УДК 614.84 004.8

ПРОГНОЗИРОВАНИЕ ПРИКЛАДНЫХ ЗАДАЧ С ПРИМЕНЕНИЕМ МЕТОДОВ МАШИННОГО ОБУЧЕНИЯ

Сабитов[1] Б.Р., Кубанычбекова[2] А.К., Осмонов[2] Э.Т., Калтаев[2] Б.Э., Орозобекова[2] А.К.
[1]КНУ им.Ж.Баласагына. [2]КГУСТА им.Н.Исанова.

Данная статья направлена на реализацию надежной модели машинного обучения, которая может эффективно прогнозировать прикладные задачи, в частности медицины. Изучается задача прогнозирования болезни пациентов, на основе имеющихся у него симптомов. Для решения задачи прогнозирования болезни пациентов построено модель с применением алгоритмов машинного обучения. Определена качество модели как матрица погрешностей.

Ключевые слова: Прогнозирование, прикладные задачи, машинное обучение, модели, медицина, оценка модели.

МАШИНАДАН ҮЙРӨӨНҮ ЫКМАЛАРЫН КОЛДОНУП КОЛДОНМО МАСЕЛЕЛЕРДИ БОЛЖОЛДОО

Сабитов[1] Б.Р., Кубанычбекова[2] А.К., Осмонов[2] Э.Т., Калтаев[2] Б.Э., Орозобекова2 А.К.
[2]Ж.Баласагын атындагы КУУ, [2]Н.Исанов атындагы КГУСТА

Бул макала колдонмо маселелерди чыгарууда, атап айтканда, медицинада, натыйжалуу машина үйрөнүү ишенимдүү моделин ишке ашырууга багытталган. Оорулуулардын белгилери боюнча ооруну болжолдоо маселеси изилденип жатат. Бейтаптардын оорусун алдын ала айтуу маселесин чечүү үчүн машина үйрөнүү алгоритмдерин колдонуу менен модель курулган. Моделдин сапаты ката матрицасы катары аныкталат.

Баштапкы сөздөр. Болжолдоо, колдонмо маселелер көйгөйлөр, машина үйрөнүү, моделдер, медицина, моделди баалоо.

FORECASTING APPLIED PROBLEMS USING MACHINE LEARNING METHODS

Sabitov[1] B., Kubanychbekova[2] A., Osmonov[2] E., Kaltaev[2] B., Orozobekova[2] A.

[1]KSU named after Zh. Balasagyn, [2]KSUCTA named after N.Isanov

This article is aimed at implementing a reliable machine learning model that can effectively predict applied problems, in particular medicine. The problem of predicting the disease of patients, based on their symptoms, is being studied. To solve the problem of predicting the disease of patients, a model was built using machine learning algorithms. The quality of the model is defined as an error matrix.

Keywords. Forecasting, applied problems, machine learning, models, medicine, model evaluation.

Введение. В настоящее время важную роль играет процесс цифровизации для многих отраслей экономики, науки, образования, медицины и производства. При этом целевое значение имеет требования, предъявляемые к базе данных процессов для построения модели. Подготовка данных является основным шагом для решения любой задачи машинного обучения. Для данной задачи прогнозирования, мы будем использовать конкретные данные, взятые из медицинского учреждения. В работе представлен открытый набор данных, которая состоит из двух файлов с расширением .csv, один для обучения, а другой для тестирования. Большое внимание уделяется к очистке данных и осуществляется как самый важный шаг для создания модели машинного обучения. Качество наших данных определяет качество нашей модели машинного обучения. Поэтому всегда необходимо очищать данные перед тем, как подавать их в модель для обучения. В нашем наборе данных все столбцы являются числовыми, целевой столбец, т.е. прогноз, является строковым типом и кодируется в числовую форму с использованием кодировщиков с применением Python технологий. После сбора и очистки данных данные они могут быть использованы для

обучения модели машинного обучения с применением алгоритмов, которые наиболее подходят для изучаемой задачи. В данной работе эти очищенные данные использованы для обучения классификаторов опорных векторов, наивного байесовского классификатора и классификатора случайного леса. Они представляют собой класс мощных алгоритмов нелинейного прогнозирования в машинном обучении. Для оценки модели и определения качества моделей мы будем использовать матрицу погрешностей.

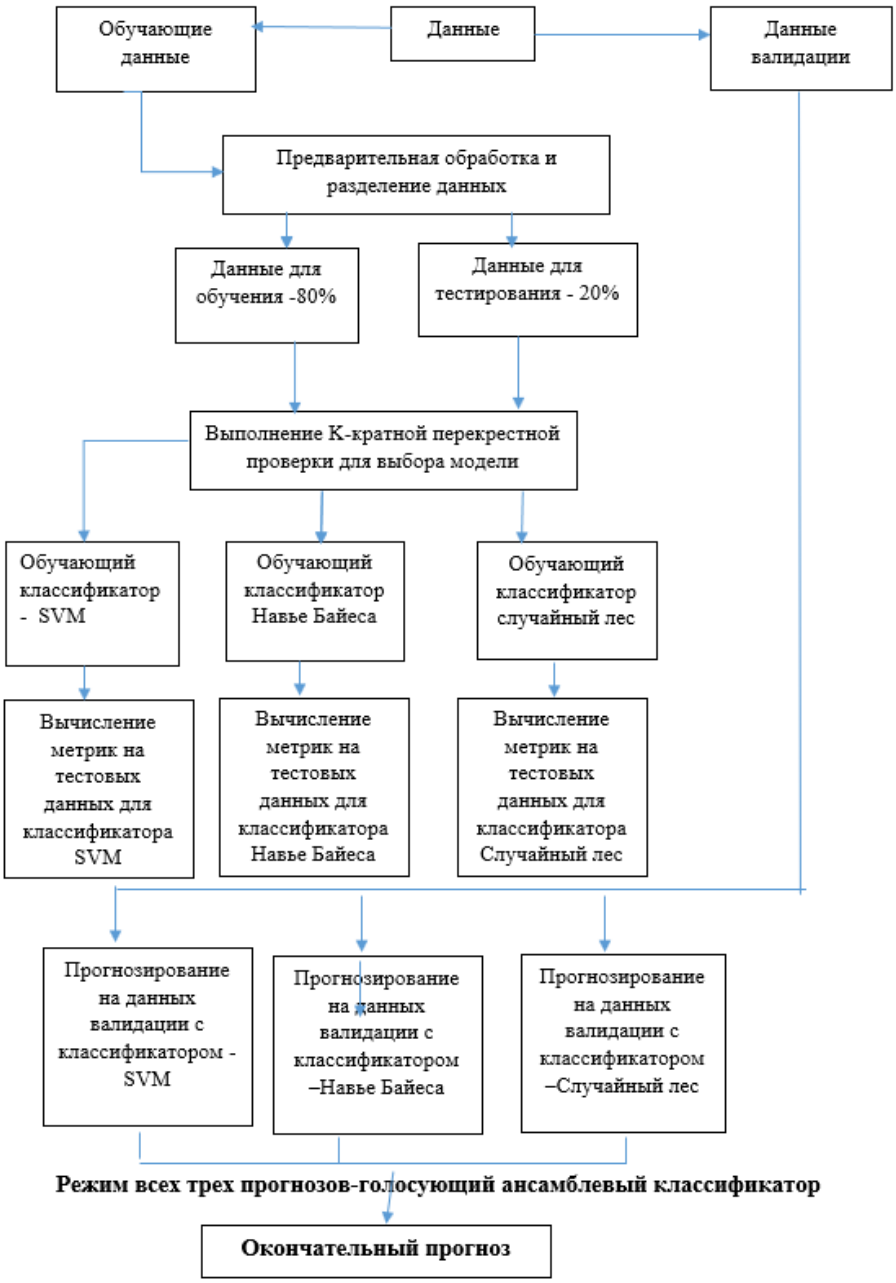


Рис.1.Схема реализации задачи прогнозирования

После обучения трех моделей мы будем прогнозировать заболевание по входным симптомам, комбинируя прогнозы всех трех моделей. Это делает наш общий прогноз более надежным и точным.
Наконец, мы определим функцию, которая принимает симптомы, разделенные запятыми, в качестве входных данных, прогнозирует заболевание на основе симптомов с использованием обученных моделей и возвращает прогнозы в формате JSON.

Рабочий процесс для реализации задачи прогнозирования представлено ниже для создания пакета программ на Python:

Методы исследования. Основные файлы базы данных обучения и тестирования модели будем загружать на локальный диск компьютера, а .csv файлы будут помещены в папку набора данных. Будем использовать среду Jupyter, системы Anaconda и создадим среду обработки базы данных для построения модели с применением библиотек Python. Использованы следующие основные пакеты машинного обучения

```
import numpy as np
import pandas as pd
from scipy.stats import mode
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.svm import SVC
from sklearn.naive_bayes import GaussianNB
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix
%matplotlib inline
```

Чтение набора данных

Сначала мы будем загружать набор данных из папок, используя библиотеку pandas. При чтении набора данных мы будем удалять нулевой столбец. Этот набор данных представляет собой чистый набор данных без нулевых значений, и все функции состоят из 0 и 1. Всякий раз, когда мы решаем задачу классификации, необходимо проверить, сбалансирован ли

наш целевой столбец или нет. Мы будем использовать гистограмму, чтобы проверить, сбалансирован ли набор данных или нет.

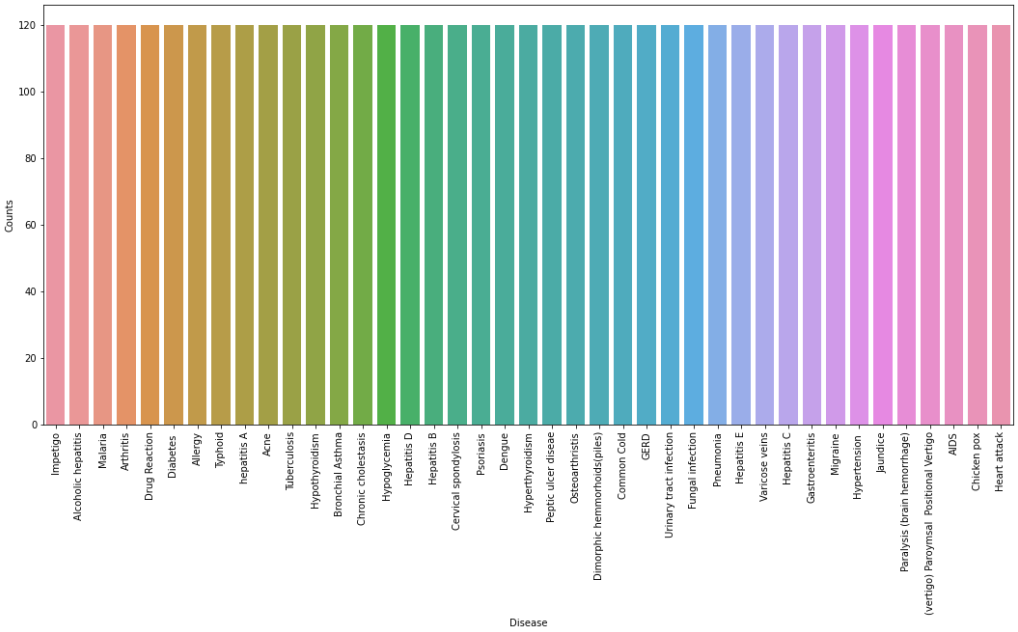


Рис.2. Гистограмма проверки сбалансированности базы данных симптомов болезней

Из приведенного выше графика видно, что набор данных является сбалансированным набором данных, т. е. имеется ровно 120 образцов для каждого заболевания, и дальнейшая балансировка не требуется. Мы можем заметить, что наш целевой столбец, то есть столбец прогноза, имеет тип данных объекта, этот формат не подходит для обучения модели машинного обучения. Итак, мы будем использовать кодировщик меток для преобразования столбца прогноза в числовой тип данных. Label Encoder преобразует метки в числовую форму, присваивая меткам уникальный индекс. Если общее количество меток равно n, то номера, присвоенные каждой метке, будут от 0 до n-1. Для кодирования целевого значения Будем использовать библиотеку LabelEncoder encoder = LabelEncoder() data["prognosis"] = encoder.fit_transform(data["prognosis"])

Разделение данных для обучения и тестирования модели.

Теперь, когда мы очистили наши данные, удалив нулевые значения и преобразовав метки в числовой формат, разделим данные для обучения и

тестирования модели. Мы будем разбивать данные в соотношении 80:20, т.е. 80% набора данных будет использоваться для обучения модели, а 20% данных будет использоваться для оценки производительности моделей.

Обучающие данные: (3936, 132), (3936,)
Данные для тестирования: (984, 132), (984,)

Построение модели

Результаты. После разделения данных мы теперь будем работать над частью моделирования. Мы будем использовать перекрестную проверку K-Fold для оценки моделей машинного обучения. Мы будем использовать классификаторы опорных векторов, гауссовский наивный байесовский классификатор и классификатор случайного леса для перекрестной проверки. Прежде чем перейти к части реализации, давайте познакомимся с k-кратной перекрестной проверкой и моделями машинного обучения.

K-Fold Cross-Validation: K-Fold cross-validation — это один из методов перекрестной проверки, в котором весь набор данных разбивается на k подмножеств, также известных как складки, затем обучение модели выполняется на k- 1 подмножестве, а оставшееся одно подмножество используется для оценки производительности модели.

Классификатор опорных векторов: Классификатор опорных векторов является дискриминационным классификатором, т.е. при наличии помеченных обучающих данных алгоритм пытается найти оптимальную гиперплоскость, которая точно разделяет выборки на разные категории в гиперпространстве.

Гауссовский наивный байесовский классификатор: это вероятностный алгоритм машинного обучения, который внутренне использует теорему Байеса для классификации точек данных.

Классификатор случайного леса: случайный лес — это алгоритм классификации контролируемого машинного обучения на основе ансамблевого обучения, который внутри использует несколько деревьев

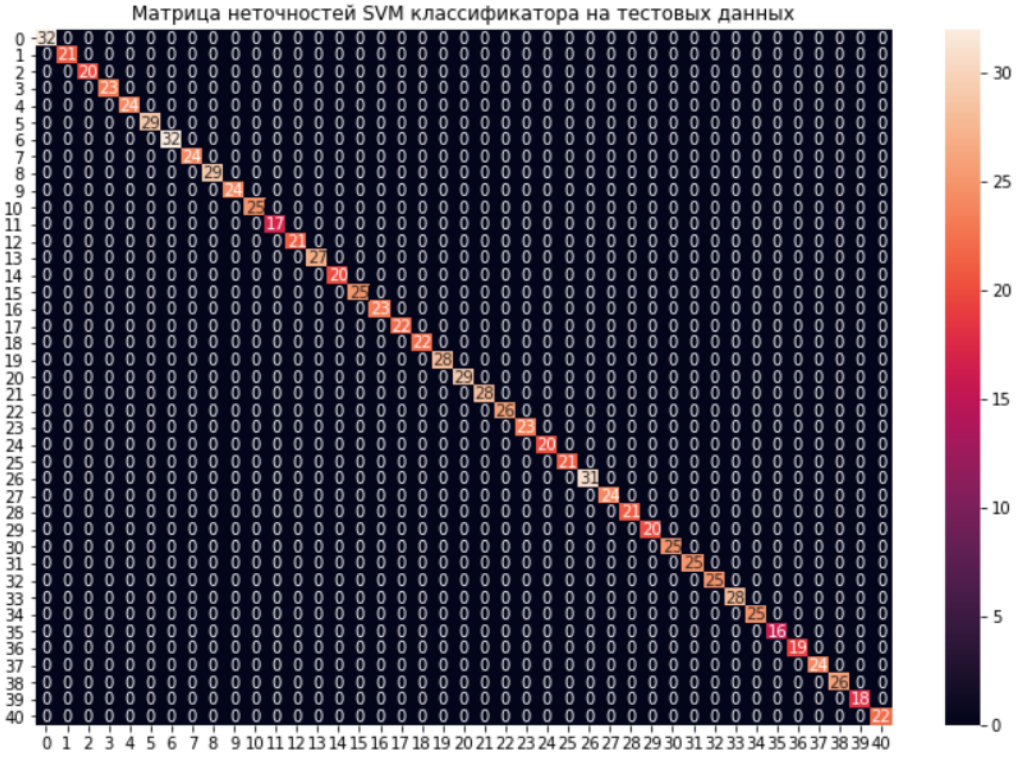
решений для выполнения классификации. В классификаторе случайного леса все внутренние деревья решений являются слабыми учениками, выходные данные этих слабых деревьев решений объединяются, т. е. режим всех прогнозов такой же, как и окончательный прогноз. Определим метрики оценки для k-кратной перекрестной проверки с использованием K-Fold Cross-Validation для выбора модели.

```
============================================================
Метод опорных векторов
Оценка: [1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
Средняя оценка: 1.0
============================================================
Гауссова Навье Байес алгоритм
Оценка: [1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
Средняя оценка: 1.0
============================================================
Алгоритм случайный лес
Оценка: [1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
Средняя оценка: 1.0
```

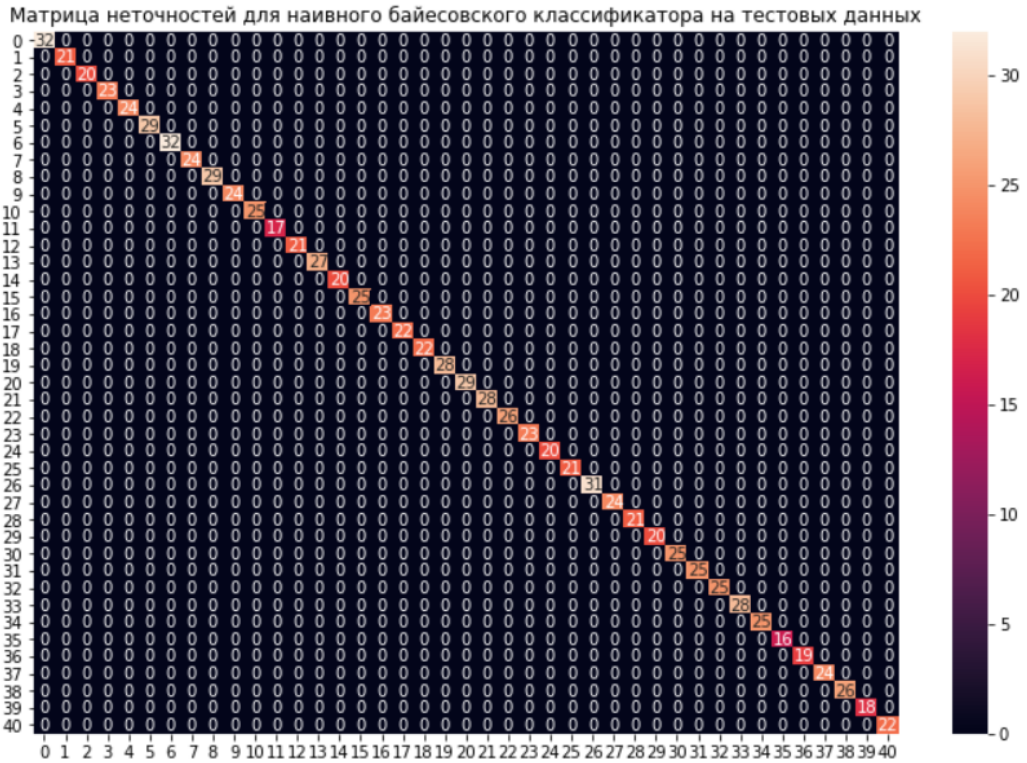
Из приведенного выше вывода мы можем заметить, что все наши алгоритмы машинного обучения работают очень хорошо, а средние баллы после k-кратной перекрестной проверки также очень высоки. Чтобы построить надежную модель, мы можем объединить, то есть взять режим прогнозов всех трех моделей, так что даже одна из моделей делает неправильные прогнозы, а две другие делают правильные прогнозы, тогда окончательный результат будет правильным. Этот подход поможет нам сделать прогнозы более точными на совершенно невидимых данных. В приведенном ниже коде мы будем обучать все три модели на данных обучения, проверять качество наших моделей с помощью матрицы погрешностей, а затем объединять прогнозы всех трех моделей как ансамбль или голосующий классификатор.

Построим классификатора путем объединения всех моделей: Training and testing SVM Classifier. Training and testing Naive Bayes Classifier Training and testing Random Forest Classifier

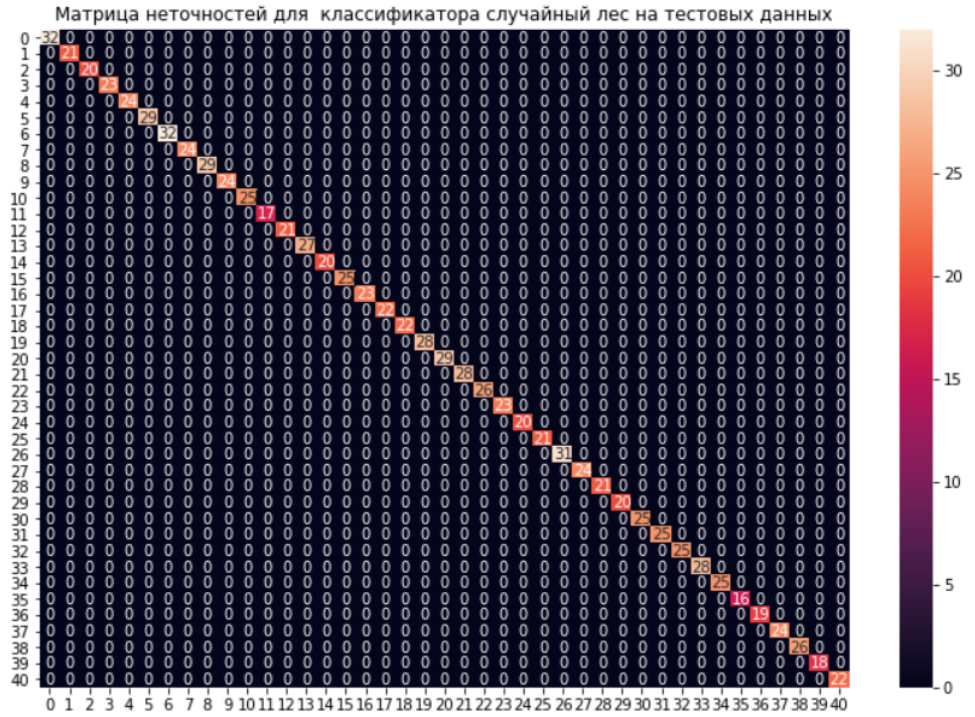
```
Точность данных обучения по классификатору SVM\: 100.0
Точность данных тестированияпо классификатору SVM\: 100.0
```




```
Точность данных обучения по классификатору - Naive Bayes Classifier\: 100.0
Точность данных тестированияпо классификатору - Naive Bayes Classifier\: 100.0
```

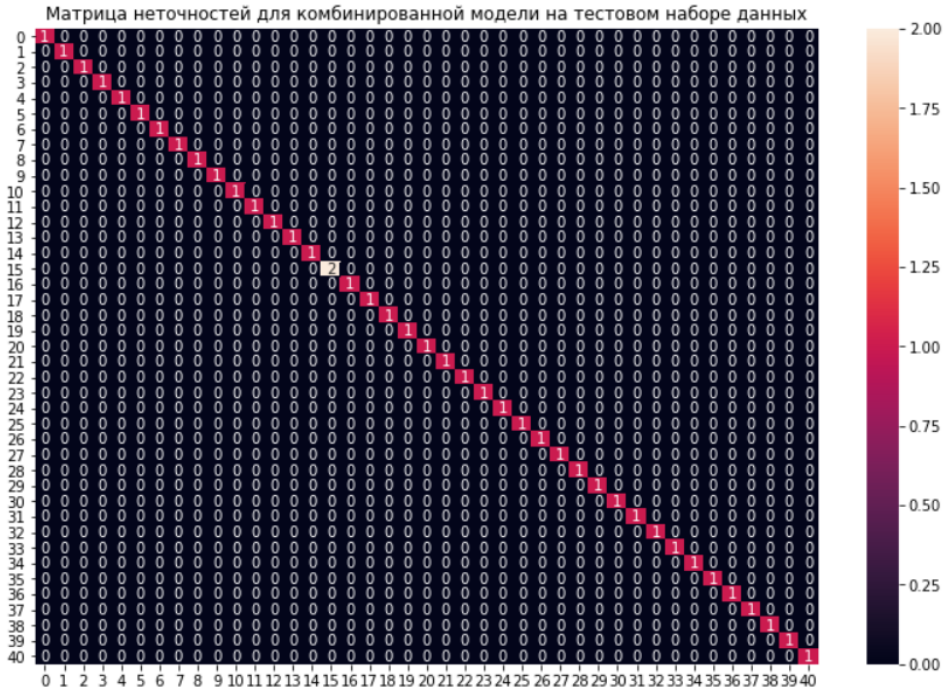


```
Точность данных обучения по классификатору- Random Forest \: 100.0
Точность данных тестированияпо классификатору-Random Forest \: 100.0
```



Из приведенных выше матриц путаницы или погрешностей, мы видим, что модели очень хорошо работают с невидимыми данными. Теперь мы будем обучать модели на всех обучающих данных, представленных в загруженном наборе данных, а затем протестировать нашу комбинированную модель на тестовых данных, представленных в наборе данных. Теперь обучим модель на целых данных для всей базы данных и проверим на тестовом наборе данных. Ниже использовано прогнозирования с использованием ансамбля со всеми классификаторами. Ниже представлен результат голосующего классификатора. Точность тестового набора данных по комбинированной модели: 100,0

```
Точность на тестовом наборе данных по комбинированной модели\:100.0
```



Мы видим, что наша комбинированная модель, точно классифицировала все точки данных. Мы подошли к заключительной части всей этой реализации, мы будем создавать функцию, которая принимает симптомы, разделенные запятыми, в качестве входных данных и выводит прогнозируемое заболевание, используя комбинированную модель на основе входных симптомов. Далее создадим функции, которая может принимать симптомы в качестве входных данных и генерировать прогнозы для болезни. Для этой цели создается индексный словарь симптомов для кодирования и симптомы переведем в числовую форму и сделаем окончательный прогноз применением всех алгоритмов машинного обучения. Вот результат

Выход:

```
{
    'rf_model_prediction': 'Грибковая инфекция',
    'naive_bayes_prediction': 'Грибковая инфекция',
    'svm_model_prediction': 'Грибковая инфекция',
    'final_prediction': 'Грибковая инфекция'
}
```

Симптомы, заданные в качестве входных данных для функции прогноза, должны быть точно такими же как 132 симптомов в наборе данных.

Заключение

В данной работе обучена модель с применением алгоритмов машинного обучения. Результаты тестирования показали, что модель на наборе данных по которой можно прогнозировать заболевания дали результаты процент прогноза которых достигают до 100% точности. Отметим, что данная технология определения болезней по симптомам носит образовательный характер и не применяется в реальной жизни без соответствующих врачебных исследований.

ЛИТЕРАТУРА

1. Ын Су – Теоретический минимум по Big Data. Всё что нужно знать о больших данных,2020.
2. Силен, Мейсман, Али – Основы Data Science и Big Data. Python и наука о данных,2020.
3. Дж. Вандер Плас – Python для сложных задач. Наука о данных и машинное обучение,2020
4. Хенрик Бринк, Джозеф Ричардс, Марк Феверолф. «Машинное обучение», 2017г.
5. Бастиан Шарден, Лука Массарон, Альберто Боскетти. «Крупномасштабное машинное обучение вместе с Python»,2017г.
6. Себастьян Рашка. «Python и машинное обучение», 2018 г.
7. Георгий Кухарев, Екатерина Каменская, Юрий Матвеев, Надежда Щеголева. «Методы обработки и распознавания изображений лиц в задачах биометрии».2018 г.
8. Петер Флах «Машинное обучение. Наука и искусство построения алгоритмов, которые извлекают знания из данных». 2019г.
9. Андреас Мюллер, Сара Гвидо. «Введение в машинное обучение с помощью Python»,2018 г.
10. Петер Флах. «Машинное обучение. Наука и искусство построения алгоритмов, которые извлекают знания из данных», 2019 г.