Independent Project Report

PREPARED BY:

Name: Chetan Bagra

MIS:112015033

Mentor: Dr. Tanmoy Hazra

Objective:

Application of decision tree in classification analysis

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, PUNE

April 2022

Introduction

We live in the age of data, where everything around us is connected to a data source, and everything in our lives is digitally recorded. For instance, the current electronic world has a wealth of various kinds of data, such as the Internet of Things (IoT) data, cybersecurity data, smart city data, business data, smartphone data, social media data, health data, COVID-19 data, and many more. The data can be structured, semi-structured, or unstructured.

Extracting insights from these data can be used to build various intelligent applications in the relevant domains. For instance, to build a data-driven automated and intelligent cybersecurity system, the relevant cybersecurity data can be used, to build personalized context-aware smart mobile applications, the relevant mobile data can be used and so on.

Machine learning (ML) have grown rapidly in recent years in the context of data analysis and computing that typically allows the applications to function in an intelligent manner. ML usually provides systems with the ability to learn and enhance from experience automatically without being specifically programmed and is generally referred to as the most popular latest technologies in the fourth industrial revolution.

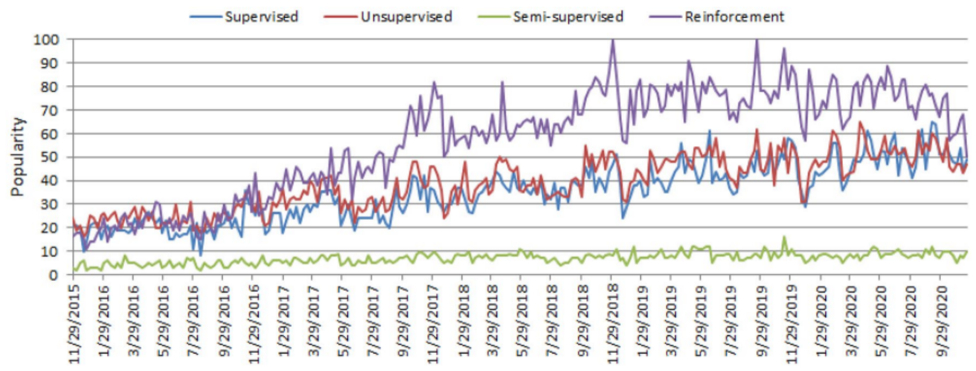


Fig. 1 The worldwide popularity score of various types of ML algorithms (supervised, unsupervised, semi-supervised, and reinforcement) in a range of 0 (min) to 100 (max) over time where x-axis represents the timestamp information and y-axis represents the corresponding score

The effectiveness and the efficiency of a machine learning solution depend on the nature and characteristics of data and the performance of the learning algorithms. In the area of machine learning algorithms, classification analysis, regression, data clustering, feature engineering and dimensionality reduction, association rule learning, or reinforcement learning techniques exist to effectively build data-driven systems.

Types of Data:

Machine learning algorithms typically consume and process data to learn the related patterns about individuals, business processes, transactions, events, and so on. Data can be of various forms, such as structured, semi-structured, or unstructured. Besides, the “metadata” is another type that typically represents data about the data.

Structured data: It has a well-defined structure, conforms to a data model following a standard order, which is highly organized and easily accessed, and used by an entity or a computer program. For instance, names, dates, addresses, credit card numbers, stock information, geolocation, etc. are examples of structured data.

Unstructured data: On the other hand, there is no pre-defined format or organization for unstructured data, making it much more difficult to capture, process, and analyse, mostly containing text and multimedia material. For example, sensor data, emails, blog entries, wikis, and word processing documents, PDF files, audio files, videos, images, presentations, web pages etc… are considered as unstructured data.

Semi-structured data: Semi-structured data are not stored in a relational database like the structured data mentioned above, but it does have certain organizational properties that make it easier to analyse. HTML, XML, JSON documents, NoSQL databases, etc., are some examples of semi-structured data.

Metadata: It is not the normal form of data, but "data about data". The primary difference between "data" and "metadata" is that data are simply the material that can classify, measure, or even document something relative to an organization's data properties. On the other hand, metadata describes the relevant data information, giving it more significance for data users.

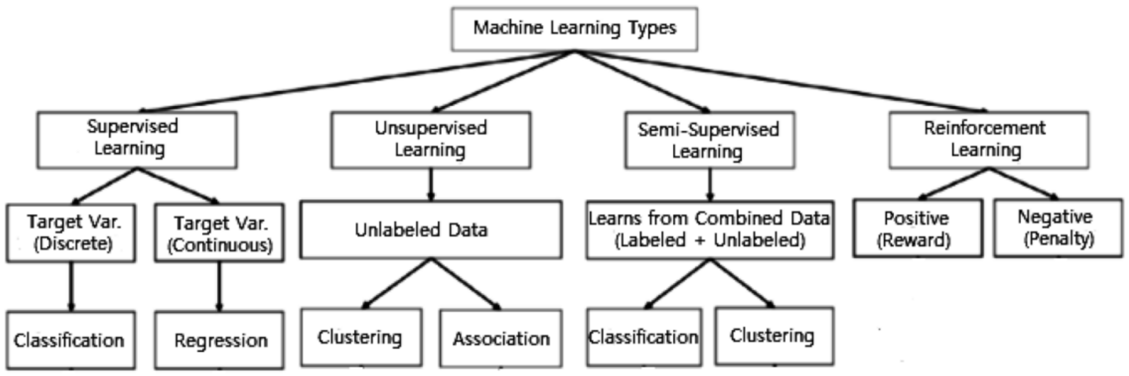
Types of Machine Learning Techniques:

Supervised: Supervised learning is typically the task of machine learning to learn a function that maps an input to an output based on sample input-output pairs. Supervised learning is carried out when certain goals are identified to be accomplished from a certain set of inputs i.e., a task driven approach. The most common supervised tasks are "classification" that separates the data, and "regression" that fts the data.

Unsupervised: Unsupervised learning analyses unlabelled datasets without the need for human interference, i.e., a data-driven process. This is widely used for extracting generative features, identifying meaningful trends and structures, groupings in results, and exploratory purposes. The most common unsupervised learning tasks are clustering, density estimation, feature learning, dimensionality reduction, finding association rules, anomaly detection, etc.

Semi-supervised: Semi-supervised learning can be defined as a hybridization of the supervised and unsupervised methods, as it operates on both labelled and unlabelled data. The ultimate goal of a semi-supervised learning model is to provide a better outcome for prediction than that produced using the labelled data alone from the model. Some application areas where semi-supervised learning is used include machine translation, fraud detection, labelling data and text classification.

Reinforcement: Reinforcement learning is a type of machine learning algorithm that enables software agents and machines to automatically evaluate the optimal behaviour in a particular context or environment to improve its efficiency i.e., an environment-driven approach. This type of learning is based on reward or penalty, and its ultimate goal is to use insights obtained from environmental activists to take action to increase the reward or minimize the risk.



Various types of machine learning techniques

Table 1 Various types of machine learning techniques with examples

Learning type	Model building	Examples
Supervised	Algorithms or models learn from labeled data (task-driven approach)	Classification, regression
Unsupervised	Algorithms or models learn from unlabeled data (Data-Driven Approach)	Clustering, associations, dimensionality reduction
Semi-supervised	Models are built using combined data (labeled + unlabeled)	Classification, clustering
Reinforcement	Models are based on reward or penalty (environment-driven approach)	Classification, control

In the further paper we will explore classification analysis using decision tree in detail.

Classification Analysis

Classification is regarded as a supervised learning method in machine learning, referring to a problem of predictive modelling as well, where a class label is predicted for a given data set. Mathematically, it maps a function (f) from input variables (X) to output variables (Y) as target, label or categories. To predict the class of given data points, it can be carried out on structured or unstructured data. For example, spam detection such as “spam” and “not spam” in email service providers can be a classification problem.

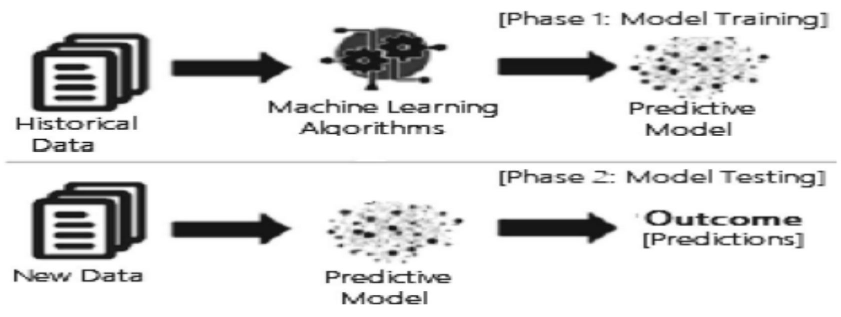


Fig. 3 A general structure of a machine learning based predictive model considering both the training and testing phase

Types of Classification:

Binary classification: It refers to the classification tasks having two class labels such as “true and false” or “yes and no”. In such binary classification tasks, one class could be the normal state, while the abnormal state could be another class. For instance, “cancer not detected” is the normal state of a task that involves a medical test, and “cancer detected” could be considered as the abnormal state.

Multiclass classification: Traditionally, this refers to those classification tasks having more than two class labels. The multiclass classification does not have the principle of normal and abnormal outcomes, unlike binary classification tasks. Instead, within a range of specified classes, examples are classified as belonging to one. For example, it can be a multiclass classification task to classify various types of network attacks in the NSL-KDD dataset, where the attack categories are classified into four class labels, such as DoS (Denial of Service Attack), U2R (User to Root Attack), R2L (Root to Local Attack), and Probing Attack.

Multi-label classification: In machine learning, multilabel classification is an important consideration where an example is associated with several classes or labels. Thus, it is a generalization of multiclass classification, where the classes involved in the problem are hierarchically structured, and each example may simultaneously belong to more than one class in each hierarchical level. For instance, Google news can be presented under the categories of a “city name”, “technology”, or “latest news”, etc.

Many classification methods have been proposed. Some of the most common and popular methods that are used in various applications are Naïve Bayes, Linear Discriminant Analysis (LDA), Logistic regression (LR), K-nearest neighbours (KNN), Support vector machine (SVM), Decision tree (DT), Random Forest (RF), Adaptive Boosting (AdaBoost), Extreme gradient boosting (XGBoost), Stochastic gradient descent (SGD) and Rule-based classification.

In this paper we will discuss Decision Tree in detail.

Decision Tree

A decision tree is a support tool with a tree-like structure that models probable outcomes, cost of resources, utilities and possible consequences. Decision trees provide a way to present algorithms with conditional control statements. They include branches that represents decision making steps that can lead to a favourable result. By sorting down the tree from the root to some leaf nodes, as shown in Fig. 4, DT classifies the instances. Instances are classified by checking the attribute defined by that node, starting at the root node of the tree, and then moving down the tree branch corresponding to the attribute value.

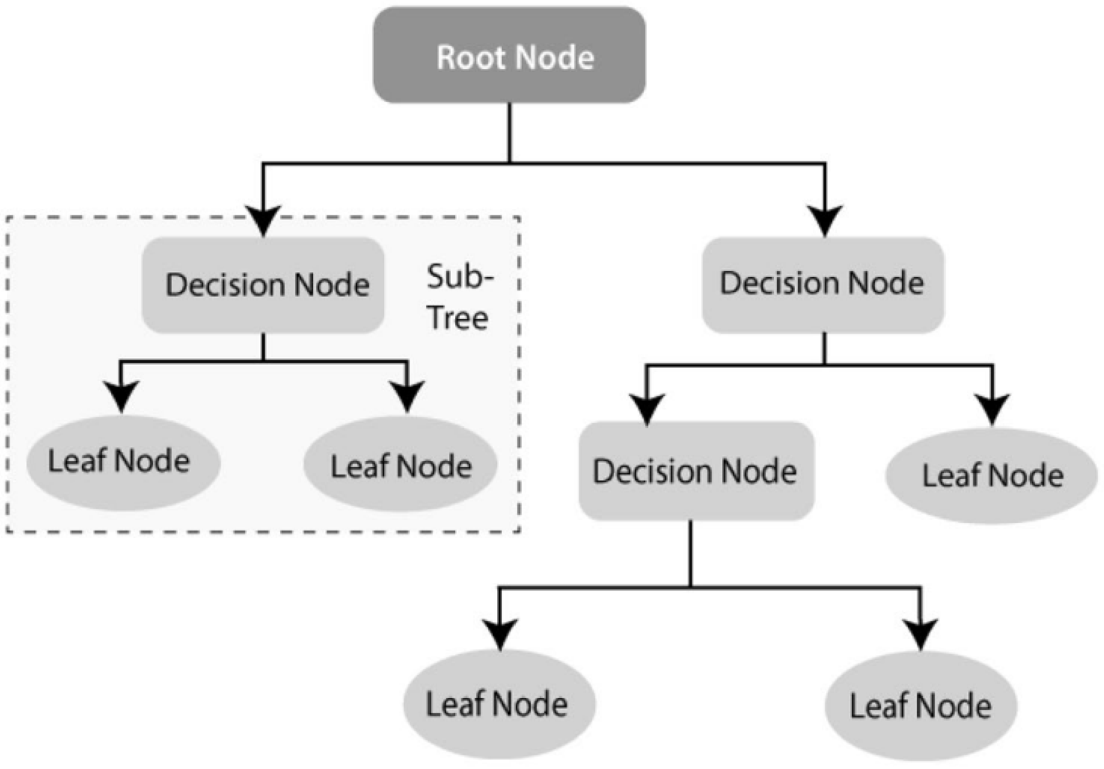


Fig. 4 An example of a decision tree structure

The Induction task

The induction task is to develop a classification rule that can determine the class of any object from its values of the attributes. The immediate question is whether or not the attributes provide sufficient information to do this. In particular, if the training set contains two objects that have identical values for each attribute and yet belong to different classes, it is clearly impossible to differentiate between these objects with reference only to the given attributes.

In such a case attributes will be termed inadequate for the training set and hence for the induction task. The basis is a universe of objects that are described in terms of a collection of attributes. Each attribute measures some important feature of an object and will be limited here to taking a (usually small) set of discrete, mutually exclusive values. For example, if the objects were Saturday mornings and the classification task involved the weather, attributes might be

outlook, with values {sunny, overcast, rain]

temperature, with values {cool, mild, hot]

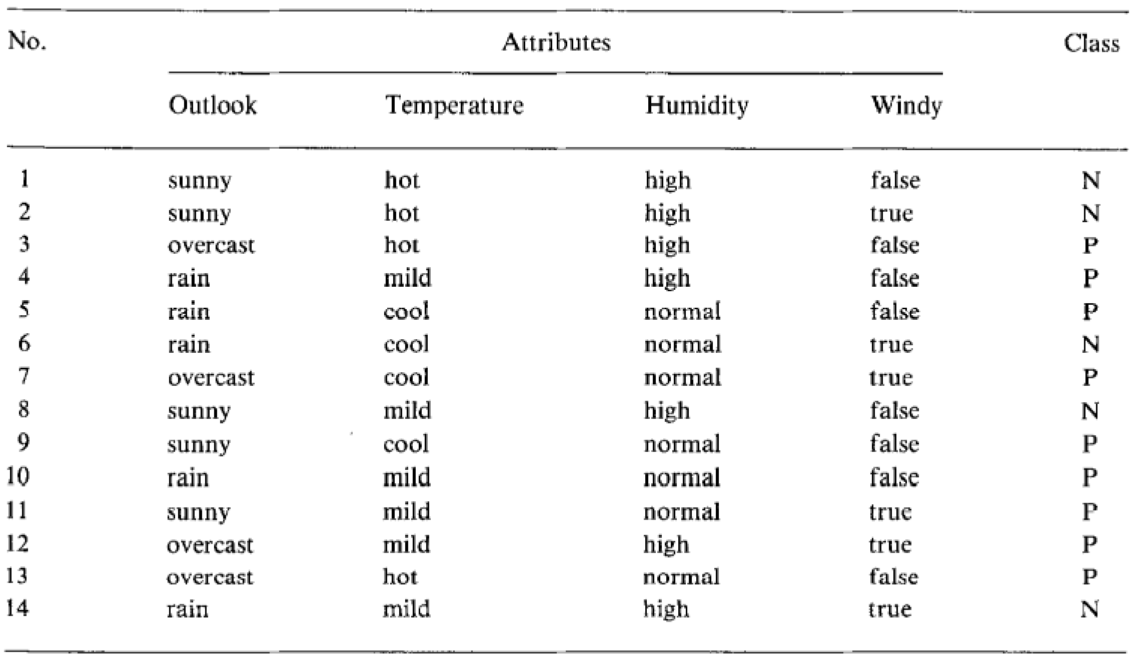
humidity, with values {high, normal]

windy, with values { true, false]

Taken together, the attributes provide a zeroth-order language for characterizing objects in the universe. A particular Saturday morning might be described as

outlook: overcast, temperature: cool, humidity: normal, windy: false

Table 1. A small training set



No.	Attributes				Class
	Outlook	Temperature	Humidity	Windy	
1	sunny	hot	high	false	N
2	sunny	hot	high	true	N
3	overcast	hot	high	false	P
4	rain	mild	high	false	P
5	rain	cool	normal	false	P
6	rain	cool	normal	true	N
7	overcast	cool	normal	true	P
8	sunny	mild	high	false	N
9	sunny	cool	normal	false	P
10	rain	mild	normal	false	P
11	sunny	mild	normal	true	P
12	overcast	mild	high	true	P
13	overcast	hot	normal	false	P
14	rain	mild	high	true	N

Leaves of a decision tree are class names, other nodes represent attribute-based tests with a branch for each possible outcome. In order to classify an object, we start at the root of the tree, evaluate the test, and take the branch appropriate to the outcome. The process continues until a leaf is encountered, at which time the object is asserted to belong to the class named by the leaf.

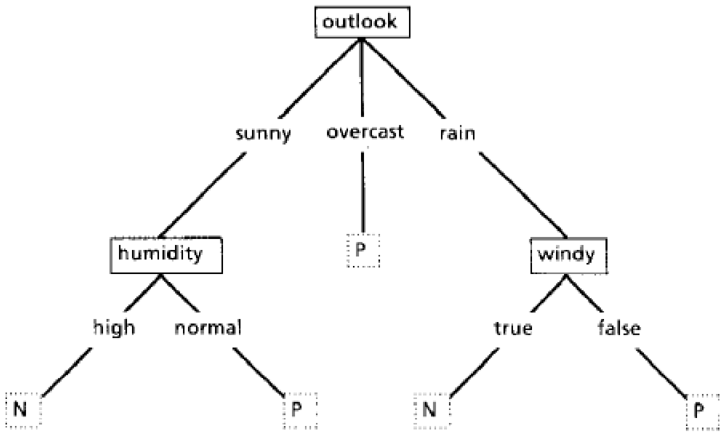


Figure 2. A simple decision tree

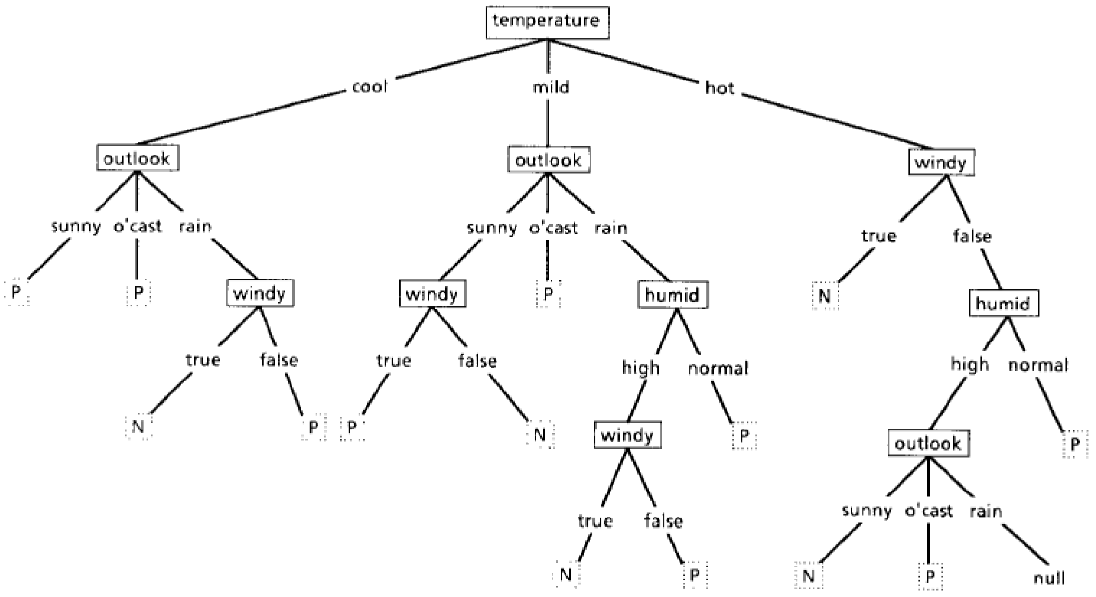


Figure 3. A complex decision tree.

If the attributes are adequate, it is always possible to construct a decision tree that correctly classifies each object in the training set, and usually there are many such correct decision trees. The essence of induction is to move beyond the training set, i.e., to construct a decision tree that correctly classifies not only objects from the training set but other (unseen) objects as well. In order to do this, the decision tree must capture some meaningful relationship between an object's class and its values of the attributes.

Given a choice between two decision trees, each of which is correct over the training set, it seems sensible to prefer the simpler one on the grounds that it is more likely to capture structure inherent in the problem. The simpler tree would therefore be expected to classify correctly more objects outside the training set. The decision tree of Figure 3, for instance, is also correct for the training set of Table 1, but its greater complexity makes it suspect as an 'explanation' of the training set.

ID3 Algorithm

As we know that there are many possible decision trees for a given data set that correctly classify the data. One approach would be to draw all possible tree and select the simplest one that classify the given data set. This approach is feasible for a small date set. ID3 was designed for the other end of the spectrum, where there are many attributes and the training set contains many objects, but where a reasonably good decision tree is required without much computation. It has generally been found to construct simple decision trees, but the approach it uses cannot guarantee that better trees have not been overlooked.

The basic structure of ID3 is iterative. A subset of the training set called the window is chosen at random and a decision tree formed from it; this tree correctly classifies all objects in the window.

All other objects in the training set are then classified using the tree. If the tree gives the correct answer for all these objects, then it is correct for the entire training set and the process terminates. If not, a selection of the incorrectly classified objects is added to the window and the process continues.

The crux of the problem is how to form a decision tree for an arbitrary collection object.

We introduce concepts of information entropy and information gain that are used to construct a decision tree using ID3 algorithm.

Information entropy

Information entropy of the given data measures the least amount of the information necessary to represent a data item from given data. The lower the information entropy, the more regular the data is, the more pattern occurs in the data and thus less amount of the information is necessary to represent it.

Suppose that we are given a probability space S with the elements 1, 2, ..., n. The probability an element "I" would be chosen from the probability space is P_i. Then the information entropy of the probability space is defined as:

$E(S)= -P_1*\log(P_1)-P_2*\log(P_2)-\ldots-P_n*\log(P_n)$ where $\log_2$ is a binary logarithm

So, the information entropy of the probability space of unbiased coin throws is:

$E = -0.5 * \log2 (0.5) - 0.5*\log2 (0.5) =0.5+0.5=1$

When the coin is based with 25% chance of a head and 75% change of a tail, then the information entropy of such space is:

E = -0.25 * log2 (0.25) - 0.75*log2 (0.75) = 0.81127812445

which is less than 1. Thus, for example, if we had a large file with about 25% of 0 bits and 75% of 1 bits, a good compression tool should be able to compress it down to about 81.12% of its size.

Information gain

The information gain is the amount of the information entropy gained as a result of a certain procedure.

We may also gain the information entropy by dividing the whole set S into sets, grouping them by a similar pattern. If we group elements by their value of an attribute A, then we define the information gain as:

$$IG(S,A) = E(S) - \sum_{v \in \text{values}(A)} \left[\frac{|S_v|}{|S|} * E(S_v) \right]$$

where S_v is a set with the elements of S that have the value v for the attribute A.

The ID3 algorithm partitions the set S according to the attribute that yields the highest information gain.

Let us build a decision tree for data given below:

Temperature	**Wind**	**Sunshine**	**Play**
Cold	Strong	Cloudy	No
Warm	Strong	Cloudy	No
Warm	None	Sunny	Yes
Hot	None	Sunny	No
Hot	Breeze	Cloudy	Yes
Warm	Breeze	Sunny	Yes
Cold	Breeze	Cloudy	No
Cold	None	Sunny	Yes
Hot	Strong	Cloudy	Yes
Warm	None	Cloudy	Yes

S= {(Cold, Strong, Cloudy, No),(Warm, Strong,Cloudy,No),(Warm,None,Sunny,Yes), (Hot,None,Sunny,No),(Hot,Breeze,Cloudy,Yes),(Warm,Breeze,Sunny,Yes),(Cold,B reeze,Cloudy,No),(Cold,None,Sunny,Yes),(Hot,Strong,Cloudy,Yes),(Warm,None,Cloudy,Ye s)}

E(S)=-(4/10) *log2 (4/10) -(6/10) *log2 (6/10) =0.97095059445

$E(S_{cold})$=-(2/3) *log2 (2/3) -(1/3) *log2 (1/3) =0.91829583405

$E(S_{warm})$=-(1/4) *log2 (1/4) -(3/4) *log2 (3/4) =0.81127812445

$E(S_{hot})$=-(1/3) *log2 (1/3) -(2/3) *log2 (2/3) =0.91829583405

Thus:

IG(S,temperature)=E(S)-[(|Scold|/|S|)*E(Scold)+(|Swarm|/|S|)*E(Swarm)+(|Shot|/|S|)*E(Shot)]

=0.97095059445-[(3/10)*0.91829583405+(4/10)*0.81127812445+(3/10)*0.91829583405]

=0.09546184424

$E(S_{none})$=0.81127812445

$E(S_{breeze})$=0.91829583405

$E(S_{strong})$=0.91829583405

Thus,

IG (S, wind)=E(S)-[(|Snone|/|S|)*E(Snone)+(|Sbreeze|/|S|)*E(Sbreeze)+(|Sstrong|/|S|)*E(Sstrong)] = 0.97095059445-[(4/10)*0.81127812445+(3/10)*0.91829583405+(3/10)*0.91829583405]

= 0.09546184424

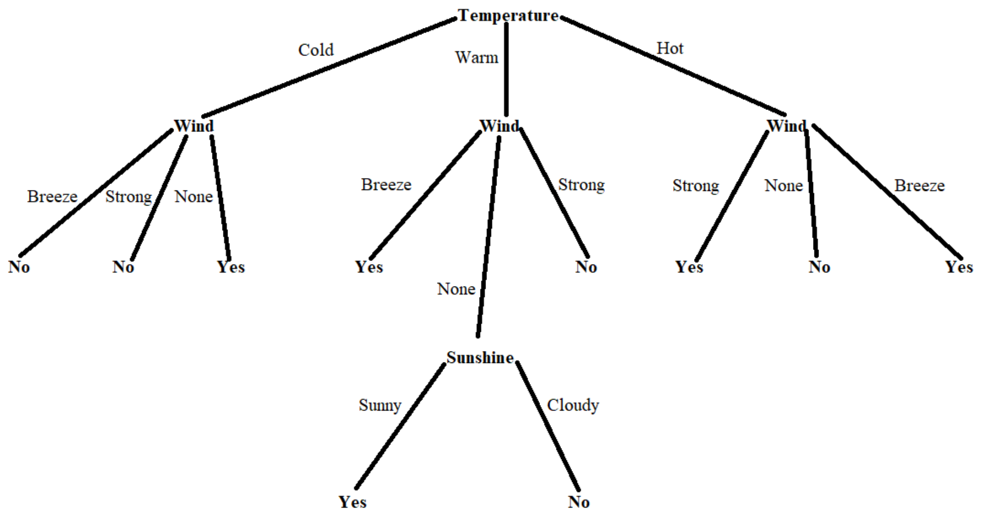
$E(S_{cloudy})$=1

$E(S_{sunny})$=0.81127812445

Thus,

IG(S,sunshine)=E(S)-[(|Scloudy|/|S|)*E(Scloudy)+(|Ssunny|/|S|)*E(Ssunny)] =0.97095059445-[(6/10)*1+(4/10)*0.81127812445]=0.04643934467

IG(S,wind) and IG(S,temperature) are greater than IG(S,sunshine). Both of them are equal; therefore, we can choose any of the attribute to form the three branches. At those branches, we would apply the algorithm further to form the rest of the decision tree.



Following decision tree is generated from the data set given above. Now if object comes with attributes as (warm, Strong, Sunny,?) then with help of decision tree we can classify that we cannot play the match.

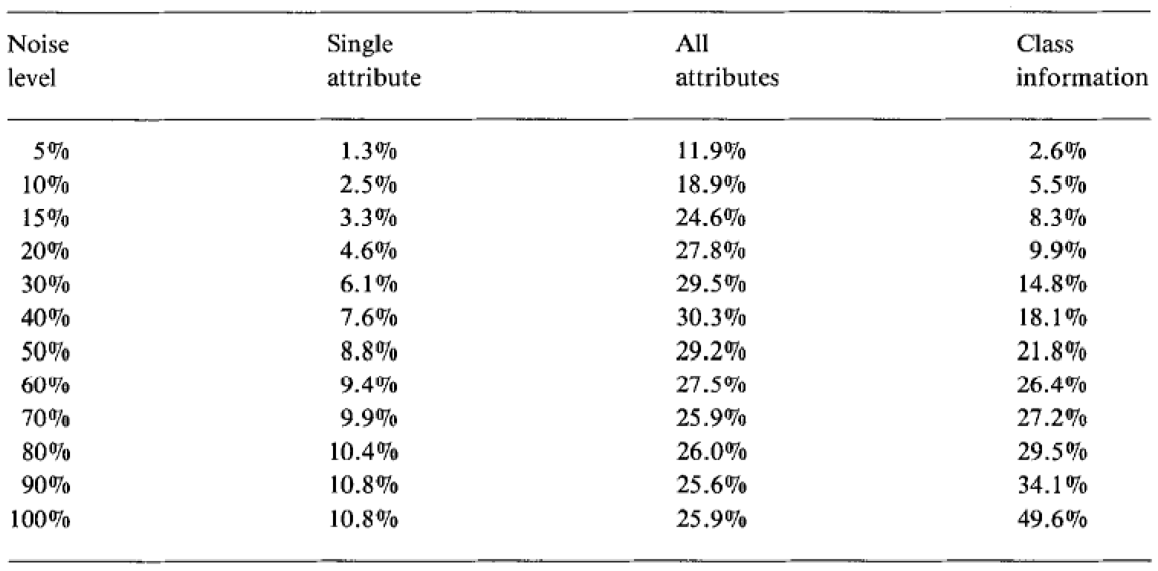
ID3's total computational requirement per iteration is proportional to the product of the size of the training set, the number of attributes and the number of non-leaf nodes in the decision tree.

No exponential growth in time or space has been observed as the dimensions of the induction task increase, so the technique can be applied to large tasks.

Gaps Found in ID3 Algorithm

- Errors in the training set may cause the attributes to become inadequate or may lead to decision trees of spurious complexity.

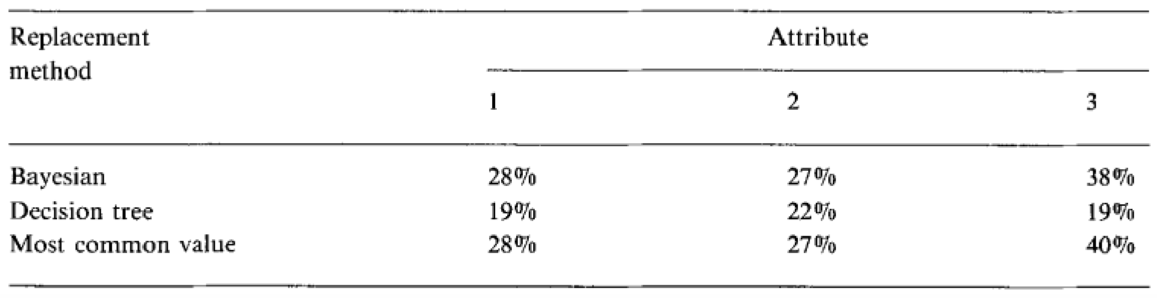
Table 2. Error rates produced by noise in a single attribute, all attributes, and class information



Noise level	Single attribute	All attributes	Class information
5%	1.3%	11.9%	2.6%
10%	2.5%	18.9%	5.5%
15%	3.3%	24.6%	8.3%
20%	4.6%	27.8%	9.9%
30%	6.1%	29.5%	14.8%
40%	7.6%	30.3%	18.1%
50%	8.8%	29.2%	21.8%
60%	9.4%	27.5%	26.4%
70%	9.9%	25.9%	27.2%
80%	10.4%	26.0%	29.5%
90%	10.8%	25.6%	34.1%
100%	10.8%	25.9%	49.6%

- Unknown attribute values.
-

Table 3. Proportion of times that an unknown attribute value is replaced by an incorrect value



Replacement method	Attribute		
	1	2	3
Bayesian	28%	27%	38%
Decision tree	19%	22%	19%
Most common value	28%	27%	40%

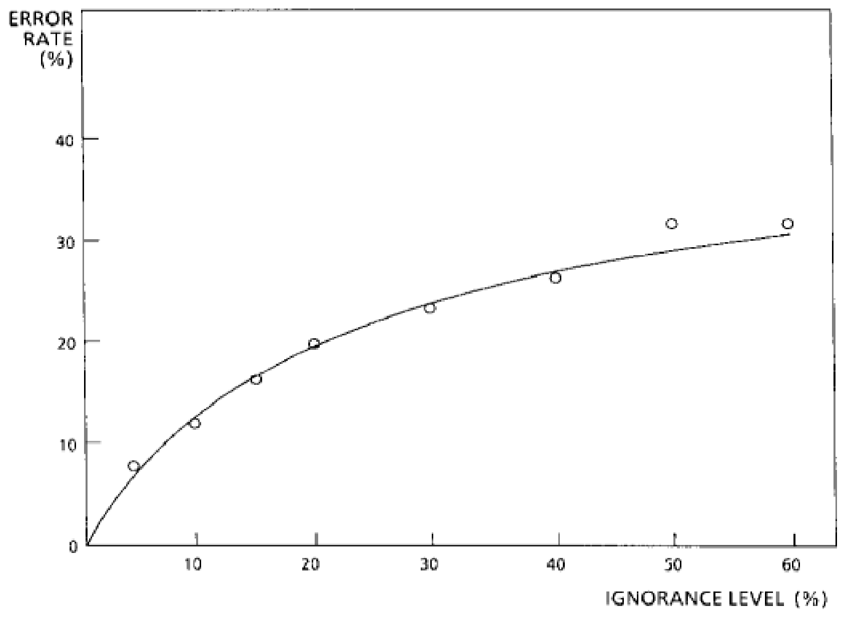


Figure 5. Error produced by unknown attribute values.

-
- Attention has recently been refocussed on the evaluation function for selecting the" best attribute-based test to form the root of a decision tree. Recall that the criterion described earlier chooses the attribute that gains most information.
- Using information gain for feature selection, the algorithm tends to select attributes with more values, which is due to the fact that the value of the information gain of this kind of attribute will be bigger than others.
- In the decision tree building process, it is difficult to control the tree size. However, most researchers have tried to improve on this using various pruning methods to avoid the occurrence of over-fitting, which has led to the decision tree building process to be completed in two steps, that is modelling and pruning. Meanwhile, it will save a lot of time if a concise decision tree is built in one-step.
- There are several logarithmic calculations in the attribute selection process, this has made the computation of information gain time consuming.
- ID3 only work with discrete data, it does not work with continuous data.

C4.5 Algorithm

C4.5 algorithm is the extension of the ID3 algorithm. It was developed to overcome some of the limitations of the ID3.

The IGR(Information gain ratio) which is used in C4.5 algorithm is less biased selection criteria. It takes information gain and normalize it with split entropy.

It creates a possibility to use continuous data.

Pruning of the tree after creation.

$$GainRatio(p,T) = \frac{Gain(p,T)}{SplitInfo(p,T)}$$

where **SplitInfo** is:

$$SplitInfo(p,test) = -\sum_{j=1}^{n} P'\left(\frac{j}{p}\right) \times \log\left(P'\left(\frac{j}{p}\right)\right)$$

P' (j/p) is the proportion of elements present at the position p, taking the value of j_{th} test.

C4.5 algorithm also deals with the continuous data.

Let's consider an example to build decision tree with C4.5 algorithm:

TABLE II. DATA SET S

Day	Outlook	Temperature	Humidity	Wind	Play
D1	Sun	Hot	**85**	Low	No
D2	Sun	Hot	**90**	High	No
D3	Overcast	Hot	**78**	Low	Yes
D4	Rain	Sweet	**96**	Low	Yes
D5	Rain	Cold	**80**	Low	Yes
D6	Rain	Cold	**70**	High	No
D7	Overcast	Cold	**65**	High	Yes
D8	Sun	Sweet	**95**	Low	No
D9	Sun	Cold	**70**	Low	Yes
D10	Rain	Sweet	**80**	Low	Yes
D11	Sun	Sweet	**70**	High	Yes
D12	Overcast	Sweet	**90**	High	Yes
D13	Overcast	Hot	**75**	Low	Yes
D14	Rain	Sweet	**80**	High	No

First Three Steps are similar to the ID3

We can find split_info as follow:

Splitinfo(outlook)=-(4/14)*log(4/14)-(5/14)*log(5/14)-(5/14)*log(5/14)

Interesting part is to find Gain for attribute Humidity.

Gain(S,Humidity)=?

We must now sort the attribute values in ascending order, the set of values is as follows:

{65, 70, 70, 70, 75, 78, 80, 80, 80, 85, 90, 90, 95, 96}

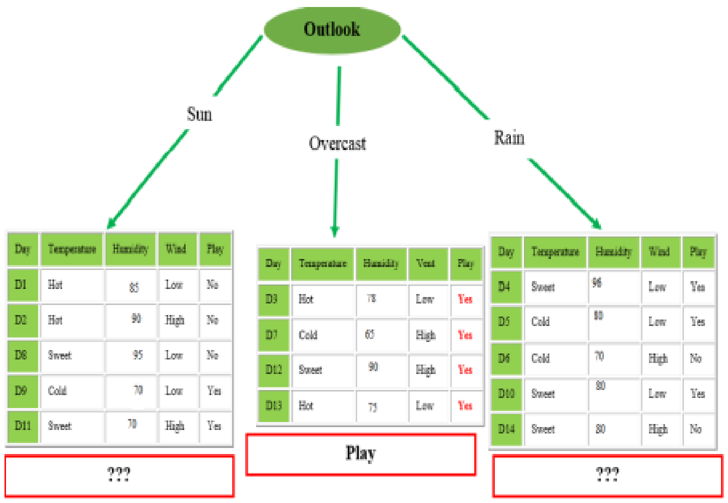
we will remove values that are repeated:

{65, 70, 75, 78, 80, 85, 90, 95, 96}

TABLE III. GAIN CALCULATION FOR THE ATTRIBUTE CONTINUOUS HUMIDITY USING C4.5 ALGORITHM

	65		70		75		78		80		85		90		95		96	
interval	≤	>	≤	>	≤	>	≤	>	≤	>	≤	>	≤	>	≤	>	≤	>
Yes	1	8	3	6	4	5	5	4	7	2	7	2	8	1	8	1	9	0
No	0	5	1	4	1	4	1	4	2	3	3	2	4	1	5	0	5	0
Entropy	0	0.961	0.811	0.971	0.721	0.991	0.65	1	0.764	0.971	0.881	1	0.918	1	0.961	0	0.94	0
Info(S,T)	0.892		0.925		0.8950		0.85		0.838		0.915		0.929		0.892		0.94	
Gain	0.048		0.015		0.045		0.09		0.102		0.025		0.011		0.048		0	

Gain (S, Humidity) = 0.102



Day	Temperature	Humidity	Wind	Play
D1	Hot	85	Low	No
D2	Hot	90	High	No
D8	Sweet	95	Low	No
D9	Cold	70	Low	Yes
D11	Sweet	70	High	Yes

Day	Temperature	Humidity	Vent	Play
D3	Hot	78	Low	Yes
D7	Cold	65	High	Yes
D12	Sweet	90	High	Yes
D13	Hot	75	Low	Yes

Day	Temperature	Humidity	Wind	Play
D4	Sweet	96	Low	Yes
D5	Cold	80	Low	Yes
D6	Cold	70	High	No
D10	Sweet	80	Low	Yes
D14	Sweet	80	High	No

Fig. 4. Root node of the C4.5 decision tree

Similar doing the work for the branches created we can create the whole decision tree.

Performance analysis of ID3 and C4.5

Size of Data Set	Algorithm	
	ID3 (%)	C4.5 (%)
14	94.15	96.2
24	78.47	83.52
35	82.2	84.12

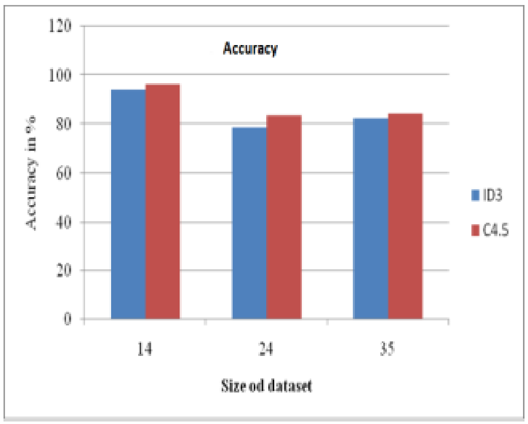


Fig. 6. Comparison of Accuracy for ID3 & C4.5 Algorithm

Size of Data Set	Algorithm	
	ID3 (%)	C4.5 (%)
14	0.215	0.0015
24	0.32	0.17
35	0.39	0.23

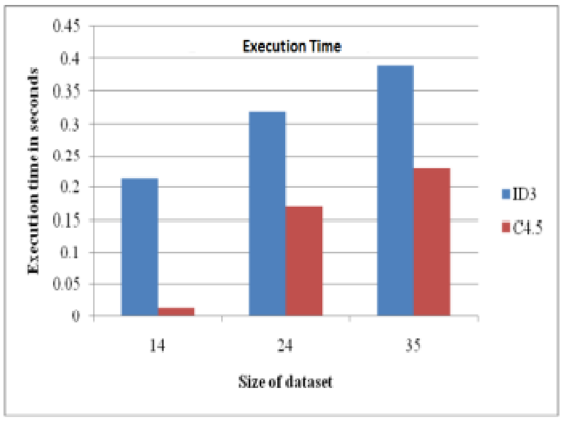


Fig. 7. Comparison of Execution Time for ID3 & C4.5 Algorithm

Gaps in C4.5 algorithm

- The pruning strategy can be optimized.
- C4.5 uses a polytree, which is more efficient with a binary tree.
- C4.5 can only be used for classification.
- The entropy model used in C4.5 has a lot of time-consuming logarithmic operations, continuous values and sorting operations.
- C4.5 In the process of constructing the tree, the numerical attribute values need to be sorted according to their size, and a split point is selected from it, so it is only suitable for the data set that can reside in the memory, when the training set is too large to fit in the memory, The program cannot run.

Application of decision tree

- Decision tree are widely used in predictive analysis and intelligent decision-making.
- Identifying criminals after crime has committed and card fraud as it happens.
- It assists retailers in better understanding consumer preferences and behaviour, better manage inventory, avoiding out-of-stock situations, and optimizing logistics and warehousing in e-commerce.

Conclusion

The aim of this paper has been to demonstrate that the technology for building decision trees from examples is fairly robust. Current commercial systems are powerful tools that have achieved noteworthy successes. The groundwork has been done for advances that will permit such tools to deal even with noisy, incomplete data typical of advanced real-world applications. Work is continuing at several canters to improve the performance of the underlying algorithms.

Consider, however, the task of developing a rule from a given set of examples for classifying an animal as a monkey, giraffe, elephant, horse, etc. A single decision tree could be produced in which these various classes appeared as leaves.

An alternative approach taken by systems would produce a collection of classification rules, one to discriminate monkeys from non-monkeys, another to discriminate giraffes from non-giraffes, and so on.

The multi-tree approach has some associated problems - the separate decision trees may classify an animal as both a monkey and a giraffe, or fail to classify it as anything, for example - but if these can be sorted out, this approach may lead to techniques for building more reliable decision trees.

Literature Survey Reference

- ❖ Referance-1
- ❖ Referance-2
- ❖ Referance-3
- ❖ Referance-4