DOI: 10.21917/ijsc.2012.0042

RANDOM FOREST BASED MISFIRE DETECTION USING KONONENKO DISCRETISER

S. Babu Devasenapati[1] and K.I. Ramachandran[2]

Department of Mechanical Engineering, Amrita Vishwa Vidyapeetham, Coimbatore, India

E-mail: [1]s_babu@cb.amrita.edu and [2]ki_ram@cb.amrita.edu

Abstract

This paper evaluates the use of random forest (RF) as a tool for misfire detection using statistical features. The engine block vibration contains hidden information about the events occurring inside the engine. Misfire detection was achieved by processing the vibration signals acquired from the engine using a piezoelectric accelerometer. The hidden information regarding misfire was decoded using feature extraction techniques. The effect of Kononenko based discretiser as feature size reduction tool and Correlation-based Feature Selection (CFS) based feature subset selection is analysed for performance improvement in the RF model. The random forest based model is found to have a consistent high classification accuracy of around 90% when designed as a multi class ,ode and reaches 100% when the conditions are clubbed to simulate a two-class mode . From the results obtained the authors conclude that the combination of statistical features and RF algorithm is well suited for detection of misfire in spark ignition engines.

Keywords:

Engine Condition Monitoring, Misfire Detection, Random Forest, Engine Combustion, Recursive Entropy Discretisation, IC Engine

1. INTRODUCTION

Maintenance and condition monitoring of an internal combustion (IC) engine is a very crucial activity required to ensure optimum performance and minimum load on the environment, by restricting emissions to minimum possible levels. Misfire in spark ignition IC engine is a major factor leading to undetected emissions and performance reduction. According to the California Air Resources Board (CARB) regulations [1] engine misfire means, “lack of combustion in the cylinder due to absence of spark, poor fuel metering, poor compression, or any other cause”. Misfire detection in an internal combustion engine is very crucial to maintain optimum performance throughout its service life and to reduce emissions. The engine diagnostic system of the vehicle should be designed to monitor misfire continuously because even with a small number of misfiring cycles, engine performance degrades, hydrocarbon emissions increase, and drivability will suffer [2]. The cylinder misfire cycle also results in a large quantity of unburned fuel being sent through the catalytic converter, which causes a reduction in its service life due to high temperature exposures [3] and also contributes to significant air pollution.

In-cylinder pressure monitoring is very reliable and accurate as individual cylinder instantaneous mean effective pressure could be calculated in real time. However, the cost of fitting each cylinder with a pressure transducer is prohibitively high. Extensive studies have been done using measurement of instantaneous crank angle speed [4] and diverse techniques have been developed to predict misfire [2]. These methods call for a high resolution crank angle encoder and associated infrastructure capable of identifying minor changes in angular velocity due to misfire. The application of these techniques becomes more challenging due to continuously varying operating conditions involving random variation in acceleration coupled with the effect of flywheel, which tries to smoothen out minor variations in angular velocity at higher speeds. Fluctuating torque experienced by the crankshaft through the drive train poses additional hurdles in decoding the misfire signals.

A detailed work reported by [5] using a combination of engine block vibration and wavelet transform to detect engine misfire and knock in a spark ignition engine. The use of engine block vibration is appreciable because it requires minimum instrumentation but the use of wavelet transforms increases the computational requirements. Misfire detection using SVM reported by [6] reports good classification efficiency but the main concern here is the computational complexity of SVM which could pose a serious challenge for implementation in an online model.

The main contribution of this study aims at developing a low cost and computationally frugal system for standalone misfire detection system capable of being integrated in to the engine controller. The system can be reconfigured at very short notice.

The present study proposes a non-intrusive engine block acceleration measurement using a piezoelectric accelerometer connected to a computer through a signal conditioner. The acquired analog vibration signals are converted to digital signals using an analog to digital converter and the discrete data files are stored in the computer for further processing. Feature extraction, feature reduction and feature subset selection techniques are employed and their classification results obtained are presented in the ensuing discussion.

The section 2 describes the experimental setup, the data acquisition methodology using accelerometer and the signal conditioning unit while section 3 describes the experimental procedure in detail. The methods involved in data preprocessing like feature extraction, feature reduction and feature subset extraction are presented in section 4 and the detailed working of the random forest and various stages of work by the algorithm is presented in section 5. The results and discussion are presented in detail under section 6 followed by conclusion in section 7, which establishes that the combination of statistical features and RF algorithm is well suited for detection of misfire in spark ignition engines.

2. EXPERIMENTAL SETUP

The misfire simulator consists of two subsystems namely, IC engine test rig and data acquisition system. They are discussed in the following subsections. The process for building the model is shown in Fig.1.

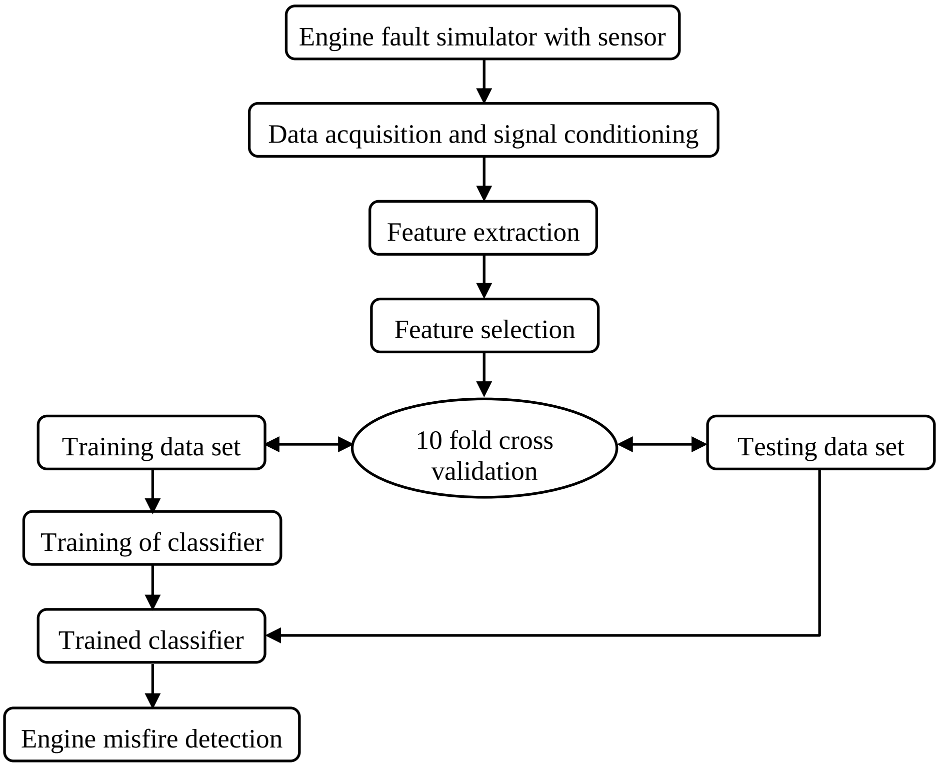

Fig.1. Expert system model building flow chart

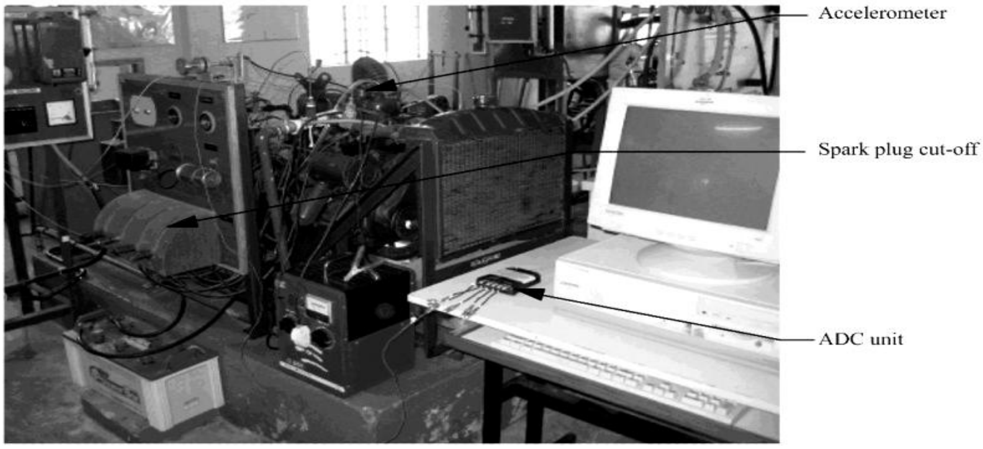

Fig.2. Experimental setup

2.1 IC ENGINE TEST RIG

The experimental setup of the engine misfire simulator consists of a four stroke vertical four cylinder gasoline (petrol) engine. Switching off the high voltage electrical supply to individual spark plugs or to a combination of spark plugs simulates the misfire. The engine accelerator is manually controlled using a screw and nut mechanism that can be locked in any desired position. The engine speed is monitored using an optical interference tachometer.

2.2 DATA ACQUISITION SYSTEM

Accelerometers have a wide operating range enabling them to detect very small and large vibrations. The vibration sensed is a reflection of the internal engine condition. The voltage output of the accelerometers is directly proportional to the vibration. A mono axial piezoelectric accelerometer and its accessories form the core equipment for vibration measurement and recording.

The accelerometer is directly mounted on the center of the engine block using adhesive as shown in Fig.2. The output of the accelerometer is connected to the signal conditioning unit that converts the analogue signal into digital form. The digitized vibration signal (in time domain) is stored in the computer for further processing.

3. EXPERIMENTAL PROCEDURE

The engine is started by electrical cranking at no load and warmed up for 15 minutes. The signal conditioner is switched on, the accelerometer is initialized and the data is recorded after the engine speed stabilizes at 1500 rpm. A sampling frequency of 24 kHz and sample length of 8192 is maintained for all conditions. The highest frequency was found to be 10 kHz. The Nyquist–Shannon sampling theorem recommends that the sampling frequency must be at least twice that of the highest measured frequency or higher, hence the sampling frequency was chosen to be 24 kHz.

Extensive trials were taken at 1500 rpm and discrete vibration signals were stored in the files. Seven cases were considered - normal running (without any fault), engine with any one-cylinder misfire individually (*i.e*. first, second, third or fourth denoted by C1m, C2m, C3m and C4m respectively). All the misfire events were simulated at 1500 rpm, the rated speed of the engine electrical generator set. A sample plot of misfire and no-misfire is presented in Figs.3a and 3b respectively.

4. FEATURE EXTRACTION

Statistical Features: Statistical analysis of vibration signals yields different parameters. The statistical parameters taken for this study are mean, standard error, median, standard deviation, sample variance, kurtosis, skewness, range, minimum, maximum and sum. These features were extracted from the vibration signals. The definitions for these features are commonly available and hence not presented.

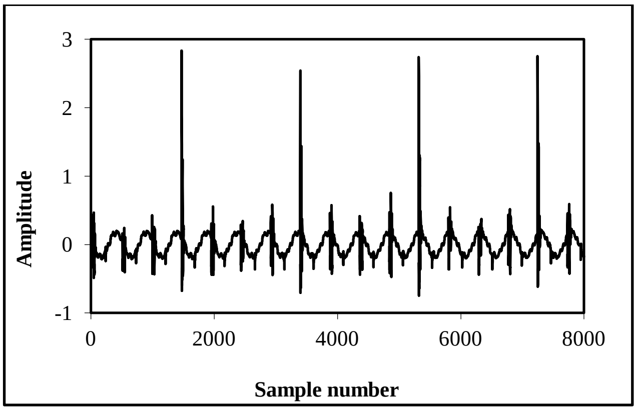

Fig.3a. Amplitude plot-cylinder1 misfire

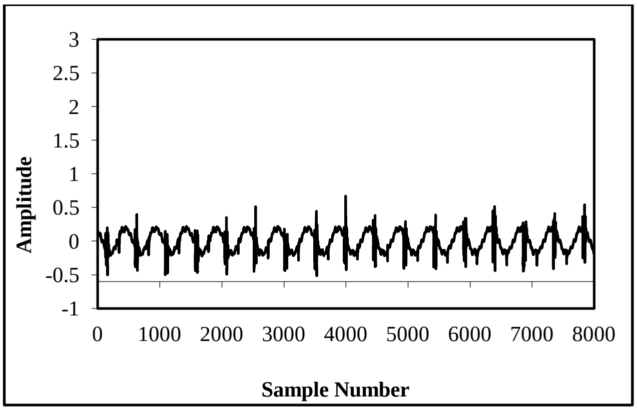

Fig.3b. Amplitude plot- no misfire

4.1 FEATURE REDUCTION

The wealth of information available in the extracted features is abundant and at times overwhelmingly large enough to distract the machine learning system leading to inferior performance. Data granulation as means of feature reduction has many advantages since it reduces the content volume and makes it easy to handle lot of information without challenging the system resources. But the technique to discretise or compress data without loss of valuable information is the key challenge. There are many techniques reported in the literature but an algorithms that can suit the given condition needs to be validated by using the transformed data in the developed model for establishing performance improvements.

The Kononenko's algorithm design [7] uses the Recursive entropy discretisation proposed by Fayyad and Irani with a minor alteration discussed in the next paragraph. To have a complete understanding of the work the Fayyad and Irani method is described as follows.

The Fayyad and Irani model [8] uses a supervised hierarchical split method where multiple ranges are created instead of binary ranges to form a tree. Multi-way splits of the numeric attribute at the same node are performed to produce discrete bins. The number of cut points is determined using the Minimum Description Length (MDL) principle. Here class information entropy is a measure of purity and it measures the amount of information which would be needed to specify to which class an instance belongs [9]. Information entropy minimization heuristic is used to select threshold boundaries by finding a single threshold that minimizes the entropy function over all possible thresholds [10]. This entropy function is then recursively applied to both of the partitions induced. Thresholds are placed half way between the two delimiting instances. At this point the MDL stopping criterion is applied to determine when to stop subdividing discrete intervals, [8]. The Kononenko's algorithm includes an adjustment for discretisation of multiple attributes. It provides a correction for the bias the entropy measure has towards an attribute with many values, [7].

4.2 FEATURE SUBSET SELECTION

Including all the features may improve the classification accuracy but the probability of over fitting the training set data and the additional computational load outweighs their consideration.

It is observed from the computations that there are significant differences in some of the feature values for different types of faults. Selecting those features is crucial for effective classification and doing it manually demands more expertise; however, the effectiveness of the manually selected features is not guaranteed. Selecting the most relevant features through suitable algorithm will yield better classification results. Here feature subset selection (FSS) is performed using Correlation based Feature Selection (CFS). CFS is an algorithm for selecting features that are highly correlated with the class but uncorrelated with each other [11]. CFS has the ability to identify irrelevant, redundant, and noisy features from relevant features as long as their relevance does not strongly depend on other features. This method is adapted for building the model since signal corruption due to noise is more predominant in IC engines. The effect of using CFS on the developed model is studied.

From a list of 11 statistical features presented the CFS has recommended the following features as most prominent ones to be used for model building. They are standard error, standard deviation, sample variance, skewness, range and minimum.

5. CLASSIFIER

New families of ensemble classifiers promoting a team of models that generate many classifiers and aggregate their results have been considered for many classification applications. Two most common methods are boosting [12] and bagging of classification trees. In boosting, successive trees give extra weight to points incorrectly predicted by earlier predictors. In the end, a weighted vote is taken for prediction. In bagging, successive trees do not depend on earlier trees instead each tree is independently constructed using a bootstrap sample of the data set. In the end, a simple majority vote is taken for prediction. Random forests proposed by [13] added an additional layer of randomness to bagging. In addition to constructing each tree using a different bootstrap sample of the data, random forests changed the construction of classification trees. Random forests are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest.

5.1 THE RF ALGORITHM

The random forests algorithm can be represented as follows[13]:

- Initially 'n' subsets of the original data are created, known as tree bootstrap samples
- For each of the bootstrap samples, an un-pruned classification tree is grown. At each node, m tree predictors are randomly sampled and the best split from among those variables is chosen instead of choosing the best split among all predictors. In this method bagging is presented as a special case of random forests obtained when 'm' tree = p, the number of predictors.
- New data is predicted by aggregating the predictions of the n trees (i.e., majority votes considered for classification).
- An estimate of the error rate can be determined using the training data as given by [14]
- At each bootstrap iteration, predict the data not in the bootstrap sample (labeled "out-of-bag", or OOB, data) using the tree grown with the bootstrap sample.
- Aggregate the OOB predictions. (On the average, each data point would be out-of-bag around 36% of the times. An aggregate of these predictions was taken.)
- The error rate is calculated and is termed as the OOB estimate of error rate.

In the next section the algorithm for building each tree is presented in detail.

5.2 BUILDING THE DECISION TREE

In the building phase, the training sample sets with discrete-valued attributes are recursively partitioned until all the records in a partition have the same class. The tree has a single root node for the entire training set. A new node is added to the decision tree for every partition. For a set of samples in a partition S, a test attribute X is selected for further partitioning the set into S1, S2, S3, ……SL. For each new set S1, S2, S3, ……SL new nodes are created and these are added to the decision tree as children of the node for S. Further, the node for S is labeled with test X, and partitions S1, S2, S3, ……SL are recursively partitioned. When all the records in a partition have identical class label, further portioning is stopped, and the leaf corresponding to it is labeled with the corresponding class. The construction of decision tree strongly depends on how a test attribute X is selected. C4.5 algorithm uses information entropy evaluation function as the selection criteria.

The entropy evaluation function is arrived at through the following steps.

Step 1: Calculate Info(S) to identify the class in the training set S.

$$Info(S) = -\sum_{i=1}^{K} \left\{ \left[freq\left(C_i, S/|S|\right) \right] \log_2 \left[freq\left(C_i, S/|S|\right) \right] \right\} \quad (1)$$

a class, I = 1,2,3,….K is the number of classes and freq(Ci, S) is the number of cases included in Ci.

Step 2: Calculate the expected information value, infoX(S) for test X to partition samples in S.

$$InfoX(S) = -\sum_{i=1}^{K} \left[\left(|S_i| / |S| \right) Info(S_i) \right] \quad (2)$$

where, K is the number of outputs for test X, Si is a subset of S corresponding to ith output and is the number of cases of subset Si.

Step 3: Calculate the information gain

$$Gain(X) = Info(S) - Info_X(S) \quad (3)$$

Step 4: Calculate the partition information value Splitinfo(X) acquiring for S, partitioned into L subsets.

$$SplitInfo(X) = -\frac{1}{2}\sum_{i=1}^{L}\left[\frac{|S_i|}{|S|}\log_2\frac{|S_i|}{|S|} + \left(1-\frac{|S_i|}{|S|}\right)\log_2\left(1-\frac{|S_i|}{|S|}\right)\right] \quad (4)$$

Step 5: Calculate the gain ratio

$$GainRatio(X) = Gain(X) - SplitInfo(X) \quad (5)$$

The GainRatio(X) compensates for the weak point of Gain(X), which represents the quantity of information provided by X in the training set. Therefore, an attribute with the highest GainRatio(X) is taken as the root of the decision tree.

It is observed that a training set in the sample space leads to a decision tree, which may be too large to be an accurate model; this is due to over-training or over-fitting. Such a fully-grown decision tree needs to be pruned by removing the less reliable branches to obtain better classification performance over the whole instance space. Pruning is required only if decision tree is used as a standalone classifier built using a single tree. The post-pruning strategy for the decision tree is not used since the random forest algorithm uses the method of voting using multiple tree models for extracting the final classification results. The issue of over fitting does not occur here due to the inherent nature of the random forest [15]. The generalization error of a forest of tree classifiers depends on the strength of the individual trees in the forest and the correlation between them [16].

6. RESULTS AND DISCUSSION

The development of the expert system for misfire detection using recursive entropy discretisation embedded Random forest algorithm is discussed with the implications or effects of the following factors

- All statistical features considered
- Feature reduction using Kononenko's algorithm and
- Features subset selection using CFS

From the experimental setup 200 signals have been acquired for each condition. The conditions are mentioned in section 2.3 and the features were extracted as mentioned in section 3. These features are pre-processed using feature reduction and features subset selection techniques and the effect of these techniques on the model is thoroughly investigated.

6.1 EVALUATION OF CLASSIFIER

Evaluation of the random forest classifier is performed using the standard tenfold cross validation process. The misclassifications details pertaining to Random forest classification without any data pre-processing is presented in the form of a confusion matrix in Table.1. C1m represents misfire in cylinder 1, C2m, C3m and C4m, represents misfire in cylinder 2, 3 and 4 respectively. Good represents no misfire in any cylinder. The diagonal elements shown in the confusion matrix represents the correctly classified points and non-diagonal elements are misclassified ones. Referring to Table.1, it is evident that the misclassification among the faulty conditions and 'good' condition is minimal. However there are misclassifications among the faulty conditions which do not compromise the overall misfire prediction accuracy. For example, consider row C1m in which 184 conditions are correctly identified as misfire in C1 but 20 are wrongly identified as misfire in C3, 9 in C3m and 5 in C4m. However 2 misfire instances are wrongly misclassified as good, which is an undesirable error. We can conclude that, as long as the system does not misclassify good as misfire or vice versa the model is robust enough for real time application.

The performance values depicted in Table.2 clearly portrays the capability of the developed model when subjected to various data preprocessing techniques. From the results obtained it is evident that including all the data gives better performance but there is a risk of performance reduction due to model over fitting the data. In a later date when the engine noise increases due to wear, there are possibilities of the model suffering setbacks due to increased misclassifications. However a judicious decision has to be taken among the available alternatives to freeze the best among the developed models. Both model B and D deliver 100% result in 2 class mode with almost similar multi class performance. Based on the processing time model D is chosen, however model B could also be considered since there is no appreciable deviation in performance.

Table.1. Confusion matrix – Random forest with all features considered

STATE	**Good**	**C1m**	**C2m**	**C3m**	**C4m**
Good	200	0	0	0	0
C1m	2	184	0	9	5
C2m	0	0	200	0	0
C3m	0	13	0	154	33
C4m	0	7	0	33	160

Table.2. Classifier performance evaluation chart

	Without data preprocessing **Model A**	With data discretisation **Model B**	With CFS (data not discretised) **Model C**	With discretised data followed by CFS **Model D**
Random forest performance	89.2	90.1	88.9	89.7
Processing time taken in seconds	3.4	0.6	2.4	0.5
Random forest performance in two-class mode	99	100	99	100

7. CONCLUSION

In a condition monitoring activity fault identification forms the major objective and fault classification comes second in priority. In this context, the present algorithm performs fault identification (differentiating between good and faulty conditions) sufficiently well since it has not misclassified any instances out of 1000 samples supplied. This is calculated by considering good as one class and misfire in all cylinders as the second class. This assumption is logically valid since misfire detection is crucial and the identification of the exact cylinder where misfire happens is not critical.

From the results presented it is encouraging to conclude that Random forest algorithm is well suited for detection of misfire in IC engines. Specifically focusing on the two-class problem result that is presented in the second row of Table.2, in which good Vs misfire in any cylinder is considered, one is able to infer that data preprocessing is absolutely necessary for improving the performance of the expert system and to reduce computational time required to arrive at a decision. The authors conclude that the model D, based on data discretisation followed by CFS is the best since it has the additional advantage of least computational complexity when compared to CFS without data discretisation, evident from the time required to run the model.

It should be noted that these results are specific to this application and cannot be generalized to other similar applications. Further studies are to be conducted on different engines at different operating conditions in order to generalize this finding.

REFERENCES

[1] California Air Resources Board, "Technical status Update and Proposed Revisions to Malfunction and Diagnostic System Requirements Applicable to 1994 and Subsequent California Passenger Cars, Light-Duty Trucks, and Medium-Duty Vehicles – (OBDII)", *CARB staff report*, 1991.

[2] Lee D and Rizzoni G, "Detection of Partial Misfire in IC Engines Using Measurement of Crankshaft Angular Velocity", *SAE Technical paper 951070*, 1995.

[3] Klenk M, Moser W, Mueller W and Wimmer W, "Misfire Detection by Evaluating Crankshaft Speed – A Means to Comply with OBDII", *SAE Technical paper 930399,* 1993.

[4] Francisco V. Tinaut, Andres Melgar, Hannes Laget and Jose I. Dominguez, "Misfire and compression fault detection through the energy model", *Mechanical Systems and Signal Processing*, Vol. 21, No. 3, pp. 1521-1535, 2007.

[5] Jinseok, Kim Manshik, Min Kyoungdoug, "Detection of misfire and knock in spark ignition engines by wavelet transform of engine block vibration signals", *Measurement Science & Technology*, Vol. 13, No. 7, pp. 1108-1114, 2002.

[6] Babu Devasenapati, Ramachandran K I and Sugumaran S, "Misfire Detection in a Spark Ignition Engine using Support Vector Machines", *International Journal of Computer Applications*, Vol. 5, No. 6, pp. 25-29, 2010.

[7] Kononenko I and Se June Hong, "Attribute selection for modelling", *Future Generation Computer Systems*, Vol. 13, No. 2-3, pp. 181-195, 1997.

[8] Fayyad U.M and Irani K.B, "Multi-interval discretization of continuous valued attributes for classification learning", *Proceedings of the International Joint Conference on Uncertainity in Artificial Intelligence*, Vol. 2, No. 1, pp. 1022-1027, 1993.

[9] James Dougherty, Ron Kohavi and Mehran Sahami, "Supervised and unsupervised discretization of continuous features", *Proceedings of Twelfth International Conference on Machine Learning*, Vol. 95, No. 10, pp. 194-202, 1995.

[10] Michael K. Ismail and Vic Ciesielsk, "An Empirical Investigation of the Impact of Discretization on Common Data Distributions", *Design and Application of Hybrid Intelligent Systems*, *Third International Conference on Hybrid Intelligent Systems*, Vol. 105, pp. 692-701, 2003.

[11] Mark A Hall, "Correlation-based Feature Selection for Discrete and Numeric Class Machine Learning", *Proceedings of the Seventeenth International Conference on Machine Learning*, Vol. 1, pp. 359 - 366, 2000.

[12] Cortes H and Drucker C, "Boosting decision trees", *Advances in Neural Information Processing Systems*, Vol. 8, pp. 479-485, 1996.

[13] Breiman, Leo, "Random Forests", *Machine Learning*, Vol. 45, pp. 5-32, 2001.

[14] Bylander T, "Estimating Generalization Error in Two-Class Datasets Using Out-of-Bag Estimates", *Machine Learning,* Vol. 48, pp. 287-297, 2002.

[15] Liaw Andy and Wiener Matthew, "Classification and Regression by random Forest", *The Newsletter of the R Project,* Vol. 2, No. 3, pp. 18-22, 2002.