

## **Chapter 01**

# **Background of Machine Learning**

## 1.1 What is Machine Learning?

Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy. Machine learning algorithm provides a way to learn machine just like human learn.

## 1.2 Broad Categories of Machine Learning Algorithm

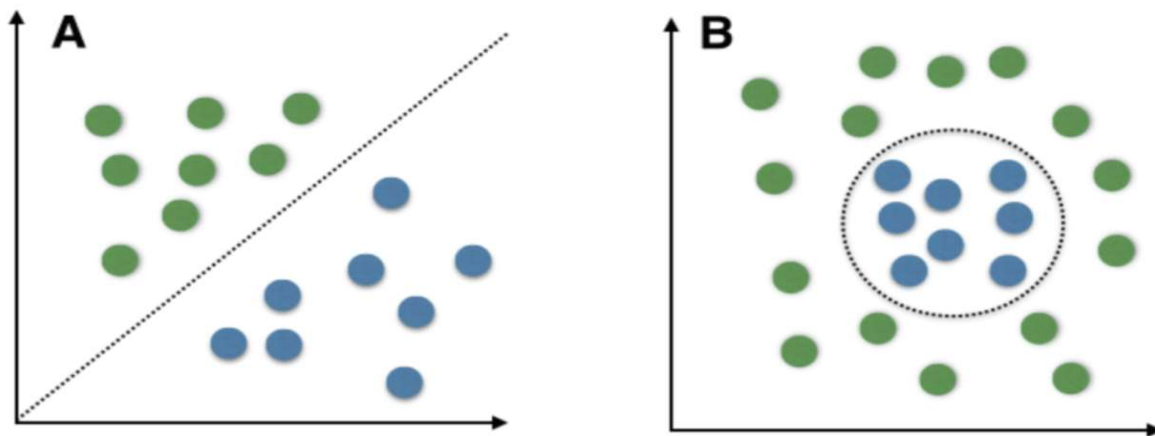
Generally three broad categories. These are

### 1.2.1 Linear machine learning algorithm

Linear decision boundary has been used for linear machine learning algorithm. Generally linear algorithm can solve linear classification and regression problems. But sometimes linear algorithms can be used for non-linear classification problems.

### 1.2.2 Non-linear machine learning algorithm

For non-linear classification and regression problems, non-linear classification algorithm has been used. Generally non-linear classification algorithm are more complicated than linear algorithms. For non-linear classification algorithm, non-linear decision boundary has been used or linear decision boundary has been used in such a way that multiple linear decision boundary can form nonlinear decision boundary.



*A: Linearly Separable Data B: Non-Linearly Separable Data*

Figure: Example of linear and non-linear decision boundary that produced by linear and non-linear classifier

### 1.2.3 Ensemble machine learning algorithm

A powerful and more advanced type of machine learning algorithm are ensemble algorithms. These are techniques that combine the predictions from multiple models in order to provide more accurate predictions.

## 1.4 Parametric and Non-parametric Machine Learning Algorithm

According to the parameter(s) that have to update weather it is fixed or variable, machine learning algorithm can be classified into parametric and non-parametric.

### 1.4.1 Parametric Model

A parametric model is one where we assume the 'shape' of the data, and therefore only have to estimate the coefficients of the model.

### 1.4.2 Non-parametric Model

A non-parametric model is one where we do not assume the 'shape' of the data, and we have to estimate the most suitable form of the model, along with the coefficients.

Difference between Parametric and Non-Parametric Methods are as follows:

Parametric model	Non-parametric model
It uses a fixed number of parameters to build the model.	It uses flexible number of parameters to build the model.
Considers strong assumptions about the data.	Considers fewer assumptions about the data.
Computationally faster	Computationally slower
Require lesser data	Require more data
Example – Logistic Regression & Naïve Bayes models	Example – KNN & Decision Tree models

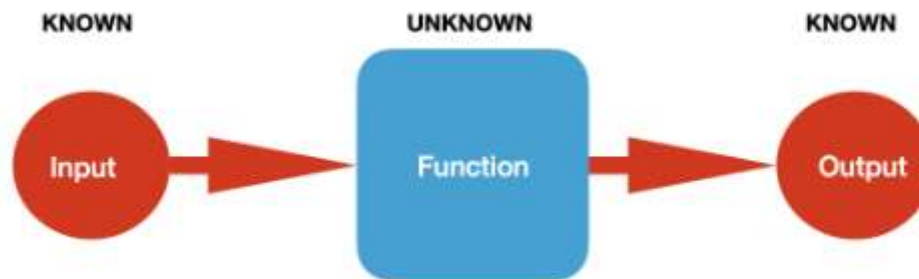
## 1.5 Supervised, Unsupervised and Semi-Supervised Learning

Based on the *nature of input data* that we provide to the machine learning algorithms, machine learning can be classified into 3 major categories.

1. **Supervised Learning**
2. **Unsupervised Learning**
3. **Semi-Supervised Learning**

### 1.5.1 Supervised Learning

Supervised Learning is a category in which we feed labeled data as input to the machine learning model.



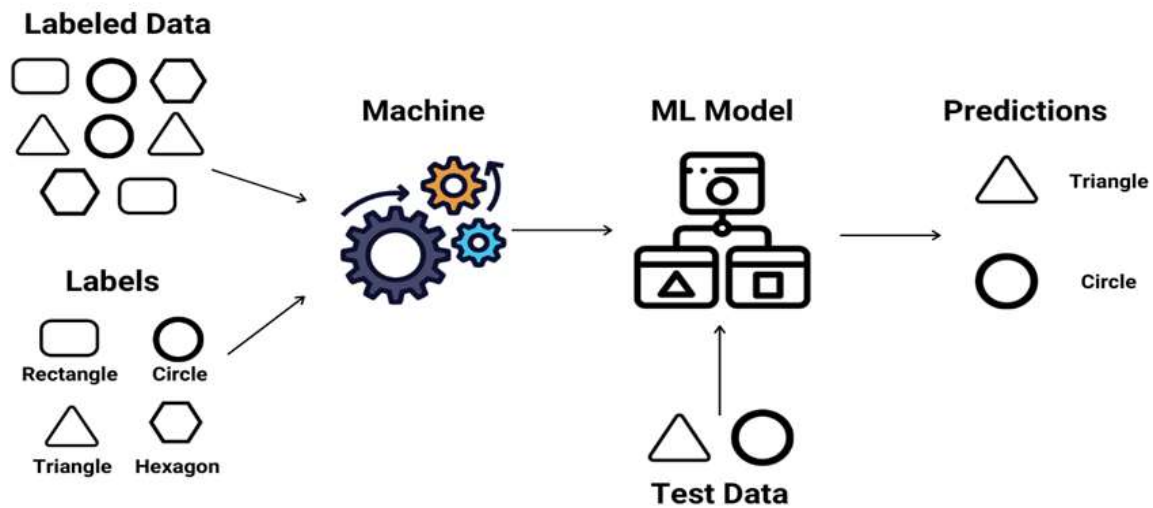
The input and output values are already known, and the machine learning algorithm learns the mapping function. Mathematically, for **Y** as the Output and **X** as the input, machine learning algorithms try to find the best mapping function **f** such that  $Y = f(X)$ .

If you observe closely, learning happens like some supervisor is supervising the process of the learning. We already know the answers; hence algorithms try to map the function so that the predicted answers must be close to the actual answers. Let's say the machine has learned a mapping function **f** predicting the values **Y'** for every **X**. Once the difference between predicted (**Y'**) and actual (**Y**) goes below a certain threshold, and it means model performance is not improving further and hence learning stops.

**Supervised learning can be further categorized as:**

**1. Classification:-** A classification problem is when the output variable is a category, such as rectangle, circle, triangle, or hexagon.

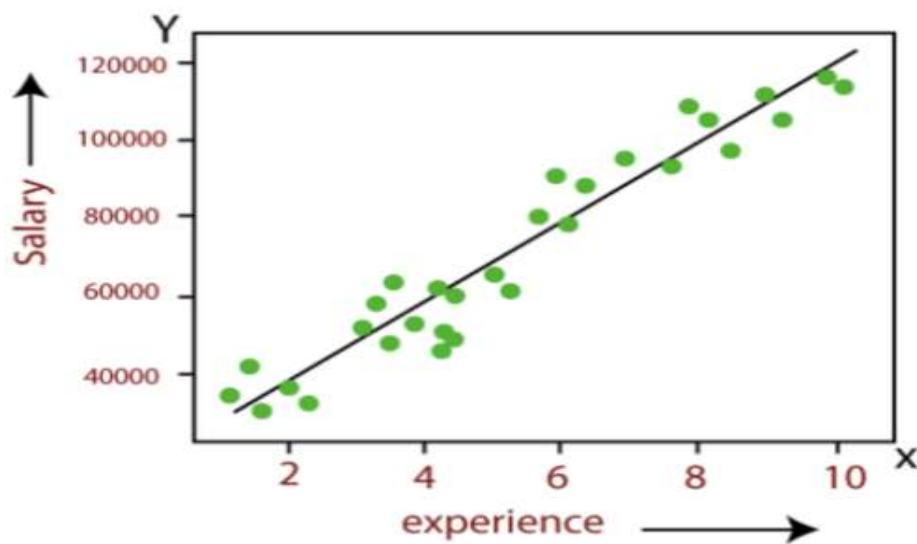
# Supervised Learning



**2. Regression:-** Regression is a supervised machine technique where the ultimate goal of the regression algorithm is to plot a best-fit line or a curve between the data.

Taking the example of the below image, there is the experience (in years) on the X-axis. For every experience, there is one salary (In per month Rupees) on the Y-axis. Green dots are the coordinates (X, Y) in the form of Input and Output data. The regression problem tries to find the continuous mapping function from input to output variables.

In the below image, if the order of the mapping function is fixed to 1, which is a linear function, the model will learn the black line shown in the image.

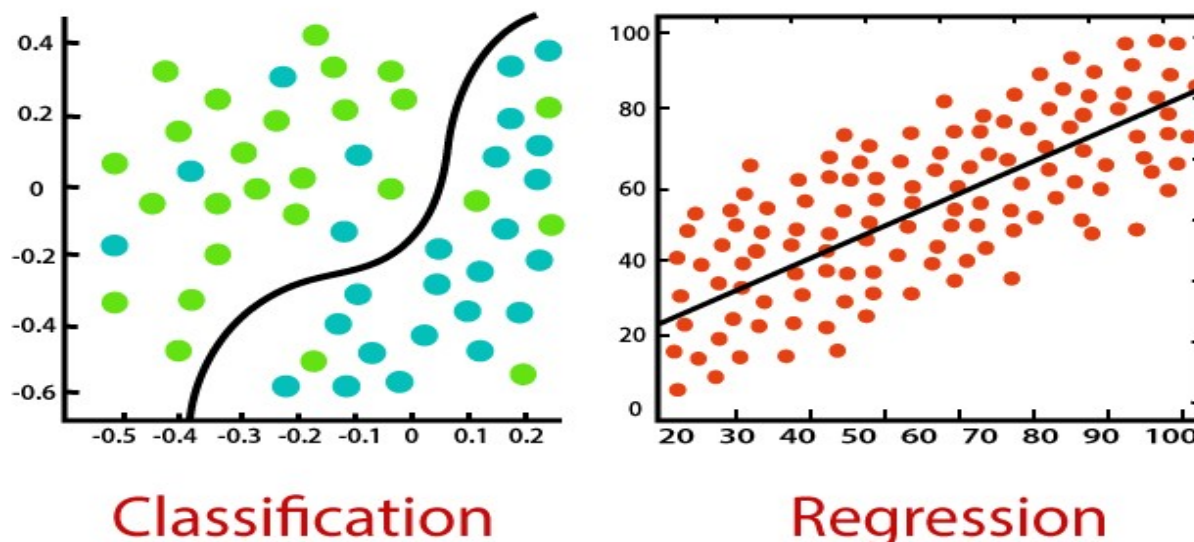


## Differences between Regression and Classification in Machine Learning

Regression and Classification algorithms are Supervised Learning algorithms. Both the both is how they are used for different machine learning problems.

The main algorithms are used for prediction in Machine learning and work with the labeled datasets. But the 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:



### 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:

Classification Algorithms can be further divided into the following types:

- Logistic Regression
- K-Nearest Neighbours
- Support Vector Machines (SVM)
- Kernel SVM
- Naïve Bayes
- Decision Tree Classification
- Random Forest Classification

## 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:

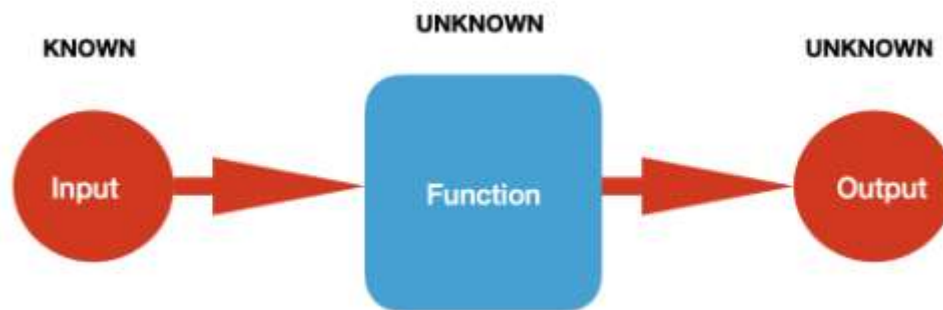
- Simple Linear Regression
- Multiple Linear Regression
- Polynomial Regression
- Support Vector Regression
- Decision Tree Regression
- Random Forest Regression

Regression Algorithm	Classification Algorithm
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.

### 1.5.2 Unsupervised Learning

Unsupervised learning is a category of machine learning in which we only have the input data to feed to the model but no corresponding output data.



Here, we know the value of input data, but *the output* and *the mapping function* both are unknown. In such scenarios, machine learning algorithms find the function that finds similarity among different input data instances and groups them based on the similarity index, which is the output of unsupervised learning.

In such learning, there is no supervision as there is no existence of output data. Hence they are called Unsupervised learning. Algorithms try to find the similarity between different instances of input data by themselves using a defined similarity index.

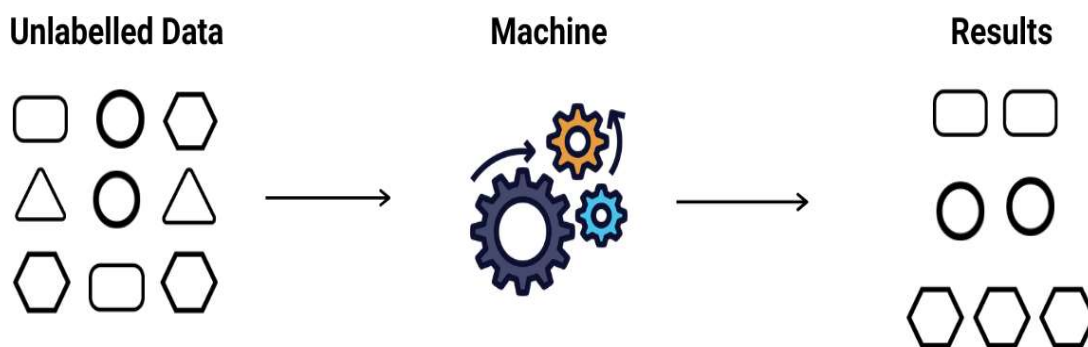
**Unsupervised Learning can further be categorized as:**

**1. Clustering (Unsupervised classification):-** A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behavior.



Taking the example of the below image, we have input data consisting of images of different shapes. Machine learning algorithms try to find the similarity among different images based on the color pixel values, size, and shapes and form the groups as outputs in which similar input instances lie.

## Unsupervised Learning



Example of Clustering Algorithms are

- Hierarchical clustering
- K-means clustering

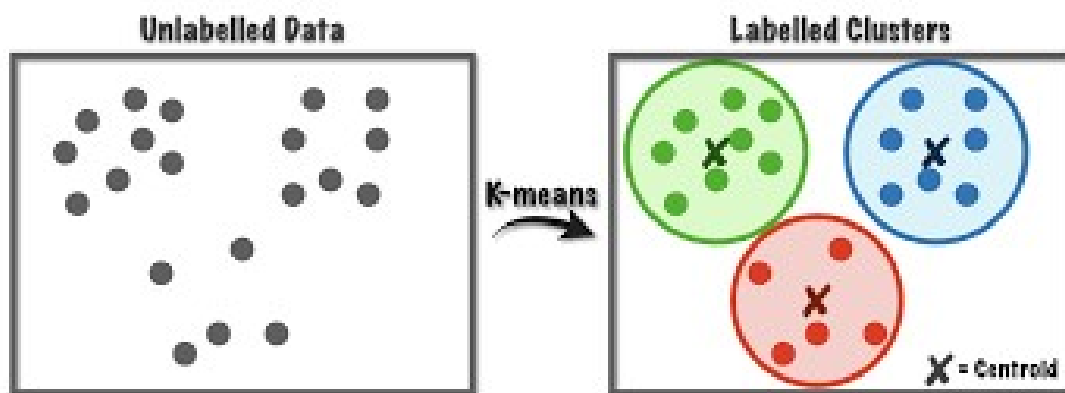


Figure: Example of K-means clustering

**2. Dimensionality Reduction:-** When the attributes of the data samples have more than three dimensions, there is no way to visualize the relationship among the attributes as we can not plot variables in more than 3 Dimensions. But without analyzing the input data, we can never be sure about the Machine Learning model's performance. To solve this purpose, there are **dimensionality reduction techniques** using which we bring down the total number of dimensions and analyze the data.

Example fo dimensioanlity reduction are

- Learning Vector Quantization (LVQ)
- Kohonen Self-Organizing Neural Networks

**3. Association:-** An association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy *A* also tend to buy *B*.



Taking the example of the below image, such learning is more about discovering rules that describe a large portion of the data. Customers who bought a banana also bought carrots, or Customers who bought a new house also bought new furniture.

**Some famous use cases of Unsupervised Learning are**

1. Market Segmentation
2. Fraud detection
3. Image Segmentation

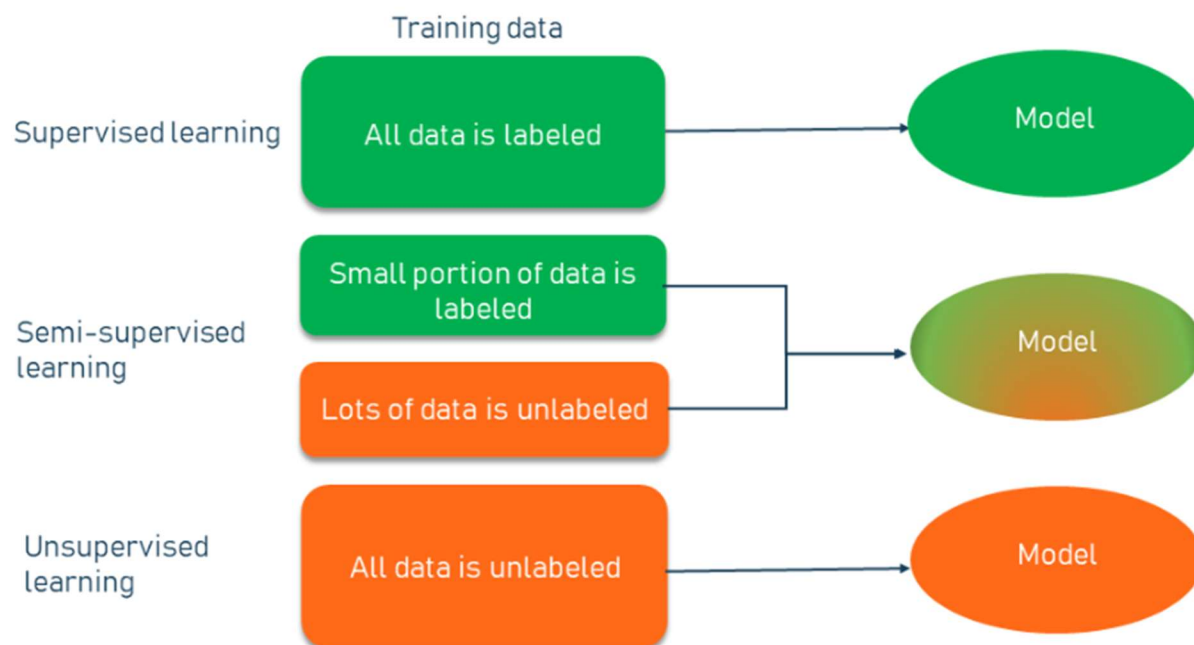
**Some Frequently used algorithms in Unsupervised Learning :**

1. K-means
2. Apriori Algorithm for learning association rule.
3. Principal Component Analysis

### 1.5.3 Semi-Supervised Learning

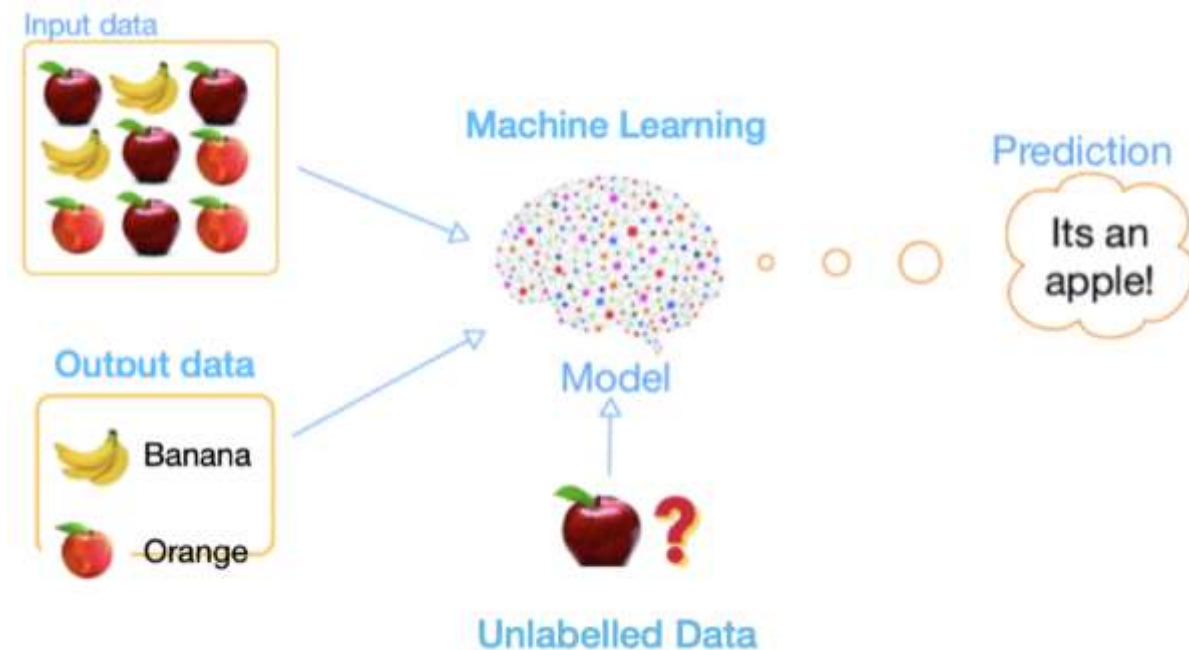
Semi-supervised learning is a category of machine learning in which we have input data, and only some of those input data are labeled as the output.

#### SUPERVISED LEARNING vs SEMI-SUPERVISED LEARNING vs UNSUPERVISED LEARNING



***Semi-supervised learning is partially supervised and partially unsupervised.***

Let's take one example from the below image to make it clear, suppose there is a bucket consisting of three fruits, apple, banana, and orange. Someone captured the image of all three but labeled only orange and banana images.



Nowadays, it has become a trend to capture a tremendous amount of data. Many big companies have already collected millions of Terrabytes of data and are still collecting. But labeling the collected data requires workforce and resources, and hence it's too expensive. And this is the main reason that many real-life databases fall in this category.

### **Some famous use cases of Semi-supervised Learning are:**

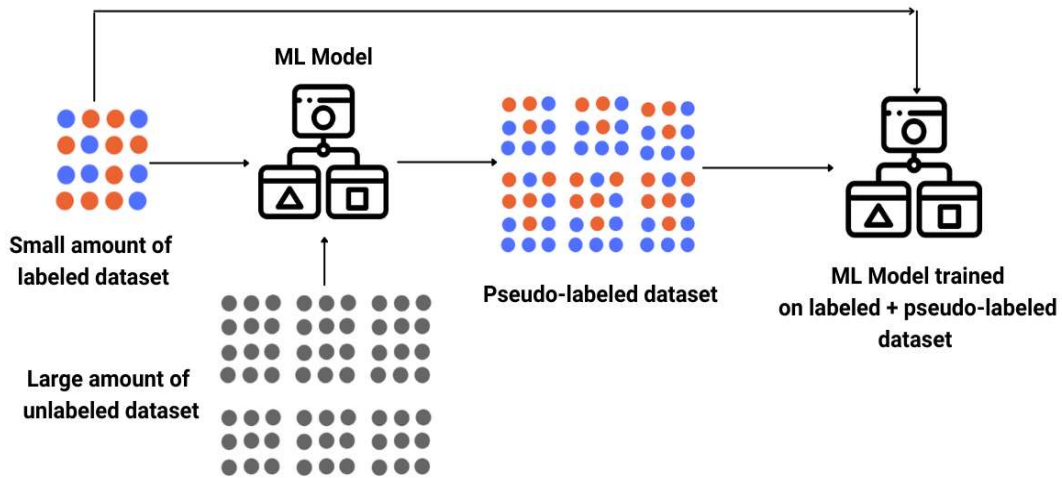
In such type of learning, one can use either.

Supervised learning on unlabeled data and use the predicted output as input for retraining other supervised learning models and test it on other unlabeled data.

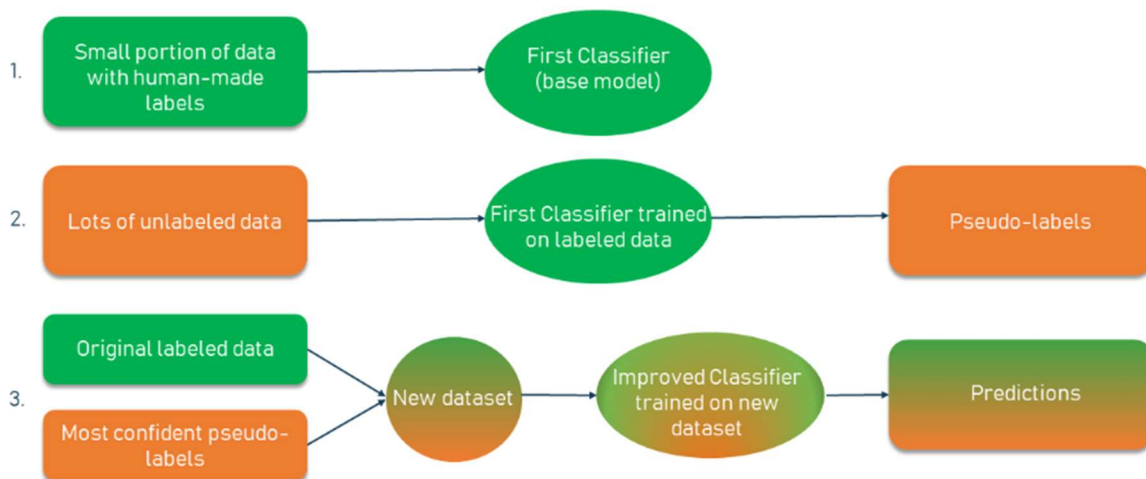
Or

Unsupervised learning to capture and learn the structure present in the data.

## Semi-supervised learning use-case



### SEMI-SUPERVISED SELF-TRAINING METHOD



For example, suppose there is a large chunk of data in the image above, and a small amount of labeled dataset is present. We can train the model using that small amount of labeled data and then predict on the unlabelled dataset. Prediction on an unlabelled dataset will attach the label with every data sample with little accuracy, termed as a Pseudo-labeled dataset. Now a new model can be trained with the mixture of the true-labeled dataset and pseudo-labeled dataset.

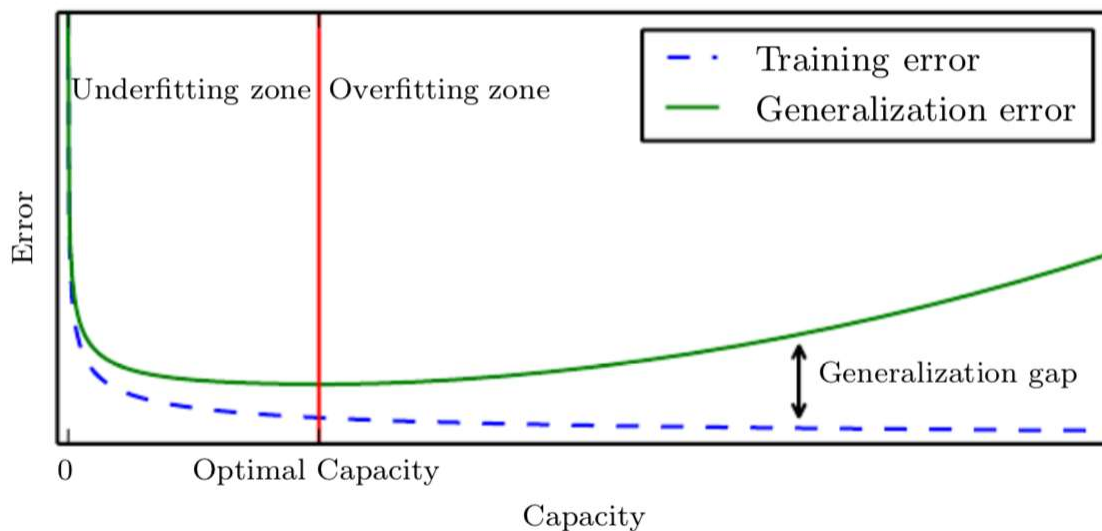
## 1.6 The Bias-Variance Trade-Off

Supervised machine learning algorithms can best be understood through the lens of the bias-variance trade-off. The prediction error for any machine learning algorithm can be broken down into three parts:

- Bias Error
- Variance Error
- Irreducible Error

The irreducible error cannot be reduced regardless of what algorithm is used. It is the error introduced from the chosen framing of the problem and may be caused by factors like unknown variables that influence the mapping of the input variables to the output variable.

### 1.6.1 Bias-Variance Trade-Off



**Fig.** Bias and Variance in typical neural network learning algorithm

#### 1.6.1.1 Bias Error?

You can see that the training error (blue dotted line) keeps on **decreasing**. In the initial phase, **it's too high (High Bias)**. Later, **it decreases (Low Bias)**.

**High Bias** means the model is not even fitting on the training data. So, we have to make the bias low.

#### How to lower the bias?

1. Increase the epochs (iterations)

2. Try a Bigger network

### 1.6.1.2 What is Variance Error?

The Variance is the difference between **validation error and training error**. In the figure, you can see that the gap between validation error and training error is increasing. That is, the variance is increasing (Overfitting).

#### What is the importance of variance?

Variance gives us the information about the generalization power of our model.

If the Variance is **high**, the model is not performing well on the validation set. We always want a **low variance**.

#### How to lower the variance?

1. Increase the training set data
2. Try Regularisation
3. Try a different Neural Network Architecture

*We always want a low bias and low variance.*

## 1.7 Overfitting and Underfitting

### 1.7.1 Generalization in Machine Learning

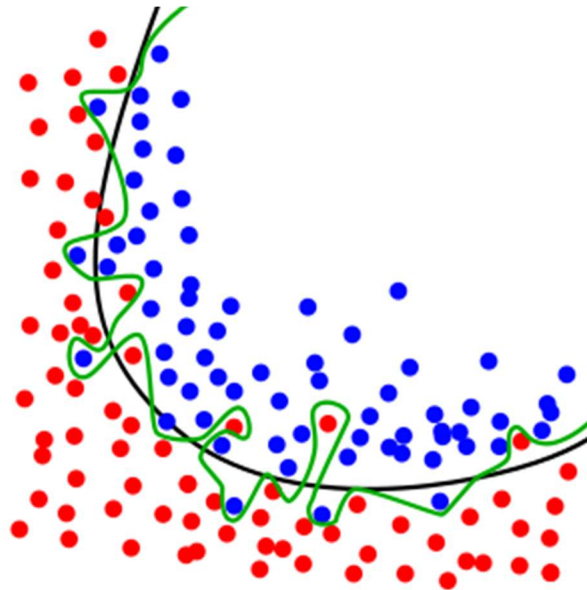
In Supervised learning, you train the machine using data which is well "**labeled**." It means some data is already tagged with the correct answer. A supervised learning algorithm learns from labeled training data, helps you to predict outcomes for unforeseen data. With training data, the outcome is already known. The predictions from the model and known outcomes are compared, and the model's parameters are changed until the two align. The point of training is to develop the model's ability to successfully **generalize**.

Generalization is a description of how well the concepts learned by a model apply to new data. Generalization is a term used to describe a model's ability to react to new data and how well the concepts learned by a machine learning model apply to specific examples not seen by the model when it was learning. That is, after being trained on a training set, a model can digest new data and make accurate predictions. **A model's ability to generalize is central to the success of a model.**



If a model has been trained too well on training data, it will be unable to generalize. It will make inaccurate predictions when given new data, making the model useless even though it is able to make accurate predictions for the training data. This is called **Overfitting**. Overfitting refers to a model that models the training data too well. Overfitting happens when a model learns the detail and noise in the training data to the extent that it negatively impacts the performance on the model on new data. This means that the noise or random fluctuations in the training data is picked up and learned as concepts by the model. The problem is that these concepts do not apply to new data and negatively impact the models ability to generalize. The inverse is also true.

**Underfitting** happens when a model has not been trained enough on the data. In the case of underfitting, it makes the model just as useless and it is not capable of making accurate predictions, even with the training data.



**Fig.** The green line represents an overfitted model and the black line represents a regularized model. While the green line best follows the training data, it is too dependent on that data and it is likely to have a higher error rate on new unseen data, compared to the black line.

Ideally, you want to select a model at the sweet spot between underfitting and overfitting. This is the goal, but is very difficult to do in practice. To understand this goal, we can look at the performance of a machine learning algorithm over time as it is learning a training data. We can plot both the skill on the training data and the skill on a test dataset we have



held back from the training process. Over time, as the algorithm learns, the error for the model on the training data goes down and so does the error on the test dataset. If we train for too long, the performance on the training dataset may continue to decrease because the model is overfitting and learning the irrelevant detail and noise in the training dataset. At the same time the error for the test set starts to rise again as the model's ability to generalize decreases. The sweet spot is the point just before the error on the test dataset starts to increase where the model has good skill on both the training dataset and the unseen test dataset. You can perform this experiment with your favorite machine learning algorithms. This is often not useful technique in practice, because by choosing the stopping point for training using the skill on the test dataset it means that the testset is no longer unseen or a standalone objective measure. Some knowledge (a lot of useful knowledge) about that data has leaked into the training procedure.

### 1.7.2 How to Detect Overfitting and Underfitting

A key challenge with Overfitting and Underfitting with machine learning in general, is that we can't know how well our model will perform on new data until we actually test it.

So to better demonstrate it we can use an example.

Let's say we want to predict if a student will land a job interview based on his/her resume. Now, assume we train a model from a dataset of 10,000 resumes and their outcomes. Next, we try the model out on the original dataset, and it predicts outcomes with 99% accuracy... wow!

But now comes the bad news. When we run the model on a new “**unseen**” dataset of resumes, we only get 50% accuracy... uh-oh!

Our model doesn't generalize well from our training data to unseen data.

Now we know that our model is **Overfitted**.

But if we try the model out on the original dataset, and it predicts outcomes with 55% accuracy or something.

Then our model is **Underfitted**.

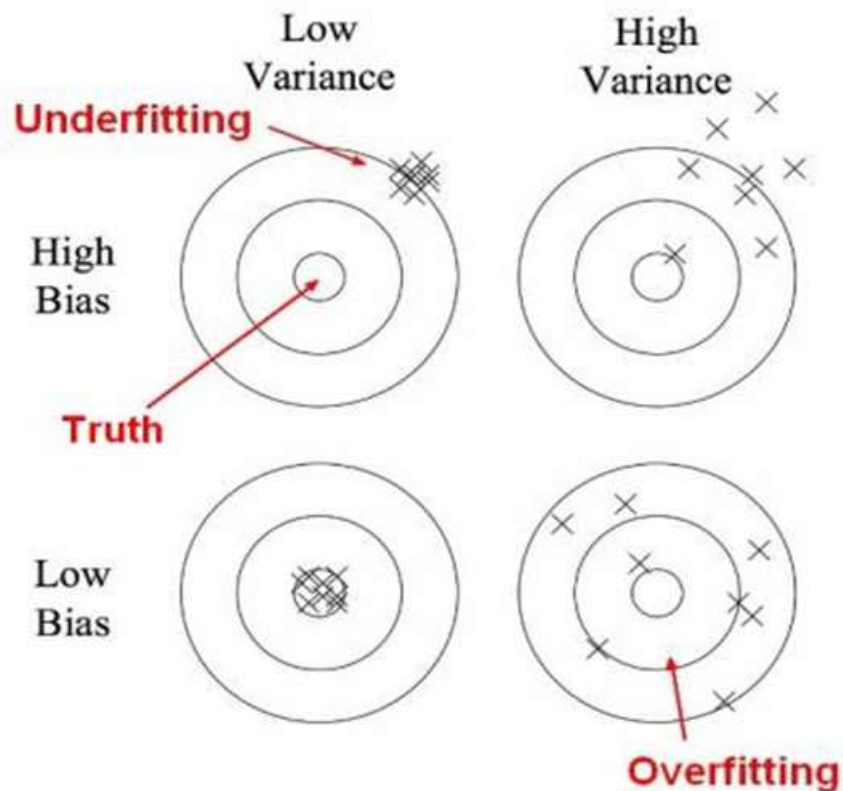
In brief,

- **Overfitting:** Good performance on the training data, poor generalization to other data.
- **Underfitting:** Poor performance on the training data and poor generalization to other data.

### 1.7.3 Reasons for Overfitting and Underfitting

**Overfitting** happens when the size of training data used is not enough, or when our model captures the noise along with the underlying pattern in data. It happens when we train our model a lot over noisy dataset. These models have **low bias** and **high variance**. These models are very complex like Decision trees which are prone to overfitting.

While **Underfitting** happens when a model is unable to capture the underlying pattern of the data. These models usually have **high bias** and **low variance**. It happens when we have very less amount of data to build an accurate model or when we try to build a linear model with a nonlinear data. Also, these kind of models are very simple to capture the complex patterns in data like Linear and logistic regression. It can also happen when the size of training data used is not enough.



**Fig.** Bias-Variance trade-off for overfitting and underfitting

### 1.7.4 How to Prevent Overfitting and Underfitting

We want to select a model at the sweet spot between underfitting and overfitting. This is the goal, but is very difficult to do in practice. Detecting overfitting and underfitting is useful, but it doesn't solve the problem. Fortunately, you have several options to try.

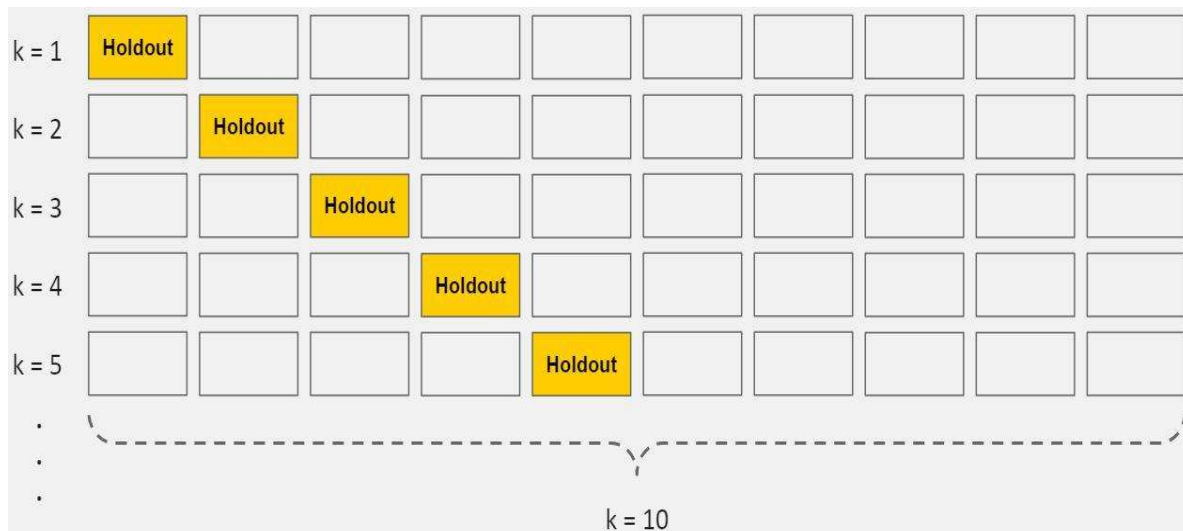
**Here are a few of the most popular solutions for Overfitting:**

- **Resampling technique to estimate model accuracy (i.e, Cross-validation)**

Cross-validation is a powerful preventative measure against overfitting. The idea is clever: Use your initial training data to generate multiple mini train-test splits. Use these splits to tune your model.

In standard k-fold cross-validation, we partition the data into k subsets, called folds. Then, we iteratively train the algorithm on k-1 folds while using the remaining fold as the test set (called the “holdout fold”).

Cross-validation allows you to tune hyperparameters with only your original training set. This allows you to keep your test set as a truly unseen dataset for selecting your final model.



- **Train with more data**

It won't work every time, but training with more data can help algorithms detect the signal better. Of course, that's not always the case. If we just add more noisy data, this technique won't help. That's why you should always ensure your data is clean and relevant.

- **Remove features**

Some algorithms have built-in feature selection. For those that don't, you can manually improve their generalizability by removing irrelevant input features. An interesting way to do so is to tell a story about how each feature fits into the model. This is

like the data scientist's spin on software engineer's [rubber duck debugging](#) technique, where they debug their code by explaining it, line-by-line, to a rubber duck.

If anything doesn't make sense, or if it's hard to justify certain features, this is a good way to identify them.

- **Regularization**

Regularization refers to a broad range of techniques for artificially forcing your model to be simpler.

The method will depend on the type of learner you're using. For example, you could prune a decision tree, use dropout on a neural network, or add a penalty parameter to the cost function in regression. Oftentimes, the regularization method is a hyperparameter as well, which means it can be tuned through cross-validation.

- **Ensembling**

Ensembles are machine learning methods for combining predictions from multiple separate models. There are a few different methods for ensembling, but the two most common are:

- **Bagging** attempts to reduce the chance overfitting complex models.
- **Boosting** attempts to improve the predictive flexibility of simple models.

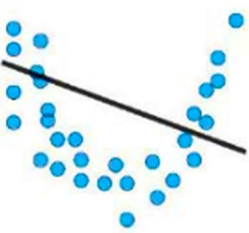


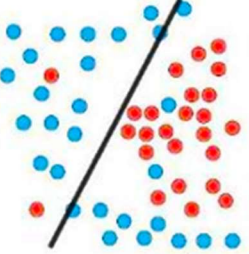
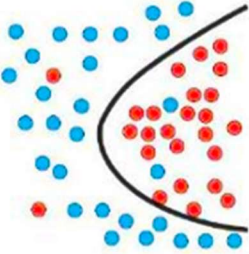
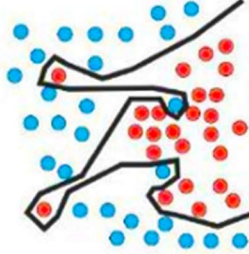
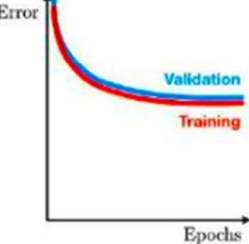
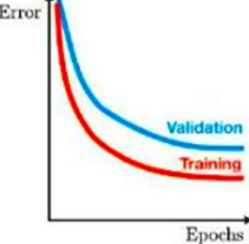
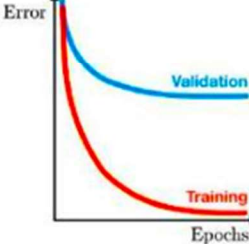
**Now for the most popular solutions for Underfitting:**

- **Increase model complexity**

As model complexity increases, performance on the data used to build the model (training data) improves.

- **Increase number of features, performing feature engineering**
- **Remove noise from the data**
- **Train with more data**

It won't work every time, but training with more data can help algorithms detect the signal better. Of course, that's not always the case. If we just add more noisy data, this technique won't help. That's why you should always ensure your data is clean and relevant.

	Under-fitting	Optimal-fitting	Over-fitting
Regression			
Classification			
Deep learning			
Remedies	<ul style="list-style-type: none"> <li>- Complexify model</li> <li>- Add More features</li> <li>- Longer train required</li> </ul>	Best fit (Nothing is required)	<ul style="list-style-type: none"> <li>- Regularize</li> <li>- Get more data</li> </ul>

**Fig.** Over-fitting and Under-fitting senerio possible remedies