CSC 529
Case Study 1: Ensemble Classification
Remya Kannan

Introduction:

Heart disease stands to be the leading cause of death among men and women. Statistics claim that one in every four are affected and possibly die of heart disease. The major challenge with the health care institutions is providing quality services, and this is especially important in underprivileged countries. Faulty clinical decisions could result in severe consequences. With medical datasets at our disposal there is a need to combine tools and the existing machine learning algorithms to intelligentially extract knowledge to predict the occurrence through accurate diagnosis. While many studies have been done to build models to accurately predict the diagnosis, there is a continual need for the search of an effective and efficient model to do the same.

Various literature has introduced new concepts in the model building process of heart disease diagnosis. Machine learning concepts such as classification, clustering, etc. have been incorporated and models developed with good results. Further, new concepts developed involving hybrid models that combine techniques to improve the model. The objective of this study is to find a model that helps predict the occurrence of heart disease based on the diagnostic results with the highest possible accuracy using the concept of ensemble classification. The UCI heart disease data set is used for this purpose.

The key to understanding the diagnostics for heart disease, is learning the cause. With it being the leading cause of death globally, it is vital to learn the occurrence and the symptoms to effectively detect and prevent it. Coronary heart disease is the most common form of heart disease and this occurs due to the plaque buildup on the patient's arteries. The reason for the buildup is due to the increasing consumption of cholesterol, high blood pressure, smoking, etc. and this leads to lowered blood flow causing stroke or heart attack. With development in technology, there are several ways to check the diagnostics to prevent the occurrence of the heart disease. The tests conducted by EKG, X-Rays, etc. provide significant amount of data that can be studied using the machine learning techniques such as SVM, Fuzzy techniques, ANN, etc. Hybrid methods combining various classifiers and clustering algorithms have been incorporated to develop a model with increased accuracy.

This study focuses on the use of the above-mentioned data mining techniques combined with the knowledge of ensemble classification, to develop a model that can predict the occurrence of the heart disease given the diagnostics data. Firstly, the dataset is explored and cleaned for use in the data mining process. Then, some of the boosting techniques (C5.0 and Stochastic gradient boosting) are explored and compared for evaluation followed by bagging techniques (Bagged CART and random forest). The study then moves on to combine classifiers by the stacking method to observe and study the results. Finally, a detailed evaluation on the effects of random forest model is explored and interpreted.

This report is divided as follows: section 1 briefly describes the data highlighting the relevant features of the dataset used in this study, section 2 explains how the data was cleaned in

preparation for model building, data analysis using the various techniques mentioned above is explored in section 3, section 4 highlights the experimental results that discusses the performance of the model, section 5 interprets and analyses the results followed by conclusion in section 6.

1. Data Description:

The heart disease dataset used in this study is from the UCI Machine Learning repository. The database consists of medical data collected from Cleveland Clinic Foundation. There are 76 attributes and 303 number of instances. The target attribute features the presence or absence of the heart disease ranging from 0: absence to 1,2,3,4: presence. For this study, it is denoted as 0: absence and 1: presence of the heart disease. The class label determines the final model prediction. The dataset is divided into training and testing dataset. The training dataset consists of 250 instances and the testing dataset has 53 instances. The feature set consists of 14 attributes where, 13 attributes represent the diagnostic results and one attribute is the target class (0,1).

2. Data Preprocessing:

The data was acquired from the UCI repository and it consists of attributes that characterize data into people with or without heart disease. Data partition aims to split the data into subsets of training and testing data. It is a process by which mutually exclusive data is partitioned using the 10-fold cross validation. To avoid bias, the records are randomly selected for partition. Partitioning helps reduce computation time during model implementation. The training dataset consists of 250 instances and the testing dataset has 53 instances

Feature selection is yet another important component of data preprocessing and involves reduction of the attributes that don't contribute to the model. There are 76 attributes to begin with, however, only 14 of the attributes contribute to the model. The attributes selected are: age (in years), sex of the patient (M=1, F=0), the type of chest pain reported by the patient (typical angina=1, atypical angina=2, non-angina pain=3, asymptotic=4), the blood pressure of the patient (in mm Hg), the cholesterol level of the patient (mg/dl), fasting blood sugar (if >120mg/dl, true=1, false=0), resting ecg results (normal=0, have ST-T wave abnormality=1, probable or definite left ventricular hypertrophy=2), maximum heart rate achieved, exercise induced angina (yes=1, no=0), depression induced by exercise, the slope of the peak exercise (upsloping=1, flat=2, down sloping=3), Number of major vessels (0-3), thal (normal=3, fixed defect=6, reversible defect=7) and diagnosis of the heart disease (yes=1, no=0).

Data cleaning takes the raw data with the selected features to clean them of the missing values and transforms it to be used by the model. It takes places over individual attributes before they are fed to the classifiers. Missing values in medical dataset can be serious as this could mean loss of information that is detrimental to diagnosis. There is a need for missing value imputation, that are replaced by the mean or 0. For this dataset, there were several missing values for thal and the blood pressure of the patients. The missing values are replaced by NA and on studying the dataset, ultimately replaced NA with 0 for model building purposes. This comes with the limitation of information loss, but the other attributes contributing towards the model make up for this loss. The data was not noisy and there were very few outliers that affected the model.

3. Data analysis:

Following the data cleaning process, the next stage of the study moves on to the exploratory analysis of the data. There is a need to find out if the variables are correlated or if outliers are present before we proceed to the model building process to avoid error in prediction. Thus, exploratory analysis can be divided into univariate analysis and bivariate analysis for this study.

3.1. Univariate analysis:

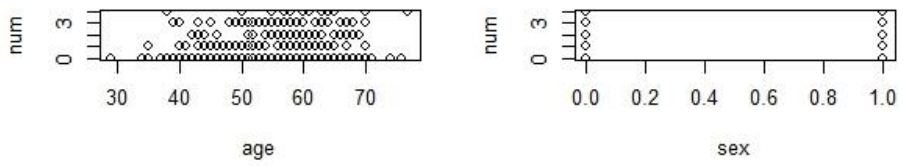
Univariate analysis includes exploring the summary statistics of the data to learn how the data is centered. The number summary of the data is calculated and the following results are obtained:

	n	mean	sd	max	min	range	nunique	nzeros	iqr	lowerbound	upperbound	noutlier
num	303	0.937	1.23	4	0	4	5	164	2	-3	5	0

	kurtosis	skewness	mode	miss	miss%	1%	5%	25%	50%	75%	95%	99%
num	-0.175	1.05	0	0	0	0	0	0	0	2	3	4

We observe that the mean of the dataset is 0.937 and the standard deviation is 1.23. It also shows that there are no outliers or missing values in the dataset. Each of the attributes are now plotted against the dependent variable (diagnosis) as shown in Fig.1.

Figure Number = 1



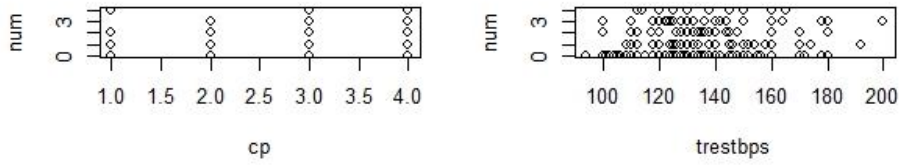
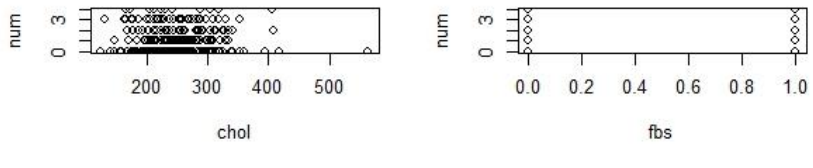
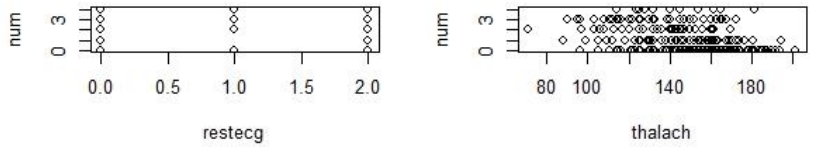


Figure Number = 2





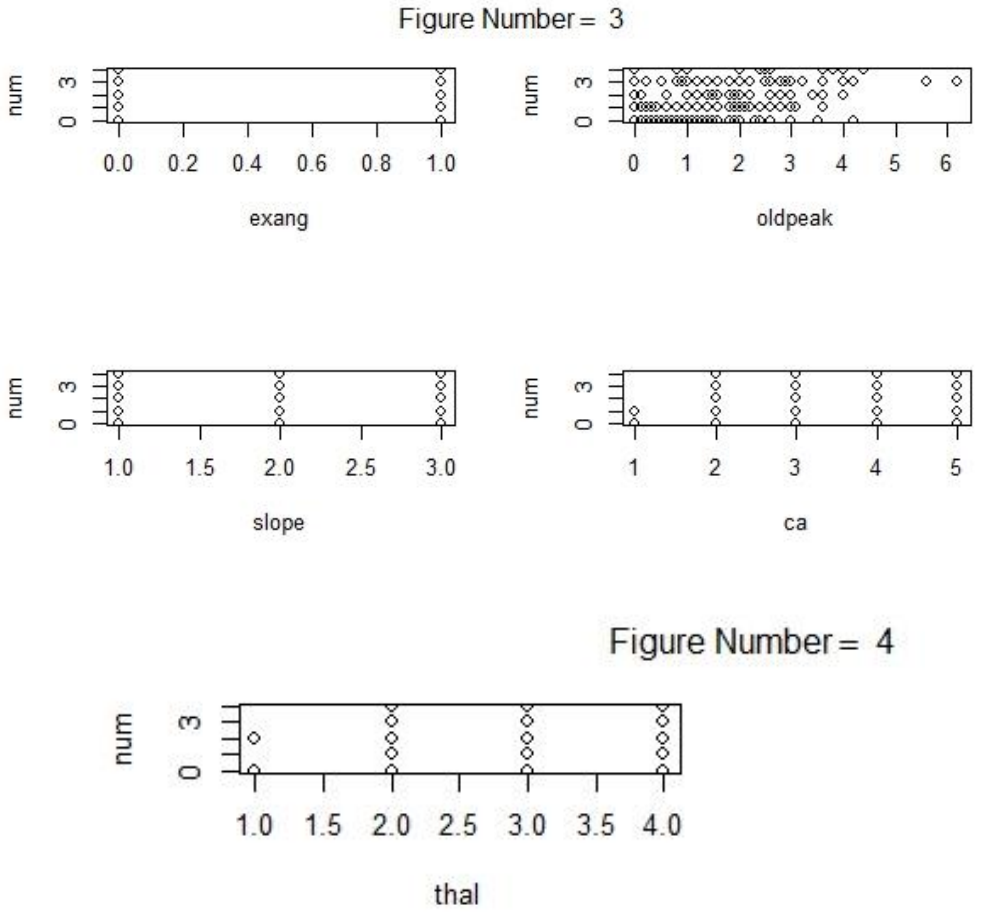


Fig 1. Plot of dependent vs independent attributes

To determine if there are any outliers, the box plot is drawn and from the plot shown in Fig.2, we observe that there are no outliers.

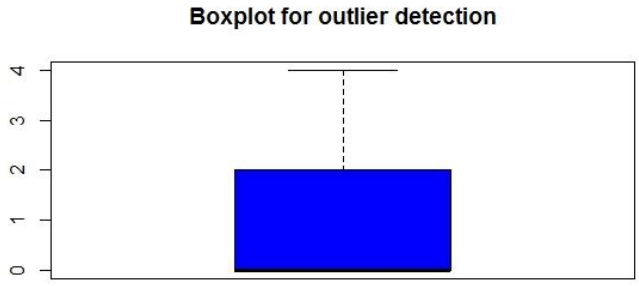


Fig 2. Boxplot for outlier detection

3.2.Bivariate Analysis:

Bivariate analysis on the data results in finding any relationship between the variables, such as correlation, covariance, etc. The correlation plot is shown in Fig.3. From the plot, we observe that attributes such as maximum heart rate achieved (thalach), exercise induced angina (exang) and the number of major blood vessels (ca) are somewhat

correlated with the other variables, but not significant enough (<0.7) to affect the model building process.

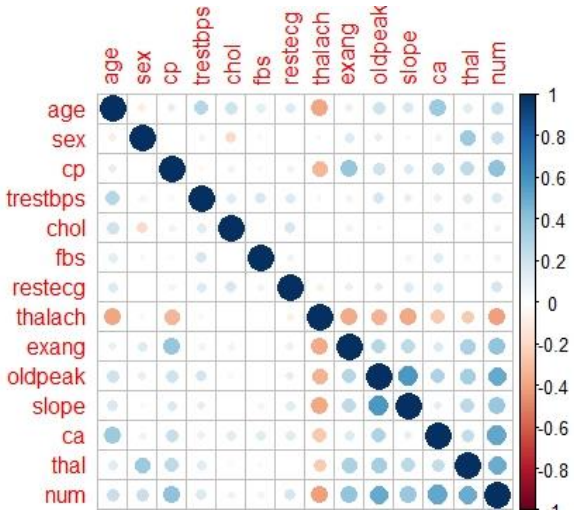


Fig.3. Correlation plot of the variables

3.3. Multivariate analysis:

PCA can be used as a technique for dimensionality reduction. It helps in finding those attributes that contribute towards the model building process. Thus, the results from PCA helped determine that not all the 76 attributes are useful for model building and only 14 attributes contribute to any variance in the data. As seen from the PCA plot in Fig.4 and the summary of the results below, we observe that it takes at least 12 attributes to explain 95% of the variance in the data and hence proves the importance of the selected attributes for model building.

```
Importance of components:
                          PC1    PC2    PC3    PC4    PC5    PC6    PC7    PC8    PC9
Standard deviation     1.91 1.269 1.1103 1.0550 0.9942 0.9488 0.9372 0.8896 0.8472
Proportion of Variance 0.26 0.115 0.0881 0.0795 0.0706 0.0643 0.0627 0.0565 0.0513
Cumulative Proportion  0.26 0.375 0.4628 0.5423 0.6129 0.6772 0.7400 0.7965 0.8478
                         PC10   PC11   PC12   PC13   PC14
Standard deviation     0.7526 0.6863 0.6527 0.5953 0.5598
Proportion of Variance 0.0405 0.0336 0.0304 0.0253 0.0224
Cumulative Proportion  0.8882 0.9219 0.9523 0.9776 1.0000
```

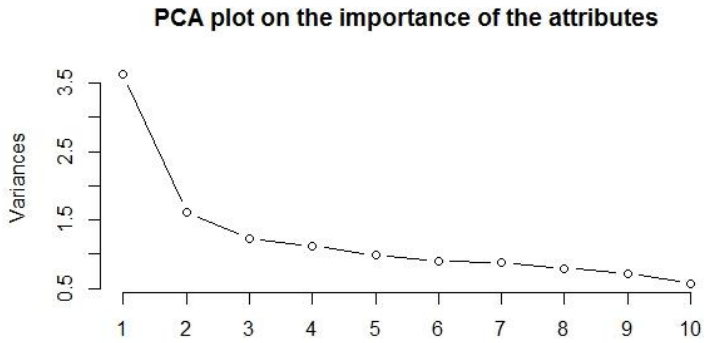


Fig 4. PCA plot on importance of attributes

4. Experimental Results:

This section highlights the results obtained from the ensemble techniques used in the study. The main technique studied in detail is the Random Forest approach. However, to learn the concept of ensemble classification in detail and to compare and contrast between different techniques, methods including boosting algorithms, bagging algorithms and stacking are also explored.

4.1.Boosting algorithms:

Boosting algorithms help convert weak learners to strong learners and subsequently helps improve the accuracy of prediction of the model using the concept of ensemble learning. The algorithms explored in this model are the C5.0 and Stochastic gradient boosting.

4.1.1. C5.0:

The results of C5.0 are as follows:

```
C5.0

303 samples
 13 predictor
  2 classes: 'No', 'Yes'

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 273, 272, 273, 274, 272, 272, ...
Resampling results across tuning parameters:

  model  winnow  trials  Accuracy  Kappa
  rules  FALSE    1      0.803     0.601
  rules  FALSE   10      0.796     0.588
  rules  FALSE   20      0.803     0.603
  rules   TRUE    1      0.785     0.565
  rules   TRUE   10      0.797     0.590
  rules   TRUE   20      0.793     0.582
  tree   FALSE    1      0.784     0.564
  tree   FALSE   10      0.796     0.587
  tree   FALSE   20      0.793     0.581
  tree    TRUE    1      0.780     0.557
  tree    TRUE   10      0.788     0.572
  tree    TRUE   20      0.789     0.575

Accuracy was used to select the optimal model using  the largest value.
The final values used for the model were trials = 1, model = rules and winnow
 = FALSE.
```

The accuracy of the model is 80.3%.

4.1.2. Stochastic Gradient Boosting:

The results of gradient boosting are as follows:

```
Stochastic Gradient Boosting

303 samples
 13 predictor
  2 classes: 'No', 'Yes'

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 273, 272, 273, 274, 272, 272, ...
Resampling results across tuning parameters:

  interaction.depth  n.trees  Accuracy  Kappa
  1                   50      0.831     0.657
  1                  100      0.830     0.656
  1                  150      0.834     0.665
  2                   50      0.824     0.644
  2                  100      0.820     0.635
  2                  150      0.814     0.624
  3                   50      0.814     0.624
  3                  100      0.811     0.618
  3                  150      0.805     0.606

Tuning parameter 'shrinkage' was held constant at a value of 0.1
Tuning
 parameter 'n.minobsinnode' was held constant at a value of 10
Accuracy was used to select the optimal model using  the largest value.
The final values used for the model were n.trees = 150, interaction.depth =
 1, shrinkage = 0.1 and n.minobsinnode = 10.
```

The accuracy of the model is 83.1%.

4.1.3. Summary of the results:

The summary of the results of both the boosting algorithms are as follows:

```
Call:
summary.resamples(object = boosting_results)

Models: c5.0, gbm
Number of resamples: 30

Accuracy
      Min. 1st Qu. Median  Mean 3rd Qu.  Max. NA's
c5.0 0.677   0.733  0.800 0.803   0.839 0.967    0
gbm  0.700   0.800  0.836 0.834   0.870 0.967    0

Kappa
      Min. 1st Qu. Median  Mean 3rd Qu.  Max. NA's
c5.0 0.340   0.461  0.596 0.601   0.679 0.932    0
gbm  0.405   0.594  0.672 0.665   0.738 0.933    0
```

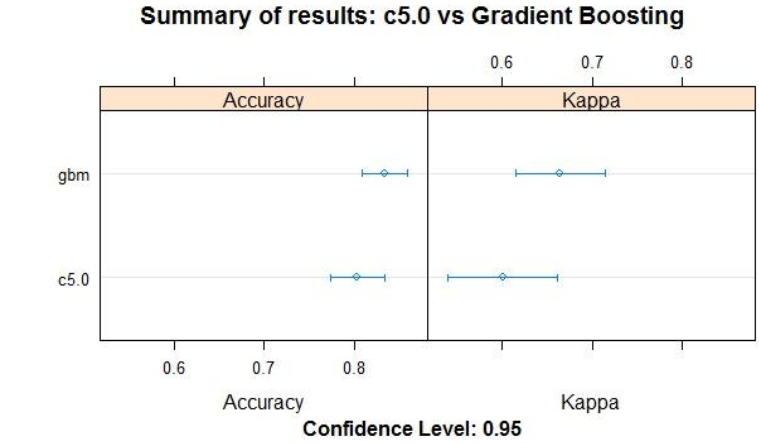


Fig.5. Summary of results of boosting algorithms

As seen from the result above, we can confirm that the Gradient Boosting algorithm has a higher accuracy than the C5.0 boosting algorithm,

4.2. Bagging Algorithms:

Bagging algorithms help improve the stability and accuracy of the model using ensemble learning techniques. The bagging algorithms used in this study are: Bagged CART and Random Forests.

4.2.1. Bagged CART:

The results of this approach are as follows:

```
Bagged CART

303 samples
 13 predictor
  2 classes: 'No', 'Yes'

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 273, 272, 273, 274, 272, 272, ...
Resampling results:

  Accuracy  Kappa
  0.807     0.61
```

From the result above, the accuracy of the model is 80.7% using this algorithm.

4.2.2. Random Forest:

The results are as follows:

```
Random Forest

303 samples
 13 predictor
  2 classes: 'No', 'Yes'

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 273, 272, 273, 274, 272, 272, ...
Resampling results across tuning parameters:

  mtry  Accuracy  Kappa
   2    0.827     0.649
  10    0.797     0.590
  18    0.801     0.598

Accuracy was used to select the optimal model using  the largest value.
The final value used for the model was mtry = 2.
```

Using Random tree bagging algorithm, the accuracy of the model is 82.7%.

4.2.3. Summary of results:

The summary of the results of both the bagging algorithms are as follows:

```
Call:
summary.resamples(object = bagging_results)

Models: treebag, rf
Number of resamples: 30

Accuracy
         Min. 1st Qu. Median  Mean 3rd Qu.  Max. NA's
treebag 0.667   0.767  0.806 0.807   0.860 0.933    0
rf      0.710   0.779  0.833 0.827   0.867 0.966    0

Kappa
         Min. 1st Qu. Median  Mean 3rd Qu.  Max. NA's
treebag 0.332   0.527  0.609 0.610   0.720 0.864    0
rf      0.418   0.548  0.662 0.649   0.733 0.930    0
```

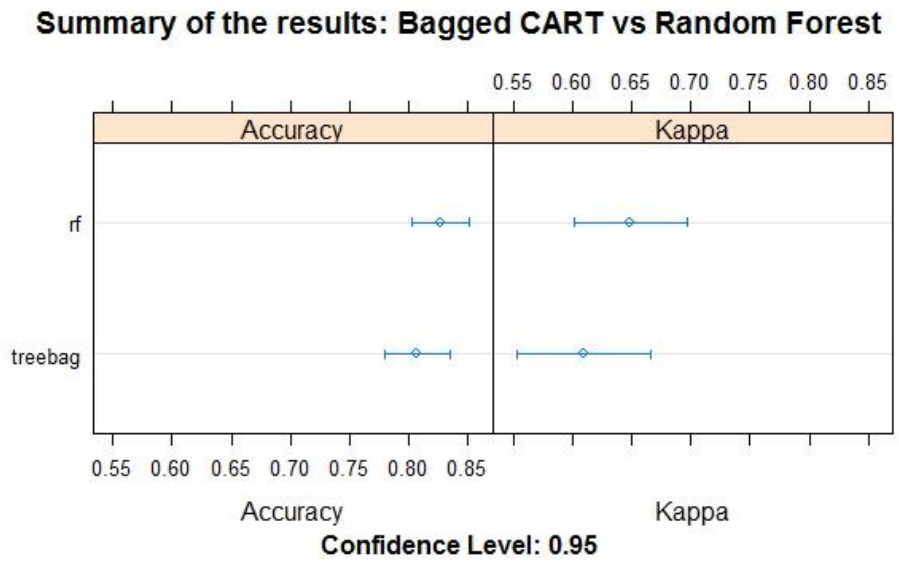


Fig.6. Summary of results of bagging algorithms

From the above results, we may conclude that Random forest has higher accuracy than bagging CART algorithm.

4.3.Stacking Algorithms:

Stacking algorithms are a method of ensemble learning that incorporates multiple classifiers to improve the accuracy of the model. First regular stacking is attempted using an ensemble of classifiers such as lda, glm, svm, etc. followed by stacking using glm vs random forest to see which method works best.

4.3.1. Stacking:

The results of stacking using an ensemble of classifiers is as follows:

```
call:
summary.resamples(object = results)

Models: lda, rpart, glm, knn, svmRadial
Number of resamples: 30

Accuracy
            Min. 1st Qu. Median  Mean 3rd Qu.  Max. NA's
lda       0.710   0.800  0.833 0.837   0.871 0.967    0
rpart     0.645   0.708  0.767 0.775   0.833 0.966    0
glm       0.710   0.795  0.820 0.830   0.870 0.966    0
knn       0.500   0.600  0.639 0.659   0.731 0.833    0
svmRadial 0.667   0.781  0.833 0.825   0.870 0.933    0

Kappa
              Min. 1st Qu. Median  Mean 3rd Qu.  Max. NA's
lda         0.3950   0.595  0.665 0.670   0.743 0.933    0
rpart       0.2790   0.408  0.527 0.545   0.663 0.930    0
glm         0.3950   0.579  0.641 0.656   0.735 0.931    0
knn        -0.0181   0.189  0.263 0.307   0.457 0.658    0
svmRadial   0.3360   0.559  0.662 0.648   0.735 0.867    0
```

The correlation between the models were calculated and the correlation matrix is as follows:

	lda	rpart	glm	knn	svmRadial
lda	1.000	0.5923	0.880	0.2871	0.861
rpart	0.592	1.0000	0.619	0.0854	0.680
glm	0.880	0.6188	1.000	0.4241	0.833
knn	0.287	0.0854	0.424	1.0000	0.199
svmRadial	0.861	0.6798	0.833	0.1992	1.000

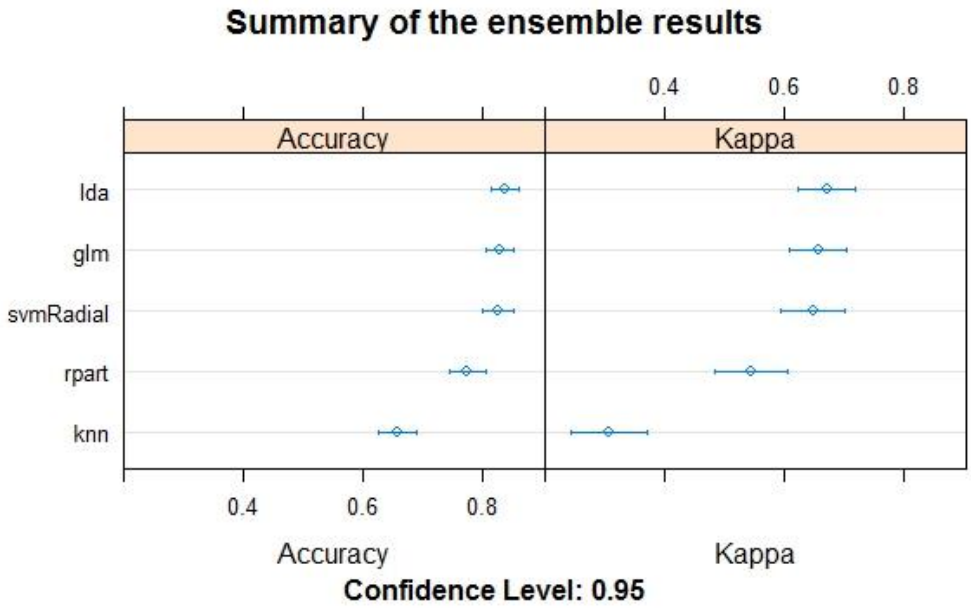


Fig.7. Summary of the ensemble results

4.3.2. Stacking using linear regression:

The results of stacking using glm are as follows:

```
A glm ensemble of 2 base models: lda, rpart, glm, knn, svmRadial

Ensemble results:
Generalized Linear Model

909 samples
  5 predictor
  2 classes: 'No', 'Yes'

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 818, 818, 818, 818, 818, 818, ...
Resampling results:

  Accuracy  Kappa
  0.831     0.658
```

Using linear regression combined with the stacking model, an accuracy of 83.1% is achieved.

4.3.3. Stacking using random forest:

The results of stacking using random forest is as follows:

```
A rf ensemble of 2 base models: lda, rpart, glm, knn, svmRadial

Ensemble results:
Random Forest

909 samples
  5 predictor
  2 classes: 'No', 'Yes'

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 818, 818, 818, 818, 818, 818, ...
Resampling results across tuning parameters:

  mtry  Accuracy  Kappa
  2     0.843     0.682
  3     0.838     0.673
  5     0.837     0.671

Accuracy was used to select the optimal model using  the largest value.
The final value used for the model was mtry = 2.
```

The random forest technique combined with the stacking model increased the accuracy by about 84.3%. Thus, this is a better model than the one using linear regression.

4.4.Random Forest:

Random forests incorporate multiple deep decision trees to average their results to train a model. This reduces the possibility of over-fitting which is a challenge when working with decision trees. This section gives the result of the Random Forest technique. The dataset is divided into training and testing data. First, we develop a singular model tree. Subsequently, we develop the random forest on this model tree. The random seed is set to 1234. The singular model tree is built and tested on the testing data with the following misclassification table:

```
Confusion Matrix and Statistics


pred.model.tree No Yes
            No  24   5
            Yes  8  16

               Accuracy : 0.755
                 95% CI : (0.617, 0.862)
    No Information Rate : 0.604
    P-Value [Acc > NIR] : 0.0156

                  Kappa : 0.5
 Mcnemar's Test P-Value : 0.5791

            Sensitivity : 0.750
            Specificity : 0.762
         Pos Pred Value : 0.828
         Neg Pred Value : 0.667
             Prevalence : 0.604
         Detection Rate : 0.453
   Detection Prevalence : 0.547
      Balanced Accuracy : 0.756

       'Positive' Class : No
```

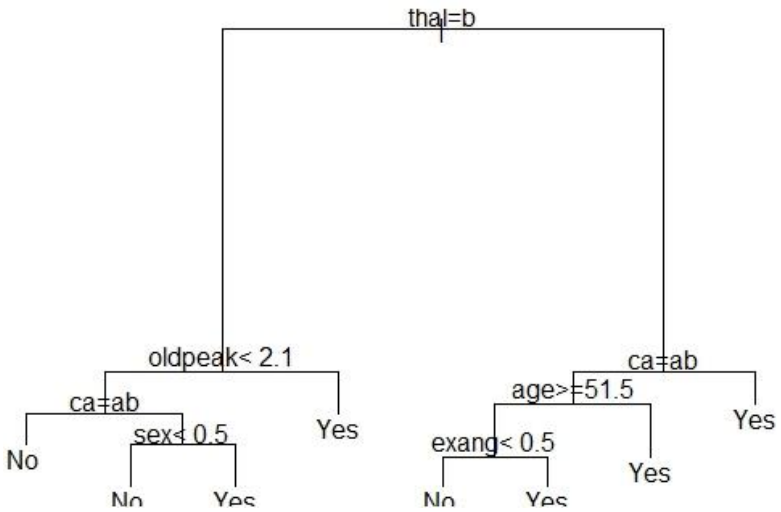


Fig.8. Singular model tree

After a sample of the data was created, the Random forest model was run for varying depths of forest (20, 30, 50, 75). The ideal depth was found to be 75 with the following results:

```
Call:
 randomForest(formula = num ~ ., data = train, ntree = 75)
               Type of random forest: classification
                     Number of trees: 75
No. of variables tried at each split: 3

        OOB estimate of  error rate: 19.2%
Confusion matrix:
     No Yes class.error
No  110  22       0.167
Yes  26  92       0.220
```

With the selection for the ideal depth, the next step would be to tune the forest to select the optimal number of variables at each split. The results are as follows:

```
      mtry OOBError
2.OOB    2    0.224
3.OOB    3    0.200
4.OOB    4    0.200
> print(best.m)
3.OOB 4.OOB
    3     4
```

Now, adding these optimal variable parameters to build the random forest, we get the following result:

```
Call:
 randomForest(formula = num ~ ., data = train, mtry = best.m,      importance = TRUE, nt
ree = 75)
               Type of random forest: classification
                     Number of trees: 75
No. of variables tried at each split: 3

        OOB estimate of  error rate: 19.2%
Confusion matrix:
     No Yes class.error
No  110  22       0.167
Yes  26  92       0.220
```

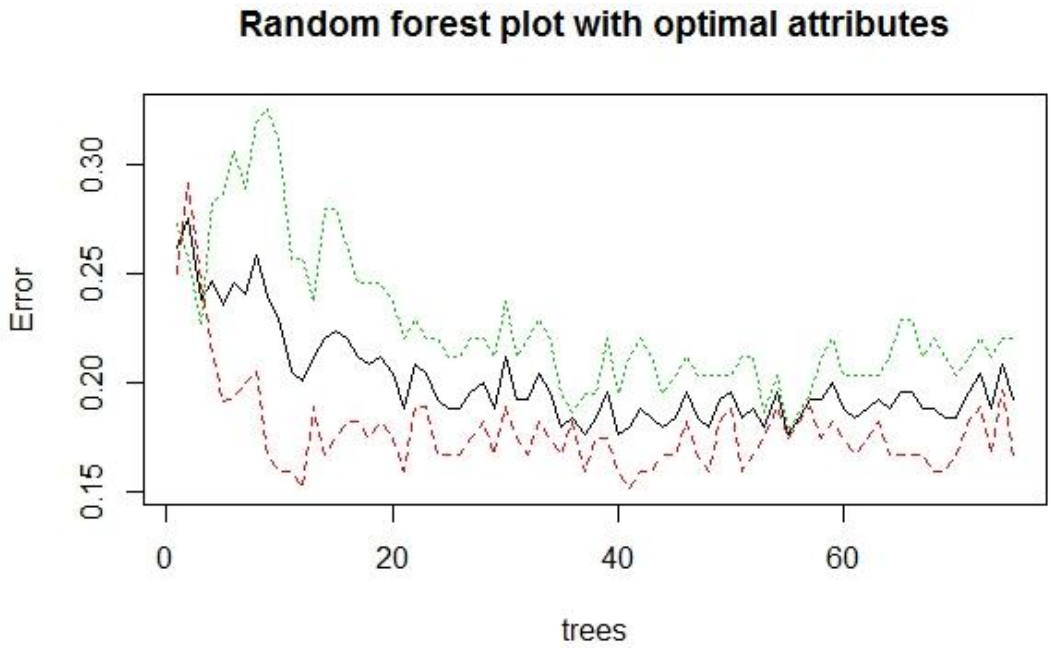


Fig.9. Random forest plot with optimal variables

Now that the model is built, we look at the plot shown below to understand the model from its coefficients, i.e., the effect of each of the variables on prediction/classification:

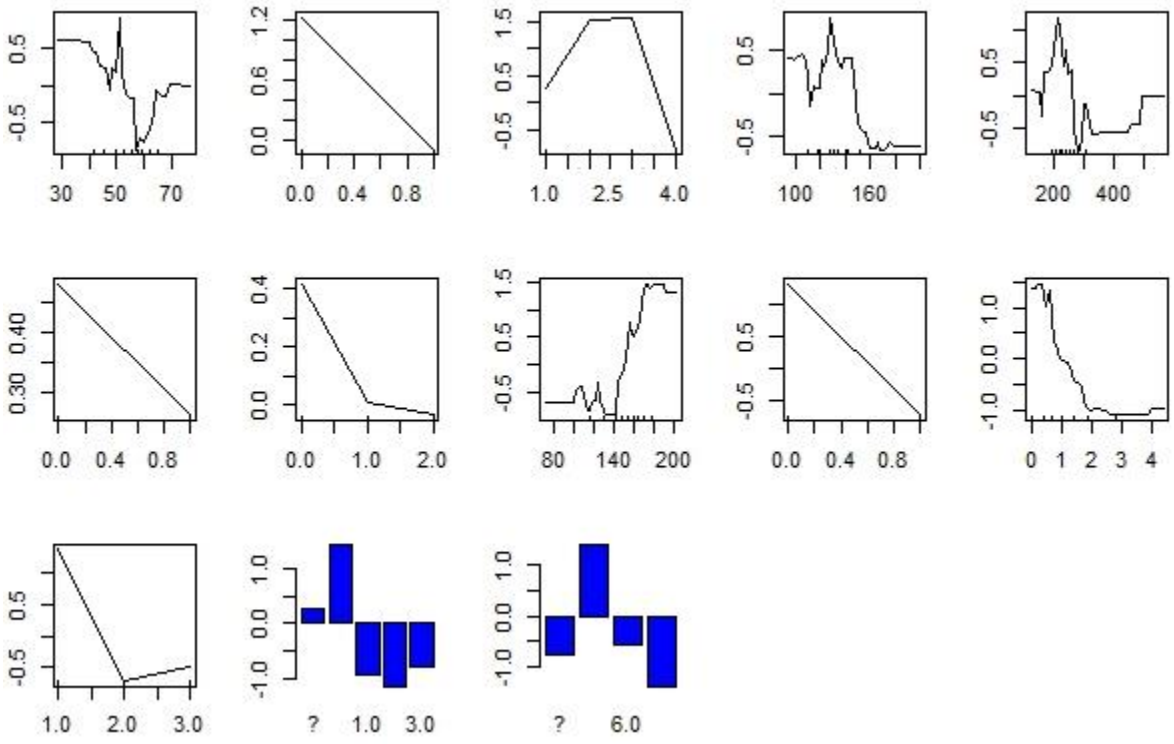


Fig.10. Partial plot of model

Now that the model is built on the training dataset, it is tested on the test dataset and evaluated using performance metrics such as ROC, AUC, the True positive and true negative rate.

The results from running the model on the test set is as follows:

```
Confusion Matrix and Statistics


pred.randF No Yes
       No  26   4
       Yes  6  17

               Accuracy : 0.811
                 95% CI : (0.68, 0.906)
    No Information Rate : 0.604
    P-Value [Acc > NIR] : 0.00105

                  Kappa : 0.612
 Mcnemar's Test P-Value : 0.75183

            Sensitivity : 0.812
            Specificity : 0.810
         Pos Pred Value : 0.867
         Neg Pred Value : 0.739
             Prevalence : 0.604
         Detection Rate : 0.491
   Detection Prevalence : 0.566
      Balanced Accuracy : 0.811

       'Positive' Class : No
```

The next step is to evaluate the model and the first step towards that is to calculate the variable importance. The results are as follows:

	No	Yes	MeanDecreaseAccuracy	MeanDecreaseGini
age	1.984	0.079	1.562	10.82
sex	6.119	2.727	6.692	5.25
cp	3.672	4.602	5.155	11.13
trestbps	1.434	-0.852	0.369	8.76
chol	-1.473	-1.090	-1.700	9.23
fbs	0.226	-2.281	-1.252	1.22
restecg	0.776	1.018	1.122	2.47
thalach	4.125	1.326	4.215	15.19
exang	1.727	3.736	4.074	6.18
oldpeak	5.332	3.027	6.164	12.48
slope	2.862	3.077	3.592	6.76
ca	6.626	6.960	9.029	15.99
thal	7.998	5.650	8.584	17.33

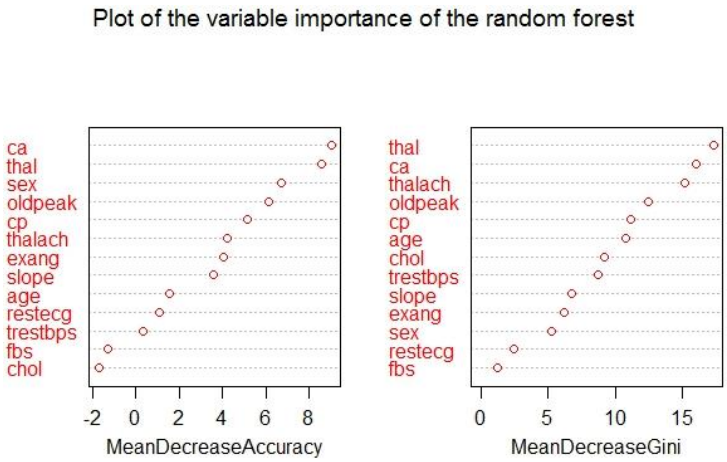


Fig.11. Variable importance

Subsequently, the AUC, ROC and accuracy were calculated. The AUC value of the model is 0.89, the accuracy of the model is 81.1% and the ROC curve is as seen below:

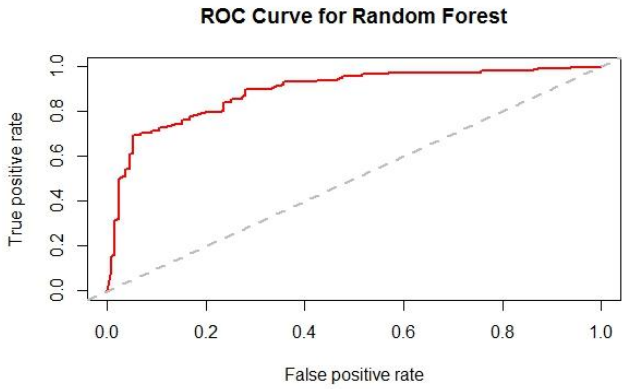


Fig.12. ROC curve of the model

5. Experimental Analysis:

The objective of this study is to find a model using the ensemble learning techniques that can predict the diagnosis of the presence or absence of the heart disease in a patient based on the data from various tests such as ECG, EKG, etc. This could aid many medical institutions to predict the occurrence of heart disease given certain parameters and in turn help prevent it.

This study started with the exploration of the dataset. After the initial data analysis of missing values imputation, feature selection and check for correlation, the data was deemed good for model building.

In this study, various ensemble learning classification and prediction models were designed and developed. The model was used to diagnose and classify the presence or

absence of heart disease. First, models with C5.0 and stochastic gradient boosting algorithms were developed. C5.0 model resulted in an accuracy of 80.3% and the model using gradient boosting algorithm had an accuracy of 83.1%. We can see that the Gradient boosting algorithm produces a more accurate model with an accuracy of 83.1% as seen in Fig.5. Next the models used bagging algorithms like bagging CART and random forest. The model using bagging CART has accuracy of 80.7% while the model using random forest algorithm has an accuracy of 82.7%. Thus, the model using random forests performed better, as shown in Fig.6. The next technique used was stacking algorithms where an initial stacking model was developed using classifiers, LDA, Regression Trees, Logistic regression, knn and svm. We observe that LDA and glm creates a more accurate model with an accuracy of 71%, as shown in Fig.7. To perform stacking and get good results, we need for the models to have low-correlation. If the predictions for the sub-models, are highly correlated, then they would make similar predictions, which is not helpful. On running tests for correlation, we observe that all pairs of predictions have low correlation (<0.7). Combining the predictions of the classifiers using simple linear model increased the accuracy to 83.1%. This was higher than any of the models tested above. Using random forests, we see even more increase in the accuracy to 84.3%. This led to building a model with the random forest ensemble and exploring the results in detail.

Random forests comprise of an ensemble of classification trees that are calculated on random subsets of data with a random set of variables and helps to increase the accuracy of the model. Since they are truly random in nature, on dividing the dataset into training and testing data, we set a random seed to ensure data is selected at random. Initially, a model tree is developed and tested using the testing dataset. From the confusion matrix, we can conclude with 95% confidence that the model is appropriate for prediction as the p-value < 0.05(0.01) and will predict the outcome with an accuracy of 75.5%.

We now run the model with an ensemble of random forest algorithm with the varying depths of 30,50 and 75. We found the model with the forest depth of 75 to perform the best as the accuracy was low in the other models. On finding the ideal depth, the next focus is on finding the optimal number of parameters to be used in each split by tuning the forest. The optimal number of attributes per split was found to be 3 and this is then fed to the new model running the random forest algorithm. Now that we have developed our model, we look at the plot of partial dependence of each predictor in Fig.10, to understand the structure of the model by examining its coefficients. We then test the model on the testing set and run performance metrics on the model. The first metric to be tested is the attribute importance. Higher the value of mean decrease accuracy or mean decrease gini score, higher the importance of the attribute in the model. Thus, in the plot in Fig.11, thal is the most important attribute in the model followed by the number of major blood vessels, the heart rate achieved and the type of chest pain. If we observe the variables, they are important in the diagnosis of the heart disease as they portray the key traits in the medical data of the patient. We now calculate the ROC and the AUC as the next metric to rate the model's performance. Looking at the ROC curve in Fig.12, we observe that the plot is towards the true positive side, which means that very few/none classes are misclassified. As the ROC gets closer to the optimal point of perfection, AUC

gets closer to 1. The AUC value for the model is 0.89 which indicates that the model is appropriate for prediction. This is further confirmed by the true positive rate and the true negative rate. This is very important especially in the dataset under consideration as any misclassification error can prove to be fatal. **Thus, we may conclude that we are 95% confident that the model can predict the diagnosis of a patient with heart disease with an accuracy of 81.1% as the p-value is <0.05**.

Conclusion:

The objective of the study is to find an effective and efficient model using the ensemble learning techniques that can predict the diagnosis of the presence or absence of the heart disease in a patient based on the medical data with relatively good accuracy. The heart disease dataset from Cleveland Clinic Foundation is used for this purpose. The dataset has 303 instances and 76 attributes and was split into training and testing data for model building.

After data cleaning, ensemble techniques such as boosting, bagging, and stacking algorithms are implemented. The model using the C5.0 model resulted in an accuracy of 80.3% and the model using gradient boosting algorithm had an accuracy of 83.1%. Models that used bagging algorithms such as classification and regression tree bagging and random forest, resulted in an accuracy of 80.7% and 82.7% respectively. The next approach explored is stacking algorithm that uses algorithms such as, LDA, Regression Trees, Logistic regression, knn and svm for model development. The model is then run using linear regression and random forest and resulted in an accuracy of 83.1% and 84.3% respectively. Thus, the stacking model using random forests performed the best.

The study then focuses to work on a model using random forest. A singular model tree is built with an accuracy of 75.5%. Using random forest of depth 75, a model is developed resulting in an accuracy of 81.1% with an AUC value of 0.89. Thus, we may conclude that an effective and efficient model is built using random forest with relatively higher accuracy to predict the diagnosis of heart disease in a patient.

For future work, a variety of other ensemble learning techniques may be applied such as Adaboost and stacking algorithms can be used in datasets to develop a better model. This study can also extend over other datasets to see the impact of ensemble learning in different fields of analysis. There is a never-ending search for a model that can provide results with high accuracy with minimum computational complexity that can be used in a variety of fields as sensitive as the medical data that could potentially help the society and the less privileged.