Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Проверено:

_____ _____
подпись *дата*

Учебная дисциплина
Вычислительные алгоритмы в программировании

Студенческий научно-исследовательский проект
на тему

«Диагностика и прогнозирование заболевания *Covid*-19»

Выполнил студент:

_____ _____
Подпись, дата *Инициалы, фамилия*

Руководитель

_____ А. Г. Давыдовский
Подпись, дата

Минск 2021

Задание на выполнение студенческого научно-исследовательского проекта по учебной дисциплине «Вычислительные алгоритмы в программировании»

Шуринов Никита Александрович

(фамилия, имя, отчество)

1 Тема проекта: «Диагностика и прогнозирование заболевания *Covid*-19»

2 Срок сдачи студентом проекта

3 Исходные данные к проекту: Статистические данные сайтов сети, научная литература, ОС Windows 10, приложение Microsoft Word 2007-2016, приложение Sublime Text.

Цель проекта: создание нейронной сети, диагностирующей риск заболевания *Covid*-19 на основе состояния больного.

СОДЕРЖАНИЕ

РЕФЕРАТ

«Нейронная сеть диагностики *Covid*-19» / Н.А. Шуринов. – Минск: БГУИР, 2021. – 25 с.

Пояснительная записка 25 с., 21 рисунка, 5 источников, 1 приложение.

Цель проектирования: изучение принципов диагностики и прогнозирования заболеваний. Составление и проектирование нейронной сети, способную выявить вероятность заболевания *Covid*-19.

Методология проведения работы: в процессе решения поставленных задач использованы принципы диагностики, а также знания написания нейронных сетей на языке программирования *Python*.

Результаты работы: изучено функционирование нейронных сетей на основе прогнозирования заболеваний. Применены знания о технологиях и принципах прогнозирования заболеваний при создании нейронной сети.

ВВЕДЕНИЕ

В данном проекте требуется создать нейронную сеть, способную рассчитывать вероятность заболевания *Covid*-19, также будет рассматриваться статистика заболевания и принципы диагностики. Основной проблемой является выявление главного признака заболевания.

Целью данного проекта является решение проблемы, путем применения нейронной сети для демонстрации основных проблем заболевания, а также вероятность заболевания. В процессе выполнения проекта будет создана нейронная сеть выполняющая диагностику заболевания.

Для решения проблемы необходимо составить ответы на следующие задачи:

– описание основных проблем заболевания;

– разработка нейронной сети, для диагностики заболевания.

ОБЪЕКТ И ПРЕДМЕТ ПРОЕКТА

Объектом данного проекта является прогнозирование и диагностика заболеваний. Эта область представляет собой последовательность действий, которые нужно совершить для получения модели прогнозирования. Также одним из объектов проекта является определение основных технологий диагностики и их практическое использование.

Под предметом проекта подразумевается реализация нейронной сети, выполняющая прогноз заболевания *Covid*-19.

МАТЕРИАЛЫ, ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ, МАТЕМАТИЧЕСКИЕ МОДЕЛИ, АЛГОРИТМЫ И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ

Для написания данного исследовательского проекта были использованы теоретические материалы описанные в главе «Список использованных источников». При создании использовались навыки программирования на языке *Python*. Для создания нейронной сети были применены навыки программирования на языке *Python* и знания создания самой сети.

Список программ использованных для реализации научно-исследовательского проекта:

– Sublime text;
– Microsoft Word;
– Microsoft PowerPoint.

ХАРАКТЕРИСТИКА ОРГАНИЗАЦИИ И ВЫПОЛНЕНИЯ ПРОЕКТА

Данный научно-исследовательский проект организован в три этапа:

– получение информации о принципах прогнозирования и диагностики;

– описание основных методов прогнозирования и диагностики, и их практическое применение;

– создание нейронной сети, с задачей выполнения прогнозирования вероятности заболевания.

Практическая часть проекта будет реализована в среде разработки Sublime Text при помощи языка программирования *Python*. Будут представлены результаты выполнения диагностики, а также листинг программного кода в приложении А.

РЕЗУЛЬТАТЫ ВЫПОЛНЕНИЯ ИССЛЕДОВАНИЯ

Введение в прогнозирование

Метод прогнозирования представляет собой последовательность действий, которые нужно совершить для получения модели прогнозирования.

Модель прогнозирования есть функциональное представление, адекватно описывающее исследуемый процесс и являющееся основой для получения его будущих значений.

В настоящее время принято использовать английские аббревиатуры названий как моделей, так и методов. Например, существует знаменитая модель прогнозирования авторегрессия проинтегрированного скользящего среднего с учетом внешнего фактора (*auto regression integrated moving average extended, ARIMAX*). Эту модель и соответствующий ей метод обычно называют *ARIMAX*, а иногда моделью (методом) Бокса-Дженкинса по имени авторов.

Сначала классифицируем методы

Если посмотреть внимательно, то быстро выясняется, что понятие «*метод прогнозирования*» гораздо шире понятия «*модель прогнозирования*». В связи с этим на первом этапе классификации обычно делят методы на две группы: интуитивные и формализованные (см. рисунок 1).

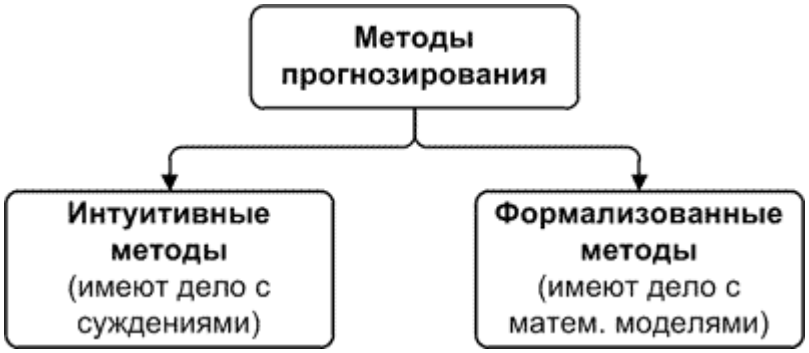


Рисунок 1 – Методы прогнозирования

Интуитивные методы прогнозирования имеют дело с суждениями и оценками экспертов. На сегодняшний день они часто применяются в маркетинге, экономике, политике, так как система, поведение которой необходимо спрогнозировать, или очень сложна и не поддается математическому описанию, или очень проста и в таком описании не нуждается.

Формализованные методы — описанные в литературе методы прогнозирования, в результате которых строят модели прогнозирования, то есть определяют такую математическую зависимость, которая позволяет вычислить будущее значение процесса, то есть сделать прогноз.

Общая классификация моделей.

Здесь необходимо переходить к классификации моделей прогнозирования. На первом этапе модели следует разделить на две группы: модели предметной области и модели временных рядов (см. рисунок 2).

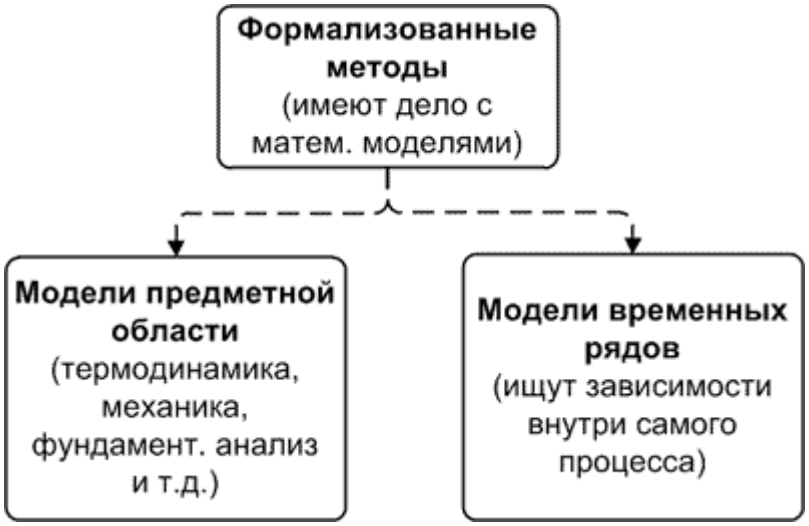


Рисунок 2 – Формализованные методы

Модели предметной области — такие математические модели прогнозирования, для построения которых используют законы предметной области. Например, модель, на которой делают прогноз погоды, содержит уравнения динамики жидкостей и термодинамики. Прогноз развития популяции делается на модели, построенной на дифференциальном уравнении. Прогноз уровня сахара крови человека, больного диабетом, делается на основании системы дифференциальных уравнений. Словом, в таких моделях используются зависимости, свойственные конкретной предметной области. Такого рода моделям свойственен индивидуальный подход в разработке.

Модели временных рядов — математические модели прогнозирования, которые стремятся найти зависимость будущего значения от прошлого внутри самого процесса и на этой зависимости вычислить прогноз. Эти модели универсальны для различных предметных областей, то есть их общий вид не меняется в зависимости от природы временного ряда. Мы можем использовать нейронные сети для прогнозирования температуры воздуха, а после аналогичную модель на нейронных сетях применить для прогноза биржевых индексов. Это обобщенные модели, как кипяток, в которые если бросить продукт, то он сварится вне зависимости от его природы.

Классификация модели временных рядов.

Модели временных рядов можно разделить на две группы: статистические и структурные.

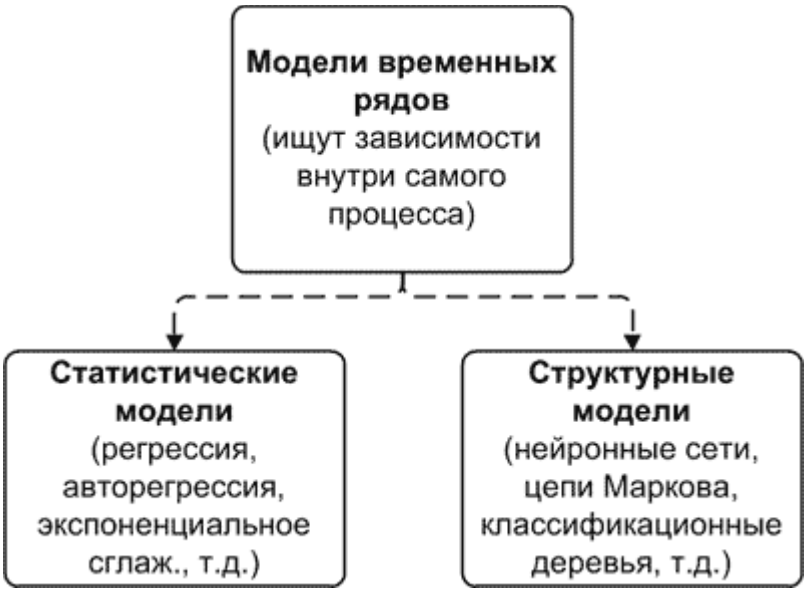


Рисунок 3 – Модели временных рядов

В статистических моделях зависимость будущего значения от прошлого задается в виде некоторого уравнения. К ним относятся:

1. регрессионные модели (линейная регрессия, нелинейная регрессия);
2. авторегрессионные модели (*ARIMAX, GARCH, ARDLM*);
3. модель экспоненциального сглаживания;
4. модель по выборке максимального подобия;
5. и т.д.

В структурных моделях зависимость будущего значения от прошлого задается в виде некоторой структуры и правил перехода по ней. К ним относятся:

1. нейросетевые модели;
2. модели на базе цепей Маркова;
3. модели на базе классификационно-регрессионных деревьев;
4. и т.д.

Для обоих групп я указала основные, то есть наиболее распространенные и подробно описанные модели прогнозирования. Однако на сегодняшний день моделей прогнозирования временных рядов имеется уже громадное количество и для построения прогнозов, например, стали использовать SVM (*support vector machine*) модели, *GA (genetic algorithm*) модели и многие другие.

Система прогнозирование на базе нейронных сетей

Все больше внимания уделяется оптимизации процессов, в-основном, в виде снижения затрат на производство продукции. Снижения затрат можно достигнуть модернизированием оборудования, но данный подход влечет за собой множество затрат на проектирование, покупку, реконструкцию и пр., а также сопровождается недополученной прибылью во время простоя реконструируемого объекта. Но также возможно использовать математический подход для поиска неэффективности в технологическом процессе, об и этом и пойдет речь далее.

Нейронная сеть представляют собой систему соединённых и взаимодействующих между собой простых процессоров (нейронов).

Структурная схема нейронной сети (зеленый цвет – входной слой нейронов, синий – скрытый(промежуточный) слой нейронов, желтый – выходной слой нейронов) (см. рисунок 4).

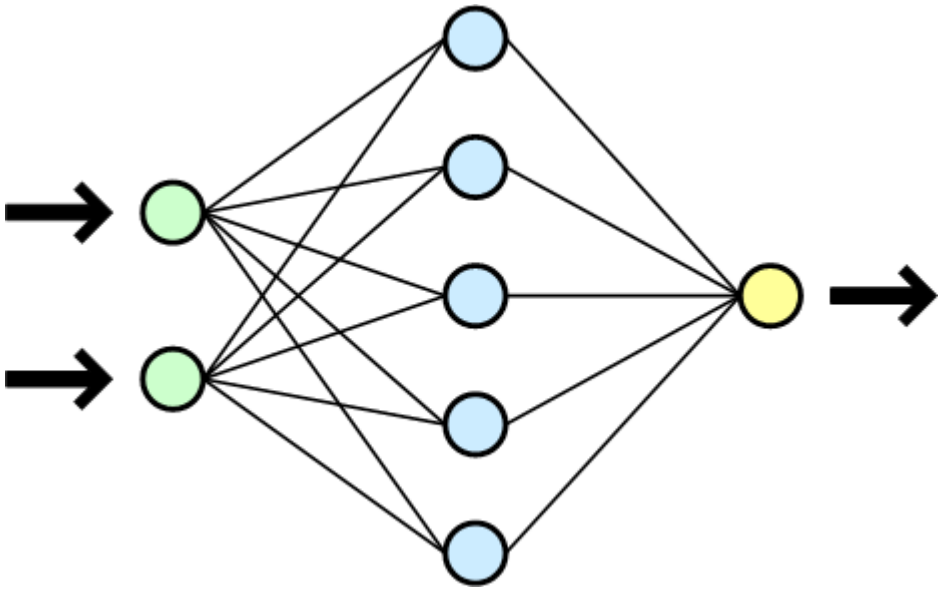


Рисунок 4 – Структурная схема нейронной сети

Нейрон – базовый элемент нейронной сети, единичный простой вычислительный процессор способный воспринимать, преобразовывать и распространять сигналы, в свою очередь объединение большого количества нейронов в одну сеть позволяет решать достаточно сложные задачи (см. рисунок 5).

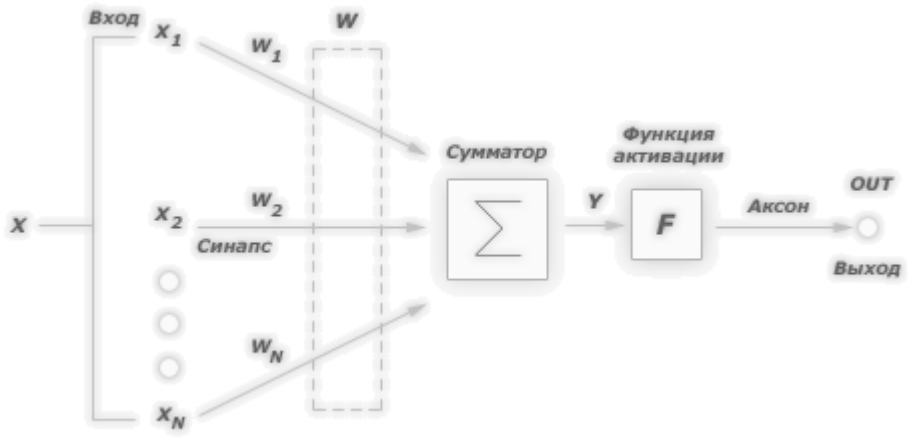


Рисунок 5 – Схема сети

Нейросетевой подход свободен от модельных ограничений, он одинаково годится для линейных и сложных нелинейных задач, а также задач классификации. Обучение нейронной сети в первую очередь заключается в изменении «силы» связей между нейронами. Нейронные сети масштабируемы, они способны решать задачи как в рамках единичного оборудования, так и в масштабах заводов в-целом.

Цель — прогнозирование содержания серы в продукте с максимально возможной точностью, что в свою очередь позволит держать основные технологические параметры в оптимальных значениях как для качества продукта, так и с точки зрения оптимизации процесса.

Единицы измерения — ppm (одна миллионная доля).

Входные данные — исторические значения технологических параметров объекта.

Данные для проверки прогноза сети — ежесуточные лабораторные анализы содержания серы.

Всего было использовано 531 наблюдение, общая выборка была поделена следующим образом: 70% наблюдений выборки использовалось для обучения сети, 30% использовалось в качестве контрольной выборки для оценки качества обучения сети и дальнейшего сравнения сетей между собой. Среднее содержание серы во всех наблюдениях составило 316,7ppm. Всего по результатам обучения было отобрано 4 сети, сети имеют следующую конфигурацию:

Сеть	№1:	20-22-1
Сеть	№2:	20-26-1
Сеть	№3:	20-27-1
Сеть	№4:	20-16-1

Конфигурация сетей представлена в виде AA-BB-C, где AA – количество нейронов во входном слое, BB – количество нейронов в скрытом слое, C – количество нейронов в выходном слое.

Обучение сетей производилось в специализированных пакетах, на данный момент их великое множество (*SPSS, Statistica* и пр), ниже приведены гистограммы распределения ошибок обученных сетей на всем множестве наблюдений:

Рисунок 3. Гистограмма распределения ошибки для сети №1.

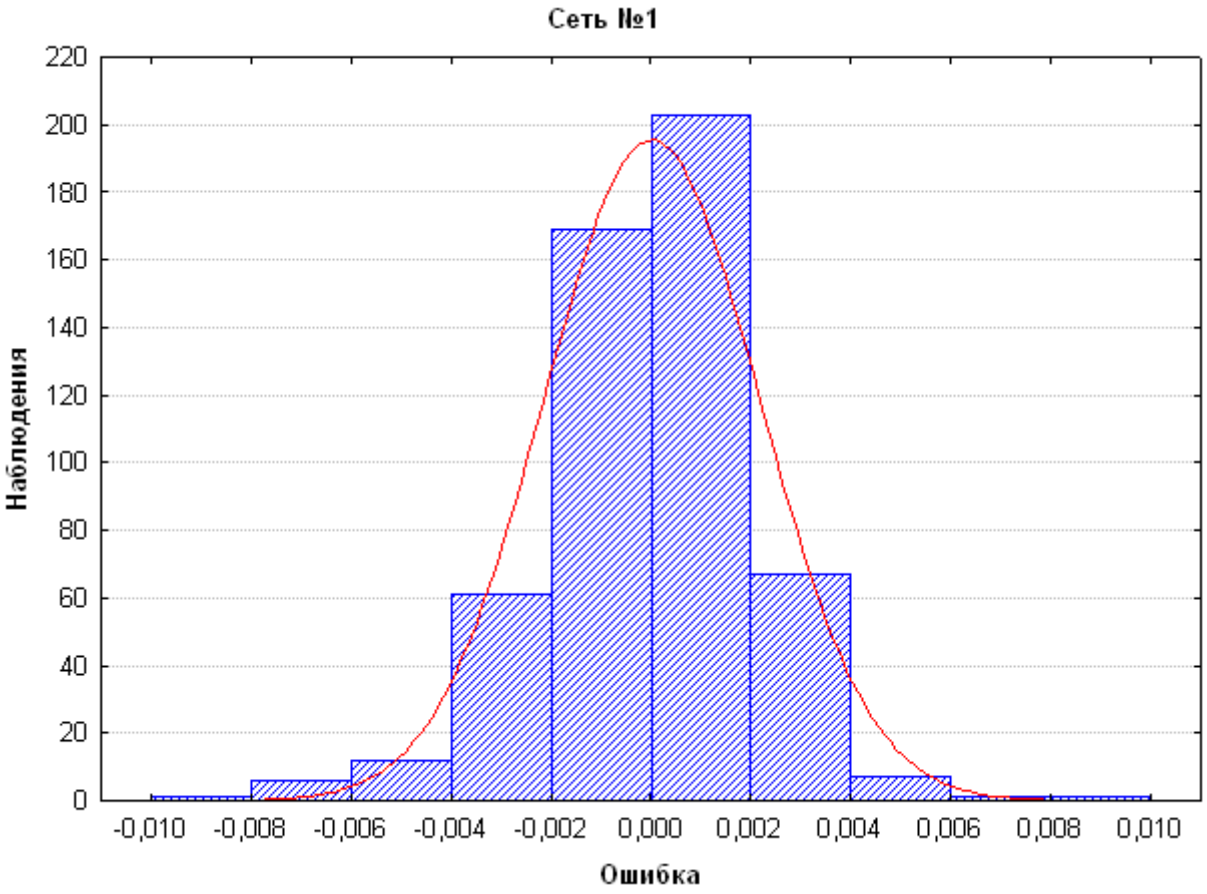
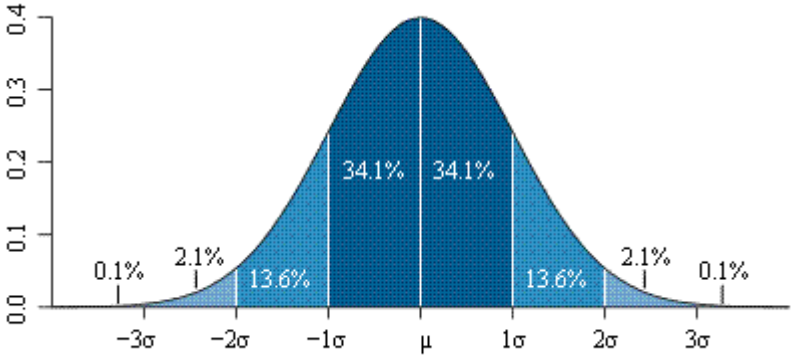


Рисунок 6 – Гистограмма распределения ошибки для сети №1.

По полученным гистограмме можно сделать вывод, что ошибка сети подчиняется нормальному закону распределения, т.е. можно разделить размер ошибки на 3 области (для упрощения распределение считается нормализованным):



±σ1 (область 1 сигма — величина ошибки в 68% процентах прогнозов находится в данном диапазоне);
±σ2 (область 2 сигма — величина ошибки в 95% процентах прогнозов находится в данном диапазоне);
±σ3 (область 3 сигма — грубые ошибки, промахи, менее чем в 5% процентах случаев, величина ошибки больше, чем в области ±σ2).

Ошибки по областям распределения:

№ сети и ±σ1 (68% прогнозов)
Сеть №1: ±16,4ppm
Сеть №2: ±18,3ppm
Сеть №3: ±19ppm
Сеть №4: ±18,6ppm

№ сети и ±σ2 (95% прогнозов)
Сеть №1: ±43,9ppm
Сеть №2: ±47,6ppm
Сеть №3: ±42,8ppm
Сеть №4: ±41ppm

Причина грубых ошибок (промахов) в области ±σ3 – это работа сети с данными сильно отличающимися от тех, которые присутствовали в обучающей выборке.
Также важным показателем качества обучения нейронной сети является величина средней абсолютной ошибки.

Размер средней абсолютной ошибки:

Сеть №1 — 14,4ppm
Сеть №2 — 13,4ppm
Сеть №3 — 14,3ppm
Сеть №4 — 13,6ppm

Ниже представлены графики зависимости содержания серы в продукте

(лабораторный анализ) и величины абсолютной ошибки:

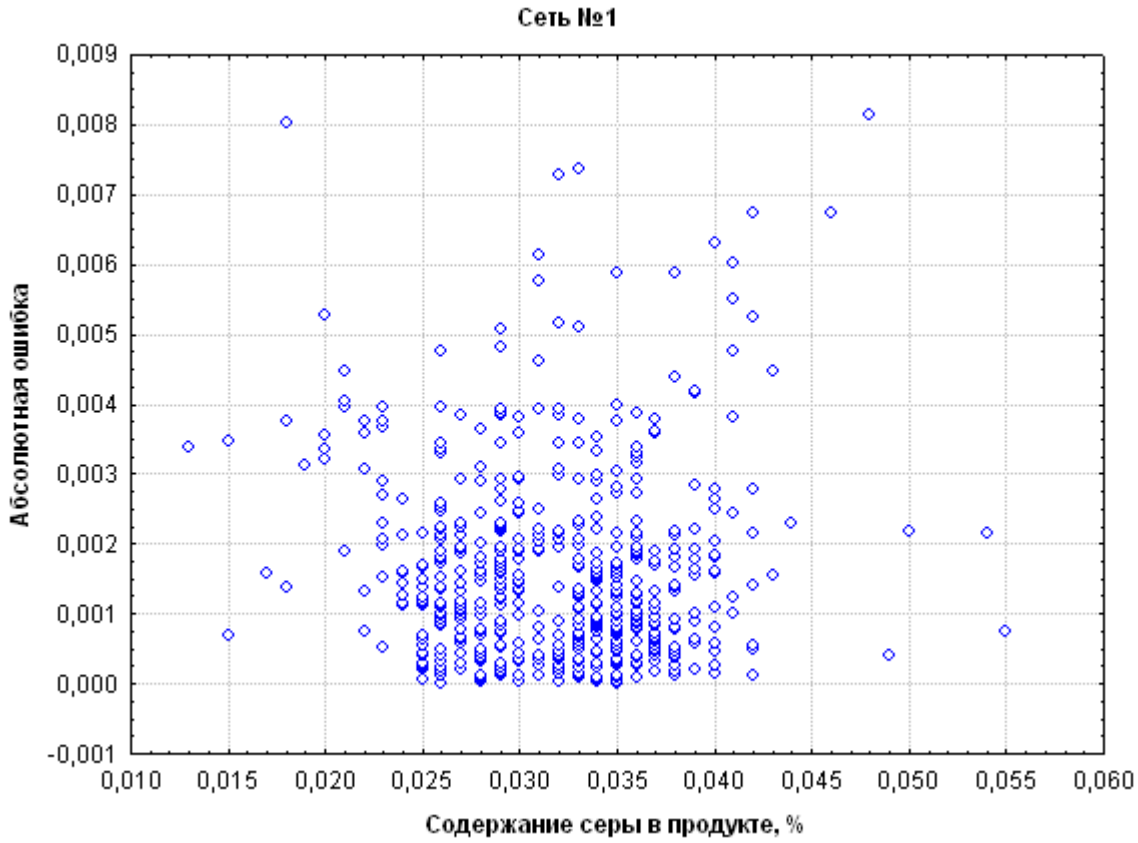


Рисунок 7. График зависимости содержания серы и абсолютной ошибки

Пути улучшения:

- **Учет времени реакции**
- В силу того что цикл реакции (от момента замера характеристик сырья и его прохождения по всей установке до дальнейшей точки замера характеристик конечного продукта) имеет определенную длительность, то для более высокой корреляции данных требуется точное сопоставления параметров сырья к параметрам продукта, что позволит увеличить точность прогноза.
- **Фильтрация шума**
- Показания, замеряемые датчиками, помимо полезной составляющей сигнала также включают в себя шум. Данный шум незначительно, но искажает процесс обучения сети и, соответственно, ее последующие прогнозы как следствие обучения, для этого требует учет шумовой составляющей с последующим добавлением фильтров перед входами нейросети. Также возможна фильтрация выхода нейросети для более плавного изменения прогноза. Спектр фильтров на сегодняшний день достаточно обширен: от простейших фильтров медианы и экспоненты до вейвлетов.
-

- **Повышение частоты анализов**
- Увеличение количества замеров содержания серы в течение дня, что позволит увеличить количество данных для обучения сети и в свою очередь позволит получить более качественную сеть.
- **Повышение точности лабораторного анализа**
- При наличии технической возможности, повышение точности анализа (при увеличении точности на порядок) позволит сделать данные более гибкими для сети, т.к. для одного и того же значения серы существует большой разброс независимых параметров, что в свою очередь влечет увеличение ошибки нейросети.
- **Увеличение количества входных переменных**
- Хочется отметить, что на самом деле даже незначительная корреляция данных с целевым параметром имеет достаточно большое значение, поэтому следует использовать максимально возможное количество параметров на объекте, а также, возможно, использовать данные с объекта, предшествующего текущему по технологической цепочке.

Создание нейронной сети, прогнозирующую вероятность заболевания *Covid*-19

На данный момент *Covid*-19 одна из самых распространённых болезней. По статистике на данный момент *Covid*-19 болеет около 231 миллиона человек на всей планете. Для предотвращения заболевания данной болезнью требуется соблюдать правила гигиены, но этого бывает недостаточно и поэтому люди заболевают. Правда болезни бывают разными, такие как обычная простуда или повышение температуры, но они могут привлечь к себе наиболее страшные болезни, такие как грипп или тот же самый *Covid*-19. Поэтому была создана нейронная сеть, которая будет выявлять прогноз вероятности заболевания этим самым *Covid*-19.

Данная нейронная сеть будет принимать несколько параметров. Параметры:

1. пол;
2. возраст;
3. наличие или отсутствие типичных для пневмонии симптомов (кашель, отдышка, слабость и др.);
4. дополнительные заболевания (сахарный диабет);
5. ширина распределения эритроцитов;
6. значение гематокрита;
7. уровень С-реактивного белка.

По данным параметрам будет создана выборка из 45 человек с разными параметрами (*parametrs.xlsx*) (см. рисунок 9) [3][4][5].

1	19	0	0	11	43	5	0
0	20	0	0	11	37	7	0
1	17	1	1	11	50	20	1
0	18	1	0	10	53	3	0
0	50	0	1	20	40	15	1
1	70	0	0	15	50	8	1
1	27	0	1	14	50	10	1
0	45	1	1	14	46	17	1
1	70	0	0	11	43	7	0
0	15	1	1	14	47	5	1
1	30	1	0	18	56	15	1
1	13	1	0	13	40	7	0
1	37	0	1	15	47	13	1
0	76	0	1	11	46	14	1
0	54	1	0	13	44	8	1

Рисунок 9 – Выборка

Далее поэтапно будет описано написание программного кода создания нейронной сети, для прогнозирования вероятности заболевания:

При помощи библиотеки *Keras* появляется возможность создания нейронной сети с минимальным количеством операций. В качестве модели нейронной сети используется последовательная *Sequenta*l из модуля *keras.models* с заданием слоев *keras.layers* типа *Dense* (см. рисунок 10).

```
from keras.models import Sequential
from keras.layers import Dense
import numpy
```

Рисунок 10 – Импортирование библиотек

Для последующей воспроизводимости результатов зафиксируем генератор случайных чисел при помощи функции *random.seed*() из библиотеки *numpy*. Считаем данные из *dataset* (см. рисунок 11).

```
numpy.random.seed(2)
dataset = numpy.loadtxt("parametrs.xlsx", delimiter=",")
```

Рисунок 11 Функция для считывания данных

Разделим данные на матрицу признаков *X* и вектор целевой переменной *Y* (последний столбец *dataset*) (см. рисунок 12).

```
X, Y = dataset[:,0:7], dataset[:,7]
```

Рисунок 12 – Матрица *X* и *Y*

Создаем модель нейронной сети(см. рисунок 13).

```
model = Sequential()
```

Рисунок 13 – Создание модели сети

Опишем структуру модели нейронной сети. Определим входной, выходной и скрытые слои. Наша нейронная сеть будет иметь плотную (*Dense*) структуру – каждый нейрон связан со всеми нейронами следующего слоя. Выходной слой будет состоять из единственного нейрона, определяющего вероятность заболевания *Covid*-19 (см. рисунок 14).

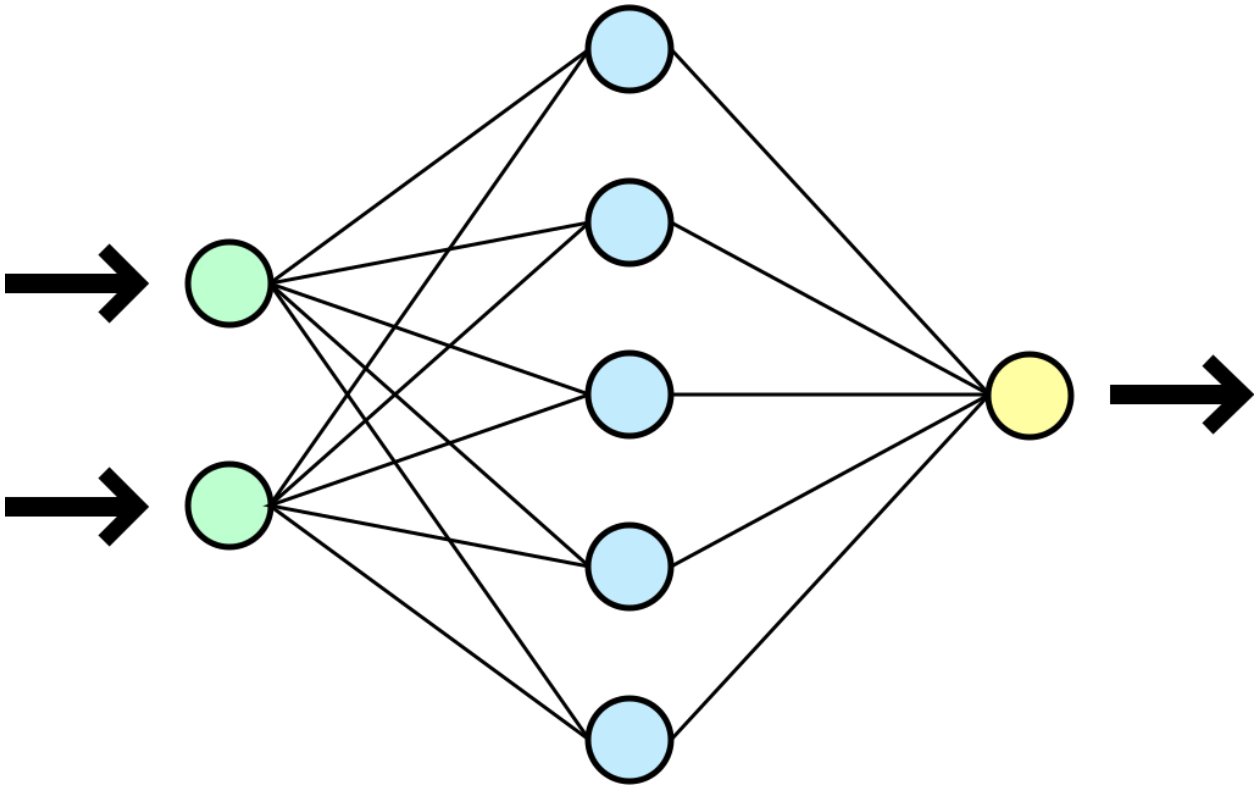


Рисунок 14 – Модель нейронной сети

Слой добавляется к модели методом *add*(). Для входного слоя необходимо указать число признаков *input_dim*, равное в данном случае 7 (см. рисунок 15).

```
model.add(Dense(12, input_dim=7, activation='relu'))
```

Рисунок 15 Назначение нейронной сети

Если наборы признаков образуют многомерную таблицу, то вместо параметра *input_dim* можно использовать параметр *input_shape*, принимающий кортеж с количеством элементов в каждом из измерений.

В качестве функции активации для всех слоев, кроме выходного, будем использовать функцию *ReLU*. Для выходного слоя воспользуемся сигмоидной функцией для определения конечной вероятности риска заболевания (см. рисунок 16).

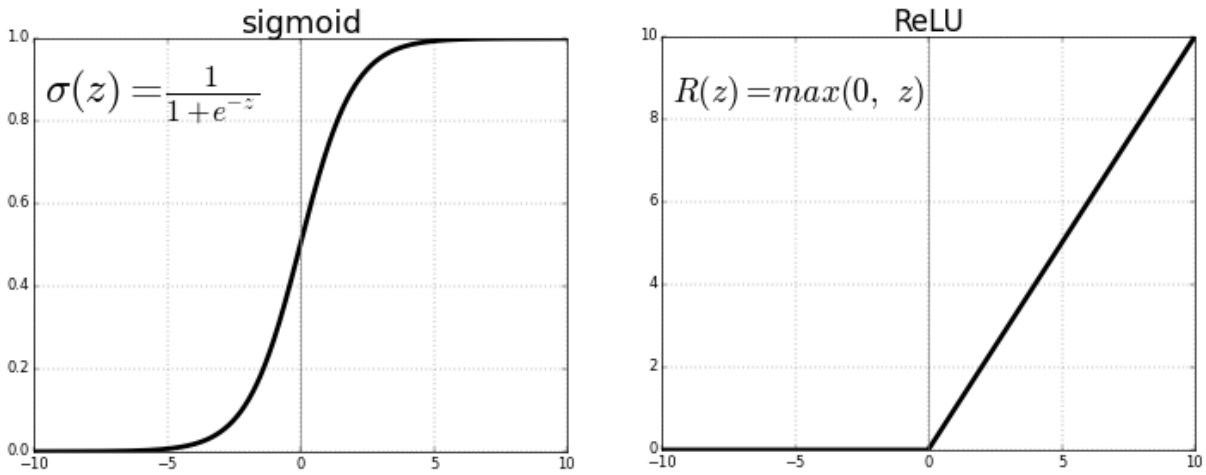


Рисунок 16 – Функции активации

Создадим три скрытых слоя и один выходной слой нашей нейронной сети (см. рисунок 17).

```
model.add(Dense(14, activation='relu'))
model.add(Dense(7, activation='relu'))
model.add(Dense(9, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
```

Рисунок 17 – Создание слоёв

Первые числа, передаваемые *Dense*, это количества нейронов, экспериментально оптимизированные в результате вариации структуры нейронной сети. Можно изменять количество скрытых слоев и содержащихся в них нейронов, чтобы добиться лучшего качества предсказательности модели.

Перед тем, как начать тренировать модель, ее нужно скомпилировать при помощи метода *compile*() (см. рисунок 18).

```
model.compile(loss="binary_crossentropy", optimizer="adam", metrics=['accuracy'])
```

Рисунок 18 – Компиляция кода

Методу передается три параметра (см. рисунок 17).

loss – функция потерь – объект, который модель стремиться минимизировать;

optimizer – оптимизатор, мы используем встроенный метод стохастической оптимизации *adam*, описанный в публикации Дедерика Кингма и Джимми Ба;

metrics – список метрик оптимизации, для задач классификации используется метрику *'accuracy'*.

Для обучения нейронной сети применяем метод *fit*() (см. рисунок 19).

```
model.fit(X, Y, epochs = 1000, batch_size=10)
```

Рисунок 19 – Метод

Параметр *epochs* – "эпохи" – количество проходов нейронной сети по всем записям *dataset* (выбирается исходя из того, насколько быстро модель с каждым новым проходом приближается к желаемой предсказательной точности), *batch_size* – количество объектов выборки, берущихся за один шаг. В процессе обучения API будет выводить соответствующие строчки с величинам функции потерь и метрики для каждой из эпох.

Оценим результат обучения нейронной сети. Метод *evaluate*() возвращает значения функции потерь и метрики для обученной модели (см. рисунок 20).

```
scores = model.evaluate(X, Y)
print("\n%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))
```

Рисунок 20 – Вывод результата обученной сети

Последняя строчка в форматированном виде выводит точность прогноза по нашей модели для заданной метрики *accuracy* (см. рисунок 21).

```
acc: 87.89%
```

Рисунок 21 – Полученный результат

ЗАКЛЮЕНИЯ И ВЫВОДЫ

В ходе выполнения данного проекта были исследованы принципы и методы прогнозирования, так как они являются важнейшей частью, для данного проекта, выявления вероятности заболевания. Также были рассмотрены принципы параметров заболевания и их важность.

Затем основываясь на понятии прогнозирования были исследованы основные технологии разработки нейронных сетей на основе диагностики. Были выявлены основные виды разработок программного обеспечения исходя из особенностей сред разработки.

Выполнение задание было реализовано на платформе *Python*. При помощи данной платформы была создана нейронная сеть, способная диагностировать и прогнозировать вероятность заболевания, по параметрам данные ему.

ЛИТЕРАТУРА

[1] Введение в прогнозирование [Электронный ресурс] – Режим доступа: https://habr.com/ru/post/177633/

[2] Система прогнозирование на базе нейронных сетей [Электронный ресурс] – Режим доступа: https://habr.com/ru/post/171019/

[3] Ширина распространения эритроцитов [Электронный ресурс] – Режим доступа: celt.ru/depart/bio_lab/uslugi/shirina-raspredeleniya-ehritrocitov-rdw/

[4] Гематокрит [Электронный ресурс] – Режим доступа: https://citilab.ru/articles/gematokrit-norma-po-vozrastu-prichiny-povyshennykh-i-ponizhennykh-znacheniy/

[5] С–реактивный белок [Электронный ресурс] – Режим доступа: https://helix.ru/kb/item/06-182

ПРИЛОЖЕНИЕ А
(обязательное)
Листинг программного кода

```
from keras.models import Sequential

from keras.layers import Dense

import numpy


numpy.random.seed(2)



dataset = numpy.loadtxt("parametrs.xlsx", delimiter=",")


X, Y = dataset[:,0:7], dataset[:,7]


model = Sequential()

model.add(Dense(12, input_dim=7, activation='relu'))

model.add(Dense(15, activation='relu'))

model.add(Dense(8, activation='relu'))

model.add(Dense(10, activation='relu'))

model.add(Dense(1, activation='sigmoid'))


model.compile(loss="binary_crossentropy", optimizer="adam", metrics=['accuracy'])


model.fit(X, Y, epochs = 1000, batch_size=10)


scores = model.evaluate(X, Y)

print("\n%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))
```