IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 23, Issue 2, Ser. II (Mar. – Apr. 2021), PP 51-62
www.iosrjournals.org

Decision Trees for Nuclear Power Plant Performance Optimization

S.Narasimhan[1*], Dr.V.Rajendran[2]

[1]*VELS Institute of Science, Technology and Advanced Studies (VISTAS) Chennai, India*

[1]*Bharatiya Nabhikiya Vidyut Nigam Limited BHAVINI Kalpakkam, India*

[2]*Department of ECE VELS Institute of Science, Technology and Advanced Studies (VISTAS) Chennai, India*

Abstract. *This paper deals with application of decision tree algorithms based on data mining methodologies for optimizing the representative process of Nuclear Power Plant by estimating the steady state parametric values of key performance indicator variables. With this the operator will be in apposition to estimate the optimum values of the critical parameters, tune the process to the estimated values so that the process efficiency can be improved with the available resources. He will also be able to forecast process upset conditions and plan for maintenance and surveillance schedules with these predictions. Expert systems using data mining is a promising area of process optimization. It plays an important role in improving the plant capacity factor, increases the availability and reduces the down times. Operator Assistance by systems based on data mining techniques will act as a cognitive aid to assess the process conditions, take quick decisions and corrective actions.*

In this paper, we introduce classification models using data mining methods to improve the performance of a typical nuclear reactor cooling pump lubrication system process. In this study, we have built the models based on Decision Tree Algorithm. Various models of decision tree algorithms along with their boosting and bagging methods are developed using a training data set and a holdout test data set derived from the plant data for a month of steady operation. On evaluation of the developed models with calculated metrics, we found that the classification models based on decision trees are in fact useful for fair prediction of key parameters in a multiclass environment with class imbalance.

Keywords: *Adaboost, Area Under the Curve, Bagging Tree,C4.5, Conditional Inference Trees, Confusion Matrix, Data modelling, Data mining, Data Transformation, Decision Trees, Random forest, Resource Operating Characteristics Curves*

Date of Submission: 20-03-2021 Date of Acceptance: 04-04-2021

I. Introduction

Nuclear Energy contributes to about 3.2 percent of the total power generated in India. Nuclear power plants involves high capital cost, long gestation period and are highly regulated based on various safety, environmental and societal regulations. All the Nuclear Power Plants in our country are operated as base load stations requiring a near constant load factors. If it shuts down due to any grid disturbances, process upset situations or malfunction of equipment, after correcting the fault, the restart time is typically 36 hours due to constraints from nuclear physics and thermodynamic considerations. As seen from the recent Fukushima Nuclear Accident due to Tsunami in Japan, the impact of any beyond design basis incidents will have larger implications in the society and the reputation of the nation altogether. Further the restoration efforts are high resource intensive, time consuming and require a very huge cost. Yet the contribution of Nuclear Energy in a power starving country like us cannot be dispensed with. Hence, to be competitive in the power generation market, the Nuclear Plant Process systems are designed, engineered and operated with high safety margin and high thermodynamic efficiency. The operators are required to be extensively trained and licensed by the regulator to understand the process stability, act swiftly during a process upset condition, monitor and understand the behavior of the process at all states of the reactor and take quick control or safety action as warranted using their knowledge, skill and experience.

Hence providing him a computerized expert systems will enhance the capability of the decision making process by the operator. The Expert System can acquire the data from the plant wide distributed control systems, have machine language algorithms to understand the process dynamics and develop data mining models to provide predictions/recommendations on the process to the operator in a cognitive mode. The system can be fairly accurate to be dependent upon and can predict under all operating regimes of the process.

We propose in this study the performance evaluation of classification models based on Decision Tree

Injunction Techniques on a critical process system of oil lubrication for reactor coolant pump. In this paper, we apply classification models based on decision tree algorithms on the plant process data to make predictions on the identified key performance variables. Different decision tree models are developed and evaluated for their performance. Further use of bagging and boosting models based on Decision Trees are also developed and evaluated to identify the suitability and best fit so that we can deploy similar models for other safety systems after due review and clearance. The models are developed for the key parameters of the process taken for this study.

II. Literature Review

Flynn, Ritchie, and Cregan[1] demonstrated the power plant monitoring and control using data oriented models and algorithms utilizing the Distributed Control System(DCS) data. During the process upset conditions the DCS would be flooded with many alarm signals resulting in critical parameters of the process losing the attention of the plant operator. Further, many of the process parameters are correlated and show similar trends and behavior. Hence, the author suggests development of an expert system using data mining algorithms that will establish the relationships among the parameters under steady state condition. The algorithm will also help to identify abnormal conditions from the original trained observations.

Li *et al.*[2] Proposes similar power plant process optimization using data mining techniques using the Plant data Acquisition systems using fuzzy association rule mining.

Ogilvie, Swidenbank, and Hogg [3] also used association rule mining on the data from the historian servers to analyze the behavior of steam generator system for a thermal power plant.

In our earlier paper [4] we used data analytics for identifying the drift in triplicated sensors for monitoring a safety parameter in a nuclear power plant. We had proposed a single online soft parameter derived from the data model. Further in another paper by us [5], we had applied principal component analysis and k-means clustering to identify stable operating regimes of a process in a nuclear power plant. We had also applied association rule mining techniques and multi collinearity analysis to identify Key Performance Indicators (KPI) and their optimum values for a stable process.

In the paper[6] we have specifically applied classification algorithms based on association rules on the key performance parameters identified for the stable process of the nuclear plant. The estimation models were built for the identified KPIs and they were evaluated for their performance using confusion matrix.

Davidov, Tsur, and Rappoport [7] have demonstrated the data mining algorithms Support Vector Machines(SVM) and Naïve Bayes to classify data for the web analysis project. The result gave us a confidence that SVM and Naïve Bayes can be considered models for further research on expanding their domains of applications. The models are trained on the steady state condition of the process under class imbalance conditions and were evaluated with a test data set. The results are under publication.

Ahmad *et al*. [8] discussed the high variability of renewable generation necessitating forecasting for optimal balancing and dispatch of generation in a smarter grid against the challenge to improve the accuracy . The paper investigated the accuracy, stability and computational cost of Random Forest and Extra Trees for predicting hourly PV generation output, and compared their performance with support vector regression (SVR).

Gaikwad, D. P. *et al.*[9] developed Intrusion Detection System for the computer network security using Bagging Ensemble method. The selections of relevant features are required to improve the accuracy of the classifier. The proposed intrusion detection system is evaluated in terms of classification accuracy, true positives, false positive and model building time. It was observed that proposed system achieved the highest classification accuracy of 99.7166% using cross validation. It exhibits higher classification accuracy than all classifiers except C4.5 classifier on test data set.

Relan *et al*.[10] also applied decision trees for intrusion detection technology for network security evaluating the performance in terms of its effectiveness and the number of feature to be examined. They have proposed two techniques, C4.5 Decision tree algorithm and C4.5 Decision tree with Pruning, using feature selection. The Experimental Result shows that, C4.5 decision tree with pruning approach is giving better results with all most 98% of accuracy.

The above studies have motivated us to deploy decision trees and their ensemble models for nuclear power plant process parameter estimation. Various models were developed and evaluated for their performance to assess their suitability for better performance and expert system development.

The rest of the paper is organized as follows:

In section 3, a brief description of Decision Tree, Bagging Tree, Adaboost and Random Forest algorithms are presented. In section-4, the data set is introduced. In section 5, we had described the data preparation and model building. In section-6, models are evaluated based on the metrics. In section-7, we conclude and highlight the possible future work.

III. Relevant Data Mining Algorithms

3.1 Decision Tree Induction

A decision tree is a flowchart-like tree structure, with roots, nodes, leaves and branches. A tree starts with a root node. Every internal non-leaf node tests an attribute and every terminal leaf node holds a class label. The outcome of a test branches to a leaf or a non-leaf node.

Given a unknown tuple, X, a path is traced from the root to a leaf node, which holds the class prediction for that tuple. Decision trees can easily be converted to classification rules.

The selection of an attribute for splitting and branching a given data partition, D, of class-labelled training tuples into individual classes is done either heuristically or based on Information Gain, Gain Ratio and Gini Index. Using these scores, the attribute/feature which yields maximum information gain is chosen as the splitting attribute for the given tuples.

Let node N hold the tuples of partition$D$. The splitting attribute for node N will be the one that minimizes the information needed to classify the tuples in the resulting partitions. The attribute will be the one with highest information gain and results in lowest number of tests to classify a given tuple and a simple tree.
The expected information needed to classify a tuple in D is given by[11]

$$Info(D) = -\sum_{i=1}^{m} p_i log_2(p_i) \qquad (1)$$

Where p_i is the nonzero probability that an arbitrary tuple in D belongs to class C_i and is estimated by$|C_{i,D}|/|D|$. $Info(D)$ is the average amount of information needed to identify the class label of a tuple in D. $Info(D)$ is also known as the entropy of$D$.
Now, suppose we were to partition the tuples in D on some attribute A having v distinct values, $\{a_1, a_2, \dots a_v\}$, as observed from the training data. Attribute A can be used to split D into v partitions or subsets, $\{D_1, D_2, \dots D_v\}$, where D_j contains those tuples in D that have outcome a_j of A. These partitions would correspond to the branches grown from node N. However, the partitions may contain tuples from different classes rather than from a single class. Then the amount of information needed to arrive at an exact classification is measured by[11]

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j) \qquad (2)$$

The term $\frac{|D_j|}{|D|}$ acts as the weight of the j th partition. $Info_A(D)$ is the expected information required to classify a tuple from D based on the partitioning by$A$. Information gain is defined as the difference between the original information requirement (i.e., based on just the proportion of classes) and the new requirement (i.e., obtained after partitioning on A.[11]

$$Gain(A) = Info(D) - Info_A(D) \qquad (3)$$

The attribute A with the highest information gain, $Gain(A)$, is chosen as the splitting attribute at node N, so that the amount of information still required to finish classifying the tuples is minimal (i.e., minimum $Info_A(D)$).

Decision tree classifiers are best suited for exploratory research as it does not require any domain knowledge or parameter setting. Decision trees can handle multidimensional data. The representation of the information in tree form is easily understandable. The learning and classification steps of decision tree induction are simple and fast. In general, decision tree classifiers have good accuracy. However, successful use may depend on the data at hand.
A decision tree classifier is not sensitive to mislabeled data, handles irrelevant features, and is computationally efficient in training and prediction. It performs best when the training set is an accurate representation of the population [9].However it tends to over fit to the training data, resulting in a drop in accuracy and performance on the test data. That is, it does not generalize well. For a binary classification problem with a large, mixed (numerical and categorical) feature set, the decision tree is an apt classifier. It can handle irrelevant features and is scalable.
C4.5 uses an extension to information gain known as gain ratio. It applies a kind of normalization to information gain using a "split information" value defined analogously with $Info(D)$ as [11]

$$Split\ Info_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times log_2\left(\frac{|D_j|}{|D|}\right) \qquad (4)$$

This value represents the potential information generated by splitting the training data set D, into v partitions, corresponding to the v outcomes of a test on attribute A. The gain ratio is defined as[11]

$$Gain\ Ratio(A) = \frac{Gain(A)}{Split\ Info_A(D)} \quad (5)$$

The attribute with the maximum gain ratio is selected as the splitting attribute.
The Gini index is used in CART. The Gini index measures the impurity of D, a set of training tuples, as[11]

$$Gini(D) = 1 - \sum_{i-1}^{m} p_i^2 \quad (6)$$

where p_i is the probability that a tuple in D belongs to class C_i and is estimated by$\left|C_{i,D}\right|/|D|$. The sum is computed over m classes.

When a decision tree is built, the tree is pruned to avoid over fitting the data due to noise or outliers. Pruned trees tend to be smaller and less complex and, thus, easier to comprehend. They are usually faster and better at correctly classifying independent test data (i.e., of previously unseen tuples) than unpruned trees.

Two decision tree algorithms in R package Conditional Inference Tree (ctree) and J48 pruned Tree (C4.5) have been demonstrated in this study.

Conditional inference trees embed tree-structured regression models into a well-defined theory of conditional inference procedures. This non-parametric class of regression trees is applicable to all kinds of regression problems, including nominal, ordinal, numeric, censored as well as multivariate response variables and arbitrary measurement scales of the covariates. The methods are described in Hothorn et al. [12], Zeileis et al. [13] and Strobl et al. [14].

J48 generates unpruned or pruned C4.5 decision trees [15]. C4.5 uses a method called pessimistic pruning, which uses error rate estimates to make decisions regarding sub tree pruning. However, it does not require the use of a prune set. Instead, it uses the training set to estimate error rates. It adjusts the error rates obtained from the training set by adding a penalty, so as to counter the bias incurred.

3.2 Ensemble Methods in Decision Tree Induction

3.2.1 Bagging Trees

A bagging tree combines a series of k learned models (or base classifiers), with the aim of creating an improved composite classification model,M_*. A given data set,D is used to create k training sets,$D_1, D_2, \ldots D_k$, to generate k learned models (or base classifiers), $M_1, M_2, \ldots M_k$. Given a new data tuple to classify, the base classifiers each vote by returning a class prediction. The ensemble returns a class prediction based on the majority votes of the base classifiers. An ensemble tends to be more accurate than its base classifiers. The base classifiers may make mistakes, but the ensemble will misclassify X only if over half of the base classifiers are in error.
The pseudo algorithm for bagging tree is as shown in fig.1[11]

Algorithm: Bagging. The bagging algorithm—create an ensemble of classification models for a learning scheme where each model gives an equally weighted prediction.

Input:

- D, a set of d training tuples;
- k, the number of models in the ensemble;
- a decision tree learning scheme.

Output: The ensemble—a composite model, M_*.

Method:

(1) **for** i = 1 to k **do** // create k models:
(2) create bootstrap sample, D_i , by sampling D with replacement;
(3) use D_i and the learning scheme to derive a model, M_i ;
(4) **end for**

To use the ensemble to classify a tuple, X:
let each of the k models classify X and return the majority vote;

Figure.1. Pseudo Algorithm for Bagging[11]

We have used the "ipred" package in R to demonstrate the bagging tree models. Ipred provides improved predictive models by indirect classification and bagging for classification, regression and survival problems as well as resampling based estimators of prediction error. Bagging for classification and regression trees were suggested by Leo Breiman [16] in order to stabilize trees. The trees in this function are computed using the implementation in the rpart package in R.

3.2.2 AdaBoost

Unlike bagging trees, in boosting models, weights are also assigned to each training tuple. A series of k classifiers is iteratively learned. After a classifier, M_i , is learned, the weights are updated to allow the subsequent classifier,M_{i+1}, to "pay more attention" to the training tuples that were misclassified by M_i . The final boosted classifier,M_*, combines the votes of each individual classifier, where the weight of each classifier's vote is a function of its accuracy.

AdaBoost (short for Adaptive Boosting) is a popular boosting algorithm. The pseudo algorithm for AdaBoost tree is as shown in figure.2.[11]

Algorithm: AdaBoost. Create an ensemble of classifiers. Each one gives a weighted vote.

Input:

- D, a set of d training tuples;
- k, the number of models in the ensemble;
- a decision tree learning scheme.

Output: A composite model.

Method:

(1) initialize the weight of each tuple in D to $1/d$;
(2) **for** $i = 1\ to\ k$ **do** // for each round:
(3) sample D with replacement according to the tuple weights to obtain D_i ;
(4) use training set D_i to derive a model, M_i ;
(5) Compute $error(M_i) = \sum_{j=1}^{d} w_j \times err(X_j)$,where $err(X_j)$ is the misclassification of tupleX_j. If the tuple was misclassified, then $err(X_j)$ is 1; otherwise, it is 0.
(6) **if** $error(M_i) > 0.5$ **then**
(7) go back to step 3 and try again;
(8) **end if**
(9) **for** each tuple in D_i that was correctly classified **do**
(10) multiply the weight of the tuple by $error(M_i)/(1 - error(M_i))$; // update weights
(11) normalize the weight of each tuple;
(12) **end for**

To use the ensemble to classify tuple, X:

(1) initialize weight of each class to 0;
(2) **for** $i = 1\ to\ k$ **do** // for each classifier:
(3) $w_i = log\frac{(1-error(M_i))}{error(M_i)}$; // weight of the classifier's vote
(4) $c = M_i(X)$; // get class prediction for X from M_i
(5) add w_i to weight for class c
(6) **end for**
(7) return the class with the largest weight;

Figure.2. Pseudo Algorithm for Boosting[11]

AdaBoost algorithm is demonstrated using the R package adabag in this study.
It implements Freund and Schapire's Adaboost.M1 algorithm [17] using classification trees as individual classifiers. Once these classifiers have been trained, they can be used to predict on new data. Also, cross validation estimation of the error can be done. The weight updating coefficient is set as 'Breiman' where

$$w_i = 0.5log\frac{1 - error(M_i)}{error(M_i)} \qquad (7)$$

3.2.3 Random Forests

Random Forests are an improvement over bagged decision trees. In Bagging, the decision trees can have a lot of structural similarities and in turn have high correlation in their predictions. Ensembles works better if the predictions from the sub-models are uncorrelated or at best weakly correlated. The individual decision trees are generated using a random selection of attributes at each node to determine the split. In Random Forest, the individual decision trees are generated using a random selection of attributes at each node to determine the split. The resulting predictions from all of the sub trees have less correlation. Each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest. Each tree votes and the most popular class is returned.

Random forests are comparable in accuracy to Boosting, yet are more robust to errors and outliers. They are faster than either bagging or boosting.

The random forest algorithm is demonstrated using Breiman and Cutler's Random Forests for Classification and Regression, Version 4.6-14 [18]in the R package.

IV. The Data Sets for the study

The subject Nuclear power plant has a Distributed digital Control System (DCS) that scans all the process sensors, collects the data and logs in the historian. We collected the archived data from the historian for the key parameters of the process of the oil lubrication for reactor coolant pump for one month of stable operating performance The pump circulates the coolant to take away the thermal energy from the fission reaction in the core to produce steam in steam generators. The oil cooling system will circulate the oil in the mechanical seal assemblies of the pump to provide cooling, lubrication as well as sealing for the pump shaft assembly.

The data was collected from the server for the analysis at a sampling rate of 5 seconds as listed in Table.1. After preprocessing of the data and removing the outliers the total data tuples taken for the study for the above period is about two lakhs.

Table.1.Process Parameters and Ranges of Measurements

Sl.No	Parameter ID	Parameter Description	Range of Measurement
1	Temperature-A	Oil inlet temperature to bottom mechanical seal	0 to 600^o C
2	Temperature-B	Return oil temperature after cooling by blowers	0 to 600^o C
3	Pressure-A	Oil inlet pressure to bottom mechanical seal	0-7 kg/cm^2
4	Flow-A	Oil outlet flow from mechanical seal	0-50 m^3/hr

V. Model Building

The historian data in the process computers of DCS for the selected variables are imported to data analysis tool 'R' version 3.6.3. We converted the time series data to categorical attributes by slicing the ranges to 25 bins. The data frame now consists of 20585 data tuples of four categorical variables with 25 classes. Further, we have partitioned the data frame to training set and a test set in the ratio 70:30. Since the data is pertaining to steady state condition of the process, dataset indicated a class imbalance in each of the attributes.

The models were implemented R language version 3.6.3 using the training data set. The various settings are as listed in Table.2.

Table.2.Model Configurations in "R"

Model	Configuration Parameters
Conditional Inference Tree	Library- party,xtrafo = ptrafo, ytrafo = ptrafo, subset = NULL, weights = NULL
C4.5	Library –Rweka 04-43, J48 pruned tree
Bagging Tree	Library -ipred, bootstrap replications 25, keepX=TRUE.
AdaBoost	Library-adabag, boos =TRUE, mfinal = 100, coeflearn = 'Breiman',method AdaBoost.M1"
Random Forest	Library-randomForest 4.6-14, Type of random forest: classification, Number of trees: 500, No. of variables tried at each split: 2

We used the predictions made by the models on test data set to evaluate their performance for each of the variable.

VI. Model Evaluation

The performance of the models were evaluated using the confusion matrix in R package "caret"[19]. The performance indicators as derived from the confusion matrix that are used for model evaluation and selection are as per the Table.3.

Table.3.Model Metrics for Evaluation

Sl.No	Model Metrics	Description	Derivation
1	Global Accuracy/Recognition rate	percentage of test data tuples that are correctly classified	$\frac{TP + TN}{P + N} \times 100$
2	No Information Rate/Prevalence	largest class percentage in the data	$\frac{TP + FN}{P + N}$
2	F1-Score	Harmonic average of precision and recall	$\frac{2 \times Precision(\frac{TP}{TP + FP}) \times Recall(\frac{TP}{TP + FN})}{Precision + Recall}$
3	Model Fitness(SSPN)	Fitness score on model prediction accuracy	$Sensitivity(\frac{TP}{TP + FN}) \times Specificity(\frac{TN}{TN + FP}) \times Positive\ Prediction\ value(\frac{TP}{TP + FP}) \times Negative\ Prediction\ value(\frac{TN}{TN + FN}) \times 1000$
4	Area Under the Curve (AUC) Value	The area covered in a plot with False Positive Rate (1-Specificity) on the X-axis and True Positive Rate (Sensitivity) on the Y-Axis. The diagonal line represents a random guessing with equal probability of True and False Positive Rates. The accuracy of the model is based on how close is the curve to the diagonal line and the area covered.	

TP- True Positives, FP-False Positives, TN-True Negatives, FN-False Negatives, P-Total Positives, N-Total Negatives

6.1 Global Accuracy

Global Accuracy indicates the overall recognition performance of the model for the true positives and true negatives out of the test set. The Global Accuracy values are calculated as per the Table.3 for all models in each parameters and represented in the Figure.4.

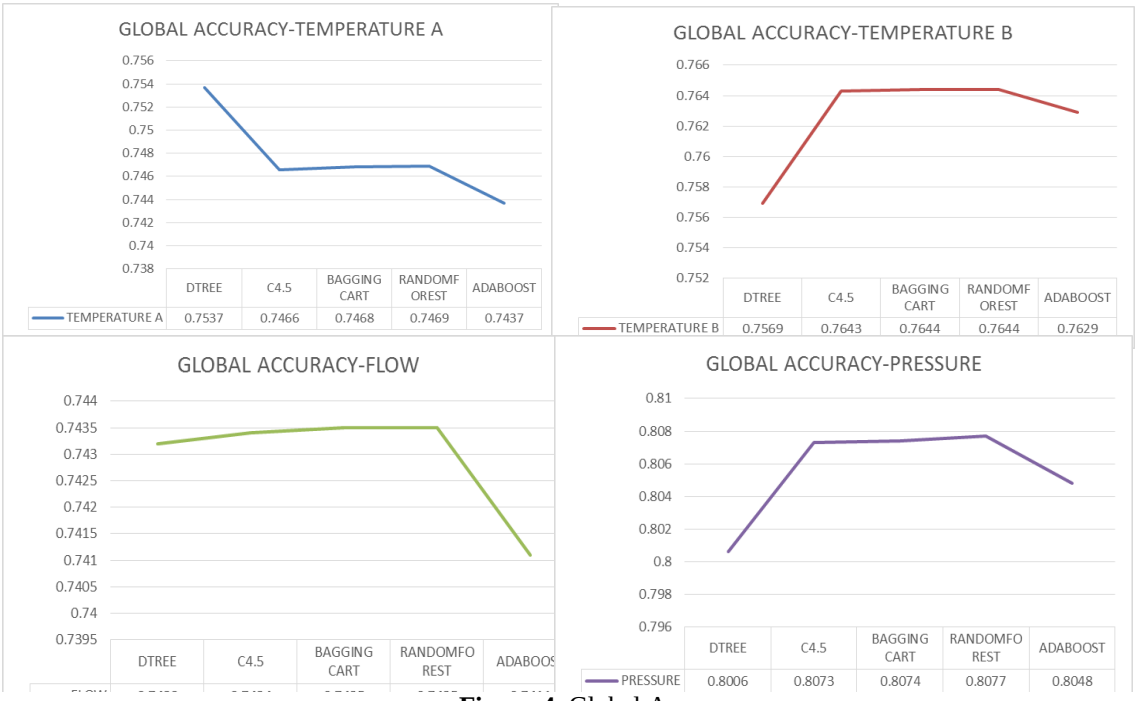

Figure.4. Global Accuracy

In this study, it is found that all the models exhibit more close values ranging from 0.7432(74.32%) to 0.8077(80.77%), lowest being Conditional Inference Tree for Flow and highest being Random Forest for Pressure. The overall accuracy values for the models for the parameter Pressure is highest compared to other variables.

For all the parameters, the models C4.5, Bagging Cart and random Forest Models exhibit a near equal and consistent performance marginally higher than other models. Further Ada Boost model exhibits a marginally lower performance than the other three models for all the cases.

6.2 No Information Rate (NIR)

The metric No Information Rate indicates how common the desired class in the prediction is. It is just the largest class percentage in the data. The idea is that a useful model should do better than you could do by always predicting the most common class. Higher rate indicates that the model has encountered a higher class imbalance. Accuracy of the models should be higher than No information rate to consider the prediction by the model as significant. The NIR values are calculated as per the Table.3. for all models in each parameters and represented in the Figure.5.

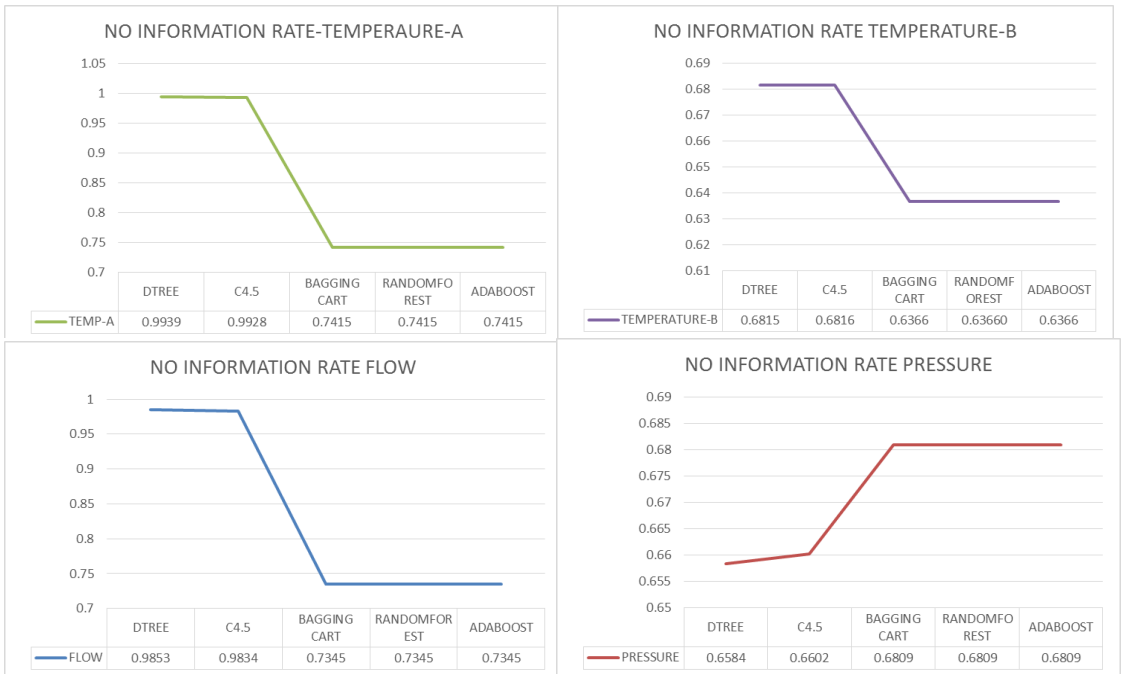

Figure.5. No Information Rate (NIR)

As seen by the plot, the NIR values are nearly 1 for Temperature-A and Flow for the models Conditional Inference Trees and C4.5. Whereas their global accuracies are in the range 0.74 to 0.75. Hence dependability of the models for these parameters are lower. However for these parameters the ensemble models exhibit a NIR value comparable to Accuracy and hence can be considered. For all the other cases the NIR values are lesser than Accuracy values.

6.3 F1 Score

Generally accuracy indicates if a model is trained correctly and how it performs in a homogeneous set of classes. However, it is not a complete metric for evaluation, when there is class imbalance. Even though precision and recall are individually calculated for the data set of the given class, a single overall evaluation is provided by the F1 score. An F1 score is considered perfect when it's 1, while the model is a total failure when it's 0. The F1 scores are calculated as per the Table.3 for the most prevalent class in each parameters and represented in the Figure.6.

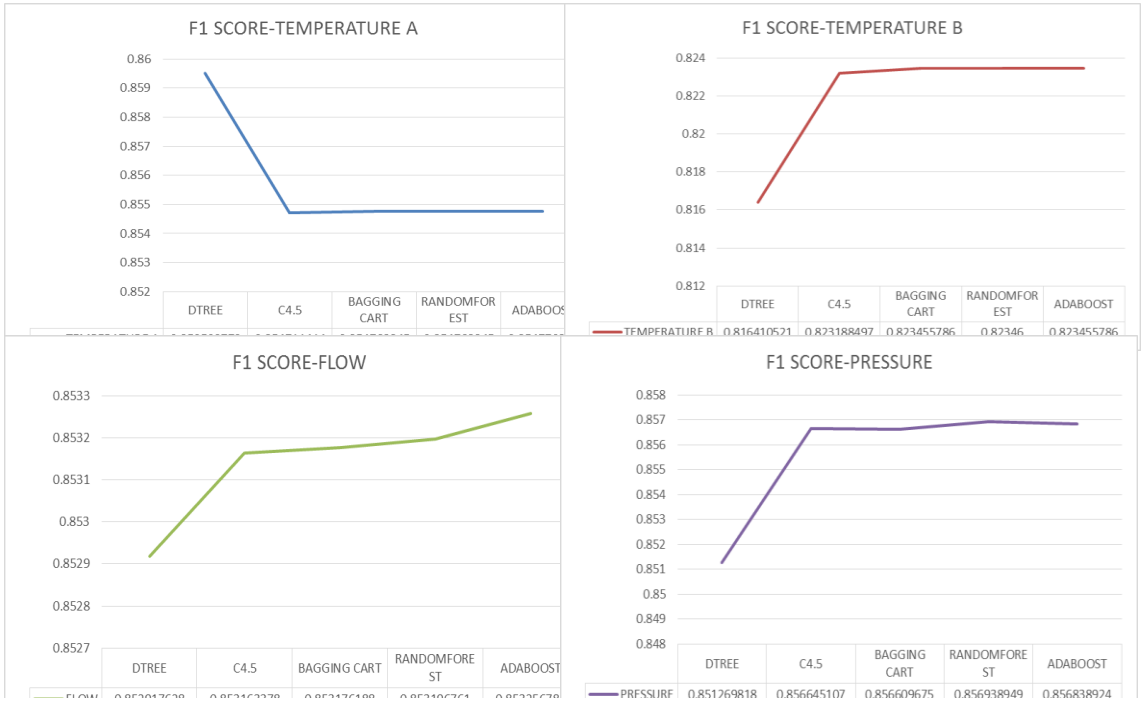

Figure.6. F1 Score

The values indicates that all the models are having a good score ranging from 0.816 to 0.859. Differences in their scores are very minimum and had to be identified with a high resolution of the axes. Again the performance of the models C4.5, Bagging Cart and random Forest are near equal and consistent for all the parameters.

6.4 Model Fitness (SSPN)

As can be seen from the global accuracy and F1 score, the performance of the models are near equal with only marginal differences among the two. Hence it becomes essential to derive comprehensive metrics which can better highlight the performance for meaningful comparisons. For a multiclass environment, the performance of the model is best assessed by combining individual metrics for the desired class into a single fitness function. We chose the SSPN fitness function that is based on all the four statistical indexes: sensitivity (SE), specificity (SP), positive predictive value (PPV), and negative predictive value (NPV). The fitness function SSPN is given as

$$SSPN = SE \times SP \times PPV \times NPV \qquad (8)$$

Thus, for evaluating the fitness f of an individual model, the following equation is used:

$$f = SSPN \times 1000 \qquad (9)$$

Which obviously ranges from 0 to 1000, with 1000 corresponding to the ideal. The models are ranked based on the fitness scores.
The values of SE, SP, PPV and NPV calculated for the desired class are tabulated in Table.4 for all the models and all the parameters.

Table.4.Model Metrics for SSPN

S.No	Parameter	Model	Sensitivity(SE)/ Recall	Specificity(SP)	Positive Predicted Value(PPV)/ Precision	Negative Predicted Value(NPV)
1	Temperature-A	Conditional Inference Trees	0.75382	0.95578	0.99965	0.02346
		C4.5	0.74653	0.95516	0.99956	0.02668
		Bagging Tree	0.99956	0.02712	0.74661	0.95585
		Random Forest	0.99956	0.02712	0.74661	0.95585
		AdaBoost	0.74645	1.000	1.000	0.02586
2	Temperatue-B	Conditional Inference Trees	0.78830	0.69430	0.84660	0.60520
		C4.5	0.796	0.7047	0.8523	0.6174

		Bagging Tree	0.8523	0.6185	0.7965	0.7051
		Random Forest	0.8523	0.6184	0.7965	0.705
		AdaBoost	0.7963	0.7044	0.8519	0.6182
3	Flow	Conditional Inference Trees	0.74453	0.91189	0.99824	0.05057
		C4.5	0.7452	0.89834	0.99771	0.05605
		Bagging Tree	0.74406	0.05679	0.74532	0.89262
		Random Forest	0.99764	0.05654	0.74529	0.89652
		AdaBoost	0.74406	0.97669	0.99958	0.99899
4	Pressure	Conditional Inference Trees	0.86550	0.67650	0.83750	0.72300
		C4.5	0.8701	0.6866	0.8436	0.7313
		Bagging Tree	0.8425	0.7343	0.8712	0.686
		Random Forest	0.84370	0.73240	0.8706	0.6871
		AdaBoost	0.8698	0.9995	0.6322	0.9998

The f score of the models under study calculated for the most prevalent class in each parameters are plotted as in Figure.7.

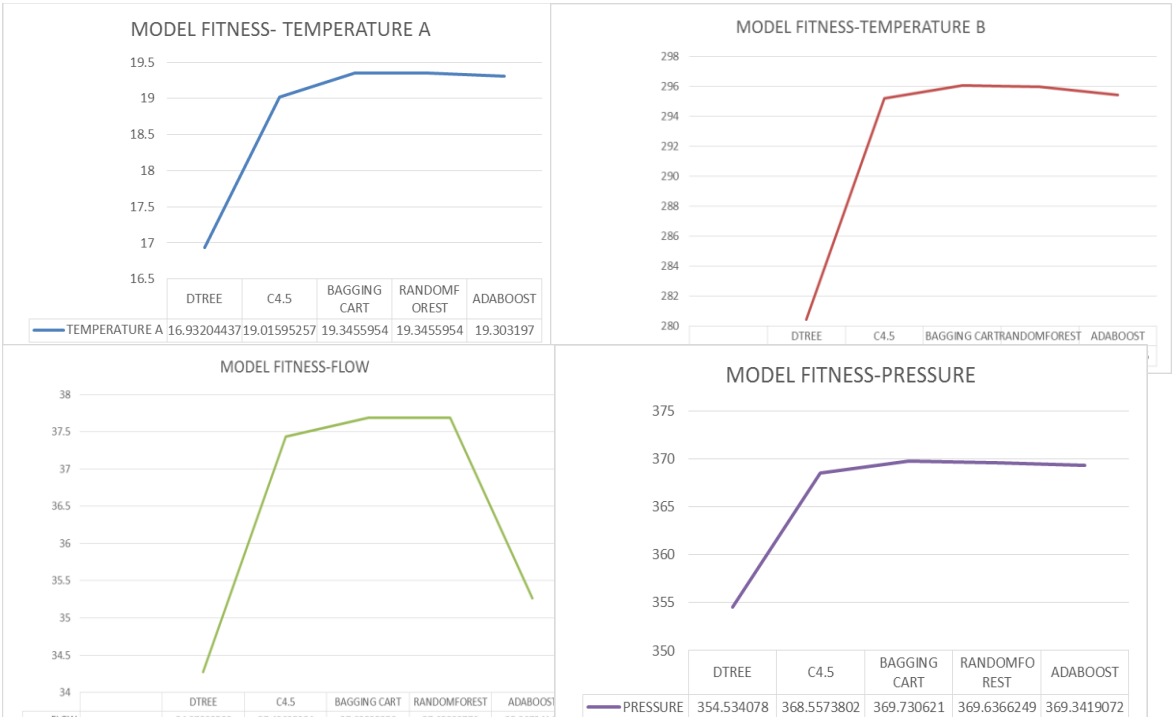

Figure.7. Model Fitness (SSPN)

The plot indicates that the score is very low in the range of 17 to 38 for the parameters Temperature A & Flow. This is due to high Type-1 error in ensemble models and high Type-2 error in Conditional Inference Trees & C4.5 for the prediction of negative classes. Further there is a very high class imbalance seen in the data tuples for these parameters with the prevalence values being around 0.98 for all the models. Similarly the score in case of Temperature B & Pressure is in the range of 280 to 370. The prevalence values for the desired class in these parameters are in the range 0.65 to 0.68.

The f score indicates that Conditional Inference Trees are inferior in their performance when compared to other models for each of the variable. The performance of the models C4.5, Bagging Cart and random Forest are higher and consistent for all the parameters. Out of the three, Bagging Cart and random Forest models have shown marginally higher performance compared to C4.5. Adaboost model had a lower performance in case of flow. For other parameters the model performed comparable (slightly lower) than other ensemble models.

6.5 Area under the Curve (AUC)

Receiver operating characteristics (ROC) analysis is a yet another visualization of prediction performance based on the trade-offs between sensitivity and specificity. To compare the models in a comprehensive scale, the receiver operating characteristics curve is plotted with 1-specificity also called FPR on x-axis and sensitivity (TPR) on the y-axis. The Area under the Curve (AUC) is calculated using the R package 'pROC' as per ref [20]. The area under the ROC curve indicates the measure of goodness for prediction by the model. A value of 0.5 indicates random and useless classification while 1 would indicate perfect classifier. The

various AUC values for the models under this study are calculated using the data in Table.3 and are plotted in Figure.8.

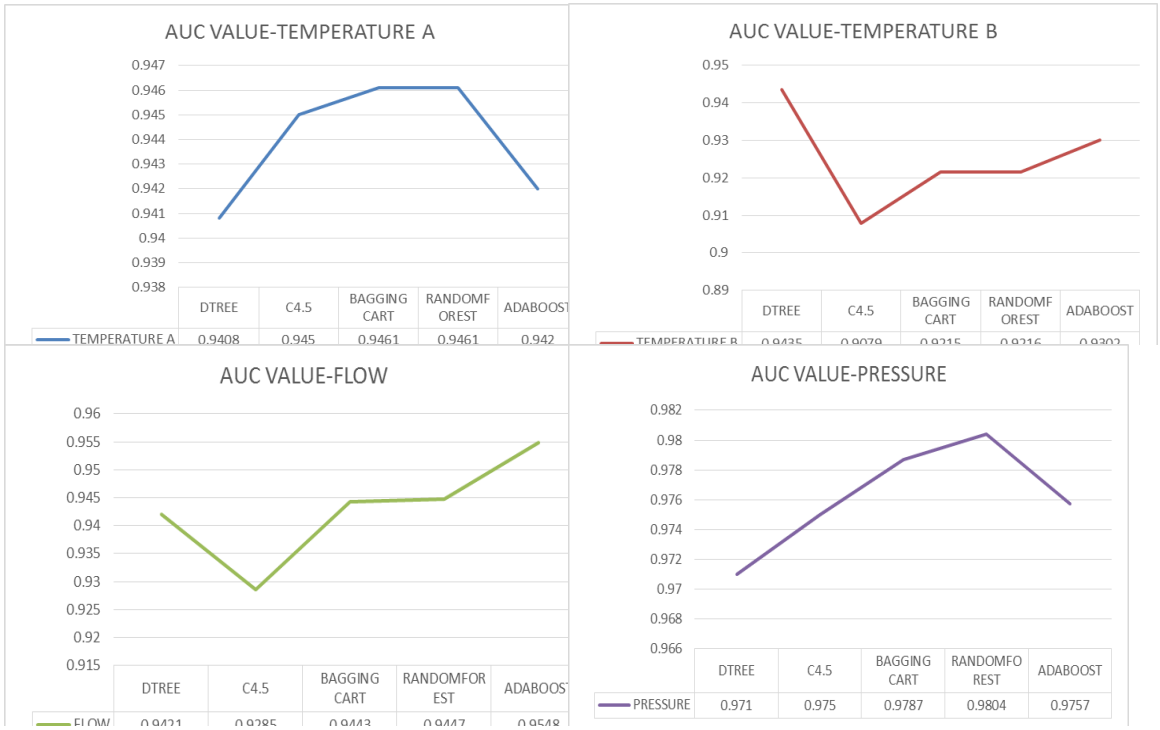

Figure.8. Area under the Curve (AUC) values

From the plots it is found that the models cannot be ignored for prediction classification as the AUC values are ranging from 0.9079 to 0.9804. Random Forest model exhibited highest value for all the parameters except Temperature-B. Conditional Inference Tree model performed the best for temperature-B but was lowest for temperature-A and Pressure. Adaboost model performed the best for Flow and better than bagging tree and random Forest for Temperature-B. But for other parameters the model performed poorer.

Similarly C4.5 model was lowest in its AUC Value for the parameters Temperature-B and Flow. Both Bagging Tree and random Forest performed consistently equally for all the parameters better than other models.

VII. Conclusion

This study dealt the development of various Decision Tree classification models that are trained with the key process variables of a Nuclear Power Plant Process that are recorded and available in the Distributed Control System (DCS) and evaluated their performance to estimate of a real time process data under steady state conditions without giving a time treatment while processing. With these models that can use the real time data, we can optimise the values of the key parameters, and predict their values when there is a shift in the process to help the operator to understand whether the plant is operated under an optimum efficiency. Different supervised machine-learning algorithms based on Decision Trees and their ensemble models using bagging and boosting techniques were developed and evaluated for their suitability for estimating the key performance variables. The performance of the models were assessed using a test data set calculating various metrics based on their predictions. The study results show that all the models are closer in terms of Global Accuracy values between 74.32% to 80.77%. Two ensemble models viz. Bagging Tree and Random Forest have a marginally better performance. The models Conditional Inference Trees and C4.5 are not dependable for Temperature A and Flow as their NIR values are higher than Accuracy.

On the basis of derived selection measures, all the models have a comparable performance for all the parameters in terms of F1 scores. The performance of the models C4.5, Bagging Cart and random Forest are near equal and consistent for all the parameters. On SSPN index, Conditional Inference Trees are found to be inferior to the other models. On this score also the ensemble models Bagging Tree and Random Forest have a better performance with C4.5 closely comparable.

On the basis of AUC values, both ensemble models Bagging Tree and random Forest performed consistently equally for all the parameters better than other models.

Thus although utility of all decision tree models for Process parameters prediction cannot be neglected, ensemble like bagging Trees and random Forests have given a better results.

Further study can be done on the performance by modifying the class levels with lower bin sizes, using custom ensemble techniques with multiple type of models instead of single type like Decision Trees.

References

[1] D. Flynn, J. Ritchie, and M. Cregan, "Data mining techniques applied to power plant performance monitoring," in *IFAC Proceedings Volumes (IFAC-PapersOnline)*, 2005, p. Volume-38,Issue-1,pages-369-374.

[2] J. Q. Li, C. L. Niu, J. Z. Liu, and L. Y. Zhang, "Research and application of data mining in power plant process control and optimization," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)(Vol. 3930 LNAI, pp. 149–158).*, 2006, pp. 149–158, doi: 10.1007/11739685_16.

[3] T. Ogilvie, E. Swidenbank, and B. W. Hogg, "Use of Data Mining Techniques in the Performance Monitoring and Optimisation of a Thermal Power Plant," in *IEE Colloquium on Knowledge Discovery and Data Mining (1998/434), London, UK*, 1998, pp. 7/1-7/4, doi: 10.1049/ic:19980647.

[4] V. R. S.Narasimhan, "Application of data mining techniques for sensor drift analysis to optimize nuclear power plant performance," *Int. J. Innov. Technol. Explor. Eng.*, vol. 9, no. 1, pp. 3087–3095, 2019, doi: 10.35940/ijitee.A9139.119119.

[5] V. R. S.Narasimhan, "Optimization of a Process System in Nuclear Power Plant- A Data Mining Approach," *Grenze Int. J. Eng. Technol. Spec. Issue*, vol. Grenze ID:, no. 6.2.1, pp. 1–11, 2020.

[6] S. Narasimhan and R. Velayudham, "Classification Models Based on Association Rules for Estimation of Key Process Variables in Nuclear Power Plant," vol. 100, no. 4, pp. 315–330, 2020.

[7] D. Davidov, O. Tsur, and A. Rappoport, "Enhanced sentiment learning using twitter hashtags and smileys," in *Coling 2010 - 23rd International Conference on Computational Linguistics, Proceedings of the Conference*, 2010.

[8] M. W. Ahmad, M. Mourshed, and Y. Rezgui, "Tree-based ensemble methods for predicting PV power generation and their comparison with support vector regression," *Energy*, vol. 164, pp. 465–474, 2018, doi: 10.1016/j.energy.2018.08.207.

[9] D. P. Gaikwad and R. C. Thool, "Intrusion detection system using Bagging with Partial Decision Tree base classifier," in *Procedia Computer Science*, 2015, vol. 49, no. 1, pp. 92–98, doi: 10.1016/j.procs.2015.04.231.

[10] N. G. Relan and D. R. Patil, "Implementation of network intrusion detection system using variant of decision tree algorithm," in *2015 International Conference on Nascent Technologies in the Engineering Field, ICNTE 2015 - Proceedings*, 2015, pp. 3–7, doi: 10.1109/ICNTE.2015.7029925.

[11] J. P. Jiawei Han, Micheline Kamber, *Data mining : concepts and techniques*, 3rd ed. Morgan Kaufmann Publishers, 2012.

[12] T. Hothorn, K. Hornik, and A. Zeileis, "Unbiased Recursive Partitioning: A Conditional Inference Framework," *J. Comput. Graph. Stat.*, vol. 15, no. 3, pp. 651–674, Sep. 2006, doi: 10.1198/106186006X133933.

[13] A. Zeileis, T. Hothorn, and K. Hornik, "Model-Based Recursive Partitioning," *J. Comput. Graph. Stat.*, vol. 17, no. 2, pp. 492–514, Jun. 2008, doi: 10.1198/106186008X319331.

[14] C. Strobl, A.-L. Boulesteix, A. Zeileis, and T. Hothorn, "Bias in random forest variable importance measures: Illustrations, sources and a solution," *BMC Bioinformatics*, vol. 8, no. 1, p. 25, 2007, doi: 10.1186/1471-2105-8-25.

[15] S. L. Salzberg, "C4.5: Programs for Machine Learning by J. Ross Quinlan. Morgan Kaufmann Publishers, Inc., 1993," *Mach. Learn.*, vol. 16, no. 3, pp. 235–240, 1994, doi: 10.1007/BF00993309.

[16] L. Breiman, "Bagging Predictors," *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, 1996, doi: 10.1023/A:1018054314350.

[17] Y. Freund, R. E. Schapire, and M. Hill, "Experiments with a New Boosting Algorithm," 1996.

[18] L. Breiman, "Random Forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.

[19] M. Kuhn, "Classification and Regression Training [R package caret version 6.0-86]," 2020, Accessed: Jun. 10, 2020. [Online]. Available: https://cran.r-project.org/package=caret.

[20] X. Robin *et al.*, "pROC : an open-source package for R and S + to analyze and compare ROC curves," *BMC Bioinformatics*, no. 12, p. Article No.77, 2011, doi: 10.1186/1471-2105-12-77.

S.Narasimhan, et. al. "Decision Trees for Nuclear Power Plant Performance Optimization." *IOSR Journal of Computer Engineering (IOSR-JCE)*, 23(2), 2021, pp. 51-62.