**Unit IV**

**Classification and Prediction:**

* Classification and prediction are two forms of data analysis that can be used to extract models describing important data classes or to predict future data trends.
* Classification predicts categorical (discrete, unordered) labels, *prediction* models continuous valued functions. For example, we can build a classification model to categorize bank loan applications as either safe or risky, or a prediction model to predict the expenditures of potential customers on computer equipment given their income and occupation.
* A predictor is constructed that predicts a continuous-valued function, or ordered value, as opposed to a categorical label. Regression analysis is a statistical methodology that is most often used for numeric prediction.
* Many classification and prediction methods have been proposed by researchers in machine learning, pattern recognition, and statistics. Most algorithms are memory resident, typically assuming a small data size.
* Recent data mining research has built on such work, developing scalable classification and prediction techniques capable of handling large disk-resident data.

**Issues Regarding Classification and Prediction:**

**1. Preparing the Data for Classification and Prediction:** The following preprocessing steps may be applied to the data to help improve the accuracy, efficiency, and scalability of the classification or prediction process.

**(i)Data cleaning:**

This refers to the preprocessing of data in order to remove or reduce *noise* (by applying smoothing techniques) and the treatment of *missing values* (e.g., by replacing a missing value with the most commonly occurring value for that attribute, or with the most probable value based on statistics). Although most classification algorithms have some mechanisms for handling noisy or missing data, this step can help reduce confusion during learning.

**(ii)Relevance analysis:**

Many of the attributes in the data may be *redundant*.

Correlation analysis can be used to identify whether any two given attributes are statistically related. For example, a strong correlation between attributes *A*1 and *A*2 would suggest that one of the two could be removed from further analysis. A database may also contain *irrelevant* attributes. Attribute subset selection can be used in these cases to find a reduced set of attributes such that the resulting probability distribution of the data classes is as close as possible to the original distribution obtained using all attributes. Hence, relevance analysis, in the form of correlation analysis and attribute subset selection, can be used to detect attributes that do not contribute to the classification or prediction task. Such analysis can help improve classification efficiency and scalability.

**(iii)Data Transformation and Reduction**

The data may be transformed by normalization, particularly when neural networks or methods involving distance measurements are used in the learning step. Normalization involves scaling all values for a given attribute so that they fall within a small specified range, such as -1 to +1 or 0 to 1.The data can also be transformed by *generalizing* it to higher-level concepts. Concept hierarchies may be used for this purpose. This is particularly useful for continuous valued attributes. For example, numeric values for the attribute *income* can be generalized to discrete ranges, such as *low, medium*, and *high*. Similarly, categorical attributes, like *street*, can be generalized to higher-level concepts, like *city.*

Data can also be reduced by applying many other methods, ranging from wavelet transformation and principle components analysis to discretization techniques, such as binning, histogram analysis, and clustering.

**Comparing Classification and Prediction Methods:**

**Accuracy:** The accuracy of a classifier refers to the ability of a given classifier to correctly predict the class label of new or previously unseen data (i.e., tuples without class label information). The accuracy of a predictor refers to how well a given predictor can guess the value of the predicted attribute for new or previously unseen data.

**Speed:** This refers to the computational costs involved in generating and using the given classifier or predictor.

**Robustness:** This is the ability of the classifier or predictor to make correct predictions given noisy data or data with missing values.

**Scalability:** This refers to the ability to construct the classifier or predictor efficiently given large amounts of data.

**Interpretability:** This refers to the level of understanding and insight that is provided by the classifier or predictor.

Interpretability is subjective and therefore more difficult assess.

**Classification by Decision Tree Induction:**

Decision tree induction is the learning of decision trees from class-labeled training tuples.

A decision tree is a flowchart-like tree structure, where

Each internal node denotes a test on an attribute.

Each branch represents an outcome of the test.

Each leaf node holds a class label.

The topmost node in a tree is the root node.



* The construction of decision tree classifiers does not require any domain knowledge or parameter setting, and therefore I appropriate for exploratory knowledge discovery.
* Decision trees can handle high dimensional data.
* Their representation of acquired knowledge in tree form is intuitive and generally easy to assimilate by humans.
* The learning and classification steps of decision tree induction are simple and fast.
* In general, decision tree classifiers have good accuracy.
* Decision tree induction algorithms have been used for classification in many application areas, such as medicine, manufacturing and production, financial analysis, astronomy, and molecular biology.

**Algorithm for Decision Tree Induction:**



The algorithm is called with three parameters:

Data partition

Attribute list

Attribute selection method

The parameter attribute list is a list of attributes describing the tuples. Attribute selection method specifies a heuristic procedure for selecting the attribute that ―best‖ discriminates the given tuples according to class. The tree starts as a single node, *N*, representing the training tuples in *D.*

* If the tuples in *D* are all of the same class, then node *N* becomes a leaf and is labeled with that class
* All of the terminating conditions are explained at the end of the algorithm.
* Otherwise, the algorithm calls Attribute selection method to determine the splitting criterion.
* The splitting criterion tells us which attribute to test at node N by determining the ―best‖ way to separate or partition the tuples in D into individual classes.
* There are three possible scenarios. Let *A* be the splitting attribute. *A* has *v* distinct values, {*a*1, *a*2, … ,*av*}, based on the training data.**1 A is discrete-valued:**
* In this case, the outcomes of the test at node N correspond directly to the known values of A.
* A branch is created for each known value, aj, of A and labeled with that value.
* Aneed not be considered in any future partitioning of the tuples.

**2 A is continuous-valued:**

* In this case, the test at node N has two possible outcomes, corresponding to the conditions A <=split point and A >split point, respectively where split point is the split-point returned by Attribute selection method as part of the splitting criterion.

**3 A is discrete-valued and a binary tree must be produced:**

* The test at node N is of the form―A€SA?‖. SA is the splitting subset for A, returned by Attribute selection method as part of the splitting criterion. It is a subset of the known values of A.



1. If A is Discrete valued (b)If A is continuous valued (c) IfA is discrete-valued and a binary tree must be produced:

**Bayesian Classification:**

Bayesian classifiers are statistical classifiers.

They can predict class membership probabilities, such as the probability that a given tuple belongs to a particular class.

Bayesian classification is based on Bayes’ theorem.

**Bayes’ Theorem:**

Let X be a data tuple. In Bayesian terms, X is considered ―evidence, and it is described by measurements made on a set of n attributes.

Let H be some hypothesis, such as that the data tuple X belongs to a specified class C.

For classification problems, we want to determine P(H|X), the probability that the hypothesis H holds given the ―evidence‖ or observed data tuple X.

P(H|X) is the posterior probability, or a posteriori probability, of H conditioned on X.

Bayes’ theorem is useful in that it provides a way of calculating the posterior probability, *P*(*H*|***X***), from *P*(*H*), *P*(***X***|*H*), and *P*(***X***).



**Example:** Predicting a class label using naive Bayesian classification. We wish to predict the class label of a tuple using naive Bayesian classification, given the same training data as. The data tuples are described by the attributes age, income, student, and credit rating. The class label attribute, buys computer, has two distinct values (namely, {yes, no}). Let C1 correspond to the class buys computer = yes and C2 correspond to buys computer = no. The tuple we wish to classify is

**X = (age = youth, income = medium, student = yes, credit rating = fair)**

We need to maximize P(X|Ci)P(Ci), for i = 1, 2. P(Ci), the prior probability of each class, can be computed based on the training tuples:

P(buys computer = yes) = 9/14 = 0.643

P(buys computer = no) = 5/14 = 0.357

To compute P(X|Ci), fori = 1, 2, we compute the following conditional probabilities:

P(age = youth | buys computer = yes) = 2/9 = 0.222

P(age = youth | buys computer = no) = 3/5 = 0.600

P(income = medium | buys computer = yes) = 4/9 = 0.444

P(income = medium | buys computer = no) = 2/5 = 0.400

P(student = yes | buys computer = yes) = 6/9 = 0.667

P(student = yes | buys computer = no) = 1/5 = 0.200

P(credit rating = fair | buys computer = yes) = 6/9 = 0.667

P(credit rating = fair | buys computer = no) = 2/5 = 0.400

Using these probabilities, we obtain

P(X|buys computer = yes) **=**  P(age = youth | buys computer = yes)

× P(income = medium | buys computer = yes)

× P(student = yes | buys computer = yes)

× P(credit rating = fair | buys computer = yes)

× P(credit rating = fair | buys computer = yes)

**=** 0.222 × 0.444 × 0.667 × 0.667 = 0.044.

Similarly,

P(X|buys computer = no) = 0.600 × 0.400 × 0.200 × 0.400 = 0.019.

To find the class, Ci , that maximizes P(X|Ci)P(Ci), we compute

P(X|buys computer = yes)P(buys computer = yes) = 0.044 × 0.643 = 0.028

P(X|buys computer = no)P(buys computer = no) = 0.019 × 0.357 = 0.007

Therefore, the na¨ıve Bayesian classifier predicts buys computer = yes for tuple X

**Naive Bayesian Classification**

The naïve Bayesian classifier, or simple Bayesian classifier, works as follows:

**1.**Let D be a training set of tuples and their associated class labels. As usual, each tuple is represented by an n-dimensional attribute vector, X = (x1, x2, …,xn), depicting n measurements made on the tuple from n attributes, respectively, A1, A2, …, An.

**2.** Suppose that there are m classes, C1, C2, …, Cm. Given a tuple, X, the classifier will predict that X belongs to the class having the highest posterior probability, conditioned on X. That is, the naïve Bayesian classifier predicts that tuple X belongs to the class Ci if and only if



Thus we maximize P(Cij**X**). The classCifor which P(Cij**X**) is maximized is called the maximum posteriori hypothesis. By Bayes’ theorem



**3.**As P(X) is constant for all classes, only P(X|Ci)P(Ci) need be maximized. If the class prior probabilities are not known, then it is commonly assumed that the classes are equally likely, that is, P(C1) = P(C2) = …= P(Cm), and we would therefore maximize P(X|Ci). Otherwise, we maximize P(X|Ci)P(Ci).

**4.** Given data sets with many attributes, it would be extremely computationally expensive to compute P(X|Ci). In order to reduce computation in evaluating P(X|Ci), the naive assumption of class conditional independence is made. This presumes that the values of the attributes are conditionally independent of one another, given the class label of the tuple. Thus,



We can easily estimate the probabilities *P*(*x*1|*Ci*), *P*(*x*2|*Ci*), : : : , *P*(*xn*|*Ci*) from the training tuples. For each attribute, we look at whether the attribute is categorical or continuous-valued. For instance, to compute *P*(***X***|*Ci*), we consider the following:

If *Ak* is categorical, then *P*(*xk*|*Ci*) is the number of tuples of class *Ci* in *D* having the value *xk*for *Ak*, divided by |*Ci*,*D*| the number of tuples of class *Ci* in *D*.

If *Ak* is continuous-valued, then we need to do a bit more work, but the calculation is pretty straightforward.

A continuous-valued attribute is typically assumed tohave a Gaussian distribution with a mean *μ* and standard deviation, defined by



In order to predict the class label of *X*, *P*(*X*j*Ci*)*P*(*Ci*) is evaluated for each class *Ci*. The classifier predicts that the class label of tuple *X* is the class *Ci* if and only if



**Cluster Analysis:**

* The process of grouping a set of physical or abstract objects into classes of similar objects

is called clustering.

* A cluster is a collection of data objects that are similar to one another within the same cluster and are dissimilar to the objects in other clusters.
* A cluster of data objects can be treated collectively as one group and so may be considered as a form of data compression.
* Cluster analysis tools based on k-means, k-medoids, and several methods have also been built into many statistical analysis software packages or systems, such as S-Plus, SPSS, and SAS.

**Applications:**

* Cluster analysis has been widely used in numerous applications, including market research, pattern recognition, data analysis, and image processing.
* In business, clustering can help marketers discover distinct groups in their customer bases and characterize customer groups based on purchasing patterns.
* In biology, it can be used to derive plant and animal taxonomies, categorize genes with similar functionality, and gain insight into structures inherent in populations.
* Clustering may also help in the identification of areas of similar land use in an earth observation database and in the identification of groups of houses in a city according to house type, value, and geographic location, as well as the identification of groups of automobile insurance policy holders with a high average claim cost.
* Clustering is also called data segmentation in some applications because clustering
* Partitions large data sets into groups according to their *similarity*.
* Clustering can also be used for outlier detection, Applications of outlier detection include the detection of credit card fraud and the monitoring of criminal activities in electronic commerce.

**Requirements of Clustering in Data Mining:**

**Scalability:**

Many clustering algorithms work well on small data sets containing fewer than several hundred data objects; however, a large database may contain millions of objects. Clustering on a sample of a given large data set may lead to biased results. Highly scalable clustering algorithms are needed.

**Ability to deal with different types of attributes:**

Many algorithms are designed to cluster interval-based (numerical) data. However, applications may require clustering other types of data, such as binary, categorical (nominal), and ordinal data, or mixtures of these data types.

**Discovery of clusters with arbitrary shape:**

Many clustering algorithms determine clusters based on Euclidean or Manhattan distance measures. Algorithms based on such distance measures tend to find spherical clusters with similar size and density. However, a cluster could be of any shape. It is important to develop algorithms that can detect clusters of arbitrary shape.

**Minimal requirements for domain knowledge to determine input parameters:**

Many clustering algorithms require users to input certain parameters in cluster analysis (such as the number of desired clusters). The clustering results can be quite sensitive to input parameters. Parameters are often difficult to determine, especially for data sets containing high-dimensional objects. This not only burdens users, but it also makes the quality of clustering difficult to control.

**Ability to deal with noisy data:**

Most real-world databases contain outliers or missing, unknown, or erroneous data. Some clustering algorithms are sensitive to such data and may lead to clusters of poor quality.

**Incremental clustering and insensitivity to the order of input records:**

Some clustering algorithms cannot incorporate newly inserted data (i.e., database updates) into existing clustering structures and, instead, must determine a new clustering from scratch. Some clustering algorithms are sensitive to the order of input data. That is, given a set of data objects, such an algorithm may return dramatically different clustering depending on the order of presentation of the input objects. It is important to develop incremental clustering algorithms and algorithms that are insensitive to the order of input.

**High dimensionality:**

A database or a data warehouse can contain several dimensions or attributes. Many clustering algorithms are good at handling low-dimensional data, involving only two to three dimensions. Human eyes are good at judging the quality of clustering for up to three dimensions. Finding clusters of data objects in high dimensional space is challenging, especially considering that such data can be sparse and highly skewed.

**Constraint-based clustering:**

Real-world applications may need to perform clustering under various kinds of constraints. Suppose that your job is to choose the locations for a given number of new automatic banking machines (ATMs) in a city. To decide upon this, you may cluster households while considering constraints such as the city’s rivers and highway networks, and the type and number of customers per cluster. A challenging task is to find groups of data with good clustering behavior that satisfy specified constraints.

**Interpretability and usability:**

Users expect clustering results to be interpretable, comprehensible, and usable. That is, clustering may need to be tied to specific semantic interpretations and applications. It is important to study how an application goal may influence the selection of clustering features and methods.

**Major Clustering Methods:**

Partitioning Methods

Hierarchical Methods

Density-Based Methods

Grid-Based Methods

Model-Based Methods

**Partitioning Methods:**

A partitioning method constructs *k* partitions of the data, where each partition represents a cluster and *k* <= *n*. That is, it classifies the data into *k* groups, which together satisfy the following requirements:

* Each group must contain at least one object, and
* Each object must belong to exactly one group.

A partitioning method creates an initial partitioning. It then uses an iterative relocation technique that attempts to improve the partitioning by moving objects from one group to another. The general criterion of a good partitioning is that objects in the same cluster are close or related to each other, whereas objects of different clusters are far apart or very different.

**4.2.2 Hierarchical Methods:**

A hierarchical method creates a hierarchical decomposition of the given set of data objects. A hierarchical method can be classified as being either agglomerative or divisive, based on how the hierarchical decomposition is formed.

The agglomerative approach, also called the bottom-up approach, starts with each object forming a separate group. It successively merges the objects or groups that are close to one another, until all of the groups are merged into one or until a termination condition holds.

The divisive approach, also called the top-down approach, starts with all of the objects in the same cluster. In each successive iteration, a cluster is split up into smaller clusters, until eventually each object is in one cluster, or until a termination condition holds.

Hierarchical methods suffer from the fact that once a step (merge or split) is done, it can never be undone. This rigidity is useful in that it leads to smaller computation costs by not having to worry about a combinatorial number of different choices.

There are two approaches to improving the quality of hierarchical clustering:

Perform careful analysis of object ―linkages‖ at each hierarchical partitioning, such as in Chameleon, or

Integrate hierarchical agglomeration and other approaches by first using a hierarchical agglomerative algorithm to group objects into micro clusters, and then performing macro clustering on the micro clusters using another clustering method such as iterative relocation.

**Tasks in Data Mining:**

Clustering High-Dimensional Data

Constraint-Based Clustering

**Clustering High-Dimensional Data:**

* It is a particularly important task in cluster analysis because many applications require the analysis of objects containing a large number of features or dimensions.
* For example, text documents may contain thousands of terms or keywords as features, and DNA micro array data may provide information on the expression levels of thousands of genes under hundreds of conditions.
* Clustering high-dimensional data is challenging due to the curse of dimensionality.
* Many dimensions may not be relevant. As the number of dimensions increases, thedata become increasingly sparse so that the distance measurement between pairs ofpoints become meaningless and the average density of points anywhere in the data islikely to be low. Therefore, a different clustering methodology needs to be developedfor high-dimensional data.
* *DEPT OF CSE & IT VSSUT, Burla*
* CLIQUE and PROCLUS are two influential subspace clustering methods, which search for clusters in subspaces ofthe data, rather than over the entire data space.
* Frequent pattern–based clustering, another clustering methodology, extracts distinct frequent patterns among subsets of dimensions that occur frequently. It uses such patterns to group objects and generate meaningful clusters.

**Constraint-Based Clustering:**

* It is a clustering approach that performs clustering by incorporation of user-specified or application-oriented constraints.
* A constraint expresses a user’s expectation or describes properties of the desired clustering results, and provides an effective means for communicating with the clustering process.
* Various kinds of constraints can be specified, either by a user or as per application requirements.
* Spatial clustering employs with the existence of obstacles and clustering under user-specified constraints. In addition, semi-supervised clustering employs for pair wise constraints in order to improve the quality of the resulting clustering.

**Classical Partitioning Methods:** The mostwell-known and commonly used partitioningmethods are

The *k*-Means Method

k-Medoids Method

**Centroid-Based Technique: The** *K***-Means Method:**

The k-means algorithm takes the input parameter, k, and partitions a set of n objects in to clusters so that the resulting intracluster similarity is high but the intercluster similarity is low. Cluster similarity is measured in regard to the mean value of the objects in a cluster, which can be viewed as the cluster’s centroid or center of gravity.

The *k*-means algorithm proceeds as follows.

* First, it randomly selects *k* of the objects, each of which initially represents a cluster mean or center.
* For each of the remaining objects, an object is assigned to the cluster to which it is the most similar, based on the distance between the object and the cluster mean.
* It then computes the new mean for each cluster.
* This process iterates until the criterion function converges.

Typically, the square-error criterion is used, defined as



WhereE is the sum of the square error for all objects in the data set pis the point in space representing a given object miis the mean of cluster Ci

**The k-means partitioning algorithm:** The *k*-means algorithm for partitioning, where each cluster’s center is represented by the mean value of the objects in the cluster.





*k***-Medoids Method:**

* The k-means algorithm is sensitive to outliers because an object with an extremely large value may substantially distort the distribution of data. This effect is particularly exacerbated due to the use of the square-error function.
* Instead of taking the mean value of the objects in a cluster as a reference point, we can pick actual objects to represent the clusters, using one representative object per cluster. Each remaining object is clustered with the representative object to which it is the most similar.
* Thepartitioning method is then performed based on the principle of minimizing the sum of the dissimilarities between each object and its corresponding reference point. That is, an absolute-error criterion is used, defined as



Where *E* is the sum of the absolute error for all objects in the data set

***p*** is the point in space representing a given object in cluster *Cj*

**oj** is the representative object of Cj

* The initial representative objects are chosen arbitrarily. The iterative process of replacing representative objects by non-representative objects continues as long as the quality of the resulting clustering is improved.
* This quality is estimated using a cost function that measures the average dissimilaritybetween an object and the representative object of its cluster.
* To determine whether a non-representative object, oj random, is a good replacement for a current representativeobject, oj, the following four cases are examined for each of the non-representative objects.

**Case 1:** currently belongs to representative object, oj. If ojis replaced by orandomasa representative object and p is closest to one of the other representative objects, oi,i≠j,

then p is reassigned to oi.

**Case 2:** currently belongs to representative object, oj. If ojis replaced by orandomasa representative object and p is closest to orandom, then p is reassigned to orandom.

**Case 3:** currently belongs to representative object, oi, i≠j. If ojis replaced by orandomas a representative object and p is still closest to oi, then the assignment does notchange.

**Case 4:** currently belongs to representative object, oi, i≠j. If ojis replaced byorandomas a representative object and p is closest to orandom, then p is reassigned toorandom.



**The***k***-MedoidsAlgorithm:** The k-medoids algorithm for partitioning based on medoid or central objects.



The *k*-medoids method ismore robust than *k*-means in the presence of noise and outliers, because a medoid is lessinfluenced by outliers or other extreme values than a mean. However, its processing is costlier than the *k*-means method.

**Hierarchical Clustering Methods:**

* A hierarchical clustering method works by grouping data objects into a tree of clusters.
* The quality of a pure hierarchical clusteringmethod suffers fromits inability to performadjustment once amerge or split decision hasbeen executed. That is, if a particular merge or split decision later turns out to have been apoor choice, the method cannot backtrack and correct it.

Hierarchical clustering methods can be further classified as either agglomerative or divisive, depending on whether the hierarchical decomposition is formed in a bottom-up or top-down fashion.

**Agglomerative hierarchical clustering:**

* 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.
* Most hierarchical clustering methods belong to this category. They differ only in their definition of intercluster similarity.

**Divisive hierarchical clustering:**

* This top-down strategy does the reverse of agglomerativehierarchical clustering by starting with all objects in one cluster.
* It subdividesthe cluster into smaller and smaller pieces, until each object forms a cluster on itsown or until it satisfies certain termination conditions, such as a desired number ofclusters is obtained or the diameter of each cluster is within a certain threshold.