Lecture 5
Big Data Descriptive Analytics

Dr. Lydia Wahid

Agenda

- Introduction
- Clustering
- Applying Clustering on Big Data
- Association Rules
- Applying Association Rules on Big Data

Introduction

Introduction

- **Descriptive Analytics** use **Descriptive Data Mining** techniques which use **Unsupervised Machine Learning** techniques.

Clustering

Clustering

- **Clustering** is used to identify groups of similar objects in a multivariate data sets.
- As a data mining function, cluster analysis serves as a tool to gain insight into the distribution of data to observe characteristics of each cluster.
- Clustering analysis is broadly used in many applications in data mining:
  - Clustering can be used in **image processing**, for example, in a video k-means analysis can be used to **identify objects** in the video.
  - Clustering can help markets **characterize their customer groups** based on the purchasing patterns.
  - Clustering helps in **classifying documents** on the web for **information discovery**.
  - Clustering is used in **outlier detection** applications as detection of **credit card fraud**.

Clustering: Methods

- There are different **types of clustering** methods, including:
  - Partitioning clustering
  - Hierarchical clustering
  - Fuzzy clustering
  - Density-based clustering
  - Grid-based clustering
  - Model-based clustering
  - Constraint-based clustering

Clustering: Methods

- **Partitioning clustering:**
  - Suppose we are given a database of ‘n’ objects, the partitioning method constructs ‘k’ partition of data. Each partition will represent a cluster where each cluster contains at least one object, and each object must belong to exactly one cluster.
  - For a given number of partitions (say k), the partitioning method will create an initial partitioning.
  - Then it uses the iterative relocation technique to improve the partitioning by moving objects from one cluster to other.

Clustering: Methods

- **Hierarchical clustering:**
  - This method creates a hierarchical decomposition of the given set of data objects. There are two approaches here:
    - **Agglomerative Approach:** Known also as the bottom-up approach. In this, we start with each object forming a separate group. It keeps on merging the objects or groups that are close to one another until the termination condition holds.
    - **Divisive Approach:** Known also as the top-down approach. In this, we start with all of the objects in the same cluster. In the continuous iteration, a cluster is split up into smaller clusters until the termination condition holds

Clustering: Methods

Fuzzy clustering:

- Fuzzy clustering is also known as **soft** method. Standard clustering approaches produce partitions, in which each observation belongs to only one cluster. This is known as **hard** clustering.
- In Fuzzy clustering, items can be a member of more than one cluster. Each item has a set of membership coefficients corresponding to the degree of being in a given cluster.

Density-based clustering:

- In density-based clustering, clusters are defined as areas of **higher density** than the remainder of the data set. Objects in sparse areas - that are required to separate clusters - are usually considered to be **noise** points.

Clustering: Methods

Grid-based clustering:

- In this method, the objects together form a **grid**. The object space is divided into **finite number of cells** that form a grid structure.
- Cells are chosen randomly until all cells are traversed.
- The density of a cell ‘ c’ is calculated and if this density is greater than threshold density, then mark cell ‘c’ as a new cluster.
- The densities of the neighboring cells are calculated and based on the threshold value it is decided to merge it with cell ‘c’.

Clustering: Methods

Model-based clustering:

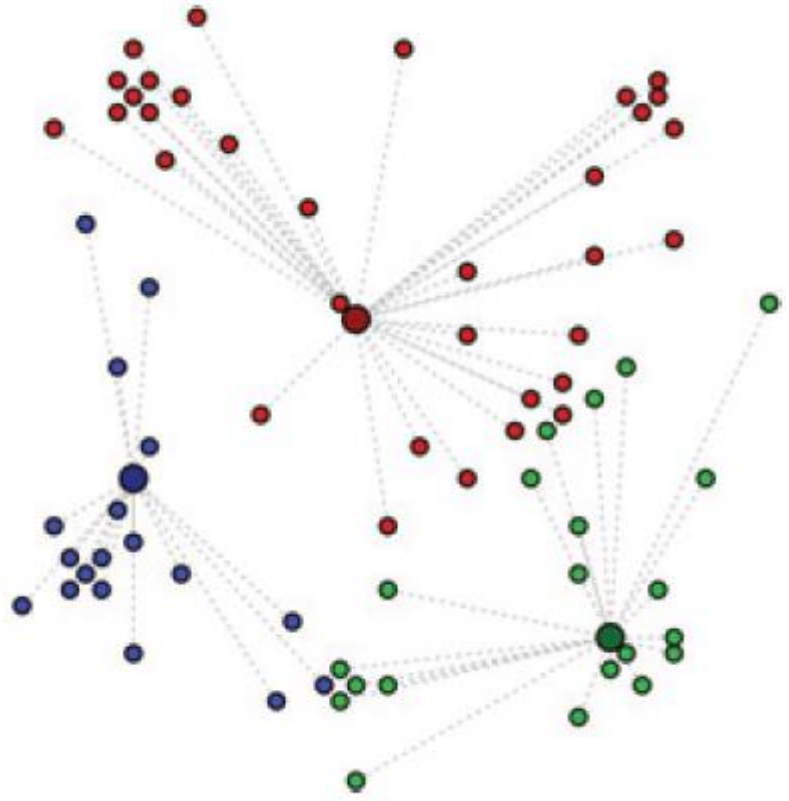
- **Model-based clustering** assumes that the data were generated by a **model** and tries to recover the original **model** from the data. The **model** that we recover from the data then defines **clusters** and an assignment of objects to **clusters**. A commonly used criterion for estimating the **model** parameters is maximum likelihood.
- This method provides a way to automatically determine the number of clusters based on standard statistics, taking outlier or noise into account.

Constraint-based clustering:

- In this method, the clustering is performed by the incorporation of **user or application-oriented constraints.** A constraint refers to the user expectation or the properties of desired clustering results. Constraints provide us with an interactive way of communication with the clustering process.

Clustering: K-means

- 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.
- It is a Partitioning clustering method.



Possible k-means clusters for k=3

Clustering: K-means

- 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 to each centroid. Assign each point to the closest centroid. This defines the first k clusters.
     - For a given point, **pi** (pi1, pi2, ..pin) and a centroid, **q**, (q1, q2,..qn) where n is the number of features, the distance, **d**, between pi and q, is expressed as shown in the following equation:

$$d(p_i, q) = \sqrt{\sum_{j=1}^{n} (p_{ij} - q_j)^2}$$

Clustering: K-means

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

3. Compute the centroid, the center of mass, of each newly defined cluster from Step 2. The centroid, ***q,*** of a cluster of **m** points, where each point pi has n features(pi1, pi2, ..pin), is calculated as in the following equation:

$$(q_1, q_2, \ldots, q_n) = \left(\frac{\sum_{I=1}^{m} p_{I1}}{m}, \frac{\sum_{I=1}^{m} p_{I2}}{m}, \ldots, \frac{\sum_{I=1}^{m} p_{In}}{m} \right)$$

4. Repeat Steps 2 and 3 until the algorithm converges to an answer.
   - Convergence can be reached when the computed centroids do not change or the centroids and the assigned points oscillate back and forth from one iteration to the next.

Clustering: K-means

- **Determining the Number of Clusters (i.e. K):**
  - The value of k can be chosen based on a reasonable guess or some predefined requirement.
  - However, even then, it would be good to know how much better or worse having k clusters versus k – 1 or k + 1 clusters would be in explaining the structure of the data.
  - Next, a heuristic using the **Within Sum of Squares** (WSS) metric is examined to determine a reasonably **optimal value of k**.

Clustering: K-means

Determining the Number of Clusters (i.e. K):

- **WSS** is defined as shown in the following equation:

$$WSS = \sum_{I=1}^{M} d(p_I, q^{(I)})^2 = \sum_{I=1}^{M} \sum_{J=1}^{n} \left(p_{IJ} - q_J^{(I)}\right)^2$$

- **WSS** is the sum of the squares of the distances between each data point and the closest centroid. **M** is the number of objects in the dataset. The term $\boldsymbol{q^{(i)}}$ indicates the closest centroid that is associated with the ith point. If the points are relatively close to their respective centroids, the WSS is relatively small. Thus, if k + 1 clusters do not greatly reduce the value of WSS from the case with only k clusters, there may be little benefit to adding another cluster.

Clustering: K-means

- **Advantages:**
  - Fast convergence for clustering small datasets
  - Easy to implement
- **Drawbacks:**
  - Computationally expensive for large datasets
  - Doesn't guarantee to converge to a global minimum. It is sensitive to the centroids' initialization. Different setups may lead to different results.
  - Strong sensitivity to outliers

Clustering: K-means++

- **K-means++**
  - K-means algorithm is sensitive to the initialization of the centroids.
  - **K-means++** is an algorithm for choosing the initial values for the K-means clustering algorithm to ensure a smarter initialization of the centroids
  - Apart from initialization, the rest of the algorithm is the same as the standard K-means algorithm.
  - First centroid is chosen randomly from the data points.
  - The next centroid is chosen from the data points such that **the probability of choosing a point as centroid is directly proportional to its distance from the nearest, previously chosen centroid**.
  - K-means++ is more likely to converge and run faster than K-means.

Clustering: k-medoids

- **K-Medoids** (also called as Partitioning Around Medoid - **PAM**):
  - It is known to be less sensitive to outliers than k-means.
  - The PAM algorithm searches for k representative objects in a data set (**k medoids**) and then assigns each object to the closest medoid in order to create clusters.
  - These medoids are actual observations from the dataset and not computed points (mean value) like in the case of k-means.
  - Its aim is to minimize the sum of dissimilarities between the objects in a cluster and the center of the same cluster (medoid).

Applying Clustering on Big Data

Applying Clustering on Big Data

- We now consider how to reformulate **K-means** algorithms for solving the clustering problem on **Big Data**.
- We will formulate this algorithm using **MapReduce** technique.
- Each iteration of standard k-means can be divided into two phases, the first of which **computes the sets of points closest to mean μi** , and the second of which **computes new means** of these sets.
- These two phases correspond to the **Map** and **Reduce** phases of the MapReduce algorithm.

Applying Clustering on Big Data

- The **Map phase** operates on each point x in its split.
- For a given x, find the mean μi (the cluster) which minimizes the **squared distance between x and the mean.**
- Then a <key-value> pair is emitted with this **mean's index i as key** and the **value x i.e <i, x>**
- Note that in order for the Map function to compute the distance between a point x and each of the means, each machine in our distributed cluster **must have the current set of means**. We must therefore **broadcast** the new means at each iteration.

Applying Clustering on Big Data

- The output of the map phase is huge, so a **combiner** is used to minimize the size of the data.
- The combiner calculates the average of the data instances for each cluster id, along with the number of the instances.
- It outputs **< i, (cluster mean, number of instances)>**

Applying Clustering on Big Data

- The **reduce phase** calculates the means by iterating over the values of the same cluster index.
- The shared means values have to be updated to the new computed values.
- **All the mentioned procedure is repeated until the convergence criterion is met.**

Association Rules

Association Rules

- Association rules method is an unsupervised learning method.
- This is a descriptive method often used to discover **interesting relationships hidden in a large dataset**.
- The disclosed relationships can be represented as rules or frequent itemsets.
- Here are some possible questions that association rules can answer:
  - Which products tend to be purchased together?
  - Of those customers who are similar to this person, what products do they tend to buy?
  - Of those customers who have purchased this product, what other similar products do they tend to view or purchase?

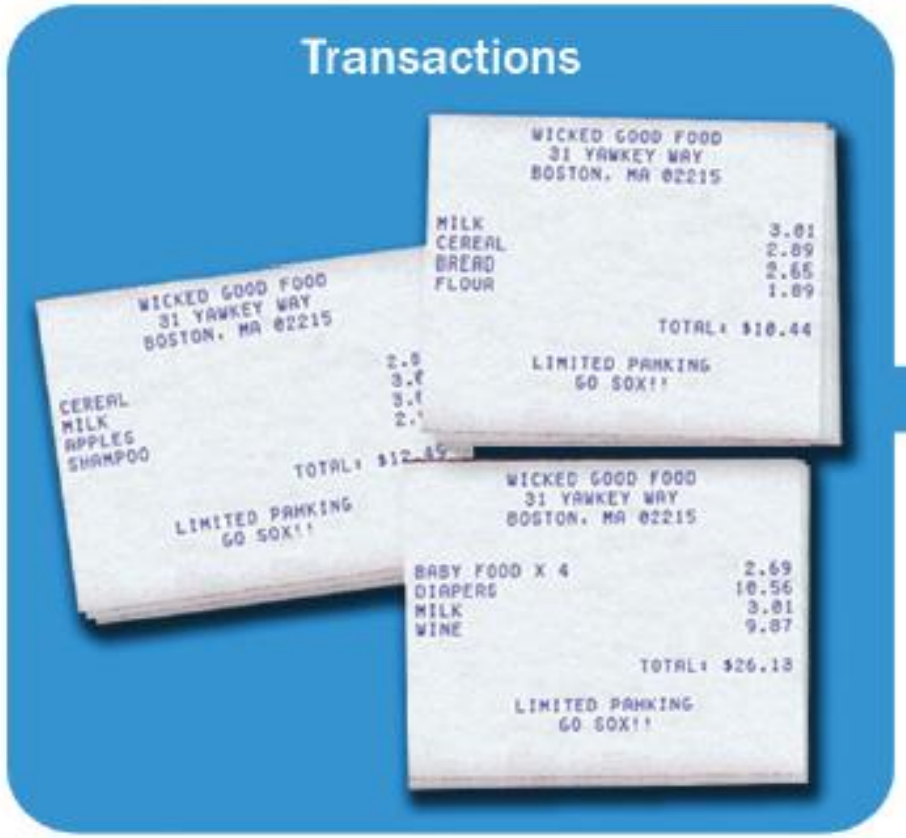
Association Rules: Overview

- For example, given a large collection of transactions in which each transaction consists of one or more items:

⇒association rules go through the items being purchased to see **what items are frequently bought together** and to **discover a list of rules that describe the purchasing behavior**.

- **The goal with association rules is to discover interesting relationships among the items.**
- The relationship occurs too frequently to be random and is meaningful from a business perspective.

Association Rules: Overview



The general logic behind association rules

Rules

Cereal	➡	Milk (90%)
Bread	➡	Milk (40%)
Milk	➡	Cereal (23%)
Milk	➡	Apples (10%)
...		...
...		...
...		...

when cereal is purchased, 90% of the time milk is also purchased.

Association Rules: Overview

- Each of the uncovered rules is in the form **X** → **Y**, meaning that when item **X** is **observed**, item **Y** is **also observed**.
- In fact, because of their popularity in mining customer transactions, association rules are sometimes referred to as ***market basket analysis.***
- Each transaction can be viewed as the shopping basket of a customer that contains one or more items.
- This is also known as an ***itemset***.
- An itemset containing ***k*** items is called a ***k-itemset***.

Association Rules: Overview

- The term ***itemset*** generally refers to a collection of items or individual entities that contain some kind of relationship.
- This could be:
  - a set of retail items purchased together in one transaction,
  - a set of hyperlinks clicked on by one user in a single session,
  - a set of tasks done in one day,
  - others

Association Rules: Apriori Algorithm

- **Apriori** is one of the earliest and the most fundamental algorithms for generating association rules.
- It pioneered the use of support for pruning the itemsets and controlling the exponential growth of candidate itemsets.
- This approach eliminates the need for all possible itemsets to be enumerated within the algorithm, since the number of all possible itemsets can become exponentially large.

Association Rules: Apriori Algorithm

- One major component of Apriori is **support**.
- Given an itemset *L*, the **support** of *L* is the percentage of transactions that contain *L*.
- For example, if 80% of all transactions contain itemset {bread}, then the support of {bread} is 0.8.
- A ***frequent itemset*** has items that appear together often enough. The term "often enough" is formally defined with a ***minimum support*** criterion.
- If the minimum support is set at 0.5, any itemset can be considered a frequent itemset if at least 50% of the transactions contain this itemset.

Association Rules: Apriori Algorithm

- If an itemset is considered frequent, then any subset of the frequent itemset must also be frequent. If an itemset is infrequent, all its supersets will be infrequent.
- This is referred to as the ***Apriori property*** (or ***downward closure property***).
- For example, if 60% of the transactions contain {bread,jam}, then at least 60% of all the transactions will contain {bread} or {jam}.
- In other words, when the support of {bread,jam} is 0.6, the support of {bread} or {jam} is at least 0.6.
- **The Apriori property provides the basis for the Apriori algorithm.**

Association Rules: Apriori Algorithm steps

- **Step1: Generate 1-itemsets (L1)**
  - **Determine all the possible items** (or 1-itemsets, for example {bread}, {eggs}, {milk}, …)
  - **then identify which among them are frequent:**
    - Assuming the minimum support threshold (or the minimum support criterion) is set at 0.5:
    - **the algorithm identifies and retains those itemsets** that appear in at least 50% of all transactions
    - and **discards (or "prunes away") the itemsets** that have a support less than 0.5.

Association Rules: Apriori Algorithm steps

- **Step2: Generate 2-itemsets (L2)**
  - **the identified frequent 1-itemsets are paired into 2-itemsets** (for example, {bread,eggs}, {bread,milk}, {eggs,milk}, …)
  - and again evaluated to **identify the frequent 2-itemsets among them**.

Association Rules: Apriori Algorithm steps

- **Step3: Generate 3-itemsets (L3)**
  - Here we use **Apriori Property** for the generation of the candidate set of three itemsets. We perform two steps: **Join** and **Prune**
  - For the generation of 3-itemsets we compute **L2 join L2**. Condition of joining Lk-1 and Lk-1 is that it should have **(K-2) elements in common**.
  - After that, we move to the **Prune step** to reduce the size: Check if all subsets of these item sets are frequent or not (what triplets can we make that do not contain any pair that was eliminated?)
  - After that, **compute the support** of the remaining 3-itemsets and identify which among them are frequent and eliminate the others.

Association Rules: Apriori Algorithm steps

- The same procedure is repeated in the **next iterations** of the Apriori algorithm.
- **At each iteration**, the algorithm checks whether the support criterion can be met;
  - if it can, the algorithm grows the itemset, repeating the process until it runs out of support or until the itemsets reach a predefined length.

Association Rules: Apriori Algorithm steps

- To **generate the association rules**, find all the rules of the frequent itemset such that these rules have **higher confidence value than the threshold or minimum confidence**.
- Confidence is the percent of transactions that contain both X and Y out of all the transactions that contain X and computed as follows:

$$Confidence(X \rightarrow Y) = \frac{Support(X \wedge Y)}{Support(X)}$$

Association Rules: Apriori Algorithm example

- **Example:** Suppose we have the following dataset that has various transactions, and from this dataset, we need to find the frequent itemsets and generate the association rules using the Apriori algorithm:

TID	ITEMSETS
T1	A, B
T2	B, D
T3	B, C
T4	A, B, D
T5	A, C
T6	B, C
T7	A, C
T8	A, B, C, E
T9	A, B, C

Association Rules: Apriori Algorithm example

- **Step-1**: Assume minimum support count is 2, Calculate the support count for each itemset and prune those items with support count less than the minimum support count.

Itemset	Support_Count
A	6
B	7
C	5
D	2
E	1

Itemset	Support_Count
A	6
B	7
C	5
D	2

Item E is pruned since it doesn't satisfy the minimum support count criteria

Association Rules: Apriori Algorithm example

- **Step-2**: Create pairs of the frequent itemsets that are identified in Step-1 and again calculate the support count for each itemset and prune those itemsets with support count less than the minimum support count.

Itemset	Support_Count
{A, B}	4
{A,C}	4
{A, D}	1
{B, C}	4
{B, D}	2
{C, D}	0

Itemset	Support_Count
{A, B}	4
{A, C}	4
{B, C}	4
{B, D}	2

Association Rules: Apriori Algorithm example

- **Step-3**: Expand the frequent itemsets that are identified in Step-2 to include 3 items and again calculate the support count for each itemset and prune those itemsets with support count less than the minimum support count.

Itemset	Support_Count
{A, B, C}	2
{A, B, D}	0
{B, C, D}	1

Itemset	Support_Count
{A, B, C}	2

Note that the supports of {A,B,D} and {B,C,D} are not computed since their 2-itemsets subsets are not frequent i.e. {A,D} and {C,D} then any superset is also not frequent. Apriori property is used here.

Association Rules: Apriori Algorithm example

- **Step-4**: Generating the association rules; we will now apply on the last frequent itemset reached (generally frequent itemsets from the other iterations are considered).

Rules	Support	Confidence
A ^B → C	2	Sup{(A ^B) ^C}/sup(A ^B)= 2/4=0.5=50%
B^C → A	2	Sup{(B^C) ^A}/sup(B ^C)= 2/4=0.5=50%
A^C → B	2	Sup{(A ^C) ^B}/sup(A ^C)= 2/4=0.5=50%
C→ A ^B	2	Sup{(C^(A ^B)}/sup(C)= 2/5=0.4=40%
A→ B^C	2	Sup{(A^(B ^C)}/sup(A)= 2/6=0.33=33.33%
B→ B^C	2	Sup{(B^(B ^C)}/sup(B)= 2/7=0.28=28%

Association Rules: Apriori Algorithm example

- **Step-4**: We will exclude the rules that have less confidence than the minimum threshold (50%).
- Therefore, the rules selected will be:
  - **{A, B} → C**
  - **{B, C} → A**
  - **{A, C} → B**

Association Rules: Evaluation of Candidate Rules

- We have seen two measures which are: **Support** and **Confidence.**
- There are other measures such as: **Lift** and **Leverage**
- ***Lift*** measures how many times more often X and Y occur together than expected if they are statistically independent of each other. Lift is a measure of how X and Y are really related rather than coincidentally happening together:

$$Lift(X \rightarrow Y) = \frac{Support(X \wedge Y)}{Support(X) * Support(Y)}$$

A larger value of lift (greater than 1) suggests a greater strength of the association between X and Y.

Association Rules: Evaluation of Candidate Rules

- ***Leverage*** is a similar notion, but instead of using a ratio, leverage uses the difference. Leverage measures the difference in the probability of X and Y appearing together in the dataset compared to what would be expected if X and Y were statistically independent of each other.

$$Leverage(X \rightarrow Y) = Support(X \wedge Y) - Support(X) * Support(Y)$$

A larger leverage value indicates a stronger relationship between X and Y.

Association Rules: Evaluation of Candidate Rules

- Confidence is able to identify **trustworthy rules**, but it cannot tell whether a **rule is coincidental**. **Why?**

- A high-confidence rule can sometimes be misleading because confidence **does not consider support** of the itemset in the **rule consequent**.

- Measures such as lift and leverage not only ensure interesting rules are identified but also **filter out the coincidental rules**.

Applying Association Rules on Big Data

Applying Association Rules on Big Data

- A MapReduce job splits the input transaction database into various blocks.
- The **map task** parses one transaction at a time and extracts each itemset included in the transaction it received as input.
- After processing, the mapper sends the itemset to the partitioner by emitting the itemset and frequency as <key, value> pair, where **'key 'is a candidate itemset and "value" is 1**.

Applying Association Rules on Big Data

- The **partition task** collects all the intermediate <key, value> pair emitted from the map task as its input.
- Based on the **key size**, i.e., the size of each itemset, the partitioner specifies that all the values for each itemset are grouped together and maps all the values of a single key to go to the same reducer.
- The output of all partitioner are shuffled and exchanged to make the list of values associated with the same key as <key,list(value)> pairs.

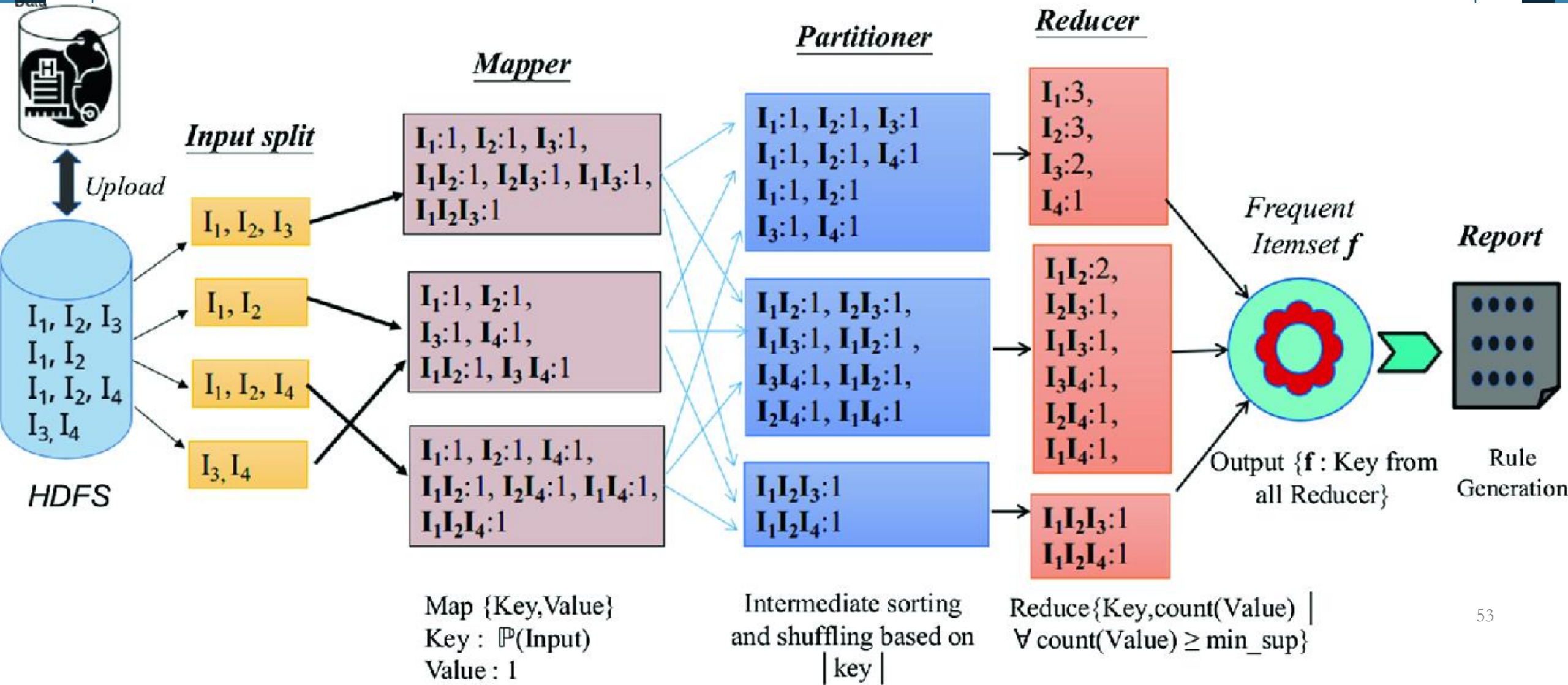
Applying Association Rules on Big Data

- The **reduce task** collects each key passing all the values emitted against the same key as arguments, i.e., <key,list(value)> pairs emitted by partitioner task.

- Then, it **sums up** the values of respective keys. Candidate itemset whose sum of values is **supportcount<minimum_support_count** is discarded.

- The result from all reducers is written to the output file.

Applying Association Rules on Big Data



References

- Bodoia, M. (2016). MapReduce Algorithms for k-means Clustering. Stanford University
- Bhattacharya N., Mondal S., Khatua S. (2019) A MapReduce-Based Association Rule Mining Using Hadoop Cluster—An Application of Disease Analysis. In: Saini H., Sayal R., Govardhan A., Buyya R. (eds) Innovations in Computer Science and Engineering. Lecture Notes in Networks and Systems, vol 74.

Thank You