A New Ensenble Learning based Optimal Prediction Model for Cardiovascular Diseases

B. Srınıvasa Rao

Department of Computer Science Engineering, Gokaraju Rangaraju Institute of Engineering and Technology, Hyderabad.

Abstract: The present paperreports an optimal machine learning model for an effective prediction of cardiovascular diseases that uses the ensemble learning technique.The present research work gives an insight about the coherent way of combining Naive Bayes and Random Forest algorithm using ensemble technique. It also discusses how the present model is different from other traditional approaches.The present experimental results manifest that the present optimal machine learning model is more efficient than the other models.

1 Introduction

Many techniques have been designed for the analysis of heart diseases. But there are many problems in complexity analysis and scalability of data [1, 2]. They are not good in predicting accurate values. This may be due to limited data sets or very high complexity [3]. To overcome these limitation, Naive Bayes Algorithm and Random Forest Algorithm has been used together using ensemble technique.Various approaches have been used to analyse the cardiovascular datasets such as Logistic Regression, Support Vector Machine(SVM), Feed Forward Neural Networks(FFNN)[4],which have had varying degrees of success. A limitation of many of these tech-niques is the lack of accuracy and high time complexity problems [5]. Such approaches do not generalize well to models containing large number of features and lesser size of datasets. Therefore, a solution is to use multiple Machine Learning Algorithms and minimise the tradeoff between Time complexity and accuracy.One such approach is Ensemble. It is one of the paradigms of Machine Learning but unlike traditional learning methods two or more Machine Learning Methods are combined. It is a form of learning method where it learns by making the dataset interact with multiple classification algorithms. Here,a brief survey of the ensemble learning methods that have been applied using Naive Bayes and Random Forest is presented.Recently, there have been many advances in the field of machine learning and more specifically ensemble learning. These approaches employ Hybrid Machine Learning Algorithms and have been used to achieve state-of-the- art results across a variety of disciplines such as Image Processing and Data Science [6]. The combination of multiple classification algorithms with approaches has led to the emergence of Ensemble Learning. We briefly discuss these recent advances .The main objective of the present work are: (i)To develop an efficient and quick method to predict the occurrence of a cardio-vascular disease. (ii)To harness the power of ensemble learning by combining Naive Bayes and Ran-dom Forest algorithms using Voting Classifier. (iii)To balance the bias variance trade offs by selecting the appropriate algorithms with appropriate weights. (iv) To make a user friendly application for prediction of the heart disease by taking input of various characteristics in text format.

2 Literature survey

Ensemble Learning is used to combine multiple machine learning algorithms, espe-cially classification algorithms to provide better accuracy and precision. The Python's SciKitLearn package provides the implementation methods for Naive Bayes and Ran-dom Forest Algorithm and is combined using Voting Classifier by giving appropriate weights. Efficient data visualization is done using Matplotlib and Seaborn.Ensemble learning is a process by which two or more classification algorithms are com-bined to produce a new model which has a higher accuracy rate and prediction. Ensem-ble learning is mainly used in the improvement of the performance of a model or reduce the maximum likelihood error or a selection of a bad model, selecting the best features, dimensionality reduction, iterative and intelligent learning and error correction. This paper deals with the implementation of ensemble learning using Naive Bayes Algo-rithm and Random Forest Algorithm.The Neural Network approach is based on learning from its own features through the multi- layered perceptrons powered by the activation function which is usually a sigmoid function. There was a study conducted by SY. Huang, AH Chen, CH. Cheng, PS Hong and EJ.Lin. The classification model was trained by using Learning Vector Quantization Algorithm which is an Artificial Neural Network(ANN) learning Algo-rithm. There are

* Corresponding author: author@e-mail.org

three steps in this process. The first step includes selection of 14 high priority features that are the most important in the dataset which are i.e., chest pain type, no of vessels colored, resting ecg, age, sex, old peak(op), fasting blood sugar, exercise induced angina, slope, max heart rate, trestbps, cholesterol and thal. The second step includes the implementation of Artificial Neural Network(ANN) algorithm for classi-fication [7]. The third step is the development of the prediction system. The accuracy rate of the prediction obtained from the study is about 81%.The problem with this approach was the time to train the algorithm. Since hospital records are vast and might contain millions of rows, it is costly to train the model using Neural Networks as the time taken to train the model is very high and hence not feasiblein real case scenarios. KNN classification is a novel method of classifying with directly using the training set. For the classification of the test data the K value is calculated. The K denotes the number of K-Nearest Neighbors. For every row in the test data, the distances between all the training data are calculated ad sorted by distance [8]. Finally, majority voting is used and class labels are assigned to the test data.This method produced an accuracy of 83.19 at k=15. The accuracy could be im-proved even more by normalizing the data at the time of data pre- processing. But the problem arises when there are multiple dependant features as the algorithm is based on distance based learning. This in turn increases the computational cost. Although the algorithm is robust to noisy data i.e outliers , it is difficult to chose the exact k value. Naive Bayes classifier is a classification algorithm that is based on Bayes Theorem.It is a combination of multiple algorithms belonging to the same family where all the algorithms follow a common principle, i.e each pair of features that are classified is not dependent on each other. Hidden Naive Thomas Bayes is a higher exactness classifi-cation as compared to Naive Bayes, with reference to the dependencies of the feature attributes. HNB is analogous to theorem Classifier wherever the inflexible entangle-ment is removed and inspiration is taken from all options under consideration [9]. in case of Hidden Naive Bayes, parent node is generated for every feature which mixes the links from different options.The accuracy rate and precision obtained after the cross validation reports were very high. This project was an inspiration to include Naive Bayes Algorithm in our model. The only major drawback of this algorithm is the excessive overfitting of the Model.The Support Vector Machine (SVM) is a very effective and powerful classification model that is prevalent in the medical research industry. The ability to learn and gener-alize the features makes SVM a novel technique for classification models especially for the Cardiovascular Disease Prediction (CDP) [10]. The complexity and accuracy de-pends on the kernel function and the number of parameters. The role of Support Vector Machine (SVM) in the Heart Disease Prediction is to optimize the features repeatedly and increase the margin between this target classes for obtaining higher accuracy rate for the prediction model. Support Vector Machines(SVM) are also called large margin classifiers due to the distance or boundary by which they separate the classes. The SVM can implemented using various kernels of which Linear Kernel and Gaussian Kernel are the most promi-nent. Structural Risk Minimization approach was used while using SVM [11]. The main problem arises while choosing a kernel. The variance and bias tradeoffs mustbe taken care during the implementation. The accuracy achieved by this method was recorded about 91.After analysing and evaluating various traditional and modern learning methods, it has been decided to use ensemble learning as the core learning method for the development of the application system [12]. The two algorithms that are used in the system are Naive Bayes Algorithm and Random Forest Algorithm.Naive Bayes is used for faster convergence of the cost function which indeed in-creases the performance speed [13]. Random Forest is used to overcome the problem of overfitting and decrease the variance. characteristics and situation. In case of overfitting the model tries really hard to fit the curve. Although the data points are scattered and are far apart, it tries to cover all the points and hence ends up being overfit. Overfitting generally occurs incase of polynomial models where the degree of the equation is higher. In case of underfitting, the curve is basically a straight line and hence covers very less data points and is not of much use.From the above figure it can be observed that Overfitting changes under different

3 Prediction System for Cardiovascular Diseases

3.1 Innovation and Idea behind the present project work.

The idea behind the project is to develop a system that can reduce the difficulty and stress in predicting the cardiovascular status of a person, In general if a person who is experiencing a chest pain or a person who generally wants to know his/her respective cardiovascular status need to visit a hospital or a doctor, but in this case there might be complications like money and time which would in some cases might stop the respective person following this way.So what if a person could predict his cardiovascular status by simply using an application that can be even operated while staying home, so there should be a system that could remove all these complications before actually being treated. But the accuracy rates needs to be high alongside with low time complexity rates so that the person is not misguided. In-order to satisfy both these conditions this project uses two algorithms namely Naive Bayes classifier and random forest algorithm which are the best fit for the above purposes. These both are further combined using ensemble which helps in producing the best combination by allocating preferences to each of them.The project also contains a comparison chart that also helps in guiding the person the difference between a healthy heart and an unhealthy heart so that the per-son is fully aware of his heart status.

3.2 Purpose of the present project

The purpose of this project is to provide an efficient system that is helpful in prediction of cardiovascular status of a person. The system helps in reducing the time and costs wasted in actual visit of a hospital or doctor before actually identifying the need of treatment. This project gives the best results while not compromising with complexity and accuracy so that accurate and efficient results are produced.

3.3 Scope of the project

The scope of this project is from a single user to multiple users and to hospitals and medical organizations. It can also be used by persons of any age and any gender. It also involves in integrating the best possible algorithms that will provide the best accuracy rates with low time complexity on the other hand [14]. The scope is immense and has uses in many areas of its field. Convenience to users and universal applicability is what the project aims at.

3.4 Present System

The present system has been highlighted as parts in the literature survey. Most of the existing projects had either accuracy or low time complexity i.e the projects could only provide only one of these but not both at the same time, while some of them had failed in handling huge amount of data. So the existing have many functionalities but failed in any one of the aspect which therby led to lesser customer acceptance. However, the limitations of the present system are: No security for user data. No authentication or security provided. Time Complexities are very high.Scalability Problems while dealing with Big Data.

3.5 Desception of Proposed System

This is a system that could reduce the burden in the prediction of cardiovascular disease as shown in fig.3.1. This system removes all the troubles such as time and money involved while going on for a checkup.The user can easily predict the results by simply using his personal computer or a mobile phone or any display device that has the access to internet.

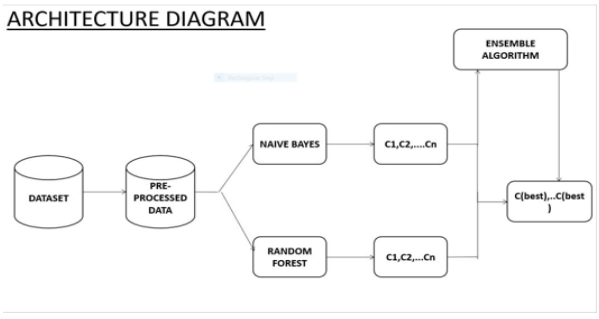

Fig. 3.1: Architecture Diagram

This system uses most efficient and reliable procedures and algorithms to produce quick and accurate results,there are two main algorithms used in this system which are Naive Bayes classifier and Random Forest algorithm where Naive bayes uses proba-bilistic approach by which results can be formed using an individual piece of data from the respective dataset while Random forest algorithm is used for giving the mean value from a given set of data, thereby producing more accurate results.

3.6 Advantages

Instead of completely replacing the existing system innovation comes from incorporat-ing the best parts from each system in order to provide a better system among the lot. This would bring down the difficulty in predicting the cardiovascular status of personwithout the waste of money or time while providing best results.This would allow the user to simply stay in his home or office and predict his cardio-vascular status with out making schedules or appointments to visit a hospital or meet a doctor. The main advantage of this project is that it could produce good accuracy rate with low time complexity and can handle huge amount of data at the same time.This project is well trained based on a dataset that contains huge amount, this project is also observed to accuracy rate of 90 percent which can be considered excellent because it also balances with time complexity and data handling.

3.7 Trade-offs

The existing system is limited to ARFF format whereas the proposed system supports both CSV and graphical format. The existing system uses single classification model whereas the proposed system uses combination of two models. The existing system produces primitive visualisation whereas the proposed sys-tem produces high graphic visualisation using seaborn and matplotlib.

4 Experiment

The software requirements are: (a) Backend: Python 3 (b) Frontend: HTML, CSS and Bootstrap 4. The Hardware requirements are: RAM - advised to have >32GB, Graphic Card, Nvidia GTX 1071, Processor - Intel Core i7-8750H,Storage - 512GB SSD.

5 Data Pre-Processingand Visualization

5.1.1 Data-Import

The project is run on Python platform. The implementation of the algorithms is done by importing the sklearn package. For Naive Bayes, Gaussian Naive Bayes is used for the implementation of the algorithm. Gaussian Naive Bayes, which is generally represented by GNB is nothing but a Naive Bayes algorithm with a Gaussian curve or a Normal distribution curve.The

dataset is obtained from the Cleveland Hospital in the United States.Thedataset contains 337 records with 13 features and the target class consists of 0 and 1. The dataset is open publicly on the kaggle official website.

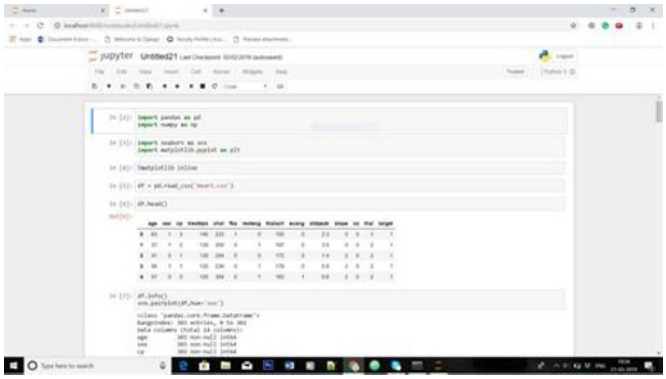

Fig. 5.1: Sample Structure of the dataset

5.1.2 Data Visiulization

The Data Visualisation is done using the python packages Seaborn and Matplotlib which can be imported directly. Seaborn has a higher graphical visualization features as compared to that of matplotlib. The pair plot feature from the Seaborn gives us an overview of the normal distribution of the results. The heatmap is used to find any miss-ing values in the dataset and can be replaced easily either by taking the weighted mean or by replacing them by mode of the column.

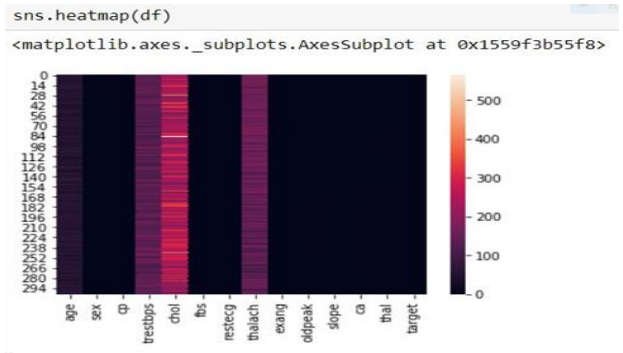

Figure 5.2: Heat Map of the Dataset

5.1.3Data Splitting

The Train Test Split feature from the Scikit Learn is used to split the dataset into training set and test set. The ratio is given as 80:20 where 80% is given for the training set andrest 20% for the test set.

```
predictions = rfc.predict(X_test)
```

```
print(confusion_matrix(predictions,y_test))
print('/n')
print(classification_report(predictions,y_test))
```

```
[[24  3]
 [ 5 29]]
/n
              precision    recall  f1-score   support

           0       0.83      0.89      0.86        27
           1       0.91      0.85      0.88        34

avg / total       0.87      0.87      0.87        61
```

```
from sklearn.ensemble import VotingClassifier
ensemble=VotingClassifier(estimators=[('Decision Tree',gnb), ('Random Forest', rfc)],
                          voting='soft', weights=[1,2]).fit(X_train,y_train)
```

Fig 5.3: Train Test Split

5.2 Algorithm Selectionand Model Fitting

5.2.1 Random Forest Algorithm

The Random Forest Algorithm is similar to Decision trees where the model keeps on di-viding into branches without losing the maintenance of the characteristic values. These characteristic values are the values that are fed to the model as an input. These input values are processed using the random forest classifier to obtain the prediction values.The Random Forest Classifier is imported from the ensemble package which is in-tern inherited from the Scikit Learn package.The Random Forest Classifier contains an argument called n_estimators which decides the number of trees that must be present in the forest. The Random Forest function contains various arguments like Bootstrap, criteria, min impurity, verbose, random_state, oobScore, njobs, n_estimators etc. The main mandatory arguments are classweight, criteria and n_estimators. The fit functionis used by instantiating the object and passing the arguments. Before the instantiation, the n_estimators is used in the Random Forest Classifier. Then X_train and Y_train are passed into the fit function.

```
from sklearn.ensemble import RandomForestClassifier
```

```
rfc = RandomForestClassifier(n_estimators=9800)
```

```
rfc.fit(X_train,y_train)
```

```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=None, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=9800, n_jobs=1,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False)
```

```
predictions = rfc.predict(X_test)
```

```
print(confusion_matrix(predictions,y_test))
print('/n')
print(classification_report(predictions,y_test))
```

```
[[24  3]
 [ 5 29]]
/n
              precision    recall  f1-score   support

           0       0.83      0.89      0.86        27
           1       0.91      0.85      0.88        34

avg / total       0.87      0.87      0.87        61
```

Fig. 5.4: Random Forest Classifier

5.2.2 Naive Bayes Algorithm

The Naive Bayes Algorithm is a data mining technique used to find the relation or the similarity between the characteristics and is used to find the most probable characters that affect the results of the model. The Naive Bayes algorithm is known for its easy implementation and the accuracy rate it gives even for smaller data sets. In this project, Gaussian Naive Bayes algorithm is selected. The Gaussian Naive Bayes algorithm is nothing but a Naive Bayes with a Normal Distribution which has a bell curve. TheGaussian Naive Bayes algorithm can be imported from the scikitlearn package.

```
from sklearn.naive_bayes import GaussianNB

gnb = GaussianNB()

gnb.fit(X_train,y_train)
GaussianNB(priors=None)

predictions = gnb.predict(X_test)

from sklearn.metrics import confusion_matrix,classification_report

print(confusion_matrix(predictions,y_test))
print('/n')
print(classification_report(predictions,y_test))

[[26  5]
 [ 3 27]]
/n
             precision    recall  f1-score   support

          0       0.90      0.84      0.87        31
          1       0.84      0.90      0.87        30

avg / total       0.87      0.87      0.87        61
```

Fig 5.5: Gaussian Naive Bayes

5.2.3 Voting Classifier Using Ensemble

Both the algorithms, i.e. Naive Bayes and Random Forest are combined using a method called Voting Classifier. The Voting classifier can be imported from the Ensemble pack-age. The Voting Classifier has parameters like estimators, n_jobs, weights, n_classes etc. But the main features that are given preference are weights and estimators. The estimators attribute defines the number of classifiers to be combined. In case of the project the number of estimators are 2. The weights attribute decides the level of pref-erence given to a particular classifier. In case of the project, a weight of 1 is given to the Naive Bayes Classifier and 2 is given to the Random Forest Classifier. The Model can be trained using the fit method. Then the predictions can be done using the prediction method and filtering them into a data frame.

```
from sklearn.ensemble import VotingClassifier
ensemble=VotingClassifier(estimators=[('Decision Tree',gnb), ('Random Forest', rfc)],
                          voting='soft', weights=[1,2]).fit(X_train,y_train)

predictions = ensemble.predict(X_test)

C:\Users\ADITYA\Anaconda4\lib\site-packages\sklearn\preprocessing\label.py:151: Deprec
array is ambiguous. Returning False, but in future this will result in an error. Use `
not empty.
  if diff:

print(confusion_matrix(predictions,y_test))
print('/n')
print(classification_report(predictions,y_test))

[[26  4]
 [ 3 28]]
/n
             precision    recall  f1-score   support

          0       0.90      0.87      0.88        30
          1       0.88      0.90      0.89        31

avg / total       0.89      0.89      0.89        61
```

Fig 5.6: Gaussian Naive Bayes

5.2.4 Model Prediction

The accuracy and precision can be obtained from the confusion matrix and classification report. Both the confusion matrix and classification report are imported from metrics. The classification report is the basis for the accuracy of model and how well the model is working for a given data set. The ratio of data given to the X_train and Y_train also effects the accuracy of the model. The classification report contains accuracy, precision and f1 score. The confusion matrix contains the number of false positives and false negatives.

```
print(confusion_matrix(predictions,y_test))
print('/n')
print(classification_report(predictions,y_test))

[[26  4]
 [ 3 28]]
/n
             precision    recall  f1-score   support

          0       0.90      0.87      0.88        30
          1       0.88      0.90      0.89        31

avg / total       0.89      0.89      0.89        61
```

Fig.5.7: Classification Report

5.3 Integrating Model With Front End Using Flask

The Machine Learning model and the Front End Application is integrated and deployed using Python Flask Framework. After importing all the required imports, the classes are instantiated. Flask uses a routing technique to map with the reuired URL. This method is known as object mapping. After the required target URL is fount, page is rendered using render_template method. This method return the requested web page if present with correct syntactical credentials. Otherwise returns a generic error 404 page stating the web page is not found.

```
from flask import Flask,render_template,url_for,request
import pandas as pd
import numpy as np
import pickle
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.externals import joblib
from sklearn.cross_validation import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.ensemble import VotingClassifier
from sklearn.ensemble import RandomForestClassifier


app = Flask(__name__)

@app.route('/')
def home():
    return render_template('home1.html')

@app.route('/predict',methods=['POST'])
```

Fig.5.8: Model integration using Flask-1

Flask is a light-weight web framework used used for deploying web projects. The Pickle package is used to save the trained model so that it is not required to train the model every time when the prediction is needed. The pickle package contains a sub package called joblib. The joblib package contains various functionalities including, saving the model, encryption and decryption of the model, changing the state of the model, retrieval of the previous model and saving the existing model etc. The saved model can be used anytime during the requirement

of a prediction. The predict function in the flask is a user defined function. The data frame is loaded into the application using the read_csv file. The X contains the labelled data and Y contains the target values. Then the data is split into 80:20 ratio using train test split. Then the function is instantiated and the fit function function is used to fit model by passing the X_train and Y_train. Then the predict function is used to predict the model. The model created is saved using the pickle package. The dump function is used to save the model for using it for later prediction purposes. The model is saved and named as "model1.pkl".

```
        my_prediction = hybrid_model.predict(ex1)
        print(my_prediction)
    return render_template('result.html',prediction = my_prediction)


@app.route('/predictquick',methods=['POST'])
def predictquick():
    df = pd.read_csv('Heart.csv')
    X = df.drop('target',axis=1)
    y = df['target']
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
    gnb = GaussianNB()
    gaussian = gnb.fit(X_train,y_train)
    predictions = gaussian.predict(X_test)
    pickle.dump(gaussian, open('model1.pkl','wb'))
```

Fig. 5.9: Model integration using Flask-1

5.4 Using The Jupyter Note Book

Jupyter Notebook is an open source python notebook used for incremental compilations and debugging. It is different from other compilers from the fact that Jupyter Notebook compiles the code section wise and gives the output there itself. The notebook can be used online or can be used as a local host. The directories can be opened directly from the notebook menu and each file is a separate document.

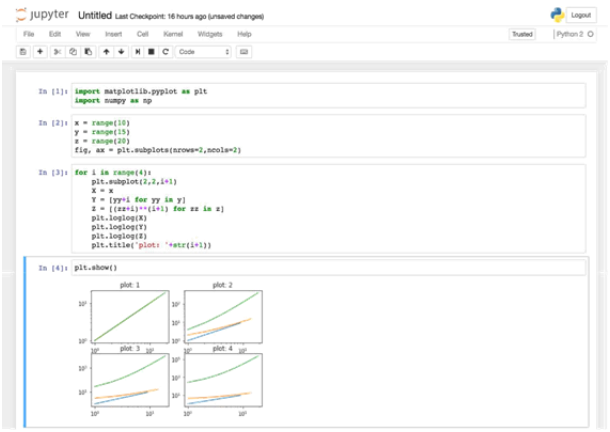

Figure 5.10: Jupyter Notebook File

On the top corner in the jupyter notebook, there is an option for creating a new file. In the drop there are various formats of file. By default it chooses a python format. For executing a section or a block of code shift and enter are to be pressed combined.

Figure 5.11: Creating a new file

Jupyter notebook is directly distributed by anaconda. That means once the anaconda distribution software is downloaded and installed, then the jupyter notebook is automat-ically installed. Hence it supports the processes of clustering of servers in a database.Clustering of a database means a data is divided into sub servers or buckets and every bucket is a assigned some data. In this way even if one of the server fails, there are other servers that are present and have the data replicated data is present. This helps to prevent data redundancy.

6. Results

The model is integrated using the python Flask framework. The python file is run from the anaconda command line prompt. The service can be obtain from the localhost with port 8080. The anaconda is an open source software that contains all the python required files and packages. The python distribution file contains all the distributions as an anaconda parent file. Before executing the "app.py" file, it is required to set the path to the appropriate location. Choosing the local host is mandatory as the code cannot be directly induced as a production code. After the debugging, the service is run on the local host with the port 5000.

```
(base) C:\Users\ADITYA\Desktop\Heart Disease>python app.py
C:\Users\ADITYA\Anaconda4\lib\site-packages\sklearn\cross_validation.py:41: DeprecationWarning:
.18 in favor of the model_selection module into which all the refactored classes and functions
of the new CV iterators are different from that of this module. This module will be removed in
  "This module will be removed in 0.20.", DeprecationWarning)
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Restarting with stat
C:\Users\ADITYA\Anaconda4\lib\site-packages\sklearn\cross_validation.py:41: DeprecationWarning:
.18 in favor of the model_selection module into which all the refactored classes and functions
of the new CV iterators are different from that of this module. This module will be removed in
  "This module will be removed in 0.20.", DeprecationWarning)
 * Debugger is active!
 * Debugger PIN: 177-193-830
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Fig.6.1: Connecting to the local host

The home page contains 13 input fields to be filled. All the fields are of the numeric type and are mandatory to be filled. Every field is assigned a variable in the back end. Since the request is of the post type, the response is collected as a JSON object. JSON stands for JavaScript Object Notation. The JSON object is of the string type and hence various string manipulation techniques can be used on the response data.

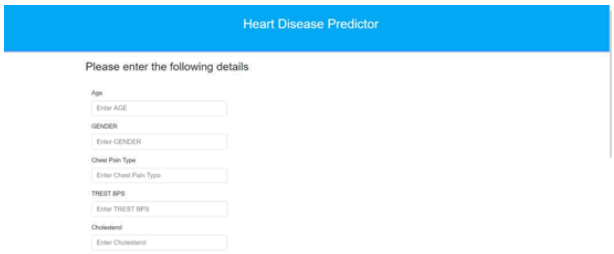

Fig.6.2: Front End Home Page-1

All the values as input are collected as a response in the back end by the flask framework. These 13 characteristic values are taken as an input for the prediction and are predicted using the already trained model. The model that is trained is present in the application as a stored pickle folder. When the service is run for the first time, the model is not built yet and hence needed to be trained. After the model is trained, it is stored as a pickle file and is named as specified by us. Next time, when the application is accessed for prediction, the model need not be trained as the model is already present as a saved file. The characteristic values are directly put into a numpy array. This numpy array is used an input for the prediction using the model.

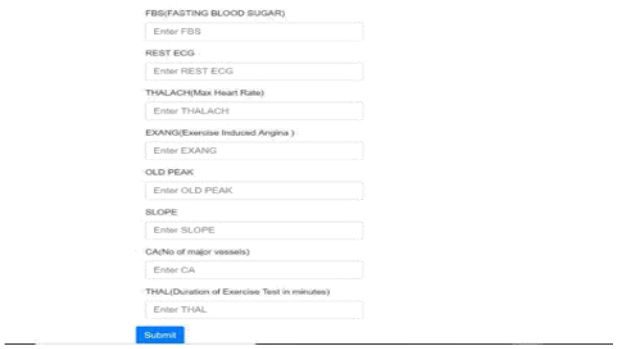

Fig.6.3: Front End Home Page-2

For the given input, after processing the results, the output is obtained in a new web page. If the prediction is positive(with a target 0), then a it is shown in a blue text. Where as if the prediction is a negative prediction, then the result in shown in a red text.

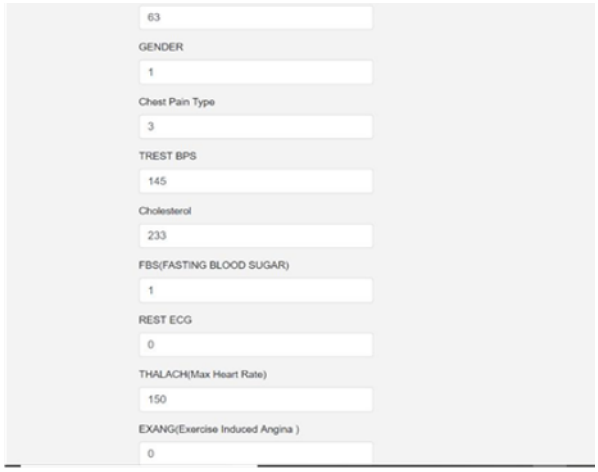

Fig.6.4: Input Values

For the given input, after processing the results, the output is obtained in a new webpage. If the prediction is positive(with a target 0).

Fig. 6.5: Positive Result

For the given input, after processing the results, if the output is obtained in a red text it means the prediction is negative. In this the patient is warned and is advised to meet the doctor as soon as possible.

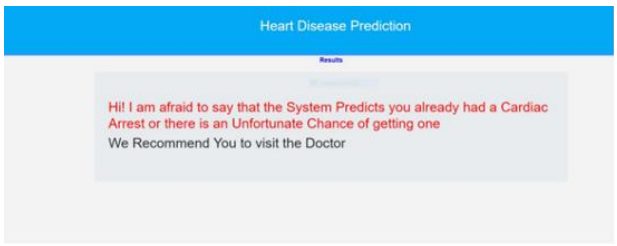

Fig.6.6: Negative Result

Table 6.1: Comparison of Various Approaches

Algorithm	Accuracy	Precision	Variance	Overfitting	Complextiy
SVM	82	82	No	No	Medium
Neural Networks	85	86	Yes	No	High
Naive Bayes	86	86	Yes	No	Less
Random Forest	86	87	No	Yes	Medium
KNN	85	85	No	No	Medium
Logistic Regres-sion	84	86	Yes	No	Less
Combined Approach	89	90	Yes	Yes	Medium

7.Conclusion

It is evident that by using Voting Classifier and combining multiple classification al-gorithms leads to better accuracy and precision. When Gaussian Naive Bayes is used separately, the accuracy is 87% and when Random Forest Classifier is used separately the accuracy is again 87%. But when both the algorithms are combined using Voting Classifier with proper weights, the accuracy increased to 89%. This enhances the power of Ensemble Learning. The accuracy can be improved further by setting the number of estimators to an optimum value. These parameters can be calculated

easily by using the gradient descent method.

In this paper, various approaches to analyse and predict cardiovascular diseases using machine learning algorithms have been discussed. Emphasis is made on the recent breakthroughs in Hybrid Machine Learning, and the contrast between Accu-racy(Precision) and Complexity is also highlighted. Recent advances in combining classification techniques using ensemble which enable to decrease the trade offs be-tween bias and variance have also been discussed. The paper provides a brief overview of techniques to combine classification models and hopes to serve as reference point for further work and study.

References

1. Chauhan, A., Jain, A., Sharma, P., and Deep, V. "*Heart disease prediction using evolutionary rule learning*." (4th International Conference on Computational Intelligence & Communication Technology (CICT), IEEE, 1 2018).
2. Fu, J., Sun, J., and Wang, K. "*Spark–a big data processing platform for ma-chine learning.*" (Industrial Informatics-Computing Technology, Intelligent Technology, Industrial Information Integration (ICIICII), IEEE, 48 2016).
3. Gnaneswar, B. and Jebarani, M. E. "*A review on prediction and diagnosis of heart failure.*" (International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS), IEEE, 1 2017).
4. Hasan, S., Mamun, M., Uddin, M., and Hossain, M. "*Comparative analysis of classification approaches for heart disease prediction.*" (International Confer-ence on Computer, Communication, Chemical, Material and Electronic Engineering (IC4ME2), IEEE, 1, 2018).
5. Jabbar, M. and Samreen, S. "*Heart disease prediction system based on hidden naïve bayes classifier.*" (International Conference on Circuits, Controls, Commu-nications and Computing (I4C), IEEE, 1 2016).
6. Kuo, M.-H., Chrimes, D., Moa, B., and Hu, W. "*Design and construction of a big data analytics framework for health applications.*" (Smart City/SocialCom/Sustain-Com, IEEE, 631 2015).
7. Meena, G., Chauhan, P. S., and Choudhary, R. R. "*Empirical study on clas-sification of heart disease dataset-its prediction and mining.*" (International Con-ference on Current Trends in Computer, Electrical, Electronics and Communication (CTCEEC), IEEE, 1041 2017).
8. Prabakaran, N. and Kannadasan, R. "*Prediction of cardiac disease based on pa-tient's symptoms*." (Second International Conference on Inventive Communication and Computational Technologies (ICICCT), IEEE, 794 2018).
9. Saxena, K., Sharma, R., et al. Pro-cedia Computer Science, **85**, 962 (2016).
10. Selvakumar, P. and Rajagopalan, S. "*SshâATstructure risk minimization based support vector machine for heart disease prediction.*" (2nd International Confer-ence on Communication and Electronics Systems (ICCES), IEEE, 84 2017).
11. Sun, J. and Reddy, C. K. "*Big data analytics for healthcare*." (Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 1525 2013).
12. Suvarna, C., Sali, A., and Salmani, S. "*Efficient heart disease prediction system using optimization technique.*" (International Conference on Computing Method-ologies and Communication (ICCMC), IEEE, 374 2017).
13. Vijayashree, J. and SrimanNarayanaIyengar, N. C. International Journal of Bio-Science and Bio-Technology, **8**, 139 (2016).
14. Zolfaghar, K., Meadem, N., Teredesai, A., Roy, S. B., Chin, S.-C., and Muckian, B. "*Big data solutions for predicting risk-of-readmission for congestive heart fail-ure patients.*" (Big Data, IEEE, 64 2013).