**DS CP – Algorithm Modification Of Random Forest**

**Abstract –**

Imagine you are trying to classify the outcome of an event for a large dataset. How will you choose the best algorithm to do that? For the above problem we have various algorithms like Random Forest, Support Vector Machine, Neural Networks, Naive Bayes, K-Nearest Neighbours, Decision Tree, Logistic Regression etc. One of the most used methods is Random Forest algorithm, which uses multiple Decision Trees to make predictions. However, this algorithm relies on a complex calculation called Information Gain, which measures the how much information a certain attribute has its impact in the data. Information Gain is a function that uses initial entropy and weighted entropy to find impact which can be slow and inaccurate to compute. Is there a better way to calculate information gain? In this research, we have explored a different way to calculate the Gain using the Association function expression. It is a value which is calculated with the help of three values i.e. no. positive attributes, no. of negative attributes and no. of types of attributes. Also, it is a powerful mathematical tool that can approximate any function by using its derivatives. We further modified the Random Forest algorithm by replacing the conventional expression with the association function expression in the Gian formula. We then tested our modified algorithm on various datasets and compared its performance with the original Gain formula.

**Keyword –**

Random forest, decision tree, classification, prediction, Information Gain, Association function, modification

**Methodology –**

In this project we have implemented a modified version of Random Forest algorithm, which is a machine learning method that uses multiple decision trees to make predictions and classify the data. Information Gain is a entity which tells about the impact of that attribute on the prediction class. We first calculate Association function (AF) for each attribute and these obtained values are further used to calculate the normalised gain for each attributes. Now this normalised gain is combined with old (initial) gain of attributes to get new gain for each attribute which is used as standard for making decisions.

**Pseudo code –**

Import all the important libraries

Class Node:

Define the structure for the node of decision tree

Class Decision Tree

Define the structure of decision tree and implement few functions for splitting, growing, and information gain from the dataset, etc.

Class Random Forest

Define the structure of random forest and implement few functions for fitting the model, predicting outcomes, etc.

Main methodology (main function)

For each dataset apply original and modified algorithm one by one and print the accuracy, precision and recall

Call the main function

Evaluate and analyse the result and compare the performance of the modified algorithm with the original one.

Data Source:

We have used two random datasets to test the performance of our modified Random Forest algorithm and then finally applied it on the dataset with class imbalance named Placement status.

Placement Status: This dataset is obtained from Kaggle which has different attributes like CGPA, soft skills training, SSC marks, etc. and predict the placement status of the following students based on his respective attributes.

Soft Skills Rating: This is a dataset that consist of 1599 samples of soft skills rating. Each sample has 11 features that are related to the placement status of an institute such as fixed CGPA, extra-curricular activities, placement training and so on. The task is to predict the placement status of a student based on his achievements.

Algorithm Selection: We chose the Random Forest algorithm as our base model because it is robust and versatile machine learning method that can handle both classification and regression problems. Also, it has advantages such as reducing overfitting, handling missing values, and providing feature importance scores.

Algorithm Modification: Then we have modified the Random Forest algorithm by using the Association function expression to approximate the normalised gain in the Information gain formula. The Association Gain expression for normalised gain is:

By using this approximation, we can calculate the normalised gain for specific attributes.

After calculating the association function we can calculate the Normalised gain from the above found values.

Where 0 < K <= m

After calculating the normalised gain, we have to multiply the original gain value with the normalised gain value and use the obtained value.

We have implemented this modification by creating a custom function that takes an array of probabilities as an input and returns an array of approximated logarithms as output. Then we have used the above function in our Entropy formula for calculation of each node split.

Algorithm Training: We have trained our modified Random Forest algorithm on each training subset using different hyperparameters such as number of trees (n\_trees), maximum depth (max\_depth), minimum samples split (min\_samples\_split) and so on. We then used a grid search technique to find the optimal combination of hyperparameters for each dataset.

Algorithm Testing: Then we have tested our modified Random Forest algorithm on each testing subset and evaluated its performance using varioius metrics such as accuracy, precision, recall, F1-score, etc. We have also compared its performance with the original version of the Random Forest algorithm on each dataset.

**Original Formula:**

**Modified Formula :**

**Calculations –**

We will consider 10 values from a feature present in our dataset (CM1) named: CYCLOMATIC\_DENSITY

We will be finding the absolute value for each observation

The values are as follow:

|  |
| --- |
| 4.4 |
| 4 |
| 4.8 |
| 4.4 |
| 4.5 |
| 4.2 |
| 4 |
| 3.5 |
| 3.9 |
| 4.4 |

For first value (4.4),

E (0.2) = + + + + + + + + + = 1.6094

similarly,

E (0.13) = + …………………+ = 2.0402

E (0.15) = + …………………+ = 1.8971

E (0.17) = + …………………+ = 1.7719

E (0.12) = + …………………+ = 2.1202

E (0.2) = + …………………+ = 1.6094

E (0.14) = + …………………+ = 1.9661

E (0.28) = + …………………+ = 1.2729

E (0.11) = + …………………+ = 2.2072

E (0.17) = + …………………+ = 1.771

|  |  |
| --- | --- |
| Value | Entropy |
| 0.2 | 1.6094 |
| 0.13 | 2.0402 |
| 0.15 | 1.8971 |
| 0.17 | 1.7719 |
| 0.12 | 2.1202 |
| 0.2 | 1.6094 |
| 0.14 | 1.9661 |
| 0.28 | 1.2729 |
| 0.11 | 2.2072 |
| 0.17 | 1.7719 |

**Result and Discussion –**

The existing technique use Entropy to calculate the quality of each node split in the decision tree. Here we are using the modified Entropy (using Taylor series) formula because we want to explore a different way to calculate the natural logarithm used in the original Entropy formula. By modifying the algorithm, we have enhanced the potential increase in the accuracy, precision and recall of our dataset. After computing the results for random datasets, we have applied the modified algorithm on the main dataset.

Here is the result obtained on applying original and modified algorithm on random datasets:

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Precision | Recall | Accuracy |
| Wine (original) | 0.65 | 0.57 | 0.68 |
| Wine (modified) | 0.69 | 0.60 | 0.69 |
| Diabetes(original) | 0.67 | 0.62 | 0.75 |
| Diabetes(modified) | 0.69 | 0.64 | 0.77 |

From the results obtained by the random datasets, we can clearly observe the change in accuracy, precision and recall in the original Random Forest algorithm and the modified Random Forest algorithm. In Wine dataset, the accuracy has increased by 1% where as in Diabetes dataset, the accuracy has increased by 2%.

Similarly, the below table shows the comparison between the results obtained on applying the original Entropy formula and the modified Entropy formula using Taylor’s series on CM1 dataset:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Parameters | Precision (0) | Precision (1) | Recall (0) | Recall (1) | Accuracy |
| Original formula | 0.88 | 0.50 | 0.98 | 0.11 | 0.8695 |
| Modified formula | 0.88 | 1.00 | 1.00 | 0.11 | 0.8840 |

From this table we can easily conclude that, the goal of achieving increased accuracy with a modified mathematics is successfully achieved through the modified algorithm. The accuracy has been increased by 2%. Below is the graph showing the class distribution before oversampling:

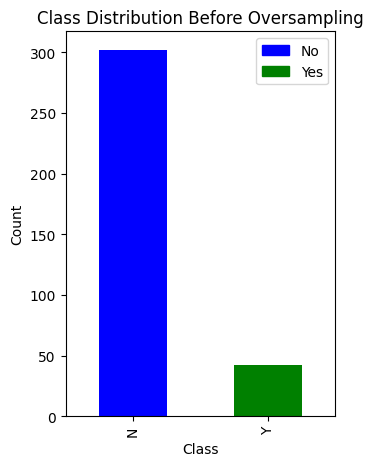


Figure 1

Below is the ROC curve for training as well as testing dataset for both original Random Forest Algorithm and modified Random Forest algorithm.

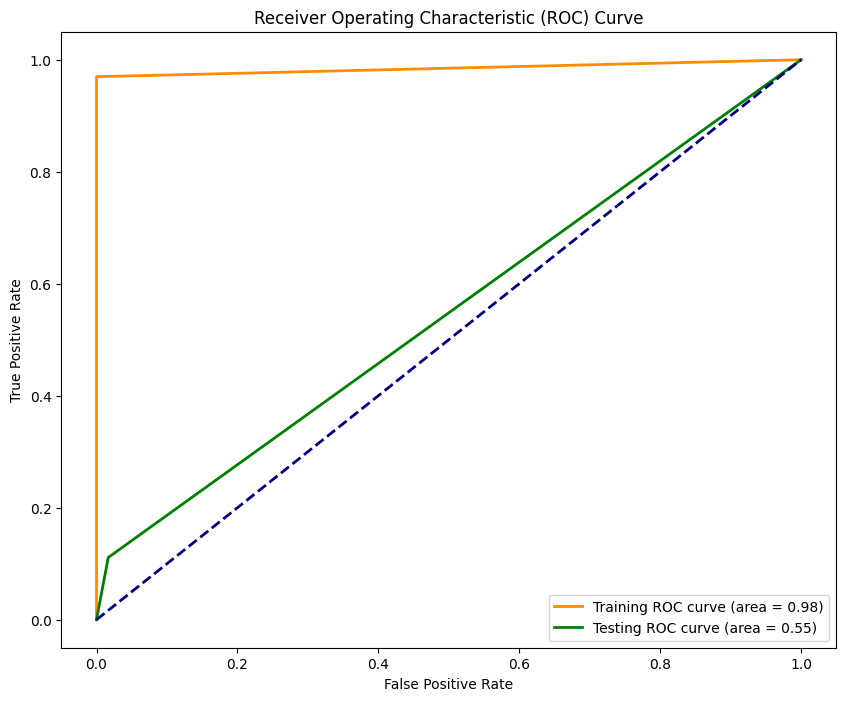


Figure 2.1 ROC curve for original Random Forest Algorithm

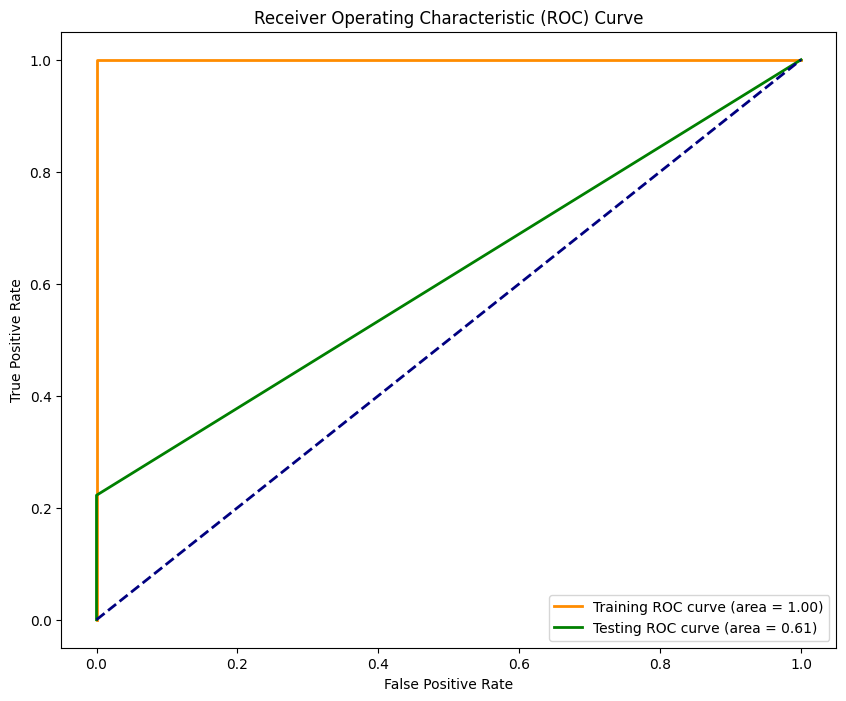


Figure 2.2 ROC curve for modified Random Forest Algorithm

From the above ROC curves, we can conclude that area under the Training ROC curve has increased by 2% and area under the Testing ROC curve has increased by 6%.