Data Anonymizer

Overview : This document contains 5 sections, section-1 is about introduction to the project, section-2 and 3 contains the design and development of project, section-4 contains the user manual and section-5 contains future work.

1. Description

1.1. Context :

The digital world of now is composed of personalized data services and there is a huge demand for data, especially for personal data. This may take the form of financial data, health data, Internet trans- actions, or even data based on GPS location. Large volumes of data can now be employed quite efficiently to gain new insights on a certain phenomenon, and data mining algorithms can give a lot of information about persons, events or entities. The purposes for which one requires meaningful personal data are very diversified, for example, research or public health policy purposes, to develop new services and products, to enhance the efficiency and effectiveness of new drugs, or even to simply condition people's behaviour, when they make a purchase on an online store.

1.2. Motivation:

In recent years, escalation of technology led to an increase in the capability to record and store personal data about consumers and individuals. With this information almost anyone can track or know more about a person's life. This raised concerns on personal data misuses in many different ways. To mitigate these issues, some de-identification methodologies have recently been proposed that, in some well controlled circumstances, allow for the re-use of personal data in privacy-preserving ways. So one such particular method is Data Anonymization.

Data *anonymization* ensures that even if *de-identified* data is stolen, it is very hard to *re- identify* it. This report will present about a web tool that provides a way to easily anonymize data.

Anonymization:

Anonymity, means simply that a person is not identifiable. The *anonymization* process is intended to irreversibly remove the association between an individual and some information that can identify this person. Personal data is *anonymized* to protect the privacy of subjects when storing or disclosing data.

1.3. Scope of Project:

There are tools available which transforms the dataset into anonymized dataset. Some of them are open source, others are private and they need to be installed in the system to use. But there are no web applications that intend to the anonymization.

The main objective of this project is to create web application which make it easy for a user to anonymize the data. It means the user can upload the dataset and can choose configuration or create a new configuration with in the web interface for de-identification, after that the de-identified or anonymised data can be downloaded. The important task of this project was to create an interface for the user such that a new configuration can be developed on the dataset.

Design and Development

2. Architecture:

In order to give the web application some structure, a layered architecture was created, grouping the different components in sections. This layered separation is often used on *web* platforms, on this particular architecture is separated in three layers (tiers). Each of these layers is responsible for a particular and unique processing, they communicate with each other through different frame works and languages, so, aggregating all those tiers in a single framework simplifies hugely this interconnection process . The layers were called "Presentation Tier", "Business Tier", and "Data Tier”.

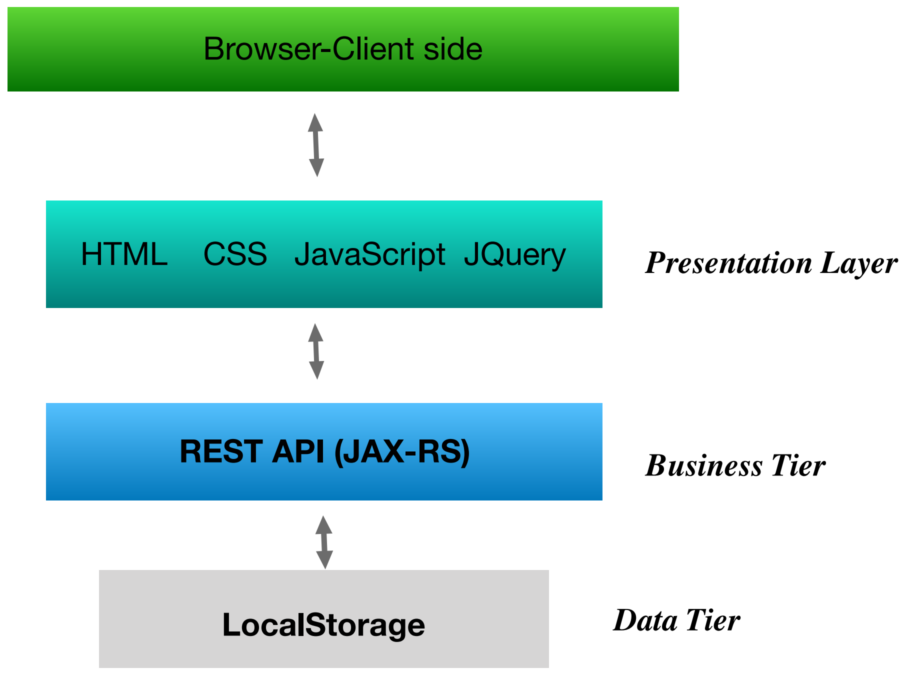


Figure: Web App Architecture

Presentation Layer:

This layer contains the user oriented functionalities, they are responsible for managing user interaction with the system, and generally consists of components that provide a common bridge to the business layer core. Presentation tier components allow users to interact with the application.

Business Tier:

This layer implements the core functionality of the system and encapsulates the relevant business logic. It generally consists of elements that execute rules and separates them from the user interface and data access**.**

Data Tier:

The data tier provides access to the databases and storage devices used by a three-tier applications. It is responsible for retrieving data and transforming into a suitable format for the rest of the application. Essentially, this layer stores and retrieves information to the business tier for processing and eventually to the presentation tier. On adapting the *ARX API* there was no need to use a database, because saving the files locally was sufficient enough to perform the *anonymization*.

3. Technologies:

3.1. Client-side

In the application *front-end*, we used five technologies: HyperText Markup Language (HTML), Cascading Style Sheets (CSS), *JavaScript* , *JQuery* and Asynchronous JavaScript and XML (*AJAX*). The *web* page presented on the user's browser was built on HTML, created from scratch, exhibiting a simple but intuitive interface that guide the user through a configuration process that sometimes could be a complex task. In order to turn the HTML more attractive, we applied CSS, giving a better visual presentation to the interface

Another technology was used in order to send and receive data from the server, is called JavaScript Object Notation (JSON), and is useful for parsing JavaScript objects into text. This is needed when the client wants to exchange data with the server, because this data has to be in text format. JSON converts JavaScripts objects in text, and vice-versa, the data is saved as pairs of key/value and they are kept as an array of strings. This technology was used in all the AJAX requests created, parsing the data sent to the server, or making the reverse process for the received array of data.

3.2. Server-side

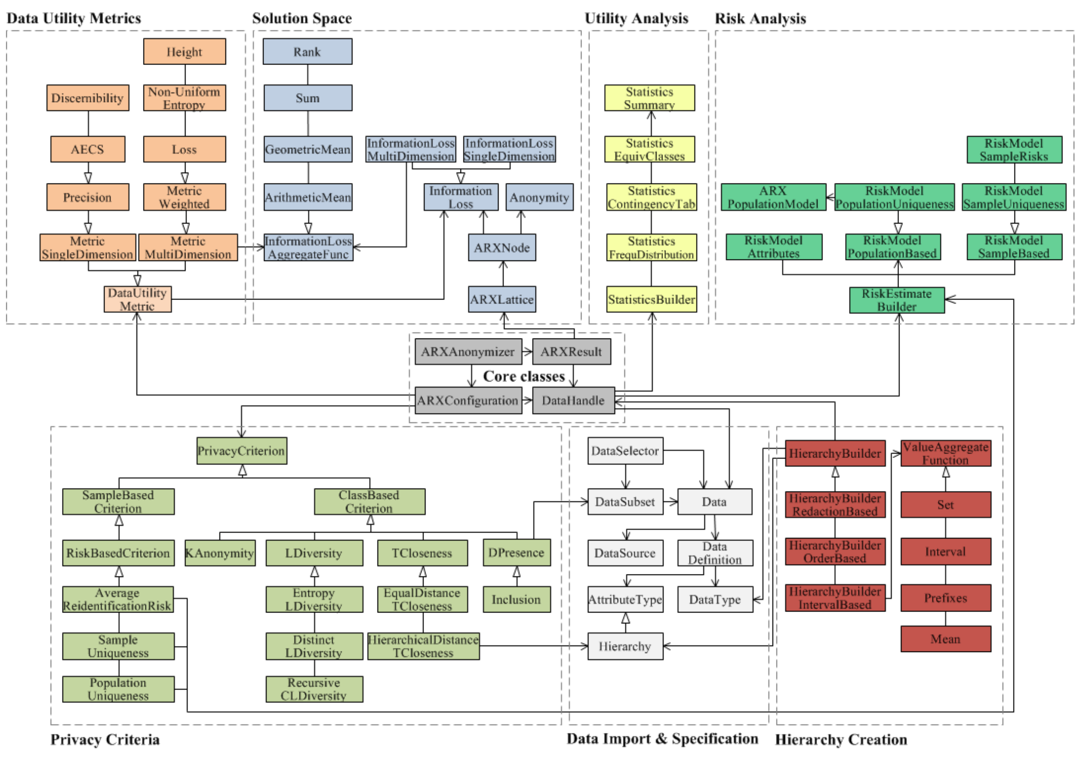
- **JERSEY (JAX-RS):** Jersey RESTful Web Services framework is open source, production quality framework for developing RESTful Web Services in Java that provides support for JAX-RS APIs. Jersey provides it's own API that extend the JAX-RS toolkit with additional features and utilities to further simplify RESTful service and client development.Another advantage is the use of "pom.xml" file that simplifies a lot the *Maven* configuration, this file contains information (dependencies and *plugins*) about the project that are used by *Maven* to build the project. For creating the RESTful services in this project jersey is being used.

- **REST:** A *web* application without resources don't serve for much, they are usually something that can be stored on a computer, e.g., an electronic document, a *dataset* or the result of executing an algorithm. These resources must have an URL, this way the servers and clients can trade them unmistakably due to their unique address . When a GET request is made, the server uses a representation that captures the current state of the resource required, this contains information (size, encoding or data) about that resource depending on what was requested. In *RESTful* systems, clients and servers communicate through messages that follow a predefined protocol, called HTTP. It defines different kinds of HTTP request, the most common are the following: GET, POST, PUT, DELETE.

- **Tomcat:** To deploy the services and resources, apache tomcat is being used as server.

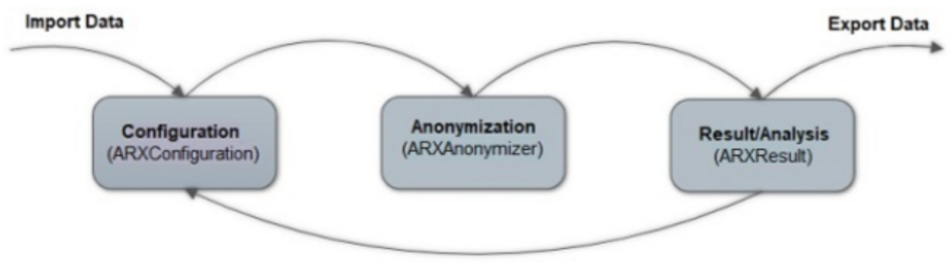
3.3 ARX API:

The *ARX* open source tool allows the user to change structured sensitive personal data, using Statical Disclosure Control (SDC), into data that can be shared securely. The objective is to reconstruct *datasets* in compliance with well-known syntactic privacy models, that will reduce the success of attacks, preventing privacy breaches *ARX* focus into PHI, being able to remove direct identifiers and introduce constraints on indirect identifiers. These QID are always assumed by the tool, as being known by the attacker, but they cannot be erased from the *datasets* because they are needed for processing.



ARX API has been chosen for anonymization process because of its simplicity and organised code, with a good documentation, plus a deductive online *javadoc* with all packages and classes included in the API. The aim of the programming interface is to provide *de-identification* methods to other software systems.

In the process of Anonymization the important classes are ***ARXConfiguration***, ***ARXAnonymizer***, ***ARXResult*** as shown in Figure 9**.**

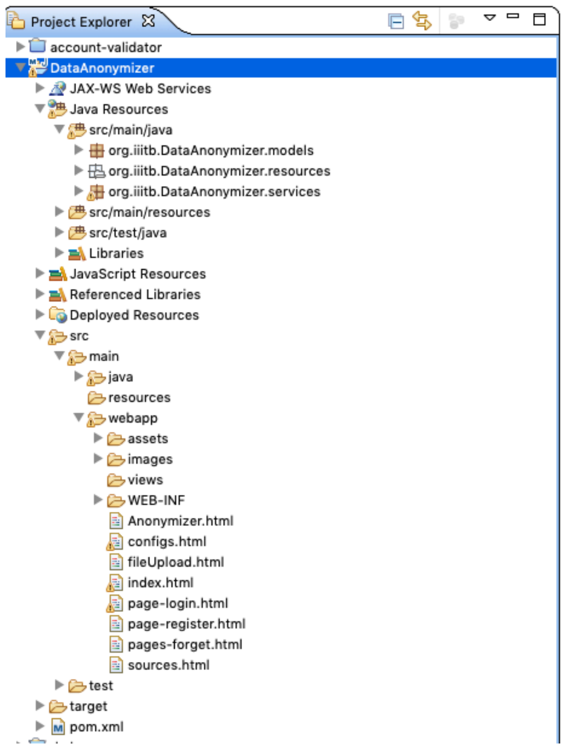


Import Data
Export Data
Configuration (ARXConfiguration)
Anonymization (ARXAnonymizer)
Result/Analysis (ARXResult)

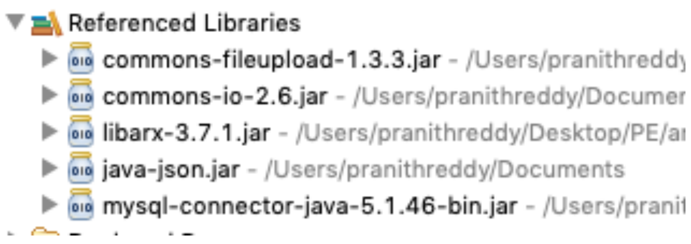
Figure 9: *Anonymization* Process

The *ARXConfiguration* defines a group of settings (privacy model, suppression rate, ...) that are sent to the *ARXAnonymizer*, this configuration allows multiple parameters, offering the user an opportunity to create suitable *de-identified datasets*. *ARXAnonymizer* offers several methods to define parameters and execute the *ARX* algorithm, according with the configuration given by *ARXConfiguration*, the outcome of this process will produce a result of the type *ARXResult*. The *ARXResult* receives that result and presents it to the user, the class also includes a series of methods that gives useful information, such as, execution time or information loss

3.4. Project Folder Structure and Dependencies:



```
Project Explorer
▶ account-validator
▼ DataAnonymizer
  ▶ JAX-WS Web Services
  ▼ Java Resources
    ▼ src/main/java
      ▶ org.iiitb.DataAnonymizer.models
      ▶ org.iiitb.DataAnonymizer.resources
      ▶ org.iiitb.DataAnonymizer.services
    ▶ src/main/resources
    ▶ src/test/java
    ▶ Libraries
  ▶ JavaScript Resources
  ▶ Referenced Libraries
  ▶ Deployed Resources
  ▼ src
    ▼ main
      ▶ java
        resources
      ▼ webapp
        ▶ assets
        ▶ images
          views
        ▶ WEB-INF
          Anonymizer.html
          configs.html
          fileUpload.html
          index.html
          page-login.html
          page-register.html
          pages-forget.html
          sources.html
    ▶ test
  ▶ target
  ▶ pom.xml
```



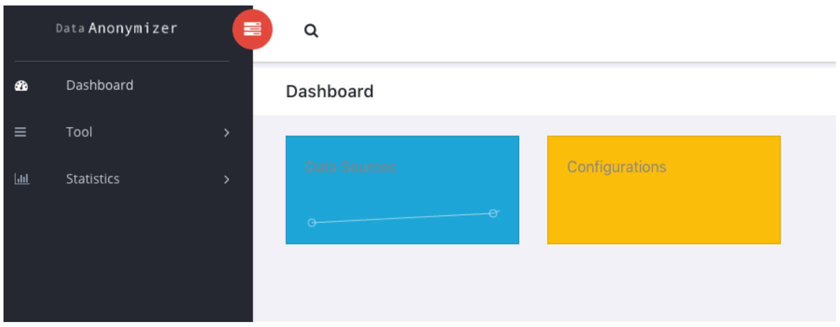
```
▼ Referenced Libraries
  ▶ commons-fileupload-1.3.3.jar - /Users/pranithreddy
  ▶ commons-io-2.6.jar - /Users/pranithreddy/Documer
  ▶ libarx-3.7.1.jar - /Users/pranithreddy/Desktop/PE/ai
  ▶ java-json.jar - /Users/pranithreddy/Documents
  ▶ mysql-connector-java-5.1.46-bin.jar - /Users/pranit
```

The data that is uploaded to the server or data anonymizer will reside in a directory named **Anonymization_resources** of root folder. Inside the directory it again contains folders named data sources, configurations, hierarchies for their respective files. Also the output folder contains the recent anonymized file.

4. Interface/User Manual:

The interface of an application should be intuitive in order to be used by the user. Hence, this web app consists of friendly interface, means can easily be understood and navigated by user.

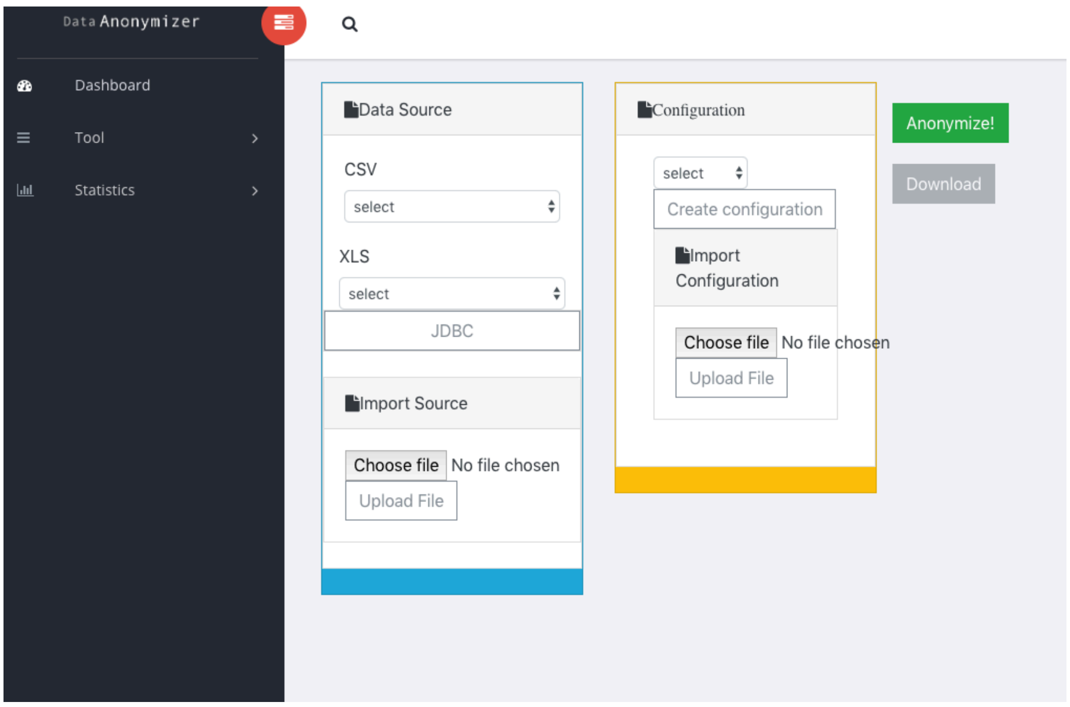
For anonymization of data, the user initially needs to be logged in using the login page. After login, he will be redirected to index page, which consists ***Dashboard*** containing two cards '***Data Sources***' and "***Configurations***'.



Data Sources: By clicking the data sources card the user will be redirected to the data sources page. It consists of all the data sources that are uploaded by the user and also has an option to add or delete any data source from the server.

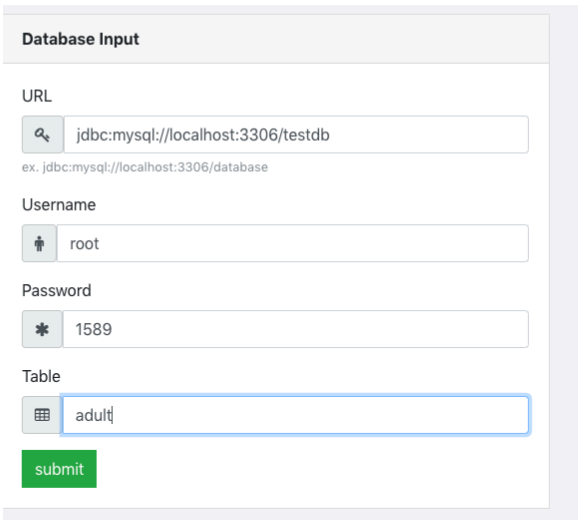
Configurations: By clicking the configurations card from the index page, the user will be redirected towards configurations page. It consists of all the configurations that has been created or uploaded by the user. Similar to the data sources even the user can create or upload or delete configurations.

To anonymize the data, the user needs to go the anonymised page by clicking the ***'Tool'*** tab on the left menu then clicking ***'Anonymizer'*** icon in the sub menu of tool. Then the page loads up with a wizard containing Data source and Configuration card's.



4.1. ***Data Source:*** The user can select which type of data source to be used, as of now there are three different type's that are being supported by this application **CSV**, **XLS**, **JDBC**. Then from the menu's displayed the user can select the data source, incase of JDBC the user needs to click on the "JDBC" Button. After that a form will be displayed in which the user needs to enter the details for database connection.

The form contains *URL*, *Username*, *Password* and *table* name which consists of data that should be used to anonymize. From the below figure it can be seen clearly as of how the URL should be followed with the database name. Here the database id 'testdb' and the table is 'adult'



Database Input

URL: jdbc:mysql://localhost:3306/testdb

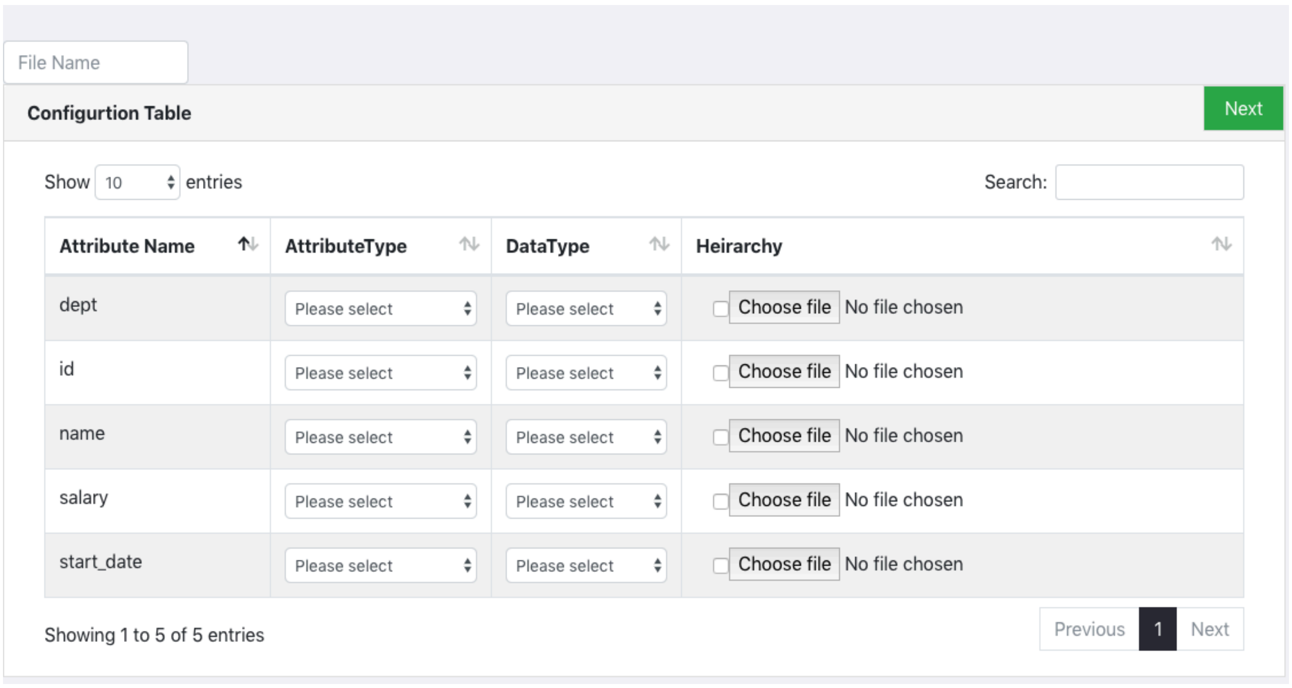
ex. jdbc:mysql://localhost:3306/database

Username: root

Password: 1589

Table: adult

4.2. Configuration: The user can select the configurations from the drop down menu or create a new configuration or can choose to upload from the system. For creating a new configuration the user needs to click 'Create Configuration' button. Then the user will be displayed a table consisting of the attributes(column's) from the respective data source that was selected. For each attribute's information a row will be given, so that user needs to fill in.



File Name

Configurtion Table

Show 10 entries

Search:

Attribute Name	AttributeType	DataType	Heirarchy
dept	Please select	Please select	Choose file No file chosen
id	Please select	Please select	Choose file No file chosen
name	Please select	Please select	Choose file No file chosen
salary	Please select	Please select	Choose file No file chosen
start_date	Please select	Please select	Choose file No file chosen

Showing 1 to 5 of 5 entries

At the top left corner the file name should be specified. For each attribute typically the attribute type should be selected i.e **SENSITIVE, INSENSITIVE, IDENTIFYING, QUASI IDENTIFYING.** Similarly for data type any of '**string**', '**int**' , '**dob**' should be selected. Then for uploading generalization hierarchy the user needs to upload the clicking 'choose file' button. After filling out the form the user needs click '**next**' button on the top right side of table. Then the interface will show up another form which contains information related to use of privacy model.

4.3. Privacy Model:

Typically most of the anonymization algorithms are based on Generalization and Suppression of QID(quasi identifier) attributes. They are,

k-Anonymity:

This algorithm requires that each QID tuple appear in at least *k-1* records, this will ensure at minimal, that released data processed with *k-anonymity* will be difficult to re-identify. The QIDs contains information that is more likely to find over the *dataset*, so this type of attribute is more vulnerable to *re-identification*. *k-anonymity* uses *generalization* and *suppression* methods. *Generalization* implicates replacing a value with a redundant but semantically similar value. *Suppression* involves not publishing a value at all.

Vulnerability- If the sensitive information is equal for all *k* individuals, then *k-anonymity* cannot protect that fact to be disclosed (*Homogeneity attack)*

l-diversity:

This algorithm requires a high entropy on the distribution of SAs(sensitive attribute) for each QID. Overall, *l-diversity* is effective, intuitive and solve most of the *k-anonymity* failures. Besides, a trusted privacy against attacks is used, even when collectors don't have any kind of information about the attacker's level of knowledge. The main idea behind *l-diversity*, is the well balancing dispersion of SAs between all the groups included on the *datasets*.

Vulnerability- Similarity attacks, that happens because it considers the diversity of SAs in the group, but it is not concerned with the semantic proximity of the values.

t-closeness:

Due to the fact that previous privacy models had some vulnerabilities, a new one emerged, the *t-closeness* algorithm. It requires that the distribution of a SA in any equivalence class must be similar to the attributes distribution in the overall *dataset*, this way, the chances of learning individual's information are lower. In order to introduce and manage gaps between values of SAs, *t-closeness* uses the *Earth Mover Distance* metric [10], receiving a precise distance between the two distributions.

Hierarchies:

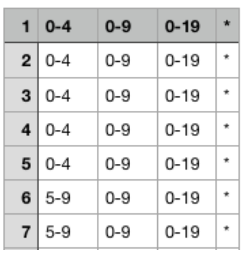
Defining the intervals is important in the configuration process, all the solutions below use this mechanism, but sometimes is not clear what their function is, neither the right way to configure them. So, the objective here is to explain what are these hierarchies, what are they used for and how to correctly define them.

Generalization hierarchies are typically used in *de-identifying datasets* with PII. Using this type of configuration will reduce the precision of attribute values, i.e., instead of having simple values, the data will be presented as intervals. Imagining an attribute containing the patient's age, from 0 to 100 years old, a hierarchical option, in this case, could be using decade intervals. This will result on ten levels, and the data will be presented as: "[0, 9]", "[10,19]" and so on. Defining bigger interval (fewer levels of hierarchies) will decrease the risk of *re-identification*.

Hierarchies are normally used for categorical attributes, to increase the utility of *anonymized datasets*, "categorisation" is often combined with tuple *suppression*, i.e., data records inconsistent with *privacy criteria* are automatically removed from the *dataset*.

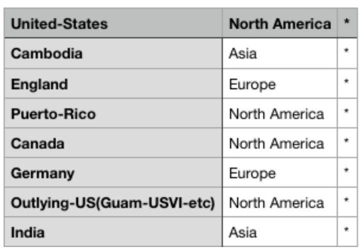
Data and *generalization* hierarchies can be imported from many different types, providing compatibility with a wide range of data processing tools.In this tool the hierarchies can be supplied by the user in the form of 'CSV' files, which each file contains the column headers specifying the level fo the hierarchy and rows contain intervals or generalised attributes.

adult_hierarchy_age



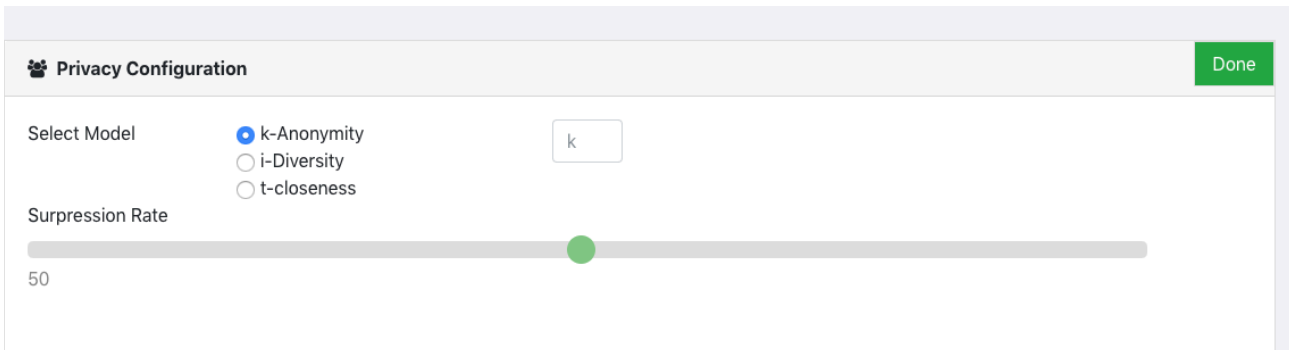
1	0-4	0-9	0-19	*
2	0-4	0-9	0-19	*
3	0-4	0-9	0-19	*
4	0-4	0-9	0-19	*
5	0-4	0-9	0-19	*
6	5-9	0-9	0-19	*
7	5-9	0-9	0-19	*

adult_hierarchy_native-country



United-States	North America	*
Cambodia	Asia	*
England	Europe	*
Puerto-Rico	North America	*
Canada	North America	*
Germany	Europe	*
Outlying-US(Guam-USVI-etc)	North America	*
India	Asia	*

As we discussed in the above, the user needs to select a privacy model that needs to be applied on the dataset. As of now it supports only 3 models such as ***k-anonymity***, ***l-diversity***, ***t-closeness.*** Along with the model the user also needs to mention the ***suppression rate*** that should be used while anonymising the data by using the slider. Up on clicking 'Done' on the top right side button, if the data is valid the system will generate a new configuration file in **xml** format. The user can select the newly generated configuration file from the configuration card.



Select Model [✓ k-Anonymity] i-Diversity t-closeness

k

Surpression Rate

50

4.4. Anonymize:

When the user has both Data source and configuration selected or in place, then by clicking the ***anonymize*** button the anonymization process can be initiated. If the process has completed without any error then the system will send an alert saying "success" otherwise it will return the error. After the anonymization the user can download the anonymized data by clicking the '***Download***' button which is below on the '***anonymize***' button. After the anonymization process the Data Source file will be automatically deleted from the server in order for the system to maintain privacy.

5. Future work:

There are lot of improvements that can be done to this web application from front-end to back-end. The below are some of them:

★ Improving Interface design- The design of this version is composed of single wizard which contains all the features. So, some of these features can be split into different pages. A lot of improvements can be done to the UI by creating better visuals such as 'loading feature' when a process is running, warnings, pop up windows instead of alerts.

★ Database storage- In this version the files uploaded by the user are being stored in a local storage due to ARX API in the server which is a kind of limitation. This can be solved by storing the files in a database instead of server. Also for better privacy indication i.e taking one more step and creating a in-memory database where the file is being stored in the RAM rather than disk.

★ Input and output views- When the file has been uploaded by the user, there is no option in the present system to view the input. This is same in the case of output too as it can only be downloaded and cannot be viewed through the application. Hence we can provide a results tab in which the results can be seen alongside with input and compared with it. The rows that had been modified should be presented in a different way(colour), indicating the user what was *anonymized*.

★ Risk analysis and utility- Presenting a risk analysis tab which contains the time spent or the level of information loss.An analysis on *re-identification* risk per attribute is presented, these values give an estimated percentage of an *re-identification* attack being successful only having access to one or more attributes data.

★ Hierarchy Creation- Similar to creating the configuration there can be an interface for each attribute to specify the intervals or levels or generalisations. Maybe an automatic system for setting the top and bottom values, this will help a lot for the user defining these intervals.

★ Privacy Models - More privacy models can be added to the application which makes it more complete and making the results more robust.

References:

1. ARX - Open Source Data Anonymization Software[https://github.com/arx-deidentifier/arx]

2. A Web Anonymizer Platform for Datasets with Personal Information [Christophe da Silva Ferreira]

3. Raghunathan B. (2008) The Complete Book of Data Anonymization - From Planning to Implementation. CRC Press Taylor & Francis Group, Boca Raton, Florida, USA.

4. Elmagarmid A. and Sheth A. (2008) *Privacy-Preserving Data Mining: Models and Algorithms*. Springer Science+Business Media, LLC, New York, USA.

5.ARX Data Anonymization tool[https://arx.deidentifier.org/development/api/]

6.A modified efficient global K-anonymity Algorithm [http://dpi-journals.com/index.php/JRST/article/viewFile/2718/1830]

7. Sweeney L. (2002) Achieving k-Anonymity Privacy Protection using Generalization and Suppression. School of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA.

8. Narayanan A. and Shmatikov V. (2008) Robust De-anonymization of Large Sparse Datasets. Proceedings of the 2008 IEEE Symposium on Security and Privacy, Pages 111-125, IEEE Computer Society Washington, Washington DC, USA.

9.GarfinkelS.(2015)NISTIR8053-De-Identification of Personal Information. National Institute of Standards and Technology, US Department of Commerce, Gaithersburg, Maryland, USA.

10. Zhang Q., Koudas N., Srivastava D. and Yu T. (2007) *Aggregate Query Answering on Anonymized Tables*. IEEE 23rd International Conference on Data Engineering, 2007, Istanbul, Turkey.