Prediction of Heart disease using Machine Learning Algorithm

Gunjan Jhawar , gunjanjh@usc.edu, 12/4/2017

***Abstract*—In the recent years, heart diseases have been identified as one of the leading causes of death. The healthcare industry is still lagging in implementing effective ways to store and retrieve medical records which leads to less usable data for implementing effective machine learning algorithm. This report describes a project which compares machine learning algorithm in case of limited training and testing examples. It encompasses the comparative study of methods to predict missing data. Random forest regressor performs the best for predicting the missing values with an accuracy of 54.34%. SVM performs the best for multi and binary classification.**

***Index Terms*— Support Vector Machine, Logistic Regression, Decision Trees, Adaboost, Random forest regressor, K cross validation, Normalization**

I. Problem Statement and Goals

One in every four people is afflicted with or dies of heart disease, and in United states, over 610,000 afflicted Americans lose their lives annually[1]. However, these diseases are controllable and preventable which presents us the opportunity to predict its level of presence so that it can be treated accordingly. This is the motivation behind the project.

Due to the complexity of healthcare and a slower rate of technology adoption, the industry lags behind in implementing effective machine learning in analytic strategies[2].

This project comprises of multi-class classification to segregate data into absence(0) - presence(5) of heart disease. This project implements the following steps:

1. It compares the accuracy of methods which are used to predict missing values.
2. Using the best method for predicting missing values in the dataset and conducting comparative study of how algorithm performs when there is sparse and limited number of training and testing examples along with more than binary classes to classify the data.
3. Compare how the same algorithm performs if we cluster the class and make them binary.

The Cleveland Dataset is used from the UCI Machine Learning Repository which comprises of 303 samples and 76 features. The project uses 13 salient features out of the 76. The method used for predicting the missing values are:

Drop missing value containing feature, fill in missing value with -9 as done for other dataset in the data and random forest regressor. These methods are done with normalization and without normalizing the data to see which one does better.

Also, the subset of feature is taken randomly to compute how The classification algorithm used are: Support Vector Machine, Decision Trees(Adaboost) and Multi class logistic regression. K cross validation done on the data using k = n-1 where n is the number of features.

II. Prior and Related Work

None

III. Project Formulation and Setup

An analysis is done on the Cleveland dataset which includes pre processing as the data contains missing values for feature 11: the slope of the peak exercise segment and feature 12: number of blood vessels colored by fluoroscopy.

A. *Preprocessing*

a) *Dealing with missing values*

1. *Ignoring the features with missing values*

Data can have missing values for a number of reasons such as observations that were not recorded and data corruption.[3] The simplest method involves locating the feature vector consisting of missing values indicated by 'Nan', if any, ignore that feature vector.

2. *Replacing the missing values with a value*

It is not a preferred method to fill missing values but is specific to this dataset and project. The data folder contains information of other places other than Cleveland and they used -9.0 to fill the missing attribute. Filling the missing values with -9.0 for Cleveland dataset as well to see how well it performs for Cleveland dataset.

3. *Using random forest regressor to predict the missing values*

There are couple of methods to predict missing values other than random forest regressor like Bayesian formalism, clustering algorithms(K- mean\ Median etc.)[4].

The Random Forest is one of the most effective algorithms for predictive analysis. It is good at handling tabular data with numerical and/ or categorical features.The random forest is a type of additive model that makes predictions by combining decisions from a sequence of base models.

$$g(x) = f_0(x) + f_1(x) + \cdots$$

Where g is the sum of simple base models f_i. Here, each base classifier is a simple decision tree. All base models are constructed independently using a different subsample of the data [5].

b) Normalizing data

Data normalization is a technique used to control and eliminate redundancy. It can be done using first normal form, second normal form or taking the maximum of the column vector.It casts the data to a specific range and is helpful when there is a big difference in the range of different features.

After the pre- processing is done on the data, the classification algorithm are implemented taking 0 -12 features (detail included in section 7) and then classified as training data presented in column 13 as class ranging from 0 -4 where they indicate the level of presence of heart disease 0 showing absence to 4 showing presence.

B. Classification algorithm

1. Support Vector Machine

This is a class of supervised learning algorithms which trade offs accurachy for generalization error. SVM's build a hyperplane, and the examples of other class are all on the other side.

Consider input data form $(x_i\ y_i)$ where y_i are class labels. The hyperplane H maximizes the minimum distance from any training data point.

$$\{ x \in H | < w, x > + b = 0\}$$

where w is a vector orthogonal to the hyperplane and <> represents the dot product.

The resultant decision function has the following form

$$y(x) = sgn(\sum_{i=1}^{m} \alpha\, y_i\ < x_i, x > + b)$$

Thus the optimal margin hyperplane is represented as a linear combination of training points. Consequently, the decision function for classifying points with respect to the hyperplane involves dot products between points.

When the samples are not linearly separable, a kernel function is used to transform the data to higher dimensional space where it is linearly separable. The kernel gives the dot product of two examples in the higher dimensional space without actually transforming them into that space.

Two most commonly used kernel functions are polynomial functions and gaussian radial basis function.

However, many data sets may not be able to find separating hyperplane at all, either because the kernel function is inappropriate for training data or data contains mislabeled example.

In this case, a slack variable ρ and error penalty C are introduced.

Though the concept of SVM was originally proposed for binary classification, various methods have been proposed to use SVM for multi class problems. "One against one" and "One against all" methods are most popular.[6]

2. Decision Trees: Adaboost

Decision tree is a flowchart like structure in which internal node represents test on an attribute, each branch represents outcome of test and each leaf node represents class label and then decision is taken after computing all attributes. A path from root to leaf represents classification rules.[7] It is an adaptive basis function model and can be expressed as follows

$$f(x) = E(x|y) = \sum_{m-1}^{M} w\, I(x \in R_m) = \sum_{m-1}^{M} w\, \emptyset(x; v_m)$$

Most boosting algorithms have been restricted to reducing the multi- class classification problem to multiple two- class problems.[8]

3. Multi class logistic classifier

Classification using logistic regression is a supervised learning method. It predicts the probability of occurrence of an event by fitting data to a logistic function. In some cases, multi class logistic regression well fits features. It has the formula

$$p(y = c|x, W) = \frac{\exp(wT_c\, x)}{\sum_{c-1}^{C} \exp(wT_c x)}$$

where $p(y = c|x, W)$ is the predictive probability, y is class type of c. w is weight of class c approximated by maximum posterior probability.[7]

It can be optimized using L= BFGS which stands for limited memory Broyden-Fletcher-Goldfarb-Shanno and it is an optimization algorithm that is popular for parameter estimation.[10]

C. Combine classes to make multi class to binary

The method proposed combines classes based on their level of presence of heart disease as 0-2 gets to low chance of heart disease and 3-4 comes in the group of high chance of heart disease.

D. K cross validation

Cross evaluation is a model evaluation method that is better than the residuals. The problem with residuals is that they do not give an indication of how well the learner will do when it is asked to make new predictions. One way to overcome this problem is not to use the entire data while training.

The data set is divided into k subsets, and one of the k subsets is used for testing and k-1 for training. The average across all k trials is computed. The advantage of the method is that every data point gets a chance to be in the test data once and training k – l times if experiment is done l times. [10]

IV. METHODOLOGY

A) Dealing with missing values

1. We import the data as text file. Converting that text file into data frame so that it could be used with matrix containing samples and features.
2. Implementing 3 different techniques to make a comparative study of dealing with missing values: Ignore features with missing values, Filling in missing value with -9.0 and using random forest regressor to predict missing values. Normalization of data using normalization function from library.
3. Divide the data into train and testing with the ratio of 85:25 with random state.
4. Divide the training data into train and validation with the ratio of 85:25 with random state.
5. Using multi class logistic regression to classify data sets into 5 groups.
6. Performing the same experiment with different values of C (error penalty), tol(tolerance) and max iterations.
7. Report the accuracy of above methods.

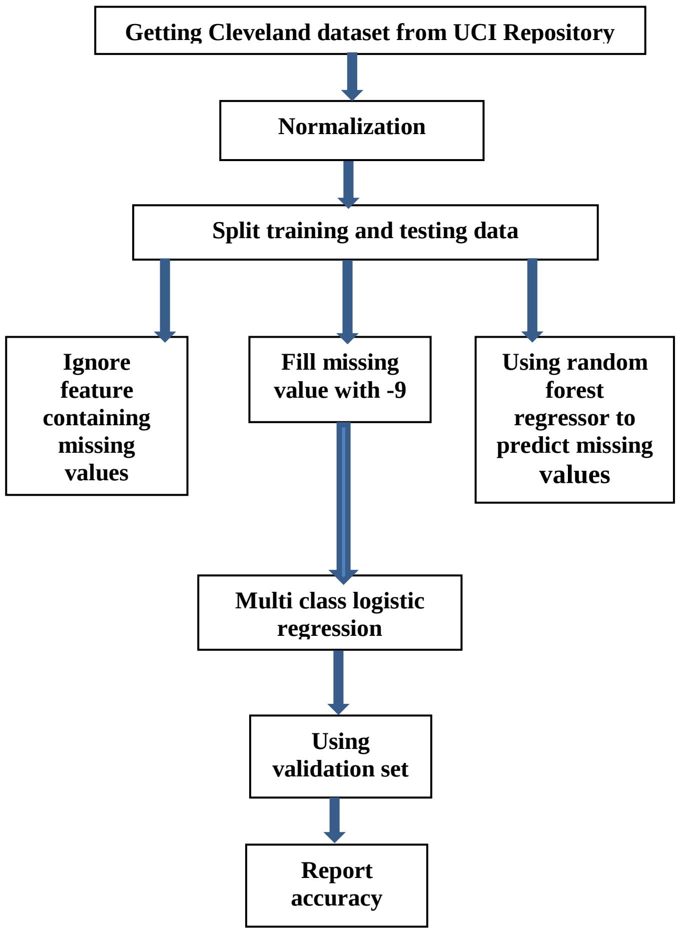
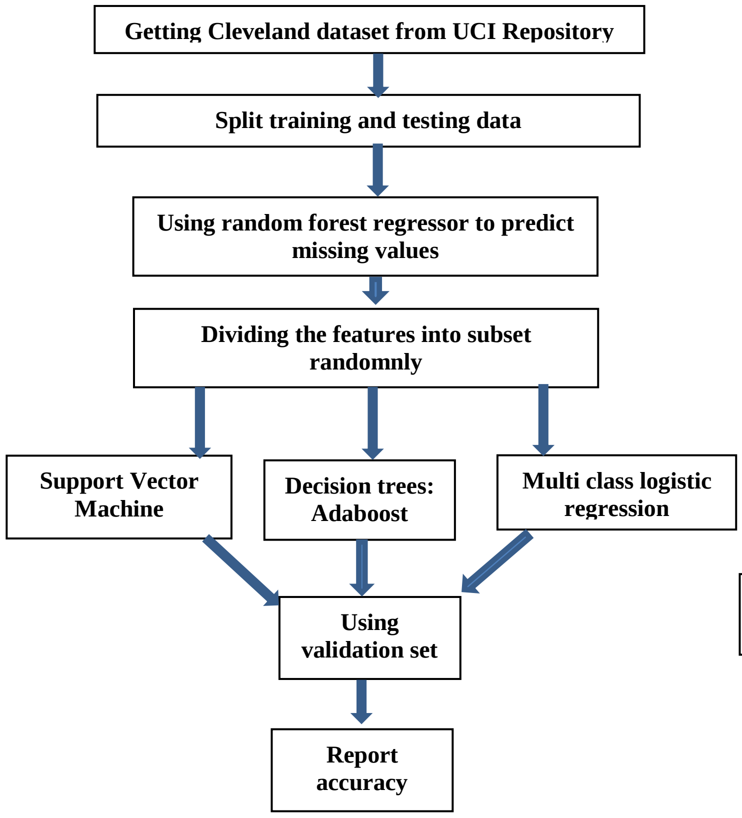

Getting Cleveland dataset from UCI Repository
Normalization
Split training and testing data
Ignore feature containing missing values
Fill missing value with -9
Using random forest regressor to predict missing values
Multi class logistic regression
Using validation set
Report accuracy

Fig. 1. Comparison of methods for predicting Missing Values

B) Multi class classification

1. We import data as text file. Converting that text file into data frame so that it could be used with matrix containing samples and features.
2. Dividing the data to train and test in the ratio of 75:25 with random state .Using random forest regressor to predict the missing values.
3. Dividing the features into subset and carrying out for different number of features.
4. Performing multi class classification using SVM, decision trees(adaboost) and multi class logistic regression.
5. Performing the same experiment with different values of C (error penalty), tol(tolerance) and max iterations.
6. Report the accuracy of above methods.

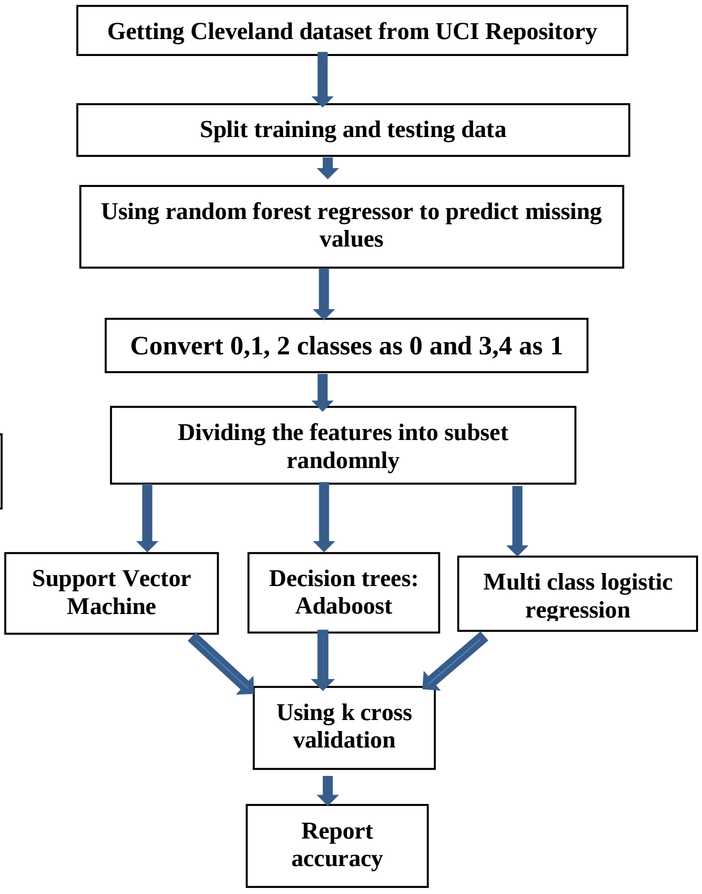


Getting Cleveland dataset from UCI Repository
Split training and testing data
Using random forest regressor to predict missing values
Dividing the features into subset randomnly
Support Vector Machine
Decision trees: Adaboost
Multi class logistic regression
Using validation set
Report accuracy

Fig. 2. Comparison of algorithm for multi class classification

C) Multi class to binary class classification

1. We import data as text file. Converting that text file into data frame so that it could be used with matrix containing samples and features.
2. Dividing the data to train and test in the ratio of 75:25 with random state = 4.Using random forest regressor to predict the missing values.
3. Consider class in 0,1,2 as 0 and 3,4 as 1 to convert multi class to binary class.
4. Dividing the features into subset and carrying out for different number of features.
5. Performing multi class classification using SVM, decision trees(adaboost) and multi class logistic regression.
6. Perform k cross validation where k = n-1 which is the number of features.
7. Performing the same experiment with different values of C (error penalty), tol(tolerance) and max iterations.
8. Report the accuracy of above methods.



Getting Cleveland dataset from UCI Repository
Split training and testing data
Using random forest regressor to predict missing values
Convert 0,1, 2 classes as 0 and 3,4 as 1
Dividing the features into subset randomnly
Support Vector Machine
Decision trees: Adaboost
Multi class logistic regression
Using k cross validation
Report accuracy

Fig. 3. Comparison of algorithm for binary class classification

V. IMPLEMENTATION

It covers the details of the parameters and libraries used for the sections described above.

A) Dealing with missing values

- Feature space

Dataset used: Cleveland data from UCI Repository under file name: processed.cleveland.data[11]:
13 salient features out of 76
1. #3 (age)
2. #4 (sex) 1 = male; 0 = female
3. #9 (cp)
chest pain type
-- Value 1: typical angina
-- Value 2: atypical angina
-- Value 3: non-anginal pain
-- Value 4: asymptomatic

4. #10 (trestbps) resting blood pressure (in mm Hg on admission to the hospital)
5. #12 (chol) serum cholestoral in mg/dl

6. #16 (fbs) fasting blood sugar > 120 mg/dl (1 = true; 0 = false)
7. #19 (restecg) resting electrocardiographic results
-- Value 0: normal
-- Value 1: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV)
-- Value 2: showing probable or definite left ventricular hypertrophy by Estes' criteria

8. #32 (thalach) maximum heart rate achieved

9. #38 (exang) exercise induced angina (1 = yes; 0 = no)
10. #40 (oldpeak) ST depression induced by exercise relative to rest
11. #41 (slope) the slope of the peak exercise ST segment
-- Value 1: upsloping
-- Value 2: flat
-- Value 3: downsloping

12. #44 (ca) number of major vessels (0-3) colored by flourosopy
13. #51 (thal) 3 = normal; 6 = fixed defect; 7 = reversable defect

- Pre- processing and feature extraction

In order to deal with the missing values, following three methods were used:
Ignore features with missing values
Filling in missing values with -9
Using random forest regressor to predict missing values

Ignore: Firstly, 'genfromtxt', function of numpy library was used to import data and tag the missing values as 'nan'.
The features containing missing values, even if one, were located and that whole feature was removed from the data. So feature 11 and 12 was removed.

Fill: Firstly, 'genfromtxt', function of numpy library was used to import data and tag the missing values as 'nan'.
The dataset containing missing values were located and then filled with -9.

Predict: Firstly, 'genfromtxt', function of numpy library was used to import data and tag the missing values as 'nan'. The rows containing 'nan' were tagged as nans matrix which was used as testing and the ones not containing 'nan' was stored as notnans and was used as training data for the random forest regressor. The notnans was again split into train and validation set with the ratio of 75:25.
Random forest got the best result at depth of 30. Using the best parameter, it was used to predict the nan values in test sample. Then the nans and notnans were combined to form the whole dataset.

- Normalization

All feature except the last column vector containing the ouput class(0 -4) was normalized using l1 norm using'normalize' function from scikit library.

- Training process: Multi class logistic regression

The data was divided into train and testing in the rato of 85:15 with random state of 7. Then the training data was again divided into training and testing in the ratio of 85:15 with random state = 8.

Using scikit library, logistic regression was implemented with parameters: multi class = 'multinomial', solver = 'newton-cg', penalty = 'l2', different values of C, max iterations and tolerance(tol).
The model is fitted on X_train and y _train.
Overfitting is prevented using error penalty.

- Validation and testing set

Different combinations of value of C (error penalty), max iterations (max_iteration) and tol(tolerance) were implemented to find the best result on validation set. Predict function is using from scikit library to predict the value on X cross. Then the accuracy is computed using accuracy_score.

Testing set is done using the best value of the parameters. The predict function is used on the testing set and then accuracy score is computed.

Model	Parameter	Validation result
Ignoring features with missing value	C = 10 Max_iter = 100 Tol = 1000	76.92307692
	C = 1 Max_iter = 100 Tol = 10	76.92307692
	C = 1000 Tol = 0.001 Max_iter = 100	69.79487179
Filling missing values with -9.0	C = 10 Max_iter = 100 Tol = 10	71.79487179
	C = 1000 Max_iter = 100 Tol = 0.001	71.78417990
Random forest regressor: Predict values	C = 10 Max_iter = 100 Tol = 10	66.66666667
	C = 1000 Max_iter = 100 Tol = 0.001	79.48717949

B) Multi class classification

- Feature space

Same as part A

- Pre processing and feature extraction

From the Part A result we know that random forest regressor is the best method of all for the given dataset and physics. Using the same method as described predict the missing values.

- Dividing features into subset randomnly

Using random to compute random index to consider those feature and see which subset of features performs well. This is done as the number of samples are less, so as to make comparative number of features with respect to samples.

- Training process

The algorithms used to compare here are: Support Vector Machine, Decision Trees: Adaboost and Multi class logistic regression.

Support Vector Machine: The data was divided into train and testing in the rato of 75:25 with random state of 2. Then the training data was again divided into training and testing in the ratio of 85:25 with random state = 4.

Using scikit library, with different values of C, max iterations and tolerance(tol).
Kernel used is linear.
The model is fitted on X_train and y train.

Decision trees: Adaboost: The data was divided into train and testing in the rato of 75:25 with random state of 5. Then the training data was again divided into training and testing in the ratio of 85:25 with random state = 2.

Using scikit library with parameters (max_depth = 3), n_estimators = 25, learning_rate = 0.0001, algorithm = "SAMME".
Kernel used is linear.
The model is fitted on X_train and y train.

Logistic Regression: The data was divided into train and testing in the rato of 75:25 with random state of 8. Then the training data was again divided into training and testing in the ratio of 75:25 with random state = 8.

Using scikit library, logistic regression was implemented with parameters: multi class = ‘multinomial’, solver = ‘newton-cg’, penalty = ‘l2’, different values of C, max iterations and tolerance(tol).
The model is fitted on X_train and y train.
Overfitting is prevented using error penalty.

- Validation and testing set

Different combinations of value of C (error penalty), max iterations (max_iteration) and tol(tolerance) were implemented to find the best result on validation set. Predict function is using from scikit library to predict the value on X cross. Then the accuracy is computed using accuracy_score.

Testing set is done using the best value of the parameters. The predict function is used on the testing set and then accuracy score is computed.

Model	Parameter	Feature Subset/Validation result

SVM	C = 10
	Tol = 0.01
	Max_depth = 30
	Kernel = linear
		6/ 52.6315789
		3/61.4035088
		10/61.4035088
		2/49.122807
		7/61.4035088
		9/57.8947368
		8/59.6491228

Decision Trees	Max_depth = 3
	n_estimators = 25
	learning_rate = 0.0001
		10/61.4035088
		6/56.1403509
		3/52.6315789
		4/57.8947368
		2/42.1052632
		8/42.1052632
		7/56.1403509
		5/56.1403509

Logistic regression	C = 10
	Max_iter = 1000
	Tol = 0.001
		8/59.64912281
		3/64.9122807
		6/61.40350877
		4/63.15789474
		10/64.9122807
		5/63.15789474
		2/57.89473684
		9/64.9122807

C) Multi class to binary class classification

Exactly same procedure as B except:

- Preprocessing
  Convert class 0,1,2 as 0 and 3,4 as 1

If our data satisfies = y>2 then it is true assign it 1 and if it is false assign it 0.

Using the same condition on all classes and cluster them into groups of 0 and 1.

- Using k cross validation

Using shuffle and split function from scikit model selection to shuffle the split of training set to assign some as training data while some as cross validation data with parameters n_splits = 11, test_size = .25, random_state.

Model	Parameters	Validation Result

SVM	C = 100
	Tol = 0.0001
	Kernel = rbf
		6/84.21053
		8/84.2105263
		2/84.210563
		7/84.2105263
		3/82.4561404
		4/78.9473684
		5/84.2105264
		9/84.2105263

Decision Trees: Adaboost	Max_depth = 3
	N_estimators = 500
	Learning_rate = 0.0001
	Cv_split = 8
		3/52.6315789
		8/42.105632
		7/56.1403509
		2/43.8596491
		6/56.1403509
		4/57.8947368
		5/56.1403509
		9/59.6491228

Logistic regression	C = 100
	Max_iter = 1000
	Tol = 0.0001
	Cv = split = 8
		7/89.2307692
		2/90.7692308
		10/90.7692308
		5/90.76235308
		6/90.7623508
		9/89.2307692
		8/89.2307692
		4/90.7692308

TABLE I. SIMULATION PARAMETERS

VI. RESULTS AND DISCUSSION

After considering the model/ algorithm which performs the for the parameters taken we consider the best parameters and check which model performs the best on the test data.

A) DEALING WITH MISSING VALUES

Model/ Parameters	Test accuracy
Ignoring features with missing value / C = 1000Tol = 0.001 Max_iter = 100	50.00
Filling missing values with -9.0	52.17391304
Random forest regressor: Predict values	54.34782609

B) MULTI CLASS CLASSIFICATION

Model/ Subset	Test accuracy
SVM/10	63.15789474
Decision trees/10	61.8421053
Logistic regression	55.26315789

C) BINARY CLASS CLASSIFICATION

Model/ Subset	Test accuracy
SVM/2	84.2105632
Decision trees/9	64.44736842
Logistic regression/2	78.08695652

TABLE II. SIMULATION PARAMETERS OF TEST DATA

VII. CONTRIBUTION OF EACH MEMBER

This project was done individual by Gunjan Jhawar.

VIII. CONCLUSION

We observe that random forest regressor performs better than ignoring the data or filling the missing values with a random number. It feels intuitive as ignoring the datasets lessen the opportunity to learn from data as we outrightly ignore the whole data even if one data point is missing. The procedure of guessing a value to fill the dataset is difficult because to guess a number, the physics of the data and problem has to be closely looked upon and even then it is not necessary that it will get a good result. While, the random forest constructs a group of decision trees and shoews the best one which fits the data.

We observe that SVM performs better in case of less number of data. As SVM does not need a lot of data to perform well which our result validates.

Logistic regression performs the worse of all. Logistic regression is not really a classification algorithm but can build as a classifier by setting the appropriate threshold value.

We can consider multi class as binary class like 0,1,2 as 0 and 3,4 as 1. Classify them as binary class. Then consider 0,1,2 as multi class and convert that into binary like 0,1 as 0 and 2 as 1 and continue this method. This is the future scope of the project.

REFERENCES

[1] Centers for Disease Control and Prevention (CDC), Heart disease in the United States

[2] https://www.healthcatalyst.com/data-mining-in healthcare

[3] https://machinelearningmastery.com/handle-missing-data-python/

[4] https://developerzen.com/data-mining-handling-missing-values-the-database-bd2241882e72

[5] https://turi.com/learn/userguide/supervised-learning/random_forest_regression.html

[6] Svm based decision support system for heart disease classification with integer-coded genetic algorithm to select critical features –sumit bhatia, praveen prakash, g. n. pillai

[7] Experimental comparisons of multi-class classifiers

[8] Freund y, schapire r e. a decision-theoretic generalization of on-line learning and an

applicationThe HTK Book (for HTK Version 3.4.1). Engineering Department, Cambridge University, Young, S., Evermann, G., Gales, M., Hain, T., Kershaw, D, Liu, X., Moore, G., Odell, J., Ollason, D., Povey, D., Valtchev, V., Woodland, P., 2006.

[9] https://msdn.microsoft.com/enus/library/azure/dn905853.aspx

[10] https://www.cs.cmu.edu/~schneide/tut5/node42.html

[11]http://archive.ics.uci.edu/ml/machine-learning-databases/heart-disease/