ABINAYA.S, AP/CSE

SUBJECT CODE : AL3451

Strictly as per Revised Syllabus of

ANNA UNIVERSITY

Choice Based Credit System (CBCS)

Semester - IV (AI&DS / CS&BS)

MACHINE LEARNING

Iresh A. Dhotre

M.E. (Information Technology)

Ex-Faculty, Sinhgad College of Engineering,

Pune.

TECHNICAL PUBLICATIONS

SINCE 1993

An Up-Thrust for Knowledge

MACHINE LEARNING

Subject Code : AL3451

Semester - IV (Artificial Intelligence and Data Science / Computer Science and Business Systems)

Published by :

TECHNICAL PUBLICATIONS
SINCE 1993 — An Up-Thrust for Knowledge

Amit Residency, Office No.1, 412, Shaniwar Peth,
Pune - 411030, M.S. INDIA, Ph.: +91-020-24495496/97
Email : info@technicalpublications.in Website : www.technicalpublications.in

Printer :
Yogiraj Printers & Binders
Sr.No. 10/1A,
Ghule Industrial Estate, Nanded Village Road,
Tal. - Haveli, Dist. - Pune - 411041.

ISBN 978-93-5585-275-5

AU 21

PREFACE

The importance of **Machine Learning** is well known in various engineering fields. Overwhelming response to my books on various subjects inspired me to write this book. The book is structured to cover the key aspects of the subject **Machine Learning.**

The book uses plain, lucid language to explain fundamentals of this subject. The book provides logical method of explaining various complicated concepts and stepwise methods to explain the important topics. Each chapter is well supported with necessary illustrations, practical examples and solved problems. All the chapters in the book are arranged in a proper sequence that permits each topic to build upon earlier studies. All care has been taken to make students comfortable in understanding the basic concepts of the subject.

The book not only covers the entire scope of the subject but explains the philosophy of the subject. This makes the understanding of this subject more clear and makes it more interesting. The book will be very useful not only to the students but also to the subject teachers. The students have to omit nothing and possibly have to cover nothing more.

I wish to express my profound thanks to all those who helped in making this book a reality. Much needed moral support and encouragement is provided on numerous occasions by my whole family. I wish to thank the **Publisher** and the entire team of **Technical Publications** who have taken immense pain to get this book in time with quality printing.

Any suggestion for the improvement of the book will be acknowledged and well appreciated.

Author
D. A. Dhotre

Dedicated to God.

SYLLABUS

Machine Learning (AL3451)

UNIT I INTRODUCTION TO MACHINE LEARNING

Review of Linear Algebra for machine learning; Introduction and motivation for machine learning; Examples of machine learning applications, Vapnik-Chervonenkis (VC) dimension, Probably Approximately Correct (PAC) learning, Hypothesis spaces, Inductive bias, Generalization, Bias variance trade-off. **(Chapter - 1)**

UNIT II SUPERVISED LEARNING

Linear Regression Models : Least squares, single & multiple variables, Bayesian linear regression, gradient descent, Linear Classification Models : Discriminant function - Perceptron algorithm, Probabilistic discriminative model - Logistic regression, Probabilistic generative model - Naïve Bayes, Maximum margin classifier - Support vector machine, Decision Tree, Random Forests **(Chapter - 2)**

UNIT III ENSEMBLE TECHNIQUES AND UNSUPERVISED LEARNING

Combining multiple learners : Model combination schemes, Voting, Ensemble Learning - bagging, boosting, stacking, Unsupervised learning : K-means, Instance Based Learning : KNN, Gaussian mixture models and Expectation maximization. **(Chapter - 3)**

UNIT IV NEURAL NETWORKS

Multilayer perceptron, activation functions, network training - gradient descent optimization - stochastic gradient descent, error backpropagation, from shallow networks to deep networks - Unit saturation (aka the vanishing gradient problem) - ReLU, hyperparameter tuning, batch normalization, regularization, dropout. **(Chapter - 4)**

UNIT V DESIGN AND ANALYSIS OF MACHINE LEARNING EXPERIMENTS

Guidelines for machine learning experiments, Cross Validation (CV) and resampling - K-fold CV, bootstrapping, measuring classifier performance, assessing a single classification algorithm and comparing two classification algorithms - t test, McNemar's test, K-fold CV paired t test. **(Chapter - 5)**

TABLE OF CONTENTS

UNIT I

UNIT II

UNIT III

UNIT IV

UNIT V

UNIT I

1 Introduction to Machine Learning

Syllabus

Review of Linear Algebra for machine learning; Introduction and motivation for machine learning; Examples of machine learning applications, Vapnik-Chervonenkis (VC) dimension, Probably Approximately Correct (PAC) learning, Hypothesis spaces, Inductive bias, Generalization, Bias variance trade-off.

Contents

1.1 Review of Linear Algebra for Machine Learning

- Linear algebra is the study of linear combinations. It is the study of vector spaces, lines and planes, and some mappings that are required to perform the linear transformations. It includes vectors, matrices and linear functions. It is the study of linear sets of equations and its transformation properties.
- Linear algebra is the study of vectors and linear functions. Linear algebra is about linear combinations. That is, using arithmetic on columns of numbers called vectors and arrays of numbers called **matrices**, to create new columns and arrays of numbers.
- Linear algebra is the study of lines and planes, vector spaces and mappings that are required for linear transforms.
- The general linear equation is represented as,

$$a_1x_1 + a_2x_2 \ldots\ldots\ldots + a_nx_n = b$$

where ,

a = Represents the coefficients

x = Represents the unknowns

b = Represents the constant

- Formally, a vector space is a set of vectors which is closed under addition and multiplication by real numbers. A *subspace* is a subset of a vector space which is a vector space itself, e.g. the plane $z = 0$ is a subspace of R^3
- If all vectors in a vector space may be expressed as linear combinations of $v_1,\ldots,v_k$, then $v_1,\ldots,v_k$ span the space.
- A basis is a set of linearly independent vectors which span the space. The dimension of a space is the # of "degrees of freedom" of the space; it is the number of vectors in any basis for the space.
- A basis is a maximal set of linearly independent vectors and a minimal set of spanning vectors.
- Two vectors are orthogonal if their dot product is 0. An orthogonal basis consists of orthogonal vectors. An orthonormal basis consists of orthogonal vectors of unit length.
- Functions of several variables are often presented in one line such as,

$$f(x, y) = 3x + 5y$$

- Vector Addition : Numbers → Both 3 and 5 are numbers and so is 3 + 5

$$\text{3-vectors} : \begin{pmatrix}1\\1\\0\end{pmatrix} + \begin{pmatrix}0\\1\\1\end{pmatrix} = \begin{pmatrix}1\\2\\1\end{pmatrix}$$

- Polynomials : If $p(x) = 1+x-2x^2+3x^3$ and $q(x) = x+3x^2-3x^3+x^4$ then their sum $p(x) + q(x)$ is the new polynomial $1+2x+x^2+x^4$.
- Power series : If $f(x) = 1+x+\frac{1}{2!}x^2+\frac{1}{3!}x^3+\ldots$ and $g(x) = 1+x+\frac{1}{2!}x^2-\frac{1}{3!}x^3+\ldots$ then $f(x) + g(x) = 1+\frac{1}{2!}x^2+\frac{1}{4!}x^4+\ldots$ is also a power series.
- Functions : If $f(x) = e^x$ and $g(x) = e^{-x}$ then their sum $f(x) + g(x)$ is the new function 2 cosh x.

1.1.1 Transposes and Inner Products (dot product)

- A collection of variables may be treated as a single entity by writing them as a vector. For example, the three variables x_1, x_2 and x_3 may be written as the vector

$$x = \begin{bmatrix}x_1\\x_2\\x_3\end{bmatrix}$$

- Vectors can be written as column vectors where the variables go down the page or as row vectors where the variables go across the page.
- To turn a column vector into a row vector we use the transpose operator $X^T = [X_1, X_2, X_3]$
- The transpose operator also turns row vectors into column vectors. We now define the inner product of two vectors

$$x^T y = [x_1, x_2, x_3]\begin{bmatrix}y_1\\y_2\\y_3\end{bmatrix}$$

$$= x_1y_1 + x_2y_2 + x_3y_3$$

$$= \sum_{i=1}^{3} x_iy_i$$

which is seen to be a scalar. The outer product of two vectors produces a matrix

$$xy^T = \begin{bmatrix}x_1\\x_2\\x_3\end{bmatrix}[y_1, y_2, y_3]$$

$$= \begin{bmatrix} x_1y_1 & x_1y_2 & x_1y_3 \\ x_2y_1 & x_2y_2 & x_2y_3 \\ x_3y_1 & x_3y_2 & x_3y_3 \end{bmatrix}$$

- An N × M matrix has N rows and M columns. The ij^{th} entry of a matrix is the entry on the j^{th} column of the i^{th} row. Given a matrix A, the ij^{th} entry is written as A_{ij}. When applying the transpose operator to a matrix the i^{th} row becomes the i^{th} column. That is, if

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

Then

$$A^T = \begin{bmatrix} a_{11} & a_{21} & a_{31} \\ a_{12} & a_{22} & a_{32} \\ a_{13} & a_{23} & a_{33} \end{bmatrix}$$

- A matrix is symmetric if $A_{ij} = A_{ji}$. Another way to say this is that, for symmetric matrices, $A = A^T$.
- Two matrices can be multiplied if the number of columns in the first matrix equals the number of rows in the second.
- Multiplying A, an N × M matrix, by B, an M × K matrix, results in C, an N x K matrix. The ij^{th} entry in C is the inner product between the i^{th} row in A and the j^{th} column in B.

- Example :

$$\begin{bmatrix} 2 & 3 & 4 \\ 5 & 6 & 7 \end{bmatrix} \begin{bmatrix} 1 & 3 & 7 & 2 \\ 4 & 3 & 4 & 1 \\ 5 & 6 & 4 & 2 \end{bmatrix} = \begin{bmatrix} 34 & 39 & 42 & 15 \\ 64 & 75 & 87 & 30 \end{bmatrix}$$

Given two matrices A and B we note that

$$(AB)^T = B^T A^T$$

Example 1.1.1 *For the matrices* $A = \begin{bmatrix} 3 & -1 \\ -2 & 5 \end{bmatrix}$ *and* $B = \begin{bmatrix} 6 & 4 \\ 7 & -2 \end{bmatrix}$

Find C = AB by inner product method.

Solution : Here the matrix C will also be 2 × 2, with

$$c_{11} = [3 \quad -1] \begin{bmatrix} 6 \\ 7 \end{bmatrix} = 18 + (-7) = 11,$$

$$c_{12} = [3 \quad -1] \begin{bmatrix} 4 \\ -2 \end{bmatrix} = 12 + 2 = 14,$$

$$c_{21} = [-2 \quad 5] \begin{bmatrix} 6 \\ 7 \end{bmatrix} = -12 + 35 = 23,$$

$$c_{22} = [-2 \quad 5] \begin{bmatrix} 4 \\ -2 \end{bmatrix} = -8 + (-10) = -18,$$

$$C = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix} = \begin{bmatrix} 11 & 14 \\ 23 & -18 \end{bmatrix}$$

1.1.2 Outer Product

- In linear algebra, the outer product of two coordinate vectors is a matrix. If the two vectors have dimensions n and m, then their outer product is an n × m matrix.
- More generally, given two tensors (multidimensional arrays of numbers), their outer product is a tensor. The outer product of tensors is also referred to as their tensor product and can be used to define the tensor algebra.
- The outer product contrasts with :
  a) The dot product, which takes a pair of coordinate vectors as input and produces a scalar.
  b) The Kronecker product, which takes a pair of matrices as input and produces a block matrix.
- Let A and B be m × n and n × p matrices respectively. The product C = AB is the matrix

$$C = a_{*1}B_{1*} + a_{*2}B_{2*} + a_{*3}B_{3*} + \ldots + a_{*n}B_{n*}$$

That is, C is the m × p matrix given by the sum of all the m × p outer product matrices obtained from multiplying each column of A times the corresponding row of B.

Properties of an outer product :

1. The result of an outer product is m × n rectangular matrix.
2. The outer product is not commutative. That is, $u \otimes v \neq v \otimes u$

3. Multiply the second vector v with the resultant product $w = u \otimes v$ gives a vector of the first factor u scaled by the square norm of the second factor v. That is,

$$wv = uv^T v = u\|v\|^2$$

Example 1.1.2 *For the matrices* $A = \begin{bmatrix} 3 & -1 \\ -2 & 5 \end{bmatrix}$ *and* $B = \begin{bmatrix} 6 & 4 \\ 7 & -2 \end{bmatrix}$

Find C = AB by outer product method

Solution :

$$C = a_{*1}B_{1*} + a_{*2}B_{2*} = \begin{bmatrix} 3 \\ -2 \end{bmatrix} [6 \quad 4] + \begin{bmatrix} -1 \\ 5 \end{bmatrix} [7 \quad -2]$$

$$= \begin{bmatrix} 18 & 12 \\ -12 & -8 \end{bmatrix} + \begin{bmatrix} -7 & 2 \\ 35 & -10 \end{bmatrix}$$

$$C = \begin{bmatrix} 11 & 14 \\ 23 & -18 \end{bmatrix}$$

1.1.3 Inverse

Given a matrix X its inverse X^{-1} is defined by the properties

$$X^{-1}X = I$$

$$XX^{-1} = I$$

where I is the identity matrix. The inverse of a diagonal matrix with entries d_{ij} is another diagonal matrix with entries $1/d_{ij}$. This satisfies the definition of an inverse,

$$\begin{bmatrix} 4 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 6 \end{bmatrix} \begin{bmatrix} 1/4 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1/6 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- If X has no inverse, we say X is singular or non-invertible.

Properties of the Inverse :

1. If A is a square matrix and B is the inverse of A, then A is the inverse of B, since AB = I = BA. So we have the identity $(A^{-1})^{-1} = A$.
2. Notice that $B^{-1}A^{-1}AB = B^{-1}IB = I = ABB^{-1}A^{-1}$, so $(AB)^{-1} = B^{-1}A^{-1}$

1.1.4 Eigen Values and Eigen Vectors

- The eigen vectors x and eigen values λ of a matrix A satisfy → $Ax = \lambda x$.
- If A is an n × n matrix, then x is an n × 1 vector and λ is a constant. The equation can be rewritten as $(A - \lambda I)\, x = 0$, where I is the n × n identity matrix.
- Since x is required to be nonzero, the eigen values must satisfy $\det(A - \lambda I) = 0$, which is called the characteristic equation.
- Solving it for values of λ gives the eigen values of matrix A.
- When $AX = \lambda X$ for some $X \neq 0$, we call such an X an **eigen vector** of the matrix A. The eigen vectors of A are associated to an eigen value. Hence, if λ_1 is an eigen value of A and $AX = \lambda_1 X$, we can label this eigen vector as X_1. Note again that in order to be an eigen vector, X must be nonzero.
- There is also a geometric significance to eigen vectors. When you have a **nonzero** vector which, when multiplied by a matrix results in another vector which is parallel to the first or equal to **0**, this vector is called an eigen vector of the matrix. This is the meaning when the vectors are in $\mathbb{R}^n$.

Example 1.1.3 *Find all eigen values and their eigen space for*

$$A = \begin{bmatrix} 3 & -2 \\ 1 & 0 \end{bmatrix}$$

Solution :

$$A - \lambda I = \begin{bmatrix} 3 & -2 \\ 1 & 0 \end{bmatrix} - \lambda \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 3 & -2 \\ 1 & 0 \end{bmatrix} - \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix}$$

$$= \begin{bmatrix} 3-\lambda & -2 \\ 1 & -\lambda \end{bmatrix}$$

The characteristic equation is

$$\det(A - \lambda I) = (3-\lambda)(-\lambda) - (-2) = 0$$

$$\lambda^2 - 3\lambda + 2 = 0$$

$$(\lambda - 1)(\lambda - 2) = 0$$

We find eigen values

$$\lambda_1 = 1, \lambda_2 = 2$$

We next find eigen vectors associated with each eigen value. For $\lambda_1 = 1$,

$$\bar{0} = (A - \lambda_1 I)\, \bar{x} = \begin{bmatrix} 3-1 & -2 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 2 & -2 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Example 1.1.4 *Given that 2 is an eigen value for* $A = \begin{bmatrix} 4 & -1 & 6 \\ 2 & 1 & 6 \\ 2 & -1 & 8 \end{bmatrix}$

Find a basis of its eigen space.

Solution :

$$A - 2I = \begin{bmatrix} 4-2 & -1 & 6 \\ 2 & 1-2 & 6 \\ 2 & -1 & 8-2 \end{bmatrix} = \begin{bmatrix} 2 & -1 & 6 \\ 2 & -1 & 6 \\ 2 & -1 & 6 \end{bmatrix} \rightarrow \begin{bmatrix} 2 & -1 & 6 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Therefore, $(A - 2I)\vec{x} = \vec{0}$ becomes

$$2x_1 - x_2 + 6x_3 = 0 \quad \text{or} \quad x_2 = 2x_1 + 6x_3,$$

where we select x_1 and x_3 as free variables only to avoid fractions. Solution set in parametric form is

$$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} x_1 \\ 2x_1 + 6x_3 \\ x_3 \end{bmatrix} = x_1 \begin{bmatrix} 1 \\ 2 \\ 0 \end{bmatrix} + x_3 \begin{bmatrix} 0 \\ 6 \\ 1 \end{bmatrix}$$

A basis for the eigen space :

$$\vec{u}_1 = \begin{bmatrix} 1 \\ 2 \\ 0 \end{bmatrix} \quad \text{and} \quad \vec{u}_2 = \begin{bmatrix} 0 \\ 6 \\ 1 \end{bmatrix}$$

Example 1.1.5 *Find all eigen values for*

$$A = \begin{bmatrix} 5 & -2 & 6 & -1 \\ 0 & 3 & -8 & 0 \\ 0 & 0 & 5 & 4 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

Solution :

$$A - \lambda I = \begin{bmatrix} 5-\lambda & -2 & 6 & -1 \\ 0 & 3-\lambda & -8 & 0 \\ 0 & 0 & 5-\lambda & 4 \\ 0 & 0 & 1 & 1-\lambda \end{bmatrix}$$

$$\det (A - \lambda I) = (5-\lambda) \det \begin{bmatrix} 3-\lambda & -8 & 0 \\ 0 & 5-\lambda & 4 \\ 0 & 1 & 1-\lambda \end{bmatrix}$$

$$= (5-\lambda)(3-\lambda) \det \begin{bmatrix} 5-\lambda & 4 \\ 1 & 1-\lambda \end{bmatrix}$$

$$= (5-\lambda)(3-\lambda)[(5-\lambda)(1-\lambda) - 4] = 0$$

There are 4 roots :

$$(5-\lambda) = 0 \Rightarrow \lambda = 5$$

$$(3-\lambda) = 0 \Rightarrow \lambda = 3$$

$$(5-\lambda)(1-\lambda) - 4 = 0 \Rightarrow \lambda^2 - 6\lambda + 1 = 0$$

$$\Rightarrow \lambda = \frac{6 \pm \sqrt{36-4}}{2} = 3 \pm 2\sqrt{2}$$

Example 1.1.6 *For the given matrix A, find out eigen values and eigen vectors*

$$A = \begin{bmatrix} -6 & 3 \\ 5 & 5 \end{bmatrix}$$

Solution :

$$A - \lambda I = \begin{bmatrix} -6 & 3 \\ 5 & 5 \end{bmatrix} - \lambda \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} -6 & 3 \\ 5 & 5 \end{bmatrix} - \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix}$$

$$= \begin{bmatrix} -6-\lambda & 3 \\ 5 & 5-\lambda \end{bmatrix}$$

$$= (-6-\lambda)(5-\lambda) - (5)(3)$$

$$= \lambda^2 + \lambda - 45$$

$$= (\lambda - 6.22)(\lambda + 7.22)$$

1.1.5 Singular Values and Singular Vectors

- A singular value and pair of singular vectors of a square or rectangular matrix A are a nonnegative scalar σ and two nonzero vectors u and v so that

$$A_v = \sigma u,$$

$$A^H u = \sigma v$$

- The term "eigen value" is a partial translation of the German "eigenwert." A complete translation would be something like "own value" or "characteristic value," but these are rarely used. The term "singular value" relates to the distance between a matrix and the set of singular matrices.

- Eigen values play an important role in situations where the matrix is a transformation from one vector space onto itself. Systems of linear ordinary differential equations are the primary examples.
- The values of λ can correspond to frequencies of vibration or critical values of stability parameters or energy levels of atoms.
- Singular values play an important role where the matrix is a transformation from one vector space to a different vector space, possibly with a different dimension. Systems of over-or underdetermined algebraic equations are the primary examples.
- The definitions of eigen vectors and singular vectors do not specify their normalization. An eigen vector x or a pair of singular vectors u and v, can be scaled by any nonzero factor without changing any other important properties.
- Eigen vectors of symmetric matrices are usually normalized to have Euclidean length equal to one, $\|x\|_2 = 1$.
- On the other hand, the eigen vectors of nonsymmetric matrices often have different normalizations in different contexts.
- Singular vectors are almost always normalized to have Euclidean length equal to one, $\|u\|_2 = \|v\|_2 = 1$. You can still multiply eigen vectors or pairs of singular vectors, by – 1 without changing their lengths.

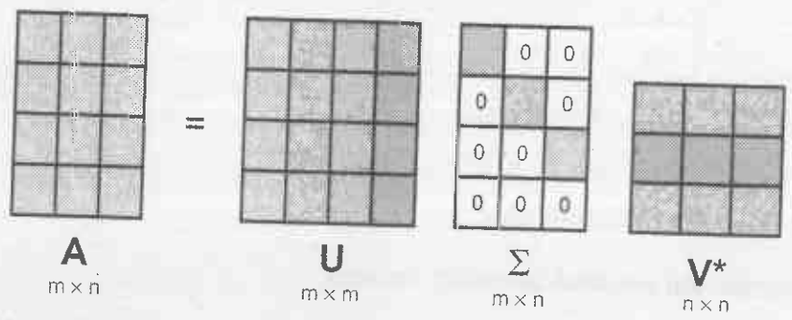


Fig. 1.1.1

- SVD decomposes a matrix into three other matrixes. SVD factors a single matrix into matrix U, D and V^* respectively.

where

U and V^* are orthogonal matrices.

D is a diagonal matrix of singular values.

1.2 Introduction and Motivation for Machine Learning

- Machine Learning (ML) is a sub-field of Artificial Intelligence (AI) which concerns with developing computational theories of learning and building learning machines.
- **Learning** is a phenomenon and process which has manifestations of various aspects. Learning process includes gaining of new symbolic knowledge and development of cognitive skills through instruction and practice. It is also discovery of new facts and theories through observation and experiment.
- **Machine Learning Definition :** 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.
- Machine learning is programming computers to optimize a performance criterion using example data or past experience. Application of machine learning methods to large databases is called **data mining.**
- It is very hard to write programs that solve problems like recognizing a human face. We do not know what program to write because we don't know how our brain does it. Instead of writing a program by hand, it is possible to collect lots of examples that specify the correct output for a given input.
- A machine learning algorithm then takes these examples and produces a program that does the job. The program produced by the learning algorithm may look very different from a typical hand-written program. It may contain millions of numbers. If we do it right, the program works for new cases as well as the ones we trained it on.
- Main goal of machine learning is to devise learning algorithms that do the learning automatically without human intervention or assistance. The machine learning paradigm can be viewed as "programming by example." Another goal is to develop computational models of human learning process and perform computer simulations.
- The goal of machine learning is to build computer systems that can adapt and learn from their experience.
- Algorithm is used to solve a problem on computer. An algorithm is a sequence of instruction. It should carry out to transform the input to output. For example, for addition of four numbers is carried out by giving four number as input to the algorithm and output is sum of all four numbers. For the same task, there may be various algorithms. It is interested to find the most efficient one, requiring the least number of instructions or memory or both.
- For some tasks, however, we do not have an algorithm.

How Machines Learn ?

- Machine learning typically follows three phases :
  1. **Training :** A training set of examples of correct behavior is analyzed and some representation of the newly learnt knowledge is stored. This is some form of rules.
  2. **Validation :** The rules are checked and, if necessary, additional training is given. Sometimes additional test data are used, but instead, a human expert may validate the rules, or some other automatic knowledge - based component may be used. The role of the tester is often called the **opponent**.
  3. **Application :** The rules are used in responding to some new situation.
- Fig. 1.2.1 shows phases of ML.

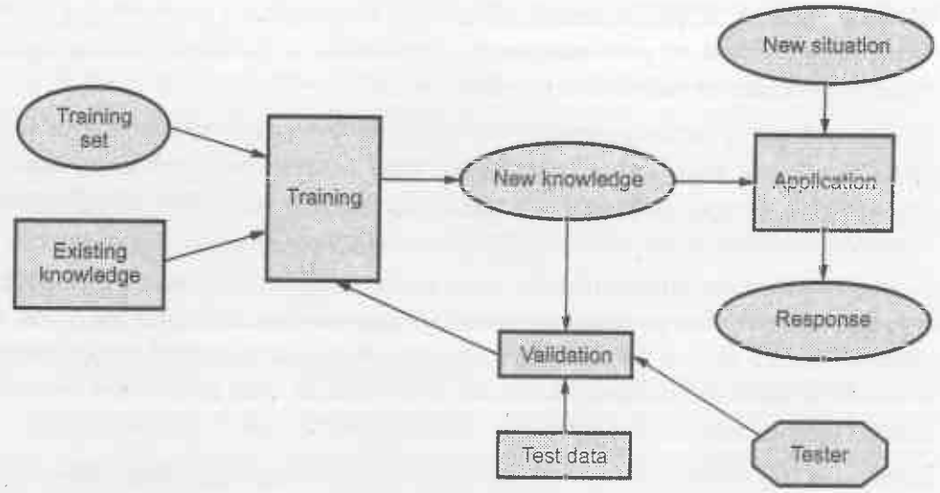


Fig. 1.2.1 Phases of ML

1.2.1 Why Machine Learning is Important ?

- Machine learning algorithms can figure out how to perform important tasks by generalizing from examples.
- Machine learning provides business insight and intelligence. Decision makers are provided with greater insights into their organizations. This adaptive technology is being used by global enterprises to gain a competitive edge.
- Machine learning algorithms discover the relationships between the variables of a system (input, output and hidden) from direct samples of the system.
- **Following are some of the reasons :**
  1. Some tasks cannot be defined well, except by examples. For example : Recognizing people.

  2. Relationships and correlations can be hidden within large amounts of data. To solve these problems, machine learning and data mining may be able to find these relationships.
  3. Human designers often produce machines that do not work as well as desired in the environments in which they are used.
  4. The amount of knowledge available about certain tasks might be too large for explicit encoding by humans.
  5. Environments change time to time.
  6. New knowledge about tasks is constantly being discovered by humans.
- Machine learning also helps us find solutions of many problems in computer vision, speech recognition and robotics. Machine learning uses the theory of statistics in building mathematical models, because the core task is making inference from a sample.
- Learning is used when :
  1. Human expertise does not exist (navigating on Mars),
  2. Humans are unable to explain their expertise (speech recognition)
  3. Solution changes in time (routing on a computer network)
  4. Solution needs to be adapted to particular cases (user biometrics)

1.2.2 Ingredients of Machine Learning

The ingredients of machine learning are as follows :

1. **Tasks :** The problems that can be solved with machine learning. A task is an abstract representation of a problem. The standard methodology in machine learning is to learn one task at a time. Large problems are broken into small, reasonably independent sub-problems that are learned separately and then recombined.
   - Predictive tasks perform inference on the current data in order to make predictions. Descriptive tasks characterize the general properties of the data in the database.
2. **Models :** The output of machine learning. Different models are geometric models, probabilistic models, logical models, grouping and grading.
   - The model-based approach seeks to create a modified solution tailored to each new application. Instead of having to transform your problem to fit some standard algorithm, in model-based machine learning you design the algorithm precisely to fit your problem.
   - Model is just made up of set of assumptions, expressed in a precise mathematical form. These assumptions include the number and types of variables in the problem

domain, which variables affect each other, and what the effect of changing one variable is on another variable.

- Machine learning models are classified as : Geometric model, probabilistic model and logical model.

3. **Features :** The workhorses of machine learning. A good feature representation is central to achieving high performance in any machine learning task.

- Feature extraction starts from an initial set of measured data and builds derived values intended to be informative, non redundant, facilitating the subsequent learning and generalization steps.
- Feature selection is a process that chooses a subset of features from the original features so that the feature space is optimally reduced according to a certain criterion.

1.3 Types of Machine

- Learning is essential for unknown environments, i.e. when designer lacks the omniscience. Learning simply means incorporating information from the training examples into the system.
- Learning is any change in a system that allows it to perform better the second time on repetition of the same task or on another task drawn from the same population. One part of learning is acquiring knowledge and new information; and the other part is problem-solving.
- Supervised and Unsupervised Learning are the different types of machine learning methods. A computational learning model should be clear about the following aspects :
  1. **Learner :** Who or what is doing the learning. For example : Program or algorithm.
  2. **Domain :** What is being learned ?
  3. **Goal :** Why the learning is done ?
  4. **Representation :** The way the objects to be learned are represented.
  5. **Algorithmic technology :** The algorithmic framework to be used.
  6. **Information source :** The information (training data) the program uses for learning.
  7. **Training scenario :** The description of the learning process.
- Learning is constructing or modifying representation of what is being experienced. Learn means to get knowledge of by study, experience or being taught.

- Machine learning is a scientific discipline concerned with the design and development of the algorithm that allows computers to evolve behaviors based on empirical data, such as form sensors data or database.
- Machine learning is usually divided into two main types : Supervised learning and unsupervised learning.

Why do Machine Learning ?

1. To understand and improve efficiency of human learning.
2. Discover new things or structure that is unknown to humans (Example : Data mining).
3. Fill in skeletal or incomplete specifications about a domain.

1.3.1 Supervised Learning

- Supervised learning is the machine learning task of inferring a function from supervised training data. The training data consist of a set of training examples. The task of the supervised learner is to predict the output behavior of a system for any set of input values, after an initial training phase.
- **Supervised learning** in which the network is trained by providing it with input and matching output patterns. These input-output pairs are usually provided by an external teacher.
- Human learning is based on the past experiences. A computer does not have experiences.
- A computer system learns from data, which represent some "past experiences" of an application domain.
- To learn a target function that can be used to predict the values of a discrete class attribute, e.g., approve or not-approved and high-risk or low risk. The task is commonly called : Supervised learning, classification or inductive learning.
- Training data includes both the input and the desired results. For some examples the correct results (targets) are known and are given in input to the model during the learning process. The construction of a proper training, validation and test set is crucial. These methods are usually fast and accurate.
- Have to be able to generalize : give the correct results when new data are given in input without knowing a priori the target.
- Supervised learning is the machine learning task of inferring a function from supervised training data. The training data consist of a set of training examples. In supervised learning, each example is a pair consisting of an input object and a desired output value.

- A supervised learning algorithm analyzes the training data and produces an inferred function, which is called a classifier or a regression function. Fig. 1.3.1 shows supervised learning process.

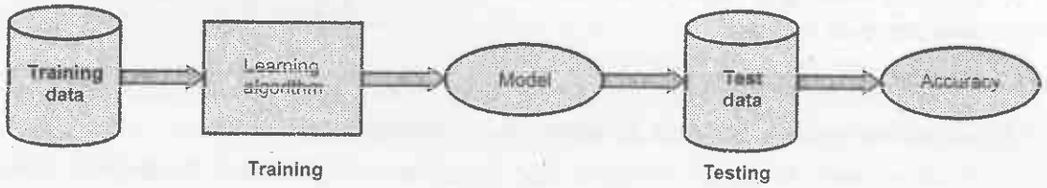


Fig. 1.3.1 Supervised learning process

- The learned model helps the system to perform task better as compared to no learning.
- Each input vector requires a corresponding target vector.
  Training Pair = (Input Vector, Target Vector)
- Fig. 1.3.2 shows input vector.

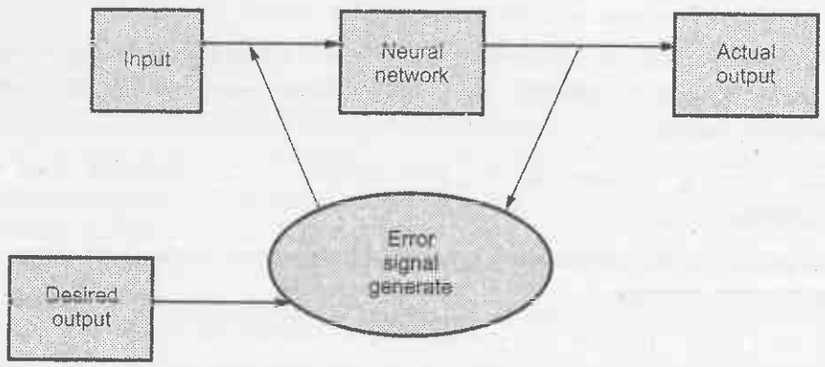


Fig. 1.3.2 Input vector

- Supervised learning denotes a method in which some input vectors are collected and presented to the network. The output computed by the net-work is observed and the deviation from the expected answer is measured. The weights are corrected according to the magnitude of the error in the way defined by the learning algorithm.
- Supervised learning is further divided into methods which use reinforcement or error correction. The perceptron learning algorithm is an example of supervised learning with reinforcement.

- In order to solve a given problem of supervised learning, following steps are performed :
  1. Find out the type of training examples.
  2. Collect a training set.
  3. Determine the input feature representation of the learned function.
  4. Determine the structure of the learned function and corresponding learning algorithm.
  5. Complete the design and then run the learning algorithm on the collected training set.
  6. Evaluate the accuracy of the learned function. After parameter adjustment and learning, the performance of the resulting function should be measured on a test set that is separate from the training set.

1.3.2 Unsupervised Learning

- The model is not provided with the correct results during the training. It can be used to cluster the input data in classes on the basis of their statistical properties only. Cluster significance and labeling.
- The labeling can be carried out even if the labels are only available for a small number of objects representative of the desired classes. All similar inputs patterns are grouped together as clusters.
- If matching pattern is not found, a new cluster is formed. There is no error feedback.
- External teacher is not used and is based upon only local information. It is also referred to as **self-organization**.
- They are called unsupervised because they do not need a teacher or super-visor to label a set of training examples. Only the original data is required to start the analysis.
- In contrast to supervised learning, unsupervised or self-organized learning does not require an external teacher. During the training session, the neural network receives a number of different input patterns, discovers significant features in these patterns and learns how to classify input data into appropriate categories.
- Unsupervised learning algorithms aim to learn rapidly and can be used in real-time. Unsupervised learning is frequently employed for data clustering, feature extraction etc.

- Another mode of learning called **recording learning** by Zurada is typically employed for associative memory networks. An associative memory networks is designed by recording several idea patterns into the networks stable states.

1.3.3 Difference between Supervised and Unsupervised Learning

Sr. No.	Supervised learning	Unsupervised learning
1.	Desired output is given.	Desired output is not given.
2.	It is not possible to learn larger and more complex models than with supervised learning.	It is possible to learn larger and more complex models with unsupervised learning.
3.	Use training data to infer model	No training data is used.
4.	Every input pattern that is used to train the network is associated with an output pattern.	The target output is not presented to the network.
5.	Trying to predict a function from labeled data.	Try to detect interesting relations in data.
6.	Supervised learning requires that the target variable is well defined and that a sufficient number of its values are given.	For unsupervised learning typically either the target variable is unknown or has only been recorded for too small a number of cases.
7.	**Example** : Optical character recognition.	**Example** : Find a face in an image.
8.	We can test our model.	We can not test our model.
9.	Supervised learning is also called classification.	Unsupervised learning is also called clustering.

1.3.4 Semi-supervised Learning

- **Semi-supervised** learning uses both labeled and unlabeled data to improve supervised learning. The goal is to learn a predictor that predicts future test data better than the predictor learned from the labeled training data alone.
- **Semi-supervised** learning is motivated by its practical value in learning faster, better and cheaper.
- In many real world applications, it is relatively easy to acquire a large amount of unlabeled data x.
- For example, documents can be crawled from the Web, images can be obtained from surveillance cameras, and speech can be collected from broadcast. However, their corresponding labels y for the prediction task, such as sentiment orientation,

intrusion detection and phonetic transcript, often requires slow human annotation and expensive laboratory experiments.

- In many practical learning domains, there is a large supply of unlabeled data but limited labeled data, which can be expensive to generate. For example : Text processing, video-indexing, bioinformatics etc.
- **Semi-supervised** Learning makes use of both labeled and unlabeled data for training, typically a small amount of labeled data with a large amount of unlabeled data. When unlabeled data is used in conjunction with a small amount of labeled data, it can produce considerable improvement in learning accuracy.
- **Semi-supervised** learning sometimes enables predictive model testing at reduced cost.
- **Semi-supervised classification :** Training on labeled data exploits additional unlabeled data, frequently resulting in a more accurate classifier.
- **Semi-supervised clustering :** Uses small amount of labeled data to aid and bias the clustering of unlabeled data.

1.3.5 Reinforced Learnings

- User will get immediate feedback in supervised learning and no feedback from unsupervised learning. But in the reinforced learning, you will get delayed scalar feedback.
- Reinforcement learning is learning what to do and how to map situations to actions. The learner is not told which actions to take. Fig. 1.3.3 shows concept of reinforced learning.

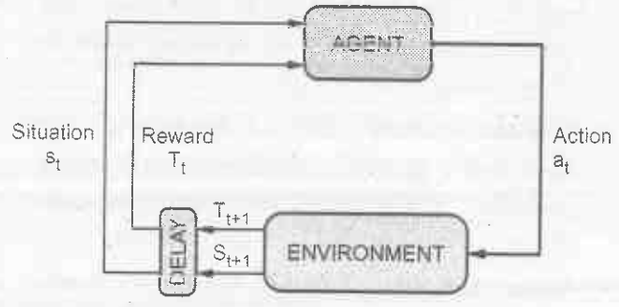


Fig. 1.3.3 Reinforced learning

- Reinforced learning is deals with agents that must sense and act upon their environment. It combines classical Artificial Intelligence and machine learning techniques.
- It allows machines and software agents to automatically determine the ideal behavior within a specific context, in order to maximize its performance. Simple reward feedback is required for the agent to learn its behavior; this is known as the reinforcement signal.

- Two most important distinguishing features of reinforcement learning is trial-and-error and delayed reward.
- With reinforcement learning algorithms an agent can improve its performance by using the feedback it gets from the environment. This environmental feedback is called the reward signal.
- Based on accumulated experience, the agent needs to learn which action to take in a given situation in order to obtain a desired long term goal. Essentially actions that lead to long term rewards need to reinforced. Reinforcement learning has connections with control theory, Markov decision processes and game theory.
- **Example of reinforcement learning :** A mobile robot decides whether it should enter a new room in search of more trash to collect or start trying to find its way back to its battery recharging station. It makes its decision based on how quickly and easily it has been able to find the recharger in the past.

1.3.5.1 Elements of Reinforcement Learning

- Reinforcement learning elements are as follows :
  1. Policy 2. Reward function
  3. Value function 4. Model of the environment
- Fig. 1.3.4 shows elements of RL.
- **Policy :** Policy defines the learning agent behavior for given time period. It is a mapping from perceived states of the environment to actions to be taken when in those states.
- **Reward function :** Reward function is used to define a goal in a reinforcement learning problem. It also maps each perceived state of the environment to a single number.
- **Value function :** Value functions specify what is good in the long run. The value of a state is the total amount of reward an agent can expect to accumulate over the future, starting from that state.

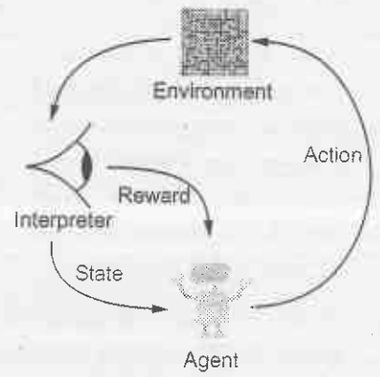


Fig. 1.3.4 : Elements of reinforcement learning

- **Model of the environment :** Models are used for planning.
- **Credit assignment problem :** Reinforcement learning algorithms learn to generate an internal value for the intermediate states as to how good they are in leading to the goal.

- The learning decision maker is called the **agent**. The agent interacts with the environment that includes everything outside the agent.
- The agent has sensors to decide on its state in the environment and takes an action that modifies its state.
- The reinforcement learning problem model is an agent continuously interacting with an environment. The agent and the environment interact in a sequence of time steps. At each time step t, the agent receives the state of the environment and a scalar numerical reward for the previous action, and then the agent then selects an action.
- Reinforcement Learning is a technique for solving Markov decision problems.
- Reinforcement learning uses a formal framework defining the interaction between a learning agent and its environment in terms of states, actions, and rewards. This framework is intended to be a simple way of representing essential features of the artificial intelligence problem.

1.3.6 Difference between Supervised, Unsupervised and Reinforcement Learning

Supervised learning	Unsupervised learning	Reinforcement learning
Supervised learning requires that the target variable is well defined and that a sufficient number of its values are given.	For unsupervised learning typically either the target variable is unknown or has only been recorded for too small a number of cases.	Reinforcement learning is learning what to do and how to map situations to actions. The learner is not told which actions to take.
Supervised Learning deals with two main tasks regression and classification.	Unsupervised learning deals with clustering and associative rule mining problems.	Reinforcement Learning deals with exploitation or exploration, Markov's decision processes, Policy Learning, Deep Learning and value learning.
The input data in supervised learning in labelled data.	Unsupervised learning uses unlabelled data.	The data is not predefined in reinforcement learning.
Learns by using labelled data.	Trained using unlabelled data without any guidance.	Works on interacting with the environment.
Maps the labeled inputs to the known outputs.	Understands patterns and discovers the output.	Follows the trial and error method.

1.4 Examples of Machine Learning Applications

- Examples of successful applications of machine learning :

Here are several examples :

1 **Optical character recognition :** Categorize images of handwritten characters by the letters represented.

2 **Face detection :** Find faces in images (or indicate if a face is present)

3 **Spam filtering :** Identify email messages as spam or non-spam topic spotting: categorize news articles (say) as to whether they are about politics, sports, entertainment, etc.

4 **Spoken language understanding :** Within the context of a limited domain, determine the meaning of something uttered by a speaker to the extent that it can be classified into one of a fixed set of categories.

1.4.1 Face Recognition and Medical Diagnosis

Face recognition

- Face recognition task is effortlessly and every day we recognize our friends, relative and family members. We also recognition by looking at the photographs. In photographs, they are in different pose, hair styles, background light, makeup and without makeup.
- We do it subconsciously and cannot explain how we do it. Because we can't explain how we do it, we can't write an algorithm.
- Face has some structure. It is not a random collection of pixel. It is symmetric structure. It contains predefined components like nose, mouth, eye, ears. Every person face is a pattern composed of a particular combination of the features. By analyzing sample face images of a person, a learning program captures the pattern specific to that person and users it to recognize if a new real face or new image belongs to this specific person or not.
- Machine learning algorithm creates an optimized model of the concept being learned based on data or past experience.
- In the case of face recognition, the input is an image, the classes are people to be recognized and the learning program should learn to associate the face images to identities. This problem is more difficult than optical character recognition because there are more classes, input image is larger and a face is 3D and differences in pose and lighting cause significant changes in the image.

Medical diagnosis

- In medical diagnosis, the input are the relevant information about the patient and the classes are the illness. The inputs contain the age of patient's, gender, past medical history and current symptoms.

- Some tests may not have been applied to the patient and thus these inputs would be missing. Tests take time, may be costly and may inconvience the patient so we do not want to apply them unless we believe that they will give us valuable information.
- In the case of a medical diagnosis, a wrong decision may lead to a wrong or no treatment and in cases of doubt it is preferable that the classifier reject and defer decision to a human expert.

1.4.2 Google Home and Amazon Alexa

Amazon Alexa / Siri

- Every time Alexa or Siri make a mistake when responding to our request, it uses the data it receives based on how it responded to the original query to improve the next time. If an error was made, it takes that data and learns from it. If the response was favourable, the system notes that as well.
- Data and machine learning are responsible for the explosive growth of digital voice assistants. They continue o get better with the more experiences they have and the data they accumulate.
- When user make a request of Alexa, the microphone on the device records command. This recording is sent to over the internet to the cloud. If user are talking to Alexa, the recording is sent to Alexa Voice Services (AVS). This cloud-based service will review the recording and interpret user request. Then, the system will send a relevant response back to the device.
- Amazon breaks down user "orders" into individual sounds. It then consults a database containing various words' pronunciations to find which words most closely correspond to the combination of individual sounds.
- It then identifies important words to make sense of the tasks and carry out corresponding functions. For instance, if Alexa notices words like "sport" or "basketball", it would open the sports app.
- Amazon's servers send the information back to our device and Alexa may speak. If Alexa needs to say anything back, it would go through the same process described above, but in reverse order.

Google Home :

- Google services such as its image search and translation tools use sophisticated machine learning which allow computers to see, listen and speak in much the same way as human do.
- To perform its functions, Google Assistant relies on Artificial Intelligence (AI) technologies such as natural language processing and machine learning to understand what the user is saying and to make suggestions or act on that language input.

- The Google Home can play music, but it's primarily designed as a vehicle for Google Assistant -- Google's voice - activated virtual helper that's connected to the internet.
- The Google Home is always listening to its environment, but it won't record what we are saying or respond to our commands until we speak one of its pre-programmed wake words -- either "OK, Google" or "Hey, Google."
- TF-IDF, is a numerical statistic that is intended to reflect how important a word is to a document from collection corpus. It is often used as a weighting factor for searches of information retrieval, text mining and user modeling.
- The TD-IDF value increases proportionally to the number of times a word appears in the document but it is often offset by the frequency of the word in the corpus, which helps to adjust for the fact that some words appear more frequently.

1.4.3 Unmanned Vehicles

- An Unmanned Aerial Vehicle (UAV), sometimes known as a drone, is an aircraft or airborne system that is controlled remotely by an onboard computer or a human operator. The ground control station, aircraft components, and various types of sensors make up the UAV system.
- UAVs are categorized depending on their endurance, weight and altitude range. They can be used for multiple commercial and military applications.
- Machine learning is the process of using, storing and finding patterns within massive amounts of data, which can eventually be fed into algorithms. It's basically a process of using the data accumulated by the machine or device that allows computers to develop their own algorithm so that humans won't have to create challenging algorithms manually.
- Unmanned ground vehicles are classified into two broad types, remotely operated and autonomous.
- Autonomous unmanned ground vehicles comprise several technologies that allow the machine to be self - acting and self - regulating, sans human intervention. The technology was initially developed to aid ground forces in the transfer of heavy equipment.
- However, the technology has witnessed significant evolution over the years, giving rise to more tactical vehicles designed to assist in surveillance or IED search-and-destroy missions.
- For example, unmanned ships in the course of the voyage, the default route is to ensure the obstruction of the premise of a straight line navigation.

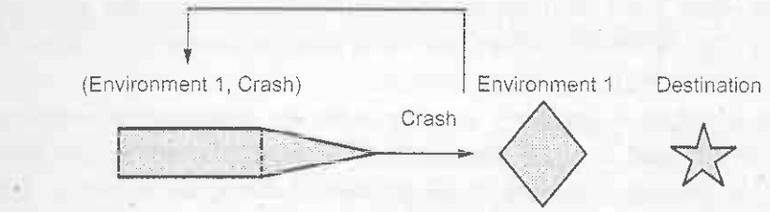


Fig. 1.4.1 First time lane

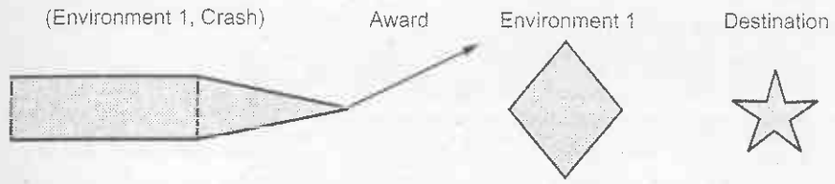


Fig. 1.4.2 The second time

- During the course of the voyage, the hull is changed by the intensity and direction of the waves and is unpredictable. It is clear for the unmanned boat itself.
- Therefore, unmanned ships in the process of navigation, continue to train the perception of the surrounding environment and make the appropriate strategy, if the results of the implementation of the strategy in line with the default route to be rewarded.

1.5 Vapnik-Chervonenkis (VC) Dimension

- Vapnik-Chervonenkis (VC) dimension provides a measure of the complexity of a space of functions, and which allows the probably approximately correct framework to be extended to spaces containing an infinite number of functions.
- The Vapnik-Chervonenkis dimension is a measure of the complexity or capacity of a class of functions $f(\alpha)$. The VC dimension measures the largest number of examples that can be explained by the family $f(\alpha)$.
- The Vapnik-Chervonenkis dimension, VC(H), of hypothesis space H defined over instance space X is the size of the largest finite subset of X shattered by H. If arbitrarily large finite sets of X can be shattered by H, then $VC(H) = \infty$.
- The basic argument is that high capacity and generalization properties are at odds :
  1. If the family $f(\alpha)$ has enough capacity to explain every possible dataset, we should not expect these functions to generalize very well.

2. On the other hand, if functions f(α) have small capacity but they are able to explain our particular dataset, we have stronger reasons to believe that they will also work well on unseen data.

- **Shattering a set of examples :** Assume a binary classification problem with N examples in R^D and consider the set of $2^{|N|}$ possible dichotomies. For instance, with N = 3 examples, the set of all possible dichotomies is **{(000), (001), (010), (011), (100), (101), (110), (111)}**. A class of functions f(α) is said to shatter the dataset if, for every possible dichotomy, there is a function in f(α) that models it.
- Consider as an example a finite concept class $C = \{c_1, \ldots, c_4\}$ applied to three instance vectors with the results :

	X_1	X_2	X_3
c_1	1	1	1
c_2	0	1	1
c_3	1	0	0
c_4	0	0	0

Then :

$\pi_c(\{X_1\}) = \{(0), (1)\}$ shattered

$\pi_c(\{X_1, X_3\}) = \{(0, 0), (0, 1), (1, 0), (1, 1)\}$ shattered

$\pi_c(\{X_2, X_3\}) = \{(0, 0), (1, 1)\}$ not shattered

- The **VC dimension VC(f)** is the size of the largest dataset that can be shattered by the set of functions f (α). If the VC dimension of (α) is h, then there exists at least one set of h points that can be shattered by (α), but in general it will not be true that every set of h points can be shattered.
- **Example of VCdim : axis aligned rectangles**
- If we had five points, then at most four of the points determine the minimal rectangle that contains the whole set. Then no rectangle is consistent with the labeling that assigns these four boundary points "+" and assigned the remaining point a "-". Therefore,

  VCdim (axis-aligned rectangles ε R^2) = 4
- The VC dimension cannot be accurately estimated for non-linear models such as neural networks. The VC dimension may be infinite requiring infinite amount of data.

1.6 Probably Approximately Correct (PAC) Learning

- Computational learning theory provides a formal framework in which to precisely formulate and address questions regarding the performance of different learning algorithms so that careful comparisons of both the predictive power and the computational efficiency of alternative learning algorithms can be made.
- Three key aspects that must be formalized are the way in which the learner interacts with its environment the definition of successfully completing the learning task and a formal definition of efficiency of both data usage (sample complexity) and processing time (time complexity).

Probably Approximately Correct Learning :

- A concept class C is said to be PAC learnable using a hypothesis class H if there exists a learning algorithm L such that for all concepts in C, for all instance distributions D on an instance space X, $\forall \varepsilon, \delta\ (0 < \varepsilon, \delta < 1)$, L, when given access to the Example oracle, produces, with probability at least $(1 - \delta)$, a hypothesis h from H with error no more than ε.
- PCA is a nice formalism for deciding how much data you need to collect in order for a given classifier to achieve a given probability of correct predictions on a given fraction of future test data.
- To understand what this model is all about, it's probably easiest just to give an example. Say there's a hidden line on the chalk board.
- Given a point on the board, we need to classify whether it's above or below the line. To help, we'll get some sample data, which consists of random points on the board and whether each point is above or below the line.
- After seeing, say, twenty points, you won't know exactly where the line is, but you'll probably know roughly where it is. And using that knowledge, you'll be able to predict whether most future points lie above or below the line.
- Suppose we have agreed that predicting the right answer "most of the time" is okay. Is any random choice of twenty points going to give you that ability ? No, because you could get really unlucky with the sample data, and it could tell you almost nothing about where the line is. Hence the "Probably" in PAC.
- X is the set of all possible examples. *D* is the distribution from which the examples are drawn
- H is the set of all possible hypotheses, $c \in H$
- *m* is the number of training examples. Then

  error(h) = Pr(h(x) ≠ c(x) | x is drawn from X with D)

  where h is approximately correct if *error(h)* $\leq \varepsilon$

- *Hypothesis h(X) is consistent with m examples and has an error of at most with probability* 1 – δ. This is a worst-case analysis. Note that the result is independent of the distribution *D*.
- **Curse of dimensionality :** If the number of features d is large, the number of samples n, may be too small for accurate parameter estimation.
- For accurate estimation, n should be much bigger than d^2, otherwise model is too complicated for the data, overfitting
- **Consistent learner :** A consistent learner is one that returns some hypothesis h from the hypothesis class H that is consistent with a random sequence of m examples. A consistent learner is a MAP learner, if all hypotheses are a-priori equally likely.

1.7 Hypothesis Spaces

- Search Space : The space of all feasible solutions is called search space. Each point in the search space represent one feasible solution. Each feasible solution can be "marked" by its value or fitness for the problem.
- If we are solving some problem, we are usually looking for some solution, which will be the best among others.
- We are looking for our solution, which is one point or more among feasible solutions, that is one point in the search space.
- The looking for a solution is then equal to a looking for some extreme (minimum or maximum) in the search space. The search space can be whole known by the time of solving a problem, but usually we know only a few points from it and we are generating other points as the process of finding solution continues.
- Genetic algorithm employs a randomized beam search method to seek a maximally fit hypothesis.

Motivation

- The solution(s) to machine learning tasks are often called **hypotheses**, because they can be expressed as a hypothesis that the observed positives and negatives for a categorization is explained by the concept learned for the solution.
- The hypotheses have to be represented in some representation scheme, and, as usual with AI tasks, this will have a big effect on many aspects of the learning methods.
- General definition of a hypothesis : "A hypothesis is a statement of a relationship between two or more variables."
- Sometimes, it is necessary to evaluate the performance of learned hypotheses.

Reason for using hypotheses :

- Learning from a limited-size database indicating the effectiveness of different medical treatments, it is important to understand as precisely as possible the accuracy of the learned hypotheses.
- The evaluating hypotheses are an integral component of many learning methods.
- It is important to understand the likely errors inherent in estimating the accuracy of the pruned and unpruned tree.
- Estimating the accuracy of a hypothesis is relatively straightforward when data is plentiful.
- An estimator is any random variable used to estimate some parameter of the underlying population from which a sample is drawn.
  1. The estimation bias of an estimator Y for an arbitrary parameter p is E[Y] – p, If the estimation bias is 0, then Y is an unbiased estimator for p.
  2. The variance of an estimator Y for an arbitrary parameter p is simply the variance of Y.

1.7.1 Population Evolution and the Schema Theorem

- It is developed by John Holland.
- If a chromosome is of length n then it contains 3^n schemata (as each position can have the value 0, 1 or *)
- In theory, this means that for a population of M individuals we are evaluating up to $M3^n$ schemata.
- But, bear in mind that some schemata will not be represented and others will overlap with other schemata.
- This is exactly what we want. We eventually want to create a population that is full of fitter schemata and we will have lost weaker schemata. It is the fact that we are manipulating M individuals but $M3^n$ schemata that gives genetic algorithms what has been called *implicit parallelism*.
- *Length :* is defined as the distance between the start of the schema and the end of the schema minus one.
- *Order :* is defined as the number of defined positions.
- *Fitness ratio :* is defined as the ratio of the fitness of a schema to the average fitness of the population.

*	*	0	*	0	*	1

Here length = 6 and order = 3

- In this analysis we assume that the GA is a way of processing genotype features rather then genotypes themselves - a feature being simply a set of values in specific positions. A particular feature is defined in terms of a **schema**.
- This is a genotype-like string with specific values in some positions and 'don't care' values (asterisks) in others. An example is : *10**0**** .
- This schema has ten characters in all, including seven "don't care" values. It will match any 10-character genotype with a 1 in the second position, a 0 in the third position and a 0 in the sixth position.
- The longer the length of the schema, the more chance there is of the schema being disrupted by a crossover operation. This implies that shorter schemata have a better chance of surviving from one generation to the next. In turn, this implies that if we know that certain attributes of a problem fit well together then these should be placed as close as possible together in the coding.
- This observation is also true for the order of the chromosome. If we are not worried about the number of defined positions (i.e. we allow as many '*' as possible) then a crossover operation has less chance of disrupting good schemata. Intuitively, it would seem better to have short, low-order schema. This is only based on empirical evidence but it is widely believed that these assumptions are true and the following theory makes some sense of this.
- Using a technique where we choose parents relative to their fitness (e.g. roulette wheel selection), fitter schema should find their way from one generation to another.
- Intuitively, if a schema is fitter than average then it should not only survive to the next generation but should also increase its presence in the population. If Φ is the number of instances of any particular schema S within the population at time t, then at t + 1 we would expect

  $\Phi(S, t+1) > \Phi(S)$

to hold for above average fitness schemata.

- Going one stage further we can estimate the number of schema present at t + 1

$$\Phi(S, t+1) = \Phi(S, t)\, n \frac{f(S)}{\sum f_i}$$

where

n is the size of the population

f(S) is the fitness of the schema

$\sum f_i$ is the fitness of the population

$$\Phi(S, t+1) = \Phi(S, t) \frac{f(s)}{f_{avg}}$$

f_{avg} is the average fitness of the population.

- If a particular schema stays a constant, c, above the average we can say even more about the **effects of reproduction**

$$\Phi(S, t+1) = \Phi(S, t) \frac{(f_{avg} + c\, f_{avg})}{f_{avg}}$$

$$= \Phi(S, t)\,(1 + c)$$

- Setting t = 0

  $\Phi(S, t) = \Phi(S, t)(1+c)^t$

- Notice that the number of schema rises exponentially.

1.7.2 Sample Error and True Error

- The sample error ($err_s(h)$) of h with respect to target function (f) and data sample (S) is the proportion of examples h misclassifies. The sample test error is the mean error over the test sample.

$$err_s(h) = \frac{1}{n} \sum_{i=1}^{n} L(f(x_i), h(x_i))$$

- The **true error** of hypothesis h with respect to target function f and distribution D is the probability that h will misclassify an instance drawn at random according to D.

  $Err(h) = E\,[L(f(x), h(x))]$

- An $error_D(h)$ is the **true error** of hypothesis h with respect to the target function f and data distribution D. It is the probability h will misclassify an instance drawn at random according to D.
- An $error_s(h)$ is the **sample error** of hypothesis h with respect to the target function f and data sample set S. It is the proportion of examples in S that h misclassifies.

1.8 Inductive Bias

- The Candidate-Elimination algorithm will converge toward the true target concept provided it is given accurate training examples and provided its initial hypothesis space contains the target concept.
- What if the target concept is not contained in the hypothesis space ?
- Can we avoid this difficulty by using a hypothesis space that includes every possible hypothesis ?

- How does the size of this hypothesis space influence the ability of the algorithm to generalize to unobserved instances ?
- How does the size of the hypothesis space influence the number of training examples that must be observed ?
- In EnjoySport example, we restricted the hypothesis space to include only conjunctions of attribute values. Because of this restriction, the hypothesis space is unable to represent even simple disjunctive target concepts such as "Sky = Sunny or Sky = Cloudy."

Example	Sky	Air Temp	Humidity	Wind	Water	Forecast	Enjoy Sport
1	Sunny	Warm	Normal	Strong	Cool	Change	YES
2	Cloudy	Warm	Normal	Strong	Cool	Change	YES
3	Rainy	Warm	Normal	Strong	Cool	Change	NO

- From first two examples : S2 : <?, Warm, Normal, Strong, Cool, Change>
- This is inconsistent with third examples, and there are no hypotheses consistent with these three examples PROBLEM : We have biased the learner to consider only conjunctive hypotheses. We require a more expressive hypothesis space.
- The obvious solution to the problem of assuring that the target concept is in the hypothesis space H is to provide a hypothesis space capable of representing every teachable concept.

Inductive Bias - Fundamental Property of Inductive Inference :

- A learner that makes no a priori assumptions regarding the identity of the target concept has no rational basis for classifying any unseen instances.
- Inductive Leap : A learner should be able to generalize training data using prior assumptions in order to classify unseen instances.
- The generalization is known as inductive leap and our prior assumptions are the inductive bias of the learner.
- Inductive Bias (prior assumptions) of Candidate-Elimination algorithm is that the target concept can be represented by a conjunction of attribute values, the target concept is contained in the hypothesis space and training examples are correct.

Inductive Bias - Formal Definition

- Consider a concept learning algorithm L for the set of instances X. Let c be an arbitrary concept defined over X, and let Dc = {<x , c(x)>} be an arbitrary set of training examples of c.

- Let $L(x_i, D_c)$ denote the classification assigned to the instance x_i by L after training on the data D_c.
- The inductive bias of L is any minimal set of assertions B such that for any target concept c and corresponding training examples D_c the following formula holds.

$$(\forall x_i \in X)[(B \wedge D_c \wedge x_i) \angle (x_i, D_c)]$$

Three Learning Algorithms :

- ROTE-LEARNER : Learning corresponds simply to storing each observed training example in memory. Subsequent instances are classified by looking them up in memory. If the instance is found in memory, the stored classification is returned. Otherwise, the system refuses to classify the new instance. **Inductive Bias : No inductive bias**
- CANDIDATE-ELIMINATION : New instances are classified only in the case where all members of the current version space agree on the classification. Otherwise, the system refuses to classify the new instance. **Inductive Bias : the target concept can be represented in its hypothesis space.**
- FIND-S : This algorithm, described earlier, finds the most specific hypothesis consistent with the training examples. It then uses this hypothesis to classify all subsequent instances. **Inductive Bias: the target concept can be represented in its hypothesis space, and all instances are negative instances unless the opposite is entailed by its other knowledge.**

1.9 Bias Variance Trade-Off

- In the experimental practice we observe an important phenomenon called the bias variance dilemma.
- In supervised learning, the class value assigned by the learning model built based on the training data may differ from the actual class value. This error in learning can be of two types, errors due to 'bias' and error due to 'variance'.
- Fig. 1.9.1 shows bias-variance trade off.
- Give two classes of hypothesis (e.g. linear models and k-NNs) to fit to some training data set, we observe that the more flexible hypothesis class has a low bias term but a higher variance term. If we have parametric family of hypothesis, then we can increases the flexibility of the hypothesis but we still observe the increase of variance.

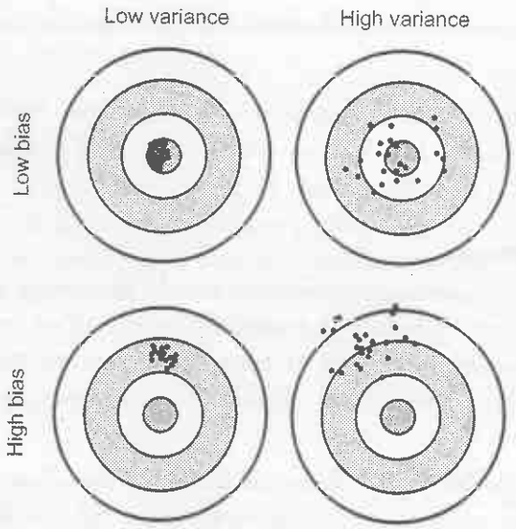


Fig. 1.9.1 Bias-variance trade off

- The **bias-variance-dilemma** is the problem of simultaneously minimizing two sources of error that prevent supervised learning algorithm from generalizing beyond their training set :
  1. The bias is error from erroneous assumptions in the learning algorithm. High bias can cause an algorithm to miss the relevant relations between features and target outputs.
  2. The variance is error from sensitivity to small fluctuations in the training set. High variance can cause overfitting : modeling the random noise in the training data, rather than the intended outputs.
- In order to reduce the model error, the designer can aim at reducing either the bias or the variance, as the noise components is irreducible.
- As the model increases in complexity, its bias is likely to diminish. However, as the number of training examples is kept fixed, the parametric identification of the model may strongly vary from one DN to another. This will increase the variance term.
- At one stage, the decrease in bias will be inferior to the increase in variance, warning that the model should not be too complex. Conversely, to decrease the variance term, the designer has to simplify its model so that it is less sensitive to a specific training set. This simplification will lead to a higher bias.

Example 1.9.1

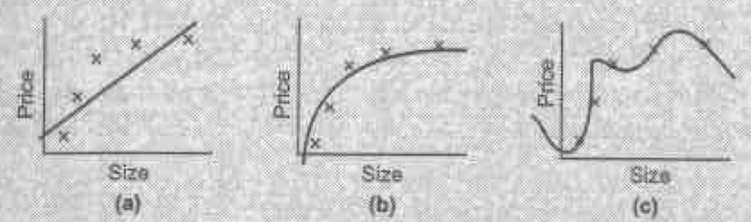


Fig. 1.9.2

Explain the above Fig. 1.9.2 (a), (b) and (c).

Solution :

- Given Fig. 1.9.2 is related to overfitting and underfitting.

Underfitting (High bias and low variance) :

- A statistical model or a machine learning algorithm is said to have underfitting when it cannot capture the underlying trend of the data.
- It usually happens when we have less data to build an accurate model and also when we try to build a linear model with a non-linear data.

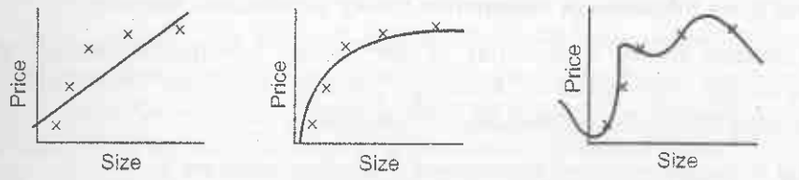


Fig. 1.9.3

- In such cases the rules of the machine learning model are too easy and flexible to be applied on 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.

Overfitting (High variance and low bias) :

- A statistical model is said to be overfitted, when we train it with a lot of data.
- When a model gets trained with so much of data, it starts learning from the noise and inaccurate data entries in our data set.

- 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.

1.10 Two Marks Questions with Answers

Q.1 Define learning.

Ans. : Learning is a phenomenon and process which has manifestations of various aspects. Learning process includes gaining of new symbolic knowledge and development of cognitive skills through instruction and practice. It is also discovery of new facts and theories through observation and experiment.

Q.2 Define machine learning.

Ans. : 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.

Q.3 What is an influence of information theory on machine learning ?

Ans. : Information theory is measures of entropy and information content. Minimum description length approaches to learning. Optimal codes and their relationship to optimal training sequences for encoding a hypothesis.

Q.4 What is meant by target function of a learning program ?

Ans. : Target function is a method for solving a problem that an AI algorithm parses its training data to find. Once an algorithm finds its target function, that function can be used to predict results. The function can then be used to find output data related to inputs for real problems where, unlike training sets, outputs are not included.

Q.5 Define useful perspective on machine learning.

Ans. : One useful perspective on machine learning is that it involves searching a very large space of possible hypotheses to determine one that best fits the observed data and any prior knowledge held by the learner.

Q.6 Describe the issues in machine learning ?

Ans. : Issues of machine learning are as follows :

- What learning algorithms to be used ?
- How much training data is sufficient ?
- When and how prior knowledge can guide the learning process ?

- What is the best strategy for choosing a next training experience ?
- What is the best way to reduce the learning task to one or more function approximation problems ?
- How can the learner automatically alter its representation to improve its learning ability ?

Q.7 What is decision tree ?

Ans. : • Decision tree learning is a method for approximating discrete-valued target functions, in which the learned function is represented by a decision tree.

- A decision tree is a tree where each node represents a feature(attribute), each link(branch) represents a decision(rule) and each leaf represents an outcome(categorical or continues value).
- A decision tree or a classification tree is a tree in which each internal node is labeled with an input feature. The arcs coming from a node labeled with a feature are labeled with each of the possible values of the feature.

Q.8 What are the nodes of decision tree ?

Ans. : • A decision tree has two kinds of nodes

1. Each leaf node has a class label, determined by majority vote of training examples reaching that leaf.
2. Each internal node is a question on features. It branches out according to the answers.

- Decision tree learning is a method for approximating discrete-valued target functions. The learned function is represented by a decision tree

Q.9 Why tree pruning useful in decision tree induction ?

Ans. : When a decision tree is built, many of the branches will reflect anomalies in the training data due to noise or outliers. Tree pruning methods address this problem of **overfitting** the data. Such methods typically use statistical measures to remove the least reliable branches.

Q.10 What is tree pruning ?

Ans. : Tree pruning attempts to identify and remove such branches, with the goal of improving classification accuracy on unseen data

Q.11 What is RULE POST-PRUNING ?

Ans. : • It is method for finding high accuracy hypotheses.

- Rule post-pruning involves the following steps :
  1. Infer decision tree from training set
  2. Convert tree to rules - one rule per branch

3. Prune each rule by removing preconditions that result in improved estimated accuracy
4. Sort the pruned rules by their estimated accuracy and consider them in this sequence when classifying unseen instances

Q.12 Why convert the decision tree to rules before pruning ?

Ans. : • Converting to rules allows distinguishing among the different contexts in which a decision node is used.

- Converting to rules removes the distinction between attribute tests that occur near the root of the tree and those that occur near the leaves.
- Converting to rules improves readability. Rules are often easier for to understand

Q.13 Define probably approximately correct learning.

Ans. : A concept class C is said to be PAC learnable using a hypothesis class H if there exists a learning algorithm L such that for all concepts in C, for all instance distributions D on an instance space X, $\forall \varepsilon, \delta (0 < \varepsilon, \delta < 1)$, L, when given access to the Example oracle, produces, with probability at least $(1 - \delta)$, a hypothesis h from H with error no more than ε.

Q.14 What is inductive learning ?

Ans. : In inductive learning, the learner is given a hypothesis space H from which it must select an output hypothesis and a set of training examples -
$D = \{(x_1, f(x_1)) \ldots, (x_n, f(x_n))\}$ where $f(x_i)$ is the target value for the instance x_i.
The desired output of the learner is a hypothesis h from H that is consistent with these training examples.

❑❑❑

UNIT II

2 Supervised Learning

Syllabus

Linear Regression Models : Least squares, single & multiple variables, Bayesian linear regression, gradient descent, Linear Classification Models : Discriminant function - Perceptron algorithm, Probabilistic discriminative model - Logistic regression, Probabilistic generative model - Naive Bayes, Maximum margin classifier - Support vector machine, Decision Tree, Random Forests

Contents

2.1 Regression

- Regression finds correlations between dependent and independent variables. If the desired output consists of one or more continuous variable, then the task is called as regression.
- Therefore, regression algorithms help predict continuous variables such as house prices, market trends, weather patterns, oil and gas prices etc.
- Fig. 2.1.1 shows regression.

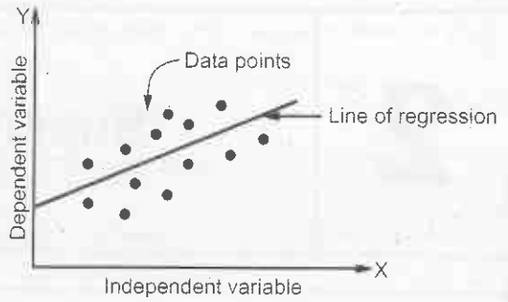


Fig. 2.1.1 Regression

- When the targets in a dataset are real numbers, the machine learning task is known as regression and each sample in the dataset has a real-valued output or target.
- Regression analysis is a set of statistical methods used for the estimation of relationships between a dependent variable and one or more independent variables. It can be utilized to assess the strength of the relationship between variables and for modelling the future relationship between them.
- The two basic types of regression are linear regression and multiple linear regression.

2.1.1 Linear Regression Models

- Linear regression is a statistical method that allows us to summarize and study relationships between two continuous (quantitative) variables.
- The objective of a linear regression model is to find a relationship between the input variables and a target variable.
  1. One variable, denoted x, is regarded as the predictor, explanatory or independent variable.
  2. The other variable, denoted y, is regarded as the response, outcome or dependent variable.
- Regression models predict a continuous variable, such as the sales made on a day or predict temperature of a city. Let's imagine that we fit a line with the training point that we have. If we want to add another data point, but to fit it, we need to change existing model.
- This will happen with each data point that we add to the model; hence, linear regression isn't good for classification models.

- Regression estimates are used to explain the relationship between one dependent variable and one or more independent variables. Classification predicts categorical labels (classes), prediction models continuous - valued functions. Classification is considered to be supervised learning.
- Classifies data based on the training set and the values in a classifying attribute and uses it in classifying new data. Prediction means models continuous - valued functions, i.e. predicts unknown or missing values.
- **The regression line** gives the average relationship between the two variables in mathematical form.
- For two variables X and Y, there are always two lines of regression.
- **Regression line of X on Y** Gives the best estimate for the value of X for any specific given values of Y :

$$X = a + b\,Y$$

where
- a = X - intercept
- b = Slope of the line
- X = Dependent variable
- Y = Independent variable

- **Regression line Y on X :** Gives the best estimate for the value of Y for any specific given values of X :

$$Y = a + bx$$

where
- a = Y - intercept
- b = Slope of the line
- Y = Dependent variable
- x = Independent variable

- By using the least squares method (a procedure that minimizes the vertical deviations of plotted points surrounding a straight line) we are able to construct a best fitting straight line to the scatter diagram points and then formulate a regression equation in the form of :

$$\hat{y} = a + bX$$

$$\hat{y} = \overline{y} + b(x - \overline{x})$$

- Regression analysis is the art and science of fitting straight lines to patterns of data. In a linear regression model, the

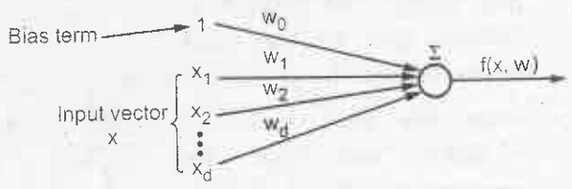


Fig. 2.1.2

variable of interest ("dependent" variable) is predicted from k other variables ("independent" variables) using a linear equation. If Y denotes the dependent variable and $X_1,\ldots,X_k$, are the independent variables, then the assumption is that the value of Y at time t in the data sample is determined by the linear equation :

$$Y_1 = \beta_0 + \beta_1 X_{1t} + \beta_2 X_{2t} + \ldots + \beta_k X_{kt} + \varepsilon_t$$

where the betas are constants and the epsilons are independent and identically distributed normal random variables with mean zero.

- At each split point, the "error" between the predicted value and the actual values is squared to get a "Sum of Squared Errors (SSE)". The split point errors across the variables are compared and the variable/point yielding the lowest SSE is chosen as the root node/split point. This process is recursively continued.
- Error function measures how much our predictions deviate from the desired answers.

Mean-squared error $J_n = \frac{1}{n} \sum_{i=1\ldots n} (y_i - f(x_i))^2$

Advantages :

a. Training a linear regression model is usually much faster than methods such as neural networks.

b. Linear regression models are simple and require minimum memory to implement.

c. By examining the magnitude and sign of the regression coefficients you can infer how predictor variables affect the target outcome.

2.1.2 Least Squares

- The method of least squares is about estimating parameters by minimizing the squared discrepancies between observed data, on the one hand, and their expected values on the other.
- Considering an arbitrary straight line, $y = b_0 + b_1 x$, is to be fitted through these data points. The question is "Which line is the most representative" ?
- What are the values of b_0 and b_1 such that the resulting line "best" fits the data points ? But, what

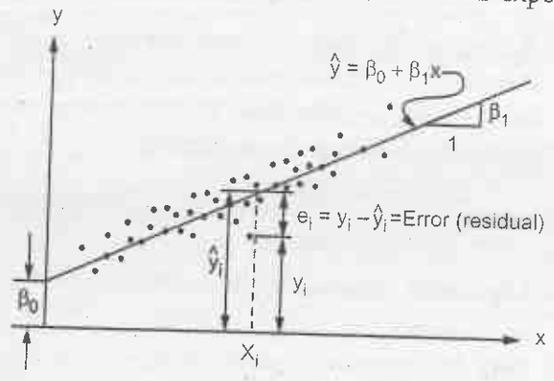


Fig. 2.1.3

goodness-of-fit criterion to use to determine among all possible combinations of b_0 and b_1 ?

- The Least Squares (LS) criterion states that the sum of the squares of errors is minimum. The least-squares solutions yields **y(x)** whose elements sum to 1, but do not ensure the outputs to be in the range [0,1].
- How to draw such a line based on data points observed ? Suppose a imaginary line of y = a + bx.
- Imagine a vertical distance between the line and a data point E = Y – E(Y).
- This error is the deviation of the data point from the imaginary line, regression line. Then what is the best values of a and b ? A and b that minimizes the sum of such errors.

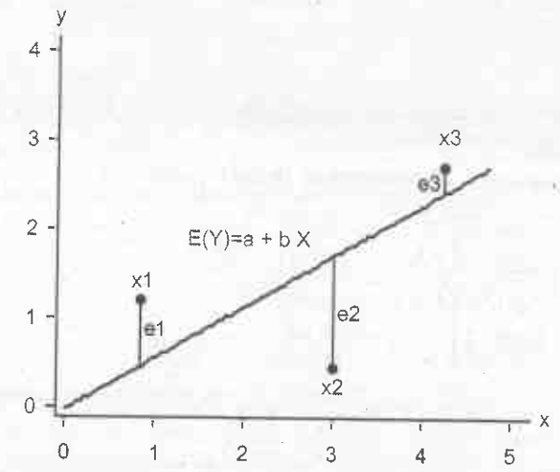


Fig. 2.1.4

- Deviation does not have good properties for computation. Then why do we use squares of deviation ? Let us get a and b that can minimize the sum of squared deviations rather than the sum of deviations. This method is called **least squares.**
- Least squares method minimizes the sum of squares of errors. Such a and b are called least squares estimators i.e. estimators of parameters α and β.
- The process of getting parameter estimators (e.g., a and b) is called **estimation.** Lest squares method is the estimation method of Ordinary Least Squares (OLS).

Disadvantages of least square

1. Lack robustness to outliers
2. Certain datasets unsuitable for least squares classification
3. Decision boundary corresponds to ML solution

Example 2.1.1 *Fit a straight line to the points in the table. Compute m and b by least squares.*

Points	x	y
A	3.00	4.50
B	4.25	4.25
C	5.50	5.50
D	8.00	5.50

Solution : Represent in matrix form :

$$\begin{bmatrix} 3.00 & 1 \\ 4.25 & 1 \\ 5.50 & 1 \\ 8.00 & 1 \end{bmatrix} \begin{bmatrix} m \\ b \end{bmatrix} = \begin{bmatrix} 4.50 \\ 4.25 \\ 5.50 \\ 5.50 \end{bmatrix} + \begin{bmatrix} v_A \\ v_B \\ v_C \\ v_D \end{bmatrix}$$

$$X = \begin{bmatrix} m \\ b \end{bmatrix} = (A^T A)^{-1} (A^T L)$$

$$= \begin{bmatrix} 121.3125 & 20.7500 \\ 20.7500 & 4.0000 \end{bmatrix}^{-1} \begin{bmatrix} 105.8125 \\ 19.7500 \end{bmatrix} = \begin{bmatrix} 0.246 \\ 3.663 \end{bmatrix}$$

$$V = AX - L$$

$$= \begin{bmatrix} 3.00 & 1 \\ 4.25 & 1 \\ 5.50 & 1 \\ 8.00 & 1 \end{bmatrix} \begin{bmatrix} 0.246 \\ 3.663 \end{bmatrix} - \begin{bmatrix} 4.50 \\ 4.25 \\ 5.50 \\ 5.50 \end{bmatrix} = \begin{bmatrix} -0.10 \\ 0.46 \\ -0.48 \\ 0.13 \end{bmatrix}$$

2.1.3 Multiple Regression

- Regression analysis is used to predict the value of one or more responses from a set of predictors. It can also be used to estimate the linear association between the predictors and responses. Predictors can be continuous or categorical or a mixture of both.
- If multiple independent variables affect the response variable, then the analysis calls for a model different from that used for the single predictor variable. In a situation where more than one independent factor (variable) affects the outcome of a process, a multiple regression model is used. This is referred to as multiple linear regression model or multivariate least squares fitting.

- Let $z_1 ; z_2 ; \ldots ; z_r$ be a set of r predictors believed to be related to a response variable Y. The linear regression model for the j^{th} sample unit has the form

$$Y_j = \beta_0 + \beta_1 z_{j1} + \beta_2 z_{j2} + \ldots + \beta_r z_{jr} + \varepsilon_j$$

where ε is a random error and β_i, $i = 0, 1, \ldots, r$ are un-known regression coefficients.

- With n independent observations, we can write one model for each sample unit so that the model is now

$$Y = Z\beta + \varepsilon$$

where Y is $n \times 1$, Z is $n \times (r+1)$, β is $(r+1) \times 1$ and ε is $n \times 1$

- In order to estimate β, we take a least squares approach that is analogous to what we did in the simple linear regression case.
- In matrix form, we can arrange the data in the following form :

$$X = \begin{bmatrix} 1 & x_{11} & x_{12} & \ldots & x_{1k} \\ 1 & x_{21} & x_{22} & \ldots & x_{2k} \\ \vdots & \vdots & & & \vdots \\ 1 & x_{N1} & x_{N2} & \ldots & x_{Nk} \end{bmatrix} \quad Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \quad \hat{a} = \begin{bmatrix} \hat{\beta}_0 \\ \hat{\beta}_1 \\ \vdots \\ \hat{\beta}_k \end{bmatrix}$$

where $\hat{\beta}_j$ are the estimates of the regression coefficients.

2.1.4 Difference between Simple Regression and Multiple Regression

Simple regression	Multiple regression
One dependent variable Y predicted from one independent variable X	One dependent variable Y predicted from a set of independent variables ($X_1, X_2 \ldots X_k$)
One regression coefficient	One regression coefficient for each independent variable
r^2 : Proportion of variation in dependent variable Y predictable from X	R^2 : Proportion of variation in dependent variable Y predictable by set of independent variables (X's)

2.1.5 Bayesian Linear Regression

- Bayesian linear regression allows a useful mechanism to deal with insufficient data, or poor distributed data. It allows user to put a prior on the coefficients and on the noise so that in the absence of data, the priors can take over. A prior is a distribution on a parameter.
- If we could flip the coin an infinite number of times, inferring its bias would be easy by the law of large numbers. However, what if we could only flip the coin a handful of times? Would we guess that a coin is biased if we saw three heads in

three flips, an event that happens one out of eight times with unbiased coins? The MLE would overfit these data, inferring a coin bias of p =1.

- A Bayesian approach avoids overfitting by quantifying our prior knowledge that most coins are unbiased, that the prior on the bias parameter is peaked around one-half. The data must overwhelm this prior belief about coins.
- Bayesian methods allow us to estimate model parameters, to construct model forecasts and to conduct model comparisons. Bayesian learning algorithms can calculate explicit probabilities for hypotheses.
- Bayesian classifiers use a simple idea that the training data are utilized to calculate an observed probability of each class based on feature values.
- When Bayesian classifier is used for unclassified data, it uses the observed probabilities to predict the most likely class for the new features.
- Each observed training example can incrementally decrease or increase the estimated probability that a hypothesis is correct.
- Prior knowledge can be combined with observed data to determine the final probability of a hypothesis. In Bayesian learning, prior knowledge is provided by asserting a prior probability for each candidate hypotheses and a probability distribution over observed data for each possible hypothesis.
- Bayesian methods can accommodate hypotheses that make probabilistic predictions. New instances can be classified by combining the predictions of multiple hypotheses, weighted by their probabilities.
- Even in cases where Bayesian methods prove computationally intractable, they can provide a standard of optimal decision making against which other practical methods can be measured.
- Uses of Bayesian classifiers are as follows :
  1. Used in text-based classification for finding spam or junk mail filtering.
  2. Medical diagnosis.
  3. Network security such as detecting illegal intrusion.
- The basic procedure for implementing Bayesian Linear Regression is :
  i) Specify priors for the model parameter.
  ii) Create a model mapping the training inputs to the training outputs.
  iii) Have a Markov Chain Monte Carlo (MCMC) algorithm draw samples from the posterior distributions for the parameters

2.1.6 Gradient Descent

- Goal : Solving minimization nonlinear problems through derivative information
- First and second derivatives of the objective function or the constraints play an important role in optimization. The first order derivatives are called the **gradient** and the second order derivatives are called the **Hessian matrix**.
- Derivative based optimization is also called **nonlinear**. Capable of determining search directions" according to an objective function's derivative information.
- Derivative based optimization methods are used for :
  1. Optimization of nonlinear neuro-fuzzy models
  2. Neural network learning
  3. Regression analysis in nonlinear models
- Basic descent methods are as follows :
  1. Steepest descent
  2. Newton-Raphson method

Gradient Descent :

- Gradient descent is a first-order optimization algorithm. To find a local minimum of a function using gradient descent, one takes steps proportional to the negative of the gradient of the function at the current point.
- Gradient descent is popular for very large-scale optimization problems because it is easy to implement, can handle black box functions, and each iteration is cheap.
- Given a differentiable scalar field f (x) and an initial guess x_1 , gradient descent iteratively moves the guess toward lower values of "f" by taking steps in the direction of the negative gradient $-\nabla f(x)$.
- Locally, the negated gradient is the steepest descent direction, i.e., the direction that x would need to move in order to decrease "f" the fastest. The algorithm typically converges to a local minimum, but may rarely reach a saddle point, or not move at all if x_1 lies at a local maximum.
- The gradient will give the slope of the curve at that x and its direction will point to an increase in the function. So we change x in the opposite direction to lower the function value :

$$X_{k+1} = x_k - \lambda \nabla f(x_k)$$

The $\lambda > 0$ is a small number that forces the algorithm to make small jumps

Limitations of Gradient Descent :

- Gradient descent is relatively slow close to the minimum : technically, its asymptotic rate of convergence is inferior to many other methods.
- For poorly conditioned convex problems, gradient descent increasingly 'zigzags' as the gradients point nearly orthogonally to the shortest direction to a minimum point

Steepest Descent :

- Steepest descent is also known as gradient method.
- This method is based on first order Taylor series approximation of objective function. This method is also called saddle point method. Fig. 2.1.5 shows steepest descent method.

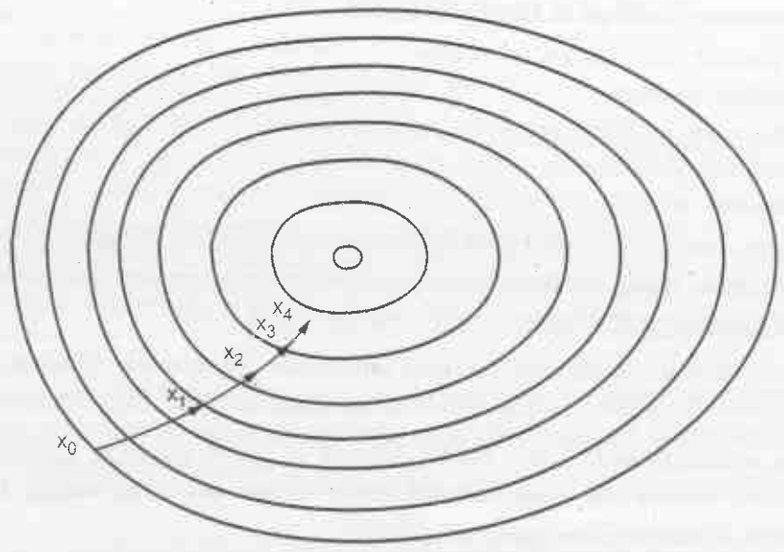


Fig. 2.1.5 Steepest descent method

- The Steepest Descent is the simplest of the gradient methods. The choice of direction is where f decreases most quickly, which is in the direction opposite to $\nabla f(x_i)$. The search starts at an arbitrary point x0 and then go down the gradient, until reach close to the solution.
- The method of steepest descent is the discrete analogue of gradient descent, but the best move is computed using a local minimization rather than computing a gradient. It is typically able to converge in few steps but it is unable to escape local minima or plateaus in the objective function.
- The gradient is everywhere perpendicular to the contour lines. After each line minimization the new gradient is always orthogonal to the previous step direction.

Consequently, the iterates tend to zig-zag down the valley in a very inefficient manner.

- The method of Steepest Descent is simple, easy to apply, and each iteration is fast. It also very stable; if the minimum points exist, the method is guaranteed to locate them after at least an infinite number of iterations.

2.2 Linear Classification Models

- A classification algorithm (Classifier) that makes its classification based on a linear predictor function combining a set of weights with the feature vector.
- A linear classifier does classification decision based on the value of a linear combination of the characteristics. Imagine that the linear classifier will merge into it's weights all the characteristics that define a particular class.
- Linear classifiers can represent a lot of things, but they can't represent everything. The classic example of what they can't represent is the XOR function.

2.2.1 Discriminant Function

- Linear Discriminant Analysis (LDA) is the most commonly used dimensionality reduction technique in supervised learning. Basically, it is a preprocessing step for pattern classification and machine learning applications. LDA is a powerful algorithm that can be used to determine the best separation between two or more classes.
- LDA is a supervised learning algorithm, which means that it requires a labelled training set of data points in order to learn the linear discriminant function.
- The main purpose of LDA is to find the line or plane that best separates data points belonging to different classes. The key idea behind LDA is that the decision boundary should be chosen such that it maximizes the distance between the means of the two classes while simultaneously minimizing the variance within each class's data or within-class scatter. This criterion is known as the Fisher criterion.
- LDA is one of the most widely used machine learning algorithms due to its accuracy and flexibility. LDA can be used for a variety of tasks such as classification, dimensionality reduction, and feature selection.

- Suppose we have two classes and we need to classify them efficiently, then using LDA, classes are divided as follows :

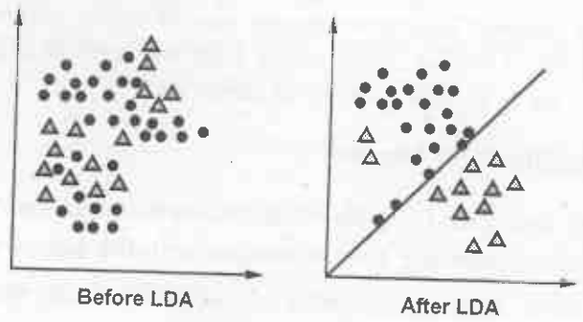


Fig. 2.2.1 LDA

- LDA algorithm works based on the following steps :
  a) The first step is to calculate the means and standard deviation of each feature.
  b) Within class scatter matrix and between class scatter matrix is calculated
  c) These matrices are then used to calculate the eigenvectors and eigenvalues.
  d) LDA chooses the k eigenvectors with the largest eigenvalues to form a transformation matrix.
  e) LDA uses this transformation matrix to transform the data into a new space with k dimensions.
  f) Once the transformation matrix transforms the data into new space with k dimensions, LDA can then be used for classification or dimensionality reduction
- Benefits of using LDA :
  a) LDA is used for classification problems.
  b) LDA is a powerful tool for dimensionality reduction.
  c) LDA is not susceptible to the "curse of dimensionality" like many other machine learning algorithms.

2.2.2 Logistic Regression

- Logistic regression is a form of regression analysis in which the outcome variable is binary or dichotomous. A statistical method used to model dichotomous or binary outcomes using predictor variables.
- **Logistic component :** Instead of modeling the outcome, Y, directly, the method models the log odds (Y) using the logistic function.

- **Regression component :** Methods used to quantify association between an outcome and predictor variables. It could be used to build predictive models as a function of predictors.
- **In simple** logistic regression, logistic regression with 1 predictor variable.

Logistic Regression :

$$\ln\left(\frac{P(Y)}{1-P(Y)}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_k X_k$$

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_k X_k + \varepsilon$$

- With logistic regression, the response variable is an indicator of some characteristic, that is, a 0/1 variable. Logistic regression is used to determine whether other measurements are related to the presence of some characteristic, for example, whether certain blood measures are predictive of having a disease.
- If analysis of covariance can be said to be a t test adjusted for other variables, then logistic regression can be thought of as a chi-square test for homogeneity of proportions adjusted for other variables. While the response variable in a logistic regression is a 0/1 variable, the logistic regression equation, which is a linear equation, does not predict the 0/1 variable itself.

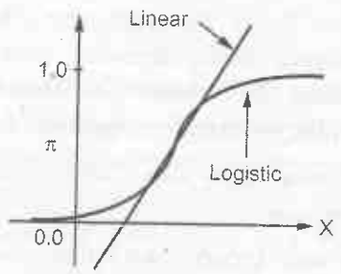


Fig. 2.2.2

- Fig. 2.2.2 shows Sigmoid curve for logistic regression.
- The linear and logistic probability models are :

Linear Regression :

$$p = a_0 + a_1 X_1 + a_2 X_2 + \ldots + a_k X_k$$

Logistic Regression :

$$\ln[p/(1-p)] = b_0 + b_1 X_1 + b_2 X_2 + \ldots + b_k X_k$$

- The linear model assumes that the probability p is a linear function of the regressors, while the logistic model assumes that the natural log of the odds $p/(1-p)$ is a linear function of the regressors.
- The major advantage of the linear model is its interpretability. In the linear model, if a 1 is 0.05, that means that a one-unit increase in X1 is associated with a 5 % point increase in the probability that Y is 1.

- The logistic model is less interpretable. In the logistic model, if b1 is 0.05, that means that a one - unit increase in X1 is associated with a 0.05 increase in the log odds that Y is 1. And what does that mean? I've never met anyone with any intuition for log odds.

2.3 Probabilistic Generative Model

- Generative models are a class of statistical models that generate new data instances. These models are used in unsupervised machine learning to perform tasks such as probability and likelihood estimation, modelling data points, and distinguishing between classes using these probabilities.
- Generative models rely on the Bayes theorem to find the joint probability. Generative models describe how data is generated using probabilistic models. They predict P(y | x), the probability of y given x, calculating the P(x,y), the probability of x and y.

2.3.1 Naive Bayes

- Naive Bayes classifiers are a family of simple probabilistic classifiers based on applying Bayes' theorem with strong independence assumptions between the features. It is highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem.
- A Naive Bayes Classifier is a program which predicts a class value given a set of attributes.
- For each known class value,
  1. Calculate probabilities for each attribute, conditional on the class value.
  2. Use the product rule to obtain a joint conditional probability for the attributes.
  3. Use Bayes rule to derive conditional probabilities for the class variable.
- Once this has been done for all class values, output the class with the highest probability.
- Naive bayes simplifies the calculation of probabilities by assuming that the probability of each attribute belonging to a given class value is independent of all other attributes. This is a strong assumption but results in a fast and effective method.
- The probability of a class value given a value of an attribute is called the conditional probability. By multiplying the conditional probabilities together for each attribute for a given class value, we have a probability of a data instance belonging to that class.

Conditional Probability

- Let A and B be two events such that P(A) > 0. We denote P(B | A) the probability of B given that A has occurred. Since A is known to have occurred, it becomes the new sample space replacing the original S. From this, the definition is ,

$$P(B/A) \equiv \frac{P(A \cap B)}{P(A)}$$

OR

$$P(A \cap B) = P(A)\, P(B/A)$$

- The notation P(B | A) is read **"the probability of event B given event A"**. It is the probability of an event B given the occurrence of the event A.
- We say that, the probability that both A and B occur is equal to the probability that A occurs times the probability that B occurs given that A has occurred. We call **P(B | A) the conditional probability of B given A**, i.e., the probability that B will occur given that A has occurred.
- Similarly, the conditional probability of an event A, given B by,

$$P(A/B) \equiv \frac{P(A \cap B)}{P(B)}$$

- The probability P(A | B) simply reflects the fact that the probability of an event A may depend on a second event B. If A and B are mutually exclusive $A \cap B = \phi$ and P(A | B) = 0.
- Another way to look at the conditional probability formula is :

$$P(\text{Second/First}) = \frac{P(\text{First choice and second choice})}{P(\text{First choice})}$$

- Conditional probability is a defined quantity and cannot be proven.
- The key to solving conditional probability problems is to :
  1. Define the events.
  2. Express the given information and question in probability notation.
  3. Apply the formula.

Joint Probability

- A joint probability is a probability that measures the likelihood that two or more events will happen concurrently.
- If there are two independent events A and B, the probability that A and B will occur is found by multiplying the two probabilities. Thus for two events A and B, the special rule of multiplication shown symbolically is :

P(A and B) = P(A) P(B).

- The general rule of multiplication is used to find the joint probability that two events will occur. Symbolically, the general rule of multiplication is,

$P(A \text{ and } B) = P(A)\, P(B|A)$.

- The probability $P(A \cap B)$ is called the joint probability for two events A and B which intersect in the sample space. Venn diagram will readily shows that

$$P(A \cap B) = P(A) + P(B) - P(A \cup B)$$

Equivalently :

$$P(A \cap B) = P(A) + P(B) - P(A \cap B) \leq P(A) + P(B)$$

- The probability of the union of two events never exceeds the sum of the event probabilities.
- A tree diagram is very useful for portraying conditional and joint probabilities. A tree diagram portrays outcomes that are mutually exclusive.

Bayes Theorem

- Bayes' theorem is a method to revise the probability of an event given additional information. Bayes's theorem calculates a conditional probability called a posterior or revised probability.
- Bayes' theorem is a result in probability theory that relates conditional probabilities. If A and B denote two events, $P(A|B)$ denotes the conditional probability of A occurring, given that B occurs. The two conditional probabilities $P(A|B)$ and $P(B|A)$ are in general different.
- Bayes theorem gives a relation between $P(A|B)$ and $P(B|A)$. An important application of Bayes' theorem is that it gives a rule how to update or revise the strengths of evidence-based beliefs in light of new evidence a posteriori.
- A **prior probability** is an initial probability value originally obtained before any additional information is obtained.
- A **posterior probability** is a probability value that has been revised by using additional information that is later obtained.
- Suppose that $B_1, B_2, B_3 \ldots B_n$ partition the outcomes of an experiment and that A is another event. For any number, k, with $1 \leq k \leq n$, we have the formula :

$$P(B_k/A) = \frac{P(A/B_k) \cdot P(B_k)}{\sum_{i=1}^{n} P(A/B_i) \cdot P(B_i)}$$

2.3.2 Difference between Generative and Discriminative Models

Generative model	Discriminative models
Generative models can generate new data instances.	Discriminative models discriminate between different kinds of data instances
Generative model revolves around the distribution of a dataset to return a probability for a given example.	Discriminative model makes predictions based on conditional probability and is either used for classification or regression.
Generative models capture the joint probability p(X, Y), or just p(X) if there are no labels.	Discriminative models capture the conditional probability p(Y | X).
A generative model includes the distribution of the data itself, and tells you how likely a given example is.	A discriminative model ignores the question of whether a given instance is likely, and just tells you how likely a label is to apply to the instance.
Generative models are used in unsupervised machine learning to perform tasks such as probability and likelihood estimation	The discriminative model is used particularly for supervised machine learning.
Example : Gaussians, Naïve Bayes	Example : Logistic regression, SVMs

2.4 Maximum Margin Classifier : Support Vector Machine

- Support Vector Machines (SVMs) are a set of supervised learning methods which learn from the dataset and used for classification. SVM is a classifier derived from statistical learning theory by Vapnik and Chervonenkis.

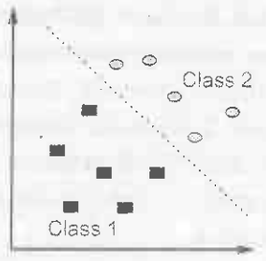


Fig. 2.4.1 Two class problem

- An SVM is a kind of large-margin classifier : It is a vector space based machine learning method where the goal is to find a decision boundary between two classes that is maximally far from any point in the training data
- Given a set of training examples, each marked as belonging to one of two classes, an SVM algorithm

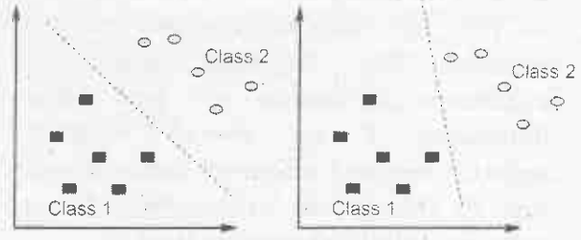


Fig. 2.4.2 Bad decision boundary of SVM

builds a model that predicts whether a new example falls into one class or the other. Simply speaking, we can think of an SVM model as representing the examples as points in space, mapped so that each of the examples of the separate classes are divided by a gap that is as wide as possible.

- New examples are then mapped into the same space and classified to belong to the class based on which side of the gap they fall on.

Two Class Problems

- Many decision boundaries can separate these two classes. Which one should we choose ?
- Perceptron learning rule can be used to find any decision boundary between class 1 and class 2.
- The line that maximizes the minimum margin is a good bet. The model class of "hyper-planes with a margin of m" has a low VC dimension if m is big.
- This **maximum-margin** separator is determined by a subset of the data points. Data points in this subset are called "support vectors". It will be useful **computationally** if only a small fraction of the data points are support vectors, because we use the support vectors to decide which side of the separator a test case is on.

Example of Bad Decision Boundaries

- SVM are primarily two-class classifiers with the distinct characteristic that they aim to find the optimal hyperplane such that the expected generalization error is minimized. Instead of directly minimizing the empirical risk calculated from the training data, SVMs perform structural risk minimization to achieve good generalization.
- The empirical risk is the average loss of an estimator for a finite set of data drawn from P. The idea of risk minimization is not only measure the performance of an estimator by its risk, but to actually search for the estimator that minimizes risk over distribution P. Because we don't know distribution P we instead minimize empirical risk over a training dataset drawn from P. This general learning technique is called **empirical risk minimization.**

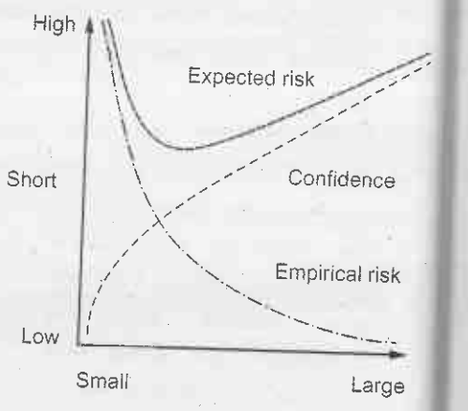


Fig. 2.4.3 Empirical risk

- Fig. 2.4.3 shows empirical risk.

Good Decision Boundary : Margin Should Be Large

- The decision boundary should be as far away from the data of both classes as possible. If data points lie very close to the boundary, the classifier may be consistent but is more "likely" to make errors on new instances from the distribution. Hence, we prefer classifiers that maximize the minimal distance of data points to the separator.

1. **Margin (m) :** the gap between data points & the classifier boundary. The Margin is the minimum distance of any sample to the decision boundary. If this hyperplane is in the canonical form, the margin can be measured by the length of the weight vector. The margin is given by the projection of the distance between these two points on the direction perpendicular to the hyperplane.

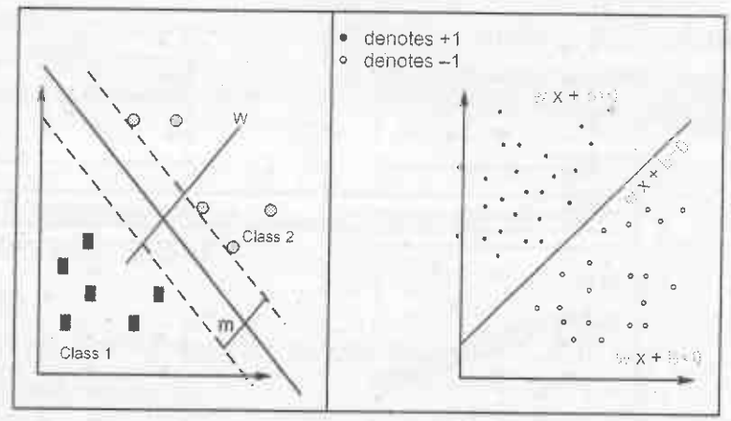


Fig. 2.4.4 Good decision boundary

Margin of the separator is the distance between support vectors.

$$\text{Margin (m)} = \frac{2}{||w||}$$

2. **Maximal margin classifier :** a classifier in the family F that maximizes the margin. Maximizing the margin is good according to intuition and PAC theory. Implies that only support vectors matter; other training examples are ignorable.

Example 2.4.1 *For the following figure find a linear hyperplane (decision boundary) that will separate the data.*

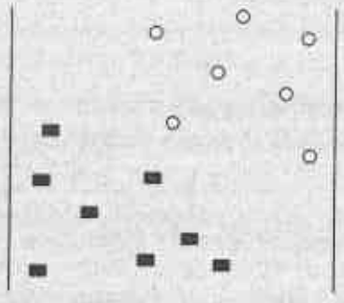


Fig. 2.4.5

Solution :

1. Define what an optimal hyperplane is : **maximize margin**
2. Extend the above definition for non-linearly separable problems : **have a penalty term for misclassifications**

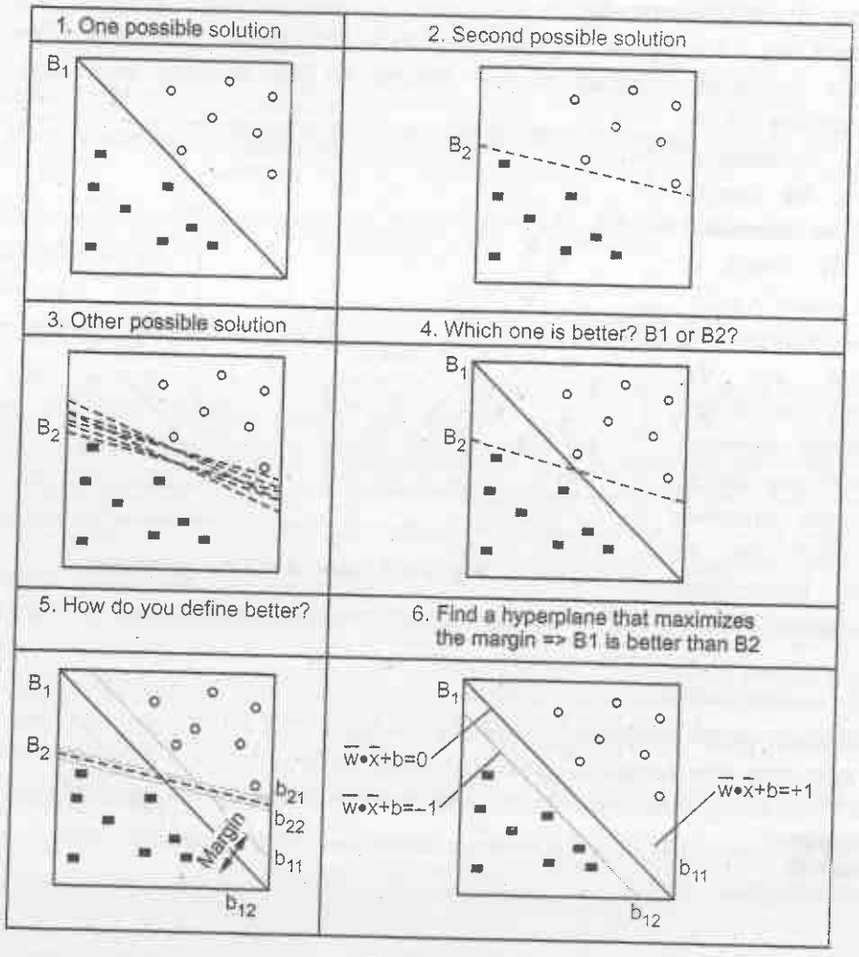


Fig. 2.4.6

3. Map data to high dimensional space where it is easier to classify with linear decision surfaces : **reformulate problem so that data is mapped implicitly to this space**

2.4.1 Key Properties of Support Vector Machines

1. Use a single hyperplane which subdivides the space into two half-spaces, one which is occupied by Class 1 and the other by Class 2

2. They maximize the margin of the decision boundary using quadratic optimization techniques which find the optimal hyperplane.
3. Ability to handle large feature spaces.
4. Overfitting can be controlled by soft margin approach
5. When used in practice, SVM approaches frequently map the examples to a higher dimensional space and find margin maximal hyperplanes in the mapped space, obtaining decision boundaries which are not hyperplanes in the original space.
6. The most popular versions of SVMs use non-linear kernel functions and map the attribute space into a higher dimensional space to facilitate finding "good" linear decision boundaries in the modified space.

2.4.2 SVM Applications

- SVM has been used successfully in many real-world problems,
  1. Text (and hypertext) categorization
  2. Image classification
  3. Bioinformatics (Protein classification, Cancer classification)
  4. Hand-written character recognition
  5. Determination of SPAM email.

2.4.3 Limitations of SVM

1. It is sensitive to noise.
2. The biggest limitation of SVM lies in the choice of the kernel.
3. Another limitation is speed and size.
4. The optimal design for multiclass SVM classifiers is also a research area.

2.4.4 Soft Margin SVM

- For the very high dimensional problems common in text classification, sometimes the data are linearly separable. But in the general case they are not, and even if they are, we might prefer a solution that better separates the bulk of the data while ignoring a few weird noise documents.
- What if the training set is not linearly separable ? *Slack variables* can be added to allow misclassification of difficult or noisy examples, resulting margin called *soft*.
- A *soft-margin* allows a few variables to cross into the margin or over the hyperplane, allowing **misclassification.**
- We penalize the crossover by looking at the number and distance of the **misclassifications.** This is a trade off between the hyperplane violations and the

margin size. The slack variables are bounded by some set cost. The farther they are from the soft margin, the less influence they have on the prediction.

- All observations have an associated slack variable,
  1. Slack variable = 0 then all points on the margin.
  2. Slack variable > 0 then a point in the margin or on the wrong side of the hyperplane
  3. C is the tradeoff between the slack variable penalty and the margin.

2.4.5 Comparison of SVM and Neural Networks

Support Vector Machine	Neural Network
Kernel maps to a very-high dimensional space	Hidden Layers map to lower dimensional spaces
Search space has a unique minimum	Search space has multiple local minima
	Classification extremely efficient
Very good accuracy in typical domains	Very good accuracy in typical domains
Kernel and cost the two parameters to select	Requires number of hidden units and layers
Training is extremely efficient	Training is expensive

Example 2.4.2 *From the following diagram, identify which data points (1, 2, 3, 4, 5) are support vectors (if any), slack variables on correct side of classifier (if any) and slack variables on wrong side of classifier (if any). Mention which point will have maximum penalty and why ?*

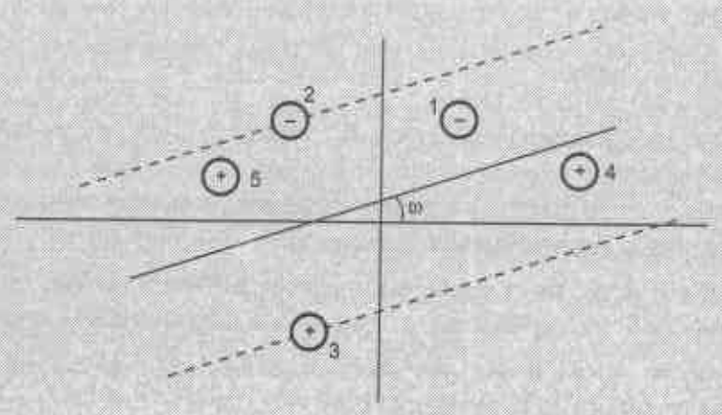


Fig. 2.4.7

Solution :

- Data points 1 and 5 will have maximum penalty.
- Margin (m) is the gap between data points & the classifier boundary. The margin is the minimum distance of any sample to the decision boundary. If this hyperplane is in the canonical form, the margin can be measured by the length of the weight vector.
- Maximal margin classifier : A classifier in the family F that maximizes the margin. Maximizing the margin is good according to intuition and PAC theory. Implies that only support vectors matter; other training examples are ignorable.
- What if the training set is not linearly separable ? Slack variables can be added to allow **misclassification** of difficult or noisy examples, resulting margin called soft.
- A soft-margin allows a few variables to cross into the margin or over the hyperplane, allowing **misclassification.**
- We penalize the crossover by looking at the number and distance of the misclassifications. This is a trade off between the hyperplane violations and the margin size. The slack variables are bounded by some set cost. The farther they are from the soft margin, the less influence they have on the prediction.
- All observations have an associated slack variable

1. Slack variable = 0 then all points on the margin.
2. Slack variable > 0 then a point in the margin or on the wrong side of the hyperplane.
3. C is the tradeoff between the slack variable penalty and the margin.

2.5 Decision Tree

- A decision tree is a simple representation for classifying examples. Decision tree learning is one of the most successful techniques for supervised classification learning.
- In decision analysis, a decision tree can be used to visually and explicitly represent decisions and decision making. As the name goes, it uses a tree-like model of decisions.
- Learned trees can also be represented as sets of if-then rules to improve human readability.
- A decision tree has two kinds of nodes
  1. Each leaf node has a class label, determined by majority vote of training examples reaching that leaf.

2. Each internal node is a question on features. It branches out according to the answers.

- Decision tree learning is a method for approximating discrete-valued target functions. The learned function is represented by a decision tree.
- A learned decision tree can also be re-represented as a set of if-then rules. Decision tree learning is one of the most widely used and practical methods for inductive inference.
- It is robust to noisy data and capable of learning disjunctive expressions.
- Decision tree learning method searches a completely expressive hypothesis

2.5.1 Decision Tree Representation

- Goal : Build a decision tree for classifying examples as positive or negative instances of a concept
- Supervised learning, batch processing of training examples, using a preference bias.
- A decision tree is a tree where
  a. Each non-leaf node has associated with it an attribute (feature).
  b. Each leaf node has associated with it a classification (+ or –).
  c. Each arc has associated with it one of the possible values of the attribute at the node from which the arc is directed.
- Internal node denotes a test on an attribute. Branch represents an outcome of the test. Leaf nodes represent class labels or class distribution.
- A decision tree is a flow-chart-like tree structure, where each node denotes a test on an attribute value, each branch represents an outcome of the test, and tree leaves represent classes or class distributions. Decision trees can easily be converted to classification rules.

Decision Tree Algorithm

- To generate decision tree from the training tuples of data partition D.

Input :

1. Data partition (D) 2. Attribute list 3. Attribute selection method

Algorithm :

1. Create a node (N)
2. If tuples in D are all of the same class then
3. Return node (N) as a leaf node labeled with the class C.

4. If attribute list is empty then return N as a leaf node labeled with the majority class in D
5. Apply attribute selection method(D, attribute list) to find the "best" splitting criterion;
6. Label node N with splitting criterion;
7. If splitting attribute is discrete-valued and multiway splits allowed
8. Then attribute list -> attribute list -> splitting attribute
9. For (each outcome j of splitting criterion)
10. Let D_j be the set of data tuples in D satisfying outcome j;
11. If D_j is empty then attach a leaf labeled with the majority class in D to node N;
12. Else attach the node returned by Generate decision tree(D_j, attribute list) to node N;
13. End of for loop
14. Return N;

- Decision tree generation consists of two phases : Tree construction and pruning
- In tree construction phase, all the training examples are at the root. Partition examples recursively based on selected attributes.
- In tree pruning phase, the identification and removal of branches that reflect noise or outliers.
- There are various paradigms that are used for learning binary classifiers which include :
  1. Decision Trees
  2. Neural Networks
  3. Bayesian Classification
  4. Support Vector Machines

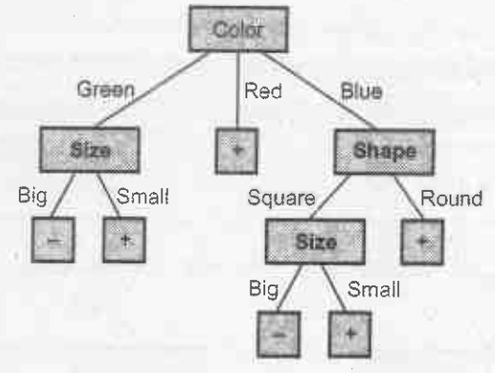


Fig. 2.5.1 Decision tree

Example 2.5.1 *Using following feature tree, write decision rules for majority class.*

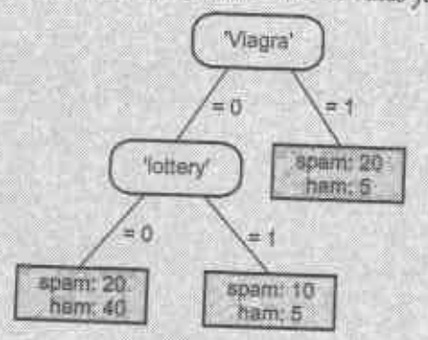


Fig. 2.5.2

Solution : Left Side : A feature tree combining two Boolean features. Each internal node or split is labelled with a feature, and each edge emanating from a split is labelled with a feature value. Each leaf therefore corresponds to a unique combination of feature values. Also indicated in each leaf is the class distribution derived from the training set.

- Right Side : A feature tree partitions the instance space into rectangular regions, one for each leaf.

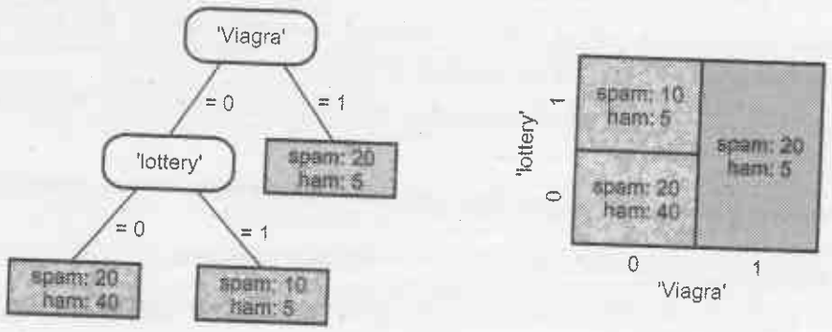


Fig. 2.5.3

- The leaves of the tree in the above figure could be labelled, from left to right, as ham - spam - spam, employing a simple decision rule called majority class.

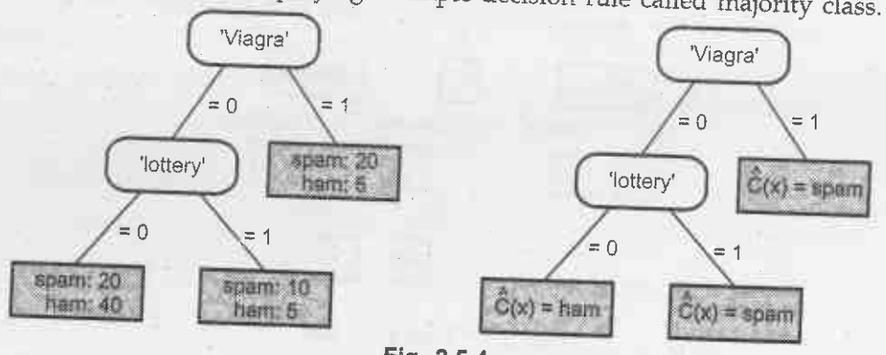


Fig. 2.5.4

- **Left side :** A feature tree with training set class distribution in the leaves.
- **Right side :** A decision tree obtained using the majority class decision rule.

2.5.2 Appropriate Problem for Decision Tree Learning

- Decision tree learning is generally best suited to problems with the following characteristics :
  1. Instances are represented by attribute-value pairs. Fixed set of attributes, and the attributes take a small number of disjoint possible values.
  2. The target function has discrete output values. Decision tree learning is appropriate for a boolean classification, but it easily extends to learning functions with more than two possible output values.
  3. Disjunctive descriptions may be required. Decision trees naturally represent disjunctive expressions.
  4. The training data may contain errors. Decision tree learning methods are robust to errors, both errors in classifications of the training examples and errors in the attribute values that describe these examples.
  5. The training data may contain missing attribute values. Decision tree methods can be used even when some training examples have unknown values.
  6. Decision tree learning has been applied to problems such as learning to classify.

2.5.3 Advantages and Disadvantages of Decision Tree

Advantages :

1. Rules are simple and easy to understand.
2. Decision trees can handle both nominal and numerical attributes.
3. Decision trees are capable of handling datasets that may have errors.
4. Decision trees are capable of handling datasets that may have missing values.
5. Decision trees are considered to be a nonparametric method.
6. Decision trees are self-explantory.

Disadvantages :

1. Most of the algorithms require that the target attribute will have only discrete values.
2. Some problem are difficult to solve like XOR.
3. Decision trees are less appropriate for estimation tasks where the goal is to predict the value of a continuous attribute.

4. Decision trees are prone to errors in classification problems with many class and relatively small number of training examples.

2.6 Random Forests

- Random forest is a famous system learning set of rules that belongs to the supervised getting to know method. It may be used for both classification and regression issues in ML. It is based totally on the concept of ensemble studying, that's a process of combining multiple classifiers to solve a complex problem and to enhance the overall performance of the model.
- As the call indicates, "Random forest is a classifier that incorporates some of choice timber on diverse subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and primarily based on most of the people's votes of predictions, and it predicts the very last output.
- The more wider variety of trees within the forest results in better accuracy and prevents the hassle of overfitting.

2.6.1 How Does Random Forest Algorithm Work ?

- Random forest works in two-section first is to create the random woodland by combining N selection trees and second is to make predictions for each tree created inside the first segment.
- The working technique may be explained within the below steps and diagram :

Step - 1 : Select random K statistics points from the schooling set.

Step - 2 : Build the selection trees associated with the selected information points (Subsets).

Step - 3 : Choose the wide variety N for selection trees which we want to build.

Step - 4 : Repeat step 1 and 2.

Step - 5 : For new factors, locate the predictions of each choice tree and assign the new records factors to the category that wins most people's votes.

- The working of the set of rules may be higher understood by the underneath example :
- Example : Suppose there may be a dataset that includes more than one fruit photo. So, this dataset is given to the random wooded area classifier. The dataset is divided into subsets and given to every decision tree. During the training section, each decision tree produces a prediction end result and while a brand new

statistics point occurs, then primarily based on the majority of consequences, the random forest classifier predicts the final decision. Consider the underneath picture :

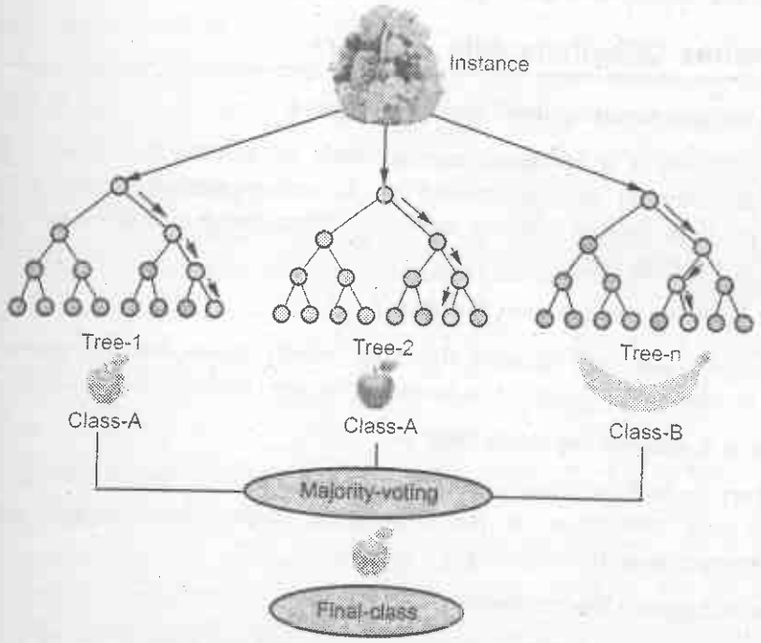


Fig. 2.6.1 Example of random forest

2.6.2 Applications of Random Forest

There are specifically 4 sectors where random forest normally used :

1. **Banking :** Banking zone in general uses this algorithm for the identification of loan danger.
2. **Medicine :** With the assistance of this set of rules, disorder traits and risks of the disorder may be recognized.
3. **Land use :** We can perceive the areas of comparable land use with the aid of this algorithm.
4. **Marketing :** Marketing tendencies can be recognized by the usage of this algorithm.

2.6.3 Advantages of Random Forest

Random forest is able to appearing both classification and regression responsibilities.

- It is capable of managing large datasets with high dimensionality.
- It enhances the accuracy of the version and forestalls the overfitting trouble.

2.6.4 Disadvantages of Random Forest

- Although random forest can be used for both class and regression responsibilities, it isn't extra appropriate for regression obligations.

2.7 Two Marks Questions with Answers

Q.1 What do you mean by least square method ?

Ans. : Least squares is a statistical method used to determine a line of best fit by minimizing the sum of squares created by a mathematical function. A "square" is determined by squaring the distance between a data point and the regression line or mean value of the data set.

Q.2 What is linear Discriminant function ?

Ans. : LDA is a supervised learning algorithm, which means that it requires a labelled training set of data points in order to learn the Linear Discriminant function.

Q.3 What is a support vector in SVM ?

Ans. : Support vectors are data points that are closer to the hyperplane and influence the position and orientation of the hyperplane. Using these support vectors, we maximize the margin of the classifier.

Q.4 What is Support Vector Machines ?

Ans. : A Support Vector Machine (SVM) is a supervised machine learning model that uses classification algorithms for two-group classification problems. After giving an SVM model sets of labeled training data for each category, they're able to categorize new text.

Q.5 Define logistic regression.

Ans. : Logistic regression is supervised learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.

Q.6 List out types of machine learning.

Ans. : Types of machine learning are supervised, semi-supervised, unsupervised and reinforcement learning.

Q.7 What is Random forest ?

Ans. : Random forest is an ensemble learning technique that combines multiple decision trees, implementing the bagging method and results in a robust model with low variance.

Q.8 What are the five popular algorithms of machine learning ?

Ans. : Popular algorithms are Decision Trees, Neural Networks (back propagation), Probabilistic networks, Nearest Neighbor and Support vector machines.

Q.9 What is the function of 'Supervised Learning' ?

Ans. : Functions of 'Supervised Learning' are Classifications, Speech recognition, Regression, Predict time series and Annotate strings.

Q.10 What are the advantages of Naive Bayes ?

Ans. : In Naïve Bayes classifier will converge quicker than discriminative models like logistic regression, so you need less training data. The main advantage is that it can't learn interactions between features.

Q.11 What is regression ?

Ans. : Regression is a method to determine the statistical relationship between a dependent variable and one or more independent variables.

Q.12 Explain linear and non-linear regression model.

Ans. : In linear regression models, the dependence of the response on the regressors is defined by a linear function, which makes their statistical analysis mathematically tractable. On the other hand, in nonlinear regression models, this dependence is defined by a nonlinear function, hence the mathematical difficulty in their analysis.

Q.13 What is regression analysis used for ?

Ans. : Regression analysis is a form of predictive modelling technique which investigates the relationship between a dependent (target) and independent variable (s) (predictor). This technique is used for forecasting, time series modelling and finding the causal effect relationship between the variables.

Q.14 List two properties of logistic regression.

Ans. :

1. The dependent variable in logistic regression follows Bernoulli Distribution.
2. Estimation is done through maximum likelihood.

Q.15 What is the goal of logistic regression ?

Ans. : The goal of logistic regression is to correctly predict the category of outcome for individual cases using the most parsimonious model. To accomplish this goal, a model is created that includes all predictor variables that are useful in predicting the response variable.

Q.16 Define supervised learning.

Ans. : Supervised learning in which the network is trained by providing it with input and matching output patterns. These input-output pairs are usually provided by an external teacher.

❑❑❑

Notes

UNIT III

3 Ensemble Techniques and Unsupervised Learning

Syllabus

Combining multiple learners : Model combination schemes, Voting, Ensemble Learning - bagging, boosting, stacking, Unsupervised learning : K-means, Instance Based Learning : KNN, Gaussian mixture models and Expectation maximization.

Contents

3.1 Combining Multiple Learners

- When designing a learning machine, we generally make some choices like parameters of machine, training data, representation, etc. This implies some sort of variance in performance. For example, in a classification setting, we can use a parametric classifier or in a multilayer perceptron, we should also decide on the number of hidden units.
- Each learning algorithm dictates a certain model that comes with a set of assumptions. This inductive bias leads to error if the assumptions do not hold for the data.
- Different learning algorithms have different accuracies. The no free lunch theorem asserts that no single learning algorithm always achieves the best performance in any domain. They can be combined to attain higher accuracy.
- Data fusion is the process of fusing multiple records representing the same real-world object into a single, consistent, and clean representation. Fusion of data for improving prediction accuracy and reliability is an important problem in machine learning.
- Combining different models is done to improve the performance of deep learning models. Building a new model by combination requires less time, data, and computational resources. The most common method to combine models is by averaging multiple models, where taking a weighted average improves the accuracy.

1. Generating Diverse Learners :

- Different Algorithms : We can use different learning algorithms to train different base-learners. Different algorithms make different assumptions about the data and lead to different classifiers.
- Different Hyper-parameters : We can use the same learning algorithm but use it with different hyper-parameters.
- Different Input Representations : Different representations make different characteristics explicit allowing better identification.
- Different Training Sets : Another possibility is to train different base-learners by different subsets of the training set.

3.1.1 Model Combination Schemes

- Different methods are used for generating final output for multiple base learners are Multiexpert and multistage combination.

1. Multiexpert combination.

- Multiexpert combination methods have base-learners that work in parallel.
  a) Global approach (learner fusion) : given an input, all base-learners generate an output and all these outputs are used, such as voting and stacking
  b) Local approach (learner selection) : in mixture of experts, there is a gating model, which looks at the input and chooses one (or very few) of the learners as responsible for generating the output.
- Multistage combination : Multistage combination methods use a serial approach where the next multistage combination base-learner is trained with or tested on only the instances where the previous base-learners are not accurate enough.
- Let's assume that we want to construct a function that maps inputs to outputs from a set of known N_{train} input-output pairs.
- Let's assume that we want to construct a function that maps inputs to outputs from a set of known N_{train} input-output pairs.

$$D_{train} = \{(x_i, y_i)\}_{i=1}^{N_{train}}$$

where $x_i \in X$ is a D dimensional feature input vector, $y_i \in Y$ is the output.

- Classification : When the output takes values in a discrete set of class labels $Y = \{c_1; c_2; \ldots c_x\}$, where K is the number of different classes. Regression consists in predicting continuous ordered outputs, Y = R.

3.1.2 Voting

- The simplest way to combine multiple classifiers is by voting, which corresponds to taking a linear combination of the learners. Voting is an ensemble machine learning algorithm.
- For regression, a voting ensemble involves making a prediction that is the average of multiple other regression models.
- In classification, a hard voting ensemble involves summing the votes for crisp class labels from other models and predicting the class with the most votes. A soft voting ensemble involves summing the predicted probabilities for class labels and predicting the class label with the largest sum probability.
- Fig. 3.1.1 shows Base-learners with their outputs.

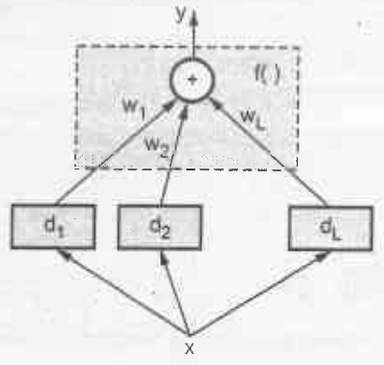


Fig. 3.1.1 Base-learners with their outputs

- In this methods, the first step is to create multiple **classification/regression** models using some training dataset. Each base model can be created using different splits of the same training dataset and same algorithm, or using the same dataset with different algorithms, or any other method.
- Learn multiple alternative definitions of a concept using different training data or different learning algorithms. It combines decisions of multiple definitions, e.g. using weighted voting.
- Fig. 3.1.2 shows general idea of Base-learners with model combiner.

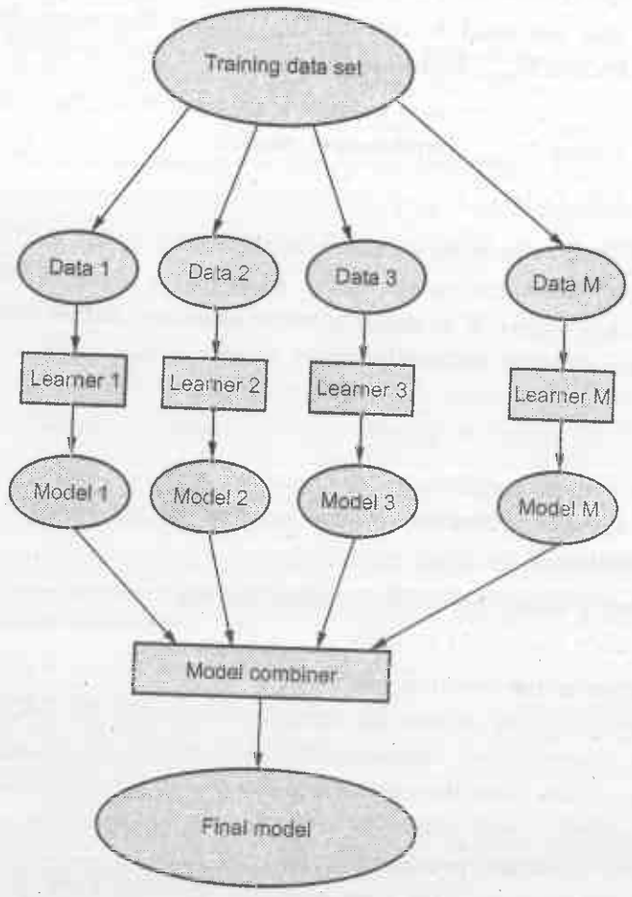


Fig. 3.1.2

- When combining multiple independent and diverse decisions each of which is at least more accurate than random guessing, random errors cancel each other out, and correct decisions are reinforced. Human ensembles are demonstrably better.

- Use a single, arbitrary learning algorithm but manipulate training data to make it learn multiple models.

3.1.3 Error-Correcting Output Codes

- In **Error-Correcting** Output Codes main classification task is defined in terms of a number of subtasks that are implemented by the base-learners. The idea is that the original task of separating one class from all other classes may be a difficult problem.
- So, we want to define a set of simpler classification problems, each specializing in one aspect of the task, and combining these simpler classifiers, we get the final classifier.
- Base-learners are binary classifiers having output – 1/ + 1, and there is a code matrix W of K × L whose K rows are the binary codes of classes in terms of the L base-learners d_j.
- Code matrix W codes classes in terms of learners
- One per class L = K

$$W = \begin{bmatrix} +1 & -1 & -1 & -1 \\ -1 & +1 & -1 & -1 \\ -1 & -1 & +1 & -1 \\ -1 & -1 & -1 & +1 \end{bmatrix}_{K \times L}$$

- The problem here is that if there is an error with one of the base-learners, there may be a **misclassification** because the class code words are so similar. So the approach in error-correcting codes is to have L > K and increase the Hamming distance between the code words.
- One possibility is pairwise separation of classes where there is a separate base-learner to separate C_i from C_j, for i < j.
- Pairwise L = K(K – 1)/2

$$W = \begin{bmatrix} +1 & +1 & +1 & 0 & 0 & 0 \\ -1 & 0 & 0 & +1 & +1 & 0 \\ 0 & -1 & 0 & -1 & 0 & +1 \\ 0 & 0 & -1 & 0 & -1 & -1 \end{bmatrix}$$

- Full code $L = 2^{(K-1)} - 1$

$$W = \begin{bmatrix} -1 & -1 & -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & +1 & +1 & +1 & +1 \\ -1 & +1 & +1 & -1 & -1 & +1 & +1 \\ +1 & -1 & +1 & -1 & +1 & -1 & +1 \end{bmatrix}$$

- With reasonable L, find W such that the Hamming distance between rows and between columns are maximized.
- Voting scheme are

$$y_i = \sum_{j=1}^{L} W_{ij} d_j$$

and then we choose the class with the highest Y_i.

- One problem with ECOC is that because the code matrix W is set a priori, there is no guarantee that the subtasks as defined by the columns of W will be simple.

3.2 Ensemble Learning

- The idea of ensemble learning is to employ multiple learners and combine their predictions. If we have a committee of M models with uncorrelated errors, simply by averaging them the average error of a model can be reduced by a factor of M.
- Unfortunately, the key assumption that the errors due to the individual models are uncorrelated is unrealistic; in practice, the errors are typically highly correlated, so the reduction in overall error is generally small.
- Ensemble modeling is the process of running two or more related but different analytical models and then synthesizing the results into a single score or spread in order to improve the accuracy of predictive analytics and data mining applications.
- Ensembles of classifiers is a set of classifiers whose individual decisions combined in some way to classify new examples.
- Ensemble methods combine several decision trees classifiers to produce better predictive performance than a single decision tree classifier. The main principle behind the ensemble model is that a group of weak learners come together to form a strong learner, thus increasing the accuracy of the model.
- Why do ensemble methods work ?
- Based on one of two basic observations :
  1. Variance reduction : If the training sets are completely independent, it will always helps to average an ensemble because this will reduce variance without affecting bias (e.g., bagging) and reduce sensitivity to individual data points.

  2. Bias reduction : For simple models, average of models has much greater capacity than single model Averaging models can reduce bias substantially by increasing capacity and control variance by Citting one component at a time.

3.2.1 Bagging

- Bagging is also called Bootstrap aggregating. Bagging and boosting are meta - algorithms that pool decisions from multiple classifiers. It creates ensembles by repeatedly randomly resampling the training data.
- Bagging was the first effective method of ensemble learning and is one of the simplest methods of arching. The meta - algorithm, which is a special case of the model averaging, was originally designed for classification and is usually applied to decision tree models, but it can be used with any type of model for classification or regression.
- Ensemble classifiers such as bagging, boosting and model averaging are known to have improved accuracy and robustness over a single model. Although unsupervised models, such as clustering, do not directly generate label prediction for each individual, they provide useful constraints for the joint prediction of a set of related objects.
- For given a training set of size n, create m samples of size n by drawing n examples from the original data, with replacement. Each *bootstrap sample* will on average contain 63.2 % of the unique training examples, the rest are replicates. It combines the m resulting models using simple majority vote.
- In particular, on each round, the base learner is trained on what is often called a "**bootstrap replicate**" of the original training set. Suppose the training set consists of n examples. Then a bootstrap replicate is a new training set that also consists of n examples, and which is formed by repeatedly selecting uniformly at random and with replacement n examples from the original training set. This means that the same example may appear multiple times in the bootstrap replicate, or it may appear not at all.
- It also decreases error by decreasing the variance in the results due to *unstable learners,* algorithms (like decision trees) whose output can change dramatically when the training data is slightly changed.
- **Pseudocode :**
  1. Given training data $(x_1, y_1), \ldots, (x_m, y_m)$
  2. For $t = 1, \ldots, T$:
     a. Form bootstrap replicate dataset S_t by selecting m random examples from the training set with replacement.

b. Let h_t be the result of training base learning algorithm on S_t.

3. Output combined classifier :
$H(x) = \text{majority}(h_1(x), \ldots, h_T(x))$

Bagging Steps :

1. Suppose there are N observations and M features in training data set. A sample from training data set is taken randomly with replacement.
2. A subset of M features is selected randomly and whichever feature gives the best split is used to split the node iteratively.
3. The tree is grown to the largest.
4. Above steps are repeated n times and prediction is given based on the aggregation of predictions from n number of trees.

Advantages of Bagging :

1. Reduces over - fitting of the model.
2. Handles higher dimensionality data very well.
3. Maintains accuracy for missing data.

Disadvantages of Bagging :

1. Since final prediction is based on the mean predictions from subset trees, it won't give precise values for the classification and regression model.

3.2.2 Boosting

- Boosting is a very different method to generate multiple predictions (function estimates) and combine them linearly. Boosting refers to a general and provably effective method of producing a very accurate classifier by combining rough and moderately inaccurate rules of thumb.
- Originally developed by computational learning theorists to guarantee performance improvements on fitting training data for a ***weak learner*** that only needs to generate a hypothesis with a training accuracy greater than 0.5. Final result is the weighted sum of the results of weak classifiers.
- A learner is weak if it produces a classifier that is only slightly better than random guessing, while a learner is said to be strong if it produces a classifier that achieves a low error with high confidence for a given concept.
- Revised to be a practical algorithm, AdaBoost, for building ensembles that empirically improves generalization performance. Examples are given weights. At

each iteration, a new hypothesis is learned and the examples are reweighted to focus the system on examples that the most recently learned classifier got wrong.

- Boosting is a bias reduction technique. It typically improves the performance of a single tree model. A reason for this is that we often cannot construct trees which are sufficiently large due to thinning out of observations in the terminal nodes.
- Boosting is then a device to come up with a more complex solution by taking linear combination of trees. In presence of high - dimensional predictors, boosting is also very useful as a regularization technique for additive or interaction modeling.
- To begin, we define an algorithm for finding the rules of thumb, which we call a weak learner. The boosting algorithm repeatedly calls this weak learner, each time feeding it a different distribution over the training data. Each call generates a weak classifier and we must combine all of these into a single classifier that, hopefully, is much more accurate than any one of the rules.
- Train a set of weak hypotheses : $h_1, \ldots, h_T$. The combined hypothesis H is a weighted majority vote of the T weak hypotheses. During the training, focus on the examples that are misclassified.

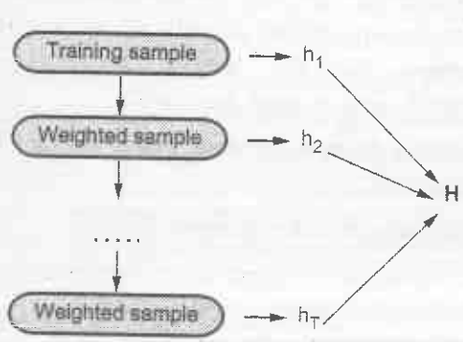


Fig. 3.2.1

AdaBoost :

- AdaBoost, short for "Adaptive Boosting", is a machine learning meta - algorithm formulated by Yoav Freund and Robert Schapire who won the prestigious "Gödel Prize" in 2003 for their work. It can be used in conjunction with many other types of learning algorithms to improve their performance.
- It can be used to learn weak classifiers and final classification based on weighted vote of weak classifiers.
- It is linear classifier with all its desirable properties. It has good generalization properties.

- To use the weak learner to form a highly accurate prediction rule by calling the weak learner repeatedly on different distributions over the training examples.
- Initially, all weights are set equally, but each round the weights of incorrectly classified examples are increased so that those observations that the previously classifier poorly predicts receive greater weight on the next iteration.
- **Advantages of AdaBoost :**
  1. Very simple to implement
  2. Fairly good generalization
  3. The prior error need not be known ahead of time.

Disadvantages of AdaBoost :

1. Suboptimal solution
2. Can over fit in presence of noise.

Boosting Steps :

1. Draw a random subset of training samples d1 without replacement from the training set D to train a weak learner C1
2. Draw second random training subset d2 without replacement from the training set and add 50 percent of the samples that were previously falsely classified/misclassified to train a weak learner C2
3. Find the training samples d3 in the training set D on which C1 and C2 disagree to train a third weak learner C3
4. Combine all the weak learners via majority voting.

Advantages of Boosting :

1. Supports different loss function.
2. Works well with interactions.

Disadvantages of Boosting :

1. Prone to over-fitting.
2. Requires careful tuning of different hyper - parameters.

3.2.3 Stacking

- Stacking, sometimes called stacked generalization, is an ensemble machine learning method that combines multiple heterogeneous base or component models via a meta-model.

- The base model is trained on the complete training data, and then the meta-model is trained on the predictions of the base models. The advantage of stacking is the ability to explore the solution space with different models in the same problem.
- The stacking based model can be visualized in levels and has at least two levels of the models. The first level typically trains the two or more base learners(can be heterogeneous) and the second level might be a single meta learner that utilizes the base models predictions as input and gives the final result as output. A stacked model can have more than two such levels but increasing the levels doesn't always guarantee better performance.
- In the classification tasks, often logistic regression is used as a meta learner, while linear regression is more suitable as a meta learner for regression-based tasks.
- **Stacking** is concerned with combining multiple classifiers generated by different learning algorithms $L_1,\ldots,L_N$ on a single dataset S, which is composed by a feature vector $S_i = (x_i, t_i)$
- The stacking process can be broken into two phases :
  1. Generate a set of base - level classifiers $C_1,\ldots,C_N$ where $C_i = L_i(S)$
  2. Train a meta - level classifier to combine the outputs of the base - level classifiers.
- Fig. 3.2.2 shows stacking frame.

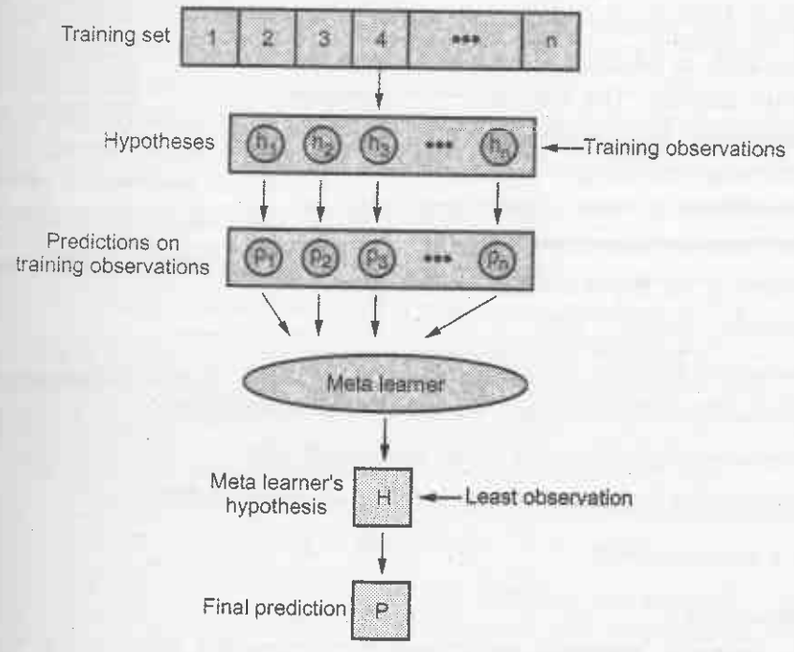


Fig. 3.2.2 Stacking frame

- The training set for the meta - level classifier is generated through a leave - one - out cross validation process.

$$\forall_i = 1, \ldots, n \text{ and } \forall_k = 1, \ldots, N : C_k^i$$
$$= L_k(S - s_i)$$

- The learned classifiers are then used to generate predictions for $s_i : \hat{y}_i^k = C_k^i(x_i)$
- The meta - level dataset consists of examples of the form $((\hat{y}_k^1, \ldots, \hat{y}_i^n), y_i)$ where the features are the predictions of the base - level classifiers and the class is the correct class of the example in hand.
- Why do ensemble methods work ?
- Based on one of two basic observations :
  1. **Variance reduction :** If the training sets are completely independent, it will always helps to average an ensemble because this will reduce variance without affecting bias (e.g. - bagging) and reduce sensitivity to individual data points.
  2. **Bias reduction :** For simple models, average of models has much greater capacity than single model Averaging models can reduce bias substantially by increasing capacity and control variance by Citting one component at a time.

3.2.4 Adaboost

- AdaBoost also referred to as adaptive boosting is a method in Machine Learning used as an ensemble method. The maximum not unusual algorithm used with AdaBoost is selection trees with one stage meaning with decision trees with most effective 1 split. These trees also are referred to as decision stumps.

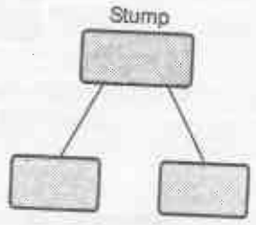


Fig. 3.2.3 Adaboost

- The working of the AdaBoost version follows the beneath-referred to path :
  - Creation of the base learner.
  - Calculation of the total error via the beneath formulation.
  - Calculation of performance of the decision stumps.
  - Updating the weights in line with the misclassified factors.

Creation of a new database :

AdaBoost ensemble :

- In the ensemble approach, we upload the susceptible fashions sequentially and then teach them the use of weighted schooling records.

- We hold to iterate the process till we gain the advent of a pre-set range of vulnerable learners or we can not look at further improvement at the dataset. At the end of the algorithm, we are left with some vulnerable learners with a stage fee.

3.2.5 Difference between Bagging and Boosting

Sr. No.	Bagging	Boosting
1.	Bagging is a technique that builds multiple homogeneous models from different subsamples of the same training dataset to obtain more accurate predictions than its individual models	Boosting refers to a group of algorithms that utilize weighted averages to make weak learning algorithms stronger learning algorithms.
2.	Learns them independently from each other in parallel	Learns them sequentially in a very adaptative way
3.	It helps in reducing variance.	It helps in reducing bias and variance.
4.	Every model receives an equal weight.	Models are weighted by their performance.

3.3 Clustering

- Given a set of objects, place them in groups such that the objects in a group are similar (or related) to one another and different from (or unrelated to) the objects in other groups.
- Cluster analysis can be a powerful data-mining tool for any organization that needs to identity discrete groups of customers, sales transactions, or other types of behaviors and things. For example, insurance providers use cluster analysis to detect fraudulent claims and banks used it for credit scoring.
- Cluster analysis uses mathematical models to discover groups of similar customers based on the smallest variations among customers within each group.
- Cluster is a group of objects that belong to the same class. In another words the similar object are grouped in one cluster and dissimilar are grouped in other cluster.
- Clustering is a process of partitioning a set of data in a set of meaningful subclasses. Every data in the sub class shares a common trait. It helps a user understand the natural grouping or structure in a data set.
- Various types of clustering methods are partitioning methods, hierarchical clustering, fuzzy clustering, density based clustering and model based clustering.
- Cluster anlysis is process of grouping a set of data objects into clusters.

- **Desirable properties of a clustering algorithm are as follows :**

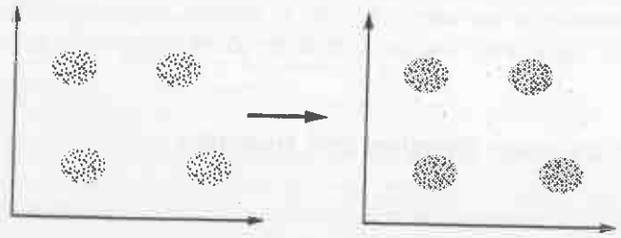


Fig. 3.3.1

1. Scalability (in terms of both time and space)
2. Ability to deal with different data types
3. Minimal requirements for domain knowledge to determine input parameters.
4. Interpretability and usability.

- Clustering of data is a method by which large sets of data are grouped into clusters of smaller sets of similar data. Clustering can be considered the most important unspervised learning problem.
- A cluster is therefore a collection of objects which are "similar" between them and are dissimilar" to the objects belonging to other clusters. Fig. 3.3.1 shows cluster.
- In this case we easily identify the 4 clusters into which the data can be divided; the similarity criterion is distance : two or more objects belong to the same cluster if they are "close" according to a given distance (in this case geometrical distance). This is called **distance-based clustering.**

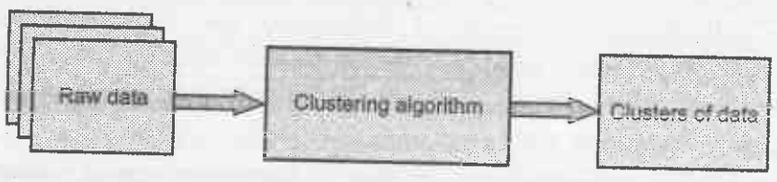


Fig. 3.3.2

- Clustering means grouping of data or dividing a large data set into smaller data sets of some similarity.
- A clustering algorithm attempts to find natural groups components or data based on some similarity. Also, the clustering algorithm finds the centroid of a group of data sets.
- To determine cluster membership, most algorithms evaluate the distance between a point and the cluster centroids. The output from a clustering algorithm is

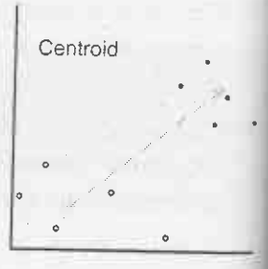


Fig. 3.3.3

basically a statistical description of the cluster centroids with the number of components in each cluster.

- **Cluster centroid** : The centroid of a cluster is a point whose parameter values are the mean of the parameter values of all the points in the cluster. Each cluster has a well defined centroid.
- **Distance :** The distance between two points is taken as a common metric to as see the similarity among the components of population. The commonly used distance measure is the euclidean metric which defines the distance between two points $p = (p_1, p_2, \ldots)$ and $q = (q_1, q_2, \ldots)$ is given by,

$$d = \sum_{i=1}^{k} (p_i - q_i)^2$$

- The goal of clustering is to determine the intrinsic grouping in a set of unlabeled data. But how to decide what constitutes a good clustering ? It can be shown that there is no absolute "best" criterion which would be independent of the final aim of the clustering. Consequently, it is the user which must supply criterion, in such a way that the result of the clustering will suit their needs.
- Clustering analysis helps construct meaninful partitioning of a large set of objects Cluster analysis has been widely used in numerous applications, including pattern recognition, data analysis, image processing etc.
- Clustering algorithms may be classified as listed below :
  1. Exclusive clustering
  2. Overlapping clustering
  3. Hierarchical clustering
  4. Probabilisitic clustering.
- A good clustering method will produce high quality clusters high intra - class similarlity and low inter - class similarity. The quality of a clustering result depends on both the similarity measure used by the method and its implementation. The quality of a clustering method is also measured by it's ability to discover some or all of the hidden patterns.
- Clustering techniques types : The major clustering techniques are,
  a) Partitioning methods
  b) Hierarchical methods
  c) Density - based methods.

3.3.1 Unsupervised Learning : K-means

- K-Means clustering is heuristic method. Here each cluster is represented by the center of the cluster. "K" stands for number of clusters, it is typically a user input to the algorithm; some criteria can be used to automatically estimate K.
- This method initially takes the number of components of the population equal to the final required number of clusters. In this step itself the final required number of clusters is chosen such that the points are mutually farthest apart.
- Next, it examines each component in the population and assigns it to one of the clusters depending on the minimum distance. The centroid's position is recalculated everytime a component is added to the cluster and this continues until all the components are grouped into the final required number of clusters.
- Given K, the K-means algorithm consists of four steps :
  1. Select initial centroids at random.
  2. Assign each object to the cluster with the nearest centroid.
  3. Compute each centroid as the mean of the objects assigned to it.
  4. Repeat previous 2 steps until no change.
- The $x_1, \ldots, x_N$ are data points or vectors of observations. Each observation (vector x_i) will be assigned to one and only one cluster. The C(i) denotes cluster number for the i^{th} observation. K-means minimizes within-cluster point scatter :

$$W(C) = \frac{1}{2} \sum_{K=1}^{K} \sum_{C(i)=K} \sum_{C(j)=K} \| x_i - x_j \|^2$$

$$= \sum_{K=1}^{K} N_k \sum_{C(i)=K}^{K} \| x_i - m_K \|^2$$

where

m_K is the mean vector of the K^{th} cluster.

N_K is the number of observations in K^{th} cluster.

K-Means Algorithm Properties

1. There are always K clusters.
2. There is always at least one item in each cluster.
3. The clusters are non-hierarchical and they do not overlap.
4. Every member of a cluster is closer to its cluster than any other cluster because closeness does not always involve the 'center' of clusters.

The K-Means Algorithm Process

1. The dataset is partitioned into K clusters and the data points are randomly assigned to the clusters resulting in clusters that have roughly the same number of data points.
2. For each data point.
   a. Calculate the distance from the data point to each cluster.
   b. If the data point is closest to its own cluster, leave it where it is.
   c. If the data point is not closest to its own cluster, move it into the closest cluster.
3. Repeat the above step until a complete pass through all the data points results in no data point moving from one cluster to another. At this point the clusters are stable and the clustering process ends.
4. The choice of initial partition can greatly affect the final clusters that result, in terms of inter - cluster and intracluster distances and cohesion.

- K-means algorithm is iterative in nature. It converges, however only a local minimum is obtained. It works only for numerical data. This method easy to implement.
- **Advantages of K-Means Algorithm :**
  1. Efficient in computation
  2. Easy to implement.
- **Weaknesses**
  1. Applicable only when *mean* is defined.
  2. Need to specify K, the *number* of clusters, in advance.
  3. Trouble with noisy data and *outliers*.
  4. Not suitable to discover clusters with *non-convex shapes*.

3.4 Instance Based Learning : kNN

- k-Nearest Neighbour is one of the only Machine Learning algorithms based totally on supervised learning approach.
- k-NN algorithm assumes the similarity between the brand new case/facts and available instances and placed the brand new case into the category that is maximum similar to the to be had classes.
- k-NN set of rules shops all of the to be had facts and classifies a new statistics point based at the similarity. This means when new data seems then it may be effortlessly categorised into a properly suite class by using k-NN algorithm.

- k-NN set of rules can be used for regression as well as for classification however normally it's miles used for the classification troubles.
- k-NN is a non-parametric algorithm, because of this it does no longer makes any assumption on underlying data.
- It is also referred to as a lazy learner set of rules because it does no longer research from the training set immediately as a substitute it shops the dataset and at the time of class, it plays an movement at the dataset.
- The kNN set of rules at the schooling section simply stores the dataset and when it gets new data, then it classifies that statistics into a class that is an awful lot similar to the brand new data.
- Example : Suppose, we've an picture of a creature that looks much like cat and dog, but we want to know both it is a cat or dog. So for this identity, we are able to use the kNN algorithm, because it works on a similarity degree. Our kNN version will discover the similar features of the new facts set to the cats and dogs snap shots and primarily based on the most similar functions it will place it in both cat or canine class.

3.4.1 Why Do We Need kNN ?

- Suppose there are two categories, i.e., category A and category B and we've a brand new statistics point x1, so this fact point will lie within of these classes. To solve this sort of problem, we need a k-NN set of rules. With the help of k-NN, we will without difficulty discover the category or class of a selected dataset. Consider the underneath diagram :

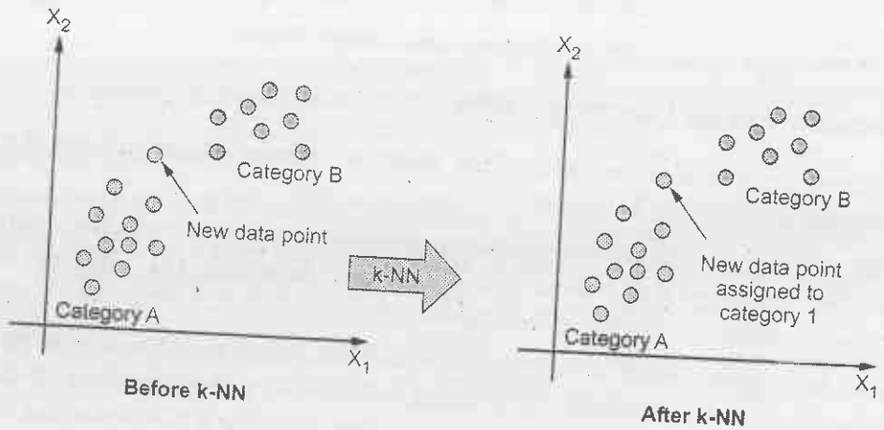


Fig. 3.4.1 Why do we need kNN ?

3.4.2 How Does kNN Work ?

- The k-NN working can be explained on the basis of the below algorithm :

Step - 1 : Select the wide variety k of the acquaintances.

Step - 2 : Calculate the Euclidean distance of k variety of friends.

Step - 3 : Take the k nearest neighbors as according to the calculated Euclidean distance.

Step - 4 : Among these ok pals, count number the number of the data points in each class.

Step - 5 : Assign the brand new records points to that category for which the quantity of the neighbor is maximum.

Step - 6 : Our model is ready.

- Suppose we've got a brand new information point and we want to place it in the required category. Consider the under image

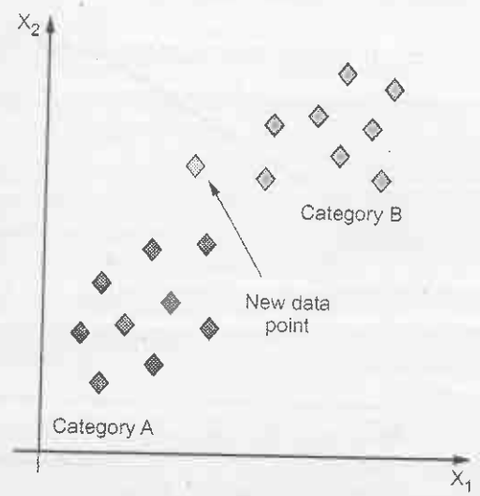
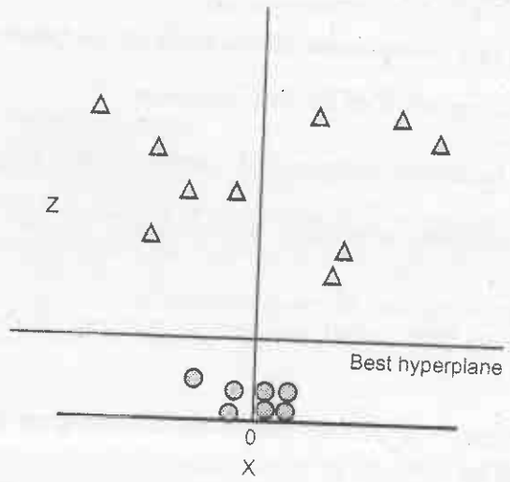


Fig. 3.4.2 kNN example

- Firstly, we are able to pick the number of friends, so we are able to select the ok = 5.
- Next, we will calculate the Euclidean distance between the facts points. The Euclidean distance is the gap between points, which we've got already studied in geometry. It may be calculated as :



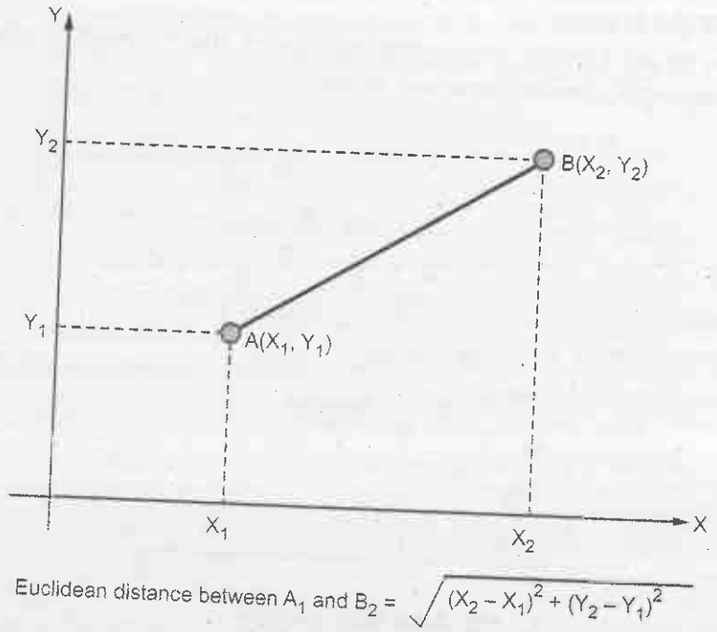


Fig. 3.4.3 kNN example continue

- By calculating the Euclidean distance we got the nearest acquaintances, as 3 nearest neighbours in category A and two nearest associates in class B. Consider the underneath image.

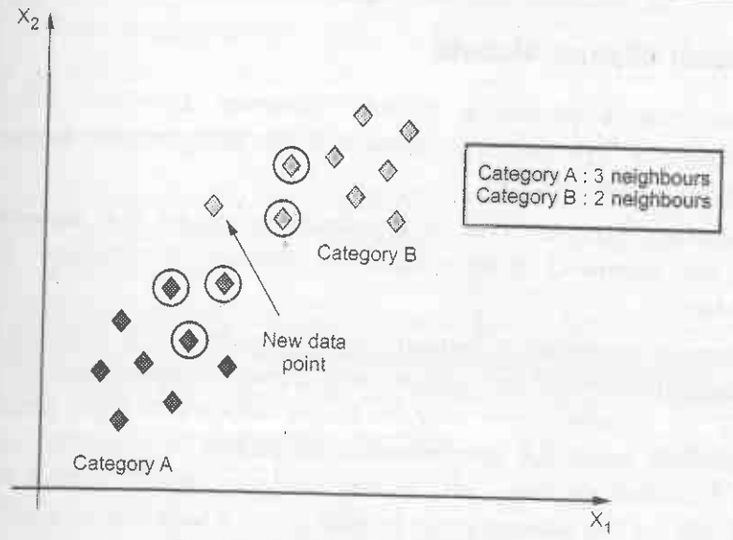


Fig. 3.4.4 kNN example continue

- As we are able to see the three nearest acquaintances are from category A, subsequently this new fact point must belong to category A.

3.4.3 Difference between K-means and kNN

Sr. No.	K-means	kNN
1.	K-Means is an unsupervised machine learning algorithm used for clustering.	kNN is a supervised machine learning algorithm used for classification.
2.	K-Means is an eager learner	k-NN is a lazy learner
3.	It is used for Clustering	It is used mostly for Classification, and sometimes even for Regression
4.	K' in K-Means is the number of clusters the algorithm is trying to identify/learn from the data	K' in KNN is the number of nearest neighbours used to classify or predict a test sample
5.	K-means require unlabelled data. It gathers and groups data into k number of clusters.	kNN require labelled data and will give new data points accordingly to the k number or the closest data points.

3.5 Gaussian Mixture Models

- Gaussian mixture models is a "soft" clustering algorithm, where each point probabilistically "belongs" to all clusters. This is different than k-means where each point belongs to one cluster.
- The Gaussian mixture model is a probabilistic model that assumes all the data points are generated from a mix of Gaussian distributions with unknown parameters.
- For example, in modeling human height data, height is typically modeled as a normal distribution for each gender with a mean of approximately 5'10" for males and 5'5" for females. Given only the height data and not the gender assignments for each data point, the distribution of all heights would follow the sum of two scaled (different variance) and shifted (different mean) normal distributions. A model making this assumption is an example of a Gaussian mixture model.
- Gaussian mixture models do not rigidly classify each and every instance into one class or the other. The algorithm attempts to produce K-Gaussian distributions that would take into account the entire training space. Every point can be associated with one or more distributions. Consequently, the deterministic factor would be the probability that each point belongs to a certain Gaussian distribution.
- Gaussian mixture model consists of two parts : Mean vectors and covariance matrices.
- A Gaussian distribution is defined as a continuous probability distribution that takes on a bell-shaped curve. Another name for Gaussian distribution is the normal distribution.
- In a one dimensional space, the probability density function of a Gaussian distribution is given by :

$$f(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

where μ is the mean and σ^2 is the variance

- Gaussian mixture models can be used for a variety of use cases, including identifying customer segments, detecting fraudulent activity and clustering images.
- GMMs have a variety of real-world applications. Some of them are listed below,
  a) Used for signal processing
  b) Used for customer churn analysis
  c) Used for language identification

  d) Used in video game industry
  e) Genre classification of songs.

3.5.1 Expectation - Maximization

- In Gaussian mixture models, an expectation - maximization method is a powerful tool for estimating the parameters of a Gaussian mixture model. The expectation is termed E and maximization is termed M.
- Expectation is used to find the Gaussian parameters which are used to represent each component of Gaussian mixture models. Maximization is termed M and it is involved in determining whether new data points can be added or not.
- The Expectation-Maximization (EM) algorithm is used in maximum likelihood estimation where the problem involves two sets of random variables of which one, X, is observable and the other Z, is hidden.
- The goal of the algorithm is to find the parameter vector ϕ that maximizes the likelihood of the observed values of X, $L(\phi|X)$.
- But in cases where this is not feasible, we associate the extra hidden variables Z and express the underlying model using both, to maximize the likelihood of the joint distribution of X and Z, the complete likelihood $L_c(\phi|X,Z)$.

3.5.2 EM Algorithm

- Expectation-Maximization (EM) is an iterative method used to find maximum likelihood estimates of parameters in probabilistic models, where the model depends on unobserved, also called latent, variables.
- EM alternates between performing an expectation (E) step, which computes an expectation of the likelihood by including the latent variables as if they were observed, and maximization (M) step, which computes the maximum likelihood estimates of the parameters by maximizing the expected likelihood found in the E step.
- The parameters found on the M step are then used to start another E step, and the process is repeated until some criterion is satisfied. EM is frequently used for data clustering like for example in Gaussian mixtures.
- In the **Expectation step**, find the expected values of the latent variables (here you need to use the current parameter values).
- In the **Maximization step**, first plug in the expected values of the latent variables in the log-likelihood of the augmented data. Then maximize this log-likelihood to reevaluate the parameters.

- Expectation-Maximization (EM) is a technique used in point estimation. Given a set of observable variables X and unknown (latent) variables Z we want to estimate parameters θ in a model.
- The Expectation Maximization (EM) algorithm is a widely used maximum likeli-hood estimation procedure for statistical models when the values of some of the variables in the model are not observed
- The EM algorithm is an elegant and powerful method for finding the maximum likelihood of models with hidden variables. The key concept in the EM algorithm is that it iterates between the expectation step (E-step) and maximization step (M-step) until convergence.
- In the E-step, the algorithm estimates the posterior distribution of the hidden variables Q given the observed data and the current parameter settings; and in the M-step the algorithm calculates the ML parameter settings with Q fixed.
- At the end of each iteration the lower bound on the likelihood is optimized for the given parameter setting (M-step) and the likelihood is set to that bound (E-step), which guarantees an increase in the likelihood and convergence to a local maximum, or global maximum if the likelihood function is unimodal.
- Generally, EM works best when the fraction of missing information is small and the dimensionality of the data is not too large. EM can require many iterations, and higher dimensionality can dramatically slow down the E-step.
- EM is useful for several reasons: conceptual simplicity, ease of implementation and the fact that each iteration improves $l(\theta)$. The rate of convergence on the first few steps is typically quite good, but can become excruciatingly slow as you approach local optima.
- Sometimes the M-step is a constrained maximization, which means that there are constraints on valid solutions not encoded in the function itself.
- Expectation maximization is an effective technique that is often used in data analysis to manage missing data. Indeed, expectation maximization overcomes some of the limitations of other techniques, such as mean substitution or regression substitution. These alternative techniques generate biased estimates-and, specifically, underestimate the standard errors. Expectation maximization overcomes this problem.

3.6 Two Marks Questions with Answers

Q.1 What is unsupervised learning ?

Ans. : In an unsupervised learning, the network adapts purely in response to its inputs. Such networks can learn to pick out structure in their input.

Q.2 What is semi-supervised learning ?

Ans. : Semi-supervised learning uses both labeled and unlabeled data to improve supervised learning.

Q.3 What is ensemble method ?

Ans. : Ensemble methods is a machine learning technique that combines several base models in order to produce one optimal predictive model. It combine the insights obtained from multiple learning models to facilitate accurate and improved decisions.

Q.4 What is cluster ?

Ans. : Cluster is a group of objects that belong to the same class. In other words the similar object are grouped in one cluster and dissimilar are grouped in other cluster.

Q.5 Explain clustering.

Ans. : Clustering is a process of partitioning a set of data in a set of meaningful subclasses. Every data in the subclass shares a common trait. It helps a user understand the natural grouping or structure in a data set.

Q.6 What is bagging ?

Ans. : Bagging is also known as Bootstrap aggregation, ensemble method works by training multiple models independently and combining later to result in a strong model.

Q.7 Define boosting.

Ans. : Boosting refers to a group of algorithms that utilize weighted averages to make weak learning algorithms stronger learning algorithms

Q.8 What is k-Nearest Neighbour Methods ?

Ans. :
- The k-Nearest Neighbor (kNN) is a classical classification method and requires no training effort, critically depends on the quality of the distance measures among examples.
- The kNN classifier uses mahalanobis distance function. A sample is classified according to the majority vote of the its nearest k training samples in the feature space. Distance of a sample to its neighbors is defined using a distance function.

Q.9 Which are the performance factors that influence kNN algorithm ?

Ans. : The performance of the kNN algorithm is influenced by three main factors :

1. The distance function or distance metric used to determine the nearest neighbors.
2. The decision rule used to derive a classification from the k-nearest neighbors.
3. The number of neighbors used to classify the new example.

Q.10 What is K-means clustering ?

Ans. : k-means clustering is heuristic method. Here each cluster is represented by the center of the cluster. The k-means algorithm takes the input parameter, k, and partitions a set of n objects into k-clusters so that the resulting intracluster similarity is high but the intercluster similarity is low.

Q.11 List the properties of K-Means algorithm.

Ans. : 1. There are always k clusters.

2. There is always at least one item in each cluster.

3. The clusters are non-hierarchical and they do not overlap.

Q.12 What is stacking ?

Ans. : Stacking, sometimes called stacked generalization, is an ensemble machine learning method that combines multiple heterogeneous base or component models via a meta-model.

Q.13 How do GMMs differentiate from K-means clustering ?

Ans. : GMMs and K-means, both are clustering algorithms used for unsupervised learning tasks. However, the basic difference between the is that k-means is a distance-based clustering method while GMMs is a distribution based clustering method.

❑❑❑

UNIT IV

4 Neural Networks

Syllabus

Multilayer perceptron, activation functions, network training - gradient descent optimization - stochastic gradient descent, error backpropagation, from shallow networks to deep networks -Unit saturation (aka the vanishing gradient problem) - ReLU, hyperparameter tuning, batch normalization, regularization, dropout.

Contents

4.1 Perceptron

- The perceptron is a feed-forward network with one output neuron that learns a separating hyper-plane in a pattern space.
- The "n" linear Fx neurons feed forward to one threshold output Fy neuron. The perceptron separates linearly separable set of patterns.

4.1.1 Single Layer Perceptron

- The perceptron is a feed-forward network with one output neuron that learns a separating hyper-plane in a pattern space. The "n" linear Fx neurons feed forward to one threshold output Fy neuron. The perceptron separates linearly separable set of patterns.
- SLP is the simplest type of artificial neural networks and can only classify linearly separable cases with a binary target (1, 0).
- We can connect any number of McCulloch-Pitts neurons together in any way we like. An arrangement of one input layer of McCulloch-Pitts neurons feeding forward to one output layer of McCulloch-Pitts neurons is known as a **Perceptron**.
- A single layer feed-forward network consists of one or more output neurons, each of which is connected with a weighting factor W_{ij} to all of the inputs X_i.
- The Perceptron is a kind of a single-layer artificial network with only one neuron. The Percepton is a network in which the neuron unit calculates the linear combination of its real-valued or boolean inputs and passes it through a threshold activation function. Fig. 4.1.1 shows Perceptron.

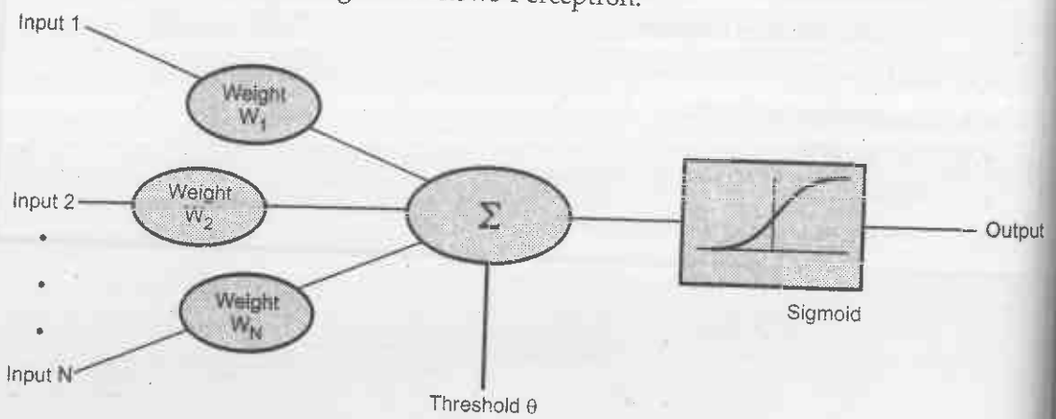


Fig. 4.1.1 Perceptron

- The Perceptron is sometimes referred to a Threshold Logic Unit (TLU) since it discriminates the data depending on whether the sum is greater than the threshold value.

- In the simplest case the network has only two inputs and a single output. The output of the neuron is :

$$y = f\left(\sum_{i=1}^{2} W_i X_i + b\right)$$

- Suppose that the activation function is a threshold then

$$f = \begin{cases} 1 & \text{if } s > 0 \\ -1 & \text{if } s \leq 0 \end{cases}$$

- The Perceptron can represent most of the primitive boolean functions : AND, OR, NAND and NOR but can not represent XOR.
- In single layer perceptron, initial weight values are assigned randomly because it does not have previous knowledge. It sum all the weighted inputs. If the sum is greater than the threshold value then it is activated i.e. output = 1.

Output

$W_1X_1 + W_2X_2 + ... + W_nX_n > \theta \Rightarrow 1$

$W_1X_1 + W_2X_2 + ... + W_nX_n \leq \theta \Rightarrow 0$

- The input values are presented to the perceptron, and if the predicted output is the same as the desired output, then the performance is considered satisfactory and no changes to the weights are made.
- If the output does not match the desired output, then the weights need to be changed to reduce the error.
- The weight adjustment is done as follows :

$$\Delta W = \eta \times d \times x$$

Where x = Input data

d = Predicted output and desired output.

η = Learning rate

- If the output of the perceptron is correct then we do not take any action. If the output is incorrect then the weight vector is $W \rightarrow W + \Delta W$.
- The process of weight adaptation is called **learning**.
- Perceptron Learning Algorithm :

1. Select random sample from training set as input.
2. If classification is correct, do nothing.
3. If classification is incorrect, modify the weight vector W using

$$W_i = W_i + \eta d(n) X_i(n)$$

Repeat this procedure until the entire training set is classified correctly.

4.1.2 Multilayer Perceptron

- A Multi-Layer Perceptron (MLP) has the same structure of a single layer perceptron with one or more hidden layers. An MLP is a network of simple neurons called perceptrons.
- A typical multilayer perceptron network consists of a set of source nodes forming the input layer, one or more hidden layers of computation nodes and an output layer of nodes.
- It is not possible to find weights which enable single layer perceptrons to deal with non-linearly separable problems like XOR : See Fig. 4.1.2.

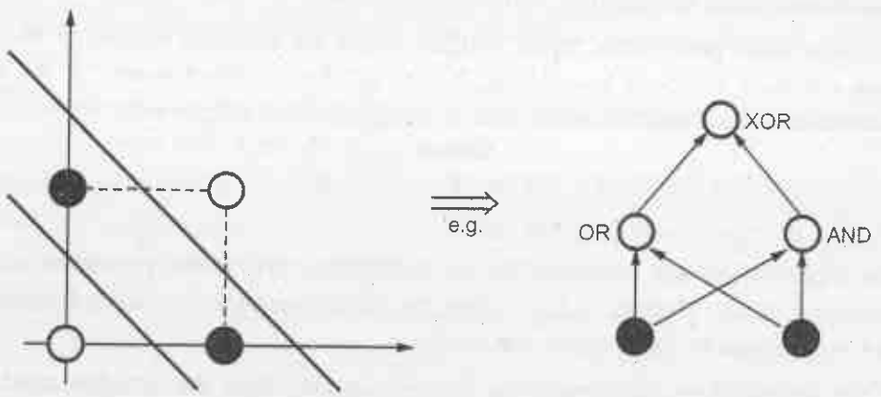


Fig. 4.1.2

4.1.3 Limitation of Learning in Perceptron : Linear Separability

- Consider two-input patterns (X_1, X_2) being classified into two classes as shown in Fig. 4.1.3. Each point with either symbol of x or 0 represents a pattern with a set of values (X_1, X_2).
- Each pattern is classified into one of two classes. Notice that these classes can be separated with a single line L. They are known as linearly separable patterns.

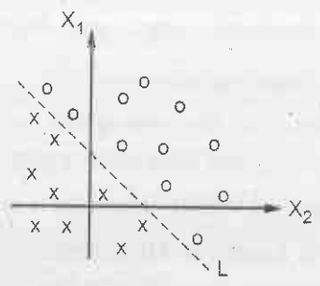


Fig. 4.1.3 Two class

- Linear separability refers to the fact that classes of patterns with η-dimensional vector $x = (x_1, x_2, \ldots x_n)$ can be separated with a single decision surface. In the case above, the line L represents the decision surface.
- If two classes of patterns can be separated by a decision boundary, represented by the linear equation then they are said to be linearly separable. The simple network can correctly classify any patterns.
- Decision boundary (i.e., **W, b** or **q**) of linearly separable classes can be determined either by some learning procedures or by solving linear equation systems based on representative patterns of each classes.
- If such a decision boundary does not exist, then the two classes are said to be linearly inseparable.
- Linearly inseparable problems cannot be solved by the simple network, more sophisticated architecture is needed.
- Examples of linearly separable classes

1. Logical **AND** function

Patterns (bipolar)

x_1	x_2	y
-1	-1	-1
-1	1	1
1	-1	1
1	1	1

Decision boundary

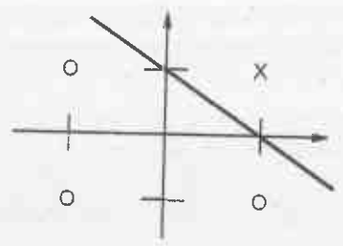
$w_1 = 1$

$w_2 = 1$

$b = -1$

$q = 0$

$-1 + x_1 + x_2 = 0$

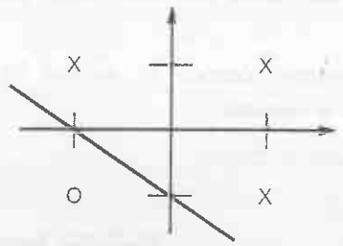


X : Class I (y = 1)
O : Class II (y = –1)

Fig. 4.1.4

2. Logical **OR** function

Patterns (bipolar)			Decision boundary
x_1	x_2	y	$w_1 = 1$
– 1	– 1	– 1	$w_2 = 1$
– 1	1	1	b = 1
1	– 1	1	q = 0
1	1	1	$1 + x_1 + x_2 = 0$



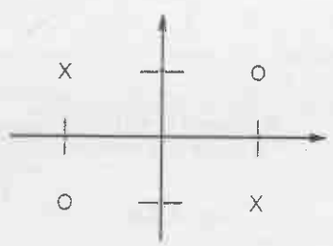
X : Class I (y = 1)
O : Class II (y = –1)

Fig. 4.1.5

- Examples of linearly inseparable classes

1. Logical **XOR** (exclusive OR) function

Patterns (bipolar)		
x_1	x_2	y
– 1	– 1	– 1
– 1	1	1
1	– 1	1
1	1	– 1



X : Class I (y = 1)
O : Class II (y = –1)

Fig. 4.1.6

- No line can separate these two classes, as can be seen from the fact that the following linear inequality system has no solution.

$$\begin{cases} b - w_1 - w_2 < 0 & (1) \\ b - w_1 + w_2 \geq 0 & (2) \\ b + w_1 - w_2 \geq 0 & (3) \\ b + w_1 + w_2 < 0 & (4) \end{cases}$$

because we have $b < 0$ from (1) + (4), and $b >= 0$ from (2) + (3), which is a contradiction.

4.2 Activation Functions

- Activation functions also known as transfer function is used to map input nodes to output nodes in certain fashion.
- The activation function is the most important factor in a neural network which decided whether or not a neuron will be activated or not and transferred to the next layer.
- Activation functions help in normalizing the output between 0 to 1 or – 1 to 1. It helps in the process of backpropagation due to their differentiable property. During backpropagation, loss function gets updated, and activation function helps the gradient descent curves to achieve their local minima.
- Activation function basically decides in any neural network that given input or receiving information is relevant or it is irrelevant.
- These activation function makes the multilayer network to have greater representational power than single layer network only when non-linearity is introduced.
- The input to the activation function is sum which is defined by the following equation.

$$\text{sum} = I_1 W_1 + I_2 W_2 + \ldots + I_n W_n$$

$$= \sum_{j=1}^{n} I_j W_j + b$$

- **Activation Function : Logistic Function**

$$f(sum) = \frac{1}{(1+e^{-s * sum})}$$

$$= (1+s^{-s \times sum})^{-1}$$

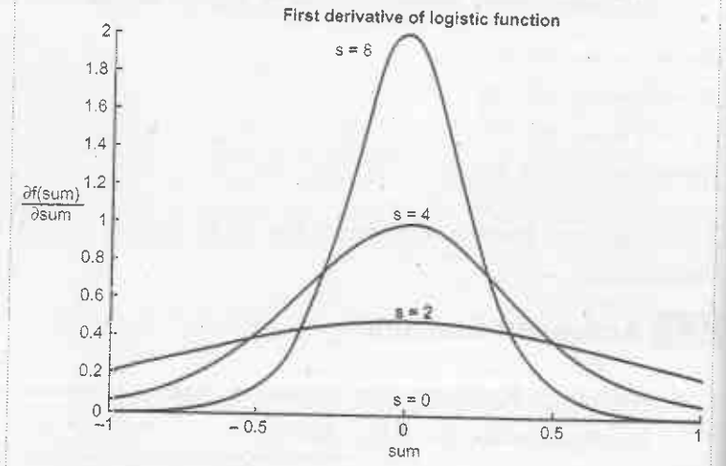


Fig. 4.2.1

- **Logistic function** monotonically increases from a lower limit (0 or - 1) to an upper limit (+1) as sum increases. In which values vary between 0 and 1, with a value of 0.5 when I is zero.
- **Activation Function : Arc Tangent**

$$f(sum) = \frac{2}{\pi} \tan^{-1} (s \times sum)$$

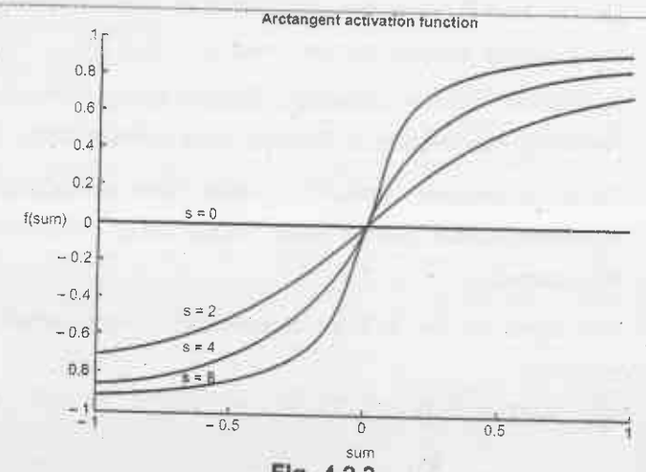


Fig. 4.2.2

- **Activation Function : Hyperbolic Tangent**

$$f(sum) = \tanh (s * I)$$

$$= \frac{e^{s \times sum} - e^{-s \times sum}}{e^{s \times sum} + e^{-s \times sum}}$$

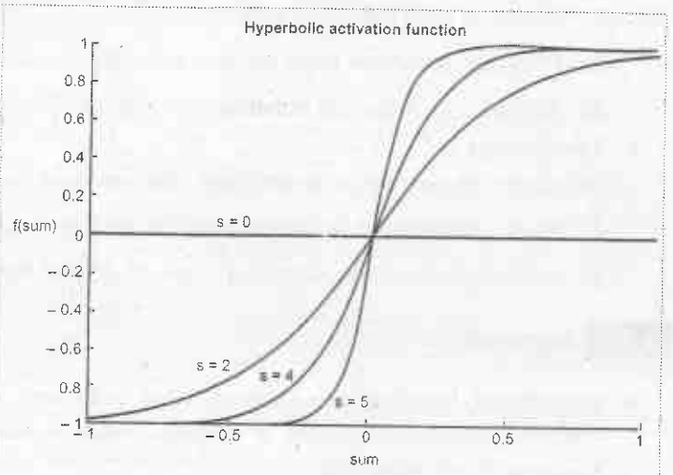


Fig. 4.2.3

4.2.1 Identity or Linear Activation Function

- A linear activation is a mathematical equation used for obtaining output vectors with specific properties.
- It is a simple straight line activation function where our function is directly proportional to the weighted sum of neurons or input.
- Linear activation functions are better in giving a wide range of activations and a line of a positive slope may increase the firing rate as the input rate increases.
- Fig. 4.2.4 shows identity function.

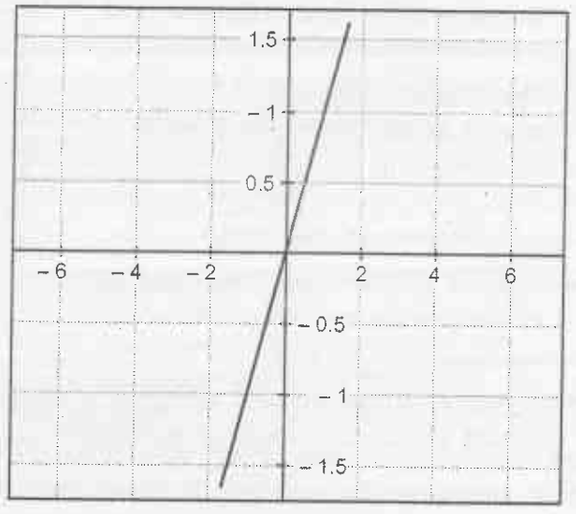


Fig. 4.2.4

- The equation for linear activation function is :

$$f(x) = a.x$$

When a = 1 then f(x) = x and this is a special case known as identity.

- **Properties :**
  1. Range is – infinity to + infinity.
  2. Provides a convex error surface so optimisation can be achieved faster.
  3. df(x)/dx = a which is constant. So cannot be optimised with gradient descent.
- **Limitations :**
  1. Since the derivative is constant, the gradient has no relation with input.
  2. Back propagation is constant as the change is delta x.
  3. Activation function does not work in neural networks in practice.

4.2.2 Sigmoid

- A sigmoid function produces a curve with an "S" shape. The example sigmoid function shown on the left is a special case of the logistic function, which models the growth of some set.

$$\text{sig}(t) = \frac{1}{1+e^{-t}}$$

- In general, a sigmoid function is real-valued and differentiable, having a non-negative or non-positive first derivative, one local minimum and one local maximum.
- The logistic sigmoid function is related to the hyperbolic tangent as follows :

$$1 - 2\,\text{sig}(x) = 1 - 2\frac{1}{1+e^{-x}} = -\tanh\frac{x}{2}$$

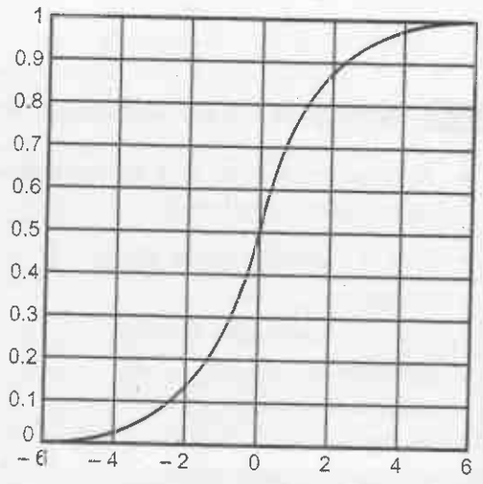


Fig. 4.2.5

- Sigmoid functions are often used in artificial neural networks to introduce nonlinearity in the model.
- A neural network element computes a linear combination of its input signals and applies a sigmoid function to the result.
- A reason for its popularity in neural networks is because the sigmoid function satisfies a property between the derivative and itself such that it is computationally easy to perform.

$$\frac{d}{dt}\text{sig}(t) = \text{sig}(t)\,(1-\text{sig}(t))$$

- Derivatives of the sigmoid function are usually employed in learning algorithms.

4.3 Gradient Descent Optimization

- Gradient Descent is an optimization algorithm in gadget mastering used to limit a feature with the aid of iteratively moving towards the minimal fee of the characteristic.
- We essentially use this algorithm when we have to locate the least possible values which could fulfill a given fee function. In gadget getting to know, greater regularly that not we try to limit loss features (like Mean Squared Error).

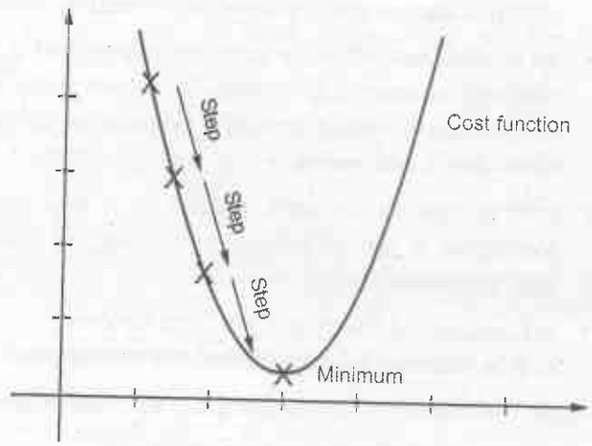


Fig. 4.3.1 Gradient descent algorithm

By minimizing the loss characteristic, we will improve our model and Gradient Descent is one of the most popular algorithms used for this cause.

- The graph above shows how exactly a Gradient Descent set of rules works.
- We first take a factor in the value function and begin shifting in steps in the direction of the minimum factor. The size of that step or how quickly we ought to converge to the minimum factor is defined by Learning Rate. We can cowl more location with better learning fee but at the risk of overshooting the minima. On the opposite hand, small steps/smaller gaining knowledge of charges will eat a number of time to attain the lowest point.
- Now, the direction wherein algorithm has to transport (closer to minimal) is also important. We calculate this by way of using derivatives. You need to be familiar with derivatives from calculus. A spinoff is largely calculated because the slope of the graph at any specific factor. We get that with the aid of finding the tangent line to the graph at that point. The extra steep the tangent, would suggest that more steps would be needed to reach minimum point, much less steep might suggest lesser steps are required to reach the minimum factor.

4.3.1 Stochastic Gradient Descent

- The word 'stochastic' means a system or a process that is linked with a random probability. Hence, in Stochastic Gradient Descent, a few samples are selected randomly instead of the whole data set for each iteration.

- Stochastic Gradient Descent (SGD) is a type of gradient descent that runs one training example per iteration. It processes a training epoch for each example within a dataset and updates each training example's parameters one at a time.
- As it requires only one training example at a time, hence it is easier to store in allocated memory. However, it shows some computational efficiency losses in comparison to batch gradient systems as it shows frequent updates that require more detail and speed.
- Further, due to frequent updates, it is also treated as a noisy gradient. However, sometimes it can be helpful in finding the global minimum and also escaping the local minimum.
- Advantages of Stochastic gradient descent :
  a) It is easier to allocate in desired memory.
  b) It is relatively fast to compute than batch gradient descent.
  c) It is more efficient for large datasets.
- Disadvantages of Stochastic Gradient Descent :
  a) SGD requires a number of hyperparameters such as the regularization parameter and the number of iterations.
  b) SGD is sensitive to feature scaling.

4.4 Error Backpropagation

- **Backpropagation** is a training method used for a multi-layer neural network. It is also called the generalized delta rule. It is a gradient descent method which minimizes the total squared error of the output computed by the net.
- The **backpropagation** algorithm looks for the minimum value of the error function in weight space using a technique called the delta rule or gradient descent. The weights that minimize the error function is then considered to be a solution to the learning problem.
- **Backpropagation** is a systematic method for training multiple layer ANN. It is a generalization of Widrow-Hoff error correction rule. 80 % of ANN applications uses **backpropagation.**
- Fig. 4.4.1 (See on next page) shows backpropagation network.
- Consider a simple neuron :
  a. Neuron has a summing junction and activation function.
  b. **Any** non linear function which differentiable everywhere and increases everywhere with sum can be used as activation function.

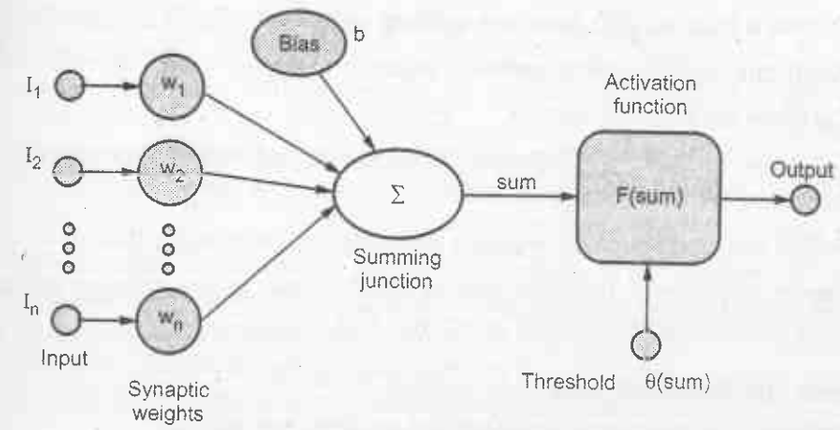


Fig. 4.4.1 Backpropagation network

  c. Examples : Logistic function, Arc tangent function, Hyperbolic tangent activation function.
- These activation function makes the multilayer network to have greater representational power than single layer network only when non-linearity is introduced.
- **Need of hidden layers :**
  1. A network with only two layers (input and output) can only represent the input with whatever representation already exists in the input data.
  2. If the data is discontinuous or non-linearly separable, the innate representation is inconsistent and the mapping cannot be learned using two layers (Input and Output).
  3. Therefore, hidden layer(s) are used between input and output layers
- **Weights** connects unit (neuron) in one layer only to those in the next higher layer. The output of the unit is scaled by the value of the connecting weight and it is fed forward to provide a portion of the activation for the units in the next higher layer.
- **Backpropagation** can be applied to an artificial neural network with any number of hidden layers. The training objective is to adjust the weights so that the application of a set of inputs produces the desired outputs.
- **Training procedure :** The network is usually trained with a large number of input - output pairs.
  1. Generate weights randomly to small random values (both positive and negative) to ensure that the network is not saturated by large values of weights.

2. Choose a training pair from the training set.
3. Apply the input vector to network input.
4. Calculate the network output.
5. Calculate the error, the difference between the network output and the desired output.
6. Adjust the weights of the network in a way that minimizes this error.
7. Repeat steps 2 - 6 for each pair of input-output in the training set until the error for the entire system is acceptably low.

Forward pass and backward pass :

- Backpropagation neural network training involves two passes.
  1. In the forward pass, the input signals moves forward from the network input to the output.
  2. In the backward pass, the calculated error signals propagate backward through the network, where they are used to adjust the weights.
  3. In the forward pass, the calculation of the output is carried out, layer by layer, in the forward direction. The output of one layer is the input to the next layer.
- In the reverse pass,
  a. The weights of the output neuron layer are adjusted first since the target value of each output neuron is available to guide the adjustment of the associated weights, using the delta rule.
  b. Next, we adjust the weights of the middle layers. As the middle layer neurons have no target values, it makes the problem complex.
- **Selection of number of hidden units :** The number of hidden units depends on the number of input units.
  1. Never choose **h** to be more than twice the number of input units.
  2. You can load p patterns of I elements into $\log_2$ p hidden units.
  3. Ensure that we must have at least 1/e times as many training examples.
  4. Feature extraction requires fewer hidden units than inputs.
  5. Learning many examples of disjointed inputs requires more hidden units than inputs.
  6. The number of hidden units required for a classification task increases with the number of classes in the task. Large networks require longer training times.

Factors influencing Backpropagation training

- The training time can be reduced by using :
  1. **Bias :** Networks with biases can represent relationships between inputs and outputs more easily than networks without biases. Adding a bias to each neuron is usually desirable to offset the origin of the activation function. The weight of the bias is trainable similar to weight except that the input is always +1.
  2. **Momentum :** The use of momentum enhances the stability of the training process. Momentum is used to keep the training process going in the same general direction analogous to the way that momentum of a moving object behaves. In backpropagation with momentum, the weight change is a combination of the current gradient and the previous gradient.

4.4.1 Advantages and Disadvantages

Advantages of backpropagation :

1. It is simple, fast and easy to program.
2. Only numbers of the input are tuned and not any other parameter.
3. No need to have prior knowledge about the network.
4. It is flexible.
5. A standard approach and works efficiently.
6. It does not require the user to learn special functions.

Disadvantages of backpropagation :

1. Backpropagation possibly be sensitive to noisy data and irregularity.
2. The performance of this is highly reliant on the input data.
3. Needs excessive time for training.
4. The need for a matrix-based method for backpropagation instead of mini - batch.

4.5 Shallow Networks

- The terms shallow and deep refer to the number of layers in a neural network; shallow neural networks refer to a neural network that have a small number of layers, usually regarded as having a single hidden layer and deep neural networks refer to neural networks that have multiple hidden layers. Both types of networks perform certain tasks better than the other and selecting the right network depth is important for creating a successful model.
- In a shallow neural network, the values of the feature vector of the data to be classified (the input layer) are passed to a hidden layer of nodes (neurons) each of

which generates a response according to some activation function, g, acting on the weighted sum of those values, z.

- The responses of each unit in the hidden layer is then passed to a final, output layer (which may consist of a single unit), whose activation produces the classification prediction output.

4.6 Deep Network

- Deep learning is a new area of machine learning research, which has been introduced with the objective of moving machine learning closer to one of its original goals. Deep learning is about learning multiple levels of representation and abstraction that help to make sense of data such as images, sound and text.
- 'Deep learning' means using a neural network with several layers of nodes between input and output. It is generally better than other methods on image, speech and certain other types of data because the series of layers between input and output do feature identification and processing in a series of stages, just as our brains seem to.
- Deep Learning emphasizes the network architecture of today's most successful machine learning approaches. These methods are based on "deep" multi - layer neural networks with many hidden layers.

4.6.1 TensorFlow

- TensorFlow is one of the most popular frameworks used to build deep learning models. The framework is developed by Google Brain Team.
- Languages like C++, R and Python are supported by the framework to create the models as well as the libraries. This framework can be accessed from both - desktop and mobile.
- The translator used by Google is the best example of TensorFlow. In this, the model is created by adding the functionalities of text classification, natural language processing, speech or handwriting recognition, image recognition, etc.
- The framework has its own visualization toolkit, named TensorBoard which helps in powerful data visualization of the network along with its performance.
- One more tool added in TensorFlow, TensorFlow Serving, can be used for quick and easy deployment of the newly developed algorithms without introducing any change in the existing API or architecture.
- TensorFlow framework comes along with a detailed documentation for the users to adapt it quickly and easily, making it the most preferred deep learning framework to model deep learning algorithms.

- Some of the characteristics of TensorFlow is :
  - Multiple GPU supported.
  - One can visualize graphs and queues easily using TensorBoard.
  - Powerful documentation and larger support from community.

4.6.2 Keras

- If you are comfortable in programming with Python, then learning Keras will not prove hard to you. This will be the most recommended framework to create deep learning models for ones having a sound of Python.
- Keras is built purely on Python and can run on the top of TensorFlow. Due to its complexity and use of low - level libraries, TensorFlow can be comparatively harder to adapt for the new users as compared to Keras. Users those who are beginners in deep learning, and find its models difficult to understand in TensorFlow generally prefer Keras as it solves all complex models in no time.
- Keras has been developed keeping in mind the complexities in the deep learning models, and hence it can run quickly to get the results in minimum time. Convolutional as well as Recurrent Neural networks are supported in Keras. The framework can run easily on CPU and GPU.
- The models in Keras can be classified into 2 categories :

1. Sequential model :

The layers in the deep learning model are defined in a sequential manner. Hence the implementation of the layers in this model will also be done sequentially.

2. Keras functional API :

Deep learning models that has multiple outputs, or has shared layers, i.e. more complex models can be implemented in Keras functional API.

4.6.3 Difference between Deep Network and Shallow Network

Sr. No.	Deep network	Shallow network
1.	Deep network contains many hidden layers.	Shallow network contains only one hidden layer.
2.	Deep network can compactly express highly complex functions over input space	Shallow networks with one Hidden layer cannot place complex functions over the input space.

3.	Training in DN is easy and no issue of local minima in DN	Shallow network is more difficult to train with our current algorithms
4.	DN can fit functions better with less parameters than a shallow network	Shallow net's needs more parameters to have better fit

4.7 Vanishing Gradient Problem

- The vanishing gradient problem is a problem that user face, when we are training Neural Networks by using gradient-based methods like **backpropagation**. This problem makes it difficult to learn and tune the parameters of the earlier layers in the network.
- The vanishing gradient problem is essentially a situation in which a deep multilayer feed-forward network or a Recurrent Neural Network (RNN) does not have the ability to propagate useful gradient information from the output end of the model back to the layers near the input end of the model.
- It results in models with many layers being rendered unable to learn on a specific dataset. It could even cause models with many layers to prematurely converge to a substandard solution.
- When the backpropagation algorithm advances downwards or backward going from the output layer to the input layer, the gradients tend to shrink, becoming smaller and smaller till they approach zero. This ends up leaving the weights of the initial or lower layers practically unchanged. In this situation, the gradient descent does not ever end up converging to the optimum.
- Vanishing gradient does not necessarily imply that the gradient vector is all zero. It implies that the gradients are minuscule, which would cause the learning to be very slow.
- The most important solution to the vanishing gradient problem is a **specific** type of neural network called Long Short-Term Memory Networks (LSTMs).
- Indication of vanishing gradient problem :
  a) The parameters of the higher layers change to a great extent, while the parameters of lower layers barely change.
  b) The model weights could become 0 during training.
  c) The model learns at a particularly slow pace and the training could stagnate at a very early phase after only a few iterations.
- Some methods that are proposed to overcome the vanishing gradient problem :
  a) Residual neural networks (ResNets)

  b) Multi-level hierarchy
  c) Long Short Term Memory (LSTM)
  d) Faster hardware
  e) ReLU
  f) Batch normalization

4.8 ReLU

- Rectified Linear Unit (ReLU) solve the vanishing gradient problem. ReLU is a non-linear function or piecewise linear function that will output the input directly if it is positive, otherwise, it will output zero.
- It is the most commonly used activation function in neural networks, especially in Convolutional Neural Networks (CNNs) and Multilayer perceptron's.
- **Mathematically**, it is expressed as

  $$f(x) = \max(0, x)$$

  where x : **input to neuron**
- Fig. 4.8.1 shows ReLU function

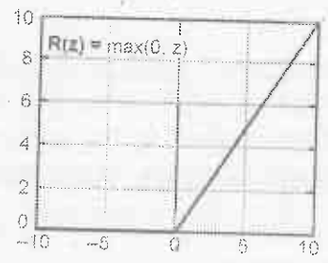


Fig. 4.8.1 ReLU function

- The derivative of an activation function is required when updating the weights during the **back-propagation** of the error. The slope of ReLU is 1 for positive values and 0 for negative values. It becomes **non-differentiable** when the input x is zero, but it can be safely assumed to be zero and causes no problem in practice.
- ReLU is used in the hidden layers instead of Sigmoid or tanh. The ReLU function solves the problem of computational complexity of the Logistic Sigmoid and Tanh functions.
- A ReLU activation unit is known to be less likely to create a vanishing gradient problem because its derivative is always 1 for positive values of the argument.

- **Advantages of ReLU function**
  a) ReLU is simple to compute and has a predictable gradient for the backpropagation of the error.
  b) Easy to implement and very fast.
  c) The calculation speed is very fast. The ReLU function has only a direct relationship.
  d) It can be used for deep network training.
- **Disadvantages of ReLU function**
  a) When the input is negative, ReLU is not fully functional which means when it comes to the wrong number installed, ReLU will die. This problem is also known as the Dead Neurons problem.
  b) ReLU function can only be used within hidden layers of a Neural Network Model.

4.8.1 LReLU and ERELU

1. LReLU

- The Leaky ReLU is one of the most well-known activation function. It is the same as ReLU for positive numbers. But instead of being 0 for all negative values, it has a constant slope (less than 1.).
- Leaky ReLU is a type of activation function that helps to prevent the function from becoming saturated at 0. It has a small slope instead of the standard ReLU which has an infinite slope.
- Leaky ReLUs are one attempt to fix the "dying ReLU" problem. Fig. 4.8.2 shows LReLU function.

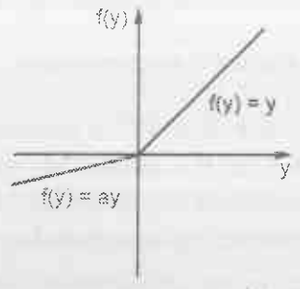


Fig. 4.8.2 LReLU function

- The leak helps to increase the range of the ReLU function. Usually, the value of a is 0.01 or so.

- The motivation for using LReLU instead of ReLU is that constant zero gradients can also result in slow learning, as when a saturated neuron uses a sigmoid activation function

2. EReLU

- An Elastic ReLU (EReLU) considers a slope randomly drawn from a uniform distribution during the training for the positive inputs to control the amount of non-linearity.
- The EReLU is defined as : **EReLU(x) = max(Rx; 0)** in the output range of [0;1) where R is a random number
- At the test time, the EReLU becomes the identity function for positive inputs.

4.9 Hyperparameter Tuning

- Hyperparameters are parameters whose values control the learning process and determine the values of model parameters that a learning algorithm ends up learning.
- While designing a machine learning model, one always has multiple choices for the architectural design for the model. This creates a confusion on which design to choose for the model based on its optimality. And due to this, there are always trials for defining a perfect machine learning model.
- The parameters that are used to define these machine learning models are known as the hyperparameters and the rigorous search for these parameters to build an optimized model is known as hyperparameter tuning.
- Hyperparameters are not model parameters, which can be directly trained from data. Model parameters usually specify the way to transform the input into the required output, whereas hyperparameters define the actual structure of the model that gives the required data.

4.9.1 Layer Size

- Layer size is defined by the number of neurons in a given layer. Input and output layers are relatively easy to figure out because they correspond directly to how our modeling problem handles input and ouput.
- For the input layer, this will match up to the number of features in the input vector. For the output layer, this will either be a single output neuron or a number of neurons matching the number of classes we are trying to predict.
- It is obvious that a neural network with 3 layers will give better performance than that of 2 layers. Increasing more than 3 doesn't help that much in neural networks. In the case of CNN, an increasing number of layers makes the model better.

4.9.2 Magnitude : Learning Rate

- The amount that the weights are updated during training is referred to as the step size or the learning rate. Specifically, the learning rate is a configurable hyper-parameter used in the training of neural networks that has a small positive value, often in the range between 0.0 and 1.0.
- For example, if learning rate is 0.1, then the weights in the network are updated 0.1 * (estimated weight error) or 10 % of the estimated weight error each time the weights are updated. The learning rate hyper-parameter controls the rate or speed at which the model learns.
- Learning rates are tricky because they end up being specific to the dataset and even to other hyper-parameters. This creates a lot of overhead for finding the right setting for hyper-parameters.
- Large learning rates () make the model learn faster but at the same time it may cause us to miss the minimum loss function and only reach the surrounding of it. In cases where the learning rate is too large, the optimizer overshoots the minimum and the loss updates will lead to divergent behaviours.
- On the other hand, choosing lower learning rate values gives a better chance of finding the local minima with the trade-off of needing larger number of epochs and more time.
- **Momentum** can accelerate learning on those problems where the high-dimensional weight space that is being navigated by the optimization process has structures that mislead the gradient descent algorithm, such as flat regions or steep curvature.

4.10 Normalization

- Normalization is a data preparation technique that is frequently used in machine learning. The process of transforming the columns in a dataset to the same scale is referred to as normalization. Every dataset does not need to be normalized for machine learning.
- Normalization makes the features more consistent with each other, which allows the model to predict outputs more accurately. The main goal of normalization is to make the data homogenous over all records and fields.
- Normalization refers to rescaling real-valued numeric attributes into a 0 to 1 range. Data normalization is used in machine learning to make model training less sensitive to the scale of features.

- Normalization is important in such algorithms as k-NN, support vector machines, neural networks, and principal components. The type of feature preprocessing and normalization that's needed can depend on the data.

4.10.1 Batch Normalization

- It is a method of adaptive reparameterization, motivated by the difficulty of training very deep models. In Deep networks, the weights are updated for each layer. So the output will no longer be on the same scale as the input.
- When we input the data to a machine or deep learning algorithm we tend to change the values to a balanced scale because, we ensure that our model can generalize appropriately.
- Batch normalization is a technique for standardizing the inputs to layers in a neural network. Batch normalization was designed to address the problem of internal covariate shift, which arises as a consequence of updating multiple-layer inputs simultaneously in deep neural networks.
- Batch normalization is applied to individual layers, or optionally, to all of them : In each training iteration, we first normalize the inputs by subtracting their mean and dividing by their standard deviation, where both are estimated based on the statistics of the current mini-batch.
- Next, we apply a scale coefficient and an offset to recover the lost degrees of freedom. It is precisely due to this normalization based on batch statistics that batch normalization derives its name.
- We take the output $a^{[i-1]}$ from the preceding layer, and multiply by the weights W and add the bias b of the current layer. The variable I denotes the current layer.

$$Z^{[i]} = W^{[i]}a^{[i-1]} + b^{[i]}$$

- Next, we usually apply the non-linear activation function that results in the output $a^{[i]}$ of the current layer. When applying batch norm, we correct our data before feeding it to the activation function.
- To apply batch norm, calculate the mean as well as the variance of current z.

$$\mu = \frac{1}{m}\sum_{j=1}^{m} Z_j$$

- When calculating the variance, we add a small constant to the variance to prevent potential divisions by zero.

$$\sigma^2 = \frac{1}{m}\sum_{j=1}^{m}(Z_j - \mu)^2 + \epsilon$$

- To normalize the data, we subtract the mean and divide the expression by the standard deviation.

$$Z^{[i]} = \frac{Z^{[i]} - \mu}{\sqrt{\sigma^2}}$$

- This operation scales the inputs to have a mean of 0 and a standard deviation of 1.
- Advantages of Batch Normalisation :
  a) The model is less delicate to hyperparameter tuning.
  b) Shrinks internal covariant shift.
  c) Diminishes the reliance of gradients on the scale of the parameters or their underlying values.
  d) Dropout can be evacuated for regularization.

4.11 Regularization

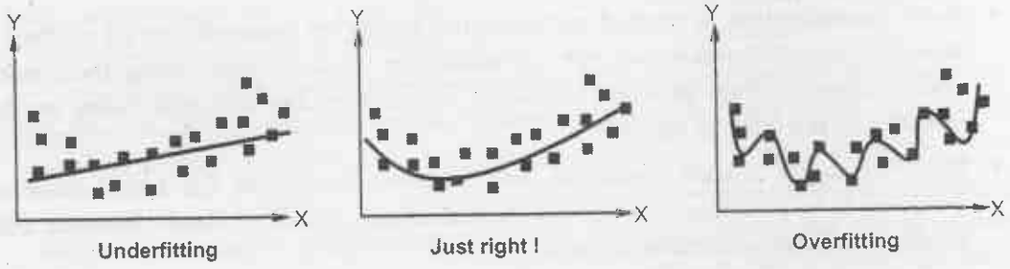


Fig. 4.11.1

- Just have a look at the above figure, and we can immediately predict that once we try to cover every minutest feature of the input data, there can be irregularities in the extracted features, which can introduce noise in the output. This is referred to as "Overfitting".
- This may also happen with the lesser number of features extracted as some of the important details might be missed out. This will leave an effect on the accuracy of the outputs produced. This is referred to as "Underfitting".
- This also shows that the complexity for processing the input elements increases with overfitting. Also, neural networks being a complex interconnection of nodes, the issue of overfitting may arise frequently.
- To eliminate this, regularization is used, in which we have to make the slightest modification in the design of the neural network, and we can get better outcomes.

4.11.1 Regularization in Machine Learning

- One of the most important factors that affect the machine learning model is overfitting.
- The machine learning model may perform poorly if it tries to capture even the noise present in the dataset applied for training the system, which ultimately results in overfitting. In this context, noise doesn't mean the ambiguous or false data, but those inputs which do not acquire the required features to execute the machine learning model.
- Analyzing these data inputs may surely make the model flexible, but the risk of overfitting will also increase accordingly.
- One of the ways to avoid this is to cross validate the training dataset, and decide accordingly the parameters to include that can increase the efficiency and performance of the model.
- Let this be the simple relation for linear regression :

$$Y \approx b_0 + b_1X_1 + b_2X_2 + \ldots\ b_pX_p$$

Where Y = Learned relation

β = Co-efficient estimators for different variables and/or predictors (X)

- Now, we shall introduce a loss function, that implements the fitting procedure, which is referred to as "Residual Sum of Squares" or RSS.
- The co-efficient in the function is chosen in such a way that it can minimize the loss function easily.

Hence,

$$RSS = \sum_{i=1}^{n}\left(Y_i - \beta_0 - \sum_{j=1}^{p}\beta_j X_{ij}\right)^2$$

- Above equation will help in adjusting the co-efficient function depending on the training dataset.
- In case noise is present in the training dataset, then the adjusted co-efficient won't be generalized when the future datasets will be introduced. Hence, at this point, regularization comes into picture and makes this adjusted co-efficient shrink towards zero.
- One of the methods to implement this is the ridge regression, also known as L2 regularization. Lets have a quick overview on this.

4.11.2 Ridge Regression (L2 Regularization)

- Ridge regression, also known as L2 regularization, is a technique of regularization to avoid the overfitting in training data set, which introduces a small bias in the training model, through which one can get long term predictions for that input.
- In this method, a penalty term is added to the cost function. This amount of bias altered to the cost function in the model is also known as ridge regression penalty.
- Hence, the equation for the cost function, after introducing the ridge regression penalty is as follows :

$$\sum_{i=1}^{m}(y_i - y_i')^2 = \sum_{i=1}^{n}\left(Y_i - \sum_{j=1}^{n}\beta_j \times X_{ij}\right)^2 + \lambda \sum_{j=0}^{n}\beta_j^2$$

Here, λ is multiplied by the square of the weight set for the individual feature of the input data. This term is ridge regression penalty.

- It regularizes the co-efficient set for the model and hence the ridge regression term deduces the values of the coefficient, which ultimately helps in deducing the complexity of the machine learning model.
- From the above equation, we can observe that if the value to zero, the last term on the right - hand side will tend to zero king the above equation a representation of a simple linear regressio
- Hence, lower the value of, the model will tend t gression.
- This model is important to execute the ne ks for machine learning, as there would be risks of failure for gen ear regression models, if there are dependencies found between it . Hence, ridge regression is used here.

4.11.3 Lasso Regression (L zation)

- One more technique to the overfitting, and thus the complexity of the model is the lasso regr
- Lasso regression sta for Least Absolute and Selection Operator and is also sometimes known a egularization.
- The equation for t sso regression is almost same as that of the ridge regression, except fo ange that the value of the penalty term is taken as the absolute weights.
- The advantage of tak the absolute values is that its slope can shrink to 0, as compared to the rid gression, where the slope will shrink it near to 0.

- The following equation gives the cost function defined in the Lasso regression :

$$\sum_{i=1}^{m}(y_i - y_i')^2 = \sum_{i=1}^{n}\left(Y_i - \sum_{j=1}^{n}\beta_j \times X_{ij}\right)^2 + \lambda \sum_{j=0}^{n}|\beta_j|^2$$

- Due to the acceptance of absolute values for the cost function, some of the features of the input dataset can be ignored completely while evaluating the machine learning model and hence the feature selection and overfitting can be reduced to much extent.
- On the other hand, ridge regression does not ignore any feature in the model and includes it all for model evaluation. The complexity of the model can be reduced using the shrinking of co-efficient in the ridge regression model.

4.11.4 Dropout

- Dropout was introduced by "Hinton et al"and this method is now very popular. It consists of setting to zero the output of each hidden neuron in chosen layer with some probability and is proven to be very effective in reducing overfitting.
- Fig. 4.11.2 shows dropout regulations.

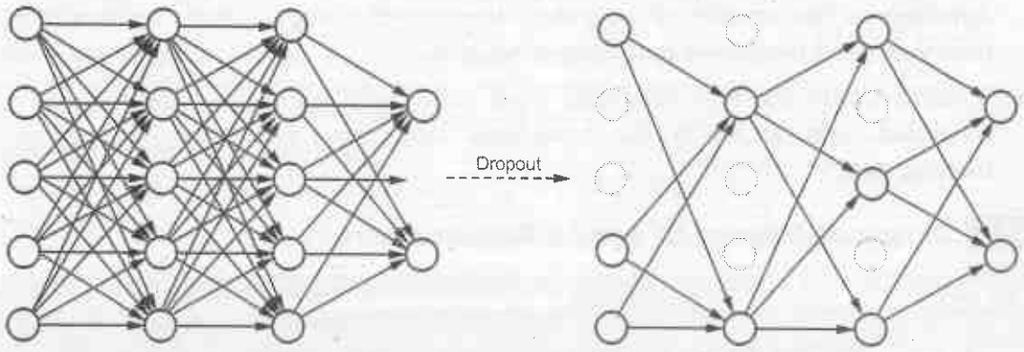


Fig. 4.11.2 Dropout regulation

- To achieve dropout regularization, some neurons in the artificial neural network are randomly disabled. That prevents them from being too dependent on one another as they learn the correlations. Thus, the neurons work more independently and the artificial neural network learns multiple independent correlations in the data based on different configurations of the neurons.
- It is used to improve the training of neural networks by omitting a hidden unit. It also speeds training.

- Dropout is driven by randomly dropping a neuron so that it will not contribute to the forward pass and back-propagation.
- Dropout is an inexpensive but powerful method of regularizing a broad family of models.

4.11.5 DropConnect

- DropConnect, known as the generalized version of Dropout, is the method used for regularizing deep neural networks. Fig. 4.11.3 shows dropconnect.

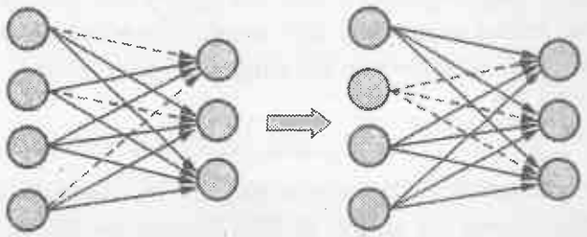


Fig. 4.11.3 Dropconnect

- DropConnect has been proposed to add more noise to the network. The primary difference is that instead of randomly dropping the output of the neurons, we randomly drop the connection between neurons.
- In other words, the fully connected layer with DropConnect becomes a sparsely connected layer in which the connections are chosen at random during the training stage.

4.11.6 Difference between L1 and L2 Regularization

Sr. No.	L1 Regularization	L2 Regularization
1.	Penelizes the sum of absolute value of weights.	Penalizes the sum of square weights.
2.	It has a sparse solution.	It has a non-sparse solution.
3.	It gives multiple solutions.	It has only one solution.
4.	Constructed in feature selection.	No feature selection.
5.	Robust to outliers.	Not robust to outliers.

6.	It generates simple and interpretable models.	It gives more accurate predictions when the output variable is the function of whole input variables.
7.	Unable to learn complex data patterns.	Able to learn complex data patterns.
8.	Computationally inefficient over non-space conditions.	Computationally efficient because of having analytical solutions.

4.12 Two Marks Questions with Answers

Q.1 Explain multilayer perceptron.

Ans. : The Multilayer Perceptron (MLP) model features multiple layers that are interconnected in such a way that they form a feed-forward neural network. Each neuron in one layer has directed connections to the neurons of a separate layer. It consists of three types of layers : the input layer, output layer and hidden layer.

Q.2 What is vanishing gradient problem ?

Ans. : When back-propagation is used, the earlier layers will receive very small updates compared to the later layers. This problem is referred to as the vanishing gradient problem. The vanishing gradient problem is essentially a situation in which a deep multilayer feed-forward network or a Recurrent Neural Network (RNN) does not have the ability to propagate useful gradient information from the output end of the model back to the layers near the input end of the model.

Q.3 Explain advantages deep learning.

Ans. : Advantages of deep learning :

- No need for feature engineering.
- DL solves the problem on the end-to-end basis.
- Deep learning gives more accuracy.

Q.4 Explain back propagation.

Ans. : Backpropagation is a training method used for a multi-layer neural network. It is also called the generalized delta rule. It is a gradient descent method which minimizes the total squared error of the output computed by the net.

Q.5 What is hyperparameters ?

Ans. : Hyperparameters are parameters whose values control the learning process and determine the values of model parameters that a learning algorithm ends up learning.

Q.6 Define ReLU.

Ans. : Rectified Linear Unit (ReLU) solve the vanishing gradient problem. ReLU is a nonlinear function or piecewise linear function that will output the input directly if it is positive, otherwise, it will output zero.

Q.7 What is vanishing gradient problem ?

Ans. : The vanishing gradient problem is a problem that user face, when we are training Neural Networks by using gradient-based methods like backpropagation. This problem makes it difficult to learn and tune the parameters of the earlier layers in the network.

Q.8 Define normalization.

Ans. : Normalization is a data pre-processing tool used to bring the numerical data to a common scale without distorting its shape.

Q.9 What is batch normalization ?

Ans. : It is a method of adaptive reparameterization, motivated by the difficulty of training very deep models. In Deep networks, the weights are updated for each layer. So the output will no longer be on the same scale as the input.

Q.10 Explain advantages of ReLU function.

Ans. : Advantages of ReLU function :

a) ReLU is simple to compute and has a predictable gradient for the backpropagation of the error.

b) Easy to implement and very fast.

c) It can be used for deep network training.

Q.11 Explain ridge regression.

Ans. : Ridge regression, also known as L2 regularization, is a technique of regularization to avoid the overfitting in training data set, which introduces a small bias in the training model, through which one can get long term predictions for that input.

Q.12 Explain dropout.

Ans. : Dropout was introduced by "Hinton et al" and this method is now very popular. It consists of setting to zero the output of each hidden neuron in chosen layer with some probability and is proven to be very effective in reducing overfitting.

Q.13 Explain disadvantages of deep learning.

Ans. : Disadvantages of deep learning

- DL needs high-performance hardware.
- DL needs much more time to train
- it is very difficult to assess its performance in real world applications
- it is very hard to understand

Q.14 Explain need of hidden layers.

Ans. :

1. A network with only two layers (input and output) can only represent the input with whatever representation already exists in the input data.
2. If the data is discontinuous or non-linearly separable, the innate representation is inconsistent, and the mapping cannot be learned using two layers (Input and Output).
3. Therefore, hidden layer(s) are used between input and output layers.

Q.15 Explain activation functions.

Ans. : Activation functions also known as transfer function is used to map input nodes to output nodes in certain fashion. It helps in normalizing the output between 0 to 1 or – V1 to 1. The activation function is the most important factor in a neural network which decided whether or not a neuron will be activated or not and transferred to the next layer.

❑❑❑

Notes

UNIT V

5 Design and Analysis of Machine Learning Experiments

Syllabus

Guidelines for machine learning experiments, Cross Validation (CV) and resampling - K-fold CV, bootstrapping, measuring classifier performance, assessing a single classification algorithm and comparing two classification algorithms - t test, McNemar's test, K-fold CV paired t test.

Contents

5.1 Machine Learning Life Cycle

- The Machine Leaning (ML) model management and the delivery of highly performing model is as important as the initial build of the model by choosing right dataset. The concepts around model retraining, model versioning, model deployment and model monitoring are the basis for machine learning operations that helps the data science teams deliver highly performing models.
- The use of machine leaning has increased substantially in enterprise data analytics scenarios to extract valuable insights from the business data. Hence, it is very important to have an ecosystem to build, test, deploy and maintain the enterprise grade machine learning models in production environments.
- The ML model development involves data acquisition from multiple trusted sources, data processing to make suitable for building the model, choose algorithm to build the model, build model, compute performance metrics and choose best performing model.
- The model maintenance plays critical role once the model is deployed into production. The maintenance of machine learning model includes keeping the model up to date and relevant in tune with the source data changes as there is a risk of model becoming outdated in course of time.
- Machine learning model lifecycle refers to the process that covers right from source data identification to model development, model deployment and model maintenance. At high level, the entire activities fall under two broad categories, such as ML model development and ML model operations.
- The machine learning lifecycle process is shows in Fig. 5.1.1 and it includes the following phases :
  1. Business goal identification
  2. ML problem framing
  3. Data processing (Data collection, data preprocessing, feature engineering)
  4. Model development (Training, tuning, evaluation)
  5. Model deployment (Inference, prediction)
  6. Model monitoring.
- **Business goal :** An organization considering ML should have a clear idea of the problem and the business value to be gained by solving that problem. We must be able to measure business value against specific business objectives and success criteria.
- **ML problem framing :** In this phase, the business problem is framed as a machine learning problem : What is observed and what should be predicted (known as a

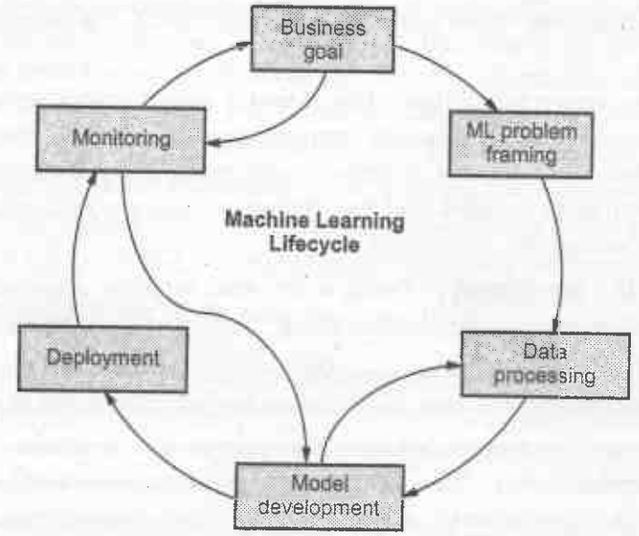


Fig. 5.1.1 Machine learning lifecycle process

label or target variable). Determining what to predict and how performance and error metrics must be optimized is a key step in this phase.

- **Data processing :** Training an accurate ML model requires data processing to convert data into a usable format. Data processing steps include collecting data, preparing data and feature engineering that is the process of creating, transforming, extracting, and selecting variables from data.
- **Model development :** Model development consists of model building, training, tuning and evaluation. Model building includes creating a pipeline that automates the build, train and release to staging and production environments.
- **Deployment :** After a model is trained, tuned, evaluated and validated, we can deploy the model into production. we can then make predictions and inferences against the model.
- **Monitoring :** Model monitoring system ensures your model is maintaining a desired level of performance through early detection and mitigation.

5.2 Guidelines for Machine Learning Experiments

- **Aim of the study :** What are the objectives (e.g. assessing the expected error of an algorithm, comparing two learning algorithm on a particular problem, etc.).
- **Selection of the response variable :** what should we use as the quality measure (e.g. error, precision and recall, complexity, etc.)
- **Choice of factors and levels :** What are the factors for the defined aim of the study (factors are hyperparameters when the algorithm is fix and want to find

best **hyperparameters**, if we are comparing algorithms, the learning algorithm is a factor).

- **Choice of experimental design :** Use factorial design unless we are sure that the factors do not interact. Replication number depends on the dataset size; it can be kept small when the dataset is large. Avoid using small datasets which leads to responses with high variance and the differences will not be significant and results will not be conclusive.
- **Performing the experiment :** Doing a few trial runs for some random settings to check that all is as expected, before doing the factorial experiment. All the results should be reproducible.
- **Statistical analysis of the data :** Conclusion we get should not be due to chance.
- Conclusions and **recommendations** : One frequently conclusion is the need for further experimentation. There is always a risk that our conclusions be wrong, especially if the data is small and noisy. When our expectations are not met, it is most helpful to investigate why they are not.

5.2.1 Dataset Preparation

- Machine learning is about learning some properties of a data set and applying them to new data. This is why a common practice in machine learning to evaluate an algorithm is to split the data at hand in two sets, one that we call a training set on which we learn data properties and one that we call a testing set, on which we test these properties.
- In training data, data are assign the labels. In test data, data labels are unknown but not given. The training data consist of a set of training examples.
- The real aim of supervised learning is to do well on test data that is not known during learning. Choosing the values for the parameters that minimize the loss function on the training data is not necessarily the best policy.
- The training error is the mean error over the training sample. The test error is the expected prediction error over an independent test sample.
- Problem is that training error is not a good estimator for test error. Training error can be reduced by making the hypothesis more sensitive to training data, but this may lead to over fitting and poor generalization.
- Training set : A set of examples used for learning, where the target value is known.
- Test set : It is used only to assess the performances of a classifier. It is never used during the training process so that the error on the test set provides an unbiased estimate of the generalization error.
- Training data is the knowledge about the data source which we use to construct the classifier.

- In a dataset, a training set is implemented to build up a model, while a test (or validation) set is to validate the model built. Data points in the training set are excluded from the test (validation) set. Usually, a dataset is divided into a training set, a validation set (some people use 'test set' instead) in each iteration or divided into a training set, a validation set and a test set in each iteration.
- In machine learning, we basically try to create a model to predict the test data. So, we use the training data to fit the model and testing data to test it. The models generated are to predict the results unknown which is named as the test set.

5.3 Cross Validation (CV) and Resampling

- Validation techniques in machine learning are used to get the error rate of the ML model, which can be considered as close to the true error rate of the population. If the data volume is large enough to be representative of the population, you may not need the validation techniques.
- In machine learning, model validation is referred to as the process where a trained model is evaluated with a testing data set. The testing data set is a separate portion of the same data set from which the training set is derived. The main purpose of using the testing data set is to test the generalization ability of a trained model.
- **Cross-validation** is a technique for evaluating ML models by training several ML models on subsets of the available input data and evaluating them on the complementary subset of the data. Use cross-validation to detect overfitting, ie, failing to generalize a pattern.
- In general, ML involves deriving models from data, with the aim of achieving some kind of desired behavior, e.g., prediction or classification.
- But this generic task is broken down into a number of special cases. When training is done, the data that was removed can be used to test the performance of the learned model on ``new'' data. This is the basic idea for a whole class of model evaluation methods called **cross validation**.
- Types of cross validation methods are holdout, K - fold and leave-one-out.
- The holdout method is the simplest kind of cross validation. The data set is separated into two sets, called the training set and the testing set. The function approximate fits a function using the training set only.
- K - fold cross validation is one way to improve over the holdout method. The data set is divided into k subsets, and the holdout method is repeated k times. Each time, one of the k subsets is used as the test set and the other k-1 subsets are put together to form a training set. Then the average error across all k trials is computed.

- Leave-one-out cross validation is K - fold cross validation taken to its logical extreme, with K equal to N, the number of data points in the set. That means that N separate times, the function approximate is trained on all the data except for one point and a prediction is made for that point.

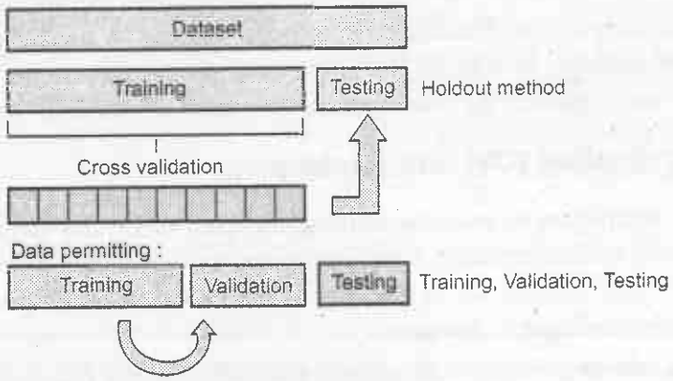


Fig. 5.3.1

5.3.1 K - Fold Cross Validation

- K - fold CV is where a given data set is split into a K number of sections/folds where each fold is used as testing set at some point.
- Lets take the scenario of 5-Fold cross validation (K = 5). Here, the data set is split into 5 folds.
- In the first interaction, the first fold is used to test the model and the rest are used to train the model. In the second iteration, 2nd fold is used as the testing set while the rest serve as the training set. This process is repeated until each fold of the 5 folds has been used as the testing set.
- K - fold cross validation is performed as per the following steps :

1. Partition the original training data set into k equal subsets. Each subset is called a fold. Let the folds be named as $f_1, f_2, \ldots f_k$.
2. For i = 1 to i = k

- Keep the fold f_i as validation set and keep all the remaining k – 1 folds in the cross validation training set.

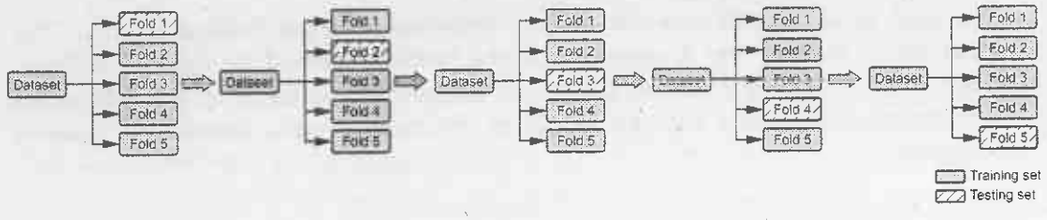


Fig. 5.3.2

3. Estimate the accuracy of your machine learning model by averaging the accuracies derived in all the k cases of cross validation.

- In the k - fold cross validation method, all the entries in the original training data set are used for both training as well as validation. Also, each entry is used for validation just once.
- The advantage of this method is that it matters less how the data gets divided. Every data point gets to be in a test set exactly once and gets to be in a training set $k-1$ times. The variance of the resulting estimating is reduced as k is increased.
- The disadvantage of this method is that the training algorithm has to be rerun scratch k times, which means it takes k times as much computation to make an evaluation. A variant of this method is to randomly divide the data into a test and training set k different times.
- The advantage of doing this is that you can independently choose how large each test set is and how many trials you average over.

5.3.2 Bootstrapping

- Bootstrapping is a method of sample reuse that is much more general than cross-validation. The idea is to use the observed sample to estimate the population distribution. Then samples can be drawn from the estimated population and the sampling distribution of any type of estimator can itself be estimated.
- The bootstrap is a flexible and powerful statistical tool that can be used to quantify the uncertainty associated with a given estimator or statistical learning method. For example, it can provide an estimate of the standard error of a coefficient, or a confidence interval for that coefficient.
- Suppose that we wish to invest a fixed sum of money in two financial assets that yield returns of X and Y respectively, where X and Y are random quantities. We will invest a fraction α of our money in X and will invest the remaining $1 - \alpha$ in Y .
- We wish to choose α to minimize the total risk, or variance, of our investment. In other words, we want to minimize Var $(\alpha X + (1 - \alpha)Y)$.
- One can show that the value that minimizes the risk is given by,

$$\alpha = \frac{\sigma_Y^2 - \sigma_{XY}}{\sigma_X^2 + \sigma_Y^2 - 2\sigma_{XY}},$$

where σ_X^2 = Var(X), σ_Y^2 = Var(Y) and σ_{XY} = Cov(X,Y)

- But the values of σ_X^2, σ_Y^2 and σ_{XY} are unknown.

- We can compute estimates for these quantities, $\hat{\sigma}_X^2$, $\hat{\sigma}_Y^2$ and $\hat{\sigma}_{XY}$, using a data set that contains measurements for X and Y.
- We can then estimate the value of α that minimizes the variance of our investment using,

$$\hat{\alpha} = \frac{\hat{\sigma}_Y^2 - \hat{\sigma}_{XY}}{\hat{\sigma}_X^2 + \hat{\sigma}_Y^2 - 2\hat{\sigma}_{XY}}$$

- To estimate the standard deviation of $\hat{\alpha}$, we repeated the process of simulating 100 paired observations of X and Y and estimating α 1,000 times.
- We thereby obtained 1,000 estimates for α, which we can call $\hat{\alpha}_1, \hat{\alpha}_2, \ldots, \hat{\alpha}_{1000}$.
- For these simulations the parameters were set to $\sigma_X^2 = 1$, $\sigma_Y^2 = 1.25$ and $\sigma_{XY} = 0.5$ and so we know that the true value of α is 0.6.
- The mean over all 1,000 estimates for α is,

$$\hat{\alpha} = \frac{1}{1000}\sum_{r=1}^{1000}\hat{\alpha}_r = 0.5996,$$

very close to $\alpha = 0.6$ and the standard deviation of the estimates is,

$$\sqrt{\frac{1}{1000-1}\sum_{r=1}^{1000}(\hat{\alpha}_r - \hat{\alpha})^2} = 0.083$$

- This gives us a very good idea of the accuracy of $\hat{\alpha}$: $SE(\hat{\alpha}) \approx 0.083$.
- So roughly speaking, for a random sample from the population, we would expect $\hat{\alpha}$ to differ from α by approximately 0.08, on average.
- There are three forms of bootstrapping which differ primarily in how the population is estimated.
  1. Nonparametric (Resampling)
  2. Semiparametric (Adding noise)
  3. Parametric. (Simulation)

1. **Nonparametric bootstrap :** In the nonparametric bootstrap a sample of the same size as the data is taken from the data with replacement. If we measure 10 samples, we create a new sample of size 10 by replicating some of the samples that we have already seen and omitting others.
2. **Semiparametric bootstrap :** The resampling bootstrap can only reproduce the items that were in the original sample. The semiparametric bootstrap assumes that the population includes other items that are similar to the observed sample by sampling from a smoothed version of the sample histogram. It turns out that this can be done very simply by first taking a sample with replacement from the observed sample and then adding noise.

3. **Parametric bootstrap :** Parametric bootstrapping assumes that the data comes from a known distribution with unknown parameters. We estimate the parameters from the data that you have and then you use the estimated distributions to simulate the samples.

5.4 Measuring Classifier Performance

- A binary classification rule is a method that assigns a class to an object, on the basis of its description.
- The performance of a binary classifier can be assessed by tabulating its predictions on a test set with known labels in contingency table or confusion matrix, with actual classes in rows and predicted classes in columns.
- Measures of performance need to satisfy several criteria :
  1. They must coherently capture the aspect of performance of interest;
  2. They must be intutive enough to become widely used, so that the same measures are consistently reported by researches, enabling community-wide conclusions to be drawn;
  3. They must be computationally tractable, to match the rapid growth in scale of modem data collection.
  4. They must be simple to report as a single number for each method-dataset combination.
- Performance metrics for binary classification are designed to captured tradeoffs between four fundamental population quantities : True positives, false positives, true negatives and false negatives.
- The evaluation measures in classification problems are defined from a matrix with the numbers of examples correctly and incorrectly classified for each class, named confusion matrix. The confusion matrix for a binary classification problem is shown below.

True class	Predicted class	
	Positive	Negative
Positive	True positive	False negative
Negative	False positive	True negative

- A confusion matrix contains about actual and predicted classifications done by a classification system. Performance of such systems is commonly using data in the matrix. Confusion matrix is also called a contingency table.
  1. **False positives :** Examples predicted as positive, which are from the negative class.

2. **False negatives :** Examples predicted as negative, whose true class is positive.
3. **True poisitives :** Examples correctly predicted as pertaining to the positive class.
4. **True negatives :** Examples correctly predicted as belongings to the negative class.

- The evaluation measure most used in practice is the accuracy rate. It evaluates the effectiveness of the classifier by its percentage of correct predictions.

$$\text{Accuracy rate} = \frac{|\text{True negatives}| + |\text{True positives}|}{|\text{False negatives}| + \text{False positives}| + |\text{True negatives}| + |\text{True positives}|}$$

- The complement of accuracy rate is the error rate, which evaluates a classifier by its percentage of incorrect predictions.

$$\text{Error rate} = \frac{|\text{False negatives}| + |\text{False positives}|}{|\text{False negatives}| + |\text{False positives}| + |\text{True negatives}| + |\text{True positives}|}$$

Error rate = 1 – (Accuracy rate)

- The recall and specificity measures evaluate the effectiveness of a classifier for each class in the binary problem. The recall is also known as sensitivity or true positive rate. Recall is the proportion of examples belonging to the positive class which were correctly predicted as positive.
- The specificity is a statistical measures of how well a binary classification test correctly identifies the negative cases.

$$\text{Recall (R)} = \frac{|\text{True positive}|}{|\text{True positive}| + |\text{False negative}|}$$

$$\text{Specificity} = \frac{|\text{True negative}|}{|\text{False positive}| + |\text{True positive}|}$$

- True positive Rate (TPR) is also called sensitivity, hit rate and recall.

$$\text{Sensitivity} = \frac{\text{Number of true positives}}{\text{Number of true positive} + \text{Number of false positive}}$$

- A statistical measure of how well a binary classification test correctly identifies a condition. Probability of correctly labeling members of the target class.
- No single measures tells the whole story. A classifier with 90 % accuracy can be useless if 90 percent of the population does not have cancer and the 10 % that do are misclassified by the classifier. Use of multiple measures recommended.

5.4.1 Accuracy and ROC Curves

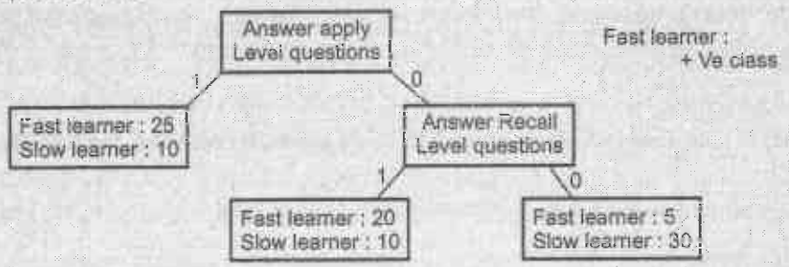
- Binary classification accuracy metrics quantify the two types of correct predictions and two types of errors. Typical metrics are accuracy (ACC), precision, recall, false positive rate, F1-measure. Each metric measures a different aspect of the predicative model.

- Accuracy (ACC) measures the fraction of correct predictions. Precision measures the fraction of actual positives among those examples that are predicted as positive. Recall measures how many actual positives were predicted as positive. F1-measure is the harmonic mean of precision and recall.

ROC Curve

- Receiver Operating Characteristics (ROC) graphs have long been used in signal detection theory to depict the tradeoff between hit rates and false alarm fates over noisy channel. Recent years have seen an increase in the use of ROC graphs in the machine learning community.
- An ROC plot plots true positive rate on the Y-axis false positive rate on the X-axis; a single contingency table corresponds to a single point in an ROC plot.
- The performance of a ranker can be assessed by drawing a piecewise linear curve in an ROC plot, known as an ROC curve. The curve starts in (0, 0), finishes in (1, 1) and is monitorically non-decreasing in both axes.
- A useful technique for organizing classifiers and visualizing their performance. Especially useful for domains with skewed class distribution and unequal classification error costs.
- It allows to create ROC curve and a complete sensitivity/specificity report. The ROC curve is a fundamental tool for diagnostic test evaluation.
- In a ROC curve the true positive rate (Sensitivity) is plotted in function of the false positive rate (100 Specificity) for different cut-off points of a parameter. Each point on the ROC curve represents a sensitivity/specificity pair corresponding to a particular decision threshold. The area under the ROC curve is a measure of how well a parameter can distinguish between two diagnostic groups.
- Each point on an ROC curve connecting two segments corresponds to the true and false positive rates achieved on the same test set by the classifier obtained from the ranker by splitting the ranking between those two segments.
- An ROC curve is convex if the slopes are montonically non-increasing when moving along the curve from (0, 0) to (1, 1). A concavity in an ROC curve, i.e., two or more adjacent segments with increasing slopes, indicates a locally worse than random ranking. In this we would get better ranking performance by joining the segments involved in the concavity, thus creating a coarser classifier.

Example 5.4.1 *i) Find contingency table ii) Find recall iii) Precision iv) Negative recall v) False positive rate*



Solution : Contingency table

	Predicted			Total
Faster Learner	25	20	5	50
Slow Learner	10	10	30	50
Total	35	30	35	100

Actual

$$\text{Precision} = \frac{\text{True Positive}}{\text{Actual Results}} \quad \text{or} \quad \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$

$$\text{Recall} = \frac{\text{True Positive}}{\text{Predicted Results}} \quad \text{or} \quad \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

Calculate precision and recall

Precision = 25/35 = 0.714

Recall = 25/30 = 0.833

False positive rate = (False positive) / (false positive + true negative)

= 10/(10 + 30) = 0.25

Example 5.4.2 *Consider following confusion matrix and calculate following i) Sensitivity of classifier ii) Specificity of classifier.*

Confusion Matrix		Predicted		Total
Actual	+	6	10	18
	−	4	8	12
Total		12	18	30

Solution : Given data : TP = 8, FN = 10, FP = 4, TN = 8

- Sensitivity (SN) is calculated as the number of correct positive predictions divided by the total number of positives. It is also called recall (REC) or true positive rate (TPR)

$$\text{Sensitivity (SN)} = \frac{TP}{TP+FN} = \frac{8}{8+10} = 0.444$$

- Specificity (SP) is calculated as the number of correct negative predictions divided by the total number of negatives. It is also called true negative rate (TNR).

$$\text{Specificity (SP)} = \frac{TN}{TN+FP} = \frac{8}{8+4} = 0.666$$

Example 5.4.3 *Consider the following 3-class confusion matrix. Calculate precision and recall per class. Also calculate weighted average precision and recall for classifier.*

	Predicted		
Actual	15	2	3
	7	15	8
	2	3	45

Solution :

	Predicted			
	15	2	3	20
Actual	7	15	8	30
	2	3	45	50
	24	20	56	100

$$\text{Classifier Accuracy} = \frac{15+15+45}{100} = \frac{75}{100} = 0.75$$

Calculate per-class precision and recall :

$$\text{First class} = \frac{15}{24} = 0.63 \quad \text{and} \quad \frac{15}{20} = 0.75$$

$$\text{Second class} = \frac{15}{20} = 0.75 \quad \text{and} \quad \frac{15}{30} = 0.50$$

$$\text{Third class} = \frac{45}{56} = 0.8 \quad \text{and} \quad \frac{45}{50} = 0.9$$

Example 5.4.4 *Prove that : i) FPR = 1 – TPR ii) FNR = 1 – TPR*

Solution : i) FPR = 1–TPR

False Positive Rate (FPR) = 1 – True Negative Rate (TNR)

$$FPR = FP/N = FP/(FP + TN)$$

$$FPR = 1 - TNR$$

ii) FNR = 1 – TPR

False Negative Rate = 1 – True Positive Rate

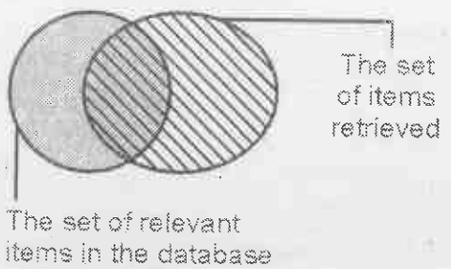
$$FNR = FN/(FN + TP)$$

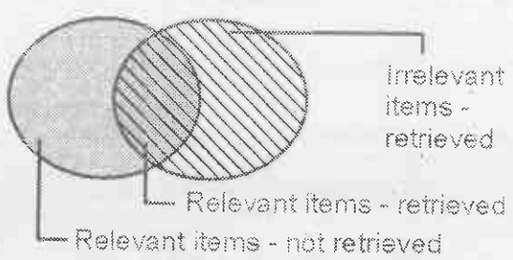
$$FNR = 1 - TPR$$

5.4.2 Precision and Recall

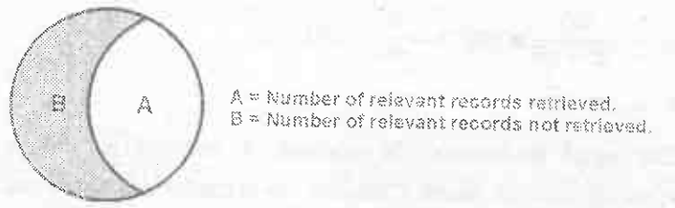
- **Relevance :** Relevance is a subjective notion. Different users may differ about the relevance or non-relevance of particular documents to given questions.
- In response to a query, an IR system searches its document collection and returns a ordered list of responses. It is called the retrieved set or ranked list. The system employs a search strategy or algorithm and measure the quality of a ranked list.
- A better search strategy yields a better ranked list and better ranked lists help the user fill their information need.
- Precision and recall are the basic measures used in evaluating search strategies. As shown in the first two figures, these measures assume :

1. There is a set of records in the database which is relevant to the search topic
2. Records are assumed to be either relevant or irrelevant.
3. The actual retrieval set may not perfectly match the set of relevant records.



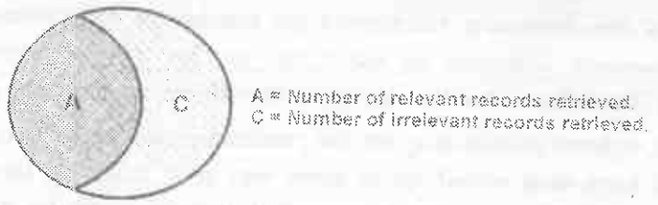


- **Recall** is the ratio of the number of relevant records retrieved to the total number of relevant records in the database. It is usually expressed as a percentage.



$$\text{Recall} = \frac{A}{A+B} \times 100\ \%$$

Precision is the ratio of the number of relevant records retrieved to the total number of irrelevant and relevant records retrieved. It is usually expressed as a percentage.



$$\text{Precison} = \frac{A}{A+C} \times 100\ \%$$

- As recall increases, the precision decreases and recall decreases the precision increases.

Example 5.4.5 *Assume the following :*

A database contains 80 records on a particular topic

A search was conducted on that topic and 60 records were retrieved.

Of the 60 records retrieved, 45 were relevant.

Calculate the precision and recall scores for the search.

Solution : Using the designations above :

A = The number of relevant records retrieved,

B = The number of relevant records not retrieved, and

C = The number of irrelevant records retrieved.

In this example A = 45, B = 35 (80 – 45) and C = 15 (60 – 45).

$$\text{Recall} = \frac{45}{45+35} \times 100\ \%$$

$$\text{Recall} = \frac{45}{80} \times 100\ \%$$

$$\textbf{Recall} = \textbf{56.25 \%}$$

$$\text{Precision} = \frac{A}{A+C} \times 100\ \%$$

$$\text{Precision} = \frac{45}{45+15} \times 100\ \% = \frac{45}{60} \times 100$$

Precision = 75 %

Example 5.4.6 *20 found documents, 18 relevant, 3 relevant documents are not found, 27 irrelevant are as well not found. Calculate the precision and recall and fallout scores for the search.*

Solution : Precision : 18/20 = 90 %

Recall : 18/21 = 85.7 %

Fall-out : 2/29 = 6.9 %

- Recall is a non-decreasing function of the number of docs retrieved. In a good system, precision decreases as either the number of docs retrieved or recall increases. This is not a theorem, but a result with strong empirical confirmation.
- The set of ordered pairs makes up the precision-recall graph. Geometrically when the points have been joined up in some way they make up the precision-recall curve. The performance of each request is usually given by a precision-recall curve. To measure the overall performance of a system, the set of curves, one for each request, is combined in some way to produce an average curve.
- Assume that set R_q containing the relevant document for q has been defined. Without loss of generality, assume further that the set R_q is composed of the following documents :

$$R_q = \{ d_3, d_5, d_9, d_{25}, d_{39}, d_{44}, d_{56}, d_{71}, d_{89}, d_{123} \}$$

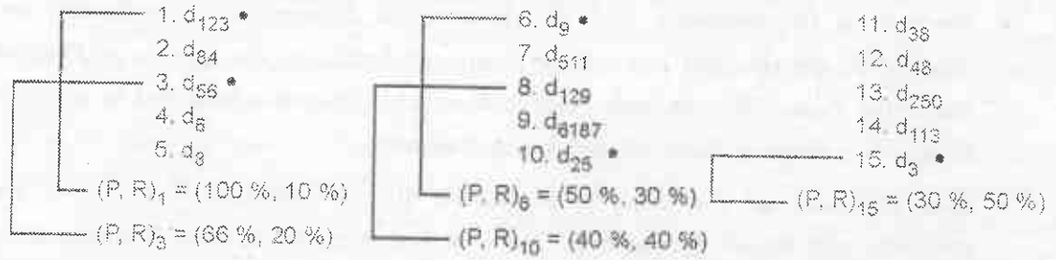
There are ten documents which are relevant to the query q.

- For the query q, a ranking of the documents in the answer set as follows.

Ranking for query q :

1. d_{123} *	6. d_9 *	11. d_{38}
2. d_{84}	7. d_{511}	12. d_{48}
3. d_{56} *	8. d_{129}	13. d_{250}
4. d_6	9. d_{187}	14. d_{113}
5. d_8	10. d_{25} *	15. d_3 *

- The documents that are relevant to the query q are marked with star after the document number. Ten relevant documents, five included in Top 15 .



- Fig 5.4.1 shows the curve of precision versus recall. By taking various numbers of the top returned documents (levels of recall), the evaluator can produce a precision-recall curve.

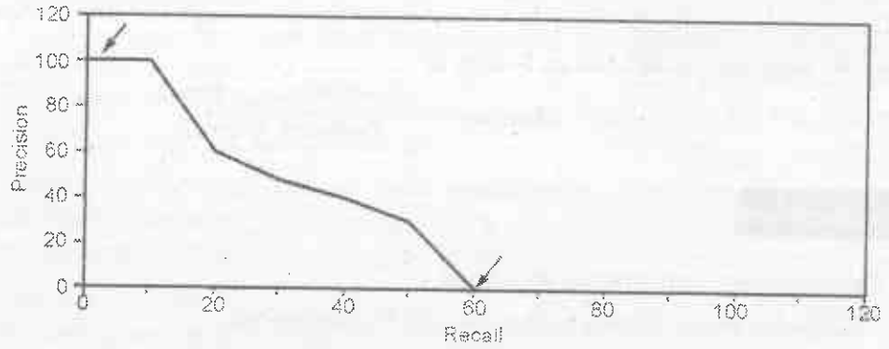


Fig. 5.4.1 Precision versus recall curve

- The precision versus recall curve is usually plotted based on 11 standard recall level: 0 %,10 %,....,100 %.
- In this example : The precisions for recall levels higher than 50 % drop to 0 because no relevant documents were retrieved. There was an interpolation for the recall level 0 %.
- Since the recall levels for each query might be distinct from the 11 standard recall levels.

5.4.3 F - Measure

- The F measure is a measure of a test's accuracy and is defined as the weighted harmonic mean of the precision and recall of the test. The F - measure or F - score is one of the most commonly used "single number" measures in Information Retrieval, Natural Language Processing and Machine Learning.
- F-measure comes from Information Retrieval (IR) where Recall is the frequency with which relevant documents are retrieved or 'recalled' by a system, but it is known elsewhere as Sensitivity or True Positive Rate (TPR).

- Precision is the frequency with which retrieved documents or predictions are relevant or 'correct', and is properly a form of Accuracy, also known as Positive Predictive Value (PPV) or True Positive Accuracy (TPA). F is intended to combine these into a single measure of search 'effectiveness'.
- High precision and low accuracy is possible due to systematic bias. One of the problems with Recall, Precision, F - measure and Accuracy as used in Information Retrieval is that they are easily biased.
- The F-measure balances the precision and recall. The result is a value between 0.0 for the worst F-measure and 1.0 for a perfect F - measure.
- The formula for the standard F1 - score is the harmonic mean of the precision and recall. A perfect model has an F-score of 1.

$$\text{F - Measure} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Review Questions

1. *Define following terms with suitable example :*
   i) Confusion matrix ii) False positive rate iii) True positive rate.
2. *What is a contingency table/matrix ? What is the use of it ?*
3. *Explain true positive, true negative false positives, false negatives and class ratio.*
4. *What is a contingency table ? What does it represent ?*
5. *What is multiple linear regression ? How will it be different from simple linear regression ?*

5.5 Multiclass Classification

- Multiclass classification is a machine learning classification task that consists of more than two classes, or outputs. For example, using a model to identify animal types in images from an encyclopedia is a multiclass classification example because there are many different animal classifications that each image can be classified as. Multiclass classification also requires that a sample only have one class.
- Each training point belongs to one of N different classes. The goal is to construct a function which, given a new data point, will correctly predict the class to which the new point belongs.

- There are many scenarios in which there are multiple categories to which points belong, but a given point can belong to multiple categories. In its most basic form, this problem decomposes trivially into a set of unlinked binary problems, which can be solved naturally using techniques for binary classification.
- Common model for classification is the Support Vector Machine (SVM). An SVM works by projecting the data into a higher dimensional space and separating it into different classes by using a single (or set of) hyperplanes. A single SVM does binary classification and can differentiate between two classes. In order to differentiate between K classes, one can use (K – 1) SVMs. Each one would predict membership in one of the K classes.

5.5.1 Weighted Average

- **Mean Average Precision (MAP)** is also called average precision at seen relevant documents. It determine precision at each point when a new relevant document gets retrieved. Average of the precision value obtained for the top k documents, each time a relevant doc is retrieved.
- Avoids interpolation, use of fixed recall levels. MAP for query collection is arithmetic averaging. Average precision - recall curves are normally used to compare the performance of distinct IR algorithms.
- Use P = 0 for each relevant document that was not retrieved. Determine average for each query, then average over queries :

$$\text{MAP} = \frac{1}{N} \sum_{j=1}^{N} \frac{1}{Q_j} \sum_{i=1}^{Q_j} P(\text{doc}_i)$$

where Q_j = Number of relevant document for query j

N = Number of queries

$P(\text{doc}_i)$ = Precision at i^{th} relevant document

Precision - recall appropriateness :

- Precision and recall have been extensively used to evaluate the retrieval performance of IR algorithms. However, a more careful reflection reveals problems with these two measures :
- First, the proper estimation of maximum recall for a query requires detailed knowledge of all the documents in the collection.
- Second, in many situations the use of a single measure could be more appropriate.
- Third, recall and precision measure the effectiveness over a set of queries processed in batch mode.

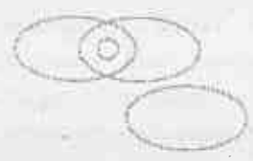
- Fourth, for systems which require a weak ordering though, recall and precision might be inadequate.

Single value summaries :

- Average precision - recall curves constitute standard evaluation metrics for information retrieval systems. However, there are situations in which we would like to evaluate retrieval performance over individual queries. The reasons are two fold :
  1. First, averaging precision over many queries might disguise important anomalies in the retrieval algorithms under study.
  2. Second, we might be interested in investigating whether a algorithm outperforms the other for each query.
- In these situations, a single precision value can be used.

5.5.2 Multiclass Classification Techniques

- Each training point belongs to one of N different classes. The goal is to construct a function which, given a new data point, will correctly predict the class to which the new point belongs. The multi-class classification problem refers to assigning each of the observations into one of k classes.
- A common way to combine pair wise comparisons is by voting. It constructs a rule for discriminating between every pair of classes and then selecting the class with the most winning two-class decisions. Though the voting procedure requires just pair wise decisions, it only predicts a class label.
- Example of multi-label classification is as follows :

1. Is it eatable ? 2. Is it sweet ? 3. Is it a fruit ? 4. Is it a banana ?	1. Is it a banana ? 2. Is it an apple ? 3. Is it an orange ? 4. Is it a pineapple ?	1. Is it a banana ? 2. Is it yellow ? 3. Is it sweet ? 4. Is it round ?
		
Nested/Hierarchical	Exclusive/Multi-class	General/Structured

- Fig. 5.5.1 and 5.5.2 shows binary and multiclass classification.

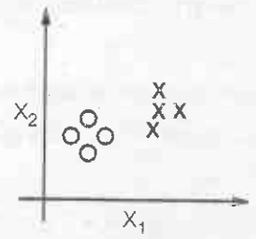


Fig. 5.5.1 Binary classification

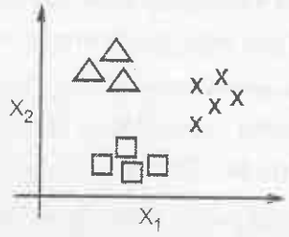


Fig. 5.5.2 Multiclass classification

- Multiclass classification through binary classification :

1. One Vs All (OVA) :

- For each class build a classifier for that class vs the rest. Build N different binary classifiers.
- For this approach, we require N = K binary classifiers, where the k^{th} classifier is trained with positive examples belonging to class k and negative examples belonging to the other K – 1 classes.
- When testing an unknown example, the classifier producing the maximum output is considered the winner, and this class label is assigned to that example.
- It is simple and provides performance that is comparable to other more complicated approaches when the binary classifier is tuned well.

2. All-Vs-All (AVA) :

- For each class build a classifier for those class vs the rest. Build N (N – 1) classifiers, one classifier to distinguish each pair of classes i and j .
- A binary classifier is built to discriminate between each pair of classes, while discarding the rest of the classes.
- When testing a new example, a voting is performed among the classifiers and the class with the maximum number of votes wins.

3. Calibration

- The decision function f of a classifier is said to be calibrated or well-calibrated if P (x is correctly classified | f(x) = s) ≃ s
- Informally f is a good estimate of the probability of classifying correctly a new datapoint x which would have output value x. Intuitively if the "raw" output of a classifier is g you can calibrate it by estimating the probability of x being well classified given that g(x)=y for all y values possible.

4. Error-Correcting Output-Coding (ECOC)

- Error correcting code approaches try to combine binary classifiers in a way that lets you exploit de-correlations and correct errors.
- This approach works by training N binary classifiers to distinguish between the K different classes. Each class is given a codeword of length N according to a binary matrix M . Each row of M corresponds to a certain class.
- The following table shows an example for K = 5 classes and N = 7 bit code words.

	f_1	f_2	f_3	f_4	f_5	f_6	f_7
Class 1	0	0	0	0	0	0	0
Class 2	0	1	1	0	0	1	1
Class 3	0	1	1	1	1	0	0
Class 4	1	0	1	1	0	1	0
Class 5	1	1	0	1	0	0	1

- Each class is given a row of the matrix. Each column is used to train a distinct binary classifier. When testing an unseen example, the output codeword from the N classifiers is compared to the given K code words, and the one with the minimum hamming distance is considered the class label for that example.

Example 5.5.1 *Consider the following three-class confusion matrix.*

		Predicted		
		15	2	3
Actual		7	15	8
		2	3	45

Calculate precision and recall per class. Also calculate weighted average precision and recall for the classifier.

Solution :

		Predicted			
		15	2	3	20
Actual		7	15	8	30
		2	3	45	50
		24	20	56	100

Classifier accuracy $= \frac{15+15+45}{100} = \frac{75}{100} = 0.75$

Calculate per-class precision and recall :

First class $= \frac{15}{24} = 0.63$ and $\frac{15}{20} = 0.75$

Second class $= \frac{15}{20} = 0.75$ and $\frac{15}{30} = 0.50$

Third class $= \frac{45}{56} = 0.8$ and $\frac{45}{50} = 0.9$

Example 5.5.2 *Prove with an example : Accuracy = 1 – Error rate.*

Solution : Accuracy is the percent of correct classifications. Error rate is the percent of incorrect classifications. Classification accuracy is a misleading measure of performance when the data are not perfectly balanced. This is because a classifier may take advantage of an imbalanced dataset and trivially achieve a classification accuracy equal to the fraction of the majority class.

Review Questions

1. *Explain construction of multi-class classifier,*
   i) One Vs all approach ii) One Vs one approach
   iii) Error correcting output codes approach.
2. *Explain any two approaches to construct multiclass classifier.*

5.6 t - Test

- When a small sample (size < 30) is considered, the tests are inapplicable because the assumptions we made for large sample tests, do not hold good for small samples.
- In case of small samples it is not possible to assume,
  i) That the random sampling distribution of a statistics normal
  ii) The sample values are sufficiently close to population values to calculate the S.E. of estimate.
- Thus an entirely new approach is required to deal with problems of small samples. But one should note that the methods and theory of small samples are applicable to large samples but its converse is not true
- When sample sizes are small, as is often the case in practice, the Central Limit Theorem does not apply. One must then impose stricter assumptions on the population to give statistical validity to the test procedure. One common assumption is that the population from which the sample is taken has a normal probability distribution to begin with.
- Degree of freedom (df) : By degree of freedom we mean the number of classes to which the value can be assigned arbitrarily or at will without voicing the restrictions or limitations placed.
- For example, we are asked to choose any 4 numbers whose total is 50. Clearly we are at freedom to choose any 3 numbers say 10, 23, 7 but the fourth number, 10 is fixed since the total is 50 [50 – (10 + 23 + 7) = 10]. Thus we are given a restriction, hence the freedom of selection of number is 4 – 1 = 3.
- The degree of freedom (df) is denoted by ν(nu) or df and it is given by ν = n – k, where n = number of classes and k = number of independent constrains.

5.6.1 t - Test for Single Mean

- When the sample values come from a normal distribution, the exact distribution of "t" was worked out by W. S. Gossett. He called it a **t - distribution.**
- Unfortunately, there is not one t - distribution. There are different t - distributions for each different value of n. If n = 7 there is a certain t - distribution but if n = 13 the t - distribution is a little different. We say that the variable t has a t - distribution with n–1 degrees of freedom.
- Suppose a simple random sample of size n is drawn from a population. If the population from which the sample is taken follows a normal distribution, the distribution of the random variable,

$$t = \frac{\bar{x} - \mu_0}{s / \sqrt{n}}$$

follows **Student's t - Distribution** with n – 1 degrees of freedom.

- The sample mean is $\bar{x}$ and the sample standard deviation is s.
- The **degrees of freedom** are the number of free choices left after a sample statistic such as is calculated. When you use a t - distribution to estimate a population mean, the degrees of freedom are equal to one less than the sample size.

  d.f. = n – 1

Assumptions :

1. Population is normal although this assumption can be relaxed if sample size is "large".
2. Random sample was drawn from the population of interest.

- Based on the comparison of calculated 't' value with the theoretical 't' value from the table, we conclude :

Shape of student's t - distribution

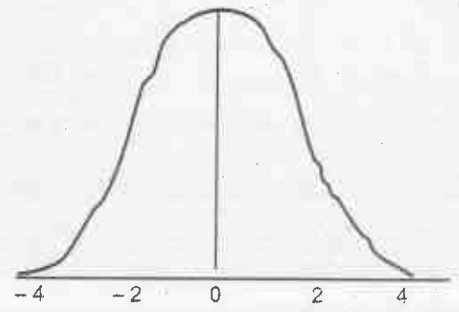


Fig. 5.6.1

5.6.2 Properties of Students t - Distribution

1. The t - distribution is different for different degrees of freedom.
2. The t - distribution is centered at 0 and symmetric about 0.
3. The total area under the curve is 1. The area to the left of 0 is 1/2 and the area to the right of 0 is 1/2.
4. As the magnitude of t increases the graph approaches but never equals 0.
5. The area in the tails of the t - distribution is larger than the area in the tails of the normal distribution.
6. The shape of the t-distribution is dependent on the sample size n.

6. As sample size n increases, the distribution becomes approximately normal.
7. The standard deviation is greater than 1.
8. The mean, median, and mode of the t-distribution are equal to zero.
9. The area in the tails of the t - distribution is a little greater than the area in the tails of the standard normal distribution, because we are using s as an estimate of σ, thereby introducing further variability.
10. As the sample size n increases the density of the curve of t get closer to the standard normal density curve. This result occurs because as the sample size n increases, the values of s get closer to σ, by the law of large numbers.

T - critical values

- Critical values for various degrees of freedom for the t - distribution are (compared to the normal)

n	Degrees of freedom	$t_{0.025}$
6	5	2.571
16	15	2.131
31	30	2.042
101	100	1.984
1001	1000	1.962
Normal	"Infinite"	1.960

5.6.3 t - Test for Correlation Coefficients

- The correlation coefficient, ρ (rho), is a popular statistic for describing the strength of the relationship between two variables.
- The correlation coefficient is the slope of the regression line between two variables when both variables have been standardized by subtracting their means and dividing by their standard deviations. The correlation ranges between plus and minus one.
- When ρ is used as a descriptive statistic, no special distributional assumptions need to be made about the variables (Y and X) from which it is calculated.

- When hypothesis tests are made, you assume that the observations are independent and that the variables are distributed according to the bivariate-normal density function.
- However, as with the t-test, tests based on the correlation coefficient are robust to moderate departures from this normality assumption.
- The population correlation ? is estimated by the sample correlation coefficient r. Note we use the symbol R on the screens and printouts to represent the population correlation.
- t - test for correlation coefficients formul

$$t = r\sqrt{\frac{n-2}{1-r^2}}$$

With degrees of freedom equal to n – 2.

- The steps to be followed for the t - test for correlation coefficient is listed below :

1. State the null hypothesis and alternative hypothesis.

   $H_0 = \rho = 0$

   $H_a = \rho \neq 0$

   Here ρ is the population correlation coefficient.
2. State the significance level.
3. Find the test statistic of correlation coefficient with the above-defined formula.
4. To make a decision, use the critical value approach or the p - value approach
5. Finally, state the conclusion.

- The above test is conducted with the supposition that the association is linear between the variables and originate from a normal distribution that is bivariate.
- The t-test is always used for population correlation coefficient of zero. So, in order to test the population correlation coefficient other than zero, z-test for correlation coefficient is used to test the significance of the correlation coefficient.

5.7 McNemar's Test

- The McNemar test is a non-parametric test for paired nominal data. It is used for finding a change in proportion for the paired data. It compare the performance of two classifiers on N items from a single test set.
- McNemar's test is used to compare the performance of two classifiers on the same test set. This test works if there are a large number of items on which A and B make different predictions.

- McNemar's test is applied to 2 × 2 contingency tables with matched pairs of subjects to determine whether the row and column marginal frequencies are equal.
- The three main assumptions for the test are :
  1. We must have one nominal variable with two categories and one independent variable with two connected groups.
  2. The two groups in the dependent variable must be mutually exclusive.
  3. Sample must be a random sample.

5.8 K - fold CV Paired t Test

- We use k - fold cross-validation to get K training/validation set pairs. To train the two classification algorithms on the training sets T_i; where i = 1; ...; K and test on the validation sets V_i.
- The error percentages of the classifiers on the validation sets are recorded as p_i^1 and p_i^2.
- If the two classification algorithms have the same error rate, then we expect them to have the same mean, or equivalently, that the difference of their means is 0.
- The difference in error rates on fold i is $p_i = p_i^1 - p_i^2$. This is a paired test; that is, for each i, both algorithms see the same training and validation sets.
- When this is done K times, we have a distribution of p_i containing K points. Given that p_i^1 and p_i^2 are both (approximately) normal, their difference p_i is also normal. The null hypothesis is that this distribution has 0 mean
  ($H_0 : \mu = 0$ vs. $H_1 : \mu \neq 0$)
- We define :

$$m = \frac{\sum_{t=1}^{N} p_i}{K}, \quad S^2 = \frac{\sum_{t=1}^{K} (p_i - m)^2}{K-1}$$

  Under the null hypothesis that $\mu = 0$, we have a statistic that is t-distributed with K – 1 degrees of freedom :

$$\frac{\sqrt{k}\,(m-0)}{S} = \frac{\sqrt{k}\,(m)}{S} \sim t_{k-1}$$

- Thus the K - fold cv paired t - test rejects the hypothesis that two classification algorithms have the same error rate at significance level α if this value is outside the interval $(-t_{\alpha/2,K-1}, t_{\alpha/2,K-1})$.
- If we want to test whether the first algorithm has less error than the second, we need a one-sided hypothesis and use a one-tailed test :

  $H_0 : \mu \geq 0$ vs. $H_1 : \mu < 0$
- If the test rejects, out claim that the first one has significantly less error is supported.

- Advantage is that each test set is independent of others. But the training sets still overlap. This overlap may prevent the test from obtaining a good estimate of the amount of variation that would be observed if each training set were completely independent of previous training sets.
- The variance in the t statistic maybe sometimes underestimated, the means are occasionally poorly estimated and this may result in large t values.

5.9 Two Marks Questions with Answers

Q.1 Define Bootstrapping.

Ans. : Bootstrapping is a method of sample reuse that is much more general than cross-validation. The idea is to use the observed sample to estimate the population distribution. Then samples can be drawn from the estimated population and the sampling distribution of any type of estimator can itself be estimated.

Q.2 What is confusion matrix ?

Ans. : The evaluation measures in classification problems are defined from a matrix with the number of examples correctly and incorrectly classified for each class, named confusion matrix.

Q.3 What is cross-validation ?

Ans. : Cross-validation is a technique for validating the model efficiency by training it on the subset of input data and testing on previously unseen subset of the input data.

Q.4 Explain McNemar's test.

Ans. :

- The McNemar test is a non-parametric test for paired nominal data. It is used for finding a change in proportion for the paired data. It compare the performance of two classifiers on N items from a single test set.
- McNemar's test is used to compare the performance of two classifiers on the same test set. This test works if there are a large number of items on which A and B make different predictions.

Q.5 What is K - fold cross-validation ?

Ans. : K-fold cross-validation approach divides the input dataset into K groups of samples of equal sizes. These samples are called **folds**. For each learning set, the prediction function uses k – 1 folds and the rest of the folds are used for the test set.

Q.6 What is a T - test ?

Ans. : The t - test compares the means (averages) of two populations to determine how different they are from each other. The test generates a t-score and p-value, which quantify exactly how different each population is and the likelihood that this difference can be explained by chance or sampling error.

Q.7 List the applications of cross-validation.

Ans. :

- This technique can be used to compare the performance of different predictive modeling methods.
- It has great scope in the medical research field.
- It can also be used for the meta-analysis, as it is already being used by the data scientists in the field of medical statistics.

Q.8 Explain merits and demerits of t-test.

Ans. : Merits :

1. Easy to gather data.
2. Determine source data.
3. Essential for generalization.

Demerits :

1. It may contains small amount of noise.
2. If the data collected violates the assumption of the t - test, then the output is unreliable.
3. T-test cannot be used for multiple comparisons

❑❑❑

SOLVED MODEL QUESTION PAPER

[As Per New Syllabus]

Machine Learning

Semester - IV (AI&DS / CS&BS)

Time : Three Hours] **[Maximum Marks : 100**

Answer ALL Questions

PART A - (10 × 2 = 20 Marks)

Q.1 *What is decision tree ?* **[Refer Two Marks Q.7 Chapter - 1]**

Q.2 *What is inductive learning ?* **[Refer Two Marks Q.14 Chapter - 1]**

Q.3 *Define supervised learning?* **[Refer Two Marks Q.16 Chapter - 2]**

Q.4 *What is random forest?* **[Refer Two Marks Q.7 Chapter - 2]**

Q.5 *List the properties of k-Means algorithm.* **[Refer Two Marks Q.11 Chapter - 3]**

Q.6 *Explain clustering* **[Refer Two Marks Q.5 Chapter - 3]**

Q.7 *Explain back propagation.* **[Refer Two Marks Q.4 Chapter - 4]**

Q.8 *What is batch normalization?* **[Refer Two Marks Q.9 Chapter - 4]**

Q.9 *Explain merits and demerits of t-test.* **[Refer Two Marks Q.8 Chapter - 5]**

Q.10 *What is confusion matrix?* **[Refer Two Marks Q.2 Chapter - 5]**

PART B - (5 × 13 = 65 Marks)

Q.11 a) i) *What is supervised learning? Explain difference between supervised and unsupervised learning.* **[Refer sections 1.3.1 and 1.3.3]** [7]

ii) *Discuss eigen values and eigen vectors.* **[Refer section 1.1.4]** [6]

OR

b) i) *What is hypothesis spaces ? Discuss about population evolution and the schema theorem.* **[Refer section 1.7]** [7]

ii) *What s machine learning? Why machine learning is important ?* **[Refer section 1.2]** [6]

Q.12 a) i) *What is support vector machine? Explain key properties of SVM. Compare SVM with neural network.* **[Refer section 2.4]** [7]

ii) *What is random forest? Explain working of random forest.* **[Refer section 2.6]** [6]

OR

b) i) *Explain in detail logistic regression.* **[Refer section 2.2.2]** [7]

ii) *What is regression? Explain difference between simple regression and multiple regression.* **[Refer section 2.1]** [6]

Q.13 a) *What is clustering? Discuss working of K-means clustering. Explain difference between K-means and kNN* **[Refer sections 3.3 and 3.4.3]** [13]

OR

b) i) *Write short note on Gaussian mixture models and expectation maximization.* **[Refer section 3.5]** [7]

ii) *Explain concept of ensemble learning. Discuss the bagging and boosting.* **[Refer section 3.2]** [6]

Q.14 a) i) *What is activation functions ? Explain in detail identity or linear activation function and sigmoid.* **[Refer section 4.2]** [7]

ii) *What is perceptron? Explain single layer perceptron.* **[Refer section 4.1]** [6]

OR

b) i) *Define Keras and TensorFlow. What is difference between Deep Network and Shallow Network?* **[Refer section 4.6]** [7]

ii) *What is use of regularization in machine learning ? Explain difference between L1 and L2 regularization.* **[Refer section 4.11]** [6]

Q.15 a) i) *Explain multiclass classification techniques.* **[Refer section 5.5.2]** [7]

ii) *Write short note on precision and recall.* **[Refer section 5.4.2]** [6]

OR

b) i) *Explain McNemar's test.* **[Refer section 5.7]** [7]

ii) *Discuss machine learning life cycle.* **[Refer section 5.1]** [6]

PART C - (1 × 15 = 15 Marks)

Q.16 a) *Discuss k-NN clustering algorithm? Why do we need k-NN? Explain working with example.* **[Refer section 3.4]** [15]

OR

b) i) *Consider the following 3-class confusion matrix. Calculate precision and recall per class. Also calculate weighted average precision and recall for classifier.* **[Refer example 5.4.3]** [7]

		Predicted		
Actual		15	2	3
		7	15	8
		2	3	45

ii) *Find all eigen values for*

$$A = \begin{bmatrix} 5 & -2 & 6 & -1 \\ 0 & 3 & -8 & 0 \\ 0 & 0 & 5 & 4 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$ **[Refer example 1.1.5]** [8]

❑❑❑

Notes