Bench Marking of Classification Algorithms: Decision Trees and Random Forests – A Case Study using R

Manish Varma Datla
Department of Computer Science Engineering
Manipal Institute of Technology, Manipal
Manipal – 576104, India
mvdmanish4@gmail.com

***Abstract*—Decision Trees and Random Forests are leading Machine Learning Algorithms, which are used for Classification purposes. Through the course of this paper, a comparison is made of classification results of these two algorithms, for classifying data sets obtained from Kaggle's Bike Sharing System and Titanic problems. The solution methodology deployed is primarily broken into two segments. First, being Feature Engineering where the given instance variables are made noise free and two or more variables are used together to give rise to a valuable third. Secondly, the classification parameters are worked out, consisting of correctly classified instances, incorrectly classified instances, Precision and Accuracy. This process ensured that the instance variables and classification parameters were best treated before they were deployed with the two algorithms i.e. Decision Trees and Random Forests. The developed model has been validated by using Systems data and the Classification results. From the model it can safely be concluded that for all classification problems Decision Trees is handy with small data sets i.e. less number of instances and Random Forests gives better results for the same number of attributes and large data sets i.e. with greater number of instances. R language has been used to solve the problem and to present the results.**

Keywords—Decision Trees, Random Forests, Feature Engineering, Instance Variables, R language.

I. INTRODUCTION

The fundamental to intelligent behavior lies in the ability to learn classifications while handling data. Decision Trees and Random Forests lie at the heart of classification and are the most popular algorithms used for handling data. While Decision Trees find data features and thresholds that best splits the data into separate classes, Random Forests are an ensemble of Decision Trees. Other techniques like the Boosted decision trees, ID3 and C4.5 learn from their accuracy on the data. Each classifier is trained one by one, data that is poorly represented by earlier classifiers are given a higher weight age so that subsequent classifiers pay more attention to points where the errors are large.

In Decision Trees the ability to find data features and thresholds that best splits the data into separate classes happens recursively until data has been split into homogeneous or mostly homogeneous groups. Any machine learning algorithm typically works in two phases, one being the training phase and the other being the testing or the performance phase. During the training phase, the Decision Tree is built upon the existing instances available with the training data-set and during the testing or performance phase the newly available instances are classified on the basis of the model obtained from the training data-set.

Decision Trees is simple but yet a very effective classification algorithm. The effectiveness of Decision Trees as a classification algorithm lies in its ability to immediately identify the most important features. Human readable rules of classification obtained by using Decision Trees is one of the biggest advantages that this classification algorithm provides.

Using a data structure called Peano Count Tree (P-tree), decision trees were provided with a new method for classification. While Decision Trees exhibited fast speeds at test times, they were relatively slow during training time and hence were provided with a new data structure (P-tree) for classification. Prior to the usage of P-tree as a data structure, decision trees were rendered impractical for applications with real-time learning requirements [3], but the usage of P-tree had enhanced the efficiency and scalability of classification. One of the major drawbacks that Decision Trees pose is its inability to sort all numerical attributes in order to decide where to split a node. This in-turn results in Decision Trees becoming costly both in terms of space and time when it is operated on large data-sets i.e., data with more number of instances.

Random Forests are an ensemble of Decision Trees. During learning, tree nodes are split using a random subset of data features. All trees vote to produce a final answer. A novel on-line random forest algorithm, combining ideas from on-line bagging, extremely randomized forests and an on-line decision tree growing procedure was proposed by several authors. Additionally, a temporal weighting scheme was added for adaptively discarding some trees based on their out-of-bag-

978-1-4673-6667-0/15/$31.00

error in given time intervals and consequently growing of new trees. So at each decision split the features are randomly selected in Random Forests. Random selection of the features improves the prediction power and results in higher efficiency, and thereby reduces correlation between trees. Through the use of bagging on samples and the voting scheme through which the decision is made, under most circumstances Random Forests provides with a better performance when compared to Decision Trees.

In this paper, Decision Trees and the Random Forests classification methods are put to test, to evaluate their performance in classification of the large and small datasets. The idea is to establish the optimum method to be used for classification based on the size of datasets. Data obtained from Kaggel's "Forecast use of a city bike share system" problem and "Titanic: Machine Learning from Disaster" problem have been employed, in-order to study the nature of these two Machine learning algorithms. Given data-set has been primarily segmented into two sets .i.e. the training data set and the test data set. First the algorithm is developed and trained on the training data-set. The methodology obtained through training data set, is then deployed on the test data-set and the results obtained are compared to test the efficiency of the model developed.

II. Classification Process

The Life cycle of generating and deploying the model is categorized as follows:

A. Building/Research the Model

During this phase, the train and test data provided, will be utilized in understanding the problem model. The functional relationship among various variables is studied by plotting each variable vs. other and inspecting the same.

B. Maintaining and Monitoring the Model

This phase primarily comprises of the feature engineering, where new variables are created in order to enhance the model. Depending on the efficiency of the model developed, the appropriate model was deployed with algorithm chosen.

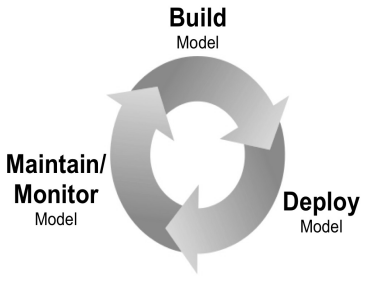

Fig. 1. Model – Life Cycle.

R Language has been used as the platform to perform the comparative study between the two models of classification algorithms. The rich list of libraries available, have been utilized for the implementation of the algorithms and also to generate statistical information, statistical comparisons, ggplot's and histograms to conduct analysis on the given trained data-set.

In this section, the methodology involved in the classification of the Decision Trees and the Random Forests for large and small datasets is presented. The objective of this comparison is creating a benchmark model, which will be useful for various classification scenarios. It will also help in the selection of appropriate model.

C. Defining the problem

The first step is to identify goals. Based on the defined goal, the correct series of tools can be applied to the data to build the corresponding behavioral model.

D. Data selection

At this step, the data relevant to the analysis is decided on and retrieved from the data collection.

E. Data transformation

Also known as data consolidation, it is a phase in which the selected data is transformed into forms appropriate for the mining procedure.

F. Modelling

This method is a subset of Prediction model sector. Based on the data and the desired outcomes, a machine learning algorithm or combination of algorithms is selected for analysis. These algorithms not only include classical techniques such as statistics, neighborhoods and clustering but also next generation techniques such as decision trees, networks and rule based algorithms. The specific algorithm is selected based on the particular objective to be achieved and the quality of the data to be analyzed.

G. Evaluation and Deployment

Based on the results of the data mining algorithms, an analysis is conducted to determine key conclusions from the analysis and create a series of recommendations for consideration.

H. Deploying the Model

This phase involves in researching on the appropriate Machine Learning Algorithm that would suit the newly developed model or the given test and train model. During this phase the Decision Trees and Random Forests algorithms were deployed to find the efficient algorithm that would suit the model.

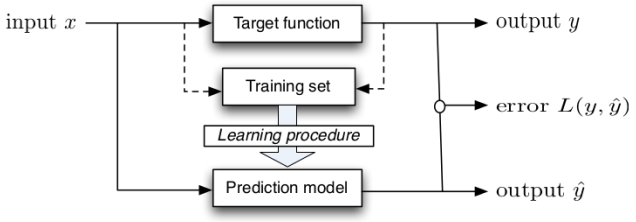

Fig. 2. Model Development

III. CLASSIFICATION METHODS

Machine Learning classification methods have for long been used in the field of data mining, computer vision and many other fields of computer science. As these classification algorithms involve methods from statistics, artificial intelligence and database management they categorize as key elements in data interpretation and data visualization. Also a large part of the information discovered by data mining or computer vision can be learnt from the discriminative model vs generative model distinction and can be applied to new data. In this paper two of the foremost classification methods Decision Trees and Random Forests are being compared.

A. Decision Trees

Decision Tree Algorithms, as the name suggests, are algorithms that can be represented graphically by a tree with many branches and nodes. The Tree construction may be seen as a variable selection method. At each node the issue is to select the variable to divide upon and how to perform the split.

Instances are classified going down from the root to the leaf nodes. The nodes in the tree are tests for specific attributes. The leaves contain the predictions for the response variable. Decision tree algorithms differ in fact from the method of selecting the attribute used to test the instances at each node. Trees can be used for both classification and regression problems. In a classification tree the predicted value is one of the possible levels of the response variable.

The attribute that alone classifies best the data is used at the root, then descending the tree the second best attribute is used to split further the data and so on.

Fig 3, depicts how a Decision Tree is built on a sample data in R. In this scenario 'rpart' library in R was deployed on the height of human beings to build the Tree. And as the discussion above suggests the instance of height was classified going down from root to leaf nodes. Percentage values in the node depicts the number of human beings in each category.

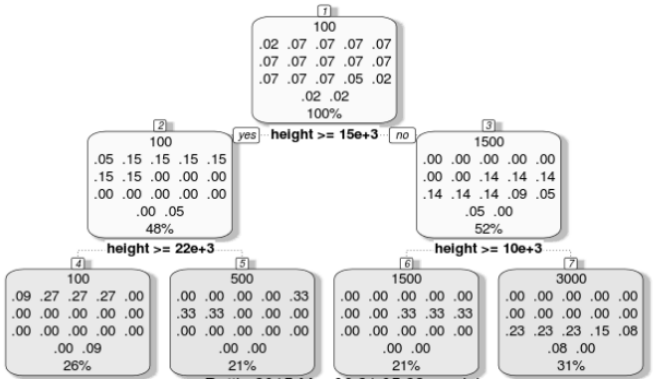

Fig. 3. Decision Tree Sample

The deployment of Decision Trees and the application of this classification algorithm can be observed in various fields. Comparing data statistically, data interpretation, data visualization, text classification etc. are the fields where they are used. On the basis of its type of implementation Decision Tree algorithm can be classified into different categories. In Stock market, it can be used for statistics. Its implementation is wide and can be applied across various sectors raging from companies to schools for maintaining their records and in hospitals it can be used for diagnosis of diseases i.e., brain tumor, Cancer, heart problems, Hepatitis etc.

B. Random Forests

Random Forests lie at the heart of classification. Random Forests is a cluster of Decision Trees i.e., it is an algorithm that consists of many decision trees. The "bagging" approach lies is the core and the random selection of features to build a collection of decision trees with controlled variance is the crux. The idea behind it is to build several trees, to have the instance classified by each tree, and to give a "vote" at each class. The instance's class is to the class with the highest number of votes, the class that occurs the most within the leaf in which the instance is placed. In this algorithm, all trees are grown as follows: [3]

- Firstly, let the number of instances considered for training be A. Sample randomly A instances with replacement and use this sample as the training set for growing one tree.
- Considering B to be the number of input variables. Select a number m smaller than B such that at each node, m variables are selected randomly from the B variables, and select the best split within the m variables. The value of m is held constant during the forest growing.
- Each tree is grown un-pruned to its largest extent possible.

Random Forests are considered general purpose vision tools and are considered as efficient. As the number m of variables that are selected increases, the correlation and strength of the forest increases. So a smaller m results in a

smaller correlation and strength. Apart from traditional classification, Random Forests also provides a wide range of visual cues that are enabled naturally like color, shape, texture and depth. In computer vision community, Random Forests are utilized for research in fields such as class recognition, bi-layer video segmentation, image classification and person identification. Hence Random Forests lie at the core of classification.

IV. Data Definition, Feature Engineering and Result Set

In order to compare the two classification models, datasets from Kaggel's "Forecast use of a city bike share system" problem and "Titanic: Machine Learning From Disaster" problem were considered. The dataset is divided into two parts- the training data and the test data. The algorithm and the related feature engineering is done on the training data, and methodology so obtained is deployed on the test data to obtain the results.

The process for the following two problems is stated as follows:

A. Problem Statement: "Forecast use of a city bikeshare system"

To combine historical usage patterns with weather data in order to forecast bike rental demand.

1) Data Set for "Forecast use of a city bikeshare system": The variables are defined as data fields as shown below.

Data Fields

datetime - hourly date + timestamp
season - 1 = spring, 2 = summer, 3 = fall, 4 = winter
holiday - whether the day is considered a holiday
workingday - whether the day is neither a weekend nor holiday
weather - 1: Clear, Few clouds, Partly cloudy, Partly cloudy
2: Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist
3: Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds
4: Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog
temp - temperature in Celsius
atemp - "feels like" temperature in Celsius
humidity - relative humidity
windspeed - wind speed
casual - number of non-registered user rentals initiated
registered - number of registered user rentals initiated
count - number of total rentals

2) Feature Engineering for "Forecast use of a city bikeshare system":

Initial data provided consists of coded variables such as season, holiday, 'workingday', weather, etc. These coded variables aren't factorized. Hence we have to modify using the factor() function.

The first variable on which feature engineering is performed is the 'datetime' variable. The reason for doing this is because there is more probability of a person renting a bike at 4pm as opposed to 4am.Since the 'datetime' is a string ("2012-02-02 01:00:00"), we use substring() on it to return just the portion of the string that we want. Since this variable is a combination of date and time, this variable has been used to generate the time variable. The time variable has 24 different values hence it is now converted to factor type. The 'datetime' string is also used to generate the month and the year variables.

Using the above 'datetime' variable and applying the same process a new variable with strings like '01/01/2011' is generated. Now these variables are turned into their actual names, such as Monday or Tuesday. This gives rise to a variable that can be factorized into seven levels. To do this, we use a combination of R's weekdays() and Date() functions.

Using the newly created day's column, we can see if there might be any ideas around certain days being more or less popular. This is carried out using the aggregate command and also by creating a plot to visualize some of the trends we found in the 'datetime' variable. The plot has been developed using the R and ggplot2 package. The darker grey in Fig 4. shows "more popular" bike rental times.

From the above we can conclude that Sunday is the day of the week where bikes are less frequently rented. Using this fact we shall create two variables i.e. 'Sunday' variable and the 'weekend' variable. Since the weather in class 4 resembles the weather in class 3, all the variables of class 4 will be converted to class 3. A new variable named 'daypart' has been created. Instead of looking at each hour individually, they are grouped into classes to give rise to various parts of the day (morning, afternoon etc...) as certain parts of the day may lend themselves to more or less frequent bike rentals than just an hour. The 'daypart' is divided into 6 hour blocks.

Finally all the US holidays in the year 2011 and 2012 have been set to '0', as there will be no bike rentals on these particular days.

```
formula = count ~ hour + daypart + atemp + temp + humidity
+ year + month + day + workingday + season + Sunday +
holiday
```

The formula variable has been obtained, to be used as an input parameter for 'rpart'(decision trees)' and 'randomForest'(random forests) functions that are used in the building of a tree, during its learning process. As a part of the learning process to build the trees in both the cases i.e., in case of Decision Trees and Random Forests, the respective libraries 'rpart ' and 'randomForest' are with input parameters which include data, formula, method (anova) and sample size.

3) Results for "Forecast use of a city bikeshare system":

a) Decision Trees: Decision Trees are executed in R using the library 'rpart'.

```
fit <- rpart (count ~ hour + daypart + atemp + temp + humidity
+ year + month + day + workingday + season + Sunday +
holiday, data=train, method="anova")
```

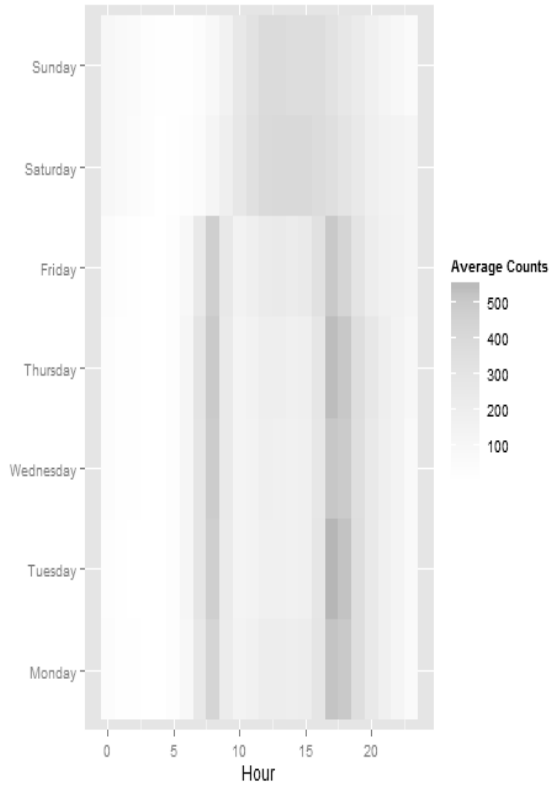

Fig. 4. Featuring Engineering ggplot

b) Random Forests:

For the usage of Random Forests, the entire data should be available. So the missing values have been replaced by the mean of the available values of a given variable.Random Forests are executed in 'R' using the library 'randomForest'.

```
fit <- randomForest(as.factor(count) ~ hour + daypart + atemp
+ temp + humidity + year + month + day + workingday +
season + Sunday + holiday, data=train, importance=TRUE,
ntree=500,sampsize =2)
```

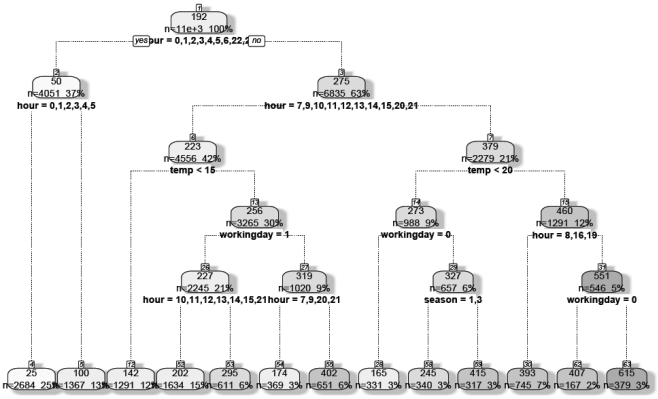

Fig. 5. Decision Tree

The number of trees that are being formed is set to 500 as shown in Fig 5. Sample size sets the number of rows in each of the tree to 2.

B. Problem Statement: Titanic: Machining Learning from Disaster

The analysis of what sorts of people were likely to survive i.e., which passenger is likely to survive.

1) Data Set for "Titanic: Machine Learning from Disaster":

The variables are defined as data fields as shown below.

```
VARIABLE DESCRIPTIONS:
survival        Survival
                (0 = No; 1 = Yes)
pclass          Passenger Class
                (1 = 1st; 2 = 2nd; 3 = 3rd)
name            Name
sex             Sex
age             Age
sibsp           Number of Siblings/Spouses Aboard
parch           Number of Parents/Children Aboard
ticket          Ticket Number
fare            Passenger Fare
cabin           Cabin
embarked        Port of Embarkation
                (C = Cherbourg; Q = Queenstown; S = Southampton)
```

2) Feature Engineering for "Titanic: Machine Learning from Disaster":

A variable 'Survived' is created in the test dataset and it is assumed that all the passengers on board are dead i.e. the value of newly created variable is set to '0'.Using the table () and the prop.table() functions we predict the number of males and females who are alive and dead.

TABLE I. SURVIVAL TABLE

Sex	0 (Dead)	1 (Alive)
Female	0.2579618	0.7420382
Male	0.8110919	0.1889081

From the above Table I we can conclude that the number of females who have survived are larger in number when compared to the number of males who have survived. Therefore we set the 'Survived' variable to 1 when the 'Sex' variable is equal to 'female'.

A new variable 'Child' is created to check if the individual is greater than 18 or less than 18. If the age is less than 18 the value of the 'Child' variable is set to 1, indicating the individual is a child. When the age is greater than 18 the value of the 'Child' variable is set 0, indicating the individual is an adult.

Using the aggregate command, a relation is developed among the 'Child', 'Sex' and the 'Survived' variable as shown in Table II.

TABLE II. AGE SURVIVAL TABLE

S. No.	Child	Sex	Survived
1	0	female	0.7528958
2	1	female	0.6909091
3	0	male	0.1657033
4	1	male	0.3965517

In order to classify on the basis of fares, a new variable 'new_fare' is created and divided into 4 classes. The 'new_fare' is divided into less than $10, between $10 and $20, $20 to $30 and more than $30.

A new variable named 'title' is created to differentiate the individuals on the basis of titles associated with their names. The titles are obtained from the 'Name' variable using the strsplit() and sapply() function. The 'title' column consists of several titles which won't be of much of a use, hence the value in the 'title' column is condensed to two values i.e. 'Sir' and 'Lady'.

Since the values of Number of Siblings/Spouses Aboard and Number of Parents/Children Aboard are given a new variable named 'Familysize' can be obtained by adding the above two variables along with the individual. A family with large 'Familysize' has less probability of survival.

Another possibility arises such that a particular family might have had trouble of surviving. Hence a new variable named 'FamilyIdentity' was created, where the families were identified individually. This variable can be obtained by combining the 'FamilySize' with the 'Surname' obtained from the 'Name' variable. The 'FamilyIdentity' is set to 'Small' for all, where 'Familysize' is less than 2.

Finally the missing values in each variable have been replaced by the mean of all the values for a given variable.

3) Results for "Titanic: Machine Learning from Disaster":

a) Decision Trees: Decision Trees are executed in R using the library 'rpart'.

```
fit <- rpart(Survived ~ Pclass + Sex + Age + SibSp + Parch +
Fare + Embarked, data=train, method="class")
```

b) Random Forests:

The accuracy table tests to see how worse the model performs without each variable, so a high decrease in accuracy would be expected for very predictive variables.

For the usage of Random Forests, the entire data should be available so the missing values have been replaced by the mean of the available values of a given variable.

Random Forests are executed in 'R' using the library 'randomForest'.

```
Fit ← randomForest(as.factor(Survived) ~ Pclass + Sex + Age
+ SibSp + Parch + Fare + Embarked + title + Familysize +
FamilyIdentity, data=train, importance=TRUE, ntree=2000)
```

The number of trees that are being formed is set to 2000.

V. CONCLUSION

The classifications results of Decision Trees and Random Forests for the two problems are compared. Datasets of varying sizes were considered for both the instances. "Forecast use of a city bike share system" has been considered for a large data-set and "Titanic: Machine Learning from Disaster" has been taken as a small data-set.

Parallels have been drawn for this model to be taught of as discriminative model vs generative model distinction. Various instances were taken into consideration to avoid the problem of over fitting. The 10-fold cross validation was used, where 8/10 was taken into consideration for training the algorithm and remaining for testing purposes. This process was repeated 10 times, so as to avoid the problem of over fitting. Parameters such as minimum node size, and maximum standard deviation of samples at a node were used to restrict tree size as the Classification trees generated, had hundreds of nodes and these were to be reduced by pruning to simplify the tree.

The results were aimed at finding the efficiency of the classification algorithms Decision Trees and Random Forests on data-sets of varying sizes. On validating the performance of these classification algorithms it was observed that the performance of Decision Trees was exponentially better than Random Forests when small data-sets i.e. less number of instances were considered. On the other hand Random Forests proved to give better results when compared to Decision Trees for same number of attributes and large data-sets i.e. with greater number of instances was taken into consideration.

As "Forecast use of a city bike share system" data-set, has been taken as the larger data-set, Random Forests was executed on the data-set in 'R' using the 'random forest' library. The results depicted that when the number of instances or the size of the dataset was increased from 4293 to 6494, the correctly classified instances increased from 59.23% to 91.27% for Random Forests. A significant change that was identified was the need for pruning had comparatively decreased. Also in the case of Decision Trees, the ability to predict beyond the minimum and maximum limits of the response variable in the training data was not possible, while in the case of Random Forests, they were not very sensitive to outliers in training data and, accuracy and variable importance were generated automatically.

TABLE III. DECISION TREES VS RANDOM FORESTS RESULTS

Algorithm	Titanic Problem	Bike Sharing Problem
Decision Trees	0.844969	0.778124
Random Forests	0.813430	0.8783412

The values in percentage depict the correctly classified instances on the two problem statements used to compare the classification algorithms as shown in Table III. In the case of the Titanic problem (smaller data-set) Decision Trees successfully produced 0.84% correctly classified instances, while Random Forests produced 0.81% correctly classified instances. In the case of Bike Sharing Problem (larger data-set) Decision Tress produced 0.77% correctly classified instances, while Random Forests produced 0.87% correctly classified instances. As projected by the data from the table, Random Forests gives better results in case of large data-sets(as in the case of the bike sharing problem),decision trees give better results in the case of smaller datasets(as in the case of the Titanic problem).

Thereby, it can be concluded that in the cases of large number of instances Random Forests achieves increased classification performance and yields results that are accurate and precise. This is implied in all scenarios such as covering the missing values problem in the datasets and thus besides accuracy, it also overcomes the over-fitting problem generated due to missing values in the datasets.

Thus it can be safely stated that for all classification problems Decision Trees is handy with small data sets i.e. less number of instances and Random Forests gives better results for the same number of attributes and large data sets i.e. with greater number of instances.

REFERENCES

[1] J. R. Quinlan, "C4.5 Programs for Machine Learning," Morgan Kaufmann, 1993.

[2] J. R. Quinlan, "Learning decision tree classiers," ACM Computing Surveys, 28(1), 1996.

[3] Ron Appel, Thomas Fuchs, Piotr Dollar and Pietro Perona, Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28, "Quickly Boosting Decision Trees - Pruning Underachieving Features Early"

[4] B V Chowdary, Annapurna Gummadi, UNPG Raju and B Anuradha Ravindra Changala, "Decision Tree Induction Approach for Data Classification Using Peano Count Trees," International Journal of Advanced Research in Computer Science and Software Engineering, Volume 2, Issue 4, April 2012, pp 475-479

[5] Yael Ben-Haim Elad Tom-Tov, "A Streaming Parallel Decision Tree Algorithm," Journal of Machine Learning Research 11 (2010) 849-872