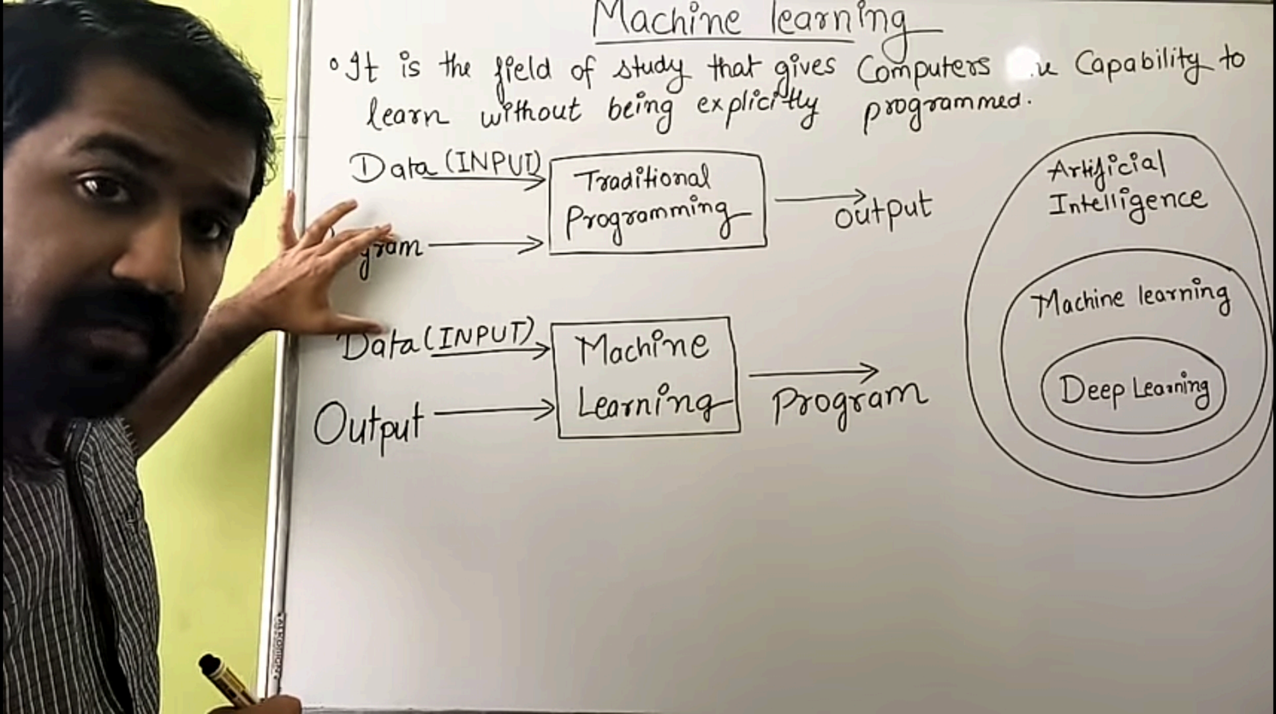
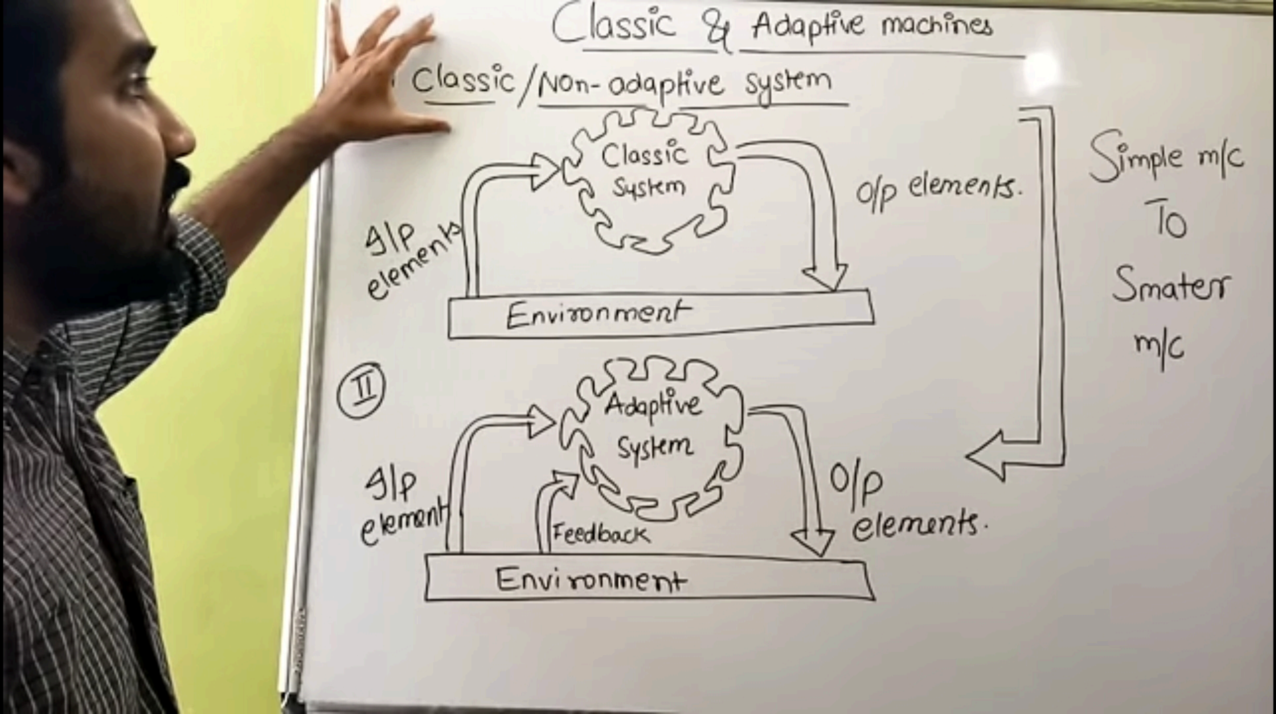
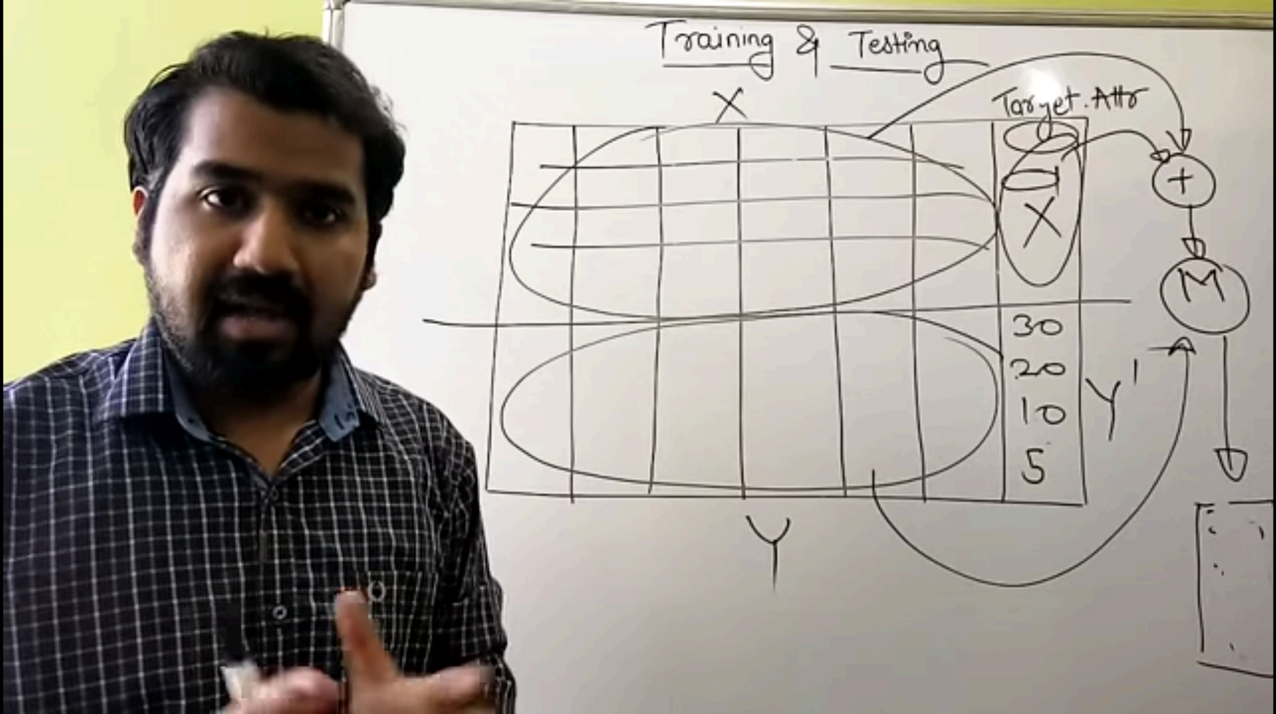
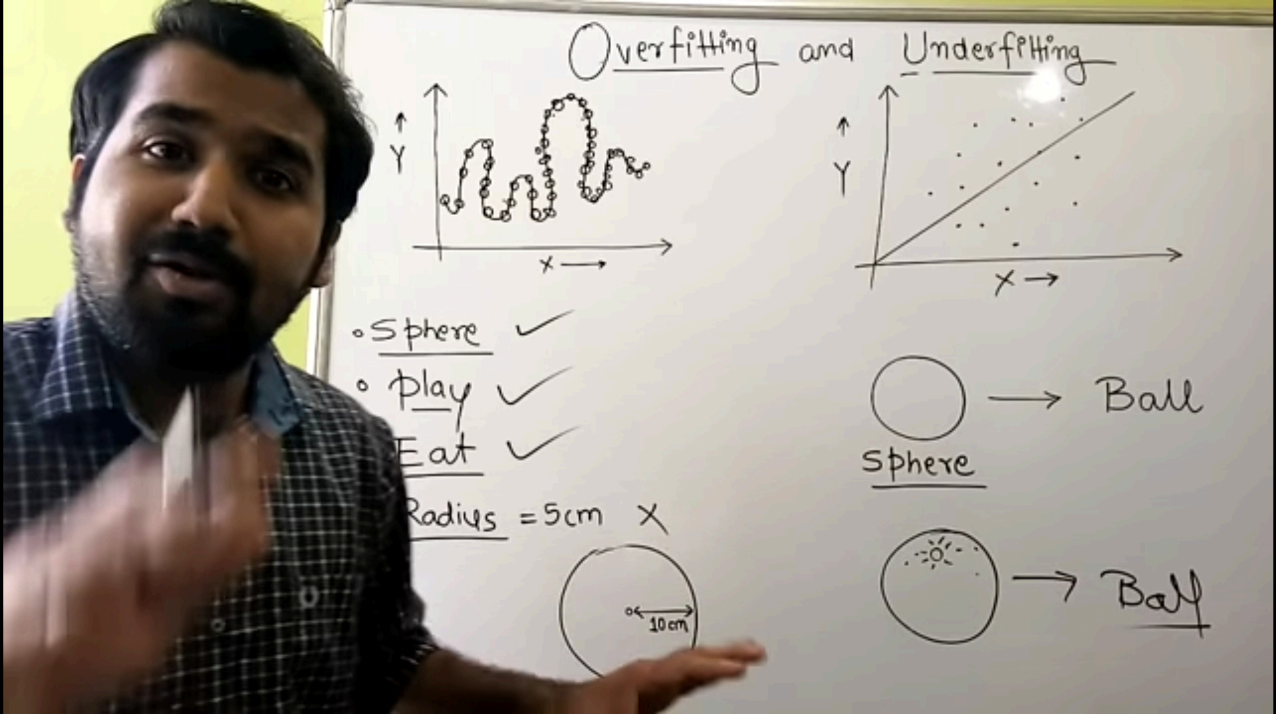

Machine learning
• It is the field of study that gives Computers Capability to learn without being explicitly programmed.
Data (INPUT)
gram
Traditional Programming
output
Data (INPUT)
Output
Machine Learning
Program
Artificial Intelligence
Machine learning
Deep Learning


Classic & Adaptive machines
Classic/Non-adaptive system
Classic System
i/p elements
o/p elements.
Environment
II
Adaptive System
i/p element
Feedback
o/p elements.
Environment
Simple m/c
To
Smater
m/c



Training & Testing
X
Target. Attr
+
M
30
20
10
5
Y'
Y


Overfitting and Underfitting
Y
X
Y
X
Sphere
Play
Eat
Radius = 5cm X
10 cm
Ball
Sphere
Ball

management. Furthermore, *the global machine learning (ML) market is expected to grow* [8] *from $21.17 billion in 2022 to $209.91 billion by 2029, at a CAGR of 38.8% in forecast period*.

Components of machine learning

There are tens of thousands of machine learning algorithms and hundreds of new algorithms are developed every year.

Every machine learning algorithm has three components:

- **Representation**: This implies how to represent knowledge. Examples include decision trees, sets of rules, instances, graphical models, neural networks, support vector machines, model ensembles and others.
- **Evaluation**: This is the way to evaluate candidate programs (hypotheses). Examples include accuracy, prediction and recall, squared error, likelihood, posterior probability, cost, margin, entropy k-L divergence and others.
- **Optimization**: Last but not the least, optimization is the way candidate programs are generated and is known as the search process. For example, combinatorial optimization, convex optimization, and constrained optimization.

All machine learning algorithms are a combination of these three components and a framework for understanding all algorithms.

Machine learning classification

Machine Learning algorithms can be classified into:

1. Supervised Algorithms:
   - Linear Regression,
   - Logistic Regression,
   - KNN classification,
   - Support Vector Machine (SVM),
   - Decision Trees,
   - Random Forest,
   - Naive Bayes' theorem
2. Unsupervised Algorithms: K Means Clustering
3. Reinforcement Algorithm

Let us dig a bit deeper into these machine learning basics algorithms.

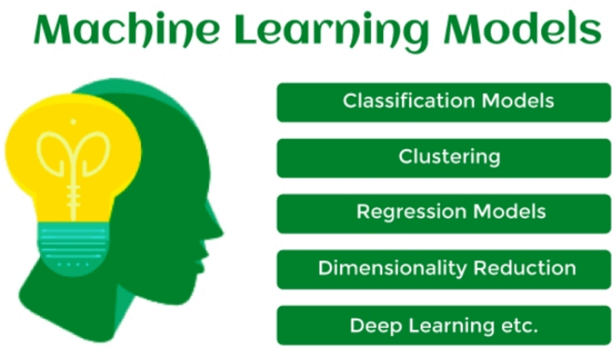
Supervised Machine Learning Algorithms

In this type of algorithm, the data set on which the machine is trained consists of labelled data or simply said, consists of both the input parameters as well as the required output.

Let's take the previous example of facial recognition and once we have identified the

Machine Learning Models

A machine learning model is defined as a mathematical representation of the output of the training process. Machine learning is the study of different algorithms that can improve automatically through experience & old data and build the model. A machine learning model is similar to computer software designed to recognize patterns or behaviors based on previous experience or data. The learning algorithm discovers patterns within the training data, and it outputs an ML model which captures these patterns and makes predictions on new data.

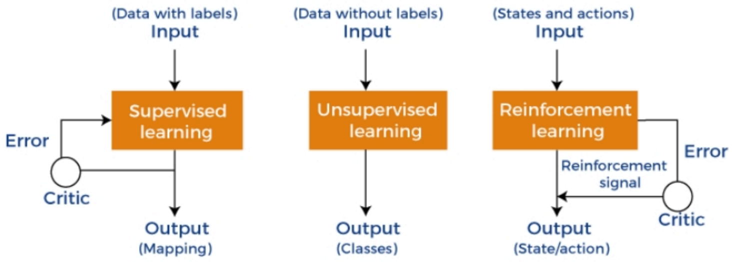


different business goals and data sets.

Classification of Machine Learning Models:

Based on different business goals and data sets, there are three learning models for algorithms. Each machine learning algorithm settles into one of the three models:

- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning



Supervised Learning is further divided into two categories:

- Classification
- Regression

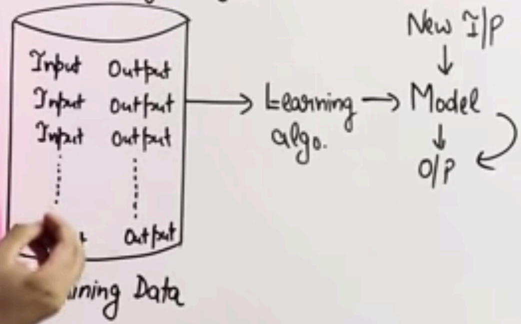
Unsupervised Learning is also divided into below categories:

- Clustering
- Association Rule
- Dimensionality Reduction

1. Supervised Machine Learning Models

"Supervised Learning"

- → Training Data
- → Both Inputs & outputs
- → Classification
- → Naive Bayes algo.

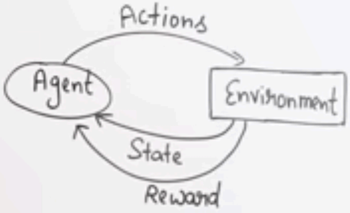


"Unsupervised Learning"

- → Only Inputs
- → Clustering
- → K-Mean

"Reinforcement Learning"

- → Reward / Penalty
- → Q-Learning



Machine Learning is all about using the right *features* to build the right *models* that achieve the right *tasks*.

Features : the workhorses of Machine Learning.

Models : the output of Machine Learning.

Tasks : the problems that can be solved with Machine Learning.

> *Features determine a large part of the success of an ML application, because a model is only as good as its features.*

In this article, our main focus is gonna be on the **MODELS.** Let's get on with our modeling trip...✈️

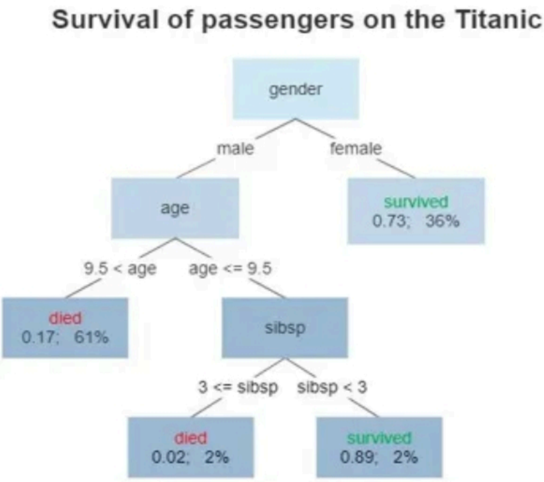
Models are the central concept in machine learning as they are what one learns from data in order to solve a given task. There is a huge variety of machine learning models available. This is particularly due to the omnipresence of tasks that machine learning aims to solve. The 3 most common groups of models that we see are : ***Logical***, ***Geometric*** and ***Probabilistic*** Models. Let's have a thorough look at each category of models.

LOGICAL MODELS

"Logical" because models of this kind can easily be translated into ***rules*** that humans can understand, such as ., *if lottery = 1 then class = Y = spam.* Such rules are easily arranged in a tree structure, which we refer to as a ***feature tree***. The idea of such a tree is that features are used to iteratively partition the instance space. The leaves of the tree therefore correspond to rectangular areas in the instance space (or hyperrectangles, more generally) which we will call instance space segments, or segments for short. Depending on the task that we are solving, we can label the leaves with a class, a probability, a real value and so on.

> *Feature trees whose leaves are labelled with classes are commonly called* **decision trees.**

A simple logical model is shown below:



Source — Wikipedia

A tree showing survival of passengers on the Titanic(“sibsp” is the number of spouses or siblings aboard). The figures under the leaves show the probability of survival and the percentage of observations in the leaf. Summarizing: Your chances of survival were good if you were (i) a female or (ii) a male younger than 9.5 years with strictly less than 3 siblings.

To gain a deeper understanding of logical models, we need to understand the concept of *concept learning,* which I’ll be discussing in my upcoming blogs. Wow! We’re just two steps away... Can’t wait more (๑>ᴗ<๑). Let’s quickly deep dive into our Geometric Models.

GEOMETRIC MODELS

Before learning what a geometric model is?, what are it’s types?, How they work? etc. Let’s first learn ***what an “instance space” actually is??.***

> *An* ***instance space*** *is the set of all possible or describable instances, whether they are present in our data set or not. Usually this set has some geometric structure. For instance, if all features are numerical, then we can use each feature as a coordinate in a*

> *An* ***instance space*** *is the set of all possible or describable instances, whether they are present in our data set or not. Usually this set has some geometric structure. For instance, if all features are numerical, then we can use each feature as a coordinate in a Cartesian coordinate system.*

Geometric models/feature learning is a technique of combining machine learning and computer vision to solve visual tasks. These models define similarity by considering the geometry of the instance space. Here, features could be described as points in two dimensions (x- and y-axis) or a three dimensional space (x, y, and z). Even when features are not intrinsically geometric, they could be modeled in a geometric manner. However, it is important to remember that the Cartesian instance space has as many coordinates as there are features, which can be tens, hundreds, thousands or even more. Such high-dimensional spaces are hard to imagine, but common in machine learning. Geometric concepts that may apply to higher-dimensional spaces are often prefixed with "hyper-": for example, a decision boundary of infinite dimensions is called a ***hyperplane***. A major advantage of geometric classifiers is that they are easy to visualize as long as we stick to two or three dimensions.

Geometric models are basically of two types :-

1. A geometric model that is constructed directly in instance space, using geometric concepts such as lines or planes are used to segment the instance space known as **Linear Models.**
2. A geometric model that uses distance as a metric to represent the similarity between the instances is known as **Distance based Models.** The distance metrics commonly used are, ***Euclidean, Minkowski, Manhattan,*** and ***Mahalanobis.***

Linear Models

To classify the instances, linear model uses the following equation, $f(x) = a + bx$, if x and f(x) are scalars and If x = (x1, . . .,xd) is a vector and f (x) is a scalar, then f is of the form *f(x)* =

Linear Models

To classify the instances, linear model uses the following equation, $f(x) = a + bx$, if x and f(x) are scalars and If $x = (x1, \ldots, xd)$ is a vector and f (x) is a scalar, then f is of the form $f(x) = a + b1x1 + \ldots + bd\ xd = a + b \cdot x$ with $b = (b1, \ldots, bd)$. The equation f (x) = 0 defines a plane in Rd perpendicular to the normal vector b.

Linear models exist for all predictive tasks, including classification, probability estimation and regression. For example.,

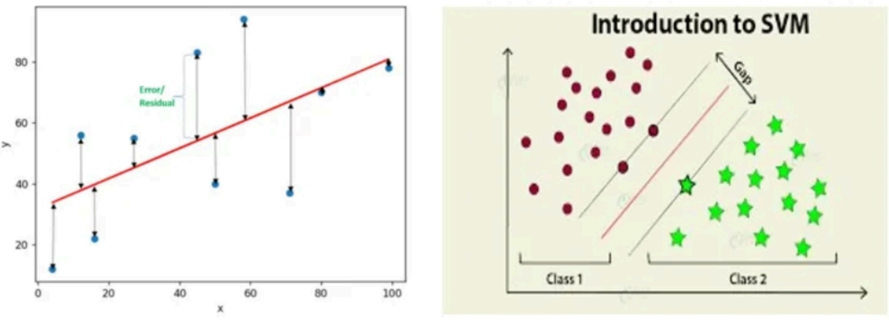
a) Hours spent studying Vs. Marks scored by students

b) Amount of rainfall Vs. Agricultural yield

c) Electricity usage Vs. Electricity bill

d) Suicide rates Vs. Number of stressful people

There are different linear models like the least-squares method (a mathematical regression analysis), Support Vector Machine (which best segregates two or more classes using a hyper-plane).



Least Squares (left) and SVM (right)

Distance Based Models

A very useful geometric concept in machine learning is the notion of distance. If the distance between two instances is small then the instances are similar in terms of their feature

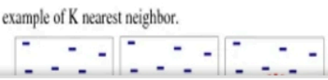
Distance Based Models

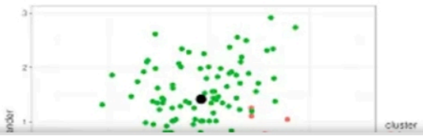
A very useful geometric concept in machine learning is the notion of distance. If the distance between two instances is small then the instances are similar in terms of their feature values, and so nearby instances would be expected to receive the same classification or belong to the same cluster. In a Cartesian coordinate system, distance can be measured by *Euclidean distance*, which is the square root of the sum of the squared distances along each coordinate.

To classify a new instance, we retrieve from memory the most similar training instance (i.e., the training instance with smallest Euclidean distance from the instance to be classified), and simply assign that training instance's class. This classifier is known as the ***nearest-neighbor*** classifier. Endless variations on this simple yet powerful theme exist: we can retrieve the k most similar training instances and take a vote (***k-nearest neighbor***).

Consequently, we can use the mean of a set of nearby points as a representative exemplar for those points. Suppose we want to cluster our data into K clusters, and we have an initial guess of how the data should be clustered. We then calculate the means of each initial cluster, and reassign each point to the nearest cluster mean. We repeat these two steps (calculating the cluster means and reassigning points to clusters) until no change occurs. This clustering algorithm, which is called ***K-means***.

Another distance-based clustering is the ***Hierarchical clustering***, algorithm that builds hierarchy of clusters. This algorithm starts with all the data points assigned to a cluster of their own. Then two nearest clusters are merged into the same cluster. In the end, this algorithm terminates when there is only a single cluster left.





A probability model/method is based on the theory of probability, or the fact that randomness play a role in predicting future events.

Let X denote the variables we know about, e.g., our instance's feature values; and let Y denote the target variables we're interested in, e.g., the instance's class. The key question in machine learning is how to model the relationship between X and Y. The statistician's approach is to assume that there is some underlying random process that generates the values for these variables, according to a well-defined but unknown probability distribution.

Since X is known for a particular instance but Y may not be, we are particularly interested in the conditional probabilities P(Y |X) where we predict the value of Y based on X. Naïve Bayes is an example of Probabilistic models, which follows Bayes theorem.

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

Bayes Theorem

P(Y/X) = Posterior probability (probability of hypothesis is true given the evidence)

P(X/Y) = Likelihood ratio (probability of seeing the evidence if the hypothesis is true)

P(Y) = Class Prior probability (probability of hypothesis is true, before any evidence is present)

P(X) = Predictor Prior probability (probability of observing the evidence)

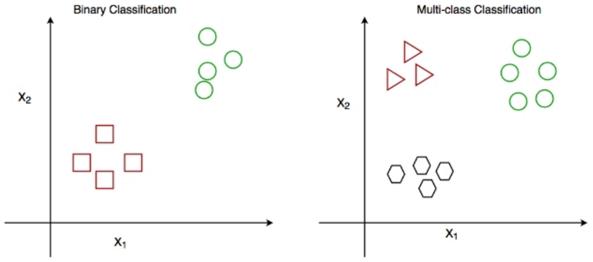
We now have some idea on what a probabilistic model looks like. In many cases this will be a matter of estimating the model parameters from data, which is usually achieved by straightforward counting

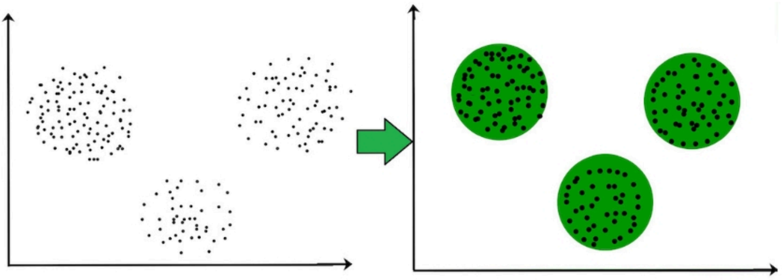
ML | Classification vs Clustering

Prerequisite: Classification and Clustering

As you have read the articles about classification and clustering, here is the difference between them.

Both Classification and Clustering is used for the categorization of objects into one or more classes based on the features. They appear to be a similar process as the basic difference is minute. In the case of Classification, there are predefined labels assigned to each input instance according to their properties whereas in clustering those labels are missing.

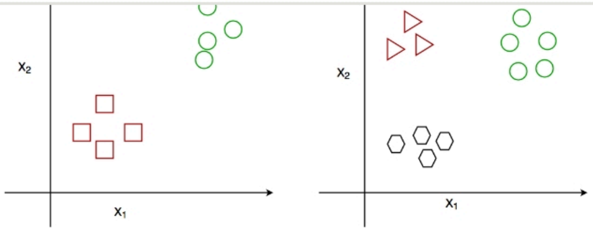


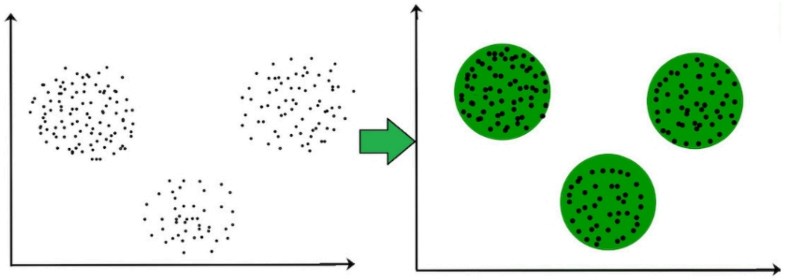


Comparison between Classification and Clustering:

Parameter	CLASSIFICATION	CLUSTERING
Type	used for supervised learning	used for unsupervised learning
Basic	process of classifying the input instances based on their corresponding class labels	grouping the instances based on their similarity without the help of class labels
Need	it has labels so there is need of training and testing dataset for verifying the model created	there is no need of training and testing dataset
Complexity	more complex as compared to clustering	less complex as compared to classification
Example Algorithms	Logistic regression, Naive Bayes classifier, Support vector machines, etc.	k-means clustering algorithm, Fuzzy c-means clustering algorithm, Gaussian (EM) clustering algorithm, etc.

Differences between Classification and Clustering





Comparison between Classification and Clustering:

Parameter	CLASSIFICATION	CLUSTERING
Type	used for supervised learning	used for unsupervised learning
Basic	process of classifying the input instances based on their corresponding class labels	grouping the instances based on their similarity without the help of class labels
Need	it has labels so there is need of training and testing dataset for verifying the model created	there is no need of training and testing dataset
Complexity	more complex as compared to clustering	less complex as compared to classification
Example Algorithms	Logistic regression, Naive Bayes classifier, Support vector machines, etc.	k-means clustering algorithm, Fuzzy c-means clustering algorithm, Gaussian (EM) clustering algorithm, etc.

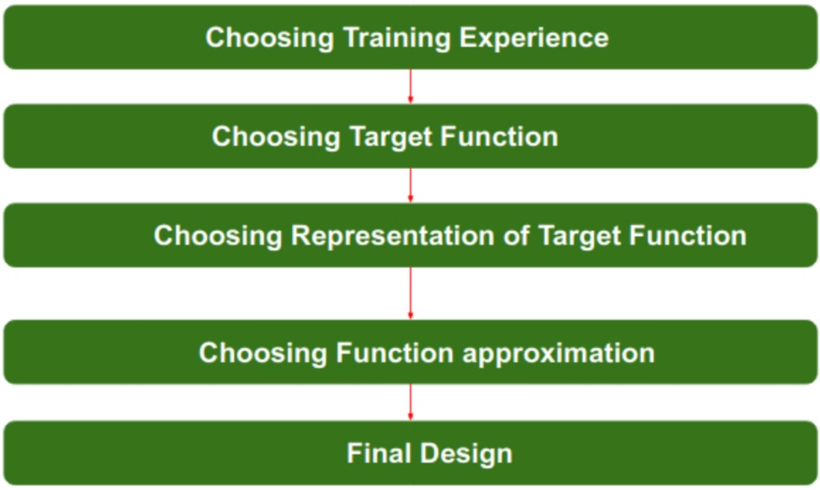
Differences between Classification and Clustering

1. Classification is used for supervised learning whereas clustering is used for unsupervised learning.
2. The process of classifying the input instances based on their corresponding class labels is known as classification whereas grouping the instances based on their similarity without the help of class labels is known as clustering.
3. As Classification have labels so there is need of training and testing dataset for verifying the model created but there is no need for training and testing dataset in clustering.
4. Classification is more complex as compared to clustering as there are many levels in the classification phase whereas only grouping is done in clustering.
5. Classification examples are Logistic regression, Naive Bayes classifier, Support vector machines, etc. Whereas clustering examples are k-means clustering algorithm, Fuzzy c-means clustering algorithm, Gaussian (EM) clustering algorithm, etc.

Article Tags : Difference Between Machine Learning

- **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”

Steps for Designing Learning System are:



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

Questions to Ask During a Feasibility Study

I like to use the following template when doing a feasibility study for a predictive model.

1. **Training Data** — Does training data need to be collected? If so, how much time and money will it cost?

2. **Predictive Features** — According to domain experts, what factors are likely to predict the target variable? Is that data accessible to you?

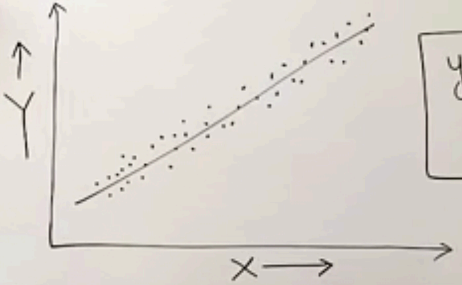
3. **Data Sources** — What data sources will you need to gain access to? If internal, do you have support from data engineers? If external, how much will vendor data cost?

4. **Production** — What is the level of effort to develop, deploy, and maintain your model in production?

Training Data

Linear Regression

- dependent Variable is continuous in nature.



$$y = 0.9 + 1.2x_1 + 2x_2 + 4x_3 + 1x_4$$

- Simple linear equation

$$y = \alpha_0 + \alpha_1 x_1$$

$$y = c + mx$$

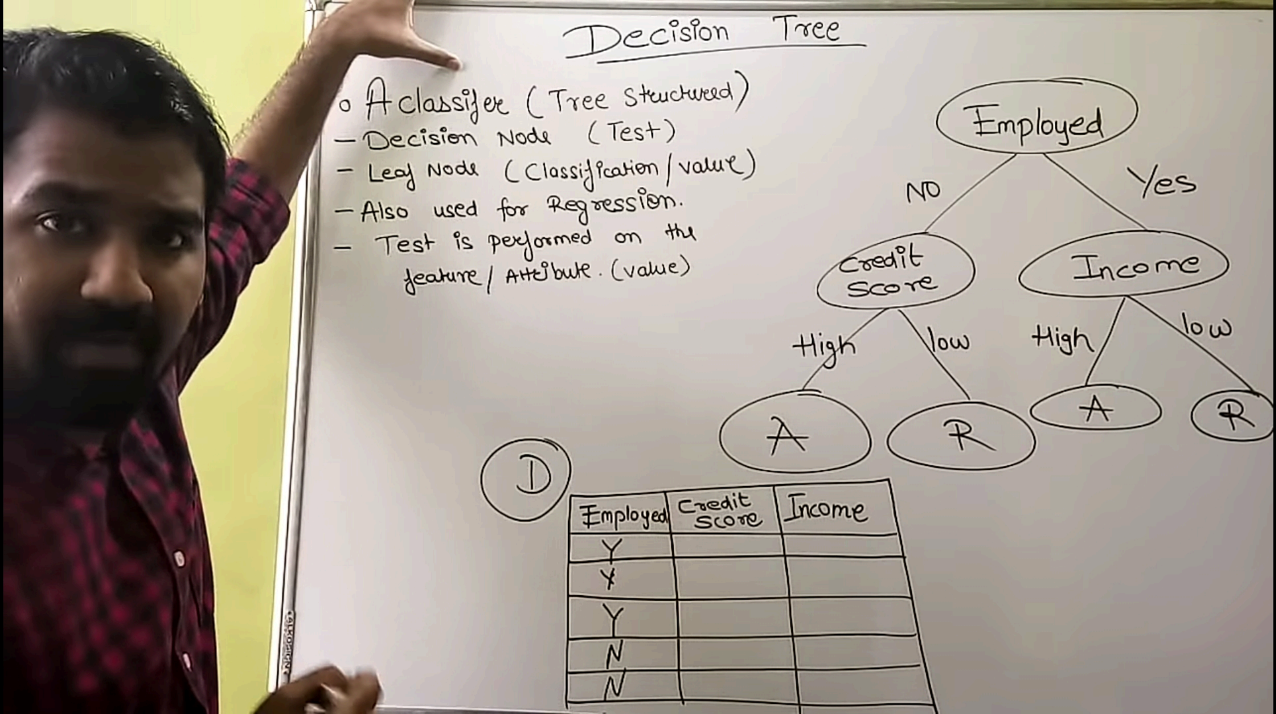
- Multiple linear equation

$$y = \alpha_0 + \alpha_1 x_1 + \alpha_2 x_2 \cdots + \alpha_m x_m$$

α_i = Reg. Coeff.

x_i = Independent Var.

y = Dependent Var.


Decision Tree
o A classifier (Tree Structured)
- Decision Node (Test)
- Leaf Node (Classification / value)
- Also used for Regression.
- Test is performed on the feature / Attribute. (value)
Employed
NO
Yes
Credit Score
Income
High
low
High
low
A
R
A
R
D
Employed
Credit Score
Income
Y
Y
Y
N
N
5 MINUTES ENGINEERING

ML | Underfitting and Overfitting

When we talk about the Machine Learning model, we actually talk about how well it performs and its accuracy which is known as prediction errors. Let us consider that we are designing a machine learning model. A model is said to be a good machine learning model if it generalizes any new input data from the problem domain in a proper way. This helps us to make predictions about future data, that the data model has never seen. Now, suppose we want to check how well our machine learning model learns and generalizes to the new data. For that, we have overfitting and underfitting, which are majorly responsible for the poor performances of the machine learning algorithms.

Before diving further let's understand two important terms:

- **Bias:** Assumptions made by a model to make a function easier to learn. It is actually the error rate of the training data. When the error rate has a high value, we call it High Bias and when the error rate has a low value, we call it low Bias.
- **Variance:** The difference between the error rate of training data and testing data is called variance. If the difference is high then it's called high variance and when the difference of errors is low then it's called low variance. Usually, we want to make a low variance for generalized our model.

Underfitting: A statistical model or a machine learning algorithm is said to have underfitting when it cannot capture the underlying trend of the data, i.e., it only performs well on training data but performs poorly on testing data. *(It's just like trying to fit undersized pants!)* Underfitting destroys the accuracy of our machine learning model. Its occurrence simply means that our model or the algorithm does not fit the data well enough. It usually happens when we have fewer data to build an accurate model and also when we try to build a linear model with fewer non-linear data. In such cases, the rules of the machine learning model are too easy and flexible to be applied to such minimal data and therefore the model will probably make a lot of wrong predictions. Underfitting can be avoided by using more data and also reducing the features by feature selection.

In a nutshell, Underfitting refers to a model that can neither performs well on the training data nor generalize to new data.

Reasons for Underfitting:

1. High bias and low variance
2. The size of the training dataset used is not enough.
3. The model is too simple.
4. Training data is not cleaned and also contains noise in it.

Techniques to reduce underfitting:

1. Increase model complexity
2. Increase the number of features, performing feature engineering
3. Remove noise from the data.
4. Increase the number of epochs or increase the duration of training to get better results.

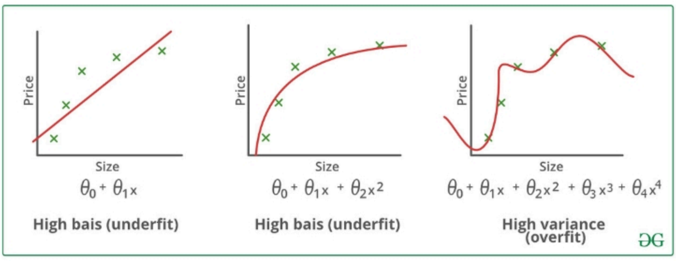
Overfitting: A statistical model is said to be overfitted when the model does not make accurate predictions on testing data. When a model gets trained with so much data, it starts learning from the noise and inaccurate data entries in our data set. And when testing with test data results in High variance. Then the model does not categorize the data correctly, because of too many details and noise. The causes of overfitting are the non-parametric and non-linear methods because these types of machine learning algorithms have more freedom in building the model based on the dataset and therefore they can really build unrealistic models. A solution to avoid overfitting is using a linear algorithm if we have linear data or using the parameters like the maximal depth if we are using decision trees.

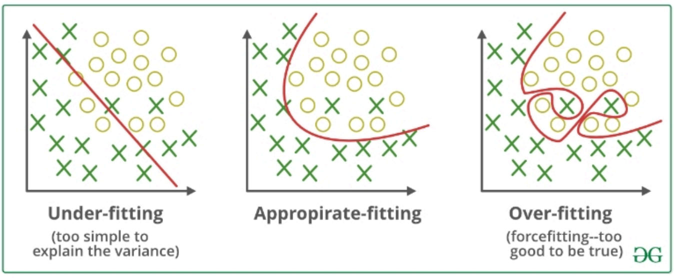
In a nutshell, Overfitting is a problem where the evaluation of machine learning algorithms on training data is different from unseen data.

Reasons for Overfitting are as follows:

1. High variance and low bias
2. The model is too complex
3. The size of the training data

Examples:





Techniques to reduce overfitting:

1. Increase training data.
2. Reduce model complexity.
3. Early stopping during the training phase (have an eye over the loss over the training period as soon as loss begins to increase stop training).
4. Ridge Regularization and Lasso Regularization
5. Use dropout for neural networks to tackle overfitting.

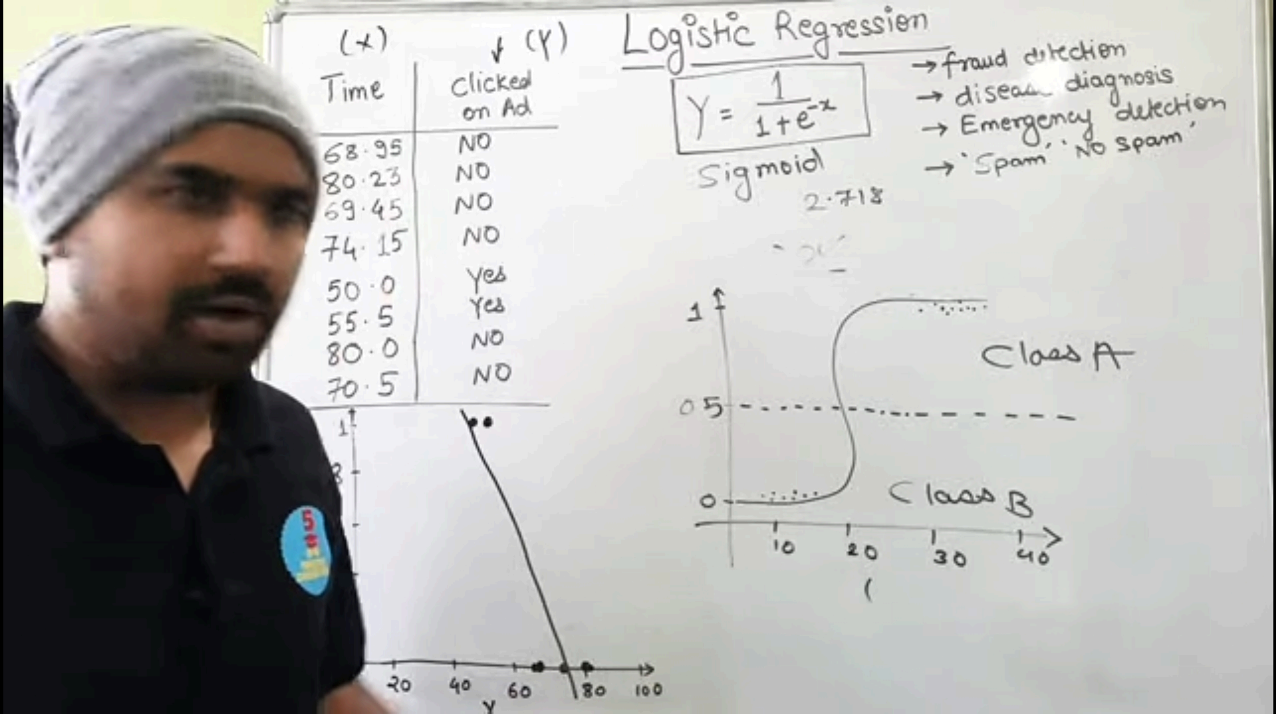
Good Fit in a Statistical Model: Ideally, the case when the model makes the predictions with 0 error, is said to have a *good fit* on the data. This situation is achievable at a spot between overfitting and underfitting. In order to understand it, we will have to look at the performance of our model with the passage of time, while it is learning from the training dataset.

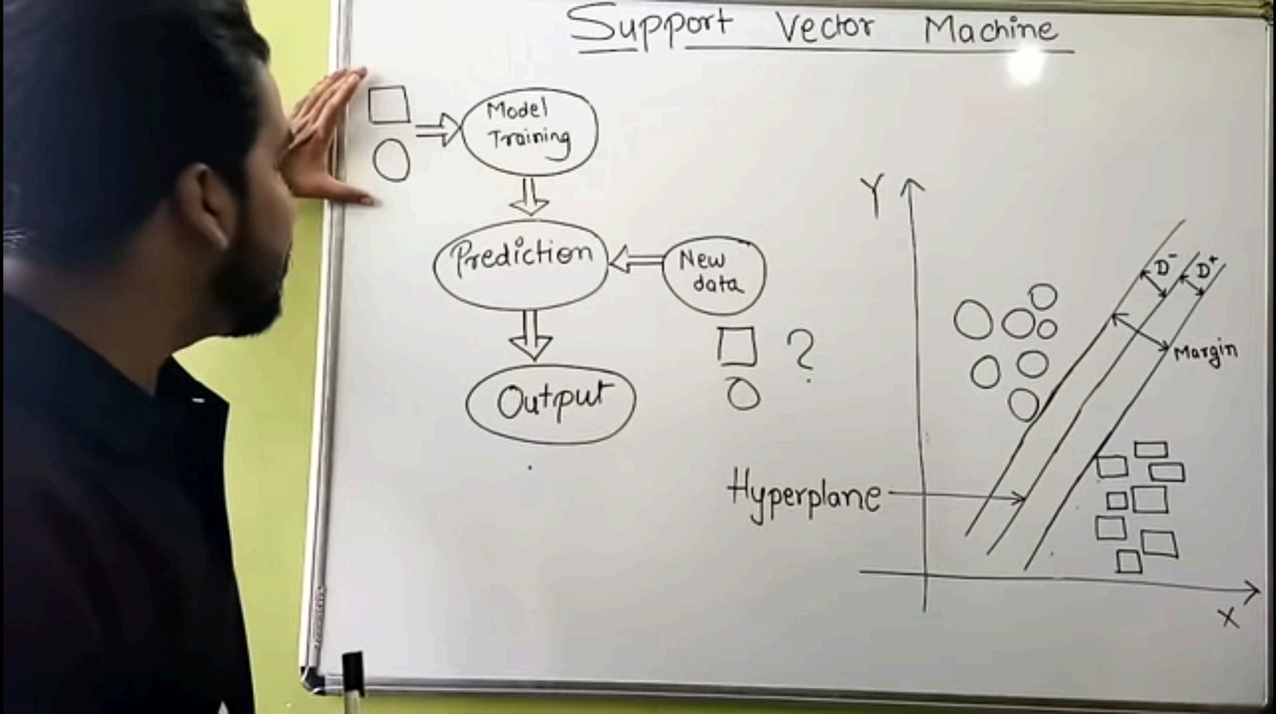
With the passage of time, our model will keep on learning, and thus the error for the model on the training and testing data will keep on decreasing. If it will learn for too long, the model will become more prone to overfitting due to the presence of noise and less useful details. Hence the performance of our model will decrease. In order to get a good fit, we will stop at a point just before where the error starts increasing. At this point, the model is said to have good skills in training datasets as well as our unseen testing dataset.

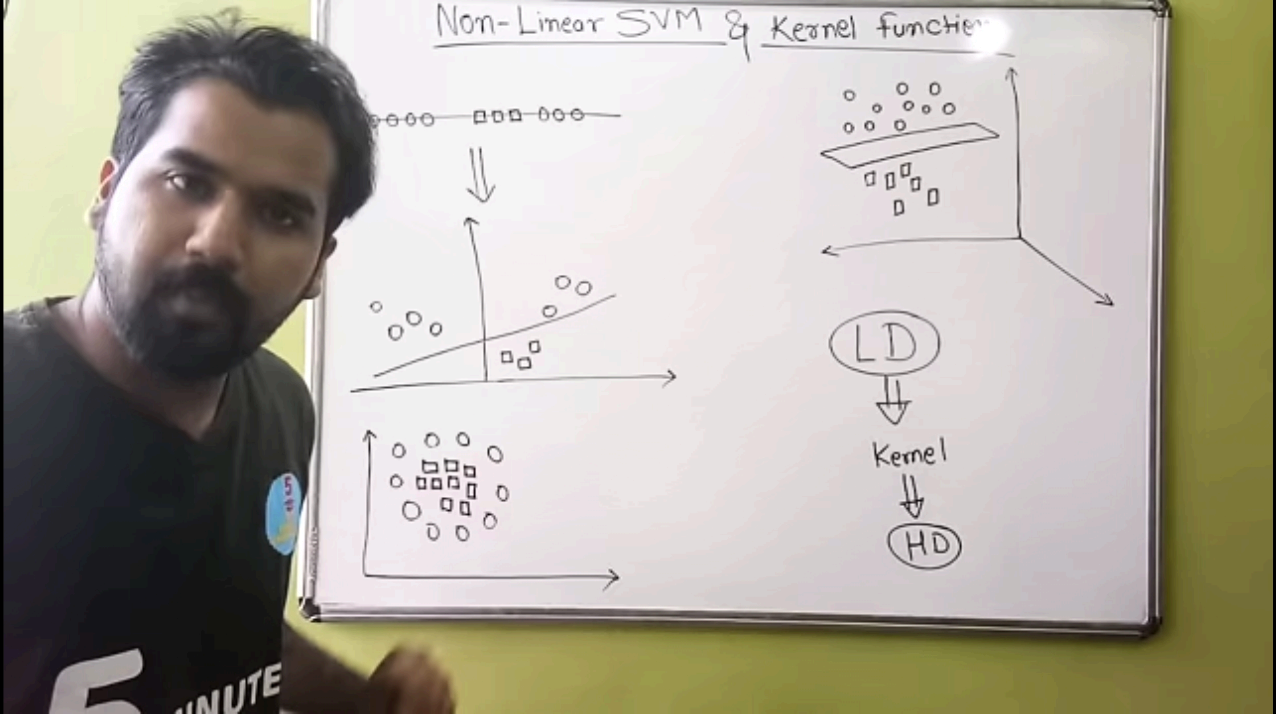
Article Tags : Computer Subject Machine Learning

Recommended Articles

1. [How to Solve Overfitting in Random Forest in Python Sklearn?]


Logistic Regression
(x)
(Y)
Time
Clicked on Ad
68.95 NO
80.23 NO
69.45 NO
74.15 NO
50.0 Yes
55.5 Yes
80.0 NO
70.5 NO
Y = 1 / (1+e^-x)
Sigmoid
2.718
→ fraud detection
→ disease diagnosis
→ Emergency detection
→ 'Spam' 'No spam'
1
0.5
0
Class A
Class B
10
20
30
40
20
40
60
80
100


Support Vector Machine
Model Training
Prediction
New data
Output
?
Y
X
D^-
D^+
Margin
Hyperplane


Non-Linear SVM & Kernel
LD
Kernel
HD

Polynomial Regression

$$y = \alpha_0 + \alpha_1 x_1$$

$$y = \alpha_0 + \alpha_1 x_1 + \alpha_2 x_2 + \alpha_3 x_3 + \alpha_4 x_4$$

$$\boxed{y = \alpha_0 + \sum_{i=1}^{m} \alpha_i x_i}$$

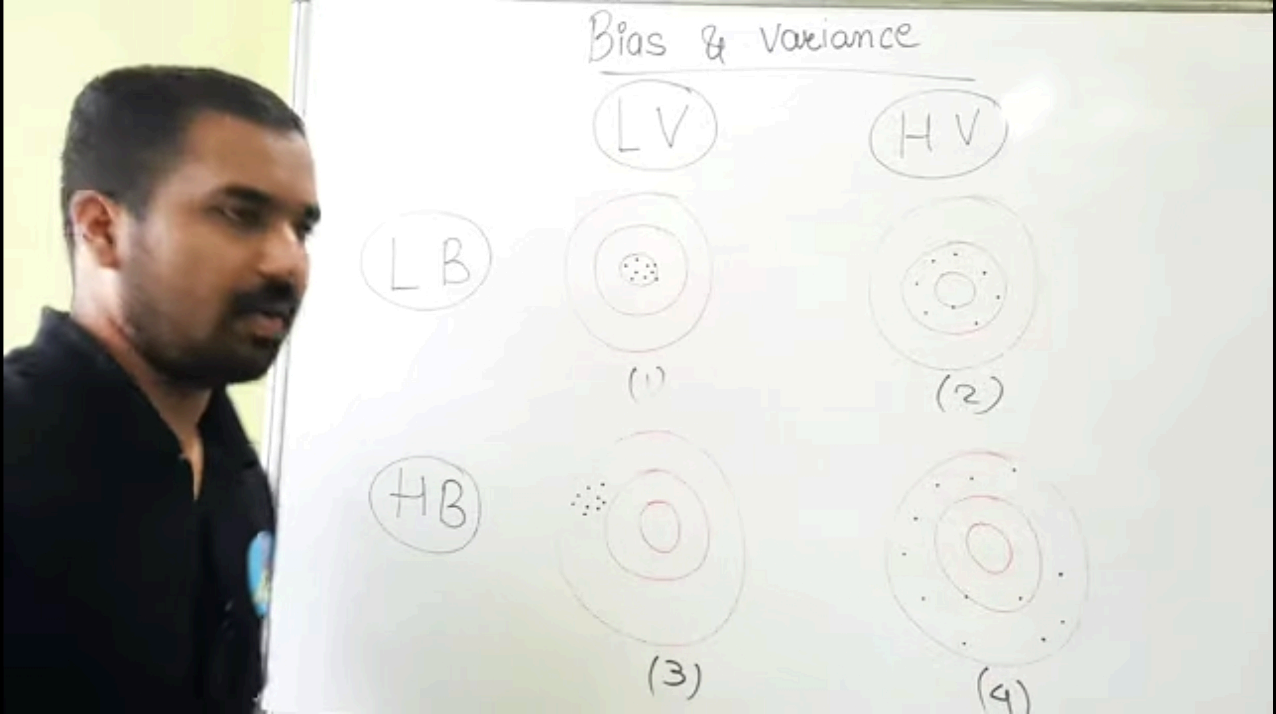
0 degree poly : $\boxed{y = \text{Constant}}$

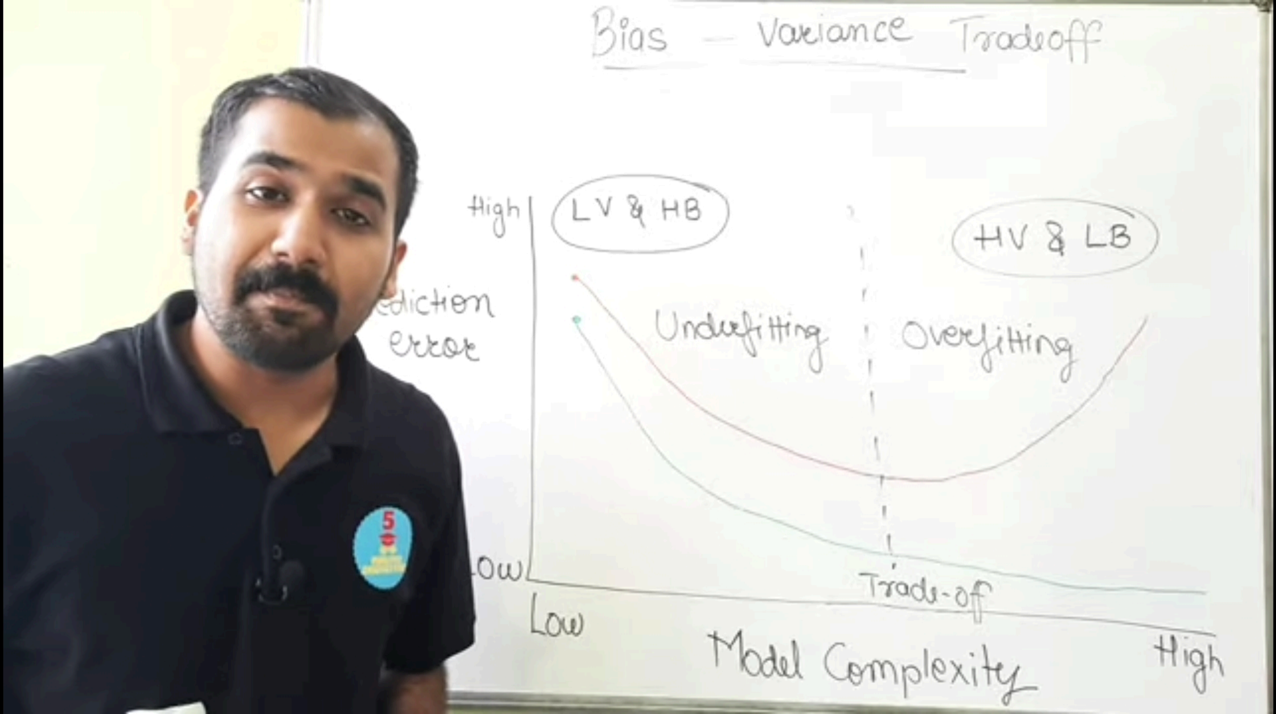
1 degree poly : $y = mx^1 + c$

2 degree poly : $y = ax^2 + bx + c$

$$\boxed{y = \alpha_0 + \alpha_1 x + \alpha_2 x^2 + \alpha_3 x^3 + \cdots + \alpha_n x^n}$$

$$\boxed{y = \alpha_0 + \sum_{i=1}^{m} \alpha_i x_i + \boxed{F_p}}$$


Bias & variance
LV
HV
LB
HB
(1)
(2)
(3)
(4)


Bias - Variance Tradeoff
High
LV & HB
HV & LB
Underfitting
Overfitting
Low
Trade-off
Low
Model Complexity
High

Q

X	Y	XY	X^2
1	3	3	1
2	4	8	4
3	5	15	9
4	7	28	16
10	19	54	30

$$a = \frac{(\Sigma y)(\Sigma x^2) - (\Sigma x)(\Sigma xy)}{n(\Sigma x^2) - (\Sigma x)^2}$$

$$b = \frac{n(\Sigma xy) - (\Sigma x)(\Sigma y)}{n(\Sigma x^2) - (\Sigma x)^2}$$

$$\boxed{Y = bx + a}$$

$$a = \frac{(19)(30) - (10)(54)}{(4)(30) - (100)}$$

$$= \frac{570 - 540}{120 - 100}$$

$$= 30/20 = \frac{3}{2} = 1.5$$

$$b = \frac{(4)(54) - (10)(19)}{(4)(30) - (100)}$$

$$= \frac{216 - 190}{120 - 100}$$

$$= 26/20 = 13/10 = 1.3$$

$$\boxed{Y = 1.3X + 1.5}$$

Q.

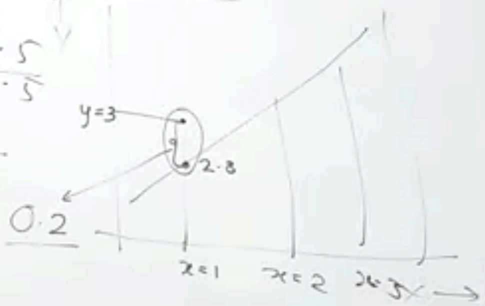
X	Y	XY	X^2	P	Error
1	3	3	1	2.8	0.2
2	4	8	4	4.1	0.1
3	5	15	9	5.4	0.4
4	7	28	16	6.7	0.3
10	19	54	30		

$\boxed{X = 10}$ $\boxed{Y = 14.5}$

$$a = \frac{(\Sigma y)(\Sigma x^2) - (\Sigma x)(\Sigma xy)}{n(\Sigma x^2) - (\Sigma x)^2}$$

13
1.5
14.5

$$b = \frac{n(\Sigma xy) - (\Sigma x)(\Sigma y)}{n(\Sigma x^2) - (\Sigma x)^2}$$



$$\boxed{Y = 1.3X + 1.5}$$

(x, y)	x^2	xy
$(-2, -1)$	4	2
$(1, 1)$	1	1
$(3, 2)$	9	6

$$\sum x = -2 + 1 + 3 = 2$$

$$\sum y = -1 + 1 + 2 = 2$$

$$\sum x^2 = 4 + 1 + 9 = 14$$

$$\sum xy = 2 + 1 + 6 = 6$$

$$N = 3$$

$$m = \frac{N \sum(xy) - \sum x \sum y}{N \sum(x^2) - (\sum x)^2} =$$

$$\frac{3 \cdot 6 - 2 \cdot 2}{3 \cdot 14 - 2^2} = \frac{7}{19}$$

$$b = \frac{\sum y - m \sum x}{N} = \frac{2 - \frac{7}{19} \cdot 2}{3} = \frac{24}{57}$$

so the line is:

$$57y = 33x + 24$$

[Least Squares Regression](#)