МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет Вычислительной техники

Кафедра Вычислительная техника

Направление подготовки 09.04.01 «Информатика и вычислительная техника»

Профиль Вычислительные машины, комплексы, системы и сети

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА МАГИСТРА

на тему

Глубокие нейронные сети. Обучение.

Студент 3.06.2020 Шаповалов Игорь Сергеевич
(подпись, дата)

Руководитель _____ Зинкин С.А.
(подпись, дата) 3.06.2020

Нормоконтролёр _____ Кучин А.В.
(подпись, дата)

Рецензент 3.06.2020 Афонин А.Ю. нач. отдела НПП, Криптософт
(подпись, дата)

Работа допущена к защите (протокол заседания кафедры от 03.06.2020 № 12)

Заведующий кафедрой _____ Митрохин М.А.
(подпись)

Работа защищена с отметкой Отлично (протокол заседания ГЭК от 16.06.20 № 8)

Секретарь ГЭК _____ Гурин Е.И.
(подпись)

Пенза, 2020

Федеральное государственное бюджетное образовательное учреждение
высшего образования
ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет Вычислительной техники

Кафедра Вычислительная техника

«Утверждаю»
Заведующий кафедрой
М.А. Митрохин
«30» 09 2019 г.

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
МАГИСТРА

1. Студент Шаповалов Игорь Сергеевич гр. 18ВВм1 факультета вычислительной техники.

2.Тема работы: «Глубокие нейронные сети. Обучение».

Тема утверждена приказом ПГУ № № 1039/о от 30.09.2019 г.

3. Руководитель работы д.т.н., профессор, Зинкин С.А.

4. Задание на работу (назначение разработки, исходные данные и т.п.).
Назначение разработки – пользовательское приложение, предназначенное для прогнозирования наличия или отсутствие болезни у пациента по заполненной информации о нём.
Возможности системы: выбор файла для обучения нейронной сети, заполнение факторов, влияющих на болезнь, прогнозирование результата.

5. Перечень подлежащих разработке вопросов.
Исследование сфер применения нейронных сетей, изучение способов обучения нейронной сети, исследование применения нейросетей в медицине, анализ преимуществ и недостатков нейронных сетей.

6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1.	Анализ предметной области	10	13.03.20	
2.	Выбор архитектуры приложения, языков программирования, среды разработки и способа взаимодействия	10	25.03.20	
3.	Определение способа организации пользовательского интерфейса	10	03.04.20	
4.	Разработка программы	20	11.04.20	
5.	Разработка пользовательского приложения	20	25.04.20	
6.	Тестирование и отладка	10	13.05.20	
7.	Оформление выпускной работы	20	26.05.20	

Дата выдачи задания "26" сентября 2019 г.

Руководитель магистерской работы _____ Зинкин С.А.
(подпись, дата) 26.09.19

Задание к исполнению принял студент _____ 26.09.19 Шаповалов И.С.
(подпись, дата)

Работу к защите допустить
Декан факультета _____ Фионова Л.Р.
(подпись, дата)

РЕФЕРАТ

Магистерская работа содержит 88 листов, 38 рисунков, 1 таблицу, 11 источников, 2 приложения.

НЕЙРОННЫЕ СЕТИ В МЕДИЦИНЕ, PYTHON, НЕЙРОН, ПРОГНОЗ.

Цель работы – разработать программу на основе нейронных сетей. Программа на основе полученных критериев пациентов предсказать наличие или отсутствие у него болезни.

Программа должна получать данные с файла и провести некоторый анализ, разделить данные на тренировочный и тестовый набор, и на основе тренировки спрогнозировать результат для каждой строки тестовых данных и посчитать процент правильных предсказанний. После чего необходимо разработать программу с простым интерфейсом для пользователя, который будет иметь возможность после заполнения факторов о наличии некоторых своих особенностей получать прогноз наличия или отсутствия болезни.

В результате проведенной работы реализован проект, удовлетворяющий поставленной задаче, а именно, создано приложение, которое с помощью тренировочного набора по болезням, а затем на основе полученного результата спрогнозировать наличие или отсутствие болезни у пациента.

Данное приложение рекомендуется использовать в области медицины, где имеется набор данных для обучения сети и необходимо предсказать наличие проблем со здоровьем, чтобы вовремя предпринять меры для предотвращения или излечения болезни.

Содержание

РЕФЕРАТ 2
Содержание 3
ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ 5
Введение 6
1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 7
1.1 Сферы применения нейронных сетей 7
1.2 Общая характеристика искусственных нейронных сетей 10
1.3 Обучение нейронной сети 13
1.4 Обоснование языка программирования 14
1.5 Выбор программной среды разработки 17
1.6 Преимущества и недостатки нейронных сетей 22
1.7 Нейросетевой анализ медицинских данных 23
1.8 Применение нейронных сетей в области анализа патологий в кровеносных сосудах 26
1.9 Глубокое обучение 29
1.9.1. Методы глубинного обучения 30
1.9.2. Важность глубинного обучения 31
1.9.3 Микросервисы глубинного изучения 32
2 РЕАЛИЗАЦИЯ ПОЛЬЗОВАТЕЛЬСКОГО ПРИЛОЖЕНИЯ 33
2.1 Постановка задачи 33
2.2 Структура данных приложений 34
2.3 Выбор архитектуры приложения и способа организации пользовательского интерфейса 34
2.4 Описание разработки программы 35
2.5 Отладка и тестирование 41
3 ОПИСАНИЕ ПРОГРАММЫ 43
3.1 Описание пользовательского приложения 43
3.2 Описание тестовых данных 43
3.3 Руководство пользователя для работы с программой 45

ЗАКЛЮЧЕНИЕ 54
СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 55
Приложение А Листинги программы 57
Приложение А.1 Файл «urinary.py» 57
Приложение А.2 Файл «design.py» 58
Приложение А.3 Файл «mainPain.py» 78
Приложение В Презентация 82

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ

ANN – Artificial Neural Networks

ИИ – Искуственный интеллект

LSTM - Long short-term memory

ML - Machine learning

ИТ - Информационные технологии

ИНС - Искусственные нейронные сети

Введение

ИИ начал проникать в науку о жизни из-за оцифрованных и часто массивных наборов данных, которые требуют анализа. Это позволяет упростить программирование машин, поскольку устраняет большую часть описания функций ячеек. ИИ должен быть показан только набор из двух типов клеток, например, раковых и нераковых, без подробного описания каждой функции.

ИИ движется к тому, чтобы трансформировать медицину и где-то даже заменить работников реальной медицины. Каждый год мы наблюдаем появление новых и более совершенных решений. Это, кстати, обеспечивает целый ряд преимуществ, одним из наиболее важных из которых является сокращение времени, необходимого для постановки диагноза, который позволяет медицинским работникам лучше расставить приоритеты в случае пациента.

Прямо сейчас уже есть достаточное количество успешных разработок. Искусственный интеллект и глубокое обучение могут анализировать гораздо больше факторов и случаев, чем работники здравоохранения. Чтобы быть более точным, мы можем использовать ИИ для исследования генома, разработки лекарств, медицинской визуализации. Устройства на основе ИИ могут изучать, анализировать большие объемы информации и принимать решения гораздо быстрее, чем люди.

В рамках данной выпускной квалификационной работы требуется разработать систему анализа данных на основе нейронных сетей с возможностью предсказания заболевания у женщин.

1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1 Сферы применения нейронных сетей

Сфер применения нейронных сетей в современном мире достаточно. Ниже приведены некоторые из областей, где используется ANN. Это предполагает, что ANN имеет междисциплинарный подход в своей разработке и применении.

1. **Распознавание речи**

Речь играет важную роль в общении человека с человеком. Поэтому для людей естественно ожидать разработку речевых интерфейсов для взаимодействия с компьютерами. В современную эпоху для общения с машинами людям по-прежнему нужны сложные языки, которые трудно изучать и использовать. Чтобы ослабить этот коммуникационный барьер, может быть простое решение - общение на разговорном языке, понятном для машины.

В этой области достигнут большой прогресс, однако такие системы все еще сталкиваются с проблемой ограниченного словарного запаса или грамматики, а также с проблемой переподготовки системы для разных говорящих субъектов в разных условиях. ANN играет важную роль в этой области. Следующие ANN были использованы для распознавания речи:

- Многослойные сети
- Многослойные сети с рекуррентными подключениями
- Карта самоорганизации Кохонена

Наиболее полезной сетью для этого является карта самоорганизации Кохонена, в которую вводятся короткие отрезки речевого сигнала. Он будет отображать тот же тип фонем, что и выходной массив, называемый методом извлечения признаков. После извлечения функций с помощью некоторых

акустических моделей в качестве внутренней обработки он распознает высказывание.

2. **Распознавание персонажей**

Это интересная проблема, которая входит в общую область распознавания образов. Многие нейронные сети были разработаны для автоматического распознавания рукописных символов, букв или цифр. Ниже приведены некоторые ANN, которые использовались для распознавания символов:

- Многослойные нейронные сети, такие как нейронные сети Backpropagation.
- Неокогнитрон

Хотя нейронные сети с обратным распространением имеют несколько скрытых слоев, схема соединения от одного уровня к другому локализована. Аналогичным образом, у неокогнитрона также есть несколько скрытых слоев, и его обучение делается слой за слоем для такого рода приложений.

3. **Приложение для проверки подписи**

Подписи являются одним из наиболее полезных способов авторизации и аутентификации человека в юридических транзакциях. Метод проверки подписи - это метод, не основанный на видении.

Для этого приложения первый подход заключается в извлечении элемента или, вернее, геометрического набора элементов, представляющего подпись. С этими наборами функций мы должны обучать нейронные сети, используя эффективный алгоритм нейронной сети. Эта обученная нейронная сеть будет классифицировать подпись как подлинную или поддельную на стадии проверки.

4. **Распознавание человеческого лица**

Это один из биометрических методов для идентификации данного лица. Это типичная задача из-за характеристики «не лица» изображения. Однако, если нейронная сеть хорошо обучена, то ее можно разделить на два класса, а именно изображения с лицами и изображения без лиц.

Во-первых, все входные изображения должны быть предварительно обработаны. Затем размерность этого изображения должна быть уменьшена. И, наконец, он должен быть классифицирован с использованием алгоритма обучения нейронной сети. Следующие нейронные сети используются в учебных целях с предварительно обработанным изображением: многослойная нейросеть с прямой связью, обученная с помощью алгоритма обратного распространения.

Для уменьшения размерности анализ главных компонентов используется CAPCA.

5. **Прогнозирование.**

Прогнозирование требуется в значительной степени при принятии повседневных деловых решений (например, продажи, распределение финансовых средств между продуктами, использование мощностей), в экономической и денежно-кредитной политике, в сфере финансов и фондового рынка. Чаще всего проблемы прогнозирования являются сложными, например, прогнозирование цен на акции является сложной проблемой с множеством основных факторов (некоторые известны, некоторые не видны). Традиционные модели прогнозирования накладывают ограничения с точки зрения учета этих сложных нелинейных отношений. ANN, применяемые правильно, могут обеспечить надежную альтернативу, учитывая ее способность моделировать и извлекать невидимые функции и взаимосвязи. Кроме того, в отличие от этих традиционных моделей, ANN не накладывает никаких ограничений на входные и остаточные распределения.

Например, в этой области ведутся дополнительные исследования - последние достижения в использовании LSTM и рекуррентных нейронных сетей для прогнозирования.

ANN являются мощными моделями, которые имеют широкий спектр применения. Выше я перечислил несколько выдающихся из них, но они имеют далеко идущие приложения во многих различных областях в медицине, безопасности, банковском деле / финансах, а также в правительстве, сельском хозяйстве и обороне.

1.2 Общая характеристика искусственных нейронных сетей

Многое из того, что человечество желает достичь искусственно, уже сделано природой. Человеческий мозг наделен великолепной нейронной сетью, изучение которой ведется и по сей день. Генетически, биологические нейронные сети устроены довольно сложно и человеку без соответствующей технической подготовки будет сложно понять процесс работы естественной нейросети.

В упрощённой трактовке можно сказать так: биологическая нейронная сеть — это та часть нервной системы, что находится в мозге человека. Совокупность нейронов и сети позволяет нам думать, принимать решения и воспринимать окружающий мир. Биологический нейрон – клетка, основные составляющие которой: ядро, отростки, тела и иные компоненты для тесной связи с тысячами нейронов. По этой связи передаются электрохимические импульсы, приводящие нейронную сеть в состояние спокойствия или возбуждения. Например, перед сдачей экзаменов или другим важным событием, порождается импульс и распространяется по всей нейронной сети в головном мозге, проводя сеть в состояние возбуждения. Затем по нервной системе передается это состояние другим органам, что приводит к учащению сердцебиения, частому морганию ресниц и прочим проявлениям волнения.

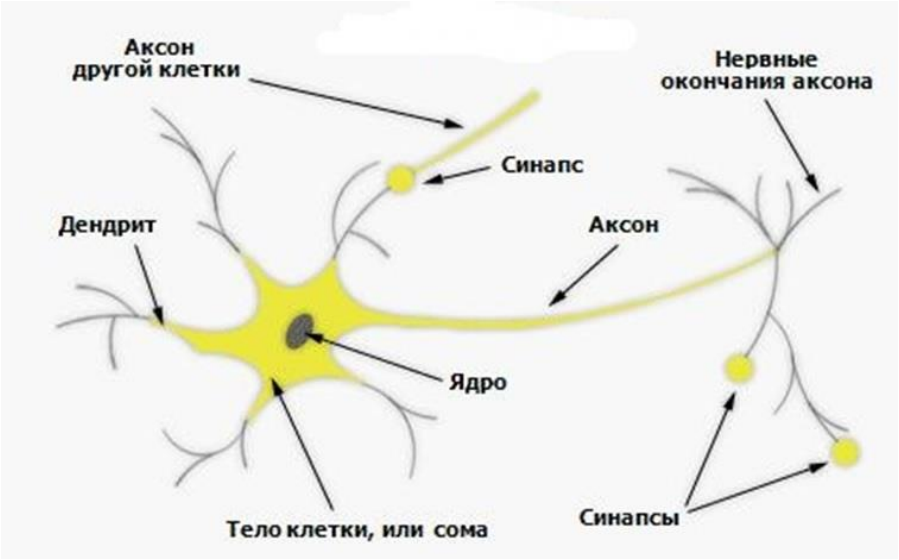


Рисунок 1 – Структура нейрона

По упрощенной теории естественной нейросети рассмотрим составляющие нейрона. Состоит он из тела клетки и ядра. К телу относится множество ответвлений, называемых дендритами. Длинные дендриты называться Аксонами. С их помощью проходит связь между нейронами с помощью синапсов (места контакта двух нейронов, по которым проходит импульс). В этом случае можно уловить закономерность. На дендрит приходит сигнал (значит это вход), по аксону (выход) передается на другую нейронную клетку. В точке синаптической передачи импульса, его частота и амплитуда может изменяться (переменные, составляющие сложного уравнения, влияющие на состояние сигнала). Примерно так работает естественная нейросеть в голове каждого человека. В таком случае, почему бы не создать искусственную нейросеть наподобие естественной, откинув биологическую составляющую. Дендрит будет выступать в роли входа, аксон – выход, тело нейрона в виде сумматора, а кодом, состоящим из 0 и 1 можно регулировать частоту, импульс сигнала, перед подачей на сумматор в разделе «Веса». Это основная составляющая нейрона. В целом, математическую и графическую модель нейрона можно записать так.

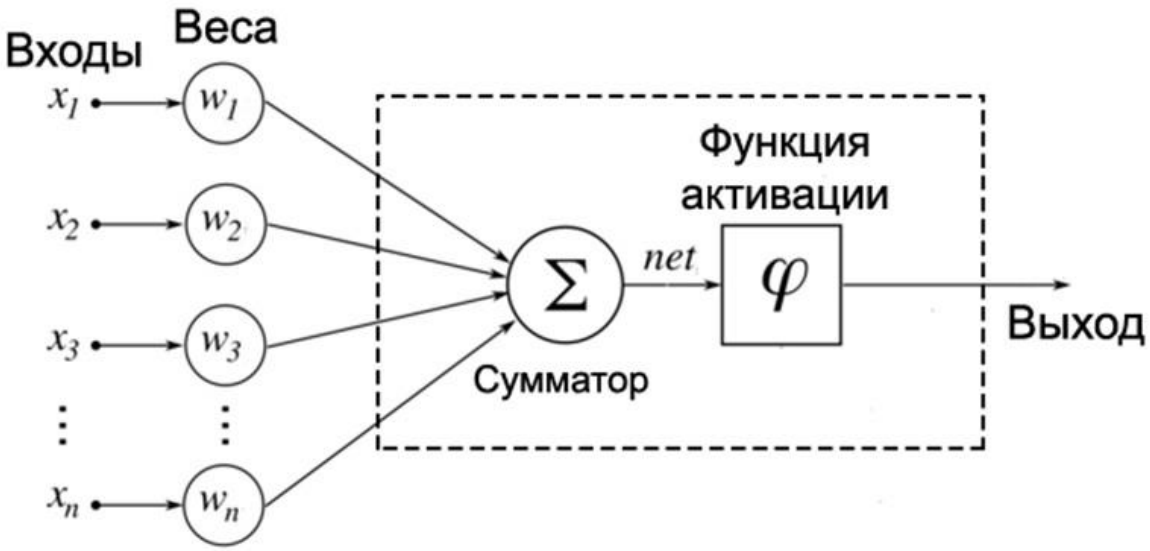


Рисунок 2 - Математическая модель нейрона

Нарисовать и представить алгоритм работы можно с помощью абстрактной схемы. Есть несколько методов представления схем нейронных сетей, но для наглядности проще будет использовать кружки со стрелками.

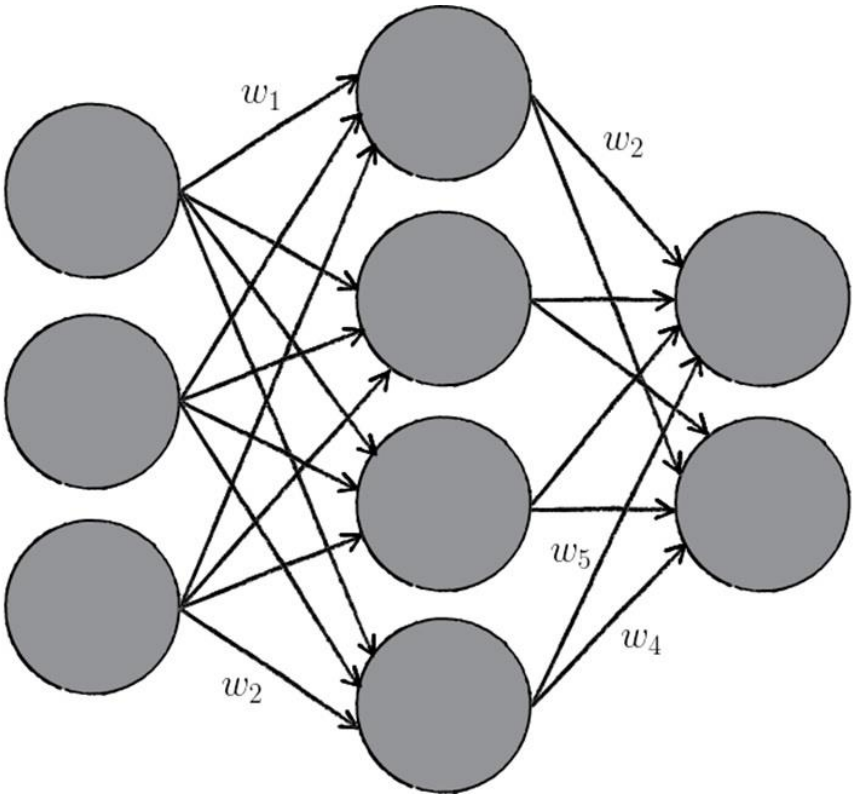


Рисунок 3 - Схема нейронной сети

Изучив основные аспекты, можно дать определение искусственной нейронной сети – это построенная по математическим правилам модель

естественной нейросети, которая воплощена в жизнь с помощью программных и аппаратных составляющих.

Типичная нейронная сеть содержит большое количество искусственных нейронов, называемых единицами, расположенными в ряд слоев.

В типичной искусственной нейронной сети, включает в себя разные слои:

Входной слой - содержит те единицы (искусственные нейроны), которые получают входные данные из внешнего мира, в которых сеть будет учиться, распознавать или обрабатывать иным образом.

Выходной слой - содержит блоки, которые отвечают на информацию о том, как изучается любая задача.

Скрытый слой - эти единицы находятся между входным и выходным слоями. Работа скрытого слоя состоит в том, чтобы преобразовать входные данные во что-то, что модуль вывода может каким-либо образом использовать.

Большинство нейронных сетей полностью связаны, что означает, что каждый скрытый нейрон полностью связан с каждым нейроном в своем предыдущем (входном) слое и на следующем (выходном) слое.

1.3 Обучение нейронной сети

Процесс калибровки значений весов и смещений сети называется обучением нейронной сети, правильному выполнению желаемой функции. Методы обучения или алгоритмы могут быть классифицированы на:

1. **Контролируемое обучение**

При контролируемом обучении данные будут представлены в виде пар (вход, желаемый вывод), а затем алгоритм обучения адаптирует весовые коэффициенты и смещения в зависимости от сигнала ошибки между реальным выходом сети и желаемым выводом, как показано в рисунке ниже.

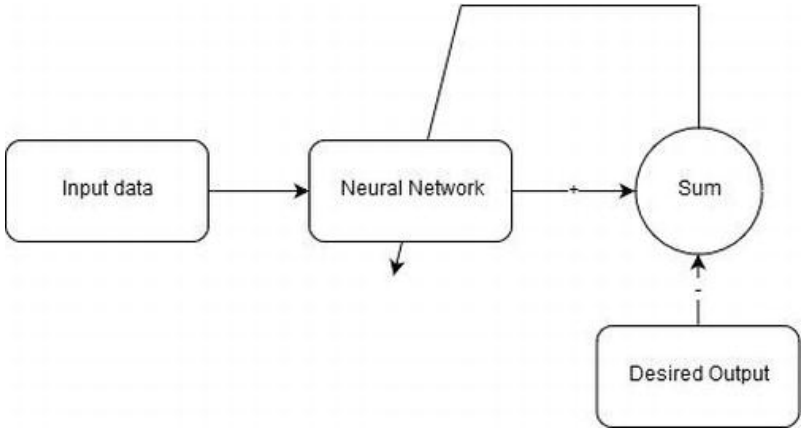


Рисунок 4 - Контролируемое обучение

В качестве показателя эффективности системы мы можем думать в терминах среднеквадратичной ошибки или суммы квадратичных ошибок по обучающей выборке, определенной как функция свободных параметров (то есть синаптических весов) системы.

2. Неконтролируемое обучение

Для осуществления обучения без учителя используется правило конкурентного обучения. Например, мы можем использовать нейронную сеть, которая состоит из двух уровней - входного уровня и конкурентного уровня. Входной слой получает доступные данные. Конкурентный слой состоит из нейронов, которые конкурируют друг с другом (в соответствии с правилом обучения) за «возможность» реагировать на особенности, содержащиеся во входных данных (рисунок 5).[6]

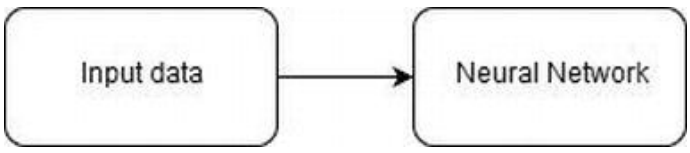


Рисунок 5 - Обучение без присмотра

1.4 Обоснование языка программирования

Машинное обучение (ML) - это тип программирования, который позволяет компьютерам автоматически извлекать уроки из предоставленных

им данных и улучшать их на основе опыта без преднамеренного программирования.

Он основан на алгоритмах, которые анализируют данные, изучают их, и делают прогнозы или интеллектуальные решения автономным способом.

С этой умной характеристикой машинного обучения его часто взаимозаменяют с искусственным интеллектом (ИИ).

Однако, если быть точным, ML - это только часть искусственного интеллекта. Машинное обучение - это просто прикладной искусственный интеллект, основанный на идее, что машинам необходимо предоставить доступ к данным, чтобы они самостоятельно изучали и анализировали их.

Почему же именно Python удобно использовать для машинного обучения?

1. Python легко понять.

Машинное обучение - это просто распознавание шаблонов в ваших данных, чтобы иметь возможность самостоятельно вносить улучшения и принимать разумные решения.

Python является наиболее подходящим языком программирования для этого, потому что его легко понять, и вы можете прочитать его сами.

Его удобочитаемость, несложность и возможность быстрого создания прототипов делают его популярным языком среди разработчиков и программистов по всему миру.

2. Python поставляется с большим количеством библиотек.

Многие из этих встроенных библиотек предназначены для машинного обучения и искусственного интеллекта и могут быть легко применены «из коробки».

Некоторые из библиотек:

- scikit-learn для интеллектуального анализа данных и машинного обучения;

- Tensorflow , библиотека нейронных сетей высокого уровня;
- pylearn2, который также идеально подходит для интеллектуального анализа данных и машинного обучения, но более гибкий, чем scikit-learn.

3. Python позволяет легкую и мощную реализацию.

Что делает Python одним из лучших решений для машинного обучения, так это его простая и мощная реализация.

С другими языками программирования начинающим программистам или студентам необходимо сначала ознакомиться с языком, прежде чем использовать его для ОД или ИИ.

Это не так с Python. Даже если у вас есть только базовые знания языка Python, вы уже можете использовать его для машинного обучения из-за огромного количества библиотек, ресурсов и инструментов, доступных для вас.

Кроме того, вы будете тратить меньше времени на написание кода и отладку ошибок на Python, чем на Java или C ++.

Программисты ML и AI, скорее всего, скорее потратят свое время на создание своих алгоритмов и эвристик, чем на отладку своего кода на предмет синтаксических ошибок.

4. Дружественный синтаксис и удобочитаемость на уровне человека

Python - это объектно-ориентированный язык программирования, использующий современные сценарии и дружественный синтаксис.

Разработанная с почти читабельностью на уровне человека, скриптовая природа Python позволяет кодировщикам и программистам проверять свои гипотезы и очень быстро запускать свои алгоритмы.

Это причина, по которой языки структурного программирования, такие как Java, Perl и C ++, требующие жесткого кодирования, обычно не одобряются для машинного обучения.

Подводя итог, будь вы опытный программист или начинающий программист, вы можете многое сделать с помощью Python, который идеально подходит для выполнения сложного набора задач машинного обучения.

Все вышеупомянутые причины делают Python предпочтительным и востребованным языковым навыком в мире ИТ.

5. Сообщество

Наконец, Python предоставляет широкую поддержку. Поскольку многие люди, как программисты, так и обычные пользователи, рассматривают Python как стандарт, его сообщество поддержки огромно, что еще больше увеличивает популярность Python.

1.5 Выбор программной среды разработки

Spyder (ранее Pydee) - свободная среда разработки для Python. Она кроссплатформенна и доступна для Windows, Linux и MacOS. Название Spyder расшифровывается как Scientific Python Development EnviRonment, то есть научная среда разработки для python. Программа создавалась для научных расчетов, и в этой сфере она действительно удобна.

Согласно аннотации разработчиков, Spyder является:

Мощной интерактивной средой разработки для языка программирования Python с продвинутыми возможностями редактирования, интерактивного тестирования, отладки и интроспекции. Средой численных расчетов благодаря поддержке IPython (улучшенный интерактивный интерпретатор Python) и популярных библиотек Python, таких как NumPy (линейная алгебра), SciPy (обработка данных, численные расчеты и др.) и matplotlib (интерактивная 2D/3D визуализация).

Частью модуля spyderlib для Python, предоставляющего мощные виджеты на PyQt4, такие как редактор кода, консоль Python (встраиваемая в приложения), редактор списков/кортежей и массивов NumPy.

Рисунок 6 - Значок среды Spyder

Рассмотрим некоторые возможности Spyder.

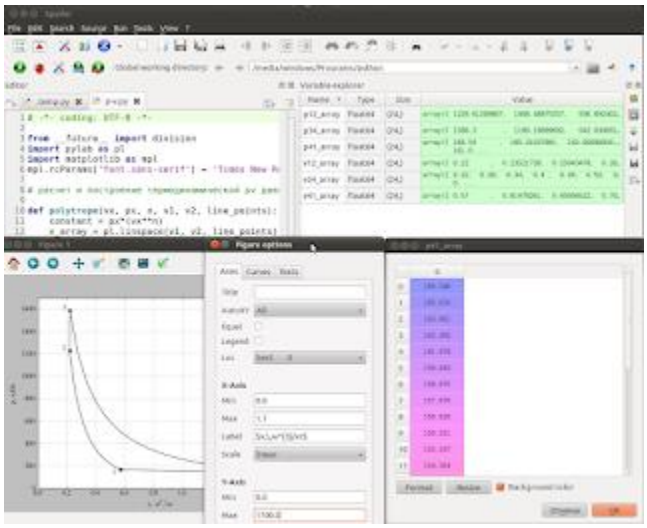


Рисунок 7 - Работа в Spyder

Просмотр значений переменных

Пожалуй одно из главных фишек - возможность удобной работы с переменными. В процессе выполнения программы они выводятся на панели в виде списка с возможностью просмотра их значений. Имеется даже возможность строить графики по данным массива. Очень удобно при отладке программы и поиске логических ошибок.

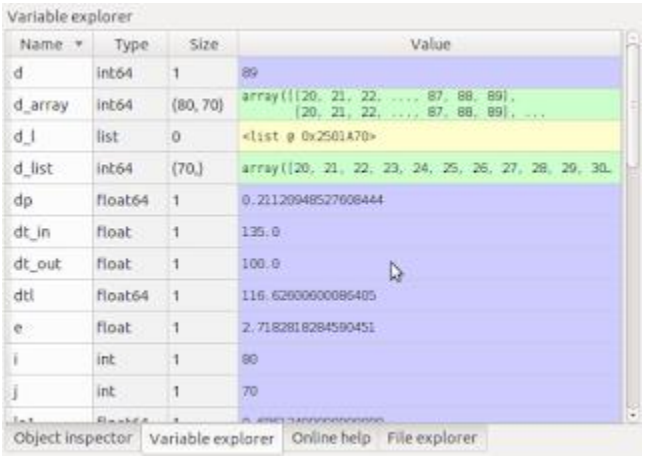


Рисунок 8 - Список переменных

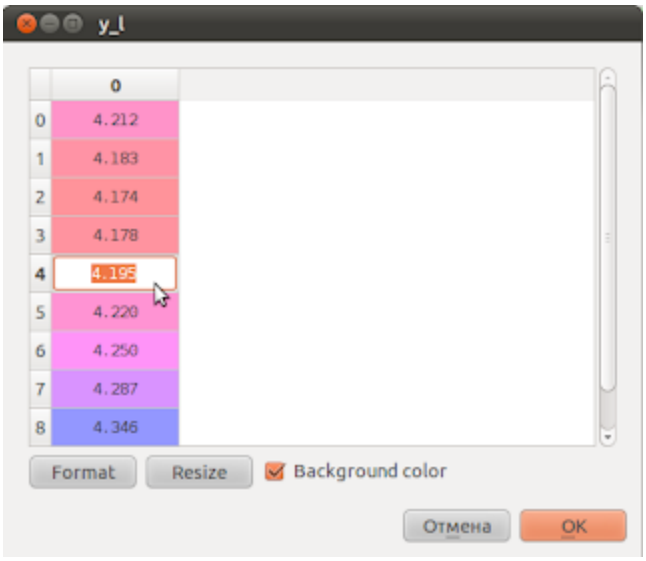


Рисунок 9 - Просмотр массива numpy и редактирование значения элемента

Spyder предоставляет большие возможности по работе с консолями Python или IPython. Можно создавать требуемое количество консолей и взаимодействовать с ними как с отдельными процессами. Ваши программы могут запускаться в новой или в существующей консоли.

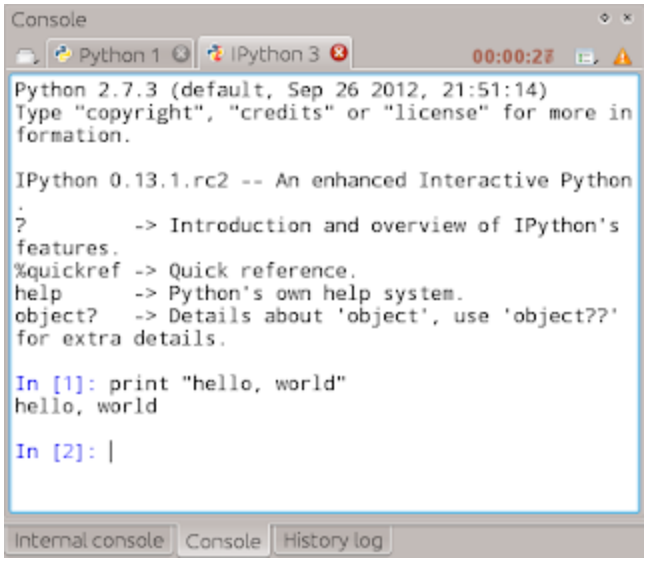


Рисунок 10 - Интеграция matplotlib и formlayout

При запуске программ, использующих для визуализации библиотеку matplotlib появляется возможность редактировать параметры графика, например, легенду, параметры осей, подписи, стили линий с помощью формы, реализуемой библиотекой formlayout. Это удобно, так как не требуется перезапускать программу чтобы просто изменить параметры оси или что-то подобное.

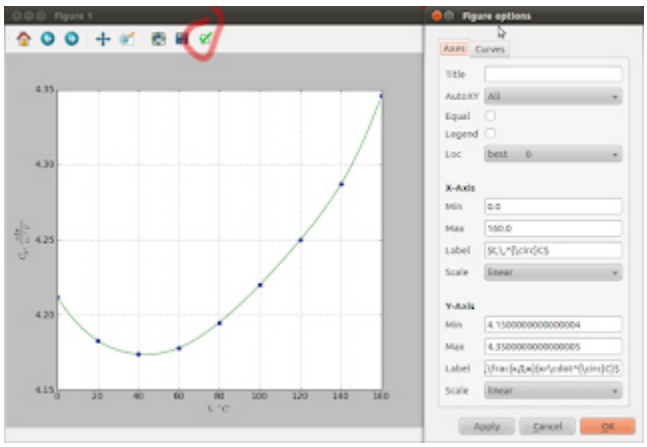


Рисунок 11 - Настройка внешнего вида графика matplotlib с помощью формы formlayout

Мгновенный анализ на ошибки

С помощью pyflakes в реальном времени код программы проверяется на ошибки, и в случае их нахождения пользователь получает подсказки.

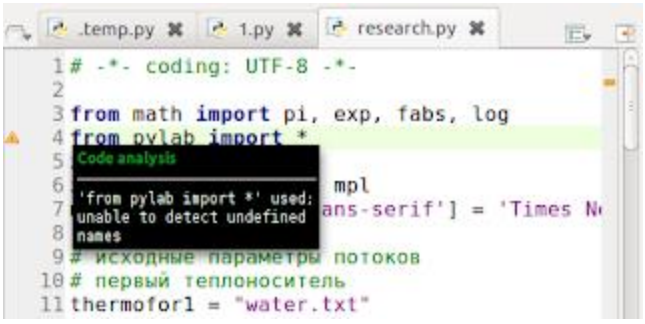


Рисунок 12 - Интроспекция кода

Имеются возможности: автодополнения, перехода к выбранному объекту или функции, просмотра кода используемых модулей.

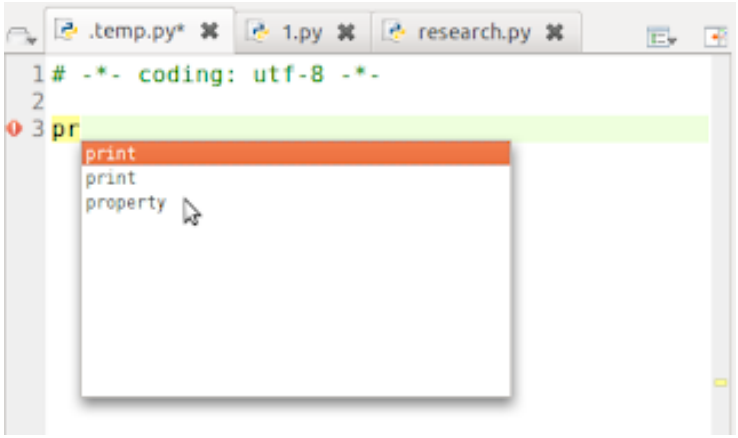


Рисунок 13 – Автодополнение

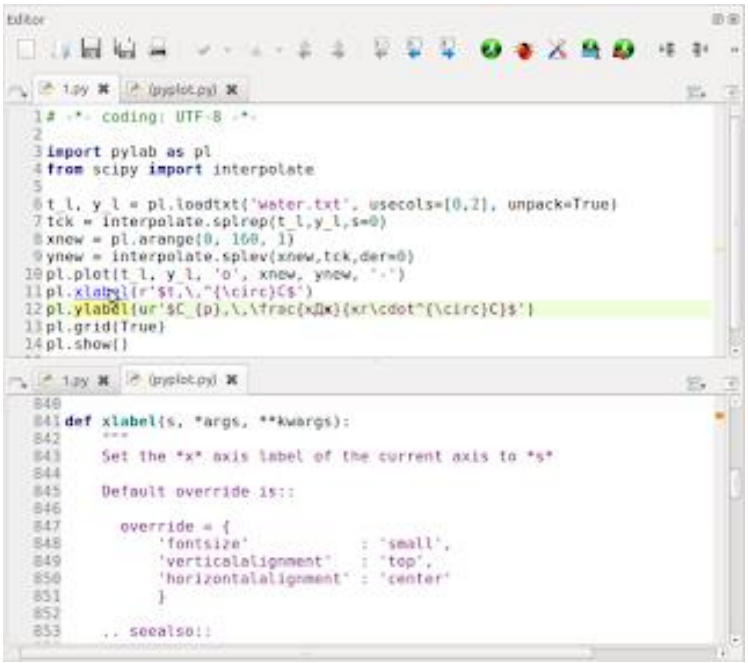


Рисунок 14 - Просмотр кода функции

Имеется возможность просмотра документации или исходных кодов любых объектов Python (классов, функций, модулей). Также доступна онлайн

документация модулей Python (сгенерированная в формат html). Для документации в Python используется библиотека Sphinx. Очень удобная опция, позволяющая лишний раз не лезть в интернет за информацией о библиотечных функциях.

Рисунок 15 - Справочная информация по библиотетекам

1.6 Преимущества и недостатки нейронных сетей

1. Преимущества нейронных сетей:

- Нейронная сеть может выполнять задачи, которые не может выполнять линейная программа.
- Когда элемент нейронной сети выходит из строя, он может продолжаться без каких-либо проблем по своей параллельной природе.

- Нейронная сеть учится и не нуждается в перепрограммировании.
- Это может быть реализовано в любом приложении.
- Это может быть выполнено без каких-либо проблем.

2. Ограничения нейронных сетей:

- Нейронная сеть нуждается в обучении для работы.
- Архитектура нейронной сети отличается от архитектуры микропроцессоров, поэтому ее необходимо эмулировать.
- Требует большого времени обработки для больших нейронных сетей.[11]

1.7 Нейросетевой анализ медицинских данных

Первое упоминание об ИНС относится в 1943 году, когда в одной из публикаций У. Маккалох и У. Питтс на электрический схеме представили модель нейронной сети. Далее в 1950-х годах исследовательская лаборатория IBM начинает активную работу в данном направлении, однако предпочтение отдается классическим методам статической обработки и применение ИНС отходит на дальний план. Кроме того, массивность оборудования в то время также могло препятствовать возникновению симпатии к данному подходу анализа данных. Лишь 1985 году основательно окреп интерес к нейронным сетям, Американский институт физики стал проводить ежегодные встречи «Нейронные сети для вычислений».

На сегодняшний день существует множество моделей используемых архитектур ИНС, которые различаются своей вычислительной сложностью, степенью сходства с живыми нейронами головного мозга, а также обладающие исключительной и единственной в своем создании уникальностью. В связи с этим ИНС не подлежат стандартам каких-либо классификаций в сравнении с традиционными статистическими методами. Существующие нейронные сети способны работать как с числовыми данными, лежащими в определенном ограниченном диапазоне, так и

нечисловыми параметрами, например, графическими изображениями различных конфигураций. Однако нестандартный масштаб количественных характеристик, наличие пропущенных значений, вариабельность номинальных переменных, преобразование качественных параметров в числовую функцию или объявление их незначащими, создают дополнительные проблемы в работе ИНС и искажают выходной результат.

Существует множество примеров использования искусственных нейронных сетей в области здравоохранения. Однако подавляющее большинство работ выполнено зарубежными исследователями и касается возможностей использования ИНС только в диагностике различных клинических ситуаций. В 1990 году, в стенах Калифорнийского университета была создана первая искусственная нейронная сеть, её создателем является Вильям Бакстом. Созданная нейронная сеть использовалась для распознавания инфаркта миокарда у пациентов, поступающих в больницу с жалобами наструю боль в груди. В результате обучения на 20 различных клинических параметрах, созданная Бакстом ИНС продемонстрировала точность при обнаружении ИМ равную 92 процентам и дала только 4% случаев сигналов ложной тревоги.

В 2011 году Джоти Сони и др. на основе искусственной нейронной сети создали веб интерфейс позволяющий прогнозировать риск развития сердечной недостаточности. В качестве базы данных были взяты 303 истории болезни пациентов больницы Кливленда с изучением всех имеющихся лабораторных данных, общее количество которых составило 13 закодированных числовых параметра. Обученный при помощи 3 машинных алгоритмов с контролируемым управлением в Java-платформе, взвешенный ассоциативный классификатор (WAC) предсказал риск развития сердечно-сосудистых заболеваний с точностью 81,51. Натан Хут и Доминик Аронский в 2005 году использовали технологии на основе искусственных нейронных сетей для прогнозирования выживаемости трансплантатов печени сроком около дней.

Чувствительность разработанной ими модели, обученной по клиническим и лабораторным данным пациентов полученной из сети базы Sharing за 2000–2002 г., составила 95%.

В 2014 году Дэниэл Каллан и др. с помощью определенных процедур проверки на контрастных изображениях, обучили SLR зеркальный декодер как метод классификации с целью автоматизированного обнаружения пациентов с хронической болью в спине. Согласно отчету, в процессе распознавания многомерных моделей функциональной деятельности нижней теменной областях коры головного мозга общая точность работы ИНС при классификации в группы составила 92,3%. Также установлено, что паттерны активности ГМ могут быть использованы в качестве диагностических маркеров хронических болей. В 2014 году Тоуни Л.К. создал искусственную нейронную сеть с одним скрытым нейронным слоем. Данная ИНС позволяла идентифицировать злокачественные воспаления лимфатических узлов, точность определения которых составила 99,2%.

В кардиологии ИНС применяются для решения большого спектра задач. Они могут применятся для анализа шумов сердца, аритмий, определения внутрижелудочковой эктопической активности и диагностировании ряда сердечно-сосудистых заболеваний. Обучение искусственных нейронных сетей проведено для распознавания изменений сегмента ST-T, определения числа вовлеченных в атеросклеротический процесс сосудов.

Таким образом, использование нейросетевых технологий имеет множество определенных перспектив. На данным момент прогнозирование заболеваний становится все более трудоемким, это связано с тем, что с каждым годом используется все большее число диагностически важных маркеров, чья обработка занимает большое количество времени. Применение информационных и нейросетевых технологий в области медицины требует понимания механизмов обработки данных компьютерной программой, а также знанием механизмов развития заболеваний. Проводимые эксперименты

с применением нейросетевых технологий говорят о большой перспективности данного направления и о большом количестве ее еще неизученных возможностях.[3]

1.8 Применение нейронных сетей в области анализа патологий в кровеносных сосудах

После определения возможности использования искусственных нейронных сетей в медицине, рассмотрим уже существующие исследования в этой сфере.

В настоящее время существует ряд отечественных и зарубежных авторов, исследующих применение нейронных сетей для выявления диагноза пациентов в разных областях медицины. Одним из примеров применения искусственных нейронных сетей в медицине является работа аспиранта Саратовского государственного университета им. Н.Г. Чернышевского Аравина О.И. «Применение искусственных нейронных сетей для анализа патологий в кровеносных сосудах».

Основной задачей исследования в данной области является анализ результатов полученных после ультразвуковой допплерографии артерий нижних конечностей и нахождения положения атеросклеротических бляшек в продольном срезе сонной артерии. В ходе исследования были смоделированы несколько искусственных нейронных сетей, способных решать поставленные задачи с процентом ошибки не более 9.

Недостаточность необходимого кровообращения обусловливает тяжесть ишемического синдрома при окклюзионных заболеваниях брюшной аорты и ее ветвей, а также локализации стеноза. С помощью ультразвуковой допплерографии, возможно не только подтвердить наличие стеноза, но также появляется возможность его локализации и определить его протяженность.

Основную роль при постановке диагноза стеноза в нижних конечностях при помощи допплерографии играют роль как прямые, так и косвенные признаки. Основным способом определения прямых признаков является анализ кровотока бедренной, подколенной и задних большеберцовых вен. Косвенным критерием непроходимости глубоких вен, в свою очередь, является увеличение скорости кровотока в поверхностных венах. Точный диагноз можно установить только в случае наличия обоих критериев. Основным способом нахождения и определения данных признаков является сравнение кровотока в венах в симметричных точках противоположной нижней конечности. Основой для анализа допплеровских сигналов кровотока полученных при обследовании артерий нижних конечностей являются параметры, которые при опыте дают достоверную возможность локации и дифференцирования нормальных и патологических сигналов сосуда (Табл. 1.1).

Артерия	Справа		Слева	
	линейная скорость кровотока	индекс пульсации	линейная скорость кровотока	индекс пульсации
Брюшная аорта	54,8	2,37	–	–
Наружная подвздошная артерия	86,5	4,85	90,2	4,49
Бедренная артерия	52,4	4,32	58,5	5,06
Подколенная артерия	51,2	5,45	47,5	4,33
Задняя большеберцовая артерия	35,3	3,0	12	2,16
Передняя большеберцовая артерия	29,4	4,55	10	1,08
Артерии тыла стопы	12,6	–	6	–
Магистральная почечная артерия	19,8	–	–	–

Рисунок 16 - Данные доплеровского исследования нижних конечностей

Основная задача работы заключалась в том, чтобы, основываясь на выявленных ранее параметрах, диагностировать наличии стеноза в исследуемых конечностях.

Для решения поставленной задачи была выбрана модель двухслойной нейронной сети прямого распространения. Входной слой искусственной нейронной сети состоял из 36 нейронов, данное количество обусловлено количеством исходных параметров для постановки диагноза. Количество нейронов скрытого слоя обычно подбирается экспериментально. В результате экспериментов был выбран скрытый слой, состоящий из 16 нейронов, при этом выходной слой состоял из 2 нейронов, которые соответствуют диагнозам в исследуемых сосудах.

Считается, что если в результате работы искусственной нейронной сети с данными, что не участвовали в процессе её обучения, процент ошибки находится в заранее обговоренных пределах, то сеть обладает хорошей обучающей способностью. Способность нейронной сети к обобщению определяется многими факторами, один из которых является размер обучающей выборки. Обучающая выборка обычно формируется из примеров, каждый из которых представляет собой результат эксперимента предметной области с определенным ответом. В качестве обучающей выборки были составлены 1000 исходных параметров допплерографии кровотока с соответствующим им диагнозом. Следующим шагом была обучена и протестирована нейронная сеть заданной архитектуры. Тестирование проводилось на выборке, составленной из параметров допплерографии кровотока, не участвующих при обучении нейронной сети. Погрешность определения диагноза на тестирующих данных составила приблизительно 8%.

В работе Аверина О.И. был рассмотрен способ применения искусственной нейронной сети для определения патологии в сосудах. Данная работа является отличным примером использования модели искусственной нейронной сети. Результаты обучения и проверки работоспособности спроектированной сети показывает её успешное применение для решения поставленных задач и способность находить

сложные закономерности и взаимосвязи между различными объектами, относящимися к одному классу данных. После того как сеть обучена, она становится надежным и недорогим инструментом анализа патологий в кровеносных сосудах.[3]

1.9 Глубокое обучение

Чтобы ответить на этот вопрос, нужно понять, как оно взаимодействует с машинным обучением, нейросетями и искусственным интеллектом. Для этого используем метод визуализации с помощью концентрических кругов:

Внешний круг – это искусственный интеллект в целом (например, компьютеры). Чуть дальше – машинное обучение, а совсем в центре – глубинное обучение и искусственные нейросети. Грубо говоря, глубинное обучение – просто более удобное название для искусственных нейросетей. «Глубинное» в этом словосочетании обозначает степень сложности (глубины) нейросети, которая зачастую может быть весьма поверхностной. Создатели первой нейросети вдохновлялись структурой коры головного мозга. Базовый уровень сети, перцептрон, является по сути математическим аналогом биологического нейрона. И, как и в головном мозге, в нейросети могут появляться пересечённые друг с другом перцептроны. Первый слой нейросети называется входным. Каждый узел этого слоя получает на вход какую-либо информацию и передает ее на последующие узлы в других слоях. Чаще всего между узлами одного слоя нет связей, а последний узел цепочки выводит результат работы нейросети. Узлы посередине называются скрытыми, поскольку не имеют соединений с внешним миром, как узлы вывода и ввода. Они вызываются только в случае активации предыдущих слоев.

Глубинное обучение – это по сути техника обучения нейросети, которая использует множество слоев для решения сложных проблем (например, распознавания речи) с помощью шаблонов. В восьмидесятых годах

большинство нейросетей были однослойными в силу высокой стоимости и ограниченности возможностей данных. Если рассматривать машинное обучение как ответвление или вариант работы искусственного интеллекта, то глубинное обучение – это специализированный тип такого ответвления. Машинное обучение использует компьютерный интеллект, который не дает ответа сразу. Вместо этого код будет запускаться на тестовых данных и, исходя из правильности их результатов, корректировать свой ход. Для успешности этого процесса обычно используются разнообразные техники, специальное программное обеспечение и информатика, описывающая статические методы и линейную алгебру.[3]

1.9.1. Методы глубинного обучения

Методы глубинного обучения делятся на два основных типа:

- Обучение с учителем
- Обучение без учителя

Первый способ использует специально отобранные данные, чтобы добиться желаемого результата. Он требует довольно много человеческого вмешательства, ведь данные приходится выбирать вручную. Однако он удобен для классификации и регрессии. Представьте, что вы владелец компании и хотите определить влияние премий на продолжительность контрактов с вашими подчиненными. При наличии заранее собранных данных, метод обучения с учителем был бы незаменим и очень эффективен. Второй же способ не подразумевает заранее заготовленных ответов и алгоритмов работы. Он направлен на выявление в данных скрытых шаблонов.

Обычно его используют для кластеризации и ассоциативных задач, например для группировки клиентов по поведению. «С этим также выбирают» на Amazon – вариант ассоциативный задачи. В то время как метод обучения с учителем довольно часто вполне удобен, его более сложный вариант все же

лучше. Глубинное обучение зарекомендовало себя как нейросеть, не нуждающаяся в надзоре человека.[3]

1.9.2. Важность глубинного обучения

Компьютеры уже давно используют технологии распознавания определенных черт на изображении. Однако результаты были далеки от успеха.

Компьютерное зрение оказало на глубинное обучение невероятное влияние. Именно эти две техники в данный момент решают все задачи на распознавание. В частности, в распознавании лиц на фотографиях с помощью глубинного обучения преуспел Facebook. Это не простое улучшение технологии, а поворотный момент, изменяющий все более ранние представления: «Человек может с вероятностью в 97.53% определить, один ли человек представлен на двух разных фотографиях. Программа, разработанная командой Facebook, может делать это с вероятностью в 97.25% вне зависимости от освещения или того, смотрит ли человек прямо в камеру или повернут к ней боком».

Распознавание речи тоже претерпело значительные изменения. Команда Baidu – одного из лидирующих поисковиков Китая – разработала систему распознавания речи, сумевшую опередить человека в скорости и точности написания текста на мобильных устройствах. На английском и мандаринском. Что особенно занимательно – написание общей нейросети для двух абсолютно разных языков не потребовало особенного труда: «Так исторически сложилось, что люди видели Китайский и Английский, как два совершенно разных языка, поэтому и подход к каждому из них требовался различный», — говорит начальник исследовательского центра Baidu, Andrew Ng. «Алгоритмы обучения сейчас настолько обобщены, что вы можете просто обучаться». Google использует глубинное обучение для управления энергией в дата-центрах компании. Они смогли сократить затраты ресурсов для

охлаждения на 40%. Это около 15% повышения эффективности энергопотребления и миллионы долларов экономии.

1.9.3 Микросервисы глубинного изучения

Вот краткий обзор сервисов, связанных с глубинным обучением. Illustration Tagger. Дополненный Illustration2Vec, этот сервис позволяет отмечать изображения с рейтингом «защищенный», «сомнительный», «опасный», «копирайт» или «общий» для того, чтобы заранее понять содержание картинки.

Классификатор возраста использует технологии анализа фотографии для определения возраста человека. Places 365 Classifier использует заранее натренированную нейросеть в сочетании с базой данных за 2016 год для определения местоположение человека по фотографии (например, деревня, аптека, номер гостиной, горы и так далее). Не стоит забывать и о InceptionNet – прямом наследнике InceptionNet от Google. Эта нейросеть на основе анализа фотографии машины выдает пять лучших моделей, соответствующих этому автомобилю.[3]

2 РЕАЛИЗАЦИЯ ПОЛЬЗОВАТЕЛЬСКОГО ПРИЛОЖЕНИЯ

В данной главе описана реализация программы на основе машинного обучения по прогнозированию женской болезни недержания мочи. клиентской и серверной частей приложения. Освещены основные особенности реализации, связанные с программированием на выбранных языках. Целью данного раздела является объяснить принцип работы и внутреннее устройство системы в целом.

2.1 Постановка задачи

Необходимо разработать программу на основе машинного обучения на языке программирования Python по прогнозированию женской болезни недержания мочи. Первоначально нужно собрать данные существующих болезней для проверки эффективности программы. Представить их в нужном формате для распознавания программой. Затем обучить нейронную сеть на части подготовленных данных и протестировать работоспособность сети на оставшейся части данных. И в итоге разработать простой интерфейс для прогнозирования наличия болезни одного конкретного человека.

Многомодульность программы. Необходимо разбить программу на модули. Это повысит наглядность и легкость восприятия кода программы, а также облегчит поиск и исправление возможных ошибок.

Режим работы видеосистемы-графический.

Пользовательский интерфейс. Необходимо определиться с выбором доступного и интуитивно понятного меню для возможности эффективной работы с программой.

Операционная система - MS Windows(сервер).

Среда программирования – Spyder, Qt Designer.

Язык программирования – Python(сервер).

2.2 Структура данных приложений

Для реализации программы был выбран язык программирования Python, который поддерживается средой разработки Spider.

Взаимодействие с пользователем построено на использовании кнопок и текстовых полей для ввода. Интерфейс был разработан в среде Qt Designer, а функционал описан на языке Python.

На рисунке 6 показаны UML-диаграммы вариантов использования программы.

Рисунок 6 - UML-диаграмма вариантов использования программы

Программа представляет собой окно с кнопками для выбора файла и предсказания результата и текстовыми полям для ввода факторов риска наличия болезни.

2.3 Выбор архитектуры приложения и способа организации пользовательского интерфейса

Консольная программа для тестирования работы модели была написана на языке Python. Интерфейс разработан в среде Qt Designer, а затем конвертирован в описание на языке программирования Python и дополнен описанием функционала. На рисунке 8 изображена UML-диаграмма состояний программы.

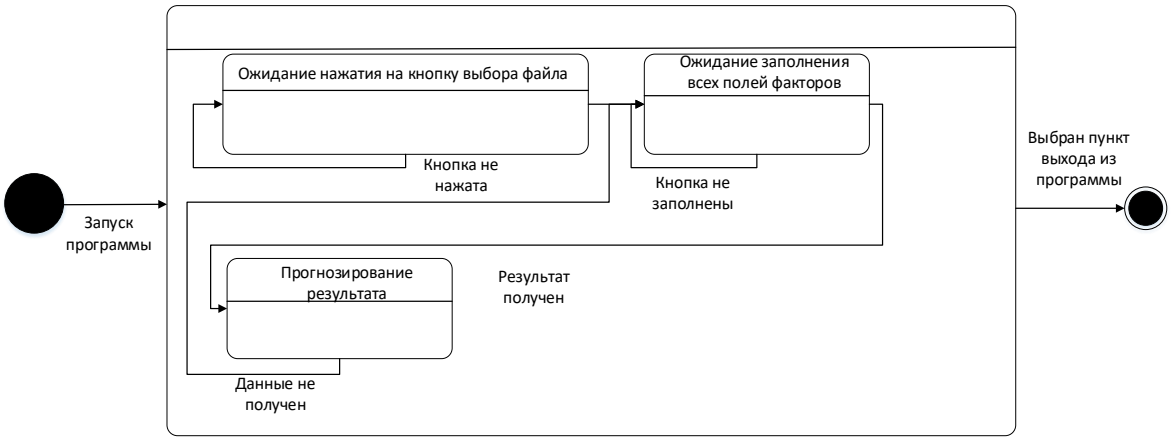


Рисунок 8 - UML-диаграмма состояний программы

2.4 Описание разработки программы

Для написания программы-клиента был использован язык программирования Python.

В качестве среды программирования сервера была выбрана программа Spyder, а интерфейс разработан в среде QT Designer.

Интерфейс программы клиента представлен в виде экранной формы с элементами управления, так как этот способ представления функций наиболее удобен для восприятия и может быть легко освоен даже не подготовленным пользователем.

Для удобства работы медиков и обычных пользователей (пациентов) с программой была проведена подготовка файлов с целью упрощения запуска программы. В первоначальном варианте для работы с программой пользователю необходимо установить python на компьютер, все дополнительные библиотеки, используемые в программе, и даже после этого возможны возникающие ошибки в работе с программой.

При работе с python существует удобная библиотека pyinstaller, которая помогает адаптировать файлы .py в .exe с дополнительными зависимости. Файл .exe можно будет запустить без сложных установок для пользователей.

Для создания нужного исполняемого файла необходимо в консоле с расположением в директории с файлами .py

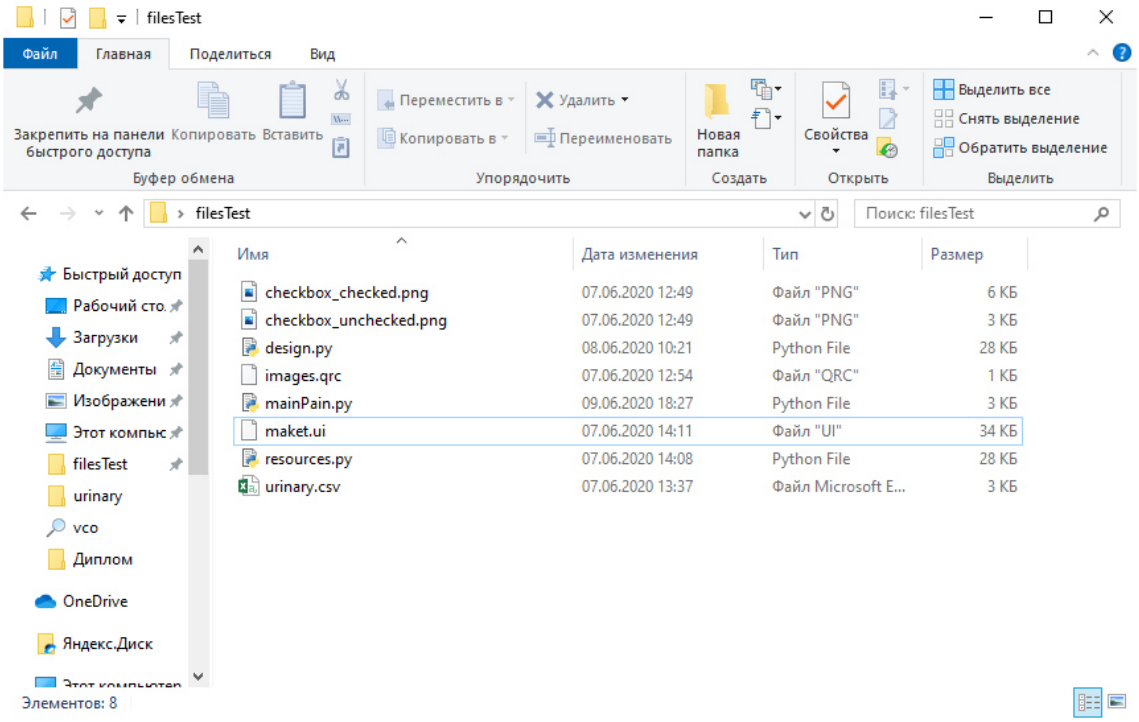


Рисунок 17 - Директория с файлами

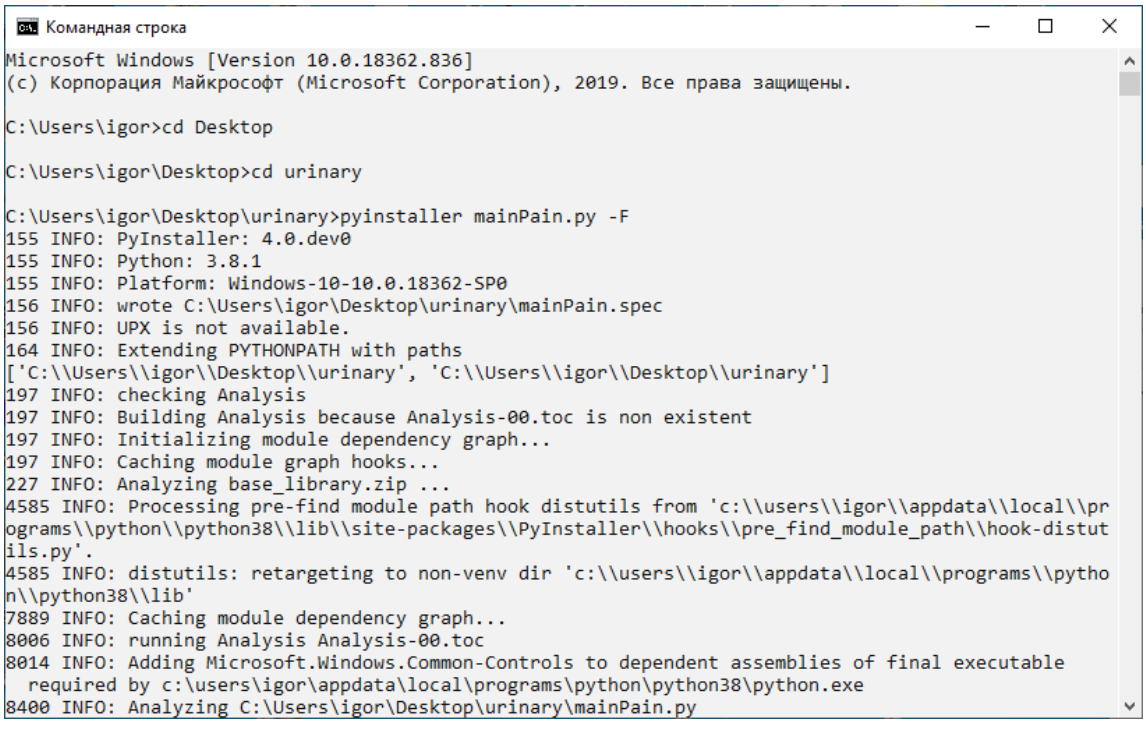


Рисунок 18 - Запуск pyinstaller

После выполнения команды в текущей папке с файлами сформируются дополнительные файлы и папки.

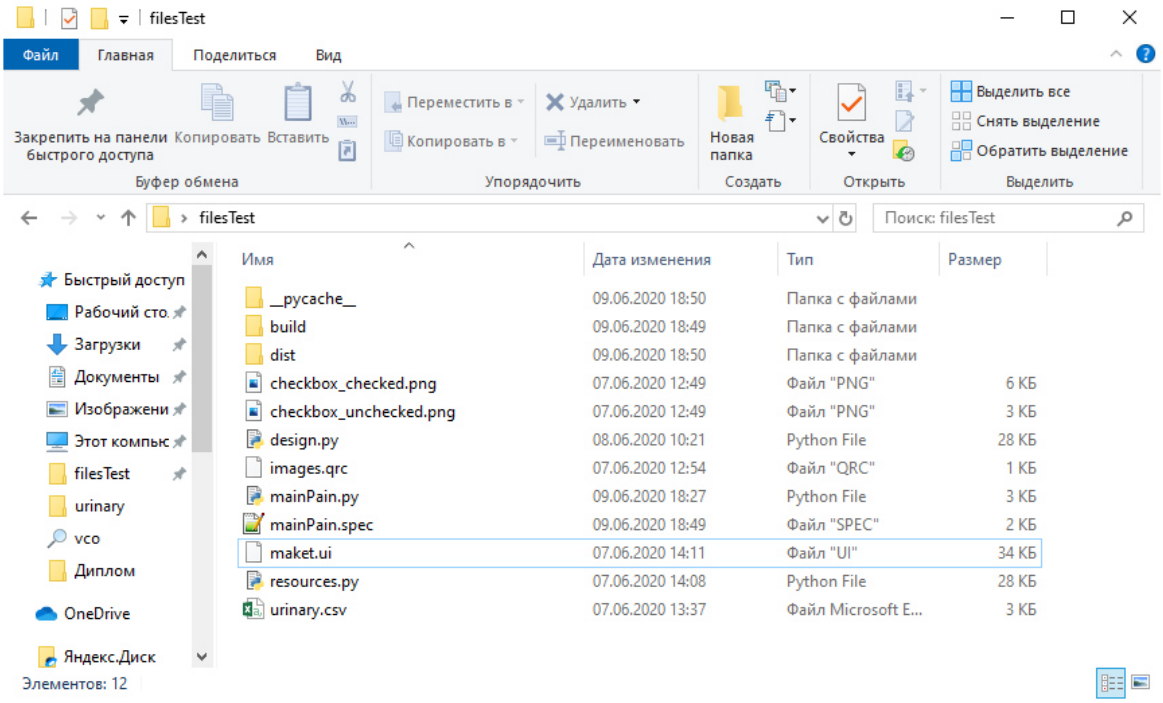


Рисунок 19 - Папка после выполнения pyinstaller

Нужный исполняемый файл будет расположен

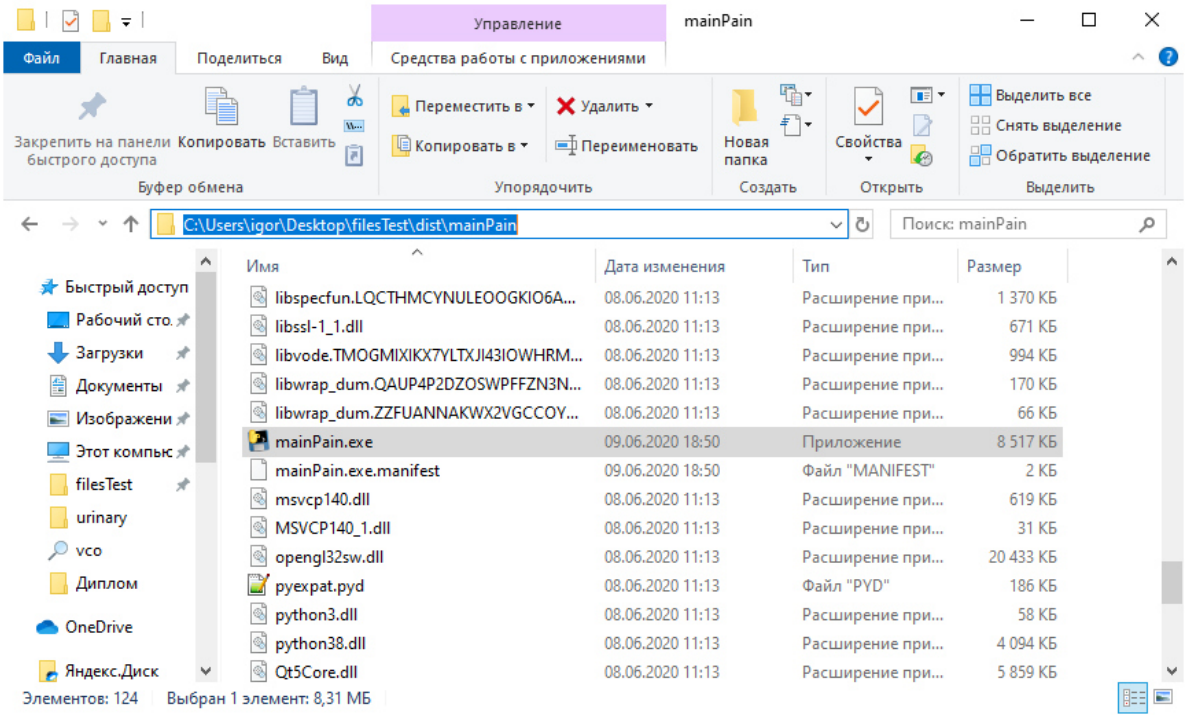


Рисунок 20 - Расположение исполняемого файла

Но при первоначальном запуске возникают ошибки с отстутсвием файла vcomp140.dll и отсутствием некоторых библиотек для модуля sklearn. Для исправления, файл .dll был добавлен по пути выше, а нужные библиотеки добалены в файл mainPain.spec в переменную hiddenimport.

После чего файл будет выглядить следующим образом.

```
# -*- mode: python ; coding: utf-8 -*-

block_cipher = None


a = Analysis(['mainPain.py'],
             pathex=['C:\\Users\\igor\\Desktop\\urinary'],
             binaries=[],
             datas=[],
             hiddenimports=['pandas._libs.tslibs.timedeltas',
                'sklearn.neighbors.typedefs','sklearn.utils._cython_blas',
                'sklearn.neighbors._quad_tree','sklearn.tree._utils'],
             hookspath=[],
             runtime_hooks=[],
             excludes=[],
             win_no_prefer_redirects=False,
             win_private_assemblies=False,
             cipher=block_cipher,
             noarchive=False)
pyz = PYZ(a.pure, a.zipped_data,
```

Рисунок 21 - Файл mainPain.spec

После выполнения всех перечисленных действий программа работает успешна и готова к использованию неопытными пользователями.

При запуске программы на экране появляется экранная форма с кнопкой выбора файла для обучения нейронной сети, для этого необходима выбрать подготовленный файл в формате .csv.

Первоначально были собраны данные пациентов в формате .doc. Их вид представлен на рисунке ниже.

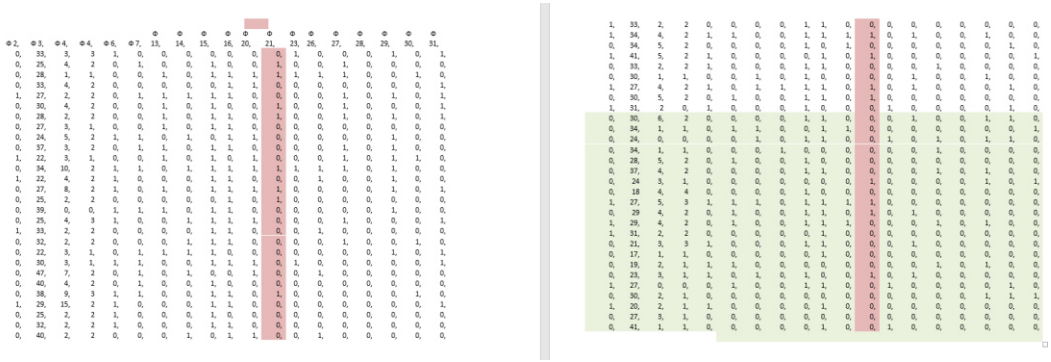


Рисунок 22 - Факторы болезни пациентов

Для распознования программой их необходимо представить в формате .csv в виде, представленном ниже. Здесь в первой строке представлены наименования факторов, для удобства понимания, а далее соответсвующие

данные по каждому опрошенному пациенту. В последнем столбце расположен критерий наличия или отсутсвия проблемы недержания мочи и пациенток.

	A	B	C	D	E	F	G	H
1	heredity,bmi,pregnancy,childbirth,difficult,operation,constipation,copd,infections,gra							
2	0,33,3,3,1,0,0,0,0,0,0,0,1,0,0,1,1,0,1,1							
3	0,25,4,2,0,1,0,0,1,0,0,1,0,0,1,0,0,0,0,1							
4	0,28,1,1,0,0,1,0,1,1,1,1,1,1,1,0,0,1,0,1							
5	0,33,4,2,0,0,0,0,0,1,1,0,0,0,0,0,0,0,1,1							
6	1,27,2,2,0,1,1,1,1,1,0,0,0,0,1,0,1,0,1,1							
7	0,30,4,2,0,0,1,0,1,0,0,1,0,0,1,0,0,0,1,1							
8	0,28,2,2,0,0,1,0,1,1,0,1,0,0,1,0,1,0,1,1							
9	0,27,3,1,0,0,1,0,1,1,0,0,0,0,0,0,0,0,0,1							
10	0,24,5,2,1,1,0,1,0,1,1,0,0,0,0,0,0,1,0,0,1							
11	0,37,3,2,0,1,1,0,1,1,0,0,0,0,0,1,0,1,0,0,1							
12	1,22,3,1,0,0,1,0,1,0,1,0,0,0,1,0,1,1,0,1							
13	0,34,10,2,1,1,0,1,1,1,1,1,1,1,1,1,0,1,0,0,1							
14	1,22,4,2,1,0,0,0,1,1,0,0,0,1,0,0,1,0,0,1							
15	0,27,8,2,1,0,1,0,1,1,1,1,0,0,0,0,0,1,0,1,1							
16	0,25,2,2,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0,1							
17	0,39,0,0,1,1,1,0,1,1,0,0,0,0,0,0,0,1,0,0,1							

Рисунок 23 - Данные в формате .csv

Данные в программе будут использоваться в формате DataFrame с помощью библиотеки pandas. После считывания данных с подготовленного файла необходимо их нормализовать к единому виду от 0 до 1.

Затем разделить используемые данные на две выборки для обучения и тестирования работы программы прогнозирования. Где 80 процентов будет использовано для обучения и 20 процентов для тестирования программы. В итоге выполнено обучение, сделано прогнозирование и подсчитан процент ошибки. С использованием метода логистической регрессии процент правильно спрогнозированной болезни равен 75.

```
In [1]: runfile('C:/Users/igor/Desktop/urinary.py', wdir='C:/Users/igor/Desktop')
0.75
```

Рисунок 24 – Доля правильных предсказаний

Было проведена попытка улучшения результата предсказания с помощью применения разных методов обучения: логистической регрессии, SVC, рандомного леса, Гаусса, градиентного спуска.

В итоге на диаграмме ниже мы мы можем увидеть, что методом градиентного спуска удалось добиться более 80 процентов успешного предсказания.

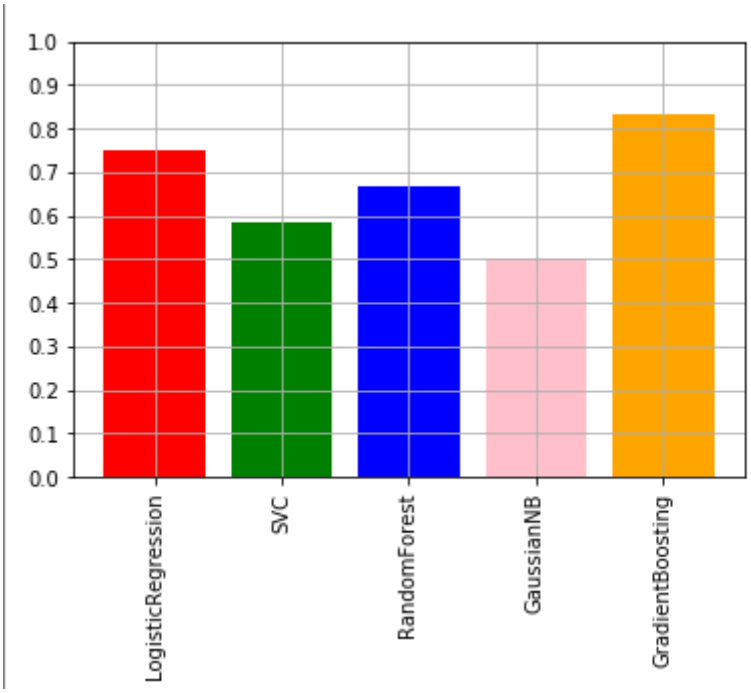


Рисунок 25 - Сравнение результатов разных методов

К примеру, после применения метода логистической регрессии мы можем отследить расстановку весов для каждого фактора. На основе этого можно сделать вывод, что при анализе данных наиболее важными факторами для модели были такие, как: copd(ХОБЛ), constipation(Хронические запоры), nephroptosis(Грыжи), heredity(Наследственность).

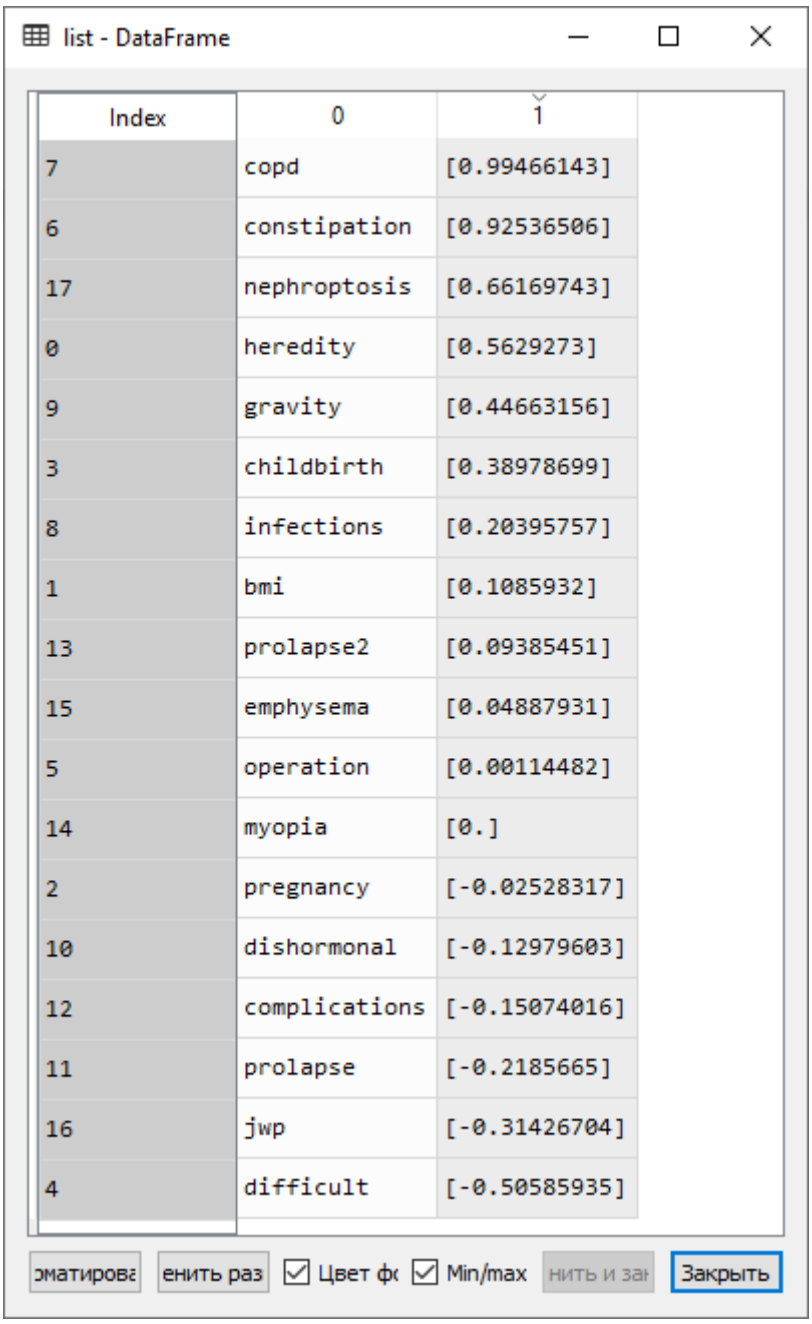


Рисунок 26 - Расстановка весов

В программе Qt Designer был разработан интерфейс для пользовательской программы, где возможно по введеным критериям предсказать наличие или отсутсвие болезни отдельного пациента.

2.5 Отладка и тестирование

В качестве среды разработки программы был выбран Spyder, содержащий в себе различные средства для разработки и отладки модулей и

программ. А для разработки интерфейса был выбран удобный Qt Designer. Для отладки программы использовались различные инструменты: точка останова в браузере, пошаговое выполнение кода программы, анализ содержимого глобальных и локальных переменных, результаты работы некоторых функций, трассировка, вывод переменных в консоль.

Тестирование проводилось в рабочем порядке, в процессе разработки и после завершения написания программы, все выявленные ошибки были устранены.

3 ОПИСАНИЕ ПРОГРАММЫ

3.1 Описание пользовательского приложения

Для запуска пользовательского приложения необходимо с помощью консоли запустить файл программы с расширением .exe, затем в отрывшемся окне сверху нажать на кнопку «Выбрать файл», после чего в обозревателе выбрать нужный файл для обучения нейронной сети, путь выбранного файла отобразится справа.

Затем необходимо ознакомиться с рашифровкой факторов и произвести заполенение текстовых полей в соотсветсвии с имеющимися симптомами.

Последним шагом будет нажатие кнопки «Спрогнозировать», что позволит получить результат наличия или отсустствия болезни у тестируемого пользователя.

Описание всех событий клиентского приложения показано в таблице 1.

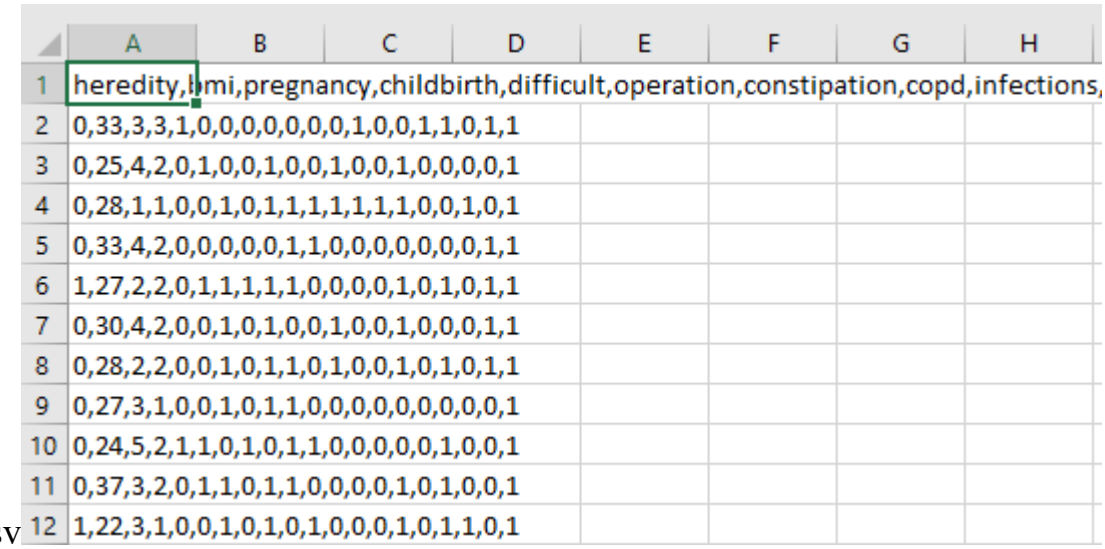
Таблица 1 – Описание состояний программы-пользователя

Событие	Действие Пользователя	Действие программы
Выбор файла	Нажата вкладка	Переход на соответсвующую вкладку
Заполнение тектовых полей	Нажата кнопка “Выбрать файл”	Выбор файла и хэширование его содержимого
Прогнозирование результата	Нажатие на кнопку “Спрогнозировать”	Генерация ключей

3.2 Описание тестовых данных

Было опрошено 58 пациентов, внешний вид данных представлен на рисунке ниже. У 37 выявлено событие (недержание мочи), у 21 пациентки недержание мочи отсутствует. Информация по болезни представлена в последнем столбце «target», где 1 – болезнь присутствует, 0 – болезнь отсутсвует.

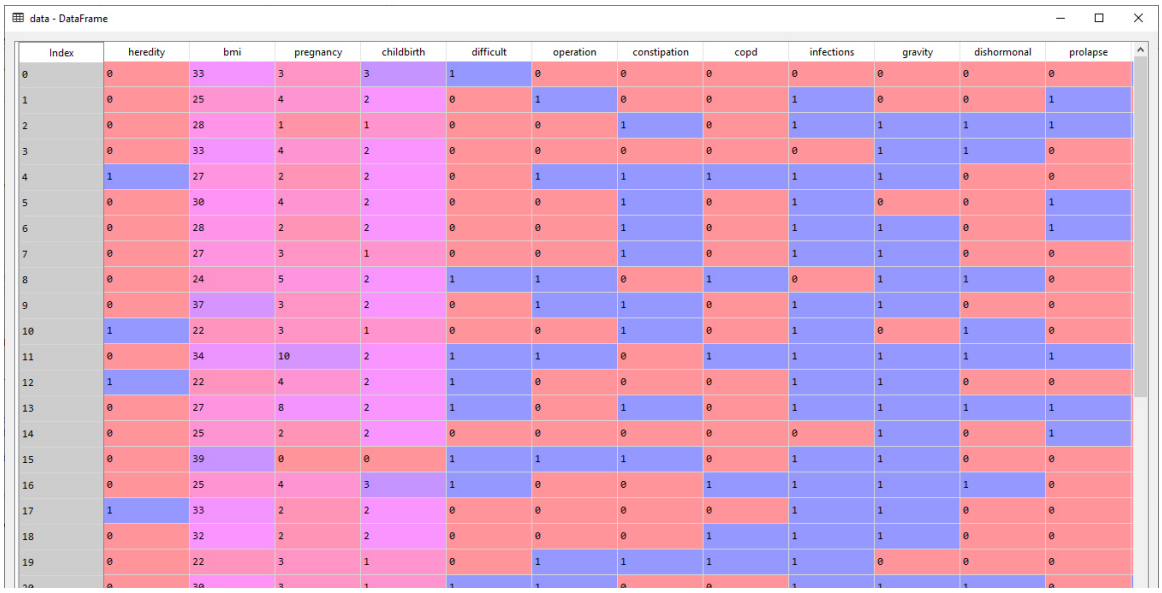
Первоначально данные будут в сложночитабельном виде в формате .csv



	A	B	C	D	E	F	G	H
1	heredity,bmi,pregnancy,childbirth,difficult,operation,constipation,copd,infections,							
2	0,33,3,3,1,0,0,0,0,0,0,0,1,0,0,1,1,0,1,1							
3	0,25,4,2,0,1,0,0,1,0,0,1,0,0,1,0,0,0,0,1							
4	0,28,1,1,0,0,1,0,1,1,1,1,1,1,1,0,0,1,0,1							
5	0,33,4,2,0,0,0,0,0,1,1,0,0,0,0,0,0,0,1,1							
6	1,27,2,2,0,1,1,1,1,1,0,0,0,0,1,0,1,0,1,1							
7	0,30,4,2,0,0,1,0,1,0,0,1,0,0,1,0,0,0,1,1							
8	0,28,2,2,0,0,1,0,1,1,0,1,0,0,1,0,1,0,1,1							
9	0,27,3,1,0,0,1,0,1,1,0,0,0,0,0,0,0,0,0,1							
10	0,24,5,2,1,1,0,1,0,1,1,0,0,0,0,0,1,0,0,1							
11	0,37,3,2,0,1,1,0,1,1,0,0,0,0,1,0,1,0,0,1							
12	1,22,3,1,0,0,1,0,1,0,1,0,0,0,1,0,1,1,0,1							

Рисунок 27 - Данные в формате csv

Затем данные данные будут считаны программой и уже блягодаря отображению Spyder в более удобном виде представлены ниже.



Index	heredity	bmi	pregnancy	childbirth	difficult	operation	constipation	copd	infections	gravity	dishormonal	prolapse
0	0	33	3	3	1	0	0	0	0	0	0	0
1	0	25	4	2	0	1	0	0	1	0	0	1
2	0	28	1	1	0	0	1	0	1	1	1	1
3	0	33	4	2	0	0	0	0	0	1	1	0
4	1	27	2	2	0	1	1	1	1	1	0	0
5	0	30	4	2	0	0	1	0	1	0	0	1
6	0	28	2	2	0	0	1	0	1	1	0	1
7	0	27	3	1	0	0	1	0	1	1	0	0
8	0	24	5	2	1	1	0	1	0	1	1	0
9	0	37	3	2	0	1	1	0	1	1	0	0
10	1	22	3	1	0	0	1	0	1	0	1	0
11	0	34	10	2	1	1	0	1	1	1	1	1
12	1	22	4	2	1	0	0	0	1	1	0	0
13	0	27	8	2	1	0	1	0	1	1	1	1
14	0	25	2	2	0	0	0	0	0	1	0	1
15	0	39	0	0	1	1	1	0	1	1	0	0
16	0	25	4	3	1	0	0	1	1	1	1	0
17	1	33	2	2	0	0	0	0	1	1	0	0
18	0	32	2	2	0	0	0	1	1	1	0	0
19	0	22	3	1	0	1	1	1	1	0	0	0

Рисунок 28 - Данные в программе

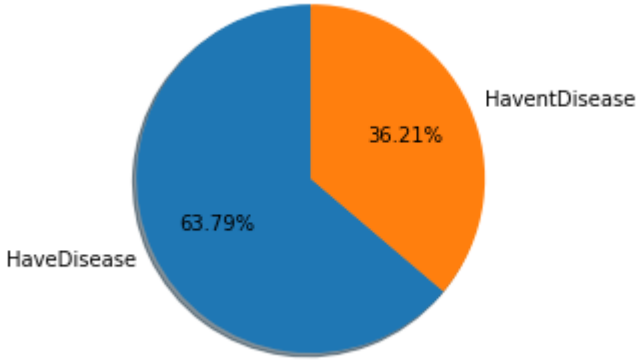


Рисунок 29 - Соотношение больных и здоровых

3.3 Руководство пользователя для работы с программой

Архив с файлами необходимо разахивировать в удобное место на компьютере.

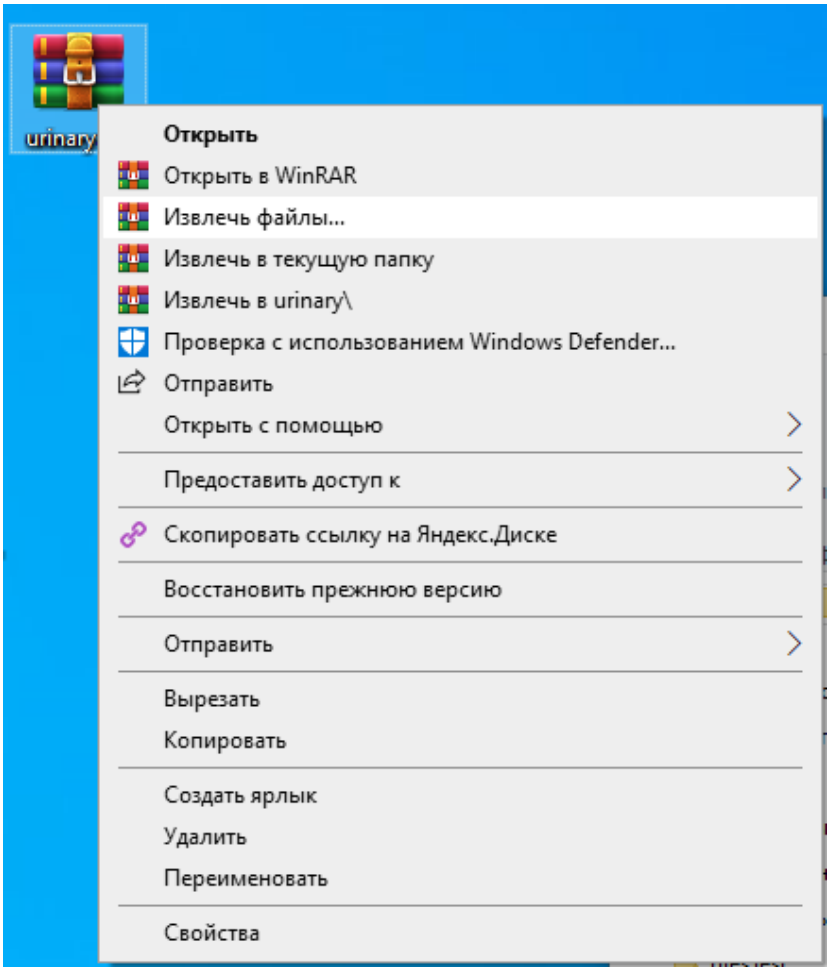


Рисунок 30 - Архив с файлами программы

После этого пользователь получит папку со всеми файлами.

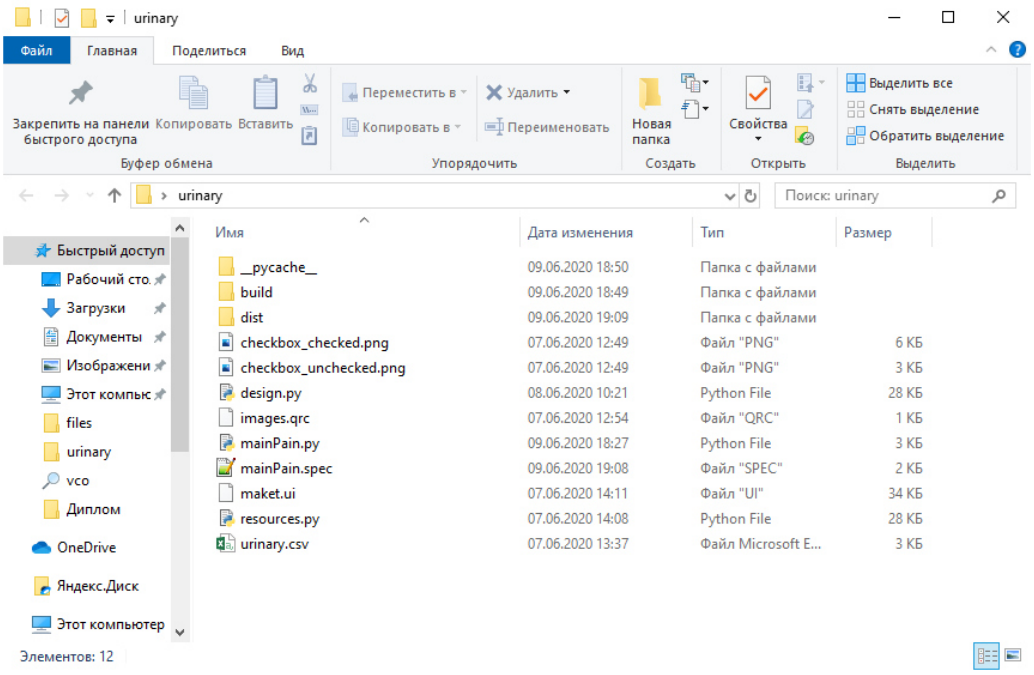


Рисунок 31 - Папка после разархивирования

Файл для обучения модели виден на рисунке выше и называется urinary.csv. Из текущей папки необходимо зайти в папку dist, затем в папку mainPain, пролистать вниз и найти файл mainPain.exe. Исполняемый файл для запуска программы будет по пути …\urinary\dist\mainPain.

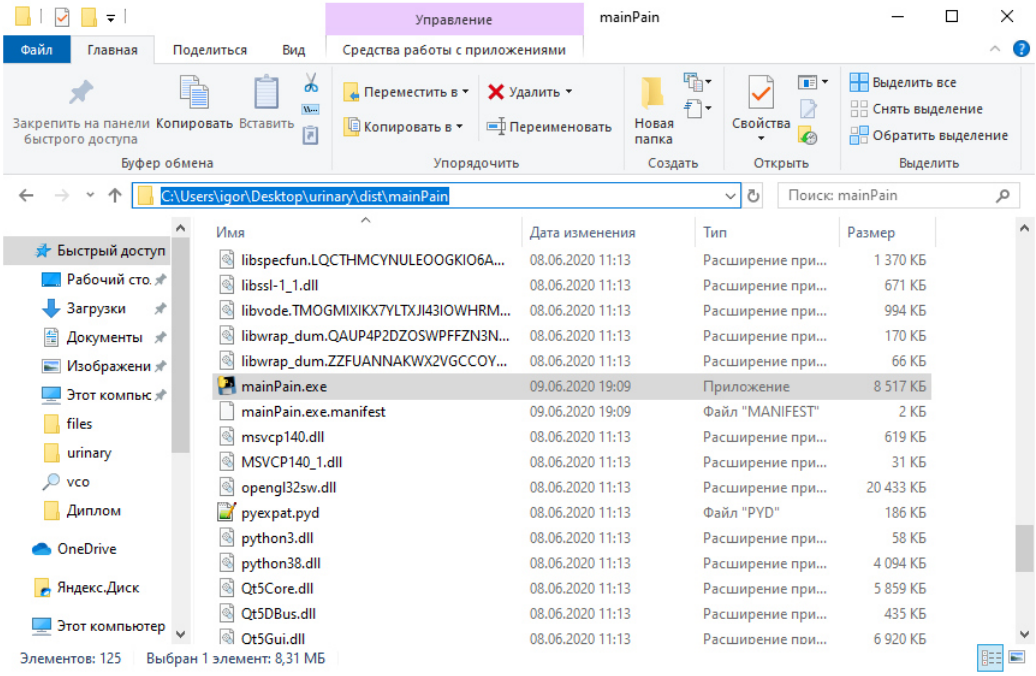


Рисунок 32 - Исполняемый файл mainPain.exe

После двойного щелчка по исполняемого файлу откроется стартовое окно программы. По кнопке «Выбрать файл» нужно выбрать файл для

обучения модели. Далее заполнить все необходимые поля факторов. Для заполнения факторов 2-4 нужно уточнить их расшифровку ниже и ввести числовые данные тестируемого пациента. В остальные поля нужно поставить галочки, если имеется наличие признака по описанию снизу. И в конечном итоге, после выполнения всех действий нужно нажать кнопку «Спрогнозировать» и получить данные о возможном наличии болезни.

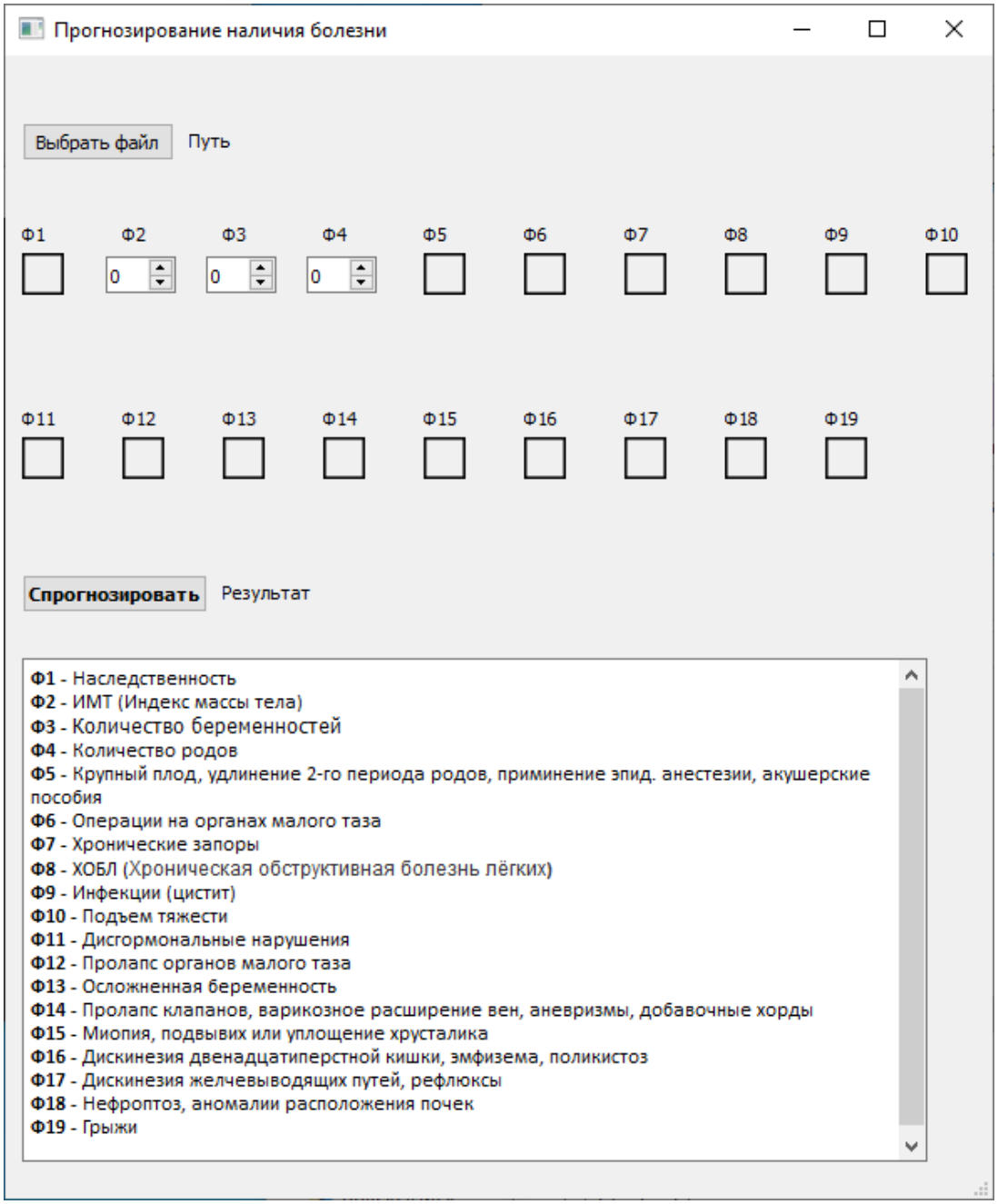


Рисунок 33 - Стартовое окно программы

После нажатия на кнопку «Выбрать файл» и пользователь перейдёт к выбору файла для обучения в обозревателе. Нужный файл urinary.csv будет лежать в корневой папке проекта.

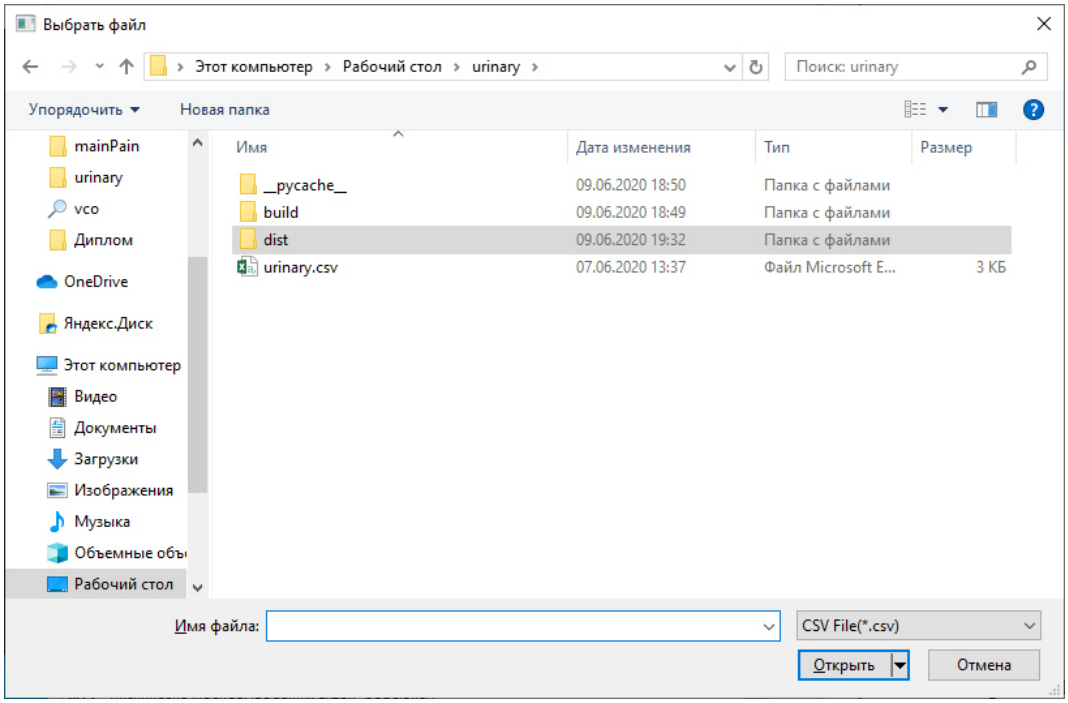


Рисунок 34 - Выбор файла

Справа от кнопки отобразится путь выбранного файла.

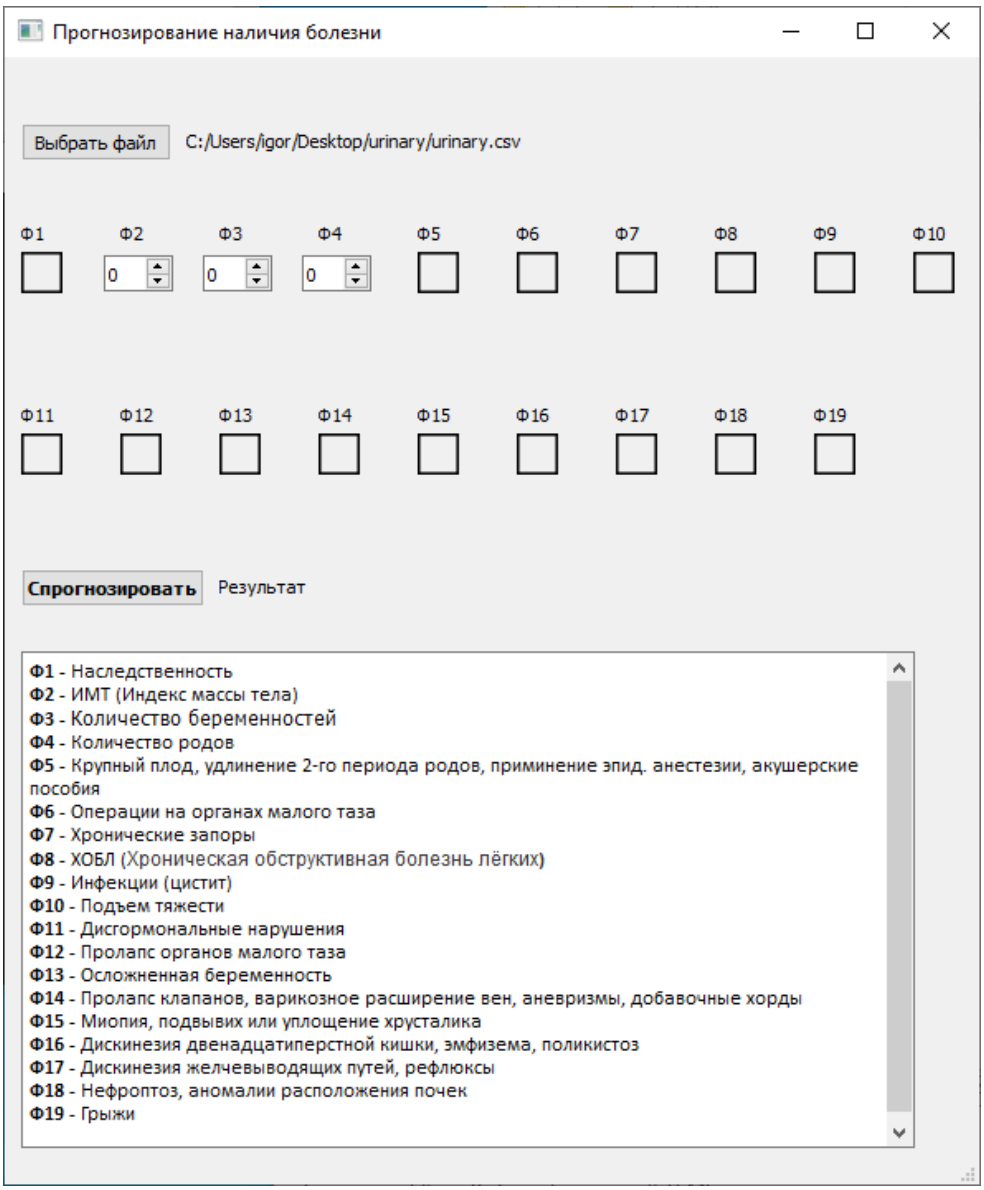


Рисунок 35 - Вывод пути файла

Следующим шагом будет заполнение текстовых полей с факторами пациента.

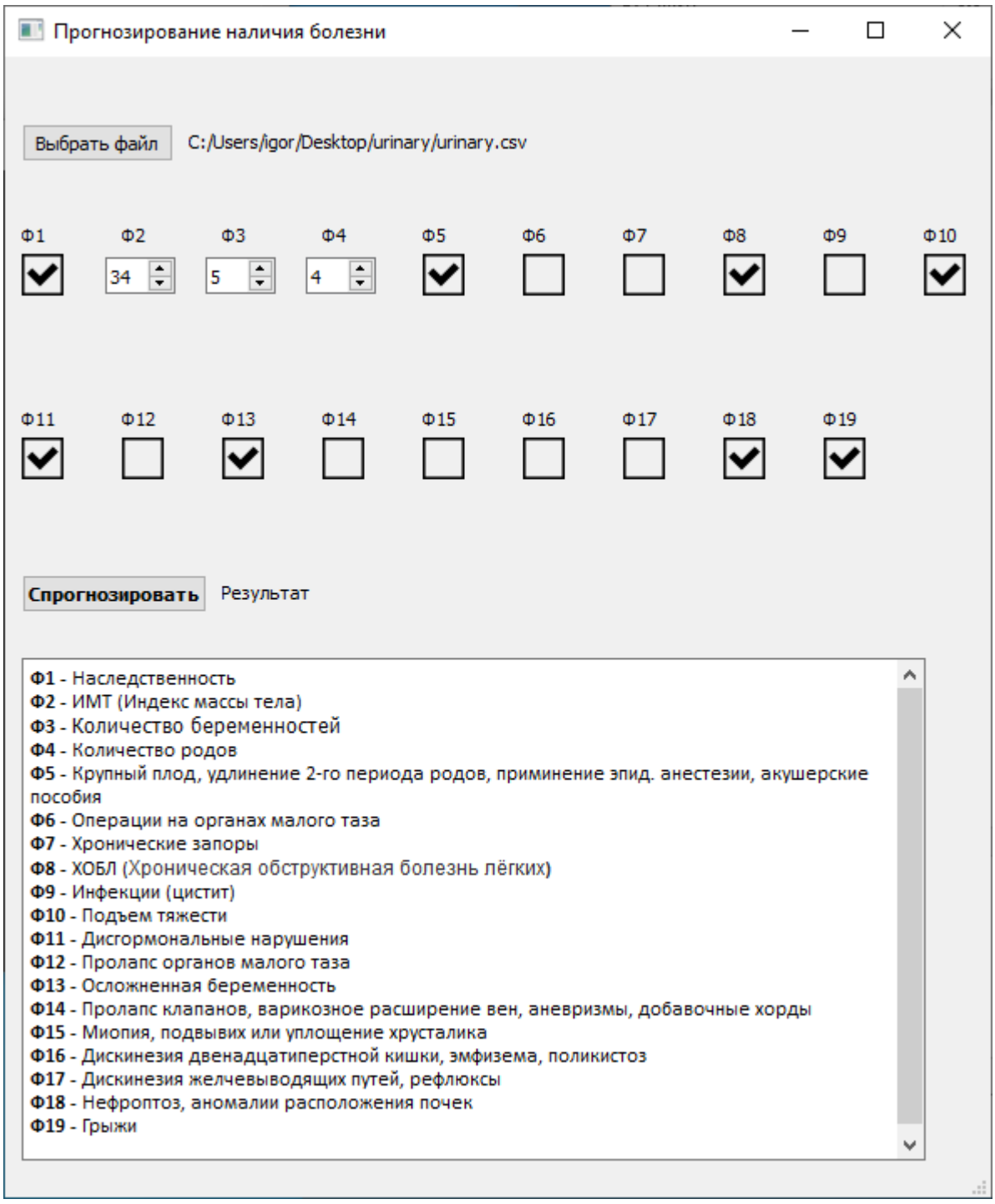


Рисунок 36 - Заполнение текстовых полей

И в итоге после нажатия кнопки «Спрогнозировать» пользователь получит предсказание наличия или отсутствия болезни.

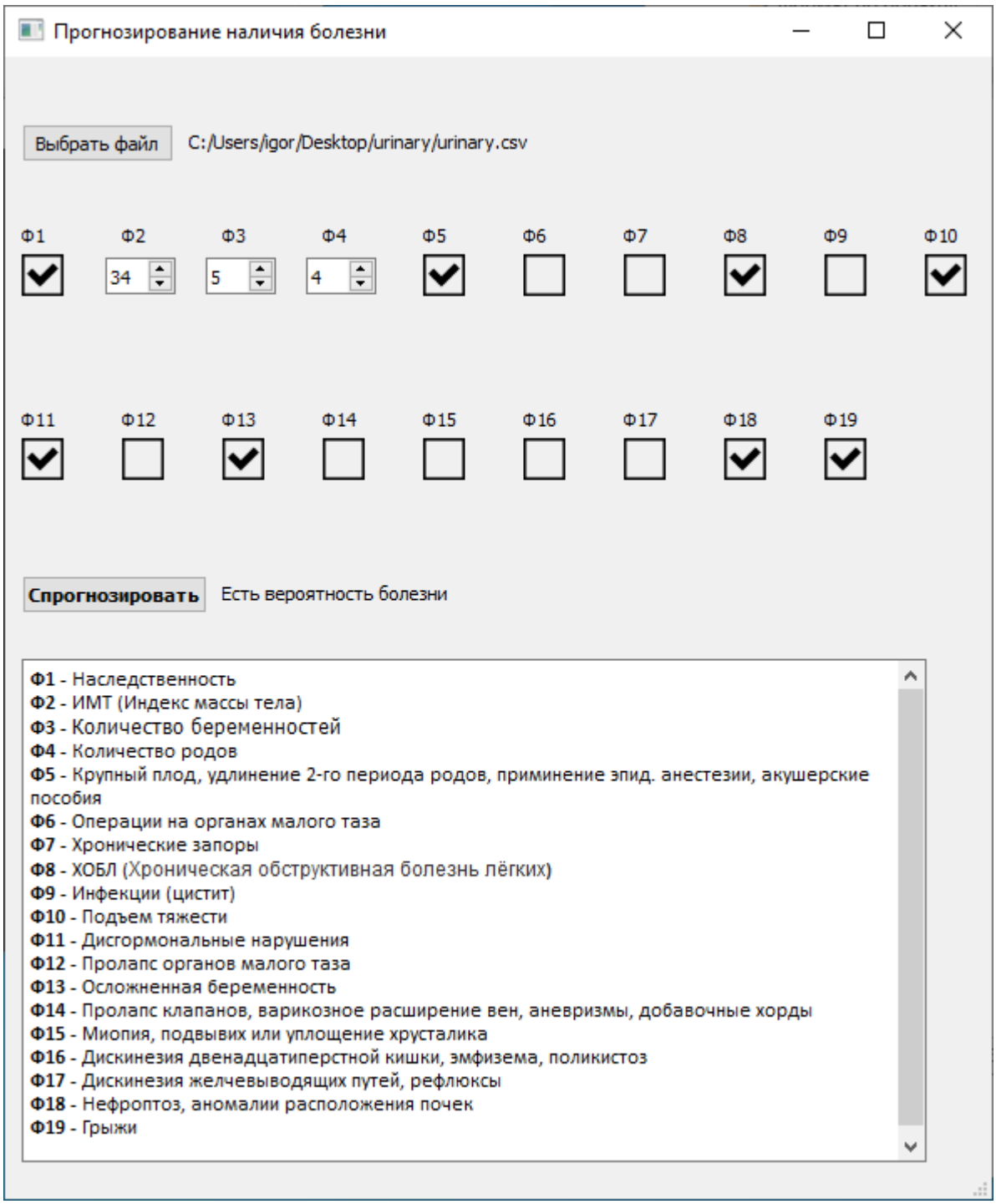
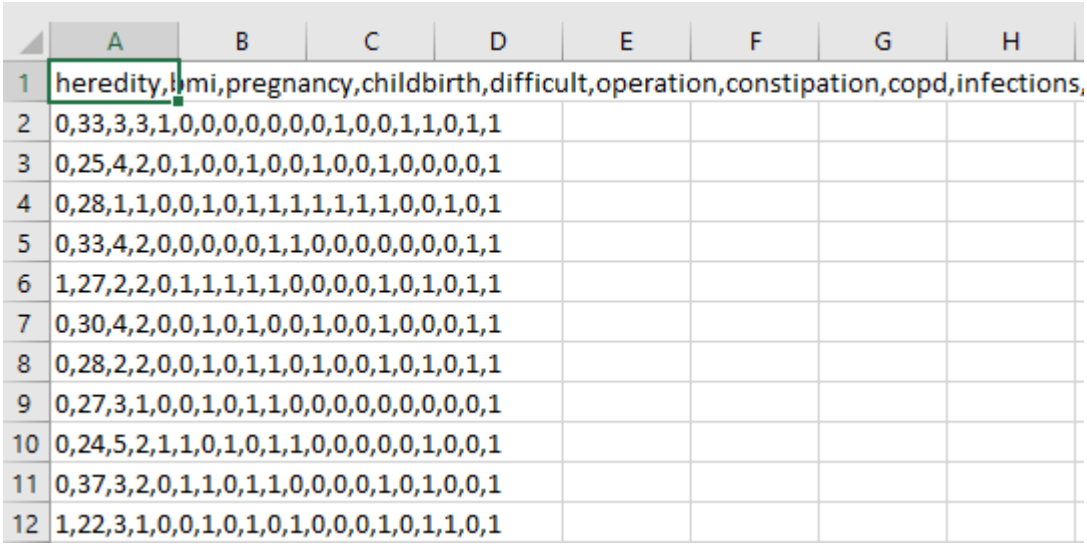


Рисунок 37 - Результат работы программы

Файл с данными urinary.csv выглядит так, как показано на рисунке ниже. Также файл может изменяться в сторону убавления или добавления количества пациентов для обучения сети. Для этого необходимо соответственно удалить строки или добавить строки с заполненными факторами новых пацинетов, где 19 числовых данных будут характеризовать наличие признаков по колонкам по порядку, 2-4 будут количественными

числовыми данными, а остальные факторы, кроме последнего, будут показывать наличие (1) или отсутствие (0) признака у пациента. Последний столбец target отвечает за наличие (1) или отсутсвие (0) болезни у пациента.

Для корректировки количества самих факторов придётся изменять программный код, поэтому обычному пользователю этого делать не стоит.



	A	B	C	D	E	F	G	H
1	heredity,bmi,pregnancy,childbirth,difficult,operation,constipation,copd,infections,							
2	0,33,3,3,1,0,0,0,0,0,0,0,1,0,0,1,1,0,1,1							
3	0,25,4,2,0,1,0,0,1,0,0,1,0,0,1,0,0,0,0,1							
4	0,28,1,1,0,0,1,0,1,1,1,1,1,1,1,0,0,1,0,1							
5	0,33,4,2,0,0,0,0,0,1,1,0,0,0,0,0,0,0,1,1							
6	1,27,2,2,0,1,1,1,1,1,0,0,0,0,1,0,1,0,1,1							
7	0,30,4,2,0,0,1,0,1,0,0,1,0,0,1,0,0,0,1,1							
8	0,28,2,2,0,0,1,0,1,1,0,1,0,0,1,0,1,0,1,1							
9	0,27,3,1,0,0,1,0,1,1,0,0,0,0,0,0,0,0,0,1							
10	0,24,5,2,1,1,0,1,0,1,1,0,0,0,0,0,0,1,0,0,1							
11	0,37,3,2,0,1,1,0,1,1,0,0,0,0,1,0,1,0,0,1							
12	1,22,3,1,0,0,1,0,1,0,1,0,0,0,1,0,1,1,0,1							

Рисунок 38 - Файл с данными

ЗАКЛЮЧЕНИЕ

Результатом работы является проект на основе машинного обучения по прогнозированию наличия или отсутсвия болезни недержания мочи у пациенток.

Программа написана на языке Python и упакована в архив с файлами, который можно распоковать на любом компьютере с ОС Windows и запустить исполняемый файл программы. Данный проект полностью удовлетворяет поставленной цели.

К особенностям проекта можно отснести его возможность использования файла обучения, данные которого можно изменять, дополняя количество текущих данных пациентов для обучения моедли или корректируя данные текущих пациентов. При наличии новых данных, врач имеет возможность дополнить файл и улучшить качество обучения сети, что повлияет на качество итогового результата прогнозирования.

Отладка и тестирование, проведенные на созданной программе, позволили исправить ошибки в программе и добиться получения корректных результатов ее работы.

Работа над проектом позволила расширить знания и улучшить навыки создания нейронных сетей и их обучения для прогнозирования.

В качестве расширения и доработки данного программного продукта можно рассматривать разработку ПО под ОС Linux, доработку в сторону использования прогнозирования для других болезней, усовершенствование файла обучения для более качественного прогноза модели, увеличение гибкости программы в плане возможности изменения количества факторов самим пользователем.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1) Каниа Алексеевич Кан: Нейронный сети. Эволюция, SelfPub; 2018. – 288с.
2) Тарик Рашид: Создаем нейронную сеть, Пер. с англ. — СПб. : ООО “Альфа-книга”, 2017. — 272 с.
3) А.Е. Белов: Применение искусственых нейронных сетей для диагностирования сердечно-сосудистых заболеваний, Барнаул, 2019 – 41с.
4) Kaggle [Электронный ресурс] – Статья с сайта Kaggle. URL: https://www.kaggle.com/sushantupadhyay/heart-disease-data-analysis-max-score-93-44
5) LiberBear [Электронный ресурс] – Статья из сайта. URL: https://liberbear.com/blog/pyqt5-python/
6) Intechopen [Электронный ресурс] – Статья из сайта. URL: https://www.intechopen.com/books/digital-systems/neural-network-principles-and-applications
7) Scikit-learn [Электронный ресурс] – Статья с сайта. URL: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html
8) Nanonets [Электронный ресурс] – Статья с сайта Nanonets. URL: https://nanonets.com/blog/deep-learning-for-medical-imaging/
9) Xenonstack [Электронный ресурс] – Статья с сайта Xenonstack. URL: https://www.xenonstack.com/blog/artificial-neural-network-applications/
10) Towards data science [Электронный ресурс] – Статья с сайта Towards data science. URL: https://towardsdatascience.com/classification-using-neural-networks-b8e98f3a904f

11) Obscurityway [Электронный ресурс] – Статья с сайта. URL: http://obscurityway.blogspot.com/2011/03/spyder.html

Приложение А
Листинги программы

Приложение А.1
Файл «urinary.py»

```
import warnings
warnings.filterwarnings('ignore')

# Any results you write to the current directory are saved as output.
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sn

#%matplotlib inline
data = pd.read_csv('urinary.csv')
data.head(2)
data.info

data.describe()

HaveDisease = len(data[data.target == 1])
HaventDisease = len(data[data.target == 0])
plt.pie(x=[HaveDisease, HaventDisease], explode=(0, 0), labels=['HaveDisease', 'HaventDisease'], autopct='%1.2f%%', shadow=True, startangle=90)
plt.show()

# removing target columns from dataset
```

```
X = data.drop(['target'], axis=1)
y = data.target

X = (X - X.min())/(X.max()-X.min())
X.head()

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=2)

from sklearn.linear_model import LogisticRegression
lr = LogisticRegression()
lr.fit(X_train, y_train)
# checking the score at test data
print(lr.score(X_test, y_test))
res = lr.predict(X_test)
print(res)
df = pd.DataFrame(zip(X.columns, np.transpose(lr.coef_)))

from sklearn.ensemble import GradientBoostingClassifier
gbc = GradientBoostingClassifier()
gbc.fit(X_train, y_train)
print(gbc.score(X_test, y_test))
```

Приложение А.2

Файл «design.py»

```
from PyQt5 import QtCore, QtGui, QtWidgets


class Ui_MainWindow(object):
```

```
    def setupUi(self, MainWindow):
        MainWindow.setObjectName("MainWindow")
        MainWindow.resize(1069, 777)
        MainWindow.setMinimumSize(QtCore.QSize(30, 30))
        MainWindow.setStyleSheet("QCheckBox::indicator:checked
{\n"
"  image: url(:/images/checkbox_checked.png);\n"
"}\n"
"\n"
"QCheckBox::indicator:unchecked {\n"
"   \n"
"    image: url(:/images/checkbox_unchecked.png);\n"
"}\n"
"QCheckBox::indicator:unchecked {\n"
"    image: url(:/images/checkbox_unchecked.png);\n"
"}\n"
"QCheckBox::indicator:checked {\n"
"    image: url(:/images/checkbox_checked.png);\n"
"}")
        self.centralwidget = QtWidgets.QWidget(MainWindow)
        self.centralwidget.setObjectName("centralwidget")
        self.label = QtWidgets.QLabel(self.centralwidget)
        self.label.setGeometry(QtCore.QRect(10, 100, 47, 13))
        self.label.setObjectName("label")
        self.label_2 = QtWidgets.QLabel(self.centralwidget)
        self.label_2.setGeometry(QtCore.QRect(70, 100, 47, 13))
        self.label_2.setObjectName("label_2")
        self.label_3 = QtWidgets.QLabel(self.centralwidget)
        self.label_3.setGeometry(QtCore.QRect(190, 100, 47, 13))
        self.label_3.setObjectName("label_3")
        self.label_4 = QtWidgets.QLabel(self.centralwidget)
```

```
self.label_4.setGeometry(QtCore.QRect(130, 100, 47, 13))
self.label_4.setObjectName("label_4")
self.label_6 = QtWidgets.QLabel(self.centralwidget)
self.label_6.setGeometry(QtCore.QRect(370, 100, 47, 13))
self.label_6.setObjectName("label_6")
self.label_7 = QtWidgets.QLabel(self.centralwidget)
self.label_7.setGeometry(QtCore.QRect(310, 100, 47, 13))
self.label_7.setObjectName("label_7")
self.label_8 = QtWidgets.QLabel(self.centralwidget)
self.label_8.setGeometry(QtCore.QRect(250, 100, 47, 13))
self.label_8.setObjectName("label_8")
self.label_9 = QtWidgets.QLabel(self.centralwidget)
self.label_9.setGeometry(QtCore.QRect(490, 100, 47, 13))
self.label_9.setObjectName("label_9")
self.label_10 = QtWidgets.QLabel(self.centralwidget)
self.label_10.setGeometry(QtCore.QRect(430, 100, 47, 13))
self.label_10.setObjectName("label_10")
self.label_11 = QtWidgets.QLabel(self.centralwidget)
self.label_11.setGeometry(QtCore.QRect(190, 210, 47, 13))
self.label_11.setObjectName("label_11")
self.label_12 = QtWidgets.QLabel(self.centralwidget)
self.label_12.setGeometry(QtCore.QRect(370, 210, 47, 13))
self.label_12.setObjectName("label_12")
self.label_13 = QtWidgets.QLabel(self.centralwidget)
self.label_13.setGeometry(QtCore.QRect(10, 210, 47, 13))
self.label_13.setObjectName("label_13")
self.label_14 = QtWidgets.QLabel(self.centralwidget)
self.label_14.setGeometry(QtCore.QRect(250, 210, 47, 13))
self.label_14.setObjectName("label_14")
self.label_15 = QtWidgets.QLabel(self.centralwidget)
self.label_15.setGeometry(QtCore.QRect(310, 210, 47, 13))
```

```
        self.label_15.setObjectName("label_15")
        self.label_16 = QtWidgets.QLabel(self.centralwidget)
        self.label_16.setGeometry(QtCore.QRect(130, 210, 47, 13))
        self.label_16.setObjectName("label_16")
        self.label_17 = QtWidgets.QLabel(self.centralwidget)
        self.label_17.setGeometry(QtCore.QRect(430, 210, 47, 13))
        self.label_17.setObjectName("label_17")
        self.label_18 = QtWidgets.QLabel(self.centralwidget)
        self.label_18.setGeometry(QtCore.QRect(490, 210, 47, 13))
        self.label_18.setObjectName("label_18")
        self.label_19 = QtWidgets.QLabel(self.centralwidget)
        self.label_19.setGeometry(QtCore.QRect(70, 210, 47, 13))
        self.label_19.setObjectName("label_19")
        self.label_20 = QtWidgets.QLabel(self.centralwidget)
        self.label_20.setGeometry(QtCore.QRect(550, 100, 47, 13))
        self.label_20.setObjectName("label_20")
        self.pushButton                                          =
QtWidgets.QPushButton(self.centralwidget)
        self.pushButton.setGeometry(QtCore.QRect(10,   310,   111,
23))
        font = QtGui.QFont()
        font.setBold(True)
        font.setUnderline(False)
        font.setWeight(75)
        self.pushButton.setFont(font)
        self.pushButton.setObjectName("pushButton")
        self.pushButton_2                                        =
QtWidgets.QPushButton(self.centralwidget)
        self.pushButton_2.setGeometry(QtCore.QRect(10,   40,   91,
23))
        self.pushButton_2.setObjectName("pushButton_2")
```

```
        self.label_21 = QtWidgets.QLabel(self.centralwidget)
        self.label_21.setGeometry(QtCore.QRect(110, 40, 391, 21))
        self.label_21.setObjectName("label_21")
        self.label_22 = QtWidgets.QLabel(self.centralwidget)
        self.label_22.setGeometry(QtCore.QRect(130,   310,   311,
21))
        self.label_22.setObjectName("label_22")
        self.textBrowser                                        =
QtWidgets.QTextBrowser(self.centralwidget)
        self.textBrowser.setGeometry(QtCore.QRect(10,  360,  541,
301))
        self.textBrowser.setObjectName("textBrowser")
        self.checkBox_3 = QtWidgets.QCheckBox(self.centralwidget)
        self.checkBox_3.setEnabled(True)
        self.checkBox_3.setGeometry(QtCore.QRect(250,   110,   30,
41))
        self.checkBox_3.setMinimumSize(QtCore.QSize(30, 30))
        self.checkBox_3.setStyleSheet("QCheckBox {\n"
"    spacing: 5px;\n"
"}\n"
"\n"
"QCheckBox::indicator {\n"
"    width: 25px;\n"
"    height: 25px;\n"
"}\n"
"QCheckBox::indicator:checked {\n"
"    image: url(:/images/checkbox_checked.png);\n"
"}\n"
"\n"
"QCheckBox::indicator:unchecked {\n"
"    image: url(:/images/checkbox_unchecked.png);\n"
```

```
"}")
        self.checkBox_3.setText("")
        self.checkBox_3.setChecked(False)
        self.checkBox_3.setAutoRepeat(False)
        self.checkBox_3.setAutoExclusive(False)
        self.checkBox_3.setObjectName("checkBox_3")
        self.checkBox_4 = QtWidgets.QCheckBox(self.centralwidget)
        self.checkBox_4.setEnabled(True)
        self.checkBox_4.setGeometry(QtCore.QRect(310,  110,  30,
41))
        self.checkBox_4.setMinimumSize(QtCore.QSize(30, 30))
        self.checkBox_4.setStyleSheet("QCheckBox {\n"
"    spacing: 5px;\n"
"}\n"
"\n"
"QCheckBox::indicator {\n"
"    width: 25px;\n"
"    height: 25px;\n"
"}\n"
"")
        self.checkBox_4.setText("")
        self.checkBox_4.setChecked(False)
        self.checkBox_4.setAutoRepeat(False)
        self.checkBox_4.setAutoExclusive(False)
        self.checkBox_4.setObjectName("checkBox_4")
        self.checkBox_6 = QtWidgets.QCheckBox(self.centralwidget)
        self.checkBox_6.setEnabled(True)
        self.checkBox_6.setGeometry(QtCore.QRect(430,  110,  30,
41))
        self.checkBox_6.setMinimumSize(QtCore.QSize(30, 30))
        self.checkBox_6.setStyleSheet("QCheckBox {\n"
```

```
"    spacing: 5px;\n"
"}\n"
"\n"
"QCheckBox::indicator {\n"
"    width: 25px;\n"
"    height: 25px;\n"
"}\n"
"")
        self.checkBox_6.setText("")
        self.checkBox_6.setChecked(False)
        self.checkBox_6.setAutoRepeat(False)
        self.checkBox_6.setAutoExclusive(False)
        self.checkBox_6.setObjectName("checkBox_6")
        self.checkBox_7 = QtWidgets.QCheckBox(self.centralwidget)
        self.checkBox_7.setEnabled(True)
        self.checkBox_7.setGeometry(QtCore.QRect(490,  110,  30,
41))
        self.checkBox_7.setMinimumSize(QtCore.QSize(30, 30))
        self.checkBox_7.setStyleSheet("QCheckBox {\n"
"    spacing: 5px;\n"
"}\n"
"\n"
"QCheckBox::indicator {\n"
"    width: 25px;\n"
"    height: 25px;\n"
"}\n"
"")
        self.checkBox_7.setText("")
        self.checkBox_7.setChecked(False)
        self.checkBox_7.setAutoRepeat(False)
        self.checkBox_7.setAutoExclusive(False)
```

```
        self.checkBox_7.setObjectName("checkBox_7")
        self.checkBox_5 = QtWidgets.QCheckBox(self.centralwidget)
        self.checkBox_5.setEnabled(True)
        self.checkBox_5.setGeometry(QtCore.QRect(370,  110,  30,
41))
        self.checkBox_5.setMinimumSize(QtCore.QSize(30, 30))
        self.checkBox_5.setStyleSheet("QCheckBox {\n"
"    spacing: 5px;\n"
"}\n"
"\n"
"QCheckBox::indicator {\n"
"    width: 25px;\n"
"    height: 25px;\n"
"}\n"
"")
        self.checkBox_5.setText("")
        self.checkBox_5.setChecked(False)
        self.checkBox_5.setAutoRepeat(False)
        self.checkBox_5.setAutoExclusive(False)
        self.checkBox_5.setObjectName("checkBox_5")
        self.checkBox_8 = QtWidgets.QCheckBox(self.centralwidget)
        self.checkBox_8.setEnabled(True)
        self.checkBox_8.setGeometry(QtCore.QRect(550,  110,  30,
41))
        self.checkBox_8.setMinimumSize(QtCore.QSize(30, 30))
        self.checkBox_8.setStyleSheet("QCheckBox {\n"
"    spacing: 5px;\n"
"}\n"
"\n"
"QCheckBox::indicator {\n"
"    width: 25px;\n"
```

```
"    height: 25px;\n"
"}\n"
"")
        self.checkBox_8.setText("")
        self.checkBox_8.setChecked(False)
        self.checkBox_8.setAutoRepeat(False)
        self.checkBox_8.setAutoExclusive(False)
        self.checkBox_8.setObjectName("checkBox_8")
        self.checkBox_15                                        =
QtWidgets.QCheckBox(self.centralwidget)
        self.checkBox_15.setEnabled(True)
        self.checkBox_15.setGeometry(QtCore.QRect(370,  220,  30,
41))
        self.checkBox_15.setMinimumSize(QtCore.QSize(30, 30))
        self.checkBox_15.setStyleSheet("QCheckBox {\n"
"    spacing: 5px;\n"
"}\n"
"\n"
"QCheckBox::indicator {\n"
"    width: 25px;\n"
"    height: 25px;\n"
"}\n"
"")
        self.checkBox_15.setText("")
        self.checkBox_15.setChecked(False)
        self.checkBox_15.setAutoRepeat(False)
        self.checkBox_15.setAutoExclusive(False)
        self.checkBox_15.setObjectName("checkBox_15")
        self.checkBox_10                                        =
QtWidgets.QCheckBox(self.centralwidget)
        self.checkBox_10.setEnabled(True)
```

```
        self.checkBox_10.setGeometry(QtCore.QRect(70,  220,  30,
41))
        self.checkBox_10.setMinimumSize(QtCore.QSize(30, 30))
        self.checkBox_10.setStyleSheet("QCheckBox {\n"
"    spacing: 5px;\n"
"}\n"
"\n"
"QCheckBox::indicator {\n"
"    width: 25px;\n"
"    height: 25px;\n"
"}\n"
"")
        self.checkBox_10.setText("")
        self.checkBox_10.setChecked(False)
        self.checkBox_10.setAutoRepeat(False)
        self.checkBox_10.setAutoExclusive(False)
        self.checkBox_10.setObjectName("checkBox_10")
        self.checkBox_14                                        =
QtWidgets.QCheckBox(self.centralwidget)
        self.checkBox_14.setEnabled(True)
        self.checkBox_14.setGeometry(QtCore.QRect(310,  220,  30,
41))
        self.checkBox_14.setMinimumSize(QtCore.QSize(30, 30))
        self.checkBox_14.setStyleSheet("QCheckBox {\n"
"    spacing: 5px;\n"
"}\n"
"\n"
"QCheckBox::indicator {\n"
"    width: 25px;\n"
"    height: 25px;\n"
"}\n"
```

```
"")
        self.checkBox_14.setText("")
        self.checkBox_14.setChecked(False)
        self.checkBox_14.setAutoRepeat(False)
        self.checkBox_14.setAutoExclusive(False)
        self.checkBox_14.setObjectName("checkBox_14")
        self.checkBox_11                                        =
QtWidgets.QCheckBox(self.centralwidget)
        self.checkBox_11.setEnabled(True)
        self.checkBox_11.setGeometry(QtCore.QRect(130,  220,  30,
41))
        self.checkBox_11.setMinimumSize(QtCore.QSize(30, 30))
        self.checkBox_11.setStyleSheet("QCheckBox {\n"
"    spacing: 5px;\n"
"}\n"
"\n"
"QCheckBox::indicator {\n"
"    width: 25px;\n"
"    height: 25px;\n"
"}\n"
"")
        self.checkBox_11.setText("")
        self.checkBox_11.setChecked(False)
        self.checkBox_11.setAutoRepeat(False)
        self.checkBox_11.setAutoExclusive(False)
        self.checkBox_11.setObjectName("checkBox_11")
        self.checkBox_9 = QtWidgets.QCheckBox(self.centralwidget)
        self.checkBox_9.setEnabled(True)
        self.checkBox_9.setGeometry(QtCore.QRect(10,   220,   30,
41))
        self.checkBox_9.setMinimumSize(QtCore.QSize(30, 30))
```

```
        self.checkBox_9.setStyleSheet("QCheckBox {\n"
"    spacing: 5px;\n"
"}\n"
"\n"
"QCheckBox::indicator {\n"
"    width: 25px;\n"
"    height: 25px;\n"
"}\n"
"")
        self.checkBox_9.setText("")
        self.checkBox_9.setChecked(False)
        self.checkBox_9.setAutoRepeat(False)
        self.checkBox_9.setAutoExclusive(False)
        self.checkBox_9.setObjectName("checkBox_9")
        self.checkBox_13                                        =
QtWidgets.QCheckBox(self.centralwidget)
        self.checkBox_13.setEnabled(True)
        self.checkBox_13.setGeometry(QtCore.QRect(250,  220,  30,
41))
        self.checkBox_13.setMinimumSize(QtCore.QSize(30, 30))
        self.checkBox_13.setStyleSheet("QCheckBox {\n"
"    spacing: 5px;\n"
"}\n"
"\n"
"QCheckBox::indicator {\n"
"    width: 25px;\n"
"    height: 25px;\n"
"}\n"
"")
        self.checkBox_13.setText("")
        self.checkBox_13.setChecked(False)
```

```
        self.checkBox_13.setAutoRepeat(False)
        self.checkBox_13.setAutoExclusive(False)
        self.checkBox_13.setObjectName("checkBox_13")
        self.checkBox_12                                        =
QtWidgets.QCheckBox(self.centralwidget)
        self.checkBox_12.setEnabled(True)
        self.checkBox_12.setGeometry(QtCore.QRect(190,  220,  30,
41))
        self.checkBox_12.setMinimumSize(QtCore.QSize(30, 30))
        self.checkBox_12.setStyleSheet("QCheckBox {\n"
"    spacing: 5px;\n"
"}\n"
"\n"
"QCheckBox::indicator {\n"
"    width: 25px;\n"
"    height: 25px;\n"
"}\n"
"")
        self.checkBox_12.setText("")
        self.checkBox_12.setChecked(False)
        self.checkBox_12.setAutoRepeat(False)
        self.checkBox_12.setAutoExclusive(False)
        self.checkBox_12.setObjectName("checkBox_12")
        self.checkBox_16                                        =
QtWidgets.QCheckBox(self.centralwidget)
        self.checkBox_16.setEnabled(True)
        self.checkBox_16.setGeometry(QtCore.QRect(430,  220,  30,
41))
        self.checkBox_16.setMinimumSize(QtCore.QSize(30, 30))
        self.checkBox_16.setStyleSheet("QCheckBox {\n"
"    spacing: 5px;\n"
```

```
"}\n"
"\n"
"QCheckBox::indicator {\n"
"    width: 25px;\n"
"    height: 25px;\n"
"}\n"
"")
        self.checkBox_16.setText("")
        self.checkBox_16.setChecked(False)
        self.checkBox_16.setAutoRepeat(False)
        self.checkBox_16.setAutoExclusive(False)
        self.checkBox_16.setObjectName("checkBox_16")
        self.checkBox_17                                        =
QtWidgets.QCheckBox(self.centralwidget)
        self.checkBox_17.setEnabled(True)
        self.checkBox_17.setGeometry(QtCore.QRect(490,  220,  30,
41))
        self.checkBox_17.setMinimumSize(QtCore.QSize(30, 30))
        self.checkBox_17.setStyleSheet("QCheckBox {\n"
"    spacing: 5px;\n"
"}\n"
"\n"
"QCheckBox::indicator {\n"
"    width: 25px;\n"
"    height: 25px;\n"
"}\n"
"")
        self.checkBox_17.setText("")
        self.checkBox_17.setChecked(False)
        self.checkBox_17.setAutoRepeat(False)
        self.checkBox_17.setAutoExclusive(False)
```

```
        self.checkBox_17.setObjectName("checkBox_17")
        self.spinBox = QtWidgets.QSpinBox(self.centralwidget)
        self.spinBox.setGeometry(QtCore.QRect(60, 120, 42, 22))
        self.spinBox.setObjectName("spinBox")
        self.spinBox_2 = QtWidgets.QSpinBox(self.centralwidget)
        self.spinBox_2.setGeometry(QtCore.QRect(120,   120,   42,
22))
        self.spinBox_2.setObjectName("spinBox_2")
        self.checkBox = QtWidgets.QCheckBox(self.centralwidget)
        self.checkBox.setEnabled(True)
        self.checkBox.setGeometry(QtCore.QRect(10, 110, 30, 41))
        self.checkBox.setMinimumSize(QtCore.QSize(30, 30))
        self.checkBox.setAutoFillBackground(False)
        self.checkBox.setStyleSheet("QCheckBox {\n"
"    spacing: 5px;\n"
"}\n"
"\n"
"QCheckBox::indicator {\n"
"    width: 25px;\n"
"    height: 25px;\n"
"}\n"
"QCheckBox::indicator:checked {\n"
"    image: url(:/images/checkbox_checked.png);\n"
"}\n"
"\n"
"QCheckBox::indicator:unchecked {\n"
"    image: url(:/images/checkbox_unchecked.png);\n"
"}")
        self.checkBox.setText("")
        self.checkBox.setAutoRepeat(False)
        self.checkBox.setAutoExclusive(False)
```

```
        self.checkBox.setObjectName("checkBox")
        self.spinBox_3 = QtWidgets.QSpinBox(self.centralwidget)
        self.spinBox_3.setGeometry(QtCore.QRect(180,   120,   42,
22))
        self.spinBox_3.setObjectName("spinBox_3")
        MainWindow.setCentralWidget(self.centralwidget)
        self.menubar = QtWidgets.QMenuBar(MainWindow)
        self.menubar.setGeometry(QtCore.QRect(0, 0, 1069, 21))
        self.menubar.setObjectName("menubar")
        MainWindow.setMenuBar(self.menubar)
        self.statusbar = QtWidgets.QStatusBar(MainWindow)
        self.statusbar.setObjectName("statusbar")
        MainWindow.setStatusBar(self.statusbar)

        self.retranslateUi(MainWindow)
        QtCore.QMetaObject.connectSlotsByName(MainWindow)

    def retranslateUi(self, MainWindow):
        _translate = QtCore.QCoreApplication.translate
        MainWindow.setWindowTitle(_translate("MainWindow",
"Прогнозирование наличия болезни"))
        self.label.setText(_translate("MainWindow", "Ф1"))
        self.label_2.setText(_translate("MainWindow", "Ф2"))
        self.label_3.setText(_translate("MainWindow", "Ф4"))
        self.label_4.setText(_translate("MainWindow", "Ф3"))
        self.label_6.setText(_translate("MainWindow", "Ф7"))
        self.label_7.setText(_translate("MainWindow", "Ф6"))
        self.label_8.setText(_translate("MainWindow", "Ф5"))
        self.label_9.setText(_translate("MainWindow", "Ф9"))
        self.label_10.setText(_translate("MainWindow", "Ф8"))
        self.label_11.setText(_translate("MainWindow", "Ф14"))
```

```
        self.label_12.setText(_translate("MainWindow", "Ф17"))
        self.label_13.setText(_translate("MainWindow", "Ф11"))
        self.label_14.setText(_translate("MainWindow", "Ф15"))
        self.label_15.setText(_translate("MainWindow", "Ф16"))
        self.label_16.setText(_translate("MainWindow", "Ф13"))
        self.label_17.setText(_translate("MainWindow", "Ф18"))
        self.label_18.setText(_translate("MainWindow", "Ф19"))
        self.label_19.setText(_translate("MainWindow", "Ф12"))
        self.label_20.setText(_translate("MainWindow", "Ф10"))
        self.pushButton.setText(_translate("MainWindow",
"Спрогнозировать"))
        self.pushButton_2.setText(_translate("MainWindow",
"Выбрать файл"))
        self.label_21.setText(_translate("MainWindow", "Путь"))
        self.label_22.setText(_translate("MainWindow",
"Результат"))
        self.textBrowser.setHtml(_translate("MainWindow",
"<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.0//EN\"
\"http://www.w3.org/TR/REC-html40/strict.dtd\">\n"
"<html><head><meta name=\"qrichtext\" content=\"1\" /><style
type=\"text/css\">\n"
"p, li { white-space: pre-wrap; }\n"
"</style></head><body style=\" font-family:\'MS Shell Dlg 2\';
font-size:8.25pt; font-weight:400; font-style:normal;\">\n"
"<p style=\" margin-top:0px; margin-bottom:0px; margin-left:0px;
margin-right:0px; -qt-block-indent:0; text-indent:0px;\"><span
style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt;
font-weight:600; color:#000000;\">Ф1</span><span style=\" font-
family:\'Calibri\',\'sans-serif\'; font-size:9pt;
color:#000000;\"> </span><span style=\" font-
family:\'Calibri\',\'sans-serif\'; font-size:9pt; font-
weight:600; color:#000000;\">-</span><span style=\" font-
```

family:\'Calibri\',\'sans-serif\'; font-size:9pt; color:#000000;\"> Наследственность</p>\n"

"<p style=\" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;\"><span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt; font-weight:600; color:#000000;\">Ф2 - <span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt; color:#000000;\">ИМТ (Индекс массы тела)</p>\n"

"<p style=\" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;\"><span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt; font-weight:600; color:#000000;\">Ф3 - <span style=\" font-size:9pt; color:#000000; vertical-align:bottom;\">Количество беременностей<span style=\" font-size:8pt; vertical-align:bottom;\"> </p>\n"

"<p style=\" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;\"><span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt; font-weight:600; color:#000000;\">Ф4 - <span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt; color:#000000;\">Количество родов</p>\n"

"<p style=\" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;\"><span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt; font-weight:600; color:#000000;\">Ф5 - <span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt; color:#000000;\">Крупный плод, удлинение 2-го периода родов, приминение эпид. анестезии, акушерские пособия</p>\n"

"<p style=\" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;\"><span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt; font-weight:600; color:#000000;\">Ф6<span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt; color:#000000;\"> <span style=\" font-

family:\'Calibri\',\'sans-serif\'; font-size:9pt; font-weight:600; color:#000000;\">-<span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt; color:#000000;\"> Операции на органах малого таза</p>\n"

"<p style=\" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;\"><span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt; font-weight:600; color:#000000;\">Ф7 - <span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt; color:#000000;\">Хронические запоры</p>\n"

"<p style=\" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;\"><span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt; font-weight:600; color:#000000;\">Ф8 - <span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt; color:#000000;\">ХОБЛ (<span style=\" font-family:\'Arial\',\'Helvetica\',\'sans-serif\'; font-size:9pt; color:#333333; background-color:#ffffff;\">Хроническая обструктивная болезнь лёгких <span style=\" font-family:\'Arial\',\'Helvetica\',\'sans-serif\'; font-size:8pt; font-weight:600; color:#333333; background-color:#ffffff;\">)</p>\n"

"<p style=\" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;\"><span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt; font-weight:600; color:#000000;\">Ф9 - <span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt; color:#000000;\">Инфекции (цистит)</p>\n"

"<p style=\" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;\"><span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt; font-weight:600; color:#000000;\">Ф10 - <span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt; color:#000000;\">Подъем тяжести</p>\n"

"<p style=\" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;\"><span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt; font-weight:600;\">Ф11 -<span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt;\"> Дисгормональные нарушения</p>\n"

"<p style=\" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;\"><span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt; font-weight:600; color:#000000;\">Ф12 - <span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt;\">Пролапс органов малого таза</p>\n"

"<p style=\" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;\"><span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt; font-weight:600; color:#000000;\">Ф13 - <span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt;\">Осложненная беременность</p>\n"

"<p style=\" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;\"><span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt; font-weight:600; color:#000000;\">Ф14 - <span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt; color:#000000;\">Пролапс клапанов, варикозное расширение вен, аневризмы, добавочные хорды</p>\n"

"<p style=\" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;\"><span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt; font-weight:600; color:#000000;\">Ф15 - <span style=\" font-family:\'Calibri\',\'sans-serif\'; font-size:9pt; color:#000000;\">Миопия, подвывих или уплощение хрусталика</p>\n"

"<p style=\" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;\"><span

```
style=\"  font-family:\'Calibri\',\'sans-serif\';  font-size:9pt;
font-weight:600;  color:#000000;\">Ф16  -</span><span  style=\"
font-family:\'Calibri\',\'sans-serif\';           font-size:9pt;
color:#000000;\"> Дискинезия двенадцатиперстной кишки, эмфизема,
поликистоз</span></p>\n"
"<p style=\" margin-top:0px; margin-bottom:0px; margin-left:0px;
margin-right:0px;  -qt-block-indent:0;  text-indent:0px;\"><span
style=\"  font-family:\'Calibri\',\'sans-serif\';  font-size:9pt;
font-weight:600;  color:#000000;\">Ф17  -  </span><span  style=\"
font-family:\'Calibri\',\'sans-serif\';           font-size:9pt;
color:#000000;\">Дискинезия        желчевыводящих        путей,
рефлюксы</span></p>\n"
"<p style=\" margin-top:0px; margin-bottom:0px; margin-left:0px;
margin-right:0px;  -qt-block-indent:0;  text-indent:0px;\"><span
style=\"  font-family:\'Calibri\',\'sans-serif\';  font-size:9pt;
font-weight:600;  color:#000000;\">Ф18  -  </span><span  style=\"
font-family:\'Calibri\',\'sans-serif\';           font-size:9pt;
color:#000000;\">Нефроптоз,        аномалии        расположения
почек</span></p>\n"
"<p style=\" margin-top:0px; margin-bottom:0px; margin-left:0px;
margin-right:0px;  -qt-block-indent:0;  text-indent:0px;\"><span
style=\"  font-family:\'Calibri\',\'sans-serif\';  font-size:9pt;
font-weight:600;  color:#000000;\">Ф19  -  </span><span  style=\"
font-family:\'Calibri\',\'sans-serif\';           font-size:9pt;
color:#000000;\">Грыжи</span></p>\n"
"<p  style=\"-qt-paragraph-type:empty;  margin-top:0px;  margin-
bottom:0px;   margin-left:0px;   margin-right:0px;   -qt-block-
indent:0;         text-indent:0px;         font-size:8pt;\"><br
/></p></body></html>"))
import resources
```

Приложение А.3

Файл «mainPain.py»

```
from PyQt5.QtGui import *
```

```
from PyQt5.QtCore import *
from PyQt5.QtWidgets import *

from array import *
import pandas as pd
import design
import sys
class Example(QMainWindow, design.Ui_MainWindow):
    def __init__(self):
        super().__init__()
        self.setupUi(self)
        #getFileNameButton = QPushButton("Open File")
        self.pushButton_2.clicked.connect(self.getFileName)
        self.pushButton.clicked.connect(self.calculate)

    def getFileName(self):
        filename, filetype = QFileDialog.getOpenFileName(self,
                             "Выбрать файл",
                             ".",
                             "CSV File(*.csv);;All Files(*)")

        self.label_21.setText(filename)

    def calculate(self):

        data_patient                                                =
pd.DataFrame({'heredity':float(self.checkBox.checkState()/2),'bm
i':float(self.spinBox.text()),'pregnancy':float(self.spinBox_2.t
ext()),'childbirth':float(self.spinBox_3.text()),

'difficult':float(self.checkBox_3.checkState()/2),'operation':fl
```

```
oat(self.checkBox_4.checkState()/2),'constipation':float(self.ch
eckBox_5.checkState()/2),'copd':float(self.checkBox_6.checkState
()/2),

'infections':float(self.checkBox_7.checkState()/2),
'gravity':float(self.checkBox_8.checkState()/2),'dishormonal':fl
oat(self.checkBox_9.checkState()/2),'prolapse':float(self.checkB
ox_10.checkState()/2),

'complications':float(self.checkBox_11.checkState()/2),'prolapse
2':float(self.checkBox_12.checkState()/2),'myopia':float(self.ch
eckBox_13.checkState()/2),'emphysema':float(self.checkBox_14.che
ckState()/2),

'jwp':float(self.checkBox_15.checkState()/2),'nephroptosis':floa
t(self.checkBox_16.checkState()/2),'hernias':float(self.checkBox
_17.checkState()/2),'target':float(0)},index=[0])
        data = pd.read_csv(self.label_21.text())
        data = data.append(data_patient)
        # removing target columns from dataset
        data_in = data.drop(['target'], axis=1)
        data_out = data.target
        data_out.drop(data_out.index[len(data_out)-
1],inplace=True) # drop last n rows

        data_in  =  (data_in  -  data_in.min())/(data_in.max()-
data_in.min())

        data_patient = data_in.tail(1)
        data_in.drop(data_in.index[len(data_in)-1],inplace=True)
# drop last n rows

        from sklearn.ensemble import GradientBoostingClassifier
```

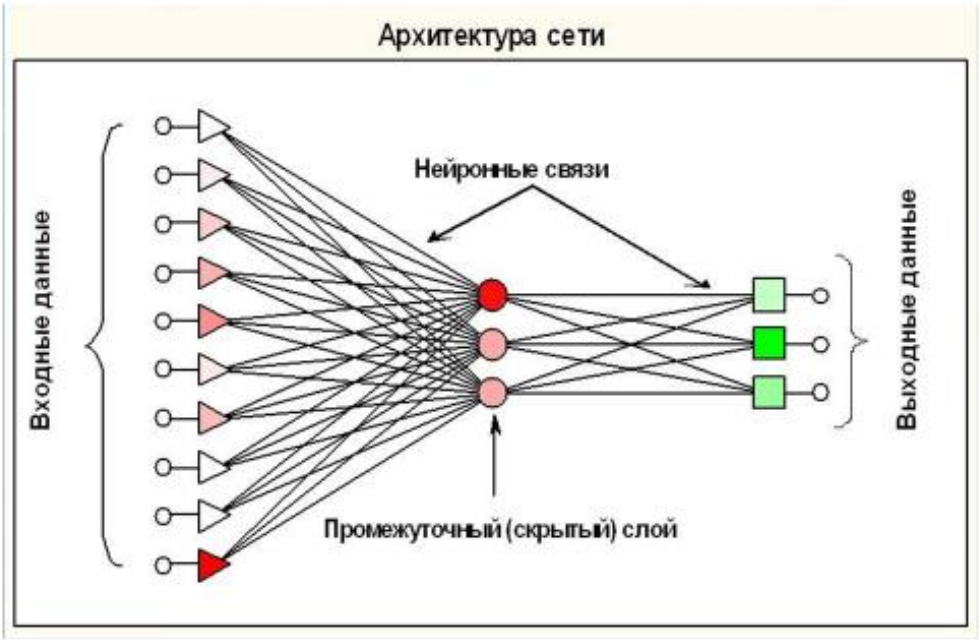
```
        gbc = GradientBoostingClassifier()
        gbc.fit(data_in, data_out)
        res = gbc.predict(data_patient)

        # checking the score at test data
        if res[0]:
            self.label_22.setText("Есть вероятность болезни")
        else:
            self.label_22.setText("Шансы заболевания малы")

if __name__ == '__main__':
    app = QApplication(sys.argv)
    form = Example()
    form.show()
    app.exec()
```

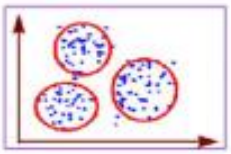
Приложение В

Презентация

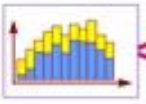


Применение нейросетей

- Классификация
- Кластеризация
- Аппроксимация функций
- Оптимизация
- Прогнозирование
- Автоассоциативная память
- Управление
- Визуализация многомерных данных



$y = x_1^3 - 4x_3^2x_6^5 + x_3^2$

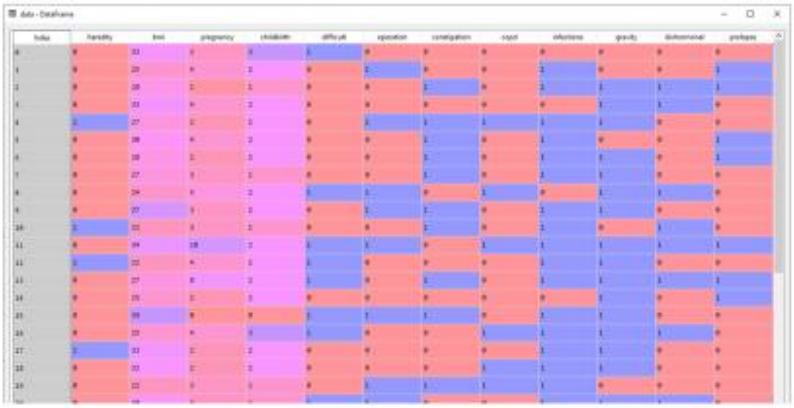


ФАЙЛ С ДАННЫМИ О ПАЦИЕНТАХ

	A	B	C	D	E	F	G	H
1	heredity,bmi,pregnancy,childbirth,difficult,operation,constipation,copd,infections,gra							
2	0,33,3,3,1,0,0,0,0,0,0,0,1,0,0,1,1,0,1,1							
3	0,25,4,2,0,1,0,0,1,0,0,1,0,0,1,0,0,0,0,1							
4	0,28,1,1,0,0,1,0,1,1,1,1,1,1,1,0,0,1,0,1							
5	0,33,4,2,0,0,0,0,0,1,1,0,0,0,0,0,0,0,1,1							
6	1,27,2,2,0,1,1,1,1,1,0,0,0,0,1,0,1,0,1,1							
7	0,30,4,2,0,0,1,0,1,0,0,1,0,0,1,0,0,0,1,1							
8	0,28,2,2,0,0,1,0,1,1,0,1,0,0,1,0,1,0,1,1							
9	0,27,3,1,0,0,1,0,1,1,0,0,0,0,0,0,0,0,0,1							
10	0,24,5,2,1,1,0,1,0,1,1,0,0,0,0,0,1,0,0,1							
11	0,37,3,2,0,1,1,0,1,1,0,0,0,0,1,0,1,0,0,1							
12	1,22,3,1,0,0,1,0,1,0,1,0,0,0,1,0,1,1,0,1							
13	0,34,10,2,1,1,0,1,1,1,1,1,1,1,1,1,0,1,0,0,1							
14	1,22,4,2,1,0,0,0,1,1,0,0,0,1,0,0,1,0,0,1							
15	0,27,8,2,1,0,1,0,1,1,1,1,0,0,0,0,1,0,1,1							
16	0,25,2,2,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,1							
17	0,39,0,0,1,1,1,0,1,1,0,0,0,0,0,0,1,0,0,1							

5

ДАННЫЕ, СЧИТАННЫЕ ПРОГРАММОЙ

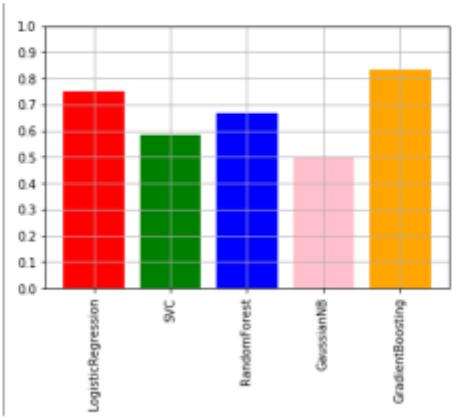


6

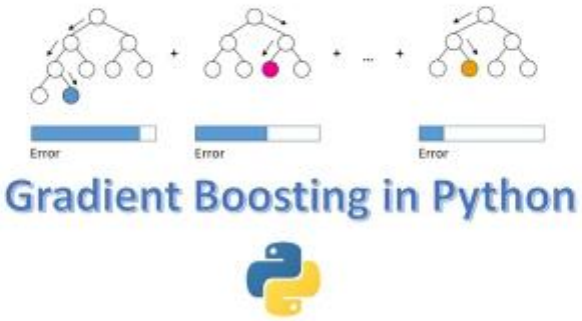
ОСНОВНОЙ АЛГОРИТМ РАБОТЫ ПРОГРАММЫ

1. Подключение необходимых библиотек
2. Считывание файла с данными
3. Разделение данных на входной и выходной сигнал
4. Нормализация входных данных
5. Обучение модели
6. Прогнозирование результата

СРАВНЕНИЕ РЕЗУЛЬТАТОВ МЕТОДОВ ОБУЧЕНИЯ



МЕТОД ГРАДИЕНТНОГО СПУСКА

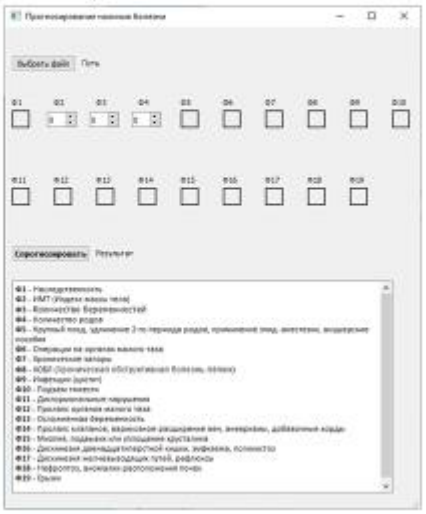


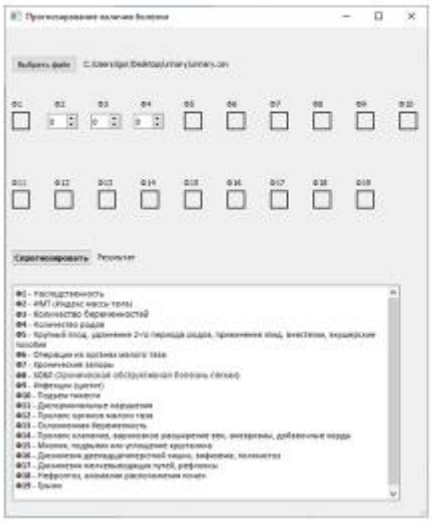
9

ВЗАИМОДЕЙСТВИЕ ПОЛЬЗОВАТЕЛЯ С ПРОГРАММОЙ

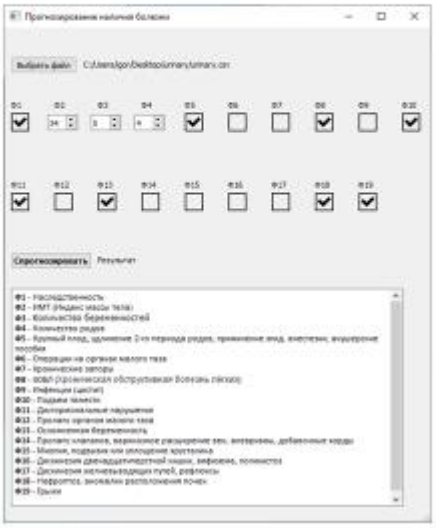
10

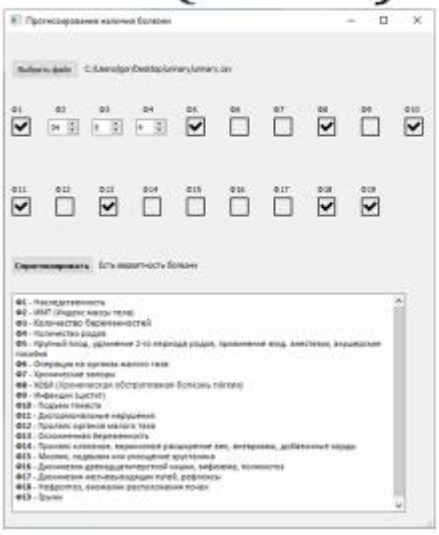
ЭКРАН ПРИЛОЖЕНИЯ(ПЕРВЫЕ ШАГИ)





ЭКРАН ПРИЛОЖЕНИЯ(ИТОГ)





- Спасибо за внимание!

Выступал: Шаповалов Игорь

Студент группы 18ВВм1