MALLA REDDY INSTITUTE OF TECHNOLOGY &SCIENCE
(SPONSORED BY MALLA REDDY EDUCATIONAL SOCIETY)
Permanently Affiliated to JNTUH & Approved by AICTE, New Delhi
NBA Accredited Institution, An ISO 9001:2015 Certified, Approved by UK Accreditation Centre
Granted Status of 2(f) & 12(b) under UGC Act. 1956, Govt. of India.

DATA MINING

FACULTY INCHARGE
P . Subba Rao

HOD C.S.E
Dr. T. Srikanth

Syllabus

UNIT - I Data Mining: Data–Types of Data–, Data Mining Functionalities–InterestingnessPatterns–Classification of Data Mining systems–Datamining Task primitives –Integration of Data mining system with a Data warehouse–Major issuesin Data Mining–Data Preprocessing.

UNIT - II Association Rule Mining: Mining Frequent Patterns–Associations andcorrelations – Mining Methods– Mining Various kinds of Association Rules–CorrelationAnalysis–ConstraintbasedAssociationmining.GraphPatternMining,SPM.

UNIT-III Classification: Classification and Prediction–Basic concepts–Decision tree induction–Bayesian classification,Rule–based classification,Lazy learner.

UNIT-IVClusteringandApplications:Clusteranalysis–TypesofDatainClusterAnalysis–Categorization of Major Clustering Methods– Partitioning Methods,Hierarchical Methods– Density–Based Methods, Grid–Based Methods, OutlierAnalysis.

UNIT - V Advanced Concepts: Basic concepts in Mining data streams–Mining Time–series data—Mining sequence patterns in Transactional databases– Mining Object–Spatial–Multimedia–Text and Webdata–SpatialDatamining–Multimedia Datamining–TextMining–Mining theWorldWideWeb.

TEXTBOOKS:1.DataMining–ConceptsandTechniques–JiaweiHan&MichelineKamber, 3rdEdition Elsevier.

2. Data Mining Introductory andAdvanced topics – Margaret H Dunham, PEA.

REFERENCE BOOK: 1. Ian H. Witten and Eibe Frank, Data Mining: Practical MachineLearning Tools and Techniques(SecondEdition),MorganKaufmann,2005

TEXT BOOKS:

T1: Data base Management Systems, Raghu Ramakrishnan, Johannes Gehrke, McGraw Hill Education (India) Private Limited, 3rd Edition. (Part of UNIT-I, UNIT-II, UNIT-III, UNIT-V)

T2: Data base System Concepts, A. Silberschatz, Henry. F. Korth, S. Sudarshan, McGraw Hill Education(India) Private Limited l, 6th edition.(Part of UNIT-I, UNIT-IV)

REFERENCE BOOKS:

R1: Database Systems, 6th edition, R Elmasri, Shamkant B.Navathe, Pearson Education.

R2: Database System Concepts, Peter Rob & Carlos Coronel, Cengage Learning.

R3: Introduction to Database Management, M. L. Gillenson and others, Wiley Student Edition.

R4: Database Development and Management, Lee Chao, Auerbach publications, Taylor & Francis Group. Introduction to Database Systems, C. J. Date, Pearson Education.

LECTURE NOTES

UNIT-1

1. DataMining

1.1 Data–TypesofData

•Data mining is the process of discovering interesting patterns and knowledge from large amounts of data.

•The data sources can include databases, data warehouses, the Web, other information repositories, or data that are streamed into the system dynamically.

•As a knowledge discovery process, it typically involves data cleaning, dataintegration,dataselection,datatransformation,patterndiscovery,patternevaluation,and knowledge presentation.

TypesofData:

The most basic forms of data for mining applications are database data , datawarehouse data and transactional data . Data mining can also be applied to other forms of data (e.g., data streams, ordered/sequence data, graph or networked data, spatial data, text data, multimedia data, and the WWW).

1.Flat Files2.Database
Data3.DataWarehouse
4.Transactional
Data5.Multimedia
Data6.Spatial
Data7.TimeSeriesData
8.WorldWideWeb(WWW)

Flat files:

•Flatfilesisdefinedasdatafilesintextformorbinaryformwithastructurethatcanbe easily extracted by data mining algorithms.

•Data stored in flat files have no relationship or path among themselves

Flat files are represented by data dictionary.Eg:CSVfile.

Database data:

A database system, also called a database management system(DBMS),consists of a collection of interrelated data, known as a database, and a set of software programs to manage and access the data.

•A relational database is a collection of tables, each of which is assigned a uniquename. Each table consists of a set of attributes (columns or fields) and usually storesa large set of tuples (records or rows)

DataWarehouses:

•A data warehouse is a repository of information collected from multiple sources,stored under a unified schema, and usually residing at a single site.

TransactionalData:

• In general,each record in a transactional database captures a transaction,such as a customer's purchase, a flight booking,or a user's clicks on a web page.

•A transaction typically includes a unique transaction identity number (trans ID) and a list of the items making up the transaction, such as the items purchased in the transaction.

•Transactional databases is a collection of data organized by time stamps, date, etc to represent transaction in databases.

•This type of database has the capability to roll back or undo its operation when atransaction is not completed or committed.

Multimedia Data:

•Multimedia databases consists audio, video, images and text media.

•They can be stored on Object-Oriented Databases.

•They are used to store complex information in a pre-specified formats.

•Ex:Byminingvideodataof a hockey game, we can detect video sequencescorresponding to goals.

Spatial Data:

•Store geographical information.

•Stores data in theformofcoordinates,topology,lines,polygons,etc.

•**Application**: Maps, Global positioning, etc.

TimeSeriesData:

•Timeseriesdatabasescontainsstockexchange dataanduserloggedactivities.

•Handles array of numbers indexed by time, date, etc.

•It requires real-time analysis.

•Stock exchange data can be mined to uncover trends that could help you planinvestment strategies.

WorldWideWeb(WWW):

•Web mining can help us learn about the distribution of information on the WWW ingeneral, characterize and classify web pages, and uncover web dynamics and theassociation and other relationships among different web pages, users, communities,and web-based activities.

1.2 DataMiningFunctionalities

There are a number of data mining

functionalities.These include

1. Characterization and Discrimination

2. The mining of frequent patterns, associations, and correlations.

3. Classification and Regression

4. Clustering

analysisand5.Outlier analysis

Data mining functionalities are used to specify the kinds of patterns to be found indata mining tasks.

•In general,such tasks can be classified into two categories:descriptive and predictive.

•Descriptive mining tasks characterize properties of the data in a target data set.Predictive mining tasks perform induction on the current data in order to make predictions.

Data characterization:

•It is a summarization of the general characteristics or features of a target class ofdata.Thedatacorrespondingtotheuser-specifiedclassaretypicallycollectedbyaquery.

For example, to study the characteristics of software products with sales that increased by 10% in the previous year.

Theoutputofdatacharacterizationcanbepresentedinvariousforms.Examplesinclude pie charts, bar charts, curves, multidimensional data cubes, and multidimensional tables, including crosstabs.

Data discrimination:

•It is a comparison of the general features of the target class data objects against thegeneral features of objects from one or multiple contrasting classes.

•The target and contrasting classes can be specified by a user,and the corresponding dataobjects can be retrieved through database queries.

•For example, a user may want to compare the general features of software productswithsalesthatincreasedby10%lastyearagainstthosewithsalesthatdecreasedby at least 30% during the same period.

Frequent patterns:

• These are patterns that occur frequently in data.

• There are many kinds of frequent patterns, including frequent itemset, frequent subsequences (also known as sequential patterns), and frequent substructures.

•Afrequentitemsettypicallyreferstoasetofitemsthatoftenappeartogetherinatransactional data set—for example, milk and bread, which are frequently bought together in grocery stores by many customers.

•A frequently occurring subsequence, such as the pattern that customers, tend to purchase first a laptop, followed by a digital camera, and then a memory card, is a(frequent) sequential pattern.

•A substructure can refer to different structural forms(e.g.,graphs,trees,orlattices)that may be combined with itemsets or subsequences.

Association analysis:

•Also known as Market Basket Analysis for its wide use in retail sales, Associationanalysisaimstodiscoverassociationsbetweenitemsoccurringtogetherfrequently.A ssociation analysis is based on rules having 2 parts:

1.antecedent

(if)2.consequent(then

)

•An antecedent is an item in a collection, which, when found, also indicates a certain chance of finding a consequent in the collection. In other words, they are associated. From the data association inferences like,

•If a customer buys potato chips, he is about 60% like to buy soft drinks along with it.So, over a big chunk of data, you see this association with a certain degree of confidence.

Classification:

Classification is a data mining technique that categorizes items in a collection, based on some predefined properties. A training set containing items whose properties are known and is used to train the system to predict the category of items from an unknown collection of items. It uses methods like if-then, decision trees or neural networks to predict a class or essentially classify a collection of items.

Clustering analysis:

Cluster Analysis, fundamentally similar to classification, where similar data are grouped together with the difference being that a class label is not known.Clustering algorithms group data based on similar features and dissimilarities. Used in image processing, pattern recognition and bioinformatics, clustering is a popular functionality of data mining.

Outlier analysis:

A dataset may contain objects that do not comply with the general behavior or model of the data. These data objects are outliers. Many data mining methods discard outliers as noise or exceptions.

Ex: fraud detection.

1.3 InterestingnessPatterns

•A datamining system has the potential to generate thousands or even millions of patterns, or rules.

• "Are all of the patterns interesting?" Typically, the answer is no—only a small fraction of the patterns potentially generated would actually be of interest to a given user.

•This raises some serious questions for data mining.

1. What makes a pattern interesting?

2.Can a data mining system generate all of the interesting patterns?

3. Can the system generate only the interesting ones?"

•To answer the first question, a pattern is interesting if it is (1) easily understood byhumans, (2) valid on new or test data with some degree of certainty, (3) potentiallyuseful, and (4) novel. A pattern is also interesting if it validates a hypothesis that theuser sought to confirm.An interesting pattern represents knowledge.

•Several objective measures of pattern interestingness exists.

•Theyarebasedonthestructuresofdiscoveredpatternsandstatisticsunderlyingthem.

•An objective for association rules of the form X=>Y is rule **support**, representing thepercentageoftransactionsfromthetransactiondatabasethatthegivenrulesatisfies.

•Anotherobjectivemeasureforassociationis**confidence**,whichassessesthedegree of certainty of detected associations.

•Support and confidence are defines as

•Support(X=>Y)=P(XUY)

•Confidence(X=>Y)=P(X|Y)

•Although objective measures help identify interesting measures they are insufficient unless combined with subjective measures that reflects the needs and interests of particular user.

•Subjective measures are based on user beliefs in the data.

•The second question—"Can a data mining system generate all of the interestingpatterns?"—referstothecompletenessofadataminingalgorithm.Itisoftenunrealistic and inefficient for data mining systems to generate all possible patterns.Instead, user provided constraints and interestingness measures should be used to focus the search.

•Finally, the third question—"Can a data mining system generate only interestingpatterns?"—is an optimization problem in data mining. It is highly desirable for datamining systems to generate only interesting patterns.This would be efficient for users and data mining systems because neither would have to search through the patterns generated to identify the truly interesting ones.

•Measures of pattern interestingness are essential for the efficient discovery of patterns of value to the given user.

•Such measures can be used after the datamining step in order to rank the discovered patterns according to their interestingness, filtering out the uninteresting ones.

1.4 DataMiningTaskPrimitives

•Each user will have a data mining task in mind, that is, some form of data analysisthat he or she would like to have performed.

•Adataminingtaskcan be specified in the form of a data mining query, which isinput to the data mining system.

•Adata mining query is defined in terms of data mining task primitives.

•These primitives allow the user to interactively communicate with the data miningsystemduringdiscoveryinordertodirectthemining process, or examine thefindingsfrom different angles or depths.

•**The set of task-relevant data to be mined**: This specifies the portions of thedatabaseorthesetofdatainwhichtheuserisinterested. This includes thedatabase attributes or data warehouse dimensions of interest.

•**The kind of knowledge to be mined:** This specifies the data mining functions to beperformed,suchascharacterization,discrimination,associationorcorrelationanalysis, classification, prediction, clustering, outlier analysis, or evolution analysis.

•**Thebackgroundknowledgetobeusedinthediscoveryprocess:**This knowledge about the domain to be mined is useful for guiding the knowledge discovery process and for evaluating the patterns found. Concept hierarchies are a popular form of background knowledge, which allow data to be mined at multiple levels of abstraction.

•**Theinterestingness measures and thresholds for pattern evaluation:** Theymaybeusedtoguidetheminingprocessor,afterdiscovery,toevaluatethe discovered patterns. Different kinds of knowledge may have different interestingness measures.Forexample,interestingness measures for association rules include support and confidence.

•**The expected representation for visualizing the discovered patterns:**This refers to the form in which discovered patterns are to be displayed, which may include rules, tables, charts, graphs, decision trees, and cubes.

1.5 Integration of a Data Mining System with a Database or dataWarehouseSystem

•A critical question in the design of a data mining system is how to integrate or couple the DM system with database(DB) system or a data warehouse(DW) system.

•When a DM system works in an environment that requires it to communicate withotherinformationsystemcomponentssuchasDBandDWsystems,possible integration schemes include

1.No coupling

2.Loosecoupling

3.Semitight coupling

4.Tightcoupling

No coupling:

•No coupling means that a DM system will not utilize any function of a DW or DB system. It may fetch data from a particular source, process data using some datamining algorithms and then store the mining results in another file.

• Such a system suffers from several drawbacks.

•Firsta DB provides a great deal of flexibility and efficiency at storing, organizing ,accessing and processing data. Without using a DB/DW system, a DM system may spend a substantial amount of time finding, collecting, and transforming data.

•Without coupling of such systems a DM system will need to use other tools toextract data, making it difficult to integrate such a system into an information processing environment.

•Thus no coupling represents a poor design.

Loose coupling:

•Loose coupling means that a DM system will use some facilities of a DB or DMsystem, fetching the data from a data repository managed by these systems,performing data mining and then storing the mining results either ina file or in adesignatedplace in a DB orDW.

•Loose coupling is better than No Coupling because it can fetch portions of datastored in databases or data warehouses by using query processing, indexing andother system facilities.

•Because mining does not explore data structures and query optimization methodsprovidedbyDBorDMitisdifficultforloosecouplingtoachievehighscalabilityandgood performance with large datasets.

Semitight coupling:

Semitightcouplingmeansbesideslinkinga DMsystemtoaDB/DWsystemefficientimplementations of a few essential data mining primitives can be provided in theDB/DW system. These primitives include sorting, indexing, aggregation, histogramanalysis, multiway join, statistical measures such as sum, count, max, min and soon.

TightCoupling:

TightcouplingmeansthataDMsystemissmoothlyintegratedintoaDB/DWsystem.The data mining subsystem is treated as one functional component of an informationsystem. Data mining queries and functions are optimized based on mining queryanalysis, data structures, indexing schemes and query processing methods of a DBor DW systems.

1.6 MajorissuesinDataMining

Major issues in data mining are regarding

1. Mining methodology and user interaction
2. Performance and
3. Diverse data types

1. **Mining methodology and user interaction issues:** These reflects the kind of data mined, the ability to mine knowledge at multiple granularities, the use of domain knowledge, ad hoc mining and knowledge visualization.

- **Mining different kinds of knowledge in databases:** Because different users can be interested in different kinds of knowledge, data mining should cover a wide spectrum of data analysis and knowledge discovery tasks, including data characterization, discrimination, association and correlation analysis and so on.. These tasks use the same database in different ways.
- **Interactive mining of knowledge at multiple levels of abstraction:** Because it is difficult to know exactly what can be discoveredwithin a database, the data mining process should be interactive.
- Interactive mining allows users to focus the search for patterns, providing and refining data mining requests based on returned results. Specifically knowledge should be mined by drilling down, rolling up and pivoting through the data space and knowledge space interactively.
- **Presentation and visualization of data mining results:** Discovered knowledge should be expressed in high level languages, visual representation or other expressive forms so that the knowledge can be easily understood and directly usable by humans. This requires the system to adopt expressive knowledge representation techniques such as trees, tables, rules, graphs, charts, crosstabs, matrices.
- **Handling noisy or incomplete data:** The data stored in a database may reflect noise, exceptional cases, or incomplete data objects. When mining data regularities, these objects may confuse the process causing the knowledge model constructed to overfit the data. As a

result, the accuracy of the discovered patterns can be poor.

- **Pattern Evaluation-the interestingness problem:** A data mining system can uncover thousands of patterns. Many of the patterns discovered may be uninteresting to the given user, either because they represent common knowledge or lack novelty. Several challenges remain regarding the development of techniques to assess the interestingness of discovered patterns.
- **Incorporation of background Knowledge:** Background knowledge or information regarding the domain under study, may be used to guide the discovery process and allow the discovered patterns to be expressed in concise terms and at different levels of abstraction.

- **2. Performance Issues:** These include efficiency, scalability and parallelization of data mining algorithms.
- **Efficiency and scalability of data mining algorithms:**
- To effectively extract information from a huge amount of data in databases, data mining algorithms must be efficient and scalable.
- The running time of data mining algorithms must be predictable and acceptable in large databases.
- **Parallel, distributed and incremental mining algorithms:**

The huge size of many databases, the wide distribution of data and the computational complexity of some data mining methods are factors motivating the development of parallel and distributed data mining algorithms. Such algorithms divide the data into partitions which are processed in parallel. The results from the partitions are then merged.

3. Issues relating to the diversity of database types:

Handling of relational and complex types of data:

The databases may contain complex data objects, hypertext, multimedia data, spatial data, temporal data. It is unrealistic to expect one system to mine all kinds of data, given the diversity of data types and different goals of data mining. Specific data mining systems should be constructed for mining specific kinds of data.

- **Mining information from heterogenous databases and global information systems**:

LAN and WAN computer networks connect many sources of data, forming huge, distributed and heterogeneous databases. The discovery of knowledge from different sources of structured, semistructured and unstructured data with diverse data semantics poses great challenges to data mining.

1.7 DataPreprocessing

- Today’s real-world databases are highly susceptible to noisy, missing, and inconsistent data due to their typically huge size (often several gigabytes or more) and their likely origin from multiple, heterogenous sources. Low-quality data will lead to low-quality mining results.
- There are a number of data preprocessing techniques.

1. Data cleaning
2. Data integration
3. Data transformation
4. Data reduction

Data Cleaning:

Real-world data tend to be incomplete, noisy, and inconsistent. Data cleaning (or data cleansing) routines attempt to fill in missing values, smooth out noise while identifying outliers, and correct inconsistencies in the data.

Methods for data cleaning:

1 Missing Values

2 Noisy Data

Missing Values:This situation arises when some data is missing in the data. It can be handled in various ways.

Some of them are:

Ignore the tuple: This is usually done when the class label is missing (assuming the mining task involves classification). This method is not very effective, unless the tuple contains several attributes with missing values. It is especially poor when the percentage of missing values per attribute varies considerably.

Fill in the missing value manually: In general, this approach is time-consuming and may not be feasible given a large data set with many missing values.

Use a global constant to fill in the missing value: Replace all missing attribute values by the same constant, such as a label like “Unknown” or $-\infty$. If missing values are replaced by, say, “Unknown,” then the mining program may mistakenly think that they form an interesting concept, since they all have a value in common—that of “Unknown.” Hence, although this method is simple, it is not foolproof.

Use the attribute mean to fill in the missing value: For example, suppose that the average income of AllElectronics customers is $56,000. Use this value to replace the missing value for income.

Use the attribute mean for all samples belonging to the same class as the given tuple: For example, if classifying customers according to credit risk, replace the missing value with the average income value for customers in the same credit risk category as that of the given tuple.

Use the most probable value to fill in the missing value: This may be determined with regression, inference-based tools using a Bayesian formalism, or decision tree induction. For example, using the other customer attributes in your data set, you may construct a decision tree to predict the missing values for income.

2. **Noisy data:**

Noise is a random error or variance in a measured variable. Given a numerical attribute such as, say, price, how can we “smooth” out the data to remove the noise.

The following are data smoothing techniques:

a) **Binning:** Binning methods smooth a sorted data value by consulting its “neighborhood,” that is, the values around it. The sorted values are distributed into a number of “buckets,” or bins. In this example, the data for price are first sorted and then partitioned into equal-frequency bins of size 3 (i.e., each bin contains three values). In smoothing by bin means, each value in a bin is replaced by the mean value of the bin. For example, the mean of the values 4, 8, and 15 in Bin 1 is Therefore, each original value in this bin is replaced by the value 9. Similarly, smoothing by bin medians can be employed, in which each bin value is replaced by the bin median. In smoothing by bin boundaries, the minimum and maximum values in a given bin are identified as the bin boundaries. Each bin value is then replaced by the closest boundary value.

b) **Regression:** Data can be smoothed by fitting the data to a function, such as with regression. Linear regression involves finding the “best” line to fit two attributes (or variables), so that one attribute can be used to predict the other. Multiple linear regression is an extension of linear regression, where more than two attributes are involved and the data are fit to a multidimensional surface.

3. **Clustering:** Outliers may be detected by clustering, where similar values are organized into groups, or “clusters.” Intuitively, values that fall outside of the set of clusters may be considered outliers

2.Data Integration:

Combines data from multiple sources into a coherent data store, as in data warehousing. These sources may include multiple databases, data cubes, or flat files. There are a number of issues to consider during data integration.

Schema integration and object matching can be tricky. How can equivalent real-world entities from multiple data sources be matched up? This is referred to as the entity identification problem. For example, how can the data analyst or the computer be sure that customer id in one database and cust number in another refer to the same attribute?

Redundancy is another important issue. An attribute (such as annual revenue, for instance) may be redundant if it can be "derived" from another attribute or set of attributes

3.Data Transformation:

In data transformation, the data are transformed or consolidated into forms appropriate for mining. Data transformation can involve the following:

Smoothing: which works to remove noise from the data. Such techniques include binning, regression, and clustering.

Aggregation, where summary or aggregation operations are applied to the data. For example, the daily sales data may be aggregated so as to compute monthly and annual total amounts. This step is typically used in constructing a data cube for analysis of the data at multiple granularities.

Generalization of the data, where low-level or "primitive" (raw) data are replaced by higher-level concepts through the use of concept hierarchies. For example, categorical 2.4 Data Integration and Transformation 71 attributes, like street, can be generalized to higher-level concepts, like city or country. Similarly, values for numerical attributes, like age, may be mapped to higher-level concepts, like youth, middle-aged, and senior.

Normalization, where the attribute data are scaled so as to fall within a small specified range, such as −1.0 to 1.0, or 0.0 to 1.0.

Attribute construction (or feature construction), where new attributes are constructed and added from the given set of attributes to help the mining process.

4. Data Reduction

Data reduction techniques can be applied to obtain a reduced representation of the data set that is much smaller in volume, yet closely maintains the integrity of the original data. That is, mining on the reduced data set should be more efficient yet produce the same (or almost the same) analytical results. Strategies for data reduction include the following:

1. **Data cube aggregation**, where aggregation operations are applied to the data in the construction of a data cube.
2. **Attribute subset selection**, where irrelevant, weakly relevant, or redundant attributes or dimensions may be detected and removed.
3. **Dimensionality reduction**, where encoding mechanisms are used to reduce the data set size.
4. **Numerosity reduction**, where the data are replaced or estimated by alternative, smaller data representations such as parametric models (which need store only the model parameters instead of the actual data) or nonparametric methods such as clustering, sampling, and the use of histograms.
5. **Discretization and concept hierarchy generation**, where raw data values for attributes are replaced by ranges or higher conceptual levels. Data discretization is a form of numerosity reduction that is very useful for the automatic generation of concept hierarchies. Discretization and concept hierarchy generation are powerful tools for data mining, in that they allow the mining of data at multiple levels of abstraction.

   Data Cube Aggregation

   Imagine that you have collected the data for your analysis. These data consist of the AllElectronics sales per quarter, for the years 2002 to 2004. You are, however, interested in the annual sales (total per year), rather than the total per quarter. Thus the data can be aggregated so that the resulting data summarize the total sales per year instead of per quarter. The resulting data set is smaller in volume, without loss of information necessary for the analysis task. Data cubes store multidimensional aggregated information. Each cell holds an aggregate data value, corresponding to the data point in multidimensional space.

Concept hierarchies may exist for each attribute, allowing the analysis of data at multiple levels of abstraction

Data cubes provide fast access to precomputed, summarized data, thereby benefiting on-line analytical processing as well as data mining.

- The cube created at the lowest level of abstraction is referred to as the base cuboid. The base cuboid should correspond to an individual entity of interest, such as sales or customer. In other words, the lowest level should be usable, or useful for the analysis. A cube at the highest level of abstraction is the apex cuboid.
- **Attribute Subset Selection**
- Data sets for analysis may contain hundreds of attributes, many of which may be irrelevant to the mining task or redundant.
- Although it may be possible for a domain expert to pick out some of the useful attributes, this can be a difficult and time-consuming task, especially when the behavior of the data is not well known.
- In addition, the added volume of irrelevant or redundant attributes can slow down the mining process.
- Attribute subset selection reduces the data set size by removing irrelevant or redundant attributes (or dimensions). The goal of attribute subset selection is to find a minimum 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.
- Heuristic methods that explore a reduced search space are commonly used for attribute subset selection.
- Basic heuristic methods of attribute subset selection include the following techniques,

1. Stepwise forward selection: The procedure starts with an empty set of attributes as the reduced set. The best of the original attributes is determined and added to the reduced set. At each subsequent iteration or step, the best of the remaining original attributes is added to the set.

2. Stepwise backward elimination: The procedure starts with the full set of attributes. At each step, it removes the worst attribute remaining in the set.

3. Combination of forward selection and backward elimination: The stepwise forward selection and backward elimination methods can be combined so that, at each step, the procedure selects the best attribute and removes the worst from among the remaining attributes.

4. Decision tree induction: Decision tree algorithms, such as ID3, C4.5, and CART, were originally intended for classification. Decision tree induction constructs a flowchart like structure where each internal (nonleaf) node denotes a test on an attribute, each branch corresponds to an outcome of the test, and each external (leaf) node denotes a class prediction.

- At each node, the algorithm chooses the "best" attribute to partition the data into individual classes. When decision tree induction is used for attribute subset selection, a tree is constructedfromthegivendata.Allattributesthatdonotappearinthetreeareassumed to be irrelevant. The set of attributes appearing in the tree form the reduced subset of attributes.
- **Dimensionality Reduction**
- In dimensionality reduction, data encoding or transformations are applied so as to obtain a reduced or "compressed" representation of the original data. If the original data can be reconstructed from the compressed data without any loss of information, the data reduction is called lossless. If, instead, we can reconstruct only an approximation of the original data, then the data reduction is called lossy

- The effective method of lossy dimensionality reduction is

Wavelet transforms

- **Wavelet transforms**
- The discrete wavelet transform (DWT) is a linear signal processing technique that, when applied to a data vector X, transforms it to a numerically different vector, X ‘ , of wavelet coefficients. The two vectors are of the same length. When applying this technique to data reduction, we consider each tuple as an n-dimensional data vector, that is, X = (x1,x2,...,xn), depicting n measurements made on the tuple from n database attributes.“How can this technique be useful for data reduction if the wavelet transformed data are of the same length as the original data?” The usefulness lies in the fact that the wavelet transformed data can be truncated. A compressed approximation of the data can be retained by storing only a small fraction of the strongest of the wavelet coefficients. For example, all wavelet coefficients larger than some user-specified threshold can be retained. All other coefficients are set to 0. The resulting data representation is therefore very sparse, so that operations that can take advantage of data sparsity are computationally very fast if performed in wavelet space.
- **Numerosity Reduction**
- “Can we reduce the data volume by choosing alternative, ‘smaller’ forms of data representation?” Techniques of numerosity reduction can indeed be applied for this purpose. These techniques may be parametric or nonparametric. For parametric methods, a model is used to estimate the data, so that typically only the data parameters need to be stored, instead of the actual data.
- The numerosity reduction techniques are
- Regression and Log-Linear Models
- Regression and log-linear models can be used to approximate the given data. In (simple) linear regression, the data are modeled to fit a straight line. For example, a random variable, y (called a response variable), can be modeled as a linear function of another random variable, x (called a predictor variable), with the equation y = wx+b, where the variance of y is assumed to be constant. In the context of data mining, x and y are numerical database attributes. The coefficients, wand b (called regression coefficients), specify the slope of the line and the Y-intercept, respectively.
- **Log-linear models** approximate discrete multidimensional probability distributions. Given a set of tuples in n dimensions (e.g., described by n attributes), we can consider each tuple as a point in an n-dimensional space. Log-linear models can be used to estimate the probability of each point in a multidimensional space for a set of discretized attributes, based on a smaller subset of dimensional combinations. This allows a higher-dimensional data space to be constructed from lower dimensional spaces. Log-linear models are therefore also useful for dimensionality reduction

- **Histograms** use binning to approximate data distributions and are a popular form of data reduction. A histogram for an attribute, A, partitions the data distribution of A into disjoint subsets, or buckets. If each bucket represents only a single attribute-value/frequency pair, the buckets are called singleton buckets. Often, buckets instead represent continuous ranges for the given attribute.
- **Clustering** techniques consider data tuples as objects. They partition the objects into groups or clusters, so that objects within a cluster are “similar” to one another and “dissimilar” to objects in other clusters. Similarity is commonly defined in terms of how “close” the objects are in space, based on a distance function. The “quality” of a cluster may be represented by its diameter, the maximum distance between any two objects in the cluster.
- **Sampling** can be used as a data reduction technique because it allows a large data set to be represented by a much smaller random sample (or subset) of the data.

- **Data Discretization and Concept Hierarchy Generation:**
- Data discretization techniques can be used to reduce the number of values for a given continuous attribute by dividing the range of the attribute into intervals. Interval labels can then be used to replace actual data values. Replacing numerous values of a continuous attribute by a small number of interval labels thereby reduces and simplifies the original data.
- This leads to a concise,easy-to-use,knowledge-levelrepresentationofminingresult Data Discretization and Concept Hierarchy Generation Discretization techniques can be categorized based on how the discretization is performed, such as whether it uses class information or which direction it proceeds (i.e., top-down vs. bottom-up). If the discretization process uses class information, then we say it is supervised discretization. Otherwise, it is unsupervised.
- If the process starts by first finding one or a few points (called split points or cut points) to split the entire attribute range, and then repeats this recursively on the resulting intervals, it is called top-down discretization or splitting. This contrasts with bottom-up discretization or merging, which starts by considering all of the continuous values as potential split-points, removes some by merging neighborhood values to form intervals, and then recursively applies this process to the resulting intervals.
- A concept hierarchy for a given numerical attribute defines a discretization of the attribute. Concept hierarchies can be used to reduce the data by collecting and replacing low-level concepts (such as numerical values for the attribute age) with higher-level concepts (such as youth, middle-aged, or senior).

1.8 Classification of Data Mining Systems

Classification according to the kinds of databases mined: A data mining system can be classified according to the kinds of databases mined. Database systems can be classified according to different criteria (such as data models, or the types of data or applications involved), each of which may require its own data mining technique.

- For instance, if classifying according to data models, we may have a relational, transactional, object-relational, or data warehouse mining system. If classifying according to the special types of data handled, we may have a spatial, time-series, text, stream data, multimedia data mining system, or a World Wide Web mining system

Classification according to the kinds of knowledge mined: Data mining systems can be categorized according to the kinds of knowledge they mine, that is, based on data mining functionalities, such as characterization, discrimination, association and correlation analysis, classification, prediction, clustering, outlier analysis, and evolution analysis. A comprehensive data mining system usually provides multiple and/or integrated data mining functionalities.

- Data mining systems can also be categorized as those that mine data regularities (commonly occurring patterns) versus those that mine data irregularities (such as exceptions, or outliers).

Classification according to the kinds of techniques utilized: Data mining systems can be categorized according to the underlying data mining techniques employed. These techniques can be described according to the degree of user interaction involved (e.g., autonomous systems, interactive exploratory systems, query-driven systems) or the methods of data analysis employed (e.g., database-oriented or data warehouse– oriented techniques, machine learning, statistics, visualization, pattern recognition, neural networks, and so on). A sophisticated data mining system will often adopt multiple data mining techniques or work out an effective, integrated technique that combines the merits of a few individual approaches

- **Classification according to the applications adapted:** Data mining systems can also be categorized according to the applications they adapt. For example, data mining systems may be tailored specifically for finance, telecommunications, DNA, stock markets, e-mail, and so on. Different applications often require the integration of application-specific methods. Therefore, a generic, all-purpose data mining system may not fit domain-specific mining tasks.

ASSIGNMENT QUESTIONS

1) Explain Data Mining as a KDD process.
2) Explain the functionalities of data mining.
3) Explain data pre-processing techniques.
4) Explain data mining task primitives

OBJECTIVE QUESTIONS

1. The steps involved in knowledge discovery process are data cleaning, Data integration,Data selection,Data transformation, Data mining,Pattern evaluation,Knowledge presentation.
2. Attribute subset selection reduces the data set size by removing irrelevant or redundant attributes.
3. The steps of KDD process, in which the several data sources are combined refers to data integration.
4. Normalization is a strategy where the attribute data are scaled so as to fall within a smaller range.
5. Outlier Analysis can be defined as the data object which does not comply with the general behavior.
6. KDD in data mining refers to as Knowledge discovery of data.
7. The mapping or classification of a class with some predefined group or class is known as Data Discrimination

NPTEL LINKS

https://nptel.ac.in/courses/106/105/106105174/

Unit -II

2. Mining Frequent Patterns, Associations, and Correlations

2.1 Association Rule Mining: Mining Frequent Patterns
2.2 Associations andcorrelations
2.3 Mining Methods
2.4 Mining Various kinds of Association Rules
2.5 CorrelationAnalysis
2.6 ConstraintbasedAssociationmining.
2.7 GraphPatternMining

2.1 Association Rule Mining: Mining Frequent Patterns

- Frequent patterns are patterns (such as itemsets, subsequences, or substructures) that appear in a data set frequently. For example, a set of items, such as milk and bread, that appear frequently together in a transaction data set is a frequent itemset.

- A subsequence, such as buying first a PC, then a digital camera, and then a memory card, if it occurs frequently in a shopping history database, is a (frequent) sequential pattern.
- A substructure can refer to different structural forms, such as subgraphs, subtrees, or sublattices, which may be combined with itemsets or subsequences. If a substructure occurs frequently, it is called a (frequent) structured pattern.
- Finding such frequent patterns plays an essential role in mining associations, correlations, and many other interesting relationships among data.
- Moreover, it helps in data classification, clustering, and other data mining tasks as well.

- How can we find frequent itemsets from large amounts of data, where the data are either transactional or relational?
- With massive amounts of data continuously being collected and stored, many industries are becoming interested in mining such patterns from their databases. The discovery of interesting correlation relationships among huge amounts of business transaction records can help in many business decision-making processes, such as catalog design, cross-marketing, and customer shopping behavior analysis.
- **Frequent Itemsets, Closed Itemsets, and Association Rules:**
- LetI = {I1, I2,..., Im} be a set of items.
- Let D, the task-relevant data, be a set of database transactions where each transaction T is a set of items such that $T \subseteq I$.
- Each transaction is associated with an identifier, called TID. Let A be a set of items.
- A transaction T is said to contain A if and only if $A \subseteq T$. An association rule is an implication of the form $A \Rightarrow B$, where $A \subset I$, $B \subset I$, and $A \cap B = \varphi$.
- The rule $A \Rightarrow B$ holds in the transaction set D with support s, where s is the percentage of transactions in D that contain $A \cup B$ (i.e., the union of sets A and B, or say, both A and B).
- This is taken to be the probability, $P(A \cup B)$.
- The rule $A \Rightarrow B$ has confidence c in the transaction set D, where c is the percentage of transactions in D containing A that also contain B. This is taken to be the conditional probability, $P(B|A)$.
- That is, $\text{support}(A \Rightarrow B) = P(A \cup B)$
- $\text{confidence}(A \Rightarrow B) = P(B|A)$
- A set of items is referred to as an itemset.
- An itemset that contains k items is a k-itemset.
- The set {computer, antivirus software} is a 2-itemset.
- The occurrence frequency of an itemset is the number of transactions that contain the itemset. This is also known, simply, as the frequency, support count, or count of the itemset.
- **Frequent Pattern Mining**

Market basket analysis is just one form of frequent pattern mining. In fact, there are many kinds of frequent patterns, association rules, and correlation relationships. Frequent pattern mining can be classified in various ways, based on the following criteria:

Based on the completeness of patterns to be mined: We can mine the complete set of frequent itemsets, the closed frequent itemsets, and the maximal frequent itemsets, given a minimum support threshold. We can also mine constrained frequent itemsets (i.e., those that satisfy a set of user-defined constraints), approximate frequent itemsets (i.e., those that derive only approximate support counts for the mined frequent itemset).

Based on the levels of abstraction involved in the rule set: Some methods for association rule mining can find rules at differing levels of abstraction. For example, suppose that a set of association rules mined includes the following rules where X is a variable representing a customer:

buys(X, “computer”) ⇒ buys(X, “HP printer”)
buys(X, “laptop computer”) ⇒ buys(X, “HP printer”)

- the items bought are referenced at different levels of abstraction (e.g., “computer” is a higher-level abstraction of “laptop computer”). We refer to the rule set mined as consisting of multilevel association rules.
- **Based on the number of data dimensions involved in the rule:** If the items or attributes in an association rule reference only one dimension, then it is a single-dimensional association rule.
- buys(X, “computer”) ⇒ buys(X, “antivirus software”)

The above one is single-dimensional association rules because they each refer to only one dimension, buys.
If a rule references two or more dimensions, such as the dimensions age, income, and buys, then it is a multidimensional association rule. The following rule is an example of a multidimensional rule:
age(X, “30...39”)∧income(X, “42K...48K”)⇒buys(X, “high resolution TV”)

- **Based on the types of values handled in the rule:** If a rule involves associations between the presence or absence of items, it is a Boolean association rule. If a rule describes associations between quantitative items or attributes, then it is a quantitative association rule. In these rules, quantitative values for items or attributes are partitioned into intervals.
- **Based on the kinds of rules to be mined:** Frequent pattern analysis can generate various kinds of rules and other interesting relationships. Association rules are the most popular kind of rules generated from frequent patterns. Typically, such mining can generate a large number of rules, many of which are redundant or do not indicate a correlation relationship among itemsets.
- **Based on the kinds of patterns to be mined:** Many kinds of frequent patterns can be mined from different kinds of data sets. Our focus is on frequent itemset mining, that is, the mining of frequent itemsets (sets of items) from transactional or relational data sets. However, other kinds of frequent patterns can be found from other kinds of data sets. Sequential pattern mining searches for frequent subsequences in a sequence data set, where a sequence records an ordering of events. For example, with sequential pattern mining, we can study the order in which items are frequently purchased.
- In general, association rule mining can be viewed as a two-step process:
- 1. Find all frequent itemsets: By definition, each of these itemsets will occur at least as frequently as a predetermined minimum support count, min sup.
- 2. Generate strong association rules from the frequent itemsets: By definition, these rules must satisfy minimum support and minimum confidence.

Efficient and Scalable Frequent Itemset Mining Methods

2.2 The methods used for finding frequent itemsets are

1.The Apriori Algorithm: Finding Frequent Itemsets Using Candidate Generation
2. Mining Frequent Itemsets without Candidate Generation

- Apriori is a seminal algorithm proposed by R. Agrawal and R. Srikant in 1994 for mining frequent itemsets for Boolean association rules. The name of the algorithm is based on Efficient and Scalable Frequent Itemset Mining Methods the fact that the algorithm uses prior knowledge of frequent itemset properties, as we shall see following. Apriori employs an iterative approach known as a level-wise search, where k-itemsets are used to explore (k+1)-itemsets.
- First, the set of frequent 1-itemsets is found by scanning the database to accumulate the

count for each item, and collecting those items that satisfy minimum support. The resulting set is denoted L1. Next, L1 is used to find L2, the set of frequent 2-itemsets, which is used to find L3, and so on, until no more frequent k-itemsets can be found. The finding of each Lk requires one full scan of the database.

- A two-step process is followed, consisting of join and prune actions.

1. **The join step:**To find Lk, a set of candidate k-itemsets is generated by joining Lk−1 with itself. This set of candidates is denoted Ck. Let l1 and l2 be itemsets in Lk−1. The notation li [j] refers to the jth item in li (e.g., l1[k−2] refers to the second to the last item in l1). By convention, Apriori assumes that items within a transaction or itemset are sorted in lexicographic order. For the (k −1)-itemset, li , this means that the items are sorted such that li [1] < li [2] < ... < li [k − 1]. The join, Lk−1 on Lk−1, is performed, where members of Lk−1 are joinable if their first (k − 2) items are in common. That is, members l1 and l2 of Lk−1 are joined if (l1[1] = l2[1]) ∧ (l1[2] = l2[2]) ∧...∧(l1[k−2] = l2[k−2]) ∧(l1[k−1] < l2[k−1]). The condition l1[k−1] < l2[k−1]simply ensures that no duplicates are generated. The resulting itemset formed by joining l1 and l2 is l1[1], l1[2],..., l1[k −2], l1[k −1], l2[k −1].

- 2. **The prune step**:Ck is a superset of Lk, that is, its members may or may not be frequent, but all of the frequent k-itemsets are included inCk. A scan of the database to determine the count of each candidate in Ck would result in the determination of Lk (i.e., all candidates having a count no less than the minimum support count are frequent by definition, and therefore belong to Lk). Ck, however, can be huge, and so this could involve heavy computation. To reduce the size of Ck, the Apriori property is used as follows. Any (k − 1)-itemset that is not frequent cannot be a subset of a frequent k-itemset. Hence, if any (k − 1)-subset of a candidate k-itemset is not in Lk−1, then the candidate cannot be frequent either and so can be removed from Ck
- **The Apriori Algorithm:**

In the first iteration of the algorithm, each item is a member of the set of candidate 1-itemsets, C1.

1. The algorithm simply scans all of the transactions in order to count the number of occurrences of each item.

2. Suppose that the minimum support count required is 2, that is, min sup = 2.

The set of frequent 1-itemsets, L1, can then be determined. It consists of the candidate 1-itemsets satisfying minimum support. In our example, all of the candidates in C1 satisfy minimum support.

3. To discover the set of frequent 2-itemsets, L2, the algorithm uses the join L1 on L1 to generate a candidate set of 2-itemsets, C2.

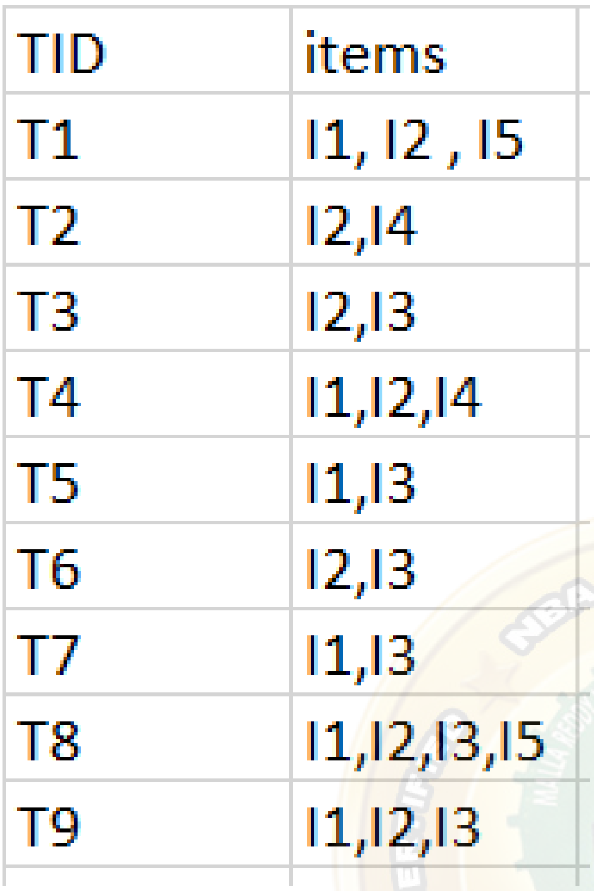
4. Next, the transactions in D are scanned and the support count of each candidate itemset inC2 is accumulated.

5. The set of frequent 2-itemsets, L2, is then determined, consisting of those candidate 2-itemsets in C2 having minimum support.

6. From the join step, we first getC3 = L2 on L2 = {{I1, I2, I3}, {I1, I2, I5}, {I1, I3, I5}, {I2, I3, I4}, {I2, I3, I5}, {I2, I4, I5}}. Based on the Apriori property that all subsets of a frequent itemset must also be frequent, we can determine that the four latter candidates cannot possibly be frequent. We therefore remove them from C3, thereby saving the effort of unnecessarily obtaining their counts during the subsequent scan of D to determine L3.

7. The transactions in D are scanned in order to determine L3, consisting of those candidate 3-itemsets in C3 having minimum support.

8. The algorithm uses L3 on L3 to generate a candidate set of 4-itemsets, C4. Although the join results in {{I1, I2, I3, I5}}, this itemset is pruned because its subset {{I2, I3, I5}} is not frequent. Thus, C4 = φ, and the algorithm terminates, having found all of the frequent itemsets.



TID	items
T1	I1, I2 , I5
T2	I2,I4
T3	I2,I3
T4	I1,I2,I4
T5	I1,I3
T6	I2,I3
T7	I1,I3
T8	I1,I2,I3,I5
T9	I1,I2,I3

- **Generating Association Rules from Frequent Itemsets:**
- To generate strong association rules from them (where strong association rules satisfy both minimum support and minimum confidence). This can be done using Equation for confidence,
- confidence(A ⇒ B) = P(B|A) = support count(A∪B) /support count(A)
- The conditional probability is expressed in terms of itemset support count, where support count(A∪B) is the number of transactions containing the itemsets A∪B, and support count(A) is the number of transactions containing the itemset A.
- Based on this equation, association rules can be generated as follows:
- For each frequent itemset l, generate all nonempty subsets of l.
- For every nonempty subset s of l, output the rule “s ⇒ (l − s)” if support count(l) /support count(s) ≥ min conf, where min conf is the minimum confidence threshold.
- Suppose the data contain the frequent itemset l = {I1, I2, I5}.
- What are the association rules that can be generated from l?
- The nonempty subsets of l are {I1, I2}, {I1, I5}, {I2, I5}, {I1}, {I2}, and {I5}.
- The resulting association rules are as shown below, each listed with its confidence:
- I1∧I2 ⇒ I5, confidence = 2/4 = 50%
- I1∧I5 ⇒ I2, confidence = 2/2 = 100%
- I2∧I5 ⇒ I1, confidence = 2/2 = 100%
- I1 ⇒ I2∧I5, confidence = 2/6 = 33%
- I2 ⇒ I1∧I5, confidence = 2/7 = 29%
- I5 ⇒ I1∧I2, confidence = 2/2 = 100%
- If the minimum confidence threshold is, say, 70%, then only the second, third, and last rules

above are output, because these are the only ones generated that are strong.

- **Mining Frequent Itemsets without Candidate Generation:**
- The Apriori candidate generate-and-test method significantly reduces the size of candidate sets, leading to good performance gain. However it can suffer from two nontrivial costs:

1. It may need to generate a huge number of candidate set.
2. It may need to repeatedly scan the database and check a large set of candidates by pattern matching. It is costly to go over each transaction in the database to determine the support of the candidate itemsets.

“Can we design a method that mines the complete set of frequent itemsets without candidate generation?”

- An interesting method in this attempt is called frequent-pattern growth, or simply FP-growth, which adopts a divide-and-conquer strategy as follows:
- First, it compresses the database representing frequent items into a frequent-pattern tree, or FP-tree, which retains the itemset association information.
- It then divides the compressed database into a set of conditional databases (a special kind of projected database), each associated with one frequent item or “pattern fragment”.
- Then it mines each such database separately.

TID	List of item Ids
T100	I1, I2, I5
T200	I2, I4
T300	I2, I3
T400	I1, I2, I4
T500	I1, I3
T600	I2, I3
T700	I1, I3
T800	I1, I2, I3, I5
T900	I1, I2, I3

- The first scan of the database is the same as Apriori, which derives the set of frequent items (1-itemsets) and their support counts (frequencies). Let the minimum support count be 2. The set of frequent items is sorted in the order of descending support count. This resulting set or list is denoted L.
- Thus, we have L ={{I2: 7}, {I1: 6}, {I3: 6}, {I4: 2}, {I5: 2}}.
- An FP-tree is then constructed as follows. First, create the root of the tree, labeled with “null.” Scan database D a second time. The items in each transaction are processed in L order (i.e., sorted according to descending support count), and a branch is created for each transaction.
- For example, the scan of the first transaction, “T100: I1, I2, I5,” which contains three items (I2, I1, I5 in L order), leads to the construction of the first branch of the tree with three nodes, (I2: 1), (I1:1), and (I5: 1), where I2 is linked as a child of the root, I1 is linked to I2, and I5 is linked to I1.
- The second transaction, T200, contains the items I2 and I4 in L order, which would result in a branch where I2 is linked to the root and I4 is linked to I2. However, this branch would share a common prefix, I2, with the existing path for T100.
- Therefore, we instead increment the count of the I2 node by 1, and create a new node,(I4: 1), which is linked as a child of(I2: 2)
- In general, when considering the branch to be added for a transaction, the count of each node along a common prefix is incremented by 1, and nodes for the items following the prefix are created and linked accordingly.

- The FP-tree is mined as follows. Start from each frequent length-1 pattern (as an initial suffix pattern), construct its conditional pattern base (a “subdatabase,” which consists of the set of prefix paths in the FP-tree co-occurring with the suffix pattern), then construct its (conditional) FP-tree, and perform mining recursively on such a tree. The pattern growth is achieved by the concatenation of the suffix pattern with the frequent patterns generated from a conditional FP-tree.
- Mining of the FP-tree is summarized as follows:
- We first consider I5, which is the last item in L, rather than the first. The reason for starting at the end of the list will become apparent as we explain the FP-tree mining process. I5 occurs in two branches of the FP-tree.
- The paths formed by these branches are (I2, I1, I5: 1) and (I2, I1, I3, I5: 1). Therefore, considering I5 as a suffix, its corresponding two prefix paths are (I2, I1: 1) and (I2, I1, I3: 1), which form its conditional pattern base.
- Its conditional FP-tree contains only a single path, hI2: 2, I1: 2i; I3 is not included because its support count of 1 is less than the minimum support count. The single path generates all the combinations of frequent patterns: {I2, I5: 2}, {I1, I5: 2}, {I2, I1, I5: 2}.
- For I4, its two prefix paths form the conditional pattern base, {{I2 I1: 1}, {I2: 1}}, which generates a single-node conditional FP-tree, (I2: 2), and derives one frequent pattern, {I2, I1: 2}. Notice that although I5 follows I4 in the first branch, there is no need to include I5 in the analysis here because any frequent pattern involving I5 is analyzed in the examination of I5.
- Similar to the above analysis, I3’s conditional pattern base is {{I2, I1: 2}, {I2: 2}, {I1: 2}}. Its conditional FP-tree has two branches, (I2: 4, I1: 2) and (I1: 2), generates the set of patterns, {{I2, I3: 4}, {I1, I3: 4}, {I2, I1, I3: 2}}. Finally, I1’s conditional pattern base is {{I2: 4}}, whose FP-tree contains only one node, (I2: 4), which generates one frequent pattern, {I2, I1: 4}.

2.4 Mining Various kinds of Association Rules

Kinds of Association Rules

1. Multilevel association rules,

2. Multidimensional association rules,

3. Quantitative association rules.

Multilevel association rules involve concepts at different levels of abstraction. Multidimensional association rules involve more than one dimension or predicate (e.g., rules relating what a customer buys as well as the customer’s age.) Quantitative association rules involve numeric attributes that have an implicit ordering among values (e.g., age).

- **Mining Multilevel Association Rules**
- For many applications, it is difficult to find strong associations among data items at low or primitive levels of abstraction due to the sparsity of data at those levels. Strong associations discovered at high levels of abstraction may represent commonsense knowledge.

EX:

TID	Items Purchased
T100	IBM-ThinkPad-T40/2373, HP-Photosmart-7660

T200	Microsoft-Office-Professional-2003, Microsoft-Plus!- Digital-Media
T300	Logitech-MX700-Cordless-Mouse, Fellowes-Wrist-Rest
T400	Dell-Dimension-XPS, Canon-PowerShot-S400
T500	IBM-ThinkPad-R40/P4M, Symantec-Norton-Antivirus-2003

The concept hierarchy has five levels, respectively referred to as levels 0 to 4, starting with level 0 at the root node for all (the most general abstraction level).

Here, level 1 includes computer, software, printer&camera, and computer accessory, level 2 includes laptop computer, desktop computer, office software, antivirus software, . . . , and level 3 includes IBM desktop computer, . . . , Microsoft office software, and so on.

Level 4 is the most specific abstraction level of this hierarchy.

It consists of the raw data values.

It is difficult to find interesting purchase patterns at such raw or primitive-level data. For instance, if “IBM-ThinkPad-R40/P4M” or “Symantec-Norton-Antivirus-2003” each occurs in a very small fraction of the transactions, then it can be difficult to find strong associations involving these specific items. Few people may buy these items together, making it unlikely that the itemset will satisfy minimum support.

- Association rules generated from mining data at multiple levels of abstraction are called multiple-level or multilevel association rules. Multilevel association rules can be mined efficiently using concept hierarchies under a support-confidence framework.
- A number of variations to this approach are

1. **Using uniform minimum support for all levels:** When a uniform minimum support threshold is used, the search procedure is simplified. The method is also simple in that users are required to specify only one minimum support threshold.

An Apriori-like optimization technique can be adopted, based on the knowledge that an ancestor is a superset of its descendants: The search avoids examining itemsets containing any item whose ancestors do not have minimum support.

2. Using reduced minimum support at lower levels (referred to as reduced support): Each level of abstraction has its own minimum support threshold. The deeper the level of abstraction, the smaller the corresponding threshold is. For ex, the minimum support thresholds for levels 1 and 2 are 5% and 3%, respectively.

3. Using item or group-based minimum support (referred to as group-based support): Because users or experts often have insight as to which groups are more important than others, it is sometimes more desirable to set up user-specific, item, or group based minimal support thresholds when mining multilevel rules. For example, a user could set up the minimum support thresholds based on product price, or on items of interest, such as by setting particularly low support thresholds for laptop computers and flash drives in order to pay particular attention to the association patterns containing items in these categories

- **Mining Multidimensional Association Rules from Relational Databases and Data Warehouse**

For instance, in mining our AllElectronics database, we may discover the Boolean association rule buys(X, “digital camera”) ⇒ buys(X, “HP printer”)

we can refer to the above rule as a singledimensional or intradimensional association rule because it contains a single distinct predicate (e.g., buys) with multiple occurrences (i.e., the predicate occurs more than once within the rule).

Suppose, however, that rather than using a transactional database, sales and related information are stored in a relational database or data warehouse. Such data stores are multidimensional, by definition.

For instance, in addition to keeping track of the items purchased in sales transactions, a relational database may record other attributes associated with the items, such as the quantity purchased or the price, or the branch location of the sale.

Additional relational information regarding the customers who purchased the items, such as customer age, occupation, credit rating, income, and address, may also be stored.

- Considering each database attribute or warehouse dimension as a predicate, we can therefore mine association rules containing multiple predicates, such as
- age(X, “20...29”)∧occupation(X, “student”)⇒buys(X, “laptop”)

Association rules that involve two or more dimensions or predicates can be referred to as multidimensional association rules.

- **Mining Quantitative Association Rules**

Quantitative association rules are multidimensional association rules in which the numeric attributes are dynamically discretized during the mining process so as to satisfy some mining criteria, such as maximizing the confidence or compactness of the rules mined**.**

We focus specifically on how to mine quantitative association rules having two quantitative attributes on the left-hand side of the rule and one categorical attribute on the right-hand side of the rule. That is,

Aquan1 ∧Aquan2 ⇒Acat

where Aquan1 and Aquan2 are tests on quantitative attribute intervals (where the intervals are dynamically determined), and Acat tests a categorical attribute from the task-relevant data. Such rules have been referred to as two-dimensional quantitative association rules, because they contain two quantitative dimensions. For instance, suppose you are curious about the association relationship between pairs of quantitative attributes, like customer age and income, and the type of television (such as high-definition TV, i.e., HDTV) that customers like to buy. An example of such a 2-D quantitative association rule is

age(X, “30...39”)∧income(X, “42K...48K”)⇒buys(X, “HDTV”)

- “How can we find such rules?”
- An approach used in a system called ARCS (Association Rule Clustering System), which borrows ideas from image processing. Essentially, this approach maps pairs of quantitative attributes onto a 2-D grid for tuples satisfying a given categorical attribute condition. The grid is then searched for clusters of points from which the association rules are generated. The following steps are involved in ARCS:
- **Binning:** Quantitative attributes can have a very wide range of values defining their domain. Just think about how big a 2-D grid would be if we plotted age and income as axes, where

each possible value of age

- was assigned a unique position on one axis, and similarly, each possible value of income was assigned a unique position on the other axis! To keep grids down to a manageable size, we instead partition the ranges of quantitative attributes into intervals. These intervals are dynamic in that they may later be further combined during the mining process. The partitioning process is referred to as binning, that is, where the intervals are considered "bins." Three common binning strategies area as follows:
- Equal-width binning, where the interval size of each bin is the same.
- Equal-frequency binning, where each bin has approximately the same number of tuples assigned to it.
- Clustering-based binning, where clustering is performed on the quantitative attribute to group neighboring points (judged based on various distance measures) into the same bin.
- ARCS uses equal-width binning, where the bin size for each quantitative attribute is input by the user. A 2-D array for each possible bin combination involving both quantitative attributes is created. Each array cell holds the corresponding count distribution for each possible class of the categorical attribute of the rule right-hand side. By creating this data structure, the task-relevant data need only be scanned once. The same 2-D array can be used to generate rules for any value of the categorical attribute, based on the same two quantitative attributes.
- **Finding frequent predicate sets:** Once the 2-D array containing the count distribution for each category is set up, it can be scanned to find the frequent predicate sets.
- **Clustering the association rules:**

age(X, 34)∧income(X, "31K...40K")⇒buys(X, "HDTV")

age(X, 35)∧income(X, "31K...40K")⇒buys(X, "HDTV")

age(X, 34)∧income(X, "41K...50K")⇒buys(X, "HDTV")

age(X, 35)∧income(X, "41K...50K")⇒buys(X, "HDTV").

These rules are quite "close" to one another, forming a rule cluster on the grid. Indeed, the four rules can be combined or "clustered" together to form the following simpler rule, which subsumes and replaces the above four rules:

ARCS employs a clustering algorithm for this purpose. The algorithm scans the grid, searching for rectangular clusters of rules. In this way, bins of the quantitative attributes occurring within a rule cluster may be further combined, and hence further dynamic discretization of the quantitative attributes occurs.

2.2 Association Analysis to Correlation Analysis

The support and confidence measures are insufficient at filtering out uninteresting association rules. To tackle this weakness, a correlation measure can be used to augment the support-confidence framework for association rules. This leads to correlation rules of the form

A ⇒ B [support, confidence. correlation].

That is, a correlation rule is measured not only by its support and confidence but also by the correlation between itemsets A and B. There are many different correlation measures from which to choose various correlation measures to determine which would be good for mining large data sets.

- **Lift** is a simple correlation measure that is given as follows. The occurrence of itemset A is independent of the occurrence of itemset B if P(A ∪ B) = P(A)P(B); otherwise, itemsets A and B are dependent and correlated as events. This definition can easily be extended to more than two itemsets. The lift between the occurrence of A and B can be measured by computing
- lift(A, B) = P(A∪B) /P(A)P(B)
- If the resulting value of Equation is less than 1, then the occurrence of A is negatively correlated with the occurrence of B. If the resulting value is greater than 1, then A and B are positively correlated, meaning that the occurrence of one implies the occurrence of the other. If the resulting value is equal to 1, then A and B are independent and there is no correlation between them.
- The second correlation measure is the χ 2 measure. To compute the

χ 2 value, we take the squared difference between the observed and expected value for a slot (A and B pair) in the contingency table, divided by the expected value. This amount is summed for all slots of the contingency table.

To compute the correlation using χ 2 analysis, we need the observed value and expected value (displayed in parenthesis) for each slot of the contingency table. From the table, we can compute the χ 2 value as follows:

χ 2 = Σ (observed - expected) 2 /expected =

(4,000−4,500) 2 /4,500 + (3,500−3,000) 2 3,000 + (2,000−1,500) 2/ 1,500 + (500−1,000) 2/ 1,000 = 555.6.

- The third measure is all confidence. Given an itemset X = {i1, i2,..., ik}, the all confidence of X is defined as all conf(X) = sup(X) max item sup(X) = sup(X) max{sup(ij)|∀ij∈ X} , (5.24) where max{sup(ij)|∀ij∈ X} is the maximum (single) item support of all the items in X, and hence is called the max item sup of the itemset X. The all confidence of X is the minimal confidence among the set of rules ij → X −ij , where ij∈ X. Given two itemsets A and B, the cosine measure of A and B is defined as
- cosine(A, B) = P(A∪B) /√ P(A)×P(B) = sup(A∪B) /√sup(A)×sup(B)
- The cosine measure can be viewed as a harmonized lift measure: the two formulae are similar except that for cosine, the square root is taken on the product of the probabilities of A and B. This is an important difference, however, because by taking the square root, the cosine value is only influenced by the supports of A, B, and A ∪ B, and not by the total number of transaction.

2.6Constraint based association mining

A data mining process may uncover thousands of rules from a given set of data, most of which end up being unrelated or uninteresting to the users. Often, users have a good sense of which “direction” of mining may lead to interesting patterns and the “form” of the patterns or rules they would like to find. Thus, a good heuristic is to have the users specify such intuition or expectations as constraints to confine the search space. This strategy is known as constraint-based mining. The constraints can include the following:

- **Knowledge type constraints**: These specify the type of knowledge to be mined, such as association or correlation.
- **Data constraints**: These specify the set of task-relevant data.
- **Dimension/level constraints**: These specify the desired dimensions (or attributes) of the data, or levels of the concept hierarchies, to be used in mining.
- **Interestingness constraints**: These specify thresholds on statistical measures of rule interestingness, such as support, confidence, and correlation.
- **Rule constraints**: These specify the form of rules to be mined. Such constraints may be expressed as metarules (rule templates), as the maximum or minimum number of predicates that can occur in the rule antecedent or consequent, or as relationships among attributes, attribute values, and/or aggregates.
- **Metarule-Guided Mining of Association Rules:**

Metarules allow users to specify the syntactic form of rules that they are interested in mining. The rule forms can be used as constraints to help improve the efficiency of the mining process. Metarules may be based on the analyst's experience, expectations, or intuition regarding the data or may be automatically generated based on the database schema.

Suppose that as a market analyst for AllElectronics, you have access to the data describing customers (such as customer age, address, and credit rating) as well as the list of customer transactions. You are interested in finding associations between customer traits and the items that customers buy. However, rather than finding all of the association rules reflecting these relationships, you are particularly interested only in determining which pairs of customer traits promote the sale of office software. A metarule can be used to specify this information describing the form of rules you are interested in finding.

An example of such a metarule is

P1(X, Y)∧P2(X, W)⇒buys(X, "office software"),

- where P1 and P2 are predicate variables that are instantiated to attributes from the given database during the mining process, X is a variable representing a customer, and Y and W take on values of the attributes assigned to P1 and P2, respectively. Typically, a user will specify a list of attributes to be considered for instantiation with P1 and P2. Otherwise, a default set may be used.
- In general, a metarule forms a hypothesis regarding the relationships that the user is interested in probing or confirming. The data mining system can then search for rules that match the given metarule. For instance, Rule matches or complies with Metarule .
- age(X, "30...39")∧income(X, "41K...60K")⇒buys(X, "office software")
- **Constraint Pushing: Mining Guided by Rule Constraints**

Rule constraints specify expected set/subset relationships of the variables in the mined rules, constant initiation of variables, and aggregate functions. Users typically employ their knowledge of the application or data to specify rule constraints for the mining task. These rule constraints may be used together with, or as an alternative to, metarule-guided mining. In this section, we examine rule constraints as to how they can be used to make the mining process more efficient. Let's study an example where rule constraints are used to mine hybrid-dimensional association rules.

A closer look at mining guided by rule constraints. Suppose that AllElectronics has a sales multidimensional database with the following interrelated relations: sales(customer name, item name, TID) lives in(customer name, region, city) item(item name, group, price) transaction(TID, day, month, year) where lives in, item, and transaction are three dimen

2.7Graph Pattern Mining

- Graphs become increasingly important in modeling complicated structures, such as circuits, images, chemical compounds, protein structures, biological networks, social networks, the Web, workflows, and XML documents. Many graph search algorithms have been developed in chemical informatics, computer vision, video indexing, and text retrieval.
- Among the various kinds of graph patterns, frequent substructures are the very basic patterns that can be discovered in a collection of graphs.
- They are useful for characterizing graph sets, discriminating different groups of graphs, classifying and clustering graphs, building graph indices, and facilitating similarity search in graph databases
- **Methods for Mining Frequent Subgraphs**
- We denote the vertex set of a graph g by V(g) and the edge set by E(g). A label function, L, maps a vertex or an edge to a label. A graph g is a subgraph of another graph g 0 if there exists a subgraph isomorphism from g to g 0 . Given a labeled graph data set, D = {G1,G2,...,Gn}, we define support(g) (or f requency(g)) as the percentage (or number) of graphs in D where g is a subgraph. A frequent graph is a graph whose support is no less than a minimum support threshold, min sup**.**
- "How can we discover frequent substructures?"
- The discovery of frequent substructures usually consists of two steps.
- In the first step, we generate frequent substructure candidates.
- The frequency of each candidate is checked in the second step.
- Various methods for frequent substructure mining are
- an Apriori-based approach and
- a pattern-growth approach

Apriori-based approach:

- Apriori-based frequent substructure mining algorithms share similar characteristics with Apriori-based frequent itemset mining algorithms.
- The search for frequent graphs starts with graphs of small "size," and proceeds in a bottom-up manner by generating candidates having an extra vertex, edge, or path.
- Sk is the frequent substructure set of size k.
- AprioriGraph adopts a level-wise mining methodology.
- At each iteration, the size of newly discovered frequent substructures is increased by one.

- These new substructures are first generated by joining two similar but slightly different frequent subgraphs that were discovered in the previous call to AprioriGraph.
- The frequency of the newly formed graphs is then checked. Those found to be frequent are used to generate larger candidates in the next round.
- Apriori-based algorithms for frequent substructure mining include AGM, FSG, and a path-join method.
- AGM shares similar characteristics with Apriori-based itemset mining. FSG and the path-join method explore edges and connections in an Apriori-based fashion. Each of these methods explores various candidate generation strategies.
- The **AGM algorithm** uses a vertex-based candidate generation method that increases the substructure size by one vertex at each iteration of AprioriGraph.
- Two size-k frequent graphs are joined only if they have the same size-(k−1) subgraph. Here, graph size is the number of vertices in the graph.
- The newly formed candidate includes the size-(k−1) subgraph in common and the additional two vertices from the two size-k patterns. Because it is undetermined whether there is an edge connecting the additional two vertices, we actually can form two substructures.
- The **FSG algorithm** adopts an edge-based candidate generation strategy that increases the substructure size by one edge in each call of AprioriGraph. Two size-k patterns are merged if and only if they share the same subgraph having k − 1 edges, which is called the core. Here, graph size is taken to be the number of edges in the graph. The newly formed candidate includes the core and the additional two edges from the size-k patterns. Each candidate has one more edge than these two patterns.
- In the third Apriori-based approach, an **edge-disjoint path** method was proposed, where graphs are classified by the number of disjoint paths they have, and two paths are edge-disjoint if they do not share any common edge. A substructure pattern with k +1 disjoint paths is generated by joining substructures with k disjoint paths.
- Apriori-based algorithms have considerable overhead when joining two size-k frequent substructures to generate size-(k + 1) graph candidates.
- **Pattern Growth Approach:**

The Apriori-based approach has to use the breadth-first search (BFS) strategy because of its level-wise candidate generation. In order to determine whether a size-(k + 1) graph is frequent, it must check all of its corresponding size-k subgraphs to obtain an upper bound of its frequency. Thus, before mining any size-(k +1) subgraph, the Apriori-like approach usually has to complete the mining of size-k subgraphs. Therefore, BFS is necessary in the Apriori-like approach. In contrast, the pattern-growth approach is more flexible regarding its search method. It can use breadth-first search as well as depth-first search (DFS), the latter of which consumes less memory.

A graph g can be extended by adding a new edge e. The newly formed graph is denoted by gx e. Edge e may or may not introduce a new vertex to g. If e introduces a new vertex, we denote the new graph by g x f e, otherwise, g xb e, where f or b indicates that the extension is in a forward or backward direction.

- For each discovered graph g, it performs extensions recursively until all the frequent graphs with g embedded are discovered. The recursion stops once no frequent graph can be generated.

- PatternGrowthGraph is simple, but not efficient. The bottleneck is at the inefficiency of extending a graph. The same graph can be discovered many times. For example, there may exist n different (n − 1)-edge graphs that can be extended to the same n-edge graph. The repeated discovery of the same graph is computationally inefficient. We call a graph that is discovered a second time a duplicate graph.

- The gSpan algorithm is designed to reduce the generation of duplicate graphs. It need not search previously discovered frequent graphs for duplicate detection. It does not extend any duplicate graph, yet still guarantees the discovery of the complete set of frequent graphs

- To traverse graphs, it adopts a depth-first search. Initially, a starting vertex is randomly chosen and the vertices in a graph are marked so that we can tell which vertices have been visited. The visited vertex set is expanded repeatedly until a full depth-first search (DFS) tree is built. One graph may have various DFS trees depending on how the depth-first search is performed. The vertex labels are x, y, and z; the edge labels are a and b. Alphabetic order is taken as the default order in the labels. When building a DFS tree, the visiting sequence of vertices forms a linear order. We use subscripts to record this order, where i< j means vi is visited before v j when the depth-first search is performed. A graph G subscripted with a DFS tree T is written as GT . T is called a DFS subscripting of G.

- Given a DFS tree T, we call the starting vertex in T, v0, the root. The last visited vertex, vn, is called the right-most vertex. The straight path from v0 to vn is called the right-most path.

- PatternGrowth extends a frequent graph in every possible position, which may generate a large number of duplicate graphs. The gSpan algorithm introduces a more sophisticated extension method. The new method restricts the extension as follows: Given a graph G and a DFS tree T in G, a new edge e can be added between the right-most vertex and another vertex on the right-most path (backward extension); or it can introduce a new vertex and connect to a vertex on the right-most path (forward extension). Because both kinds of extensions take place on the right-most path, we call them right-most extension.

ASSIGNMENT QUESTIONS

1) a) Illustrate with an Apriori algorithm for the dataset given below. Consider minimum support =2.

TID	List of items
001	milk, dal, sugar, bread
002	Dal, sugar, wheat,jam
003	Milk, bread, curd, paneer
004	Wheat, paneer, dal, sugar
005	Milk, paneer, bread
006	Wheat, dal, paneer, bread

b) Explain a method that mines the complete set of frequent item sets without candidate generation by taking the above example.

2) What is correlation analysis? Explain the measures of correlation analysis.

3) Mining Various kinds of association rules.

4) What is Graph pattern mining?

OBJECTIVE QUESTIONS

1) FP growth algorithm requires fewer scans of data.
2) Frequency of occurrence of an itemset is called as Support Count
3) A good heuristic is to have the users specify such intuition or expectations as constraints to confine the search space is called Constraint based mining
4) Apriori-based approach uses BFS strategy.
5) FP Tree stands for frequent pattern tree.
6) Association rule Miningtechniques are used to detect relationships or associations between specific values of categorical variables in large data sets.
7) Four measures of correlation analysis are cosine, lift ,sye square and confidence.
8) The occurrence of itemset A is independent of the occurrence of itemset B if

$P(A \cup B) = P(A)P(B)$.

NPTEL LINKS

https://nptel.ac.in/courses/106/105/106105174/

UNIT -III

3.Classification and Prediction

3.1 Basic Concept of Classification

3.1 Basic Concept of Classification

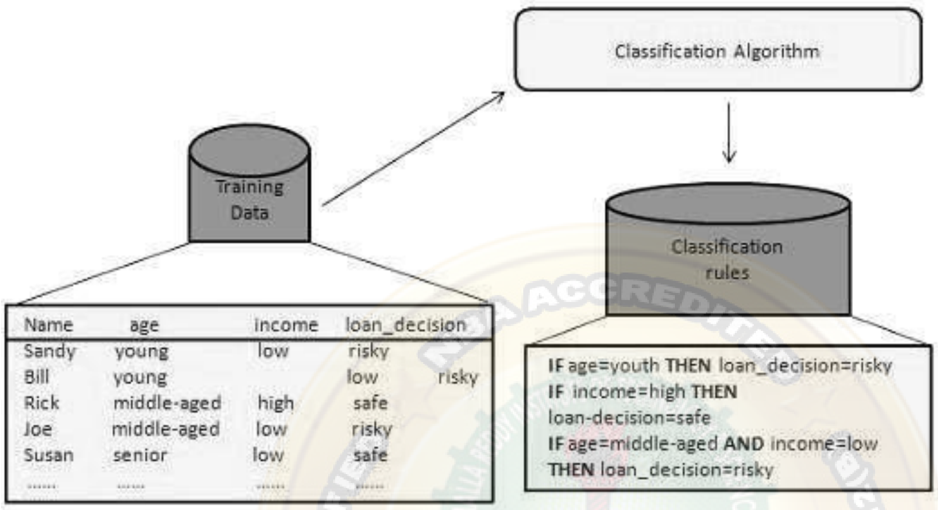
- There are two forms of data analysis that can be used for extracting models describing important classes or to predict future data trends. These two forms are as follows −
- Classification
- Prediction

Classification models predict categorical class labels; and prediction models predict 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 in dollars of potential customers on computer equipment given their income and occupation.

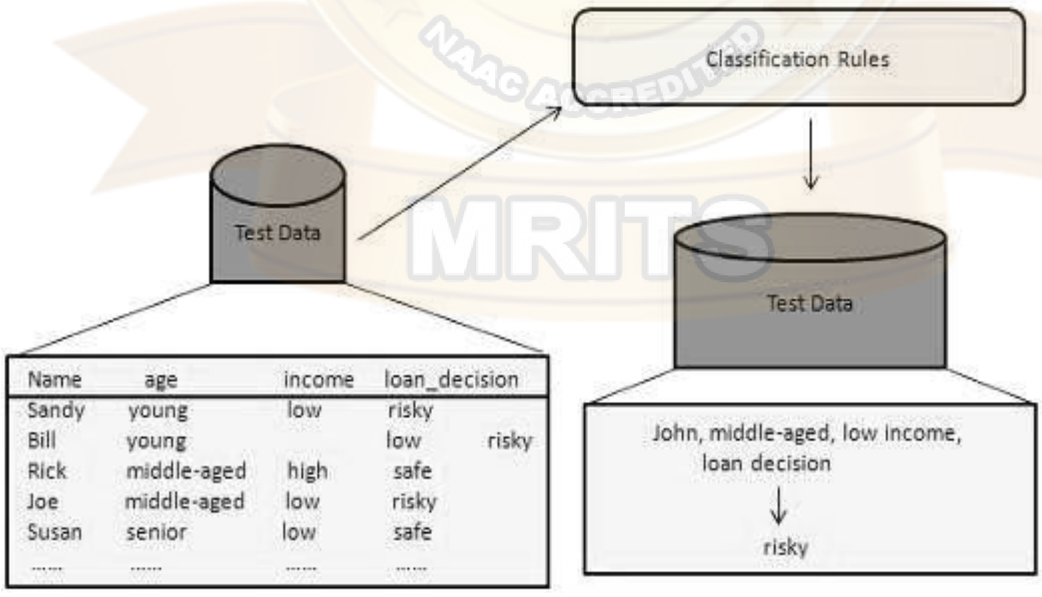
- What is classification?
- **Classification** is the problem of identifying to which of a set of categories (subpopulations), a new observation belongs to, on the basis of a training set of data containing observations and whose categories membership is known.
- Following are the examples of cases where the data analysis task is Classification −
- A bank loan officer wants to analyze the data in order to know which customer (loan applicant) are risky or which are safe.
- A marketing manager at a company needs to analyze a customer with a given profile, who will buy a new computer.
- In both of the above examples, a model or classifier is constructed to predict the categorical labels. These labels are risky or safe for loan application data and yes or no for marketing data.
- What is prediction?
- Following are the examples of cases where the data analysis task is Prediction −
- Suppose the marketing manager needs to predict how much a given customer will spend during a sale at his company. In this example we are bothered to predict a numeric value. Therefore the data analysis task is an example of numeric prediction. In this case, a model or a predictor will be constructed that predicts a continuous-valued-function or ordered value.
- Data classification is a two-step process:
- Building the Classifier or Model
- Using Classifier for Classification

In the first step, a classifier is built describing a predetermined set of data classes or concepts. This is the learning step (or training phase), where a classification algorithm builds the classifier by analyzing or "learning from" a training set made up of database tuples and their associated class labels.

- Each tuple that constitutes the training set is referred to as a category or class. These tuples can also be referred to as sample, object or data points.


Classification Algorithm
Training Data
Classification rules
Name age income loan_decision
Sandy young low risky
Bill young low risky
Rick middle-aged high safe
Joe middle-aged low risky
Susan senior low safe
IF age=youth THEN loan_decision=risky
IF income=high THEN loan-decision=safe
IF age=middle-aged AND income=low THEN loan_decision=risky

- Using Classifier for Classification
- In this step, the classifier is used for classification. Here the test data is used to estimate the accuracy of classification rules. The classification rules can be applied to the new data tuples if the accuracy is considered acceptable.


Classification Rules
Test Data
Test Data
Name age income loan_decision
Sandy young low risky
Bill young low risky
Rick middle-aged high safe
Joe middle-aged low risky
Susan senior low safe
John, middle-aged, low income, loan decision
risky

Issues Regarding Classification and Prediction

- The major issue is preparing the data for Classification and Prediction. Preparing the data involves the following activities −
- **Data Cleaning** − Data cleaning involves removing the noise and treatment of missing values. The noise is removed by applying smoothing techniques and the problem of missing values is solved by replacing a missing value with most commonly occurring value for that

attribute.

- **Relevance Analysis** − Database may also have the irrelevant attributes. Correlation analysis is used to know whether any two given attributes are related.
- **Data Transformation and reduction** − The data can be transformed by any of the following methods.
    - **Normalization** − The data is transformed using normalization. Normalization involves scaling all values for given attribute in order to make them fall within a small specified range. Normalization is used when in the learning step, the neural networks or the methods involving measurements are used.
    - **Generalization** − The data can also be transformed by generalizing it to the higher concept. For this purpose we can use the concept hierarchies.

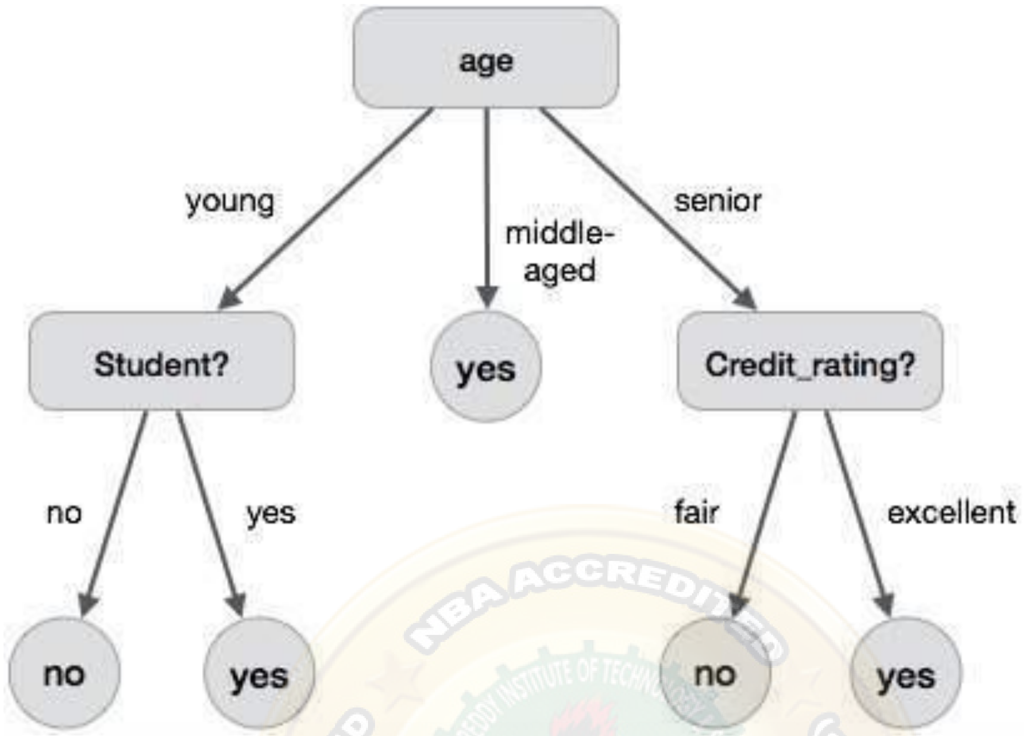
Comparison of Classification and Prediction Methods

- **Accuracy** − Accuracy of classifier refers to the ability of classifier. It predict the class label correctly and the accuracy of the predictor refers to how well a given predictor can guess the value of predicted attribute for a new data.
- **Speed** − This refers to the computational cost in generating and using the classifier or predictor.
- **Robustness** − It refers to the ability of classifier or predictor to make correct predictions from given noisy data.
- **Scalability** − Scalability refers to the ability to construct the classifier or predictor efficiently; given large amount of data.
- **Interpretability** − It refers to what extent the classifier or predictor understands.

3.2 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 (non leaf node) denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (or terminal node) holds a class label. The topmost node in a tree is the root node.

The following decision tree is for the concept buy computer that indicates whether a customer at a company is likely to buy a computer or not.



-
- “How are decision trees used for classification?”
- Given a tuple, X, for which the associated class label is unknown, the attribute values of the tuple are tested against the decision tree. A path is traced from the root to a leaf node, which holds the class prediction for that tuple. Decision trees can easily be converted to classification rules.
- “Why are decision tree classifiers so popular?”
- The construction of decision tree classifiers does not require any domain knowledge or parameter setting, and therefore is 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 Algorithm**
- A machine researcher named J. Ross Quinlan in 1980 developed a decision tree algorithm known as ID3 (Iterative Dichotomiser). Later, he presented C4.5, which was the successor of ID3. ID3 and C4.5 adopt a greedy approach. In this algorithm, there is no backtracking; the trees are constructed in a top-down recursive divide-and-conquer manner.
- Algorithm: Generate decision tree. Generate a decision tree from the training tuples of data partition D.
- Input: Data partition, D, which is a set of training tuples and their associated class labels;
- attribute list, the set of candidate attributes;
- Attribute selection method, a procedure to determine the splitting criterion that “best”

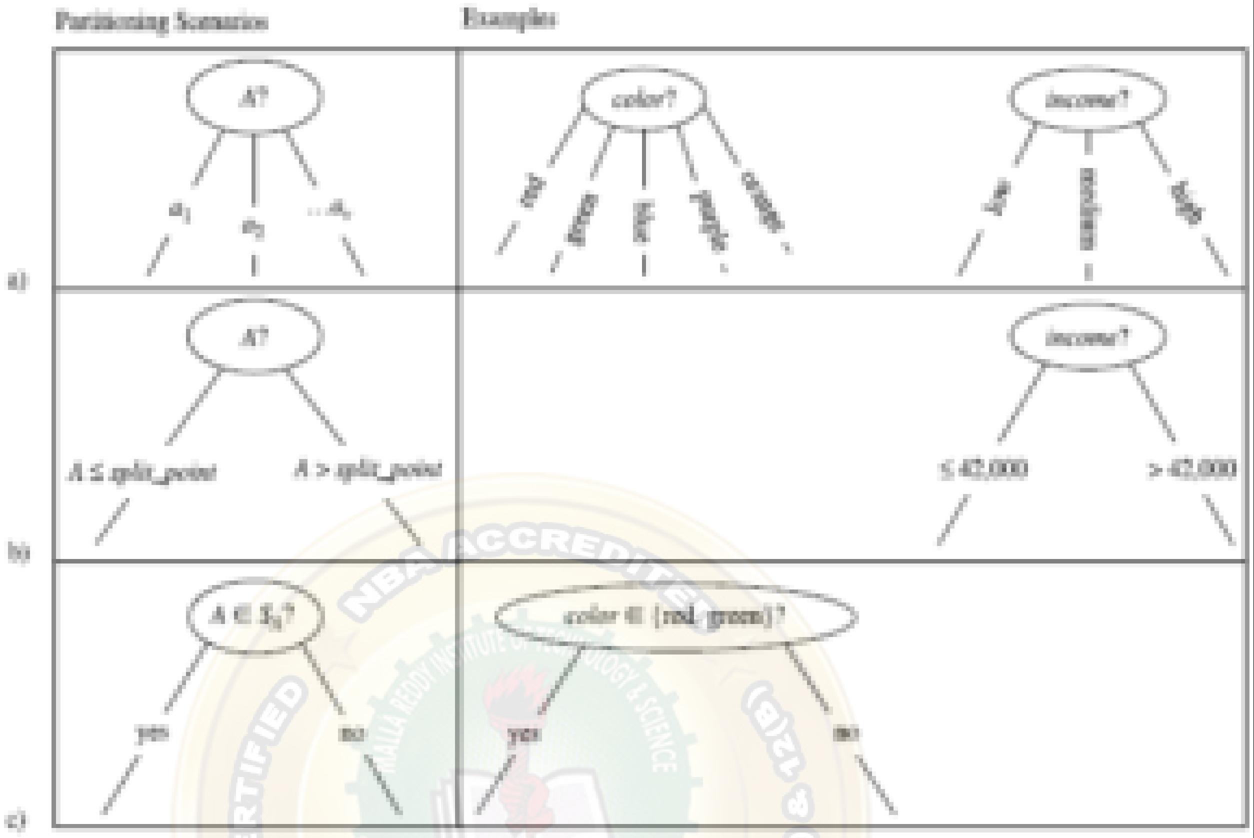
partitions the data tuples into individual classes. This criterion consists of a splitting attribute and, possibly, either a split point or splitting subset.

- Output: A decision tree.
- Method: (1) create a node N;
- (2) if tuples in D are all of the same class, C then
- (3) return N as a leaf node labeled with the class C;
- (4) if attribute list is empty then
- (5) return N as a leaf node labeled with the majority class in D; // majority voting
- (6) apply Attribute selection method(D, attribute list) to find the "best" splitting criterion;
- (7) label node N with splitting criterion;
- (8) if splitting attribute is discrete-valued and multiway splits allowed then // not restricted to binary trees
- (9) attribute list ← attribute list − splitting attribute; // remove splitting attribute
- (10) for each outcome j of splitting criterion // partition the tuples and grow subtrees for each partition
- (11) let Dj be the set of data tuples in D satisfying outcome j; // a partition
- (12) if Dj is empty then
- (13) attach a leaf labeled with the majority class in D to node N;
- (14) else attach the node returned by Generate decision tree(Dj , attribute list) to node N; endfor
- (15) return N;
- The algorithm is called with three parameters: D, attribute list, and Attribute selection method.
- We refer to D as a data partition. Initially, it is the complete set of training tuples and their associated class labels.
- 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. This procedure employs an attribute selection measure, such as information gain or the gini index. Whether the tree is strictly binary is generally driven by the attribute selection measure. Some attribute selection measures, such as the gini index, enforce the resulting tree to be binary. Others, like information gain, do not, therein allowing multiway splits.
- 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.
- 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.

- The splitting criterion also tells us which branches to grow from node N with respect to the outcomes of the chosen test. More specifically, the splitting criterion indicates the splitting attribute and may also indicate either a split-point or a splitting subset.
- The node N is labeled with the splitting criterion, which serves as a test at the node . A branch is grown from node N for each of the outcomes of the splitting criterion. The tuples in D are partitioned accordingly.
- There are three possible scenarios:
- . Let A be the splitting attribute. A has v distinct values, {a1, a2,..., 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. Partition Dj is the subset of class-labeled tuples in D having value aj of A. Because all of the tuples in a given partition have the same value for A, then A need not be considered in any future partitioning of the tuples. Therefore, it is removed from attribute list.
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. Two branches are grown from N and labeled according to the above outcomes.
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. If a given tuple has value aj of A and if aj∈ SA, then the test at node N is satisfied. Two branches are grown from N.
4. The left branch out of N is labeled 'yes' so that D1 corresponds to the subset of class-labeled tuples in D that satisfy the test. The right branch out of N is labeled 'no' so that D2 corresponds to the subset of class-labeled tuples from D that do not satisfy the test.



- **Attribute Selection Measures :**
- An attribute selection measure is a heuristic for selecting the splitting criterion that "best" separates a given data partition, D, of class-labeled training tuples into individual classes.
- Attribute selection measures are also known as splitting rules because they determine how the tuples at a given node are to be split. The attribute selection measure provides a ranking for each attribute describing the given training tuples. The attribute having the best score for the measure is chosen as the splitting attribute for the given tuples. If the splitting attribute is continuous-valued or if we are restricted to binary trees then, respectively, either a split point or a splitting subset must also be determined as part of the splitting criterion. The tree node created for partition D is labeled with the splitting criterion, branches are grown for each outcome of the criterion, and the tuples are partitioned accordingly.
- Three popular attribute selection measures—

  1.information gain,

  2. gain ratio, and

  3. gini index.

  Tree Pruning:

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

  Pruned trees tend to be smaller and less complex and, thus, easier to comprehend. They are usually faster and better at correctly classifying independent test data than unpruned trees.

There are two common approaches to tree pruning:

prepruning and

postpruning

Prepuning

In the prepruning approach, a tree is “pruned” by halting its construction early.

Upon halting, the node becomes a leaf. The leaf may hold the most frequent class among the subset tuples or the probability distribution of those tuples.

When constructing a tree, measures such as statistical significance, information gain, Gini index, and so on can be used to assess the goodness of a split. If partitioning the tuples at a node would result in a split that falls below a prespecified threshold, then further partitioning of the given subset is halted. There are difficulties, however, in choosing an appropriate threshold. High thresholds could result in oversimplified trees, whereas low thresholds could result in very little simplification

Postpruning

The second and more common approach is postpruning, which removes subtrees from a “fully grown” tree. A subtree at a given node is pruned by removing its branches and replacing it with a leaf. The leaf is labeled with the most frequent class among the subtree being replaced.

- The complexity pruning algorithm used in CART is an example of the postpruning approach. This approach considers the cost complexity of a tree to be a function of the number of leaves in the tree and the error rate of the tree. It starts from the bottom of the tree. For each internal node, N, it computes the cost complexity of the subtree at N, and the cost complexity of the subtree at N if it were to be pruned. The two values are compared. If pruning the subtree at node N would result in a smaller cost complexity, then the subtree is pruned. Otherwise, it is kept.

3.4 Rule-Based Classification

We use rule-based classifiers, where the learned model is represented as a set of IF-THEN rules.

- Using IF-THEN Rules for Classification
- Rule Extraction from a Decision Tree
- Rule Induction Using a Sequential Covering Algorithm
- **Using IF-THEN Rules for Classification**

Rules are a good way of representing information or bits of knowledge. A rule-based classifier uses a set of IF-THEN rules for classification. An IF-THEN rule is an expression of the form

IF condition THEN conclusion.

An example is rule R1,

R1: IF age = youth AND student = yes THEN buys computer = yes

The “IF”-part (or left-hand side) of a rule is known as the rule antecedent or precondition.

The “THEN”-part (or right-hand side) is the rule consequent. In the rule antecedent, the condition consists of one or more attribute tests (such as age = youth, and student = yes) that are logically ANDed.

The rule’s consequent contains a class prediction .

R1 can also be written as

R1: (age = youth) ∧ (student = yes) ⇒ (buys computer = yes).

- If the condition (that is, all of the attribute tests) in a rule antecedent holds true for a given tuple, we say that the rule antecedent issatisfied (or simply, that the rule is satisfied) and that the rule covers the tuple.
- A rule R can be assessed by its coverage and accuracy.
- Given a tuple, X, from a class labeled data set, D, let n_{covers} be the number of tuples covered by R; $n_{correct}$ be the number of tuples correctly classified by R; and |D| be the number of tuples in D. We can define the coverage and accuracy of R as

$$coverage(R) = n_{covers}/ |D|$$

$$accuracy(R) = n_{correct}/ n_{covers}$$

- How we can use rule-based classification to predict the class label of a given tuple, X.
- If a rule is satisfied by X, the rule is said to be triggered. For example, suppose we have
- X= (age = youth, income = medium, student = yes, credit rating = fair). We would like to classify X according to buys computer. X satisfies R1, which triggers the rule.
- If R1 is the only rule satisfied, then the rule fires by returning the class prediction for X.
- If more than one rule is triggered, we have a potential problem. What if they each specify a different class? Or what if no rule is satisfied by X?
- If more than one rule is triggered, we need a conflict resolution strategy to figure out which rule gets to fire and assign its class prediction to X. There are many possible strategies.

Among them are two strategies namely size ordering and rule ordering.

The size ordering scheme assigns the highest priority to the triggering rule that has the “toughest” requirements, where toughness is measured by the rule antecedent size. That is, the triggering rule with the most attribute tests is fired. The rule ordering scheme prioritizes the rules beforehand.

The ordering may be class based or rule-based.

With class-based ordering, the classes are sorted in order of decreasing “importance,” such as by decreasing order of prevalence.

With rule-based ordering, the rules are organized into one long priority list, according to some measure of rule quality such as accuracy, coverage, or size or based on advice from domain experts

- **Rule Extraction from a Decision Tree**
- Decision tree classifiers are a popular method of classification—it is easy to understand how decision trees work and they are known for their accuracy. Decision trees can become large and difficult to interpret.
- We will see how to build a rule based classifier by extracting IF-THEN rules from a decision tree. In comparison with a decision tree, the IF-THEN rules may be easier for humans to understand, particularly if the decision tree is very large.
- R1: IF age = youth AND student = no THEN buys computer = no
- R2: IF age = youth AND student = yes THEN buys computer = yes
- R3: IF age = middle aged THEN buys computer = yes
- R4: IF age = senior AND credit rating = excellent THEN buys computer = yes
- R5: IF age = senior AND credit rating = fair THEN buys computer = no

3.5 Lazy learners

- Eager learners, when given a set of training tuples, will construct a generalization (i.e., classification) model before receiving new (e.g., test) tuples to classify.
- In contrast a lazy approach, in which the learner instead waits until the last minute before doing any model construction in order to classify a given test tuple. That is, when given a training tuple, a lazy learner simply stores it and waits until it is given a test tuple.
- Only when it sees the test tuple does it perform generalization in order to classify the tuple based on its similarity to the stored training tuples.
- Unlike eager learning methods, lazy learners do less work when a training tuple is presented and more work when making a classification or prediction. Because lazy learners store the training tuples or "instances," they are also referred to as instancebased learners, even though all learning is essentially based on instances.

Two examples of lazy learners:

- k-nearestneighbor classifiers and
- case-based reasoning classifiers

Nearest-neighbor classifiers:

- Nearest-neighbor classifiers are based on learning by analogy, that is, by comparing a given test tuple with training tuples that are similar to it. The training tuples are described by n attributes. Each tuple represents a point in an n-dimensional space.
- In this way, all of the training tuples are stored in an n-dimensional pattern space. When

given an unknown tuple, a k-nearest-neighbor classifier searches the pattern space for the k training tuples that are closest to the unknown tuple. These k training tuples are the k "nearest neighbors" of the unknown tuple.

- "Closeness" is defined in terms of a distance metric, such as Euclidean distance. The Euclidean distance between two points or tuples, say, $X1 = (x11, x12,..., x1n)$ and $X2 = (x21, x22,..., x2n)$, is

$$dist(X1, X2) = \sqrt{\sum_{n}^{i=1} (x1i - x2i)\ 2}$$

In other words, for each numeric attribute, we take the difference between the corresponding values of that attribute in tuple X1 and in tuple X2, square this difference, and accumulate it. The square root is taken of the total accumulated distance count. Typically, we normalize the values of each attribute.

- This helps prevent attributes with initially large ranges (such as income) from outweighing attributes with initially smaller ranges (such as binary attributes).
- Min-max normalization, for example, can be used to transform a value v of a numeric attribute A to v' in the range [0, 1] by computing

$v' = v - minA / maxA - mina$

where minA and maxA are the minimum and maximum values of attribute A. Chapter 2 describes other methods for data normalization as a form of data transformation.

For k-nearest-neighbor classification, the unknown tuple is assigned the most common class among its k nearest neighbors. When k = 1, the unknown tuple is assigned the class of the training tuple that is closest to it in pattern space. Nearest neighbor classifiers can also be used for prediction, that is, to return a real-valued prediction for a given unknown tuple. In this case, the classifier returns the average value of the real-valued labels associated with the k nearest neighbors of the unknown tuple.

- "But how can distance be computed for attributes that not numeric, but categorical, such as color?"
- For categorical attributes, a simple method is to compare the corresponding value of the attribute in tuple X1 with that in tuple X2.
- If the two are identical (e.g., tuples X1 and X2 both have the color blue), then the difference between the two is taken as 0.
- If the two are different (e.g., tuple X1 is blue but tuple X2 is red), then the difference is considered to be 1.
- "What about missing values?"
- In general, if the value of a given attribute A is missing in tuple X1 and/or in tuple X2, we assume the maximum possible difference. Suppose that each of the attributes have been mapped to the range [0, 1].
- For categorical attributes, we take the difference value to be 1 if either one or both of the corresponding values of A are missing.

- If A is numeric and missing from both tuples X1 and X2, then the difference is also taken to be 1.
- If only one value is missing and the other (which we'll call v') is present and normalized, then we can take the difference to be either |1 − v ' | or |0 − v ' | (i.e., 1−v ' or v '), whichever is greater.
- **Case-Based Reasoning:**
- Case-based reasoning (CBR) classifiers use a database of problem solutions to solve new problems.
- Unlike nearest-neighbor classifiers, which store training tuples as points in Euclidean space, CBR stores the tuples or "cases" for problem solving as complex symbolic descriptions.
- CBR has also been applied to areas such as engineering and law, where cases are either technical designs or legal rulings, respectively. Medical education is another area for CBR, where patient case histories and treatments are used to help diagnose and treat new patients.
- When given a new case to classify, a case-based reasoner will first check if an identical training case exists.
- If one is found, then the accompanying solution to that case is returned. If no identical case is found, then the case-based reasoner will search for training cases having components that are similar to those of the new case.
- Conceptually, these training cases may be considered as neighbors of the new case.
- If cases are represented as graphs, this involves searching for subgraphs that are similar to subgraphs within the new case. The case-based reasoner tries to combine the solutions of the neighboring training cases in order to propose a solution for the new case.

ASSIGNMENT QUESTIONS

1)What is Prediction? Explain about Decision tree induction classification technique.

2) Explain about Bayesian Classification

3) Explain Lazy Learners.

OBJECTIVE QUESTIONS

1) A Decision Tree is a predictive model.
2) The left hand side of the association rule is called antecedent.
3) Case-based learning is an approach to the design of learning algorithms that is inspired by the fact that when people encounter new situations, they often explain them by reference to familiar experiences, adapting the explanations to fit the new situation.

4) Euclidean distance measure is the distance between two points as calculated using the Pythagoras theorem.

NPTEL LINK

https://nptel.ac.in/courses/106/105/106105174/

https://nptel.ac.in/courses/106/105/106105174/

UNIT-IV

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.

- The following are typical 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.

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 clusterings 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 highdimensional 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.

4.1 Types of Data in Cluster Analysis

Interval-scaled variables

binary variables

categorical,

Ordinal and

ratio-scaled variables

Interval scaled variables:

Interval-scaled variables are continuous measurements of a roughly linear scale.

Typical examples include weight and height, latitude and longitude coordinates (e.g., when clustering houses), and weather temperature.

The measurement unit used can affect the clustering analysis. For example, changing measurement units from meters to inches for height, or from kilograms to pounds for weight, may lead to a very different clustering structure.

In general, expressing a variable in smaller units will lead to a larger range for that variable, and thus a larger effect on the resulting clustering structure.

To help avoid dependence on the choice of measurement units, the data should be standardized.

- Standardizing measurements attempts to give all variables an equal weight.
- However, in some applications, users may intentionally want to give more weight to a certain set of variables than to others.
- For example, when clustering basketball player candidates, we may prefer to give more weight to the variable height.
- “How can the data for a variable be standardized?” To standardize measurements, one choice is to convert the original measurements to unitless variables.
- Given measurements for a variable f , this can be performed as follows.

1. Calculate the mean absolute deviation,

sf : sf = 1 /n (|x1 f −mf |+|x2 f −mf |+···+|xn f −mf |),

where x1 f ,..., xn f are n measurements of f , and mf is the mean value of f , that is, mf = 1 n (x1 f +x2 f +···+xn f).

- 2. Calculate the standardized measurement, or z-score:

zi f = xi f −mf /sf

After standardization, or without standardization in certain applications, the dissimilarity (or similarity) between the objects described by interval-scaled variables is typically computed based on the distance between each pair of objects.

The most popular distance measure is Euclidean distance, which is defined as d(i, j) = q (xi1 −x j1) 2+ (xi2 −x j2) 2 +···+ (xin −x jn) 2 ,

where i = (xi1, xi2,..., xin) and j = (x j1, x j2,..., x jn) are two n-dimensional data objects.

Another well-known metric is Manhattan (or city block) distance, defined as d(i, j) = |xi1 −x j1|+|xi2 −x j2|+···+|xin −x jn|.

- Both the Euclidean distance and Manhattan distance satisfy the following mathematic requirements of a distance function.

1. $d(i, j) \geq 0$: Distance is a nonnegative number.
2. $d(i, j) = 0$: The distance of an object to itself is 0.
3. $d(i, j) = d(j, i)$: Distance is a symmetric function.
4. $d(i, j) \leq d(i, h)+d(h, j)$: Going directly from object i to object j in space is no more than making a detour over any other object h (triangular inequality)

- **Binary Variables :**

To compute the dissimilarity between objects described by either symmetric or asymmetric binary variables.

A binary variable has only two states: 0 or 1, where 0 means that the variable is absent, and 1 means that it is present.

Given the variable smoker describing a patient, for instance, 1 indicates that the patient smokes, while 0 indicates that the patient does not.

How can we compute the dissimilarity between two binary variables?

One approach involves computing a dissimilarity matrix from the given binary data.

- If all binary variables are thought of as having the same weight, we have the 2-by-2 contingency table ,
- where q is the number of variables that equal 1 for both objects i and j,
- r is the number of variables that equal 1 for object i but that are 0 for object j,
- s is the number of variables that equal 0 for object i but equal 1 for object j, and
- t is the number of variables that equal 0 for both objects i and j.
- The total number of variables is p, where $p = q+r+s+t$.
- "What is the difference between symmetric and asymmetric binary variables?"
- A binary variable is **symmetric** if both of its states are equally valuable and carry the same weight; that is, there is no preference on which outcome should be coded as 0 or 1.
- One such example could be the attribute gender having the states male and female. Dissimilarity that is based on symmetric binary variables is called symmetric binary dissimilarity.
- Its dissimilarity (or distance) measure, can be used to assess the dissimilarity between objects i and j.
- $d(i, j) = r +s /q+r+s+t$.
- A binary variable is **asymmetric** if the outcomes of the states are not equally important, such as the positive and negative outcomes of a disease test.
- By convention, we shall code the most important outcome, which is usually the rarest one, by 1 (e.g., HIV positive) and the other by 0 (e.g., HIV negative).
- Given two asymmetric binary variables, the agreement of two 1s (a positive match) is then considered more significant than that of two 0s (a negative match).

- Therefore, such binary variables are often considered “monary” (as if having one state).
- The dissimilarity based on such variables is called asymmetric binary dissimilarity, where the number of negative matches, t, is considered unimportant and thus is ignored in the computation.
- d(i, j) = r +s/ q+r +s .
- **Categorical, Ordinal, and Ratio-Scaled Variables:**
- **Categorical Variables**
- A categorical variable is a generalization of the binary variable in that it can take on more than two states. For example, map color is a categorical variable that may have, say, five states: red, yellow, green, pink, and blue.
- Let the number of states of a categorical variable be M.
- The states can be denoted by letters, symbols, or a set of integers, such as 1, 2,..., M.
- Notice that such integers are used just for data handling and do not represent any specific ordering.
- “How is dissimilarity computed between objects described by categorical variables?”
- The dissimilarity between two objects i and j can be computed based on the ratio of mismatches:
- d(i, j) = p−m/ p
- where m is the number of matches (i.e., the number of variables for which i and j are in the same state), and p is the total number of variables.
- **Ordinal Variables**

A discrete ordinal variable resembles a categorical variable, except that the M states of the ordinal value are ordered in a meaningful sequence. Ordinal variables are very useful for registering subjective assessments of qualities that cannot be measured objectively.

For example, professional ranks are often enumerated in a sequential order, such as assistant, associate, and full for professors.

Ratio scaled variables

A ratio-scaled variable makes a positive measurement on a nonlinear scale, such as an exponential scale, approximately following the formula AeBt or Ae−Bt where A and B are positive constants, and t typically represents time.

Common examples include the growth of a bacteria population or the decay of a radioactive elements.

4.2 Categorization of Major Clustering Methods

The major clustering methods can be classified into the following categories.

- Partitioning methods
- Hierarchical methods

- Density-based methods
- Grid-based methods
- Model-based methods

Partitioning Methods:

- Given a database of n objects or data tuples, 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:

  (1) each group must contain at least one object, and

  (2) each object must belong to exactly one group.

  Given k, the number of partitions to construct, 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.
- There are various kinds of other criteria for judging the quality of partitions.
- Most applications adopt one of a few popular heuristic methods, such as (1) the k-means algorithm, where each cluster is represented by the mean value of the objects in the cluster, and

  (2) the k-medoids algorithm, where each cluster is represented by one of the objects located near the center of the cluster. These heuristic clustering methods work well for finding spherical-shaped clusters in small to medium-sized databases.

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 (the topmost level of the hierarchy), 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.

Density-based methods:

Most partitioning methods cluster objects based on the distance between objects.

Such methods can find only spherical-shaped clusters and encounter difficulty at discovering clusters of arbitrary shapes.

Other clustering methods have been developed based on the notion of density.

Their general idea is to continue growing the given cluster as long as the density (number of objects or data points) in the "neighborhood" exceeds some threshold; that is, for each data point within a given cluster, the neighborhood of a given radius has to contain at least a minimum number of points. Such a method can be used to filter out noise (outliers) and discover clusters of arbitrary shape.

Grid-based methods:

Grid-based methods quantize the object space into a finite number of cells that form a grid structure.

All of the clustering operations are performed on the grid structure (i.e., on the quantized space).

The main advantage of this approach is its fast processing time, which is typically independent of the number of data objects and dependent only on the number of cells in each dimension in the quantized space.

STING is a typical example of a grid-based method.

Model-based methods:

Model-based methods hypothesize a model for each of the clusters and find the best fit of the data to the given model.

A model-based algorithm may locate clusters by constructing a density function that reflects the spatial distribution of the data points. It also leads to a way of automatically determining the number of clusters based on standard statistics, taking "noise" or outliers into account and thus yielding robust clustering methods.

- Given D, a data set of n objects, and k, the number of clusters to form, a partitioning algorithm organizes the objects into k partitions ($k \leq n$), where each partition represents a cluster.

4.3Partitioning Methods

- The clusters are formed to optimize an objective partitioning criterion, such as a dissimilarity function based on distance, so that the objects within a cluster are "similar," whereas the objects of different clusters are "dissimilar" in terms of the data set attributes.
- The most well-known and commonly used partitioning methods are
- 1.k-means,

  2. k-medoids, and their variations.

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

The k-means algorithm takes the input parameter, k, and partitions a set of n objects into k clusters so that the resulting intracluster similarity is high but the intercluster similarity is low.

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.

"How does the k-means algorithm work?" 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

E = k ∑ i=1 ∑ p∈Ci |p−mi | 2 ,

4.4 Hierarchical clustering method

- A hierarchical clustering method works by grouping data objects into a tree of 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.
- In general, there are two types of hierarchical clustering methods: **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 In general, there are two types of hierarchical clustering methods: 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.
- One promising direction for improving the clustering quality of hierarchical methods is to integrate hierarchical clustering with other clustering techniques, resulting in multiple-phase clustering.
- Three such methods are introduced
- The first, called **BIRCH**, begins by partitioning objects hierarchically using tree structures, where the leaf or low-level nonleaf nodes can be viewed as "microclusters" depending on the scale of resolution. It then applies other clustering algorithms to perform macroclustering on the microclusters.
- The second method, called **ROCK**, merges clusters based on their interconnectivity.
- The third method, called **Chameleon**, explores dynamic modeling in hierarchical clustering.

BIRCH: Balanced Iterative Reducing and Clustering Using Hierarchies

BIRCH is designed for clustering a large amount of numerical data by integration of hierarchical clustering and other clustering methods such as iterative partitioning.

It overcomes the two difficulties of agglomerative clustering methods:

(1) scalability and

(2) the inability to undo what was done in the previous step.

BIRCH introduces two concepts,

- Clustering feature and
- Clustering feature tree (CF tree), which are used to summarize cluster representations.

Given n d-dimensional data objects or points in a cluster, we can define the centroid x0, radius R, and diameter D of the cluster as follows:

Centroid $x0 = \sum_{i=1}^{n} xi/n$

Radius R= (xi-x0)/n

Diameter= √∑i=1 to n ∑j=1 to n(xi-xj)2/n(n-1)

- where R is the average distance from member objects to the centroid, and
- D is the average pairwise distance within a cluster.
- Both R and D reflect the tightness of the cluster around the centroid.
- A clustering feature (CF) is a three-dimensional vector summarizing information about clusters of objects.
- Given n d-dimensional objects or points in a cluster, {xi}, then the CF of the cluster is defined as
- CF = {n, LS, SS},
- where n is the number of points in the cluster, LS is the linear sum of the n points (i.e., $\sum_{i=1}^{n} xi$, and
- SS is the square sum of the data points (i.e.,)$\sum_{i=1}^{n} xi2$
- For example, suppose that we have two disjoint clusters, C1 and C2, having the clustering features, CF1 and CF2, respectively. The clustering feature for the cluster that is formed by merging C1 and C2 is simply CF1 +CF2.
- Clustering feature. Suppose that there are three points, (2, 5), (3, 2), and (4, 3), in a cluster, C1. The clustering feature of C1 is CF1 = {3,(2+3+4,5+2+3),(2^2+3^2 +4^2 ,5^2 +2^2 +3^2)} = {3,(9,10),(29,38)}.
- A CF tree is a height-balanced tree that stores the clustering features for a hierarchical clustering.
- A non-leaf node in a tree has descendants or “children.” The non-leaf nodes store sums of the CFs of their children, and thus summarize clustering information about their children.
- A CF tree has two parameters:
- Branching factor, B, and
- Threshold, T.
- The branching factor specifies the maximum number of children per non-leaf node.
- The threshold parameter specifies the maximum diameter of subclusters stored at the leaf nodes of the tree. These two parameters influence the size of the resulting tree.
- BIRCH applies a multiphase clustering technique: a single scan of the data set yields a basic good clustering, and one or more additional scans can (optionally) be used to further improve the quality.
- **Phase 1**: BIRCH scans the database to build an initial in-memory CF tree, which can be viewed as a multilevel compression of the data that tries to preserve the inherent clustering structure of the data.
- **Phase 2**: BIRCH applies a (selected) clustering algorithm to cluster the leaf nodes of the CF tree, which removes sparse clusters as outliers and groups dense clusters into larger ones.

- For Phase 1, the CF tree is built dynamically as objects are inserted. Thus, the method is incremental. An object is inserted into the closest leaf entry (subcluster). If the diameter of the subcluster stored in the leaf node after insertion is larger than the threshold value, then the leaf node and possibly other nodes are split. After the insertion of the new object, information about it is passed toward the root of the tree.

- **ROCK: A Hierarchical Clustering Algorithm for Categorical Attributes**
- ROCK (RObust Clustering using linKs) is a hierarchical clustering algorithm that explores the concept of links (the number of common neighbors between two objects) for data with categorical attributes.
- ROCK takes a more global approach to clustering by considering the neighborhoods of individual pairs of points.
- If two similar points also have similar neighborhoods, then the two points likely belong to the same cluster and so can be merged.
- More formally, two points, pi and pj , are neighbors if sim(pi , pj) $\geq \theta$, where sim is a similarity function and θ is a user-specified threshold.
- The number of links between pi and pj is defined as the number of common neighbors between pi and pj . If the number of links between two points is large, then it is more likely that they belong to the same cluster.
- ROCK's concepts of neighbors and links are illustrated in the following example, where the similarity between two "points" or transactions, Ti and Tj , is defined with the **Jaccard coefficient** as

  sim(Ti , Tj) = |Ti ∩Tj |/ |Ti∪Tj |

  Based on these ideas, ROCK first constructs a sparse graph from a given data similarity matrix using a similarity threshold and the concept of shared neighbors.

  It then performs agglomerative hierarchical clustering on the sparse graph.

  A goodness measure is used to evaluate the clustering.

  Chameleon: A Hierarchical Clustering Algorithm Using Dynamic Modeling

  Chameleon is a hierarchical clustering algorithm that uses dynamic modeling to determine the similarity between pairs of clusters.

  In Chameleon, cluster similarity is assessed based on how well-connected objects are within a cluster and on the proximity of clusters.

  That is, two clusters are merged if their **interconnectivity** is high and they are **close** together.

  It can automatically adapt to the internal characteristics of the clusters being merged.

  How does Chameleon work?

  Chameleon uses a k-nearest-neighbor graph approach to construct a sparse graph, where each vertex of the graph represents a data object, and there exists an edge between two vertices (objects) if one object is among the k-most-similar objects of the other.

The edges are weighted to reflect the similarity between objects.

- Chameleon uses a graph partitioning algorithm to partition the k-nearest-neighbor graph into a large number of relatively small subclusters.
- It then uses an agglomerative hierarchical clustering algorithm that repeatedly merges subclusters based on their similarity.
- To determine the pairs of most similar subclusters, it takes into account both the **interconnectivity** as well as the **closeness** of the clusters.

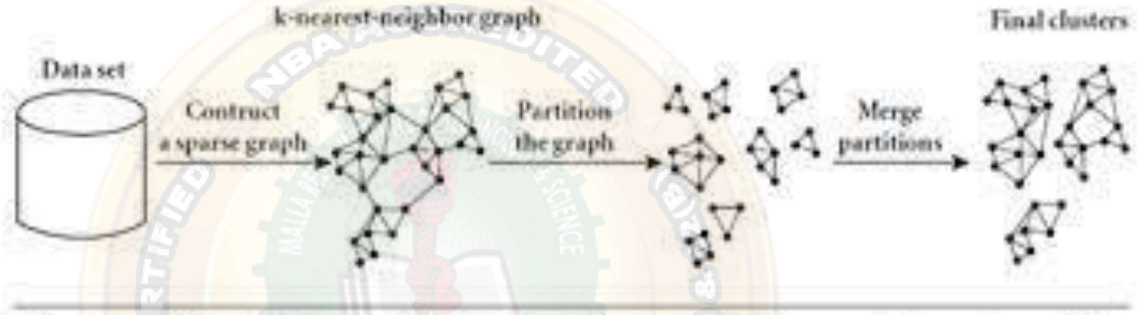


Figure 7.9 Chameleon: Hierarchical clustering based on *k*-nearest neighbors and dynamic modeling. Based on [KHK99].

- The graph-partitioning algorithm partitions the k-nearest-neighbor graph such that it minimizes the edge cut. That is, a cluster C is partitioned into subclusters Ci and Cj so as to minimize the weight of the edges that would be cut should C be bisected into Ci and Cj . Edge cut is denoted EC(Ci , Cj) and assesses the absolute interconnectivity between clusters Ci and Cj .
- Chameleon determines the similarity between each pair of clusters Ci and Cj according to their relative interconnectivity, RI(Ci , Cj), and their relative closeness, RC(Ci , Cj).
- The relative interconnectivity, RI(Ci , Cj), between two clusters,CiandCj , is defined as the absolute interconnectivity between Ci and Cj , normalized with respect to the internal interconnectivity of the two clusters, Ci and Cj . That is,
- RI(Ci , Cj) = |EC{Ci , Cj} |/ 1 /2 (|ECCi |+|ECCj |)
- where EC{Ci , Cj} is the edge cut, defined as above, for a cluster containing bothCi and Cj . Similarly, ECCi (or ECCj) is the minimum sum of the cut edges that partition Ci (or Cj) into two roughly equal parts.
- The relative closeness, RC(Ci , Cj), between a pair of clusters,CiandCj , is the absolute closeness between Ci and Cj , normalized with respect to the internal closeness of the two clusters, Ci and Cj.
- Chameleon has been shown to have greater power at discovering arbitrarily shaped clusters of high quality than several well-known algorithms such as BIRCH and densitybased DBSCAN.

NPTEL LINK

https://nptel.ac.in/courses/106/105/106105174/

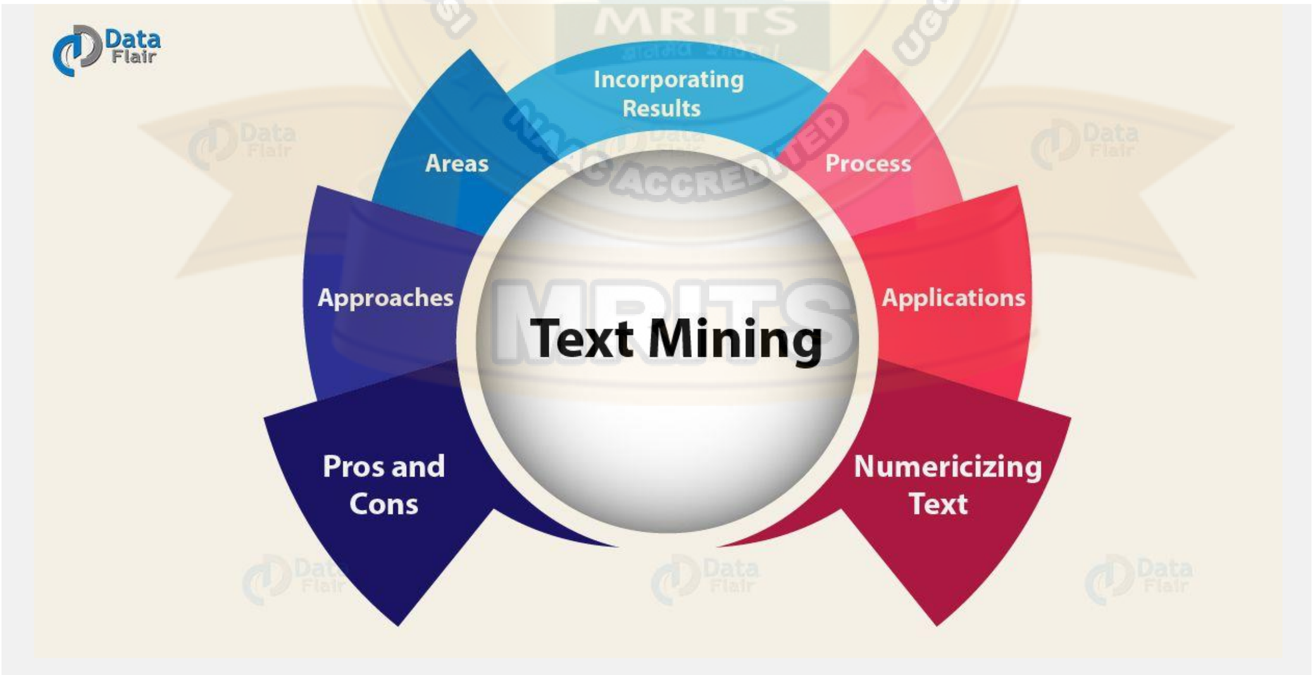
https://nptel.ac.in/courses/106/105/106105174/

UNIT-V

What is Text Mining:

Text Mining is also known as Text Data Mining. The purpose is too unstructured information, extract meaningful numeric indices from the text. Thus, make the information contained in the text accessible to the various algorithms. Information can extracte to derive summaries contained in the documents. Hence, you can analyze words, clusters of words used in documents. In the most general terms, text mining will "turn text into numbers". Such as predictive data mining projects, the application of unsupervised learning methods.

Through this Text Mining Tutorial, we will learn what is Text Mining, a process of Text Mining, Text Mining Applications, approaches, issues, areas, and Advantages and Disadvantages of Text Mining.



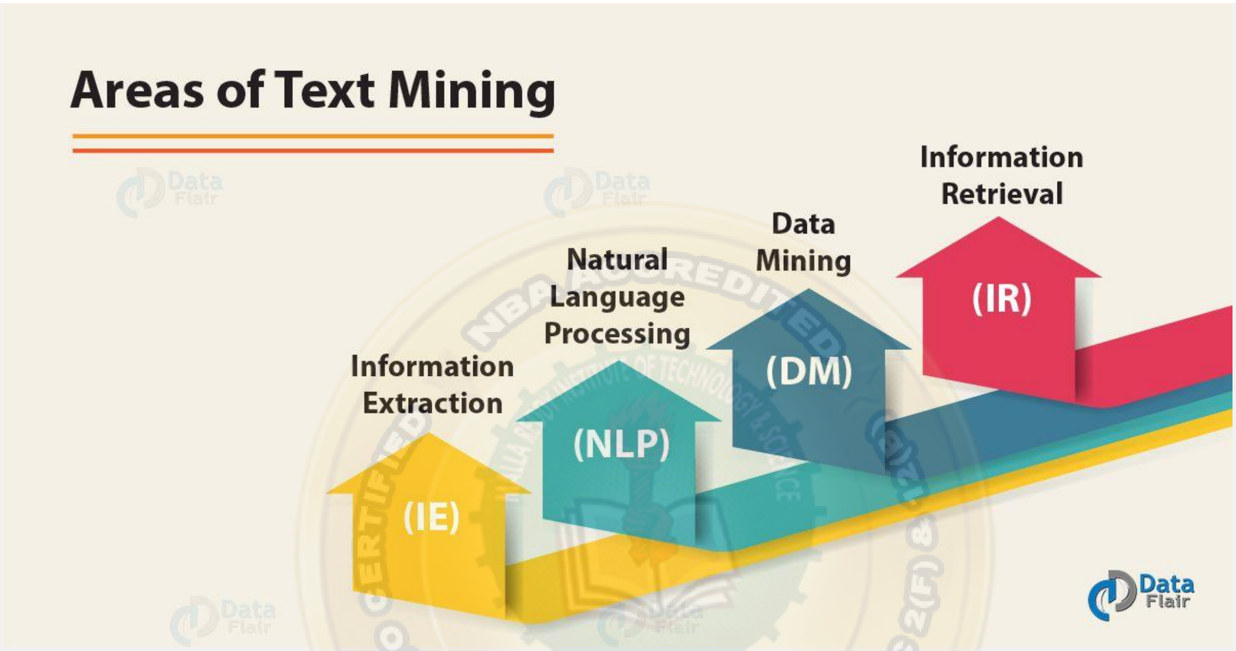
Text Mining in Data Mining – Concepts, Process & Applications

2. What is Text Mining?

Text Mining is also known as Text Data Mining. The purpose is too unstructured information, extract meaningful numeric indices from the text. Thus, make the information contained in the text accessible to the various algorithms. Information can extracte to derive summaries contained in the documents. Hence, you can analyze words, clusters of words used in documents. In the most general terms, text mining will "turn text into numbers". Such as predictive data mining projects, the application of unsupervised learning methods.

3. Areas of Text Mining in Data Mining

Following are the areas of text mining in Data Mining:



Areas of Text Mining in Data Mining

a. Information Retrieval (IR)

Information retrieval is regarded as an extension to document retrieval. That the documents that are returned are processed to condense. Thus document retrieval follow by a text summarization stage. That focuses on the query posed by the user. IR systems help in to narrow down the set of documents that are relevant to a particular problem. As text mining involves applying very complex algorithms to large document collections. Also, IR can speed up the analysis significantly by reducing the number of documents.

b. Data Mining (DM)

Data mining can loosely describe as looking for patterns in data. It can more characterize as the extraction of hidden from data. Data mining tools can predict behaviours and future trends. Also, it allows businesses to make positive, knowledge-based decisions. Data mining tools can answer business questions. Particularly that have traditionally been too time-consuming to resolve. They search databases for hidden and unknown patterns.

c. Natural Language Processing (NLP)

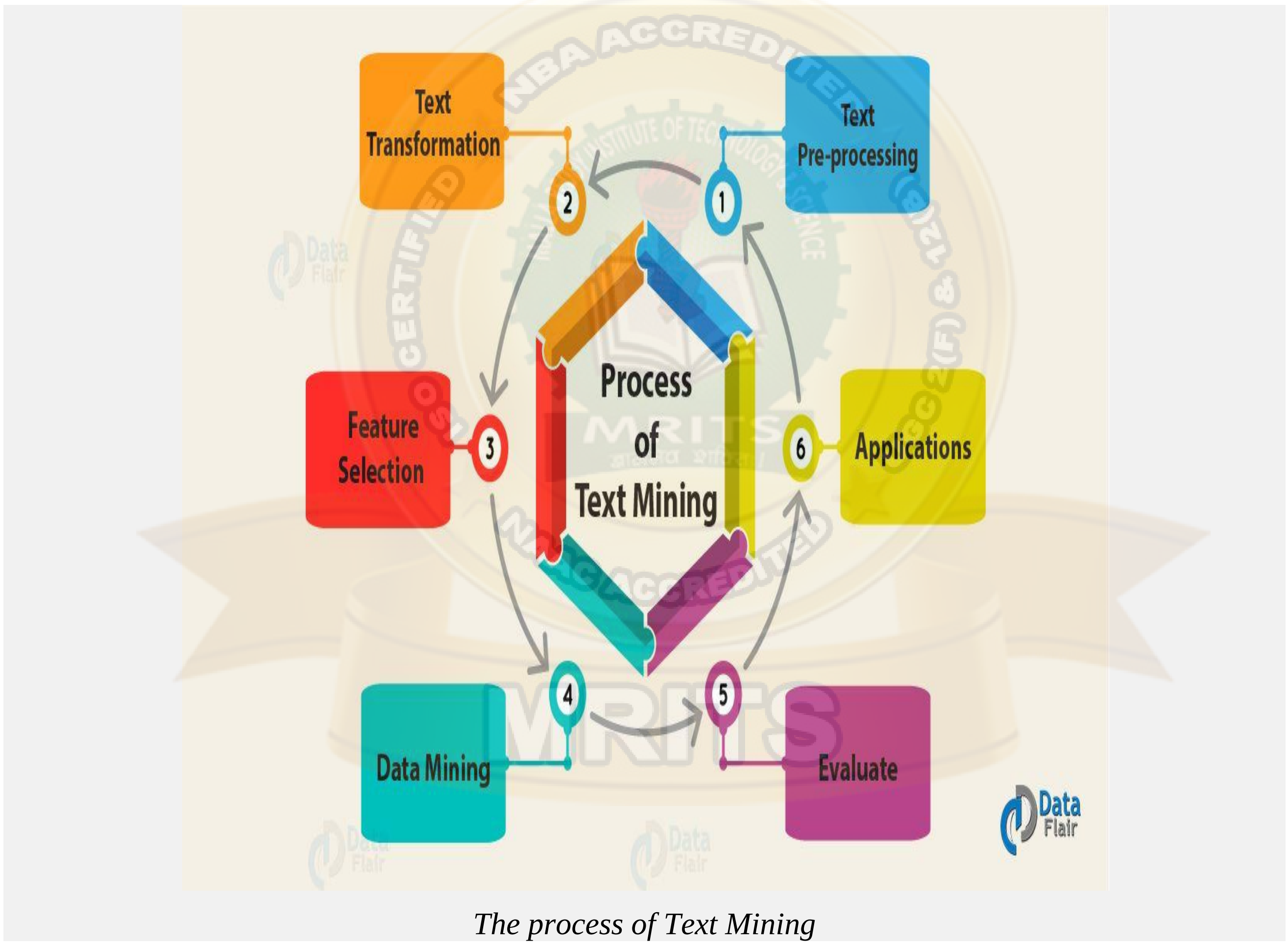
NLP is one of the oldest and most challenging problems. It is the study of human language. So those computers can understand natural languages as humans do. NLP research pursues the vague question of how we understand the meaning of a sentence or a document. What are the indications we use to understand who did what to whom? The role of NLP in text mining is to deliver the system in the information extraction phase as an input.

d. Information Extraction (IE)

Information Extraction is the task of automatically extracting structured information from unstructured. In most of the cases, this activity includes processing human language texts by means of NLP.

4. Text Mining Process

A process of Text mining involves a series of activities to perform to mine the information. These activities are:



The process of Text Mining

a. Text Pre-processing

It involves a series of steps as shown in below:

- **Text Cleanup**

Text Cleanup means removing any unnecessary or unwanted information. Such as remove ads from web pages, normalize text converted from binary formats.

- **Tokenization**

Tokenizing is simply achieved by splitting the text into white spaces.

- **Part of Speech Tagging**

Part-of-Speech (POS) tagging means word class assignment to each token. Its input is given by the tokenized text. Taggers have to cope with unknown words (OOV problem) and ambiguous word-tag mappings.

b. Text Transformation (Attribute Generation)

A text document is represented by the words it contains and their occurrences. Two main approaches to document representation are:

i. Bag of words

ii. Vector Space

c. Feature Selection (Attribute Selection)

Feature selection also is known as variable selection. It is the process of selecting a subset of important features for use in model creation. Redundant features are the one which provides no extra information. Irrelevant features provide no useful or relevant information in any context.

d. Data Mining

At this point, the Text mining process merges with the traditional process. Classic Data Mining techniques are used in the structured database. Also, it resulted from the previous stages.

e. Evaluate

Evaluate the result, after evaluation, the result discard.

f. Applications

Text Mining applies in a variety of areas. Some of the most common areas are

- **Web Mining**

These days web contains a treasure of information about subjects. Such as persons, companies, organizations, products, etc. that may be of wide interest. Web Mining is an application of data mining techniques. That need to discover hidden and unknown patterns from the Web. Web mining is an activity of identifying term implied in a large document collection. It says C which denotes by a mapping i.e. $C \rightarrow p$ [10].

- **Medical**

Users exchange information with others about subjects of interest. Everyone wants to understand specific diseases, to inform about new therapies. Also, these expert forums also represent seismographs for medical. E-mails, e-consultations, and requests for medical advice. That is via the internet have been analyzed using quantitative or qualitative methods.

- **Resume Filtering**

Big enterprises and headhunters receive thousands of resumes from job applicants every day. Extracting information from resumes with high precision and recall is not
easy. Automatically extracting this information can the first step in filtering resumes. Hence,

automating the process of resume selection is an important task.

5. Approaches to Text Mining in Data Mining

Using well-tested methods and understanding the results of text mining. Once a data matrix has been computed from the input documents. And words found in those documents, various well-known analytic techniques. AS it is used for further processing those data including methods for clustering. “Black-box” approaches to text mining and extraction of concepts. There are text mining applications which offer “black-box” methods. That need to extract “deep meaning” from documents with little human effort. These text mining applications rely on proprietary algorithms.

6. Numericizing Text

Following are issues and considerations for Numericizing Text.

i. Large numbers of large documents
Examples of scenarios using large numbers of small were given earlier. But, if your intent is to extract “concepts” from only a few documents that are very large. Then analyses are less powerful because the “number of cases” in this case is very small. While the “number of variables” (extracted words) is very large.
ii. Excluding certain characters, short words, numbers, etc
Excluding numbers, certain characters can be done easily. But before the indexing of the input documents starts. You may also want to exclude “rare words,”. As defined as those that only occur in a small percentage of the processed documents.
iii. Include lists, exclude lists (stop-words)
This is useful when you want to search for particular words. Also, classifying the input documents based on the frequencies. Also, “stop-words,” i.e., terms that are to exclude from the indexing can define. Typically, a default list of English stop words includes “the”, “a”, “of”, “since,”. That is words that are used in the respective language very frequently. But communicate very little unique information about the contents of the document.
iv. Synonyms and phrases
Synonyms, such as “sick” or “ill”, or words that are used in particular phrases. Where they denote unique meaning and can combine for indexing.
For example-
“Microsoft Windows” might be such a phrase. That is a specific reference to the computer operating system. But has nothing to do with the common use of the term “Windows”. As it might, for example, use in descriptions of home improvement projects.
v. Stemming algorithms
An important pre-processing step before indexing of input documents. As it begins is the stemming of words. The term “stemming” refers to the reduction of words to their roots. So that, for example,

different grammatical forms.
vi Support for different languages
Stemming, synonyms, the letters that are permitted in words. Also, are highly language dependent operations. Therefore, support for different languages is important.
7. Incorporating Text Mining Results

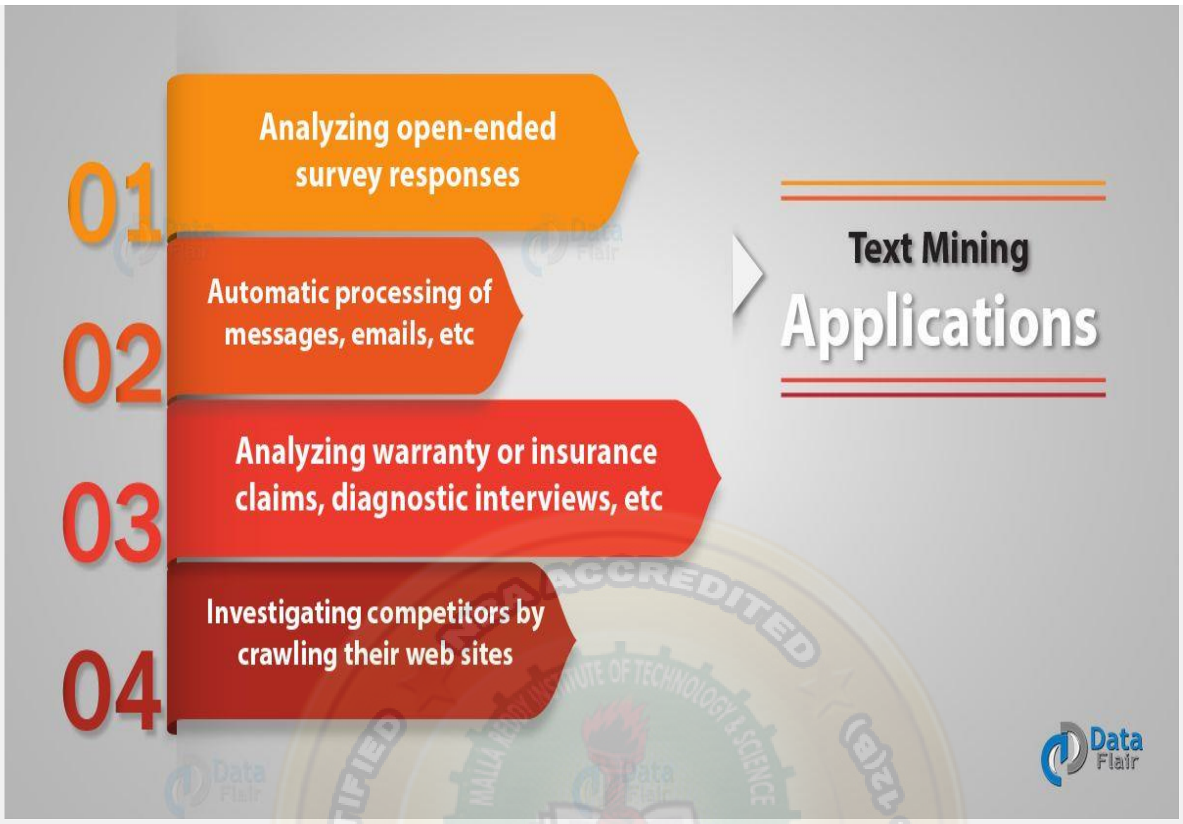
Incorporating Text Mining Results in Data Mining Projects, after significant words have been extracted from a set of input documents. And after singular value decomposition has been applied to extract salient semantic dimensions. Typically the next and most important step is to use the extracted information.

a. Graphics (visual data mining methods)
Depending on the purpose of the analyses, in some instances. We need extraction of semantic dimensions alone. As it can be a useful outcome if it clarifies the underlying structure.

b. Clustering and factoring
You can use cluster analysis methods to identify groups of documents. Also, to identify groups of similar input texts. This type of analysis also useful in the context of market research studies.
For example- of new car owners. You can also use Factor Analysis and Principal Components and Classification Analysis.

c. Predictive data mining
Another possibility is to use the raw as predictor variables in mining projects.
8. Text Mining Applications

Unstructured text is very common. And may represent the majority of information available to a particular research.


01
Analyzing open-ended survey responses
02
Automatic processing of messages, emails, etc
03
Analyzing warranty or insurance claims, diagnostic interviews, etc
04
Investigating competitors by crawling their web sites
Text Mining
Applications
Data Flair

Text Mining Application

a. Analyzing open-ended survey responses

In survey research, it is not uncommon to include various open-ended questions. That is pertaining to the topic under investigation. The idea is to permit respondents to express their "views". Also, opinions without constraining them to particular dimensions or a particular response format.

b. Automatic processing of messages, emails, etc

Another common application is to aid in the automatic classification of texts.

For example-

It is possible to "filter" out automatically most undesirable "junk email". That is based on certain terms or words that are not likely to appear in legitimate messages. Although, instead identify undesirable electronic mail. In this manner, such messages can automatically discard. Such automatic systems for classifying electronic messages can also be useful in applications. That messages need to route to the most appropriate department. At the same time, the emails are screened for inappropriate or obscene messages. That are automatically returned to the sender with a request to remove the offending words or content.

c. Analyzing warranty or insurance claims, diagnostic interviews, etc

In some business domains, the majority of information is collected in open-ended.

For example-

Warranty claims or initial medical interviews can summarize in brief narratives. Increasingly, those notes are collected electronically. So those types of narratives are readily available for input. This information can then usefully exploit to, Likewise, in the medical field. Also, open-ended descriptions by patients of their own symptoms. That might yield useful clues for the actual medical

diagnosis.
d. Investigating competitors by crawling their websites
Another type of application is to process the contents of Web pages in a particular domain.
For example-
You could go to a Web page, and begin “crawling” the links you find there to process all Web pages that are referenced. In this manner, you could derive a list of terms and documents available at that site. Hence determine the most important terms and features that are described.

9. Advantages & Disadvantages of Text Mining

Following are the pros and cons of Text Mining in Data Mining:

a. Advantages of Text Mining

Web mining essentially has many advantages. That make this technology attractive to corporations including the government agencies. This technology has enabled e-commerce to do personalized marketing. As it includes eventually results in higher trade volumes. The government agencies are using this technology to classify threats. The predicting capability can benefit the society by identifying criminal activities. The companies can establish a better customer relationship. Exactly by giving them exactly what they need. Companies can understand the needs of the customer better. Further, they can react to customer needs faster.

b. Disadvantages of Text Mining

Web mining the technology itself doesn’t create issues. Although, this technology when used on data of personal nature might cause concerns.
The most criticized ethical issue involving web mining is the invasion of privacy. Privacy is considered lost when information concerning an individual is obtained. The obtained data will analyze, and clustered to form profiles. Also, the data will make anonymous before clustering. So that no individual can link directly to a profile. But usually, the group profiles are used as if they are personal profiles. Thus these applications de-individualize the users by judging them by their mouse clicks.
De-individualization can define as a tendency of judging and treating people. Particularly, on the basis of group characteristics.
Another important concern is that the companies collecting the data. That is for a specific purpose might use the data for a totally different purpose. And this essentially violates the user’s interests.
The growing trend of selling personal data as a commodity encourages website owners. That is to trade personal data obtained from their site. This trend has increased the amount of data being captured. Also, traded increasing the likeliness of one’s privacy being invaded.
Let’s revise **Data Mining Classification algorithms** & **Cluster analysis**
So, this was all about Text Mining in data Mining. Hope you like our explanation.

What is Web Content Mining:

1. The extraction of certain information from the unstructured raw data text of unknown structures is referred to as **Web content mining**. A set of information extraction tools is brought forward in order to identify and collect **content** items, such as Text Extraction and Wrapper Induction. Learn more in: Web Mining: A Conceptual Overview on Intelligent Information Retrieval Systems
2.It is the process of extraction of information from **web** document, video, audio, text, structured records such as lists and tables. Learn more in: Web Usage Mining: Concept and Applications at a Glance
3.The process of extracting useful information from the **content**s of the **Web** documents. Learn more in: The State of the Art in Web Mining

Web mining can be broadly divided into three categories:

1. Web Content Mining

2. Web Structure Mining

3. Web Usage Mining.

1 .WEB CONTENT MINING

Web content mining targets the knowledge discovery, in which the main objects are the traditional collections of multimedia documents such as images, video, and audio, which are embedded in or linked to the web pages.

It is also quite different from Data mining because Web data are mainly semi-structured and/or unstructured, while Data mining deals primarily with structured data. Web content mining is also different from Text mining because of the semi-structure nature of the Web, while Text mining focuses on unstructured texts. Web content mining thus requires creative applications of Data mining and / or Text mining techniques and also its own unique approaches. In the past few years, there was a rapid expansion of activities in the Web content mining area. This is not surprising because of the phenomenal growth of the Web contents and significant economic benefit of such mining. However, due to the heterogeneity and the lack of structure of Web data, automated discovery of targeted or unexpected knowledge information still present many challenging research problems.

Web content mining could be differentiated from two points of view: Agent-based approach or Database approach. The first approach aims on improving the information finding and filtering. The second approach aims on modeling the data on the Web into more structured form in order to apply standard database querying mechanism and data mining applications to analyze it.

WEB CONTENT MINING PROBLEMS/CHALLENGES:

Data/Information Extraction: Extraction of structured data from Web pages, such as products and search results is a difficult task. Extracting such data allows one to provide services. Two main types of techniques, machine learning and automatic extraction are used to solve this problem.

Web Information Integration and Schema Matching: Although the Web contains a huge amount of data, each web site (or even page) represents similar information differently. Identifying or matching semantically similar data is a very important problem with many practical applications.

Opinion extraction from online sources: There are many online opinion sources, e.g., customer reviews of products, forums, blogs and chat rooms. Mining opinions (especially consumer opinions) is of great importance for marketing intelligence and product benchmarking.

Knowledge synthesis: Concept hierarchies or ontology are useful in many applications. However, generating them manually is very time consuming. A few existing methods that explores the information redundancy of the Web will be presented. The main application is to synthesize and organize the pieces of information on the Web to give the user a coherent picture of the topic domain.

Segmenting Web pages and detecting noise: In many Web applications, one only wants the main content of the Web page without advertisements, navigation links, copyright notices. Automatically segmenting Web page to extract the main content of the pages is interesting problem.

All these tasks present major research challenges and their solutions.

2.WEB STRUCTURE MINING:

Web Structure Mining focuses on analysis of the link structure of the web and one of its purposes is to identify more preferable documents. The different objects are linked in some way. The intuition is that a hyperlink from document A to document B implies that the author of document. A thinks document B contains worthwhile information. Web structure mining helps in discovering similarities between web sites or discovering important sites for a particular topic or discipline or in discovering web communities.

Simply applying the traditional processes and assuming that the events are independent can lead to wrong conclusions. However, the appropriate handling of the links could lead to potential correlations, and then improve the predictive accuracy of the learned models.

The goal of Web structure mining is to generate structural summary about the Web site and Web page. Technically, Web content mining mainly focuses on the structure of inner-document, while Web structure mining tries to discover the link structure of the hyperlinks at the inter-document level. Based on the topology of the hyperlinks, Web structure mining will categorize the Web pages and generate the information, such as the similarity and relationship between different Web sites.

Web structure mining can also have another direction – discovering the structure of Web document itself. This type of structure mining can be used to reveal the structure (schema) of Web pages; this would be good for navigation purpose and make it possible to compare/integrate Web page schemes. This type of structure mining will facilitate introducing database techniques for accessing information in Web pages by providing a reference schema.

3.WEB USAGE MINING:

Web Usage Mining focuses on techniques that could predict the behavior of users while they are interacting with the WWW. Web usage mining, discover user navigation patterns from web data, tries to discovery the useful information from the secondary data derived from the interactions of the users while surfing on the Web. Web usage mining collects the data from Web log records to discover user access patterns of web pages. There are several available research projects and commercial tools that analyze those patterns for different purposes. The insight knowledge could be utilized in personalization, system improvement, site modification, business intelligence and usage characterization.

The only information left behind by many users visiting a Web site is the path through the pages they have accessed. Most of the Web information retrieval tools only use the textual information, while they ignore the link information that could be very valuable. In general, there are mainly four kinds of data mining techniques applied to the web mining domain to discover the user navigation pattern:

Association Rule mining

Sequential pattern

Clustering

Classification

WEB USAGE MINING

INTRODUCTION

Users interact frequently with different web sites and can access plenty of information on WWW. The World Wide Web is growing continuously and huge amount of data is generated due to users' numerous interactions with web sites. Web Usage Mining is the application of data mining techniques to discover the useful and interesting patterns from web usage data. It supports to know frequently accessed pages, predict user navigation, improve web site structure etc. The web usage data consists of the data from web server logs, browser logs, proxy server logs and user profiles. In Web Usage Mining, data mining techniques are applied to pre-processed web log data in order to find interesting and useful patterns. Visitors' browsing behavior is recorded into web server log. By analyzing log files, different questions can be answered such as: • What pages are being accessed frequently? • From what search engine are visitors coming? • Which browser and operating systems are most commonly used by visitors? • What are the most recent visits per page? • Who is visiting which page? E-commerce has provided effective way of doing business by electronic transactions through internet. For Web Usage Mining in e-commerce, the data of customer profile, inventory and demographic information from other relational databases are integrated with web usage data and visitors' behavior patterns can be discovered by applying data mining techniques such as Association Rules, Sequential Analysis, Clustering and Classification. Web Usage Mining can help e-commerce companies to improve the web site, attract visitors, to provide personalized and adaptive service to regular user, identify potential customers for e-commerce, supporting business intelligence and marketing decisions etc

DESCRIPTION OF WEB USAGE MINING PROCESS

The Web Usage Mining is the application of data mining technique to discover the useful patterns from web usage data. It can discover the user access patterns by mining log files and associated data of particular web site. Figure3.1 shows the process of Web Usage Mining consisting steps Data Collection, Pre-processing, Pattern Discovery and Pattern Analysis.

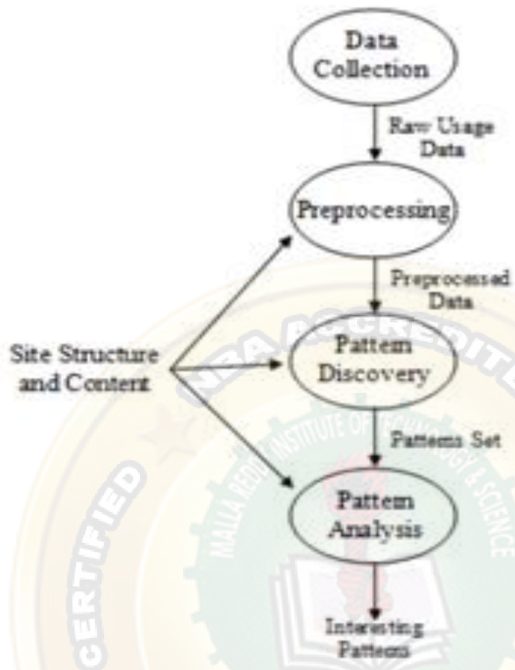


Figure 3.1: The Process of Web Usage Mining

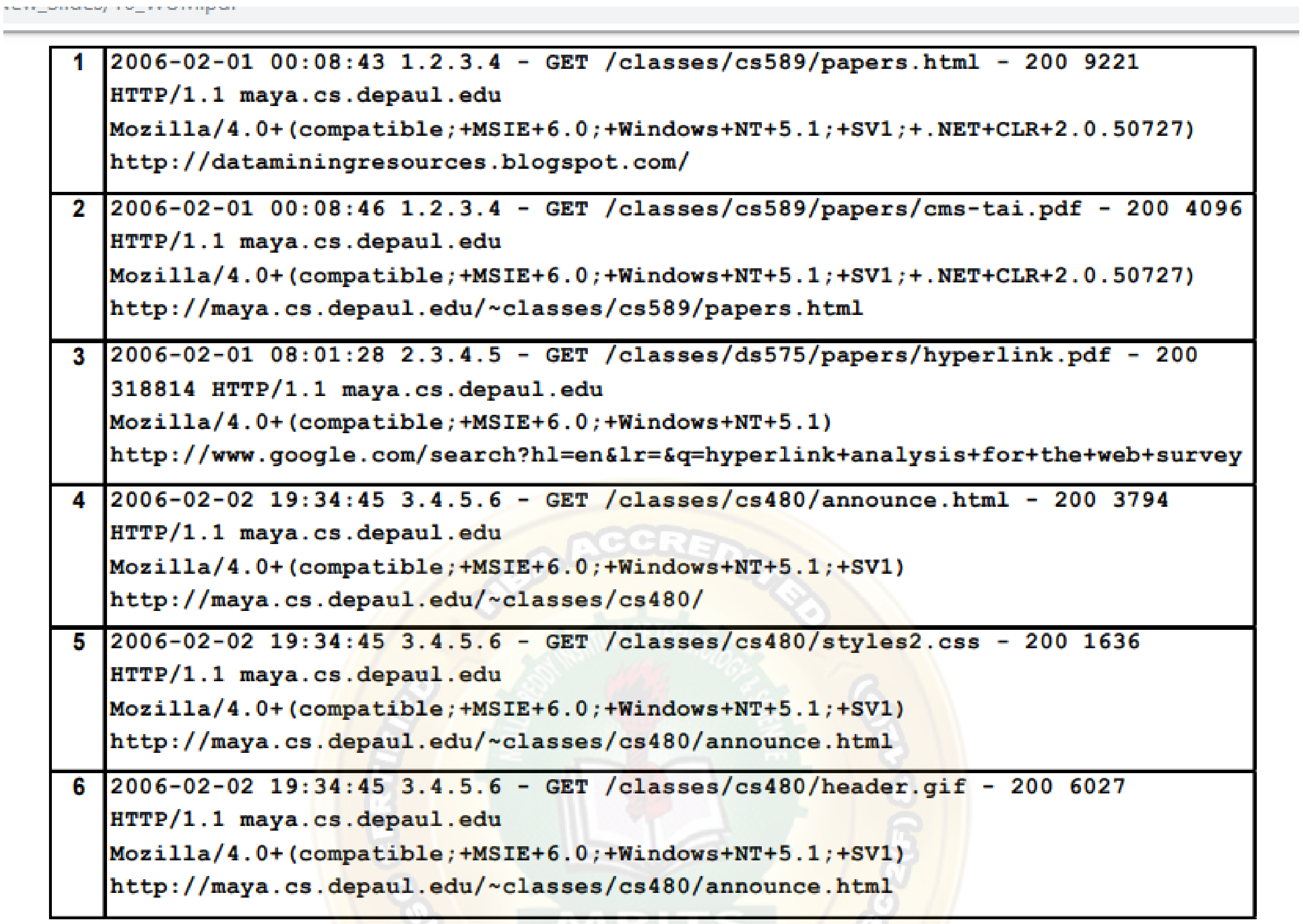
The web log files on the web server are major source of data for Web Usage Mining. When user requests resources of web server, each request is recorded in the web logfile on web server. As a consequence, users browsing behavior is recorded into the web log file. In Web Usage Mining, data can be collected from server log files that include web server access logs and application server logs. Data is also obtained from site files and operational databases. The data collected in web log file is incomplete and not suitable for mining directly. Pre-processing is necessary to convert the data into suitable form for pattern discovery. Pre-processing can provide accurate, concise data for data mining. Data pre-processing, includes data cleaning, user identification, user sessions identification, path completion and data integration [45]. To discover the novel, potentially useful and interesting information, several methods and data mining algorithms are applied such as Path Analysis, Association Rules, Sequential Patterns, Clustering, Classification etc. Pattern analysis techniques are used to highlight overall patterns in data and to filter out uninteresting patterns. The techniques like Knowledge Query Mechanism, OLAP and visualization are used for pattern analysis.

Web Server Log

The user navigates the pages of the website to access the information. The user agent

or browser sends requests to the website server whenever new pages or resources are

accessed by the user. The server receives and handles the request and data for the requested resource is sent back to the browser. The information related to that resource requests are recorded on the server in a file. This file is called an internet server log, or a web log. The log files keep a lot of information about each user access to the web server. Each line or entry in the web log represents a request for a resource. The different web server support different log format. The exact content of an entry varies from log format to log format. Nearly all the server log file contains information such as Visitor’s IP address, access date and time, URL of page accessed, the status code returned by the server, bytes transferred to the user etc. Many different web log formats like NCSA Common, Microsoft IIS Log, NCSA Combined etc. are in use. Almost every internet server platform has its own unique format, and some formats can in addition be customized by the server administrator. Figure 3.2 represents the sample web server log entries.



1	2006-02-01 00:08:43 1.2.3.4 - GET /classes/cs589/papers.html - 200 9221 HTTP/1.1 maya.cs.depaul.edu Mozilla/4.0+(compatible;+MSIE+6.0;+Windows+NT+5.1;+SV1;+.NET+CLR+2.0.50727) http://dataminingresources.blogspot.com/
2	2006-02-01 00:08:46 1.2.3.4 - GET /classes/cs589/papers/cms-tai.pdf - 200 4096 HTTP/1.1 maya.cs.depaul.edu Mozilla/4.0+(compatible;+MSIE+6.0;+Windows+NT+5.1;+SV1;+.NET+CLR+2.0.50727) http://maya.cs.depaul.edu/~classes/cs589/papers.html
3	2006-02-01 08:01:28 2.3.4.5 - GET /classes/ds575/papers/hyperlink.pdf - 200 318814 HTTP/1.1 maya.cs.depaul.edu Mozilla/4.0+(compatible;+MSIE+6.0;+Windows+NT+5.1) http://www.google.com/search?hl=en&lr=&q=hyperlink+analysis+for+the+web+survey
4	2006-02-02 19:34:45 3.4.5.6 - GET /classes/cs480/announce.html - 200 3794 HTTP/1.1 maya.cs.depaul.edu Mozilla/4.0+(compatible;+MSIE+6.0;+Windows+NT+5.1;+SV1) http://maya.cs.depaul.edu/~classes/cs480/
5	2006-02-02 19:34:45 3.4.5.6 - GET /classes/cs480/styles2.css - 200 1636 HTTP/1.1 maya.cs.depaul.edu Mozilla/4.0+(compatible;+MSIE+6.0;+Windows+NT+5.1;+SV1) http://maya.cs.depaul.edu/~classes/cs480/announce.html
6	2006-02-02 19:34:45 3.4.5.6 - GET /classes/cs480/header.gif - 200 6027 HTTP/1.1 maya.cs.depaul.edu Mozilla/4.0+(compatible;+MSIE+6.0;+Windows+NT+5.1;+SV1) http://maya.cs.depaul.edu/~classes/cs480/announce.html

Figure 3.2: Sample Web Server Log Entries

Web servers can be configured to write different fields into the log file in different formats. The most common fields used by web servers are the followings: IP Address, Login Name, User Name, Timestamp, Request, Status, Bytes, Referrer, User agent.

The explanation of fields included in sample web log entries of Figure 3.2 is as under.

1) IP Address: IP address of the remote host that made the request. If DNS host name is available then it will be substituted in place of IP address.

2) Login Name: Login name of the user making the HTTP request. If there is no value, a dash (-) is substituted instead of the login name.

3) User Name: The name of the authenticated user who access the server by making the HTTP request, anonymous users are represented by a dash (–)

4) Timestamp: Time-stamp indicates date and time of the visit with the web server's

time zone.

5) Request: It indicates HTTP request. It consists of three pieces of information: the HTTP method, the requested resource and HTTP protocol version.

6) Status Code: The success or failure of the http request is indicated by HTTP status code. For example, "200" is success, “404” for File Not Found, “500” for Internal Server Error.

7) Bytes transferred: It indicates number of bytes transferred to the user as part of the HTTP request.

8) Referrer: The URL of the page the visitor was on when he clicked to come to current page.

9) User agent: It represents the software used by the visitor to access the site. It could be a web browser, web robot or link checker etc. It also indicates the operating system used by the client. MSIE 7.0 means Internet Explorer 7.

Windows NT 6.0 indicates Windows Vista. Windows NT6.1 indicates Windows7. A visitor request for web page can generate different web log records. The first record documents the request of user to particular web page and remaining records are created for requests made to obtain images on the requested page. The quantity of information stored in the log files is very large. The raw web log data is usually diverse and incomplete and difficult to be used directly for further pattern mining. The web log data can be preprocessed in order to obtain session information for all users and data reduction techniques can be applied to web log data in order to obtain raw sets for data mining. Before it can be used for anything at all, data must be extracted from the log file and analyzed.

UNSTRUCTURED DATA MINING TEXT:

Unstructured Data Mining Text document is the form of unstructured data. Most of the data that is available on web is unstructured data. The research of applying data mining techniques to unstructured data is known as knowledge discovery in texts [3].

1) Information Extraction: To extract information from unstructured data that is present on web pattern matching is used. It traces the keywords and phrases and then finds out the connection of keywords within text. When large volume of text is there then the technique is very useful. Information extraction transforms unstructured text to more structured form. First, from extracted data the information is mined, then using different types of rules, the missed out information is found. Information extraction making incorrect predictions on data is discarded [4] [5].

2) Topic Tracking: This technique checks the documents viewed by the user and studies the user profile. It predicts the documents related to users interest. The topic tracking applied by yahoo, user give a keyword and if anything related to keyword pops then the user is informed about that. This technique can be applied by many fields. The two fields where it is used is medical field and education field. In medical field doctors easily come to know about the latest treatments. In education field it is used to find out the latest reference for research related work. The disadvantage of the technique is that when we search for our topic then it may provide us with information which is not related to our topic [4] [6].

3) Summarization: The technique is used to reduce the length of the document by maintaining the important points. It helps the user to decide whether to read the topic or not. The time taken by the technique to summarize the document is less than the time taken by the user to read the first paragraph [4]. The summarization technique uses two methods that is the extractive method and the abstractive method. The extractive method selects a subset of phrases, sentences and words to form the summary from the original text. The abstractive method builds an internal semantic representation and then uses natural language generation technique to create the summary. This summary may contain words which are not present in the original document [6].

4) Categorization: This technique identifies the main theme by placing the documents in a predefined set of group. The technique counts the number of words in the document and this decides the main topic. According to the topic the rank is given to the document. The documents with majority contents on particular topic are given first rank. This technique helps in providing customer support to the industries and business [4] [5].

5) Clustering: The technique is used to group similar documents. In this grouping of documents is not done on the basis of predefined topics. It is done on fly basis. Some documents may appear in different group. As a result useful documents are not omitted from search results. This technique helps user to select the topic of interest [4].

6) Information Visualization: Visualization utilizes feature extraction and key term indexing. Documents having similarity are found out through visualization. Large textual materials are represented as visual maps or hierarchy where browsing facility is allowed. It helps in visually analyzing the content. The user can interact by scaling, zooming and creating sub maps of the graphs

EPISODE DISCOVERY PROCESS:

The flow of messages posted in blogs and social networks is an important and valuable source of information that can be analyzed, modeled (through the extraction of hidden relationships) and from which information can be predicted, which is the focus of our work. For example, companies may be interested in the prediction of what will be said about them in social networks. Similarly, this prediction can be a way to recommend items. We consider that the sooner an event is predicted, the more useful this prediction is for the company or the person concerned, since this allows to have enough time to act before the occurrence of the event. Predicting distant events is thus the focus of our work. Temporal data mining is related to the mining of sequential patterns ordered by a given criterion such as time or position (Laxman and Sastry, 2006). Episode mining is the appropriate pattern discovery task related to the case the data is made up of a single long sequence. An episode is a temporal pattern made up of "relatively close" partially ordered items (or events), which often appears throughout the sequence or in a part of it (Mannila et al., 1997). When the order of items is total, the episode is said to be serial.

• KDD process of analyzing alarm sequences

• Discovery and post-processing of large pattern collections

• TASA, Telecommunication Alarm Sequence Analyzer

Goal: discovery of useful and interesting knowledge

1. Understanding the domain

2. Collecting and cleaning data

3. Discovery of patterns

4. Presentation and analysis of results

5. Making onclusions and utilizing results Pattern discovery is only a part of the KDD process (but the central one)

Questions implied by the KDD process model:

• How to know what could be interesting?

• How to ensure that correct and reliable discoveries can be made?

• How to discover potentially interesting patterns?

• How to make the results understandable for the user?

• How to use the results?

HIERARCHY OF CATEGORIES:

Concept Hierarchy reduce the data by collecting and replacing low level concepts (such as numeric values for the attribute age) by higher level concepts (such as young, middle-aged, or senior).

Concept hierarchy generation for numeric data is as follows:

- **Binning (see sections before)**
- **Histogram analysis (see sections before)**
- **Clustering analysis (see sections before)**

- **Entropy-based discretization**
- **Segmentation by natural partitioning**
- **Binning**
  - In binning, first sort data and partition into (equi-depth) bins then one can smooth by bin means, smooth by bin median, smooth by bin boundaries, etc.
- **Histogram analysis**
  - Histogram is a popular data reduction technique
  - Divide data into buckets and store average (sum) for each bucket
  - Can be constructed optimally in one dimension using dynamic programming
  - Related to quantization problems.
- **Clustering analysis**
  - Partition data set into clusters, and one can store cluster representation only
  - Can be very effective if data is clustered but not if data is "smeared"
  - Can have hierarchical clustering and be stored in multi-dimensional index tree structures
- **Entropy-based discretization**
  - Given a set of samples S, if S is partitioned into two intervals S1 and S2 using boundary T, the entropy after partitioning

    $$E(S,T) = \frac{|S_1|}{|S|} Ent(S_1) + \frac{|S_2|}{|S|} Ent(S_2)$$

    is

    – S1 & S2 correspond to samples in S satisfying conditions A<v &amp;="" a="">=v
  - The boundary that minimizes the entropy function over all possible boundaries is selected as a binary discretization.
  - The process is recursively applied to partitions obtained until some stopping criterion is met, e.g., Ent (S)- E(T,S)>δ
  - Experiments show that it may reduce data size and improve classification accuracy
- **Segmentation by natural partitioning**
  - 3-4-5 rule can be used to segment numeric data into relatively uniform, "natural" intervals.
  - If an interval covers 3, 6, 7 or 9 distinct values at the most significant digit, partition the range into 3 equi-width intervals
  - If it covers 2, 4, or 8 distinct values at the most significant digit, partition the range into 4 intervals
  - If it covers 1, 5, or 10 distinct values at the most significant digit, partition the range into 5 intervals

Concept hierarchy generation for categorical data is as follows:

- **Specification of a set of attributes, but not of their partial ordering**
  - Auto generate the attribute ordering based upon observation that attribute defining a high level concept has a smaller # of distinct values than an attribute defining a lower level concept
  - Example : country (15), state_or_province (365), city (3567), street (674,339)
- **Specification of only a partial set of attributes**
  - Try and parse database schema to determine complete hierarchy

What is Text Clustering?

Automatic document organization, topic extraction, information retrieval and filtering all have one thing in common. They require text clustering (sometimes also known as document clustering) to be done quickly and accurately.

If you've never heard of text clustering, this post will explain what it is, what it does, and how its currently being used to aid businesses. We'll also briefly discuss how a business could employ text clustering too!

Text clustering definition:

First, let's define text clustering. Text clustering is the application of cluster analysis to text-based documents. It uses machine learning and natural language processing (NLP) to understand and categorize unstructured, textual data.

First, let's define text clustering. Text clustering is the application of cluster analysis to text-based documents. It uses machine learning and natural language processing (NLP) to understand and categorize unstructured, textual data.

How it works Typically, descriptors (sets of words that describe topic matter) are extracted from the document first. Then they are analyzed for the frequency in which they are found in the document compared to other terms. After which, clusters of descriptors can be identified and then auto-tagged.

From there, the information can be used in any number of ways. Google's search engine is probably the best and most widely known example. When you search for a term on Google, it pulls up pages that apply to that term, but have you ever wondered how Google can analyze billions of web pages to deliver an accurate and fast result?

It's because of text clustering! Google's algorithm breaks down unstructured data from web pages and turns it into a matrix model, tagging pages with keywords that are then used in search results!

Example
To help you understand the process, it's best to visualize an example:
Let's simulate how text clustering would analyze (and tag) this sentence.
First, all punctuation is removed:
let us simulate how text clustering would analyze and tag this sentence

Then, all but the sentence's descriptors are removed:

simulate how text clustering analyze tag sentence

At this point, its harder to visualize as a computer will be assigning each word a weighted value for use in tagging.

Business use cases

Perhaps one of the best parts of text clustering is its ability to be used in a wide variety of business settings. Text clustering can be used anywhere from product development to customer support. Let’s take a look at a few examples in which a business could employ text clustering.

1. Creating a product roadmap

Your customers and target audience are talking all over the web about the products and features they want, but, traditionally, it’s difficult to aggregate all the data and turn it into an actionable report. It’s hard to know just how many really want a feature based on a handful of reviews and forum posts.

But with text clustering, all of your customer and target audience’s reviews can be analyzed and used to create a roadmap of features and products they’ll love!

You can even analyze competitor reviews to find potential deal breakers as well!

2. Identify recurring support issues

Your customer support team gets asked the same questions day in and day out. But, it’s hard to truly analyze the pain points your customers may have when adopting products and address them correctly. Text clustering will enable you to not only see how frequent (or infrequent) an issue is, but also may help identify the root of the issue with additional tags.

3. Creating better marketing copy

Another use case for text clustering is in your marketing copy. Depending on your organization you may have run thousands of different ads and have plenty of data with it. But understanding how the language of the ad impacted performance can be tough.

It’s difficult to spot trends in unstructured data such as marketing copy which is where text clustering can come into play. It can analyze and break down the topics and words which have the highest conversion rates enabling you to create highly relevant, highly converting web copy.