Пермский филиал федерального государственного автономного
образовательного учреждения высшего образования
«Национальный исследовательский университет
«Высшая школа экономики»

Факультет экономики, менеджмента и бизнес-информатики

Джемадинов Эскендер Рустемович

АВТОМАТИЧЕСКОЕ ВЫЯВЛЕНИЕ "СМЫСЛОВЫХ ЭЛЕМЕНТОВ" ТЕКСТОВ, ВЛИЯЮЩИХ НА ИХ ЦЕЛЕВУЮ ХАРАКТЕРИСТИКУ

Выпускная квалификационная работа

по направлению подготовки 38.04.05 «Бизнес-информатика»

образовательная программа

«Информационная аналитика в управлении предприятием»

Рецензент
К.т.н., доцент департамента
анализа данных и
искусственного интеллекта
НИУ ВШЭ
Д.А. Ильвовский

Руководитель
Д.т.н., профессор,
профессор кафедры
информационных
технологий в бизнесе
НИУ ВШЭ-Пермь

_____
Л. Н. Ясницкий

Пермь, 2019 год

Аннотация

Автор: Джемадинов Эскендер Рустемович, студент 2 курса магистратуры группы ИАУП 17–1 НИУ ВШЭ г. Пермь. Тема магистерской диссертации – автоматическое выявление «смысловых элементов» тестов, влияющих на их целевую характеристику. В данной работе предлагается подход к выявления таких элементов в текстах программным путём с последующим построением на их основе прогностических моделей машинного обучения. Для анализа были взяты данные с соревнования Google по выявлению токсичности комментариев и построены модели машинного обучения. В работе показывается, что

Текущее исследование включает в себя 2 главы:

1. Теоретические аспекты изучаемой темы, описание основных моделях представления текстового контента и краткие теоретические сведения о используемых моделях машинного обучения.

2. Описание предлагаемой методологии и практическая реализация моделей машинного обучения на примере реальных данных, комментариев социальной сети с оценкой их точности работы.

Количество страниц – 42, количество изображений – 14, количество таблиц – 3.

Оглавление

Введение

В настоящее время анализ текстовых данных является одним из приоритетных направлений Data Mining. Выявление ключевых особенностей текстов, влияющих на некую целевую характеристику, позволяет оценить эффективность текстового материала.

В качестве примера рассмотрим благотворительный Фонд и набор текстовых материалов размещаемым этим фондом с целью привлечения финансирования. Тогда, целевой характеристикой такого текстового материала может являться сумма пожертвований, привлечённых фондом с помощью этого текстового материала.

Другим примером может служить социальная сеть, в которой пользователи пишут текстовые сообщения, называемые постами, с целью донесения своей точки зрения до других пользователей социальной сети. Целевой характеристикой поста в этом случае может являться эмоциональная реакция пользователя, выраженная одобрением этого поста, посредством нажатия кнопки “Мне нравится”.

Умение выделять особенности текстового материала, связанные с высокой или низкой целевой характеристикой, необходимо для оптимизации и улучшения текстового контента в рамках определенных задач компаний и бизнеса.

Данная проблема была частично рассмотрена в работах [1][2]. Кроме того эта проблема как правило рассматривается только для текстового материала на английском языке, в то время как тексты на русском языке могут иметь свою специфику. Такая специфика русского языка требует корректировки существующих или создания новых методик анализа текстовой информации. При этом, большинство существующих исследования в области автоматического анализа текстовой информации решают отличные задачи от вышеупомянутой.

Таким образом, целью работы является автоматическое выявление “смысловых элементов” текстов, влияющих на некоторую целевую характеристику, связанную с качеством этих текстов.

Данная цель обусловила реализацию следующих задач:

1. Анализ специфики области текстового анализа данных;
   1.1. Изучение литературы;
   1.2. Изучение существующих алгоритмов;

2. Разработка методики автоматического анализа смысловых элементов текстов;
3. Подбор корпуса текстов предметной области для проверки методики;
4. Достоверная оценка эффективности предлагаемой методики.

Объектом исследования являются тексты на естественном языке.

Предметом исследования являются методы автоматического анализа текстовой информации.

Используемыми методами исследования будут являться моделирование, анализ, синтез и абстрагирование.

Теоретическая значимость данного исследования заключается в теоретическом обосновании предлагаемого подхода, а его практическая значимость обуславливается возможность создания текстов с большей эффективностью с точки зрения некоторой целевой характеристики.

В данной работе имеется две главы, а также введение, заключение и список литературы. Первая глава, обзорная, содержит анализ предметной области, анализ технологий Text Mining, а также обзор используемых моделей машинного обучения. А вторая посвящена реализации моделей машинного обучения и оценке их эффективности.

Глава 1. Обзор технологий

В данной главе рассмотрены основные модели embeddings для слов и текстов, которые позволяют представить текстовую информацию в удобном для анализа виде. Также в данной главе рассматриваются методы машинного обучения, необходимые для работы с embeddings слов и текстов.

1.1. Предпосылки к разработке Word Embeddings

В русскоязычной среде пока что нет единого определения термина *«embedding»*. Дословно с английского языка это переводится как «включение», но обычно данных понятием определяют сопоставление определенной сущности (слова, вершины графа) некоторому вектору [3]. Соответственно, совокупность слов будет представлена совокупностью векторов в некотором пространстве, расположение которых определяется семантической близостью слов, их образующих. Однако, такому представлению предшествовали и другие, более простые способы структурирования информации [4].

Самым простым способом создания такого векторного пространства является проставление в каждом векторе единицы в позиции, соответствующей номеру слова в заранее сформированном словаре. Данный подход носит название *one-hot encoding (OHE).* Однако, полученные вектора не будут отражать семантическую близость между закодированными словами.

В таком случае для представления слов следует немного абстрагироваться от их формальных значений и определить устную речь или текст, как некоторое множество слов. Тогда представление текста будет эквивалентно сумме OHE-векторов, входящих в него слов. Таким образом, на выходе получится вектор подсчёта количества слов в тексте. Данная концепция организации и хранения данных называется “мешок слов” (*bag of words, BoW*). Основными недостатками этого подхода является потеря информации об упорядоченности и взаимном расположении элементов множества слов текста и проблема синонимичности некоторых токенов, т.е. неспособность отобразить синонимические связи между словами. Впрочем, несмотря на эти недостатки, тексты уже можно сравнивать на основании косинусного расстояния между полученными векторами.

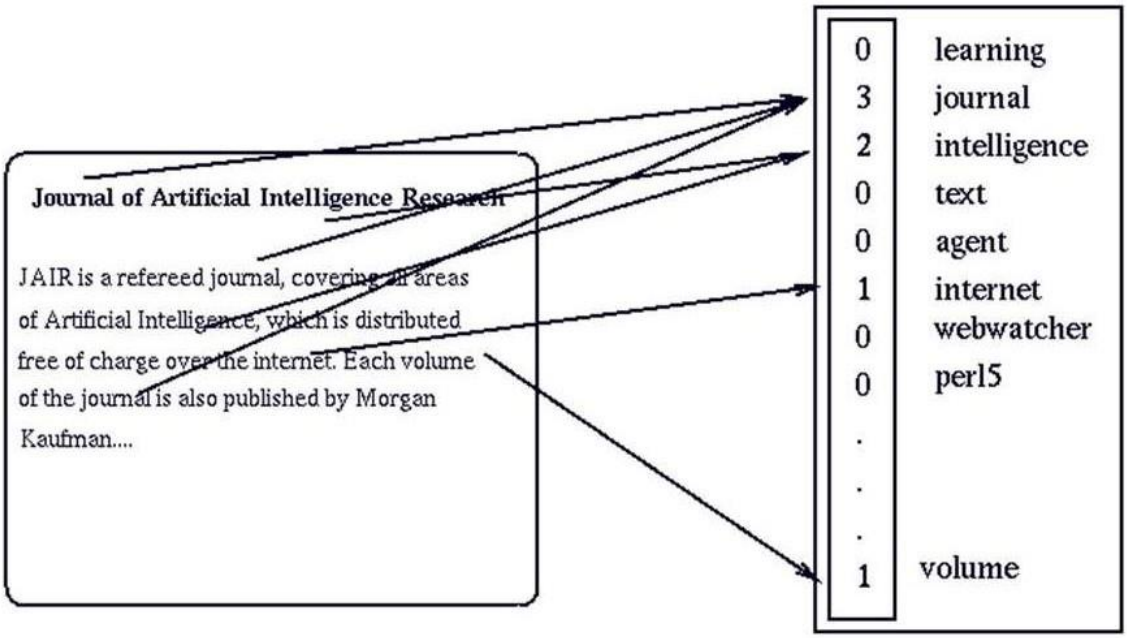

Рисунок 1.1 Модель embedding

Косинусное расстояние определяется следующим образом. Пусть заданы два вектора *x* и *y*. Тогда, скалярное произведение и косинус угла связаны между собой следующим соотношением:

$$\langle x, y\rangle = \|x\| \, \|y\| * cos\theta. \tag{1.1}$$

Следовательно, косинусное расстояние определяется как:

$$\rho_{\cos}(x, y) = \arccos\left(\frac{\langle x, y\rangle}{\|x\| \, \|y\|}\right) = \arccos\left(\frac{\sum_{i=1}^{d} x_i y_i}{(\sum_{i=1}^{d} x_i)^{\frac{1}{2}} (\sum_{i=1}^{d} y_i)^{\frac{1}{2}}}\right). \tag{1.2}$$

Такая косинусная мера используется для анализа схожести текста, определена некоторым интервалом и способна более корректно отображать отсутствие связи между векторами слов…. Каждый текст представлен в виде вектора, а каждая компонента является словом из словаря. Компонента представлена в бинарном виде и обозначается 1, если исходное слово встречается в тексте и 0 в противном случае. Тогда, чем больше слов встречается в двух текстах одновременно, тем больше косинусное расстояние между ними.

Term	Documents								
computer	c1	c2	c3	c4	c5	m1	m2	m3	m4
EPS	0	0	1	1	1	1	0	0	1
humana	1	1	0	0	0	1	0	1	1
interface	1	0	0	1	1	0	1	0	1
response	1	1	1	1	0	0	1	0	0
system	0	1	1	1	1	0	0	0	1
time	0	1	0	1	1	0	0	1	0
user	1	0	1	1	0	0	1	0	1
graph	0	1	1	1	0	0	0	0	0
minore	0	0	0	0	0	0	0	1	1
survey	0	0	0	1	1	1	0	0	1
trees	0	1	0	1	1	0	1	1	1

Рисунок 1.2. Матрица соответствия

Корпус текстов также может быть представлен в виде прямоугольной матрицы соответствия частоты встречаемых слов документам см. рис. 1.2.

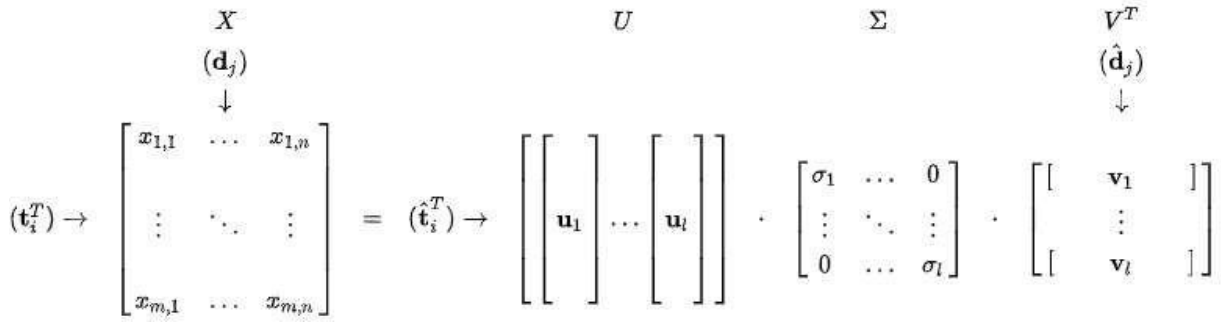

Рисунок 1.3. Матричное представление

Для последующего анализа эту матрицу “слово – документ можно разложить как произведения матриц “слово – тема” и “тема – документ”. В частности, это можно сделать с помощью сингулярного разложения матриц (SVD), см. рис. 1.3.Здесь t_i – слова, а d_i – документы.

Billy always listens to his mother
He always does what she says
if his mother says brush your teeth
Billy brushes his teeth
Billy is a very good boy a good boy listens to his mother
his mother does not have to ask him again

Рисунок 1.4. Корпус текста

Для начала инициализируется самый примитивный корпус текста см. рис 1.4.

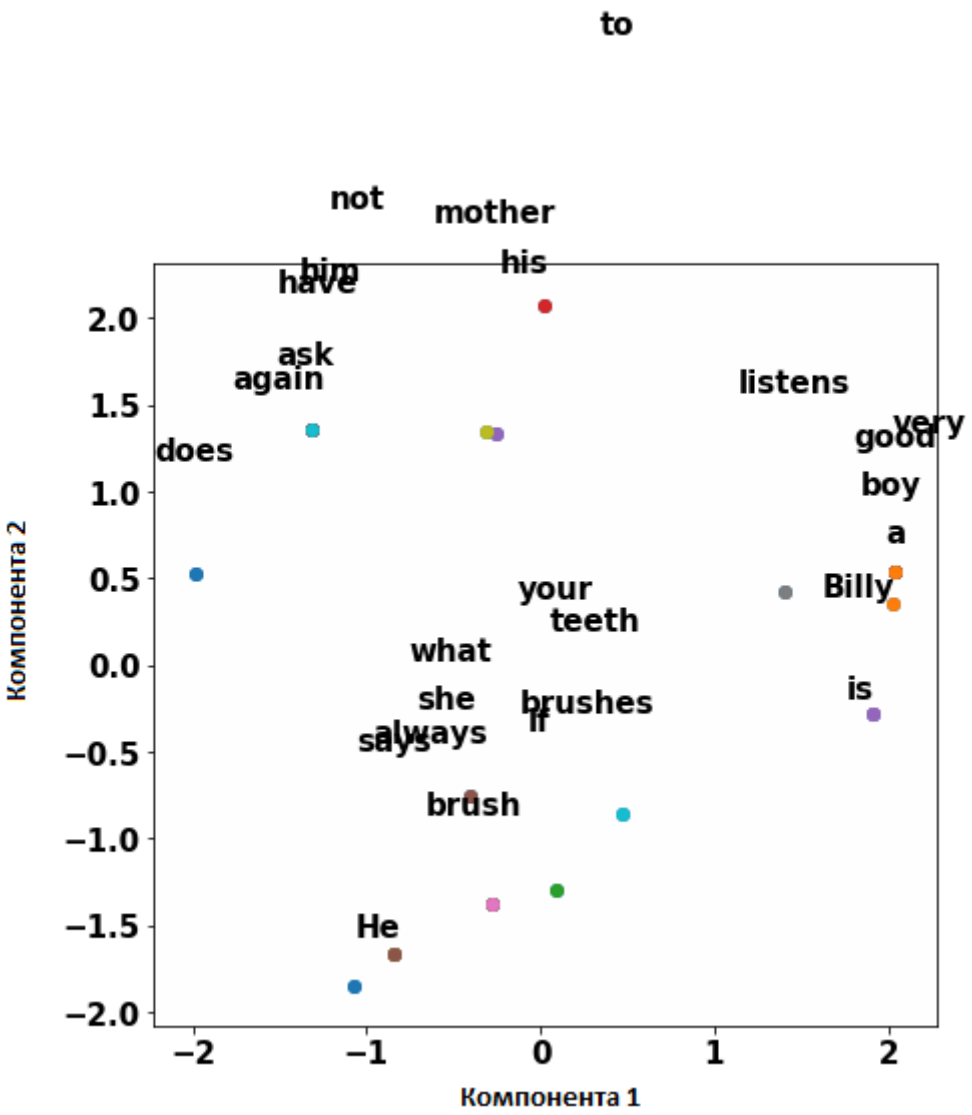

Рис. 1.5. Пространство слов

Далее преобразуем его в векторное пространство и декомпозируем с помощью SVD. Полученный результат представлен на рисунке 1.5.

Отсюда можно понять, что в целом смысловые единицы объединены между собой, но на достаточно примитивном уровне. Например, *“brush”* и *“teeth”*, *“Billy”* и *“boy”* находятся рядом, но в общем, пространство довольно сильно зашумлено. Кроме того, результат сильно зависит от корпуса текстов, с которым предстоит работать.

В дальнейшем матрица тема-документ была усовершенствована, что привело к появлению формулы TF-IDF. Эта аббревиатура означает *“term frequency — inverse document frequency”(частота слова – обратная частота документа)*. TF-IDF — статистическая мера, используемая для оценки важности слова в контексте

документа, являющегося частью корпуса текстов. Вес некоторого слова пропорционален частоте его употребления в документе и обратно пропорционален частоте употребления слова во всех документах коллекции. Формула для вычисления выглядит следующим образом:

$$TF - IDF(w, d, C) = \frac{count(w, d)}{count(d)} * \log(\frac{\sum_{d' \in C} 1(w, d')}{|C|}) \quad (1.3)$$

Здесь TF (*term frequency* — частота слова) — это отношение числа вхождений слова к общему числу слов в документе, а IDF (*inverse document frequency* — обратная частота документа) — это обратная величина к количеству текстов из располагаемого корпуса, в которых слово встречается хотя бы единожды. Учёт IDF снижает вес широкоупотребительных слов. Для каждого уникального слова в пределах конкретного корпуса текстов существует только одно значение IDF [5].

Метрика TF-IDF используется при анализе текстов и для информационного поиска. Например, TF-IDF может использоваться как один из критериев релевантности документа пользовательскому запросу, а также при расчёте степени близости документов в задачах кластеризации.

1.2. Word Embeddings: Word2vec

Описанные в предыдущей подглаве подходы являются эффективными для корпусов с относительно небольшим объёмом словарей и малым количеством текстов. Однако современное развитие технологий предполагает появление новых методов анализа текстов и оценки их семантической близости.

В 2013 году чешский аспирант Томаш Миколов предложил новый подход к word embedding, который и стал известен как word2vec [6]. В основе его подхода лежит так называемая «концепция локальности», а именно предположение о том, что слова, семантически близкие друг другу расположены в одинаковых окружениях.

Модель, предложенная Томашем Миколовым предполагает предсказание вероятности схожести слов по их контексту. Таким образом, обучаются такие вектора слов, чтобы вероятности, которые присваиваются моделью слову были близки к вероятности встречи этих слов в окружении в реальном тексте:

$$P(\omega_0|\omega_c) = \frac{e^{s(\omega_c,\omega_0)}}{\sum_{\omega_i \in V} e^{s(\omega_i,\omega_c)}}, \quad (1.4)$$

где ω_0 – вектор целевого слова, а ω_0 – вектор контекста, вычисленный усреднением контекстов вокруг данного слова Здесь. $s(\omega_c, \omega_0)$ – это функция, определяющая характеристическое число для двух векторов. Характеристическое число определяет близость слов в эвклидовым пространстве, и соответственно смысловую близость слов. Таким может являться, например, косинусное расстояние.

Вышеописанная формула носит название *softmax*, или иначе «мягкий максимум», где под термином «мягкий» понимается дифференцируемый. Это необходимо для обучения модели с помощью алгоритма обратного распространения ошибки.

Процесс обучения модели заключается в следующем: последовательно выбирается (2*k+1) слов, а слово в центре является тем словом, которое должно быть предсказано. Таким образом, окружающие слова являются контекстом длины k с каждой стороны. Для каждого слова существует отдельный вектор, который изменяется по мере обучения модели.

Такой подход носит название *CBOW – continuous bag of words*. Первое слово предполагает, что модели последовательно подаются на вход наборы слов текста, а BoW указывает на то, что порядок слов в контексте не играет роли.

Томашем Миколовым также был предложен другой подход, *skip-gram* – прямо противоположный *CBOW*, который предполагает угадывания контекста по заданному слову. В остальном модель не претерпевает изменения.

Стоит заметить, что, несмотря на то, что в модели не заложена семантика, натренированная модель *word2vec* может улавливать семантические свойства слов. Рассмотрим рисунке 1.6., где приведен пример из диссертации Томаша Миколова. Полученные отношения слов «мужчина» и «женщина», и «дядя» и «тётя» являются очевидными для естественного языка, но для других моделей добиться такого же соотношения векторов можно было только с помощью специальных ухищрений. В данном случае – это происходит автоматически на основании корпуса текстов. Кроме семантических связей, эта технология определяет и синтаксические связи, справа

отражено соотношение единственного и множественного числа, также полученного автоматически

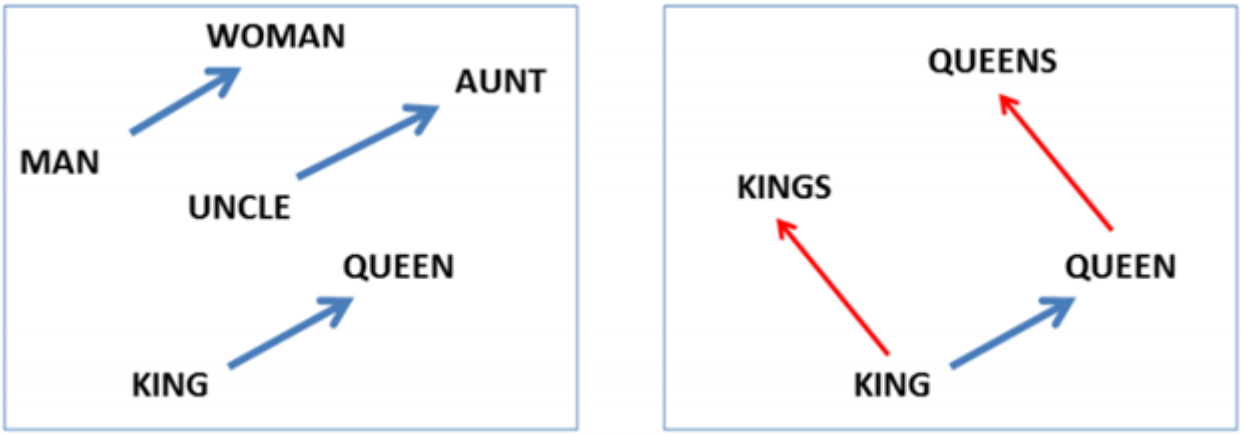

Рис. 1.6. Пространство слов embeddings

1.3. BERT модели и их сущность

Одной из самых больших проблем в обработке естественного языка (*Natural Language Process*) является нехватка данных для обучения. Поскольку *Natural Language Process* является многоотраслевой областью с множеством различных задач, большинство наборов данных для конкретных задач содержат только несколько тысяч или несколько сотен тысяч учебных примеров, помеченных человеком. Однако современные модели *Natural Language Process*, основанные на глубоком обучении, способны показывать эффективный результат на большой выборке данных, чего порой недостаточно в случае узкоспециального текстового анализа.

Чтобы помочь восполнить этот пробел, исследователи разработали различные методы для обучения моделей корпусами текстов общей лексики, используя огромное количество текста в Интернете. Такая предварительно обученная модель затем может быть точно настроена на задачи *Natural Language Process* с малыми данными, такие как ответы на вопросы и анализ настроений, что приводит к существенному повышению точности по сравнению с обучением этим набором данных с нуля. Недавно предложили новую *языковую модель*, именуемую «BERT» [17].

С этим выпуском любой человек в мире может обучить свою собственную современную систему ответа на вопросы (или множество других моделей) примерно за 30 минут на одном облачном GPU или за несколько часов с использованием одного графического процессора. Релиз включает в себя исходный код, созданный поверх TensorFlow, и несколько предварительно обученных моделей представления языка.

Идея основывается на недавней работе в области контекстного представления перед обучением, включая Semi-supervised Sequence Learning, Generative Pre-Training, ELMo, and ULMFit [17]. Однако, в отличие от предыдущих моделей BERT - это первое двунаправленное, языковое представление, предварительно обученное с использованием только простого текстового корпуса (в данном случае, Википедии).

Предварительно обученные представления могут быть контекстно-свободными или контекстными, и контекстные представления могут быть однонаправленными или двунаправленными. Модели без контекста, такие как *word2vec* или *GloVe* [3], генерируют представление (числовой вектор) каждого слова в словаре. Например, слово «банки» будет иметь одинаковое представление без контекста в «банковском счете» и «стеклянной тары». Вместо этого контекстные модели генерируют представление каждого слова на основе других слов в том же предложении. При этом однонаправленные модели порождают такое представление только на основании предыдущих слов в предложении. Например, в предложении «Я получил доступ к банковскому счету» однонаправленная контекстная модель будет представлять «банк» на основе «Я получил доступ к», но не «счет». Однако BERT представляет «банк», используя как предыдущий, так и следующий контекст одного предложение: «Я получил доступ к ... аккаунту». Такая модель называется двунаправленной.

Если двунаправленность настолько эффективна, почему это не было сделано раньше? Чтобы понять это, необходимо учитывать, что однонаправленные модели эффективно обучаются путем прогнозирования каждого слова, обусловленного предыдущими словами в предложении. Тем не менее, невозможно обучить двунаправленные модели, просто обуславливая каждое слово своими предыдущими и следующими словами, поскольку это позволило бы предсказанному слову косвенно «увидеть себя» в многослойной модели. Чтобы решить эту проблему, создатели

BERT использовали простую технику маскирования некоторых слов во входных данных, а затем вводим каждое слово в двух направлениях, чтобы предсказать замаскированные слова.

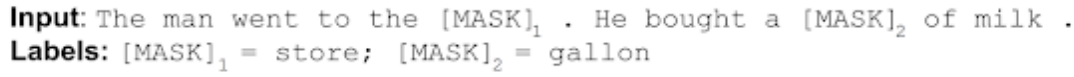

Рис. 1.7. Пример замены слов

На рисунке 1.7. показан пример маскировки.

Хотя эта идея существует уже очень давно, BERT впервые используется для предварительной подготовки глубокой нейронной сети. BERT также учится моделировать отношения между предложениями, выполняя предварительную подготовку по очень простой задаче, которая может быть сгенерирована из любого текстового корпуса: учитывая два предложения A и B, является ли B фактическим следующим предложением, которое следует после A в корпусе, или просто случайное предложение?

Рис. 1.8. Пример сопоставления предложений

На рисунке 1.8. показан пример сопоставления предложений.

Таким образом, BERT представляется эффективным способом анализа текстовой информации и должен быть добавлен к сравнению.

1.4. Используемые методы машинного обучения

Для анализа эффективности алгоритма необходимо построение прогностических моделей, в данной главе будет приведено теоретическое обоснование и анализ трёх самых распространенных методов машинного обучение.

1.4.1. Логистическая регрессия

Логистическая регрессия является видом множественной регрессии, суть которой заключается в анализе взаимосвязи между несколькими предикторами (независимыми переменными) и зависимой переменной (целевой).

Бинарная логическая регрессия, применяется только в случае, когда количество классов равняется двум, т.е. зависимая переменная может принимать

только два значения(например, возврат взятого кредита/дефолт, наличие/отсутствие болезни).

Все регрессионные модели могут быть записаны в виде следующей функции:

$$y = F(x_1, x_2, \dots, x_n) \quad (1.5)$$

В случае множественной линейной регрессии, например, предполагается, что выходная переменная является функцией от независимых переменных, т.е. (формула 6):

$$y = a + b_1 x_1 + b_2 x_2 + \cdots + b_n x_n \quad (1.6)$$

Такую формулу можно использовать для подсчёта вероятности события, предварительно вычислив параметрические коэффициенты. Например, если мы рассмотрим задачу из банковского сектора – предсказание возвращения займа, со значениями 0 и 1, где ноль будет означать дефолт, а единица – возвращение заёмщиком кредита. Впрочем, здесь существуем проблема. Множественной регрессии не известно, что выходная переменная бинарна. Это, в свою очередь приводит к тому, что результат предсказания может выходить за пределы диапазона [0;1], что недопустимо по условиям данной задачи. Таким образом, для множественной регрессии не существует ограничений для диапазона значений целевой переменной.

В данном случае задача регрессии может быть переформулирована следующим образом: вместо бинарного предсказания, предсказывается бинарная переменная со значением в интервале [0,1] в независимости от значения предикторов. Это осуществляется благодаря логит-преобразованию, регрессионному уравнению, приведенному ниже:

$$p = \frac{1}{1 - e^{-y}} \quad (1.7)$$

где p обозначается вероятность того, что требуемой событие наступит; e – основание натурального логарифма, равное 2,71; y –уравнение регрессии.

Данная зависимость вероятности от переменной y отражена на графике ниже (рис. 1.7.):

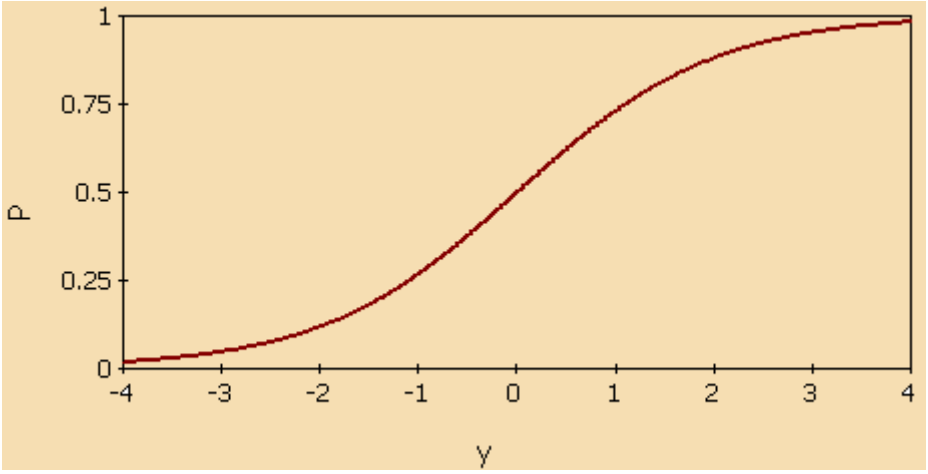

Рисунок 1.7. Логистическая кривая [8]

Необходимо акцентировать внимание на этом преобразовании. Допустим, что речь идёт о нашей зависимой переменной в терминах основной вероятности P, лежащей в интервале [0;1]. Тогда выраженная вероятность P будет выглядеть следующим образом :

$$P = log_e(\frac{P}{1-P}) \quad (1.8)$$

Данное преобразование называется логит-преобразованием или логистическим.

В теории P’ может равняться любому значению. Данное преобразование решает проблему ограничении в интервале [0,1] для зависимой переменной (вероятности), после чего преобразованные значения можно использовать в линейном регрессионном уравнении. Если произвести логистическое преобразование двух частей вышеописанного уравнения, то получится модель линейной регрессии.

Для нахождения коэффициентов линейной регрессии можно воспользоваться несколькими способами. Часто используют метод максимального правдоподобия. Он применяется в математической статистике для получения оценок параметров генеральной совокупности по выборке. В нём используется функция правдоподобия, выражающая плотность вероятности совместного появления результатов выборок Y_1, $Y_2, \dots, Y_k$:

$$L(Y1, Y2, \dots, Yk;\ \theta) = p(Y1; \theta) \cdot \dots \cdot p(Yk;\ \theta) \quad (1.9)$$

Согласно данному методу в качестве оценки неизвестного параметра принимается некоторое значение Θ=Θ($Y1,\dots,Yk$), которое максимизирует функцию L.

Можно значительно упростить вышеописанную задачу, если максимизировать не функцию L, а непосредственно натуральный логарифм *ln(L)*, поскольку функции достигают максимума при одном и том же значении θ:

$$L(Y;\theta) = ln(L(Y;\theta)) \rightarrow \max \tag{1.10}$$

В случае, если бинарная переменная логистической регрессии независима, выкладки можно продолжить следующим образом.

Будем полагать, что P_i вероятность появления единицы: P_i=Prob(Y_i=1). Эта вероятность зависит от $\mathbf{X_i W}$, где $\mathbf{X_i}$ – строка матрицы регрессоров, **W** – вектор коэффициентов :

$$P_i = F(X_i W), F(z) = log_e(\frac{P}{1-P}). \tag{1.11}$$

Логарифмическая функция правдоподобия в данном случае равна (формула 12):

$$L^* = \sum_{i \in I1} \ln P_i(W) + \sum_{i \in I0} \ln\big(1 - P_i(W)\big) = \sum_{i=1}^{k} \big[Y_i \ln P_i(W) - (1 - Y_i) \ln\big(1 - P_i(W)\big)\big] \tag{1.12}$$

где I0, I1 – множества наблюдений, для которых $Y_i = 0$ и $Y_i = 1$.

Тогда градиент **g** и гессиан H функции правдоподобия равны (формулы 1.13–1.14):

$$g = \sum_i (Y_i - P_i) X_i \tag{1.13}$$

$$H = -\sum_i P_i(1 - P_i) X_i^T X_i \leq 0 \tag{1.14}$$

Поскольку Гессиан в любом случае отрицательный, поэтому логарифмическая функция правдоподобия всегда вогнута. Для нахождения максимального значения необходимо использовать метод Ньютона, поскольку он всегда сходится (выполнено условие сходимости метода) :

$$W_{t+1} = W_t - \left(H(W_t)\right)^{-1} g_t(W_t) = W_t - \Delta W_t \quad (1.15)$$

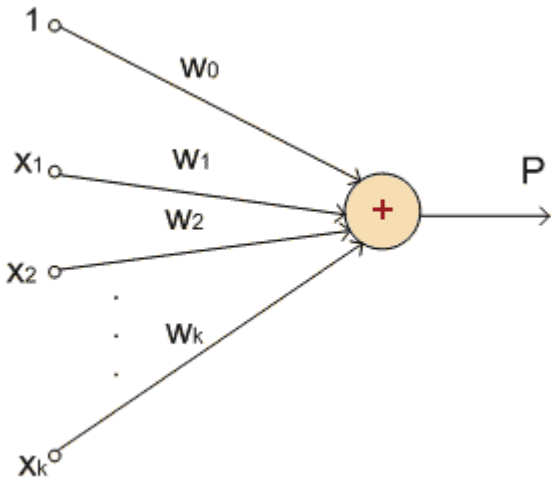

Рисунок 1.8. Представление логистической регрессии в виде нейронной сети[8]

Логистическую регрессию также можно представить как однослойную нейронную сеть, в которой в качестве функции активации используется сигмоида. Её веса это коэффициенты логистической регрессии, а вес поляризации – константа данного регрессионного уравнения (См. рис.1.8).

Однослойная нейронная сеть способна успешно решать только задачу линейной сепарации. Таким образом, возможности по моделированию нелинейных зависимостей у логистической регрессии отсутствуют. Однако для оценки качества модели логистической регрессии существует эффективный инструмент ROC-анализа [8].

Для расчета коэффициентов логистической регрессии применяют любые градиентные методы.

1.4.2. Деревья решений

Деревья решений – это совокупность правил, представленная в последовательной иерархической структуре, где каждому объекту поставлен в соответствие узел, дающий решение.

Под понятием “правило” понимается логическое следствие вида "если ... то ...".

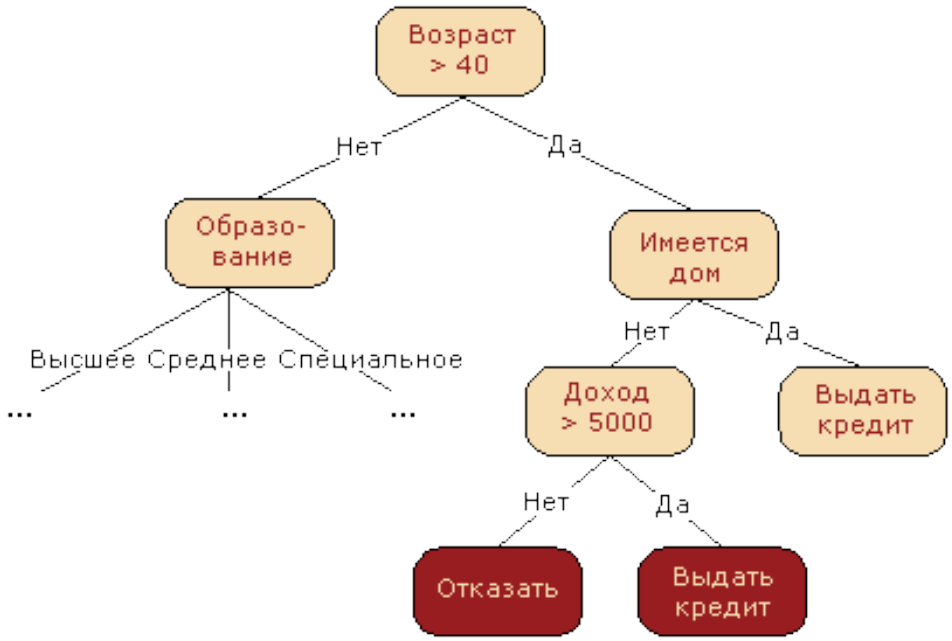

Рисунок 1.9. Деревья решений [9]

Сфера использования деревьев решений довольно обширна, но все задачи решаемые с их помощью могут быть сгруппированы в три класса:

1. Описание данных: Деревья решений позволяют хранить данные в компактной форме.
2. Регрессия: Если выходная переменная имеет непрерывное значение, то деревья решений позволяют предсказать значение целевой переменной, основываясь на зависимых. Сюда можно отнести, например, задачи численного прогнозирования.
3. Классификация: Деревья решений способны эффективно решать задачу классификации, т.е. сопоставления объектов с одним из заранее известных классов.

Алгоритм построения деревьев решений следующий. Пусть существует некоторое обучающее множество T, элементы которого характеризуется m атрибутами, причем один из них позволяет однозначно определить принадлежность объекта к какому-либо классу.

Идею построения деревьев решений из множества T, предложенную Хантом, рассмотрим в соответствии с трактовкой Р. Куинлену [10].

Пусть существует некоторое множество классов, обозначенных $\{C_1, C_2, \ldots C_k\}$, тогда возможно существование 3 ситуации:

1. Множество T содержит один или несколько примеров, которые можно отнести к классу C_k. Тогда деревом решений для данного множества T будет лист, определяющий класс C_k;
2. Множество T является пустым множеством. Тогда это снова лист, и класс, ассоциированный с листом, выбирается из другого множества отличного от T, например, из множества, ассоциированного с родителем;
3. Множество T содержит примеры, принадлежащие различным классам. В данном случае множество T разбивается на некоторые подмножества. Для этого выбирается признак, который имеет по меньшей мере два отличных друг от друга значений O_1, O_2, ... O_n. Множество T разбивается на подмножества T_1, T_2, ... T_n. Каждое такое подмножество T_i содержит все примеры, содержащее значение O_i для выбранного признака. Это будет продолжаться рекурсивно, пока конечное множество не будет состоять из объектов одного класса.

Данный подход является основой многих алгоритмов построения деревьев решений и известен под названием “разделяй и властвуй”. При использовании данной методики дерево решений будет строиться сверху вниз.

При построении деревьев необходимо акцентировать внимание на следующих вопросах: выбору критерия атрибута, по которому пойдет разбиение, остановки обучения и отсечения ветвей. Рассмотрим все эти вопросы по порядку.

Для построения дерева необходимо найти такое условие, которое бы разбивало существующее множество, ассоциированное с этим узлом на подмножества. В качестве проверки выбирается один из атрибутов. Общая идея выбора атрибута заключается в следующем: выбранный атрибут должен разбить множество таким образом, чтобы подмножества состояли из элементов, принадлежащих одному классу, или чтобы это разбиение было максимально приближенно к такому, т.е. количество объектов из другого класса было минимальным.

К основному алгоритму можно также добавить несколько правил:

1. Использование статистических методов, для оценки необходимости ранней остановки алгоритма. При этом может снизиться точность

предсказательной модели, поэтому иногда целесообразно использовать отсечение.

2. Ограничение глубины дерева. Если разбиение приводит тому, что текущее значение глубины дерева превышает максимальное, то необходимо остановить построение.
3. Разбиение должно быть нетривиальным, т.е. получившиеся узлы должны содержать не менее заданного количества примеров.

Этот список эвристических правил не окончательный может быть продолжен, но на сегодняшний день это наиболее эффективные из них. К этому стоит подходит с осторожностью, поскольку многие правила могут распространяться только на какие-то частные случаи.

Часто алгоритмы построения деревьев, строят сложные и переполненные узлами и данными ветвистые деревья, которые очень сложно интерпретировать. Кроме того, такое дерево разбивает данные на большое количество подмножеств, в каждом из которых находится сравнительно небольшое число объектов.

Ценность правила, основанного на двух или трёх объектах выборки крайне мала и для анализа данных оно непригодно. Более приемлимо строить деревья, состоящие из меньшего количества узлов, которым бы соответствовало подавляющее количество элементов из обучающей выборки.

В таком, можно предположить, что наиболее простым способом будет построение всех возможных вариантов деревьев, соответствующих обучающему множеству, и из них выбрать дерево с наименьшей глубиной. Однако, это задача является NP-полной, что было показано Л. Хайфилем (L. Hyafill) и Р. Ривестом (R. Rivest) [7], и, как известно, этот класс задач не имеет эффективных методов решения.

Для решения вышеописанной проблемы применяется так называемое отсечение ветвей (pruning).

Под точностью распознавания дерева решений будем понимать отношение верно классифицированных объектов при обучении к общему количеству объектов обучающей выборки, а под ошибкой – количество ошибочно классифицированных. Тогда допустим, что известен способ оценки ошибки дерева, ветвей и листьев. В таком случае корректно использование следующего алгоритма:

1. построить дерево;

2. отсечь или заменить поддеревом те ветви, которые не приведут к возрастанию ошибки.

Отсечение ветвей происходит снизу вверх, двигаясь с листьев дерева, отмечая узлы как листья, либо заменяя их поддеревом.
В большинстве практических задач отсечение показывает хорошие результаты, что позволяет говорить о корректности использования подобной методики [9].

В силу этих и многих других причин, методология деревьев решений является важным инструментом в работе каждого специалиста, занимающегося анализом данных.

1.4.3. Градиентный бустинг

Градиентный бустинг – метод машинного обучения, для задач регрессий и классификации, который основывается на комбинации моделей с низкой точностью предсказаний, ансамбль которых даёт результат выше, чем модели по отдельности. Обычно используются деревья решений.

Цель обучения с учителем – минимизация функции потерь. Рассмотрим математическую составляющую градиентного бустинга. В качестве функции потерь будет использована MSE – среднеквадратическая ошибка:

$$Loss = MSE = \sum(y_i - y_i^p)^2, \quad (1.19)$$

Соответственно, необходимо построить предсказательную модель таким образом, чтобы минимизировать функцию потерь. Для этого используется градиентный спуск, обновляющий предсказания с заданной скоростью обучения (параметр learning_rate) с целью поиска значений с минимальной функцией MSE:

$$Ly_i^p = y_i^p + \alpha \cdot \sum(y_i - y_i^p)^2 / \ \delta y_i^p, \quad (1.20)$$

Предсказания обновляются таким образом, что сумма квадратов ошибок стремиться к нулю и предсказанные значения в наибольшей степени соответствовали реальным.

Первое предположение линейной регрессии, что сумма отклонений равна 0, т.е. отклонения случайно распределены в окрестности нуля.

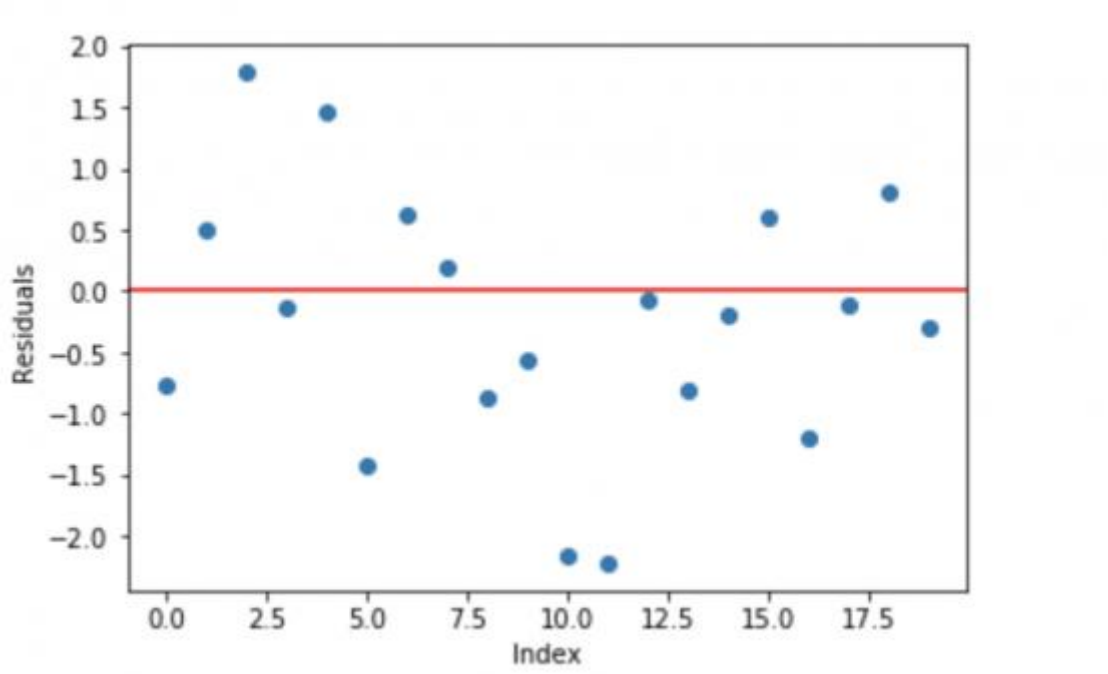

Рисунок 1.11 . Нормальное распределение выборки отклонений со средним 0

Итак, суть градиентного бустинга градиентного бустинга — **итеративное применение паттернов отклонений и улучшение предсказания**. Когда достигается момент отсутствия паттернов отклонений, построение модели прекращается (в противном случае возможно переобучение). Алгоритмически, осуществляется минимизация функции потерь.

Таким образом, модель градиентного бустинга основывается на следующих шагах [11]:

1. Построение примитивных моделей и анализ ошибок;
2. Выявление точек, не вписывающихся в модель;
3. Добавление моделей для обработки нестандартных случаев выявленных ранее;
4. Объединение моделей в ансамбль и определение веса каждого предсказателя.

1.4.4. Рекуррентные нейронные сети

Рекуррентные нейронные сети (RNN) представляют собой мощный и надежный тип нейронных сетей и относятся к наиболее перспективным алгоритмам, существующим на данный момент, поскольку они являются единственными с внутренней памятью.

RNN относительно старые, как и многие другие алгоритмы глубокого обучения. Первоначально они были созданы в 1980-х годах, но могут продемонстрировать свой реальный потенциал только через несколько лет из-за увеличения доступной вычислительной мощности.

Из-за своей внутренней памяти RNN могут запоминать важные вещи о полученном входных данных, что позволяет им очень точно предсказать, что будет дальше.

По этой причине они являются предпочтительным алгоритмом для последовательных данных, таких как временные ряды, речь, текст, финансовые данные, аудио, видео, погода и многое другое, потому что они могут сформировать более глубокое понимание последовательности и ее контекста по сравнению с другими алгоритмы.

Рекуррентные нейронные сети дают прогнозирующие результаты на последовательных данных, которые не могут дать другие подходы.

Под последовательными данными понимаются упорядоченные данные, где связанные вещи следуют друг за другом. Примерами являются финансовые данные или последовательность ДНК. Наиболее популярный тип последовательных данных - это, возможно, данные временных рядов, которые представляют собой просто ряд точек данных, перечисленных во временном порядке.

RNN и нейронные сети с прямой связью названы по способу передачи информации. В нейронной сети с прямой связью информация перемещается только в одном направлении, от входного слоя через скрытые слои к выходному слою. Информация движется прямо через сеть. Из-за этого информация никогда не касается узла дважды.

Нейронные сети с прямой связью не имеют памяти о входе, который они получили ранее, и поэтому плохо предсказывают, что будет дальше. Поскольку сеть с прямой связью учитывает только текущий вход, у нее нет понятия порядка во времени. Они просто не могут ничего вспомнить о том, что происходило в прошлом, кроме их тренировок.

В RNN информация циклически повторяется. Когда он принимает решение, он принимает во внимание текущий вклад, а также то, что он узнал из входов, которые он получил ранее.

Рекуррентная нейронная сеть имеет кратковременную память. В сочетании с LSTM она также имеют долговременную память.

Здесь уместно проиллюстрировать это примером. Представим, что существует нормальная нейронная сеть с прямой связью, и в качестве входных данных дайте ей слово «нейрон», и она обрабатывает слово символ за символом. Когда он достигает символа «р», он уже забыл о «н», «е» и «й», что делает практически невозможным для этого типа нейронной сети предсказать, какой объект последует дальше.

Рекуррентная нейронная сеть способна запомнить именно это благодаря своей внутренней памяти. Он производит вывод, копирует вывод и возвращает его обратно в сеть.

Рекуррентные нейронные сети добавляют непосредственное прошлое к настоящему.

Поэтому рекуррентная нейронная сеть имеет два входа: настоящее и недавнее прошлое. Это важно, потому что последовательность данных содержит важную информацию о том, что будет дальше, поэтому RNN может делать то, что другие алгоритмы не могут.

Нейронная сеть с прямой связью, как и все другие алгоритмы глубокого обучения, назначает матрицу весов своим входам и затем выдает выходные данные.

Кроме того, в то время как нейронные сети с прямой связью отображают один вход на один выход, RNN могут отображать один на многие, многие на многие (перевод) и многие на один (классификация голоса) [12].

Глава 2 . Разработка методики оценки числовой характеристики текста

В данной главе рассмотрены методики выявления ключевых смыслов текста, применимые к реальной задаче классификации, построение предсказательной модели и оценка точности результата.

2.1.Математическое описание подхода

Для оценки заинтересованности пользователя в текстовой информацией необходимо анализировать текст, выявляя некоторые «текстовые смыслы».

Для решения проблемы выявления смыслов предлагается подход, основанный на комбинации подходов TF-IDF и word-embeddings (*word2vec, Glove*). Под понятием embeddings здесь понимается сопоставление слов некоторому вектору в многомерном метрическом пространстве. Слова, встречающиеся в сходных контекстах, будут иметь близкие точки в этом пространстве [13]. Также применяется *TF-IDF* – классическая мера, используемая для оценки важности слова в контексте документа, являющегося частью корпуса документов [5].

Основная идея предлагаемого подхода состоит в обобщении метрики *TF-IDF* на случай схожих (с точки зрения расстояния в векторном пространстве *embeddings*) слов, которая помимо самого слова (и его лексических форм) учитывает также наличие в тексте других, близких к нему по значению слов[14].

Для обобщения метрики *TF-IDF*, в простейшем случае, можно ввести такой порог, что если расстояние между словами меньше этого порога, то они считаются неразличимыми, и тогда расчет индекса *TF-IDF* не сильно отличается от классического случая. Во втором, предлагается ввести функцию сходства $w(t,\tau)$ слов t и τ, возвращающая 1 для сходства слова t с самим собой и значение из диапазона [0; 1] для сходства слова t с произвольным словом τ. Примером может служить линейная функция:

$$w = 1 - \frac{1}{\Delta} min(dist(t,\tau), \Delta) \qquad (2.1)$$

где *dist* функция расстояния между словами в пространстве *embeddings*, а Δ – максимальное расстояние между схожими словами.

Тогда, формулы для вычисления взвешенных *TF* и *IDF* будут выглядеть следующим образом:

$$TF(t,d) = \frac{\sum_{\tau \in d} n(\tau,d) \cdot w(t,\tau)}{\sum_{\tau \in d} n(\tau,d)} \quad (2.2)$$

$$IDF(t,D) = log\left[\frac{|D|}{\sum_{d \in D} max_{\tau \in d} w(t,\tau)}\right] \quad (2.3)$$

где τ и t – некоторые слова, d – документ (мультимножество слов), D – корпус документов (множество документов), n – функция подсчёта количества слов в документе d.

Таким образом, данная модель может быть реализована в решении задач классификации и регрессии, суть которых будет рассмотрена далее.

2.2.Задача классификации

Задача классификации - это задача, в которой существует конечное множество объектов, разделенных на классы. Выборкой в задаче называется множество объектов, для которых определенно к какому именно классу они принадлежат. Принадлежность объектов другой совокупности к какому-либо классу неизвестная. Задача заключается в сопоставлении объектов неизвестной выборки к заранее известным классам [15]. Классификацией объекта называется то название класса, которое выдается алгоритмом классификации для этого объекта.

В области математической статистики, данная задача иногда называется задачей дискриминантного анализа. Такая задача решается различными методами машинного обучения, такими как нейронная сеть, логистическая регрессия и т.д..

Если же отсутствует множество размеченных объектов, то существует другой метод эксперимента – обучение без учителя, который обычно используется для решения задачи кластеризации. Существующее множество объектов необходимо разделить на определенные классы, опираясь только на сходство между ними. Иногда, в некоторых ситуациях часто отсутствует различие между проблемами кластеризации и задачами классификации [16].

2.3.Задача регрессии

Регрессионный анализ — метод моделирования данных и исследования их свойств. Существует два типа переменных: зависимая переменная (переменная отклика) и независимой переменной (объясняющей переменной). Регрессия задаёт функцию соответствия независимой переменной и параметров с добавленной случайной переменной. Параметры модели подбираются, чтобы наилучшим образом аппроксимировать данные и минимизировать ошибку. Критерием качества модели обычно выступает (СКО) – среднеквадратическая ошибка, которая определяется как сумма квадратов разностей значения аппроксимирующей функции и значения в точке [15].

Регрессия — зависимость математического ожидания случайной величины от одной или нескольких других случайных величин, то есть $E(y|x) = f(x)$. Регрессионным анализом называется поиск аппроксимирующей функции f, которая отражает эту зависимость. Регрессия может быть представлена в виде суммы неслучайной и случайной составляющих.

$$y = f(x) + v_1 \tag{2.4}$$

где *f* — функция, а *v* — свободный член.

Линейная регрессия определяет, что функция *f(x)* зависит от параметров ω линейно. При этом линейная зависимость от свободной переменной *x* необязательна.

$$y = f(\omega, x) + v = \sum_{j=1}^{N} \omega_j g_j(x) + v \tag{2.5}$$

Значения параметров находят с помощью МНК (метода наименьших квадратов).

Полученные разности $y_i - f(x_i)$ между фактическими значениями зависимой переменной и восстановленными называются **регрессионными остатками**. Одной из важных оценок критерия качества полученной зависимости является сумма квадратов остатков :

$$SSE = |f(x_i - y_i)| = \sum_{i=1}^{N} \left(y_i - f(\omega, x_i)\right)^2 \tag{2.6}$$

где *SSE* — Sum of Squared Errors.

Дисперсия остатков вычисляется по формуле (см. формулу 2.6)

$$\bar{\sigma}^2 = \frac{SSE}{N-2} = MSE \tag{2.7}$$

где MSE — Mean Square Error(среднеквадратичная ошибка).

2.4.Описание данных

В данной работе использовались данные благотворительного фонда “Дед Морозим” и Google.

Рассмотрим подробнее данные благотворительного фонда «Дедморозим». Данные представляют собой текстовые сообщения, с призывом пожертвовать некоторую сумму для той или иной кампании по сбору средств и соответствующие транзакции за период с 2012 по 2019 год (см. таблицы 2.1– 2.2).

Таблица 2.1. Финансовые данные

Дата	Данные отправителя	Сумма	Назначение платежа
01.11.2017	Х. АНДРЕЙ ИВАНОВИЧ	30	Благотворительный взнос
01.11.2017	Г. ЕВГЕНИЙ ЛЕОНИДОВИЧ	50	Благотворительный взнос
01.11.2017	К. НАТАЛЬЯ ВЛАДИМИРОВНА	50	Благотворительный взнос
01.11.2017	Е.ИРИНА АЛЕКСАНДРОВНА	50	Благотворительный взнос
01.11.2017	К. ЕЛЕНА МИХАЙЛОВНА	50	Благотворительный взнос
01.11.2017	Б. ИРИНА НИКОЛАЕВНА	88,24	Благотворительный взнос
01.11.2017	С. ИРИНА НИКОЛАЕВНА	100	Благотворительный взнос
01.11.2017	К. ЛЮБОВЬ ВЛАДИМИРОВНА	100	Благотворительный взнос
01.11.2017	К. МАРИНА ВЛАДИМИРОВНА	100	Благотворительный взнос
01.11.2017	Б. ОКСАНА ПАВЛОВНА	100	Благотворительный взнос100
01.11.2017	М. ЛЮБОВЬ ПЕТРОВНА	101	Благотворительный взнос

Таблица 2.2. Текстовые данные

	Date	**id**	**owner_id**	**text**
0	1542797705	12932	-13457319	Огненное шоу поможет Егору дышать В ближайшее воскресенье 25 ноября с 20:00 до 21:30 в [club55357848|саду Миндовского] состоится очередной «благотворительный покрутон» — [club21248950|выступление артистов огненного жанра] под открытым небом. Мероприятие бесплатное, но в течение всего вечера можно будет оставить посильную сумму в копилке детского здоровья #дедморозим. Все вырученные средства отправятся в поддержку Егора Ильина, на чудо для которого необходимо собрать 1 220 000 рублей. Узнать историю Егора можно по ссылке: https://dedmorozim.ru/campaign/egor-ilin/

				До встречи на «покрутоне»! Чудеса верят в вас!
1	1542978450	12949	-13457319	За неделю в Перми обнаружились 205 Дедов Морозов и Снегурочек Всего за неделю 205 человек выбрали на сайте «Дедморозим» детскую новогоднюю мечту себе по душе и зарезервировали письма, оставив комментарий. Еще 169 желаний ждут, когда же найдутся Деды Морозы и Снегурочки, которым по силам будет их исполнить. Детские письма будут добавляться на сайт #дедморозим в течение ближайших трёх недель, вплоть до общего сбора исполненных желаний 15-16 декабря. Напомним, вы можете осуществить мечту ребёнка из детского дома самостоятельно, выбрав письмо на сайте https://dedmorozim.ru/pisma/ и доставив подарок координаторам до середины декабря. А до самого Нового года, 31 декабря, можно помочь детским желаниям исполниться, оплатив их. Так осуществятся все без исключения всамделишные мечты детей, которые будут встречать Новый год без мам и пап. Даже из самых небольших пожертвований складывается возможность осуществить самую большую мечту ребёнка. Перечислите посильную сумму любым удобным способом: — прямо сейчас на сайте «Дедморозим»: https://dedmorozim.ru/pisma/ — отправив СМС на номер 7715 с текстом «Дедморозим» пробел «сумма пожертвования», например «Дедморозим 300» (сообщение без кавычек) — в «Сбербанк Онлайн»: «Переводы и платежи» – «Благотворительность и социальная помощь» – «Благотворительные фонды» – «Дедморозим» (недоступно на iOS) — в меню банкомата: «Платежи наличными» или «Платежи нашего региона» —«Прочие» — «Благотворительные платежи» — «Фонд Дедморозим» Чудеса верят в вас!
2	1542970955	12947	-13457319	Ещё 73 пермяка готовы поделиться лекарством от рака После донорской акции, прошедшей на выставке «Медицина и здоровье» на Пермской ярмарке, Национальный регистр доноров костного мозга увеличился на 73

				потенциальных донора. А это значит, что у пациентов с раком крови теперь больше шансов вылечиться! Примерно половина из всех вступивших на [club36501454|Пермской ярмарке] в ряды потенциальных доноров — студенты [club25483167|Медицинского университета]. Ребята оказались прекрасными потенциальными донорами, они не только хорошо знают, что такое костный мозг, но и понимают, насколько это ответственное и важное дело. Большое детское спасибо! Хотите, чтобы в вашем учебном заведении или компании нашлось лекарство от рака? Расскажите нам об этом! Если у вас на работе (учёбе) более пятидесяти желающих сдать кровь единомышленников, выездную донорскую акцию можно провести прямо у вас в офисе! Кстати, поделиться лекарством от рака можно в любой удобный день в ближайшей [club71775110|лаборатории «МедЛабЭкспресс»]. Вы уже в регистре потенциальных доноров? Поддержите проект #донорствоума посильной суммой или подключите регулярное пожертвование: https://dedmorozim.ru/projects/donorstvo-uma Лекарство от рака есть в каждом! Поделитесь им. #лекарствоотрака #дедморозим

Где *date*– дата поста, id– идентификатор поста, owner_id– владелец поста, text – текст поста.

Далее необходимо осуществить предварительную обработку данных. Транзакции необходимо агрегировать, а текст необходимо очистить от небуквенных символов, удалить ссылки, хештеги и осуществить лемматизацию слов, т.е. привести слова к начальной форме.

Таким образом, получается датасет, готовый к дальнейшему построению моделей машинного обучения.

В свою очередь, для задачи классификации используется набор данных, составленный командой AI Conversation, основанная Jigsaw и Google (обе части Alphabet) [18], которые разрабатывает технологию для защиты голоса в разговоре. Основное внимание уделяется моделям машинного обучения, которые могут

идентифицировать токсичность в онлайн-разговорах, где токсичность определяется как «что-либо грубое, неуважительное или иным образом способное заставить кого-либо покинуть дискуссию» [18].

Этот конкурс продолжением конкурса от 2017 года по классификации комментариев, где создавались модели с несколькими предикторами для распознавания общей токсичности сообщения, а также нескольких её подтипов. Например, таким подтипом являются, христиане, буддисты и т.д.… Этот датасет используется для создания моделей токсичности, которые оптимально работают в широком диапазоне коммуникативной деятельности.

В этом случае необходимо создать модель, которая распознает токсичность и сводит к минимуму непреднамеренный уклон этого типа в отношении упоминаний идентичностей.

Данные выглядят следующим образом (см. таблицу 2.3).

Таблица 2.3. Текстовые данные (комментарии)

id	Target	comment_text	severe_toxicity	obscene
59848	0.000000	This is so cool. It's like, 'would you want yo...	0.000000	0.000000
59849	0.000000	Thank you!! This would make my life a lot less...	0.000000	0.000000
59852	0.000000	This is such an urgent design problem; kudos t...	0.000000	0.000000
59855	0.000000	Is this something I'll be able to install on m...	0.000000	0.000000
59856	0.893617	haha you guys are a bunch of losers.	0.021277	0.000000
59859	0.66666	ur a sh*tty comment.	0.047619	0.638095
59861	0.457627	hahahahahahahahhha suck it.	0.050847	0.305085
59863	0.000000	FFFFUUUUUUUUUUUUUU	0.000000	0.000000
239575	0.000000	The ranchers seem motivated by mostly by greed...	0.000000	0.000000
239576	0.000000	It was a great show. Not a combo I'd of expect...	0.000000	0.000000
239578	0.000000	Wow, that sounds great.	0.000000	0.000000
239579	0.440000	This is a great story. Man. I wonder if the pe...	0.000000	0.293333
239582	0.000000	This seems like a step in the right direction.	0.000000	0.000000
239583	0.600000	It's ridiculous that these guys are being call...	0.000000	0.100000
239584	0.500000	This story gets more ridiculous by the hour! A...	0.000000	0.000000
239585	0.000000	I agree; I don't want to grant them the legiti...	0.000000	0.000000

239589	0.000000	Interesting. I'll be curious to see how this w...	0.000000	0.000000
239590	0.000000	Awesome! I love Civil Comments!	0.000000	0.000000
239591	0.000000	I'm glad you're working on this, and I look fo...	0.000000	0.000000
239592	0.500000	Angry trolls, misogynists and Racists", oh my....	0.000000	0.000000
239593	0.000000	Nice to some attempts to try to make comments ...	0.000000	0.000000
239594	0.000000	One would hope that the purpose of introducing...	0.000000	0.000000
239596	0.000000	Comments will be randomly chosen and be review...	0.000000	0.000000
239597	0.000000	She would be a major improvement for city coun...	0.000000	0.000000
239598	0.000000	I agree! Comments have so much potential to be...	0.000000	0.000000
239600	0.000000	Great question! It's one we're asked a lot. We...	0.000000	0.000000

В данных, предоставленных для исследования дан текст отдельного комментария пользователей некоторой социальной площадки. Он находится в столбце **comment_text**. Каждый комментарий в тренировочном наборе данных имеет метку токсичности (целевую переменную), и модели должны прогнозировать целевую токсичность для тестового набора данных. Этот атрибут (и все остальные) являются дробными значениями, которые представляют долю людей, которые считают, что атрибут применяется к данному комментарию. Для оценки будут рассмотрены примеры тестового набора с целью значение токсичности которых не меньше 0.5 в положительном классе (токсичный).

Данные также имеют несколько дополнительных атрибутов подтипа токсичности в зависимости от определенной социальной группы. Модели не должны прогнозировать эти атрибуты для конкуренции, они включены в качестве дополнительного пространства для исследований (**severe_toxicity** и **obscene**).

2.5.Оценка точности моделей машинного обучения

Для проверки предложенного подхода взята совокупность комментариев датасета Jigsaw[18]. Данные, представляют собой текстовые посты, а также. В качестве метрики точности используется ROC AUC.

Кривая ROC (кривая рабочих характеристик приемника) представляет собой график, показывающий эффективность модели классификации при всех порогах классификации[19]. Эта кривая отображает два параметра:

1. количества носителей признака, верно классифицированных как несущих признак.
2. долей объектов от общего количества объектов, не несущих признака, ошибочно классифицированных как несущих признак.

Задача классификации состоит в том, чтобы относить ранее неизвестные сущности к тому или иному классу. Примером такой задачи может быть постановка диагноза по медицинским анализам. В этом случае есть два класса результатов: положительный (positive) и отрицательный (negative). Тогда на выходе классификатора может наблюдаться четыре различных ситуации:

1. положительный результат классификации, и истинное значение тоже положительное, то это ситуации существования истинно-положительного значения (true-positive, TP)
2. положительный результат классификации, но истинное значение отрицательное, то это ситуации существования ложно-положительного значения (false-positive, FP)
3. отрицательный результат классификации, и истинное значение тоже отрицательное, то это ситуации существования истинно-отрицательного значения (true-negative, TN)
4. отрицательный результат классификации, но истинное значение положительно, то речь идет о ложно-отрицательном значении (false-negative, FN)

True Positive Rate (TPR) является синонимом отзыва и поэтому определяется следующим образом:

$$TPR = \frac{TP}{TP + FN} \tag{2.8}$$

False positive rate (FPR) определяется следующим образом:

$$FPR = \frac{FP}{FP + TN} \quad (2.9)$$

Кривая ROC отображает отношение TPR к FPR при различных пороговых значениях классификации. Понижение порога классификации классифицирует больше предметов как положительные, увеличивая как ложные, так и истинные положительные результаты.

Для тестирования моделей предлагается разделить датасет с данными в соотношении 80/20 (тренировочные/тестовые). Обучение будет проводиться на 80%, тестирование на 20%. Для лучшего прогноза предлагается оптимизировать параметры модели с помощью функции GridSearchCV библиотеки sk-learn.. Это поможет оптимально подобрать параметры гипермодели для достижения наилучших результатов. Эта функция решает задачу оптимизации гиперпараметров алгоритма машинного обучения, т.е. метапараметров самого алгоритма, таких как скорости схождения алгоритма, соотношение весов ошибок разного рода и т.д.. Полученный на выходе оптимальный набор параметров позволяет увеличить точность предсказания на независимом наборе данных, путём оптимизации целевой функции потерь.

Подбор гиперпараметров состоит из:

1. оценщика (регрессора или классификатора, например, sklearn.svm.SVC ());
2. пространства параметров;
3. алгоритма поиска и отбора;
4. кросс-валидации;
5. метрики качества.

Некоторые модели допускают специализированные, эффективные стратегии поиска параметров, описанные ниже. В scikit-learn представлены два общих подхода к выборке кандидатов для поиска: для заданных значений GridSearchCV рассматривает все комбинации параметров, в то время как RandomizedSearchCV может выбирать указанное число кандидатов из пространства параметров с заданным распределением.

Обычно небольшое подмножество этих параметров может оказать большое влияние на прогнозирующую или вычислительную производительность модели, в то время как для других можно оставить значения по умолчанию.

Поиск в сетке, предоставляемый GridSearchCV, полностью перебирает параметры из сетки значений параметров, указанных в параметре param_grid.

Например, следующий param_grid:

```
param_grid = [
  {'C': [1, 10, 100, 1000], 'kernel': ['linear']},
  {'C': [1, 10, 100, 1000], 'gamma': [0.001, 0.0001], 'kernel': ['rbf']},
 ]
```

указывает, что должны быть исследованы две сетки: одна с линейным ядром и значениями C в [1, 10, 100, 1000], а вторая - с ядром RBF, и перекрестным произведением значений C в диапазоне [1, 10 , 100, 1000] и значения гаммы в [0,001, 0,0001].

Экземпляр GridSearchCV реализует обычный API оценщика: при «подгонке» его к набору данных оцениваются все возможные комбинации значений параметров и сохраняется лучшая комбинация.

В то время как использование сетки настроек параметров в настоящее время является наиболее широко используемым методом оптимизации параметров, другие методы поиска имеют более благоприятные свойства. RandomizedSearchCV реализует рандомизированный поиск по параметрам, где каждый параметр выбирается из распределения по возможным значениям параметров. Это имеет два основных преимущества по сравнению с исчерпывающим поиском:

Бюджет может быть выбран независимо от количества параметров и возможных значений.

Добавление параметров, которые не влияют на производительность, не снижает эффективность.

Указание того, как следует выбирать параметры, выполняется с помощью словаря, очень похожего на указание параметров для GridSearchCV. Кроме того, с помощью параметра n_iter указывается вычислительный бюджет, представляющий собой число выбранных кандидатов или итераций выборки. Для каждого параметра можно указать либо распределение по возможным значениям, либо список дискретных вариантов (которые будут выбраны равномерно):

```
       {'C': scipy.stats.expon(scale=100), 'gamma':
scipy.stats.expon(scale=.1),
         'kernel': ['rbf'], 'class_weight':['balanced', None]}
```

В этом примере используется модуль scipy.stats, который содержит множество полезных распределений для параметров выборки, таких как expon, gamma ,iform или randint. В принципе, можно передать любую функцию, которая предоставляет метод rvs (случайная переменная выборка) для выборки значения. Вызов функции rvs должен предоставлять независимые случайные выборки из возможных значений параметров при последовательных вызовах.

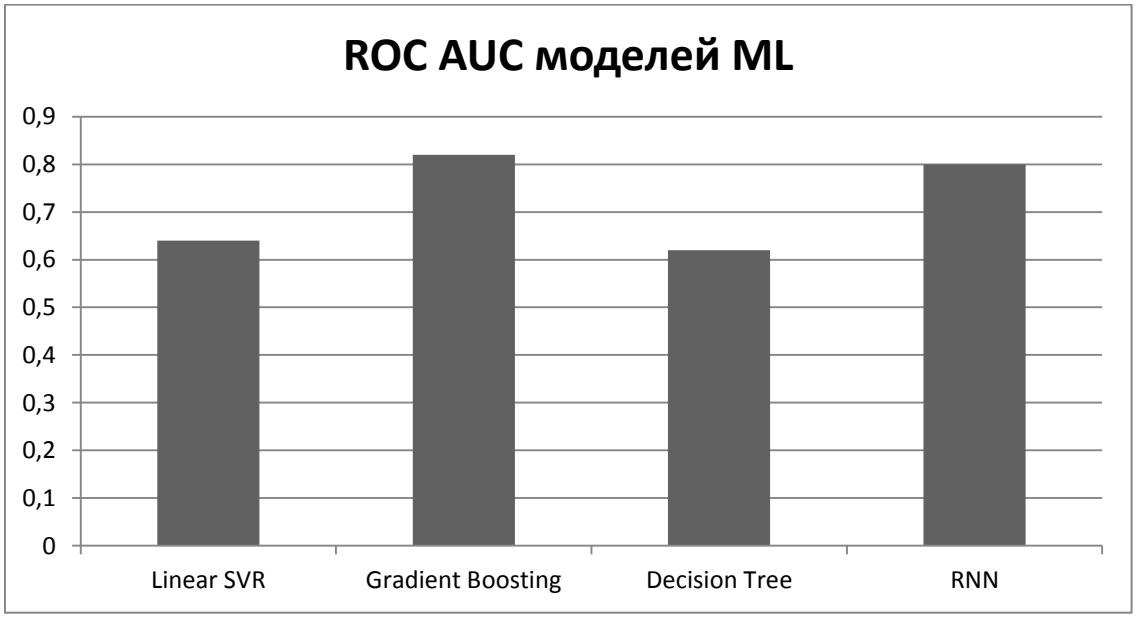

Рисунок 2.1. ROC AUC моделей (без оптимизации)

На рисунке 2.1. приведены результаты анализа для модели без оптимизации параметров.

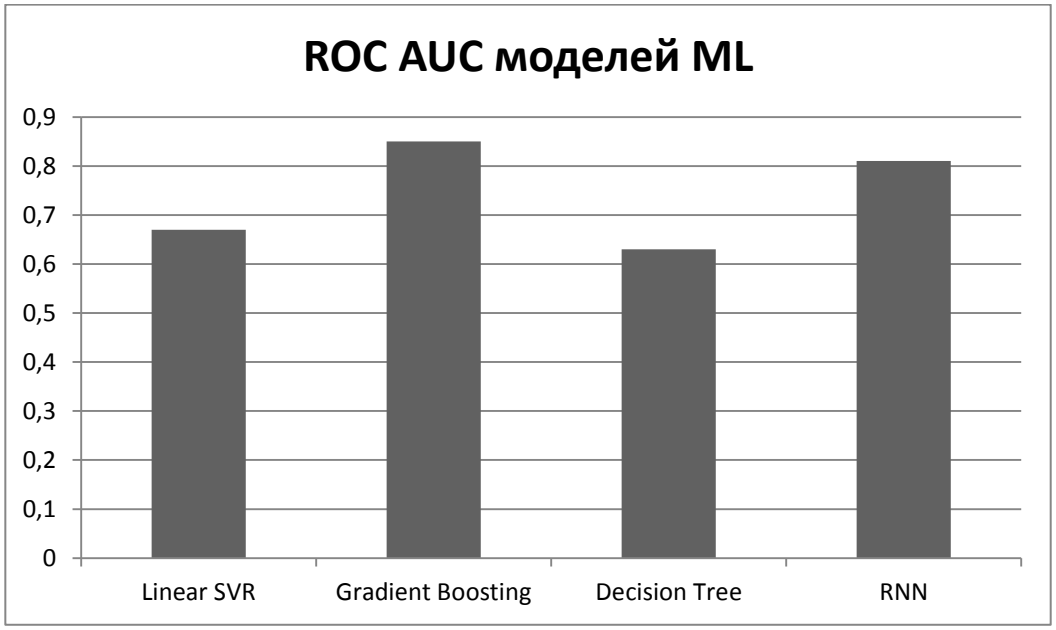

Рисунок 2.2. ROC AUC моделей (с оптимизацией)

Для дальнейшего анализа используется градиентный бустинг из программного пакета lightgbm, поскольку значение метрики в этом случае наиболее высокое. Например, на рисунке 2.2. показана для точность для моделей, основанных на *TF-IDF*.

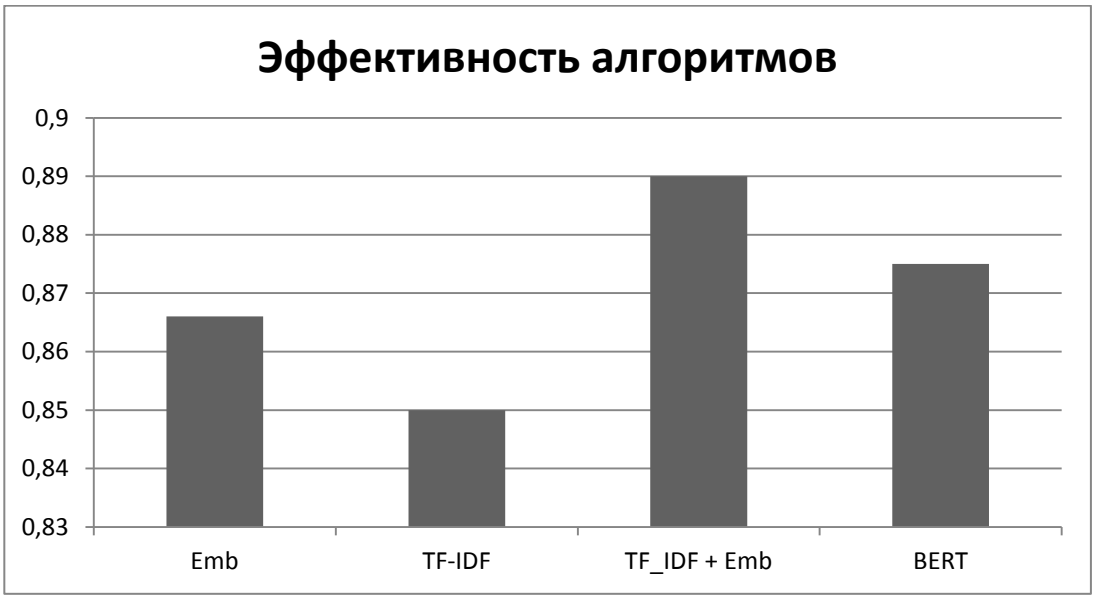

Рисунок 2.3. Эффективность алгоритмов

Далее строятся модели градиентного бустинга, основанных на *embeddings* (метод *doc2vec), TF-IDF* и модели, предложенной в разделе 2.1 и интегрирующей в себе *TF-IDF* и *embeddings*. Показатели AUC этих моделей 0.866, 0.85, 0.89 соответственно (см. рис. 2.3).

Таким образом, можно заметить, что предложенная модель эффективней с точки зрения качества предсказания, чем альтернативные подходы и может использоваться для анализа и числовой оценки текстовой информации.

Заключение

Анализ текстовой информации – активно развивающееся направление в области Data Science. Построенные модели машинного обучения помогают решать задачи в самых разных сферах общества. Данная работа посвящена исследованию целевой характеристики текстового контента. В данной работе рассматривался вопросы, касающиеся выявления смысловых особенностей текста, влияющих на его целевую характеристику.

В данной работе был предложен метод и апробирован на практике. Для исследования были взяты данные из социальных сетей, а именно пользовательские комментарии, содержащие оскорбительные высказывания по отношению к различным социальным группам. Своевременное выявление токсических и оскорбляющих комментариев является важной задачей, реализация которой позволит своевременно блокировать запрещенный контент, а также аккаунты пользователей экстремисткой направленности, а также повысить комфорт участников дискуссии и её продуктивность. Также это помогло бы избежать судебных исков за оскорбление чести и достоинства, а также предотвратить экстремизм и разжигание ненависти к социальным группам.

При выполнении дипломной работы, было установлено, что оскорбительные сообщения не всегда содержат бранную лексику, которая в свою очередь может использоваться для написания сообщений, не содержащих оскорбительного элемента.

Также немаловажным является предобработка данных, поскольку при её некорректном осуществлении часть информации может потеряться.

Результатом является разработанная модель, объединяющая в себе преимущества *embeddings* и *TF-IDF*, точность предсказания которой выше чем у использующихся отдельно моделей. Также, на момент написания данного заключения результат моих научных изысканий входил в топ 5% решений, на публичном лидерборде, т.е. публичной таблице результатов участников соревнования “Jigsaw Unintended Bias in Toxicity Classification” [27] на платформе онлайн соревнований по машинному обучению Kaggle. Что в свою очередь, является показателем перспективности разрабатываемого в этой работе подхода.

Также данной направление исследований может быть продолжено, поскольку

также могут быть выявлены и другие эффективные подходы, для повышения предсказательной способности модели. Также данные исследования могут быть расширены на смежные сферы задачи области Text Mining. Например, предсказание наиболее релевантных пользователю текстовых постов.

Библиографический список

1. Lihao Ge, Teng-Sheng Moh, "Improving text classification with word embedding", Big Data (Big Data) 2017 IEEE International Conference on, pp. 1796-1805, 2017.

2. Chunzi Wu, Bai Wang, "Extracting Topics Based on Word2Vec and Improved Jaccard Similarity Coefficient", Data Science in Cyberspace (DSC) 2017 IEEE Second International Conference on, pp. 389-397, 2017.

3. KOŠTIAL, Martin, and DAŘENA, František. 2018. Using Word Embeddings for Analysing Texts from the Educational Domain. PEFnet 2017 – 21 St European Scientific Conference of Doctoral Students, pp. 129–136. Mendel University in Brno. ISBN 978-80-7509-555-8.

4. Андреев А.М., Березкин Д.В., Морозов В.В., Симаков К.В. Автоматическая классификация текстовых документов с использованием нейросетевых алгоритмов и семантического анализа.

5. Ramos, Juan. "Using tf-idf to determine word relevance in document queries." Proceedings of the first instructional conference on machine learning. Vol. 242. 2003.

6. T. Mikolov, K. Chen, G. Corrado, J. Dean, "Efficient Estimation of Word Representations in Vector Space", Proc. Workshop at ICLR, 2013.

7. Laurent Hyafil, Ronald L. Rivest, Constructing optimal binary decision trees is NP-complete, Information Processing Letters, Volume 5, Issue 1, 1976, Pages 15-17, ISSN 0020-0190

8. Шахиди А. Деревья решений — общие принципы работы // basegroup.ru: Технологии Анализа данных. 2011. URL: https://basegroup.ru/community/articles/description (дата обращения: 03.05.2019)

9. Шахиди А. Логистическая регрессия | BaseGroup Labs // basegroup.ru: Технологии Анализа данных. 2011. URL: https://basegroup.ru/deductor/function/algorithm/logistic-regressiona

10. Quinlan, J.R. (1983b). Induction of Decision Trees. University of

Illinois at Urbana-Champaign

11. Natekin, Alexey & Knoll, Alois. (2013). Gradient Boosting Machines, A Tutorial. Frontiers in neurorobotics. 7. 21. 10.3389/fnbot.2013.00021.

12. Sherstinsky A. Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network // arXiv.org e-Print archive. 2018. URL: https://arxiv.org/abs/1808.03314l (дата обращения: 07.05.2019).

13. T. Mikolov, I. Sutskever, K. Chen, G. Corrado, J. Dean, "Distributed Representations of Words and Phrases and their Compositionality", Proc. NIPS, 2013.

14. Aizawa A. An information-theoretic perspective of tf–idf measures //Information Processing & Management. – 2003. – Т. 39. – №. 1. – С. 45-65.

15. Шитиков В. К., Мастицкий С. Э. Классификация, регрессия и другие алгоритмы Data Mining с использованием R //Режим доступу до ресурсу: http://ranalytics. blogspot. com/2012/03/f. html#. WnmcePll_IW. – 2017.

16. Dumais S., Platt J., Heckerman D. Inductive Learning Algorithms and Representation for Text Categorization.

17. Devlin J. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding// arXiv.org e-Print archive. 2018. URL: https://arxiv.org/abs/1810.04805 (дата обращения: 07.05.2019).

18. Narkhede S. Understanding AUC - ROC Curve // Towards Data Science e. 2018. URL: https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5 (дата обращения: 07.05.2019).

19. Jigsaw Unintended Bias in Toxicity Classification // Kaggle e. 2018. URL: https://www.kaggle.com/c/jigsaw-unintended-bias-in-toxicity-classification (дата обращения: 12.05.2019).

20. Jigsaw Unintended Bias in Toxicity Classification // Kaggle e. 2018. URL: https://www.kaggle.com/c/jigsaw-unintended-bias-in-toxicity-classification/leaderboard (дата обращения: 12.05.2019).