

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
(наименование института полностью)

Кафедра «Прикладная математика и информатика»
(наименование кафедры)

02.03.03 Математическое обеспечение и администрирование информационных систем
(код и наименование направления подготовки)

Мобильные и сетевые технологии
(направленность (профиль) / специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему «Разработка интеллектуальной системы анализа тональности комментариев пользователей»

Обучающийся

Д.С. Сорокина
(Инициалы Фамилия)

(личная подпись)

Руководитель

к.т.н, В.С. Климов

(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2023

Аннотация

Тема бакалаврской работы – «Разработка интеллектуальной системы анализа тональности комментариев пользователей». Развитие технологий автоматизированного анализа отзывов пользователей является актуальной задачей для бизнеса, так как они позволяют получать оперативную обратную связь от клиентов и принимать на ее основе управленческие решения. Технологии машинного обучения могут использоваться для повышения степени автоматизации анализа отзывов клиентов. Данное исследование направлена на разработку интеллектуального алгоритма для анализа тональности комментариев клиентов.

Объектом исследования бакалаврской работы является оценка тональности комментариев пользователей.

Предметом исследования бакалаврской работы является система для анализа тональности комментариев пользователей.

Цель выпускной квалификационной работы – разработка интеллектуальной системы анализа тональности комментариев пользователей.

Задачами исследования являются:

- анализ технологий text mining;
- разработка алгоритма для анализа тональности текста;
- программная реализация алгоритма и его тестирование.

Данная работа состоит из введения, трех глав, заключения и списка используемой литературы.

В первой главе работы приводится анализ технологий text mining, во второй главе приводятся описание разработанного алгоритма анализа текста комментариев пользователей, в третьей главе описана практическая реализация предложенных решений.

Бакалаврская работа состоит из 43 страниц текста, 26 рисунков, 1 таблицы и 20 источников.

Abstract

The topic of the bachelor's thesis is "Development of intelligent system for analyzing the tone of user comments". The development of technologies for automated analysis of user feedback is an urgent task for business, as they allow you to receive prompt feedback from customers and make management decisions based on it. Machine learning technologies can be used to increase the degree of automation of customer feedback analysis. This research aims to develop an intelligent algorithm to analyze the tone of customer comments.

The object of the undergraduate research is the evaluation of the tone of user comments.

The subject of the bachelor's study is a system to analyze the tone of user comments.

The purpose of the graduate qualification work is to develop an intelligent system to analyze the tone of user comments.

The objectives of the research are:

- analysis of text mining technologies;
- development of an algorithm for analyzing the tone of the text;
- software implementation of the algorithm and its testing.

This paper consists of an introduction, three chapters, a conclusion and a list of references.

The first chapter of the work contains an analysis of text mining technology, the second chapter contains a description of the developed algorithm for analyzing text comments from users, the third chapter describes the practical implementation of the proposed solutions.

The bachelor's work consists of 43 pages of text, 26 figures, 1 table and 20 sources.

Оглавление

Введение.....	5
Глава 1 Анализ текущего состояния исследований в области определения тональности текста.....	8
1.1 Технологии анализа тональности текста (sentiment analysis).....	8
1.2 Обзор программных аналогов для оценки тональности текста.....	13
Глава 2 Разработка алгоритма для классификации тональности текста.....	17
2.1 Алгоритм использования разрабатываемой системы.....	17
2.2 Выбор модели векторизации текста.....	19
2.3 Выбор модели классификации.....	21
Глава 3 Разработка приложения для классификации тональности текста.....	25
3.1 Особенности в реализации программного обеспечения.....	25
3.2 Тестирование классификации текстов.....	31
Заключение.....	41
Список используемой литературы и используемых источников.....	42

Введение

При развитии своей предпринимательской деятельности, представители бизнес-сообщества ориентируются в том числе и на отзывы клиентов об предоставляемых ими услуг и товаров. Отзывы клиентов является одним из важнейших источников информации об удовлетворенности коммерческой деятельностью компании. В конечном счете клиентоориентированность обеспечивает большинству компаний устойчивую позицию на рынке товаров и услуг за счет выработки лояльности клиентов и формирования постоянного потока новых заказов.

В настоящее время в глобальной сети существует большое количество специализированных ресурсов, форумов, а также социальных сетей, где клиенты компании делятся своим мнением о качестве получаемых услуг. Анализ отзывов клиентов позволяет диагностировать текущие проблемы, имеющиеся при оказании компанией соответствующих услуг. Поэтому поиск отзывов, анализ их содержания, а также подведение статистики на основе полученной из отзывов информации является одной из актуальных задач современного бизнес-сообщества.

В рамках данной работы исследуются технологии анализа текстовых комментариев пользователей для оценки их тональности. Для анализа текстовой информации применяются методы text mining.

В качестве общедоступного источника отзывов пользователей используется сервис «Народный рейтинг банков», расположенный на по адресу <https://www.banki.ru/services/responses/list/>. Этот сервис выбран в качестве источника данных для обучающей выборки, так как в нем содержится одновременно и текстовый комментарий пользователей и в виде числа от «1» до «5» удовлетворенность пользователя предоставляемыми банковскими услугами. Оценка пользователей в виде числа используется в качестве метки класса, а текст комментариев в качестве входных данных для

разрабатываемой системы анализа тональности. Оценка «1» обозначает резко негативную тональность комментария и недовольство качеством оказанной услуги, а оценка «5» обозначает позитивный комментарий и полную удовлетворенность пользователя.

В рамках данной бакалаврской работы сужение области до анализа комментариев про оказание банковских услуг было выбрано осмысленно, по следующим причинам:

- чем шире практическая область, из которой анализируется текстовая информация, тем больше должна быть обучающая выборка для настройки модели. Так, например, для обучения генеративной нейронной сети, используемой в модели ChatGPT потребовался анализ миллиардов текстовых документов на разные темы;

- увеличение разнообразия текстовой информации приводит к усложнению и расширению применяемых методов предобработки текстовых данных. При этом определения оптимальных параметров работы данных методов становится слишком объемной задачей, которую затруднительно решить единолично.

При разработке системы анализа тональности комментариев пользователей, ключевыми проблемами, снижающих точность получаемых результатов являются:

- наличие в тексте комментариев сарказма. Например, словосочинение «спасибо банку» может присутствовать, как в позитивном комментарии, так и в негативном («спасибо банку, за то, что несколько часов провел в очереди»). Хотя исследования на тему алгоритмического определения и фильтрации сарказма в тексте ведутся, но не удалось найти готового автоматизированного решения способного нивелировать данную проблему.

- неоднозначность оценок от разных пользователей. Затрагивая одну и ту же проблему, разные пользователи дают различные оценки обслуживания. Например, озвучивая в текстовом комментарии проблему очередей одни

пользователь ставит оценку «2», а другой пользователь - «3». Такая неоднозначность приводит к снижению точности классификации комментариев пользователей. Для решения данной проблемы в работе предлагается перемаркировать классы, снизив их количество: отзывы с оценками «5» и «4» считать «положительными», а отзывы с оценками «3», «2», «1» считать «отрицательными».

Системы анализа тональности комментариев пользователей разрабатывались на языке Python. В качестве модели классификации использовался XgBoost Classifier, для конвертирования текста комментариев пользователя в числовой вид использовался подход «Bag-of-words».

С учетом предложенных улучшений точность классификации комментариев пользователя достигла 83%.

Полученные в ходе выполнения бакалаврской работы решения можно адаптировать под анализ тональности текста комментариев пользователей из других предметных областей.

Глава 1 Анализ текущего состояния исследований в области определения тональности текста

1.1 Технологии анализа тональности текста (sentiment analysis)

Определение тональности текста (sentiment analysis) - это одна из задач в области обработки естественного языка и анализа текста, основанная на применении вычислительной лингвистики и направленная на идентификацию, извлечение, количественную оценку автора текста к предмету описания. Анализ тональности широко применяется при оценке качества предоставляемых услуг и товаров со стороны бизнеса и предполагает получение данных из различных источников, таких как опросы, комментарии пользователей, оставленных в социальных сетях.

С развитием глубоких языковых моделей, таких как RoBERTa, стало возможным подвергать анализу более сложные и объемные текстовые данные. Так, например, языковая модель RoBERTa используется в англоязычном сегменте для определения отношения авторов к описываемым ими событиям в текстовых новостях информационных порталов.

Основной задачей анализа тональности является классификация полярности рассматриваемого фрагмента текста на разных уровнях детализации:

- на уровне документа;
- на уровне предложения;
- на признака/аспекта.

В общем случае требуется определить к какому классу эмоций относится анализируемый фрагмент текста (рисунок 1):

- положительные эмоции («positive»);
- негативные эмоции («negative»);
- нейтральные эмоции («neutral»).

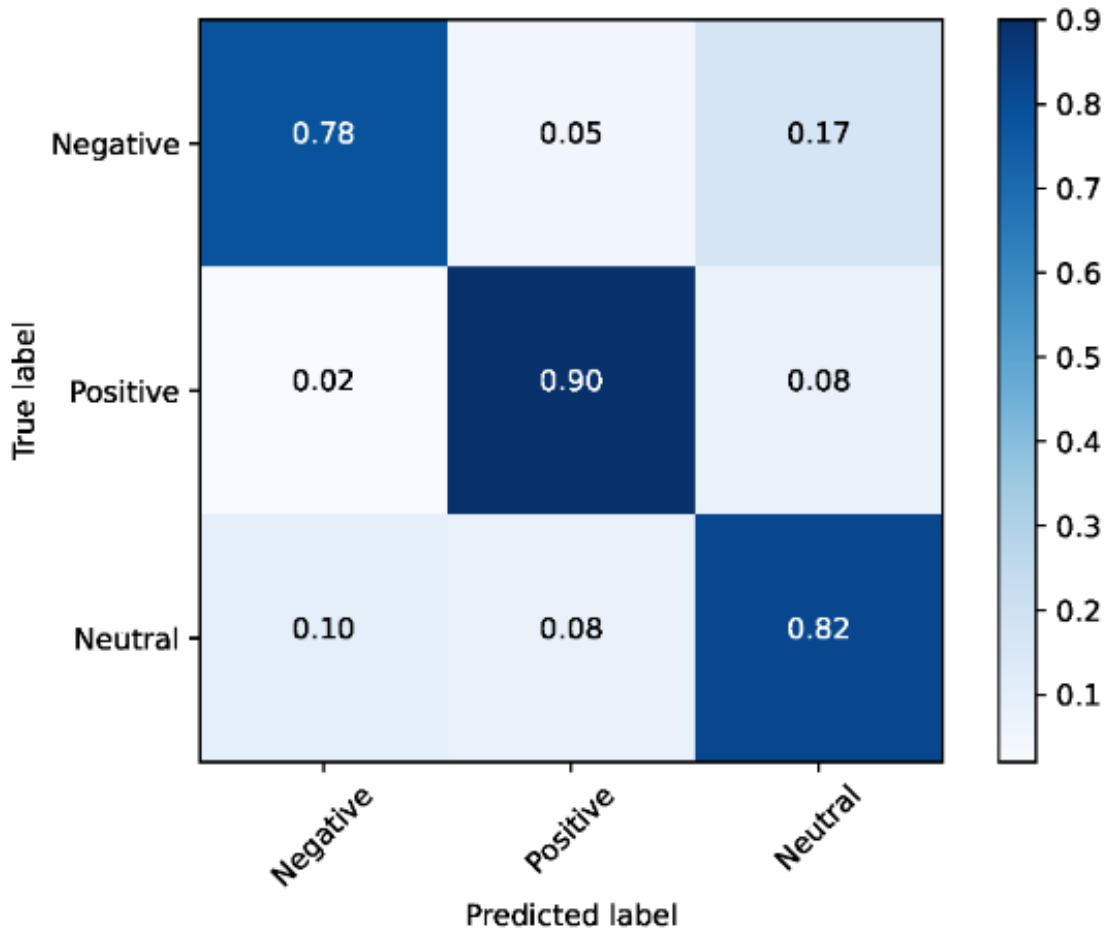


Рисунок 1 – Вариант матрицы ошибок для при распознавании 3 классов эмоций

Существует также вариант расширенной классификации настроений, выходящей за рамки полярной (негативно-позитивной) модели. В этом варианте классов эмоций уже 7 (рисунок 2):

- удовольствие («Happy»);
- гнев («Angry»);
- отвращение («Disgust»);
- печаль («Sad»);
- страх («Fear»);
- удивление («Surprise»);
- нейтральные эмоции («Neutral»).

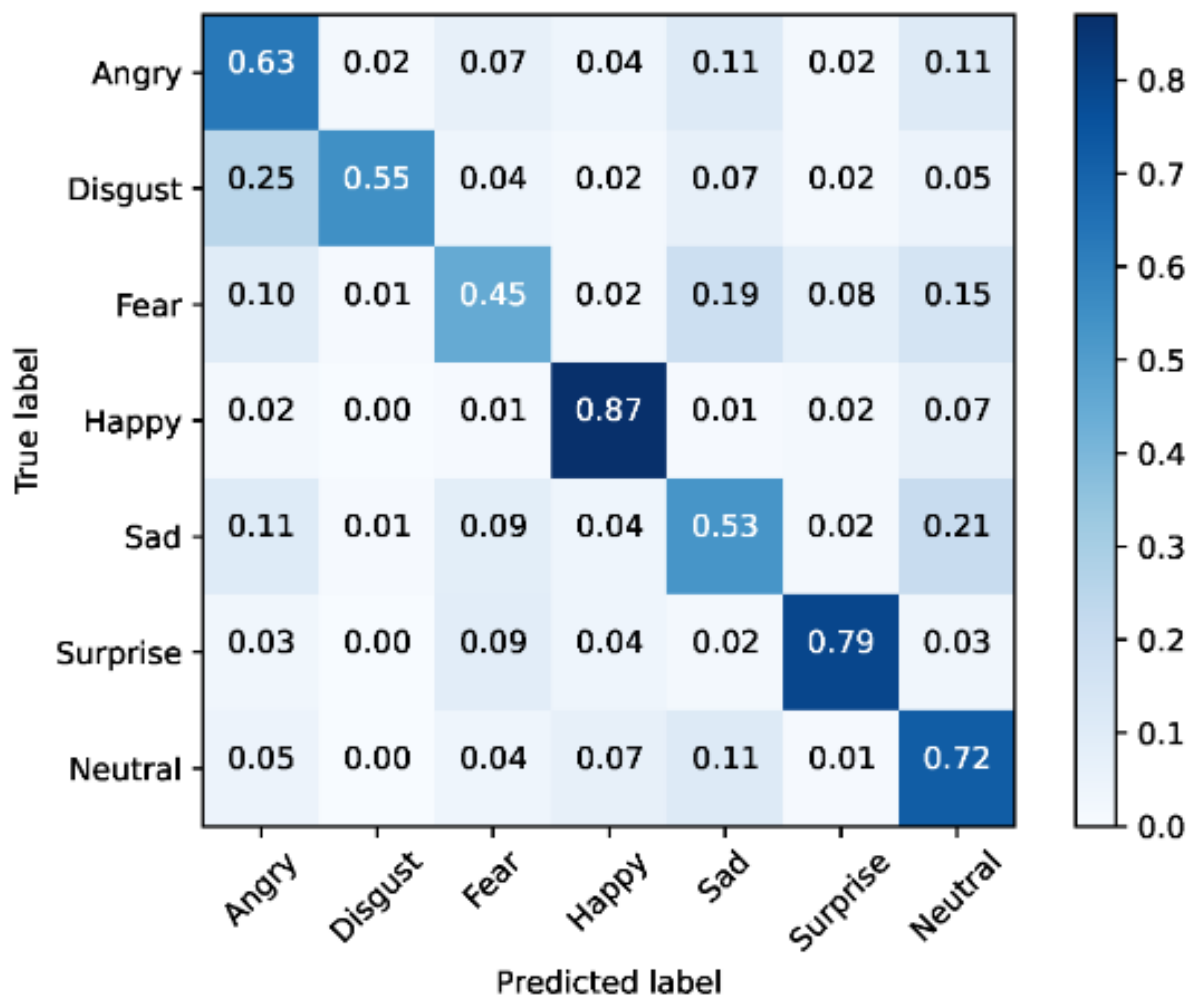


Рисунок 2 – Вариант матрицы ошибок для при распознавании 7 классов эмоций

До появления термина «сентиментальный анализ» Stone опубликовал исследования о возможности количественной оценки паттернов (шаблонных выражений), встречающихся в тексте. Полученные результаты он предполагал связать с исследованиями, которые изучали психологическое состояние человека на основе анализа речевого поведения [23].

Впоследствии авторами Volcani и Fogel был предложен метод, связывающий настроения авторов с отдельными слова и фразами в тексте относительно различных эмоциональных шкал. На данный метод был получен патент "System and method for determining and controlling the impact of text" (USA Issued 7,136,877, 2001).

На основе их исследования была разработана система EffectCheck, которая позволяет переписывать текст без изменения смысла за счет замены отдельных фраз с целью повышения или понижения уровня вызываемых эмоций по каждой шкале. Перечень шкал эмоций показан на рисунке 2.

Во многих других исследованиях для упрощения задачи анализа тональности текста применяются биполярные шкалы (положительные-отрицательные эмоции).

В работе Turney применяются различные методы для определения полярности отзывов о товарах [25]. А в работе Pang эти же методы используются для определения полярности рецензий на фильмы [16]. При этом в обоих исследованиях анализа тональности выполнялась на уровне текстового документа.

В другом исследовании Pang и Lee расширили базовую задачу классификации рецензии на фильм с положительных и отрицательных до прогнозирования оценки по четырехзвездочной шкале [15].

В своем исследовании Snyder провел углубленный анализ ресторанных рецензий и научился предсказывать оценки различных аспектов ресторана по пятизвездочной шкале, таких, например, как еда, атмосфера и др [22].

Проблемы объединения различных подходов машинного обучения и лексического анализа докладывались на весеннем симпозиуме AAAI 2004 года, где лингвисты, IT-специалиста и другие исследователи определили перспективы и сформулировали задачи по развитию сентиментального анализа, а также определили эталонные наборы данных для тестирования новых методов определения настроения по тексту [17].

Также стоит отметить, что не редко в исследованиях по сентиментальному анализу текста игнорируется нейтральный класс («Neutral») исходя из предположения, что нейтральные тексты лежат вблизи границ бинарного классификатора (негативно-позитивная классификация).

Однако существует примеры исследований, в которых добавления

нейтрального класса повышает точность классификации основанной на использовании таких методов, как Max Entropy и SVM.

Также в исследованиях отмечается, что в каждом отдельном случае на конкретном наборе данных стоит отдельно проверять будет ли выигрыш в точности после добавления класса для нейтральных текстов.

Существует два способа работы с нейтральными текстами (классом «Neutral»):

- в первом способе при определении тональности текста сначала определяется, относится ли он к нейтральным текстам и нет. Дальнейшему анализу подвергаются только те тексты, которые не относятся к нейтральному классу и только в этом они оцениваются с точки зрения позитивных и негативных настроений;

- во втором способе анализируемый текст подвергается трехсторонней классификации за один шаг. Этот подход часто включает оценку распределения вероятностей по всем категориям (например, с использованием Naïve Bayes, реализованного в библиотеке NLTK).

Использование нейтрального класса зависит от характера анализируемых данных. Если имеющиеся в обучающей выборке текстовые данные, относящиеся к классам «нейтральные», «негативные» и «позитивные» имеют существенные отличия, то удовлетворительных результатов можно добиться, отфильтровав нейтральные высказывания с последующим обучением классификатора на текстовых фрагментах только с позитивным и негативным настроениями. Но если такой подход применить к обучающей выборке, в которой в основном сосредоточены нейтральные текстовые фрагменты с небольшими отклонениями в сторону позитивных и негативных аффектов, то такая стратегия приведет к снижению точности работы конечного классификатора.

Другим методом определения настроения является использование системы шкалирования, при которой словам, обычно ассоциирующимся с

негативным, нейтральным или позитивным настроением, присваивается соответствующий номер по шкале от -10 до +10 (от самого негативного до самого позитивного) или просто от 0 до положительного верхнего предела, например +4. Это позволяет перейти к более сложному пониманию настроения текста, поскольку теперь можно оценивать вклад каждого отдельно взятого понятия в общее настроение анализируемого текста.

В некоторых вариациях такого подхода определяются слова-модификаторы, которые усиливают и ослабляют оценку настроения отдельно взятого выражения, предложения или всего текста.

Существуют и другие виды анализа настроения текста, такие как анализ настроения на основе аспектов, анализ настроения по градациям, многоязычный анализ настроения и классификация эмоций.

Можно отметить, что в настоящее время технологии анализа настроений текста активно развиваются.

1.2 Обзор программных аналогов для оценки тональности текста

В настоящее время существует ряд автоматизированных систем для определения тональности текстовых данных. Рассмотрим наиболее известные из них. Подробное сравнение аналогов представлено на рисунке 3.

Stanford NLP является платформой для обработки естественного языка, разработанной Стенфордским государственным университетом. Полный функционал платформы распространяется на коммерческой основе, но существует демонстрационная версия программного обеспечения, которая включает в себя англоязычную языковую модель, обученную на текстах рецензий фильмов. В основе работы системы лежат рекурсивные нейронные сети (модификация сети BERT).

Название системы	Поддерживаемые языки	Сайт	Классификатор	Описание
Stanford NLP	Английский	nlp.stanford.edu	Нейронная сеть BERT	В открытом доступе демо-модель, обученная на основе рецензий фильмов.
Sentiment140	Английский, Испанский	sentiment140.com	Модель на основе «distant supervision»	Модель обученная на основе данных из сервиса Twitter. Поддерживает визуализацию с помощью инфографики.
IBM Watson NLP	Английский	ibm.com	Нейронная сеть BERT	Поисковая и текстово-аналитическая платформа ориентированная на работу с корпоративными данными.
ВААЛ	Русский	www.vaal.ru	Эвристическая модель на основе словарей	Позволяет выявлять психологические качества авторов. Поддерживает автоматическую категоризацию текстов.

Рисунок 3 – Сравнение систем для оценки тональности текста

Sentiment140 представляет из себя сервис позволяющий определять принадлежность сообщения пользователя к одному из 3 классов: позитивное, негативное, нейтральное сообщение. Разработчик заявляет, что языковая модель данной системы обучалась на сообщениях из сервиса Twitter. Так же у данного сервиса существует функционал обеспечивающий возможность получения по указанному пользователю Twitter выборку его позитивных, негативных и нейтральных публичных сообщений. Сервис поддерживает работу с двумя языками – испанским и английским.

IBM Watson NLP является прямым аналогом Stanford NLP. Данная платформа использует глубокое обучение для извлечения смысла и метаданных из неструктурированной текстовой информации. Представленный в платформе функционал позволяет решать такие задачи, как категоризация и классификация текста, извлечение ключевых слов, классификация эмоций и тональности. Система IBM Watson NLP работает на основе модификации нейронной сети BERT.

Система «ВААЛ» является отечественной разработкой на правленной на классификацию и генерирования текстовой информации. Работа системы основана на исследованиях в области психолингвистики и одной из ее особенностей является возможность эмоционально-лексического анализ текстов. Работа системы старится на основе системы словарей.

Стоит отметить, что ни одна из этих систем не является свободно распространяемым программным обеспечением.

Выводы по главе 1

Приведем выводы по первой главе работы:

– в ходе анализа литературных данных установлено, что изучение и совершенствование сентиментального анализа является актуальной задачей в области искусственного интеллекта, так как позволяет автоматизировать

процессы, связанные с обработкой текстовых данных;

– анализ тональности текста в сфере бизнеса может применяться для автоматизированного определения по отзывам пользователей их отношения к предоставляемым компанией услугам;

– в научной литературе есть примеры применения сентиментально анализа для определения оценки по тексту рецензии к фильму, а также прогнозирования оценки ресторана от клиента по содержанию его текстового отзыва;

– сентиментальный анализ основан на количественной оценке паттернов (шаблонных выражений), встречающихся в тексте и их связи с психологическим состоянием человека;

– проведен сравнительный анализ наиболее популярных систем для анализа тональности текста – Standford NLP, Sentiment140, IBM Watson NLP и «ВААЛ». Ни одна из этих систем не является свободно распространяемым программным обеспечением.

Глава 2 Разработка алгоритма для классификации тональности текста

2.1 Алгоритм анализа текстовых данных

Предложенный алгоритм определения тональности текста основан на совместной работе двух моделей: модели векторизации текстовых данных и классификации.

Модель векторизации нужна для преобразования текстовых фрагментов в числовые векторы, которые удобны для анализа с помощью различных алгоритмов классификации. А модель классификации нужна для того, чтобы связать компоненты векторов к классами тональности текста (рисунок 4).

Для получения настроенной модели векторизации выполняются следующие шаги:

- фильтрация текста от небуквенных символов (знаков препинания, спецсимволов, тэгов, оставшихся после парсинга текста);
- удаление стоп-слов, к которым относятся союзы, предлоги, частицы;
- подбор параметров векторизации модели;
- обучение модели, которое включает в себя предварительный анализ текстовых фрагментов, имеющих в обучающей выборке.

Получения настроенной модели классификации основано на выполнении следующих этапов:

- формирование обучающей выборки, путем обработки всех ее элементов с помощью модели векторизации;
- автоматизированный подбор параметров модели классификации, заключающийся в тестировании ее работы при их различных сочетаниях;
- обучение классификатора на тестовой выборке данных.



Рисунок 4 – Схема определения тональности текста, основанная на использовании моделей векторизации и классификации

Таким образом, наиболее важными задачами создания алгоритма определения тональности текста является выбор моделей векторизации и классификации текста.

2.2 Выбор модели векторизации текста

Существует несколько способов векторизации текстовых данных основанных на применении различных подходов. Наиболее известные модели векторизации:

- Bag-of-words;
- Word2Vec.

Модель векторизации Bag-of-words на наборе текстовых документов обучается следующим образом. Каждый текстовый фрагмент разбивается на отдельные слова. Одновременно с этим для всех текстовых фрагментов создается общий словарь слов. Заполнение словаря осуществляется путем последовательного перебора всех слов, имеющихся в текстовых фрагментах. Как только встречается слово, которого нет в словаре, оно добавляется в его конец. В конечном счете формируется словарь всех уникальных слов. При составлении словаря в него не включаются «стоп-слова» к которым относятся предлоги, союзы и частицы.

Модель векторизации Bag-of-words при преобразовании текстового фрагмента в числовой вектор работает с получившимся словарем уникальных слов. При векторизации текста осуществляет подсчет того, сколько раз каждое слово из словаря встретилось в анализируемом тексте. Таким образом, длина числового вектора, получаемого на выходе всегда равна количеству слов в словаре. А значение каждого компонента числового вектора равно частоте появления соответствующего слова в анализируемом тексте. Поясняющая схема работы модели векторизации Bag-of-words показана на рисунке 5.

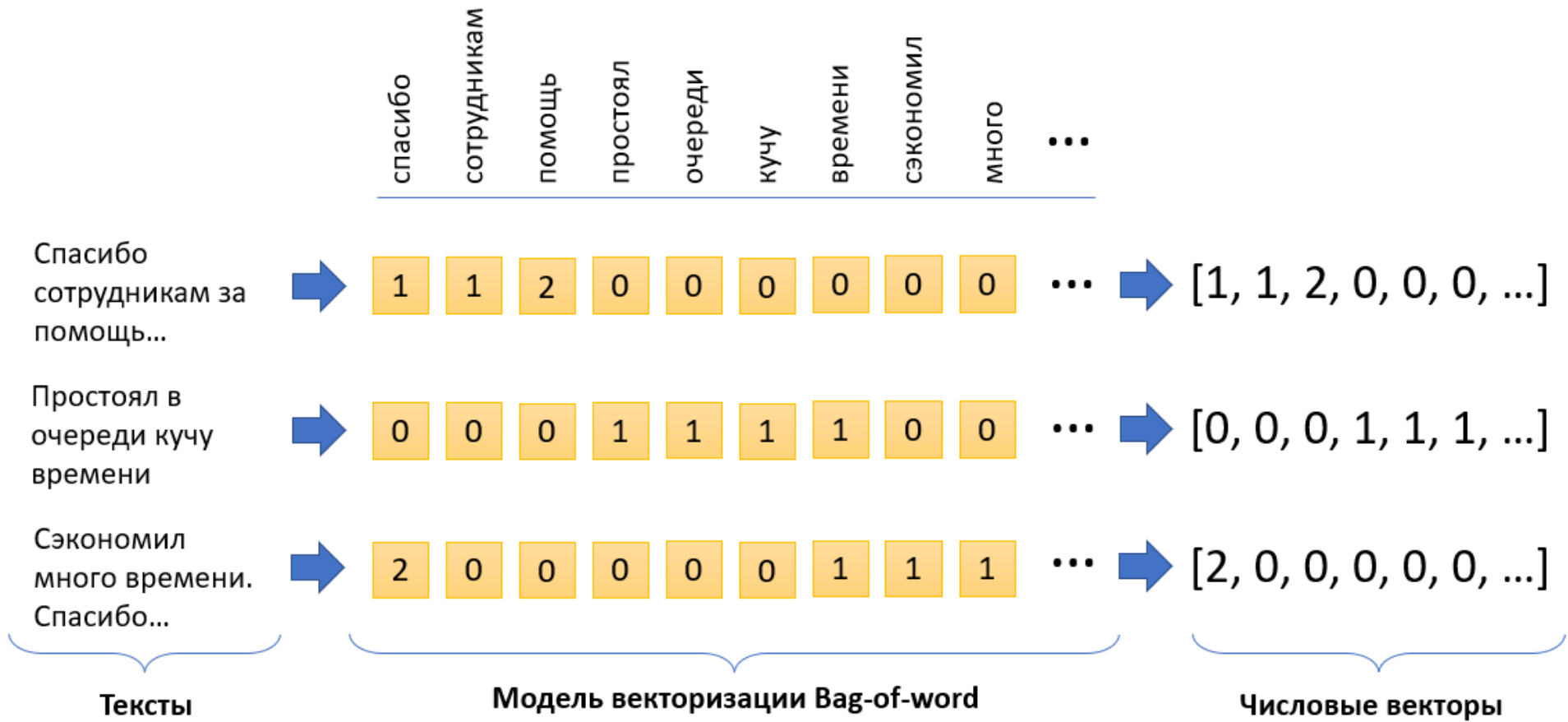


Рисунок 5 – Схема работы модели векторизации Bag-of-words

К преимуществам данной модели векторизации Bag-of-words можно отнести:

- простота математического аппарата;
- более высокая скорость обучения модели Bag-of-words по сравнению с Word2Vec;
- возможность ее применения для малых наборов текстовых данных и для текстовых фрагментов произвольной длины.

Альтернативным подходом является использование модели Word2Vec. Обучение данной модели векторизации на наборе текстовых фрагментов является более сложной задачей по сравнению Bag-of-words. Это связано с тем, что модель Word2Vec умеет оценивать близость различных слов по смыслу на основе частоты и близости их появления в текстах. При этом одним из ключевых понятий данной модели является ширина контекстного окна, обозначающего расстояние в предложении между двумя анализируемыми словами.

Ключевым преимуществом Word2Vec по сравнению с Bag-of-words является возможность учета контекста слов при векторизации текстовых фрагментов. Однако преимуществу достигается за счет значительного увеличения времени обучения модели Word2Vec и необходимости наличия большой выборки текстовых фрагментов.

Для того чтобы сохранить универсальность предлагаемого алгоритма для классификации тональности текста и не зависеть от размера обучающей выборки предложено использование модели Bag-of-words.

2.3 Выбор модели классификации

Алгоритмы построения классификаторов наиболее эффективно работают с данными, представленными в числовом виде. Поэтому после преобразования текстовых данных в числовые векторы для их последующего

анализа становится возможным применение различных алгоритмов обучения классификаторов:

- алгоритмы обучения нейронных сетей;
- алгоритмы построения метрических классификаторов по типу kNN;
- алгоритмы построения деревьев принятия решений.

В общем случае, выбор алгоритма, обеспечивающего получение наиболее точного классификатора для решения поставленной задачи является сложной исследовательской задачей. Это связано с тем, что никогда заранее не известно какой из алгоритмов покажет лучшие результаты. Однако в машинном обучении существует подход, который позволяет нивелировать проблему с подбором оптимального алгоритма. Данный подход носит название градиентный бустинг.

Основная идея градиентный бустинг заключается в отказе от построения одиночного классификатора. Вместо одиночного классификатора предложено обучать набор (ансамбль) менее точных (т.н. слабых) классификаторов. Хотя по отдельности каждый из слабых классификаторов, участвующих в ансамбле, обладает невысокой точностью работы, но работая вместе эти классификаторы дополняют друг друга, обеспечивая высокую точность прогнозирования.

Градиентный бустинг отличается от других ансамблевых алгоритмов построения классификаторов тем, что на каждой следующей итерации ансамбль дополняется классификатором, настроенным на компенсацию ошибок в работе предыдущих классификаторов. Схема градиентного бустинга показана на рисунке 6.

Наиболее известным алгоритмом градиентного бустинга является XGBoost. В данном алгоритме в качестве слабых классификаторов используются деревья принятия решений.

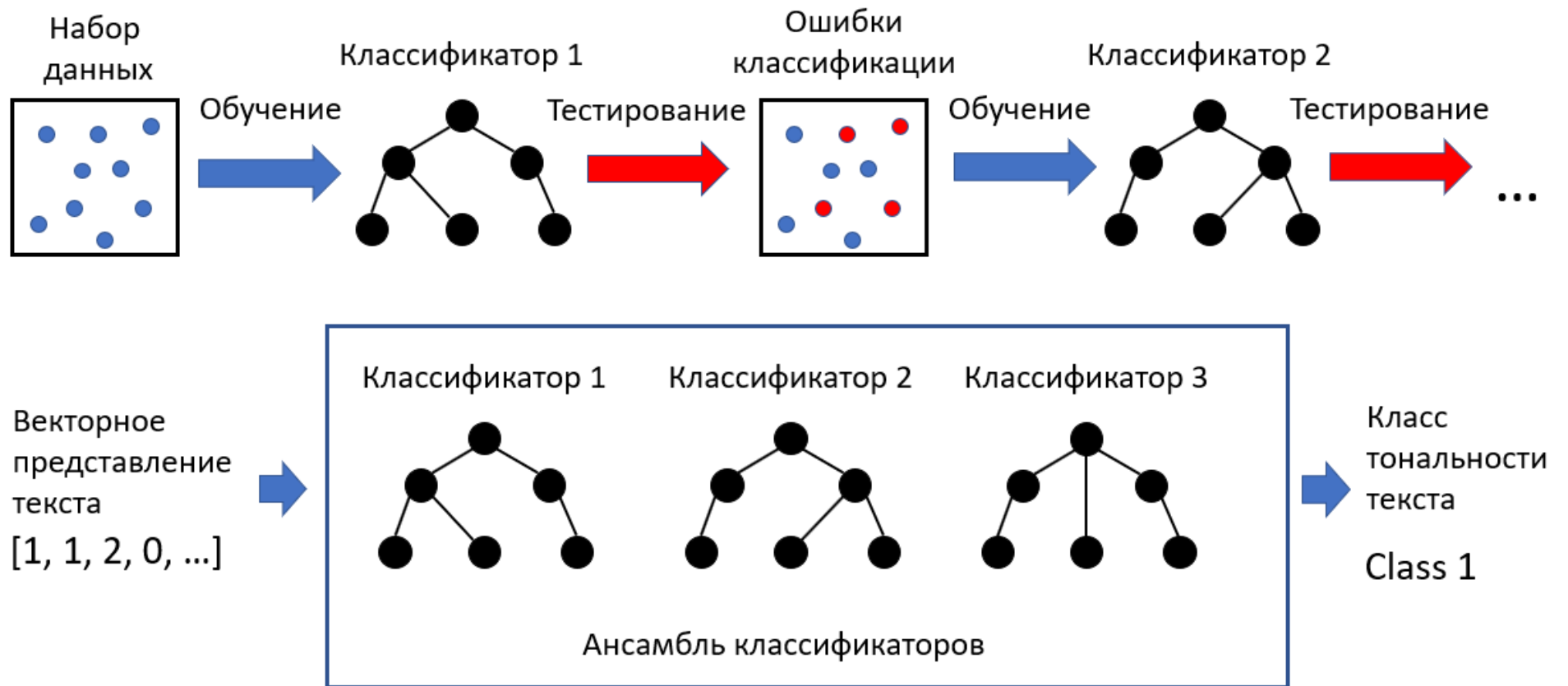


Рисунок 6 – Схема работы градиентного бустинга при построении ансамбля классификаторов

Таким образом, в предложенном алгоритме определения тональности текста используется модель векторизации Bag-of-words и классификатор XGBoost classifier.

Выводы по главе 2

Приведем выводы по второй главе:

- предложен алгоритм определения тональности текста, который включает в себя преобразование текстовых данных в векторный вид при помощи модели векторизации Bag-of-words и анализ полученных векторов с помощью классификатора XGBoost classifier;

- в предложенном алгоритме этап векторизации текста включает в себя фильтрацию текста от небуквенных символов, удаление стоп-слов, подбор параметров векторизации и обучение модели векторизации, заключающееся в построении словаря уникальных слов;

- в предложенном алгоритме этап классификации текста включает в себя формирование обучающей выборки, подбор параметров классификатора и обучение классификатора методом градиентного бустинга.

Глава 3 Разработка приложения для классификации тональности текста

3.1 Особенности в реализации программного обеспечения

Для реализации описанного в предыдущей главе алгоритма анализа тональности текста на языке Python было разработано программное обеспечение, реализующее следующий функционал:

- загрузка обучающей выборки для построения языковой модели из файлов формата csv;
- очистка текстовых данных от посторонних символов;
- преобразование текстовых фрагментов в числовые векторы с использованием метода Bag-of-words;
- обучение на основе числовых векторов классификатора XGB classifier для определения класса настроений анализируемого текста;
- тестирование обученного классификатора текста и визуализация результатов в виде матрицы ошибок.

В программном обеспечении использовались библиотеки, включающие в себя реализацию методов анализа текстовых данных (рисунок 7).

```
import numpy as np
import pandas as pd
import re
from nltk.corpus import stopwords
from bs4 import BeautifulSoup
from sklearn.feature_extraction.text import CountVectorizer
import xgboost as xgb
from sklearn.model_selection import cross_val_score
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import accuracy_score
import os
```

Рисунок 7 – Подключение библиотек

Подробное описание назначения использованных библиотек показано в таблице 1.

Таблица 1 – Библиотеки используемые в реализованном программном обеспечении

Название библиотеки	Назначение
Numpy	Использование математических функций и методов по работе с числовыми массивами
Pandas	Работа с табличными данными
Re	Поддержка регулярных выражений для очистки текстов от посторонних символов
BeautifulSoup	Извлечение комментариев пользователей из html страниц
xgboost	Реализация классификатора на основе градиентного бустинга
sklearn	Использование методов для построения матрицы ошибок, оценки точности классификатора, автоматизированного поиска оптимальных параметров классификатора

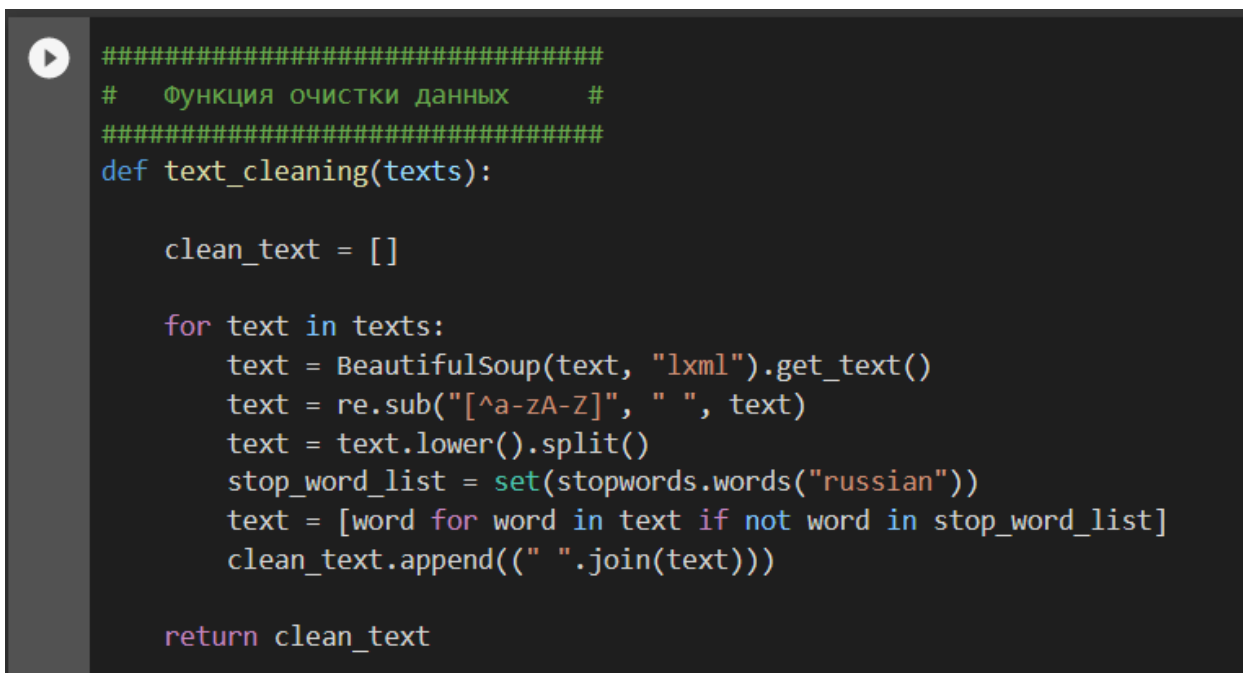
Чтобы подготовить текст для анализа его необходимо очистить от посторонних символов и знаков препинания. Для реализации очистки текста была написана функция `text_cleaning()`, которая получает на в ход список `texts` текстовых фрагментов, подвергаемых очистке.

Цикл `for` итерационно перебирает текстовые фрагменты и подвергает и[очистке.

Очистка текста от посторонних символов осуществляется с использованием регулярного выражения, оставляющего только буквы и пробелы и удаляющего любые другие символы. Затем все полученные слова

переводятся к нижнему регистру с помощью метода `lower()`. После этого осуществляется удаление стоп-слов, которые включают в себя предлоги, союзы и частицы.

С помощью метода `append()` на каждой итерации результат очистки очередного текстового фрагмента добавляется к коллекции `clean_text`, которая в виде списка возвращается функцией `text_cleaning()` (рисунок 8).

A screenshot of a code editor showing a Python function named `text_cleaning`. The function takes a list of strings `texts` as input and returns a list of cleaned strings `clean_text`. The code uses `BeautifulSoup` to parse XML, `re.sub` to remove non-alphabetic characters, `lower()` to convert to lowercase, and `split()` to split into words. A set of Russian stopwords is used to filter out unwanted words. The cleaned words are then joined back together with spaces.

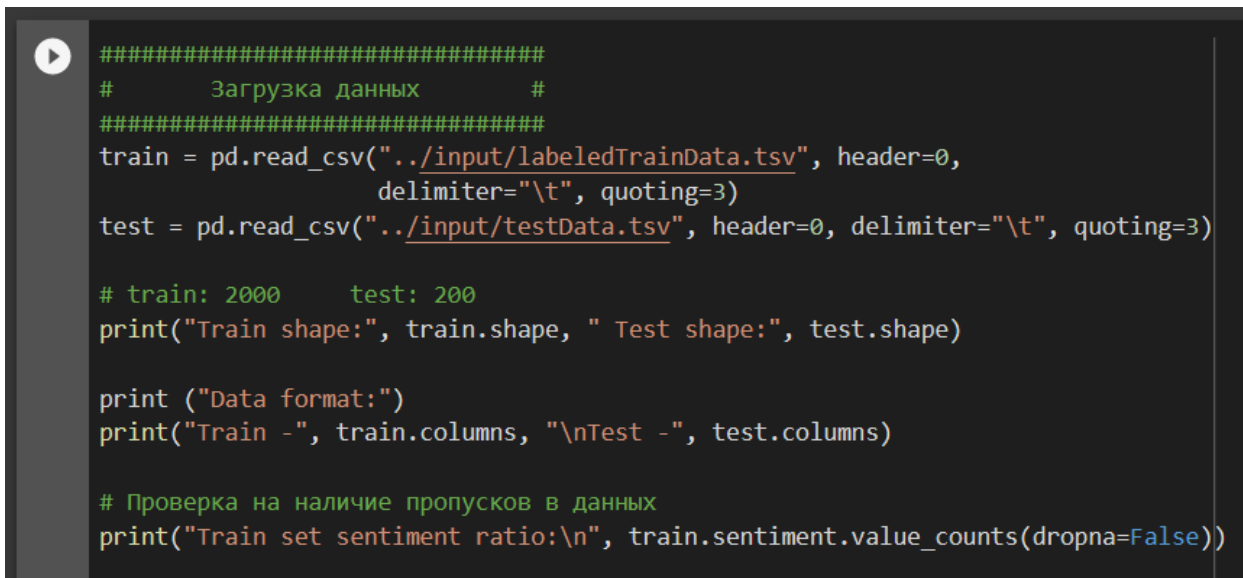
```
#####  
#   Функция очистки данных   #  
#####  
def text_cleaning(texts):  
  
    clean_text = []  
  
    for text in texts:  
        text = BeautifulSoup(text, "lxml").get_text()  
        text = re.sub("[^a-zA-Z]", " ", text)  
        text = text.lower().split()  
        stop_word_list = set(stopwords.words("russian"))  
        text = [word for word in text if not word in stop_word_list]  
        clean_text.append(" ".join(text))  
  
    return clean_text
```

Рисунок 8 – Очистка данных

Приложение умеет работать обучающими и тестовыми выборками данных, хранящихся в текстовых файлах формата `tsv`. Отличительной особенностью данного формата является применения знака табуляции для разграничения хранящихся в них данных.

Для загрузки обучающей и тестовой выборок данных применяется метод `read_csv()`, реализуемый библиотекой `pandas`. В качестве параметров данному методу передаются: путь до файла; номер строки `header` в которой содержатся названия столбцов; символ `delimiter`, используемый для разделения данных (`<<t>` – знак табуляции).

После загрузки файлов для удобства пользователя на экран выводится информация о размерности загруженных данных – количество столбцов и строк отдельно для обучающей и тестовой выборок данных (рисунок 9).

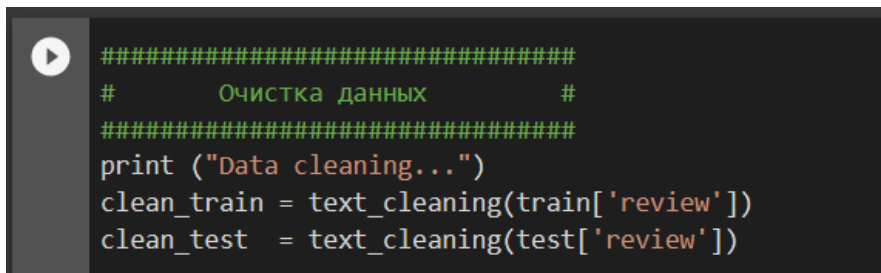
A screenshot of a code editor with a dark background and light-colored text. The code is in Python and uses the pandas library to load data from TSV files. It includes comments in Russian and English, and prints the shapes of the loaded data frames. The code is as follows:

```
#####  
#      Загрузка данных      #  
#####  
train = pd.read_csv("../input/labeledTrainData.tsv", header=0,  
                    delimiter="\t", quoting=3)  
test = pd.read_csv("../input/testData.tsv", header=0, delimiter="\t", quoting=3)  
  
# train: 2000      test: 200  
print("Train shape:", train.shape, " Test shape:", test.shape)  
  
print ("Data format:")  
print("Train -", train.columns, "\nTest -", test.columns)  
  
# Проверка на наличие пропусков в данных  
print("Train set sentiment ratio:\n", train.sentiment.value_counts(dropna=False))
```

Рисунок 9 – Загрузка данных

После загрузки тестовой и обучающих выборок данных из tsv файлов осуществляется их очистка с использованием описанной выше функции `text_cleaning()`. При этом очистке подвергаются столбцы содержащие основные текстовые фрагменты (столбцы «review»).

Перед стартом процесса очистки текстовых фрагментов пользователю выводится сообщение «Data cleaning», чтобы он мог контролировать процесс работы программы.

A screenshot of a code editor with a dark background and light-colored text. The code is in Python and uses a function named `text_cleaning()` to clean the 'review' column of the training and test data frames. The code is as follows:

```
#####  
#      Очистка данных      #  
#####  
print ("Data cleaning...")  
clean_train = text_cleaning(train['review'])  
clean_test = text_cleaning(test['review'])
```

Рисунок 10 – Очистка данных

После этапа очистки текстовых фрагментов следующим шагом является преобразование текстов в числовые векторы с использованием метода Bag-of-words. Сначала пользователю выводится на экран сообщение о том, что начался процесс построения модели векторизации. Затем инициализируется модель векторизации путем вызова конструктора `CountVectorizer()`. Инициализируемая модель хранится под именем `vectorizer`. При инициализации в конструктор передаются следующие параметры:

- `analyzer` – анализируемые элементы модели, в нашем случае элементами являются слова;
- `tokenizer` – метод токенизации слов, в нашем случае токенизация не применяется;
- `max_features` – ограничение по количеству уникальных слов при настройке модели векторизации, в нашем случае модель ограничена 5000 слов.

Процедуру векторизации проходят все элементы обучающей и тестовой выборки данных. Результаты векторизации фрагментов текста сохраняются в переменные `bow_train` и `bow_test`.

Программный код для преобразования текста в числовые векторы показан на рисунке 11.

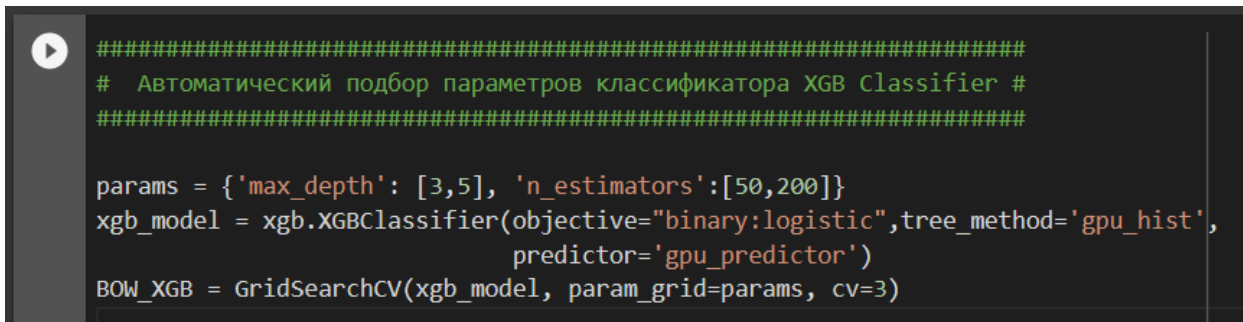
```
#####  
# Преобразование текстов в числовые векторы #  
#####  
# Bag of Words  
print ("Building Bag-of-Words...")  
vectorizer = CountVectorizer(analyzer = "word", tokenizer = None,  
                             max_features = 5000)  
bow_train = (vectorizer.fit_transform(clean_train)).toarray()  
bow_test = (vectorizer.transform(clean_test)).toarray()
```

Рисунок 11 – Преобразование текстов в числовые векторы

После того, как все фрагменты текста преобразованы в числовые векторы на их основе производится обучение классификатора. В данной работе применяется классификатор XGB Classifier, основанный на градиентном бустинге.

Для подбора параметров градиентного бустинга применяется метод GridSearchCV. Данный метод проверяет точность работы классификатора XGB Classifier при всех возможных сочетаниях параметров, указанных в переменной params и определяет наилучшее их сочетание.

Программный код для автоматического подбора параметров градиентного бустинга показан на рисунке 12.

A screenshot of a code editor showing Python code for automatic parameter selection of an XGB Classifier. The code is as follows:

```
#####  
# Автоматический подбор параметров классификатора XGB Classifier #  
#####  
  
params = {'max_depth': [3,5], 'n_estimators':[50,200]}  
xgb_model = xgb.XGBClassifier(objective="binary:logistic",tree_method='gpu_hist',  
                             predictor='gpu_predictor')  
BOW_XGB = GridSearchCV(xgb_model, param_grid=params, cv=3)
```

Рисунок 12 – Автоматический подбор параметров классификатора

Подбор оптимальных параметров градиентного бустинга с использованием GridSearchCV занимает несколько часов времени, так как осуществляет многократное обучение классификатора XGB Classifier. При этом для с помощью метода mean() пользователю рассчитывается и выводится на экран средняя точность работы классификатора достигнутая в результате подбора параметров градиентного бустинга.

Определив оптимальные параметры градиентного бустинга программа с помощью метода fit() осуществляет обучение классификатора XGB Classifier уже на всей имеющейся обучающей выборке текстовых данных.

```
#####
# Обучение классификатора XGB Classifier #
#####
# XGB Classifier
print ("Model building - XGB...")

BOW_XGB = xgb.XGBClassifier(max_depth=7, n_estimators=300,
                            objective="binary:logistic", random_state=1,
                            tree_method='gpu_hist', predictor='gpu_predictor')
BOW_XGB_scores = cross_val_score(BOW_XGB, bow_train, train.sentiment, cv=3,
                                  n_jobs=-1)
print("Averaged CV Accuracy: %0.2f (+/- %0.2f)" % (BOW_XGB_scores.mean(),
                                                  BOW_XGB_scores.std() * 2))

BOW_XGB.fit(bow_train, train.sentiment)

# Make prediction
result = BOW_XGB.predict(bow_test)

submission = pd.DataFrame(data={"id":test["id"], "sentiment":result})
submission.to_csv("BagOfWord-XGB.csv", index=False, quoting=3)
print("Done")
```

Рисунок 13 – Обучение классификатора и тестирование его работы

Для сохранения полученной модели векторизации сначала формируется DataFrame и затем он экспортируется в текстовый файл с помощью метода to_csv().

Обученный текстовый классификатор XGB Classifier храниться в переменной BOW_XGB. В последствии его можно использовать для определения класса настроения для произвольного текстового фрагмента.

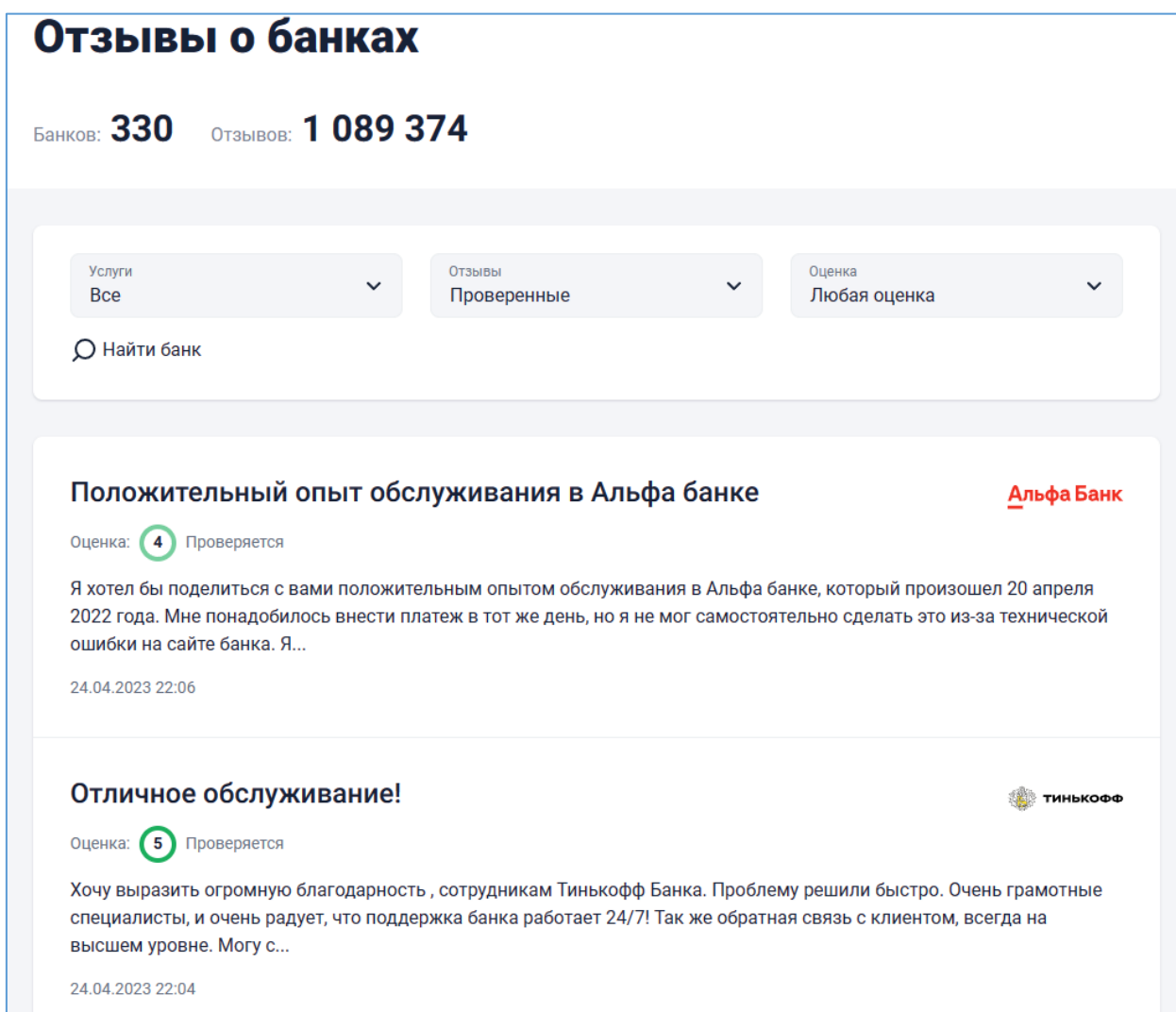
3.2 Тестирование классификации текстов

Для того чтобы проверить эффективность разработанного алгоритма для классификации тональности текста, предполагающего использование метода векторизации Bag-of-word и классификатора XGB Classifier необходимо разработать методику тестирования.

Предложенная методика тестирования алгоритма включает себя проверку работы классификатора тональности текста на тестовой выборке

данных, построение по результатам тестирования матрицы ошибок и оценка матрицы ошибок для корректировки работы алгоритма под конкретную выборку данных.

Исходными данными для тестирования предложенного алгоритма определения тональности текста являются отзывы клиентов с сервиса banki.ru (рисунок 14). Целью работы алгоритма является – научиться точно предсказывать оценки клиентов (от 1 до 5) на основе текста отзыва.



The screenshot displays the 'Отзывы о банках' (Reviews about banks) section on the banki.ru website. At the top, it shows 'Банков: 330' (Banks: 330) and 'Отзывов: 1 089 374' (Reviews: 1 089 374). Below this are three filter buttons: 'Услуги Все' (Services All), 'Отзывы Проверенные' (Reviews Verified), and 'Оценка Любая оценка' (Rating Any rating). A search bar with the text 'Найти банк' (Find bank) is also present. Two review cards are visible. The first card is for 'Альфа Банк' (Alfa Bank) with a rating of 4, titled 'Положительный опыт обслуживания в Альфа банке' (Positive service experience in Alfa Bank). The second card is for 'Тинькофф' (Tinkoff) with a rating of 5, titled 'Отличное обслуживание!' (Excellent service!).

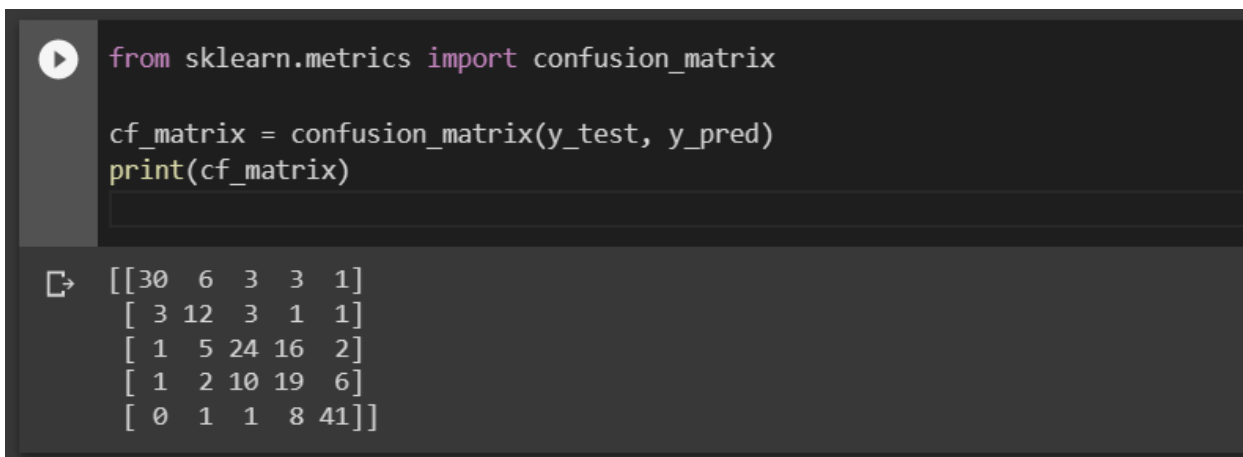
Рисунок 14 – Сервис banki.ru с отзывами клиентов

В качестве обучающей выборки были использованы первые 2000 отзывов клиентов на банковские услуги. Дополнительные 200 отзывов были

использованы для тестирования точности работы классификатора.

Для построения матрицы ошибок на тестовой выборке производится сопоставление реальных оценок `y_test` клиентов и оценок `y_pred` спрогнозированных классификатором.

Для построения матрицы ошибок используется метод `confusion_matrix`, реализованный в библиотеке `sklearn`. Программный код для построения матрицы ошибок представлен на рисунке 15.



```
from sklearn.metrics import confusion_matrix

cf_matrix = confusion_matrix(y_test, y_pred)
print(cf_matrix)
```

```
[[30  6  3  3  1]
 [ 3 12  3  1  1]
 [ 1  5 24 16  2]
 [ 1  2 10 19  6]
 [ 0  1  1  8 41]]
```

Рисунок 15 – Построение матрицы ошибок

Для визуализации матрицы ошибок в виде диаграммы с цветовой шкалой и поясняющими текстовыми подписями используется библиотека `matplotlib`, а также надстройка к этой библиотеке под названием `seaborn`.

Для создания диаграммы с цветовой шкалой используется метод `heatmap` из надстройки `seaborn`. При этом выбрана нейтральная голубая цветовая схема, которая задается через параметр `color`.

Задание заголовка диаграммы осуществляется с помощью метода `set_title()`, а задание подписей к осям задается с помощью методов `set_xlabel()` и `set_ylabel()`.

Программный код для визуализации матрицы ошибок показан на рисунке 16.

```
import seaborn as sns
import matplotlib.pyplot as plt

ax = sns.heatmap(cf_matrix, annot=True, cmap='Blues')

ax.set_title('Confusion Matrix with labels\n');
ax.set_xlabel('\nPredicted Class')
ax.set_ylabel('Actual Class ');

ax.xaxis.set_ticklabels(['1', '2', '3', '4', '5'])
ax.yaxis.set_ticklabels(['1', '2', '3', '4', '5'])

plt.show()
```

Рисунок 16 – Программный код для визуализации матрицы ошибок

Анализ диаграммы матрицы ошибок позволяет определить при определении какого класса текстов у классификатора возникает больше всего проблем. У идеального классификатора, не совершающего ошибок, все значения, кроме тех, что расположены на главной диагонали равны нулю.

В нашем случае диаграмма матрицы ошибок для классификатора, прогнозирующего оценки пользователей по 5 бальной шкале на основе анализа текста их отзывов из сервиса banki.ru представлена на рисунке 17.

Как видно из диаграммы, наименьше всего трудностей возникает при определении оценок «1» и «5» по данным из тестовой выборки. Это видно по элементам с наиболее темным фоном на главной диагонали диаграммы.

Наибольшее количество сложностей у классификатора возникает при определении оценок «2», «3» и «4». Как видно, классификатор часто путает эти классы друг с другом.

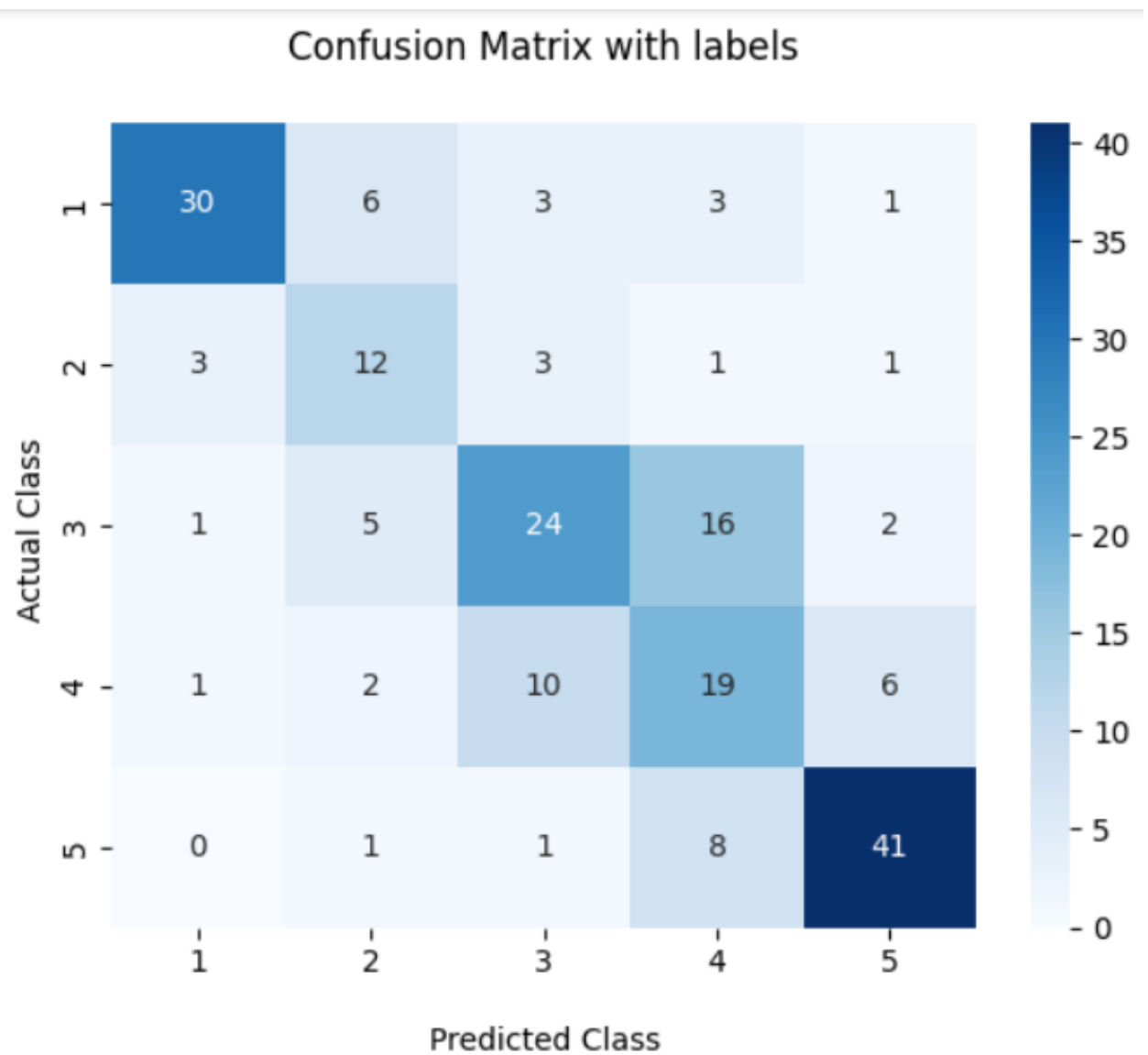


Рисунок 17 – Матрица ошибок при тестировании классификатора в случае 5 классов

Для того, что определить точность классификации отзывов пользователей, а также понять какие классы хуже всего поддаются прогнозированию можно воспользоваться методом `classification_report`, реализованным в библиотеке `sklearn`.

Данный метод рассчитывает точность работы классификатора для каждого класса отдельно, а также рассчитывает итоговую точность классификации на тестовой выборке данных.

```

from sklearn.metrics import classification_report

target_names = ['class 1', 'class 2', 'class 3', 'class 4', 'class 5']
print(classification_report(y_test, y_pred, target_names=target_names))

```

	precision	recall	f1-score	support
class 1	0.86	0.70	0.77	43
class 2	0.46	0.60	0.52	20
class 3	0.59	0.50	0.54	48
class 4	0.40	0.50	0.45	38
class 5	0.80	0.80	0.80	51
accuracy			0.63	200
macro avg	0.62	0.62	0.62	200
weighted avg	0.65	0.63	0.64	200

Рисунок 18 – Формирование отчета по результатам тестирования классификатора

На основе полученного отчета (рисунок 18) можно сделать следующие **выводы**:

- точность распознавания отзывов с оценкой «1» составляет 86%;
- точность распознавания отзывов с оценкой «5» составляет 80%;
- точность распознавания отзывов с оценкой «3» составляет 59%;
- точность распознавания отзывов с оценкой «2» составляет 46%;
- точность распознавания отзывов с оценкой «4» составляет 40%;
- суммарная точность работы классификатора составляет 63%.

Т.е. сгенерированный методом `classification_report` отчет подтверждает выводы сделанные на основе анализа диаграммы классов.

Анализ причин возникновения ошибок при классификации отзывов показал, что ключевыми проблемами, снижающих точность получаемых результатов являются:

- наличие в тексте комментариев сарказма. Например, словосочинение «спасибо банку» может присутствовать, как в позитивном комментарии, так

и в негативном («спасибо банку, за то, что несколько часов провел в очереди»). Предполагается, что точность классификации можно повысить вручную отсеяв комментарии с сарказмом из обучающей выборки данных.

- неоднозначность оценок от разных пользователей. Затрагивая одну и ту же проблему, разные пользователи дают различные оценки обслуживания. Например, озвучивая в текстовом комментарии проблему очередей одни пользователь ставит оценку «2», а другой пользователь - «3». Такая неоднозначность приводит к снижению точности классификации комментариев пользователей. Для решения данной проблемы в работе предлагается перемаркировать классы, снизив их количество: отзывы с оценками «5» и «4» считать «положительными» (class 0), а отзывы с оценками «3», «2», «1» считать «отрицательными» (class 1) (рисунок 19).

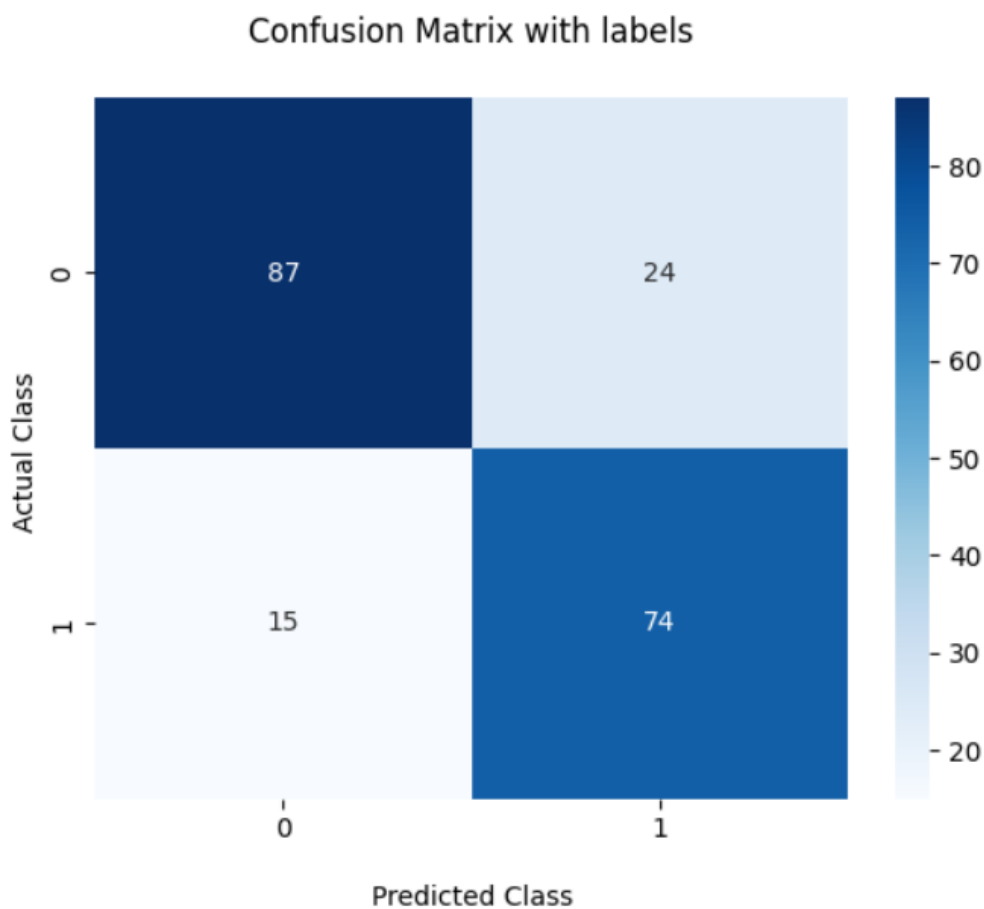


Рисунок 19 – Матрица ошибок при тестировании классификатора в случае 2 классов

Повторив обучение и тестирование классификатора комментариев пользователей с учетом перемаркировки классов были получены результаты представленные на рисунках 19 и 20.

```
from sklearn.metrics import classification_report

target_names = ['class 0', 'class 1']
print(classification_report(y_test, y_pred, target_names=target_names))
```

	precision	recall	f1-score	support
class 0	0.85	0.78	0.82	111
class 1	0.76	0.83	0.79	89
accuracy			0.81	200
macro avg	0.80	0.81	0.80	200
weighted avg	0.81	0.81	0.81	200

Рисунок 20 – Формирование отчета по результатам тестирования классификатора после перемаркировки классов

На основе полученного отчета (рисунок 18) можно сделать следующие выводы:

- точность распознавания отзывов относящихся к class 0 (позитивные отзывы) составляет 85%;
- точность распознавания отзывов относящихся к class 1 (негативные отзывы) составляет 76%;
- суммарная точность работы классификатора составляет 81%.

Таким образом удалось добиться определения тональности комментариев пользователей с итоговой точностью 81%, что является хорошим показателем.

Выводы по главе 3

Сформулируем выводы по третьей главе работы:

- в ходе анализа литературных данных установлено, что изучение и совершенствование сентиментального анализа является актуальной задачей в области искусственного интеллекта, так как позволяет автоматизировать процессы, связанные с обработкой текстовых данных;
- анализ тональности текста в сфере бизнеса может применяться для автоматизированного определения по отзывам пользователей их отношения к предоставляемым компанией услугам;
- в научной литературе есть примеры применения сентиментально анализа для определения оценки по тексту рецензии к фильму, а также прогнозирования оценки ресторана от клиента по содержанию его текстового отзыва;
- сентиментальный анализ основан на количественной оценке паттернов (шаблонных выражений), встречающихся в тексте и их связи с психологическим состоянием человека;
- проведен сравнительный анализ наиболее популярных систем для анализа тональности текста – Stanford NLP, Sentiment140, IBM Watson NLP и «ВААЛ». Ни одна из этих систем не является свободно распространяемым программным обеспечением;
- предложен алгоритм определения тональности текста, который включает в себя преобразование текстовых данных в векторный вид при помощи модели векторизации Bag-of-words и анализ полученных векторов с помощью классификатора XGBoost classifier;
- в предложенном алгоритме этап векторизации текста включает в себя фильтрацию текста от небуквенных символов, удаление стоп-слов, подбор параметров векторизации и обучение модели векторизации, заключающееся в построении словаря уникальных слов;

– в предложенном алгоритме этап классификации текста включает в себя формирование обучающей выборки, подбор параметров классификатора и обучение классификатора методом градиентного бустинга.

– на языке программирования python разработано программное обеспечение для определения тональности текстов. Для преобразования текстов в числовой формат применяется подход Bag-of-words, а в качестве классификатора используется XgBoost Classifier.

– предложена методика тестирования эффективности алгоритмов распознавания тональности текста, которая включает себя проверку работы классификатора на тестовой выборке данных, построение по результатам тестирования матрицы, которая впоследствии используется для корректировки работы алгоритма под конкретную выборку данных.

– разработанное программное обеспечение протестировано на отзывах клиентов о качестве предоставляемых банковских услуг, собранных с сайта banki.ru. При этом точность классификации отзывов пользователей составляет 81%.

Заключение

При выполнении бакалаврской работы были получены следующие результаты:

- на языке программирования python разработано программное обеспечение для определения тональности текстов. Для преобразования текстов в числовой формат применяется подход Bag-of-words, а в качестве классификатора используется XgBoost Classifier.

- предложена методика тестирования эффективности алгоритмов распознавания тональности текста, которая включает себя проверку работы классификатора на тестовой выборке данных, построение по результатам тестирования матрицы, которая впоследствии используется для корректировки работы алгоритма под конкретную выборку данных.

- разработанное программное обеспечение протестировано на отзывах клиентов о качестве предоставляемых банковских услуг, собранных с сайта banki.ru. При этом точность классификации отзывов пользователей составляет 81%.

Список используемой литературы и используемых источников

1. Левченко, С.В. Разработка метода кластеризации слов по смысловым характеристикам с использованием алгоритмов Word2Vec / Левченко С.В. // Новые информационные технологии в автоматизированных системах. 2017. №20. URL: <https://cyberleninka.ru/article/n/razrabotka-metoda-klasterizatsii-slov-po-smyslovym-harakteristikam-s-ispolzovaniem-algoritmov-word2vec> (дата обращения: 05.04.2023).

2. Нугуманова, А.Б. Обогащение модели Bag-of-words семантическими связями для повышения качества классификации текстов предметной области / А.Б. Нугуманова, И.А. Бессмертный, П. Пецина, Е.М. Байбурин // Программные продукты и системы. 2016. №2 (114). URL: <https://cyberleninka.ru/article/n/obogaschenie-modeli-bag-of-words-semanticheskimi-svyazyami-dlya-povysheniya-kachestva-klassifikatsii-tekstov-predmetnoy-oblasti> (дата обращения: 06.04.2023).

3. Полозов, И.К. Применение технологии Word2Vec в задаче выделения инверторов тональности / Полозов И.К., Волкова И.А. // МНИЖ. 2020. №4-1 (94). URL: <https://cyberleninka.ru/article/n/primenenie-tehnologii-word2vec-v-zadache-vydeleniya-invertorov-tonalnosti> (дата обращения: 19.04.2023).

4. Проскурин, А.А. Объектно-ориентированная реализация обработки текста на основе алгоритма continuous bag of words / Проскурин А.А., Авсеева О.В. // Объектные системы. 2016. №13. URL: <https://cyberleninka.ru/article/n/obektno-orientirovannaya-realizatsiya-obrabotki-teksta-na-osnove-algoritma-continuous-bag-of-words> (дата обращения: 01.04.2023).

5. Рожкин, П.А. Конструирование системы интеллектуального поиска ответов на вопросы обучающихся на онлайн-курсе на основе WORD2VEC / Рожкин Павел Александрович, Нехаев Игорь Николаевич,

Маркин Кирилл Анатольевич // ИАС. 2018. №1. URL: <https://cyberleninka.ru/article/n/konstruirovaniye-sistemy-intellektualnogo-poiska-otvetov-na-voprosy-obuchayuschih-sya-na-onlayn-kurse-na-osnove-word2vec> (дата обращения: 24.04.2023).

6. Ромашко, Д.А. Применение word2vec в задаче кластеризации оперонов / Ромашко Дмитрий Александрович, Медведев Александр Юрьевич // Программные системы и вычислительные методы. 2018. №1. URL: <https://cyberleninka.ru/article/n/primenenie-word2vec-v-zadache-klasterizatsii-operonov> (дата обращения: 11.04.2023).

7. Тарасова, А.Н. Сентиментальный анализ постов в социальных сетях посредством Python / Тарасова А.Н., Иванов К.О. // Символ науки. 2022. №3-1. URL: <https://cyberleninka.ru/article/n/sentimentalnuy-analiz-postov-v-sotsialnyh-setyah-posredstvom-python> (дата обращения: 20.04.2022).

8. Dragoni, M. The FeatureSent System at ESWC-2018 Challenge on Semantic Sentiment Analysis / Mauro Dragoni // Semantic Web Evaluation Challenge SemWebEval 2018: Semantic Web Challenges. – Springer Nature Switzerland AG, 2018. – pp. 216–231.

9. Kharlamov, A. Dynamic Semantic Network Analysis of Unstructured Text Corpora / Alexander Kharlamov, Galina Gradoselskaya, Sofia Dokuka // International Conference on Analysis of Images, Social Networks and Texts AIST 2017: Analysis of Images, Social Networks and Texts. – Springer International Publishing AG, 2018. – pp. 392–403.

10. Kharlamov, A.A. Social Network Sentiment Analysis and Message Clustering / Alexander A. Kharlamov, Andrey V. Orekhov, Svetlana S. Bodrunova, Nikolay S. Lyudkevich // International Conference on Internet Science, INSCI 2019: Internet Science. – Springer Nature Switzerland AG, 2019. – pp. 18–31.

11. Kosseim, L. Generating grammatical and lexical anaphora in assembly instructional texts / Leila Kosseim, Agnès Tutin, Richard Kittredge, Guy Lapalme

// European Workshop on Trends in Natural Language Generation EWNLG 1993: Trends in Natural Language Generation An Artificial Intelligence Perspective. – Springer-Verlag Berlin Heidelberg, 1996. – pp. 260–271.

12. Logumanov, A. Sentiment Analysis of Telephone Conversations Using Multimodal Data / Alexander Gafuanovich Logumanov, Julius Dmitrievich Klenin & Dmitry Sergeevich Botov // International Conference on Analysis of Images, Social Networks and Texts AIST 2018: Analysis of Images, Social Networks and Texts. – Springer Nature Switzerland AG, 2018. – pp. 88–98.

13. Maslova, N. Neural Network Doc2vec in Automated Sentiment Analysis for Short Informal Texts / Natalia Maslova, Vsevolod Potapov // International Conference on Speech and Computer SPECOM 2017: Speech and Computer. – Springer International Publishing AG, 2017. – pp. 546–554

14. Noll, T. Exploring the Syntonic Side of Major-Minor Tonality / Thomas Noll, David Clampitt // International Conference on Mathematics and Computation in Music MCM 2019. – Springer Nature Switzerland AG, 2019. – pp. 125–136.

15. Pang, B. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales / Pang Bo, Lee Lillian // Proceedings of the Association for Computational Linguistics (ACL), 2005. – pp. 115–124.

16. Pang, B. Thumbs up? Sentiment Classification using Machine Learning Techniques / Pang, Bo; Lee, Lillian; Vaithyanathan, Shivakumar // Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2002. – pp. 79–86.

17. Qu, Y. Exploring attitude and affect in text: Theories and applications. / Qu Yan, James Shanahan, Janyce Wiebe // In AAAI Spring Symposium, Technical report SS-04-07. – AAAI Press, Menlo Park, CA., 2004.

18. Samsonovich, A. On the Possibility of Regulation of Human Emotions via Multimodal Social Interaction with an Embodied Agent Controlled by eBICA-Based Emotional Interaction Model / Alexei V. Samsonovich, Zhen

Liu, Ting Ting Liu // International Conference on Artificial General Intelligence AGI 2022: Artificial General Intelligence. – Springer Nature Switzerland AG, 2023. – pp. 374–383.

19. Scholz, T. Extraction of Statements in News for a Media Response Analysis / Thomas Scholz, Stefan Conrad // International Conference on Application of Natural Language to Information Systems NLDB 2013: Natural Language Processing and Information Systems. – Springer-Verlag Berlin Heidelberg, 2013. – pp. 1–12.

20. Shamshurin, I. Data Representation in Machine Learning-Based Sentiment Analysis of Customer Reviews / Ivan Shamshurin // International Conference on Pattern Recognition and Machine Intelligence PReMI 2011. – Springer-Verlag Berlin Heidelberg, 2011. – pp. 254–260.

21. Shu, Y. Emotion Recognition from Music Enhanced by Domain Knowledge / Yangyang Shu, Guandong Xu // Pacific Rim International Conference on Artificial Intelligence PRICAI 2019: Trends in Artificial Intelligence. – Springer Nature Switzerland AG, 2019. – pp. 121–134.

22. Snyder, B. Multiple Aspect Ranking using the Good Grief Algorithm / Snyder Benjamin, Barzilay Regina // Proceedings of the Joint Human Language Technology/North American Chapter of the ACL Conference (HLT-NAACL), 2007 – pp. 300–307.

23. Stone P.J. The general inquirer: A computer approach to content analysis / Stone Philip J., Dexter C. Dunphy, and Marshall S. Smith // MIT Press, Cambridge, MA. – 1966. – p. 302

24. Timonin, A. Conceptual Modeling of the Social Environment for Information Support in Management Processes / Alexey Y. Timonin, Alexander M. Bershadsky, Alexander S. Bozhday // International Conference on Electronic Governance and Open Society: Challenges in Eurasia EGOSE 2019 – Part of the Communications in Computer and Information Science book series. – Springer Nature Switzerland AG, 2020. – pp. 138–151.

25. Turney, P. Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews / Turney Peter // Proceedings of the Association for Computational Linguistics, 2002. – pp. 417–424.

26. Wörner, J. Supporting Product Development by a Trend Analysis Tool Applying Aspect-Based Sentiment Detection / Janik Wörner, Daniel Konadl, Isabel Schmid, Susanne Leist // International Conference on Design Science Research in Information Systems and Technology DESRIST 2022: The Transdisciplinary Reach of Design Science Research. – Springer Nature Switzerland AG, 2022. – pp. 68–80.