UNIT - IV

4 Naïve Bayes and Support Vector Machine

4.1 : Bayes Theorem

Q.1 What is Bayes theorem ? How to select Hypotheses ?

Ans. : • In machine learning, we try to determine the best hypothesis from some hypothesis space H, given the observed training data D.

- In Bayesian learning, the best hypothesis means the most probable hypothesis, given the data D plus any initial knowledge about the prior probabilities of the various hypotheses in H.
- Bayes theorem provides a way to calculate the probability of a hypothesis based on its prior probability, the probabilities of observing various data given the hypothesis, and the observed data itself.
- 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.
- This 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.
- If A and B are two random variables

$$P(A/B) = \frac{P(B/A)P(A)}{P(B)}$$

- In the context of classifier hypothesis h and training data I.

$$p(h/I) = \frac{P(I/h)P(h)}{P(I)}$$

Where (h) = Prior probability of hypothesis h

(I) = Prior probability of training data I

(h|I) = Probability of h given I

P (I|h) = Probability of I given h

Choosing the Hypotheses

- Given the training data, we are interested in the most probable hypothesis. The learner considers some set of candidate hypotheses H and it is interested in finding the most probable hypothesis $h \in H$ given the observed data D.
- Any such maximally probable hypothesis is called a maximum a posteriori (MAP) hypothesis h_{MAP}
- Maximum a posteriori hypothesis (h_{MAP}).

$$h_{MAP} = \underset{h \in H}{\text{argmax}}\, P(h/I)$$

$$= \underset{h \in H}{\text{argmax}} \frac{P(I/h)P(h)}{P(I)}$$

$$= \underset{h \in H}{\text{argmax}}\, P(I/h)P(h)$$

- If every hypothesis is equally probable, $P(h_i) = P(h_j)$ for all h_i and h_j in H.
- P(I/h) is often called the likelihood of the data I given h. Any hypothesis that maximizes P(I/h) is called a maximum likelihood (ML) hypothesis, h_{ML}.

$$h_{ML} = \underset{h \in H}{\text{argmax}}\ P(I/h)$$

Q.2 What is Naïve Bayes ? Classifiers ?

Ans. : • 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.

Q.3 At a certain university, 4% of men are over 6 feet tall and 1% of women are over 6 feet tall. The total student population is divided in the ratio 3:2 in favour of women. If a student is selected at random from among all those over six feet tall, what is the probability that the student is a woman ?

Ans. : Let us assume following :

M = {Student is Male},

F = {Student is Female},

T = {Student is over 6 feet tall}.

Given data : $P(M) = 2/5$,

$P(F) = 3/5$,

$P(T|M) = 4/100$

$P(T|F) = 1/100$.

We require to find $P(F|T)$?

Using Bayes' Theorem we have :

$$P(F/T) = \frac{P(T/F)\,P(F)}{P(T/F)\,P(F) + P(T/M)\,P(M)}$$

$$= \frac{\frac{1}{100} \times \frac{3}{5}}{\frac{1}{100} \times \frac{3}{5} + \frac{4}{100} \times \frac{2}{5}} = \frac{\frac{3}{500}}{\frac{3}{500} + \frac{8}{500}}$$

$$P(F/T) = \frac{3}{11}$$

Q.4 Bag contains 5 red balls and 2 white balls. Two balls are drawn successivly without replacement. Draw the probability tree for this.

Sol. : Let R_1 = for the event of getting a red ball on the first draw, W_2 for getting a white ball on the second draw, and so forth. Here's the probability tree.

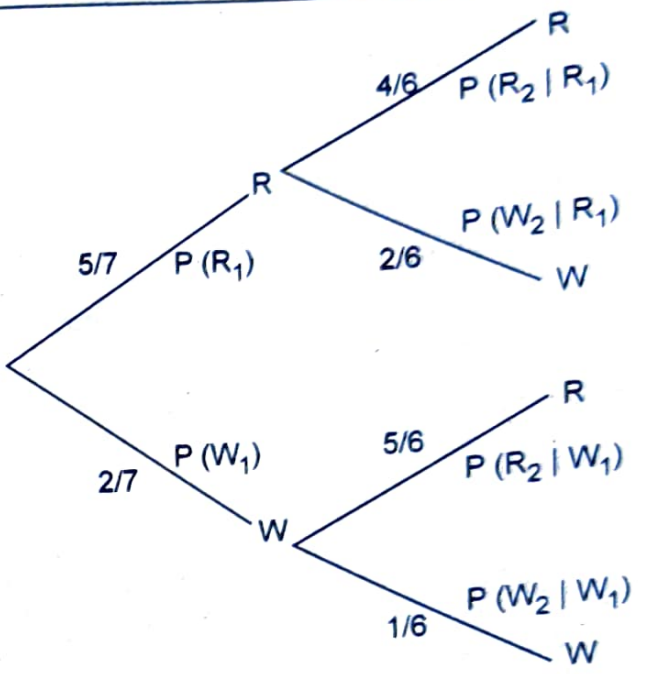


Fig. Q.4.1

4.2 : Naïve Bayes in Scikit-Learn

Q.5 What is Bernoulli naive Bayes ? Explain mean and variance.

Ans. : • The Binomial distribution gives the general form of the probability distribution for the random variable r, whether it represents the number of heads in n coin tosses or the number of hypothesis errors in a sample of n examples.

The detailed form of the Binomial distribution depends on the specific sample size n and the specific probability p or $error_D(h)$.

• Binomial distribution applies as follows :

1. There is a base, or underlying, experiment whose outcome can be described by a random variable (Y). The random variable can take on two possible values.
2. The probability that Y = 1 on any single trial of the underlying experiment is given by some constant p, independent of the outcome of any other experiment.

3. A series of n independent trials of the underlying experiment is performed, producing the sequence of independent, identically distributed random variables $Y_1, Y_2, \ldots, Y_n$.

Mean and Variance

- Two properties of a random variable that are often of interest are its expected value (also called its mean value) and its variance. The expected value is the average of the values taken on by repeatedly sampling the random variable

$$\text{Mean } (\mu) = \sum_{i=0}^{n} x^{n} C_r p^x q^{n-x}$$

$$= nC_1 pq^{n-1} + 2nC_2 p^2 q^{n-2} + \ldots n\, nC_n\, p^n$$

$$= {}^{n}C_1 pq^{n-1} + \frac{2n(n-1)}{1\times 2} p^2 q^{n-2} + \ldots + \frac{n(n-1)}{1\times 2\times 3\times \ldots n} p^n$$

$$= np\left[q^{n-1} + (n-1)\, pq^{n-2} + \frac{(n-1)(n-2)}{2!} p^2 q^{n-3} + \ldots + p^{n-1}\right]$$

$$= np\,(q+p)^{n-1}$$

Using binomial theorem $(p + q = 1)$.

Therefore $\mu = np\,(1)$

$$\mu = np$$

Variance (σ^2) V (X) :

$$V(X) = E(X^2) - [E(X)]^2 = \sum_{x=0}^{n} x^2 p(x) - \mu^2$$

$$= \sum_{x=0}^{n} {}^{n}C_x p^x q^{n-x} x^2 - \mu^2$$

$$= 1\times 2\, {}^{n}C_2\, p^2 q^{n-2} + 3\times 2\, p^3 q^{n-3} + \ldots + {}^{n}C_n\, n(n-1)\, np$$

$$+ \sum {}^{n}C_x\, x\, p^x q^{n-x} \mu^2$$

$$= n(n-1)\, p^2 \sum_{x=2}^{n} {}^{n-2}C_{x-2} p^{x-2} q^{n-x} + np - n^2 p^2$$

$$= n(n-1)\, p^2\, (p+q)^{n-2} + np - n^2 p^2$$

$$= n(n-1)p^2 + np - n^2p^2$$

$$= n^2p^2 - np^2 + np - n^2p^2$$

$$= np - np^2 = np(1-p)$$

$$\sigma^2 = npq \qquad (q = 1 - p)$$

- The standard deviation (σ) of the binomial distribution is $\sqrt{npq}$.

Q.6 Consider the example of the Binomial distribution

X	0	1	2	3	4	5
P (X = x)	0.004	0.041	0.165	0.329	0.329	0.132

Calculate the mean value of distribution.

Ans. : $\mu = xP(X=0) + xP(X=1) + xP(X=2) + xP(X=3)$

$+ xP(X=4) + xP(X=5)$

$= 0\times(0.004) + 1\times(0.0041) + 2\times(0.165) + 3\times(0.329)$

$+ 4\times(0.329) + 5\times(0.132)$

$= 0 + 0.0041 + 0.33 + 0.987 + 1.316 + 0.66 = 3.2971$

4.3 : Support Vector Machine (SVM)

Q.7 Explain Support Vector Machine ? What is two class problems ?

Ans. : • 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.

- 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 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.

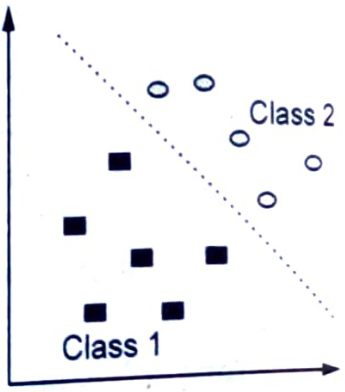


Fig. Q.7.1 Two class problem

Q.8 What is empirical risk minimization ?

Ans. : • SVM are primarily two-class classifiers with the distinct characteristic that they aim to find the optimal hyper-plane such that the expected generalization error is minimized.

- Fig. Q.8.1 shows empirical risk.

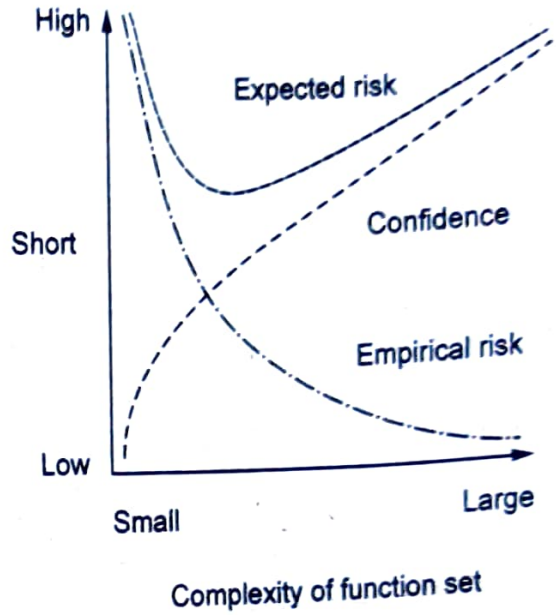


Fig. Q.8.1 Empirical risk

- 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.**

Q.9 Compare SVM and NN.

Ans. :

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 not efficient.	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

Q.10 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 ?

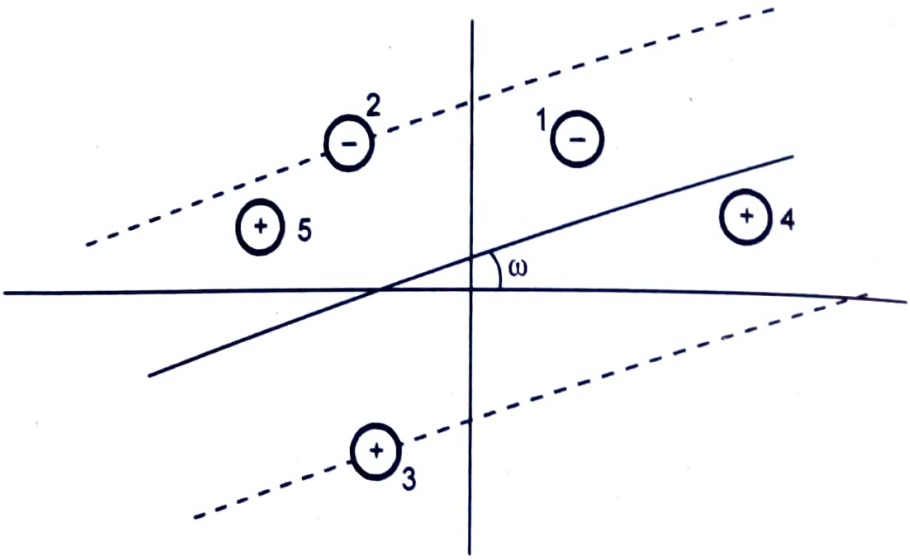


Fig. Q.10.1

Ans. : • Data points 1 and 5 will have maximum penalty.

- Margin (m) is the gap between data points and 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.

Q.11 Explain key properties of SVM.

Ans. : 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.

Q.12 Explain soft margin SVM.

Ans. : • 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.

Q.13 List application and benefits of SVM.

Ans. : 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

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.

4.4 : Kernel Based Classification

Q.14 Explain kernel methods for non-linearity.

Ans. : • Kernel methods refer to a family of widely used nonlinear algorithms for machine learning tasks like classification, regression, and feature extraction.

• Any non-linear problem (classification, regression) in the original input space can be converted into linear by making non-linear mapping ϕ into a feature space with higher dimension and shown in Fig. Q.14.1

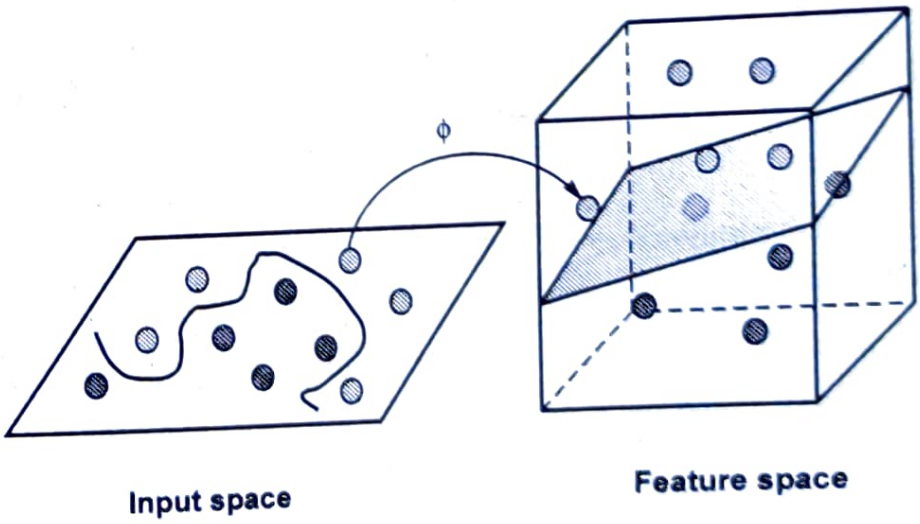


Fig. Q.14.1 : Mapping of space

- Often we want to capture nonlinear patterns in the data.
  a. Nonlinear Regression : Input-output relationship may not be linear
  b. Nonlinear Classification : Classes may not be separable by a linear boundary
- Kernels : Make linear models work in nonlinear settings.
- Kernels, using a feature mapping ϕ, map data to a new space where the original learning problem becomes easy.
- Consider two data points $x = \{x_1; x_2\}$ and $z = \{z_1; z_2\}$. Suppose we have a function k which takes as inputs x and z and computes.

$$\begin{aligned} k(x, z) &= (x^T z)^2 = (x_1 z_1 + x_2 z_2)^2 \\ &= x_1^2 z_1^2 + x_2^2 z_2^2 + 2 x_1 x_2 z_1 z_2 \\ &= (x_1^2, \sqrt{2}\, x_1 x_2, x_2^2)^T (z_1^2, \sqrt{2}\, z_1 z_2 z_2^2) \\ &= \phi(x)^T \phi(z) \quad \text{(an inner product)} \end{aligned}$$

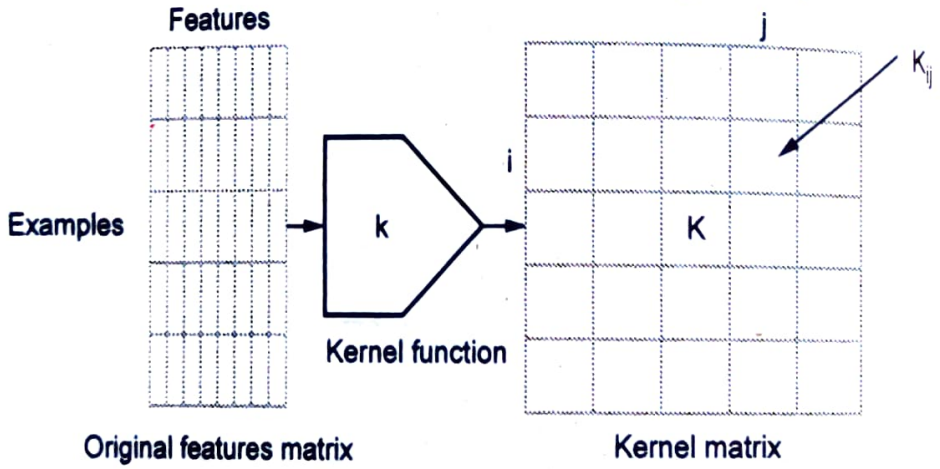


Fig. Q.14.2

- The above k implicitly defines a mapping ϕ to a higher dimensional space

$$\phi(x) = \{x_1^2, \sqrt{2}\, x_1 x_2 x_2^2\}$$

- We didn't need to pre-define/compute the mapping ϕ to compute k(x, z).

- The function k is known as the kernel function.
- K : N × N matrix of pairwise similarities between examples in F space.

Advantages

1. The kernel defines a similarity measure between two data points and thus allows one to incorporate prior knowledge of the problem domain.
2. Most importantly, the kernel contains all of the information about the relative positions of the inputs in the feature space and the actual learning algorithm is based only on the kernel function and can thus be carried out without explicit use of the feature space.
3. The number of operations required is not necessarily proportional to the number of features.

Q.15 Explain Controlled Support Vector Machines.

Ans. : • When large real datasets is used with support vector machine, it can extract a very large number of support vectors to increase accuracy and that can slow down the whole process.

- To allow finding out a trade-off between precision and number of support vectors, Scikit-learn provides an implementation called NuSVC, where the parameter nu (bounded between 0 and 1) can be used to control at the same time the number of support vectors and training errors.
- NuSVC is defined as

```
class sklearn.svm.NuSVC(nu=0.5, kernel='rbf', degree=3,
gamma=0.0, coef0=0.0, shrinking=True, probability=False,
tol=0.001, cache_size=200)
```

- Similar to SVC but uses a parameter to control the number of support vectors. The implementation is based on libsvm.

Parameters :

1. nu : float, optional (default=0.5)

- An upper bound on the fraction of training errors and a lower bound of the fraction of support vectors. Should be in the interval (0, 1].

2. kernel : string, optional (default='rbf')

- Specifies the kernel type to be used in the algorithm. One of 'linear', 'poly', 'rbf', 'sigmoid', 'precomputed'. If none is given 'rbf' will be used.

3. degree : int, optional (default=3)

- degree of kernel function is significant only in poly, rbf, sigmoid

4. gamma : float, optional (default=0.0)

- Kernel coefficient for rbf and poly, if gamma is 0.0 then 1/n_features will be taken.

5. coef0 : float, optional (default=0.0)

- Independent term in kernel function. It is only significant in poly/sigmoid.

6. probability : boolean, optional (default=False) :

- Whether to enable probability estimates. This must be enabled prior to calling predict_proba.

7. shrinking : boolean, optional (default=True) :

END...

UNIT - V

5 Decision Trees and Ensemble Learning

5.1 : Decision Trees

Q.1 What is a decision tree ? Explain

Ans. : Decision Trees

- A decision tree is a simple representation for classifying examples. Decision tree learning is one of the most successful techniques for supervised classification learning.
- A decision tree or a classification tree is a tree in which each internal (non-leaf) 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. Each leaf of the tree is labeled with a class or a probability distribution over the classes.
- A decision tree is a simple representation for classifying examples. 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. Each leaf of the tree is labeled with a class or a probability distribution over the classes.
- In this method a set of training examples is broken down into smaller and smaller subsets while at the same time an associated decision tree get incrementally developed. At the end of the learning process, a decision tree covering the training set is returned.
- The key idea is to use a decision tree to partition the data space into cluster (or dense) regions and empty (or sparse) regions.

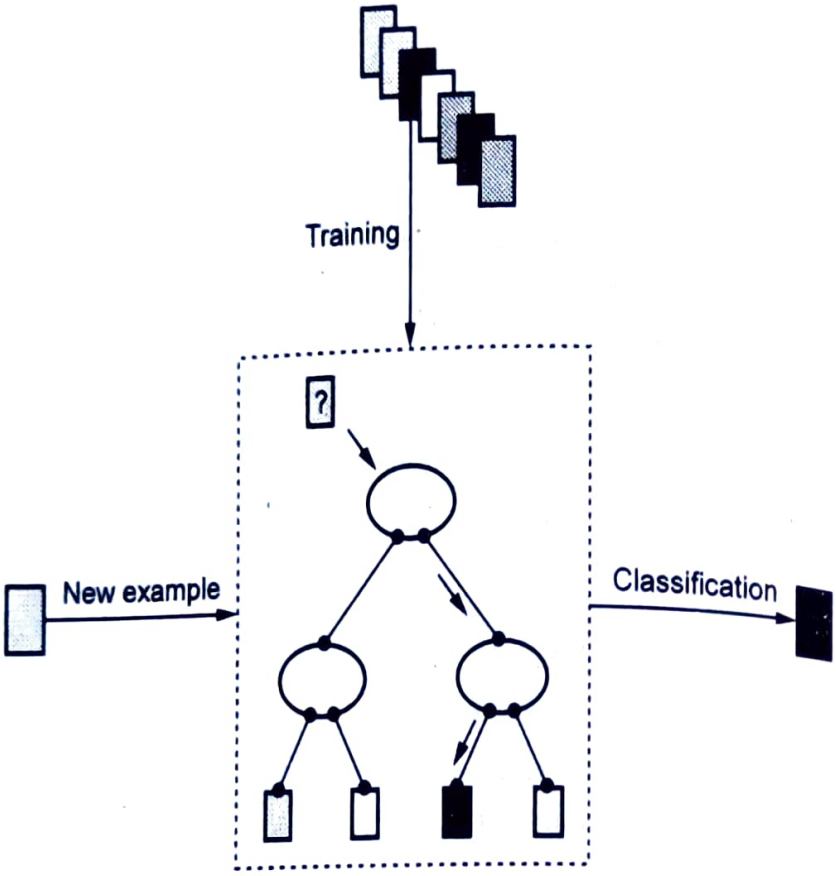


Fig. Q.1.1

- Decision tree consists of
  1. Nodes : test for the value of a certain attribute
  2. Edges : correspond to the outcome of a test and connect to the next node or leaf
  3. Leaves : terminal nodes that predict the outcome (Refer Fig. Q.1.1)
- In Decision Tree Learning, a new example is classified by submitting it to a series of tests that determine the class label of the example. These tests are organized in a hierarchical structure called a decision tree.

Q.2 Explain decision tree algorithm.

Ans. : • 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(Dj , attribute list) to node N;
13. End of for loop
14. return N;

Q.3 Explain ranking and probability estimation trees.

Ans. : • Decision trees are one of the most effective and widely used classification methods. Many applications require class probability estimates and Probability Estimation Trees (PET) have the same attractive features as classification trees. But decision trees have been found to provide poor probability estimates.

- A tree is defined as a set of logical conditions on attributes ; a leaf represents the subset of instances corresponding to the conjunction of conditions along its branch or path back to the root. A simple approach to ranking is to estimate the probability of an instance's membership in a class and assign that probability as the instance's rank. A decision tree can easily be used to estimate these probabilities.
- Rule learning is known for its descriptive and therefore comprehensible classification models which also yield good class predictions. In some application areas, we also need good class probability estimates.
- For different classification models, such as decision trees, a variety of techniques for obtaining good probability estimates have been proposed and evaluated.
- In classification rule mining one search for a set of rules that describes the data as accurately as possible. As there are many different generation approaches and types of generated classification rules.
- A probabilistic rule is an extension of a classification rule, which does not only predict a single class value, but a set of class probabilities, which form a probability distribution over the classes. This probability distribution estimates all probabilities that a covered instance belongs to any of the class in the data set, so we get one class probability per class.
- Error rate does not consider the probability of the prediction, so it is consider in PET. Instead of predicting a class, the leaves give a probability. It is very useful when we do not want just the class, but examples most likely to belong to a class. No additional effort in learning PET compared to decision tree.
- Building decision trees with accurate probability estimates, called probability estimation trees. A small tree has a small number of leaves, thus more examples will have the same class probability.

That prevents the learning algorithm from building an accurate PET.

- On the other hand, if the tree is large, not only may the tree overfit the training data, but the number of examples in each leaf is also small and thus the probability estimates would not be accurate and reliable. Such a contradiction does exist in traditional decision trees.
- Decision trees acting as probability estimators, however, are often observed to produce bad probability estimates. There are two types of probability estimation trees - a single tree estimator and an ensemble of multiple trees.
- Applying a learned PET involves minimal computational effort, which makes the tree-based approach particularly suited for a fast reranking of large candidate sets.
- For simplicity, all attributes are assumed to be numeric. For n attributes, each input datum is then given by an n-tuple $X = (x_1, \ldots, x_n) \in R^n$
- Let $X = \{x^{(1)}, \ldots, x^{(R)}\} \subset R^n$ be the set of training items.
- A probability estimation tree is introduced as a binary tree T with $s \geq 0$ inner nodes $D^T = \{d_1, d_2, \ldots, d_s\}$ and leaf nodes $E^T = \{e_0, e_1, \ldots, e_s\}$ with $E^T \cap D^T = \emptyset$ Each inner node $d_i, i \in \{1, 2, \ldots, s\}$ is labeled by an attribute $\alpha_i^T \in \{1, \ldots, n\}$, while each leaf node $e_j, j \in \{1, 2, \ldots, s\}$ is labeled by a probability $p_j^T \in [0, 1]$.
- The arcs in A^T correspond to conditions on the inputs. Since it is a binary tree and every inner node has exactly two children. By splitting inputs at each decision node until a leaf is reached, the PET partitions the input space into n-dimensional cartesian blocks :

$$H_j^T = \prod_{\alpha=1}^{n} h(l_{j,\alpha}^T, u_{j,\alpha}^T)$$

Q.4 List the advantages and disadvantages of decision tree.

Ans. : Advantages and Disadvantages of Decision Trees

Advantages :

1. Decision trees can handle both nominal and numeric input attributes.
2. Decision tree representation is rich enough to represent any discrete value classifier.
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. It is self-explanatory and when compacted they are also easy to follow.

Disadvantages

1. Most of the algorithms require that the target at-tribute will have only discrete values.
2. Most decision-tree algorithms only examine a single field at a time
3. Decision trees are prone to errors in classification problems with many class.
4. As decision trees use the "divide and conquer" method, they tend to per-form well if a few highly relevant attributes exist, but less so if many complex interactions are present

Q.5 Consider the following six training examples, where each example has three attributes : color, shape and size. Color has three possible values : red, green and blue. Shape has two possible values : square and round. Size has two possible values : big and small.

Example	Color	Shape	Size	Class
1	red	square	big	+
2	blue	square	big	+

3	red	round	small	–
4	green	square	small	–
5	red	round	big	+
6	green	square	big	–

Which is best attribute for the root node of decision tree ?

Ans. :

$H(class) = H(3/6, 3/6) = 1$

$H(class \mid color) = 3/6 * H(2/3, 1/3) + 1/6\ H(1/1, 0/1) + 2/6\ H(0/2, 2/2)$

1 of 6 is blue

2 of 6 are green

1 of the red is negative

2 of the red are positive

3 out of 6 are red

$= 1/2 * (-2/3 \log_2 2/3 - 1/3 \log_2 1/3)$

$+ 1/6 * (-1 \log_2 1 - 0 \log_2 0) + 2/6 * (- 0 \log_2 0 - 1 \log_2 1)$

$= 1/2 * (-2/3(\log_2 2 - \log_2 3) - 1/3(\log_2 1 - \log_2 3))$

$+ 1/6 * 0 + 2/6 * 0$

$= 1/2 * (-2/3(1 - 1.58) - 1/3(0 - 1.58)) = 1/2 * 0.914 = 0.457$

$I(class; color) = H(class) - H(class \mid color)$

$= 1.0 - 0.457 = 0.543$

$H(class \mid shape) = 4/6\ I(2/4, 2/4) + 2/6\ I(1/2, 1/2)$

$$= 4/6 * 1.0 + 2/6 * 1.0 = 1.0$$

$$I(\text{class; shape}) = H(\text{class}) - H(\text{class} \mid \text{shape})$$

$$= 1.0 - 1.0 = 0.0$$

$$H(\text{class} \mid \text{size}) = 4/6\ I(3/4, 1/4) + 2/6\ I(0/2, 2/2) = 0.541$$

$$I(\text{class; size}) = H(\text{class}) - H(\text{class} \mid \text{size})$$

$$= 1.0 - 0.541 = 0.459$$

Max(0.543, 0.0, 0.459) = 0.543, so color is best. Make the root node's attribute color and partition the examples for the resulting children nodes as shown :

The children associated with values green and blue are uniform, containing only – and + examples, respectively. So make these children leaves with classifications – and +, respectively.

Q.6 Explain impurity measures in decision tree.

Ans. : • The node impurity makes reference to the mixture of classes in the training examples covered by the node. When all the examples in a given node belong to the same problem class, then the node is said to be pure. The more equal proportions of classes there are in a node, the more impure the node is.

Let $|D|$ = Total number of data entries.

$|D_i|$ = Number of data entries classified as i

$p_i = \frac{|D_i|}{|D|}$ = Ratio of instance classified as i.

- **Impurity measure** defines how well e classes are separated. In general the impurity measure should satisfy : Largest when data are split evenly for attribute values

$$p_i = \frac{1}{\text{Number of classes}}$$

Should be 0 when all data belong to the same class.

- Two methods are used for measuring impurity : Gini Index and Entropy based measure
- Entropy based measure

$$I(D) = \text{Entropy } (D) = -\sum_{i=1}^{k} p_i \log p_i$$

Example for k = 2

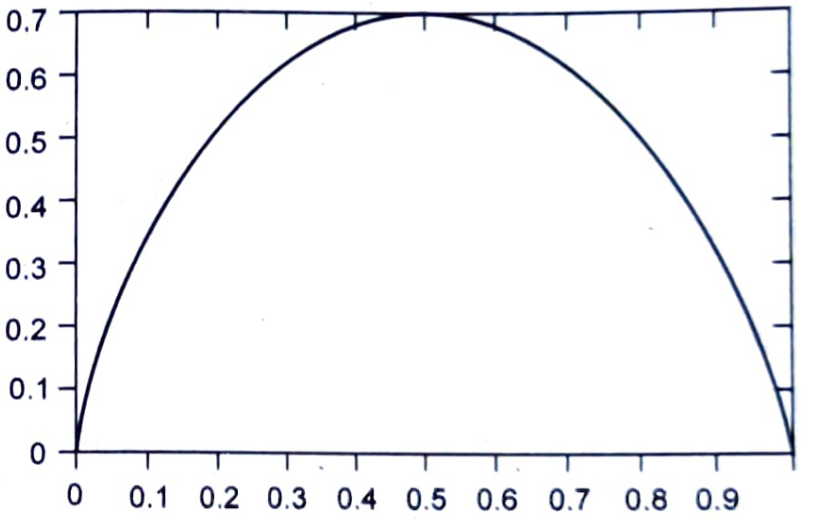


Fig. Q.6.1

- Gini Index measure :

$$I(D) = \text{Gini } (D) = 1 - \sum_{i=1}^{k} p_i^2$$

- Here, the sum is always extended to all classes. This is a very common measure and it is used as a default value by scikit-learn.
- Given a sample, the Gini impurity measures the probability of a misclassification if a label is randomly chosen using the probability distribution of the branch.

Q.7 Explain decision tree classification with Scikit-learn.

Ans. : Decision Tree Classification with Scikit-Learn

- scikit-learn contains the *"DecisionTreeClassifier"* class, which can train a binary decision tree with Gini and cross-entropy impurity measures.
- *DecisionTreeClassifier* accepts (as most learning methods) several hyperparameters that control its behavior. In this case, we used the Information Gain (IG) criterion for splitting learning data, told the method to build a tree of at most three levels, and to accept a node as a leaf if it includes at least five training instances.
- let's consider a dataset with three features and three classes :

  from sklearn.datasets import make_classification
- Entropy is a measure of disorder in a set, if we have zero entropy, it means all values are the same, while it reaches its maximum when there is an equal number of instances of each class.
- At each node, we have a certain number of instances and we measure its entropy.
- Let's consider a classification with default Gini impurity :

  from sklearn.tree import DecisionTreeClassifier

  from sklearn.model_selection import cross_val_score
- To export a trained tree, it is necessary to use the built-in function export_graphviz():

  from sklearn.tree import export_graphviz

5.2 : Ensemble Learning

Q.8 Write a note on : Ensemble Learning.

Ans. : • 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.
- Ensemble 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.
- There are two approaches for combining models : **voting and stacking.**
- In **voting**, no learning takes place at the meta level when combining classifiers by a voting scheme. Label that is most often assigned to a particular instance is chosen as the correct prediction when using voting.
- **Stacking** is concerned with combining multiple classifiers generated by different learning algorithms $L_1, ..., L_N$ 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 $C_1, ..., C_N$ Where $C_i = L_i(S)$

2. Train a meta-level classifier to combine the outputs of the base-level classifiers

- Fig. Q.8.1 shows stacking frame.

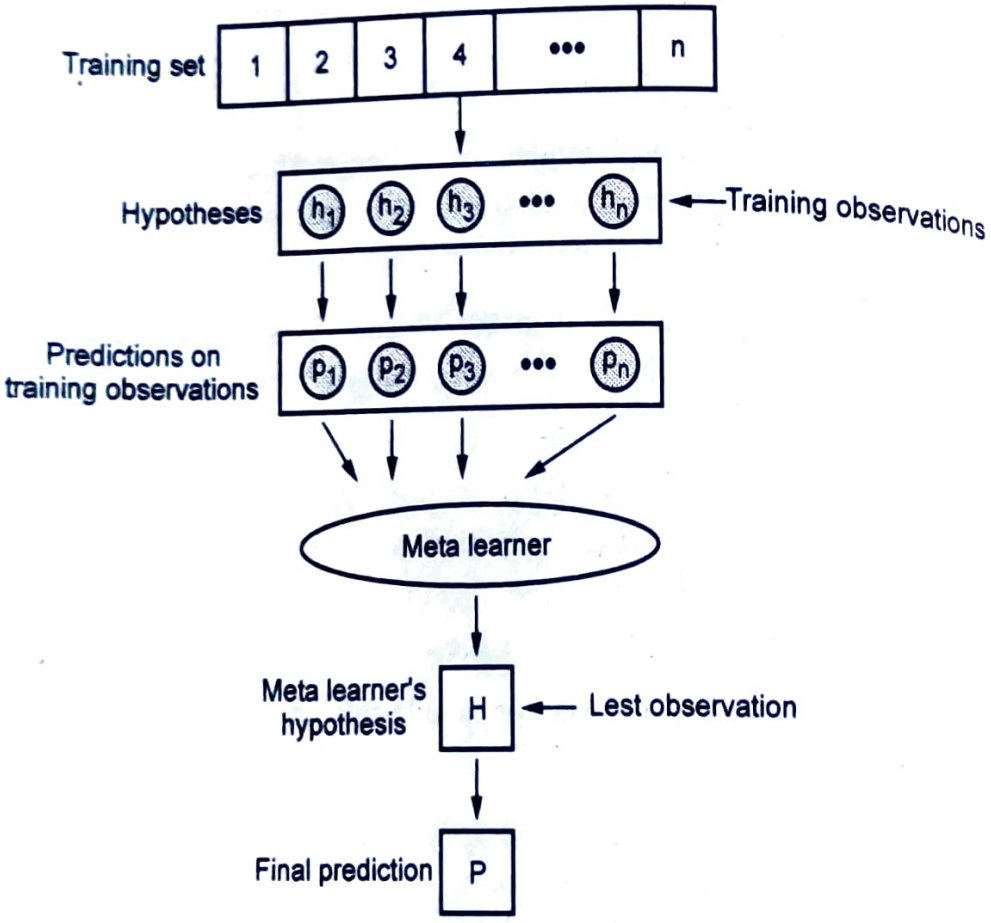


Fig. Q.8.1 Stacking frame

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

$$\forall i = 1, \ldots n \text{ and } \forall k = 1, \ldots\ldots, N : C_k^i$$

$$= L_k (S - s_i)$$

- 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 $((\hat{y}_i^1, \ldots, \hat{y}_i^n), y_i)$, 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.

Q.9 What is Bagging ? Explain Bagging steps. List its advantages and disadvantages.

Ans. : • 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.

Q.10 Explain boosting steps. List advantages and disadvantages of boosting.

Ans. : Boosting Steps :

1. Draw a random subset of training samples d_1 without replacement from the training set D to train a weak learner C_1
2. Draw second random training subset d_2 without replacement from the training set and add 50 percent of the samples that were previously falsely classified/misclassified to train a weak learner C_1
3. Find the training samples d_3 in the training set D on which C_1 and C_1 disagree to train a third weak learner C_3
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.

5.3 : Clustering Fundamentals

Q.11 What is cluster and distance based clustering? List the name of clustering algorithm. Explain application of clustering.

Ans. : Clustering Fundamentals

- 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 unsupervised 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. Q.11.1 shows cluster.

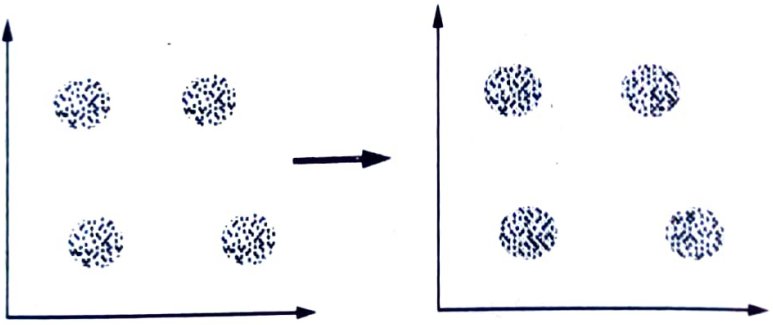
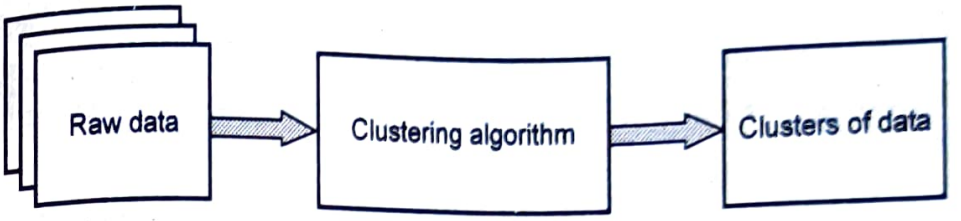
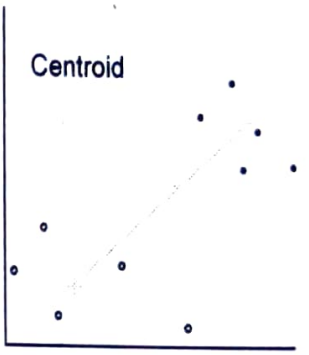


Fig. Q.11.1 Cluster

- 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.**
- 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 of 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 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 clusters. 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 a population. The commonly used distance measure is the Euclidean metric which defines the distance between two points $p = (p_1, p_2, ...)$ and $q = (q_1, q_2, ...)$ 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 this criterion, in such a way that the result of the clustering will suit their needs.

- Clustering analysis helps construct meaningful 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. Probabilistic clustering
- A good clustering method will produce high quality clusters with high intra-class similarity 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 its ability to discover some or all of the hidden patterns.

Examples of Clustering Applications

1. **Marketing :** Help marketers discover distinct groups in their customer bases and then use this knowledge to develop targeted marketing programs.
2. **Land use :** Identification of areas of similar land use in an earth observation database.
3. **Insurance :** Identifying groups of motor insurance policy holders with a high average claim cost.
4. **Urban planning :** Identifying groups of houses according to their house type, value, and geographical location.
5. **Seismology :** Observed earth quake epicenters should be clustered along continent faults.

Q.12 Explain K-means algorithm process.

Ans. : 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.

Q.13 Explain silhouettes.

Ans. : • Silhouette refers to a method of interpretation and validation of clusters of data.

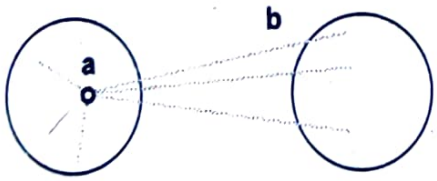
- Silhouettes are a general graphical aid for interpretation and validation of cluster analysis. This technique is available through the silhouette function. In order to calculate silhouettes, two types of data are needed :
  1. The collection of all distances between objects. These distances are obtained from application of dist function on the coordinates of the elements in mat with argument method.
  2. The partition obtained by the application of a clustering technique.
- For each element, a silhouette value is calculated and evaluates the degree of confidence in the assignment of the element :
  1. Well-clustered elements have a score near 1.
  2. Poorly-clustered elements have a score near – 1.
- Thus, silhouettes indicate the objects that are well or poorly clustered. Silhouette coefficient combines ideas of both cohesion and separation, but for individual points, as well as clusters and clustering's.
- For an individual point, I

a = Average distance of i to the points in the same cluster

b = min (average distance of i to points in another cluster)

Silhouette coefficient of i :

$$s = 1 - a/b \text{ if } a < b$$



Silhouette coefficient

- **Cohesion :** Measures how closely related are objects in a cluster.

Separation : Measure how distinct or well-separated a cluster is from other clusters.

Q.14 Explain density-based spatial clustering of applications with noise. List its advantages and disadvantages.

Ans. : Density-Based Spatial Clustering of Applications with Noise : The DBSCAN algorithm can identify clusters in large spatial data sets by looking at the local density of database elements, using only one input parameter.

- The DBSCAN can also determine what information should be classified as noise or outliers.
- In DBSCAN, clustering happens based on two important parameters :
  1. neighbourhood (n) - cutoff distance of a point from (core point - discussed below) for it to be considered a part of a cluster. Commonly referred to as epsilon (abbreviated as eps).
  2. minimum points (m) - minimum number of points required to form a cluster. Commonly referred to as minPts.
- By using the density distribution of nodes in the database, DBSCAN can categorize these nodes into separate clusters that define the different classes. DBSCAN can find clusters of arbitrary shape, as can be seen in the following Fig. Q.14.1.

Fig. Q.14.1

- To find a cluster, DBSCAN starts with an arbitrary point p and retrieves all points density-reachable from p with respect to Eps and MinPts.

- If p is a core point, this procedure yields a cluster with respect to Eps and MinPts . If p is a border point, no points are density-reachable from p and DBSCAN visits the next point of the database.
- MinPts is a minimum number of points in the given neighborhood N(p).
- Three category for each point are as follows :
  1. Core point : if its density is high
  2. Border point : density is low (but in the neighborhood of a core point)
  3. Noise point : any point that is not a core point nor a border point
- Fig Q.14.2 shows category of points.

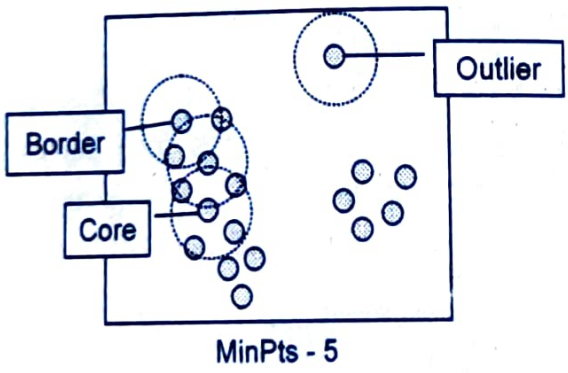


Fig Q.14.2 : Category of points

Algorithm :

1. Arbitrary select a point p
2. Retrieve all points density-reachable from p with repsect to Eps and MinPts.
3. If p is a core point, a cluster is formed.
4. If p is a border point, no points are density-reachable from p and DBSCAN visits the next point of the database.
5. Continue the process until all of the points have been processed

- DBSCAN algorithm have done a good job classifying all the clusters. The DBSCAN algorithm has a solution to all these problems as it can find those complicated cluster shapes very quickly, with only one assigned input parameter. A value for this parameter is also suggested to the user.

Advantages

1. Clusters can have arbitrary shape and size
2. Number of clusters is determined automatically
3. Can separate clusters from surrounding noise
4. Can be supported by spatial index structures

Disadvantages

1. Input parameters may be difficult to determine
2. In some situations very sensitive to input parameter setting

Q.15 What is spectral clustering? List its advantages and disadvantages.

Ans. : Spectral Clustering : • Spectral clustering techniques make use of the spectrum (eigenvalues) of the similarity matrix of the data to perform dimensionality reduction before clustering in fewer dimensions. The similarity matrix is provided as an input and consists of a quantitative assessment of the relative similarity of each pair of points in the dataset

- Given data points $x_1, \ldots x_N$, pairwise affinities $A_{ij} = A(x_i, x_j)$
- Build similarity graph shown in Fig. Q.15.1.

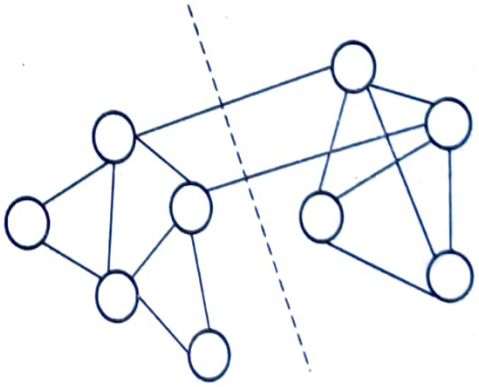


Fig. Q.15.1 : Similarity graph

- Clustering = find a cut through the graph
- Define a cut-type objective function
- The low-dimensional space is determined by the data. Spectral clustering makes use of the spectrum of the graph for dimensionality reduction.
- Projection and clustering equates to graph partition by different min-cut criteria

Advantages :

1. Does not make strong assumptions on the statistics of the clusters
2. Easy to implement.
3. Good clustering results.
4. Reasonably fast for sparse data sets of several thousand elements.

Disadvantages :

1. May be sensitive to choice of parameters
2. Computationaly expensive for large datasets

5.4 : Evaluation Methods based on Ground

Q.16 What is Homogeneity ? Explain completeness.

Ans. : Homogeneity

- Homogeneity metric of a cluster labeling given a ground truth. A clustering result satisfies homogeneity if all of its clusters contain only data points which are members of a single class.
- This metric is independent of the absolute values of the labels : a permutation of the class or cluster label values won't change the score value in any way.
- To define the concepts of entropy H(X) and conditional entropy H(X|Y), which measures the uncertainty of X given the knowledge of Y.

- Therefore, if the class set is denoted as C and the cluster set as K, H(C|K) is a measure of the uncertainty in determining the right class after having clustered the dataset.
- To have a homogeneity score, it's necessary to normalize this value considering the initial entropy of the class set H(C) :

$$h = 1 - \frac{H(C|K)}{H(C)}$$

- In scikit-learn, there's the built-in function homogeneity_score() that can be used to compute this value : **from sklearn.metrics import homogeneity_score**

Completeness

- A complementary requirement is that each sample belonging to a class is assigned to the same cluster.
- A clustering result satisfies completeness if all the data points that are members of a given class are elements of the same cluster.
- This metric is independent of the absolute values of the labels : a permutation of the class or cluster label values won't change the score value in any way.
- This measure can be determined using the conditional entropy H(K|C), which is the uncertainty in determining the right cluster given the knowledge of the class. Like for the homogeneity score, we need to normalize this using the entropy H(K) :

$$c = 1 - \frac{H(C|K)}{H(K)}$$

- We can compute this score (on the same dataset) using the function completeness_score() :

from sklearn.metrics import completeness_score

Q.17 What is adjusted rand index ?

Ans. : Adjusted Rand Index

- The adjusted rand index measures the similarity between the original class partitioning (Y) and the clustering.

- If total number of samples in the dataset is n, the rand index is defined as :

$$R = \frac{a+b}{\binom{n}{2}}$$

- Rand index is defined as the number of pairs of objects that are either in the same group or in different groups in both partitions divided by the total number of pairs of objects.
- The Rand index lies between 0 and 1.
- When two partitions agree perfectly, the Rand index achieves the maximum value 1.
- A problem with Rand index is that the expected value of the Rand index between two random partitions is not a constant.
- This problem is corrected by the adjusted Rand index that assumes the generalized hyper-geometric distribution as the model of randomness.
- The adjusted Rand index has the maximum value 1, and its expected value is 0 in the case of random clusters.
- A larger adjusted Rand index means a higher agreement between two partitions. The adjusted Rand index is recommended for measuring agreement even when the partitions compared have different numbers of clusters.

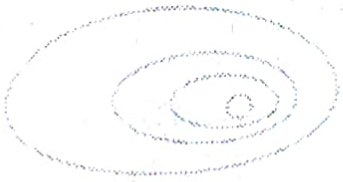
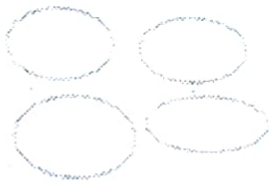
5.5 : Introduction to Meta Classifier

Q.18 Explain various multiclass classification.

Ans. : Multiclass Classification

- 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. Q.18.1 and Q.18.2 shows binary and multiclass classification.

Multi-class 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.

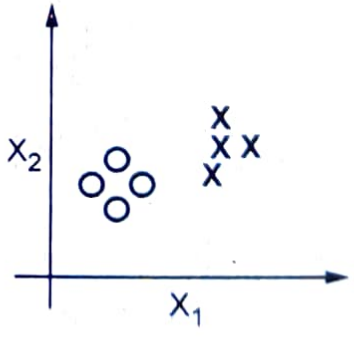


Fig Q.18.1 Binary classification

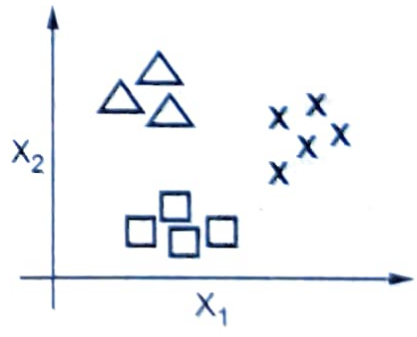


Fig Q.18.2 Multi-class classification

- 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.

Q.19 Explain concepts of weak and eager learner.

Ans. : Concepts of weak and eager learner

- Eager learning is a learning method in which the system tries to construct a general, input-independent target function during training of the system, as opposed to lazy learning, where generalization beyond the training data is delayed until a query is made to the system.
- Combining several weak learners to give a strong learner. It is a kind of multiclassifier systems and meta-learners. Ensemble typically applied to a single type of weak learner.
- **Lazy learning** (e.g., instance-based learning) : Simply stores training data (or only minor processing) and waits until it is given a test tuple
- **Eager learning** (the above discussed methods) : Given a set of training tuples, constructs a classification model before receiving new (e.g., test) data to classify
- Lazy : less time in training but more time in predicting

- Lazy method effectively uses a richer hypothesis space since it uses many local linear functions to form an implicit global approximation to the target function
- Eager : must commit to a single hypothesis that covers the entire instance space

END...

UNIT - VI

6 Clustering Techniques

6.1 : Hierarchical Clustering

Q.1 What is hierarchical clustering ? Explain in detail.

Ans. : • Hierarchical clustering is a widely used data analysis tool. The idea is to build a binary tree of the data that successively merges similar groups of points. Visualizing this tree provides a useful summary of the data.

- Hierarchical clustering is based on the general concept of finding a hierarchy of partial clusters, built using either a bottom-up or a top-down approach.
- Hierarchical clustering can be performed with either a distance matrix or raw data. When raw data is provided, the software will automatically compute a distance matrix in the background.
- This method use distance matrix as clustering criteria. This method does not require the number of clusters K as an input, but needs a termination condition.
- Hierarchical clustering is a widely used data analysis tool.
- The idea is to build a binary tree of the data that successively merges similar groups of points. Visualizing this tree provides a useful summary of the data.
- Hierarchical clustering arranges items in a hierarchy with a treelike structure based on the distance or similarity between them.
- The graphical representation of the resulting hierarchy is a tree-structured graph called a dendrogram.

- The main output of Hierarchical Clustering is a dendrogram, which shows the hierarchical relationship between the clusters
- Hierarchical clustering methods can be further classified as either agglomerative or divisive, depending on whether the hierarchical decomposition is formed in a bottom-up (merging) or top-down (splitting) fashion.
- Hierarchical clustering does not require the specification of a pre-defined number of clusters. Fig. Q.1.1 shows clustering and hierarchical clustering.

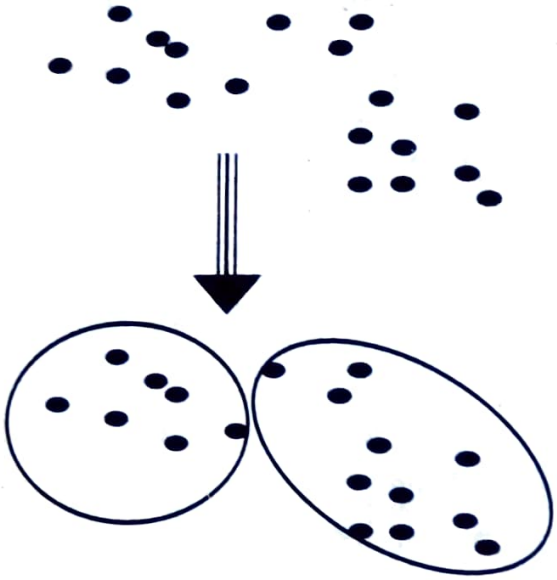


Fig. Q.1.1 (a) Clustering

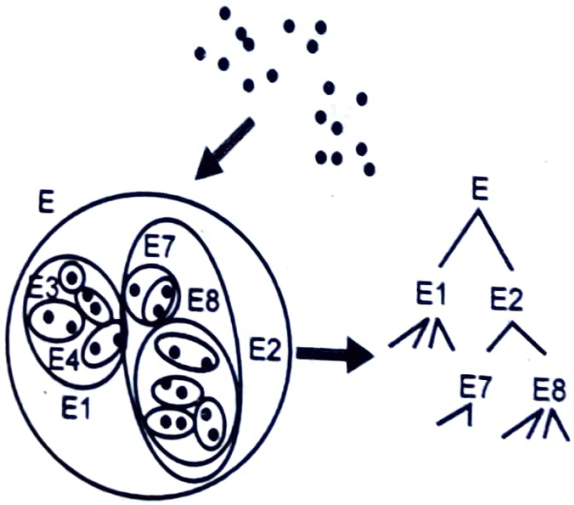


Fig. Q.1.1 (b) Hierarchical clustering

- Use distance matrix as clustering criteria. This method does not require the number of clusters **k** as an input, but needs a termination condition
- The core idea of hierarchical clustering is to create different levels of clusters. Each upper level cluster is conceptually the result of a merging of the clusters at lower levels. At the lowest level each cluster contains a single observation. At the highest level there is a single cluster that contains all observations.

Q.2 Explain agglomerative hierarchical clustering.

Ans. : • This bottom-up strategy starts by placing each object in its own cluster and then merges these atomic clusters into larger and larger clusters, until all of the objects are in a single cluster or until certain termination conditions are satisfied.

- Hierarchical clustering typically works by sequentially merging similar clusters, as shown above. This is known as agglomerative hierarchical clustering.
- Initially, AGNES places each objects into a cluster of its own. The clusters are then merged step-by-step according to some criterion.
- For example, cluster C_1 and C_2 may be merged if an object in C_1 and object in C_2 form the minimum Euclidean distance between any two objects from different clusters.
- In the agglomerative hierarchical approach, we start by defining each data point to be a cluster and combine existing clusters at each step.

Steps :

1. Compute the Euclidean Distance between every pair of patterns. Let the smallest distance be between patterns Xi and Xj.
2. Create a cluster C composed of Xi and Xj. Replace Xi and Xj by cluster vector C in the training set (the average of Xi and Xj).

3. Go back to 1, treating clusters as points, though with an appropriate weight, until no point is left.

- Fig. Q.2.1 shows working of both cluster.
- Each level of the hierarchy represents a specific grouping of the whole data set into disjoint clusters.
- The user must provide an additional criterion in order to extract from a hierarchical clustering an ordinary clustering: a criterion to pick the level of the hierarchy that corresponds to the natural clustering. One such criterion is the gap statistic.
- Since hierarchical clustering is a greedy search algorithm based on a local search, the merging decision made early in the agglomerative process are not necessarily the right ones.

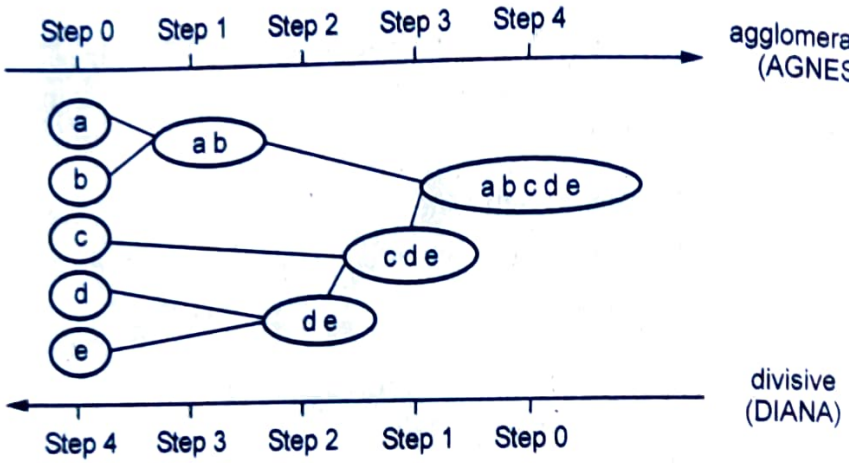


Fig. Q.2.1

Q.3 What is dendrogram ? Explain with example.

Ans. : • Decompose data objects into a several levels of nested partitioning (tree of clusters), called a dendrogram. A clustering of the data objects is obtained by cutting the dendrogram at the desired level, then each connected component forms a cluster.

- The agglomerative hierarchical clustering algorithms available in this program module build a cluster hierarchy that is commonly displayed as a tree diagram called a **dendrogram.**
- They begin with each object in a separate cluster. At each step, the two clusters that are most similar are joined into a single new

cluster. Once fused, objects are never separated. The eight methods that are available represent eight methods of defining the similarity between clusters.

- Suppose we wish to cluster the bivariate data shown in the following scatter plot. In this case, the clustering may be done visually. The data have three clusters and two singletons, 6 and 13.

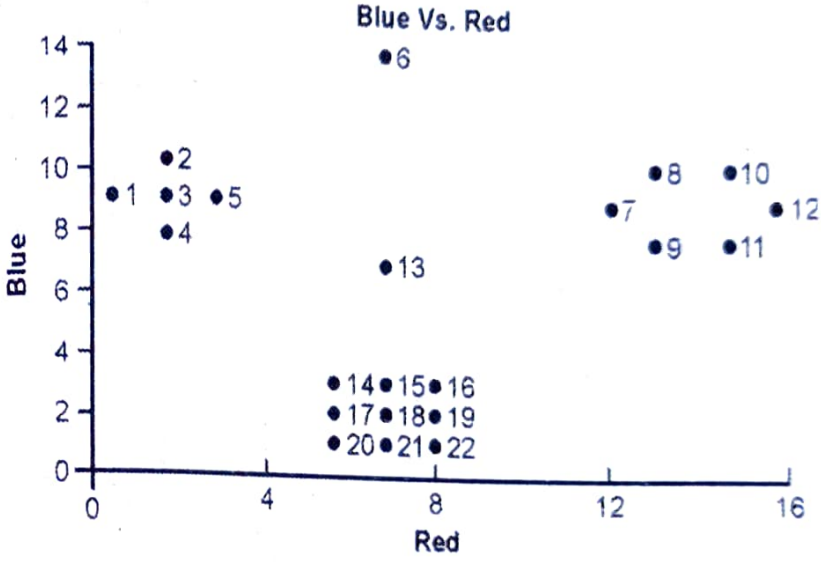


Fig. Q.3.1

- Following is a dendrogram of the results of running these data through the Group Average clustering algorithm.
- The horizontal axis of the dendrogram represents the distance or dissimilarity between clusters. The vertical axis represents the objects and clusters. The dendrogram is fairly simple to interpret. Remember that our main interest is in similarity and clustering.
- Each joining of two clusters is represented on the graph by the splitting of a horizontal line into two horizontal lines. The horizontal position of the split, shown by the short vertical bar, gives the distance (dissimilarity) between the two clusters.
- Looking at this dendrogram, you can see the three clusters as three branches that occur at about the same horizontal distance.

The two outliers, 6 and 13, are fused in rather arbitrarily at much higher distances. This is the interpretation.

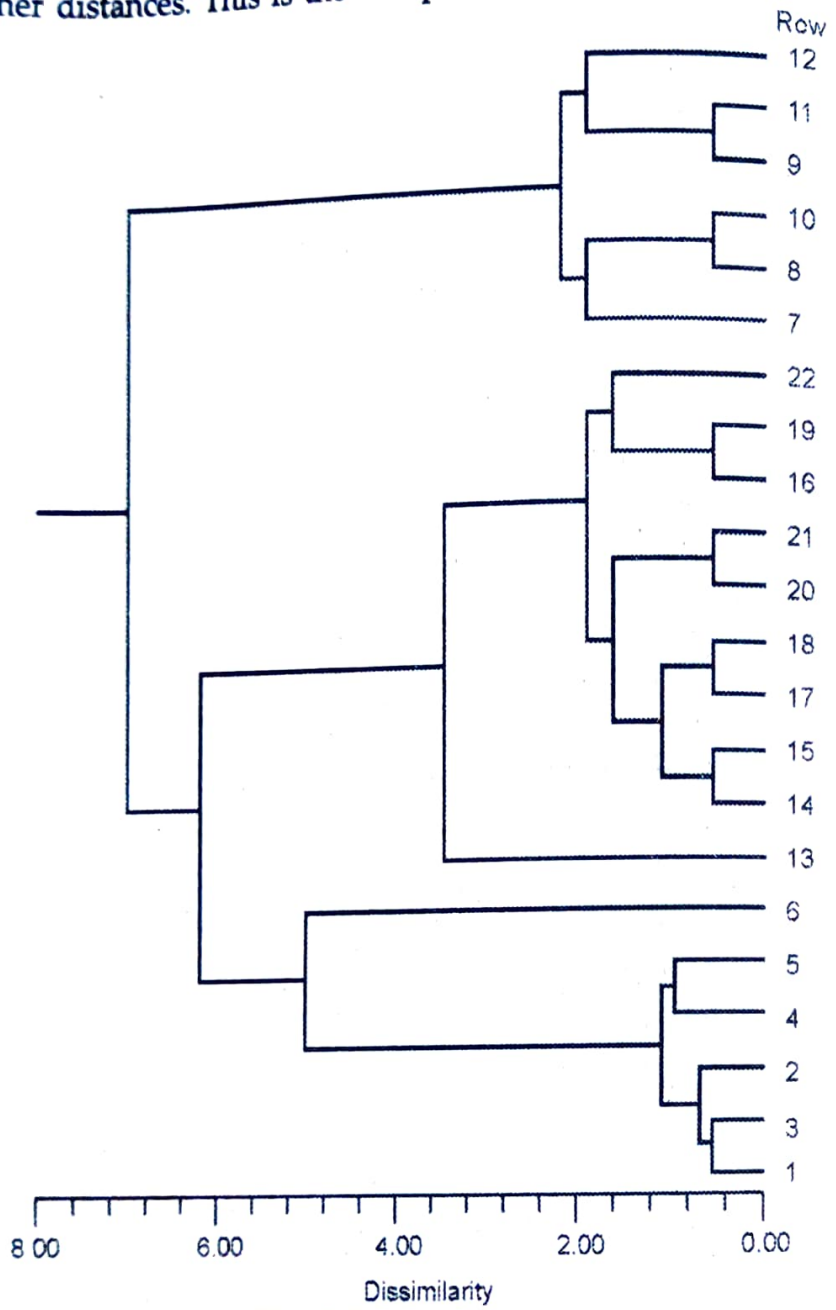


Fig. Q.3.2 Dendrogram

Q.4 Explain connectivity constraints of clustering.

Ans. : • Scikit-learn also allows specifying a connectivity matrix, which can be used as a constraint when finding the clusters to merge.

- In this way, clusters which are far from each other (nonadjacent in the connectivity matrix) are skipped.
- A very common method for creating such a matrix involves using the k-nearest neighbors graph function, that is based on the number of neighbors a sample has.

sklearn.datasets.make_circles(n_samples = 100, shuffle=True,noise=None, random_state=None, factor=0.8)

- It makes a large circle containing a smaller circle in 2d. A simple toy dataset to visualize clustering and classification algorithms.

Parameters :

1. n_samples : int, optional (default=100) → The total number of points generated. If odd, the inner circle will have one point more than the outer circle.
2. shuffle : bool, optional (default=True) → Whether to shuffle the samples.
3. noise : double or None (default=None) → Standard deviation of Gaussian noise added to the data.
4. random_state : int, RandomState instance or None (default) → Determines random number generation for dataset shuffling and noise. Pass an int for reproducible output across multiple function calls. See Glossary.
5. factor : 0 < double < 1 (default=.8) → Scale factor between inner and outer circle.

6.2 : Introduction to Recommendation Systems

Q.5 What is a recommendation systems ?

Ans. : • Recommendation system is a subclass of information filtering system that seeks to predict the "rating" or "preference" a user would give to an item.

- Recommender Systems (RSs) are software tools and techniques providing suggestions for items to be of use to a user.

- The suggestions relate to various decision-making processes, such as what items to buy, what music to listen to, or what online news to read. Item is the general term used to denote what the system recommends to users.
- Recommender systems typically produce a list of recommendations in one of two ways, through collaborative filtering or through content-based filtering.
- Collaborative filtering systems work by collecting user remark in the form of ratings for items in a given field and exploiting similarities in rating actions amongst several users in determining how to recommend an item. Collaborative filtering systems recommend an item to a user based on opinions of other users.
- Content-Based Recommending : Recommendations are based on information on the content of items rather than on other users' opinions. Uses a machine learning algorithm to induce a profile of the users preferences from examples based on a feature description of content.

Q.6 Explain the following :
a) Naïve user based systems b) Content based systems

Ans. : a) Naïve user based systems :

- Naïve Bayes is a probabilistic approach to inductive learning, and belongs to the general class of Bayesian classifiers. These approaches generate a probabilistic model based on previously observed data.
- Let us assume, a set of users represented by feature vectors.

$$U = \{\bar{u}_1, \bar{u}_2, \ldots\ldots, \bar{u}_n\} \text{ where } \bar{u}_n \in R^n$$

- The set of items are represented by

$$I = \{i_1, i_2, \ldots\ldots, i_m\}$$

- Let's assume also that there is a relation which associates each user with a subset of items, items for which an explicit action or feedback has been performed:

$$g(\bar{u}) \rightarrow \{i_1, i_2, \ldots\ldots, i_k\} \text{ where } k \in (0, m)$$

- In a user-based system, the users are periodically clustered and therefore, considering a generic user u, we can immediately determine the ball containing all the users who are similar to our sample :

$$B_R(\bar{u}) = \{\bar{u}_1, \bar{u}_2, \ldots\ldots, \bar{u}_k\}$$

b) Content based systems :

- Any systems implementing a content-based recommendation approach analyze a set of documents and/or descriptions of items previously rated by a user, and build a model or profile of user interests based on the features of the objects rated by that user.
- The recommendation process basically consists in matching up the attributes of the user profile against the attributes of a content object.
- The result is a relevance judgment that represents the user"s level of interest in that object. If a profile correctly reflects user preferences, it is of tremendous advantage for the effectiveness of an information access process.
- Advantages of Content-Based Approach :
  1. No need for data on other users.
  2. Able to recommend to users with unique tastes.
  3. Able to recommend new and unpopular items
  4. Can provide explanations of recommended items by listing content-features that caused an item to be recommended.
- Disadvantages of Content-Based Method
  1. Requires content that can be encoded as meaningful features.
  2. Users' tastes must be represented as a learnable function of these content features.
  3. Unable to exploit quality judgments of other users.

6.3 : Model Free Collaborative Filtering

Q.7 What is collaborative filtering ? Explain model free collaborative filtering.

Ans. : • Collaborative filtering is a method of making automatic predictions (filtering) about the interests of a single user by collecting preferences or taste information from many users (collaborating).

- Collaborative Filtering (CF) uses given rating data by many users for many items as the basis for predicting missing ratings and/or for creating a top-N recommendation list for a given user, called the active user.
- Formally, we have a set of users $U = \{u_1, u_2, \ldots, u_m\}$ and a set of items $I = \{i_1, i_2, \ldots, i_n\}$. Ratings are stored in "m × n" user-item rating matrix.
- The problem of collaborative filtering is to predict how well a user will like an item that he has not rated given a set of historical preference judgments for a community of users.
- Here we try to model a user-item matrix based on the preferences of each user (rows) for each item (columns). For example :

$$M_{u \times I} = \begin{pmatrix} 0 & 1 & 4 & 3 & 0 & 4 & 3 & \cdots & 5 \\ 2 & 1 & 2 & 3 & 0 & 0 & 4 & \cdots & 1 \\ 0 & 2 & 0 & 3 & 1 & 2 & 4 & \cdots & 2 \\ 5 & 0 & 0 & 1 & 2 & 1 & 3 & \cdots & 1 \\ 3 & 0 & 0 & 3 & 0 & 1 & 0 & \cdots & 4 \\ 1 & 4 & 1 & 0 & 3 & 5 & 0 & \cdots & 3 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 2 & 3 & 1 & 2 & 4 & 4 & \cdots & 0 \\ 1 & 3 & 2 & 0 & 0 & 2 & 2 & \cdots & 1 \end{pmatrix}$$

- The ratings are bounded between 1 and 5 (0 means no rating) and our goal is to cluster the users according to their rating vector.

- In order to build the model, we first need to define the user-item matrix as a Python dictionary with the structure:

{user_1 : { item1: rating, item2: rating, ... }, ..., user_n: ...}

Q.8 Explain user based and item based collaborative filtering.

Ans. : • There are two types of collaborative filtering algorithms: user based and item based.

1. User based

- User-based collaborative filtering algorithms work off the premise that if a user (A) has a similar profile to another user (B), then A is more likely to prefer things that B prefers when compared with a user chosen at random.
- The assumption is that users with similar preferences will rate items similarly. Thus missing ratings for a user can be predicted by first finding a neighborhood of similar users and then aggregate the ratings of these users to form a prediction.
- The neighborhood is defined in terms of similarity between users, either by taking a given number of most similar users (k nearest neighbors) or all users within a given similarity threshold.
- Popular similarity measures for CF are the Pearson correlation coefficient and the Cosine similarity.
- For example, a collaborative filtering recommendation system for television tastes could make predictions about which television show a user should like given a partial list of that user's tastes (likes or dislikes).
- Note that these predictions are specific to the user, but use information gleaned from many users. This differs from the simpler approach of giving an average score for each item of interest, for example based on its number of votes.
- User-based CF is a memory-based algorithm which tries to mimics word-of mouth by analyzing rating data from many individuals.

- The two main problems of user-based CF are that the whole user database has to be kept in memory and that expensive similarity computation between the active user and all other users in the database has to be performed.

2. Item-based collaborative filtering

- Item-based CF is a model-based approach which produces recommendations based on the relationship between items inferred from the rating matrix.
- The assumption behind this approach is that users will prefer items that are similar to other items they like.
- The model-building step consists of calculating a similarity matrix containing all item-to-item similarities using a given similarity measure.
- Popular are again Pearson correlation and Cosine similarity. All pair-wise similarities are stored in n × n similarity matrix S.
- Item-based collaborative filtering has become popularized due to its use by YouTube and Amazon to provide recommendations to users.
- This algorithm works by building an item-to-item matrix which defines the relationship between pairs of items.
- When a user indicates a preference for a certain type of item, the matrix is used to identify other items with similar characteristics that can also be recommended.
- Item-based CF is more efficient than user-based CF since the model is relatively small (N × k) and can be fully pre-computed. Item-based CF is known to only produce slightly inferior results compared to user-based CF and higher order models which take the joint distribution of sets of items into account are possible.

Q.9 Discuss memory based and model based algorithm ? List its advantages and disadvantages.

Ans. : 1. Memory-based algorithms :

- Operate over the entire user-item database to make predictions.
- Statistical techniques are employed to find the neighbors of the active user and then combine their preferences to produce a prediction.
- Memory-based algorithms utilize the entire user-item database to generate a prediction. These systems employ statistical techniques to find a set of users, known as neighbors that have a history of agreeing with the target user.
- Once a neighborhood of users is formed, these systems use different algorithms to combine the preferences of neighbors to produce a prediction or top-N recommendation for the active user. The techniques, also known as nearest-neighbor or user-based collaborative filtering are more popular and widely used in practice.
- Dynamic structure. More popular and widely used in practice.

Advantages

1. The quality of predictions is rather good.
2. This is a relatively simple algorithm to implement for any situation.
3. It is very easy to update the database, since it uses the entire database every time it makes a prediction.

Disadvantages

1. It uses the entire database every time it makes a prediction, so it needs to be in memory it is very, very slow.
2. Even when in memory, it uses the entire database every time it makes a prediction, so it is very slow.
3. It can sometimes not make a prediction for certain active users/items. This can occur if the active user has no items in common with all people who have rated the target item.

4. Overfits the data. It takes all random variability in people's ratings as causation, which can be a real problem. In other words, memory-based algorithms do not generalize the data at all.

2. Model-based algorithms :

- Input the user database to estimate or learn a model of user ratings, then run new data through the model to get a predicted output.
- A prediction is computed through the expected value of a user rating, given his/her ratings on other items.
- Static structure. In dynamic domains the model could soon become inaccurate.
- Model-based collaborative filtering algorithms provide item recommendation by first developing a model of user ratings. Algorithms in this category take a probabilistic approach and envision the collaborative filtering process as computing the expected value of a user prediction, given his/her ratings on other items.
- The model building process is performed by different machine learning algorithms such as Bayesian network, clustering and rule-based approaches. The Bayesian network model formulates a probabilistic model for collaborative filtering problem.
- The clustering model treats collaborative filtering as a classification problem and works by clustering similar users in same class and estimating the probability that a particular user is in a particular class C and from there computes the conditional probability of ratings.
- The rule-based approach applies association rule discovery algorithms to find association between co-purchased items and then generates item recommendation based on the strength of the association between items.

Advantages

1. Scalability : Most models resulting from model-based algorithms are much smaller than the actual dataset, so that even for very large datasets, the model ends up being small enough to be used efficiently. This imparts scalability to the overall system.
2. Prediction speed : Model-based systems are also likely to be faster, at least in comparison to memory-based systems because, the time required to query the model is usually much smaller than that required to query the whole dataset.
3. Avoidance of over fitting : If the dataset over which we build our model is representative enough of real-world data, it is easier to try to avoid over-fitting with model-based systems.

Disadvantages

1. Inflexibility : Because building a model is often a time- and resource-consuming process, it is usually more difficult to add data to model-based systems, making them inflexible.
2. Quality of predictions : The fact that we are not using all the information (the whole dataset) available to us, it is possible that with model-based systems, we don't get predictions as accurate as with model-based systems. It should be noted, however, that the quality of predictions depends on the way the model is built. In fact, as can be seen from the results page, a model-based system performed the best among all the algorithms we tried.

Q.10 Briefly discuss singular value decomposition.

Ans. : • Singular Value Decomposition (SVD) is a matrix factorization technique commonly used for producing low-rank approximations.

- The singular value decomposition of a matrix A is the factorization of A into the product of three matrices $A = UDV^T$ where the columns of U and V are ortho-normal and the matrix D is diagonal with positive real entries.

- The columns of V in the singular value decomposition, called the right singular vectors of A, always form an orthogonal set with no assumptions on A. The columns of U are called the left singular vectors and they also form an orthogonal set.
- The singular values are the diagonal entries of the S matrix and are arranged in descending order. The singular values are always real numbers. If the matrix A is a real matrix, then U and V are also real.
- To understand how to solve for SVD, let's take the example of the matrix :

$$A = \begin{bmatrix} 2 & 4 \\ 1 & 3 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

- In this example the matrix is a 4×2 matrix. We know that for an $n \times n$ matrix W, then a nonzero vector **x** is the eigenvector of W if : $W\mathbf{x} = \lambda \mathbf{x}$
- For some scalar λ. Then the scalar λ is called an eigenvalue of A, and **x** is said to be an eigenvector of A corresponding to λ .
- So to find the eigenvalues of the above entity we compute matrices AA^T and A^TA As previously stated, the eigenvectors of AA^T make up the columns of U so we can do the following analysis to find U.

$$AA^T = \begin{bmatrix} 2 & 4 \\ 1 & 3 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 2 & 4 & 0 & 0 \\ 1 & 3 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 20 & 14 & 0 & 0 \\ 14 & 10 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} = W$$

- "Now that we have a n x n matrix we can determine the eigenvalues of the matrix W.

Since $W\mathbf{x} = \lambda \mathbf{x}$ then $(W - \lambda I)\mathbf{x} = 0$

$$\begin{bmatrix} 20-\lambda & 14 & 0 & 0 \\ 14 & 10-\lambda & 0 & 0 \\ 0 & 0 & -\lambda & 0 \\ 0 & 0 & 0 & -\lambda \end{bmatrix} .x = (W - \lambda I)\, x = 0$$

- For a unique set of eigenvalues to determinant of the matrix $(W-\lambda I)$ must be equal to zero. Thus from the solution of the characteristic equation, $|W - \lambda I| = 0$.
- SVD-based recommendation generation technique leads to very fast online performance, requiring just a few simple arithmetic operations for each recommendation but computing the SVD is very expensive.

6.4 Fundamentals of Deep Networks

Q.11 Explain deep learning. What are the challenges in deep learning ?

Ans. : • 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.

Challenges in Deep learning :

- They need to find and process massive datasets for training.

- One of the reasons deep learning works so well is the large number of interconnected neurons, or free parameters, that allow for capturing subtle nuances and variations in data.
- Due to the sheer number of layers, nodes, and connections, it is difficult to understand how deep learning networks arrive at insights.
- Deep-learning networks are highly susceptible to the butterfly effect-small variations in the input data can lead to drastically different results, making them inherently unstable.

Q.12 What is neuron ? Explain basic components of biological neurons.

Ans. : • Artificial neural systems are inspired by biological neural systems. The elementary building block of biological neural systems is the neuron.

- The brain is a collection of about 10 billion interconnected neurons. Each neuron is a cell [right] that uses biochemical reactions to receive, process and transmit information. Fig. Q.12.1 shows biological neural systems.

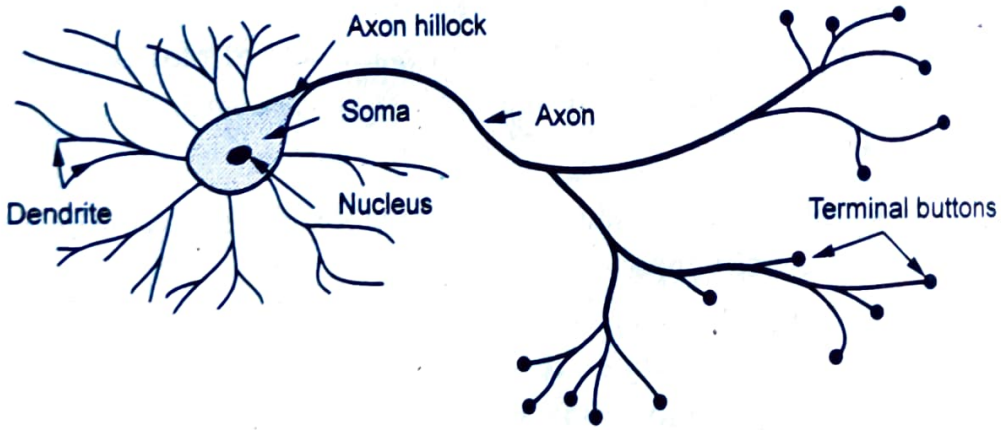


Fig. Q.12.1 Schematic of biological neuron

- The single cell neuron consists of the cell body or soma, the dendrites and the axon. The dendrites receive signals from the axons of other neurons. The small space between the axon of one neuron and the dendrite of another is the synapse. The afferent dendrites conduct impulses toward the soma. The efferent axon conducts impulses away from the soma.

Basic Components of Biological Neurons

1. The majority of ***neurons*** encode their activations or outputs as a series of brief electrical pulses (i.e. spikes or action potentials).
2. The neuron's ***cell body (soma)*** processes the incoming activations and converts them into output activations.
3. The neuron's ***nucleus*** contains the genetic material in the form of DNA. This exists in most types of cells, not just neurons.
4. ***Dendrites*** are fibres which emanate from the cell body and provide the receptive zones that receive activation from other neurons.
5. ***Axons*** are fibres acting as transmission lines that send activation to other neurons.
6. The junctions that allow signal transmission between the axons and dendrites are called ***synapses***. The process of transmission is by diffusion of chemicals called ***neurotransmitters*** across the synaptic cleft.

- Comparison between Biological NN and Artificial NN

Biological NN	Artificial NN
soma	unit
Axon, dendrite	Dendrite
Synapse	Weight
Potential	Weighted sum
Threshold	Bias weight
Signal	Activation

Q.13 Define activation function. What is necessity of activation functions ?

Ans. : • Activation function decides, whether a neuron should be activated or not by calculating weighted sum and further adding bias with it. The purpose of the activation function is to introduce non-linearity into the output of a neuron.

- In a neural network, each neuron has an activation function which specifies the output of a neuron to a given input. Neurons are switches that output 1 when they are sufficiently activated and a 0 when not.

Q.14 Define activation function. Explain the purpose of activation function in multilayer neural networks. Give any two activation functions.

Ans. : • An activation function f performs a mathematical operation on the signal output. The activation functions are chosen depending upon the type of problem to be solved by the network. There are a number of common activation functions in use with neural networks.

- Fig. Q.14.1 shows position of activation function. Unit step function is one of the activation function.

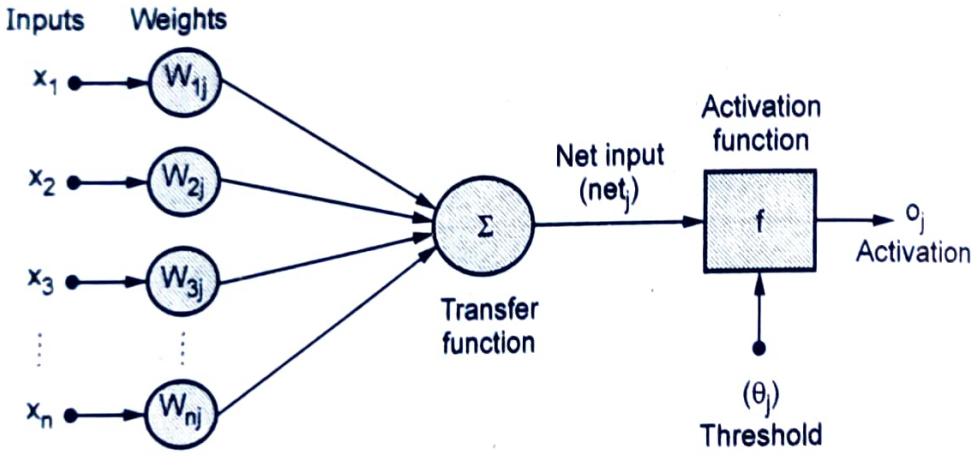


Fig. Q.14.1 Position of activation function

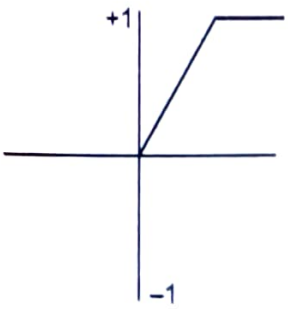
- The cell body itself is considered to have two functions. The first function is integration of all weighted stimuli symbolized by the summation sign. The second function is the activation which transforms the sum of weighted stimuli to an output value which is sent out through connection y.

- Typically the same activation function is used for all neurons in any particular layer. In a multi-layer network if the neurons have linear activation functions the capabilities are no better than a

single layer network with a linear activation function. Hence in most cases nonlinear activation functions are used.

1. **Linear activation function** : The linear activation function will only produce positive numbers over the entire real number range. The linear activation function value is 0 if the argument is less than a lower boundary, increasing linearly from 0 to +1 for arguments equal or larger than the lower boundary and less than an upper boundary, and +1 for all arguments equal or greater than a given upper boundary.
2. **Sigmoid activation function** : The sigmoid function will only produce positive numbers between 0 and 1. The sigmoid activation function is most used for training data that is also between 0 and 1. It is one of the most used activation functions. A sigmoid function produces a curve with an "S" shape. Logistic and hyperbolic tangent functions are commonly used sigmoid functions. The sigmoid functions are extensively used in back propagation neural networks because it reduces the burden of complication involved during training phase.

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



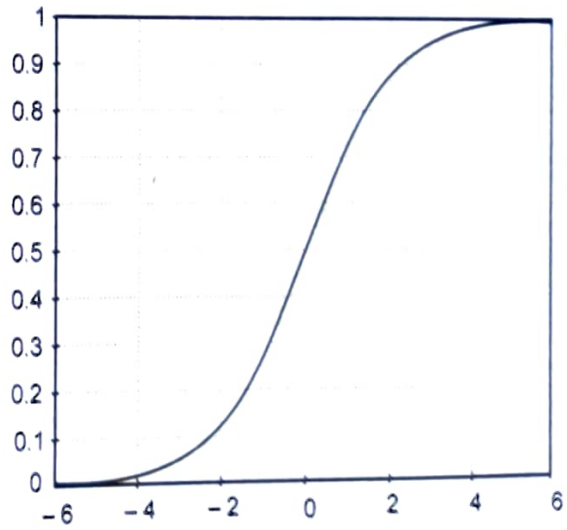


Fig. Q.14.2 (a) Linear functions Fig. Q.14.2 (b) Sigmoid functions

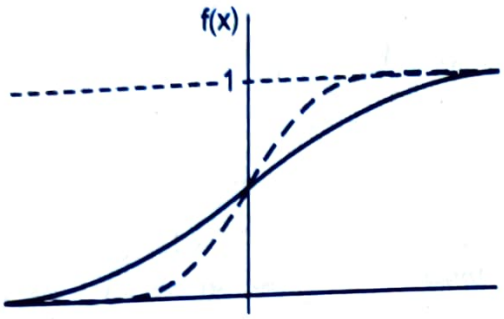


Fig. Q.14.2 (c) Binary sigmoid function

3. **Binary sigmoid :** The logistic function, which is a sigmoid function between 0 and 1 are used in neural network as activation function where the output values are either binary or varies from 0 to 1. It is also called as binary sigmoid or logistic sigmoid.

$$f(x) = \frac{1}{1+e^{-x}}$$

4. **Bipolar sigmoid :** A logistic sigmoid function can be scaled to have any range of values which may be appropriate for a problem. The most common range is from –1 to 1. This id called bipolar sigmoid.

$$f(x) = -1 + \frac{2}{1+e^{-x}}$$

The bipolar sigmoid is also closely related to the hyperbolic tangent function.

$$\tan h\,(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)}$$

$$= \frac{1 - \exp(-2x)}{1 + \exp(-2x)}$$

Q.15 What is perceptron ? Define the architecture of a perceptron ?

Ans. : • 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 set of patterns.

Architecture of a 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. Q.15.1 shows Perceptron.

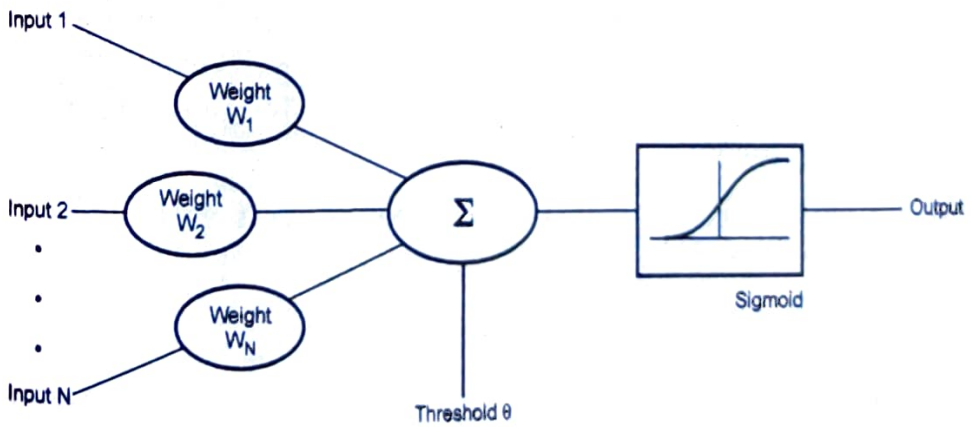


Fig. Q.15.1

- 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

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

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

- 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.

Q.16 What do you mean by zero-centering ?

Ans. : • Feature normalization is often required to neutralize the effect of different quantitative features being measured on different scales. If the features are approximately normally distributed, we can convert them into z-scores by centring on the mean and dividing by the standard deviation. If we don't want to assume normality we can centre on the median and divide by the interquartile range.

- Sometimes feature normalization is understood in the stricter sense of expressing the feature on a [0,1] scale. If we know the feature's highest and lowest values h and l, then we can simply apply the linear scaling.
- Feature calibration is understood as a supervised feature transformation adding a meaningful scale carrying class information to arbitrary features. This has a number of important advantages. For instance, it allows models that require scale, such as linear classifiers, to handle categorical and ordinal features. It

also allows the learning algorithm to choose whether to treat a feature as categorical, ordinal or quantitative.

- The goal of both types of normalization is to make it easier for your learning algorithm to learn. In feature normalization, there are two standard things to do :
  1. Centering : Moving the entire data set so that it is centered around the origin.
  2. Scaling : Rescaling each feature so that one of the following holds :
     a) Each feature has variance 1 across the training data.
     b) Each feature has maximum absolute value 1 across the training data.
- The goal of centering is to make sure that no features are arbitrarily large.

Q.17 Write short note on Tanh and ReLU neurons.

Ans. : • Tanh is also like logistic sigmoid but better. The range of the tanh function is from (-1 to 1). Tanh is also sigmoidal (s - shaped).

• Fig. Q.17.1 shows tanh v/s Logistic Sigmoid.

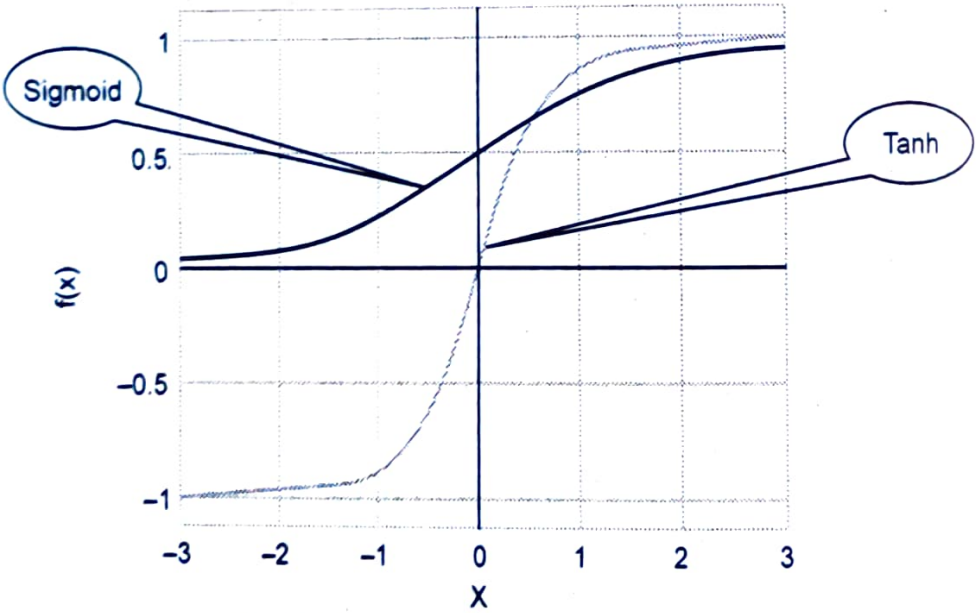


Fig. Q.17.1 : tanh v/s Logistic Sigmoid

- Tanh neuron is simply a scaled sigmoid neuron.
- Problems resolved by Tanh
  1. The output is not zero centered
  2. Small gradient of sigmoid function
- ReLU (Rectified Linear Unit) is the most used activation function in the world right now. Since, it is used in almost all the convolution neural networks or deep learning.
- Fig. Q.17.2 shows ReLU v/s Logistic Sigmoid.

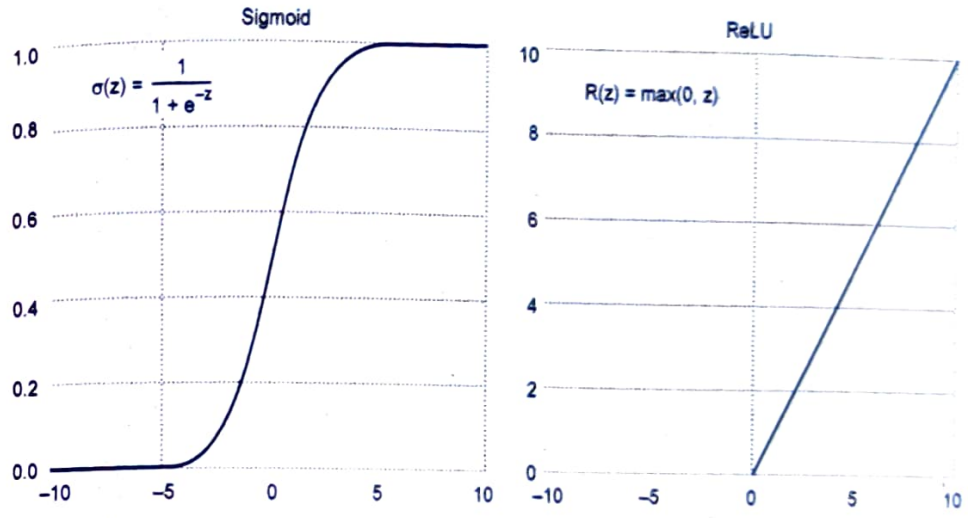


Fig. Q.17.2 : ReLU v/s Logistic Sigmoid

- As you can see, the ReLU is half rectified (from bottom). f(z) is zero when z is less than zero and f(z) is equal to z when z is above or equal to zero.
- Compared to tanh/sigmoid neurons that involve expensive operations (exponentials, etc.), the ReLUcan be implemented by simply thresholding a matrix of activations at zero.

Function	Advantages	Disadvantages
Sigmoid	1. Outputin range (0,1)	1. Saturated Neurons 2. Not zero entered 3. Small gradient 4. Vanishing gradient
Tanh	1. Zero centered, 2. Output in range(-1,1)	1. Saturated Neurons
ReLU	1. Computational efficiency, 2. Accelerated convergence	1. Dead Neurons, 2. Not zero centered

END...

APRIL-2019 [IN SEM] 245 | Solved Paper

Course 2015

Time : 1 Hour] **[Maximum Marks : 30**

Instructions to the candidates :

1) *Attempt questions Q.1 or Q.2, Q.3 or Q.4, Q.5 or Q.6.*

2) *Neat diagrams must be drawn wherever necessary.*

3) *Assume suitable data if necessary.*

Q.1 a) *Define machine learning and state two examples or applications of machine learning in our day to day lives.* [5]

Ans. : Refer Q.2 of Chapter - 1.

Example of machine learning :

1. Medical diagnosis : Machine learning can be used in the techniques and tools that can help in the diagnosis of diseases. It is used for the analysis of the clinical parameters and their combination for the prognosis example prediction of disease progression for the extraction of medical knowledge for the outcome research, for therapy planning and patient monitoring. These are the successful implementations of the machine learning methods. It can help in the integration of computer-based systems in the healthcare sector.

2. Image recognition is one of the most common uses of machine learning. There are many situations where you can classify the object as a digital image. For example, in the case of a black and white image, the intensity of each pixel is served as one of the measurements. In colored images, each pixel provides 3 measurements of intensities in three different colors - red, green and blue (RGB).

b) *What do you mean by supervised and unsupervised learning ? Explain one example of each.* **(Refer Q.5 and Q.6 of Chapter - 1)** [5]

OR

Q.2 **a)** *What is Principal Component Analysis (PCA), when it is used ?* **(Refer Q.15 of Chapter - 1)** [5]

b) *What do you mean by dictionary learning ? What are its applications ?* **(Refer Q.10 of Chapter - 2)** [5]

Q.3 **a)** *Justify the statement : Raw data has a significant impact on feature engineering process.* [5]

Ans. : • Feature engineering is the process of using domain knowledge of the data to create features that make machine learning algorithms work.

- If feature engineering is done correctly, it increases the predictive power of machine learning algorithms by creating features from raw data that help facilitate the machine learning process. Feature Engineering is an art.
- Feature engineering is the process by which knowledge of data is used to construct explanatory variables, features, that can be used to train a predictive model.
- Engineering and selecting the correct features for a model will not only significantly improve its predictive power, but will also offer the flexibility to use less complex models that are faster to run and more easily understood
- One form of feature engineering is to decompose raw attributes into features that will be easier to interpret patterns from.
- For example, decomposing dates or timestamp variables into a variety of constituent parts may allow models to discover and exploit relationships.
- Common time frames for which trends occur include : Absolute time, day of the year, day of the week, month, hour of the day, minute of the hour, year, etc. Breaking dates up into new features such as this will help a model better represent structures or seasonality in the data.
- For example, if you were investigating ice cream sales, and created a "Season of Sale" feature, the model would recognize a peak in the summer season.

- However, an "Hour of Sale" feature would reveal an entirely different trend, possibly peaking in the middle of each day.

b) *Explain different mechanisms for managing missing features in a dataset.* **(Refer Q.4 of Chapter - 2)** [5]

OR

Q.4 a) *With reference to feature engineering, explain data scaling and normalization tasks.* [5]

Ans. : Data scaling and normalization tasks :

- Generic dataset is made up of different values which can be drawn from different distributions, having different scales. Machine learning algorithm is not naturally able to distinguish among these various situations. For this reason it is always preferable to standardize datasets before processing them.
- Standardization : To transform data so that it has zero mean and unit variance. Also called scaling.
- Use function **sklearn.preprocessing.scale()**
- Parameters :

X : Data to be scaled

with_mean : Boolean. Whether to center the data (make zero mean)

with_std : Boolean (whether to make unit standard deviation

- Normalization : To transform data so that it is scaled to the [0,1] range.
- Use function **sklearn.preprocessing.normalize()**
- Parameters :

X : Data to be normalized

norm : which norm to use : l1 or l2

axis : whether to normalize by row or column

- Normalizing in scikit-learn refers to rescaling each observation (row) to have a length of 1.

- This preprocessing can be useful for sparse datasets (lots of zeros) with attributes of varying scales when using algorithms that weight input values such as neural networks and algorithms that use distance measures such as K-Nearest Neighbors.
- You can normalize data in Python with scikit-learn using the Normalizer class.
- Centering and scaling happen independently on each feature by computing the relevant statistics on the samples in the training set. Mean and standard deviation are then stored to be used on later data using the transform method.
- Standardization of a dataset is a common requirement for many machine learning estimators : they might behave badly if the individual features do not more or less look like standard normally distributed data.
- **Examples :**

```
>>>Hide the prompts and output>>> from sklearn.preprocessing
import StandardScaler
>>> data = [[0, 0], [0, 0], [1, 1], [1, 1]]
>>> scaler = StandardScaler()
>>> print(scaler.fit(data))
StandardScaler(copy=True, with_mean=True, with_std=True)
>>> print(scaler.mean_)
[0.5 0.5]
>>> print(scaler.transform(data))
[ [-1. -1.]
 [-1. -1.]
 [ 1.  1.]
 [ 1.  1.]]
>>> print(scaler.transform([[2, 2]]))
[[3. 3.]]
```

b) *What are the criteria or methodology for creation of Training and Testing data sets in machine learning methods ?* [5]

Ans. : Creating Training and Test Sets :

- 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.

Q.5 a) *What do you mean by a linear regression ? Which applications are best modeled by linear regression ?* **(Refer Q.1 of Chapter - 3)** [5]

b) *Write a short note on : Types of regression.* [5]

Ans. : Types of regression are Linear regression, Logistic regression, Polynomial regression, Stepwise regression, Ridge regression and Lasso regression. Also refer Q.6 of Chapter - 3.

OR

Q.6 *Write short notes on (any 2)* **[10]**

a) *Linearly and non-linearly separable data*
(Refer Q.10 of Chapter - 3)

b) *Classification techniques*

c) *ROC curve* **(Refer Q.21 of Chapter - 3)**

Ans. : Classification techniques :

- Classification techniques are as follows Linear Classifiers : Logistic regression, naive Bayes classifier, nearest neighbor, support vector machines, decision trees, boosted trees, random forest and neural networks. Also refer Q.11 of Chapter - 3.

MAY-2019 [END SEM][5561]-688 | Solved Paper

Course 2015

Time : 2 $\frac{1}{2}$ Hours] **[Maximum Marks : 70**

Instructions to the candidates :

1) ***Solve Q.1 or Q.2, Q.3 or Q.4, Q.5 or Q.6, Q.7 or Q.8.***
2) ***Assume suitable data if necessary.***
3) ***Neat diagrams must be drawn wherever necessary.***
4) ***Figures to the right indicates full marks.***

Q.1 a) *With reference to machine learning, explain the concept of adaptive machines.* **(Refer Q.1 of Chapter - 1)** **[6]**

b) *Explain the role of machine learning algorithms in following applications.*
i) Spam filtering ii) Natural language processing. **[6]**

Ans. : I) Spam filtering

- E-mail provides a perfect way to send millions of advertisements at no cost for the sender, and this unfortunate fact is nowadays extensively exploited by several organizations.
- As a result, the e-mailboxes of millions of people get cluttered with all this so-called unsolicited bulk e-mail also known as "spam" or "junk mail".
- Machine learning methods of recent are being used to successfully detect and filter spam emails.
- Different categories of spam filtering techniques that have been widely applied to overcome the problem of email spam.

1. Content Based Filtering Technique : Content based filtering is usually used to create automatic filtering rules and to classify emails using machine learning approaches, such as Naïve Bayesian classification, Support Vector Machine, K Nearest Neighbor, Neural Networks.

- This method normally analyses words, the occurrence, and distributions of words and phrases in the content of emails and used then use generated rules to filter the incoming email spams.

2. Case Base Spam Filtering Method : Case base or sample base filtering is one of the popular spam filtering methods. Firstly, all emails both non-spam and spam emails are extracted from each user's email using collection model.

- Subsequently, pre-processing steps are carried out to transform the email using client interface, feature extraction, and selection, grouping of email data, and evaluating the process.
- The data is then classified into two vector sets. Lastly, the machine learning algorithm is used to train datasets and test them to decide whether the incoming mails are spam or non-spam

ii) Natural language processing

- The role of machine learning and AI in natural language processing (NLP) and text analytics is to improve, accelerate and automate the underlying text analytics functions and NLP features that turn unstructured text into useable data and insights

c) *Explain role of machine learning the following common un-supervised learning problems :* [8]

i) Object segmentation ii) Similarity detection

Ans. : **i) Object segmentation :** Object segmentation is the process of splitting up an object into a collection of smaller fixed-size objects in order to optimize storage and resources usage for large objects. S3 multi-part upload also creates segmented objects, with an object representing each part.

ii) Similarity detection : In contrast to symmetry detection, automatic similarity detection is much harder and more time-consuming. The symmetry factored embedding and the symmetry factored distance can be used to analyze symmetries in points sets. A hierarchical approach was used for building a graph of all subparts of an object.

OR

Q.2 a) *Explain data formats for supervised learning problem with example.* [6]

Ans. : • In a supervised learning problem, there will always be a dataset, defined as a finite set of real vectors with m features each :

$$X = \{\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n\} \text{ where } \bar{x}_i \in \mathbb{R}^m$$

- To consider each X as drawn from a statistical multivariate distribution D. For is purposes, it's also useful to add a very important condition upon the whole dataset X.

- All samples to be **independent and identically distributed (i.i.d)**. This means all variables belong to the same distribution D, and considering an arbitrary subset of m values, it happens that :

$$P(\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_m) = \prod_{i=1}^{m} P(\bar{x}_i)$$

- The corresponding output values can be both numerical-continuous or categorical. In the first case, the process is called **regression**, while in the second, it is called **classification**.
- Examples of numerical outputs are :

$$Y = \{y_1, y_2, \ldots, y_n\} \text{ where } y_n \in (0,1) \text{ or } y_i \in \mathbb{R}^+$$

b) *What is categorical data ? What is its significance in classification problems ?* **(Refer Q.3 of Chapter - 2)** [6]

c) *Explain the Lasso and ElasticNet types of regression.*
(Refer Q.6 and Q.5 of Chapter - 3) [8]

Q.3 a) *What problems are faced by SVM when used with real datasets ?* **(Refer Q.15 of Chapter - 4)** [3]

b) *Explain the non-linear SVM with example.*
(Refer Q.14 of Chapter - 4) [5]

c) *Write short notes on :* [9]

i) Bernoulli naive Bayes **(Refer Q.5 of Chapter - 4)**

ii) Multinomial naive Bayes

iii) Gaussian naive Bayes

Ans. : ii) Multinomial naive Bayes :

- Multinomial distribution is useful to model feature vectors where each value represents, the number of occurrences of a term or its relative frequency.
- If the feature vectors have n elements and each of them can assume k different values with probability p_k, then :

$$P(X_1 = x_1 \cap X_2 = x_2 \cap \ldots\ldots \cap X_k = x_k) = \frac{n!}{\Pi_i x_i!} \Pi_i p_i^{x_i}$$

- The **sklearn.feature_extraction** module can be used to extract features in a format supported by machine learning algorithms from datasets consisting of formats such as text and image.
- The class DictVectorizer can be used to convert feature arrays represented as lists of standard Python dict objects to the NumPy/SciPy representation used by scikit-learn estimators.

- DictVectorizer implements what is called one-of-K or "one-hot" coding for categorical features. Categorical features are "attribute-value" pairs where the value is restricted to a list of discrete of possibilities without ordering.
- The DictVectorizer is used when features are stored in dictionaries.
- Example :

From sklearn, feature - extraction import Dict Vectorizer

```
x = [{'f1' : 'NP', 'f2' : 'in', 'f3' : False, 'f4' : 7},
     {'f1' : 'NP', 'f2' : 'on', 'f3' : True, 'f4' : 2},
     {'f1' : 'VP', 'f2' : 'in', 'f3' : False, 'f4' : 9}]
vec = DictVectorizer()
Xe = vec.fit_transform(X)
print(Xe.toarray ())
print(vec.vocabulary_).
```

- **The result :**

```
[[1, 0, 1, 0, 0, 7,]
 [1, 0, 0, 1, 1, 2,]
 [0, 1, 1, 0, 0, 9,]]
{'f4' : 5, 'f2 = in' : 2, 'f1 = NP' : 0, 'f1 = VP' : 1,
 'f2 = on' : 3, 'f3' : 4}
```

iii) Gaussian Naïve Bayes

- Gaussian naive Bayes is useful when working with continuous values whose probabilities can be modeled using a Gaussian distribution :

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

```
# Naive Bayes
from sklearn.naive_bayes import GaussianNB
clf = GaussianNB()
clf.fit(X_train, y_train)
pred = clf.predict(X_test)
```

OR

Q.4 a) *Define Bayes theorem. Elaborate Naive Bayes classifier working with example.* **(Refer Q.1 and Q.2 of Chapter - 4)** **[8]**

b) *What are linear support vector machines ? Explain with example.* **[4]**

Ans. : • Let's consider a dataset of feature vectors we want to classify :

$$X = \{\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n\} \text{ where } \bar{x}_i \in \mathbb{R}^m$$

• For simplicity, we assume it as a binary classification and we set our class labels as – 1 and 1 :

$$Y = \{y_1, y_2, \ldots, y_n\} \text{ where } y_n \in \{-1, 1\}$$

• Goal is to find the best separating hyperplane, for which the equation is :

$$\bar{w}^T \bar{x} + b = 0 \text{ where } \bar{w} = \begin{pmatrix} w_1 \\ \vdots \\ w_m \end{pmatrix} \text{ and } \bar{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_m \end{pmatrix}$$

• classifier written as :

$$\bar{y} = f(\bar{x}) = \text{sgn}(\bar{w}^T \bar{x} + b)$$

• In a realistic scenario, the two classes are normally separated by a margin with two boundaries where a few elements lie. Those elements are called **support vectors**.

c) *Explain with example the variant of SVM, the support vector regression.* **[5]**

Ans. : • Support Vector Machine can also be used as a regression method, maintaining all the main features that characterize the algorithm (maximal margin).

• The Support Vector Regression (SVR) uses the same principles as the SVM for classification, with only a few minor differences.

• First of all, because output is a real number it becomes very difficult to predict the information at hand, which has infinite possibilities.

• In the case of regression, a margin of tolerance (epsilon) is set in approximation to the SVM which would have already requested from the problem.

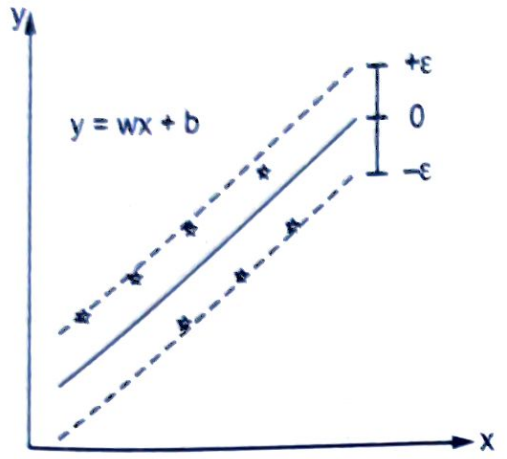


Fig. 1

- Solution :

$$\min \frac{1}{2} ||w||^2$$

- Constraints :

$$y_i - wx_i - b \leq \varepsilon$$

$$wx_i + b - y_i \leq \varepsilon$$

- But besides this fact, there is also a more complicated reason, the algorithm is more complicated therefore to be taken in consideration.
- However, the main idea is always the same: to minimize error, individualizing the hyperplane which maximizes the margin, keeping in mind that part of the error is tolerated.

Q.5 a) *Explain the structure of binary decision tree for a sequential decision process.* **[8]**

Ans. : • A binary decision tree is a structure based on a sequential decision process.

- Starting from the root, a feature is evaluated and one of the two branches is selected. This procedure is repeated until a final leaf is reached, which normally represents the classification target.
- Considering other algorithms, decision trees seem to be simpler in their dynamics; however, if the dataset is splittable while keeping an internal balance, the overall process is intuitive and rather fast in its predictions.
- Decision trees can work efficiently with unnormalized datasets because their internal structure is not influenced by the values assumed by each feature.
- The decision tree always achieves a score close to 1.0, while the logistic regression has an average slightly greater than 0.6.
- However, without proper limitations, a decision tree could potentially grow until a single sample is present in every node. Also refer section 5.1

b) *With reference to clustering, explain the issue of "optimization of clusters".* [5]

Ans. :

- The first method is based on the assumption that an appropriate number of clusters must produce a small inertia.
- However, this value reaches its minimum (0.0) when the number of clusters is equal to the number of samples; therefore, we can't look for the minimum, but for a value which is a trade-off between the inertia and the number of clusters.
- Given a partition of a proximity matrix of similarities into clusters, the program finds a partition with K classes that maximizes a fit criterion. Different options are available for measuring fit.
- The default option (correlation) maximizes the correlation between the data matrix X and a structure matrix A in which a(i,j) = 1 if nodes i and j have been placed in the same class and a(i,j) = 0 otherwise.
- Thus, a high correlation is obtained when the data values are high within-class and low between-class. This assumes similarity data as input.
- For dissimilarity data, the program maximizes the negative of the correlation. Another measure of fit is the density function, which is simply the average data value within classes.
- There is also a pseudo correlation measure that seeks to measure the difference between the average value within classes and the average value between classes. The routine uses a tabu search combinatorial optimization algorithm.

c) *Explain evaluation methods for clustering algorithms.*

(Refer Q.16 and Q.17 of Chapter - 5) [4]

OR

Q.6 a) *With reference to meta classifiers, explain the concepts of weak and eager learner.* **(Refer Q.19 of Chapter - 5)** [8]

b) *Write short notes on :* [9]

i) Adaboost ii) Gradient tree boosting iii) Voting classifier

Ans. : i) 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.

ii) Gradient tree boosting : It is a technique that allows to build tree ensemble step by step with the goal of minimizing a target los function.

- The generic output of the ensemble can be represented as :

$$Y_E = \sum \alpha_i f_i(\bar{x})$$

- Here, $f_i(x)$ is a function representing a weak learner.

- The algorithm is based on the concept of adding a new decision tree at each step so as to minimize the global loss function using the steepest gradient descent method.

$$y_E^{n+1} = y_E^n + \alpha_{n+1} f_{n+1}(\bar{x})$$

After introducing the gradient, the previous expression becomes :

$$y_E^{n+1} = y_E^n + \alpha_{n+1} \sum_i \nabla L(y_{T_i}, y_{E_i})$$

where y_{T_i} is a target class

- Scikit-learn implements the GradientBoostingClassifier class, supporting two classification loss functions : Binomial/multinomial negative log-likelihood and exponential.

iii) Voting classifier : A very interesting ensemble solution is offered by the class VotingClassifier, which is not an actual classifier but a wrapper for a set of different ones that are trained and evaluated in parallel.

- The final decision for a prediction is taken by majority vote according to two different strategies :
  - **Hard voting :** In this case, the class that received the major number of votes, $N_c(y_t)$ will be chosen :

$$\tilde{y} = \arg\max(N_c(y_t^1), N_c(y_t^2), \ldots, Nc(y_t^n))$$

  - **Soft voting :** In this case, the probability vectors for each predicted class (for all classifiers) are summed up and averaged. The winning class is the one corresponding to the highest value :

$$\tilde{y} = \arg\max \frac{1}{\text{Nclassifiers}} \sum_{\text{classifier}} (p_1, p_2, \ldots, p_n)$$

Q.7 a) *With reference to hierarchical clustering, explain the issue of connectivity constraints.* [8]

Ans. : Connectivity constraints

- Scikit-learn also allows specifying a connectivity matrix, which can be used as a constraint when finding the clusters to merge.
- In this way, clusters which are far from each other (nonadjacent in the connectivity matrix) are skipped.

- A very common method for creating such a matrix involves using the k-nearest neighbors graph function, that is based on the number of neighbors a sample has.

sklearn.datasets.make_circles(n_samples = 100, shuffle=True,noise=None, random_state=None, factor=0.8)

- It makes a large circle containing a smaller circle in 2d. A simple toy dataset to visualize clustering and classification algorithms.

Parameters :

1. n_samples : int, optional (default=100) → The total number of points generated. If odd, the inner circle will have one point more than the outer circle.
2. shuffle : bool, optional (default=True) → Whether to shuffle the samples.
3. noise : double or None (default=None) → Standard deviation of Gaussian noise added to the data.
4. random_state : int, RandomState instance or None (default) → Determines random number generation for dataset shuffling and noise. Pass an int for reproducible output across multiple function calls. See Glossary.
5. factor : 0 < double < 1 (default=.8) → Scale factor between inner and outer circle.

b) *What are building blocks of deep networks, elaborate.* [8

Ans. : • The building block of the deep neural networks is called the sigmoid neuron. Deep network also includes neural network, perceptrons, feed forward neural network, Tanh and ReLU neuron. Also refer Q.17 of Chapter - 6.

OR

Q.8 a) *With reference to deep learning, explain the concept of deep architecture ?* [8

Ans. : • Deep learning architectures are based on a sequence of heterogeneous layers which perform different operations organized in a computational graph.

- The output of a layer, correctly reshaped, is fed into the following one, until the output, which is normally associated with a loss function to optimize.

- A **fully connected layer** is made up of n neurons and each of them receives all the output values coming from the previous layer.
- It can be characterized by a weight matrix, a bias vector, and an activation function :

$$\bar{y} = f(W\bar{x} + \bar{b})$$

- They are normally used as intermediate or output layers, in particular when it's necessary to represent a probability distribution.
- **Convolutional layers** are normally applied to bidimensional inputs. They are based on the discrete convolution of a small kernel k with a bidimensional input

$$(k * Y) = Z(i, j) = \sum_m \sum_n k(m, n) Y(i-m, j-n)$$

- A layer is normally made up of n fixed-size kernels, and their values are considered as weights to learn using a back-propagation algorithm.
- More than one pooling layer is used to reduced the complexity when the number of convolutions is very high. Their task is to transform each group of input points into a single value using a predefined strategy.
- A **dropout layer** is used to prevent overfitting of the network by randomly setting a fixed number of input elements to 0. This layer is adopted during the training phase, but it's normally deactivated during test, validation, and production phases.

b) *Justify with elaboration the following statement :*

The k-means algorithm is based on the strong initial condition to decide the number of clusters through the assignment of 'k' initial centroids or means. [8]

Ans. : • The k-means algorithm is based on the strong initial condition to decide the number of clusters through the assignment of k initial centroids or means :

$$K^{(0)} = \{\mu_1^{(0)}, \mu_2^{(0)}, \ldots, \mu_k^{(0)}\}$$

- Then the distance between each sample and each centroid is computed and the sample is assigned to the cluster where the distance is minimum.
- This approach is often called minimizing the inertia of the clusters, which is defined as follows :

$$SS_{W_i} = \sum_t \| x_t - \mu_i \|^2 \; \forall i \in (1, k)$$

- The process is iterative, once all the samples have been processed, a new set of centroids $K^{(1)}$ is computed, and all the distances are recomputed. The algorithm stops when the desired tolerance is reached.
- When the centroids become stable and, therefore, the inertia is minimized. This approach is quite sensitive to the initial conditions, and some methods have been studied to improve the convergence speed. One of them is called k-means^{++}
- Let's consider a simple example with a dummy dataset :

```
from sklearn.datasets import make_blobs
nb_samples = 1000
X, _ = make_blobs(n_samples=nb_samples, n_features=2,
centers=3, cluster_std=1.5)
```

- Let's consider the case of concentric circles. scikit-learn provides a built-in function to generate such datasets :

```
from sklearn.datasets import make_circles
>>> nb_samples = 1000
>>> X, Y = make_circles(n_samples=nb_samples, noise=0.05)
```

- k-means converged on the two centroids in the middle of the two half-circles, and the resulting clustering is quite different from what we expected.
- Moreover, if the samples must be considered different according to the distance from the common center, this result will lead to completely wrong predictions. It's obvious that another method must be employed.

END...