

DATA ANONYMIZATION PIPELINE

A PROJECT REPORT

Submitted by

MODI SHRESHTHA P

190130111081

In partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING

In

Electronics and Communication Engineering

GOVERNMENT ENGINEERING COLLEGE

GANDHINAGAR



Gujarat Technological University, Ahmedabad

May, 2023



GOVERNMENT ENGINEERING COLLEGE GANDHINAGAR

CERTIFICATE

This is to certify that the project report submitted along with the project entitled **Data Anonymization Pipeline** has been carried out by **Modi Shreshtha P** under my guidance in partial fulfilment for the degree of Bachelor of Engineering in Electronics and Communication, 8th Semester of Gujarat Technological University, Ahmedabad during the academic year 2022-23.

Prof Himanshu Nayak

Internal Guide

Prof. Tejaskumar Sheth

Head of the Department



CIN: U72200GJ2013PTC076894

Internship Completion Letter

Date: 01st May 2023

To Whomsoever ever it may concern,

This is to certify that **Shreshtha Modi** (190130111081) has completed her internship as a cloud developer at **Eternal soft solutions**

During her internship she worked on building a **data anonymization pipeline** and was exposed to python, AWS, docker, snowflake, powerBI

She was found to be honest, hardworking and inquisitive and we wish her all the best for her career

Best Regards

Nirav Shah

Director

Eternal Web Pvt Ltd





GOVERNMENT ENGINEERING COLLEGE GANDHINAGAR

DECLARATION

We hereby declare that the Internship/ Project report submitted along with the Internship/Project entitled **Data Anonymization Pipeline** Submitted in partial fulfillment for the degree of Bachelor of Engineering in **Electronics And Communication** to Gujarat Technological University, Ahmedabad, is a bonafide record of original project work carried out by me at GEC Gandhinagar. Under the supervision of **Prof. Himanshu Nayak** and that no Part of this report has been directly copied from any students reports or taken from any other source, without providing due reference.

Shreshtha Modi

ACKNOWLEDGEMENT

Firstly, I would like to thank **Mr Nirav Shah.** at **Eternal soft solutions** for giving me the opportunity to intern and learn at his organization. During my tenure at Eternal, I learnt a lot of invaluable technical and interpersonal skills

I am highly indebted to my faculty guide **Prof Himanshu Nayak** for his constant guidance and support throughout the internship and **Prof Tejas Sheth.** (Head of Department, Government Engineering College Gandhinagar) for his constructive criticism and mentorship throughout the internship

My sincere thanks to **Dr Shweta Dave** (Principal, Government Engineering College Gandhinagar) and the entire staff of Government Engineering College Gandhinagar for giving me the opportunity to intern at an organization and learn and for creating a holistic environment where I was able to learn and grow.

Lastly, I want to thank all my friends and colleagues who challenged me to push my limits and helped me with all of my queries. I would also like to thank all the researchers and open source developers for their efforts in crafting products that I could build upon and products which continue to help a lot of people all around the world

Modi Shreshtha Pragnesh

190130111081

ABSTRACT

Data is a crucial part of our everyday lives. With virtually every person having an online footprint on the internet, it is safe to say that an average person leaves behind a good amount of digital footprint. Something which can be used to trace back to the user and something which can tell a lot about the user. The given project aims at creating an end to end data anonymization pipeline using python. The given project takes in the data, the generalization hierarchies and the number of other indistinguishable values 'k' in the data and anonymizes the data. The data is anonymized in such a way that the statistical interference of the data is still maintained and so is the privacy of the data. With connection options like minds db and powerbi, users can analyze their data with ease and generate powerful and insightful information from the data. This project implements the privacy technique k-anonymity using various optimization algorithms such as mondrian, clustering. OLA, Greedy algorithm etc and then compares all the algorithms to find the best one across several popular large datasets

List Of Figures

Fig 1.1 Organization Chart	2
Fig 2.1 software development lifecycle	3
Fig 3.8.1 Static anonymization	11
Fig 3.8.2 Dynamic Anonymization	11
Fig 3.8.3 Process of Anonymization	12
Fig 3.9.1 Proposed solution 1	12
Fig 3.9.2 Proposed solution 2	13
Fig 4.4 A possible number of use cases of the current system .	19
Fig 4.6 Flow and features of the system	20
Fig 5.1 The overall flow design of the system	21
Fig 5.2 The user interface of the project	23
Fig 5.3 Interface after anonymization	24
Fig 5.4 The Flowchart of hierarchy generation	26
Fig 5.5 A simple Hierarchy generation	26
Fig 5.6 Flowchart for mondrian algorithm	27
Fig 5.7 Flowchart for tdg algorithm	29
Fig 5.8 Implementing K anonymity using OLA	30
Fig 5.9 Implementing clustering k anonymity	31
Fig 6.1 Snippet of code for generation in hierarchy	36
Fig 6.2 Implementation of mondrian	37
Fig 6.3 Implementation of clustering	38
Fig 6.4 Generalization Hierarchies	39
Fig 6.5 Data Anonymization	39
Fig 6.6 Data Anonymization	39
Fig 6.7 Data Anonymization	39
Fig 6.8 GUI Implementation	41

Fig 6.9 Comparison of various datasets and practices	43
Fig 6.10 Experimental setup	43
Fig 6.11 Data Before vs after anonymization	43
Fig 6.12 Loss metrics for various datasets	44
Fig 6.13 Initial GUI	45
Fig 6.14 Visualization using titanic dataset	46
Fig 7.1 F1 score for various models	49
Fig 7.2 Accuracy	49
Fig 7.3 Precision	50
Fig 7.4 Recall	50
Fig 7.5 Values of different k for mgm data using mondrian	51

List Of Tables

Table 3.4 Project Development Plan with goals and Milestones	8
Table 5.2 An overview of the various data sources supported, types of columns and values supported	24
Table 7.1 Test cases for GUI	47
Table 7.2 Test cases for the algorithm.	48

Abbreviations

Identifiers: Identifiers are the attributes in data which can be used to detect sensitive information of the data e.g name, Email address

Quasi-identifiers: Attributes of the data which are not sensitive on their own but can be used in combination with other sensitive attributes to reveal the information about the data

API: Application Programming Interface

GUI: Graphical user interface

OLA: Optimal lattice algorithm

SQL: Structured Query language

DB: Database

DP: DYnamic programming

K-ANONYMITY: Anonymization technique

SVM: Support vector machine, a machine learning algorithm

KNN: K nearest neighbor, a classification algorithm

XGBOOST: Gradient boosting technique

F1 score: A classification accuracy metric which is harmonic mean of precision and recall

PowerBI: A visualization algorithm

Table of Contents

Declaration	i
Acknowledgment	ii
Abstract	iii
List of figures	iv
List of Tables	v
List of Abbreviations	vi
Table of Contents	vii
Chapter 1 Overview of the Company	1
1.1 About Eternal	1
1.2 Scope of work	1
1.2.1 Principles of Eternal	1
1.2.2 Vision of Eternal	1
1.3 Overview of the Departments	1
1.4 Organization chart	2
Chapter 2 Solutions Lifecycle Process	3
2.1 Stages of the life cycle	3
2.1.1 Planning	3
2.1.2 Architecture Planning	3
2.1.3 Development	3
2.1.4 Verify	3
2.1.5 Test	3
2.1.6 Deploy	3
2.1.7 Cost Estimation	4
Chapter 3 Project Abstract	5
3.1 About Project	5

Declaration	i
3.2 Objectives	5
3.3 Project Scope	6
3.4 Literature Review	6
3.4.1 Technology Review	6
3.4.2 Algorithm Review	6
3.5 Project Planning	7
3.5.1 Project Development Plan and Implementation...	7
3.6 Cost Analysis and estimation	8
3.6.1 Development cost	8
3.6.2 Deployment cost	9
3.7 Project Excerpt	9
3.8 An introduction to data privacy and anonymization	9
3.8.1 Why is data anonymization needed	10
3.8.1.1 Types of anonymization	10
3.9 Solutions Proposed	12
3.9.1 Proposed solution 1	13
3.9.2 Proposed solution 2	13
3.10 Scope of the project and what it can and cannot do	14

Chapter 4 System Analysis	17
4.1 Problems and Weaknesses of the Current System	17
4.2 Requirements of New System	17
4.3 System Feasibility	18
4.3.1 Economical Feasibility	18
4.3.2 Feasibility with other systems	18
4.3.3 Research Feasibility	18
4.4 How Does the system fit in current objective of organisation	19
4.5 Implementation of the solution using current systems and technologies	20
4.6 Features and functions of the system	20
Chapter 5 System Design	21
5.1 User Interface design	22
5.2 Data Design	23
5.3 Hierarchy Generation	25
5.4 Algorithm Generation	27
5.4.1 The Mondrian Algorithm	27
5.4.2 The top down greedy approach	29
5.4.3 Implementing K anonymity using clustering	30
5.4.4 Implementing K anonymity using OLA	31
5.5 Inter connection generation	33
5.5.1 Connecting to powerBI platform via API	33
5.5.2 Connecting to mindsdb via API	34
Chapter 6 Implementation	35
6.1 About	35
6.2 Implementation Platform	35
6.3 Hierarchy Generation Implementation	36
6.4 Implementation of algorithms	37
6.4.1 Mondrian	37
6.4.2 Clustering	38
6.5 Applying Generalisation hierarchies to the data	39
6.6. Anonymizing the data	39
6.7 Structure of the product	40
6.8 GUI Implementation	40

6.9 Results	41
6.9.1 Comparison across datasets to find the influence of anonymization techniques	41
6.9.2 Data before vs after anonymization	44
6.9.3 GUI demo	45
6.9.4 Visualisation using powerBI	47
Chapter 7 Testing	46
7.1 Code Testing	46
7.1.1 GUI testing	46
7.1.2 Algorithm Testing	47
7.2 Hypothesis and analysis testing	48
Chapter 8 Conclusion	52
8.1 Overall analysis of the internship and the project	52
8.2 Problems encountered and possible solutions	52
8.3 Summary of the internship	53
8.4 Limitations and future work	53
Appendix	54
References	55

1 OVERVIEW OF THE COMPANY

1.1 ABOUT ETERNAL

Eternal soft solutions is a web development based consulting company located in Ahmedabad, India. Established in 2009, Eternal has established itself as a pioneer in the field of cloud based solutions provider. Eternal provides solutions for a wide range of businesses ranging from individual freelancers to small or medium sized startups to even global businesses

The name ‘Eternal’ means everlasting, or something that lasts forever and that meaning has been taken literally across every solution that eternal provides. The solutions built by eternal are scalable, robust and stand the test of time

1.2 SCOPE OF WORK

Eternal provides services across a vast variety of sectors such as hospitality, management, tourism, education and development and food and restaurant industry. They have worked with big giants like google, yamaha and honey splash

1.2.1 Principles of Eternal

- Openness and trust
- Teamwork and support
- Ingenuity

1.2.2 Mission of Eternal

Eternal’s mission is to delight our clients by satisfying their needs through effective utilisation of the best and the most promising new technologies and exceed their expectations by thriving on a continuous improvement of our Services, Business Processes, and Capabilities of its people.

1.3 OVERVIEW OF THE DEPARTMENTS

Eternal soft solutions has two divisions: Cloud development division and Web Development division

- The cloud development division takes care of designing the architecture of the project, determining the project scope, monitoring the health and status of the project and debugging errors. They give solutions centred mainly around the popular cloud providers AWS and GCP but occasionally add tools such as Docker, Jenkins and Kubernetes
- The web development division is a lot more hands-on and implements the solutions given by the cloud development team using front end and back end technologies such as PHP, MYSQL, Apache, NODEJS, React, HTML, CSS, Python, C++, AWS and GCP

1.4 ORGANISATION CHART

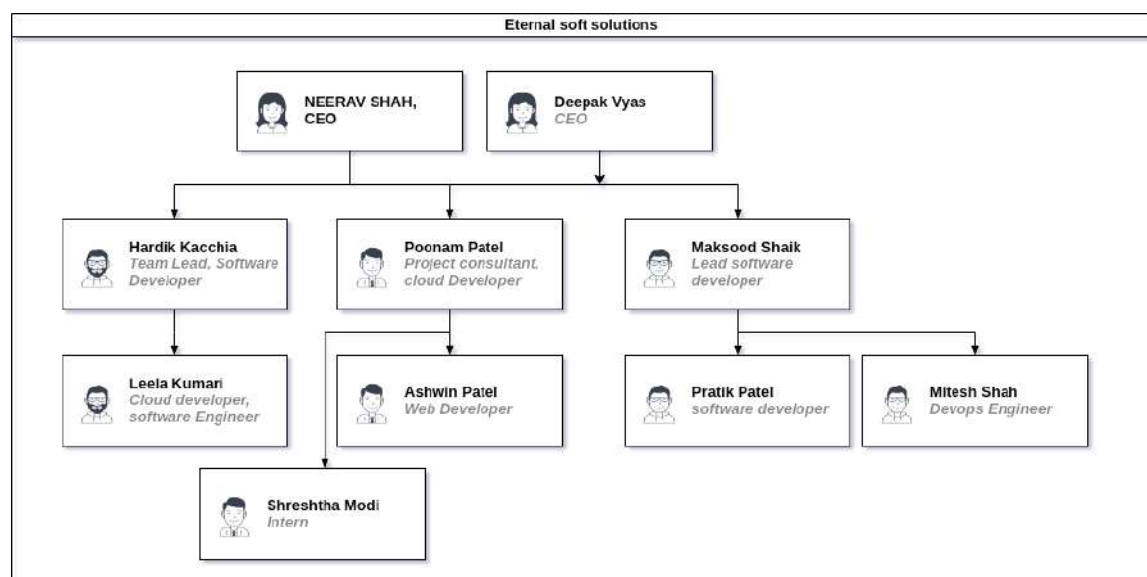


Fig 1.1 Organization chart of Eternal soft solutions

2 SOLUTIONS LIFE CYCLE PROCESS

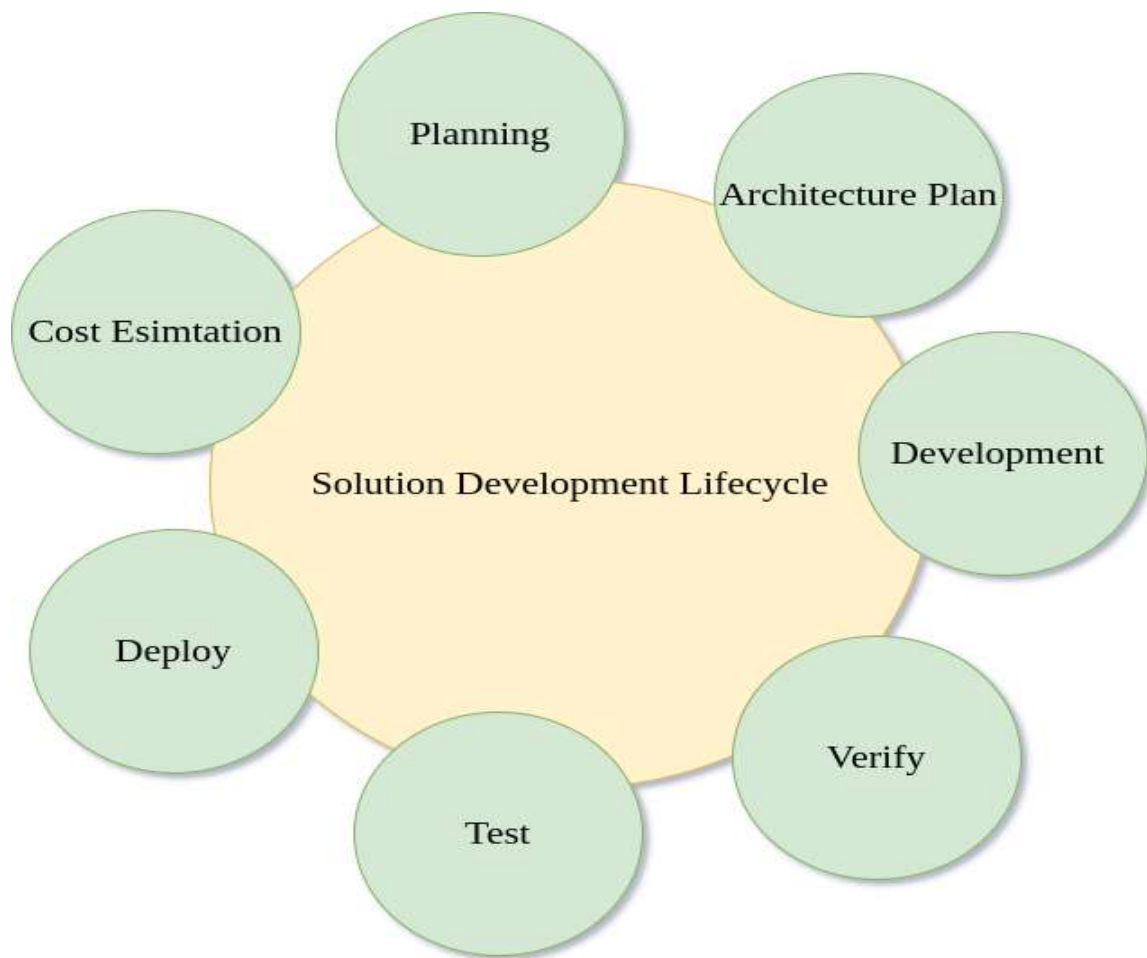


Fig 2.1 Software Development Lifecycle

2.1 STAGES OF THE LIFE CYCLE

2.1.1 Planning

Often the most crucial stage of the solutions lifecycle, planning involves talking and sitting down with the clients to understand their needs and design solutions accordingly

2.1.2 Architecture Plan

Architecture plan is often the courtesy of the solution architect, usually taken up by the most experienced person in the company who has deep knowledge of all the software development trends and best practices. They sit down and build a plan for the software and the tech stack being used so that they can serve their customers better

2.1.3 Development

This stage of the lifecycle refers to the actual coding and building of the solution according to the plan developed by the solution architect

2.1.4 Verify

This phase verifies if the solution is made according to the requirements of the clients. If the solution is verified, it moves on to testing

2.1.5 Test

Testing phase involves challenging your code in variety of domains to make sure that the code works as intended and is safe and optimal

2.1.6 Deploy

Deployment refers to the code being deployed on the client's servers and the actual machines in the real world

2.1.7 Cost Estimation

After the code has been running in the real world for a few iterations, next step is to estimate the cost based on previous usage and making sure that the solution stands the test of time at the cheapest rate possible

Thus, the cycle is not a linear process and a lot of the changes can be made at any point of the cycle

3. PROJECT ABSTRACT

3.1 ABOUT THE PROJECT

Data is a crucial part of every individual's daily life. According to a report by DOMO in 2020, it was estimated that every minute, the following data is generated on the internet:

- 347,222 Instagram stories are posted
- 500 hours of video are uploaded to YouTube
- 41.6 million messages are sent on WhatsApp
- 4.5 million videos are viewed on Snapchat
- 4.7 billion searches are made on Google
- 188 million emails are sent

It would not be a stretch to say that we live in the world of data. However, with a lot of data, comes a lot of responsibilities. It is becoming increasingly difficult to safeguard the identity of the user data and protect the sensitive information of the user online as a lot of tools already exist which can “connect” the different parts of data with one another to reveal information that should have been kept private. The given project suggests an end to end data anonymization pipeline using static anonymization techniques such as K-Anonymity, L-diversity and T-closeness to safeguard the user data. Unique thing about the proposed approach is that the data, after anonymization, still retains a lot of statistical significance and can be used to develop state of art research models and generate insights from the data.

3.2 OBJECTIVES:

There have been various goals while designing the solution for this project but the main idea boils down to making sure that the solution brings about a change in the field of data anonymization as there is need for implementation of more state-of the art methods keeping the modern database structure in mind

- To provide a solution such that the data of the clients is safeguarded and personal
- To design the algorithm keeping the security aspect of the data in mind and to make sure that none of the data gets leaked during any stage of the software development lifecycle
- To design a solution in such a way that it supports the clients native data structure
- To design the solution in such a way that the statistical nuance of the data is not lost

3.3 PROJECT SCOPE

The given solution is implemented in python 3.10.0 but can be easily replaced and modified to suit almost all the versions of python. Moreover, the additional option to combine the API with the UI integration opens up the scope for a lot more end users to benefit from the project

3.4 LITERATURE REVIEW

All of the given algorithms for the project have been implemented using python and some parts of implementation has also been done using Java

The project was developed using SCRUM and JIRA technologies where the ticket system has been prevalent

3.4.1 Overview of the technologies

Python: Python is an open source object oriented programming language that is used in variety of tasks such as Machine Learning, Data Science, Web Development, Cryptography, Game development etc

3.4.2 Review of the algorithms

K Anonymity was first introduced by L.Sweeney of the Carlie Melington university where she proposed a paper titled “K-ANONYMITY, A MODEL FOR PROTECTING PRIVACY”

This project takes a lot of inspiration from the pseudo code presented in that chapter. Moreover, we also acknowledge that since data attacking techniques have gotten a lot more sophisticated, we implement a lot of other versions than just the basic top-down greedy approach

Building upon the previous approach, mondrian (basic and advanced) are implemented based upon the paper titled Mondrian Multidimensional K Anonymity by the University of Madison

We also further develop the project by allowing the option to leave in insensitive attributes and sensitive names using relaxed mondrian approach given in Pycon 2019

3.5 PROJECT PLANNING

3.5.1 Project Development plan and implementation

Index	Milestone	Expected Completion date	Actual Completion date
1	Basic K-anonymity using top down greedy approach	20/02/2023	20/02/2023
2	Implementing Basic Mondrian algorithm	05/03/2023	09/03/2023
3	Implementing K Anonymity using clustering and NCP	08/03/2023	13/03/2023
4	Implementing relaxed Mondrian	12/03/2023	15/03/2023
5	Testing of the algorithms	12/03/2023	16/03/2023
6	Critical analysis of the algorithms	15/03/2023	16/03/2023
7	Hierarchy Generation	17/03/2023	20/03/2023
8	Finalising the datasets	21/03/2023	21/03/2023
9	Adding support for	14/04/2023	17/04/2023

	various integrations		
10	Creating User Interface	01/05/2023	29/04/2023
11	Testing and model building	04/05/2023	03/05/2023

Table 3.4 Project Development Plan with goals and Milestones

3.6 COST ANALYSIS AND ESTIMATION

3.6.1 Development cost:

The development cost associated with the project depends on the algorithm that we are trying to develop. The development cost comes with three factors:

- The technology being used:

The technology influences the cost of the project heavily as open source languages like python are a lot more likely to have support integration rather than closed source tools like MATLAB

- Algorithms being implemented:
 1. NP-HARD: NP-HARD problems are usually compute intensive and hence are difficult to optimise moreover they require significant prerequisites
- Features being added along with the algorithms:
More the features, more cost it will be to add them to the project

Development cost = (No of NP-HARD algorithms* features being added)

3.6.3 Deployment cost:

The cost associated with the project comes with the size of the dataset and the number of the iterations and the depth of the data associated. Moreover, snowflake offers 400 USD credits off for the first month. After that, the users are charged on a per-second basis and the charge depends on the tier that the user chooses for their snowflake account and the amount of the data they have

cost= (size of data)*seconds the data was running*69

3.7 PROJECT EXCERPT

The given project is a derivative of a client's project which was later revised and is now available as an open-source Github repository open for anyone to use and contribute to.

The client in question is a US-Based healthcare startup/hospital which has to deal with patients on a daily basis and hence a lot of legacy data. A good chunk of the data contains sensitive information about patients such as their names, social security number, zip codes, email-addresses etc. A lot of the data is potentially really helpful to the research community and can be used to derive scientific insights from the same. The task here is to build an end-to-end data anonymization pipeline which takes care of the data anonymization and data validation before releasing the data to the public. The proposed solution uses snowflake to store and retrieve legacy data and python script to anonymize the given patients data. The data is then stored to another public-facing database where users can integrate it with no-code machine learning and data visualisation platforms like Mindsdb and Powerbi (API support is provided for the same).. The given project implements anonymization algorithms mentioned above in python from scratch without needing the arx api/local arx server and effectively eliminates a lot of the constraints that came alongside using the module

3.8 AN INTRODUCTION TO DATA PRIVACY AND ANONYMIZATION:

Data privacy is a fundamental right of every individual in this day and age. With people's online presence increasing, data privacy is important now more than ever as handling large amounts of data and protecting the rights of an individual is a complex task

Data Anonymization is a form of privatising the data before making it public or sending it to the intended audience. Data anonymization refers to masking sensitive user data in a way that the identity of the user is maintained and if the data is released, the data can not be traced back to the user. Data anonymization is the tradeoff between usability of the

data in terms of statistical parameters and privacy of the users and to maintain the same statistical models and domain knowledge are combined to make sure that the anonymization is done properly.

3.8.1 Why is data anonymization needed?

One of the biggest examples which states the importance of anonymizing the data is the 2006 AOL data search leak. Here are some interesting statistics about the data leak:

- On 4th of August 2006, search data of approximately 650,000 users along with 20 Million search results were leaked
- The data was removed relatively quickly on 7th of August 2006
- The AOL did not identify the users in the data as the names of the users were not explicitly mentioned in the data
- However, a popular newspaper magazine called New York times were able to identify the users by cross referencing them with other sources like phone book listing

Another popular data breach is that of the Netflix where the data was leaked and the researchers at the university were able to trace it back to the users

Although proper care should be taken that the data does not get leaked in the first place, it is equally necessary to make sure that if the data gets leaked, the sensitive information is not revealed to the users.

To ensure the privacy of the data, a five parameter framework is used:

1. Ensure that the data is safe
2. Ensure that the people working on the data are safe
3. Ensure that the scope of the project is viable
4. Ensure that the proper compliant standards are set up to ensure safety in place
5. Disclose of the output data can be monitored to ensure that sensitive data is not leaked

3.8.1.1 Types of anonymization

There are two types of anonymization:

1. *Static*
2. *Dynamic*

Static anonymization: Static anonymization refers to the anonymization of the data all at once and then the data is being released to the public or the third party source or vendor. In static anonymization, often a subset of the original data is released after anonymization to the users. Popular static anonymization tools and softwares are ARX, Amnesia etc

Dynamic Anonymization: Dynamic anonymization refers to the anonymization of the data using queries. Often the full dataset is released to the public and the anonymization of the dataset is happening in real-time. Dynamic anonymization is considered more reliable than the static one because static anonymization needs to specify the anonymization techniques very carefully otherwise the subset of the data can be extracted multiple times to paint the picture of the actual data. Some popular dynamic anonymization techniques are: The popular R package diffpriv, Google's RAPPOR etc

Static Anonymisation

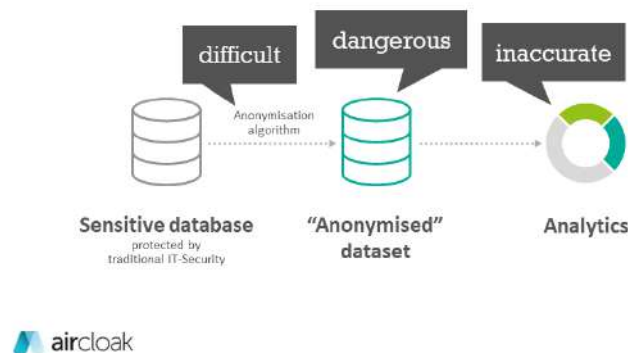


Figure 3.8.1 Static Anonymization

Interactive / Dynamic Anonymisation

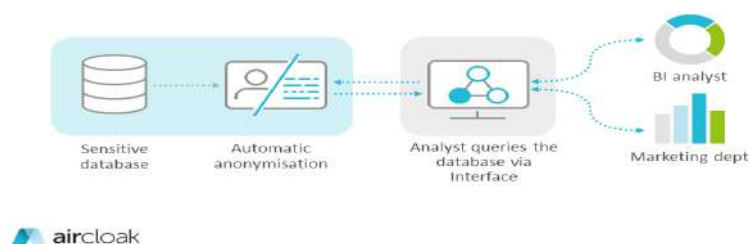


Figure 3.8.2 Dynamic Anonymization

Identifiers	Quasi-Identifiers			Confidential Attributes			Perturbed Quasi-Identifiers			Confidential Attributes		
SSN	Gender	Age	Zip Code	Hourly Wage	Political Affiliation		Gender	Age	Zip code	Hourly Wage	Political Affiliation	
432-55-1356	M	22	94024	\$34	Democrat	→	M	24	94***	\$34	Democrat	} k-Anonymized Records
123-70-4351	F	26	94305	\$42	Republican		M	24	94***	\$42	Republican	
471-65-3560	M	24	94024	\$18	Republican		M	24	94***	\$18	Republican	
351-34-2819	M	40	90210	\$40	Democrat		F	40	90***	\$40	Democrat	
241-41-7632	F	38	90210	\$41	Independent		F	40	90***	\$41	Independent	
501-33-2094	F	42	90213	\$37	Democrat		F	40	90***	\$37	Democrat	

Figure 3.8.3 The process of anonymization

3.9 SOLUTIONS PROPOSED

3.9.1 Proposed solution 1:

Anonymization of the data using aws databrew

Diagram

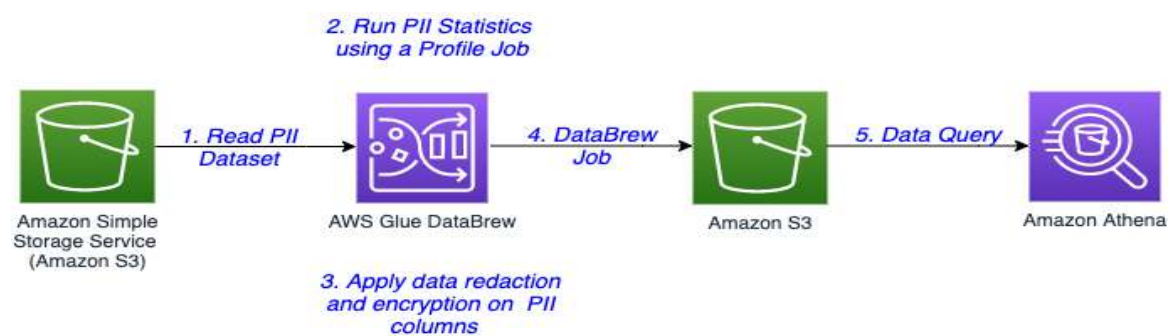


Figure 3.9 Architectural Diagram of Data Anonymization using AWS Databrew

Components:

1. **AWS S3:** As explained in the previous solution, s3 is an object based service used to store the data
2. **AWS Glue DataBrew:** Is a service by amazon that aids data scientists and machine learning engineers in cleaning and normalising the data.
3. **AWS Athena:** Amazon Athena is a serverless, interactive analytics service built on open-source frameworks, supporting open-table and file formats. Athena provides a simplified, flexible way to analyse petabytes of data where it lives.

Explanation:

the data in question to be anonymized is stored in an s3 bucket. Moreover, macie can be used to detect sensitive data from the already present data. The data from s3 is then loaded into the aws glue data brew which has jobs which mask the sensitive PII column-wise and after the data is masked, the masked data is then stored into s3 which creates a external table on top of athena

Feedback from the mentor and drawbacks:

Using aws macie to detect sensitive data might work for use cases where the user might have accidentally entered the sensitive data such as credit card information inside the s3 bucket. In our use case, the data that we want to mask is just not sensitive data, but the data that can be used in conjunction with other data to identify the users as well which is something where macie fails. Additionally, the aws data brew has limited support for masking the sensitive data and if the attacker knows the method or the technique using which the data was masked, it might be easy for the attacker to re-identify the data. Overall, we need a solution that is fast, has versioning capability, is able to handle legacy data and also uses a predefined algorithm to anonymize a data with a known schema while being cost optimised

3.9.2 Proposed solution 2

Data Anonymization Pipeline using Snowflake and airflow

Diagram

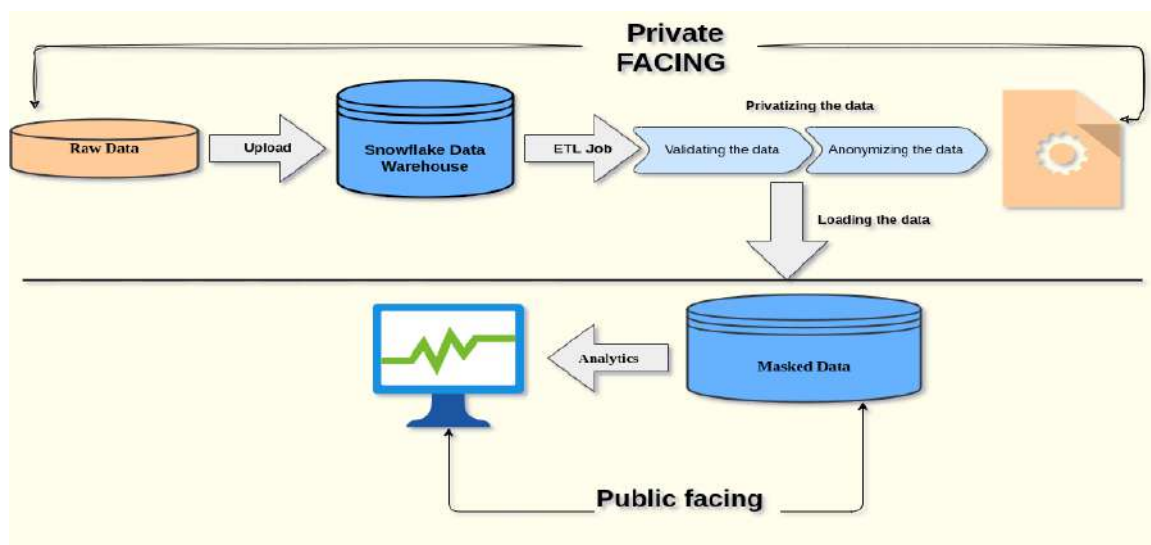


Figure 3.10 Proposed solution 2

Components:

Snowflake: Snowflake is a SaaS data warehouse solution designed solely for the cloud, it supports popular cloud providers like gcp, aws and azure. The snowflake can be used to build data warehouse and data lake right from your browser as snowflake is a managed service

PYARXAAS: Is a popular python module which is used as a static anonymization tool. Pyarxaas provides a wrapper to access functions for your local arx instance.

Apache Airflow: Apache airflow is an open source, etl tool that is used to automate the data loading and data transforming pipeline

Explanation:

The following solution provides an end to end pipeline for anonymizing data that gives users the flexibility to scale the solution as per their demand or data. Here is the process of pipeline:

1. Takes in the raw data either from the source or from a sql database. What makes the solution truly customizable is that snowflake can take in data from multiple sources Snowflake natively supports AVRO, Parquet, CSV, JSON and ORC hence data from pretty much any source can be taken
2. The raw data is then loaded into the snowflake to create a data warehouse. The access to the given snowflake is heavily restricted. This step is also customizable and scalable as if you want to give certain users access to the raw data warehouse,

you can simply assign their role access and they can be managed using snowflake organisation. Thus, reducing the costs and increasing the flexibility

3. After that, an ETL job is run on the data warehouse which validates the data if need, sets the hierarchy of anonymization for the data and actually anonymizes the data. The anonymization being used here is a function written manually and will be triggered after a certain time when the data is loaded in the data warehouse. Here, since we already know the project columns and all the details about the data being present inside we will use static anonymization instead of dynamic but dynamic anonymization can be used as well
4. The anonymized data is then stored in another snowflake data table or this can be also be connected to the cloud provider of your choice but an interesting thing about snowflake is that it really lets you create stunning visualisations using the snowflake console itself
5. The masked data which is not public facing can be used to generate charts, analysis and visualisation

Comments from the mentor and drawbacks:

One of the biggest advantages of this solution is the scalability in terms of compute and storage. Since snowflake is a hybrid between the shared nothing and shared everything architecture, it is really easy to scale up or down. The manual data anonymization gives the users greater flexibility to implement their own algorithm. However, one potential challenge to solve is trying to anonymize the data automatically without having the schema of the data. This is still a challenge since pyarxaas is a static anonymization algorithm

References:

<https://aircloak.com/data-anonymisation-software-differences-between-static-and-interactive-anonymisation>

3.10 SCOPE OF THE PROJECT: WHAT IT CAN AND CAN NOT DO

The data anonymization solution provided in the given project will have a humble impact as it can be modified and applied to a variety of sql databases, csv files, excel spreadsheets and dataframes. Moreover, support is also provided for snowflake tables and s3 instances. However, these will require a lot more configuration.

The given project not only suppresses the sensitive and identifying attributes in the dataset but it also takes care of the quasi-identifying attributes.

The project also takes care of both categorical and numerical variables. However, the user will need to define the categorical variables themselves

The given project can not generate the hierarchies for anonymization with sophistication, only basic functions are provided to generate the hierarchies. It is recommended that users generate their own hierarchies and then use the project to anonymize the data. Moreover, the project is more suited towards social science and healthcare data but can still be applied to other types of data

4 SYSTEM ANALYSIS

4.1 PROBLEMS AND WEAKNESSES OF THE CURRENT SYSTEM

Not a lot of work has been open-sourced for the general public to use and hence there is little to no awareness about the dangers of a data breach and how to protect your personal data online. Among the commonly prevalent data anonymization techniques such as roman cipher or substitution, masking and randomization, there a wide range of disadvantages:

- They can easily be decoded as they are relatively simple algorithms
- They can not be used for anonymization of sensitive data as they do not take quasi-identifiers into account
- The masked or randomised data loose their statistical significance as they can not be used as meaningful data

Although more sophisticated algorithms exist, they are mostly closed source and require significant time and or compute. However, several open source alternatives such as ARX often have limited capabilities and are often outdated

4.2 REQUIREMENT OF NEW SYSTEM

For the client, the patient's data holds the utmost priority as that data should not be seen by the malicious hackers and the technologies being used should be feasible and scalable without little to no overhead and should require little to no manual configuration. Clients also want complete control over their data and explicitly request that their data does not leave their system. Hence, it made the most sense to design the given solution

4.3 SYSTEM FEASIBILITY

4.3.1 Economical Feasibility

Economical feasibility of the system refers to the analysis of the finances of the current system. Whether or not after all costs considered the solution will be economically feasible or not. The beauty of the given solution is such that much of it's feasibility lies with the user and the features that they want to use. E.g if the user is an avid snowflake user with large amounts of data in there, standard snowflake costs will apply regardless.

However, the system is a lot more economically feasible E.g We convert all floating point values to Integers and hence the storage required for that column will be effectively reduced in half

4.3.2 Feasibility with other systems

The current solution is especially feasible with other systems as it supports a variety of sql databases and has snowflake integration support thus making it really easy for the users to import and export data. Moreover, it stores all the data at the grassroots level in the form of a pandas dataframe hence there is no need for conversion when using the data for analysis

4.3.3 Research Feasibility

The given system implements the data anonymization pipeline using the state-of-art methods and all of the implementation is open source meaning the users can add modify, improve and contribute to the development of the algorithms

4.4 HOW DOES THE SYSTEM FITS IN THE CURRENT OBJECTIVE OF THE ORGANISATION

The current system integrates really well with the overall objective of the organisation as this solution can be used by managers and the people at the upper management to preserve anonymity when asking for feedback and constructive criticism. Moreover, it can also be used to release the impressive feats achieved by the company without breaking the company-client confidentiality



Fig 4.4 A possible number of use cases of the current system with the objective of the organisation

4.5 IMPLEMENTATION OF THE SOLUTION USING PRESENT SYSTEM AND TECHNOLOGIES

The system is implemented using python, snowflake, powerbi, Minds Db and SQL

- Python: The reason for choosing python over any other languages such as Java or C++ was because the client has all of their systems running in python and hence it provides seamless integration. Moreover, a lot of Machine Learning and analysis has been traditionally based on python hence designing the system in python would reduce the learning curve required for anonymizing the data
- Snowflake: Snowflake is a cloud as a service platform which provides data warehousing capabilities. A lot of organisations use snowflake these days and it supports a variety of data types and data integration platforms
- POWERBI: PowerBI is a tool by microsoft that is used to visualise the data automatically. With the help of power BI, we can generate beautiful charts and visualisations that would have been difficult otherwise
- Minds DB: Minds DB is a coming of age database solution that takes in data in the form of “AI Tables” and allows you to use pretty much any machine learning algorithm using a simple SQL query. Integration of this can be used potentially to automate the machine learning modelling process as well and allowing the developers to spend time on the things that really matter

4.6 FEATURES AND FUNCTIONALITY OF THE SYSTEM

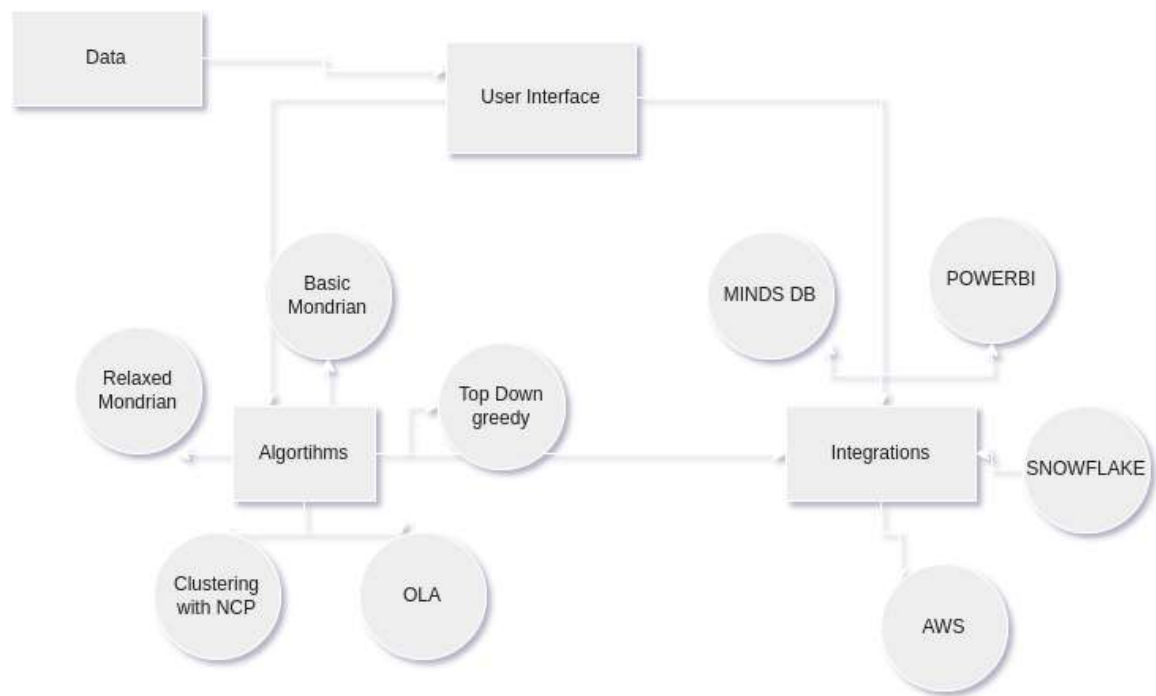


Fig 4.6 The flow and features of the system

Here are some of the salient features of the system:

- Is versatile, can be recalibrated to suit your business needs
- Supports a variety of analysis tools
- Stores and uses data in standard formats such as SQL tables and pandas dataframes
- Is open source
- Can be tweaked to meet a variety of data needs
- Can be extended to support a variety of services and softwares

5 SYSTEM DESIGN

System design and planning refer to the process of creating a blueprint or a roadmap for a project. It involves identifying and defining the requirements, creating an architecture, and designing the various components of the system that need to be built.

System design and planning are crucial before making a project because they help to ensure that the project is well thought out and well organised. It helps to identify potential problems or challenges that may arise during the project's implementation and allows for the necessary adjustments to be made in advance. It also helps to identify the scope and complexity of the project and determine the feasibility of its implementation within the given constraints, such as budget, time, and resources.

My project mainly has three components: The user interface, the algorithms and the APIS to connect to external sources. Out of all the three the user is only required to interact with the user interface hence it made the most sense to make the user interface as intuitive as possible

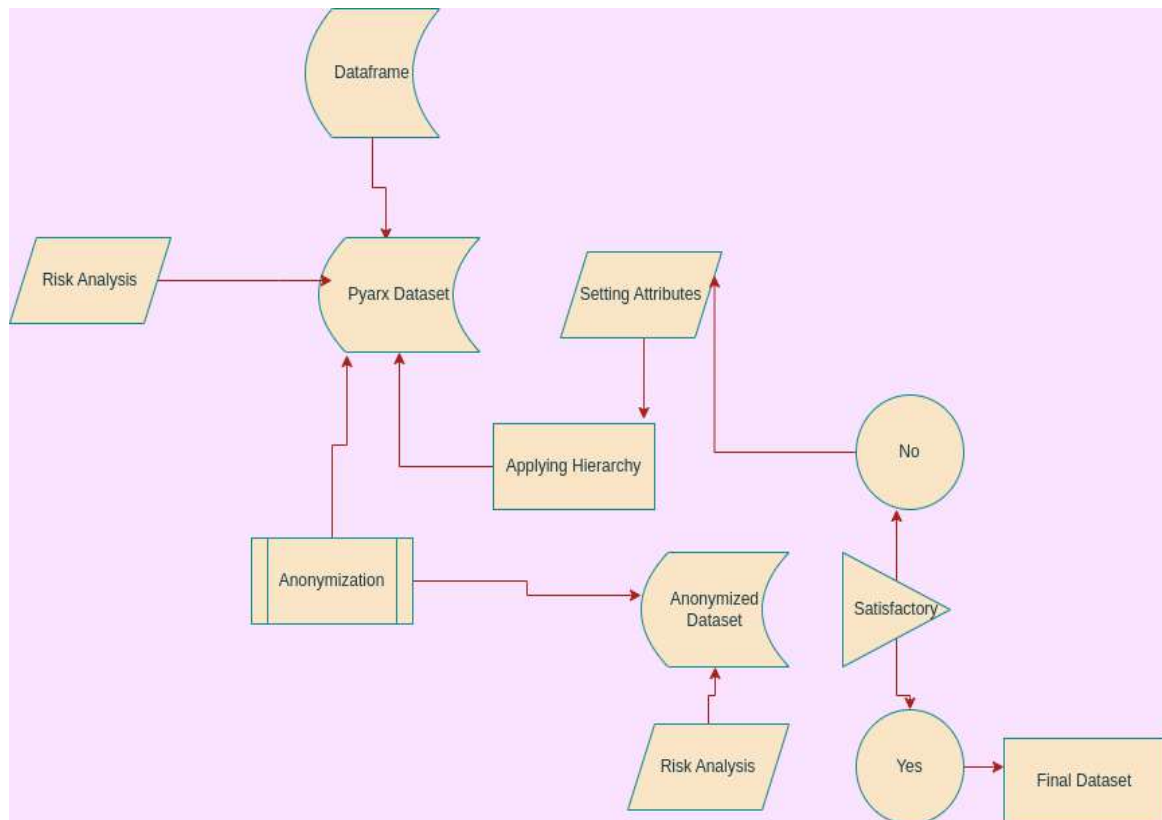


Fig 5.1 The overall flow and design of the system

5.1 USER INTERFACE DESIGN

Before starting with the user interface I wanted to identify the components that users want to access. These components are the parameters that the user should be able to change in their anonymization such as data being uploaded, the value of k , the anonymization metric etc. Hence my plan was to include them and then let my code do the rest of the work. Here is what the initial interface looks like when the user first accesses the project

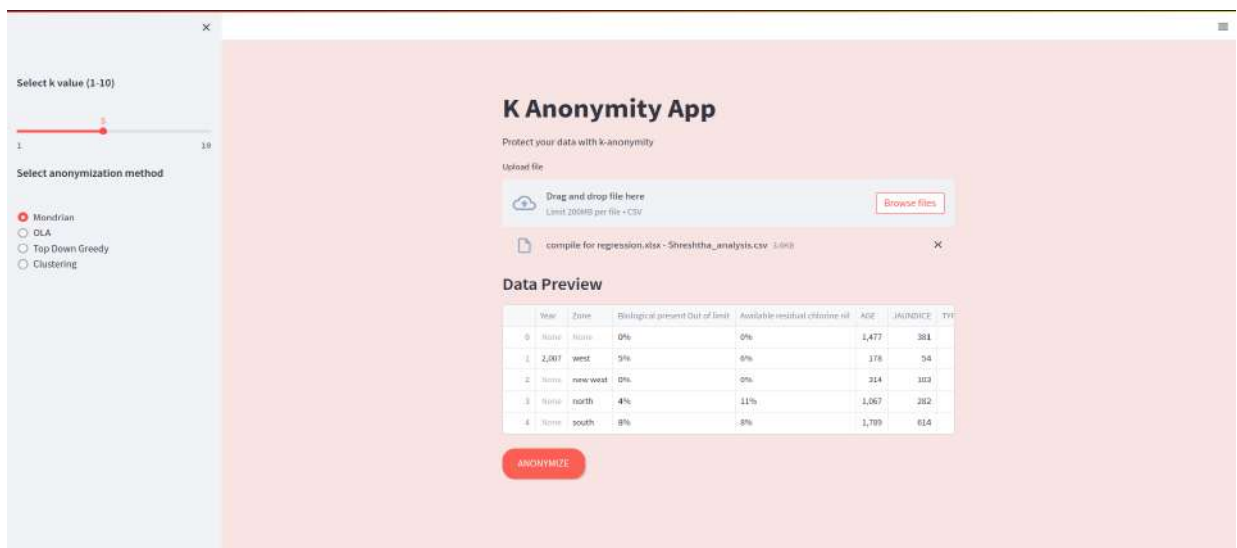


Fig 5.2 The user interface of the project

As it can be seen, when the user first uploads the dataframe, it generates a data preview to get a better idea of the data and lets the user select the method of anonymization and the value of k

After clicking on the anonymize option the interface then changes to show the anonymized data and asks user if they want to analyse their data and also gives the option to connect to external apps

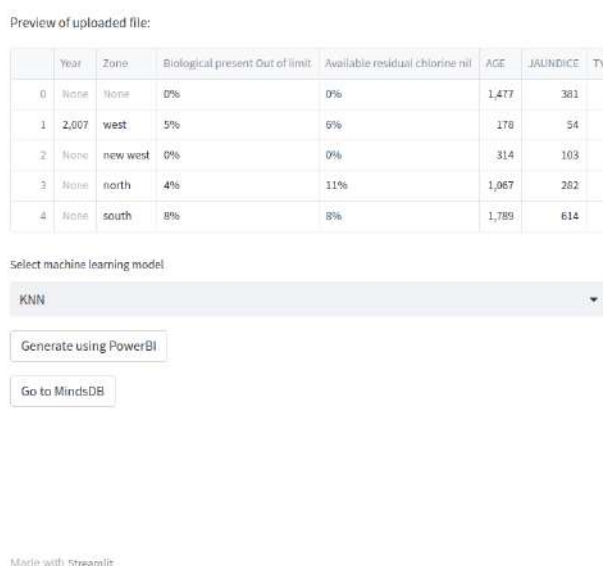


Fig 5.3 Interface after anonymization

5.2 DATA DESIGN

Data design refers to the schema or the system which is used to define the structure of the data that is used for anonymization

<i>Serial NO</i>	Data Source	Columns	Data Types supported
1	csv	Categorical, numerical, date time	Int, char, boolean, date time
2	xlsx	Categorical, numerical, date time	Int, char, date time, boolean
3	sql	Categorical, numerical, date time	Int, char
4	Pandas dataframe	categorical, numerical, date time, series	Int, char, date time, boolean
5	Nosql db	N/A	N/A
6	json	Categorical, numerical	Int, char

Table 5.2 An overview of the various data sources supported, types of columns supported and the values of the column supported

5.3 HIERARCHY GENERATION

Hierarchies refer to a set of categories or levels that can be used to generalise sensitive attributes of a dataset. The purpose of hierarchies is to provide a way to anonymize the data while preserving its usefulness for analysis.

Hierarchies are important in k-anonymity because they allow the data to be transformed in a way that protects the privacy of individuals without sacrificing the quality or accuracy of the data. By replacing specific values with more general categories, such as age ranges or zip code regions, the data can still be analysed for patterns and trends without revealing personal information about individuals.

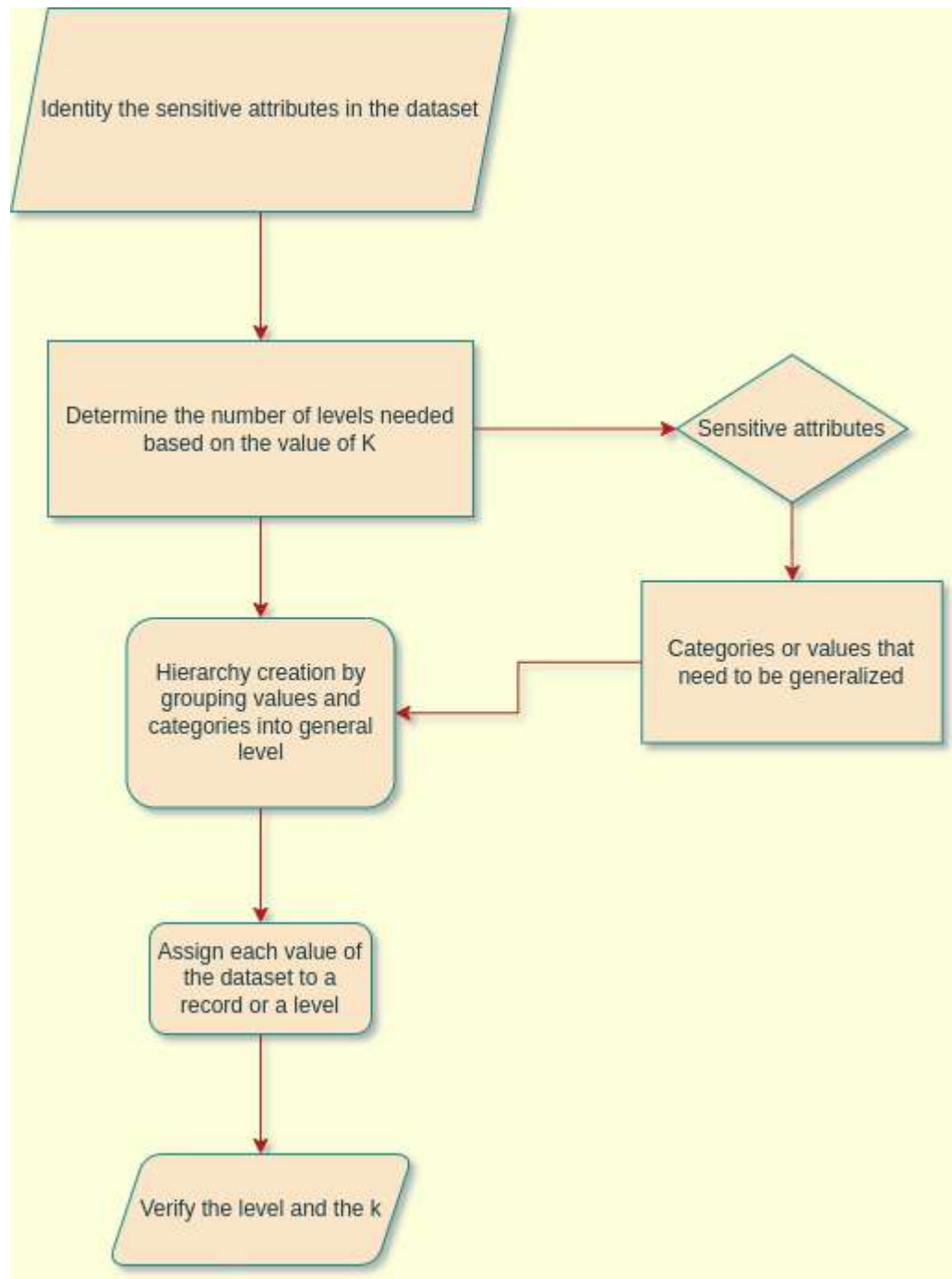


Fig 5.4 The flowchart of the hierarchy generation

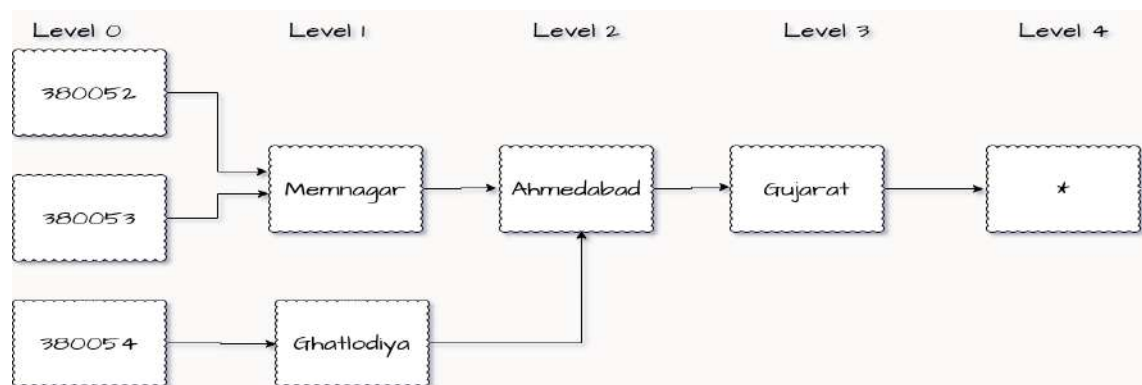


Fig 5.5 A sample hierarchy generation

5.4 ALGORITHM GENERATION

K-anonymity is a privacy-preserving technique used to protect sensitive information in a dataset while allowing analysis and data mining. The goal of k-anonymity is to ensure that no individual in the dataset can be identified from the information disclosed, while still maintaining the utility and accuracy of the data for analysis purposes.

K-anonymity works by generalising or suppressing certain identifying attributes in the dataset, such as names or addresses, to make it more difficult to link a record to a specific individual. To achieve k-anonymity, a dataset must be transformed so that each record is indistinguishable from at least k-1 other records in the dataset. This means that even if an attacker were to obtain the released data, they would not be able to identify any individual's sensitive information with certainty.

K-anonymity is important because it can protect sensitive information in datasets from being used for unauthorised or malicious purposes. This technique is commonly used in medical, financial, and governmental contexts, where the data can contain sensitive information that must be protected to comply with regulations or to maintain privacy. By using k-anonymity, organisations can minimise the risk of data breaches and protect the privacy of individuals whose data is being collected and analysed.

The project implements k-anonymity technique using several algorithms such as:

5.4.1 The Mondrian algorithm

The Mondrian algorithm is a privacy-preserving data anonymization technique that achieves k-anonymity by partitioning the dataset into smaller subsets or partitions.

The algorithm works by recursively partitioning the dataset along its attributes, such that each partition contains at least k records and the values within each partition are as homogeneous as possible. Each partition is then generalised by replacing the values of

some attributes with ranges or categories, while keeping the remaining attributes unchanged.

The partitioning process is guided by a metric that measures the degree of homogeneity of the records within each partition. The metric can be defined based on different criteria, such as information loss or distance between records.

Pros:

- Is computationally efficient
- Can handle both categorical and numerical values
- can achieve high levels of k-anonymity while preserving the utility of the data for analysis purposes.

Cons:

- Can get stuck into local optimum
- Not suitable for complex datasets
- Does not give the option to leave sensitive attributes in

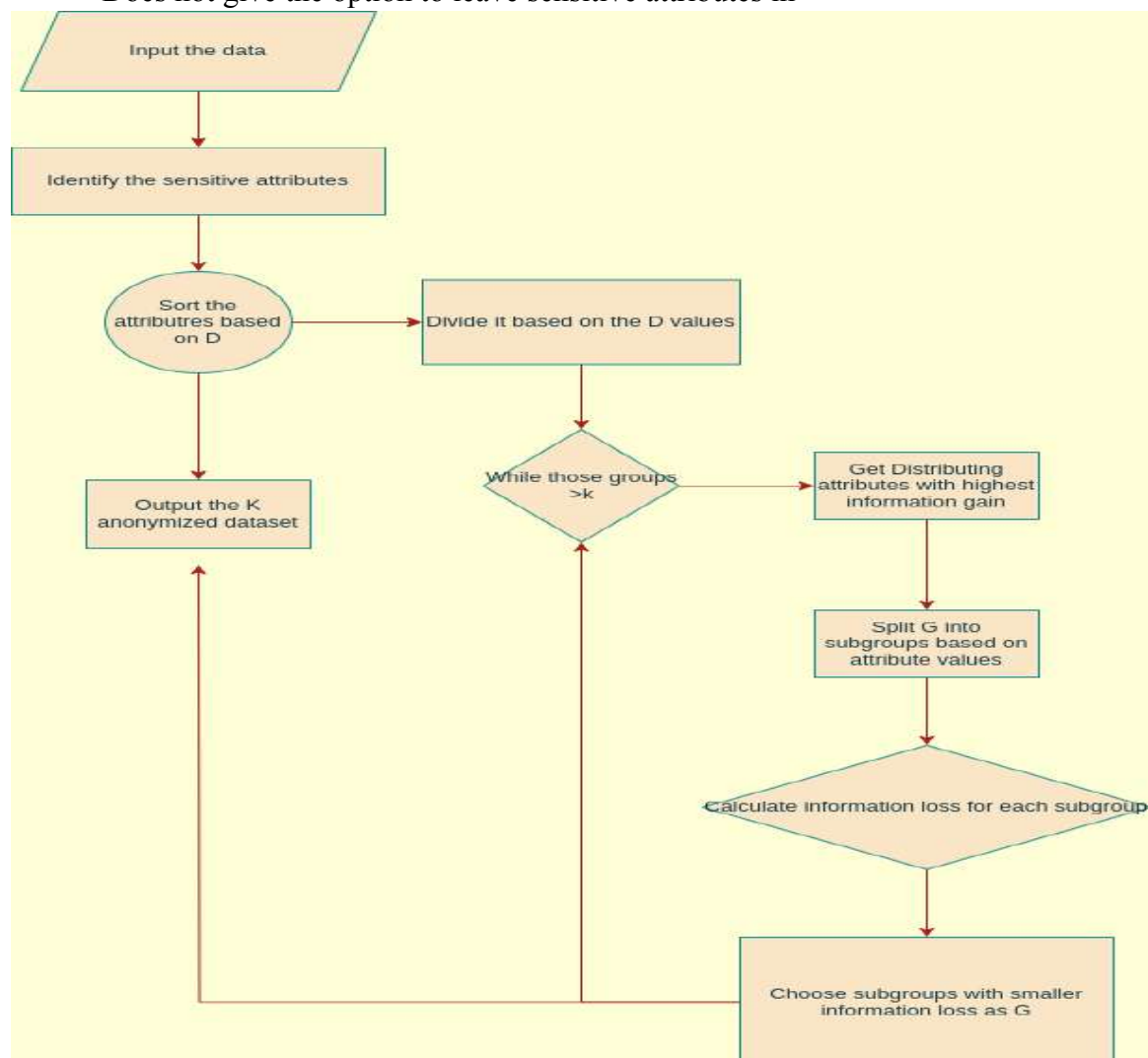


Fig 5.6 Flowchart for implementing the mondrian algorithm

5.4.2 The Top-Down Greedy Approach

This approach starts with the original dataset and recursively partitions it into smaller subsets or groups of records, until each group satisfies the k-anonymity requirement. The partitions are created by selecting the attribute that provides the maximum reduction in the number of unique values, and then applying a generalisation or suppression method to that attribute.

Pros:

- Simple
- Efficient
- Flexible

Cons:

- Risk of information loss
- Limited scalability

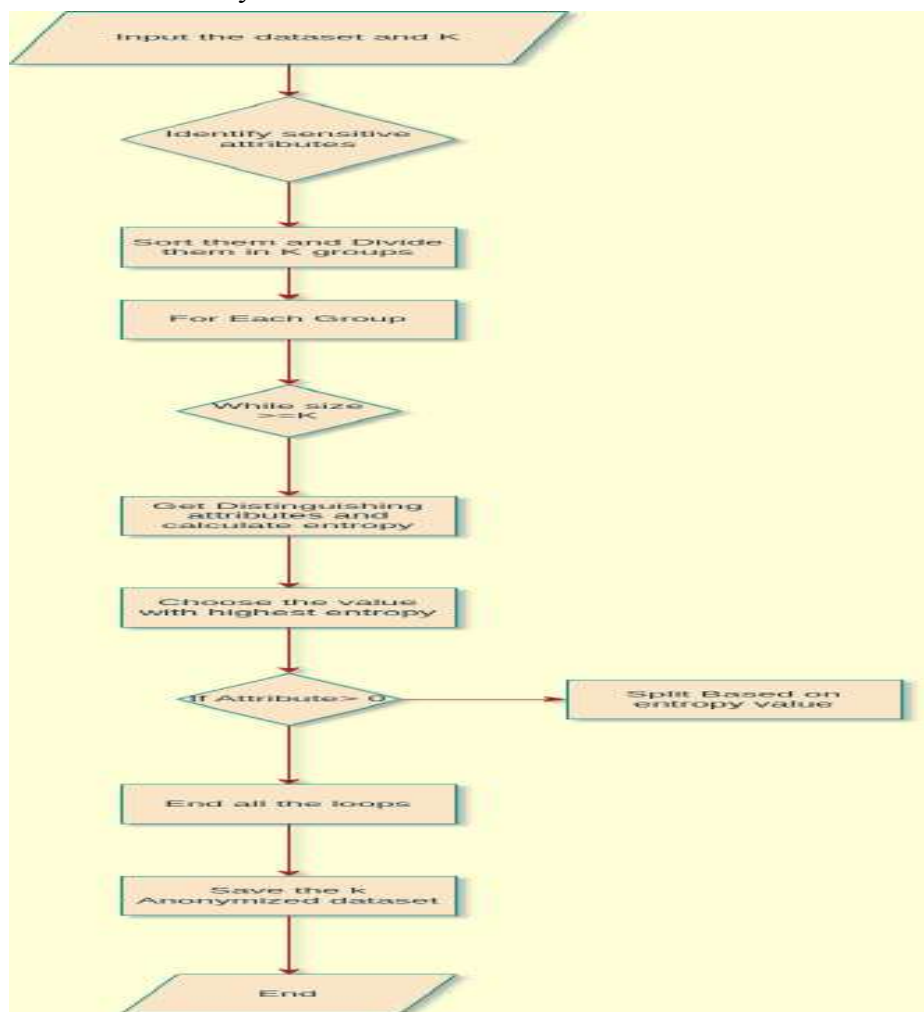


Fig 5.7 Flowchart of the pseudocode of top down greedy algorithm

5.4.3 Implementing K anonymity using the clustering algorithm

This approach is done using grouping together records that are similar in terms of their attribute values, while ensuring that each group satisfies the k-anonymity requirement. The normalised cost penalty function is a common cost function used in clustering-based k-anonymity techniques.

In this approach, the clustering algorithm is modified to take into account the k-anonymity requirement by penalising clusters that contain fewer than k records. The cost function takes into account both the intra-cluster distance (i.e., the similarity between records within a cluster) and the inter-cluster distance (i.e., the dissimilarity between clusters), and aims to minimise the overall distance while satisfying the k-anonymity constraint.

Pros:

- Fast
- Scalable

Cons:

- Sensitive to parameter training
- Risk of information loss
- Limited security

5.4.4 Implementing K Anonymity using OLA

In this approach, the original dataset is first sorted according to a specific attribute, and then each record is assigned a new value based on its rank or position in the sorted list. The rank-based values are then perturbed or randomised to add noise and prevent attackers from inferring the original attribute values. Finally, the dataset is grouped into k-anonymous partitions based on the perturbed rank-based values.

Pros:

- Preserves data utility
- Easy to implement
- Provides strong privacy boundary assurance

Cons:

- Limited scalability
- Limited flexibility
- Vulnerable to certain types of attacks such as attacks which exploit the correlation between attributes and distribution of the attributes

Thus, all of the above algorithms each have their own set of pros and cons and each of these can be used for variety of datasets

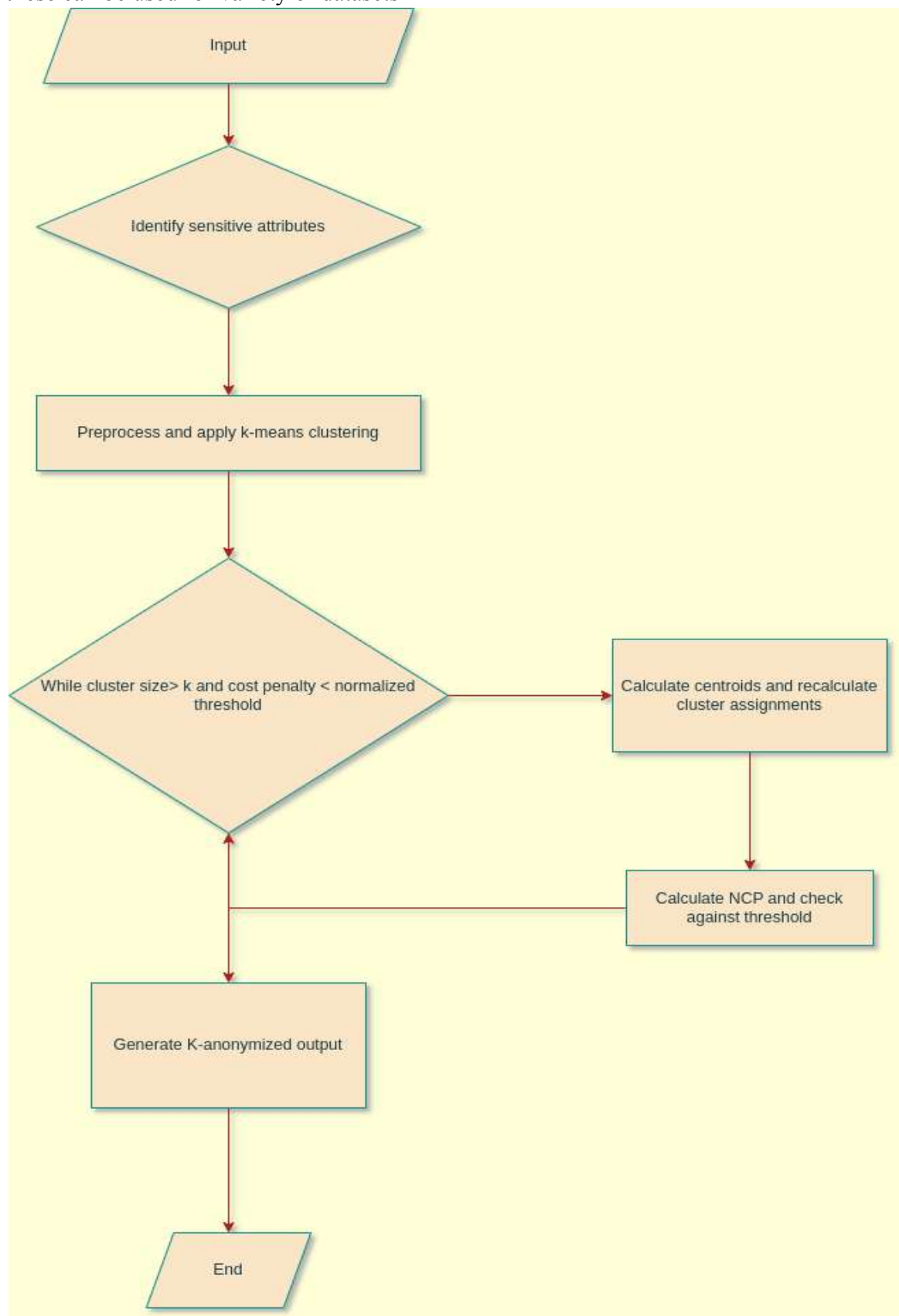


Fig 5.8 Flowchart of implementing K anonymity using clustering

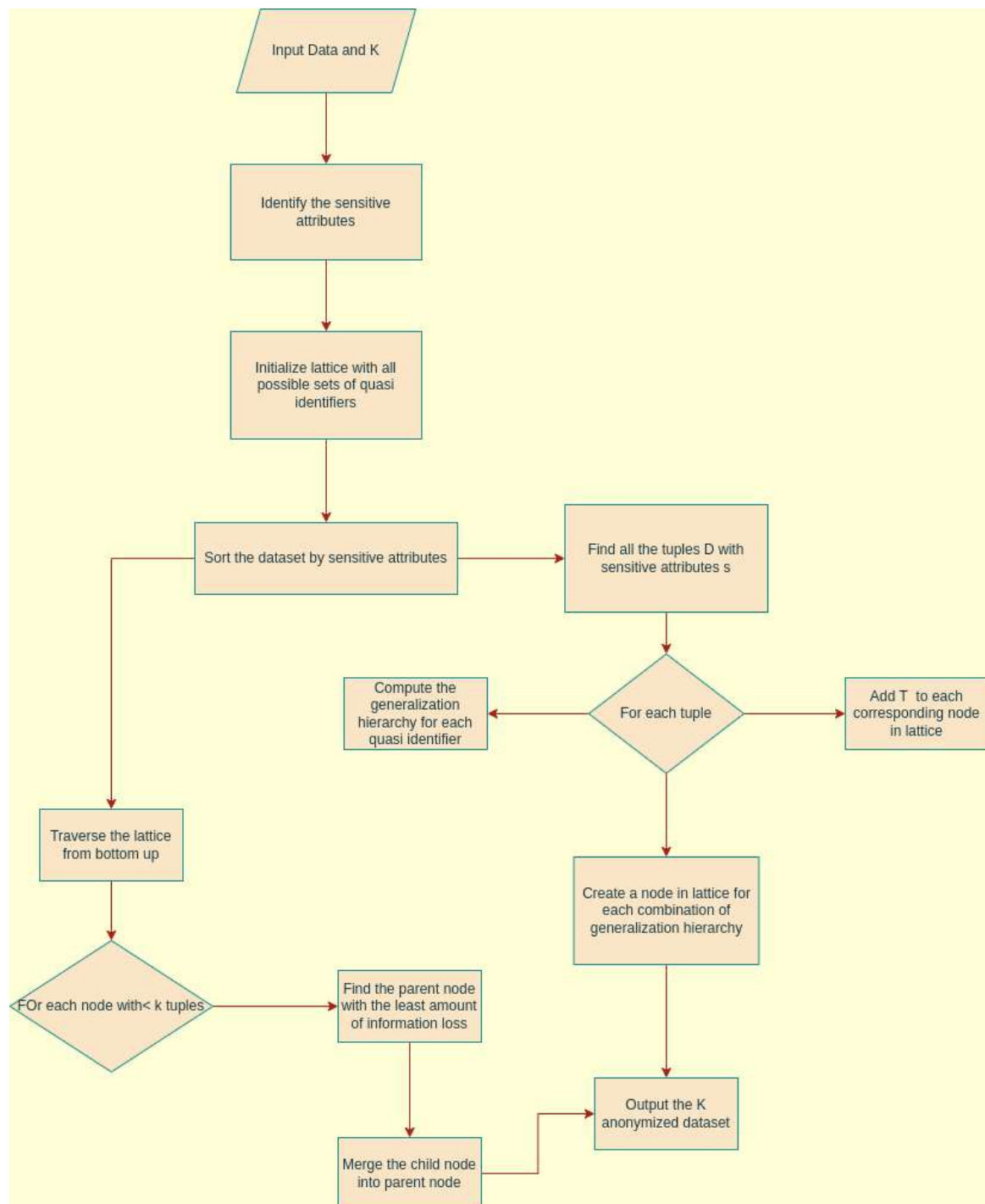


Fig 5.9 Implementing K anonymity using OLA

5.5 INTERCONNECTIONS GENERATION

5.5.1 Connecting to the PowerBI platform via the api

```
df = pd.read_csv('your_data.csv')
json_data = df.to_json(orient='records')

api_url = 'https://api.powerbi.com/beta/your_api_endpoint'
headers = {'Content-Type': 'application/json'}
response = requests.post(api_url, headers=headers, json=json_data)

print('Data sent successfully!' if response.status_code == 200 else
      f'Error sending data: {response.text}')
```

The given pseudocode converts the data in json format, gets the url and then sends the json data over to that url

The given function really extends the functionality of the project as the user can directly use this and generate powerful dashboards using powerBI

5.5.2 Connecting to Mindsdb via api

Mindsdb is a no-code machine learning tool which allows the users to analyze the data using sql queries

```
df = pd.read_csv('your_data.csv')

api_url = 'http://localhost:47334/predict'
model_name = 'your_model_name'

payload = {'data': df.to_dict(orient='records'), 'model_name':
model_name}
```

```
predictions = requests.post(api_url,  
                             json=payload).json()['data'][0]['predicted_values']  
  
print(predictions)
```

The given pseudo code reads the data, then the host api endpoints of minds db, it converts the dataframe to a dictionary and model name takes in the machine learning model that you want to use

6 IMPLEMENTATION

6.1 ABOUT

The given chapter provides detail about all the implementation of the pseudocode and the formation of the overall project system

6.2 IMPLEMENTATION PLATFORM

The given system has been implemented using python, an IDE called Visual Studio code and Jupyter Notebook in a linux environment. Since the iterations and computations can get tedious while making the project, a desktop with at least 20 GB of RAM is used and recommended

6.3 HIERARCHY GENERATION IMPLEMENTATION:

```
def read_hierarchy_from_dataframe(df):  
    headers = list(df.columns)  
    gen_hier = []  
    for header in headers[1:]:  
        gen_hier.append({})  
    for row in df.itertuples(index=False):  
        for i, value in enumerate(row[1:]):  
  
            if i == 0:  
                parent = value  
            else:  
                try:  
                    gen_hier[i-1][value].append(parent)  
                except KeyError:  
                    gen_hier[i-1].update({value: [parent]})  
    return gen_hier
```

```

hierarchy=pd.read_csv("/home/shreshtha/Downloads/agehier.csv")
my_hierarchy = read_hierarchy_from_dataframe(hierarchy)

```

The given below code reads the hierarchy files and sends them to anonymization from the hierarchy folder

For the cases where hierarchy is not defined manually the code generates them for you. Here is the code snippet for what it looks like:

```

10
11 def hierarchy(path: str, q1_name: str) -> list:
12     """Read q1 hierarchy from file.
13     filenames follow a specific format: gee_hier_q1_name.csv
14
15     :param path: Path to where the hierarchy file is stored
16     :param q1_name: Name of the q1 (header in dataset)
17     :return: list: List of function argument for that type of generalization
18     """
19     return [substitution, read_gen_hierarchy(path, q1_name)]
20
21 def age(data, level):
22     """Transform age data to a predefined generalization state.
23
24     :param data: List or string: Data containing one or more age values
25     :param level: int: Generalization step to which the data should be transformed
26     :return: list of generalized age data
27     """
28     # Use the generic function "segmentation" with predefined arguments
29     return segmentation(data, level, 1, 100, [5, 10, 20, ""])
30
31 def segmentation(data, level, min_num, max_num, div_list):
32     """Transforms numerical data to a segmented state
33
34     Parameters:
35     data: list or string
36         Data containing one or more numerical values
37     level: int
38         Generalization step to which the data should be transformed
39     min_num: int
40         Start of numeric range
41     max_num: int
42         End of numeric range
43     div_list: list
44         Contains value in what range data should be grouped for each generalization step
45     Returns:
46     list of generalized numerical data
47     """
48     ret = []
49     # Check if data is already a list/range or if it is a single value
50     if not isinstance(data, list) and not isinstance(data, range):
51         values = [int(data)]
52     else:
53         values = list(map(int, data))
54     seg = div_list[level]
55     # Check if the last level is not an integer segmentation and thus a substitution
56     if len(div_list) == level and not isinstance(seg, int):
57         return [seg]
58

```

Fig 6.1 Snippet of the code for generation of hierarchy

6.4 IMPLEMENTATION OF THE ALGORITHMS

6.4.1 Mondrian

1. Load your data into a pandas dataframe.
2. Identify the sensitive attributes in your data that you want to protect.
3. Define the hierarchy for each sensitive attribute. This can be done using either the generalisation hierarchy or the value hierarchy approach.
4. Define the desired anonymity level (k) for your dataset.
5. Define the splitting criterion for partitioning your dataset. This can be done using either the diversity criterion or the entropy-based criterion.

6. Implement the Mondrian algorithm by recursively partitioning your dataset into k-anonymous partitions using the chosen splitting criterion and the defined hierarchies. The algorithm should stop partitioning when all partitions are k-anonymous or when further partitioning would violate the k-anonymity requirement.
7. Anonymize the dataset by replacing the original values with their corresponding generalised values according to the defined hierarchies.

```

383
384 def mondrian(att_trees, data, k, QI_num, SA_num):
385     """
386     basic Mondrian for k-anonymity.
387     This function support both numeric values and categoric values.
388     For numeric values, each iterator is a mean split.
389     For categoric values, each iterator is a split on QI.
390     The final result is returned in 2-dimensional list.
391     """
392     init(att_trees, data, k, QI_num, SA_num)
393     result = []
394     middle = []
395     wtemp = []
396     for i in range(QI_LEN):
397         if IS_CAT[i] is False:
398             QI_RANGE.append(ATT_TREES[i].range)
399             wtemp.append((0, len(ATT_TREES[i].sort_value) - 1))
400             middle.append(ATT_TREES[i].value)
401         else:
402             QI_RANGE.append(len(ATT_TREES[i]) * '-')
403             wtemp.append(len(ATT_TREES[i]) * '-')
404             middle.append('-')
405     whole_partition = Partition(data, wtemp, middle)
406     start_time = time.time()
407     anonymize(whole_partition)
408     rtime = float(time.time() - start_time)
409     ncp = 0.0
410     for partition in RESULT:
411         r_ncp = 0.0
412         for i in range(QI_LEN):
413             r_ncp += get_normalized_width(partition, i)
414         temp = partition.middle
415         for i in range(len(partition)):
416             temp_for_SA = []
417             for s in range(len(partition.member(i)) - len(SA_INDEX), len(partition.member(i))):
418                 temp_for_SA = [partition.member(i)[s]]
419             result.append(temp + temp_for_SA)
420         r_ncp += len(partition)
421     ncp += r_ncp
422     # convert to GCP percentage
423     ncp /= QI_LEN
424     ncp /= len(data)
425     ncp *= 100
426     if len(result) != len(data):
427         print('Losing records during anonymization!')
428     pdb.set_trace()
429     if __DEBUG__:
430         print("k=kd" % k)
431         print("rate of partitions")

```

Fig 6.2 Implementation of mondrian

6.4.2 Clustering

1. import the required libraries such as NumPy, pandas, and scikit-learn.
2. Load the data into a pandas DataFrame.
3. Preprocess the data by scaling or normalizing it, and handling missing or categorical values.
4. Select the appropriate clustering algorithm and import it from scikit-learn.
5. Create an instance of the clustering algorithm with the desired parameters.
6. Fit the data to the clustering algorithm using the fit method.
7. Extract the cluster labels using the labels_ attribute of the fitted clustering model.
8. Visualize the clusters using matplotlib or any other plotting library.

```

429     QI_LEN = len(data[0]) - 1
430 else:
431     QI_LEN = QI_num
432     for i in range(QI_LEN):
433         LCA_CACHE.append(dict())
434         if isinstance(ATT_TREES[i], NumRange):
435             IS_CAT.append(False)
436             QI_RANGE.append(ATT_TREES[i].range)
437         else:
438             IS_CAT.append(True)
439             QI_RANGE.append(len(ATT_TREES[i]))
440
441
442 def clustering_based_h_anon(att_trees, data, k, QI_num, SA_num, type_alg):
443     """
444     the main function of clustering based k-anon
445     """
446     loli(att_trees, data, SA_num, QI_num)
447     result = []
448     start_time = time.time()
449     if type_alg == 'knn':
450         print("Begin to KNN Cluster based on MCP")
451         clusters = clustering_knn(data, k)
452     elif type_alg == 'kmember':
453         print("Begin to K-member Cluster based on MCP")
454         clusters = clustering_kmember(data, k)
455     elif type_alg == 'ske':
456         print("Begin to SKA Cluster based on MCP")
457         clusters = clustering_ske(data, k)
458     else:
459         print("Please choose merge algorithm types")
460         print("knn | kmember")
461         return 0, (0, 0)
462     rtime = float(time.time() - start_time)
463     ncp = 0.0
464     for cluster in clusters:
465         final_result = []
466         for i in range(len(cluster)):
467             # Custom For non QI Values
468             tmp = []
469             for s in range(len(cluster.member[i]) - len(SA_INDEX), len(cluster.member[i])):
470                 tmp += [cluster.member[i][s]]
471             final_result.append(cluster.gen_result + tmp)
472     result.extend(final_result)
473     ncp += cluster.information_loss
474     ncp /= len(DATA)
475     ncp /= QI_LEN
476     ncp *= 100
477     if __DEBUG__:
478         print("MCP=", ncp)
479     return (result, [ncp, rtime])

```

Fig 6.3 Implementation of clustering

6.5 APPLYING GENERALISATION HIERARCHIES TO THE DATA

After making sure that our data exists and generalisation hierarchies exist for the data as well, we actually set those generalisation hierarchies to the data

```

53 def apply_generalization(data, strat, level, dictionary):
54     tmp_array = []
55     gen_index = []
56     count = len(dictionary)-1
57
58     for v in data:
59         if isinstance(strat, list):
60             args = []
61             for arg_len in range(1, len(strat)):
62                 args.append(strat[arg_len])
63             vg = strat[0](v, level, *tuple(args))
64         else:
65             vg = strat(v, level)
66
67         if isinstance(vg, list):
68             vg = vg[0]
69         if vg not in tmp_array:
70             count += 1
71             dictionary.update({count: vg})
72             tmp_array.append(vg)
73             gen_index.append(count)
74
75         else:
76             for key, val in dictionary.items():
77                 if vg == val:
78                     gen_index.append(key)
79                     break
80
81     return gen_index

```

Fig 6.4 Generalization hierarchies

6.6 ANONYMIZING THE DATA

This is a crucial step as anonymizing the data script generalizes the hierarchy using the function in the above file and then selects the model and anonymizes the data. As a result, we can simply run anonymization by typing in the terminal

Python3 anonymization -data -model - k

```

1 # -*- coding: utf-8 -*-
2
3 import math
4
5
6 # Initialize the algorithm
7 def calculate(nodes, ac):
8     min_k_nodes = [nodes[-1]]
9     topnode = [nodes[-1]]
10
11     evaluate(nodes, ac, topnode, min_k_nodes)
12
13     return min_k_nodes
14
15
16 def sort(nodes):
17     sorted_array = []
18     part_array = []
19     for n in reversed(nodes):
20         h = n.level
21
22         try:
23             part_array[h].append(n)
24         except KeyError:
25             part_array.update({h: [n]})
26
27     for n in sorted(part_array):
28         sorted_array.extend(part_array[n])
29
30     return sorted_array
31
32
33 def evaluate(nodes, ac, topnode, min_k_nodes):
34     for n in nodes:
35         if n.level == math.floor((nodes[-1].level + nodes[0].level) / 2):
36             if check_kanon(n, ac):
37                 test = createsubarray(n, topnode[-1], True, True)
38                 tsok = True
39                 for skn in reversed(min_k_nodes):
40                     if skn in test:
41                         min_k_nodes.remove(skn)
42                         continue
43                 sknsubarray = createsubarray(skn, topnode[-1], True, True)
44                 if n in minsubarray:
45                     tsok = False
46             if tsok:
47                 min_k_nodes.append(n)
48
49     if n.iskanon:

```

Fig 6.5 Data Anonymization

```

98 def check_kanon(node, ac):
99     if node.iskanon is not None:
100         return False
101     else:
102         # Calculation if node is k-anon
103         if ac.calculate_kanon(node):
104             node.iskanon = True
105             return True
106         else:
107             node.iskanon = False
108             return False
109
110
111 def createsubarray(bot, top, direction, only_create_subarray=False):
112     """Creates a subpart of the generalization lattice
113
114     :param bot: bot/start node from where the subpart starts
115     :param top: top/end node where the subpart stops
116     :param direction: Calculate from bot to top (True) or top to bot (False)
117     :param only_create_subarray: Specified if the function is used as utility or for the algorithm
118     :return: List of nodes that are in the subpart of the generalization lattice
119     """
120     # Checks how to use this function
121     if not only_create_subarray:
122         # Check if subarray was already created before
123         if top.id in bot.mem:
124             return None
125         # Mark that this subarray was already created
126         bot.mem.append(top.id)
127
128     # Initialize array with bot and top node
129     subarray = [bot, top]
130     # Create rest of the subarray
131     iterate(subarray, bot, top, direction)
132
133     return subarray
134
135
136 def iterate(list, bot, top, direction):
137     if direction is True:
138         attr_range = range(len(bot.attributes))
139         for n in bot.children:
140             is_valid = True
141             if n not in list:
142                 if n.level == top.level:
143                     return True
144             for num in attr_range:
145                 if n.attributes[num] > top.attributes[num]:

```

Fig 6.6 Data Anonymization

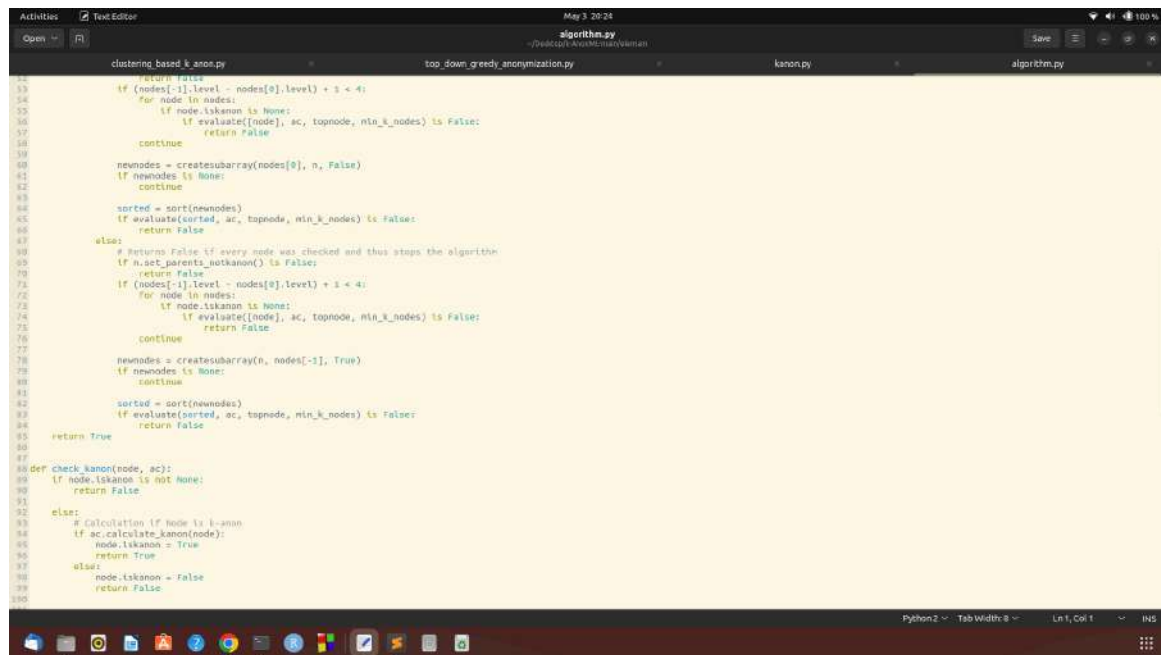


Fig 6.7 Data Anonymization

6.7 STRUCTURE OF THE PROJECT

The given project contains the following folders

- Dataset: Contains data being used for the project
- Generalisation: Contains hierarchies being used for the project
- Results: Contains results which are obtained after computation
- Mondrian: Contains all the files needed to implement the mondrian algorithm. This is the folder the main script looks for when user enters mondrian as the anonymization algorithm
- TDG: Contains all the files needed to implement the top down greedy algorithm. This is the folder the main script looks for when user enters tdg as the anonymization algorithm
- OLA: Contains all the files needed to implement the ola algorithm. This is the folder the main script looks for when user enters ola as the anonymization algorithm
- Clustering based: Contains all the files needed to implement the clustering algorithm. This is the folder the main script looks for when user enters clustering as the anonymization algorithm

6.8 GUI Implementation

The data runs perfectly fine with the scripts but GUI is also included for the users who are not as tech savvy or simply don't want to get into the details of the algorithm. The GUI has options to upload your data, select the quasi-identifying and the sensitive

columns and select k ,anonymization metrics, option to generate charts using powerBI and machine learning using mindsdb

```

1 import streamlit as st
2 import pandas as pd
3
4 # Set page title
5 st.set_page_config(page_title="K Anonymity App")
6
7 # Set background color and button styles
8 st.markdown(
9     """
10     <style>
11     .stApp {
12         background-color: #F8E4E3;
13     }
14     .stButton>button {
15         background-color: #FE5F55;
16         color: white;
17         box-shadow: 2px 2px 2px rgba(0,0,0,0.2);
18         border-radius: 20px;
19         border: none;
20         padding: 12px 24px;
21         font-size: 16px;
22         font-weight: bold;
23         text-transform: uppercase;
24         transition: all 0.3s ease 0s;
25     }
26     .stButton>button:hover {
27         background-color: #F8E4E3;
28         color: #FE5F55;
29         box-shadow: 4px 4px 4px rgba(0,0,0,0.2);
30         transform: translate(-2px, -2px);
31     }
32     </style>
33     """,
34     unsafe_allow_html=True,
35 )
36
37 # Set app title and subtitle
38 st.title("K Anonymity App")
39 st.write("Protect your data with k-anonymity")
40
41 # Upload file and display dataframe
42 file = st.file_uploader("Upload file", type=["csv"])
43 if file is not None:
44     st.subheader("Data Preview")

```

Fig 6.8 GUI implementation

6.9 RESULTS

6.9.1 Comparison across datasets to find the influence of anonymization techniques and data

The main motivation of this project was to see how different algorithms are created but also creating something that the end users can use as well. As a part of the implementation, I all of the implemented algorithms on four different datasets:

1. Adult: The Adult Dataset 7 (also known as Census Income Dataset) contains

45,222 entries derived from the 1994 United States census database. Following the usual parameters for this use case, we use the attributes sex, age, race, marital-status, education, native-country, workclass, and occupation as QIDs and salary-class as the binary target variable (with two categories, $\leq 50K$ and $> 50K$).

2. Cahousing: The California Housing Prices Dataset 8 contains 20,640 entries. We choose the attributes housing_median_age, median_house_value, 7 median_income as well as the coordinates (longitude and latitude) as QIDs (as these seem most relevant concerning privacy) and ocean_proximity as the target variable. Three target values (NEAR BAY, ISLAND and NEAR OCEAN) were merged (giving a new, larger NEAR OCEAN class) to account for the imbalance in the data, leading to three classes $< 1H$ OCEAN, INLAND, and NEAR OCEAN.
3. MGM: The Mammographic Mass Dataset 11 (Elter, Schulz-Wendtland and Wittenberg 2007) contains 830 entries with data from mammography analyses using the Breast Imaging-Reporting and Data System (BI-RADS), patient age and ground truth, i. e. whether the tissue lesions are malignant or benign. We use all attributes (age, shape, bi_rads_assessment, margin and density) as QIDs and severity as the target variable (with two possible values: benign and malignant)
4. CMC: The Contraceptive Method Choice Dataset 9 contains 1,473 entries. The dataset is a subset of the 1987 National Indonesia Contraceptive Prevalence Survey,¹⁰ which contains demographic and socioeconomic characteristics of nonpregnant women as well as the type of contraception they used. We select three attributes (wife_age, wife_edu, num_children) as QIDs and contraceptive_method as the target variable (with three possible values: no_use, short-term and long-term)

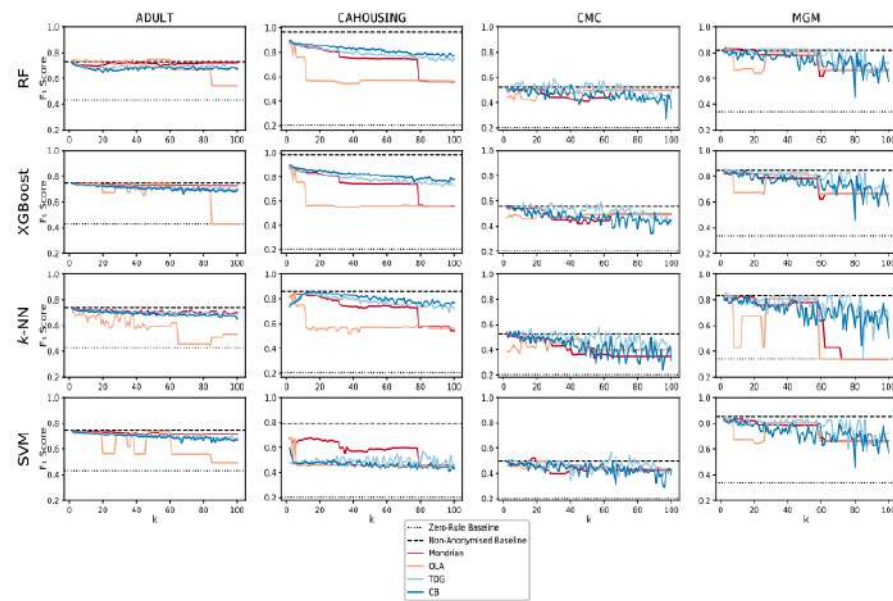


Fig 6.9 Comparison of various datasets and practices

Then i used machine learning algorithms to compare the result before and after anonymization

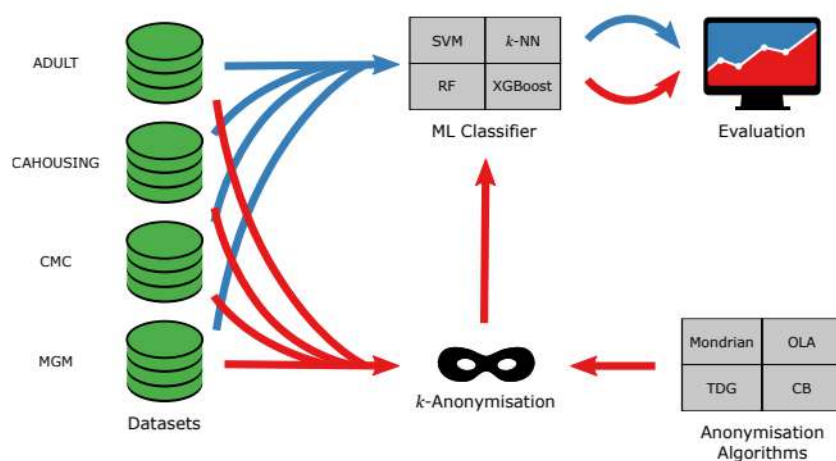


Fig 6.10 Experimental setup

These were the general trends observed:

- With an increasingly strong k-anonymity constraint, classification performance generally degrades (with the degree of degradation strongly dependent on dataset and anonymisation method).
- For some datasets, even for very large k of up to 100 (which is far higher than the values used in practice nowadays), the performance loss remains within acceptable limits
- Mondrian provides a better basis for subsequent classification than other anonymization algorithms investigated, i. e. Optimal Lattice Anonymization, Top-Down Greedy Anonymisation and k-NN ClusteringBased Anonymization.

6.9.2 Data Before vs After anonymization

Before anonymization					After anonymization with k = 2				
id	age	city_birth	zip_code	disease	id	age	city_birth	zip_code	disease
1	55	San Giovanni	17049	Cancer	1	50-100	San Giovanni	17***	Cancer
2	23	Comina	26151	AIDS	2	0-50	Italy	*****	AIDS
3	17	Marina Di Camerota	73015	AIDS	3	0-50	Italy	*****	AIDS
4	62	San Giovanni	17028	Autism	4	50-100	San Giovanni	17***	Autism
5	40	Marina Di Camerota	58014	Autism	5	0-50	Italy	*****	Autism

Fig 6.11 Data before vs after anonymization

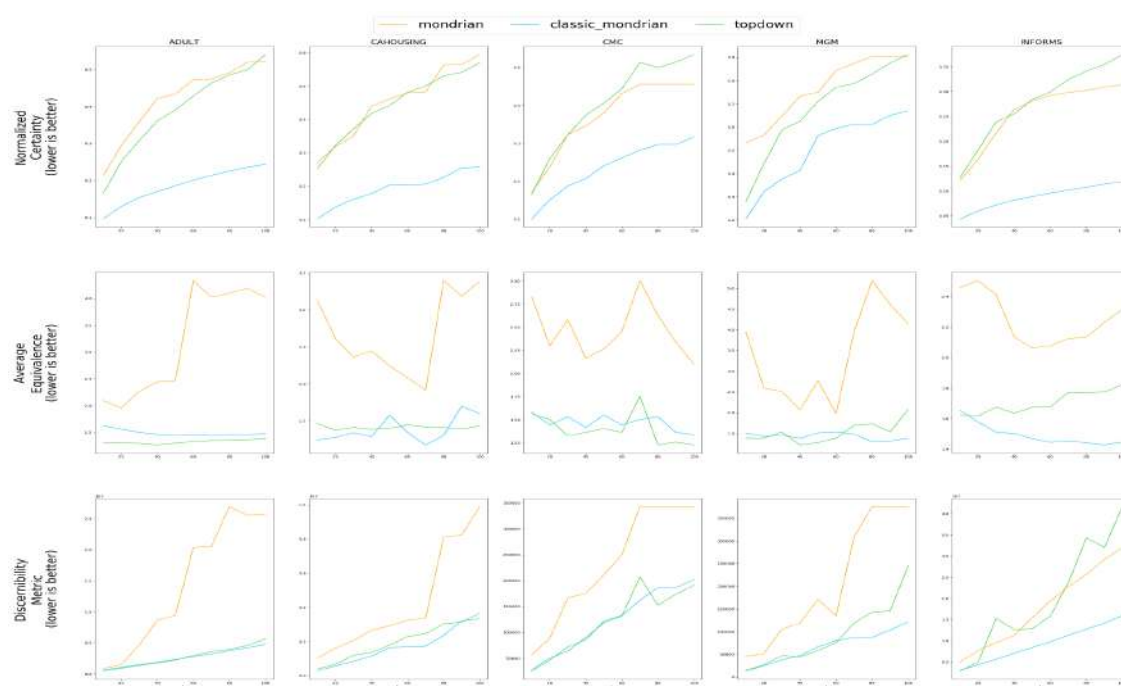


Fig 6.12 Loss Metrics for various datasets

6.9.3 GUI demo

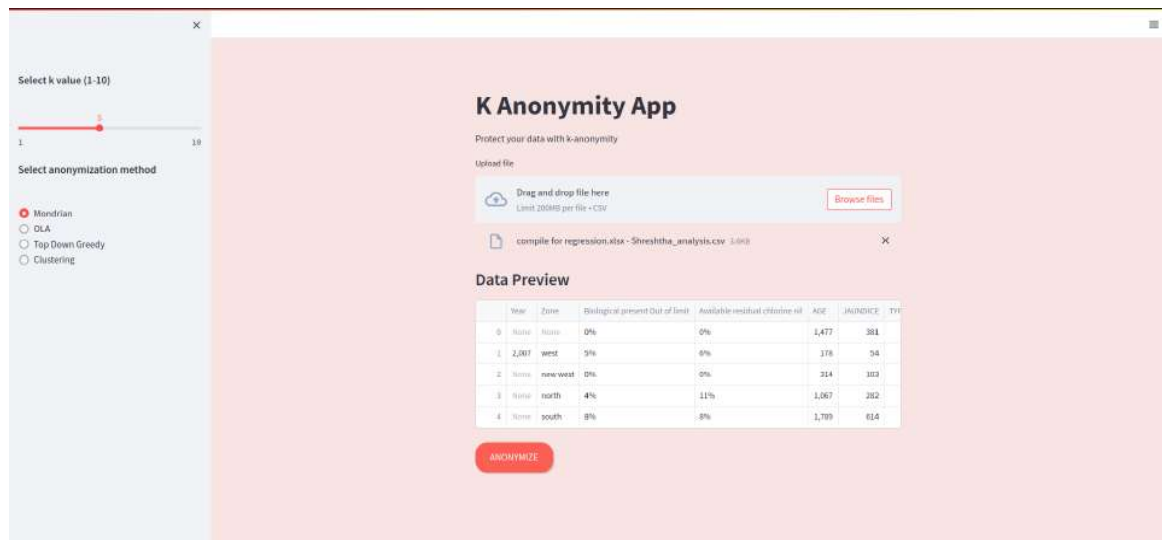


Fig 6.13 Initial GUI

6.9.4 Visualisation using powerBI

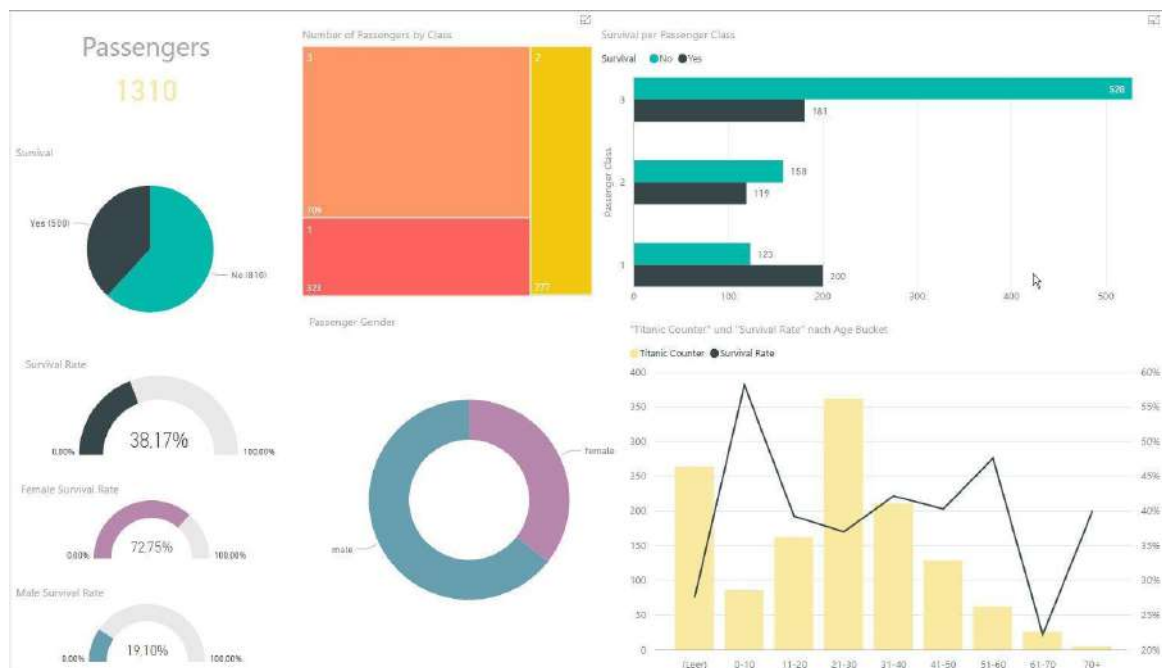


Fig 6.14 Visualisation using titanic dataset

7 TESTING

Testing refers to checking the functionality of a given project with respect to a specific set of requirements. Usually, better the testing and code coverage, less the chances of failure. Testing allows you to verify that the code you've written works as intended and meets the project requirements. It helps ensure that the code is free of bugs and errors, and that it performs as expected.

Catching bugs and errors early in the development process can save time and money in the long run. By identifying issues early, you can avoid costly fixes and rework down the line.

Testing helps to mitigate risk by identifying potential issues and vulnerabilities before they become bigger problems. It allows you to identify and address security issues, performance bottlenecks, and other risks that could impact the project's success.

Testing is not a one-time event, but rather an ongoing process that allows you to continually improve the quality of your code. By collecting data and analysing the results of your tests, you can identify areas for improvement and make adjustments to optimise performance and functionality.

The given project has tests which are divided into two broad categories:

7.1 CODE TESTING

As the name suggests, code testing is done to ensure that all of the code and its derivatives are working as expected. Code testing is divided into three phases:

7.1.1 GUI TESTING

GUI testing is done to ensure that GUI is working as planned and there are no unplanned bugs or features that do not work according to the plan. Here are a few of the test cases we did as a part of this phase

Test No	Test case	Result
---------	-----------	--------

001	Being able to load the webapp and the background and UX are loaded as expected	pass
002	Upload button works and only takes data of upto 200 MB	Pass
003	Able to select quasiidentifying columns	pass
004	Able to select value of k and privacy model	pass
005	Mindsdb and powerbi button works as expected	pass

Table 7.1 Test cases for GUI

7.1.2 Algorithm Testing

This testing ensures that algorithms work as expected and is done a lot more aggressively, here are only a few basic test cases for the same

Sr NO	Test case	Result
001	Making sure algorithms are able to handle sensitive attributes	pass
002	Sending float values to	All pass except mandorian

	columns	
003	Large dataset with complex correlation	OLA and top down failed
004	Able to handle very large and very small k values	pass
005	Able to anonymize the data based on given k and suppression level and model	pass

Table 7.2 Test cases for algorithms

Thus, testing made sure to catch the basic bugs and the rest will be taken care as time progresses

7.2 HYPOTHESIS AND ANALYSIS TESTING

Taking the datasets mentioned in the previous chapter i.e. adult, cmc and mgm and cahousing, I tested them using machine learning models such as svm, knn, rf and xgboost to see the impact anonymization had on the accuracy of the prediction. These serves two important purposes:

1. It tells us about what kind of data is suited to which algorithm thus drawing a meaningful conclusion about what algorithm suits which data better
2. Gives me direction on how to move my data forward

Metrics used for accuracy testing in classification:

The F1 score is a measure of a model's accuracy that combines precision and recall into a single value. It is the harmonic mean of precision and recall, where precision is the number of true positive results divided by the number of true positive plus false positive results, and recall is the number of true positive results divided by the number of true positive plus false negative results.

The F1 score ranges from 0 to 1, with a higher score indicating better performance. A perfect F1 score of 1 means that the model has achieved both perfect precision and perfect recall, while a score of 0 indicates that the model has failed to correctly identify any positive cases.

The F1 score is particularly useful when the data is imbalanced, meaning that one class is more prevalent than the other. In such cases, accuracy can be misleading, as a model that always predicts the more prevalent class will have a high accuracy even if it is not performing well on the minority class. The F1 score, on the other hand, takes both precision and recall into account and can provide a more accurate measure of a model's performance in these situations.

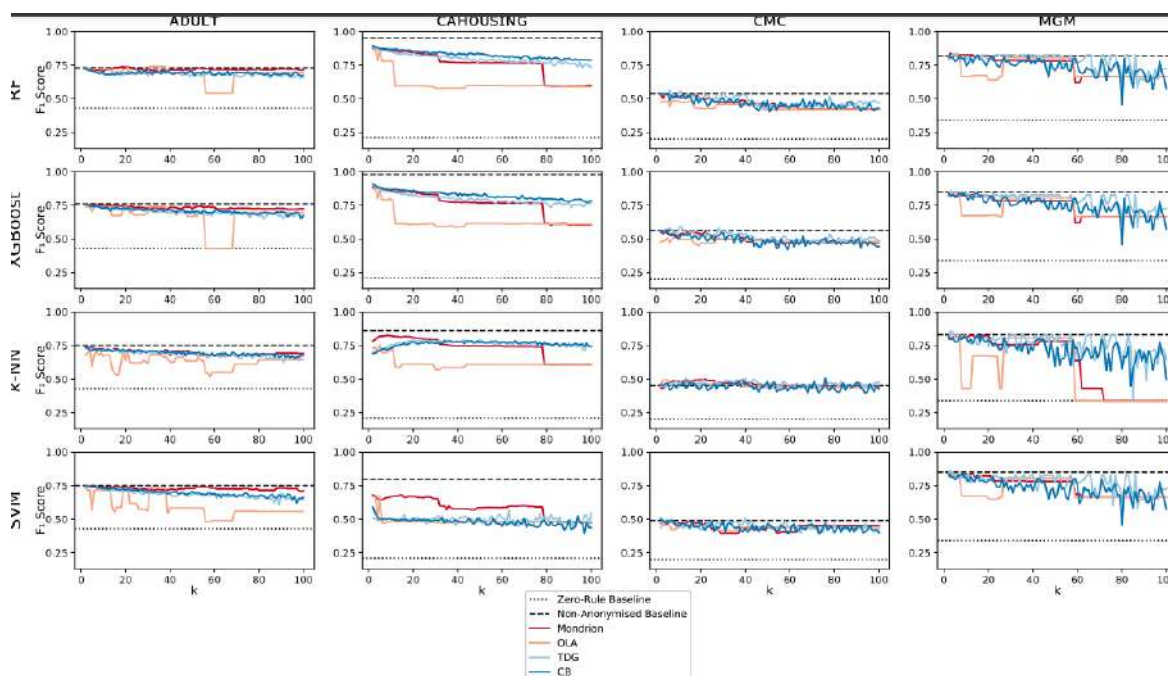


Fig 7.1 F1 score for various datasets and models

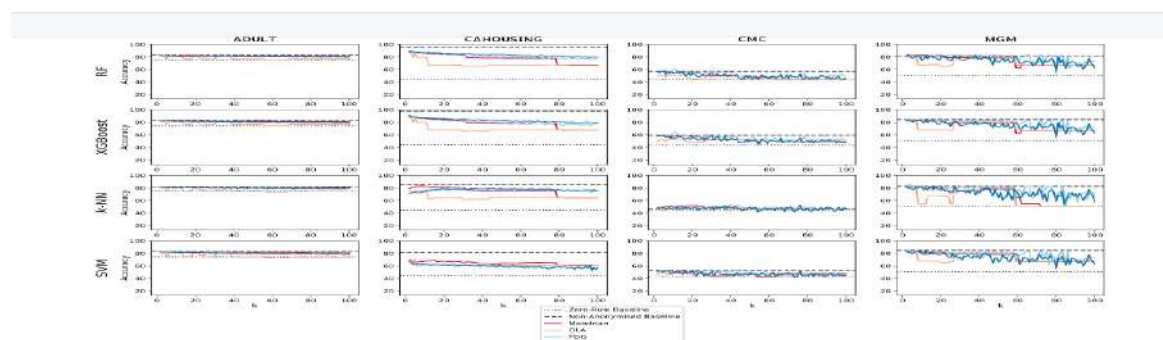


Fig 7.2 Accuracy

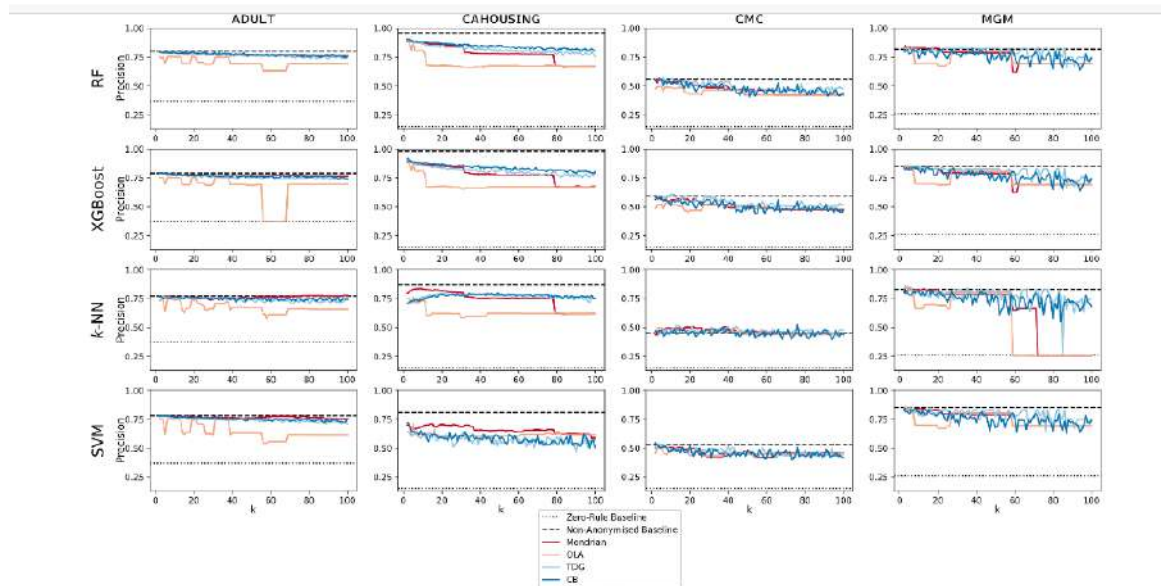


Fig 7.3 Precision

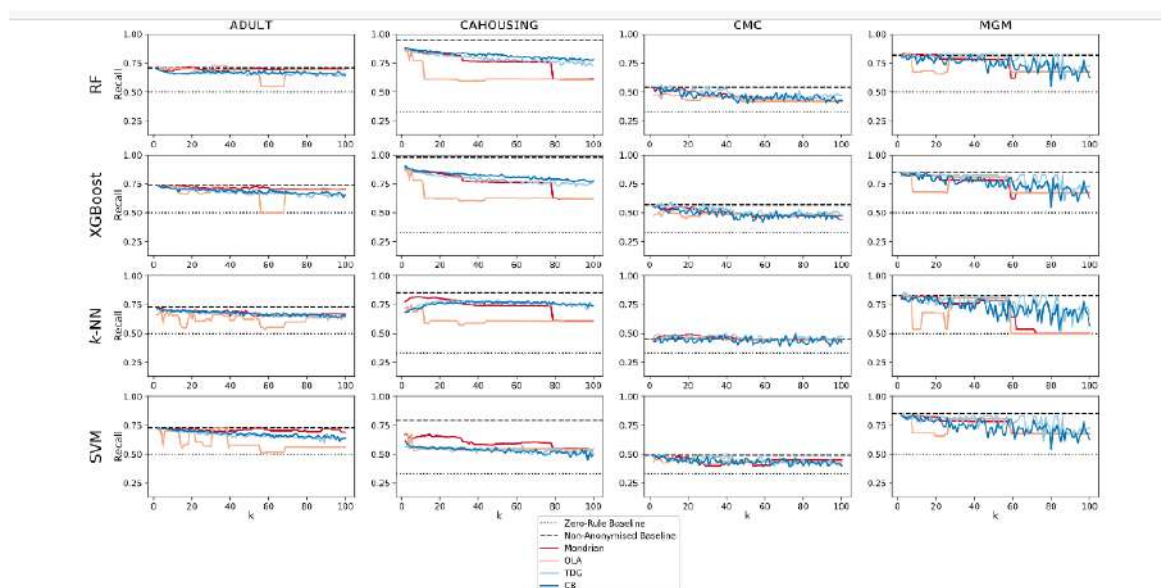


Fig 7.4 Recall

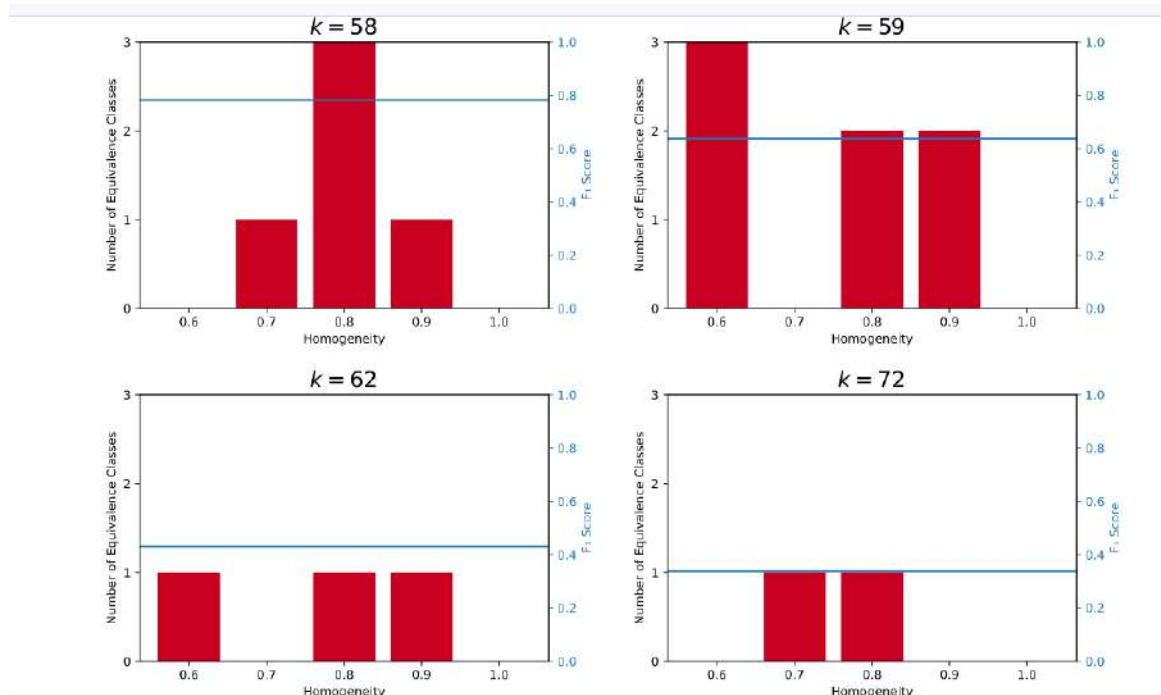


Fig 7.5 Values of different k for mgm data using mondrian

8 CONCLUSION

8.1 OVERALL ANALYSIS OF THE INTERNSHIP AND THE PROJECT

During my time at Eternal soft solutions working on data anonymization pipeline, I realised the value of open source and asking for help when needed. I was taken through the various sections of cloud, software development and research which made me realise how seemingly different domains of engineering are all connected fundamentally. Here were a few learnings and takeaways:

- Knowing when to ask for help and how to ask for help are important
- If you get stuck on something for more than 30 minutes, it is always a good idea to ask for help
- I learnt the value of my own data and digital footprint and how people steal your information
- I learnt about the principles of clean code and how to organise your data better

Technical takeaways:

- Became proficient with data structures and algorithms
- Understood end to end machine learning process
- Learnt about different ways to implement data anonymization in python
- Learnt about python libraries such as scikit learn, matplotlib, pandas, numpy, seaborn, tkinter, streamlit etc

Overall analysis:

To sum things up, the project has been a wonderful experience as i got to build a system that users could use and implemented algorithms such that it can be used for research

8.2 PROBLEMS ENCOUNTERED AND POSSIBLE SOLUTIONS

- My first day working with Linux, I was told to boot from source and I forgot to install a necessary package which caused a lot of trouble. To avoid this, i would recommend following the instructions really closely
- We originally decided on using python module pyarxaas and wasted a few weeks configuring the same as there were not a lot of resources available however it used to cause constant dependency issues and there was not enough documentation or support. The best we could do was to write things from scratch as creating a virtual environment was having issues with the frontend
- While implementing the mondrian, there were issues with sensitive attributes hence implemented OLA which could deal with them
- Tkinter was not giving the required results and was too hard to configure hence we switched to streamlit

- Algorithms were not optimised but used chatgpt, internet, books and help from the mentors to reduce the computation time and space
- My system kept crashing frequently and had issues with the monitor and that set me back quite a few days but it was an old hardware bug and hence i eventually got it replaced

8.3 SUMMARY OF THE INTERNSHIP

The data anonymization pipeline created helps solve the problem of data privacy within the organization and for the client as they can freely share their data while keeping certain parts of the data indistinguishable. Not only that, the project also has k-anonymity implemented using various optimization algorithms and techniques hence it can be suited for a variety of data needs. The interface allows a lot of users who are not familiar with the algorithm to also get full benefits of the algorithm. Thus, the project is expected to have benefits across a wide range of sections. It can also be used to feed basically any machine learning algorithm available

8.4 LIMITATIONS AND FUTURE WORK

- One of the biggest limitations that I find with this is that it takes way too long to compute the anonymization. Hence, for a dataset of considerable complexity, this tool would take a modest amount of time.
- Albeit the integration with powerbi, users still have to generate the data themselves.
- It has trouble with floating point integers
- There is a lot to be documented as a lot of the powerful functions of the data still remain undocumented
- This tool only provides support for k-anonymity whereas a lot of successor algorithms like differential privacy are better
- This tool only works with python and is not a stand alone application
- Future work can include working on the algorithms and adding more dynamic anonymization algorithms and adding more documentation
- Furthermore, having a standalone app like ARX that the users can use would also be good

Data Anonymization pipeline

Team ID : 303902

Internal Guide: Prof Himanshu Nayak

By: Shreshtha Modi(190130111081)

Branch: Electronics and Communication Engineering

Introduction:

- With the rise in people using internet and the dotcom boom, virtually everyone has an online footprint these days
- Although the data privacy laws have gotten stricter and people's personal information is not as easily accessible from the internet these days, there are still ways that people with malicious intentions can use some aspect of the data as it is or combine it with other identities to generate a lot of the insights about a person
- Hence it is very important to take anonymize the data in such a way that both identifiers and quasi-identifiers are taken care of
- The given project proposes an end to end data anonymization pipeline along with integration for several platforms for a seamless experience

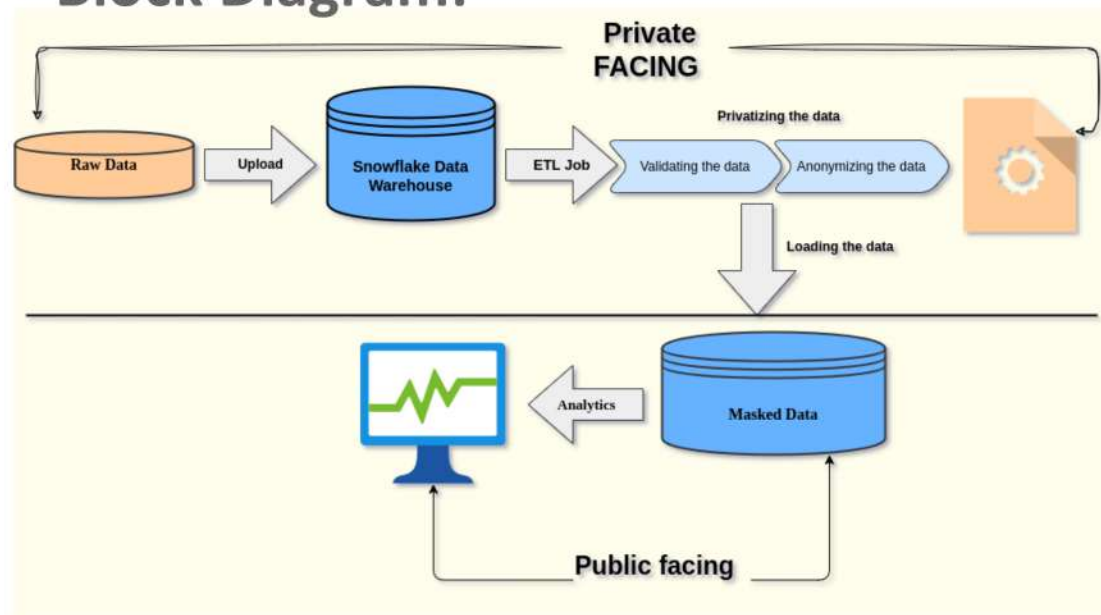
Objectives

- To implement an end to end service which lets users input the data and the hierarchies and then anonymize the data with various anonymization algorithms such as Mondrian and OLA
- To add the support for services such as mindsdb, power Bi and snowflake so that users can analyse the data using simple SQL commands
- To provide a friendly graphical user interface with which the users can interact with to make sure that they are not exposed to the inner workings
- To implement state-of-the art algorithms using python
- To compare the findings of the algorithms using a machine learning model before and after the anonymization and using datasets of various types, sizes and domains and drawing a suitable conclusion on which algorithm works better for what type of the data

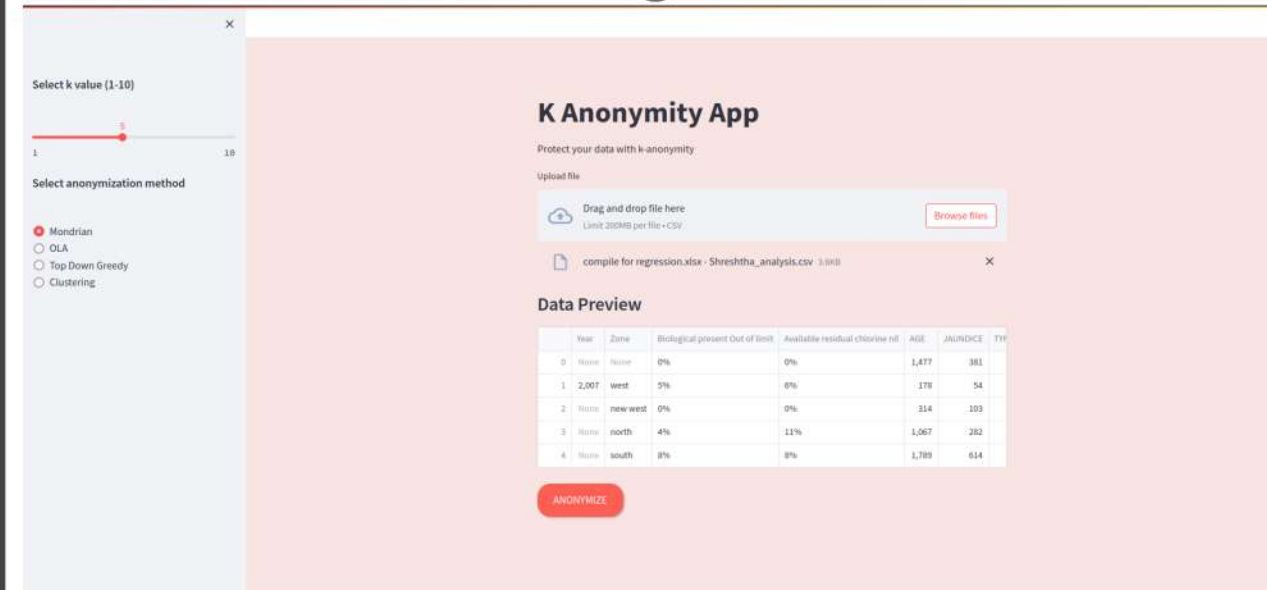
Outcome:

A python module which can help users anonymize the data using various techniques

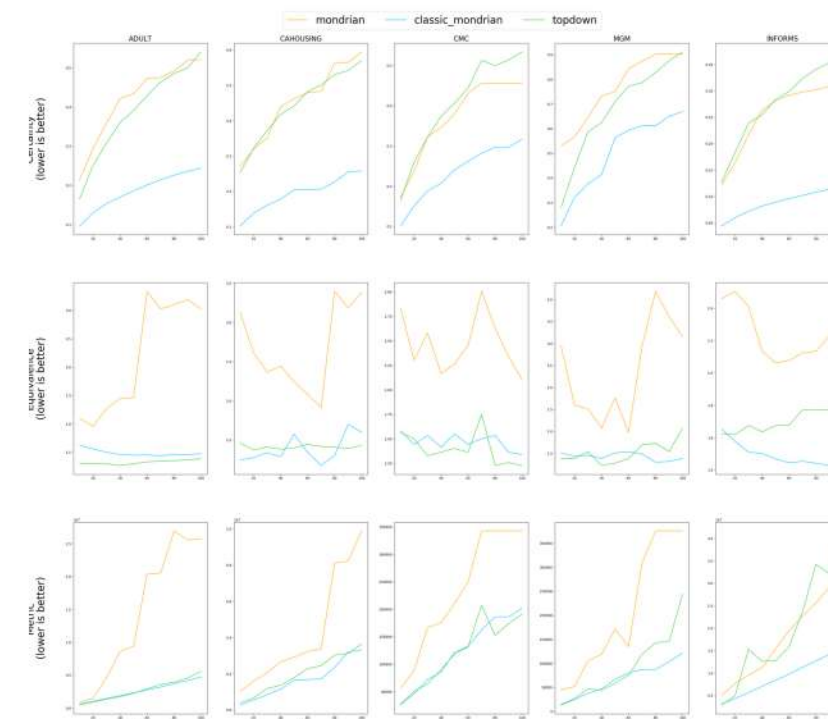
Block Diagram:



Results and Findings:



GUI for the modules

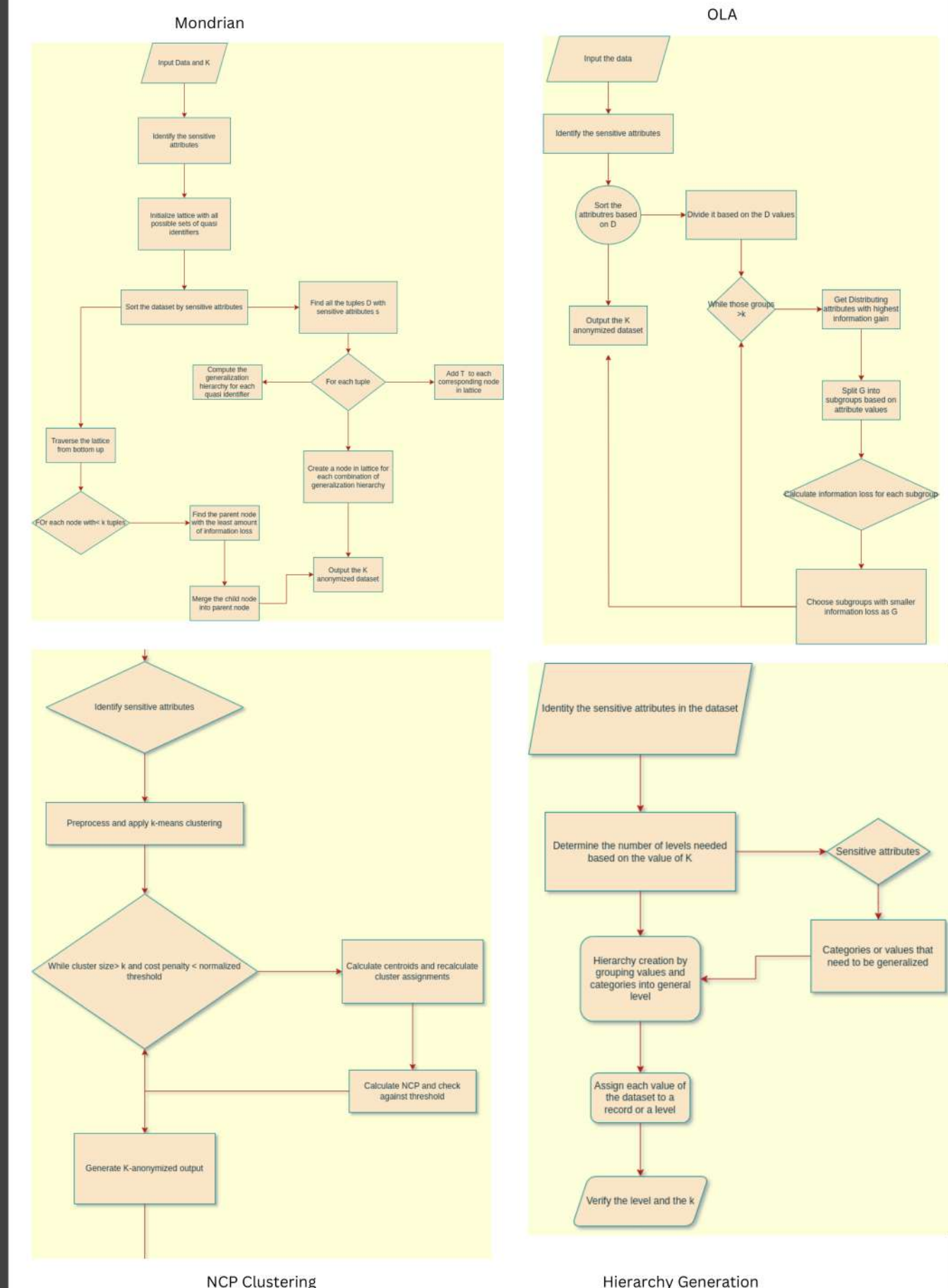


F1 score for the anonymized data using classification models



Visualisation of anonymized titanic data using power Bi integration

Flow-diagrams for algorithms



Conclusion

The given project is extremely useful for companies looking to comply with various data privacy laws such as HIPAA and GDPR. Since the implementation is open source, a lot of organisations would benefit from it as they can customise the code as per their needs. This eliminates the need for data analysts as it provides integrations for no-code platforms such as minds db and power-bi. Moreover, this can suit all datatypes and data with smaller and larger size as there are several k-anonymity implemented which suit various data needs. The following module can also be used to build upon research in the field of data privacy

REFERENCES


[Introducing PII data identification and handling using AWS Glue DataBrew](#)

[Data Anonymization Software - Differences Between Static and Interactive Anonymization | Aircloak](#)

[An Improved Algorithm for K-anonymity | SpringerLink](#)

[k-anonymity - Wikipedia](#)

[Protecting Privacy Using k-Anonymity - PMC](#)

 [Anonymization techniques and Differential Privacy](#)

 [2012-04-11 - : K-Anonymity in Social Networks: A Clustering Approach - CERIAS S...](#)

[Mondrian Multidimensional K-Anonymity](#)

[GitHub - qiyuangong/Mondrian: Python Implementation for Mondrian Multidimensional K-Anonymity \(Mondrian\).](#)

[Utility-Based k-Anonymization](#)

<https://stackoverflow.com/questions/11390143/what-is-meant-by-k-anonymity-and-l-diversity-and-what-is-difference-between-the>

<https://groups.inf.ed.ac.uk/tulips/papers/tahaei2022pets.pdf>