Diagnosing patients with heart disease using different classifiers

Akshat Shukla, Ankit Ashok Baheti and **Hardik Shah**
College of Computer and Information Science, Northeastern University
Email: {shukla.ak, baheti.a, shah.hardik}@husky.neu.edu

Abstract

A heart attack is one of the leading causes of death in the United States. More than 1.5 million Americans suffer a heart attack every year, and almost half a million die, according to the American Heart Association. This domain problem requires early-action to resolve the disease. Hence, machine learning predictions based on old patient's data can help predict statistics from any future patient's report, whether they are more prone to getting any kind of Cardiovascular Disease (CVD), with a certain accuracy. A heart attack prediction system can be valuable to doctors to make an accurate decision in early stages to save a patient's life. According to various medical sources, there are numerous parameters like age, sex, chest pain, cholesterol, blood sugar, resting electrocardiographic results, etc. that are used by medical institutions across the world in researching the cause and diagnoses of CVDs. In this paper, we use some of the popular Machine Learning algorithms namely, Decision Trees, Random Forest, Support Vector Machines, Neural Networks, and K Nearest Neighbors for prediction of heart disease based on the data of around 300 patients with 14 major characteristics to build a system for prediction of heart attacks and provide detailed comparative analysis for all the above algorithms on testing data along with graphical representation. These analyses can be used to select the most suitable algorithm to predict more data.

Introduction

According to the Centers for Disease Control and Prevention [1], about 610,000 people die of heart disease in the United States every year–that's 1 in every 4 deaths. Heart disease is the leading cause of death for both men and women. More than half of the deaths due to heart disease in the last decade were in men, which suggests that sex is one of the major factors in deciding the risk of having a CVD [2].

The medical field has a lot of raw patient data which can be a useful resource in the research and study of diagnoses of many diseases. One such disease which has a large scope of research for treatment methods is Cardiovascular Disease. Using machine learning, this problem can be characterized into statistical insights which will, in turn, be useful for doctors to help in the judgment and treatment of the disease. There are many well-known algorithms to make predictions and classification models for this problem but choosing the best algorithm which provides optimum accuracy for prediction for this particular problem still remains a work in progress. This paper attempts to portray comparative analyses of 5 of the most-known machine learning algorithms that train their respective models for predicting accuracy over test data of future patients. The statistical graphs and classification reports describing the accuracy, precision, recall, and F-score for each of these algorithms can help us narrow down on the best approach to use in solving this problem. The implementation of this project uses the raw data obtained from the UCI's machine learning repository [3] which involves data describing 13 most critical features which contribute most to the risk of having a heart disease for about 303 patients. This data was collected and investigated by medical institutions [4].

The first approach we studied is Decision Tree, using the ID3 algorithm, to classify the data into binary classification to design a decision tree with nodes out of the 13 features in the pre-processed data which cause branching of the tree ordered according to the most appropriate feature and providing the result at the leaf node. The Support Vector Machine algorithm tries to predict the data by maximizing the margin of the decision boundary. The Random Forest algorithm selects the best features after ranking all 13 features based on their importance factor, to create a forest of decision trees and tries to classify between low-risk and high-risk patients. Neural Networks creates a prediction model consisting of a specified number of hidden layers which produces maximum accuracy for predicted values and minimizes the error of the prediction of the heart disease. The KNN algorithm tries to cluster the data for classification into 2 clusters (high and low-risk individuals) by finding the distance between the nearest neighbors and predict the data based on these clusters.

The entire paper has been compiled under the following 5 sections which outlay the background and related work, implementation and comparative analysis based on the

implementation of this project. Section II deals with the background of the problem and followed by related work being published and implemented to solve the problem defined in Section III. The project description is stated in Section IV, which is followed by statistical results and detailed analysis of the performances in Section V. The paper culminates with the conclusion of work, suggest the optimal machine learning algorithm most suited to classify the given problem in Section VI.

Related Work

Previous work related to the same domain include, but are not limited to, a Heart Attack Prediction System [5], which worked on the same University of California, Irvine's machine learning repository dataset and used Rapid Miner tool to find out the best fitting algorithm for the classification of this data and proposed system that found reliable insights in which Naïve Bayes proved to be the best when compared to Decision Trees, K-Nearest Neighbor, and Random Forest for this particular dataset. Naïve Bayes also proved to be highly accurate in classification on a different dataset of heart diseases obtained from AFIC, Pakistan [6]. Moreover, while classifying datasets related to heart diseases obtained from National Health and Nutrition Examination Survey, SVM managed to achieve the highest accuracy as against other aforementioned general techniques, standing at an accuracy of 83% with and Boosting being used for predictions [7]. A different approach in the same domain problem was taken by D. K. Ravish.et al [8], worked on the Physio net Electrocardiography database with 7 ECG data features in addition to indigenous clinical data with 6 clinical features which together are used to train a Neural Network model which can be used to classify the heart diseases and to predict any abnormalities in heart functioning. Similar to a web interface designed to inform the users about the prediction by the system based on his/her data [5], cardiac monitoring systems which use mobile monitoring servers and gateways [9] have also been used to better understand these predictions and help doctors make a suitable decision ahead.

Project Description

The University of California, Irvine's machine learning repository dataset provides a well-documented 14 feature dataset of about 303 patients including the binary classification of risk of heart attack in comma separated value format. The pipeline in Fig.1 shows the overall workflow of the project.

As shown in the Fig.1, the project can be broken down into three stages:

- **Data Pre-processing**

The main purpose of this step is to remove the noise from the raw data retrieved in order to construct prediction models. The following three steps were implemented on the raw data:

1. Clean the raw data to remove any missing attribute (‘?’) and replace it with -1.
2. Replace all the -1 in respective columns with the mode of that column.

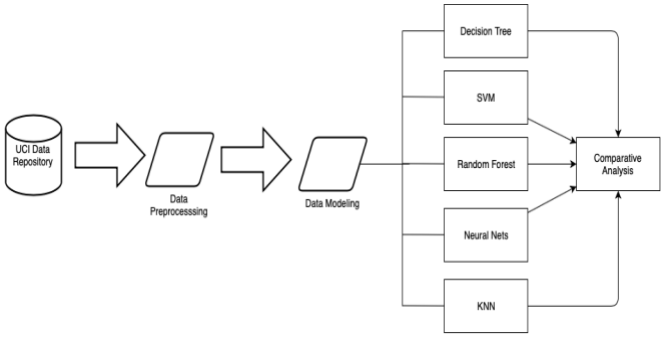

Fig. 1 Overview of workflow

3. Normalizing the data according to the mean normalization [10] equation provided below to make sure that objective function of the algorithm works properly, and values vary in the specific range.

$$x' = \frac{x - \text{average}(x)}{\max(x) - \min(x)}$$

where x is an original value, x' is the normalized value.

4. The normalized data-frame is statistically summarized as follows:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000
mean	54.438944	0.679868	3.158416	131.689769	246.693069	0.148515	0.990099	149.607261	0.326733	1.039604	1.600660	0.663366	4.722772
std	9.038662	0.467299	0.960126	17.599748	51.776918	0.356198	0.994971	22.875003	0.469794	1.161075	0.616226	0.934375	1.938383
min	29.000000	0.000000	1.000000	94.000000	126.000000	0.000000	0.000000	71.000000	0.000000	0.000000	1.000000	0.000000	3.000000
25%	48.000000	0.000000	3.000000	120.000000	211.000000	0.000000	0.000000	133.500000	0.000000	0.000000	1.000000	0.000000	3.000000
50%	56.000000	1.000000	3.000000	130.000000	241.000000	0.000000	1.000000	153.000000	0.000000	0.800000	2.000000	0.000000	3.000000
75%	61.000000	1.000000	4.000000	140.000000	275.000000	0.000000	2.000000	166.000000	1.000000	1.600000	2.000000	1.000000	7.000000
max	77.000000	1.000000	4.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.000000	6.200000	3.000000	3.000000	7.000000

Table 1. Feature Summarization

The above table shows the respective column features. These features are age, sex, cp (chest pain), trestbps (resting blood pressure), chol (cholesterol), fbs (fasting blood sugar), restecg (resting electrocardiographic results), thalach (heart rate) ,exang (exercise induced angina), oldpeak (ST depression induced by exercise relative to rest), slope (slope of the peak exercise ST segment), ca (number of major vessels), thal.

- **Decision Trees**

A decision tree is a flowchart-like structure in which each internal node represents a “test” on an attribute, each branch represents the outcome of the said test, each

leaf node represents a class label. The path from the root to a leaf represents the classification rule. Our approach for decision trees is using the ID3 algorithm to create a tree that takes input a vector of attributes and outputs a single value that represents a particular class label which in our case is either 0 or 1.

The ID3 algorithm is a recursive algorithm which begins by determining the best attribute which will act as the root of the tree. On each iteration, it iterates through all unused attributes and calculates the entropy and information gain of each attribute. The attribute which has the least entropy or maximum information gain is selected as the root node. The algorithm then recurves on the remaining attributes to construct the tree.

The entropy for each attribute can be calculated as follows:

$$\mathrm{H}(S) = \sum_{x \in X} -p(x) \log_2 p(x)$$

where S is the current dataset, x is the particular class from a set of classes.

The information gain is basically the difference in Entropy before the split and after the split on an attribute.

$$IG(S, A) = \mathrm{H}(S) - \sum_{t \in T} p(t)\mathrm{H}(t)$$

where H(S) is the entropy before the split and H(t) is entropy after the split.

The decision tree can be represented as follows:

- Calculate the entropy of every attribute of the data set.
- Partition the set into subsets using the attribute for which the resulting entropy after splitting is minimized; or, equivalently, information gain is maximum
- Make a decision tree node containing that attribute.
- Recur on subsets using remaining attributes.

Pseudo code for ID3 is as follows:

Function DTL (examples, attributes, default)

returns a decision tree

If examples are empty then return the default

Else if all examples have the same classification then return the classification

Else if attributes are empty then return Mode(examples)

Else

best ← choose-attribute (attribute, examples)

tree ← a new decision tree with root best

For each value vi of best:

examples ← {elements of examples with best = vi}

subtree ← DTL (example(i), attributes - best, Mode(examples))

add a branch to a tree with label vi and subtree subtree

return tree

- **Support Vector Machine**

Support Vector Machine is supervised machine learning models that are used for classification and regression analysis. Support Vector Machine takes input set of training points which belong to two or more class labels, it builds a model based on these data points and tries to predict the correct label for future test data.

Support Vector Machine [11] tries to find the hyperplane dividing the data points (in our case, we have 13 feature columns) into two halves and find a maximum margin between two sets. Since, there can be many hyperplanes that divide the dataset, the main aim of the margin is to make sure that correct hyperplane is selected such that when a new dataset is tested, the data points are placed in a proper class.

Let (x_i, y) represent a dataset where x_i is the vector of attributes and y is the class label.

The hyperplane for the above set of data can be represented using the equation:

$$\vec{w} \cdot \vec{x} - b = 0,$$

and vector w represents a vector normal to the hyperplane. Since we want to divide the data into two classes, the values of y can be either +1 or -1, we can construct two hyperplanes and maximize the distance between these hyperplanes. The hyperplanes can be represented using below equations:

$$\vec{w} \cdot \vec{x} - b = 1$$

$$\vec{w} \cdot \vec{x} - b = -1.$$

The geometric distance between these two planes is $\frac{2}{\|\vec{w}\|}$. In order to maximize the distance, we need to minimize $\|\vec{w}\|$. We also want to make sure that new data points do not lie on these hyperplanes. Hence, we add constraints:

$$\vec{w} \cdot \vec{x}_i - b \geq 1, \text{ if } y_i = 1$$

$$\vec{w} \cdot \vec{x}_i - b \leq -1, \text{ if } y_i = -1$$

Hence, putting all this together and generalizing the equations, support vector machine is done by solving the below optimization problem:

"Minimize $\|\vec{w}\|$ subject to $y_i(\vec{w} \cdot \vec{x}_i - b) \geq 1$, for $i = 1, \ldots, n$"

The vector w and b that solve the above equation, can now be put into a sigmoid function which basically acts a predictor to output either +1 or -1 for the test data. The

following Fig. 2 shows the basic concept of SVM with margins:

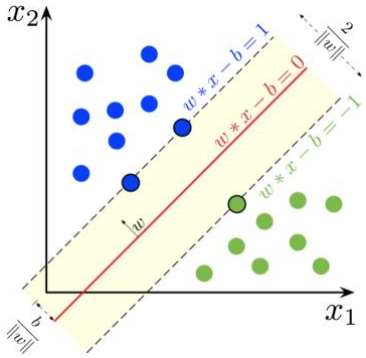

Fig.2 Margins for SVM

- **Random Forest**

  Random Forest is a supervised classification algorithm. Random Forest is basically a group of decision trees. Hence, the higher the number of decision trees in the forest, higher is the accuracy. Random Forest uses the prediction from all the trees to make a final prediction. It makes the final prediction either from the mode of classes or the mean prediction of classes. Since random forest uses a more than one approach in finding the correction label for test data, it is a ensemble algorithm.

  An important inference that can be generated from the Random Forest is feature importance. Feature importance [12] is an important analysis that helps to determine the best of the features if dataset consists of a large number of features. Features importance is basically the number of samples that reach the node divided by the total number of samples.

  Because the Random Forest is a combination of a large number of decision trees, it solves the overfitting problem.

  The Algorithm Random Forest is as follows [13]:

  1. Select k random features from a total of m features.
  2. Among the k features, select the best feature.
  3. Split the tree based on the best feature and repeat the steps to create a decision tree.
  4. Repeat the above steps to create a forest by building n decision trees.
  5. Predict the test data on the above created random forest.

- **Artificial Neural Networks**

  Artificial Neural Networks are highly used as supervised algorithms for classification of data and deliver good accuracy results as well. A Neural Network traditionally has 3 types of layers: an input layer (13 units for the 13 feature columns in this problem), an output layer (which is 1 unit for this problem) and a certain number of hidden layers in the middle. Deciding the units of neurons in the hidden layers is a very important part of deciding your overall neural network architecture. Though these layers do not directly interact with the external environment, they have a tremendous influence on the final output and the accuracy of the model. Both the number of hidden layers and the number of neurons in each of these hidden layers must be carefully considered. There are many rules-of-thumb which can be used to determine a good number of hidden layer units while constructing the neural network, but one can simply choose it to be ⅔ of the number of input layer units (13 feature columns) plus the number of output layer units [14]. The following Fig. 3 shows the concept of Neural Network:

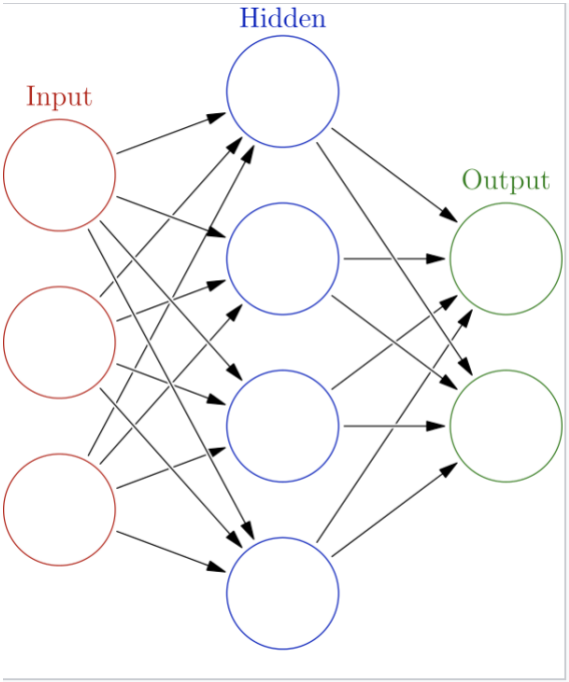

Fig. 3 Artificial Neural Network Layers

- **k-Nearest Neighbors**

  k-Nearest Neighbor is a simple, non-parametric classifier. Non-parametric classifier means there are no assumptions about the underlying data. In KNN, K is the number of nearest neighbors.

  Algorithm for KNN:

  1. If P is the point for prediction, then find the K nearest data points from P. The distance from point P can be calculated using Euclidean distance, Manhattan distance, and Cosine distance.

2. Based on the labels for the K points, Point P is assigned the label which is the maximum label for the K points.

Experiments and Model Analysis

The 300 rows dataset was divided into 250 rows for training data and 50 rows for test data. The results and comparisons of each model are as follows:

- ***Decision Tree***

  We have implemented decision tree in two ways. In first method, we implemented algorithm that is provided above in the decision tree description and the second method is implemented using the Scikit-learn library.

  At the end of both the methods, we have obtained the following confusion matrix:

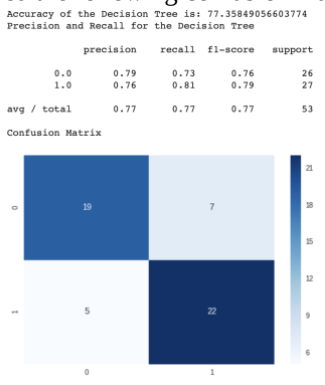

Fig. 4

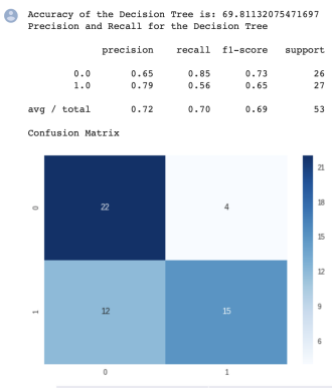

Fig. 5

The result from Fig. 4 is using the ID3 algorithm in which the accuracy is 77.35% whereas the accuracy from Scikit-learn library's decision tree is 69.8% as shown in Fig. 5. This is because Scikit-learn predicted less True positives (15) compared to ID3's true positive (22) thereby decreasing the overall prediction. The accuracy is calculated on total of true positives and true negatives divided by total test data. Splitting of different nodes in both the tree leads to generation of two different tree in both cases i.e. there might be a node which might have minimum entropy in one tree but not in other, thereby resulting to difference in accuracy. Based on these results, we had a doubt that the decision tree overfits the model for the training data. Hence, we decided to run the data set on Random Forest.

- ***Random Forest***

  Based on the analysis from decision tree, we ran random forest algorithm on all of our 13 features using Scikit-learn library and generated 10 decision trees to make sure that decision tree does not overfit the data. Thus, after training the model on 250 dataset and testing on remaining 53 datasets, we have the following result in Fig. 6:

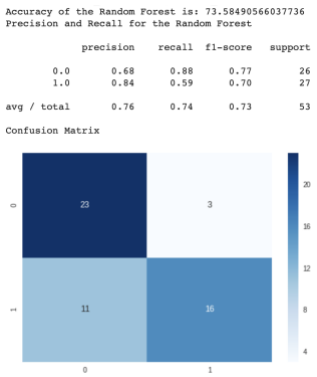

Fig. 6

The accuracy obtained from random forest is 73.59% on test data which shows that random forest tries to not overfit the model, and since it calculates the result based on several decision trees, it tries to give importance to features that affects the most in all decision trees. This led us to find more insights on feature importance.

Hence, based on the model trained above, we calculated features importance for the random forest. The following graph in Fig. 7 shows the result of feature importance ranking as shown in Fig. 8:

Fig. 7 Feature Importance Graph

```
Feature ranking:
1. feature 2 - cp (0.224430)
2. feature 12 - thal (0.196120)
3. feature 9 - oldpeak (0.115312)
4. feature 11 - ca (0.107373)
5. feature 7 - thalach (0.074184)
6. feature 4 - chol (0.072450)
7. feature 3 - trestbps (0.065642)
8. feature 0 - age (0.062212)
9. feature 10 - slope (0.029684)
10. feature 1 - sex (0.021000)
11. feature 8 - exang (0.015978)
12. feature 6 - restecg (0.011040)
13. feature 5 - fbs (0.004575)
```

Fig.8

We see that cp, thal, oldpeak, ca and thalach are the top 5 features. So, according to these top features based on feature importance, there might be a chance that these features influence the overall decision.

Hence, from the top 5 features we, take the top 2 features and convert the whole data frame into these two features using the pandas library. Once, we have the new data frame, we again normalize the data to prevent too much deviation on the new data. Now, we again try to fit the data to the random forest model and calculate the accuracy and results are as shown in Fig. 9.

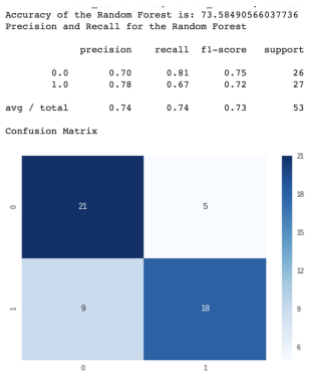

Fig. 9

From Fig.9 we see that the accuracy of the random forest from just the data based on two frames is that 73.58% which is same as the accuracy from the overall data. Hence, the top features play an important factor in making decisions and can influence the overall decision.

- ***Support Vector Machine***

  We also ran our dataset on the Support Vector Machine algorithm. We provided two implementations of SVM, in first one we wrote our own radial basis function kernel for non-linear classification of the data. The other implementation was achieved using Scikit-learn library. The result that we get from both algorithms is as follows:

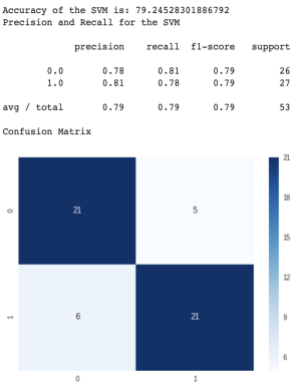

Fig. 10

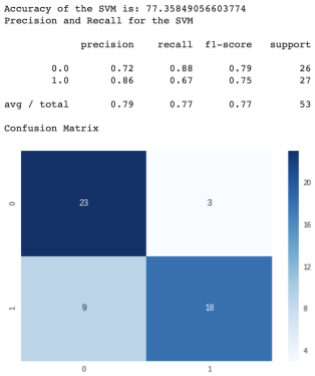

Fig. 11

Fig.10 shows the result of our implementation in which accuracy is 79.24% and Fig.11 shows the Scikit-learn implementation of SVM using linear kernel and in this the accuracy is 77.35% that is both of them are close enough and predict data well.

Since, we used linear kernel from Scikit-learn library, we then tried to find the analysis between different type of kernels. Thus, we visualized the data on 3 different types of kernels: Linear, Radial basis function and Polynomial using the Principal Component Analysis to reduce the features for

plotting from 13 to 2. We got the following graph shown in Fig. 12:

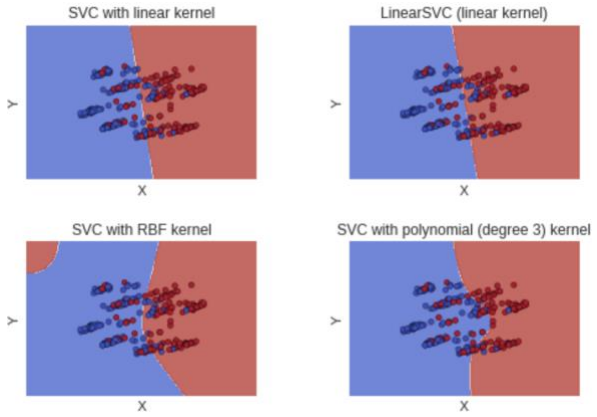

Fig. 12

From the above figure, we see that almost all kernels fit the model well but linear and polynomial have the best fitting. Hence, we ran SVM using linear kernel (Fig. 11) and SVM using polynomial kernel in order to analyze the kernel and models and found the following result in Fig. 13:

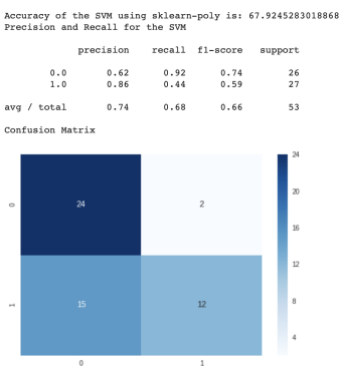

Fig. 13

We see that polynomial kernel overfits the data and hence the accuracy decreases. However, the above graph in Fig. 12 showed a quite a good fit was because the graphs were created on the whole data.

- ***Neural Networks***

  We also trained our model using Neural Network. The input to the neural net is 13 features and we are using one hidden layer. Using these, we trained the model and predicted the data.

  The following results were obtained as shown in Fig. 14:

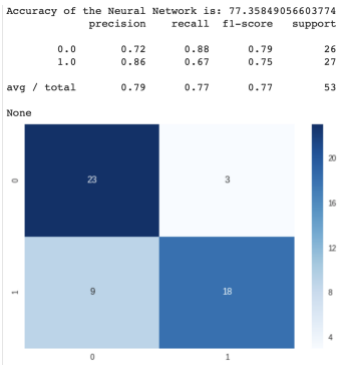

Fig. 14

The result is 77.35% on the test data. However, then tried to train the model using more hidden layers since dataset has more features, but because there is less data, the results varied too much on increasing hidden layers and hence we concluded on only one hidden layer.

- ***k-Nearest Neighbors***

  Heart attack prediction was also done using KNN algorithm. The KNN algorithm we ran has the following, parameters: 3 nearest neighbors and finding distance using Manhattan, Cosine and Euclidean distance to analyze which works best for the dataset. The following results were obtained by running the algorithm on three distance as shown in Fig. 15, Fig. 16, Fig. 17 respectively:

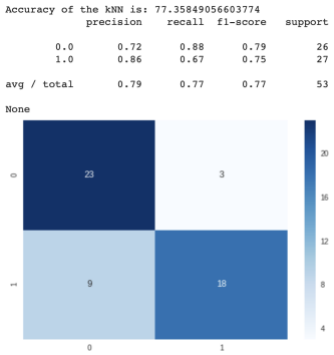

Fig. 15

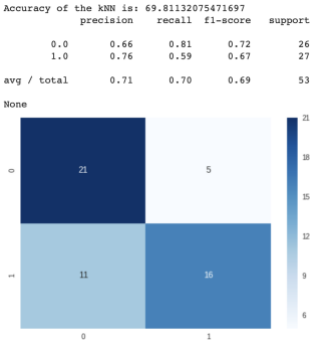

Fig. 16

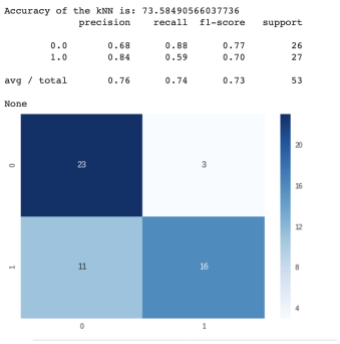

Fig. 17

The result obtained from Manhattan is 77.35%, Euclidean is 73.58% and Cosine is 69.01%.

Conclusion

This paper has presented a detailed comparative analysis of various classification machine learning algorithms such as Decision Trees, Random Forest, Support vector machine (SVM), Artificial Neural Networks and k-Nearest Neighbor for heart disease prediction of patients. Based on the comparisons on accuracy, the following Table 2 summarizes the analysis of the aforementioned algorithms:

Algorithms	Accuracy
Decision Tree	77.35%, 71.69% (Scikit-learn)
Random Forest	73.58% (Scikit-learn)
Support Vector Machine	79.24%, 77.35% (linear kernel Scikit-learn), 67.92% (poly kernel Scikit-learn)
Artificial Neural Networks	77.35% (Scikit-learn)
k-Nearest Neighbor	77.35% (Scikit-learn

Table 2. Accuracy Analysis

Based on the above results, it makes much more sense to classify the given data using Support Vector Machine algorithm, which can give an accuracy of 79.24%.

References

Heart Disease Facts (2017, November 28), Retrieved December 3, from https://www.cdc.gov/heartdisease/facts.htm

Cardiovascular disease (2018, December 1), Retrieved December 3, from https://en.wikipedia.org/wiki/Cardiovascular_disease

Dua, D. and Karra Taniskidou, E. (2017). *UCI Machine Learning Repository* [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

Heart Disease Databases (1988, July 22), contributed by Hungarian Institute of Cardiology. Budapest: Andras Janosi, M.D., University Hospital, Zurich, Switzerland: William Steinbrunn, M.D., University Hospital, Basel, Switzerland: Matthias Pfisterer, M.D., V.A. Medical Center, Long Beach and Cleveland Clinic Foundation:Robert Detrano, M.D., Ph.D.

Sushmita Manikandan, 2017. Heart attack prediction system. In *2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*, IEEE, Chennai, India.

M. Saq1ain, W. Hussain, N. A. Saqib, M. A. Khan, 2016. Identification of heart failure by using unstructured data of cardiac patients. In *2016 45th International Conference on Parallel Processing Workshops (ICPPW)*, pp. 426-431, IEEE, Philadelphia, PA, USA.

S. Palaniappan, R. Awang, 2008. Intelligent heart disease prediction system using data mining techniques. In *2008 IEEE/ACS International Conference on Computer Systems and Applications*, pp. 108-115, IEEE, Doha, Qatar.

D. K. Ravish; K.J. Shanthi; Nayana R Shenoy; S. Nisargh, 2014. Heart function monitoring, prediction and prevention of Heart Attacks: Using Artificial Neural Networks. In *2014 International Conference on Contemporary Computing and Informatics (IC3I)*, IEEE, Mysore, India.

K. Kwon, H. Hwang, H. Kang, K. G. Woo, K. Shim, 2013. A remote cardiac monitoring system for preventive care. In *2013 IEEE International Conference on Consumer Electronics (ICCE)*, pp. 197-200, IEEE, Las Vegas, NV, USA.

Feature scaling (2018, November 25), Retrieved December 4, from https://en.wikipedia.org/wiki/Feature_scaling

Support vector machines (2018, December 7), Retrieved December 4,from https://en.wikipedia.org/wiki/Support_vector_machine

The Mathematics of Decision Trees, Random Forest and Feature Importance in Scikit-learn and Spark (2018, May 11), Retrieved December 4, from https://medium.com/@srnghn/the-mathematics-of-decision-trees-random-forest-and-feature-importance-in-scikit-learn-and-spark-f2861df67e3

How the random forest algorithm works in machine learning (2017, May 22), Retrieved December 5, from http://dataaspirant.com/2017/05/22/random-forest-algorithm-machine-learing/

How to choose the number of hidden layers and nodes in a feedforward neural network? (2010, September), Retrieved December 6, from https://stats.stackexchange.com/questions/181/how-to-choose-the-number-of-hidden-layers-and-nodes-in-a-feedforward-neural-netw