Project : Heart Disease Prediction
Project Group: 21

Abhishek Vyas : MT19086
Amrit Mohan Kaushik : MT18022
Chirag Chawla : MT19089
Megha Mathur : MT19104

Description :

Problem Statement : Nowadays heart related problems are becoming very common like diabetes, high blood pressure, high cholesterol, irregular pulse rates. And we can't predict heart disease based on these heart related problems, so we could not take the important measure to save life which causes the mortality rate to increase..

We could somehow predict the heart diseases based on certain parameters with machine learning algorithms, in its early stages, so that by taking preventive measures we can save life. That's what we are doing in this project, based on certain features which were given to us,we'll predict whether the person is having heart disease or not.

Steps followed in Code:

- The dataset is taken from "UCI Repository".
- First we have read the dataset. The data was given for some 303 patients and had 13 features which are as follows : Age, Sex, Thal, ca, slope, fbs, chol, trestbps, cp , restecg, exang, oldpeak, thalach.
- We had checked for any kind of None values, in our dataset we didn't find any of the None values.
- In the given dataset, the two features named "age" and "sex" are non-medical based features, which may have less impact on the classification. So we will try to apply the classification model with two different approaches :
  - By including the features "age" and "sex" i.e total 13 features.
  - By excluding the features "age" and "sex" i.e total 11 features.
- In order to train our model, we need some train dataset, and in order to test our model we need a test dataset. So we have splitted our dataset into train and test subsets by using sklearn method in the ratio of 70:30 i.e the 70% dataset will be treated as train dataset, and rest 30% of dataset is treated as test dataset.
- Here we are using cross validation techniques i.e. taking 5 fold cross validation in order to improve our results.
- Then we have implemented 9 different models. Here are the 9 models :
  - MULTI-LAYER PERCEPTRON MODEL : In Neural Network, the multi-layer perceptron model is built with various layers. Each layer has it's output stream size and an activation function. In our implementation, there are four layers and the last layer uses "softmax" as an activation function. The other three layers use "relu" as an activation function. This model tries to reduce the loss after each epoch. In our implementation, the "sparse categorical cross-entropy" function is

used for calculating loss and “adam” is used as an optimizer. Each layer passes it’s output as an input to the next layer and the last layer finally produces the labels.

- SVM MODEL : Support vector machine evaluates the hyperplane to separate the data points. The dimension of the hyperplane depends on the number of features. There can be multiple hyperplanes possible to separate the same set of data points, so the svm algorithm finds out the hyperplane which is having maximum margin with the data points.
- LOGISTIC REGRESSION : Logistic regression is suitable for the classification that is having categorical target value. It has the following parameters :
  Output : 0 or 1
  Hypothesis : Z = WX + B (W is weight matrix, and X is Input feature matrix), it is implemented using the Sklearn library.
- RANDOM FOREST : Random Forest is a tree based algorithm which runs a decision tree algorithm over different subsets of training set, and then votes for the class label i.e assign the class label with maximum occurrences from decision tree over different subsets. It is implemented using Sklearn library.
- NAIVE BAYES MODEL : In Naive-Bayes algorithm Bayes theorem and conditional probability are used. From the train data firstly, prior probabilities are calculated. These prior probabilities are finally used to find out the probability of an instance being in a particular class. Whichever class gets a higher probability, the data gets labeled accordingly.
- KNN MODEL : In the KNN model each input instance or row is considered as a vector. With the testing node firstly euclidean distance is calculated from all the nodes. Then find out the k-nearest node. Finally, assign the highest occurring label among those k-nearest nodes.
- DECISION TREE CLASSIFIER : Decision-tree is the classification model that could predict the labels of test data target using input-features provided. Here each internal node is labeled with input features that are being provided and an outgoing edge from that node contains the label of the target. In Decision-Tree, we have used Entropy as a distance metric.
- LANGUAGE MODEL : Here we are using a bigram model, for a particular test record we are finding the probability of class label with respect to each feature based on the bigrams of the train data. For example : P(class = 1 | age = 50) : This Probability can be evaluated by finding the count of the bigram “50 1”, and the unigram “50”, then P(class = 1 | age = 50) = Count(50 1) / Count(50), and in addition to that we have also applied add-1 smoothing technique to prevent the probability value 0.
- HYBRID APPROACH USING LASSO(LINEAR MODEL) AND RANDOM-FOREST : This is a hybrid approach which is named as, Hybrid Random Forest and Linear Model. This model combines the advantages of random forest and linear models. Firstly, a linear model i.e., “Lasso” is used to refine the best features out of all the features present in the dataset. These

features are finally used to implement the random forest algorithm and generate the labels. After finding the set of best features, the Random forest model is trained and finally, labels are generated for test data.

- For each of the models, we have found the accuracy and f-score values and we have taken the mean of the accuracy values from 5 folds to get the final accuracy.

Steps to be followed to execute our Code:

- Open the code in any python editor tool.
- Just set the path for the dataset.
- And then execute the code.