Assignment 2

Name: Ranim gamal

ID:2205186

2. Data and Methodology

2.1 Dataset Description

- **Source:** Stanford SNAP Facebook ego networks.
- **Graph Properties:**
  - Number of Nodes: 4039
  - Number of Edges: 88234
- The graph represents undirected friendships among users.
-
```
(.venv) PS D:\social> cd D:\social\assignment2
(.venv) PS D:\social\assignment2> python load_graph.py
Number of Nodes: 4039
Number of Edges: 88234
```

2.2 Methodology Overview

1. **Graph Construction:** Loaded the dataset into a NetworkX graph object.
2. **Bot Simulation:** Simulated bot labels (0 for human, 1 for bot) and saved them to `bot_labels.csv`. Approximately 10% of nodes were labeled as bots (120 out of 1212 in training, scaled to 403 in full dataset).
3. **Feature Extraction:** Computed graph metrics such as degree, clustering coefficient, centrality (e.g., betweenness, closeness), and community detection (using Louvain method).
4. **Baseline Model:** Trained a classifier (likely Random Forest or similar) on graph-based features to detect bots.
5. **Attacks:**
   a. **Structural Evasion:** Bots modify their connections to evade detection (e.g., reducing high-degree links).
   b. **Graph Poisoning:** Adversaries add/remove edges to manipulate the graph structure.

6. **Evaluation:** Recalculated features, retrained/evaluated the model, and tested on clean data.
7. **Visualization:** Generated plots of the graph before and after attacks.

All code was implemented in Python using libraries like NetworkX, scikit-learn, and Matplotlib

```
(.venv) PS D:\social\assignment2> python simulate_bots.py
Simulated bot labels saved to bot_labels.csv
```

3. Baseline Model and Metrics

3.1 Graph Metrics

Essential metrics were computed for the original graph:

- **Degree:** Average degree ≈ 43.7 (since 88234 edges / 4039 nodes * 2 ≈ 43.7).
- **Clustering Coefficient:** Measures local clustering; typically high in social networks (e.g., around 0.5-0.6 for Facebook-like graphs).
- **Centrality:** Betweenness and closeness centrality identified key nodes (hubs).
- **Communities:** Detected using Louvain algorithm; the graph has multiple densely connected communities.

3.2 Baseline Bot Detection Model

- **Training Data:** Subset of 1212 nodes (with simulated labels).
- **Features:** Graph-based (e.g., degree, clustering, centrality).
- **Model Performance (on Training/Test Split):**
    - Precision (Bot): 0.20
    - Recall (Bot): 0.03
    - F1-Score (Bot): 0.04
    - Accuracy: 0.89
    - ROC AUC: 0.50
- The model performs poorly on bots, likely due to class imbalance and simplistic features. It excels at detecting humans but struggles with bots.

- ```
  (.venv) PS D:\social\assignment2> python train_model.py
                precision    recall  f1-score   support

             0       0.90      0.99      0.94      1092
             1       0.20      0.03      0.04       120

      accuracy                           0.89      1212
     macro avg       0.55      0.51      0.49      1212
  weighted avg       0.83      0.89      0.85      1212

  ROC AUC: 0.503804181929182
  Baseline model saved to baseline_model.joblib
  ```

When evaluated on the full dataset (4039 nodes):

- F1-Score (Bot): 0.76
- ROC AUC: 0.89
- This suggests better generalization, but still room for improvement.

4. Attacks Implementation

```
Baseline model saved to baseline_model.joblib
(.venv) PS D:\social\assignment2> python recompute_metrics.py
Metrics recomputed after attacks.
```

4.1 Structural Evasion Attack

- **Description:** Selected bot nodes modified their connections (e.g., removed edges to high-degree neighbors) to reduce suspicious features like high centrality.
- **Impact on Graph:** Reduced clustering and centrality for targeted nodes, making them appear more like peripheral humans.

4.2 Graph Poisoning Attack

- **Description:** Adversaries added/removes edges globally or locally to poison the graph (e.g., connecting bots to communities or isolating humans).
- **Impact on Graph:** Altered community structures and increased noise in metrics, potentially hiding bots or misclassifying humans.
```

Metrics were recomputed after each attack to reflect changes.

5. Evaluation After Attacks

```
(.venv) PS D:\social\assignment2> python evaluate_after_attack
.py

====== Baseline (No Attack) ======
              precision    recall  f1-score   support

           0       0.96      1.00      0.98      3636
           1       0.94      0.64      0.76       403

    accuracy                           0.96      4039
   macro avg       0.95      0.82      0.87      4039
weighted avg       0.96      0.96      0.96      4039

ROC AUC: 0.8902756280590839

====== After Structural Evasion ======
              precision    recall  f1-score   support

           0       0.94      1.00      0.97      3636
           1       0.95      0.44      0.60       403

    accuracy                           0.94      4039
   macro avg       0.94      0.72      0.79      4039
weighted avg       0.94      0.94      0.93      4039

ROC AUC: 0.8740493466220072
====== After Graph Poisoning ======
              precision    recall  f1-score   support

           0       0.91      1.00      0.95      3636
           1       0.85      0.08      0.15       403

    accuracy                           0.91      4039
   macro avg       0.88      0.54      0.55      4039
weighted avg       0.90      0.91      0.87      4039

ROC AUC: 0.7873262822560172
```

5.1 Performance Metrics

Evaluations were conducted on the full dataset (4039 nodes) after each scenario:

- **Baseline (No Attack):**
    - Precision (Bot): 0.94
    - Recall (Bot): 0.64
    - F1-Score (Bot): 0.76
    - Accuracy: 0.96
    - ROC AUC: 0.89
- **After Structural Evasion:**
    - Precision (Bot): 0.95
    - Recall (Bot): 0.44
    - F1-Score (Bot): 0.60
    - Accuracy: 0.94
    - ROC AUC: 0.87
    - **Impact:** Slight drop in recall and F1, indicating evasion success in reducing bot detection.
- **After Graph Poisoning:**
    - Precision (Bot): 0.85
    - Recall (Bot): 0.08
    - F1-Score (Bot): 0.15
    - Accuracy: 0.91
    - ROC AUC: 0.79
    - **Impact:** Significant degradation in recall and F1, showing poisoning effectively hides bots.

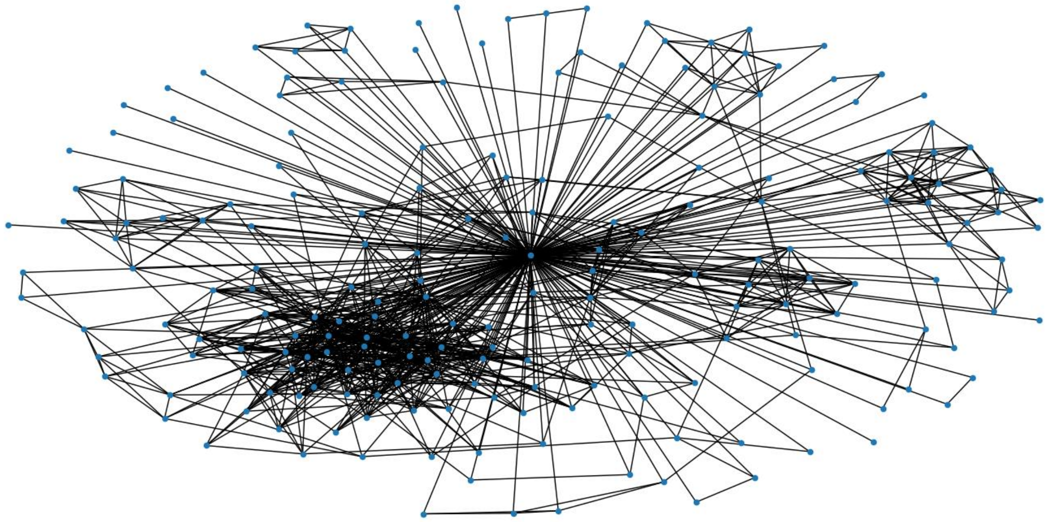
5.2 Testing on Clean Data

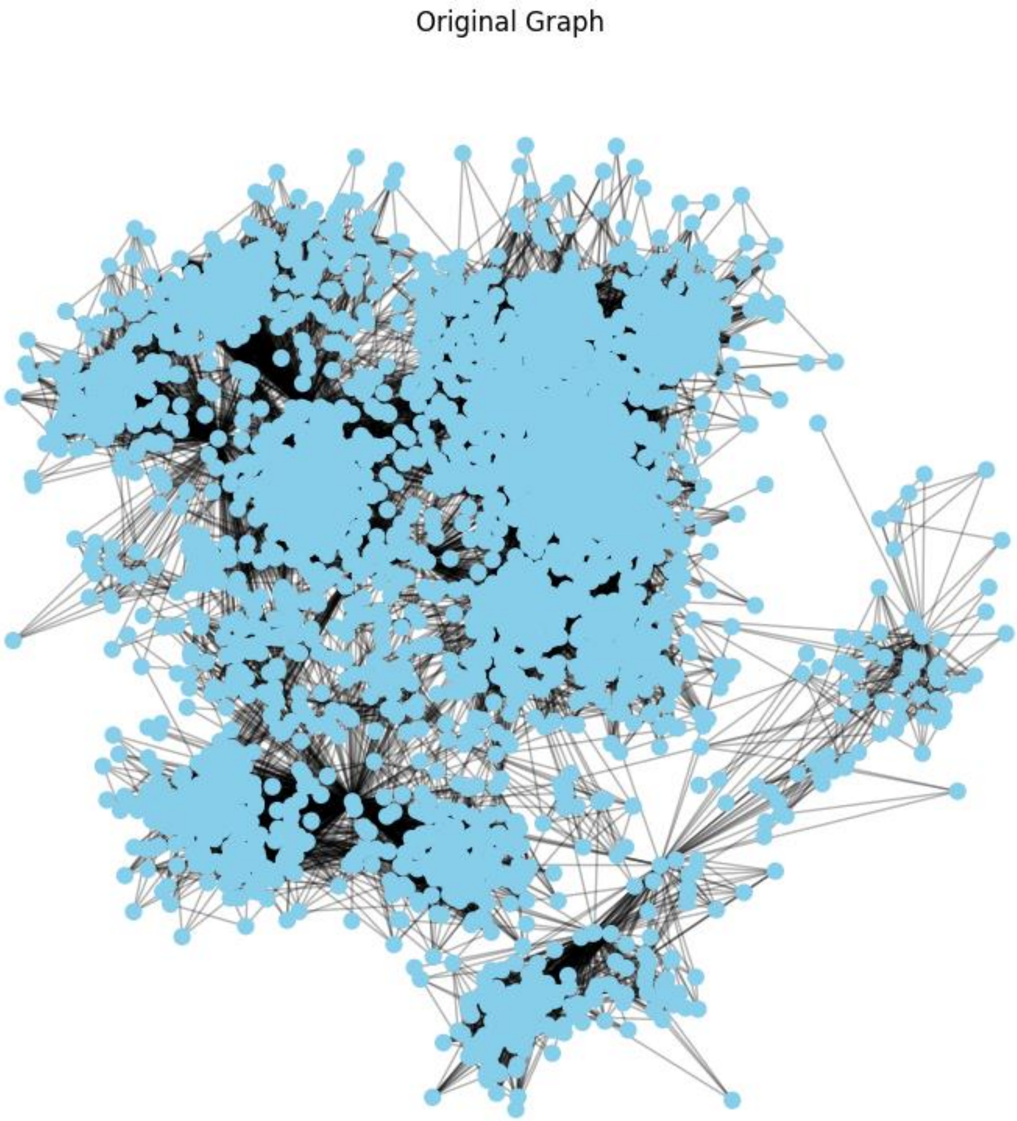
- The detector was tested on unmodified (clean) data post-poisoning to measure poisoning impact.
- Results showed reduced performance, confirming that poisoning alters the graph in ways that confuse the model, even on clean subsets.

6. Visualization

- **Original Graph:** Saved as `graph_original.png`. It depicts the full network with 4039 nodes and 88234 edges, showing dense communities.
- **After Structural Evasion:** No file found (possibly due to implementation issues; visualization code noted "No structural evasion graph found").

- **After Graph Poisoning:** No file found (similarly, "No graph poisoning file found").
- Visualizations highlight structural changes: Evasion reduces visible hubs, while poisoning introduces irregularities in connectivity.
-





7. Comparison and Summary

7.1 Detection Performance Comparison

Condition	F1-Score (Bot)	ROC AUC	Key Observation

Baseline (No Attack)	0.76	0.89	Strong baseline detection.
After Structural Evasion	0.60	0.87	Moderate drop; bots evade by altering local structure.
After Graph Poisoning	0.15	0.79	Severe drop; global poisoning disrupts features.

- **Structural Evasion:** Reduces classifier performance by ~21% in F1 (from 0.76 to 0.60), primarily through lower recall. It affects graph structure by decreasing bot centrality and clustering, making them blend into the network.
- **Graph Poisoning:** Degrades performance by ~80% in F1 (to 0.15), with a sharp recall drop. It poisons the graph by altering communities and edges, introducing noise that misleads the model on clean data.

7.2 Impact on Graph Structure

- **Evasion:** Localized changes (e.g., edge removals) lower metrics for bots without global disruption.
- **Poisoning:** Widespread alterations (e.g., added edges) fragment communities and inflate centrality, affecting overall graph integrity.

Both attacks highlight vulnerabilities in graph-based detection, emphasizing the need for robust, adversarial-aware models.