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)

09.03.03 Прикладная информатика
(код и наименование направления подготовки)

Бизнес-информатика
(направленность (профиль) / специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему «Разработка приложения для классификации текста»

Обучающийся Д.Э. Седов
(Инициалы Фамилия) (личная подпись)

Руководитель к.т.н, В.С. Климов
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2022

Аннотация

Тема бакалаврской работы – «Разработка приложения для классификации текста».

Актуальность работы обусловлена необходимостью повышения уровня автоматизации бизнес-процессов, связанных с выполнением классификации текстовой информации при решении коммерческих задач.

Объектом исследования бакалаврской работы является процесс классификации текста.

Предметом исследования бакалаврской работы является система классификации текстовой информации.

Цель выпускной квалификационной работы – разработка универсальной системы классификации текстов, обеспечивающей повышение уровня автоматизации бизнес-процессов при анализе текстовой информации.

Методы исследования – технологии проектирования информационных систем, технологии программирования.

Практическая значимость бакалаврской работы заключается в разработке универсальной программной системы, обеспечивающей автоматизацию процесса классификации текста.

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

Результаты бакалаврской работы представляют научно-практический интерес и могут быть рекомендованы к использованию в лице бизнес-аналитиков в сфере автоматизации информационных систем.

Бакалаврская работа состоит из 41 страниц текста, 17 рисунков, 2 таблицы и 20 источников.

Оглавление

Введение

Классификация текстовой информации является одной из часто решаемых задач в области бизнес-аналитики. Например, когда требуется провести анализ оценки качества предоставляемых компанией услуг, обычно проводится классификация текста отзывов клиентов, которые они оставляют в различных социальных сетях и на форумах. При этом под классификацией понимается не только маркировка положительный отрицательной и отрицательных отзывов, но распознавание темы отзыва, т.е. определение конкретного типа услуги, на который написан отзыв.

На основе классификации текста отзывов можно определить:

- на какую услугу клиенты чаще всего обращают внимание;
- что клиентов подталкивает к использованию услуг компании;
- чем клиенты недовольны;
- какие проблемы возникают у клиентов при получении услуг компании;
- какое соотношение довольных и недовольных клиентов.

В широком смысле, процессы бизнес-аналитики направлены на превращение необработанных данных взятых из различных источников в наглядное представление, на основе которого можно сделать полезные выводы для оптимизации деятельности компании. Бизнес аналитика включает в себя выполнение четырех основных этапов – сбор данных, их анализ, визуализация и принятие на их основе управленческих решений.

На этапе сбора данных осуществляется агрегирование структурированных и неструктурированных данных из различных источников информации. Обычно на этом этапе используется метод ETL (Extract, Transform, Load). Собранные данные преобразуются в модель, расположенную в хранилище данных. При этом для хранилища выбирается такая структура данных, которая оптимизирована для быстрого извлечения и анализа информации с использованием специальных программных

комплексов.

На втором этапе используются специальные алгоритмы, предназначенные для обнаружения закономерностей в данных. Именно на этом этапе в случае необходимости решается задача классификации текстовой информации. И большинство исследований из области бизнес-аналитики направлено на повышение степени автоматизации выполнения этого этапа.

Третий этап направлен на формирование отчета с бизнес-аналитикой, основанного на использовании средств визуализации данных. При визуализации могут применяться интерактивные информационные панели, графики и диаграммы различных видов.

Четвертый этап является заключительным, и он направлен на устранение неэффективностей, возникающих в основной деятельности компании, выполнение стратегического планирования. Благодаря выполнению четвертого этапа компания осуществляет адаптацию своей деятельности к изменениям внешней бизнес среды.

Данная бакалаврская работа направлена на повышения степени автоматизации выполнения второго этапа бизнес-анализа. При этом для рассмотрения выбрана задача классификации текста. А в качестве инструмента для повышения степени автоматизации будут применяться алгоритмы машинного обучения.

Цель выпускной квалификационной работы – разработка универсальной системы классификации текстов, обеспечивающей повышение уровня автоматизации бизнес-процессов при анализе текстовой информации.

Для достижения данной цели необходимо выполнить следующие задачи:

- анализ технологий классификации текстов;
- разработка алгоритм классификации текста;

– реализация и тестирование программного обеспечения для классификации текстов.

Методы исследования – методы и технологии проектирования информационных систем, технологии программирования.

Практическая значимость бакалаврской работы заключается в разработке универсальной программной системы, обеспечивающей автоматизацию процесса классификации текста.

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

В первой главе приводится описание решаемой задачи классификации текста, рассматриваются технологии решения данной задачи на основе алгоритмов машинного обучения. Также в данной главе приводятся построенные диаграммы по моделированию процесса классификации текста и дается обзор исследований на тему классификации текстов. В тексте главы даны ссылки на современные научные статьи по теме исследования.

Вторая глава посвящена разработке практических решений по классификации текстов. Приводится описание таких этапов как предобработка и векторизация текста, разметка и балансировка данных.

В третьей главе представлен процесс разработки программного обеспечения для классификации текста, а также приведены результаты тестирования программного обеспечения.

В заключении описываются результаты выполнения выпускной квалификационной работы.

Бакалаврская работа состоит из 41 страниц текста, 17 рисунков, 2 таблицы и 20 источников.

Глава 1 Анализ современных подходов по классификации текста в бизнес-аналитике

1.1 Обзор литературных источников по классификации текста в бизнес-аналитике

По данным компании Microsoft, бизнес-аналитика – это процесс сбора и обработки прошлых и текущих данных, направленный на получение полезных сведений, направленных на принятие стратегических бизнес-решений. Процесс бизнес-аналитики включает в себя 4 этапа (рисунок 1):

- первый этап – сбор и преобразование данных из различных источников информации;
- второй этап – анализ данных, для обнаружения тенденций и закономерностей;
- третий этап – визуализация данных в виде отчета;
- четвертый этап – принятие управленческих решений на основе результатов анализа данных.

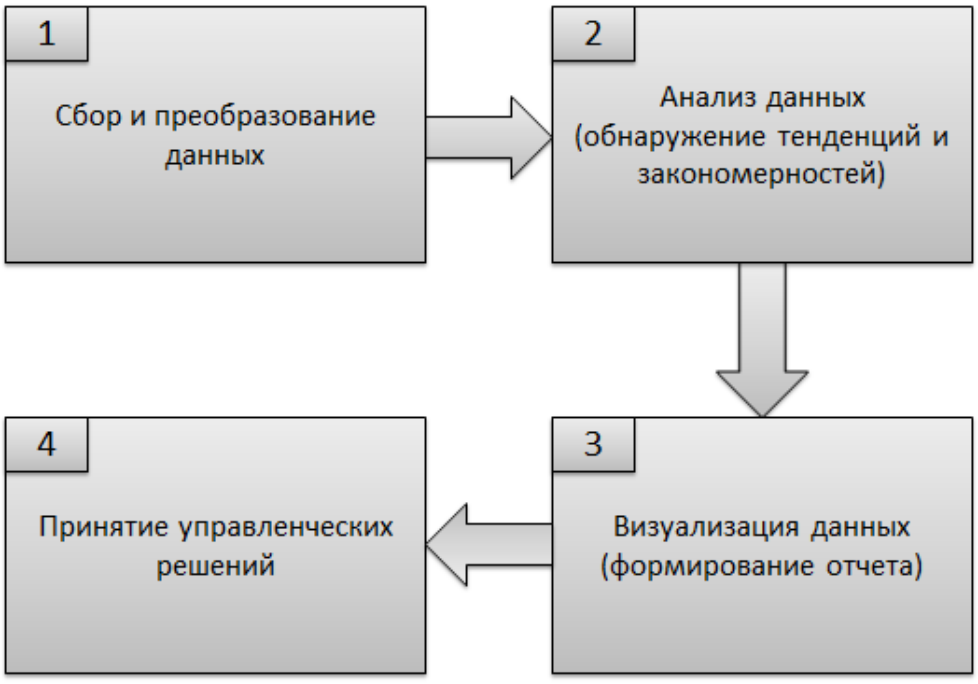


Рисунок 1 – Основные этапы процесса бизнес-аналитики

При выполнении второго этапа бизнес-аналитики решаются различные задачи анализа имеющихся данных. На этом этапе могут быть построены классификационные или регрессионные модели данных, проведен аффинитивный анализ данных или определена стратегия решения оптимизационной задачи. С развитием технологий на втором этапе бизнес-аналитики помимо анализа числовых данных, появилась возможность анализировать и текстовую информацию. Анализ текстовой информации может применяться, например, для мониторинга степени лояльности клиентов к компании на основе комментариев из социальных сетей. Для этого производится парсинг комментариев о компании из социальных сетей за определенный временной период и их последующей классификацией на основе изложенного текста к одному из классов: положительный отзыв, нейтральный отзыв и отрицательный отзыв. Как использовать такую полученную статистическую информацию при управлении предприятием решается на следующих этапах (третий и четвертый этап) бизнес-аналитики. В рамках же данной бакалаврской работы рассматривается только вопрос выполнения классификации текста с использованием современных технологий анализа данных.

В работе [7] описаны наиболее популярные способы обработки естественного языка. Также показаны практические способы по использования языка Python и библиотеки NLTK при решении задач анализа текстовых данных. Также в работе показаны примеры решения таких задач, как предобработка текста, извлечение информации из текстовых данных, классификация документов, оценка эмоциональной окраски текста, управление лингвистическими данными.

В статье [13] рассматриваются способы решения проблем классификации текстовой информации. Также в работе описываются алгоритмы токенизации и удаления стоп-слов и приводится информация о важности контроля регистра символов и правильной обработки аббревиатур.

В работе приведены примеры стемминга и лемматизации текстовых данных. Приводится сравнительный характеристика наиболее известных методов классификации и векторизации текстовых данных. Также приводится описание различных метрик, таких как точность, полнота, F-мера, а также микро и макроусреднение.

В статье [8] рассмотрено влияние применяемых способов векторизации документов на результаты классификации текстовых данных. В работе рассмотрены такие методы векторизации, как Term Frequency, TF-IDF, Word2Vec, Doc2Vec и Sparse Composite Document Vectors. При этом для построения классификаторов используются такие модели, как Logistic Regression (логистическая регрессия), Naive Bayes (Байесовский классификатор), Random Forests (случайный лес), SVM (метод опорных векторов). Точность работы различных сочетаний методов векторизации и классификации оценивалась с помощью F-меры. В результатах описано, что классификаторы Logistic Regression и Naive Bayes показали наилучшую точность в сочетании с методом векторизации Term Frequency. При этом наихудшие результаты показала связка методов Random Forest и Doc2Vec. Для классификатора SVM не было выявлено существенной разницы в выборе метода векторизации.

В статье [1] описаны особенности работы нейросетевого текстового классификатора Bert. Приведены рекомендации по настройке классификатора, а также описаны различия между его модификациями Bert_base и Bert_large.

В другой статье [12] приведены дополнительные особенности функционирования нейросетевого текстового классификатора Bert, а также показаны результаты его тестирования на 8 различных наборах данных. В работе описывается влияние структуры нейронной сети на особенности взаимодействия классификатора с текстовой информацией. Также описаны

последствия переобучения нейронной сети, влияющие на качество классификации.

1.2 Моделирование процесса классификации текста

Результаты анализа литературных источников показали, что все современные подходы по классификации текстовых данных основаны на настройке (обучении) классификаторов. В общем случае, классификатор текста – это модель, реализованная в виде программного модуля. При подаче на вход такого модуля текста, на выходе он возвращает метку класса (категорию текста) [2], [3], [4].

Настройка классификатора текста производится автоматически на основе текстовых примеров, хранящихся в обучающей выборке [14], [15], [16], [17]. Схема процесса обучения текстового классификатора показана ниже (рисунок 2). В процессе обучения текстового классификатора исходные текстовые данные подвергаются предобработке, в результате которой производится трансформация данных из текстового вида в векторный формат в соответствии с выбранным алгоритмом векторизации (Term Frequency, TF-IDF, Word2Vec, Doc2Vec или Sparse Composite Document Vectors) [18], [19], [20].

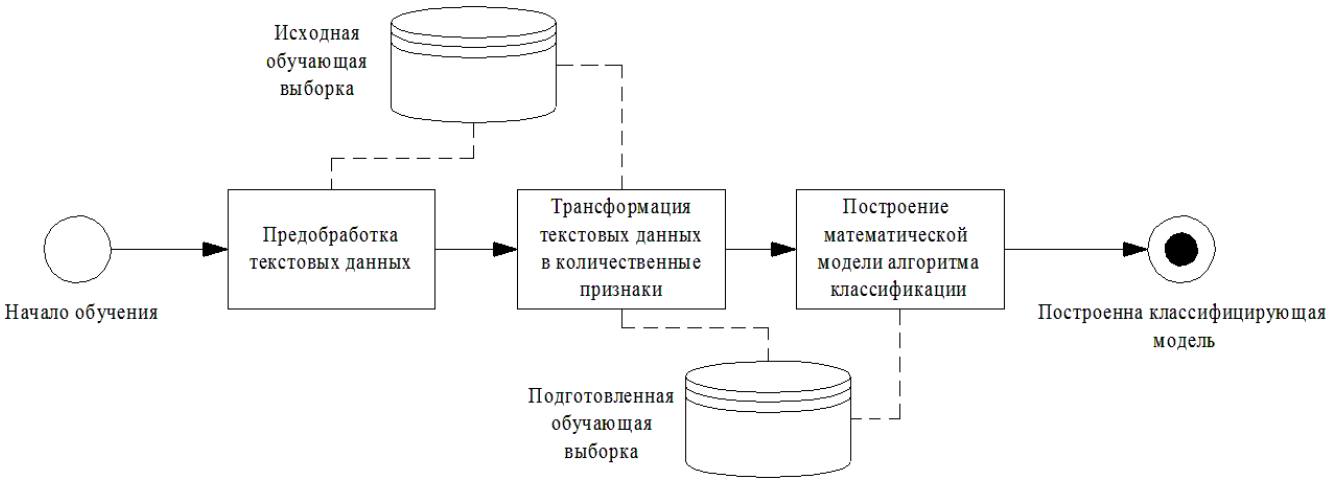


Рисунок 2 – Схема процесса обучения текстового классификатора

Обучение текстового классификатора выполняется в течение нескольких итераций. После того, как модель обучена, ее можно использовать для классификации произвольных текстовых данных. Схема процесса классификации нового документа с помощью обученной модели представлена ниже (рисунок 3). При классификации документа также, как и при обучении модели, осуществляется предобработка текстовых данных и их трансформация в векторное представление.

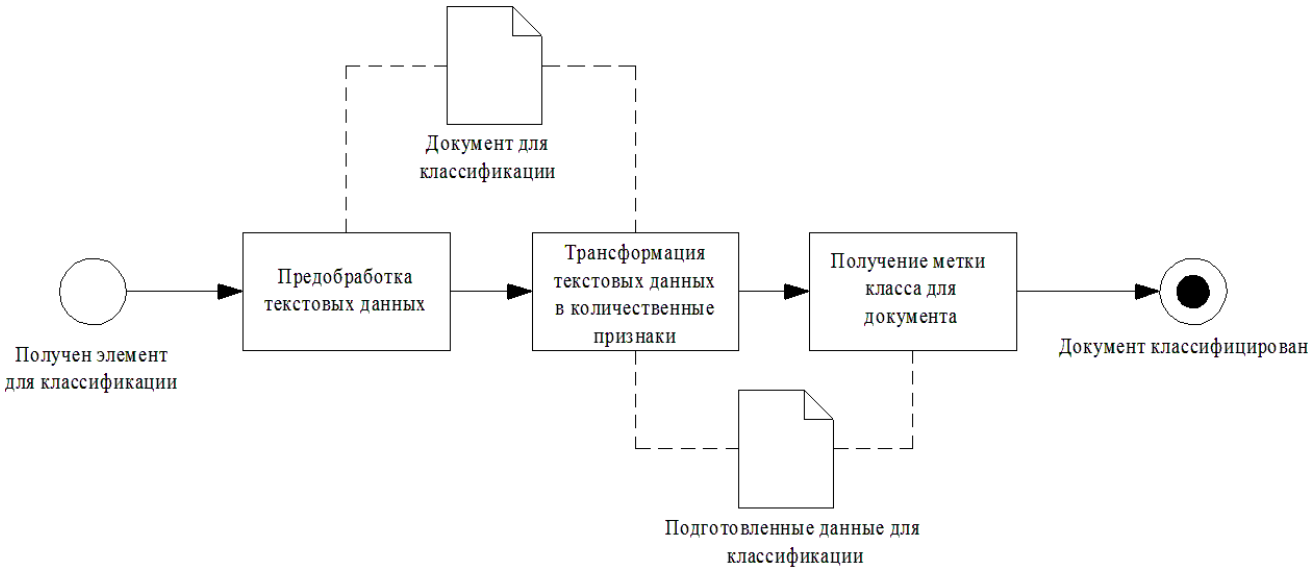


Рисунок 3 – Схема процесса классификации нового документа

На основе анализа процесса обучения классификатора и процесса классификации документа составлена диаграмма классов программного модуля для классификации текстовых данных (рисунок 4). Для правильной работы классификатор должен содержать в себе методы:

- загрузки обучающей выборки из файла;
- запуска процесса обучения классификатора;
- классификации нового документа.

При этом в полях класса «Классификатор» должна храниться: обучающая выборка, тип используемого классификатора, перечень стоп-слов и переключатель использования стоп-слов. Разработана диаграмма

последовательности для процесса взаимодействия текстового классификатора с аналитической информационной системой (рисунок 5).

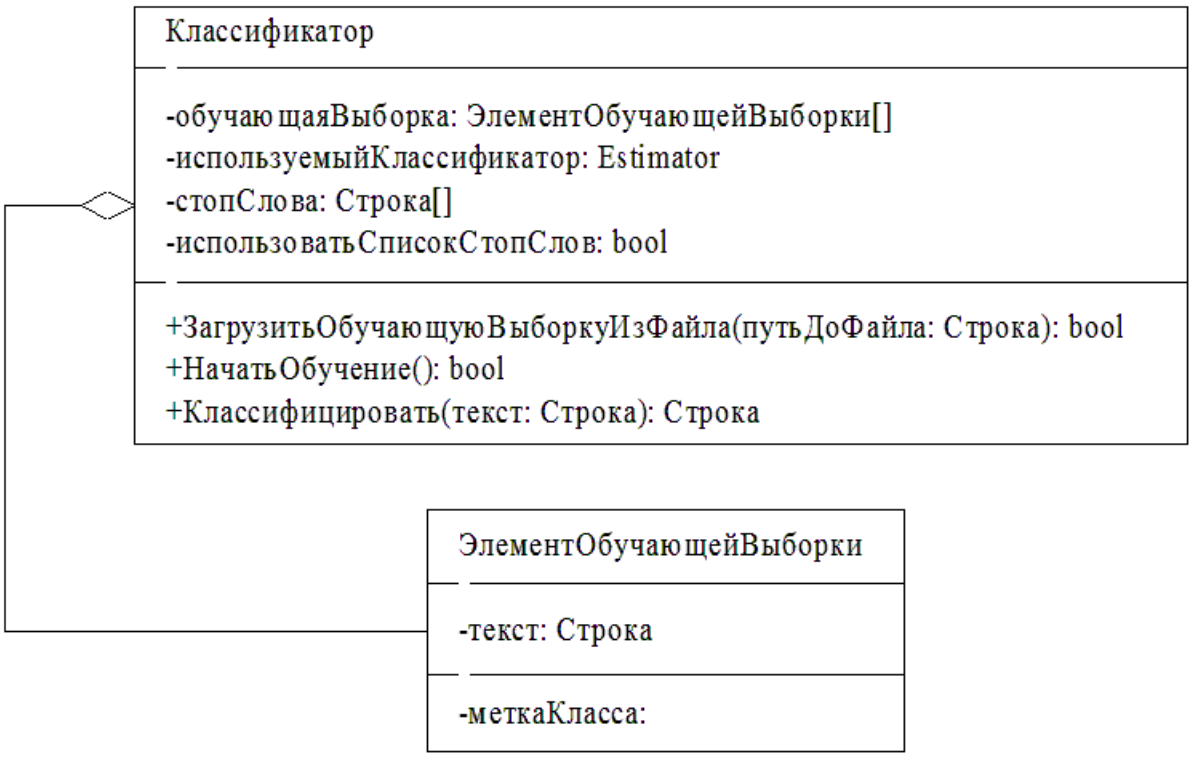


Рисунок 4 – Диаграмма классов текстового классификатора

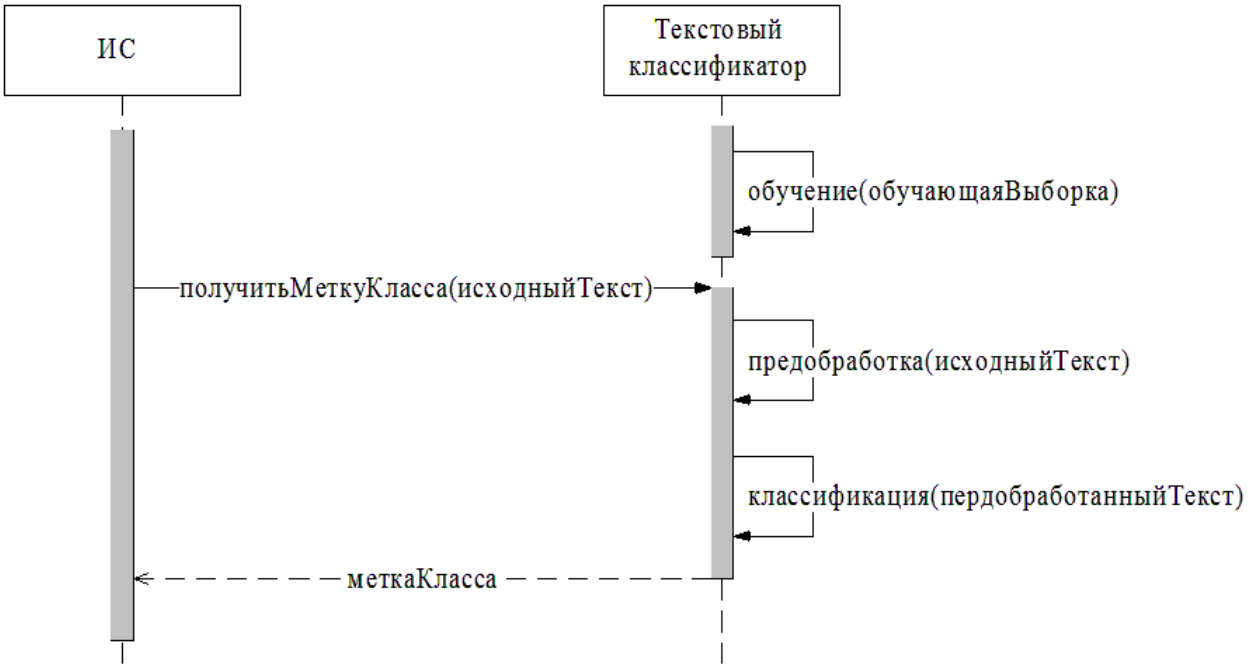


Рисунок 5 – Диаграмма последовательности для процесса взаимодействия текстового классификатора с аналитической информационной системой

Так как текстовый классификатор выполняет только свою частную задачу в процессе бизнес-аналитики, то он должен уметь взаимодействовать с аналитической информационной системой.

Выводы по главе 1

Приведем выводы по первой главе бакалаврской работы:

– в ходе анализа литературных источников установлено, что процесс бизнес-аналитики направлен на сбор, анализ и визуализацию данных о деятельности компании с целью принятия оптимальных управленческих решений.

– установлено, что классификация текста выполняется на втором этапе процесса бизнес-аналитики и может быть использована, например, для разметки отзывов и комментариев при анализе лояльности клиентов к компании.

– актуальной проблемой в области бизнес-аналитики является разработка и совершенствование программного обеспечения, способного автоматизировано выполнять процесс классификации текста с использованием современных подходов.

– проведено моделирование процесса классификации текста, в ходе которого построены схема процесса обучения текстового классификатора, схема процесса классификации нового документа, диаграмма классов текстового классификатора, диаграмма последовательности процесса взаимодействия текстового классификатора с аналитической информационной системой.

Глава 2 Проектирование системы для автоматизированной классификации текста

2.1 Описание задачи классификации

Задача классификации текста относится к области компьютерной лингвистики и ее можно описать следующим образом. Заранее известны все классы (категории) текстовых документов. В обучающей выборке для каждого класса содержатся примеры текстовых документов. На основе этих примеров, образующих обучающую выборку, требуется построить классификатор, способный самостоятельно определять класс любого текстового документа с заданной точностью.

Классификация текста активно применяется при распределении сайтов по тематическим каталогам, в адаптивных системах для фильтрации спама, в социальных сетях (при определении типа показываемой рекламы) и т. д. [11]

В рамках данной бакалаврской работы будет рассматриваться задача тематической классификации текстовых документов (новостных статей).

Задача классификации формализовано описывается следующим образом. Пусть имеется множество X описаний объектов. Фактически X – это прямоугольная матрица, в которой количество строк соответствует количеству объектов в обучающей выборке, а количество столбцов соответствует количеству m признаков объектов. Для каждого объекта обучающей выборки известен его класс и эти значения классов объединены в вектор Y . Таким образом, обучающую выборку можно записать как $X^m = \{(x_1, y_1),...,(x_m, y_m)\}$. На этой обучающей выборке необходимо настроить (обучить) алгоритм α, способный классифицировать объекты, т.е. выполнять отображение $\alpha : X \to Y$ [5].

В задаче тематической классификации новостных статей необходимо определить тематику статей в зависимости от их текстового содержания. В

качестве источника данных для обучающей выборки будет проводится парсинг статей из новостных агрегаторов Google News (рисунок 6) и Yandex новости (рисунок 7).

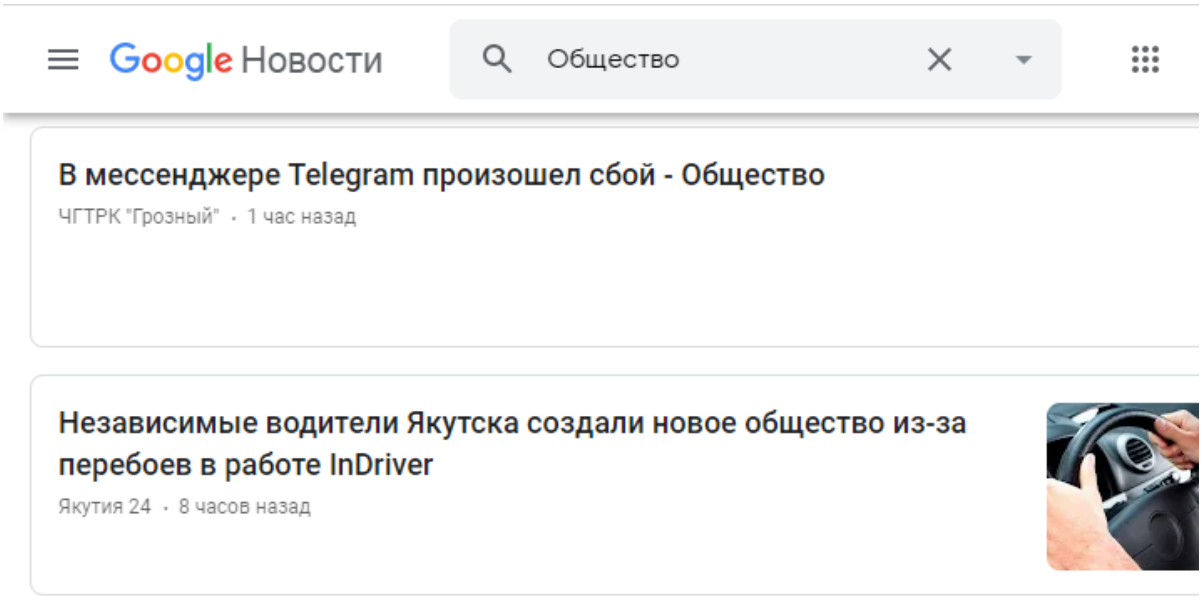


Рисунок 6 – Новостной агрегатор Google News, как источник данных для обучающей выборки

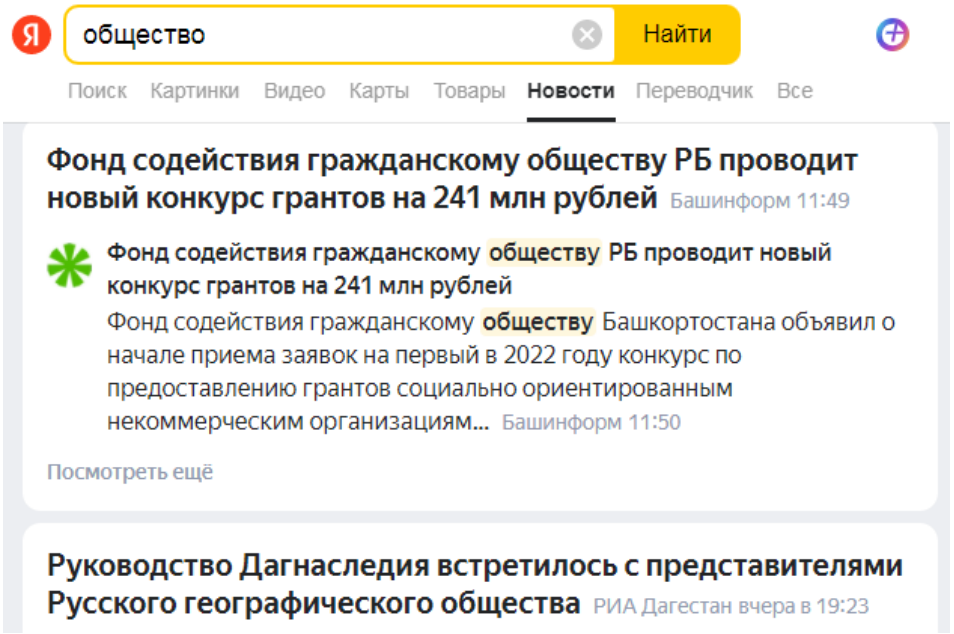


Рисунок 7 – Новостной агрегатор Yandex новости, как источник данных для обучающей выборки

В процессе анализа контента на новостных агрегаторах было выделено 7 основных тематических направлений, которые будут использоваться в качестве меток классов:

- общество;
- экономика;
- спорт;
- наука;
- технологии;
- новости культуры,
- традиции.

Задача классификации текстовых данных будет выглядеть следующим образом: на основе имеющих в новостных агрегаторах данных требуется обучить текстовый классификатор, способный автоматизировано определять тематическое направление анализируемого текста. При этом требуется определить наилучшее сочетание методов векторизации и классификации, обеспечивающее максимальную точность работы классификатора.

2.2 Предобработка и векторизация текста

Первым этапом работы с текстовыми данными является преобразование признакового пространства. Данный этап необходим по причине того, что стандартные методы классификации предназначены для работы с данными, представленными в виде числовых векторов. Поэтому, чтобы преобразовать текст в числовой вектор необходимо выполнить сначала предобработку текста, а затем к предобработанному тексту применить один из методов векторизации [9], [10].

В текстовом анализе есть свой используемый понятий аппарат, который отличается от привычного обывательского восприятия. Поэтому приведем ниже расшифровку используемых в анализе текста понятий. Под

понятием «Документ» подразумевается совокупность токенов, принадлежащих одной смысловой единице. Токен – это набор символов в документе, разделенный пробелами. Корпус – это анализируемое множество документов.

Предобработка текста направленна на устранение аномалий и неоднородностей в словах. Неоднородности в словах связаны с тем, что у одного слова может быть большое количество его форм. Кроме того, предобработка текста позволяет получать более компактные векторы на этапе векторизации и увеличивает точность текстовой классификации.

Процедура предобработки текста включает в себя выполнение следующих шагов: приведение символов к нижнему регистру, токенизация, удаление стоп-слов, стемминг и лемматизация.

Смена регистра символов на нижний является стандартным этапом предобработки слов, необходимым для приведения всех слов к единому формату.

После смены регистра выполняется токенизация. Это процесс разделения текста на слова на основе анализа пробелов и знаков препинания.

Так как в тексте наиболее часто встречаемыми словами являются союзы «и», «а» и «или», которые не оказывают значительного влияния на определения тематики текста, то они исключаются из рассмотрения. Данный этап носит название «удаление стоп-слов».

Заключительными этапами предобработки текста являются стемминг и леммантизация. Стемминг – это нахождение основы слова, а леммантизация – это приведение слова к нормальной форме. Например, при выполнении этих этапов токены в различных формах «столы», «столам», «столов» будут приведены к нормальной форме – «стол».

После завершения предобработки текста выполняется процедура векторизации. Все алгоритмы векторизации текста работают похожим образом: на вход подается корпус текстовых данных и в результате, на

выходе, генерируется множество векторов (по одному вектору на каждый документ). Путем сравнения отдельных полученных векторов можно определять тематическую близость документов и делать предположения о принадлежности документа к одному из классов.

В данной работе будут рассматриваться следующий список методов векторизации: TF-IDF, Word2Vec, Doc2Vec, FastText, GloVe, Universal-Sentence-Encoder и Bert.

Рассмотрим метод векторизации TF-IDF. Он основан на использовании одноименной статистической меры TF-IDF, направленной на оценку значимости слов с учетом всего содержимого текстового документа. В этом методе считается, что вес каждого слова пропорционален частоте его появления в документе, и обратно пропорционален частоте его употребления в остальных документах текстового корпуса. Размер числового вектора, описывающего текстовый документ, обычно равен размеру словаря. Если размер словаря слишком большой и это мешает эффективному анализу текстовых данных, то его необходимо сократить. Для этого в нем следует оставить только слова с высокой частотой появления в текстовом корпусе.

Мера TF-IDF состоит из двух элементов. Первый элемент – это значение Term Frequency, который показывает частоту появления слова в документе. Второй элемент – это значение Inverse Document Frequency, который показывает инверсию частоты документа. Эти элементы рассчитываются по формуле ниже.

$$TFtoken_i = \frac{n_i}{N_i} \quad IDFtoken = \log \frac{p}{P} \tag{2}$$

где n_i – сколько раз встречается токен в i-ом документе, N_i – общее количество токенов в i-ом документе, p – количество документов, в которых встречается токен, P – общее количеств документов.

Итоговое значение TF-IDF рассчитывается, как произведение элемента TF на элемент IDF:

$$TFIDF = TF \times IDF \quad (3)$$

Рассмотрим метод векторизации Word2Vec. Данный метод основан на использовании нейронных сетей, которые применяются для формирования векторного представлений слов. В методе Word2vec можно использовать следующие модели работы с данными: Skip-gram и Continuous Bag of Words (CBOW). Их отличия заключаются в том, что модель CBOW предсказывает текущее слово из окна окружающих контекстных слов, в то время как модель Skip-gram предсказывает окружающие контекстные слова по текущему слову.

«Принцип работы Word2Vec заключается в нахождение связей между контекстами слов согласно предположению, что слова, находящиеся в схожих контекстах, имеют тенденцию обозначать похожие вещи, т.е. быть семантически близкими. Более формально задача стоит так: максимизация косинусной близости между векторами слов (скалярное произведение векторов), которые появляются рядом друг с другом, и минимизация косинусной близости между векторами слов, которые не появляются друг рядом с другом. Рядом друг с другом в данном случае значит в близких контекстах» [1].

В методе Word2Vec для численной оценки сходства двух векторных представлений используется косинусная мера, которая рассчитывается по формуле, представленной ниже:

$$similarity(A,B) = \cos(\theta) = \frac{AB}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \quad (1)$$

где A и B – векторное представление признаков, $\cos(\Theta)$ – косинусная мера сходства.

На первом этапе работы Word2vec строит векторное представление для отдельных слов текстового документа. Для получения векторного

представления текстового документа проводится суммирование векторного представления слов с учетом частоты их появления в тексте. Таким образом, итоговое векторное представление документа рассчитывается так, как это показано на формуле ниже:

$$DV = \frac{\sum_{i=1}^{n} w2v_i \times tfidf_i}{n} \quad (4)$$

где DV – итоговый вектор документа, $w2v_i$ – векторное представление i-го слова в документе, $tfidf_i$ – частота встречаемости i-го слова во всем корпусе, n – количество слов в документе.

Рассмотрим метод векторизации Doc2Vec. Он является результатом развития метода Word2Vec, и направлен на применение той же логики к уровню текстовых документов, а не отдельных слов.

Каждый абзац и каждое слово отображается в уникальные векторы. Вектор абзаца и векторы слова усредняются или объединяются, чтобы предсказать следующее слово на основе контекста. Маркер абзаца действует как память, которая запоминает то, чего не хватает в текущем контексте или в теме абзаца.

В методе Doc2Vec используются следующие архитектуры данных: Distibuted Memory (DM) и Distributed Bag of Words (DBOW) (рисунок 8).

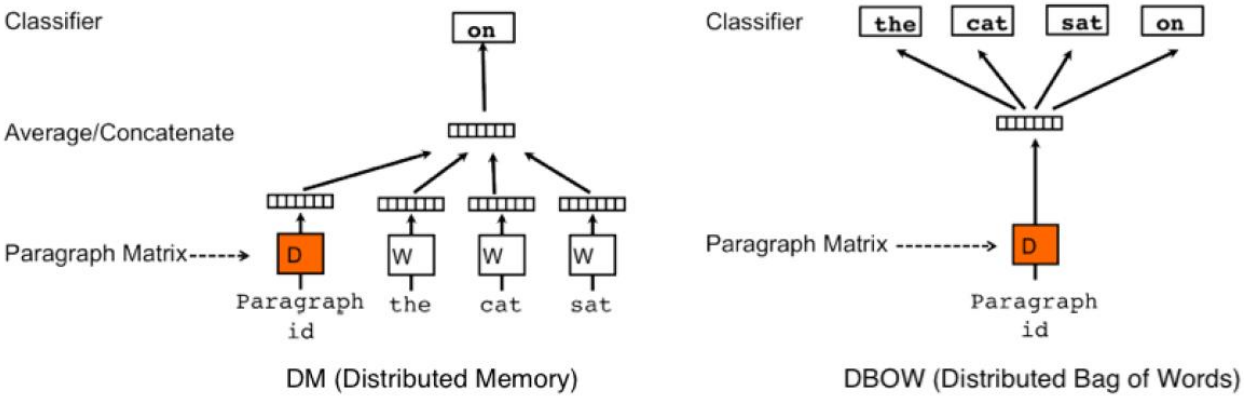


Рисунок 8 – Архитектуры DM и DBOW

В модели DM, которая аналогична CBOW в Word2Vec, векторы абзаца получаются путем обучения нейронной сети задаче определения центрального слова на основе контекстных слов и контекстного абзаца. В модели DBOW, которая аналогична Skip-gram в Word2Vec, векторы абзаца получаются путем обучения нейронной сети задаче прогнозирования распределения вероятности слов в абзаце по случайно выбранному слову из абзаца.

Рассмотрим метод векторизации FastText. В этом методе, для получения векторного представления слов применяются модели Skip-gram и CBOW. Также в этом методе реализована технология subword, которая трансформирует анализируемое слово в цепочки символов заданной длины. В результате такой трансформации получаются n-граммы. Например, для слова «замок», при значении n равном 3 можно получить следующие n-граммы [за, зам, амо, мок, ок] к которой также добавляется исходный набор символов [замок].

За счет использования технологии subword многократно увеличивается размерность признаков текста. С одной стороны, увеличение количества признаков теоретически увеличивает точность классификации текста, но существенно снижает скорость обучения классификатора. Для решения проблем большой размерности признаков обычно применяются хэш-индексы.

Рассмотрим метод векторизации GloVe. Этот метод основан на использовании алгоритмов обучения без учителя (название группы алгортмов) для векторизации слов. В качестве архитектуры данных в данном методе применяются матрицы совместного вхождения слов. Значения данной матрицы рассчитываются для всего текстового корпуса, и они определяют, как часто каждое слово появляется в контексте других слов. «GloVe минимизирует разницу между произведением векторов слов и логарифмом вероятности их совместного появления с помощью стохастического

градиентного спуска» [1].

На первом этапе работы GloVe строит векторное представление для отдельных слов текстового документа. Для получения векторного представления текстового документа проводится суммирование векторного представления слов с учетом частоты их появления в тексте. Таким образом, итоговое векторное представление документа рассчитывается так, как это показано на формуле ниже:

$$DV = \frac{\sum_{i=1}^{n} glove_i \times tfidf_i}{n} \quad (5)$$

где DV – итоговый вектор документа, $glove_i$ – векторное представление i-го слова в документе, $tfidf_i$ – частота встречаемости i-го слова во всем корпусе, n – количество слов в документе.

Рассмотрим метод векторизации Universal-Sentence-Encoder (USE). Данный метод кодирует текстовые документы в числовое векторное представление, которое впоследствии применяется при классификации текста, поиске семантического сходства, кластеризации и решении других задач обработки естественного языка.

Особенность данного метода заключается в том, что он подходит для работы с текстовыми документами небольшой и средней длины (от одного предложения до нескольких абзацев). Входными данными при использовании Universal-Sentence-Encoder является текст переменной длины, а на выходе всегда генерируется 512-мерный числовой вектор.

Рассмотрим технологию Bidirectional Encoder Representations from Transformers (BERT). Данная технология решает как задачу векторизации текстовой информации, так и задачу классификации текста. BERT работает на основе предобученной нейронной сети, разработанной компанией Google. Особенность использования BERT заключаются в том, что технология обучается сразу на основе всего набора слов, присутствующих в текстовых

документах. «Она предназначена для предварительного обучения глубоких двунаправленных представлений из немаркированного текста путем совместной обработки левого и правого контекста на всех уровнях. BERT обеспечивает наиболее плотные векторные представления на этапе кодирования текста. Он кодирует информацию о взаимных употреблениях слов, словосочетаний и других языковых единиц по отношению друг к другу, то есть учитывает контекст. Если поменять порядок слов в предложении, то вектора на выходе будут другими» [13].

2.3 Разметка и балансировка данных для обучения

Обучение текстовых классификаторов осуществляется на основе обучающей выборки данных. Обучающая выборка состоит из текстовых документов, причем класс каждого документа из выборки должен быть определен заранее, еще до начала процесса обучения.

В области классификации текстовых документов особое внимание уделяется балансировки классов обучающей выборки. Это означает, что объем текстовых данных в каждом классе обучающей выборки должен быть приблизительно одинаковым. Несбалансированная по классам обучающая выборка приводит к снижению точности работы обучаемых классификаторов.

В данной работе обучающая выборка собирались автоматизировано путем загрузки данных с новостных сервисов Google News и Yandex новости.

Каждой загруженной новостной статье соответствует набор тегов, которые были проставлены новостными сервисами. На основании тегов производилась разметка данных обучающей выборки. Из 151841 новостной статьи были размечены 69955 (рисунок 9).

```
'Экономика': "economy",

'Шоу-бизнес': "entertainment",
'Новости культуры': "entertainment",
'Культура': "entertainment",

'Наука':"science",
```

Рисунок 9 - Разметка новостный статей по “тегам”

Также была произведена балансировка классов. Самый малочисленный класс насчитывает 4845 статей, а самый обширный - 20141. Балансировка производилась путем удаления элементов обучающей выборки до тех пор, пока в каждом классе не осталось по 4845 элементов.

2.4 Применяемые методы классификации

Теперь, когда сформирована обучающая выборка и выбраны применяемые методы векторизации текстовых данных, необходимо выбрать модель классификации. В рамках данной работы рассматриваются следующие модели классификации: LogisticRegression, SVM, Single-layer perceptron (SLP), Bert и Gpt-2.

При обучении моделей классификации LogisticRegression, SVM, Single-layer perceptron будут использоваться текстовые данные обучающей выборки, прошедшие предобработку и процедуру векторизации с использованием различных методов. Таким образом, можно будет понять, как методы векторизации влияют на конечную точность текстового классификатора.

В модели Bert и Gpt-2 уже встроены собственные методы векторизации, основанные на предобученных нейронных сетях. Поэтому при использовании этих моделей не требуется подбирать метод векторизации и данные не требуют какой-либо дополнительной предобработки.

Коротко рассмотрим принципы работы моделей классификации.

Логистическая регрессия (LogisticRegression) – модель линейного классификатора, позволяющего оценивать вероятности принадлежности анализируемых объектов существующим классам. Вероятности принадлежности объекта к каждому классу оценивается на основе векторного предоставления признаков. Модель логистической регрессии используется в том случае, когда пространство значений может быть разделено границей или множеством границ на два и более класса. Параметры этих границ подбираются на основе данных, хранящихся в обучающей выборке. Графическая интерпретация логистической регрессии показана ниже (рисунок 10).

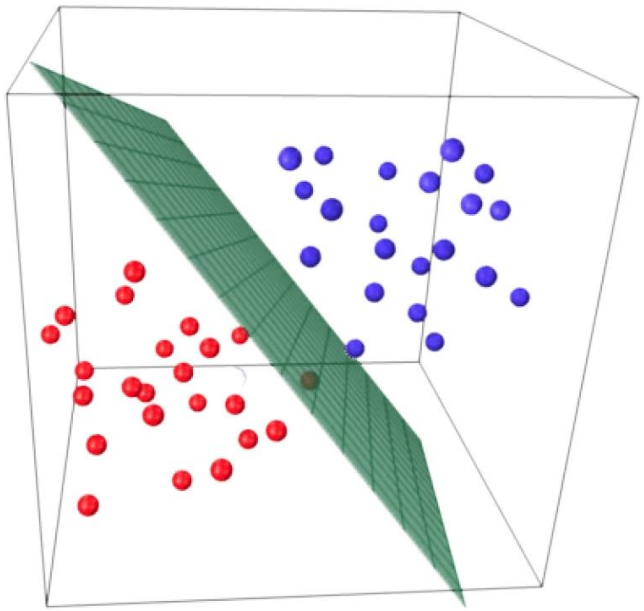


Рисунок 10 - Основная идея логистической регрессии

Метод опорных векторов (SVM) – это модель классификации данных, основанная на построении гиперплоскости, разделяющей признаковое пространство на классы. Параметры гиперплоскости определяются в процессе анализа данных из обучающей выборки. Метод опорных векторов может применяться при количестве классов больше или равное двум.

Однослойный перцептрон (Single-layer perceptron) – это модель

классификации данных, основанная на использовании нейронная сети. На вход нейронной сети подается векторное представление анализируемого текста, а на выходе нейронной сети определяется вероятность принадлежности анализируемого объекта к одному из классов.

Архитектура однослойного персептрона для задачи с 3 классами объектов показана на рисунке 11. Для настройки правильной работы такого классификатора применяется метод обратного распространения ошибки, который производит подбор весовых коэффициентов W.

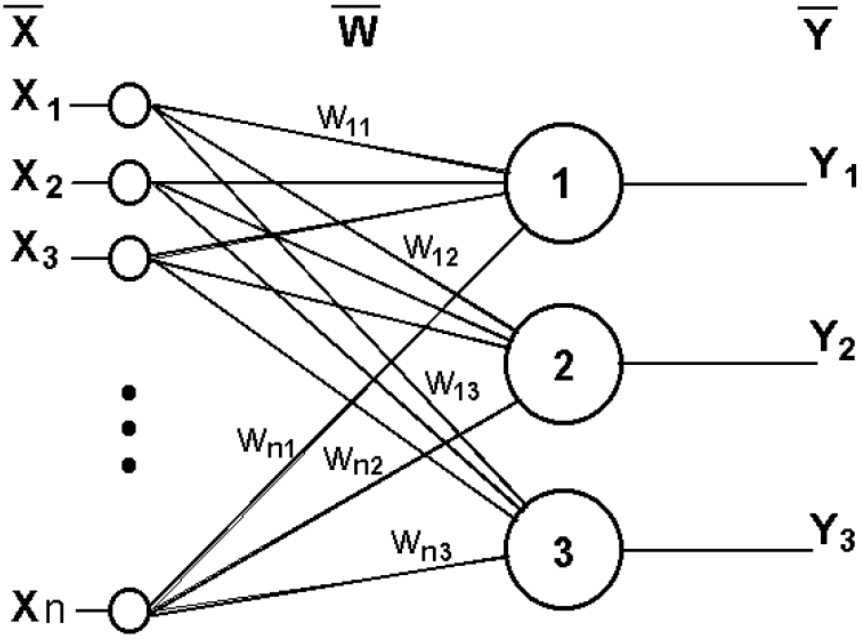


Рисунок 11 - Архитектура однослойного перцептрона

BERT – это предобученный классификатор данных, созданный компанией Google, работающий на основе нейронной сети. Нейронная сеть позволяет языковой модели изучать контекст слова на основе всех окружающих его слов. Предварительно обученная модель BERT, за счет добавления дополнительных слоев в нейронную сеть, может быть настроена на решение таких задач естественного языка, как генерирование ответов на вопросы, автореферирование текста и т.д. [7]. Кроме того, модель BERT является компонентом поисковой системы Google.

Для решения задачи классификации используется модифицированная модель BERT с одним дополнительным слоем для классификации данных.

Generative Pre-trained Transformer 2 (GPT-2) – это, также как и BERT, модель классификации текста, основанная на использовании нейронных сетей. Однако в этой модели используется другая архитектура нейронной сети.

Выводы по главе 2

Приведем выводы по второй главе бакалаврской работы:

− при классификации текста необходимо использовать сочетание методов двух разных видов: метода векторизации, который преобразует исходный текст документа в числовой вектор и метод классификации, который на основе числового вектора определяет класс текстового документа;

− рассмотрены особенности работы таких методов векторизации как TF-IDF, Word2Vec, Doc2Vec, FastText, GloVe, Universal-Sentence-Encoder, Bert, а также особенности работы таких методов классификации как LogisticRegression, SVM, Single-layer perceptron (SLP), Bert и GPT-2.

− установлено, что заранее не известно, какое сочетание методов покажет наилучшие результаты при классификации текста, поэтому разрабатываемое программное обеспечение направлено на сравнительное тестирование различных методов. При этом решаемая задача классификации – автоматизированное определение тематики текстовых документов (новостных статей).

Глава 3 Разработка приложения для классификации текста

3.1 Особенности реализации программного модуля

Последовательность работы программного обеспечения включает в себя выполнение следующих этапов:

- сбор новостных статей;
- предобработка данных;
- векторизация данных;
- обучение классификатора;
- классификация статей.

В качестве языка программирования для написания программного модуля был выбран Python, так как при его использовании становится доступно большое количество бесплатных библиотек, реализующих алгоритмы анализа текста.

Для разработки программного модуля использовалась облачная среда Colab, бесплатно предоставляемая компанией Google. Примечательно, что посредством данной среды предоставляется удаленный доступ к суперкомпьютерам Google, на которых, собственно, и выполняется написанный программный код.

Разработанный программный модуль обладает графическим интерфейсом пользователя, через который можно управлять параметрами сбора данных для обучающей выборки, выбирать используемые методы векторизации и классификации, получать статистические данные выполнения каждого этапа анализа, сравнивать различные сочетания методов векторизации и классификации по точности и проводить классификацию произвольно выбранного текстового документа.

Интерфейс программы разделен на две области: в левой области находится интерактивный список для перемещения между различными

блоками программного модуля, а в правой части находится содержимое блоков с различными элементами управления (кнопки, календарь для выбора дат, текстовые поля, списки с возможностью выбора элементов) (рисунок 12).

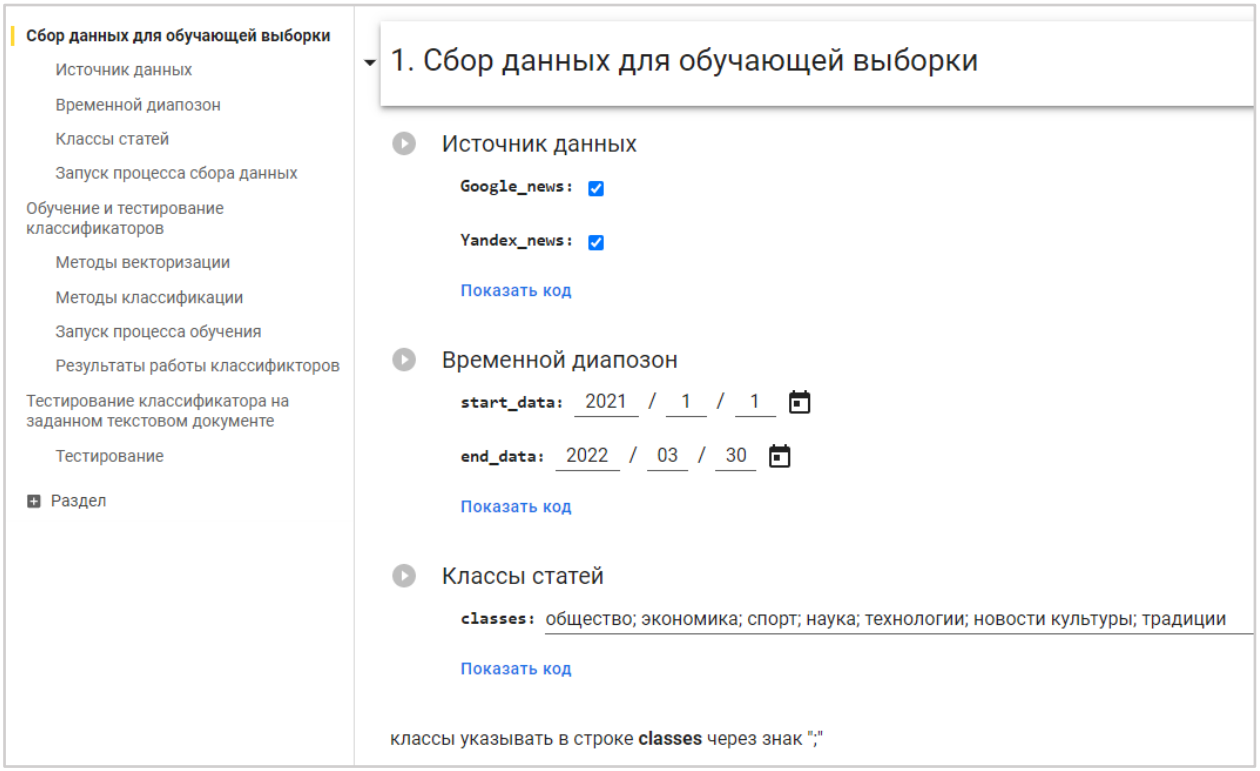


Рисунок 12 – Интерфейс программного модуля с разделом «Сбор данных для обучающей выборки»

Алгоритм работы с программным модулем следующий. Сначала необходимо собрать данные для формирования обучающей выборки, на основе которой будут обучаться текстовые классификаторы. Для этого в разделе «Сбор данных для обучающей выборки» выбираются источники данных для загрузки примеров новостных статьей. Предусмотрена возможность выбора нескольких источников: Google news и Yandex новости. Затем задается временной диапазон загружаемых статей. От диапазона зависит размер обучающей выборки. После, с помощью текстовое поля через знак «;» задаются классы текстовых статей (рисунок 12). Название классов

следует задавать без сокращений, так как эти строковые значения используются в API новостных агрегаторов для фильтрации новостей.

Процесс сбора параметров обучающей выборки запускается с помощью кнопки «Начать сбор данных» и может занимать длительное время (рисунок 13). Как только сбор данных для обучающей выборки завершится, на экран будет выведено: затраченное временя в секундах, количество загруженных текстовых документов и количество текстовых документов после балансировки классов. Балансировка классов – это выравнивание количества элементов в каждом классе путем удаления элементов из самых крупных классов.

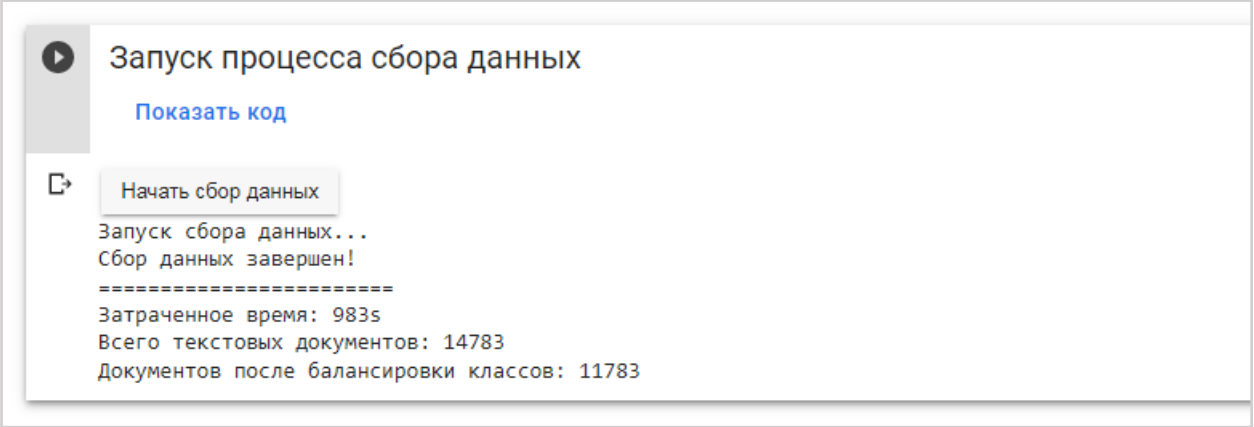


Рисунок 13 – Запуск процесса сбора данных

После того, как сформирована обучающая выборка, можно переходить к обучению текстовых классификаторов. Для настройки параметров обучения в программном модуле предусмотрен раздел «Обучение и тестирование классификаторов» (рисунок 14).

В этом разделе можно выбрать из списка используемые методы векторизации данных, а также применяемые методы классификации данных (рисунки 14 и 15).

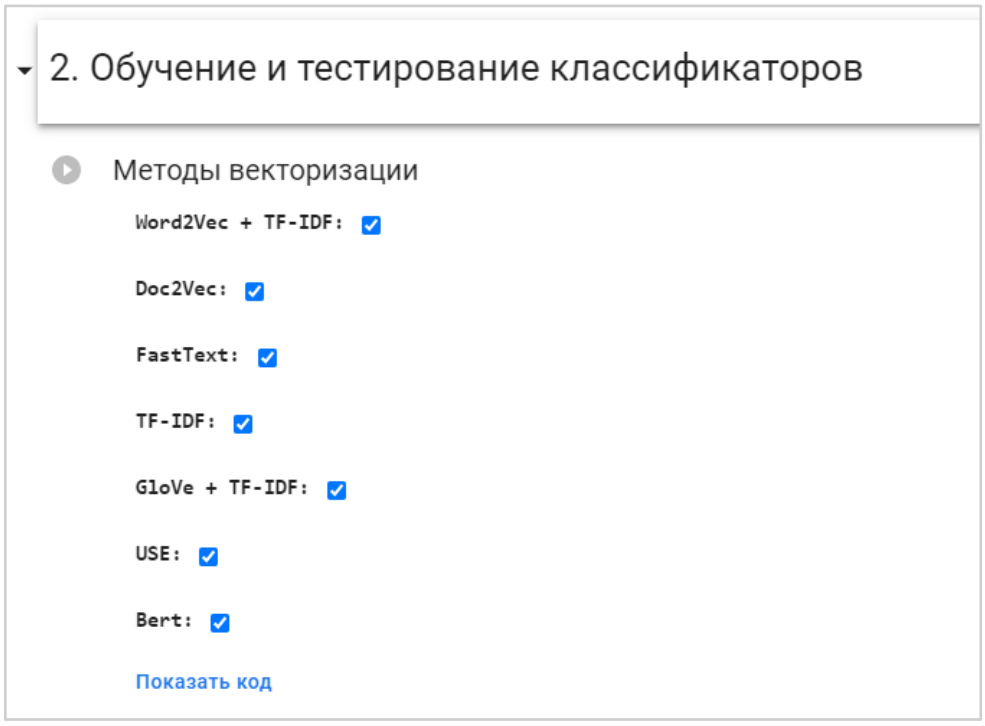


Рисунок 14 – Выбор методов векторизации

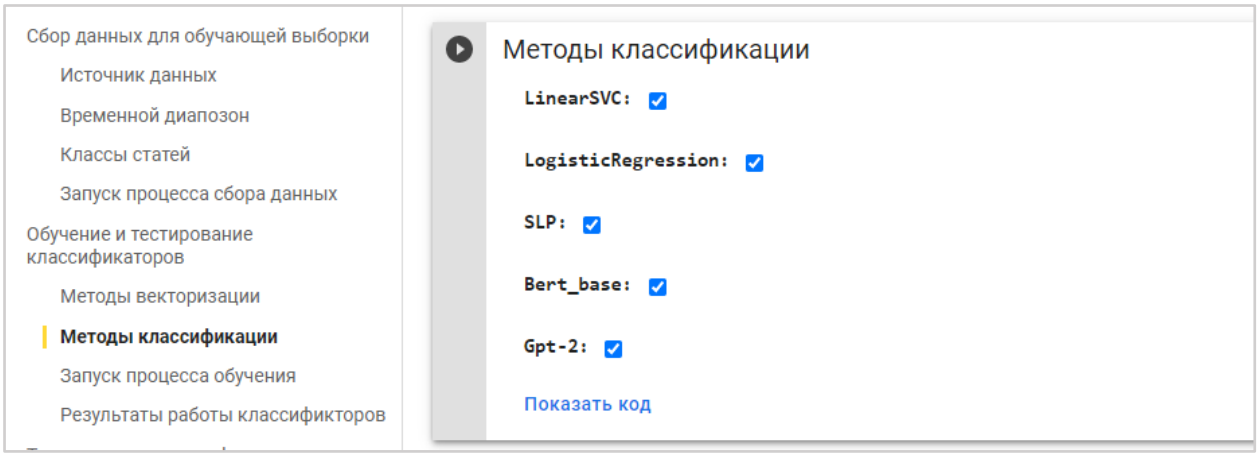
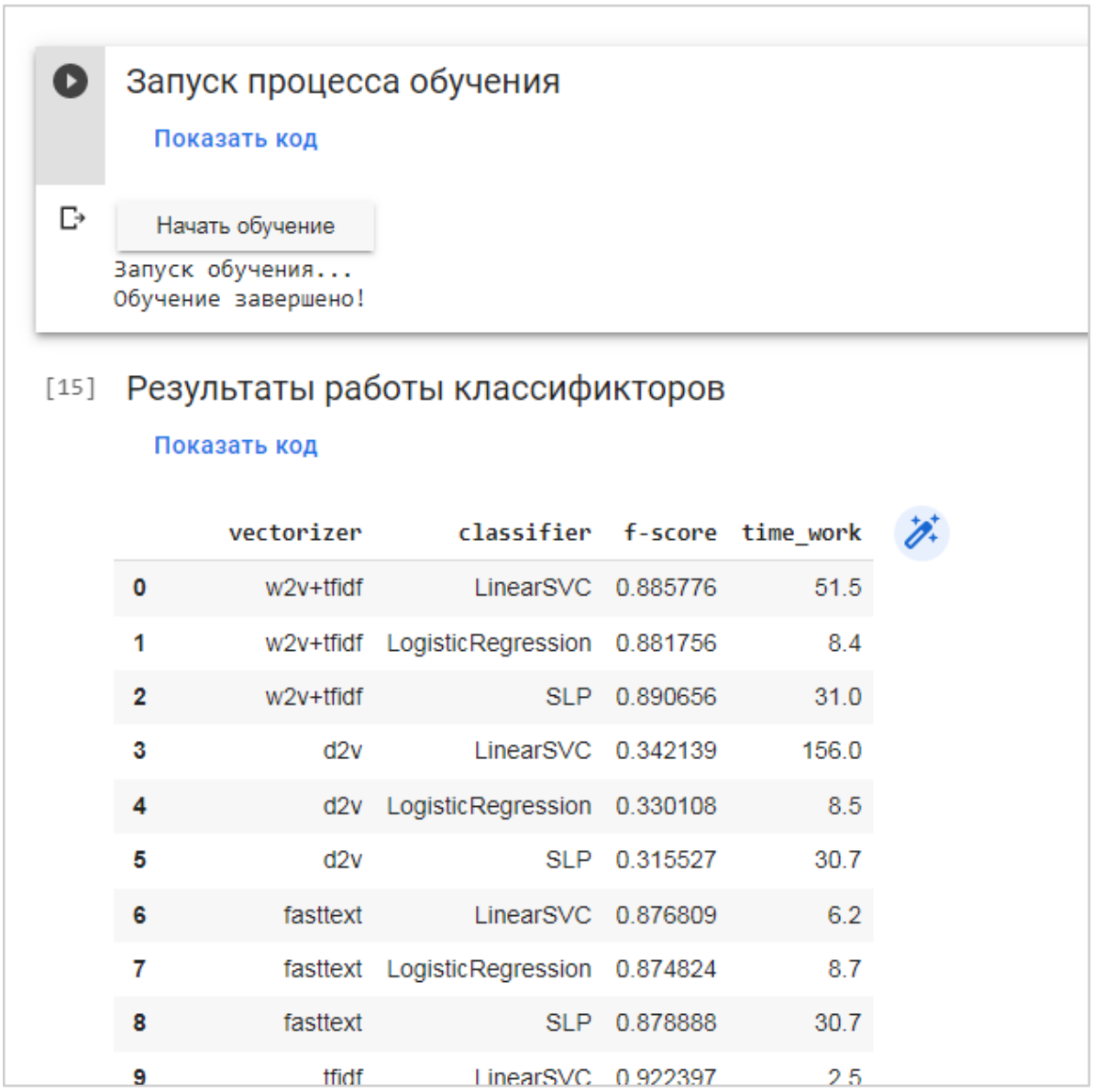


Рисунок 15 – Выбор методов классификации

После выбора используемых методов можно запустить процедуру обучения классификаторов, нажав на кнопку «Начать обучение» (рисунок 16). Процесс обучение классификаторов может занимать значительную часть времени, особенно если были выбраны все методы векторизации и классификации.

При окончании процесса обучения классификаторов запускается

процедура их тестирования. Тестирование проводится на исходном наборе данных (на обучающей выборке до балансировки).



	vectorizer	classifier	f-score	time_work
0	w2v+tfidf	LinearSVC	0.885776	51.5
1	w2v+tfidf	LogisticRegression	0.881756	8.4
2	w2v+tfidf	SLP	0.890656	31.0
3	d2v	LinearSVC	0.342139	156.0
4	d2v	LogisticRegression	0.330108	8.5
5	d2v	SLP	0.315527	30.7
6	fasttext	LinearSVC	0.876809	6.2
7	fasttext	LogisticRegression	0.874824	8.7
8	fasttext	SLP	0.878888	30.7
9	tfidf	LinearSVC	0.922397	2.5

Рисунок 16 – Запуск обучения классификаторов и таблица для сравнения результатов работы классификаторов

Результат тестирования классификаторов отображается в виде таблицы (рисунок 16), со следующими столбцами:

– vectorizer – применяемый метод векторизации в классификаторе;

– classifier – применяемый метод классификации в классификаторе;

– f-score – точность работы классификатора (F-мера), которая рассчитывается по формуле (6);

– time_work – время затраченное на тестирование классификатора.

В программном модуле предусмотрена возможность проверки работы обученных текстовых классификаторов на произвольном текстовом документе. Для этого необходимо воспользоваться элементами управления из раздела «Тестирование классификатора на заданном текстовом документе».

Пример классификации текстового документа (экономической статьи) показан на снимке экрана (рисунок 17).

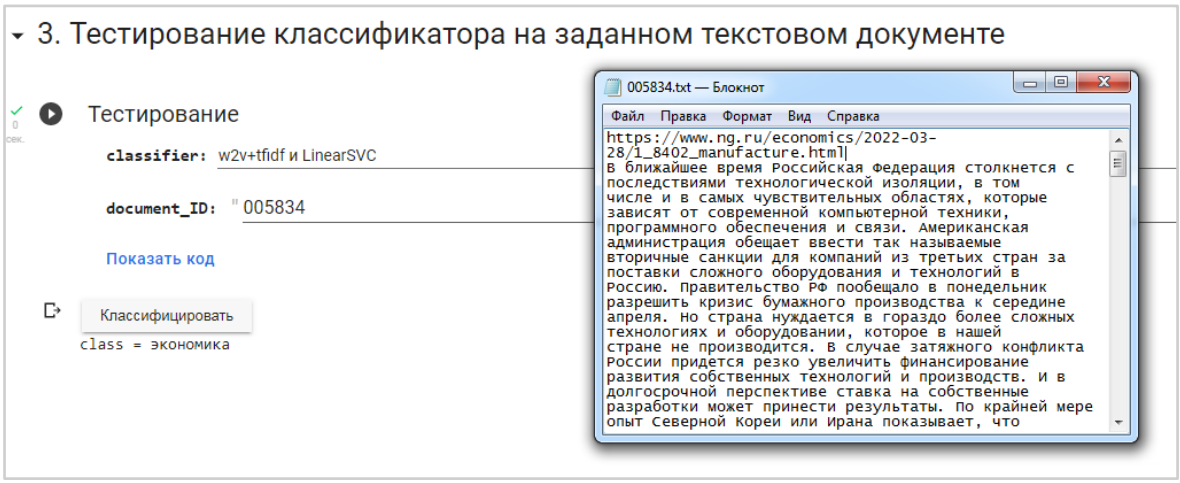


Рисунок 17 – Проверка выбранного классификатора на произвольном таксовом документе

Для этого в выпадающем списке «classifier» необходимо выбрать один из обученных на предыдущем этапе классификаторов. Классификаторы обозначаются путем соединения названий методов векторизации и классификации (например, классификатор «w2v+tdidf и LinearSVC»).

Чтобы загрузить анализируемый текст в программу, необходимо создать текстовый файл с расширением txt, туда скопировать анализируемый текст и разместить этот файл в папке с программным модулем. А в самой

программе в поле document_ID необходимо указать название текстового документа без расширения.

3.3. Результаты тестирования приложения

После построения всех классификаторов и векторизаторов проводились обучение моделей, оценка точности, времени работы, а также сравнительный анализ полученных результатов. Кроме того, была выведена корреляция между точностью классификации и выбором векторизатора.

Время обучения каждого классификатора с различными векторными моделями отличалось. Для оценки скорости работы оценивались пары векторизатор-классификатор, а также среднее время, которое затрачивает классификатор на обучение [6].

Для оценки точности была использована F-мера, представляющая собой гармоническое среднее между точностью *(precision)* и полнотой *(recall).*

Точность системы в пределах класса — это доля документов, действительно принадлежащих данному классу относительно всех документов, которые система отнесла к этому классу. Полнота системы — это доля найденных классификатором документов, принадлежащих классу относительно всех документов этого класса в тестовой выборке.

$$F = 2\frac{Precision \times Recall}{Precision + Recall}$$
$$Precision = \frac{TP}{TP + FP}$$
$$Recall = \frac{TP}{TP + FN} \quad (6)$$

где TP – истинно-положительное решение; TN – истинно-отрицательное решение; FP – ложно-положительное решение; FN – ложно-отрицательное решение.

В ходе исследования проведено 23 различных эксперимента. Ниже представлены полученные результаты (таблица 1)

Таблица 1 – результаты классификации

vectorizer	**classifier**	**f-score**	**time_work**
w2v+tfidf	LinearSVC	0,885776	51,5
	LogisticRegression	0,881756	8,4
	SLP	0,890656	31,0
d2v	LinearSVC	0,342139	156,0
	LogisticRegression	0,330108	8,5
	SLP	0,315527	30,7
fasttext	LinearSVC	0,876809	6,2
	LogisticRegression	0,874824	8,7
	SLP	0,878888	30,7
tfidf	LinearSVC	0,922397	2,5
	LogisticRegression	0,912016	65,0
	SLP	0,923895	67,0
glove+tfidf	LinearSVC	0,839742	64,0
	LogisticRegression	0,836995	8,1
	SLP	0,853434	31,0
use	LinearSVC	0,874475	4,9
	LogisticRegression	0,866728	13,9
	SLP	0,876990	31,4
bert_multi_cased	LinearSVC	0,833720	165,0
	LogisticRegression	0,778309	17,5
	SLP	0,789664	34,8
none	Bert_base	0,926333	18355,0
none	Gpt-2	0,923658	23983,0

Таблица 2 – Влияние векторизаторов на точность классификации

Векоризатор	**Среднее значение точности**
Word2Vec + TF-IDF	0.886063
Doc2Vec	0.329258
FastText	0.876840
TF-IDF	0.919436
GloVe + TF-IDF	0.843390
USE	0.872731
Bert	0.800564

Лучшие показатели точности были достигнуты на Bert_base модели - 92.6333 %. Самое лучшее соотношение скорости обучения к точности были получены на паре TF-IDF + LinearSVC.

Также была выведены среднее значение точности для каждого векторизатора (таблица 2). Данное значение бралось как среднее арифметическое всех показателей. Наилучшие результаты классификации были получены у методов, в которых применялся TF-IDF векторизатор.

Выводы по главе 3

Приведем выводы по второй главе бакалаврской работы:

− на языке программирования python разработан программный модуль, выполняющий загрузку, предобработку, классификацию текстовых данных и сравнение точности работы классификаторов.

− работа программного модуля протестирована на задаче автоматического определения тематики статей из новостных агрегарторов Google news и Yandex новости.

Заключение

При выполнении бакалаврской работы были получены следующие результаты:

– в ходе анализа литературных источников установлено, что процесс бизнес-аналитики направлен на сбор, анализ и визуализацию данных о деятельности компании с целью принятия оптимальных управленческих решений;

– установлено, что классификация текста выполняется на втором этапе процесса бизнес-аналитики и может быть использована, например, для разметки отзывов и комментариев при анализе лояльности клиентов к компании;

– выявлено, что актуальной проблемой в области бизнес-аналитики является разработка и совершенствование программного обеспечения, способного автоматизировано выполнять процесс классификации текста с использованием современных подходов;

– проведено моделирование процесса классификации текста, в ходе которого построены схема процесса обучения текстового классификатора, схема процесса классификации нового документа, диаграмма классов текстового классификатора, диаграмма последовательности процесса взаимодействия текстового классификатора с аналитической информационной системой;

– установлено, что при классификации текста необходимо использовать сочетание методов двух разных видов: метода векторизации, который преобразует исходный текст документа в числовой вектор и метод классификации, который на основе числового вектора определяет класс текстового документа;

– проведен анализ особенностей работы таких методов векторизации как TF-IDF, Word2Vec, Doc2Vec, FastText, GloVe, Universal-

Sentence-Encoder, Bert, а также особенности работы таких методов классификации как LogisticRegression, SVM, Single-layer perceptron (SLP), Bert и GPT-2;

− установлено, что заранее не известно, какое сочетание методов покажет наилучшие результаты при классификации текста, поэтому разрабатываемое программное обеспечение направлено на сравнительное тестирование различных методов. При этом решаемая задача классификации – автоматизированное определение тематики текстовых документов (новостных статей).

− на языке программирования python разработан программный модуль, выполняющий загрузку, предобработку, классификацию текстовых данных и сравнение точности работы классификаторов.

− работа программного модуля протестирована на задаче автоматического определения тематики статей из новостных агрегарторов Google news и Yandex новости.

Таким образом, цель бакалаврской работы достигнута.

Список используемой литературы и используемых источников

1. Агеев М. С. Автоматическая рубрикация текстов: методы и проблемы / М.С. Агеев, Б.В. Доброе, Н.В. Лукашевич // Ученые записки казанского государственного университета, 2008. – №4. – с. 25-41

2. Айвазян, С. А. Прикладная статистика: классификация и снижение размерности / Айвазян С. А., Бухштабер В. М., Енюков И. С., Мешалкин Л. Д. – М.: Финансы и статистика, 2009.

3. Корелов, С.В. Предобработка текстов электронных писем в задаче обнаружения спама / С.В. Корелов, А.М. Петров, Л.Ю. Ротков, А.А. Горбунов // Труды учебных заведений связи, 2020. – №4. – с. 80-91

4. Леоненков А. В. Объектно-ориентированный анализ и проектирование с использованием UML и IBM Rational Rose [Электронный ресурс] : учебное пособие. М. : Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2020. 317 с. [Электронный ресурс]. URL: https://www.iprbookshop.ru/97554.html (дата обращения: 06.09.2021).

5. Маннинг, К.Д. Введение в информационный поиск / Г Кристофер Д. Маннинг, Прабхакар Рагхаван, Хайнрих Шютце. : Пер. с англ. - М. : ООО “И.Д. Вильямс”, 2014 - 528с.

6. Мкртычев С.В., Гущина О.М., Очеповский А.В. Прикладная информатика. Бакалаврская работа [Электронный ресурс] : электрон. учеб-метод. пособие. Тольятти. ТГУ: Изд-во ТГУ, 2019. 1 оптический диск.

7. Bird, S. Natural Language Processing with Python / Steven Bird, Ewan Klein, Edward Loper. – Published by O’Reilly Media, Inc., 2009. – 502p.

8. Amasaki, S. The Effects of Vectorization Methods on Non-Functional Requirements Classification / Sousuke Amasaki, Pattara Leelaprute // 2018 44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), 2018. – IEEE, Prague, Czech Republic, 2018. – pp.55-78.

9. Bugueno, M. Learning to combine classifiers outputs with the

transformer for text classification / Margarita Bugueno, Marcelo Mendoza // Intelligent Data Analysis, 2020 – № 24. – pp. 15-41

10. Business Process Model and Notation [Электронный ресурс]. URL: https://www.omg.org/spec/BPMN/2.0/About-BPMN/ (дата обращения: 22.08.2021).

11. Gao, G. Research on Routing Selection Algorithm Based on Genetic Algorithm / Guohong Gao, Baojian Zhang, Xueyong Li, Jinna Lv // International Conference on Intelligent Computing and Information Science – International Conference, ICICIS 2011, Chongqing, China, January 8-9, 2011. Proceedings, Part II: Intelligent Computing and Information Science. – Springer-Verlag Berlin Heidelberg 2011. – pp. 353-358

12. Jurafsky, D. Speech and Languge Processing: An Introduction to Natural Language Proccerring, Computation Linguistics and Speech Recognition / Jurafsky, Daniel; H. James, Martin. – Stanford University, 2021. – 613 p.

13. Kowsari, K. Text Classification Algorithms: A Survey / Kamran Kowsari, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura Barnes, Donald Brown // Machine Learning on Scientific Data and Information. – Cornell University, 2019. – pp. 100-168.

14. Lee, Z. A Hybrid Search Algorithm of Ant Colony Optimization and Genetic Algorithm Applied to Weapon-Target Assignment Problems / Zne-Jung Lee, Wen-Li Lee // International Conference on Intelligent Data Engineering and Automated Learning – 4th International Conference, IDEAL 2003, Hong Kong, China, March 21-23, 2003. Revised Papers: Intelligent Data Engineering and Automated Learning. – Springer-Verlag Berlin Heidelberg 2003. – pp. 278-285

15. Smith, M. Using Genetic Programming for Feature Creation with a Genetic Algorithm Feature Selector / Matthew G. Smith, Larry Bull // International Conference on Parallel Problem Solving from Nature – 8th International Conference, Birmingham, UK, September 18-22, 2004. Proceedings: Parallel Problem Solving from Nature - PPSN VIII. – Springer-Verlag Berlin

Heidelberg 2004. – pp. 1163-1171

16. Srividhya, V. Evaluating Preprocessing Techniques in Text Categorization / V. Srividhya, R. Anitha // International Journal of Computer Science and Application Issue 2010. – pp. 49-51.

17. Sun, C. How to Fine-Tune BERT for Text Classification? / Chi Sun, Xipeng Qiu, Yige Xu, Xuanjing Huang // Computation and Language, 2020. – Cornell University, 2020. – pp. 23-45.

18. Tian, B. A Feature Selection Algorithm for Big Data Based on Genetic Algorithm / Bo Tian, Weizhi Xiong // International Conference on Mechatronics and Intelligent Robotics – Proceedings of the International Conference on Mechatronics and Intelligent Robotics (ICMIR2017) - Volume 1: Recent Developments in Mechatronics and Intelligent Robotics. – Springer International Publishing AG 2018. – pp. 159-163

19. Zhang, Ch. An Effective Feature Selection Scheme via Genetic Algorithm Using Mutual Information / Chunkai Zhang, Hong Hu // International Conference on Fuzzy Systems and Knowledge Discovery – Second International Conference, FSKD 2005, Changsha, China, August 27-29, 2005, Proceedings, Part II : Fuzzy Systems and Knowledge Discovery. – Springer-Verlag Berlin Heidelberg 2005. – pp. E1-E1

20. Zhang, M. Using Back Propagation Algorithm and Genetic Algorithm to Train and Refine Neural Networks for Object Detection / Mengjie Zhang, Victor Ciesielski // International Conference on Database and Expert Systems Applications – DEXA 1999: Database and Expert Systems Applications. – Springer-Verlag Berlin Heidelberg 1999. – pp. 626-635