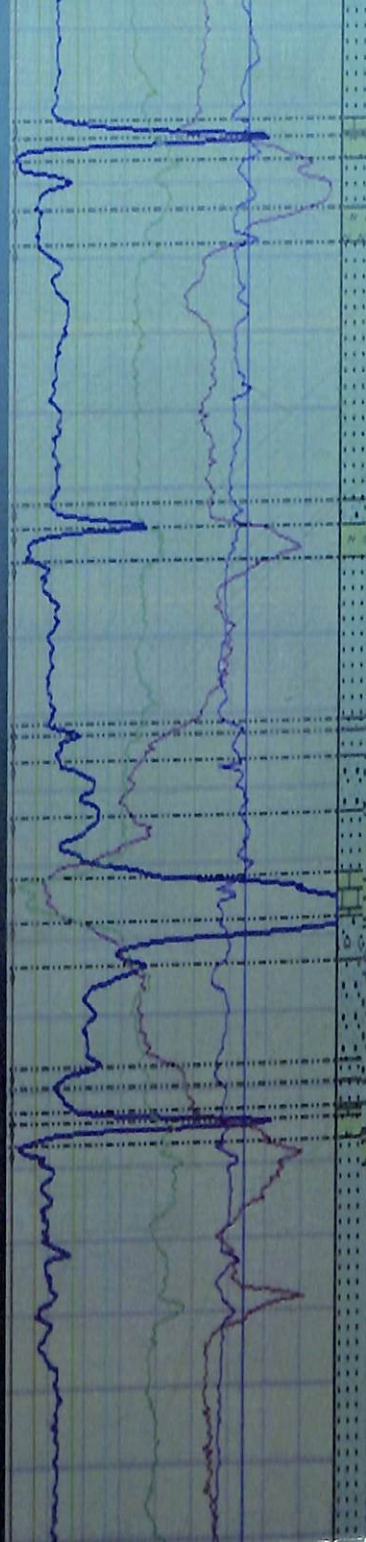
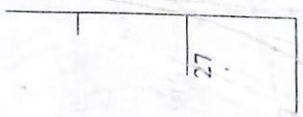
Мухамедиев Р. И.

Методы машинного обучения в задачах геофизических исследований





Мухамедиев Р. И.

Методы машинного обучения в задачах геофизических исследований

Алматы
2016

Мухамедиев Р. И. Методы машинного обучения
в задачах геофизических исследований. – Алматы. – 2016. – 200 с.
ISBN 978-9934-14-876-7

В монографии рассматриваются вопросы применения методов машинного обучения в решении задач геофизического исследования скважин по добыче урана в Республике Казахстан.

Обсуждаются аспекты развития информационно-коммуникационных технологий, задачи геофизического исследования скважин, применяемые программные средства, способы автоматизации решения задачи литологической классификации на базе методов машинного обучения и полученные экспериментальные результаты.

Книга предназначена для специалистов в области информационных технологий, разрабатывающих интеллектуальные системы в добывающей промышленности, докторантов, аспирантов и студентов старших курсов, интересующихся практическими приложениями методов машинного обучения и интеллектуального анализа данных.

Одобрено Ученым Советом университета имени Сулеймана Демиреля протокол №10 от 26.05.2016г.

Рецензенты: д.т.н., проф. Амиргалиев Е. Н., dr.sc.ing., проф. Гопеенко В. И., д.т.н., проф. Никульчев Е.В.

Табл. 25. Ил. 40. Библиогр.: 133 назв.

ISBN 978-9934-14-876-7

Содержание

ПРЕДИСЛОВИЕ

Казахстан является мировым лидером в производстве урана, с 2009 года обеспечивая более трети общего объема данного ресурса. При этом наблюдается постоянный рост в объемах производства – за последние семь лет производство урана в Казахстане возросло почти в 3,5 раза.

Добыча урана на месторождениях Казахстана ведется методом подземного скважинного выщелачивания, который относится к числу малозатратных, экологически безвредных способов добычи. При этом экономические показатели процесса добычи зависят от скорости и точности интерпретации геофизических данных. Ошибочный или неточный анализ геофизических данных приводит к потерям скважин, неоправданным трудозатратам и в конечном счете снижает экономические показатели добычи.

Использование информационных технологий в процессе интерпретации данных геофизического исследования скважин (ГИС) урана позволяет отойти от традиционных методов сбора и обработки информации, решая проблему обработки больших объемов данных и уменьшая субъективность интерпретации отдельными экспертами. Однако классические программные средства, позволяющие автоматизировать процессы сбора, обработки и хранения данных ГИС, не обладают достаточными диагностирующими способностями.

Возможность интерпретации результатов ГИС могут предоставить современные разработки в области так называемого слабого искусственного интеллекта (ИИ). Таким инструментом на текущий момент являются методы машинного обучения (МО), точнее, методы обучения с учителем *(supervised learning),* к числу которых относится несколько групп алгоритмов, включая и искусственные нейронные сети (ИНС). Несмотря на относительную простоту их построения и функционирования, системы машинного обучения позволяют накапливать уже известные закономерности ГИС, обобщать факты и давать вполне корректные оценки в ситуациях, когда на входе представлены зашумленные данные.

На практике при применении МО для анализа данных внимание исследователей должно акцентироваться не только на самом процессе интерпретации, но и на этапах подготовки исходных данных. Качество, состав и формат входных данных напрямую влияют на процесс обучения системы и впоследствии -- на качество классификации.

Наравне с предобработкой данных значительно улучшить результат позволяет постобработка данных интерпретации, учитывающая специфику данных ГИС и физические особенности области исследования, результаты работы других алгоритмов распознавания, в том числе имитирующих методику распознавания экспертами. Правильная подготовка (предобработка) исходных данных для интерпретации и постобработка

результатов интерпретации позволят повысить надежность литологической классификации, что, в свою очередь, улучшит экономические показатели всего процесса добычи урана.

Таким образом, для комплексной высококачественной автоматической интерпретации данных ГИС на основе изучения объекта исследования, которым в данном случае является процесс интерпретации данных геофизического исследования скважин, требуется разработать методы, алгоритмы и программное обеспечение автоматизации процесса классификации данных геофизического исследования скважин на урановых месторождениях пластово-инфильтрационного типа с применением методов машинного обучения.

Для достижения указанной цели необходимо выполнить следующие задачи:

- оценить актуальность и возможность использования методов машинного обучения в задачах интерпретации данных геофизического исследования скважин;
- разработать методы предварительной обработки данных, релевантные данным предметной области (в данном случае это данные каротажа);
- проанализировать показатели оценки точности алгоритмов машинного обучения;
- выполнить сравнительный анализ алгоритмов машинного обучения, пригодных для решения задачи классификации данных геофизического исследования скважин;
- разработать метод синтеза результатов алгоритмов классификации;
- разработать программное обеспечение системы распознавания литологического состава скважин на урановых месторождениях.

Результаты научных исследований существенно зависят от применяемой методологии исследования. В свою очередь методология исследования определяется некоторыми базовыми положениями. В нашем случае были сделаны три предположения. Во-первых, предположение об отсутствии строгой математической модели литологического расчленения скважин по данным каротажа. Это предположение повлекло за собой необходимость использования методов, работающих в слабоформализованной области, а именно методов машинного обучения. Второе предположение касалось отсутствия строго обоснованной методики предобработки данных для их применения в машинном обучении. Это предположение повлекло за собой использование вычислительных экспериментов для выявления лучших методов предобработки. Третье предположение касалось реализации программной системы, точнее, отсутствия таковой системы в настоящий момент. В результате

потребовалось создание прототипов отдельных компонентов программной системы интерпретации каротажных данных.

Основываясь на этих базовых предположениях, при выполнении исследований были использованы: информационно-аналитический анализ, экспертное оценивание, вычислительные эксперименты, моделирование и разработка прототипов программной системы.

В рамках информационно-аналитического анализа исследована проблема по данным отечественных и зарубежных авторов и иным открытым источникам, в том числе доступным корпоративным материалам, выполнена классификация существующих технологий в области исследования, собраны данные для анализа, рассмотрены основные алгоритмы машинного обучения, предназначенные для решения задачи классификации литологических слоев урановых месторождений.

Экспертное оценивание использовано как источник данных для алгоритмов машинного обучения, так и объект исследования. В последнем случае проведено сравнение результатов классификации, выполненных отдельными экспертами, и оценены сравнительные показатели качества классификации.

В ходе вычислительных экспериментов проанализированы различные методы предобработки данных, алгоритмы машинного обучения с использованием как синтезированных, так и реальных данных каротажа различных месторождений.

В процессе исследования рассмотрены математические модели методов машинного обучения.

Прототипирование компонентов программной системы проводилось на всех этапах исследования в процессе вычислительных экспериментов, что позволило создать законченную программную систему.

Решение перечисленных выше задач последовательно описано в разделах работы.

В первом разделе проанализированы сложившиеся научные предпосылки, связанные с развитием информационно-коммуникационных технологий (ИКТ), построена классификация современных доменов ИКТ, дана оценка месту и возможностям интеллектуальных методов, а также экономическому эффекту, связанному с применением методов машинного обучения в задачах интерпретации данных каротажа скважин.

Во втором разделе описаны основные виды каротажа и применяемые для их обработки программные средства[1].

В третьем разделе приведена таксономия методов машинного обучения, описаны математические модели методов, схема настройки системы машинного обучения на решение задачи классификации, способы предобработки данных и показатели оценки качества работы системы.

[1] Раздел написан совместно с Я. И. Кучиным.

В четвертом разделе приведены результаты вычислительных экспериментов, выполненных с применением методов машинного обучения, в том числе в сравнении с экспертным оцениванием.

В пятом разделе описан прототип системы автоматической интерпретации.

В заключении кратко описаны общие итоги работы.

В результате решения описанных задач обоснована возможность использования методов и алгоритмов МО в задаче интерпретации данных геофизического исследования скважин на пластово-инфильтрационных месторождениях урана. Разработаны и апробированы методы предварительной обработки данных, выбраны алгоритмы МО, релевантные поставленной задаче. Разработано программное обеспечение для автоматизированной интерпретации данных каротажа.

Настоящая работа не могла бы осуществиться без вклада участников научной группы проекта № 2318/ГФ3. В частности, анализ научного домена информационно-коммуникационных технологий, представленный в первом разделе, был выполнен с участием А. Абдильмановой. Анализ систем обработки данных ГИС и постановка многих задач исследования выполнены при активном участии Я. И. Кучина. Исследование методов предобработки данных и методов машинного обучения, описанных в четвертом разделе, проводилось К. Якуниным, П. Гриценко, Ж. Нурушевым, С. Саиновой, А. Абдильмановой. Разработка системы интерпретации данных каротажа осуществлялась с участием С. Х. Искакова, З. Исабаева, Ж. Нурушева, П. Гриценко, К. Якунина. Подготовка приложений и общая верстка работы выполнялись Е. Л. Мухамедиевой.

Автор выражает искреннюю признательность всем перечисленным сотрудникам, а также руководству ИИВТ МОН РК и лично профессорам М. Н. Калимолдаеву, Е. Н. Амиргалиеву, доц. Н. Р. Юничевой за поддержку в процессе выполнения исследований.

Написание любой книжки – это процесс, в который волей-неволей вовлечены самые близкие люди, и поэтому автору хочется выразить слова благодарности супруге Елене, чье деятельное участие, помощь и стоическое терпение позволили завершить эту работу.

Взыскательный читатель, скорее всего, найдет в тексте работы много недостатков. Ну что же, как говорится, «не ошибается только тот, кто ничего не делает». Конструктивные замечания, направленные по адресу ravil@inbox.lv с пометкой *book*, будут с благодарностью приняты автором.

Рисунки, приведенные в книге, в полноцветном виде можно найти на сайте www.geoml.info. Там же размещены полные версии приложений.

1. СОВРЕМЕННЫЕ ТЕНДЕНЦИИ РАЗВИТИЯ ИКТ

С некоторой долей условности можно утверждать, что примерно каждые 10 лет совершается смена парадигм развития информационно-коммуникационных технологий. Эти изменения происходили с момента появления компьютеров, а затем их широкого использования в качестве машин по переработке информации и заканчивая тем, что мы назвали эпохой разумных сервисов и «думающих» машин (рисунок 1.1).

Рисунок 1.1. Парадигмы развития ИКТ

В настоящее время в связи с развитием глобального информационного общества *(Global Information Society)* [1] происходит переход на новый уровень всех составляющих ИКТ. Изменениям подвергаются все три больших области исследований и технологий (рисунок 1.2): «Облако» *(Cloud)*, под которым понимают ресурсы информационно-вычислительных систем, «Труба» *(Pipe)* – коммуникационная среда или сеть и «Устройства» *(Devices)*. Каждый из этих доменов характеризуется своими «смыслами», описывающими соответствующие компоненты, системы, области исследований и т.п.

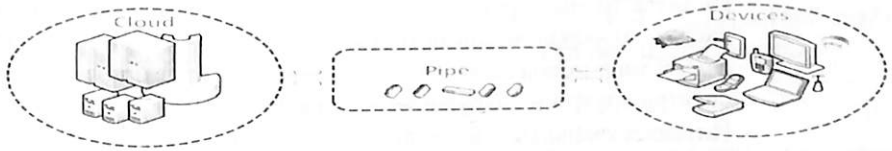


Рисунок 1.2. Основные домены ИКТ

Объединение технологий беспроводных сенсорных сетей *(Wireless Sensor Network)*, систем межмашинной коммуникации *(Machine-to-Machine – M2M)*, широкополосного доступа к сети на базе новых протоколов связи, технологий встроенных систем [2] и мобильного программирования [3] составит основу для разработки эффективных информационных систем. Эти технологии обеспечат высокий уровень надежности и малые временные задержки при дистанционном мониторинге и передаче данных разного объема.

1.1 Новые коммуникации: вызовы и перспективы

Рассматривая текущее состояние коммуникационной технологии, можно констатировать резкий рост трафика (на 30% в год), объема контента (до 10^21 байт к 2020 году) и количества подключаемых устройств (до 50 млрд. к 2020 году) [1]. Такой быстрый рост предполагает новые подходы к построению сетей нового поколения. Необходимы новые стандарты и новый способ использования сетей. Цель состоит в создании интеллектуальной адаптивной инфокоммуникационной структуры, которая должна повысить коэффициент использования современных инфокоммуникаций, составляющий сегодня только от 10 до 20% от теоретически достижимых [4]. Кроме этого, требования быстро развивающегося интернета вещей *(Internet of Things – IoT)* приводят к появлению новых протоколов связи, объединенных общим термином *5G*. Необходимость новых протоколов вызвана тем, что существует ряд приложений, которые требуют очень малых временных задержек в сети, высокого уровня надежности сети и быстрой передачи данных разного объема. Эти цели планируется достичь путем внедрения комплекса современных технологий и повышения качества использования радиочастотного ресурса [5] (рисунок 1.3):

- Новый радиоинтерфейс с малыми сотами *New Air Interface (Small Cells)* с опорой на новые формы колебаний *(New Wave Form)*, новые формы дуплекса *(New Duplexing)*, облегченные протоколы канального уровня *(Light MAC)*, высокие порядки модуляции *(Higher Order Modulation)*, эффективные методы компенсации внутрисистемных помех *(Interference Cancelation/Utilization)*, многомерные антенные системы *(Massive MIMO)* [6].
- Новая архитектура в радиосети *(New NW Architecture)* – распределение и управление в гетерогенной архитектуре *HetNet*, реконфигурируемые радио- и сетевые элементы.

- Радиочастотный ресурс – использование высоких диапазонов частот, включая миллиметровый, новый режим лицензирования, совместное использование спектра, комбинированное применение спектра внутри и снаружи помещений.
- Интеллектуальные и адаптивные сети – стохастическое и адаптивное использование сетевых ресурсов, обнаружение доступного спектра и его использование на принципах когнитивного радио, самоуправляемые и автоматизированные сети *(SDN – Software-Defined Network)*.

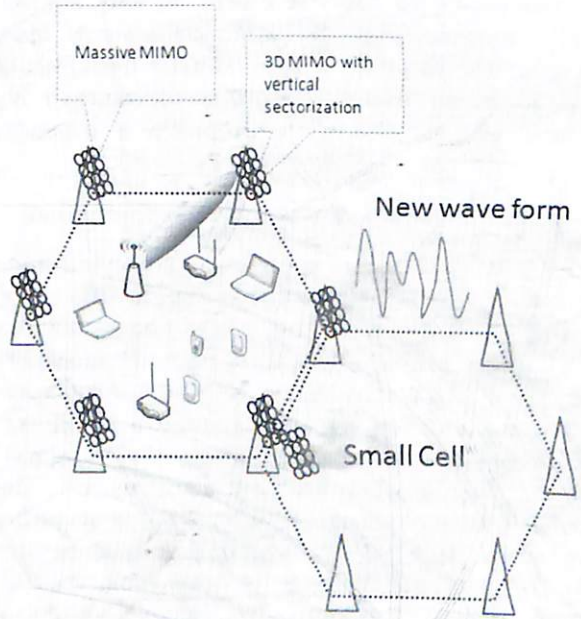


Рисунок 1.3. Современные технологии телекоммуникации

По оценкам экспертов [5] [6] [7] [8], сети *5G* дадут возможность:

- роста объема передаваемых данных более чем в 1000 раз в каждой области обслуживания;
- увеличения от 10 до 100 раз типовых скоростей передачи данных на стороне пользователя;
- удлинения в 10 раз срока службы батарей для абонентских устройств с низкой мощностью;
- снижения в 10 раз задержки в цепочке *end-to-end* (менее 1 мс против 10 мс в *LTE*).

Собираемые «быстрыми» сетями данные будут обрабатываться совокупностью информационных систем, объединенных понятием «облако».

Использование новых телекоммуникационных систем – следующий шаг в развитии информационных систем в добывающей промышленности.

1.1.1 Технологии в «облаке»

Ключевыми аспектами информационных технологий становятся способность системы обрабатывать большие данные [7] [9], то есть огромные совокупности разнородной и часто неструктурированной информации, «разумность», поскольку без соответствующих алгоритмов решение первой задачи малореально, и работа напрямую с устройствами.

1.1.1.1 Большие данные

Современная индустрия обработки данных продемонстрировала неготовность к работе с действительно большими объемами данных [10] [8]. Ответом послужило появление в 2008 году новой парадигмы – Больших данных *(Big Data)* [9] [11]. Решение проблемы поиска закономерностей в неструктурированных данных за ограниченное время потребовало новых подходов. В области СУБД предпочтение отдается не традиционным *SQL*-системам, а стеку обработки: *NoSQL, MapReduce, Hadoop* и графовым СУБД [12] [13] [14]. Средства аналитики и оценки строятся на базе *Hadoop (Hive* – бизнес-аналитика, *WibiData* – веб-аналитика в *Hadoop, SkyTree* – аналитическая обучающаяся платформа, *Platfora* – система формирования заданий к *Hadoop, HiBench* – набор тестов производительности, *HiTune* – средство динамического анализа производительности распределенных приложений *Hadoop)*. Некоторые университеты, например *MIT*, организовали лаборатории по работе с большими данными. Лаборатория *MIT* начала работу по проекту, объединяющему большие данные и искусственный интеллект *(http://bigdata.csail.mit.edu/)*. Тем не менее у информационной индустрии пока нет целостного видения инструментариев обработки и стеков обработки таких данных. Сферы приложений больших данных широки, начиная от управления информационной инфраструктурой (в т.ч. интеллектуального), анализа социальных явлений [15] и заканчивая обработкой данных государственных служб в системах электронного правительства [16] и др.

1.1.1.2 Машинное обучение

Машинное обучение *(Machine Learning – ML)* – это ключевое направление, в рамках которого происходит поиск технологий для обработки данных в «облаке». Основные научные направления, связанные с машинным обучением:

- Имитация методов мышления человека, например, в виде нейроморфного компьютера *IBM* [17]. Это сфера сильного искусственного интеллекта *(Strong AI)*.

- Синтез алгоритмов распознавания, классификации и т.п. для достижения лучшего качества распознавания *(Ensemble of Learning Algorithms)*. Математическая база этого направления заложена в работах Ю. И. Журавлева [18] [19].
- Распознавание смысла и скрытых закономерностей *(Data Mining)*.
- Мультиагентные [20] [21] [22] и иные распределенные самоорганизуемые системы: аниматы, мобильные роботы и т.п.

МО – один из способов анализа данных, который используется в самых разных сферах человеческой деятельности. МО применяется и в сфере межмашинного и человекомашинного взаимодействия, где к числу сравнительно новых и бурно развивающихся направлений его применения относится интернет вещей.

1.1.1.3 Интернет вещей

Интернет вещей возник в 2008–2009 годах вместе с появлением новых видов радиочастотной идентификации и коммуникации *(RFID – Radio Frequency Communication)*, беспроводных сенсорных сетей *(WSN – Wireless Sensor Network)*, всепроникающих сенсорных сетей, коммуникаций ближнего радиуса действия *(NFC– Near Field Communication)* и межмашинных коммуникаций *(M2M)*. Интеграция указанных технологий с интернетом привела к тому, что число предметов (приборов, устройств, вещей), подключенных к интернету, превысило число пользователей-людей. В настоящий момент несколько крупных организаций координируют работу в области *IoT* [23]. Количество приложений *IoT* стремительно растет [24].

Элементы *IoT* в сочетании с новыми телекоммуникационными технологиями и интеллектуальными методами обработки данных должны придать новое качество информационным системам, когда многие сложные задачи будут решаться полностью автоматически, существенно снижая нагрузку на экспертов. Реализация этого потенциала зависит от правильного выбора научных направлений, исходя из места Республики Казахстан в международной научной кооперации.

1.1.2 Место Республики Казахстан в международном научном процессе

В 2013 году Республика Казахстан занимала 70–90-е позиции среди 144 стран, оцениваемых международными рейтингами. Международный экономический форум разместил Казахстан на 90-м месте по доступности современных технологий вслед за Турцией, Латвией и Азербайджаном. В числе первых двадцати стран по этому показателю находятся Швеция, Финляндия, Гонконг, Франция и Германия [25]. Казахстан был на 104-й позиции по наличию ученых и инженеров, на 71-м месте по поддержке высоких технологий правительством, на 90-м по взаимодействию промышленности, университетов в научных исследованиях.

Выделив группы стран в соответствии с таблицей «12.02 *Quality of scientific research institutions*» из [25], можно определить в них группу лидеров, представленных следующим списком: Израиль, Швейцария, Великобритания, Бельгия, Катар, США, Австралия, Нидерланды, Швеция, Германия, япония и Сингапур (Israel, Switzerland, United Kingdom, Belgium, Qatar, United States, Australia, Netherlands, Sweden, Germany, Japan и Singapore). Аналогичным образом, вторая группа представляет собой страны со средними показателями: Южная Корея, Норвегия, Малайзия, Россия, Франция, Китай, Турция, Латвия, Азербайджан и Казахстан (Korea, Rep Norway, Malaysia, Russian Federation, France, China, Turkey, Latvia, Azerbaijan и Kazakhstan).

Для сравнения структуры публикационной активности в разных странах построена Таблица 1.1, которая демонстрирует пять главных направлений научных публикаций в различных странах в течение трех временных периодов[2]. Области исследований расположены в порядке убывания числа публикаций в данной области в конкретной стране.

Характерно, что распределение числа публикаций по областям исследований в странах лидирующей группы достаточно похоже. На первом месте – медицина, на 2–3-м местах – инженерные науки и биохимия, далее – компьютерные и социальные науки, физика. Среди ведущих стран можно отметить существенное увеличение количества публикаций в области компьютерных наук. В течение рассматриваемых трех временных периодов место компьютерных наук изменилось с 6.75 до 4.75 (цифры озачают место, занимаемое публикациями по компьютерным наукам, усредненное по всем странам первой группы). Значительное развитие в последние годы получили также социальные науки, биохимия и молекулярная биология.

В тоже время, в странах второй группы структура публикационной активности более разнообразна. Рост числа публикаций по компьютерным наукам также отмечается, но их доля ниже - с 8.2 по 6.07. Значительную долю по сравнению со странами первой группы занимают публикации в области инженерии, физики, математики и химии.

Сравнивая Казахстан со странами с похожими природными условиями, становится видно, что в стране меньше исследований в области агрокультуры, компьютерных и социальных наук. Информационно-коммуникационные технологии, где по ряду направлений наблюдается взрывной рост числа публикаций, становятся локомотивом развития во многих сферах хозяйственной жизни, производства и науки.

[2] Таблица 1.1 построена по данным базы Scopus

Таблица 1.1. Изменение структуры публикаций

Countries (leading countries and others based on the Global Competitiveness Report 2012-2013)	Subject of the greatest number of publications in 2004-2005 based on data from http://www.scimagojr.com (map generator)T						Subject of the greatest number of publications in 2007-2008 based on data from http://www.scimagojr.com (map generator)						Subject of the greatest number of publications in 2011-2012 based on data from http://www.scimagojr.com (map generator)					
Leading Countries	1	2	3	4	5	RCS	1	2	3	4	5	RCS	1	2	3	4	5	RCS
Israel	Medi	Physi	Bioch	Engi	Math	6	Medi	Phys	Bioc	Engi	Com	5	Medic	Phys	Bioch	Com	Engi	4
Switzerland	Medi	Physi	Bioch	Engi	Mate	7	Medi	Phys	Bioc	Engi	Mate	6	Medic	Phys	Bioch	Engi	Com	5
United Kingdom	Medi	Engin	Bioch	Phys	Mate	9	Medi	Engi	Bioc	Phys	Soci	6	Medic	Bioch	Engi	Soci	Phys	6
Belgium	Medi	Physi	Engi	Mate	Chei	8	Medi	Engi	Phys	Bioch	Com	5	Medic	Engi	Bioch	Phys	Com	5
Qatar	Medi	Engin	Phys	Math	Chei	6	Medi	Engi	Com	Math	Phys	3	Medic	Engi	Com	Math	Soc	3
United States	Medi	Engin	Bioch	Phys	Mate	6	Medi	Engi	Bioc	Phys	Com	5	Medic	Engi	Bioch	Phys	Com	5
Australia	Medi	Engin	Bioch	Agri	Com	5	Medi	Engi	Com	Bioch	Agri	3	Medic	Engi	Bioch	Agri	Soc	6
Netherlands	Medi	Bioch	Engi	Phys	Mate	7	Medi	Bioch	Engi	Phys	Com	5	Medic	Bioch	Engi	Phys	Com	5
Sweden	Medi	Bioch	Engi	Phys	Mate	8	Medi	Bioch	Engi	Phys	Mate	6	Medic	Bioch	Engi	Phys	Com	5
Germany	Medi	Physi	Engi	Bioch	Mate	8	Medi	Phys	Engi	Bioch	Mate	7	Medic	Phys	Engi	Bioc	Com	5
Japan	Medi	Engin	Phys	Mate	Bioc	7	Medi	Engi	Phys	Mate	Bioch	7	Medic	Engi	Phys	Bioc	Mate	6
Singapore	Engi	Mater	Phys	Com	Medi	4	Engi	Com	Phys	Medi	Mate	2	Engin	Com	Medi	Mate	Phys	2
Average rating of CS						6.75						5						4.75
Other Countries	1	2	3	4	5	RCS	1	2	3	4	5	RCS	1	2	3	4	5	RCS
24 Korea, Rep	Engi	Physi	Mate	Medi	Com	5	Engi	Phys	Mate	Medi	Com	5	Engin	Medi	Mate	Phys	Com	5
Norway	Medi	Agric	Engi	Bioch	Eart	9	Medi	Engi	Agri	Bioch	Eart	6	Medic	Engi	Agri	Bioc	Soc	7
28 Malaysia	Engi	Medic	Mate	Chei	Phys	6	Engi	Com	Medi	Mate	Phys	2	Engin	Com	Mate	Phys	Medi	2
Russian Federation	Phys	Mater	Engi	Chei	Math	9	Phys	Mate	Chei	Engi	Math	8	Physi	Mate	Chei	Engi	Math	11
France	Medi	Physi	Engi	Bioch	Mate	8	Medi	Phys	Engi	Bioch	Com	5	Medic	Phys	Engi	Bioc	Com	5
China	Engi	Physi	Mate	Chei	Com	5	Engi	Phys	Mate	Com	Chei	4	Engin	Com	Mate	Phys	Medi	2
Turkey	Medi	Engin	Mate	Bioch	Chei	11	Medi	Engi	Phys	Mate	Chei	8	Medic	Engi	Agri	Phys	Mate	7
Latvia	Phys	Mater	Engi	Chei	Bioc	9	Phys	Engi	Mate	Chei	Math	6	Engin	Mate	Phys	Agri	Com	5
Azerbaijan	Medi	Physi	Engi	Chei	Mate	11	Medi	Phys	Chei	Engi	Mate	7	Physi	Engi	Chei	Medi	Mate	6
Kazakhstan	Phys	Chem	Mate	Engi	Math	10	Phys	Chei	Mate	Engi	Math	13	Physi	Mate	Engi	Earth	Math	12
Average rating of CS						8.2						6.3						6.07

Примечание к таблице 1.1:

CS, RCS	Computer Science, Rating of CS	Che	Chemistry
Med	Medicine	Mat	Materials Science
Phy	Physics & Astronomy	Com	Computer Science
Eng	Engineering	Math	Maths
Agr	Agricultural & Biological Sciences	Soc	Social Sciences
Bio	Biochemistry, Genetics & Molecular Biology	Earth	Earth & Planetary Sciences

1.1.1 Краткий анализ изменений числа научных публикаций в домене ИКТ

Современные системы высокого уровня строятся с использованием нескольких взаимосвязанных информационных технологий: машинное обучение, мультиагентные системы, системы сбора и обработки данных, включая большие данные, облачные и кластерные системы, геоинформационные системы и т.д.

В последние годы домен ИКТ пополнился новыми направлениями исследований, к числу которых относятся *Big Data, Bioinformatics (Computational Biology), Cloud Computing, Cyber-Physical Systems, Embedded Systems, Information Security, Internet of Things, Human-Machine Systems, Mobile Computing, Machine Learning, Machine-to-Machine, Multi-Agent Systems, Neural Networks, Robotics, Visualization, Augmented Reality, SDN, 5G, e-Governance, Smart City, Smart Grid* и др.

Выбор указанных областей исследований для анализа относительно субъективен. Часть этих концепций относится к научным исследованиям, другие, по существу, являются технологиями. В таблице 1.2 показано ежегодное число публикаций в каждой из упомянутых областей. Колонка 2 таблицы содержит ключевые слова, использованные в процессе поиска. Следует отметить, что данные получены в ноябре 2014 года.

Существенный рост числа публикаций отмечен в таких доменах, как *Big Data, Augmented Reality.* Наблюдается «второе рождение» машинного обучения и визуализации, рост *5G (IMT-Advanced), Internet of Things* и некоторое снижение интереса в областях *Cloud Computing* и «классическая» робототехника.

Для анализа взаимосвязей между субдоменами ИКТ выполнен поиск статей по двум ключевым словам в базе научных публикаций *EBSCO (Library, Information Science & Technology Abstracts, Academic Search Complete).*

В результате получена таблица, в которой по диагонали дано количество публикаций в выбранной области исследований, а числа, которые стоят на пересечении, показывают количество публикаций с двумя ключевыми словами. Таблица 1.3 отражает бинарные отношения между исследуемыми доменами ИКТ.

В качестве дополнительной иллюстрации граф на рисунке 1.4 показывает упомянутые взаимосвязи по данным *www.Sciencedirect.com.* Запросы по числу публикаций учитывали период с 1995 по 2015 год. Число публикаций указано в тысячах.

Для визуализации относительной величины компонентов ИКТ и их связей построена семантическая сеть [26] [27] (рисунок 1.5) на основе исследования количества публикаций в каждой предметной области. Поскольку количество публикаций в каждой из рассматриваемых областей значительно отличается (разница достигает двух порядков), семантические

Таблица 1.1. Изменение структуры публикаций

Countries (leading countries and others based on the Global Competitiveness Report 2012-2013)	Subject of the greatest number of publications in 2004-2005 based on data from http://www.scimagojr.com (map generator)T						Subject of the greatest number of publications in 2007-2008 based on data from http://www.scimagojr.com (map generator)						Subject of the greatest number of publications in 2011-2012 based on data from http://www.scimagojr.com (map generator)					
Leading Countries	1	2	3	4	5	RCS	1	2	3	4	5	RCS	1	2	3	4	5	RCS
Israel	Medi	Physi	Bioch	Engi	Math	6	Medi	Phys	Bioc	Engi	Com	5	Medic	Phys	Bioch	Com	Engi	4
Switzerland	Medi	Physi	Bioch	Engi	Mate	7	Medi	Phys	Bioc	Engi	Mate	6	Medic	Phys	Bioch	Engi	Com	5
United Kingdom	Medi	Engin	Bioch	Phys	Mate	9	Medi	Engi	Bioc	Phys	Soci	6	Medic	Bioch	Engi	Soci	Phys	6
Belgium	Medi	Physi	Engi	Mate	Chei	8	Medi	Engi	Phys	Bioc	Com	5	Medic	Engi	Bioch	Phys	Com	5
Qatar	Medi	Engin	Phys	Math	Chei	6	Medi	Engi	Com	Math	Phys	3	Medic	Engi	Com	Math	Soc	3
United States	Medi	Engin	Bioch	Phys	Mate	6	Medi	Engi	Bioc	Phys	Com	5	Medic	Engi	Bioch	Phys	Com	5
Australia	Medi	Engin	Bioch	Agric	Com	5	Medi	Engi	Com	Bioc	Agric	3	Medic	Engi	Bioch	Agric	Soc	6
Netherlands	Medi	Bioch	Engi	Phys	Mate	7	Medi	Bioc	Engi	Phys	Com	5	Medic	Bioch	Engi	Phys	Com	5
Sweden	Medi	Bioch	Engi	Phys	Mate	8	Medi	Bioc	Engi	Phys	Mate	6	Medic	Bioch	Engi	Phys	Com	5
Germany	Medi	Physi	Engi	Bioch	Mate	8	Medi	Phys	Engi	Bioc	Mate	7	Medic	Phys	Engi	Bioc	Com	5
Japan	Medi	Engin	Phys	Mate	Bioc	7	Medi	Engi	Phys	Mate	Bioc	7	Medic	Engi	Phys	Bioc	Mate	6
Singapore	Engi	Mater	Phys	Com	Medi	4	Engi	Com	Phys	Medi	Mate	2	Engin	Com	Medi	Mate	Phys	2
Average rating of CS						6.75						5						4.75
Other Countries	1	2	3	4	5	RCS	1	2	3	4	5	RCS	1	2	3	4	5	RCS
24 Korea, Rep	Engi	Physi	Mate	Medi	Com	5	Engi	Phys	Mate	Medi	Com	5	Engin	Medi	Mate	Phys	Com	5
Norway	Medi	Agric	Engi	Bioch	Earth	9	Medi	Engi	Agric	Bioc	Earth	6	Medic	Engi	Agric	Bioc	Soc	7
28 Malaysia	Engi	Medic	Mate	Chei	Phys	6	Engi	Com	Medi	Mate	Phys	2	Engin	Com	Mate	Phys	Med	2
Russian Federation	Phys	Mater	Engi	Chei	Math	9	Phys	Mate	Chei	Engi	Math	8	Physi	Mate	Chei	Engi	Math	11
France	Medi	Physi	Engi	Bioch	Mate	8	Medi	Phys	Engi	Bioc	Com	5	Medic	Phys	Engi	Bioc	Com	5
China	Engi	Physi	Mate	Chei	Com	5	Engi	Phys	Mate	Com	Chei	4	Engin	Com	Mate	Phys	Med	2
Turkey	Medi	Engin	Mate	Bioch	Chei	11	Medi	Engi	Phys	Mate	Chei	8	Medic	Engi	Agric	Phys	Mate	7
Latvia	Phys	Mater	Engi	Chei	Bioc	9	Phys	Engi	Mate	Chei	Math	6	Engin	Mate	Phys	Agric	Com	5
Azerbaijan	Medi	Physi	Engi	Chei	Mate	11	Medi	Phys	Chei	Engi	Mate	7	Physi	Engi	Chei	Medi	Mate	6
Kazakhstan	Phys	Chem	Mate	Engi	Math	10	Phys	Chei	Mate	Engi	Math	13	Physi	Mate	Engi	Earth	Math	12
Average rating of CS						8.2						6.3						6.07

Примечание к таблице 1.1:

CS, RCS	Computer Science, Rating of CS	Che	Chemistry
Med	Medicine	Mat	Materials Science
Phy	Physics & Astronomy	Com	Computer Science
Eng	Engineering	Math	Maths
Agr	Agricultural & Biological Sciences	Soc	Social Sciences
Bio	Biochemistry, Genetics & Molecular Biology	Earth	Earth & Planetary Sciences

1.1.1 Краткий анализ изменений числа научных публикаций в домене ИКТ

Современные системы высокого уровня строятся с использованием нескольких взаимосвязанных информационных технологий: машинное обучение, мультиагентные системы, системы сбора и обработки данных, включая большие данные, облачные и кластерные системы, геоинформационные системы и т.д.

В последние годы домен ИКТ пополнился новыми направлениями исследований, к числу которых относятся *Big Data, Bioinformatics (Computational Biology), Cloud Computing, Cyber-Physical Systems, Embedded Systems, Information Security, Internet of Things, Human-Machine Systems, Mobile Computing, Machine Learning, Machine-to-Machine, Multi-Agent Systems, Neural Networks, Robotics, Visualization, Augmented Reality, SDN, 5G, e-Governance, Smart City, Smart Grid* и др.

Выбор указанных областей исследований для анализа относительно субъективен. Часть этих концепций относится к научным исследованиям, другие, по существу, являются технологиями. В таблице 1.2 показано ежегодное число публикаций в каждой из упомянутых областей. Колонка 2 таблицы содержит ключевые слова, использованные в процессе поиска. Следует отметить, что данные получены в ноябре 2014 года.

Существенный рост числа публикаций отмечен в таких доменах, как *Big Data, Augmented Reality*. Наблюдается «второе рождение» машинного обучения и визуализации, рост *5G (IMT-Advanced), Internet of Things* и некоторое снижение интереса в областях *Cloud Computing* и «классическая» робототехника.

Для анализа взаимосвязей между субдоменами ИКТ выполнен поиск статей по двум ключевым словам в базе научных публикаций *EBSCO (Library, Information Science & Technology Abstracts, Academic Search Complete)*.

В результате получена таблица, в которой по диагонали дано количество публикаций в выбранной области исследований, а числа, которые стоят на пересечении, показывают количество публикаций с двумя ключевыми словами. Таблица 1.3 отражает бинарные отношения между исследуемыми доменами ИКТ.

В качестве дополнительной иллюстрации граф на рисунке 1.4 показывает упомянутые взаимосвязи по данным *www.Sciencedirect.com*. Запросы по числу публикаций учитывали период с 1995 по 2015 год. Число публикаций указано в тысячах.

Для визуализации относительной величины компонентов ИКТ и их связей построена семантическая сеть [26] [27] (рисунок 1.5) на основе исследования количества публикаций в каждой предметной области. Поскольку количество публикаций в каждой из рассматриваемых областей значительно отличается (разница достигает двух порядков), семантические

Таблица 1.2. Ежегодное число публикаций

Область исследований	Ключевые слова	005	006	007	008	009	010	011	012	013	014
Augmented Reality	Augmented Reality	4010	4490	4760	5230	6050	8320	10 900	12 900	13 200	21 400
	AR	1320	1530	1710	1820	2240	3260	3870	4430	4870	3200
Big Data	Big Data	408	428	486	727	1060	1320	2330	7870	21 200	35 600
Bioinformatics	Bioinformatics	180 000	190 000	214 000	239 000	248 000	233 000	185 000	129 000	78 600	73 000
	Computational Biology	14 000	16 500	18 700	21 800	26 600	30 500	37 400	41 500	35 800	33 100
Cloud Computing	Cloud Computing	1220	963	1330	3990	11 600	21 500	32 000	42 900	43 200	41 100
Cyber-Physical Systems	Cyber-Physical Systems	7	51	65	257	421	833	1510	2370	3390	670
	CPS		8	12	36	60	131	289	427	646	14
Embedded Systems	Embedded Systems	12 900	14 900	17 400	17 800	19 600	21 600	21 700	22 300	22 100	16 000
Information Security	Information Security	15 600	20 200	23 700	28 700	35 100	36 600	37 800	35 900	33 100	23 000
	InfoSec	295	335	319	335	334	392	462	453	474	323
Internet of Things	Internet of Things	168	339	393	797	1140	3010	6300	10 100	12 600	8820
	IoT			9	99	142	667	1780	3050	3920	3200
Human-Machine Systems	Human-Machine Systems	488	458	594	590	864	878	811	1080	1420	828
Mobile Computing	Mobile Computing	14 600	16 300	20 700	28 300	32 200	35 000	34 400	37 700	30 000	28 000
Machine Learning	Machine Learning	47 000	58 300	64 600	70 900	79 600	83 700	83 000	69 500	55 400	96 500
Machine-to-Machine	Machine-to-Machine	1500	1330	1420	1480	1880	2100	2820	4050	4470	2710
Multi-Agent Systems	Multi-Agent Systems	7670	8440	8940	9410	11 000	10 400	10 900	11 300	11 700	8000
Neural Networks	Neural Networks	78 700	91 000	86 000	104 000	96 400	105 000	82 700	85 400	62 500	93 300
Robotics	Robotics	52 500	63 000	62 900	66 700	70 400	70 300	61 300	54 600	42 600	43 600
Visualization	Visualization	77 700	83 500	87 700	91 400	90 800	88 400	76 900	76 200	78 600	95 300
Intelligent Transport System	Intelligent Transport System	610	937	801	670	753	869	878	942	974	532
Self-Organized Network	Self-Organized Network	198	272	274	312	367	359	394	409	399	259
E-Governance	E-Government	1790	1670	2000	2110	2320	2670	2960	3320	3290	1920
Software-Defined Networking	Software Define Networks	3	4	1	6	4	24	52	288	836	810
5G	5G, IMT-Advanced			46	48	59	66	83	110	169	206

Таблица 1.3. Матрица бинарных отношений

Keyword Domain	Augmented reality	Big Data	Bioinformatics	Cloud computing	Cyber-Physical systems	Embedded systems	Information Security	Internet of things	HUMAN-machine systems	Mobile computing	Machine Learning	machine to machine communication	Multi-agent systems	Neural Networks	Robotics	Visualization	Intelligent transport system	E-Governance	E-Government	Software Defined Networking	5G, IMT advanced	Smart city	smart grid
Keyword of research	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	
Augmented reality	1621																						
Big Data	4	3698																					
Computational biology, system biology, molecular biology	0	50	27246																				
Cloud computing	17	181	35	7697																			
Cyber-Physical systems	1	3	0	8	204																		
Embedded systems		6	19	25	23	12926																	
Information Security	0	323	0	163	6	58	14999																
Internet of things	5	38	0	69	10	29	23	1669															
HUMAN-machine systems	16	3	0	4	2	13	5	6	1496														
Mobile computing	53	52	4	254	4	64	49	28	7	4313													
Machine Learning	11	135	665	10	2	77	54	12	52	15	17908												
machine to machine communication		4	2		1	19	4	42	37	6	16	873											
Multi-agent systems	1	2	21	15	4	33	11	8	5	12	124	2	3816										
Neural Networks	9	53	700	20	48	159	46	10	30	19	2962	16	127	68041									
Robotics	93	14	99	43	6	229	7	9	211	32	565	28	187	1185	40316								
Visualization	209	124	431	85	6	82	37	4	38	31	343	10	26	566	595	61274							
Intelligent transport system	0	0	0	0	2	3	4	3	16	1	5	1	5	5	5	0	283						
E-Governance	0	12		17	11	2	53	2	18	1	5	4	5	4	1	10	2	2626					
Software Defined Networking	0	4	0	35	1	2	1	3	2	1	0	2	1	0	0	2	3	0	125				
5G, IMT advanced	1	0	0	1	0	0	2	0	0	2	4	0	1	3	9	1	0	0	0	186			
Smart city		23	0	1	1	8	1	17	1	4	2	5	1	3	6	4	2	9	3	0	638		
smart grid		8	2	22	16	24	18	18	1	3	14	13	36	50	5	9	138	1	7	0	27	2145	

Таблица 1.2. Ежегодное число публикаций

Область исследований	Ключевые слова	005	006	007	008	009	010	011	012	013	014
Augmented Reality	Augmented Reality	4010	4490	4760	5230	6050	8320	10 900	12 900	13 200	21 400
	AR	1320	1530	1710	1820	2240	3260	3870	4430	4870	3200
Big Data	Big Data	408	428	486	727	1060	1320	2330	7870	21 200	35 600
Bioinformatics	Bioinformatics	180 000	190 000	214 000	239 000	248 000	233 000	185 000	129 000	78 600	73 000
	Computational Biology	14 000	16 500	18 700	21 800	26 600	30 500	37 400	41 500	35 800	33 100
Cloud Computing	Cloud Computing	1220	963	1330	3990	11 600	21 500	32 000	42 900	43 200	41 100
Cyber-Physical Systems	Cyber-Physical Systems	7	51	65	257	421	833	1510	2370	3390	
	CPS		8	12	36	60	131	289			670

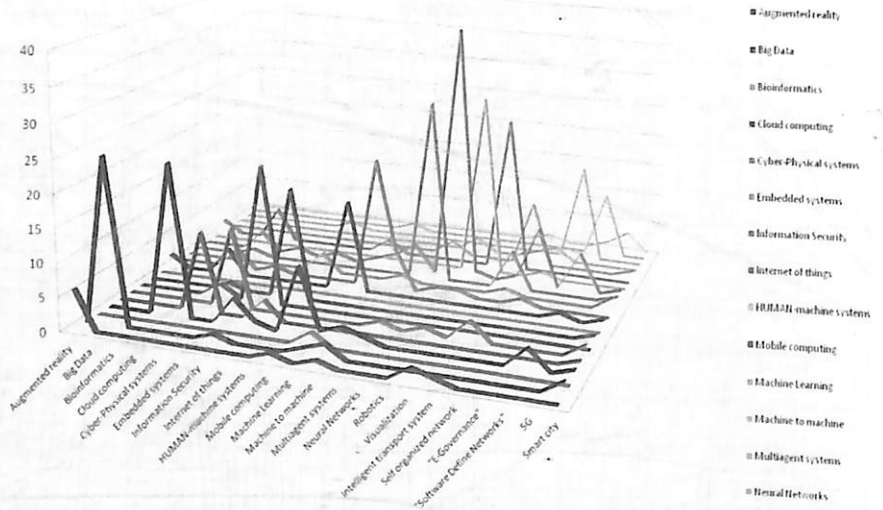


Рисунок 1.4. Взаимосвязи между субдоменами ИКТ (по данным ScienceDirect)

Предложенный метод анализа областей исследований в домене ИКТ впервые описан в [26], а в [28] [29] исследованы взаимосвязи между областями исследований. Выполняя поиск по двум ключевым словам, выявлены близкие понятия, а также группы понятий, в которых исследования являются наиболее актуальными в настоящее время. В частности, используя предложенную метрику оценки близости понятий, выявлены пары близких областей исследований в домене ИКТ.

Построенная на базе предложенной метрики семантическая сеть позволяет выделить группы взаимосвязанных предметных областей, например, робототехнику и машинное обучение, мультиагентные системы и человекомашинное взаимодействие. Облачные и мобильные вычисления рассматриваются в группе с большими данными. Интернет вещей, мобильные вычисления, технологии межмашинного взаимодействия и встроенные системы также составляют группу. Весьма широко в научных исследованиях применяется визуализация, быстро развивающейся частью которой является дополненная реальность. Часто к наиболее перспективным областям с точки зрения получения научных результатов принадлежат именно смежные области, лежащие на стыке понятий.

Например, несмотря на относительно малое количество публикаций по теме *Augmented Reality,* эта область демонстрирует резкий рост в последние годы. Следовательно, *Augmented Reality* в связке с *Visualization* и *Mobile Computing* является весьма перспективной областью

исследований. Можно также отметить *Embedded Systems, Neural Networks* и *Cyber-Physical Systems, Machine Learning* и *Robotics, Machine Learning* и *Multi-Agent Systems.*

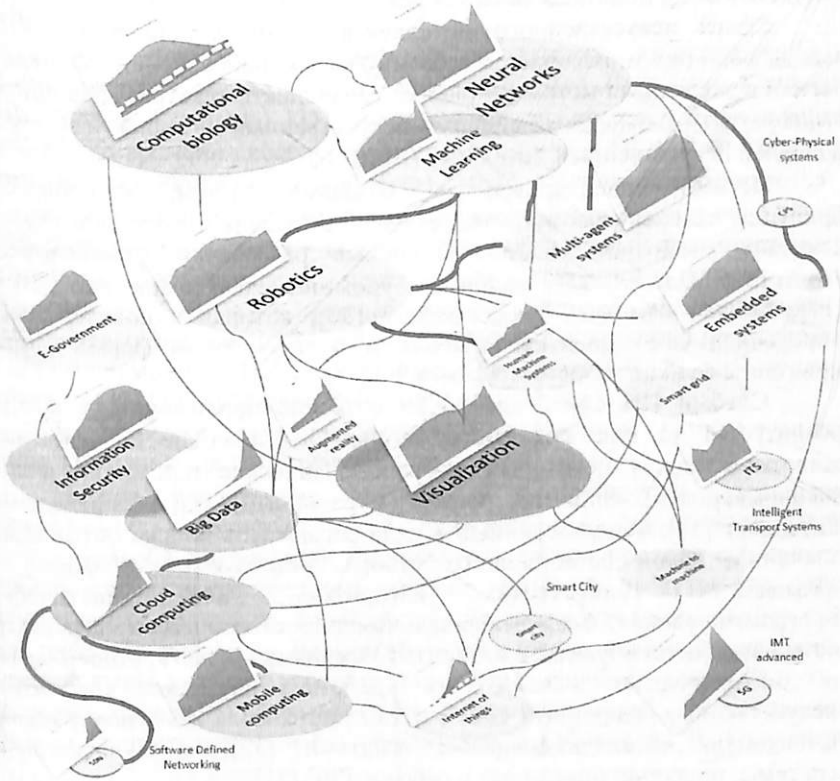


Рисунок 1.5. Семантическая сеть понятий ИКТ

В контексте дальнейшего изложения можно отметить, что проведенный анализ демонстрирует широкое применение методов машинного обучения в самых различных сферах исследований и разработок. Не является исключением и сфера добычи полезных ископаемых, где интеллектуальные методы могут дать существенный экономический эффект за счет повышения скорости и качества принимаемых решений.

1.2 Интеллектуальные методы в процессах добычи полезных ископаемых

Интеллектуальные методы – обширный раздел современной информационной науки, предназначенный для внедрения в практику методов принятия решений, обычно применяемых человеком. Синонимичным понятием является искусственный интеллект.

Сфера искусственного интеллекта *(Artificial Intelligence – AI)* весьма обширна и включает в себя множество направлений, начиная от логики и заканчивая методами оценки тональности текстов. Традиционно выделяют так называемый сильный искусственный интеллект *(Strong AI)* и слабый искусственный интеллект *(Weak AI)*.

Первый ориентирован на создание систем, выполняющих присущие человеку высокоинтеллектуальные задачи, в конечном счете на создание мыслящих машин. Подобные разработки финансируются агентством *DARPA,* как, например, упоминавшаяся выше лаборатория [17]. Можно отметить также работу [30], которая одной из первых определила само понятие «мысль» и в популярной форме описала принципы функционирования мозга.

Слабый ИИ ориентирован на создание приложений, в которых реализуется та или иная интеллектуальная способность человека и животных или в некоторых случаях физические или биологические закономерности, например, роевой (муравьиный) или распределенный интеллект [31], использованный в ряде алгоритмов поиска оптимальных значений; закон естественного отбора, реализуемый в форме так называемых генетических алгоритмов или генетического программирования; физические закономерности процессов нагревания и остывания, используемые в алгоритме симуляции отжига; строение мозга и биологические основы его работы, служащие прототипом искусственных нейронных сетей, и т.п. Часто сюда же относят область, именуемую интеллектуальными агентами [32], и мультиагентные системы, подробно описанные в работах [20] [21].

По мере своего развития ИИ как наука, находясь в авангарде научных исследований, постепенно меняет свое содержание. Если в начале своего развития к сфере ее интересов относились такие задачи, как биоидентификация, распознавание текста и т.п., то в дальнейшем они превратились, по существу, в сферу технологий, широко применяемых в прикладных науках, разработках и промышленности [33]. Возникают новые сферы исследований. Одним из успешных направлений искусственного интеллекта, нашедших очень много приложений, является машинное обучение.

Машинное обучение – обширный раздел искусственного интеллекта, изучающий методы построения алгоритмов и программ, способных обучаться [34].

К методам машинного обучения относится широкий класс алгоритмов, начиная от деревьев принятия решений, генетических алгоритмов, байесовских сетей и заканчивая искусственными нейронными сетями.

ИНС широко используются в задачах классификации и распознавания образов [32] [33]. Отдельный класс сетей (сети Хопфилда, Хемминга и Кохонена) [35] используется как средство ассоциативной памяти. Со времени своего возникновения в конце 50-х годов прошлого столетия (персептрон Розенблатта) ИНС прочно ассоциируются с искусственным интеллектом, поскольку имитируют важные особенности естественного интеллекта – способность к обучению и ассоциативность. Обширная библиография, посвященная нейронным сетям и их приложениям, отражает неослабевающий интерес исследователей к поиску новых приложений идей коннективизма и решению практических задач с помощью ИНС (например, библиография в классических изданиях [32] [33] включает более 2000 источников).

В процессе изучения аппарата нейронных сетей возникло несколько основных направлений.

Первое направление связано с возможностью применения формируемых нейронных сетей для решения классических задач вычислительной математики. Возможность такого подхода объясняется особенностями искусственного нейрона, который представляет собой элементарный процессор, а нейронная сеть из искусственных нейронов – параллельную структуру. Возможность распараллеливания вычислений и потенциальная высокая устойчивость сети к ошибкам открыли путь к созданию аппаратно формируемых нейронных сетей. Теоретические аспекты данного направления обоснованы в работах А. В. Галушкина [36].

Второе направление связано с поисками способов обучения нейронной сети и переходом от однослойной сети нейронов к многослойной. Теоретическая ограниченность решения сложных задач классификации объектов с применением однослойных сетей обоснована в книге [37]. В то же время многослойные сети не имеют таких ограничений и могут моделировать разделяющие функции практически любой степени сложности. В рамках этого направления исследуются архитектуры нейронных сетей и их способность к решению практических задач. При этом применяются многочисленные эмуляторы, например, *Neuro Office, NeuroPro, Matlab, NeuroStock, Deductor, Alyuda*

NeuroIntelligence и другие, моделирующие работу нейронной сети на стандартном компьютере. Применяются также библиотеки программ, разработанных для языков *Python* [38], C++ и т.п. Реализация алгоритмов нейронных сетей представлена также в широко известных системах машинного обучения *RapidMiner* [39], *Weka* [40].

Третье направление связано с решением вопросов построения универсального нейронного компьютера, способного обучаться и решать после обучения сложные задачи классификации.

Массовое внимание получило второе из упоминавшихся направлений, так как оно позволяет эмулировать нейронную сеть на стандартном персональном компьютере и использовать возможности обучения нейронных сетей. Важной задачей при использовании современных ИНС с прямым распространением сигналов является формирование обучающей выборки. Обучающую выборку и набор параметров для обучения ИНС формирует исследователь. При этом в некоторых случаях исследователь руководствуется мнением экспертов. Тем самым нейронная сеть становится способной «работать как эксперт» без необходимости выявления или моделирования причинно-следственной связи «факт-вывод».

Начиная с 70-х годов прошлого столетия искусственные нейронные сети стали применяться в задачах петрографии как средство анализа каротажных данных, в литологии, оценке минерально-сырьевой базы, сейсмическом зондировании [41] и т.п. [42] [43] [44] [45] [46] [47] [48] [49] [50] [51]. Применению нейронных сетей в решении практических задач интерпретации каротажных данных в области нефтедобычи посвящена работа [52]. В работах [53] [54] [55] описаны некоторые результаты применения нейронных сетей прямого распространения для интерпретации данных геофизического исследования скважин при добыче урана. Отметим, что добыча урана на месторождениях Казахстана ведется методом подземного скважинного выщелачивания, который относится к числу малозатратных, экологически безопасных способов добычи [56].

При этом экономические показатели процесса добычи зависят от скорости и точности интерпретации геофизических данных. В основном применяются электрические методы: каротаж методом кажущихся сопротивлений (КС), методом потенциалов самопроизвольной поляризации (ПС) и индукционный каротаж (ИК). Широкое применение кернового опробования при анализе пород невозможно в силу медленности процесса получения данных. Результаты каротажа представляются в виде каротажных диаграмм, на основании которых эксперт делает заключение о глубине залегания и качестве пород (рисунок 2.4 в разделе 2).

Ошибочный или неточный анализ геофизических данных приводит к потерям скважин, неоправданным трудозатратам и в конечном счете снижает экономические показатели добычи. Поскольку интерпретация данных каротажа носит во многом эмпирический характер, точные закономерности отсутствуют, становится очевидным возможность применения обучаемых систем, в частности, нейронных сетей. Однако, несмотря на достоинства ИНС, главное из которых – это способность ИНС решать слабоформализованные задачи [36], в процессе их использования имеются существенные проблемы:

- Неоднозначность мнений экспертов.
- Необходимость большого и равного количества примеров из разных классов.
- Невозможность нейронной сети объяснить полученный результат.
- Необходимость тщательной предварительной подготовки данных (очистка от аномальных значений, нормирование, сглаживание).

Перечисленные проблемы проявились на первом этапе исследования по созданию нейросетевой системы интерпретации данных каротажа на пластово-инфильтрационных месторождениях Казахстана [57].

Несмотря на указанные проблемы, задача автоматической интерпретации является крайне актуальной с учетом большого числа пробуриваемых скважин и требований по оперативной обработке данных. По приблизительным оценкам, возможный экономический результат при использовании системы автоматической интерпретации данных каротажа на месторождениях Казахстана может составить до 2 млн. долларов в год (в 2014 году). Данная оценка получена на основе приблизительной стоимости одной скважины (около 40 тыс. долларов) и примерного количества скважин, пробуриваемых ежегодно (от 1 до 2 тыс.). Предполагая возможность неправильной интерпретации данных и, соответственно, потери скважин в пределах 1 процента, можно оценить экономические потери величиной около 2 млн. долларов. При точной интерпретации потери скважин могут существенно снизиться, что и приведет к указанному выше экономическому эффекту.

1.3 Заключение по разделу 1

В данном разделе проанализированы сложившиеся научные предпосылки, связанные с развитием информационно-коммуникационных технологий, построена классификация современных доменов ИКТ, дана оценка месту, возможностям и экономическому

эффекту, полученному в результате применения интеллектуальных методов в задачах интерпретации данных каротажа скважин.

Анализ публикационной активности в области ИКТ показывает большой потенциал развития в Республике Казахстан по сравнению с ведущими странами в этой области и в сравнении со странами, имеющими сходный природный потенциал. Рассматривая современные направления исследований в виде семантической сети, очевидно, что в настоящее время сложились предпосылки как общего, так и частного характера, делающие возможным применение интеллектуальных методов во многих областях исследований и производства, в том числе в области добычи полезных ископаемых. Например, методы машинного обучения широко используются в робототехнике, в мультиагентных и кибер-физических системах. Их применение оправданно в задачах обработки больших данных, задачах визуализации, в компьютерной биологии, человекомашинных системах. При этом слабый ИИ, к которому относится МО, содержит ряд алгоритмов и методов, способных решать слабоформализованные задачи, для которых отсутствуют строгие формальные методы. Одним из широко исследуемых и применяемых подходов МО являются искусственные нейронные сети, которые, несмотря на имеющиеся недостатки, связанные в первую очередь со сложностями получения обучающего множества примеров удовлетворительного качества, способны демонстрировать хорошие результаты в задачах классификации.

Невзирая на некоторые трудности, вызванные применением методов МО, повышение точности интерпретации и исключение грубых ошибок интерпретаторов способны привести к существенному экономическому эффекту. Применение точных автоматических и своевременных методов литологического расчленения скважин урановых месторождений РК может сократить экономические потери на сумму до 2 млн. долларов в год.

Использование указанных методов требует выполнения ряда последовательных шагов, связанных с подготовкой исходных данных, обучением системы искусственного интеллекта и собственно распознаванием. Поиск оптимальных путей применения методов МО для автоматической интерпретации данных электрического каротажа рассматривается в данной работе.

2 ГЕОФИЗИЧЕСКИЕ ИССЛЕДОВАНИЯ СКВАЖИН И СРЕДСТВА АВТОМАТИЗАЦИИ ОБРАБОТКИ ДАННЫХ[3]

На современном этапе развития геологоразведочных работ геофизические исследования скважин стали одним из основных источников информации о составе и свойствах геологических объектов, условиях их залегания [58].

Они широко используются на различных стадиях поисков, разведки и отработки месторождений полезных ископаемых и в целом могут быть охарактеризованы как технологический процесс получения количественной информации о геологических объектах, реализуемый путем совместного использования технических средств измерений, а также методического, алгоритмического, петрофизического и метрологического обеспечения.

При отработке пластово-инфильтрационных месторождений урана геофизическим методам исследования скважин отводится особое место и роль. Связано это прежде всего с тем, что подавляющее большинство скважин в этом случае проходится без отбора керна. Поэтому ГИС является практически единственной информационной поддержкой всех разноплановых видов работ, выполняемых при отработке этих месторождений.

Подземное скважинное выщелачивание (ПСВ) – это сложный физико-химический процесс, протекающий в горных породах. Динамика этого процесса, его особенности определяются природными факторами: фациально-литологическими особенностями рудовмещающих горизонтов, физическими свойствами рудовмещающей толщи и всего разреза в целом. Контроль за ходом выщелачивания и управление процессом ПСВ предполагает тщательное изучение и учет всех этих природных факторов, что, в свою очередь, предъявляет повышенные требования к максимальной полноте всей информации о среде.

Объекты исследований при ГИС – геологическая среда и элементы конструкции скважины. Они характеризуются определенными физическими свойствами, вещественным составом, определенными геометрическими размерами и могут быть описаны в виде конкретной геологической и петрофизической модели.

Конечная цель ГИС – получение наиболее исчерпывающих обобщенных представлений и оценок состояния этой геологической среды в принятой при отработке месторождений методом ПСВ системе классификации.

[3] Раздел написан совместно с Я. И. Кучиным.

Большинство геологических, технических и геотехнологических задач, решение которых является конечной целью ГИС, может быть получено лишь в результате комплексной интерпретации данных различных методов ГИС, основанных на разных физических явлениях.

Поэтому вопрос комплексирования, заключающийся в выборе набора методов для решения поставленных задач, является одним из основных, определяющих требования, предъявляемые к получаемой по данным ГИС информации [59]. Естественно, при этом подразумевается, что будет строго соблюдаться принцип разумной достаточности, т.е. выбранный комплекс будет строго обоснован и будет включать в себя оптимальное сочетание отдельных методов и видов исследований.

Кроме того, корректность решения поставленных перед ГИС задач весьма часто определяется возможностью привлечения для их решения дополнительной геолого-петрографической и другой априорной информации.

Эпигенетические пластово-инфильтрационные месторождения урана, отрабатываемые способом ПСВ, приурочены к водонасыщенным проницаемым горизонтам. Рудовмещающим считают водоносный горизонт, ограниченный водоупорами и менее проницаемыми породами, представленными глинами или другими непроницаемыми породами и песками. В наиболее общем случае в качестве таких водоупоров может быть принят горизонт, являющийся менее проницаемым по отношению к рудовмещающему. В пределах этого рудовмещающего горизонта выделяют интервалы оруденения, представленного балансовыми рудами, и интервалы оруденения, локализованного в непроницаемых породах, которые относят к технологическому забалансу.

При отработке месторождений урана способом ПСВ ГИС являются основным, а зачастую и единственным методом получения наиболее полной информации об особенностях геологического разреза и характеристики уранового оруденения по каждой конкретной скважине. Они основаны на изучении естественных и искусственных физических полей во внутрискважинном, околоскважинном и межскважинном пространстве и проводятся с целью:

- изучения геологического разреза по всему стволу скважины в целом;
- детального изучения фациально-литологического строения рудовмещающих горизонтов;
- выявления рудных интервалов и параметров уранового оруденения (мощность, средние содержания, стволовые запасы);
- исследования и оценки технического состояния скважин;

- контроля за разработкой рудных залежей и оценки полноты извлечения металла из недр;
- оценки ущерба, наносимого недрам при отработке месторождений.

Полный технологический цикл работ, проводимых при добыче урана методом ПСВ, состоит из целого ряда отдельных этапов и стадий их проведения. Геофизические исследования зависят от задач, решаемых на каждом этапе работ на участке, и от условий, в которых их нужно проводить.

Полученные в ходе исследований данные обрабатываются с помощью программных продуктов, основные особенности которых описаны ниже в разделе 2.2.

2.1 Краткая характеристика методов каротажа

2.1.1 Гамма-каротаж (интегральный)

Основан на регистрации гамма-излучения естественных радиоактивных элементов (ЕРЭ), содержащихся в горных породах, пересеченных скважиной. Измеряемая величина – скорость счета гамма-квантов в импульсах в минуту (имп/мин).

Основная расчетная величина – мощность экспозиционной дозы в микрорентгенах в час (МЭД, мкР/ч).

Измеряемая величина определяется концентрацией, составом и пространственным распределением ЕРЭ, плотностью ρn и эффективным атомным номером Zэфф пород.

Гамма-каротаж (ГК) является одним из наиболее эффективных и распространенных методов ГИС. Методу отводится исключительная роль и особое место при всех без исключения видах работ, проводимых на радиоактивные руды.

Входит в число основных и обязательных методов и при работах на другие виды минерального сырья, включая работы на нефть и газ.

При проведении гамма-каротажа на урановых месторождениях используются скважинные приборы с кристаллическими детекторами *NaJ(Tl)* размерами 30х70, 18х40 мм, окруженные свинцовыми экранами 0,9–1,1 мм и 1,3–1,5 мм соответственно. Применение свинцовых экранов позволяет существенно уменьшить зависимость результатов измерений от значений Zэфф в пределах продуктивных горизонтов, т.е. устранить влияние литологического состава пород на результаты измерений. Это в значительной степени снижает картировочные возможности метода по расчленению разреза скважин, однако дает возможность с высокой

степенью точности и достоверности определять мощность, концентрацию и стволовые запасы урана в скважинах.

Минимальные требования к методическому обеспечению заключаются в наличии зависимостей:

- градуировочных, позволяющих перейти от скорости счета (в имп/мин) к мощности экспозиционной дозы, выраженной в микрорентгенах в час, или к эквивалентной массовой доле урана, выраженной в промилле урана (ppmU): 1 $ppmU$ = 1 г/т урана = $1 \cdot 10^{-4}$ % урана;
- поправочных, учитывающих влияние на МЭД бурового раствора, влажности руд, обсадной колонны и сдвиг радиоактивного равновесия между ураном, радием и радоном.

Система метрологического обеспечения метода включает:

- установку нижнего энергетического порога регистрации гамма-излучения –20 ± 5 кэВ;
- определение цены деления и нелинейности;
- определение пересчетного коэффициента Ко.

Для проведения гамма-каротажа используется комплексный скважинный прибор, позволяющий одновременно выполнять электрокаротаж в модификациях кажущихся сопротивлений и естественного электрического поля.

2.1.2 Электрокаротаж

Электрический каротаж – это метод исследования горных пород, основанный на регистрации параметров естественного или искусственного электрического поля.

Электрический каротаж, основанный на регистрации параметров естественного электрического поля, представляет собой каротаж потенциалов самопроизвольной поляризации. Измеряемой величиной является разность электрического потенциала ПС $(\Delta U_{ПС})$. Единица измерения – милливольт (мВ). Электрический каротаж, основанный на регистрации параметров искусственно создаваемого электрического поля, включает:

- боковое каротажное зондирование (БКЗ);
- боковой каротаж (БК);
- боковой микрокаротаж (БМК);
- стандартный каротаж (СК).

Все они объединяются под общим названием «каротаж сопротивлений».

Измеряемой величиной является кажущееся удельное электрическое сопротивление (ρ_K) среды. Единица измерения – ом-метр (Ом·м). При отработке месторождений урана выполняется стандартный электрокаротаж подошвенными градиент-зондами, размеры которых выбраны постоянными для данного района (месторождения) работ.

В скважинах, заполненных промывочной жидкостью на непроводящей основе, а также обсаженных полиэтиленовыми (непроводящими) трубами, электрокаротаж с целью литолого-стратиграфического расчленения разреза скважин не выполняется.

Данные стандартного электрокаротажа являются одними из основных для получения информации о литолого-стратиграфическом и фациально-литологическом строении разреза скважин. Кроме того, они используются для оценки фильтрационных свойств пород, слагающих рудовмещающий горизонт.

Минимальные требования к методическому обеспечению заключаются в наличии корреляционных зависимостей, связывающих геоэлектрические, гранулометрические параметры с фильтрационными свойствами пород.

Метрологическое обеспечение стандартного каротажа заключается в оценке постоянства кажущихся сопротивлений (ρ_K), полученных над опорным геоэлектрическим горизонтом. Сравниваются значения, полученные в идентичных условиях измерений с учетом данных о диаметре скважин и плотности бурового раствора.

2.1.3 Индукционный каротаж

Индукционный каротаж основан на измерении кажущейся удельной электрической проводимости $\delta_к$ пород в переменном электромагнитном поле в частотном диапазоне от десятков до сотен килогерц. В методе реализованы варианты измерения как активной компоненты кажущейся удельной электрической проводимости $\delta_л^a$, которая пропорциональна ЭДС, так и реактивной компоненты $\delta_л^p$, пропорциональной ЭДС, сдвинутой по фазе относительно тока генераторной цепи зонда на величину π/г. Единица измерения – сименс на метр (См/м), дробная – миллисименс на метр (мСм/м).

Типовые условия применения метода – скважины, заполненные любой промывочной жидкостью и вскрывшие породы с удельным электрическим сопротивлением менее 500 Ом·м. Является основным методом при определении мест перетоков технологических растворов из

продуктивных в вышележащие горизонты и оценке их растекания в процессе ПСВ.

Минимальные требования к методическому обеспечению заключаются в наличии зависимостей, отражающих влияние на показание зонда диаметра скважины и удельного сопротивления промывочной жидкости.

2.1.4 Кавернометрия

Кавернометрия (КМ) – метод ГИС, позволяющий определять среднее значение диаметра скважины и его изменения по стволу скважины.

Измеряемая величина – диаметр скважины в миллиметрах (мм).

2.2 Системы обработки данных ГИС, применяемые на урановых месторождениях Казахстана

В Казахстане имеется всего несколько программных продуктов, используемых для интерпретации данных ГИС на урановых месторождениях [60], все они сделаны силами самих предприятий по добыче урана. Это связано в основном с упадком отрасли после развала СССР. Зарубежные программные продукты не соответствуют действующим инструкциям, утвержденным в Государственном Комитете по Запасам Республики Казахстан.

2.2.1 Система интерпретации «Кобра»

Система «Кобра» (ТОО «Сигма», Киргизия) (рисунок 2.1) представляет собой набор программ для интерпретации данных ГИС, написанных под *ОС DOS*.

До недавнего времени использовалась на предприятиях НАК «Казатомпром». Имеет минимальные возможности для комплексной интерпретации данных ГИС, морально устарела.

2.2.2 Система GikLet

Разработана в начале 2000-х годов в ТОО «ГРК» (А. В. Белых) (рисунок 2.2). Написана на *Visual Basic* в рамках *MS Excel*. Рассчитана в основном на интерпретацию групп скважин.

Из-за ограничений *MS Excel* не очень удобна для оперативной интерпретации, особенно электрокаротажа. Используется на некоторых предприятиях НАК «Казатомпром».

2.2.3 Система комплексной интерпретации данных ГИС «Альфа»

До недавнего времени в ТОО «Геотехносервис» не было единой системы камеральной интерпретации данных ГИС. Для интерпретации данных ГИС и построения паспортов скважин использовался целый ряд программных продуктов, созданных различными производителями в течение последних 15–20 лет. Используемые продукты разработаны для разных операционных систем *(ОС DOS* и *ОС Windows)*, ориентированы на различные форматы исходных и конечных данных. Поэтому процесс интерпретации включал несколько подготовительных этапов, необходимых для построения паспорта скважины.

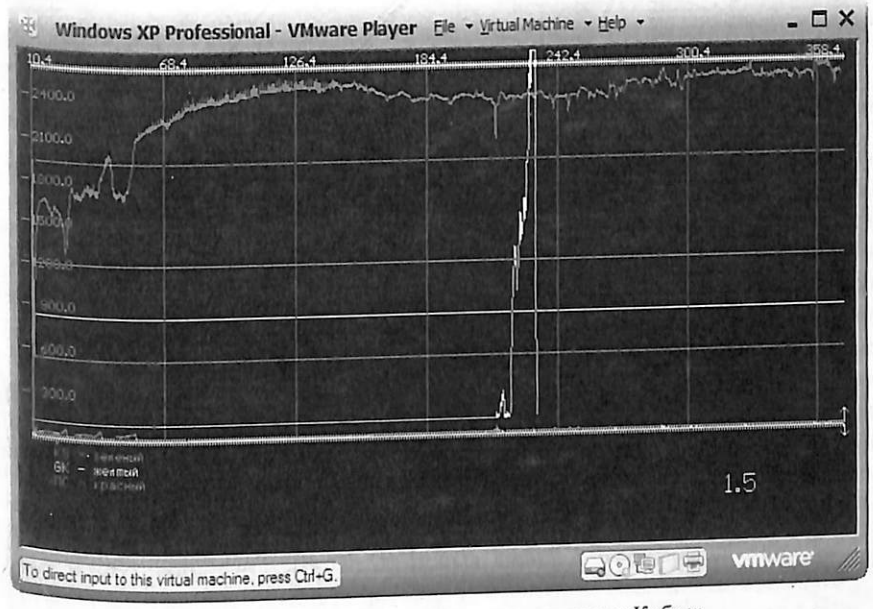


Рисунок 2.1. Окно интерпретации системы «Кобра»

Система комплексной интерпретации данных ГИС «Альфа»[4] была разработана в ТОО «Геотехносервис» в 2007 г. (авторское свидетельство № 482). Система «Альфа» включает в себя модули интерпретации электрокаротажа, гамма-каротажа, термометрии, токового каротажа, контрольных каротажей, расходометрии, редактор кривых и планшет данных ГИС (рисунок 2.3). Таким образом, принятая идеология

[4] Автором и разработчиком программы является Я. И. Кучин.

предусматривает выполнение всех работ по интерпретации данных ГИС в рамках единой системы [61].

Основой системы являются два *COM*-сервера в подгружаемых модулях *(dll)*.

Первый отвечает за загрузку и сохранение исходных данных ГИС в различных форматах *(dat*-файлов и *las*-файлов), а также за обеспечение регистрации и использования других форматов данных.

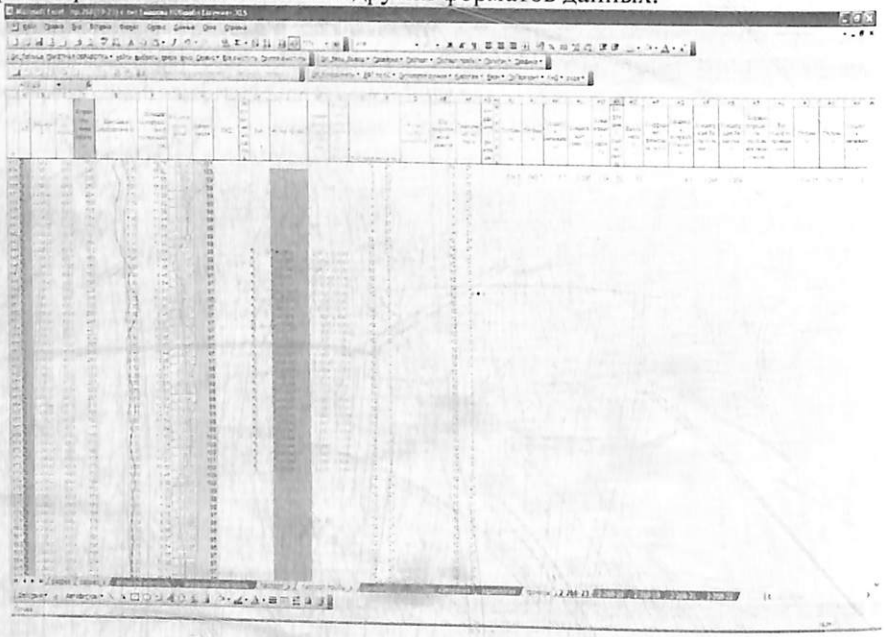


Рисунок 2.2. Окно интерпретации системы GikLet

Второй модуль осуществляет отображение данных ГИС на экране в различных масштабах, а также вывод на печать. Он используется всеми интерпретационными модулями (редактор кривых, планшет данных ГИС и др.) для отображения данных на экране.

В системе также реализован «Рабочий стол», где хранятся все исходные данные и результаты интерпретации, а также дополнительная информация (глубина начала и конца интервала, время создания интерпретационной версии). Все интерпретационные модули имеют доступ к «Рабочему столу», который осуществляет обмен данными.

Остальные модули решают только специфичные задачи. Редактор кривых предназначен для корректировки кривых (сдвиг вверх-вниз, добавление констант и др.).

Модуль интерпретации электрокаротажа предназначен для литологического расчленения пород и расчета коэффициентов фильтрации. Результаты интерпретации электрокаротажа используются при интерпретации гамма-каротажа, а также для построения паспорта скважины.

Модуль интерпретации гамма-каротажа предназначен для интерпретации данных ГК с целью определения мощностей рудных интервалов и массовых долей урана.

Модуль интерпретации токового каротажа предназначен для определения фактической глубины посадки фильтра.

Планшет данных ГИС предназначен для построения паспорта скважины. Кроме того, система предусматривает возможность дальнейшего расширения, т.е. подключение дополнительных интерпретационных модулей для других видов каротажа.

2.2.4 Интерпретации данных электрокаротажа в системе «Альфа»

Электрический каротаж – это метод исследования горных пород, основанный на регистрации параметров естественных или индуцированных электрических полей. Данные стандартного электрокаротажа являются одними из основных для получения информации о литолого-стратиграфическом и фациально-литологическом строении разреза скважин. Кроме того, результаты электрокаротажа используются для оценки фильтрационных свойств пород, слагающих рудовмещающий горизонт, позволяют производить литолого-стратиграфическое и фациально-литологическое расчленение скважины, а также оценку коэффициентов фильтрации по данным каротажа сопротивлений. Окно интерпретации (рисунок 2.4) состоит из главного окна, разделенного на 7 треков, и библиотеки литотипов. В первом треке находится шкала глубин, во втором и четвертом – каротажные кривые. Третий трек предназначен для литологической колонки соседней скважины, ориентировка на него позволяет повысить качество интерпретации, особенно для скважин с закисленных блоков. Пятый трек представляет собой литологическую колонку интерпретируемой скважины. В шестом треке находятся средние значения кажущихся сопротивлений для каждой литологической разности, а в седьмом – значения коэффициентов фильтрации. Замена литотипа осуществляется переносом соответствующего значка из библиотеки литотипов. При выборе литологической разности в литологической колонке появляется контекстное меню, содержащее следующие пункты:

- заменить литотип;
- удалить верхнюю границу;

– удалить нижнюю границу;
– добавить границу.

Изменение границ литологических разностей происходит путем перемещения их с помощью мыши. При любом изменении границ производится мгновенный пересчет коэффициентов фильтрации. Различные диапазоны значений коэффициентов фильтрации отображаются разными цветами. Результаты интерпретации электрокаротажа (литологическая колонка и коэффициенты фильтрации) сохраняются на «Рабочем столе» для последующего их использования при интерпретации гамма-каротажа и построения паспорта скважины (рисунок 2.5).

В настоящее время компьютерная интерпретация электрокаротажа проводится только по кривой КС (рисунок 2.6), остальные кривые (ПС, ИК, КМ, ГК) используются для ручной корректировки человеком-интерпретатором. Кроме того, при закислении скважины автоматическая интерпретация по одной лишь КС зачастую становится невозможной.

Исчерпывающей теории, которая могла бы служить основой для автоматического литологического расчленения с учетом всех видов каротажа, сейчас нет.

Очевидно, что комплексная интерпретация с учетом всех данных ГИС требует больших затрат времени и профессионализма интерпретатора, в то время как технология производства -- быстрого принятия решений.

2.3 Заключение по разделу 2

Для выполнения литологического расчленения скважин по добыче урана методом подземного выщелачивания широко используются три метода электрического каротажа: КС, ПС и ИК.

Полученные данные в настоящее время обрабатываются с помощью программных продуктов «Кобра», *GikLet,* «Альфа». Система «Кобра» не содержит средств автоматизации процесса интерпретации. Система «Альфа», разработанная в 2007 году, включает в себя модули интерпретации электрокаротажа, гамма-каротажа, термометрии, токового каротажа, контрольных каротажей, расходометрии, редактор кривых и планшет данных ГИС. При этом компьютерная (автоматическая) интерпретация электрокаротажа проводится только по кривой КС, остальные данные (ПС, ИК, КМ, ГК) используются для ручной корректировки человеком-интерпретатором. Кроме того, при закислении скважины автоматическая интерпретация по одной лишь КС зачастую становится невозможной. Можно отметить также, что в настоящее время отсутствует законченная формальная модель, которая могла бы служить

основой для автоматического литологического расчленения с учетом всех видов каротажа. Очевидно, что поскольку комплексная интерпретация с учетом всех данных ГИС происходит вручную, то необходимы большие затраты времени и профессионализм интерпретатора, в то время как технология производства зачастую требует быстрого принятия решений. Это означает, что необходима система, позволяющая выполнять интерпретацию данных каротажа в автоматическом режиме, возможно, с предварительным обучением на данных, проинтерпретированных экспертами.

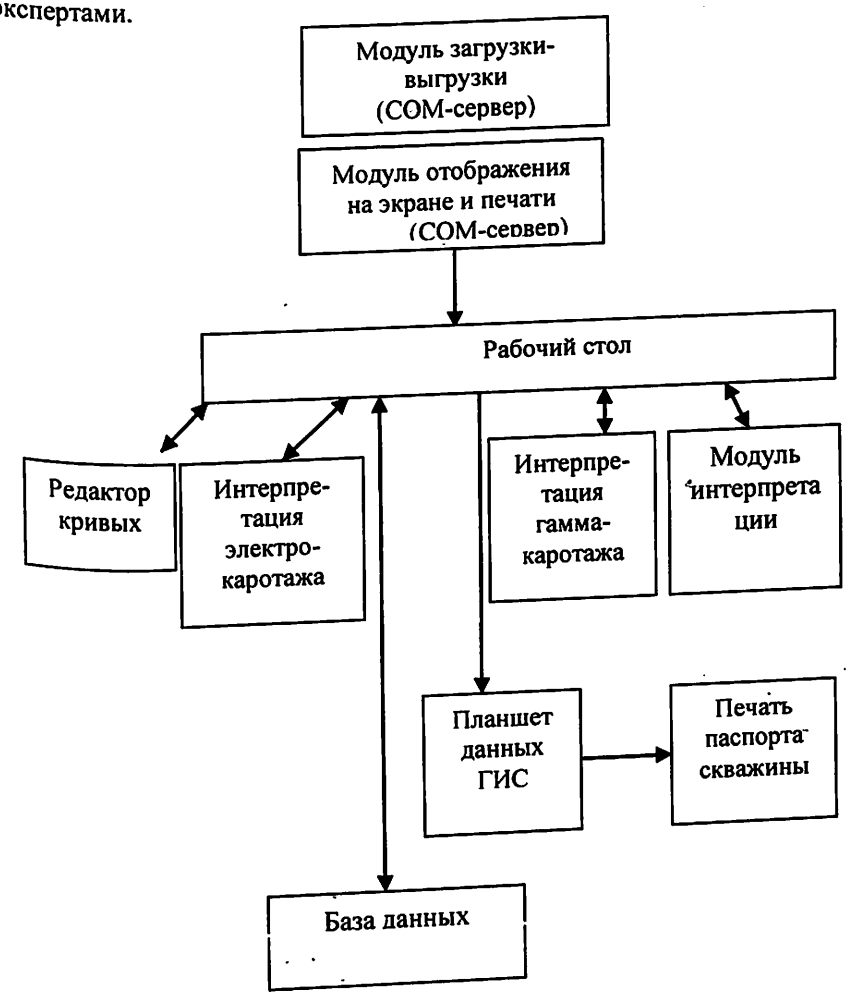


Рисунок 2.3. Архитектура системы «Альфа»

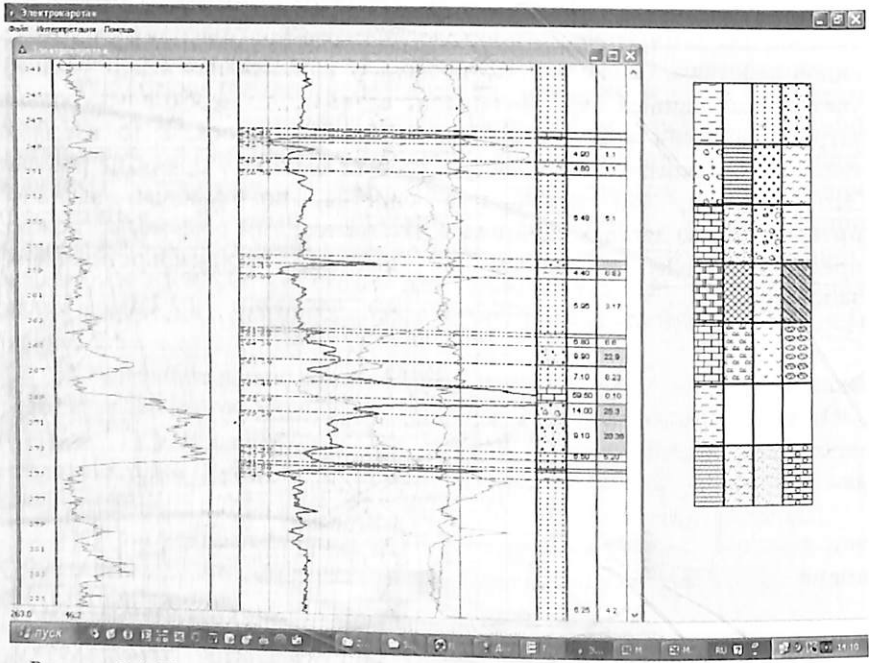


Рисунок 2.4. Главное окно программы интерпретации электрокаротажа системы «Альфа»

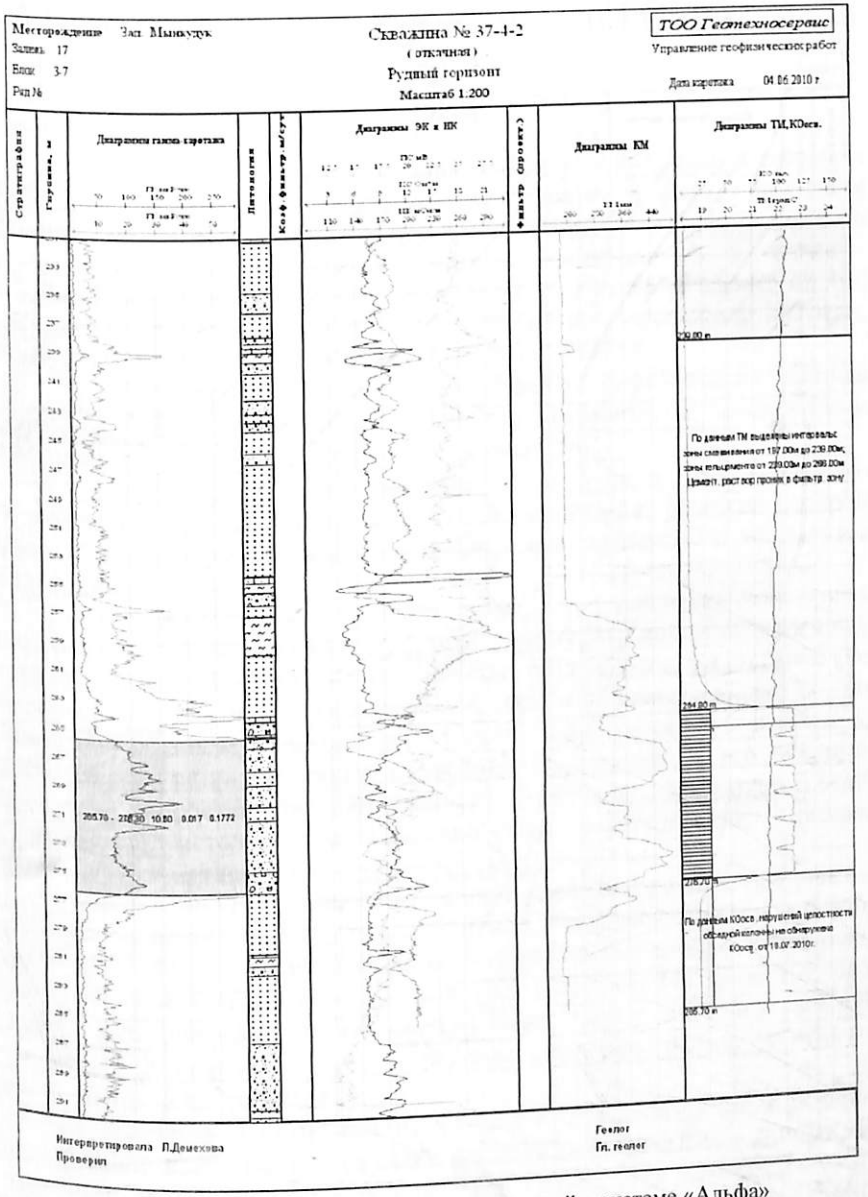


Рисунок 2.5. Паспорт скважины, построенный в системе «Альфа»

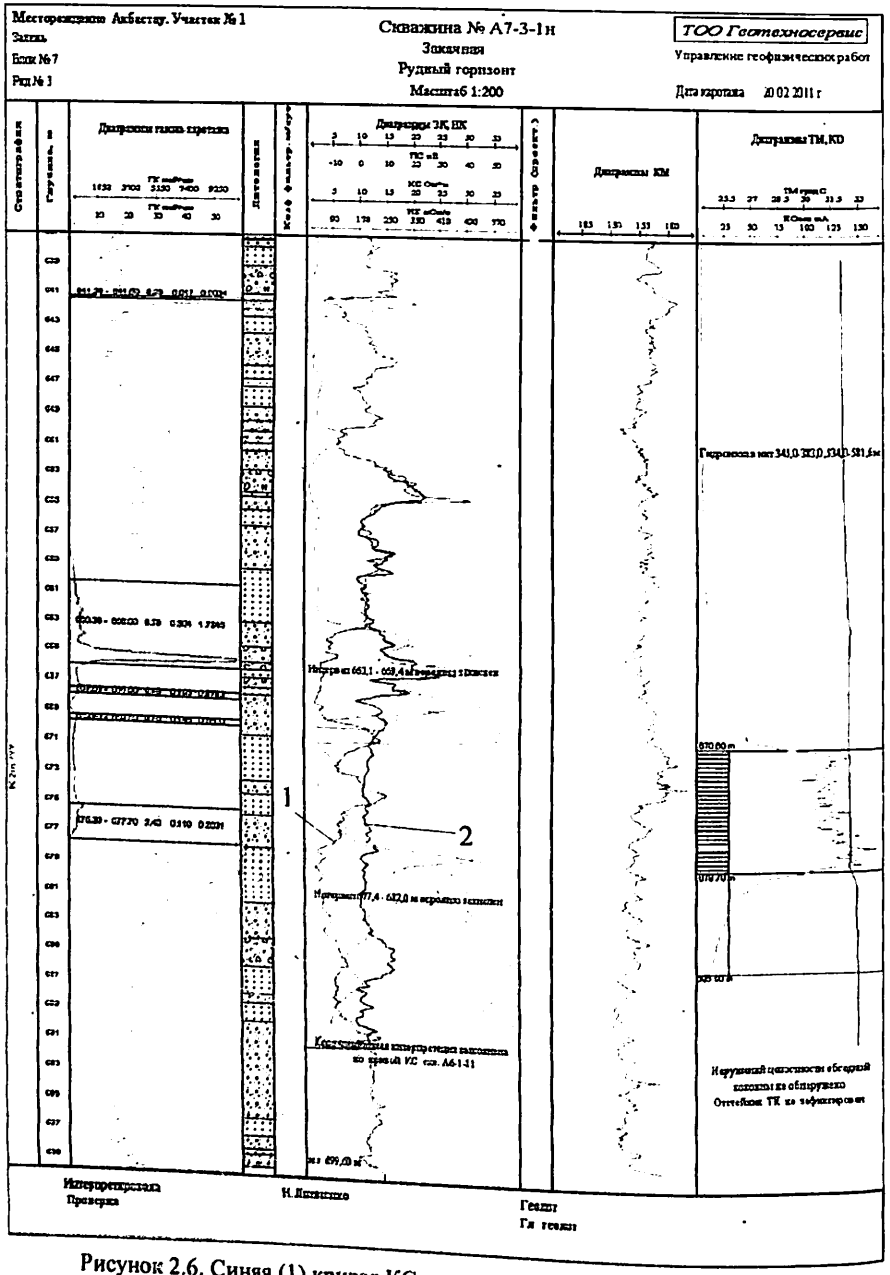


Рисунок 2.6. Синяя (1) кривая КС с закисленной скважины, черная (2) с незакисленной соседней скважины. Видны два интервала закисления

3 АВТОМАТИЗАЦИЯ ПРОЦЕССА ИНТЕРПРЕТАЦИИ ГЕОФИЗИЧЕСКИХ ДАННЫХ

3.1 Методы машинного обучения

Ошибочный или неточный анализ геофизических данных приводит к потерям скважин, неоправданным трудозатратам и в конечном счете снижает экономические показатели добычи. Как отмечалось выше, по оценкам экспертов, неправильная интерпретация данных, а точнее – ошибка в выделении непроницаемых пластов, только в одном из 100 случаев может вызвать экономические потери в объеме около 200 млн. тенге в год (более 1 млн. долларов по курсу 2014 года).

Автоматизация процесса интерпретации геофизических данных позволит уменьшить количество ошибок, связанных с человеческим фактором, и улучшить экономические показатели.

По своей сути задача интерпретации данных геофизического исследования скважин является слабоформализуемой, решение которой можно искать в рамках научного направления, именуемого машинным обучением.

Интерпретация данных часто связана с классификацией, когда определенный объект нужно отнести к одному из ранее определенных классов, кластеризацией, когда объекты разделяются на заранее не определенные группы (кластеры), и прогнозированием, когда по некоторому объему исходных данных, описывающих, например, предысторию развития процесса, необходимо определить его будущее состояние в пространстве или времени. В случаях, когда нет строгих формальных методов классификации или кластеризации, широко используются методы МО.

Методы МО [62] включают широкий класс алгоритмов, начиная от деревьев решений, генетических алгоритмов, метрических методов, таких как *k-NN, SVM,* статистических методов, байесовских сетей и заканчивая искусственными нейронными сетями [32] [33] [63]. По существу, это направление призвано решать центральную задачу интеллектуальной системы, предваряющую все остальные действия, оценку текущего состояния объекта (ситуации).

С конца прошлого столетия методы МО используются в задачах петрографии и литологии как средство анализа каротажных данных. В частности, этому посвящен ряд работ [53] [54] [55]. Однако области приложений МО гораздо шире. Они включают медицину [64] [65] [66], биологию [67], робототехнику, городское хозяйство и промышленность [68], сферу обслуживания, экологию [69], системы связи нового типа [70], астрономию [71] и т.д.

Ниже мы рассмотрим таксономию методов МО, ключевые алгоритмы и особенности их применения.

3.1.1 Типы алгоритмов машинного обучения

Машинное обучение как дисциплина, являющаяся частью обширного направления, именуемого «искусственный интеллект», по существу, реализует потенциал, заложенный в идее ИИ. Основное ожидание, связанное с МО, заключается в реализации потребности в гибких, адаптивных, обучаемых алгоритмах или методах вычислений[5]. В результате обеспечиваются новые функции систем и программ.

Возможности МО, то есть способность обучаться и обеспечивать рекомендации на уровне экспертов в узкой предметной области, реализуются алгоритмами, которые делятся на две большие группы:

- Обучение без учителя *(Unsupervised Learning) (UL)* [72].
- Обучение с учителем *(Supervised Learning) (SL)* [73].

Кроме этого, иногда выделяют:

- Обучение с подкреплением *(Reinforcement Learning) (RL)* [74].
- Полууправляемое обучение *(Semi-Supervised Learning) (SSL)* [75].

Главная задача, решаемая алгоритмами МО, заключается в отнесении наблюдаемого объекта к тому или иному классу для принятия последующего решения автоматически или человеком. Такие задачи распространены очень широко. В качестве примера можно указать на задачи, возникающие в процессе движения мобильного автономного робота и связанные с распознаванием образов предстоящего пути; задачи распознавания лиц, мимики, эмоций; анализ действия пользователя при получении услуг в системах электронной коммерции, который дает возможность проводить как оптимизацию интерфейса, так и планировать действия системы. В целом это анализ данных в различных информационных системах, позволяющий выполнять предсказания состояний или классификацию объектов. Различаются способы решения указанной задачи.

Методы *UL* решают задачу кластеризации, когда множество заранее не обозначенных объектов разбивается на группы путем автоматической процедуры, исходя из свойств этих объектов. При этом количество групп (кластеров) может быть заранее задано или формироваться автоматически. К числу таких алгоритмов относятся

[5] «Метод вычислений» – термин, введенный Д. Кнутом для отделения строго обоснованных алгоритмов от эмпирических методов, обоснованность которых часто подтверждается практикой.

теория адаптивного резонанса *(Adaptive Resonance Theory – ART)* и самоорганизующиеся карты (S*self-Organizing Map – SOM)* или карты Кохонена [76], а также обширная группа алгоритмов кластеризации (*k-means, mixture models, hierarchical clustering* и др.) [77] [78].

SL решает задачу классификации, когда в потенциально бесконечном множестве объектов выделяются конечные группы некоторым образом обозначенных объектов. Обычно формирование групп выполняется экспертом. При этом эксперт может объяснять, а может и не объяснять, по каким причинам он выполнил первоначальную классификацию.

Алгоритм классификации должен, используя эту первоначальную классификацию как образец, отнести следующие необозначенные объекты к той или иной организованной экспертом группе, исходя из свойств этих объектов. *SL* включает большой набор алгоритмов или семейств алгоритмов, которые часто разделяются на линейные и нелинейные классификаторы – в зависимости от формы (гиперплоскости или гиперповерхности), разделяющей классы объектов. В двумерном случае линейные классификаторы разделяют классы единственной прямой, тогда как нелинейные классификаторы – линией (рисунок 3.1).

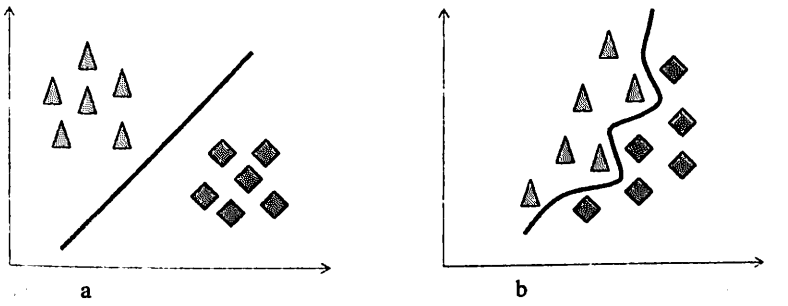


Рисунок 3.1. Линейный (a) и нелинейный (b) классификаторы

Подходы к классификации алгоритмов МО представлены, в частности, в работах [79] [80]. Таксономия алгоритмов МО, не претендующая на исчерпывающую полноту, может быть представлена в виде следующей иерархической структуры [81] [82].

- Обучение без учителя (UL)
  - ART [82]
    - ART1
    - ART2
    - ART3
    - ARTMAP

- Fuzzy-ART [83]
- Fuzzy-ARTMAP
- SOM [84]
  - Generative Topographic Map (GTM) [85]
- Cluster algorithms [86]
  - k-means
    - K-Means++
    - K-Medoids
    - Fuzzy C-Means Clustering Algorithm (FCM)
    - Soft K-Means Clustering Algorithm (SKM)
    - K-Harmonic Means Clustering Algorithm (KHM)
    - Kernel K-Means Clustering Algorithm (KKM)
  - Spectral Clustering Algorithm (SCA)
  - Density models (DM)
  - Subspace models:
    - mixture models (MM)
    - hierarchical clustering (HC)

– Обучение с учителем (supervised learning) (SL)
- Linear classifiers
  - Linear Discriminant Analysis Classifier (LDA)
  - Logical regression (LR)
  - Naive bayes Classifier (NBC)
  - Perceptron (P)
- Non-linear classifiers
  - Quadratic Classifier (QC)
  - Diagonal Linear Discriminant Analysis (DLDA)
  - Support Vector Classification (SVM) (Linear SVM и Non-linear SVM)
  - Logistic regression (LogR)
  - k-Nearest-Neighbor (k-NN)
  - Decision Tree (DT)
    - Random Forest (RF)
  - Neural Networks (NN)
    - Bayesian Networks (BN)

– Обучение с подкреплением (reinforcement learning) (RL)
- Q-Learning
  - Deterministic Q-Learning (DQL)
- Monte-Carlo Methods (MCM)
- Temporal Difference Methods (TDM)
- Sarsa

– Полууправляемое обучение (semi-supervised learning) (SSL)

На рисунке 3.2 представлена таксономия алгоритмов МО в виде графа. Каждый из перечисленных алгоритмов, по существу, образует некоторое семейство модифицируемых под те или иные потребности программ и алгоритмов, часто различающихся вычислительной сложностью, сложностью реализации и автоматизации процесса обучения, способностью классифицировать только два типа *(binary classification)* или сразу несколько типов объектов.

В настоящей работе, исходя из ее постановки, известно выполненное экспертами литологическое расчленение пород по многим скважинам, которые можно использовать для обучения системы МО. Следовательно, основной задачей проектируемой системы машинного обучения является классификация на основе имеющихся примеров *SL*.

3.1.2 Схема настройки системы машинного обучения

Применение методов машинного обучения в задачах, для которых строгая математическая модель отсутствует, а имеются только экспертные оценки, часто бывает оптимальным способом решения. Обучаемая система, в частности искусственная нейронная сеть, способна воспроизвести закономерность, которую сложно или невозможно формализовать. В задачах «обучения с учителем» часто затруднительно определить качество экспертных оценок. К таким задачам, в частности, относятся и задачи выявления рисков заболеваний, оценки качества продуктов, распознавания речи, предсказания уровня котировок акций на финансовых рынках, распознавания литологических типов на урановых месторождениях по данным электрического каротажа. Несмотря на то, что эксперты задают перечень актуальных признаков объектов, диапазоны измеряемых физических величин могут перекрываться, а экспертные оценки могут быть противоречивыми или содержать ошибки. Например, на рисунке 3.3 показаны точки, соответствующие породам (по экспертным оценкам), в пространстве трех видов каротажа для месторождения Буденовское. (Коды литологических типов расшифровываются в приложении 1 «Коды литотипов».) Видно, что точки, соответствующие разным литологическим типам, существенно перемешаны в пространстве признаков и, соответственно, не могут быть разделены простыми, например линейными, способами.

Кроме этого, данные, представленные для классификации, могут содержать аномальные значения и ошибки, связанные с физическими особенностями процессов их получения. Соответственно, и обученная система может интерпретировать данные с ошибками.

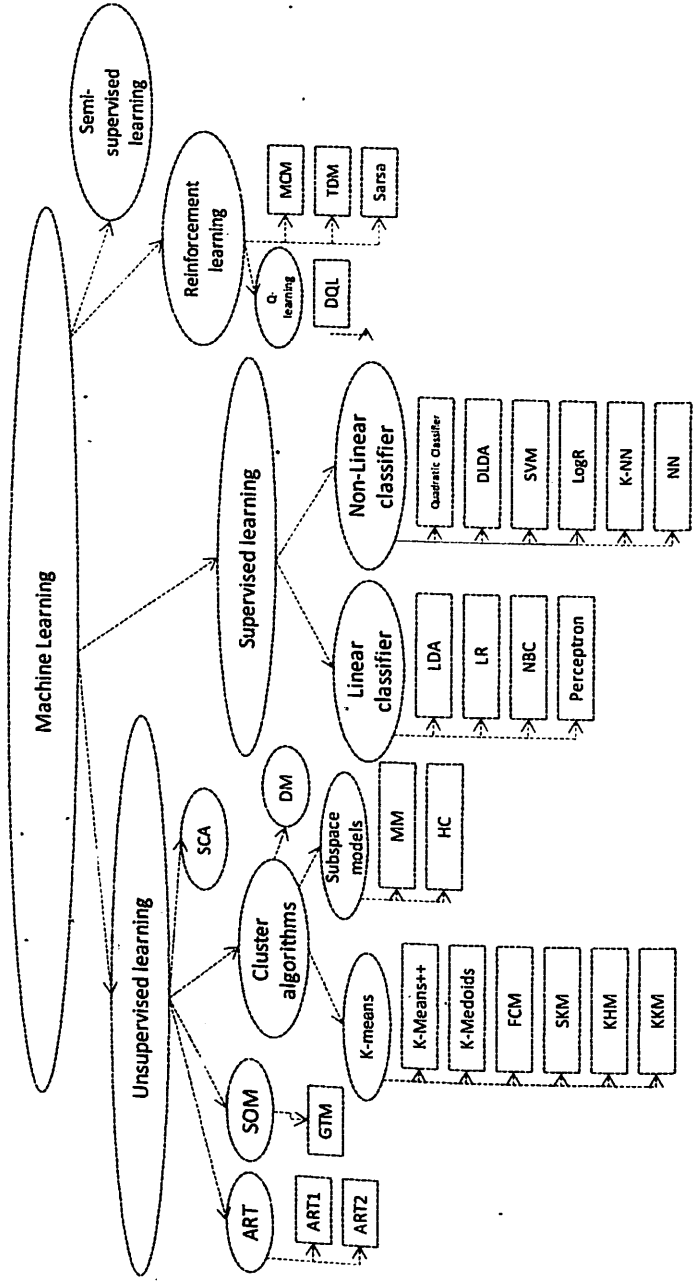


Рисунок 3.2. Таксономия алгоритмов машинного обучения

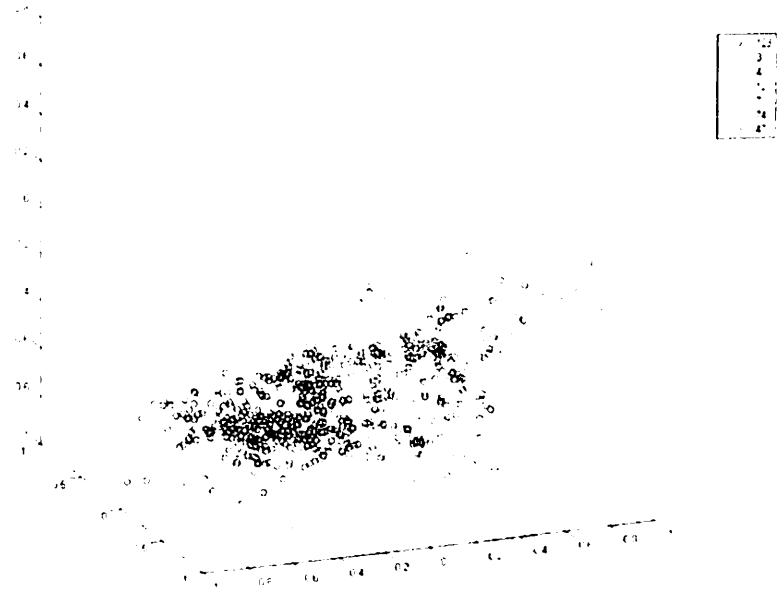


Рисунок 3.3. Ответы экспертов в трехмерном (ИК, КС и ПС) пространстве признаков

Анализ применимости методов машинного обучения, методы подготовки данных для использования указанных методов, а также сравнение алгоритмов между собой являются необходимым условием разработки научно обоснованного комплекса программ автоматической интерпретации каротажных данных.

Общая схема настройки алгоритмов машинного обучения на решаемую задачу приведена на рисунке 3.4.

В соответствии с этой схемой нам необходимо определить саму задачу машинного обучения в формальном виде, выбрать алгоритмы, которые будут использоваться в системе, определить показатели качества алгоритмов машинного обучения, выбрать и апробировать методы предобработки данных.

Технология настройки является итеративной процедурой, при которой для правильной работы системы приходится многократно возвращаться к предыдущим блокам настройки параметров. Описанная схема апробирована в процессе проведения около 2000 вычислительных экспериментов. В связи с тем, что в данной работе решается задача классификации литологических типов, рассмотрим подробнее формальную постановку задачи классификации.

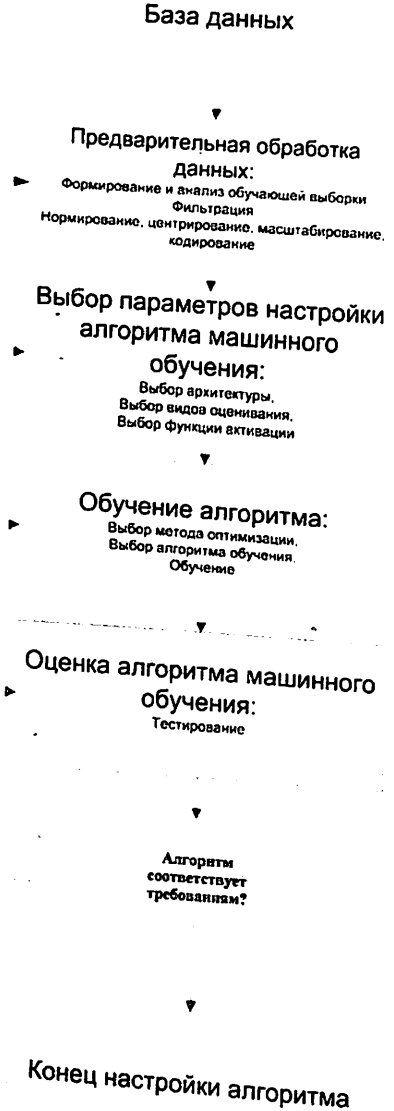


Рисунок 3.4. Обобщенный алгоритм настройки системы машинного обучения на решаемую задачу

3.1.3 Задача классификации

Формальная постановка задачи машинного обучения (задача обучения по примерам или задача обучения с учителем) известна и заключается в следующем [87]. Пусть имеются два пространства: X (пространство допустимых объектов), Y (пространство ответов или меток) и (целевая) функция

y: X → Y, которая задана лишь в конечном множестве точек (обучающей выборке, прецедентах *(sample set)):*

$y(x^1),\ldots,y(x^m)$,

т.е. известны метки объектов $x^1,\ldots,x^m$. Требуется построить алгоритм A (или «обучить» алгоритм), который по объекту x определяет значение y(x) (или «достаточно близкое» значение, если допускается неточное решение).

При конечном множестве $Y = \{1,2,\ldots,l\}$ задачу называют задачей классификации (на l непересекающихся классов). В этом случае можно считать, что множество X разбито на классы $K1,\ldots,Kl$, где $Ki = \{x \in X \mid y(x) = i\}$ при $i \in \{1,2,\ldots,l\}$:

$$X = \bigcup_{i=1}^{l} K_i$$

При $Y = \{(\alpha_1,\ldots,\alpha_l) \mid \alpha_1,\ldots,\alpha_l \in \{0,1\}\}$ говорят о задаче классификации на l пересекающихся классов. Здесь i-й класс – $Ki = \{x \in X \mid y(x) = (\alpha_1,\ldots,\alpha l), \alpha_i = 1\}$.

Часто в задаче может быть введена функция потерь или стоимости $J(A(x), y(x))$, которая описывает, насколько «плох» наш ответ $A(x)$ при верном ответе $y(x)$. В задаче классификации можно считать, что

$$J(A(x), y(x)) = \begin{cases} 1, A(x) \neq y(x) \\ 0, A(x) = y(x) \end{cases}$$

а в задаче регрессии –

$$J(A(x), y(x)) = |A(x) - y(x)|$$

или

$$J(A(x), y(x)) = (A(x) - y(x))^2.$$

Задачу обучения по примерам можно рассматривать как задачу оптимизации, которую можно решать путем поиска минимального значения функции стоимости $J(\theta)$ по всем доступным примерам, определяемую как сумма квадратов разности «предсказываемого» значения и реального значения y по множеству примеров m. При этом

«подбирается» гипотеза $h_\theta(x)$, которая при некотором наборе параметров $\theta_i \in \Theta$ обеспечивает минимальное значение $J(\theta)$.

$$J(\theta) = \min \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2, \qquad (3.1)$$

где m – множество примеров, h_θ – функция гипотезы, которая может быть линейной ($h_\theta = \theta_0 + \theta_1 x$) или нелинейной (например, $h_\theta = \theta_0 + \theta_1 x + \theta_2 x^2$) с различным набором параметров $\theta_i \in \Theta$.

Забегая вперед, можно сказать, что для подбора параметров θ_i необходимо, чтобы параметры $x_j \in X$ (для многомерного случая) были выражены единицами одинаковой размерности и примерно одинаковой величины. Чаще всего путем нормализации стремятся представить все параметры в виде чисел в диапазоне $0 \le x \le 1$ или $-1 \le x \le 1$. Вообще говоря, выбор функции нормализации зависит от класса задачи. Кроме того, в процессе предварительной обработки данных могут быть использованы методы, обеспечивающие исключение аномальных значений, исключение шумов, например высокочастотных, путем сглаживания и т.п. Выбор этих методов также зависит от класса задачи.

После того как параметры нормализованы и данные приведены к нужному виду, выполняется поиск функции гипотезы $h_\theta(x)$, которая минимизирует стоимость $J(\theta)$. Для решения этой задачи используется большое число алгоритмов, часть которых описана ниже.

3.1.4 Регрессионные алгоритмы и алгоритмы классификации данных

3.1.4.1 Линейная регрессия

Задача линейной регрессии формулируется как поиск минимальной функции стоимости (см. формулу 3.1) при условии, что функция гипотезы является линейной $h_\theta = \theta_0 + \theta_1 x$. Очевидно, что подобная функция реализует линейный классификатор (рисунок 3.1а). Для нахождения оптимальной функции $h_\theta(x)$ применяется алгоритм градиентного спуска *(gradient descent)*, суть которого заключается в

последовательном изменении параметров θ_0, θ_1, используя следующее выражение:

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1), \qquad (3.2)$$

где α – параметр обучения, а $\frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$ является производной функции стоимости по θ_j. Знак := означает присваивание в отличие от знака равенства (=) в алгебраических выражениях.

При этом шаги алгоритма выполняются так, что вначале происходит одновременное изменение обоих параметров на основании выражения 3.2 и только затем присваивание им новых значений. Другими словами, алгоритмическая последовательность одного шага алгоритма для случая двух параметров, выраженная на псевдокоде, будет следующей:

$$temp0 := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1);$$

$$temp1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1);$$

$$\theta_0 := temp0;$$

$$\theta_1 := temp1.$$

В зависимости от параметра обучения α алгоритм может достигать минимума (сходиться) или при слишком большом α не сходиться.

Наиболее простой и в реализации, но не самый оптимальный по времени выполнения пакетный алгоритм градиентного спуска *(Batch Gradient Descent)* использует все обучающие примеры на каждом шаге алгоритма. Вместо алгоритма градиентного спуска для нахождения параметров θ можно использовать матричное выражение

$$\Theta = (X^T X)^{-1} X^T y, \qquad (3.3)$$

где Θ – вектор параметров, $(X^T X)^{-1}$ – обратная матрица $X^T X$, X^T – транспонированная матрица X.

Преимуществом матричных операций является то, что нет необходимости подбирать параметр α и выполнять несколько итераций алгоритма. Недостаток связан с необходимостью получения обратной матрицы, сложность вычисления которой пропорциональна $O(n^3)$, а также невозможностью получения обратной матрицы в некоторых случаях.

3.1.4.2 Полиномиальная регрессия

В отличие от линейной регрессии, полиномиальная оперирует нелинейной функцией гипотезы вида $h_\theta = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + ... + \theta_n x^n$, что позволяет строить экстраполирующие кривые (гиперповерхности) сложной формы. Однако с увеличением числа параметров существенно возрастает вычислительная сложность алгоритма. Кроме этого, существует опасность «переобучения», когда кривая становится слишком сложной формы и хорошо соответствует обучающему множеству, но дает большую ошибку на тестовом. В случае переобучения, когда классификатор теряет способность к обобщению, применяют регуляризацию, снижающую влияние величин высокого порядка:

$$J(\theta) = \min \frac{1}{2m}[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^{n} \theta_j^2]$$

Увеличение параметра λ приводит к усилению способности обобщения алгоритма. В пределе при очень большом значении параметра функция гипотезы превращается в прямую.

3.1.4.3 Логистическая регрессия

Применяется в случае, если необходимо разделить объекты двух классов, например, на «негативные» и «позитивные». То есть набор обучающих примеров построен таким образом, что $y \in \{0,1\}$.

В этом случае от функции гипотезы требуется выполнение условия $0 \le h_\theta(x) \le 1$, что достигается применением сигмоидальной (логистической) функции $h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$,

где Θ – вектор параметров.

Можно записать также

$$h_\theta(x) = g(\Theta^T x),$$

где $g(z)$ – сигмоидальная функция.

Отметим, что сигмоидальная функция широко применяется и в нейронных сетях в качестве активационной функции нейронов, поскольку является непрерывно дифференцируемой и тем самым гарантирует сходимость алгоритмов обучения нейронной сети. Примерный вид сигмоиды показан на рисунке 3.5.

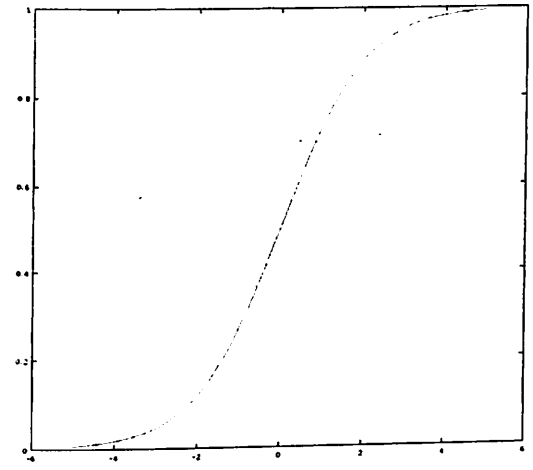


Рисунок 3.5. Сигмоидальная функция

$h_\theta(x)$ может рассматриваться как вероятность того, что объект является «позитивным» $h_\theta(x) \ge 0.5$ или «негативным» $h_\theta(x) < 0.5$. В сложных случаях, требующих нелинейной границы разделения, например, в виде окружности (рисунок 3.6), – $h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$.

Подбор параметров Θ после выбора функции гипотезы выполняется так, чтобы минимизировать функцию стоимости вида

$$J(\theta) = -\frac{1}{m}[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)}))]$$

Как и в случае линейной регрессии, минимизация достигается с помощью алгоритма градиентного спуска *(gradient descent)*, но также применяются *Conjugate gradient* [88], *BFGS, L-BFGS* [89].

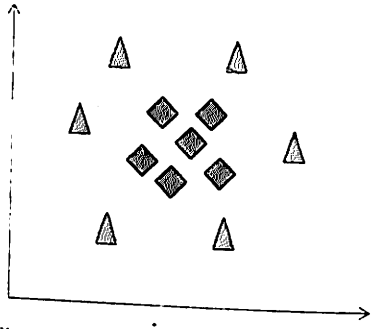


Рисунок 3.6. Нелинейная граница разделения объектов разных классов

Логистический классификатор может быть применен и в отношении нескольких классов. В этом случае для каждого класса классификатор настраивается отдельно. Класс, к которому принадлежит новый объект, вычисляется расчетом значений всех функций гипотез и выбором из них максимального значения $\max_i h_\theta^{(i)}(x)$, где i – номер класса.

Другими словами, объект принадлежит к тому классу, функция гипотезы которого максимальна.

Как и в случае с линейной регрессией, для увеличения обобщающей способности алгоритма применяют регуляризацию (последнее слагаемое в нижеследующей формуле), которая позволяет уменьшить влияние величин высокого порядка:

$$J(\theta) = [-\frac{1}{m}\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)}))] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

3.1.4.4 Искусственные нейронные сети

ИНС, или *Artificial Neural Networks (ANN)*, – аппарат, который активно исследуется начиная с 40-х годов прошлого столетия. ИНС как часть теории коннективизма прошла значительный путь от эпохи

завышенных ожиданий, затем, в 70-х годах, разочарований и до широко применяемой технологии в настоящее время. Связь между биологическими нейронами и возможностями их моделирования с помощью логических вычислений установлена в работе *Warren S. McCulloch, Walter Pitts* [90], в работе Розенблатта [91] описана модель персептрона, недостатки однослойного персептрона отражены в работах М. Минского и С. Пейперта [37] [92]. В 1974 году *Paul Werbos* предложил алгоритм обратного распространения *(back propagation)* [93] [94], пригодный для обучения многослойного персептрона или нейронной сети.

Наиболее популярная и полезная в нашем случае архитектура ИНС – сеть прямого распространения, в которой нелинейные элементы (нейроны) представлены последовательными слоями, а информация распространяется в одном направлении *(Feedforward Neural Networks)* [95]. В 1989 году в работах *G. Cybenco* [96] и *K. Hornik* и др. [97] показано, что такая сеть способна аппроксимировать функции практически любого вида.

Значительный вклад в теорию коннективизма внесли отечественные ученые [36] [98] [99] [100], доказавшие возможность решения классических вычислительных задач в нейросетевом базисе, тем самым заложив фундаментальную основу построения нейрокомпьютеров.

Применение аппарата ИНС направлено на решение широкого круга вычислительно сложных задач, таких как оптимизация, управление, обработка сигналов, распознавание образов, предсказание, классификация.

Рассмотрим использование ИНС с прямым распространением сигнала. В такой сети отдельный нейрон представляет собой логистический элемент, состоящий из входных элементов, сумматора, активационного элемента и единственного выхода (рисунок 3.7).

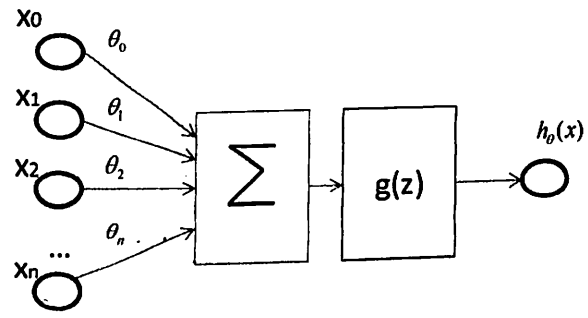


Рисунок 3.7. Схема классического нейрона

Выход нейрона определяется формулами:

$$z = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 \ + \ldots + \theta_n x_n,$$

$$h_\theta(x) = g(z)$$

где $g(z)$ – сигмоидальная функция.

Для упрощения схемы сумматор и активационный элемент объединяют, тогда многослойная сеть может выглядеть так, как показано на рисунке 3.8. Сеть содержит четыре входных нейрона, четыре нейрона в скрытом слое и один выходной нейрон. На рисунке входные нейроны обозначены символом x, нейроны скрытого слоя – символами $a_1^{(2)}$, $a_2^{(2)}$, $a_3^{(2)}$, $a_0^{(2)}$ и выходного слоя – символом $a_1^{(3)}$. Если нейронная сеть имеет несколько слоев, то первый слой называют входным, а последний – выходным. Все слои между ними называются скрытыми. Для нейронной сети с L-слоями выход первого слоя нейронов (входного слоя) определяется выражением: $a^{(1)} = x$

На входе следующего слоя имеем:

$z^{(2)} = \Theta^{(1)} a^{(1)}$

Выход второго слоя:

$a^{(2)} = g(z^{(2)}) + a_0^{(2)}$

Для последующих слоев (или для всех «скрытых» слоев):

$$a^{(i)} = g(z^{(i)}) + a_0^i = g(\ \Theta^{(i-1)} a^{(i-1)}) + a_0^i \qquad (3.4)$$

Для выходного слоя:

$$a^{(L)} = h_\theta(x) = g(z^{(L)}) \qquad (3.5)$$

Например, для сети на рисунке 3.8 выход каждого нейрона скрытого слоя можно рассчитать так же, как и для одиночного нейрона:

$$a_1^{(2)} = g(\theta_{10}^{(1)} x_0 + \theta_{11}^{(1)} x_1 + \theta_{12}^{(1)} x_2 + \theta_{13}^{(1)} x_3) \qquad (3.6)$$

$$a_2^{(2)} = g(\theta_{20}^{(1)} x_0 + \theta_{21}^{(1)} x_1 + \theta_{22}^{(1)} x_2 + \theta_{23}^{(1)} x_3)$$

$$a_3^{(2)} = g(\theta_{30}^{(1)} x_0 + \theta_{31}^{(1)} x_1 + \theta_{32}^{(1)} x_2 + \theta_{33}^{(1)} x_3)$$

Выход нейронной сети определяется выражением:

$$a^{(3)} = h_\theta(x) = g(\theta_{10}^{(2)} a_0^{(2)} + \theta_{11}^{(2)} a_1^{(2)} + \theta_{12}^{(2)} a_2^{(2)} + \theta_{13}^{(2)} a_3^{(2)}) \qquad (3.7)$$

Достоинством нейронной сети является возможность классификации сразу нескольких классов. В этом случае выходной слой содержит количество нейронов, равное числу классов. Например, при необходимости классифицировать объекты двух классов методом голосования («победитель забирает все») получим сеть (рисунок 3.9).

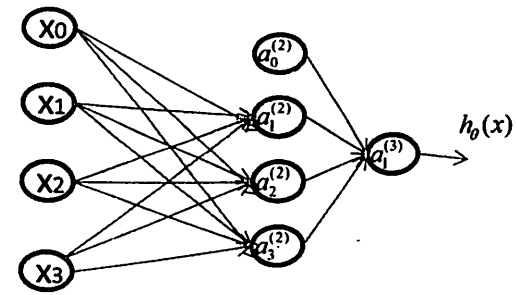


Рисунок 3.8. Схема многослойной сети

Для настройки весов θ нейронной сети (обучения сети) используют функцию стоимости, напоминающую функцию стоимости для логистической регрессии (3.8).

$$J(\Theta) = -\frac{1}{m}[\sum_{i=1}^{m}\sum_{k=1}^{K} y_k^{(i)} \log(h_\Theta(x^{(i)}))_k + (1 - y_k^{(i)}) \log(1 - h_\Theta(x^{(i)}))_k] + \frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{i=1}^{S_l}\sum_{j=1}^{S_{l+1}}(\Theta_{ji}^l)^2 \qquad (3.8)$$

где L – количество слоев нейронной сети;

S_l – количество нейронов в слое l;

K – количество классов (равно количеству нейронов в выходном слое);

Θ – матрица весов.

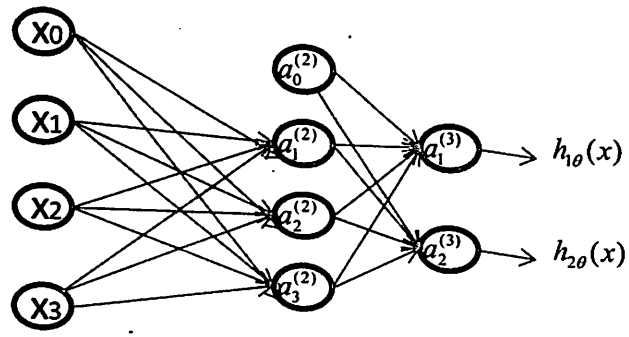


Рисунок 3.9. Схема многослойной сети с двумя выходами

Для обучения многослойной ИНС используют алгоритм обратного распространения ошибки *(Back Propagation Error – BPE)* и различные модификации, направленные на ускорение процесса обучения.

Суть алгоритма *BPE* заключается в следующем.

Для тренировочного набора примеров

$\{(x^{(1)}, y^{(1)}),...,(x^{(m)}, y^{(m)})\}$

Устанавливаем величину ошибки

$d_{ij}^{(l)} = 0$

для всех l, i, j.

Устанавливаем выход первого слоя нейронов

$a^{(1)} = x^{(i)}$

Шаг 1. Вычисляем сигнал на выходе сети, выполняя расчет для каждого нейрона в каждом слое, как показано в выражениях 3.4, 3.5, 3.6 и 3.7.

Шаг 2. Используя полученный результат на выходе сети $a^{(L)} = h_\theta{}^{(i)}(x)$ и необходимое для данного примера выходное значение $y^{(i)}$, рассчитываем ошибку выходного слоя $\delta^{(L)} = a^{(L)} - y^{(i)}$, где L – количество слоев нейронной сети.

Шаг 3. «Возвращаем» ошибку, распространяя ее обратно по сети с учетом значения производной:

$\delta^{(L-1)} = \Theta^{(L-1)}\delta^{(L)} .* g'(z^{L-1})$,

где знак .* – символ поэлементного умножения, g' – производная.

При этом

$g'(z^{L-1}) = a^{(L-1)} .* (1 - a^{(L-1)})$

Шаг 4. Модифицируем веса сети с учетом значения ошибки для всех слоев $i \in L$:

$\Theta^{(i)} = \Theta^{(i)} + \rho\delta^{(i)}a^{(i)}$,

где i – номер слоя сети, ρ – параметр обучения *(learning rate)* (0< ρ <1), $\Theta^{(i)}$ – матрица весов слоя i, $\delta^{(i)}$ – рассчитанное значение ошибки i-го слоя.

Шаги алгоритма повторяют либо до достижения некоторого значения ошибки, либо определенное число раз.

Выполнение алгоритма достаточно длительный процесс, и для его ускорения применяют несколько подходов.

Во-первых, это использование функции гиперболического тангенса (вместо сигмоидальной) в качестве активационной функции, например, следующего вида:

$$th(z) = \frac{2\alpha}{1 + e^{-\beta z}} - \alpha, \qquad (3.9)$$

где α и β – некоторые константы, например, α=1.716 и β = 0.667.

График функции th(z) показан на рисунке 3.10. Сигмоидальная функция и функция гиперболического тангенса приведены в сравнении на рисунке 3.11.

Видно, что функция th(z) более «динамична», растет значительно быстрее при увеличении аргумента по сравнению с сигмоидой.

Во-вторых, для ускорения обучения применяют «постоянную импульса» *(momentum constant)*, изменяя правило модификации весов:

$$\Theta^{(i)} = \Theta^{(i)} + \mu\rho\delta^{(i)}a^{(i)} + \rho\delta^{(i)}a^{(i)},$$

где μ – momentum constant, 0< μ <1.

В-третьих, можно изменять параметр обучения (*learning rate*) ρ в зависимости от изменения среднеквадратической ошибки сети, используя следующие эмпирические правила [101]:

1. Если изменение среднеквадратической ошибки сети имеет тот же знак, что и на предыдущем шаге алгоритма, то параметр обучения следует увеличить.

2. Если изменение среднеквадратической ошибки сети имеет другой знак, чем на предыдущем шаге алгоритма, то параметр обучения следует уменьшить.

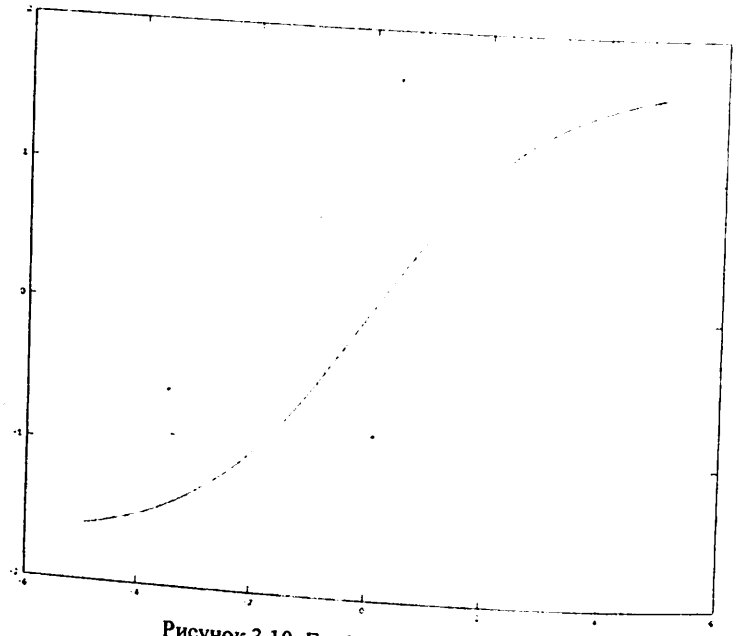


Рисунок 3.10. График функции th(z)

3.1.4.1 Алгоритм k ближайших соседей (k-Nearest-Neighbor – k-NN)

Алгоритм [102] [103] основан на подсчете количества объектов каждого класса в сфере (гиперсфере) с центром в распознаваемом (классифицируемом) объекте. Классифицируемый объект относят к тому классу, объектов которого больше всего в этой сфере. В данном методе предполагается, что веса выбраны единичными для всех объектов.

Если веса не одинаковы, то вместо подсчета количества объектов можно суммировать их веса. Таким образом, если в сфере вокруг распознаваемого объекта 10 эталонных объектов класса А весом 2 и 15 ошибочных/пограничных объектов класса Б с весами 1, то классифицируемый объект будет отнесен к классу А.

Веса объектов в сфере можно представить как обратно пропорциональные расстоянию до распознаваемого объекта. Таким образом, чем ближе объект, тем более значимым он является для данного распознаваемого объекта.

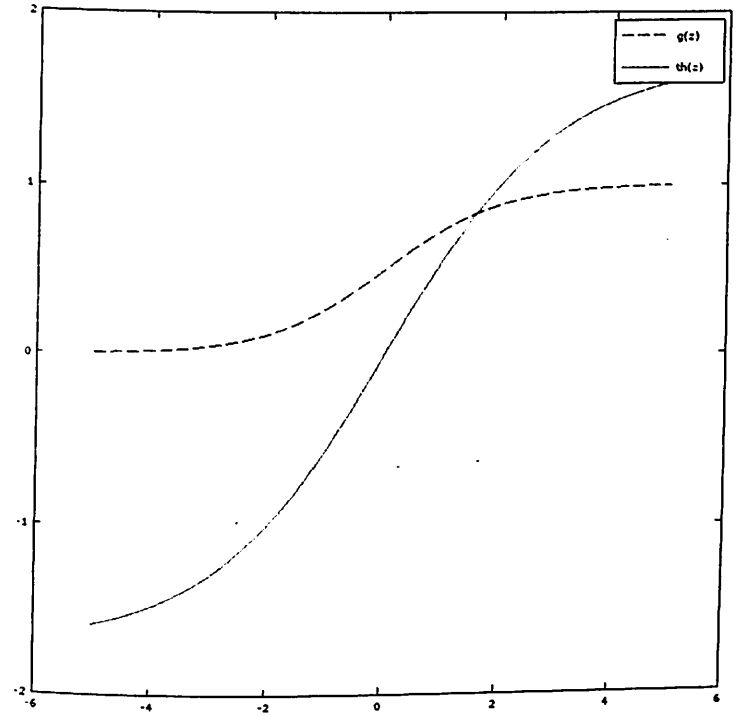


Рисунок 3.11. Графики сигмоидальной функции и функции гиперболического тангенса

В итоге метрический классификатор можно описать так:

$$a(u; X^l) = \arg\max_{y \in Y} \sum_{i=1}^{l} [y_u^{(i)} = y] w(i,u),$$

где w(i, u) – вес i-го соседа распознаваемого объекта u;
a(u;X^l) – класс объекта u, распознанный по выборке X^l.

Радиус гиперсферы может быть как фиксированным, так и изменяемым. Причем в случае с изменяемым радиусом радиус для каждой точки подбирается так, чтобы количество объектов в каждой сфере было одинаковым. Тогда при распознавании в областях с разной плотностью выборки количество «соседних» объектов (по которым и происходит распознавание) будет одинаково. Таким образом, исключается ситуация, когда в областях с низкой плотностью не хватает данных для классификации.

3.1.4.2 Алгоритм опорных векторов (Support Vector Classification – Linear SVM и Non-linear SVM)

Этот алгоритм [104] относится к группе граничных методов: он определяет классы при помощи границ областей. В основе метода лежит понятие плоскостей решений. Плоскость решения разделяет объекты с разной классовой принадлежностью. В пространствах высоких размерностей вместо прямых необходимо рассматривать гиперплоскости – пространства, размерность которых на единицу меньше, чем размерность исходного пространства. В R^3, например, гиперплоскость – это двумерная плоскость.

Метод опорных векторов отыскивает образцы, находящиеся на границах классов (не меньше двух), т.е. опорные векторы, и решает задачу нахождения разделения множества объектов на классы с помощью линейной решающей функции. Метод опорных векторов строит классифицирующую функцию f(x) в виде:

$$f(x) = sign(\langle w, s\rangle + b),$$

где < , > – скалярное произведение;

w – нормальный (перпендикулярный) вектор к разделяющей гиперплоскости;

b – вспомогательный параметр, который равен по модулю расстоянию от гиперплоскости до начала координат. Если параметр b равен нулю, гиперплоскость проходит через начало координат.

Объекты, для которых f(x)=1 попадают в один класс, а объекты с f(x)=-1 – в другой.

С точки зрения точности классификации лучше всего выбрать такую прямую, расстояние от которой до каждого класса максимально. Такая прямая (в общем случае – гиперплоскость) называется оптимальной разделяющей гиперплоскостью. Задача состоит в выборе w и b, максимизирующих это расстояние.

В случае нелинейного разделения существует способ адаптации машины опорных векторов. Нужно вложить пространство признаков Rn в пространство H большей размерности с помощью отображения: $\varphi = Rn \rightarrow H$.

Тогда решение задачи сводится к линейно разделимому случаю, т.е. разделяющую классифицирующую функцию вновь ищут в виде: $f(x) = sign(\langle w, \varphi(x)\rangle + b)$.

Возможен и другой вариант преобразования данных – перевод в полярные координаты:

$$\begin{cases} x_1 = r\cos(\varphi), \\ x_2 = r\sin(\varphi). \end{cases}$$

3.1.4.3 Линейный дискриминантный анализ (Linear Discriminant Analysis Classifier – LDAC) [105]

В математической статистике классификацию часто называют дискриминацией. Дискриминантный анализ – раздел многомерного статистического анализа, который включает статистические методы классификации многомерных наблюдений.

Линейный дискриминантный анализ (LDA) – метод статистики и машинного обучения для поиска линейной комбинации переменных, наилучшим образом разделяющих два или более класса объектов или событий. LDA пытается выразить зависимую переменную (метку класса) через линейную комбинацию других признаков или измерений. Полученная комбинация может быть использована в качестве линейного классификатора. Признаки, использующиеся для отличия одного класса (подмножества) от другого, часто называются дискриминантными переменными.

Пусть обучающая выборка записана в матрицы X_1 и X_2, в которых имеется по I_1 и I_2 строк (объектов). Число столбцов (переменных) одинаково. Исходные предположения состоят в следующем:

- каждый класс (k=1 или 2) представляется нормальным распределением;
- ковариационные матрицы этих двух классов одинаковые $\Sigma_1 = \Sigma_2 = \Sigma$.

Классификационное правило в LDA следующее: новый образец x относится к тому классу, к которому он ближе в метрике Махаланобиса:

$$d_k = (x - \mu_k)\sum^{-1}(x - \mu_k)', k = 1,2$$

На практике неизвестные математические ожидания и ковариационная матрица заменяются их оценками

$$m_k = \frac{1}{I_k}\sum_{i=1}^{I_k} x_i,$$

$$S = \frac{1}{I_1 + I_2 - 2}(\tilde{X}_1'\tilde{X}_1 + \tilde{X}_2'\tilde{X}_2),$$

где $\tilde{X}_k$ – центрированная матрица X_k.

Если приравнять расстояния $d_1=d_2$ в вышеприведенной формуле, можно найти уравнение кривой, которая разделяет классы. При этом квадратичные члены сокращаются и уравнение становится линейным:

$$xw_1' - v_1 = xw_2' - v_2,$$

где

$$w_k = m_k S^{-1},$$

$$v_k = 0.5 m_k S^{-1} m_k'.$$

Величины, стоящие в разных частях уравнения, называются *LDA-счетами*, f1 и f2. Образец относится к классу 1, если f1 > f2, и, наоборот, к классу 2, если f1 < f2.

3.1.4.4 Алгоритм Diagonal Linear Discriminant Analysis (DLDA)

Является разновидностью линейного дискриминантного анализа, описанного выше. Этот алгоритм был предложен в 2002 году [106] как улучшенная модификация LDA для многомерных данных (задач высокой размерности). Когда все плотности классов имеют одинаковые диагональные ковариационные матрицы:

$\Delta=\text{diag}(\sigma_1^2 \ldots \sigma_G^2)$,

классификационное правило будет выглядеть следующим образом:

$$d_k = \sum_{i=1}^{G} \frac{(x_i - \mu_{ki})^2}{\sigma_i^2}.$$

3.2 Оценка качества методов МО

Вследствие обилия методов выбор их для применения в конкретной задаче МО может быть непростым. Естественно, что набор показателей для оценки качества алгоритма зависит от предметной области и цели, поставленной перед системой МО. Чаще всего для того, чтобы сравнить алгоритмы и методы между собой и, возможно, с результатами работы экспертов, используют показатели качества классификации алгоритмов и кривые обучаемости.

Назначение показателей качества – дать оценку, показывающую, насколько классификация или предсказание, выполненное с применением методов МО, отличается от таковой, выполненной экспертами или другим алгоритмом. При этом часто применяют простейший показатель – процент (доля) правильно классифицированных примеров. Для оценки

ошибок 1-го и 2-го рода применяют также еще несколько важных показателей: «точность» *(precision)*, «полнота» *(recall)*, и обобщающие показатели – *T1 Score* и *Kappa*. Их применение особенно важно в случае неравных по объему классов, когда количество объектов одного типа значительно превосходит количество объектов другого типа.

Другим важным показателем применяемого метода МО является его способность обучаться, то есть улучшать свои показатели точности при увеличении числа примеров. Может оказаться, что метод, который показывает очень хорошие результаты на тренировочном множестве примеров, дает неудовлетворительный результат на тестовом множестве. То есть не обладает нужной степенью обобщения. Баланс между способностью обобщения и точностью может быть найден с помощью «кривых обучаемости», которые в общем случае способны показать, может ли тот или иной метод улучшать свой результат так, чтобы показатели качества как на тренировочном, так и на тестовом множестве были примерно равны и удовлетворяли требованиям предметной области исследования.

Третий показатель, который становится особенно важным в задачах с большим объемом данных, – скорость обучения и классификации. Методы ускорения работы алгоритмов МО в задачах с большими данными рассматриваются в разделе «Машинное обучение в задачах с большим объемом данных».

3.2.1 Показатели оценки качества классификации

В настоящее время в задачах машинного обучения для оценки качества классификации наиболее часто используется **доля правильных ответов (accuracy) или Correct Classification Rate (CCR)** – относительное количество корректно классифицированных примеров (процент (доля) правильно классифицированных примеров):

$$Ac = \frac{N_t}{N},$$

где N_t – количество корректно классифицированных примеров, N – общее число объектов.

Этот показатель является весьма важным, однако если количество объектов в классах существенно *неравное* (так называемые неравномерные, или «перекошенные», классы – *skewed classes)*, то может случиться так, что очень плохой классификатор будет давать большое значение *Ac*. Например, если объектов 1-го типа 90% от всего числа объектов, а объектов 2-го типа только 10%, то классификатору достаточно отвечать всегда, что он распознал объект 1-го типа, и доля

правильных ответов достигнет 90%. Таким образом, даже если алгоритм никогда правильно не распознает объект класса 2, он все равно будет иметь высокий показатель *Ac*. При этом, если распознавание объектов 2-го класса исключительно важно, показатель *Ac* будет попросту вводить в заблуждение. Для того чтобы избежать подобной неадекватной оценки, рассматривается еще несколько важных показателей: «точность» *(precision)*, «полнота» *(recall)*, и обобщающий показатель – *T1 Score (гармоническое среднее или F мера)*, которые рассчитываются с помощью следующих выражений:

Precision: $$P = \frac{T_p}{(T_p + F_p)}$$

Recall: $$R = \frac{T_p}{(T_p + F_n)}$$

T1 Score: $$T1Score = \frac{2PR}{(P+R)}$$

Поясним приведенные выражения.

Рассмотрим случай классификации двух классов (или одного класса номер 1 и всех остальных классов, которым присвоим номер 0). В этом случае возможны следующие ситуации:

		Реальный класс (Actual class)	
		1	0
Предсказанный класс (Predicted class)	1	**True positive**	False positive
	0	False negative	**True negative**

Случаи *True positive (Tp)* и *True negative* являются случаями правильной работы классификатора, соответственно, *False negative (Fn)* и *False positive (Fp)* – случаями неправильной работы. При этом *Fn* можно рассматривать как признак излишне пессимистического (осторожного) классификатора, *Fp* – наоборот, как признак излишне оптимистического, или неосторожного, классификатора. Тогда

Precision: $$P = \frac{T_p}{(T_p + F_p)}$$

будет показывать часть правильно распознанных объектов заданного класса по отношению к общему числу объектов, принятых классификатором за объекты заданного класса. С другой стороны,

Recall: $$R = \frac{T_p}{(T_p + F_n)}$$

будет показывать отношение правильно распознанных объектов к общему числу объектов данного класса.

Оба показателя – и *P*, и *R* – отражают «путаницу» классификатора. Однако *P* показывает, насколько классификатор оптимистичен в своих оценках или как часто он «любит» (низкое значение *P)* присоединять объекты других классов к заданному. В то время как *R* показывает, насколько классификатор «пессимистичен» в своих оценках, то есть как часто он «отбрасывает» (низкое значение *R)* объекты нужного класса.

Разумеется, желательно, чтобы оба этих показателя стремились к 1. Для некой усредненной оценки применяют

$$T1Score = \frac{2PR}{(P+R)},$$

который, как видно из формулы, также стремится к 1, если оба показателя, *P* и *R*, близки к 1.

Отметим, что использование простого усреднения Average = (P + R)/2 может привести к тому, что мы получим неверное представление о свойствах алгоритма. Например, пусть имеется три алгоритма, показывающие следующие оценки *precision* и *recall:*

	Precision (P)	Recall (R)	Average	T1 Score
Algorithm 1	0.55	0.44	0.495	0.4888889
Algorithm 2	0.71	0.12	0.415	0.2053012
Algorithm 3	0.03	1	0.515	0.0582524

Видно, что простое среднее (колонка *Average)* дает высшую оценку совершенно негодному алгоритму 3, который практически все объекты ошибочно принимает за искомый *(P* очень мало). В то же время *T1 Score* показывает более корректный результат, отдавая высший балл алгоритму 1, который показывает близкие оценки *precision* и *recall* и, следовательно, более взвешен в своих оценках.

Показатель *Kappa* более робастный по сравнению с показателем точности, который представляет собой просто процентное отношение правильно распознанных объектов к общему числу объектов. Впервые предложен Кохеном для сравнения рейтингов людей в дихотомических (бинарных) задачах классификации [107]. В настоящее время активно

используется в известных пакетах программ [108]. Рассчитывается следующим образом.

Пусть имеется матрица ошибок *(error matrix/confusion matrix)*, в которой на главной диагонали расположены правильные ответы, а цифры вне главной диагонали представляют собой ошибочные результаты, причем n_{ij} – количество объектов, классифицированных экспертом как объект класса j, а системой как объект класса i. Также можно определить количество объектов, классифицированных как объекты класса i:

$$n_{i.} = \sum_j n_{ij},$$

количество объектов, классифицированных как объекты класса j:

$$n_{.j} = \sum_i n_{ij}.$$

Используя матрицу ошибок $N = n_{ij}$, статистический показатель *Kappa* определяется следующим выражением:

$$K = \frac{O^C - p_e}{1 - p_e},$$

где p_e – процент корректно классифицированных объектов при изменении

$$p_e = \frac{\sum_i n_{.i} n_{i.}}{n^2}.$$

При этом

$$O^c = \frac{\sum_{i=1}^{k} n_{ii}}{|T|},$$

где T можно интерпретировать как общее количество объектов, а сумма определяет количество корректно распознанных объектов (сумма цифр на главной диагонали матрицы ошибок).

Кроме «обычного» показателя *Kappa* также используются:

- Kappa Location.
- Kappa Histo.
- Kappa No.
- Weighted Kappa.

А также показатели:

- weighted mean recall;
- weighted mean precision;
- spearman rho;
- kendall tau;
- absolute error;
- relative error;
- relative error lenient.

В [109] приводится всесторонний анализ семейства показателей *Kappa*.

Для более тонкой оценки разрабатываемых алгоритмов применяют также показатели ошибок, рассчитанные на части выборки: ошибка на контрольной выборке, ошибка скользящего контроля, и методы контроля: контроль по фолдам, контроль на случайной подвыборке.

Ошибка на контрольной выборке. Множество объектов с известным значением функции y разбивают на обучающую выборку $\{x_t\}_{t=1}^{m}$, на которой настраивают (обучают) алгоритм A, и контрольную выборку $\{x_t\}_{t=1}^{q}$, на которой проверяют качество работы.

Ошибка скользящего контроля. Для некоторого натурального k проводят всевозможные разбиения выборки на две части: с k и m – объектами k. Первую выборку считают контрольной, вторую – обучающей.

Контроль по блокам (фолдам). Выборку разбивают на k блоков (их иногда называют фолдами). Проводят k экспериментов, используя в i-м i-й блок в качестве контрольной выборки, а объединение остальных блоков – в качестве обучающей выборки.

Контроль на случайной подвыборке. Случайным образом берут часть выборки в качестве контрольной, а остальную часть назначают обучающей. Проводят серию экспериментов. Усредняют результаты по числу экспериментов. Часто разбиение делают методом бутстреп.

Кроме того, в таких задачах, где числа представителей классов сильно различаются, по-особому может рассчитываться функция стоимости (ошибки), например, в случае двух неравновесных классов *(skewed classes)*:

$$J(\theta) = \min \sum_{k=1}^{2} \frac{1}{|\{t \mid y(x_t) = k\}|} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2.$$

3.2.2 «Обучаемость» алгоритмов

Оценка алгоритмов распознавания путем сравнения *accuracy* (точности) или иного показателя качества *(Kappa, precision, recall)* обладает тем недостатком, что не дает возможности оценить алгоритмы в динамике изменения объема обучающей выборки. В частности, если говорить о нейронных сетях, то на показатели точности существенно влияет количество скрытых слоев и количество тренировочных примеров. При использовании линейной регрессии ее порядок, для *k-NN* – радиус окружности ближайших соседей и т.п. При этом важно учесть способность алгоритма обучаться, переобучаться *(overfit)* или недообучаться *(underfit)*. Правильный баланс между *underfit* и *overfit* означает поиск такого алгоритма и его параметров, которые были бы способны показать приемлемые результаты как на обучающем, так и на тестовом множестве (или множестве *cross validation)*. Недообученный алгоритм будет показывать одинаково плохие результаты и на тестовом, и на обучающем множествах, в то время как переобученный будет демонстрировать высокий результат на обучающем множестве и низкий на тестовом. Представим для случая регрессии соответствующие формулы кривых, экстраполирующих распределение тренировочных примеров так, как показано ниже:

A. $\Theta_0 + \Theta_1 x$ *– high bias (underfit)*

B. $\Theta_0 + \Theta_1 x + \Theta_2 x^2$ *– just right*

C. $\Theta_0 + \Theta_1 x + \Theta_2 x^2 + \Theta_3 x^3 + \Theta_4 x^4$ *– high variance (overfit)*

Результаты экстраполяции при некотором гипотетическом распределении объектов тренировочного множества показаны на рисунке 3.12.

При этом показатели ошибки на тренировочном *(train)* и тестовом множествах *(cross validation – cv)* определяют по идентичным формулам (меняется лишь набор примеров):

$$J_{train}(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

$$J_{cv}(\theta) = \frac{1}{2m_{cv}}\sum_{i=1}^{m_{cv}}(h_\theta(x_{cv}^{(i)}) - y_{cv}^{(i)})^2$$

где m – тренировочное множество примеров, m_{cv} – тестовое множество примеров *(cross validation – cv)*;

h_θ – функция гипотезы, которая может быть линейной ($h_\theta = \theta_0 + \theta_1 x$) или нелинейной (например, $h_\theta = \theta_0 + \theta_1 x + \theta_2 x^2$) с различным набором параметров $\theta_i \in \Theta$.

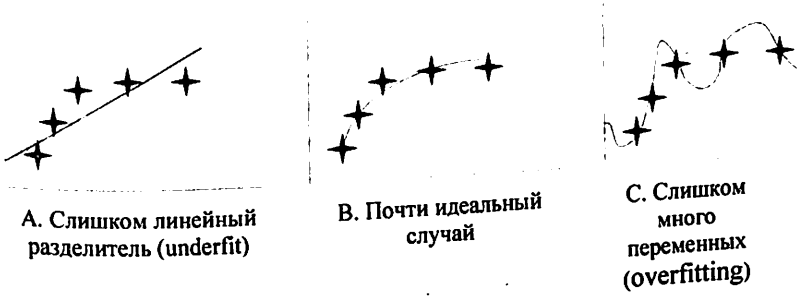


Рисунок 3.12. Иллюстрация недотренированности и перетренированности алгоритма МО

Параметры модели, определяющие функцию h(Θ), рассчитываются с помощью тренировочного множества, а проверяются с помощью примеров из тестового множества.

Забегая вперед, можно сказать, что для компенсации излишних переменных в случае переобучения в регрессионной модели применяют регуляризацию, добиваясь, чтобы переменные с более высоким показателем степени оказывали меньшее влияние. Формула оценки стоимости с учетом регуляризации следующая:

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda\sum_{j=1}^{n}\theta_j^2$$

где λ – параметр регуляризации.

В случае использования нейронной сети роль, аналогичную регуляризации, выполняет уменьшение числа скрытых слоев нейронной сети.

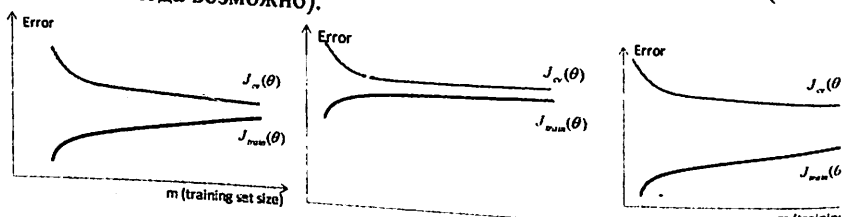
Использование регуляризации или уменьшение числа скрытых слоев увеличивает способности обобщения алгоритма машинного обучения и, соответственно, снижает способность к обучению в смысле гибкости настройки на тонкие различия между классами.

Отметим, что системы машинного обучения можно разделить на более линейные *(high bias)*, которые обладают сравнительно малой способностью к формированию сложных интерполяционных кривых, и системы с высокой вариативностью *(high variance)*, которые способны

формировать кривые (поверхности, гиперповерхности) сложной формы. Поведение этих алгоритмов (моделей) различается при увеличении числа тренировочных примеров. Первые, как правило, обобщают результаты, часто не учитывая некоторых, возможно существенных, различий между обучающими примерами. Вторые, напротив, «отслеживают» все нюансы, возможно случайные, но в то же время недостаточно обобщают. Для первых характерна недотренированность, в то время как для вторых – перетренированность (рисунок 3.12). Оценить способности модели при однократном эксперименте, как правило, невозможно, поскольку и первые, и вторые могут давать близкие показатели ошибок.

В связи с этим обучаемость алгоритмов МО оценивают с помощью так называемых кривых обучения *(learning curves)*, которые строят, рассчитывая показатели ошибок при постепенно увеличивающемся числе обучающих примеров. Построив кривые ошибок на тестовом и обучающем множествах, оценивают алгоритм посредством следующих эмпирических закономерностей:

- В нормальной ситуации, при хорошем алгоритме, при увеличении числа тренировочных примеров ошибка на тренировочном множестве немного возрастает, а ошибка на тестовом множестве снижается (рисунок 3.13a).
- Если система сравнительно линейна, увеличение числа обучающих примеров принесет мало пользы. Ошибка и на тренировочном, и на тестовом множествах будет примерно одинаковой и большой (рисунок 3.13b).
- Если система с высокой вариативностью, увеличение числа обучающих примеров приведет к снижению величины ошибки на тестовом множестве, однако будет существенно отличаться от ошибки на тренировочном множестве (рисунок 3.13c). Для еще большего снижения ошибки на тестовом множестве можно значительно увеличить тренировочное множество (что не всегда возможно).



a – нормальная ситуация, обучаемый алгоритм с хорошей способностью к обобщению

b – трудно обучаемый алгоритм, система слишком линейна (high bias)

c – алгоритм с высокой вариативностью (high variance)

Рисунок 3.13. Кривые обучения (learning curves)

Таким образом, для того, чтобы оценить, к какой из двух групп принадлежит исследуемый алгоритм (слишком линейный или слишком гибкий), рекомендуется исследовать кривую ошибок обучения при увеличении размера обучающего множества. Например, если кривые показывают сходимость, но при этом высокий уровень ошибок, то это может свидетельствовать о линейности модели (невозможности ее обучить).

При обнаружении нежелательных свойств алгоритма можно попытаться настроить его, изменить объем обучающего множества примеров или выбрать дополнительные свойства объектов, учитывая следующее:

- Увеличение размера тренировочного множества полезно при высокой вариативности алгоритма (много слоев нейронной сети, высокий порядок регрессии), когда программа не обладает нужной степенью обобщения, а настраивается в большой мере на тренировочный набор примеров и не может нормально классифицировать примеры из тестового множества (ошибка переобучения).
- Сокращение числа используемых свойств или параметров полезно при высокой вариативности алгоритма (много слоев нейронной сети, высокий порядок регрессии), то есть вновь в тех случаях, когда присутствует переобучение, но в то же время количество обучающих примеров невозможно существенно увеличить.
- Использование дополнительных свойств полезно при слишком линейных алгоритмах (низкий порядок регрессии, мало нейронов в скрытых слоях сети или мало скрытых слоев), когда программа и на тестовом, и на тренировочном наборе будет показывать одинаково плохие результаты (ошибка недообученности).
- Использование специальных синтезированных (полиномиальных) свойств, представляющих более высокие степени и произведения от основных (x_1^2, x_2^2, $x_1 x_2$, ...), также полезно при слишком линейных алгоритмах (низкий порядок регрессии, мало слоев нейронной сети) (недообученная модель).

3.2.3 Метод сравнительной оценки качества классификации

Для создания какой-либо обучаемой распознающей системы (например, искусственной нейронной сети) крайне важно качество обучающей выборки (далее – ОВ). На практике внутренние параметры и архитектура такой системы менее значимы, чем данные, используемые для обучения.

Распознающая искусственная нейронная сеть в процессе обучения делит пространство признаков на определенные области, соответствующие распознаваемым категориям. После обучения (при использовании ИНС) входные данные будут отнесены к одной из областей пространства признаков, то есть к определенной категории. В случае плохого качества ОВ области могут быть определены некорректно, что приводит к ошибкам при распознавании.

Однако получение идеальной или даже хорошей ОВ часто невозможно (что может быть связано с природой распознаваемых явлений/объектов).

Тем не менее определение предела точности распознавания для конкретного набора обучающих данных является крайне важной задачей при построении распознающей системы.

Очевидно, что метод определения качества не должен зависеть от распознающей системы (в нашем случае от нейронной сети). Поэтому для оценки нижней границы предела распознавания с помощью некоторого алгоритма МО можно использовать метрику сравнительной точности независимых алгоритмов выражающуюся в следующей методике (алгоритме) [110]:

1. Формирование обучающей выборки и ее предобработка (нормировка, сглаживание и т.п.).
2. Использование алгоритмов классификации и оценка качества путем сравнения с ответами экспертов. Назовем данную оценку оценкой А.
3. Использование сформированной обучающей выборки для обучения выбранного алгоритма МО и применение ее для распознавания и оценки качества путем сравнения с ответами экспертов. Назовем данную оценку оценкой В.
4. Сравнение результатов пунктов 2 и 3.
5. Если $B \geq A$, делаем вывод о состоятельности применения данного алгоритма МО для целей распознавания и о состоятельности методов подготовки и применения указанного алгоритма.

Отметим, что случай B=A (где = можно рассматривать как примерное равенство) может свидетельствовать как в пользу дальнейших исследований методов применения алгоритма МО, так и в пользу поиска

новых, отличных от данного алгоритма методов распознавания/классификации. Отметим также, что в силу изложенного выше получение оценок А и В является нетривиальной задачей, зависящей от требований к системе распознавания. Например, если для простых задач А и В могут быть оценками точности *Ac*, то для более сложных случаев неоднородных классов уместнее использовать интегрированную оценку на базе *Ac* и *T1 Score*.

3.3 Предварительная обработка (препроцессирование) данных

Для применения методов МО требуется привести обрабатываемые данные к определенному виду, который позволил бы подать их на вход алгоритмов обучения и анализа. При этом выполняются важные этапы предварительной обработки данных, включающие, как правило, очистку от аномальных значений; нормировку данных; «сглаживание»; переформатирование данных; формирование входного набора данных, которое, например, может включать формирование так называемого плавающего окна данных, необходимое при анализе закономерностей, представленных последовательностями данных; согласование данных, например, когда один набор данных «сдвинут» по времени, расстоянию, спектру и т.п. относительно другого. Отметим, что если набор алгоритмов машинного обучения известен и свойства этих алгоритмов в основном хорошо изучены, то процессы предобработки данных разнообразны и напрямую зависят от предметной области и качества имеющихся данных. Часто к перечисленному выше «классическому» набору добавляются дополнительные этапы, позволяющие в конечном счете повысить качество интерпретации данных.

3.3.1 Устранение аномальных значений

3.3.1.1 Алгоритм поиска аномалий на основе нормального распределения

Алгоритм поиска аномалий на основе нормального распределения основан на предположении, что все множество «правильных» объектов образует распределение Гаусса (нормальное распределение), то есть величины x распределены в соответствии с нормальным законом распределения, определяемым математическим ожиданием - μ и среднеквадратическим отклонением - δ^2:

$$x \sim N(\mu, \delta^2),$$

что графически можно представить рисунком 3.14.

При этом вероятность того или иного значения рассчитывается по известной формуле:

$$p(x;\mu,\delta^2)=\frac{1}{\sqrt{2\pi\delta}}\exp(-\frac{(x-\mu)^2}{2\delta^2})$$

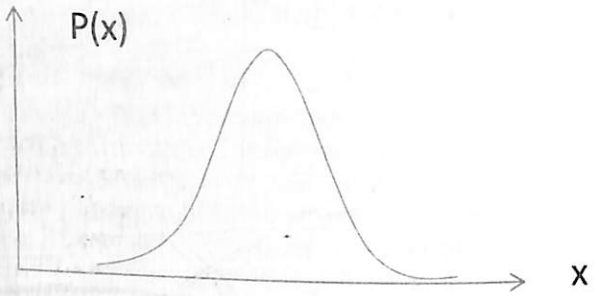


Рисунок 3.14. Примерный график распределения вероятностей объектов *x*, подчиняющихся нормальному (Гауссову) закону

Если свойств у объектов несколько, то вероятности будут подчиняться многомерному распределению Гаусса. Например, при наличии двух свойств объектов распределение вероятностей будет выглядеть как на рисунке 3.15.

$$p(x)=\prod_{j=1}^{n} p(x_j;\mu_j,\delta_j{}^2)$$

При этом , где n – число свойств каждого объекта, подлежащего классификации (число параметров).

Таким образом, алгоритм вычисления аномальных объектов заключается в следующем.

Шаг 1. На базе примеров m из обучающего набора определяются параметры многомерного распределения Гаусса - $\{\mu_1,...,\mu_n;\delta_1^2,...,\delta_n^2\}$:

$$\mu_j=\frac{1}{m}\sum_{i=1}^{m}x_j^{(i)}$$

$$\delta^2{}_j=\frac{1}{m}\sum_{i=1}^{m}(x_j^{(i)}-\mu_j)^2$$

где $x_j^{(i)}$ – j-й параметр объекта x в обучающем примере i из множества примеров m.

Шаг 2. Для каждого нового экземпляра x вычисляется его вероятность:

$$p(x)=\prod_{j=1}^{n} p(x_j;\mu_j,\delta_j{}^2)\ \prod_{=j=1}^{n}\frac{1}{\sqrt{2\pi\delta_j}}\exp(-\frac{(x_j-\mu_j)^2}{2\delta_j{}^2})$$

Шаг 3. Если полученная вероятность меньше некоторого порогового значения ε, то данный объект x считается аномальным

if $p(x)<\varepsilon$ then x is anomaly.

На практике обучение системы поиску аномальных значений заключается в определении порогового значения ε. Подбор данной границы может осуществляться на базе имеющихся примеров (примерно так, как это делается в алгоритмах обучения с учителем). После серии опытов устанавливается такая граница ε, чтобы все (или большая часть) «неправильные» объекты детектировались системой как аномальные.

Графически используемый подход к поиску аномалий иллюстрируется рисунком 3.16.

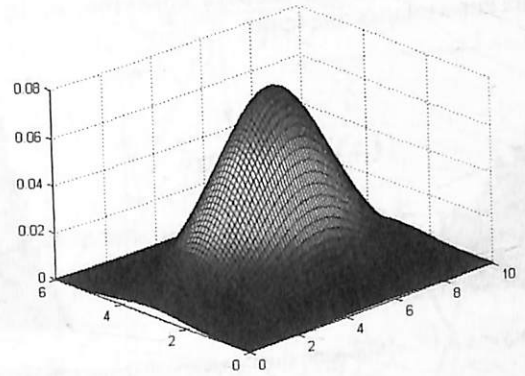


Рисунок 3.15. Распределение вероятностей объектов x, подчиняющихся многомерному нормальному (Гауссову) закону

Отметим, что если распределение отличается от нормального, то можно нормализовать его, вычислив логарифм от x или степень от значений x. Применение логарифма может нормализовать несимметричные распределения (рисунок 3.17).

Рисунок 3.16. Исключение аномальных объектов

Очевидно, что алгоритмы обучения с учителем вполне пригодны для поиска аномальных объектов. Тогда аномальные объекты представляют отдельный класс объектов, которые можно детектировать, используя алгоритмы *k-NN* или логистическую регрессию. В обоих случаях необходим подбор тех свойств объектов, которые существенны для выделения аномальных значений.

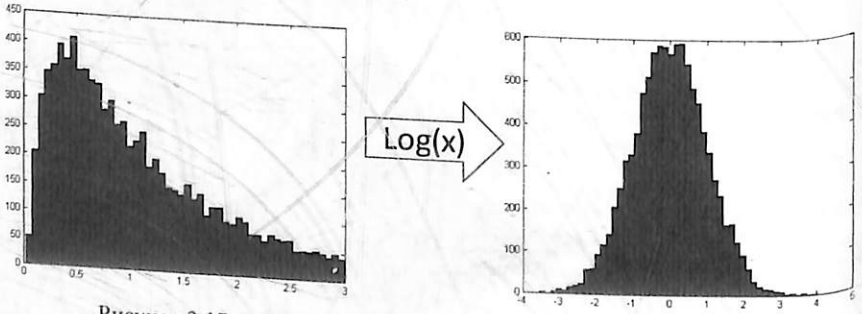


Рисунок 3.17. «Нормализация» распределения вероятностей

3.3.1.2 Использование методов МО для поиска аномалий

3.3.2 Методы нормировки и центрирования данных

Линейная нормировка

Для очистки данных от аномальных значений и удобства работы проводят нормирование и центрирование данных, способствуя тому, чтобы каждая компонента входного вектора находилась на отрезке от 0 до 1 или от –1 до 1. При известном диапазоне изменения входной переменной можно использовать простейший вид преобразования:

$$p = \frac{(x - x_{\min})(b - a)}{(x_{\max} - x_{\min})} + a$$

где $[a, b]$ – диапазон приемлемых входных сигналов; $[x_{\min}, x_{\max}]$ – диапазон изменения значений входной переменной; p – преобразованный входной сигнал.

Проблемой при применении данного способа преобразования (линейной нормировки) может быть большой диапазон изменения значений входной переменной. В таких случаях можно использовать преобразование по формулам сигмоидальной функции или гиперболического тангенса. Забегая вперед, отметим, что при обработке данных электрического каротажа именно линейная нормировка, описанная выше, показала наилучший результат.

При кодировании количественных переменных необходимо в общем случае учесть содержательное значение признака, его расположение на интервале значений, точность измерения. Преобразование может быть выполнено с помощью двух выражений:

$$x_i = \frac{x_i - M(x_i)}{q(x_i)}$$

или

$$x_i = \frac{x_i - M(x_i)}{\max |x_i - M(x_i)|},$$

где x_i – i-я координата входного вектора X.

Выборочная оценка математического ожидания x_i (среднее значение):

$$M(x_i) = 1/n \sum_{i=1}^{n} x_i$$

Выборочная оценка среднеквадратического отклонения:

$$q(x_i) = \sqrt{(1/n \sum_{i=1}^{n} (x_i - M(x_i))^2}$$

Для того чтобы сделать значимыми малые изменения больших величин (например, когда значение входной переменной может достигать 10 000, а для анализа существенным является изменение величины на 1), применяются три вида предобработки числовых данных – модулярный, функциональный и позиционный. Для учета малых изменений каждое значение кодируется не единственным значением, а вектором, формируемым по описываемым ниже правилам.

Модулярная предобработка. Задается набор положительных чисел $y_i,\ldots,y_k$. Рассчитываем каждую компоненту вектора Z следующим образом:

$$z_i = \frac{((x \bmod y_i) + y_i)(b-a)}{2y_i} + a,$$

где $[a, b]$, как и ранее, – диапазон приемлемых входных сигналов.

Поясним, что при вычислении $x \bmod y$ возвращается остаток от деления числа x на y, например, если x = 5, y = 3, x mod y = 2.

x			
5			
10			
15			

Функциональная предобработка. В общем случае отображение входного признака x в k-мерный вектор Z происходит следующим образом. Выбирается k чисел, удовлетворяющих условию

$$x_{\min} < y_i < \ldots < y_k < x_{\max},$$

вычисляются элементы вектора Z

$$z_i = \frac{(\phi(x-y_i) - \phi_{\min})(b-a)}{\phi_{\max} - \phi_{\min}} + a,$$

где ϕ – функция, определенная на интервале $[x_{\max} - y_k, x_{\min} - y_1]$, а $\phi_{\max}$, $\phi_{\min}$ – максимальное и минимальное значения функции на этом интервале.

Позиционная предобработка [111]. Подход в этом случае примерно такой же, как при построении позиционных систем счисления. Выбирается положительная величина y, такая, что $y^k \geq (x_{\max} - x_{\min})$.

Сдвинем параметр x так, чтобы он принимал только положительные значения и ноль. Для вычисления вектора Z используем следующие формулы:

$$z_o = (x - x_{\min}) \bmod y,$$

$$z_1 = ((x - x_{\min})/y) \bmod y,$$

$$z_k = ((x - x_{\min})/y^k) \bmod y.$$

Другие способы преобразования входных значений – возведение в степень, извлечение корня, взятие обратных величин, вычисление экспоненты и логарифмов, а также определенные комбинации переменных – произведения, частные и т.п., которые могут уменьшить размерность входного вектора данных. Ниже также рассматриваются методы очистки данных от шумов на основе вейвлет-анализа и преобразования Фурье.

3.3.3 «Сглаживание» данных и устранение рассогласований

В силу специфики предметной области дополнительно рассматривались методы сглаживания и устранение сдвига каротажных данных относительно друг друга.

Кроме нормировки данных для повышения качества распознавания могут быть использованы различные методы сглаживания, позволяющие устранить шум и иные неинформативные составляющие данных. В качестве методов сглаживания часто рассматриваются два вида дискретных преобразований: анализ Фурье и вейвлет-преобразование.

Преобразование Фурье раскладывает входной сигнал на ряд гармонических функций (гармоники), после чего некоторые из них можно исключить и «собрать» сигнал заново. Такая фильтрация хорошо подходит для стационарных шумов, например Гауссовского. Причем если полезная информация о сигнале содержится в низкочастотных гармониках, а шум – в высокочастотном диапазоне, то такой подход позволяет легко удалить одну из составляющих, например высокочастотную, а с ней вместе и шум. Таким образом, процесс сглаживания происходит следующим образом: сигнал раскладывается на гармоники, выбирается определенная частота, и функции с превышением данного показателя считаются шумом и удаляются из общего сигнала.

Вейвлет-преобразование локализовано по времени, т.е. позволяет фильтровать данные отдельно на небольших участках, иными словами, может быть использовано для фильтрования нестационарных шумов, таких как шум от работы двигателя, нестабильность электромагнитного поля и т.п. Суть вейвлет-разложения схожа с Фурье, однако вместо

гармонических функций используются небольшие, локализованные по времени «маленькие волны» (буквальный перевод слова *wavelet)*. Сам же метод вейвлет-преобразования позволяет рассматривать и анализировать сигнал на разных масштабах (поэтому метод иногда называют математическим микроскопом). Сглаживание представляет собой нахождение матрицы коэффициентов разложения (числа, на которые нужно умножить вейвлет, параллельно перенесенный на определенную точку пространства), выбор порогового значения шума, обнуление всех коэффициентов, меньших, чем пороговое значение, и восстановление сигнала по оставшимся ненулевым коэффициентам.

На практике дополнительные погрешности могут вносить характерные именно для данной предметной области ошибки. Например, фактором, способным повлиять на качество предварительной обработки данных, является сдвиг кривой ИК относительно кривых ПС и КС. КС и ПС снимаются одним прибором, а ИК – другим. Вследствие этого из-за различных факторов (неточность начальной установки, усадка почвы и т.п.) кривая ИК может быть смещена по глубине относительно кривых ПС и КС. Данные ИК могут быть сопоставлены с другими показателями глубины. Обычно это смещение невелико и постоянно. Например, в ряде случаев ИК смещается на величину до 0,5 метра в большую или меньшую сторону.

В процессе проведения экспериментов оценивалось влияние на качество распознавания вида нормировки, сдвига каротажных данных, сглаживания данных, использования дополнительных параметров (глубины), исключение сдвига каротажных кривых.

3.4 Машинное обучение в задачах с большим объемом данных

Основная проблема применения методов машинного обучения в классификационных или регрессионных задачах с большим объемом данных (больших данных) заключается в вычислительной сложности расчета функций стоимости (3.10) и соответствующих параметров функции гипотезы (3.11) вследствие большого количества примеров. Напомним, что

$$J(\theta) = \min \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2, \qquad (3.10)$$

где m – множество примеров, h_θ – функция гипотезы, и

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1), \qquad (3.11)$$

где α – параметр обучения, $\frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$ является производной функции стоимости по θ_j, знак := означает присваивание.

Например, стандартный алгоритм градиентного спуска – это итеративная процедура, выполняемая путем сравнительно большого количества элементарных шагов.

Предположим, что мы имеем 100 000 000 примеров, 10 параметров и вычисляем параметры в среднем за 500 итераций градиентного спуска. Количество требуемых вычислений составит $5*10^{11}$. В предположении, что компьютер способен вычислить 1 млн. итераций градиентного спуска в минуту, получаем порядка 140 часов для вычисления только одного набора параметров функции гипотезы. При необходимости построения кривой обучаемости время расчета возрастет пропорционально количеству точек на кривой и может достичь совершенно неприемлемых значений. Использование матричных операций (формула 2.3) хотя и устраняет итерации, но становится невозможным в силу большой вычислительной сложности получения обратной матрицы.

Для преодоления «проклятия размерности» алгоритма градиентного спуска в случае больших объемов данных предложено 2 алгоритма: стохастического градиентного спуска *(Stochastic Gradient Descent – SGD)* [112] и мини-пакетного градиентного спуска *(Mini-Batch Gradient Descent – MBGD* [113]).

Алгоритм SGD

Алгоритм работает следующим образом:

1. Вначале случайно переупорядочиваются примеры из обучающего множества.

2. По всем примерам m из обучающего множества для каждого из параметров n вычисляется новое значение параметра. На псевдокоде для *SGD* можно записать:

```
for iter := 1 to K
    for i := 1 to m
        for j := 1 to n
            θ_j := θ_j − α(h_θ(x^(i)) − y^(i)) x_j^(i)
        end
    end
```

end

Количество итераций K = 1,...,10.

В то же время для алгоритма пакетного градиентного спуска *(Batch Gradient Descent – BGD)* псевдокод выглядит следующим образом:

Do

for i := 1 to m

for j := 1 to n

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

end

end

enddo

Таким образом, на каждом из шагов m алгоритма *BGD* производится трудоемкое суммирование, в то время как в *SGD* используется только один пример на каждой итерации. Оценка вычислительной сложности *SGD* – $O(m)$ по сравнению с $O(m^2)$ для *BGD* при условии, что число параметров n много меньше m. Однако применение *SGD* приводит к тому, что в результате работы будет найдено приближенное решение, а не глобальный минимум функции стоимости. Кроме этого, *SGD*, так же как и описываемый ниже *MBGD*, не свободен от проблемы сходимости. То есть их результат может не улучшаться, а даже ухудшаться с увеличением количества обработанных примеров.

Алгоритм MBGD

Алгоритм на каждой итерации использует только часть (b) примеров:

for iter := 1 to K

for i := 1 to m with step b

for j := 1 to n

$$\theta_j := \theta_j - \alpha \frac{1}{b} \sum_{k=i}^{k+b} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

end

end

end

Его вычислительная сложность может быть оценена как $O(bm)$, что при условии b<<m может быть сведено к $O(m)$.

Кроме упомянутых *SGD* и *MBGD* для преодоления проблемы большого количества вычислений можно применить распараллеливание вычислений [114] [115], которое при большом количестве независимых

процессов может позволить решать задачу нахождения параметров функции гипотезы с помощью *BGD* за приемлемое время.

MapReduce

Суть метода, называемого *MapReduce*, заключается в следующем. Расчет суммы

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

выполняется на отдельных машинах или процессорах. В случае, если таких процессоров B штук, расчет может быть выполнен следующим образом:

$$Sum_{j0} := \sum_{i=1}^{m/B} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)},$$

$$Sum_{j1} := \sum_{i=(m/b)+1}^{m/B+m/B} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)},$$

...

$$Sum_{jk} := \sum_{i=k*(m/b)+1}^{m/B+k*(m/B)} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)},$$

...

$$Sum_{jB-1} := \sum_{i=(B-1)*(m/b)+1}^{m/B+(B-1)*(m/B)} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)},$$

$$\theta_j := \theta_j - \alpha \frac{1}{m} \left(\sum_{k=0}^{B-1} Sum_{jk}\right).$$

Например, если число процессоров 3, а число примеров 300 000:

$$Sum_{j0} := \sum_{i=1}^{100000} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)},$$

$$Sum_{j1} := \sum_{i=100001}^{200000} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)},$$

$$Sum_{j3} := \sum_{i=200001}^{300000} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}.$$

$$\theta_j := \theta_j - \alpha \frac{1}{300000} (\sum_{k=0}^{3} Sum_{jk})$$

Очевидно, что вычислительная сложность метода может быть оценена как $O(\frac{m^2}{B})$. При малых значениях B увеличение скорости вычислений будет незначительным, но при числе процессоров, сопоставимых с числом процессоров в современных суперкомпьютерных кластерах (1000–1 000 000), увеличение скорости может достигать нескольких порядков.

Применение машинного обучения в работе с большими данными имеет серьезные перспективы. Предлагаются специальные языки и платформы для реализации потенциала машинного обучения при работе с большими данными. Например, в [116] предложен язык *(SystemML)* и описаны способы его применения для реализации методов машинного обучения в больших кластерах *MapReduce*. В [117] описана платформа *MLBase*, обеспечивающая постановку задач и применение машинного обучения вместе с операторами высокого уровня для доступа к данным.

3.5 Заключение по разделу 3

Сфера искусственного интеллекта весьма обширна и включает в себя множество направлений, начиная от логических рассуждений и заканчивая методами оценки тональности текстов. Традиционно выделяют так называемый сильный искусственный интеллект и слабый искусственный интеллект *(Weak AI – WAI)*. Первый ориентирован на создание систем, выполняющих присущие человеку высокоинтеллектуальные задачи, в конечном счете на создание мыслящих машин. *WAI* ориентирован на создание приложений, в которых реализуется та или иная интеллектуальная способность человека или животных. Потенциал, заложенный в идее *WAI*, осуществляется с помощью машинного обучения.

В разделе рассмотрены методы машинного обучения как часть методов *WAI*, пригодные для анализа в том числе больших данных. Сформирована таксономия методов МО. Методы обучения с учителем описаны подробнее, так как именно их использование позволяет автоматизировать процесс решения задачи литологического расчленения скважин урановых месторождений.

Описана схема настройки методов машинного обучения для решения задачи. Приведена формальная постановка задачи МО и описаны некоторые часто используемые алгоритмы (линейная регрессия, полиномиальная регрессия, логистическая регрессия, искусственные нейронные сети, алгоритм *k-NN, SVN, LDAC, DLDA)*. Описаны показатели оценки точности классификации: доля правильных ответов *(accuracy)*, «точность» *(precision)*, «полнота» *(recall)*, и обобщающие показатели – *T1 Score, Kappa*. Дано понятие обучаемости методов машинного обучения и описано использование его на практике (способы интерпретации кривой обучения) для выбора подходящего метода или его настройки. Подробно описаны методы предобработки данных, включая методы устранения аномальных значений, нормировки. Кратко рассмотрены проблемы применения систем МО в задачах с большим объемом данных и способы их решения методами распараллеливания вычислений и «огрубления» алгоритма градиентного спуска.

Развитие методов машинного обучения идет вместе с их практическим использованием, в результате чего увеличивается количество приложений, появляются специальные методы для решения прикладных задач, развиваются методы комитетного синтеза, предлагаются платформы и языки, в том числе декларативного типа, призванные упростить использование методов в задачах с большими данными.

4 ПРИМЕНЕНИЕ МЕТОДОВ МАШИННОГО ОБУЧЕНИЯ В ЗАДАЧЕ КЛАССИФИКАЦИИ ПОРОД НА УРАНОВЫХ МЕСТОРОЖДЕНИЯХ

4.1 Задача литологического расчленения скважин[6]

В процессе литологического расчленения скважины основную информацию о вскрываемых скважиной пластах несут каротажные диаграммы. Формально они представляют собой одномерные функции, которые устанавливают связь между глубиной и каким-либо измеряемым параметром пород (в зависимости от рассматриваемого типа каротажа). Каротажная диаграмма может быть представлена как функция от глубины скважины:

$$u = f(d) + \xi, \tag{4.1}$$

где u – значение метода каротажа;
d – глубина;
ξ – случайная компонента (шум или помеха).

Поскольку диаграмма представлена в цифровом виде с равномерным шагом квантования, то глубина может быть выражена через шаг квантования Δd и номер отсчета i:

$$d_i = i * \Delta d. \tag{4.2}$$

Шаг квантования Δd выбирается, исходя из условия минимальной потери точности при оцифровке. Согласно теореме Котельникова, любая непрерывная функция $f(t)$, спектр которой не содержит частот выше ω, может быть заменена совокупностью значений этой функции $f(t_i)$, если выполняется условие:

$$\Delta t = t_{i+1} - t_i \le \frac{1}{2\omega}. \tag{4.3}$$

Так как $\Delta t = \Delta h / v$, $\omega = v / h_{min}$, где v – скорость проведения каротажа, h_{min} – минимальная мощность прослоя, подлежащего выделению на кривой, получим:

$$\Delta d \le \frac{h_{min}}{2}. \tag{4.4}$$

[6] Формализация выполнена совместно с Я. И. Кучиным.

Погрешность интерпретации также равна Δd по причине того, что имеющиеся экспертные оценки, полученные на основании аналогового представления сигнала, не имеют шага квантования.

В случае, когда оцифровка каротажной кривой велась с неравномерным шагом квантования, ее можно представить таблично заданной функцией (u_i, d_i). Система неравномерного квантования применяется крайне редко и, как правило, по соответствующим формулам приводится к равномерной.

Заключения интерпретатора также возможно описать функциональной зависимостью от глубины скважины:

$$(y^1,...,y^k) = f(d), \tag{4.5}$$

где $y^1,...,y^k$ – результаты качественной интерпретации по различным типам литологических пластов ($y' = \{0,1\}$);
d – глубина.

В случае, когда интерпретация проводится по оцифрованным диаграммам, то и результаты имеют дискретный вид:

$$(y^1,...,y^k) = f(i*\Delta d), \tag{4.6}$$

где $y^1,...,y^k$ – результаты качественной интерпретации по различным типам литологических пластов;
Δd – шаг квантования по глубине;
i – номер отсчета глубины.

Таким образом, основная задача качественной интерпретации сводится к получению зависимостей между каротажными диаграммами и типами литологических пластов на указанной глубине скважины. В математической нотации:

$$(y^1,...,y^k) = f(u^1,\ldots,u^k), \tag{4.7}$$

где $y^1,...,y^k$ – результаты качественной интерпретации по различным типам литологических пластов и характера насыщения коллекторов ($y' = \{0,1\}$) для отсчета i;
$u^1,...,u^k$ – значения каротажных диаграмм для отсчета i.

Функцию связи между значениями каротажных методов и результатами интерпретации предлагается реализовать с использованием аппарата машинного обучения.

Однако применение методов машинного обучения к неподготовленным данным может быть неэффективным и не давать необходимых результатов. Следовательно, при их использовании необходимо подготовить исходные данные. То есть при интерпретации результатов ГИС используются не исходные каротажные диаграммы, а данные, прошедшие этап предварительной подготовки, который формально можно описать как:

$$(p^1,\ldots,p^r) = g(u^1,\ldots,u^k), \quad (4.8)$$

где $(p^1,\ldots,p^r)$ – вектор подготовленных данных для отсчета i;

g – функция подготовки данных, которая преобразует входной вектор $(u^1,\ldots,u^k)$ во входной вектор подготовленных данных $(p^1,\ldots,p^r)$;

$(u^1,\ldots,u^k)$ – вектор исходных данных для отсчета i.

Возможна ситуация, когда $r \neq k$, так как при подготовке данных могут использоваться методы, изменяющие размерность исходных данных (например, «окна данных»).

Исходя из этого, окончательное решение задачи качественной интерпретации сводится к получению зависимостей между подготовленными каротажными данными и типами литологических пластов на указанной глубине скважины. Это можно записать в следующем виде:

$$(y^1,\ldots,y^k) = f(p^1;\ldots,p^r). \quad (4.9)$$

Как было сказано выше, для реализации функции f предлагается использовать методы машинного обучения, в частности, искусственные нейронные сети. При этом входным вектором для указанных методов будет вектор $(p^1,\ldots,p^r)$, а выходным – $(y^1,\ldots,y^k)$. Важной задачей является поиск методов подготовки исходных данных и последовательности их применения.

4.2 Качество экспертной классификации

В силу особенностей классификации каротажных данных точная оценка качества классификации на основании объективных данных невозможна, поскольку получение точных данных о реальном распределении пород вдоль оси скважины затруднено. Керновое опробование делается не для всех скважин и не по всей глубине интерпретации. По этой причине система машинного обучения (классификации) опирается на данные экспертного оценивания. То есть система классификации обучается в основном на базе данных экспертов. При этом, однако, возникает проблема оценки качества самой экспертной интерпретации. Для анализа возможностей алгоритмов машинного обучения можно синтезировать некоторые данные, соответствующие по тем или иным параметрам реальным данным каротажа. Используя эти синтезированные данные, удовлетворяющие некоторым базовым физическим принципам работы каротажных приборов и грунтов, можно проверить качество системы машинного обучения. Разумеется, такая оценка не претендует на исчерпывающую полноту, но в то же время может дать представление о предельных возможностях алгоритмов и некоторое, косвенное, представление об экспертных оценках. С другой стороны, еще один путь, позволяющий оценить границы

противоречивости экспертного оценивания, заключается в сравнении экспертных оценок нескольких экспертов.

4.2.1 Синтезированная (искусственная) скважина[7]

Единственным способом прямого и достоверного определения литологического состава пород вдоль ствола скважины является отбор керна. Однако и он не дает 100% достоверной информации, поскольку процент извлечения, как правило, не превышает 80% для осадочных пород, характерных для урановых месторождений Казахстана. Кроме того, погрешности возникают при привязке керна по глубине по данным каротажа, а также при самом описании керна. Эти погрешности трудно определить и учесть. Ввиду отсутствия достоверных сведений о литологическом строении разреза ствола скважины, когда и данные электрического каротажа, и отбор керна дают лишь приблизительные значения, представляется перспективным моделирование записанного сигнала каротажа при заданном распределении пород вдоль ствола скважины с известными физическими свойствами. Это позволит получить набор пусть искусственных, но полностью определенных значений каротажа, четко привязанных к определенным породам.

Физические свойства каждой породы, в частности кажущееся сопротивление, лежат в определенном диапазоне. Распределение внутри диапазона может быть изучено на этапе разведки и в лабораторных исследованиях.

Задав распределение мощности пластов, для каждого 10-сантиметрового пропластка внутри пласта выбирается кажущееся сопротивление из заданного диапазона с соответствующим распределением, что позволяет получить кривую распределения кажущегося сопротивления вдоль ствола скважины. Для моделирования зарегистрированной кривой КС необходимо также учитывать параметры скважинного прибора (тип зонда, расстояние между электродами), диаметр скважины, свойства бурового раствора и др. В итоге можно получить смоделированную запись каротажа сопротивлений, которая соответствует заданному распределению пород, при этом достоверность информации о распределении пород будет известна со 100% точностью. Аналогичным образом моделируется и кривая ПС. Это дает возможность проверить работу различных алгоритмов машинного обучения и улучшить их. Максимальное приближение этой модели к реальным условиям позволит повысить качество распознавания на реальных данных и определить верхний предел точности распознавания.

[7] Метод предложен Я. И. Кучиным, реализован К. О. Якуниным.

Для реализации описанной модели разработана программа генерации каротажных данных, позволяющая сгенерировать данные произвольного количества синтетических (искусственных) скважин. Полученные данные, в свою очередь, «обрабатываются» моделью скважинного прибора длиной 1 метр.

В графическом виде результаты генерации данных КС и данных, «полученных» моделью скважинного прибора, приведены на рисунке 4.1.

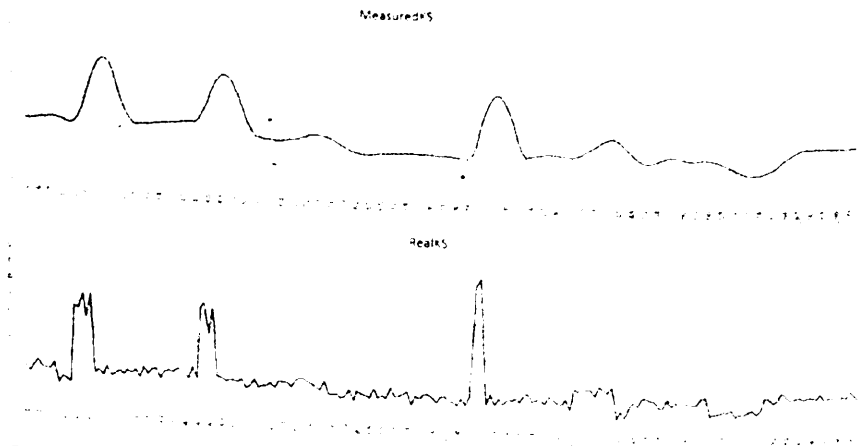


Рисунок 4.1. Генерируемые данные КС (внизу) и результат их «измерения» (сверху)

Видно, что в результате использования прибора пласт низкой мощности с высоким показателем КС при «измерении» имеет значительно меньший показатель из-за усреднения.

Описание и листинг программы генерации каротажных данных приведены в приложении 7 «Программа генерации каротажных данных» и приложении 8 «Листинг программы генерации каротажных данных».

4.2.2 Результаты применения алгоритмов МО к данным синтезированной скважины

Синтезированные по указанным выше правилам данные были обработаны и проинтерпретированы с помощью искусственной нейронной сети (таблица 4.1) и алгоритма *k-NN*. Кривая обучения показана на рисунке 4.2.

Алгоритм *k-NN* демонстрирует несколько худшие результаты обучаемости (рисунки 4.3 и 4.4).

Результаты свидетельствуют о том, что искусственные нейронные сети способны показать высокий результат классификации на данных

синтезированной скважины. Соответственно, синтезированная скважина упрощает выявление различий в алгоритмах классификации.

Таблица 4.1. Результаты работы алгоритма искусственной нейронной сети на данных синтезированных скважин

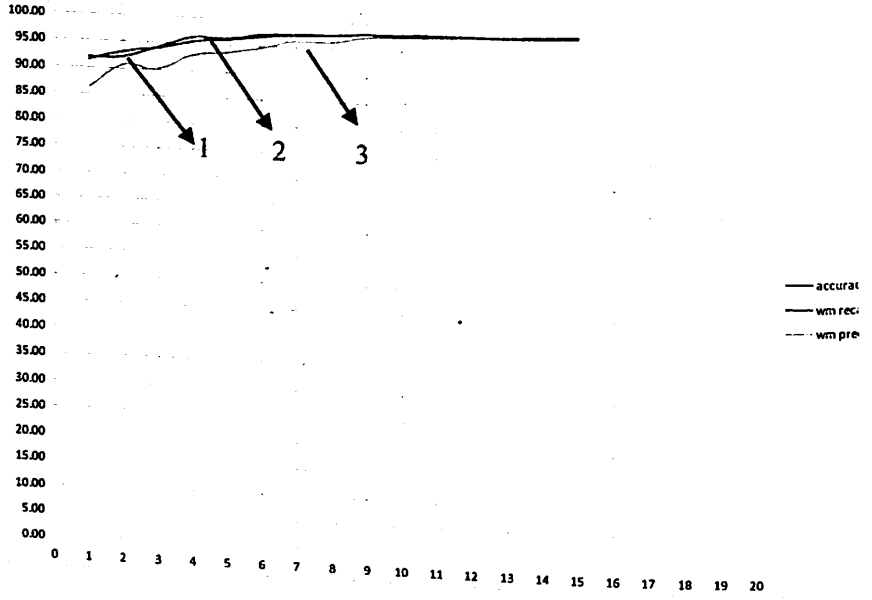
Условный номер скважины	Accuracy	Kappa	WM recall	WM precision	T1 Score
1	0.9122	0.889	0.9174	0.8605	0.888039
2	0.9295	0.911	0.9196	0.9053	0.912394
3	0.9391	0.923	0.9411	0.8993	0.919725
4	0.9545	0.942	0.9621	0.9281	0.944794
5	0.9591	0.949	0.9625	0.9355	0.948808
6	0.9678	0.96	0.9713	0.9458	0.95838
7	0.974	0.967	0.9733	0.9621	0.967668
8	0.9748	0.968	0.9761	0.9623	0.969151
9	0.9789	0.973	0.9788	0.9733	0.976042
10	0.9792	0.974	0.9783	0.9744	0.976346
11	0.9801	0.975	0.9775	0.9762	0.97685
12	0.9809	0.976	0.9812	0.9781	0.979648
Average	0.960841667	0.950583	0.9616	0.941741667	0.951487

4.2.3 Сравнение экспертных оценок

Противоречивость экспертных оценок вносит дополнительные трудности в процесс обучения системы МО. Несмотря на то, что именно экспертные оценки используются для обучения, эксперты проводят классификацию по-разному. Для сравнительного анализа качества экспертных оценок были выбраны три скважины с номерами 2100, 2104 и 4939. Данные каротажа указанных скважин были переданы трем экспертам, условно обозначенным буквами D, L и T. Кроме этого, для скважин 2100 и 4939 известны данные кернового опробования *(kern)*. Основываясь на полученных от экспертов данных литологического расчленения и данных кернового опробования, был проведен расчет основных показателей качества *(accuracy, recall, precision, Kappa)* при попарном сравнении, когда данные одного из экспертов принимались за эталон, а данные второго эксперта с ними сравнивались (таблица 4.2).

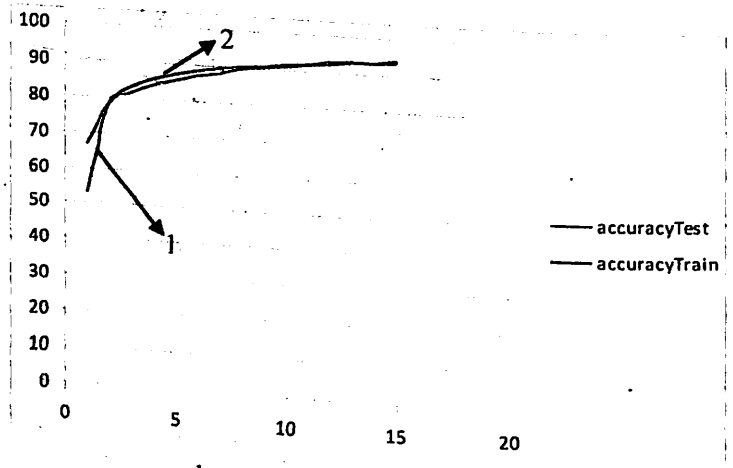
Видно, что среднее значение точности для экспертов составляет *accuracy* = 0.67, а усредняющий показатель разброса *T1 Score* = 0.6.

Сравнивая оценки экспертов с данными кернового опробования, получаем цифры *accuracy* = 0.5, а *T1 Score* = 0.27.



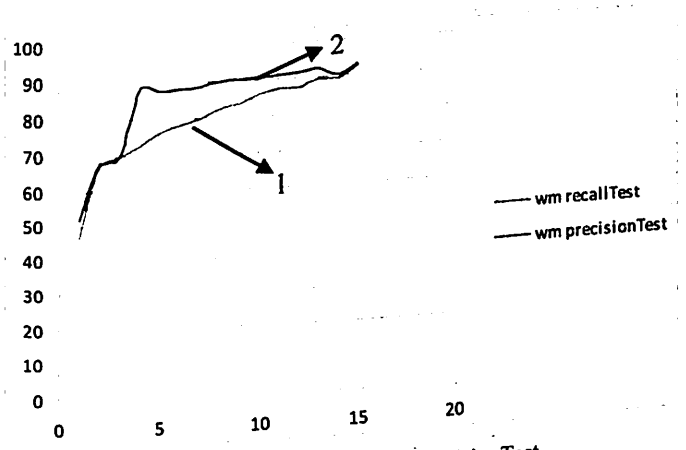
1 – accuracy, 2 – wm recall, 3 – wm precision

Рисунок 4.2. Кривая обучения нейросетевого классификатора для случая синтезированной скважины



1 – accuracyTest, 2 – accuracyTrain

Рисунок 4.3. Изменение показателя accuracy при увеличении числа обучающих примеров синтезированной скважины для алгоритма k-NN



1 – wm recallTest, 2 – wm precisionTest

Рисунок 4.4. Изменение показателей recall и precision при увеличении числа обучающих примеров синтезированной скважины для алгоритма k-NN

В то же время, учитывая субъективность экспертов, можно отметить, что по скважине 2100 и 2104 эксперты D и L больше всего согласны между собой (*accuracy* > 0.8). Со значениями усредненного керна больше всего совпадает «мнение» эксперта D *(accuracy* = 0.69). По скважине 4939 эксперты D и T больше всего согласны между собой (*accuracy* = 0.77). С результатами усредненного керна больше всего совпадают оценки эксперта L *(accuracy* = 0.60).

При сравнениях экспертов с данными кернового опробования показатели качества существенно ниже, чем только при сравнении экспертов (таблица 4.3).

Исходные данные для расчета показателей качества приведены в приложении 2 «Сравнительная оценка точности распознавания экспертами». Анализируя данные указанного приложения, можно видеть, что для некоторых критичных пород (глина) мнения экспертов совпадают в 70–90% случаев, однако значительно хуже согласуются с данными кернового опробования.

4.3 Экспериментальная оценка методов предварительной обработки данных

Предварительная обработка данных часто является ключевым элементом, способствующим получению хороших результатов при применении методов МО. Для выбора и настройки методов предобработки были проведены вычислительные эксперименты, которые оценивали влияние отдельных методов (видов нормировки, сглаживания,

сдвига кривых, удаления аномальных значений, формирования плавающего окна данных) на результаты литологического расчленения. При этом мы придерживались гипотезы о независимости результатов указанных методов друг от друга. То есть для каждого метода отрабатывался наилучший подход в порядке, указанном в приложении 3 «Описание методики проведения экспериментов и применяемого программного обеспечения».

Таблица 4.2. Попарное сравнение

Пары экспертов	Accuracy	Kappa	Recall	Precision	T1 Score
2100					
D vs L	0.81	0.70	0.63	0.61	0.618018
D vs T	0.71	0.54	0.42	0.49	0.455325
L vs T	0.80	0.67	0.46	0.52	0.493058
4939					
D vs L	0.3317	0.16	0.5979	0.4975	0.543099
D vs T	0.7706	0.66	0.7941	0.6879	0.737195
L vs T	0.3762	0.19	0.6086	0.5495	0.577542
2104					
D vs L	0.8409	0.76	0.8445	0.8205	0.832327
D vs T	0.6551	0.49	0.5493	0.5424	0.545828
L vs T	0.7213	0.57	0.5845	0.6011	0.592684
Average experts	0.67	0.53	0.61	0.59	0.60
2100					
Kern vs D	0.693	0.39	0.3624	0.3487	0.355418
Kern vs L	0.6444	0.23	0.2775	0.2423	0.258708
Kern vs T	0.6505	0.21	0.274	0.2441	0.258187
4939					
Kern vs D	0.1749	0.04	0.2011	0.2295	0.214363
Kern vs L	0.6089	0.37	0.3066	0.3986	0.346599
Kern vs T	0.2096	0.04	0.2004	0.2231	0.211142
Average kern	0.50	0.21	0.27	0.28	0.27

После предварительных исследований, включающих около 2000 вычислительных экспериментов, были отработаны и экспериментально подтверждены следующие этапы предварительной обработки данных каротажа [118] [119] [120] [121] [122]:

Таблица 4.3. Средние значения качественных показателей

	Accuracy	Kappa	Recall	Precision	T1 Score
Experts	0.67	0.53	0.61	0.59	0.60
Kern vs experts	0.50	0.21	0.27	0.28	0.27

- Удаление аномальных значений («закисления» и «выбросов»), очистка выборки от шумовых точек.
- Нормировка.
- Вейвлет-сглаживание.
- «Переворот» ПС (связано с особенностями физического процесса получения данных).
- Исключение сдвига каротажных кривых.
- Очистка данных по методу «ближайших соседей».
- Формирование плавающего окна данных.

По итогам экспериментов для использования выбраны следующие из указанных пунктов:

1. Удаление аномальных значений.
2. Линейная нормировка.
3. Очистка данных по методу «ближайших соседей» (в некоторых экспериментах).
4. Формирование плавающего окна данных.

Обработанные таким образом данные использовались в дальнейшем для обучения. Методы препроцессинга данных для распознавания включают практически те же самые этапы, за исключением пункта 4, выполнение которого в данном случае может привести к потере данных.

Перечисленные этапы предобработки использовались в экспериментах не только с ИНС, но и с другими методами МО.

4.3.1 Нормировка

Одним из важнейших методов предварительной обработки данных является нормировка. Нормировка необходима для нивелирования слишком большого разброса во входных данных. Очевидно, что большая разница в значениях входных сигналов приводит к потере информативности, так как слабый сигнал становится незаметен на фоне сильного. Для выравнивания влияния сигналов разной амплитуды применяется нормировка. Нормирование сигнала может быть линейным

(сжатие диапазона) и нелинейным (нормировка сигмоидой, гиперболическим тангенсом и т.п.).

Нелинейная нормировка (сигмоида и гиперболический тангенс) отрицательно повлияла на качество распознавания (таблица 4.4, таблица 4.5). Причем это верно для сигмоид и гиперболического тангенса с разными коэффициентами при x.

Линейная нормировка, напротив, значительно (на 15–20%) улучшила качество распознавания (рисунки 4.5, 4.6). В ходе экспериментов была выявлена проблема с разным уровнем записи – на разных скважинах диапазоны одних и тех же показателей могут значительно отличаться.

Таблица 4.4. Использование сигмоиды для нормировки сигнала

Анализ влияния метода нормировки на качество распознавания				
Количество скважин	Коэффициент при сигмоиде	Номер «сырой»[8]	CCR по Validation	CCR по «сырой»
8 (не обработанные сигмоидой)		9	62%	67%
8 (обработанные сигмоидой)	1	9	67%	53%
8 (не обработанные сигмоидой)		7	65%	47%
8 (обработанные сигмоидой)	1	7	60%	46%
8 (не обработанные сигмоидой)		9	62%	67%
8 (обработанные сигмоидой)	1,5	9	67%	48%
8 (не обработанные сигмоидой)		7	67%	48%
8 (обработанные сигмоидой)	1,5	7	55%	55%
8 (не обработанные сигмоидой)		9	62%	65%
8 (обработанные сигмоидой)	2	9	67%	50%
8 (не обработанные сигмоидой)		7	65%	45%
8 (обработанные сигмоидой)	2	7	60%	42%

[8] «Сырая» скважина не участвует в обучении, а используется для оценки результатов.

Таблица 4.5. Использование гиперболического тангенса для нормировки сигнала

Анализ влияния метода нормировки на качество распознавания				
Количество скважин	Коэффициент при сигмоиде	Номер «сырой»	CCR по Validation	CCR по «сырой»
8 (не обработанные гип. тангенсом)		9	71%	59%
8 (обработанные гип. тангенсом)	0,8	9	71%	54%
8 (не обработанные гип. тангенсом)		7	64%	47%
8 (обработанные гип. тангенсом)	0,8	7	65%	43%
8 (не обработанные гип. тангенсом)		9	71%	59%
8 (обработанные гип. тангенсом)	1	9	67%	66%
8 (не обработанные гип. тангенсом)		7	64%	47%
8 (обработанные гип. тангенсом)	1	7	60%	46%
8 (не обработанные гип. тангенсом)		9	71%	59%
8 (обработанные гип. тангенсом)	1,2	9	68%	49%
8 (не обработанные гип. тангенсом)		7	64%	47%
8 (обработанные гип. тангенсом)	1,2	7	57%	46%

Таким образом, линейная нормировка решает эту проблему. Однако важно проводить линейную нормировку каждой скважины отдельно, иначе при нормировке всей ОВ качество распознавания не улучшится.

4.3.2 Сглаживание

Кроме нормировки данных для повышения качества распознавания могут быть использованы различные методы сглаживания, позволяющие устранить шум и иные неинформативные составляющие данных. В

качестве методов сглаживания рассматривались два вида дискретных преобразований: анализ Фурье и вейвлет-преобразование.

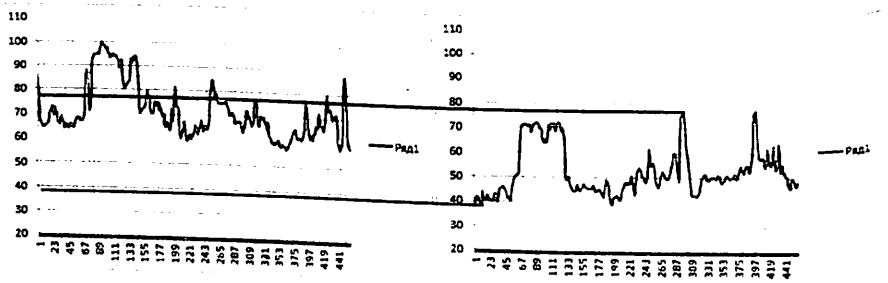


Рисунок 4.5. Графики КС по двум разным скважинам

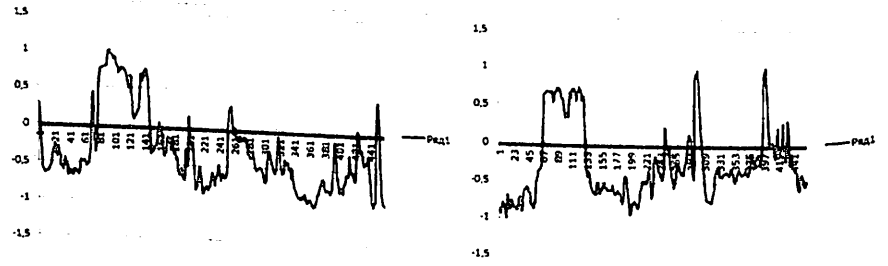


Рисунок 4.6. Графики КС по двум разным скважинам после нормировки

Преобразование Фурье [3] раскладывает на гармоники весь сигнал, и фильтрация хорошо подходит для стационарных шумов, например Гауссовского. Суть преобразования Фурье – разложение сигнала на ряд гармонических функций (синусоид или косинусоид). Причем если полезная информация о сигнале содержится в низкочастотных гармониках, а шум в высокочастотном диапазоне, то такой подход позволяет легко удалить одну из составляющих, например высокочастотную, а с ней вместе и шум. Таким образом, процесс сглаживания происходит следующим образом: сигнал раскладывается на гармоники, выбирается определенная частота, и функции с превышением данного показателя считаются шумом и удаляются из общего сигнала.

В каротажных данных наличие высокочастотных шумов маловероятно. Снятие данных происходит с заданным интервалом. Устранение высокочастотной составляющей приведет к потере информативности сигнала (мелкие флуктуации пород перестанут быть заметны). Вследствие этого на данном этапе было принято решение не использовать преобразование Фурье для сглаживания данных. Однако в каротажных данных возможны низкочастотные шумы, связанные с

изменением амплитуды сигнала из-за закисления пород. Этот момент требует дополнительного исследования. Возможно, преобразование Фурье может быть полезным на закисленных скважинах. Закисление (и искажение сигнала) происходит неравномерно, плавно изменяясь вместе с изменением концентрации кислоты. То есть закисление – это фактически низкочастотный шум, и фильтрация Фурье может удалить этот шум из сигнала.

Вейвлет-преобразование локализовано по времени, т.е. позволяет фильтровать данные отдельно на небольших участках, иными словами, может применяться для фильтрования нестационарных шумов, таких как шум от работы двигателя, нестабильность электромагнитного поля и т.п. Суть вейвлет-разложения схожа с Фурье, однако вместо гармонических функций используются небольшие, локализованные по времени «маленькие волны» (буквальный перевод слова *wavelet*). Сам же метод вейвлет-преобразования позволяет рассматривать и анализировать сигнал на разных масштабах. Сглаживание представляет собой нахождение матрицы коэффициентов разложения (числа, на которые нужно умножить вейвлет, параллельно перенесенный на определенную точку пространства), выбор порогового значения шума, обнуление всех коэффициентов, меньших, чем пороговое значение, и восстановление сигнала по оставшимся ненулевым коэффициентам. Существует несколько видов вейвлет-функций: вейвлет Хаара, вейвлеты Добеши, вейвлеты Гаусса, вейвлет Мейера и др. В процессе исследования был выбран вейвлет Добеши. Данное вейвлет-преобразование выбрано по двум причинам: это дискретные, а не непрерывные вейвлеты (а мы имеем дело с дискретными представлениями сигналов), и это не один вейвлет, а семейство, то есть имеется возможность подстраивать порядок вейвлета для достижения лучшего результата. В процессе экспериментов использовалось три (глубина разложения = масштаб) варианта параметров вейвлет-сглаживания (порядок вейвлета Добеши и глубина разложения/масштаб) (таблица 4.6).

Таблица 4.6. Влияние вейвлет-сглаживания сигнала на качество распознавания

Вейвлет (порядок вейвлета – масштаб)	CCR по множеству Validation	% по «сырой»
1-1	47,4	60,33
1-3	41,8	34,22
7-2	46,2	57,66
Без сглаживания	48,8	58

Как видно, более мягкие параметры фильтрации с вейвлетом более высокого порядка (а значит, с большим количеством дискретных значений, то есть более «гладкого») значительно результат не изменили. Однако сглаживание с параметрами 1-3 (рисунок 4.7), приводившее к значительным изменениям сигнала, результаты явно и значительно ухудшило.

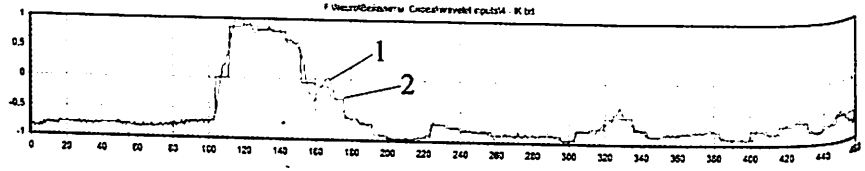


Рисунок 4.7. Сглаженный сигнал с параметрами 1-3. Красный сигнал (1) – необработанный, зеленый (2) – сглаженный с параметрами 1-3

Полученные результаты позволяют утверждать, что значительных улучшений в качестве распознавания при применении вейвлет-сглаживания нет. В принципе, сеть не замечает мелких флуктуаций, и небольшое отклонение (1–2% от диапазона, примерно такого порядка шумы сглаживаются вейвлетом) сигнала вверх или вниз на качество распознавания может теоретически повлиять даже положительно (хотя проведенные опыты показывают, что результаты распознавания сглаженных и оригинальных сигналов практически идентичны).

Таким образом, фильтрация незначительных шумов не дает результатов, и применять ее для улучшения качества обучения нецелесообразно. Однако вейвлет-сглаживание может быть применено, например, для упрощения распознавания экстремумов КС, информация о которых необходима при разделении входных данных на категории (глава «Возможные варианты подачи данных на входы НС»). Геологи-эксперты при интерпретации данных каротажа делят сигналы на категории, основываясь в основном на экстремумах КС. Однако мелкие шумы усложняют автоматизацию этого процесса, так как приходится отличать мелкие экстремумы от шумов и реальные, значимые экстремумы КС. Вейвлет-сглаживание может решить эту проблему, сгладив незначительные экстремумы, так что автоматизация этого процесса будет значительно упрощена.

4.3.3 Исключение сдвига кривых каротажа относительно друг друга

Фактором, влияющим на качество предварительной обработки данных, является сдвиг кривой ИК относительно кривых ПС и КС. КС и ПС снимаются одним прибором, а ИК – другим. Вследствие этого из-за различных факторов (неточность начальной установки, усадка почвы и т.п.) кривая ИК может быть смещена по глубине относительно кривых ПС и КС. Данные ИК могут быть сопоставлены с другими показателями глубины. Обычно это смещение невелико и постоянно. Например, в ряде случаев ИК смещается на величину до 0,5 метра в большую или меньшую сторону. Для исключения влияния данного фактора разработаны специальные программы *(corrGis, corrPy)*, которые позволяют находить сдвиг кривых на основе корреляционных методов и корректировать данные во избежание сдвига.

Анализ сдвига кривых относительно друг друга и исключение его должны были либо повысить качество распознавания, либо оставить его на прежнем уровне. Второе предположение связано с тем, что при обучении сети она теоретически способна самостоятельно распознать сдвиг при условии подачи на вход плавающего окна данных. В ходе экспериментов подавалось плавающее окно с параметрами 5+1+5 (по пять точек выше и ниже текущей плюс текущая точка). Выяснилось, что сдвиг влияет на результаты распознавания крайне незначительно (не более 3%). Результаты экспериментов приведены в таблице 4.7.

Отметим, что когда для проверки гипотезы значения сдвига были искусственно изменены на противоположные (*-1), то результат не изменился.

Это позволяет сделать вывод, что при подаче на вход сети данных в виде плавающего окна с параметрами 5+1+5 небольшой сдвиг (до 10 точек) кривой ИК относительно кривых ПС и КС не влияет на результаты распознавания.

4.3.4 Очистка от шума

Шумом часто называются объекты на границе классов, которые бесполезны для формирования обучающего множества, так как «смазывают» границы классов. Для очистки обучающей выборки от шума применялся алгоритм *k-NN*. Применение алгоритма позволяет значительно уменьшить объем обучающей выборки (рисунки 4.8, 4.9).

Рисунок 4.8 показывает в псевдотрехмерном виде исходный обучающий набор в пространстве признаков ИК, КС, ПС. Как видно, границы классов размыты и пересекаются. Отмечен также большой

разброс значений – часто из-за инерционности процесса измерения и размеров скважинного прибора.

4.3.5 Дополнительные параметры

Глубина. Геофизики-эксперты используют данные эталонного разреза, по которым они сверяют свои результаты. В предположении, что сеть сможет использовать параметр глубины как эталонную скважину, были проведены эксперименты для исследования влияния глубины как дополнительного параметра. Результаты экспериментов приведены в таблице 4.8. Как видно, отличия в распознавании данных как «сырой» скважины, так и множества *Validation* составляют около процента, что не дает основания делать вывод о том, что параметр глубины существенно влияет на качество распознавания.

Таблица 4.7. Анализ влияния корректировки сдвига данных ИК относительно ПС и КС на качество распознавания

Количество скважин	Тип корреляции	Номер «сырой» скважины	CCR по Validation	CCR по «сырой»
8 (n.c.)	сдвиг ИК по КС	9	62%	67%
8 (corrected)	сдвиг ИК по КС	9	59%	62%
8 (n.c.)	сдвиг ИК по КС	7	65%	45%
8 (corrected)	сдвиг ИК по КС	7	60%	43%
8 (n.c.)	сдвиг ИК и ПС по КС	9	62%	67%
8 (corrected)	сдвиг ИК и ПС по КС	9	67%	68%
8 (n.c.)	сдвиг ИК и ПС по КС	7	65%	45%
8 (corrected)	сдвиг ИК и ПС по КС	7	68%	48%
8 (n.c.)	сдвиг ИК и ПС по КС	6	63%	68%
8 (corrected)	сдвиг ИК и ПС по КС	6	63%	68%

Примечания к таблице:

1. *(corrected) – данные, откорректированные по максимуму взаимнокорреляционной функции.*
2. *(n.c.) – данные со сдвигом кривой ИК относительно кривых ПС и КС (не откорректированные).*
3. *Для работы использовалась архитектура сети с двумя скрытыми слоями [33-39-33-8].*
4. *Обучение проводилось до получения ошибки обучения (training error) = 90%*

4.3.6 Плавающее окно данных

В процессе экспериментов проверялось предположение о влиянии размера плавающего окна данных. Плавающее окно данных – метод, позволяющий вводить на вход программы МО не отдельные «точечные» данные, а их последовательности, дающие возможность учитывать в большей или меньшей мере закономерности изменения данных по времени или, как в нашем случае, по глубине [123]. Тогда на вход ИНС или иного алгоритма МО подается не три значения (ИК, ПС и КС), а сразу несколько таких «троек». Например, если плавающее окно данных состоит из 11 показателей, расположенных последовательно по глубине (плавающее окно с параметрами 5+1+5 – пять точек выше текущей и 5 ниже), то в этом случае на вход системы МО подается 33 значения. Следующее подаваемое на вход значение получается сдвигом окна по глубине на одну позицию (точку) и подачей части предыдущих значений вместе с одним новым. В экспериментах оценивалось качество распознавания при подаче плавающих окон 3+1+3, 5+1+ 5, 10+1+10. В большинстве экспериментов лучшие результаты (1–5% улучшения показателя *accuracy)* получались при подаче плавающего окна данных размером 5+1+5. Данный размер был принят за основной с учетом также и физических параметров каротажного прибора (размер около 1 метра), который как раз охватывает зону в 10–11 интервалов (точек) по глубине (показания снимаются через каждые 10 сантиметров).

4.4 Применение ИНС для классификации данных каротажа

4.4.1 Общие замечания, применяемые программные средства и методы оценки качества обучения сети

Проведение вычислительных экспериментов основано на использовании пакета *Alyuda NeuroIntelligence* и *RapidMiner*. Пакет компании *Alyuda Research* [124] позволяет разработать необходимую структуру сети, обучить ее с применением корректного, при необходимости автоматического разбиения обучающей выборки на группы (обучающая, проверочная (тестовая) и поверочная – *Training, Validation* и *Testing)*. В процессе обучения возможно применение набора базовых алгоритмов обучения. Пакет позволяет также визуализировать процесс обучения и поверки результата.

Пакет имеет возможность изменять архитектуру сети с поддержкой нескольких скрытых слоев. Пакет позволяет изменять функции активации нейронов (линейная, сигмоида и гиперболический тангенс). Пакет также содержит ряд функций для оценки качества обучения сети и выявления слабых сторон сети: таблицы с значениями сигналов на выходных

нейронах, таблицы и графики, показывающие количество верных/неверных ответов по отдельным породам.

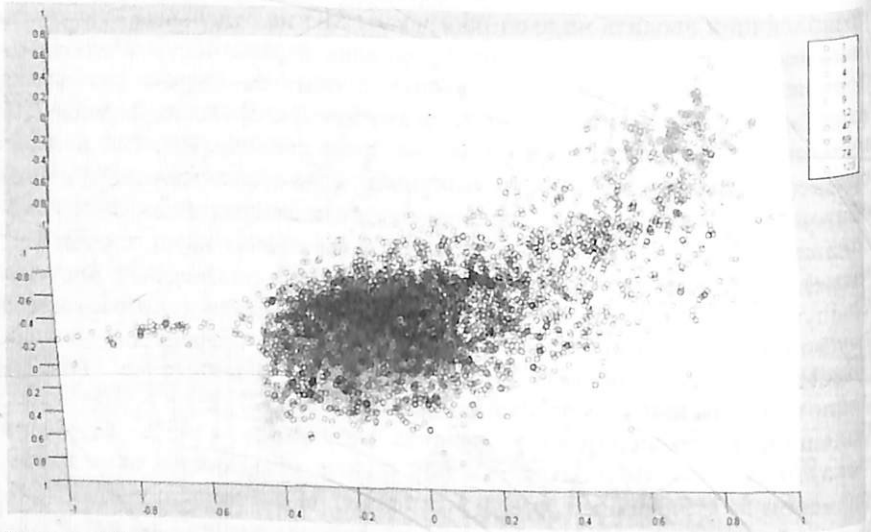


Рисунок 4.8. Ответы экспертов (обучающее множество) в пространстве признаков ИК, КС, ПС

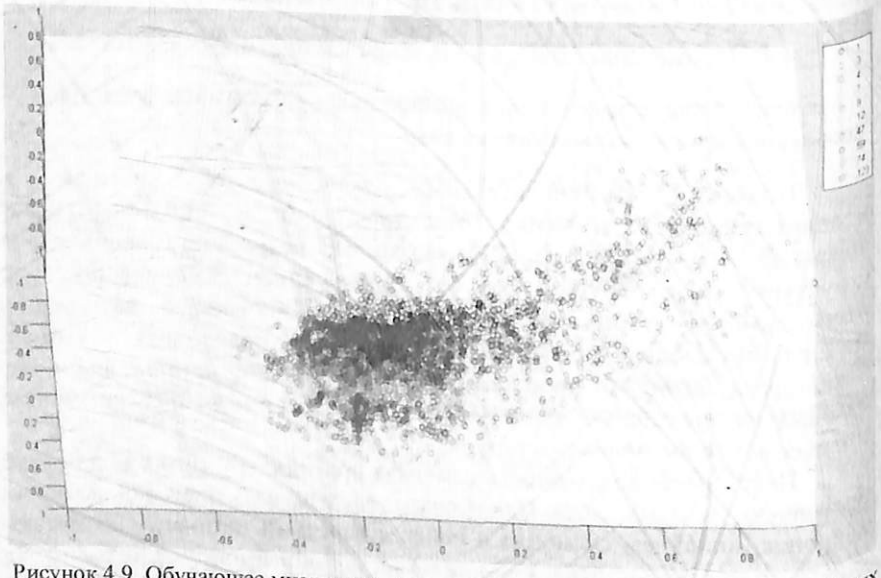


Рисунок 4.9. Обучающее множество, очищенное с помощью алгоритма k ближайших соседей (k-NN)

Таблица 4.8. Анализ влияния глубины как дополнительного параметра на качество распознавания

Количество скважин	№ «сырой»	CCR по Validation	CCR по «сырой»	ID опыта
18 (w.depth)	1	60,8	59,11	1
18	1	62	60	2
18 (w.depth)	2	58,2	70	3
18	2	60	73	4
18 (w.depth)	3	61,7	77,77	5
18	3	62	79	6
18 (w.depth)	4	54	44	7
18	4	60	41	8
18 (w.depth)	5	61	69	9
18	5	62	69,1	10
18 (w.depth)	6	62	71,2	11
18	6	62	71	12
18 (w.depth)	7	62	49	13
18	7	63	51	14
18 (w.depth)	8	59,8	66,4	15
18	8	60	68,5	16
18 (w.depth)	9	59,9	64,22	17
18	9	62	62,3	18

Примечания:

1. (w.depth) – *данные с дополнительным параметром (глубиной).*
2. *Для работы использовалась архитектура сети с двумя скрытыми слоями [33-39-33-10].*
3. *Обучение проводилось до получения ошибки обучения (training error) = 90%.*

Для обработки сигналов применялось программное обеспечение *Excel* из пакета *Microsoft Office*. *Excel* использован для написания макросов на языке *Basic*. Для ускорения работы были разработаны шаблоны, необходимые для обработки сигналов, а также для создания матриц ответов (ответы сети и ответы экспертов). Перечень шаблонов включает:

1. Шаблон для подсчета процента правильных результатов по «сырой» скважине.
2. Шаблон, формирующий матрицу ответов сети/ответов эксперта.
3. Шаблон, выполняющий линейную нормировку (т.е. сжатие диапазона) с возможностью изменять диапазон для сжатия.
4. Шаблоны для обработки сигналов сигмоидой и гиперболическим тангенсом с возможностью изменения коэффициента при x.
5. Шаблон, выполняющий нормировку дисперсионным методом.

Приложения на языке *Python* разработаны для упрощения формирования обучающей выборки с выделением «сырой» скважины (той, которая не участвует в обучении, а используется для оценки результатов), формирования «идеальной скважины» (набора данных, составленных из данных разных скважин), определения и коррекции сдвига данных ИК и др.

Для реализации вейвлет-сглаживания использовалась библиотека *WaveUtils*.

В приложении 3 «Описание методики проведения экспериментов и применяемого программного обеспечения» упомянуты применяемые программные средства, обеспечивающие проведение экспериментов.

В процессе обучения система самостоятельно проводит поверку качества обучения с выдачей процентного результата в виде *CCR* (процент правильных ответов сети). *CCR* на множестве *Training* не может быть использован в качестве критерия качества обучения, так как это множество участвует в обучении сети. Более показательна оценка *CCR* на множестве *Validation*, которое не участвует в обучении. Однако в реальных ситуациях необходимо выполнять распознавание данных новых скважин. Поэтому в качестве наиболее важного критерия качества обучения в экспериментах была выбрана оценка *CCR*, которая получается при использовании обученной сети для распознавания данных не участвовавших в обучении (так называемых «сырых») скважин, которые образуют тестовое множество. Данная оценка позволяет наиболее объективно оценить качество архитектуры нейронной сети, алгоритма обучения, нормировки, сглаживания и т.п. методов предобработки данных. В ходе экспериментов все данные, включая и все три множества, *Training*, *Validation*, *Testing*, и данные «сырой» скважины, обрабатывались абсолютно одинаково, то есть к ним применялись одинаковые методы нормировки, сглаживания и т.п., отработка которых и являлась одной из целей экспериментов.

В общей сложности было проведено порядка 2 тысяч экспериментов. Из них около 500 – предварительные эксперименты, выполненные до отработки описанной ниже методики проведения

экспериментов. Результаты экспериментов критически анализировались, и необходимые изменения вносились в методику. Таким образом была получена методика, которая дает стабильные, достаточно обоснованные результаты, но требует проведения большого объема опытов.

Наиболее масштабные серии опытов состояли из 200–300 экспериментов (9 «сырых» скважин, 15–30 и более экспериментов на каждую).

Среди проведенных экспериментов: тестирования влияния определенных преобразований на качество распознавания ИНС, исследование влияния архитектуры и других параметров ИНС на качество распознавания, исследование алгоритмов обучения и т.п.

4.4.2 Алгоритм обучения нейронной сети

Пакет *Alyuda NeuroIntelligence* предлагает несколько алгоритмов обучения нейронной сети: *Quick Propagation, Conjugate Gradient Descent, Quasi-Newton, Limited Memory Quasi-Newton, Levenberg-Marquardt, Online Back Propagation, Batch Back Propagation.* Различие между алгоритмами заключается в скорости обучения, сходимости результата (некоторые алгоритмы, например *Leven-M,* «зависают»), разбросе результатов от запуска к запуску (разнице между минимальным и максимальным значениями). Для выбора наиболее подходящего алгоритма обучения были проведены эксперименты, результаты которых представлены в таблице 4.9.

Как видно из таблицы 4.10, алгоритм *Conjugate Gradient Descent* дает в среднем близкие к лучшему результаты и в то же время обладает минимальным разбросом и дисперсией. По этой причине он был выбран в качестве базового для проведения всех экспериментов.

Алгоритм *Conjugate Gradient Descent* (градиентный спуск) представляет собой классический метод обратного распространения ошибки, в котором коэффициент скорости обучения изменяется динамически (меняется с каждой итерацией), что обеспечивает хорошую скорость обучения и сходимость. В частности, экспериментально установлено, что данный алгоритм, в отличие от *Levenberg-Marquardt,* стабильно работает с сетями с большим (1500 и более) количеством связей (в то время как *Levenberg-Marquardt* на таких сетях зависает на первой итерации, даже при использовании мощного компьютера). Алгоритм градиентного спуска позволяет достигнуть максимального результата в среднем за 40–60 итераций, в то время как *Online Batch* достигает максимума лишь после 100 и более итераций.

Таблица 4.9. Оценка качества алгоритма обучения нейронной сети

№ эксперимента	Quick	Conjugate Gradient Descent	Quasi-Newton	Levenberg-Marquardt	Online Batch
1	56.8	59.3	60.1	N/A	49.7
2	56.4	57.3	59.6		50.5
3	52.6	59.1	57.3		47.6
4	59.1	57	61		49.1
5	51.5	56.8	58.1		53.7
6	51.9	59.5	60.2		54.4
7	61.8	60.1	54.8		55.6
8	52.6	57.6	58.2		49.9
9	45.7	57.8	58.7		48.3
10	50.6	61.2	62.6		49

Примечание. В таблицу записан лучший CCR по множеству Validation после обучения сети до 90% ошибки обучения (training error).

Таблица 4.10. Показатели качества алгоритмов обучения

Показатель	Quick	Conjugate Gradient Descent	Quasi-Newton	Levenberg-Marquardt	Online Batch
Разброс	16.1	4.4	7.8	0	8
Среднее	53.9	58.6	59.1	нет результата	50.8
Дисперсия	195.38	19.681	42.004	нет результата	69.136

4.4.3 Архитектура нейронной сети

Архитектура нейронной сети – это количество нейронов на слоях и количество этих слоев. Причем оптимальное количество нейронов на скрытом слое/слоях зависит от количества входных и выходных нейронов. Как слишком большое, так и слишком малое количество весов может ухудшить результаты распознавания. К сожалению, конкретной формулы, теории или метода для определения оптимального количества нейронов не существует, то есть единственный точный метод – перебор и проверка различных вариантов [5]. Однако при сколько-нибудь пригодном для практики числе нейронов получение точного решения становится невозможным из-за комбинаторного взрыва при поиске решения. Поэтому для практического применения разработаны

эвристические методы. Пакет *Alyuda* позволяет производить поиск сети, однако результат этого поиска нельзя назвать абсолютно точным показателем качества архитектуры не только вследствие применения эвристических методов, но также потому, что во внимание принимается *CCR* по *Validation,* а не по «сырой» скважине». Тем не менее для поиска оптимальной архитектуры применялся пакет *Alyuda,* который выполняет подбор архитектуры сети автоматически. Эксперименты по «ручному» построению многослойной нейронной сети использовались для подтверждения результатов автоматического поиска.

4.4.4 Ход экспериментов

Пошагово (укрупненно) рассмотрим ход проведения экспериментов:

1. В ПО *Alyuda* загружаются данные (без «сырой» скважины).
2. Разбиение на множества *Training, Validation* и *Testing* по умолчанию происходит случайно, однако мы используем разбиение в определенном порядке *(Training->Validation->Testing),* таким образом, что во множества попадают целые скважины, а не случайные обрывки. Это позволяет лучше судить о качестве обучения и лучше распознать момент переобучения (момент, когда *CCR* по *Training* все еще растет, а *CCR* по *Validation* падает, поскольку сеть, теряя способность абстрагировать и обобщать, начинает запоминать примеры).
3. В качестве алгоритма обучения используем *Conjugate Gradient Descent,* обучаем или до 90% *CCR* по *Training* (как показали многочисленные опыты, к этому моменту переобучение наступает в 100% случаев), или до 200 итераций (аналогично обычно хватает 50–100 итераций).
4. После завершения процесса обучения записываем результат *CCR* по *Validation (Alyuda* использует не последний, когда сеть уже переобучена, а лучший).
5. Загружаем в обученную сеть «сырую» скважину, сохраняем результаты, сравниваем с результатами интерпретации, которые получены экспертами.
6. Записываем результат распознавания по «сырой» скважине.
7. Повторяем пункты 3–6 10–30 раз, поскольку веса инициализируются каждый раз случайно.

(Подробно алгоритм проведения экспериментов описан в приложении 3 «Описание методики проведения экспериментов и применяемого программного обеспечения».)

Случайная инициализация весов нейронной сети вызывает существенный разброс *CCR.* Разброс *CCR* по множеству *Validation* при

использовании алгоритма градиентного спуска небольшой (в реальных экспериментах разброс обычно около ±1–2%), однако разброс результатов по «сырой» скважине оказывается больше. Эксперименты показали, что разброс результатов по «сырой» скважине, даже при минимальном разбросе *CCR* по множеству *Validation,* может достигать 10% и даже 15%. Поэтому в большинстве случаев каждый опыт повторялся от 5 до 10 и более раз, чтобы получить статистически удовлетворительный набор результатов. Исходя из этого, можно сказать, что разница результатов менее 5% не может считаться достаточным основанием для выбора того или иного метода предварительной обработки данных. Такой разброс может присутствовать даже при самом тщательном проведении экспериментов из-за случайной инициализации весов.

4.5 Сравнительный анализ методов машинного обучения

В качестве исходного набора данных, на которых были проведены вычислительные эксперименты, использовались результаты трех видов электрического каротажа (индукционный каротаж, каротаж методом кажущихся сопротивлений, каротаж потенциалов самопроизвольной поляризации), проведенного на 18 скважинах месторождения Буденовское и 12 скважинах месторождения Инкай.

Данные каротажа каждой скважины месторождения Буденовское были проинтерпретированы экспертами, которые выделили 10 основных литотипов (крупные и мелкозернистые пески, глина, смеси песка и глины, галечник и т.п.). Указанные данные, содержащие в общей сложности около 8000 значений, использовались как для обучения ИНС, так и для оценки качества их интерпретации.

Данные месторождения Инкай обрабатывались похожим образом. Отличие заключалось в том, что эти данные вместо экспертных оценок имели оценки по грансоставу, полученные методом кернового опробования, и не имели показателей индукционного каротажа. Общее количество литологических типов по данному месторождению составило около 120, которые были объединены экспертом с получением 20 значимых для целей исследования литотипов.

В процессе обучения ИНС множество данных разбивается на три группы: собственно обучающий набор данных, тестовое и поверочное множество. В задачах, где необходимо обучить сеть распознаванию рядов, используется скользящее окно данных [123]. В нашем случае данные на вход сети поступали в виде скользящего окна данных с параметрами 5+1+5 (пять точек выше + текущая + пять точек ниже текущей). Разбиение такого множества на обучающий, тестовый и

поверочный наборы затруднительно (каждая группа из 11 значений связана с другой), поэтому в работах [121] [125] предложено наряду со стандартной методикой оценки (тестовое и поверочное множество данных) использовать для оценки качества распознавания данные скважин, которые никак не участвовали в процессе обучения (такие скважины были условно названы «сырыми»). Это позволяет полностью исключить возможность запоминания ИНС или иным алгоритмом классификации данных распознаваемого набора.

4.5.1 Результаты, полученные по данным месторождения Буденовское

Проведенные вычислительные эксперименты (язык *Python,* пакет *FANN)* позволили получить результаты, показанные в таблицах 4.11, 4.12, 4.13, 4.14, 4.15, 4.16, 4.17. Расчет осуществлялся с помощью разработанного комплекса программ *ANNClassificator,* описание которого приведено в приложении 6.

Отметим, что полученные результаты показывают существенное влияние на качество распознавания данных каротажа кажущихся сопротивлений.

Классификационные алгоритмы *LDAC, SVM, DLDA, k-NN* оценивались в разных сочетаниях и также с применением плавающего окна данных (таблицы 4.12, 4.13, 4.14, 4.15, 4.16).

Результаты экспериментов показывают, что алгоритм *k-NN* демонстрирует в среднем наилучший результат (исключая ИНС). При этом для каротажа кажущихся сопротивлений *(KS)* данный алгоритм – абсолютный рекордсмен. Этот алгоритм показывает также наиболее близкие по сравнению с ИНС результаты в случае комбинаций входов. Очевидно, что *LDAC* и *k-NN* могут использоваться для оценки нижней границы распознавания ИНС.

4.5.2 Сравнительный анализ «обучаемости» ИНС и k-NN

Как показано выше, одним из важных показателей методов МО является их способность к обучению, которая представляет, по существу, некоторый баланс между способностью метода обобщать, с одной стороны, и точно настраиваться на конкретные данные – с другой. В принципе, алгоритм (или метод), а также его параметры подбираются так, чтобы обеспечить разумный баланс между слишком линейной *(high bias)* и слишком сложной *(high variance)* интерполяционной или разделительной кривой (гиперповерхностью в случае большого числа параметров объектов) (см. раздел 1.2.2). Правильность подбора метода и параметров можно оценить, исследуя способности метода обучаться,

переобучаться *(overfit)* или недообучаться *(underfit)*. Недообученный алгоритм будет показывать одинаково плохие результаты как на тестовом, так и на тренировочном множестве, в то время как переобученный будет демонстрировать высокий результат на тренировочном и низкий на тестовом. В нормальной ситуации при увеличении числа тренировочных примеров ошибка на тренировочном множестве немного возрастает, а на тестовом множестве снижается. Такое поведение алгоритма говорит о балансе между *overfitting* и *underfit*.

Таблица 4.11. Результаты применения ИНС по всем комбинациям входных данных:ИК, КС, ПС, КС-ИК, КС-ПС, ПС-ИК, КС-ПС-ИК

№ эксперимента	ИК	КС	ПС	КС-ИК	КС-ПС	ПС-ИК	КС-ПС-ИК
1	2	3	4	5	6	7	8
1	30	33	29	32	32	32	30
2	54	73	41	76	74	51	74
3	50	58	37	59	62	46	60
4	35	51	43	50	48	33	48
5	50	67	41	69	69	47	67
6	47	71	32	71	69	42	69
7	43	47	38	52	50	39	47
8	47	66	33	70	69	42	66
9	32	52	27	57	56	33	56
10	51	67	34	65	65	49	64
11	64	63	45	63	60	48	62
12	32	55	47	54	58	40	54
13	32	49	47	51	54	38	52
14	40	68	29	74	73	36	73
15	12	18	30	18	17	18	18
16	62	72	45	74	71	51	70
17	44	55	33	53	55	39	52
18	47	64	44	68	70	48	70
Среднее	43	57	37	59	58	41	57

Примечание. Колонки 2–8 содержат среднее после 10 итераций значение результата распознавания в %. Строки 1–18 – значения качества распознавания для каждой из 18 скважин при подаче на вход сети результатов разного вида каротажа и их сочетаний. Последняя колонка – все доступные виды каротажа. Последняя строка – среднее значение процента распознавания по всем 18 скважинам.

Таблица 4.12. Результаты применения LDAC

Комбинация входа	Без окна	Окно 3	Окно 5	Окно 7
КС-ПС-ИК	0,53	0,58	0,58	0,57
КС	0,50	0,54	0,54	0,54
ПС	0,28	0,28	0,27	0,28
ИК	0,33	0,34	0,34	0,34
КС-ПС	0,52	0,57	0,57	0,57
КС-ИК	0,52	0,57	0,57	0,57
ПС-ИК	0,33	0,34	0,34	0,34

Таблица 4.13. Результаты применения Linear SVM

Комбинация входа	Без окна	Окно 3	Окно 5	Окно 7
КС-ПС-ИК	0,46	0,50	0,51	0,51
КС	0,41	0,44	0,44	0,44
ПС	0,28	0,28	0,28	0,27
ИК	0,35	0,35	0,35	0,35
КС-ПС	0,43	0,46	0,46	0,47
КС-ИК	0,46	0,50	0,50	0,50
ПС-ИК	0,35	0,35	0,35	0,35

Таблица 4.14. Результаты применения Non-linear SVM

Комбинация входа	Без окна	Окно 3	Окно 5	Окно 7
КС-ПС-ИК	0,52	0,31	0,28	0,28
КС	0,55	0,58	0,50	0,41
ПС	0,35	0,33	0,32	0,30
ИК	0,43	0,45	0,43	0,40
КС-ПС	0,54	0,40	0,31	0,28
КС-ИК	0,53	0,43	0,34	0,30
ПС-ИК	0,43	0,31	0,28	0,27

переобучаться *(overfit)* или недообучаться *(underfit)*. Недообученный алгоритм будет показывать одинаково плохие результаты как на тестовом, так и на тренировочном множестве, в то время как переобученный будет демонстрировать высокий результат на тренировочном и низкий на тестовом. В нормальной ситуации при увеличении числа тренировочных примеров ошибка на тренировочном множестве немного возрастает, а на тестовом множестве снижается. Такое поведение алгоритма говорит о балансе между *overfitting* и *underfit*.

Таблица 4.11. Результаты применения ИНС по всем комбинациям входных данных:ИК, КС, ПС, КС-ИК, КС-ПС, ПС-ИК, КС-ПС-ИК

№ эксперимента	ИК	КС	ПС	КС-ИК	КС-ПС	ПС-ИК	КС-ПС-ИК
1	2	3	4	5	6	7	8
1	30	33	29	32	32	32	30
2	54	73	41	76	74	51	74
3	50	58	37	59	62	46	60
4	35	51	43	50	48	33	48
5	50	67	41	69	69	47	67
6	47	71	32	71	69	42	69
7	43	47	38	52	50	39	47
8	47	66	33	70	69	42	66
9	32	52	27	57	56	33	56
10	51	67	34	65	65	49	64
11	64	63	45	63	60	48	62
12	32	55	47	54	58	40	54
13	32	49	47	51	54	38	52
14	40	68	29	74	73	36	73
15	12	18	30	18	17	18	18
16	62	72	45	74	71	51	70
17	44	55	33	53	55	39	52
18	47	64	44	68	70	48	70
Среднее	43	57	37	59	58	41	57

Примечание. Колонки 2–8 содержат среднее после 10 итераций значение результата распознавания в %. Строки 1–18 – значения качества распознавания для каждой из 18 скважин при подаче на вход сети результатов разного вида каротажа и их сочетаний. Последняя колонка – все доступные виды каротажа. Последняя строка – среднее значение процента распознавания по всем 18 скважинам.

Таблица 4.12. Результаты применения LDAC

Комбинация входа	Без окна	Окно 3	Окно 5	Окно 7
КС-ПС-ИК	0,53	0,58	0,58	0,57
КС	0,50	0,54	0,54	0,54
ПС	0,28	0,28	0,27	0,28
ИК	0,33	0,34	0,34	0,34
КС-ПС	0,52	0,57	0,57	0,57
КС-ИК	0,52	0,57	0,57	0,57
ПС-ИК	0,33	0,34	0,34	0,34

Таблица 4.13. Результаты применения Linear SVM

Комбинация входа	Без окна	Окно 3	Окно 5	Окно 7
КС-ПС-ИК	0,46	0,50	0,51	0,51
КС	0,41	0,44	0,44	0,44
ПС	0,28	0,28	0,28	0,27
ИК	0,35	0,35	0,35	0,35
КС-ПС	0,43	0,46	0,46	0,47
КС-ИК	0,46	0,50	0,50	0,50
ПС-ИК	0,35	0,35	0,35	0,35

Таблица 4.14. Результаты применения Non-linear SVM

Комбинация входа	Без окна	Окно 3	Окно 5	Окно 7
КС-ПС-ИК	0,52	0,31	0,28	0,28
КС	0,55	0,58	0,50	0,41
ПС	0,35	0,33	0,32	0,30
ИК	0,43	0,45	0,43	0,40
КС-ПС	0,54	0,40	0,31	0,28
КС-ИК	0,53	0,43	0,34	0,30
ПС-ИК	0,43	0,31	0,28	0,27

Таблица 4.15. Результаты применения DLDA

Комбинация входа	Без окна	Окно 3	Окно 5	Окно 7
КС-ПС-ИК	0,44	0,54	0,52	0,50
КС	0,28	0,28	0,28	0,28
ПС	0,33	0,36	0,36	0,37
ИК	0,44	0,54	0,52	0,50
КС-ПС	0,45	0,55	0,53	0,51
КС-ИК	0,33	0,34	0,33	0,33
ПС-ИК	0,45	0,54	0,52	0,51

Таблица 4.16. Результаты применения алгоритма k-NN (50 соседних точек)

Комбинация входа	Без окна	Окно 3	Окно 5	Окно 7
КС-ПС-ИК	0,52	0,54	0,53	0,52
КС	0,54	0,59	0,59	0,58
ПС	0,33	0,33	0,34	0,35
ИК	0,41	0,41	0,41	0,41
КС-ПС	0,52	0,56	0,54	0,54
КС-ИК	0,52	0,56	0,56	0,55
ПС-ИК	0,39	0,40	0,40	0,40

Таблица 4.17. Сравнительные результаты применения алгоритмов к данным месторождения Буденовское

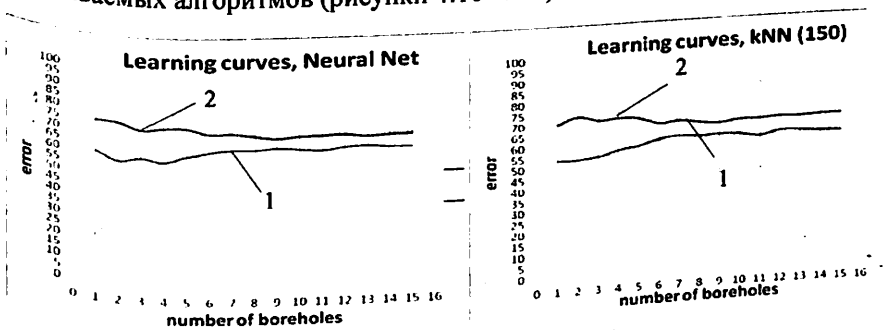
Комбинация входа	ИНС	LDAC	Linear SVM	Non-linear SVM	DLDA	k-NN
КС-ПС-ИК	57	58	50	31	54	54
КС	57	54	44	58	28	59
ПС	37	28	28	33	36	33
ИК	43	34	35	45	54	41
КС-ПС	58	57	46	40	55	56
КС-ИК	59	57	50	43	34	56
ПС-ИК	41	34	35	31	54	40
Среднее	50.29	46	41.14	40.14	45	48.42

Примечание. В качестве результатов алгоритмов LDAC, SVM, DLDA, k-NN взяты лучшие показатели, полученные при применении плавающего окна размером 3.

Для выявления указанных качеств были проведены дополнительные вычислительные эксперименты в пакете *RapidMiner* с использованием алгоритмов ИНС и *k-NN*, которые показали лучшие результаты в предыдущих экспериментах [133]. Для проведения экспериментов по оценке кривых обучения были отобраны 30 скважин месторождения Инкай: 15 скважин для обучения (множество данных *train*) и 15 скважин для тестирования (множество данных *test*). Ход эксперимента заключался в следующем: вначале для обучения алгоритма использовалась одна скважина. Далее ее же данные применялись для классификации (множество *train*). Затем, используя обученный метод, классифицировались данные другой (тестовой) скважины (из множества *test*). Этот шаг повторялся, но уже для двух тренировочных и тестовых скважин, потом для трех и т.п.

В результате был получен набор данных, представленный в приложении 6 «Исходные данные для построения кривых обучения *(learning curves)*».

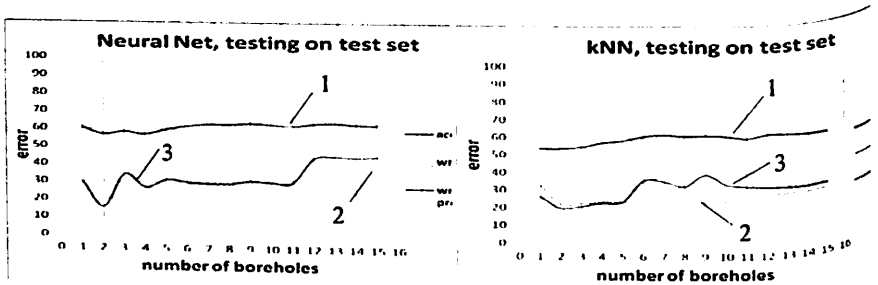
На основе полученных данных построены зависимости показателя точности *(accuracy)*, взвешенного показателя «точности» *(weighted mean precision2)* и взвешенного показателя «полноты» *(weighted mean recall)* от количества участвовавших в обучении наборов данных для обоих сравниваемых алгоритмов (рисунки 4.10–4.12).



1 – accuracy1 (test), 2 – accuracy2 (train)

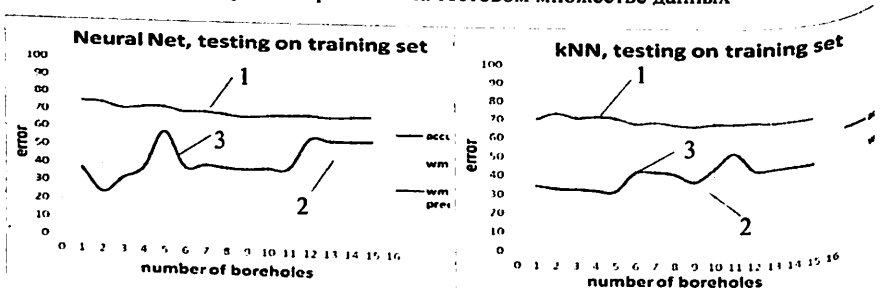
Рисунок 4.10. Зависимость между показателями точности по тестовым (accuracy1) и обучающим (accuracy2) множествам данных при применении ИНС и k-NN

Как видно, хотя средние показатели точности алгоритмов довольно близки (ИНС на 2–3% лучше), тем не менее для алгоритма *k-NN* расстояние между кривыми точности для обучающего и тестового набора данных больше (рисунок 4.10). Это говорит о том, что способность обучения данного алгоритма ниже, то есть он несколько более линеен.



1 – accuracy1; 2 – wm recall1; 3 – wm precision1

Рисунок 4.11. Зависимость показателей CCR (accuracy), взвешенного показателя «точности» (weighted mean precision2) и взвешенного показателя «полноты» (weighted mean recall) ИНС от количества участвовавших в обучении наборов данных при тестировании на тестовом множестве данных



1 – accuracy2; 2 – wm recall2; 3 – wm precision2

Рисунок 4.12. Зависимость CCR (accuracy), взвешенного показателя «точности» (weighted mean precision2) и взвешенного показателя «полноты» (weighted mean recall) ИНС от количества участвовавших в обучении наборов данных при тестировании на обучающем множестве данных

Более того, взвешенный показатель «точности» *(weighted mean precision2)* и взвешенный показатель «полноты» *(weighted mean recall)* алгоритма *k-NN* при заданном параметре числа близлежащих соседей (k=150) существенно ниже (рисунки 4.11, 4.12).

4.5.3 Сравнительный анализ качества распознавания на месторождениях Буденовское и Инкай

Сравнительный анализ качества распознавания для данных месторождений позволяет оценить степень субъективности экспертных оценок, поскольку на месторождении Инкай использовались данные кренового опробирования, в отличие от Буденовского (таблица 4.18), где имеются только данные экспертных оценок.

Для анализа данные по месторождению Инкай были предварительно обработаны так, чтобы из приблизительно сотни рассматриваемых там литологических типов сформировать вначале 24, а затем 8 основных типов путем объединения. В последнем случае число литологических типов на обоих месторождениях совпадает. Полученные результаты указаны в таблице 4.19.

Видно, что качество распознавания на Инкае ниже, но, как и на Буденовском, при увеличении числа выходов сети оно увеличивается, и при трех выходах (ИНС3) качество становится практически одинаковым (отличие 4%) (таблица 4.20).

Кроме прямого сравнения интерес представляет также анализ качества распознавания отдельных пород. Для этого использованы таблицы смежности (таблицы 4.21 и 4.22), которые показывают процент правильности распознавания отдельных пород, а также позволяют видеть, как именно ИНС «путает» породы.

Видно, что в обоих случаях качество распознавания выше для тех пород, количество примеров которых больше. В то же время для пород 4, 7 и 47 качество распознавания при обучении сети с помощью экспертов выше (Буденовское).

Можно отметить, что в обоих случаях 59 в основном расценивается как 12 (большая часть ошибок), 1 как 123 и 12, 47 как 3 и 7, 7 отличается в зависимости от месторождения, причем на Буденовском этот тип распознается практически без ошибок (!), 4 часто принимается за 3, 12 за 123 и 3. Пород 123 и 3 много на обоих месторождениях, и ошибки тоже похожи.

То есть, несмотря на общую похожесть результатов, для некоторых пород месторождение Буденовское (оценки выполнены экспертами) показывает лучшие результаты, что требует дополнительного анализа. Показательна оценка дисперсии (таблица 4.20).

Видно, что для Инкая дисперсия невелика, то есть если сеть ошибается, то она ошибается «стабильно», в то время как для Буденовского дисперсия в несколько раз выше (41% по сравнению с 14%).

4.6 Заключение по разделу 4

В данном разделе описаны экспериментальные результаты, полученные при применении методов МО для обработки данных электрического каротажа скважин на урановых месторождениях пластово-инфильтрационного типа.

Сформулирована задача литологического расчленения скважин, которая в формальной форме описывает процесс классификации данных геофизического исследования скважин.

Таблица 4.18. Результаты по месторождению Буденовское

Скважина	ИНС1	ИНС2	ИНС3
1	0.39	0.70	0.92
2	0.80	0.96	0.99
3	0.65	0.94	0.98
4	0.51	0.85	0.95
5	0.74	0.94	0.98
6	0.75	0.94	0.99
7	0.56	0.88	0.96
8	0.72	0.90	0.93
9	0.55	0.89	0.99
10	0.72	0.94	0.97
11	0.69	0.95	0.98
12	0.65	0.94	0.97
13	0.61	0.92	0.96
14	0.76	0.94	0.98
15	0.18	0.52	0.85
16	0.77	0.94	0.96
17	0.56	0.76	0.86
18	0.74	0.93	0.96
Среднее	0.63	0.88	0.95

Примечание. В качестве входных данных для обучения и распознавания использованы KS и PS. Для проведения экспериментов были сформированы сети с несколькими выходами.

ИНС1 — *1 из 1 правильный (сеть предлагает только один ответ).*
ИНС2 — *1 из 2 правильный (сеть предлагает два ответа).*
ИНС3 — *1 из 3 правильный (сеть предлагает 3 ответа).*

Экспериментально исследованы методы предобработки данных, которая является одной из самых важных при решении задачи обучения и

распознавания с помощью методов МО. Описаны проведенные вычислительные эксперименты с применением пакетов *Alyuda NeuroIntelligence, RapidMiner,* библиотек *WaveUtils, FANN,* программных средств собственной разработки в соответствии с общей схемой настройки системы МО. Были исследованы различные способы нормировки, сглаживания, исключения сдвига данных, подготовки обучающей выборки, использования параметров глубины, экстремумов КС. При этом использование различных методов сглаживания сигнала, исключения шумов не дало ощутимого прироста в качестве классификации. Вместе с тем эксперименты показали высокую эффективность линейной нормировки данных по сравнению с нелинейными методами. В результате предложена и экспериментально отработана простая методика подготовки данных, включающая удаление аномальных значений, линейную нормировку и центрирование данных (для каждой скважины отдельно), форматирование данных для формирования плавающего окна данных. Основные методы предобработки реализованы в библиотеке программ *Preprocessing Module* (приложение 4 «Описание программного комплекса *Preprocessing Module*»).

Оценено качество экспертной классификации и показано, что оно существенно отличается от данных, полученных путем кернового апробирования. Существенно отличаются и результаты каждого из 3 экспертов, участвовавших в эксперименте.

Предложен метод апробации алгоритмов машинного обучения, получивший название «синтезированная скважина», который позволяет сгенерировать данные каротажа, исходя из предположений о физическом процессе снятия показаний. Методы МО на таких данных показали очень хорошие результаты классификации.

Экспериментально оценены алгоритмы МО. Показано, что ИНС прямого распространения обеспечивает в среднем лучшие показатели ошибок классификации и лучшую обучаемость (лучшее качество кривой обучения).

Исходя из полученных результатов сравнительного анализа экспертного оценивания и результатов, полученных на синтезированных данных, можно говорить о том, что показатели качества распознавания, получаемые с применением методов МО (в частности, искусственные нейронные сети), хотя и несколько хуже, чем у людей-экспертов, в то же время приближаются к ним. Следовательно, применение методов МО, во-первых, вполне оправданно, во-вторых, может быть несколько улучшено путем использования дополнительных методов предобработки. Поиск таких методов составляет предмет дальнейших исследований.

Таблица 4.19. Результаты по месторождению Инкай

Скважина	ИНС1	ИНС2
1	0.59	0.86
2	0.48	0.85
3	0.53	0.85
4	0.54	0.81
5	0.62	0.90
6	0.50	0.71
7	0.42	0.68
8	0.54	0.80
9	0.46	0.78
10	0.58	0.78
11	0.43	0.74
12	0.41	0.68
13	0.56	0.78
14	0.45	0.82
15	0.55	0.85
16	0.43	0.82
17	0.49	0.75
18	0.30	0.64
19	0.47	0.76
20	0.47	0.83
21	0.57	0.84
22	0.48	0.75
23	0.60	0.78
24	0.39	0.74
Среднее	0.50	0.78

Таблица 4.20. Дисперсия результатов для месторождений Буденовское и Инкай

Месторождение	ИНС1	ИНС2	ИНС3
Буденовское	0.413817	0.217813	0.029894
Инкай	0.136998	0.093404	0.046456

Таблица 4.21. Таблица смежности Буденовского

Ответ сети/ Правильный ответ	123	3	12	4	7	47	1	59
123	1344	432	442	15	6	6	10	1
3	486	1354	42	419	4	31	1	6
12	318	17	671	1	12	1	82	17
4	44	130	5	371	24	74	1	1
7	33	30	12	48	1249	165	1	1
47	12	30	0	62	35	71	0	0
1	0	0	0	0	0	0	0	0
59	0	0	1	0	0	0	0	0
Правильно распознаны	1344	1354	671	371	1249	71	0	0
Всего примеров	2237	1993	1173	916	1330	348	95	26
Процент распознавания	0.60	0.68	0.57	0.41	0.94	0.20	0	0

Таблица 4.22. Таблица смежности Инкая

Ответ сети/ Правильный ответ	123	3	12	4	7	47	1	59
123	4156	1402	2015	214	66	9	853	2
3	1905	6118	586	2633	386	117	83	2
12	686	172	1405	21	27	1	751	51
4	35	371	12	494	264	1	7	1
7	20	159	22	291	563	18	3	1
47	0	0	0	0	0	0	0	0
1	294	121	1036	14	59	0	1502	17
59	80	60	25	21	11	3	116	15
Правильно распознаны	4156	6118	1405	494	563	0	1502	15
Всего примеров	7176	8403	5101	3688	1376	149	3315	89
Процент распознавания	0.58	0.73	0.28	0.13	0.41	0.00	0.45	0.17

5 СИСТЕМА РАСПОЗНАВАНИЯ ЛИТОЛОГИЧЕСКОГО СОСТАВА СКВАЖИН НА УРАНОВЫХ МЕСТОРОЖДЕНИЯХ

5.1 Введение

Для реализации алгоритмов классификации необходимо создать программный инструмент, обеспечивающий полный цикл действия системы, начиная от ввода данных и заканчивая их классификацией на базе выбранных методов машинного обучения. Разработка такого инструмента позволит провести необходимые вычислительные эксперименты, оценить качество работы реализованных алгоритмов машинного обучения и иные практические потребности. Ниже описан прототип системы классификации данных электрического каротажа, включая требования к системе, описание способа реализации системы, архитектуры базы данных (БД), интерфейса системы. Описание модуля предварительной обработки данных *Preprocessing Module,* на котором отрабатывались методы предобработки системы, приведено в приложении 4. В качестве дальнейшего развития системы предложено использовать мультиагентный подход [127].

5.2 Требования к системе

Перед исследовательской группой была поставлена задача разработать систему для реализации части вышеописанных алгоритмов.

Цель создания системы – разработка удобного и функционального инструмента с гибкой архитектурой для возможных изменений.

Выполняемые системой задачи:

1. Настройка месторождений и скважин.
2. Загрузка необработанных данных электрического каротажа – ПС, КС, ИК.
3. Пошаговая предобработка загруженных данных: а) определение закисления и выбросов; б) выставление на один уровень; в) линейная нормировка; г) исключение сдвига каротажных кривых.
4. Обучение и тестирование нейронной сети.
5. Распознавание пород по данным электрического каротажа с помощью трех алгоритмов: а) с использованием обучаемой системы на базе ИНС с прямым распространением сигнала; б) «классический способ» определения пород по данным КС; в) метрический метод (реализация алгоритма *k-NN).*
6. Анализ результатов работы алгоритмов с возможностью сравнения результатов различных алгоритмов и редактирования результатов распознавания.

Краткое описание подходов к реализации системы приведено в работе [119] [126].

5.3 Способ реализации

Учитывая сложность алгоритмов распознавания, а также возможность добавления новых алгоритмов распознавания, команда разработчиков решила выбрать клиент-серверную архитектуру приложения. При этом основная логика системы была сосредоточена на стороне базы данных, в хранимых процедурах *(stored procedure).*

В качестве сервера выступает СУБД *PostgreSQL.* Для разработки *Desktop* клиента был выбран *Delphi 7* с преднастроенным фреймворком. Для работы с нейронной сетью – обучение, тестирование и интерпретация – использовалась библиотека *FANN (http://leenissen.dk/ fann/wp/).* Взаимодействие с библиотекой *FANN* происходит на серверной стороне.

Система имеет большую графическую составляющую, куда входит работа с результатами распознавания: отображение графиков КС, ПС и ИК, а также графическое отображение пород в сопоставлении с графиком на каждом участке скважины. Имеется возможность редактирования результатов распознавания непосредственно на графике, используя графические обозначения литологического состава.

Система применяется исследовательской командой для проведения вычислительных экспериментов.

5.4 Архитектура БД

Ниже (рисунок 5.1) отображена архитектура базы данных с отображением библиотек по работе с искусственными нейронными сетями.

Основными объектами БД являются:

1. таблица с хранением загруженных данных электрического каротажа *(karotaj_data_tab);*
2. таблица с предобработанными данными электрического каротажа *(norm_data_tab);*
3. таблица с результатами интерпретации данных *(result_tab).*

Вспомогательные объекты:

1. скважины (well_tab);
2. месторождение (deposit_tab);
3. параметры нейронной сети (net_parameters_tab);
4. методы предобработки данных *(normal_method_tab).*

5.5 Интерфейс системы

Система имеет графический интерфейс, позволяющий взаимодействовать с пользователем для выполнения следующих задач:

– Пошаговая предобработка загруженных данных.

– Данные электрического каротажа предварительно обрабатываются. В рамках предобработки осуществляются шаги, описанные выше в разделе о выполняемых задачах системы.

– Система выполняет шаги предобработки последовательно, при этом показывая на графике изменения данных (рисунок 5.2).

5.6 Интерпретация данных электрического каротажа

Конечным результатом работы системы является распознавание литологических пород на основе данных электрического каротажа (рисунок 5.3).

На графике отображаются показания каротажа значений ПС и КС. Слева от графика находятся три колонки. В первой колонке показаны породы, соответствующие графику. Вторая и третья колонки являются вспомогательными для сравнения с результатами распознавания других алгоритмов той же самой скважины, а также других скважин, например соседних.

5.7 Выставление уровней для алгоритма определения по графику КС

Для более точной работы алгоритма распознавания на основе графика КС необходима его предварительная настройка – выставление уровня литологических пород. Настройка может производиться отдельно для каждого месторождения или группы скважин. Для данной задачи в системе был разработан отдельный функционал (рисунок 5.4).

5.8 Усовершенствование инструмента распознавания

В целях создания полнофункционального инструмента для выполнения дальнейших исследований перед командой разработчиков поставлены следующие задачи:

1. Реализация алгоритма комитетного синтеза на базе *Simple Integrator of Post-processing Stage (SIPP)*, рассмотренного в [53].
5. Перевод системы на *web*-платформу с выделенным сервером в интернете для обеспечения возможностей удаленной работы.
6. Реализация в системе дополнительных алгоритмов распознавания.
7. Реализация в системе дополнительных алгоритмов комитетного синтеза.

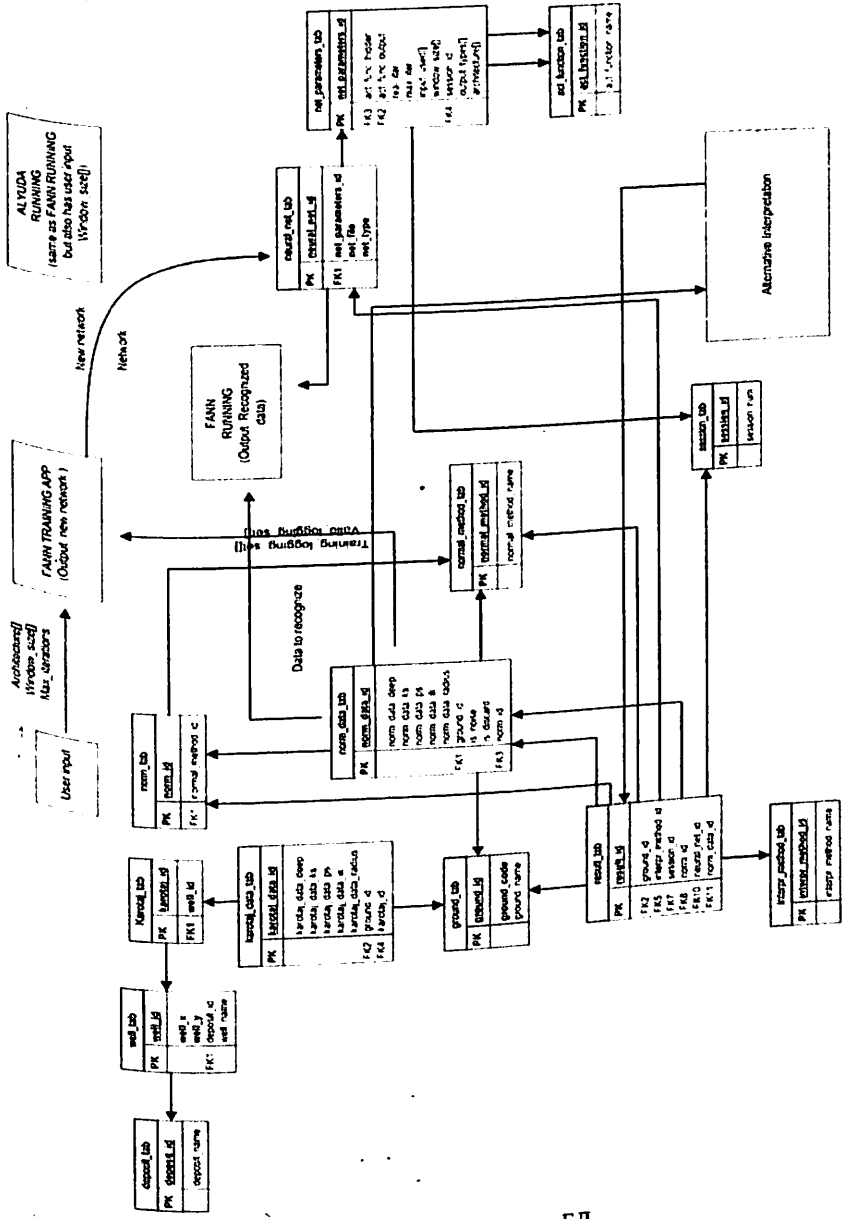


Рисунок 5.1. Архитектура БД

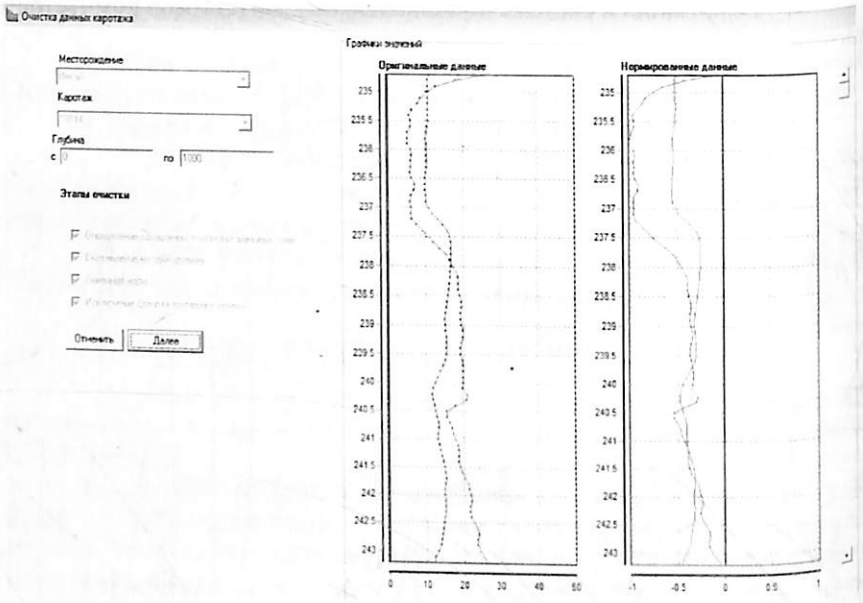


Рисунок 5.2. Пошаговая предобработка загруженных данных

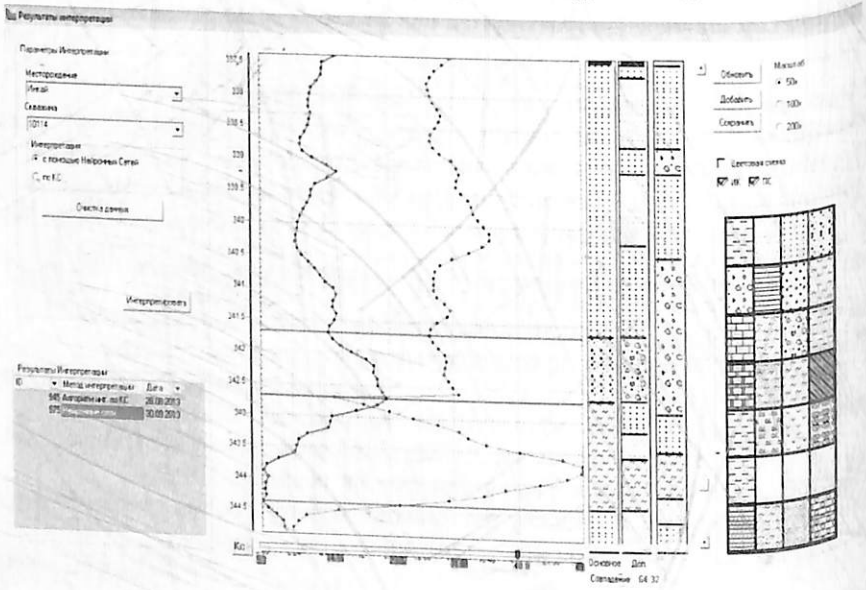


Рисунок 5.3. Результат интерпретации данных каротажа

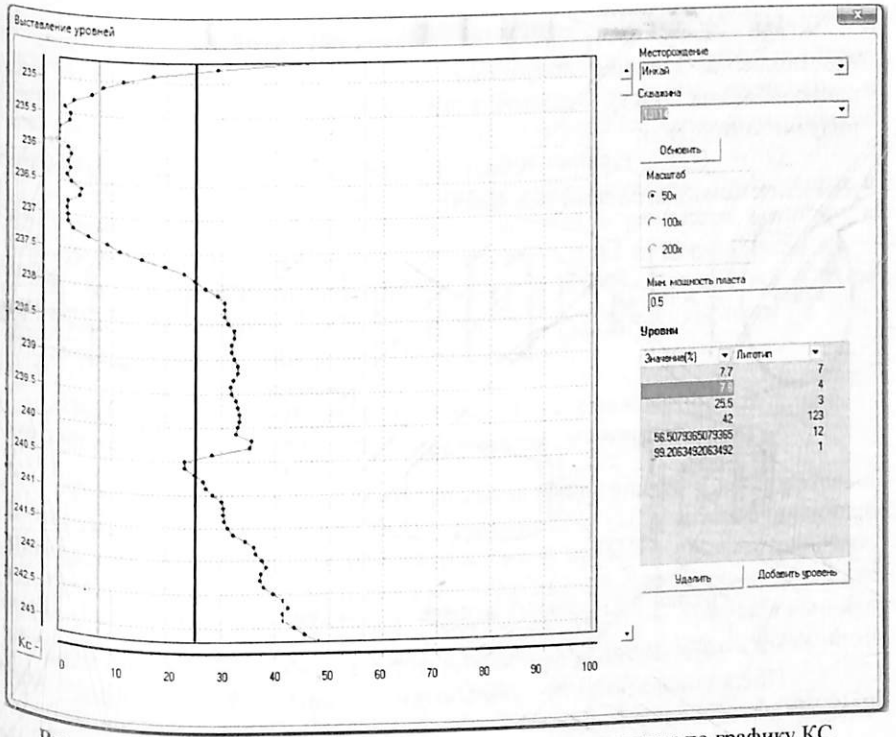


Рисунок 5.4. Выставление уровней для алгоритма определения по графику КС

5.9 Реализация платформы распознавания с использованием мультиагентного подхода

Следующий этап улучшения системы может быть связан с построением системы комитетного синтеза алгоритмов на базе многоагентного подхода. Комитет классификаторов позволяет добиться улучшения результатов классификации при соблюдении условия максимальной декорреляции между отдельными классификаторами, построенными возможно на базе одного алгоритма, как, например, описано в [129], где рассматривается ансамбль SVM классификаторов.

Мультиагентный подход, в свою очередь, как метод реализации, позволяет добиться высокой степени параллелизма, упростить проектирование, реализацию, обновление системы, а также обеспечить самоорганизацию системы [130]. Примерная схема многоагентной системы классификации *(MACS – Multi-Agent Classification System)* приведена на рисунке 5.5.

MACS может быть формально описана как

MAS = {A, D, VF},

где A = {A1,...,A6} – множество агентов, состоящее из подмножеств агентов получения данных (A1), предобработки (A2), распознавания (A3), комитетного синтеза (A4), обучения (A5), визуализации (A6).

D = {D1,...,D5} – блоки промежуточной памяти для обмена данными между агентами («доски»).

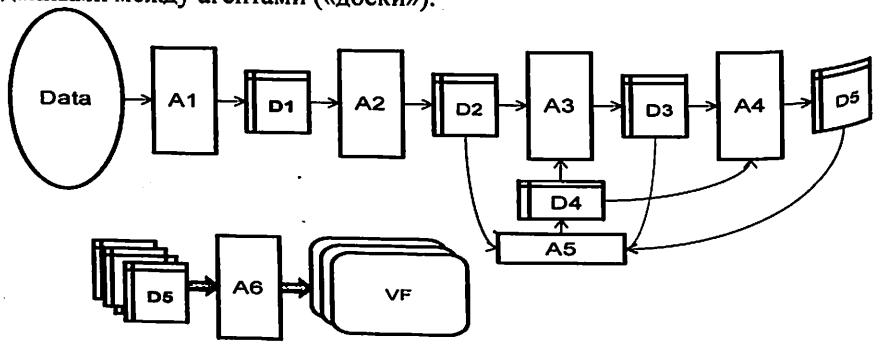


Рисунок 5.5. Схема мультиагентной системы классификации

VF – множество графических форм визуализации результатов.

Последовательность обработки данных с помощью *MACS* заключается в следующем.

Данные извлекаются с помощью различных агентов получения данных (A1), которые размещают полученные данные на D1. Описание данных содержит идентификатор, источник информации, дату и время получения, предметную область и другие сведения, необходимые для точной идентификации набора. Агенты предобработки (A2) обрабатывают данные путем нормализации, сглаживания, устранения аномальных значений, фильтрации и т.п. Обработанные данные снабжаются соответствующим ярлыком и помещаются на D2. При этом данные, которые уже проанализированы экспертом (в задаче обучения с учителем), снабжаются соответствующей меткой и могут быть использованы агентами обучения (A5) для настройки агентов распознавания (классификации).

Агенты распознавания (A3) обрабатывают данные и помещают результат классификации на D3, откуда они могут быть извлечены агентами комитетного синтеза (A4) для постобработки. Часть агентов обучения (A5) служит для настройки систем комитетного синтеза. Общая задача агентов данного множества – формировать системы настройки агентов распознавания (веса нейронных сетей, правила продукционные или нечеткие и т.п.) и агентов комитетного синтеза.

Агенты визуализации (A6) получают данные с D1–D5 и визуализируют их в соответствующих графических или текстовых окнах. Часть агентов может выдавать информацию о своем состоянии в реальном масштабе времени, которая также может быть визуализирована для учета хода процесса обучения и распознавания.

При решении задачи распознавания работа системы заключается в последовательной передаче данных от агентов с меньшим номером к агентам с большим номером – от группы агентов A1 к A2 и далее до A4.

При настройке системы используются агенты группы A5. Объем использования агентов, оценки времени настройки системы и т.п. являются предметом дальнейших исследований.

5.10 Заключение по разделу «Система распознавания литологического состава скважин на урановых месторождениях»

В целях решения задачи «Реализация метрических и нейросетевых алгоритмов классификации» было разработано ПО, обеспечивающее применение искусственных нейронных сетей прямого распространения, алгоритма *k-NN* и других методов классификации в задаче классификации данных электрического каротажа. Разработана система, обеспечивающая загрузку и предобработку данных, обучение и тестирование нейронной сети и распознавание пород с помощью трех методов:

1. Использование обучаемой системы на базе ИНС с прямым распространением сигнала.
2. «Классический способ» определения пород по данным КС.
3. Метрический метод (реализация алгоритма *k-NN)*.

Развитие системы возможно путем использования большего числа алгоритмов с применением комитетного синтеза, который при прочих равных условиях способен повысить качество распознавания. При этом представляется перспективным использование мультиагентного подхода при применении множества алгоритмов МО и формировании комитета алгоритмов. Предполагается, что агентный подход может обеспечить высокую степень параллелизма и возможности некоторой самоорганизации системы распознавания, а также облегчить последующую модификацию и сопровождение программного комплекса.

ЗАКЛЮЧЕНИЕ

В настоящее время сложились предпосылки как общего, так и частного характера, делающие возможным применение интеллектуальных методов во многих областях исследований и производства, в том числе в области добычи полезных ископаемых.

Использование указанных методов позволяет снизить затраты, связанные с добычей ископаемых. При этом сфера применения интеллектуальных методов весьма широка. Методы машинного обучения широко используются в робототехнике, в мультиагентных и кибер-физических системах. Построенная в первом разделе классификация новых областей исследования в области ИКТ отражает картину широкого применения интеллектуальных методов. При этом слабый ИИ, к которому относится МО, содержит ряд алгоритмов и методов, способных решать трудноформализуемые задачи. Одним из широко исследуемых и применяемых подходов являются искусственные нейронные сети, которые, несмотря на имеющиеся недостатки, связанные, в первую очередь, со сложностями получения обучающего множества примеров удовлетворительного качества, могут демонстрировать хорошие результаты в задачах классификации.

Невзирая на некоторые трудности, вызванные применением методов МО, повышение точности интерпретации и исключение грубых ошибок интерпретаторов способны привести к существенному экономическому эффекту. Можно предположить, что применение точных автоматических и своевременных методов литологического расчленения скважин урановых месторождений РК может сократить экономические потери на сумму до 2 млн. долларов в год.

Использование методов МО требует выполнения ряда последовательных шагов, связанных с подготовкой исходных данных, обучением системы искусственного интеллекта и собственно распознаванием. Поиск оптимальных путей применения методов машинного обучения для автоматической интерпретации данных электрического каротажа составляет суть представленной работы.

Во втором разделе описаны основные виды каротажа скважин по добыче урана методом подземного выщелачивания. Широко используются три метода электрического каротажа: каротаж сопротивлений (КС); электрический каротаж, основанный на регистрации параметров естественного электрического поля, – каротаж потенциалов самопроизвольной поляризации (ПС); индукционный каротаж (ИК). В настоящее время данные каротажа обрабатываются с помощью программных продуктов «Кобра», *GikLet*, «Альфа». Системы «Кобра» и *GikLet* не содержат средств автоматизации процесса интерпретации. Система «Альфа», разработанная в 2007 году, включает в себя модули интерпретации электрокаротажа, гамма-каротажа, термометрии, токового каротажа, контрольных каротажей, расходометрии, редактор кривых и планшет данных ГИС. При этом компьютерная (автоматическая) интерпретация электрокаротажа проводится только по кривой КС, остальные данные (ПС, ИК, КМ, ГК) используются для ручной корректировки человеком-интерпретатором. Кроме того, при закислении скважины автоматическая интерпретация по одной лишь КС зачастую становится невозможной. Также отсутствует законченная формальная модель, которая могла бы служить основой для автоматического литологического расчленения с учетом всех видов каротажа. Таким образом, необходимы большие затраты времени и профессионализм интерпретатора, в то время как технология производства зачастую требует быстрого принятия решений. Это означает, что необходима система, позволяющая выполнять интерпретацию данных каротажа в автоматическом режиме, возможно, с предварительным обучением на данных, проинтерпретированных экспертами.

С этой целью в третьем разделе рассмотрены методы МО как часть методов искусственного интеллекта, пригодных для анализа данных электрического каротажа. Сформирована таксономия методов МО и выявлено, что в данной работе, исходя из ее постановки, требуется использование методов обучения с учителем *(SL)*. В разделе описана схема настройки методов МО для решения задачи классификации литологических слоев. Сделана формальная постановка задачи литологического расчленения и описаны некоторые часто используемые алгоритмы (линейная регрессия, полиномиальная регрессия, логистическая регрессия, искусственные нейронные сети, алгоритм *k-NN, SVN, LDAC, DLDA)*. Приведены показатели оценки точности классификации: доля правильных ответов *(accuracy)*, «точность» *(precision)*, «полнота» *(recall)*, и обобщающие показатели – *T1 Score, Kappa*, позволяющие оценивать качество классификации и сравнивать методы классификации между собой. Описано понятие кривой обучения и способы ее интерпретации для выбора метода МО и/или его настройки. Подробно описаны методы предобработки данных, включая методы устранения аномальных значений, нормировки.

Для выбора методов обработки и предобработки данных выполнены вычислительные эксперименты, которые описаны в четвертом разделе наряду с экспериментальными результатами, полученными при применении методов МО в задаче обработки данных электрического каротажа скважин на урановых месторождениях пластово-инфильтрационного типа. Эксперименты проведены с

использованием данных нескольких десятков скважин двух месторождений. Экспериментально исследованы методы предобработки данных. Выполнены вычислительные эксперименты с применением пакетов *Alyuda, RapidMiner,* библиотек *WaveUtils, FANN,* программных средств собственной разработки в соответствии с общей схемой настройки системы МО. Исследованы различные способы нормировки, сглаживания, исключения сдвига данных, подготовки обучающей выборки, использования параметров глубины, экстремумов КС. При этом применение различных методов сглаживания сигнала, исключения шумов не дало ощутимого прироста в качестве классификации. Эксперименты показали высокую эффективность линейной нормировки данных по сравнению с нелинейными методами. В результате предложена и экспериментально отработана методика подготовки данных, включающая удаление аномалий, линейную нормировку и центрирование данных (для каждой скважины отдельно), форматирование данных для формирования плавающего окна данных. Оценено качество экспертной классификации и показано, что оно существенно отличается от данных, полученных путем кернового опробования. Существенно отличаются и результаты каждого из 3 экспертов.

Предложен и описан метод оценки алгоритмов МО, получивший название «синтезированная скважина», который позволяет сгенерировать данные каротажа, исходя из предположений о физическом процессе снятия показаний. Методы МО на таких данных показали очень хорошие результаты классификации. При этом многослойные искусственные нейронные сети прямого распространения обеспечивают в среднем лучшие показатели ошибок классификации и лучшее качество кривой обучения.

Исходя из результатов сравнительного анализа экспертного оценивания и результатов, полученных на синтезированных данных, сделан вывод о том, что показатели качества распознавания, получаемые с применением методов МО (в частности, искусственных нейронных сетей), сравнимы с таковыми у людей-экспертов. Следовательно, использование указанных методов, во-первых, вполне оправданно, во-вторых, качество автоматической интерпретации может быть повышено путем улучшения существующих тренировочных наборов данных и применения дополнительных методов предобработки.

Основные методы предобработки реализованы в библиотеке программ *Preprocessing Module* (приложение 4 «Описание программного комплекса *Preprocessing Module*»).

В пятом разделе описано разработанное ПО, обеспечивающее применение искусственных нейронных сетей, алгоритма *k-NN* и других методов классификации в задаче интерпретации данных электрического каротажа. Разработана система, обеспечивающая загрузку данных, предобработку, обучение и тестирование искусственной нейронной сети и распознавание пород с помощью трех методов:

- Использование обучаемой системы на базе ИНС с прямым распространением сигнала.
- «Классический способ» определения пород только по данным КС.
- С помощью алгоритма *k-NN.*

При применении нескольких алгоритмов МО и формировании комитета алгоритмов представляется перспективным применение мультиагентного подхода, который может обеспечить высокую степень параллелизма, возможности самоорганизации системы распознавания, а также упрощение последующей модификации и сопровождения системы.

Таким образом, в работе обоснована возможность использования методов и алгоритмов машинного обучения в задаче интерпретации данных геофизического исследования скважин на пластово-инфильтрационных месторождениях урана, разработаны и апробированы методы предварительной обработки данных, выбраны алгоритмы, релевантные поставленной задаче, и разработано программное обеспечение для автоматизированной интерпретации данных каротажа. Работа имеет существенное практическое значение в связи с большим количеством добываемого в Казахстане урана. Вместе с тем предложенная методология применения методов машинного обучения может быть использована и в обработке данных месторождений других полезных ископаемых (нефть, газ, руды).

ЛИТЕРАТУРА

1. Голышко А. В. Информационное общество тренды и перспективы // Электросвязь. – 2013. – № 4. – С. 4-9.

2. Muhamedyev R. et al. Embedded System. Almaty, IITU, 2013.- 420 pp.(in Russian).

3. Muhamedyev R. et al. Mobile programming. – Almaty: IITU, 2012. – 420 p.

4. Аджемов А.С. Теоретические границы и возможности их достижения//Электросвязь. – 2013. – N 11. – С. 15-18.

5. Скрынников В.Г. Будущий облик 5 // Электросвязь. – 2013. – № 10. – С. 34-38.

6. Erik G. Larsson, Ove Edfors, Fredrik Tufvesson, Thomas L. Marzetta Massive MIMO for Next Generation Wireless Systems // IEEE Communications Magazine. – 2014. – Vol. 52, № 2. – P. 186-195.

7. Тихвинский В.О. Концептуальные аспекты создания 5G // Электросвязь. – 2013. – № 10. – С. 29-34.

8. Тихвинский В.О., Бочечка Г.С. Перспективы сетей 5G и требования к качеству их обслуживания // Электросвязь. – 2014. – № 11. – С. 40-43.

9. Chen Min, Mao Shiwen, Liu Yunhao. Big Data: A Survey // Mobile Networks & Applications. – 2014. – Vol. 19 (2). – P. 171-209.

10. Волков Д. В поисках сокровищ. СУБД // Открытые системы. – 2014. – № 1. – С. 1.

11. Черняк Л. Серьезно о технологиях больших данных // Открытые системы. –2014. – № 1. – С. 12-15.

12. Александр Алексиянц, Антон Коршунов, Сергей Кузнецов. СУБД для социальных сетей // Открытые системы. – 2014. – № 2. – С. 7.

13. Сергей Майданов, Вадим Сухомлинов. Оценка технологий Больших Данных // Открытые системы. – 2013. – № 02. – 6 с.

14. Черняк Л. Инструменты для ковбоев // Открытые системы. – 2014. – № 1. – С. 48-50.

15. Hye-Chung Kum, Krishnamurthy A.., Machanavajjhala A., Ahalt, S.C. Social Genome: Putting Big Data to Work for Population Informatics // IEEE Computer Society. – 2014. – Vol. 47,№ 1. – P. 56-63.

16. Big Data and the promise of better government (IDC) // Proceedings of the International Scientific-Practical Conference "Smart Government: Science and Technology". – Astana, 2014. – P. 7.

17. Srinivasa N., Cruz-Albrecht, J.M. Systems of Neuromorphic Adaptive Plastic Scalable Electronics // IEEE Engineering in Medicine and Biology Society. – 2014. – Vol. 3 (1). – P. 51-56.

18. Журавлёв Ю.И. Об алгебраическом подходе к решению задач распознавания или классификации // Проблемы кибернетики. – М.: Наука, 1978. – Вып. 33. – С. 5–68.

19. Журавлев Ю.И. Об алгебраических методах в задачах распознавания и классификации. Математические методы и их применение. Распознавание. Классификация. Прогноз. – Академия наук СССР, 1988. – Вып. 1. – С. 9-16.

20. Городецкий В.И. Самоорганизация и многоагентные системы // Известия РАН. Теория и системы управления. – 2012. – № 2. – С. 92–120.

21. Городецкий В.И. Самоорганизация и многоагентные системы. Приложения и технологии разработки // Известия РАН. Теория и системы управления. – 2012. – № 3. – С. 55–75. .

22. Leite, A. R., et al. Distributed Constraint Optimization Problems: Review and perspectives // Expert Systems with Applications. – 2014. – Vol. 41. – P.13. .

23. Самсонов М. Ю., Гребешков А.Ю., Росляков А.В., Ваняшин С. В. Стандартизация Интернета вещей // Электросвязь. – 2013. – № 8. – С. 10-13.

24. Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, Marimuthu Palaniswami. Internet of Things (IoT): A vision, architectural elements, and future directions // Future Generation Computer Systems. – 2013. – Vol. 29. – P. 1645–1660.

25. K. Schwab, X. Sala-i-Martín, B. Brende. The Global Competitiveness Report 2012–20013: Full Data //World Economic Forum within the framework of The Global Benchmarking Network. - Geneva.- 527 pp.

26. Muhamedyev R I, Kalimoldaev M. N., Uskenbayeva R K . Semantic network of ICT domains and applications // Proceedings of the 2014 Conference on Electronic Governance and Open Society: Challenges in Eurasia. – New York, 2014. – P. 178-186.

27. Ravil I. Muhamedyev etc. Revelation of new ICT domains for upcoming Kazakhstan's participation//// Proceedings of the 2014 Conference on Electronic Governance and Open Society: Challenges in Eurasia. – New York, 2015. – P. 178-186.

28. A. Abdilmanova, A Khamitov, R Muhamedyev. Relationship of semantic concepts of ICT domains // Proceeding of the 13th International Scientific Conference "Information Technologies and Management ". – Riga, 2015. – P.106-108.

29. Ravil I. Muhamedyev, Yedilkhan N. Amirgaliyev, Maksat N. Kalimoldayev, Alim N. Khamitov, Ainur Abdilmanova. Selection of the most prominent lines of research in ICT domain // ICCECO. - 2015.

30. Сандра Блейксли, Джефф Хокинс «Об интеллекте». – М., СПб, Киев: Издательский дом «Вильямс», 2007. – 240 с.

31. Weiß G. Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence. – Cambridge: MIT Press, 1999. – 648 p.

32. Stuart Russell and Peter Norvig. Artificial Intelligence: A modern approach. – New Jersey: Upper Saddle River, 2010. – 1078 p.

33. M. Tim Jones. Artificial Intelligence: A Systems Approach. – Hingham, Massachusetts, New Delhi: INFINITY SCIENCE PRESS LLC, 2008. – 500 p.

34. Machine learning. http://en.wikipedia.org/wiki/Machine_learning 10.10.2014.

35. Т. Кохонен. Ассоциативная память. – М.: Мир, 1980. – 240 с.

36. Нейрокомпьютеры: Учеб. Пособие для вузов.- М.: Изд-во МГТУ им. Н.Э. Баумана, 2004. – 320 с.

37. Minsky M. L., Papert S.A. Perceptrons. An Introduction to Computational Geometry. – MIT, 1969. – 252 p.

38. Python for artificial Intelligence https://wiki.python.org: 3.12.2014.

39. Rapidminer https://rapidminer.com: 10.08.2015.

40. Weka. The University of Waikato http://www.cs.waikato.ac.nz/ml/weka: 5.06.2015.

41. Van der Baan, M. and Jutten, C. Neural networks in geophysical applications // Geophysics. – 2000. – № 65(4). – P. 1032-1047.

42. Baldwin, J. L, R. M. Bateman and C. L. Wheatley. Application of a neural network to the problem of mineral identification from well logs // The Log Analyst. – 1990. – № 3. – P. 279-293.

43. Benaouda B., Wadge G., Whitmark R.B., Rothwell R. G., MacLeod C. Inferring the lithology of borehole rocks by applying neural network classifiers to downhole logs - an example from the Ocean Drilling Program // Geophysical Journal International. – 1999.

44. Saggaf M. M., Nebrija Ed. L. Estimation of missing logs by regularized neural networks //AAPG Bulletin. – 2003. – № 8. – P. 1377-1389.

45. В.А. Тененёв, Б.А. Якимович, М.А. Сенилов, Н.Б. Паклин. Интеллектуальные системы интерпретации геофизических исследований скважин // Штучний інтелект. – 2002. – № 3. – С.338.

46. Klaus Yelbig and Sven Treitel. Computational Neural Networks For Geophysical Data Processing. Editor: Mary M. Poulton. – 2001. – 335 p.

47. M. Borsaru, B. Zhou, T. Aizawa, H. Karashima, T. Hashimoto. Automated lithology prediction from PGNAA and other geophysical logs // Applied Radiation and Isotopes. – 2006. – № 64. – P. 272–282.

48. Rogers S.J., Chen H.C., Kopaska-Merkel D.C.t Fang J.H. Predicting permeability from porosity using artificial neural networks //AAPG Bulletin. – 1995. – P. 786-1797.

49. Kapur L., Lake L., Sepehrnoori K., Herrick D., Kalkomey C. Facies prediction from core and log data using artificial neural network technology // Transactions of the 39th Society of Professional Well Log Analysts Annual Logging Symposium. – 1998. – P.1.

50. Алёшин С.П., А.Л. Ляхов. Нейросетевая оценка минерально-сырьевой базы региона по данным геофизического мониторинг // Нові технології № 1 (31). – 2011. – С. 39-43.

51. Rogers S J, Fang J H, Karr C L, and Stanley D A. Determination of lithology from well logs using a neural network // AAPG Bulletin. – 1992. – № 76(5). – P. 731–739.

52. Костиков Д.В. Инструментальные средства интерпретации геофизических исследований скважин на основе преобразованных каротажных диаграмм с помощью многослойной нейронной сети: диссертация к.т.н. – М: РГБ, 2007. – 189 с.

53. R. Muhamediyev, E. Amirgaliev, S. Iskakov, Y. Kuchin, E. Muhamedyeva. Integration of Results of Recognition Algorithms at the Uranium Deposits // Journal of ACIII. – 2014. – Vol. 18, № 3. – P. 347-352.

54. Амиргалиев Е.Н., Искаков С.Х., Кучин Я.В., Мухамедиев Р.И. Интеграция алгоритмов распознавания литологических типов // Проблемы информатики. Сибирское отделение РАН. – 2013. – № 4 (21). – С. 11-20.

55. Амиргалиев Е.Н., Искаков С.Х., Кучин Я.В., Мухамедиев Р.И. Методы машинного обучения в задачах распознавания пород на урановых месторождениях //Известия НАН РК. – 2013. – № 3. – С.82-88.

56. Яшин С.А. Подземное скважинное выщелачивание урана на месторождениях Казахстана // Горный журнал. – 2008. – № 3. – С. 45-49.

57. "Development of methods of data boreholes interpretation by using artificial neural network" (On request of "Geotehnoserviss" ltd) (in russian – «Разработка методики для интерпретации данных ГИС с помощью нейронных сетей» (2011). .

58. Методические рекомендации по комплексу геофизических методов исследования скважин при подземном выщелачивании урана. – Алматы: ЗАО НАК «Казатомпром». ТОО ИВТ, 2003. – 36 с.

59. Техническая инструкция по проведению геофизических исследований в скважинах на пластово инфильтрационных месторождениях урана. – Алматы: ТОО ГРК, 2010. – 44 с.

60. Мухамедиев Р. И., Кучин Я. И. Средства автоматизации обработки данных геофизического исследования скважин на месторождениях урана пластово-инфильтрационного типа // Электронный журнал Cloud of Science. – 2015. – Т. 2, № 3 – P. 13 (принято к печати).

61. Кучин Я.И. Система комплексной интерпретации результатов геофизических исследований скважин на пластово-инфильтрационных месторождениях урана. – Алматы: Вестник Академии Наук, 2008. – 12 с.

62. David Kriesel. A Brief Introduction to Neural Networks. http://www.dkriesel.com/en/science/neural_networks: 3.09.2015.

63. Guoqiang Peter Zhang. Neural Networks for Classification: A Survey // IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART C: APPLICATIONS AND REVIEWS, 2000. – Vol. 30. – №. 4.

64. Joseph A. Cruz and David S. Wishart. Applications of Machine Learning in Cancer Prediction and Prognosis // Cancer Informatics. - 2006. - V. 2. P. 59–77.

65. Shoeb, Ali H., and John V. Guttag. Application of machine learning to epileptic seizure detection // International Conference on Machine Learning. - 2010. - P. 975-982.

66. Mannini, Andrea, and Angelo Maria Sabatini. Machine learning methods for classifying human physical activity from on-body accelerometers// Sensors 10. V. 2. - 2010. P. 1154-1175.

67. Ballester, Pedro J., and John BO Mitchell. A machine learning approach to predicting protein–ligand binding affinity with applications to molecular docking// Bioinformatics 26. V. 9. - 2010. P. 1169-1175.

68. Farrar, Charles R., and Keith Worden. Structural health monitoring: a machine learning perspective.- John Wiley & Sons. - 2012. - 66 pp.

69. Frederich Recknagel. Application Of macine Learning To Ecological Modelling // Ecological Modelling. V. 146. -.2001. P. 303-310.

70. Clancy, Charles, Joe Hecker, Erich Stuntebeck, and Tim O. Shea. Applications of machine learning to cognitive radio networks // Wireless Communications, IEEE 14. V. 4. - 2007. - P. 47-52.

71. Ball, Nicholas M., and Robert J. Brunner. Data mining and machine learning in astronomy // Journal of Modern Physics D 19. V.07. - 2010.- P. 1049-1106.

72. Hastie T., Tibshirani R., Friedman J. Unsupervised learning // Springer New York, 2009. – P. 485-585.

73. Kotsiantis, Sotiris B., I. Zaharakis, and P. Pintelas. Supervised machine learning: A review of classification techniques.Emerging artificial Intelligence Applications in Computer Engineering. - IOS Press, 2007. - P. 3-24.

74. Csaba Szepesv´ari. Algorithms for Reinforcement Learning. Synthesis Lectures on Artificial Intelligence and Machine Learning. - Morgan & Claypool Publishers, 2009. - P. 98. .

75. Xiaojin Zhu. Semi-Supervised Learning Literature Survey // Computer Sciences. –2008. – P. 60-86.

76. Kohonen, Teuvo. Self-Organized Formation of Topologically Correct Feature Maps // Biological Cybernetics. – Vol. 43 (1). – P. 59–69.

77. A.K. Jain, M.N. Murty, P.J. Flynn. Data Clustering: A Review // ACM Computing Surveys. – Vol. 31, №. 3. – P. 264-323.

78. Wesam Ashour Barbakh, Ying Wu, Colin Fyfe. Review of Clustering Algorithms. Non-Standard Parameter Adaptation for Exploratory Data Analysis // Studies in Computational Intelligence. – 2009. – Vol. 249. – P. 7-28.

79. Taiwo Oladipupo Ayodele. Types of Machine Learning Algorithms // New Advances in Machine Learning. – 2010. – P. 19-48.

80. Hamza Awad Hamza Ibrahim et al. Taxonomy of Machine Learning Algorithms to classify realtime Interactive applications // International Journal of Computer Networks and Wireless Communications. – 2012. – Vol. 2, № 1. – P. 69-73.

81. Мухамедиев Р. И., Мухамедиева Е. Л., Кучин Я. И. Таксономия методов машинного обучения и оценка качества классификации и обучаемости //Cloud of Science. Т. 2. № 3.– С. 359-378.

82. Muhamedyev R. Machine learning methods: An overview //CMNT. - 19(6). - 2015. - p. 14-29.

83. Serrano-Gotarredona T., Linares-Barranco B., Andreou A. G. Adaptive Resonance Theory Algorithms //Adaptive Resonance Theory Microchips // Springer US, 1998. – P. 1-38.

84. Carpenter G. A., Grossberg S., Rosen D. B. Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system // Neural networks. – 1991. – Т. 4. – №. 6. – С. 759-771.

85. Kohonen T. The self-organizing map //Neurocomputing. – 1998. – V. 21. – №. 1. – P. 1-6.

86. Bishop C. M., Svensén M., Williams C. K. I. GTM: The generative topographic mapping //Neural computation. – 1998. – V. 10. – №. 1. – P. 215-234.

87. Jain A. K. Data clustering: 50 years beyond K-means //Pattern recognition letters. – 2010. – V. 31. – №. 8. – P. 651-666.

88. Дьяконов А. Г. Анализ данных, обучение по прецедентам, логические игры, системы WEKA, RapidMiner и MatLab (Практикум на ЭВМ кафедры математических методов прогнозирования): учебное пособие. – М.: Изд. отдел факультета ВМК МГУ им. М.В. Ломоносова, 2010.

89. Martin Fodslette Møller. A scaled conjugate gradient algorithm for fast supervised learning // Neural Networks. – 1993. – Vol. 6, Issue 4. – P. 525-533.

90. Dong C. Liu, Jorge Nocedal. On the limited memory BFGS method for large scale optimization // Mathematical Programming. – 1989. – Vol. 45, Issue 1-3. – P. 503-528.

91. Warren S. McCulloch, Walter Pitts. A logical calculus of the ideas immanent in nervous activity // The bulletin of mathematical biophysics. – 1943. – Vol. 5, Issue 4. – P. 115-133.

92. Rosenblatt, F. The perceptron: A probabilistic model for information storage and organization in the brain // Psychological Review. – 1958. – Vol. 65 (6). – P. 386-408.

93. Marvin Minsky, Seymour Papert. Perceptrons, expanded edition. The MIT Press, 1987. – 308 p.

94. Werbos P. Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences. – Harvard University, 1974. – 38 p.

95. Werbos P.J. Backpropagation: past and future // IEEE International Conference on Neural Networks. – San Diego, 1988. – Vol. 1. – P. 343-353. .

96. David Saad. Introduction. On-Line Learning in Neural Networks. – Cambridge University Press, 1998. – P. 3-8.

97. Cybenco G. Approximation by superpositions of a sigmoidal function // Mathematics of Control Signals and Systems. – 1989. – Vol. (4). – P. 304-314.

98. Hornik K. et al. Multilayer feedforward networks are universal approximators // Neural Networks. – 1989. – Vol. 2. – P. 359-366.

99. Галушкин А.И. Решение задач в нейросетевом логическом базисе // Нейрокомпьютеры: разработка, применение. – Москва: Радиотехника, 2006. – № 2. – С. 49-71.

100. Галушкин А.И. Нейронные сети: основы теории. – Горячая линия – Телеком, 2010. – 496 с.

101. Ясницкий Л.Н. Введение в искуственный интеллект: учебное пособие для вузов. – М.: Академия, 2008. – 176 с.

102. Negnevitsky. M. Artificial Intelligence: A Guide to Intelligent Systems. - Harlow, England, 2005.- 415 pp.

103. Dudani, Sahibsingh A. The Distance-Weighted k-Nearest-Neighbor Rule // Systems, Man and Cybernetics. – 1976. – Vol. SMC-6, Iss. 4. – P. 325-327.

104. K-nearest neighbor algorithm http://en.wikipedia.org/wiki/K-nearest_neighbor_algorithm: 5.07.2012.

105. Support vector machine // http://en.wikipedia.org/wiki/Support_vector_machine: 22.02.2012.

106. Linear discriminant analysis. http://en.wikipedia.org/wiki/Linear_discriminant_analysis: 10.11.2012.

107. Dudoit S., Fridlyand J., Terence P. Comparison of Discrimination Methods for the Classification of Tumors Using Gene Expression // Data Journal of the American Statistical Association. – 2002. – Vol. 97, Issue 457. – P. 77-87.

108. Cohen J. A coefficient of agreement for nominal scales // Educational and Psychological Measurement. – 1960. – P. 37-46.

109. RapidMiner and RapidAnalytics. http://www.rapid-i.com/downloads/brochures/RapidMiner_Fact_Sheet.pdf : 11.09.2014.

110. David M W Powers. The Problem with Kappa // Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics. – Avignon, France, 2012. – P. 345–355.

111. Амиргалиев Е.Н., Искаков С.А, Кучин Я. В., Мухамедиев Р.И., Мухамедиева Е.Л. Сравнительная оценка качества распознавания литологических слоев на урановых месторождениях // Межд. науч.-практ. конф. «ИКТ: Наука, образ., инновации». - Алматы, 2013. - 9 с.

112. Е. М. Миркес. Нейрокомпьютер. Проект стандарта. – Новосибирск: Наука, Сибирская издательская фирма РАН, 1998. – 337 с.

113. Léon Bottou. Large-Scale Machine Learning with Stochastic Gradient Descent // Proceedings of COMPSTAT. – 2010. – P. 177-186.

114. Leon Bottou. Online learning and stochastic approximation // On-Line Learning in Neural Networks. Edited by David Saad. – Cambridge University Press, 1998. – P. 9-43.

115. Cheng-Tao Chu etc. Map-Reduce for Machine learning on multicore. Advances in Neural Information Processing Systems // Proceedings of the 2006 Conference. – MIT Press, 2007. – P. 281-310.

116. Jure Leskovec, Anand Rajaraman and Jeffrey David Ullman. Mining of Massive Datasets. – Cambridge University Press, 2014. – 476 p.

117. Ghoting, Amol, Rajasekar Krishnamurthy, Edwin Pednault, Berthold Reinwald, Vikas Sindhwani, Shirish Tatikonda, Yuanyuan Tian, and Shivakumar Vaithyanathan. SystemML: Declarative machine learning on MapReduce // InData Engineering. - 2011. - P.231-242. .

118. Kraska, Tim, Ameet Talwalkar, John C. Duchi, Rean Griffith, Michael J. Franklin, and Michael I. Jordan. MLbase: A Distributed Machine-learning System // Conference on Innovative Dta Systems Research (CIDR). – 2013. – P. 7-9.

119. Kuchin Y., Muhamedyev R., Muhamedyeva L. Analysis of the data of geophysical research by means neuronet methods // VIII international scientific conference on Electronics And Computer technologies, IKECCO'. – Almaty: SDU, 2011. – P. 289-297.

120. Muhamediyev R., Kuchin Y., Muhamedyeva E., Nurushev Z., Yakunin K., Gricenko P. Методика подготовки данных ГИС для анализа нейросетевыми методами // II Intern. scientific-practical conference Information-innovation technologies.- Almaty, 2011. -С.375-383.

121. Muhamediyev R., Kuchin Y., etc. The analysis of the data of geophysical research of boreholes by means of artificial neural networks // Fourth International Conference "Informatics in Scientific Knowledge". - Varna, 2012. - P.198-206.

122. Muhamediyev R., Kuchin Y, Gricenko P., Muhamedyeva Elena. Recognition of Geological Rocks At the Bedded-infiltration Uranium Fields by Using Neural Networks // IEEE Conference on Open Systems. – Kuala Lumpur, 2012. – P. 102-107.

123. Амиргалиев Е.Н., Кучин Я. В., Мухамедиев Р.И. и другие. Оценка качества нейросетевого распознавания литологических слоев на урановых месторождениях // Матер. науч.-практ. конф. «Актуальные проблемы информатики и процессов управ.». – Алматы, 2012.- 9с. .

124. Kostikov D.V. Tools of interpretation of geophysical researches of boreholes on the basis of the transformed logging diagrams by means of a multilayered neural network. – M.: Russian state library, 2007. – P. 189.

125. Alyuda http://www.alyuda.com/companyinfo.htm: 10.09.2012.

126. Muhamediyev R., Kuchin Y., Muhamedyeva E. Geophysical Research of Boreholes: Artificial Neural Networks Data Analysis // The 6th International Conference on Soft Computing and Intelligent Systems. – Kobe, 2012. – P. 825-829.

127. Bektemyssova G. et al. Construction of recognition system at the uranium production process //Control, Automation and Systems (ICCAS), 2014 14th International Conference on. – IEEE, 2014. – С. 1462-1465.

128. Kuchin Y., Muhamedyev R., Muhamedyeva L. Analysis of the data of geophysical research by means neuronet methods // VIII international scientific

conference on Electronics And Computer technologies, IKECCO'. – Almaty: SDU, 2011. – P. 289-297. .

129. Демидова Л. А., Никульчев Е. В. , Соколова Ю. С. Классификация больших данных: использование SVM-ансамблей и SVM-классификаторов с модифицированным роевым алгоритмом// Cloud of Science.- Т.3.-N 1.- 2016. С.5-42.

130. Muhamedyev R., Iskakov S., Gricenko P., Yakunin K., Y. Kuchin. Integration of results from Recognition Algorithms and its realization at the uranium production process // 8th IEEE International Conference AICT. – Astana, 2014. – P. 188-191.

131. Ovidiu Vermesan, Peter Friess Internet of Things: Converging Technologies for Smart Environments and Integrated Ecosystems. – Denmark: River Publishers, 2013. – 383 p.

132. Muhamedyev R. et al. Mobile programming. – Almaty: IITU, 2012. – P. 420.

133. Muhamedyev R. et al. Comparative analysis of classification algorithms //Application of Information and Communication Technologies (AICT), 2015 9th International Conference on. – IEEE, 2015. – C. 96-101.

ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

CCR – Correct Classification Rate

DLDA – Diagonal Linear Discriminant Analysis

IoT – Internet of Things

k-NN – алгоритм k ближайших соседей

LDA – Linear Discriminant Analysis

LDAC – Linear Discriminant Analysis Classifier (классификатор на основе линейного дискриминантного анализа)

Linear SVM и Non-linear SVM – линейный и нелинейный SVM

M2M – Machine-to-Machine

MACS – Multi-Agent Classification System

MBGD – Mini-Batch Gradient Descent

RFID – Radio Frequency Communication

SDN – Software-Defined Networking

SGD – Stochastic Gradient Descent

SIPP – Simple Integrator of Post-processing Stage

SL – supervised learning

Strong AI – сильный искусственный интеллект

SVM – Support Vector Machine Classification (классификатор, основанный на методе опорных векторов)

UL – unsupervised learning

Weak AI – слабый искусственный интеллект

БД – база данных

ГИС – геофизические исследования в скважинах

ГК – гамма-каротаж

ЕРЭ – естественные радиоактивные элементы

ИИ (AI) – искусственный интеллект (Artificial Intelligence)

ИК (IK) – индукционный каротаж

ИКТ – информационно-коммуникационные технологии

ИНС (ANN, NN) – искусственная нейронная сеть

КМ – кавернометрия

КС (KS) – каротаж сопротивлений

МО – машинное обучение

ПС (PS) – каротаж потенциалов самопроизвольной поляризации

ПСВ – подземное скважинное выщелачивание

СУБД – система управления базами данных

ГЕОФИЗИЧЕСКИЕ ОПРЕДЕЛЕНИЯ И ТЕРМИНЫ

Автономный скважинный прибор – скважинный прибор, содержащий измерительное, регулирующее или индикаторное устройства, а также источник питания.

Боковое каротажное зондирование – каротаж сопротивления с использованием нескольких однотипных зондов разной длины.

Вмещающие породы – породы, контактирующие с каротажным пластом или рудным телом.

Гамма-каротаж (ГК) – радиоактивный каротаж, основанный на измерении естественной гамма-активности горных пород.

Глинистая корка в скважине – слой глинистых частиц, оседающих на стенке скважины в результате фильтрации промывочной жидкости в пласт.

Длина индукционного зонда – расстояние между серединами главной генераторной и измерительной катушек индукционного каротажного зонда.

Длина радиоактивного каротажного зонда – расстояние между серединами источника и детектора излучений радиоактивного каротажного зонда.

Зона проникновения фильтрата промывочной жидкости – часть пласта, в которую проник фильтрат промывочной жидкости.

Индукционный каротаж (ИК) – электромагнитный каротаж, основанный на измерении кажущейся удельной электропроводности горных пород.

Индукционный каротажный зонд – электромагнитный каротажный зонд, содержащий две или более катушек индуктивности, расположенных на оси скважины. В зависимости от числа катушек индукционные каротажные зонды могут быть двух-, трехкатушечными и т.д.

Интерпретация данных каротажа – обработка результатов геофизических исследований в скважинах с целью изучения геологического разреза, выделения и оценки полезных ископаемых.

Кавернометрия скважины – измерение среднего диаметра скважины.

Каротаж – геофизические исследования в скважинах с целью изучения вскрытого скважиной геологического разреза и выявления полезных ископаемых.

Каротаж потенциалов самопроизвольной поляризации (ПС) – электрический каротаж, основанный на измерении потенциалов самопроизвольной поляризации.

Каротаж сопротивления (КС) – электрический каротаж, основанный на измерении кажущегося удельного электрического сопротивления горных пород.

Каротажная кривая – график изменения каротажных значений по скважине.

Каротажные значения – значения измеряемой при каротаже величины в точках скважины.

Каротажный пласт – прослой или несколько смежных прослоев, объединенных по близким каротажным значениям в соответствии с заданными критериями.

Комплексный скважинный прибор – скважинный прибор, предназначенный для проведения геофизических исследований несколькими методами.

Линия глин – линия, проведенная по участкам кривой самопроизвольной поляризации, соответствующим пластам глин.

Межскважинные исследования – геофизические исследования в скважинах с целью изучения массива горных пород в межскважинном пространстве, поиска и разведки месторождений полезных ископаемых и решения инженерно-геологических задач.

Мощный пласт – каротажный пласт, каротажные значения против которого близки к значениям против пласта бесконечной мощности.

Нейтронный каротаж (НК) – радиоактивный каротаж, основанный на измерении характеристик нейтронного излучения, сопровождающего распад естественных радиоактивных элементов в горных породах.

Околоскважинные исследования – геофизические исследования в скважинах с целью изучения массива горных пород в околоскважинном пространстве, поиска и разведки месторождений полезных ископаемых и решения инженерно-геологических задач.

Опорный пласт – каротажный пласт с известной физической характеристикой.

Пласт бесконечной мощности – каротажный пласт, при дальнейшем увеличении мощности которого значения на каротажной кривой не изменяются.

Пласт высокого сопротивления – каротажный пласт, удельное электрическое сопротивление которого больше удельного электрического сопротивления вмещающей среды.

Пласт низкого сопротивления – каротажный пласт, удельное электрическое сопротивление которого меньше удельного электрического сопротивления вмещающей среды.

Плотностной гамма-гамма-каротаж (ПГГК) – гамма-гамма-каротаж, основанный на измерении жесткой составляющей рассеянного гамма-излучения.

Приведенные каротажные значения – каротажные значения, приведенные к заданным условиям.

Промытая зона – ближайшая к скважине часть зоны проникновения.

Прослой – геологическое тело, однородное по изучаемому физическому свойству, ограниченное двумя поверхностями раздела, которые в пределах рассматриваемой области можно считать параллельными.

Радиоактивный каротаж – каротаж, основанный на измерении характеристик полей ионизирующих излучений.

Радиоактивный каротажный зонд – каротажный зонд ионизирующего излучения, применяемый в скважинной аппаратуре радиоактивного каротажа.

Расчленение разреза скважин – установление последовательности залегания пластов и определение их границ по данным каротажа.

Рудное тело – естественное скопление руды произвольной формы в земной коре, по своим физическим характеристикам отличающееся от вмещающих пород.

Скважинный прибор – прибор, предназначенный для проведения геофизических исследований в скважине.

Стандартный электрический каротаж – каротаж потенциалов самопроизвольной поляризации и каротаж сопротивления потенциал-зондом и (или) градиент-зондом, длину которых устанавливают в соответствии с геолого-геофизическими условиями района.

Существенные каротажные значения – каротажные значения против пласта, используемые при интерпретации каротажных кривых.

Токовый каротаж (ТК) – электрический каротаж, основанный на измерении сопротивления заземления электродов.

Тонкий пласт – каротажный пласт, на каротажные значения против которого влияют физические свойства соседних пластов.

Фильтрат промывочной жидкости – промывочная жидкость, отфильтровавшаяся в пласт.

Электрический каротаж – каротаж, основанный на измерении характеристик электрического поля, возникающего самопроизвольно или создаваемого искусственно.

Электрический каротажный зонд – каротажный зонд, содержащий измерительные и (или) токовые электроды и применяемый в скважинной аппаратуре электрического каротажа. В зависимости от назначения различают: зонд самопроизвольной поляризации, зонд электродных потенциалов, зонд сопротивления.

ОПРЕДЕЛЕНИЯ И ТЕРМИНЫ ФИЗИЧЕСКИХ СВОЙСТВ И ПАРАМЕТРОВ ОБЪЕКТОВ ИНТЕРПРЕТАЦИИ

Видимая мощность пласта – расстояние между точками пересечения скважины с кровлей и подошвой пласта.

Глинистость горной породы – совокупность глинистых включений в горной породе.

Диаметр зоны проникновения – диаметр однородного концентрического слоя, эквивалентного по влиянию на кажущееся сопротивление зоне проникновения.

Истинная мощность – кратчайшее расстояние между кровлей и подошвой пласта.

Потенциал самопроизвольной поляризации в скважине – потенциал, созданный в скважине токами самопроизвольной поляризации. Включает в себя диффузионный, диффузионно-абсорбционный и фильтрационный потенциал.

Самопроизвольная поляризация в скважине – самопроизвольное образование поля электрических токов в скважине и вблизи нее.

Удельное электрическое сопротивление горной породы – сопротивление горной породы проходящему через нее электрическому

току, отнесенное к единице поперечного сечения и длины образца породы.

Удельное электрическое сопротивление зоны проникновения – удельное электрическое сопротивление однородного концентрического слоя, эквивалентного по влиянию на кажущееся сопротивление зоне проникновения.

Удельное электрическое сопротивление пласта – удельное электрическое сопротивление части пласта, не затронутой проникновением промывочной жидкости.

Эффективная мощность пласта – суммарная мощность проницаемых прослоев в пласте.

Эффективная плотность каротажного пласта – разность плотностей каротажного пласта и вмещающих пород.

ОПРЕДЕЛЕНИЯ И ТЕРМИНЫ ИНТЕРПРЕТАЦИИ ДАННЫХ МЕТОДАМИ МАШИННОГО ОБУЧЕНИЯ

Идеальная скважина – набор данных, составленных из данных разных скважин так, чтобы обеспечить примерно равное представительство разных пород.

ИНС (ANN) – искусственные нейронные сети, в контексте данной работы – сети с прямым распространением сигнала (Feedforward Neural Networks).

Искусственный интеллект (ИИ) (Artificial Intelligence – AI) – область исследований, посвященных созданию компьютеров и программ с интеллектуальным поведением.

Качество классификации – семейство показателей, предназначенных для оценки качества систем машинного обучения и иных систем классификации. Включает показатели: *доля правильно классифицированных объектов, точность, полнота, ошибки классификации (classification error)* и т.п.

Матрица ошибок (error matrix/confusion matrix) – матрица, в которой на главной диагонали расположены правильные ответы, а цифры вне главной диагонали представляют собой ошибочные результаты, причем n_{ij} – количество объектов, классифицированных экспертом как объект класса j, а системой – как объект класса i.

Машинное обучение (МО) (Machine Learning – ML) – раздел искусственного интеллекта, рассматривающий методы построения алгоритмов и программ, способных обучаться.

Ошибка обучения – термин, косвенно определяющий качество обучения системы МО. Низкое значение ошибки обучения, приближающееся к 0%, как правило, означает «переобучение». Переобученная система способна распознавать только примеры из обучающей выборки. Для обеспечения возможности распознавания примеров, не входящих в обучающую выборку, система МО обучается до некоторого уровня ошибки обучения.

Ошибки классификации (classification error) – относительное количество ошибочно классифицированных примеров.

Показатель Карра – показатель, служащий для сравнения рейтингов в дихотомических (бинарных) задачах классификации.

Постобработка – интерпретация, проверка, корректировка или иная обработка результатов работы системы МО.**«Сырая» скважина** – скважина, данные которой никаким образом не использовались при обучении нейронной сети. Таким образом, процент правильных результатов по «сырой» скважине или результат по «сырой» скважине – один из основных параметров для оценки результативности, качества обучения, архитектуры, нормировки и других методов предобработки и классификации данных.

ПРЕДМЕТНЫЙ УКАЗАТЕЛЬ[9]

[9] Приведенные термины как на русском, так и на английском языке могут отличаться от общепринятых и представлены исключительно для удобства поиска.

ПРИЛОЖЕНИЕ 1

Коды литотипов

Название литотипа	Код	Альтернативное название (принятое при разведке)
алевролит	6	алевриты
песок мелкозернистый	4	пески мелкозернистые, мелкозернистые до тонкозернистых
песок среднезернистый	3	пески среднезернистые
песок разнозернистый	123	пески разнозернистые
гравий,галька	1	песчано-гравийные отложения
плотные т-м-з пески	44	пески тонкозернистые
песок крупнозернистый	112	пески крупно-грубозернистые с гравием до 20%
глина	7	глина
известняк	9	карбонатные породы
песок м-з глинистый	47	пески мелкозернистые глинистые
песок разнозернистый с гравием	12	пески разнозернистые с гравием (>20%)
алевропесчаник	64	алевриты песчаные
доломит	9	карбонатные породы
лигниты	907	углистые и глинисто-углистые прослои
глина запесоченная	74	глина песчаная
палеозойские породы	pl	палеозойские породы
песчаник на карбонатном цементе	59	песчаники на карбонатном цементе
алевролит глинистый	67	алевриты глинистые
гипс	81	гипсы
песок м-з с катунами	457	пески мелкозернистые с катунами и обрывками глин

Название литотипа	Код	Альтернативное название (принятое при разведке)
песок м-з алевритистый	46	пески мелкозернистые алевритистые
слабо проницаемые породы	PN	пески мелкозернистые глинистые
проницаемые породы	P	пески среднезернистые
непроницаемые породы	N	песчаники разнозернистые с гравием на неясном цементе
песчаник глинистый	57	песчаник на глинистом цементе
песок м-з глинистый	517	пески разнозернистые глинистые (>30% глин)
т-м-з пески	477	пески мелкозернистые, мелкозернистые до тонкозернистых
гравелиты	96	гравелиты

ПРИЛОЖЕНИЕ 2

Сравнительная оценка точности распознавания экспертами

D, L и T - условные обозначения экспертов, kern - данные усредненного кернового опробирования. 1, 2, 3, 4, 5, 7 - номера литологических типов.

Скважина 2100

D vs L		D						
		true1	true3	true4	true5	true6	true7	Precision
L	pred1	226	0	3	9	0	0	94.96%
	pred3	29	102	5	0	0	6	71.83%
	pred4	0	36	64	0	0	1	63.37%
	pred5	1	0	0	0	0	2	0.00%
	pred6	0	0	0	0	0	0	0.00%
	pred7	1	5	2	0	0	21	72.41%
	recall	87.94%	71.33%	86.49%	0.00%	0.00%	70.00%	413

Accuracy	0.81
Kappa	0.70
WMRecall	0.63
WMPrecision	0.61
T1Score	0.618018

D vs T		D						
		true1	true3	true4	true5	true6	true7	Precision
T	pred1	235	3	0	0	0	7	98.74%
	pred3	28	107	0	0	0	0	75.35%
	pred4	0	101	0	0	0	0	0.00%
	pred5	3	0	0	0	0	0	0.00%
	pred6	0	0	0	0	0	21	72.41%
	pred7	5	3	0	0	0	75.00%	363
	recall	86.72%	50.00%	0.00%	0.00%	0.00%		

Accuracy	70.76
Kappa	0.54
WMRecall	42.34
WMPrecision	49.30
T1Score	45.56

L vs T

T \ L	true1	true3	true4	true5	true6	true7	Precision
pred1	251	2	0	0	0	4	97.67%
pred3	3	137	0	0	0	3	95.80%
pred4	2	72	0	0	0	0	0.00%
pred5	9	0	0	0	0	0	0.00%
pred6	0	0	0	0	0	0	0.00%
pred7	6	3	0	0	0	21	70.00%
recall	92.62 %	64.02 %	0.00%	0.00%	0.00%	75.00 %	409

Accuracy	0.80
Kappa	0.67
WMRecall	46.33%
WMPrecision	52.69%
T1Score	0.493058

kern vs D

D \ skern	true1	true3	true4	true5	true6	true7	Precision
pred1	202	20	0	2	0	1	89.78%
pred3	16	19	0	0	0	1	52.78%
pred4	1	43	0	0	0	0	0.00%
pred5	0	1	0	0	0	1	0.00%
pred6	0	0	0	0	0	0	0.00%
pred7	3	11	0	1	0	7	31.82%
recall	90.99 %	20.21 %	0.00%	0.00%	0.00%	70.00 %	228

Accuracy	0.693
Kappa	0.39
WMRecall	0.3624
WMPrecision	0.3487
T1Score	0.355418

kern vs L

L \ skern	true1	true3	true4	true5	true6	true7	Precision
pred1	205	13	0	5	0	2	91.11%
pred3	33	1	0	0	0	2	2.78%
pred4	1	43	0	0	0	0	0.00%
pred5	1	0	0	0	0	1	0.00%
pred6	0	0	0	0	0	0	0.00%
pred7	8	8	0	0	0	6	27.27%
recall	82.66 %	1.54%	0.00%	0.00%	0.00%	54.55 %	212

Accuracy	0.6444
Kappa	0.23
WMRecall	0.2775
WMPrecision	0.2423
T1Score	0.258708

kern vs T

T \ skern	true1	true3	true4	true5	true6	true7	Precision
pred1	207	14	0	0	0	4	92.00%
pred3	34	1	0	0	0	1	2.78%
pred4	2	42	0	0	0	0	0.00%
pred5	2	0	0	0	0	0	0.00%
pred6	0	0	0	0	0	0	0.00%
pred7	11	5	0	0	0	6	27.27%
recall	80.86%	1.61%	0.00%	0.00 %	0.00%	54.55 %	214

Accuracy	0.6505
Kappa	0.21
WMRecall	0.274
WMPrecision	0.2441
T1Score	0.258187

D vs L

L \ D	true1	true3	true4	true5	true6	true7	Precision
pred1	33	96	2	0	0	0	25.19%
pred3	4	35	264	0	0	8	11.25%
pred4	0	0	43	0	0	9	82.69%
pred5	5	0	0	5	0	1	45.45%
pred6	0	0	0	0	0	0	0.00%
pred7	1	2	13	0	0	85	84.16%
recall	76.74 %	26.32%	13.35 %	100.00%	0.00%	82.52 %	201

Accuracy	0.3317
Kappa	0.16
WMRecall	0.5979
WMPrecision	0.4975
T1Score	0.543099

D vs T

T \ D	true1	true3	true4	true5	true6	true7	Precision
pred1	94	30	7	0	0	0	71.76%
pred3	31	243	35	0	0	2	78.14%
pred4	0	8	44	0	0	0	84.62%
pred5	7	0	0	3	0	1	27.27%
pred6	0	0	0	0	0	0	0.00%
pred7	1	5	12	0	0	83	82.18%
recall	70.68%	84.97 %	44.90%	100.00%	0.00%	96.51 %	467

Accuracy	0.7706
Kappa	0.66
WMRecall	0.7941
WMPrecision	0.6879
T1Score	0.737195

L vs T

T \ L	true1	true3	true4	true5	true6	true7	Precision
pred1	39	0	4	0	0	0	90.70%
pred3	88	38	7	0	0	0	28.57%
pred4	4	234	73	0	0	11	22.67%
pred5	2	0	0	3	0	0	60.00%
pred6	0	0	0	0	0	0	0.00%
pred7	0	14	14	0	0	75	72.82%
recall	29.32%	13.29 %	74.49%	100.00%	0.00%	87.21 %	228

Accuracy	0.3762
Kappa	0.19
WMRecall	0.6086
WMPrecision	0.5495
T1Score	0.577542

kern vs D

D \ skern	true1	true3	true4	true5	true6	true7	Precision
pred1	14	10	0	3	0	1	50.00%
pred3	34	22	0	0	0	4	36.67%
pred4	54	253	36	8	0	26	9.55%
pred5	5	1	0	0	0	0	0.00%
pred6	5	0	12	0	0	36	0.00%
pred7	19	25	4	0	0	34	41.46%
recall	10.69 %	7.07 %	69.23 %	0.00 %	0.00 %	33.66 %	106

Accuracy	0.1749
Kappa	0.04
WMRecall	0.2011
WMPrecision	0.2295
T1Score	0.214363

kern vs L

		skern						
		true1	true 3	true 4	true 5	true 6	true7	Precision
L	pred1	13	5	10	0	0	0	46.43%
	pred3	4	47	5	0	0	4	78.33%
	pred4	10	62	275	5	0	25	72.94%
	pred5	4	0	2	0	0	0	0.00%
	pred6	0	5	8	0	0	40	0.00%
	pred7	12	14	22	0	0	34	41.46%
	recall	30.23%	35.34%	85.40%	0.00%	0.00%	33.01%	369

Accuracy	0.6089
Kappa	0.37
WMRecall	0.3066
WMPrecision	0.3986
T1Score	0.346599

kern vs T

		skern						
		true 1	true 3	true 4	true 5	true 6	true7	Precision
T	pred1	18	10	0	0	0	0	64.29%
	pred3	44	12	0	0	0	4	20.00%
	pred4	39	242	72	3	0	21	19.10%
	pred5	0	0	6	0	0	0	0.00%
	pred6	4	5	8	0	0	36	0.00%
	pred7	28	17	12	0	0	25	30.49%
	recall	13.53%	4.20%	73.47%	0.00%	0.00%	29.07%	127

Accuracy	0.2096
Kappa	0.04
WMRecall	0.2004
WMPrecision	0.2231
T1Score	0.211142

Скважина 2100

D vs L

		D						
		true 1	true 3	true 4	true 5	true 6	true7	Precision
L	pred1	197	35	5	0	0	1	82.77%
	pred3	1	226	20	0	0	0	91.50%
	pred4	0	34	44	0	0	0	56.41%
	pred5	1	0	0	6	0	0	85.71%
	pred6	0	0	0	0	0	0	0.00%
	pred7	2	1	0	1	0	61	93.85%
	recall	98.01%	76.35%	63.77%	85.71%	0.00%	98.39%	534

Accuracy	0.8409
Kappa	0.76
WMRecall	0.8445
WMPrecision	0.8205
T1Score	0.832327

D vs T

		D						
		true 1	true 3	true 4	true 5	true 6	true7	precision
T	pred1	126	111	0	0	0	1	52.94%
	pred3	19	192	34	0	0	2	77.73%
	pred4	0	29	40	0	0	9	51.28%
	pred5	7	0	0	0	0	0	0.00%
	pred6	0	0	0	0	0	0	0.00%
	pred7	5	2	0	0	0	58	89.23%
	recall	80.25%	57.49%	54.05%	0.00%	0.00%	82.86%	416

Accuracy	0.6551
Kappa	0.49
WMRecall	0.5493
WMPrecision	0.5424
T1Score	0.545828

L vs T	T	L						
			true 1	true 3	true 4	true 5	true 6	true7	Precision
		pred1	116	85	0	0	0	0	57.71%
		pred3	32	236	26	0	0	2	79.73%
		pred4	0	11	48	0	0	10	69.57%
		pred5	7	0	0	0	0	0	0.00%
		pred6	0	0	0	0	0	0	0.00%
		pred7	2	2	0	0	0	58	93.55%
		recall	73.89%	70.66%	64.86%	0.00%	0.00%	82.86%	458

Accuracy	0.7213
Kappa	0.57
WMRecall	0.5845
WMPrecision	0.6011
T1Score	0.592684

Примечание. Для скважины 2104 сравнение экспертов с усредненным с керном невозможно, так как данные керна получены только для глубин 459.3-462 метров, в то время как экспертные оценки даны для глубин 459-522 метров.

ПРИЛОЖЕНИЕ 3

Описание методики проведения экспериментов и применяемого программного обеспечения

Схема, приведенная ниже, описывает шаги экспериментального процесса (рисунок 3.1).

Знаком ⊘ обозначены шаги, которые не привели к существенному улучшению результатов.

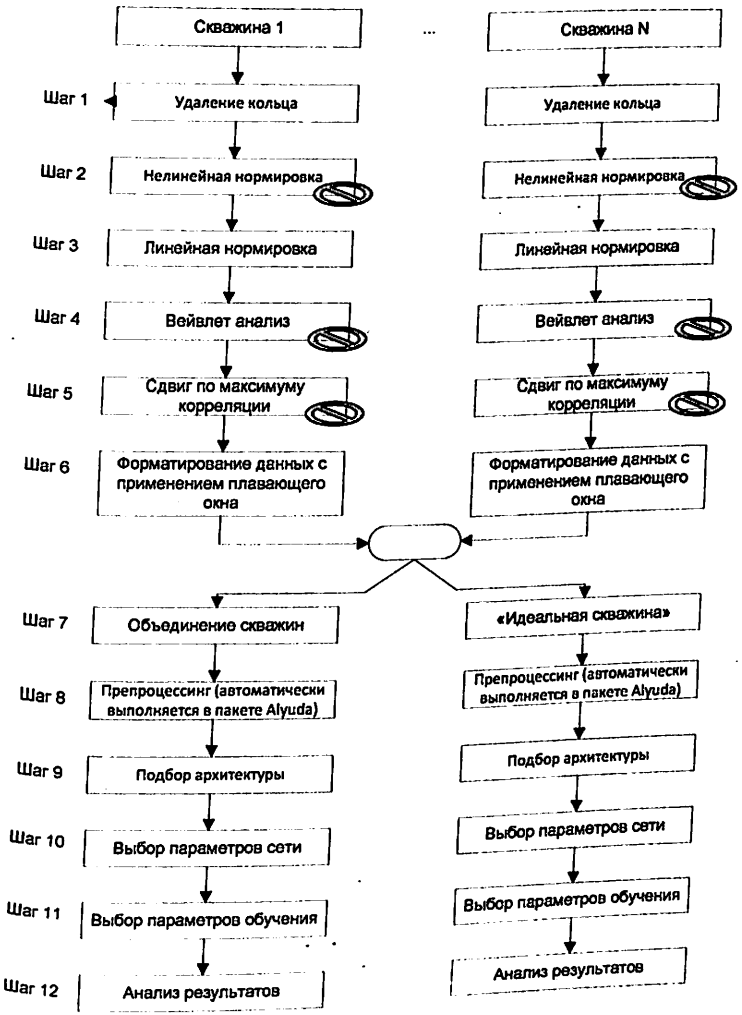


Рисунок 3.1. Пошаговая схема выполнения вычислительных экспериментов

Шаг 1. Удаление кольца фильтра: в процессе проведения экспериментов производилось вручную.

Программное обеспечение: отсутствует.

Статус: необходим.

Примечание: нет.

Шаг 2. Нелинейная нормировка

Использовалось 2 функции: гиперболический тангенс, сигмоида.

Программное обеспечение: шаблон Excel (Шаблон для дисперсии + сигмоида.xls).

Статус: согласно результатам экспериментов, нелинейная нормировка не принесла положительных результатов.

Примечание: должна производиться до формирования обучающей выборки.

Шаг 3. Линейная нормировка. Имеется в используемом пакете *Alyuda Software*. Но не может быть использована, поскольку нормировка должна проводиться до объединения скважин вследствие того, что записи имеют разный уровень сигнала.

Статус: является необходимым элементом обработки данных.

Примечание: должна производиться до формирования обучающей выборки.

Программное обеспечение:

- Шаблон Excel (Шаблон для Нормировки.xls).
- Консольное приложение (globalNorm.py).

Шаг 4. Вейвлет-сглаживание. Проверялось множество разных комбинаций, от самого грубого вейвлета Хаара до самых гладких вейвлетов высоких порядков.

Статус: согласно результатам экспериментов, не оказало положительного эффекта.

Примечание: должно производиться до формирования обучающей выборки.

Программное обеспечение: графическое приложение (WaveDemo.exe).

Шаг 5. Сдвиг по максимуму корреляции

Статус: согласно результатам экспериментов, не оказал положительного эффекта. Применялся как сдвиг по ИК, так и сдвиг по ПС. Были использованы разные алгоритмы корреляции.

Примечание: должно производиться до формирования обучающей выборки.

Программное обеспечение:

- Консольное приложение (corrGis.py).
- Консольное приложение (corrPy.py).

Шаг 6. «Идеальная» скважина – выборка из всех скважин, где количество примеров каждой породы одинаково, во избежание «перетягивания» весов нейронной сети в сторону наиболее часто встречающихся пород (123, 3, 74).

Статус: согласно результатам экспериментов, при использованных 4400 строках вместо 9000 был достигнут средний процент выше, чем при 9000, а именно 67% вместо 64%. Другими словами, данный подход использует меньший объем данных для обучения и дает лучший результат.

Примечание: должно производиться после форматирования данных методом плавающего окна. При этом только 7 типов пород прошли порог в 5% (количество примеров для трех пород (1, 9, 59) было очень маленьким (менее 0,5% от всего количества), по этой причине они не были включены в «идеальную» скважину).

Программное обеспечение: нет, формирование «идеальных» скважин происходило вручную (Идеальная.rar).

Шаг 7. Объединение скважин

Статус: необходимый элемент обработки данных.

Примечание: должно производиться после форматирования данных методом плавающего окна. Это позволяет избежать появления фантомных примеров, то есть примеров, возникающих на стыке данных двух скважин.

Программное обеспечение:

- Графическое приложение (ExcelProg.exe).
- Консольное приложение (connecter.py).

- Шаг 8. Препроцессинг – предобработка данных.
- Статус: необходимый элемент обработки данных.
- Примечания:

– В процессе экспериментов применялось статичное разделение данных на множества *Training*; *Validation* и *Testing*. Статичный *(Specific order)* – не случайный, который применяется в пакете *Alyuda* по умолчанию.

– Деление данных производилось построчно, а именно 17 скважин на обучение, по 450 строк в каждой скважине (11*450 строк – *Training*, 3*450 строк – *Validation*, 3*450 строк – *Testing*).

Рисунок 3.2 иллюстрирует этот процесс.

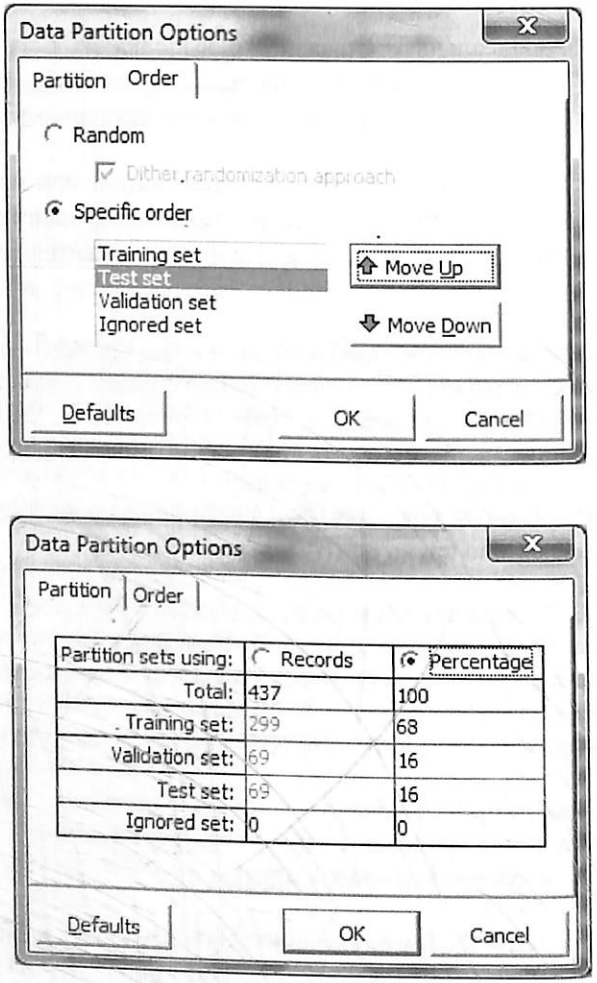


Рисунок 3.2. Деление скважин

– Правильные ответы помечались как *Categorical* (рисунок 3.3).

– Программное обеспечение: *Alyuda Software*.

– Шаг 9. Поиск архитектуры нейронной сети

– Статус: необходимый.

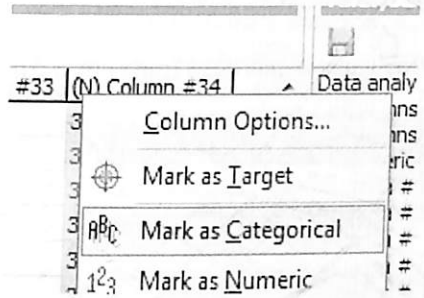


Рисунок 3.3. Отметка правильных ответов

Примечания:

– Требует значительного времени (около 26 часов). Параметры, заданные для обучения *(Train)*, и параметры нейронов использованы в поиске архитектуры сети (описаны в шагах 10 и 11).

– Нет теоретических способов подборки архитектуры, поэтому был использован метод подбора по конечному результату, реализованный в *Alyuda Software* (рисунок 3.4).

– Параметры поиска указаны на рисунке 3.4.

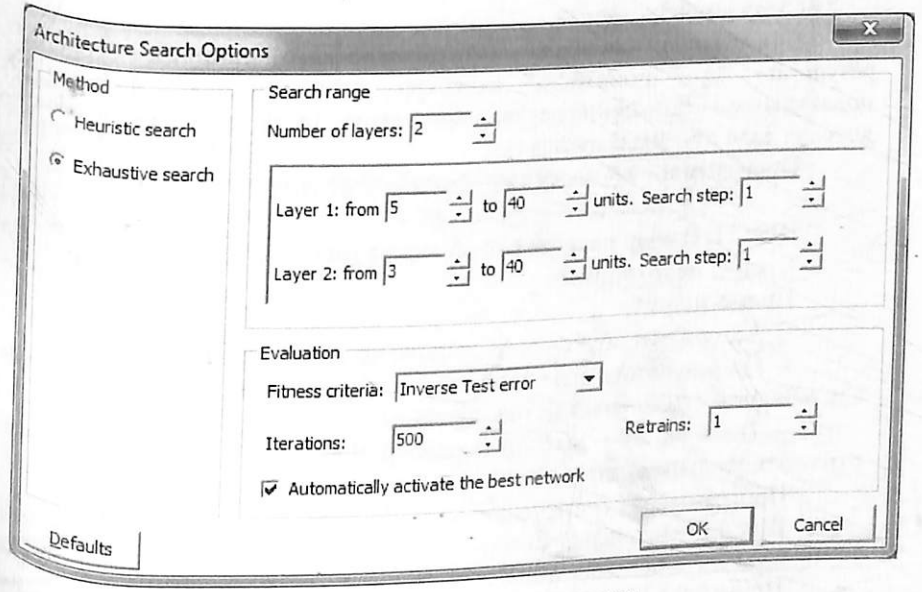


Рисунок 3.4. Выбор параметров поиска

Программное обеспечение: *Alyuda Software*.

Шаг 10. Выбор параметров сети (рисунок 3.5):

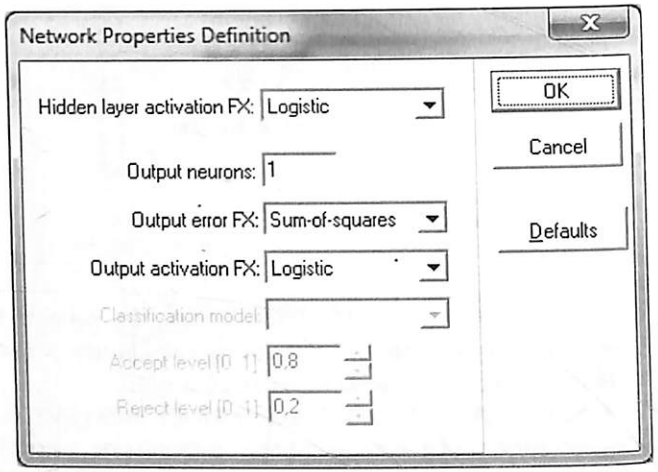


Рисунок 3.5. Выбор параметров сети

Статус: необходимый.

Примечание: функция гиперболического тангенса огрубляла результат, что понижало качество распознавания, поэтому в подавляющем большинстве экспериментов была выбрана сигмоида, которая дала лучшие результаты.

Программное обеспечение: *Alyuda Software*.

Шаг 11. Выбор параметров обучения (рисунок 3.6)

Статус: необходимый.

Примечания:

– Алгоритмы обучения расположены в порядке популярности.

– По результатам экспериментов градиентный спуск показал себя как наиболее стабильный метод обучения.

– *Track on set* – выбран параметр *Validation,* т.к. в большей мере отражает реальный результат.

Программное обеспечение: *Alyuda Software*.

Шаг 12. Анализ результатов

Статус: необходимый.

Примечание: необходим анализ среднего процента по нескольким «сырым» скважинам.

Программное обеспечение:

– Для сопоставления правильных ответов и ответов обученной сети использовалась функция *Excel* «ЕСЛИ».

– Для построения матрицы совпадений *(Confusion matrix)* использован шаблон (Confusion Matrix.xls).

Матрица совпадений *(Confusion matrix)* – матрица результатов распознавания, которая показывает, на каких породах сеть ошибается больше всего.

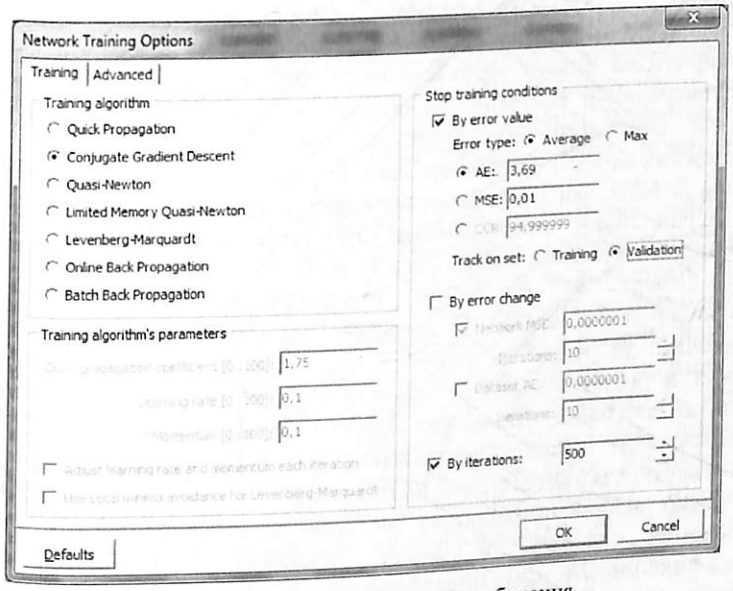


Рисунок 3.6. Выбор параметров обучения

ПРИЛОЖЕНИЕ 4

Описание программного комплекса Preprocessing module

Область применения

Системы для классификации различных объектов активно используются во многих сферах человеческой деятельности, в том числе в геологии. Геологические исследования заключаются в получении информации об определенных свойствах пород на конкретных глубинах (например, электромагнитных свойствах) с последующей классификацией (распознаванием) пород на каждой глубине. Классификация может производиться как экспертом-геологом, так и интеллектуальной системой на основе методов машинного обучения, к которым относятся искусственные нейронные сети (ИНН), группа метрических методов (наиболее яркий представитель – метод ближайших соседей *(k-NN))*, метод опорных векторов, статистические методы и др.

Существуют программные продукты, реализующие алгоритмы машинного обучения, такие как *Alyuda NeuroIntelligence, RapidMiner, Weka, Orange, MatLab* и др. Поскольку методики и практические реализации алгоритмов и моделей машинного обучения изучены достаточно подробно, в ряде случаев можно достигать результатов, близких к оптимальным, на данном наборе данных. Однако хороший результат достижим только при корректных исходных данных. При этом универсального метода достижения подобной корректности пока не существует. Это означает, что на практике большое значение приобретает не только выбор подходящих методов классификации, но и методы приведения данных к виду, дающему наилучший результат классификации. Методы предобработки, к которым относят нормировку, очистку от аномальных и шумовых значений, иные линейные и нелинейные преобразования, зависят от решаемой задачи и предметной области. Таким образом, если классификационные алгоритмы являются в значительной мере универсальными, то сочетание методов предобработки данных зачастую уникально для каждой практической задачи.

Preprocessing Module решает проблему предобработки и предварительного анализа данных электрического каротажа скважин по добыче урана методом подземного выщелачивания.

Назначение

Preprocessing Module может использоваться в качестве модуля предобработки данных и предварительного анализа данных электромагнитного каротажа.

В том числе возможны следующие сценарии использования:

1. Использование *Preprocessing Module* в качестве модуля предобработки с целью последующего вывода обработанных данных в формате, читаемом программами, реализующими алгоритмы классификации данных с целью классификации литотипов для дальнейшего использования либо с целью проведения экспериментов для выявления оптимальных распознающих моделей и/или вариантов предобработки, а также других закономерностей.

2. Использование *Preprocessing Module* в качестве модуля классификации метрическим алгоритмом. Поскольку *Preprocessing Module* совмещает в себе функционал обработки и классификации, он может быть использован как полноценная классифицирующая модель. Целью такого использования может быть постановка экспериментов, а также вывод результатов интерпретации для дальнейшего применения и анализа.

3. Использование *Preprocessing Module* в качестве инструмента для анализа данных. Гибкость функционала позволяет проводить широкий спектр экспериментов, в том числе использовать классы *Preprocessing Module* как фреймворк для любого анализа данных. Возможность вывода данных в различных форматах позволяет применять для анализа такое программное обеспечение, как *MatLab* и другие. Таким образом, *Preprocessing Module* может быть использован в качестве инструмента предварительного анализа данных, для выявления статистических свойств данных, влияния отдельных параметров на результат распознавания, качество и точность данных, наличие и количество шумовых и аномальных значений и др.

Функциональные возможности

Функционал программы можно разделить на:

Обработку для любых наборов данных

Сюда относятся функции *norm, clean_anomalies, clean_noisy_points, window*. Данные методы могут использоваться для любых данных и не привязаны к особенностям данных геологического каротажа.

2. Функции обработки данных геологического каротажа, специфические для этого типа данных

К этой группе функционала можно отнести: *shift_corr, inverse_PS, align_level*. Эти функции позволяют производить обработку данных,

полезную для данных геологического каротажа. Практическая полезность этих функций при использовании на других данных сомнительна.

3. Методы предварительного анализа данных, утилиты

Другие функции, такие как *crude_define, for_matlab, count_anomalies, er-ror_percent_per_soil, divide_to_classes* и прочие, представляют возможности для исследования, анализа данных, а также предоставляют функционал для удобного вывода данных с целью использования в других программах.

Для доступа к функционалу *Preprocessing Module* необходимо использовать два основных класса – *Sample* (выборка) и *BoreHole* (отдельное множество данных, скважина). Ниже приводятся описания действия всех функций с указанием принимаемых аргументов и возвращаемых параметров.

Класс Sample

Sample(vector<const char*>) – конструктор класса *Sample.* Принимает вектор С-строк с адресами файлов с данными. Формат ввода данных задается в классе *Point,* перегруженном операторе std::istream& operator>> (std::istream& stream, Point& point). По умолчанию формат – глубина-гамма-каротаж-КС-ПС-литотип.

vector<int> give_soils(); – функция возвращает связанный список литотипов пород (целые числа) в порядке появления в выборке.

vector<int> give_num_points_per_soil(); – функция возвращает связанный список с количествами точек каждой породы в порядке появления (порядок такой же, как в give_soils();

double clean_anomalies(const int, double); – функция поиска аномальных значений в выборке. Возвращает долю аномальных точек (от 0 до 1), помечает свойство объекта Point isAnomaly = true; принимает количество ближайших соседей (для типичного сценария использования 50-500) и предел, после которого точку следует считать аномалией (обычно 0.1-0.5, подбирается под данные).

void clean_noisy_points(const int, const double); – функция поиска шумовых точек на основе метрических методов. Помечает шумовые объекты класса Point – свойство isNosiy = true; принимает количество ближайших соседей (для типичного сценария использования 50-500) и предел, после которого точку следует считать аномалией (обычно 0.1-0.5, подбирается под данные).

void norm(BoreHole&); – функция нормировки выборки и скважины для распознавания. Принимает скважину, вместе с которой будет нормироваться.

void align_level(); – исключает разность уровней записей различных скважин в выборке путем линейного сдвига. Центральное (среднее) значение оказывается равным нулю.

void shift_corr(); – исключение сдвига каротажных кривых относительно друг друга путем корреляционного анализа.

void inverse_PS(int, int, int); – функция исключения инвертации кривой ПС. Принимает минимальный интервал, который следует считать инвертированным, порядок и уровень вейвлета Добеши для сглаживания (сглаживается копия выборки, данные не сглаживаются, а только исключается инвертирование ПС).

void crude_clean_anomalies(BoreHole&, const int, double); – очистка контрольной скважины (первый аргумент) от аномальных точек. Принимает скважину, которую необходимо очистить (первый аргумент), количество ближайших соседей (для типичного сценария использования 50-500) и предел, после которого точку следует считать аномалией (обычно 0.1-0.5, подбирается под данные).

void window(string file, string number_of_crude, int radius); – принимает название файла, номер скважины и радиус плавающего окна. Записывает данные, отформатированные по принципу плавающего окна, в файл file.

double crude_define(int, BoreHole&); – принимает количество ближайших соседей и скважины для распознавания. Возвращает качество распознавания (если есть контрольные данные), записывает результаты интерпретации в отправленную скважину *BoreHole*. Для интерпретации используются метрические методы.

double crude_define_trend(int, BoreHole&); – принцип работы аналогичен double crude_define(int, BoreHole&);, однако в расчет берутся интервалы возрастания и убывания кривой КС – они интерпретируются как две отдельные выборки.

BoreHole getBoreHole(int id); – возвращает скважину под номером id (нумерация с нуля) в формате объекта класса *BoreHole*.

void for_matlab(); – записывает каждую породу в отдельный файл для последующего использования во внешних программах. Формат поддерживается пакетами *MatLab* и др.

void for_matlab_for_method_1(const char*); – записывает аномальные и не аномальные точки в отдельные файлы для последующего использования во внешних программах. Формат поддерживается пакетами *MatLab* и др.

void for_matlab_by_boreholes(string); – записывает каждую скважину в отдельный файл для последующего использования во внешних программах. Формат поддерживается пакетами *MatLab* и др.

vector<double> error_percent_per_soil(); – возвращает связанный список результатов интерпретации каждой отдельной породы. Порядок такой же, как в функции give_soils() (в порядке появления пород в выборке).

vector<double> error_percent_per_soil_trend(); – принцип работы такой же, как у функции vector<double> error_percent_per_soil();, но при интерпретации принимаются в расчет промежутки возрастания/убывания кривой КС.

void trend_marking(); – пометить промежутки возрастания/убывания кривой КС. Информация записывается в объекты *Point*.

vector<vector<Point>> divide_to_classes(); – возвращает данные, разделенные на породы для дальнейшего анализа. Порядок пород такой же, как в функции give_soils() (в порядке появления пород в выборке).

vector<Point> get_nearest_points(Point &point, int number, vector<Point>& src_KS, vector<Point>& src_PS); – возвращает связанный список из number точек, ближайших к point, где список точек – src_KS/src_PS(вектор КС и ПС). Используется оптимизированный аппроксимирующий алгоритм, абсолютная точность не гарантируется, но достаточна для метрических методов распознавания/очистки/анализа данных.

void b_b_interval_corelation(int b_1_index, int b_2_index, int b_1_begin, int b_2_begin, int number_of_points); – записывает коэффициент корреляции отрезков b_1 и b_2 в файл.

vector<vector< double >> define_on_probabilities(BoreHole& borehole, double& result); – распознавание статистическим методом. Возвращает связанный список вероятности принадлежности каждой точки к каждому классу. Порядок классов такой же, как в функции give_soils() (в порядке появления в выборке).

Класс BoreHole

BoreHole(const string file_name); – конструктор класса *BoreHole*. Принимает адрес файла, считывает из него данные. Формат ввода данных задается в классе *Point*, перегруженном операторе std::istream& operator>> (std::istream& stream, Point& point). По умолчанию формат – глубина-гамма-каротаж-КС-ПС-литотип.

vector<int> give_soils(); – функция возвращает связанный список литотипов пород (целые числа) в порядке появления в выборке.

vector<int> give_num_points_per_soil(); – функция возвращает связанный список с количествами точек каждой породы в порядке появления (порядок такой же, как в give_soils();).

void align_level(); – исключает разность уровней записей различных скважин в выборке путем линейного сдвига. Центральное (среднее) значение оказывается равным нулю.

void shift_corr(); – исключение сдвига каротажных кривых относительно друг друга путем корреляционного анализа.

void inverse_PS(int, int, int); – функция исключения инвертации кривой ПС. Принимает минимальный интервал, который следует считать инвертированным, порядок и уровень вейвлета Добеши для сглаживания (сглаживается копия выборки, данные не сглаживаются, а только исключается инвертирование ПС).

void window(string file, string number_of_crude, int radius); – принимает название файла, номер скважины и радиус плавающего окна. Записывает данные, отформатированные по принципу плавающего окна, в файл file.

void trend_marking(); – пометить промежутки возрастания/убывания кривой КС. Информация записывается в объекты *Point*.

Основные технические характеристики

Preprocessing Module – это система из C++ классов: *Point* (одна запись данных, точка в пространстве признаков), *BoreHole* (набор точек, множество данных, скважина) и *Sample* (выборка, состоящая из нескольких скважин).

При разработке были использованы только стандартные библиотеки C/C++, включая стандартную библиотеку шаблонов *STL*. Общий объем кода составляет около 3000 строк.

Логика обработки/классификации содержится в более чем 40 функциях, реализованных в классах *Sample* и *BoreHole*.

Структура хранения данных такова, что, с одной стороны, поддерживает высокую производительность алгоритмов, не требующих случайного доступа (подавляющее большинство реализованных алгоритмов производит последовательный доступ к данным, а не случайный доступ), с другой – оптимальное использование памяти при хранении больших объемов данных и масштабируемость на любые объемы данных (с учетом физических и аппаратных ограничений).

Все реализованные алгоритмы обработки настраиваемы, интегрированы и совместимы, что позволяет гибко конфигурировать процесс предобработки, менять последовательность алгоритмов предобработки, совершать вывод и анализ данных на любом этапе обработки, а также использовать *Preprocessing Module* как фреймворк для различных экспериментов и анализа данных.

Имеется возможность вывода данных в различных форматах для анализа во внешних инструментах, таких как *MatLab*. Наличие реализованных перегруженных операторов вывода позволяет быстро и удобно изменить формат вывода как отдельных записей, так и всей выборки без дополнительного рефакторинга кода.

Использование библиотеки *STL* позволило увеличить производительность, обеспечить масштабируемость модуля и кросс-платформенность *Preprocessing Module*. В частности, были использованы структуры данных *vector* для хранения данных, *map* и *set* для оптимальной реализации ряда алгоритмов обработки данных, а также оптимальные алгоритмы (сортировки, поиска и др.) из стандартной библиотеки *STL*.

ПРИЛОЖЕНИЕ 5
Программа ANNClassificator

Описание программы

Программа интерпретации методами машинного обучения: DLDA, LDAC, KNN, Linear SVM, NON-Linear SVM, Artificial Neural Networks.

Демонстрационное приложение:

Example.exe – приложение классифицирует данные геологических исследований скважин методами knn, LDAC, DLDA и neural network и выводит график результатов.

Вход:

На вход программы необходимо прописать путь до файла с конфигурациями.

Вывод:

Результаты по различным методам обучения и их график.

Формат файла конфигураций:

Должен быть записать в JSON формате, и иметь следующие переменные:

Columns_names – названия столбцов данных в том порядке, в каком они лежат в файлах с обучающей выборкой.

Result_column - название столбца результата

Input_used – словарь который хранит в ключах используемые параметры, а в значениях радиус плавающего окна.

Filters – словарь который хранит в ключах название столбцов фильтров (аномалии, шумы), а в значениях тип фильтрации. Столбец фильтр это одномерный массив из единичек и нулей, при единичном значении указывает на то, что эти данные не должны быть использованы. Есть два типа фильтрации: “WINDOW”, “POINT”. WINDOW исключает пример если хоть один из элементов в окне по фильтру является единицой, POINT исключает только если целевой элемент по фильтру является единицей.

Train_data_path – путь до файла с обучающей выборкой

Check_data_path – путь до файла с данными для распознавания

Max_iterations – максимальное количество итераций обучения нейронной сети

Architecture - архитектура нейронной сети (только скрытые слои)

Train_proportion – соотношение данных для обучения (остальная часть будет использована для выборки теста)

Result_file – файл для сохранения данных распознавания нейронной сетью.

Пример файла конфигурации

```
{
"columns_names" : ["KS","PS","IK","SOIL
CODE","DISCARD","NOISE"],
"result_column": "SOIL CODE",
"input_used" : {"KS":5, "PS":5, "IK":5},
"filters" : {"DISCARD":"WINDOW"},
"train_data_path" : "data/single/1.txt",
"check_data_path" : "data/single/2.txt",
"max_iterations" : 300,
"architecture" : [3, 10],
"train_percentage" : 0.75,
"result_file" : "res.xls"
}
```

Требования для запуска скомпилированного приложения

– Windows XP, 7, 8

Требования для запуска исходного кода

– интерпретатор Python 2.6
– установленные библиотеки mlpy, Matplotlib, numpy, xlwt.
– Windows, Linux, Mac OS

ПРИЛОЖЕНИЕ 6

Исходные данные для построения кривых обучаемости (Learning cruves)

6.1. Матрицы ошибок (Error matrix/confusion matrix) использованные для расчета показателей accuracy, recall, kappa.

Training set
Training and testing on the same 2 boreholes

	true 3.0	true 1.0	true 4.0	true 7.0	true 6.0	true 5.0	class precision
pred. 3.0	489	83	148	51	2	0	63.26 %
pred. 1.0	138	828	15	6	0	2	83.72 %
pred. 4.0	0	0	0	0	0	0	0.00 %
pred. 7.0	0	0	0	0	0	0	0.00 %
pred. 6.0	0	0	0	0	0	0	0.00 %
pred. 5.0	0	0	0	0	0	0	0.00 %
class recall	77.99%	90.89%	0.00 %	0.00 %	0.00 %	0.00 %	

Test set
Test on the 2 boreholes

	true 7.0	true 4.0	true 3.0	true 1.0	true 6.0	true 5.0	true 2.0	class precision
pred. 7.0	0	0	0	0	0	0	0	0.00%
pred. 4.0	0	0	0	0	0	0	0	0.00%
pred. 3.0	116	257	427	61	16	0	0	48.69%
pred. 1.0	2	51	328	700	0	10	18	63.12%
pred. 6.0	0	0	0	0	0	0	0	0.00%
pred. 5.0	0	0	0	0	0	0	0	0.00%
pred. 2.0	0	0	0	0	0	0	0	0.00%
class recall	0.00 %	0.00 %	56.56%	91.98 %	0.00 %	0.00 %	0.00%	

Training and testing on the same 3 boreholes

	true 3.0	true 1.0	true 4.0	true 7.0	true 6.0	true 5.0	class precision
pred. 3.0	591	138	185	22	7	0	62.67 %
pred. 1.0	188	1181	28	7	1	20	82.88 %
pred. 4.0	89	5	150	38	33	0	47.62 %
pred 7.0	0	0	0	0	0	0	0.00%
pred 6.0	0	0	0	0	0	0	0.00%
pred 5.0	0	0	0	0	0	0	0.00%
class recall	68.09 %	89.20 %	41.32 %	0.00%	0.00%	0.00%	

Test on the 3 boreholes

	true 7.0	true 4.0	true 3.0	true 1.0	true 6.0	true 5.0	true 2.0	class precision
pred. 7.0	4	0	0	0	0	1	0	80.00 %
pred. 4.0	114	161	52	2	42	0	0	43.40 %
pred. 3.0	17	377	482	91	3	0	0	49.69 %
pred. 1.0	2	64	430	1077	0	10	18	67.27 %
pred. 6.0	0	0	0	0	0	0	0	0.00%
pred. 5.0	0	0	0	0	0	0	0	0.00%
pred. 2.0	0	0	0	0	0	0	0	0.00%
class recall	2.92%	26.74 %	50.00 %	92.05 %	0.00%	0.00%	0.00%	

Training and testing on the same 4 boreholes

	true 3.0	true 1.0	true 4.0	true 7.0	true 6.0	true 5.0	class precision
pred. 3.0	903	201	323	38	23	0	60.69%
pred. 1.0	231	1704	37	7	1	20	85.20%
pred. 4.0	76	0	96	18	25	0	44.65%
pred. 7.0	0	0	0	0	0	0	0.00%
pred. 6.0	0	2	5	6	8	0	38.10%
pred. 5.0	0	0	0	0	0	0	0.00%
class recall	74.63 %	89.36 %	20.82 %	0.00%	14.04 %	0.00%	

Test on the 4 boreholes

	true 7.0	true 4.0	true 3.0	true 1.0	true 6.0	true 5.0	true 2.0	class precision
pred. 7.0	0	0	0	0	0	0	0	0.00%
pred. 4.0	32	37	11	0	0	0	0	46.25%
pred. 3.0	107	703	770	112	32	0	0	44.66%
pred. 1.0	2	62	566	1406	0	10	18	68.12%
pred. 6.0	27	1	0	0	13	1	0	30.95%
pred. 5.0	0	0	0	0	0	0	0	0.00%
pred. 2.0	0	0	0	0	0	0	0	0.00%
class recall	0.00%	4.61%	57.16 %	92.62 %	28.89 %	0.00%	0.00%	

Training and testing on the same 5 boreholes

	true 3.0	true 1.0	true 4.0	true 7.0	true 6.0	true 5.0	class precision
pred. 3.0	1259	299	367	49	28	0	62.89 %
pred. 1.0	264	1919	27	7	1	20	85.75 %
pred. 4.0	105	0	121	28	26	0	43.21 %
pred. 7.0	0	2	5	24	10	0	58.54 %
pred. 6.0	0	0	0	0	14	0	100.00 %
pred. 5.0	0	0	0	0	0	0	0.00%
class recall	77.33 %	86.44 %	23.27 %	22.22 %	17.72 %	0.00%	

Test on the 5 boreholes

	true 7.0	true 4.0	true 3.0	true 1.0	true 6.0	true 5.0	true 2.0	class precision
pred. 7.0	41	21	17	3	18	1	0	40.59 %
pred. 4.0	28	112	50	4	0	0	0	57.73 %
pred. 3.0	110	802	1079	340	27	0	18	45.76 %
pred. 1.0	11	53	447	1747	0	32	0	75.69 %
pred. 6.0	7	9	6	0	0	0	0	0.00%
pred. 5.0	0	0	0	0	0	0	0	0.00%
pred. 2.0	0	0	0	0	0	0	0	0.00%
class recall	20.81 %	11.23 %	67.48 %	83.43 %	0.00%	0.00%	0.00%	

Training and testing on the same 6 boreholes

	true 3.0	true 1.0	true 4.0	true 7.0	true 6.0	true 5.0	class precision
pred. 3.0	1830	395	449	89	44	1	65.17 %
pred. 1.0	448	2229	52	11	9	48	79.69 %
pred. 4.0	103	0	147	50	24	0	45.37 %
pred. 7.0	15	4	19	64	45	0	43.54 %
pred. 6.0	0	0	0	0	0	0	0.00%
pred. 5.0	0	0	0	0	0	0	0.00%
class recall	76.38 %	84.82 %	22.04 %	29.91 %	0.00%	0.00%	

Test on the 6 boreholes

	true 7.0	true 4.0	true 3.0	true 1.0	true 6.0	true 5.0	true 2.0	class precision
pred. 7.0	105	66	44	4	22	1	0	43.39 %
pred. 4.0	22	112	118	9	11	0	0	41.18 %
pred. 3.0	112	782	1319	417	12	0	0	49.02 %
pred. 1.0	14	68	521	2136	0	32	18	76.59 %
pred. 6.0	0	0	0	1	0	0	0	0.00%
pred. 5.0	0	0	0	0	0	0	0	0.00%
pred. 2.0	0	0	0	0	0	0	0	0.00%
class recall	41.50 %	10.89 %	65.88 %	83.21 %	0.00%	0.00%	0.00%	

Training and testing on the same 7 boreholes

	true 3.0	true 1.0	true 4.0	true 7.0	true 6.0	true 5.0	class precision
pred. 3.0	2021	388	478	130	46	1	65.96 %
pred. 1.0	598	2598	73	21	9	48	77.62 %
pred. 4.0	80	0	129	32	22	0	49.05 %
pred. 7.0	25	7	21	101	49	0	49.75 %
pred. 6.0	0	0	0	0	0	0	0.00%
pred. 5.0	0	0	0	0	0	0	0.00%
class recall	74.19 %	86.80 %	18.40 %	35.56 %	0.00%	0.00%	

Test on the 7 boreholes

	true 7.0	true 4.0	true 3.0	true 1.0	true 6.0	true 5.0	true 2.0	class precision
pred. 7.0	115	99	53	9	34	2	0	36.86 %
pred. 4.0	11	99	136	4	1	0	0	39.44 %
pred. 3.0	116	820	1528	480	26	2	0	51.41 %
pred. 1.0	16	81	585	2552	0	42	18	77.47 %
pred. 6.0	0	0	0	0	0	0	0	0.00%
pred. 5.0	0	0	0	0	0	0	0	0.00%
pred. 2.0	0	0	0	0	0	0	0	0.00%
class recall	44.57 %	9.01%	66.38 %	83.81 %	0.00%	0.00%	0.00%	

Training and testing on the same 8 boreholes

	true 3.0	true 1.0	true 4.0	true 7.0	true 6.0	true 5.0	class precision
pred. 3.0	2338	580	671	118	45	2	62.28 %
pred. 1.0	552	2802	36	21	9	52	80.70 %
pred. 4.0	107	3	150	86	27	0	40.21 %
pred. 7.0	18	7	27	102	45	4	50.25 %
pred. 6.0	0	0	0	0	0	0	0.00%
pred. 5.0	0	0	0	0	0	0	0.00%
class recall	77.55 %	82.61 %	16.97 %	31.19 %	0.00%	0.00%	

Test on the 8 boreholes

	true 7.0	true 4.0	true 3.0	true 1.0	true 6.0	true 5.0	true 2.0	class precision
pred. 7.0	148	112	70	12	50	2	0	37.56 %
pred. 4.0	48	169	221	19	22	0	0	35.28 %
pred. 3.0	84	942	1727	595	31	5	0	51.03 %
pred. 1.0	14	58	530	2805	5	42	18	80.79 %
pred. 6.0	0	0	0	0	0	0	0	0.00%
pred. 5.0	0	0	0	0	0	0	0	0.00%
pred. 2.0	0	0	0	0	0	0	0	0.00%
class recall	50.34 %	13.19 %	67.78 %	81.75 %	0.00%	0.00%	0.00%	

Training and testing on the same 9 boreholes

	true 3.0	true 1.0	true 4.0	true 7.0	true 6.0	true 5.0	class precision
Pred. 3.0	2561	874	762	168	52	3	57.94 %
Pred. 1.0	565	3203	41	22	10	52	82.28 %
Pred. 4.0	72	0	149	57	25	0	49.17 %
Pred. 7.0	24	11	32	87	52	3	41.63 %
Pred. 6.0	0	0	0	0	0	0	0.00%
Pred. 5.0	0	0	0	0	0	0	0.00%
class recall	79.48 %	78.35 %	15.14 %	26.05 %	0.00%	0.00%	

Test on the 9 boreholes

	true 7.0	true 4.0	true 3.0	true 1.0	true 6.0	true 5.0	true 2.0	class precision
pred. 7.0	151	75	59	11	44	2	0	44.15 %
pred. 4.0	17	125	137	6	11	0	0	42.23 %
pred. 3.0	125	1198	2009	662	63	5	0	49.46 %
pred. 1.0	14	63	568	3189	10	45	18	81.62 %
pred. 6.0	0	0	0	0	0	0	0	0.00%
pred. 5.0	0	0	0	0	0	0	0	0.00%
pred. 2.0	0	0	0	0	0	0	0	0.00%
class recall	49.19 %	8.56%	72.45 %	82.45 %	0.00%	0.00%	0.00 %	

Training and testing on the same 10 boreholes

	true 3.0	true 1.0	true 4.0	true 7.0	true 6.0	true 5.0	class precision
pred. 3.0	2531	826	665	102	32	3	60.86 %
pred. 1.0	680	3723	67	25	10	58	81.59 %
pred. 4.0	232	6	335	137	55	0	43.79 %
pred. 7.0	41	16	40	129	42	3	47.60 %
pred. 6.0	0	0	0	0	0	0	0.00%
pred. 5.0	0	0	0	0	0	0	0.00%
class recall	72.65 %	81.45 %	30.26 %	32.82 %	0.00%	0.00%	

Test on the 10 boreholes

	true 7.0	true 4.0	true 3.0	true 1.0	true 6.0	true 5.0	true 2.0	class precision
pred. 7.0	171	88	73	14	59	1	0	42.12 %
pred. 4.0	63	281	303	29	22	0	0	40.26 %
pred. 3.0	86	1067	1858	756	33	8	0	48.79 %
pred. 1.0	19	94	688	3600	14	55	18	80.21 %
pred. 6.0	0	0	0	0	0	0	0	0.00%
pred. 5.0	0	0	0	0	0	0	0	0.00%
pred. 2.0	0	0	0	0	0	0	0	0.00%
class recall	50.44 %	18.37 %	63.59 %	81.84 %	0.00%	0.00%	0.00%	

Training and testing on the same 11 boreholes

	true 3.0	true 1.0	true 4.0	true 7.0	true 6.0	true 5.0	class precision
pred. 3.0	2861	950	804	130	56	3	59.55 %
pred. 1.0	722	4175	67	30	11	58	82.46 %
pred. 4.0	193	3	331	141	77	0	44.43 %
pred. 7.0	24	15	35	113	39	3	49.34 %
pred. 6.0	0	0	0	0	0	0	0.00%
pred. 5.0	0	0	0	0	0	0	0.00%
class recall	75.29 %	81.18 %	26.76 %	27.29 %	0.00%	0.00%	

Test on the 11 boreholes

	true 7.0	true 4.0	true 3.0	true 1.0	true 6.0	true 5.0	true 2.0	class precision
pred. 7.0	177	65	43	10	57	1	0	50.14 %
pred. 4.0	91	246	286	32	26	0	0	36.12 %
pred. 3.0	147	1229	2129	818	94	7	0	48.12 %
pred. 1.0	21	99	790	3852	16	56	18	79.39 %
pred. 6.0	0	0	0	0	0	0	0	0.00%
pred. 5.0	0	0	0	0	0	0	0	0.00%
pred. 2.0	0	0	0	0	0	0	0	0.00%
class recall	40.60 %	15.01 %	65.55 %	81.75 %	0.00%	0.00%	0.00%	

Training and testing on the same 12 boreholes

	true 3.0	true 1.0	true 4.0	true 7.0	true 6.0	true 5.0	class precision
pred. 3.0	2889	854	794	117	28	3	61.66 %
pred. 1.0	960	4678	123	31	11	32	80.17 %
pred. 4.0	261	6	389	136	71	0	45.08 %
pred. 7.0	61	16	68	242	73	3	52.27 %
pred. 6.0	0	0	0	0	0	0	0.00%
pred. 5.0	0	2	0	0	0	36	94.74 %
class recall	69.26 %	84.20 %	28.31 %	46.01 %	0.00%	48.65 %	

Test on the 12 boreholes

	true 7.0	true 4.0	true 3.0	true 1.0	true 6.0	true 5.0	true 2.0	class precision
pred. 7.0	262	163	115	17	88	1	0	40.56 %
pred. 4.0	61	331	305	45	38	0	0	42.44 %
pred. 3.0	100	1165	2123	729	65	8	18	50.67 %
pred. 1.0	23	127	960	4362	17	30	0	78.78 %
pred. 6.0	0	0	0	0	0	0	0	0.00%
pred. 5.0	0	0	0	1	0	30	0	96.77 %
pred. 2.0	0	0	0	0	0	0	0	0.00%
class recall	58.74 %	18.53 %	60.61 %	84.63 %	0.00%	43.48 %	0.00%	

Training and testing on the same 13 boreholes

	true 3.0	true 1.0	true 4.0	true 7.0	true 6.0	true 5.0	class precision
pred. 3.0	3291	1132	1006	121	21	3	59.04 %
pred. 1.0	790	4747	81	28	11	39	83.34 %
pred. 4.0	365	14	521	192	86	0	44.23 %
pred. 7.0	42	10	56	243	65	3	58.00 %
pred. 6.0	0	0	0	0	0	0	0.00%
pred. 5.0	0	6	0	0	0	34	85.00 %
class recall	73.33 %	80.34 %	31.31 %	41.61 %	0.00%	43.04 %	

Test on the 13 boreholes

	true 7.0	true 4.0	true 3.0	true 1.0	true 6.0	true 5.0	true 2.0	class precision
pred. 7.0	239	106	77	11	75	2	0	46.86 %
pred. 4.0	110	532	411	67	81	0	0	44.30 %
pred. 3.0	102	1282	2453	892	76	10	0	50.94 %
pred. 1.0	18	89	869	4474	13	31	18	81.17 %
pred. 6.0	0	0	0	0	0	0	0	0.00%
pred. 5.0	0	0	0	2	0	28	0	93.33 %
pred. 2.0	0	0	0	0	0	0	0	0.00%
class recall	50.96 %	26.48 %	64.38 %	82.15 %	0.00%	39.44 %	0.00%	

Training and testing on the same 14 boreholes

	true 3.0	true 1.0	true 4.0	true 7.0	true 6.0	true 5.0	class precision
pred. 3.0	3393	1026	1081	116	20	3	60.17 %
pred. 1.0	1017	5326	125	38	11	47	81.14 %
pred. 4.0	348	12	496	165	80	0	45.05 %
pred. 7.0	54	15	77	311	72	3	58.46 %
pred. 6.0	0	0	0	0	0	0	0.00%
pred. 5.0	1	6	0	0	0	37	84.09 %
class recall	70.50 %	83.41 %	27.88 %	49.37 %	0.00%	41.11 %	

Test on the 14 boreholes

	true 7.0	true 4.0	true 3.0	true 1.0	true 6.0	true 5.0	true 2.0	class precision
pred. 7.0	273	148	111	14	103	2	0	41.94 %
pred. 4.0	105	560	371	58	77	2	0	47.74 %
pred. 3.0	87	1459	2412	789	104	8	0	49.64 %
pred. 1.0	21	143	1092	4911	20	30	18	78.77 %
pred. 6.0	0	0	0	0	0	0	0	0.00%
pred. 5.0	0	0	0	2	0	38	0	95.00 %
pred. 2.0	0	0	0	0	0	0	0	0.00%
class recall	56.17 %	24.24 %	60.51 %	85.05 %	0.00%	47.50 %	0.00%	

Training and testing on the same 15 boreholes

	true 3.0	true 1.0	true 4.0	true 7.0	true 6.0	true 5.0	class precision
pred. 3.0	3549	1050	1148	122	21	2	60.23 %
pred. 1.0	1115	5773	161	39	12	50	80.74 %
pred. 4.0	378	16	580	193	77	0	46.62 %
pred. 7.0	50	16	63	313	73	3	60.42 %
pred. 6.0	0	0	0	0	0	0	0.00%
pred. 5.0	2	8	0	0	0	49	83.05 %
class recall	69.67 %	84.12 %	29.71 %	46.93 %	0.00%	47.12 %	

Test on the 15 boreholes

	true 7.0	true 4.0	true 3.0	true 1.0	true 6.0	true 5.0	true 2.0	class precision
pred. 7.0	295	125	96	11	96	2	0	47.20 %
pred. 4.0	114	708	406	77	87	2	0	50.79 %
pred. 3.0	95	1632	2449	752	123	12	0	48.37 %
pred. 1.0	21	162	1227	5219	20	32	18	77.91 %
pred. 6.0	0	0	0	0	0	0	0	0.00%
pred. 5.0	0	0	0	4	0	55	0	93.22 %
pred. 2.0	0	0	0	0	0	0	0	0.00%
class recall	56.19 %	26.95 %	58.62 %	86.08 %	0.00%	53.40 %	0.00%	

6.2. Показатели accuracy, recall, kappa алгоритма k-NN

Training and testing on the same boreholes						Testing by using test set					
Identi-ficator of borehole	The number of boreholes	accuracy 2	wm recall2	wm precision2	kappa2	Identificator of borehole	The number of boreholes	Accu-racy1	wm recall1	wm preci-sion1	kappa1
10027	1	70.27	39.19	34.63	0.461	10202	1	53.61	33.21	26.41	0.225
10028	2	73.5	35.54	33.2	0.546	10203	2	53.58	21.73	19.85	0.278
10029	3	71.23	35.4	32.98	0.535	10208	3	54.6	22.57	21.01	0.304
10030	4	71.86	33.09	32.38	0.525	10209	4	57.01	23.5	22.68	0.337
10074	5	71.19	32.51	32.07	0.519	10210	5	58.3	23.44	23.53	0.351
10075	6	68.12	30.56	41.99	0.474	10213-1	6	60.8	23.88	35.37	0.377
10076	7	68.87	30.49	42.16	0.481	10217	7	61.71	23.8	34.96	0.386
10114	8	67.52	30.26	41.09	0.464	10229	8	60.98	24.72	32.15	0.382
10132	9	66.7	29.25	37	0.445	10259-1	9	61.32	23.62	38.72	0.38
10133	10	67.88	30.53	43.93	0.465	10271	10	60.97	23.73	33.24	0.37
10134	11	67.87	30.95	52.34	0.466	10174	11	59.55	23.49	31.83	0.353
10135	12	68.34	33.89	43.02	0.481	10175	12	61.52	26.74	31.26	0.389
10138	13	68.03	34.72	43.27	0.486	10176	13	61.22	27.61	30.91	0.392
10162	14	68.52	35.31	44.01	0.493	10177	14	60.73	27.73	30.77	0.389
10166	15	68.72	35.82	44.11	0.498	10198	15	60.43	28.51	31.28	0.391
	average	69.714	32.68	37.143	0.4915		average	58.288	24.42	28.79	0.34

6.3. Показатели accuracy, recall, kappa алгоритма ИНС для сети с одним скрытым слоем размером 20 нейронов, входной слой - 22 нейрона (22-20-7)

Neural Net	
Hidden layer	1
Neurons in the hidden layer -20	20
train cycle	500
learn rate	0.01

Testing by using test set						Training and testing on the same boreholes					
Identificator of borehole	The number of boreholes	accuracy 1	wm recall1	wm preci-sion1	kappa1	Identificator of borehole	The number of boreholes	Accu-racy2	wm recall2	wm preci-sion2	kappa2
10202	1	60.73	37.74	30.13	34.50	10027	1	75.32	42.18	37.23	55.3
10203	2	56.75	21.22	15.97	30.00	10028	2	74.74	28.15	24.5	54.4
10208	3	58.5	24.53	34.34	36.20	10029	3	71.64	33.1	32.19	53.4
10209	4	56.93	26.18	27.14	32.60	10030	4	72.8	33.14	38.11	53.7
10210	5	59.78	26.14	31.40	37.60	10074	5	72.94	37.83	58.4	54.9
10213-1	6	61.76	28.78	30.15	40.10	10075	6	70.28	35.52	38.96	51.4
10217	7	62.88	29.11	29.31	41.10	10076	7	70.51	35.83	40.4	51.4
10229	8	62.74	30.44	29.24	42.10	10114	8	69.11	34.72	38.91	49.7
10259-1	9	63.60	30.38	31.07	42.80	10132	9	67.99	33.17	38.5	47.4
10271	10	62.87	30.60	30.20	41.70	10133	10	68.85	36.2	38.97	49.7
10174	11	62.11	28.99	30.54	40.60	10134	11	69	35.09	39.3	49.5
10175	12	63.56	38	44.17	43.10	10135	12	69.29	46.07	55.65	50.6
10176	13	64.02	37.63	45.23	44.60	10138	13	68.46	44.94	54.93	50.3
10177	14	63.24	39.07	44.73	43.60	10162	14	68.9	45.38	54.82	50.6
10198	15	63.05	40.18	45.36	43.70	10166	15	69.06	46.26	55.18	50.8
	average	60.654	28.512	28.90	39.62		average	71.418	34.984	38.617	51.54

Hidden layer
Neurons in the hidden layer -20
train cycle
learn rate

Testing by using test set						Training and testing on the same boreholes					
Identificator of borehole	The number of bore-holes	accuracy 1	wm recall1	wm precision1	kappa1	Identificator of borehole	The number of bore-holes	accuracy 2	wm recall2	wm precision2	kappa2
10202	1	60.63	41.98	49.69	0.347	10027	1	77.91	52.15	63.28	0.601
10203	2	57.65	24.22	25.95	0.322	10028	2	75.94	32.74	40.52	0.569
10208	3	58.7	27.20	30.65	0.369	10029	3	72.68	39.9	50.21	0.552
10209	4	58.87	29.55	29.21	0.363	10030	4	73.36	42.03	41.73	0.555
10210	5	59.74	25.96	30.55	0.379	10074	5	73.49	39.74	48.24	0.56
10213-1	6	61.79	28.46	31.03	0.403	10075	6	70.06	36.14	52.12	0.51
10217	7	63.03	29.81	29.66	0.422	10076	7	70.13	35.55	41.27	0.508
10229	8	62.18	30.65	28.37	0.413	10114	8	69.05	35.94	39.07	0.498
10259-1	9	62.72	29.97	29.95	0.416	10132	9	68.18	35.46	49.26	0.48
10271	10	63.06	31.12	30.26	0.424	10133	10	69.08	36.7	39.05	0.502
10174	11	62.56	29.66	31.06	0.416	10134	11	69.25	36.08	45.7	0.502
10175	12	63.64	31.63	31.25	0.431	10135	12	69.61	38	41.03	0.51
10176	13	64.15	38.41	43.64	0.449	10138	13	69.19	48.37	55.84	0.515
10177	14	63.21	39.85	43.77	0.440	10162	14	69.14	47.46	56.08	0.512
10198	15	62.32	32.88	31.16	0.427	10166	15	69.05	39.3	41.55	0.508
	average	60.837	29.892	31.53	0.401		average	71.988	38.635	46.48	0.5255

ПРИЛОЖЕНИЕ 7

Программа генерации каротажных данных[10]

Программа для генерации каротажных данных или, так называемых, синтетических скважин была разработана на языке C++.

Рассмотрим типичный пример использования:

```
for(int i = 0; i < 30; i++){
        boreholeLyth = vector<int>();
        boreholeRealKS = vector<double>();
        boreholeMeasuredKS = vector<double>();
        boreholeRealPS = vector<double>();
        boreholeMeasuredPS = vector<double>();

        init();
        generateLyth(1000);
        measure();
        string filename = "Test";
        char *a = new char[5];
        string num = itoa(i, a, 10);
        filename += num;
        filename += ".txt";
        outputBorehole(filename);
}
return 0;
```

Данный код проводит генерацию 30 скважин глубиной 1000 точек. BoreholeLyth – вектор, содержащий коды литотипов для каждых 10 сантиметров синтетической скважины. BoreHoleRealKS, BoreHoleRealPS – вектора, содержащие сгенерированные значения КС и ПС соответственно. Эти вектора заполняются в процессе выполнения функции generateLyth. BoreholeMeasuredKS и boreholeMeasuredPS – вектора, содержащие измеренные значения КС и ПС, заполняемые в процессе выполнения функции measure. Запись происходит в файлы по скважинам, с помощью функции outputBorehole.

Функция init производит чтение параметров генерации с файлов конфигурации. Рассмотрим содержимое, формат и выбранные в процессе работы с экспертами-геологами параметры.

[10] автор идеи - Кучин Я.И., реализация программы - Якунин К.О.

Файлы конфигурации litho_PS_conf, litho_KS_conf и litho_depth_conf задают, соответственно, границы и функцию распределения для генерации значений ПС, КС и мощности пласта.

Формат этих файлов:

<mode> <lyth> <begin> <end> <?numberOfCoofs> <?coofs>
<mode> <lyth> <begin> <end> <?numberOfCoofs> <?coofs>
…

Где mode – выбор функции распределения.

i – нормальное распределение, обрезанная по 2 сигмам. Математическое ожидание (mean) находится в середине интервала [begin; end], а дисперсия (stDev) выбрана равной (mean - begin)/deviation, где deviation = 2. То есть, с вероятностью 2 сигма значения попадают в интервал [begin; end], а остальные значения приравниваются границам интервала.

c – задание дискретной функции распределения с помощью коэффициентов.

Lyth – код литотипа

Begin и end – минимальные и максимальные возможные значения показателя.

NumberOfCoofs и coofs – параметры, необходимые при выборе mode=c – количество и значения коэффициентов дискретной функции распределения.

Примеры:

i 7 2 20 – параметр для породы 7 будет генерироваться на интервале от 2 до 20, используя нормальное распределение с обрезкой по 2 сигмам

c 4 25 45 5 1 3 4 6 1 – параметр для породы 4 будет генерироваться на интервале от 25 до 45, функция распределения задаётся 5 дискретными значениями (1 3 4 6 1). То есть, вероятность выбора значения 25 составляет 1/(1+3+4+6+1), значения 30 - 3/(1+3+4+6+1) и т.д. Точность задания функции распределения может быть любой, т.е., она может задаваться любым количеством коэффициентов

instrument_coof_KS_conf и instrument_coof_PS_conf – коэффициенты, задающие влияние соседних значений КС и ПС на результаты имитации измерения (функция measure). Вектора BoreHoleRealKS, BoreHoleRealPS обрабатываются плавающим окном, длиной равной количеству коэффициентов, происходит подсчёт взвешенного среднего и результат записывается в вектора BoreholeMeasuredKS и boreholeMeasuredPS.

Поскольку выбор режима генерации значений должен задаваться независимо, был использован полиморфический подход: абстрактный класс Randomizer реализуется классами IntervalRandmoizer и

CoofRandomizer (при этом возможно расширить функционал, добавив другие варианты генерации). Такой подход (паттерн проектирования Стратегия) позволяет задавать алгоритмы генерации независимо для каждого параметра и каждого литотипа.

map<int, Randomizer*> lythotopesKS;
map<int, Randomizer*> lythotopesPS;
map<int, Randomizer*> lythotopesDepth;

Эти объекты map содержат указатели на объекты классов, реализующих (наследующихся от) Randomizer.

В процессе работы с экспертами были выбраны следующие параметры генерации:

litho_depth_conf
i 7 2 20
i 4 2 50
i 3 2 50
i 123 2 50
i 12 2 50
i 59 2 5
litho_KS_conf
i 7 3 7
i 4 5 10
i 3 8 12
i 123 10 14
i 12 12 17
i 59 16 46
litho_PS_conf
i 7 16 25
i 4 12 17
i 3 10 14
i 123 8 12
i 12 5 10
i 59 3 7
instrument_coof_KS_conf
0.03333 0.06666 0.093333 0.113333 0.12666 0.13333 0.12666 0.113333 0.093333 0.06666 0.03333
instrument_coof_PS_conf
0.25 0.5 0.25

ПРИЛОЖЕНИЕ 8

Листинг программы генерации каротажных данных[11]

```
#include<iostream>
#include<fstream>
#include<map>
#include<vector>
#include<time.h>
#include<random>
#include<string>

using namespace std;

const int deviation = 2;
default_random_engine generator;

class Randomizer;

map<int, Randomizer*> lythotopesKS;
map<int, Randomizer*> lythotopesPS;
map<int, Randomizer*> lythotopesDepth;
vector<int> setLyth;

vector<double> instrCoofKS;
vector<double> instrCoofPS;

vector<int> boreholeLyth;
vector<double> boreholeRealKS;
vector<double> boreholeMeasuredKS;
vector<double> boreholeRealPS;
vector<double> boreholeMeasuredPS;

////////////////////////////////////////////////////////
class Randomizer{

public:

  double begin, end;
  virtual int getRandomInt() = 0;
  virtual double getRandomDouble() = 0;
};

class IntervalRandmoizer : public Randomizer{

public:

  IntervalRandmoizer(){

  }

  IntervalRandmoizer(double b, double e){
```

[11] Программа написана Якуниным К.О.


```
        begin = b;
        end = e;
  }

  int getRandomInt(){

        double mean = (begin + end)/2.0;
        double stDev = (mean - begin)/deviation;
        normal_distribution<double> distributionD(mean, stDev);
        int depth = (int)(distributionD(generator)+0.5);
        return depth;
  }

  double getRandomDouble(){

        double mean = (begin + end)/2.0;
        double stDev = (mean - begin)/deviation;
        normal_distribution<double> distribution(mean, stDev);
        double value = distribution(generator);

        return value;
  }
};

class CoofRandomizer : public Randomizer{

public:
  vector<int> probCoofs;

  CoofRandomizer(){

  }

  CoofRandomizer(vector<int> pc, int b, int e){

        probCoofs = pc;
        begin = b;
        end = e;

  }

  int getRandomInt(){

        int sum = 0;
        for(int i = 0; i < probCoofs.size(); i++){

              sum += probCoofs[i];
        }

        int index = 0;
        int r = rand()%sum;
        sum = 0;
```

```
		for(int i = 0; i < probCoofs.size(); i++){

			sum += probCoofs[i];
			if(r < sum){

				break;

			}

			index++;
		}

		//cout << "!!!" << probCoofs.size() << "	" << begin
<< " " << end << " " <<  ((end - begin)/(double)probCoofs.size()) << " " <<
index << endl << endl << endl;
		int value = (int)(begin + ((end -
begin)/(double)probCoofs.size())*(double)index);

		return value;
	}

	double getRandomDouble(){
		int sum = 0;
		for(int i = 0; i < probCoofs.size(); i++){

			sum += probCoofs[i];
		}

		int index = 0;
		int r = rand()%sum;
		sum = 0;
		for(int i = 0; i < probCoofs.size(); i++){

			sum += probCoofs[i];
			if(r < sum){

				break;
			}

			index++;
		}

		//cout << "!!!" << probCoofs.size() << "	" << begin
<< " " << end << " " <<  ((end - begin)/(double)probCoofs.size()) << " " <<
index << endl << endl << endl;
		double value = (begin + ((end -
begin)/(double)probCoofs.size())*(double)index);

		return value;
	}

};
```



```
//////////////////////////////////////////////

void init(){

  ifstream f;
  f.open("litho_KS_conf.txt");

  int lyth; double begin, end; char mode;

  while(f >> mode){
	if(mode == 'i'){
		f >> lyth >> begin >> end;

		lythotopesKS[lyth] = new IntervalRandmoizer(begin,
end);

		setLyth.push_back(lyth);
	}
	if(mode == 'c'){

		vector<int> c;
		int t, n;
		f >> lyth >> begin >> end >> n;
		for(int i = 0; i < n; i++){

			f >> t;
			c.push_back(t);

		}

		lythotopesKS[lyth] = new CoofRandomizer(c, begin,
end);

		setLyth.push_back(lyth);
	}
}

f.close();

f.open("litho_PS_conf.txt");

while(f >> mode){

	if(mode == 'i'){
		f >> lyth >> begin >> end;

		lythotopesPS[lyth] = new IntervalRandmoizer(begin,
end);

	}
	if(mode == 'c'){

		vector<int> c;
		int t, n;
		f >> lyth >> begin >> end >> n;
		for(int i = 0; i < n; i++){
```

```
                        f >> t;
                        c.push_back(t);

                    }
                    lythotopesPS[lyth] = new CoofRandomizer(c, begin,
end);

            }
        }

        f.close();

        if(lythotopesKS.size() != lythotopesPS.size()){

            cout << "INVALID CONFIGURATION!!!" << endl;
            exit(0);
        }

        f.open("litho_depth_conf.txt");
        while(f >> mode){

            if(mode == 'i'){

                f >> lyth >> begin >> end;

                lythotopesDepth[lyth] = new
IntervalRandmoizer(begin, end);

            }
            if(mode == 'c'){

                vector<int> c;
                int t, n;
                f >> lyth >> begin >> end >> n;
                for(int i = 0; i < n; i++){

                    f >> t;
                    c.push_back(t);

                }

                lythotopesDepth[lyth] = new CoofRandomizer(c,
begin, end);
            }
        }

        f.close();

        if(lythotopesKS.size() != lythotopesDepth.size()){

            cout << "INVALID CONFIGURATION!!!" << endl;
            exit(0);
```



```
    }

    f.open("instrument_coof_KS_conf.txt");

    while(f >> begin){

        instrCoofKS.push_back(begin);
    }

    f.close();

    f.open("instrument_coof_PS_conf.txt");

    while(f >> begin){

        instrCoofPS.push_back(begin);

    }
    f.close();
}

void generateLyth(int size){

    int overKS = 0, overDepth = 0;

    int prevLyth = -1;

    //vector<int> bell(9, 0);
    for(int i = 0; i < size;){

        int r;
        int lyth;

        r = rand()%setLyth.size();
        lyth = setLyth[r];

        while(lyth == prevLyth){

            r = rand()%setLyth.size();
            lyth = setLyth[r];
        }
        prevLyth = lyth;

        int depth = lythotopesDepth[lyth]->getRandomInt();
        if(depth > ((lythotopesDepth[lyth]))->end){
            depth = ((lythotopesDepth[lyth]))->end;
            overDepth++;
        }
        if(depth < ((lythotopesDepth[lyth]))->begin){
            depth = ((lythotopesDepth[lyth]))->begin;
            overDepth++;
        }
```

```
            //bell[(depth/2)-1]++;////////
            cout << depth << " " << lyth << endl;

            for(int j = i; j < i + depth; j++){
                    if(j >= size){

            //cout << "!!!!!" << overDepth << " " << overKS;
                            return;
                    }
                    boreholeLyth.push_back(lyth);
                    double value = lythotopesKS[lyth]-
>getRandomDouble();
                    if(value < ((lythotopesKS[lyth]))->begin){
                            value = ((lythotopesKS[lyth]))->begin;
                            overKS++;
                    }
                    if(value > ((lythotopesKS[lyth]))->end){
                            value = ((lythotopesKS[lyth]))->end;
                            overKS++;
                    }
                    boreholeRealKS.push_back(value);
                    value = lythotopesPS[lyth]->getRandomDouble();
                    if(value < ((lythotopesPS[lyth]))->begin){
                            value = ((lythotopesPS[lyth]))->begin;
                    }
                    if(value > ((lythotopesPS[lyth]))->end){
                            value = ((lythotopesPS[lyth]))->end;
                    }
                    boreholeRealPS.push_back(value);
            }
            i = i + depth;
        }
        /*
        cout << "BELL!!" << endl;
        for(int i = 0; i < 9; i++){
                cout << bell[i] << " ";
        }
        cout << endl;*/
      }

      void measure(){

        for(int i = 10; i < boreholeLyth.size() - 1; i++){

                double measValue = 0;
                for(int j = -10; j <= 0; j++){
                        measValue +=
instrCoofKS[j+10]*boreholeRealKS[i+j];
                }
                boreholeMeasuredKS.push_back(measValue);
        }

        for(int i = 10; i < boreholeLyth.size() - 1; i++){

                double measValue = 0;
                for(int j = -1; j <= 1; j++){
                        measValue += instrCoofPS[j+1]*boreholeRealPS[i+j];
                }
                boreholeMeasuredPS.push_back(measValue);
        }
      }
      void outputBorehole(string filename){
        ofstream file;
        file.open(filename);

        for(int i = 10; i < boreholeLyth.size() - 1; i++){

                file << boreholeMeasuredKS[i-10] << "\t" <<
boreholeRealKS[i] << "\t" << boreholeMeasuredPS[i-10] << "\t" <<
boreholeRealPS[i] << "\t" << boreholeLyth[i] << endl;
                //file << i << "\t" << boreholeMeasuredKS[i-10] << "\t" <<
boreholeMeasuredPS[i-10] << "\t" << boreholeLyth[i] << "\t" << 0 << "\t" <<
0 << endl;
        }

        file.close();
      }

      int main(){
        srand(time(NULL));

        for(int i = 0; i < 30; i++){
                boreholeLyth = vector<int>();
                boreholeRealKS = vector<double>();
                boreholeMeasuredKS = vector<double>();
                boreholeRealPS = vector<double>();
                boreholeMeasuredPS = vector<double>();

                init();
                generateLyth(1000);
                measure();
                string filename = "Test";
                char *a = new char[5];
                string num = itoa(i, a, 10);
                filename += num;
                filename += ".txt";
                outputBorehole(filename);
        }
        return 0;
      }
```

Научное издание

Мухамедиев Равиль Ильгизович

МЕТОДЫ МАШИННОГО ОБУЧЕНИЯ В ЗАДАЧАХ ГЕОФИЗИЧЕСКИХ ИССЛЕДОВАНИЙ

Корректор *Голубева Ж.*
Оператор верстки *Мухамедиева Е.Л.*

Тираж 500 экз.

$$J(\Theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{k=1}^{K}\right.$$

$$\left.+(1-y_k^{(i)})\log(1-h_\Theta(x^{(i)}))_k\right]+\frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{i=1}^{S_l}\sum_{j=1}^{S_{l+1}}(\Theta_{ji}^{l})^2$$

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

$$p = \frac{(x - x_{\min})(b-a)}{(x_{\max} - x_{\min})} + a$$