Heart Disease Prediction System Using Multilayer Perceptron and Support Vector Machine

By Megha Suvarna, Mrunal Ghanwat
CS:5100 Foundations of Artificial intelligence
Northeastern University

Abstract

Heart Disease is a leading cause of death worldwide. Mortality rate because of heart-disease has been increased in recent years. It is vital that patients improve their health conditions from time to time. If the patients can understand the severity of heart disease based on their current health conditions beforehand, it would help them improve their health to reduce heart-disease chances in the future. Many health care industries collect an abundant amount of data which includes hidden information which can be used to predict the heart disease and make effective decisions. To make such predictions and decisions we are using machine learning techniques on data. In this project, we are creating heart disease prediction system using a supervised learning technique to classify users into 5 categorized classes as 1,2,3,4 to indicate the severity of the presence of a disease and zero for the absence of heart disease. We are using Support Vector Machine and Multilayer Perceptron to build a classification model to predict the severity of a disease. Using different possible variations of these algorithms, the analysis is generated to choose efficient algorithm for a system.

Introduction

Heart disease death rates have increased for the first time in decades and stroke death rates also have gone up, according to new federal statistics(link opens in new window) that show a drop in U.S. life expectancy. About 610,000 people die of heart disease in the United States every year. Every year about 735,000 Americans have a heart attack. Out of these, 525,000 are a first heart attack and 210,000 happen in people who have already had a heart attack. About 15% of people who have a heart attack will die from it. Almost half of sudden cardiac deaths happen outside a hospital. Heart disease remained the No. 1 killer of Americans in 2015, with stroke staying at the No. 5 spot. Stroke death rates increased from 36.5 deaths per 100,000 Americans in 2014 to 37.6 in 2015. Heart disease death rates have declined since at least 1969 but have plateaued in recent years. Heart disease costs the United States about $200 billion each year. This total includes the cost of health care services, medications, and lost productivity.

A number of factors have been shown that increases the risk of Heart disease:

- Family History.
- Poor diet
- Smoking
- High Blood Cholesterol
- High Blood Pressure
- Physical inactivity
- Hyper tension
- Obesity

About half of Americans (49%) have at least one of these three risk factors. Several other medical conditions and lifestyle choices can also put people at a higher risk for heart disease, including Diabetes,Overweight and obesity,Poor diet,Physical inactivity,Excessive alcohol use.

Many health care industries collect an abundant amount of data which includes hidden information which can be used to predict the heart disease and make effective decisions. Most hospitals today use sort of hospital information systems to manage huge voluminous amounts of patients data. Artificial Intelligence application in healthcare can have tremendous potential and usefulness.

Knowing the warning signs and symptoms of a heart attack are better so that you can act fast if you or someone you know might be having a heart attack. The chances of survival are greater when emergency treatment begins quickly. The diagnosis of heart disease with an automated prediction about the heart condition of the patient can help the doctor and patient know the presence or severity of the heart disease and make further treatment effective. To make such predictions and decisions we are using machine learning techniques on data. In this project, we are creating heart disease prediction system using a supervised learning technique Support Vector Machine and Multilayered Perceptron to classify users into 5 categorized classes 1,2,3,4 to indicate the severity of the presence of a disease and zero for the absence of heart disease.

Background Study

Health care industry today generates large amount of complex date about patients, hospital records, disease diagnosis, electronic patient records, medical devices etc. Large amount of data is a key resources to be processed and analyzed for knowledge extraction that enables support for cost- savings and decision making. Data mining and Machine learning techniques which are applied to medical data include association rule mining for finding frequent patterns, prediction, classification and clustering. Traditionally Machine learning techniques were used in different domains like Wireless sensor network, Weather prediction, sales forecasting etc., However it is introduced relatively late into the health care domain. This has led to the development of intelligent system and decision support system in health care domain for accurate diagnosis of disease, predicting the severity of various disease and remote health monitoring. Especially the Data techniques are more useful in predicting heart disease, lung cancer and several other diseases.There have been a lot of studies and research done that have focus on diagnosis of heart disease. Several machine learning algorithms have been applied and different probabilities for different methods have been achieved.

Supervised learning is the machine learning task of learning a function that maps an input to an output based on example input-output pairs. It infers a function from labeled training data consisting of a set of training examples. In supervised learning, each example is a pair consisting of an input object (typically a vector) and a desired output value (also called the supervisory signal). A supervised learning algorithm analyzes the training data and produces an inferred function, which can be used for mapping new examples. An optimal scenario will allow for the algorithm to correctly determine the class labels for unseen instances.

Support Vector Machine

In machine learning, support-vector machines are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.

In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.

The learning of the hyperplane in linear SVM is done by transforming the problem using some linear algebra. This is where the kernel plays role.

For linear kernel the equation for prediction for a new input using the dot product between the input x and each support vector x_i is calculated as follows: $f(x) = B(0) + sum(ai * (x, x_i))$

This is an equation that involves calculating the inner products of a new input vector (x) with all support vectors in training data. The coefficients $B(0)$ and ai (for each input) must be estimated from the training data by the learning algorithm.

The polynomial kernel can be written as $K(x, x_i) = 1 + sum(x * x_i)^d$ and exponential as $K(x, x_i) = exp(-gamma * sum((xx_i))$.

The Regularization parameter tells the SVM optimization how much you want to avoid misclassifying each training example.

The gamma parameter defines how far the influence of a single training example reaches, with low values meaning far and high values meaning close. In other words, with low gamma, points far away from plausible seperation line are considered in calculation for the seperation line. Where as high gamma means the points close to plausible line are considered in calculation.

And finally last but very importrant characteristic of SVM classifier. SVM to core tries to achieve a good margin. A margin is a separation of line to the closest class points. A good margin is one where this separation is larger for both the classes. A good margin allows the points to be in their respective classes without crossing to other class.

For large values of C, the optimization will choose a smaller-margin hyperplane if that hyperplane does a better job of getting all the training points classified correctly. Conversely, a very small value of C will cause the optimizer to look for a larger-margin separating hyperplane, even if that hyperplane misclassifies more points.

Multilayer Perceptron

Multilayer Perceptron (MLP) is a supervised learning algorithm that learns a function $f(.) : R^m -> R^0$ by training on a dataset, where m is the number of dimensions for input and O is the number of dimensions for output. Given a set of features $X = x_1, x_2, ..., x_m$ and a target y, it can learn a non-linear function approximator for either classification or regression. There can be one or more non-linear layers, called hidden layers between the input layer and output layer.

The leftmost layer, known as the input layer, consists of a set of neurons $x_i|x_1, x_2, ..., x_m$ representing the input features. Each neuron in the hidden layer transforms the values from the previous layer with a weighted linear summation $w_1x_1 + w_2x_2 + ... + w_mx_m$ followed by a non-linear activation function $g(.) : R -> R$

We are using a heart-disease dataset provided by UCI Machine learning Repository. This dataset consists of 4 different .csv files containing records for patients explaining their health conditions. Each record contains 14 attributes and a classification result 0 to 4 - explaining the severity of heart disease.

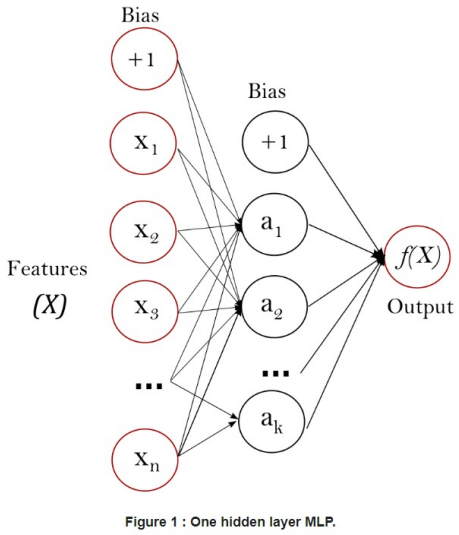

Figure 1: Multi-layer perceptron with 1 hidden layer

Related Work

This problem has been addressed in the past using various data mining techniques and machine learning algorithms.

A heart disease prediction system is been developed using orthogonal Local Preserving projection and hybrid classifier. In the hybrid classifier, group search optimizer algorithm was associated with Levenberg-Marquardt Training algorithm in the neural network. The best weight acquired from LM algorithm was associated with GSO algorithm. The proposed LM algorithm based neural network was prepared under n number of iterations and another best weight was acquired which improved the classification accuracy. The result concluded that the performance measures such as accuracy, sensitivity, and specificity achieve optimal value in GSO-LM algorithm while compared to existing approaches.

Multiple linear logistic regression technique has been used to describe the data with 13 attributes from a Cleveland heart disease data set and to explain the relationship between one dependent variable and two or more independent variables. Analyzing the correlation and directionality of the data, fitting the line, and evaluating the validity and usefulness of the model are the different stages of multiple linear regression model. The regression line represents the estimated disease chance for a given combination of the input factors.

Using the K-Nearest Neighbors, decision trees and Nave Bayes classifier the prediction accuracy has been checked for Cleveland heart disease dataset. The paper have considered only 8 Parameters out of 13 to see if prediction accuracy is comparable with other algorithms accuracy using 13 parameters. With KNN algorithms, the clusters are formed using non-medical (age and sex) and medical parameters(CP, Trestbps, Trestbpd, Restecg, Thalrest, and Exang). The parameters have been given different weights and studies over which weights possibly give the best accurate results.

There are have been studies developed using Neural Networks to address the issue of predicting heart disease severity. Different classification algorithms have been used for comparing accuracies. Thus we decided to use Multilayer perceptron and SVM supervised algorithms. With use of multiple variations of these algorithms, Multilayer perceptron has been studied for finding out best possible accuracy based on activation function and hidden layers size. SVM has been experimented with linear and non-linear kernels.

We wanted to compare with other algorithms implemented so far.Questions such as what would be the accuracies achieved by these 2 algorithms and which one could be considered as a better solution has been addressed.

Project Description

A. Dataset

The dataset for implementation of a proposed system is been retrieved from UCI machine learning data repository. The dataset contains 4 different files Cleveland.csv, Hungarian.csv, Longbeach.csv, Switzerland.csv containing records of patients explaining their health conditions. According to the UCI, this database has 76 attributes, but all published references of this dataset contains the 14 important attributes which are significantly responsible for prediction. So, we are considering these 13 attributes to create a prediction system. The dataset attributes are described in Figure 2. Some of these attributes are categorical, some are Real Number values. The Labels are classified into 5 different classes 0 to 5, i.e., 0 being the absence of heart disease and 1, 2, 3, 4 indicating severity of heart disease occurrence in future.

B. Data Preprocessing

The Dataset contains records with missing values. The dataset has been prepressed to remove duplicate records and records with missing values. For classification using Multilayer perceptron the features are needed to be scaled and normalized. The dataset has been normalized using decimal scale normalization.

Decimal Scale Normalization

Decimal scale normalization based on the movement of decimal points of values of attribute. The decimal points numbers are moved depends on the maximum absolute values of attribute. The decimal scale normalization formula is,

$$d^{'} = \frac{d^{'}}{10^{m}}$$

where m is the smallest integer that max $|d^{'}| < 1$

The description of attributes

Age	Age of a patient
Sex	1 = Male, 0 = Female
CP	*Chest pain type* Value 1: typical angina Value 2: atypical angina Value 3: non-angina pain Value 4: asymptotic
thestbps	Resting blood pressure (in mm Hg)
Chol	Serum cholesterol (in mg/dl)
Restecg	Resting electrographic results Value 0: normal Value 1: having ST-T were abnormality (T were inversions and/or ST elevation or depression of >0.05 mV) Value 2: showing probable or define left ventricular hypertrophy by Ester criteria
Fbs	(fasting blood sugar >120 mg/dl)
thalach	Maximum heart rate achieved
exang	Exercise induced Angina 1 = yes, 0 = no
oldpeak	ST depression induced by exercise relative to rest
slope	Slope of the peak exercise ST segment Value 1: upsloping Value 2: flat Value 3: down sloping
ca	Number of major vessels colored by fluoroscopy (0-3)
thal	Defect type 3 = normal; 6 = fixed defect; 7 = reversable defect

Figure : Dataset Attributes

Figure 2: attribute list and its description

C. Implementation

1. MultiLayer Perceptron

Features: 13
Classification labels: 0 to 4

The features are scaled using decimal scale normalization before applying neural network for classification. Features applied on the input layer of a network are fired to next layer neuron using weighted linear summation. At each neuron the value is calculated as:

$$Y = \sum(Weight + input) + bias$$

The activation functions are applied on the value of a neuron and based on value received after applying activation, the neuron is fired to next layer.
The different Activation functions used:
1. Rectified linear function, relu

$$f(x) = max(0, x)$$

2. Hyperbolic tan function, tanh

$$f(x) = tanh(x)$$

3. Sigmoid function logistic

$$f(x) = 1/(1 + exp(-x))$$

Also, we implemented Multilayer Perceptron with backpropagation (without using any activation function).The results of these implementation are discussed in next section.

1. Support Vector Machine

To predict the labels for the given data we first extract training features and labels and then we ask the SVM model to predict the labels for test set. The basic code block snippet looks like below (fig 3):

```
def run(self):
    svc_scores = []
    trainingData, trainingLabels = Dataset().getTrainingDataAndLabels()
    trainingLabelsForLinear = trainingLabels

    for index, value in enumerate(trainingLabelsForLinear):
        if value > 0.0:
            trainingLabelsForLinear[index] = 1.0


    # features - training data = 0 to 700 records from data
    features = trainingData[:700]
    # labels = training labels 0 to 250 records from data
    labels = trainingLabels[:700]
    linearLabels = trainingLabelsForLinear[:700]

    # features to test = 250 above records
    testFeatures = trainingData[700:]

    # labels of test data to check accuracy later
    testLabels = trainingLabels[700:]
    testLinearLabels = trainingLabelsForLinear[700:]


    #linear kernel
    svc_classifier = svm.SVC(C=3, kernel='linear')
    svc_classifier.fit(features, linearLabels)
    # print "accuracy of training at:  ", svc_classifier.score(features, linearLabels) * 100, "\n"
    # print "accuracy of testing at:  ", svc_classifier.score(testFeatures, testLinearLabels) * 100, "\n"
    svc_scores.append(svc_classifier.score(testFeatures, testLinearLabels))
    prediction = svc_classifier.predict(testFeatures)
    print "------------ linear----------"
    print "Prediction for test data:\n",prediction, "\n"
    print "Accuracy of prediction",(accuracy_score(testLinearLabels,prediction, normalize=True)), "\n"
```

Figure 3: SVM code snippet

This is very basic implementation with only value C altered to increase the accuracy. It assumes default values of tuning parameters (kernel = linear, and gamma = 1)

Further parameter tuning is done by changing the kernel to RBF and varing Gamma and C values.This determines if the accuracy can be increased.

Experiments

Initially, we implemented different variations of Multilayer Perceptron and Support Vector Machine algorithms. The Experiments results we received are such:
For Multilayer perceptron different hidden layer counts seemed to give different results. Based on activation function used on the feature data, number hidden layers have been varied to have sufficiently significant results on a data. The hidden layer size has been chosen based on the criteria that it should be any number greater than 2/3rd of input layer size and less than input layer size.

$$2/3(size\ of\ input\ layer) < (hidden\ layer\ size) < (input\ layer\ size)$$

Whereas input layer size is nothing but number of total features applied to input layer (13 for this project) and Hidden layer indicates number of neurons in each hidden layer. Hidden layer size is kept constant across all activation functions. The performance of each activation function seems to be varied based on number of hidden layers introduced between the input and output layer. Number of iterations are also taken into considerations to find what would be possible iterations needed for network to converge.

Initially we implemented the neural network with rectified linear function relu. With 3 hidden layers, the accuracy achieved is 99.40% on the testing data. Relu converged at 3000 iterations with 3 hidden layers. We ran another configuration of Relu with 5 hidden layers and iterations variation between 2000 and 3000 (Relu implementation converged between 2000 and 3000). The performance accuracy score seemed to decrease to 92.34%. Therefore, we considered Relu with 3 hidden layers and 3000 iterations as the better variation.

The neural network implementation with hyperbolic tan function tanh gives a performance of 90.63% performance with 3 hidden layer count, it converged around 1000 iterations. Another implementation of tanh with 6 hidden layers converges around 1000 iterations giving the performance of accuracy score 93.36%. Neural network with tanh activation seems to converge around 500 to 1000 iterations.

Sigmoid activation function has shown comparatively poor performance than use of above 2 activation functions. We ran variation of neural networks for sigmoid, the algorithm converges at just 500 iterations with accuracy of just 56.63%. Increasing hidden layer to 4 or 5, the algorithm takes around 2000 iterations to converge, giving a slightly higher accuracy of 67.38%. Looking at performance of a sigmoid function, it seems using a logistic activation function (sigmoid) isn't a good choice.

Implementing neural network with backpropagation seems to have poorest performance of all the above. For both any hidden layer count 3 to 6, algorithms converged around 2000 iterations with performance varying between 56.38% to 60.67%, thus implementing neural networks with backpropagation for prediction has a really bad performance.

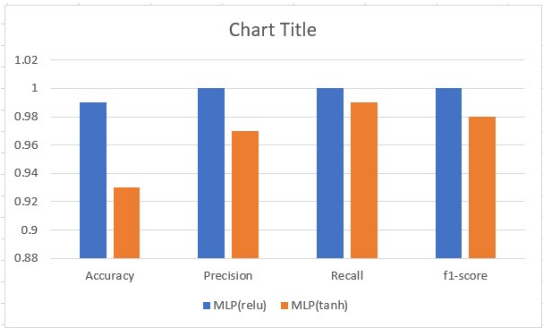

Figure 4: MLP Tanh and MLP Relu Performance

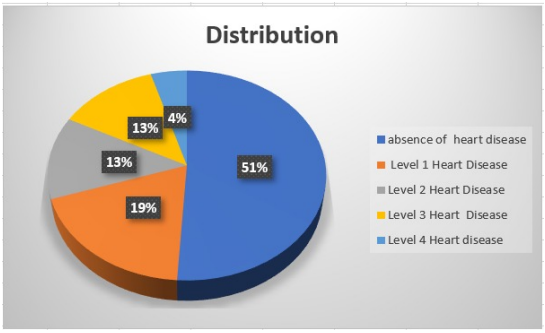

Figure 5: Distribution of patients using MLP Classifier

Given pie chart shows the results of test labels classification based on heart disease severity. By referring the figure 4 and 5, we can observe that Out of 196 records 100 records are classified as healthy with precision 1.0. 37 records classified as level 1 heart disease with precision of 1.0. 25 people classified with level 2 heart disease with precision 1.0, 25 people classified with level 3 heart disease with precision 1.0, and 9 people classified with precision of 0.9 with heart disease occurrence for sure.

So We will consider the performances of hyperbolic tan(tanh) and rectified linear(relu) while comparing it to the variations of SVM algorithm.

For Support Vector Machine, we trained the data with two different kernels. The most trivial kernel is the linear kernel which uses inner products of the original data. We started with checking the accuracy by performing linear SVM where kernel is set to linear and C is set to 1. This resulted in prediction accuracy of 81.12%. Next varying the parameter C to 2 the accuracy increased to 81.6%. We observed that further increasing the C value made a drop in the accuracy by 1%.

For RBF kernel, we need to determine two parameters C and Gamma. Since the features are not that large it is suitable to use RBF kernel. Initially the C parameter was set to 10 to determine the accuracy which resulted in 80%. Increasing the C parameter in terms of 10,50,100 increased the performance to 98% and it remained constant after a point. Adding up the gamma parameter with value 0.01 improved the performance by 1% with accuracy 99.48%. Fig 6 below depicts the performance of the above two classifiers.

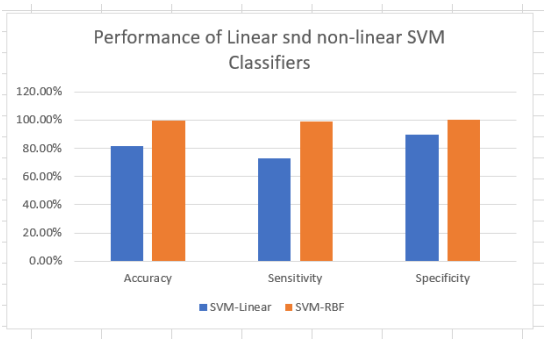

Figure 6: SVM Classifier Performance comaprison

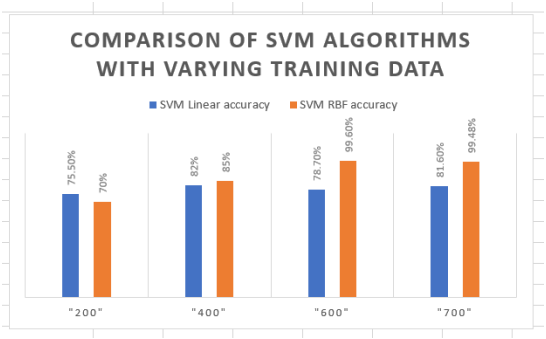

Figure 7: SVM performance with different number of samples

From the results above in fig 7, we could see that: The accuracy apparently increased as the amount of training data

is increased for RBF kernel classifier. Below is the distribution of patients categorized to different classes using SVM Classifier with RBF kernel.

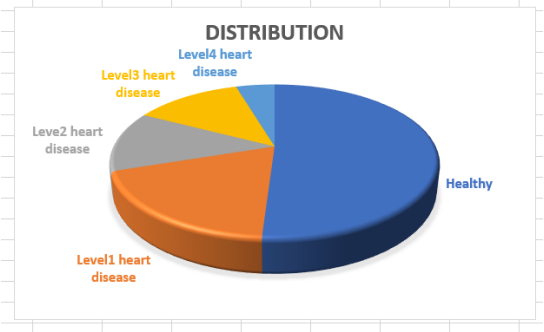

Figure 8: Distribution of patients by SVM Classifier

As per fig 8 and the testing data of 196 patients, below are the exact numbers belonging to healthy and other severity level of heart disease:
100 of them are healthy(No heart disease)
37 of them are categorized to level 1 severity
25 of them are categorized to level 2 severity
25 of them are categorized to level 3 severity
9 of them are categorize to level 4 severity

Below is the statistics of TP,FP,TN,FN where:
TP(True Positive) denotes the number of records classified as true while they were actually true.
FN(False Negative) denotes the number of records classified as false while they were actually true.
FP(False Positive) denotes the number of records classified as true while they were actually false.
TN(True Negative) denotes the number of records classified as false while they were actually false.

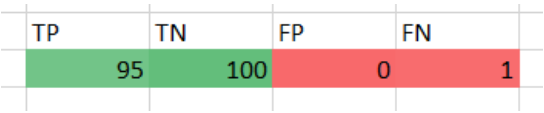

Figure 9: Confusion Matrix Values

Figure 10 shows the comparison analysis of best variations for algorithms MLP Relu and SVM with rbf kernel. We can see that Accuracy score for both variations seem aligned with each other for equal number of test labels. The MLP NN with relu activation function seem have a better precision rate as compared to SVM with rbf kernel. That mean false positives have been classifies with significant number of records in SVM rbf kernel.Both the algorithms also have significantly aligned recall with a values 97.5% for SVM and 99% for MLP NN. From the above observation we can recognize that MLP neural network with the use of activation function relu seems to give perfect accuracy for classification. Thus MLP neural network has better precision accuracy in both the algorithms.

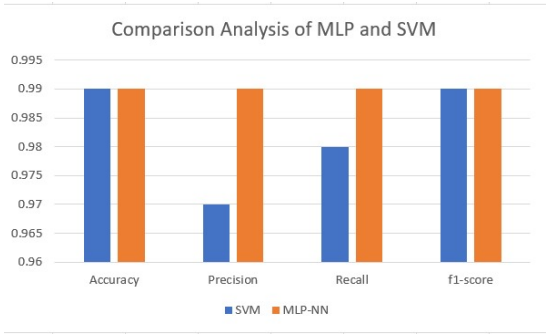

Figure 10: Comparison of SVM and MLP Classifier

Conclusion

In this paper, different classifiers are studied and the experiments are conducted to find the best classifier for predicting presence of heart disease in a patient. We propose an approach to predict the heart diseases using Machine Learning algorithms. Support Vector Machine classifier and Multilayer Perceptron were used for diagnosis of patients with heart diseases. Observation shows that SVM with RBF kernel shows better performance and is having more accuracy, when compared with other classification method. The best algorithm based on the patients data is Multi-layer Perceptron Classification with accuracy of 99.48%, sensitivity of 100% and specificity of 100% and the total time taken to build the model is at 2.34 seconds.

References

1. Improved Study of Heart Disease Prediction System using Data Mining Classification Techniques. Available from: https://www.researchgate.net/publication/258651311_Improved_Study_of_Heart_Disease_Prediction_System_using_Data_Mining_Classification_Techniques[accessedApr232019]
2. K. Polaraju, D. Durga Prasad, Prediction of Heart Disease using Multiple Linear Regression Model, International Journal of Engineering Development and Research Development, ISSN:2321-9939, 2017
3. https://www.researchgate.net/publication/313717803_Heart_disease_prediction_system_using_k-Nearest_neighbor_algorithm_with_simplified_patient's_health_parameters
4. http://www.alliedacademies.org/articles/a-novel-approach-for-diagnosing-heart-disease-with-hybrid-classifier.pdf
5. https://scikit-learn.org/stable/modules/neural_networks_supervised.html
6. https://medium.com/machine-learning-101/chapter-2-svm-support-vector-machine-theory-f0812effc72
7. https://medium.com/machine-learning-101/chapter-2-svm-support-vector-machine-coding-edd8f1cf8f2d
8. https://scikit-learn.org/stable/modules/neural_networks_supervised.html#multi-layer-perceptron

9. https://pdfs.semanticscholar.org/caf8/b4d1b1cabe259aa84d12e44c6cdbdf908c81.pdf
10. https://www.sciencedirect.com/science/article/pii/S187705091630638X
11. https://www.cdc.gov/heartdisease/facts.htm
12. https://archive.ics.uci.edu/ml/datasets/heart+Disease
13. https://en.wikipedia.org/wiki/Support-vector_machine
14. https://en.wikipedia.org/wiki/Supervised_learning
15. https://www.cdc.gov/dhdsp/data_statistics/fact_sheets/hds_index.htm
16. https://www.heart.org/en/news/2018/05/01/heart-disease-stroke-death-rates-increase-following-decades-of-progress
17. https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/