PRIVACY SHIELD: SECURING SOCIAl SPHERE

A Project report submitted in partial fulfilment of the requirements For the award of the Degree of

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE ENGINEERING
By

Thangudu Dhilleswara Rao
(320136410107)
Menaweli Navya
(320136410068)
P. Rohit Palika
(320136410077)
Polarasi Keerthi
(320136410088)

Under the esteemed guidance of
Shri. Syed Mujib Rahaman
Associate Professor
Department of Computer Science Engineering

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

Dr L. BULLAYYA COLLEGE OF ENGINEERING

(Permanently Affiliated to Andhra University, Visakhapatnam)
New Resapuvanipalem, Visakhapatnam-530013

Year of submission: 2024

Dr L. BULLAYYA COLLEGE OF ENGINEERING

New Resapuvanipalem, Visakhapatnam-530013

Department of Computer Science Engineering

Bonafide Certificate

This is to certify that Mr. **Thangudu Dhileswara Rao**, Ms.**Menaweli Navya**, Ms.**Polarasi Keerthi,** Mr.**Rohit Palika** bearing register numbers 320136410107, 320136410068, 320136410088, 320136410077 students of 4th Year B. Tech in Computer Science Engineering, has carried out the project work titled “**PRIVACY SHIELD: SECURING SOCIAl SPHERE** ” at Dr L. Bullayya College of Engineering, Visakhapatnam during the academic year 2023-24.

Project Supervisor
Shri. Syed Mujib Rahaman
Associate Professor
Dept. of Computer Science Engineering

Head of the Department
Dr D. Madhavi
Professor
Dept. of Computer Science Engineering

ABSTRACT

In today's world, we all use social media to connect with friends and share our lives. But sometimes, our personal information on these platforms is seen by the wrong people. This is a big problem called "privacy breaches." Privacy breaches in social media refer to unauthorized access, sharing, or use of personal information on individuals social media profiles. These breaches occur due to various factors, including lack of privacy settings, data leaks, and malicious actors.

The consequences of such breaches can be severe, ranging from identity theft to personal information exposure, leading to potential harm, financial loss, and emotional distress for users. We want to keep your private information just for you and your chosen friends, so you can enjoy social media without worrying about your privacy. By understanding the problem at its core and proposing practical solutions, this research aims to safeguard the online privacy and security of social media users in an increasingly interconnected digital world.

Project goal is to design an optimization algorithm that achieves a trade-off between self-disclosure utility and their privacy. We implemented two privacy preserving algorithms to defend against an inference attack that is privacy preserving algorithm (PPA) targeting high utility and social relation based multidimensional knapsack problems with greedy heuristics with low computational complexity.

The project aims to develop an optimization algorithm that balances self-disclosure utility and privacy. Specifically, we implemented two privacy-preserving algorithms to counter inference attacks. These algorithms address the high utility and social relation aspects through multidimensional knapsack problems employing greedy heuristics with low computational complexity.

ACKNOWLEDGEMENT

We would like to express our heartfelt gratitude to Prof D. Deepak Chowdary, Principal, and to Shri. Syed Mujib Rahaman, Associate Professor, Dr. L. Bullayya College of Engineering for giving us the opportunity to do this project. We thank Dr. D. Madhavi, HOD, Department of CSE, Dr. L. Bullayya College of Engineering, Visakhapatnam, for her guidance in producing this work.

We are deeply indebted to our project guide Associate Prof. Shri. Syed Mujib Rahaman, Dr. Lankapalli. Bullayya College of Engineering, Visakhapatnam, for guiding us throughout the project in spite of his busy schedule.

Apart from the efforts, the success of this project depends largely on the encouragement of another faculty of CSE, Dr. Lankapalli. Bullayya College of Engineering, Visakhapatnam. We take this opportunity to express our gratitude to the entire faculty who has been instrumental in successfully completing this project. Also deserving of thanks to our family and friends, for their support and for their confidence in our achievements.

Dhilleswara Rao.T
Navya .M
Keerthi.P
Rohit.P

DECLARATION

This is to declare that the Project work entitled “Privacy Shield: Securing Social Sphere ” under the Research Cluster “Privacy-Preserving network security in Data science and Deep Learning” is a bonafide work done by us with the esteemed guidance of Shri. Syed Mujib Rahaman, Associate Professor, Dr. Lankapalli Bullayya College of Engineering. This project report is being submitted in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Science Engineering during the academic year 2023-2024. This project possesses originality as it is not extracted from any source and it has not been submitted to any other institutions and universities.

T.Dhilleswara Rao
(320136410107)

Menaweli Navya
(320136410068)

P. Keerthi
(320136410088)

P. Rohit
(320136410077)

Visakhapatnam

Date:

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1.INTRODUCTION

In this digital era, social networks have become an integral part of our lives, connecting us in various domains like communication, education, and entertainment. However, with the convenience of sharing comes the responsibility of safeguarding our privacy.

Privacy Shield is our initiative to address the growing concerns surrounding privacy in the social media landscape. We understand that users want to enjoy the benefits of social interaction without compromising their personal information.

This project aims to provide a robust shield that ensures your online presence remains secure and your data stays private. In this documentation, we will walk you through the key features and methodologies employed in "Privacy Shield." From understanding potential threats to implementing privacy settings effectively, we're committed to empowering users with the knowledge and tools needed to navigate the social sphere securely.

Social networking sites are defined as "A website that provides a virtual community for those interested in a certain subject or just to 'hang out' together,".

1.1. Background and Motivation

The motivation behind "**Privacy Shield: Securing Social Sphere"** is simple – we want everyone to feel confident and secure while using social media. With the increasing concerns about privacy, it's essential to have a reliable shield that protects our personal space in the digital realm.

Our motivation is to empower you, the user, with the knowledge and tools to navigate the social sphere without worrying about your privacy. We believe that everyone deserves a safe and secure online experience, and "Privacy Shield" is our way of making that a reality.

The Background and Motivation for addressing privacy breaches in social media stem from the increasing reliance on these platforms for communication, networking, and sharing personal information. As social media usage continues to surge globally, concerns about privacy and data security have become more pronounced.

1. **Rise of Social Media:** Social media platforms have become integral parts of daily life, facilitating connections, information dissemination, and entertainment for billions of users worldwide.
2. **Data Collection Practices:** Social media companies collect vast amounts of user data, including personal information, preferences, and online behaviours, to tailor content, target advertisements, and enhance user experiences.
3. **User Trust and Confidence:** Maintaining user trust is paramount for social media platforms to sustain their user base and reputation. Addressing privacy breaches demonstrates a commitment to protecting users' personal information and fostering a secure online environment.
4. **Legal and Ethical Imperatives:** Adhering to privacy regulations and ethical principles is not only a legal requirement but also a moral obligation for social media companies. Proactively addressing privacy breaches aligns with corporate responsibility and ethical business practices.

1.2. Problem Statement:

Many individuals, while enjoying the benefits of online connectivity, face challenges in safeguarding their privacy. Instances of data breaches, unauthorised access, and misuse of personal details are increasingly prevalent, creating a need for a reliable solution.

The problem at hand is that users lack effective tools and straightforward guidance to ensure their privacy within the social sphere. Concerns about potential threats, information disclosure, and the intricacies of privacy settings often leave users feeling vulnerable and uncertain about how to navigate the digital landscape secThe major problem in online social networks is how to preserve user’s privacy. Generally, online social networks provide a platform to publish the data in such a way that users' privacy is protected and allow the maximum utilization of the data (i.e, the published urely).

The Project- **Privacy Shield** aims to address this problem by providing a comprehensive solution that simplifies the process of securing personal information on social media platforms. This project is motivated by the desire to empower users with accessible tools and knowledge, making online privacy a tangible reality for everyone.

data is capable of predicting new important decisions from the ML model). A design greedy based method that preserves privacy of a data and allows data analyser or data analytics to utilize data at maximum level to perform knowledge discovery. For example, the data utility feature of instagram provides recommendations of ads, reels etc to the user. At the same time, the user's privacy should also be protected.

This problem can be solved in two categories, one is by considering characteristics of the social actor while the other one is by considering social links of the actor.

In the first category we assume that the characteristics of each social actor is independent i.e; the change of characteristics of one social actor will not affect the other social actor.

○ Here for every public attribute of the user there will be a utility value, self-privacy disclosure value and the threshold value.
○ The total utility of a social actor is calculated by the sum of utility values of his published attributes.
○ We will select the attributes of the user in such a way that it has maximum utility and its self privacy disclosure value will not exceed the privacy protection threshold.

● Each edge involves two social actors. Unlike the first category, which considers social actors independent of each other in the second category, publishing the connection of a social actor will also affect the other social actor involved in that particular connection.

○ Here we are considering all the protection constraints including all social actors and their secrets.
○ Here the attributes and social connections are selected in such a way that maximum utility is achieved while self privacy disclosure value will not exceed the privacy protection threshold.

1.3. Applications:

1.Social Network Analysis:

- Understanding how individuals connect and interact in social networks like Facebook, Twitter, or LinkedIn.
- Predicting new friendships or relationships between users based on their existing connections.

2.Fraud Detection:

- Identifying suspicious behavior or fraudulent activities in networks, such as financial transactions or communication networks.
- Predicting potential fraud cases based on patterns and anomalies detected in the network structure.

3.Cybersecurity:

- Detecting potential cybersecurity threats or attacks by analyzing network traffic patterns and identifying anomalies.
- Predicting the likelihood of a cyberattack based on historical data and network behavior.

4.Recommendation Systems:

- Recommending new products or services to users based on their interactions or similarities with other users.
- Suggesting new connections or contacts to users in professional networking platforms.

2.REQUIREMENT ELICITATION AND ANALYSIS

It's essential to outline the specific needs and expectations that our project aims to achieve. The widespread use of social media brings people together, but it also raises concerns about the safety of personal information.

2.1 Existing System:

The existing system lacks a comprehensive solution to address the increasing concerns related to privacy breaches on social media platforms. Currently, social media users face challenges in safeguarding their personal information, with limited tools and guidance available to ensure effective privacy settings. Users often grapple with the complexities of privacy configurations, leading to potential vulnerabilities in their online presence. There is a lack of real-time monitoring for potential threats, and users may not receive timely alerts regarding suspicious activities or unauthorized access attempts. Educational resources on best practices for online privacy are also limited.

2.2 Proposed System:

The proposed system, "**Privacy Shield: Securing Social Sphere**," aims to revolutionize the way users protect their privacy on social media platforms. It introduces a user-friendly interface that simplifies the configuration of privacy settings, ensuring that users, regardless of technical expertise, can navigate the system with ease.

The system provides a comprehensive set of privacy settings, offering granular control over the sharing of personal information. To empower users, the system includes educational resources, such as tooltips, FAQs, and easy-to-understand documentation, guiding them on best practices for online privacy.

Real-time threat monitoring is a key feature of the proposed system, enabling the immediate detection of potential threats such as unauthorized access attempts or suspicious activities. Users can personalize security alerts based on their preferences, staying informed about any changes or potential breaches in their privacy settings.

The system seamlessly integrates with major social media platforms, ensuring a consistent and secure experience for users across different networks. Regular updates and maintenance are incorporated to address emerging privacy concerns, stay current with social media platform changes, and enhance overall system security.

2.3 Future Scope

The future scope of the PRIVACY SHIELD: SECURING SOCIAL SPHERE project encompasses several key areas aimed at further strengthening user privacy and security in the social media landscape:

- **Enhanced Privacy Features:** Continuously evolve and enhance the privacy features offered by PRIVACY SHIELD to provide users with more granular control over their personal information and interactions on social media platforms.
- **Advanced Threat Detection:** Develop and implement advanced threat detection mechanisms to proactively identify and mitigate emerging privacy risks, including data breaches, identity theft, and social engineering attacks.
- **User Education and Awareness:** Expand educational resources and awareness campaigns to empower users with knowledge about privacy risks, best practices for protecting their personal information, and how to effectively utilize the privacy features offered by PRIVACY SHIELD.
- **Global Privacy Standards Compliance:** Ensure compliance with evolving privacy regulations and standards globally, and actively participate in shaping future privacy policies to promote user-centric privacy protections in the social media sphere.

2.4 Feasibility Study:

The feasibility study for the "Privacy Shield" project involves assessing its technical, operational, and economic viability. The study aims to determine whether the proposed system is practical and achievable within the given constraints.

Technical Feasibility:

- Evaluate the technical capabilities and requirements for implementing the Privacy Shield system.
- Assess the compatibility with existing social media platforms.
- Ensure that real-time monitoring and security alert mechanisms are technically feasible. Operational Feasibility:
- Examine how well the proposed system aligns with the operational processes of social media users.
- Consider the ease of integration and use within the existing social media landscape.

- Analyse the potential impact on users' daily interactions with social media platforms.

Economic Feasibility:

- Estimate the overall costs associated with developing, implementing, and maintaining the Privacy Shield system.
- Assess the potential return on investment, considering the value it brings to users in terms of enhanced privacy and security.

2.5. System Requirements:

The system requirements for **"Privacy Shield: Securing Social Sphere"** include:

2.5.1.Functional Requirements:

- User data encryption: All user data stored or transmitted by social media platforms must be encrypted using strong encryption algorithms.

- Data anonymization: User data should be anonymized whenever possible to make it more difficult to identify and track individual users.

- Differential privacy: Differential privacy should be used to provide statistical information about user data without revealing the data itself.

- Secure multi-party computation: Secure multi-party computation should be used to allow multiple social media platforms to compute functions on user data without revealing their data to each other.

- Granular access control: Users should have granular access control over their data, allowing them to choose who can access their data and for what purposes.

- Data minimization: Social media platforms should collect only the data that is necessary for a specific purpose.
- Transparency: Social media platforms should be transparent about their data collection and use practices.
- Accountability: Social media platforms should be accountable for their data collection and use practices.

2.5.2. Non-Functional Requirements:

- Performance: The Privacy Shield should not significantly impact the performance of social media platforms.
- Scalability: The Privacy Shield should be scalable to support the needs of a large number of users.
- Security: The Privacy Shield must be secure and resistant to attacks.
- Usability: The Privacy Shield should be easy to use for both users and social media platform operators.
- Interoperability: The Privacy Shield should be interoperable with existing social media platforms.
- Maintainability: The Privacy Shield should be easy to maintain and update.
- Cost-effectiveness: The Privacy Shield should be cost-effective to develop and implement.

Software Requirement Specification Table:

Requirement ID	**Requirement**	**Requirement Description**	**Priority**
REQ- 1	Manage Users	The Administrator can manage user data in the system, including adding,editing or deleting user	High
REQ-2	Manage System Settings	The Administrator can configure the privacy manage systems settings. This could include setting Privacy defaults , managing access controls, or defining data retention policies .	High
REQ-3	Calculate Privacy	The social actor can calculate some privacy metric relate to the profile or data. The system might provide tools to assess the users privacy exposure based on their settings and social connections.	Medium
REQ-4	Generate Adjacency Matrix	The Social actor can generate a visual representation of their social connections, possibly in the context of privacy management.	Medium
REQ-5	Access Resources	The social actors can access educational resources through the management systems. The system likely act as an intermediary ,potentially anonymizing or protecting the users privacy while they access the resources.	Medium

3.SYSTEM DESIGN

3.1 Object Oriented Analysis and Design:

The system design for the provided code involves several components and steps to perform link prediction on a social network graph. Here's a high-level overview:

1.Data Loading:

- The system loads the edges (or links) of the social network graph from input files (Output.edges and Initial.edges).

2.Graph Construction:

- The loaded edges are used to construct a networkx graph object (G), representing the social network.

3.Link Prediction Algorithms:

- The system applies different link prediction algorithms to predict new edges in the social network graph.
- Implemented algorithms include:
  - Triadic closure
  - Jaccard coefficient
  - Resource allocation index
  - Adamic-Adar index
  - Preferential attachment

4.Prediction Calculation:

- The predicted new edges are calculated based on the output of each link prediction algorithm.

5.Accuracy Evaluation:

- The system evaluates the accuracy of the predicted new edges by comparing them to the initial edges provided.
- Accuracy is calculated as the percentage of predicted edges that exist in the initial edges.

6.Output Display:

- The system displays the accuracy of each link prediction algorithm, providing insights into their effectiveness in predicting new edges.

7.Error Handling:

- The system should include error handling mechanisms to deal with potential issues such as missing input files or invalid data formats.

Design Goals:

Design goals are the qualities that the system should focus on. Many design goals can be inferred from the non-functional requirements or from the application domain.

- **User friendly**: The system is user friendly because it is easy to use and understand.
- **Reliability**: Proper checks are there for any failure in the system if they exist.

3.1.1. Scenarios:

1.Social Media Advertising:

- Scenario: A digital marketing agency wants to optimize its social media advertising campaigns by targeting users who are likely to engage with their ads or make purchases.
- Application: The code can be used to predict potential connections between users who exhibit similar interests or purchasing behaviors. By leveraging link prediction algorithms, the agency can identify target audiences more effectively and tailor their advertising campaigns to reach the most relevant users.

2.Fraud Detection in Financial Transactions:

- Scenario: A financial institution wants to detect fraudulent activities in its network of financial transactions, such as money laundering or fraudulent transfers.
- Application: The code can be used to analyze the network of financial transactions and predict potential fraudulent connections between accounts or individuals. By employing link prediction algorithms, the institution can identify suspicious patterns or relationships indicative of fraudulent behavior and take proactive measures to prevent financial fraud.

Analysis and Design with UML:

A UML diagram is a diagram based on the UML (Unified Modelling Language) with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system.

UML is a modern approach to modelling and documenting software. In fact, it’s one of the most popular business process modelling techniques. It is based on diagrammatic representations of software components. As the old proverb says: “a picture is worth a thousand words”. By using visual representations, we are able to better understand possible flaws or errors in software or business processes.

3.1.2 Use Case Diagram

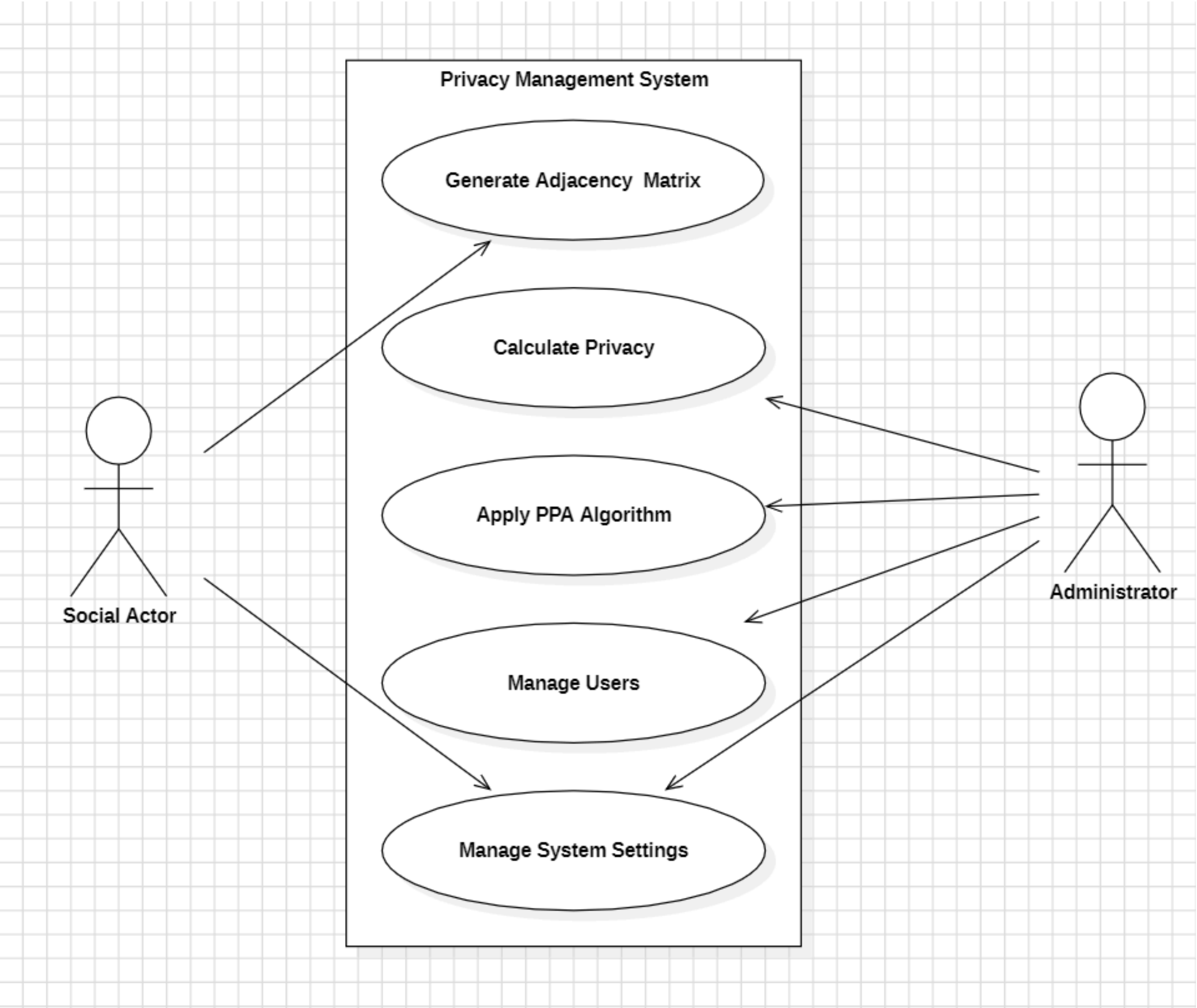
In the Unified Modelling Language (UML), a use case diagram can summarize the details of your system’s users (also known as actors) and their interactions with the system.This UML diagram has three primary components that we can see:

- Functional Requirements
- Actors
- Relationships

The use case diagram for a privacy management system. Here's a breakdown of the elements and their relationships:

Actors:

- **Social Actor:** This actor represents a social media user who interacts with the privacy management system. They can perform actions like creating a profile, managing their privacy settings, and accessing educational resources.
- **Administrator:** This actor represents an administrator of the privacy management system. They are responsible for maintaining the system settings and potentially managing user data (depending on the specific system).



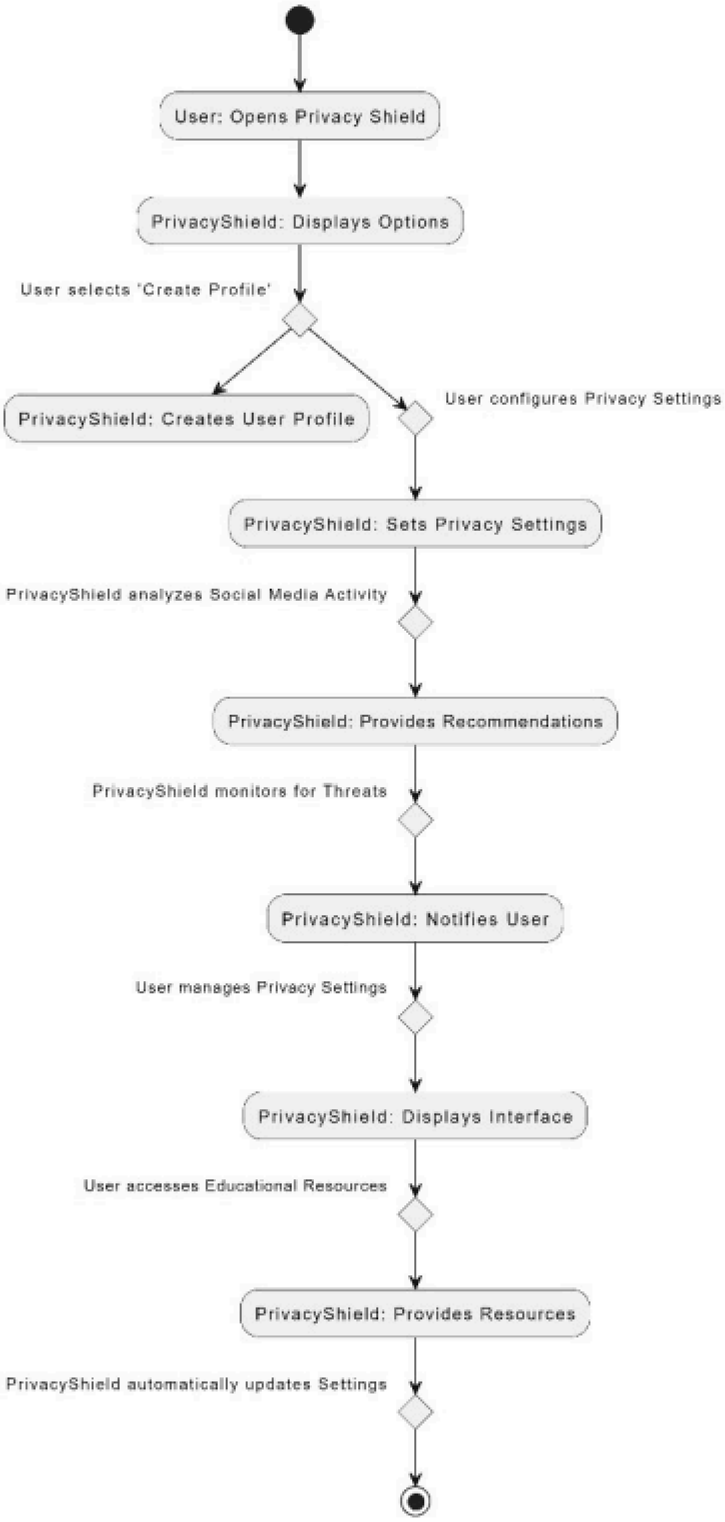
Fig(a): Use Case Diagram for the project

3.1.3. Activity Diagram:

An activity diagram is a visual representation of the flow of activities or steps in a system or process.Activity diagram is basically a flowchart to represent the flow from one activity to another activity.

The activity diagram depicts a system with a privacy shield that a user can interact with to manage their social media privacy. Let's walk through the user journey:

1. **User opens the Privacy Shield:** The process begins with the user initiating the interaction by opening the privacy shield.
2. **Privacy Shield displays options:** Upon opening, the privacy shield presents the user with various options.
3. **User selects "Create Profile"**: From the provided options, the user selects "Create Profile".
4. **Privacy Shield creates user profile:** The privacy shield then takes action by creating a user profile.
5. **User configures privacy settings:** Next, the user configures their privacy settings on the newly created profile.
6. **Privacy Shield sets privacy settings:** The privacy shield applies the user-configured privacy settings.


User: Opens Privacy Shield
PrivacyShield: Displays Options
User selects 'Create Profile'
PrivacyShield: Creates User Profile
User configures Privacy Settings
PrivacyShield: Sets Privacy Settings
PrivacyShield analyzes Social Media Activity
PrivacyShield: Provides Recommendations
PrivacyShield monitors for Threats
PrivacyShield: Notifies User
User manages Privacy Settings
PrivacyShield: Displays Interface
User accesses Educational Resources
PrivacyShield: Provides Resources
PrivacyShield automatically updates Settings

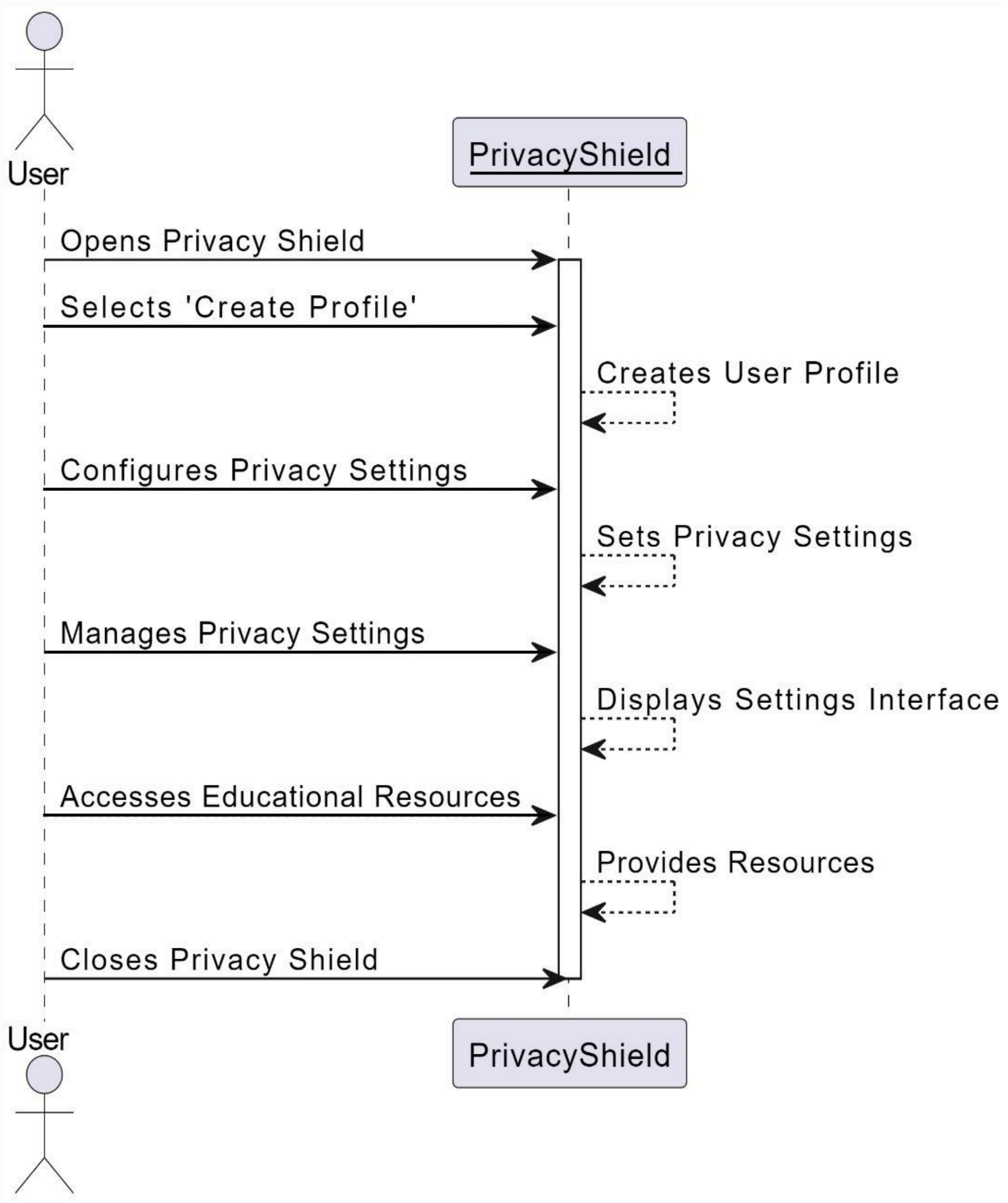
Fig(b): Activity Diagram for the project

3.1.4 Sequence Diagram:

A sequence diagram is the most commonly used interaction diagram. It simply depicts interaction between objects in a sequential order i.e., the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function.

It depicts how a user interacts with a privacy shield to access educational resources. Here's a breakdown of the sequence:

1. **User initiates:** The user initiates the interaction by opening the privacy shield.
2. **Select action:** The user selects the action "create profile".
3. **System creates profile:** The privacy shield creates a user profile based on the user's selection.
4. **Configure settings:** The user configures their privacy settings on the newly created profile.
5. **Set user preference:** The privacy shield sets the user's privacy settings according to their configuration.
6. **Manage settings (optional):** The user can manage their privacy settings again if needed. The diagram shows this step as optional.
7. **Display settings:** The privacy shield displays the settings interface to the user for confirmation or further changes.
8. **Access resources:** The user takes action to access educational resources.
9. **Provide resources:** The privacy shield provides the user with access to the educational resources.
10. **Close privacy shield:** The user closes the privacy shield.



Fig(c): Sequence Diagram for the project

3.1.5. Class diagram:

The class diagram shows how different classes work together to generate adjacency lists for a graph. Here's a breakdown of the classes and their relationships:

- Graph: This class represents the overall graph data structure. It likely has attributes to store the number of vertices and potentially the edges themselves.
- Edge: This class likely represents an edge in the graph. It might have attributes such as source and destination vertices.
- PPA Algorithm: This class implements the PPA algorithm, which is used to generate adjacency lists from a graph. It likely has a method ppa_algorithm that takes graph parameters as input and generates the adjacency list.
- Privacy Calculator: This class seems to be a helper class that calculates privacy metrics, possibly related to edge privacy. It has a method calculate_privacy that takes the edges, number of vertices, and another parameter (possibly number of edges) as input.

There is an undirectional association between Graph and PPA Algorithm. This means that these two classes collaborate in some way, possibly with the PPA Algorithm using the graph information to generate the adjacency list.

There is an undirectional association between Graph and Privacy Calculator. This suggests that these two classes also collaborate, possibly with the Privacy Calculator using the graph information to compute privacy metrics.

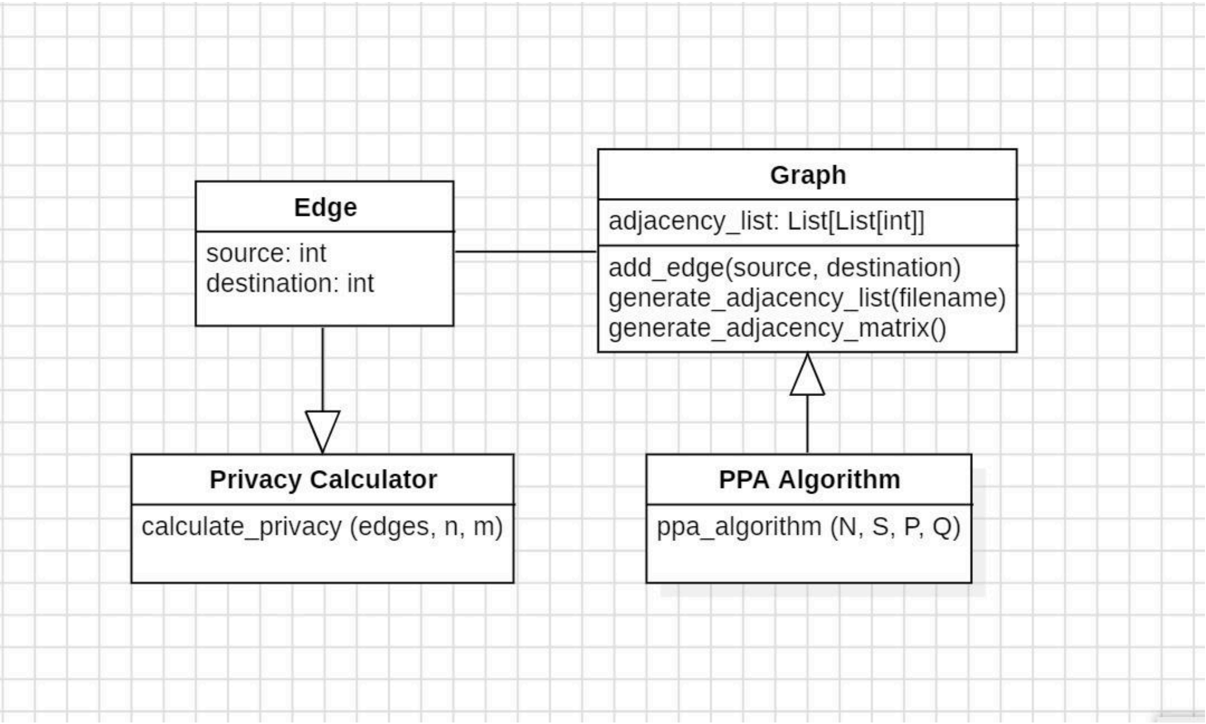

Edge
source: int
destination: int
Graph
adjacency_list: List[List[int]]
add_edge(source, destination)
generate_adjacency_list(filename)
generate_adjacency_matrix()
Privacy Calculator
calculate_privacy (edges, n, m)
PPA Algorithm
ppa_algorithm (N, S, P, Q)

Fig (d) :Class Diagram for the project

4.IMPLEMENTATION DETAILS

4.1 Software Environment

1.Programming Language:

- Python: The code is written in Python, a versatile and widely-used programming language known for its simplicity and readability.

2.Libraries and Modules:

- pandas: Used for data manipulation and analysis, particularly for handling input files and tabular data.
- numpy: Essential for numerical computations and array operations, often used for mathematical calculations and data manipulation.
- networkx: A Python library for the creation, manipulation, and study of complex networks (i.e., graphs), including social network graphs.
- tqdm: Provides a progress bar for tracking the progress of iterative processes, helpful for monitoring the execution of loops or tasks.
- matplotlib: Used for creating visualizations and plots, which can be useful for analyzing and interpreting graph data.
- scikit-learn: Provides machine learning algorithms and tools for tasks such as classification, regression, and model evaluation. In this case, it may be used for logistic regression and performance evaluation metrics.

3.File Input/Output:

- Input files: The code reads input data from external files named "Output.edges" and "Initial.edges", which presumably contain information about the edges (links) of the social network graph.

4.Execution Environment:

- Operating System: The code can be executed on various operating systems such as Windows, macOS, or Linux, as long as Python and the required libraries are installed.
- Python Interpreter: The code is executed using a Python interpreter (e.g., CPython) that interprets and executes the Python code instructions.

5.Development Tools:

- Integrated Development Environment (IDE): Developers may use IDEs such as PyCharm, Visual Studio Code, or Jupyter Notebook for writing, testing, and debugging the code.
- Text Editor: Alternatively, developers can use a simple text editor like Sublime Text or Atom for editing the code files.\

List Functions:

Lists are one of 4 built-in data types in Python used to store collections of data,Lists are used to store multiple items in a single variable.Lists are created using square brackets.
Example: Create a List: thislist = ["apple", "banana", "cherry"].
Few functions of lists are

Range: The range() function returns a sequence of numbers, starting from 0 by default, and increments by 1 (by default), and stops before a specified number.

Append: A call to .append() will place new items in the available space.

Remove: The remove() method removes the first matching element (which is passed as an argument) from the list.

Files :

Files are named locations on disk to store related information. They are used to permanently store data in a non-volatile memory (e.g. hard disk).
Since Random Access Memory (RAM) is volatile (which loses its data when the computer is turned off), we use files for future use of the data by permanently storing them.

When we want to read from or write to a file, we need to open it first. When we are done, it needs to be closed so that the resources that are tied with the file are freed.

Hence, in Python, a file operation takes place in the following order:

- Open a file
- Read or write (perform operation)
- Close the file

Open : To open a file, use Python's built-in open() method. This function returns a file object, often known as a handle, which can be used to read or change a file.

When we open a file, we can choose the mode. We choose whether to read r, write w, or add a to the file in mode.

Write : In order to write into a file in Python, we need to open it in write w or append a mode.
Read: To read a file in Python, we must open the file in reading r mode.

Close : Closing a file will free up the resources that were tied with the file. It is done using the close() method available in Python.

NetworkX Functions:

NetworkX is a Python-based software package for creating, manipulating, and studying complex networks' structure, dynamics, and function. It's used to investigate big, complicated networks that are represented as graphs with nodes and edges. We can load and store complex networks with networkx. We may create a variety of random and traditional networks, study their structure, establish network models, create new network algorithms, and draw them.

nx.Graph(edgeList) : A Graph is a collection of nodes (vertices) along with identified pairs of nodes (called edges, links, etc). The function nx.Graph() generates a graph for given edge list.

The following Networkx function are used to predict edges in a network:

- Triadic closure
- Jaccard Coefficient
- Resource Allocation Index
- Adamic Adar Index
- Preferential Attachment

Triadic closure:

If two vertices are connected to the same third vertices, the tendency for them to share a connection is Triadic Closure

comm_neighb(X, Y) = |N(X) ∩N(Y)|, where N(X) is the set of all neighbors of X.

Jaccard Coefficient:

It's calculated by dividing the total number of neighbors by the number of common neighbors. It is defined as the size of the intersection divided by the size of the union of two finite sample sets and is used to determine how similar two finite sample sets are.

The Networkx built-in function jaccard coefficient always returns a list of three tuples (u, v, p), where u, v is the new edge that will be added next with a probability measure of p. (p is the Jaccard Coefficient of nodes u and v).

nx.jaccard_coefficient(G)

Metrics:

Accuracy score: This function calculates subset accuracy in data set: the set of labels predicted for a sample must exactly match the corresponding set of labels in y true.

sklearn.metrics.accuracy_score (y_true, y_pred)

Precision score:

The precision is the ratio tp / (tp + fp) where tp is the number of true positives and fp the number of false positives. The precision is intuitively the ability of the classifier not to label as positive a sample that is negative.

The best value is 1 and the worst value is 0.

sklearn.metric.precision_score(y_true, y_pred)

Recall:

The recall is the ratio tp / (tp + fn) where tp is the number of true positives and fn the number of false negatives. The recall is intuitively the ability of the classifier to find all the positive samples.

The best value is 1 and the worst value is 0. sklearn.metrics.recall_score(y_true, y_pred)

F1 score:

The F1 score can be thought of as a harmonic mean of precision and recall, with the best value being 1 and the poorest being 0. Precision and recall both make an equal proportion to the F1 score. The F1 score is calculated as follows:

F1 = 2 * (precision * recall) / (precision + recall)

sklearn.metrics.f1_score(y_true, y_pred)

4.2. Software technologies:

1.Python:

- Python is the primary programming language used for implementing the code. It offers simplicity, readability, and a vast ecosystem of libraries and tools suitable for data analysis, machine learning, and network analysis tasks.

2.pandas:

- pandas is a Python library widely used for data manipulation and analysis. In this code, pandas may be used for handling input files, processing tabular data, and performing data transformations.

3.numpy:

- numpy is a fundamental package for numerical computing with Python. It provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays efficiently.

4.networkx:

- networkx is a Python library used for the creation, manipulation, and study of complex networks or graphs. It provides functionalities for generating graphs, computing network properties, and performing graph algorithms, which are essential for analyzing social network graphs in this code.

5.tqdm:

- tqdm is a Python library that adds a progress bar to loops and iterables, making it easier to track the progress of tasks. It is used in this code to provide a visual indication of the progress when iterating over edges or performing computations.

6.matplotlib:

- matplotlib is a plotting library for Python used to create static, interactive, and animated visualizations. In this code, matplotlib may be utilized for visualizing graph structures, plotting accuracy metrics, or displaying other relevant data visualizations.

7.scikit-learn:

- scikit-learn is a machine learning library for Python that provides simple and efficient tools for data mining and data analysis tasks. In this code, scikit-learn may be employed for implementing machine learning models, such as logistic regression, and for evaluating model performance using classification metrics.

4.3. ALGORITHM

In our project, the privacy-preservation in online social network data sharing is formulated as knapsack problem. We propose two social network data disclosure methods to solve this problem. Following are the two methods :
1. Privacy preserving algorithm.
2. Social relation based multidimensional knapsack problem algorithm. (Sd-kp).

4.3.1.Privacy Preserving Algorithm (PPA) :

- In this algorithm we are co-relating the social network privacy preserving problem to the Knapsack problem and then solving it.
- We are considering every edge of the Social network graph as an item in a knapsack.
- The total contribution of the selected items to social actor n's secret as the self privacy disclosure rate is considered as the weight of the edge/item.
- The utility of the selected items/nodes is considered as profit gained.
- The aim is to find the maximum utility possible with the minimum self privacy disclosure rate. i,e

$$max(x)= \sum_{i=1}^{|E_n|} p_i x_i$$

$$\text{s.t} \qquad \Phi_N(u_k, s_{kj}, x) \leq \theta_{k,j},$$

$$j=1, \ldots, |S_{u_k}|, k = 1, \ldots, |V_N|,$$

$$x_i \in \{0,1\}, i=1, \ldots, |E_N|$$

$$\text{where } \theta_{k,j} = \exp(\epsilon) \Pr\{t_{uk}(s_{k,j}) = 1\} + \delta_{k,j}$$

Algorithm steps of PPA:

Step-1: Initialize

- S = {S1 , S2 *S* , S3…..Sn} (Secrets of a social actor)
- N = {N1 , N2 *N* , N3…..Nn} (Neighbor of a social actor)

- θ = { 1 , 2 , 3….. θ θ θ θ θn} (Secret threshold list)

Step-2: Calculate Privacy using Jaccard Coefficient P = { 1 , 2 , 3….. *p* n}

Pj(eu,v) = |*Nu* ∩ *Nv*| |*Nu* ∪ *Nv* |

For every social relation existing between social actor u and social actor v :
P(Social actor u, Social actor v) = *Total Common neighbors of u and v*
Total neighbors of u or v

Step-3: Initialize

- C = Vn (Vertex set of social actors)
- Sel = Ø
- Vmax = 0
- *l* = [1, 2, 3,....,n] → Do Step-4 to Step-10 until *l* not equal to Ø

Step-4 : Initialize

- ρmax= -1
- S = -1
- Wsel = Ø Do Step- 5 to Step-8 for every i in *l*

Step-5: J = [1, 2, 3,....,n]
Do Step-6 for every j in J

Step-6: Calculate wj using the below formula
Wj ← |C∩*Ni* ∩*S j* | |C∩*Ni* |
Step 7 : Calculate ρ using below formula

ρ← *P i k*=1 *m* ∑ *wj* / θ *j*

Do Step- 8 if ρ > ρmax

Step 8 :

- ρmax = ρ
- s=i
- Wsel = Wj

Do Step 9 if wj<= j θ for every j in {1,2,3,...m}

Step-9:

- Sel = Sel or {s}
- C = C and Ns
- pmax = pmax + ps

Step-10: remove s from *l*

Step-11 : Return pmax , Sel

Explanation of Privacy Preserving Algorithm :

- We have used the greedy approach to solve this knapsack edge masking problem.
- The main idea is to select the edges of the nodes which have high utility and with minimal leakage of information about private attributes while satisfying all the privacy requirements.
- Initially the utility of all the edges is pre-computed.
- In every iteration weights of all secrets is computed and for every edge we will find ρ, which is the ratio of edge utility to the total sum of ratios of weights of secrets to its threshold value. The edge with maximum ρ and satisfying all privacy constraints is selected.

4.3.2 Social relation based d-kp (S-dkp):

The fundamental cause of the high complexity is because self-privacy disclosure takes into account all of the relationships between qualities and social relationships. On the one hand, omitting the correlations and treating all public information as (conditionally) independent may reduce privacy protections because two public attributes/social relations may give more information than the sum of the information presented individually. However, it will also simplify the problem by determining the weight of each public attribute or social relationship. The aim is to find the maximum utility possible with the minimum self privacy disclosure rate. i,e

$$max(x) = \sum_{i=1}^{|E_n|} p_i x_i$$

$$\text{s.t} \quad \Phi_N(u_k, s_{kj}, x) \le \theta_{k,j},$$

$$j=1, \ldots, |S_{u_k}|, k = 1, \ldots, |V_N|,$$

$$x_i \in \{0,1\}, i=1, \ldots, |E_N|$$

$$\text{where } \theta_{k,j} = \exp(\epsilon) \Pr\{t_{uk}(s_{k,j}) = 1\} + \delta_{k,j}$$

Algorithm steps of s-dkp :

Step-1: Initialize

- secret neighbourhood setlist S = {S1 , S2 *S* , S3…..Sn} ,
- item neighbourhood setlist N = {N1 , N2 *N* , N3…..Nn}, and
- secret threshold list θ = { 1 , 2 , 3….. θ θ θ θ θn}

Step-2: Calculate Privacy using Jaccard Coefficient P = { 1 , 2 , 3….. *p* n}

Pj(eu,v) = |*Nu* ∩ *Nv*| |*Nu* ∪ *Nv* |

For every social relation existing between social actor u and social actor v :
P(Social actor u, Social actor v) = *Total Common neighbors of u and v Total neighbors of u or v*

Step-3: Initialize

- C = Vn (Vertex set of social actors)
- Sel = Ø
- pmax = 0
- *l* = [1, 2, 3,....,n]

Do Step-4 to Step-13 until *l* not equal to Ø

Step-4 : Initialize

- ρmax= MIN_INT
- S = -1
- Isel = Ø Do Step- 5 to Step-13 for every i in *l*

Step-5:

- J = N[i] (friends of i)

Do Step-6 to Step 8 for j in J

Step-6 : Calculate Information gain for node i and node j using below formula

- Ii = 0
- Ij =0
- Ii = log(1/(nCr(n-1, len(N[i]))))
- Ij = log(1/(nCr(n-1, len(N[N[i][j]]))))

Step-7 : Calculate ρ using below formula

- ρ = P[i][j]/(Ii+Ij)

Do step 8 if ρ> ρmax

Step-8 :

- s = N[i][j]
- ρ = ρmax
- Isel = Ij

Step-9 :

- flag = True
- J = θ[s]

Do step-10 for every j in J , if Isel > log(θ[s][j])

Step-10 :

- flag=False

Do step 11 if flag=True

Step-11:

- Sel = Sel or {s}
- pmax= pmax + sum(ps)

Step 12: return Sel, pmax

Explanation of S-dkp :

- We have used the greedy approach to solve this multidimensional knapsack edge masking problem.
- The main idea is to select the edges of the nodes which have high utility and with minimal leakage of information about social relations while satisfying all the privacy requirements.
- Initially the utility of all the edges is pre-computed.
- In every iteration mutual information disclosed of all the edges is computed and for every edge we will find ρ, which is the ratio of edge utility to the total sum of information disclosure of both the nodes involved in the social relation. The edge with maximum ρ and satisfying all privacy constraints is selected.

Inference attack techniques:

Triadic Closure:

Triadic Closure occurs when two vertices are connected to the identical third vertices.

comm_neighb(*X*, *Y*) = |*N*(*X*) *\cap N*(*Y*)|, *where N*(*X*) *is the set of all neighbors of X*

Jaccard:

It is calculated by dividing the total number of neighbors by the number of common neighbors. It is defined as the size of the intersection divided by the size of the union of two finite sample sets, and it is used to quantify the similarity between two finite sample sets.

Jaccard Coefficient(*X*, *Y*) = |*N*(*X*) *\cap N*(*Y*)|/|*N*(*X*) *\cup N*(*Y*)|

Resource Allocation:

Research Allocation Index outperforms a number of similarity-based approaches for predicting missing links in a complex network with less time complexity. It's a fraction of a resource that a node can send to another node via their shared neighbors.

Research Allocation Index(*X*, *Y*) = Σ *u \epsilon N*(*X*) *\cap N*(*Y*) 1/|*N*(*u*)|

Adamic Adar Index:

In 2003, this metric was established to anticipate missing links in a network based on the number of common links between two nodes. It is calculated as follows:

Adamic Adar Index(*X*, *Y*) = Σ *u \epsilon N*(*X*) *\cap N*(*Y*) 1/*log*(|*N*(*u*)|)

Preferential Attachment:

The more linked a node is, the more likely it is to obtain new linkages, which is referred to as preferential attachment. More neighbors are attracted to nodes with a higher degree.

Preferential Attachment(*X*, *Y*) = |*N*(*X*)|. |*N*(*Y*)

5.TESTING

1.input Data Validation:

- Ensure that the input files ("Output.edges" and "Initial.edges") contain valid data representing the edges of the social network graph. Verify that the data format is consistent and appropriate for processing.

2.Graph Construction:

- Confirm that the networkx graph (G) is constructed correctly from the input edges and that it accurately represents the social network topology.

3.Link Prediction:

- Test each link prediction algorithm (e.g., triadic closure, Jaccard coefficient, resource allocation) individually to verify that it produces plausible predictions.
- Manually inspect a subset of predicted edges to assess their relevance and coherence with the existing network structure.

4.Accuracy Evaluation:

- Calculate the accuracy of each link prediction algorithm using known initial edges as ground truth.
- Verify that the accuracy metrics (e.g., percentage of correctly predicted edges) are calculated correctly and reflect the performance of the algorithms.

5.Unit Tests:

- Write unit tests for critical functions in the code, such as the link prediction algorithms and accuracy evaluation metrics.
- Test edge cases, such as empty input files or graphs with minimal nodes and edges, to ensure robustness.

6.Integration Testing:

- Run the entire code pipeline with sample input data and verify that it executes without errors.
- Compare the output accuracy metrics with expected values based on the input data and predictions.

7.Performance Testing:

- Assess the computational performance of the code, particularly for large social network graphs.
- Measure the execution time of key operations, such as graph construction and link prediction, and ensure that they meet acceptable performance standards.

8.Error Handling:

- Introduce intentional errors, such as invalid input file formats or missing files, and verify that the code handles them gracefully with informative error messages.

9..Documentation Verification:

- Review the code documentation to ensure that it accurately describes the purpose of each function, input parameters, and expected outputs.
- Confirm that the documentation is clear and understandable for users who may need to use or modify the code in the future.

Test Case ID	**Requirement Name**	**Tase Case Description**	**Expected Output**	**Result**
TC01	List Users	The administrator attempts to view a list of users currently in the system.	The system displays a list of all users with their usernames,IDS ,or other relevant profile information.	Pass
TC02	Configure privacy Settings	The administrator Attempts to define default privacy settings for all users.	The Systems saves the new privacy settings configuration and displays a confirmation message	Pass
TC03	Manage Access controls	The administrators attempts to grant an administrator access to view user data	The system updates the access control settings and displays a confirmation message.	Pass
TC04	Generate Adjacency Matrix	The social actor attempts to generate a visual representation of their social connections when they have no connections.	The system displays a message indicating there are no social connections to represent.	Pass
TC05	Calculate Privacy	The social actor request to calculate a privacy metric based on their profile and social connections	The system calculates a privacy score or report and displays it to the social actor..	Pass

6. CONCLUSION

Our Project " Privacy Shield : Securing Social Sphere" lays the foundation for further research and development in this crucial area. Future work can explore the effectiveness of the PPAs against various inference attacks, integrate user-controllable privacy settings, and establish robust evaluation methods. Through continued research and refinement, this approach has the potential to revolutionise social media privacy, fostering a safer and more secure online environment for everyone.

Social media has become an ingrained part of our lives, but privacy concerns loom large. This project addressed the critical issue of privacy breaches by proposing a novel approach to balancing user self-disclosure and privacy.

We introduced two privacy-preserving algorithms (PPAs) and Social relation based d-kp (S-dkp) that utilise multidimensional knapsack problems and greedy heuristics to make informed decisions about what information users share. These algorithms target high utility for users while considering social relations and maintaining low computational complexity.

By implementing these PPAs, social media platforms can empower users to control their online presence and safeguard their privacy without sacrificing the benefits of self-disclosure.

We investigated the online social network data sharing with defense against the inference attack and formulated the optimization problem which maximizes the utility with privacy guarantee and user privacy concerns.The PPA algorithm always has the best performance compared to that of sd-kp in terms of utility. In terms of computational complexity SDKP outperforms PPA.

Future Work : Implement our algorithms on larger dataset and solve privacy leakage problem in terms community information.

REFERENCES

- "Privacy-Preserving Machine Learning for Social Data" by Seraphin Calo, Kamalika Chaudhuri, Anand D. Sarwate.

- "Privacy Shield Principles" - Detailed information on the principles of the Privacy Shield framework.

- "Privacy Preserving Data Mining: Techniques, Applications and Trends" by Mehmet Kuzu, Ammar Mohammed, Yucel Saygin.

- "Privacy-Enhancing Technologies for Social Networks" by Frederik Armknecht, Anja Jerichow, Jens Lindemann.

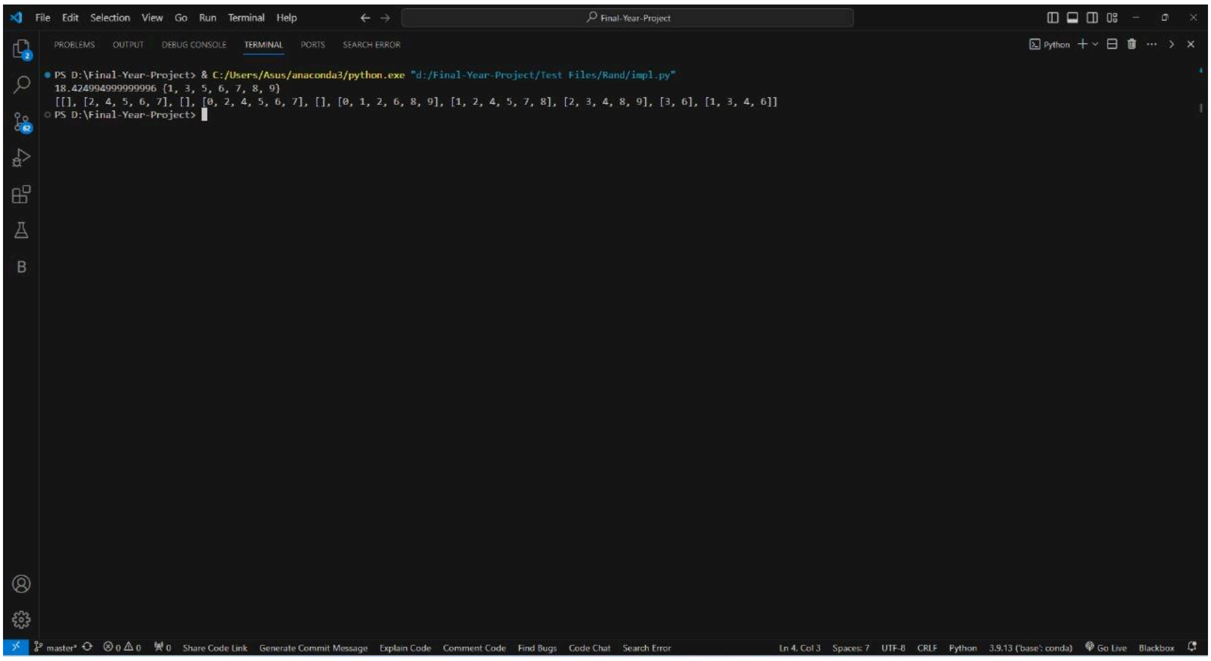
- Weihao Li and Hui Li Xidian University, Xi'an, China," LRDM: Local Record-Driving Mechanism for Big Data Privacy Preservation in Social Networks " IEEE First International Conference on Data Science in Cyberspace , 2016.

- Qian Wang , Member, IEEE, Yan Zhang, Xiao Lu, Zhibo Wang, Member, IEEE, Zhan Qin, Student Member, IEEE, and Kui Ren, Fellow, IEEE," Real-Time and Spatio-Temporal Crowd-Sourced Social Network Data Publishing with Differential Privacy", JULY/AUGUST 2018.

APPENDIX

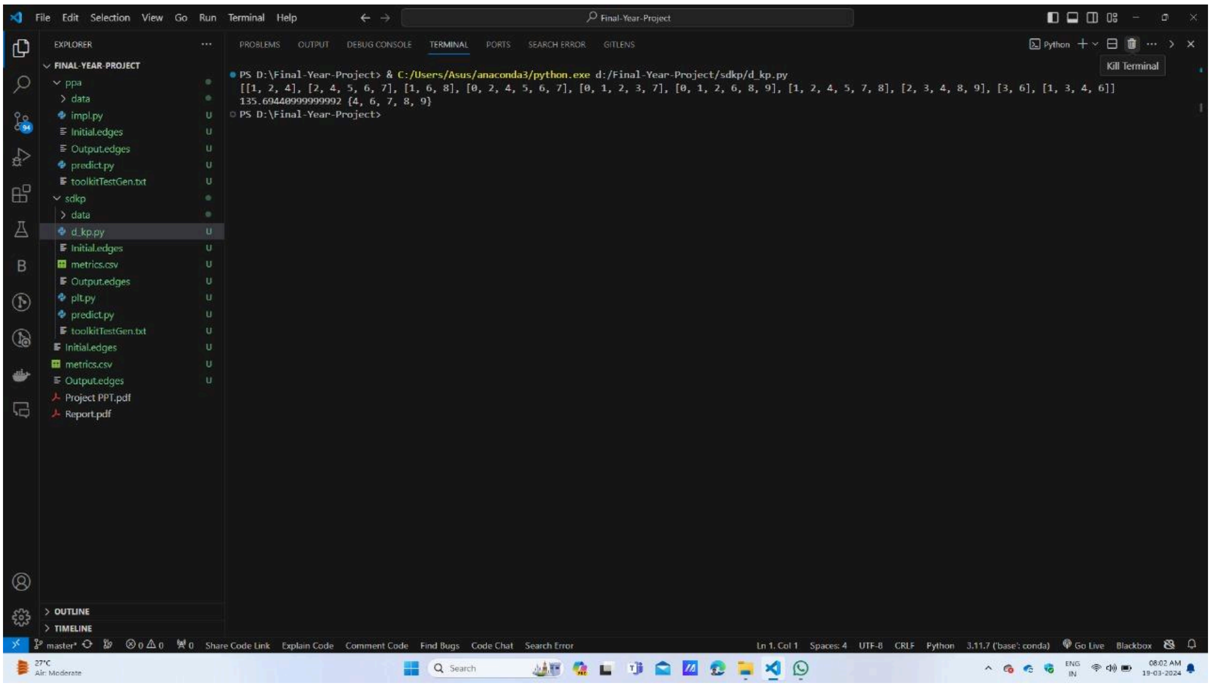
A. Input/ Output Screens:

(1st Algorithm Output):



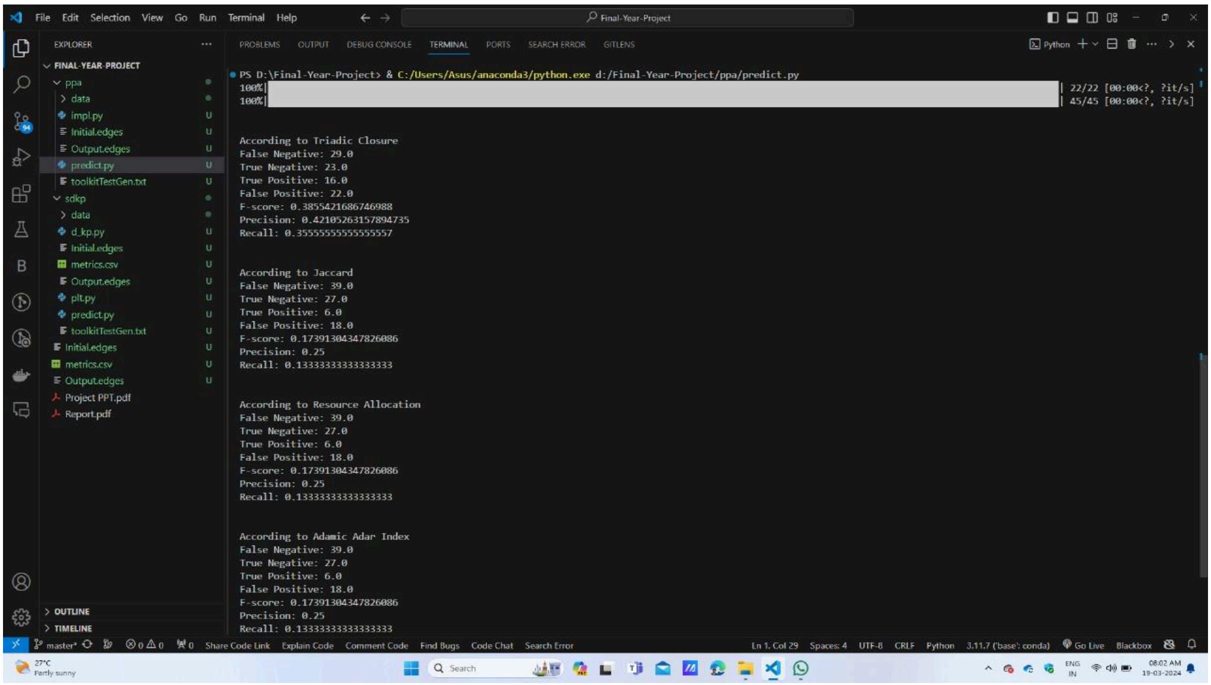
```
PS D:\Final-Year-Project> & C:/Users/Asus/anaconda3/python.exe "d:/Final-Year-Project/Test Files/Rand/impl.py"
18.424994999999996 {1, 3, 5, 6, 7, 8, 9}
[[], [2, 4, 5, 6, 7], [], [0, 2, 4, 5, 6, 7], [], [0, 1, 2, 6, 8, 9], [1, 2, 4, 5, 7, 8], [2, 3, 4, 8, 9], [3, 6], [1, 3, 4, 6]]
PS D:\Final-Year-Project>
```

(2 algorithm Output):



```
PS D:\Final-Year-Project> & C:/Users/Asus/anaconda3/python.exe d:/Final-Year-Project/sdkp/d_kp.py
[[1, 2, 4], [2, 4, 5, 6, 7], [1, 6, 8], [0, 2, 4, 5, 6, 7], [0, 1, 2, 3, 7], [0, 1, 2, 6, 8, 9], [1, 2, 4, 5, 7, 8], [2, 3, 4, 8, 9], [3, 6], [1, 3, 4, 6]]
135.69440999999992 {4, 6, 7, 8, 9}
PS D:\Final-Year-Project>
```

InferenceAttacks:



```
PS D:\Final-Year-Project> & C:/Users/Asus/anaconda3/python.exe d:/Final-Year-Project/ppa/predict.py
100%|████████████████████████████████████████| 22/22 [00:00<?, ?it/s]
100%|████████████████████████████████████████| 45/45 [00:00<?, ?it/s]


According to Triadic Closure
False Negative: 29.0
True Negative: 23.0
True Positive: 16.0
False Positive: 22.0
F-score: 0.3855421686746988
Precision: 0.42105263157894735
Recall: 0.35555555555555557


According to Jaccard
False Negative: 39.0
True Negative: 27.0
True Positive: 6.0
False Positive: 18.0
F-score: 0.17391304347826086
Precision: 0.25
Recall: 0.13333333333333333


According to Resource Allocation
False Negative: 39.0
True Negative: 27.0
True Positive: 6.0
False Positive: 18.0
F-score: 0.17391304347826086
Precision: 0.25
Recall: 0.13333333333333333


According to Adamic Adar Index
False Negative: 39.0
True Negative: 27.0
True Positive: 6.0
False Positive: 18.0
F-score: 0.17391304347826086
Precision: 0.25
Recall: 0.13333333333333333
```

B.Sample Code

- **Privacy preserving algorithms :**

(Allocating Nodes For Data)

```
# Initialize parameters
n = 10
m = 5
edges = []
with open("Test Files/Trial/Rand/data/edges.txt") as file1:
    for i in range(n):
        edges.append(file1.readline().split(" "))
        edges[i] = list(map(int, edges[i]))

# Compute neighbor nodes
N = [[] for _ in range(n)]
for i in range(n):
    for j in range(n):
        if edges[i][j] == 1 and i != j:
            N[i].append(j)

# Write neighbor nodes to file
with open("Initial.edges", "w") as file1:
    for i in range(len(N)):
        for j in range(len(N[i])):
            file1.write(str(i) + " " + str(j) + "\n")

# Read privacy values
privacy = []
with open("Test Files/Trial/Rand/data/privacy.txt") as file1:
    for i in range(n):
        privacy.append(file1.readline().split(" "))
        privacy[i] = privacy[i][0:-1]
        privacy[i] = list(map(float, privacy[i]))

# Compute privacy values for edges
P = [[] for _ in range(n)]
for i in range(n):
    for j in range(n):
        if edges[i][j] == 1 and i != j:
            P[i].append(privacy[i][j])
```

```
# Read secret data
secrets = []
with open("Test Files/Trial/Rand/data/secrets.txt") as file1:
    for i in range(n):
        secrets.append(file1.readline().split(" "))
        secrets[i] = list(map(int, secrets[i]))

# Compute secret information
S = [[] for _ in range(n)]
for i in range(n):
    for j in range(m):
        if secrets[i][j] == 1:
            S[i].append(j)

# Read threshold values
thresholdlist = []
with open("Test Files/Trial/Rand/data/thershold.txt") as file1:
    for i in range(n):
        thresholdlist.append(file1.readline().split(" "))
        thresholdlist[i] = thresholdlist[i][0:-1]
        thresholdlist[i] = list(map(float, thresholdlist[i]))

# Compute threshold values for secrets
Q = [[] for _ in range(n)]
for i in range(n):
    for j in range(m):
        if secrets[i][j] == 1:
            Q[i].append(thresholdlist[i][j])
            if Q[i][-1] == 0:
                Q[i][-1] = 0.1

# Initialize variables
C = [i for i in range(n)]
Sel = set()
Pmax = 0
l = [i for i in range(n)]

# Perform selection process
while len(l):
    Rmax = -1
    s = -1
    Wsel = {}
    w = [0 for _ in range(m)]
    for i in l:
        for j in range(m):
            X = [val for val in C if val in N[i]]
            if len(X):
                sec = S[X[0]]
            else:
                sec = []
            for val in X:
                sec = [v for v in sec if v in S[val]]
            numerator = len(sec)
            if len(X):
                w[j] = numerator / len(X)
            else:
                w[j] = 0
        S1 = sum([w[j] / Q[i][j] for j in range(len(Q[i]))])
        if S1 == 0:
            R = 0
        else:
            R = sum(P[i]) / S1
        if R > Rmax:
            s = i
            Rmax = R
            Wsel = w
        flag = True
        for j in range(len(Q[s])):
            if Wsel[j] > Q[s][j]:
                flag = False
                break
        if flag:
            Sel.add(s)
            C = [val for val in C if val in N[s]]
            Pmax += sum(P[s])
        if s in l:
            l.remove(s)

# Output results
print(Pmax, Sel)
OutputList = [[] for _ in range(n)]
for i in Sel:
    OutputList[i] = N[i]

with open("Output.edges", "w") as file1:
    for i in range(len(OutputList)):
        for j in range(len(OutputList[i])):
            file1.write(str(i) + " " + str(j) + "\n")
```

```
print(OutputList)
```