# ASSIGNING WEIGHTS TO TRAINING INSTANCES INCREASES CLASSIFICATION ACCURACY

Dr. Dewan Md. Farid[1] and Prof. Dr. Chowdhury Mofizur Rahman[2]

[1]Assistant Professor, Department of Computer Science & Engineering,
United International University, Dhaka-1209, Bangladesh
dewanfarid@cse.uiu.ac.bd
[2]Pro-Vice Chancellor, United International University, Dhaka-1209, Bangladesh
cmr@uiu.ac.bd

***ABSTRACT***

*The decision tree (DT) approach is most useful in classification problem. In conventional decision tree learning the weights of every training instances are set to one or equal value, which contradicts general intuition. In this paper, we proposed a new decision tree learning algorithm by assigning appropriate weights to each training instance in the training data that increases classification accuracy of the decision tree model. The main advantage of this proposed approach is to set appropriate weights to training instances using naïve Bayesian classifier before trying to construct the decision tree. In our approach the training instances are assigned to weight values based on the posterior probability. The training instances having less weight values are either noisy or posses unique characteristics compared to other training instances. The experimental results manifest that the proposed approach for decision tree construction can achieve high classification accuracy with compare to traditional decision tree algorithms on different types of benchmark datasets from UCI machine learning repository.*

***KEYWORDS***

*Bayesian Classifier, Classification, Decision Tree, Training Instance, Weights*

## 1. INTRODUCTION

The decision tree (DT) learning is most powerful and popular decision support tools of machine learning in classification problems, which is used in many real world applications like: medical diagnosis, radar signal classification, weather prediction, credit approval, and fraud detection etc. It has several advantages: DT is simple to understand and can deal with huge volume of dataset, because the tree size is independent of the dataset size. DT model can be combined with other machine learning models. DT can be constructed from dataset with many attributes and each attribute having many attribute values. Once the decision tree construction is complete, it can used to classify seen or unseen training instances. To make a decision using a DT, start at the root node and follow the tree down the branches until a leaf node representing the class is reached. There have been many decision tree algorithms like ID3 [1], C4.5 [2], CART [3], and SPRINT [4] but optimal decision tree construction for large volume of dataset is still a problem.

The decision tree, which is also known as classification tree or regression tree is a method commonly used in data mining or machine learning. The goal is to create a model that predicts the value of a target variable based on several input variables. DT building algorithms may initially build the tree and then prune it for more effective classification. With pruning technique, portions of the tree may be removed or combined to reduce the overall size of the tree. The time and space complexity of constructing a decision tree depends on the size of the dataset, the

DOI : 10.5121/ijdkp.2013.3102 

number of attributes in the dataset, and the shape of the resulting tree. Decision trees are used to classify data with common attributes. Each decision tree represents a rule set which categorizes data according to these attributes. A decision tree consists of nodes, leaves, and edges. A node of a tree specifies an attribute by which the data is to be partitioned. Each node has a number of edges which are labelled according to a possible value of the attribute in the parent node. An edge connects either two nodes or a node and a leaf. Leaves are labelled with a decision value for categorization of the data.

A naive Bayesian (NB) classifier is a simple probabilistic classifier in machine learning technique, which provides a probabilistic approach for performing supervised learning [5], [6]. It provides an optimal way to predict the class of an unknown example, and widely used in many field of data mining, image processing, bio-informatics and information retrieval etc. In Bayesian classifier conditional probabilities for each attribute value are calculated from the given dataset and then these probabilities are used to classify the known or unknown examples. The advantage of NB classifier is that it only requires a small amount of training dataset to estimate the parameters necessary for classification. In this paper, we proposed a new decision tree learning algorithm by assigning appropriate weights to training instances, which improve the classification accuracy. The weights of the training instances are calculated using naïve Bayesian classifier. Weight of each training instance is calculated with the maximum value of the class conditional probabilities. Our proposed algorithm calculated the information gain by using these weights and builds the decision tree model for decision making.

The remainder of this paper is organized as follows. Section 2 provides an overview of decision tree algorithms and naïve Bayesian classifier. Section 3 provides our proposed algorithm and an illustrative example using a small training dataset. Experimental results using benchmark data sets are presented in section 4. Finally, section 5 makes some concluding remarks along with suggestions for further improvement.

## 2. MINING ALGORITHMS

This section describes the basic underpinning concepts of decision tree algorithms and the naïve Bayesian classifier.

### 2.1. Decision Tree Learning

The ID3 algorithm builds decision tree using information theory, which choose splitting attributes from a dataset with the highest information gain. The amount of information associated with an attribute value is related to the probability of occurrence. The concept used to quantify information is called entropy, which is used to measure the amount of randomness from a dataset. When all data in a set belong to a single class, there is no uncertainty, and then the entropy is zero. The objective of decision tree classification is to iteratively partition the given data set into subsets where all elements in each final subset belong to the same class. The entropy calculation is shown in equation 1. Given probabilities $p_1, p_2,..,p_s$ for different classes in the data set

$$Entropy: H(p_1,p_2,\ldots p_s) = \sum_{i=1}^{s} (p_i \log(1/p_i)) \quad (1)$$

Given a data set, $D$, $H(D)$ finds the amount of entropy in class based subsets of the data set. When that subset is split into s new subsets $S = \{D_1, D_2,\ldots,D_s\}$ using some attribute, we can again look at the entropy of those subsets. A subset of data set is completely ordered and does not need any further split if all examples in it belong to the same class. The ID3 algorithm calculates the information gain of a split by using equation 2 and chooses that split which provides maximum information gain.

$$Gain\ (D,S) = H(D) - \sum_{i=1}^{s} p(D_i)H(D_i) \quad (2)$$

The C4.5 (upgraded version of ID3) algorithm uses highest *Gain Ratio* in equation 3 for splitting purpose that ensures a larger than average information gain.

$$GainRatio(D,S) = \frac{Gain(D,S)}{H\left(\frac{|D_1|}{|D|},\ldots,\frac{|D_s|}{|D|}\right)} \quad (3)$$

The C5.0 algorithm (upgraded version of C4.5) improves the performance of building trees using boosting, which is an approach to combining different classifiers. But boosting does not always help when the training data contains a lot of noise. When C5.0 performs a classification, each classifier assigns a vote and the example is assigned to the class with the most number of votes. CART (Classification and Regression Trees) is a process of generating a binary tree for decision making. CART handles missing data and contains a pruning strategy. The SPRINT (Scalable Parallelizable Induction of Decision Trees) algorithm uses an impurity function called *gini* index to find the best split.

$$gini\ (D) = 1 - \sum p_j^2 \quad (4)$$

Where, $p_j$ is the probability of class $C_j$ in data set $D$. The goodness of a split of $D$ into subsets $D_1$ and $D_2$ is defined by

$$gini_{split}(D) = n_1/n(gini(D_1)) + n_2/n(gini(D_2)) \quad (5)$$

The split with the best *gini* value is chosen. A number of research projects for optimal feature selection and classification have been done, which adopt hybrid strategy involving evolutionary algorithm and inductive decision tree learning [7], [8], [9], [10].

## 2.2. Naive Bayesian Classifier

Naïve Bayesian (NB) classifier is a simple probabilistic classifier based on probability model, which can be trained very efficiently in a supervised learning. The NB classifier is given as input a set of training examples each of which is described by attributes $A_1$ through $A_k$ and an associated class, $C$. The objective is to classify an unseen example whose class value is unknown but values for attributes $A_1$ through $A_k$ are known and they are $a_1, a_2, \ldots, a_k$ respectively. The optimal prediction of the unseen example is the class value $c_i$ such that $P(C=c_i|A_1=a_1,\ldots A_k=a_k)$ is maximum. By Bayes rule this probability equals to

$$\arg\max_{c_i \in C} \frac{P(A_1=a_1,\ldots A_k=a_k \mid C=c_i)}{P(A_1=a_1,\ldots A_k=a_k)} P(C=c_i) \quad (6)$$

Where $P(C=c_i)$ is the prior probability of class $c_i$, $P(A_1=a_1,\ldots A_k=a_k)$ is the probability of occurrence of the description of a particular example, and $P(A_1=a_1,\ldots A_k=a_k|C=c_i)$ is the class conditional probability of the description of a particular example. The prior probability of a class can be estimated from training data. The probability of occurrence of the description of particular examples is irrelevant for decision making since it is the same for each class value $c$. Learning is therefore reduced to the problem of estimating the class conditional probability of all possible description of examples from training data. The class conditional probability can be written in expanded from as follows:

$$P(A_1=a_1,\ldots A_k=a_k|C=c_i) = P(A_1=a_1 |\ A_2=a_2\ \wedge \ldots A_k=a_k\ \wedge\ C=c_i) * P(A_2=a_2 |\ A_3=a_3\ \wedge \ldots A_k=a_k\ \wedge\ C=c_i) * P(A_3=a_3 |\ A_4=a_4\ \wedge \ldots A_k=a_k\ \wedge\ C=c_i) * P(A_4=a_4\ \wedge \ldots A_k=a_k\ \wedge\ C=c_i) \quad (7)$$

In NB, it is assumed that outcome of attribute $A_i$ is independent of the outcome of all other attributes $A_j$, given $c$. Thus class conditional probabilities become: $P(A_1=a_1,\ldots A_k=a_k|C=c_i) = \prod_{i=1}^{k} P(A_i = a_i \mid C = c_i)$ If the above value is inserted in equation 6 it becomes:

$$\equiv \arg\max_{c_i \in C} P(C=c) \prod_{i=1}^{k} P(A_i = a_i \mid C = c_i) \quad (8)$$

In Naïve Bayesian classifier, the probability values of equation 8 are estimated from the given training data. These estimated values are then used to classify unknown examples.

## 3. PROPOSED DECISION TREE LEARNING ALGORITHM

This section describes our proposed decision tree leaning algorithm.

### 3.1. Proposed Algorithm

Given a training dataset, the proposed algorithm initializes the weights of each training instance, $W_i$ by highest posterior probability for that training instance. Estimating the prior probability $P(Cj)$ for each class is calculated by how often each class occurs in the training data. For each attribute, $A_i$, the number of occurrences of each attribute value $A_{ij}$ can be counted to determine $P(A_{ij})$. Similarly, the probability $P(A_{ij} \mid C_j)$ can be estimated by how often each attribute value occurs in the class, $Cj$ in the training data. The probability $P(A_{ij} \mid C_j)$ are estimated for all values of attributes. The algorithm uses these probabilities to initialize the weights of each training instance. This is done by multiplying the probabilities of the different attribute values from the training instances. Suppose the training instance $e_i$ has independent attribute values $\{A_{i1}, A_{i2},\ldots,A_{ip}\}$. We already know $P(A_{ik} \mid C_j)$, for each class $C_j$ and attribute $A_{ik}$. We then estimate $P(e_i \mid C_j)$ by

$$P(e_i \mid C_j) = P(C_j) \prod_{k=1 \to p} P(A_{ij} \mid C_j) \quad (9)$$

To initialize the weight into a training instance, we can estimate the likelihood of $e_i$ in each class. The probability that $e_i$ is in a class is the product of the conditional probabilities for each attribute value. The posterior probability $P(C_j \mid e_i)$ is then found for each class. Then the weight of the training instance is assigned with the highest posterior probability for that training instance.

Now the algorithm finds the best splitting attribute with highest information gain value using the weights of training instances in training dataset to form a decision tree. Create a node and label with splitting attribute. First node is the root node of the decision tree. For each branch of the node, partition the instances and grow sub training datasets $D_i$ by applying splitting predicate to training dataset, $D$. For each sub training datasets $D_i$, if examples in $D_i$, are all of same class value, $C_i$ then the leaf node labeled with $C_i$. Else the process continues until each final subset belongs to the same class value or leaf node created. Then the decision tree construction is complete. The main procedure of proposed algorithm is described as follows.

Algorithm 1: Decision Tree Learning Algorithm using Weights

1. Calculate the probabilities $P(C_j)$ for each class $C_j$ and $P(A_{ij} \mid C_j)$ for each attribute values from training data, $D$.
2. Calculate the posterior probability for each instance in $D$. $P(_{ei} \mid _{Cj}) = P(_{Cj}) \prod P(_{Aij} \mid _{Cj})$
3. Assign the weights of training instances in $D$ with Maximum Likelihood (ML) of posterior probability $P(C_j|e_i)$; $W_i = P_{ML}(C_j|e_i)$.
4. Find the best splitting attribute with highest information gain value using the assigned weights, $W_i$ in training dataset, $D$.
5. Create a node and label with splitting attribute. [First node is the root node, $T$ of the decision tree]

6. For each branch of the node, partition the training instances and grow sub training datasets $D_i$ by applying splitting predicate to training dataset $D$.
7. For each sub training datasets $D_i$, if the training instances in $D_i$, are all of same class value, $C_i$ then the leaf node labeled with $C_i$. Else continues steps 4 to 7 until each final subset belong to the same class value or leaf node created.
8. When the decision tree construction is complete, classify the test dataset.

### 3.2. An Illustrative Example

To illustrate the operation of our proposed algorithm, we consider a small dataset in Table 1 described by four attributes namely Outlook, Temperature, Humidity, and Wind, which represent the weather condition of particular day. Each attribute has some attribute values. The Play attribute in Table 1 represents the class of each instance, which says whether a particular weather condition is suitable for playing tennis or not.

Table 1. Playing tennis dataset.

| Day | Outlook | Temp. | Hum. | Wind | Play |
|---|---|---|---|---|---|
| $D_1$ | Sunny | Hot | High | Weak | No |
| $D_2$ | Sunny | Hot | High | Strong | No |
| $D_3$ | Overcast | Hot | High | Weak | Yes |
| $D_4$ | Rain | Mild | High | Weak | Yes |
| $D_5$ | Rain | Cool | Normal | Weak | Yes |
| $D_6$ | Rain | Cool | Normal | Strong | No |
| $D_7$ | Overcast | Cool | Normal | Strong | Yes |
| $D_8$ | Sunny | Mild | High | Weak | No |
| $D_9$ | Sunny | Cool | Normal | Weak | Yes |
| $D_{10}$ | Rain | Mild | Normal | Weak | Yes |
| $D_{11}$ | Sunny | Mild | Normal | Strong | Yes |
| $D_{12}$ | Overcast | Mild | High | Strong | Yes |
| $D_{13}$ | Overcast | Hot | Normal | Weak | Yes |
| $D_{14}$ | Rain | Mild | High | Strong | No |

Now the prior probability for each class, and conditional probabilities for each attribute value are calculated using the playing tennis dataset in Table 1 and these probabilities are enumerated below:

Prior probability for each class:

P(Play=Yes) = 9/14 = 0.642
P(Play=No) = 5/14 = 0.375

Conditional probabilities for each attribute value:

P(Outlook=Sunny | Play=Yes) = 2/9 = 0.222
P(Outlook=Sunny | Play=No) = 3/5 = 0.6
P(Outlook=Overcast | Play=Yes) = 4/9 = 0.444
P(Outlook=Overcast | Play=No) = 0/5 = 0.0
P(Outlook=Rain | Play=Yes) = 3/9 = 0.3
P(Outlook=Rain | Play=No) = 2/5 = 0.4
P(Temperature=Hot | Play=Yes) = 2/9 = 0.222
P(Temperature=Hot | Play=No) = 2/5 = 0.4
P(Temperature=Mild | Play=Yes) = 4/9 = 0.444
P(Temperature=Mild | Play=No) = 2/5 = 0.4
P(Temperature=Cool | Play=Yes) = 3/9 = 0.333
P(Temperature=Cool | Play=No) = 1/5 = 0.2
P(Humidity=High | Play=Yes) = 3/9 = 0.333

P(Humidity=High | Play=No) = 4/5 = 0.8
P(Humidity=Normal | Play=Yes) = 6/9 = 0.666
P(Humidity=Normal | Play=No) = 1/5 = 0.2
P(Wind=Weak | Play=Yes) = 6/9 = 0.666
P(Wind=Weak | Play=No) = 2/5 = 0.4
P(Wind=Strong | Play=Yes) = 3/9 = 0.333
P(Wind=Strong | Play=No) = 3/5 = 0.6

Now calculate the posterior probabilities for training instances and assign the weights of each training instance with highest posterior probability using dataset in Table 1. Table 2 shows the assigned weights of training instances in training dataset.

Table 2. Assigned weights in training examples.

| Day | Play=Yes | Play=No | Weights |
|---|---|---|---|
| $D_1$ | 0.007 | 0.027 | $W_1 = 0.027$ |
| $D_2$ | 0.003 | 0.043 | $W_2 = 0.043$ |
| $D_3$ | 0.014 | 0.0 | $W_3 = 0.014$ |
| $D_4$ | 0.018 | 0.018 | $W_4 = 0.018$ |
| $D_5$ | 0.028 | 0.002 | $W_5 = 0.028$ |
| $D_6$ | 0.142 | 0.003 | $W_6 = 0.142$ |
| $D_7$ | 0.021 | 0.0 | $W_7 = 0.021$ |
| $D_8$ | 0.014 | 0.013 | $W_8 = 0.014$ |
| $D_9$ | 0.010 | 0.005 | $W_9 = 0.010$ |
| $D_{10}$ | 0.037 | 0.004 | $W_{10} = 0.037$ |
| $D_{11}$ | 0.014 | 0.010 | $W_{11} = 0.014$ |
| $D_{12}$ | 0.014 | 0.0 | $W_{12} = 0.014$ |
| $D_{13}$ | 0.014 | 0.0 | $W_{13} = 0.014$ |
| $D_{14}$ | 0.009 | 0.027 | $W_{14} = 0.027$ |

Now, calculate the information gain for each attribute (outlook, temperature, humidity, and wind) using weights, and then selects the one with highest information gain. The gain values for four attributes are given bellow:

$$\text{Info(playing tennis data set)} = -\left(\frac{\sum_{i=YES} W_i}{\sum_{i=1}^{14} W_i}\log_2\frac{\sum_{i=YES} W_i}{\sum_{i=1}^{14} W_i}\right)-\left(\frac{\sum_{i=NO} W_i}{\sum_{i=1}^{14} W_i}\log_2\frac{\sum_{i=NO} W_i}{\sum_{i=1}^{14} W_i}\right)$$

$$= -\left(\frac{0.326}{0.423}\log\frac{0.326}{0.423}\right)-\left(\frac{0.097}{0.423}\log\frac{0.097}{0.423}\right) = 0.777$$

$$\text{Info(outlook)} = \frac{\sum_{i=SUNNY} W_i}{\sum_{i=1}^{14} W_i}\left\{-\left(\frac{\sum_{i=SUNNY(YES)} W_i}{\sum_{i=SUNNY} W_i}\log\frac{\sum_{i=SUNNY(YES)} W_i}{\sum_{i=SUNNY} W_i}\right)-\left(\frac{\sum_{i=SUNNY(NO)} W_i}{\sum_{i=SUNNY} W_i}\log\frac{\sum_{i=SUNNY(NO)} W_i}{\sum_{i=SUNNY} W_i}\right)\right\}$$

$$+\frac{\sum_{i=OVER} W_i}{\sum_{i=1}^{14} W_i}\left\{-\left(\frac{\sum_{i=OVER(YES)} W_i}{\sum_{i=OVER} W_i}\log\frac{\sum_{i=OVER(YES)} W_i}{\sum_{i=OVER} W_i}\right)-\left(\frac{\sum_{i=OVER(NO)} W_i}{\sum_{i=OVER} W_i}\log\frac{\sum_{i=OVER(NO)} W_i}{\sum_{i=OVER} W_i}\right)\right\}+\frac{\sum_{i=RAIN} W_i}{\sum_{i=1}^{14} W_i}\left\{-\left(\frac{\sum_{i=RAIN(YES)} W_i}{\sum_{i=RAIN} W_i}\log\frac{\sum_{i=RAIN(YES)} W_i}{\sum_{i=RAIN} W_i}\right)-\left(\frac{\sum_{i=RAIN(NO)} W_i}{\sum_{i=RAIN} W_i}\log\frac{\sum_{i=RAIN(NO)} W_i}{\sum_{i=RAIN} W_i}\right)\right\}$$

$$=\frac{0.108}{0.423}\left\{-\left(\frac{0.038}{0.108}\log\frac{0.038}{0.108}\right)-\left(\frac{0.07}{0.108}\log\frac{0.07}{0.108}\right)\right\}+\frac{0.063}{0.423}\left\{-\left(\frac{0.063}{0.063}\log\frac{0.063}{0.063}\right)-\left(\frac{0.0}{0.063}\log\frac{0.0}{0.063}\right)\right\}$$

$$+\frac{0.252}{0.423}\left\{-\left(\frac{0.225}{0.252}\log\frac{0.225}{0.252}\right)-\left(\frac{0.027}{0.252}\log\frac{0.027}{0.252}\right)\right\} = 0.53$$

$$\text{Info(Temperature)}= \frac{\sum_{i=HOT} W_i}{\sum_{i=1}^{14} W_i}\left\{-\left(\frac{\sum_{i=HOT(YES)} W_i}{\sum_{i=HOT} W_i}\log_2\frac{\sum_{i=HOT(YES)} W_i}{\sum_{i=HOT} W_i}\right)-\left(\frac{\sum_{i=HOT(NO)} W_i}{\sum_{i=HOT} W_i}\log_2\frac{\sum_{i=HOT(NO)} W_i}{\sum_{i=HOT} W_i}\right)\right\}$$

$$+\frac{\sum_{i=Mild} W_i}{\sum_{i=1}^{14} W_i}\left\{-\left(\frac{\sum_{i=MILD(YES)} W_i}{\sum_{i=MILD} W_i}\log_2\frac{\sum_{i=MILD(YES)} W_i}{\sum_{i=MILD} W_i}\right)-\left(\frac{\sum_{i=MILD(NO)} W_i}{\sum_{i=MILD} W_i}\log_2\frac{\sum_{i=MILD(NO)} W_i}{\sum_{i=MILD} W_i}\right)\right\}+\frac{\sum_{i=COOL} W_i}{\sum_{i=1}^{14} W_i}\left\{-\left(\frac{\sum_{i=COOL(YES)} W_i}{\sum_{i=COOL} W_i}\log_2\frac{\sum_{i=COOL(YES)} W_i}{\sum_{i=COOL} W_i}\right)-\left(\frac{\sum_{i=COOL(NO)} W_i}{\sum_{i=COOL} W_i}\log_2\frac{\sum_{i=COOL(NO)} W_i}{\sum_{i=COOL} W_i}\right)\right\}$$

$$=\frac{0.098}{0.423}\left\{-\left(\frac{0.028}{0.098}\log_2\frac{0.028}{0.098}\right)-\left(\frac{0.07}{0.098}\log_2\frac{0.07}{0.098}\right)\right\}+\frac{0.124}{0.423}\left\{-\left(\frac{0.097}{0.124}\log_2\frac{0.097}{0.124}\right)-\left(\frac{0.027}{0.124}\log_2\frac{0.027}{0.124}\right)\right\}$$

$$+\frac{0.201}{0.423}\left\{-\left(\frac{0.201}{0.201}\log_2\frac{0.201}{0.201}\right)-\left(\frac{0.0}{0.201}\log_2\frac{0.0}{0.201}\right)\right\}=0.42$$

Info(Humidity)=

$$\frac{\sum_{i=HIGH} W_i}{\sum_{i=1}^{14} W_i}\left\{-\left(\frac{\sum_{i=HIGH(YES)} W_i}{\sum_{i=HIGH} W_i}\log_2\frac{\sum_{i=HIGH(YES)} W_i}{\sum_{i=HIGH} W_i}\right)-\left(\frac{\sum_{i=HIGH(NO)} W_i}{\sum_{i=HIGH} W_i}\log_2\frac{\sum_{i=HIGH(NO)} W_i}{\sum_{i=HIGH} W_i}\right)\right\}+\frac{\sum_{i=NORMAL} W_i}{\sum_{i=1}^{14} W_i}\left\{-\left(\frac{\sum_{i=NORMAL(YES)} W_i}{\sum_{i=NORMAL} W_i}\log_2\frac{\sum_{i=NORMAL(YES)} W_i}{\sum_{i=NORMAL} W_i}\right)-\left(\frac{\sum_{i=NORMAL(NO)} W_i}{\sum_{i=NORMAL} W_i}\log_2\frac{\sum_{i=NORMAL(NO)} W_i}{\sum_{i=NORMAL} W_i}\right)\right\}$$

$$=\frac{0.157}{0.423}\left\{-\left(\frac{0.06}{0.157}\log_2\frac{0.06}{0.157}\right)-\left(\frac{0.097}{0.157}\log_2\frac{0.097}{0.157}\right)\right\}+\frac{0.266}{0.423}\left\{-\left(\frac{0.266}{0.266}\log_2\frac{0.266}{0.266}\right)-\left(\frac{0.0}{0.266}\log_2\frac{0.0}{0.266}\right)\right\}=0.177$$

Info(Wind) =

$$\frac{\sum_{i=WEAK} W_i}{\sum_{i=1}^{14} W_i}\left\{-\left(\frac{\sum_{i=WEAK(YES)} W_i}{\sum_{i=WEAK} W_i}\log_2\frac{\sum_{i=WEAK(YES)} W_i}{\sum_{i=WEAK} W_i}\right)-\left(\frac{\sum_{i=WEAK(NO)} W_i}{\sum_{i=WEAK} W_i}\log_2\frac{\sum_{i=WEAK(NO)} W_i}{\sum_{i=WEAK} W_i}\right)\right\}+\frac{\sum_{i=STRONG} W_i}{\sum_{i=1}^{14} W_i}\left\{-\left(\frac{\sum_{i=STRONG(YES)} W_i}{\sum_{i=STRONG} W_i}\log_2\frac{\sum_{i=STRONG(YES)} W_i}{\sum_{i=STRONG} W_i}\right)-\left(\frac{\sum_{i=STRONG(NO)} W_i}{\sum_{i=STRONG} W_i}\log_2\frac{\sum_{i=STRONG(NO)} W_i}{\sum_{i=STRONG} W_i}\right)\right\}$$

$$=\frac{0.162}{0.423}\left\{-\left(\frac{0.135}{0.162}\log_2\frac{0.135}{0.162}\right)-\left(\frac{0.027}{0.162}\log_2\frac{0.027}{0.162}\right)\right\}+\frac{0.261}{0.423}\left\{-\left(\frac{0.191}{0.261}\log_2\frac{0.191}{0.261}\right)-\left(\frac{0.07}{0.261}\log_2\frac{0.07}{0.261}\right)\right\}=0.766$$

The information gains of four attributes are follows:

Outlook = 0.777-0.53= 0.247, Temperature = 0777-0.42= 0.357

Humidity = 0.777-0.177= 0.6, Wind = 0.777-0.766= 0.011

The gain value of humidity attribute is maximum than other attributes, so root node of decision tree will be humidity.

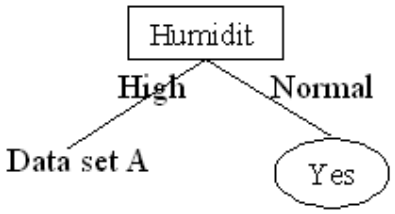


Figure 1. Root node of the tree.

Table 3. Subset data A for Humidity=High.

| Day | Outlook | Temp. | Wind | Play | Weight |
|---|---|---|---|---|---|
| $D_1$ | Sunny | Hot | Weak | No | $W_1$=0.027 |
| $D_2$ | Sunny | Hot | Strong | No | $W_2$=0.043 |
| $D_3$ | Overcast | Hot | Weak | Yes | $W_3$=0.014 |
| $D_4$ | Rain | Mild | Weak | Yes | $W_4$=0.018 |
| $D_8$ | Sunny | Mild | Weak | Yes | $W_8$=0.014 |
| $D_{12}$ | Overcast | Mild | Strong | Yes | $W_{12}$=0.014 |
| $D_{14}$ | Rain | Mild | Strong | No | $W_{14}$=0.027 |

Again calculate the information gain for each attribute (outlook, temperature, and wind) using subset data A with the weights, and then select the highest gain value. The gain values for remaining three attributes are given bellow:

$$\text{Info(Subset data A)} = \left\{-\left(\frac{W_3+W_4+W_8+W_{12}}{W_1+W_2+W_3+W_4+W_8+W_{12}+W_{14}}\log\frac{W_3+W_4+W_8+W_{12}}{W_1+W_2+W_3+W_4+W_8+W_{12}+W_{14}}\right)\right.$$

$$\left.-\left(\frac{W_1+W_2+W_{14}}{W_1+W_2+W_3+W_4+W_8+W_{12}+W_{14}}\log\frac{W_1+W_2+W_{14}}{W_1+W_2+W_3+W_4+W_8+W_{12}+W_{14}}\right)\right\}=\left\{-\left(\frac{0.06}{0.157}\log\frac{0.06}{0.157}\right)-\left(\frac{0.097}{0.157}\log\frac{0.097}{0.157}\right)\right\}=0.96$$

$$\text{Info (Outlook)} =\frac{W_1+W_2+W_8}{0.157}\left\{-\left(\frac{W_8}{W_1+W_2+W_8}\log\frac{W_8}{W_1+W_2+W_8}\right)-\left(\frac{W_1+W_2}{W_1+W_2+W_8}\log\frac{W_1+W_2}{W_1+W_2+W_8}\right)\right\}$$

$$+\frac{W_3+W_{12}}{0.157}\left\{-\left(\frac{W_3+W_{12}}{W_3+W_{12}}\log\frac{W_3+W_{12}}{W_3+W_{12}}\right)-\left(\frac{0.0}{W_3+W_{12}}\log\frac{0.0}{W_3+W_{12}}\right)\right\}+\frac{W_4+W_{14}}{0.157}\left\{-\left(\frac{W_4}{W_4+W_{14}}\log\frac{W_4}{W_4+W_{14}}\right)-\left(\frac{W_{14}}{W_4+W_{14}}\log\frac{W_{14}}{W_4+W_{14}}\right)\right\}=0.624$$

$$\text{Info(Temperature)} =\frac{W_1+W_2+W_3}{0.157}\left\{-\left(\frac{W_3}{W_1+W_2+W_3}\log\frac{W_3}{W_1+W_2+W_3}\right)-\left(\frac{W_1+W_2}{W_1+W_2+W_3}\log\frac{W_1+W_2}{W_1+W_2+W_3}\right)\right\}$$

$$+\frac{W_3+W_8+W_{12}+W_{14}}{0.157}\left\{-\left(\frac{W_3+W_8+W_{12}}{W_3+W_8+W_{12}+W_{14}}\log\frac{W_3+W_8+W_{12}}{W_3+W_8+W_{12}+W_{14}}\right)\right.$$

$$\left.-\left(\frac{W_{14}}{W_3+W_8+W_{12}+W_{14}}\log_2\frac{W_{14}}{W_3+W_8+W_{12}+W_{14}}\right)\right\}=0.788$$

$$\text{Info(Wind)}=\frac{W_1+W_3+W_4+W_8}{0.157}\left\{-\left(\frac{W_3+W_4+W_8}{W_1+W_3+W_4+W_8}\log\frac{W_3+W_4+W_8}{W_1+W_3+W_4+W_8}\right)-\left(\frac{W_1}{W_1+W_3+W_4+W_8}\log\frac{W_1}{W_1+W_3+W_4+W_8}\right)\right\}$$

$$+\frac{W_2+W_{12}+W_{14}}{0.108}\left\{-\left(\frac{W_{12}}{W_2+W_{12}+W_{14}}\log\frac{W_{12}}{W_2+W_{12}+W_{14}}\right)-\left(\frac{W_2+W_{14}}{W_2+W_{12}+W_{14}}\log\frac{W_2+W_{14}}{W_2+W_{12}+W_{14}}\right)\right\}=0.807$$

The gains of attributes of subset data A are:

Outlook = 0.96-0.624= 0.336, Temperature = 0.96-0.788= 0.172, and
Wind = 0.96-0.807= 0.153

The gain value of outlook attribute is maximum than other attributes using subset data A. So, after the root node (humidity) there will be outlook node.

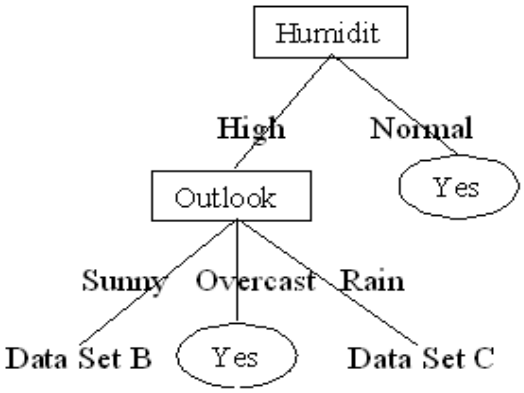


Figure 2. Tree after humidity and outlook node.

Table 4. Subset data B for outlook=sunny.

| Day | Temp. | Wind | Play | Weight |
|---|---|---|---|---|
| $D_1$ | Hot | Weak | No | $W_1$=0.027 |
| $D_2$ | Hot | Strong | No | $W_2$=0.043 |
| $D_8$ | Mild | Weak | Yes | $W_8$=0.014 |

Table 5. Subset data C for outlook=rain.

| Day | Temp. | Wind | Play | Weight |
|---|---|---|---|---|
| $D_4$ | Mild | Weak | Yes | $W_4$=0.018 |
| $D_{14}$ | Mild | Strong | No | $W_{14}$=0.027 |

Similarly, we will calculate the information gain of attributes of subset data B and C, and get the complete decision tree.

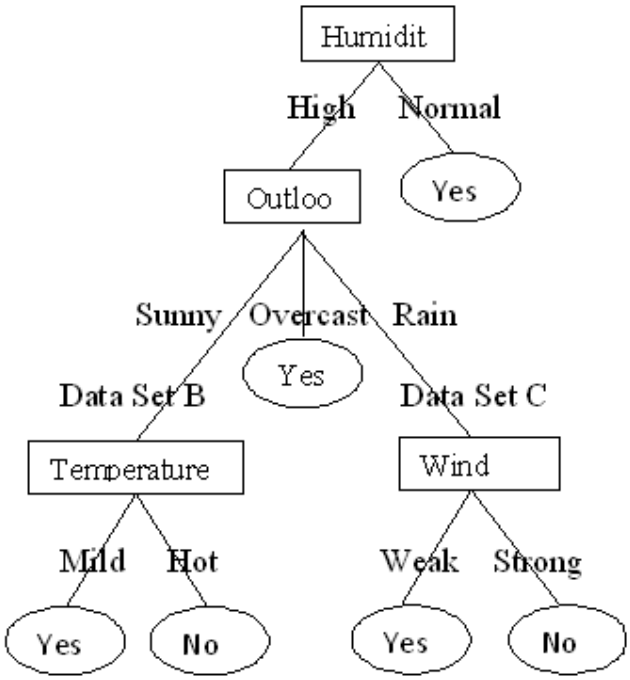


Figure 3. Complete decision tree using playing tennis dataset.

# 4. EXPERIMENTAL ANALYSIS

In this section, we describe the datasets and the experimental results.

## 4.1. Datasets

A set of data items called the dataset, which is the very basic concept of data mining and machine learning research. A dataset is roughly equivalent to a two-dimensional spreadsheet or database table. Table 6 describes about the datasets from UCI machine learning repository, which are used in experimental analysis [12].

1. Iris Plants Database: This is one of the best known dataset in the pattern recognition literature. This dataset contains 3 class values (Iris Setosa, Iris Versicolor, and Iris Virginica), where each class refers to a type of iris plant. There are 150 instances (50 in each of three classes) and 4 attributes (sepallength, sepalwidth, petallength, and petalwidth) in this dataset. One class is linearly separable from the other 2 classes.

2. Image Segmentation Data: The goal of this dataset is to provide an empirical basis for research on image segmentation and boundary detection. There are 1500 data instances in this dataset with 19 attributes and all the attributes are real. There are 7 class attribute values: brickface, sky, foliage, cement, window, path, and grass.

3. Large Soybean Database: There are 35 attributes in this dataset and all attributes are nominalized. There are 683 data instances and 19 class values in this dataset.

4. Fitting Contact Lenses Database: It is very small dataset with only 24 data instances, 4 attibutes and 3 class attribute values (soft, hard, and none). All the attribute values are nominal in this dataset. The instances are complete and noise free and 9 rules cover the training set.

5. NSL-KDD Dataset: The Knowledge Discovery and Data Mining 1999 (KDD99) competition data contains simulated intrusions in a military network environment. It is often used a benchmark to evaluate handling concept drift. NSL-KDD dataset is the new version of the KDD99 dataset, which solved some of the inherent problems of the KDD99 dataset [25]. Although, NSL-KDD dataset still suffers from some of the problems that discussed by McHugh [24]. The main advantage of NSL-KDD dataset is that the training and testing data points are reasonable, so it become affordable to run the experiments on the complete set of training and testing dataset without the need to randomly select a small portion of dataset. Each record in NSL-KDD dataset consists of 41 attributes and 1 class attribute. NSL-KDD dataset does not include redundant and duplicate examples in training dataset.

Table 6. Dataset Descriptions.

| Dataset | No of Attributes | Attribute Types | No of Instances | No of Class |
|---|---|---|---|---|
| Iris Plants | 4 | Real | 150 | 3 |
| Image Segmentation | 19 | Real | 1500 | 7 |
| Large Soybean | 35 | Nominal | 683 | 19 |
| Fitting Contact Lenses | 4 | Nominal | 24 | 3 |
| NSL-KDD | 41 | Real & Nominal | 25192 | 23 |

## 4.2. Results

We implement our algorithm in Java. The code for decision tree has been adapted from the Weka machine learning open source repository (http://www.cs.waikato.ac.nz/ml/weka). Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. The experiments were run on an Intel Core 2 Duo Processor 2.0 GHz processor (2 MB Cache, 800 MHz FSB) with 1 GB of RAM. The performance of decision tree models is measured by counting the proportion of correctly classified instances using 10-fold cross-validation. We compare the performance of our proposed DT learning algorithm with C4.5 and CART algorithms. The C4.5

algorithm is the upgraded version of ID3 decision tree learning algorithm. CART (Classification and Regression Trees) is a process of generating a binary tree, which can handle missing data and contain pruning strategy. The experimental results are shown in Table 7, Table 8, and Table 9.

Table 7. Result of C4.5 decision tree model.

| Dataset | Correctly Classified Instances | Incorrectly Classified Instances | Classification Rate (%) | Misclassification Rate (%) |
|---|---|---|---|---|
| Iris Plants | 144 | 6 | 96 | 4 |
| Image Segmentation | 1434 | 64 | 95.73 | 4.26 |
| Large Soybean | 625 | 58 | 91.50 | 8.49 |
| Fitting Contact Lenses | 20 | 4 | 83.33 | 16.66 |
| NSL-KDD | 22781 | 2411 | 90.42 | 9.57 |

Table 8. Result of CART decision tree model.

| Dataset | Correctly Classified Instances | Incorrectly Classified Instances | Classification Rate (%) | Misclassification Rate (%) |
|---|---|---|---|---|
| Iris Plants | 143 | 7 | 95.33 | 4.66 |
| Image Segmentation | 1426 | 74 | 95.06 | 4.93 |
| Large Soybean | 622 | 61 | 91.06 | 8.93 |
| Fitting Contact Lenses | 19 | 5 | 79.16 | 20.83 |
| NSL-KDD | 21233 | 3959 | 84.28 | 15.71 |

Table 9. Result of Proposed Decision Tree Learning Algorithm.

| Dataset | Correctly Classified Instances | Incorrectly Classified Instances | Classification Rate (%) | Misclassification Rate (%) |
|---|---|---|---|---|
| Iris Plants | 146 | 4 | 97.33 | 2.44 |
| Image Segmentation | 1439 | 61 | 95.93 | 4.06 |
| Large Soybean | 637 | 46 | 93.26 | 6.73 |
| Fitting Contact Lenses | 22 | 2 | 91.66 | 8.33 |
| NSL-KDD | 23145 | 2047 | 91.87 | 8.12 |

## 5. CONCLUSIONS

This paper presents a new algorithm for decision tree construction based on traditional machine learning algorithms, which adjusts the weights of training data based on probabilities and split the dataset into sub-dataset until all the sub-dataset belongs to the same class. The main advantage of this proposed algorithm is to set appropriate weights of training instances based on naïve Bayesian classifier before trying to construct a decision tree model. In conventional decision tree algorithm weights of every instance is set to equal value which contradicts general intuition. The experimental results proved that the proposed algorithm can achieve high classification rate on different benchmark datasets from UCI machine learning repository. The future research issues will be to know and experiment more with the unique instances we have found out and to test it extensively in real world problem domains.

## ACKNOWLEDGEMENTS

The support for this research is received from the Department of Computer Science and Engineering, United International University, Dhaka, Bangladesh.

## REFERENCES

[1] J. R. Quinlan, "Induction of Decision Tree," Machine Learning Vol. 1, 1986, pp. 81-106.

[2] J. R. Quinlan, "C4.5: Programs for Machine Learning," Morgan Kaufmann Publishers, San Mateo, CA, 1993.

[3] L. Breiman, J. H. Friedman, R. A. Olshen and C.J. Stone, "Classification and Regression Trees," Statistics probability series, Wadsworth, Belmont, 1984.

[4] John Shafer, Rakesh Agarwal, and Manish Mehta, "SPRINT: A Scalable Parallel Classifier for Data Maining," in Proceedings of the VLDB Conference, Bombay, India, September 1996.

[5] Kononenko I, "Comparison of inductive and naïve Bayesian learning approaches to automatic knowledge acquisition," in Wieling, B. (Ed), Current trend in knowledge acquisition, Amsterdam, IOS press. 1990.

[6] Langely, P., Iba, W., Thomas, and K., "An analysis of Bayesian classifier," in Proceedings of the 10th national Conference on Artificial Intelligence (San Matro, CA: AAAI press), 1992, pp. 223-228.

[7] D. Turney, "Cost-Sensitive Classification: Empirical Evaluation of a Hybrid Genetic Decision Tree Induction Algorithm," Journal of Artificial Intelligence Research, 1995, pp. 369-409.

[8] J. Bala, K. De Jong, J. Haung, H. Vafaie and H. Wechsler, "Hybrid Learning using Genetic Algorithms and Decision Trees for Pattern Classification," in Proceedings of 14th International Conference on Artificial Intelligence, 1995.

[9] C. Guerra-Salcedo, S. Chen, D. Whitley, and Stephen Smith, "Fast and Accurate Feature Selection using Hybrid Genetic Strategies," in Proceedings of the Genetic and Evolutionary Computation Conference, 1999.

[10] S. R. Safavian and D. Landgrebe, "A Survey of Decision Tree Classifier Methodology, " IEEE Transactions on Systems, Man and Cybernetics 21(3), 1991, pp. 660-674.

[11] C. Blake, E Keogh, and J. Merz, "UCI Repository of Machine Leanrning Database," Irvine, CA: University of California, Department of Information and Computer Science, 2000.

[12] The Archive UCI Machine Learning Datasets. http://archive.ics.uci.edu/ml/datasets/

## Authors

**Dr. Dewan Md. Farid** received B.Sc. in Computer Science and Engineering from Asian University of Bangladesh in 2003, M.Sc. in Computer Science and Engineering from United International University, Bangladesh in 2004, and Ph.D. in Computer Science and Engineering from Jahangirnagar University, Bangladesh in 2012. He is an Assistant Professor in the Department of Computer Science and Engineering, United International University, Bangladesh. He has published 1 book chapter, 8 international journals and 12 international conferences in the field of data mining, machine learning, and intrusion detection. He has participated and presented his papers in international conferences at Malaysia, Portugal, Italy, and France. Dr. Farid is a member of IEEE and IEEE Computer Society. He worked as a visiting researcher at ERIC Laboratory, University Lumière Lyon 2 – France from 01-09-2009 to 30-06-2010. He received Senior Fellowship I & II awarded by National Science & Information and Communication Technology (NSICT), Ministry of Science & Information and Communication Technology, Government of Bangladesh, in 2008 and 2011 respectively.

**Professor Dr. Chowdhury Mofizur Rahman** had his B.Sc. (EEE) and M.Sc. (CSE) from Bangladesh University of Engineering and Technology (BUET) in 1989 and 1992 respectively. He earned his Ph.D. from Tokyo Institute of Technology in 1996 under the auspices of Japanese Government scholarship. Prof Chowdhury is presently working as the Pro Vice Chancellor and acting treasurer of United International University (UIU), Dhaka, Bangladesh. He is also one of the founder trustees of UIU. Before joining UIU he worked as the head of Computer Science & Engineering department of Bangladesh University of Engineering & Technology which is the number one technical public university in Bangladesh. His research area covers Data Mining, Machine Learning, AI and Pattern Recognition. He is active in research activities and published around 100 technical papers in international journals and conferences. He was the Editor of IEB journal and worked as the moderator of NCC accredited centers in Bangladesh. He worked as the organizing chair and program committee member of a number of international conferences held in Bangladesh and abroad. At present he is acting as the coordinator from Bangladesh for EU sponsored eLINK project. Prof Chowdhury has been working as the external expert member for Computer Science departments of a number of renowned public and private universities in Bangladesh. He is actively contributing towards the national goal of converting the country towards Digital Bangladesh.