

UNIT II – MINING DATA STREAMS

QUESTION BANK

PART A (2 Marks each)

1. What you meant by stream data model? **Imp**
2. What is estimating moments?
3. What you meant by decaying window?
4. What is stock market predictions? **Imp**

PART B (5 Marks each)

5. Explain the components of streaming data architecture. **Imp**
6. Explain stream computing.
7. Explain Bloom filter analysis.
8. Explain Flajolet-Martin algorithm
9. Explain how sampling data in a stream. **Imp**
10. Explain DGIM algorithm.
11. Briefly explain the real time sentiment analyse using Tweets.

PART C (15 Marks each)

12. Explain in detail the stream data architecture. **Imp**
13. Explain in detail filtering streams. **Imp**
14. How to count distinct elements in a stream? Explain with the help of algorithm.
15. How counting 1's in a window? Explain with the help of algorithm. **Imp**
16. Explain in detail RTAP with applications. **Imp**

NOTES

STREAM DATA MODEL

- Stream is data in motion.
- Data generated continuously at high velocity and in large volumes is known as **streaming data**.
- A **stream data source** is characterized by continuous time-stamped logs in real time.
- Example: A user clicking a link on a web page, Stream data sources include:
 - ✓ Server and security logs
 - ✓ Clickstream data from websites and apps
 - ✓ IoT sensors
 - ✓ Real-time advertising platforms
- Following are the different ways of modelling data stream, querying, processing, and management:
 1. Graph model
 2. Relation-oriented stream-tuples model
 3. Object-based data stream model

- 4. XML-based data stream model
- 5. Window-based data stream model

STREAM DATA ARCHITECTURE

- It is a dedicated network of software components capable of ingesting and processing large amounts of stream data from many sources.
- It ingests data as it is generated in its raw form, stores it, and incorporate different components for real-time data processing and manipulation.
- Streaming architecture is a unique characteristics of data streams.
- It generates massive amounts of data (terabytes 10^{12} to petabytes 10^{15}).
- It is semi-structured.
- Figure shows that large data blocks in the received stream store at HDFS compatible data store or static data at disk.
- Data shards load at memory from data stores or disks for future use.
- Streaming data shards load at memory in real-time applications.
- A user application uses a query repository, which continuously sends queries for processing of the shards in-memory.
- The responses of queries save at an output buffer before they are finally retrieved by the application.

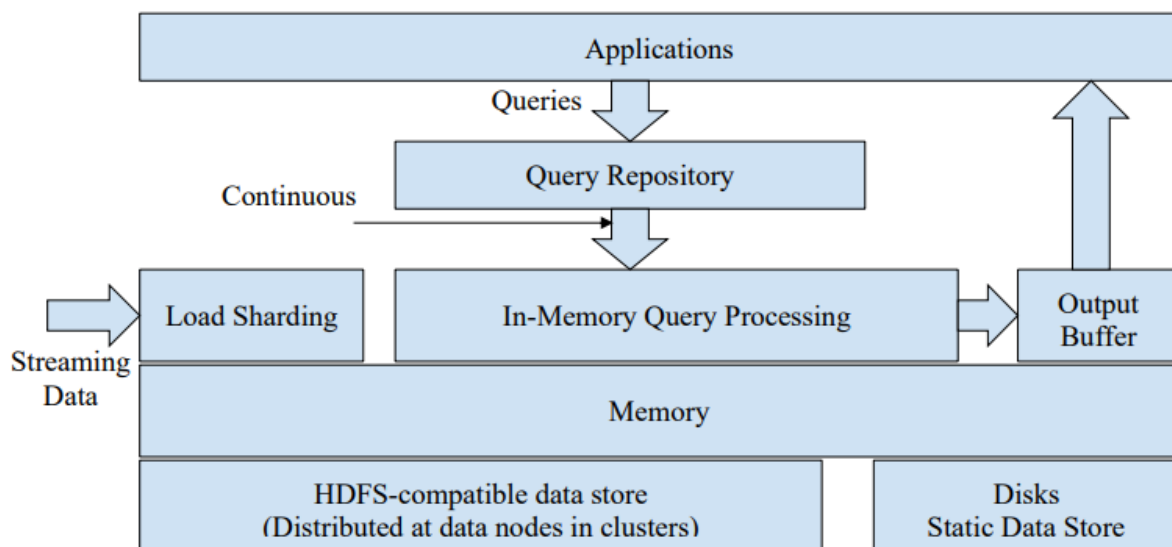









Figure : Data stream architecture for processing

COMPONENTS OF STREAMING DATA ARCHITECTURE



- Streaming data architecture include the following key component:
 1. **Message brokers:**
 - It takes data from a source, transforms it into a standard message format, and streams it on an ongoing basis, to make it available for use.
 - The popular stream processing tools are open-source software **Apache Kafka**, or **PaaS (Platform as a Service)**.

- Its components are:
 -  Azure Event Hub
 -  Azure IoT Hub
 -  GCP Cloud Pub/Sub or GCP Confluent Cloud
 -  These are cloud-native event streaming platform powered by Apache Kafka.




2. Processing tools:

- The output data streams from message broker or stream processor needs to be transformed and structured to get analysed further using analytics tools.
- The result of such analysis can be some actions, alerts, dynamic dashboards or even new data streams.
- Important open source frameworks, which focus on processing streamed data are:
 -  Apache Storm
 -  Apache Spark Streaming
 -  Apache Flink.

3. Data Analytical tools:

- Once streaming data is prepared for consumption by the stream processor and processing tool, it needs to be analyzed to provide value.
- Tools are:
 -  Apache Cassandra: It is a distributed database and it provides low latency serving of streaming events to applications.
 -  Elasticsearch: It can receive streamed data from Kafka topics directly.

4. Streaming data storage:

- Storage cost is in general relatively low, therefore organizations store their streaming data.
- A **data lake** is the most flexible and cheap option for storing event data, but it is quite challenging to properly set it up and maintain.
- The data can be store in a data warehouse.
- Data storage tools are:
 -  Kafka
 -  Databricks/Spark
 -  BigQuery

STREAM COMPUTING

- Stream computing is a way to analyse and process Big Data in real time to gain current insights.
- Stream computing has many uses, such as financial sectors for business intelligence, risk management, marketing management, etc.
- Stream computing is also used in search engines and social network analysis.

- A high-performance computer system that analyses multiple data streams from many sources live.
- Stream computing uses software algorithms that analyses the data in real time as it streams in to increase speed and accuracy when dealing with data handling and analysis.
- The stream computing consists of:
 - ✓ Collecting data readings from collections of software or hardware sensors in stream form (i.e., as an infinite series of tuples)
 - ✓ analysing the data
 - ✓ Producing actionable results(possibly in stream format as well).
- The stream computing system of IBM is known as **System S**.
 - ✓ This system runs on 800 microprocessors.
 - ✓ The System S software enables software applications to split up tasks and then reassemble the data into an answer.
- The stream computing technology of ATI is known as **Graphics Processors (GPUs)** to works in:
 - ✓ High-performance
 - ✓ Low-latency CPUs to solve complex computational problems.
- ATI's stream computing technology is derived from a class of applications that run on the GPU instead of a CPU.
- Stream computing pulls the data from the stream, processes the data, and streams it back out as a single flow. Such computing is required to process huge amounts of data at a high speed.
- Usually, a Big Data stream computing is implemented in a distributed clustered environment, as the amount of data is enormous.
- Rate of receiving data in stream is high, and the results are required in real time to make appropriate decisions or to predict new trends in the immediate future.
- Stream computing is one effective way to support Big Data by providing extremely low-latency velocities with massively parallel processing architectures.
- The efficiency of stream computing algorithms is measured using some fundamental characteristics such as:
 1. Number of passes (scans) the algorithm must make over the stream
 2. Available memory
 3. Running time of the algorithm.

SAMPLING DATA IN A STREAM

- Stream sampling is the process of collecting a representative sample of the elements of a data stream.
- Sampling in a data stream means the process of selecting a few data items from the incoming stream of data items for analysis.
- The sample is usually much smaller than the entire stream, but can be designed to retain many important characteristics of the stream, and can be used to estimate many important aggregates on the stream. Since we cannot store the entire stream, one obvious approach is to store a sample.
- Two different problems:
 - ✓ Sample a fixed proportion of elements in the stream (say 1 in 10)

- ✓ Maintain a random sample of fixed size over a finite stream.
- Unlike sampling from a stored data set, stream sampling must be performed online, when the data arrives.
- Methods of obtaining representative sample data items from a stream can be classified in two categories: **probabilistic and non-probabilistic**.
- **Probabilistic sampling** is a statistical technique used for making a choice of data items for processing. The basis of the choice is the probability of sampling the data items.
- Five probabilistic sampling methods are:
 1. Simple random sampling
 2. Systematic sampling
 3. Cluster sampling
 4. Stratified sampling
 5. Multistage sampling
- **Non-probabilistic sampling** uses arbitrary or purposive sample selection instead of sampling based on a randomized selection. This introduces bias and increases variance to the measurement data.
- General Sampling Problem are encountered while trying to find a sample of data item from an infinite length of data stream are:
 - Unknown size of data set
 - Applications that need continuous analysis, such as surveillance analysis
 - Irregular data rates are in the case of data network analysis

FILTERING STREAMS

- Due to the nature of data streams, stream filtering is one of the most useful and practical approaches to efficient stream evaluation.
- **Stream filtering is the process of selection or matching instances of a desired pattern in a continuous stream of data.**
- The **filtering steps for a stream** are:
 - (i) Accept the tuples that meet the criterion in the stream
 - (ii) Pass the accepted tuples to another process as a stream
 - (iii) Discard remaining tuples.
- Several **filtering techniques** are:
 - ✚ **Bloom Filter and its variants**
 - ✚ **Stream Quotient Filter (SQF)**
 - ✚ **Particle filter**
 - ✚ **XML filters** etc.

BLOOM FILTER ANALYSIS

- Bloom filter is a simple space-efficient data structure introduced by **Burton Howard Bloom** in 1970.
- **The filter matches the membership of an element in a dataset.**
- The filter has been widely used in **applications** such as:
 - ✓ Database applications

- ✓ Intrusion detection systems
- ✓ Query filtering
- ✓ Routing applications.
- The filter is basically a bit vector of length m that represents a set $S = \{x_1, x_2, \dots, x_n\}$ of n elements.
- Initially all bits are set to 0.
- Then, define k independent hash functions, h_1, h_2, \dots, h_k . each of which maps (hashes) some element x in set S to one of the m array positions with a uniform random distribution.
- Number k is constant, and much smaller than m .
- For each element $x \in S$.
- Figure (1) shows an example of a bloom filter with $k=3$. The filter is a bit vector of length 10.
- When a value x is inserted into the Bloom filter, the bits at position $h_1(x)$, $h_2(x)$, and $h_3(x)$ are set to 1.
- To detect whether a value y is in the filter or not, the bits of position $h_1(y)$, $h_2(y)$, and $h_3(y)$ are thus checked.

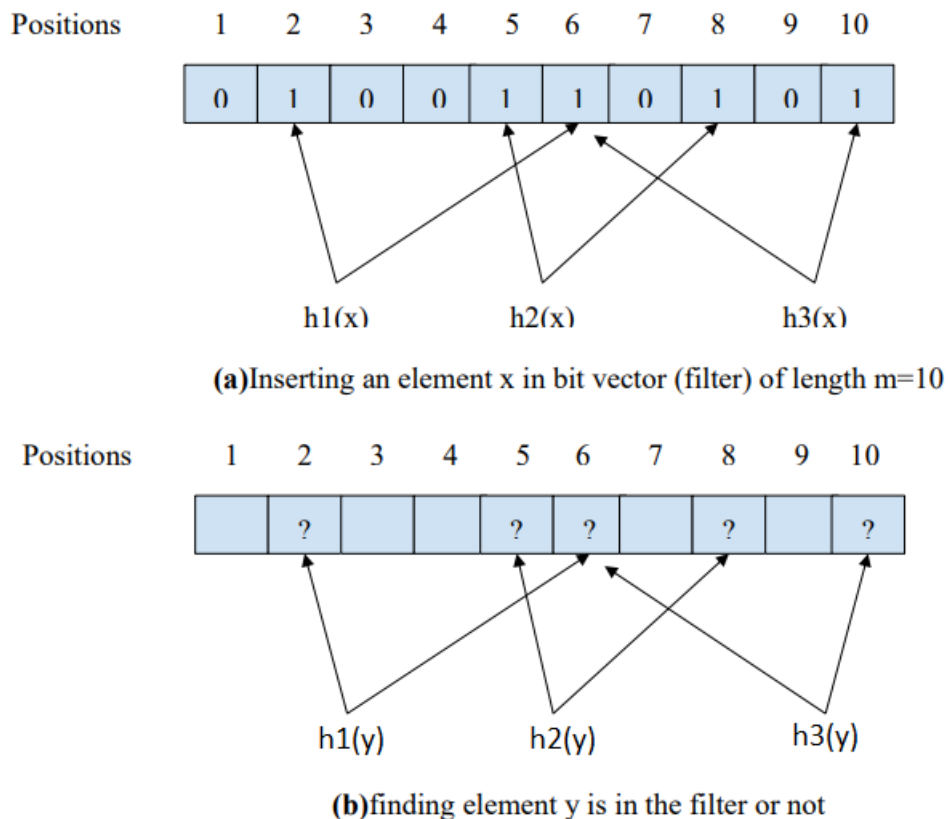


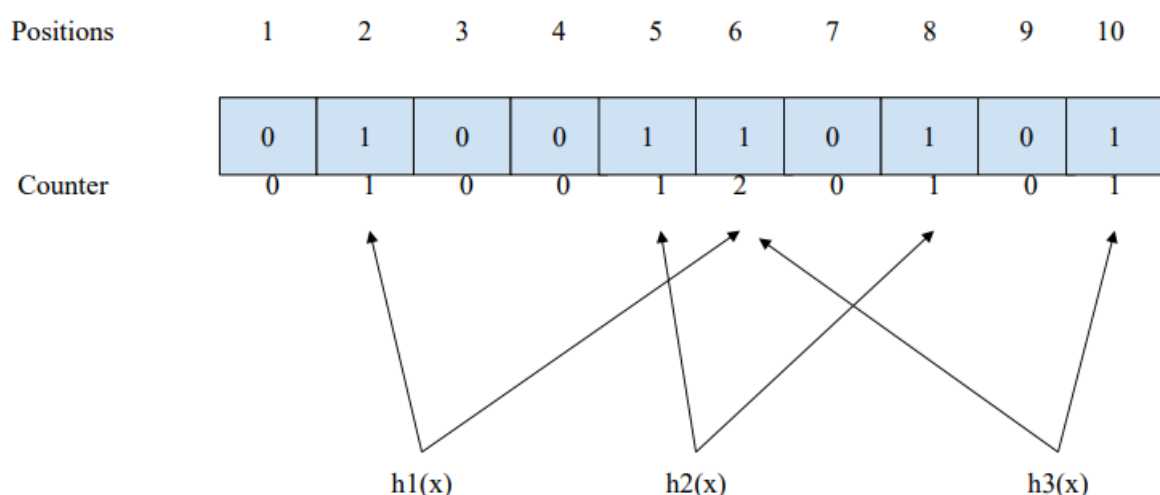
Fig: 1

- If any of the bits is 0, undoubtedly y is not a member of S .
- If all $h_i(x)$ are found to be 1, y may be in S .

- The bits may have by chance been set to 1 during the insertion of other elements. Thus, there is a chance of incorrect assumption. This is a false positive.

COUNTING BLOOM FILTER

- Process of deleting a particular element in the Bloom filter requires that the corresponding positions computed by k hash functions in the bit vector, be set to zero.
- For example, the bit position 6 is set to one by two hash functions $h_1(x)$ and $h_3(x)$ in figure 1 (a). Thus, it is not possible to delete an element stored in the filter.
- In order to perform deletion of the element, a method called **Counting Bloom filter**, a variant of Bloom filter can be used.
- The counting filter maintains a counter for each bit in the Bloom filter.
- The counters corresponding to the k hash values are incremented or decremented, whenever an element is added to the filter or removed from the filter respectively.
- As soon as a counter changes from 0 to 1, the corresponding bit in the bit vector is set to 1.
- When a counter changes from 1 to 0, the corresponding bit in the bit vector is set to 0.
- The counter basically maintains the number of elements that hashed to the corresponding bit by any of the k hash functions.
- The below figure illustrates the use of a Counting Bloom filter.



COUNTING DISTINCT ELEMENTS IN A STREAM

- The count-distinct problem relates to finding the number of dissimilar elements in a data stream.

- The stream of data contains repeated elements.
- This is a well-known problem in networking and databases.
- Several applications require finding the dissimilar or distinct elements.
- For example, packets passing through a router, unique visitors to a website, etc.

THE FLAJOLET-MARTIN ALGORITHM

- Flajolet-Martin algorithm estimates the m , number of distinct elements, in a stream or a database in one pass.
- The stream consisting of n elements with m unique elements runs in $O(n)$ time and needs $O(\log(m))$ memory.
- Thus, the space consumption calculates with the maximum number of possible distinct elements in the stream, which makes it innovative.
- The following are the features of Flajolet-Martin algorithm:
 1. Hash-based algorithm
 2. Needs several repetitions to get a good estimate
 3. The more different elements in the data, the more different hash values are obtained.
 4. Different hash values suggests the chances of one of these values will be unusual.

Problem

Find an estimation of the number of distinct elements in the following stream using FM algorithm:

$s = 2, 3, 1, 2, 3, 4, 3, 1, 2, 3, 1, 4$

Solution

Apply hash function on input stream				Binary Equivalent	Trailing Zeroes
$f(2)$	$7*2+2 \bmod 5$	$16 \bmod 5$	1	001	0
$f(3)$	$7*3+2 \bmod 5$	$23 \bmod 5$	3	011	0
$f(1)$	$7*1+2 \bmod 5$	$9 \bmod 5$	4	100	2
$f(2)$	$7*2+2 \bmod 5$	$16 \bmod 5$	1	001	0
$f(3)$	$7*3+2 \bmod 5$	$23 \bmod 5$	3	011	0
$f(4)$	$7*4+2 \bmod 5$	$30 \bmod 5$	0	000	0
$f(3)$	$7*3+2 \bmod 5$	$23 \bmod 5$	3	011	0
$f(1)$	$7*1+2 \bmod 5$	$9 \bmod 5$	4	100	2
$f(2)$	$7*2+2 \bmod 5$	$16 \bmod 5$	1	001	0
$f(3)$	$7*3+2 \bmod 5$	$23 \bmod 5$	3	011	0
$f(1)$	$7*1+2 \bmod 5$	$9 \bmod 5$	4	100	0
$f(4)$	$7*4+2 \bmod 5$	$30 \bmod 5$	0	000	0

Consider a hash function, say $f(a)=7s + 2 \bmod 5$.

Now, apply hash function on the input stream, perform the bit calculation and trailing zeroes to get:

- (i) The maximum number of trailing zeros from the binary equivalent trailing zero values, $r=2$
- (ii) The distinct value $R = 2^r = 2^2 = 4$
- (iii) Therefore, $R = 4$, means there are four (4) distinct values as 2,3,1,4.

ESTIMATING MOMENTS

- Assume a random variable X , where X refers to a variable, such as number of distinct elements x in a data stream.
- Assume that variable x has probabilistic distribution in values around the mean value x .
- Probabilistic distribution means probability of variable having value found = x varying with variable X .
- Expected value among the distributed X_i values where i varies from 0 to n will depend upon the expected distinct element count.
- Expected value will be m for expected number of distinct elements in the data stream, and much less than m for wide variance.
- The variance is the square of the standard deviation in m from the expected value, the second central moment of a distribution, and the σ^2 or $\text{var}(x)$ represents covariance of the random variable with itself.
- Moments (0,1,2 ...) refer to expected values to the powers of (0,1,2, ...) random-variable variance.

COUNTING 1'S IN A WINDOW

- The sliding window model for data stream algorithms is a popular model for infinite data stream processing.
- Window refers to the time interval during which stream raised and processed the queries.
- The receiving of data elements is taking place one by one.
- Statistical computations are over a sliding time-window of size N (not over the whole stream) in time-units.
- Window covers the most recent data items arrived & focuses on recent data and hence provides more significant and relevant data in real-world applications.
- The network traffic analysis requires analysis based on the recent past. This is more informative and useful than analysis based on stale data.
- A useful model of stream processing is the one in which queries are processed for a window of length N , where N corresponds to the most-recent elements received.
- Usually, it is so that N is very large and cannot be stored on a storage device, or there are so many streams that elements from windows for all cannot be stored.
- Let us consider a counting problem in which **we need to count the number of 1's present in the last k bits received** (where $k \leq N$, and N is the window length), in a given stream of 0's and 1's.
- The obvious solution is to store the most recent N bits.

- When a new bit comes in, discard the first bit. This will result in the exact answer.
- If there is not enough memory to store the N bits (assume N is 1 Billion), the solution can be obtained by using the **Datar-Gionis-Indyk-Motwani (DGIM) algorithm**.

Datar-Gionis-Indyk-Motwani (DGIM) algorithm

- Each bit that comes in the stream has a timestamp, same as the position of the bit in the stream.
 - First bit has a timestamp of 1, the second bit has a timestamp 2 and so on.
- Distinguish the positions within the window of length N.
- Take the window size as a multiple of 2.
- Represent the timestamp as $\log_2 N$.
- Divide the window into buckets, consisting of:
 - The timestamp of its right (most recent) end.
 - The number of 1's in the bucket. This number must be a power of 2.
 - The number of 1's is referred to as the size of the bucket.
- Example-1001011-the number of 1s is 4-Hence the bucket size is 4.

Rules that must be followed when representing a stream by buckets

- The right end of a bucket is always a position with a 1.
- Every bucket must contain one bit 1.
- No buckets can be formed without a bit 1.
- All sizes must be a power of 2.
- The size of the buckets must increase as we move on to the left.

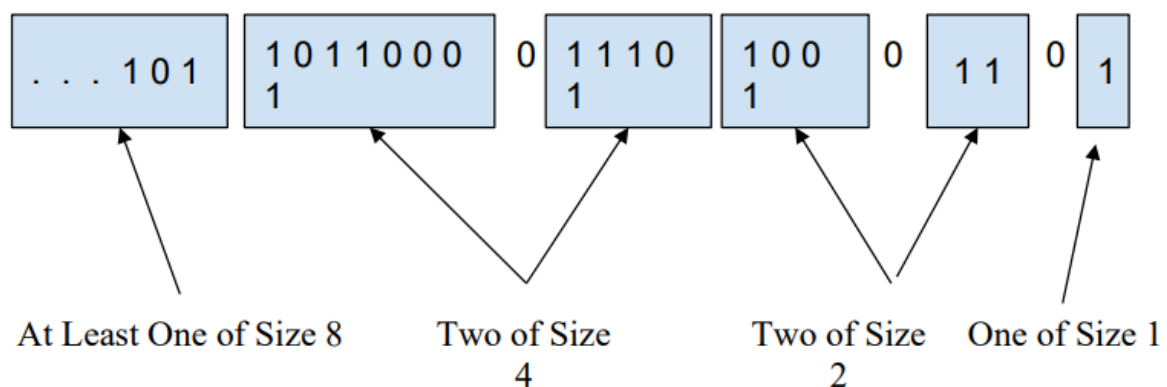


Figure 2-6: Dividing stream into buckets as per DGIM algorithm

- The time complexity of the DGIM algorithm is $O(\log N)$.

DECAYING WINDOW

- Decaying windows are useful in applications which need identification of most common elements.
- The use of the decaying window concept is when **more weight assigns to recent elements**.

- The technique computes a smooth aggregation of all the 1's ever seen in the stream, with **decaying weights**.
- When it further appears in the stream, less weight is given.

REAL TIME ANALYTICAL PLATFORM (RTAP)

- Real-time application relates to responsiveness. Data, when generated fast, needs fast processing as well.
- An application sometimes requires updating the information at the same rate at which it receives data.
- Late decisions sometimes cause loss of great opportunities.
- The term 'analytics' implies the identification of meaningful patterns from data.
- Thus, real-time analytics signifies finding meaningful patterns in data at the actual time of receiving it.
- Real-Time Analytics Platform (RTAP) manages and processes data and helps timely decision making.
- Following are some of the widely used RTAPs:
 - ✚ **Apache SparkStreaming** - a Big Data platform for data stream analytics in real time.
 - ✚ **Cisco Connected Streaming Analytics (CSA)** - a platform that delivers insights from high-velocity streams of live data from multiple sources and enables immediate action.
 - ✚ **Oracle Stream Analytics (OSA)** - a platform that provides a graphical interface to "Fast Data". Users can analyze streaming data as it arrives based on conditions and rules.
 - ✚ **SQLStreamBlaze** - an analytics platform offering a real-time, easy-to-use and powerful visual development environment for developers and analysts.
 - ✚ **IBM Stream Computing** - a data streaming tool that analyzes a broad range of streaming data - unstructured text, video, audio, geospatial data, and sensor data.

RTAP APPLICATIONS

- Fraud detection systems for online transactions
- Log analysis for understanding usage pattern
- Click analysis for online recommendations
- Push notifications to the customers for location-based advertisements for retail
- Action for emergency services such as fires and accidents in an industry
- Social media

REAL TIME SENTIMENT ANALYSIS USING TWEETS

- The case study provides the method of access of real-time social media information using Twitter. Tweets are received from the twitter stream, pre-processed and then analyzed to extract the features. The method includes the following steps:
 1. Collect a comprehensive training dataset that consists of data about potential users of a system.

2. Pull the specific tweets in real-time using Twitter API, then process and load this data into a persistent storage.
3. The cleaning of the data proceeds with punctuations, stop words, URLs, common emoticons, and hashtags, and reference deletion. Multiple consecutive letters in a word are reduced to two ('tooooooooo much' is replaced with 'too much'). Spell checking is also performed to words that have been identified as misspelled in order to infer the correct word.
4. Classify various types of tweets and segment them after estimating the influence of each tweet using the predictive analysis library. Perform sentiment analysis by identifying whether people are tweeting positive or negative statements about some actions.
5. Use linguistic concepts, perform opinion mining, analyze data and bring out powerful insights. Use important features of the analysis based on machine learning. Thus, applications learn by analysing ever-increasing amounts of data.
6. The goal is to build the model for predicting the sentiments from tweets.

STOCK MARKET PREDICTIONS

- Stock market data is an example of a real-time data stream.
- Data stream algorithms compute the values over a time-window of stock trades.
- This window has fixed size and contains n stock trades.
- A sale-purchase is counted as one trade. The parameter studied is **Volume-Weighted Average Price (VWAP)**.
- Assume stock trade i has price P_i , with S_i shares changing hands, then compute the volume-weighted average (VWAP) stock prices from a stream of stock sales.

$$P_{VWAP} \text{ (Price for VWAP)} = \frac{\sum P_i S_i}{\sum S_i}$$