Healthy Heart?

Table of Contents

Table of Figures

I. PROJECT DESCRIPTION

1. PROJECT TITLE AND TEAM MEMBERS

Project Title: Healthy Heart?

Team Members:

1. Neha Goud Baddam
2. Reshmi Chowdary Divi
3. Purandhara Maharshi Chidurala

GitHub Link: https://github.com/nehabaddam/SDAI-Project.git

2. GOAL AND OBJECTIVES

i. MOTIVATION

As the cause of death due to heart disease is increasing due to the sedentary lifestyle and unhealthy habits, it's important to curb it by helping people self-access their health using AI, at their homes without having to visit any hospital. After knowing their risk of getting CVD, they can change their habits or visit a medical expert for more advice. This is the main purpose of the "Healthy Heart?".

ii. SIGNIFICANCE

There is much research going on in the field of medicine to identify the risk factors of CVD, but few factors have been identified that contribute to the CVDs like diabetes, High blood pressure, food habits, lifestyle changes, obesity, etc. Studies are showing that reducing these risk factors for heart disease can help in preventing heart disease. By analyzing these factors there is still a scope to calculate the risk factor and warn people about their health, which can help them have a healthy life ahead. "Healthy Heart?" plays a significant role by using previous research data and the latest data about the above-mentioned factors to accurately predict the risk factor using AI algorithms.

iii. OBJECTIVES

The main objective of "Healthy Heart?" would be to take input (basic questions about the factors that cause heart disease) from the user and calculate the risk of having a heart disease based on the inputs given. Even if a person is healthy now they take this assessment, they can find out their risk of having heart disease and improve their lifestyle to make themselves healthier.

iv. FEATURES

"Healthy Heart" focuses on predicting the risk factor accurately. Firstly, we train the model with the existing data and use some latest data for validation and testing. Firstly. We shall try to implement it using different models and check the model that produces maximum accuracy. We shall be using the model with the maximum accuracy for predicting the risk factor. Once the model is trained, we can host the website to get the input from the user dynamically and produce the risk factor as output.

We are focusing on using the following classification models:

- Random Forest

Input: The webpage will be hosted that consists of many input variables like height, weight, age, diabetes(yes/no), hypertension(yes/no), etc. We are hoping to consider below inputs:

Attribute	Description
age	Age (int)
height	Height (int)
weight	Weight (float)
gender	Gender (categorical code)
sys_bp	Systolic blood pressure (int)
dia_bp	Diastolic blood pressure (int)
cholesterol	Cholesterol (1: normal, 2: above normal, 3: well above normal)
glucose	Glucose (1: normal, 2: above normal, 3: well above normal)
smoke	Smoking (binary)
alco	Alcohol intake (binary)
active	Physical activity (binary)
CVD	Presence or absence of cardiovascular disease (binary)

Output: A calculated risk factor.

II. INCREMENT - 2

1. INTRODUCTION

Artificial intelligence (AI) is taking over the world by providing its services in many ways by making the world a better place to live. We have been using artificial intelligence in everyday life for shopping online, streaming videos, smart houses, automation, agricultural fields, health industries, etc. In America, the leading cause of death is due to heart disease, this could be due to many factors like lifestyle, genetics, food habits, etc. It is important to spread awareness about cardiovascular diseases (CVD) and help people lead better life by predicting their risk of getting CVD. “Healthy Heart?” is a web application that will use AI to predict the risk of a person having a heart disease by taking a few major parameters that cause CVD as an input.

2. RELATED WORK (BACKGROUND)

Cardiovascular diseases (CVDs) are the leading cause of death globally. According to projections, 17.9 million deaths globally in 2019—or 32% of all fatalities—were caused by CVDs. 85% of these deaths were caused by heart attacks and strokes. Low- and middle-income countries have a high rate of CVD deaths.[2]

The main behavioral risk factors for heart disease and stroke include poor eating habits, inactivity, cigarette use, and alcohol abuse. Many people experience elevated blood pressure, elevated blood glucose, and elevated blood lipids, as well as overweight and obesity due to these factors. These "intermediate risk factors" indicate an increased risk of consequences like heart attack, stroke, and heart failure and can be assessed in primary care settings.[2]

It has been established that lowering the risk of cardiovascular disease entails giving up smoking, reducing salt intake, increasing fruit and vegetable consumption, exercising frequently, and avoiding dangerous alcohol usage. Health policies that support environments where healthy options are both affordable and accessible are essential if people are to acquire and maintain healthy behaviors.[2]

By identifying those who are most vulnerable to CVDs and ensuring they receive the right care, premature deaths can be prevented. Access to noncommunicable disease drugs and core health technology in all primary healthcare facilities is essential to ensuring that people in need receive care and counseling.[4]

We are grateful to the WHO for providing the information we needed for our background investigation.

3. MODEL

i. ARCHITECTURE DIAGRAM

The architecture consists of 2 main modules

1. User: The user opens the webpage and enters his/her details based on the questions asked. Once the user enters all his details and clicks on continue button, the model predicts the output and displays it on the webpage.

2. Flask Application: The flask application consists of 3 main parts

    a. Web Pages: The home page and result page are hosted by the flask.
    b. Data: The existing data is collected to train the model
    c. Model: The model is trained using the Data, random forest classifier is used. It is compiled and executed using flask run command and we use it later for predictions. Once the

System Architecture Diagram:

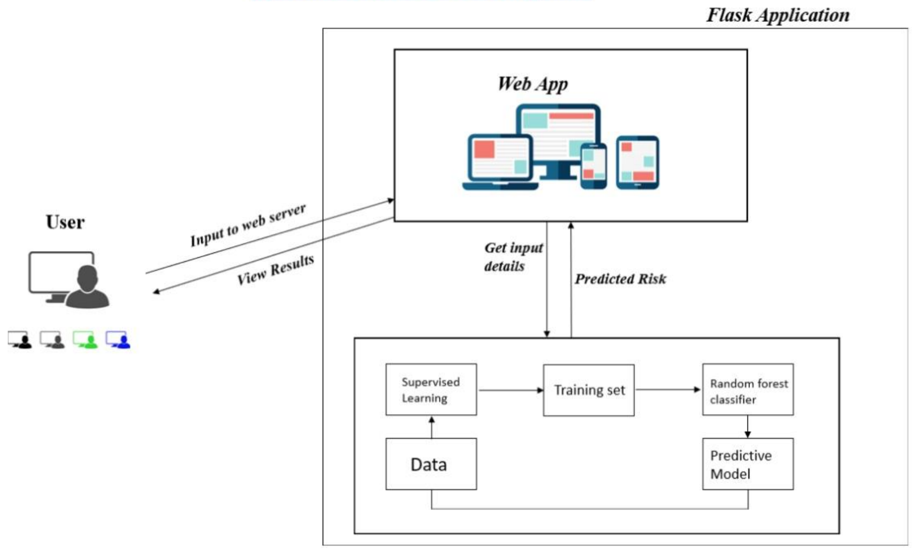


Figure 1: System Architecture Diagram

ii. WORKFLOW DIAGRAM

Workflow Diagram

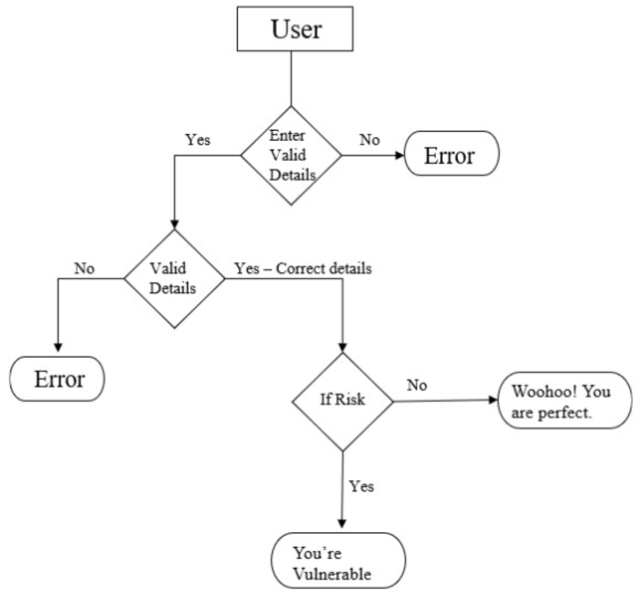


Figure 2: Workflow Diagram

Firstly, the web application is compiled and hosted, the user opens the URL and enters details.

- If entered details are missing, error is thrown.
- If all details are entered and few are invalid, error is thrown.
- If all the details are entered correctly and continue button is clicked, model will predict the risk for the user based on the inserted details.
- If predicted output indicates risk, “You’re vulnerable” message is popped.
- If predicted output does not indicate risk, “Woohoo! You are perfect” message is popped.

4. DATASET

i. DETAILED DESCRIPTION OF DATASET

Cleveland residents were screened between 2000 and 2006 to assess CVD risk, identify high-risk individuals, understand the risk factors associated with CVD, and create a cohort for follow-up. Most of the methods used in the survey were questionnaires, along with physical measures, lab tests, and other techniques. Every two years, blood was drawn, and socioeconomic information was gathered once a year during follow-up.

The 76 attributes in this database have been reduced to the top 14 for consideration.

To this day, ML researchers in particular use the Cleveland database. The "target" field alludes to the patient's having heart illness. It is an integer value between zero and one (presence). Investigations into the Cleveland. [1]

Selected Attributes:

Attribute	Description
age	Age in years(int)
height	Height in Cms (int)
weight	Weight in Kgs (float)
gender	Gender (1=Male 2=Female) (categorical code)
sys_bp	Systolic blood pressure (int)
dia_bp	Diastolic blood pressure (int)
cholesterol	Cholesterol (1: normal, 2: above normal, 3: well above normal)
glucose	Glucose (1: normal, 2: above normal, 3: well above normal)
smoke	Smoking (binary)
alco	Alcohol intake (binary)
active	Physical activity (binary)
CVD	Presence or absence of cardiovascular disease (binary)

The selected 14 attributes cover Quantitative data, Categorical data, and Binary data.

ii. Detail design of Features with diagram

Here in this project, we are trying to implement a Random Forest algorithm. As tree splits are unaffected by scaling, the Random Forest Classifier is unaffected.

With cross-validation, hyperparameter tuning was carried out. A compromise between consistency and runtime, the number of trees produced stable results at around 300 trees, and 500 trees were ultimately chosen.

Although more tests (for tree size and parameters) were run than are currently in the code, they were omitted for readability and runtime considerations.

With the help of the random forest algorithm, the current study forecasts a patient's risk of developing heart disease. If the risk factor indicates 1, it is concluded that the patient is likely to develop CVD. If the risk factor indicates 0, it is concluded that the patient has a healthy heart. 14 attributes totaling 69000 data samples were collected to predict heart disease. Eighty percent of the dataset was used for training, and twenty percent was used for testing.

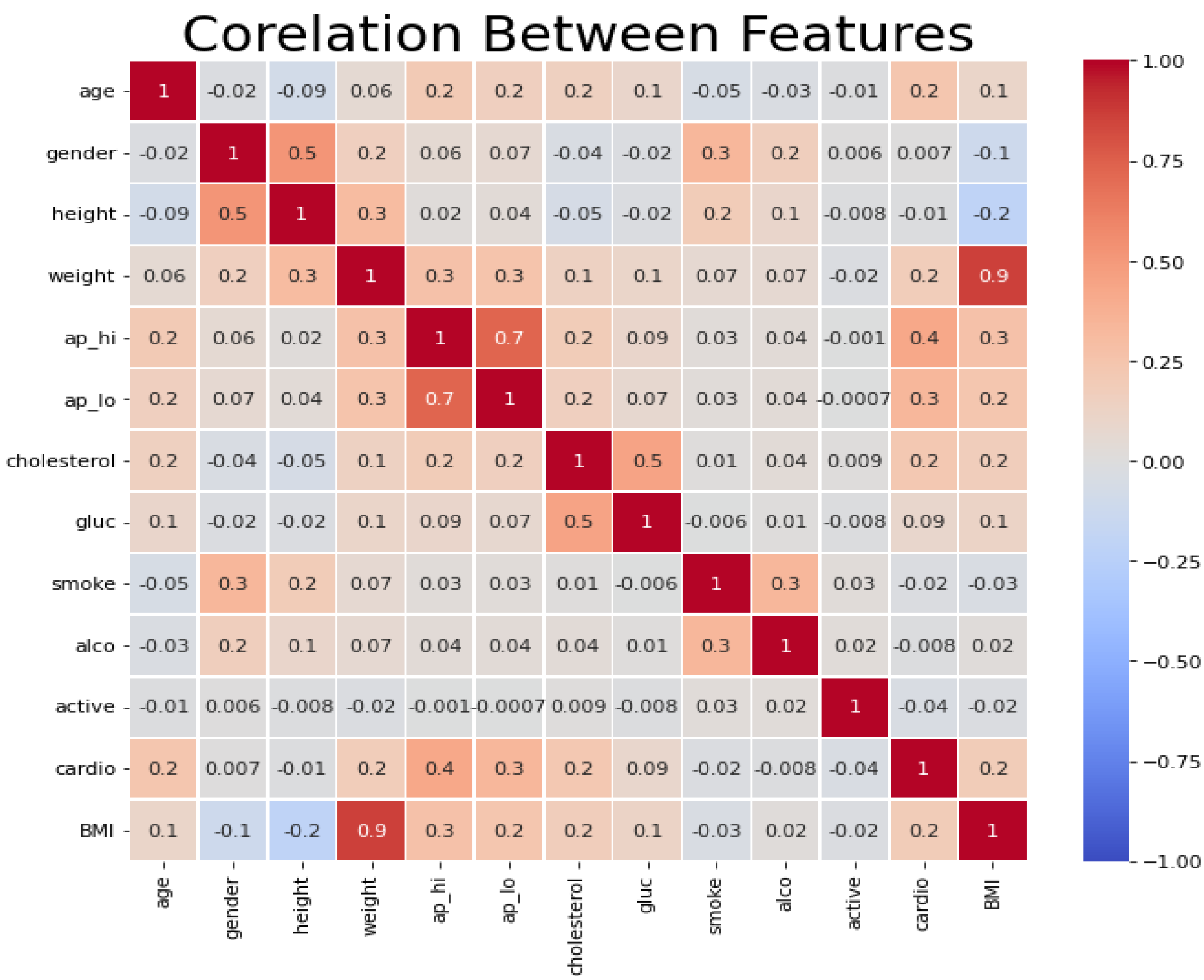


Figure 3: Features Correlation Matrix

	id	age	gender	height	weight	sys_bp	dia_bp	cholesterol	glucose	smoke	alco	active	target
0	0	18393	2	168	62.0	110	80	1	1	0	0	1	0
1	1	20228	1	156	85.0	140	90	3	1	0	0	1	1
2	2	18857	1	165	64.0	130	70	3	1	0	0	0	1
3	3	17623	2	169	82.0	150	100	1	1	0	0	1	1
4	4	17474	1	156	56.0	100	60	1	1	0	0	0	0
5	8	21914	1	151	67.0	120	80	2	2	0	0	0	0
6	9	22113	1	157	93.0	130	80	3	1	0	0	1	0
7	12	22584	2	178	95.0	130	90	3	3	0	0	1	1
8	13	17668	1	158	71.0	110	70	1	1	0	0	1	0
9	14	19834	1	164	68.0	110	60	1	1	0	0	0	0

Figure 4: Raw Data

We obtained the correlation matrix after performing exploratory data analysis, which correlates the attributes of the data set.
On the testing data set, we are using the random forest algorithm to generate a confusion matrix.

More sophisticated metrics, such as sensitivity, specificity, and AUC, are obtained from the confusion matrix and can aid in our decision-making during the classification process.

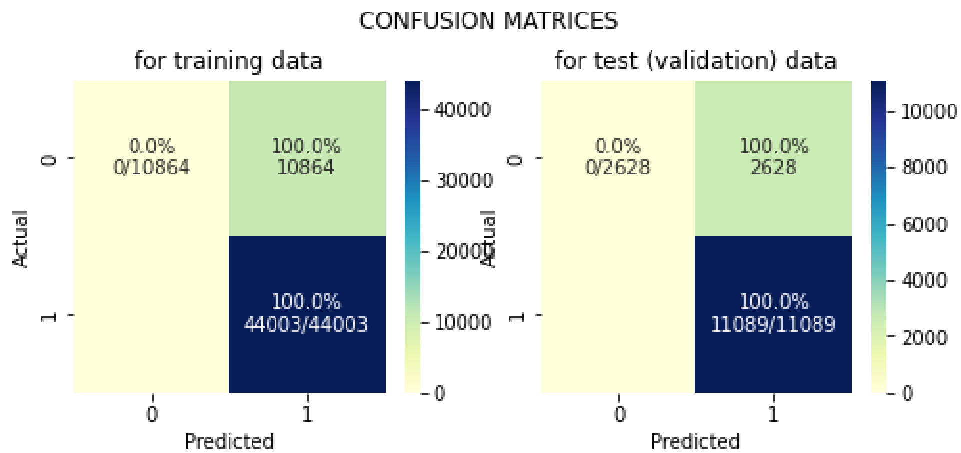


Figure 5: Confusion Matrix

5. ANALYSIS OF DATA

i. DATA PRE-PROCESSING

a. *Exploratory Data Analysis*

Performing the Analysis on the raw data that is exported from the data source. We have observed that there are few duplicate rows detected in the analysis, which is around 25000. Finally, there are no duplicates.

```
[5] # printing the length of the dataframe
    length = df.shape[0]*df.shape[1]
    print('Length of the data frame : ', length)

    # printing missing values
    missing_vals = df.isna().sum().sum()
    print('Missing values: ', missing_vals, '\n')

    # checking for duplicates
    df_dup = df.duplicated().sum()
    if df_dup:
        print('Duplicates Rows : {}'.format(df_dup))
    else:
        print('No duplicates')

    Length of the data frame :  909974
    Missing values:  0

    No duplicates
```

Figure 6: Exploratory Data Analysis

b. *Profile of the Data*

Observing that the age is being considered concerning the number of days. That must be changed into years by dividing the value by 365. Systolic blood pressure "sys_bp" and Diastolic blood pressure "dia_bp" cannot be negative

	age	gender	height	weight	sys_bp	dia_bp	cholesterol	glucose	smoke	alco	active	target
count	69998.000000	69998.000000	69998.000000	69998.000000	69998.000000	69998.000000	69998.000000	69998.000000	69998.000000	69998.000000	69998.000000	69998.000000
mean	19468.892454	1.349567	164.359296	74.205639	128.817109	96.630747	1.366839	1.226464	0.088131	0.053773	0.803737	0.499700
std	2467.272520	0.476837	8.210060	14.395955	154.013595	188.475211	0.680228	0.572277	0.283487	0.225571	0.397172	0.500003
min	10798.000000	1.000000	55.000000	10.000000	-150.000000	-70.000000	1.000000	1.000000	0.000000	0.000000	0.000000	0.000000
25%	17664.000000	1.000000	159.000000	65.000000	120.000000	80.000000	1.000000	1.000000	0.000000	0.000000	1.000000	0.000000
50%	19703.000000	1.000000	165.000000	72.000000	120.000000	80.000000	1.000000	1.000000	0.000000	0.000000	1.000000	0.000000
75%	21327.000000	2.000000	170.000000	82.000000	140.000000	90.000000	2.000000	1.000000	0.000000	0.000000	1.000000	1.000000
max	23713.000000	2.000000	250.000000	200.000000	16020.000000	11000.000000	3.000000	3.000000	1.000000	1.000000	1.000000	1.000000

Figure 7: Data Profile

If sys_bp and dia_bp are more than 180 and 120 mmHg respectively then it is a hypertensive crisis, which is an emergency case. Therefore, max values are not realistic.

c. Outlier detection - sys_bp, dia_bp

Outlier Detection in systoli blood pressure and diastolic blood pressure columns

```
[ ] # Determining the outliers
    outliers = len(df[(df["sys_bp"]>=280) | (df["dia_bp"]>=220) | (df["dia_bp"] < 0) | (df["sys_bp"] < 0) | (df["sys_bp"]<df["dia_bp"])])

    print(f'total {outliers} outliers')
    print(f'percent missing: {round(outliers/len(df)*100,1)}%')

    total 1275 outliers
    percent missing: 1.8%
```

According to the graph below, it can be assumed that values for sys bp and dia bp that are greater than 280 mm Hg and 120 mm Hg, respectively, will be eliminated as outliers. They cannot have negative values, so they also have positive values.

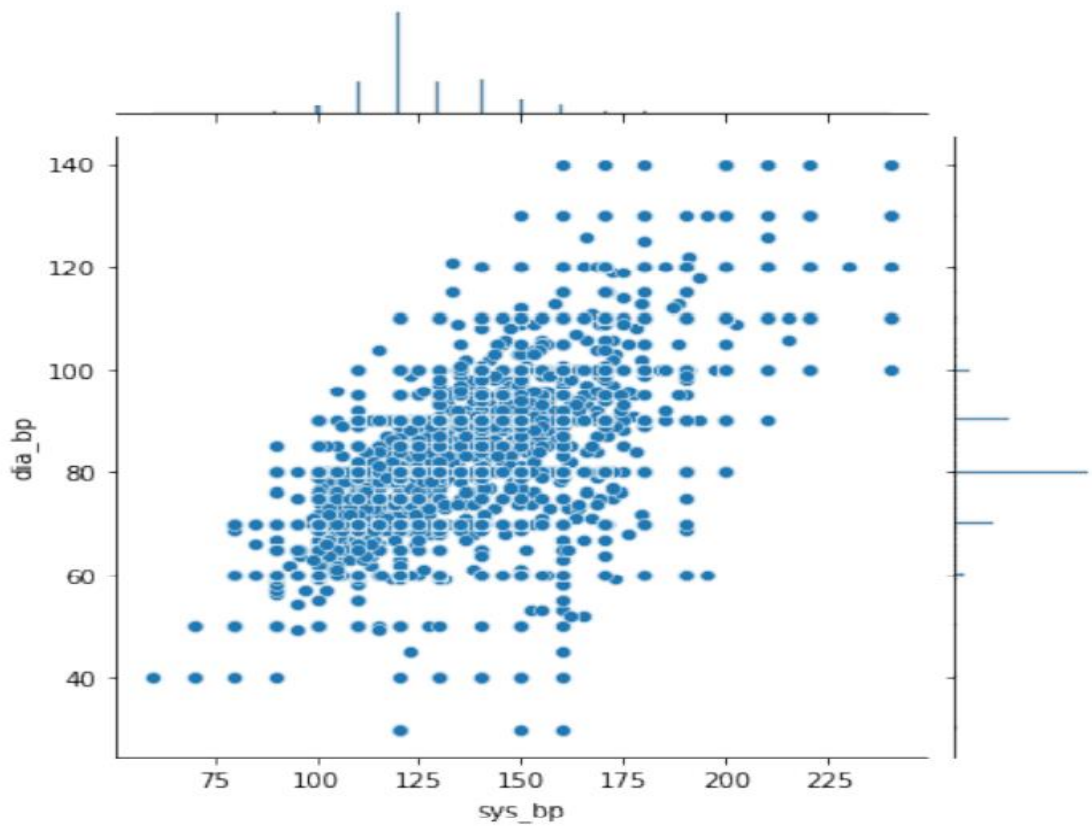


Figure 8: Outliers in systolic and diastolic blood pressures

Note:

- Pulse pressure can not be negative which is the difference between the Systolic and diastolic blood pressures.
- Systolic blood pressure and Diastolic blood pressure cannot be negative
- Values larger than 180 mm Hg and 120 mm Hg for ap_hi and ap_lo respectively are an hypertensive crisis, which is in an emergency case

1. **Systolic blood pressure** is the pressure when the heart beats – while the heart muscle is contracting (squeezing) and pumping oxygen-rich blood into the blood vessels.
2. **Diastolic blood pressure** is the pressure on the blood vessels when the heart muscle relaxes.

d. IQR filtering - height, weight

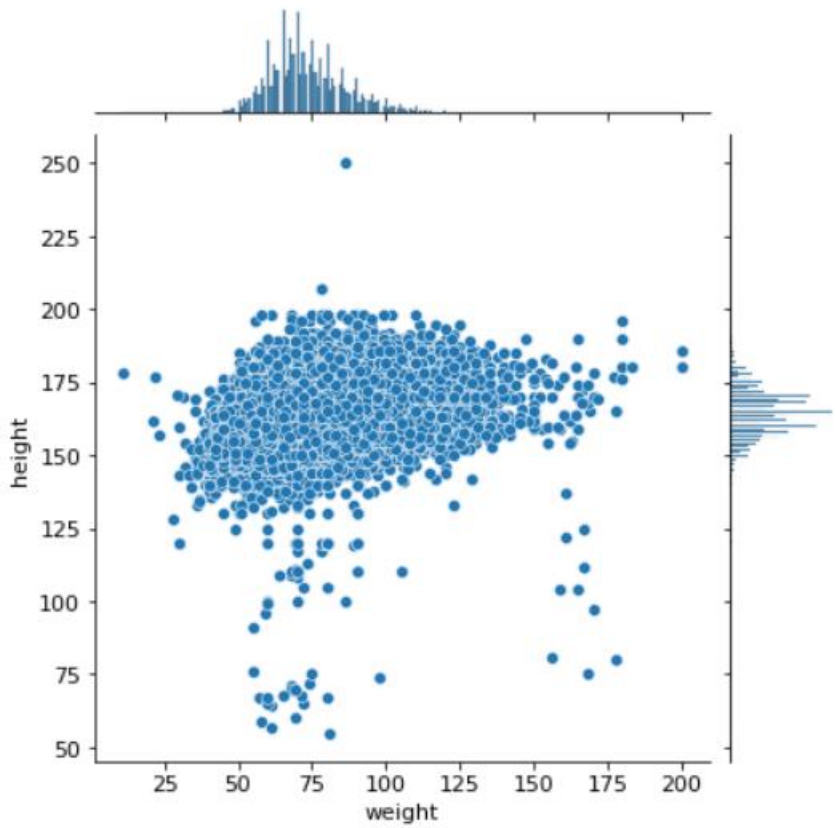


Figure 9: Outliers in weight and height columns

From the plot, it is observed that the smallest person has been recorded at 54cm and the tallest at 251cm

```
Feature: height
Percentiles: 5th=152.0, 95th=178.0, IQR=26.0
Identified outliers: 46

Feature: weight
Percentiles: 5th=55.0, 95th=100.0, IQR=45.0
Identified outliers: 20
```

After cleaning the outliers, each person's BMI is calculated

	age	gender	height	weight	sys_bp	dia_bp	cholesterol	glucose	smoke	alco	active	target	BMI
0	0.138060	2	168	62.0	110	80	1	1	0	0	1	0	22.0
1	0.151833	1	156	85.0	140	90	3	1	0	0	1	1	34.9
2	0.141543	1	165	64.0	130	70	3	1	0	0	0	1	23.5
3	0.132280	2	169	82.0	150	100	1	1	0	0	1	1	28.7
4	0.131162	1	156	56.0	100	60	1	1	0	0	0	0	23.0

Figure 10: New BMI column addition

e. Scaling the Data

We have several features with various scales. Here, the data will be transformed using the Standardscaler library so that the mean is 0 and the standard deviation is 1. It essentially organizes the data.

```
[ ] #we perform some Standardization
    df_scaled=df_cleaned.copy()

    columns_to_scale = ['age', 'weight', 'sys_bp', 'dia_bp','cholesterol','gender','BMI','height']

    scaler = StandardScaler()
    df_scaled[columns_to_scale] = scaler.fit_transform(df_cleaned[columns_to_scale])

    df_scaled.head()
```

	age	gender	height	weight	sys_bp	dia_bp	cholesterol	glucose	smoke	alco	active	target	BMI
0	-0.434234	1.36696	0.453577	-0.853013	-0.999463	-0.137129	-0.537112	1	0	0	1	0	-1.047946
1	0.309309	-0.73155	-1.063547	0.772519	0.799812	0.923668	2.409079	1	0	0	1	1	1.437583
2	-0.246221	-0.73155	0.074296	-0.711663	0.200053	-1.197926	2.409079	1	0	0	0	1	-0.758931
3	-0.746238	1.36696	0.580004	0.560493	1.399570	1.984465	-0.537112	1	0	0	1	1	0.242988
4	-0.806613	-0.73155	-1.063547	-1.277065	-1.599221	-2.258723	-0.537112	1	0	0	0	0	-0.855269

Figure 11: data Scaling

f. Performing some standardizing using minmaxscaler

The data is standardizes using minmaxscalar. We are using age, weight, sys_bp, dia_bp, cholesterol, gender, BMI and height columns for scaling.

```
#we perform some Standardization using minmaxscaler
df_scaled_mm=df_cleaned.copy()

columns_to_scale_mm = ['age', 'weight', 'sys_bp', 'dia_bp','cholesterol','gender','BMI','height']

mmscaler = MinMaxScaler()
df_scaled_mm[columns_to_scale_mm] = mmscaler.fit_transform(df_cleaned[columns_to_scale_mm])

df_scaled_mm.head()
```

	age	gender	height	weight	sys_bp	dia_bp	cholesterol	glucose	smoke	alco	active	target	BMI
0	0.588076	1.0	0.585106	0.246377	0.277778	0.454545	0.0	1	0	0	1	0	0.231557
1	0.730159	0.0	0.457447	0.413043	0.444444	0.545455	1.0	1	0	0	1	1	0.495902
2	0.624003	0.0	0.553191	0.260870	0.388889	0.363636	1.0	1	0	0	0	1	0.262295
3	0.528455	1.0	0.595745	0.391304	0.500000	0.636364	0.0	1	0	0	1	1	0.368852
4	0.516918	0.0	0.457447	0.202899	0.222222	0.272727	0.0	1	0	0	0	0	0.252049

Figure 12: Data Standardization

g. Data Preparation for training and testing

Using train_test_split we are dividing the data accordingly, using different parameters like random_state, and test_size. Here, in this dataset, we are considering the target column to be predicted by the model. In this case, we'll use the 80:20 split ratio. Thus, 80% of the dataset is used for the

```
X_train shape is  (54865, 12)
X_test shape is  (13717, 12)
y_train shape is  (54865,)
y_test shape is  (13717,)
```

ii. GRAPH MODEL WITH EXPLANATION

The data is distributed as shown in the below graphs.

a. ***Graph model to show distribution of CVD by gender***
We can see the distribution of CVD among Males and Females; we can see that the CVDs are most common among the Males.

Distribution of CVD by gender:

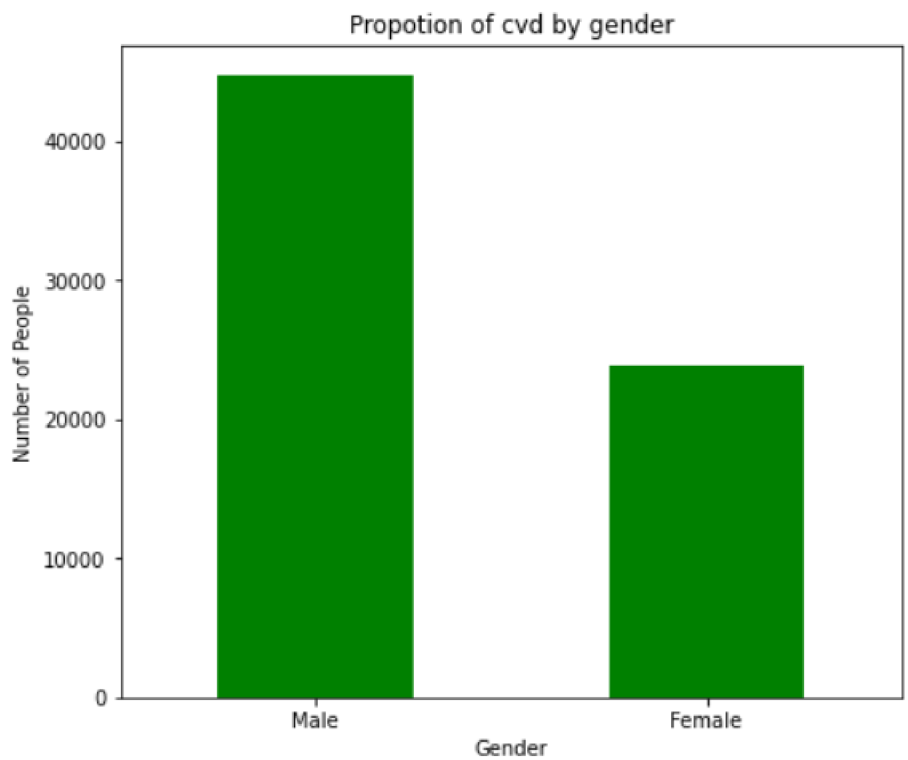


Figure 13: Distribution of CVD by gender

b. ***Graph model to show distribution of CVD by age, gender, height, and weight***
We can see the distribution of CVD based on age, gender, height and weight.

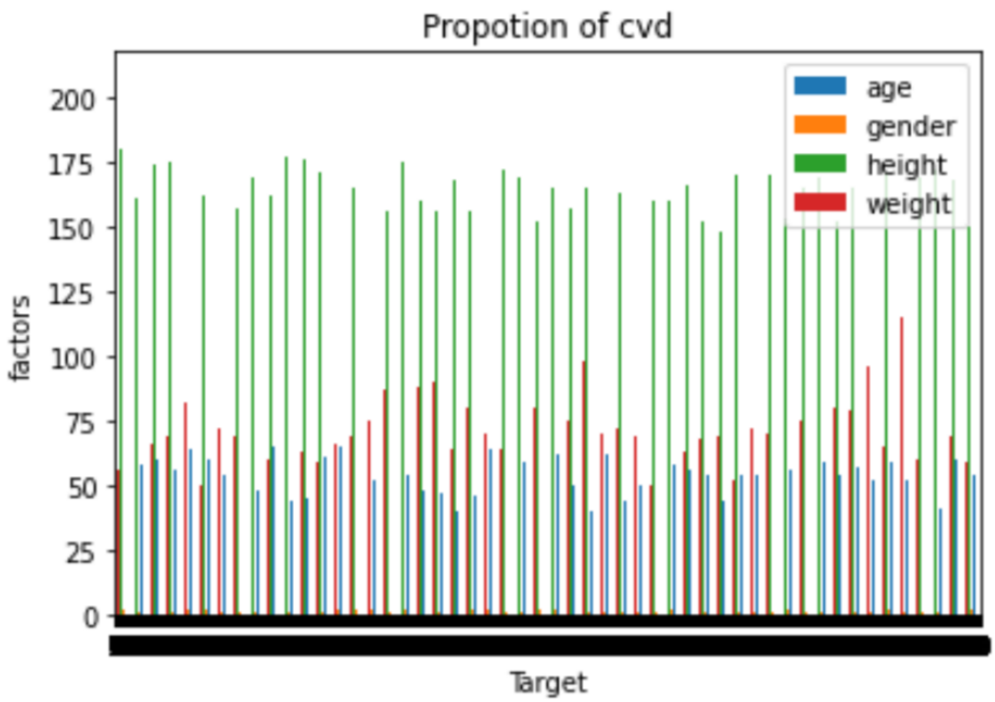


Figure 14: Distribution of CVD based on age, gender, height, and weight.

c. ***Graph model to show distribution of CVD by cholesterol, glucose, smoke, alcohol and active CVD.***

We can see the distribution of CVD based on cholesterol, glucose, smoke, alcohol and active CVD.

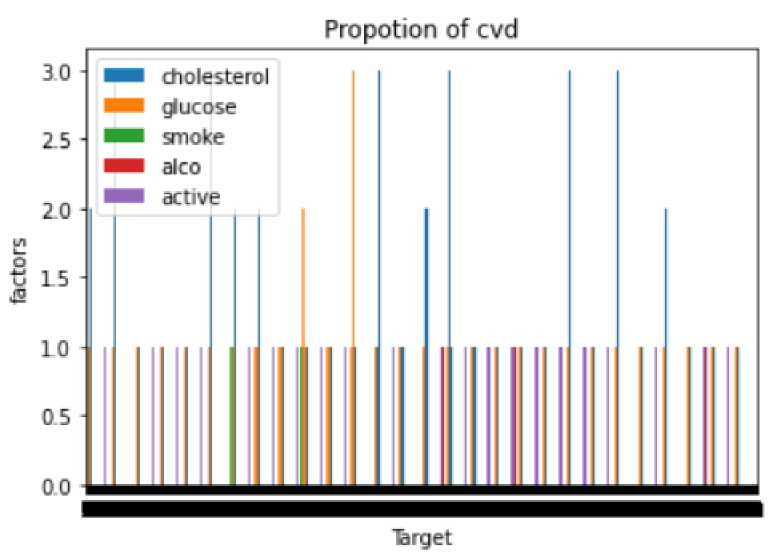


Figure 15: Distribution of CVD based on cholesterol, glucose, smoke, alcohol, and active CVD.

d. ***Outlier detection - sys_bp, dia_bp***

Below is the graph to represent outliers in the data, considering systolic and diastolic blood pressures.

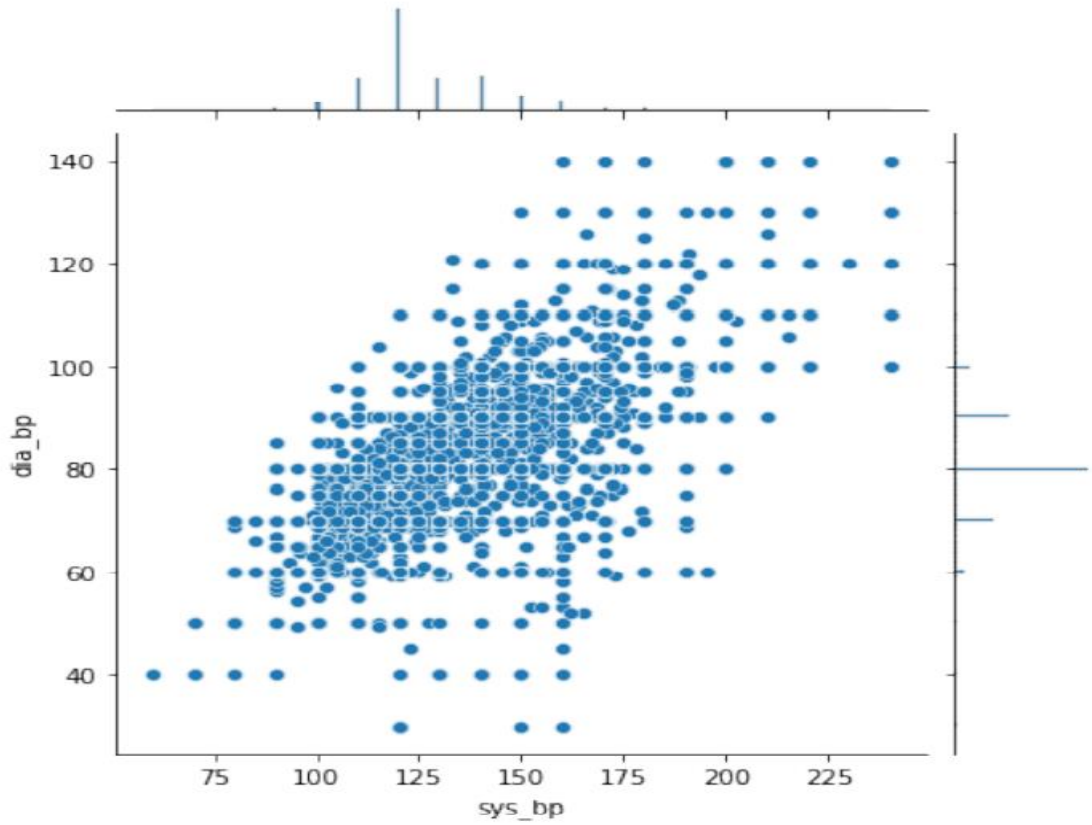


Figure 16: Graph of outliers in systolic and diastolic blood pressures

e. ***IQR filtering - height, weight:***

Below is the graph to identify the outliers in data considering the height and weight columns.

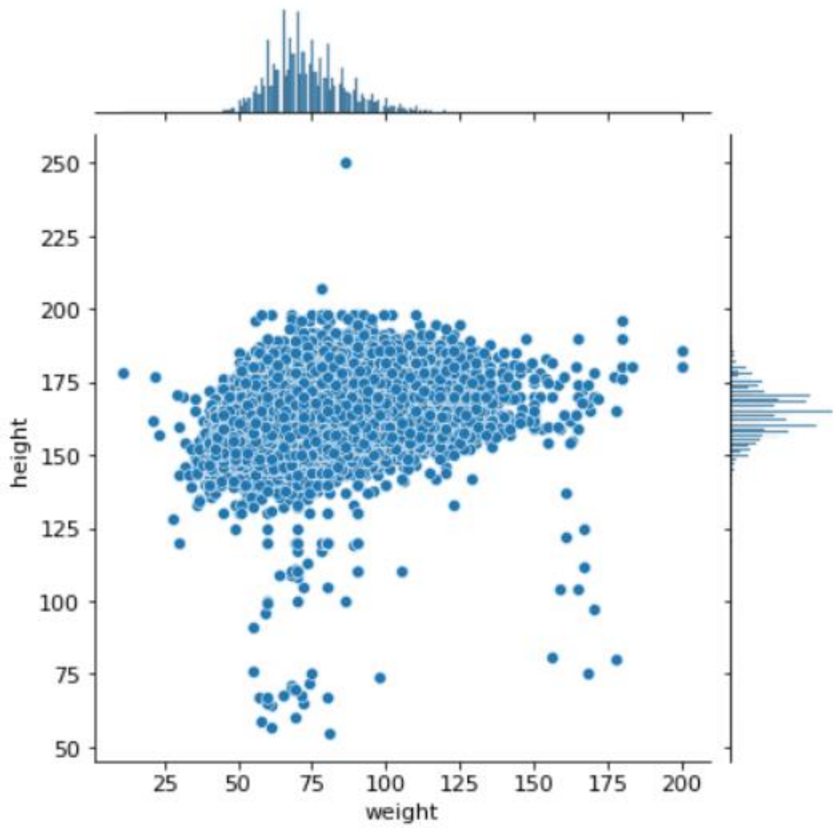


Figure 17: Graph of outliers in height and weight

6. IMPLEMENTATION

After acknowledging the advantages and the accuracy of the Random Forest Classifier, we are implementing this as our ML model.

i. ALGORITHMS / PSEUDOCODE

a. Random Forest Classifier - Model Training

A random forest is a meta-estimator that fits several decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. The sub-sample size is controlled with the max_samples parameter if bootstrap=True (default), otherwise the whole dataset is used to build each tree. By Importing the sklearn library we are importing a random forest classifier.

Using a pre-defined function for this object called GridSearchCV, which is useful to find the best fit combination out of a given, we are performing the model train against it.

Here are the parameters that are given to find the best.

```
rf = RandomForestClassifier(n_estimators=500, max_depth=20, max_features='sqrt', max_leaf_nodes=750)
rf.fit(X_train,y_train)

y_predict = rf.predict(X_test)
y_predicted = np.array(y_predict > 0.5, dtype=float)

rndForest_acc = accuracy_score(y_test, y_predicted)
cm = confusion_matrix(y_test, y_predicted)
rndForest_tpr = cm[1][1] /(cm[1][0] + cm[1][1])
rndForest_report = classification_report(y_test, y_predicted)

print(X_test.columns)
print(X_test.head(3))
```

Figure 18: Model Algorithm

b. Web Application

We are using HTML for developing web-application. A form is developed to take 14 attributes as input from the user.

```
<form id="stripe-login" action="/result" method="POST">
  <div class="field padding-bottom--24">
    <label for="email">Age (In years) </label>
    <input type="text" id="age" name="age">
  </div>
    <div class="field padding-bottom--24">
      <label for="email">Gender (1: Male, 2: Female)</label>
      <input type="text" id="gender" name="gender">
    </div>
    <div class="field padding-bottom--24">
      <label for="email">Height (In Cms)</label>
      <input type="text" id="height" name="height">
    </div>
    <div class="field padding-bottom--24">
      <label for="email">Weight (In Kgs)</label>
      <input type="text" id="weight" name="weight">
    </div>
    <div class="field padding-bottom--24">
      <label for="email">Systolic Blood Pressure</label>
      <input type="text" id="sys_bp" name="sys_bp">
    </div>
    <div class="field padding-bottom--24">
      <label for="email">Diastolic Blood Pressure</label>
      <input type="text" id="dia_bp" name="dia_bp">
    </div>
    <div class="field padding-bottom--24">
      <label for="email">Cholestrol (1: normal, 2: above normal, 3: well above normal)</label>
      <input type="text" id="cholesterol" name="cholesterol">
    </div>
    <div class="field padding-bottom--24">
      <label for="email">Are you Diabetic? (1: Yes, 0: No)</label>
      <input type="text" id="glucose" name="glucose">
    </div>
    <div class="field padding-bottom--24">
      <label for="email">Do you Smoke? (1: Yes, 0: No) </label>
      <input type="text" id="smoke" name="smoke">
    </div>
    <div class="field padding-bottom--24">
      <label for="email">Do you consume Alcohol? (1: Yes, 0: No) </label>
      <input type="text" id="alco" name="alco">
    </div>
    <div class="field padding-bottom--24">
      <label for="email">Do you have any active Cardio Vascular Diseases? (1: Yes, 0: No) </label>
      <input type="text" id="active" name="active">
    </div>


  <div class="field padding-bottom--24">
    <input type="Submit" class="file_submit" name="submit" value="Continue">
  </div>
</form>
```

Figure 19: Input details Form Code

Below code is to display the predicted output.

```
<div class="formbg-inner padding-horizontal--48">
  <center><h2>{{note}}</h2></center>
</div>
```

Figure 20: Result webpage code

c. Prediction Model Algorithm

The inputs from the HTML form are taken and output is predicted by the already trained model. If the predicted output is 1, we consider the user as vulnerable to CVD else we consider user as healthy.

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

@app.route('/result',methods=['POST', 'GET'])
def result():

    d = {'age':[], 'gender':[], 'height':[], 'weight':[], 'sys_bp':[], 'dia_bp':[], 'cholesterol':[],
       'glucose':[], 'smoke':[], 'alco':[], 'active':[], 'BMI':[]}
    output = request.form.to_dict()

    d["age"].append(output["age"])
    d["gender"].append(int(output["gender"]))
    d["height"].append(int(output["height"]))
    d["weight"].append(int(output["weight"]))
    d["sys_bp"].append(int(output["sys_bp"]))
    d["dia_bp"].append(int(output["dia_bp"]))
    d["cholesterol"].append(int(output["cholesterol"]))
    d["glucose"].append(int(output["glucose"]))
    d["smoke"].append( int(output["smoke"]))
    d["alco"].append(int(output["alco"]))
    d["active"].append(int(output["active"]))
    bmi = round(int(output["weight"])/((int(output["height"])*0.0328084)),1)
    d["BMI"].append(bmi)

    df1 = pd.DataFrame(d)
    pred = rf.predict(df1)
    if pred[0]==1:
      note = "You're vulnerable"
    elif pred[0]==0:
      note = "Woohoo! you look perfect"
    else:
      note = "No result"
    return render_template('result.html', note= note)


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

Figure 21: Prediction Model Algorithm

ii. EXPLANATION OF IMPLEMENTATION

1. Firstly, we must install python and then flask in the system we are trying to run the application from.
2. Once installed, download the project zip file, and extract it on Desktop.

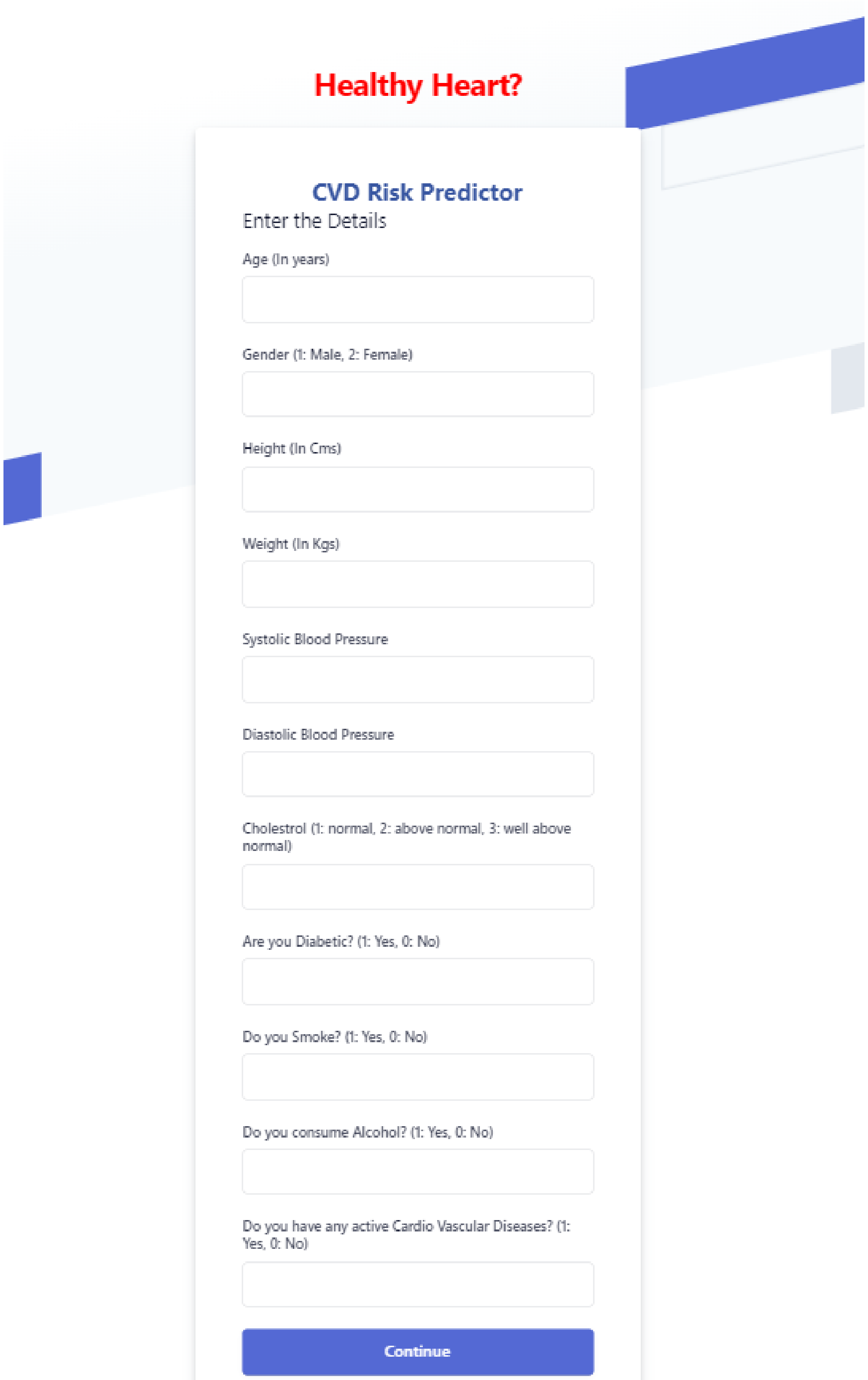
3. From the extracted folder open a terminal and run the following commands.

d. py -m venv env
e. .\env\Scripts\activate
f. flask run

```
PS C:\Users\badda\OneDrive\Desktop\flask_app>  py -m venv env
PS C:\Users\badda\OneDrive\Desktop\flask_app> .\env\Scripts\activate
.\env\Scripts\activate : File C:\Users\badda\OneDrive\Desktop\flask_app\env\Scripts\Activate.ps1 cannot be loaded
because running scripts is disabled on this system. For more information, see about_Execution_Policies at
https:/go.microsoft.com/fwlink/?LinkID=135170.
At line:1 char:1
+ .\env\Scripts\activate
+ ~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : SecurityError: (:) [], PSSecurityException
    + FullyQualifiedErrorId : UnauthorizedAccess
PS C:\Users\badda\OneDrive\Desktop\flask_app> Set-ExecutionPolicy Unrestricted -Scope Process
PS C:\Users\badda\OneDrive\Desktop\flask_app> .\env\Scripts\activate
(env) PS C:\Users\badda\OneDrive\Desktop\flask_app> flask run
Index(['age', 'gender', 'height', 'weight', 'sys_bp', 'dia_bp', 'cholesterol',
       'glucose', 'smoke', 'alco', 'active', 'BMI'],
      dtype='object')
             age  gender  height  weight  sys_bp  dia_bp  cholesterol  glucose  smoke  alco  active   BMI
50568  49.495890       2     173    71.0     120      80            3        1      1     0       1  12.5
26002  48.712329       2     170    75.0     100      70            1        1      0     0       0  13.4
29129  43.610959       1     174    74.0     110      80            1        1      0     0       1  13.0
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
```

Figure 22: Run Instructions

4. Flask run command will execute the python code that will train the model for future predictions using the existing data in csv file. It will also host the webpage on local server.
5. Now copy the link on which the application is running and paste it in your web browser.
6. The website is hosted as shown below.



7. Once, we enter all the input values, risk factor is predicted.

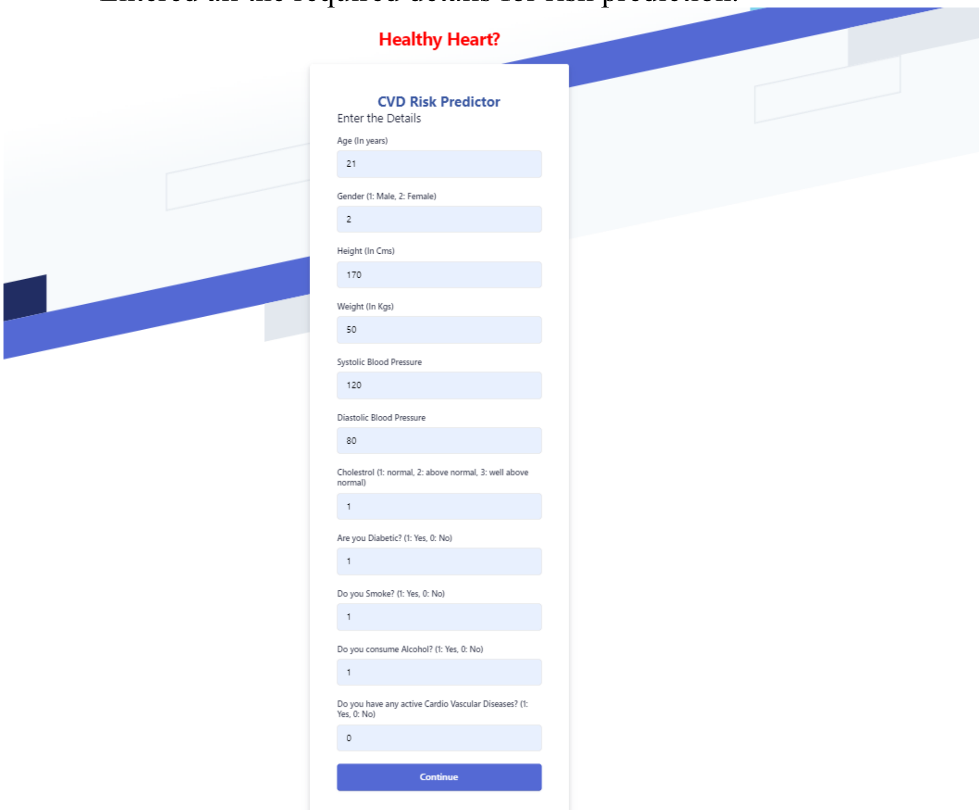
7. Results

i. Application Results

Once the application is hosted. We must enter all the inputs.

<u>Test case:1:</u> Test with healthy person data.

- Entered all the required details for risk prediction.



- Below the risk prediction after submitting details.

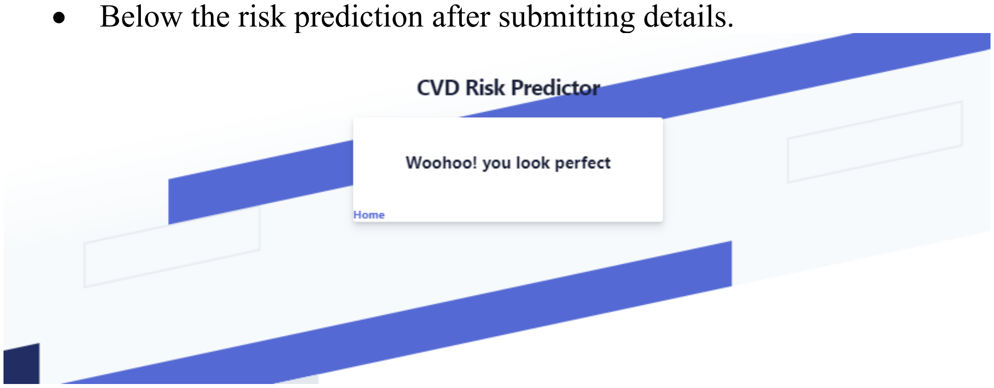
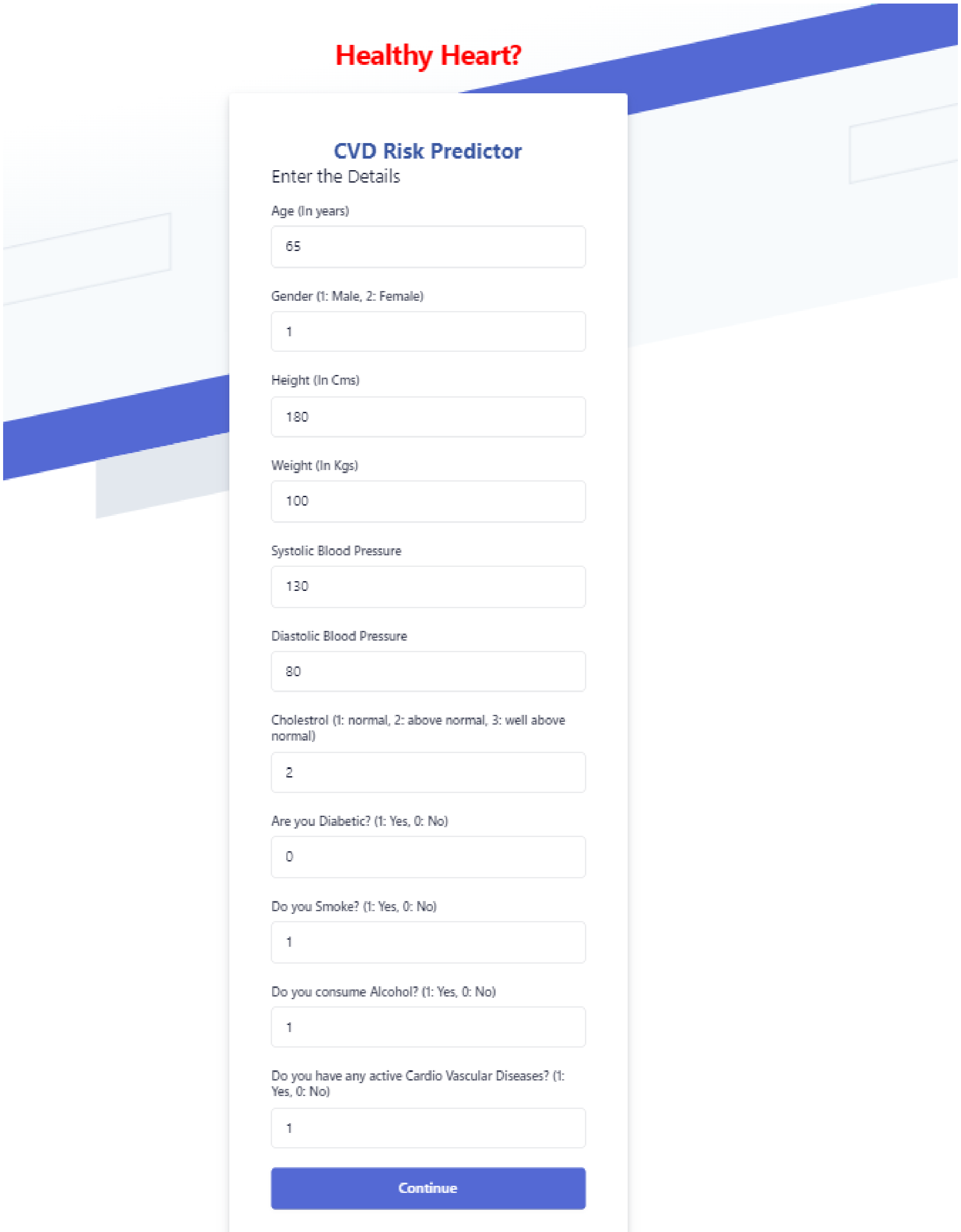


Figure 23: Test Case 1 – Healthy User

- Click on home button to return to the home page.

<u>Test Case.2:</u> Test with Unhealthy patient details.

- Entered all the required fields with below details as shown in screenshot.


Healthy Heart?
CVD Risk Predictor
Enter the Details
Age (In years)
65
Gender (1: Male, 2: Female)
1
Height (In Cms)
180
Weight (In Kgs)
100
Systolic Blood Pressure
130
Diastolic Blood Pressure
80
Cholestrol (1: normal, 2: above normal, 3: well above normal)
2
Are you Diabetic? (1: Yes, 0: No)
0
Do you Smoke? (1: Yes, 0: No)
1
Do you consume Alcohol? (1: Yes, 0: No)
1
Do you have any active Cardio Vascular Diseases? (1: Yes, 0: No)
1
Continue

- Risk prediction for above entered details are as below output.

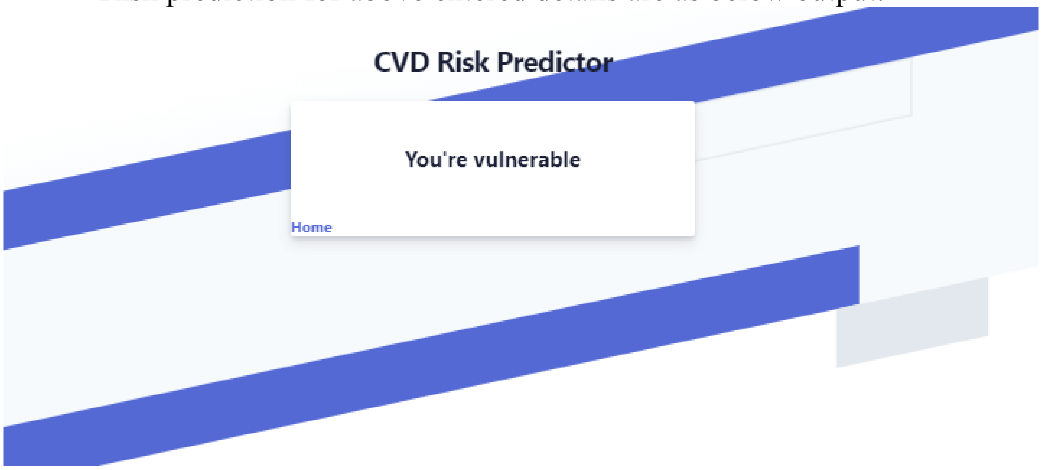


Figure 24: Test Case 2 – Unhealthy User

- Click on home button to return to the home page.

<u>Test Case.3:</u> Test with missing or wrong values

- Few details are not entered in the form.

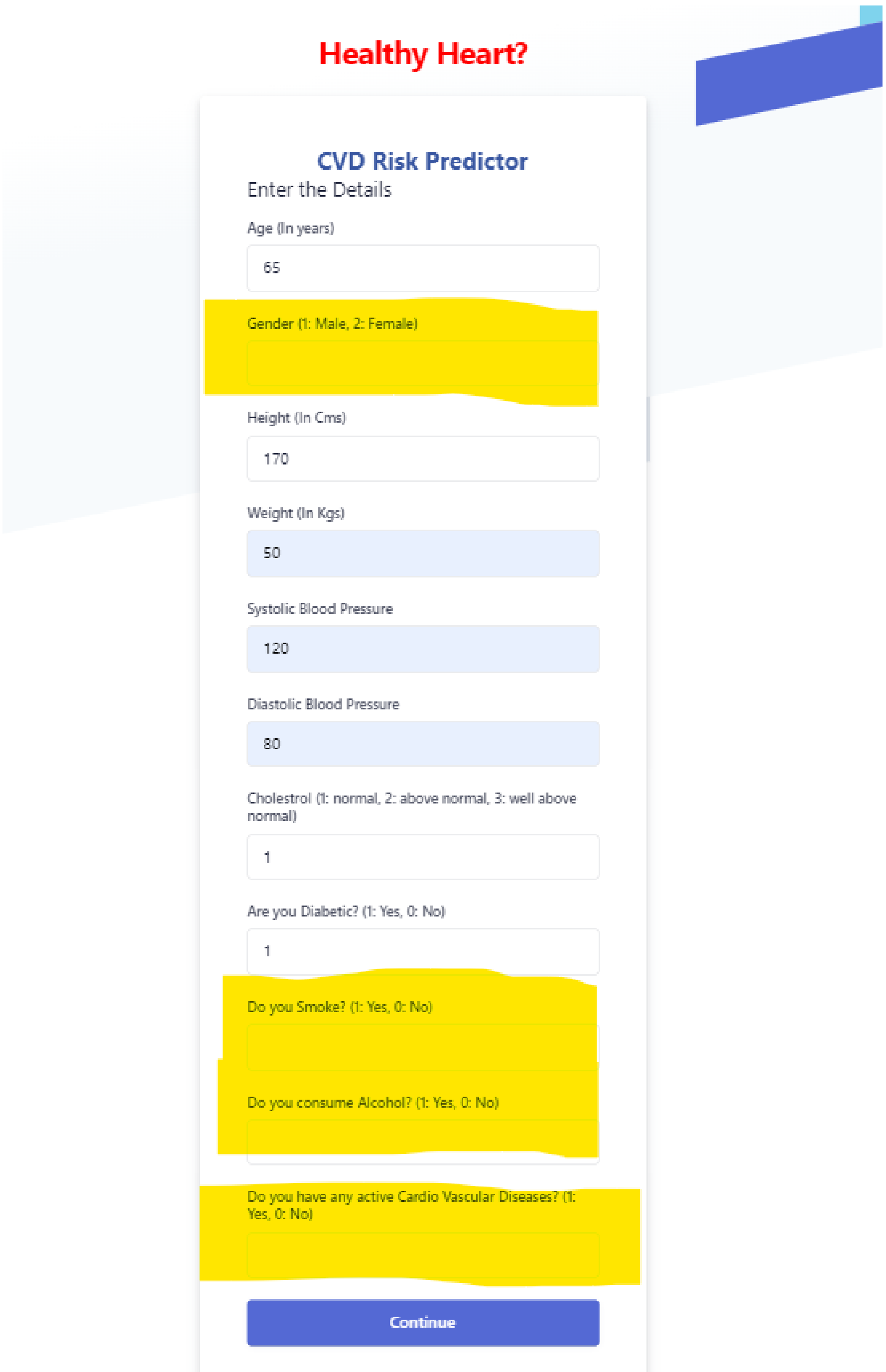


Figure 25: Test case 3 - Missing Values

- Error message displayed as there are missing values in the form.

Internal Server Error

The server encountered an internal error and was unable to complete your request. Either the server is overloaded or there is an error in the application.

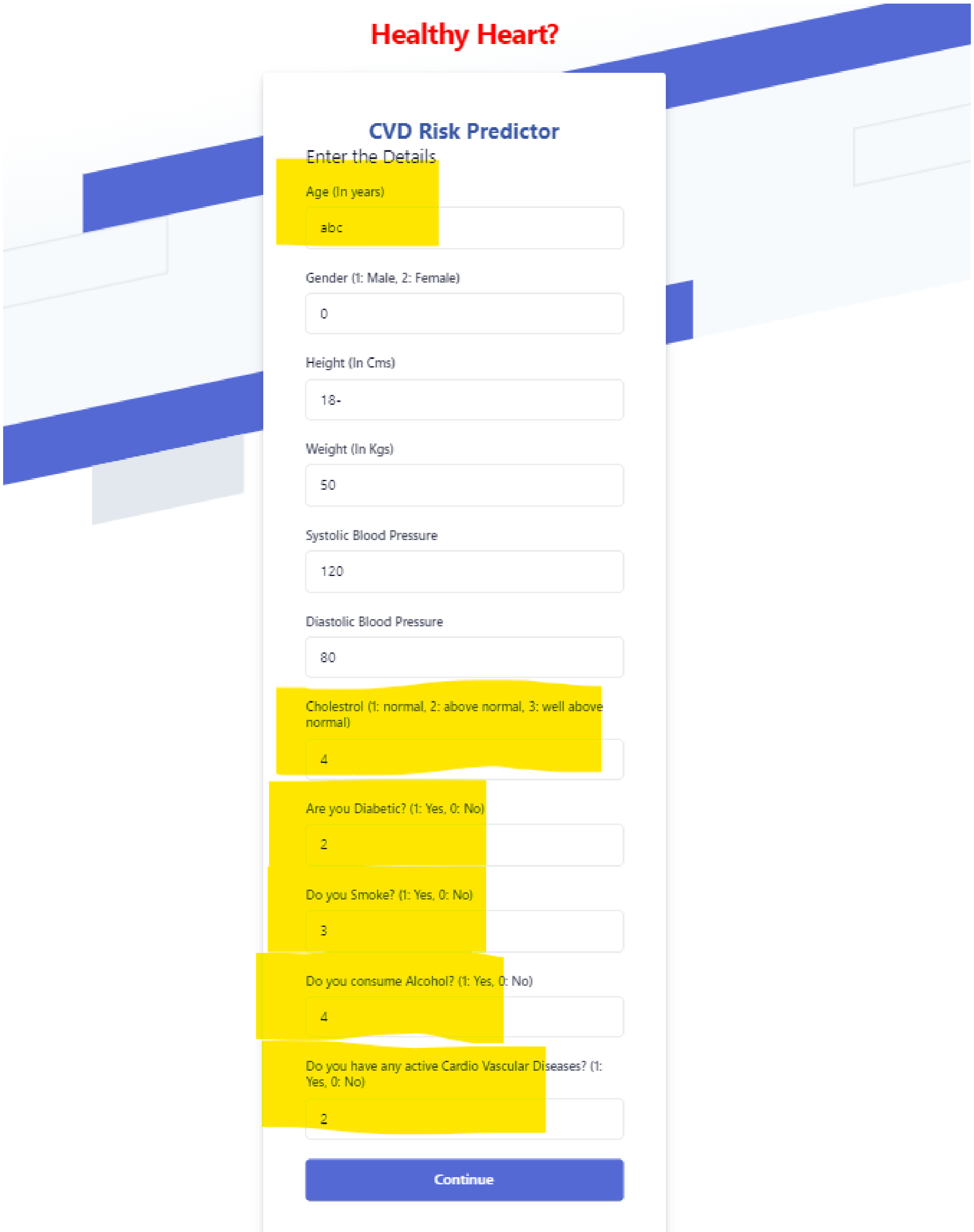


Figure 26: Test Case 4 - Incorrect details

Internal Server Error

The server encountered an internal error and was unable to complete your request. Either the server is overloaded or there is an error in the application.

ii. ALGORITHM PERFORMANCE OUTPUT

Here is an outcome that Grid Search has thrown for the best estimator. These Parameter values will be used to train the actual model to get the best output.

```
Accuacy:  0.7333236130349202
True Positive Rate:  0.6963490650044524
Report:               precision    recall  f1-score   support

           0       0.72      0.77      0.75      6979
           1       0.74      0.70      0.72      6738

    accuracy                           0.73     13717
   macro avg       0.73      0.73      0.73     13717
weighted avg       0.73      0.73      0.73     13717

True Positive :  4692
True Negative:  5367
```

Figure 27: Algorithm performance output

After performing the training with the outcome, the above picture shows the results. We can find the best output would occur at max-leaf nodes = 400

8. PROJECT MANAGEMENT

i. IMPLEMENTATION STATUS REPORT

Work completed:

Description: We have used the already-created model in Increment-1 to create a website, which will take input from the users. We also need to host the website. Once done with coding, we also need to test the application to predict accurate risk factors.

Responsibility (Task, Person):

a. Neha Goud Baddam: Designing a web application using a flask that will take user inputs (all 14 attributes) and generate risk factor as output.
b. Reshmi Chowdary Divi: Documentation.
c. Purandhara Maharshi Chidurala: Testing the web application and hosting the website.

Contributions (members/percentage):
a. Neha Goud Baddam: 40%
b. Reshmi Chowdary Divi: 30%
c. Purandhara Maharshi Chidurala: 30%

Issues: No issues as of now. Application is working as expected.

9. References/Bibliography References

1. Cleveland Database: Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: the University of California, School of Information and Computer Science.

2. Cardiovascular Diseases – WHO: cardiovascular diseases World Health Organization. World Health Organization. Available at: https://www.who.int/health-topics/cardiovascular-diseases/ (Accessed: November 20, 2022).

3. Heart Disease Prediction using Artificial Intelligence: Zaibunnisa L. H. Malik, Momin Fatema, Nikam Pooja, Gawandar Ankita, 2021, Heart Disease Prediction using Artificial Intelligence, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) NREST – 2021 (Volume 09 – Issue 04)

4. Mayo Clinic Health System: Created by Mayo Foundation for Medical Education and Research using content from Framingham Heart Study Cardiovascular Disease 10-Year BMI-Based Risk Score Calculator, Framingham Heart Study General Cardiovascular Disease 30-Year Lipid-Based and BMI-Based Calculators, and ACC/AHA Pooled Cohort Equations CV Risk Calculator.