SAPTHAGIRI NPS UNIVERSITY
UNMATCHED EXCELLENCE, UNLIMITED POTENTIAL

1st Semester

Fundamentals of AI and ML (24BTPHY106)

SAPTHAGIRI NPS UNIVERSITY
UNMATCHED EXCELLENCE, UNLIMITED POTENTIAL

Module-4
Machine Learning

Introduction to Machine Learning—Learning from Examples — Forms of Learning—Machine Learning Life cycle—Supervised Learning —Regression vs Classification—Learning Decision Trees — Ensemble Learning — Developing Machine Learning Systems-Case Study

Introduction to Machine Learning

Machine learning (ML) is a subdomain of artificial intelligence (AI) that focuses on developing systems that learn—or improve performance—based on the data they ingest. Artificial intelligence is a broad word that refers to systems or machines that resemble human intelligence. Machine learning and AI are frequently discussed together, and the terms are occasionally used interchangeably, although they do not signify the same thing. A crucial distinction is that, while all machine learning is AI, not all AI is machine learning.

What is Machine Learning?

Machine Learning is the field of study that gives computers the capability to learn without being explicitly programmed. ML is one of the most exciting technologies that one would have ever come across. As it is evident from the name, it gives the computer that makes it more similar to humans: The ability to learn. Machine learning is actively being used today, perhaps in many more places than one would expect.

Learning from Examples:

What is Learning?

Learning is the process of improving performance based on observations.

Examples:

Jotting down a shopping list.

Einstein inferring a new theory of the universe.

Why we use machine learning?

Machine learning is used to make decisions based on data. By modelling the algorithms on the bases of historical data, Algorithms find the patterns and relationships that are difficult for humans to detect. These patterns are now further use for the future references to predict solution of unseen problems.

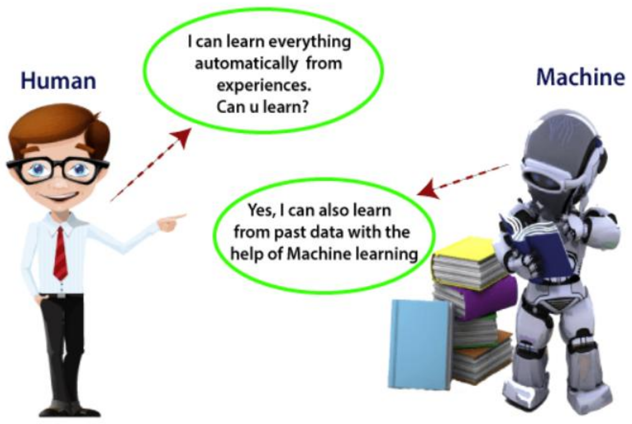

Human
I can learn everything automatically from experiences. Can u learn?
Machine
Yes, I can also learn from past data with the help of Machine learning

Fig: Illustration of Machine Learning (Source: javapoint)

Let's say we have a complex problem in which we need to make predictions. Instead of writing code, we just need to feed the data to generic algorithms, which build the logic based on the data and predict the output. Our perspective on the issue has changed as a result of machine learning. The Machine Learning algorithm's operation is depicted in the following block diagram:

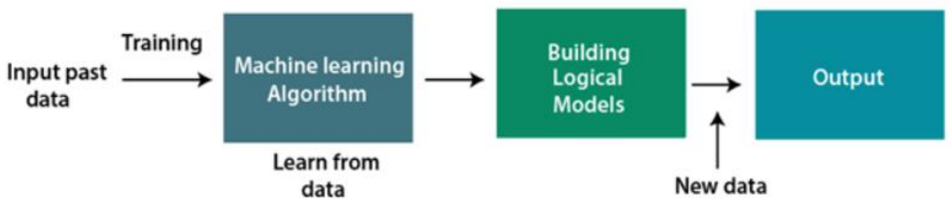


Fig: Machine Learning Process

By providing computer systems with a large amount of data and allowing them to automatically explore the data, build models, and predict the required output, we can train machine learning algorithms. The cost function can be used to determine the amount of data and the machine learning algorithm's performance. We can save both time and money by using machine learning.

Following are some key points which show the importance of Machine Learning:

- Rapid increment in the production of data
- Solving complex problems, which are difficult for a human
- Decision making in various sector including finance
- Finding hidden patterns and extracting useful information from data.

Features of Machine learning

- Machine learning is data driven technology. Large amount of data generated by organizations on daily bases. So, by notable relationships in data, organizations make better decisions.
- Machine can learn itself from past data and automatically improve.
- From the given dataset it detects various patterns on data.
- For the big organizations branding is important and it will become more easy to target relatable customer base.
- It is similar to data mining because it is also deals with the huge amount of data.

Applications of Machine Learning

Machine learning is a buzzword for today's technology, and it is growing very rapidly day by day. We are using machine learning in our daily life even without knowing it such as Google Maps, Google assistant, Alexa, etc. Below are some most trending real-world applications of Machine Learning:

- Robotics: Autonomous navigation and interaction with dynamic environments.
- Finance: Predicting stock prices, managing portfolios, detecting fraud.
- Healthcare: Diagnosing diseases, personalizing treatment plans.
- Natural Language Processing: Language translation, sentiment analysis, chatbots.
- Computer Vision: Facial recognition, object detection, image classification.

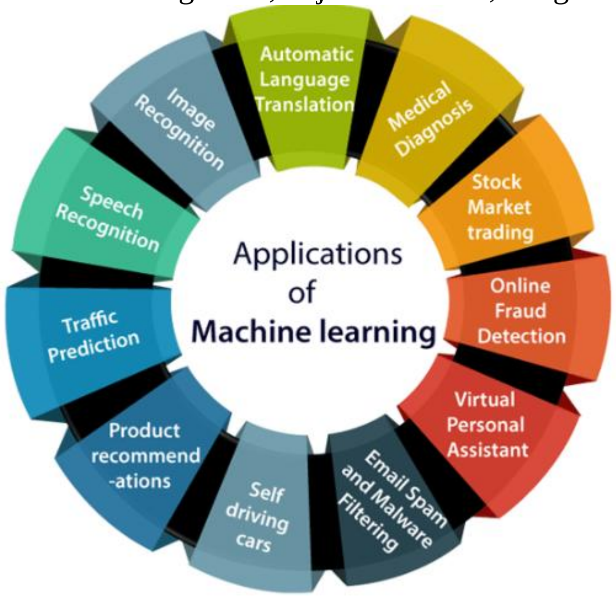


Fig: Applications of Machine Learning

Forms of learning:

Any component of an agent program can be improved by machine learning. The improvements, and the techniques used to make them, depend on these factors:
• Which *component* is to be improved.
• What *prior knowledge* the agent has, which influences the model it builds.
• What *data* and *feedback* on that data is available.

At a broad level, machine learning can be classified into three types:

- Supervised learning
- Unsupervised learning
- Reinforcement learning

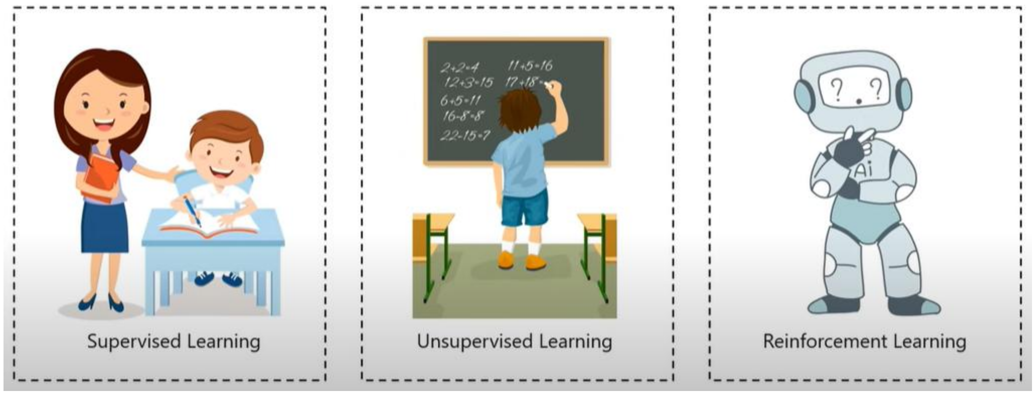


Fig: Types of Machine learning (Source: edureka)

1) Supervised Learning

In supervised learning, sample labelled data are provided to the machine learning system for training, and the system then predicts the output based on the training data. The system uses labelled data to build a model that understands the datasets and learns about each one. After the training and processing are done, we test the model with sample data to see if it can accurately predict the output. The mapping of the input data to the output data is the objective of supervised learning. The managed learning depends on oversight, and it is equivalent to when an understudy learns things in the management of the educator.

Example: Consider a scenario where you have to build an image classifier to differentiate between lions and dogs. If you feed the datasets of dogs and lions labelled images to the algorithm, the machine will learn to classify between a dog or a lion from these labelled images. When we input new dog or lion images that it has never seen before, it will use the learned algorithms and predict whether it is a dog or a lion.

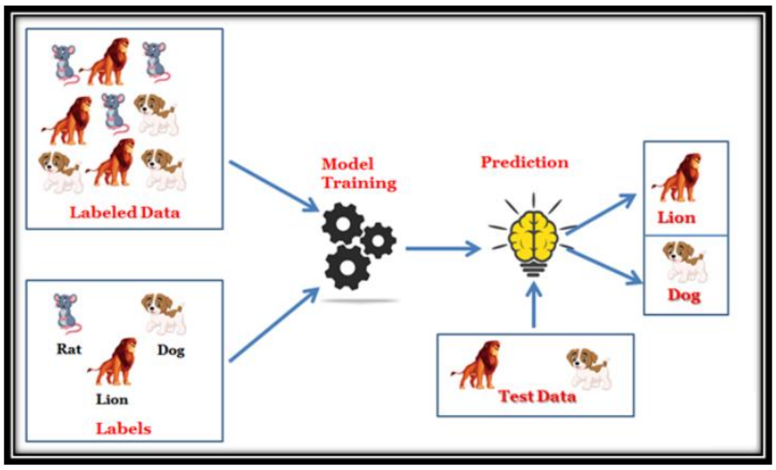


Fig: Illustration of supervised learning

Supervised learning can be grouped further in two categories of algorithms:

- Classification
- Regression

2) Unsupervised Learning

Unsupervised learning is a learning method in which a machine learns without any supervision. The training is provided to the machine with the set of data that has not been labelled, classified, or categorized, and the algorithm needs to act on that data without any supervision. The goal of unsupervised learning is to restructure the input data into new features or a group of objects with similar patterns. In unsupervised learning, we don't have a predetermined result. The machine tries to find useful insights from the huge amount of data.

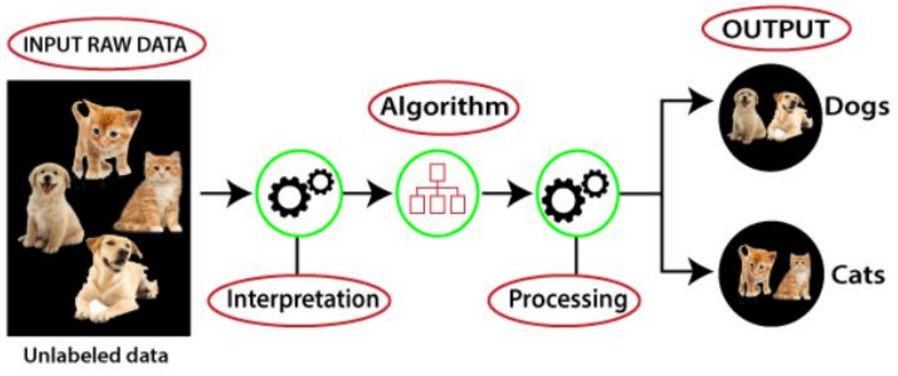


Fig: Illustration of unsupervised learning

It can be further classifieds into two categories of algorithms:

- Clustering
- Association

3) Reinforcement Learning

Reinforcement learning is a feedback-based learning method, in which a learning agent gets a reward for each right action and gets a penalty for each wrong action. The agent learns automatically with these feedbacks and improves its performance. In reinforcement learning, the agent interacts with the environment and explores it. The goal of an agent is to get the most reward points, and hence, it improves its performance.

Example: Consider that you are training an AI agent to play a game like chess. The agent explores different moves and receives positive or negative feedback based on the outcome. Reinforcement Learning also finds applications in which they learn to perform tasks by interacting with their surroundings.

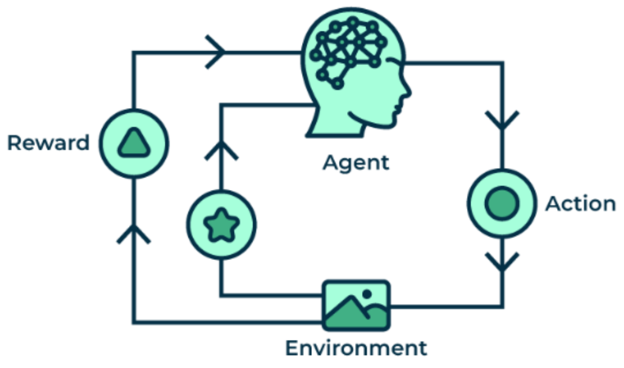


Fig: Illustration of reinforcement learning

While supervised, unsupervised, and reinforcement learning are the foundational types of machine learning, the field is broad and includes many specialized techniques designed to address specific challenges. Understanding these variations can help in choosing the right approach for different types of problems and datasets. Semi-supervised learning is one such type.

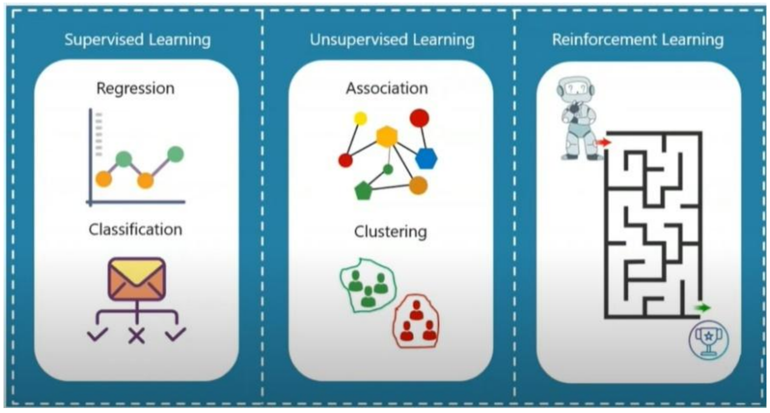


Fig: Types of supervised, unsupervised and reinforcement learning

Semi-Supervised learning is a type of Machine Learning algorithm that lies between Supervised and Unsupervised machine learning. It represents the intermediate ground between Supervised (With Labelled training data) and Unsupervised learning (with no labelled training data) algorithms and uses the combination of labelled and unlabelled datasets during the training period.

Although Semi-supervised learning is the middle ground between supervised and unsupervised learning and operates on the data that consists of a few labels, it mostly consists of unlabelled data. As labels are costly, but for corporate purposes, they may have few labels. It is completely different from supervised and unsupervised learning as they are based on the presence & absence of labels.

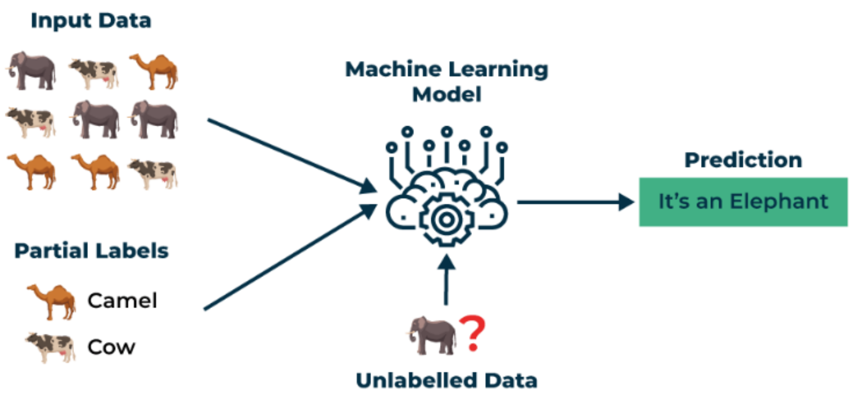


Fig: Illustration of semi-supervised learning

To overcome the drawbacks of supervised learning and unsupervised learning algorithms, the concept of Semi-supervised learning is introduced. The main aim of semi-supervised learning is to effectively use all the available data, rather than only labelled data like in supervised learning. Initially, similar data is clustered along with an unsupervised learning algorithm, and further, it helps to label the unlabelled data into labelled data. It is because labelled data is a comparatively more expensive acquisition than unlabelled data.

We can imagine these algorithms with an example. Supervised learning is where a student is under the supervision of an instructor at home and college. Further, if that student is self-analysing the same concept without any help from the instructor, it comes under unsupervised learning. Under semi-supervised learning, the student has to revise himself after analyzing the same concept under the guidance of an instructor at college.

The Machine Learning Lifecycle

Machine learning has given the computer systems the abilities to automatically learn without being explicitly programmed. But how does a machine learning system work? So, it can be described using the life cycle of machine learning. Machine learning life cycle is a cyclic process to build an efficient machine learning project. The main purpose of the life cycle is to find a solution to the problem or project.

Machine learning life cycle involves seven major steps, which are given below:

- Gathering Data
- Data preparation
- Data Wrangling
- Analyse Data
- Train the model
- Test the model
- Deployment

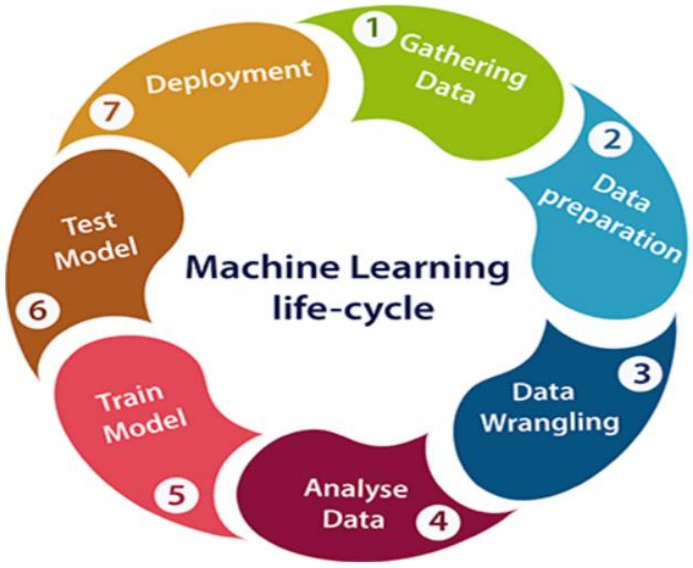


Fig: Illustration of Machine Learning Life cycle

Once we define the problem statement then these are the steps in ML Life cycle:

1. **Gathering Data:**

Data Gathering is the first step of the machine learning life cycle. The goal of this step is to identify and obtain all data-related problems.

In this step, we need to identify the different data sources, as data can be collected from various sources such as files, database, internet, or mobile devices. It is one of the most important steps of the life cycle. The quantity and quality of the collected data will determine the efficiency of the output. The more will be the data, the more accurate will be the prediction.

This step includes the below tasks:

- Identify various data sources
- Collect data
- Integrate the data obtained from different sources

By performing the above task, we get a coherent set of data, also called as a dataset. It will be used in further steps.

2. **Data preparation**

After collecting the data, we need to prepare it for further steps. Data preparation is a step where we put our data into a suitable place and prepare it to use in our machine learning training.

In this step, first, we put all data together, and then randomize the ordering of data.

This step can be further divided into two processes:

- Data exploration:

It is used to understand the nature of data that we have to work with. We need to understand the characteristics, format, and quality of data. A better understanding of data leads to an effective outcome. In this, we find Correlations, general trends, and outliers.

- Data pre-processing:

Now the next step is pre-processing of data for its analysis.

3. **Data Wrangling**

Data wrangling is the process of cleaning and converting raw data into a useable format. It is the process of cleaning the data, selecting the variable to use, and transforming the data in a proper format to make it more suitable for analysis in the next step. It is one of the most important steps of the complete process. Cleaning of data is required to address the quality issues.

It is not necessary that data we have collected is always of our use as some of the data may not be useful. In real-world applications, collected data may have various issues, including:

- Missing Values
- Duplicate data
- Invalid data
- Noise

So, we use various filtering techniques to clean the data.

It is mandatory to detect and remove the above issues because it can negatively affect the quality of the outcome.

4. **Data Analysis**

Now the cleaned and prepared data is passed on to the analysis step. This step involves:

- Selection of analytical techniques
- Building models
- Review the result

The aim of this step is to build a machine learning model to analyze the data using various analytical techniques and review the outcome. It starts with the determination of the type of the problems, where we select the machine learning techniques such as Classification, Regression, Cluster analysis, Association, etc. then build the model using prepared data, and evaluate the model. Hence, in this step, we take the data and use machine learning algorithms to build the model.

5. **Train Model**

Now the next step is to train the model, in this step we train our model to improve its performance for better outcome of the problem. We use datasets to train the model using various machine learning algorithms. Training a model is required so that it can understand the various patterns, rules, and, features.

6. **Test Model**

Once our machine learning model has been trained on a given dataset, then we test the model. In this step, we check for the accuracy of our model by providing a test dataset to it. Testing the model determines the percentage accuracy of the model as per the requirement of project or problem.

7. **Deployment**

The last step of machine learning life cycle is deployment, where we deploy the model in the real-world system.

If the above-prepared model is producing an accurate result as per our requirement with acceptable speed, then we deploy the model in the real system. But before deploying the project, we will check whether it is improving its performance using available data or not. The deployment phase is similar to making the final report for a project.

Supervised Learning:

A machine is said to be learning from past Experiences (data feed-in) with respect to some class of tasks if its Performance in a given Task improves with the Experience. For example, assume that a machine has to predict whether a customer will buy a specific product let’s say “Antivirus” this year or not. The machine will do it by looking at the previous knowledge/past experiences i.e. the data of products that the customer had bought every year and if he buys an Antivirus every year, then there is a high probability that the customer is going to buy an antivirus this year as well. This is how machine learning works at the basic conceptual level.

Supervised learning is a machine learning technique that is widely used in various fields such as finance, healthcare, marketing, and more. It is a form of machine learning in which the algorithm is trained on labelled data to make predictions or decisions based on the data inputs. In supervised learning, the algorithm learns a mapping between the input and output data. This mapping is learned from a labelled dataset, which consists of pairs of input and output data. The algorithm tries to learn the relationship between the input and output data so that it can make accurate predictions on new, unseen data.

Let us discuss what learning for a machine is as shown below media as follows:

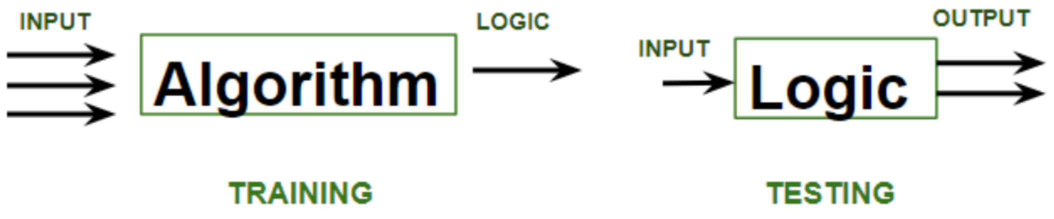
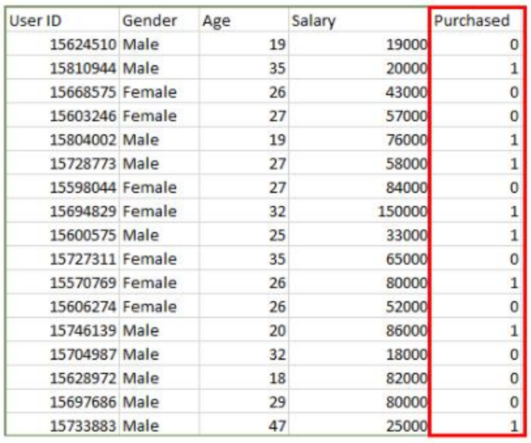


Fig: Learning for a machine

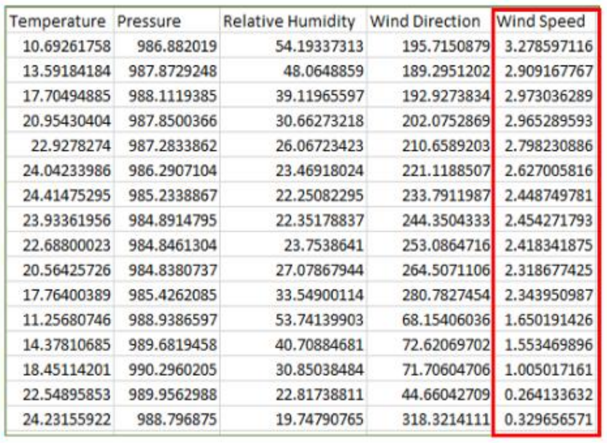
Supervised learning is where the model is trained on a labelled dataset. A labelled dataset is one that has both input and output parameters. In this type of learning both training and validation, datasets are labelled as shown in the figures below.

The labelled dataset used in supervised learning consists of input features and corresponding output labels. The input features are the attributes or characteristics of the data that are used to make predictions, while the output labels are the desired outcomes or targets that the algorithm tries to predict.



User ID	Gender	Age	Salary	Purchased
15624510	Male	19	19000	0
15810944	Male	35	20000	1
15668575	Female	26	43000	0
15603246	Female	27	57000	0
15804002	Male	19	76000	1
15728773	Male	27	58000	1
15598044	Female	27	84000	0
15694829	Female	32	150000	1
15600575	Male	25	33000	1
15727311	Female	35	65000	0
15570769	Female	26	80000	1
15606274	Female	26	52000	0
15746139	Male	20	86000	1
15704987	Male	32	18000	0
15628972	Male	18	82000	0
15697686	Male	29	80000	0
15733883	Male	47	25000	1

Figure A: CLASSIFICATION



Temperature	Pressure	Relative Humidity	Wind Direction	Wind Speed
10.69261758	986.882019	54.19337313	195.7150879	3.278597116
13.59184184	987.8729248	48.0648859	189.2951202	2.909167767
17.70494885	988.1119385	39.11965597	192.9273834	2.973036289
20.95430404	987.8500366	30.66273218	202.0752869	2.965289593
22.9278274	987.2833862	26.06723423	210.6589203	2.798230886
24.04233986	986.2907104	23.46918024	221.1188507	2.627005816
24.41475295	985.2338867	22.25082295	233.7911987	2.448749781
23.93361956	984.8914795	22.35178837	244.3504333	2.454271793
22.68800023	984.8461304	23.7538641	253.0864716	2.418341875
20.56425726	984.8380737	27.07867944	264.5071106	2.318677425
17.76400389	985.4262085	33.54900114	280.7827454	2.343950987
11.25680746	988.9386597	53.74139903	68.15406036	1.650191426
14.37810685	989.6819458	40.70884681	72.62069702	1.553469896
18.45114201	990.2960205	30.85038484	71.70604706	1.005017161
22.54895853	989.9562988	22.81738811	44.66042709	0.264133632
24.23155922	988.796875	19.74790765	318.3214111	0.329656571

Figure B: REGRESSION

Both the above figures have labelled data set as follows:

Figure A: It is a dataset of a shopping store that is useful in predicting whether a customer will purchase a particular product under consideration or not based on his/ her gender, age, and salary.
Input: Gender, Age, Salary
Output: Purchased i.e. 0 or 1; 1 means yes the customer will purchase and 0 means that the customer won't purchase it.

Figure B: It is a Meteorological dataset that serves the purpose of predicting wind speed based on different parameters.
Input: Dew Point, Temperature, Pressure, Relative Humidity, Wind Direction
Output: Wind Speed

Training the system: While training the model, data is usually split in the ratio of 80:20 i.e. 80% as training data and the rest as testing data. In training data, we feed input as well as output for 80% of data. The model learns from training data only. We use different machine learning algorithms (which we will discuss in detail in the next articles) to build our model. Learning means that the model will build some logic of its own.

Once the model is ready then it is good to be tested. At the time of testing, the input is fed from the remaining 20% of data that the model has never seen before, the model will predict some value and we will compare it with the actual output and calculate the accuracy.

Supervised learning can be further divided into several different types other than classification and regression (which will be discussed in later part), each with its own unique characteristics and applications. Here are some of the most common types of supervised learning algorithms:

- *Linear Regression:* Linear regression is a type of regression algorithm that is used to predict a continuous output value. It is one of the simplest and most widely used algorithms in supervised learning. In linear regression, the algorithm tries to find a linear relationship between the input features and the output value. The output value is predicted based on the weighted sum of the input features.
- *Logistic Regression:* Logistic regression is a type of classification algorithm that is used to predict a binary output variable. It is commonly used in machine learning applications where the output variable is either true or false, such as in fraud detection or spam filtering. In logistic regression, the algorithm tries to find a linear relationship between the input features and the output variable. The output variable is then transformed using a logistic function to produce a probability value between 0 and 1.
- *Decision Trees:* Decision tree is a tree-like structure that is used to model decisions and their possible consequences. Each internal node in the tree represents a decision, while each leaf node represents a possible outcome. Decision trees can be used to model complex relationships between input features and output variables.

  A decision tree is a type of algorithm that is used for both classification and regression tasks.
  - *Decision Trees Regression:* Decision Trees can be utilized for regression tasks by predicting the value linked with a leaf node.
  - *Decision Trees Classification:* Random Forest is a machine learning algorithm that uses multiple decision trees to improve classification and prevent overfitting.
- *Random Forests:* Random forests are made up of multiple decision trees that work together to make predictions. Each tree in the forest is trained on a different subset of the input features and data. The final prediction is made by aggregating the predictions of all the trees in the forest.

  Random forests are an ensemble learning technique that is used for both classification and regression tasks.
  - *Random Forest Regression:* It combines multiple decision trees to reduce overfitting and improve prediction accuracy.
  - *Random Forest Classifier*: Combines several decision trees to improve the accuracy of classification while minimizing overfitting.
- *Support Vector Machine(SVM):* The SVM algorithm creates a hyperplane to segregate n-dimensional space into classes and identify the correct category of new data points. The extreme cases that help create the hyperplane are called support vectors, hence the name Support Vector Machine.

  A Support Vector Machine is a type of algorithm that is used for both classification and regression tasks
  - *Support Vector Regression:* It is a extension of Support Vector Machines (SVM) used for predicting continuous values.

 - *Support Vector Classifier:* It aims to find the best hyperplane that maximizes the margin between data points of different classes.

- *K-Nearest Neighbors (KNN):* KNN works by finding k training examples closest to a given input and then predicts the class or value based on the majority class or average value of these neighbors. The performance of KNN can be influenced by the choice of k and the distance metric used to measure proximity. However, it is intuitive but can be sensitive to noisy data and requires careful selection of k for optimal results.

  A K-Nearest Neighbors (KNN) is a type of algorithm that is used for both classification and regression tasks.
  - *K-Nearest Neighbors Regression:* It predicts continuous values by averaging the outputs of the k closest neighbors.
  - *K-Nearest Neighbors Classification:* Data points are classified based on the majority class of their k closest neighbors.
- *Gradient Boosting*: Gradient Boosting combines weak learners, like decision trees, to create a strong model. It iteratively builds new models that correct errors made by previous ones. Each new model is trained to minimize residual errors, resulting in a powerful predictor capable of handling complex data relationships.

  A Gradient Boosting is a type of algorithm that is used for both classification and regression tasks.
  - *Gradient Boosting Regression*: It builds an ensemble of weak learners to improve prediction accuracy through iterative training.
  - *Gradient Boosting Classification:* Creates a group of classifiers to continually enhance the accuracy of predictions through iterations

Advantages of Supervised Learning

- The power of supervised learning lies in its ability to accurately predict patterns and make data-driven decisions across a variety of applications. Here are some advantages listed below:
- Labelled training data benefits supervised learning by enabling models to accurately learn patterns and relationships between inputs and outputs.
- Supervised learning models can accurately predict and classify new data.
- Supervised learning has a wide range of applications, including classification, regression, and even more complex problems like image recognition and natural language processing.
- Well-established evaluation metrics, including accuracy, precision, recall, and F1-score, facilitate the assessment of supervised learning model performance.

Disadvantages of Supervised Learning

- Although supervised learning methods have benefits, their limitations require careful consideration during problem formulation, data collection, model selection, and evaluation. Here are some disadvantages listed below:
- Overfitting: Models can overfit training data, which leads to poor performance on new, unseen data due to the capture of noise.
- Feature Engineering: Extracting relevant features from raw data is crucial for model performance, but this process can be time-consuming and may require domain expertise.
- Bias in Models: Training data biases can lead to unfair predictions.

- Supervised learning heavily depends on labeled training data, which can be costly, time-consuming, and may require domain expertise.

Regression vs Classification

Regression and Classification algorithms are Supervised Learning algorithms. Both the algorithms are used for prediction in Machine learning and work with the labelled datasets. But the difference between both is how they are used for different machine learning problems.

The main difference between Regression and Classification algorithms that Regression algorithms are used to predict the continuous values such as price, salary, age, etc. and Classification algorithms are used to predict/Classify the discrete values such as Male or Female, True or False, Spam or Not Spam, etc.

Consider the below diagram:

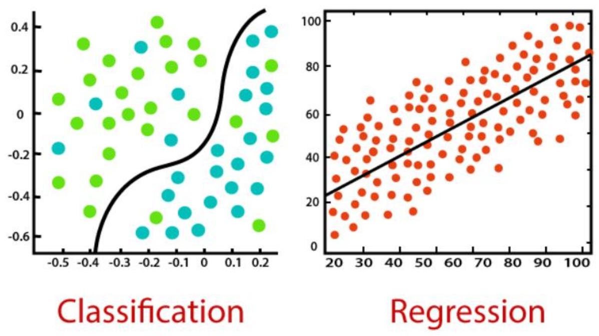


Fig: Illustration of Classification and Regression

Classification:

Classification is a process of finding a function which helps in dividing the dataset into classes based on different parameters. In Classification, a computer program is trained on the training dataset and based on that training, it categorizes the data into different classes.

The task of the classification algorithm is to find the mapping function to map the input(x) to the discrete output(y).

Example: The best example to understand the Classification problem is Email Spam Detection. The model is trained on the basis of millions of emails on different parameters, and whenever it receives a new email, it identifies whether the email is spam or not. If the email is spam, then it is moved to the Spam folder.

Types of ML Classification Algorithms:

- Decision Tree
- Random Forest Classifier
- K – Nearest Neighbors
- Support Vector Machine

Regression:

Regression is a process of finding the correlations between dependent and independent variables. It helps in predicting the continuous variables such as prediction of Market Trends, prediction of House prices, etc.

The task of the Regression algorithm is to find the mapping function to map the input variable(x) to the continuous output variable(y).

Example: Suppose we want to do weather forecasting, so for this, we will use the Regression algorithm. In weather prediction, the model is trained on the past data, and once the training is completed, it can easily predict the weather for future days.

Types of Regression Algorithm:

- Lasso Regression
- Ridge Regression
- XGBoost Regression
- LGBM Regression

Difference between Regression and Classification

Regression	Classification
In Regression, the output variable must be of continuous nature or real value.	In Classification, the output variable must be a discrete value.
The task of the regression algorithm is to map the input value (x) with the continuous output variable(y).	The task of the classification algorithm is to map the input value(x) with the discrete output variable(y).
Regression Algorithms are used with continuous data.	Classification Algorithms are used with discrete data.
In Regression, we try to find the best fit line, which can predict the output more accurately.	In Classification, we try to find the decision boundary, which can divide the dataset into different classes.
Regression algorithms can be used to solve the regression problems such as Weather Prediction, House price prediction, etc.	Classification Algorithms can be used to solve classification problems such as Identification of spam emails, Speech Recognition, Identification of cancer cells, etc.
The regression Algorithm can be further divided into Linear and Non-linear Regression.	The Classification algorithms can be divided into Binary Classifier and Multi-class Classifier.

Learning Decision Trees:

Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.

In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.

The decisions or the test are performed on the basis of features of the given dataset. It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions. It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure. In order to build a tree, we use the CART algorithm, which stands for Classification and Regression Tree algorithm.

A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.

Below diagram explains the general structure of a decision tree:

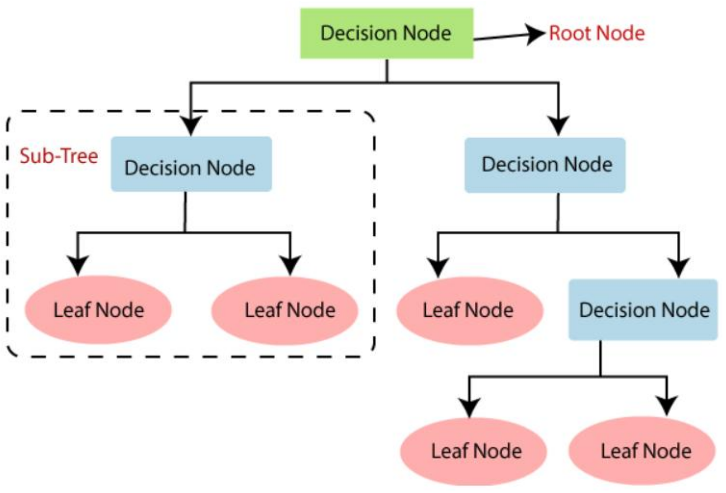


Fig: Illustration of Decision Tree

Why use Decision Trees?

There are various algorithms in Machine learning, so choosing the best algorithm for the given dataset and problem is the main point to remember while creating a machine learning model. Below are the two reasons for using the Decision tree:

Decision Trees usually mimic human thinking ability while making a decision, so it is easy to understand.

The logic behind the decision tree can be easily understood because it shows a tree-like structure.

Decision Tree Terminologies

- **Root Node:** Root node is from where the decision tree starts. It represents the entire dataset, which further gets divided into two or more homogeneous sets.
- **Leaf Node:** Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.
- **Splitting:** Splitting is the process of dividing the decision node/root node into sub-nodes according to the given conditions.
- **Branch/Sub Tree:** A tree formed by splitting the tree.
- **Pruning:** Pruning is the process of removing the unwanted branches from the tree.
- **Parent/Child node:** The root node of the tree is called the parent node, and other nodes are called the child nodes.

How does the Decision Tree Algorithm Work?

In a decision tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree. This algorithm compares the values of root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node.

For the next node, the algorithm again compares the attribute value with the other sub-nodes and move further. It continues the process until it reaches the leaf node of the tree. The complete process can be better understood using the below algorithm:

- **Step-1:** Begin the tree with the root node, says S, which contains the complete dataset.
- **Step-2:** Find the best attribute in the dataset using **Attribute Selection Measure (ASM).**
- **Step-3:** Divide the S into subsets that contains possible values for the best attributes.
- **Step-4:** Generate the decision tree node, which contains the best attribute.
- **Step-5:** Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

Example: Suppose there is a candidate who has a job offer and wants to decide whether he should accept the offer or Not. So, to solve this problem, the decision tree starts with the root node (Salary attribute by ASM). The root node splits further into the next decision node (distance from the office) and one leaf node based on the corresponding labels. The next decision node further gets split into one decision node (Cab facility) and one leaf node. Finally, the decision node splits into two leaf nodes (Accepted offers and Declined offer). Consider the below diagram:

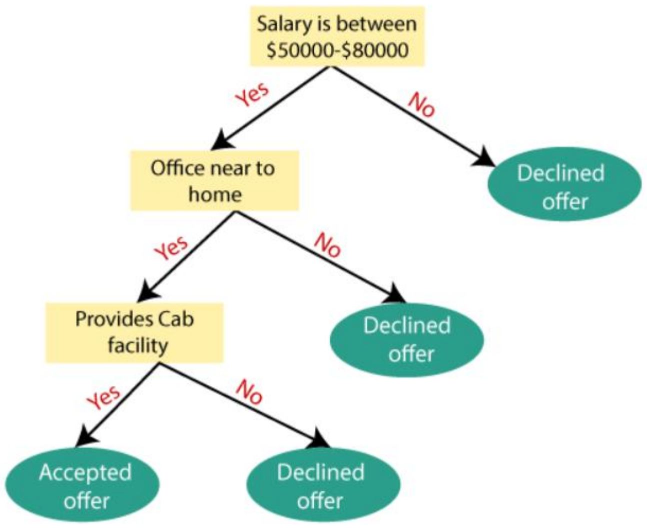


Fig: Decision Tree Example

Attribute Selection Measures

While implementing a Decision tree, the main issue arises that how to select the best attribute for the root node and for sub-nodes. So, to solve such problems there is a technique which is called as Attribute selection measure or ASM. By this measurement, we can easily select the best attribute for the nodes of the tree. There are two popular techniques for ASM, which are:

- Information Gain
- Gini Index

1. **Information Gain:**

Information gain is the measurement of changes in entropy after the segmentation of a dataset based on an attribute.
It calculates how much information a feature provides us about a class.
According to the value of information gain, we split the node and build the decision tree.
A decision tree algorithm always tries to maximize the value of information gain, and a node/attribute having the highest information gain is split first. It can be calculated using the below formula:
E
n

Information Gain= Entropy(S)- [(Weighted Avg) *Entropy(each

t
ropy: Entropy is a metric to measure the impurity in a given attribute. It specifies randomness in data. Entropy can be calculated as:
W
h

```
Entropy(s)= -P(yes)log2P(yes)- P(no)log2P(no)
```

e
re,

- S= Total number of samples
- P(yes)= probability of yes
- P(no)= probability of no

2. **Gini Index:**

Gini index is a measure of impurity or purity used while creating a decision tree in the CART (Classification and Regression Tree) algorithm.
An attribute with the low Gini index should be preferred as compared to the high Gini index.
It only creates binary splits, and the CART algorithm uses the Gini index to create binary splits.
Gini index can be calculated using the below formula:

$$\text{Gini Index}= 1-\sum_j P_j^2$$

Pruning: Getting an Optimal Decision tree
Pruning is a process of deleting the unnecessary nodes from a tree in order to get the optimal decision tree.
A too-large tree increases the risk of overfitting, and a small tree may not capture all the important features of the dataset. Therefore, a technique that decreases the size of the learning tree without reducing accuracy is known as Pruning. There are mainly two types of tree pruning technology used:

- Cost Complexity Pruning
- Reduced Error Pruning.

Advantages of the Decision Tree

- It is simple to understand as it follows the same process which a human follows while making any decision in real-life.
- It can be very useful for solving decision-related problems.
- It helps to think about all the possible outcomes for a problem.
- There is less requirement of data cleaning compared to other algorithms.

Disadvantages of the Decision Tree

- The decision tree contains lots of layers, which makes it complex.
- It may have an overfitting issue, which can be resolved using the Random Forest algorithm.
- For more class labels, the computational complexity of the decision tree may increase.

Model Selection and Optimization in Machine Learning

Model selection and optimization are critical steps in building effective machine learning models. They involve choosing the best model from a set of candidates and fine-tuning it to achieve optimal performance. This process ensures that the model generalizes well to new, unseen data.

Model Selection

Model selection refers to the process of evaluating and choosing between different machine learning algorithms or models. It includes the following steps:

- **Define the Problem:** Understand the nature of the problem and the type of data available.
- **Select Candidate Models:** Choose a set of models to evaluate. These can include linear models, tree-based models, neural networks, etc.
- **Evaluation Metric:** Choose an appropriate evaluation metric (e.g., accuracy, precision, recall, F1-score, AUC-ROC) based on the problem.
- **Cross-Validation:** Use techniques like k-fold cross-validation to assess the performance of each model on different subsets of the data.
- **Comparison:** Compare the performance of the models based on the chosen metric and select the best-performing model.

Cross-Validation

Cross-validation is a statistical method used to estimate the performance of machine learning models. The most common form is k-fold cross-validation, where the data is split into k subsets (folds). The model is trained on k-1 folds and tested on the remaining fold. This process is repeated k times, and the results are averaged to get an overall performance estimate.

Hyperparameter Optimization

Hyperparameter optimization involves finding the best set of hyperparameters for a model. Hyperparameters are configuration settings used to control the learning process, which are not learned from the data. Common methods include:

- **Grid Search:** A brute-force method that exhaustively searches through a specified subset of hyperparameters.
- **Random Search:** Randomly samples hyperparameter combinations and evaluates them.
- **Bayesian Optimization:** Uses a probabilistic model to select the most promising hyperparameters to evaluate, balancing exploration and exploitation.
- **Gradient-Based Optimization:** Methods like gradient descent can sometimes be adapted for hyperparameter tuning, particularly in neural networks.

Model Evaluation Metrics

Choosing the right evaluation metric is crucial. Different metrics are used depending on the nature of the problem (classification, regression, etc.):

- **Classification Metrics:** Accuracy, precision, recall, F1-score, AUC-ROC, log loss.
- **Regression Metrics:** Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), R-squared.

Regularization Techniques

Regularization helps to prevent overfitting by adding a penalty term to the loss function. Common techniques include:

- **Regularization (Lasso):** Adds an absolute value of coefficients as penalty.
- **L2 Regularization (Ridge):** Adds the square of coefficients as penalty.
- **Elastic Net:** Combines L1 and L2 regularization.

Ensemble Methods (Detailed explanation is given later)

Ensemble methods combine multiple models to improve performance. Common techniques include:

Practical Considerations

- **Data Preprocessing**: Ensure data is clean and properly preprocessed (handling missing values, normalization, etc.).
- **Feature Engineering:** Create meaningful features that can improve model performance.
- **Model Complexity:** Balance model complexity with the amount of available data to avoid overfitting and underfitting.
- **Computational Resources:** Consider the computational cost and time required for training and hyperparameter tuning.

Model selection and optimization are essential for building robust machine learning models. By carefully selecting models, tuning hyperparameters, and using appropriate evaluation metrics, we can create models that perform well on new, unseen data. Regularization and ensemble methods further enhance model performance and generalization.

Theory of Learning in Machine Learning

The theory of learning in machine learning encompasses the principles and methodologies that enable machines to learn from data. It combines aspects of statistics, probability, and computer science to build models that can make predictions or decisions based on data.

Learning Paradigms

There are several paradigms in machine learning, each suited for different types of problems and data:

- **Supervised Learning:** Learning from labeled data where the algorithm is trained on input-output pairs. Common algorithms include linear regression, decision trees, and neural networks.

- **Unsupervised Learning:** Learning from unlabeled data to identify patterns or structure. Common algorithms include clustering (e.g., K-means) and dimensionality reduction (e.g., PCA).
- **Semi-Supervised Learning:** Combines labeled and unlabeled data to improve learning accuracy.
- **Reinforcement Learning:** Learning by interacting with an environment and receiving rewards or penalties. Common algorithms include Q-learning and Deep Q-Networks (DQN).

Key Concepts in Learning Theory

- **Hypothesis Space:** The set of all possible models that can be learned from the data. Selecting a suitable hypothesis space is crucial for effective learning.
- **Inductive Bias:** The set of assumptions a learning algorithm makes to generalize from the training data to unseen data. Inductive bias helps in choosing the hypothesis space.
- **Overfitting and Underfitting:** Overfitting occurs when a model learns the noise in the training data, performing well on training data but poorly on unseen data. Underfitting occurs when a model is too simple to capture the underlying pattern in the data.
- **Bias-Variance Tradeoff:** A fundamental tradeoff in machine learning where high bias (underfitting) leads to a simple model, and high variance (overfitting) leads to a complex model. The goal is to find a balance that minimizes total error.

PAC Learning Framework

The Probably Approximately Correct (PAC) learning framework is a theoretical framework to understand learning:

- PAC Learnability: A class of functions is PAC-learnable if there exists an algorithm that can learn it with high probability and within a polynomial time, given enough training samples.
- Sample Complexity: The number of training samples required to ensure that a hypothesis is PAC-learnable.
- VC Dimension: The Vapnik-Chervonenkis (VC) dimension is a measure of the capacity of a statistical model, defined as the size of the largest set of points that the model can shatter. Higher VC dimension indicates a more complex model.

Statistical Learning Theory

Statistical learning theory provides a framework for understanding the process of learning from a probabilistic perspective:

- Empirical Risk Minimization (ERM): The process of minimizing the error on the training data. ERM can lead to overfitting.
- Structural Risk Minimization (SRM): Aims to balance the complexity of the hypothesis space and the empirical risk to minimize the overall risk, addressing the overfitting problem.
- Generalization Bounds: Theoretical bounds that quantify the difference between the training error and the true error on unseen data. They help in understanding how well a model will generalize.

Optimization in Learning

Optimization is central to learning algorithms:

- Convex Optimization: Many learning algorithms involve convex optimization problems, which are easier to solve due to the absence of local minima.
- Gradient Descent: A common optimization technique used in training models, where the parameters are updated iteratively to minimize the loss function.
- Stochastic Gradient Descent (SGD): A variant of gradient descent that uses a single or a subset of training samples to update the parameters, providing computational efficiency.

Regularization Techniques

Regularization helps prevent overfitting by adding a penalty to the loss function:

- L1 Regularization (Lasso): Encourages sparsity by adding the absolute value of the coefficients as a penalty.
- L2 Regularization (Ridge): Adds the square of the coefficients as a penalty, encouraging smaller coefficients.

Ensemble Learning (Detailed explanation given later)

Ensemble learning combines multiple models to improve performance

The No Free Lunch Theorem

The No Free Lunch Theorem states that no single learning algorithm works best for all problems. The performance of a learning algorithm depends on the specific problem and data characteristics. Therefore, selecting and tuning algorithms for specific tasks is crucial.

The theory of learning in machine learning provides a foundation for understanding how algorithms learn from data, balance complexity, and generalize to new data. It involves various paradigms, concepts, and techniques that guide the development and application of effective learning algorithms. By leveraging these theoretical insights, practitioners can build robust and efficient machine learning models

Nonparametric Models in Machine Learning

Nonparametric models are a class of machine learning models that do not assume a fixed form or parameter count for the underlying data distribution. Instead, they have a flexible structure that can grow with the data. These models are particularly useful when the data distribution is unknown or complex.

Characteristics of Nonparametric Models

- Flexibility: Nonparametric models can adapt to the underlying structure of the data without predefined parameters.
- Data Dependency: The complexity of nonparametric models increases with the amount of data, allowing them to capture intricate patterns.
- No Assumptions: Unlike parametric models, nonparametric models do not make strong assumptions about the data distribution.

Types of Nonparametric Models

- **k-Nearest Neighbors (k-NN):**

Concept: Classifies a data point based on the majority class of its k nearest neighbors.

Strengths: Simple, intuitive, and effective for small datasets with clear patterns.

Weaknesses: Computationally expensive for large datasets, sensitive to the choice of k and distance metric.

- **Decision Trees:**

Concept: Uses a tree-like structure to make decisions based on feature values, splitting data at each node.

Strengths: Interpretable, can handle both numerical and categorical data, requires little data preprocessing.

Weaknesses: Prone to overfitting, especially with deep trees.

- **Random Forests:**

Concept: An ensemble method that combines multiple decision trees to improve accuracy and reduce overfitting.

Strengths: Robust to overfitting, handles high-dimensional data well, and provides feature importance.

Weaknesses: Less interpretable than single decision trees, computationally intensive.

- **Kernel Density Estimation (KDE):**

Concept: Estimates the probability density function of a random variable using a kernel function.

Strengths: Provides a smooth estimate of the data distribution, useful for visualizing data.

Weaknesses: Choice of kernel and bandwidth significantly affects the estimate, computationally intensive for large datasets.

- **Support Vector Machines (SVM) with RBF Kernel:**

Concept: A powerful classification algorithm that can be adapted to nonparametric settings using the Radial Basis Function (RBF) kernel.

Strengths: Effective in high-dimensional spaces, robust to outliers, can model complex boundaries.

Weaknesses: Computationally intensive, requires careful tuning of hyperparameters.

- **Gaussian Processes (GP):**

Concept: A Bayesian approach to regression that assumes a distribution over functions and uses observed data to update beliefs.

Strengths: Provides uncertainty estimates, flexible and can model complex functions.

Weaknesses: Computationally expensive for large datasets, requires careful selection of the kernel function.

Advantages of Nonparametric Models

- **Flexibility:** Can model complex relationships and capture subtle patterns in data.
- **Adaptability:** The model complexity grows with the data, allowing it to improve with more data.
- **Fewer Assumptions:** Less reliance on prior assumptions about the data distribution, making them more applicable to diverse datasets.

Disadvantages of Nonparametric Models

- Computational Complexity: Often require significant computational resources, especially for large datasets.
- Overfitting: High flexibility can lead to overfitting if not properly regularized or tuned.
- Interpretability: Some nonparametric models, especially ensemble methods, can be challenging to interpret.

Applications of Nonparametric Models

- **Classification:** Nonparametric models like k-NN, decision trees, and random forests are widely used for classification tasks in various domains, including image recognition, text classification, and bioinformatics.
- **Regression:** Models like Gaussian Processes and decision trees are used for regression tasks in fields such as finance, economics, and engineering.
- **Density Estimation:** Kernel Density Estimation is used for estimating data distributions in fields like epidemiology, economics, and environmental science.
- **Anomaly Detection:** Nonparametric models are effective in identifying anomalies in datasets where the distribution is not well understood, such as fraud detection and network security.

Nonparametric models provide a flexible and powerful framework for machine learning tasks. Their ability to adapt to the underlying data structure without strong assumptions makes them suitable for a wide range of applications. However, their computational complexity and potential for overfitting require careful consideration and tuning. By leveraging the strengths of nonparametric models, practitioners can build robust models capable of capturing complex patterns in data.

Ensemble Learning:

Ensemble means ‘a collection of things’ and in Machine Learning terminology, Ensemble learning refers to the approach of combining multiple ML models to produce a more accurate and robust prediction compared to any individual model. It implements an ensemble of fast algorithms (classifiers) such as decision trees for learning and allows them to vote.

- Ensemble learning is a machine learning technique that combines the predictions from multiple individual models to obtain a better predictive performance than any single model. The basic idea behind ensemble learning is to leverage the wisdom of the crowd by aggregating the predictions of multiple models, each of which may have its own strengths and weaknesses. This can lead to improved performance and generalization.

Benefits of using Ensemble learning

- Combines strengths of multiple models, leading to more accurate predictions.
- Averaging predictions reduces individual errors, leading to stable performance.
- Boosting focuses on correcting errors, leading to less biased predictions.
- Captures diverse features from different models, leading to a better understanding of the problem.

Types of Ensemble learning:

1) **Bagging**
    - Goal: Reduce variance and improve stability
    - Method: Trains multiple models on different subsets of the training data (with replacement)
    - Prediction: Combines the predictions of individual models, typically through averaging
    - Examples: Random Forest, Random Subspace
2) **Boosting**
    - Goal: Reduce bias and improve accuracy
    - Method: Sequentially builds models, focusing on correcting previous model's errors
    - Prediction: Combines the predictions of individual models, typically through weighted averaging
    - Examples: AdaBoost, XGBoost, LightGBM
3) **Stacking**
    - Goal: Leverage the strengths of multiple models
    - Method: Trains a meta-model to combine the predictions of multiple base models
    - Prediction: The meta-model produces the final prediction
    - Examples: Super Learner, Blending

Applications:

In ensemble learning, the power lies in combining the strengths of multiple models to achieve better performance on a specific task. Let's break down how ensemble learning is applied in different fields:

Finance:

Stock Price Prediction: By combining predictions from various models trained on historical data, financial analysts can get a more robust forecast of future stock prices. This can consider factors like market trends, company performance, and economic indicators.

Healthcare:

Disease Diagnosis: Ensembles can analyze patient data (symptoms, medical history, test results) from multiple models to improve the accuracy of disease diagnosis. This can lead to earlier detection and better treatment plans.

Natural Language Processing (NLP):

Sentiment Analysis: Ensembles can combine the analysis of different algorithms to understand the sentiment (positive, negative, neutral) expressed in text data like social media posts or customer reviews. This leads to a more nuanced understanding of public opinion.

Computer Vision:

Object Detection: Combining the strengths of various object recognition models allows for more accurate detection of objects in images and videos. This is crucial for self-driving cars and security systems.

Recommendation Systems:

Recommending Products: Ensemble methods can analyze user behavior data and combine predictions from different recommendation algorithms to suggest products relevant to each user's preferences. This personalizes the shopping experience.

These are just a few examples. Ensemble learning brings together diverse perspectives from various models, leading to more robust, reliable, and accurate predictions across various applications.

Fig: Applications of Ensemble learning

Developing Machine Learning Systems:

According to Arthur Samuel *“Machine Learning enables a Machine to Automatically learn from Data, improve performance from an Experience and predict things without explicitly programmed.”*

In Simple Words, when we fed the Training Data to Machine Learning Algorithm, this algorithm will produce a mathematical model and with the help of the mathematical model, the machine will make a prediction and take a decision without being explicitly programmed. Also, during training data, the more machine will work with it the more it will get experience and the more efficient result is produced.

Example: In Driverless Car, the training data is fed to Algorithm like how to Drive Car in Highway, Busy and Narrow Street with factors like speed limit, parking, stop at signal etc. After that, a Logical and Mathematical model is created on the basis of that and after that, the car will work according to the logical model. Also, the more data the data is fed the more efficient output is produced.

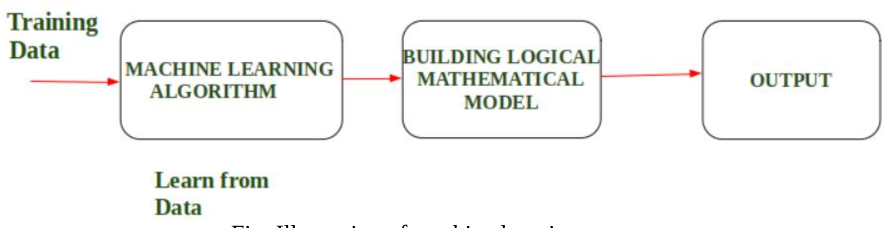

Training Data
MACHINE LEARNING ALGORITHM
BUILDING LOGICAL MATHEMATICAL MODEL
OUTPUT
Learn from Data

Fig: Illustration of machine learning process

Designing a Learning System in Machine Learning:
According to Tom Mitchell, "A computer program is said to be learning from experience (E), with respect to some task (T). Thus, the performance measure (P) is the performance at task T, which is measured by P, and it improves with experience E."
Example: In Spam E-Mail detection,

Task, T: To classify mails into Spam or Not Spam.

Performance measure, P: Total percent of mails being correctly classified as being "Spam" or "Not Spam".

Experience, E: Set of Mails with label "Spam"

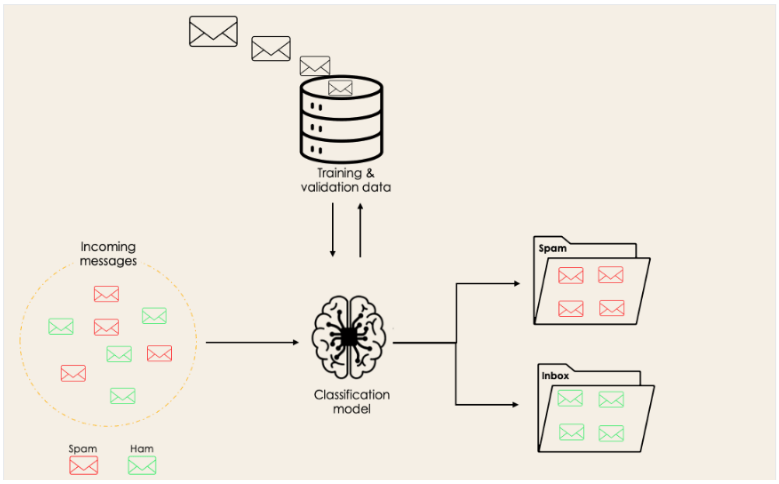


Fig: Illustration Spam e-mail detection

Steps for Designing Learning System are:

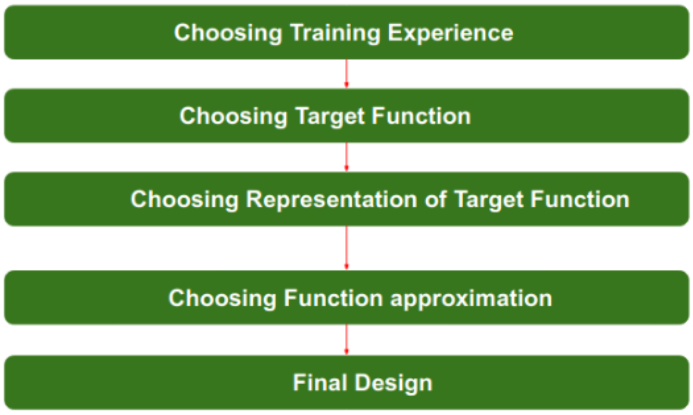

Choosing Training Experience
Choosing Target Function
Choosing Representation of Target Function
Choosing Function approximation
Final Design

Fig: Steps for Design Learning Systems

Step 1: *Choosing the Training Experience:* The very important and first task is to choose the training data or training experience which will be fed to the Machine Learning Algorithm. It is important to note that the data or experience that we fed to the algorithm must have a significant impact on the Success or Failure of the Model. So Training data or experience should be chosen wisely.

Below are the attributes which will impact on Success and Failure of Data:

- The training experience will be able to provide direct or indirect feedback regarding choices.

  For example: While Playing chess the training data will provide feedback to itself like instead of this move if this is chosen the chances of success increases.
- Second important attribute is the degree to which the learner will control the sequences of training examples.

  For example: when training data is fed to the machine then at that time accuracy is very less but when it gains experience while playing again and again with itself or opponent the machine algorithm will get feedback and control the chess game accordingly.
- Third important attribute is how it will represent the distribution of examples over which performance will be measured.

  For example, a Machine learning algorithm will get experience while going through a number of different cases and different examples.

Thus, Machine Learning Algorithm will get more and more experience by passing through more and more examples and hence its performance will increase.

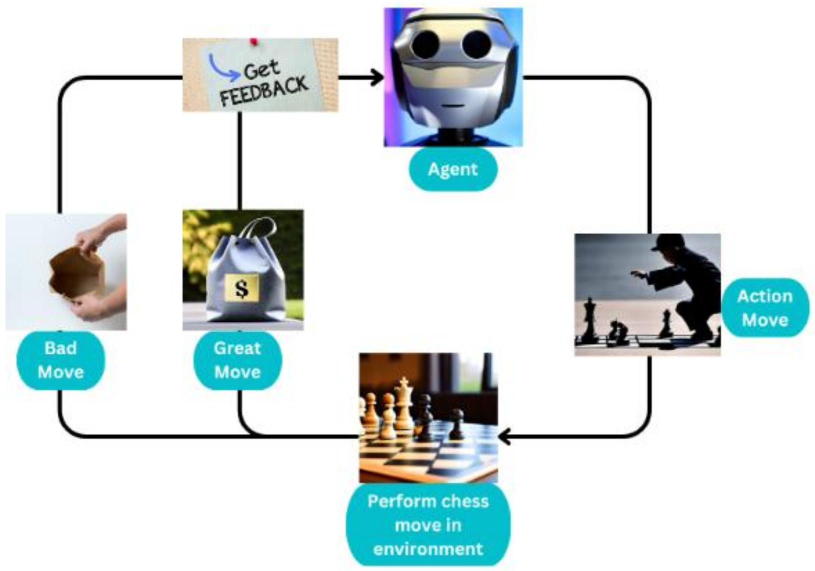


Fig: Illustration of chess game using ML

Step 2: *Choosing target function:* The next important step is choosing the target function. It means according to the knowledge fed to the algorithm the machine learning will choose Next-Move function which will describe what type of legal moves should be taken.

For example: While playing chess with the opponent, when opponent will play then the machine learning algorithm will decide what be the number of possible legal moves taken in order to get success.

Step 3: *Choosing Representation for Target function:* When the machine algorithm will know all the possible legal moves the next step is to choose the optimized move using any representation i.e. using linear Equations, Hierarchical Graph Representation, Tabular form etc. The Next-Move function will move the Target move like out of these move which will provide more success rate.

For Example: while playing chess machine have 4 possible moves, so the machine will choose that optimized move which will provide success to it.

Step 4: *Choosing Function Approximation Algorithm:* An optimized move cannot be chosen just with the training data. The training data had to go through with set of example and through these examples the training data will approximates which steps are chosen and after that machine will provide feedback on it.

For Example: When a training data of Playing chess is fed to algorithm so at that time it is not machine algorithm will fail or get success and again from that failure or success it will measure while next move what step should be chosen and what is its success rate.

Step 5: *Final Design:* The final design is created at last when system goes from number of examples, failures and success, correct and incorrect decision and what will be the next step etc.

Example: Deep-Blue is an intelligent computer which is ML-based won chess game against the chess expert Garry Kasparov, and it became the first computer which had beaten a human chess expert.

Questions:

*****END*****