

INTELLICORP AN ENSEMBLE MODEL TO PREDICT CROP USING MACHINE LEARNING ALGORITHMS

submitted in partial fulfillment of the requirements for

the award of

Bachelor of Engineering

in

Computer Science and Engineering

By

PASTHAPURAM HARIKRISHNA

(38110193)

KETHAMUKKALA PHANINDRA

(38110250)



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade “A” by NAAC JEPPIAAR

NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600

119

MARCH 2022



SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

SCHOOL OF COMPUTING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the Bonafide work of PASTHAPURAM HARIKRISHNA (38110193), Kethamukkala Phanindra (38110250) who carried out the project entitled "**CARDIREST: A Machine Learning Approach to Identify Cardiovascular Disease**" under my supervision from November 2021 to March 2022.

**Internal Guide
DR.L.K Goshila Grace**

Head of the Department

Submitted for Viva-voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I **PASTHAPURAM HARIKRISHNA(38110193)**, Kethamukkala Phanindra **(38110250)** hereby declare that the Project Report entitled “**CARDIREST: A Machine Learning Approach to Identify Cardiovascular Disease**” done by us under the guidance of DR.L.K Goshila Grace (Internal) is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering in Computer Science and Engineering.

DATE:

PLACE:

SIGNATURE OF THE CANDIDATES

1.

Pasthapuram Harikrishna- 30110193

2.

Kethamukkala Phanindra - 30110250

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to the Board **of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph.D., Dean**, School of Computing **Dr. L. Lakshmanan M.E., Ph.D.** , and **Dr. S. Vigneshwari M.E., Ph.D.** Head of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. L. k. Goshila Grace M.E., Ph.D.** for her valuable guidance, suggestions, and constant encouragement that paved the way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

In this paper, We present a system for identifying cardiac disease that is both efficient and accurate that is based on machine learning techniques in this study. Some of the classification techniques used in the system include, logistic regression, , K-nearest neighbour, Nave bays, and decision trees, as well as standard feature selection algorithms. The strategies for feature selection are used to improve classification accuracy while reducing the classification system's execution time. In addition, to acquire best practises in model assessment and hyper parameter tweaking, the leave one subject out cross-validation method was used. The performance of the classifier is evaluated using performance measurement metrics. The classifiers' performance was assessed using features picked using feature selection methods. The proposed feature selection technique (FCMIM) appears to be practicable when used with a classifier support vector machine to develop a high-level intelligent system to identify heart sickness, according to the experimental results. Furthermore, the proposed method can be easily implemented in the healthcare industry to detect cardiac issues.

TABLE OF CONTENTS

Chapter no.	Title	Pg no.
1	INTRODUCTION	1
	1.1 OUTLINE OF THE PROJECT	1
	1.2 MACHINE LEARNING	1
2	LITERATURE SURVEY	3
3	ML ALGORITHM	6
	3.1 ALGORITHM USED	6
	3.1.1 Random Forest	6
	3.1.2 KNN	6
	3.1.3 Logistic Regression	7
4	DATASET	8
	4.1 ATTRIBUTE INFORMATION	8
5	SOFTWARE REQUIREMENTS	14
	5.1 SOFTWARE USED	14
	5.1.1 Pycharm	14
	5.1.2 Python 3.7 (64-bit)	14
	5.2 LIBRARIES USED	15
	METHODOLOGY	15
6	6.1 WORKFLOW DIAGRAM	15

	6.1.1. Random Forest	18
	6.1.2 Logistic Regression	20
	6.1.3. KNN	21
7	APPLICATION WORKING	25
	7.1 Machine Learning	26
	7.2. Web Application	26
8	RESULTS AND DISCUSSION	32
9	CONCLUSION	35
	REFERENCES	36
	APPENDIX	42
	A.SOURCE CODE	43
	B.SCREENSHOTS	49
	C.PLAGIARISM REPORT	54

CHAPTER 1

INTRODUCTION

1.1 OUTLINE OF THE PROJECT

Heart disease (HD) is the critical health issue and numerous people have been suffered by this disease around the world .The HD occurs with common symptoms of breath shortness, physical body weakness and, feet are swollen.

Researchers try to come across an efficient technique for the detection of heart disease, as the current diagnosis techniques of heart disease are not much effective in early time identification due to several reasons, such as accuracy and execution time.The diagnosis and treatment of heart disease is extremely difficult when modern technology and medical experts are not available

The effective diagnosis and proper treatment can save the lives of many people. According to the European Society of Cardiology, 26 million approximately people of HD were diagnosed and diagnosed 3.6 million annually.Most of the people in the United States are suffering from heart disease. Diagnosis of HD is traditionally done by the analysis of the medical history of the patient, physical examination report and analysis of concerned symptoms by a physician. But the results obtained from this diagnosis method are not accurate in identifying the patient of HD. Moreover, it is expensive and computationally difficult to analyze.

1.2 MACHINE LEARNING

Machine learning could be a subfield of computer science (AI). The Goal of machine learning typically is to know the structure of information and match that data into models which will be understood and used by folks. Although machine learning could be a field inside technology, it differs from ancient process

approaches. In ancient computing, algorithms are sets of expressly programmed directions employed by computers to calculate or downside solve. Machine learning algorithms instead give computers to coach on knowledge inputs and use applied math analysis so as to output values that fall inside a particular variable. Thanks to this, machine learning facilitates computers in building models from sample knowledge so as to modify decision-making processes supported knowledge inputs.

CHAPTER 2

LITERATURE REVIEW

- According to Benjamin EJ et al. [1,] there are seven major risk factors for heart disease: smoking, lack of physical activity, poor diet, obesity, high cholesterol, diabetes, and high blood pressure. They also talked about heart disease statistics, such as stroke and cardiovascular illness.
- In their experiments, Abhay Kishore et al. [2] found that recurrent neural networks outperform other algorithms such as CNN, Nave Bayes, and SVM in terms of accuracy. As a result, neural systems perform well in the prediction of cardiac disease. They also developed a system that could predict quiet cardiac failures and alert the client as soon as possible.
- When compared to other computations, M. Nikhil Kumar et al. [3] employed numerous algorithms– Decision tree, random forest, Na-ve Bayes, KNN, Support vector machine, logistic model tree technique, and Naive Bayes calculation. They used the cardiac illness dataset from the UCI repository. In addition, the J48 algorithm was quicker to develop and had good results.
- Amandeep Kaur et al. [4] examined various algorithms for heart disease prediction, including artificial neural networks, K–nearest neighbour, Nave Bayes, and Support vector machines.
- Sahaya Arthy et al. [5] examine the existing data mining-based work on heart disease prediction. In most cases, data mining methods are used to predict cardiac diseases. They also discuss the databases used, such as the UCI cardiology dataset, as well as the tools used, such as Weka, Quick Excavator, Information Dissolver, Apache Mahout, Clatter, Bottom, R, and others. They come to the conclusion that using a single algorithm improves prediction accuracy. However, combining two or more algorithms can help to strengthen and improve the accuracy of heart disease prediction
- Chala Beyene et al.[6] Proposed a methodology to read the circumstance of HD to overcome the problem of opinion of HD. It bettered the actuality

methodology by choosing Naïve Bayes, J48, and SVM for predicting the circumstance of HD for early automatic opinion in short time in order to support the rates of services and reduce costs to save the life of individualities. This methodology uses various attributes of HD so as to spot whether a patent has HD or not. The comparison of study within the dataset is employed WEKA software.

- P.Sai Chandrasekhar Reddy et al.[7]Recommended ANN algorithms for HD prophecy system in DM. The main end of this predicting system is to reduce cost of a opinion like different type of test was done to make a decision for opinion of HD. so, they have proposed a new system to prognosticate the condition of the case predicated on their parameters analogous as age, blood pressure, eyeblink rate, cholesterol,etc. and estimate if a case has HD or not. The proposed system is handed its delicacy in java.

- Dwivedi t al.[8]Concentrated to estimate the performance of different ML algorithms for HD prophecy. The comparison between different algorithms analogous as Naïve Bayes, KNN, Logistic Retrogression and Bracket tree in order to identify the high performance for predictin

CHAPTER 3

ML ALGORITHM

In machine learning, tasks are generally classified into broad categories. These categories are based on how learning is received or how feedback on the learning is given to the system developed. Two of the most widely adopted machine learning methods are: -

Supervised learning which trains algorithms based on example input and output data that is labeled by humans.

Unsupervised learning provides the algorithm with no labeled data in order to allow it to find structure within its input data.

3.1 ALGORITHM USED

Because more than one class may be given to a single instance, multi-label classification (MLC) is the best and optimum option. The MLC is embedded into the algorithms used for categorization and model building, which are as follows:

1. Random Forest
2. KNN
3. Logistic Regression

3.1.1 *Random Forest*

This is an AI strategy used to predict and describe the connection between independent and dependent data values. It predicts the dependent variable by analyzing the relation between the other independent variables.

3.1.2 *KNN*

This is an AI strategy used to solve problems like regression and classification. The algorithm gives an outcome that is based on the predictions of the decision trees by taking the average or mean of the output from different trees. It can be used to predict what will happen in the future.

3.1.3. *Logistic Regression*

This is an AI strategy used to solve problems like regression and classification. The algorithm gives an outcome that is based on the predictions of the decision trees by taking the average or mean of the output from different trees. It can be used to predict what will happen in the future.

Dataset

Dataset –his database contains 306 attributes, but all published experiments refer to using a subset of 14 of them. In particular, the Cleveland database is the only one that has been used by ML researchers to this date. The "goal" field refers to the presence of heart disease in the patient. It is integer valued from 0 (no presence) to 4. Dimension Attribute-age, gender ,cp ,trestbps, chol, fbs, restecg ,thalach, exang, oldpeak, slope ,ca, thal, target.

Collecting data allows you to capture a record of past events so that we can use data analysis to find recurring patterns. From those patterns, you build predictive models using machine learning algorithms that look for trends and predict future changes.

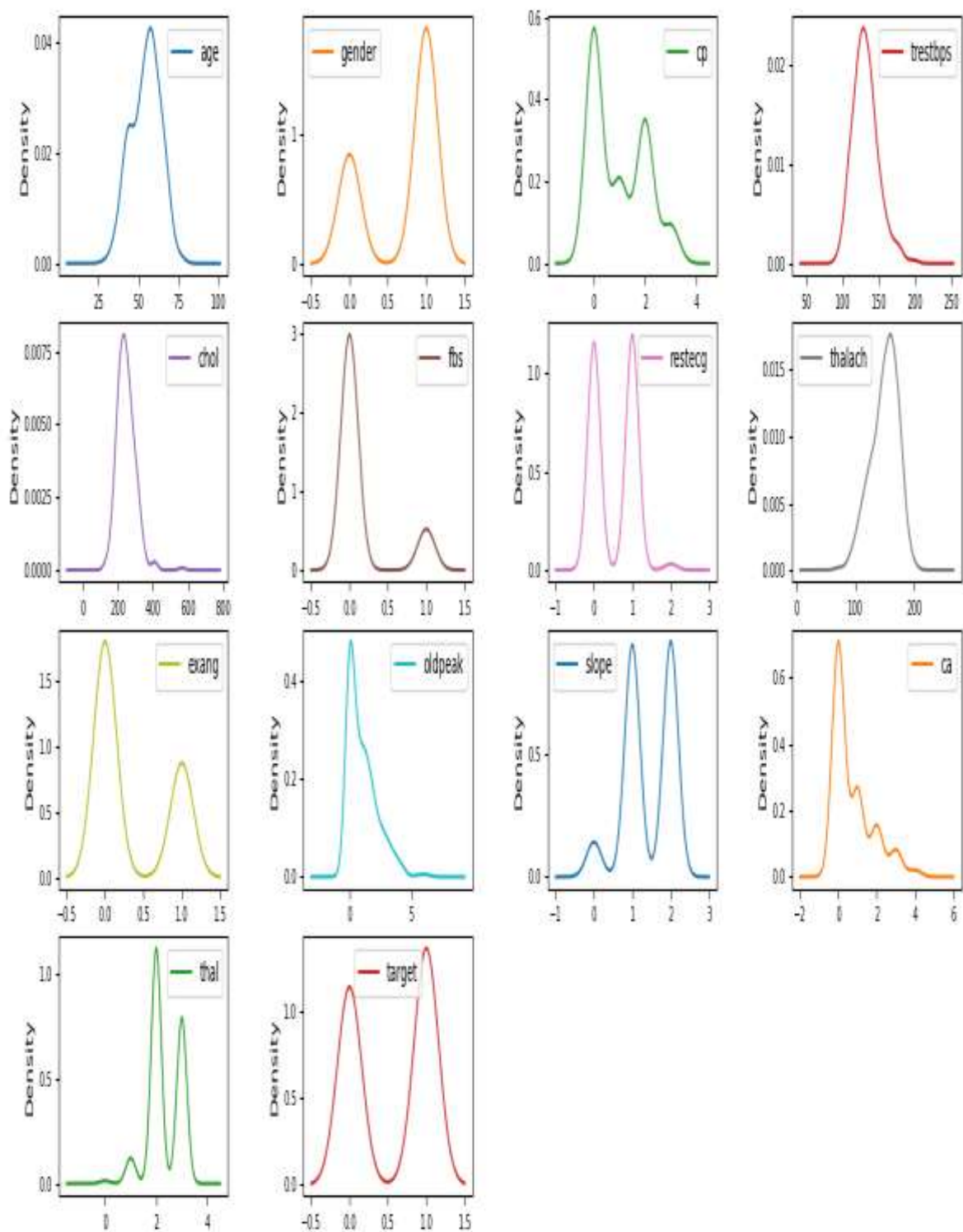
Predictive models are only as good as the data from which they are built, so good data collection practices are crucial to developing high-performing models.

The data need to be error-free (garbage in, garbage out) and contain relevant information for the task at hand. For example, a loan default model would not benefit from tiger population sizes but could benefit from gas prices over time. In this module, we collect the data from kaggle dataset archives. This dataset contains the information of divorce in previous years.

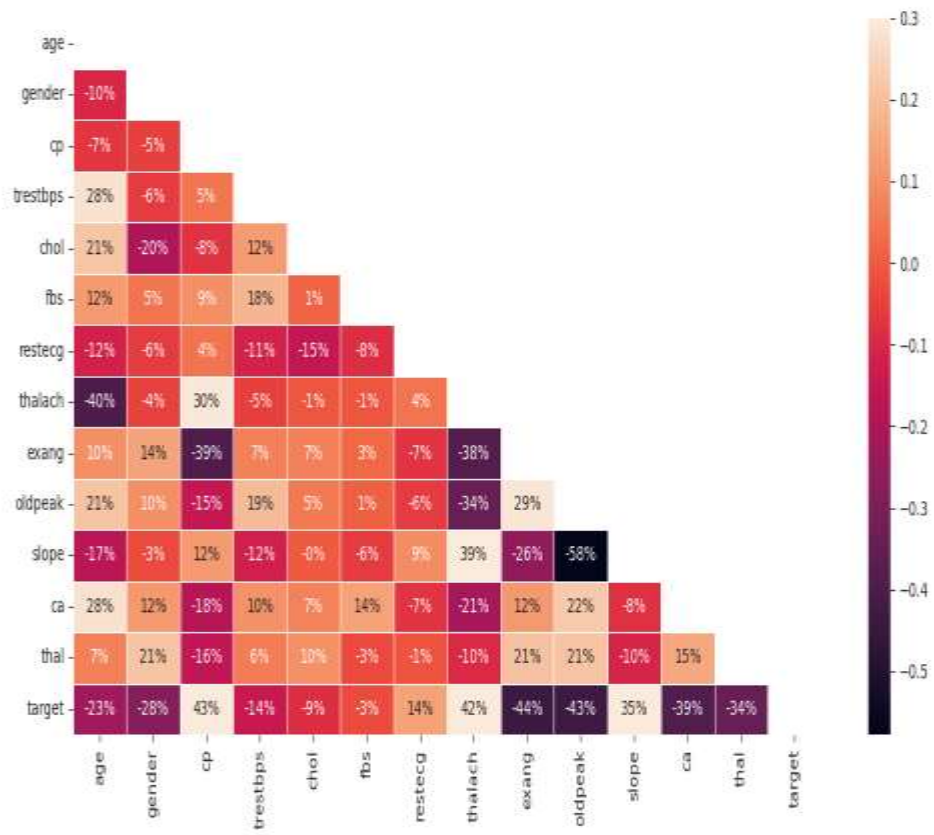
S no	Attributes	Description	
1	Age	In Years	
2	Sex	Male, Female	
3	Cp	Angina, Abnang, Notang, Asympt	
4	Trstbps	Resting blood pressure in mm hg	
5	Chol	Serum cholesterol in mg/dl	
6	Fbs	1 if fasting blood sugar is greater than 120 mg/dl, 2 if fasting blood sugar is less than 120 mg/dl	
7	Restecg	Electrocardiac graphic results	
8	Thalach	Maximum heart rate observed	
9	Exang	Exercise with angina has occurred	
10	Oldpeak	ST depression induced through exercise	

11	Slope	Slope of the ST segment	
12	Thal	By fluoroscopy, the number of main vessels ranges from 0 to 3.	
13	Ca	Heart status	
14	target	Output class	

Experiment Results



Correlation Map



CHAPTER 5

SOFTWARE REQUIREMENTS

5.1 SOFTWARE USED

5.1.2 PYCHARM

PyCharm is one of the most popular Python IDEs. There is a multitude of reasons for this, including the fact that it is developed by JetBrains, the developer behind the popular IntelliJ IDEA IDE that is one of the big 3 of Java IDEs and the “smartest JavaScript IDE” WebStorm. Having the support for web development by leveraging FLASK is yet another credible reason.

The main reason Pycharm for the creation of this IDE was for Python programming, and to operate across multiple platforms like Windows, Linux, and macOS. The IDE comprises code analysis tools, debugger, testing tools, and also version control options. It also assists developers in building Python plugins with the help of various APIs available. The IDE allows us to work with several databases directly without getting it integrated with other tools. Although it is specially designed for Python, HTML, CSS, and Javascript files can also be created with this IDE. It also comes with a beautiful user interface that can be customized according to the needs using plugins.

2. Python 3.7 (64-bit)

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. It is high-level built-in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms and can be freely distributed.

5.2 Libraries Used:

```
import warnings
import pickle
import pandas                as pd
import numpy                 as np
from sklearn.preprocessing   import OneHotEncoder
from sklearn.compose         import ColumnTransformer
from sklearn.model_selection import train_test_split, cross_val_score,
RandomizedSearchCV, GridSearchCV
from sklearn.ensemble        import RandomForestClassifier
from sklearn.linear_model    import LogisticRegression
from sklearn.neighbors       import KNeighborsClassifier
from sklearn.metrics         import accuracy_score, confusion_matrix,
precision_score, recall_score
warnings.filterwarnings('ignore')
```

CHAPTER 6

METHODOLOGY

7. METHODOLOGY

Workflow Diagram:

The proposed method is been developed using various ML strategies like Random Forest, Logistic Regression, and, Gradient Boosting combined to make an ensemble model as shown in fig.1 to predict and recommend the right crop.

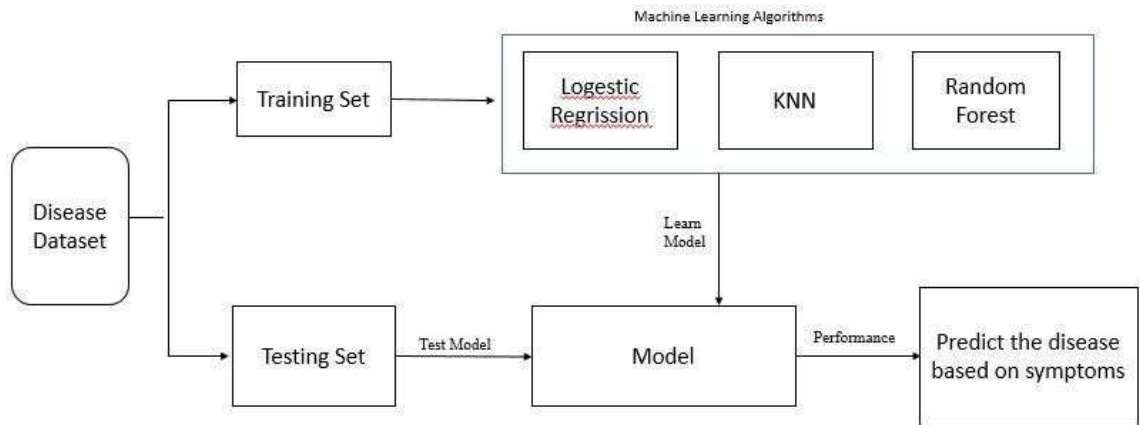


Fig.1 System flow diagram of the proposed method.

Because more than one class may be given to a single instance, multi-label classification (MLC) is the best and optimum option. The MLC is embedded into the algorithms used for categorization and model building, which are as follows:

- # Random Forest
- # Logistic Regression
- # K-Nearest Neighbour

Logistic regression

This is an ML strategy used to predict and describe the connection between independent and dependent data values. It predicts the dependent variable by analyzing the relation between the other independent variables. Here is the table of metrics for the particular machine learning algorithm which is in table 1.

Accuracy Score	89.55%
Precision Score	88.57%
Recall Score	91.18%

Table 1. report of proposed Logistic Regression.

Random Forest

This is an ML strategy used to solve problems like regression and classification. The algorithm gives an outcome that is based on the predictions of the decision trees by taking the average or mean of the output from different trees. It can be used to predict what will happen in the future. Here is the table of metrics for the particular machine learning algorithm which is in table 2.

Score	88.06%
-------	--------

Precision ScoreAccuracy	94.29%
Recall Score	84.62%

Table 2. report of proposed Random Forest.

K-Nearest Neighbour

This is an strategy that is used to solve problems like regression, classification, and other tasks. It is greedy and over lifts the training dataset quickly. The algorithm is penalized by penalizing various parts of the algorithm and the overall performance is improved. Here is the table of metrics for the particular machine learning algorithm which is in table 3.

Accuracy Score	79.10%
Precision Score	69.57%
Recall Score	88.89%

CHAPTER 7

APPLICATION WORKING

The resultant ensembled prediction model for the Heart Diseases prediction system was created using machine learning algorithms and is stored in the form of a python pickle file in order to make it easy to create an application. We have decided to create a web application and further develop it.

We used a flask to create our web application with which we were able to fulfill the requirement of the working application. Now let's look over the system architecture of the created flask application and deeply understand the workflow.

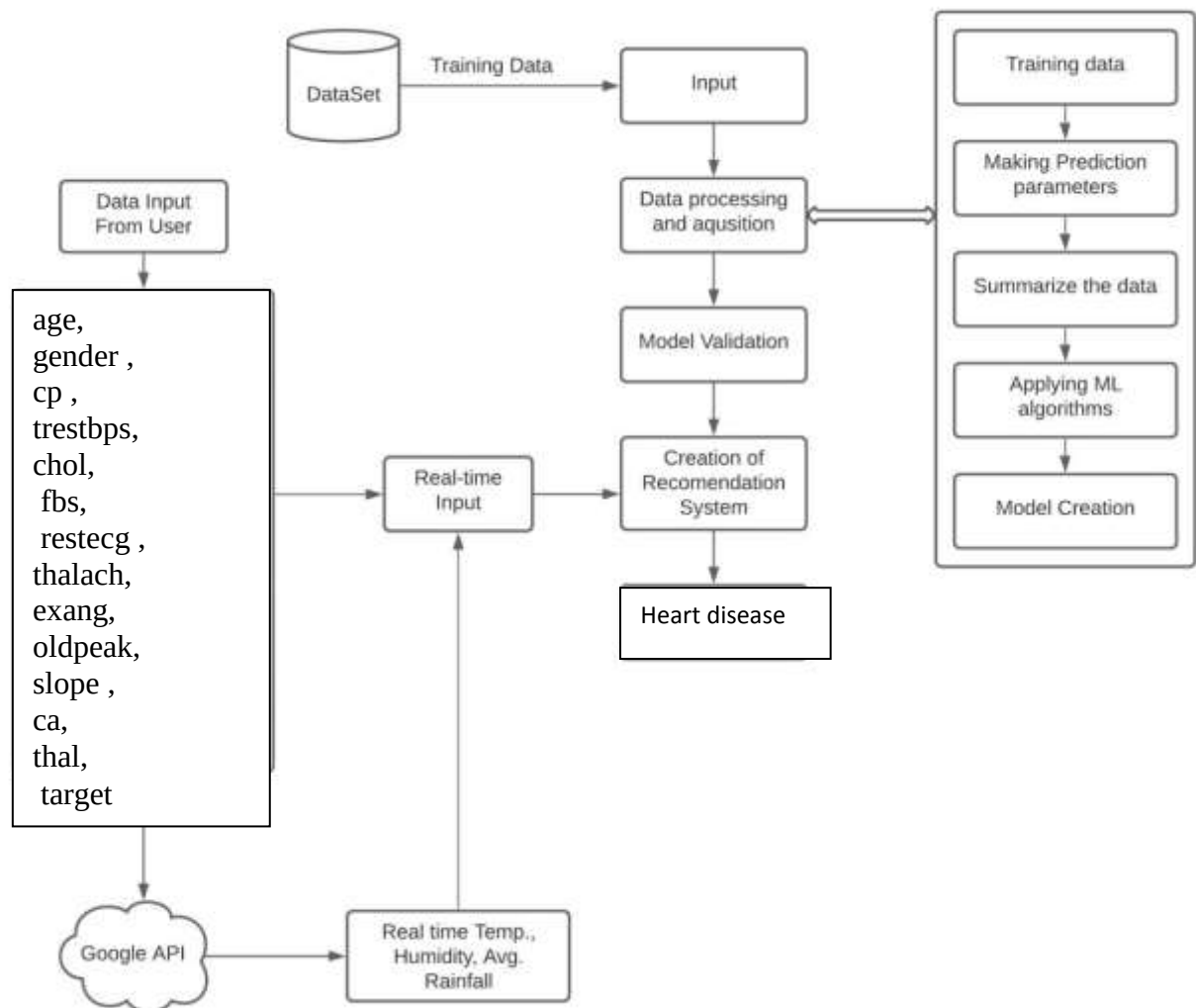


Fig. 7.1 System Architecture of Flask Application

The System architecture of the flask application has been divided into two different parts which have been explained in the further report.

1. Machine Learning
2. Web Application

7.1 Machine Learning

Because training a machine learning model requires expensive computational resources, we don't want our model to learn the model from the training data all over again. Pickle is a Python module that allows you to serialize and deserialize Python object structures. In other words, it's the process of transforming a Python object into a byte stream in order to save it to a file/database, keep the program state between sessions, or send data over the network. The original object hierarchy can be recreated by unpickling the pickled byte stream. This procedure is comparable to object serialization in Java or .Net.

As we have completed the Prediction model from the machine learning part we obtained an ensemble model which is been used in the flask application by creating a pickle file named EnsembleModel.pkl and we created an EnsembleModelclassifier directory where we will later store the files and data for our web application. Within this EnsembleModelclassifier directory, we create a.pkl_objects subdirectory to save the serialized Python objects to our local drive. where we have overridden the pickle file with the created ensemble model using pickle's dump() method, we then serialized the trained Ensemble model as well as the stop word set from the NLTK library so that we can avoid installing the NLTK vocabulary on our server.

7.2. Web Application

Flask is a compact and lightweight Python web framework that provides essential tools and capabilities for building online applications in Python. Because you can construct a web application rapidly using only a single Python file, it allows developers greater flexibility and is a more accessible framework for

beginning developers. Flask is also extensible, and it doesn't require complicated boilerplate code or a specific directory structure to get started.

Data Entry from the user, First we need to get the soil test done from the soil and irrigation department from the local panchayat or the local soil testing office from which we can acquire the independent values like the soil moisture, soil contents such as age, gender ,cp ,trestbps, chol, fbs, restecg ,thalach, exang, oldpeak, slope ,ca, thal, target value which will be our key data points for making the predictions.

Now let's understand the working of the weather/google API:

We want to keep the focus of the example on using and evaluating weather prediction data from a Weather API, so we'll utilize as few web technologies as possible. This may then be used as a starting point for creating a more comprehensive weather prediction. To display the forecast within the website, we'll utilize JavaScript, HTML, and CSS. To keep the JavaScript short and basic, we'll utilize jQuery, a popular tool for retrieving data and formatting HTML. The jQuery library isn't required; it only aids in keeping the code shorter and more understandable in this scenario.

```
def weather_fetch(city_name):
    api_key = config.weather_api_key
    base_url = "http://api.openweathermap.org/data/2.5/weather?"
    complete_url = base_url + "appid=" + api_key + "&q=" + city_name
    response = requests.get(complete_url) x = response.json()

    if x["cod"] != "404":
        y = x["main"]

        temperature = round((y["temp"] - 273.15), 2)
        humidity = y["humidity"]
        return temperature, humidity
    else:
        return None
```

We need to figure out how to get the weather forecast data before we can start building the web page. Typically, services like these are housed in the cloud since they manage a large quantity of meteorological data, including the prediction as well as historical information for thousands of weather stations that can go back many years. A weather API is developed to allow applications, such as the web page we're creating, to simply obtain the data. We need a Weather API for a web page that allows us to retrieve data over a web connection. So, we created an `api_key` which helps us connect with the server of the weather API and helps us get the response which is the temperature, humidity, and rainfall of the particular entry.

The rendering of the flask application goes in such way while connecting the home page and the inputs page for making the predictions:

```
<!DOCTYPE html>
<html >
<head>
  <meta charset="UTF-8">
  <link href="/static/index_HD.css" rel="stylesheet">

  <title>Heart_Disease_API</title>
  <link rel = "icon" href = "/static/love.png" >

</head>

<body class="heart__analysis__container">
  <!-- </img> -->
  <div class="login container">
    <h1 class="header">
      <a href="/" ></a>
      <span class="title">
        Heart Disease Analysis
      </span>
    </h1>

    <!-- Main Input For Receiving Query to our ML -->
    <div class="form__container">
      <form action="{{ url_for('predict') }}" method="post" class="form">

        <!-- <label>Age</label> -->
        <div class="row">
          <div class="col-25">
            <label for="fname">Age</label>
          </div>
          <div class="col-75">
            <input name="Age" type="number" min="1" max="100"
placeholder="Age">
          </div>
        </div>

        <!-- <label>Gender</label> -->
        <div class="row">
          <div class="col-25">
```

```

        <label for="fname">Gender</label>
    </div>
    <div class="col-75">
        <select id="Gender" name="Gender">
            <option value="1">Male</option>
            <option value="0">Female</option>
        </select>
    </div>
</div>

<div class="row">
    <div class="col-25">
        <label>Chest pain type</label>
    </div>
    <div class="col-75">
        <select id="Chest pain type" name="CP">
            <option value="0">Typical Angina</option>
            <option value="1">Atypical Angina</option>
            <option value="2">Non-Anginal pain</option>
            <option value="3">Asymptomatic</option>
        </select>
    </div>
</div>

<div class="row">
    <div class="col-25">
        <label>Resting blood pressure</label>
    </div>
    <div class="col-75">
        <input type="text" name="RBP" placeholder="Resting blood
pressure(0-300)" required="required" autocomplete="off"/>
    </div>
</div>

<div class="row">
    <div class="col-25">
        <label>Serum cholestoral</label>
    </div>
    <div class="col-75">
        <input type="text" name="SC" placeholder="Serum cholestoral(0-
600)mg/dl" required="required" autocomplete="off"/>
    </div>
</div>

<div class="row">
    <div class="col-25">
        <label>Fasting blood sugar</label>
    </div>
    <div class="col-75">
        <select id="Fasting blood sugar" name="FBS">
            <option value="1">Yes</option>
            <option value="0">No</option>
        </select>
    </div>
</div>

<div class="row">
    <div class="col-25">
        <label>Resting electrocardiographic results</label>
    </div>
    <div class="col-75">
        <select id="Resting electrocardiographic results" name="RER">
            <option value="0">Normal</option>
            <option value="1">ST-T wave abnormality</option>
            <option value="2">Showing probable or definite left
ventricular hypertrophy by Estes' criteria</option>
        </select>
    </div>
</div>

```

```

        </select>
    </div>
</div>

<div class="row">
    <div class="col-25">
        <label>Maximum heart rate achieved</label>
    </div>
    <div class="col-75">
        <input type="text" name="MHR" placeholder="Maximum heart rate
achieved(0-300)" required="required" autocomplete="off" />
    </div>
</div>

<div class="row">
    <div class="col-25">
        <label>Exercise induced angina</label>
    </div>
    <div class="col-75">
        <select id="Exercise induced angina" name="EIA">
            <option value="1">Yes</option>
            <option value="0">No</option>
        </select>
    </div>
</div>

<div class="row">
    <div class="col-25">
        <label>ST depression</label>
    </div>
    <div class="col-75">
        <input type="text" name="ST" placeholder="ST depression induced
by exercise relative to rest(0-10)" required="required"
autocomplete="off"/>
    </div>
</div>

<div class="row">
    <div class="col-25">
        <label>ST segment</label>
    </div>
    <div class="col-75">
        <select id="The slope of the peak exercise ST segment"
name="SST">
            <option value="1">Flat</option>
            <option value="0">Upsloping</option>
            <option value="2">Downsloping</option>
        </select>
    </div>
</div>

<div class="row">
    <div class="col-25">
        <label>Number of major vessels(Flourosopy)</label>
    </div>
    <div class="col-75">
        <select id="Number of major vessels colored by flourosopy"
name="Flourosopy">
            <option value="0">0</option>
            <option value="1">1</option>
            <option value="2">2</option>
            <option value="3">3</option>
            <option value="4">4</option>
        </select>
    </div>
</div>

```

```

</div>

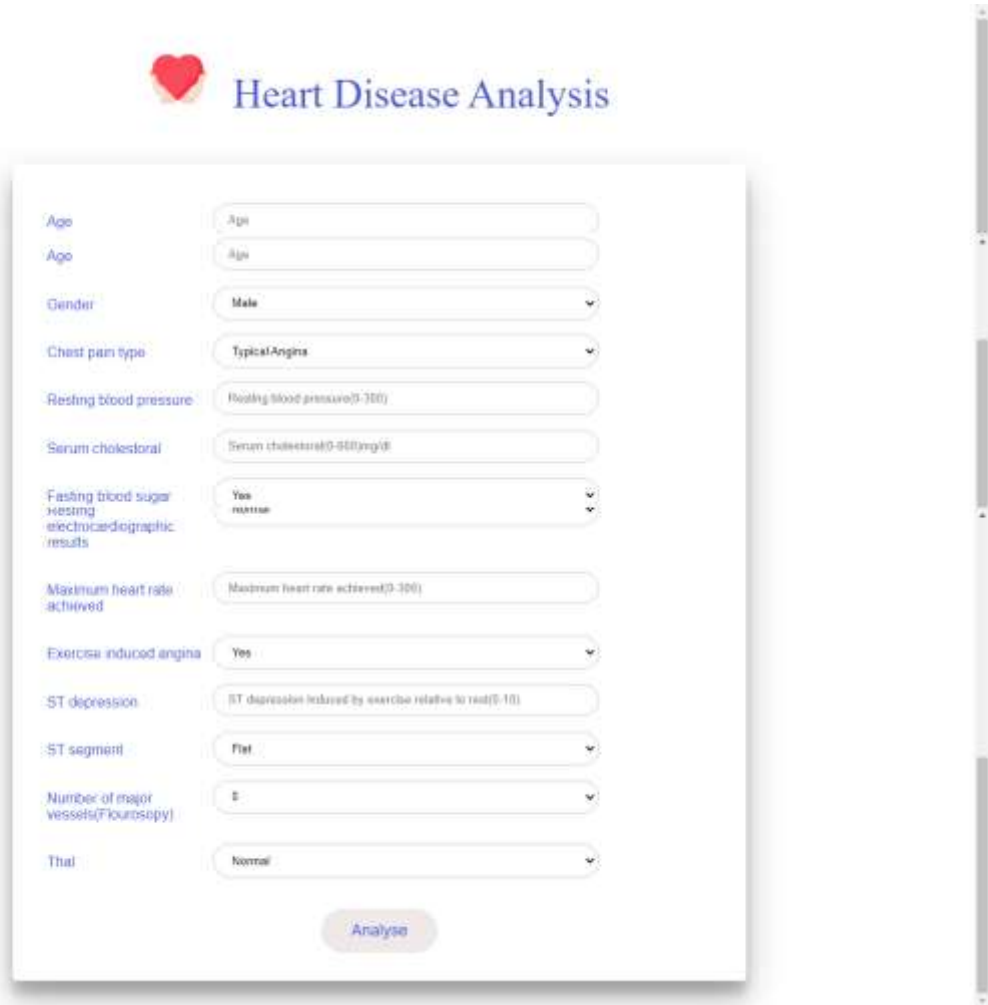
<div class="row">
  <div class="col-25">
    <label>Thal</label>
  </div>
  <div class="col-75">
    <select id="Thal" name="Thal">
      <option value="1">Normal</option>
      <option value="2">Fixed defect</option>
      <option value="3">Reversable defect</option>
    </select>
  </div>
</div>

<span class="button__wrapper">
  <button type="submit" class="analyse__btn" >Analyse</button>
</span>

</form>
</div>
<div class='footer'>
  <div class="contact">
    <a target="_blank" href=""></a>
  </div>
  <p class="footer__description">Made by HARIKRISHNA & PHANINDRA</p>
</div>
</div>
</body>
</html>

```

Fig 7.2 Application Screenshots



The screenshot shows a web application titled "Heart Disease Analysis" with a red heart icon. The form contains the following fields and options:

- Age: Two text input fields.
- Gender: A dropdown menu with "Male" selected.
- Chest pain type: A dropdown menu with "Typical Angina" selected.
- Resting blood pressure: A text input field with the placeholder "Resting blood pressure(0-300)".
- Serum cholesterol: A text input field with the placeholder "Serum cholesterol(0-600)mg/dl".
- Fasting blood sugar (fasting electrocardiographic results): A dropdown menu with "Yes" selected.
- Maximum heart rate achieved: A text input field with the placeholder "Maximum heart rate achieved(0-300)".
- Exercise induced angina: A dropdown menu with "Yes" selected.
- ST depression: A text input field with the placeholder "ST depression induced by exercise relative to rest(0-10)".
- ST segment: A dropdown menu with "Flat" selected.
- Number of major vessels (Fluoroscopic): A dropdown menu with "0" selected.
- Thal: A dropdown menu with "Normal" selected.

An "Analyse" button is located at the bottom of the form.

As we are on the page for making the predictions now we get the inputs from the user and hit the prediction key inorder to make the prediction.

render Heart Disease Prediction result page

```
<!DOCTYPE html>
<html >
<head>
  <meta charset="UTF-8">
  <link href="/static/index_HD.css" rel="stylesheet">

  <title>Heart_Disease_API</title>
  <link rel = "icon" href = "/static/love.png" >
</head>

<body class="heart__analysis__container">
  <!-- </img> -->
  <div class="login container">
    <h1 class="header">
      <a href="/" ></a>
      <span class="title">
        Heart Disease Analysis
      </span>
    </h1>
```

```

<div class="prediction__text">
  {{prediction_text}}
</div>

<script>
function goBack() {
  window.history.back();
}
</script>

<span class="button__wrapper">
  <button onclick="goBack()" class="analyse__btn" >Click here to test
another patient</button>
</span>

</form>
</div>
<div class='footer'>
  <div class="contact">
    <a target="_blank" href="https://www.linkedin.com/in/Subhamp7"></a>
  </div>
  <p class="footer__description">Made by HARIKRISHNA & PHANINDRA</p>
</div>
</div>
</body>
</html>

```

Fig 7.3 Application Output Screenshots



Heart Disease Analysis

Hurray!!! You are Safe

[Click here to test another patient](#)



Made by HARIKRISHNA & PHANINDRA



Heart Disease Analysis

OOPS!!! You need medical attention

[Click here to test another patient](#)



Made by HARIKRISHNA & PHANINDRA

CHAPTER 8

RESULTS AND DISCUSSION

In comparison to KNN and Random Forest, Logistic Regression fared very well towards the end of our trial. The accuracy of the model generated with Logistic Regression is 91 percent, which is 3 percent higher than Random Forest and approximately 11 percent higher than KNN. which is shown in **table 8.1.** and **fig.8.1.**

Table 8.1. Accuracy of the individual machine learning algorithm and ensemble model

Random Forest

Accuracy Score	88.06%
Precision Score	94.29%
Recall Score	84.62%

Logistic Regression

Accuracy Score	89.55%
Precision Score	88.57%
Recall Score	191.18%

K-Nearest Neighbour

Accuracy Score	78.691235
Precision Score	85.29565433
Recall Score	78.38346574

In machine learning, a classification report is a performance evaluation indicator. It's used to demonstrate your trained classification model's accuracy, recall, F1 Score, and support. This post is for you if you have never used it to analyze the performance of your model. In this post, I'll walk you through an introduction to machine learning classification reports and how to use Python to construct them.

It's one of the measures used to assess the success of a classification-based machine learning model. It shows the accuracy, recall, F1 score, and support of your model. It allows us to have a better grasp of our trained model's overall performance. To comprehend a machine learning model's categorization report, you must be familiar with all of the metrics shown. I've discussed all of the indicators below for a clear comprehension of the categorization report of your machine learning model:

Metrics	Definition
Precision	Precision is defined as the ratio of true positives to the sum of true and false positives.
Recall	Recall is defined as the ratio of true positives to the sum of true positives and false negatives.
F1 Score	The F1 is the weighted harmonic mean of precision and recall. The closer the value of the F1 score is to 1.0, the better the expected performance of the model is.
Support	Support is the number of actual occurrences of the class in the dataset. It doesn't vary between models, it just diagnoses the performance evaluation process.

Here is the classification report of the created ensemble model which shows the precision, recall, f1-score, and support which is shown in below Fig.8.1.

	precision	recall	f1-score	support
0	1.00	1.00	1.00	28
1	1.00	1.00	1.00	30
2	0.97	1.00	0.98	31
3	1.00	1.00	1.00	34
4	1.00	1.00	1.00	26
5	1.00	1.00	1.00	29
6	1.00	1.00	1.00	28
7	1.00	1.00	1.00	30
8	0.97	0.97	0.97	31
9	1.00	1.00	1.00	26
10	1.00	1.00	1.00	22
11	1.00	1.00	1.00	27
12	1.00	1.00	1.00	28
13	1.00	0.97	0.99	36
14	1.00	1.00	1.00	29
15	1.00	1.00	1.00	30
16	1.00	1.00	1.00	34
17	1.00	1.00	1.00	39
18	1.00	1.00	1.00	28
19	1.00	1.00	1.00	32
20	0.97	0.97	0.97	37
21	1.00	1.00	1.00	25
show more (open the raw output data in a text editor) ...				
accuracy			1.00	660
macro avg	1.00	1.00	1.00	660
weighted avg	1.00	1.00	1.00	660

Fig.8.1. Classification report of proposed ensemble model

CHAPTER 9

CONCLUSION

Although there are several limitations to overcome to be able to implement ML algorithms in clinical practice, overall ML algorithms showed promising results. However, selecting the proper algorithms for the appropriate research questions, comparison to human experts, validation cohorts, and reporting of all possible evaluation matrices are needed for study interpretation in the correct clinical context. Most importantly, prospective studies comparing ML algorithms to conventional risk models are needed. Once validated in that way, ML algorithms could be integrated with electronic health record systems and applied in clinical practice, particularly in high resources areas.

REFERENCES

[1

-] Benjamin EJ et.al Heart Disease and Stroke Statistics 2018 At-a-Glance (2018)
- [2] Abhay Kishore, Ajay Kumar, Karan Singh, Maninder Punia, Yogita Hambir Department of Computer Engineering, Army Institute of Technology, Pune, Maharashtra Professor, Department of Computer Engineering, Army Institute of Technology, Pune, Maharashtra Heart Attack Prediction Using Deep Learning (2018)
- [3] M. Nikhil Kumar, K. V. S. Koushik, K. Deepak Department of CSE, VR Siddhartha Engineering College, Vijayawada, Andhra Pradesh, India Prediction Heart Diseases using Data mining and machine learning algorithms and tools. (2018)
- [4] Amandeep Kaur, Jyoti Arora Dept of CSC Desh Bhagat University, Punjab, India heart disease prediction using data mining techniques: a survey (2018)
- [5] A. Sahaya Arthy, G. Murugeswari A survey on heart disease prediction using data mining techniques (April 2018)
- [6] R. Detrano, A. Janosi, W. Steinbrunn, M. Pfisterer, J.-J. Schmid, S. Sandhu, K. H. Guppy, S. Lee, and V. Froelicher, "International application of a new probability algorithm for the diagnosis of coronary artery disease," *The American journal of cardiology*, vol. 64, no. 5, pp. 304–310, 1989.
- [7] B. Edmonds, "Using localised 'gossip' to structure distributed learning," 2005.
- [8] Fsd fsdf Bayu Adhi Tama, 1 Afriyan Firdaus, 2 Rodiyatul FS, "Detection of Type 2 Diabetes Mellitus with Data Mining Approach Using Support Vector Machine", Vol. 11, issue 3, pp. 12-23, 2008.
- [9] Yu-Xuan Wang, QiHui Sun, Ting-Ying Chien, Po-Chun Huang, "Using Data Mining and Machine Learning Techniques for System Design Space Exploration and Automatized Optimization", *Proceedings of the 2017 IEEE International Conference on Applied System Innovation*, vol. 15, pp. 1079-1082, 2017.
- [10] Zhiqiang Ge, Zhihuan Song, Steven X. Ding, Biao Huang, "Data Mining and Analytics in the Process Industry: The Role of Machine Learning", *2017 IEEE Translations and content mining are permitted for academic research only*, vol. 5, pp. 20590-20616, 2017

APPENDIX

A. SOURCE CODE

Importing essential libraries and modules

```
# -*- coding:

try:
    #importing the libraries
    import warnings
    import pickle
    import pandas                as pd
    import numpy                 as np
    from sklearn.preprocessing   import OneHotEncoder
    from sklearn.compose         import ColumnTransformer
    from sklearn.model_selection import train_test_split, cross_val_score,
RandomizedSearchCV, GridSearchCV
    from sklearn.ensemble        import RandomForestClassifier
    from sklearn.linear_model     import LogisticRegression
    from sklearn.neighbors        import KNeighborsClassifier
    from sklearn.metrics          import accuracy_score, confusion_matrix,
precision_score, recall_score
    warnings.filterwarnings('ignore')

except Exception as e:
    print("Unable to import the libraries",e)

#=====Data
Preprocessing=====
#loading the dataset
dataset=pd.read_csv('heart_data.csv', sep='\t' )

#to check the different unique values in the dataset
for index in dataset.columns:
    print(index,dataset[index].unique())
print(dataset.dtypes)
print('The number of missing dataset:', dataset.isnull().sum().sum())

#splitting the dataset to independent and dependent sets
dataset_X=dataset.iloc[:, 0:13].values
dataset_Y=dataset.iloc[:, 13:14].values

#columns to be encoded: cp(2), restecg(6), slope(10), ca(11), thal(12)
ct=ColumnTransformer([ ('encoder', OneHotEncoder(drop='first'),
[2,6,10,11,12])], remainder='passthrough')
dataset_X = ct.fit_transform(dataset_X)

#splitting data to training set and test set
X_train, X_test, Y_train, Y_test =train_test_split(dataset_X, dataset_Y,
test_size=0.3 , random_state=0)

#=====Evaluation=====
=====
#scores

def scores(pred,test,model):
    print('\n=====Scores for {} =====\n'.format(model))
    print(f"Accuracy Score : {accuracy_score(pred,test) * 100:.2f}% ")
    print(f"Precision Score : {precision_score(pred,test) * 100:.2f}% ")
    print(f"Recall Score : {recall_score(pred,test) * 100:.2f}% ")
    print("Confusion Matrix :\n" ,confusion_matrix(pred,test))
```

```

#=====LR_Tunned=====
=====
#logistic regression

lr      = LogisticRegression()
solvers = ['newton-cg', 'lbfgs', 'liblinear']
penalty = ['l2']
c_values = [100, 10, 1.0, 0.1, 0.01]

# define grid search
grid_lr = {"solver":solvers,
           "penalty":penalty,
           "C":c_values}

grid_search_lr = GridSearchCV(estimator=lr, param_grid=grid_lr, n_jobs=-1,
cv=10, scoring='accuracy',error_score=0)

#getting the best parameters
grid_result_lr = grid_search_lr.fit(X_train,Y_train)
best_grid_lr   =grid_result_lr.best_estimator_
best_grid_lr.fit(X_train, Y_train)
Y_pred_knn_t   =best_grid_lr.predict(X_test)
scores(Y_pred_knn_t,Y_test,'KNeighbors_Classifier_Tunned')

#=====KNN=====
=====
#K Nearest Neighbour

#tunning the k value
accuracy=[]
for index in range(1,20):
    knn = KNeighborsClassifier(n_neighbors=index)
    score=cross_val_score(knn,X_train, Y_train,cv=10)
    accuracy.append(score.mean())

best_k=accuracy.index(max(accuracy))

#fitting the model with best k value
knn = KNeighborsClassifier(n_neighbors = best_k)
knn.fit(X_train, Y_train)
Y_pred_knn=knn.predict(X_test)
scores(Y_pred_knn,Y_test,'KNeighbors_Classifier')

#=====KNN_Tunned=====
=====
#K Nearest Neighbour with Hyper parameter

knn_t = KNeighborsClassifier()
n_neighbors = range(1,20)
weights = ['uniform', 'distance']
metric = ['euclidean', 'manhattan', 'minkowski']

# define grid search
grid = {'n_neighbors' : n_neighbors,
        'weights'      : weights,
        'metric'       : metric}

grid_search_knn = GridSearchCV(estimator=knn_t, param_grid=grid, n_jobs=-1,
cv=10 ,scoring='accuracy',error_score=0)

#getting the best parameters
grid_result_knn = grid_search_knn.fit(X_train,Y_train)
best_grid_knn   =grid_result_knn.best_estimator_
best_grid_knn.fit(X_train, Y_train)

```

```

Y_pred_knn_t =best_grid_knn.predict(X_test)
scores(Y_pred_knn_t,Y_test,'KNeighbors_Classifier_Tunned')

#=====RF=====
=====
#Random Forest

n_estimators      = [int(x) for x in np.linspace(start =10, stop = 200,
num = 10)]
max_features      = ['auto', 'sqrt','log2']
max_depth         = [int(x) for x in np.linspace(10, 1000,10)]
min_samples_split = [2, 5, 10,14,None]
min_samples_leaf  = [1, 2, 4,6,8,None]

random_grid = {'n_estimators'      : n_estimators,
               'max_features'      : max_features,
               'max_depth'         : max_depth,
               'min_samples_split' : min_samples_split,
               'min_samples_leaf'  : min_samples_leaf}

rf=RandomForestClassifier()

randomcv=RandomizedSearchCV(estimator=rf,param_distributions=random_grid,n_
iter=300,cv=10,
                           random_state=100,n_jobs=-1)

#getting the best parameters
randomcv.fit(X_train,Y_train)
best_grid=randomcv.best_estimator_
best_grid.fit(X_train,Y_train)
pred_2=best_grid.predict(X_test)
scores(pred_2,Y_test,'RandomForestClassifier')

#=====Saving the
models=====
#saving model to disk

pickle.dump(best_grid_lr, open('ml_model.pkl', 'wb'))
pickle.dump(ct,          open('encoder.pkl', 'wb'))

#=====Testing model
response=====
#test the pickle file

def test_model(row number):
    model=pickle.load(open('ml_model.pkl', 'rb'))
    value,real=dataset_X[row_number,:].reshape(1,-
1),dataset_Y[row_number,:]
    print("\n The value predicted is : {} and the real value is : {}
").format(model.predict(value), real))

test_model(102)
print('\nCompleted')

```

Accuracy & Graph

```

import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sn
from matplotlib import pyplot
from sklearn.metrics import precision_score, recall_score,
confusion_matrix, accuracy_score, roc_curve

```

```

from sklearn import metrics
data=pd.read_csv('heart_data.csv', sep='\t')

data.plot(kind='density' , subplots=True , layout=(4,4) , sharex=False ,
          fontsize=8 , figsize=(10,10))
plt.tight_layout()
plt.title('Density Map', y=1.1)

plt.show()

mask = np.zeros_like(data.corr())
mask[np.triu_indices_from(mask)] = True
with sns.axes_style("ticks"):
    f, ax = plt.subplots(figsize=(9, 5))
    ax = sns.heatmap(data.corr(), mask=mask,
vmax=.3,annot=True,fmt=".0%",linewidth=0.5,square=False)

plt.title('Correlation Map', y=1.1)

plt.show()

a = pd.get_dummies(data['cp'], prefix="cp")
b = pd.get_dummies(data['thal'], prefix="thal")
c = pd.get_dummies(data['slope'], prefix="slope")
d = pd.get_dummies(data['gender'], prefix="sex")

updated_clms = [data, a,b,c,d]
data = pd.concat(updated_clms, axis=1)
data.head()
data = data.drop(columns = ['cp','thal', 'slope', 'gender'])

from sklearn.preprocessing import StandardScaler
standardScaler = StandardScaler()
columns_to_scale = ['age', 'trestbps', 'chol', 'thalach', 'oldpeak']
data[columns_to_scale] = standardScaler.fit_transform(data[columns_to_scale])
y = data.target
X = data.drop(['target'], axis = 1)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size =
0.22,random_state=42)

from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors = 2) # n_neighbors means k
knn.fit(X_train, y_train)
prediction = knn.predict(X_test)
print("{} KNN Accuracy Score: {:.2f}%".format(2, knn.score(X_test,
y_test)*100))
print(f"Precision Score : {precision_score(prediction,y_test) * 100:.2f}%")
print(f"Recall Score : {recall_score(prediction,y_test) * 100:.2f}% ")
print("Confusion Matrix :\n",confusion_matrix(prediction,y_test))

from sklearn.ensemble import RandomForestClassifier
rf = RandomForestClassifier(n_estimators=1,random_state=0)
rf.fit(X_test,y_test)
acc = rf.score(X_test,y_test)*100
y_pred=rf.predict(X_test)
cf_matrix = confusion_matrix(y_test.T, y_pred)
print("Accuracy of Random Forest: {:.2f}%".format(acc))
print(f"Precision Score : {precision_score(y_pred,y_test) * 100:.2f}% ")
print(f"Recall Score : {recall_score(y_pred,y_test) * 100:.2f}% ")

```

```

print("Confusion Matrix :\n", cf_matrix)

from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(random_state = 2)
classifier.fit(X_train, y_train)
y_prediction = classifier.predict(X_test)
cm = confusion_matrix(y_test, y_prediction)
print (f"Accuracy of Logistic Regression : { accuracy_score(y_test, y_prediction) * 100:.2f}% ")
print(f"Precision Score : {precision_score(y_prediction,y_test) * 100:.2f}% ")
print(f"Recall Score : {recall_score(y_prediction,y_test) * 100:.2f}% ")
print ("Confusion Matrix : \n", cm)

# plot_confusion_matrix of knn
array = [[29,11],
         [3,24]]
df_confusion_matrix = pd.DataFrame(array, index = [i for i in "01"],
                                   columns = [i for i in "01"])
plt.figure(figsize = (2,2))
sn.heatmap(df_confusion_matrix, annot=True)
plt.title('Confusion matrix of knn', y=1.1)
plt.ylabel('Actual label')
plt.xlabel('Predicted label')
plt.show()

# plot_confusion_matrix of RF
array = [[26,6],
         [2,33]]
df_cf_matrix = pd.DataFrame(array, index = [i for i in "01"],
                             columns = [i for i in "01"])
plt.figure(figsize = (2,2))
sn.heatmap(df_cf_matrix, annot=True)
plt.title('Confusion matrix of random forest', y=1.1)
plt.ylabel('Actual label')
plt.xlabel('Predicted label')
plt.show()

# plot_confusion_matrix of LR
array = [[29,3],
         [4,31]]
df_cm = pd.DataFrame(array, index = [i for i in "01"],
                     columns = [i for i in "01"])
plt.figure(figsize = (2,2))
sn.heatmap(df_cm, annot=True)
plt.title('Confusion matrix of logistic regression', y=1.1)
plt.ylabel('Actual label')
plt.xlabel('Predicted label')
plt.show()

# calculate roc curves

lr_fpr, lr_tpr, _ = roc_curve(y_test, y_prediction)
# plot the roc curve for the model

pyplot.plot(lr_fpr, lr_tpr, marker='.', label='accuracy')
# axis labels
plt.title('ROC AUC for a Logistic Regression', y=1.1)
pyplot.xlabel('False Positive Rate')
pyplot.ylabel('True Positive Rate')
# show the legend
pyplot.legend()

```

```
# show the plot
pyplot.show()
```

Application code

Home Page

```
<!DOCTYPE html>
<html >
<head>
  <meta charset="UTF-8">
  <link href="/static/index_HD.css" rel="stylesheet">

  <title>Heart_Disease_API</title>
  <link rel = "icon" href = "/static/love.png" >

</head>

<body class="heart_analysis_container">
  <!-- </img> -->
  <div class="login container">
    <h1 class="header">
      <a href="/" ></a>
      <span class="title">
        Heart Disease Analysis
      </span>

    </h1>

    <!-- Main Input For Receiving Query to our ML -->
    <div class="form__container">
      <form action="{ url_for('predict') }}" method="post" class="form">

        <!-- <label>Age</label> -->
        <div class="row">
          <div class="col-25">
            <label for="fname">Age</label>
          </div>
          <div class="col-75">
            <input name="Age" type="number" min="1" max="100"
placeholder="Age">
          </div>
        </div>

        <!-- <label>Gender</label> -->
        <div class="row">
          <div class="col-25">
            <label for="fname">Gender</label>
          </div>
          <div class="col-75">
            <select id="Gender" name="Gender">
              <option value="1">Male</option>
              <option value="0">Female</option>
            </select>
          </div>
        </div>

        <div class="row">
          <div class="col-25">
            <label>Chest pain type</label>
          </div>
          <div class="col-75">
            <select id="Chest pain type" name="CP">
```

```

        <option value="0">Typical Angina</option>
        <option value="1">Atypical Angina</option>
        <option value="2">Non-Anginal pain</option>
        <option value="3">Asymptomatic</option>
    </select>
</div>
</div>

<div class="row">
    <div class="col-25">
        <label>Resting blood pressure</label>
    </div>
    <div class="col-75">
        <input type="text" name="RBP" placeholder="Resting blood
pressure(0-300)" required="required" autocomplete="off"/>
    </div>
</div>

<div class="row">
    <div class="col-25">
        <label>Serum cholestoral</label>
    </div>
    <div class="col-75">
        <input type="text" name="SC" placeholder="Serum cholestoral(0-
600)mg/dl" required="required" autocomplete="off"/>
    </div>
</div>

<div class="row">
    <div class="col-25">
        <label>Fasting blood sugar</label>
    </div>
    <div class="col-75">
        <select id="Fasting blood sugar" name="FBS">
            <option value="1">Yes</option>
            <option value="0">No</option>
        </select>
    </div>
</div>

<div class="row">
    <div class="col-25">
        <label>Resting electrocardiographic results</label>
    </div>
    <div class="col-75">
        <select id="Resting electrocardiographic results" name="RER">
            <option value="0">Normal</option>
            <option value="1">ST-T wave abnormality</option>
            <option value="2">Showing probable or definite left
ventricular hypertrophy by Estes' criteria</option>
        </select>
    </div>
</div>

<div class="row">
    <div class="col-25">
        <label>Maximum heart rate achieved</label>
    </div>
    <div class="col-75">
        <input type="text" name="MHR" placeholder="Maximum heart rate
achieved(0-300)" required="required" autocomplete="off" />
    </div>
</div>

<div class="row">
    <div class="col-25">

```

```

        <label>Exercise induced angina</label>
    </div>
    <div class="col-75">
        <select id="Exercise induced angina" name="EIA">
            <option value="1">Yes</option>
            <option value="0">No</option>
        </select>
    </div>
</div>

<div class="row">
    <div class="col-25">
        <label>ST depression</label>
    </div>
    <div class="col-75">
        <input type="text" name="ST" placeholder="ST depression induced
by exercise relative to rest(0-10)" required="required"
autocomplete="off"/>
    </div>
</div>

<div class="row">
    <div class="col-25">
        <label>ST segment</label>
    </div>
    <div class="col-75">
        <select id="The slope of the peak exercise ST segment"
name="SST">
            <option value="1">Flat</option>
            <option value="0">Upsloping</option>
            <option value="2">Downsloping</option>
        </select>
    </div>
</div>

<div class="row">
    <div class="col-25">
        <label>Number of major vessels(Flourosopy)</label>
    </div>
    <div class="col-75">
        <select id="Number of major vessels colored by flourosopy"
name="Flourosopy">
            <option value="0">0</option>
            <option value="1">1</option>
            <option value="2">2</option>
            <option value="3">3</option>
            <option value="4">4</option>
        </select>
    </div>
</div>

<div class="row">
    <div class="col-25">
        <label>Thal</label>
    </div>
    <div class="col-75">
        <select id="Thal" name="Thal">
            <option value="1">Normal</option>
            <option value="2">Fixed defect</option>
            <option value="3">Reversable defect</option>
        </select>
    </div>
</div>

```

```

        <span class="button__wrapper">
            <button type="submit" class="analyse__btn" >Analyse</button>
        </span>

    </form>
</div>
<div class='footer'>
    <div class="contact">
        <a target="_blank" href=""></a>
    </div>
    <p class="footer__description">Made by HARIKRISHNA & PHANINDRA</p>
</div>

</div>
</body>
</html>

```

Output Page

```

<!DOCTYPE html>
<html >
<head>
    <meta charset="UTF-8">
    <link href="/static/index_HD.css" rel="stylesheet">

    <title>Heart_Disease_API</title>
    <link rel = "icon" href = "/static/love.png" >
</head>

<body class="heart__analysis__container">
    <!-- </img> -->
    <div class="login container">
        <h1 class="header">
            <a href="/" ></a>
            <span class="title">
                Heart Disease Analysis
            </span>

        </h1>

        <div class="prediction__text">
            {{prediction_text}}
        </div>

        <script>
function goBack() {
    window.history.back();
}
</script>

        <span class="button__wrapper">
            <button onclick="goBack()" class="analyse__btn" >Click here to test
another patient</button>
        </span>

```

```

    </form>
</div>
<div class='footer'>
    <div class="contact">
        <a target="_blank" href="https://www.linkedin.com/in/Subhamp7"></a>
    </div>
    <p class="footer__description">Made by HARIKRISHNA & PHANINDRA</p>
</div>

</div>
</body>
</html>

```

FLASK CODE(backend)

Fask .py

```

import pickle
import numpy
from flask import Flask, request, render_template

app = Flask(__name__, static_url_path='/static')

#loading the models
model = pickle.load(open('ml_model.pkl', 'rb'))
enco = pickle.load(open('encoder.pkl', 'rb'))

@app.route('/')
def home():
    return render_template('index.html')

@app.route('/predict', methods=['GET', 'POST'])
def predict():
    '''
    For rendering results on HTML GUI
    '''
    try:
        attributes = ["Age", "Gender", "CP", "RBP", "SC", "FBS", "RER",
                     "MHR", "EIA", "ST", "SST", "Flourosopy",
                     "Thal"]
        attributes_val = [float(request.form[items]) for items in
attributes]
        attributes_array = (numpy.array(attributes_val)).reshape(1,-1)
        attributes_encode = enco.transform(attributes_array)
        prediction = model.predict(attributes_encode)

        if(prediction==1):
            output = 'OOPS!!! You need medical attention'
        else:
            output = 'Hurray!!! You are Safe'
    except:
        output="Sorry, some internal issue occured, Please try again, if
the issue presist inform via LinkedIn"

    return render_template('index_predict.html', prediction_text=output)

if __name__ == "__main__":
    app.run(debug=True)

```

```
import requests

url = "http://localhost:5000/predict"
r = requests.post(url, json={
    "Age": 45,
    "Gender": 1,
    "CP": 3,
    "RBP": 120,
    "SC": 110,
    "FBS": 1,
    "RER": 1,
    "MHR": 130,
    "EIA": 1,
    "ST": 26,
    "SST": 1,
    "Flourosopy": 3,
    "Thal": 1})

print(r.json())
```

C. PLAGIARISM REPORT

CARDIREST (1).docx

ORIGINALITY REPORT

2%

SIMILARITY INDEX

1%

INTERNET SOURCES

1%

PUBLICATIONS

0%

STUDENT PAPERS

PRIMARY SOURCES

1

www.astesj.com

Internet Source

1%

2

Jian Ping Li, Amin Ul Haq, Salah Ud Din,
Jalaluddin Khan, Asif Khan, Abdus Saboor.
"Heart Disease Identification Method Using
Machine Learning Classification in E-
Healthcare", IEEE Access, 2020

Publication

1%

3

1library.net

Internet Source

1%

Exclude quotes ☐ On

Exclude matches ☐ Off

Exclude bibliography ☐ On