BDA QB Soln - Assignment with question and answer

Big Data Analytics (University of Mumbai)

1. Explain how R language is useful for data analysis?

Soln:

R is a programming language for statistical computing and graphics that you can use to clean, analyze, and graph your data. It is widely used by researchers from diverse disciplines to estimate and display results and by teachers of statistics and research methods

Data analysis has emerged as the most popular field of the 21st century. It is because there is a pressing need to analyze and construct insights from the data. Industries transform raw data into furnished data products. In order to do so, it requires several important tools to churn the raw data. R is one of the programming languages that provide an intensive environment for you to research, process, transform, and visualize information.

Some of the important features of R for data science application are:

R provides extensive support for statistical modelling.

R is a suitable tool for various data science applications because it provides aesthetic visualization tools.

R is heavily utilized in data science applications for ETL (Extract, Transform, Load). It provides an interface for many databases like SQL and even spreadsheets.

R also provides various important packages for data wrangling.

With R, data scientists can apply machine learning algorithms to gain insights about future events.

One of the important feature of R is to interface with NoSQL databases and analyze unstructured data.

Most common Data Science in R Libraries

Dplyr: For performing data wrangling and data analysis, we use the dplyr package. We use this package for facilitating various functions for the Data frame in R. Dplyr is actually built around these 5 functions. You can work with local data frames as well as with remote database tables. You might need to:

Select certain columns of data.

Filter your data to select specific rows.

Arrange the rows of your data into order.

Mutate your data frame to contain new columns.

Summarize chunks of your data in some way.

Ggplot2: R is most famous for its visualization library ggplot2. It provides an aesthetic set of graphics that are also interactive.The ggplot2 library implements a “grammar of graphics” (Wilkinson, 2005). This approach gives us a coherent way to produce visualizations by expressing relationships between the attributes of data and their graphical representation.

Esquisse: This package has brought the most important feature of Tableau to R. Just drag and drop, and get your visualization done in minutes. This is actually an

enhancement to ggplot2.It allows us to draw bar graphs, curves, scatter plots, histograms, then export the graph or retrieve the code generating the graph.
Tidyr: Tidyr is a package that we use for tidying or cleaning the data. We consider this data to be tidy when each variable represents a column and each row represents an observation.
Shiny: This is a very well known package in R. When you want to share your stuff with people around you and make it easier for them to know and explore it visually, you can use shiny. It's a Data Scientist's best friend.
Caret: Caret stands for classification and regression training. Using this function, you can model complex regression and classification problems.
E1071: This package has wide use for implementing clustering, Fourier Transform, Naive Bayes, SVM and other types of miscellaneous functions.
Mlr: This package is absolutely incredible in performing machine learning tasks. It almost has all the important and useful algorithms for performing machine learning tasks. It can also be termed as the extensible framework for classification, regression, clustering, multi-classification and survival analysis
Other worth mentioning R libraries:
Lubridate
Knitr
DT(DataTables)
RCrawler
Leaflet
Janitor
Plotly
Applications of R for Data Science
Top Companies that use R for Data Science:
Google: At Google, R is a popular choice for performing many analytical operations. The Google Flu Trends project makes use of R to analyze trends and patterns in searches associated with flu.
Facebook Facebook makes heavy use of R for social network analytics. It uses R for gaining insights about the behavior of the users and establishes relationships between them.
IBM: IBM is one of the major investors in R. It recently joined the R consortium. IBM also utilizes R for developing various analytical solutions. It has used R in IBM Watson – an open computing platform.
Uber: Uber makes use of the R package shiny for accessing its charting components. Shiny is an interactive web application that's built with R for embedding interactive visual graphics.

2. Explain DGIM algorithm for counting ones in a window.

Soln:

Suppose we have a window of length N on a binary stream. We want at all times to be able to answer queries of the form “how many 1’s are there in the last k bits?” for any k≤ N. For this purpose we use the DGIM algorithm.

The basic version of the algorithm 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%.

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.

Since we only need to distinguish positions within the window of length N, we shall represent timestamps modulo N, so they can be represented by log2 N bits. If we also store the total number of bits ever seen in the stream (i.e., the most recent timestamp) modulo N, then we can determine from a timestamp modulo N where in the current window the bit with that timestamp is.

We divide the window into buckets, 5 consisting of:

1. The timestamp of its right (most recent) end.
2. The number of 1’s in the bucket. This number must be a power of 2, and we refer to the number of 1’s as the size of the bucket.

To represent a bucket, we need log2 N bits to represent the timestamp (modulo N) of its right end. To represent the number of 1’s we only need log2 log2 N bits. The reason is that we know this number i is a power of 2, say 2j , so we can represent i by coding j in binary. Since j is at most log2 N, it requires log2 log2 N bits. Thus, O(logN) bits suffice to represent a bucket. There are six 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 position with a 1 is in some bucket.
- No position is in more than one bucket.
- There are one or two buckets of any given size, up to some maximum size.

- All sizes must be a power of 2.
- Buckets cannot decrease in size as we move to the left (back in time).

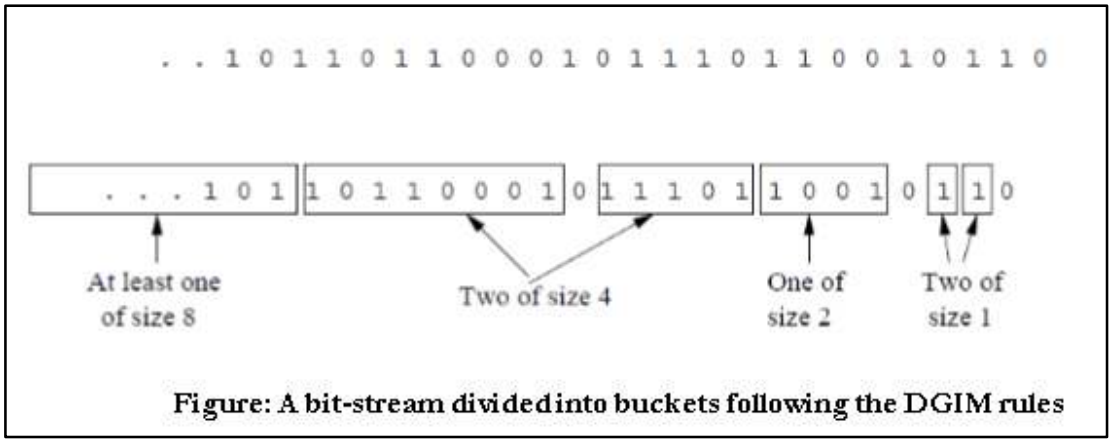


Figure: A bit-stream divided into buckets following the DGIM rules

3. What are different types of recommendation system explain in detail.

Soln:

RECOMMENDATION SYSTEM:

1. Recommendation system is widely used now-a-days.
2. It is sub class of information filtering system.
3. It is used to provide recommendation for games, movies, music, books, social tags, articles etc.
4. It is useful for experts, financial services, life insurance and social media based organization.
5. There are two types of recommendation systems:

a) Collaborative filtering.

b) Content based filtering.

COLLABORATIVE FILTERING SYSTEMS:

1. It uses community data from peer groups for recommendation.
2. These exhibits all those things that are popular among the peers.
3. It uses user's past behaviour and apply soǿe predication about user ǿay like and accordingly post

data.

4. These filtering systems recommend items based on similarity measure between

users and/or items.
5. Here user profile and contextual parameters along with the community data are used by the
recommender systems to personalize the recommendation list.
6. This is the most prominent approach in e-commerce website.
7. The basic assumption for collaborative filtering is:
a. User gives ratings to item in the catalog.
b. Customer who had similar taste in past will have similar taste in future.
c. Users who agreed in their subjective evaluations in the past will agree in the future too.
d. To find out similarity we can use Pearson's correlation co-efficient as:

$$sim(a,b) = \sum_p ep \frac{(ra,p-ra)(rb,p-rb)}{\sqrt{\sum_p ep(ra,p-ra)^2,(rb,p-rb)^2}}$$

Where A, b = users

Ra, p = rating of user 'a' for item 'p'

P = Set of items rated by both a and b

e. We can use this formula for prediction as :

$$pred(a,p) = ra + \frac{\sum_b ensim(a,b) \times (\eta b,p-rb)}{\sum_b ensim(a,b)}$$

Advantages:

1. Continuous learning for market process.
2. No knowledge engineering efforts needed.

Disadvantages:

1. Rating & feedback is required.
2. New items and users faces to cold start.

CONTENT BASED RECOMMENDATION:

1. A content based recommender works with data that the user provides, either explicitly (rating) or implicitly (clicking on a link).
2. Based on that data, a user profile is generated, which is then used to make suggestions to the user.
3. As the user provides more inputs or takes actions on the recommendations, the engine becomes more and more accurate.
4. Item profile in content based systems focuses on items and user profiles in form of weighted lists.
5. Profile are helpful to discover properties of items.
6. Consider the below examples:
   a. Some students prefer to be guided by few teachers only.
   b. Some Viewers prefer drama or movie by their favorite actors only.
   c. Few Viewers prefer old songs on other hand few viewers may prefer new songs only depending upon users sorting of songs based on year.
7. In general, there are so many classes which provides such data.
8. Few domains has common features for example a college and movie it has students, professors set and actors, director set respectively.
9. Certain ratio is maintained in such cases like every college and movie has year wise datasets as movie released in a year by director and actor and college has passing student every year etc.
10. Music song album and book has same value feature like songs writers/poet, year of release and publication year etc.

11. Consider the figure 6.6 which shows recommendation system parameters.

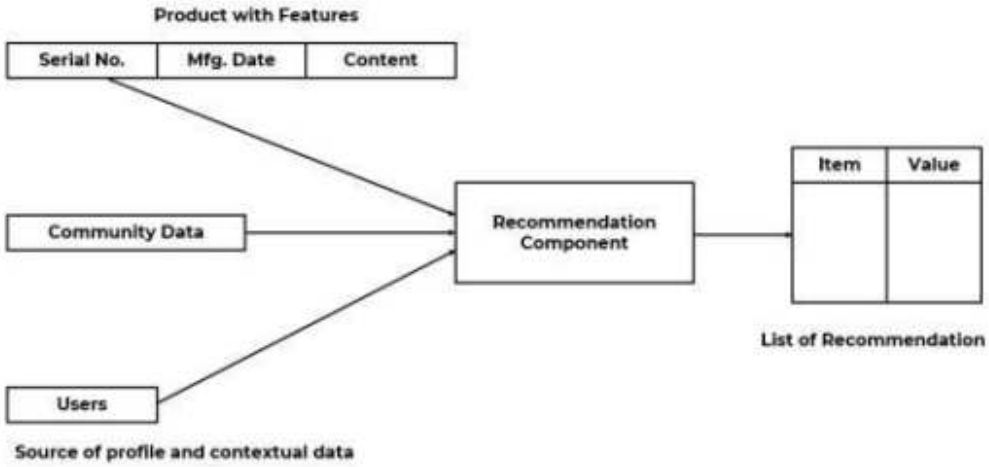


Figure 6.6: Recommendation system parameters

4. Explain recommendations based on User Ratings using appropriate examples.

Soln:

- Input
  - Only a matrix of given user–item ratings
- Output types
  - A (numerical) prediction indicating to what degree the current user will like or dislike a certain item
  - A top-N list of recommended items

USER RATING MATRIX

	Amy	Jef	Mike	Chris	Ken
The Piano	–	–	+		+
Pulp Fiction	–	+	+	–	+
Clueless	+		–	+	–
Cliffhanger	–	–	+	–	+
Fargo	–	+	+	–	?

SIMILARITY METRIC

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

- **Intuitively we want: sim(*A*, *B*) > sim(*A*, *C*)**
- **Jaccard similarity:** loses important information
  - Sim(A,B)1/5 < Sim(A,C) 2/4
  - Ignore rating values

sim A,B vs. A,C:
0.092 > -0.559

Notice cosine sim. is correlation when data is centered at 0

Cosine similarity: 0.386 > 0.322

- Considers missing ratings as "negative"
- **Solution: Centered Cosine**
- **subtract the (row) mean (10/3)**

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	2/3			5/3	−7/3		
B	1/3	1/3	−2/3				
C				−5/3	1/3	4/3	
D		0					0

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	2/3			5/3	−7/3		
B	1/3	1/3	−2/3				
C				−5/3	1/3	4/3	
D		0					0

- sim(A, B) = cos(r_A, r_B) = 0.09
- Sim(A,C) = -0.56

- Missing ratings treated as "average"
- Handles "tough raters" and "easy raters"
- Pearson Correlation

r_x = [*, _, _, *, ***]
r_y = [*, _, **, **, _]

Finding "Similar" Users

- Let r_x be the vector of user x's ratings
- **Jaccard similarity measure**
  - **Problem:** Ignores the value of the rating
- **Cosine similarity measure**
  - $\text{sim}(x, y) = \cos(r_x, r_y) = \frac{r_x \cdot r_y}{\|r_x\| \cdot \|r_y\|}$
  - **Problem:** Treats missing ratings as "negative"
- **Pearson correlation coefficient**
  - S_{xy} = items rated by both users x and y

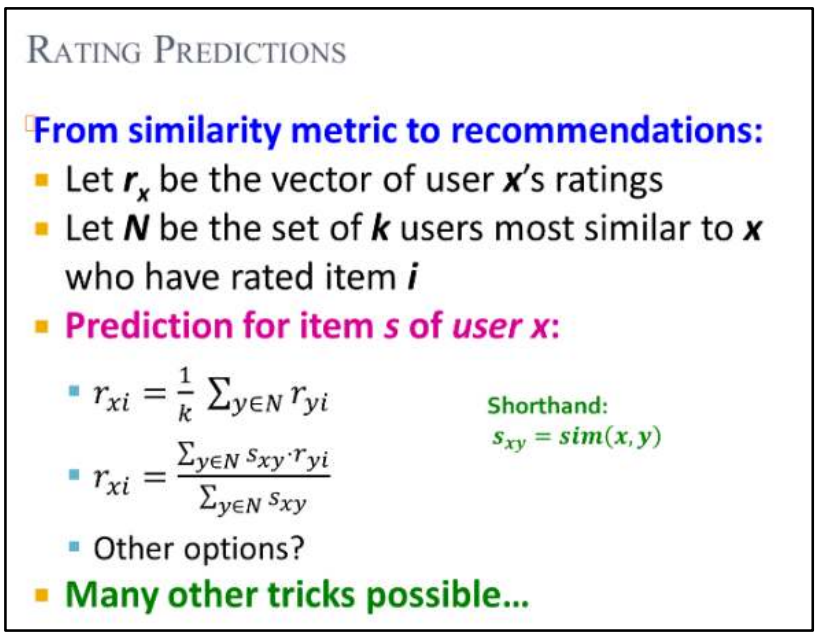
$$sim(x,y) = \frac{\sum_{s \in S_{xy}} (r_{xs} - \overline{r_x})(r_{ys} - \overline{r_y})}{\sqrt{\sum_{s \in S_{xy}} (r_{xs} - \overline{r_x})^2} \sqrt{\sum_{s \in S_{xy}} (r_{ys} - \overline{r_y})^2}}$$

r_x, r_y as sets:
r_x = {1, 4, 5}
r_y = {1, 3, 4}

r_x, r_y as points:
r_x = {1, 0, 0, 1, 3}
r_y = {1, 0, 2, 2, 0}

34

$\overline{r_x}$, $\overline{r_y}$ … avg. rating of x, y



5. With a neat sketch, explain the architecture of data-stream management system.

Soln:

In analogy to a database-management system, we can view a stream processor as a kind of data-management system, the high-level organization of which is suggested in Fig.

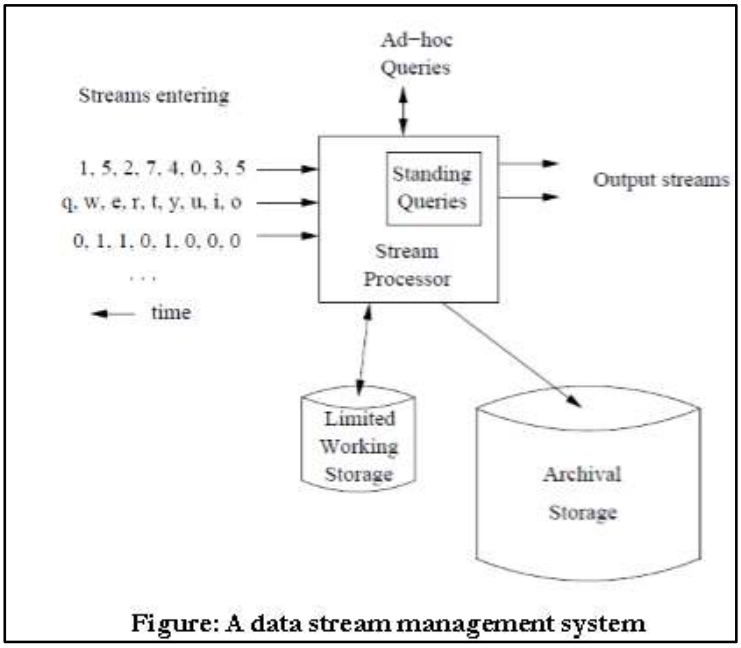


Figure: A data stream management system

Any number of streams can enter the system. Each stream can provide elements at its own schedule; they need not have the same data rates or data types, and the time between elements of one stream need not be uniform. The fact that the rate of arrival of stream elements is not under the control of the system distinguishes stream processing from the processing of data that goes on within a database-management system. The latter system controls the rate at which data is read from the disk, and therefore never has to worry about data getting lost as it attempts to execute queries. Streams may be archived in a large archival store, but we assume it is not possible to answer queries from the archival store. It could be examined only under special circumstances using time-consuming retrieval processes. There is also a working store, into which summaries or parts of streams may be placed, and which can be used for answering queries. The working store might be disk, or it might be main memory, depending on how fast we need to process queries. But either way, it is of sufficiently

limited capacity that it cannot store all the data from all the streams.

6. How are dead-ends handled in PageRank calculation?

Soln:

A dead end is a Web Page with no links out. The presence of dead ends will cause the Page Rank of some or all the pages to go to 0 in the iterative computation, including pages that are not dead ends.

Dead ends can be eliminated before undertaking a Page Rank calculation by recursively dropping nodes with no arcs out. Note that dropping one node can cause another which linked only to it to become a dead end, so the process must be recursive.

Two approaches to deal with dead ends:

1. We can drop the dead ends from the graph, and also drop their incoming arcs. Doing so may create more dead ends which also have to be dropped recursively. However eventually, we wind up with a strongly-connected component (SCC) none of whose nodes are dead ends. Recursive deletion of dead ends will remove parts of the out-components, tendrils, and tubes but leave the SCC and the in-component, as well as parts of any small isolated components.
2. We can modify the process by which random surfers are assumed to move about the Web. This method which we refer to as 'taxation' also solves the problem of spider traps. Here, we modify the calculation of Page Rank by allowing each random suffer a small probability of teleporting to a random page, rather than following an out-link from their current page, The iterative step where we compute a new vector estimate

of Page Rank v' from the current Page Rank estimate v and the transition matrix 'M' is,

$$v' = \beta M v + \frac{(1-\beta)e}{n}$$

where β is the chosen constant usually in the range of 0.8 to 0.9,
e is a vector of all 1's with the appropriate number of components,
n is the number of nodes in the Web graph.

The term (1-β)e/n does not depend on the sum of the components of the vector v, there will always be some fraction of a surfer operating on the Web. That is, when there are dead ends, the sum of the components of v, may be less than 1, but it will never reach 0.

7. Explain Flajolet-Martin algorithm.

Soln:

Flajolet-Martin algorithm approximates the number of unique objects in a stream or a database in one pass. If the stream contains n elements with m of them unique, this algorithm runs in

O(n)

O(n) time and needs

O(log(m))

O(log(m)) memory.

Algorithm:

1. Create a bit vector (bit array) of sufficient length L, such that 2L>n, the number of elements in the stream. Usually a 64-bit vector is sufficient since 264 is quite large for most purposes.
2. The i-th bit in this vector/array represents whether we have seen a hash function value whose binary representation ends in 0i. So initialize each bit to 0.
3. The i-th bit in this vector/array represents whether we have seen a hash function value whose binary representation ends in 0i. So initialize each bit to 0.
4. The i-th bit in this vector/array represents whether we have seen a hash function value whose binary representation ends in 0i. So initialize each bit to 0.
5. Once input is exhausted, get the index of the first 0 in the bit array (call this R). By the way, this is just the number of consecutive 1s (i.e. we have seen 0,00,...,0R−1 as the output of the hash function) plus one.
6. Calculate the number of unique words as 2R/ϕ, where ϕ is 0.77351. A proof for this can be found in the original paper listed in the reference section.
7. The standard deviation of R is a constant: σ(R)=1.12. (In other words, R can be off by about 1 for **1 - 0.68 = 32%** of the observations, off by 2 for about **1 - 0.95 = 5%** of the observations, off by 3 for **1 - 0.997 = 0.3%** of the observations using the Empirical rule of statistics). This implies that our count can be off by a factor of 2 for 32% of the observations, off by a factory of 4 for 5% of the observations, off by a factor of 8 for 0.3% of the observations and so on.

Example:

S=1,3,2,1,2,3,4,3,1,2,3,1

S=1,3,2,1,2,3,4,3,1,2,3,1

h(x)=(6x+1) mod 5

h(x)=(6x+1) mod 5

Assume |b| = 5

x	h(x)	Rem	Binary	r(a)
1	7	2	00010	1
3	19	4	00100	2
2	13	3	00011	0
1	7	2	00010	1
2	13	3	00011	0
3	19	4	00100	2
4	25	0	00000	5
3	19	4	00100	2
1	7	2	00010	1
2	13	3	00011	0
3	19	4	00100	2
1	7	2	00010	1

R = max(r(a)) = 5

So no. of distinct elements = $N = 2^R = 2^5 = 32$

8. Describe why traditional clustering is not suitable for community detection.

Soln:

Clustering Group sets of points based on their features

Community detection Group sets of points based on their connectivity

Community detection in networks

• A simplestrategy:

– Choose a suitable distance measure based on available data

• E.g. Path lengths;

Distance based on inverse
Tie strengths; size of largest enclosing group or common attribute;
Distance in a spectral (eigenvector) embedding; etc..
– Apply a standard clustering algorithm

Clustering is not always suitable in networks because:
• Small world networks have small diameter – And sometime integer distances – A distance-based method does not have a lot of option to represent similarities/dissimilarities
• High degree nodes are common – Connect different communities – Hard to separate communities
• Edge densities vary across the network – Same threshold does not work well everywhere

9. What is page rank? How to calculate the page rank of a web graph?

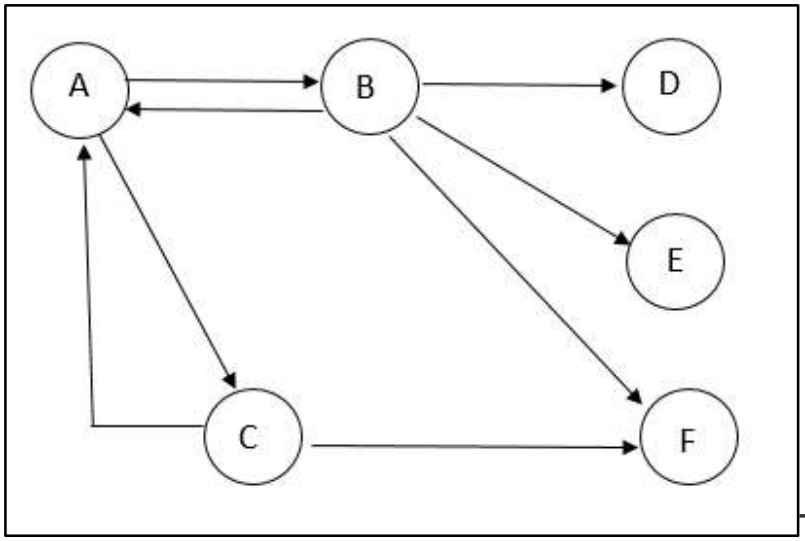
Soln:
PageRank (PR) is an algorithm used by Google Search to rank websites in their search engine results. PageRank was named after Larry Page, one of the founders of Google. PageRank is a way of measuring the importance of website pages. According to Google:

PageRank works by counting the number and quality of links to a page to determine a rough estimate of how important the website is. The underlying assumption is that more important websites are likely to receive more links from other websites.

It is not the only algorithm used by Google to order search engine results, but it is the first algorithm that was used by the company, and it is the best-known.
The above centrality measure is not implemented for multi-graphs.

Algorithm
The PageRank algorithm outputs a probability distribution used to represent the likelihood that a person randomly clicking on links will arrive at any particular page. PageRank can be calculated for collections of documents of any size. It is assumed in several research papers that the distribution is evenly divided among all documents in the collection at the beginning of the computational process. The PageRank computations require several passes, called “iterations”, through the collection to adjust approximate PageRank values to more closely reflect the theoretical true value.

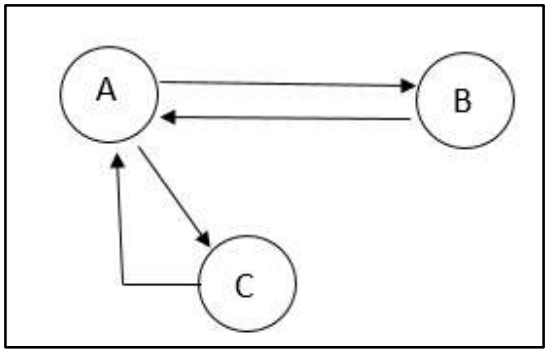
Calculating the pagerank:



Transition Matrix

$$M = \begin{bmatrix} 0 & \frac{1}{4} & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{4} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{4} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{4} & \frac{1}{2} & 0 & 0 & 0 \end{bmatrix}$$

- From the graph it is clear that D, E and F are dead ends.
- Before applying page rank algorithm remove dead ends.
- The new graph will be:



so new transition matrix will be

$$M = \begin{bmatrix} 0 & ½ & 1 \\ ½ & 0 & 0 \\ ½ & 0 & 0 \end{bmatrix}$$

β=0.85

β=0.85 and initial page rank vector

$$v = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Iteration 1:

V=βmv+(1−β)e/n

V=βmv+(1−β)e/n

$$= \frac{17}{20} \begin{bmatrix} 0 & ½ & 1 \\ ½ & 0 & 0 \\ ½ & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} + 3/20 \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}$$

$$= \frac{17}{20} \begin{bmatrix} 2 \\ ½ \\ ½ \end{bmatrix} + \begin{bmatrix} 1/20 \\ 1/20 \\ 1/20 \end{bmatrix}$$

$$= \begin{bmatrix} 34/20 \\ 17/40 \\ 17/40 \end{bmatrix} + \begin{bmatrix} 1/20 \\ 1/20 \\ 1/20 \end{bmatrix} = \begin{bmatrix} 35/20 \\ 19/40 \\ 19/40 \end{bmatrix}$$

Iteration 2

$V = \beta mv + (1 - \beta)e/n$

After Iteration 2 page rank of

Page A = 343/400

B = 127/160

C = 127/160

D = $\frac{PR(B)}{2} = \frac{127/160}{2} = \frac{127}{320}$

E = $\frac{PR(B)}{2} = \frac{127}{320}$

F = $\frac{PR(B)}{2} + \frac{PR(B)}{2} = \frac{127}{320} + \frac{127}{320}$

= $\frac{254}{320}$

10. Explain with the example the types of queries fired on stream data.

Soln:

Types of queries one wants on answer on a data stream

- **Sampling data from a stream**
  - Construct a random sample
- **Queries over sliding windows**
  - Number of items of type ***x*** in the last ***k*** elements of the stream
- **Filtering a data stream**
  - Select elements with property ***x*** from the stream
- **Counting distinct elements**
  - Number of distinct elements in the last ***k*** elements of the stream
- **Estimating moments**
  - Estimate avg./std. dev. of last ***k*** elements
- **Finding frequent elements**

APPLICATIONS (1)

- **Mining query streams**
  - Google wants to know what queries are more frequent today than yesterday
- **Mining click streams**
  - Yahoo wants to know which of its pages are getting an unusual number of hits in the past hour
- **Mining social network news feeds**
  - E.g., look for trending topics on Twitter, Facebook

APPLICATIONS (2)

- **Sensor Networks**
  - Many sensors feeding into a central controller
- **Telephone call records**
  - Data feeds into customer bills as well as settlements between telephone companies
- **IP packets monitored at a switch**
  - Gather information for optimal routing
  - Detect denial-of-service attacks

11. Explain Bloom's filter with example.

Soln:

A Bloom filter is a space-efficient probabilistic data structure that is used to test whether an element is a member of a set. For example, checking availability of username is set membership problem, where the set is the list of all registered username. The price we pay for efficiency is that it is probabilistic in nature that means, there might be some False Positive results. False positive means, it might tell that given username is already taken but actually it's not.

Interesting Properties of Bloom Filters

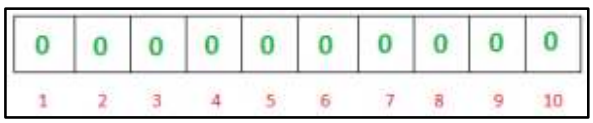
Unlike a standard hash table, a Bloom filter of a fixed size can represent a set with an arbitrarily large number of elements.

Adding an element never fails. However, the false positive rate increases steadily as elements are added until all bits in the filter are set to 1, at which point all queries yield a positive result.

Bloom filters never generate false negative result, i.e., telling you that a username doesn't exist when it actually exists.

Deleting elements from filter is not possible because, if we delete a single element by clearing bits at indices generated by k hash functions, it might cause deletion of few other elements. Example – if we delete "geeks" (in given example below) by clearing bit at 1, 4 and 7, we might end up deleting "nerd" also Because bit at index 4 becomes 0 and bloom filter claims that "nerd" is not present.

Working of Bloom Filter



A empty bloom filter is a bit array of m bits, all set to zero, like this –

empty_bit_array

We need k number of hash functions to calculate the hashes for a given input. When we want to add an item in the filter, the bits at k indices h1(x), h2(x), … hk(x) are set, where indices are calculated using hash functions.

Example – Suppose we want to enter “geeks” in the filter, we are using 3 hash functions and a bit array of length 10, all set to 0 initially. First we’ll calculate the hashes as follows:

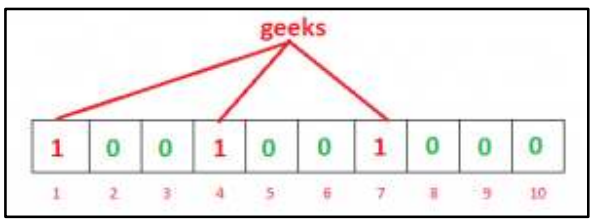
h1(“geeks”) % 10 = 1

h2(“geeks”) % 10 = 4

h3(“geeks”) % 10 = 7

Note: These outputs are random for explanation only.

Now we will set the bits at indices 1, 4 and 7 to 1



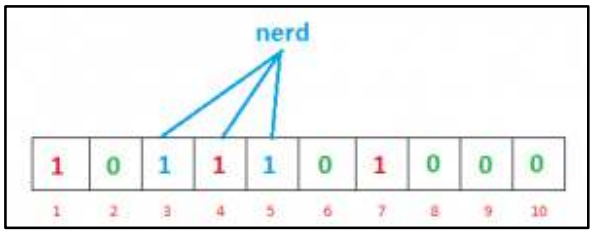
Again we want to enter “nerd”, similarly, we’ll calculate hashes

h1(“nerd”) % 10 = 3

h2(“nerd”) % 10 = 5

h3(“nerd”) % 10 = 4

Set the bits at indices 3, 5 and 4 to 1



Now if we want to check “geeks” is present in filter or not. We’ll do the same process but this time in reverse order. We calculate respective hashes using h1, h2 and h3 and check if all these indices are set to 1 in the bit array. If all the bits are set then we can say that “geeks” is probably present. If any of the bit at these indices are 0 then “geeks” is definitely not present.

False Positive in Bloom Filters

The question is why we said “probably present”, why this uncertainty. Let’s understand this with an example. Suppose we want to check whether “cat” is present or not. We’ll calculate hashes using h1, h2 and h3

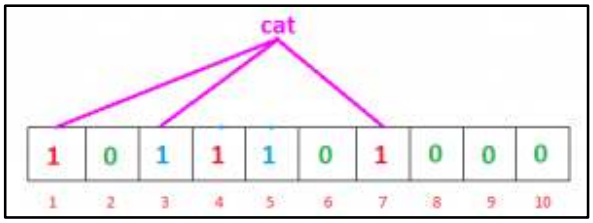
h1(“cat”) % 10 = 1

h2(“cat”) % 10 = 3

h3(“cat”) % 10 = 7

If we check the bit array, bits at these indices are set to 1 but we know that “cat” was never added to the filter. Bit at index 1 and 7 was set when we added “geeks” and bit 3

was set we added “nerd”.



So, because bits at calculated indices are already set by some other item, bloom filter erroneously claims that “cat” is present and generating a false positive result. Depending on the application, it could be huge downside or relatively okay.

We can control the probability of getting a false positive by controlling the size of the Bloom filter. More space means fewer false positives. If we want to decrease probability of false positive result, we have to use more number of hash functions and larger bit array. This would add latency in addition to the item and checking membership.

Operations that a Bloom Filter supports

insert(x) : To insert an element in the Bloom Filter.

lookup(x) : to check whether an element is already present in Bloom Filter with a positive false probability.

NOTE : We cannot delete an element in Bloom Filter.

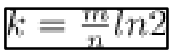
Probability of False positivity: Let m be the size of bit array, k be the number of hash functions and n be the number of expected elements to be inserted in the filter, then the probability of false positive p can be calculated as:

$$P = \left(1 - \left[1 - \frac{1}{m}\right]^{kn}\right)^k$$

Size of Bit Array: If expected number of elements n is known and desired false positive probability is p then the size of bit array m can be calculated as :

$$m = -\frac{n \ln P}{(ln2)^2}$$

Optimum number of hash functions: The number of hash functions k must be a positive integer. If m is size of bit array and n is number of elements to be inserted, then k can be calculated as :



Space Efficiency

If we want to store large list of items in a set for purpose of set membership, we can store it in hashmap, tries or simple array or linked list. All these methods require storing item itself, which is not very memory efficient. For example, if we want to store “geeks” in hashmap we have to store actual string “ geeks” as a key value pair {some_key : ”geeks”}.

Bloom filters do not store the data item at all. As we have seen they use bit array which allow hash collision. Without hash collision, it would not be compact.

Choice of Hash Function

The hash function used in bloom filters should be independent and uniformly distributed. They should be fast as possible. Fast simple non cryptographic hashes which are independent enough include murmur, FNV series of hash functions and Jenkins hashes.

Generating hash is major operation in bloom filters. Cryptographic hash functions provide stability and guarantee but are expensive in calculation. With increase in number of hash functions k, bloom filter become slow. All though non-cryptographic hash functions do not provide guarantee but provide major performance improvement.

12. Explain a social network graph clustering algorithm with example.

Soln:

Social Network:

When we think of a social network, we think of Facebook, Twitter, Google+, or another website that is called a "social network," and indeed this kind of network is representative of the broader class of networks called "social." The essential characteristics of a social network are:

1. There is a collection of entities that participate in the network. Typically, these entities are people, but they could be something else entirely
2. There is at least one relationship between entities of the network. On Facebook or its ilk, this relationship is called friends. Sometimes the relationship is all-or-nothing; two people are either friends or they are not.
3. There is an assumption of non-randomness or locality. This condition is the hardest to formalize, but the intuition is that relationships tend to cluster. That is, if entity A is related to both B and C, then there is a higher probability than average that B and C are related.

Social network as Graphs: Social networks are naturally modeled as graphs, which we sometimes refer to as a social graph. The entities are the nodes, and an edge connects two nodes if the nodes are related by the relationship that characterizes the network. If there is a degree associated with the relationship, this degree is represented by labeling the edges. Often, social graphs are undirected, as for the Facebook friends graph. But they can be directed graphs, as for example the graphs of followers on Twitter or Google+.

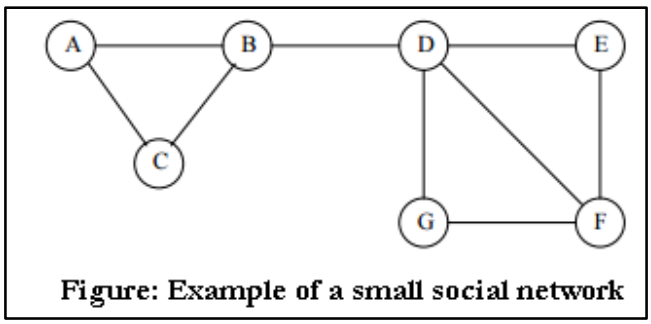


Figure: Example of a small social network

Above figure is an example of a tiny social network. The entities are the nodes A through G. The relationship, which we might think of as "friends," is represented by the edges. For instance, B is friends with A, C, and D.

Clustering of Social-Network Graphs:

Clustering of the graph is considered as a way to identify communities. Clustering of graphs involves following steps:

1. Distance Measures for Social-Network Graphs

If we were to apply standard clustering techniques to a social-network graph, our first

step would be to define a distance measure. When the edges of the graph have labels, these labels might be usable as a distance measure, depending on what they represented. But when the edges are unlabeled, as in a “friends” graph, there is not much we can do to define a suitable distance.

Our first instinct is to assume that nodes are close if they have an edge between them and distant if not. Thus, we could say that the distance d(x, y) is 0 if there is an edge (x, y) and 1 if there is no such edge. We could use any other two values, such as 1 and ∞, as long as the distance is closer when there is an edge.

2. Applying Standard Clustering Methods

There are two general approaches to clustering: hierarchical (agglomerative) and point-assignment. Let us consider how each of these would work on a social-network graph.

Hierarchical clustering of a social-network graph starts by combining some two nodes that are connected by an edge. Successively, edges that are not between two nodes of the same cluster would be chosen randomly to combine the clusters to which their two nodes belong. The choices would be random, because all distances represented by an edge are the same.

Now, consider a point-assignment approach to clustering social networks. Again, the fact that all edges are at the same distance will introduce a number of random factors that will lead to some nodes being assigned to the wrong cluster.

3. Betweenness:

Since there are problems with standard clustering methods, several specialized clustering techniques have been developed to find communities in social networks. The simplest one is based on finding the edges that are least likely to be inside the community.

Define the betweenness of an edge (a, b) to be the number of pairs of nodes x and y such that the edge (a, b) lies on the shortest path between x and y. To be more precise, since there can be several shortest paths between x and y, edge (a, b) is credited with the fraction of those shortest paths that include the edge (a, b). As in golf, a high score is bad. It suggests that the edge (a, b) runs between two different communities; that is, a and b do not belong to the same community

4. The Girvan-Newman Algorithm:

In order to exploit the betweenness of edges, we need to calculate the number of shortest paths going through each edge. We shall describe a method called the Girvan-

Newman (GN) Algorithm, which visits each node X once and computes the number of shortest paths from X to each of the other nodes that go through each of the edges. The algorithm begins by performing a breadth-first search (BFS) of the graph, starting at the node X. Note that the level of each node in the BFS presentation is the length of the shortest path from X to that node. Thus, the edges that go between nodes at the same level can never be part of a shortest path from X.

Edges between levels are called DAG edges ("DAG" stands for directed, acyclic graph). Each DAG edge will be part of at least one shortest path from root X. If there is a DAG edge (Y, Z), where Y is at the level above Z (i.e., closer to the root), then we shall call Y a parent of Z and Z a child of Y, although parents are not necessarily unique in a DAG as they would be in a tree.

5. Using betweenness to find communities:

The betweenness scores for the edges of a graph behave something like a distance measure on the nodes of the graph. It is not exactly a distance measure, because it is not defined for pairs of nodes that are unconnected by an edge, and might not satisfy the triangle inequality even when defined. However, we can cluster by taking the edges in order of increasing betweenness and add them to the graph one at a time. At each step, the connected components of the graph form some clusters. The higher the betweenness we allow, the more edges we get, and the larger the clusters become.

More commonly, this idea is expressed as a process of edge removal. Start with the graph and all its edges; then remove edges with the highest betweenness, until the graph has broken into a suitable number of connected components.

13. List different issues and challenges in stream data processing.

Soln:

Query processing in the data stream model of computation comes with its own unique challenges.

a) Unbounded Memory Requirements:

Since data streams are potentially unbounded in size, the amount of storage required to compute an exact answer to a data stream query may also grow without bound. While external memory algorithms for handling data sets larger than main memory have been studied, such algorithms are not well suited to data stream applications since they do not support continuous queries and are typically too slow for real-time response. New data is constantly arriving even as the old data is being processed; the amount of computation time per data element must be low, or else the latency of the computation will be too high and the algorithm will not be able to keep pace with the data stream.

b) Approximate Query Answering:

When we are limited to a bounded amount of memory it is not always possible to produce exact answers for data stream queries; however, high-quality approximate answers are often acceptable in lieu of exact answers. Sliding Window: One technique for producing an approximate answer to a data stream query is to evaluate the query not over the entire past history of the data streams, but rather only over sliding windows of recent data from the streams. For example, only data from the last week could be considered in producing query answers, with data older than one week being discarded.

c) Blocking Operators:

A blocking query operator is a query operator that is unable to produce the first tuple of its output until it has seen its entire input. If one thinks about evaluating continuous stream queries using a traditional tree of query operators, where data streams enter at the leaves and final query answers are produced at the root, then the incorporation of blocking operators into the query tree poses problems. Since continuous data streams may be infinite, a blocking operator that has a data stream as one of its inputs will never see its entire input, and therefore it will never be able to produce any output. Doing away with blocking operators altogether would be problematic, but dealing with them effectively is one of the more challenging aspects of data stream computation.

d) Queries Referencing Past Data:

In the data stream model of computation, once a data element has been streamed by, it cannot be revisited. This limitation means that ad hoc queries that are issued after some data has already been discarded may be impossible to answer accurately. One simple solution to this problem is to stipulate that ad hoc queries are only allowed to reference future data: they are evaluated as though the data streams began at the point when the query was issued, and any past stream elements are ignored (for the purposes of that query). While this solution may not appear very satisfying, it may turn out to be perfectly acceptable for many applications.

14. Write a script to create a dataframe. Illustrate use of pipe operator to perform operations on this data frame.

Soln:

Script to create a dataframe:

Name = c("Amiya", "Raj", "Asish")

```
Language = c("R", "Python", "Java")
Age = c(22, 25, 45)

df = data.frame(Name, Language, Age)
 print(df)
```

Output:

```
  Name  Language  Age
1 Amiya        R  22
2   Raj   Python  25
3 Asish     Java  45
```

OR

```
df = data.frame(
  "Name" = c("Amiya", "Raj", "Asish", “Omkar”, “Jay”),
  "Language" = c("R", "Python", "Java", “C”, “C++”),
  "Age" = c(22, 25, 45, 30, 28)
)
print(df)

cat("Accessing first and second row\n")
print(df[1:2, ])
```

OUTPUT:

```
   Name Language Age
1 Amiya        R  22
2   Raj   Python  25
3 Asish     Java  45
4 Omkar  C       30
5   Jay     C++   28

Accessing first and second row
   Name Language Age
1 Amiya        R  22
2   Raj   Python  25
```

Use of pipe operator:

The pipe operator, written as %>% , has been a longstanding feature of the magrittr package for R. It takes the output of one function and passes it into another function as an argument. This allows us to link a sequence of analysis steps

We will apply pipe operator on above defined dataframe df

```
df %>%  select(-Language)
  Name     Age
1 Amiya     22
2   Raj     25
3 Asish     45
4 Omkar    30
5   Jay     28

df %>%  select(-Language) %>% arrange(desc(Age))


  Name     Age
1 Asish     45
2 Omkar    30
3 Jay       28
4 Raj       25
5 Amiya     22


df %>%  select(-Language) %>% arrange(desc(Age)) %>% filter(Age > 25)

  Name     Age
1 Asish     45
2 Omkar    30
3 Jay       28
```

15. How distinct elements are identified in stream. Explain algorithm with example.

Soln:

Same as Q.7 Flajolet Martin

16. List and explain different data visualization techniques using R with help of dataframe.

Soln:
https://www.geeksforgeeks.org/data-visualization-in-r/

17. Explain built in functions for handling numeric and string data in R.

Soln:
https://www.javatpoint.com/r-built-in-functions

18. How to declare a function in R programming? Explain with example.

Soln:

```
fib <- function(n){

 fib <- c(1,1)

 for (i in 2:n-1)
  {
   fib <- c(fib,fib[i - 1] + fib[i])
  }
 print(fib)
 }
fib(5)
```

OUTPUT:
1 1 2 3 5

19. List and explain data types in R programming. How different types of operators are used to perform computations?

Soln:
https://www.guru99.com/r-data-types-operator.html

20. List and explain applications of data visualization.

Soln:

Healthcare Industries
A dashboard that visualises a patient's history might aid a current or new doctor in comprehending a patient's health. It might give faster care facilities based on illness in the event of an emergency. Instead than sifting through hundreds of pages of information, data visualisation may assist in finding trends.
Health care is a time-consuming procedure, and the majority of it is spent evaluating

prior reports. By boosting response time, data visualisation provides a superior selling point. It gives matrices that make analysis easier, resulting in a faster reaction time.(From)

Business intelligence

When compared to local options, cloud connection can provide the cost-effective “heavy lifting” of processor-intensive analytics, allowing users to see bigger volumes of data from numerous sources to help speed up decision-making.

Because such systems can be diverse, comprised of multiple components, and may use their own data storage and interfaces for access to stored data, additional integrated tools, such as those geared toward business intelligence (BI), help provide a cohesive view of an organization's entire data system (e.g., web services, databases, historians, etc.).

Multiple datasets can be correlated using analytics/BI tools, which allow for searches using a common set of filters and/or parameters. The acquired data may then be displayed in a standardised manner using these technologies, giving logical "shaping" and better comparison grounds for end users.

Military

It's a matter of life and death for the military; having clarity of actionable data is critical, and taking the appropriate action requires having clarity of data to pull out actionable insights.

The adversary is present in the field today, as well as posing a danger through digital warfare and cybersecurity. It is critical to collect data from a variety of sources, both organised and unstructured. The volume of data is enormous, and data visualisation technologies are essential for rapid delivery of accurate information in the most condensed form feasible. A greater grasp of past data allows for more accurate forecasting.

Dynamic Data Visualization aids in a better knowledge of geography and climate, resulting in a more effective approach. The cost of military equipment and tools is extremely significant; with bar and pie charts, analysing current inventories and making purchases as needed is simple.

Finance Industries

For exploring/explaining data of linked customers, understanding consumer behaviour, having a clear flow of information, the efficiency of decision making, and so on, data visualisation tools are becoming a requirement for financial sectors.

For associated organisations and businesses, data visualisation aids in the creation of patterns, which aids in better investment strategy. For improved business prospects, data visualisation emphasises the most recent trends.

Data science

Data scientists generally create visualisations for their personal use or to communicate information to a small group of people. Visualization libraries for the specified programming languages and tools are used to create the visual representations.

Open source programming languages, such as Python, and proprietary tools built for complicated data analysis are commonly used by data scientists and academics. These data scientists and researchers use data visualisation to better comprehend data sets and spot patterns and trends that might otherwise go undiscovered.

Marketing

In marketing analytics, data visualisation is a boon. We may use visuals and reports to analyse various patterns and trends analysis, such as sales analysis, market research analysis, customer analysis, defect analysis, cost analysis, and forecasting. These studies serve as a foundation for marketing and sales.

Visual aids can assist your audience grasp your main message by visually engaging them and visually engaging them. The major advantage of visualising data is that it can communicate a point faster than a boring spreadsheet.

In b2b firms, data-driven yearly reports and presentations don't fulfil the needs of people who are seeing the information. They are unable to grasp the art of engaging with their audience in a meaningful or memorable manner. Your audience will be more interested in your facts if you present them as visual statistics, and you will be more inclined to act on your discoveries.

Food delivery apps

When you place an order for food on your phone, it is given to the nearest delivery person. There is a lot of math involved here, such as the distance between the delivery executive's present position and the restaurant, as well as the time it takes to get to the customer's location.

Customer orders, delivery location, GPS service, tweets, social media messages, verbal comments, pictures, videos, reviews, comparative analyses, blogs, and updates have all become common ways of data transmission.

Users may obtain data on average wait times, delivery experiences, other records, customer service, meal taste, menu options, loyalty and reward point programmes, and product stock and inventory data with the help of the data.