

MACHINE LEARNING

UNIT-1

Basic Concepts

Definition of learning systems

A **learning system** is essentially a collection of artefacts that are ‘brought together’, in an appropriate way, in order to create an environment that will facilitate various types of **learning process**. **Learning systems** can take a variety of different forms - for example, a book, a mobile form, a computer, an online forum, a school and a university. Most **learning systems** will provide various types of **learning** resource and descriptions of procedures for using these to achieve particular **learning** outcomes. They will also embed various strategies for assessing the levels and quality of the achievement of their users.

Goals and applications of machine learning

The Goals of Machine Learning.

The goal of ML, in simple words, is to understand the nature of (human and other forms of) learning, and to build learning capability in computers. To be more specific, there are three aspects of the goals of ML.

- 1) To make the computers smarter, more intelligent. The more direct objective in this aspect is to develop systems (programs) for specific practical learning tasks in application domains.
- 2) To develop computational models of human learning process and perform computer simulations. The study in this aspect is also called cognitive modelling.
- 3) To explore new learning methods and develop general learning algorithms independent of applications.

The Applications of Machine Learning.

Machine learning 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 which 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. We probably use a learning algorithm dozens of time without even knowing it. Applications of Machine Learning include:

- **Web Search Engine:** One of the reasons why search engines like google, bing etc work so well is because the system has learnt how to rank pages through a complex learning algorithm.
- **Photo tagging Applications:** Be it facebook or any other photo tagging application, the ability to tag friends makes it even more happening. It is all possible because of a face recognition algorithm that runs behind the application.
- **Spam Detector:** Our mail agent like Gmail or Hotmail does a lot of hard work for us in classifying the mails and moving the spam mails to spam folder. This is again achieved by a spam classifier running in the back end of mail application.

Today, companies are using Machine Learning to improve business decisions, increase productivity, detect disease, forecast weather, and do many more things. With the exponential growth of technology, we not only need better tools to understand the data we currently have, but we also need to prepare ourselves for the data we will have. To achieve this goal we need to build intelligent machines. We can write a program to do simple things. But for most of times Hardwiring Intelligence in it is difficult. Best way to do it is to have some way for machines to learn things themselves. A mechanism for learning – if a machine can learn from input then it does the hard work for us. This is where Machine Learning comes in action. Some examples of machine learning are:

- **Database Mining for growth of automation:** Typical applications include Web-click data for better UX(User eXperience), Medical records for better automation in healthcare, biological data and many more.
- **Applications that cannot be programmed:** There are some tasks that cannot be programmed as the computers we use are not modelled that way. Examples include Autonomous Driving, Recognition tasks from unordered data (Face Recognition/ Handwriting Recognition), Natural language Processing, computer Vision etc.
- **Understanding Human Learning:** This is the closest we have understood and mimicked the human brain. It is the start of a new revolution, The real AI. Now, After a brief insight lets come to a more formal definition of Machine Learning
- **Arthur Samuel(1959):** "Machine Learning is a field of study that gives computers, the ability to learn without explicitly being programmed." Samuel wrote a Checker playing program which could learn over time. At first it could be easily won. But over time, it learnt all the board position that would eventually lead him to victory or loss and thus became a better chess player than Samuel itself. This was one of the most early attempts of defining Machine Learning and is somewhat less formal.
- **Tom Michel(1999):** "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E." This is a more formal and mathematical definition. For the previous Chess program
 - E is number of games.
 - T is playing chess against computer.

- P is win/loss by computer.

Aspects of developing a learning system

Training data

Training data is the real fuel to accelerate the machine learning process. It can only provide the actual inputs to the algorithms to learn the certain patterns and utilize this training to predict with the right results if such data comes again in real-life use.

Actually, training data is generated through labeling process which involves image annotation, text annotation and video annotation using certain techniques to make the objects recognizable to computer vision for machine learning training.

Labeling such data is called “**Data Annotation**” done by the well-trained and experienced annotators to annotate the images available in different formats. The best tools and techniques are used to annotate the object of interest in a image while ensuring the accuracy to train the popular AI models like self-driving cars or robotics.

How Data Annotation is Done?

To annotate the data the combination of humans and machines are used at large scale for mass production and achieve the economies of scale. To annotate the data bulk of images are uploaded on the software servers and then given access to the annotators for annotating the each image as per the requirements.

Types of Data Annotation

The data annotation service basically divided into three categories text, videos or images. And annotation job is done as per the AI project training data needs and algorithms compatibility to utilize the information from such labeled data.

- Text Annotation
- Video Annotation
- Image Annotation

Image annotation is used to train the computer vision based perception model.

And there are different types of techniques adopted in this image annotation service. Bounding box, semantic segmentation, 3D cuboid, polygons, landmark annotation and polylines annotation are the leading image annotation methods used in such tasks.

How Training Data is used in Machine Learning?

Machine learning training is performed through certain amount of data from the relevant field. And to train the AI model through machine learning certain process is followed that involves acquiring training data, using the right algorithm, validation of the model and implementation to check the prediction results in real-life use.

In fact, in machine learning labeled or unlabeled training data is used as per the supervised or unsupervised ML process. In supervised machine learning the object is categorized, classified and segmented to make it recognizable to machines.

While in unsupervised machine learning, the data is not labeled and algorithm have to group the object as per the understandings to group them accordingly with its own segmentation to recognize the same when used in future.

How to Acquire Training Data for Machine Learning?

Machine learning training is possible only when you have quality data sets, and acquiring the training data is one the most challenging job in AI world. But if you get in touch with companies like Cogito, you can get high-quality training data for Artificial Intelligence your machine learning project with best level of accuracy for right prediction.

Concept representation

In deep learning, feature representations are generally learned as a blob of ungrouped features. However, an increasing number of visual applications nourish from inferring knowledge from imagery which requires scene understanding. Semantic segmentation is a task that paves the way towards scene understanding. Deep semantic segmentation [17] uses deep learning for semantic segmentation.

Deep semantic segmentation makes dense predictions inferring labels for every pixel. It can be carried out at three different levels:

- Class segmentation: each pixel is labeled with the class of its enclosing object or region
- Instance segmentation: separate labels for different instances of the same class
- Part segmentation: decomposition of already segmented classes into their component sub-classes

CODL extends and generalizes deep semantic segmentation. In CODL, feature representations are always learned semantically segmented in a concept-oriented manner. Concept orientation means that each feature representation is associated with a concept, an instance or an attribute. These concepts, instances and attributes form a concept graph. In addition, the concept graph are generally linked to Microsoft Concept Graph, thus leveraging and integrating with the common conceptual knowledge and conceptual understanding capability provided by Microsoft Concept Graph.

A concept representation consists of a concept, its instances and attributes, and all the feature representations associated with the concept and its instances and attributes. If a concept has sub-concepts, its concept representation also consists of the 5 concept representations of its sub-concepts. Concept representations, therefore, are the same as concept-oriented feature representations, but provide a different view. The latter is data driven and provides a bottom-up view starting from feature representations; the former is concept driven and provides a top-down view starting from concepts. Due to the focus on concepts instead of low-level feature representations, concept representations provide the proper view to work with in CODL.

Supervised Concept Representation Learning

Concept representations can be learned using supervised learning. Similar to deep semantic segmentation, discussed above, it can be carried out at different levels:

- Concept level: each feature representation is labeled with the concept that owns the feature
- Instance level: separate labels for different instances of the same concept
- Attribute level: separate labels for different attributes of the same concept
- Component level: decomposition of already learned concept representations into their sub-concept representations

The concept, instance and attribute names used for labeling should be taken from Microsoft Concept Graph, if available. This provides direct link to Microsoft Concept Graph to leverage its common conceptual knowledge and conceptual understanding capability.

Function approximation

In general, a function approximation problem asks us to select a function among a well-defined class that closely matches ("approximates") a **target function** in a task-specific

way. The need for **function approximations** arises in many branches of applied mathematics, and computer science in particular.

One can distinguish two major classes of function approximation problems:

First, for known target functions approximation theory is the branch of numerical analysis that investigates how certain known functions (for example, special functions) can be approximated by a specific class of functions (for example, polynomials or rational functions) that often have desirable properties (inexpensive computation, continuity, integral and limit values, etc.).

Second, the target function, call it g , may be unknown; instead of an explicit formula, only a set of points of the form $(x, g(x))$ is provided. Depending on the structure of the domain and codomain of g , several techniques for approximating g may be applicable. For example, if g is an operation on the real numbers, techniques of interpolation, extrapolation, regression analysis, and curve fitting can be used. If the codomain (range or target set) of g is a finite set, one is dealing with a classification problem instead.

To some extent, the different problems (regression, classification, fitness approximation) have received a unified treatment in statistical learning theory, where they are viewed as supervised learning problems.

Types of Learning

Supervised learning and unsupervised learning

Supervised learning

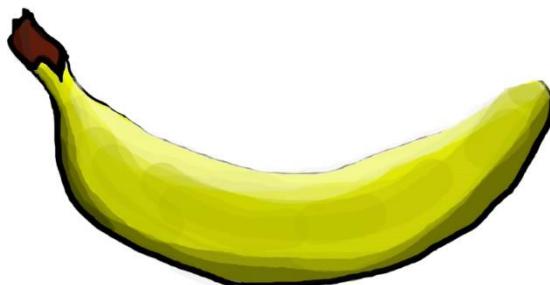
Supervised learning as the name indicates the presence of a supervisor as a teacher. Basically supervised learning is a learning in which we teach or train the machine using data which is well labeled that means some data is already tagged with the correct answer. After that, the machine is provided with a new set of examples(data) so that supervised learning algorithm analyses the training data(set of training examples) and produces a correct outcome from labeled data.

For instance, suppose you are given a basket filled with different kinds of fruits. Now the first step is to train the machine with all different fruits one by one like this:



- If shape of object is rounded and depression at top having color Red then it will be labelled as **–Apple**.
- If shape of object is long curving cylinder having color Green-Yellow then it will be labelled as **–Banana**.

Now suppose after training the data, you have given a new separate fruit say Banana from basket and asked to identify it.



Since the machine has already learned the things from previous data and this time have to use it wisely. It will first classify the fruit with its shape and color and would confirm

the fruit name as BANANA and put it in Banana category. Thus the machine learns the things from training data(basket containing fruits) and then apply the knowledge to test data(new fruit).

Supervised learning classified into two categories of algorithms:

- **Classification:** A classification problem is when the output variable is a category, such as "Red" or "blue" or "disease" and "no disease".
- **Regression:** A regression problem is when the output variable is a real value, such as "dollars" or "weight".

Unsupervised learning

Unsupervised learning is the training of machine using information that is neither classified nor labeled and allowing the algorithm to act on that information without guidance. Here the task of machine is to group unsorted information according to similarities, patterns and differences without any prior training of data.

Unlike supervised learning, no teacher is provided that means no training will be given to the machine. Therefore machine is restricted to find the hidden structure in unlabeled data by our-self.

For instance, suppose it is given an image having both dogs and cats which have not seen ever.



Thus the machine has no idea about the features of dogs and cat so we can't categorize it in dogs and cats. But it can categorize them according to their similarities, patterns, and differences i.e., we can easily categorize the above picture into two parts. First part may contain all pics having **dogs** in it and second part may contain all pics having **cats** in it. Here you didn't learn anything before, means no training data or examples.

Unsupervised learning classified into two categories of algorithms:

- **Clustering:** A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behavior.
- **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 X also tend to buy Y.

Overview of classification

Classification - Machine Learning

This is 'Classification' tutorial which is a part of the [Machine Learning course](#) offered by Simplilearn. We will learn Classification algorithms, types of classification algorithms, support vector machines(SVM), Naive Bayes, Decision Tree and Random Forest Classifier in this tutorial.

Objectives

Let us look at some of the objectives covered under this section of Machine Learning tutorial.

- Define Classification and list its algorithms
- Describe Logistic Regression and Sigmoid Probability
- Explain K-Nearest Neighbors and KNN classification Understand Support Vector Machines, Polynomial Kernel, and Kernel Trick
- Analyze Kernel Support Vector Machines with an example
- Implement the Naïve Bayes Classifier
- Demonstrate Decision Tree Classifier
- Describe Random Forest Classifier

Classification: Meaning

Classification is a type of supervised learning. It specifies the class to which data elements belong to and is best used when the output has finite and discrete values. It predicts a class for an input variable as well.

There are 2 types of Classification:

- Binomial
- Multi-Class

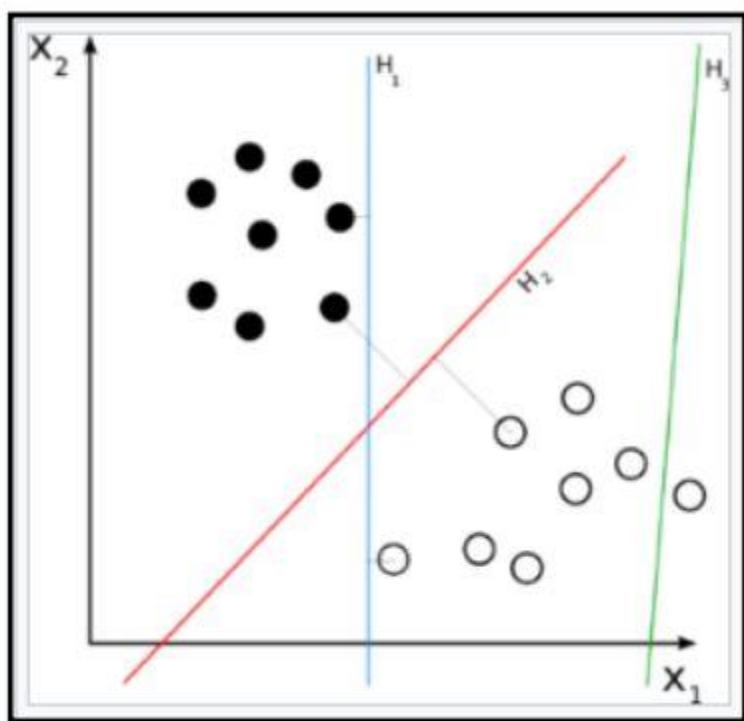
Classification: Use Cases

Some of the key areas where classification cases are being used:

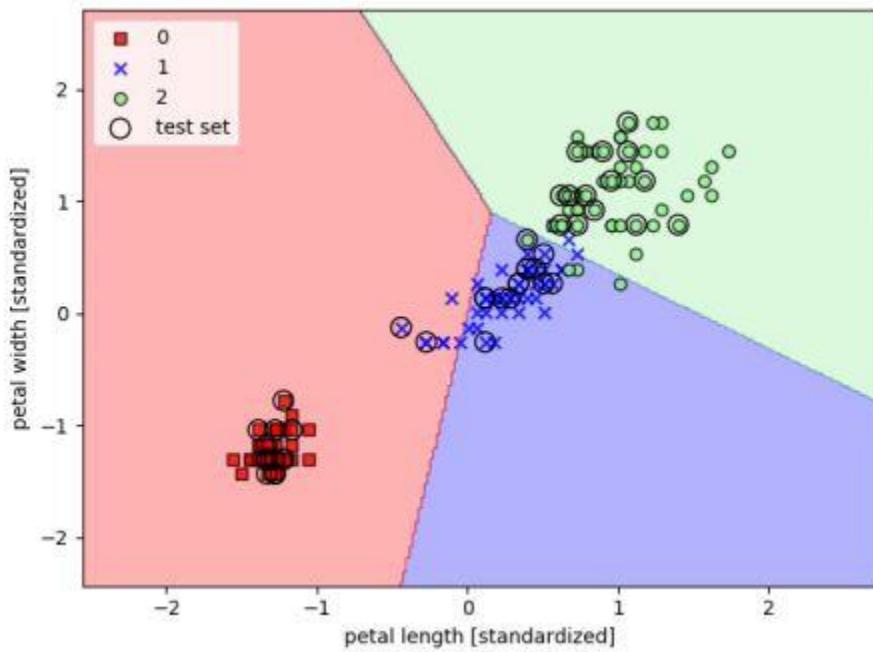
- To find whether an email received is a spam or ham
- To identify customer segments
- To find if a bank loan is granted
- To identify if a kid will pass or fail in an examination

Classification: Example

Social media sentiment analysis has two potential outcomes, positive or negative, as displayed by the chart given below.



- This chart shows the classification of the Iris flower dataset into its three sub-species indicated by codes 0, 1, and 2.



- The test set dots represent the assignment of new test data points to one class or the other based on the trained classifier model.

Types of Classification Algorithms

Let's have a quick look into the types of Classification Algorithm below.

- Linear Models
 - Logistic Regression
 - Support Vector Machines
- Nonlinear models
 - K-nearest Neighbors (KNN)
 - Kernel Support Vector Machines (SVM)
 - Naïve Bayes
 - Decision Tree Classification
 - Random Forest Classification

Training Dataset

Training Dataset: *The sample of data used to fit the model.*

The actual dataset that we use to train the model (weights and biases in the case of Neural Network). The model *sees* and *learns* from this data.

Validation Dataset

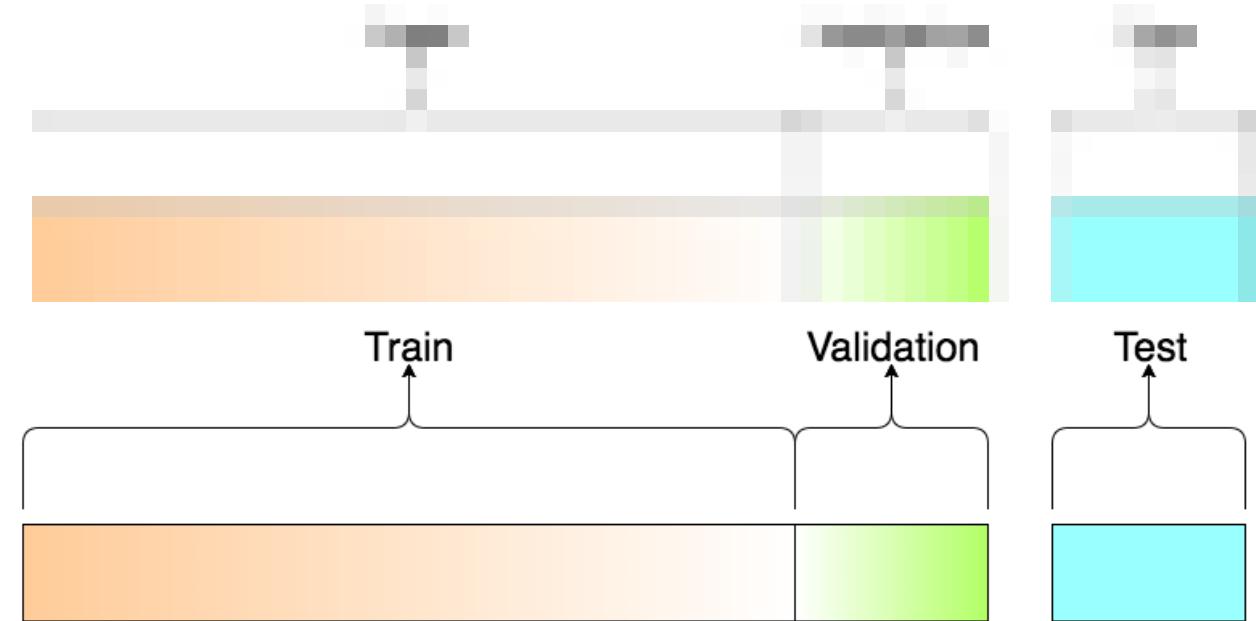
Validation Dataset: *The sample of data used to provide an unbiased evaluation of a model fit on the training dataset while tuning model hyperparameters. The evaluation becomes more biased as skill on the validation dataset is incorporated into the model configuration.*

The validation set is used to evaluate a given model, but this is for frequent evaluation. We as machine learning engineers use this data to fine-tune the model hyperparameters. Hence the model occasionally *sees* this data, but never does it “*Learn*” from this. We(mostly humans, at-least as of 2017 😊) use the validation set results and update higher level hyperparameters. So the validation set in a way affects a model, but indirectly.

Test Dataset

Test Dataset: *The sample of data used to provide an unbiased evaluation of a final model fit on the training dataset.*

The Test dataset provides the gold standard used to evaluate the model. It is only used once a model is completely trained(using the train and validation sets). The test set is generally what is used to evaluate competing models (For example on many Kaggle competitions, the validation set is released initially along with the training set and the actual test set is only released when the competition is about to close, and it is the result of the the model on the Test set that decides the winner). Many a times the validation set is used as the test set, but it is not good practice. The test set is generally well curated. It contains carefully sampled data that spans the various classes that the model would face, when used in the real world.



About the dataset split ratio

Now that you know what these datasets do, you might be looking for recommendations on how to split your dataset into Train, Validation and Test sets...

This mainly depends on 2 things. First, the total number of samples in your data and second, on the actual model you are training.

Some models need substantial data to train upon, so in this case you would optimize for the larger training sets. Models with very few hyperparameters will be easy to validate and tune, so you can probably reduce the size of your validation set, but if your model has many hyperparameters, you would want to have a large validation set as well(although you should also consider cross validation). Also, if you happen to have a model with no hyperparameters or ones that cannot be easily tuned, you probably don't need a validation set too!

All in all, like many other things in machine learning, the train-test-validation split ratio is also quite specific to your use case and it gets easier to make judgement as you train and build more and more models.

*Note on Cross Validation: Many a times, people first split their dataset into 2 — Train and Test. After this, they keep aside the Test set, and randomly choose X% of their Train dataset to be the actual **Train** set and the remaining (100-X)% to be the **Validation** set, where X is a fixed number(say 80%), the model is then iteratively trained and validated on these different sets. There are multiple ways to do this, and is commonly known as Cross Validation. Basically you use your training set to generate multiple splits of the Train and Validation sets. Cross validation avoids overfitting and is getting more and more popular, with K-fold Cross Validation being the most popular method of cross validation.*

Overfitting in Machine Learning

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 of 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.

Overfitting is more likely with nonparametric and nonlinear models that have more flexibility when learning a target function. As such, many nonparametric machine learning algorithms also include parameters or techniques to limit and constrain how much detail the model learns.

For example, decision trees are a nonparametric machine learning algorithm that is very flexible and is subject to overfitting training data. This problem can be addressed by pruning a tree after it has learned in order to remove some of the detail it has picked up.

Classification Families

Discriminant Analysis

During a study, there are often questions that strike the researcher that must be answered. These questions include questions like 'are the groups different?', 'on what variables, are the groups most different?', 'can one predict which group a person belongs to using such variables?' etc. In answering such questions, **discriminant analysis** is quite helpful.

Statistics Solutions is the country's leader in discriminant analysis and dissertation statistics.

Discriminant analysis is a technique that is used by the researcher to analyze the research data when the criterion or the dependent variable is categorical and the predictor or the independent variable is interval in nature. The term categorical variable means that the dependent variable is divided into a number of categories. For example, three brands of computers, Computer A, Computer B and Computer C can be the categorical dependent variable.

The objective of discriminant analysis is to develop discriminant functions that are nothing but the linear combination of independent variables that will discriminate between the categories of the dependent variable in a perfect manner. It enables the researcher to examine whether significant differences exist among the groups, in terms of the predictor variables. It also evaluates the accuracy of the classification.

Discriminant analysis is described by the number of categories that is possessed by the dependent variable.

As in statistics, everything is assumed up until infinity, so in this case, when the dependent variable has two categories, then the type used is two-group discriminant analysis. If the dependent variable has three or more than three categories, then the type used is multiple discriminant analysis. The major distinction to the types of discriminant analysis is that for a two group, it is possible to derive only one discriminant function. On the other hand, in the case of multiple discriminant analysis, more than one discriminant function can be computed.

There are many examples that can explain when discriminant analysis fits. It can be used to know whether heavy, medium and light users of soft drinks are different in terms of their consumption of frozen foods. In the field of psychology, it can be used to differentiate between the price sensitive and

non price sensitive buyers of groceries in terms of their psychological attributes or characteristics. In the field of business, it can be used to understand the characteristics or the attributes of a customer possessing store loyalty and a customer who does not have store loyalty.

For a researcher, it is important to understand the relationship of discriminant analysis with Regression and Analysis of Variance (ANOVA) which has many similarities and differences. Often we can find similarities and differences with the people we come across. Similarly, there are some similarities and differences with discriminant analysis along with two other procedures. The similarity is that the number of dependent variables is one in discriminant analysis and in the other two procedures, the number of independent variables are multiple in discriminant analysis. The difference is categorical or binary in discriminant analysis, but metric in the other two procedures. The nature of the independent variables is categorical in Analysis of Variance (ANOVA), but metric in regression and discriminant analysis.

The steps involved in conducting discriminant analysis are as follows:

- The problem is formulated before conducting.
- The discriminant function coefficients are estimated.
- The next step is the determination of the significance of these discriminant functions.
- One must interpret the results obtained.
- The last and the most important step is to assess the validity.

Linear Discriminant Analysis

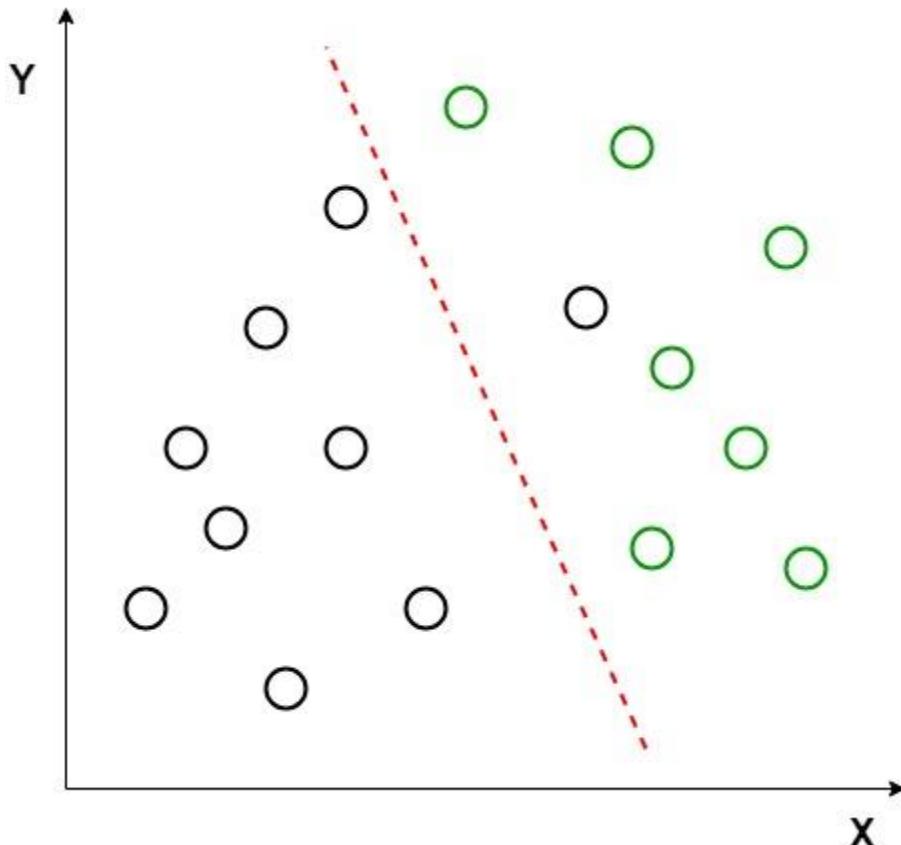
Linear Discriminant Analysis or Normal Discriminant Analysis or Discriminant Function Analysis is a dimensionality reduction technique which is commonly used for the supervised classification problems. It is used for modeling differences in groups i.e. separating two or more classes. It is used to project the features in higher dimension space into a lower dimension space.

For example, we have two classes and we need to separate them efficiently. Classes can have multiple features. Using only a single feature to classify them may result in some overlapping as shown in the below figure. So, we will keep on increasing the number of features for proper classification.



Example:

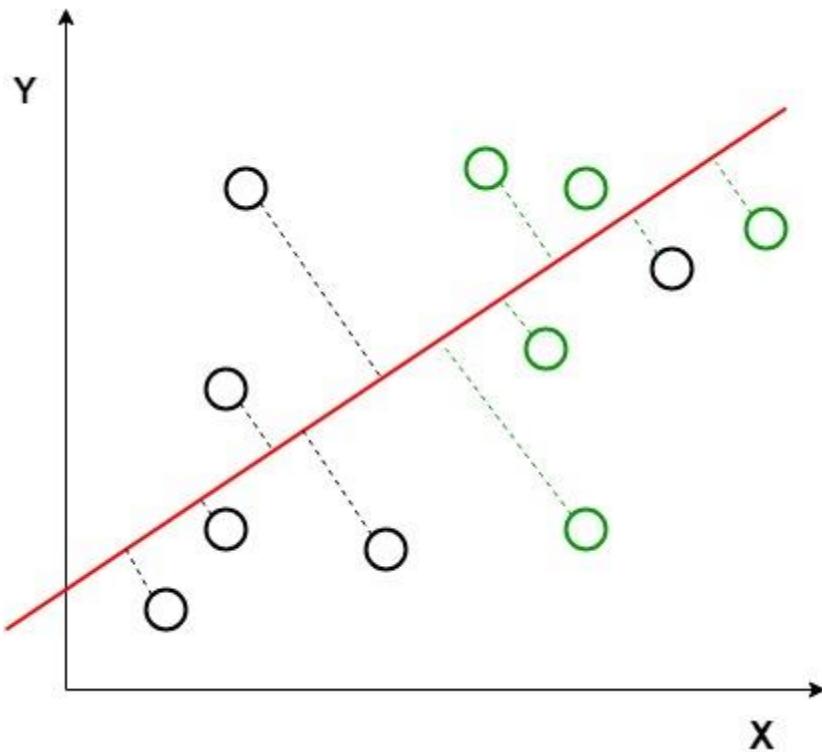
Suppose we have two sets of data points belonging to two different classes that we want to classify. As shown in the given 2D graph, when the data points are plotted on the 2D plane, there's no straight line that can separate the two classes of the data points completely. Hence, in this case, LDA (Linear Discriminant Analysis) is used which reduces the 2D graph into a 1D graph in order to maximize the separability between the two classes.



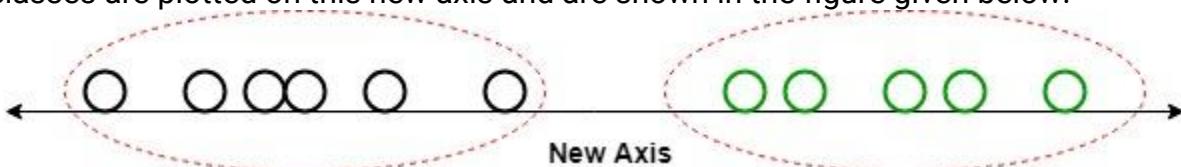
Here, Linear Discriminant Analysis uses both the axes (X and Y) to create a new axis and projects data onto a new axis in a way to maximize the separation of the two categories and hence, reducing the 2D graph into a 1D graph.

Two criteria are used by LDA to create a new axis:

1. Maximize the distance between means of the two classes.
2. Minimize the variation within each class.



In the above graph, it can be seen that a new axis (in red) is generated and plotted in the 2D graph such that it maximizes the distance between the means of the two classes and minimizes the variation within each class. In simple terms, this newly generated axis increases the separation between the data points of the two classes. After generating this new axis using the above-mentioned criteria, all the data points of the classes are plotted on this new axis and are shown in the figure given below.



But Linear Discriminant Analysis fails when the mean of the distributions are shared, as it becomes impossible for LDA to find a new axis that makes both the classes linearly separable. In such cases, we use non-linear discriminant analysis.

Extensions to LDA:

- Quadratic Discriminant Analysis (QDA):** Each class uses its own estimate of variance (or covariance when there are multiple input variables).
- Flexible Discriminant Analysis (FDA):** Where non-linear combinations of inputs is used such as splines.
- Regularized Discriminant Analysis (RDA):** Introduces regularization into the estimate of the variance (actually covariance), moderating the influence of different variables on LDA.

Applications:

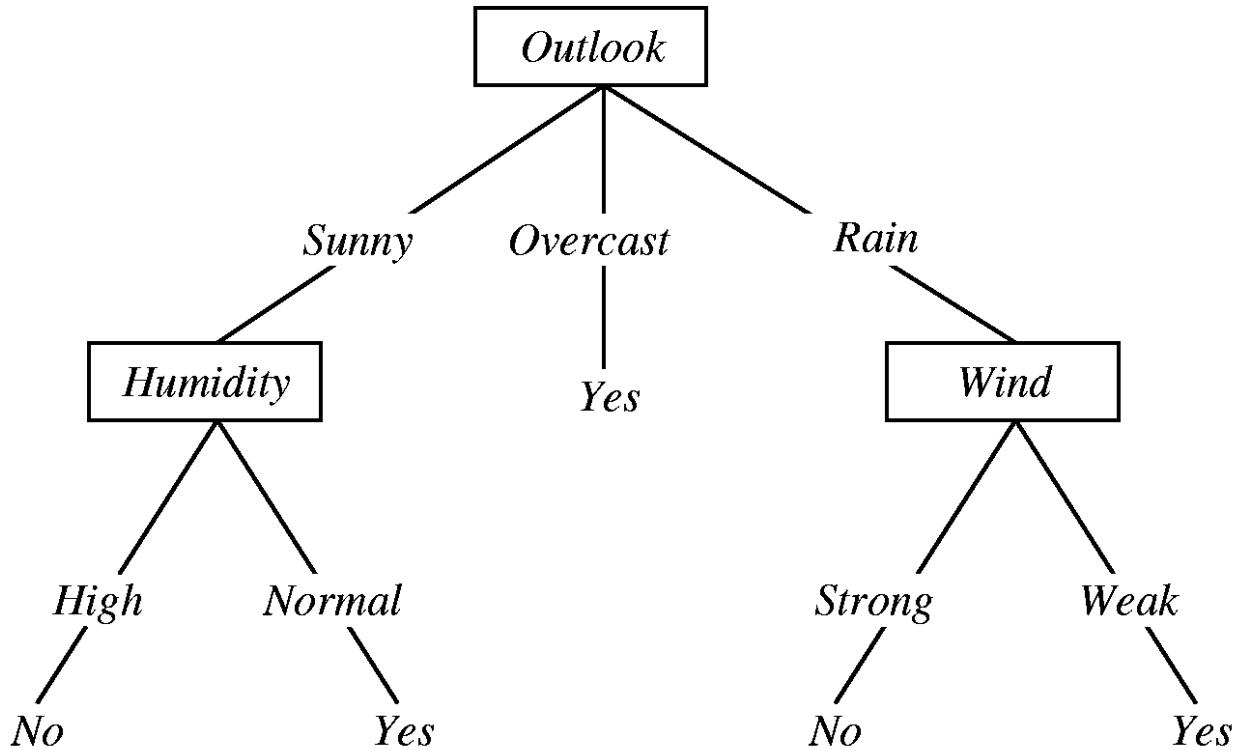
- Face Recognition:** In the field of Computer Vision, face recognition is a very popular application in which each face is represented by a very large number of pixel values. Linear discriminant analysis (LDA) is used here to reduce the number of features to a

more manageable number before the process of classification. Each of the new dimensions generated is a linear combination of pixel values, which form a template. The linear combinations obtained using Fisher's linear discriminant are called Fisher faces.

2. **Medical:** In this field, Linear discriminant analysis (LDA) is used to classify the patient disease state as mild, moderate or severe based upon the patient various parameters and the medical treatment he is going through. This helps the doctors to intensify or reduce the pace of their treatment.
3. **Customer Identification:** Suppose we want to identify the type of customers which are most likely to buy a particular product in a shopping mall. By doing a simple question and answers survey, we can gather all the features of the customers. Here, Linear discriminant analysis will help us to identify and select the features which can describe the characteristics of the group of customers that are most likely to buy that particular product in the shopping mall.

Decision Tree in Machine Learning

A decision tree is a flowchart-like structure in which each internal node represents a `test` on a feature (e.g. whether a coin flip comes up heads or tails), each leaf node represents a `class label` (decision taken after computing all features) and branches represent conjunctions of features that lead to those class labels. The paths from root to leaf represent `classification rules`. Below diagram illustrate the basic flow of decision tree for decision making with labels (Rain(Yes), No Rain(No)).



Decision Tree for Rain Forecasting

Decision tree is one of the predictive modelling approaches used in statistics, data mining and machine learning.

Decision trees are constructed via an algorithmic approach that identifies ways to split a data set based on different conditions. It is one of the most widely used and practical methods for supervised learning. Decision Trees are a non-parametric supervised learning method used for both **classification** and **regression** tasks.

Tree models where the target variable can take a discrete set of values are called **classification trees**. Decision trees where the target variable can take continuous values (typically real numbers) are called **regression trees**. Classification And Regression Tree (CART) is general term for this.

Throughout this post i will try to explain using the examples.

Data Format

Data comes in records of forms.

$$(x, Y) = (x_1, x_2, x_3, \dots, x_k, Y)$$

The dependent variable, Y, is the target variable that we are trying to understand, classify or generalize. The vector x is composed of the features, x_1, x_2, x_3 etc., that are used for that task.

Example

```
training_data = [
    ['Green', 3, 'Apple'],
    ['Yellow', 3, 'Apple'],
    ['Red', 1, 'Grape'],
    ['Red', 1, 'Grape'],
    ['Yellow', 3, 'Lemon'],
]
# Header = ["Color", "diameter", "Label"]
# The last column is the label.
# The first two columns are features.
```

Approach to make decision tree

While making decision tree, at each node of tree we ask different type of questions. Based on the asked question we will calculate the information gain corresponding to it.

Information Gain

Information gain is used to decide which feature to split on at each step in building the tree. Simplicity is best, so we want to keep our tree small. To do so, at each step we should choose the split that results in the purest daughter nodes. A commonly used measure of

purity is called information. For each node of the tree, the information value **measures how much information a feature gives us about the class. The split with the highest information gain will be taken as the first split and the process will continue until all children nodes are pure, or until the information gain is 0.**

Probabilistic classification

In machine learning, a **probabilistic classifier** is a classifier that is able to predict, given an observation of an input, a probability distribution over a set of classes, rather than only outputting the most likely class that the observation should belong to. Probabilistic classifiers provide classification that can be useful in its own right^[1] or when combining classifiers into ensembles.

Nearest Neighbour Classifier

Among the various methods of supervised statistical pattern recognition, the Nearest Neighbour rule achieves consistently high performance, without *a priori* assumptions about the distributions from which the training examples are drawn. It involves a training set of both positive and negative cases. A new sample is classified by calculating the distance to the nearest training case; the sign of that point then determines the classification of the sample. The k -NN classifier extends this idea by taking the k nearest points and assigning the sign of the majority. It is common to select k small and odd to break ties (typically 1, 3 or 5). Larger k values help reduce the effects of noisy points within the training data set, and the choice of k is often performed through cross-validation.

There are many techniques available for improving the performance and speed of a nearest neighbour classification. One approach to this problem is to pre-sort the training sets in some way (such as kd -trees or Voronoi cells). Another solution is to choose a subset of the training data such that classification by the 1-NN rule (using the subset) approximates the Bayes classifier. This can result in significant speed improvements as k can now be limited to 1 and redundant data points have been removed from the training set. These data modification techniques can also improve the performance through removing points that cause mis-classifications. Several dataset reduction techniques are discussed in the section on target detection.

The above discussion focuses on binary classification problems; there are only two possible output classes. In the digit recognition example there are ten output classes, which changes things slightly. The labelling of training samples and computing the distance are unchanged, but ties can now occur even with k odd. If all of the k nearest neighbours are from different classes we are no closer to a decision than with the single nearest neighbour rule. We will therefore revert to a 1-NN rule when all there is no majority within the k nearest neighbours.

The nearest neighbour rule is quite simple, but very computationally intensive. For the digit example, each classification requires 60,000 distance calculations between 784 dimensional vectors (28x28 pixels). The nearest neighbour code was therefore written in C in order to speed up the Matlab testing. The files are given below, but note that these are set up to read in the image database after it has been converted from the format available on the MNIST web page.