ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

###### Факультет экономических наук

##### ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

[**Статистический анализ данных и методы машинного обучения в задачах выявления страхового мошенничества**](https://lms.hse.ru/?ap&h_id=8EA86E75-5C9D-48B6-ADE6-E83C4A72B766)

по направлению подготовки Экономика

образовательная программа «Экономика и статистика»

|  |  |
| --- | --- |
|  | *Выполнил:* |
|  | студент группы БСТ194 |
|  | **Труфанов Дмитрий Михайлович**  Ф.И.О. |
|  | *Руководитель:* |
|  | к.т.н. доцент **Миронкина**  **Юлия Николаевна**  степень, звание, должность Ф.И.О. |

Москва 2023

**Оглавление**

[**Введение** 2](#_Toc134651114)

[**Глава 1. Практика применения методов машинного обучения, статистического анализа и искуственного интеллекта в задачах детекции страхового мошенничества** 7](#_Toc134651115)

[1.1. Терминология предметной области исследования, анализ мошенничества в страховых заявлениях и традиционные методы борьбы с ним 7](#_Toc134651116)

[1.2. Эффективные методы машинного обучения и статистического анализа для решения задачи обнаружения страхового мошенничества – теоретические основы инструментария исследования 11](#_Toc134651117)

[1.3. Научные исследования в сфере противодействия страховому мошенничеству 31](#_Toc134651118)

[**Глава 2. Методология обработки данных и построения моделей** 41](#_Toc134651119)

[2.1. Описание данных 41](#_Toc134651120)

[2. 2. Процесс обработки и преобразования данных, кластерный анализ 46](#_Toc134651121)

[2.3. Объяснение архитектуры свёрточной нейронной сети, используемой для обработки текстовых данных и настройка гиперпараметров методов машинного обучения 62](#_Toc134651122)

[**Глава 3. Результаты моделирования и сравнение разных моделей** 71](#_Toc134651123)

[3. 1. Объяснение оценочных показателей, используемых для измерения эффективности методов классификации 71](#_Toc134651124)

[3.2. Представление результатов использования различных методов машинного обучения на данных страхования жизни, сравнение эффективности моделей 74](#_Toc134651125)

[3.3. Представление результатов использования различных методов машинного обучения на данных страхования не-жизни, сравнение эффективности моделей 87](#_Toc134651126)

[**Заключение** 98](#_Toc134651127)

[**Список литературы** 101](#_Toc134651128)

[**Приложения** 105](#_Toc134651129)

**Введение**

Выявление страхового мошенничества является важнейшей задачей для страховых компаний. По данным Коалиции по борьбе со страховым мошенничеством [4], страховое мошенничество обходится компаниям в миллиарды долларов каждый год. Мошеннические претензии могут принимать различные формы, такие как инсценированные несчастные случаи (в том числе и ДТП), завышенные требования, ложное медицинское лечение. Страховое мошенничество не только влияет на финансовое состояние компаний, но и наносит ущерб экономике в целом, увеличивая страховые взносы и снижая доступность страховых услуг.

Для борьбы со страховым мошенничеством страховые компании применяют различные методы, включая ручное расследование (рассмотрение заявлений сотрудниками страховых компаний в частном порядке) и системы, основанные на правилах, разработанных в самих страховых компаниях. Однако эти методы имеют ограничения с точки зрения их точности, скорости и масштабируемости. В последние годы методы машинного обучения, статистический анализ и нейронные сети становятся более востребованными, так как с каждым годом качество их предсказаний в сфере детекции страхового мошенничества растёт. В этом исследовании мы стремимся изучить эффективность использования статистического анализа данных, методов машинного и глубинного обучения для выявления страхового мошенничества. Все это обуславливает несомненную актуальность и практическую значимость исследования.

*Цель работы*: разработать прикладную модель, которая будет основана на использовании свёрточных нейронных сетей, методов машинного обучения и классического статистического анализа данных и использовать как общепринятые числовые и категориальные переменные, так и текстовые данные для решения задач выявления страхового мошенничества.

*Объект исследования*: рынок и портфель страхования жизни и не-жизни в рамках личного страхования.

*Предметы исследования*: методы машинного обучения и статистического анализа, позволяющие выявлять мошеннические заявления.

*Задачи* исследования:

1. Проведение теоретического анализа научных и эмпирических исследований в сфере страхового мошенничества, выявление недостатков.
2. Использование кластерного анализа для выявления однородных групп в выборке данных.
3. Построение свёрточной нейронной сети на основе текстовых данных, которая будет решать задачу классификации.
4. Построение различных моделей машинного обучения, сравнение их качества в целом, выявление лучшей модели и выделение наиболее значимых признаков.
5. Сравнение результатов лучшей модели с использованием нейронной сети и без неё.

*Методологическую базу исследования составляют*: первичные методы обработки и анализа данных, методы преобразования текстовых данных в векторные представления, кластеризация данных с помощью метода k-means, свёрточная нейронная сеть, метод опорных векторов, логистическая регрессия, классификатор случайного леса (Random Forest Classifier) и градиентного бустинга (Gradient Boosting Classifier) и др. методы и различные статистические тесты.

В исследовании выдвинуты следующие *гипотезы*:

Использование текстовых данных, обработанных свёрточной нейронной сетью, улучшает качество классификации в задаче выявления страхового мошенничества;

Кластеризация данных может помочь другим методам машинного обучения справиться с задачей классификации;

Новые методы машинного обучения справляются с задачей детекции страхового мошенничества лучше, чем старые классические методы.

*Практическая значимость* данной работы – внесение вклада в разработку более точных и эффективных методов выявления страхового мошенничества. Используя статистический анализ данных и методы машинного обучения, страховые компании могут улучшить свои возможности по обнаружению мошенничества и снизить финансовые и экономические издержки, связанные с мошенничеством. Это же, в свою очередь, в долгосрочной перспективе может повлиять на общее снижение стоимости страховых услуг (брутто-премий), и, как следствие, на рост спроса на страховые услуги. Таким образом, увеличится общий объём рынка страхования как в России, так и во всём мире.

*Структура* выпускной квалификационной работы:

*В первой главе* будут определены термины, используемые в работе, объяснена логика основных методов машинного и глубинного обучения. Также будет представлен обзор литературы по предыдущим исследованиям в этой области.

*Во второй главе* мы разработаем методологию исследования, которая включает обработку данных (портфель личного страхования крупной российской страховой компании), в том числе и преобразование текстовых данных в векторы, изучим данные с помощью кластерного анализа, разделим выборку на тренировочную и тестовую подвыборки, а также на страхование жизни и не-жизни и остальное (страхование от несчастных случаев и болезней и медицинское страхование), построим и обучим свёрточную нейронную сеть, которая будет предсказывать вероятность того, что операция мошенническая, используя только текстовые данные из заявлений. Далее с помощью метрик классификации мы найдём оптимальный порог вероятности, после которого будем считать заявление мошенническим. Потом мы добавим результаты кластеризации, вероятностные и бинарные ответы нейронной сети как дополнительные признаки в исходные данные и обучим различные модели уже на всех имеющихся данных.

*В третьей главе* мы оценим полученные метрики различных методов выявления страхового мошенничества и сравним их эффективность. Также мы проверим, действительно ли текстовые данные и результаты кластерного анализа окажутся полезными в детекции страхового мошенничества. Наконец, мы обсудим последствия наших выводов для страховых компаний и порекомендуем лучшие практики использования методов машинного и глубинного обучения при выявлении страхового мошенничества.

*Информационная база исследования*. В нашем распоряжении портфель большой российской страховой компании по страхованию жизни, включая личное страховнаие (от несчастных случаев и болезней), общее количество страховых заявлений в котором около 59 тысяч.

Обработка данных осуществлялась с использованием языка программирования Python и встроенных в него библиотек.

В исследовании было использовано 24 различных источника (в основном иностранных), включая законодательные акты, учебные пособюия, научные статьи, интернет источники, в которых рассматривалась проблема страхового мошенничества и возможные пути её решения.

**Глава 1. Практика применения методов машинного обучения, статистического анализа и искуственного интеллекта в задачах детекции страхового мошенничества**

* 1. **Терминология предметной области исследования, анализ мошенничества в страховых заявлениях и традиционные методы борьбы с ним**

Прежде чем приступать к описанию результатов исследования, введем основные термины и определения изучаемой предметной области.

*Страховое мошенничество* — это обман страховых компаний с целью получения неправомерной выгоды. Это может включать в себя ложные заявления об ущербе, сфабрикованные или завышенные медицинские счета и многие другие мошеннические схемы.

*Страхователь* – это человек или организация, которая заключает договор страхования и оплачивает страховой взнос (страховую премию). Страхователь желает защитить свой имущественный интерес, например, застраховать свой автомобиль от угонов, страховать жизнь или здоровье.

*Страховщик* – это обычно юридическое лицо, которое предоставляет страховую защиту. Страховщик принимает на себя риски, связанные со страхованием, и выплачивает страховое возмещение в случае наступления страхового случая. Страховщик получает доход от страховых взносов (премий), диверсифицирует риски индивидуальных страхователей, управляет этими деньгами для того, чтобы заработать прибыль.

*Страховое мошенничество* может быть как со стороны покупателя (страхователя), так и со стороны продавца страхового контракта (страховщика). Страховое мошенничество со стороны продавца может включать продажу полисов от несуществующих компаний или неуплату премий по страховым случаям. Однако в данном исследовании мы будем рассматривать страховое мошенничество именно со стороны покупателя.

Страховое мошенничество является серьезной проблемой, которая имеет значительные экономические последствия. Мошенничество в общем понимании – это получение имущества другого человека при помощи обмана или злоупотребления доверием. В сфере страхования мошенничество возникает, когда совершается обман относительно страхового случая или размера страхового возмещения, которое должно быть выплачено в соответствии с законом или условиями договора страхования. Такое поведение является преступлением по статье 159.5 УК РФ [2]. Согласно данным судебного департамента при Верховном суде РФ, количество уголовных дел, зарегистрированных по ст. 159.5 УК РФ с 2014 года кратно возрастает каждый год (с 441 в 2014 году до 3224 в 2018 году). Важно отметить, что более половины случаев страхового мошенничества производится группой лиц по предварительному сговору, а более 20% - в особо крупном размере. Из всех дел по данной статье 45% было закрыто, из которых в 98% случаев причина закрытия была в том, что не удалось установить лицо, совершившее преступное деяние [3]. Соответственно, мошенники прекрасно осведомлены об уязвимостях страховых контрактов и законодательства РФ и знают способы их обхода [20]. По данным Федеральной службы страхового надзора объём мошенничества в сфере страхования оценивается в 15 миллиардов рублей [1]. Поэтому важно внедрять новые методы в задачах детекции страхового мошенничества на российском рынке.

По оценкам Национального бюро по борьбе со страховым мошенничеством в США (National Insurance Crime Bureau) и Федерального бюро расследований (Federal Bureau of Investigation) [23], страховое мошенничество приводит к потере до 40 миллиардов долларов ежегодно на американском рынке страхования не-жизни. В то же время, согласно исследованиям Коалиции против страхового мошенничества (Coalition Against Insurance Fraud), общая сумма страхового мошенничества на американском рынке оценивается в 308,6 миллиардов долларов [24].

Страховое мошенничество классифицируется на внутреннее и внешнее. Внутреннее страховое мошенничество происходит внутри страховой компании, когда сотрудники используют свой доступ и знания для совершения мошеннических действий. Внешнее страховое мошенничество, как правило, совершается сторонними лицами и направлено на получение незаконной выгоды от страховой компании. В данном исследовании будет затронут только второй тип страхового мошенничества.

Среди наиболее распространенных видов внешнего страхового мошенничества можно выделить: подделку документов, включая подделку документов об ущербе и медицинских счетов; завышение стоимости ущерба или затрат на восстановление имущества; симуляцию ущерба; мошенничество с медицинскими счетами, включая сфабрикованные медицинские услуги или медицинские услуги, которые не были оказаны; мошенничество с автомобильными страховками, включая ложные заявления об угонах, подделку документов, а также завышение цен на ремонт или замену частей; мошенничество в страховании жизни, включая подделку документов и мошенничество с медицинскими записями [25].

Страховое мошенничество не только вредит страховым компаниям, но и влияет на расходы на страхование, которые в конечном итоге перераспределяются на добросовестных потребителей в виде увеличения брутто-премий[[1]](#footnote-1). Борьба со страховым мошенничеством включает в себя использование различных технологий и аналитических методов для выявления мошеннических схем, сотрудничество с правоохранительными органами для уголовного преследования мошенников.

Традиционные методы борьбы со страховым мошенничеством представляют широкий спектр действий, направленных на предотвращение, выявление и расследование страховых мошеннических схем. Они включают в себя:

1. Аудиты и проверки. Страховые компании проводят регулярные аудиты и проверки для установления того, что клиенты предоставляют точные и правдивые данные о своих полисах. Это может включать в себя проверку медицинских записей и доказательств ущерба.
2. Сотрудничество с правоохранительными органами. Страховые компании работают с правоохранительными органами, чтобы помочь в расследовании и пресечении мошеннических схем. Они также могут предоставлять информацию о подозрительных действиях клиентов.
3. Обучение персонала. Сотрудники страховых компаний обучаются распознаванию признаков мошенничества и методам борьбы с ним. Это может включать в себя обучение по использованию новых технологий и обмену информацией о мошеннических схемах.
4. Системы отчетности. Страховые компании имеют системы отчетности, которые позволяют клиентам и сотрудникам сообщать о подозрительных действиях. Они также могут использовать анонимные системы отчетности для защиты жалобщиков.

Помимо этого, страховые компании также могут использовать экспертную оценку заявок вручную и выявлять мошеннические иски на основе заранее определенных правил, которые должны разрабатываться с помощью экспертов и постоянно обновляться (пример иных традиционных методов). Разработка этих правил должна проводиться с участием экспертов и регулярно обновляться. Однако эти подходы не гарантируют 100% точность и являются очень ресурсоёмкими с точки зрения временных затрат и, как следствие, денежных.

Традиционные методы борьбы со страховым мошенничеством являются важной частью работы страховых компаний. Однако они могут быть неэффективными в случаях, когда мошенники используют новые методы и технологии для совершения мошенничества. В связи с этим, методы машинного обучения стали новым инновационным решением в борьбе со страховым мошенничеством, которые могут значительно снизить затраты компании и уменьшить ее убытки за счет высокой точности моделей машинного обучения, статистического анализа и глубинного обучения.

**1.2. Эффективные методы машинного обучения и статистического анализа для решения задачи обнаружения страхового мошенничества – теоретические основы инструментария исследования**

Как было отмечено в предыдущей главе, часто обычных традиционных методов обнаружения страхового мошенничества оказывается недостаточно. Очевидно, что обычному рядовому сотруднику вручную невозможно проанализировать все эти заявления и по каждому сделать заключение. Как раз для этого в страховых компаниях используются разработанные системы правил, по которым автоматически выносится рекомендация. Однако данные системы правил часто имеют линейные зависимости и достаточно тривиальны. Именно для решения этой проблемы и для нахождения нетривиальных связей страховые компании начинают использовать методы машинного обучения для анализа их портфелей на наличие мошеннических заявлений. В данном параграфе будут рассмотрены методы, которые в последнее время всё чаще используются для решения задачи детекции страхового мошенничества.

Машинное обучение — это наука (и искусство) создания алгоритмов, которые могут обучаться на данных. Без явного программирования они способны делать предсказания или принимать решения с точностью, превосходящей традиционные ручные методы. Они используются для решения широкого спектра задач, таких как классификация изображений, распознавание речи, рекомендация продуктов, прогнозирование цен и т. д. В общем случае машинное обучение пытается найти общие закономерности в данных без явной их формализации [9]. Машинное обучение включает в себя множество различных алгоритмов, разработанных учёными, профессорами, энтузиастами или работниками IT компаний, которые позволяют компьютерам извлекать нетривиальные взаимосвязи из данных и делать прогнозы с достаточно высокой точностью. Статистический анализ данных является важным аспектом машинного обучения при выявлении мошенничества. Его использование помогает выявить закономерности и аномалии в данных, которые могут указывать на мошеннические операции. Статистические модели также могут быть использованы для прогнозирования вероятности мошенничества по каждой претензии на основе их исторических данных и других соответствующих факторов.

Перед применением любого метода машинного обучения нужно провести предобработку данных. Предобработка данных – это процесс очистки и преобразования данных перед их анализом. Цель предобработки данных заключается в том, чтобы получить набор данных, который можно использовать для построения модели, без потери важной информации.

Процесс предобработки данных может включать в себя следующие шаги: удаление дубликатов, обработку пропущенных значений, обработку выбросов, масштабирование данных, кодирование категориальных признаков, создание новых признаков.

Примером масштабирования данных может быть их стандартизация (при этом изначальное распределение данных остаётся неизменным). В данном методе выполняется преобразование данных, таким образом, чтобы среднее значение каждого признака стало равным 0, а стандартное отклонение равнялось 1. Формула данной функции выглядит так:

, (1.1)

где – новое значение после преобразования, – старое значение до преобразования, – среднее значение признака, – количество наблюдений (страховых заявлений), – стандартное квадратичное отклонение признака.

Стандартизация данных позволяет: уменьшить влияние выбросов в данных, что может привести к лучшей производительности алгоритмов машинного обучения; ускорить сходимость многих алгоритмов машинного обучения, например, градиентного спуска (работа данного алгоритма будет описана далее); обеспечить более точную оценку весов модели, улучшить интерпретируемость модели. Данный метод преобразования данных реализован в Python с помощью функции StandardScaler.

Также существует множество других способов шкалирования данных. Наиболее популярными из них являются: min-max scaling, которое приводит все значения к диапазону от 0 до 1; log scaling, которое используется для сокращения различий между значениями, имеющими большой разброс; quantile scaling, которое приводит данные к равномерному распределению с помощью ранговых значений. Выбор метода шкалирования зависит от конкретной задачи и особенностей данных.

Помимо методов шкалирования для числовых данных, как уже было написано ранее, используются методы обработки и перекодировки категориальных данных.

Одним из наиболее популярных является *One hot encoder*. One hot encoder (OHE) – это метод кодирования категориальных признаков, при котором каждый уникальный признак преобразуется в новый бинарный признак. Каждый из новых бинарных признаков соответствует одному значению исходного категориального признака. Например, если у нас есть категориальный признак «Цвет» со значениями «Красный», «Синий» и «Зеленый», то One hot encoder создаст три новых бинарных признака: «Цвет\_Красный», «Цвет\_Синий» и «Цвет\_Зеленый». Если значение исходного признака равно «Красный», то значение бинарного признака «Цвет\_Красный» будет равно 1, а значения двух других бинарных признаков будут равны 0. Таким образом, применение One hot encoder позволяет перевести категориальные признаки в числовые, что позволяет использовать их в алгоритмах машинного обучения, которые работают только с числовыми данными.

Ещё одним популярным кодировщиком является *Target Encoder*. Этот алгоритм кодирования категориальных признаков использует информацию о целевой переменной для присвоения числовых значений категориям. Он широко используется в задачах классификации и регрессии. В алгоритме Target Encoder каждой категории признака присваивается среднее значение целевой переменной в соответствующей категории. Причём среднее значение подходит как для задач регрессии, так и классификации, только в задаче классификации можно назвать полученное число долей единичных значений в одной категориальной группе. Например, если у нас есть категориальный признак «Тип заявителя» с двумя категориями «Выгодоприобретатель» и «Застрахованный», и мы хотим использовать его для прогнозирования бинарной целевой переменной, то для каждой категории мы будем вычислять долю мошеннических страховых заявлений с каждым типом заявителя и использовать её в качестве числового значения для этой категории. Таким образом, в данном алгоритме решается проблема многократного увеличения признаков, которая может возникнуть при использовании One hot encoder, ведь столбец останется тот же, просто вместо текстовых данных будут кодированные числовые. Однако при использовании алгоритма Target Encoding возникает проблема переобучения, когда значения признака становятся слишком зависимыми от конкретной выборки данных. Для борьбы с этим можно использовать методы борьбы с переобучением (будут обсуждаться далее).

После всех преобразований данных можно приступать к применению методов машинного обучения. Их можно разделить на две основные категории: обучение с учителем и и без учителя. При обучении с учителем алгоритм извлекает информацию из помеченных данных, где каждая точка данных связана с классом или меткой (как в нашем случае – мошенническая или не мошенническая). Алгоритм использует эти помеченные данные для построения модели, которая может предсказать класс новых, немаркированных данных. При обучении без учителя алгоритм извлекает информацию из немаркированных данных, где класс или метка неизвестны (например, задача кластеризации). Алгоритм пытается выявить закономерности или кластеры в данных, которые могут указывать на мошенничество.

Примером метода машинного обучения без учителя является кластеризация данных. Кластеризация данных – это задача разбиения множества объектов на группы (кластеры) таким образом, чтобы объекты внутри кластера были похожи друг на друга, а объекты из разных кластеров были различны. В задачах кластеризации не заданы заранее метки классов для объектов, а требуется найти их структуру на основе анализа сходства объектов между собой.

Одним из наиболее часто используемых в научных исследованиях алгоритмов кластеризации является k-means. Алгоритм k-means – это метод кластеризации, который позволяет разбить множество объектов на заранее заданное количество кластеров. Сама кластеризация – задача группировки объектов на подмножества (кластеры) на основе их сходства. Алгоритм k-means основан на расстоянии между объектами, которое вычисляется с помощью евклидовой метрики (формула 1.2).

, (1.2)

где – расстояние между объектами *i* и *j*, *m* – количество признаков (столбцов), – численное значение *k*-го признака у *i*-го объекта, – численное значение *k*-го признака у *j*-го объекта.

В контексте k-means, расстояние между точками используется для вычисления среднего значения точек в кластере и для определения ближайшего кластера для каждой точки. Кластеры образуются таким образом, чтобы минимизировать сумму квадратов расстояний между каждой точкой и центроидом[[2]](#footnote-2) (средним значением) ее кластера.

Первый шаг в алгоритме k-means – инициализация. Задаём количество кластеров (k) и случайным образом выбираем k центроидов – это будут начальные центры кластеров. Второй шаг – определение принадлежности. Для каждого объекта выборки определяем, к какому кластеру он ближе всего. Для этого вычисляем расстояние между объектом и каждым центроидом, затем присваиваем объект к кластеру с ближайшим центроидом. Третий шаг –

пересчёт центроидов. Пересчитываем координаты центроидов как среднее арифметическое координат объектов в каждом кластере. Далее шаги 2 и 3 повторяются до тех пор, пока кластеры не стабилизируются (центроиды перестанут изменять свои координаты) или не будет достигнуто максимальное количество итераций. В результате получаем разбиение исходных данных на k кластеров, где каждый кластер характеризуется своим центроидом.

Существует множество методов кластеризации данных, которые могут отличаться по принципу работы и результатам. Некоторые из них основываются на расстояниях между объектами, например, методы k-means, DBSCAN или иерархическая кластеризация. Другие методы, например, алгоритмы спектральной кластеризации, основаны на матрицах сходства между объектами.

В задаче кластеризации проблемой является оценка качества алгоритма, так как это обучение без учителя, невозможно измерить ошибку алгоритма. Были разработаны некоторые метрики, которые позволяют оценивать качество кластеризации. Четыре наиболее популярных из них:

* SSE (Sum of Squared Errors) – это сумма квадратов расстояний между каждым объектом кластера и центроидом кластера (чем меньше значение SSE, тем лучше кластеризация);
* Silhouette – это метрика, которая используется для оценки, насколько объект хорошо сгруппирован в своем кластере и насколько он отделен от соседних кластеров (значение Silhouette лежит в диапазоне от -1 до 1, где 1 – это идеальная кластеризация, а -1 – это плохая кластеризация);
* Davies-Bouldin Index – это метрика, которая используется для оценки различных кластеров внутри данных, измеряя среднее расстояние между каждым кластером и его самым близким соседом (чем меньше значение Davies-Bouldin Index, тем лучше кластеризация);
* Calinski-Harabasz Index – это метрика, которая измеряет отношение между суммой квадратов расстояний между центроидами кластеров и суммой квадратов расстояний между объектами и центроидами (чем выше значение Calinski-Harabasz Index, тем лучше кластеризация).

Далее стоит разобрать методы машинного обучения с учителем. Среди них можно выделить:

1. Линейные модели — алгоритмы, основанные на линейной комбинации признаков объекта. Например, логистическая регрессия, метод опорных векторов, линейный дискриминантный анализ.
2. Решающие деревья — алгоритмы, основанные на построении древовидной структуры, которая позволяет разбивать набор данных на подмножества с разными значениями целевой переменной.
3. Ансамблевые методы — алгоритмы, основанные на комбинации нескольких моделей машинного обучения для достижения лучшей точности предсказаний. Среди таких методов можно выделить случайный лес и градиентный бустинг (будут описаны далее).
4. Нейронные сети — алгоритмы, которые используют искусственные нейроны и слои для обработки данных. Такие модели широко применяются в задачах распознавания образов, обработки естественного языка и компьютерного зрения.

В решении задачи классификации могут оказаться эффективными как классические статистические методы (логистическая регрессия, метод опорных векторов, линейный дискриминантный анализ), так и современные методы машинного обучения (случайный лес, градиентный бустинг). Перед описанием данных методов стоит дать определение методу градиентного спуска, который используется во многих моделях машинного обучения с учителем.

*Метод градиентного спуска* – это один из базовых алгоритмов оптимизации, который применяется для минимизации функции потерь[[3]](#footnote-3). Он широко используется для обучения моделей машинного обучения, включая нейронные сети. В основе метода градиентного спуска лежит идея поиска минимума функции потерь путем итеративного шага в направлении, обратном градиенту функции (отрицательное значение градиента). Градиент - это вектор частных производных функции по каждой переменной, взятых вместе. Если функция представляет ошибку модели, то минимизация этой функции означает поиск наилучших параметров модели, которые минимизируют ошибку. Ниже представлена формула градиентного спуска.

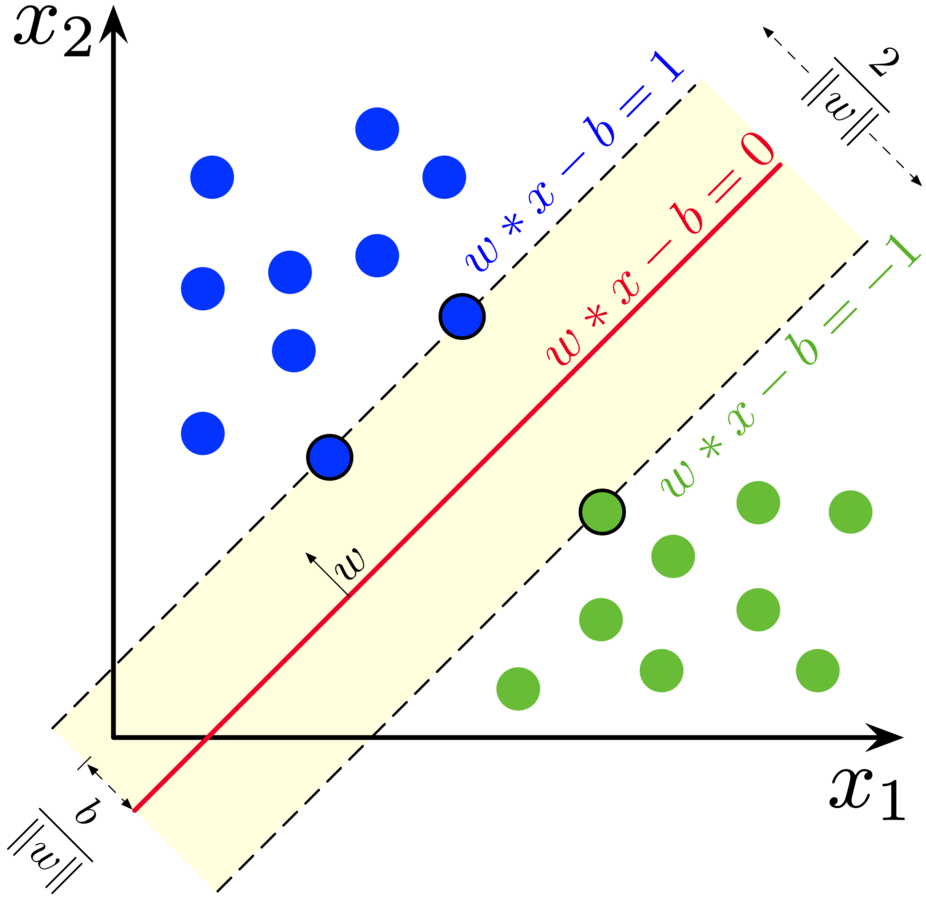
, (1.3)

где – вектор весов на t итерации, – размер шага градиентного спуска, – градиент функции потерь в предыдущей итерации.

При этом обычно градиентный спуск останавливается, когда выполняется одно из этих условий: или , где – какое-то малое значение, являющееся гиперпараметром.

*Логистическая регрессия* – это метод классификации, который использует логистическую функцию для оценки вероятности принадлежности объекта к определенному классу. Он был предложен в 1958 году Джозефом Берксоном и Мортоном Коулом. Для обучения модели логистической регрессии используется метод максимального правдоподобия. Метод максимального правдоподобия (Maximum Likelihood Estimation, MLE) – это статистический метод оценки параметров вероятностного распределения, основанный на максимизации функции правдоподобия. Идея метода максимального правдоподобия заключается в том, что мы хотим найти те значения параметров, при которых вероятность получить наблюдаемые данные максимальна. Другими словами, мы ищем те параметры, которые делают наши данные наиболее вероятными. Допустим, у нас есть выборка данных , и мы хотим оценить параметр θ нашей модели. Пусть - вероятность наблюдения значения x при параметре θ. Тогда функция правдоподобия для выборки данных будет выглядеть следующим образом: . Теперь мы хотим максимизировать эту функцию по параметру θ. Вместо этого часто максимизируют логарифм функции правдоподобия, так как при это упрощает вычисления: log L(θ|X) = . Для нахождения максимального правдоподобия мы можем использовать различные методы оптимизации, например, градиентный спуск или метод Ньютона-Рафсона[[4]](#footnote-4). Оценки параметров, полученные методом максимального правдоподобия, являются асимптотически эффективными, то есть они имеют наименьшую дисперсию среди всех несмещенных оценок параметров. Логистическая регрессия использует логистическую функцию (сигмоиду) для оценки вероятности принадлежности объекта к классу 1: . Параметры модели (вектор весов) настраиваются с использованием градиентного спуска, который позволяет найти оптимальные значения параметров модели, минимизирующие ошибку классификации.

*Метод опорных векторов (SVM)* был предложен Владимиром Вапником и Алексеем Червоненкисом в 1963 году. SVM (*Support Vector Machine*) – это алгоритм машинного обучения, используемый в задачах классификации. SVM строит гиперплоскость или набор гиперплоскостей в многомерном пространстве, которые максимально разделяют два класса данных. Идея заключается в том, чтобы найти гиперплоскость, которая максимально разделяет данные на два класса. Гиперплоскость – это (d-1)-мерное подпространство d-мерного пространства, которое разделяет данные. В случае двух классов гиперплоскость - это просто линия, разделяющая точки одного класса от точек другого. SVM находит оптимальную гиперплоскость, которая максимально разделяет два класса данных. Эта гиперплоскость находится путем решения оптимизационной задачи, в которой минимизируется величина ошибки и максимизируется отступ между двумя классами. Отступ - это расстояние от ближайшей точки к гиперплоскости до гиперплоскости. В SVM классификация новых данных осуществляется путем определения, по какую сторону гиперплоскости находится каждая новая точка, и присвоения ей соответствующего класса. При этом, чем дальше окажется точка от разделяющей гиперплоскости, тем больше модель уверена в классификации данного объекта. Существует несколько типов SVM, включая линейный SVM и нелинейный SVM. Линейный SVM используется для линейно разделимых данных, тогда как нелинейный SVM используется для данных, которые не могут быть разделены линейно и требуют преобразования признаков в пространство более высокой размерности.



*Рис. 1. Иллюстрация работы метода опорных векторов на линейно разделимых данных*

*Нелинейный SVM* – это расширение линейного SVM для задач, где граница принятия решения между двумя классами не может быть описана линейным разделителем. Для работы с нелинейными границами между классами, необходимо преобразовать исходное пространство данных в пространство бо́льшей размерности, где граница между классами станет линейной. Это делается с помощью ядерного метода, который позволяет вычислять скалярное произведение в преобразованном пространстве, не выполняя явного преобразования. Ядерная функция определяет, как данные должны быть преобразованы в новое пространство. Одной из самых распространенных ядерных функций является RBF-ядро (Radial Basis Function), которое определяется следующим образом: где – гиперпараметр, отвечающий за ширину распределения RBF-ядра. Выше на рис. 1 предсатвлен график работы SVM на линейно разделимой выборке.

*Метод LDA (Linear Discriminant Analysis)* был изобретен Рональдом А. Фишером в 1936 году. Этот метод используется для поиска линейной комбинации признаков, которая максимально разделяет данные на два или более класса. Этот метод решает задачу нахождения оптимального проекционного вектора, максимизирующего отношение межклассовой дисперсии к внутриклассовой дисперсии. Для этого сначала вычисляются средние значения признаков для каждого класса, а затем рассчитываются матрицы рассеяния внутриклассовой и межклассовой дисперсии. Они являются матрицами ковариации и могут быть использованы для оценки внутриклассовой и межклассовой дисперсии. Затем происходит вычисление обратной матрицы внутриклассовой дисперсии и умножение ее на матрицу межклассовой дисперсии. Внутриклассовая дисперсия отражает разброс значений переменных внутри каждого класса. Она вычисляется как сумма матриц ковариации каждого класса, умноженных на соответствующее количество наблюдений в каждом классе. Эта мера показывает, насколько хорошо разделены классы внутри себя. Межклассовая дисперсия отражает разброс средних значений переменных между классами. Она вычисляется как сумма матриц отклонений каждого класса от общего среднего значения, умноженных на соответствующее количество наблюдений в каждом классе. Эта мера показывает, насколько хорошо разделены классы между собой. Полученный результат является оптимальным проекционным вектором, который можно использовать для классификации новых данных. В линейном дискриминантном анализе используется отношение межклассовой дисперсии к внутриклассовой дисперсии, называемое критерием Фишера. Он показывает, насколько хорошо классы разделены друг от друга и может быть использован для выбора оптимального количества дискриминантных функций.

Ансамблевыми методами машинного обучения принято считать случайный лес и градиентный бустинг.

*Случайный лес (Random Forest Classifier, RFC)* был предложен Лео Брейманом и Адель Катлер в 2001 году. Они предложили использовать ансамбль деревьев решений, где каждое дерево строится на случайной подвыборке данных и случайном подмножестве признаков, чтобы уменьшить переобучение и повысить устойчивость модели.

Алгоритм RFC состоит из нескольких шагов. На первом выбирается случайное подмножество признаков из общего количества признаков (опционально можно выбрать случайное число наблюдений). На втором шаге для выбранного подмножества признаков строится дерево решений. Далее шаги 1 и 2 повторяются много раз, чтобы создать большое количество деревьев решений.

Классификация нового объекта осуществляется путем агрегации ответов от каждого дерева. Важно отметить, что каждое дерево может быть построено не только с использованием случайного подмножества признаков, а также со случайным подмножеством данных. Это делает RFC устойчивым к переобучению, так как каждое дерево использует только часть доступных данных и признаков. Кроме того, RFC использует метод бутстрапа для получения разных образцов данных с заменой из исходного набора данных (извлечения с повторением). Это позволяет увеличить количество обучающих примеров и уменьшить смещение модели.

Для оценки качества разбиения в алгоритмах решающих деревьев используются различные функции. Наиболее популярными являются критерий Джини и энтропия. Gini и Entropy — это критерии разделения (splitting criteria), используемые в алгоритме построения деревьев решений. Критерий Gini использует меру неопределенности Gini impurity, которая измеряет вероятность неправильной классификации элемента данных, если он случайно относится к подгруппе. Чем меньше Gini impurity, тем лучше разбиение. Он определяется следующим образом: где – это доля объектов класса *i* в одном узле, а *J* – количество классов. Критерий Entropy использует меру энтропии, которая измеряет степень неопределенности в наборе данных. Он определяется следующим образом: . Критерий Entropy более вычислительно затратен, чем критерий Gini, потому что он использует логарифмы. Однако он может привести к более равномерному разбиению, когда классы в узлах имеют примерно равные доли. Джини обычно работает лучше для задач классификации, когда классы сбалансированы, в то время как энтропия может работать лучше в случаях, когда классы несбалансированы (подходит для детекции страхового мошенничества).

В RFC используется мера важности признаков для определения того, какие признаки наиболее полезны для классификации. Она рассчитывается на основе того, как часто данный признак используется для разделения данных в деревьях. Данную меру мы будем рассматривать в следующей главе, оценивая влияние двух новых переменных на качество предсказаний классификатора.

В целом, Random forest classifier является мощным и эффективным алгоритмом для классификации данных, который использует принципы ансамбля решающих деревьев, чтобы повысить точность и устойчивость модели.

*Градиентный бустинг* был предложен Йоавом Фройндом и Робертом Шапиро в 1997 году. Они предложили использовать градиентный спуск для обучения ансамбля слабых моделей (например, деревьев решений), где каждое новое дерево строится с учетом ошибок предыдущего ансамбля. Gradient boosting classifier - это метод машинного обучения, который позволяет построить предсказательную модель для задачи классификации. Он основан на идее комбинирования нескольких слабых моделей для создания более сильной и точной модели.

Работа Gradient boosting classifier происходит в несколько этапов. Первый этап – инициализация. В начале работы создается модель, которая будет использоваться для классификации данных. Обычно это простая модель, такая как дерево решений с небольшой глубиной. Следующие этап – обучение. На этом этапе происходит обучение первой модели на тренировочных данных. Для этого строится дерево решений, которое пытается разделить данные на классы. Однако это дерево может быть неполным и иметь ошибки в классификации. Это исходное дерево называется базовой моделью. Третий этап – вычисление ошибок и остатков. На этом этапе вычисляются ошибки базовой модели для каждого объекта. Эти ошибки называются остатками и показывают, насколько базовая модель ошибается в предсказании. На четвёртом этапе происходит создание новой модели. На этом этапе создается новая модель, которая будет использоваться для корректировки ошибок базовой модели. Новая модель строится таким образом, чтобы она предсказывала остатки, которые были вычислены на предыдущем этапе. Следующим шагом является обучение новой модели. На этом этапе новая модель обучается на тренировочных данных, чтобы предсказывать остатки. Обычно используется градиентный спуск для нахождения оптимальных весов модели. Шестой этап – комбинирование моделей (ансамбль). На этом этапе базовая модель и новая модель комбинируются для улучшения качества предсказания. Для этого просто складываются предсказания базовой модели и новой модели. Последний этап – множественные итерации. Предыдущие шаги повторяются до тех пор, пока не будет достигнуто определенное количество итераций или пока качество модели не перестанет улучшаться.

Каждая новая модель учитывает ошибки базовой модели и корректирует их, что позволяет создавать более точную модель для классификации данных. Результатом работы Gradient boosting classifier является сильная модель, которая может использоваться для предсказания классов для новых данных.

В алгоритме Gradient Boosting Classifier тоже используются деревья решений в качестве базовых моделей. Алгоритм начинается с создания простого дерева. Затем, на каждой следующей итерации, строится новое дерево, которое пытается исправить ошибки предыдущего дерева. Ошибки измеряются с помощью функции потерь, которая определяет, насколько далеко прогнозы отличаются от реальных значений. После того, как дерево построено, оно используется для предсказания классов на новых данных. Все деревья в алгоритме Gradient Boosting Classifier обрабатываются последовательно. При расчете остатков используется градиент функции потерь, например, для задачи классификации это может быть бинарная кросс энтропия. В нашем случае используется функция deviance, функция потерь для бинарной классификации в Gradient Boosting Classifier. Она выражается следующей формулой: Эта функция потерь основана на логарифмической функции правдоподобия и используется для минимизации ошибки модели в процессе обучения. Она позволяет оценивать качество модели на обучающих данных, а также выбирать наилучшие гиперпараметры. Градиент функции потерь по предсказанным значениям вычисляется для каждого объекта в обучающем наборе, и затем строится новое дерево, которое пытается предсказать эти градиенты. Это делается путем разбиения объектов на подмножества в зависимости от значений признаков, чтобы минимизировать градиент функции потерь на каждом листе дерева.

Таким образом, градиентный бустинг использует градиентный спуск, чтобы на каждом шаге находить оптимальные параметры нового дерева, которые минимизируют функцию потерь на обучающем наборе. Каждое новое дерево добавляется к ансамблю, уменьшая остаточную ошибку и улучшая качество прогнозов. Конечный результат – это сумма предсказаний каждого дерева, взвешенная на коэффициент скорости обучения (learning rate), который определяет вклад каждого дерева в окончательное предсказание. На рис. 2 представлена схема работы градиентного бустинга.

Изображение выглядит как диаграмма, текст, линия, Прямоугольник

Автоматически созданное описание

*Рис. 2. Схематичный график работы градиентного бустинга по итерациям*

Помимо описанных выше методов машинного обучения, в последнее время активно становятся популярными *методы глубинного обучения (нейронные сети).* Для классификации текстов часто используют следующие типы нейронных сетей:

1. *Рекуррентные нейронные сети (RNN)*. Эти сети способны учитывать контекст и последовательность слов в тексте, что делает их хорошим выбором для задач классификации текстов, особенно для анализа последовательности слов.
2. *Сверточные нейронные сети (CNN).* Эти сети хорошо работают со входными данными в формате матрицы, что делает их хорошим выбором для задач, связанных с обработкой изображений и текстов.
3. *Комбинированные модели RNN и CNN.* Эти сети комбинируют свойства RNN и CNN, что позволяет достигнуть хороших результатов на задачах классификации текстов. Например, можно использовать сверточные слои для выделения признаков из текста, а затем передавать их на вход RNN-сети для учета контекста.
4. *Transformer-сети.* Эти сети, такие как BERT (Bidirectional Encoder Representations from Transformers), позволяют моделировать контекст и взаимосвязи между словами в тексте, используя механизмы внимания.

Наиболее популярными являются именно модели CNN. Свёрточные нейронные сети (Convolutional Neural Networks, CNN) – это тип нейронных сетей, которые могут использоваться для обработки и анализа изображений, а также для обработки текстовых данных. В обработке текстовых данных сверточные сети используются для классификации текстов по категориям или меткам. Основная идея сверточной нейронной сети заключается в том, что она извлекает признаки из входных данных, используя фильтры или ядра, которые применяются к различным участкам входных данных. Фильтры работают как окна, скользящие по входным данным и применяющиеся к каждой части текста. При этом, фильтры позволяют выделять различные признаки в текстах, например, наличие определенных слов или фраз в тексте, их частоту и распределение.

Основные компоненты свёрточной нейронной сети: Сверточный слой (Convolutional Layer); слой подвыборки (Pooling Layer); полносвязный слой (Fully-Connected Layer). Опционально есть возможность добавлять другие слои.

Свёрточные сети могут содержать один или несколько слоев свертки, каждый из которых использует несколько фильтров для извлечения признаков из входных данных. В результате, сверточная сеть создает многомерный тензор[[5]](#footnote-5), где каждая размерность представляет отдельный признак.

После этого данные проходят через слой пулинга, который уменьшает размерность многомерного тензора и удаляет избыточную информацию. Обычно в качестве функции пулинга используется функция MaxPooling, которая выбирает максимальное значение в каждой подгруппе. То есть в каждом фильтре выбирается максимальное значение, слово, имеющее самое большее влияние (в случае изображений – самый яркий пиксель, который попал в фильтр).

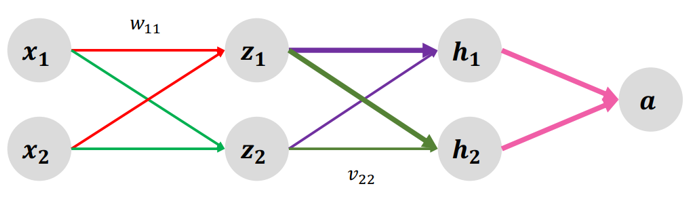
После слоя пулинга данные проходят через полносвязный слой (Dense Layer), который преобразует многомерный тензор в одномерный вектор, готовый к классификации. Полносвязный слой может содержать несколько слоев, называемых скрытыми слоями, которые позволяют модели извлекать более высокоуровневые признаки из данных.

В конечном итоге, модель выводит предсказания в виде вероятности принадлежности к определенной категории, с использованием функции активации[[6]](#footnote-6).

Основной процесс обучения свёрточной нейронной сети заключается в настройке весов фильтров в сверточном слое. Это достигается путем минимизации функции потерь, которая измеряет разницу между предсказанной и истинной метками. Обучение проводится с помощью методов обратного распространения ошибки и градиентного спуска, которые находят оптимальные значения весов.

Суть метода обратного распространения ошибки заключается в том, чтобы распространить ошибку (разность между фактическим и желаемым выходом) от выходного слоя к скрытым слоям, а затем к входному слою, и затем использовать эту информацию для корректировки весов в обратном направлении. Таким образом, веса на каждом слое будут корректироваться в соответствии с вкладом каждого нейрона в ошибку.

Алгоритм Backpropagation можно разделить на несколько шагов. Первым шагом является «прямое распространение». На вход подаются данные, которые проходят через каждый слой сети и вычисляются выходные значения на каждом слое. Эти значения передаются дальше до последнего (выходного) слоя, который выдает окончательный результат. Вторым шагом идёт расчёт ошибки. Сравнивается полученный выход с желаемым, вычисляется ошибка на выходном слое. Эта ошибка затем распространяется назад через скрытые слои, где она распределяется между нейронами на основе их вклада в выход. Третьим шагом следует обновление весов (коэффициентов). Веса на каждом слое корректируются в соответствии с вкладом каждого нейрона в ошибку. Это происходит путем вычисления градиента функции потерь по отношению к каждому весу и использования этого градиента для обновления весов на каждом слое. Далее эти шаги повторяются до тех пор, пока модель не достигнет заданного уровня точности или не будет произведено достаточное количество итераций.



*Рис. 3. Схема простой полносвязной нейронной сети с двумерным входом и одномерным выходом*

В целом, метод Backpropagation позволяет эффективно обучать многослойные нейронные сети для решения различных задач классификации и регрессии. Он может быть использован в сочетании с различными архитектурами сетей и функциями активации. Выше на рис. 3 схематично представлена работа алгоритма обратного распространения ошибки в полносвязной нейронной сети. В упрощённой схеме, представленной на рисунке, производная выхода модели по весу будет выражена так:

*.*

**1.3. Научные исследования в сфере противодействия страховому мошенничеству**

Задача обнаружения страхового мошенничества является актуальной и важной для страховых компаний, поскольку мошеннические действия могут привести к серьезным экономическим потерям. В связи с этим, в последние годы наблюдается рост интереса к применению методов машинного обучения для решения данной задачи. В данном обзоре мы рассмотрим существующие исследования и подходы, используемые для детекции страхового мошенничества с помощью машинного обучения.

Первая статья, которая будет нами рассмотрена – «*Leveraging deep learning with LDA-based text analytics to detect automobile insurance*», авторы Wei Xu и Yibo Wang. Эта статья описывает новую методологию глубинного обучения для обнаружения мошенничества в автомобильном страховании. Авторы разработали аналитику текста на основе LDA[[7]](#footnote-7) для извлечения признаков из текстового описания аварий. Она используется для определения скрытых тематик в текстовых документов. Суть LDA заключается в том, что каждый документ может быть представлен как смесь нескольких тем, а каждая тема может быть представлена как распределение вероятностей на множестве слов. При этом, каждое слово в документе связано с какой-то темой, и вероятность появления слова в документе зависит от соответствующей ему темы. Модель LDA строится на основе генеративной вероятностной модели. В результате обучения модели LDA на большой коллекции документов можно получить распределение вероятностей на темы, а также распределение вероятностей на слова для каждой темы. Эти результаты могут быть использованы для различных задач, таких как классификация документов, поиск похожих документов, анализ тональности текстов и т.д. Они исследовали текстовые и традиционные числовые признаки для обнаружения мошеннических заявок. В предложенном ими методе сначала используется LDA для извлечения текстовых признаков, скрытых в текстовых описаниях аварий, появляющихся в заявках, а затем глубинные нейронные сети обучаются на данных, которые включают текстовые и традиционные числовые признаки для обнаружения мошеннических заявок.

На основе реального набора данных их экспериментальные результаты показывают, что предложенная на основе аналитики текста модель превосходит традиционные. Кроме того, экспериментальные результаты показывают, что глубокие нейронные сети превосходят широко используемые модели машинного обучения, такие как случайный лес (Random Forest) и метод опорных векторов (SVM) [16].

Следующей статьёй, в которой тоже решалась проблема мошенничества в автомобильном страховании, является работа авторов Yanjun Zhou, Huosong Xia и Zuopeng Zhang на тему «*Auto insurance fraud identification based on a CNN-LSTM fusion deep learning model*». Эта статья предлагает использование глубинных методов машинного обучения для обнаружения мошенничества в автомобильном страховании, в частности, сочетая сверточную нейронную сеть (CNN), нейронную сеть краткосрочной и долгосрочной памяти (LSTM[[8]](#footnote-8)) и глубинную нейронную сеть (DNN[[9]](#footnote-9)). Традиционные методы обнаружения мошенничества в автомобильном страховании сильно зависят от инженерии признаков и знаний в области, что затрудняет точное и эффективное обнаружение мошенничества при большом количестве данных о заявках и высокой размерности данных. Нейронные сети имеют сильные обобщающие способности и могут автоматически совершить извлечение признаков. Предложенный метод может извлекать более абстрактные неинтерпертируемые признаки и помогает избежать сложного процесса извлечения признаков, который сильно зависит от экспертных оценок в традиционных алгоритмах машинного обучения. Эксперименты показывают, что разработанный метод может эффективно повысить точность обнаружения мошенничества в автомобильном страховании [19].

Статья «*Insurance fraud detection: Evidence from artificial intelligence and machine learning*», написанная авторами Faheem Aslam, Ahmed Imran Hunjra, Zied Ftiti, Wael Louhichi и Tahira Shams тоже предполагает использование предсказательных моделей для обнаружения мошенничества в автомобильном страховании. Они проводят отбор признаков в общедоступном наборе данных по автомобильному страхованию и выявляют наиболее значимые признаки с помощью алгоритма Boruta[[10]](#footnote-10). Основная идея Boruta заключается в том, что для всех признаков в исходном наборе данных создаются случайные копии (называемые теневыми признаками) и обучаются классификаторы на основе этого расширенного набора данных. Чтобы понять важность признака, производится сравнение его со всеми сгенерированными теневыми признаками. Сохраняются только те признаки, которые статистически более важны, чем синтезированные признаки, так как они в большей степени способствуют улучшению работы модели. Авторы применяют три предсказательные модели (логистическая регрессия, метод опорных векторов и наивный байес[[11]](#footnote-11)) для разработки механизма обнаружения мошенничества. Они вычисляют шесть метрик из полученных результатов и сравнивают их с традиционными методами обнаружения мошенничества. Эксперименты показывают, что алгоритмы машинного обучения улучшают предсказательную силу обнаружения мошенничества. Метод опорных векторов превосходит остальные алгоритмы по доле правильных ответов (Accuracy[[12]](#footnote-12)), а логистическая регрессия достигает наивысшего значения F1 меры[[13]](#footnote-13).

По мнению авторов, традиционные методы обнаружения мошенничества с автомобильным страхованием сильно зависят от инженерии признаков и знаний в области, что затрудняет точное и эффективное обнаружение мошенничества при большом количестве данных о заявках и высокой размерности данных. Модели глубинного обучения имеют сильные обобщающие способности и могут автоматически завершить извлечение признаков [5].

В статье «*The value of cross-data set analysis for automobile insurance fraud detection*» автор Meryem Yankol-Schalck представляет методику обнаружения мошенничества в автостраховании, используя анализ нескольких наборов данных. Автор исследует эффективность использования алгоритмов машинного обучения, таких как деревья решений, случайный лес, деревья решений и метод опорных векторов (SVM), в сочетании с кросс-валидацией[[14]](#footnote-14) нескольких наборов данных для обнаружения мошеннических действий в автостраховании. Принцип кросс-валидации заключается в разделении доступных данных на несколько подмножеств. Затем, на каждой итерации, одно из подмножеств используется для тестирования модели, а остальные – для ее обучения. Таким образом, каждое подмножество используется как для обучения, так и для тестирования модели. Обычно используется k-fold кросс-валидация, где k – это количество подмножеств.

Результаты показывают, что использование кросс-валидации нескольких наборов данных улучшает качество обнаружения мошенничества в автостраховании. Автор обнаружил, что алгоритмы машинного обучения, такие как случайный лес и SVM, показывают лучшую производительность в обнаружении мошенничества в автостраховании, чем деревья решений [17].

В статье «*Machine learning algorithms for fraud prediction in property insurance: Empirical evidence using real-world microdata*» авторы Matheus Kempa Severino и Yaohao Peng исследуют возможности использования алгоритмов машинного обучения для обнаружения мошенничества в страховании недвижимости. Авторы применили методы машинного обучения, такие как логистическую регрессию, дерево решений, случайный лес и градиентный бустинг, для обнаружения мошеннических действий в страховании недвижимости. Они использовали реальные данные, предоставленные страховой компанией, включая информацию о страховых полисах, пожарах и выплатах.

Результаты показали, что все алгоритмы машинного обучения, примененные в этом исследовании, способны эффективно обнаруживать мошеннические действия в страховании недвижимости. В частности, логистическая регрессия и градиентный бустинг показали наилучшую производительность с точки зрения F1 меры. Кроме того, авторы обнаружили, что включение дополнительных переменных, таких как географическое расположение и возраст здания, может значительно повысить производительность моделей машинного обучения [15].

Статья «*Modelling different types of automobile insurance fraud behaviour in the Spanish market*» была написана Manuel Artı́s, Mercedes Ayuso и Montserrat Guillén. В ней авторы рассматривают различные виды мошенничества в области автомобильного страхования в Испании и предлагают моделирование поведения мошенников для выявления мошенничества и улучшения мер борьбы с ним. Они использовали данные о более чем 100 000 автомобильных страховых полисах, проданных в Испании в течение года, чтобы создать модели мошеннического поведения. В частности, авторы рассмотрели различные типы мошенничества, такие как установка фальшивых деталей в автомобиль, подделка документов и заявлений о страховом случае и организация умышленных аварий.

Для моделирования мошеннического поведения авторы использовали логистическую регрессию, случайный лес и градиентный бустинг. Они также использовали методы преобразования данных, такие как метод главных компонент[[15]](#footnote-15) и анализ факторов для улучшения качества моделей. МГК помогает сохранить наиболее важные характеристики исходных данных и сократить размерность пространства, что позволяет более эффективно работать с данными, особенно в случае большого количества признаков.

Результаты показали, что моделирование мошеннического поведения может быть эффективным для выявления мошенничества в автомобильном страховании. Логистическая регрессия показала наилучшую производительность для определения мошеннического поведения, градиентный бустинг также показал хорошие результаты, а случайный лес не был эффективным в этой задаче. Авторы также выяснили, что определенные факторы, такие как возраст и тип автомобиля, являются важными признаками для определения мошеннического поведения [4].

Статья «*Development of an Expert System for the Automatic Detection of Automobile Insurance Fraud*» описывает разработку экспертной системы для автоматического обнаружения мошенничества в автомобильной страховой отрасли. Авторами статьи являются Эль-Башир Бельхаджи и Жорж Дионн. Они использовали экспертную систему, основанную на методах искусственного интеллекта и машинного обучения. В основе системы лежат алгоритмы, которые позволяют автоматически анализировать страховые полисы и выявлять потенциальные случаи мошенничества. Для этого были использованы различные методы обработки данных, включая анализ текстов и изображений.

Результаты показали, что экспертная система может достаточно точно определять случаи мошенничества в автомобильной страховой отрасли. В частности, система позволяет снизить количество ложноположительных и ложноотрицательных результатов, что может привести к экономической выгоде для страховых компаний [6].

Статья «*Detecting insurance claims fraud using machine learning techniques*» написана авторами Riya Roy и K. Thomas George и была опубликована в журнале «International Journal of Data Science and Analytics» в 2019 году. Целью данной статьи является исследование применения методов машинного обучения для выявления мошенничества при страховых выплатах. Для этого авторы использовали данные о страховых выплатах из базы данных страховой компании в Индии. В статье описывается процесс подготовки данных, который включал очистку, преобразование и отбор признаков. Затем авторы применили несколько методов машинного обучения для обнаружения мошеннических выплат. В частности, они использовали алгоритмы дерева решений, случайного леса и многослойного персептрона[[16]](#footnote-16). Основная идея многослойного персептрона заключается в том, чтобы найти такие веса и пороговые значения для каждого нейрона, чтобы минимизировать ошибку предсказания. Для этого используется метод обратного распространения ошибки (Backpropagation), который позволяет корректировать веса и пороговые значения с помощью градиентного спуска.

По результатам исследования было выявлено, что методы машинного обучения позволяют эффективно выявлять мошеннические выплаты, и что алгоритм случайного леса оказался наиболее точным из протестированных методов. Кроме того, авторы отметили, что использование нескольких методов машинного обучения может улучшить точность обнаружения мошенничества [13].

Статья "*Very Deep Convolutional Networks for Text Classification*" представляет исследование использования сверточных нейронных сетей (Convolutional Neural Networks, CNN) для задач классификации текста. Авторы предлагают две модели: одну для классификации текстов с фиксированной длиной и другую для классификации текстов произвольной длины. В статье описывается процесс обработки текста перед подачей на вход CNN, включая преобразование слов в векторы и использование предобученной модели word2vec[[17]](#footnote-17). В качестве базовой архитектуры CNN используется ряд сверточных слоев, последовательно соединенных с полносвязными слоями, для получения выхода в виде вероятностей принадлежности к классам.

Авторы провели эксперименты на двух различных наборах данных и сравнили результаты с другими моделями. Они утверждают, что предложенные модели позволяют достигнуть высокой точности классификации текстов, превосходя другие подходы, включая рекуррентные нейронные сети. Таким образом, статья демонстрирует эффективность использования сверточных нейронных сетей для классификации текста и предлагает две модели, которые превосходят другие методы в данной задаче [8].

В статье "*Prediction of Insurance Fraud Detection using Machine Learning Algorithms*" авторы представляют исследование использования алгоритмов машинного обучения для выявления мошеннических действий в сфере страхования. В работе были применены различные алгоритмы машинного обучения, такие как деревья решений, случайный лес, градиентный бустинг, метод опорных векторов, наивный Байесовский классификатор, Adaboost[[18]](#footnote-18), метод k-ближайших соседей (KNN[[19]](#footnote-19)), многослойный персептрон (MLP) для предсказания степени риска мошенничества в страховых случаях.

Авторы пришли к выводу, что использование алгоритмов машинного обучения может быть эффективным в выявлении мошеннических действий в сфере страхования. Они также отметили, что градиентный бустинг является наиболее эффективным алгоритмом среди всех протестированных в их исследовании [14].

В статье "*A Comprehensive Study of Healthcare Fraud Detection based on Machine Learning*", написанной Aarti M. Karandikar и Shivani S. Waghade, описывается использование методов машинного обучения для обнаружения мошенничества в области здравоохранения. Авторы исследовали различные методы машинного обучения, такие как деревья решений, наивный байесовский классификатор, многослойный персептрон и метод опорных векторов, для обнаружения мошеннических схем в здравоохранении. Результаты показали, что метод опорных векторов дает лучшие результаты в сравнении с другими методами [10].

Статья "*A fraud detection approach with data mining in health insurance*" рассматривает задачу обнаружения мошенничества в области медицинского страхования с использованием методов дата-майнинга. В статье предлагается метод, который использует алгоритмы классификации, такие как навиный байес в сочетании с решающими деревьями, для определения потенциальных случаев мошенничества на основе анализа больших объемов данных, включая информацию о медицинских услугах и платежах. Авторы статьи провели эксперименты на реальных данных из турецкой системы медицинского страхования и показали, что предложенный метод имеет высокую точность и способен обнаруживать различные виды мошенничества в области медицинского страхования. [12].

Статья "*Machine learning algorithms for document clustering and fraud detection*" автора S. Yaram, опубликованная в 2016 году, связана с мошенничеством в целом (не в страховании). В статье автор рассматривает применение алгоритмов машинного обучения для кластеризации документов и обнаружения мошенничества. Автор обсуждает основные методы кластеризации документов, такие как иерархическая кластеризация, k-means, DBSCAN и т.д. Он также обсуждает алгоритмы машинного обучения, используемые для обнаружения мошенничества, такие как решающие деревья, случайный лес, градиентный бустинг и другие. Для проверки эффективности алгоритмов автор проводит эксперименты на наборе данных документов. В результате экспериментов было показано, что применение алгоритмов кластеризации и машинного обучения позволяет значительно повысить точность обнаружения мошенничества в документах [18].

Статья "*Big Data and Specific Analysis Methods for Insurance Fraud Detection*" исследует использование методов анализа больших данных для обнаружения мошенничества в страховании. Авторы описывают основные виды мошенничества в страховании, а также проблемы, связанные с их выявлением. Далее авторы рассматривают преимущества и вызовы, связанные с использованием больших данных и анализа данных для обнаружения мошенничества в страховании. В статье также описываются основные методы машинного обучения, которые используются для обнаружения мошенничества в страховании, такие как решающие деревья, случайные леса и нейронные сети. Авторы также рассматривают методы, используемые для улучшения производительности алгоритмов машинного обучения, такие как кросс-валидация и отбор признаков. В заключение авторы отмечают, что использование больших данных и методов анализа данных может привести к более эффективному выявлению мошеннических схем и сокращению потерь, связанных с мошенничеством в страховании [7].

Исходя из всех изученных статей, мы можем сделать вывод, что нет однозначно лучшего алгоритма для детекции мошеннических страховых заявлений, так как в разных исследованиях разные алгоритмы показывали лучшие резульаты. Однако можно точно сказать, что использование методов машинного обучения позволяет приблизиться к полному решению задачи обнаружения мошеннических операций в сфере страхования, следовательно, улучшая используемые алгоритмы и разрабатывая новые, мы можем в конечном итоге в будущем полностью решить данную задачу либо минимизировать количество одобренных мошеннических заявлений.

**Глава 2. Методология обработки данных и построения моделей**

**2.1. Описание данных**

Несмотря на потенциальную мощность описанных выше алгоритмов, ни один из них не будет нормально работать, если у исследователя не будет в расположении достаточного по объёму, информатированного, качественного и корректно собранного массива данных. Они используются для обучения модели, то есть настройки параметров модели таким образом, чтобы она могла предсказывать результаты для новых данных, которые не были использованы в процессе обучения. Данные используются для определения взаимосвязей и закономерностей между признаками (факторами) и целевой переменной (то, что мы хотим предсказать). На основе этих взаимосвязей и закономерностей модель может предсказывать значения целевой переменной для новых наблюдений. Чем более качественные и разнообразные данные используются для обучения, тем более точные и устойчивые к новым данным будут предсказания модели. Поэтому важно не только иметь данные, но и правильно их предобрабатывать и подготавливать для обучения модели.

В качестве данных у нас имеется портфель крупной Российской страховой компании. Изначально в портфеле мы имеем 59656 наблюбдений (31240 из них – уникальные договоры). Из признаков нам доступно 56 столбцов: *Вид страхования, Канал продаж, Серия договора, Номер договора, Старый номер договора, Ф.И.О. Страхователь, Дата вступления договора в силу, Дата начала договора, Дата окончания договора, Дата конца, Валюта ответственности, Задолженность на дату убытка, руб., Доп. доход, Агент по договору, ФИО заявившего, Тип заявителя, Дата составления, Дата подписания, Дата принятия, Тип случая, Причина случая, Дата и время случая, Место случая, ID Застрахованного, Ф.И.О. Застрахованного, Дата рождения, Регион, Тип страх. случая, Дата страх. случая, Группа инвалидности, ИД ЭСС, Статус ЭСС, Страховой риск, по которому произошло событие, Страховая сумма по риску, в валюте ответственности, Кол-во дней нетрудоспособности, Оплаченные дни нетрудоспособности, Дата подачи полного пакета, ЗУ, % от страховой суммы, РЗУ, в валюте ответственности, Описание ЭСС, Дата отклонения ЭСС, Размер страховой выплаты в валюте ответственности, Размер страховой выплаты, рубли, Тип выплаты, Доля выгодоприобретателя, Дата составления акта, № акта, Сумма к выплате, рубли, НДФЛ, Состояние выплаты, Дата осуществления, ИД выплаты, Способ выплаты, CC main program* (категориальная переменная, созданная работниками компании)*, Risk code, СС main programm* (численная переменная, созданная работниками компании)*, Мошенничество*. При этом последний столбец является целевой переменной, которая принимает бинарные значения (1 – если заявление признано мошенническим, иначе – 0).

Конечно, многие из этих признаков окажутся неинформативными (как, например, номер договора или ФИО), однако останется всё равно достаточно большое количество полезных для анализа и предсказаний данных.

Мы можем посмотреть описательную статистику числовых признаков (см. Приложение 15). При этом из описательного анализа заранее были исключены неинформативные числовые признаки: «ИД выплаты», «№ акта», «ИД ЭСС», «ID Застрахованного», «Агент по договору», «Старый номер договора», «Номер договора», нужные для организации файлов страховой компании.

Как мы можем наблюдать, столбец «Страховая сумма по риску, в валюте ответственности» имеет значения в 59070 строках, а столбец «Кол-во дней нетрудоспособности» лишь только в 17926 строках. При этом минимальное значение второго столбца равно 1. Это значит, что в остальных строках, где нет значений, мы должны проставить 0. Это и другие преобразования будут описаны далее в параграфе обработки и преобразования данных. Также стоит отметить, что минимальное значение столбца «НДФЛ» изначально было равно отрицательному числу (что, конечно, скорее всего является ошибкой), а подавляющая часть значений имеет нули (данный вывод можно сделать исходя из нулевых значений верхнего и нижнего квартилей, которые были посчитаны отдельно), соответственно, медиана тоже равна нулю. Благодаря описательному анализу мы можем выдвинуть гипотезу о том, что данный признак будет шумовым для наших моделей и логично будет удалить его из обучающего набора данных.

Сначала был проведён тест на нормальность (нормальное распределение признаков). Для этого мы использовали тест Шапиро-Уилка и критерий согласия Пирсона.

Тест Шапиро-Уилка (Shapiro-Wilk test) — это статистический тест на нормальность распределения данных. Он используется для проверки гипотезы о том, что набор данных был взят из нормального распределения. Нулевая гипотеза – распределение является нормальным, альтернативная – выборка имеет другое распределение (такие же гипотезы будут и в двух оставшихся критериях). Если результат теста Шапиро-Уилка отвергает эту гипотезу, то можно предполагать, что данные имеют другое распределение. Формула данного теста в аппроксимации выглядит так:

, (2.1)

где – сумма квадратов отклонений от среднего,

, *,*

*, m –* вектор ожидаемых значений статистики порядка одинаково распределённых и независимых случайных величин, извлечённых из стандартного распределения, *V –* ковариационная матрица статистических данных нормального порядка,i – номер элемента в вариационном ряду, *n* – количество наблюдений в определённом признаке, – численное значение *i*-го объекта определённого признака.

Критерий согласия Пирсона (Pearson`s chi-squared test) — это статистический тест, который используется для проверки того, насколько хорошо наблюдаемые данные соответствуют ожидаемым значениям в заданном распределении. Критерий согласия Пирсона может использоваться для проверки гипотезы о том, что данные взяты из определенного распределения.

Результаты представлены ниже в табл. 1. Как мы можем наблюдать, ни одна числовая переменная не имеет нормального распределения, однако для методов машинного обучения, которые мы будем использовать в следующих главах, это не обязательное условие.

Бо́льшая часть данных, представленных в этом исследовании, состоит из текстовых данных. Причём основная часть из них – категориальные признаки. Мы их тоже будем использовать, так как это дополнительная информация, которая гипотетически может помочь нашим моделям делать более качественные предсказания.

*Таблица 1*

*Результаты проверки признаков на нормальное распределение*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | *Тест Шапиро-Уилка* | | *Критерий согласия Пирсона* | |
|  | Значение статистики | P-value | Значение статистики | P-value |
| *Задолж. на дату убыт.* | 0,055935323 | 0 | 119 547,25 | 0 |
| *Доп. доход* | 0,183552861 | 0 | 112 746,72 | 0 |
| *Страх. сум. по риску, в вал. отв.* | 0,803151369 | 0 | 51 168,10 | 0 |
| *Кол-во дней нетрудоспособ.* | 0,70979321 | 0 | 25 189,42 | 0 |
| *Опл. дн. нетрудоспособ.* | 0,666380882 | 0 | 14 534,27 | 0 |
| *ЗУ, % от страх. сум.* | 0,293629527 | 0 | 80 073,73 | 0 |
| *РЗУ, в вал. отв.* | 0,032360494 | 0 | 207 901,68 | 0 |
| *Разм. страх. выпл. в вал. отв.* | 0,292791247 | 0 | 164 989,67 | 0 |
| *Разм. страх. выпл.* | 0,296440542 | 0 | 142 427,16 | 0 |
| *Доля выгодоприобр.* | 0,056783617 | 0 | 101 552,04 | 0 |
| *Сум. к выпл.* | 0,259172022 | 0 | 152 872,82 | 0 |
| *НДФЛ* | 0,031043649 | 0 | 155 708,71 | 0 |
| *СС main programm* | 0,290199339 | 0 | 164 824,16 | 0 |
| *Разница между датами вступления договора в силу и наступления страх. случая* | 0,85858351 | 0 | 13 748,45 | 0 |

Также стоит отметить, что наша выборка несбалансированная. На этапе оценки качества построенных моделей это будет учтено, в исследовании будут использоваться метрики, устойчивые к несбалансированному распределению целевой переменной.

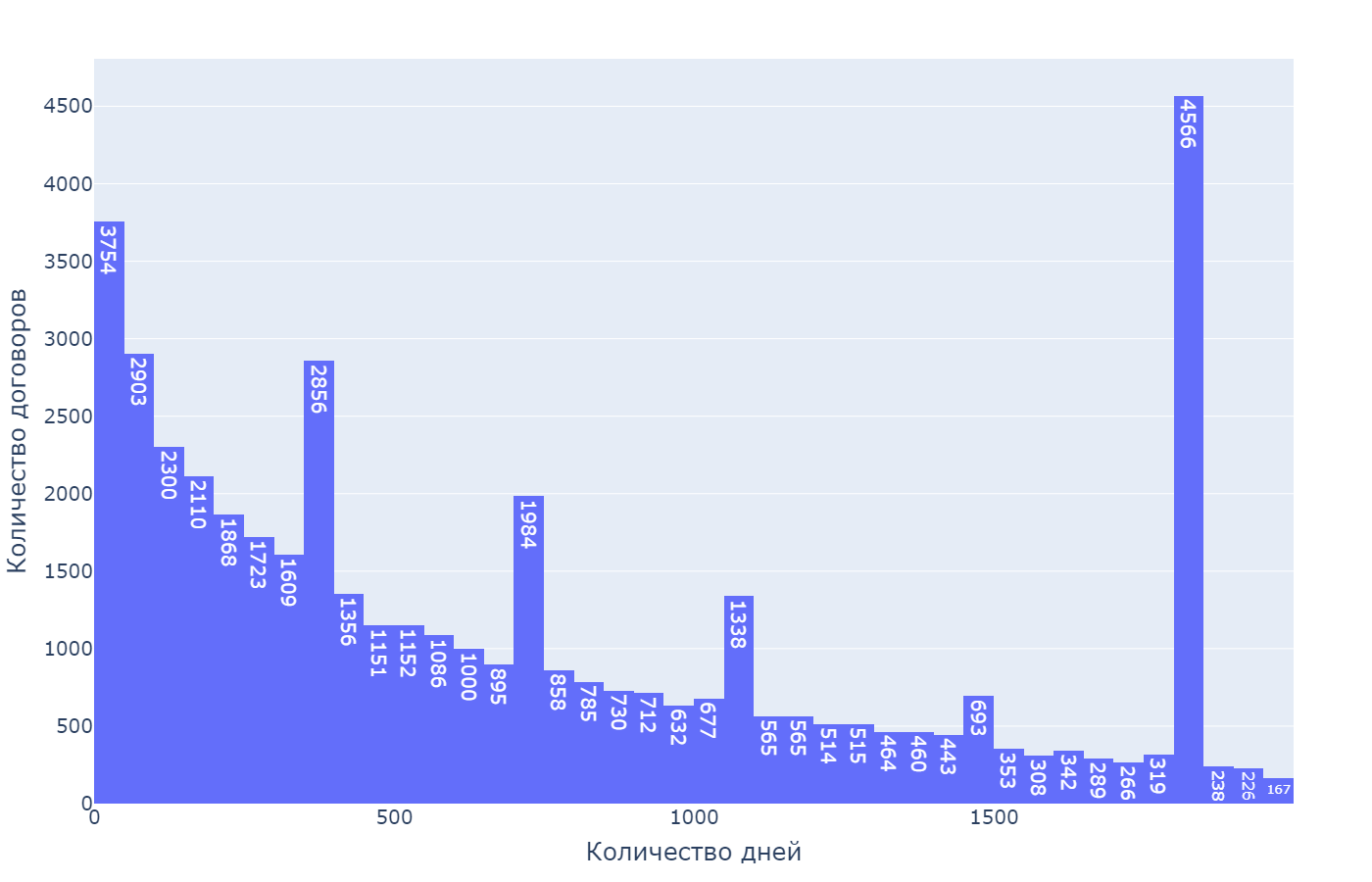
Одним из недостатков рассматриваемого нами портфеля является то, что в текстовых данных (столбец «Описание ЭСС») иногда может встречаться одна и та же информация в разных форматах (например, в одном случае будет описание «2-х лодыжечный перелом левой голени», а в другом – «2-х лодыжечный перелом лев. голени»). Более того, так как данный признак заполняется людьми вручную, иногда в этом столбце можно обнаружить орфографические ошибки. Без обработки данных это бы означало то, что одно и то же слово с ошибкой и без для модели являлись бы абсолютно разными значениями в векторах. Эти проблемы решаются с помощью использования встроенных библиотек обработки текстовой информации в Python, подробнее о которых будет рассказано в следующем параграфе.

**2. 2. Процесс обработки и преобразования данных, кластерный анализ**

После удаления дубликатов сразу по всем признакам в Python с помощью библиотеки Pandas мы получаем 59121 строку. Таким образом, мы всё ещё имеем большой объём данных. Однако, как мы уже убедились, во многих столбцах есть пропуски. Мы будем заполнять их либо нулями, как в примере, представленном в прошлом параграфе с «Кол-во дней нетрудоспособности», либо средними значениями (в зависимости от логики, которой поддаются эти пропуски). Эти действия мы будем производить далее по ходу работы после преобразований, связанных с добавлением новых признаков.

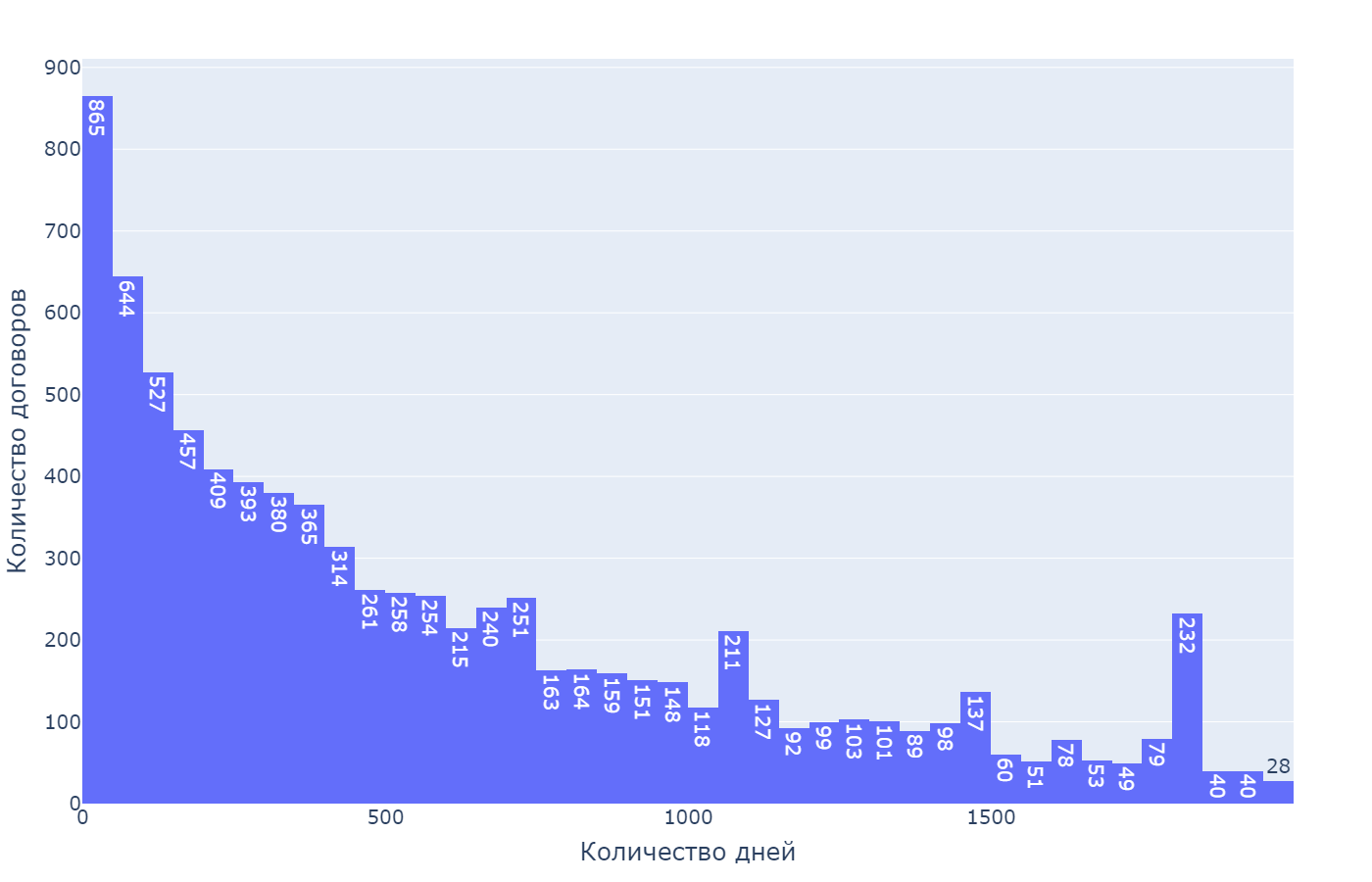
Сначала был добавлен в наши данные новый признак, который будет исчисляться в днях: *Разница между датой вступления договора в силу и наступлением страхового случая*. Это помогло нам лучше понять структуру данных и разобраться, есть ли какая-то временная зависимость между вступлением договора в силу и наступлением страхового случая в мошеннических и не мошеннических заявлениях. Изначально эта переменная тоже имела некоторые аномальные значения: могла быть отрицательной или положительной, но большой по модулю. Отрицательного значения не может быть, так как в таком случае, страховой случай наступал бы до вступления страхового договора в силу, что невозможно (рассматриваются только события, которые произошли с клиентом за срок действия договора). Более того, максимальное значение было 42343 дней, что тоже невозможно, так как это примерно 116 лет, а у нас в портфеле только личное страхование. Следовательно, это ошибки в данных и сильные выбросы, которые мы просто обрезали нижним пределом – 1, а верхним – 5000 дней (табл. 1). Помимо этого сразу было произведено разбиение выборки на тренировочную и тестовую в пропорции 3 к 1 соответственно (75% - тренировочная выборка, 25% - тестовая).

Далее построим гистограммы, где по оси ординат будет количество договоров, а по оси абсцисс – количество прошедших дней, и попробуем выявить какие-либо зависимости между количеством дней с момента вступления договора в силу до момента наступления страхового случая и мошенничеством (на графиках будут данные до 2000 дня для большей наглядности). На рис. 4 представлен график по всем не мошенническим наблюдениям.



*Рис. 4. Гистограмма зависимости количества не мошеннических заявлений от количества дней, прошедших с момента вступления договора в силу до момента наступления страхового случая*

На рис. 4 мы можем наблюдать всплески на значениях, примерно кратных 365 (количеству дней в одном году). Конечно, это может быть какой-то особенностью учёта и записи даты страховых случаев. Также это может быть связано с тем, что в портфеле существуют договоры на дожитие, которые обычно заключаются на определённое количество лет. Максимальный пик в диапозоне от 1800 до 1849 дней (5 лет).



*Рис. 5. Гистограмма зависимости количества мошеннических заявлений от количества дней, прошедших с момента вступления договора в силу до момента наступления страхового случая*

Далее построим график (рис. 5), на котором отобразим те же самые данные, но разрез будет только по мошенническим заявлениям.

Как мы можем наблюдать, частотные скачки всё ещё присутствуют, однако они намного более сглажены, чем на рис. 4. При определённом значении λ данный график частот может напоминать экспоненциальное распределение, функция плотности которого описывается формулой , где х в нашем случае – количество дней, а λ – параметр интенсивности. Мы не будем проверять, принадлежат ли распределения на рис. 4 и рис. 5 экспоненциальному закону распределения, так как в этом нет необходимости для решения нашей задачи.

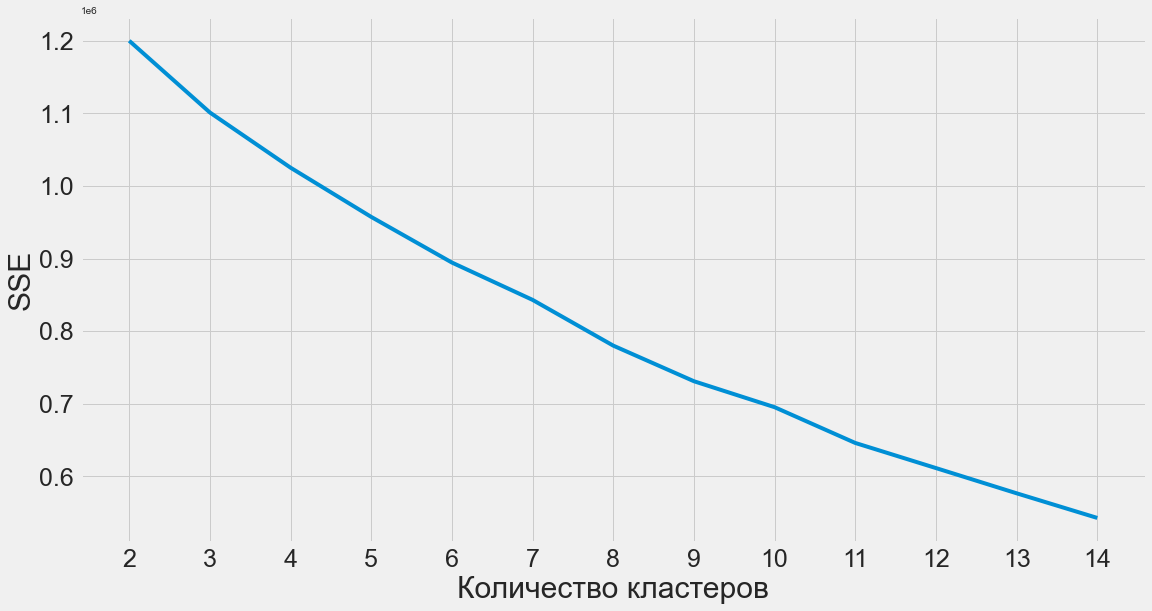
Теперь займёмся преобразованием данных. Так как в страховом портфеле очень много категориальных данных, перекодируем их в числовые признаки. Наиболее распространённый вариант кодировщика текстовых данных – One Hot Encoder. Логика данного кодировщика описана в предыдущей главе. Проблема очевидна – мы имеем большое количество столбцов, более того, в каждом из них большое количество уникальных категориальных значений. Как итог – получим огромное количество столбцов, что иррационально, так как модели будут обучаться очень долго. Поэтому мы не будем использовать данный алгоритм. Вместо этого будет использован другой алгоритм обработки категориальных данных – Target Encoder. Работа данного алгоритма также была описана в первой главе.

Далее можно провести кластерный анализ для понимания, насколько вообще выборка может быть сегментирована по каким-то однородным группам. Для этого были использованы такие переменные: «Канал продаж», «Серия договора», «Валюта ответственности», «Тип заявителя», «Место случая», «Регион», «Группа инвалидности», «Страховой риск, по которому произошло событие», «CC main program», «Risk code», «Задолженность на дату убытка, руб.», «Доп. доход», «Страховая сумма по риску, в валюте ответственности», «Кол-во дней нетрудоспособности», «Оплаченные дни нетрудоспособности», «ЗУ, % от страховой суммы», «РЗУ, в валюте ответственности», «Размер страховой выплаты в валюте ответственности», «Размер страховой выплаты, рубли», «Доля выгодоприобретателя», «Сумма к выплате, рубли», «СС main programm», «Разница между датой вступления договора в силу и наступлением страхового случая». Делать мы это будем с помощью достаточно известного и широко используемого алгоритма k-means.

Если взглянуть на формулу расчёта расстояний (1.2), то становится очевидно, что данные нужно стандартизировать. Если этого не сделать, то данные из столбца «Задолженность на дату убытка, руб», например, будут влиять на итоговые значения центроидов и кластеризацию намного сильнее, чем преобразованные ранее данные категориального признака «Регион», значения которого ∈ [0; 1]. Делать мы это будем с помощью функции StandardScaler, которая используется для стандартизации данных перед их обработкой в алгоритмах машинного обучения.

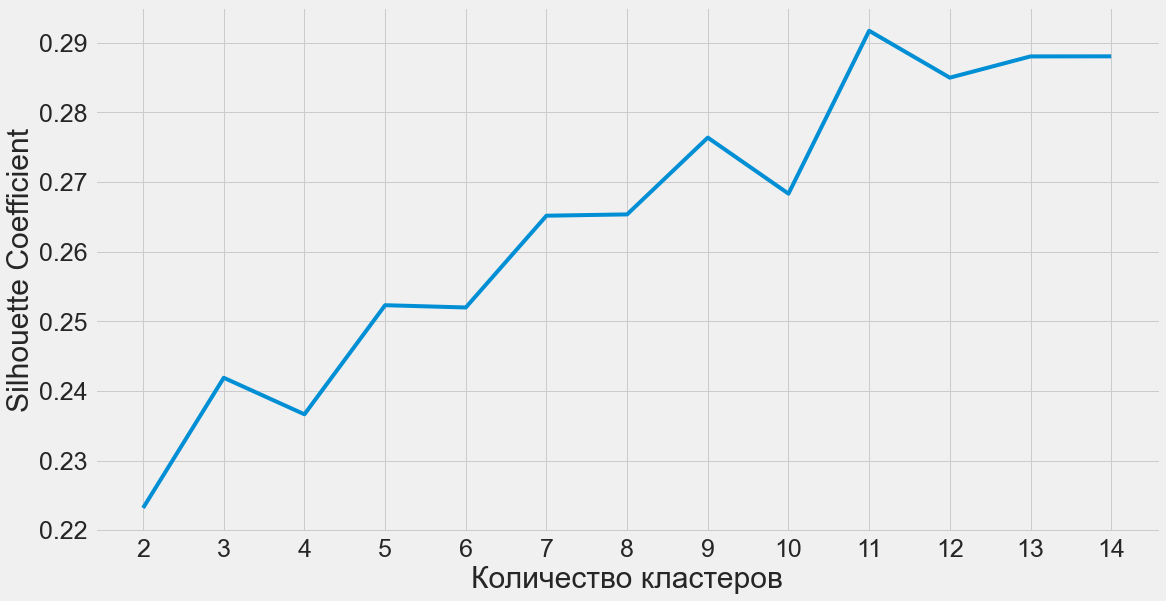
После проведения необходимых преобразований будем разбивать выборку на кластеры от 2 до 14 (для наглядности). Измерять качество кластеризации мы будем с помощью четырёх метрик: SSE, Silhouette коэффициент, Davies-Bouldin индекс и Calinski-Harabasz индекс.

Проведём визуальный анализ, чтобы понять, на сколько кластеров лучше всего делится выборка. На первом графике (рис. 6) изображена зависимость суммы квадратов расстояний между каждым объектом кластера и центроидом кластера. Очевидно, что при увеличении количества кластеров это значение становится меньше, так как центроидов больше и расстояние от них до объектов, принадлежащих определённому кластеру, уменьшается. Для вывода, на какие кластеры лучше делится выборка одного этого показателя мало.



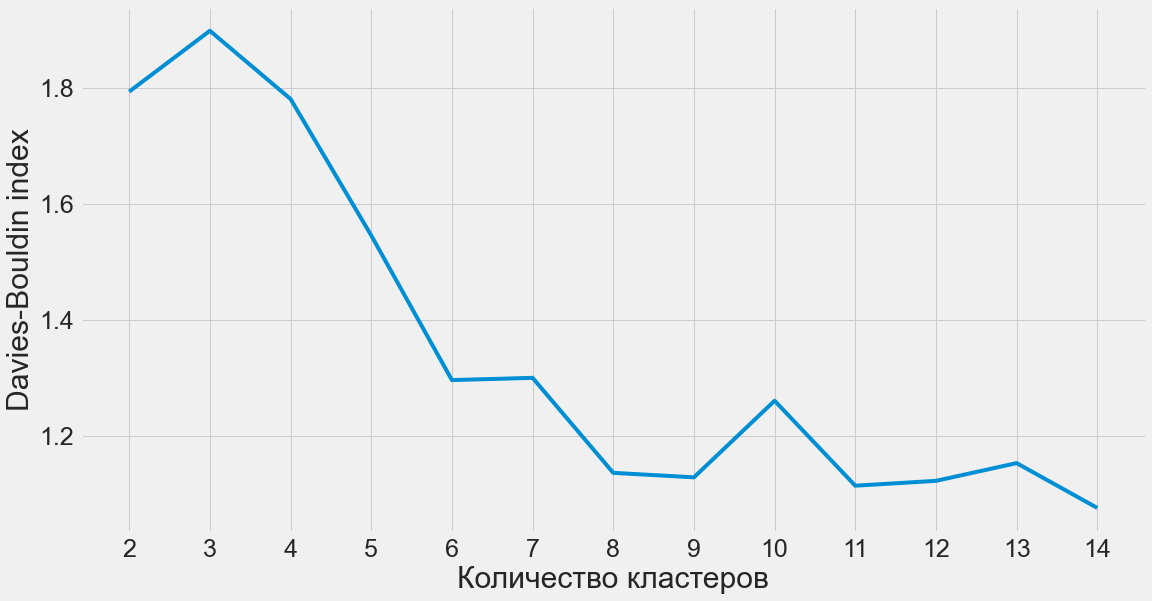
*Рис. 6. График SSE в зависимости от количества кластеров в модели k-means*

На рис. 7 мы можем наблюдать постепенное увеличение значений метрики. Здесь кажется, что 2 кластера – худшее разбиение выборки (в данном случае коэффициент получается минимальным). Однако стоит отметить, что сами различия достаточно минимальны. Если сам коэффициент ∈ [-1; 1], то у нас значения метрики варьируются всего от 0,22 до 0,29. Вообще значения немного больше нуля, что говорит нам о том, что качество кластеризации нормальное.



*Рис. 7. График Silhouette коэффициента* *в зависимости от количества кластеров в модели k-means*

На третьем графике (рис. 8) изображена метрика Davies-Bouldin в зависимости от количества кластеров. Как мы видим, наилучшего значения она достигает при максимальном количестве кластеров, однако такое разбиение выборки не целесообразно и не интерпретируемо.



*Рис. 8. График Davies-Bouldin индекса в зависимости от количества кластеров в модели k-means*

На последнем графике (рис. 9) проиллюстрирована зависимость Calinski-Harabasz индекса и количества кластеров в модели. Как мы можем наблюдать наилучшее значение достигается в случае разбиения выборки на 2 кластера.

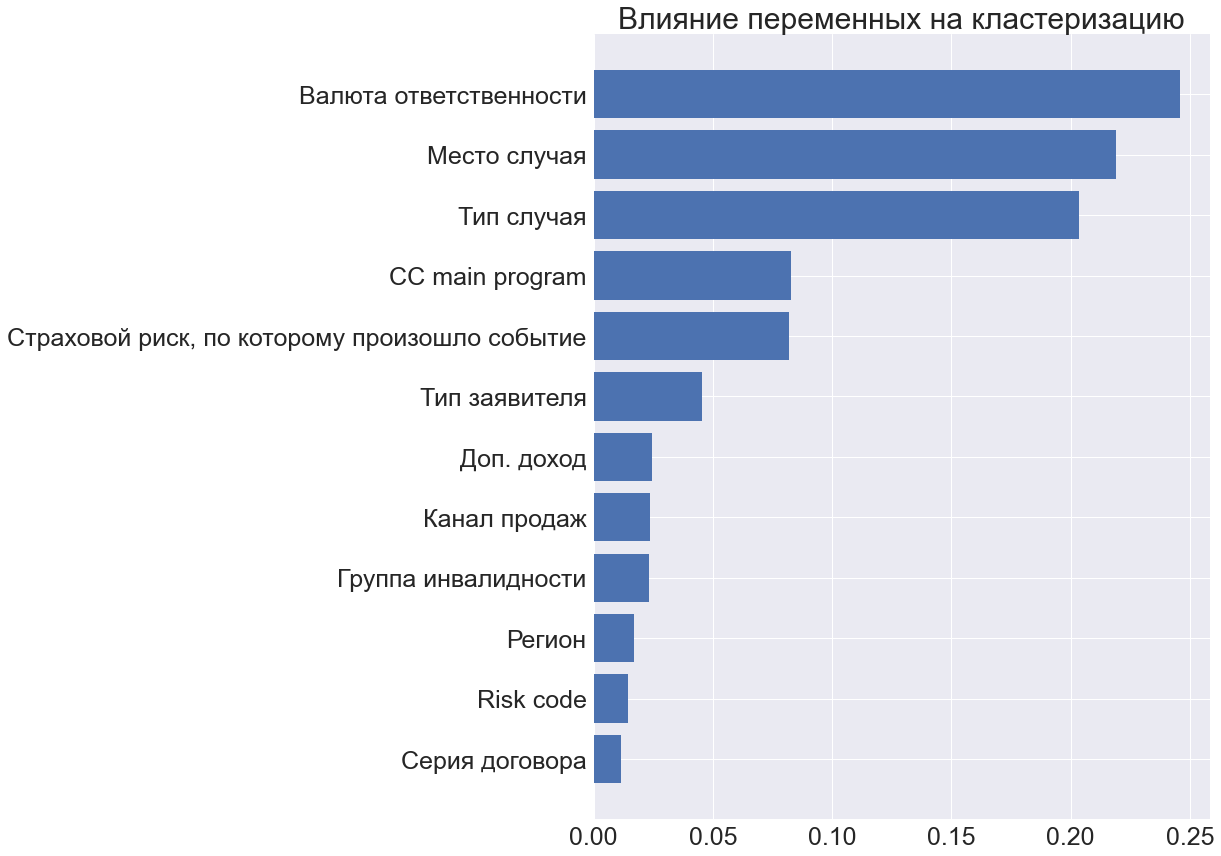


*Рис. 9. График Calinski-Harabasz индекса в зависимости от количества кластеров в модели k-means*

На последнем графике (рис. 9) проиллюстрирована зависимость Calinski-Harabasz индекса и количества кластеров в модели. Как мы можем наблюдать наилучшее значение достигается в случае разбиения выборки на 2 кластера.

Также можно посмотреть на то, какие переменные являются наиболее значимыми при разбиении на 2 кластера (так как целевая переменная принимает 2 значения) с помощью алгоритма k-means (рис. 10). Исходя из гистограммы, представленной ниже, кажется, что максимальное влияние вносят столбцы «Валюта ответственности», «Место случая» и «Тип случая». Скорее всего дело в том, что они имеют небольшое количество уникальных категориальных значений.

Взглянем на полученные центроиды кластеров (табл. 2), с помощью которых мы сможем понять, по значениям каких переменных кластеры различаются.



*Рис. 10. Гистограмма значимости переменных в разбиении данных на 2 кластера с помощью алгоритма k-means*

Признаки, которые имеют близкие по значениям центроиды: «Канал продаж», «Валюта ответственности», «Тип заявителя», «Регион», «Группа инвалидности», «CC main program», «Доля выгодоприобретателя».

В разных кластерах в среднем значения этих переменных практически не различаются. Как ни странно, в этот список попал столбец «Валюта ответственности», который, согласно гистограмме значимости признаков, является наиболее значимым при формировании кластеров. Это объясняется тем, что на гистограмме значимость признаков вычислялась не по значениям центроидов, а по количеству разбиений в деревьях решений[[20]](#footnote-20) алгоритма Random Forest (подробный разбор данного алгоритма будет в следующем параграфе).

*Таблица 2*

*Кластерные центроиды по каждому признаку*

|  |  |  |
| --- | --- | --- |
| Признак\Кластер | 0 | 1 |
| *Канал продаж* | 0,106 | 0,1067 |
| *Серия договора* | 0,1176 | 0,0596 |
| *Валюта ответственности* | 0,1072 | 0,1025 |
| *Тип заявителя* | 0,1082 | 0,0919 |
| *Тип случая* | 0,126 | 0,0375 |
| *Место случая* | 0,1147 | 0,0319 |
| *Регион* | 0,1071 | 0,1008 |
| *Группа инвалидности* | 0,1043 | 0,104 |
| *Страховой риск, по которому произошло событие* | 0,0986 | 0,0249 |
| *CC main program* | 0,1062 | 0,1059 |
| *Risk code* | 0,1275 | 0,0191 |
| *Доп. доход* | 72203 | 16498 |
| *Страх. сум. по риску, в вал. отв.* | 322106 | 89068 |
| *Опл. дн. нетрудоспособ.* | 11,372 | 30,684 |
| *ЗУ, % от страх. сум.* | 137,96 | 112,79 |
| *РЗУ, в вал. отв.* | 860,71 | 10039 |
| *Доля выгодоприобретателя* | 99,653 | 98,941 |
| *СС main programm* | 22837 | 85479 |
| *Разн. меж. дат. вступ. дог. в силу и наст. страх. случ.* | 641,57 | 1571 |

После проведённой кластеризации можно на тренировочных данных посмотреть взаимосвязь между целевой переменной и присвоенным кластером. Для этого будем использовать коэффициенты ранговой корреляции Спирмена[[21]](#footnote-21) и Кендалла[[22]](#footnote-22). Коэффициенты ранговой корреляции — это меры степени, в которой две переменные связаны между собой через их ранговые значения, а не через их точечные значения, так как у нас бинарные переменные.

Формула ранговой корреляции Спирмена:

, (2.2)

где – разность рангов индивидуальных оценок, n – количество наблюдений.

Формула ранговой корреляции Кендалла:

, (2.3)

где , Q – суммарное количество наблюдений, следующих за текущими наблюдениями с меньшим значением рангов, P – суммарное количество наблюдений, следующих за текущими наблюдениями с большим значением рангов, n – количество наблюдений.

Вычисления были произведены в Python, корреляция Спирмена и Кендалла оказались практически равны (-0,2556735494829913 и -0,2556735492377522 соответственно), p-value обоих значений стремится к нулю, следовательно, коэффициенты значимы. Таким образом, целевая переменная и результаты кластеризации на обучающей выборке имеют значимую слабую отрицательную корреляцию.

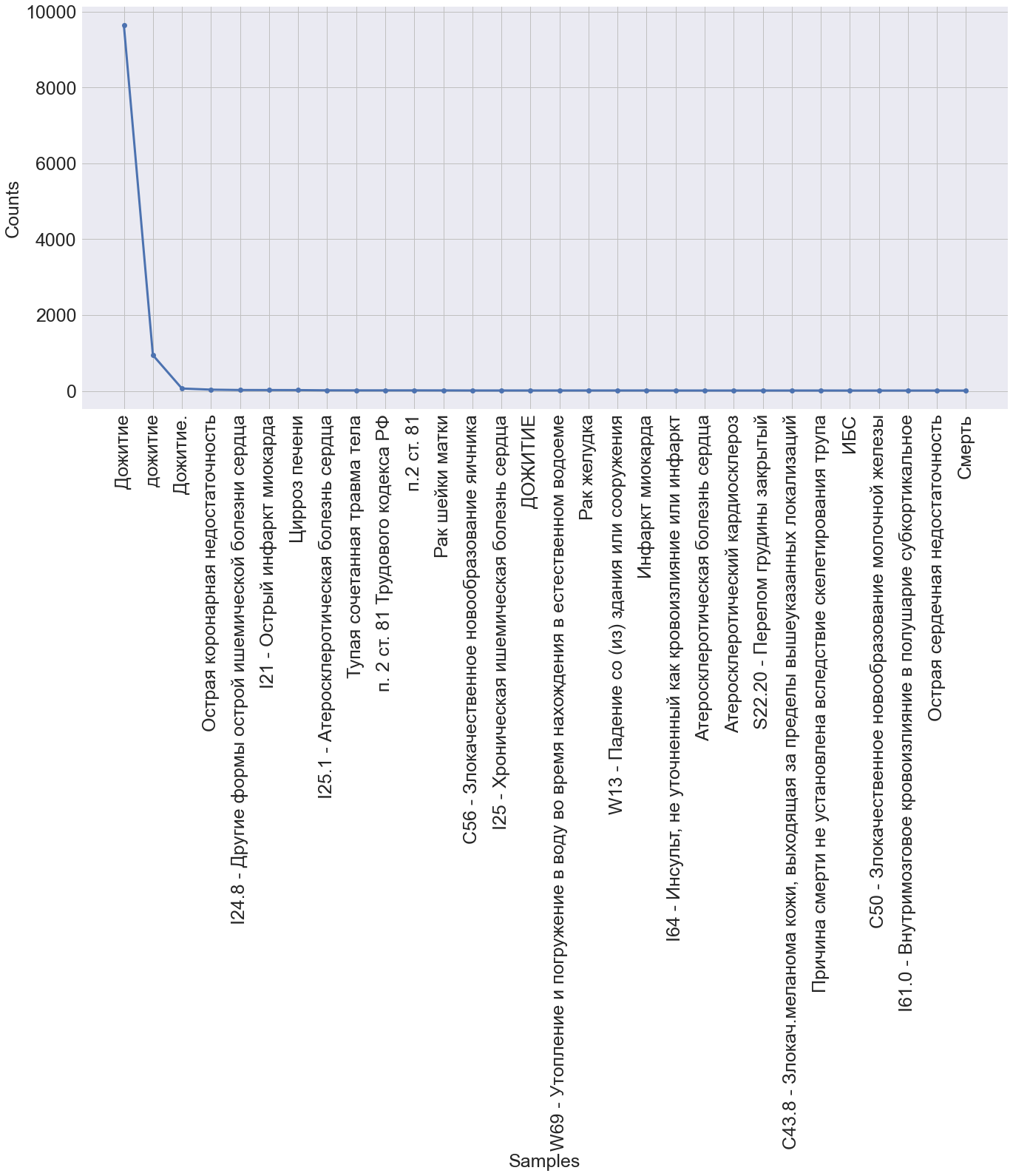
Исходя из значений центроидов, полученных в табл. 2, а также полученных значений корреляции можно выдвинуть некоторые гипотезы. Так как взаимосвязь между целевой переменной и кластером отрицательная, то можно предположить, что чем меньше количество дней между вступлением договора в силу и наступлением страхового случая, тем больше вероятность, что заявление мошенническое, так как в 0 кластере мы имеем меньшее значение центроида по данному признаку, а корреляция между столбцами «Мошенничество» и «Кластер» отрицательная. Ещё одна гипотеза – заявленный ущерб в процентах от страховой суммы в среднем больше в мошеннических заявлениях, а сам размер заявленного ущерба – меньше. Значения центроидов категориальных признаков, которые были перекодированы с помощью Target Encoder, проинтерпретировать невозможно.

Далее можно построить корреляционные матрицы. Их будет две, так как часть переменных – числовые (считается корреляция Пирсона, см. Приложение 29), а часть – категориальные (корреляция Спирмена, см. Приложение 30). В первой матрице целевая переменная слабо коррелирует с остальными числовыми признаками, максимальное значение корреляции по модулю равно 0,18. Во второй матрице можно наблюдать более сильные взаимосвязи между категориальными данными и целевой переменной. Более того, как было отмечено ранее, корреляция целевой переменной со значением кластера примерно равна -0,26.

Подводя небольшой итог по кластеризации наших данных, можно сделать вывод, что разбиение на 2 выборки, исходя из результатов кластеризации, является необоснованным, так как мы имеем текстовые данные, которые сильно зависят от того, какой договор был заключён (страхования жизни или не-жизни). Однако коэффициенты ранговой корреляции между кластером и целевой переменной являются значимыми, поэтому результаты кластеризации будут добавлены как отдельный столбец признаков, на который смогут опираться модели. Текстовые данные в заявлениях по договорам страхования жизни и не-жизни достаточно сильно различаются, так как причины, которые отражены в столбце «Описание ЭСС» в двух группах разнообразны, однако внутри группы достаточно похожи, что будет продемонстрировано ниже. Поэтому выборка была разделена на 2 подвыборки исходя из принадлежности договора к определённому виду страхования, а результаты кластеризации были записаны как отдельный признак.

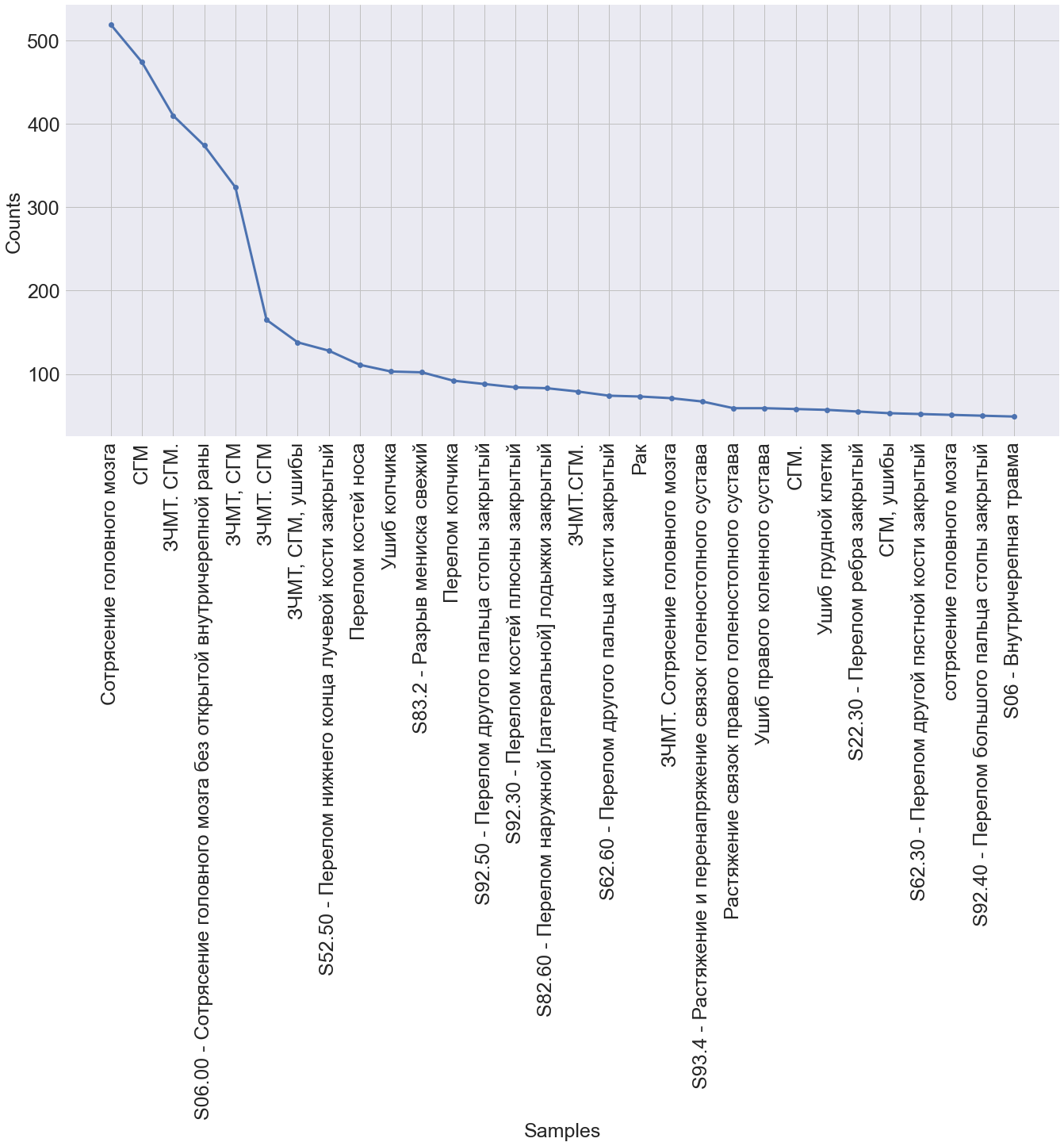
Итак, у нас есть числовые данные (в том числе и перекодированные категориальные) и текстовые, которые нам ещё предстоит преобразовать. Так как в текстовых данных каждое наблюдение может быть уникальным значением, потому что отличие двух наблюдений даже в один строковый знак, например, запятую, является для всех категориальных кодировщиков двумя разными уникальными значениями, то для их преобразования обычные кодировщики не подходят.

Для начала посмотрим на график частотности уникальных значений в текстовом столбце «Описание ЭСС». Так как у нас 2 выборки, на рис. 11 будет отражён график по данным страхования жизни, а на рис. 12 – страхования не-жизни (по 30 самых популярных уникальных значений в каждой выборке).



*Рис. 11. График частотности описаний ЭСС в данных страхования жизни*

Как мы можем наблюдать, бо́льшая часть заявлений – договоры на дожитие. Как уже было сказано выше, проблема в том, что сейчас для модели «Дожитие», «дожитие» и «Дожитие.» – различные строковые значения. Помимо преимущественная часть данных имеют одно и то же значение, поэтому можно попробовать добавить другие текстовые столбцы в выборку.



*Рис. 12. График частотности описаний ЭСС в данных страхования не-жизни*

На втором графике уже данные более разрознены, так как в этой выборке присутствуют договоры медицинского страхования. Снова можно наблюдать схожие значения, которые нужно переформатировать в одно уникальное.

Исходя из первичного визуального анализа частотности, текстовыми данными мы будем считать не только столбец «Описание ЭСС», но ещё и 2 других столбца: Тип страх. случая и Причина случая. Эти 3 признака мы соединим в один и сделаем из них одно предложение. Далее нам необходимо избавиться от проблемы различия строковых знаков в условии одинакового смысла, о которой было сказано выше (наличие или отсутствие знаков препинания, различные падежи у однокоренных существительных, разное время у глаголов и так далее). В Python нами была написана функция, которая переводит все слова в нижний регистр, далее удаляет все ненужные знаки препинания, скобки и другие символы, потом удаляет слова, не несущие смысловой нагрузки (они находятся в библиотеке «stopwords» в Python, были собраны лингвистами и программистами вручную). В списке стоп-слов для русского языка, доступном в данной библиотеке, находится около 400 слов, включая, например, «и», «бы», «в», «к», «как», «который», «над», «наш», «о», «по», «с», «свой», «уже», «это» и так далее. Таким образом, мы оставили в тексте только те слова, которые несут какую-либо смысловую нагрузку. Далее применяем метод SnowballStemmer – инструмент для стемминга (приведения слов к основе) в языках современного мира. Он был разработан Мартином Портером в 1980-х годах и с тех пор был расширен, модифицирован. Стеммеры используются в обработке естественного языка, преобразуют слова в их основу (стем). Например, для слов «дожитие», «дожития» или «дожитием» стеммер возвращает основу «дожит», для слов «смерти» и «смертельный» стеммер возвращает «смерт» и так далее. Можно сказать, что этот алгоритм приводит все слова к морфологическим корням. Однако, конечно, есть слова, в которых он может ошибаться и выдавать не корень. Но он ошибётся во всех таких словах в тексте и поэтому они всё равно будут закодированы одинаково для модели. Это полезно в задачах, связанных с анализом текстов и информации.

После всех проделанных преобразований наши текстовые данные готовы к приведению их в числовой вид. Делаем мы это тоже в Python с помощью обычного частотного токенайзера, который присваивает уникальным строковым наборам (полученным нами до этого основам слов, которые разделены пробелами) значения от 1 до n, где n – количество уникальных основ в наборе данных. Он присваивает значения в порядке частотности. Так самая частая встречающаяся основа (в нашем случае на данных страхования жизни – «дожит», на данных страхования не-жизни – «травм») получает значение равное 1, а самая редко встречающаяся – n. Так как у нас уникальных значений достаточно много, ограничиваем n сверху (максимальное значение n равно 1000). Все остальные значения, которые не попадают в данный лимит, удаляются из набора данных. Так как их частотность достаточно мала, можно пренебречь этими значениями, потому что они слабо повлияют на результат, при этом могут поспособствовать переобучению нейронной сети, да и сама нейронная сеть будет обучаться дольше. Таким образом, наблюдение, содержащее в себе текстовую информацию «Бытовая травма Травмы Ушиб копчика» после преобразования в основы будет иметь вид «бытов травм травм ушиб копчик», а после частотного преобразования с помощью токеназера – [4, 1, 1, 10, 101].

Далее необходимо уравнять все наблюдения по длине, так как предложения могут иметь разные размеры. Для этого используем метод паддинга (padding). Используя функцию pad\_sequences мы приводим все такие числовые списки к одной длине, добавляя перед каждым из них нули, количество которых равно разнице длины самого максимального предложения и длины текущего предложения. Далее делим данные на батчи для загрузки в нейронную сеть. Batch – это набор входных данных, который передается в модель одновременно во время обучения. Вместо того чтобы обучать модель на каждом примере отдельно (так называемый online learning), мы обрабатываем несколько примеров одновременно, что позволяет более эффективно использовать вычислительные ресурсы и сократить время обучения. В качестве примера, при обучении на большом наборе данных, батч может содержать, например, 32, 64 или 128 наблюдений (в нашем случае размер батча – 32). Веса модели обновляются на основе ошибки, вычисленной на всех примерах в батче, а не на каждом наблюдении отдельно. После того, как модель пройдет через все примеры в батче, веса обновляются, и процесс повторяется на следующем батче. Использование батчей имеет несколько преимуществ: ускорение процесса обучения путем уменьшения количества обновлений весов; возможность использования вычислительных ресурсов более эффективно путем параллельной обработки нескольких примеров; уменьшение шума в обновлениях весов путем усреднения градиентов для нескольких примеров в батче.

**2.3. Объяснение архитектуры свёрточной нейронной сети, используемой для обработки текстовых данных и настройка гиперпараметров методов машинного обучения**

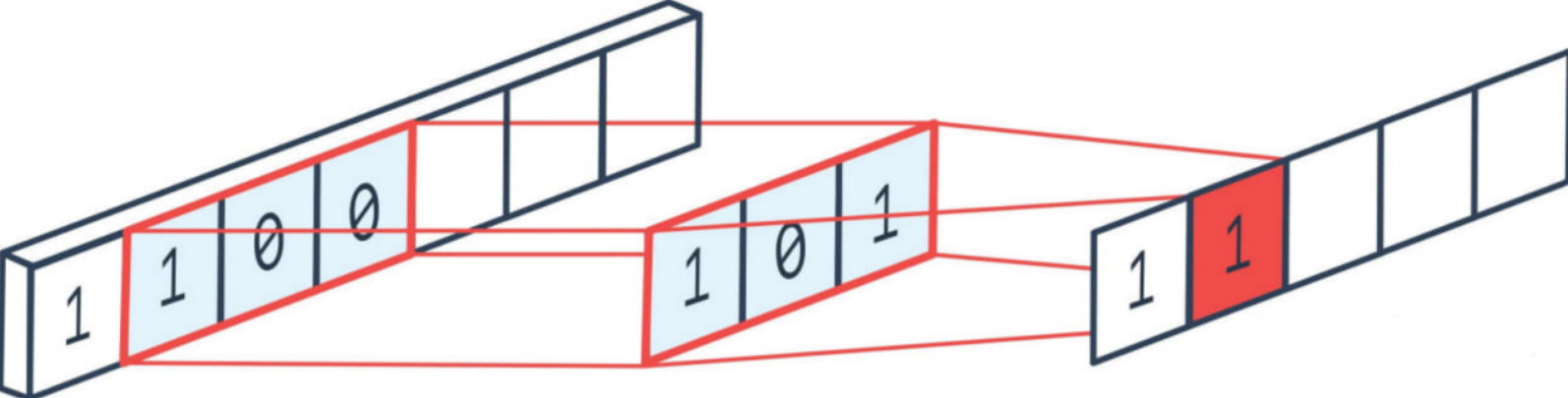
После описания работы стандартной свёрточной нейронной сети в предыдущей главе обратимся к нашей модели. Построенная нейронная сеть представляет собой модель для классификации текстовых данных. Она использует сверточные слои для извлечения признаков из векторных представлений слов и последующее объединение извлеченных признаков для получения общего представления текста (вероятность того, что заявление мошенническое).

В этой модели используется слой Embedding для преобразования каждого предложения в вектор фиксированной длины. Затем применяются три сверточных слоя Conv1D с разными размерами фильтров 2, 3, и 4 для извлечения признаков из векторов предложений. Каждый сверточный слой производит свертку с соответствующим размером фильтра, затем выполняется операция глобальной максимальной подвыборки GlobalMaxPool1D, которая извлекает наиболее значимые признаки из сверток. После этого выходы сверточных слоев объединяются в один вектор, который проходит через полносвязный слой Dense и слой Dropout, который случайным образом отключает некоторые нейроны в процессе обучения, чтобы предотвратить переобучение. На выходе используется один нейрон с сигмоидальной активацией (так как у нас всего 2 класса, нам нужно одно значение вероятности). Наша модель используется для обработки текстовых данных и их классификации на основе извлеченных признаков из векторных представлений слов.

Теперь стоит подробнее рассказать, за что отвечает каждая из представленных выше функций (все они представлены в библиотеке Keras, разработанной для построения нейронных сетей). Функция Embedding преобразует входные целочисленные значения, которые мы получили после обработки текстовых данных, в векторы фиксированной длины. Она создает таблицу, которая содержит векторные представления для каждого уникального индекса, где индексы представляют отдельные слова или символы в текстовых данных. Векторы создаются случайным образом при инициализации модели, а затем обновляются в процессе обучения. На вход этой функции передается размер словаря, то есть количество уникальных слов или символов в обрабатываемых текстовых данных, и размерность векторного представления (максимальная длина нашего предложения). Затем функция Embedding преобразует каждый индекс в соответствующий вектор фиксированной длины. Результатом является матрица формы (batch\_size, sequence\_length, embedding\_dimensions), где batch\_size - размер пакета обработки текстовых данных, sequence\_length - максимальная длина последовательности слов в каждом тексте, а embedding\_dimensions - размерность векторного представления каждого слова или символа. Эта матрица является входом для сверточных слоев нейронной сети, которые обрабатывают ее для извлечения признаков и классификации текстовых данных. Таким образом, слой Embedding помогает нейронной сети понимать взаимосвязь между словами и символами в текстовых данных, что является ключевым фактором при решении задач обработки естественного языка.

Функция Conv1D, которая используется после, является сверточным слоем для обработки одномерных последовательностей данных, таких как временные ряды или тексты. Она выполняет операцию свертки с ядром, которое перемещается по входным данным, извлекая признаки из локальных областей. На вход этой функции передаются параметры filters, kernel\_size, padding и activation. Параметр filters определяет количество фильтров, которые будут применены к входным данным. Каждый фильтр обрабатывает входные данные и извлекает некоторые признаки. Параметр kernel\_size определяет размер ядра свертки, то есть сколько элементов входных данных будет рассматриваться за один раз в одной уникальном фильтре. Параметр padding определяет способ обработки краевых элементов данных (у нас этот параметр обрезает краевые элементы). Параметр activation определяет функцию активации, которая будет применена к выходу сверточного слоя (в нашем случае используется оптимальная по ресурсным затратам функция активации ReLU, описание работы которой будет описано далее).

Во время выполнения модели, входные данные передаются в сверточный слой. Он применяет ядро свертки к входным данным и перемещает его по всему входу, извлекая признаки из локальных областей. Результатом является выходной тензор, содержащий признаки, извлеченные из входных данных. Данный процесс изображён на рис. 13:



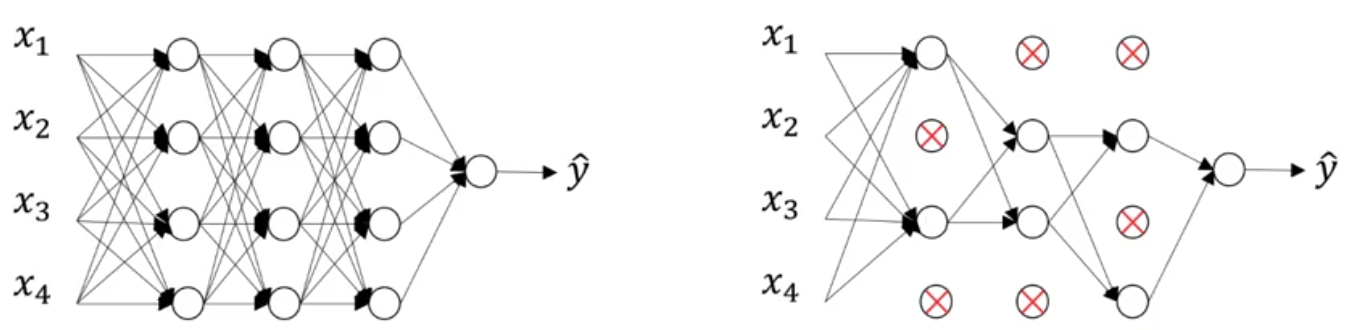
*Рис. 13. Иллюстрация работы фильтров в одномерной свёрточной нейронной сети*

Функция Conv1D позволяет извлекать признаки из одномерных последовательностей данных, таких как тексты, поэтому она широко используется в нейронных сетях для обработки текстовых данных. В частности, в модели, которую мы привели, свёрточный слой используется для извлечения признаков из векторного представления слов, полученного с помощью слоя Embedding. После этого извлеченные признаки объединяются и обрабатываются полносвязными слоями для классификации текстовых данных.

Функция GlobalMaxPool1D – это слой глобальной пулинговой операции максимума, который применяется к входным данным одномерной свёрточной нейронной сети. Этот слой извлекает наибольшие значения из всех фильтров свёрточного слоя, создавая одно числовое значение для каждого фильтра. Данный слой вычисляет максимальное значение вдоль каждого фильтра, возвращая матрицу, в которой каждый элемент – это максимальное значение по всем шагам. Таким образом, этот слой уменьшает пространственные размерности, избавляясь от шумов и оставляя только наиболее важные признаки, извлеченные из сверточных фильтров. Этот метод уменьшает количество параметров модели, что может ускорить обучение и предотвратить переобучение.

Далее после обработки фильтрами свёрточных слоёв и использования функции пулинга все 3 выхода конкатинируются в один большой тензор. Следующая функция Dense – это полносвязный слой в нейронной сети. Этот слой принимает на вход тензор и вычисляет линейную комбинацию его входных значений с матрицей весов и смещением, а затем применяет нелинейную функцию активации к полученным значениям. В этой функции определяется количество нейронов (или размерность выходного пространства) в полносвязном слое. Также для нелинейности данных используется уже выше упомянутая функция активации ReLU (Rectified Linear Unit), которая определяется как ReLU(x) = max(0, x), где x – входной тензор. Использование функции активации ReLU позволяет устранить отрицательные значения и внести нелинейность в выходные значения полносвязного слоя. Это помогает модели изучать более сложные функции, чем просто линейную комбинацию входов.

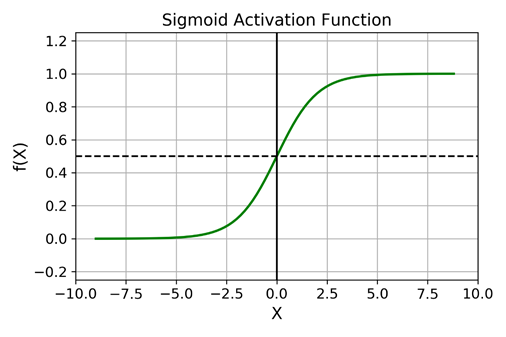
После этого используется функция Dropout, которая представляет собой слой, который применяет дропаут к предыдущему слою модели. Дропаут является техникой регуляризации, которая помогает избежать переобучения модели. Дропаут – случайное исключение (зануление) некоторых выходных значений из предыдущего слоя в процессе обучения. В каждой эпохе обучения слой Dropout выбирает случайным образом некоторые выходные значения и зануляет их в соответствии с заданным коэффициентом dropout\_rate (гиперпараметр[[23]](#footnote-23)).



*Рис. 14. Иллюстрация работы функции dropout в нейронной сети*

Таким образом, слой Dropout уменьшает связность между слоями модели, что способствует избежанию переобучения. Например, если dropout\_rate равен 0.1, то в каждой эпохе обучения 10% входных значений будут занулены. Графическое представление работы данной функции представлено на рис. 14.

Далее снова используется функция Dense, но с единственным выходным нейроном и функцией активации sigmoid. Формула сигмоидной функции представлена в виде . Данный слой является выходом нашей нейронной сети. Сигмоидная функция активации принимает любое вещественное значение и преобразует его в диапазон между 0 и 1. Она часто используется в задачах бинарной классификации, где единственный выход модели должен представлять вероятность принадлежности объекта к одному из двух классов (в нашем случае – к классу мошенничества). Однако ответ нейронной сети не является вероятностью в чистом виде. Увеличение значения и стремление его к единице лишь отражает уверенность модели в том, что заявление является мошенническим. График сигмоидной фкнции представлен на рис. 15.



*Рис. 15. График сигмоидной функции активации*

Далее стоит немного рассказать про то, как происходит обучение нейронной сети. В нашей модели для этого используется оптимизатор Adam, который является методом стохастической оптимизации, используемым для обновления весов в нейронных сетях в процессе обучения. Этот метод комбинирует идеи методов градиентного спуска и RMSProp. В нашей модели гиперпараметр скорости обучения для оптимизатора равен 0,00005. Он контролирует, как быстро будут обновляться веса в нейронной сети на каждой итерации обучения. Это длина шага градиентного спуска, если увеличить значение, то мы можем не достичь локального или глобального минимума функции потерь (функции ошибок, loss function), а если уменьшить, то модель будет сходиться дольше и может не дойти до локального или глобального минимума.

Функция потерь в нашей нейронной сети – бинарная кросс энтропия, является стандартной функцией потерь для бинарной классификации. Она используется для измерения разницы между предсказанными и настоящими значениями. Задаётся она формулой, представленной ниже:

, (2.4)

где – количество наблюдений, – верное значение целевой переменной для *i*-го объекта, – предсказанное значение вероятности *i*-го объекта.

После использования нейронной сети, как уже было сказано ранее, мы получим ответы после сигмоидной функции активации, похожие на вероятность. Далее необходимо выбрать граничное значение (threshold), после которого мы считаем, что операция является мошеннической. Выбор этого граничного знаечния зависит от конкретной задачи. Подробнее про выбор данного значения будет рассказано в следующей главе.

К нашим исходным данным добавим новую переменную – вероятность мошеннического заявления, которую мы получили с помощью свёрточной нейронной сети.

Так как нам всё же нужно выбрать граничное значение, пройдёмся по всем значениям от 0 до 1 с шагом 0,001 (1000 итераций) и выберем то число, при котором будет максимизироваться F1 метрика (в следующей главе будет описание данной метрики оценки качества моделей). После этого создадим новый признак, который будет содержать уже бинарные ответы свёрточной нейронной сети, и так же добавим его в исходные данные.

Далее начинается заключительный этап, где мы получим итоговые предсказания. Делать мы это будем, используя разные модели (как новые модели машинного обучения, так и старые статистические модели). На вход моделям будем подавать преобразованные нами ранее данные с тремя новыми признаками (результаты кластеризации и 2 выхода нейронной сети). Будут использованы следующие модели бинарной классификации: логистическая регрессия (LR), метод опорных векторов (SVM), классификатор линейного дискриминантного анализа (LDA), классификатор случайного леса (Random forest classifier) и классификатор градиентного бустинга (Gradient boosting classifier). Перве 3 модели – старые методы машинного обучения, последние 2 – новые. Подробное описание работы данных моделей было представлено в первой главе.

Также стоит отметить, что в используемых нами моделях существует большое количество гиперпараметров, от которых зависит их предсказательная способность. Чтобы подобрать оптимальные для решения нашей задачи значения гиперпараметров, мы будем использовать встроенную в Python функцию GridSearchCV. Grid search CV (cross-validation) – это метод оптимизации параметров алгоритмов машинного обучения. Он позволяет автоматически находить наилучшие значения гиперпараметров (например, параметры регуляризации или коэффициенты модели), используя перебор значений параметров по сетке. Задается набор значений параметров, которые необходимо перебрать, и строится сетка из возможных комбинаций параметров. Затем для каждой комбинации запускается обучение модели на тренировочных данных, оценивается ее качество на тестовых данных, и фиксируется результат. По завершении обучения модели для всех возможных комбинаций параметров, выбирается та, для которой оценка качества была наилучшей. Эта функция включает в себя метод кросс-валидации, который позволяет оценить качество модели на нескольких разбиениях данных, уменьшая вероятность переобучения и повышая обобщающую способность модели. Параметр k отвечает за то, на сколько подвыборок будут разбиты исходные данные.



*Рис. 16. Схема работы метода кросс-валидации*

После применения всех вышеперечисленных моделей мы получаем итоговые бинарные предсказания целевой переменной. Наши модели состоят из трёх разнообразных моделей (кластеризация с помощью алгоритма k-means, классификация с помощью свёрточной нейронной сети, основанная на текстовых данных, и одна из пяти перечисленных выше моделей), является их композицией. Помимо оценок результатов композиций моделей будут ещё оценены отдельно результаты нейронной сети и результаты лучшего из пяти алгоритмов (LR, SVM, LDA, Random Forest, Gradient Boosting) без использования ответов нейронной сети (всего 7 моделей). В следующей главе будут разобраны результаты моделирования с описанием всех использованных метрик оценки их качества.

**Глава 3. Результаты моделирования и сравнение разных моделей**

**3. 1. Объяснение оценочных показателей, используемых для измерения эффективности методов классификации**

Так как в обеих выборках (страхование жизни и не-жизни) мы имеем дисбаланс между классами (только 8,9% заявлений в страховании жизни и 17,05% в страховании не-жизни являются мошенническими), то нам необходимо использовать метрики, устойчивые к несбалансированности. Если смотреть только на долю правильных ответов (Accuracy), то в случае договоров страхования жизни при константном прогнозе, равном нулю, мы будем иметь значение около 91,1%, что является достаточно высоким знаечнием, но не отражает качество модели.

Одной из основных метрик, которой мы будем пользоваться, является ROC AUC. Метрика AUC (Area Under Curve) используется для оценки качества бинарной классификации. Она является площадью под кривой ROC (Receiver Operating Characteristic), которая показывает отношение между количеством верно-положительных и ложно-положительных результатов. Чем выше значение AUC, тем лучше производительность модели. Особенность данной метрики (и других интегральных метрик) состоит в том, что нет необходимости выбирать какое-то пороговое значение «вероятности», после которого мы считаем, что целевая переменная равна единице. Вместо этого интегральные метрики проставляют n+1 пороговых значений «вероятности», где n – размер выборки. После в каждом пороге считаются FPR и TPR и порог снова изменяется. Далее все получившиеся точки рисуются на графике и соединяются линией, а площадь под получившейся ломанной – значение метрики. Считается, что если значение метрики стремится к 0,5 при n (размер выборки) стремящемся к бесконечности, то такой классификатор является случайным (случайно проставляется 0 или 1). Если значение меньше 0,5, то классификатор справляется хуже, чем модель случайных предсказаний. При этом TPR (True Positive Rate) и FPR (False Positive Rate) – это две важные метрики, используемые для оценки производительности моделей машинного обучения в задачах классификации, а также необходимые для построения ROC AUC кривой. TPR – это доля правильно предсказанных положительных результатов (истинно положительных) от общего числа положительных результатов (положительные результаты – единицы). Формально TPR вычисляется следующим образом: TPR = TP / (TP + FN), где TP (True Positive) – количество истинно положительных результатов, которые модель предсказала верно, а FN (False Negative) – количество ложно отрицательных результатов, которые модель предсказала неверно. FPR – это, в свою очередь, доля неправильно предсказанных положительных результатов (ложно положительных) от общего числа отрицательных результатов (отрицательные результаты – нули). Формально FPR вычисляется следующим образом: FPR = FP / (FP + TN), где FP (False Positive) – количество ложно положительных результатов, которые модель предсказала неверно, а TN (True Negative) – количество истинно отрицательных результатов, которые модель предсказала верно. Метрика ROC AUC будет использоваться во всех моделях, так как она является устойчивой к несбалансированной выборке.

Часто модели машинного обучения выдают подобие вероятности, например, как наша свёрточная нейронная сеть. Исследователю, исходя из поставленной задачи, нужно правильно выбирать граничное значение, после которого модель вместо нулей начинает предсказывать единицы. Если оно будет достаточно большим, например, 0,95, то малое количество на самом деле мошеннических заявлений будут классифицированы, как мошеннические, однако среди тех заявлений, которые всё же будут классифицированы как мошеннические, будет высокая доля верных предсказаний. Метрика, которая будет максимизирована в таком случае, называется Precision (точность) и считается как доля объектов, отнесенных моделью к положительному классу, которые действительно принадлежат этому классу. Формула для расчета: Precision = TP / (TP + FP). Если же граничное значение вероятности будет слишком низким, например, 0,05, то подавляющее большинство действительно мошеннических заявлений будут обнаружены моделью, однако вместе с этим будет классифицирована как мошеннические операции большая доля тех заявлений, которые на самом деле не являются таковыми. Метрика, которая будет максимизирована в этом случае, называется Recall (полнота) и считается как доля объектов положительного класса, которые были верно классифицированы моделью. Recall показывает, какую долю объектов положительного класса модель способна обнаружить. Формула для расчета: Recall = TP / (TP + FN). Precision и Recall являются взаимозависимыми метриками: увеличение одной метрики может привести к уменьшению другой. Например, если модель сильно увеличивает количество объектов, отнесенных к положительному классу (единице), то Precision может снизиться, потому что большинство из них на самом деле принадлежит к отрицательному классу, но Recall может увеличиться, потому что модель лучше обнаруживает объекты положительного класса (большая их часть попадёт за пределы вероятностной границы). Обычно, если не считать отдельные узконаправленные задачи, берут какой-то компромисс, когда и одна, и другая метрика остаются на приемлемом уровне.

F1 метрика также является достаточно известной и популярной метрикой, которая тоже используется в оценке качества моделей в условиях несбалансированной выборки. Она представляет собой гармоническое среднее между точностью (precision) и полнотой (recall). F1 метрика важна, когда точность и полнота имеют для задачи одинаковое значение. В нашей задаче оба показателя являются критически важными для измерения качества модели. Формула для расчёта выглядит так: .

Ещё одним способом оценить качество бинарной классификации является построение матрицы ошибок. Матрица ошибок (confusion matrix) – это инструмент, используемый для измерения производительности алгоритма классификации. В табл. 3 представлена схема матрицы ошибок, этот способ оценки качества моделей также будет использован в данной работе. Все рисунки матриц ошибок разных моделей будут отражены в приложениях.

*Таблица 3*

*Матрица ошибок в модели бинарной классификации*

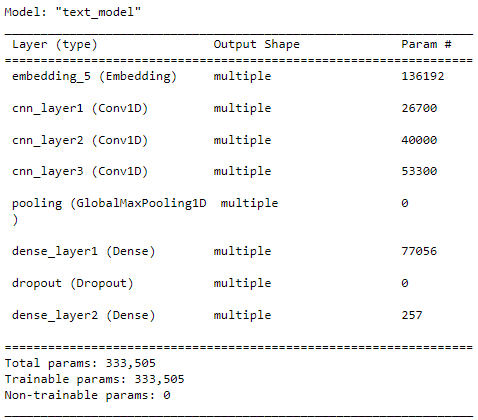
|  |  |  |  |
| --- | --- | --- | --- |
|  |  | *Предсказанные значения* | |
|  | Истина / Предсказание | Отрицательный (0) | Положительный (1) |
| *Истинные значения* | Отрицательный (0) | Истинно-отрицательный (TN) | Ложно-положительный (FP) |
| Положительный (1) | Ложно-отрицательный (FN) | Истинно-положительный (TP) |

Таким образом, у нас есть множество способов оценки качества построенных нами моделей, поэтому мы можем переходить к результатам исследования.

**3.2. Представление результатов использования различных методов машинного обучения на данных страхования жизни, сравнение эффективности моделей**

Так как наша большая выборка страховых заявлений разделена на 2 подвыборки, то сначала рассмотрим результаты решения задачи детекции мошеннических заявлений в договорах страхования жизни, а потом – страхования не-жизни. Все оценки были проведены на тестовой выборке (по 25% от каждой).

Первой промежуточной моделью, которая была построена в нашей работе, была свёрточная нейронная сеть. Общая информация по ней представлена ниже на рис.17. Наша нейронная сеть имеет 333505 обучаемых параметров, которые будут изменяться по мере обучения модели после прохождения по каждому батчу в нашей выборке (с помощью метода обратного распространения ошибки и градиентного спуска).



*Рис. 17. Общая описательная информация свёрточной нейронной сети, обученной на данных страхования жизни*

Теперь перейдём метрикам. Сначала воспользуемся встроенной функцией classification\_report в Python, которая предоставляет метрики оценки качества модели классификации на основе фактических и предсказанных значений (табл. 4). Эти метрики включают в себя долю правильных ответов (accuracy), точность (precision), полноту (recall) и F1 метрику (F1-score). Они основаны на матрице ошибок (confusion matrix), которая отображает количество верных и ложных прогнозов каждого класса.

Как мы можем заметить, у нас высокое значение Precision (0,98), и достаточно большое значение Recall (0,8) и, как следствие, F1 меры (0,88). Данные результаты говорят нам о том, что из тех наблюдений, которые нейронная сеть признала мошенническими, 98% действительно оказались мошенническими, однако нейронная сеть обнаружила 80% от всех мошеннических заявлений. Несмотря на это, Accuracy (доля верных ответов) равна 98%, что немного лучше, чем константный прогноз. Посмотрев на матрицу ошибок (см. Приложение 1), мы можем сделать вывод, что нейронная сеть хорошо справляется с детекцией немошеннических заявлений (ошибается в 6 наблюдениях), однако в мошеннических операциях модель не обнаружила 20% наблюдений (67).

*Таблица 4*

*Отчёт по классификации в страховании жизни, свёрточная нейронная сеть*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-мера | Количество |
| 0 | 0,98 | 1 | 0,99 | 3009 |
| **1** | **0,98** | **0,8** | **0,88** | **327** |
| Среднее | 0,98 | 0,9 | 0,93 | 3336 |
| Средневзвешенное | 0,98 | 0,98 | 0,98 | 3336 |

ROC кривая выглядит адекватно(см. Приложение 16). Значение ROC AUC равно 0,97, что является отличным результатом, учитывая, что предсказания были сделаны только на текстовых данных.

Если учесть значения всех метрик в совокупности, то можно сделать вывод, что мощности свёрточной нейронной сети, построенной только на текстовых данных, достаточно для решения задачи детекции страховых мошеннических заявлений на высоком уровне.

После добавления трёх новых признаков в наши данные (результаты кластеризации и 2 ответа нейронной сети) можно оценить работу каждой из описанных выше моделей.

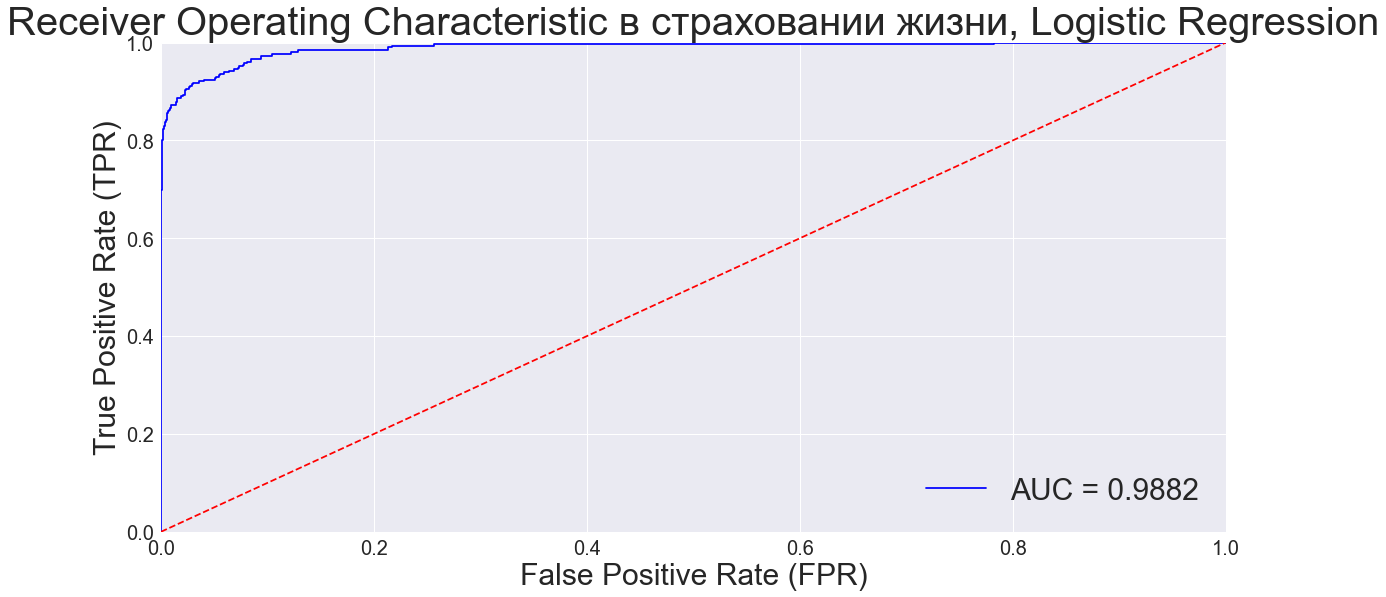
Ниже представлен отчёт классификации после использования логистической регрессии (табл. 5). Результаты моделирования не сильно отличаются от ответов нейронной сети. Значение Precision стало немного меньше (0,96), но значение Recall увеличилось (0,83). F1 мера тоже выросла на 0,01 (0,89). Данные результаты говорят нам о том, что из тех наблюдений, которые логистическая регрессия признала мошенническими, 96% действительно оказались мошенническими. При этом логистическая регрессия классифицировала как мошеннические 83% от всех мошеннических заявлений. Доля верных ответов так же равна 98%.

*Таблица 5*

*Отчёт по классификации в страховании жизни, логистическая регрессия*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-мера | Количество |
| 0 | 0,98 | 1 | 0,99 | 3041 |
| **1** | **0,96** | **0,83** | **0,89** | **328** |
| Среднее | 0,97 | 0,91 | 0,94 | 3369 |
| Средневзвешенное | 0,98 | 0,98 | 0,98 | 3369 |

Проанализировав матрицу ошибок (см. Приложение 2) , мы можем сделать вывод, что добавление логистической регрессии немного ухудшило предсказательную способность нулевых меток (11 ошибочно классифицированных заявлений), однако в то же время улучшило предсказательную способность единичных (55 ошибок). Как уже было сказано ранее, модели классификации часто решают проблему trade-off[[24]](#footnote-24) между максимизацией Precision и Recall.



*Рис. 18. График ROC кривой в страховании жизни, логистическая регрессия*

Далее посмотрим на график ROC кривой (рис. 18). Красная прерывистая линия на этом графике отражает примерное поведение модели со случайными прогнозами. Значение ROC AUC равно 0,9882, что немного выше, чем в предсказаниях нейронной сети по текстовым данным. Все остальные графики ROC кривой будут отражены в приложении.

Логистическая регрессия хорошо справилась с задачей. При этом данный алгоритм обучается быстрее остальных.

Следующий метод машинного обучения, результаты которого мы будем рассматривать – метод опорных векторов.

*Таблица 6*

*Отчёт по классификации в страховании жизни, метод опорных векторов*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-мера | Количество |
| 0 | 0,98 | 0,99 | 0,99 | 3041 |
| **1** | **0,95** | **0,84** | **0,89** | **328** |
| Среднее | 0,96 | 0,92 | 0,94 | 3369 |
| Средневзвешенное | 0,98 | 0,98 | 0,98 | 3369 |

Исходя из отчёта, представленного в табл. 6, можно предположить, что данный алгоритм справляется с задачей примерно на том же уровне, что и логистическая регрессия. Значение Precision равно 0,95, что меньше, чем у логистической регрессии на 0,01, значение Recall равно 0,84, что больше на 0,01, чем у логистической регресии. F1 метрика осталась на том же уровне (0,89). Данные результаты говорят нам о том, что из тех наблюдений, которые метод опорных векторов признал мошенническими, 95% действительно оказались мошенническими. Общая доля обнаруженных мошеннических заявлений равна 84%. Доля верных ответов снова равна 98%. Матрица ошибок метода опорных векторов (см. Приложение 3) практически не отличима от матрицы ошибок логистической регрессии. При этом значение ROC AUC равно 0,9649 (см. Приложение 17), что немного меньше, чем у логистической регрессии и свёрточной нейронной сети.

Последним из классических статистических методов яввляется линейный дискриминантный анализ (LDA). Помимо результатов данная модель предоставляет возможность посмотреть на коэффициенты, на которые домножаются значения признаков. Если добавить новое наблюдение (заявление), то по этим коэффициентам можно посчитать предсказание модели. Если оно будет ниже нуля, то заявление не мошенническое, иначе мошенническое. При этом посчитанное значение в абсолютном виде будет демонстрировать степень уверенности модели.

,

где соответственно равны столбцам переменных Канал продаж, Серия договора, Валюта ответственности, Тип заявителя, Тип случая, Место случая, Регион, Группа инвалидности, CC main program, Risk code, Доп. доход, Страховая сумма по риску, в валюте ответственности, Оплаченные дни нетрудоспособности, ЗУ, % от страховой суммы , РЗУ, в валюте ответственности, Доля выгодоприобретателя, СС main programm, Разница между датой вступления договора в силу и наступлением страхового случая, Кластер, txt\_pred\_prob, txt\_pred\_num.

В табл. 7 можно заметить результаты, очень схожие со всеми предыдущими моделями. Если модель дискриминантного анализа предсказывала мошенническое заявление, то в 96% случаев она делала это корректно. Если же немошенническое – в 98%. Всего данный классификатор обнаружил 81% из всех мошеннических заявлений, а F1 мера равна 0,88. Общая доля верных ответов, как и в предыдущих моделях, равна 98%. Кажется, что модель идеально классифицировала все немошеннические операции (Recall при нулевой метке равен 1), однако это не так (дело в округлении, 10 немошеннических заявлений были классифицированы как мошеннические). Данная информация представлена в матрице ошибок модели дискриминантного анализа (см. Приложение 4). Далее на вероятностных ответах модели был построен график ROC кривой (см. Приложение 18). Значение ROC AUC равно 0,9818, что является отличным результатом.

*Таблица 7*

*Отчёт по классификации в страховании жизни, классификатор линейного дискриминантного анализа*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-мера | Количество |
| 0 | 0,98 | 1 | 0,99 | 3041 |
| **1** | **0,96** | **0,81** | **0,88** | **328** |
| Среднее | 0,97 | 0,90 | 0,93 | 3369 |
| Средневзвешенное | 0,98 | 0,98 | 0,98 | 3369 |

В целом, все 3 классические модели статистического анализа в связке с ответами алгоритма k-means и свёрточной нейронной сети обладают отличными предсказательными способностями.

Далее будем рассматривать результаты современных моделей машинного обучения. Первая из них – градиентный бустинг. Ниже в табл. 8 представлен отчёт по классификации для данного метода машинного обучения.

*Таблица 8*

*Отчёт по классификации в страховании жизни, градиентный бустинг*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-мера | Количество |
| 0 | 0,99 | 0,99 | 0,99 | 3041 |
| **1** | **0,93** | **0,88** | **0,91** | **328** |
| Среднее | 0,96 | 0,94 | 0,95 | 3369 |
| Средневзвешенное | 0,98 | 0,98 | 0,98 | 3369 |

Значение Precision немного меньше, чем у всех классических моделей (0,93), однако значение Recall выше (0,88), соответственно, F1 мера тоже выше (0,91). Представленный отчёт говорит нам о том, что из тех наблюдений, которые градиентный бустинг признал мошенническими, 93% действительно оказались мошенническими. Общая доля обнаруженных мошеннических заявлений теперь равна 88%, а доля верных ответов снова равна 98%. Несмотря на то, что значение Precision меньше, чем в результатах классических моделей, F1 мера, которая является комбинацией Precision и Recall, оказалась наибольшей, следовательно, можно предположить, что градиентный бустинг станет лучшей моделью для решения задачи детекции мошеннических заявлений в данных страхования жизни. Матрица ошибок немного отличается от предыдущих результатов (см. Приложение 5). Случаев, в которых заявление мошенническое, а модель классифицирует их как не мошеннические, стало меньше (38 заявлений). При этом количество ложно-положительных результатов стало немного больше (21 заявление). График ROC кривой модели градиентного бустинга стремится к образу прямого угла (см. Приложение 19). Площадь под кривой равна 0,9919, что намного выше, чем в методе опорных векторов, и немного выше, чем в логистической регрессии и дискриминантном анализе.

Последняя модель машинного обучения, результаты которой мы будем оценивать – случайный лес. Для начала снова посмотрим на отчёт по классификации (табл. 9).

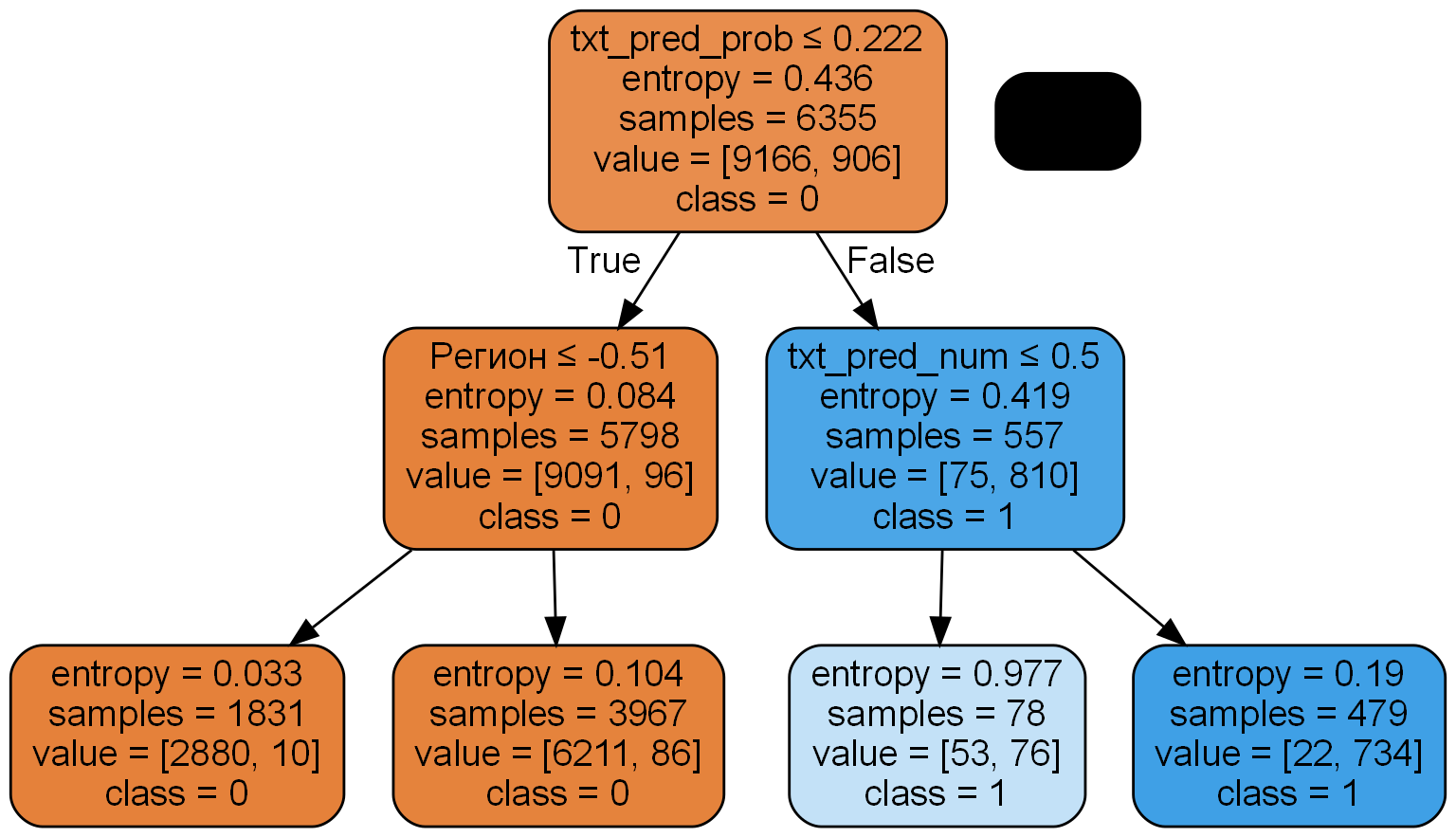
*Таблица 9*

*Отчёт по классификации в страховании жизни, случайный лес*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-мера | Количество |
| 0 | 0,98 | 1 | 0,99 | 3041 |
| **1** | **0,98** | **0,81** | **0,88** | **328** |
| Среднее | 0,98 | 0,9 | 0,94 | 3369 |
| Средневзвешенное | 0,98 | 0,98 | 0,98 | 3369 |

Результаты данной модели немного отличаются от предыдущих. Precision равен 0,98, что является максимальным значением метрики, если оценивать все предыдущие модели (только у CNN такое же значение). При этом Recall равен 0,81, что всего на 0,01 больше, чем у модели CNN, а F1 мера равна 0,88. Доля правильных ответов равна 98%. Данные результаты говорят нам о том, что из тех наблюдений, которые случайный лес признал мошенническими, 98% действительно оказались мошенническими. Но общая доля обнаруженных мошеннических заявлений равна 81%, что сопоставимо с результатами свёрточной нейронной сети.

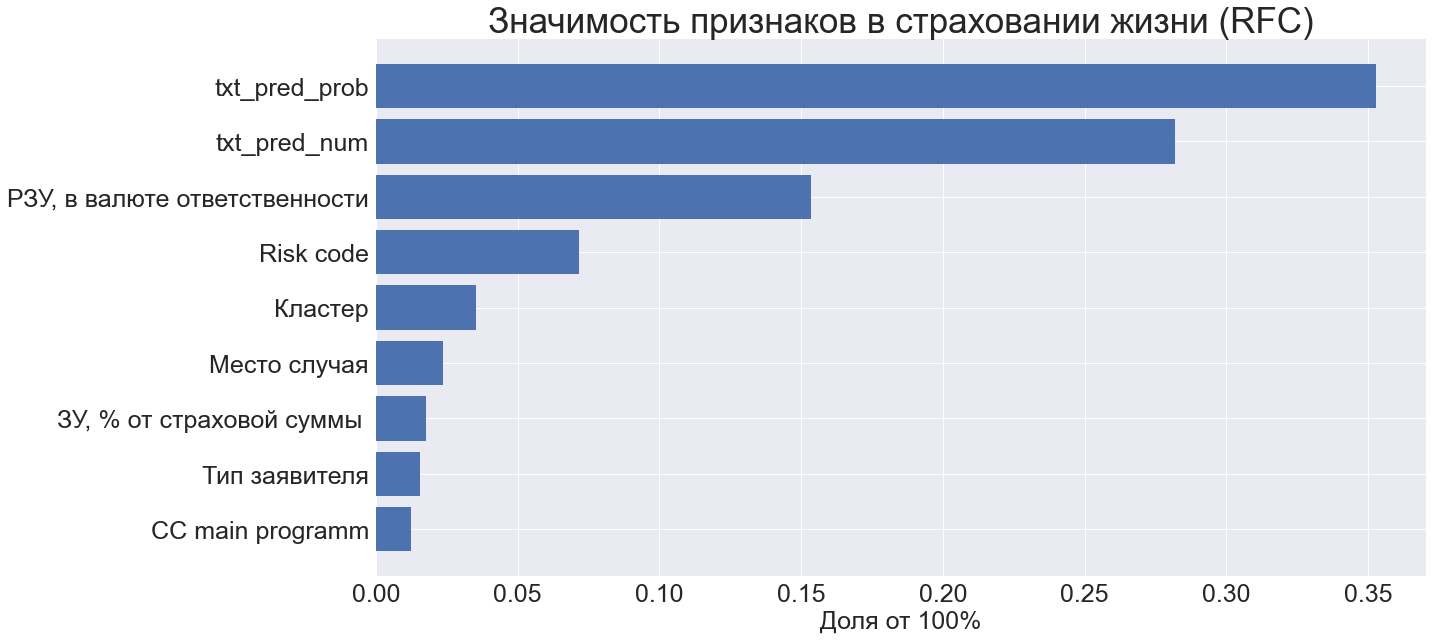
В матрице ошибок (см. Приложение 6) случайный лес делает 6 ложно-положительных ответов (то же значение у CNN), 63 ложно-отрицательных ответов (у CNN было 67). Значение ROC AUC в модели случайного леса является сопоставимым со значением в градиентном бустинге, оно равно 0,9913 (см. Приложение 20).

**

*Рис. 19. Схема одного из решающих деревьев, использованных в Random Forest*

Также мы можем посмотреть на одно из решающих деревьев, использованных в данном методе. Для примера взято дерево с глубиной, равной 2. На рис. 19 представлено одно из них. Такую же схему можно построить и в градиентном бустинге, так как он тоже использует деревья решений. Итоговая энтропия, взвешенная на количество наблюдений, всегда будет меньше, чем начальное значение энтропии.

Перед построением общей таблицы с результатами метрик, полученных разными моделями, мы можем взглянуть на значимость признаков в задаче классификации для случайного леса. Как уже было сказано в прошлой главе, у случайного леса есть возможность посмотреть данный показатель. Он основан на том, как часто решающие деревья разбивали выборку на две подвыборки, опираясь на каждый из признаков. Чем больше было этих распределение по определённому признаку, тем более данный признак является значимым для модели, так как его разбиение минимизировало энтропию. Данная гистограмма представлена на рис. 20.



*Рис. 20. Гистограмма значимости признаков в страховании жизни, случайный лес*

Исходя из полученной нами гистограммы, можно сделать вывод, что наиболее сильное влияние на конечные предсказания внесли две переменные: txt\_pred\_prob, которая является вероятностным выходом нейронной сети, и txt\_pred\_num, которая является бинарным выходом нейронной сети. Помимо этого пятым по значимости признаком для случайного леса оказались метки кластера, которые мы тоже смоделировали ранее.

Чтобы дополнительно проверить, насколько сильно ответы нейронной сети влияют на качество предсказаний моделей, возьмём модель градиентного бустинга и обучим на данных без двух добавленных переменных.

Отчёт по классификации представлен в табл. 10. Значения метрик всё ещё приемлемые, однако они стали хуже. Precision без текстовых данных увеличился, равен 0,98 (ранее 0,93). Но Recall уменьшился, значение равно 0,35 (ранее 0,88), соответственно, F1 мера уменьшилась тоже (0,51 против ранее 0,91), а общая доля верных ответов равна 0,94 (ранее 0,98).

*Таблица 10*

*Отчёт по классификации в страховании жизни, градиентный бустинг без предсказаний нейронной сети*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-мера | Количество |
| 0 | 0,93 | 1 | 0,97 | 3041 |
| **1** | **0,98** | **0,35** | **0,51** | **328** |
| Среднее | 0,96 | 0,67 | 0,74 | 3369 |
| Средневзвешенное | 0,94 | 0,94 | 0,92 | 3369 |

Если обратиться к матрице ошибок (см. Приложение 7), то можно отметить, что модель практически не делает ложно-положительных предсказаний (их всего 2), однако делает очень много ложно-отрицательных предсказаний (214 мошеннических заявлений). При этом уменьшилось общее число верно предсказанных мошеннических заявлений. Как ни странно, значение ROC AUC осталось на достаточно высоком уровне и равно 0,9672. Несмотря на это, оно всё же меньше, чем на данных с ответами нейронной сети (см. Приложение 21).

Подводя итог результатов оценки модели градиентного бустинга с выходами нейронной сети и без, можно сделать вывод, что даже такие однотипные текстовые данные, как в нашей выборке, помогают моделям машинного обучения лучше решать задачи бинарной классификации, а в частности – задачу детекции страховых мошеннических заявлений. Этот вывод подтверждается результатами оценки влияния признаков на предсказание целевой переменной в модели случайного леса.

После получения результатов построим общую таблицу для выбора модели с лучшей предсказательной способностью, где CNN – свёрточная нейронная сеть, построенная только на текстовых данных, LR – логистическая регрессия, SVM – метод опорных векторов, LDA – линейный дискриминантный анализ, GBC – градиентный бустинг, RFC – случайный лес, GBC без текста – градиентный бустинг без ответов нейронной сети (табл. 11). Так как Accuracy не отражает реальное качество моделей при несбалансированной выборке, мы будем опираться на другие метрики. Precision и Recall, как уже было написано выше, являются противоположно направленными метриками (когда одна растёт, другая уменьшается), поэтому мы будем использовать F1 меру и ROC AUC как решающие метрики, так как они являются функциями от Precision и Recall.

*Таблица 11*

*Таблица с результатами моделей на данных страхования жизни*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Метод | Accuracy | Precision | Recall | F1 мера | ROC AUC | Ответы CNN как признак | Результаты k-means как признак |
| CNN | 0,98 | 0,98 | 0,80 | 0,88 | 0,97 | - | Нет |
| LR | 0,98 | 0,96 | 0,83 | 0,89 | 0,9882 | Да | Да |
| SVM | 0,98 | 0,95 | 0,84 | 0,89 | 0,9649 | Да | Да |
| LDA | 0,98 | 0,96 | 0,81 | 0,88 | 0,9818 | Да | Да |
| GBC | 0,98 | 0,93 | 0,88 | 0,91 | 0,9919 | Да | Да |
| RFC | 0,98 | 0,98 | 0,81 | 0,88 | 0,9913 | Да | Да |
| GBC без текста | 0,94 | 0,98 | 0,35 | 0,51 | 0,9672 | Нет | Да |

Наихудшей моделью оказался классификатор градиентного бустинга, который обучался без ответов нейронной сети. Его F1 мера на тестовой выборке равна 0,51, что является низким показателем (из-за низкого значения Recall), однако значение ROC AUC высокое (0,9672).

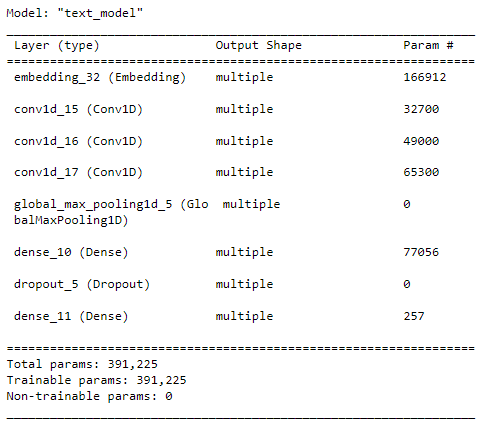
Худшими моделями из тех, которые использовали ответы нейронной сети, по совокупности значений метрик оказались линейный дискриминантный анализ и свёрточная нейронная сеть. Однако стоит отметить, что нейронная сеть обучалась лишь на текстовых данных, а классификатор линейного дискриминантного анализа – на всех (в том числе и ответах нейронной сети). Более того, значения ROC AUC и F1 метрики очень велики и схожи с результатами других моделей. Классические методы машинного обучения, такие как логистическая регрессия и метод опорных векторов, показали практически идентичные результаты. F1 мера одинаковая, однако ROC AUC немного больше у логистической регрессии (0,9882 против 0,9649). Дело в том, что ROC AUC показывает, насколько хорошо модель может различать положительные и отрицательные классы при разных порогах классификации, а F1 считается только по одному конкретному значению порога (как и другие метрики). Поэтому в конкретном значении порога, которое было задано нами ранее путём максимизации F1 меры, метод опорных векторов справляется с задачей так же, как и логистическая регрессия, однако при определённой специфике задачи и других условиях логистическая регрессия могла бы справиться лучше (особенно при низких значениях порога, близких к 0). Более того, очевидный плюс логистической регрессии заключается в том, что она работает быстрее, чем метод опорных векторов. Если рассматривать современные методы машинного обучения, такие как градиентный бустинг и случайный лес, то можно сделать вывод, что первый справился с классификацией лучше. Несмотря на практически одинаковые значения ROC AUC (0,9919 у GBC против 0,9913 у RFC), F1 мера отличается на 0,03 (0,91 у GBC против 0,88 у RFC). ROC AUC может быть высоким, потому что он учитывает истинно-отрицательные значения (True Negative Rate), количество которых велико в случае дисбаланса классов. Если бы мы брали взвешенный F1, то получили бы одинаковые результаты (0,98). Однако в нашей задаче главное – детекция мошеннических заявлений для последующей тщательной проверки, поэтому в нашем случае градиентный бустинг оказался лучшим не только среди современных моделей, но и среди всех моделей в целом.

В общем, все модели справились с задачей классификации, наилучшей оказалась модель градиентного бустинга, относительно новый метод машинного обучения. Модели могли бы справиться ещё лучше, если бы было больше значимых числовых признаков.

**3.3. Представление результатов использования различных методов машинного обучения на данных страхования не-жизни, сравнение эффективности моделей**

Далее рассмотрим, как те же самые модели будут справляться на данных страхования не-жизни. Эта выборка в 3 раза больше предыдущей, поэтому и время обучения моделей было больше. Ниже на рис. 21 представлена информация о нейронной сети.

Архитектура нейронной сети осталась такой же, как и ранее. Однако из-за того, что максимальная длина предложения в это выборке больше, количество параметров, которые придётся обучать нейронной сети, увеличилось и теперь равно 391225.



*Рис. 21. Общая описательная информация свёрточной нейронной сети, обученной на данных страхования не-жизни*

Начнём оценку качества классификации свёрточной нейронной сети с отчёта, представленного в табл. 12. Как мы можем наблюдать, нейронная сеть выдаёт низкие значения Precision (0,55) и Recall (0,46) и, как следствие, низкое значение F1 метрики (0,5). Данные результаты говорят нам о том, что из тех наблюдений, которые нейронная сеть признала мошенническими, 55% действительно оказались мошенническими, при этом нейронная сеть обнаружила всего 46% от всех мошеннических заявлений. Доля верных ответов равна 84%. Исходя из этих результатов, можем выдвинуть гипотезу, что на второй выборке итоговое качество наших моделей будет хуже, так как ответы нейронной сети будут давать меньше информации другим моделям.

По матрице ошибок (см. Приложение 8) можно отследить, что практически половину значений, которые являются мошенническими, нейронная сеть не распознаёт. Однако причисляет к мошенническим 927 заявлений, которые не являются таковыми.

*Таблица 12*

*Отчёт по классификации в страховании не-жизни, свёрточная нейронная сеть*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-мера | Количество |
| 0 | 0,89 | 0,92 | 0,91 | 11999 |
| **1** | **0,55** | **0,46** | **0,50** | **2515** |
| Среднее | 0,72 | 0,69 | 0,7 | 14514 |
| Средневзвешенное | 0,83 | 0,84 | 0,84 | 14514 |

Далее был построен график ROC кривой (см. Приложение 22). Сравнивая с предыдущими результатами нейронной сети в страховании жизни, мы можем сделать вывод, что взаимосвязи в договорах страхования не-жизни являются более нетривиальными, ROC AUC равен 0,77 (в прошлый раз предсказания нейронной сети имели качество 0,97).

Результаты моделирования с помощью логистической регрессии представлены в табл. 13. Логистическая регрессия в паре с ответами нейронной сети распознала 41% мошеннических страховых заявлений на тестовой выборке. При этом ошиблась она лишь в 17% от числа предсказанных единиц (83% из классифицированных как мошеннические действительно являются ими). Общая доля правильных ответов равна 88%, а F1 мера равна 0,55.

Матрица ошибок (см. Приложение 9) имеет другую структуру. Ложно-положительных ответов мало относительно предсказаний нейронной сети (166), о чём свидетельствует высокое значение Precision, однако очень много ложно-отрицательных предсказаний (1145).

График ROC намного лучше (см. Приложение 23), чем у нейронной сети, площадь под ним равна 0,8512 (знаечение метрики). В целом, логистическая регрессия показала приемлемые результаты.

*Таблица 13*

*Отчёт по классификации в страховании не-жизни, логистическая регрессия*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-мера | Количество |
| 0 | 0,89 | 0,98 | 0,93 | 9332 |
| **1** | **0,83** | **0,41** | **0,55** | **1956** |
| Среднее | 0,86 | 0,7 | 0,74 | 11288 |
| Средневзвешенное | 0,88 | 0,88 | 0,87 | 11288 |

Далее проанализируем результаты работы метода опорных векторов. Сначала, как и до этого, посмотрим на отчёт.

*Таблица 14*

*Отчёт по классификации в страховании не-жизни, метод опорных векторов*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-мера | Количество |
| 0 | 0,93 | 0,99 | 0,96 | 9332 |
| **1** | **0,93** | **0,67** | **0,78** | **1956** |
| Среднее | 0,93 | 0,83 | 0,87 | 11288 |
| Средневзвешенное | 0,93 | 0,93 | 0,93 | 11288 |

В табл. 14 можно заметить, что значения всех метрик (Precision, Recall, F1 мера и Accuracy) лучше, чем у логистической регрессии. Модель выявила 67% от всех мошеннических заявлений, всего верных ответов – 93%. При этом, из классифицированных мошенническими, 93% предсказаний оказались верными. Значение F1 метрики равно 0,78, что тоже намного выше, чем у прошлой модели.

На матрице ошибок (см. Приложение 10) видно, что количество ложно-положительных ответов (случай, когда модель предсказывает мошенничество, хотя на самом деле это не так) снова сократилось (всего осталось 99 из 11288 заявлений). Более того, количество ложно-отрицательных предсказаний сократилось практически в 2 раза в сравнении с ответами логистической регрессии (650 против 1145). Очевидно, что значение ROC AUC у метода опорных векторов тоже выше, чем у логистической регрессии (см. Приложение 24).

Следующая модель, которая была построена – классификатор линейного дискриминантного анализа. В табл. 15 отражён классификационный отчёт. Результаты оказались хуже, чем у предыдущих классических моделей. Из всех мошеннических заявлений лишь 37% было распознано данной моделью. Из ответов, классифицированных как мошенничество, 86% оказались верными. Таким образом, F1 метрика данной модели равна 0,52 (на 0,02 выше, чем у предсказаний нейронной сети). Общая доля парвильных ответов равна 88%.

*Таблица 15*

*Отчёт по классификации в страховании не-жизни, классификатор линейного дискриминантного анализа*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-мера | Количество |
| 0 | 0,88 | 0,99 | 0,93 | 9332 |
| **1** | **0,86** | **0,37** | **0,52** | **1956** |
| Среднее | 0,87 | 0,68 | 0,73 | 11288 |
| Средневзвешенное | 0,88 | 0,88 | 0,86 | 11288 |

В матрице ошибок данной модели (см. Приложение 11) преобладает ошибка ложно-отрицательных предсказаний (1228 заявлений). Верных ответов же всего 728. Значение метрики AUC под этой кривой (площади) равно 0,8254 (см. Приложение 25). Дискриминантный классификатор показал худшие результаты из всех трёх классических статистических моделей.

Далее оценим результаты алгоритма, который показал лучшие результаты в анализе данных страхования жизни – градиентный бустинг.

*Таблица 16*

*Отчёт по классификации в страховании не-жизни, градиентный бустинг*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-мера | Количество |
| 0 | 0,99 | 1 | 0,99 | 9332 |
| **1** | **0,98** | **0,95** | **0,97** | **1956** |
| Среднее | 0,99 | 0,97 | 0,98 | 11288 |
| Средневзвешенное | 0,99 | 0,99 | 0,99 | 11288 |

Отчёт, представленный в табл. 16, показывает нам, что градиентный бустинг обнаружил 95% мошеннических заявлений на тестовой выборке, при этом в 98% случаев предсказания являются верными. Всего предсказано верно 99%, а F1 метрика равна 0,97.

Чтобы посмотреть, как изменилась структура предсказаний, нужно снова взглянуть на матрицу ошибок (см. Приложение 12). Существенно сократилось количество ложно-отрицательных ответов в сравнении с другими моделями (модель ошиблась в 97 наблюдениях). Количество ложно-положительных ответов равно 36, а общее число верно предсказанных мошеннических заявлений – 1859.

ROC кривая очень похожа на прямой угол и имеет наивысшее значение площади под ней (0,9856) из всех представленных выше моделей (см. Приложение 26). Градиентный бустинг снова показал отличные результаты, но остался ещё один алгоритм машинного обучения, который может оказаться лучше.

Отчёт классификатора случайного леса представлен в табл. 17. Снова случайный лес уделает большее внимание одной из метрик. Мы имеем наилучшее значение Precision (1) из всех предстиавленных моделей. Это значит, что модель либо допустила очень мало ложно-положительных ошибок, либо не допустила их совсем (это можно посмотреть на матрице ошибок, потому что значения в отчёте классификатора округлены). Однако достигается такое значение тем, что мы имеем меньшее значение Recall (0,9), следовательно, модель нашла 90% от всех мошеннических заявлений в тестовой выборке, что хуже, чем результат градиентного бустинга. При этом F1 метрика равна 0,94, что тоже является отличным результатом.

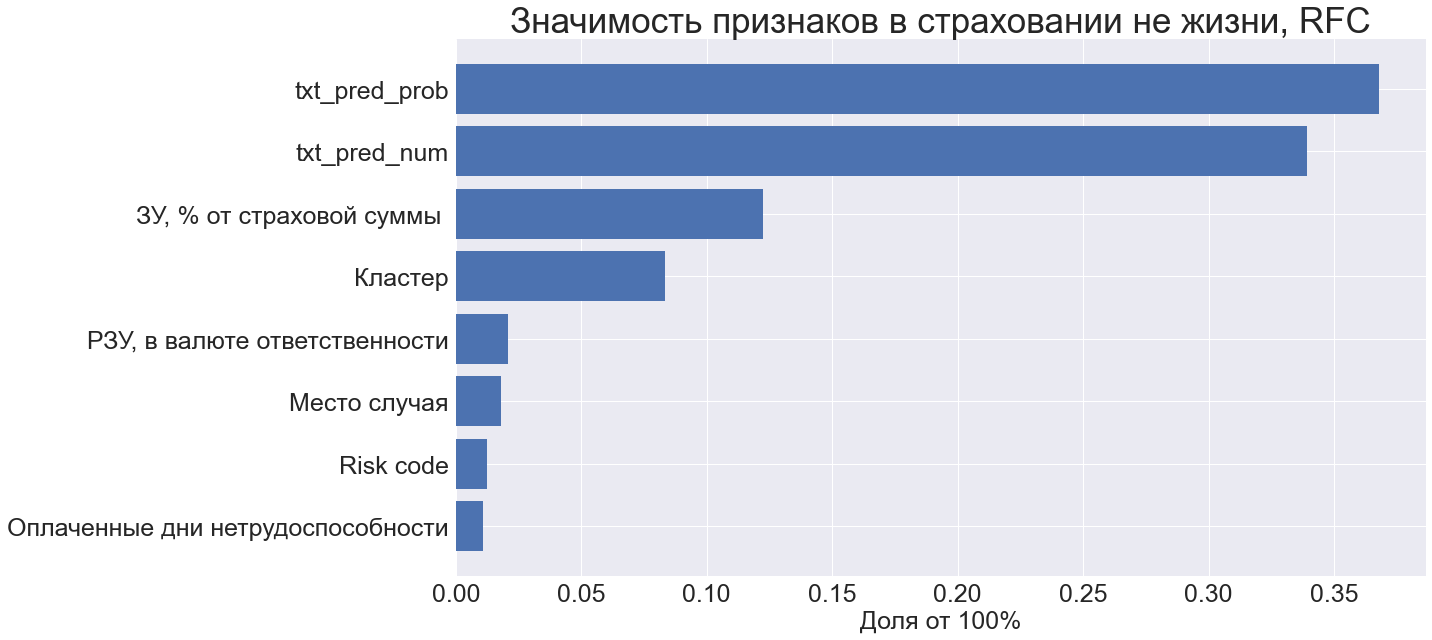
На матрице ошибок (см. Приложение 13) видно, что количество ложно-положительных ответов не равно нулю, как должно быть при Precision равном единице, однако стремится к нему (7 заявлений оказались ложно-положительными результатами). Площадь под кривой (ROC AUC) имеет высокое значение (0,9841, см. Приложение 27), однако, как мы убедились ранее, к предиктивной способности модели есть вопросы.

*Таблица 17*

*Отчёт по классификации в страховании не-жизни, случайный лес*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-мера | Количество |
| 0 | 0,98 | 1 | 0,99 | 9332 |
| **1** | **1** | **0,9** | **0,94** | **1956** |
| Среднее | 0,99 | 0,95 | 0,97 | 11288 |
| Средневзвешенное | 0,98 | 0,98 | 0,98 | 11288 |

Теперь построим гистограмму значимости признаков, но уже для портфеля страхования не-жизни. (рис. 22)



*Рис. 22. Гистограмма значимости признаков в страховании не-жизни, случайный лес*

Структура значимых признаков немного изменилась в сравнении с гистограммой на данных страхования жизни, однако предсказания нейронной сети остались двумя наиболее значимыми переменными в предсказаниях алгоритма случайного леса.

Снова проведём проверку результатов градиентного бустинга без ответов нейронной сети. Удалив переменные txt\_pred\_prob и txt\_pred\_num, мы получаем следующие результаты (табл. 18).

*Таблица 18*

*Отчёт по классификации в страховании не-жизни, градиентный бустинг без предсказаний нейронной сети*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-мера | Количество |
| 0 | 0,96 | 1 | 0,98 | 9332 |
| **1** | **1** | **0,8** | **0,89** | **1956** |
| Среднее | 0,98 | 0,9 | 0,93 | 11288 |
| Средневзвешенное | 0,97 | 0,96 | 0,96 | 11288 |

Модель верно идентифицировала 25% из всех мошеннических заявлений, при этом ошиблась в 15% положительных ответов (85% корректных). Градиентный бустинг без результатов нейронной сети имеет Accuracy, равное 0,89, а F1 мера равна 0,39 (средневзвешенная – 0,87). Без ответов нейронной сети результаты оказались посредственными. Алгоритм из 1733 мошеннических заявлений классифицирует как мошеннические всего 511, из которых количество верных предсказаний равно 433 (см. Приложение 14). Значение площади под ROC кривой (см. Приложение 28) получилось выше, чем у свёрточной нейронной сети (0,8355 против 0,71). Но в обеих этих моделях по отдельности предсказания намного хуже, чем при комбинированном использовании. Следовательно, для более корректного моделирования нужны как текстовые, так и числовые данные (под числовыми имеются в виду и перекодированные категориальные признаки).

Ниже снова будет представлена таблица с результирующими метриками по каждой модели (табл. 19).

Можно отметить, что, в отличие от градиентного бустинга, который старается найти баланс между Precision и Recall метриками, случайный лес старается максимизировать первую, минимизируя при этом количество ложно-положительных ошибок. В ситуации, когда ресурсы перепроверки страхового заявления сильно ограничены, возможно, случайный лес оказался бы лучшим решением для этой задачи.

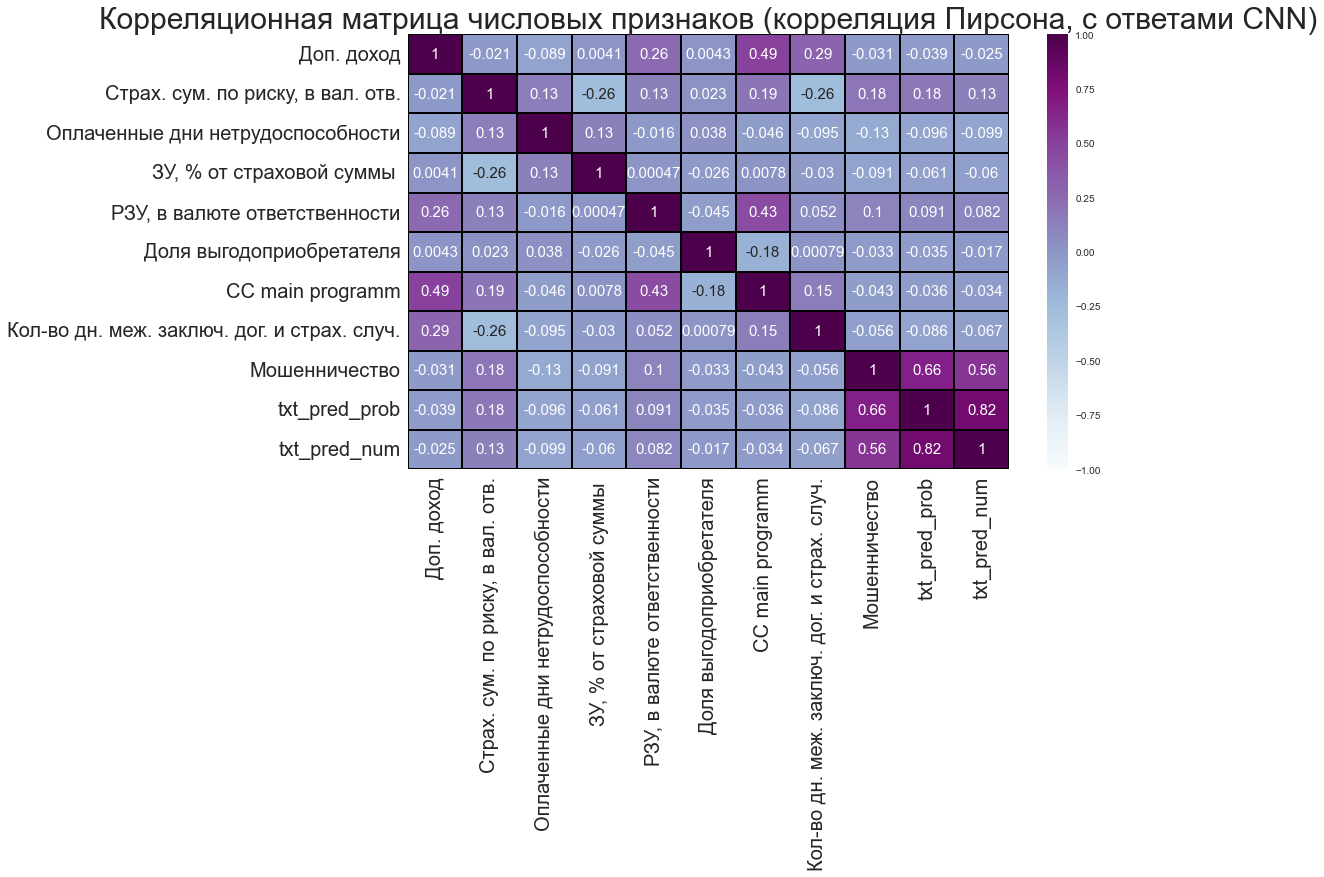
*Таблица 19*

*Таблица с результатами моделей на данных страхования не-жизни*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Метод | Accuracy | Precision | Recall | F1 мера | ROC AUC | Ответы CNN как признак | Результаты k-means как признак |
| CNN | 0,84 | 0,55 | 0,46 | 0,5 | 0,77 | - | Нет |
| LR | 0,88 | 0,83 | 0,41 | 0,55 | 0,8512 | Да | Да |
| SVM | 0,93 | 0,93 | 0,67 | 0,78 | 0,9628 | Да | Да |
| LDA | 0,88 | 0,86 | 0,37 | 0,52 | 0,8254 | Да | Да |
| GBC | **0,99** | **0,98** | **0,95** | **0,97** | **0,9856** | Да | Да |
| RFC | 0,98 | 1 | 0,9 | 0,94 | 0,9841 | Да | Да |
| GBC без текста | 0,96 | 1 | 0,8 | 0,89 | 0,977 | Нет | Да |

Может показаться, что, как и в прошлый раз, худшей моделью является свёрточная нейронная сеть (по Accuracy и ROC AUC). Однако если мы посмотрим на значение F1 меры, то худшей моделью окажется случайный лес (F1 мера равна 0,05), который идеально классифицирует те мошеннические операции, которые он обнаружил, но обнаруживает он всего 2% от всех мошеннических заявлений. В целом модель CNN является худшей, однако для нашей задачи хуже всех справился именно RFC (неадекватная модель). Если сравнивать остальные модели, то по совокупности метрик F1 и площади под ROC кривой лучшей остаётся градиентный бустинг (F1 мера равна 0,65, ROC AUC – 0,9428). Но результаты классических моделей тоже имеют высокие значения метрик, близкие к результатам градиентного бустинга. Снова метод опорных векторов справился с задачей лучше, чем логистическая регрессия, однако обучение модели у первого проходило намного дольше.

После проделанной работы для подтверждения результатов значимости предсказаний нейронной сети можно снова построить корреляционную матрицу, состоящую из числовых данных (рис. 23). Как мы можем наблюдать, значения корреляций новых переменных сильно выделяются на фоне остальных (корреляция мошенничества с вероятностным исходом равна 0,66, с бинарным – 0,56). Это подтверждает наши выводы о том, что результаты нейронной сети, построенные на текстовых данных, внесли большой вклад в улучшение точности предсказаний.



*Рис. 23. Корреляционная матрица числовых признаков с ответами свёрточной нейронной сети*

Подводя итоги, можно определённо констатировать факт, что при использовании текстовых данных, выраженных в предсказаниях одномерной свёрточной нейронной сети, предсказательная способность моделей увеличивается. Помимо этого, можно отметить, что на наших данных моделям было проще предсказывать мошеннические заявления в договорах страхования жизни, чем в страховании не-жизни. Однако из этого нельзя сделать вывод, что в целом моделирование страхования не-жизни – более сложная задача, потому что мы имеем выборку лишь из одного набора данных (один портфель страхования). Градиентный бустинг оказался наилучшим методом машинного обучения в обеих подвыборках, что подтвердило нашу гипотезу о том, что современные решения в области анализа данных справляются с задачей классификации лучше, чем классические методы машинного обучения. Но, если брать в расчёт результаты случайного леса, то верным решением будет строить несколько моделей и выбирать из них лучшую. Более того, на данных страхования жизни классические методы статистического анализа тоже отлично справились с поставленной задачей.

**Заключение**

В настоящее время страховые компании сталкиваются с проблемой обнаружения страхового мошенничества. Это является серьезной проблемой, которая приводит к значительным финансовым потерям и удорожанию страховых контрактов. В связи с этим научные исследования в области машинного обучения и анализа данных помогают компаниям более эффективно бороться с мошенничеством. Внедрение антифрод модели на основе нейронной сети и других методов машинного обучения может иметь положительное влияние на бизнес-процессы страховых компаний и привести к снижению мошенничества в целом.

В данной работе были использованы различные методы машинного обучения для решения задачи детекции страхового мошенничества. В первой части работы были обработаны категориальные данные (компаниям, у которых в страховых портфелях много категориальных признаков, рекомендуется использовать Target Encoder) и проведена кластеризация с помощью алгоритма k-means. Результаты кластеризации были добавлены как новые переменные в исходные данные. Затем были обработаны текстовые данные с помощью сверточной нейронной сети, которая позволила предсказывать вероятность мошенничества. Полученные ответы тоже были добавлены как новые признаки в датафрейм.

Далее были применены различные модели многомерного статистического анализа и машинного обучения: логистическая регрессия, метод опорных векторов, линейный дискриминантный анализ, градиентный бустинг и случайный лес. Было выяснено, что новые методы машинного обучения справляются с задачей детекции страхового мошенничества лучше, чем классические статистические методы. Более того, использование текстовых данных, выраженных в предсказаниях свёрточной нейронной сети, улучшило качество классификации. Наилучшим вариантом оказалось сочетание ответов кластеризации, нейронной сети и градиентного бустинга.

При этом страховым компаниям, ограниченным в ресурсах перепроверки тех заявлений, которые модель классифицировала мошенническими, рекомендуется оценить результаты алгоритма случайного леса на своих страховых портфелях, так как, возможно, из-за более высокого значения метрики точности (Precision), она может лучше подходить для решения задачи детекции страхового мошенничества. Также страховщикам рекомендуется в страховых заявлениях требовать у страхователей как можно больше текстовой информации, написанной именно страхователем, например, подробное описание страхового случая. Такой подход поможет выявить определённые паттерны[[25]](#footnote-25) в заявлениях, которые с бо́льшей вероятностью оказываются мошенническими.

Реализация противомошеннической (антифрод) модели на основе комбинации нейронной сети и других методов машинного обучения может иметь значительное влияние на бизнес-процессы страховых компаний.

Во-первых, разработанная модель может улучшить точность детекции мошенничества, что позволит страховым компаниям более эффективно бороться с мошенническими заявлениями. Это поможет снизить финансовые потери, связанные с выплатами по мошенническим заявлениям, и повысить доверие клиентов к страховым компаниям.

Во-вторых, антифрод модель может помочь улучшить качество обслуживания клиентов. Если компания использует эффективные алгоритмы детекции мошенничества, то клиенты могут быть уверены в том, что их премии не будут использованы для выплаты по мошенническим заявлениям, и это может улучшить их отношение к компании.

В-третьих, модель детекции мошеннических операций может помочь страховым компаниям улучшить свою репутацию. Если компания показывает высокую эффективность в борьбе с мошенничеством, то это может улучшить ее репутацию в глазах клиентов, общественности и конкурентов.

В-четвёртых, внедрение такой модели с последующим её улучшением поспособствует повышению общей стоимости страховой компании, владеющей данными технологиями, росту её капитализации на финансовом и срочном рынках, если компанией было проведено публичное размещение ценных бумаг.

Следует отметить, что уменьшение страхового мошенничества в целом также может привести к снижению стоимости страховых премий для «честных» клиентов. Если страховая компания сможет снизить количество выплат по мошенническим заявлениям, то это может позволить ей снизить свои затраты и, соответственно, уменьшить стоимость страховых премий для всех клиентов, что, в свою очередь, поспособствует увеличению спроса на страховые услуги и росту объёма рынка страхования.

**Список литературы**

1) Кузнецов, А. С. 2017. Страховое мошенничество в России и за рубежом [Электронный ресурс]. - URL: https://cyberleninka.ru/article/n/strahovoe-moshennichestvo-v-rossii-i-za-rubezhom/viewer (дата обращения: 22.04.2023).

2) Махучиев Х. М. 2017. Мошенничество в сфере страхования в контексте реформирования уголовного законодательства Российской Федерации [Электронный ресурс]. - URL: https://cyberleninka.ru/article/n/moshennichestvo-v-sfere-strahovaniya-v-kontekste-reformirovaniya-ugolovnogo-zakonodatelstva-rossiyskoy-federatsii/viewer (дата обращения: 27.04.2023).

3) Николаева, Ю.В. 2014. Страховое мошенничество как фактор, препятствующий развитию страхового рынка России. Мир науки, культуры, образования, 4(53), с. 197-198 [Электронный ресурс]. - URL: https://cyberleninka.ru/article/n/strahovoe-moshennichestvo-kak-faktor-prepyatstvuyuschiy-razvitiyu-strahovogo-rynka-rossii (Дата обращения: 27.04.2023)

4) Artı́s M., Ayuso M., Guillén M. Modelling different types of automobile insurance fraud behaviour in the Spanish market // Journal of Risk and Insurance. - 2008. - Т. 75, № 4. - P. 829-855 [Электронный ресурс]. URL: https://www.sciencedirect.com/science/article/abs/pii/S0167668798000389 (дата обращения: 10.04.2023).

5) Aslam, F., Hunjra, A. I., Ftiti, Z., Louhichi, W., & Shams, T. (2021). Insurance fraud detection: Evidence from artificial intelligence and machine learning. Journal of Business Research, 130, 684-697 [Электронный ресурс]. URL: https://www.sciencedirect.com/science/article/abs/pii/S0275531922001325 (дата обращения: 12.04.2023).

6) Belhadji, E. B., & Dionne, G. (1997). Development of an Expert System for the Automatic Detection of Automobile Insurance Fraud. Journal of Insurance Issues, 20(1), 1-22 [Электронный ресурс]. URL: https://papers.ssrn.com/sol3/papers.cfm?abstract\_id=134768 (дата обращения: 15.04.2023).

7) Bologa, A., Bologa, R., & Florea, A. (2020). Big Data and Specific Analysis Methods for Insurance Fraud Detection. Database Systems Journal, 14(4), 3-25 [Электронный ресурс]. - URL: http://dbjournal.ro/archive/14/14\_4.pdf (дата обращения: 23.04.2023).

8) Conneau, A., Schwenk, H., Barrault, L., & Lecun, Y. (2017). Very Deep Convolutional Networks for Text Classification. Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL) [Электронный ресурс]. - URL: https://aclanthology.org/E17-1104.pdf (дата обращения: 23.04.2023).

9) Géron, A. (2019). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. O'Reilly Media [Электронный ресурс]. - URL: https://www.oreilly.com/library/view/hands-on-machine-learning/9781492032632/ (дата обращения: 27.04.2023).

10) Karandikar, A. M., & Waghade, S. S. (2018). A Comprehensive Study of Healthcare Fraud Detection based on Machine Learning. International Journal of Applied Engineering Research, 13(6), 4175-4178, [Электронный ресурс]. - URL: https://www.ripublication.com/ijaer18/ijaerv13n6\_140.pdf (дата обращения: 19.04.2023).

11) Khan, M. J., & Kitipornchai, S. (2016). Fraud detection in health insurance claims: A big data approach. IEEE [Электронный ресурс]. - URL: https://ieeexplore.ieee.org/abstract/document/7823950 (дата обращения: 20.04.2023).

12) Kirlidoga, M., & Asuk, C. (2012). A fraud detection approach with data mining in health insurance. Elsevier [Электронный ресурс]. - URL: https://www.sciencedirect.com/science/article/pii/S1877042812036099 (дата обращения: 18.04.2023).

13) Roy, R., & George, K. T. (2019). Detecting insurance claims fraud using machine learning techniques. 2019 IEEE 5th International Conference on Computational Intelligence and Computing Research (ICCIC), 1-5 [Электронный ресурс]. URL: https://ieeexplore.ieee.org/abstract/document/8074258 (дата обращения: 30.04.2023).

14) Rukhsar, L., Bangyal, W. H., Nisar, K., & Nisar, S. (2016). Prediction of Insurance Fraud Detection using Machine Learning Algorithms. Informit [Электронный ресурс]. - URL: https://search.informit.org/doi/epdf/10.3316/informit.263147785515876 (дата обращения: 18.04.2023).

15) Severino, M.K. and Peng, Y. (2021) Machine learning algorithms for fraud prediction in property insurance: Empirical evidence using real-world microdata. Expert Systems with Applications [Электронный ресурс]. URL: https://www.sciencedirect.com/science/article/pii/S2666827021000372 (дата обращения: 11.04.2023).

16) Xu W., Wang Y. Leveraging deep learning with LDA-based text analytics to detect automobile insurance fraud // Decision Support Systems. – 2018. – Vol. 105. – P. 87-95 [Электронный ресурс]. URL: https://www.sciencedirect.com/science/article/abs/pii/S0167923617302130 (дата обращения: 10.04.2023).

17) Yankol-Schalck M. The value of cross-data set analysis for automobile insurance fraud detection // Expert Systems with Applications. — 2022. — Т. 184. — С. 115667 [Электронный ресурс]. URL: https://www.sciencedirect.com/science/article/abs/pii/S0275531922001556 (дата обращения: 10.04.2023).

18) Yaram, S. 2016. Machine learning algorithms for document clustering and fraud detection. IEEE Xplore [Электронный ресурс]. URL: https://ieeexplore.ieee.org/abstract/document/7823950 (дата обращения: 09.05.2023).

19) Zhou Y., Xia H., Zhang Z. Auto insurance fraud identification based on a CNN-LSTM fusion deep learning model // International Journal of Ad Hoc and Ubiquitous Computing. – 2022. – Vol. 39, No. 1-2. – P. 37-45 [Электронный ресурс]. URL: https://www.inderscienceonline.com/doi/abs/10.1504/IJAHUC.2022.120943 (дата обращения: 12.04.2023).

20) Материалы научной конференции «Наука и мир в современном мире», 2019. Определение области страхового мошенничества на основе машинного обучения [Электронный ресурс]. URL: http://science-peace.ru/files/STRNO\_2019.pdf#page=43 (дата обращения: 01.05.2023).

21) Федеральный закон «Об организации страхового дела в Российской Федерации» от 27.11.1992 N 4015-1 [Электронный ресурс]. URL: https://docs.cntd.ru/document/9003385 (дата обращения: 02.05.2023).

22) Coalition Against Insurance Fraud. (2021). Fraud Stats [Электронный ресурс]. - URL: https://insurancefraud.org/fraud-stats/ (дата обращения: 25.04.2023).

23) Federal Bureau of Investigation. (2021). Insurance Fraud [Электронный ресурс]. -URL: https://www.fbi.gov/stats-services/publications/insurance-fraud (дата обращения: 24.04.2023).

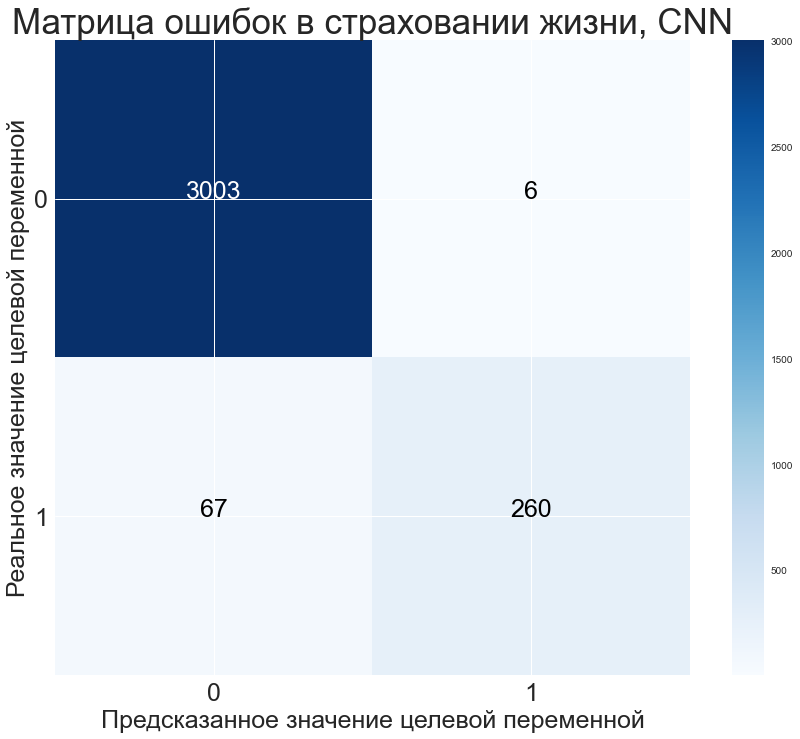
24) Insurancefraud.org. (n.d.). Fraud Statistics [Электронный ресурс]. URL: https://insurancefraud.org/fraud-stats/ (дата обращения: 29.04.2023).

25) National Association of Insurance Commissioners. (2021). Insurance Fraud Bureau: Top 6 Insurance Scams to Avoid [Электронный ресурс]. -URL: https://content.naic.org/article/insurance-fraud-bureau-top-6-insurance-scams-avoid-0 (дата обращения: 24.04.2023).

**Приложения**

**Приложение 1**

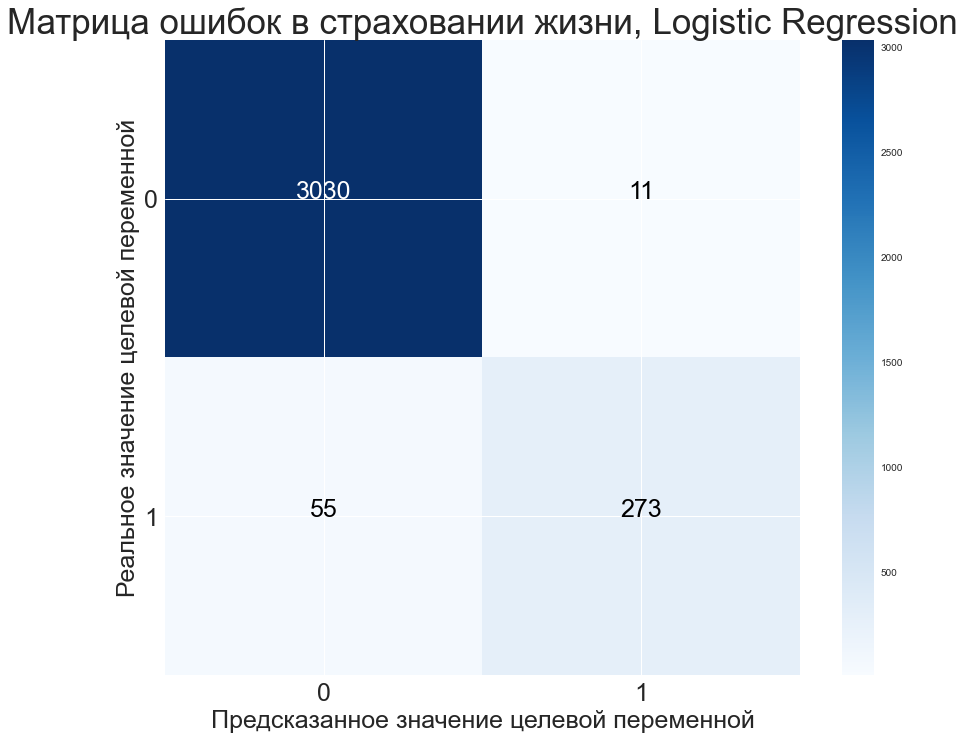
**Матрица ошибок свёрточной нейронной сети в данных страхования жизни**



*Рис. 24. Матрица ошибок в страховании жизни, свёрточная нейронная сеть*

**Приложение 2**

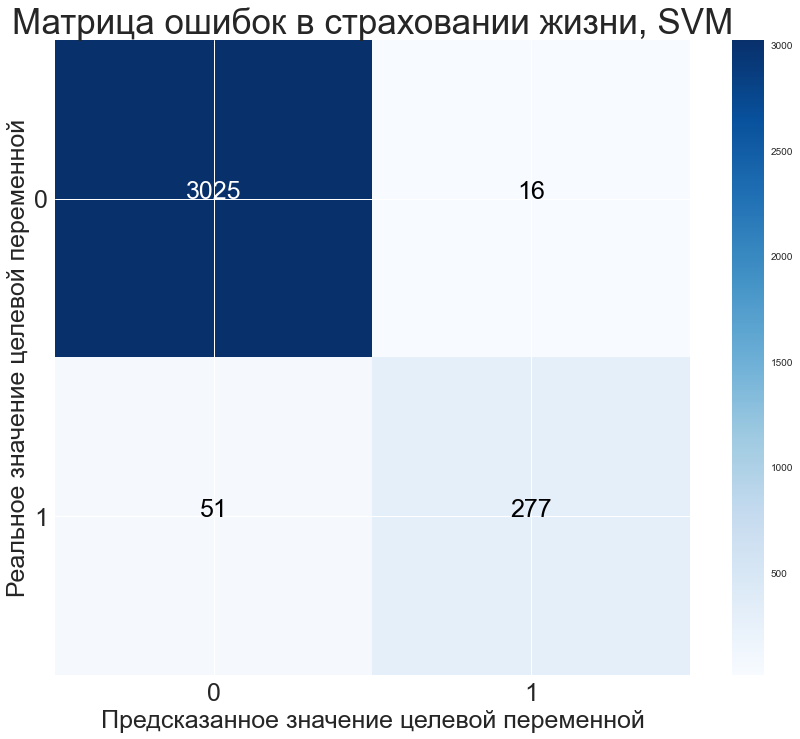
**Матрица ошибок логистической регрессии в данных страхования жизни**



*Рис. 25. Матрица ошибок в страховании жизни, логистическая регрессия*

**Приложение 3**

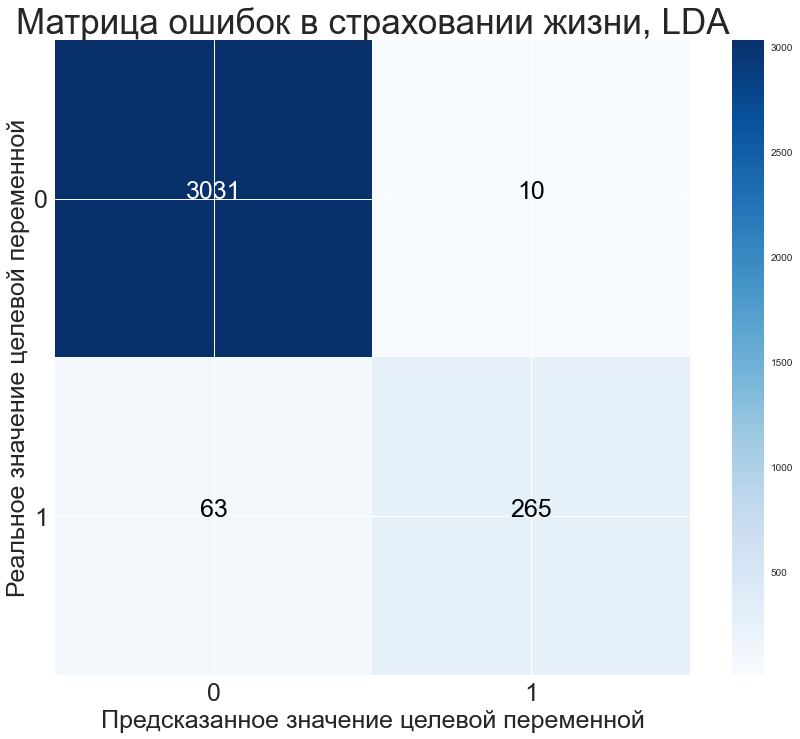
**Матрица ошибок метода опорных векторов в данных страхования жизни**



*Рис. 26. Матрица ошибок в страховании жизни, метод опорных векторов*

**Приложение 4**

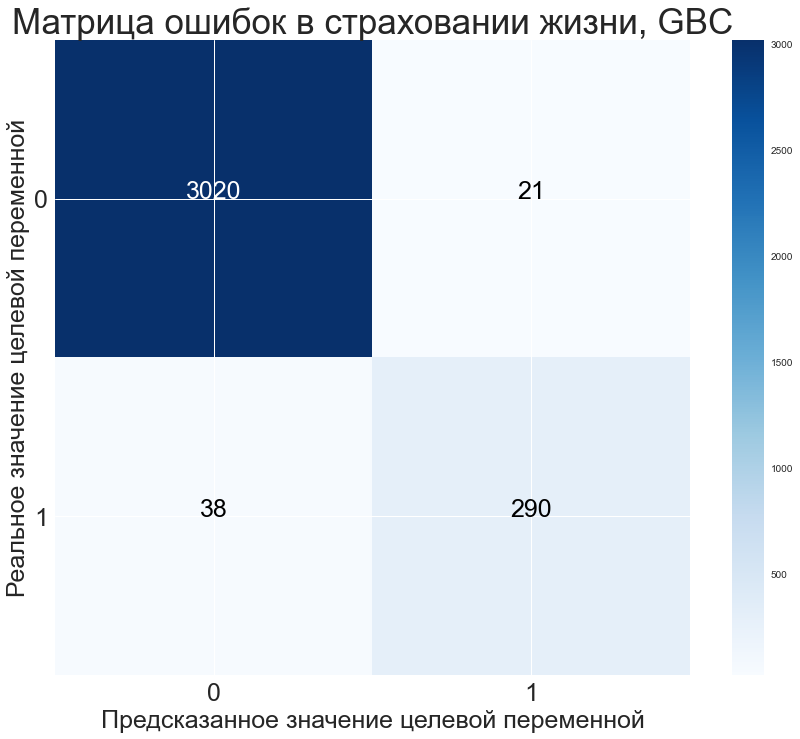
**Матрица ошибок модели линейного дискриминантного анализа в данных страхования жизни**



*Рис. 27. Матрица ошибок в страховании жизни, классификатор линейного дискриминантного анализа*

**Приложение 5**

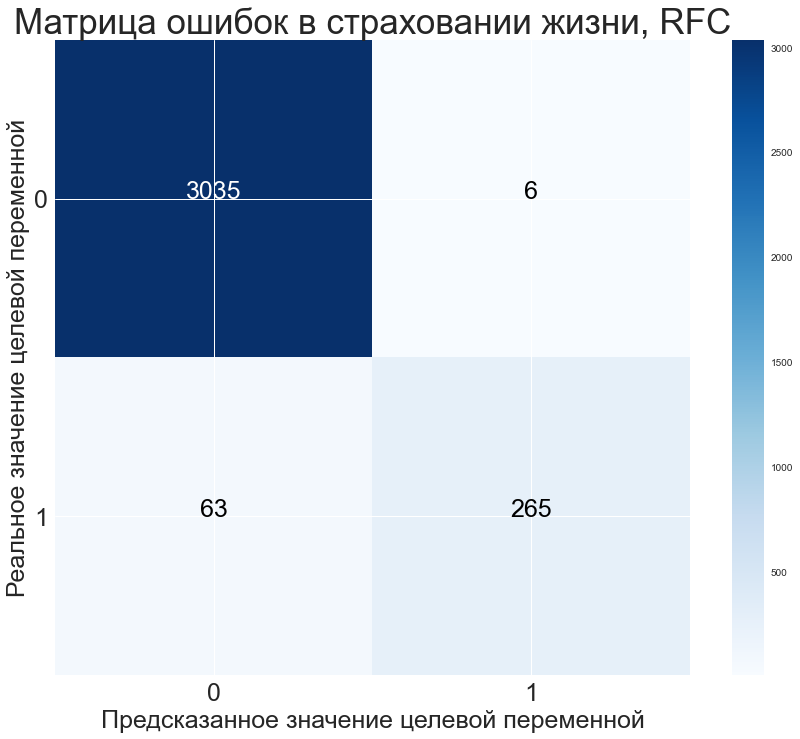
**Матрица ошибок модели градиентного бустинга в данных страхования жизни**



*Рис. 28. Матрица ошибок в страховании жизни, градиентный бустинг*

**Приложение 6**

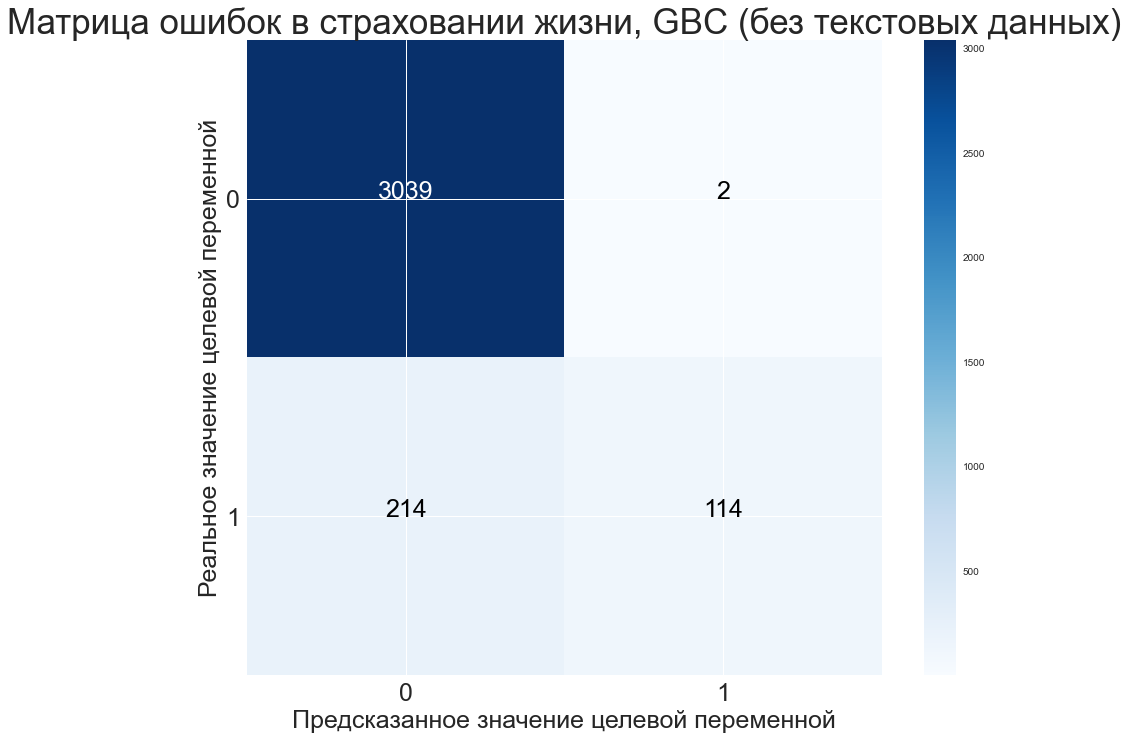
**Матрица ошибок модели случайного леса в данных страхования жизни**



*Рис. 29. Матрица ошибок в страховании жизни, случайный лес*

**Приложение 7**

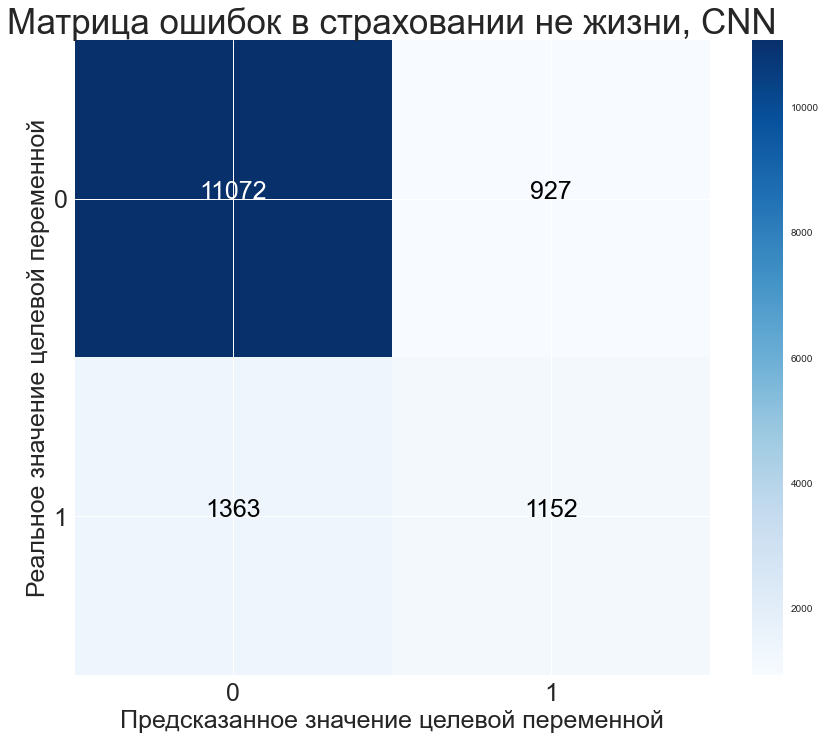
**Матрица ошибок модели градиентного бустинга в данных страхования жизни (без использования ответов нейронной сети)**



*Рис. 30. Матрица ошибок в страховании жизни, градиентный бустинг без предсказаний нейронной сети*

**Приложение 8**

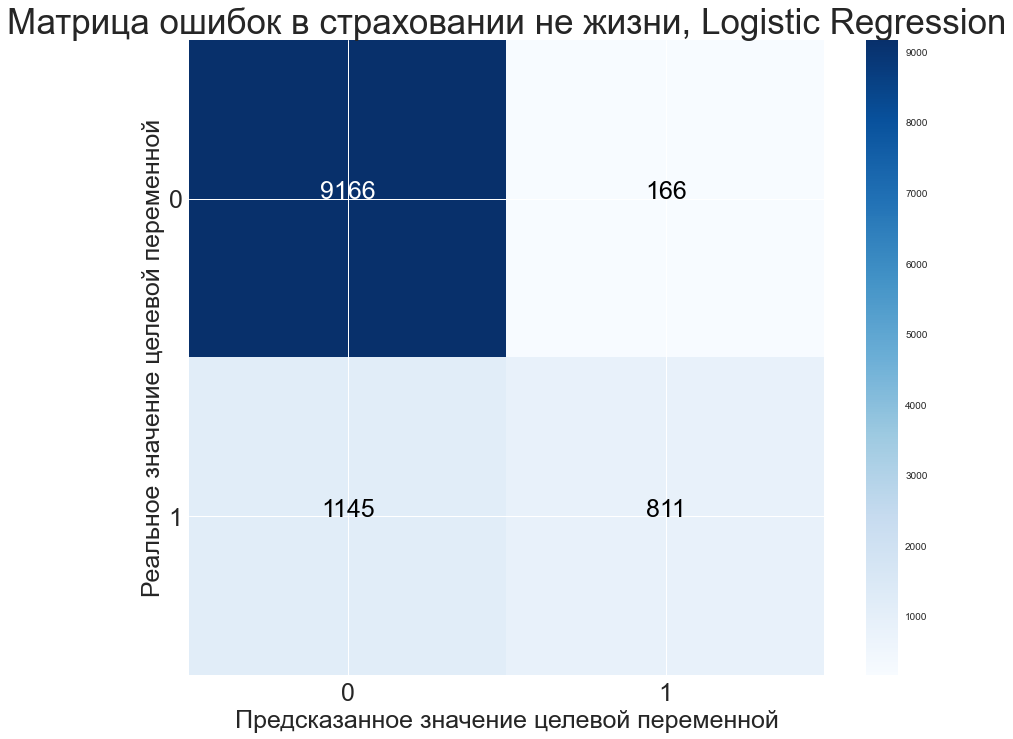
**Матрица ошибок свёрточной нейронной сети в данных страхования не-жизни**



*Рис. 31. Матрица ошибок в страховании не-жизни, свёрточная нейронная сеть*

**Приложение 9**

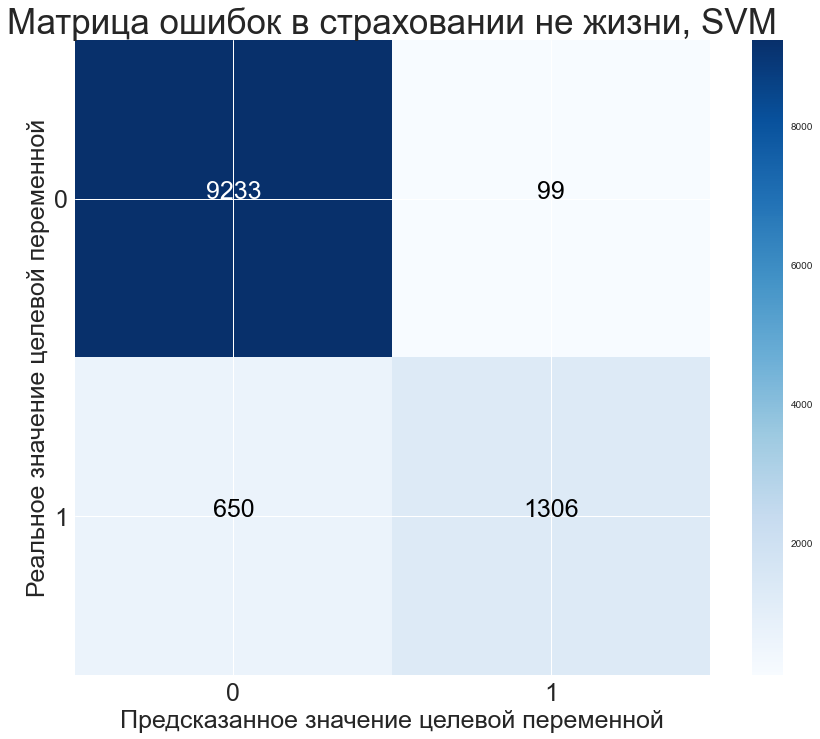
**Матрица ошибок логистической регрессии в данных страхования не-жизни**



*Рис. 32. Матрица ошибок в страховании не-жизни, логистическая регрессия*

**Приложение 10**

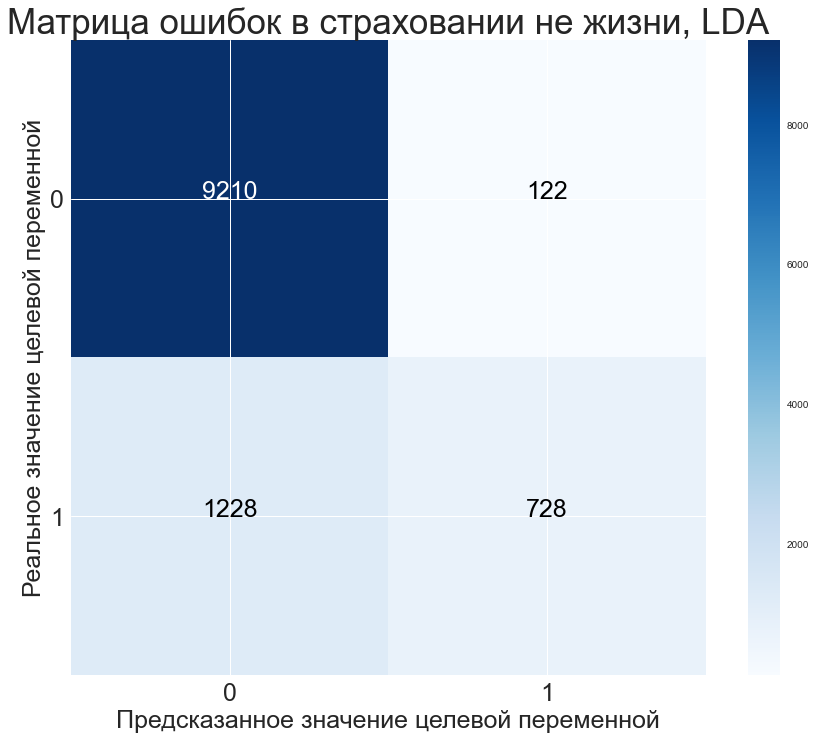
**Матрица ошибок метода опорных векторов в данных страхования не-жизни**



*Рис. 33. Матрица ошибок в страховании не-жизни, метод опорных векторов*

**Приложение 11**

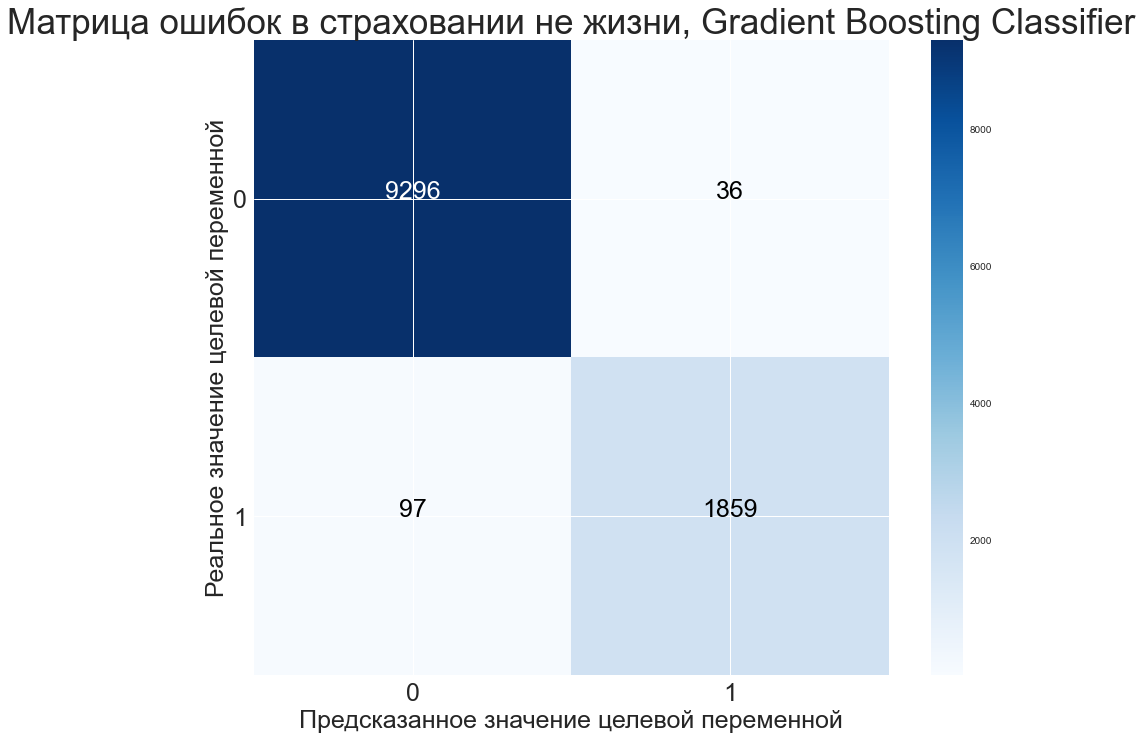
**Матрица ошибок модели линейного дискриминантного анализа в данных страхования не-жизни**



*Рис. 34. Матрица ошибок в страховании не-жизни, классификатор линейного дискриминантного анализа*

**Приложение 12**

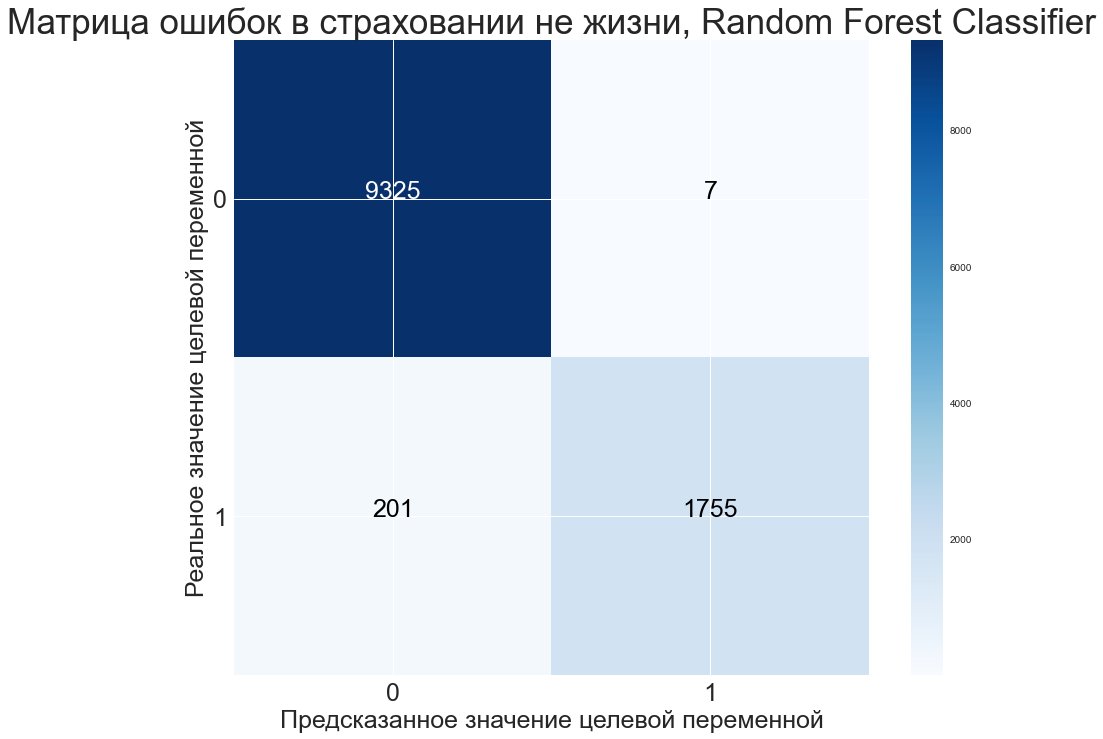
**Матрица ошибок модели градиентного бустинга в данных страхования не-жизни**



*Рис. 35. Матрица ошибок в страховании не-жизни, градиентный бустинг*

**Приложение 13**

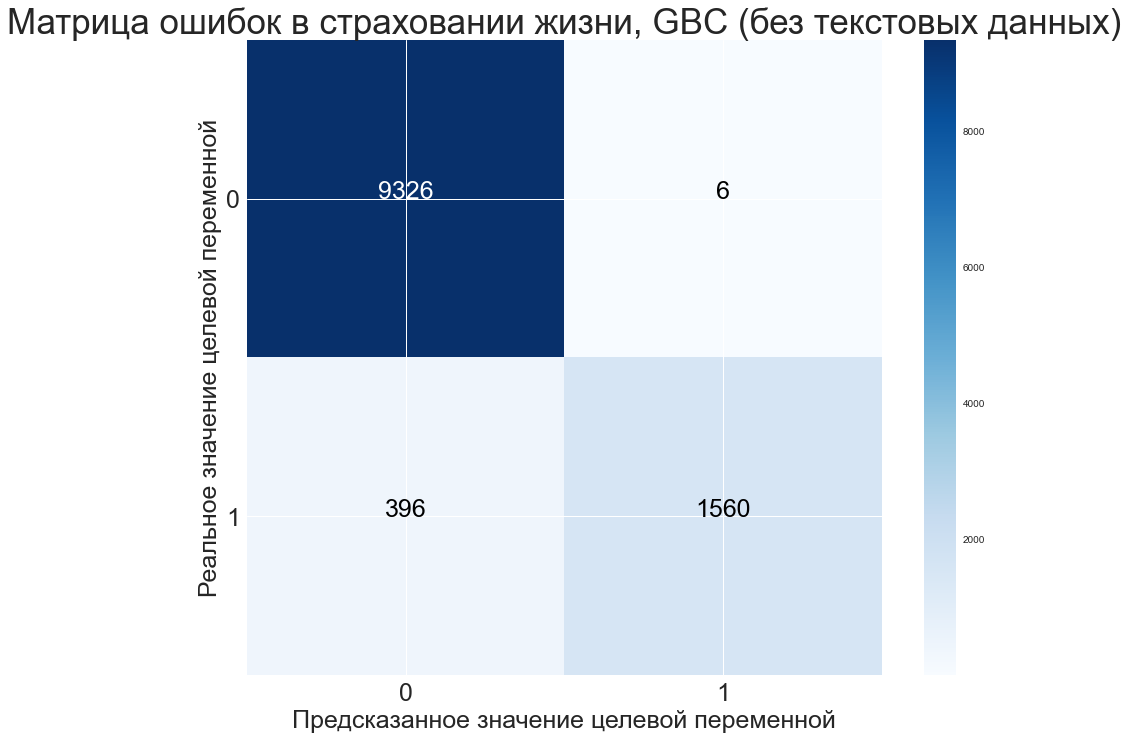
**Матрица ошибок модели случайного леса в данных страхования не-жизни**



*Рис. 36. Матрица ошибок в страховании не-жизни, случайный лес*

**Приложение 14**

**Матрица ошибок модели градиентного бустинга в данных страхования не-жизни (без использования ответов нейронной сети)**



*Рис. 37. Матрица ошибок в страховании не-жизни, градиентный бустинг без предсказаний нейронной сети*

**Приложение 15**

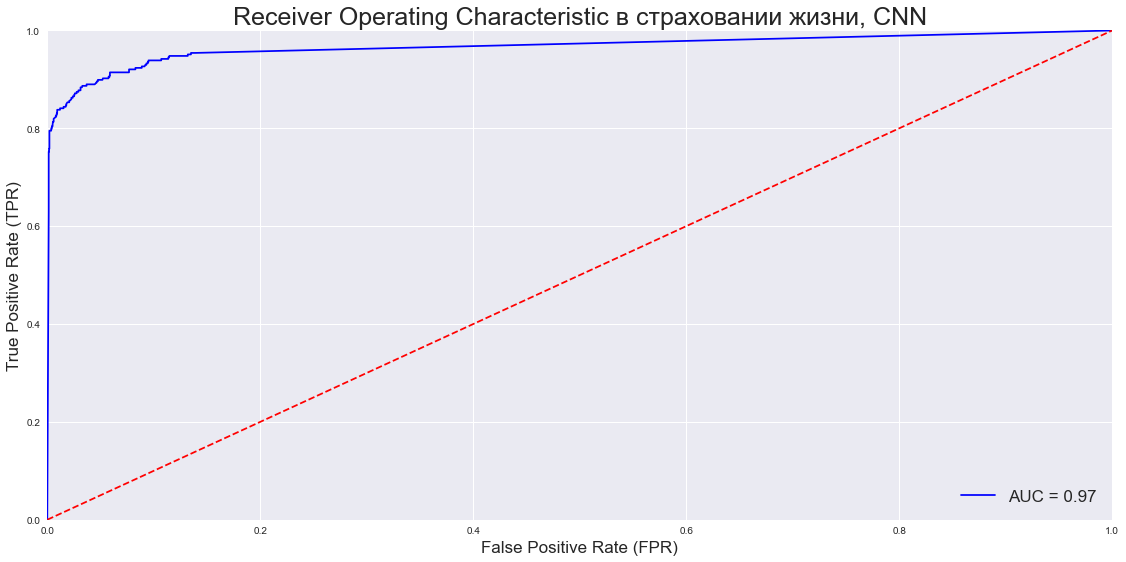
**Описательная статистика числовых признаков**

*Таблица 20*

*Описательная статистика числовых признаков*

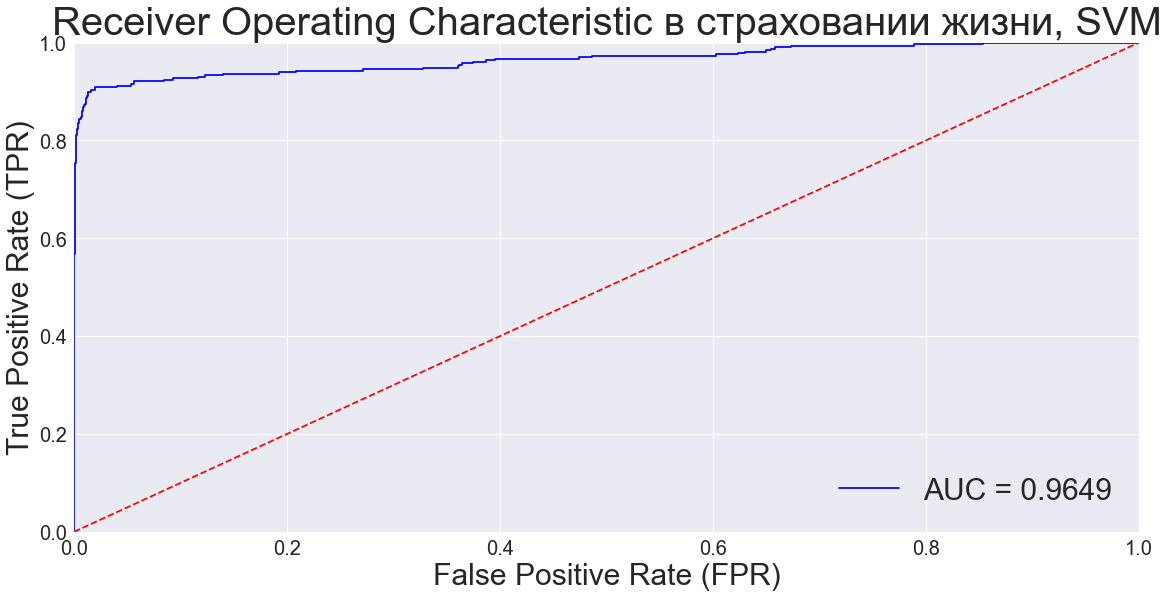
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | *Кол-во* | *Среднее* | *СКО* | *Мин.* | *Медиана* | *Макс.* |
| *Задолж. на дату убыт.* | 49 803 | 70 | 850 | 0 | 0 | 62 977 |
| *Доп. доход* | 49 094 | 1 082 | 5 366 | 0 | 0 | 425 286 |
| *Страх. сум. по риску, в вал. отв.* | 59 070 | 283 222 | 284 104 | 10 | 204 000 | 12 486 628 |
| *Кол-во дн. нетрудоспособ.* | 17 926 | 29 | 24 | 1 | 22 | 1 147 |
| *Опл. дн. нетрудоспособ.* | 42 157 | 9 | 15 | 0 | 0 | 90 |
| *ЗУ, % от страх. сум.* | 58 654 | 104 | 353 | 0 | 5 | 9 000 |
| *РЗУ, в вал. отв.* | 59 093 | 2 349 | 38 117 | 0 | 0 | 5 459 000 |
| *Разм. страх. выпл. в вал. отв.* | 54 357 | 34 520 | 95 764 | 0 | 10 131 | 12 486 628 |
| *Разм. страх. выпл.* | 54 357 | 40 499 | 115 008 | 0 | 11 600 | 12 486 628 |
| *Доля выгодоприобр.* | 58 943 | 100 | 5 | 0 | 100 | 100 |
| *Сум. к выпл.* | 49 794 | 32 578 | 98 867 | 0 | 8 400 | 12 486 628 |
| *НДФЛ* | 49 793 | 156 | 2 672 | 0 | 0 | 241 777 |
| *СС main programm* | 59 121 | 37 205 | 107 687 | 0 | 10 300 | 12 861 227 |
| *Разн. меж. дат. вступ. дог. в силу и наст. страх. случ.* | 59 121 | 851 | 806 | 1 | 622 | 4 810 |

**Приложение 16**

**График ROC кривой в страховании жизни, свёрточная нейронная сеть** 

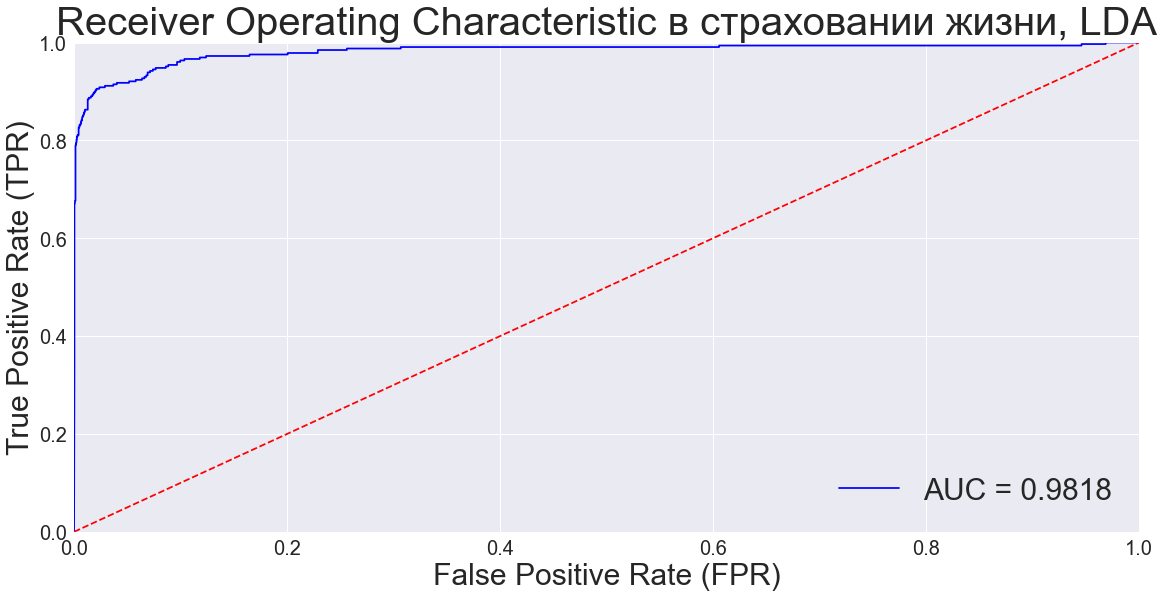
*Рис. 37. График ROC кривой в страховании жизни, свёрточная нейронная сеть*

**Приложение 17**

**График ROC кривой в страховании жизни, метод опорных векторов** 

*Рис. 38. График ROC кривой в страховании жизни, метод опорных векторов*

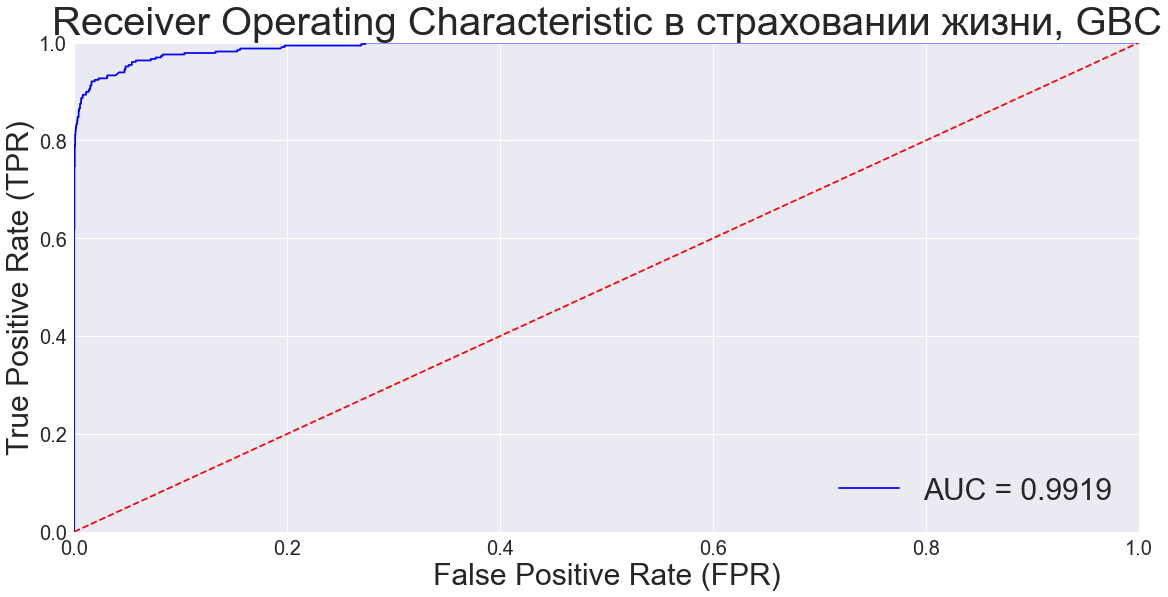
**Приложение 18**

**График ROC кривой в страховании жизни, классификатор линейного дискриминантного анализа** 

*Рис. 39. График ROC кривой в страховании жизни, классификатор линейного дискриминантного анализа*

**Приложение 19**

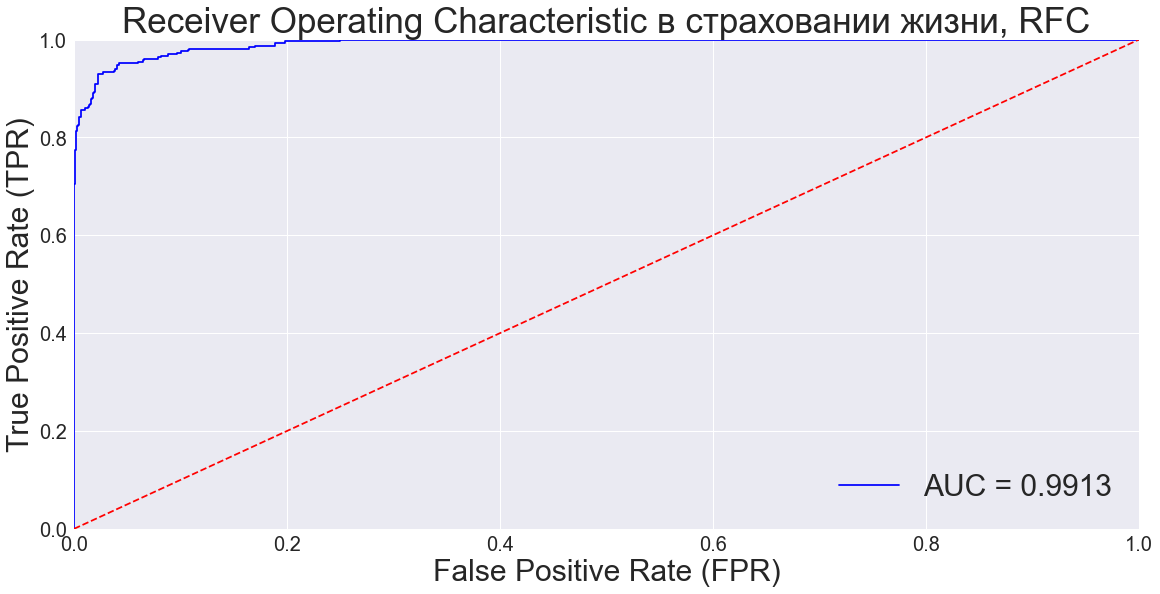
**График ROC кривой в страховании жизни, градиентный бустинг**



*Рис. 40. График ROC кривой в страховании жизни, градиентный бустинг*

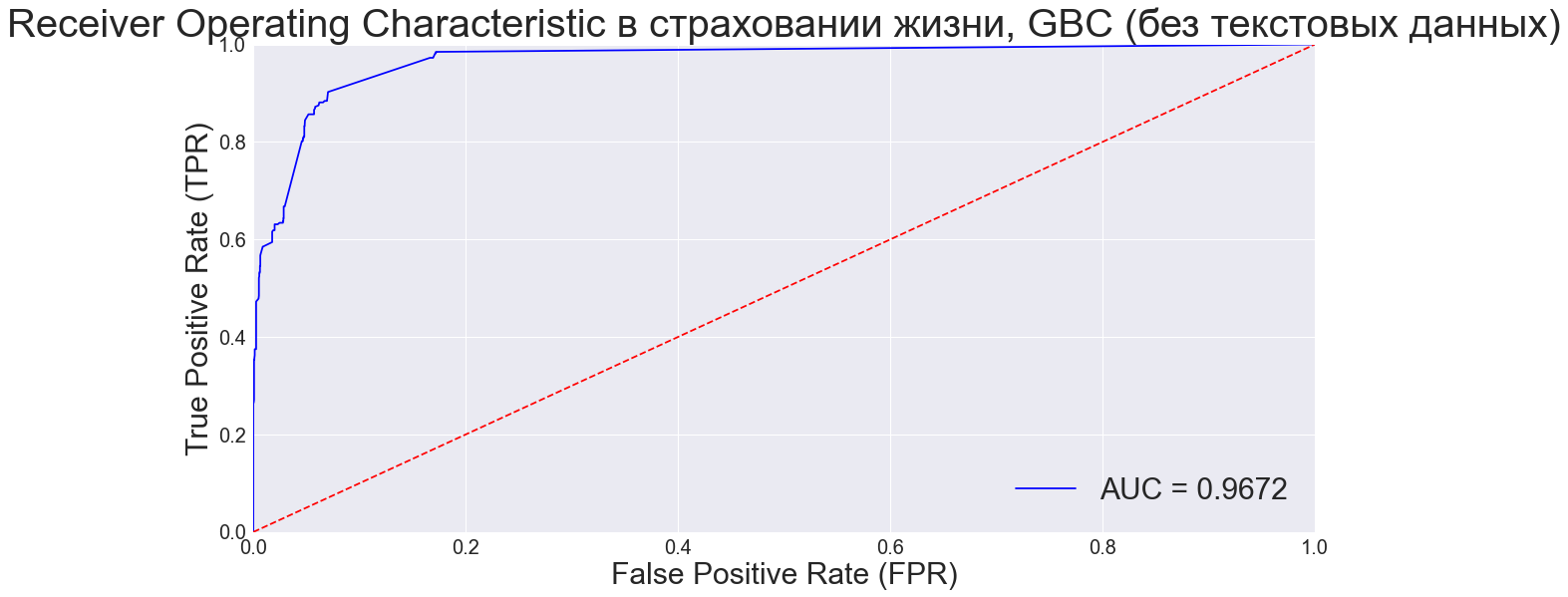
**Приложение 20**

**График ROC кривой в страховании жизни, случайный лес**



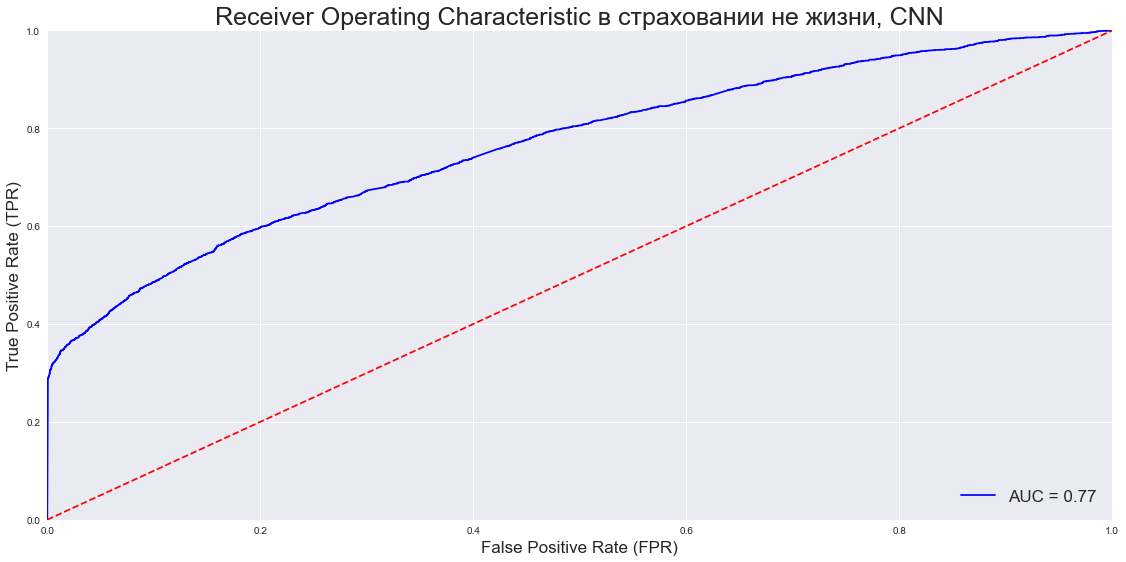
*Рис. 41. График ROC кривой в страховании жизни, случайный лес*

**Приложение 21**

**График ROC кривой в страховании жизни, градиентный бустинг без предсказаний нейронной сети** 

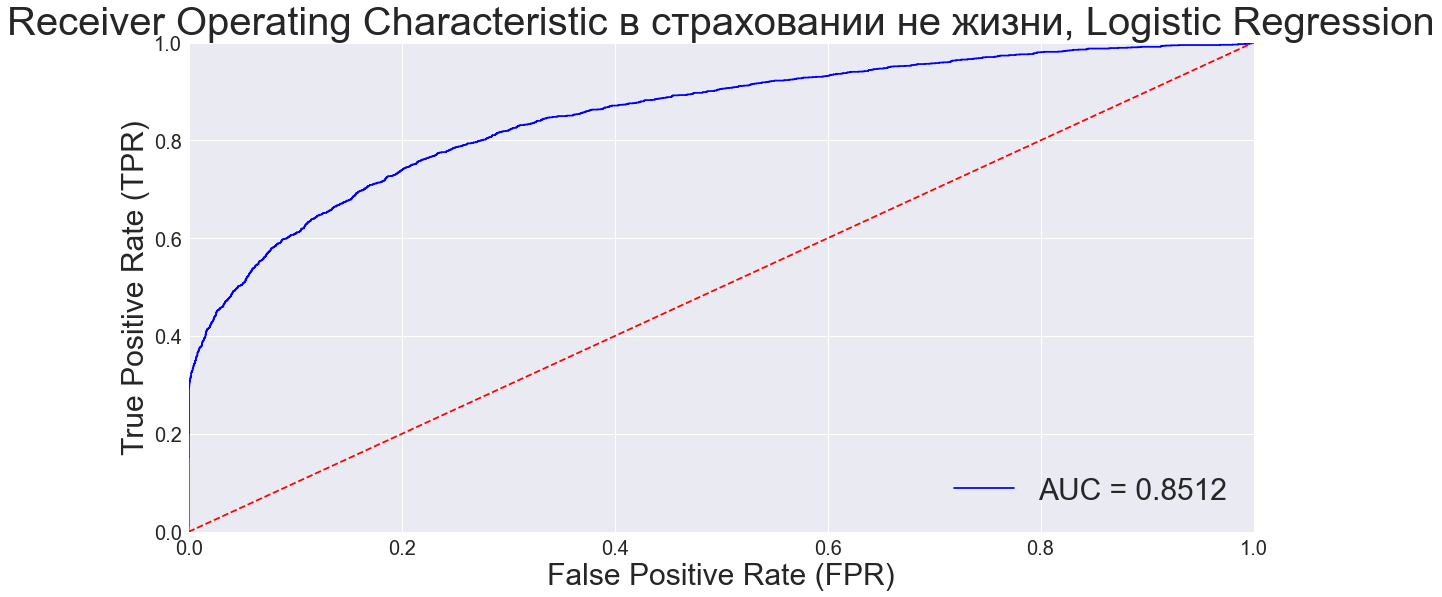
*Рис. 42. График ROC кривой в страховании жизни, градиентный бустинг без предсказаний нейронной сети*

**Приложение 22**

**График ROC кривой в страховании не-жизни, свёрточная нейронная сеть** 

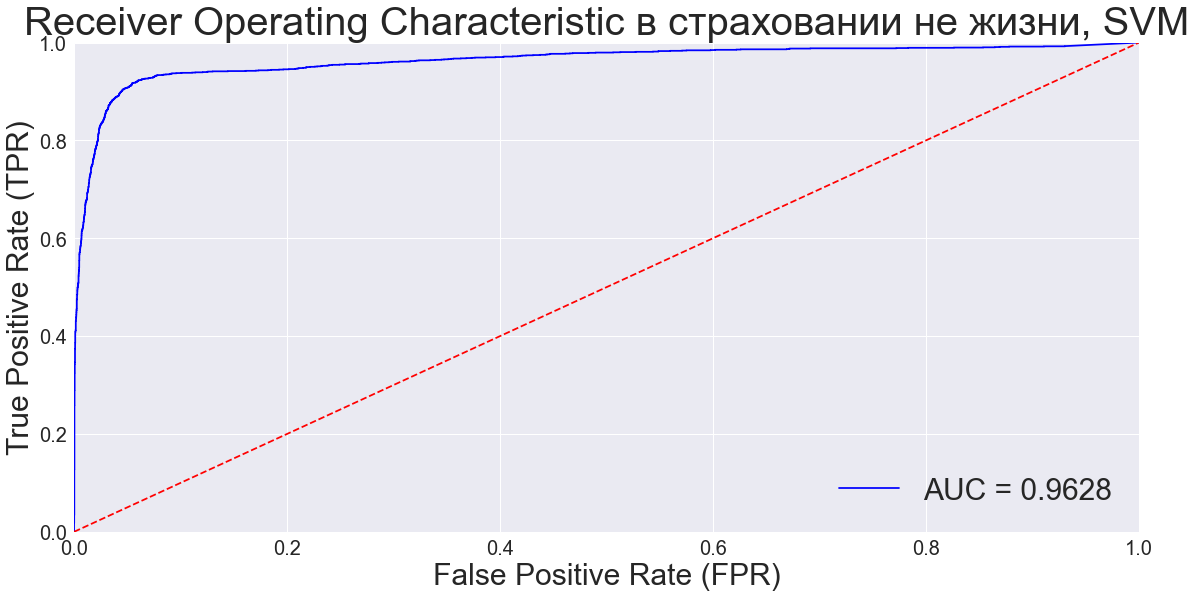
*Рис. 43. График ROC кривой в страховании не-жизни, свёрточная нейронная сеть*

**Приложение 23**

**График ROC кривой в страховании не-жизни, логистическая регрессия**

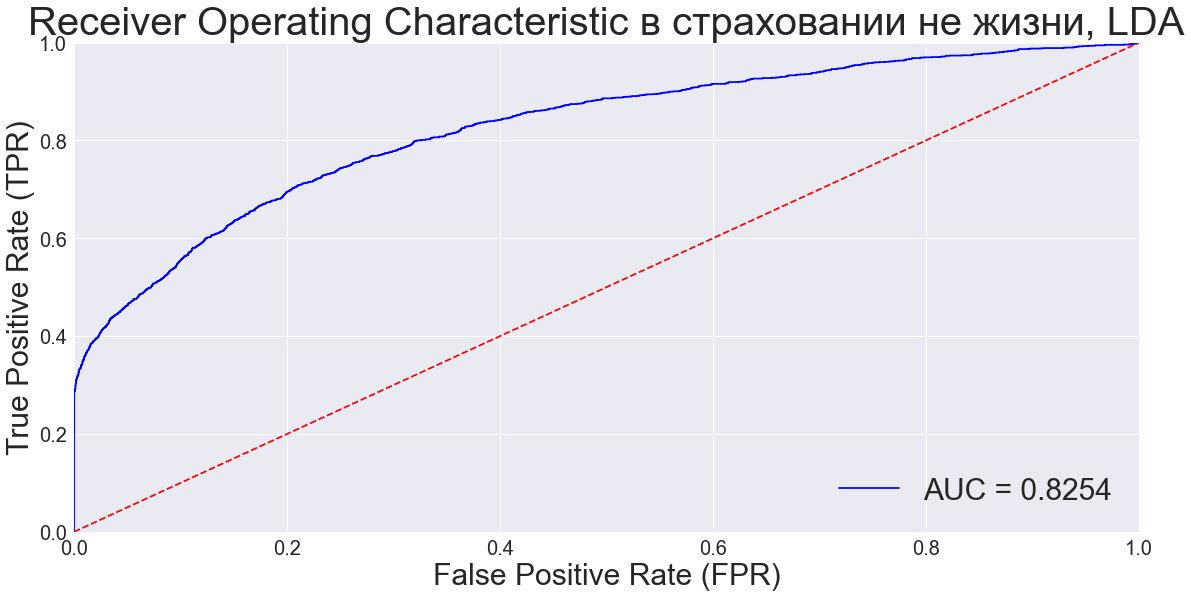
*Рис. 44. График ROC кривой в страховании не-жизни, логистическая регрессия*

**Приложение 24**

**График ROC кривой в страховании не-жизни, метод опорных векторов**

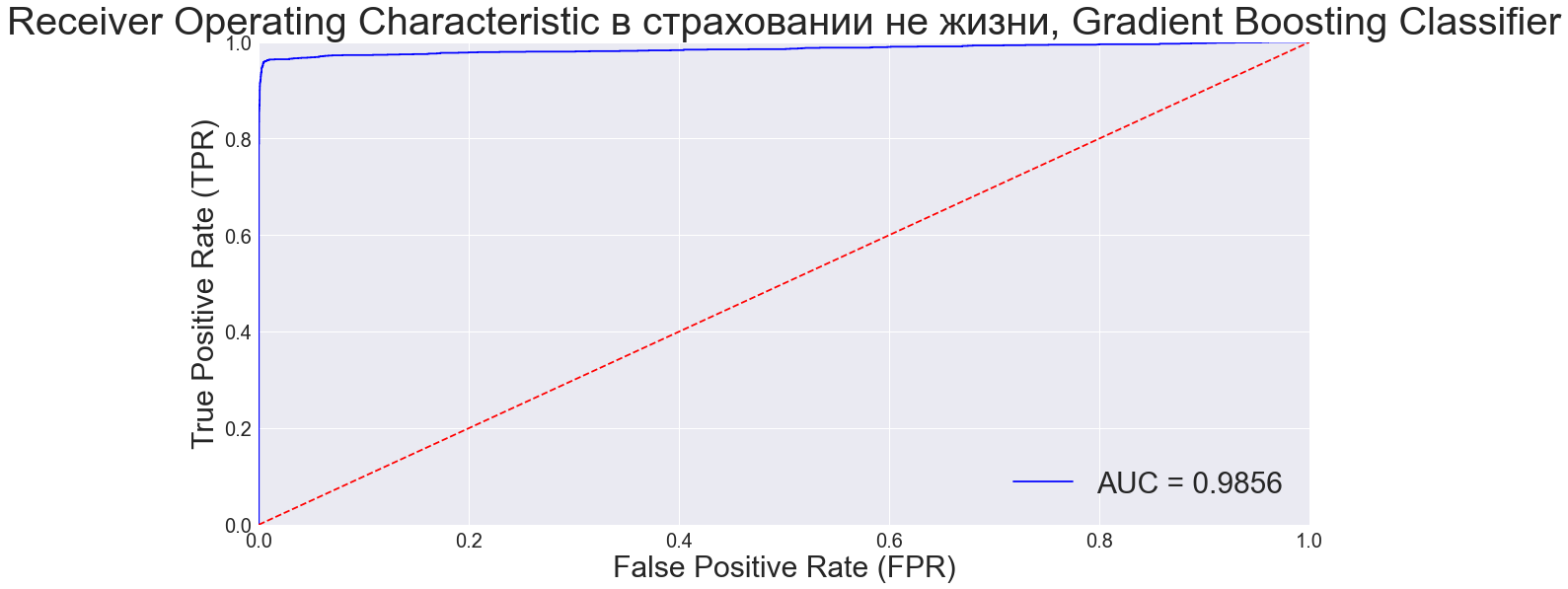
*Рис. 45. График ROC кривой в страховании не-жизни, метод опорных векторов*

**Приложение 25**

**График ROC кривой в страховании не-жизни, классификатор линейного дискриминантного анализа**

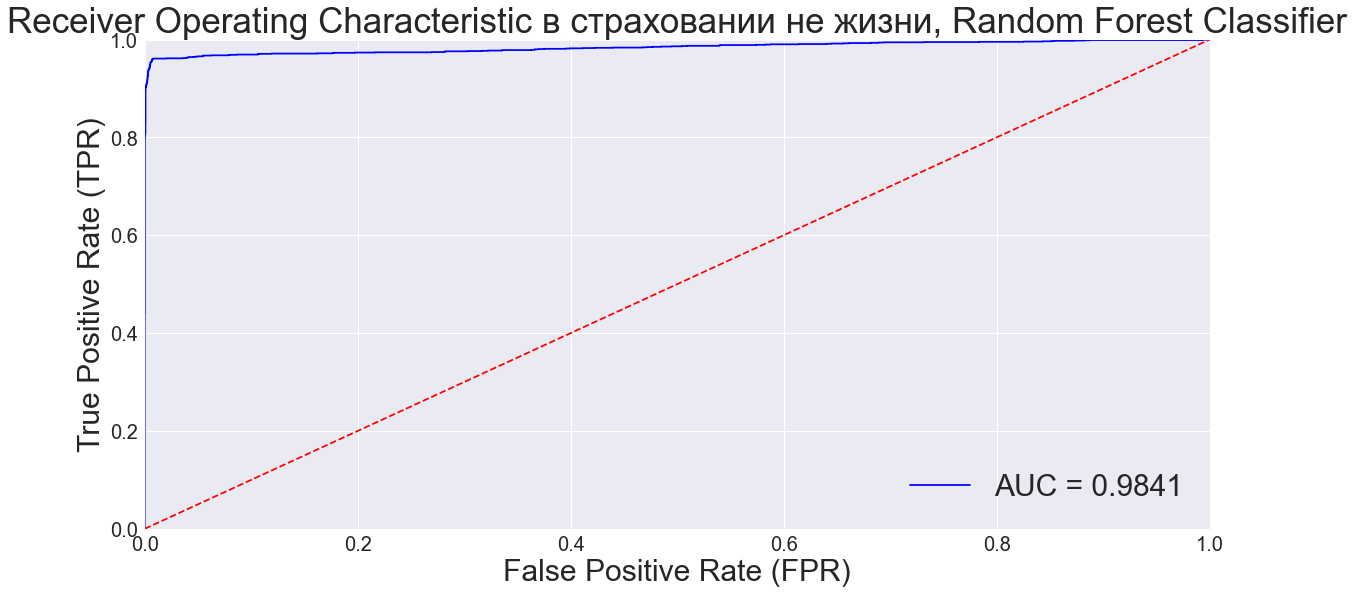
*Рис. 46. График ROC кривой в страховании не-жизни, классификатор линейного дискриминантного анализа*

**Приложение 26**

**График ROC кривой в страховании не-жизни, градиентный бустинг**

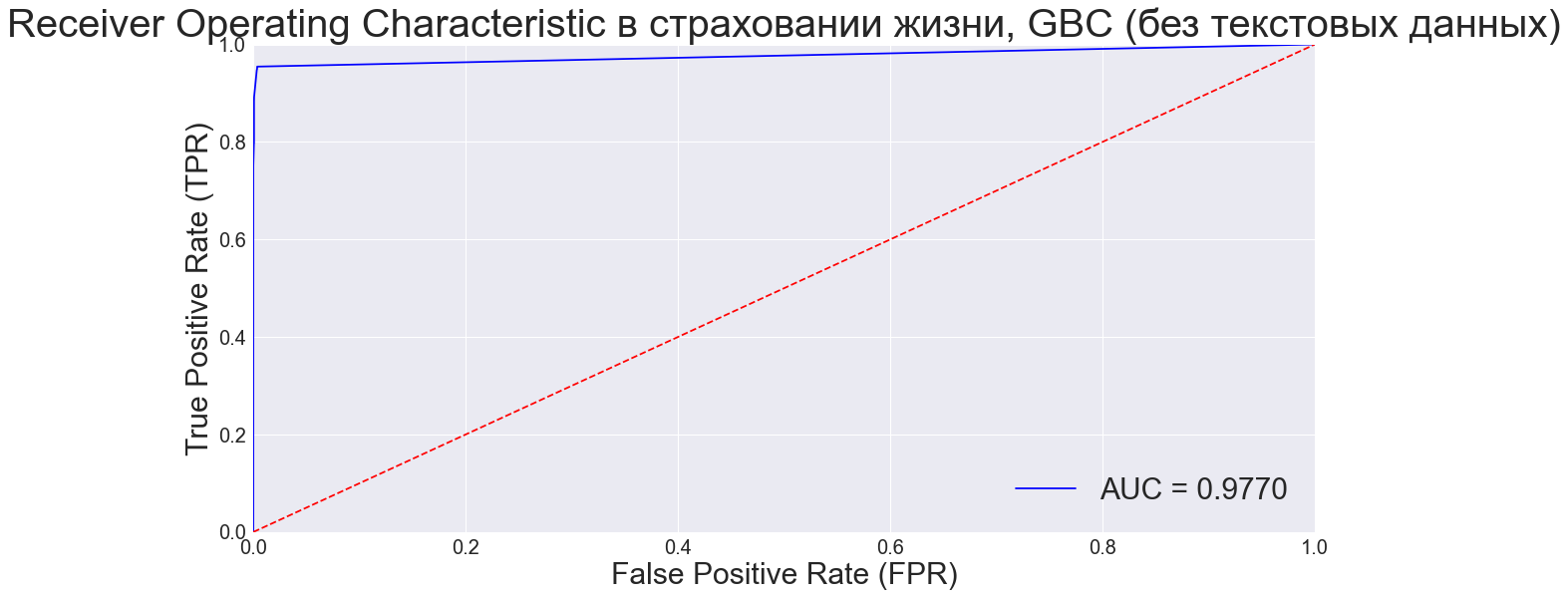
*Рис. 47. График ROC кривой в страховании не-жизни, градиентный бустинг*

**Приложение 27**

**График ROC кривой в страховании не-жизни, случайный лес** 

*Рис. 48. График ROC кривой в страховании не-жизни, случайный лес*

**Приложение 28**

**График ROC кривой в страховании не-жизни, градиентный бустинг без предсказаний нейронной сети** 

*Рис. 49. График ROC кривой в страховании не-жизни, градиентный бустинг без предсказаний нейронной сети*

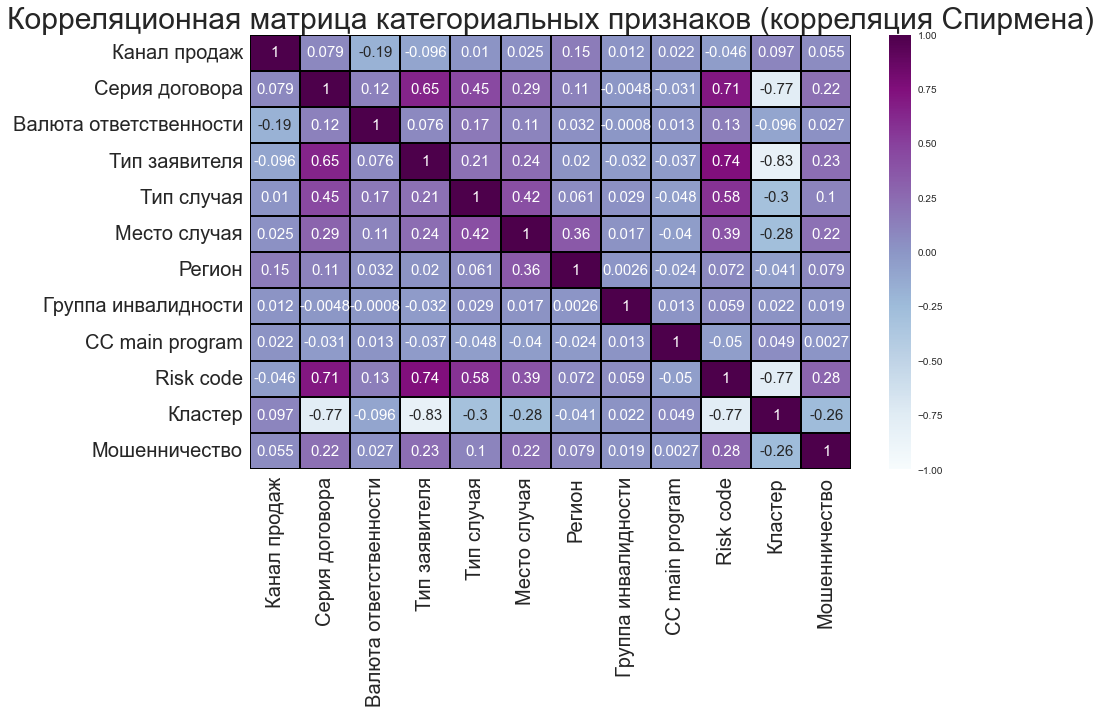
**Приложение 29**

**Корреляционная матрица числовых признаков, без ответов свёрточной нейронной сети**

*Рис. 50. Корреляционная матрица числовых признаков, без ответов свёрточной нейронной сети*

**Приложение 30**

**Корреляционная матрица категориальных признаков**



*Рис. 51. Корреляционная матрица категориальных признаков*

1. Брутто премия в страховании – это сумма, которую страхователь платит страховой компании за страховой полис. Она включает в себя стоимость страховой защиты, а также все дополнительные расходы, связанные с оформлением полиса, такие как комиссии агентов, налоги и другие сборы. [↑](#footnote-ref-1)
2. Центроиды (англ. centroids) — это центры кластеров, которые вычисляются в алгоритме k-means. Конечный результат алгоритма k-means – это набор центроидов, которые определяют сгруппированные кластеры в данных. [↑](#footnote-ref-2)
3. Функция потерь (loss function) – это функция, которая используется в задачах оптимизации модели машинного обучения для оценки расхождения между предсказанными значениями и фактическими значениями целевой переменной. Функция потерь измеряет ошибку, которую допускает модель при предсказании выходного значения на основе входных данных. Она должна быть дифференцируема, чтобы можно было посчитать её градиент. [↑](#footnote-ref-3)
4. Метод Ньютона-Рафсона – это численный метод для поиска корней нелинейных уравнений и оптимизации функций. Он основан на идеи последовательного приближения к решению путем локальной линеаризации функции в каждой точке и нахождения корня линеаризованной функции. Метод Ньютона-Рафсона применяется для решения широкого спектра задач, включая поиск экстремумов функций и решение систем нелинейных уравнений. [↑](#footnote-ref-4)
5. Тензор – это математический объект, который представляет собой многомерный массив чисел. В контексте машинного обучения и глубокого обучения тензоры часто используются для представления данных, таких как изображения, звуковые сигналы и тексты. [↑](#footnote-ref-5)
6. Функции активации – это функции, которые определяют выходные значения нейронов в нейронной сети. Они принимают входные значения и возвращают значения на основе заданного порога. Эти значения затем передаются в следующий слой нейронов или используются для получения конечного выхода модели. Функции активации обычно используются для нелинейного преобразования входных данных, чтобы нейронная сеть могла изучать сложные зависимости между данными и достигать более высокой точности. [↑](#footnote-ref-6)
7. LDA (Latent Dirichlet Allocation) – это вероятностная модель, используемая для выявления тем в наборе документов. Она позволяет определить скрытые темы, основываясь на распределении слов в документах. [↑](#footnote-ref-7)
8. LSTM (Long Short-Term Memory) – это тип рекуррентной нейронной сети (RNN), который может обрабатывать и моделировать последовательности с учётом запоминания длинных зависимостей. Он был представлен Хохрейтером и Шмидхубером в 1997 году. [↑](#footnote-ref-8)
9. DNN (Deep Neural Network) – это нейронная сеть с несколькими скрытыми слоями, которые позволяют моделировать более сложные нелинейные зависимости между входными и выходными данными. Каждый скрытый слой в DNN представляет собой набор нейронов, которые принимают входные данные и преобразуют их с помощью активационной функции. [↑](#footnote-ref-9)
10. Boruta – это алгоритм отбора признаков, который используется в машинном обучении. Он работает на основе алгоритма Random Forest и позволяет создавать рейтинг признаков от самого важного до наименее влиятельного для модели. [↑](#footnote-ref-10)
11. Наивный Байес (Naive Bayes) – это алгоритм машинного обучения, основанный на теореме Байеса. Основная идея заключается в том, чтобы оценить вероятность того, что объект принадлежит к определенному классу, и выбрать класс с наибольшей вероятностью. В наивном Байесовском классификаторе считается, что все признаки объекта независимы друг от друга. [↑](#footnote-ref-11)
12. Accuracy – это метрика классификации, которая измеряет долю правильных предсказаний модели от общего числа предсказаний. Она представляет собой отношение количества правильно классифицированных примеров к общему числу примеров в тестовом наборе данных. [↑](#footnote-ref-12)
13. F1-оценка (F1-score) – это метрика, используемая для оценки качества классификатора в задачах бинарной классификации, которая учитывает как точность (precision), так и полноту (recall), являясь их гармоническим средним. [↑](#footnote-ref-13)
14. Кросс-валидация (cross-validation) – это метод оценки производительности модели, который позволяет использовать все доступные данные для обучения и тестирования модели. [↑](#footnote-ref-14)
15. МГК (метод главных компонент) – это метод понижения размерности данных, который используется для уменьшения количества признаков в многомерном наборе данных, с сохранением при этом максимального количества информации. Главные компоненты выбираются таким образом, чтобы они описывали наибольшее количество дисперсии (или информации) в исходных данных. [↑](#footnote-ref-15)
16. Многослойный персептрон (Multilayer Perceptron, MLP) – это класс искусственных нейронных сетей, состоящий из множества взаимосвязанных узлов, называемых нейронами, которые организованы в последовательные слои. [↑](#footnote-ref-16)
17. Word2Vec – это метод машинного обучения, используемый для получения векторных представлений слов из большого набора текстовых данных. [↑](#footnote-ref-17)
18. AdaBoost (Adaptive Boosting) – это алгоритм машинного обучения, который комбинирует несколько слабых (или базовых) моделей, чтобы получить более сильную модель предсказания. Он относится к семейству алгоритмов ансамблевого обучения. [↑](#footnote-ref-18)
19. KNN (k-nearest neighbors) – это алгоритм машинного обучения, который используется для задач классификации и регрессии. Он основывается на том, что объекты, которые находятся близко друг к другу в пространстве признаков, скорее всего принадлежат к одному и тому же классу (в случае классификации) или имеют похожие значения целевой переменной (в случае регрессии). [↑](#footnote-ref-19)
20. Дерево решений – это метод машинного обучения, который использует древовидную структуру, где каждый узел представляет собой признак, каждое ребро представляет собой условие, а каждый листовой узел представляет собой прогноз. [↑](#footnote-ref-20)
21. Коэффициент корреляции Спирмена (Spearman's rank correlation coefficient) — это мера степени монотонной связи между двумя переменными. Он вычисляется по формуле, которая использует ранговые значения двух переменных. [↑](#footnote-ref-21)
22. Коэффициент корреляции Кендалла (Kendall's tau rank correlation coefficient) — это также мера степени монотонной связи между двумя переменными, но он учитывает не только конкретные ранговые значения, но и их количество. Он измеряет согласованность между ранжированиями двух переменных. [↑](#footnote-ref-22)
23. Гиперпараметры – это параметры модели машинного обучения, которые не оптимизируются непосредственно в процессе обучения модели, а устанавливаются до начала процесса обучения и контролируют ее поведение. [↑](#footnote-ref-23)
24. Проблема trade-off (или дилемма trade-off) – это ситуация, при которой улучшение одной характеристики системы приводит к ухудшению другой характеристики, и наоборот. [↑](#footnote-ref-24)
25. Паттерн (англ. pattern) означает образец, шаблон, повторяющийся элемент или структуру. [↑](#footnote-ref-25)