Содержание

Введение

На протяжении всего времени существования человечества проблема возникновения, исследования и лечения различных заболеваний у человека является важной задачей медицинской деятельности. Особенно это касается онкологических заболеваний. На сегодняшний день нет четкой причины, по которой люди заболевают раком, но существует множество способов ранней диагностики таких заболеваний [2].

В данной работе рассматривается использование результатов моделирования, соответствующих методике микроволновой радиотермометрии молочных желез на основе работы специального диагностического комплекса РТМ-01-РЭС. Также рассматриваются популярные алгоритмы классификации данных и библиотеки для языка программирования Python, реализующие данные алгоритмы.

Главной целью работы является проведение исследования влияния размеров опухоли на точность диагностики раковых заболеваний на основе данных микроволновой радиотермометрии, полученных в ходе компьютерного моделирования. Так как проведение классификации только на температурных данных может не дать хорошего результата, необходимо выявить какой из алгоритмов будет лучше работать с разными размерами опухоли и вариациями остальных параметров.

1 Алгоритмы классификации данных

Классификация данных состоит из прогнозирования определенного результата на основе уже известных данных. Чтобы предсказать результат, алгоритм обрабатывает данные, содержащие набор атрибутов и соответствующий каждому набору результат, обычно называемый атрибутом прогнозирования цели или классом. Формируется модель алгоритма, которая пытается обнаружить отношения между атрибутами, которые позволили бы предсказать результат. Такая процедура называется обучением модели, а набор данных, используемый для этого – тестовой выборкой.

Следующим шагом после обучения модели является прогнозирование – процедура определения класса, при которой используется набор данных с неизвестными классами. Такой набор данных, который содержит тот же набор атрибутов, за исключением атрибута прогнозирования, часто называют тестовой выборкой.

Алгоритм анализирует входные данные и выдает прогноз. Точность прогноза определяет, насколько «хорош» алгоритм. Например, в медицинской базе данных обучающий набор должен иметь соответствующую информацию о пациенте, записанную ранее, где атрибутом прогноза является наличие или отсутствие у пациента проблем со здоровьем.

Для определения того, какой именно алгоритм использовать для конкретной задачи можно воспользоваться схемой, изображенной на рисунке 1. Исходя из того, что в текущей задаче используется не тестовая информация и имеется 160 примеров, то были выбраны алгоритмы SVM, k-ближайших соседей и наивный байесовский классификатор, описанные ниже.

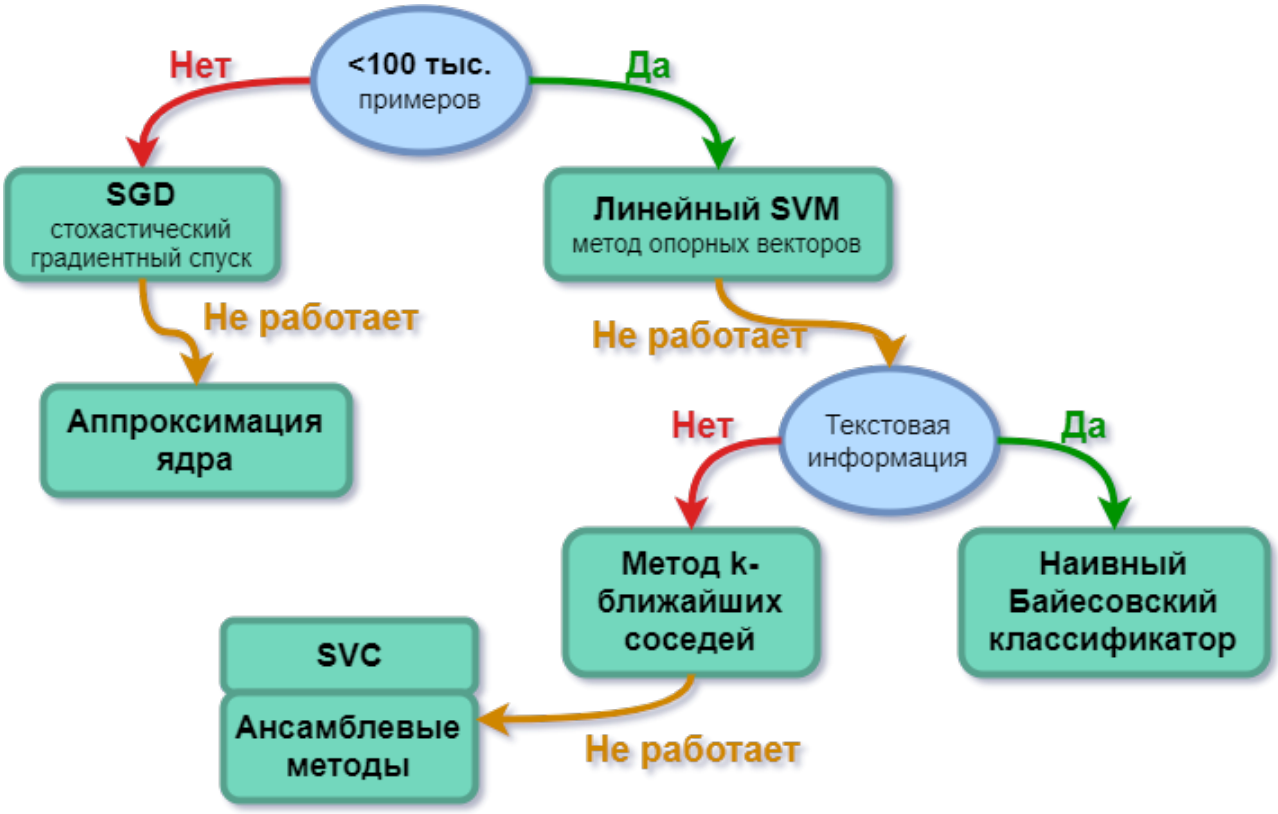

Рисунок 1 – Схема для определения алгоритма классификации для конкретной задачи

1.1 Метод опорных векторов

Метод опорных вектором или SVM – это метод статистической классификации. Он широко используется для задач различного рода и хорошо себя в них показывает.

Основной идеей метода является представление атрибутов данных в виде векторов и переход в пространство более высокой размерности, чем получившееся на этапе представления векторами. Затем ищется гиперплоскость с максимальным зазором в пространстве между объектами разных классов.

На рисунке 2 показан пример классификации методом SVM. Красной линией выделена как раз та самая гиперплоскость, четко разделяющая объекты разных классов друг от друга.

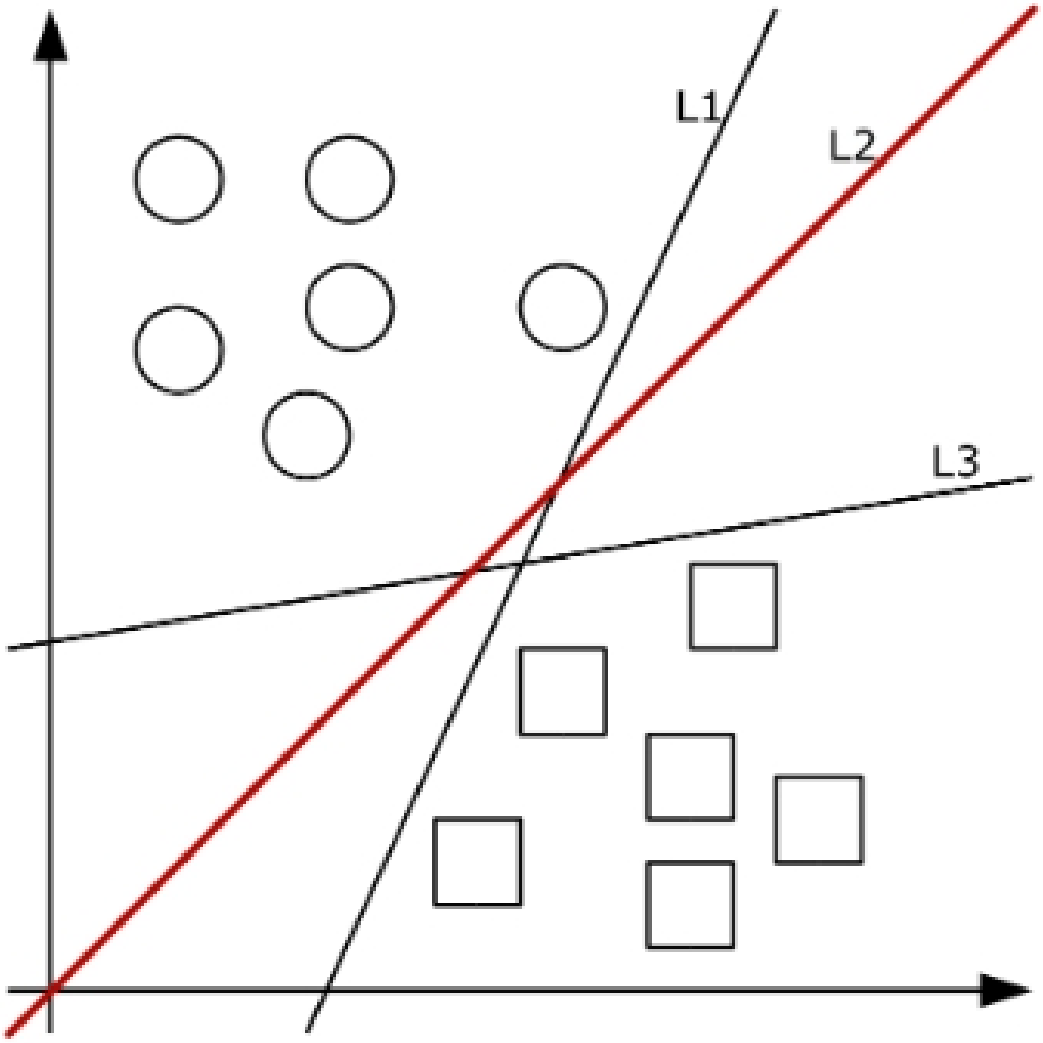

Рисунок 2 – Пример классификации методом SVM

Алгоритм может использоваться с одним из следующих видов ядер:

- Полиномиальное (однородное) $k(x, x') = (x \cdot x')^d$;
- Полиномиальное (неоднородное) $k(x, x') = (x \cdot x' + 1)^d$;
- Радиальная базисная функция $k(x, x') = exp(-\gamma||x - x'||^2)$, для $\gamma > 0$;
- Радиальная базисная функция Гаусса $k(x, x') = exp\left(-\frac{||x-x'||^2}{2\sigma^2}\right)$;
- Сигмоид $k(x, x') = tanh(kx \cdot x' + c)$.

1.2 Метод k-ближайших соседей

Алгоритм k-ближайших соседей является простым статистическим алгоритмом обучения, в котором объект классифицируется своими соседями. При классификации таким методом объект относится к классу, наиболее распространенному среди его k-ближайших соседей. Пример классификации приведен на рисунке 3, где в качестве классифицируемого объекта используется прямоугольник и существует несколько объектов известных классов – белые точки и черные. Замерив расстояние от объекта до его соседей с различными классами и основываясь на методе k-ближайших соседей данный объект будет отнесен к классу черный точек, а не белых.

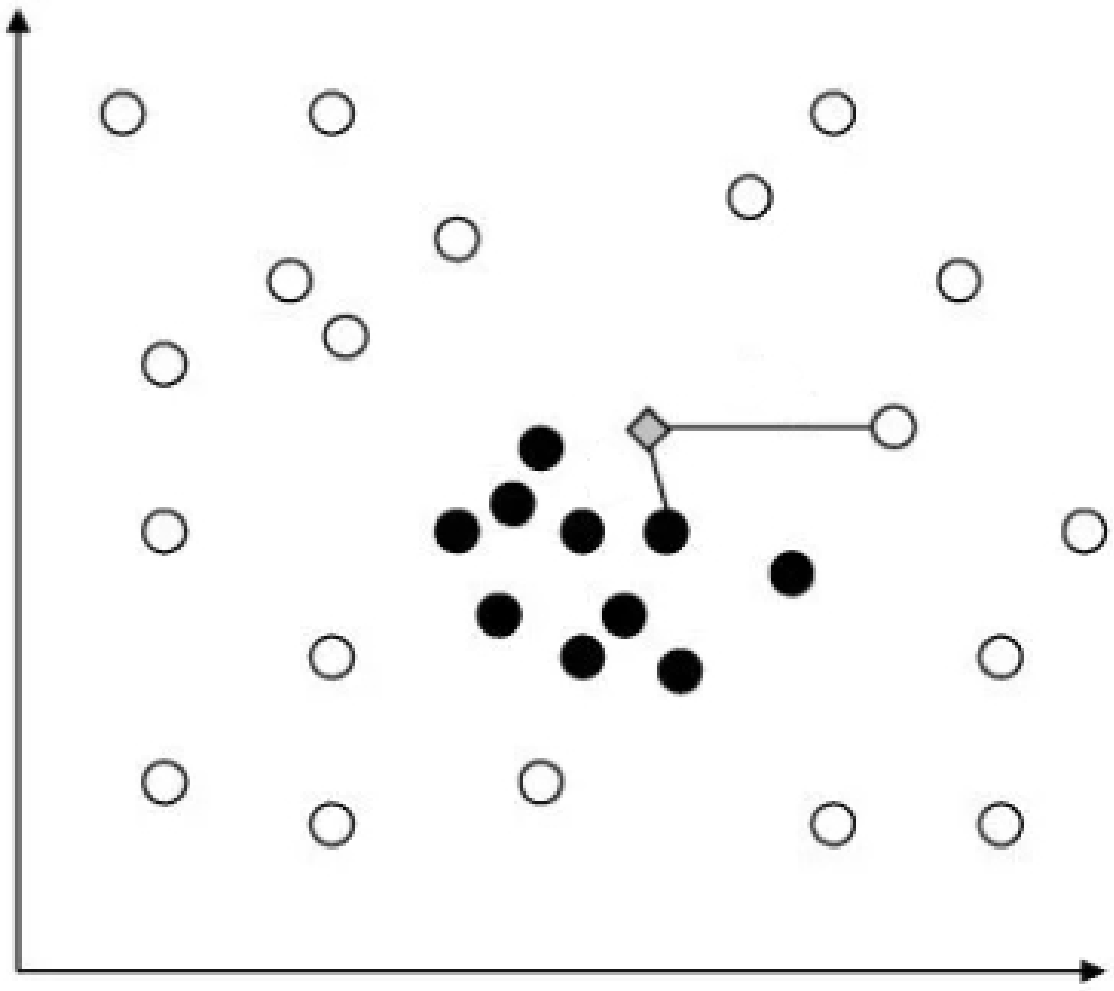

Рисунок 3 – Пример классификации методом k-ближайших соседей

При нахождении атрибутов учитывается значимость атрибутов и часто применяется прием растяжения осей, демонстрируемый в формуле (1). Использование данного приема снижает ошибку классификации.

$$D_E = \sqrt{3(x_A - y_A)^2 + (x_B - y_B)^2}, \tag{1}$$

где x_A, y_A – значения атрибута A в наборе данных, x_B, y_B – значения атрибута B.

Данный алгоритм возможно применять как для данных с маленьким количеством атрибутов, так и с достаточно большим. Важным моментом при работе с алгоритом является определение функции расстояния между значениями. Примером такой функции может быть евклидово расстояние – формула (2).

$$D_E = \sqrt{\sum_{i}^{n}(x_i - y_i)^2}, \tag{2}$$

где x_i, y_i – значения атрибутов в наборе данных.

1.3 Наивный байесовский классификатор

Наивный байесовский классификатор является простым вероятностным классификатором и основывается на применении теоремы Байеса со строгими предположениями о независимости. Хотя наивный байесовский классификатор редко применим к большинству реальных задач, но зачастую в определенных задачах он демонстрирует хорошие результаты и часто конкурирует с более сложными методами, такими как SVM и классификационным деревьями. Классификация данным методом очень зависит от распределения зависимостей атрибутов, а не от самих зависимостей.

Вероятностная модель классификатора:

$$p(C|F_1, ..., F_n) = \frac{p(C)p(F_1, ..., F_n|C)}{p(F_1, ..., F_n)}, \tag{3}$$

где C – класс модели, а F_i – классифицируемые модели.

Использование формулы (3) при классификации дает минимально значение среднего риска или математичского ожидания ошибки:

$$R(a) = \sum_{y \in Y} \sum_{\varsigma \in Y} \lambda_y P_y P_{x,y} \{a(x) = \varsigma | y\}, \tag{4}$$

где λ_y – цена ошибки при отнесении объекта класса Y к какому-либо другому классу.

2 Разработка программы для классификации с использованием библиотеки Scikit-learn

2.1 Обзор существующих библиотек для машинного обучения

На текущий момент существует множество готовых реализаций алгоритмов машинного обучения и не имеет смысла делать то же самое с нуля, если задача не имеет каких-то особенностей, делающих невозможным использование готовых библиотек. Каждая из библиотек, рассматриваемых в работе, хороша в своей области, успешно используется в решении задач и проверена временем. Рассмотрим некоторые из популярных библиотек для языка программирования Python.

2.1.1 Scikit-learn

Scikit-learn – это одна из самых популярных библиотек для языка Python, в которой реализованы основные алгоритмы машинного обучения, такие как классификация различных типов, регрессия и кластеризация данных. Библиотека распространяется свободно и является бесплатной для использования в своих проектах.

Данная библиотека создана на основе двух других – NumPy и SciPy, имеющих большое количество готовых реализаций часто используемых математических и статистических функций. Библиотека хорошо подходит для простых и средней сложности задач, а также для людей, которые только начинают свой путь в изучении машинного обучения.

2.1.2 Theano

Theano – это библиотека, в которой содержится базовый набор инструментов для машинного обучения и конфигурирования нейросетей. Так же у данной библиотеки есть встроенные методы для эффективного вычисления математических выражений, содержащих многомерные массивы.

Theano тесно интегрирована с библиотекой NumPy, что дает возможность просто и быстро производить вычисления. Главным преимуществом библиотеки является возможность использования GPU без изменения кода программы, что дает преимущество при выполнении ресуркоемких задач. Также возможно использование динамической генерации кода на языке программирования C.

2.1.3 TensorFlow

Самой популярной и масштабной по применению является библиотека TensorFlow, используемая для глубокого машинного обучения. Библиотека разрабатывается в тесном сотрудничестве с компанией Google и применяется в большинстве их проектов где используется машинное обучение. Библиотека использует систему многоуровневых узлов, которая позволяет вам быстро настраивать, обучать и развертывать искусственные нейронные сети с большими наборами данных.

Библиотека хорошо подходит для широкого семейства техник машинного обучения, а не только для глубокого машинного обучения. Программы с использованием TensorFlow можно компилировать и запускать как на CPU, так и на GPU. Также данная библиотека имеет обширный встроенный функционал логирования, собственный интерактивный визуализатор данных и логов.

2.2 Алгоритм работы и структура программы

Целью разработки программы была классификация температурных данных с разными разамерами опухоли и графическое представление получившихся результатов. Блок-схема получившейся программы представлена на рисунке 4.

На первом этапе температурные данные считываются из CSV-файла с помощью библиотеки Pandas. Название папки с файлом моделей определенного размера опухоли хранится в отдельной переменной. Затем производится создание и обучение моделей классификации методов SVM, k-ближайших соседей и наивного байесовского классификатора на данных обучающей выборки.

Следующим этам после обучение идет прогнозирование или классификация с использованием данных тестовой выборки. После получения результирующих классов из модели необходимо сравнить их с теми, которые уже известны для определения точности работы алгоритма. После обработки результатов классификации получившиеся данные отображаются на круговых диаграммах отдельно для каждого алгоритма.

Рисунок 4 – Блок-схема программы для классификации температурных данных

2.3 Визуализация данных

Для визуализации данных моделей и результатов классификации была выбрана библиотека для языка Python Matplotlib. Данная библиотека позволяет строить графики, гистограммы, круговые диаграммы, карты распределения и 3D-объекты (рисунок 5). Возможна детальная настройка цветов линий, надписей, расположения элементов и подписей к графикам, осям и значениям.

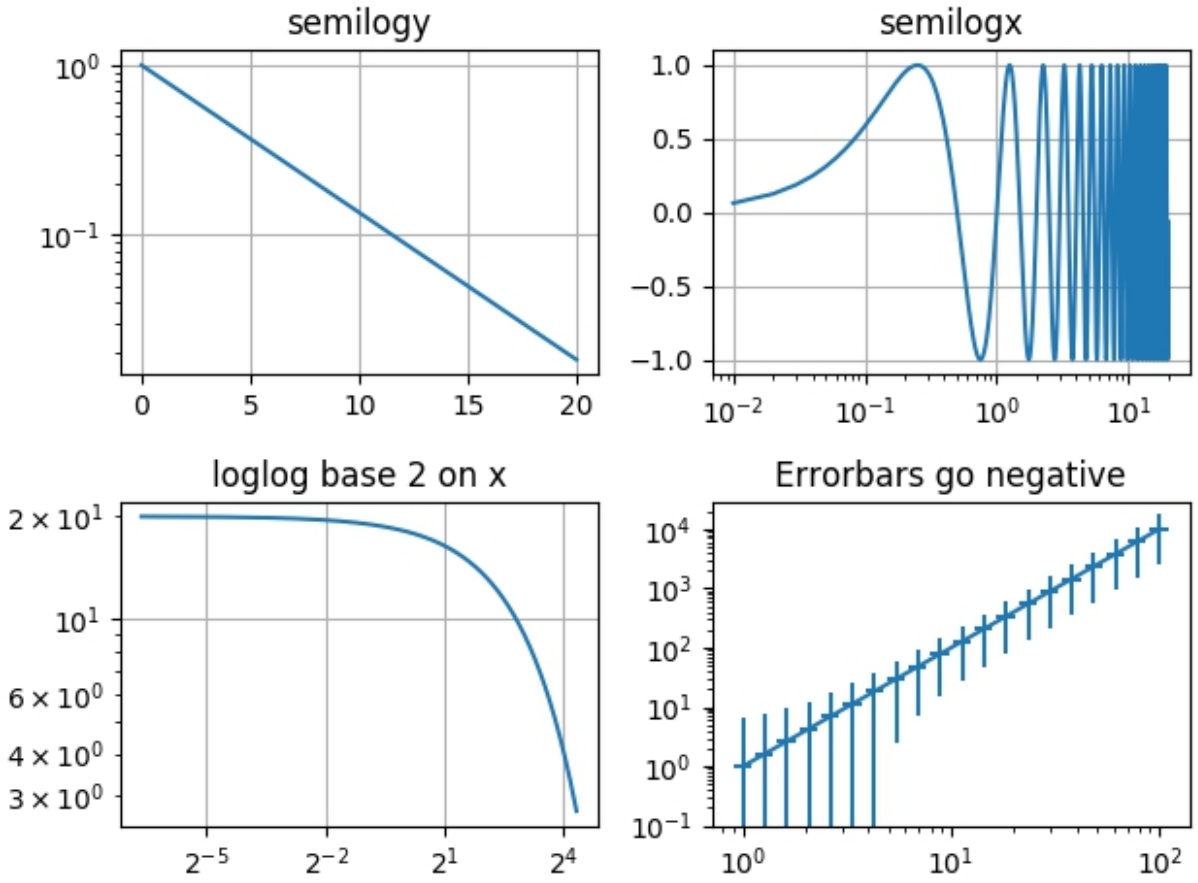

Рисунок 5 – Примеры графиков, построенных с помощью библиотеки Matplotlib

Библиотека содержит множество разных способов построения и отображения объектов, имея многослойную структуру. За счет этого имеется возможность накладывать графики друг на друга. Графики, построенные через Matplotlib отображаются в отдельном окне при запуске программы из тер-

минала, при использовани Jupter Notebook отображаются прямо в тексте. Также имеется возможность встроить графику в программу с графическим интерфейсом на Python, реализованную с помощью библиотеки Tkinter.

3 Классификация температурных данных компьютерного моделирования

3.1 Данные для классификации

В работе использовались данные компьютерного моделирования яркостной температуры молочных желез больных и здоровых пациентов. Данные были представлены в виде девяти значений температуры на поверхности кожи и девяти значений внутренней температуры, согласно методике обследования методом радиотермометрии. Схема расположения точек при замере температур представлена на рисунке 6. Отдельным атрибутом являлся класс модели. Для здоровых моделей значения класса было равно нулю, а для больных – единице. Исходя из количества классов, классификацию в данной работе можно считать бинарной.

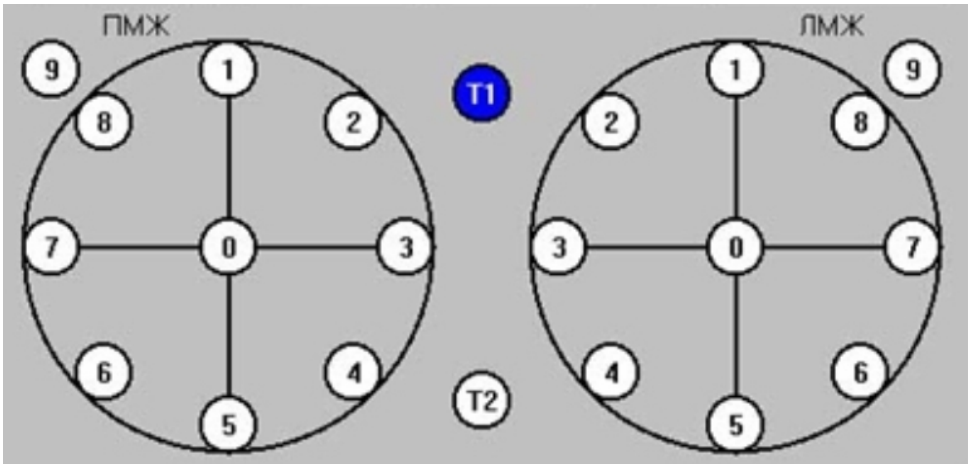

Рисунок 6 – Пример данных температурных данных компьютерного моделирования, где в столбце «target» здоровые — «0», больные — «1»

Для исследования были взяты температурные данные моделей с радиусом опухоли 0.5 см и 0.75 см. Данные были представлены в виде двух CSV-файлов, в которых находилось по 160 моделей для каждого размера опухоли соответственно (рисунок 7). Одна половина моделей состояла из здоровых пациентов, а другая из больных. На подготовительном этапе данные были разбиты на обучающую и тестовую выборки в соотношении один к четырем.

0ртм	1ртм	2ртм	3ртм	4ртм	5ртм	6ртм	7ртм	8ртм	0ик	1ик	2ик	3ик	4ик	5ик	6ик	7ик	8ик	target
33.5	33.1	33.7	32.6	32.7	33.2	33.4	33.2	33.3	29.8	29.9	30.2	30.0	30.0	31.1	29.4	29.6	29.8	0
33.7	32.8	33.3	33.2	33.5	33.5	33.2	32.9	32.4	32.9	31.9	31.8	31.4	31.5	31.0	32.1	31.1	31.4	0
33.3	32.4	32.5	34.2	33.7	33.4	32.8	32.1	32.2	31.6	31.0	30.8	32.8	31.9	31.6	31.2	30.8	30.7	1
34.2	33.2	33.9	34.0	34.3	33.5	33.2	33.5	33.2	32.6	32.3	32.6	32.1	33.0	32.3	31.6	31.0	32.3	1
35.7	34.5	34.6	34.7	33.7	35.0	34.8	35.2	34.7	35.4	33.9	33.9	34.0	34.8	34.4	34.4	34.9	33.9	0
33.4	34.0	33.8	33.9	33.5	33.9	33.1	33.1	33.5	32.1	32.3	32.3	32.4	31.8	31.7	31.3	31.2	32.2	0
35.5	34.6	33.6	34.4	34.7	34.2	33.6	33.6	33.5	33.5	33.2	32.7	32.6	32.8	32.2	31.3	30.9	32.2	1
34.8	34.0	34.1	34.4	34.3	33.8	33.9	33.9	34.2	32.1	31.5	31.5	32.0	31.4	31.7	31.0	30.6	31.2	0
35.7	34.4	34.4	34.2	34.7	34.2	34.1	35.1	35.6	34.4	33.6	32.9	33.8	33.3	33.0	33.3	34.5	34.6	1
33.9	33.0	33.1	33.3	32.8	33.5	33.0	32.5	32.7	31.9	31.2	31.9	31.6	31.9	31.8	32.4	30.3	30.9	0

Рисунок 7 – Пример данных температурных данных компьютерного моделирования, где в столбце «target» здоровые — «0», больные — «1»

3.2 Частотный анализ классифицируемых данных

Перед классификацией температурных данных был проведен частотный анализ для исследования распределения температуры в зависимости от того, в какой точке она была замерена и есть ли у данной модели опухоль. Частотный анализ был произведен с помощью библиотеки Pandas, а графическое отображение результатов с помощью Matplotlib. Точки, которые были проанализрованы соответствуют точкам на схеме измерений согласно методу РТМ (рисунок 6).

Сначала для исследования была взяла точка 0ртм. Результаты исследования представлены в виде диаграммы частот (рисунок 8), где верхняя диаграмма – это здоровые пациенты, а нижняя – больные.

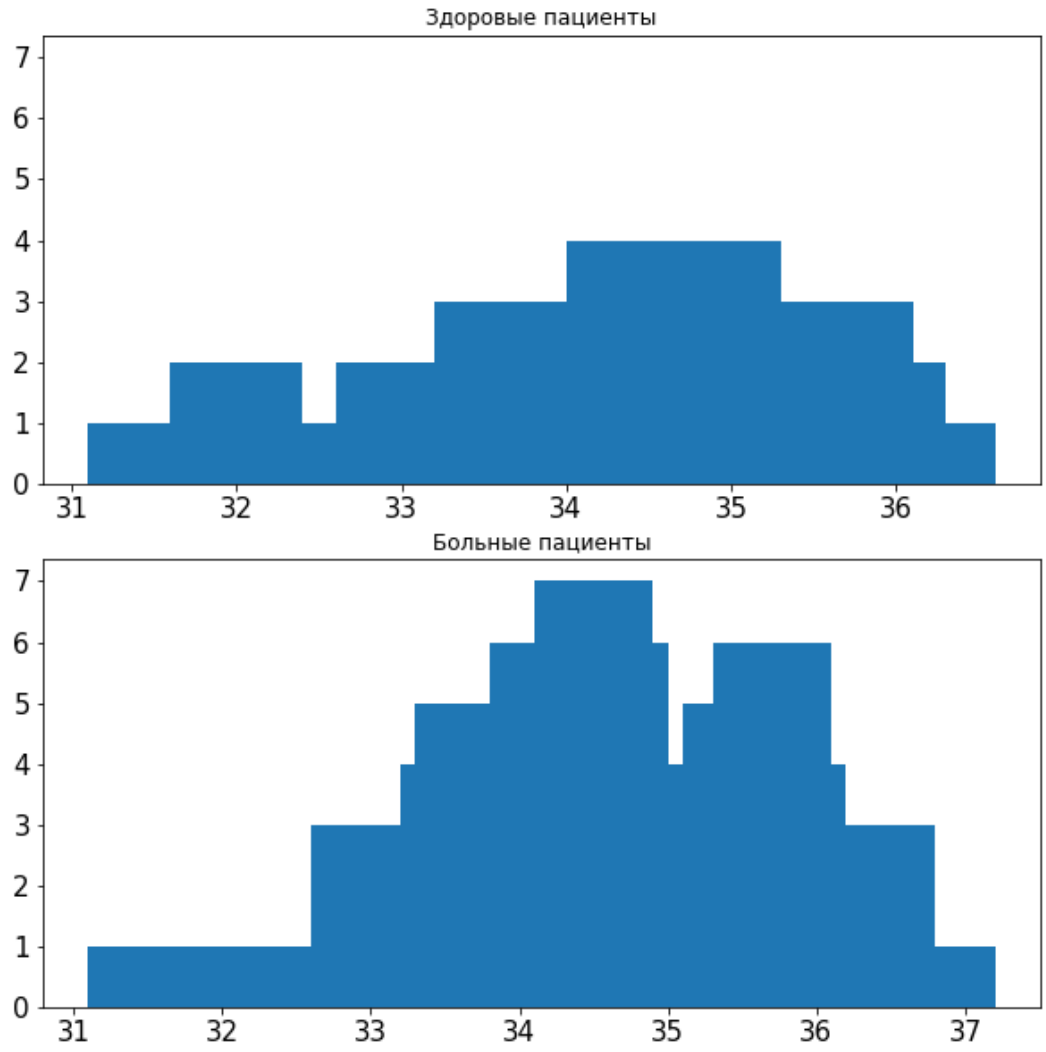

Рисунок 8 – Диаграмма частот температур в точке 0ртм. Верхняя диаграмма — здоровые пациенты, нижняя — больные

Изучив диаграммы, можно заметить, что в данной точке у пациентов чаще встречаются более высокие температуры, нежели у здоровых. Точка 0ртм больше всего подверждена воздействию от опухоли в температурном смысле, так как находится ровно в центре молочной железы.

Следующей для исследования была рассмотрена точка 3ртм, диаграмма частот температур которой представлена на рисунке 9. Для данной точки можно наблюдать схожую картину с точкой 0ртм – тут так же больше значе-

ний со средней температурой у больных пациентов, несмотря на то, что эта точка не является центральной.

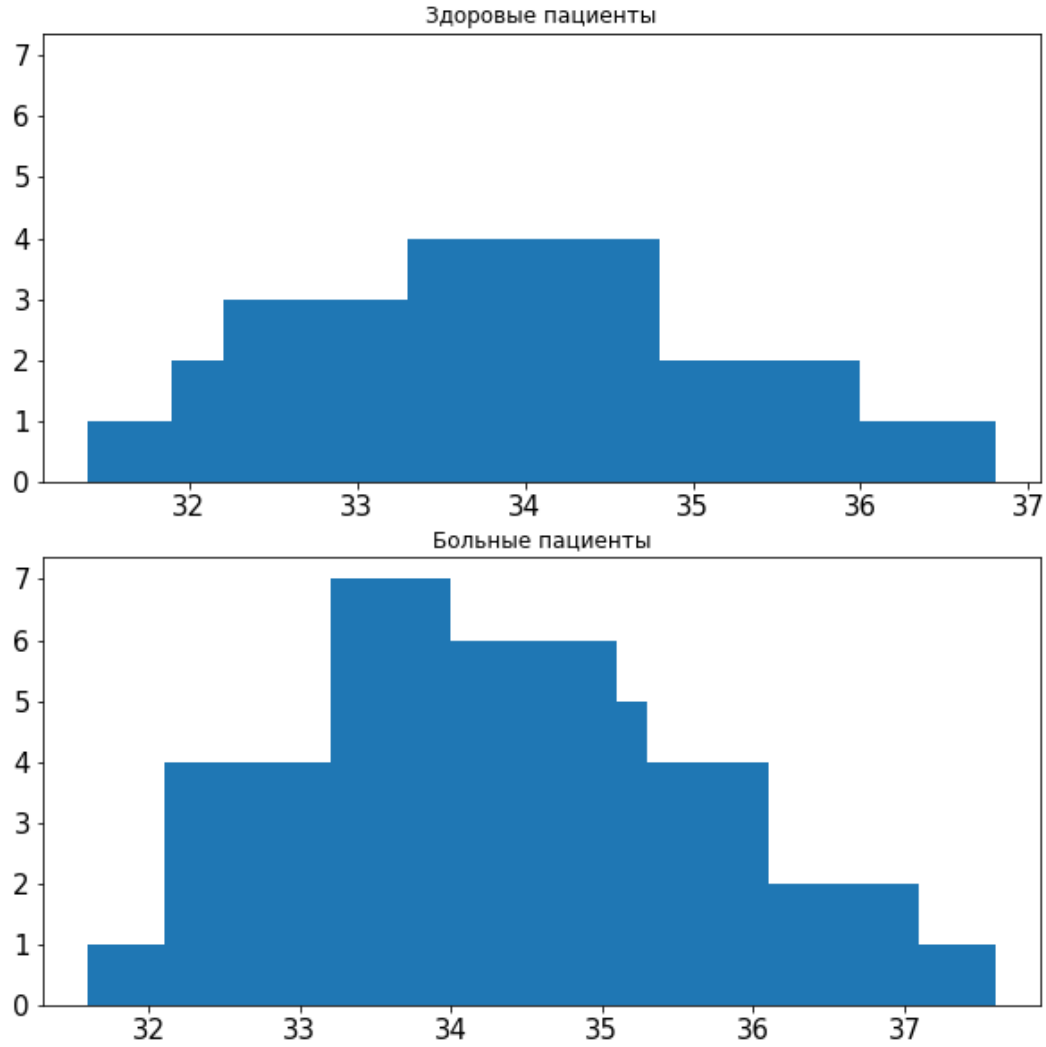

Рисунок 9 – Диаграмма частот температур в точке 3ртм. Верхняя диаграмма — здоровые пациенты, нижняя — больные

Также была рассмотрена точка 7ртм, для которой как видно из диаграммы частот температур (рисунок 10) ситуация аналогична с точками 0ртм и 3ртм.

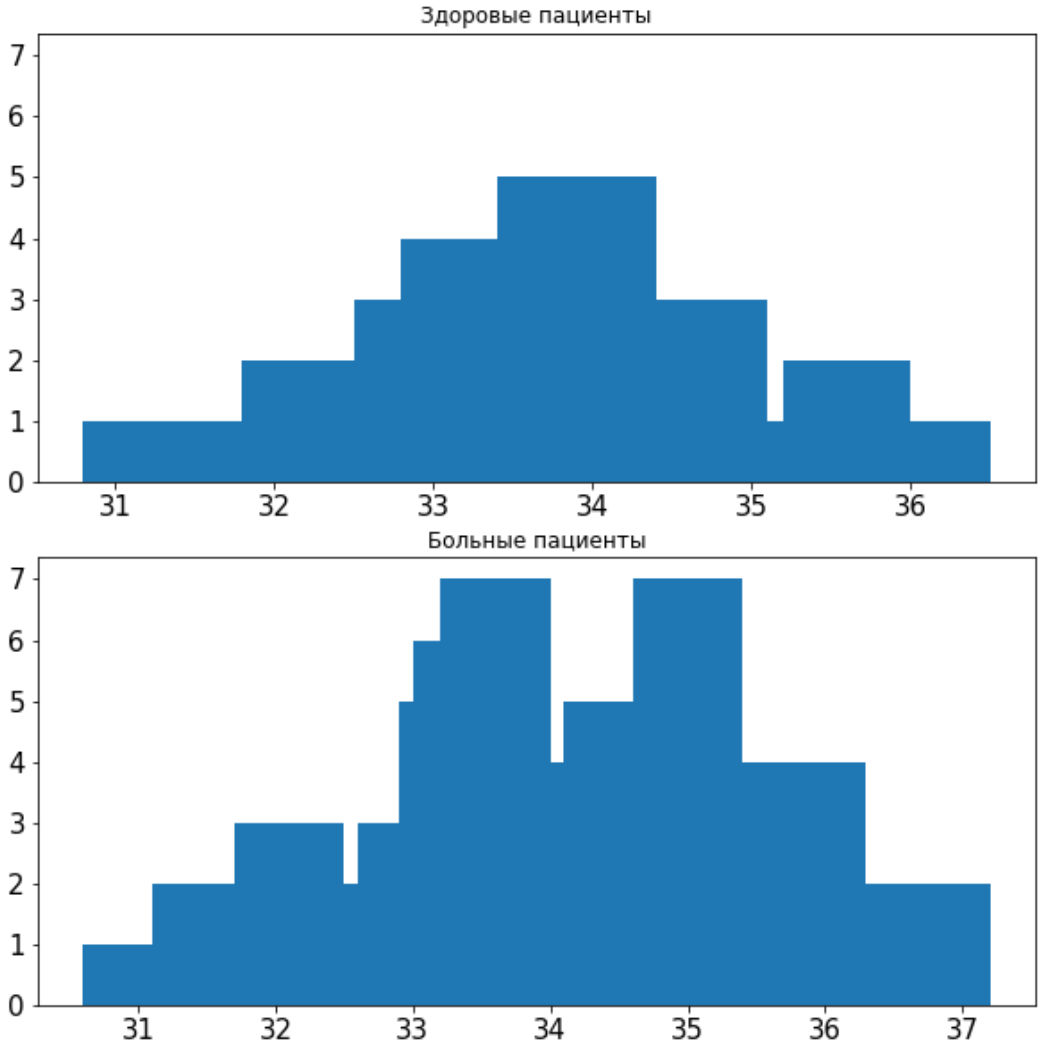

Рисунок 10 – Диаграмма частот температур в точке 7ртм. Верхняя диаграмма — здоровые пациенты, нижняя — больные

3.3 Результаты классификации для моделей с опухолью радиусом 0.5 см

Для моделей с разными размерами опухоли отдельно была посчитана точность определения класса и построены круговые диаграммы для наглядности, где как «1» отмечены верно классифицированные модели, а как «0» –

неверно классифицированные.

Как видно из диаграммы на рисунке 11, для моделей с радиусом опухоли 0.5 см лучший результат показал метод SVM, точность определения классов которого равна 57.5%. Хуже всего для данных моделей показал себя наивный байесовский классификатор с точностью классификации 27.5%

Рисунок 11 – Диаграммы с точностью определения класса для трех методов классификации моделей с размером опухоли R=0.5 см («1» – классифицировано верно, «0» – классифицировано неверно)

3.4 Результаты классификации для моделей с опухолью радиусом 0.75 см

Рассмотрим результаты классификации для моделей с радиусом опухоли 0.75 см (рисунок 12). Как видно из диаграмм, для данных моделей наивный байесовский классификатор сработал лучше, для для моделей с радиусом опухоли 0.5 см, показав результат точности равный 70%. Точность определения классов методом SVM опять оказалась лучше, чем у остальных – 72.5

Исходя из этого, можно сделать вывод, что точность классификации сильно зависит от размеров опухоли и выбранного алгоритма классификации.

Рисунок 12 – Диаграммы с точностью определения класса для трех методов классификации моделей с размером опухоли R=0.75 см («1» – классифицировано верно, «0» – классифицировано неверно)

3.5 Результаты классификации для всех моделей вместе

В качестве заключительного эксперимента было принято решение смешать вместе данные моделей с радиусом 0.5 см и 0.75 см. В качестве обучающей выборки использовались 180 моделей, а в качестве тестовой – 60 моделей. После чтения из файла данные были перемешаны между собой.

Результаты представлены на диаграмме (рисунок 12). В этот раз лучшим по точности определения класса стал наивный байесовский классификатор с точностью 70%, которая не изменилась по сравнению с предыдущим экспериментом. А метод SVM, показавший лучший результат в предыдущих двух экспериментах, в этот раз определил классы моделей с точностью 66.7%, такой же как и метод k-ближайших соседей.

Рисунок 13 – Диаграммы с точностью определения класса для трех методов классификации моделей с размером опухоли R=0.5 см и R=0.75 см вместе («1» – классифицировано верно, «0» – классифицировано неверно)

Заключение

Текст

Список литературы

1. Bardati, F. Modeling the Visibility of Breast Malignancy by a MicrowaveRadiometer [Текст] / F. Bardati, S. Iudicello. — Biomed. Engineering. – 2008. – Vol.55 (6). – С. 214-221.
2. Fear, K.E. Microwave detection of breast cancer [Текст] / K.E. Fear, M. Ctuchly. – IEEE Trans. Microwave Theory Tech. – 2000. – Vol.48 (11). – С. 1854-1863.
3. Pryor, R.W. Multiphysics Modeling Using COMSOL: A First Principles Approach [Текст] / R.W. Pryor – Jones & Barlett Publishers, Inc. – 2011. – 872 с.
4. Roger, W. Multiphysics Modeling USING COMSOL [Текст] / Roger W. Pryor. – LLC: Jones and Bartlett Publishers. – 2011. – 871 с.
5. Sherwood, L. Fundamentals of Human Physiolog [Текст] / L. Sherwood. — Belmon: Brooks/Cole – 2012. – 720 с.
6. Бирюлин, Г.В. Теплофизический расчет в конечно-элементном пакете COMSOL/FEMLAB [Текст] / Г.В. Бирюлин. – Методическое пособие. – СПб: СПбГУИТМО – 2006. – 75 с.

Приложение А
(обязательное)
Программный код

Листинг А.1 – Код программы для классификации данных компьютерного моделирования яркостной температуры

```
import numpy as np
import pandas as pd
from sklearn import svm
import matplotlib.pyplot as plt
from sklearn import neighbors
from sklearn.naive_bayes import GaussianNB
from sklearn.utils import shuffle
modelFolder = '075'
def calcParams(testing, predict):
    zdorovieTesting = 0
    bolnieTesting = 0
    for item in testing:
        if item == 0:
            zdorovieTesting += 1
        else:
            bolnieTesting += 1
    zdoroviePredict = 0
    bolniePredict = 0
    for item in predict:
        if item == 0:
            zdoroviePredict += 1
        else:
            bolniePredict += 1
chuvstv = bolniePredict / bolnieTesting
specifich = zdoroviePredict / zdorovieTesting
return chuvstv,specifich

data = pd.read_csv(modelFolder + '/data.csv',
delimiter=',',
names=['0ртм', '1ртм', '2ртм', '3ртм', '4ртм', '5ртм', '6ртм', '7ртм', '8ртм',
'0ик', '1ик', '2ик', '3ик', '4ик', '5ик', '6ик', '7ик', '8ик', 'target'])
trainingTemperatures = data[['0ртм', '1ртм', '2ртм', '3ртм', '4ртм', '5ртм',
```

```
'6ртм', '7ртм', '8ртм', '0ик', '1ик', '2ик', '3ик', '4ик', '5ик', '6ик',
'7ик', '8ик']]
trainingTemperatures = shuffle(trainingTemperatures)
trainingTemperatures.reset_index(inplace=True, drop=True)
data.head()
testingData = pd.read_csv(modelFolder + '/testing.csv',
delimiter=',',
names=['0ртм', '1ртм', '2ртм', '3ртм', '4ртм', '5ртм', '6ртм', '7ртм', '8ртм',
'0ик', '1ик', '2ик', '3ик', '4ик', '5ик', '6ик', '7ик', '8ик', 'target'])
testingTemperatures = testingData[['0ртм', '1ртм', '2ртм', '3ртм', '4ртм',
'5ртм', '6ртм', '7ртм', '8ртм', '0ик', '1ик', '2ик', '3ик', '4ик', '5ик',
'6ик', '7ик', '8ик']]

#SVM
clf = svm.SVC(gamma='scale')
clf.fit(trainingTemperatures, data.target)

classes = clf.predict(testingTemperatures)

testingData.target
res = testingData.target == classes
trueCount = 0
for el in res:
if el == True:
trueCount += 1

print('Правильно определено:', trueCount, '\nНеправильно:', len(res) - trueCount)
chuvstv, specifich = calcParams(testingData.target, classes)
print('Чувствительность:', chuvstv, '\nСпецифичность:', specifich);

labels = 'Совпадает', 'Не совпадает'
values = [trueCount, len(res) - trueCount]
explode = (0.1, 0)

fig1, ax1 = plt.subplots()
ax1.pie(values, labels=labels, autopct='%1.1f%%',
shadow=True, startangle=90)
ax1.axis('equal')

plt.show()
```

```
#KNN
clf = neighbors.KNeighborsClassifier(20, weights='uniform')
clf.fit(trainingTemperatures, data.target)

classes = clf.predict(testingTemperatures)

testingData.target
res = testingData.target == classes
trueCount = 0
for el in res:
if el == True:
trueCount += 1

print('Правильно определено:', trueCount, '\nНеправильно:', len(res) - trueCount)
chuvstv, specifich = calcParams(testingData.target, classes)
print('Чувствительность:', chuvstv, '\nСпецифичность:', specifich);

labels = 'Совпадает', 'Не совпадает'
values = [trueCount, len(res) - trueCount]
explode = (0.1, 0)

fig1, ax1 = plt.subplots()
ax1.pie(values, labels=labels, autopct='%1.1f%%',
shadow=True, startangle=90)
ax1.axis('equal')

plt.show()

#NB
gnb = GaussianNB()
clf = gnb.fit(trainingTemperatures, data.target)

classes = clf.predict(testingTemperatures)

testingData.target
res = testingData.target == classes
trueCount = 0
for el in res:
if el == True:
```

```
trueCount += 1

print('Правильно определено:', trueCount, '\nНеправильно:', len(res) - trueCount)
chuvstv, specifich = calcParams(testingData.target, classes)
print('Чувствительность:', chuvstv, '\nСпецифичность:', specifich);

labels = 'Совпадает', 'Не совпадает'
values = [trueCount, len(res) - trueCount]
explode = (0.1, 0)

fig1, ax1 = plt.subplots()
ax1.pie(values, labels=labels, autopct='%1.1f%%',
shadow=True, startangle=90)
ax1.axis('equal')

plt.show()
```