# CS8091 BIG DATA ANALYTICS

## UNIT I- INTRODUCTION TO BIG DATA

## QUESTION BANK

## PART-A

**1. What is Big Data?**

Big data is a field that treats ways to analyze, systematically extract information from, or otherwise deal with data sets that are too large or complex to be dealt with by traditional data-processing application software.

**2. List out the best practices of Big Data Analytics.**

1. Start at the End
2. Build an Analytical Culture.
3. Re-Engineer Data Systems for Analytics
4. Focus on Useful Data Islands.
5. Iterate often.

**3. Write down the characteristics of Big Data Applications.**

**a)** Data Throttling

b) Computation- restricted throttling

c) Large Data Volumes

d) Significant Data Variety

e) Benefits from Data parallelization

**4. Write down the four computing resources of Big Data Storage.**

a) Processing Capability

b) Memory

c) Storage

d) Network

**5. What is HDFS?**

Apache Hadoop is a collection of open-source software utilities that facilitate using a network of many computers to solve problems involving massive amounts of data and computation. It provides a software framework for distributed storage and processing of big data using the MapReduce programming model.

**6. What is MapReduce?**

MapReduce is a processing technique and a program model for distributed computing based on java. The MapReduce algorithm contains two important tasks, namely Map

and Reduce. Map takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key/value pairs).

**7. What is YARN?**

YARN is an Apache Hadoop technology and stands for Yet Another Resource Negotiator. YARN is a large-scale, distributed operating system for big data applications. YARN is a software rewrite that is capable of decoupling MapReduce's resource management and scheduling capabilities from the data processing component.

**8. What is Map Reduce Programming Model?**

MapReduce is a programming model and an associated implementation for processing and generating big data sets with a parallel, distributed algorithm on a cluster. The model is a specialization of the split-apply-combine strategy for data analysis.

**9. What are the characteristics of big data?**

Big data can be described by the following characteristics

**Volume -** The quantity of data generated and stored data. The size of the data determines the value and potential insight- and whether it can actually be considered big data or not.

**Variety -**The type and nature of the data. This helps people who analyze it to effectively use the resulting insight.

**Velocity -**In this context, the speed at which the data is generated and processed to meet the demands and challenges that lie in the path of growth and development.

**Variability-** Inconsistency of the data set can hamper processes to handle and manage it.

**Veracity-**The data quality of captured data can vary greatly, affecting the accurate analysis

**10. What is Big Data Platform?**

• Big Data Platform is integrated IT solution for Big Data management which combines several software systems, software tools and hardware to provide easy to use tools system to enterprises.

• It is a single one-stop solution for all Big Data needs of an enterprise irrespective of size and data volume. Big Data Platform is enterprise class IT solution for developing, deploying and managing Big Data.

**PART -B & C**

**1. What is Bigdata? Describe the main features of a big data in detail.**

## Basics of Bigdata Platform

- Big Data platform is IT solution which combines several Big Data tools and utilities into one packaged solution for managing and analyzing Big Data.

- **Big data platform** is a type of IT solution that combines the features and capabilities of several **big data** application and utilities within a single solution.
- It is an enterprise class IT **platform** that enables organization in developing, deploying, operating and managing a **big data** infrastructure /environment.

**Big Data Platform**

- Big Data Platform is integrated IT solution for Big Data management which combines several software systems, software tools and hardware to provide easy to use tools system to enterprises.
- It is a single one-stop solution for all Big Data needs of an enterprise irrespective of size and data volume. Big Data Platform is enterprise class IT solution for developing, deploying and managing Big Data.
- There are several Open source and commercial Big Data Platform in the market with varied features which can be used in Big Data environment.
- Big data platform is a type of IT solution that combines the features and capabilities of several big data application and utilities within a single solution.
- It is an enterprise class ITplatformthat enables organization in developing, deploying, operating and managing abig datainfrastructure /environment.
- Big data platform generally consists of big data storage, servers, database, big data management, business intelligence and other big data management utilities
- It also supports custom development, querying and integration with other systems.
- The primary benefit behind a big data platform is to reduce the complexity of multiple vendors/ solutions into a one cohesive solution.
- Big data platform are also delivered through cloud where the provider provides an all inclusive big data solutions and services.

## Features of Big Data Platform

Here are most important features of any good Big Data Analytics Platform:

a) Big Data platform should be able to accommodate new platforms and tool based on the business requirement. Because business needs can change due to new technologies or due to change in business process.
b) It should support linear scale-out
c) It should have capability for rapid deployment
d) It should support variety of data format
e) Platform should provide data analysis and reporting tools
f) It should provide real-time data analysis software

g) It should have tools for searching the data through large data sets

Big data is a term for data sets that are so large or complex that traditional data processing applications are inadequate.

Challenges include

- Analysis,
- Capture,
- Data Curation,
- Search,
- Sharing,
- Storage,
- Transfer,
- Visualization,
- Querying,
- Updating

Information Privacy.

- The term often refers simply to the use of predictive analytics or certain other ***advancedmethods*** to extract value from data, and seldom to a particular size of data set.

- **ACCURACY** in big data may lead to more confident decision making, and better decisions can result in greater operational efficiency, cost reduction and reduced risk.
- Big data usually includes data sets with sizes beyond the ability of commonly used
- software tools to capture, curate, manage, and process data within a tolerable elapsed
- time. Big data "size" is a constantly moving target.
- Big data requires a set of techniques and technologies with new forms of integration to
- reveal insights from datasets that are diverse, complex, and of a massive scale

**List of BigData Platforms**

a) Hadoop
b) Cloudera
c) Amazon Web Services
d) Hortonworks
e) MapR
f) IBM Open Platform
g) Microsoft HDInsight
h) Intel Distribution for Apache Hadoop
i) Datastax Enterprise Analytics

j) Teradata Enterprise Access for Hadoop
k) Pivotal HD

a) Hadoop

- Hadoop is open-source, Java based programming framework and server software which is used to save and analyze data with the help of 100s or even 1000s of commodity servers in a clustered environment.
- Hadoop is designed to storage and process large datasets extremely fast and in fault tolerant way.
- Hadoop uses HDFS (Hadoop File System) for storing data on cluster of commodity computers. If any server goes down it know how to replicate the data and there is no loss of data even in hardware failure.
- Hadoop is Apache sponsored project and it consists of many software packages which runs on the top of the Apache Hadoop system.
- Top Hadoop based Commercial Big Data Analytics Platform
- Hadoop provides set of tools and software for making the backbone of the Big Data analytics system.
- Hadoop ecosystem provides necessary tools and software for handling and analyzing Big Data.
- On the top of the Hadoop system many applications can be developed and plugged-in to provide ideal solution for Big Data needs.

b) **Cloudera**

- Cloudra is one of the first commercial Hadoop based Big Data Analytics Platform offering Big Data solution.
- Its product range includes Cloudera Analytic DB, Cloudera Operational DB, Cloudera Data Science & Engineering and Cloudera Essentials.
- All these products are based on the Apache Hadoop and provides real-time processing and analytics of massive data sets.

c) **Amazon Web Services**

- Amazon is offering Hadoop environment in cloud as part of its Amazon Web Services package.
- AWS Hadoop solution is hosted solution which runs on Amazon's Elastic Cloud Compute and Simple Storage Service (S3).
- Enterprises can use the Amazon AWS to run their Big Data processing analytics in the cloud environment.
- Amazon EMR allows companies to setup and easily scale Apache Hadoop, Spark, HBase, Presto, Hive, and other Big Data Frameworks using its cloud hosting environment.

d) **Hortonworks**

- Hortonworks is using 100% open-source software without any propriety software. Hortonworks were the one who first integrated support for Apache HCatalog.
- The Hortonworks is a Big Data company based in California.
- This company is developing and supports application for Apache Hadoop.

Hortonworks Hadoop distribution is 100% open source and its enterprise ready with following features:

- Centralized management and configuration of clusters
- Security and data governance are built in feature of the system
- Centralized security administration across the system

e) **MapR**

- MapR is another Big Data platform which us using the Unix file system for handling data.
- It is not using HDFS and this system is easy to learn anyone familiar with the Unix system.
- This solution integrates Hadoop, Spark, and Apache Drill with a real-time data processing feature.

f) **IBM Open Platform**

- IBM also offers Big Data Platform which is based on the Hadoop eco-system software.
- IBM well knows company in software and data computing.

It uses the latest Hadoop software and provides following features (IBM Open Platform Features):

- Based on 100% Open source software
- Native support for rolling Hadoop upgrades
- Support for long running applications within YEARN.
- Support for heterogeneous storage which includes HDFS for in-memory and SSD in addition to HDD
- Native support for Spark, developers can use Java, Python and Scala to written program
- Platform includes Ambari, which is a best tool for provisioning, managing & monitoring Apache Hadoop clusters
- IBM Open Platform includes all the software of Hadoop ecosystem e.g. HDFS, YARN, MapReduce, Ambari, Hbase, Hive, Oozie, Parquet, Parquet Format, Pig, Snappy, Solr, Spark, Sqoop, Zookeeper, Open JDK, Knox, Slider
- Developer can download the trial Docker Image or Native installer for testing and learning the system
- Application is well supported by IBM technology team

g) **Microsoft HDInsight**

- The Microsoft HDInsight is also based on the Hadoop distribution and it's a commercial Big Data platform from Microsoft.
- Microsoft is software giant which is into development of windows operating system for Desktop users and Server users.
- This is the big Hadoop distribution offering which runs on the Windows and Azure environment.
- It offers customized, optimized open source Hadoop based analytics clusters which uses Spark, Hive, MapReduce, HBase, Strom, Kafka and R Server which runs on the Hadoop system on windows/Azure environment.

**2. List the main characteristics of Big Data.**

**Characteristics of Big Data**

(i) Volume – The name Big Data itself is related to a size which is enormous. Size of data plays a very crucial role in determining value out of data. Also, whether a particular data can actually be considered as a Big Data or not, is dependent upon the volume of data. Hence, 'Volume' is one characteristic which needs to be considered while dealing with Big Data.

(ii) Variety – The next aspect of Big Data is its variety.Variety refers to heterogeneous sources and the nature of data, both structured and unstructured. During earlier days, spreadsheets and databases were the only sources of data considered by most of the applications. Nowadays, data in the form of emails, photos, videos, monitoring devices, PDFs, audio, etc. are also being considered in the analysis applications. This variety of unstructured data poses certain issues for storage, mining and analyzing data.

(iii) Velocity – The term 'velocity' refers to the speed of generation of data. How fast the data is generated and processed to meet the demands, determines real potential in the data.

Big Data Velocity deals with the speed at which data flows in from sources like business processes, application logs, networks, and social media sites, sensors, Mobile devices, etc. The flow of data is massive and continuous.

(iv) Variability – This refers to the inconsistency which can be shown by the data at times, thus hampering the process of being able to handle and manage the data effectively.

Benefits of Big Data Processing

Ability to process Big Data brings in multiple benefits, such as-

**Businesses can utilize outside intelligence while taking decisions**

Access to social data from search engines and sites like Facebook, twitter are enabling organizations to fine tune their business strategies.

**Improved customer service**

Traditional customer feedback systems are getting replaced by new systems designed with Big Data technologies. In these new systems, Big Data and natural language processing technologies are being used to read and evaluate consumer responses.

**Early identification of risk to the product/services, if any**

Better operational efficiency

Big Data technologies can be used for creating a staging area or landing zone for new data before identifying what data should be moved to the data warehouse. In addition, such integration of Big Data technologies and data warehouse helps an organization to offload infrequently accessed data.

**3. Explain in detail about Nature of Data and its applications.**

**Data**

- **Data** is a set of values of qualitative or quantitative variables; restated, pieces of **data** are individual pieces of information.
- **Data** is measured, collected and reported, and analyzed, whereupon it can be visualized using graphs or images.

**Properties of Data**

For examining the properties of data, reference to the various definitions of data. Reference to these definitions reveals that following are the properties of data:

a) Amenability of use
b) Clarity
c) Accuracy
d) Essence
e) Aggregation
f) Compression
g) Refinement

.**Amenability of use:** From the dictionary meaning of data it is learnt that data are facts used in deciding something. In short, data are meant to be used as a base for arriving at definitive conclusions.

**a) Clarity:** Data are a crystallized presentation. Without clarity, the meaning desired to be communicated will remain hidden.

**b) Accuracy:** Data should be real, complete and accurate. Accuracy is thus, an essential property of data.

**c) Essence:** A large quantities of data are collected and they have to be Compressed and refined. Data so refined can present the essence or derived qualitative value, of the matter.

**d) Aggregation:** Aggregation is cumulating or adding up.

e) **Compression:** Large amounts of data are always compressed to make them more meaningful. Compress data to a manageable size.Graphs and charts are some examples of compressed data.

f) **Refinement:** Data require processing or refinement. When refined, they are capable of leading to conclusions or even generalizations. Conclusions can be drawn only when data are processed or refined.

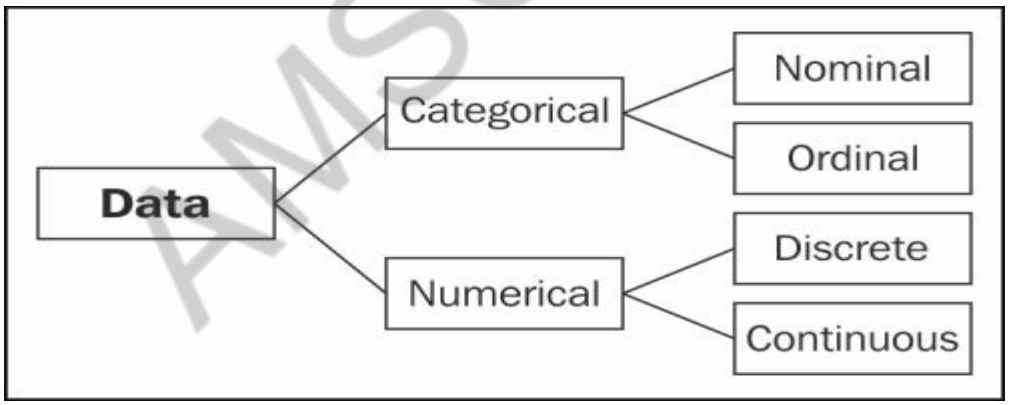
## TYPES OF DATA

- In order to understand the nature of data it is necessary to categorize them into various types.
- Different categorizations of data are possible.
- The first such categorization may be on the basis of disciplines, e.g., Sciences, Social Sciences, etc. in which they are generated.
- Within each of these fields, there may be several ways in which data can be categorized into types.

There are four types of data:

- Nominal
- Ordinal
- Interval
- Ratio

Each offers a unique set of characteristics, which impacts the type of analysis that can be performed.






The distinction between the four types of scales center on three different characteristics:

1. The **order** of responses – whether it matters or not
2. The **distance between observations** – whether it matters or is interpretable
3. The presence or inclusion of a **true zero**

## Nominal Scales

Nominal scales measure categories and have the following characteristics:

- **Order:** The order of the responses or observations does not matter.
- **Distance:** Nominal scales do not hold distance. The distance between a 1 and a 2 is not the same as a 2 and 3.

- **True Zero:** There is no true or real zero. In a nominal scale, zero is uninterruptable.

**Appropriate statistics for nominal scales:** mode, count, frequencies
**Displays:** histograms or bar charts

## Ordinal Scales

At the risk of providing a tautological definition, ordinal scales measure, well, order. So, our characteristics for ordinal scales are:

- **Order:** The order of the responses or observations matters.
- **Distance:** Ordinal scales do not hold distance. The distance between first and second is unknown as is the distance between first and third along with all observations.
- **True Zero:** There is no true or real zero. An item, observation, or category cannot finish zero.

**Appropriate statistics for ordinal scales:** count, frequencies, mode
**Displays:** histograms or bar charts

## Interval Scales

Interval scales provide insight into the variability of the observations or data.
Classic interval scales are Likert scales (e.g., 1 - strongly agree and 9 - strongly disagree) and
Semantic Differential scales (e.g., 1 - dark and 9 - light).
In an interval scale, users could respond to "I enjoy opening links to thwebsite from a company email" with a response ranging on a scale of values.

The characteristics of interval scales are:

- **Order:** The order of the responses or observations does matter.
- **Distance:** Interval scales do offer distance. That is, the distance from 1 to 2 appears the same as 4 to 5. Also, six is twice as much as three and two is half of four. Hence, we can perform arithmetic operations on the data.
- **True Zero:** There is no zero with interval scales. However, data can be rescaled in a manner that contains zero. An interval scales measure from 1 to 9 remains the same as 11 to 19 because we added 10 to all values. Similarly, a 1 to 9 interval scale is the same a -4 to 4 scale because we subtracted 5 from all values. Although the new scale contains zero, zero remains uninterruptable because it only appears in the scale from the transformation.

**Appropriate statistics for interval scales:** count, frequencies, mode, median, mean, standard deviation (and variance), skewness, and kurtosis.
**Displays:** histograms or bar charts, line charts, and scatter plots.

## Ratio Scales

Ratio scales appear as nominal scales with a true zero.
They have the following characteristics:

- **Order:** The order of the responses or observations matters.
- **Distance:** Ratio scales do do have an interpretable distance.

- **True Zero:** There is a true zero.

Income is a classic example of a ratio scale:

- Order is established. We would all prefer $100 to $1!
- Zero dollars means we have no income (or, in accounting terms, our revenue exactly equals our expenses!)
- Distance is interpretable, in that $20 appears as twice $10 and $50 is half of a $100.

For the web analyst, the statistics for ratio scales are the same as for interval scales.
**Appropriate statistics for ratio scales:** count, frequencies, mode, median, mean, standard deviation (and variance), skewness, and kurtosis.
**Displays:** histograms or bar charts, line charts, and scatter plots.

**4. Explain in detail about Storage Considerations in Big Data.**

In any environment intended to support the analysis of massive amounts of data, there must be the infrastructure supporting the data lifecycle from acquisition, preparation, integration, and execution. The need to acquire and manage massive amounts of data suggests a need for specialty storage systems to accommodate the big data applications. When evaluating specialty storage offerings, some variables to

consider include:

• Scalability, which looks at whether expectations for performance improvement are aligned with the additional of storage resources, and the degree to which the storage subsystem can support massive data volumes of increasing size.

• Extensibility, which examines how flexible the storage system's architecture is in allowing the system to be grown without the constraint of artificial limits.

• Accessibility, which looks at any limitations or constraints in providing simultaneous access to an expanding user community without compromising performance.

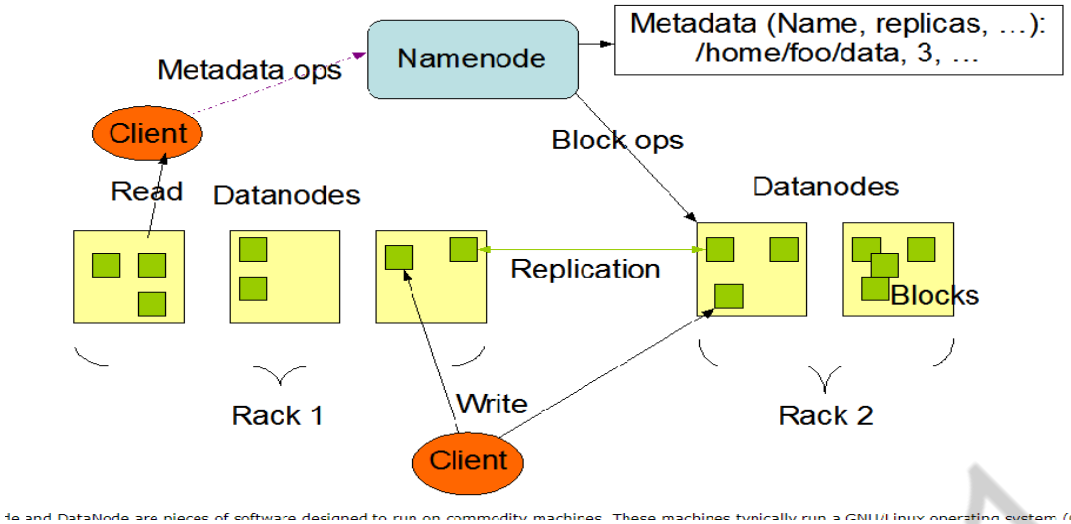
• Fault tolerance, which imbues the storage environment with the capability to recover from intermittent failures.

• High-speed I/O capacity, which measures whether the input/output channels can satisfy the demanding timing requirements for absorbing, storing, and sharing large data volumes.

• Integrability, which measures how well the storage environment can be integrated into the production environment.

Often, the storage framework involves a software layer for managing a collection of storage resources and providing much of these capabilities. The software configures storage for replication to provide a level of fault tolerance, as well as managing communications using standard protocols (such as UDP or TCP/IP) among the different processing nodes. In addition, some frameworks will replicate stored data, providing redundancy in the event of a fault or failure.

**5. Explain in detail about HDFS.**




HDFS is a distributed file system that handles large data sets running on commodity hardware. It is used to scale a single Apache Hadoop cluster to hundreds (and even thousands) of nodes. HDFS is one of the major components of Apache Hadoop, the others being MapReduce and YARN. HDFS should not be confused with or replaced by Apache HBase, which is a column-oriented non-relational database management system that sits on top of HDFS and can better support real-time data needs with its in-memory processing engine.

**Fast recovery from hardware failures**

Because one HDFS instance may consist of thousands of servers, failure of at least one server is inevitable. HDFS has been built to detect faults and automatically recover quickly.

**Access to streaming data**

HDFS is intended more for batch processing versus interactive use, so the emphasis in the design is for high data throughput rates, which accommodate streaming access to data sets.

**Accommodation of large data sets**

HDFS accommodates applications that have data sets typically gigabytes to terabytes in size. HDFS provides high aggregate data bandwidth and can scale to hundreds of nodes in a single cluster.

**Portability**

To facilitate adoption, HDFS is designed to be portable across multiple hardware platforms and to be compatible with a variety of underlying operating systems.

**6. Briefly discuss about MapReduce and YARN.**

YARN stands for “Yet Another Resource Negotiator “. It was introduced in Hadoop 2.0 to remove the bottleneck on Job Tracker which was present in Hadoop 1.0. YARN was described as a “Redesigned Resource Manager” at the time of its launching, but it has

now evolved to be known as large-scale distributed operating system used for Big Data processing.

YARN architecture basically separates resource management layer from the processing layer. In Hadoop 1.0 version, the responsibility of Job tracker is split between the resource manager and application manager

YARN also allows different data processing engines like graph processing, interactive processing, stream processing as well as batch processing to run and process data stored in HDFS (Hadoop Distributed File System) thus making the system much more efficient. Through its various components, it can dynamically allocate various resources and schedule the application processing. For large volume data processing, it is quite necessary to manage the available resources properly so that every application can leverage them.

**YARN Features**: YARN gained popularity because of the following features-

**Scalability:** The scheduler in Resource manager of YARN architecture allows Hadoop to extend and manage thousands of nodes and clusters.

**Compatibility:** YARN supports the existing map-reduce applications without disruptions thus making it compatible with Hadoop 1.0 as well.

**Cluster Utilization:** Since YARN supports Dynamic utilization of cluster in Hadoop, which enables optimized Cluster Utilization.

**Multi-tenancy:** It allows multiple engine access thus giving organizations a benefit of multi-tenancy.

The main components of YARN architecture include:

**Client**: It submits map-reduce jobs.

Resource Manager: It is the master daemon of YARN and is responsible for resource assignment and management among all the applications. Whenever it receives a processing request, it forwards it to the corresponding node manager and allocates resources for the completion of the request accordingly. It has two major components:

**Scheduler:** It performs scheduling based on the allocated application and available resources. It is a pure scheduler, means it does not perform other tasks such as monitoring or tracking and does not guarantee a restart if a task fails. The YARN scheduler supports plugins such as Capacity Scheduler and Fair Scheduler to partition the cluster resources.

**Application manager**: It is responsible for accepting the application and negotiating the first container from the resource manager. It also restarts the Application Manager container if a task fails.

**Node Manager**: It take care of individual node on Hadoop cluster and manages application and workflow and that particular node. Its primary job is to keep-up with the Node Manager. It monitors resource usage, performs log management and also kills a container based on directions from the resource manager. It is also responsible for creating the container process and start it on the request of Application master.

**Application Master**: An application is a single job submitted to a framework. The application manager is responsible for negotiating resources with the resource manager, tracking the status and monitoring progress of a single application. The application master requests the container from the node manager by sending a Container Launch Context (CLC) which includes everything an application needs to run. Once the application is started, it sends the health report to the resource manager from time-to-time.

**Container**: It is a collection of physical resources such as RAM, CPU cores and disk on a single node. The containers are invoked by Container Launch Context (CLC) which is a record that contains information such as environment variables, security tokens, dependencies etc.

**Application workflow in Hadoop YARN**:

Client submits an application

The Resource Manager allocates a container to start the Application Manager

The Application Manager registers itself with the Resource Manager

The Application Manager negotiates containers from the Resource Manager

The Application Manager notifies the Node Manager to launch containers

Application code is executed in the container

Client contacts Resource Manager/Application Manager to monitor application's status

Once the processing is complete, the Application Manager un-registers with the Resource Manager.

## UNIT II CLUSTERING AND CLASSIFICATION

## QUESTION BANK

## PART A

**1. What is classification?**

**Classification is:**
– the data mining process of
– finding a model (or function) that
– describes and distinguishes data classes or concepts,
– for the purpose of being able to use the model to predict the class of objects whose class label is unknown.
• That is, *predicts categorical class labels* (discrete or nominal).
• *Classifies the data (constructs a model) based on the training set.*
• It *predicts group membership for data instances.*

**2. List the categories of clustering methods.**
a) Partitioning methods
b) Hierarchical methods
c) Density based methods
d) Grid based methods
e) Model based methods

**3. Explain various steps in clustering process.**
• Find groups of similar data items
• Statistical techniques require some definition of "distance" (e.g. between travel profiles) while
conceptual techniques use background concepts and logical descriptionsUses:
• Demographic analysis Technologies:
• Self-Organizing Maps
• Probability Densities
• Conceptual Clustering

**4. What are the requirements of cluster analysis?**
The basic requirements of cluster analysis are
- Scalability
- Ability to deal with different types of attributes
- Ability to deal with noisy data
- Minimal requirements for domain knowledge to determine input parameters
- Constraint based clustering
- Interpretability and usability

**5. What is Cluster Analysis?**
• **Cluster analysis: A task that does**
– Grouping a set of data objects into clusters.
– Finding groups of objects such that the objects in a group will be similar (or related) to one another and different from (or unrelated to) the objects in other groups

• A cluster analysis is the process of analyzing the various clusters to organize the different
objects into meaningful and descriptive objects.

**6. What is a "decision tree"?**

It is a flow-chart like tree structure, where each internal node denotes a test on an attribute, each
branch represents an outcome of the test, and leaf nodes represent classes or class distributions.
Decision tree is a predictive model. Each branch of the tree is a classification question and
leaves of the tree are partition of the dataset with their classification.

**7. Define the concept of classification.**

Two step process
a) A model is built describing a predefined set of data classes or concepts. The model is constructed by analyzing database tuples described by attributes.
b) The model is used for classification.

**8. What are Bayesian Classifiers?**

• Bayesian Classifiers are statistical classifiers.
• They can predict class membership probabilities, such as the probability that a given sample belongs to a particular class.

**9. How will you solve a classification problem using Decision Tree?**

• Decision Tree Induction:
• Construct a decision tree using training data.
• For each ti € D apply the decision tree to determine its class ti-tuple D-Database

**10. Define k-means clustering.**

Given a collection of objects each with n measurable attributes, k-means is an analytical technique that, for a chosen value of k, identifies k clusters of objects based on the objects' proximity to the center of the k groups. The center is determined as the arithmetic average (mean) of each cluster's n-dimensional vector of attributes.

## PART B & C

### 1. Explain about k-means Clustering in detail.

Given a collection of objects each with n measurable attributes, k-means is an analytical technique that, for a chosen value of k, identifies k clusters of objects based on the objects' proximity to the centre of the k groups. The centre is determined as the arithmetic average (mean) of each cluster's n-dimensional vector of attributes. This section describes the algorithm to determine the k means as well as how best to apply this technique to several use cases. Figure 4.1 illustrates three clusters of objects with two attributes. Each object in the dataset is represented by a small dot color-coded to the closest large dot, the mean of the cluster.

**Usage**

**Medical**

Patient attributes such as age, height, weight, systolic and diastolic blood pressures, cholesterol level, and other attributes can identify naturally occurring clusters. These clusters could be used to target individuals for specific preventive measures or clinical trial participation. Clustering, in general, is useful in biology for the classification of plants and animals as well as in the field of human genetics.

**Customer Segmentation**

Marketing and sales groups use k-means to better identify customers who have similar behaviors and spending patterns. For example, a wireless provider may look at the following customer attributes: monthly bill, number of text messages, data volume consumed, minutes used during various daily periods, and years as a customer. The wireless company could then look at the naturally occurring clusters and consider tactics to increase sales or reduce the customer churn rate, the proportion of customers who end their relationship with a particular company.

**Image Processing**

Video is one example of the growing volumes of unstructured data being collected. Within each frame of a video, k-means analysis can be used to identify objects in the video. For each frame, the task is to determine which pixels are most similar to each other. The attributes of each pixel can include brightness, color, and location, the x and y coordinates in the frame. With security video images, for example, successive frames are examined to identify any changes to the clusters. These newly identified clusters may indicate unauthorized access to a facility.

Overview

To illustrate the method to find k clusters from a collection of M objects with n attributes, the two-dimensional case (n = 2) is examined. It is much easier to visualize the k-means method in two dimensions

Because each object in this example has two attributes, it is useful to consider each object corresponding to the point , where x and y denote the two attributes and i = 1, 2 … M. For a given cluster of m points (m M), the point that corresponds to the cluster's mean is called a centroid. In mathematics, a centroid refers to a point that corresponds to the center of mass for an object.

The k-means algorithm to find k clusters can be described in the following four steps.

1. Choose the value of k and the k initial guesses for the centroids.
2. Compute the distance from each data point($x_i$,$y_i$) to each centroid. Assign each point to the closest centroid. This association defines the first k clusters.
3. Compute the centroid, the center of mass, of each newly defined cluster from Step 2.
4. Repeat Steps 2 and 3 until the algorithm converges to an answer.
1. Assign each point to the closest centroid computed in Step 3.
2. Compute the centroid of newly defined clusters.
3. Repeat until the algorithm reaches the final answer.

**2. Explain about Classification of Decision trees in detail.**

A decision tree (also called prediction tree) uses a tree structure to specify sequences of decisions and consequences. Given input , the goal is to predict a response or output variable . Each member of the set is called an input variable. The prediction can be achieved by constructing a decision tree with test points and branches. At each test point, a decision is made to pick a specific branch and traverse down the tree.

Eventually, a final point is reached, and a prediction can be made. Each test point in a decision tree involves testing a particular input variable (or attribute), and each branch represents the decision being made. Due to its flexibility and easy visualization, decision trees are commonly deployed in data mining applications for classification purposes.

The input values of a decision tree can be categorical or continuous. A decision tree employs a structure of test points (called nodes) and branches, which represent the decision being made. A node without further branches is called a leaf node. The leaf nodes return class labels and, in some implementations, they return the probability scores. A decision tree can be converted into a set of decision rules. In the following example rule, income and mortgage_amount are input variables, and the response is the output variable default with a probability score.

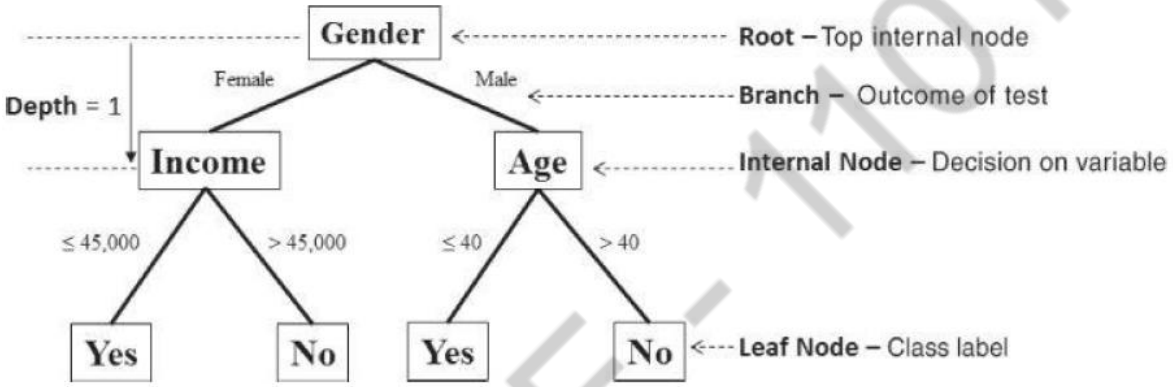
IF income < $50,000 AND mortgage_amount > $100K
THEN default = True WITH PROBABILITY 75%

Decision trees have two varieties: classification trees and regression trees. Classification trees usually apply to output variables that are categorical—often binary—in nature, such as yes or no, purchase or not purchase, and so on. Regression trees, on the other hand, can apply to output variables that are numeric or continuous,

such as the predicted price of a consumer good or the likelihood a subscription will be purchased.

Decision trees can be applied to a variety of situations. They can be easily represented in a visual way, and the corresponding decision rules are quite straightforward. Additionally, because the result is a series of logical if-then statements, there is no underlying assumption of a linear (or nonlinear) relationship between the input variables and the response variable.

The term branch refers to the outcome of a decision and is visualized as a line connecting two nodes. If a decision is numerical, the "greater than" branch is usually placed on the right, and the "less than" branch is placed on the left. Depending on the nature of the variable, one of the branches may need to include an "equal to" component.




Internal nodes are the decision or test points. Each internal node refers to an input variable or an attribute. The top internal node is called the root. The decision tree in Figure is a binary tree in that each internal node has no more than two branches. The branching of a node is referred to as a split.

Sometimes decision trees may have more than two branches stemming from a node. For example, if an input variable Weather is categorical and has three choices—Sunny, Rainy, and Snowy—the corresponding node Weather in the decision tree may have three branches labelled as Sunny, Rainy, and Snowy, respectively.

The depth of a node is the minimum number of steps required to reach the node from the root. In Figure for example, nodes Income and Age have a depth of one, and the four nodes on the bottom of the tree have a depth of two.

Leaf nodes are at the end of the last branches on the tree. They represent class labels—the outcome of all the prior decisions. The path from the root to a leaf node contains a series of decisions made at various internal nodes.

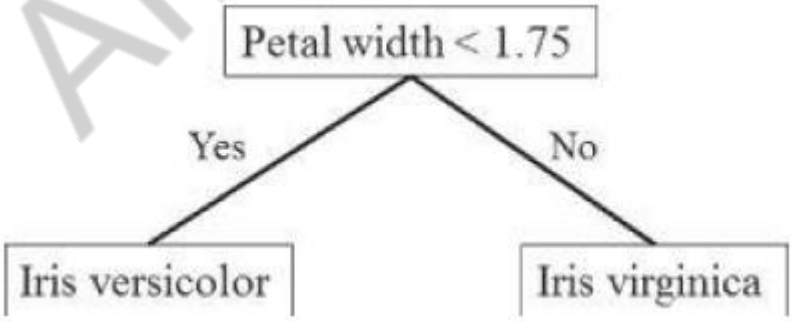
In Figure the root node splits into two branches with a Gender test. The right branch contains all those records with the variable Gender equal to Male, and the left branch contains all those records with the variable Gender equal to Female to create the depth 1 internal nodes. Each internal node effectively acts as the root of a subtree, and a best

test for each node is determined independently of the other internal nodes. The left-hand side (LHS) internal node splits on a question based on the Income variable to create leaf nodes at depth 2, whereas the right-hand side (RHS) splits on a question on the Age variable.

The decision tree shows that females with income less than or equal to $45,000 and males 40 years old or younger are classified as people who would purchase the product. In traversing this tree, age does not matter for females, and income does not matter for males.

Decision trees are widely used in practice. For example, to classify animals, questions (like cold-blooded or warm-blooded, mammal or not mammal) are answered to arrive at a certain classification. Another example is a checklist of symptoms during a doctor's evaluation of a patient. The artificial intelligence engine of a video game commonly uses decision trees to control the autonomous actions of a character in response to various scenarios. Retailers can use decision trees to segment customers or predict response rates to marketing and promotions. Financial institutions can use decision trees to help decide if a loan application should be approved or denied. In the case of loan approval, computers can use the logical if-then statements to predict whether the customer will default on the loan. For customers with a clear (strong) outcome, no human interaction is required; for observations that may not generate a clear response, a human is needed for the decision.

By limiting the number of splits, a short tree can be created. Short trees are often used as components (also called weak learners or base learners) in ensemble methods. Ensemble methods use multiple predictive models to vote, and decisions can be made based on the combination of the votes. Some popular ensemble methods include random forest [4], bagging, and boosting [5]. Section 7.4 discusses these ensemble methods more.
The simplest short tree is called a decision stump, which is a decision tree with the root immediately connected to the leaf nodes. A decision stump makes a prediction based on the value of just a single input variable.



**3. Explain in detail about Naïve Bayes Classification.**

Naïve Bayes is a probabilistic classification method based on Bayes' theorem (or Bayes' law) with a few tweaks. Bayes' theorem gives the relationship between the probabilities of two events and their conditional probabilities.

A naïve Bayes classifier assumes that the presence or absence of a particular feature of a class is unrelated to the presence or absence of other features. For example, an object can be classified based on its attributes such as shape, color, and weight. A reasonable classification for an object that is spherical, yellow, and less than 60 grams in weight may be a tennis ball. Even if these features depend on each other or upon the existence of the other features, a naïve Bayes classifier considers all these properties to contribute independently to the probability that the object is a tennis ball.

The input variables are generally categorical, but variations of the algorithm can accept continuous variables. There are also ways to convert continuous variables into categorical ones. This process is often referred to as the discretization of continuous variables. In the tennis ball example, a continuous variable such as weight can be grouped into intervals to be converted into a categorical variable. For an attribute such as income, the attribute can be converted into categorical values as shown below.

Low Income: income < $10,000
Working Class: $10,000 ≤ income < $50,000
Middle Class: $50,000 ≤ income < $1,000,000
Upper Class: income ≥ $1,000,000

The conditional probability of event C occurring, given that event A has already occurred, is denoted as , which can be found using the formula in Equation

Equation can be obtained with some minor algebra and substitution of the conditional probability:

where C is the class label and A is the observed attributes .
Equation is the most common form of the Bayes' theorem.

Mathematically, Bayes' theorem gives the relationship between the probabilities of C and A, and , and the conditional probabilities of C given A and A given C, namely and Bayes' theorem is significant because quite often is much more difficult to compute than and from the training data. By using Bayes' theorem, this problem can be circumvented.
An example better illustrates the use of Bayes' theorem. John flies frequently and likes to upgrade his seat to first class. He has determined that if he checks in for his flight at least two hours early, the probability that he will get an upgrade is 0.75; otherwise, the probability that he will get an upgrade is 0.35. With his busy schedule, he checks in at least two hours before his flight only 40% of the time. Suppose John did not receive an upgrade on his most recent attempt.

What is the probability that he did not arrive two hours early?
Let C = {John arrived at least two hours early}, and A = {John received an upgrade}, then ¬C = {John did not arrive two hours early}, and ¬A = {John did not receive an upgrade}.

John checked in at least two hours early only 40% of the time, or . Therefore,
The probability that John received an upgrade given that he checked in early is 0.75, or .
The probability that John received an upgrade given that he did not arrive two hours early is 0.35, or. Therefore, the probability that John received an upgrade can be computed as shown in Equation

Thus, the probability that John did not receive an upgrade. Using Bayes' theorem, the probability that John did not arrive two hours early given that he did not receive his upgrade is shown in Equation
Another example involves computing the probability that a patient carries a disease based on the result of a lab test. Assume that a patient named Mary took a lab test for a certain disease and the result came back positive. The test returns a positive result in 95% of the cases in which the disease is actually present, and it returns a positive result in 6% of the cases in which the disease is not present. Furthermore, 1% of the entire population has this disease. What is the probability that Mary actually has the disease, given that the test is positive?
Let C = {having the disease} and A = {testing positive}. The goal is to solve the probability of having the disease, given that Mary has a positive test result, . From the problem description, and. Bayes' theorem defines. The probability of testing positive, that is, needs to be computed first. That computation is shown in Equation
According to Bayes' theorem, the probability of having the disease, given that Mary has a positive test result, is shown in Equation.

That means that the probability of Mary actually having the disease given a positive test result is only 13.79%. This result indicates that the lab test may not be a good one. The likelihood of having the disease was 1% when the patient walked in the door and only 13.79% when the patient walked out, which would suggest further tests.

A more general form of Bayes' theorem assigns a classified label to an object with multiple attributes such that the label corresponds to the largest value of. he probability that a set of attribute values (composed of variables) should be labelled with a classification label equals the probability that the set of variables given is true, times the probability of divided by the probability of. Mathematically, this is shown in Equation.

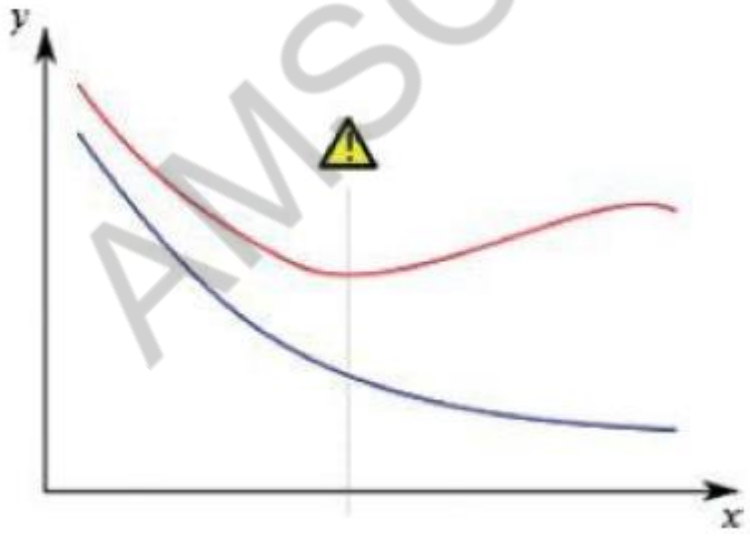
Consider the bank marketing example presented in Section 7.1 on predicting if a customer would subscribe to a term deposit. Let be a list of attributes {job, marital, education, default, housing, loan, contact, pout come}.

**4. How evaluation is performed on decision trees?**

Decision trees use greedy algorithms, in that they always choose the option that seems the best available at that moment. At each step, the algorithm selects which attribute to use for splitting the remaining records. This selection may not be the best overall, but it is guaranteed to be the best at that step. This characteristic reinforces the efficiency of decision trees. However, once a bad split is taken, it is propagated through the rest of the tree. To address this problem, an ensemble technique (such as random forest) may randomize the splitting or even randomize data and come up with a multiple tree structure. These trees then vote for each class, and the class with the most votes is chosen as the predicted class

There are a few ways to evaluate a decision tree. First, evaluate whether the splits of the tree make sense. Conduct sanity checks by validating the decision rules with domain experts, and determine if the decision rules are sound.

Next, look at the depth and nodes of the tree. Having too many layers and obtaining nodes with few members might be signs of overfitting. In overfitting, the model fits the training set well, but it performs poorly on the new samples in the testing set. Figure 7.7 illustrates the performance of an overfit model. The x-axis represents the amount of data, and the yaxis represents the errors. The blue curve is the training set, and the red curve is the testing set. The left side of the gray vertical line shows that the model predicts well on the testing set. But on the right side of the gray line, the model performs worse and worse on the testing set as more and more unseen data is introduced.



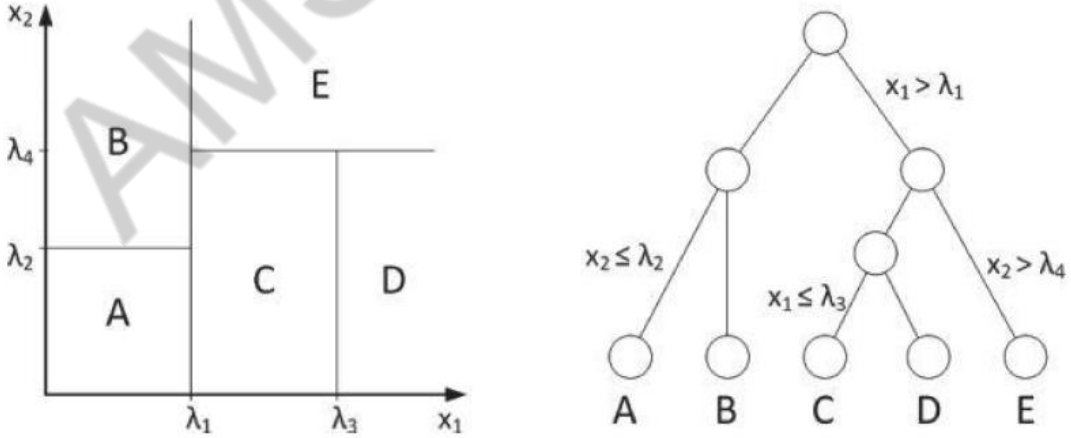

For decision tree learning, overfitting can be caused by either the lack of training data or the biased data in the training set. Two approaches [10] can help avoid overfitting in decision tree learning.

Stop growing the tree early before it reaches the point where all the training data is perfectly classified. Grow the full tree, and then post-prune the tree with methods such as reduced-error pruning and rule-based post pruning.
Last, many standard diagnostics tools that apply to classifiers can help evaluate overfitting. These tools are further discussed in Section 7.3.

Decision trees are computationally inexpensive, and it is easy to classify the data. The outputs are easy to interpret as a fixed sequence of simple tests. Many decision tree algorithms are able to show the importance of each input variable. Basic measures, such as information gain, are provided by most statistical software packages.

Decision trees are able to handle both numerical and categorical attributes and are robust with redundant or correlated variables. Decision trees can handle categorical attributes with many distinct values, such as country codes for telephone numbers. Decision trees can also handle variables that have a nonlinear effect on the outcome, so they work better than linear models (for example, linear regression and logistic regression) for highly nonlinear problems. Decision trees naturally handle variable interactions. Every node in the tree depends on the preceding nodes in the tree.

In a decision tree, the decision regions are rectangular surfaces. Figure 7.8 shows an example of five rectangular decision surfaces (A, B, C, D, and E) defined by four values— —of two attributes (and). The corresponding decision tree is on the right side of the figure. A decision surface corresponds to a leaf node of the tree, and it can be reached by traversing from the root of the tree following by a series of decisions according to the value of an attribute. The decision surface can only be axis-aligned for the decision tree.




The structure of a decision tree is sensitive to small variations in the training data. Although the dataset is the same, constructing two decision trees based on two different subsets may result in very different trees. If a tree is too deep, overfitting may occur, because each split reduces the training data for subsequent splits.

Decision trees are not a good choice if the dataset contains many irrelevant variables. This is different from the notion that they are robust with redundant variables and correlated variables. If the dataset contains redundant variables, the resulting decision tree ignores all but one of these variables because the algorithm cannot detect information gain by including more redundant variables. On the other hand, if the dataset contains irrelevant variables and if these variables are accidentally chosen as splits in the tree, the tree may grow too large and may end up with less data at every split, where overfitting is likely to occur. To address this problem, feature selection can be introduced in the data pre-processing phase to eliminate the irrelevant variables.

Although decision trees are able to handle correlated variables, decision trees are not well suited when most of the variables in the training set are correlated, since overfitting is likely to occur. To overcome the issue of instability and potential overfitting of deep trees, one can combine the decisions of several randomized shallow decision trees—the basic idea of another classifier called random forest [4]—or use ensemble methods to combine several weak learners for better classification. These methods have been shown to improve predictive power compared to a single decision tree.

For binary decisions, a decision tree works better if the training dataset consists of records with an even probability of each result. In other words, the root of the tree has a 50% chance of either classification. This occurs by randomly selecting training records from each possible classification in equal numbers. It counteracts the likelihood that a tree will stump out early by-passing purity tests because of bias in the training data.

When using methods such as logistic regression on a dataset with many variables, decision trees can help determine which variables are the most useful to select based on information gain. Then these variables can be selected for the logistic regression. Decision trees can also be used to prune redundant variables.

# CS8091 BIG DATA ANALYTICS

## UNIT III ASSOCIATION AND RECOMMENDATION SYSTEM

## QUESTION BANK

### PART-A

**1. What are Recommenders?**

• Recommenders are instances of personalization software.

• Personalization concerns adapting to the individual needs, interests, and preferences of each user.

Includes:

- Recommending
- Filtering
- Predicting (e.g. form or calendar appt. completion)

From a business perspective, it is viewed as part of Customer Relationship Management (CRM).

**2. What is Dimensionality Reduction?**

Dimension Reduction refers to:

- The process of converting a set of data having vast dimensions into data with lesser dimensions ensuring that it conveys similar information concisely.

– These techniques are typically used while solving machine learning problems to obtain better features for a classification or regression task.

**3. List out the problems on using Recommendation systems**

• Inconclusive user feedback forms

• Finding users to take the feedback surveys

• Weak Algorithms

• Poor results

• Poor Data

• Lack of Data

• Privacy Control (May NOT explicitly collaborate with recipients)

**4. List out the types of Recommender Systems.**

- Content

- Collaborative
- Knowledge

**5. What is Association Mining?**

Finding frequent patterns, associations, correlations, or causal structures among sets of items or objects in transaction databases, relational databases, and other information repositories.

**6. What is the Purpose of Apriori Algorithm?**

Apriori algorithm is an influential algorithm for mining frequent item sets for Boolean association rules. The name of the algorithm is based on the fact that the algorithm uses prior knowledge of frequent item set properties**.**

**7. List out the applications of Association rules.**

- Basket data analysis,
- cross-marketing,
- catalogue design,
- loss-leader analysis,
- clustering,
- classification

**8. Define support and confidence in Association rule mining.**

Support S is the percentage of transactions in D that contain AUB.

Confidence c is the percentage of transactions in D containing A that also contain B.

Support (A=>B) = P(AUB)

Confidence (A=>B) =P(B/A)

**9. What is Association rule?**

Association rule finds interesting association or correlation relationships among a large set of data items, which is used for decision-making processes. Association rules analyses buying patterns that are frequently associated or purchased together.

**10. Describe the method of generating frequent item sets without candidate generation.**

Frequent-pattern growth(or FP Growth) adopts divide-and-conquer strategy.

Steps:

- Compress the database representing frequent items into a frequent pattern tree or FP tree
- Divide the compressed database into a set of conditional databases
- Mine each conditional database separately

## PART B & C

### 1. Explain about the basics of Recommendation Systems

**A Common Challenge:**

- Assume you're a company selling items of some sort: movies, songs, products, etc.
- Company collects millions of ratings from users of their items
- To maximize profit / user happiness, you want to recommend items that users are likely to want

**Recommender systems**

- Systems for recommending items (e.g. books, movies, CD's, web pages, newsgroup messages) to users based on examples of their preferences.
- Many websites provide recommendations (e.g. Amazon, NetFlix, Pandora).
- Recommenders have been shown to substantially increase sales at on-line stores.
- Recommender systems are a technological proxy for a social process.
- Recommender systems are a way of suggesting like or similar items and ideas to a users specific way of thinking.
- Recommender systems try to automate aspects of a completely different information discovery model where people try to find other people with similar tastes and then ask them to suggest new things.

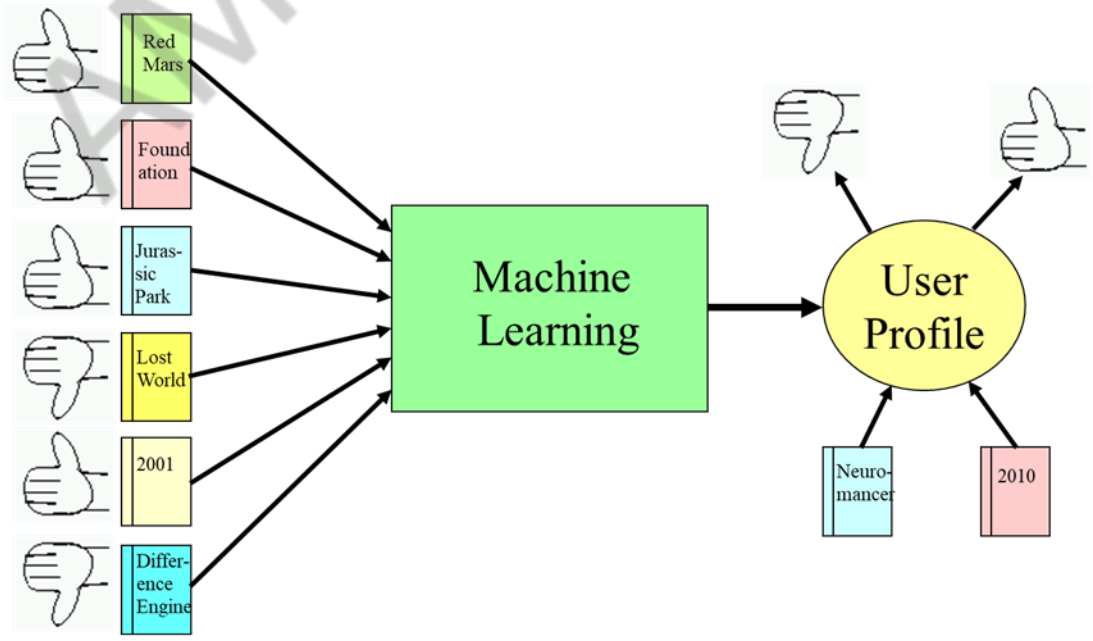
**Motivation for Recommender Systems**

Automates quotes like:

–"I like this book; you might be interested in it"

–"I saw this movie, you'll like it"

–"Don't go see that movie!"



**Usage**

- Massive E-commerce sites use this tool to suggest other items a consumer may want to purchase
- Web personalization

**Ways its used**

- Survey's filled out by past users for the use of new users
- Search-style Algorithms
- Genre matching
- Past purchase querying

**Problems on using Recommendation System**

- Inconclusive user feedback forms
- Finding users to take the feedback surveys
- Weak Algorithms
- Poor results
- Poor Data
- Lack of Data
- Privacy Control (May NOT explicitly collaborate with recipients)

**Maintenance**

- Costly
- Information becomes outdated
- Information quantity (large, disk space expansion)

**The Future of Recommender Systems**

- Extract implicit negative ratings through the analysis of returned item.
- How to integrate community with recommendations
- Recommender systems will be used in the future to predict demand for products, enabling earlier communication back the supply chain.

**2. Explain content-based filtering in detail.**

**CONTENT-BASED RECOMMENDATIONS**

• Main idea: Recommend items to customer x similar to previous items rated highly by x

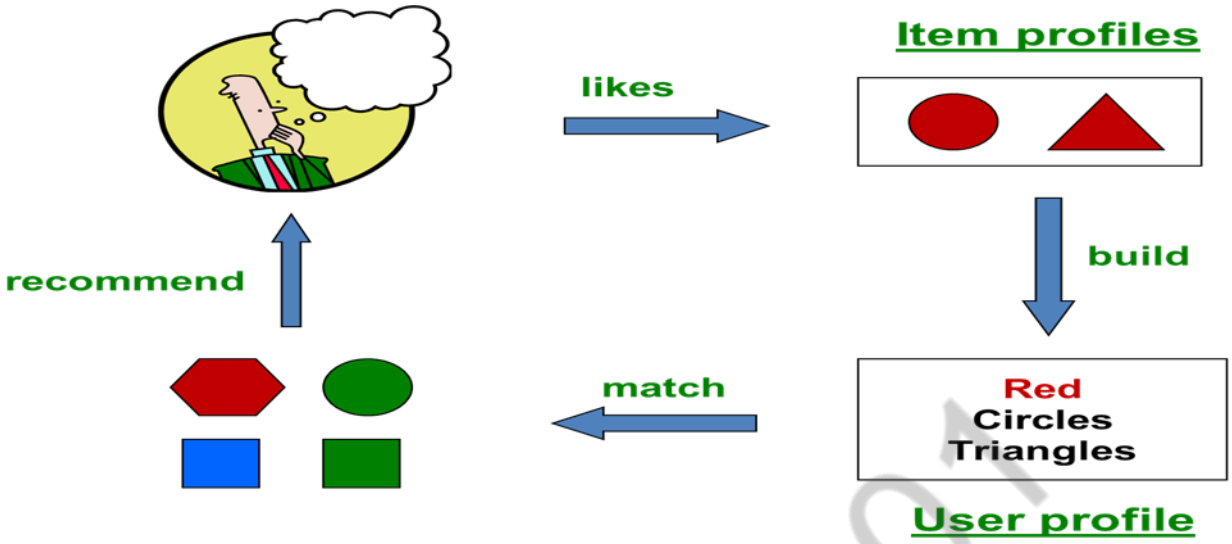
Example:

•Movie recommendations

–Recommend movies with same actor(s), director, genre, …

•Websites, blogs, news

–Recommend other sites with "similar" content




Item Profiles

- For each item, create an item profile
- Profile is a set (vector) of features
  - Movies: author, title, actor, director,…
  - Text: Set of "important" words in document
- How to pick important features?
  - Usual heuristic from text mining is TF-IDF

  (Term frequency * Inverse Doc Frequency)

  - Term … Feature
  - Document … Item
- fij = frequency of term (feature) i in doc (item) j
- ni = number of docs that mention term i
- N = total number of docs
- TF-IDF score: wij = TFij × IDFi
- Doc profile = set of words with highest TF-IDF scores, together with their scores

User Profiles and Prediction

Discovering Features of Documents

- There are many kinds of documents for which a recommendation system can be useful. For example, there are many news articles published each day, and we cannot read all of them.
- A recommendation system can suggest articles on topics a user is interested in, but
- how can we distinguish among topics?
- Web pages are also a collection of documents. Can we suggest pages a user might want to see?

- Likewise, blogs could be recommended to interested users, if we could classify blogs by topics.
- Unfortunately, these classes of documents do not tend to have readily available information giving features.
- A substitute that has been useful in practice is the identification of words that characterize the topic of a document.
- First, eliminate stop words – the several hundred most common words, which tend to say little about the topic of a document.
- For the remaining words, compute the TF.IDF score for each word in the document. The ones with the highest scores are the words that characterize the document.

**Content-Based Recommenders**

- Find me things that I liked in the past.
- Machine learns preferences through user feedback and builds a user profile
- Explicit feedback – user rates items
- Implicit feedback – system records user activity
  – Clicksteam data classified according to page category and activity, e.g. browsing a product page
  – Time spent on an activity such as browsing a page
- Recommendation is viewed as a search process, with the user profile acting as the query and the set of items acting as the documents to match.

**Recommending Items to Users Based on Content**

- With profile vectors for both users and items, we can estimate the degree to which a user would prefer an item by computing the cosine distance between the user's and item's vectors.
- As in Example 9.2, we may wish to scale var- ious components whose values are not boolean.
- The random-hyperplane and locality-sensitive-hashing techniques can be used to place (just) item profiles in buckets.
- In that way, given a user to whom we want to recommend some items, we can apply the same two techniques – random hyperplanes and LSH – to determine in which buckets we must look for items that might have a small cosine distance from the user.

**Pros: Content-based Approach**

- +: No need for data on other users
  – No cold-start or sparsity problems
- +: Able to recommend to users with unique tastes
- +: Able to recommend new & unpopular items
  – No first-rater problem
- +: Able to provide explanations
  – Can provide explanations of recommended items by listing content-features that caused an item to be recommended

**Cons: Content-based Approach**

- –: Finding the appropriate features is hard
- E.g., images, movies, music
- –: Recommendations for new users
- How to build a user profile?
- –: Overspecialization
- Never recommends items outside user's content profile
  - People might have multiple interests
  - Unable to exploit quality judgments of other users

**3. Explain about Collaborative Filtering in detail.**

Introduction u to Collaborative Filtering

- Collaborative filtering leverages product transactions to give recommendations.
- In this type of model, for a specific customer,
  - we find similar customers based on transaction history and recommend items that the customer in question hasn't purchased yet and which the similar customers tended to like.

**Collaborative Filtering**

- Match people with similar interests as a basis for recommendation.

1) Many people must participate to make it likely that a person with similar interests will be found.

2) There must be a simple way for people to express their interests.

3) There must be an efficient algorithm to match people with similar interests.

Designing Collaborative Filtering

- The standard approach to carry out collaborative filtering is non-negative matrix factorization (NMF).
- We first construct the sparse interaction matrix I which has customers in its rows and products in its columns.
  - We then try to decompose I to a product of two matrices C and P^T.
  - Here, C is the customer matrix, wherein each row is a customer and each column is the affinity of the customer to the latent feature discovered by the NMF.
  - Similarly, P is the product matrix, wherein each row is a product and each column is the extent the product embodies the latent feature.
  - These latent features are discovered by the NMF procedure.

- It is up to the modeler to decide how many latent features you want for the factorization.
- For example, suppose that we want 2 latent features in our movie recommendation engine.
- Latent feature discovered by NMF could be if whether the movie is a horror movie (that's latent feature one) and whether the movie is a romance movie (latent feature two).
- A row of C then represents the customer's affinity to horror and romance, while a row of P represents the extent the movie belongs to horror and romance.
- Similar to content-based systems, we can measure the similarity of these two vectors with cosine similarity to infer how much each customer likes each movie. Using that information, we can recommend movies to users.
- Consider user x
- Find set N of other users whose ratings are "similar" to x's ratings
- Estimate x's ratings based on ratings of users in N
- Users rate items – user interests recorded. Ratings may be:

– Explicit, e.g. buying or rating an item
– Implicit, e.g. browsing time, no. of mouse clicks

- Nearest neighbour matching used to find people with similar interests
- Items that neighbours rate highly but that you have not rated are recommended to you
- User can then rate recommended items

**Observations**

- Can construct a vector for each user (where 0 implies an item is unrated)

– E.g. for Alex: <1,0,5,4>
– E.g. for Peter <0,0,4,5>

- On average, user vectors are sparse, since users rate (or buy) only a few items.
- Vector similarity or correlation can be used to find nearest neighbour.

– E.g. Alex closest to Peter, then to George.

**Challenges for CF**

- Sparsity problem – when many of the items have not been rated by many people, it may be hard to find 'like minded' people.
- First rater problem – what happens if an item has not been rated by anyone.
- Privacy problems.
- Can combine CF with CB recommenders

– Use CB approach to score some unrated items.
– Then use CF for recommendations.

- Serendipity - recommend to me something I do not know already

– Oxford dictionary: the occurrence and development of events by chance in a happy or beneficial way.

**4. Explain in detail about Apriori Algorithm.**

- **Find all itemsets that have minimum support (*frequent itemsets*, also called large itemsets).**
- **Use frequent itemsets to generate rules.**

**E.g., a frequent itemset**
{Chicken, Clothes, Milk} [sup = 3/7]
and one rule from the frequent itemset
Clothes → Milk, Chicken [sup = 3/7, conf = 3/3]

**Step 1: Mining all frequent itemsets**
• A frequent *itemset* is an itemset whose support is ≥ minsup.
• Key idea: The apriori property (downward closure property): any subsets of a frequent itemset are also frequent itemsets
**The Algorithm**
• Iterative algo. (also called level-wise search): Find all 1-item frequent itemsets; then all 2-item frequent itemsets, and so on.
➢ In each iteration *k*, only consider itemsets that contain some *k*-1 frequent itemset.
• Find frequent itemsets of size 1: $F_1$
• From $k = 2$
➢ $C_k$ = candidates of size *k*: those itemsets of size *k* that could be frequent, given $F_{k-1}$
➢ $F_k$ = those itemsets that are actually frequent, $F_k \subseteq C_k$ (need to scan the database once).
• Single dimensional, single-level, Boolean frequent item sets
• Finding frequent item sets using candidate generation
• Generating association rules from frequent item sets
• Using the downward closure, we can prune unnecessary branches for further consideration

**APRIORI**
k = 1
- Find frequent set $L_k$ from $C_k$ of all candidate itemsets
- Form $C_{k+1}$ from $L_k$; k = k + 1
- Repeat 2-3 until $C_k$ is empty

• Details about steps 2 and 3
- Step 2: scan *D* and count each itemset in $C_k$ , if it's greater than minSup, it is frequent
- Step 3: next slide

**Apriori's Candidate Generation**
**• For k=1, $C_1$ = all 1-itemsets.**
**• For k>1, generate $C_k$ from $L_{k-1}$ as follows:**
– ***The join step***
**$C_k$ = k-2 way join of $L_{k-1}$ with itself**
**If both $\{a_1, \ldots, a_{k-2}, a_{k-1}\}$ & $\{a_1, \ldots, a_{k-2}, a_k\}$ are in $L_{k-1}$, then add $\{a_1, \ldots, a_{k-2}, a_{k-1}, a_k\}$ to $C_k$**

**(We keep items sorted).**
– ***The prune step***
**Remove $\{a_1, \ldots, a_{k-2}, a_{k-1}, a_k\}$ if it contains a non-frequent (k-1) subset**
**The Apriori principle:**
• Any subset of a frequent itemset must be frequent
• **Join Step**
$C_k$ is generated by joining $L_{k-1}$with itself
• **Prune Step**
Any (k-1)-itemset that is not frequent cannot be a subset of a frequent k-itemset

**How to Generate Candidates?**
Suppose the items in $L_{k-1}$ are listed in an order

**Step 1: self-joining $L_{k-1}$**
insert into $C_k$
select p.item$_1$, p.item$_2$, …, p.item$_{k-1}$, q.item$_{k-1}$
from $L_{k-1}$ p, $L_{k-1}$ q
where p.item$_1$=q.item$_1$, …, p.item$_{k-2}$=q.item$_{k-2}$, p.item$_{k-1}$ < q.item$_{k-1}$
**Step 2: pruning**
forall *itemsets c in* $C_k$ do

forall *(k-1)-subsets s of c* do
if *(s is not in* $L_{k-1}$*)* then delete *c* from $C_k$
**Pseudo-code:**
***$C_k$*: Candidate itemset of size k**
***$L_k$* : frequent itemset of size k**
***$L_1$* = {frequent items};**
**for (*k* = 1; *$L_k$* !=∅; *k*++) do begin**
***$C_{k+1}$* = candidates generated from *$L_k$*;**
**for each transaction *t* in database do**
**increment the count of all candidates in *$C_{k+1}$***
**that are contained in *t***
***$L_{k+1}$* = candidates in *$C_{k+1}$* with min_support**
**end**
**return $\cup_k$ *$L_k$*;**
Method:
– Let k=1
– Generate frequent itemsets of length 1
– Repeat until no new frequent itemsets are identified
- Generate length (k+1) candidate itemsets from length k frequent itemsets
- Prune candidate itemsets containing subsets of length k that are infrequent
- Count the support of each candidate by scanning the DB
- Eliminate candidates that are infrequent, leaving only those that are frequent

**5. Explain Association Rules in detail.**

Two-step approach:
1. Frequent Itemset Generation
– Generate all itemsets whose support ≥ minsup
2. Rule Generation

--Generate high confidence rules from each frequent itemset, where each rule is a binary partitioning of a frequent itemset
• Frequent itemset generation is still computationally expensive

**Power set**

• Given a set S, power set, P is the set of all subsets of S

• Known property of power sets

– If S has n number of elements, P will have N = 2n number of elements.

• Examples:

– For S = {}, P={{}}, N = 20 = 1

– For S = {Milk}, P={{}, {Milk}}, N=21=2

– For S = {Milk, Diaper}

– P={{},{Milk}, {Diaper}, {Milk, Diaper}}, N=22=4

– For S = {Milk, Diaper, Beer},

– P={{},{Milk}, {Diaper}, {Beer}, {Milk, Diaper}, {Diaper, Beer}, {Beer, Milk},

{Milk, Diaper, Beer}}, N=23=8

Example Association Rule

90% of transactions that purchase bread and butter also purchase milk

Antecedent: bread and butter

Consequent: milk

Confidence factor: 90%

Example Queries

• Find all the rules that have “Uludağ Gazozu” as consequent.

• Find all rules that have “Diet Coke” in the antecedent.

• Find all rules that have “sausage” in the antecedent and “mustard” in the consequent.

• Find all the rules relating items located on shelves A and B in the store.

• Find the “best” (most confident) k rules that have “Uludağ Gazozu” in the consequent.

Example:

I: itemset {cucumber, parsley, onion, tomato, salt, bread, olives, cheese, butter}

D: set of transactions

1 {{cucumber, parsley, onion, tomato, salt, bread},

2 {tomato, cucumber, parsley},

3 {tomato, cucumber, olives, onion, parsley},

4 {tomato, cucumber, onion, bread},

5 {tomato, salt, onion},

6 {bread, cheese}

7 {tomato, cheese, cucumber}

8 {bread, butter}}

Closed Item Set :

An itemset X is closed in a data set S if there exists no proper super-itemset5 Y such that Y has the same support count as X in S. An itemset X is a closed frequent itemset in set S if X is both closed and frequent in S.

Rule strength measures: Support and Confidence

 Find all the rules X & Y  Z with minimum confidence and support

– support, s, probability that a transaction contains {X  Y  Z}

– confidence, c, conditional probability that a transaction having {X  Y} also contains Z

 Support

 (absolute) support, or, support count of X: Frequency or occurrence of an itemset X

 (relative) support, s, is the fraction of transactions that contains X (i.e., the probability that a transaction contains X)

 An itemset X is frequent if X's support is no less than a minsup threshold

 Confidence (association rule: X Y )

 sup(X Y)/sup(x) (conditional prob.: Pr(Y|X) = Pr(X^Y)/Pr(X) )

 confidence, c, conditional probability that a transaction having X also contains Y

 Find all the rules X Y with minimum support and confidence

 sup(X Y) ≥ minsup

 sup(X Y)/sup(X) ≥ minconf

• Support: The rule holds with support sup in T (the transaction data set) if sup% of transactions contain X  Y.

 sup = Pr(X  Y)

• **Confidence:** The rule holds in *T* with confidence *conf* if *conf*% of tranactions that contain *X* also contain *Y*.

- *conf* = Pr(*Y* | *X*)

• An association rule is a pattern that states when *X* occurs, *Y* occurs with certain probability.
• *support* of *X* in *D* is *count*(*X*)/|*D*|
• For an association rule *X*⇒*Y*, we can calculate
– support (*X*⇒*Y*) = support (*XY*)
– confidence (*X*⇒*Y*) = support (*XY*)/support (*X*)
• Relate Support (S) and Confidence (C) to Joint and Conditional probabilities
• There could be exponentially many A-rules
• Interesting association rules are (for now) those whose S and C are greater than minSup and minConf (some thresholds set by data miners)

**Support count and Confidence**

• Support count: The support count of an itemset *X*, denoted by *X*.*count*, in a data set *T* is the number of transactions in *T* that contain *X*. Assume *T* has *n* transactions.
• Then,
• How is it different from other algorithms
– Classification (supervised learning -> classifiers)
– Clustering (unsupervised learning -> clusters)
• Major steps in association rule mining
– Frequent itemsets generation
– Rule derivation
• Use of support and confidence in association mining
– S for frequent itemsets
– C for rule derivation

# CS8091 – BIGDATA ANALYTICS
# UNIT – IV STREAM MEMORY
# QUESTION BANK

## PART – A

**1. What do you mean by data stream?**

In connection-oriented communication, a **data stream** is a sequence of digitally encoded coherent signals (packets of **data** or **data** packets) used to transmit or receive information that is in the process of being transmitted.

**2. Differentiate between DBMS and DSMS?**

| Database Systems (DBS) | Data Science Management System (DSMS) |
|---|---|
| • Persistent relations (relatively static, stored) | • Transient streams (on-line analysis) |
| • One-time queries | • Continuous queries (CQs) |
| • Random access | • Sequential access |
| • "Unbounded" disk store | • Bounded main memory |
| • Only current state matters | • Historical data is important |
| • No real-time services | • Real-time requirements |
| • Relatively low update rate | • Possibly multi-GB arrival rate |
| • Data at any granularity | • Data at fine granularity |
| • Assume precise data | • Data stale/imprecise |
| • Access plan determined by query processor, physical DB design | • Unpredictable/variable data arrival and characteristics |

**3. What is the definition of real time data?**

- **Real-time data** (RTD) is information that is delivered immediately after collection.
- There is no delay in the timeliness of the information provided.
- **Real-time data** is often used for navigation or tracking. Some uses of the term "**real-time data**" confuse it with the term dynamic **data**.

**4. What is Real Time Analytics Platform (RTAP)?**

- **Real Time Analytics Platform** (**RTAP**) analyzes data, correlates and predicts outcomes on a **real time** basis.
- The **platform** enables enterprises to track things in **real time** on a worldwide basis and helps in timely decision making.
- This **platform** provides us to build a range of powerful **analytic** applications.

**5. What is Sampling data in a stream?**

- **Sampling** from a finite **stream** is a special case of **sampling** from a station- ary window in which the window boundaries correspond to the first and last **stream** elements.
- The foregoing schemes fall into the category of equal-probability **sampling** because each window element is equally likely to be included in the **sample**.

**6. What are frequency moments?**

**Frequency Moments**

- Consider a stream S = {a1, a2, ..., am} with elements from a domain D = {v1, v2, ..., vn}.
- Let mi denote the frequency (also sometimes called multiplicity) of value vi ∈ D; i.e., the number of times vi appears in S.
- The k th frequency moment of the stream is defined as:

  Fk = Xn i=1 mk i (1)
- We will develop algorithms that can approximate Fk by making one pass of the stream and using a small amount of memory o(n + m).
- Frequency moments have a number of applications.
- F0 represents the number of distinct elements in the streams (which the FM-sketch from last class estimates using O(log n) space.
- F1 is the number of elements in the stream m.

**7. Write a short note on Decaying Window Algorithm.**

***Decaying Window Algorithm***

- The decaying window algorithm allows you to identify the most popular elements (trending, in other words) in an incoming data stream.
- The decaying window algorithm not only tracks the most recurring elements in an incoming data stream, but also discounts any random spikes or spam requests that might have boosted an element's frequency.
- In a decaying window, you assign a score or weight to every element of the incoming data stream. Further, you need to calculate the aggregate sum for each distinct element by adding all the weights assigned to that element. The element with the highest total score is listed as trending or the most popular.

**8. What does Real-Time Analytics Platform mean?**

- A real-time analytics platform enables organizations to make the most out of real-time data by helping them to extract the valuable information and trends from it.
- Such platforms help in measuring data from the business point of view in real time, further making the best use of data.

## PART – B

**1. Explain the Data streaming concept in detail. (13)**

## Overview of Data Stream

In more general terms, data streams can be characterized by the following:

- A data stream is potentially unbounded in size.
- The data is being generated continuously in real time.
- The data is generated sequentially as a stream. It is most often ordered by the timestamp assigned to each of the arriving records implicitly (by arrival time) or explicitly (by generation time).
- Typically, the volume of the data is very large, and the rates are high and irregular.

- AT the outset, the Data Stream is a
  - continuously arriving data flow
  - It's applications includes network traffic, credit card transaction flow, phone calling records, etc.
  - Data streams — continuous, ordered, changing, fast, huge amount
  - Traditional DBMS — data stored in finite, persistent data sets

**Data Stream**

- Large data volume ,likely structured, arriving a very high rate.
- DS is not only you see in YouTube.

## Definition

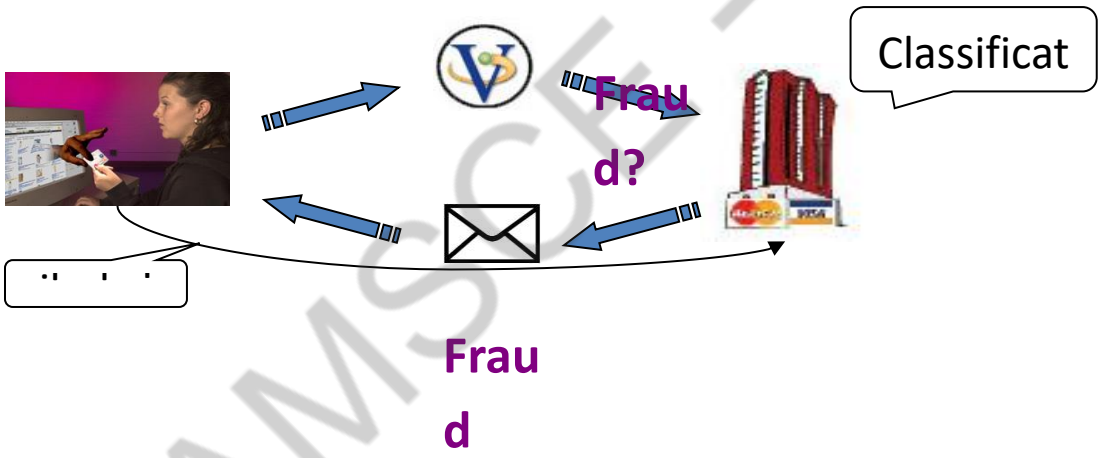
- *"A **data stream** is*
  - *a **real-time**, continuous, ordered (implicitly by arrival time or explicitly by timestamp) **sequence of items**.*
  - *It is impossible to control the order in which items arrive, nor is it feasible to locally **store** a stream in its entirety."*

The data stream is not necessarily Structured records and Unstructured such as audio or video data. Data streams are massive volumes of data, records arrive at a high rate

## Need of Data Streams

- Traditional DBMS -- data stored in finite, persistent data sets.
- New applications -- data as multiple, continuous, rapid, time-varying data streams.
- Scenarios where data streams are in actions?
  - Network monitoring and traffic engineering
  - Security applications
  - Telecom call records
  - Financial applications
  - Web logs and click-streams
  - Sensor networks
  - Manufacturing processes

A sample scenario of Fraud analysis context is depicted as follows:






**Scenarios where DSs are Challenged:**

a) Multiple, continuous, rapid, time-varying streams of data
b) Queries may be continuous (not just one-time)
   I. Evaluated continuously as stream data arrives
   II. Answer updated over time
c) Queries may be complex
   I. Beyond element-at-a-time processing
   II. Beyond stream-at-a-time processing

For example, storing one day's worth of IP network traffic flows in the AT&T IP backbone alone may require as much as 2.5TB.
In addition to the above characteristics of data streams, a large number of streaming applications require near real-time sophisticated analyses.

**Stream Classification**

- Stream Classification
  - Construct a classification model based on past records
  - Use the model to predict labels for new data
  - Help decision making

**Stream Management**

- In many data mining situations, we do not know the entire data set in advance
- **Stream Management** is important when the input rate is controlled **externally:**
  - Google queries
  - Twitter or Facebook status updates
  - We can think of the **data** as **infinite** and **non-stationary** (the distribution changes over time)

**Data Streaming**

- *Data from a stream could be interpreted differently by different use cases*.
- For example,
  - in the case of an e-commerce website, submitting an order can be a trigger to:
    - 1) a warehouse system to update inventory,
    - 2) a user recommendation system to provide suggestions to like-minded users, and
    - 3) a customer profiling system to provide additional recommendations of similar products

**Problems on Data Streams**

- **Types of queries one wants on answer on a data stream**
  a. **Sampling data from a stream**
     - Construct a random sample
  b. **Queries over sliding windows**
     - Number of items of type ***x*** in the last ***k*** elements of the stream

c. **Filtering a data stream**
   - Select elements with property ***x*** from the stream

d. **Counting distinct elements**
   - Number of distinct elements in the last ***k*** elements of the stream

e. **Estimating moments**
   - Estimate avg./std. dev. of last ***k*** elements

f. **Finding frequent elements**

## Data Stream's Characteristics

***The following are the list of Characteristics of DS:***

a) Huge volumes of continuous data, possibly infinite
b) Fast changing and requires fast, real-time response
c) Data stream captures nicely our data processing needs of today
d) Random access is expensive — single scan algorithm (*can only have one look*)!
e) Store only the summary of the data seen thus far.

- Most stream data are at pretty low-level or multi-dimensional in nature, needs multi-level and multi-dimensional processing .
- In addition to the above characteristics of data streams, a large number of streaming applications require near real-time sophisticated analyses.
- The data stream characteristics are clearly very different from those assumed for data stored in traditional DBMSs and thus make traditional DBMSs ill-suited for efficient implementation of many data stream processing applications.
- New data processing techniques are required to monitor and analyze massive volumes of data streams in real-time.

**2. Explain with a neat diagram about Stream data model and its Architecture. (13)**

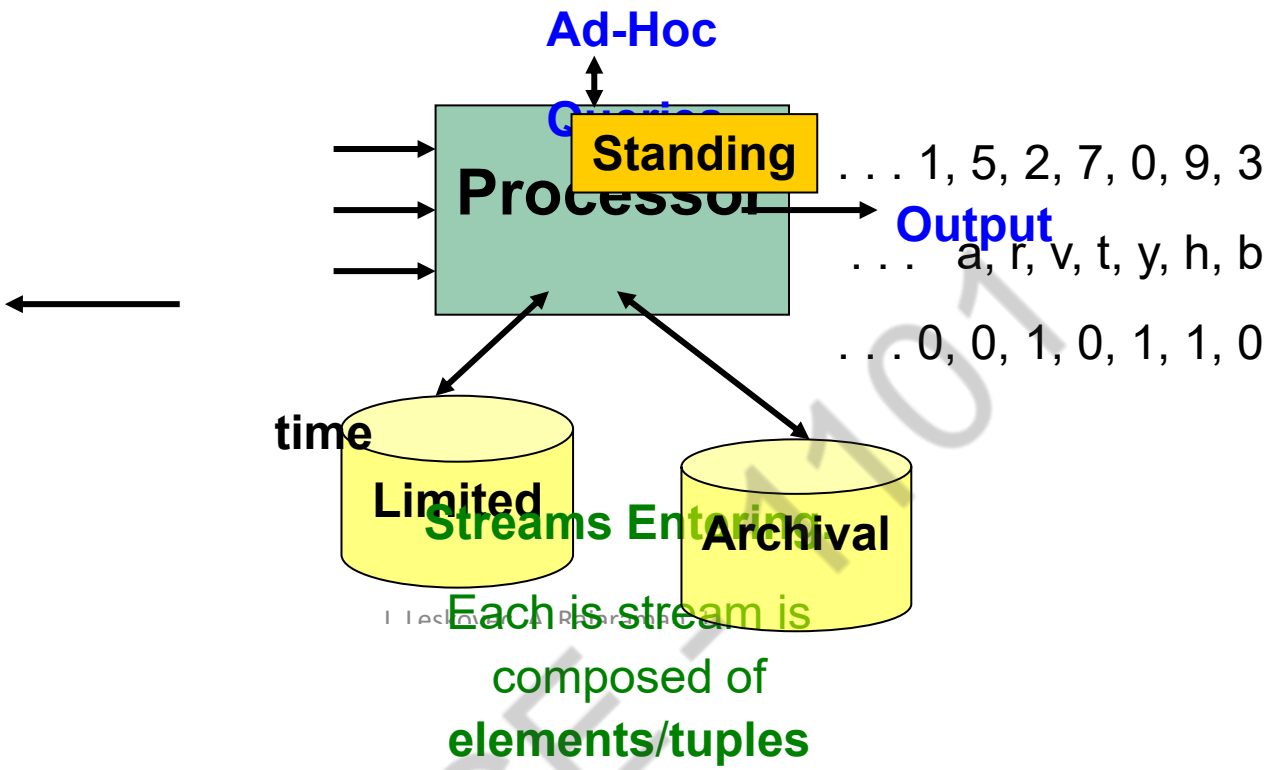
**Answer :**

**The Stream Model**

- **Input elements enters:**
  - at a rapid rate, at one or more input ports (i.e., **streams**)
  - **We call elements of the stream tuples**
    - ***The system cannot store the entire stream accessibly***

- **Q: How do you make critical calculations about the stream using a limited amount of (secondary) memory?**

**General Stream Processing Model**




**Computational Model**

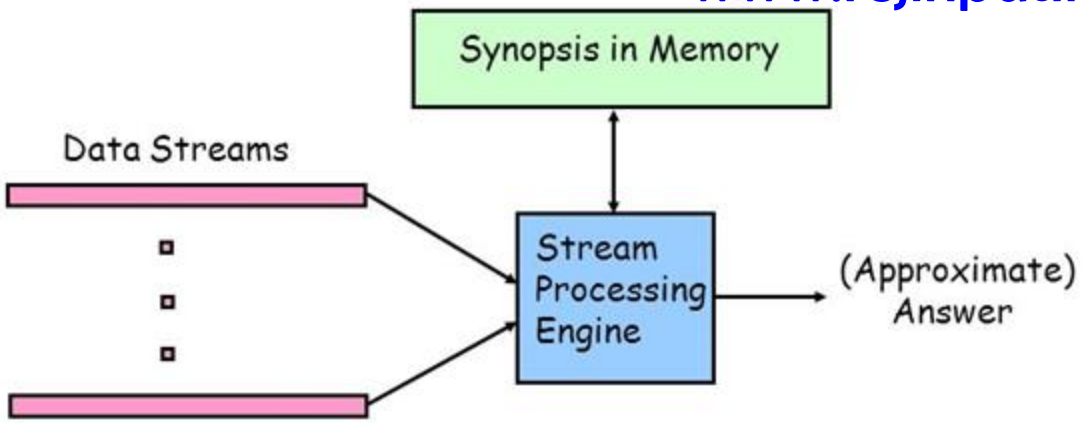
A data stream is a (massive) sequence of elements : e1, e2, … en.

Stream processing requirements

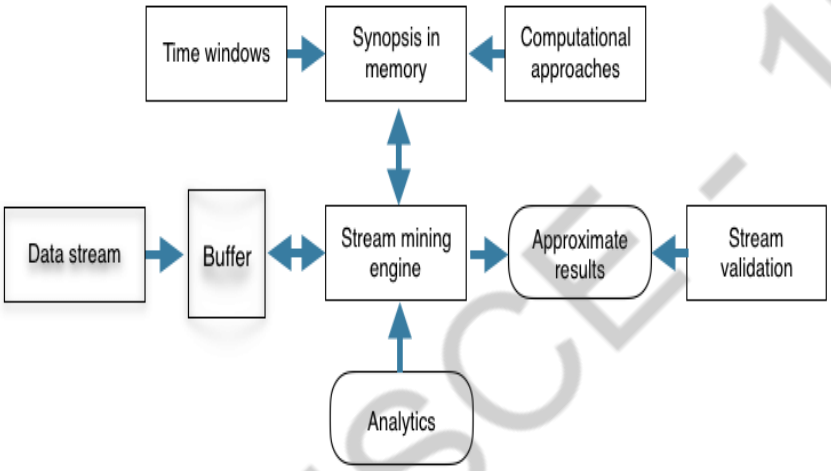
**Single pass** : Each record is examined at most once.

**Bounded storage** : Limited Memory (M) for storing synopsis.

**Real-time**: per record processing time (to maintain synopsis) must be low.




**A general model for data stream mining**






**Stream Processing Model**

- The stream can describe the underlying signal in various ways, resulting in a number of 11 different data stream models:
- • *Unordered cash register model - individual items of the stream are domain values that* rrive in no particular order and without any pre-processing.
- • *Ordered cash register model - individual items of the stream are not pre-processed* and arrive in the increasing (or decreasing) order of the domain values (e.g. in the order of the timestamp attribute values).
- • *Unordered aggregate model - individual items of the stream that belong to the same* domain arrive in a pre-processed (range values) format in no particular order.
- • *Ordered aggregate model - individual items of the stream arrive in a pre-processed* form and in the order of the domain values.

## Stream data Architecture

### What is Streaming Data Architecture?

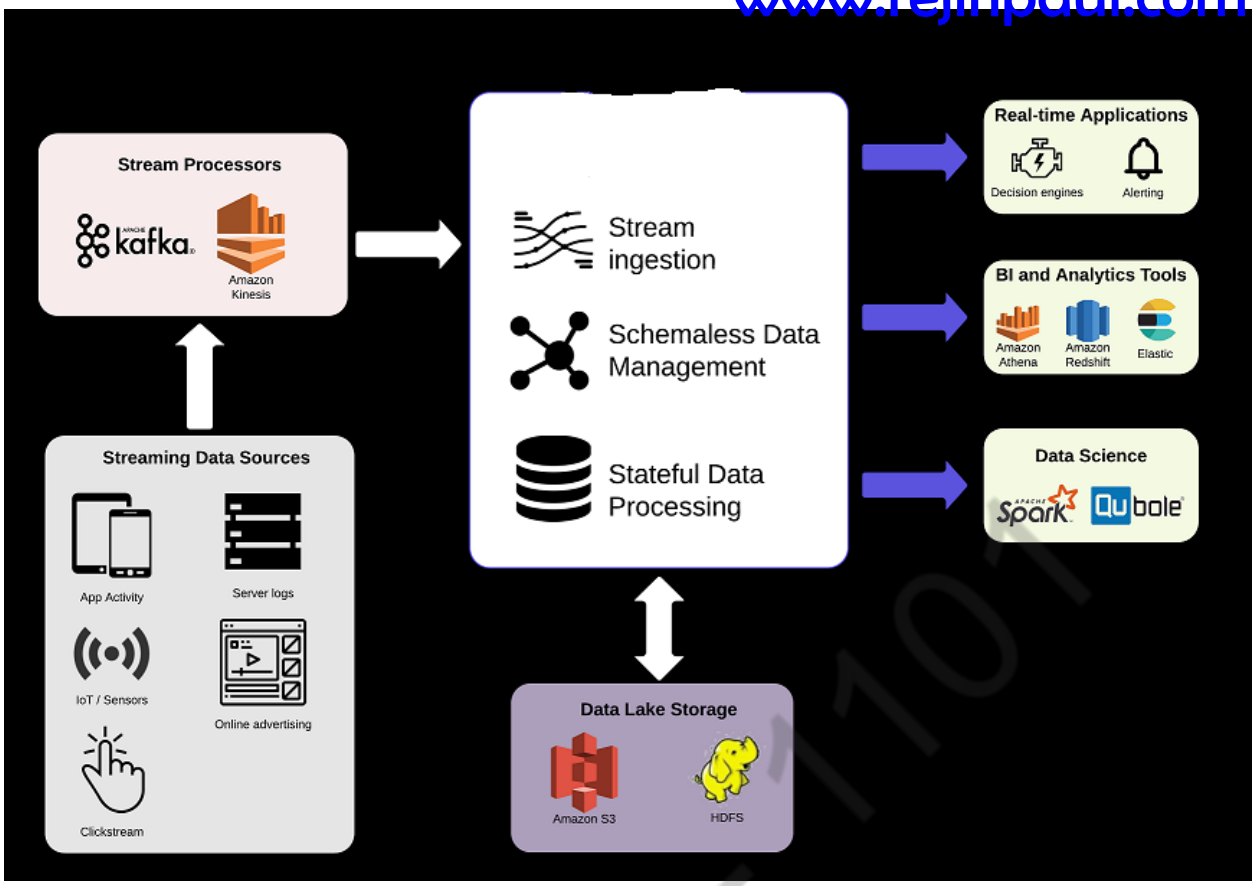
- A streaming data architecture can
  - ingest and process large volumes of streaming data from multiple sources.
- While traditional data solutions focused on
  - *writing and reading data in batches*
- A streaming data architecture:
  - ***consumes data immediately as it is generated***,
  - ***persists it to storage***, and
  - *may perform real-time processing, data manipulation and analytics*.

### Benefits of a modern streaming architecture

a) Can eliminate the need for large data engineering projects
b) Performance, high availability and fault tolerance built in
c) Newer platforms are cloud-based and can be deployed very quickly with no upfront investment
d) Flexibility and support for multiple use cases

### Modern Streaming Architecture

- In modern streaming data deployments, many organizations are adopting a full stack approach.
- Vendors are providing technology solutions, most of them based on Kafka,
  - which can take streaming data and perform the entire process,
    - from message ingestion through ETL,
    - storage management and
    - preparing data for analytics.




- **Stochastic Gradient Descent (SGD) is an example of a stream algorithm**
- **In Machine Learning we call this: Online Learning**
  - Allows for modeling problems where we have a continuous stream of data
  - We want an algorithm to learn from it and slowly adapt to the changes in data
- **Idea: Do slow updates to the model**
  - **SGD** (SVM, Perceptron) makes small updates
  - **So:** First train the classifier on training data.
  - **Then:** For every example from the stream, we slightly update the model (using small learning rate)

## Need of Stream data Model

- **Streaming data is**:
  - becoming a core component of enterprise data architecture.
- **Streaming technologies are not new**,
  - but they have considerably matured over the past year.

- ***The industry is moving from painstaking integration of technologies*** like Kafka and Storm,
  - towards **full stack solutions that provide an end-to-end streaming data architecture**.

**Data Stream Model**

- In the **data stream model**,
  - ***some or all of the input is represented as a finite sequence of integers*** (from some finite domain)
  - ***which is generally not available for random access, but instead arrives one at a time in a "stream".***

**3. Explain Filtering a stream in detail. (13)**

**Introduction to Filtering Streams**

- Another common process on streams is selection, or filtering. We want to accept those tuples in the stream that meet a criterion.
- Accepted tuples are passed to another process as a stream, while other tuples are dropped. If the selection criterion is a property of the tuple that can be calculated (e.g., the first component is less than 10), then the selection is easy to do.
- The problem becomes harder when the criterion involves lookup for membership in a set. It is especially hard, when that set is too large to store in main memory.

**A Motivating Example**

- Again let us start with a running example that illustrates the problem and what we can do about it.
- Suppose we have a set S of one billion allowed email addresses – those that we will allow through because we believe them not to be spam. The stream consists of pairs: an email address and the email itself.
- Since the typical email address is 20 bytes or more, it is not reasonable to store S in main memory.
- Thus, we can either use disk accesses to determine whether or not to let through any given stream element, or we can devise a method that requires no more main memory than we have available, and yet will filter most of the undesired stream elements.
- Suppose for argument's sake that we have one gigabyte of available main memory. In the technique known as Bloom filtering, we use that main memory as a bit array. In this case, we have room for eight billion bits, since one byte equals eight bits.
- Devise a hash function h from email addresses to eight billion buckets. Hash each member of S to a bit, and set that bit to 1. All other bits of the array remain 0.

- Since there are one billion members of S, approximately 1/8th of the bits will be 1.
- The exact fraction of bits set to 1 will be slightly less than 1/8th, because it is possible that two members of S hash to the same bit.

- When a stream element arrives, we hash its email address.
- If the bit to which that email address hashes is 1, then we let the email through. But if the email address hashes to a 0, we are certain that the address is not in S, so we can drop this stream element.
- Unfortunately, some spam email will get through. Approximately 1/8th of the stream elements whose email address is not in S will happen to hash to a bit whose value is 1 and will be let through.
- Nevertheless, since the majority of emails are spam (about 80% according to some reports), eliminating 7/8th of the spam is a significant benefit.
- Moreover, if we want to eliminate every spam, we need only check for membership in S those good and bad emails that get through the filter.
- Those checks will require the use of secondary memory to access S itself. There are also other options,
- As a simple example, we could use a cascade of filters, each of which would eliminate 7/8th of the remaining spam.

**The Bloom Filter**

A Bloom filter consists of:
1. An array of n bits, initially all 0's.
2. A collection of hash functions h1, h2, . . . , hk. Each hash function maps "key" values to n buckets, corresponding to the n bits of the bit-array.
3. A set S of m key values.

- The purpose of the Bloom filter is to allow through all stream elements whose keys are in S, while rejecting most of the stream elements whose keys are not in S.
- To initialize the bit array, begin with all bits 0. Take each key value in S and hash it using each of the k hash functions. Set to 1 each bit that is hi(K) for some hash function hi and some key value K in S.
- To test a key K that arrives in the stream, check that all of h1(K), h2(K), . . . , hk(K)
- are 1's in the bit-array. If all are 1's, then let the stream element through. If one or more of these bits are 0, then K could not be in S, so reject the stream element.

**Analysis of Bloom Filtering**

- If a key value is in S, then the element will surely pass through the Bloom filter. However, if the key value is not in S, it might still pass.
- We need to understand how to calculate the probability of a false positive, as a function of n, the bit-array length, m the number of members of S, and k, the number of hash functions.

- The model to use is throwing darts at targets. Suppose we have x targets and y darts. Any dart is equally likely to hit any target. After throwing the darts, how many targets can we expect to be hit at least once?
- The analysis is similar to the analysis and goes as follows:
    - The probability that a given dart will not hit a given target is (x − 1)/x.
    - The probability that none of the y darts will hit a given target is □x−1 x _y

.

We can w rite this expression as (1 – 1 x )x( y x ).

- Using the approximation (1−ǫ)1/ǫ = 1/e for small ǫ (recall Section 1.3.5), we conclude that the probability that none of the y darts hit a given target is e−y/x.

**FILTERING STREAMS**

Example:

- Consider the running example of Section 4.3.1. We can use the above calculation to get the true expected number of 1's in the bit array.
- Think of each bit as a target, and each member of S as a dart. Then the probability that a given bit will be 1 is the probability that the corresponding target will be hit by one or more darts. Since there are one billion members of S, we have y = 109 darts. As there are eight billion bits, there are x = 8 × 109
- targets. Thus, the probability that a given target is not hit is e−y/x = e−1/8 and the probability that it is hit is 1 − e−1/8. That quantity is about 0.1175.

- we suggested that 1/8 = 0.125 is a good approximation, which it is, but now we have the exact calculation. 2
- We can apply the rule to the more general situation, where set S has m members, the array has n bits, and there are k hash functions. The number of targets is x = n, and the number of darts is y = km. Thus, the probability that a bit remains 0 is e−km/n. We want the fraction of 0 bits to be fairly large, or else the probability that a nonmember of S will hash at least once to a 0 becomes too small, and there are too many false

positives. For example, we might choose k, the number of hash functions to be n/m or less.

- Then the probability of a 0 is at least e−1 or 37%. In general, the probability of a false
- positive is the probability of a 1 bit, which is 1 − e−km/n, raised to the kth power, i.e., (1 − e−km/n)k.

  Example : we found that the fraction of 1's in the array of our running example is 0.1175, and this fraction is also the probability of a false positive. That is, a nonmember of S will pass through the filter if it hashes to a 1, and the probability of it doing so is 0.1175.
- Suppose we used the same S and the same array, but used two different hash functions. This situation corresponds to throwing two billion darts at eight billion targets, and the probability that a bit remains 0 is e−1/4. In order to be a false positive, a nonmember of S must hash twice to bits that are 1, and this probability is (1 − e−1/4)2, or approximately 0.0493. Thus, adding a second hash function for our running example is an improvement, reducing the false-positive rate from 0.1175 to 0.0493. 2

**4. Explain the concept of Estimating Moments (13)**

**Introduction to Estimating Moments**

- The generalization of the problem of counting distinct elements in a stream is vital role on streaming Management.
- The problem, called computing "moments," involves the distribution of frequencies of different elements in the stream.
- The streaming Management includes how well we are managing moments of all orders and how best we concentrate on computing second moments, from which the general algorithm for all moments is a simple extension.

**Definition of Moments**

Suppose a stream consists of elements chosen from a universal set. Assume the universal set is ordered so we can speak of the ith element for any i.

Let mi be the number of occurrences of the ith element for any i.

Then the kth-order moment (or just kth moment) of the stream is the sum over all i of (mi) k .

***Example:***

- The 0th moment is the sum of 1 for each mi that is greater than 0.4
- That is, the 0th moment is a count of the number of distinct elements in the stream.

- We can use the method of Section 4.4 to estimate the 0th moment of a stream.
- Technically, since mi could be 0 for some elements in the universal set, we need to make explicit in the definition of "moment" that 00 is taken to be 0.
- For moments 1 and above, the contribution of mi's that are 0 is surely 0.
- The 1st moment is the sum of the mi 's, which must be the length of the stream.
- Thus, first moments are especially easy to compute; just count the length of the stream seen so far.
- The second moment is the sum of the squares of the mi 's.
- It is sometimes called the surprise number, since it measures how uneven the distribution of elements in the stream is.
- To see the distinction, suppose we have a stream of length 100, in which eleven different elements appear.
- The most even distribution of these eleven elements would have one appearing 10 times and the other ten appearing 9 times each.
- In this case, the surprise number is 102 + 10 × 9 2 = 910.
- At the other extreme, one of the eleven elements could appear 90 times and the other ten appear 1 time each.
- Then, the surprise number would be 902 + 10 × 1 2 = 8110.

**Frequency Moments**

- Consider a stream S = {a1, a2, ..., am} with elements from a domain D = {v1, v2, ..., vn}.
- Let mi denote the frequency (also sometimes called multiplicity) of value vi ∈ D; i.e., the number of times vi appears in S.
- The k th frequency moment of the stream is defined as:

$$Fk = Xn\ i=1\ mk\ i \quad (1)$$

- We will develop algorithms that can approximate Fk by making one pass of the stream and using a small amount of memory o(n + m).
- Frequency moments have a number of applications.
- F0 represents the number of distinct elements in the streams (which the FM-sketch from last class estimates using O(log n) space.
- F1 is the number of elements in the stream m.

There is no problem computing moments of any order if we can afford to keep in main memory a count for each element that appears in the stream.

However, also as in that section, if we cannot afford to use that much memory, then we need to estimate the kth moment by keeping a limited number of values in main memory and computing an estimate from these values.

For the case of distinct elements, each of these values were counts of the longest tail produced by a single hash function.

## The Alon-Matias-Szegedy Algorithm for Second Moments

- Let us assume that a stream has a particular length n.
- We shall show how to deal with growing streams in the next section. Suppose we do not have enough space to count all the mi 's for all the elements of the stream.
- We can still estimate the second moment of the stream using a limited amount of space; the more space we use, the more accurate the estimate will be.
- We compute some number of variables. For each variable X, we store: 1.
- A particular element of the universal set, which we refer to as X.element, and 2.
- An integer X.value, which is the value of the variable.
- To determine the value of a variable X, we choose a position in the stream between 1 and n, uniformly and at random. Set X.element to be the element found there, and initialize X.value to 1.
- As we read the stream, add 1 to X.value each time we encounter another occurrence of X.element.

Example :

Suppose the stream is a, b, c, b, d, a, c, d, a, b, d, c, a, a, b.

- The length of the stream is n = 15.
- Since a appears 5 times, b appears 4 times, and c and d appear three times each, the second moment for the stream is 5 2 + 42 + 32 + 32 = 59.
- Suppose we keep three variables, X1, X2, and X3 assume that at "random" we pick the 3rd, 8th, and 13th positions to define these three variables.
- When we reach position 3, we find element c, so we set X1.element = c and X1.value = 1.
- Position 4 holds b, so we do not change X1. Likewise, nothing happens at positions 5 or 6. At position 7, we see c again, so we set X1.value = 2.
- At position 8 we find d, and so set X2.element = d and X2.value = 1.
- Positions 9 and 10 hold a and b, so they do not affect X1 or X2.
- Position 11 holds d so we set X2.value = 2, and
- position 12 holds c so we set X1.value = 3.
- At position 13, we find element a, and so set X3.element = a and X3.value = 1.
- Then, at position 14 we see another a and so set X3.value = 2.
- Position 15, with element b does not affect any of the variables, so we are done, with final values X1.value = 3 and X2.value = X3.value = 2.

- ✷ We can derive an estimate of the second moment from any variable X. This estimate is n(2X.value − 1).

Example:

- Consider the three variables X1, X2 and X3.
- From X1 we derive the estimate n(2X1.value − 1) = 15 × (2 × 3 − 1) = 75. The other two variables, X2 and X3, each have value 2 at the end, so their estimates are 15 × (2 × 2 − 1) = 45.
- Recall that the true value of the second moment for this stream is 59.
- On the other hand, the average of the three estimates is 55, a fairly close approximation.

**Why the Alon-Matias-Szegedy Algorithm Works**

We can prove that the expected value of any variable constructed as in Section 4.5.2 is the second moment of the stream from which it is constructed.

Some notation will make the argument easier to follow.

Let e(i) be the stream element that appears at position i in the stream, and let c(i) be the number of times element e(i) appears in the stream among positions i, i + 1, . . . , n.

Example:

Consider the stream of Example 4.7. e(6) = a, since the 6th position holds a.

Also, c(6) = 4, since a appears at positions 9, 13, and 14, as well as at position 6.

Note that a also appears at position 1, but that fact does not contribute to c(6).

The expected value of n(2X.value − 1) is the average over all positions i between 1 and n of n(2c(i) − 1), that is E n(2X.value − 1) = 1 n Xn i=1 n(2c(i) − 1)

We can simplify the above by canceling factors 1/n and n, to get E n(2X.value − 1) = Xn i=1 2c(i).

However, to make sense of the formula, we need to change the order of summation by grouping all those positions that have the same element.

For instance, concentrate on some element a that appears ma times in the stream.

The term for the last position in which a appears must be 2 × 1 − 1 = 1.

The term for the next-to-last position in which a appears is 2 × 2 − 1 = 3.

The positions with a before that yield terms 5, 7, and so on, up to 2ma − 1, which is the term for the first position in which a appears.

That is, the formula for the expected value of 2X.value − 1 can be written: E n(2X.value − 1) = X a 1 + 3 + 5 + · · · + (2ma − 1)

Note that 1+ 3+ 5+· · ·+ (2ma−1) = (ma) 2 .

The proof is an easy induction on the number of terms in the sum. Thus, E n(2X.value − 1) = P a (ma) 2 , which is the definition of the second moment.

**5. Explain in detail on Counting ones in a Window. (13)**

- The Cost of Exact Counts To begin, suppose we want to be able to count exactly the number of 1's in the last k bits for any k ≤ N.
- Then we claim it is necessary to store all N bits of the window, as any representation that used fewer than N bits could not work.
- In proof, suppose we have a representation that uses fewer than N bits to represent the N bits in the window.
- Since there are 2N sequences of N bits, but fewer than 2N representations, there must be two different bit strings w and x that have the same representation.
- Since w 6= x, they must differ in at least one bit. Let the last k − 1 bits of w and x agree, but let them differ on the kth bit from the right end.
- Example 4.10 : If w = 0101 and x = 1010, then k = 1, since scanning from the right, they first disagree at position 1.
- If w = 1001 and x = 0101, then k = 3, because they first disagree at the third position from the right.

✶ Suppose the data representing the contents of the window is whatever sequence of bits represents both w and x.

- Ask the query "how many 1's are in the last k bits?" The query-answering algorithm will produce the same answer, whether the window contains w or x, because the algorithm can only see their representation.
- But the correct answers are surely different for these two bit-strings.
- Thus, we have proved that we must use at least N bits to answer queries about the last k bits for any k. In fact, we need N bits, even if the only query we can ask is "how many 1's are in the entire window of length N?"
- The argument is similar to that used above. Suppose we use fewer than N bits to represent the window, and therefore we can find w, x, and k as above.
- It might be that w and x have the same number of 1's, as they did in both cases of Example 4.10.

- However, if we follow the current window by any N − k bits, we will have a situation where the true window contents resulting from w and x are identical except for the leftmost bit, and therefore, their counts of 1's are unequal.
- However, since the representations of w and x are the same, the representation of the window must still be the same if we feed the same bit sequence to these representations. Thus, we can force the answer to the query "how many 1's in the window?" to be incorrect for one of the two possible window contents.

**The Datar-Gionis-Indyk-Motwani Algorithm**

- The simplest case of an algorithm called DGIM uses O(log2 N) bits to represent a window of N bits, and allows us to estimate the number of 1's in the window with an error of no more than 50%.
- An improvement of the method that limits the error to any fraction ǫ > 0, and still uses only O(log2 N) bits (although with a constant factor that grows as ǫ shrinks). To begin, each bit of the stream has a timestamp, the position in which it arrives. The first bit has timestamp 1, the second has timestamp 2, and so on.

## COUNTING ONES IN A WINDOW USING DGIM

Example 4.11 : Figure 4.2 shows a bit stream divided into buckets in a way that satisfies the DGIM rules.

At the right (most recent) end we see two buckets of size 1. To its left we see one bucket of size 2. Note that this bucket covers four positions, but only two of them are 1. Proceeding left, we see two buckets of size 4, and we suggest that a bucket of size 8exists further left.

Notice that it is OK for some 0's to lie between buckets. Also, observe from Fig. 4.2 that the buckets do not overlap; there are one or two of each size up to the largest size, and sizes only increase moving left. 2

**Storage Requirements for the DGIM Algorithm**

We observed that each bucket can be represented by O(logN) bits. If the window has length N, then there are no more than N 1's, surely. Suppose the largest bucket is of size 2j. Then j cannot exceed log2 N, or else there are more 1's in this bucket than there are 1's in the entire window. Thus, there are at most two buckets of all sizes from log2 N down to 1, and no buckets of larger
sizes.
We conclude that there are O(logN) buckets. Since each bucket can be represented in O(logN) bits, the total space required for all the buckets representing a window of size N is O(log2 N).

**Query Answering in the DGIM Algorithm**

Suppose we are asked how many 1's there are in the last k bits of the window, for some $1 \leq k \leq N$. Find the bucket b with the earliest timestamp that includes at least some of the k most recent bits. Estimate the number of 1's to be the sum of the sizes of all the buckets to the right (more recent) than bucket b, plus half the size of b itself.

***Example***

- Suppose the stream is that of Fig. 4.2, and k = 10.
- Then the query asks for the number of 1's in the ten rightmost bits, which happen to be 0110010110. Let the current timestamp (time of the rightmost bit) be t.
- Then the two buckets with one 1, having timestamps t − 1 and t − 2 are completely included in the answer.
- The bucket of size 2, with timestamp t − 4, is also completely included.
- However, the rightmost bucket of size 4, with timestamp t−8 is only partly included.
- We know it is the last bucket to contribute to the answer, because the next bucket to its left has timestamp less than t − 9 .

**********

**6. Explain the following:**

**i. Decaying Windows (6)**

**ii. RTAP Applications (7)**

**Decaying Windows**

- We have assumed that a sliding window held a certain tail of the stream, either the most recent N elements for fixed N, or all the elements that arrived after some time in the past.
- Sometimes we do not want to make a sharp distinction between recent elements and those in the distant past, but want to weight the recent elements more heavily.

**The Problem of Most-Common Elements**

- Suppose we have a stream whose elements are the movie tickets purchased all over the world, with the name of the movie as part of the element.
- We want to keep a summary of the stream that is the most popular movies "currently."
- While the notion of "currently" is imprecise, intuitively, we want to discount the popularity of a movie like Star Wars–Episode 4, which sold many tickets, but most of these were sold decades ago. On the other hand, a movie that sold 1.

**Definition of the Decaying Window**

- An alternative approach is to redefine the question so that we are not asking for a count of 1's in a window.
- Rather, let us compute a smooth aggregation of all the 1's ever seen in the stream, with decaying weights, so the further back in the stream, the less weight is given.

- Formally, let a stream currently consist of the elements a1, a2, . . . , at, where a1 is the first element to arrive and at is the current element. Let c be a small constant, such as 10−6 or 10−9.

***Decaying Window Algorithm***

- The decaying window algorithm allows you to identify the most popular elements (trending, in other words) in an incoming data stream.
- The decaying window algorithm not only tracks the most recurring elements in an incoming data stream, but also discounts any random spikes or spam requests that might have boosted an element's frequency.
- In a decaying window, you assign a score or weight to every element of the incoming data stream. Further, you need to calculate the aggregate sum for each distinct element by adding all the weights assigned to that element. The element with the highest total score is listed as trending or the most popular.

Define the exponentially decaying window for this stream to be the sum
t−1
X i=0
at−i(1 − c)i

- The effect of this definition is to spread out the weights of the stream elements as far back in time as the stream goes.
- In contrast, a fixed window with the same sum of the weights, 1/c, would put equal weight 1 on each of the most recent 1/c elements to arrive and weight 0 on all previous elements.

The distinction is suggested by Fig. 4.4. Window of length 1/c

Figure 4.4:

- A decaying window and a fixed-length window of equal weight It is much easier to adjust the sum in an exponentially decaying window than in a sliding window of fixed length. In the sliding window, we have to worry about the element that falls out of the window each time a new element arrives.
- That forces us to keep the exact elements along with the sum, or to use an approximation scheme such as DGIM.

However, when a new element at+1 arrives at the stream input, all we need to do is:
1. Multiply the current sum by 1 − c.
2. Add at+1.

- The reason this method works is that each of the previous elements has now moved one position further from the current element, so its weight is multiplied by 1 − c.
- Further, the weight on the current element is (1 − c)0 = 1, so adding at+1 is the correct way to include the new element's contribution.

**Finding the Most Popular Elements**

- Let us return to the problem of finding the most popular movies in a stream of ticket sales.6
- We shall use an exponentially decaying window with a constant c, which you might think of as 10−9.
- That is, we approximate a sliding window holding the last one billion ticket sales.
- For each movie, we imagine a separate stream with a 1 each time a ticket for that movie appears in the stream, and a 0 each time a ticket for some other movie arrives. The decaying sum of the 1's measures the current popularity of the movie.

- We imagine that the number of possible movies in the stream is huge, so we do not want to record values for the unpopular movies.
- Therefore, we establish a threshold, say 1/2, so that if the popularity score for a movie goes below this number, its score is dropped from the counting.
- For reasons that will become obvious, the threshold must be less than 1, although it can be any number less than 1.

- When a new ticket arrives on the stream, do the following:
- For each movie whose score we are currently maintaining, multiply its score by (1 − c).
- Suppose the new ticket is for movie M. If there is currently a score for M, add 1 to that score. If there is no score for M, create one and initialize it to 1.
- If any score is below the threshold 1/2, drop that score.
- It may not be obvious that the number of movies whose scores are maintained at any time is limited. However, note that the sum of all scores is 1/c.
- There cannot be more than 2/c movies with score of 1/2 or more, or else the sum of the scores would exceed 1/c.
- Thus, 2/c is a limit on the number of movies being counted at any time. Of course in practice, the ticket sales would be concentrated on only a small number of movies at any time, so the number of actively counted movies would be much less than 2/c.

- This example should be taken with a grain of salt, because, as we pointed out, there aren't enough different movies for this technique to be essential.

- Imagine, if you will, that the number of movies is extremely large, so counting ticket sales of each one separately is not feasible.

## Real-Time Analytics Platform Applications

- An ideal real-time analytics platform would help in analyzing the data, correlating it and predicting the outcomes on a real-time basis.
- The real-time analytics platform helps organizations in tracking things in real time, thus helping them in the decision-making process.
- The platforms connect the data sources for better analytics and visualization.

### *What does Real-Time Analytics Platform mean?*

- A real-time analytics platform enables organizations to make the most out of real-time data by helping them to extract the valuable information and trends from it.
- Such platforms help in measuring data from the business point of view in real time, further making the best use of data.

***Examples of real-time analytics include:***

- **Real time** credit scoring, helping financial institutions **to** decide immediately whether **to** extend credit.
- Customer relationship management (CRM), maximizing satisfaction and business results during each interaction with the customer.
- Fraud detection at points of sale.

### Why RTAP?

- Decision systems that go beyond visual analytics have an intrinsic need to analyze data and respond to situations.
- Depending on the sophistication, such systems may have to act rapidly on incoming information, grapple with heterogeneous knowledge bases, work across multiple domains, and often in a distributed manner.
- Big Data platforms offer programming and software infrastructure to help perform
- analytics to support the performance and scalability needs of such decision support systems for IoT domains.
- There has been significant focus on Big Data analytics platforms on the volume dimension of Big Data.
- In such platforms, such as MapReduce, data is staged and aggregated over time, and analytics are performed in a batch mode on these large data corpus.
- These platforms weakly scale with the size of the input data, as more distributed compute resources are made available.

- However, as we have motivated before, IoT applications place an emphasis on online analytics, where data that arrives rapidly needs to be processed and analyzed with low latency to drive autonomic decision making.

## Streaming Analytics Platforms For All Real-time Applications

The top platforms being used all over the world for Streaming analytics solutions:

### *Apache Flink*

- Flink is an open-source platform that handles distributed stream and batch data processing.
- At its core is a streaming data engine that provides for data distribution, fault tolerance, and communication, for undertaking distributed computations over the data streams.
- In the last year, the Apache Flink community saw three major version releases for the platform and the community event Flink Forward in San Francisco.
- Apache Flink contains several APIs to enable creating applications that use the Flink engine. Some of the most popular APIs on the platform are-
- DataStream API for unbounded streams, DataSet API for static data embedded in Python, Java, and Scala, and the Table API with a SQL-like language.

### *Spark Streaming*

- Apache Spark is used to build scalable and fault-tolerant streaming applications.
- With Spark Streaming, you get to use Apache Spark's language-integrated API which lets you write streaming jobs in the similar way as you write batch jobs.
- Spark Streaming supports the three languages- Java, Scala, Python. Apache Spark is being used in various leading industries today, such as- Healthcare, Finance, e-commerce, Media and Entertainment, Travel industry, etc.
- The popularity of Apache Spark adds the glitter to the platform Spark Streaming.

### *IBM Streams*

- This streaming analytics platform from IBM enables the applications developed by users to gather, analyze, and correlate information that comes to them from a variety of sources.
- The solution is known to handle high throughput rates and up to millions of events and messages per second, making it a leading proprietary streaming analytics solution for real-time applications.
- IBM Stream computing helps analyze large streams of data in the form of unstructured texts, audio, video, and geospatial, and allows for organizations to spot risks and opportunities and make efficient decisions.

**Software AG's Apama Streaming Analytics**

- Apama Streaming analytics platform is built for streaming analytics and automated action on fast-moving data on the basis of intelligent decisions.
- The software bundles up other aspects like messaging, event processing, in-memory data management and visualization and is ideal for fast-moving Big Data Analytics Solutions. Sensors that bring in loads of data from different sources can be churned using this solution in real-time.
- With Apama, you can act on high-volume business operations in real-time.

*Azure Stream Analytics*

- Azure Stream Analytics facilitates the development and deployment of low-cost solutions that can gain real-time insights from devices, applications, and sensors.
- It is recommended to be used for IoT scenarios like real-time remote management and monitoring, connected cars, etc.
- It allows to easily develop and run parallel real-time time analytics on IoT and other kinds of Big Data using a simple language that resembles SQL.
- These streaming applications and platforms are helping organizations drive their streaming analytics goals and IoT solutions with ease.
- Big Data is a source of knowledge today and organizations are increasingly trying to leverage its potential to drive their decisions and major changes.

Thus the real-time Analytics Platform is useful for Real time application development and its successful implementations.



# PART – C

## 1. CASE STUDY - STOCK MARKET PREDICTION

**You are required to make a case study on STOCK MARKET PREDICTION with following requirements:**

**i) Briefly introduce about Stock market and its prediction**
**ii) The Solution Path of the stock Market Perdition.**
**iii) Do the Empirical Study of the Stock Market Prediction.**

**CASE STUDY - STOCK MARKET PREDICTION**

**Introduction**

- Trading is the process of buying and selling of financial instruments Stock market for the trading.
- One of the most important sources for companies to raise money allows businesses to go public, or raise additional capital for expansion

- Predicting stock performance is a very large and profitable area of study
- Many companies have developed stock predictors based on neural networks
- This technique has proven successful in aiding the decisions of investors
- Can give an edge to beginning investors who don't have a lifetime of experience

**What is stock market and how it works?**

- The stock market works like an auction where investors who buy and sell shares of stocks.
- These are a small piece of ownership of a public corporation.
- Stock prices usually reflect investors' opinions of what the company's earnings will be.
- Stock trading involves buying and selling stocks frequently in an attempt to time the market.
- The goal of stock traders is to capitalize on short-term market events to sellstocks for a profit, or buy stocks at a low.
- Investors who trade stocks do extensive research, often devoting hours a day to following the market.

**Stock Market Prediction**

- Collect a sufficient amount of historical stock data
- Using the data train a neural network
- Once trained, the neural network can be used to predict stock behavior
- Forecasting stock market prices has always been challenging task for many business analyst and researchers.
- In fact, stock market price prediction is an interesting area of research for investors.
- For successful investment lot many investors are interested in knowing about future situation of market.
- Effective prediction systems indirectly help traders by providing supportive information such as the future market direction.

**What is Stock Market Prediction?**

Stock market prediction is the act of trying to determine the future value of company stock or other financial instrument traded on an exchange.

The successful prediction of a stock's future price could yield significant profit.

Generally modeled as classification or regression task predict a binary or ordinal label

Features:

Negation is important

Using all words (in naïve bayes) works well for some tasks
Finding subsets of words may help in other tasks
Hand-built polarity lexicons
Use seeds and semi-supervised learning to induce lexicons

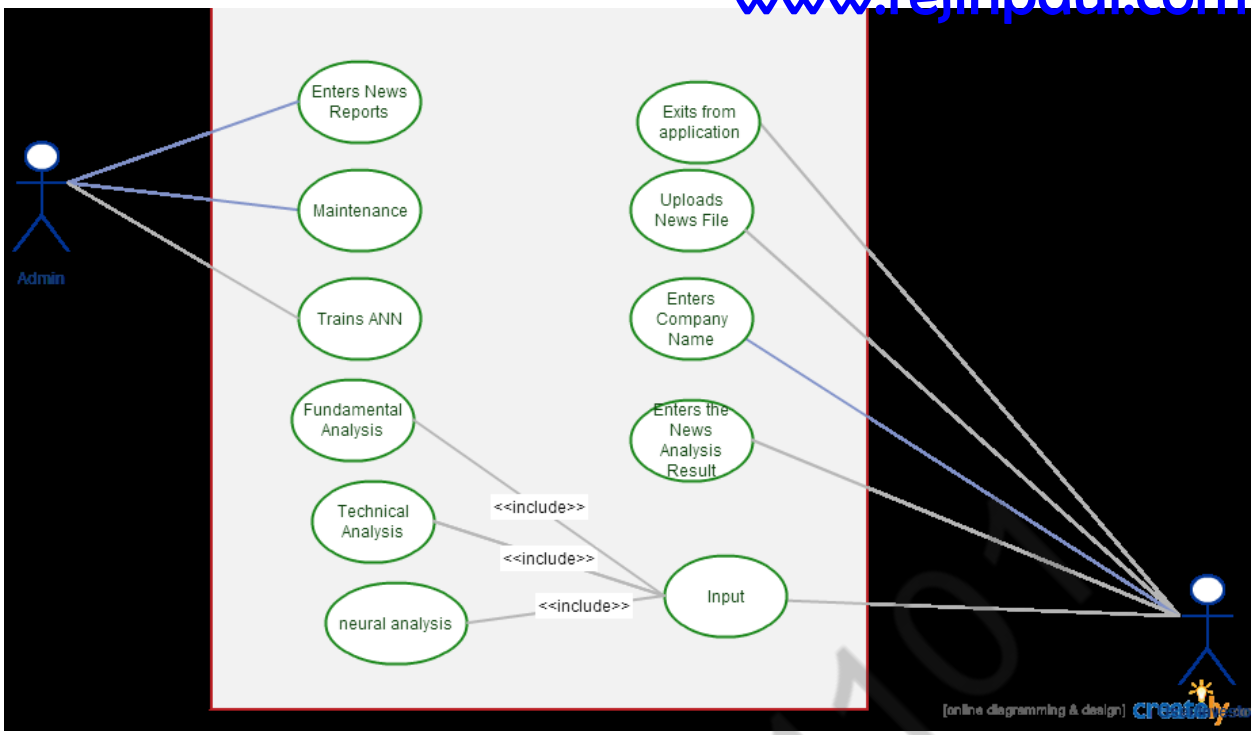
The most important for predicting stock market prices are neural networks because they are able to learn nonlinear-mappings between inputs and outputs.
It may be possible to perform better than traditional analysis and other computer-based methods with the neural-networks ability to learn-nonlinear, chaotic-systems.

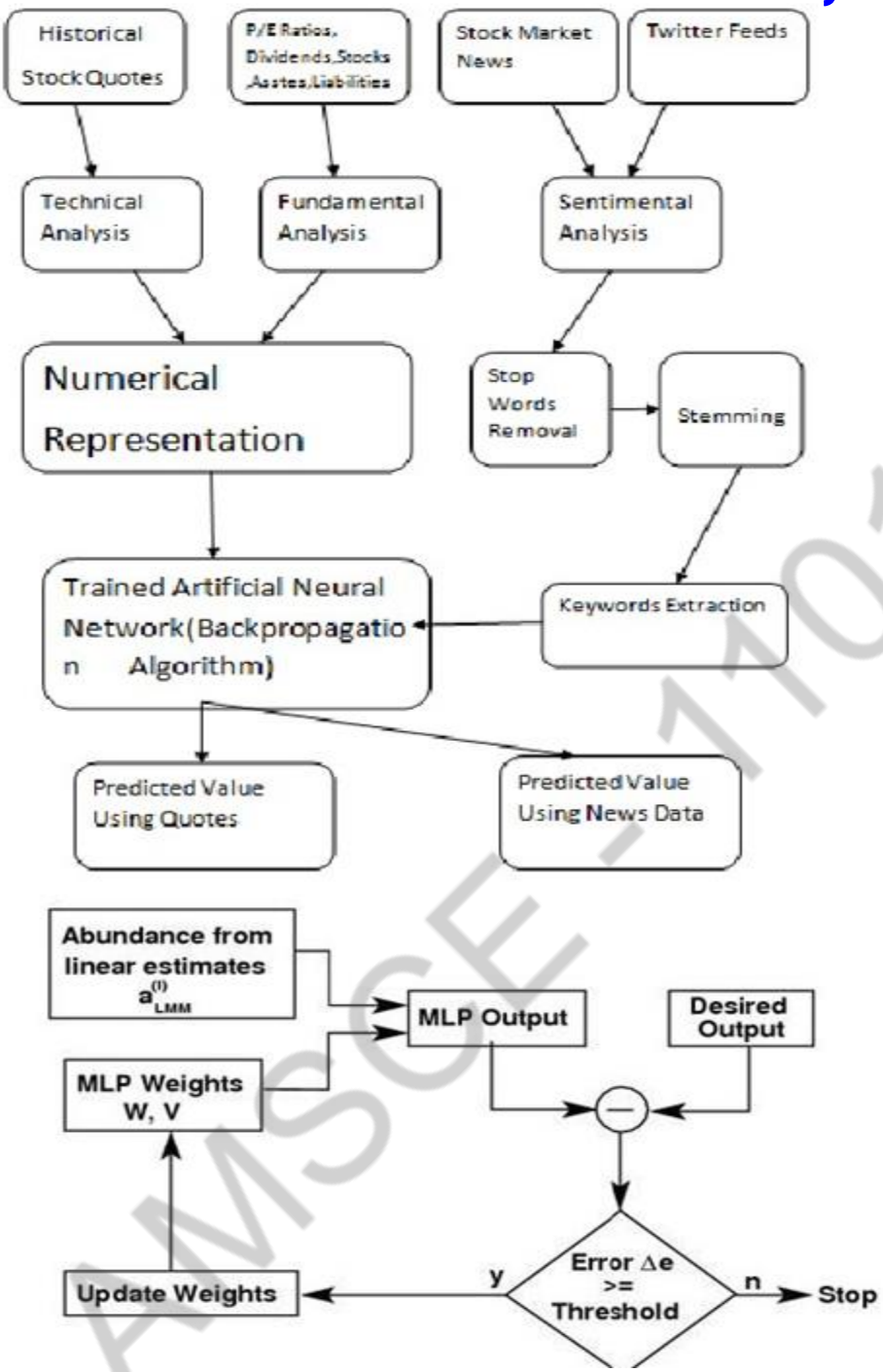
## Overview of Stock Information and Stock Market

- Basically the main objective of this project is to collect the stock information for some previous years and then accordingly predict the results for the predicting what would happen next.
- Use of two well-known techniques neural network and data mining for stock market prediction has a sense.
- Extract useful information from a huge amount of data set and data mining is also able to predict future trends and behaviors through neural network.
- Therefore, combining both these techniques could make the prediction more suitable and much more reliable and depends on which way the forecast is used for.
- So the procedures that we will be using have proven to be very applicable to the task of forecasting product demand in a logistics system.
- Many techniques, which can prove useful for forecasting-problems, have shown to be inadequate to the task of demand forecasting in logistics systems.

## The Solution Path

The following picture depicts the stock market prediction processes and its users assessment :


Admin
Enters News Reports
Maintenance
Trains ANN
Fundamental Analysis
Technical Analysis
neural analysis
<<include>>
<<include>>
<<include>>
Input
Exits from application
Uploads News File
Enters Company Name
Enters the News Analysis Result
[online diagramming & design]






**Time Series Analysis Vs NN**

TS - Instructions and rules are central

TS – A Mathematical formula define the dynamics

NN - Don't perform according to preset rules. Learns from regularities and sets its own rules

NN – Not described explicitly in mathematical terms

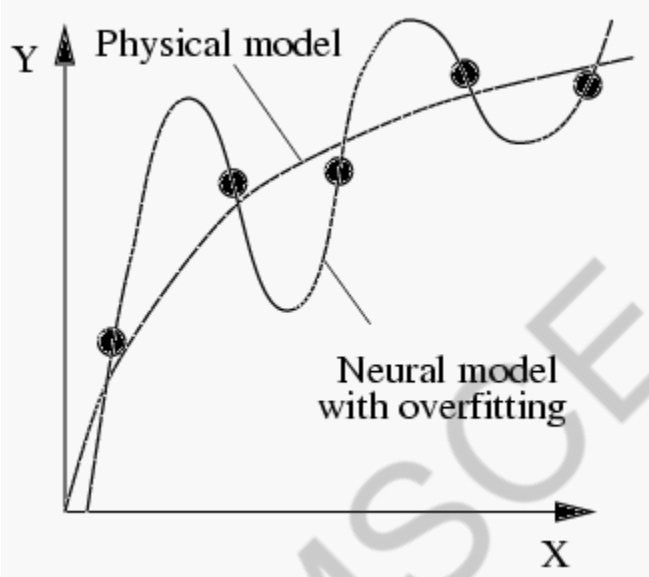
**Benifits with NN**

- Generalization ability and robustness

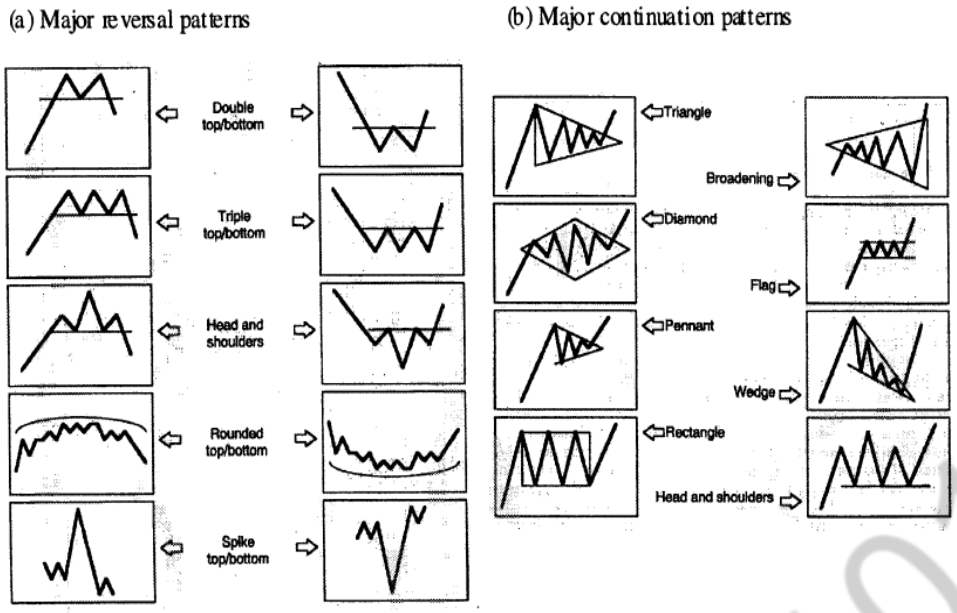
- Mapping of input/output
- No assumptions of model has to be made
- Flexilbilty

**Drawbacks with NN**

- Black-box property
- Overfitting
- Expertise for choice of input
- Training takes a lot of time



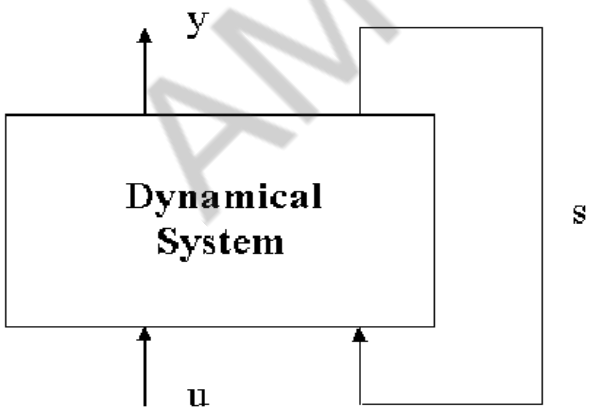

Stock Prediction is more or less like Pattern Recognition
NN is a power tool for Pattern Regnition




Stock data is highly complex and hard to model, therefore a non-linear model is benecial
A large set of interacting input series is often required to explain a specfic stock, which suites neural network

**Error Correction Neural Networks(ECNN)**

The idea is to use the previous model error as additional information to the system
Recurrent system is described as
st = f (st-1 , ut ) state transition
yt = g (st) output equation




**Developing ECNN**

Functions f and g are not specied, yt is the computed output and st describes the state.
st = f (st-1 , ut , yt-1 - ytd )
yt = g (st )
ytd is observed data
f and g can be stated as

$$\min_{f,g} \frac{1}{T} \sum_{t=1}^{T} (y_t - y_t^d)^2$$

Adding NN Role

We implement a NN
st = N(st-1 , ut , yt-1-yt-1d ; v )
yt = N(st ; w)
Now optimisation problem is
We apply an activation function
st = tanh(Ast-1 + But + D(Cst-1-yt-1d ))
yt = C (st )

$$\min_{v,w} \frac{1}{T} \sum_{t=1}^{T} (y_t - y_t^d)^2.$$

**Final ECNN**

weights v = {A, B , D} and w = {C }
A and DC could code the autoregressive structure, so non-linearity is added
st = tanh(Ast-1 + But + D tanh(Cst-1-yt-1d ))
yt = C (st )
New optimisation problem is

$$\min_{A,B,C,D} \frac{1}{T} \sum_{t=1}^{T} (y_t - y_t^d)^2$$

**Learning Algorithm**

use back-propagation technique
wk+1 = wk + □dk
dk - search direction and □ - learning rate
We use vario-eta algorithm in which we give a weight specific factor □ is related to each weight

- A performance method in itself is not sufficient for a satisfying evaluation.
- Benchmark is a different algorithm used for comparison.

- A good prediction algorithm should outperform the naive algorithm, i.e. predicted value of stock in next time step is same as the present value.
- Naive algorithm is a direct consequence of Efficient Market Hypothesis which states that the current market price is an assimilation of all information available therefore no changes of future changes can be made.

Terms

- Rkt is the k-step return at time t.

- The predicted k-step return at time t is given by capped Rkt.
- sign(x) gives the sign of the x.

**Performance measures**

Hit Rate - accounts the number of times direction of the stock is same as predicted
Return of investment – takes into account the sign and the quantity of actual return
Realised Potential – shows:

- how much of the total
- movement algorithm
- successfully identifies.

$$H_R = \frac{|\{t|R_t^k \hat{R}_t^k > 0, t = 1, \ldots, N\}|}{|\{t|R_t^k \hat{R}_t^k \neq 0, t = 1, \ldots, N\}|}$$

$$ROI = \sum_{t=1}^{T} R_t \cdot sign(\hat{R}_t).$$

$$RP = \frac{\sum_{t=1}^{T} R_t \cdot sign(\hat{R}_t)}{\sum_{t=1}^{T} |R_t|}$$

## EMPIRICAL STUDY OF THE STOCK MARKET ANALYSIS

### Empirical Study

- Well traded stocks with a reasonable spread are considered.
- Certain time invariant structures are identified and learnt quickly so in latter part of the training some weights are frozen.
- Occurrences of invariant structures were more evident in weekly rather than daily structures.

### Data series

Closing price y
Highest price during the day yH .
Lowest price during the day yL .
Volume V , the total amount of stocks traded during the day.

### Training procedure

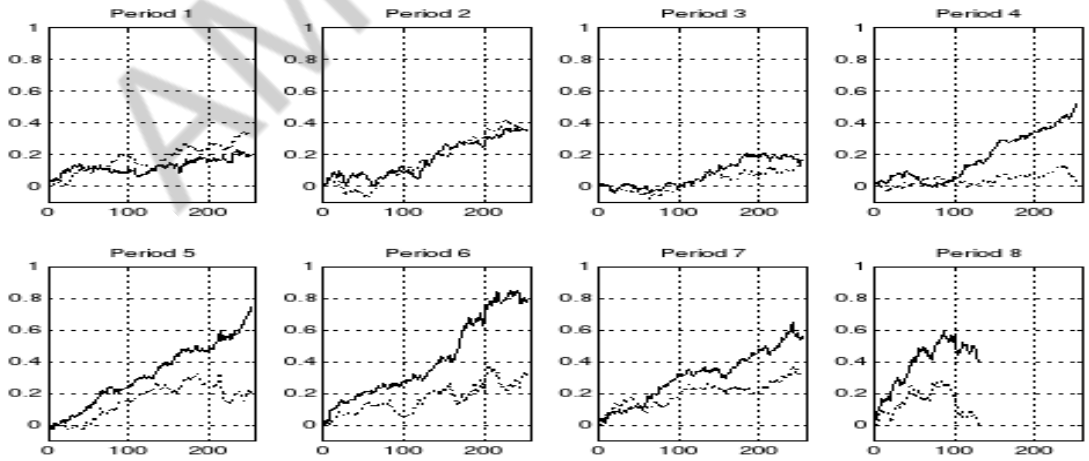
Data was divided into 3 subsets Training set, Validation set, Generalization set.
Weights were initialized uniformly in the range [-1,1].
After training waights associated with the best performance in the Validation set were selected and applied to Generalization set to get final results.

| 1-day forecast (%) | | | | |
|---|---|---|---|---|
| Period | $HR_{ECNN}$ | $RP_{ECNN}$ | $HR_{naive}$ | $RP_{naive}$ |
| 1, Jan. 93 to Dec. 93 | 52.4 | 10.8 | 54.7 | 22.3 |
| 2, Jan. 94 to Dec. 94 | 56.3 | 17.8 | 52.0 | 15.2 |
| 3, Jan. 95 to Dec. 95 | 54.5 | 11.0 | 51.4 | 8.9 |
| 4, Jan. 96 to Dec. 96 | 57.9 | 24.5 | 50.0 | 1.7 |
| 5, Jan. 97 to Dec. 97 | 59.1 | 32.4 | 52.8 | 12.5 |
| 6, Jan. 98 to Dec. 98 | 57.9 | 24.4 | 53.5 | 18.3 |
| 7, Jan. 99 to Dec. 99 | 60.1 | 26.1 | 53.9 | 12.0 |
| 8, Jan. 00 to June 00 | 56.2 | 21.0 | 48.5 | 4.3 |
| mean | **56.8** | **21.0** | **52.1** | **11.9** |

One day forecast of Swedish stock Exchange

| 1-week forecast (%) | | | | |
|---|---|---|---|---|
| Period | $HR_{ECNN}$ | $RP_{ECNN}$ | $HR_{naive}$ | $RP_{naive}$ |
| 1, Jan. 93 to Dec. 93 | 42.2 | -12.9 | 42.2 | -34.1 |
| 2, Jan. 94 to Dec. 94 | 48.9 | 22.1 | 44.4 | 0.3 |
| 3, Jan. 95 to Dec. 95 | 47.8 | -8.0 | 50.0 | 18.9 |
| 4, Jan. 96 to Dec. 96 | 62.2 | 25.7 | 55.6 | -19.0 |
| 5, Jan. 97 to Dec. 97 | 51.1 | -0.9 | 60.0 | 2.9 |
| 6, Jan. 98 to Dec. 98 | 57.8 | -6.5 | 55.6 | -1.0 |
| 7, Jan. 99 to Dec. 99 | 60.0 | 25.9 | 44.4 | -18.8 |
| mean | **52.9** | **6.5** | **50.3** | **-7.3** |

weekly forecast




Daily Prediction

Thus the Stock market prediction is depends on the human information processing capabilities and the nature and efficiency of the technique used for the same.

**********

## 2. CASE STUDY : SENTIMENT ANALYSIS

**CASE STUDY : Sentiment Analysis**

**What is Sentiment Analysis?**

- Sentiment analysis is:
  - ***the detection of attitudes***

"enduring, affectively colored beliefs, dispositions towards objects or persons"

1. **Holder (source)** of attitude
2. **Target (aspect)** of attitude
3. **Type** of attitude
   - From a set of types
     - *Like, love, hate, value, desire,* etc.
   - Or (more commonly) simple weighted **polarity**:
     - *positive, negative, neutral,* together with *strength*
4. **Text** containing the attitude
   - Sentence or entire document

**Sentiment analysis has many other names**

1. Opinion extraction
2. Opinion mining
3. Sentiment mining
4. Subjectivity analysis

**Why sentiment analysis?**

- *SA is required whether there is a need of progress or growth in terms of positive or negative opinion of an organization and do follow up actions against or for it.*
- *Movie*: is this review positive or negative?
- *Products*: what do people think about the new iPhone?
- *Public sentiment*: how is consumer confidence? Is despair increasing?

- *Politics*: what do people think about this candidate or issue?
- *Prediction*: predict election outcomes or market trends from sentiment

**Scherer Typology of Affective States**

- **Emotion**: brief organically synchronized … evaluation of a major event
  - *angry, sad, joyful, fearful, ashamed, proud, elated*
- **Mood**: diffuse non-caused low-intensity long-duration change in subjective feeling
  - *cheerful, gloomy, irritable, listless, depressed, buoyant*
- **Interpersonal stances**: affective stance toward another person in a specific interaction
  - *friendly, flirtatious, distant, cold, warm, supportive, contemptuous*
- **Attitudes**: enduring, affectively colored beliefs, dispositions towards objects or persons
  - *liking, loving, hating, valuing, desiring*
- **Personality traits**: stable personality dispositions and typical behavior tendencies
  - *nervous, anxious, reckless, morose, hostile, jealous*

**Computational work on other affective states**

- **Emotion**:
  - Detecting annoyed callers to dialogue system
  - Detecting confused/frustrated versus confident students
- **Mood**:
  - Finding traumatized or depressed writers
- **Interpersonal stances**:
  - Detection of flirtation or friendliness in conversations
- **Personality traits**:
  - Detection of extroverts

Detection of Friendliness

- Friendly speakers use collaborative conversational style
  - Laughter
  - Less use of negative emotional words
  - More sympathy
    - That's too bad I'm sorry to hear that
  - More agreement
    - I think so too
  - Less hedges
    - kind of sort of a little …

**Sentiment Analysis overview**

- ***Simplest task:***
  - Is the attitude of this text positive or negative?
- ***More complex:***

- Rank the attitude of this text from 1 to 5

- ***Advanced:***
  - Detect the target, source, or complex attitude types

- Simplest task (of Sentiment Analysis) :

***Scenario :***

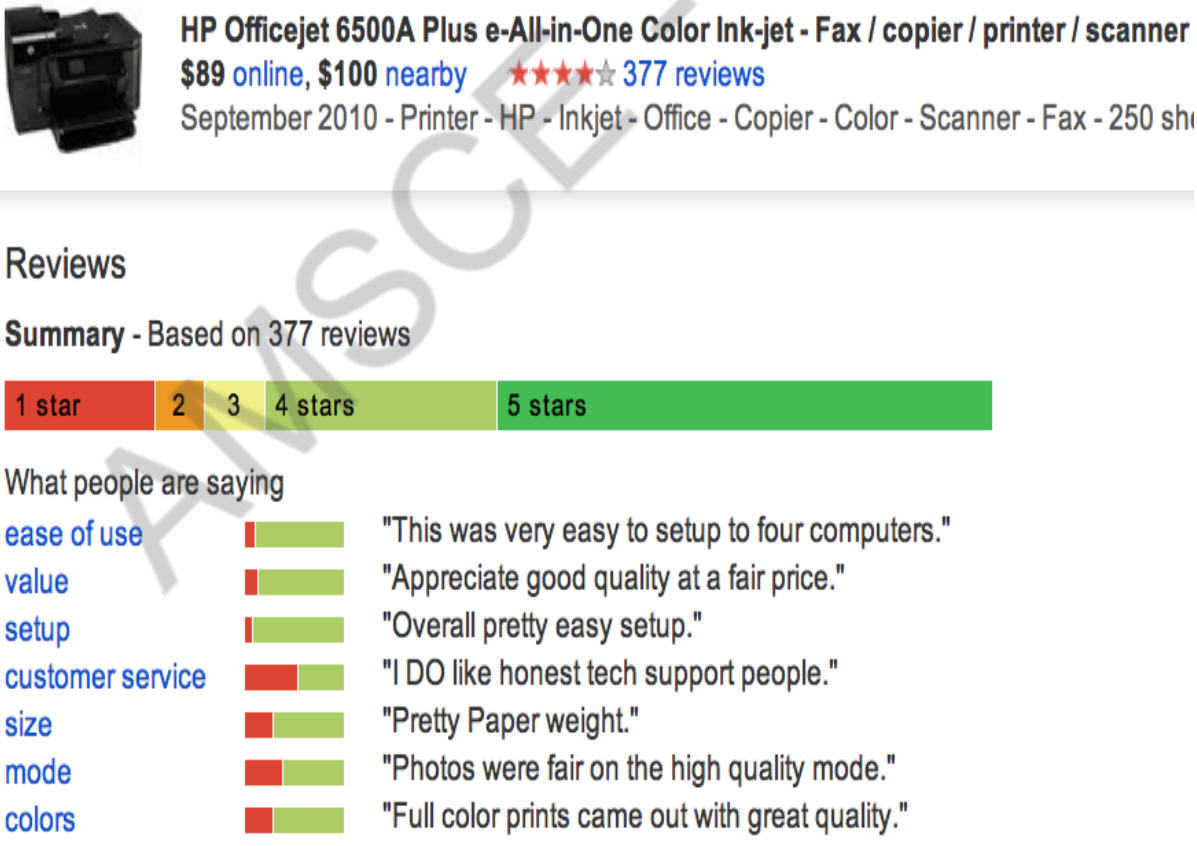
Study of Human behavioral outcomes based on his or her **Affective States.**

- **Is the attitude of this text positive or negative?**

**Positive or negative movie review?**

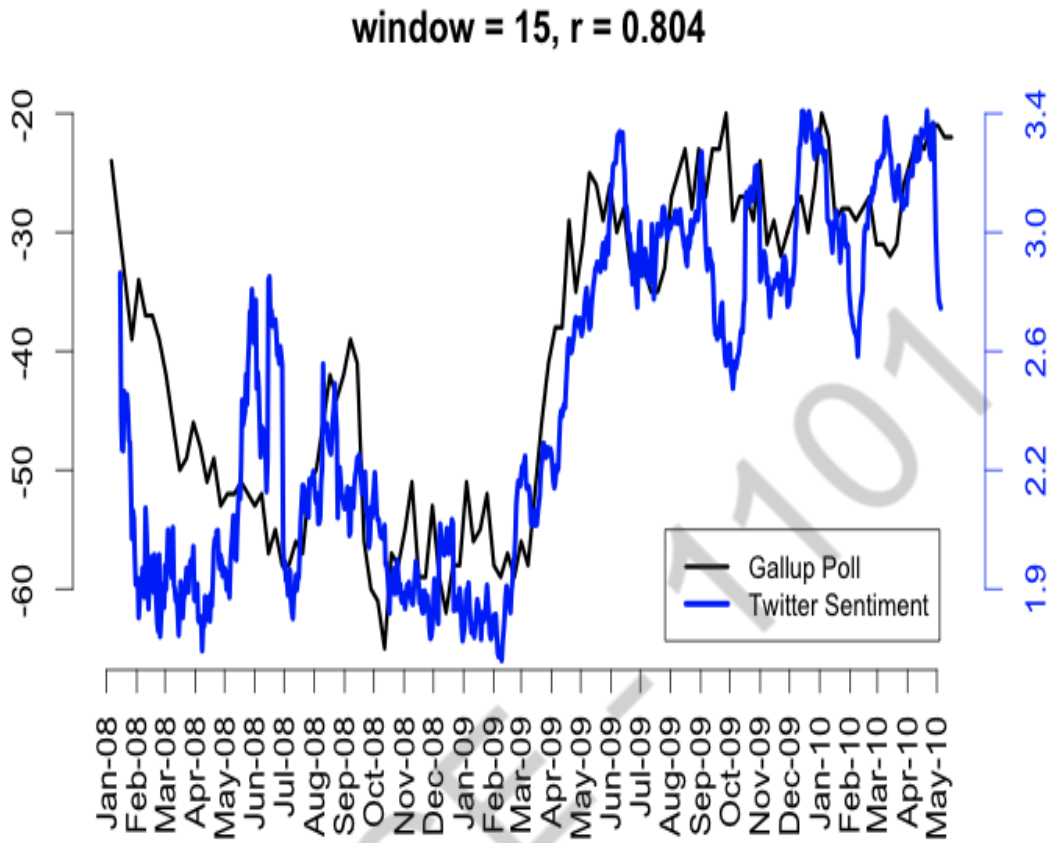
- unbelievably disappointing
- Full of zany characters and richly applied satire, and some great plot twists
- this is the greatest screwball comedy ever filmed
- It was pathetic. The worst part about it was the boxing scenes.
- ***More complex:***
  - Rank the attitude of this text from 1 to 5

Google Product Search




Twitter sentiment versus Gallup Poll of Consumer Confidence

Brendan O'Connor, Ramnath Balasubramanyan, Bryan R. Routledge, and Noah A. Smith. 2010. From Tweets to Polls: Linking Text Sentiment to Public Opinion Time Series. In ICWSM-2010




A Baseline Algorithm

**Sentiment Classification in Movie Reviews**

- Polarity detection:
  - Is an IMDB movie review positive or negative?
- Data: *Polarity Data 2.0:*
  - http://www.cs.cornell.edu/people/pabo/movie-review-data

- Tokenization
- Feature Extraction
- Classification using different classifiers
  - Naïve Bayes
  - MaxEnt
  - SVM

Sentiment Tokenization Issues

- Deal with HTML and XML markup

- Twitter mark-up (names, hash tags)
- Capitalization (preserve for words in all caps)
- Phone numbers, dates
- Emoticons
- Useful code:
  - Christopher Potts sentiment tokenizer
  - Brendan O'Connor twitter tokenizer

Potts emoticons

```
[<>]?                 # optional hat/brow
[:;=8]                # eyes
[\-o\*\']?            # optional nose
[\)\]\(\[dDpP/\:\}\{@\|\\]  # mouth
|                     #### reverse orientation
[\)\]\(\[dDpP/\:\}\{@\|\\]  # mouth
[\-o\*\']?            # optional nose
[:;=8]                # eyes
[<>]?                 # optional hat/brow
```

**Extracting Features for Sentiment Classification**

- How to handle negation
  - I **didn't** like this movie

vs

  - I really like this movie
- Which words to use?
  - Only adjectives
  - All words
    - All words turns out to work better, at least on this data

Negation

Add NOT_ to every word between negation and following punctuation:

didn't like this movie , but I

didn't NOT_like NOT_this NOT_movie but I

Reminder: Naïve Bayes

$$c_{NB} = \underset{c_j \in C}{\operatorname{argmax}} P(c_j) \prod_{i \in positions} P(w_i \mid c_j)$$

$$\hat{P}(w \mid c) = \frac{count(w, c) + 1}{count(c) + |V|}$$

**Binarized (Boolean feature) Multinomial Naïve Bayes**

- Intuition:
  - For sentiment (and probably for other text classification domains)
  - Word occurrence may matter more than word frequency
    - The occurrence of the word *fantastic* tells us a lot
    - The fact that it occurs 5 times may not tell us much more.
  - Boolean Multinomial Naïve Bayes
    - Clips all the word counts in each document at 1

**Boolean Multinomial Naïve Bayes: Learning**

From training corpus, extract *Vocabulary*

- Calculate $P(c_j)$ terms
  - For each $c_j$ in $C$ do

$docs_j \leftarrow$ all docs with class $= c_j$

$$P(c_j) \leftarrow \frac{|docs_j|}{|\text{total \# documents}|}$$

Calculate $P(w_k \mid c_j)$ terms

$Text_j \leftarrow$ single doc containing all $docs_j$

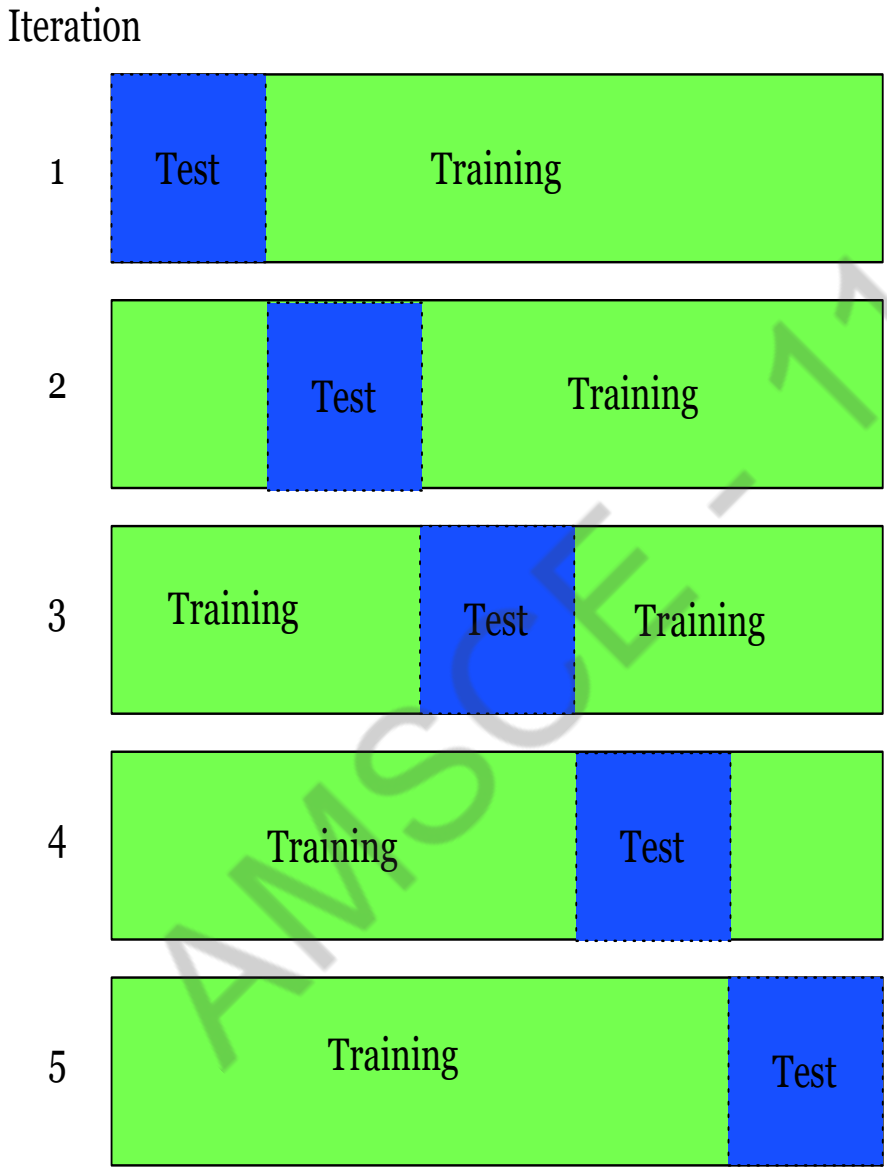
For each word $w_k$ in *Vocabulary*

$n_k \leftarrow$ # of occurrences of $w_k$ in $Text_j$

$$P(w_k \mid c_j) \leftarrow \frac{n_k + a}{n + a|Vocabulary|}$$

**Cross-Validation**

- Break up data into 10 folds
  - (Equal positive and negative inside each fold?)
- For each fold
  - Choose the fold as a temporary test set
  - Train on 9 folds, compute performance on the test fold
- Report average performance of the 10 runs




***Other issues in Classification***

- MaxEnt and SVM tend to do better than Naïve Bayes

Thus the sentiment analysis requires the much needed attention to get good recommendation for the given scenario.

# CS8091 BIG DATA ANALYTICS

## UNIT V NO SQL DATA MANAGEMENT FOR BIG DATA AND VISUALIZATION

## QUESTION BANK

### PART A

**1. Define NO SQL Database.**

A NoSQL (originally referring to "non SQL" or "non relational") database provides a mechanism for storage and retrieval of data that is modelled in means other than the tabular relations used in relational databases. Such databases have existed since the late 1960s, but did not obtain the "NoSQL" moniker until a surge of popularity, triggered by the needs of Web 2.0 companies. NoSQL databases are increasingly used in big data and real-time web applications.[5] NoSQL systems are also sometimes called "Not only SQL" to emphasize that they may support SQL-like query languages, or sit alongside SQL databases in polyglot persistent architectures.

**2. What is Schema-less Model?**

schema less means the database don't have fixed data structure, such as MongoDB, it has JSON-style data store, you can change the data structure as you wish.

**3. Define Key Value Stores.**

A key-value store is a database which uses an array of keys where each key is associated with only one value in a collection. It is quite similar to a dictionary or a map data structure. Key-value stores can be considered as the most primary and the simplest version of all the databases.

**4. Define Document Stores.**

A document-oriented database, or document store, is a computer program designed for storing, retrieving and managing document-oriented information, also known as semi-structured data

Document-oriented databases are one of the main categories of NoSQL databases, and the popularity of the term "document-oriented database" has grown[2] with the use of the term NoSQL itself. XML databases are a subclass of document-oriented databases that are optimized to work with XML documents. Graph databases are similar, but add another layer, the relationship, which allows them to link documents for rapid traversal.

**5. Define Tabular Stores.**

A tabular database, as the name implies is a database that is structured in a tabular form. It arranges data elements in vertical columns and horizontal rows. Each cell is formed by the intersection of a column and row. Each row and column is uniquely numbered to make it orderly and efficient.

**6. Define Object Data Stores.**

Object storage (also known as object-based storage) is a computer data storage architecture that manages data as objects, as opposed to other storage architectures like file systems which manages data as a file hierarchy, and block storage which manages data as blocks within sectors and tracks.

**7. What is Hive?**

•Hive is a data ware house system for Hadoop.

•It runs SQL like queries called HQL (Hive query language) which gets internally converted t map reduce jobs.

•Hive was developed by Facebook.

•Hive supports Data definition Language(DDL), Data Manipulation Language(DML) and user defined functions

**8. What is Sharding?**

Sharding is a database architecture pattern related to horizontal partitioning — the practice of separating one table's rows into multiple different tables, known as partitions. Each partition has the same schema and columns, but also entirely different rows.

**9. What is Hbase?**

HBase is an open-source non-relational distributed database modeled after Google's Bigtable and written in Java. It is developed as part of Apache Software Foundation's Apache Hadoop project and runs on top of HDFS or Alluxio, providing Bigtable-like capabilities for Hadoop.

**10. Difference Between Hbase and Hive.**

| **HBase** | **Hive** |
|---|---|
| • Despite providing SQL functionality | • Apache **HBase** is a NoSQL key/value store which runs on top of HDFS. |
| • Unlike **Hive**, **HBase** operations run in real-time on its database rather than MapReduce jobs. | • Hive does not fully runs in real-time |
| • HBase provides Interactive query. | • , **Hive** does not provide interactive querying yet - it only runs batch processes on Hadoop. |

**PART B**

**1. Explain in detail about HIVE.**

INTRODUCTION TO HIVE

• Apache Hive is a data ware house system for Hadoop that runs SQL like queries called HQL (Hive query language) which gets internally converted to map reduce jobs.

• Hive was developed by Facebook.

• It supports Data definition Language, Data Manipulation Language and user defined functions.

• Before learning Hive, you must have the knowledge of Hadoop and Java.

What is HIVE?

• Hive is a data ware house system for Hadoop.

• It runs SQL like queries called HQL (Hive query language) which gets internally converted to map reduce jobs.

• Hive was developed by Facebook.

• Hive supports Data definition Language(DDL), Data Manipulation Language(DML) and user defined functions.

DDL: create table, create index, create views.

DML: Select, Where, group by, Join, Order By

Pluggable Functions:

UDF: User Defined Function

UDAF: User Defined Aggregate Function

UDTF: User Defined Table Function

What HIVE is not?

Hive is not RDBMS.

Hive is not used for OLTP(Online Transaction Processing).

Even with small amount of data time to return the response can?t be compared to RDBMS.

Points to remember

Hive's metastore is used to persist schema i.e. table definition(table name, columns, types), location of table files, row format of table files, storage format of files.

Hive Query Language is similar to SQL and gets reduced to map reduce jobs in backend.

Hive's default database is derby.

Hive Components

- Shell: allows interactive queries
- Driver: session handles, fetch, execute
- Compiler: parse, plan, optimize
- Execution engine: DAG of stages (MR, HDFS, metadata)
- Metastore: schema, location in HDFS, SerDe
- Hadoop is great for large-data processing!
  - But writing Java programs for everything is verbose and slow
  - Not everyone wants to (or can) write Java code
- Solution: develop higher-level data processing languages
  - Hive: HQL is like SQL
  - Pig: Pig Latin is a bit like Perl

**Data Model**

- Tables
  - Typed columns (int, float, string, boolean)
  - Also, list: map (for JSON-like data)
- Partitions
  - For example, range-partition tables by date
- Buckets
  - Hash partitions within ranges (useful for sampling, join optimization)

Metastore

- Database: namespace containing a set of tables
- Holds table definitions (column types, physical layout)
- Holds partitioning information
- Can be stored in Derby, MySQL, and many other relational databases

Physical Layout

- Warehouse directory in HDFS
- E.g., /user/hive/warehouse
- Tables stored in subdirectories of warehouse
- Partitions form subdirectories of tables
- Actual data stored in flat files
- Control char-delimited text, or SequenceFiles
- With custom SerDe, can use arbitrary format

**Hive: Example**

- Hive looks similar to an SQL database
- Relational join on two tables:
- Table of word counts from Shakespeare collection
- Table of word counts from the bible

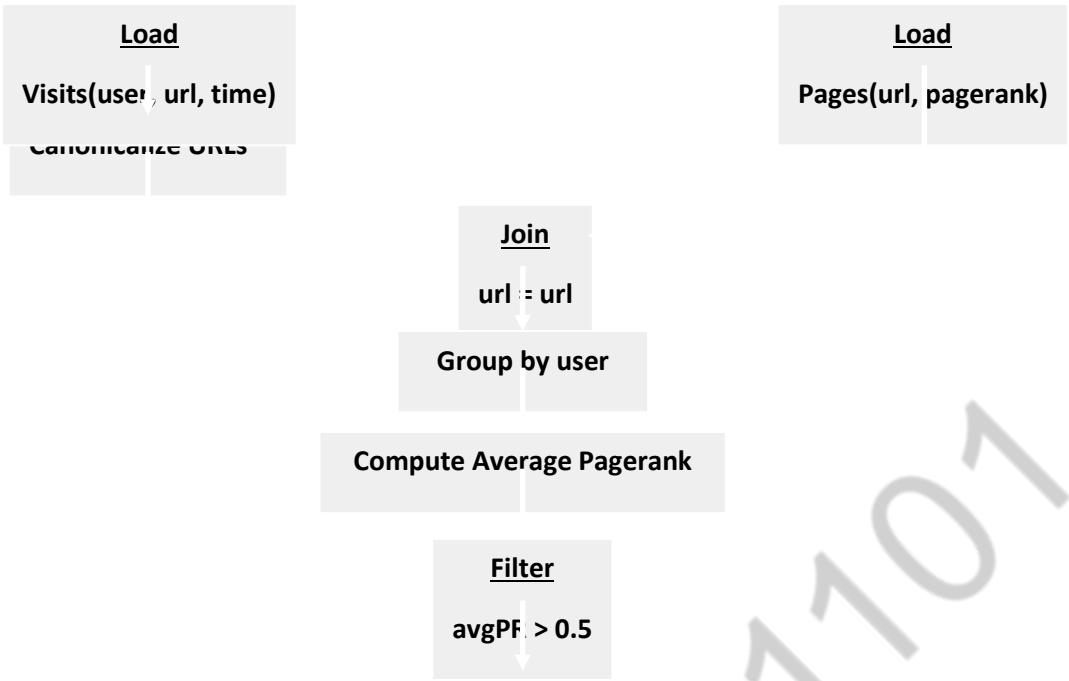
```
SELECT s.word, s.freq, k.freq FROM shakespeare s
  JOIN bible k ON (s.word = k.word) WHERE s.freq >= 1 AND k.freq >= 1
  ORDER BY s.freq DESC LIMIT 10;
```

| | | |
|---|---|---|
| the | 25848 | 62394 |
| I | 23031 | 8854 |
| and | 19671 | 38985 |
| to | 18038 | 13526 |
| of | 16700 | 34654 |
| a | 14170 | 8057 |
| you | 12702 | 2720 |
| my | 11297 | 4135 |
| in | 10797 | 12445 |
| is | 8882 | 6884 |

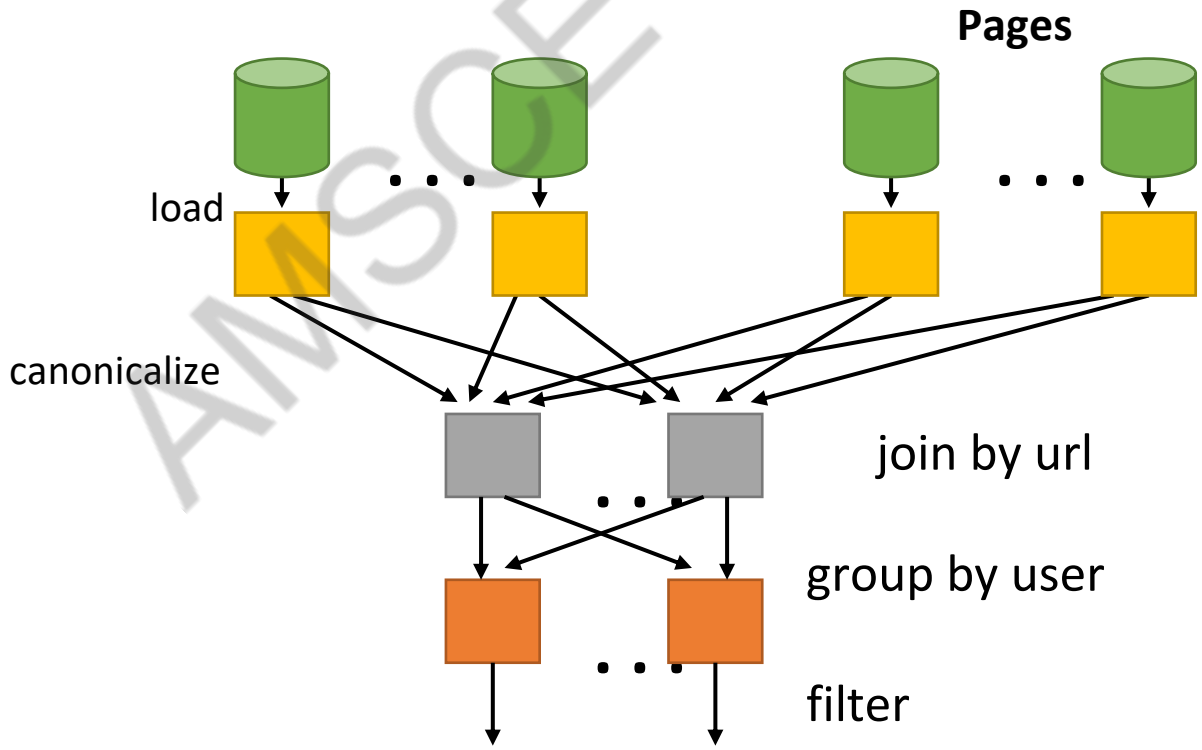
Example Data Analysis Task using HIVE

Find users who tend to visit “good” pages.

Conceptual Dataflow




System-Level Dataflow




Thus the HIVE data flow mechanism is used for data storage and retrieval make easy.

**2. Give a Survey on Analyzing Big data with Twitter.**

- Twitter (www.twitter.com) is an online social networking and micro blogging platform that enables a user to send and read short 140 character text messages, called "tweets".

Twitter has now become a global platform for mass communication. The data it generates is finding many uses:-

- in disaster management,
- brand management (understanding brand strengths, sentiments, visibility),
- politics (campaign management),
- personal (giving updates to family/friends) etc.

Features of Twitter

- It is ranked amongst the top 10 most visited websites by Alexa's web traffic analysis for the last 2 years.
- It has ~250+ million monthly active users who send about ~500+ million tweets per day (ref. Wikipedia). 78% of the twitter users access it on a mobile device and 77% are living outside US.
- It supports 35+ languages. Its vast global reach and usage has made Twitter into a remarkable platform for analysing and understanding trends and events shaping the world either on a global or a local scale.
- It is a great medium to understand ground realities of global events (like natural disasters, terror-attacks) as users tweet their impressions even before news crews are able to reach the areas.
- Given the reach, businesses, politicians, and celebrities are using it to market their messages / views / opinions to their followers. The volume, velocity and variety of data that gets generated on twitter fits the description of big data. We now explore some approaches on analysing the same.

ii. Twitter Data Analysis

- Twitter shares its data in document store format (JSON – JavaScript Object Notation) and allows developers to access it using APIs.
- Twitter APIs can be accessed only via authenticated requests. Twitter uses Open Authentication (OAuth) and each request must be signed with valid twitter credentials.

- OAuth provide a safer alternative because the user generally logs into twitter and approves sharing of his information to each application.
- The user can anytime de-authenticate certain applications from accessing his tweets.
- API users can use the REST API (for pull access – where we must specify the user credential whose data is sought) or Streaming API (for push access – once a request is made they provide continuous stream of updates with no further input required from user).
- Twitter also places something called a Rate Limit (restriction on amount of data calls that a particular application can make) based on which data gets shared.
- Once we have received data from twitter and placed it in our NOSQL database we can apply Map-Reduce / Graph theory approaches to analyse the same.

- Map Reduce - is a programming model for processing large data sets with a parallel, distributed algorithm on a cluster.
- It allows developers to write programs that process massive amounts of unstructured data in parallel across a distributed cluster or even standalone computers.
- Originally, it was developed at Google for indexing Web pages but it has now become a generic term. It has two main functions – Map and Reduce.

Map Reduce Analysis of Twitter Data

a) Trending – By analysing individual tweets and looking for certain words amongst them and using map- reduce we can filter out key words that are trending (or being used by a lot of users on twitters). It gives view of what are the key topics.

b) Sentiment Analysis – looking for keywords about brands and analysing them to compute a score of sentiment for that brand.

Graph Theory is the study of graphs, which are mathematical structures used to model pairwise relations between objects.

A "graph" is made up of "vertices" or "nodes" and lines called edges that connect them.

A graph may be undirected, meaning that there is no distinction between the two vertices associated with each edge, or its edges may be directed from one vertex to another (ref. Wikipedia).

Relationships in a social network are treated like graphs.

For example, User (A) is a friend of User (B),User (B) is a friend of User (A) and a follower of User (C). We can depict this relationship like a graph such as the one shown in figure1.

Unlike Facebook, where a connection is bi-directional (if A is a friend of B then B is a friend of A – both are friends and follow each other),

Twitter's relationships are asymmetric.

On Twitter, if A is following B then A is a friend of B, but B is not friend of A.

A will get tweets of B reporting on his home page but B will not see tweet of A until he/she explicitly starts to follow A.

We can use concepts of Graph Theory to analyze twitter data.

Graph Theory Analysis of Twitter Data

c) Influencer – Re tweeting (Degree Centrality) – Has two approaches in-degree and out-degree. How many times a user re-tweets others messages (out-degree) and how many times messages of a user are re-tweeted by others (in-degree) give a measure of influence of the user.

d) Shortest Path / Most Network Connections – Allow us to figure out who are most influences. And we can use it for other analysis as well – depth of network etc.

iii. Twitter Analytic Tools.

The tools use approaches highlighted above (on analysis) and many of their own interpretations to arrive at specific analysis for their users.

a) Twitter Counter:

- Sstarted as a self-funded start up in Amsterdam in 2008.
- It is a third party application that provides analytics of twitter usage.
- It allows some basic features as free use and other features tagged as premium which comes with subscription.

b) Twitonomy: -

- It provides a good basic statistics free and for other features user needs to sign up for a pro account

c) Twtrland:-

- It provides some niche features like best time to tweet free along with additional statistics, but keeps things like export, prediction, comparison in its paid version.

d) SocialBro: –

- It provides free access to all its features but only for 2 weeks, after which user needs to pay subscription to continue using the services.

Key Features of Twitter Analytics Tools

We have categorized features being provided by various players into following buckets and then mapped the tools for their availability / non availability in both free / paid versions.

1) Basic Stats – Number of Tweets, Followers etc.

2) Historical Data – Ability to provide historical information about various statistics.

3) Exporting Excel / PDF – Ability to export information on search results / user statistics to excel or PDF for further analysis.

4) Additional Stats – Number of Mentions, Re tweets, Influence categories – which tell how much impact a user's tweet having on his network.

5) Follower Retention / Churn – Tool allows one to become aware of the followers who are no longer following a user. It lists new followers added / left during the week.

6) Prediction – Certain tools allow prediction – what will the number of followers for a particular user based on his tweeting history / some other metric.

7) Comparison – Allow comparison of twitter statics like tweets / followers / retweets between two different users ids.

8) Ability to add other users / lists tracking – Allows users to add other users / lists of users / communities for further tracking.

9) Geospatial visualization – Whether the tool provides map based information. Tweets shown by location on a map to see a visual appeal of influence / other metrics.

10) Best Time to Tweet – Certain tools predict what will be the best time to tweet based on history of user's tweets and responses received on them.

11) Conversation Analysis – Semantic analysis on tweets.

12) Real-time Analytics – providing statics based on real-time data – tweets posted by followers

/ followed and their impact.

13) Schedule time to share Content – certain tools provide an option to send content at predefined scheduled time or based on analytics release it to the followers for maximum impact.

Table I - Comparison of Twitter Analytics Tools

Based on data given in table 1), we can see that online tools provide a lot of analytics to their users but most of them are available in the paid versions. In free versions, most tools only provide basic data like number of tweets, followers, followed, what is trending etc.

Conclusion

- Different users of Twitter have different needs of analytics of their twitter usage.
- Hence before deciding which tool to use, they should search the internet to find a tool that suits their particular need best.
- As there are many tools available and we have only compared 4 of them.
- Based on our data set, we feel that if any user is looking for a tool that provides some very good features in its free version than Twtrland / TwiterCounter are our recommendations.
- TwitterCounter for its simple metrics and ability to provide comparisons and Twtrland for some of the advanced analytics it offers free.

**3. Discuss about How E-Commerce is Using Big Data to Improve Business in detail.**

- Big Data is very broad term in the data analytics world. There are several successful big data ECommerce case studies which proof the importance.
- Many big unicorns are using big data and Hadoop to understand the behavior of their customers and increase sales. But knowing the behavior of your customers is not that easy.
- There are many challenges like data analysis, search, curation, storage, visualization, and privacy.
- But companies who have implemented big data or using data to analyze the customer behavior, sales and market perception are doing amazingly well.

Amazing Big Data Ecommerce Facts & Figures

- More than 91% marketing leaders believe companies take a decision based on the customer data. [Source: iab.net]
- 85% of large corporates are using social media to drive as a marketing tool. [Source: iab.net]
- But the sad part is 39% of marketers say they can't turn their data into actionable insight. This is the sad part of big data. [Source: iab.net]
- 86% customers are ready to pay more to get better customer experience. [source: lunch pail]
- 87% marketing managers agree to the fact that right data is at most required to measure the ROI in their own company effectively.
- Companies are successfully saving 25% budget on ads by optimizing their media inventory. [source: vemployee]
- When Amazon implemented "Customers who bought this item also bought" recommendation feature, it had increased its sales by nearly 29%. [source: Fortune]
- Magaseek, Japanese's top fashion retailer, increased its revenue by ¥40 million with optimization brought in through Analytics. [source: vemployee]
- Close to 70% users on Social media buy bases recommendations from others and another statistic says that upwards of 40% of users who give a positive review on social media purchase a product. [Source: Quora]
- 54% of the survey says big data has brought gains in multi-channels sales. [source: Wipro]

Big Data Ecommerce Case Studies to understand use of Big Data in Ecommerce sector

- Here are some of the most successful case studies in ecommerce and retail industry which will inspire you to use data even more correctly.

- These companies are the market leaders in their niche and have reported some huge profit in business after using big data.
- Let's start with the Big Data case ecommerce case studies and few big data retail case studies where big data is providing ROI.

The below successful big data ecommerce case studies show how Ecommerce industry are using Big Data, Hadoop, Machine Learning, and Analytics to scale the business.

**Alibaba Big Data Case Study**

- This China-based billion dollar Ecommerce Company and world's largest retailer has revealed how big data has helped them to increase the revenue.
- It has also helped them to use the data in the offline sector and another retail sector to grow the business.
- According to Danfeng Li (Alibaba's director of big data and technologies), Alibaba's various properties hold about 80% of China's PC, internet and app data, which it can integrate with first-party data to create singularly powerful data models.
- According to the report by Reuters, the number of mobile-first users have been increased to 42% and has reached to 410 Million. This accounts for the total of 73% GMV. Check this WARC report for more details.

**Aetna Big Data Case Study**

- AETNA is one of the largest healthcare insurance company of USA. Especially after the merger of Aetna and Humana, they have become one of the top healthcare, insurance and Wellness Company.
- With around 19 Million customers, Aetna is using big data to improve the health and diagnostic of their clients.
- After looking at the patients' metabolic syndrome-detecting tests assesses patient risk factors, patient risk factors, company is focusing on finding the top factors which will impact the patient's health most.
- These data will help 90% of patients who doesn't have a previous record, and 60% patients will increase the adherence.

**General Electric (GE) Big Data Case Study**

- GE which is mainly known for their consumer-centric business like jet engines, oil & gas plants, and several such products.
- But the industry was surprised when they launched one ad where one guy says “he got the job in software and will write code using which machine will communicate in GE”. Here is the video ad.
- In late 2011, GE made an intelligent move by bringing William Ruh from Cisco Systems to start the Big Data system in GE.
- In 2012, GE CEO Jeffrey Immelt announced to invest USD 1 Billion in the Big Data & Analytics segments over a period of four years.
- GE collects the significant data from their jet engines, turbines, trains, and medical equipment and analyzes those to enhance the business and consumer lifestyle.
- Here is a simple calculation by HP as per the details shared by GE. One of their gas turbines generates 500 GB of data. So with 12000 turbines will produce the below amount of data.
- With this GE estimates, data can boost productivity in the USA by 1.5%, which will save enough money to raise the average income of citizen by 30% over a period of 20 years.

**Kroger Big Data Case Study**

- Kroger is making use of big data through its joint venture with Dunnhumby. They collect, manage and analyze the data from their 770 Million consumers.
- Claiming 95% of sales are rung up on the loyalty card, Kroger sees an impact from its award-winning loyalty program through close to 60% redemption rates and over $12 billion in incremental revenue by using big data and analytics since 2005.

**Amazon Big Data Case Study**

- You can find plenty of Amazon big data use cases on the internet. Even Amazon is known for the Datasets for Hadoop practice.
- Amazon is an ideal example in ecommerce industry as for how big data can be used to scale the sales. For Amazon, customer satisfaction is more important than the sales or figures.
- Amazon is using the data to personalize your interaction, predicting trends and improving customer experience.
- As stated above, Amazon found around 30% increase in sales after adding the below section based on the shopping experience of other users for the same product.

**Walmart Big Data Case Study**

- This mega retail company introduced semantic data in their search platform which improved online shoppers completing a purchase by 10% to 15%. In Wal-Mart terms, that is billions of dollars.

- It was the mid of 2012 when Walmart announced the addition of Polaris, an in-house designed product to enhance the machine learning experience to their search engine. And the result of it is to the world now.

**American Express (AMEX & AIG) Big Data Case Study**

- Amex started looking at the historical transactions with 115 variables to forecast potential churn in the Australian market. And as a result, they are now able to know the 24% accounts which will get closed in next four months.
- Isn't four months sufficient to retain the customer?
- Also, American International group (AIG) is taking help of big data and visualization tools to predict and stop fraud well in advance.

**eBay Big Data Case Study**

- eBay is using big data to provide better personalization and customer experience to the users.
- Their system now handles final data velocity with 6 billion writes and 5 billion reads daily. The amount of data stored: 250 TBs.

**McDonald's Big Data Case Study**

- McDonald is one of the world's largest food chain with over 34000 local restaurants serving 69 million people in 118 countries each day.
- Their daily traffic is around 69 Million and sells around 75 burgers every second. With annual revenue of around $27 Billion and an employee strength of 750k, how can they use big data to make the customer experience better and increase sales?

McDonalds is majorly using big data to optimize the drive-thru experience. The company mainly focuses on the following three factors while making an experience better-

- Design of the drive-thru
- Information that is provided to the customer during the drive-thru and
- The people waiting in line to order at a drive-thru

And the results have been phenomenal with this implementation.