**Delhi Technological University**

Ethical Hacking

MTE PROJECT REPORT

MALWARE DETECTION USING MACHINE & DEEP LEARNING



Delhi Technological University

Bawana Road, Delhi-110042

**Under the Guidance of: -**

Mr. ASHISH BAJAJ SIR

**Submitted By: -**

Kabir Koli - 2K19/IT/064

**INTRODUCTION**

**Malware** (a [portmanteau](https://en.wikipedia.org/wiki/Portmanteau) for **malicious software**) is any [software](https://en.wikipedia.org/wiki/Software) intentionally designed to cause damage to a [computer](https://en.wikipedia.org/wiki/Computer), [server](https://en.wikipedia.org/wiki/Server_(computing)), [client](https://en.wikipedia.org/wiki/Client_(computing)), or [computer network](https://en.wikipedia.org/wiki/Computer_network). By contrast, software that causes unintentional harm due to some deficiency is typically described as a [software bug](https://en.wikipedia.org/wiki/Software_bug). A wide variety of malware types exist, including [computer viruses](https://en.wikipedia.org/wiki/Computer_virus), [worms](https://en.wikipedia.org/wiki/Computer_worm), [Trojan horses](https://en.wikipedia.org/wiki/Trojan_horse_(computing)), [ransomware](https://en.wikipedia.org/wiki/Ransomware), [spyware](https://en.wikipedia.org/wiki/Spyware), [adware](https://en.wikipedia.org/wiki/Adware), [rogue software](https://en.wikipedia.org/wiki/Rogue_software), [wiper](https://en.wikipedia.org/wiki/Wiper_(malware)) and [scareware](https://en.wikipedia.org/wiki/Scareware).

Programs are also considered malware if they secretly act against the interests of the computer user. For example, at one point, [Sony BMG](https://en.wikipedia.org/wiki/Sony_BMG) compact discs [silently installed a rootkit](https://en.wikipedia.org/wiki/Sony_BMG_copy_protection_rootkit_scandal) on purchasers' computers with the intention of preventing illicit copying, but which also reported on users' listening habits, and unintentionally created extra security vulnerabilities.

A range of [antivirus software](https://en.wikipedia.org/wiki/Antivirus_software), [firewalls](https://en.wikipedia.org/wiki/Firewall_(computing)) and other strategies are used to help protect against the introduction of malware, to help detect it if it is already present, and to recover from malware-associated malicious activity and attacks.

Research shows that over the last decade, malware have been growing exponentially, causing substantial financial losses to various organizations. Different anti-malware companies have been proposing solutions to defend attacks from these malwares. The velocity, volume, and the complexity of malware are posing new challenges to the anti-malware community. Current state-of-the-art research shows that recently, researchers and anti-virus organizations started applying machine learning and deep learning methods for malware analysis and detection. The focus of this is to present our work on detecting malware with (1) various machine learning algorithms and (2) deep learning models. Our results show that the Random Forest outperforms Deep Neural Network with

In the digital age, malware have impacted a large number of computing devices. The term malware come from malicious software which are designed to meet the harmful intent of a malicious attacker. Malware can compromise computers/ smart devices, steal confidential information, penetrate networks, and cripple critical infrastructures, etc. These programs include viruses, worms, trojans, spyware, bots, rootkits, ransomware, etc. According to Computer Economics1, financial loss due to malware attack has grown quadruple from $3.3 billion in 1997 to $13.3 billion in 2006. Every few years the definition of Year of Mega Breach has to be recalibrated based on attacks performed in that particular year. Recently in 2016, WannaCry ransomware attack crippled the computers of more than 150 countries, doing financial damage to different organizations. In 2016, Cybersecurity Ventures estimated the total damage due to malware attacks was $3 trillion in 2015 and is expected to reach $6 trillion by 2021.

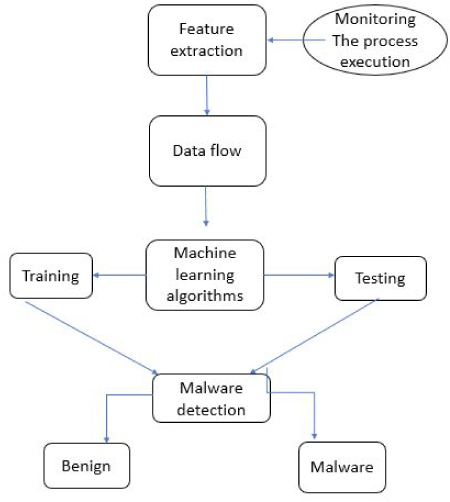
Antivirus software (such as Norton, McAfee, Avast, Kaspersky, AVG, Bitdefender, etc.) is a major line of defence for malware attacks. Traditionally, an antivirus software used the signature-based method for malware detection. Signature is a short sequence of bytes which can be used to identify known malware. But the signature-based detection system cannot provide security against zero-day attacks. Also, malware generation toolkits like Zeus can generate thousands of variants of the same malware by using different obfuscation techniques. Signature generation is often a human-driven process which is bound to become infeasible with the current malware growth.

**Experimental Setup**

The proposed approach is a multi-step process consisting of various phases performing several tasks: collection of the dataset, disassembling of executable files, feature extraction, dimension reduction, building classification models, and empirical analysis of the results based on different metrics.

**Dataset & Disassembling of Malicious and Benign Executables**

For this project I have used the dataset from the GitHub, and declared it as the malData in the project. The shape of the dataset is 138047 by 57. There are legit files and malware files, there are 41323 legit files and 96735 malware files. Then I have cleaned the dataset i.e. removing the unwanted columns.



Splitting the dataset into training and testing will help us compare accuracy of our models that we have made. We have a dataset and that dataset is given to the machine learning algorithms then that dataset is then split into training and testing dataset. Then our algorithms will give us the result and tell us the how many files are actually malware and are legit, then we will compare that result to find the accuracy of our model and then compare the accuracy of all model to find the best suitable model.

Models (Random Forest, Logistic Regression)

1. Random Forest

**What is a random forest?**

A random forest is a machine learning technique that’s used to solve regression and classification problems. It utilizes ensemble learning, which is a technique that combines many classifiers to provide solutions to complex problems.

A random forest algorithm consists of many decision trees. The ‘forest’ generated by the random forest algorithm is trained through bagging or bootstrap aggregating. Bagging is an ensemble meta-algorithm that improves the accuracy of machine learning algorithms.

The (random forest) algorithm establishes the outcome based on the predictions of the decision trees. It predicts by taking the average or mean of the output from various trees. Increasing the number of trees increases the precision of the outcome.

A random forest eradicates the limitations of a decision tree algorithm. It reduces the overfitting of datasets and increases precision. It generates predictions without requiring many configurations in packages (like scikit-learn).

**Features of a Random Forest Algorithm**

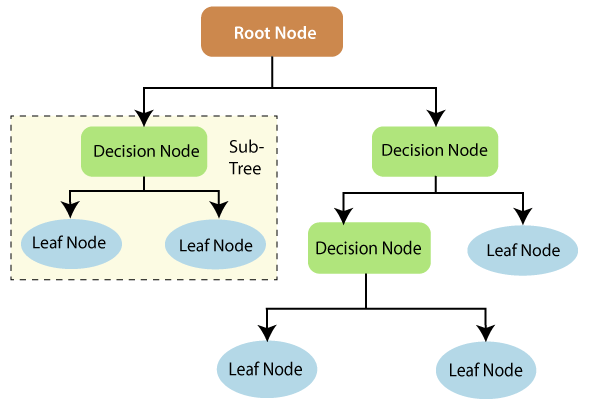
* It’s more accurate than the decision tree algorithm.
* It provides an effective way of handling missing data.
* It can produce a reasonable prediction without hyper-parameter tuning.
* It solves the issue of overfitting in decision trees.
* In every random forest tree, a subset of features is selected randomly at the node’s splitting point.

**How random forest algorithm works**

Decision trees are the building blocks of a random forest algorithm. A decision tree is a decision support technique that forms a tree-like structure. An overview of decision trees will help us understand how random forest algorithms work.

A decision tree consists of three components: decision nodes, leaf nodes, and a root node. A decision tree algorithm divides a training dataset into branches, which further segregate into other branches. This sequence continues until a leaf node is attained. The leaf node cannot be segregated further.

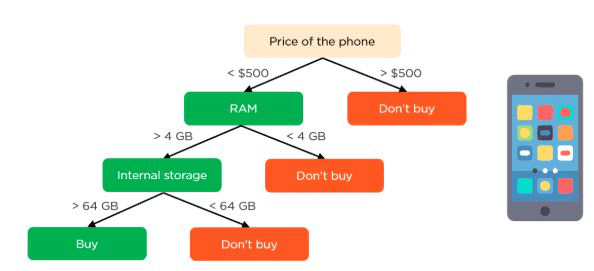
The nodes in the decision tree represent attributes that are used for predicting the outcome. Decision nodes provide a link to the leaves. The following diagram shows the three types of nodes in a decision tree.



The information theory can provide more information on how decision trees work. Entropy and information gain are the building blocks of decision trees. An overview of these fundamental concepts will improve our understanding of how decision trees are built.

Entropy is a metric for calculating uncertainty. Information gain is a measure of how uncertainty in the target variable is reduced, given a set of independent variables.

The information gain concept involves using independent variables (features) to gain information about a target variable (class). The entropy of the target variable (Y) and the [conditional entropy](https://en.wikipedia.org/wiki/Conditional_entropy) of Y (given X) are used to estimate the information gain. In this case, the conditional entropy is subtracted from the entropy of Y.

Information gain is used in the training of decision trees. It helps in reducing uncertainty in these trees. A high information gain means that a high degree of uncertainty (information entropy) has been removed. Entropy and information gain are important in splitting branches, which is an important activity in the construction of decision trees. 

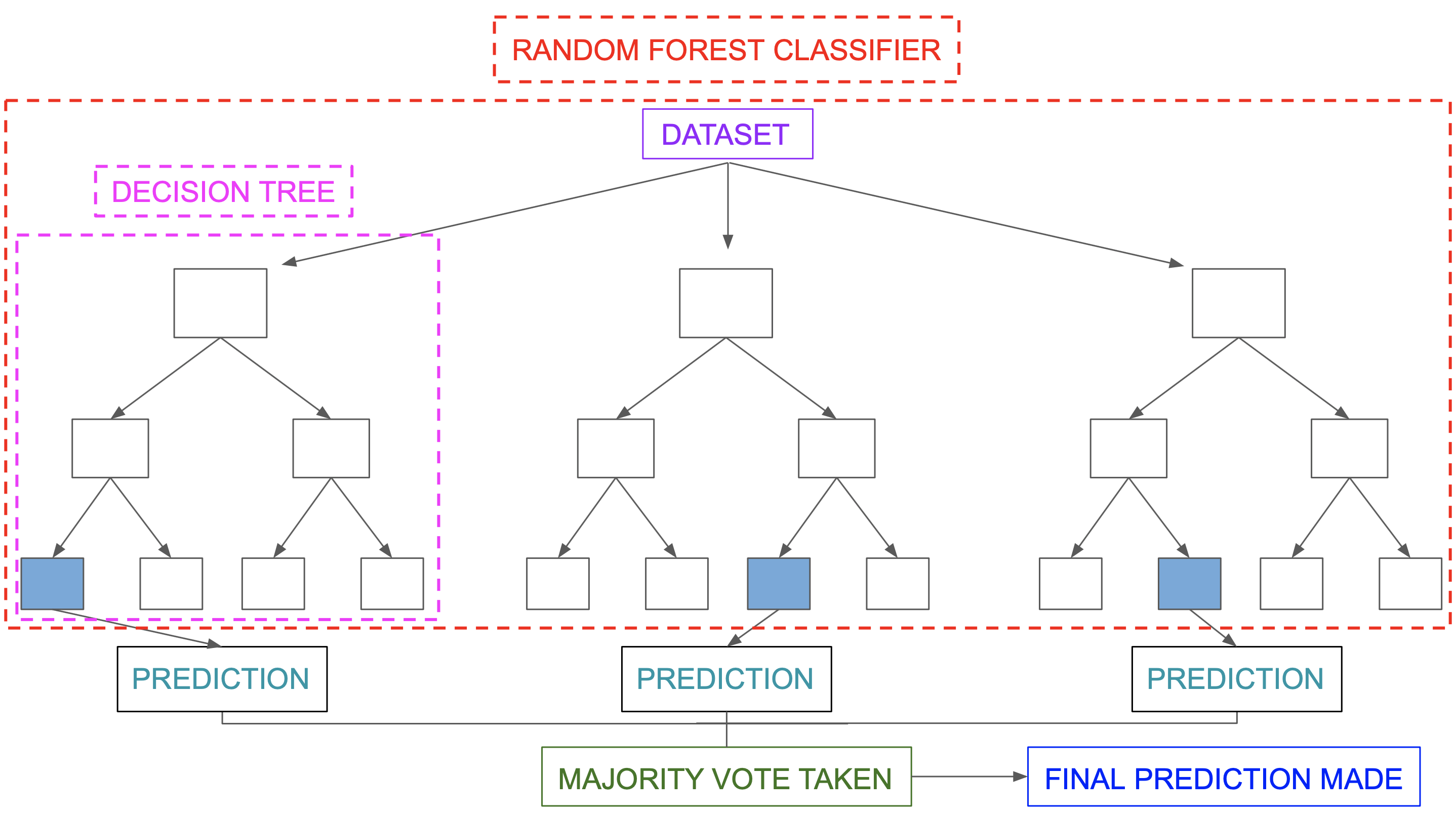
Let’s take a simple example of how a decision tree works. Suppose we want to predict if a customer will purchase a mobile phone or not. The features of the phone form the basis of his decision. This analysis can be presented in a decision tree diagram.

The root node and decision nodes of the decision represent the features of the phone mentioned above. The leaf node represents the final output, either buying or not buying. The main features that determine the choice include the price, internal storage, and Random-Access Memory (RAM). The decision tree will appear as follows.

**Classification in random forests**

Classification in random forests employs an ensemble methodology to attain the outcome. The training data is fed to train various decision trees. This dataset consists of observations and features that will be selected randomly during the splitting of nodes.

A rain forest system relies on various decision trees. Every decision tree consists of decision nodes, leaf nodes, and a root node. The leaf node of each tree is the final output produced by that specific decision tree. The selection of the final output follows the majority-voting system. In this case, the output chosen by the majority of the decision trees becomes the final output of the rain forest system. The diagram below shows a simple random forest classifier.



Applications of random forest

Some of the applications of the random forest may include:

Banking

Random forest is used in banking to predict the creditworthiness of a loan applicant. This helps the lending institution make a good decision on whether to give the customer the loan or not. Banks also use the random forest algorithm to detect fraudsters.

Health care

Health professionals use random forest systems to diagnose patients. Patients are diagnosed by assessing their previous medical history. Past medical records are reviewed to establish the right dosage for the patients.

E-commerce

Through rain forest algorithms, e-commerce vendors can predict the preference of customers based on past consumption behaviour.

**Advantages of random forest**

* It can perform both regression and classification tasks.
* A random forest produces good predictions that can be understood easily.
* It can handle large datasets efficiently.
* The random forest algorithm provides a higher level of accuracy in predicting outcomes over the decision tree algorithm.

**Disadvantages of random forest**

* When using a random forest, more resources are required for computation.
* It consumes more time compared to a decision tree algorithm.

1. Logistic regression

Every machine learning algorithm performs best under a given set of conditions. To ensure good performance, we must know which algorithm to use depending on the problem at hand. You cannot just use one particular algorithm for all problems. For example: Linear regression algorithm cannot be applied on a categorical dependent variable. This is where Logistic Regression comes in.

Logistic Regression is a popular statistical model used for binary classification, that is for predictions of the type *this or that*, *yes or no*, *A or B*, etc. Logistic regression can, however, be used for multiclass classification, but here we will focus on its simplest application. It is one of the most frequently used machine learning algorithms for binary classifications that translates the input to 0 or 1.  For example,

* 0: negative class
* 1: positive class

Some examples of classification are mentioned below:

* Email: spam / not spam
* Online transactions: fraudulent / not fraudulent
* Tumour: malignant / not malignant

What is Logistic Regression

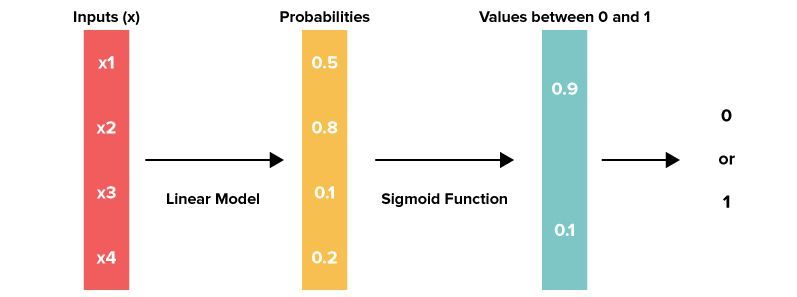
Logistic Regression is the appropriate regression analysis to conduct when the dependent variable has a binary solution. Similar to all other types of regression systems, Logistic Regression is also a type of predictive regression system. Logistic regression is used to evaluate the relationship between one dependent binary variable and one or more independent variables. It gives discrete outputs ranging between 0 and 1.

A simple example of Logistic Regression is: Does calorie intake, weather, and age have any influence on the risk of having a heart attack? The question can have a discrete answer, either “yes” or “no”.

**How Logistic Regression works?**

Logistic Regression uses a more complex cost function than Linear Regression, this cost function is called the ‘Sigmoid function’ or also known as the ‘logistic function’ instead of a linear function.

The hypothesis of logistic regression tends to limit the cost function between 0 and 1. Therefore linear functions fail to represent it as it can have a value greater than 1 or less than 0 which is not possible as per the hypothesis of logistic regression.

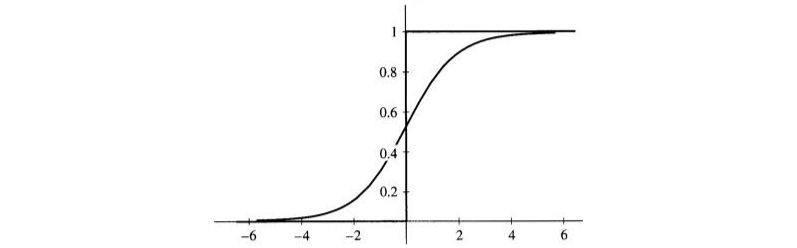


**Sigmoid function** maps any real value into another value between 0 and 1. In machine learning, we use sigmoid to map predictions to probabilities.

Formula:

Where,

f(x) = output between 0 and 1 (probability estimate)  
x = input to the function  
e = base of natural log



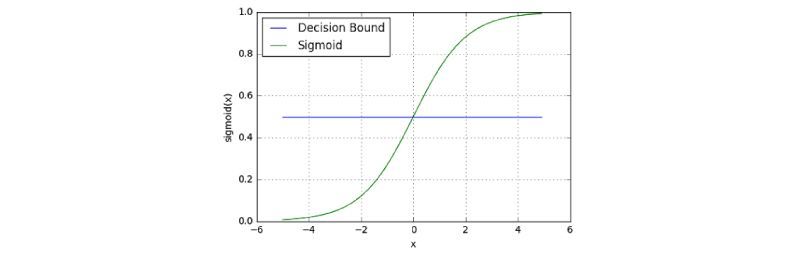
**Decision Boundary**

The prediction function returns a probability score between 0 and 1. If you want to map the discrete class (true/false, yes/no), you will have to select a threshold value above which you will be classifying values into class 1 and below the threshold value into class 2.

*p*≥0.5, *class*=1

*p*<0.5, *class*=0

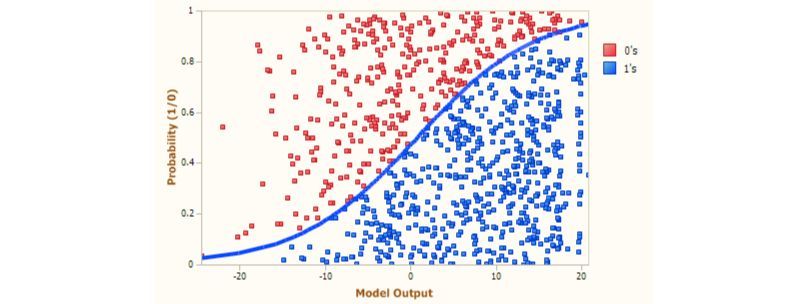
For example, suppose the threshold value is 0.5 and your prediction function returns 0.7, it will be classified as positive. If your predicted value is 0.2, which is less than the threshold value, it will be classified as negative. For logistic regression with multiple classes we could select the class with the highest predicted probability.



Our aim should be to maximize the likelihood that a random data point gets classified correctly, which is called Maximum Likelihood Estimation. Maximum Likelihood Estimation is a general approach to estimating parameters in statistical models. The likelihood can be maximized using an optimization algorithm. Newton’s Method is one such algorithm which can be used to find maximum (or minimum) of many different functions, including the likelihood function.

**When to use Logistic Regression?**

Logistic Regression is used when the input needs to be separated into “two regions” by a linear boundary. The data points are separated using a linear line as shown:



When to use Logistic Regression in Machine Learning

Based on the number of categories, Logistic regression can be classified as:

binomial: target variable can have only 2 possible types: “0” or “1” which may represent “win” vs “loss”, “pass” vs “fails”, “dead” vs “alive”, etc.

multinomial: target variable can have 3 or more possible types which are not ordered (i.e. types have no quantitative significance) like “disease A” vs “disease B” vs “disease C”.

ordinal: it deals with target variables with ordered categories. For example, a test score can be categorized as: “very poor”, “poor”, “good”, “very good”. Here, each category can be given a score like 0, 1, 2, 3.

Let us explore the simplest form of Logistic Regression, i.e. Binomial Logistic Regression. It can be used while solving a classification problem, i.e. when the y-variable takes on only two values. Such a variable is said to be a “binary” or “dichotomous” variable. “Dichotomous” basically means two categories such as yes/no, defective/non-defective, success/failure, and so on. “Binary” refers to the 0's and 1’s

**What is a Confusion Matrix?**

A confusion matrix is a summary of prediction results on a classification problem. The number of correct and incorrect predictions are summarized with count values and broken down by each class. This is the key to the confusion matrix. The confusion matrix shows the ways in which your classification model is confused when it makes predictions.

**Advantages**

* Logistic regression is easier to implement, interpret, and very efficient to train.
* It makes no assumptions about distributions of classes in feature space.
* It can easily extend to multiple classes (multinomial regression) and a natural probabilistic view of class predictions.
* It not only provides a measure of how appropriate a predictor (coefficient size) is, but also its direction of association (positive or negative).

**Disadvantages**

* If the number of observations is lesser than the number of features, Logistic Regression should not be used, otherwise, it may lead to overfitting.
* It constructs linear boundaries.
* The major limitation of Logistic Regression is the assumption of linearity between the dependent variable and the independent variables.
* It can only be used to predict discrete functions. Hence, the dependent variable of Logistic Regression is bound to the discrete number set.
* Non-linear problems can’t be solved with logistic regression because it has a linear decision surface. Linearly separable data is rarely found in real-world scenarios.

**Conclusion**

Logistic Regression is, how it works, when we should use it, comparison of Logistic and Linear Regression, the difference between the approach and usage of two estimation techniques: Maximum Likelihood Estimation and Ordinary Least Square Method, evaluation of model using Confusion Matrix and the advantages and disadvantages of Logistic Regression. We have also covered some basics of sigmoid function, cost function and gradient descent.

1. **Neural network**

Neural networks are a set of algorithms, modelled loosely after the human brain, that are designed to recognize patterns. They interpret sensory data through a kind of machine perception, labelling or clustering raw input. The patterns they recognize are numerical, contained in vectors, into which all real-world data, be it images, sound, text or time series, must be translated.

Neural networks help us cluster and classify. You can think of them as a clustering and classification layer on top of the data you store and manage. They help to group unlabelled data according to similarities among the example inputs, and they classify data when they have a [labelled dataset to train on](https://wiki.pathmind.com/supervised-learning). (Neural networks can also extract features that are fed to other algorithms for clustering and classification; so you can think of deep neural networks as components of larger machine-learning applications involving algorithms for [reinforcement learning](https://wiki.pathmind.com/deep-reinforcement-learning), classification and [regression](https://wiki.pathmind.com/logistic-regression).)

**Predictive Analytics: Regressions**

With classification, deep learning is able to establish correlations between, say, pixels in an image and the name of a person. You might call this a static prediction. By the same token, exposed to enough of the right data, deep learning is able to establish correlations between present events and future events. It can run regression between the past and the future. The future event is like the label in a sense. Deep learning doesn’t necessarily care about time, or the fact that something hasn’t happened yet. Given a time series, deep learning may read a string of number and predict the number most likely to occur next.

* Hardware breakdowns (data centres, manufacturing, transport)
* Health breakdowns (strokes, heart attacks based on vital stats and data from wearables)
* Customer churn (predicting the likelihood that a customer will leave, based on web activity and metadata)
* Employee turnover (ditto, but for employees)

The better we can predict, the better we can prevent and pre-empt. As you can see, with neural networks, we’re moving towards a world of fewer surprises. Not zero surprises, just marginally fewer. We’re also moving toward a world of smarter agents that combine neural networks with other algorithms like [reinforcement learning](https://wiki.pathmind.com/deep-reinforcement-learning) to attain goals.

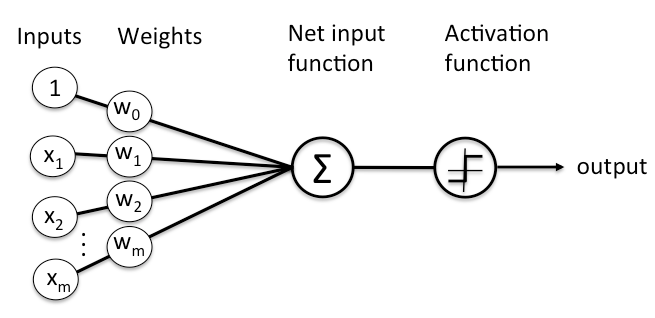
With that brief overview of [deep learning use cases](https://wiki.pathmind.com/use-cases), let’s look at what neural nets are made of.

**Neural Network Elements**

Deep learning is the name we use for “stacked neural networks”; that is, networks composed of several layers.

The layers are made of nodes. A node is just a place where computation happens, loosely patterned on a neuron in the human brain, which fires when it encounters sufficient stimuli. A node combines input from the data with a set of coefficients, or weights, that either amplify or dampen that input, thereby assigning significance to inputs with regard to the task the algorithm is trying to learn; e.g. which input is most helpful is classifying data without error? These input-weight products are summed and then the sum is passed through a node’s so-called activation function, to determine whether and to what extent that signal should progress further through the network to affect the ultimate outcome, say, an act of classification. If the signals passes through, the neuron has been “activated.”

Here’s a diagram of what one node might look like.



**Advantage**

* Storing information on the entire network: Information such as in traditional programming is stored on the entire network, not on a database. The disappearance of a few pieces of information in one place does not restrict the network from functioning.
* The ability to work with inadequate knowledge: After ANN training, the data may produce output even with incomplete information. The lack of performance here depends on the importance of the missing information.
* It has fault tolerance:  Corruption of one or more cells of ANN does not prevent it from generating output. This feature makes the networks fault-tolerant.
* Having a distributed memory: For ANN to be able to learn, it is necessary to determine the examples and to teach the network according to the desired output by showing these examples to the network. The network's progress is directly proportional to the selected instances, and if the event cannot be shown to the network in all its aspects, the network can produce incorrect output
* Gradual corruption:  A network slows over time and undergoes relative degradation. The network problem does not immediately corrode.

**Disadvantage**

* Hardware dependence: Artificial neural networks require processors with parallel processing power, by their structure. For this reason, the realization of the equipment is dependent.
* Unexplained functioning of the network: This is the most important problem of ANN. When ANN gives a probing solution, it does not give a clue as to why and how. This reduces trust in the network.
* Assurance of proper network structure: There is no specific rule for determining the structure of artificial neural networks. The appropriate network structure is achieved through experience and trial and error.
* The difficulty of showing the problem to the network: ANNs can work with numerical information. Problems have to be translated into numerical values before being introduced to ANN. The display mechanism to be determined here will directly influence the performance of the network. This depends on the user's ability.
* The duration of the network is unknown: The network is reduced to a certain value of the error on the sample means that the training has been completed. This value does not give us optimum results.

**CONCLUSION**

The rain forest algorithm is a machine learning algorithm that is easy to use and flexible. It uses ensemble learning, which enables organizations to solve regression and classification problems. This is an ideal algorithm for developers because it solves the problem of overfitting of datasets. It’s a very resourceful tool for making accurate predictions needed in strategic decision making in organizations.

Logistic regression provides a useful means for modelling the dependence of a binary response variable on one or more explanatory variables, where the latter can be either categorical or continuous. The fit of the resulting model can be assessed using a number of methods.

Neural networks are suitable for predicting time series mainly because of learning only from examples, without any need to add additional information that can bring more confusion than prediction effect. Neural networks are able to generalize and are resistant to noise. On the other hand, it is generally not possible to determine exactly what a neural network learned and it is also hard to estimate possible prediction error.