МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«Южно-Уральский государственный университет»
(национальный исследовательский университет)
«Высшая школа экономики и управления»
Кафедра «Информационные технологии в экономике»

РАБОТА ПРОВЕРЕНА
Рецензент, доцент каф. «Менеджмент», ЮУрГУ. к.т.н., доцент
_____ /А.И. Демченко/
«____» _____2018 г.

ДОПУСТИТЬ К ЗАЩИТЕ
Заведующий кафедрой «Информационные технологии в экономике», д.т.н., с.н.с.
_____ /Б.М. Суховилов/
«____» _____ 2018 г.

МЕДИЦИНСКАЯ СТАТИСТИЧЕСКАЯ АНАЛИТИЧЕСКАЯ СИСТЕМА ОПРЕДЕЛЕНИЯ ДИАГНОЗА НА ОСНОВЕ АЛГОРИТМОВ МАШИННОГО ОБУЧЕНИЯ

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ
ЮУрГУ – 09.04.03.2018.886.ВКР

Руководитель проекта, доцент, к.т.н.
_____ /Е.М. Сартасов/
«____» _____ 2018 г.

Автор проекта,
студент группы ЭУ-221
_____ /М.Д. Садофьев/
«____» _____ 2018 г.

Нормоконтролер, доцент,
_____ /Е.А. Конова/
«____» _____ 2018 г.

Челябинск 2018

АННОТАЦИЯ

Садофьев М.Д. Медицинская статистическая аналитическая система определения диагноза на основе алгоритмов машинного обучения. Челябинск ЮУрГУ, ЭУ – 221; 2018. – 53 с., 2 ил., 15 табл., библиогр. список – 24 наим.

Аналитические системы на основе Data Mining и машинного обучения применяются в медицине более 10 лет [13]. Они позволяют извлекать из медицинских показателей скрытые закономерности на основе которых можно прогнозировать развитие болезней. Выполнение такой работы человеком трудоемко, требует привлечения специалистов высокой квалификации и не всегда возможно. Данное утверждение объясняет актуальность дипломной работы.

Целью работы является разработка программного и математического обеспечения медицинской аналитической системы.

Основная тема заключается в применении современных средств анализа данных при разработке медицинской аналитической системы.

Результатами работы являются: проект по разработке математического и программного обеспечений медицинской аналитической системы.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Аналитические системы на основе алгоритмов машинного обучения успешно применяются в медицинской сфере более 10 лет [13]. Такие системы позволяют обрабатывать медицинские клинические данные пациентов в совокупности с сопутствующей демографической информацией (возраст, пол, место жительства и т.д.) и выявлять скрытые закономерности. На основе полученных закономерностей можно ставить диагнозы, прогнозировать развитие болезней и многое другое.

Выполнение такой работы человеком не всегда рентабельно, поскольку анализ может оказаться трудоемким и требовать привлечения специалистов высокой квалификации. Кроме того, закономерности, скрытые в данных, не всегда могут быть обнаружены человеком. Частичная автоматизация процесса, которую обеспечивают аналитические системы, позволяет сократить время на выполнение анализа, а значит сделать его дешевле. Снижение цены благоприятно сказывается на распространении Data Mining, особенно в сфере коммерческой медицины.

В контексте российской медицины, применение таких систем часто осложнено организационными и финансовыми проблемами. В свою очередь открытые для бесплатного использования библиотеки машинного обучения (например, Scickit Learn, TensorFlow, Pandas, и т. д.) для на сегодняшний день при грамотном использовании позволяют достичь высоких результатов [13] без глубоких научных изысканий и программирования сложных систем.

Таким образом, целесообразно создание медицинской статистической аналитической системы на основе алгоритмов машинного обучения, которая поможет медицинским аналитикам получить доступ к современным мощным и бесплатным библиотекам машинного обучения и использовать их в своей работе. При этом им не будут требоваться дополнительные знания языков программирования таких как Python или R.

Цель магистерской работы – разработка медицинской информационной системы для анализа исходных данных пациента и предсказания его диагноза и стадии протекания болезни.

Задачи магистерской работы:

1) анализ информационных медицинских аналитических систем;

2) классификация задач и обзор научных работ, посвящённых анализу данных в сфере медицины;

3) постановка задачи для проведения исследования и разработки математического и программного обеспечений;

4) обзор существующих методов интеллектуального анализа данных;

5) анализ научных работ по использованию механизмов машинного обучения в медицине и описание примеров использования механизмов;

6) разработка математического и программного обеспечений медицинской аналитической системы;

7) разработка плана коммерциализации проекта.

При работе над магистерской работой использовались научная и научно-исследовательская литература.

ГЛОССАРИЙ

Таблица 1 – Глоссарий

Термин, сокращение	**Определение**
Б	Бинарный
Д	Дата
К	Количественный
Математическое обеспечение	Совокупность математических методов, моделей, алгоритмов обработки информации, используемых при решении задач в информационной системе (функциональных и автоматизации проектирования информационных систем)
МД	Мало данных. Меньше порога в 1500 записей. Порог выбран эмпирически на основании просмотра исходных данных.
НПИ	Нет полезной информации для эксперимента.
ПД	Персональные данные. Нет доступа для проведения анализа. Кроме того, не несут полезной аналитической информации.
С	Строка
ХД	Хранилище данных
ЦПТ	Чисто с плавающей точкой
ЦЧ	Целое число

ГЛАВА 1 ТЕОРЕТИЧЕСКАЯ БАЗА

1.1 Анализ информационных медицинских аналитических систем

В литературе не встречается однозначно определённое понятие медицинской информационно-аналитической системы.

Прежде всего определим понятие и классификацию медицинской информационной системы (МИС). Различные определения МИС и классификации МИС приведены в работах [22] и [23]. Например, в [23] даётся следующее определение: «Совокупность информационных, организационных, программных и технических средств, предназначенных для автоматизации медицинских процессов и(или) организаций.»

С.А. Гаспарян [22] определяет МИС, как одну из форм организации медицинской деятельности, позволяющая медицинскому персоналу при соответствующей технологической поддержке использовать комплекс математических и технических средств, обеспечивающих сбор, хранение, обработку, анализ и выдачу медицинской информации».

Отсюда можно сделать вывод, что МИС является автоматизированной системой, обеспечивающей сбор, хранение, обработку, анализ и выдачу медицинской информации [23].

1.2 Хранилища данных

Один из авторитетных специалистов в этой области – Б.Инмон (Bill Inmon) определяет хранилища данных (ХД) как «предметно-ориентированные, интегрированные, стабильные, поддерживающие хронологию наборы данных, организованные для целей поддержки управления, призванные выступать в роли «единого и единственного источника истины», обеспечивающего менеджеров и аналитиков достоверной информацией, необходимой для оперативного анализа и принятия решений». Ценность ХД для экономистов заключается в следующем: ХД – это некая база данных масштаба предприятия, которая содержит определенную аналитическую ин-

формацию, обеспечивает ее оперативное представление в удобном для пользователя виде и обладает структурой, учитывающей отраслевую специфику деятельности организации. Типичные представители программных продуктов этой категории: SAP Business Warehouse (SAP), Informatica.

1.3 OLAP-средства

Под термином OLAP, как правило, понимают системы аналитической обработки данных в режиме реального времени. OLAP-системы обеспечивают решение многих аналитических задач: анализ ключевых показателей деятельности, маркетинговый и финансово-экономический анализ, анализ сценариев, моделирование, прогнозирование и т.д. Такие системы могут работать со всеми необходимыми данными, независимо от особенностей информационной инфраструктуры компании. С точки зрения пользователя, отличие OLAP-системы от хранилища данных заключается в предметной (а не технической) структурированности информации, при этом пользователю предоставляется возможность оперировать привычными экономическими категориями и понятиями. К типичным представителям программных продуктов этого класса относятся: Hyperion Essbase (Hyperion Solutions Corporation), Oracle OLAP (Oracle), MS Analysis Services (Microsoft), Business Objects (Business Objects), Cognos PowerPlay (Cognos), MicroStrategy.

1.4 Информационно-аналитические системы

Этот класс аналитических систем включает множество разнообразных продуктов, основная задача которых – предоставить конечные решения для менеджеров-аналитиков. Например, для банковской сферы реализованы методики дистанционного анализа, внутреннего и внешнего анализа, анализа прибыльности, рейтинговой оценки надежности банка (CAMEL), расчет рейтинга надежности банка (на основе методики В.С.Кромонова), расчет лимита межбанковского кредитования (на основе методики КБ «Европейский Трастовый Банк»), GAP-анализ.

1.5 Инструменты конечного пользователя

Инструменты конечного пользователя для выполнения запросов и построения отчетов (query and reporting tools). Такие системы обеспечивают функции построения запросов к информационно-аналитическим системам (в пользовательских терминах), интеграцию данных из нескольких источников, просмотр данных с возможностью детализации и обобщения, построение полноценных отчетов и их печать. Они предназначены для пользователей, обладающих «продвинутыми» техническими навыками. При этом профессиональных знаний в области информационных технологий не требуется, тем не менее, для экономистов такие средства не всегда бывают удобны. Как правило, модули, содержащие функции Query & Reporting, входят в состав многих OLAP-систем, но есть и отдельные программные продукты этого класса. Таким образом, четко провести грань между OLAP и Query & Reporting невозможно. Характерный пример – приложение Hyperion Essbase, которое аналитики относят к обоим классам.

В заключение подведем некоторые итоги классификации.

Во-первых, очевидно, что отнести тот или иной программный продукт к какому-то одному классу не всегда возможно, поскольку многие системы позволяют решать аналитические задачи нескольких категорий. К числу «многофункциональных» можно отнести системы таких мировых производителей, как Hyperion Solutions Corp., Cognos, Business Objects, Microsoft. Эти компании являются лидерами мирового рынка систем делового интеллекта, их продукты также активно продаются в России. Типичным примером универсальной системы может служить Hyperion Essbase – аналитическая платформа класса OLAP, предназначенная для решения довольно широкого круга задач. Будучи OLAP-системой, Hyperion Essbase также решает часть задач, относящихся к информационно-аналитическим системам, средствам интеллектуального извлечения данных, а также обеспечивает функции программных средств построения запросов и отчетов. Кроме того, в некоторых случа-

ях Hyperion Essbase может использоваться в качестве хранилища данных, а также в качестве аналитической «прослойки» в крупных компаниях, где данные распределены по многим информационным источникам.

Во-вторых, в настоящее время наибольшим спросом на рынке пользуются хранилища данных, OLAP-средства и системы data mining. Они обладают богатыми аналитическими возможностями, в том числе в части финансовых и статистических функций, которые постоянно развиваются и улучшаются. При этом они позволяют хранить и обрабатывать большие объемы информации.

В-третьих, при выборе аналитической системы необходимо учитывать степень простоты освоения и эксплуатации программы пользователями-экономистами, не владеющими техническими знаниями в профессиональном объеме. Иначе говоря, программный продукт должен быть настраиваемым под конечных пользователей и требовать при этом минимальной поддержки со стороны технических специалистов. Например, упомянутый выше Hyperion Essbase позволяет обеспечить всю рутинную работу, оставив аналитику только ту часть, которая касается собственно анализа и представления данных.

В-четвертых, при выборе аналитической системы также следует учитывать ее приспособленность к решению конкретных, интересующих конечного пользователя задач. В лучшем случае это реализуется в виде готовых отраслевых решений в конкретной предметной области.

1.6 Классификация задач и обзор научных работ, посвящённых анализу данных в сфере медицины

Современную медицину невозможно представить без использования точных и надёжных методов анализа и прогнозирования. На основании научных работ в данной сфере можно определить следующие основные классы задач анализа данных в сфере медицины:

1) задачи медицинской диагностики;

2) задачи анализа изображений (томография, рентгеновские снимки и т.п.);

3) задачи классификации и кластеризации;

4) задачи предсказания (например, предсказание заболеваемости).

Далее приведены примеры научных работ, посвящённых анализу данных в сфере медицины по выделенным задачам, а также дано краткое описание каждой из задач.

Задачи медицинской диагностики

В последние годы благодаря применению современных методов интеллектуального анализа данных Data Mining, действующих на основе правил, формализующих экспертные знания, стало возможным получение хороших результатов в медицинской диагностике [15].

Например, в работе [22] для решения задач диагностики патологии сердца, диагностики состояния щитовидной железы с наибольшей эффективностью используются методы наивного байесовского классификатора (NB), хотя его отличие от других методов несущественно. В данном исследовании для анализа данных использовалась система RapidMiner.

Другой пример, в работе [20] для решения задачи диагностики остеопороза и оценки риска остеопоротического перелома с наибольшей эффективностью используются методы Data Mining – методы исчисления вероятностей, байесовы и нейронные сети.

Задачи анализа изображений (томография, рентгеновские снимки)

В медицинских информационных системах наибольший объем занимают есть изображения, например, рентгеновские снимки, результаты магнитно-резонансной томографии и т.п. Это огромная часть медицинских данных, которыми надо эффективно управлять.

Использование метода главных компонент PCA (principal component analysis) для локализации анатомических частей сетчатки продемонстрирова-

ли в работе [13]. Метод показал хорошее совпадение с разметкой опытного офтальмолога, сделанной на 73 изображениях.

В работах [1, 4, 6, 7, 9, 10] для классификации сосудов на основе снимков магнитно-резонансной томографии использовались предварительно обученные нейронные сети. В [1, 4] представлен KNN классификатор (k-nearest neighbor classifier), изначально предложенный Staal, где каждому пикселю ставилась в соответствие вероятность его принадлежности сосуду. Искусственные нейронные сети используют весовые коэффициенты для определения соответствия между входными и выходными данными. Они могут быть настроены с использованием обучающей выборки на основе сетей обратного распространения ошибки. Пиксели окна, скользящего по изображению, использовались в качестве входных данных сети.

Задачи классификации и кластеризации

Главное назначение кластерного анализа – разбиение множества исследуемых объектов и признаков на однородные, в соответствующем понимании, группы или кластеры. Это означает, что решается задача классификации данных и выявления соответствующей структуры в ней. Методы кластерного анализа можно применять в самых различных случаях, даже в тех случаях, когда речь идет о простой группировке, в которой все сводится к образованию групп по количественному сходству. В частности, метод кластерного анализа (метод Уорда на базе пакета Statistica) применяется для задачи анализ показателей физиологических реакций бронхолегочной системы в ответ на психофизиологическое воздействие (аудиовизульную стимуляцию).

Другой пример, решение задачи идентификации состояний кровообращения пациентов по индивидуальным гемодинамическим функциям, характеризующим взаимодействие сердца и сосудов в процессе продвижения крови, и функциональную диагностику нарушений кровообращения, в том числе клинически латентных. В рамках решения данной задачи использовал-

ся Support Vector Machine (SVM, или машина опорных векторов) – алгоритм Data Mining.

Задачи предсказания

На текущий день в связи с развитием электронных медицинских карт, созданием межрегиональных медицинских баз данных в сфере здравоохранения и медицины происходит накопление большего объёма медицинских данных. Данная тенденция позволяет решать задачи прогнозирования на основе анализа данных о пациентах.

Например, для решения задач по предсказанию сердечных заболеваний с наибольшей эффективностью используются методы наивного байесовского классификатора, дерева решений [2, 11, 11].

1.7 Описание задачи

Для исследования использована база данных с информацией о мониторинге злокачественных новообразований у детей и подростков. Информация содержит персональные данные, которые не будут использованы в исследовании. Сбор информации был начат в конце 80-х годов, и позже оцифрован [23]. Данные используются сотрудниками больницы для исследований, но поскольку анализ выполняется подручными средствами (MS Excel), то соотношение результативности и трудозатрат низкое. На текущий момент база данных содержит примерно две тысячи записей. Потребителю нужна система, позволяющая упростить анализ данных.

База данных содержит в себе медицинские статистические данные о мониторинге злокачественных новообразований у детей и подростков в совокупности с сопутствующей демографической информацией (возраст, пол, место жительства и т.д.). На основе анализа базы данных возможно решить задачу прогнозирования факта смерти пациента. Задача прогнозирования смертности в разных проекциях (как популяционные показатели смертности, так и частная оценка вероятности выживаемости) в области заболевания зло-

качественными новообразованиями является актуальной, что подтверждается различными научными работами в сфере медицины [18, 20].

Выводы по главе 1

Определено понятие медицинской информационно-аналитической системы – комплекс аппаратных, программных средств, информационных ресурсов, методик, которые используются для обеспечения автоматизации аналитических работ для решения задачи сферы медицины. Проведён анализ аналитического программного обеспечения, определена его сегментация:

- средства построения хранилищ и витрин данных (data warehouse);
- инструменты оперативной аналитической обработки (On-Line Analytical Processing, OLAP) и прочие средства многомерного анализа;
- информационно-аналитические системы (Enterprise Information Systems, EIS) и системы поддержки и принятия решений (Decision Support Systems, DSS);
- средства интеллектуальной добычи данных (data mining);
- инструменты конечного пользователя для выполнения запросов и построения отчетов (query and reporting tools).

На основании характеристики каждого класса можно сделать вывод, что наиболее богатыми аналитическими возможностями обладают хранилища данных, OLAP-средства и системы data mining. Данное утверждение также подтверждается анализом задач и обзором научных работ, посвящённых анализу данных в сфере медицины, в ходе которого выделено четыре основных класса задач:

1) задачи медицинской диагностики;
2) задачи анализа изображений (томография, рентгеновские снимки и т.п.);
3) задачи классификации и кластеризации;
4) задачи предсказания (например, предсказание заболеваемости).

По каждому из классов приведены примеры конкретных аналитических задач в области медицины, большая часть и которых была решена средствами интеллектуальной добычи данных, что позволяет говорить о широкой распространенности использования механизмов интеллектуального анализа данных в медицине.

Кроме того, на основе проведённого анализа можно сделать вывод, что для различных задач медицинской аналитики эффективными оказываются различные методы, выбор которых связан со значительными затратами времени специалистов в области анализа данных и не может быть сделан медиками. Это подтверждает необходимость автоматизации процесса анализа с помощью специализированной системы, которая будет требовать от медиков минимальных экспертных знаний в области интеллектуального анализа данных.

В заключении данной главы определенна задача для проведения исследования, на примере решения которой будет разработано математическое и программное обеспечение медицинской аналитической системы: прогнозирование факта смерти пациента, больного злокачественным новообразованием на основе базы данных с информацией о мониторинге злокачественных новообразований у детей и подростков.

ГЛАВА 2 МЕТОДЫ ИНТЕЛЛЕКТУАЛЬНОГО АНАЛИЗА

2.1 Обзор методов интеллектуального анализа данных

Целью DataMining является нахождение таких моделей, которые не могу быть найдены обычными методами. И существует два вида моделей: предсказательные и описательные [20].

Предсказательные модели: позиционируются на наборе данных с известными результатами. И используются для предсказания результатов на основании других наборов данных. Это модели классификации (описывают правила, по которым можно отнести описание объекта к одному из классов) и модели последовательностей (они описывают функции, по которым можно прогнозировать изменение непрерывных числовых параметров).

Описательные модели: они уделяют особое внимание сути зависимостей в наборе данных, взаимному влиянию различных факторов, построению эмпирических моделей. Являются легкими для восприятия человеком.

Согласно классификации по стратегиям, задачи Data Mining подразделяются на следующие группы:

- обучение с учителем;
- обучение без учителя;
- другие.

Категория обучение с учителем представлена следующими задачами Data Mining: классификация, оценка, прогнозирование.

Категория обучение без учителя представлена задачей кластеризации.

В категорию другие входят задачи, не включенные в предыдущие две стратегии.

Data Mining – это не один метод, а совокупность большого числа различных методов обнаружения знаний. Базовыми методами, которые может найти технология DataMining, согласно В. А. Дюку являются [16]:

1. **Ассоциация** – применяется, когда несколько событий связаны

между собой. Например, исследования показали, что 59 % купивших чипсы берут также и газированную воду, а если есть скидка на такой комплект, то газированную воду приобретают в 79 % случаев. Если менеджеры располагают подобными данными, то им достаточно легко оценить действенность предполагаемой скидки. Наиболее известный алгоритм решения задачи поиска ассоциативных правил – алгоритм Apriori.

2. **Классификация** – выявление черт, которые будут характеризовать группу, к которой принадлежит объект, на основе обучения на уже классифицированных объектах. Для решения задачи классификации могут использоваться методы: ближайшего соседа (Nearest Neighbor); k-ближайшего соседа (k-Nearest Neighbor); байесовские сети (Bayesian Networks); индукция деревьев решений; нейронные сети (neural networks).

3. **Кластеризация** – отличается от классификации тем, что группы заранее не известны и средства DataMining самостоятельно выявляют различные однородные группы данных. Пример метода решения задачи кластеризации: обучение «без учителя» особого вида нейронных сетей – самоорганизующихся карт Кохонена.

4. **Последовательность** – применяется при существовании цепочки событий, связанных во времени. Например, при приобретении квартиры в течение месяца приобретается кухонная плита в 49 % случаев, а в течение трех недель - холодильник в 73 %.

5. **Прогнозирование** – создание или нахождение шаблонов, которые будут истинно показывать тенденция поведения необходимых показателей по временным рядам. При помощи них можно предсказать поведение системы в будущем. Для решения таких задач широко применяются методы математической статистики, нейронные сети и др.

2.2 Примеры использования механизмов машинного обучения в медицине

Предсказание сердечных заболеваний (пример 1)

Исследование описывалось в 2010 году [11].

В исследовании использовался инструмент для анализа данных Tanagra (Lumière University Lyon 2).

Использовалось 3 алгоритма. Набор данных состоял из 3000 записей с 14 признаками и был разделен на тренировочный и тестовый в соотношении 70 / 30. Результаты приведены в Таблица 2.

Таблица 2 – Результаты исследования

№п/п	**Инструмент**	**Точность (%)**	**Время (мс)**
1	Naïve Bayes	52,33	609
2	Decision List	52	719
3	KNN	45, 67	1000

Предсказание сердечных заболеваний (пример 2)

Исследование описывалось в 2008 году [11].

Описанная система названа Intelligent Heart Disease Prediction System и реализована на .net фреймворке.

Использовалось 2 классических алгоритма и нейронная сеть. Набор данных состоял из 900 записей с 15 признаками и был разделен на тренировочный и тестовый в сочетании приблизительно 50 / 50. Результаты приведены в Таблица 3

Таблица 3 – Результаты исследования

№ п/п	Инструмент	Точность (%)
1	Naïve Bayes	86,53
2	Decision Tree	89
3	Neural Network	85,53

Предсказание сердечных заболеваний (пример 3)

Исследование проводилось в 2010 году [2].

В исследовании использовался инструмент для анализа данных Weka (University of Waikato).

Набор данных состоял из 909 записей с 13 признаками. Результаты приведены в Таблица 4.

Таблица 4 – Результаты исследования

№п/п	Инструмент	Точность (%)
1	Naïve Bayes	96,5
2	Decision Tree	99,2
3	Classification via clustering	88,3

Масштаб применения Data Mining в медицине и предсказание успешности искусственного оплодотворения (пример 4)

В исследовании 2013 года, проанализированы данные об инструментах Data Mining применяемых в медицине [7].

Информация сведена в Таблица 5.

Таблица 5 – Результаты исследования

№п/п	Заболевание	Инструмент	Вид анализа	Алгоритм	Точность (%)
1	Сердечные заболевания	ODND, NCC2	Classification	Naïve Bayes	60
2	Рак	WEKA	Classification	Rules, Decision Table	97.77
3	ВИЧ / СПИД	WEKA	Classification, Association Rule Mining	J48 (Decision Tree)	81.8
4	Банк крови	WEKA	Classification	J48 (Decision Tree)	89.9
5	Рак мозга		Clustering	MAFIA	85
6	Туберкулез	WEKA	Classification	Naïve Bayes	78
7	Сахарный диабет	ANN	Classification	C4.5 (Decision Tree)	82.6
8	Гемодиализ	RST	Classification	Decision Making	75.97

Продолжение таблицы 5

№п/п	Заболевание	Инструмент	Вид анализа	Алгоритм	Точность (%)
9	Тропическая лихорадка	SPSS Modeller		C5.0 (Decision Tree)	80
10	Искусственное оплодотворение	ANN, RST	Classification		91
11	Гепатит С	SNP	Information gain	Decision Rule	73.2

В том же источнике описано исследование для предсказания успешности искусственного оплодотворения (IVF – In vitro fertilization).

Для начала, используется теория приближенных множеств (Таблица 6).

Таблица 6 – Теория приближённых множеств

Факт	Предсказание		
	Успех	Неудача	Точность (%)
Успех	17	4	80.952
Неудача	26	10	27.777
Точность (%)	39.5349	71.4286	47.368

Затем искусственная нейронная сеть с обратным распространением (Таблица 7).

Таблица 7 – Искусственная нейронная сеть с обратным распространением

№п/п	Показатели ошибки	Предсказание	
		Неудача	Успех
1	MSE	0.209522132	0.212860733
2	NMSE	1.164459543	1.18301446
3	MAE	0.23114814	0.25780224
4	Min Abs Error	9.90854E-07	6.66044E-06
5	Max Abs Error	1.015785003	0.998857054
6	R	0.498099362	0.498099362
	Точность (%)	73.07692308	75

Затем комбинация этих методов (Таблица 8).

Таблица 8 – Комбинация теории приближённых множеств и искусственной нейронной сети с обратным распространением

№п/п	Показатели ошибки	Предсказание	
		Неудача	Успех

1	MSE	0.092835478	0.110601021
2	NMSE	0.378803726	0.451293836
3	MAE	0.14313612	0.191653959
4	Min Abs Error	0.002563409	0.005851654
5	Max Abs Error	1.055555499	1.055555556
6	R	0.789058201	0.789058201
	Точность (%)	89.23076923	91.83673469

Сравнение точности методов (Таблица 9).

Таблица 9 – Сравнение точности методов

	Теория приближенных множеств	**Нейронная сеть**	**Комбинация**
Точность в предсказании успеха	47	73	90

Выводы по главе 2

Часто параметры по умолчанию оказываются для алгоритмов, реализованных в современных мощных библиотеках машинного обучения достаточными для достижения качественного результата, что подтверждает тезис о возможности их использования различными специалистами, не имеющими глубоких знаний в алгоритмах машинного обучения. Но таким специалистам требуется предоставить доступ к инструментам, на что и нацелена описываемая система.

Как указано выше, в настоящее время, различные инструменты анализа данных доступны бесплатно и представлены в различных программных библиотеках. Создание простой аналитической системы, которая будет давать медицинским аналитикам доступ к этим библиотекам определенно востребовано.

Система должна иметь достаточно простой и расширяемый интерфейс

для доступа к базе данных, в которой хранится вся информация. Расширяемость интерфейса может достигаться не инструментами в пользовательском интерфейсе, а понятным и доступным кодом, чтобы после передачи системы заказчику он мог расширить интерфейс (вводимые поля) своими силами. Система управления базами данных (СУБД) также должна выбираться с расчетом расширения.

Система должна иметь в своем составе инструменты интеллектуального анализа данных. При этом следует разделить часть системы, отвечающую за ввод данных и анализ данных.

Для обучения и проверки выборку необходимо разделить в соотношении 70/30 на обучающую и тестовую соответственно и выполнить классификацию с использованием Naïve Bayes, Decision Tree, Random Forest, Neural Network, KNN.

ГЛАВА 3 РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ СИСТЕМЫ АНАЛИЗА ДАННЫХ

3.1 Разработка программного обеспечения системы анализа данных

Для простоты реализации и расширения система реализуется на языке программирования Python 3, который является одним из самых популярных и имеет значительное количество доступных библиотек машинного обучения. Система разделена на две основные части.

1. Подсистема ввода, хранения и управления данными.
2. Подсистема анализа данных.

Общая схема системы представлена на Рисунок 1 в виде компонентной диаграммы нотации UML.

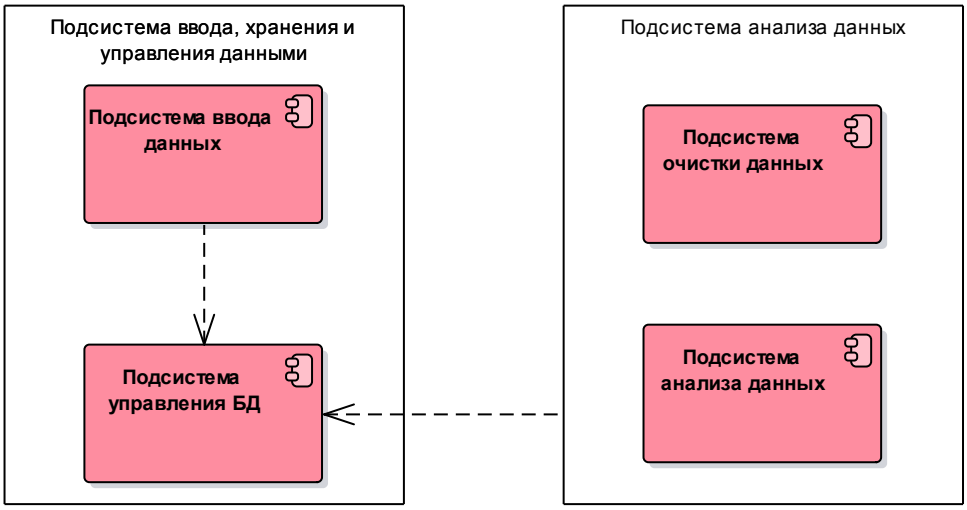

Рисунок 1 – Общая схема системы

Система анализа данных представляет собой web-сервис, который будет доступен по внутренней сети больницы, и будет принимать и отдавать данные в заданном формате.

Подсистема ввода, хранения и управления данными реализована на основе фреймворка Django работающего на основе Python 3. Подсистема анализа данных реализуется с нуля, с использованием различных библиотек

для решения конкретных задач в контексте системы. Так, например, для машинного обучения используются библиотеки Scikit Learn, TensorFlow, Pandas, Numpy, для доступа к системе через сеть библиотека Flask, и т.д.

В качестве СУБД для хранения данных выбрана PostgreSQL, поскольку фреймворк Django, на основе которого реализована подсистема ввода, хранения и управления данными имеет встроенную поддержку PostgreSQL, она относится к категории свободного программного обеспечения и в сети интернет доступно много информации об этой СУБД.

Набор данных, хранимых в системе обеспечивает требования Приказа N 135 от 19 апреля 1999 года Министерства Здравоохранения Российской Федерации «О совершенствовании системы Государственного ракового регистра» [23].

3.2 Описание исходных данных

Для исследования используется набор данных о пациентах состоящий из 72 признаков включающий в себя 1929 записей. В анализе не участвуют признаки, которые с точки зрения текущего исследования не несут ценности. В качестве целевой переменной выбирается факт смерти пациента. Однако такой переменной в данных нет, но есть признак «Дата смерти». Это признак заполнен датой или оставлен пустым. Следует преобразовать этот признак чтобы получить бинарный признак со значением да (1) и нет (0). Если в исходном признаке «Дата смерти» указана дата, следовательно, в преобразованном признаке устанавливается значение 1, иначе 0. Описания всех признаков представлены в Таблица 10.

Таблица 10 – Описание признаков в исходном наборе данных

№	Кодовое имя	Описание	Тип данных	Кол-во не пустых	Примечание
1	center	Номер медицинского центра, в	ЦЧ	-	Не участвует Одно значение

№	Кодовое имя	Описание	Тип данных	Кол-во не пустых	Примечание
		котором проводилось обследование			во всех записях
2	ident	ID пациента *	ЦЧ	-	Не участвует. ПД
3	fname	Имя и отчество	С	-	Не участвует. ПД
4	lastname	Фамилия	С	-	Не участвует. ПД
5	index	Почт. индекс	С	-	Не участвует. ПД
6	address	Адрес	С	-	Не участвует. ПД
7	adr_telephon	Телефон	С	-	Не участвует. ПД
8	bdate	Дата рождения	Д	1917	Будет преобразовано в признак «Возраст на момент постановки диагноза»
9	sex	Пол	ЦЧ	1929	
10	fddate	Дата обращения	Д	1657	Не участвует. НПИ
11	tsdate	Дата обследования	Д	1618	Не участвует. НПИ
12	tdate	Дата постановки диагноза	Д	1921	Будет преобразовано в признак «Возраст на момент постановки диагноза»

№	Кодовое имя	Описание	Тип данных	Кол-во не пустых	Примечание
13	edate	Дата последнего события (опухоли и т.д.)	Д	219	Не участвует. НПИ, МД
14	otcode	Неизвестно	ЦЧ	929	Не участвует. МД
15	ddate	Дата смерти	Д	359	Будет преобразовано в целевой признак «Факт смерти»
16	death_icd	Код причины смерти по МКБ	С	12	Не участвует. МД
17	ecode	Описание последнего события (опухоли и т.д.)	ЦЧ	786	Не участвует. НПИ, МД
18	esource	Источник сведений о событии	ЦЧ	380	Не участвует. МД
19	dcause	Причина смерти	ЦЧ	984	Не участвует. НПИ
20	dsource	Источник сведений о смерти	ЦЧ	540	Не участвует. МД
21	numtumor	№ опухоли (1-я, 2-я и т.д.)	ЦЧ	439	Не участвует. МД
22	iccc	Классификация заболевания в соответствии с международной классификации детских злокачественных опухолей	С	1912	
23	icd_10	Классификация заболевания в соответствии с МКБ-10	С	1610	

№	Кодовое имя	Описание	Тип данных	Кол-во не пустых	Примечание
24	icd_o	Классификация заболевания в соответствии с МКБ-О	С	1603	
25	fds	Диагноз	С	1906	Не участвует. Строка без определенного формата. Сложно извлечь информацию
26	circdetect	Как обнаружено заболевание (например, обратился сам, регулярный осмотр и т.д.)	ЦЧ	1628	Не участвует. НПИ
27	hyst	Гистология (факт)	ЦЧ	1899	
28	cyto	Цитология (факт)	ЦЧ	1887	
29	exp_oper	Операция (факт)	ЦЧ	1865	
30	immun	Иммуногистохимия (факт)	ЦЧ	1865	
31	cytogen	Цитогенетика (факт)	ЦЧ	1884	
32	lab_instr	Лабораторно-инстр. данные (факт)	ЦЧ	1868	
33	incentr	Пациент получал лечение в центре	ЦЧ	1028	Не участвует. НПИ
34	lasttest	Дата последнего обновления	Д	1822	Не участвует. НПИ
35	lastsource	Последний источник обновления	ЦЧ	769	Не участвует. НПИ, МД

№	Кодовое имя	Описание	Тип данных	Кол-во не пустых	Примечание
36	lfudate	Дата потери из-под наблюдения	Д	123	Не участвует. МД
37	lfucode	Причина потери из-под наблюдения	ЦЧ	595	Не участвует. МД
38	otregion	Пациент из другого региона	ЦЧ	1878	
39	date_actend	Неизвестно	Д	252	Не участвует. МД
40	concord	Получено согласие на лечение	ЦЧ	1862	
41	protocol	Протокол лечения	С	280	Не участвует. МД
42	autopsie	Проводилось ли вскрытие	С	1204	Не участвует. НПИ, МД
43	tabort	Терапия прервана	ЦЧ	703	Не участвует. МД
44	stage	Стадия заболевания	ЦЧ	1836	
45	stage_sym	Доп. символ к стадии заболевания	ЦЧ	19	Не участвует. МД
46	chirurg	Хирургический этап лечения (факт)	ЦЧ	983	Отсутствующие строки перевести в значение «нет»
47	radiolog	Лучевая терапия (факт)	ЦЧ	980	Отсутствующие строки перевести в значение «нет»
48	chimio	Химиотерапия (факт)	ЦЧ	1000	Отсутствующие строки перевес-

№	Кодовое имя	Описание	Тип данных	Кол-во не пустых	Примечание
					ти в значение «нет»
49	rdate	Дата достижения ремиссии	Д	409	Не участвует. МД
50	rem	Примечание к ремиссии	С	130	Не участвует. МД
51	t_tnm	Классификация по TNM компонент T	ЦЧ	1856	-
52	n_tnm	Классификация по TNM компонент N	ЦЧ	1855	-
53	m_tnm	Классификация по TNM компонент M	ЦЧ	1856	-
54	g_tnm	Классификация по TNM гистологическая степень злокачественности	ЦЧ	1854	-
55	encr_bas	Неизвестно	ЦЧ	1694	-
56	ct_tnm	Клиническая стадия TNM компонент T (факт)	ЦЧ	1374	Не участвует. МД
57	cn_tnm	Клиническая стадия TNM компонент N (факт)	ЦЧ	1374	Не участвует. МД
58	cm_tnm	Клиническая стадия TNM компонент M (факт)	ЦЧ	1374	Не участвует. МД
59	land	Регион проживания	ЦЧ	1904	-
60	lum	Лимфоузлы (факт	ЦЧ	1869	-

№	Кодовое имя	Описание	Тип данных	Кол-во не пустых	Примечание
		поражения)			
61	oss	Кости (факт поражения)	ЦЧ	1868	-
62	hepar	Печень (факт поражения)	ЦЧ	1866	-
63	lung	Легкое (факт поражения)	ЦЧ	1867	-
64	brain	Головной мозг (факт поражения)	ЦЧ	1865	-
65	skin	Кожа (факт поражения)	ЦЧ	1866	-
66	nephro	Почка (факт поражения)	ЦЧ	1865	-
67	herm	Половые органы (факт поражения)	ЦЧ	1865	-
68	periton	Брюшина (факт поражения)	ЦЧ	1865	-
69	kmark	Костный мозг (факт поражения)	ЦЧ	1865	-
70	unknown	Неизвестно	ЦЧ	1865	Не участвует. НПИ
71	other	Другое	ЦЧ	1866	Не участвует. НПИ
72	acc	Неизвестно	ЦЧ	1929	-

* - все признаки начиная с 2 относятся к пациенту.

3.3 Подготовка данных к анализу

Обработка данных выполняется в несколько этапов.

1. Из набора данных удаляются строки содержащие пустые значе-

ния для признаков при условии, что последний не предполагает пустых значений. При этом исходный признак удалятся и создаётся новый вычисляемый признак. Таким образом, например, дополняются признаки *chirurg*, *radiolog*, *chimio* (поз. 46, 47, 48 Таблица 10).

2. Сырые данные преобразуются в тип понятный системе, в соответствии с описаниями, переданными в параметрах метода. Это по большей части исключительно технический этап. Подавляющее большинство из передаваемых в метод *prepare* данных возможно уже находится в подходящем формате, однако явное приведение типа позволяет ожидать от системы большей стабильности и предсказуемости в процессе обработки данных.

3. Добавляются вычисляемые признаки. Как указанно выше в контексте системы под вычисляемыми признаками понимаются признаки которых нет в исходном наборе данных и которые формируются и добавляются в набор данных на основании определённой логики. Например, на основе признаков *tdate* и *bdate* (поз. 8 и 12 Таблица 10) будет сформирован новый признак *age* – возраст на момент постановки диагноза.

4. Удаляются выбросы в соответствии с описаниями, переданными в параметрах метода. Например, в исходных данных в вычисляемом поле *age* есть несколько значений больше 18 (лет), что в контексте данных явно является ошибкой, поскольку исследование проводится только по данным несовершеннолетних пациентов. Установив порог в 0.99 можно избавится от таких значений.

5. Все категориальные признаки преобразуются в бинарные. Например, признак *cytogen* (поз. 34 Таблица 10) содержит значения 0, 1. Однако эти признаки не являются количественными поскольку означают лишь метку и не могут быть сравнены между собой. Такое сравнение может исказить результаты анализа. В подобных случаях следует удалить исходный признак и создать несколько новых бинарных признаков на основе исходного. Например, необходимо удалить признак *cytogen* и создать признаки *cytogen_0*,

cytogen_1 каждый из которых будет содержать 0 или 1 в зависимости от того, какое значение исходного признака содержала строка (0 - не содержала, 1 - содержала). Такие признаки также будут вычисляемыми, поскольку они отсутствуют в поступающем наборе данных. Отличие в том, что они создаются автоматически без участи пользователя, лишь на основании типа признака (категориальные).

6. Удаляются не нужные признаки. Например, как указанно выше на основании признаков *tdate* и *bdate* сформирован новый признак *age*. Таким признак age содержит в себе информацию которую содержали признаки *tdate* и *bdate*, соответственно необходимость в этих признаках в наборе данных отпадает, и они могут быть удалены. В рамках отладочного эксперимента признаки будут удалены, однако при проведении других экспериментов удаление какого-либо из признаков зависит от параметров, передаваемых пользователем.

Набор данных полученный после обработки методом классом Prepare представлен в Таблица 11. При этом следует отметить что все признаки, обозначенные как бинарные представлены в таблице в виде одного поля. Это сделано для упрощения восприятия таблицы. На 5 этапе работы метода *prepare* они преобразованы в несколько признаков. Например, признак *cytogen* состоящий из значений 0 и 1 в возвращаемом наборе данных представлен в виде признаков *cytogen_0* и *cytogen_1*. В нулевой позиции таблицы представлена целевая переменная *dead* для отладочного эксперимента, означающая факт смерти пациента.

Таблица 11 – Набор данных после обработки методом prepare

№	Кодовое имя	Описание	Тип данных	Тип признака
0	dead	Факт смерти пациента	ЦЧ	Б
1	age	Возраст на момент постановки диагно-	ЧПТ	К

№	Кодовое имя	Описание	Тип данных	Тип признака
		за		
2	sex	Пол	ЦЧ	Б
3	iccc	Классификация заболевания в соответствии с международной классификации детских злокачественных опухолей	ЦЧ	Б
4	icd_10	Классификация заболевания в соответствии с МКБ-10	ЦЧ	Б
5	icd_o	Классификация заболевания в соответствии с МКБ-О	ЦЧ	Б
6	hyst	Гистология (факт)	ЦЧ	Б
7	cyto	Цитология (факт)	ЦЧ	Б
8	exp_oper	Операция (факт)	ЦЧ	Б
9	immun	Иммуногистохимия (факт)	ЦЧ	Б
10	cytogen	Цитогенетика (факт)	ЦЧ	Б
11	lab_instr	Лабораторно-инстр. данные (факт)	ЦЧ	Б
12	otregion	Пациент из другого региона	ЦЧ	Б
13	concord	Согласие на лечение (факт)	ЦЧ	Б
14	stage	Стадия заболевания	ЦЧ	Б
15	chirurg	Хирургический этап лечения (факт)	ЦЧ	Б

№	Кодовое имя	Описание	Тип данных	Тип признака
16	radiolog	Лучевая терапия (факт)	ЦЧ	Б
17	chimio	Химиотерапия (факт)	ЦЧ	Б
18	t_tnm	Классификация по TNM компонент T	ЦЧ	Б
19	n_tnm	Классификация по TNM компонент N	ЦЧ	Б
20	m_tnm	Классификация по TNM компонент M	ЦЧ	Б
21	g_tnm	Классификация по TNM гистологическая степень злокачественности	ЦЧ	Б
22	encr_bas	Неизвестно	ЦЧ	Б
23	land	Регион проживания	ЦЧ	Б
24	lum	Лимфоузлы (факт поражения)	ЦЧ	Б
25	oss	Кости (факт поражения)	ЦЧ	Б
26	hepar	Печень (факт поражения)	ЦЧ	Б
27	lung	Легкое (факт поражения)	ЦЧ	Б
28	brain	Головной мозг (факт поражения)	ЦЧ	Б
29	skin	Кожа (факт поражения)	ЦЧ	Б
30	nephro	Почка (факт поражения)	ЦЧ	Б
31	herm	Половые органы	ЦЧ	Б

№	Кодовое имя	Описание	Тип данных	Тип признака
		(факт поражения)		
32	periton	Брюшина (факт поражения)	ЦЧ	Б
33	kmark	Костный мозг (факт поражения)	ЦЧ	Б
34	acc	Неизвестно	ЦЧ	Б

3.4 Оценка математического обеспечения механизма прогнозирования

С целью подтверждения результативности работы механизма прогнозирования аналитической системы необходимо определить метрику оценки. В различных научных работах [например, 2] приведены формулы, использующиеся для оценки точности алгоритма. Для этого используется ряд метрик, основанных на доле истинных и ложных результатов классификации True positive (TP), True negative (TN), False positive (FP), False negative (FN). Наиболее часто используемыми метриками являются следующие.

1. Доля правильно классифицированных объектов (Accuracy) показывает вероятность того, что класс будет предсказан правильно (см. формула 1):

$$Accuracy = (TP + TN) / (TP + TN + FP + FN) \quad (1).$$

2. Точность (Precision) показывает, какая доля объектов, распознанных как объекты положительного класса, предсказана верно (см. формула 2):

$$Precision = TP / (TP + FP) \quad (2).$$

3. Полнота (Recall) показывает, какая доля объектов, реально относящихся к положительному классу, предсказана верно (см. формула 3):

$$Recall = TP / (TP + FN) \quad (3).$$

4. Максимальная точность и полнота недостижимы одновременно, в связи с этим приходится искать некий баланс, который может оцениваться с помощью гармонического среднего между точностью и полнотой (F-мера) (см.

формула 4):

$$F=2*(Precision * Recall) / (Precision + Recall) \quad (4).$$

Также в научных работах [17, 16] для оценки математических моделей используется ROC-анализ – анализ с применением ROC-кривых (англ. receiver operating characteristic, рабочая характеристика приёмника), графиков, которые отображают т соотношение между долей объектов от общего количества носителей признака, верно классифицированных как несущих признак, (англ. true positive rate, TPR, называемой чувствительностью алгоритма классификации) и долей объектов от общего количества объектов, не несущих признака, ошибочно классифицированных как несущих признак (англ. false positive rate, FPR, величина 1-FPR называется специфичностью алгоритма классификации) при варьировании порога решающего правила.

В данной работе для оценки математического обеспечения будут использоваться метрики: доля правильно классифицированных объектов (Accuracy), точность (Precision), полнота (Recall) и F-мера.

3.5 Выбор определяющих признаков и определение математического обеспечения механизма прогнозирования

Для реализации этого этапа в системе предусмотрено использование метода главных компонент на основании решения sklearn.decomposition.PCA библиотеки Scikit Learn. Получаем оценку влияния признаков на факт смерти пациента, приведенную в Таблица 12. Указаны только признаки, оказывающие хотя бы минимальный видимый эффект. При этом, для упрощения анализа полученных результатов, бинарные признаки вроде sex_0 и sex_1 сведены в единую переменную sex содержащей в себе сумму влияния признаков.

Таблица 12 – Признаки, оказывающие минимальное видимое влияние на целевую переменную dead (Смерть пациента)

№	Кодовое имя	Влияние на целевую переменную
1	age	0.69328

№	Кодовое имя	Влияние на целевую переменную
2	iccc	0.16486
3	sex	0.10447
4	icd_10	0.03225
5	icd_o	0.00440
6	stage	0.00026
7	concord	0.00009
8	cyto	0.00008
9	t_tnm	0.00005
10	hyst	0.00004
11	exp_oper	0.00004
12	immun	0.00004
13	lab_instr	0.00004
14	otregion	0.00004
15	n_tnm	0.00003
16	m_tnm	0.00001
17	g_tnm	0.00001

Видно, что переменная age (возраст) оказывает максимальное значение на факт смерти пациента, что не удивительно, поскольку речь идет о пациентах детского возраста и чем пациент старше, тем сильнее и имеет выше шанс справиться с болезнью. Для исключения очевидной составляющей исключаем переменную age. Кроме того, переменные iccc, icd_10 и icd_o означают классификацию болезни по различным системам. Эти системы классификации не идентичны, но, если рассматривать их очень обобщенно, являются схожими. Таким образом использование всех 3-х переменных вероятно станет причиной мультиколлинеарности. Если, учесть, что наибольшее влияние оказывает переменная iccc то так же представляется разумным удалить переменные icd_10 и icd_o. Кроме того, удаляются все переменные, не по-

павшие в приведенный список и не оказывающие сколько-нибудь заметного влияния на целевую переменную. После этого снова оцениваем влияние переменных с помощью метода главных компонент. Результаты представлены в Таблица 13.

Таблица 13 – Признаки, оказывающие минимальное видимое влияние на целевую переменную dead (Смерть пациента) (Повторная оценка)

№	Кодовое имя	Влияние на целевую переменную
1	iccc	0.58944
2	sex	0.34824
3	stage	0.01674
4	concord	0.00629
5	land	0.00491
6	t_tnm	0.00453
7	hyst	0.00350
8	g_tnm	0.00339
9	cyto	0.00334
10	n_tnm	0.00330
11	exp_oper	0.00321
12	immun	0.00300
13	encr_bas	0.00285
14	lab_instr	0.00284
15	otregion	0.00271
16	m_tnm	0.00171

Как видно из предыдущей таблицы вес оставшихся признаков во влиянии на целевую переменную значительно вырос. Хотя для большинства остается незначительным. На следующем этапе эксперимента будут использованы признаки из предыдущей таблицы, и отдельно признаки *iccc* и *sex*

имеющие наибольший вес, представленный в Таблица 14.

Таблица 14 – Признаки, оказывающие максимальное влияние на целевую переменную dead (Смерть пациента) (Повторная оценка)

№	Кодовое имя	Влияние на целевую переменную
1	iccc	0.54201
2	sex	0.45799

К данным описанными признаками, представленными в Таблица 13 и Таблица 14 последовательно были применены несколько алгоритмов машинного обучения: Логистическая регрессия, Решающее дерево, Random Forest, Gradient Tree, Наивный Байес для нормального распределения, Наивный Байес для распределения Бернулли. Все алгоритмы используются внутри классов-оберток, которые разделяют общий интерфейс, реализованный в виде соглашения с оговоренным набором методов с обозначенной сигнатурой.

В течении эксперимента мониторинг выполнялся по следующим параметрам: правильность (accuracy), точность (precision), полнота (recall), ф-мера (f-score), время. Ф-мера является основным параметром для сравнения. Остальные величины взяты для построения полной картины. Время оценивается только для получения общего представления о сравнении времени работы алгоритмов. Для оценки указанных параметров набор данных делится на обучающий и тестовый в сочетании 0,75 на 0,25. Результаты приведены в Таблица 15.

Таблица 15 – Результаты отладочного эксперимента

№	Алгоритм	Правильность (accuracy)	Точность (precision)	Полнота (recall)	Ф-мера (f-score)	Время
1	K-ближайших соседей (Таблица 13 – 1552 записи)	0.802	0.734	0.802	0.752	0.125

№	Алгоритм	Правильность (accuracy)	Точность (precision)	Полнота (recall)	Ф-мера (f-score)	Время
2	K-ближайших соседей (Таблица 14 – 1911 записей)	0.789	0.729	0.789	0.748	0.125
3	Логистическая регрессия (Таблица 13 – 1552 записи)	0.814	0.772	0.814	0.778	0.031
4	Логистическая регрессия (Таблица 14 – 1911 записей)	0.81	0.665	0.81	0.73	0.016
5	Решающее дерево (Таблица 13– 1552 записи)	0.765	0.745	0.765	0.754	0.016
6	Решающее дерево (Таблица 14 – 1911 записей)	0.81	0.745	0.81	0.75	~ 0.0
7	Random Forest (Таблица 13– 1552 записи)	0.822	0.782	0.822	0.781	0.234
8	Random Forest (Таблица 14 – 1911 записей)	0.81	0.745	0.81	0.75	0.234
9	Gradient Tree (Таблица 13– 1552 записи)	0.82	0.778	0.82	0.779	0.281
10	Gradient Tree (Таблица 14 – 1911 записей)	0.814	0.752	0.814	0.747	0.141
11	Наивный Байес для нормального распределения (Таблица 13– 1552 записи)	0.353	0.786	0.353	0.364	~ 0.0
12	Наивный Байес для нор-	0.236	0.734	0.236	0.173	0.016

№	Алгоритм	Правильность (accuracy)	Точность (precision)	Полнота (recall)	Ф-мера (f-score)	Время
	мального распределения (Таблица 14 – 1911 записей)					
13	Наивный Байес для распределения Бернулли (Таблица 13 – 1552 записи)	0.804	0.762	0.804	0.773	0.016
14	Наивный Байес для распределения Бернулли (Таблица 14 – 1911 записей)	0.812	0.751	0.812	0.752	~ 0.0

Выводы по главе 3

Часто параметры по умолчанию оказываются для алгоритмов, реализованных в современных мощных библиотеках машинного обучения достаточными для достижения качественного результата, что подтверждает тезис о возможности их использования различными специалистами, не имеющими глубоких знаний в алгоритмах машинного обучения. Но таким специалистам требуется предоставить доступ к инструментам, на что и нацелена описываемая система.

Алгоритмы выдают довольно высокие результаты.

ГЛАВА 4 КОММЕРЦИАЛИЗАЦИЯ ПРОЕКТА

4.1 Актуальность коммерциализации

Различные задачи медицинской аналитики требуют использования разных методов анализа, выбор которых связан со значительными затратами времени специалистов в области анализа данных и не может быть сделан медиками. Это подтверждает актуальность автоматизации процесса анализа с помощью специализированной системы, которая будет требовать от медиков минимальных экспертных знаний в области интеллектуального анализа данных.

Описанная в главе 3 архитектура Системы позволяет специалистам клинической области с минимальным погружением в специфику Data Science решать задачи медицинской аналитики.

Кроме того, структура хранения данных обеспечивает требования Приказа N 135 от 19 апреля 1999 года Министерства Здравоохранения Российской Федерации «О совершенствовании системы Государственного ракового регистра» [23]. В настоящее время в России используются несколько компьютерных программ территориального популяционного регистра, разработанных на единой методологической основе, но продолжают функционировать и программы с недостаточным объемом вводимой информации, чаще всего это программы, созданные в 80-х годах прошлого века. Функциональность данных программ ограничена задачами хранения, накопления данных и формирования выборок данных.

4.2 Цели и задачи

Целью является создание аналитической медицинской системы, назначение которой заключается в накоплении, хранении и анализе данных о мониторинге злокачественных новообразований у детей и подростков.

Для достижения поставленной цели в рамках первого года развития проекта необходимо выполнить задачи в соответствии с планом работ (см.

Рисунок).

Название задачи	Длительность	Начало	Окончание
Определение требований к системе и подготовка проектной документации	28 дней	Пн 14.01.19	Ср 20.02.19
Подписание проектной документации	3 дней	Чт 21.02.19	Пн 25.02.19
Спецификация требований к системе и системный анализ	30 дней	Вт 26.02.19	Пн 08.04.19
Разработка структуры хранения данных	20 дней	Вт 09.04.19	Пн 06.05.19
Миграция данных	10 дней	Вт 07.05.19	Пн 20.05.19
Разработка подсистема ввода, хранения и управления данными	28 дней	Вт 21.05.19	Чт 27.06.19
Заполнение основных справочников	14 дней	Пт 28.06.19	Ср 17.07.19
Разработка подсистемы анализа данных	32 дней	Чт 18.07.19	Пт 30.08.19
Настройка и тестирование	21 дней	Пн 02.09.19	Пн 30.09.19
Подготовка эксплутационной документации	10 дней	Вт 01.10.19	Пн 14.10.19
Обучение персонала	21 дней	Вт 15.10.19	Вт 12.11.19
Опытная эксплуатация	14 дней	Ср 13.11.19	Пн 02.12.19
Внесение коррективов	14 дней	Вт 03.12.19	Пт 20.12.19
Приёмка работ	1 день	Пн 23.12.19	Пн 23.12.19
Ввод в эксплуатацию	7 дней	Вт 24.12.19	Ср 01.01.20

Рисунок 2 – Календарный план работ

1. Определение требований к системе и подготовка проектной документации.
2. Подписание проектной документации.
3. Описание спецификации требований к системе и системный анализ.
4. Разработка структуры хранения данных.
5. Миграция данных.
6. Разработка подсистемы ввода, хранения и управления данными в соответствии с разработанным в ходе магистерской работы программным обеспечением.
7. Заполнение основных справочников.
8. Разработка подсистемы анализа данных в соответствии с разработанными в ходе магистерской работы программным обеспечением и математическим обеспечением.
9. Настройка и тестирование.

10. Подготовка эксплуатационной документации.
11. Обучение персонала.
12. Опытная эксплуатация.
13. Внесение коррективов в систему по итогам опытной эксплуатации.
14. Приёмка работ.
15. Ввод в эксплуатацию.

Выводы по главе 4

В данной главе составлена дорожная карта коммерциализации проекта на два года. Кроме того, составлен календарный план работ на первый год коммерциализации проекта. Предполагаемый срок разработки и тестирования такого программного продукта – 1 год.

Различные задачи медицинской аналитики требуют использования разных методов анализа, выбор которых связан со значительными затратами времени специалистов в области анализа данных и не может быть сделан медиками, что подтверждает актуальность коммерциализации проекта.

ЗАКЛЮЧЕНИЕ

В рамках работы определено понятие медицинской информационно-аналитической системы – комплекс аппаратных, программных средств, информационных ресурсов, методик, которые используются для обеспечения автоматизации аналитических работ для решения задачи сферы медицины. Проведён анализ аналитического программного обеспечения.

Проведён анализ задач и обзор научных работ, посвящённых анализу данных в сфере медицины, в ходе которого выделено четыре основных класса задач.

1) задачи медицинской диагностики;
2) задачи анализа изображений (томография, рентгеновские снимки и т.п.);
3) задачи классификации и кластеризации;
4) задачи предсказания (например, предсказание заболеваемости).

Определенна задача для проведения исследования, на примере решения которой разработан проект математического и программного обеспечения медицинской аналитической системы: прогнозирование факта смерти пациента, больного злокачественным новообразованием на основе базы данных с информацией о мониторинге злокачественных новообразований у детей и подростков.

Для разработки математического программного обеспечения был проведён обзор существующих методов интеллектуального анализа данных. Проведён анализ научных работ по использованию механизмов машинного обучения в медицине и описаны примеры их использования.

Разработан проект реализации программного обеспечения медицинской аналитической системы: система разделена на две основные части.

1) Подсистема ввода, хранения и управления данными.
2) Подсистема анализа данных.

В качестве СУБД для хранения данных выбрана PostgreSQL, подсис-

тема ввода, хранения и управления данными будет реализована на основе фреймворка Django работающего на основе Python 3. Подсистема анализа данных реализуется, с использованием различных библиотек для решения конкретных задач в контексте системы.

Определено математическое обеспечение системы: наиболее эффективными на примере решаемой задачи показали себя алгоритмы Random Forest (0,781) и Gradient Tree (0,779).

Кроме того, составлена дорожная карта коммерциализации проекта на два года и составлен календарный план работ на первый год коммерциализации проекта. Предполагаемый срок разработки и тестирования такого программного продукта – 1 год.

Таким образом, решены все поставленные в данной работе задачи и основную цель магистерской работы можно считать достигнутой.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Abramoff, M. Web-based screening for diabetic retinopathy in a primary care population: The eye check project / M. Abramoff, M. Suttorp // Telemedicine and e-Health. - 2005. - Vol. 11(6). - P. 668-674.

2. ANBARASI M., ANUPRIYA E., N.CH.S.N.IYENGAR, Enhanced Prediction of Heart Disease with Feature Subset Selection using Genetic Algorithm, International Journal of Engineering Science and Technology Vol. 2(10), 2010, 5370-5376.

3. Cao Z., Cao S., Xiong G., Guo L.Progress in Study of Encrypted Traffic Classification. In Proceedings of International standard conference on trustworthy computing and services, 2012, Beijing, China, pp. 78-86

4. Gregory, S. Nearest-neighbor methods in learning and vision: theory and practice / S. Gregory, D. Trevor, I. Piotr // Neural Information Processing / MIT Press, 2006.

5. Iqbal, M.I. Detection of vascular intersection in retina fundus image using modified cross point number and neural network technique / A.M. Aibinu, M. Nilsson, I.B. Tijani more authors // Int. Conf. Comput. Commun. Eng. - 2008. - P. 241-246.

6. Jan, J. Retinal image analysis aimed at blood vessel tree segmentation and early detection of neural-layer deterioration / J. Jan, J. Odstrcilik, J. Gazarek, R. Kolar // Computerized Medical Imaging and Graphics. - 2012. - Vol. 36(6). - P. 431-441.

7. Kheng, G.G. An automatic diabetic retinal image screening system book chapter in medical data mining and knowledge discovery / G.G. Kheng, H.S. Wynne, M. Li, H. Wang // Edited by Krzysztof Cios. - 2001. - Vol. 29. - P. 181-210.

8. M. Durairaj, V. Ranjani, Data Mining Applications In Healthcare Sector: A Study, International Journal of Engineering Science and Technology Vol. 2(10), 2013, 2277-8616.

9. Marin, D. A new supervised method for blood vessel segmentation in retinal images by using gray-level and moment invariants-based features / D. Marin, A. Aquino, M.E. Gegundez-Arias, J.M. Bravo // IEEE Transactions on Medical Imaging. - 2011. - Vol. 30(1). -P. 146-158.

10. Newey, V.R. Online artery diameter measurement in ultrasound images using artificial neural networks / V.R. Newey, D.K. Nassiri // Ultrasound Med. Biol. - 2002. - Vol. 28(2). - P. 209-216.

11. Rajkumar Asha, G.Sophia Reena, Diagnosis Of Heart Disease Using Datamining Algorithm, Global Journal of Computer Science and Technology 38 Vol. 10 Issue 10 Ver. 1.0 September 2010.

12. Sellappan Palaniappan Rafiah Awang, Intelligent Heart Disease Prediction System Using Data Mining Techniques, IJCSNS International Journal of Computer Science and Network Security, VOL.8 No.8, August 2008.

13. Sinthanayothin, C. Automated localisation of the optic disc, fovea, and retinal blood vessels from digital colour fundus images / C. Sinthanayothin, J. Boyce, H. Cook, T. Williamson // British Journal of Ophthalmology. - 1999. - Vol. 83(8). - P. 902-910.

14. Tomar D., Agarwal S. A survey on Data Mining approaches for Healthcare // International Journal of Bio-Science and Bio-Technology. – 2013. – Vol. 5 № 5. – P. 241-266.

15. Баевский Р.М. Прогнозирование состояний на грани нормы и патологии. — М.: Медицина, 1979. — 298 с.

16. Барсегян А.А., Куприянов М.С, Степаненко В.В.,. Холод И.И. Технологии анализа данных: DataMining, VisualMining, TextMining, OLAP : учеб. пособие. 2-е изд. / СПб.: 2007. 59 с.

17. Богданов Л. Ю. Оценка эффективности бинарных классификаторов на основе логистической регрессии методом ROC-анализа // Вестник СГТУ. 2010. №2с. URL: https://cyberleninka.ru/article/n/otsenka-effektivnosti-binarnyh-klassifikatorov-na-osnove-logisticheskoy-regressii-metodom-roc-analiza

(дата обращения: 26.05.2018)

18. Гайдышев И.П. Оценка качества бинарных классификаторов // Вестник ОмГУ. 2016. №1 (79). URL: https://cyberleninka.ru/article/n/otsenka-kachestva-binarnyh-klassifikatorov (дата обращения: 26.05.2018).

19. Гладких П.Г., Короткова А.С. Прогнозирование показателей смертности населения РФ от злокачественных новообразований // Здоровье и образование в XXI веке. 2015. №4. URL: https://cyberleninka.ru/article/n/prognozirovanie-pokazateley-smertnosti-naseleniya-rf-ot-zlokachestvennyh-novoobrazovaniy (дата обращения: 26.05.2018).

20. Дмитриев Г.А., Аль-Факих Али Салех Али Система диагностики и оценки риска остеопоротического перелома на основе интеллектуального анализа данных // Программные продукты и системы. 2016. №3 (115). URL: https://cyberleninka.ru/article/n/sistema-diagnostiki-i-otsenki-riska-osteoporoticheskogo-pereloma-na-osnove-intellektualnogo-analiza-dannyh (дата обращения: 20.05.2018).

21. Золотухин О. В., Кравец Б. Б., Фирсов О. В. Результаты прогнозирования смертности от рака почки для групп территорий с близкими показателями // ВНМТ. 2006. №1. URL: https://cyberleninka.ru/article/n/rezultaty-prognozirovaniya-smertnosti-ot-raka-pochki-dlya-grupp-territoriy-s-blizkimi-pokazatelyami (дата обращения: 26.05.2018).

22. Карасева Т.С. Решение задач медицинской диагностики методами интеллектуального анализа данных // Решетневские чтения. 2015. №19. URL: https://cyberleninka.ru/article/n/reshenie-zadach-meditsinskoy-diagnostiki-metodami-intellektualnogo-analiza-dannyh (дата обращения: 20.05.2018).

23. московских вузов». Российский национальный исследовательский медицинский университет имени И. И. Пирогова Министерства здравоохранения и социального развития Российской Федерации. Научно-образовательный материал «Современные информационные технологии в

здравоохранении, комплексные АИС ЛПУ» Москва, 2011. // URL: http://rsmu.ru/fileadmin/rsmu/img/about_rsmu/assoc_mosk_vuz_soc_obsluzh_obraz/2011/ n5_68_1/nom_n5_68_1_2_1_z.pdf. Дата обращения: 20.05.2018.

24. Спичак И.И., Жукавская Е.В., Башарова Е.В., Коваленко С.Г., Волкова К.Б., Билялутдинова Д.И. Этапы становления эпидемиологического мониторинга злокачественных новообразований у детей и подростков в Челябинской области. Вопросы гематологии, онкологии и иммунологии в педиатрии.- 2013. -Том 12, № 1. - С 23-24.

ПРИМЕРЫ ИСХОДНОГО КОДА

```
class FeatureType:
    BINARY = 'binary'
    PHYSICAL = 'physical'
    CATEGORICAL = 'categorical'
    ORDINAL = 'ordinal'  # TODO how to use


class DataType:
    DATETIME = 'datetime'
    FLOAT = 'float'
    INTEGER = 'integer'
    STRING = 'string'

class Feature:
    def __init__(
            self,
            feature_name: str,
            feature_type: str,
            data_type: str,
            forbid_none: bool = True,
            quantile: Optional[float] = None,
            drop: bool = False,
            comment: str = None):
        """
        Represents a data set feature to be considered during machine
learning.
        :param feature_name: feature name.
        :param feature_type: feature type. See FeatureType constants.
        :param data_type: data type. See DataType constants.
        :param forbid_none: are None values allowed or not.
        :param quantile: a quantile to be cut.
        :param drop: should the feature be removed from the data set or
not.
        :param comment: any comment.
        """

        if not self.__are_compatible(feature_type, data_type):
            raise FeatureTypeDataTypeException('Feature type {} is not
compatible with data type {}'.format(
                feature_type, data_type))

        self.feature_name = feature_name
        self.feature_type = feature_type
        self.forbid_none = forbid_none
        self.data_type = data_type
        self.quantile = quantile
        self.drop = drop
```

```
        self.comment = comment

    def __are_compatible(self, feature_type: str, data_type: str) -> bool:
        return data_type in self.__compatibility_map.get(feature_type)

    __compatibility_map = {
        FeatureType.BINARY: [DataType.INTEGER],
        FeatureType.CATEGORICAL: [DataType.DATETIME, DataType.FLOAT,
DataType.INTEGER, DataType.STRING],
        FeatureType.PHYSICAL: [DataType.INTEGER, DataType.FLOAT,
DataType.DATETIME]
    }

    def __eq__(self, other: Any) -> bool:
        return self.__dict__ == other.__dict__

    def to_dict(self):
        return self.__dict__

    @staticmethod
    def from_any(obj: Any) -> 'Feature':
        return Feature(
            feature_name=converter.get_value(obj, 'feature_name'),
            feature_type=converter.get_value(obj, 'feature_type'),
            data_type=converter.get_value(obj, 'data_type'),
            forbid_none=converter.get_value(obj, 'forbid_none', True),
            quantile=converter.cast(float, converter.get_value(obj,
'quantile')),
            drop=converter.get_value(obj, 'drop', False),
            comment=converter.get_value(obj, 'comment'),
        )

class CalculatedFeature(Feature):
    def __init__(
            self,
            expression: str,
            feature_name: str,
            feature_type: str,
            data_type: str,
            forbid_none: bool = True,
            quantile: Optional[float] = None,
            drop: bool = False,
            comment: str = None):
        """
        Represents a calculated data set feature to be considered during
machine learning.
        This feature will be added to the data set after evaluation of the
specified expression.
```

```
        :param expression: an expression to be evaluated to create this
feature.
        :param feature_name: feature name.
        :param feature_type: feature type. See FeatureType constants.
        :param data_type: data type. See DataType constants.
        :param forbid_none: are None values allowed or not.
        :param quantile: a quantile to be cut.
        :param drop: should the feature be removed from the data set or
not.
        :param comment: any comment.
        """

        self.expression = expression

        super().__init__(
            feature_name=feature_name,
            feature_type=feature_type,
            data_type=data_type,
            forbid_none=forbid_none,
            quantile=quantile,
            drop=drop,
            comment=comment)

    @staticmethod
    def from_any(obj: Any) -> 'CalculatedFeature':
        return CalculatedFeature(
            expression=converter.get_value(obj, 'expression'),
            feature_name=converter.get_value(obj, 'feature_name'),
            feature_type=converter.get_value(obj, 'feature_type'),
            data_type=converter.get_value(obj, 'data_type'),
            forbid_none=converter.get_value(obj, 'forbid_none', True),
            quantile=converter.cast(float, converter.get_value(obj,
'quantile')),
            drop=converter.get_value(obj, 'drop', False),
            comment=converter.get_value(obj, 'comment'),
        )

class Preparer:
    """
    Allows to modify data to create a data set for machine learning.
    """
    def prepare(self, source_data_set: DataFrame, target_variable: Un-
ion[Feature, CalculatedFeature],
                features: List[Feature], calculated_features:
List[CalculatedFeature] = None) -> DataFrame:
        """
        Creates a new data set from the specified data set and feature de-
scriptions passed to the method.
        :param source_data_set: a source data set.
```

```
        :param target_variable: a machine learning target variable.
        :param features: features to be included to returned data set from
the source data set.
        :param calculated_features: features to be included to returned
data set which are calculated and not presented
        in the source data set.
        :return: a new data set with the specified features and the target
variables.
        """

        data_set = self.__select_data_with_features(source_data_set, fea-
tures)
        data_set = self.__remove_none_rows(data_set, features)

        data_set = self.__convert_to_datetime(data_set, features)
        data_set = self.__convert_to_integer(data_set, features)
        data_set = self.__convert_to_string(data_set, features)
        data_set = self.__convert_to_float(data_set, features)

        all_features = self.__add_calculated_features(data_set, features,
calculated_features)

        data_set = self.__remove_outliers(data_set, all_features)
        data_set = self.__convert_categorical_features_to_binary(data_set,
all_features)
        data_set = self.__add_target_variable(source_data_set, data_set,
target_variable)

        data_set = self.__convert_to_datetime(data_set, [target_variable])
        data_set = self.__convert_to_integer(data_set, [target_variable])
        data_set = self.__convert_to_string(data_set, [target_variable])
        data_set = self.__convert_to_float(data_set, [target_variable])

        data_set = self.__drop_unused_features(data_set, all_features)
        return data_set

    @staticmethod
    def __select_data_with_features(source_data_set: DataFrame, features:
List[Feature]) -> DataFrame:
        try:
            feature_names = [f.feature_name for f in features]
            new_data_set = source_data_set[feature_names]
        except KeyError as err:
            raise PreparerException('Can not select features.
{}'.format(str(err)))
        return new_data_set

    @staticmethod
```

```
    def __remove_none_rows(data_set: DataFrame, features: List[Feature]) -
> DataFrame:
        not_none_features_names = [f.feature_name for f in features if
f.forbid_none]
        return data_set.dropna(subset=not_none_features_names)

    def __convert_to_datetime(self, data_set: DataFrame, features:
List[Feature]) -> DataFrame:
        return self.__convert_to_type(
            DataType.DATETIME, data_set, features, lambda x: pan-
das.DatetimeIndex([pandas.to_datetime(x, format='%Y-%m-
%d')]).astype(numpy.int64).values[0] / 1000000)

    def __convert_to_integer(self, data_set: DataFrame, features:
List[Feature]) -> DataFrame:
        return self.__convert_to_type(
            DataType.INTEGER, data_set, features, lambda x: pandas.np.nan
if pandas.np.isnan(x) else int(x))

    def __convert_to_string(self, data_set: DataFrame, features:
List[Feature]) -> DataFrame:
        return self.__convert_to_type(
            DataType.STRING, data_set, features, lambda x: str(x))

    def __convert_to_float(self, data_set: DataFrame, features:
List[Feature]) -> DataFrame:
        return self.__convert_to_type(
            DataType.FLOAT, data_set, features, lambda x: pandas.np.nan if
pandas.np.isnan(x) else float(x))

    @staticmethod
    def __convert_to_type(data_type: str, data_set: DataFrame, features:
List[Feature],
                          map_function: Callable[[], Any]) -> DataFrame:
        for feature_name in [f.feature_name for f in features if
f.data_type == data_type]:
            try:
                data_set[feature_name] = da-
ta_set[feature_name].map(map_function)
            except (TypeError, ValueError):
                raise PreparerException(
                    "Feature '{}' of the '{}' type can not be converted to
the '{}' data type".format(
                        feature_name, data_set[feature_name].dtype, da-
ta_type))
        return data_set

    @staticmethod
```

```
    def __add_calculated_features(data_set: DataFrame, features:
List[Feature],
                                  calculated_features:
List[CalculatedFeature]) -> List[Feature]:
        if calculated_features is not None:
            for feature in calculated_features:
                try:
                    data_set[feature.feature_name] =
eval(feature.expression)
                except Exception as err:
                    raise PreparerException("Can not execute the '{}' ex-
pression of the '{}' column. {}".format(
                        feature.expression, feature.feature_name,
str(err)))
            features += calculated_features
        return features

    @staticmethod
    def __remove_outliers(data_set: DataFrame, features: List[Feature]) ->
DataFrame:
        for feature in [f for f in features if f.quantile is not None]:
            data_set = data_set[data_set[feature.feature_name] <
                                da-
ta_set[feature.feature_name].quantile(feature.quantile)]
        return data_set

    @staticmethod
    def __convert_categorical_features_to_binary(data_set: DataFrame, fea-
tures: List[Feature]) -> DataFrame:
        categorical_feature_names = [f.feature_name for f in features if
                                     f.feature_type ==
FeatureType.CATEGORICAL and f.drop is False]
        return modification.convert_categorical_to_binary(data_set, cate-
gorical_feature_names)

    @staticmethod
    def __add_target_variable(source_data_set: DataFrame, data_set:
DataFrame,
                              target: Union[Feature, CalculatedFeature]) -
> DataFrame:
        if type(target) is CalculatedFeature:
            try:
                data_set.insert(0, target.feature_name,
eval(target.expression))
            except Exception as err:
                raise PreparerException("Can not execute the '{}' expres-
sion of the '{}' target feature. {}".format(
                    target.expression, target.feature_name, str(err)))
        else:
```

```
            try:
                data_set.insert(0, target.feature_name,
source_data_set[target.feature_name])
            except KeyError as err:
                raise PreparerException("Can not select target feature.
{}".format(str(err)))
        return data_set

    @staticmethod
    def __drop_unused_features(data_set: DataFrame, features:
List[Feature]) -> DataFrame:
        for feature_name in [f.feature_name for f in features if f.drop]:
            try:
                data_set = data_set.drop([feature_name], axis=1)
            except ValueError as err:
                raise PreparerException('Can drop unused features.
{}'.format(str(err)))
        return data_set


class Evaluator:
    """
    Allows to evaluate machine learning algorithms applied to a data set.
    """
    # noinspection PyMethodMayBeStatic
    def evaluate(self, data_set: DataFrame, target_variable_name: str, al-
gorithm: Algorithm, scale: bool = True,
                 random_state: Optional[int]=None) -> Evaluation:
        """
        Evaluates the result of the specified machine learning algorithm
applied to the specified data set.
        :param data_set: a data set with data to be passed into a algo-
rithm.
        :param target_variable_name: a name of a column with the data tar-
get variable.
        :param algorithm: an algorithm to be tested.
        :param scale: should the data be scaled before algorithm applying
or not.
        :param random_state: a seed for a pseudo random algorithm of
splitting the data to train and test samples.
        :return: evaluation result.
        """

        x, y = modification.split_to_data_and_target_variable(data_set,
target_variable_name)
        if scale:
            x = preprocessing.scale(x)

        train_and_test_data = modification.split_to_train_and_test(x, y,
random_state=random_state)
```

```
        start_time = time.time()

        algorithm.train(x_train=train_and_test_data.x_train,
y_train=train_and_test_data.y_train)
        prediction = algorithm.predict(x_test=train_and_test_data.x_test)

        accuracy = metrics.accuracy_score(train_and_test_data.y_test, pre-
diction)
        precision = metrics.precision_score(train_and_test_data.y_test,
prediction, average='weighted')
        recall = metrics.recall_score(train_and_test_data.y_test, predic-
tion, average='weighted')
        f_score = metrics.f1_score(train_and_test_data.y_test, prediction,
average='weighted')
        elapsed_time = time.time() - start_time

        return Evaluation(algorithm_name=algorithm.name, accura-
cy=accuracy, precision=precision, recall=recall,
                          f_score=f_score, elapsed_time=elapsed_time)

    def evaluate_many(self, data_set: DataFrame, target_variable_name:
str, algorithms: List[Algorithm],
                      scale: bool = True, random_state: Optional[int] =
None) -> List[Evaluation]:
        """
        Evaluates the result of the specified machine learning algorithms
applied to the specified data set.
        :param data_set: a data set with data to be passed into a algo-
rithm.
        :param target_variable_name: a name of a column with the data tar-
get variable.
        :param algorithms: algorithms to be tested.
        :param scale: should the data be scaled before algorithm applying
or not.
        :param random_state: a seed for a pseudo random algorithm of
splitting the data to train and test samples.
        :return: evaluation result.
        """
        return [self.evaluate(data_set=data_set, tar-
get_variable_name=target_variable_name, algorithm=alg, scale=scale,
                              random_state=random_state) for alg in algo-
rithms]

    @staticmethod
    def find_best(evaluations: List[Evaluation], attr: str) -> Evaluation:
        best_val = 0
        best_eval = None
        for ev in evaluations:
```

```
    val = getattr(ev, attr)
    if val > best_val:
        best_val = val
        best_eval = ev
return best_eval
```