Privacy Preservation For High Dimensional Data Using Slicing Method in Data Mining

Abhang Vikram Kishor¹, Dhole Poonam Balasaheb², Sable Balasaheb S³

¹Department of Computer Engg, AVCOE, Sangamner
²Department of Computer Engg, AVCOE, Sangamner
³Department of Computer Engg, AVCOE, Sangamner

Abstract—In recent days, privacy-preservation for high dimensional data and micro data publishing has seen rapid advances that causes ability to store and record personal data. The method of publishing the data in the web faces many challenges now a days. The data usually contains the individual personal data which are personally identifiable to any person, thus poses the difficulty of Privacy. The existing privacy preserving system has been developed using anonymization techniques of Generalization and Bucketization. Generalization loses the amount of data and at other side Bucketization fail to protect membership disclosure in addition to does not give the clear idea of separation between sensitive attribute as well as quasi-identifying attributes. To solve this problem we introduce novel technique which is useful to partitions data horizontally as well as vertically, which is called as Slicing. Our result shows that slicing useful to preserve better data utility as compare to the Generalization and also protect the membership disclosure. Our contribution on this project is preserves utility as a result of it teams extremely correlate attributes along, and preserves the correlations between such attributes and effective utilization of Hadoop Frame work to handle high dimensional data.

Keywords: Data anonymization, Hadoop, Privacy preserving, Slicing.

I. INTRODUCTION

Data mining is that the extracting the important data from the large data sets like data warehouse. Micro data contains records which of that contains data about a personal entity, data can be an individual data or a company data or unit data. Now a days various micro data anonymization techniques are introduced for data publishing. In leading extensive the Generalization and Bucketization for k-anonymity [1] and l-diversity respectively [1][4]. In both privacy preserving approaches attributes square compute partitioned off into 3 categories:(1) several attributes are identifiers that can exclusively identify an individual or personal, such as for example Name or Social Security or insurance Number; (2) several attributes are Quasi similar Identifiers (QI), which are in advance possibly will already know (it is possibly from other publicly-available databases) and which can be taken together, know how to potentially recognize an individual or personal, for e.g., Birth-date, Sex or Gender, and Zip code or Area code; (3) several attributes are Sensitive or responsive Attributes (SAs), which are unknown to the challenger and are considered responsive, such as for example Disease and Pay Salary.

The major disadvantages of approach of k-anonymity is that it link to the external data with shared data for analysis point. In generalization all the attributes are suppressed till every row is one and the same(identical). It is useful to protect identifier however it's not assurance to the absolute privacy and misplace the data in high dimensional data. In Bucketization technique each and every ones sensitive data denote “The values which are represented in well define format”. Bucketization has many limitations, primarily is fail to protect membership disclosure. As a result of Bucketization publish the similar symbol (QI) values in their unique forms, AN human being will verify whether or not a personal incorporates a record within the already disclosure data or not. Our projected Slicing algorithm with Tuple grouping formula is divided into the data each vertically and horizontally. The randomized values are permutated at every intervals of bucket and it can also useful to handle in high
Our work and analysis shows that, how we can share and publish micro data in a privacy preserving way. We can present an expressive result of this problem along three dimensions: (1) design a simple, intuitive, and well-built privacy model; (2) design a more valuable anonymization technique that works on high-dimensional data and scruffy or the sparse data; and (3) design and developed a methods for estimating privacy and utility tradeoff.

II. LITERATURE SURVEY

In this section we will discuss existing system and algorithm With its merits as well as demerits:

2.1 K-Anonymity: Samarati and Sweeney Samarati work on K-anonymity seeks to prevent the identity disclosure of micro data which is release and based on the quasi-identifier attributes. This paper shows that K-anonymity requires each an every combination of quasi-identifier values which are shared by (at least) k records in those are released data set. Each release of data must be combination of values of quasi-identifiers which can be indefinitely matched to at least k individuals. Most usual method to accomplish k-anonymity is based on generalization of the quasi-identifier attributes which is discussed in Samarati and Sweeney and suppression of records that would direct to increased generalization. up till now alternative approaches exist, like Domingo-Ferrer and Torra, which is useful on micro aggregation of the quasi-identifier attributes.

2.2 l-Diversity and t-closeness: k-anonymity protects identity disclosure, as mentioned above, it fail to protect attribute disclosure: if the values of one or several confidential attribute(s) are matching within a group of records sharing the quasi-identifier attribute values, disclosure happens. The property of l-diversity (Machanavajjhala) as an extension of k-anonymity which tries to solve the attribute disclosure problem. Achieving l-diversity in general implies it shows more distortion than just achieving k-anonymity. t-closeness (Li et al.) is another extension of k-anonymity which is design to solve the attribute disclosure problem. This t-closeness clearly solves the attribute disclosure weakness, although the original t-closeness paper did not suggest a computational method to accomplish this property nor did it mention the large utility loss that this property is likely to impose on the original data.

2.3 Various anonymization techniques like Generalization and Bucketization: These two techniques have been designed for privacy preserving micro data publishing. Generalization loses the amount of data and at other side Bucketization fail to protect membership disclosure in addition to does not give the clear idea of separation between sensitive attribute as well as quasi-identifying attributes. Tiancheng Li, Ninghui Li, Jian Zhang, Ian Molloy has worked on this problem and they introduced new technique which is known as ‘Slicing’.

Disadvantages of Existing System:
1. Existing anonymization algorithms can be used for column generalization for e.g. Mondrian. The algorithms can be useful to the suitable containing only attributes in one column to make sure the anonymity requirement.
2. Existing data analysis for e.g., query answering methods can be easily used on the sliced data.
3. Existing privacy method for membership disclosure protection include differential privacy and occurrence.

We studied all these anonymization techniques, we analysed that if we handle high dimensional data over Hadoop using slicing then we get better efficiency and time complexity will reduce. We studied Hadoop framework and how to design and developed this system over Hadoop. Our contribution on this project is preserves utility as a result of it teams extremely correlate attributes.
along, and preserves the correlations between such attributes and effective utilization of hadoop frame work to handle high dimensional data. [11, 12 ]

III. IMPLEMENTATION DETAILS

3.1 Mathematical Model:
3.1.1 Set Theory: Set S = {A, C, B, D, T }
A = (a0, a1; :::; an) ØA,
C = (c0, c1; :::; cn) ØC,
B = (b0, b1; :::; bn) ØB,
D = (d0,d1; :::; dn)ØD.
T = (t0, t1; :::; tn) ØT.
Where, Set A=represents set of attributes
Set C=represents set of columns
Set B=represents set of buckets
Set D=represents set of attribute domains
Set T=represents set of tuples
3.1.2 Venn Diagram:

![Venn Diagram](image)

3.1.3 DFA theory
Definition:
A deterministic finite automaton (DFA)
(a) A finite set of states (often denoted Q)
(b) A finite set S of symbols (alphabet)
(c) A transition function that takes as argument a state and symbol and returns a state (often denoted d) The transition function d is a function in d: Q_S = Q
(d) A start state often denoted q0
(e) A set of final or accepting states (often denoted F)
So a DFA is mathematically represented as a 5-tuple (Q; S; d; q0; F)
3.1.4 State transition diagram
Step I:
A Finite set of states (Q) = {q0, q1,q2, q3, q4} where, (a) q0 denotes initial state and q3 denotes final state q1,q2, q3 are internal states
(b) A set of final or accepting states (F) = {q4}
Step II:
State diagram:
q0 = Dataset Extraction
q1 = Generalization and Hadoop
q2 = Bucketization and Hadoop
q3 = Slicing and Hadoop
q4 = Graph Generation
Step III:
State Transition Diagram:

\[
\begin{array}{c|ccccc}
 & q0 & q1 & q2 & q3 & q4 \\
q0 & 1 & 1 & 0 & 1 & 0 \\
q1 & 0 & 1 & 1 & 0 & 0 \\
q2 & 0 & 0 & 0 & 1 & 0 \\
q3 & 0 & 0 & 0 & 1 & 1 \\
q4 & 1 & 0 & 0 & 0 & 0 \\
\end{array}
\]

Table 1. State Transition Table

3.2 Design:
Existing System: First, several existing clustering algorithms for e.g., k-means requires the calculation of the centroids. But in actual there is no notion of centroids in that setting where each an every attribute forms a data point in the clustering gap. Second one, k-medoid method is very vigorous to the survival of outliers i.e., data points that are very distant away from the rest of data points. Third, the sequence in which the data points are examined does not affect the clusters computed from the k-medoid method. Generalization loses the amount of data and at other side Bucketization fail to protect membership disclosure in addition to does not give the clear idea of separation between sensitive attribute as well as quasi-identifying attributes.

3.2.1 Previous Slicing Architecture:
Working of Fig 4. Slicing architecture as follows: Extracting dataset from the dataset. Process of Anonymity divides the records into Two. Interchanging sensitive values. Multi set values generation and Displaying record. Displaying attributes those are collective and secure data.

3.2.2 Our Proposed Architecture:
Fig 5. Slicing proposed architecture over hadoop

Below Figure shows the detail architecture of hadoop: It is the software framework for distributed processing of large data sets on computer cluster. Map and reduce have general interface Each receives sequence of records and produces records in response. A record consists of key and value. A Map Reduce job generally splits the input data-set into the independent chunks which are processed by the map tasks in a entirely parallel manner. The framework working is mapper output gives input to the Reducer. Both input and output job stored in file system. The framework takes care about scheduling tasks and monitoring them and re-executes the failed tasks. This is most useful benefit of Hadoop. Basically the compute nodes and the storage nodes are the same type, as like Map Reduce framework and the HDFS are running on the same set of nodes. This arrangement allows the framework to efficiently schedule tasks on the nodes where data is previously present, resulting in very high collective bandwidth across the cluster.

Fig 6. Hadoop architecture
Fig 7. Map Reduce process

Generally Map Reduce framework consists of a single master Job Tracker and one slave Task Tracker per cluster-node. Master is responsible for scheduling the jobs part tasks on the slaves and monitoring them, re-executing the failed tasks. Slaves node execute the tasks as directed by the master.

3.3 Platform: We are going to use a cluster of 4-6 nodes, all machines involved in the experiments has Pentium IV 2.4 GHz processors and 2GB of memory and going to run Ubuntu 12.04 or Fedora 20 LTS Linux operating system. Hadoop version 01.X and java 1.6.0_13 are going to use for performance and Eclipse Kepler for java code.

IV. RESULTS

4.1 Datasets: We analyzed that slicing gives better utility as compare to generalization. In experimental result involving the sensitive attributes, slicing is also more useful than bucketization. For this project our data set is Diabetics patient data, which contain high dimensional data.

Fig 8: Starting all nodes of Hadoop

The following window shows the two files name as part-r-t-00000 and success. The part-r-t-00000 is the output file. It stores the output of the execution. As we know the output is stored on Hadoop Distributed File System. Our Generalization i.e t1.txt, Bucketization i.e. t2.txt, Slicing i.e. t5.txt output also stored in HDFS.
This below snap shows Sliced data after performing horizontal and vertical partitioning. Sliced data reduce the size of original data. Our result shows improvement in efficiency of system as compare with existing system.

Below Graph shows result of Generalization, Bucketization, Slicing Over Hadoop. After performing horizontal and vertical partitioning data reduces the size. Its shows slicing preserve better data utility. Our result proves Slicing useful to handle high dimensional data. Our result shows better efficiency than existing system.
After vertical and horizontal partitioning data size reduced. Slicing preserve better data utility than Generalization and Bucketization. Step0 for Original Data. And Step1 for Sliced Data.

This graph shows slicing useful to preserve better data utility as compare to the Generalization and also protect the membership disclosure.

V. CONCLUSION AND FUTURE WORK
Slicing useful to preserve better data utility as compare to the Generalization and also protect the membership disclosure. Slicing useful to handle high dimensional data. Our contribution on this
A paper is preserved utility as a result of it teams extremely correlate attributes along, and preserves the correlations between such attributes and effective utilization of hadoop Frame work to handle high dimensional data.

Our future work is depend on how to anonymizing the data before it used, one can analyze these data features and use these features in data anonymization technique. As per our future work we plan to design additional efficient tuple grouping algorithms.

REFERENCES


[11] hadoop.apache.org/docs/r0.18.0/hdfs_design.pdf

[12] hadoop.apache.org/docs/r0.18.3/mapred_tutorial.html
