МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«ТЮМЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ НАУК
Кафедра программной и системной инженерии

РЕКОМЕНДОВАНО К ЗАЩИТЕ В ГЭК
Заведующий кафедрой
д.т.н., профессор
_____ *А.Г. Ивашко*
_____2021 г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
магистерская диссертация

Разработка и интеграция системы классификации документов с маркшейдерскими данными в ООО "Газпром недра"

09.04.03Прикладная информатика
Магистерская программа «Информационные системы анализа данных»

Выполнил работу
Студент 2 курса
очной формы обучения
Еремеев
Владимир
Вячеславович

Научный руководитель
к.т.н., доцент
Цыганова
Мария
Сергеевна

Рецензент
к.ф.н., доцент
Бидуля
Юлия
Владимировна

г. Тюмень, 2021

Содержание

ВВЕДЕНИЕ

ООО «Газпром недра» — специализированная многопрофильная компания, выполняющая централизованно полный цикл геологоразведочных работ на территории РФ и предоставляющая заказчикам широкий спектр уникальных геофизических и геолого-технических услуг. Производственная деятельность компании в области геологоразведки направлена на формирование предложений по развитию минерально-сырьевой базы, проведение подсчета запасов углеводородного сырья, постановку их на баланс в Государственной Комиссии по запасам, проведение опытно-промышленной эксплуатации и создание технологической схемы разработки месторождений. [1]

Одним из главных направлений деятельности ООО «Газпром недра» являются геологоразведочные работы. Геологоразведочные работы − комплекс различных специальных геологических и других работ, производимых с целью поиска, обнаружения и подготовки к промышленному освоению месторождений полезных ископаемых. Геологоразведочные работы включают изучение закономерностей размещения, условий образования, особенностей строения, вещественного состава месторождений полезных ископаемых с целью их прогнозирования, поисков, установления условий залегания, предварительной и детальной разведки, геолого-экономической оценки и подготовки к промышленному освоению. [2]

Проведение геологического изучения недр допускаются только при обеспечении безопасности жизни и здоровья работников этих предприятий и населения в зоне влияния работ, связанных с пользованием недрами. Одним из основных требований по обеспечению безопасного ведения работ, связанных с пользованием недрами, является проведение комплекса геологических, маркшейдерских и иных наблюдений, достаточных для обеспечения нормального технологического цикла работ и прогнозирования опасных ситуаций, своевременное определение и нанесение на планы горных работ опасных зон. [3] Помимо требования безопасного ведения геологоразведочных работ, маркшейдерское обеспечение необходимо для множества подготовительных работ, в частности:

организация подъезда техники,

организация разведочной площадки,

поиск источников и путей подвода воды.

Во всех этих работах, без разметки границ участков не обойтись. Поэтому службы, участвующие в этих процессах, должны формировать задачи для маркшейдерской службы.

Для формирования и контроля задач в ООО «Газпром недра» внедрена и действует система электронного документооборота. Каждый рабочий день в документообороте формируются новые документы, как поступающие из вне предприятия, так и формируемые внутри организации. Почти всегда с этими документами должны быть ознакомлены определенные заинтересованные лица, иногда сформированы задания для исполнителей и согласующих лиц. Для внутренних и исходящих документов круг лиц, которые должны ознакомится с документом формируется инициатором на основании шаблонов

и предположений инициатора. Для входящих документов круг лиц определяется сотрудником отдела документооборота.

Список согласующих для работы с документом формируется по шаблону, но иногда требуется расширить список согласующих. Такие задачи требуют внимательного изучения сути документа и подбор возможных заинтересованных лиц. Порой в процессе формирования задач на ознакомление забывают включить определенные отделы, в частности службу маркшейдеров. В следствии такой ошибки рабочие процессы маркшейдеров не выполняются или выполняются с большой задержкой, в случае если ошибка была обнаружена. Такие ошибки могут повлечь как задержку в выполнении проектов, так и штрафы для организации допустившей такую ошибку.

Для решения этой проблемы могут быть использованы различные методы обработки текстовой информации с последующей классификацией.

Для успешной подготовки и защиты выпускной квалификационной работы автором ВКР использовались средства и методы физической культуры и спорта с целью поддержания должного уровня физической подготовленности, обеспечивающую высокую умственную и физической работоспособность. В режим рабочего дня включались различные формы организации занятий физической культурой (физкультпаузы, физкультминутки, занятия избранным видом спорта) с целью профилактики утомления, появления хронических заболеваний и нормализации деятельности различных систем организма.

В рамках подготовки к защите выпускной квалификационной работы автором созданы и поддерживались безопасные условия жизнедеятельности, учитывающие возможность возникновении чрезвычайных ситуаций.

1 Описание предметной области

1.1 Маркшейдерское обеспечение

Методы и результаты маркшейдерских работ широко используются на нефте- и солепромыслах, при разведке месторождений полезных ископаемых, строительстве метрополитенов, туннелей и других сооружений. Важнейшими задачами маркшейдерской службы являются создание и ведение маркшейдерского обеспечения.

Под маркшейдерским обеспечением геологоразведочных и горных работ следует понимать маркшейдерскую геометрическую основу и документацию для решения ответственных инженерных задач на горном предприятии и выполнения оперативных производственных работ.

К основным инженерным задачам, требующим маркшейдерского обеспечения, относятся:

создание инженерных проектов и реализация их в производстве,

проведение выработок и безопасное выполнение горных работ в соответствии с проектными решениями и горно-геологическими условиями,

перспективное и текущее планирование горных работ,

оперативный подсчет запасов полезного ископаемого, обеспечивающий полноту извлечения запасов из недр и необходимое качество добываемого сырья,

охрана подрабатываемых залежей полезного ископаемого, горных выработок и сооружений, а также природных объектов. [4]

На каждое месторождение, запасы которого стоят на государственном балансе, должны быть следующие документы:

схема геодезического и маркшейдерского обоснования на район месторождения или всего участка недр с нанесенной координатной сеткой, гидросетью, пунктами и реперами,

каталог координат пунктов обоснования,

каталог координат геологоразведочных выработок и их линий,

схемы устройства пунктов обоснования с актами на скрытые работы по закладке центров в натуре (выполненные по согласованному проекту),

топографические планы поверхности.

С учетом закона "О недрах", можно сделать вывод маркшейдерское обеспечение является обязательной частью геологоразведочных работ. Также в случае проведения изыскательных работ без маркшейдерского обеспечения существуют серьёзные штрафы для компаний, участвующих в этих работах.

1.2 Система электронного документооборота

Система электронного документооборота (СЭД) — автоматизированная многопользовательская система, сопровождающая процесс управления работой иерархической организации с целью обеспечения выполнения этой организацией своих функций. При этом предполагается, что процесс управления опирается на человеко-читаемые документы, содержащие инструкции для сотрудников организации, необходимые к исполнению. [5]

СЭД в ООО «Газпром недра» основан на программном продукте 1С:Документооборот. Система обеспечивает автоматизацию полного цикла работы с документами, также позволяет упорядочить взаимодействие между сотрудниками и осуществлять контроль использования рабочего времени. Учёт документов реализован в соответствии с положениями действующей нормативной документации (ГОСТов, требований, инструкций и т. д.) и традиций делопроизводства. Программа обеспечивает многопользовательскую работу в локальной сети. Система отличается большой гибкостью, высокой степенью детализации сведений о хранящихся данных и широким спектром возможностей. Позволяет повысить эффективность управления рабочим временем, стандартизировать процессы, обеспечить полный контроль и сохранность документации и любой иной необходимой информации. функциональность системы постоянно расширяется. [6]

С начала эксплуатации 1С:Документооборот его функционал был значительно расширен сотрудниками ООО «Газпром недра». Типичный порядок формирования новых документов в СЭД выглядит примерно так:

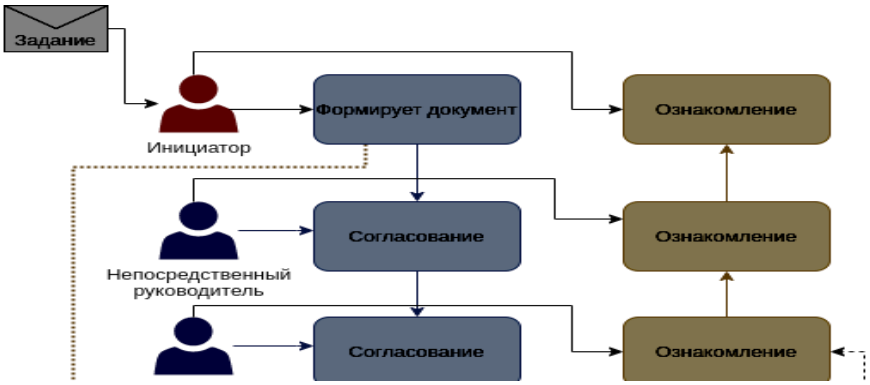
1) инициатор создает документ с адресатами. Обычно адресатами являются заместители генерального директора тех направлений к которым инициатор направляет задачи. На этом этапе иногда забывают включить маркшейдерскую службу в процессы, где их обеспечение необходимо,
2) непосредственный руководитель проверяет документ и согласовывает, если его все устраивает. Тут еще возможно устранить ошибку с маркшейдерским обеспечением,
3) руководитель службы проверяет документ и согласовывает если его все устраивает. Тут тоже возможно устранить ошибку инициатора, но вероятность ниже из-за большего объема проходящих через него документов,
4) отдел документационного обеспечения проверяет документ на шаблонность и перенаправляет документы на руководителя службы исполнителя, минуя согласования заместителями генерального директора. Это делается для ускорения движения типовых документов,
5) если же документ не шаблонный, то он согласовывается заместителем генерального директора от инициатора,
6) заместитель генерального директора от исполнителя формирует резолюцию со списком исполнителей, сроками исполнения и другими атрибутами,

7) руководитель службы и непосредственный руководитель от исполнителя уточняют задачи и распределяют их по исполнителям,

8) исполнители выполняют задачи и готовят ответный документ извещающий об исполнении поставленной задачи,

9) далее в обратном порядке извещение движется к инициатору.

Схема движения документов представлена на рисунке 1:



1.3 Система распознавания документов

Система распознавания создавалась для организации полнотекстового поиска с помощью ElasticSearch в СЭД. Построена она на микросервисной архитектуре с использованием очереди событий. Обычно такую архитектуру называют событийно-ориентированная архитектура.

Система распознавания состоит из нескольких микросервисов, выполняющих каждый свою задачу. Входными данными для системы являются сканированные документы из СЭД. Выходным результатом системы распознавания являются записи в базе данных, в которых содержатся метаданные документов и распознанный текст из них. Сама система распознавания не ведет классификацию документов. И была создана для организации полнотекстового поиска, как было сказано выше.

Для задачи связанной с классификацией маркшейдерских документов важно использовать следующие функции системы:

постановку задачи распознавания,

извлечение результата распознавания.

В остальном система распознавания довольно сложная и объем ее описания значительный.

1.4 Проблематика

Не у всех инициаторов достаточно опыта ведения геологоразведочных проектов, чтоб оценить необходимость маркшейдерского обеспечения. Также опыт сотрудников отдела документационного обеспечения порой не достаточен для правильного перенаправления документов, исходя из содержимого документов. Помимо опыта сотрудников постоянно нарастающий объем создаваемых документов в СЭД ООО «Газпром недра» повышает вероятность пропуска создания процессов на маркшейдерское обеспечение сотрудниками службы документационного обеспечения. Если включить в рабочие процессы обработки всех новых документов службу маркшейдеров, то служба будет перегружена бесполезной работой и не будет успевать справляться со своими основными обязанностями.

Классификация могла бы помочь в решении этой проблемы. Как вариант решения проблемы документы, содержащие пространственные данные, направлялись бы на ознакомление в службу маркшейдеров или в форме подсказок в карточке документа отражалась бы информация.

Ранее в ООО «Газпром недра» не выполнялась задача программной классификации документов, в связи с этим не созданы инструменты для классификации, а именно:

1) алгоритмов очистки текста и векторизации распознанных документов,
2) обучаемых алгоритмов классификации,
3) структуры хранения результатов классификации, сбора реальных меток класса от пользователей,
4) автоматического формирования новых обучающих выборок,
5) хранения моделей для классификации и выбора наиболее эффективной из существующих.

Основным и вероятно единственным потребителем результатов классификации из внешних систем будет СЭД. На данный момент еще формируется способ использования классификации документов системой электронного документооборота. Но уже очевидно, что системе классификации потребуются restful api интерфейсы для классификации и обучения.

1.5 Требования к разрабатываемой системе

Входом для классификации должен быть текст документа, а в ответ система должна выдавать метку класса. Информационная система классификации документов с маркшейдерскими данными должна выполнять бинарную классификацию по классам:

содержит документ пространственные данные,

не содержит пространственные данные

с точностью выше 90%.

Система должна периодически дообучаться на новых данных. Для дообучения на вход будет подаваться список текстов с их реальными метками классов, по итогу должна быть обученная модель и её характеристики.

Разрабатываемая система должна функционировать самостоятельно и быть готовой принять запросы на классификацию из внешних систем. С точки зрения безопасности доступ к системе необходимо предоставить ограниченному кругу сервисов и разработчикам для диагностики. Необходимости в графическом интерфейсе нет. Опционально можно создать минимальный интерфейс для диагностики качества моделей машинного обучения.

2 Актуальность работы и постановка задачи

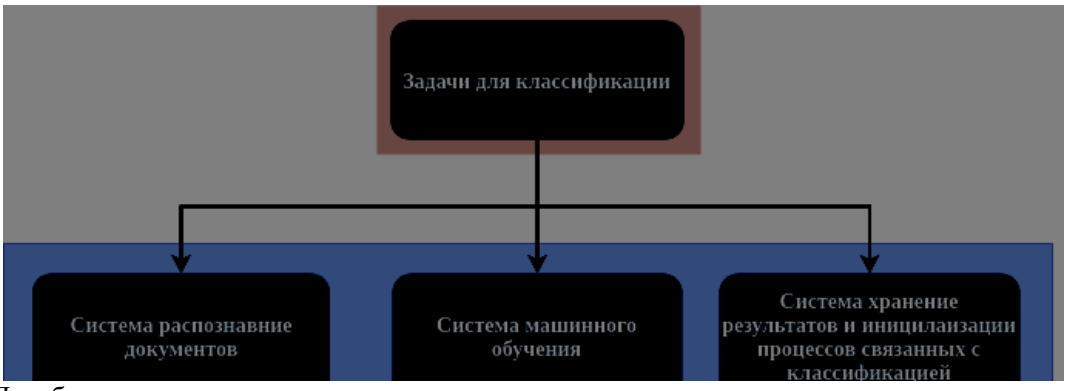
Важность маркшейдерского обеспечения выделена во введении, источник проблемы — возрастающий объем ежедневных документов, как следствие невнимательность инициаторов и сотрудников отдела документооборота при прочтении документов.

Цель системы классификации маркшейдерских документов - организация рекомендательной надстройки над 1С:Документооборот ООО «Газпром недра».

Для достижения цели системы классификации необходимо, чтоб были решенными следующие задачи:

1) преобразовать документы в текст,
2) организовать хранение для результатов распознавания,
3) выполнить предварительную обработку текста для последующей классификации,
4) найти наиболее подходящую модель машинного обучения для классификации,
5) сделать последовательность действий для удобной и быстрой для классификации документов,
6) сделать последовательность действий для обучения новых моделей,
7) организовать хранение результатов обучения для диагностики и активации,
8) сделать RestfulAPI интерфейсы для модели, результатов обучения,
9) организовать хранение с результатов классификации, реальных меток классов,
10) сделать RestfulAPI интерфейсы для работы с результатами классификации,
11) сделать интерфейсы связи с СЭД, обратную связь,
12) организовать формирование сбалансированной обучающей выборки.

Эти задачи стоит разнести по отдельным системам, для удобства масштабирования. На рисунке 2 представим схему распределения задач:



Преобразование текста и организация хранения результатов распознавания уже реализованы на предприятии.

В рамках этой работы будут реализовываться задачи описанные в разделе система машинного обучения. Система хранения и инициализация процессов связанных с классификацией будет связующим звеном между СЭД и системой машинного обучения. В ней предполагается хранение и наполнение размеченных в ходе рабочих процессов СЭД маркшейдерами данных. Периодически будет происходить формирование новых обучающих выборок и отправка их в систему машинного обучения для тренировки моделей на новых данных. Система хранения и инициализация процессов связанных с классификацией будут реализована другими участниками команды разработки. Подробнее описание взаимодействия систем представлено в разделе 4.1 этой работы.

Прежде чем создавать архитектуру информационной системы считаю необходимым выполнить поиск подходящей модели машинного обучения, способной решить задачу классификации документов. В связи с этим в первую очередь необходимо сосредоточится на предварительной обработке текста и поиске наиболее подходящей модели машинного обучения.

3 Анализ данных

3.1 Основные характеристики обрабатываемых документов

В результате исследования обнаружено, что документы в размеченной выборке обладают следующими характеристиками:

1) входящими данными являются распознанные сканированные документы и документы doc, docx,
2) документы размечены и имеют два класса: 0 — нет маркшейдерских данных, 1 — маркшейдерские данные в документе присутствуют,
3) всего 450 документов из них 150 относятся к 1 классу,
4) длина документов различная как и содержание,
5) в основном структура документов состоит из следующих разделов - шапка, обращение, тело сообщения и подписи, иногда бывают приложения. В идеале было бы хорошо извлечь только тело письма и приложение,
6) шапка и обращение в документах имеют один важный и удобный паттерн «Уважаемый(ая)», который позволяет удалить шапку из обработки,
7) а вот с окончание тела сообщения несколько сложнее — не известно сколько отступов выставила система распознавания от тела до подписи, поэтому эту часть письма стоит изучить повнимательнее,
8) в разделе подписи обязательно присутствуют должности в именительном падеже, ФИО подписантов, сокращаемых по всякому. Часто в этом разделе есть ФИО инициатора, телефон, адрес электронной почты,
9) в разделе тело сообщения находится наиболее важная информация, но как обычно бывает она разбавлена словами не несущими никакой смысловой нагрузки. Раздел тело сообщения может содержать ФИО, должности, названия организационно-структурных подразделений, название организаций, телефоны, адреса электронных почтовых ящиков,
10) в маркшейдерских документах встречается похожие записи содержащие наименование системы координат. Эти записи немного отличаются друг от друга,
11) приложения могут содержать полезную для классификации информацию, но там часто встречаются таблицы и спец символы,
12) в некоторых документах результат распознавания оставляет желать лучшего, так как часто встречаются слитые слова, слова с ошибочно распознанными символами. По всем документам огромное количество отступов, что затрудняет анализ документов.

Основные выводы этого анализа данных получены итеративным путем поэтапной чистки документов. На основе этих данных будет строится процесс предобработки и в дальнейшем модель машинного обучения.

3.2 Предварительная обработка данных

Неструктурированные тексты обычно содержат много зашумленной, ненужной, бесполезной информации, такой как повторяющиеся слова, числа, знаки препинания, стоп-слова, сокращения, орфографические ошибки, ярлыки и специфическая терминология. Поскольку каждое слово рассматривается как измерение в наборе свойств, наличие ненужных слов приводит к путанице в моделях и потере времени. С другой стороны, очистка текста от шума может повысить производительность классификаторов и ускорить процесс классификации. [7]

Обработка документа является последовательной задачей и часто улучшает результаты работы методов машинного обучения в задачах классификации.

Для удаления шапки выполняем последовательный поиск по обнаруженному паттерну «Уважаемый(ая)» и отрезаем часть документа включая это слово.

Перед дальнейшей очисткой полезные слова имеет смысл заменить кодовым словом, в моём случает такое слово это «системакоординат». Далее удаляем слова содержащие не кириллицу — чаще всего эти слова редкие или являются адресом электронной почты, номером телефона, результатом ошибочного распознавания. После избавляемся от лишних пробелов и других знаков форматирования документа. Переводим весь текст в нижний регистр для удобства дальнейшей обработки.

ФИО, должности, структурные подразделения, названия организаций на первый взгляд кажутся полезными данными, но при каком-либо изменении организационной структуры предприятия в будущем эти данные будут только мешать. К примеру уволился главный маркшейдер предприятия и модель не должна ориентироваться на его фамилию больше, или произошли изменения организационной структуре и отделы теперь по другому называются. Поэтому хорошо бы эти данные удалить, ну и как плюс это поможет очистить раздел подписи. Для удаления этих данных необходимо иметь справочники по каждой группе.

Благо эти данные есть в учетных системах предприятия. Правда и тут есть сложность, данные не везде аккуратно заполнены и их надо предобработать. К примеру ФИО сокращают по разному, поэтому лучше вытащить все слова без сокращений. Таким образом получится множество, где по отдельности хранятся фамилии, имена, отчества. Дополнительно по подразделениям и должностям нужно добавить их лемматизированные формы, может пригодится для глубокой очистки.

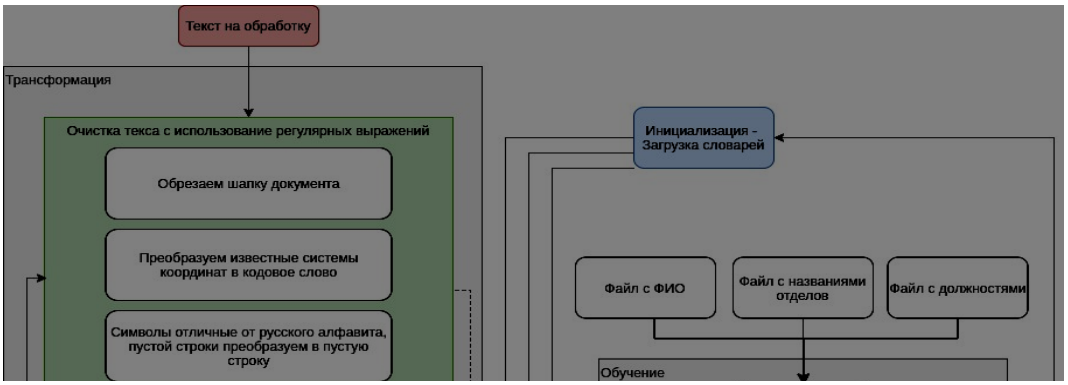
Тем не менее справочники сформированы. Вычищаем при помощи поиска совпадений все эти данные, кроме ФИО. Далее этап лемматизации.

Лемматизация - это процесс определения базовой или словарной формы (леммы) для данной формы поверхности. Традиционно формы словарной базы использовались в качестве входных функций для различных задач машинного обучения, таких как синтаксический анализ, но также находят

применение в индексировании текста, лексикографической работе, извлечении ключевых слов и во многих других приложениях с поддержкой языковых технологий. Лемматизация особенно важна для языков с богатой морфологией, где в приложениях требуется строгая нормализация. Основные трудности в лемматизации возникают из-за встречи с ранее невидимыми словами во время вывода, а также устранения неоднозначности поверхностных форм, которые могут быть склонными вариантами нескольких различных базовых форм в зависимости от контекста.[8]

Для лемматизации используем приложение Yandex MyStem, идея работы алгоритма и сравнение с другими решениями описаны в статье Ильи Сегаловича. [9] с библиотекой pymystem для использования приложения из python. В результате образуются токены, их очищаем от стоп-слов, от слов из списка ФИО, от токенов длиной меньше 2 и больше 25 символов. По окончанию склеиваем список через пробел.

Всю предобработку оборачиваю в класс трансформации входящих данных. В классе используется две функции обучение и трансформация. На рисунке 3 представлена схема предобработки текста:



Пример работы предобработки текста представлен в приложении А в таблице 1.

3.3 Векторизация

Для алгоритмов машинного обучения необходимо текст преобразовать в цифровое представление. Существует несколько разных типов векторизации текстовых данных:

прямое кодирование (one-hot encoding) считается самым простым способом преобразования токенов в тензоры и выполняется следующим образом: каждый токен представляет бинарный вектор (значения 0 или 1) единица ставится тому элементу, который соответствует номеру токена в словаре,

мешок слов (Bag of words) выделяет вектору весь документ, и каждый элемент кодируется 1 по порядку следования слов в словаре,

TF-IDF состоит из двух компонентов: Term Frequency (частотность слова в документе) и Inverse Document Frequency (инверсия частоты документа). Стоит отметить, что TF считается для токенов документа, тогда как IDF — токенов всего корпуса. В TF-IDF редкие слова и слова, которые встречаются во всех документах, несут мало информации, [10]

Word Embeddings - способ построения сжатого пространства векторов слов, использующий нейронные сети. Представления слов, вычисленные с помощью нейронных сетей, очень

интересны, потому что выученные векторы явно кодируют многие лингвистические закономерности и шаблоны. [11]

В данном случае мы работаем с длинными текстами в небольшом корпусе — 450 документов. Мешок слов и прямое кодирование достаточно примитивные методы векторизации, поэтому лучше их пропустить и использовать TF-IDF векторизацию, тем более что есть удобные средства в составе библиотеки scikit-learn. [12]

Word Embeddings наиболее часто используется в последовательной обработке слов в тексте глубокими нейронными сетями. Есть ситуации где используют сверточные нейронные сети для классификации текстов. Но в такой классификации необходимы достаточно большие выборки. В нашем случае пока размечены только 450 документов.

К маленьким наборам данных, модели поверхностного обучения обычно обеспечивают лучшую производительность, чем модели глубокого обучения при ограничение вычислительной сложности. [13] Поэтому пока Word Embeddings можно пропустить и использовать TF-IDF.

3.4 Подбор метода машинного обучения

3.4.1 Выбор алгоритмов машинного обучения

Для начала все размеченные данные надо поделить на обучающую и тестовую выборки. При разбиение данные необходимо перемешать, но сделать это так чтоб в обеих выборках было примерно одинаковое соотношение классов. Такое разбиение позволит избежать случаев когда в тестовой выборке окажется малое количество примеров редкого класса и тем самым сделать тестирование более объективным. В исследовательских целях состояние перемешивания необходимо зафиксировать, для удобства сравнения результатов.

В качестве алгоритмов обучения с учителем возьмем наиболее распространенные алгоритмы поверхностного обучения:

наивный Байесовский классификатор - простой вероятностный классификатор, основанный на применении теоремы Байеса со строгими (наивными) предположениями о независимости. Ключевое предположение наивных байесовских классификаторов состоит в том, что слова в документах условно независимы с учетом значения класса. [14],

SGDClassifier - это обобщенный линейный классификатор, который будет использовать стохастический градиентный спуск в качестве решателя [15],

логистическая регрессия - это статистическая модель, используемая для прогнозирования вероятности возникновения некоторого события путём его сравнения с логистической кривой. Эта регрессия выдаёт ответ в виде вероятности бинарного события [16],

Случайные леса представляют собой комбинацию предикторов деревьев, так что каждое дерево зависит от значений случайного вектора, выбранных независимо и с одинаковым распределением для всех деревьев в лесу. Ошибка обобщения для лесов сходится до предела по

мере того, как количество деревьев в лесу становится большим. Ошибка обобщения леса классификаторов деревьев зависит от силы отдельных деревьев в лесу и корреляции между ними. [17],

метод k-ближайших соседей (k-nearest neighbors algorithm, kNN) -метрический алгоритм для автоматической классификации объектов или регрессии. В основе алгоритма K-ближайших соседей лежит классификация немаркированной выборки путем нахождения категории с наибольшим количеством выборок в k-ближайших помеченных выборках. Это простой классификатор без построения модели, который может снизить сложность за счет быстрого получения k ближайших соседей, [18]

метод опорных векторов - набор схожих алгоритмов обучения с учителем. Основная идея метода — перевод исходных векторов в пространство более высокой размерности и поиск разделяющей гиперплоскости с максимальным зазором в этом пространстве. Метод опорных векторов основывается на принципе минимизации структурных рисков из теории вычислительного обучения. Идея минимизации структурных рисков заключается в том, что мы предполагаем, что мы гарантируем минимальную истинную ошибку, [19]

многослойный перцептрон — частный случай перцептрона Розенблатта, в котором один алгоритм обратного распространения ошибки обучает все слои. Перцептрон— математическая или компьютерная модель восприятия информации мозгом (кибернетическая модель мозга). Многослойный перцептрон - один из наиболее часто используемых типов искусственных нейронных сетей. В течение последних четырех десятилетий искусственные нейронные сети активно исследуются и используются в реальных промышленных решениях. Мощь искусственных нейронных сетей находится в стадии эксплуатации. После обучения искусственные нейронные сети становятся чрезвычайно эффективным инструментом. [20]

3.4.2 Поиск оптимальных параметров для алгоритмов

В качестве стратегии подбора параметров выбран поиск по сетке гиперпараметров с кроссвалидацией и перемешиванием с учетом изначального баланса классов. [21] Параметры для сетки выбирались к каждому классификатору отдельно, на основании наиболее вариативных параметров. К примеру для метода k-ближайших соседей используется следующая сетка параметров:

количество ближайших соседей: интервал 2-10 с шагом 1,

веса точек: все точки одинаковым весом, точки которые находятся ближе имеют больший вес.

Кроссвалидация необходима для поиска наилучшей выборки для обучения из доступных. Перемешивание с учетом соотношения баланса классов позволяет проводить обучение не опасаясь, что в выборку попадут представители одного класса.

Для кроссвалидация необходимо выбрать метрику качества, по какому параметру сравнивать выборки. Существуют различные метрики для методов машинного обучения, все они имеют свою специфику применения. Для классификации наиболее часто используют следующие метрики:

accuracy - базовая метрика. Оценивает общее соотношение корректных предсказаний модели к общему числу наблюдений в выборке,

precision - точность показывает, какая часть положительно классифицированных примеров предсказана корректно,

recall - полнота показывает, какая часть положительных примеров классифицирована корректно,

F-Measure - F-мера представляет гармоническое среднее между точностью и полнотой, позволяя оптимизировать сразу две эти метрики,

area under (roc) curve (auc) - позволяет получить интегральную оценку качества модели, не принимая во внимание эффекты от вариации порога отсечения (threshold) [22].

Маркшейдерам предпочтительнее, чтоб было меньше ошибок с пропуском документов относящихся к ним. В дальнейшем проводились испытания с поиском лучших параметров классификаторов по метрике recall для первого класса. Результаты были не очень хорошими, полнота (recall) в 1 классе значительно ухудшила точность (precission) по обоим классам. В качестве оптимальной метрики способной подобрать оптимальное соотношение ошибок выбрана метрика площадь под кривой ROC.

В библиотеке Scikit-Learn присутствуют различные модификации этой характеристики. Наиболее подходящий для текущей задачи из них считаю roc_auc_ovo_weighted. Ovo - расшифровывается как «Один против одного». Вычисляет средний AUC всех возможных попарных комбинаций классов. Weighted - вычислить метрики для каждой метки, и найти их среднее значение, взвешенное по количеству истинных экземпляров для каждой метки. [23]

Каждый из выше перечисленных классификаторов по отдельности ищет наилучшие гиперпараметры по метрике roc_auc_ovo_weighted.

3.4.3 Тестирование моделей

Далее эти классификаторы проверяются на тестовой выборке. Для наблюдения за качеством классификации удобно смотреть для каждой модели отчет о классификации и матрицу ошибок.

Отчет о классификации содержит в себе точность, полноту, F-меру по каждому классу, общую и взвешенную, общий Accuracy. Отчет о классификации позволяет оценить модель в общем и сравнить её с другими моделями. [24]

Матрица ошибок отображает в абсолютных значениях сколько правильно классифицированных экземпляров первого класса и сколько не правильно классифицированных, и все тоже самое для второго класса. [25]

Каждый классификатор оцениваем по тестовой выборке и рассматриваем в отдельности. Результаты представлены в таблице 1:

Таблица 1 — *Результаты тестирования различных классификаторов*

Название	roc_auc	Матрица ошибок	Отчет о классификации				
Линейный классификатор со стохастическим градиентным спуском в качестве решателя	0.89409	51 3 5 27		precision	recall	f1-score	support
			False	0.91	0.94	0.93	54
			True	0.90	0.84	0.87	32
			accuracy			0.91	86
			macro avg	0.91	0.89	0.90	86
			weighted avg	0.91	0.91	0.91	86
Случайный лес	0.83159	51 3 9 23		precision	recall	f1-score	support
			False	0.85	0.94	0.89	54
			True	0.88	0.72	0.79	32
			accuracy			0.86	86
			macro avg	0.87	0.83	0.84	86
			weighted avg	0.86	0.86	0.86	86
Метод k-ближайших соседей	0.86632	48 6 5 27		precision	recall	f1-score	support
			False	0.91	0.89	0.90	54
			True	0.82	0.84	0.83	32
			accuracy			0.87	86
			macro avg	0.86	0.87	0.86	86
			weighted avg	0.87	0.87	0.87	86
Многослойный перцептрон	0.86921	50 4 6 26		precision	recall	f1-score	support
			False	0.89	0.93	0.91	54
			True	0.87	0.81	0.84	32
			accuracy			0.88	86
			macro avg	0.88	0.87	0.87	86
			weighted avg	0.88	0.88	0.88	86
Наивный Байесовский классификатор	0.87211	52 2 7 25		precision	recall	f1-score	support
			False	0.88	0.96	0.92	54
			True	0.93	0.78	0.85	32
			accuracy			0.90	86
			macro avg	0.90	0.87	0.88	86
			weighted avg	0.90	0.90	0.89	86

Название	roc_auc	Матрица ошибок	Отчет о классификации				
Логистическая регрессия	0.80324	53 1 12 20		precision	recall	f1-score	support
			False	0.82	0.98	0.89	54
			True	0.95	0.62	0.75	32
			accuracy			0.85	86
			macro avg	0.88	0.80	0.82	86
			weighted avg	0.87	0.85	0.84	86
Метод опорных векторов	0.93171	50 4 2 30		precision	recall	f1-score	support
			False	0.96	0.93	0.94	54
			True	0.88	0.94	0.91	32
			accuracy			0.93	86
			macro avg	0.92	0.93	0.93	86
			weighted avg	0.93	0.93	0.93	86

Если оценить в общем результаты, то они далеко не лучшие, много зависит от разбиения выборок. Хорошие результаты при случайном разбиении были получены следующими моделями:

SGDClassifier,

RandomForestClassifier,

метод k-ближайших соседей ,

многослойный перцептрон,

наивный Байесовский классификатор,

метод опорных векторов.

Часть методов показали roc auc выше 86 %, логистическая регрессия, RandomForestClassifier показала ниже результаты. Но в отличие от логистической регрессии RandomForestClassifier не является линейным классификатором как SGDClassifier, метод опорных векторов и может пригодится в качестве разнообразия для ансамблирования.

Далее проводим испытания с различным перемешиванием данных. А также по результатам испытания замечено,что результаты, то один то другой метод опережает остальные на 2-10% процентов. Отсюда можно сделать вывод, ввиду небольшой выборки присутствует нестабильность в результатах классификации, в зависимости от разделения и перемешивания выборок. В качестве решения текущей проблемы можно использовать ансамбль моделей машинного обучения состоящий из следующих моделей:

SGDClassifier,

наивный Байесовский классификатор,

RandomForestClassifier,

метод k-ближайших соседей,

многослойный перцептрон,

метод опорных векторов.

3.4.4 Ансамбль моделей машинного обучения

Ансамблем (Ensemble, Multiple Classifier System) называется алгоритм, который состоит из нескольких алгоритмов машинного обучения, а процесс построения ансамбля называется ансамблированием (ensemble learning). В задачах классификации простейший пример ансамбля – комитет большинства, представлен формуле (1):

$$a(x) = \text{mode}\left(b_1(x), \ldots, b_n(x)\right), \quad (1)$$

где mode – мода (значение, которое встречается чаще других среди аргументов функции). Если рассмотреть задачу классификации с двумя классами {0, 1} и три алгоритма, каждый из которых ошибается с вероятностью p, то в предположении, что их ответы – независимые случайные величины, получаем, что комитет большинства этих трёх алгоритмов ошибается с вероятностью $p^2(3-2p)$. Как видно на рисунке 4,

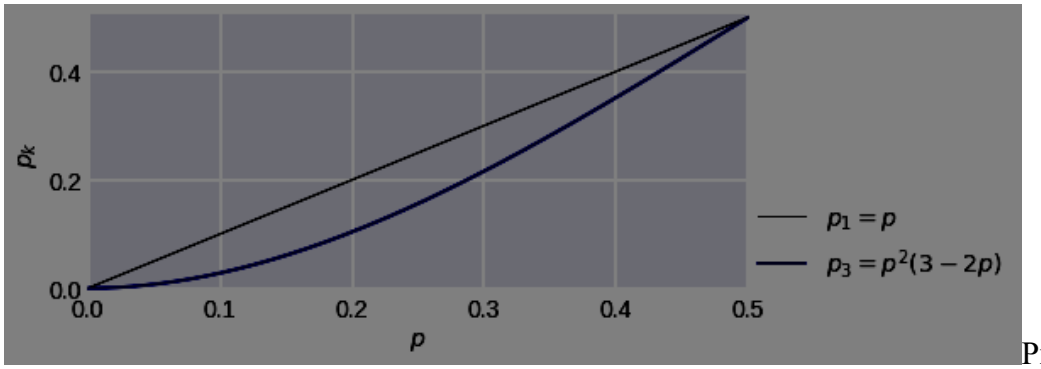


Рисунок 4 — График вероятности ошибки комитета большинства.

это выражение может быть существенно меньше p (при p=0.1 почти в два раза), т.е. использование такого ансамбля уменьшает ошибку базовых алгоритмов.

Наше теоретическое обоснование не совсем годится для реальной практической ситуации, поскольку регрессоры, классификаторы не являются независимыми:

решают одну задачу,

настраиваются на один целевой вектор,

могут быть из одной модели (или из нескольких, но всё равно небольшого числа).

Поэтому большинство приёмов в прикладном ансамблировании направлено на то, чтобы ансамбль был «достаточно разнообразным», тогда ошибки отдельных алгоритмов на отдельных

объектах будут компенсироваться корректной работой других алгоритмов. По сути, при построении ансамбля:

повышают качество базовых алгоритмов,

повышают разнообразие (diversity) базовых алгоритмов.

Как будет показано дальше, разнообразие повышают за счёт:

«варьирования» обучающей выборки (бэгинг),

«варьирования» признаков (Random Subspaces),

«варьирования» целевого вектора (ECOC, деформации целевого признака),

«варьирования» моделей (использование разных моделей, стэкинг),

«варьирование» в модели (использование разных алгоритмов в рамках одной, рандомизация в алгоритме – случайный лес).

Основные модели ансамблирования представлены в таблице 2: [26]

Таблица 2 — Модели ансамблирования

Модель или общая идея	Описание и примеры
комитеты (голосование) / усреднение	Построение независимых алгоритмов и их усреднение / голосование по ним, в том числе с помощью бэгинга и предварительной деформации ответов. Здесь же и обобщения бэгинга – случайные леса.
кодировки / перекодировки ответов	Специальные кодировки целевых значений и сведение решения задачи к решению нескольких задач. Один из самых популярных приёмов – ECOC (error-correcting output coding). Другой – настройка на разные деформации целевого признака.
стекинг (stacking)	построение метапризнаков – ответов базовых алгоритмов на объектах выборки, обучение на них мета- алгоритма
бустинг (boosting)	Построение суммы нескольких алгоритмов. Каждое следующее слагаемое строится с учётом ошибок предыдущих: AdaBoost, градиентный бустинг.
«ручные методы»	Эвристические способы комбинирования ответов базовых алгоритмов (с помощью визуализаций, обучения в специальных подпространствах и т.п.)
однородные ансамбли	Пример – нейронные сети. Формула мета-алгоритм – базовые алгоритмы разворачивается рекурсивно, применяется общая схема оптимизации полученной конструкции.

Некоторые классификаторы из исследованных используют внутри своей реализации принципы ансамблирования, поэтому особых ухищрений в дополнительном улучшении результатов

классификации не требуется и вряд ли получится выжать из них что-то ещё. Но в нашем случае для повышения стабильности в предсказаниях подходящей моделью считаю комитеты (голосование) / усреднение.

В библиотеке scikit-learn присутствует класс Votingclassifier [27], в нём наиболее интересны два параметра voting и **weights,** варианты применения:

hard — вложенные классификаторы дают свои метки класса, ансамбль выбирает ту метку, которую выбрали большая часть классификаторов,

soft — вложенные классификаторы дают свои вероятности принадлежности к классу, ансамбль берет от них среднее арифметическое. Этот параметр рекомендуется для хорошо откалиброванных моделей.

weights — позволяет выставить веса вероятностям принадлежности к классу для **каждого вложенного классификатора.**

В зависимости от разбиения эффективность основных классификаторов различается, поэтому подобрать подходящее значение для параметра weights не представляется возможным. Классификаторы были подобраны так что они имеют возможность вычислять вероятность принадлежности к классу. Ансамбль с параметром voting="hard" показал на 2% меньше roc_auc, чем ансамбль с параметров voting="soft".

Лучшие результаты работы при свободном разбиении выборки ансамбля представлены в таблице 3:

Таблица 3 — *Результаты тестирования ансамбля классификаторов*

Название	roc_auc	Матрица ошибок	Отчет о классификации				
Ансамбль классификаторов	0.9438	53 1 3 29		precision	recall	f1-score	support
			False	0.95	0.98	0.96	54
			True	0.97	0.91	0.94	32
			accuracy			0.95	86
			macro avg	0.96	0.94	0.95	86
			weighted avg	0.95	0.95	0.95	86

4 Реализация системы машинного обучения

4.1 Архитектура

Чтоб понимать место системы машинного обучения в общей задаче, на рисунке 5 представлена предполагаемая схема взаимодействия системы машинного обучения с другими информационными системами.

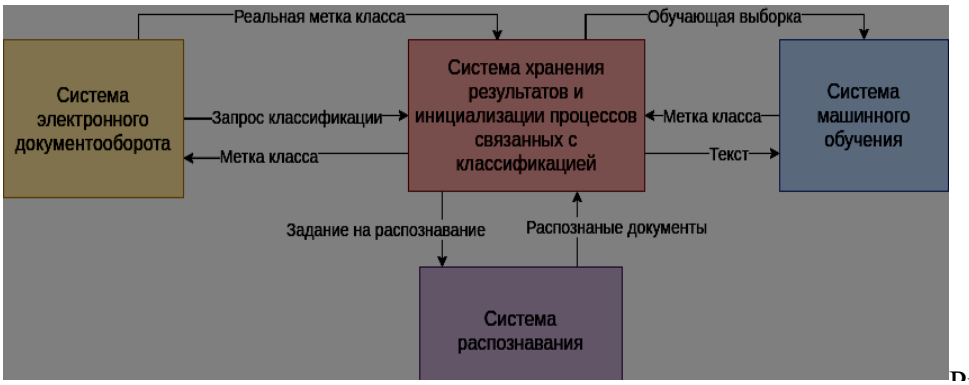


Рисунок 5 — *Структура системы машинного обучения*

Подробная структура информационной системы машинного обучения представлена на рисунке 6:

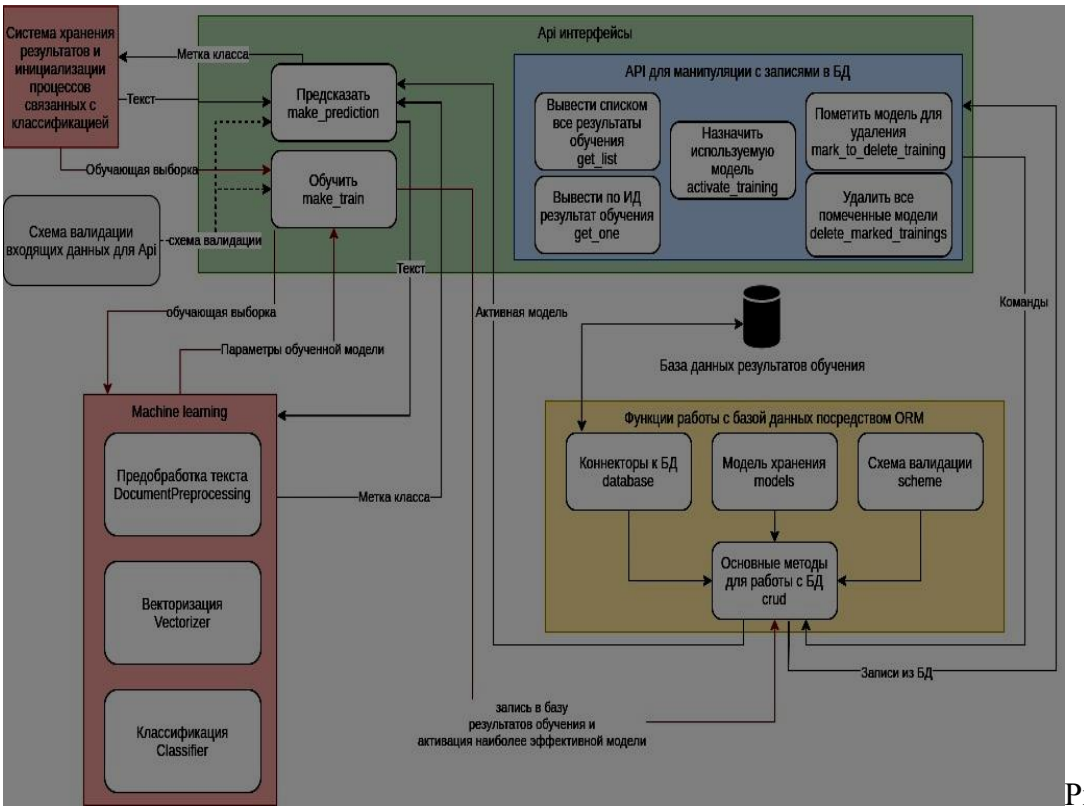


Рисунок 6 — *Структура системы машинного обучения*

Из структуры можно выделить три важных фрагмента и один дополнительный:

подсистема машинного обучения (Machine learning),

функции работы с базой данных результатов обучения,

api интерфесы,

схемы валидации для входящих данных для api.

Связь внешних сущностей с системой осуществляется через restful api интерфейсы. Сами интерфейсы и функции в них реализованные можно разделить на основные и вспомогательные. К вспомогательным относятся функции работы с записями в базе данных результатов обучения. А вот основные стоит описать подробнее:

1) предсказать выполняет следующую последовательность:
    1) читается из базы данных активную модель, её имя,
    2) загружаются модель по имени из папки хранения моделей,
    3) входящие данные от внешних систем проверяются валидатором,
    4) текст из входящих данных проходит предобработку,
    5) обработанный текст векторизуется TF-IDF векторизатором,
    6) классификатор по вектору выдает метку принадлежности к классу;
2) обучить выполняет следующую последовательность:
    1) входящие данные проверяются валидатором,
    2) выполняется предобработка текста всей выборки,
    3) обработанный текст векторизуется TF-IDF векторизатором,
    4) разбивается выборка в соотношении 80% обучающая, 20 % тестовая выборки,
    5) на обучающей выборке выполняется обучение модели,
    6) на тестовой выборке проверяются характеристики модели,
    7) модель сохраняется в файл,
    8) результаты тестирования и параметры модели записываются в базу данных с результатами обучения,
    9) из базы читается активная модель и сравнивается по roc_auc с последним обучением, в случае если roc_auc новой модели лучше, то старая модель перестает быть активной и активной моделью назначается новая.

4.2 Подсистема машинного обучения

За образ организации классов и функции считаю правильным взять, то как это организованно в scikit-learn.

Для обработки текста взять структуру с двумя главными функциями fit — обучение и transform — преобразование. Fit изначально будет считывать подготовленные словари, но в будущем можно будет

преобразовать в считывание и преобразование данных из первичных источников. Transform будет выполнять основную функцию подготовки исходного текста и его преобразования в текст необходимый для дальнейшей векторизации.

В качестве векторизатора логично взять готовый класс TfidfVectorizer из библиотеки scikit-learn, так как он выполняет все необходимые нам функции fit и transform. На этапе обучения модели fit собирает словарь, расставляет веса и transform преобразовывает входной корпус текстов в векторное представление TF-IDF. На этапе классификации будем использовать, только функцию transform, так как обученной модели важно использовать такую же размерность входных векторов, как и на этапе обучения.

В качестве модели машинного обучения стоит собрать класс, который будет реализовать следующие функции:

fit — функция в которой будут обучатся базовые классификаторы и VotingClassifier,

predict — предсказание для всего ансамбля,

predict_proba — вероятность принадлежности к классу для всего ансамбля,

score — вычисление метрик моделей, применяется для всего ансамбля,

save — сохранить модель в файл,

load — загрузить модель из файла.

Для наглядности схема классификатора представлена на рисунке 7:

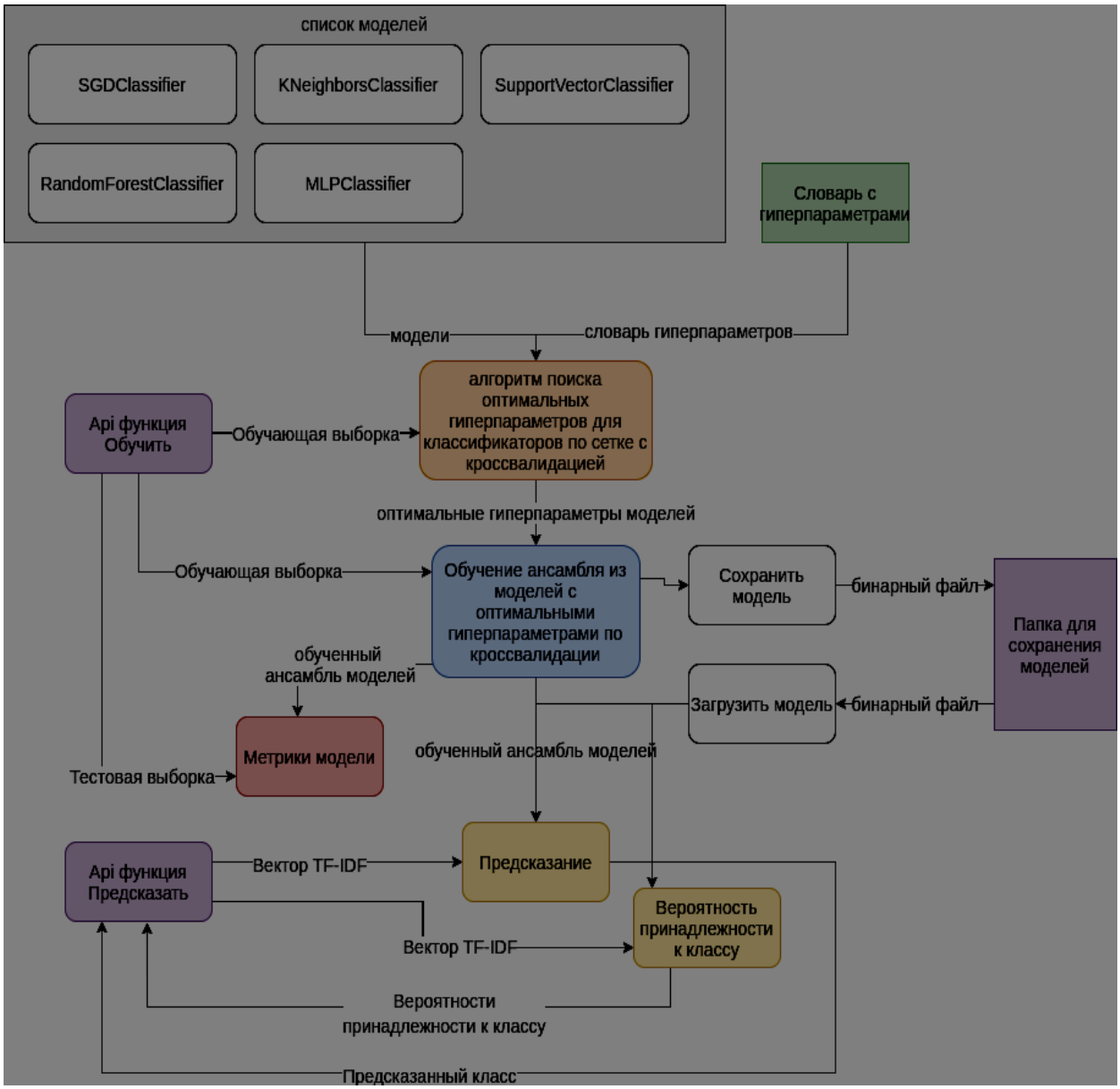


Рисунок 7 — Схема классификатора

4.3 Функции работы с базой данных результатов обучения

Основное назначение функций работы с базой данных результатов обучения — использование самых эффективных моделей классификации.

Функции работы с базой данных результатов обучения состоят из следующих блоков:

database — параметры связи с базами данных,

models — описание модели для хранения результатов обучения и создания таблиц в базе данных,

scheme — описание валидатора, для обращения к базе данных,

crud — основные функции, необходимые информационной системе для работы с базой данных.

Блок database отвечает за связь с базой данных, через ORM SQL-Alchemy. [28]

В блоке models представлена структура таблицы базы данных. Данные, направляемые в базу данных, связанны с конкретной итерацией обучения, хранятся в одной таблице, так как они плотно связанные (не могут быть использованы другими экземплярами обучения).

Поля таблицы хранения результатов обучения представлена в таблице 4:

Таблица 4 — Поля таблицы хранения результатов обучения

Поле	Тип	Описание
id	integer	Порядковый идентификатор
create_date	datetime	Дата, время тренировки модели
start_using_date	datetime	Дата, время начала эксплуатации
end_using_date	datetime	Дата,время вывода из эксплуатации
size	integer	Размер тренировочной выборки
vectorizer_name	string	Имя файла с сохраненным векторизатором
vectorizer_dict_size	integer	Размер словаря векторизатора
classifier_name	string	Имя файла с сохраненным классификатором
classifier_report	text	Отчет о классификации, собранный по тестовой выборке
classifier_matrix	text	Матрица ошибок, собранная по тестовой выборке
classifier_roc_auc	float	Площадь под кривой ошибок, взвешенная для классов
marked_to_delete	boolean	Метка на удаление
is_active	boolean	Метка текущего использования

В блоке scheme находятся такие же поля, что и в таблице хранения результатов. Его назначение валидация входящих данных на предмет поступления неверного типа данных, перед записью в базу данных.

В блоке crud реализованны основные функции, используемые через api в качестве внешних интерфейсов. В таблице 5 представлены функции созданные в crud для api:

Таблица 5 — функции в CRUD

Название функции	Описание
list_instances	Возвращает список всех результатов обучений
create_instance	Создает объект обучения, по заданным параметрам
retrive_instance_by_is_active	Возвращает запись с пометкой is_active
mark_to_delete	Ставит отметку на удаление - вносит изменения в поле marked_to_delete
delete_marked_instances	Удаляет все помеченные на удаление модели (файлы и запись в БД)
Activate_instance	Активирует указанную модель (есть параметр сравнения новой модели со старой до активации)

4.4 Интерфейсы RESTful веб-API

RESTful веб-API интерфейсы необходимы для работы внешних систем с системой машинного обучения. В качестве фреймоворка для реализации интерфейсов использован FastApi. [29]

На данный момент реализованы:

1) интерфейс для классификации. Загружает текущую активную модель, обрабатывает входящий текст, выполняет векторизаторизацию текста и выполняет классификацию обученной моделью. В итоге возвращается метка класса,

2) интерфейс для обучения. На вход подается список из текстов и их реальных классов. Текст проходит предобработку, векторизаторизацию применяется обучение модели и сама модель сохраняются в выделенной папке с уникальным именем, метрики обучения и данные об обучении добавляются в базу новой записью. Далее сравниваем старую и новую модели по параметрам размеры обучающей выборки и roc_auc (площадь под кривой ошибок, взвешенная для классов), если в результате сравнения новая модель окажется лучше старой, то активируется новая модель,

3) запрос параметров обо всех обучениях. В виде списка выводятся все записи из таблицы хранения результатов обучения,

4) запрос параметров одного обучения. Выводится выбранная запись из таблицы хранения результатов обучения,

5) ручная активация модели по идентификатору. Снимается метка активности старого обучения, выставляется дата окончания использования и активируется выбранная запись обучения, выставляется дата начала использования и очищается дата окончания использования (если заполнена), снимается метка на удаление (если такая была),

6) пометка на удаление обучения по идентификатору. Устанавливаем метку на удаление. Если выбранное обучение уже активировано, то возвращается исключение,

7) удаление всех помеченных на удаление обучений. Удаляются все записи и файлы связанные с помеченными обучениями.

Данные на входе и выходе интерфейсов, наглядно будет это оформить в таблице 1 и примеры этих данных в таблице 2 в приложении Б.

4.5 Используемые решения

В качестве языка программирования выбран Python, так как там представлено большое количество алгоритмов для машинного обучения.

Библиотека Pandas использовалась для работы с табличным представлением размеченной выборки.

Для лемматизации использован MyStem - разработка компании Yandex.

В качестве основной библиотеки для машинного обучения выбран Scikit-Learn, в ней представлены основные нужные мне алгоритмы для машинного обучения, ансамблирования, разбиения выборок, векторизации, измерения метрик классификации.

Api интерфейсы реализованы на основе фреймворка Fastapi. Этот фреймворк предоставляет по умолчанию множество полезных функций:

удобный модуль валидации Pydantic, с подробно описанными исключениями,

возможность использовать асинхронные запросы (пока не реализованы),

встроенный web-сервер Uvicorn,

пользовательский интерфейс Swagger для тестирования api интерфейсов.

В качестве ORM использован SQL-Alchemy, обладает обширной документацией, в том числе об использовании этого ORM в связке с Fastapi. Как и многие ORM позволяет подключаться к разным базам данных. В моем случае использована файловая база SQLite, так как размеры таблицы незначительны и увеличиваться будут медленно, около 12 записей в год.

ЗАКЛЮЧЕНИЕ

По результатам работы реализованы следующие пункты:

1) модуль предобработки текста из документов компании ООО «Газпром недра»,
2) классификация обработанных текстов по содержанию маркшейдерских данных с точностью выше 90%,
3) обучение моделей классификатора на новых данных,
4) автоматический выбор и использование лучшей модели по метрике roc_auc,
5) хранение характеристик всех обучений, для диагностики,
6) Restful api интерфейсы для использования обучения и классификации другими службами.

Работа открывает перспективы по классификации документов в системе электронного документооборота, не ограничиваясь лишь только маркшейдерскими данными.

Закончив свой проект, я могу сказать, что не все из того, что было задумано, получилось, например не удалось реализовать асинхронную обработку запросов к API. Для этого надо сформировать архитектуру взаимодействия с асинхронностью и реализовать блок подготовки обучающей выборки. В дальнейшем планируется создать блок подготовки обучающей выборки для автоматического дообучения модели. Помимо этого не успел реализовать дисстиляцию вектора TF-IDF для легкой обработки текста, при увеличении корпуса документов.

Если бы я начал работу заново, я бы рассмотрел варианты использования глубокого обучения и Word Embeddings векторизатора, но это требует тщательного изучения этого направления. А также изучить большее количество документов, формируемых в ООО «Газпром недра» и перестроить предобработку, для эффективного выделения тела документа и приложения не внося изменений в тело документа.

Я думаю, что я решил проблему своего проекта, так как интерфейсы для связи с классификатором, работают как и сама классификация и обучение.

Работа над проектом показала мне объем необходимых информационных систем для внедрения машинного обучения в инфраструктуру предприятия.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. https://nedra.gazprom.ru/about/
2. https://ru.wikipedia.org/wiki/геологоразведочные_работы
3. Закон "О недрах" Раздел III. Рациональное использование и охрана недр (ст.ст. 23 - 34) Статья 24. Основные требования по безопасному ведению работ, связанных с пользованием недрами
4. http://coalguide.ru/marsheyderskoe-upmeny/
5. https://ru.wikipedia.org/wiki/система_автоматизации_документооборота
6. https://ru.wikipedia.org/wiki/1С:Документооборот
7. Muhittin IŞIK, Hasan DAĞ, The impact of text preprocessing on the prediction of review ratings, Turkish Journal of Electrical Engineering & Computer Sciences, 15.11.2019
8. Kanerva, J., Ginter, F., and Salakoski, T. (2020),Universal Lemmatizer: A sequence-to-sequence model for lemmatizing Universal Dependencies treebanks, Natural Language Engineering, pp. 1.
9. Segalovich I. A Fast Morphological Algorithm with Unknown Word Guessing Induced by a Dictionary for a Web Search Engine. Proceedings of the International Conference on Machine Learning; Models, Technologies and Applications. Las Vegas, CSREA Press, 2003, pp. 273-280.
10. Gerard Salton and Christopher Buckley. Term-weighting Approachesin Automatic Text Retrieval. Information Processing & Management, 24(5):513–523, August 1988
11. T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In NIPS, 2013.
12. https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html
13. Qian Li, Hao Peng, Jianxin Li, Congyin Xia, Renyu Yang, Lichao Sun, Philip S. Yu, and Lifang He. 2020. A Survey on Text Classification: From Shallow to Deep Learning. ACM Comput. Surv. 37, 4, Article 35 (July 2020), p. 10
14. W. Dai, G. Xue, Q. Yang, and Y. Yu, “Transferring naive bayes classifiers for text classification,” in Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence, July 22-26, 2007, Vancouver, British Columbia, Canada, pp. 541, 2007.
15. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDClassifier.html
16. Scott Menard, Applied Logistic Regression Analysis. SAGE PUBLICATIONS. ISBN- 0-7619-2208-3, 2001
17. **L. Breiman. Random forests. Machine Learning, 45(1): 5–32, 2001.**
18. **T. M. Cover and P. E. Hart, “Nearest neighbor pattern classification,” IEEE Trans. Inf. Theory, vol. 13, no. 1, pp. 21–27, 1967**

19. **T. Joachims, “Text categorization with support vector machines: Learning with many relevant features,” in Machine Learning: ECML-98, 10th European Conference on Machine Learning, Chemnitz, Germany, April 21-23, 1998, Proceedings,pp. 137–142, 1998**

20. **Balabanov, T., Zankinski I., Kolev, K. (2018). Multilayer Perceptron Training Randomized by Second Instance of Multilayer Perceptron, Extended Abstracts of 13th Annual Meeting of the Bulgarian Section of SIAM, ISSN 1313-3357, Sofia, pp. 16-17**

21. **https://scikit-learn.org/stable/modules/grid_search.html#grid-search**

22. **Архипов В.А., «Сравнительный анализ метрик качества для моделей бинарной классификации на примере кредитного скоринга»,** Вестник Алтайской академии экономики и права. – 2019. – № 9 (часть 2) – С. 12-15

23. **https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_auc_score.html#sklearn.metrics.roc_auc_score**

24. **https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html?highlight=classification%20report#sklearn.metrics.classification_report**

25. **https://scikit-learn.org/stable/modules/model_evaluation.html#confusion-matrix**

26. **https://dyakonov.org/2019/04/19/ансамбли-в-машинном-обучении/comment-page-1**

27. https://scikit-learn.org/stable/modules/ensemble.html#voting-classifier

28. https://www.sqlalchemy.org/

29. https://fastapi.tiangolo.com/

Приложение А

Таблица 1 — *Пример предобработки текста (данные изменены)*

«Сырой» текст	Обработанный текст
ПАО «ГАЗПРОМ» ОБЩЕСТВО С ОГРАНИЧЕННОЙ ОТВЕТСТВЕННОСТЬЮ«ГАЗПРОМ ГЕОЛОГОРАЗВЕДКА» (ООО «Газпром геологоразведка») ЗАМЕСТИТЕЛЬГЕНЕРАЛЬНОГО ДИРЕКТОРА Генеральному директоруООО «Франт» А.В. Курагину Ул. Герцена, д. 70, г. Тюмень, Российская Федерация, 625000Тел.: (3452) 54-00-00,54-00-01, факс: (3452) 54-00-02E-mail: offic@ggr.gazprom.ru 0КП0 75782730, ОГРН 1042401809560, ИНН/КПП 2460066149Я20350001 на № от Уважаемый Анатолий Васильевич! Направляем Вам оригинал подписанного договора поставки геодезического оборудования от 01.02.2015 № Р100/15. Приложение:на 4 л. Заместитель генерального директора// по управлению персоналом/д2 и правовым вопросам/ А.Н. Балконский П.К. Безухов (3452) 54-00-04	направлять оригинал подписывать договор поставка геодезический оборудование приложение

Приложение Б

Таблица 2 — О*писание интерфейсов* RESTful веб-API *для машинного обучения*

Входящие данные (request)	Интерфейс	Исходящие данные (response)
текст	интерфейс для классификации	метка
[текст документа, реальный класс документа]	интерфейс для обучения	Новая или предыдущая действующая обученная модель, её запись из таблицы хранения результатов обучения
	запрос параметров обо всех обучениях	Все записи из таблицы хранения результатов обучения
Идентификатор обучения	Запрос параметров одного обучения по идентификатору	Выбранная запись из таблицы хранения результатов обучения
Идентификатор обучения	ручная активация модели по идентификатору	Активированная запись из таблицы хранения результатов обучения
Идентификатор обучения	пометка на удаление обучения по идентификатору	Помеченная на удаление запись из таблицы хранения результатов обучения
	удаление всех помеченных на удаление обучений	Все записи удаленные из таблицы хранения результатов обучения

Таблица 3 — П*ример данных для интерфейсов* RESTful веб-API *для машинного обучения*

Входящие данные (request)	Интерфейс	Исходящие данные (response)

{
"text": "Съешь еще этих мягких французских булок"
}
	интерфейс для классификации	{
		"predict_label": true
		}

Входящие данные (request)	Интерфейс	Исходящие данные (response)

[
{
"text": "Съешь еще этих мягких французских булок",
"label": true
},
{
"text": "Вновь он пережил банкротство, и теперь надо было подвести итог. ",
"label": true
}
]
	интерфейс для обучения	{
		"id": 5,
		"create_date": "2021-04-12T21:24:38.998251",
		"start_using_date": "2021-04-13T09:30:11.135856",
		"end_using_date": null,
		"size": 341,
		"vectorizer_name": "vector_2021_04_12_21_24.pkl",
		"vectorizer_dict_size": 8270,
		"classifier_name": "ensemble_2021_04_12_21_24.pkl",
		"classifier_report": " precision recall f1-score support\n\n False 0.96 0.93 0.94 54\n True 0.88 0.94 0.91 32\n\n accuracy 0.93 86\n macro avg 0.92 0.93 0.93 86\nweighted avg 0.93 0.93 0.93 86\n",
		"classifier_matrix": "[[50 4]\n [2 30]]",
		"classifier_roc_auc": 0.9317129629629629,
		"marked_to_delete": false,
		"is_active": true
		}

	запрос параметров обо всех обучениях	Список данных схожего формата как после обучения
Идентификатор обучения из таблицы	Запрос параметров одного обучения по идентификатору	Данные как после обучения
Идентификатор обучения из таблицы	ручная активация модели по идентификатору	Данные как после обучения
Идентификатор обучения из таблицы	пометка на удаление обучения по идентификатору	Данные как после обучения
	удаление всех помеченных на удаление обучений	Список данных схожего формата как после обучения