_valid():
            handle_uploaded_file(form.cleaned_data['file'])
    else:
        form = FileUploadForm()
    return render(request, 'load.html', {'form': form})
```

Листинг 7 – Сохранение файла

```
def handle_uploaded_file(f):
    with open(f"uploads/{f.name}", "wb+") as destination:
        for chunk in f.chunks():
            destination.write(chunk)
```

Для обработки сохраненных файлов была написана функция для извлечения и очистки текстов из загруженного пользователем файла. В листинге 8 представлена реализация.

Листинг 8 – Обработка файла и анализ текста

```
def predict():
    folder_path = "uploads"
    files = os.listdir(folder_path)
    files_modif = {}
    for file in files:
        file_path = os.path.join(folder_path, file)
        if os.path.isfile(file_path):
            mod_files = os.path.getmtime(file_path)
            files_modif[file] = mod_files
    latest_file = max(files_modif, key=files_modif.get)
    text = ""
    with open(f"uploads/{latest_file}", "rb") as filehandle:
        pdf = PdfReader(filehandle)
        pages = len(pdf.pages)
        for i in range(pages):
            page = pdf.pages[i]
            text += page.extract_text().strip()
    vectorizer = load(f'books/vectorize_sgd1_678.joblib')
    with open("books/sgd1_678.pkl", 'rb') as file:
        pickle_model = pickle.load(file)
    X_pred = pd.Series(text)
    X_pred_vec = vectorizer.transform(X_pred)
    pred = pickle_model.predict(X_pred_vec)
    return pred[0]
```

После получения метки она соотносится со словарем, и, при нахождении определенного ключа, возвращает обычный жанр в виде текста для последующего предоставления пользователю (листинг 9).

Листинг 9 – Возврат результата пользователю

```
def predict(request):
    pred = Prediction.predict()
    template = loader.get_template('predict.html')
    context = {"pred": ru_genres.get(pred[0])}
    rendered_page = template.render(context, request)
    return HttpResponse(rendered_page)
```

3.5. Тестирование приложения

Для проверки работы приложения был проведен ряд тестов. Результаты тестирования реализованного приложения представлены в таблице 4.

Таблица 4 – Результаты тестирования приложения

№	Входные данные	Ожидаемый результат	Полученный результат
1	Пользователь нажимает на кнопку выбора файла, выбирает файл и загружает его	Загруженный пользователем файл сохраняется в определенной папке	Файл сохранен
2	Пользователь нажимает на кнопку «классифицировать»	После нажатие происходит перенаправление на новую страницу с результатом классификации	Выполнено перенаправление и показан результат
3	При авторизации пользователь вводит логин и пароль и нажимает кнопку входа в аккаунт	Успешная авторизация	Пользователь авторизован
4	При авторизации пользователь вводит неверные логин и/или пароль и нажимает кнопку входа в аккаунт	Пользователь не авторизовался и получил предупреждение о неподходящих данных	Аналогичный с ожидаемым
5	Пользователь для регистрации вводит логин, электронную почту, дополнительные необязательные поля по своему усмотрению, пароль, подтверждает пароль и нажимает кнопку регистрации	Система проверяет введенные данные на корректность и создает нового пользователя	Пользователь создан
6	Пользователь нажимает на кнопку выхода из аккаунта	Успешное разлогирование и перенаправление на главную страницу	Пользователь вышел из своего аккаунта и был перенаправлен на главную страницу

Выводы по третьей главе

В главе были определены требования к приложению, описана архитектура приложения, спроектирован и реализован графический интерфейс пользователя, реализовано само web-приложение и подключена возможность обрабатывать pdf-файлы и работать с моделью машинного обучения для классификации. В конце было проведено тестирование приложения.

ЗАКЛЮЧЕНИЕ

В рамках выпускной квалификационной работы был разработан web-сервис, который по анализу текста электронных книг позволяет определить жанр литературного произведения. При этом были решены следующие задачи.

1. Произведен обзор литературы и существующих приложений по предметной области.
2. Собран свой набор данных для дальнейшего обучения
3. Подготовлена обучающая и тестовая выборки.
4. Выполнена очистка и предобработка текстовых данных для дальнейшей реализации модели машинного обучения решения задачи классификации книг.
5. Проведено обучение и тестирование нескольких методов машинного обучения с дальнейшим выбором модели с наилучшими результатами.
6. Выполнено проектирование web-приложения.
7. Реализован и протестирован web-сервис для классификации литературных произведений по жанрам.

Таким образом, все поставленные в ходе выпускной квалификационной работы задачи были выполнены.

ЛИТЕРАТУРА

1. Литературный энциклопедический словарь. [Электронный ресурс] URL: http://niv.ru/doc/encyclopedia/literature/articles/83/literatura.htm (дата обращения: 26.02.2024 г.).

2. Литература – что это, определение, термин, история, форма, место. [Электронный ресурс] URL: https://litpr.ru/literatura/ (дата обращения: 26.02.2024 г.).

3. Доля рынка электронных книг, размер, тенденции и отраслевой анализ. [Электронный ресурс] URL: https://www.mordorintelligence.com/ru/industry-reports/e-book-market (дата обращения: 27.02.2024 г.).

4. «ЛитРес» зафиксировал рост продаж цифровых книг. [Электронный ресурс] URL: https://www.rbc.ru/technology_and_media/01/09/2023/64f1d7369a7947e8113b9daa (дата обращения: 27.02.2024 г.).

5. «Литрес»: в III квартале рынок цифровых книг вырос почти на треть. [Электронный ресурс] URL: https://www.retail.ru/news/litres-v-iii-kvartale-rynok-tsifrovykh-knig-vyros-pochti-na-tret/ (дата обращения: 27.02.2024 г.).

6. Электронные книги (рынок России). [Электронный ресурс] URL: https://clck.ru/396fki (дата обращения: 27.02.2024 г.).

7. Storytel вселяется в «Дом историй». [Электронный ресурс] URL: https://www.kommersant.ru/doc/5811909 (дата обращения: 28.02.2024 г.).

8. Amazon Comprehend. [Электронный ресурс] URL: https://aws.amazon.com/ru/comprehend/features/ (дата обращения: 03.03.2024 г.).

9. IBM Watson Natural Language Understanding. [Электронный ресурс] URL: https://www.ibm.com/products/natural-language-understanding (дата обращения: 28.02.2024 г.).

10. Cloud Natural Language. [Электронный ресурс] URL: https://cloud.google.com/natural-language (дата обращения: 05.03.2024 г.).

11. Cloud Healthcare API. [Электронный ресурс] URL: https://cloud.google.com/healthcare-api/docs/how-tos/nlp (дата обращения: 10.03.2024 г.).

12. Демоверсия Natural Language API. [Электронный ресурс] URL: https://cloud.google.com/natural-language#demo (дата обращения: 10.03.2024 г.).

13. Поддержка языков Natural Language API. [Электронный ресурс] URL: https://cloud.google.com/natural-language/docs/languages (дата обращения: 10.03.2024 г.).

14. Azure AI. [Электронный ресурс] URL: https://azure.microsoft.com/ru-ru/products/ai-services/#Services (дата обращения: 20.03.2024 г.).

15. Язык ИИ Azure. [Электронный ресурс] URL: https://learn.microsoft.com/ru-RU/azure/ai-services/language-service/overview#available-features (дата обращения: 20.03.2024 г.).

16. H. Chiang, Y. Ge, C. Wu. Classification of Book Genres By Cover and Title. [Электронный ресурс] URL: http://cs229.stanford.edu/proj2015/127_report.pdf (дата обращения: 23.03.2024 г.).

17. C. Kundu, L. Zheng. Deep multi-modal networks for book genre classification based on its cover. [Электронный ресурс] // arXiv.org. 2020. Дата обновления: 15.11.2020 г. URL: https://arxiv.org/abs/2011.07658v1 (дата обращения: 26.03.2024 г.).

18. Stanford Classifer. [Электронный ресурс] URL: https://nlp.stanford.edu/software/classifier.shtml (дата обращения: 24.03.2024 г.).

19. T. Mikolov, K. Chen, G. Corrado, J. Dean. Efficient Estimation of Word Representations in Vector Space. [Электронный ресурс] // arXiv.org. 2013. Дата обновления: 07.09.2013 г. URL: https://arxiv.org/abs/1301.3781 (дата обращения: 25.03.2024 г.).

20. Genre Identification and the Compositional Effect of Genre in Literature. [Электронный ресурс] URL: https://aclanthology.org/C18-1167.pdf (дата обращения: 30.03.2024 г.).

21. Использование анализа семантической близости слов при решении задачи определения жанровой принадлежности текстов методами глубокого обучения. [Электронный ресурс] URL: https://goo.su/RbMg2vC (дата обращения: 03.04.2024 г.).

22. Convolutional neural networks for sentence classification. [Электронный ресурс] URL: https://aclanthology.org/D14-1181/ (дата обращения: 09.04.2024 г.).

23. Классификация книг по жанрам на основе текстовых описаний посредством глубокого обучения. [Электронный ресурс] URL: http://surl.li/spvdk (дата обращения: 12.04.2024 г.).

24. Классификация научных текстов методами машинного обучения. [Электронный ресурс] URL: https://cyberleninka.ru/article/n/klassifikatsiya-nauchnyh-tekstov-po-spetsialnostyam-metodami-mashinnogo-obucheniya (дата обращения: 19.04.2024 г.).

25. Определение машинного обучения. [Электронный ресурс] URL: https://www.oracle.com/cis/artificial-intelligence/machine-learning/what-is-machine-learning/ (дата обращения: 22.04.2024 г.).

26. Типы машинного обучения. [Электронный ресурс] URL: https://www.sap.com/central-asia-caucasus/products/artificial-intelligence/what-is-machine-learning.html (дата обращения: 22.04.2024 г.).

27. Где можно купить электронные книги. [Электронный ресурс] URL: https://selfpub.ru/domains/ (дата обращения: 23.04.2024 г.).

28. Сайт электронных книг AvidReaders. [Электронный ресурс] URL: https://avidreaders.ru/genre/ (дата обращения: 24.04.2024 г.).

29. Сайт электронных книг Flibusta. [Электронный ресурс] URL: https://flibusta.one/books-genres/ (дата обращения: 24.04.2024 г.).

30. Метрики классификации и регрессии. [Электронный ресурс] URL: https://education.yandex.ru/handbook/ml/article/metriki-klassifikacii-i-regressii (дата обращения: 29.04.2024 г.).

31. Документация sklearn. [Электронный ресурс] URL: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDClassifier.html#sklearn.linear_model.SGDClassifier.score (дата обращения: 29.04.2024 г.).

32. Django. [Электронный ресурс] URL: https://docs.djangoproject.com/en/4.1/ (дата обращения: 01.05.2024 г.).

33. Sklearn. [Электронный ресурс] URL: https://scikit-learn.ru/ (дата обращения: 26.04.2024 г.).

34. NLTK. [Электронный ресурс] URL: https://www.nltk.org/ (дата обращения: 27.04.2024 г.).

35. PyPDF2. [Электронный ресурс] URL: https://pypdf2.readthedocs.io/en/3.x/ (дата обращения: 24.04.2024 г.).

36. Bootstrap. [Электронный ресурс] URL: https://bootstrap-4.ru/ (дата обращения: 05.05.2024 г.).