МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «КубГУ»)

Факультет компьютерных технологий и прикладной математики
Кафедра информационных технологий

Допустить к защите
Заведующий кафедрой
канд. физ.-мат. наук, доц.
_____В.В. Подколзин
_____2024г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
(БАКАЛАВРСКАЯ РАБОТА)

ПРОГРАММНАЯ СИСТЕМА КЛАССИФИКАЦИИ ТЕКСТОВОЙ ИНФОРМАЦИИ

Работу выполнил _____ И.А. Молчанов
(подпись)

Направление подготовки 01.03.02 Прикладная математика и информатика

Направленность Программирование и информационные технологии

Научный руководитель
канд. физ-мат. наук, зав. каф., доц._____ В.В. Подколзин
(подпись)

Нормоконтролер
канд. пед. наук, доц. _____ А.В. Харченко
(подпись)

Краснодар
2024

РЕФЕРАТ

Выпускная квалификационная работа 45 с., 17 рис., 7 источников.

НЕЙРОННЫЕ СЕТИ, ДАТАСЕТЫ, КЛАССИФИКАЦИЯ, МОДЕЛЬ, НАБОР ДАННЫХ, БИБЛИОТЕКИ, КЛАСТЕРИЗАЦИЯ.

Цель работы: разработать систему классификации текстовой информации.

В процессе работы были изучены: виды нейронных сетей, задачи, решаемые с помощью искусственного интеллекта, методы решения этих задач.

В практической части был разработана обучаемая классифицирующая система.

Средства разработки: Python, библиотек Matplotlib, Seaborn, Nltk, Re, PyMorphy2, Numpy, Pandas, Sklearn, Gensim, среда разработки GoogleColab.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В данной выпускной квалификационной работе рассматривается задача реализации нейронной сети для решения задачи классификации новостных сообщений для выявления среди них фейков и семантической кластеризации текстов. Данная работа направлена на адаптирование метода классификации текстов под русский язык и соответствующую аудиторию.

Дипломная работа имеет практическое значение, так как создание системы автоматической классификации текстов и выявления фейковых новостей позволит улучшить качество информационного пространства в интернете и повысить уровень доверия к информации.

Предлагаются два набора данных: тексты первого необходимо классифицировать и обнаружить среди них заведомо ложные тексты, тексты второго необходимо кластеризовать по семантическим признакам с первым.

1 Методы обучения нейронной сети

1.1 Нейронные сети и искусственный интеллект

Под понятием «интеллектуальные системы» подразумеваются компьютерные системы, обладающие способностью к самостоятельным решениям на основе анализа и обработки масштабных данных. В разнообразии сфер их применения: медицина, банковское дело, производство, транспорт - есть место для этой технологии. Этап начальный в развитии данного термина относится к 1950-м годам, когда первые шаги на пути к пониманию человеческого мозга были сделаны в направлении создания искусственного.

Нейронные сети - одна из категорий интеллектуальных систем. За основу в этих системах берется имитация процессов, протекающих в человеческом мозге. Состоят эти сети из обширного количества взаимосвязанных искусственных нейронов, которые организуются по слоям. Применение находят в таких сферах: классификация, регрессия, кластеризация, распознавание образов и в других областях.

Преимуществ нейронных сетей множество, при этом ключевым считается способность к обучению используя доступные данные. Эта возможность способствует автоматическому выявлению закономерностей в данных, что актуально для решения новых проблем с помощью уже наработанных знаний. Кроме указанного свойства, нейронные сети оказываются эффективными в работе не только с числами, но и обрабатывая неструктурированную информацию, например, тексты и изображения. Их универсальность позволяет применяться в разнообразных прикладных задачах, увеличивая их ценность как инструмента [2].

Многообразие архитектур нейронных сетей представлено различными моделями: список содержит такие наименования, как перцептрон, рекуррентные и сверточные нейронные сети, а также сети адаптивного

резонанса и другие виды. Каждый тип архитектуры ориентирован на выполнение конкретных заданий, обладая уникальными преимуществами и потенциальными ограничениями.

Путь к созданию систем, отличающихся высокой точностью и эффективностью, через использование нейронных сетей в интеллектуальных системах, открывается, что способствует их возможности справляться с комплексными задачами и адаптироваться к варьируемым условиям. Однако такое развитие и обучение таких сетей предполагают необходимость больших объемов данных и высокие требования к вычислительной мощности, вдобавок к опыту и экспертным знаниям специалистов, задействованных в специфических областях применения нейросетей.

1.2 Виды нейронных сетей

Классификация нейронных сетей обуславливается типами задач, которые они способны решать, и методами, применяемыми для этого.

Прямые нейронные сети (англ. FeedforwardNeuralNetworks, FFNN) представляют собой одну из базовых вариаций. Эти системы организованы с подключением входного слоя, последующими скрытыми слоями и заканчиваются выходным слоем. Конструкция каждого слоя включает в себя нейроны, связанные с элементами следующего и предыдущего уровней. Широко аплицируемые в задачах, таких как: классификация и регрессия, эти сети находят разнообразное применение в данной сфере.

Сверточные нейронные сети (CNN, с английского Convolutional Neural Networks) находят применение в процессах обработки изображений и в анализе сигналов таких, как, например, аудио. Основу их составляют: сверточные слои, слои подвыборки, а также полносвязные слои. Назначение сверточных слоев заключается в извлечении признаков из данных. Уменьшение размерности данных – задача слоев подвыборки.

Рекуррентные нейронные сети (сокращённо – RNN, RecurrentNeuralNetworks) применяются при анализе последовательностей, включая, бытие текстов и временные ряды.

Сети с долгой краткосрочной памятью (англ. Long Short-Term Memory, LSTM). Поддержка обработки последовательностей высокой длины характеризует, представляющие собой разновидность рекуррентных нейронных сетей.

Генеративно-состязательные сети (англ., Generative Adversarial Networks, GAN) применяются в целях создания контента: различные типы могут включать изображение, текст, а также звук.

Автокодировщики (по-английски Autoencoders, AE) находят применение в таких задачах, как: сжатие информации, извлечение ключевых характеристик и восстановление начальных данных.

В каждом типе нейронной сети заложены свои уникальные достоинства и ограничения, актуальность которых манфестируется в разнообразии применений в дисциплинах машинного обучения и также в сфере искусственного интеллекта.

1.3 Функции нейронных сетей

Используемые в машинном обучении нейронные сети выступают инструментом, находящим применение при решении разнообразных задач, ассоциируемых с обработкой информации и анализом данных. К основным функциям, которые они выполняют, следует отнести следующие:

Классификация предполагает разделение данных на разные категории, используя предварительно данную информацию для обучения. Так, нейронная сеть, проходя процесс обучения, получает способность к разграничению изображений на категории, например: кошки, собаки.

Регрессия олицетворяет собой процесс аппроксимации функций и создания моделей для изучения взаимосвязей между входными показателями и результативными значениями.

Кластеризация задействуется в процессах, когда потребуется группировка информации, базируясь на различиях либо схожих моментах. Этот метод находит своё применение при выявлении узоров внутри данных или при формировании групп из объектов, обладающих похожими характеристиками.

Речь идет о процессе синтезирования новых данных, которые основаны на уже имеющихся материалах. Так, нейронная сеть, подвергшаяся обучению, способна производить изображения, еще не виданные, опираясь на предоставленный ей набор графических данных.

Рекомендательная способность является ключом к предложению товаров или услуг, основываясь на анализе предыдущих покупок или активности пользователя.

Технология распознавания речи в тандеме с анализом на природном языке обладает ценностью для разработки голосовых ассистентов и чат-ботов, оснащённых функцией текстовой генерации.

Каждая задача и подходящий ей вид нейронной сети подвергаются тщательному анализу с целью оценки их соответствия поставленным требованиям. При этом проводится изучение детальных аспектов задачи и проводится сравнительная оценка преимуществ и недостатков рассматриваемых моделей. Минимальное пренебрежение в процессе анализа может привести к ухудшению итогов или даже к полностью ошибочным результатам.

1.4 Структура нейронной сети

Объект, известный как модель нейронной сети, имитирует функционирование нервной системы органических существ. Эта модель

включает в себя нейроны, между которыми установлены связи, функционирующие как передатчики сигналов между данными нейронами.

Элементарной звеном в архитектуре нейронной сети предстает нейрон. Входные сигналы принимаются им, после чего происходит ряд вычислительных манипуляций; завершается процесс передачей выходного сигнала наступающему нейрону. Традиционно устройство нейрона представляется в следующем виде:

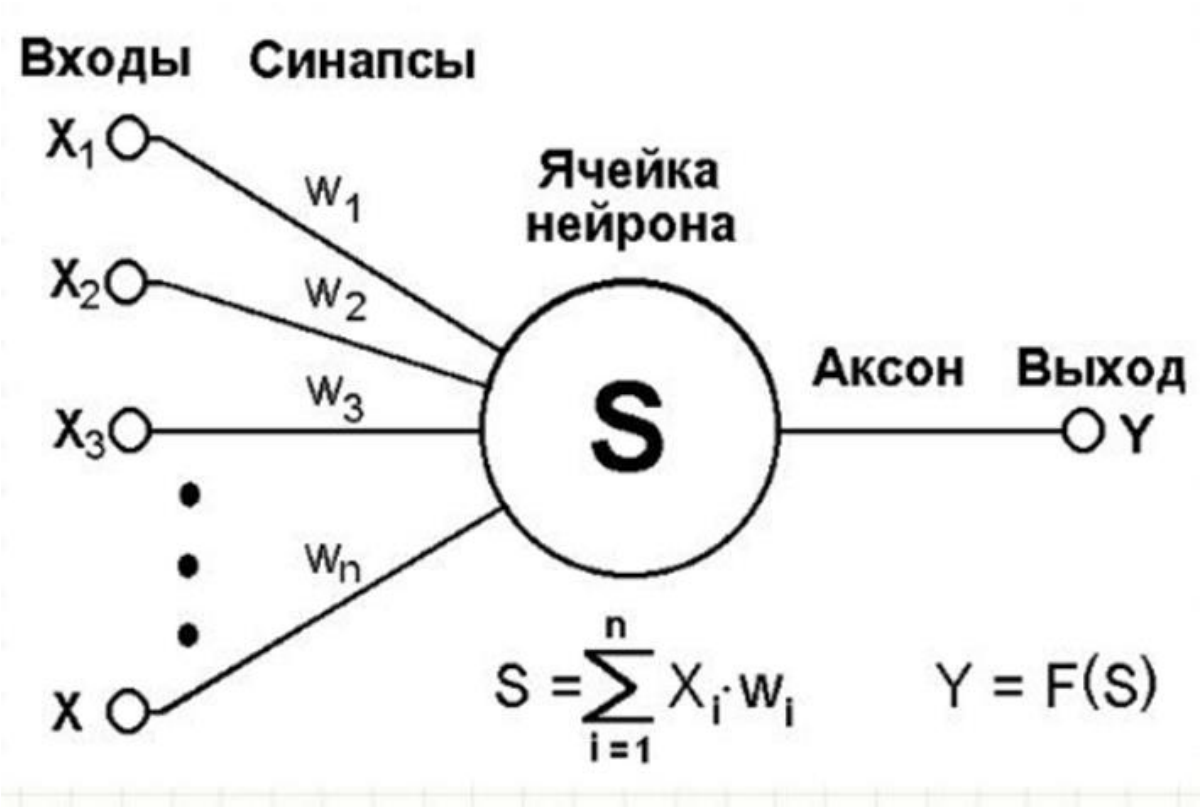


Рисунок 1 – Модель нейрона ИНС

Состав нейрона определяется списком из четырех элементов:

1) входные сигналы;
2) сумматорная функция;
3) функция активации;
4) выходной сигнал.

Первоначально, входные сигналы от поступают к нейрону как из внешней среды, так и от других нейронов. Важность каждого сигнала

определяется специфическим весовым коэффициентом, на который последний умножается. После этого блок сумматорной функции интегрирует умноженные сигналы. Происходит передача этого результата в функцию активации. В зависимости от поставленной задачи, выходное значение в функции активации может преобразовано в полезный формат, причем функции активации бывают линейные либо нелинейные. Наконец, трансформированный выходной сигнал отправляется далее: либо к последующим нейронам, либо применяется для решения целевых задач.

Слои, объединяющие нейроны в структуре нейронной сети, делятся на несколько типов: входной слой подвергается воздействию входных данных, выходной слой осуществляет передачу выходных данных. Существовать могут один или больше скрытых слоев, расположенных между ними. Особую роль играют связи, организованные между нейронами различных слоев: они назначаются как веса, что позволяет определять значимость каждого сигнала при его передаче.

Пример нейронной сети с двумя скрытыми слоями:

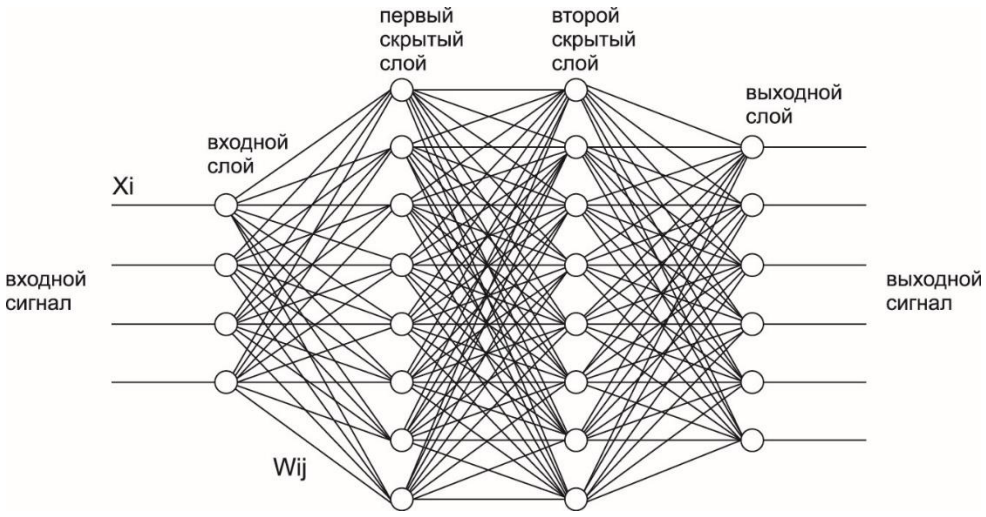


Рисунок 2 – ИНС с двумя скрытыми слоями

На рисунке представлена модель, куда входные данные подавались на начальный слой. Посредством умножения на коэффициенты веса, передача

происходит через два последующих скрытых слоя, а за тем — полученный результат передается из заключительного выходного слоя. Наличие связей каждого нейрона в слое с каждым нейроном в следующем слое объявляется характерным для полносвязной сети.

Опциональным параметром считается выбор количества скрытых слоев, и решение по данному вопросу принимают в зависимости от цели: улучшение точности или ускорение вычислений. Обязательным условием остаются наличие входного и выходного слоев. При этом, структура нейронной сети может быть адаптирована в соответствии с задачами, что обеспечивает данному инструменту высокую степень гибкости.

1.5 Требования к нейронной сети

Для признания работы, проделанной нейронной сетью, как эффективной, она должна соответствовать нескольким критериям:

Фиксация количества входных и выходных значений в абсолютное необходимое условие, обусловлено четкой связью каждого нейрона входного слоя с каждым нейроном выходного аналога. Изменение числа данных на входе и выходе приводит к изменениям в количестве связей между слоями, комплицируя процесс обучения нейросети и уводя в сторону увеличения времени для обработки данных. Эта закономерность вызывает необходимость контроля за стабильностью числа нейронов в этих критически важных участках сетевой структуры.

Справедливо утверждать, что для успешного решения задачи дизайн нейронной сети потребует подбора адекватного количества нейронов и слоев: ни слишком малого, ни чересчур обширного. Недостаточно крупные сети, в свою очередь, могут оказаться неспособными на извлечение релевантных характеристик или на выполнение нужных вычислений из предоставленных данных, что не позволяет им адекватно функционировать в поставленной задаче. В то же время, излишне большие сети сталкиваются с риском

переобучения, при котором они начинают запоминать входные данные вместо того, чтобы обрабатывать и обобщать полученную информацию, что ведет к снижению способности к обобщению и затрудняет адаптацию к новым данным.

Правильная настройка нейронной сети является ключевой: в её задачи входит определение наилучших параметров. Сюда относятся: количество нейронов, их слои, скорость обучения, а также выбор подходящих функций активации. Такие меры содействуют быстрой сходимости сети к желаемому решению и способствуют снижению риска переобучения.

Обеспечение эффективности нейронной сети требует достаточности обучающих данных. Использование большего объема данных способствует повышению точности результатов сети. В то же время, избыточное количество данных может вызвать переобучение, следовательно, выбор оптимального размера набора данных для обучения становится критически важным.

Обучение нейронной сети представляет собой процесс адаптации её весов, основываясь на анализе входных данных для повышения точности своей работы. Регулировка этих весов возможна через применение различных методик обучения: обратное распространение ошибки, генетические алгоритмы служат примерами таких методов.

Выполнение данных условий обеспечивает формирование надежных, эффективных нейросетей. Их применение возможно в широком ряде дисциплин: начиная от компьютерного зрения, переходя к обработке естественного языка, завершая машинным обучением и другими сферами.

1.6 Алгоритмы обучения

Применение разнообразных алгоритмов необходимо при обучении нейронных сетей. Существует множество таковых. Рассмотрим следующие из них:

Основным оптимизационным алгоритмом, задействованным в процессе обучения, служит градиентный спуск.

Этот метод применен для уменьшения величины функции потерь, заключающейся в модифицировании параметров модели по направлению, обратном к градиенту указанной функции потерь.

Метод, известный как обратное распространение ошибки, внедряет использование градиентного спуска.

Здесь это делается для цели минимизации расхождений между значениями, предсказанными, и значениями, фактическими, применительно к тренировочным данным.

Наиболее часто применяем в данных условиях стохастический градиентный спуск проявляет более высокую эффективность и в сравнении требует меньше вычислительных затрат, особенно на обширных данных.

Процесс обновления переменных модели осуществляется по окончании рассмотрения каждого экземпляра обучающей совокупности.

1.7 Средства, используемые для создания нейронных сетей

В процессе создания нейронной сети доступен широкий спектр инструментальных ресурсов и средств:

Наиболее популярными инструментами в арсенале машинного обучения, предназначенными для работы с нейронными сетями, выделяются такие библиотеки, как: TensorFlow, Keras, PyTorch, Caffe, MXNet, Theano. Обширный арсенал функциональных возможностей и методик, интегрированных в эти системы, обеспечивает эффективное построение, детальное обучение и качественную оценку моделей нейронных сетей.

Интегрированные среды разработки, такие как GoogleColab, PyCharm и VisualStudioCode, предоставляют возможности для создания и запуска нейронных сетей. Они обладают удобным интерфейсом, который способствует облегчению процессов отладки и разработки.

Языки программирования составляют основу для разработки нейронных сетей. К примеру: Python, Java, C++, MATLAB и R могут быть использованы в этом контексте. Самым популярным среди них выделяется Python, его применение обширно в интегрированных средах разработки и библиотеках, посвященных машинному обучению.

Применение графических процессоров (GPU) обеспечивает многократное ускорение ресурсоёмких процессов, включая вычисления и тренировку нейросетей. Рынок предлагает специализированные модели GPU для машинного обучения: NVIDIA Tesla, AMD Radeon Instinct;.

Облачные сервисы предлагают многофункциональные возможности: к ним относятся GoogleCloud, AmazonWebServices, MicrosoftAzure. Эти платформы обеспечивают необходимую инфраструктуру и инструментарий для того, чтобы вогналось создание и развертывание нейронных сетей в облачной среде.

Программное обеспечение, предназначенное для автоматизации процессов создания и конфигурирования нейронных сетей, представляет собой специализированные платформы. Использование таких инструментов способствует упрощению и автоматизации процедур построения и настройки параметров нейронных сетей. В качестве примеров таких, служат: AutoKeras, GoogleAutoML.

Средства и инструментарии при разработке нейросетей выбираются в зависимости от: условий заданной проблемы, доступных ресурсов, а также практического опыта специалиста, осуществляющего разработку.

1.8 Библиотеки обработки данных и машинного обучения

В числе инструментов, наиболее ценных и широко используемых для манипулирования данными, несомненно, занимают своё место библиотеки, известные как NumPy и Pandas.

NumPy, которую часто именуют как Numeric Python, представляет собой средство для осуществления научных вычислений в окружении Python. В арсенале этой библиотеки находится множество функций для обработки многоуровневых массивов и предоставляется широкий спектр возможностей для управления данными в них. Благодаря включению разнообразных функциональных возможностей по реализации операций линейной алгебры, созданию случайных чисел и проведению математических расчетов, NumPy способна эффективно обрабатывать массивные наборы данных [4].

Библиотека Pandas, разработанная для управления данными, располагает обширным арсеналом возможностей: начиная от организации больших массивов общественно-статистических данных и заканчивая эффективной обработкой таблиц. Заимствование данных возможно из разнообразных источников, таких как CSV, Excel, SQL, и другие. Обеспечиваются функциональность по сбору и анализу датасетов. Обработка, фильтрация, группировка и агрегация данных отличаются высокой эффективностью благодаря мощным инструментам Pandas. Вдобавок, библиотека позволяет отображать данные, что значительно упрощает их визуализацию [3].

NumPy и Pandas часто используются совместно в задачах, связанных с обработкой данных. NumPy предпочтителен для манипуляций с массивами данных, в то время как Pandas служит мощным инструментом для работы с табличными данными. Сочетание этих двух библиотек дает возможность комфортно и эффективно анализировать и обрабатывать большие объемы данных в программной среде Python.

В контексте использования языка программирования Python, двумя распространенными фреймворками для создания и обучения искусственных нейронных сетей являются: TensorFlow и Keras.

TensorFlow, являющаяся продуктом коллективного творчества команды GoogleBrain, предоставляет открытую платформу в рамках области машинного обучения. Предназначенная для создания, а также обучения

нейросетей на многообразных уровнях абстракции: платформа обладает рядом функций. К этим функциям относятся: выполнение операций с тензорами на уровне базовой сложности и более продвинутые API, способствующие разработке и тренировке модельных структур. С помощью TensorFlow доступно манипулирование данными, построение графов вычислений, а также осуществление данных вычислений при помощи различных устройств, включая, но не ограничиваясь только ими: CPU, GPU и TPU.

Keras представляет собой API высокого порядка для создания нейросетевых структур, основанный на разнообразных бэкендах, среди которых выделяется TensorFlow. Данный ресурс обеспечивает возможности построения искусственных нейронных сетей с использованием упрощенных компонент, таких как: слои, модели и методы для оптимизации. Простота интерфейса Keras способствует эффективному процессу разработки, обучения и анализа в области машинного обучения.

Во многих областях, включая: компьютерное зрение, обработку языка и глубокое изучение, находят активное использование такие инструменты, как TensorFlow и Keras. TensorFlow предлагает подход, характеризующийся более низким уровнем абстракции, что дает пользователям больший контроль над процессами создания и тренировки моделей машинного обучения. В то же время, Keras выступает с интерфейсом, который прост и доступен, что делает его особенно удобным для разработки этих моделей.

1.9 Интегрированная среда разработки

Многообразие инструментариев для конструирования нейронных сетей немало, однако рассмотрим подробно один из таких инструментов.

GoogleColab представляет собой инструмент интерактивной разработки, который предназначен для работы с кодированием в разнообразии языков программирования: в список включены Python, R, Julia. Многоцелевое использование этого инструмента охватывает создание, демонстрацию

документов, интегрирующих тексты, коды, графические решения, видеоматериали и другие элементы медиа. В сферах, где применяется GoogleColab, находятся обучение, научные исследования, разработка прототипов, компиляция и академическая публикация [1].

Основные функции GoogleColab включают:

1) задачи по созданию и редактированию документов предполагают использование таких языков программирования, как Python, R, Julia, среди прочих;

2) документ, имеющий функционал интерактивного выполнения кода, позволяет не только редактирование и запуск программного кода, но и демонстрирует результаты в виде графиков, таблиц и других медиа-элементов, показывая их непосредственно после исполнения кода;

3) настройку и корректировку документации, при этом включаются технологии инсталляции различных комплектаций и библиотек, которые предназначены для использования в машинном обучении, поддерживаются основательно;

4) интерактивную среду разработки, которая предоставляется, обеспечивая возможности для экспериментирования и частичного прототипирования кода;

5) процесс создания и обмена документами, обладающими свойствами лёгкости сохранения и возможностью публикации в сети Интернет, характеризуется простотой.

GoogleColab, отличающийся обширными возможностями, находит свое применение в множестве областей: наука о данных, обучение машин, обработка естественных языков, геномика и другие. Этот инструмент, получивший признание среди исследователей за экспериментальное программирование, обеспечивает удобную и эффективную базу для визуализации и представления результатов научных проектов.

1.10 Язык программирования

Python сохраняет свои позиции среди топовых инструментов, предназначенных для разработки и моделирования искусственных нейронных сетей. Преимущества его использования при работе с ИНС многочисленны и включают: простоту в освоении программного кода, наличие обширной библиотеки, способствующей ускорению процесса разработки, активное коммьюнити, обеспечивающее неизменную поддержку и регулярные обновления.

Доступность освоения языка Python определяет его популярность у начинающих, занимающихся машинным обучением. Лаконичная синтаксическая структура этого языка способствует простоте его использования, кроме того, он предоставляет широкий выбор библиотек и фреймворков для эффективной работы с искусственными нейронными сетями.

Python, обладая обширным арсеналом, включает в себя многие фреймворки и библиотеки. К примеру, значимые разработки: TensorFlow, Keras, PyTorch и другие. Эти инструменты облегчают задачи, связанные с разработкой и обучением нейросетей.

Python демонстрирует гибкость: создание настраиваемых нейросетей, использование различных стратегий оптимизации возможно, содействующих повышению эффективности.

Python демонстрирует высокую степень согласованности и интеграции с многообразием инструментов и программируемых языков. Средства и языки как SQL, Hadoop, Spark, а также другие множества, примером служат.

Среди языков программирования Python выделяется благодаря своему широкому сообществу разработчиков. Они обеспечивают разработку и поддержку колоссального количества библиотек и фреймворков, особенно значимых в таких сферах, как машинное обучение и технологии нейронных сетей. Это обстоятельство сопровождается наличием обширной документации и глубокой всесторонней поддержкой касаемо различных проблем и вопросов.

Программирование на Python распространяется на обширный ареал типов данных: таких как тексты, изображения, звуковые файлы, среди прочих форм данных. Гибкость этого языка способствует успешному созданию нейросетей для множества задач.

Выгоды использования Python для разработки нейросетей и применения методик машинного обучения многочисленны: простота в эксплуатации, гибкость функционала и эффективность в интеграции с разнообразными инструментальными средствами, что добавляет значимости данному языку программирования.

2 Задачи машинного обучения

2.1 Задача NLP

NaturalLanguageProcessing (NLP) относится к числу высоко востребованных и часто решаемых задач: её сфера - машинное обучение, направлено на анализ, обработку естественного языка. Применение NLP - разработка систем, предназначенных для взаимодействия с людьми на их родном языке, понимания текстовых данных, экстракции значимой информации из текстов.

В рамках NLP, используются различные методы и алгоритмы, такие как:

1) токенизация– разделение текста на отдельные слова и символы;

2) стемминг представляет собой процесс выявления основы слова, что способствует уменьшению объема словаря;

3) лемматизация– процесс приведения слова к его базовой форме и другие.

Область применения NLP обширна и многофункциональна. В сфере машинного обучения NLP находит использование в задачах анализа и обработки текстов большого объема: отзывы, новостные сообщения, твиты и другие подобные данные подлежат изучению. Кроме того, осуществляется использование NLP при создании персональных ассистентов, способных отвечать на запросы и проводить различные действия, базирующиеся на принципах естественного языка. Широкое применение NLP также просматривается в анализе текстов в таких областях, как медицина, финансы и юриспруденция.

Процесс обработки неструктурированных данных, включая тексты и изображения, представляет собой важнейший аспект в пространстве NLP. Эта задача заявляет требования к глубоким познаниям и прошлым достижениям в сферах машинного обучения и анализа данных. Корректно подобранный метод внедрения NLP может существенно усилить точность и эффективность

анализа текстов, а также способствует созданию систем взаимодействия, адаптированных к потребностям пользователя.

2.2 Задача классификации

В рамках машинного обучения классификация является процессом, когда модель устанавливает принадлежность каждого элемента из определённого ассортимента к одной из заранее заданных категорий. Задание подразумевает конструирование алгоритма, способного определять классификацию объектов по их атрибутам.

Примерами задач классификации могут служить:

1) классификация электронных писем на спам и не спам;

2) выполнение задачи распределения изображений по различным категориям: в частности, «кошки», «собаки» и «автомобили»;

3) разделение отзывов по изделиям в зависимости их тональности: возможное указание может быть позитивным, негативным или нейтралитетным;

4) классификация финансовых транзакций исследуется чрез разделение их на категории: мошеннические и не мошеннические.

В задачах классификации как принято, используемый набор данных объединяет разнообразие объектов, каждый из которых оснащен набором специфических характеристик: размером, цветом, текстурой, атрибутами прочими. Определённые данные для каждого объекта относятся к одному из заранее определённых классов.

В рамках решения задачи классификации используются многие подходы машинного обучения, к примеру, следующие:

1) логистическая регрессия — методика, применяемая для определения вероятности принадлежности объекта к конкретному классу;

2) решающие деревья — метод, подразделяющий данные на уменьшенные группы, где каждая из них ассоциируется с определённым классом;

3) k-ближайших соседей;

4) метод опорных векторов — это техника обучения, нацеленная на выявление гиперплоскости, обеспечивающей оптимальное разделение объектов различных категорий;

5) нейронные сети представляют собой методику, применяемую для классификации объектов с помощью многоуровневых структур нейронов.

Для выбора оптимального метода классификации необходимо учитывать специфику задачи, размер и характеристики набора данных, а также доступные ресурсы и время.

2.3 Задача кластеризации

В области машинного обучения задача кластеризации, которая иногда встречается и в рамках обработки естественного языка (NLP), заключается в том, что определённое множество объектов должно быть разделено на подмножества: кластеры, что не перекрываются между собой. Сущность этой задачи такова, что каждый кластер объединяет объекты, обладающие сходством по конкретному признаку, при этом гарантируется, что объекты различаются между разными кластерами.

Данная задача относится к задачам обучения без учителя.

Для решения задачи кластеризации можно использовать следующие методы:

1) метод k-средних характеризуется повторным вычислением центров масс для кластеров, сформированных на предшествующем этапе, используемым в каждой новой итерации;

2) методы инициализации охватывают различные подходы: в числе которых находится метод Forgy, случайное разбиение и алгоритм k-means++;

3) пространственная кластеризация DBSCAN, основанная на плотности, предназначена для использования в приложениях с шумовыми данными;

4) техника, основанная на «передаче сообщений» меж узлами, представляет собой метод распространения близости.

Правильный выбор метода кластеризации требует тщательного анализа поставленной задачи, учета ресурсов и затрат времени, имеющихся в распоряжении.

3 РЕАЛИЗАЦИЯ СИСТЕМЫ КЛАССИФИКАЦИИ

3.1 Постановка задачи

В данной работе главной задачей является разработка корректно работающей системой классификации новостных сообщений с дальнейшим выявлением среди них фейков и кластеризации новостей по семантическому признаку, чтобы понять, подходят ли они для текущего анализа или нет, для русскоязычного сегмента.

Задача была разделена на несколько шагов:

1) сбор данных на русском языке;
2) препроцессинг входных данных;
3) реализация классификатора;
4) реализация кластеризатора.

Задача должна удовлетворять следующим требованиям:

1) модель должна быть обучаемой;
2) модель должна предоставлять результат;
3) модель должна обладать функциональной полнотой по отношению к поставленным задачам.

3.2 Программное обеспечение для реализации

Для разработки используются следующие программы, языки программирования и дополнительные решения, упрощающие работу:

GoogleColab – легкая и удобная в использовании интерактивная среда разработки.

Python – основной язык для написания нейронной сети, в работе используется как база в силу своей простоты и содержания в себе необходимых библиотек и модулей.

Matplotlib – библиотека Python, открывающая функционал графического представления данных.

Seaborn – вспомогательная библиотека, некоторые графики проще рассматривать, благодаря ей.

Nltk –библиотека, используемая для предобработки информации, а именно содержит в себе список стоп-слов.

Re – библиотека, позволяющая обрабатывать строки в Python с помощью множества функций.

PyMorphy2 – библиотека, позволяющая произвести лемматизацию слов.

Numpy – удобная библиотека, предоставляющая возможность работать с массивами, матрицами и содержащая в себе множество полезных функций и методов.

Pandas – необходимая библиотека для анализа и обработки входных данных.

Sklearn – полезная библиотека для работы с выборками данных для нейронной сети [6].

Gensim – библиотека, содержащая в себе возможности векторизации текста.

3.3 Программная реализация

Из условий постановки задачи можно сделать вывод, что она является задачей обработки естественного языка. Соответственно необходимо собрать требующиеся данные. Они представляют из себя текстовые новостные сообщения, при этом для первой части задачи, а именно классификатора, необходимо, чтобы они были промаркированы, в зависимости от того, является новостное сообщение фейком или нет, так как предполагается обучение с учителем, в то время как для второй части задачи, – кластеризатора, подобная маркировка не требуется, потому что обучение проходит без учителя.

Таким образом собираются соответствующие данные в формате CSV. Comma-Separated Values является популярным форматом хранения данных. Его используют для представления табличных данных в виде текстового файла. Исходя из названия, он состоит из строк, каждая из которых представляет собой запись, а поля внутри строки разделены запятыми. Разделитель можно выбрать любой, запятая – тривиальный случай, также можно использовать, например, точку с запятой или табуляцию.

Выбранный формат имеет некоторые преимущества перед хранением данных в формате. Например, формат EXCEL, может автоматически выполнять следующие операции:

1) округление чисел;
2) приведение к экспоненциальной форме;
3) удаление лидирующих плюсов;
4) разбиение больших групп чисел на трехзначные группы;
5) удаление лидирующих нулей;
6) изменение дат под локальные настройки.

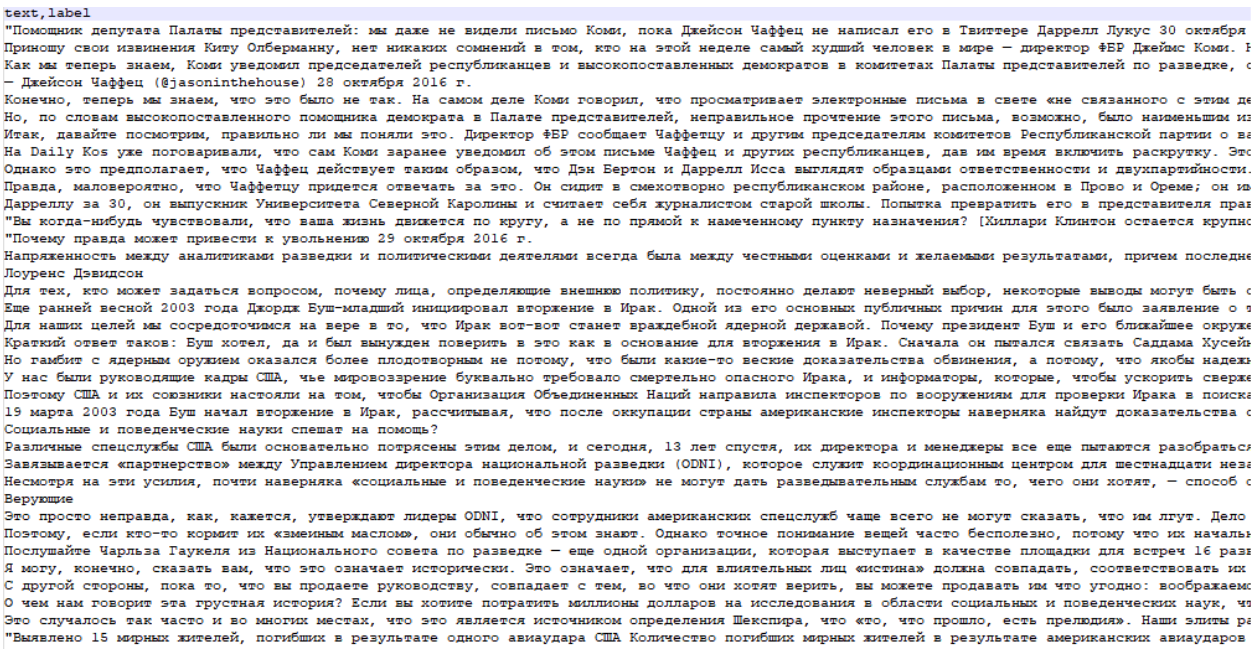


Рисунок 3 – Первый датасет

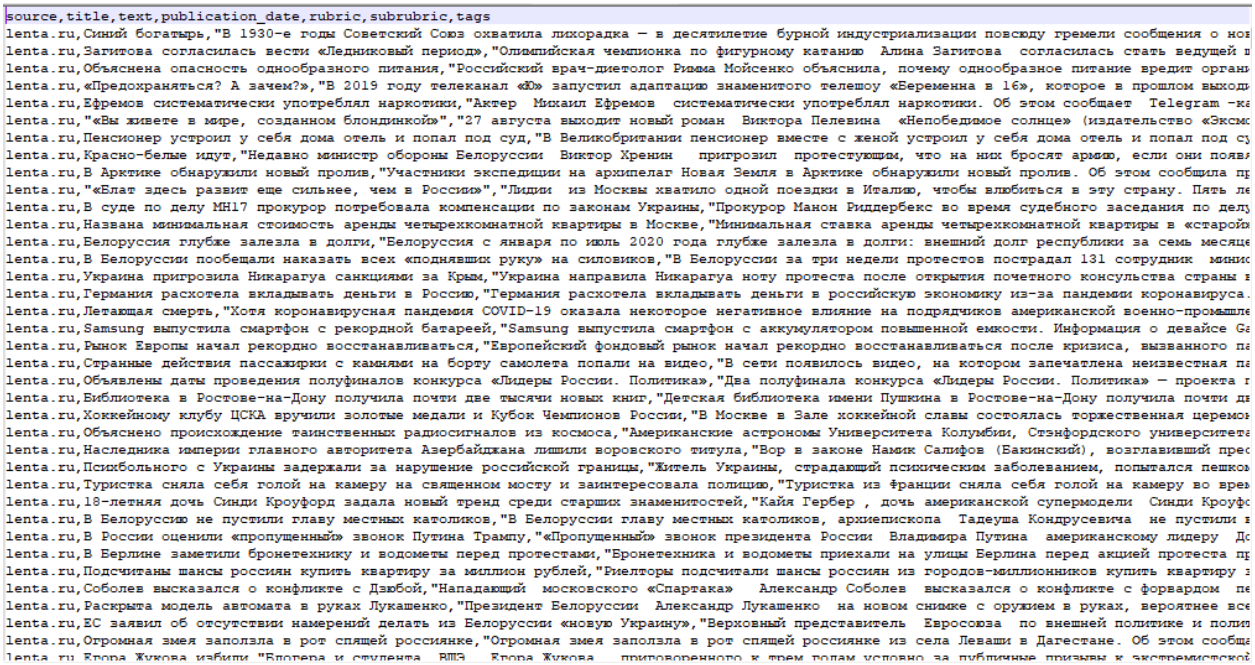
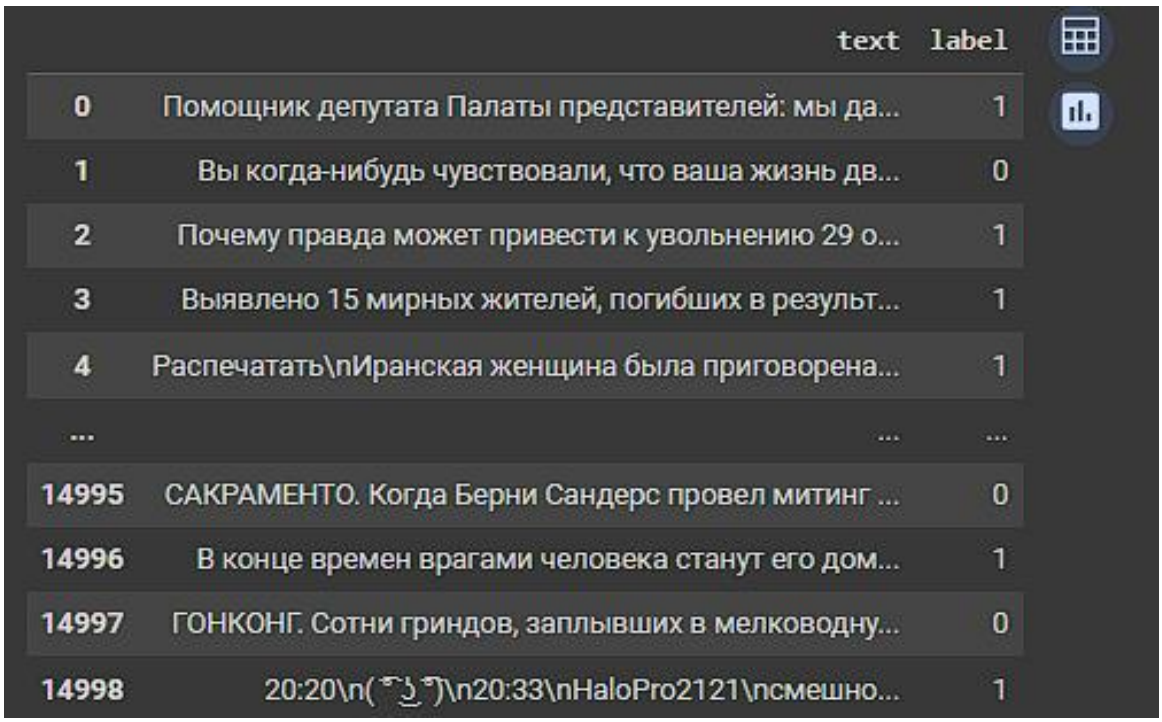


Рисунок 4 – Второй датасет

Далее обращаемся в выбранную нами среду разработки, – GoogleColab, создаем новый блокнот и загружаем к нему файлы, преждевременно дав им названия.

После загрузки происходит импортирование всех необходимых библиотек. С помощью библиотеки Pandas считываем данные из датасета, собранного для первой задачи. Он будет переведен из формата CSV в объект DataFrame, в котором хранятся табличные данные. Также появится корректное отображение символов. Теперь датасет выглядит так:



	text	label
0	Помощник депутата Палаты представителей: мы да...	1
1	Вы когда-нибудь чувствовали, что ваша жизнь дв...	0
2	Почему правда может привести к увольнению 29 о...	1
3	Выявлено 15 мирных жителей, погибших в результ...	1
4	Распечатать\nИранская женщина была приговорена...	1
...	...	...
14995	САКРАМЕНТО. Когда Берни Сандерс провел митинг ...	0
14996	В конце времен врагами человека станут его дом...	1
14997	ГОНКОНГ. Сотни гриндов, заплывших в мелководну...	0
14998	20:20\n(ツ)\n20:33\nHaloPro2121\nсмешно...	1

Рисунок 5 – Первый датасет в объекте DataFrame

На иллюстрации видно, что новостные сообщения промаркированы единицей или нулем. Здесь, соответственно:

1) «1» – правдивое новостное сообщение;

2) «0» – фейковое новостное сообщение;

Следующим шагом является описание функции, которая будет осуществлять препроцессинг подготовленных новостей в обоих частях задачи.

Для этого используются объекты и методы из библиотек re, nltk, pymorphy2 [5].

Порядок действий описываемой функции, следующий:

1) текст преобразуется в нижний регистр, данная процедура делается для того, чтобы не было необходимости учитывать различия между заглавными и строчными буквами в дальнейшей обработке или при последующем анализе текста;

2) замена символов, не являющихся буквами или цифрами, поиск таких символов производится с помощью регулярного выражения (`[^a-zA-Za-яА-Я0-9\s]`). Поиск производится среди латинского и кириллического

алфавитов, а также набора цифр; такая обработка позволяет избавиться от пунктуации, специальных символов и символов других алфавитов;

3) токенизация текста – текст разбивается на отдельные слова и по-отдельности каждое вносится в специально созданный для этого список;

4) фильтрация стоп-слов, из библиотеки nltk загружается список стоп-слов, с пометкой `russian`, что означает, что необходимо загрузить пакет именно для русского языка; стоп-слова включают в себя союзы, предлоги, частицы междометия, местоимения. Затем происходит фильтрация слов из списка, подготовленного в предыдущем пункте;

5) лемматизация слов, для этого используется объект MorphAnalyzer() из библиотеки pymorphy2, который используется для приведения слов к нормальной форме (лемматизация), каждое слово подвергается морфологическому анализу и извлекается его нормальная форма;

6) сборка обработанного текста – лемматизированные слова объединяются в одну строку, разделенную пробелами для получения окончательного результата препроцессинга.

Таким образом, после процедуры предобработки, информация приобретает следующий вид:

	text	label	clean_text
0	Помощник депутата Палаты представителей: мы да...	1	помощник депутат палата представитель видеть п...
1	Вы когда-нибудь чувствовали, что ваша жизнь дв...	0	когданибудь чувствовать ваш жизнь двигаться кр...
2	Почему правда может привести к увольнению 29 о...	1	почему правда привести увольнение 29 октябрь 2...
3	Выявлено 15 мирных жителей, погибших в результ...	1	выявить 15 мирный житель погибнуть результат о...
4	Распечатать\nИранская женщина была приговорена...	1	распечатать иранский женщина приговорить шесть...
...	...	...	...
14995	САКРАМЕНТО. Когда Берни Сандерс провел митинг ...	0	сакраменто бернуть сандерс провести митинг отк...
14996	В конце времен врагами человека станут его дом...	1	конец время враг человек стать домочадец убить...
14997	ГОНКОНГ. Сотни гриндов, заплывших в мелководну...	0	гонконг сотня гринд заплыть мелководный новозе...
14998	20:20\n(*ʖ*)\n20:33\nHaloPro2121\nсмешно...	1	2020 2033 halopro2121 смешной 2108 лягушонок п...
14999	(Хотите получить этот брифинг по электронной п...	0	хотеть получить брифинг электронный почта добр...

Рисунок 6 – Информация, прошедшая предобработку

Очистив текст от лишней информации, необходимо векторизовать текст. Эта процедура проводится для того, чтобы подготовить текстовые данные для дальнейших действий. Векторизация позволяет преобразовать наборы слов или документов в числовое представление, которое может обрабатываться с помощью алгоритмов машинного обучения. Классификация работает только с числовыми данными, поэтому такая операция является обязательным шагом в задаче.

Векторизация проводится с помощью объекта CountVectorizer, хранящегося в библиотеке scikit-learn. Суть векторизации таким методом заключается в преобразовании набора текстовых документов в матрицу токенов (уникальных слов). В CountVectorizer последовательно выполняются действия:

1) токенизация текста, используется классический токенизатор, который разделяет слова по пробелам;

2) построение словаря, строится словарь уникальных слов в наборе данных; каждому уникальному слову присваивается индекс в словаре;

3) подсчет частоты слов, для каждого текстового документа подсчитывается количество вхождений каждого слова из словаря; таким образом, документ представляется в виде вектора, где каждый элемент соответствует количеству вхождений;

4) представление в виде матрицы, после подсчета формируется матрица, в которой в столбцах записаны уникальные слова из словаря, а в строках – тексты.

После процесса векторизации датасет разделяет на две выборки тестовую и тренировочную. Такое разделение проводится с помощью функции train_test_split, которая также хранится в пакете scikit-learn. Данные делятся в пропорции 30/70, где 30% - тестовая выборка, а 70% - тренировочный набор.

Затем выбирается модель, с помощью которой будет проводиться классификация данных. Используется модель логистической регрессии. В данной задаче она показывает наилучшие значения среди других методов машинного обучения, в этом можно будет убедиться с помощью кросс-валидации. Также она имеет определенные плюсы, которые обращают выбор в ее пользу:

1) эффективность на больших данных;
2) низкая подверженность переобучению;
3) вычислительная эффективность;
4) высокая эффективность при бинарной классификации.

Логистическая регрессия работает по следующему принципу. Сама регрессия основывается на линейной модели, которая вычисляет взвешенную сумму входных признаков:

$$z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n , \quad (1)$$

где

β_0 – свободный член,

β_i – коэффициенты признаков x_i.

Линейная модель преобразуется в вероятность с помощью сигмоидной функции, которая ограничивает значения в интервале от 0 до 1:

$$\sigma(z) = \frac{1}{1+e^{-z}} \quad (2)$$

Сигмоидная функция возвращает значение, интерпретируемое как вероятность принадлежности объекта к положительному классу.

На основе полученной вероятности принимается решение о предсказании класса:

$$y = \begin{cases} 1, \text{ если } \sigma(z) \geq 0.5 \\ 0, \text{ если } \sigma(z) < 0.5 \end{cases} \quad (3)$$

Для обучения модели логистической регрессии используется функция потерь, называемой логистической функцией правдоподобия:

$$\mathcal{L} = -\frac{1}{m}\sum_{i=1}^{m}[u_i \log(y_i) + (1-u_i)\log(1-y_i)], \quad (4)$$

где

m – число объектов в обучающей выборке,

u_i – истинная метка класса,

y_i – предсказанная вероятность.

Коэффициенты модели оптимизируются одним из методов численной оптимизации, с целью минимизации функции потерь. Этот процесс продолжается, до тех пор, пока функция потерь не достигнет минимума или не будет достигнуто определенное количество итераций.

Модель обучается на тренировочной выборке данных и проверяется на тестовой. Оценочные мерки accuracy, recall, precision, f1 и матрица ошибок хранятся также в библиотеке scikit-learn. Данные метрики помогают оценить производительность модели.

1) accuracy (точность) показывает, как часто модель правильно предсказывает класс объекта;

2) precision (точность) показывает долю правильно предсказанных положительных объектов относительно всех объектов, предсказанных как положительные;

3) recall (полнота) измеряет долю правильно предсказанных объектов среди всех реальных положительных объектов;

4) f1-score (f1-мера) представляет собой среднее между precision и recall;

5) confusion matrix (матрица ошибок) – это таблица, которая позволяет визуализировать количество верно и ошибочно классифицированных примеров для каждого класса.

Результаты обучения, следующие:

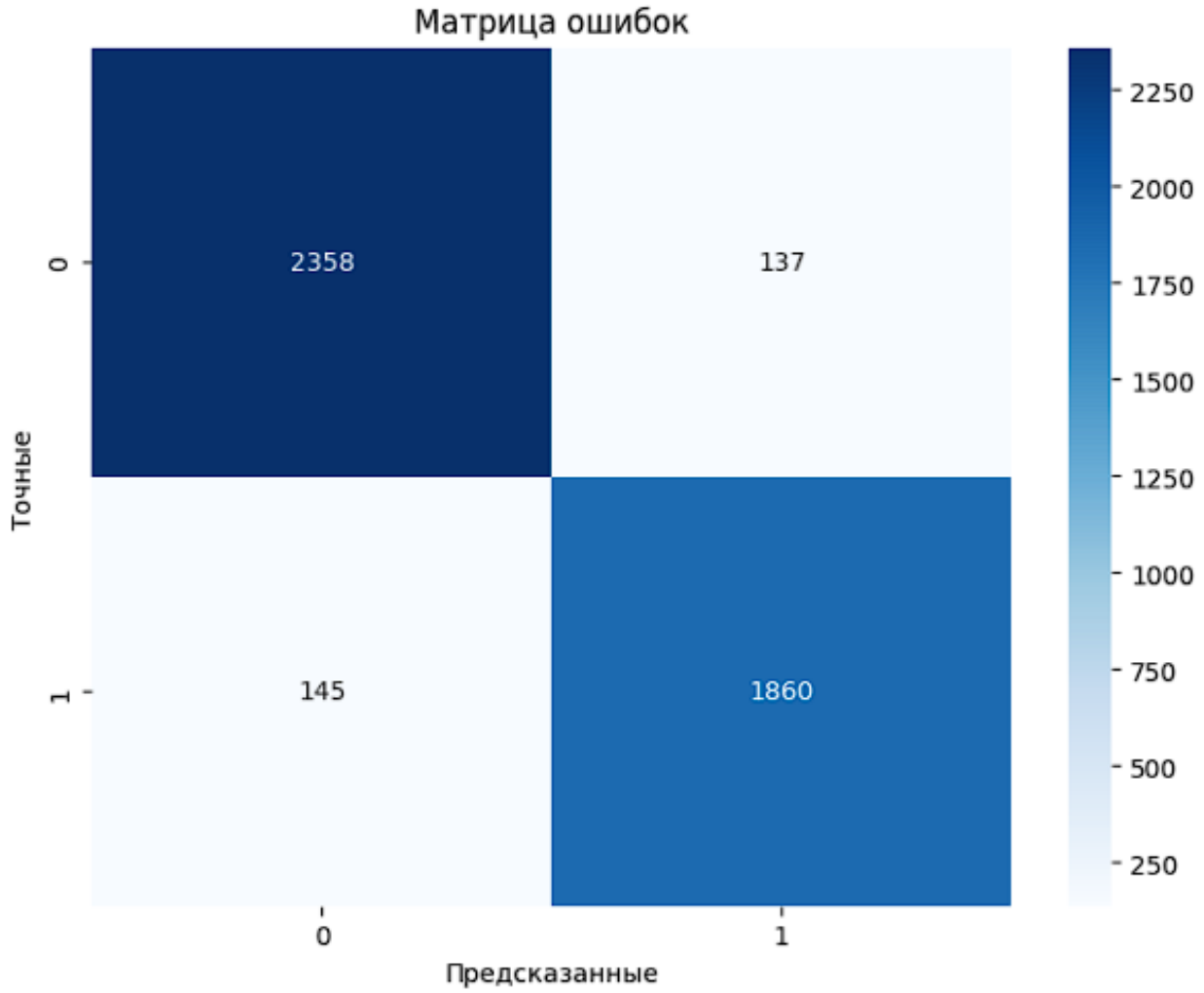


Рисунок 7 – Матрица ошибок классификации

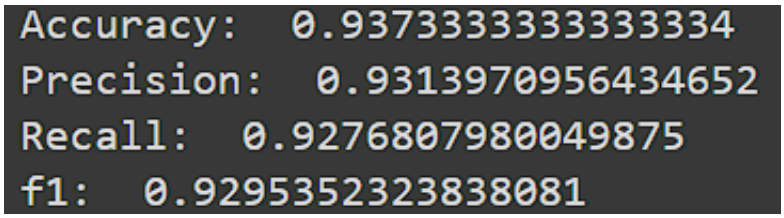


Рисунок 8 – Оценочные метрики модели

В матрице ошибок видно, что правильно предсказанных примеров – 4218, в то время как неверно предсказанных – 282. В совокупности с высокими показателями оценочных метрик это дает понять, что модель не переобучена и показала хорошие результаты.

Теперь необходимо проверить модель с помощью кросс-валидации. Она является важным методом оценки производительности модели. Ее применяют для того, чтобы проверить работоспособность модели на различных выборках. Сначала задается количество наборов *k*, на которых будет проводиться кросс-валидация, затем модель обучается на *k-1* наборе. Обучение повторяется *k* раз с сопутствующей оценкой производительности. В конце значения усредняются.

Таким образом результаты кросс-валидации:

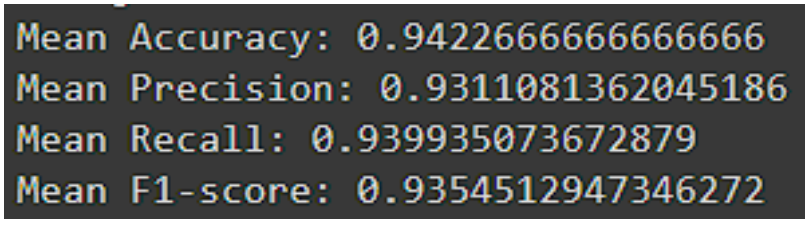
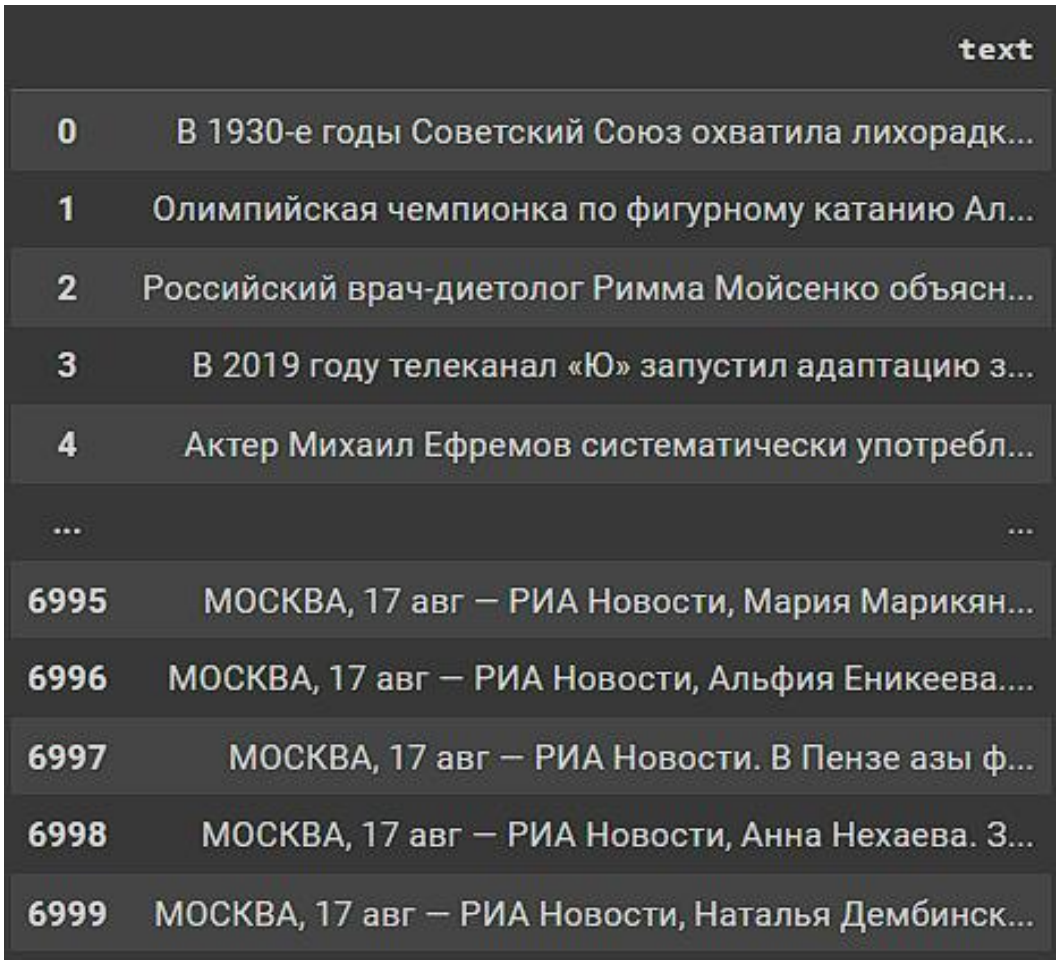


Рисунок 9 – Результаты кросс-валидации

Видно, что результаты кросс-валидации высоки, что означает, что модель действительно не переобучена, функционирует исправно и при этом способна работать с различными наборами данных.

Далее в объект DataFrame считывается второй датасет, предназначенный для кластеризации.

Обучение здесь предполагается без учителя, поэтому никакой дополнительной маркировки для текстов нет. После считывания датасет выглядит следующим образом:



	text
0	В 1930-е годы Советский Союз охватила лихорадк...
1	Олимпийская чемпионка по фигурному катанию Ал...
2	Российский врач-диетолог Римма Мойсенко объясн...
3	В 2019 году телеканал «Ю» запустил адаптацию з...
4	Актер Михаил Ефремов систематически употребл...
...	...
6995	МОСКВА, 17 авг — РИА Новости, Мария Марикян...
6996	МОСКВА, 17 авг — РИА Новости, Альфия Еникеева....
6997	МОСКВА, 17 авг — РИА Новости. В Пензе азы ф...
6998	МОСКВА, 17 авг — РИА Новости, Анна Нехаева. З...
6999	МОСКВА, 17 авг — РИА Новости, Наталья Дембинск...

Рисунок 10 – Второй датасет в объекте DataFrame

Этот набор данных также должен пройти этап предобработки перед последующим использованием в кластеризации. Аналогичным образом в нем текст приводится к нижнему регистру, спецсимволы удаляются, происходит токенизация, фильтрация стоп-слов и лемматизация. После таких действий датасет принимает следующий вид:

	text	clean_text
0	В 1930-е годы Советский Союз охватила лихорадк...	1930 год советский союз охватить лихорадка дес...
1	Олимпийская чемпионка по фигурному катанию Ал...	олимпийский чемпионка фигурный катание алина з...
2	Российский врач-диетолог Римма Мойсенко объясн...	российский врачдиетолог римма мойсенко объясни...
3	В 2019 году телеканал «Ю» запустил адаптацию з...	2019 год телеканал ю запустить адаптация знаме...
4	Актер Михаил Ефремов систематически употребл...	актёр михаил ефрем систематически употреблять ...
...	...	...
6995	МОСКВА, 17 авг — РИА Новости, Мария Марикян...	москва 17 авг риа новость мария марикян захари...
6996	МОСКВА, 17 авг — РИА Новости, Альфия Еникеева....	москва 17 авг риа новость альфий еникеев вирус...
6997	МОСКВА, 17 авг — РИА Новости. В Пензе азы ф...	москва 17 авг риа новость пенза аз финансовый ...
6998	МОСКВА, 17 авг — РИА Новости, Анна Нехаева. З...	москва 17 авг риа новость анна нехаева запомин...
6999	МОСКВА, 17 авг — РИА Новости, Наталья Дембинск...	москва 17 авг риа новость наталья дембинскай м...

Рисунок 11 – Второй датасет после предобработки

Далее тексты вновь должны пройти процесс векторизации. Для второй части задачи документы векторизуются методом TF-IDF векторизации. Данный метод используется для преобразования коллекции текстовых документов в числовой формат, при этом учитывается важность слов в доку ментах по сравнению с общим набором. Этот метод работает по следующему принципу:

Term Frequency (частота термина). TF измеряет, насколько часто слово встречается в документе. Вычисляется как отношение числа вхождений слова к общему числу слов в тексте. Чем чаще слово встречается в тексте, тем выше его показатель TF;

Inverse Document Frequency (обратная частота документа). IDF измеряет важность слова в контексте всех текстов. Он уменьшает вес слова, которое часто встречается во всех текстах и увеличивает вес слова, которое редко встречается. IDF вычисляется как логарифм отношения общего числа документов к числу документов, содержащих данный термин;

TF-IDF. Для каждого слова в текстовом документе вычисляется его TF-IDF вес. Он вычисляется как произведение TF и IDF. После таких вычислений для всех текстов, каждый из них представляется в виде вектора TF-IDF

значений слова, которые составляют документ, при этом размерность вектора соответствует размеру словаря.

Такая процедура проводится не только со вторым датасетом, но и с первым, так как новые документы будут анализироваться относительно старых: подходят ли они по смыслу текстам первого набора данных.

После этого выбирается количество кластеров, на которые модель разделит данные, а также выбирается метод, по которому кластеризация будет проводиться. Выбирается, например, 3 кластера и метод k-средних.

Метод k-средних относится к методам неконтролируемого обучения и используется для разделения набора данных на кластеры на основе их признаков. Последовательность действий метода, следующая:

1) инициализация центров кластеров, алгоритм случайным образом инициализирует центры кластеров в пространстве признаков, центры кластеров представляют собой точки данных из набора данных;

2) присваивание объектов кластерам, для каждого объекта в наборе данных вычисляется расстояние до каждого центра кластера, объект присваивается к кластеру с ближайшим центром на основе выбранной метрики расстояния, чаще всего используется евклидово расстояние;

3) пересчет центров кластеров, после присвоения всех объектов кластерам пересчитываются центры каждого кластера, новый центр кластера вычисляется как среднее арифметическое значение всех объектов, принадлежащих данному кластеру;

4) повторение шагов 2 и 3, пока изменения в центрах кластеров или принадлежности объектов к кластерам становятся незначительными или достигается максимальное количество итераций.

В процесс необходимо добавить вычисление степени схожести текстов, чтобы понять, похожи ли текста семантически друг другу в общем. Алгоритм вычисляющей функции работает на методе косинусного расстояния и заключается в следующих шагах:

1) создается список для данных текста из второго датасета; для сравнения набора слов используется словарь wv из обученной модели Word2Vec, если слово из текста присутствует в словаре, то его вектор добавляется в список;

2) вычисление среднего значения для списка, – если список из предыдущего пункта является не пустым, то вычисляется среднее значение всех векторов в нем, чтобы получить вектор представления текстов;

3) шаг 1 повторяется для первого датасета;

4) вычисление косинусного сходства между векторами, полученных в результате 1 и 3 шага;

5) функция возвращает результат от 0 до 1, где более высокое значение указывает на более близкое сходство между текстом второго датасета и текстом первого датасета.

В результате запуска функции с приведенными выше шагами получается следующий результат:

```
Оценка схожести: 0.8439300656318665
```

Рисунок 12 – Результат оценки схожести

Результат близок к единице и потому является хорошим результатом. Оставшиеся 16% необнаруженной схожести будут видны на графике.

Поиск признаков, определяющих положение кластеров также проводится с помощью метода k-means и в результате получаем следующие определяющие признаки.

В то же время система распределила объекты на кластеры следующим образом:

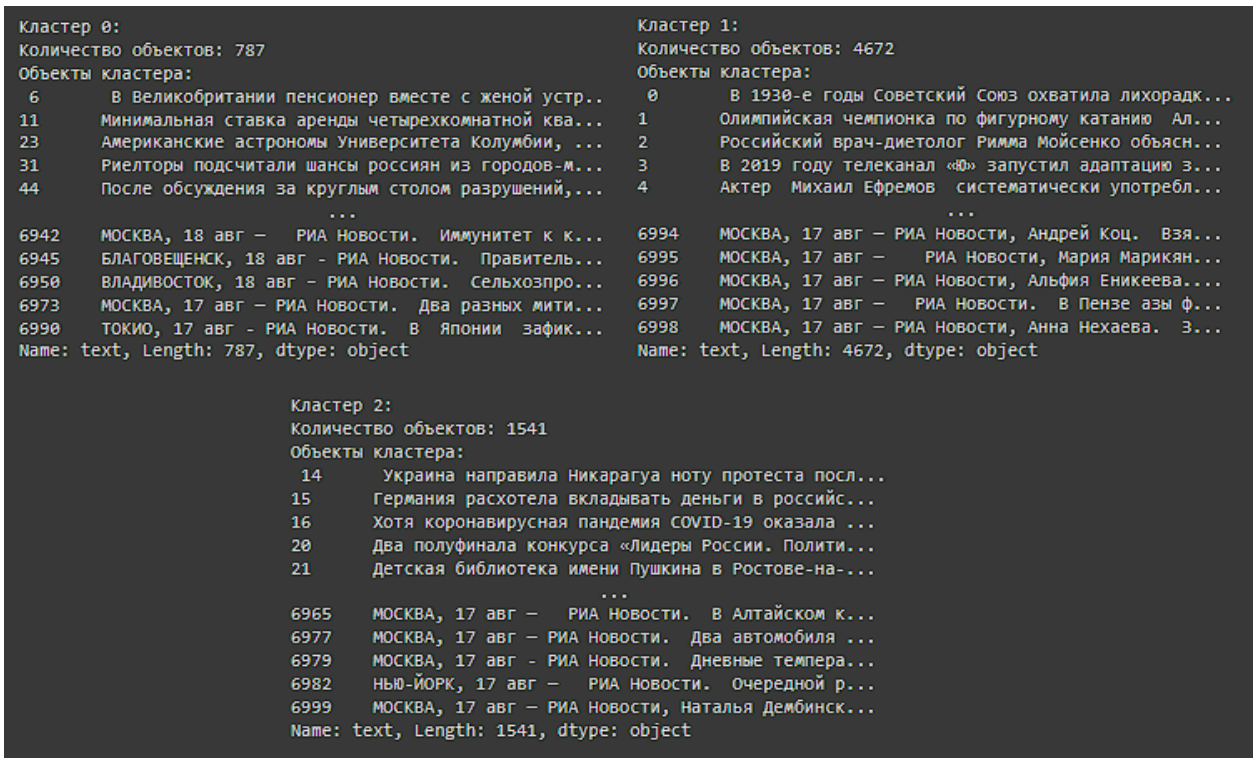


Рисунок 13 – Объекты, распределенные по кластерам

Последним этапом является графическое изображение полученных результатов. Так будет более наглядно показано, насколько корректно распределились данные по кластерам, есть ли выбросы и смешивание данных.

Для выбранных трех кластеров рисунок [7] получается следующим:

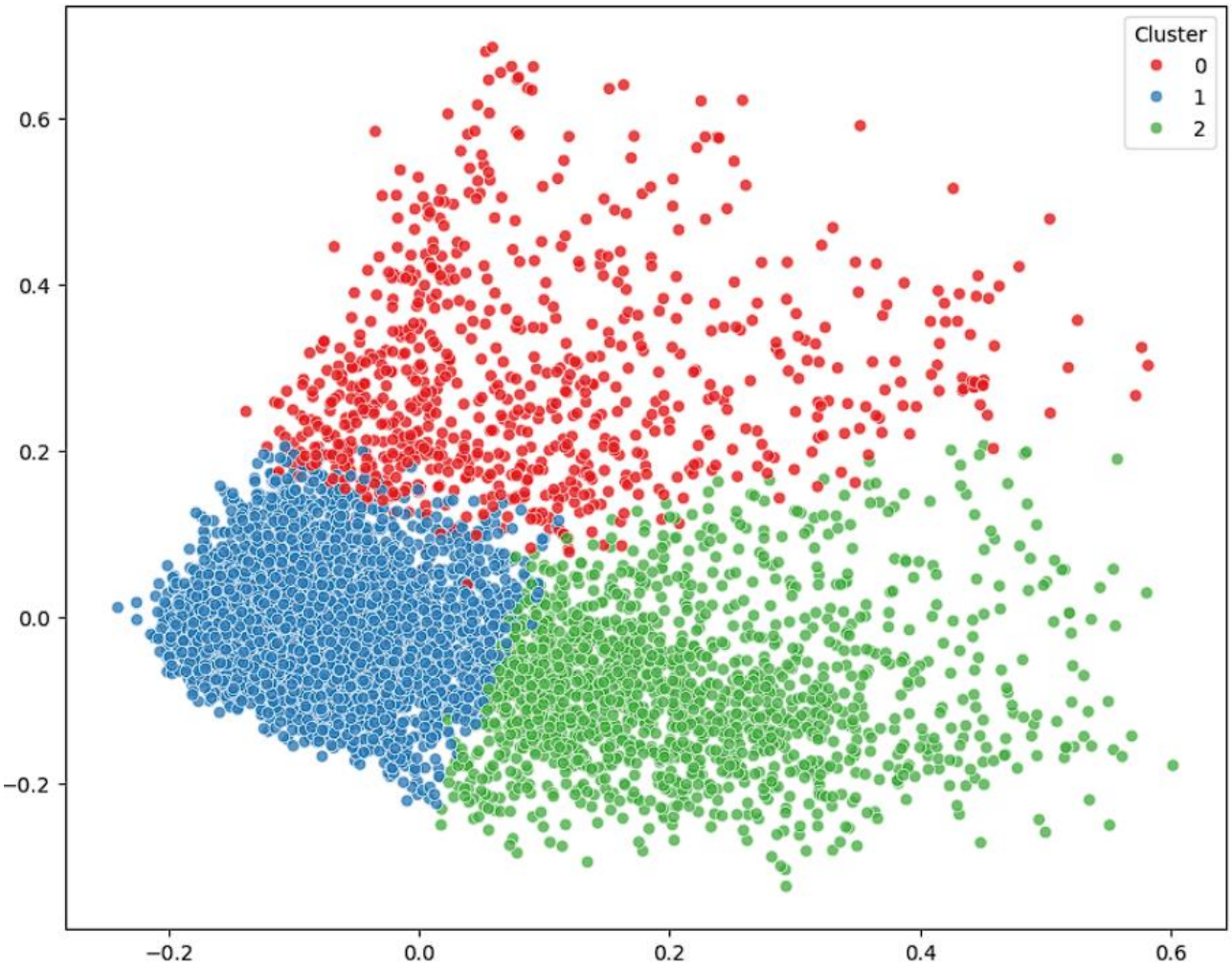


Рисунок 14 – Распределение данных по трем кластерам

На рисунке видно, что в кластерах 0 и 2 имеются выбросы, но между собой сильного смешивания у них нет. Ситуация со смешиванием данных аналогична между всеми кластерами на рисунке. Как раз это и объясняется 16% несхожести данных. Критерием смешивания данных является то, что в кластерах находятся одинаковые слова, например «российский», «человек», «год».

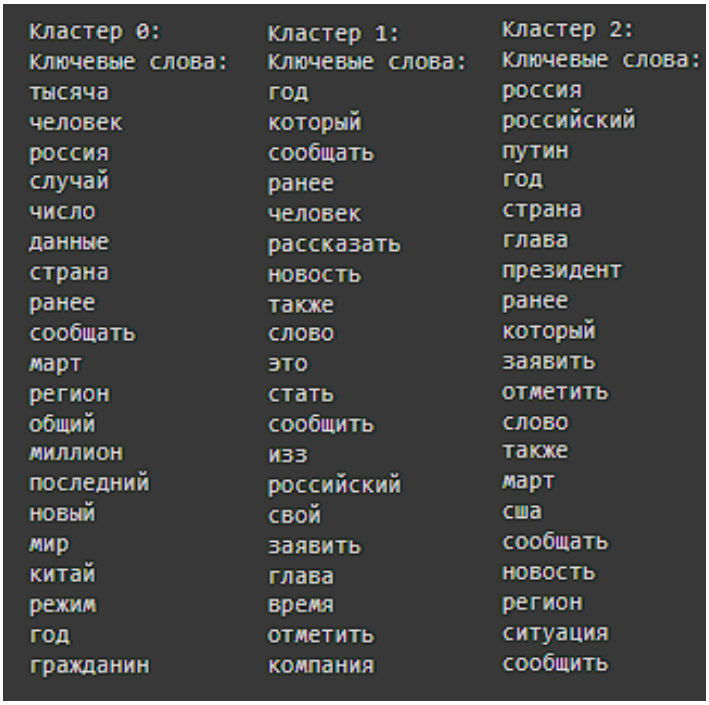


Рисунок 15 – Признаки кластеров

Выбор трех кластеров является оптимальным вариантом, так как при выборе четырех, пяти или больше кластеров происходит сильное смешивание данных.

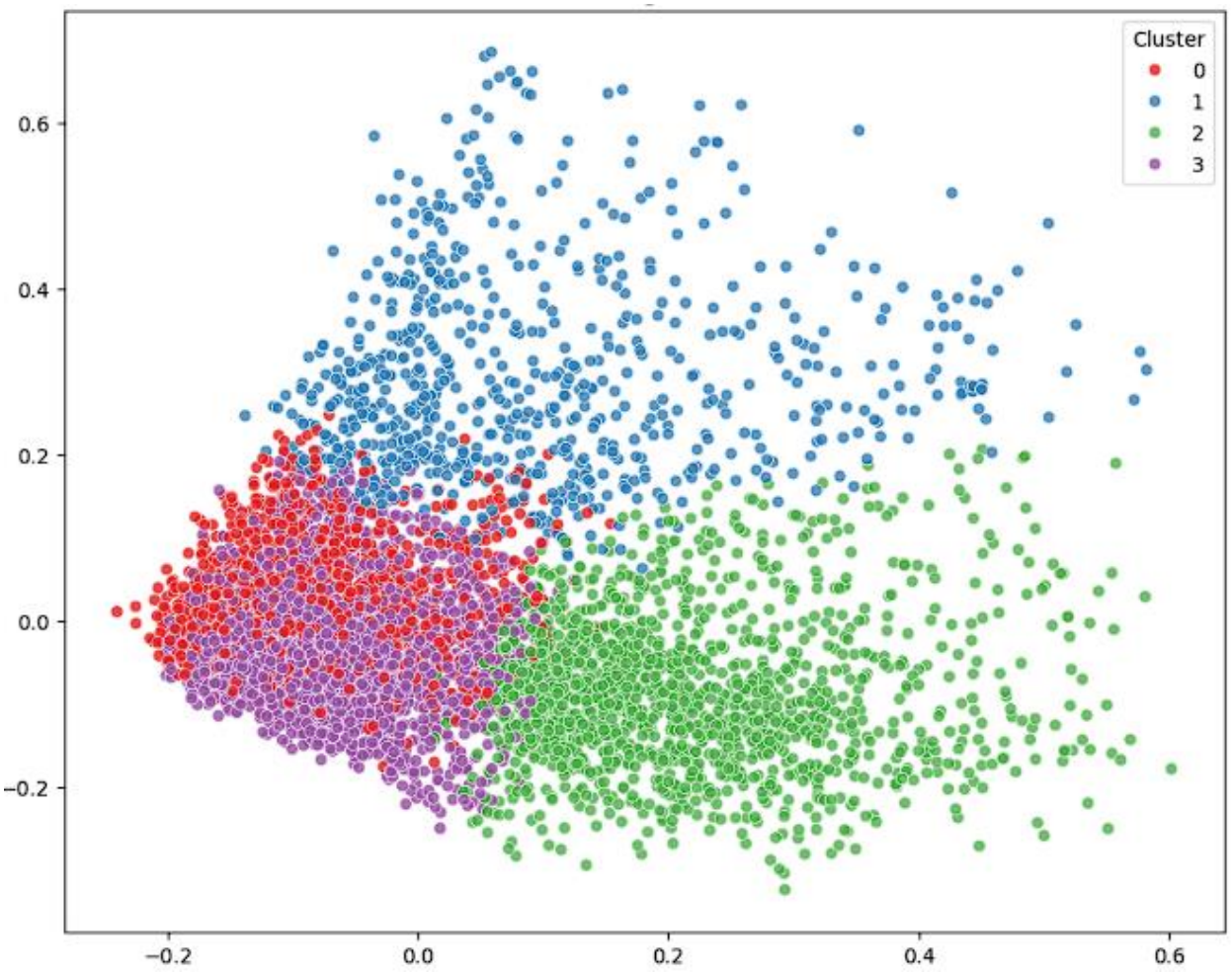


Рисунок 16 – Распределение данных по четырем кластерам

На рисунке видно, что алгоритм семантического сходства нашел некоторую тему в текстах (3 кластер), являющуюся ниже по иерархии чем тема кластера 0. При этом, смешивание данных 3 кластера с кластером 2 происходит сильнее чем смешивание кластера 0 с кластером 2. Аналогично смешивание кластера 0 с кластером 1 намного сильнее чем при варианте с тремя кластерами, в то время как кластеры 1 и 3 практически не пересекаются.

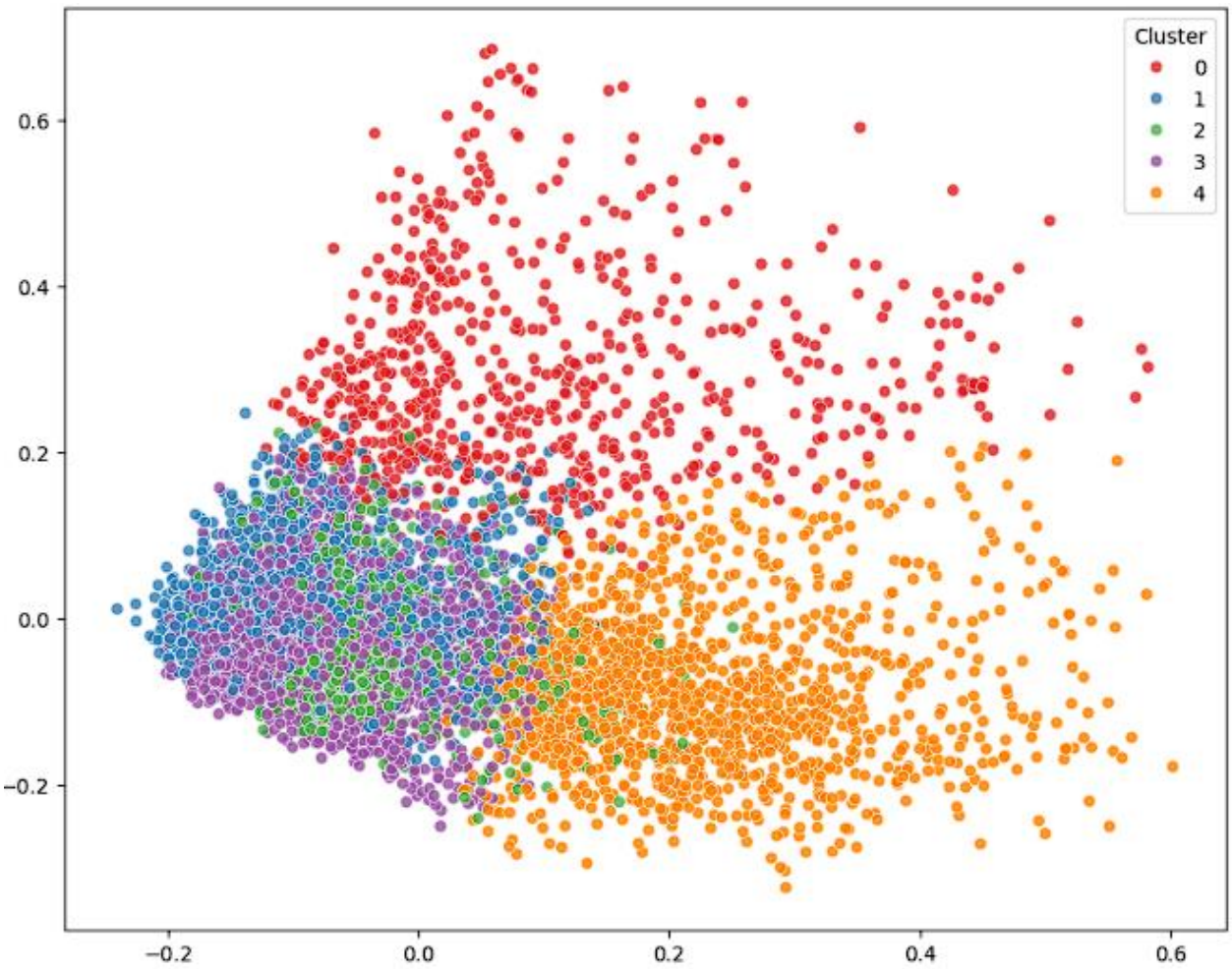


Рисунок 17 – Распределение данных по пяти кластерам

На данном рисунке видно, что алгоритм нашел еще одну подтему, при этом замешивание происходит еще сильнее: кластер 2 с большими выбросами смешивается с кластером 4, но кластеры 1,2,3 особо своей смешиваемости с кластером 0 не поменяли.

ЗАКЛЮЧЕНИЕ

Разработанная в ходе выпускной квалификационной работы система классификации текстовой информации позволяет классифицировать тексты для выявления среди них фейков, а также кластеризовать их по семантическому сходству.

В процессе изучения задачи был проведен обзор порядка построения нейронной сети, выделены её ключевые особенности, позволяющие получить более точный и удовлетворяющий результат.

Были рассмотрены различные инструменты создания и выбраны подходящие для реализации разработки. Модель была разработана с помощью Python, библиотек Matplotlib, Seaborn, Nltk, Re, PyMorphy2, Numpy, Pandas, Sklearn, Gensim в среде разработки GoogleColab.

Таким образом, была достигнута поставленная цель.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1) Google Colaboratory. Документация. : [сайт]. – 2024. URL: https://colab.research.google.com/?hl=ru_RU#scrollTo=UdRyKR44dcNI (дата обращения: 20.04.2024). – Текст : электронный

2) Что такое нейронная сеть?. : [сайт]. – 2023. – URL: https://trends.rbc.ru/trends/industry/641157be9a7947d3401fa3e8 (дата обращения: 23.04.2024). – Текст : электронный.

3) Pandas. Документация. : [сайт]. – 2024. – URL: https://pandas.pydata.org/docs/ (дата обращения: 24.04.2024). – Текст : электронный.

4) NumPy. Документация. : [сайт]. – 2024. – URL: https://numpy.org/doc/ (дата обращения: 24.04.2024). – Текст : электронный.

5) PyMorphy2. Документация. : [сайт]. – 2024. – URL: https://pymorphy2.readthedocs.io/en/stable/user/guide.html (дата обращения: 24.04.2024). – Текст : электронный.

6) Sklearn. Документация. : [сайт]. – 2024. – URL: https://scikit-learn.org/stable/index.html (дата обращения: 25.04.2024). – Текст : электронный.

7) Matplotlib. Документация. : [сайт]. – 2024. – URL: https://matplotlib.org/stable/index.html (дата обращения: 25.04.2024). – Текст : электронный.