De-anonymizing Data

CompSci 590.03
Instructor: Ashwin Machanavajjhala
Outline

• Recap

• Algorithmically De-anonymizing Netflix Data

• Algorithmically De-anonymizing Social Networks
  – Passive Attacks
  – Active Attacks
Statistical Privacy (Untrusted Collector) Problem

Server

\( f (D_B) \)

Individual 1

\( r_1 \)

Individual 2

\( r_2 \)

Individual 3

\( r_3 \)

\ldots \)

Individual N

\( r_N \)
Randomized Response

• Flip a coin
  – heads with probability $p$, and
  – tails with probability $1-p$ ($p > \frac{1}{2}$)

• Answer question according to the following table:

<table>
<thead>
<tr>
<th></th>
<th>True Answer = Yes</th>
<th>True Answer = No</th>
</tr>
</thead>
<tbody>
<tr>
<td>Heads</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Tails</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Utility Analysis

• \( \pi \): True fraction of respondents answering “yes”
• \( p \): Probability coin falls heads

• \( Y_i = 1 \), if the \( i \)th respondent says “yes”
  \( = 0 \), if the \( i \)th respondent says “no”

\[
P(Y_i = 1) = (\text{True answer = yes AND coin = heads}) \text{ OR } \]
\[
(\text{True answer = no AND coin = tails})
\]
\[
= \pi p + (1-\pi)(1-p) = p_{\text{yes}}
\]
\[
P(Y_i = 0) = \pi (1-p) + (1-\pi)p = p_{\text{no}}
\]
Utility Analysis

• Suppose n1 out of N people replied “yes”, and rest said “no”

• What is the best estimate for π?

• Likelihood: \( L = \binom{n}{n_1} p_{\text{yes}}^{n_1} p_{\text{no}}^{n-n_1} \)

• Most likely value of π: (by setting \( dL/d\pi = 0 \))

\[
\hat{\pi} = \frac{n_1/n - (1-p)}{(2p-1)}
\]
Privacy

• Adversary’s prior belief: \( P(\text{Bob’s true answer is “yes”}) = \theta \)

• Suppose Bob answers “yes”.

\[
P(\text{Bob’s true answer is “yes” | Bob says “yes”}) = \frac{P(\text{Bob says “yes” AND Bob’s true answer is “yes”})}{P(\text{Bob says yes})}
\]

\[
= \frac{P(\text{Bob says “yes” | Bob’s true answer is “yes”})P(\text{Bob’s true answer is “yes”})}{P(\text{Bob says “yes” | Bob’s true answer is “yes”})P(\text{Bob’s true answer is “yes”}) + P(\text{Bob says “yes” | Bob’s true answer is “no”})P(\text{Bob’s true answer is “no”})}
\]

\[
= \frac{p\theta}{p\theta + (1-p)(1-\theta)} \leq \frac{p}{1-p} \theta
\]
Privacy

• Adversary’s prior belief:
  \[ P(\text{Bob’s true answer is “yes”}) = \theta \]

• Suppose Bob answers “yes”.
  Adversary’s posterior belief:
  \[ P(\text{Bob’s true answer is “yes” | Bob says “yes”}) \leq \frac{p}{1-p} \theta \]

Adversary’s posterior belief is always bounded by \(p/1-p\) times the adversary’s prior belief (irrespective of what the prior is)
Privacy vs Utility tradeoff

• When $p = 1$ (return truthful answer)
  – $p/1-p = \infty$: no privacy
  – $\hat{\pi} = n1/n =$ true answer

• When $p = \frac{1}{2}$ (return random answer)
  – $p/1-p = 1$: perfect privacy
  – We cannot estimate $\hat{\pi}$ since the answers are independent of the input.
  – $P_{yes} = \pi p + (1-\pi)(1-p) = \frac{1}{2}(\pi + 1 - \pi) = \frac{1}{2} = P_{no}$
Statistical Privacy (Trusted Collector) Problem

\[ f_{\text{private}}(D_B, \varepsilon) \]

Server

\[ D_B \]

Individual 1

\[ r_1 \]

Individual 2

\[ r_2 \]

Individual 3

\[ r_3 \]

\[ \ldots \]

Individual N

\[ r_N \]
Query Answering

$\mathcal{f}_{\text{private}}(D_B, \varepsilon)$

How many allergy patients?

Correlate Genome to disease

Individual 1
$r_1$

Individual 2
$r_2$

Individual 3
$r_3$

... 

Individual N
$r_N$

$D_B$

Lecture 2 : 590.03 Fall 13
Query Answering

• Need to know the list of questions up front

• Each answer will leak some information about individuals. After answering a few questions, server **will run out of privacy budget** and not be able to answer any more questions.

• *Will see this in detail later in the course.*
Anonymous/ Sanitized Data Publishing

I won't tell you what questions I am interested in!

writingcenterunderground.wordpress.com
Anonymous/ Sanitized Data Publishing

Answer any # of questions directly on $D_B'$ without any modifications.
Naïve Anonymization

• Remove identifying attributes from the data
  – E.g., Health Insurance Portability and Protection Act
  – Remove 18 attributes regarded as Personally Identifying Information (PII)

  – Name
  – Geography smaller than state
  – Date (more detailed than year)
  – Tel / Fax / email
  – SSN
  – IDs (Medical record / Health insurance / Accounts/ Certificates / Devices)
  – Vehicle ID / License plate
  – URLs / IP addresses
  – Full face photos / biometrics / genetic code
Can re-identify individuals using other datasets ... [Sweeney IJUFKS 2002]

- Name
- SSN
- Visit Date
- Diagnosis
- Procedure
- Medication
- Total Charge

- Zip
- Birth date
- Sex

- Name
- Address
- Date Registered
- Party affiliation
- Date last voted

Quasi Identifier

- 87% of US population uniquely identified using ZipCode, Birth Date, and Sex.

Medical Data    Voter List
Today’s class

• De-anonymization: Algorithms for identifying individual records and their sensitive values from naively anonymized data using background knowledge (usually other public datasets)

• Also called
  – Record Linkage
  – Entity Resolution
  – Fuzzy matching
  – ...
  – See tutorial and lectures in big-data class for more info ...
Outline

• Recap

• Algorithmically De-anonymizing Netflix Data

• Algorithmically De-anonymizing Social Networks
  – Passive Attacks
  – Active Attacks
### Netflix Dataset

<table>
<thead>
<tr>
<th>Users</th>
<th>Movies</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>1 1 1 1</td>
<td>5 5 1</td>
</tr>
<tr>
<td>4 2 1 4</td>
<td>3 3 5</td>
</tr>
<tr>
<td>3 1 2 4</td>
<td></td>
</tr>
</tbody>
</table>

**Support:** Set (or number) of non-null attributes in a record or column
De-anonymization

• Suppose we have a table AUX
  – <name/id>, set of known movie ratings
  – E.g., a single record about someone you know
  – IMDB ratings which are public

• Goal:
  Match individuals in the Netflix data to individuals in the AUX
General Strategy for De-Anonymization

• Inputs: Private database D, and auxiliary information AUX

• Pairwise Matching:
  Compute the similarity between candidate matching pairs
  – Based on attributes of the individuals

• Record Linkage:
  For each record in AUX, find the best matching record in D (or no match) ... or vice versa

• Blocking: Identify obvious non-matches (and exclude them ...)  
  – Remaining set of pairs are candidates matches
Pairwise Matching Features

• **Comparison Vector $\gamma$:**
  For two records $x$ and $y$, compute a vector *similarity scores* of component attribute.
  
  - [ Same rating for movie $X$,  
    Same rating for movie $Y$,  
    Number of Drama movies rated in both records, .... ]

• **Similarity scores**
  
  - Boolean (match or not-match)
  - Real values based on distance functions
  - Real values based on set or vector similarity
Summary of Matching Features

- Equality on a boolean predicate
- Edit distance
  - Levenstein, Smith-Waterman, Affine
- Set similarity
  - Jaccard, Dice
- Vector Based
  - Cosine similarity, TFIDF
- Alignment-based or Two-tiered
  - Jaro-Winkler, Soft-TFIDF, Monge-Elkan
- Phonetic Similarity
  - Soundex
- Translation-based
- Numeric distance between values
- Domain-specific

Useful packages:
- Simmetrics: http://sourceforge.net/projects/simmetrics/
Summary of Matching Features

- Equality on a boolean predicate
- Edit distance
  - Levenstein, Smith-Waterman, Affine
- Set similarity
  - Jaccard, Dice
- Vector Based
  - Cosine similarity, TFIDF

Good for Text (reviews/tweets), sets, class membership, ...

- Useful packages:
  - Simmetrics: http://sourceforge.net/projects/simmetrics/

Good for Names

- Alignment-based or Two-tiered
  - Jaro-Winkler, Soft-TFIDF, Monge-Elkan
- Phonetic Similarity
  - Soundex
- Translation-based
- Numeric distance between values
- Domain-specific

Handle Typographical errors

Useful for abbreviations, alternate names.
Netflix Paper Comparison Vector

• Given 2 records \( r \) in D and \( r_{\text{aux}} \) in AUX
  
  For each movie \( m \),

  \[
  \text{Sim}(r[m], r_{\text{aux}}[m]) = \begin{cases} 
  1 & \text{if } m \text{ was rated in both records with similar values and at similar times} \\
  0 & \text{otherwise}
  \end{cases}
  \]
Pairwise Match Score

Problem: Given a vector of component-wise similarities for a pair of records \((x,y)\), compute \(P(x \text{ and } y \text{ match})\).

Solutions:

1. Weighted sum or average of component-wise similarity scores.
   \[0.05 \times \text{Sim}[m1] + 0.02 \times \text{Sim}[m2] + 0.03 \times \text{Sim}[m3] + \ldots\]
   - How to pick weights.
     - Similarity on a rare attribute (rate movie) is more predictive of match than similarity on common attribute (blockbuster)

Threshold determines match or non-match.
- Hard to tune a threshold.
Pairwise Match Score

Problem: Given a vector of component-wise similarities for a pair of records \((x,y)\), compute \(P(x \text{ and } y \text{ match})\).

Solutions:

1. Weighted sum or average of component-wise similarity scores. Threshold determines match or non-match.

2. Formulate rules about what constitutes a match.
   - \((\text{Sim}[m1] > 0.7 \text{ AND } \text{Sim}[m2] > 0.8)\)
   - \(\text{OR } (\text{Sim}[m1] > 0.9 \text{ AND } \text{Sim}[m3] > 0.9)\)
   - Manually formulating the right set of rules is hard.
Fellegi & Sunter Model [FS, Science ‘69]

- Record pair: \( r = (x,y) \) in \( A \times B \)

- \( \gamma = \gamma(r) \) is a comparison vector
  - E.g., \( \gamma = [\text{“Is x.name = y.name?”}, \text{“Is x.address = y.address?”} ...] \)
  - Assume binary vector for simplicity

- \( M \): set of matching pairs of records
- \( U \): set of non-matching pairs of records
Fellegi & Sunter Model [FS, Science ‘69]

• $r = (x,y)$ is record pair, $\gamma$ is comparison vector, $M$ matches, $U$ non-matches

• Linkage decisions are based on:

$$R(r) = \frac{m(\gamma)}{u(\gamma)} = \frac{P(\gamma \mid r \in M)}{P(\gamma \mid r \in U)}$$

• Linkage Rule: $L(t_l, t_u)$

Lecture 2 : 590.03 Fall 13
Error due to a Linkage Rule

- **Type I Error:** $r = (x,y)$ in $U$, but the linkage rule calls it a match

  \[ P(L_{\text{match}}|U) = \sum_{\gamma \in \Gamma} u(\gamma) \cdot P(L_{\text{match}}|\gamma) \]

- **Type II Error:** $r = (x,y)$ in $M$, but the linkage rule calls it a non-match

  \[ P(L_{\text{non}}|M) = \sum_{\gamma \in \Gamma} m(\gamma) \cdot P(L_{\text{non}}|\gamma) \]
Optimal Linkage Rule

- $L^* = (t_l^*, t_u^*)$ is an optimal decision rule for comparison space $\Gamma$ with error bounds $\mu$ and $\lambda$, if
  
  - $L^*$ meets the type I and type II requirements
    
    $$P(L_{match}|U) \leq \mu, \quad P(L_{non}|M) \leq \lambda$$

  - $L^*$ has the least conditional probabilities of not making a decision. That is for all other decision rules $L$ (with error bounds $\mu$ and $\lambda$),
    
    $$P(L^*_{uncertain}|U) \leq P(L_{uncertain}|U)$$
    $$P(L^*_{uncertain}|M) \leq P(L_{uncertain}|M)$$
Many methods to compute pairwise matching

- **Fellegi Sunter**
  - Assume attributes are independent (Naïve Bayes Assumption) to simplify the problem
  - Use Training datasets to compute

- **Think of this as a machine learning classification problem**
  - Given some training data
  - Classify pairs of records as matches or non-matches
  - More advanced techniques use active learning

See [tutorial](#) for more details ...
Pairwise Match Score for Netflix Attack

Problem: Given a vector of component-wise similarities for a pair of records \((x,y)\), compute \(P(x\ and\ y\ match)\).

Solutions:

1. Weighted sum or average of component-wise similarity scores.
   \[
   0.05*\text{Sim}[m1] + 0.02*\text{Sim}[m2] + 0.03*\text{Sim}[m3] + \ldots
   \]
   
   - How to pick weights.
     - Similarity on a rare attribute (rate movie) is more predictive of match than similarity on common attribute (blockbuster)

Threshold determines match or non-match.

- Hard to tune a threshold.
General Strategy for De-Anonymization

• Inputs: Private database D, and auxiliary information AUX

• Pairwise Matching:
  Compute the similarity between candidate matching pairs
  – Based on attributes of the individuals

• Record Linkage:
  For each record in AUX, find the best matching record in D (or no match) ... or vice versa

• Blocking: Identify obvious non-matches (and exclude them ...)
  – Remaining set of pairs are candidates matches
Record Linkage

• Want to find the best matching between AUX and D ...
• ... but pairwise matching may result in 2 records in AUX having a high probability of matching the same record in D
Record Linkage

Solutions:

• Pick the best match such that second best match has a very low score ... (NETFLIX ATTACK SOLUTION)

• Bipartite Matching
  – Edge weights: log odd of matching
General Strategy for De-Anonymization

• Inputs: Private database D, and auxiliary information AUX

• Pairwise Matching:
  Compute the similarity between candidate matching pairs
  – Based on attributes of the individuals

• Record Linkage:
  For each record in AUX, find the best matching record in D (or no match) ... or vice versa

• Blocking: Identify obvious non-matches (and exclude them ...)
  – Remaining set of pairs are candidates matches
Blocking

• Number of pairs of records = |AUX| \times |D|
  – Techniques can be inefficient when these databases are very large

• Blocking
  – Identify pairs of records that don’t match (with very high probability)

• Example: minHashing
**minHash (Minwise Independent Permutations)**

- Let $F_x$ be a set of features for mention $x$
  - (predicates of) attribute values
  - character ngrams
  - optimal blocking functions ...

- Let $\pi$ be a random permutation of features in $F_x$
  - E.g., order imposed by a random hash function

- $\text{minHash}(x) = \text{minimum element in } F_x \text{ according to } \pi$
Blocking based on minHash

**Surprising property:** For a random permutation $\pi$,

$$P[\text{minHash}(x) = \text{minHash}(y)] = \frac{F_x \cap F_y}{F_x \cup F_y}$$

How to build a blocking scheme such that only pairs with Jacquard similarity $> s$ fall in the same block (with high prob)?

![Diagram showing the probability that (x,y) mentions are blocked together vs. similarity(x,y)]
Locality Sensitive Hashing Functions

Suppose \( d \) is a distance metric on a domain.

A family of functions \( F \) is said to be \((d_1, d_2, p_1, p_2)\)-sensitive if for all \( f \) in \( F \),
- If \( d(x,y) < d_1 \),
  then \( P[f(x) = f(y)] > p_1 \)
- If \( d(x,y) > d_2 \),
  then \( P[f(x) = f(y)] < p_2 \)
Locality sensitive family for Jaccard

• Jaccard distance = 1 - Jaccard similarity = \( 1 - \frac{F_x \cap F_y}{F_x \cup F_y} \)

• minHash is one example of locality sensitive family that can strongly distinguish pairs that are close from pairs that are far.

• The family of minHash functions is a \((d1, d2, 1-d1, 1-d2)-sensitive\) family for the Jaccard distance.
Blocking using minHashes

• Compute minHashes using $r \times k$ permutations (hash functions)

\[
signature(x) = \begin{array}{c}
\text{Band of } r \text{ minHashes} \\
\end{array}
\]

• Signature’s that match on **1 out of** $k$ bands, go to the same block.
minHash Analysis

• Let F be a (0.2, 0.6, 0.8, 0.4)-sensitive family of minHash functions
  – Pairs with Jaccard similarity > 0.8 are close, and similarity < 0.4 are far

• Let F1 be the family constructed using a “band of r=5 minHashes” (AND construction on F)
  – F1 is (0.2, 0.6, 0.8^5, 0.4^5) = (0.2, 0.6, 0.328, 0.01)-sensitive

‘Far’ objects are blocked together with very low probability
minHash Analysis

- F is (0.2, 0.6, 0.8, 0.4)-sensitive minHash
- F1 is a “band of r=5 minHashes” (AND construction on F)
  - F1 is (0.2, 0.6, 0.8^5, 0.4^5) = (0.2, 0.6, 0.328, 0.01)-sensitive

- Let F2 be the family constructed using “k = 20 bands of r=5 minHashes each” (OR construction on F1)
  - F2 is (0.2, 0.6, 1 – (1-0.8^5)^10, 1 – (1-0.4^5)^10)
    = (0.2, 0.6, 0.98, 0.09)-sensitive

‘Close’ objects are blocked together with very high probability
‘Far’ objects are blocked together with very low probability

Lecture 2 : 590.03 Fall 13
minHash Analysis

• F is minHash family
  – (0.2, 0.6, 0.8, 0.4)-sensitive

• F1 is a “band of r=5 minHashes”
  – (0.2, 0.6, 0.328, 0.01)-sensitive

• F2 is “k =20 bands of r=5 minHashes each”
  – (0.2, 0.6, 0.98,0.09)-sensitive
Summary of General De-anonymization

Pairwise Matching:
• Construct a comparison vector between every pair of records
• Compute a similarity score between records based on the comparison vector

Record Linkage:
• Create 1-1 mappings (or say no match)
• Bipartite matching (or greedy heuristics)

Blocking:
• Eliminates obvious non-matches for efficient de-anonymization
• minHashing allows designing effective blocking criteria
Back to Netflix Attack

• Pairwise Matching:
  – Comparison Vector: for each movie, 1 if similar ratings at similar times in both records
  – Weighted sum: weights inversely proportional to popularity of movie
  – Threshold: prespecified $\alpha$

• Record Linkage:
  – Best score: Pick the record in D with highest score such that second highest score is much smaller

• Blocking: NONE
Analysis

Theorem 1: Consider a matching threshold $\alpha = 1 - \epsilon$. If the auxiliary record $r$ contains $m$ randomly chosen attributes s.t.

$$m \geq \frac{\log N - \log \epsilon}{-\log(1-\delta)}$$

Then the best matching record $r'$ in D is such that

$$Pr \left[ Sim(r, r') > 1 - \epsilon - \delta \right] > 1 - \epsilon$$
Summary of Netflix Paper

• Adversary can use a subset of ratings made by a user to uniquely identify the user’s record from the “anonymized” dataset with high probability.

• Simple algorithm provably guarantees identification of records in the Netflix dataset.

• Identification is possible even if records in AUX do not exactly match records in D.
Outline

• Recap & Intro to Anonymization

• Algorithmically De-anonymizing Netflix Data

• Algorithmically De-anonymizing Social Networks
  – Passive Attacks
  – Active Attacks
Social Network Data

- Social networks: graphs where each node represents a social entity, and each edge represents certain relationship between two entities
- Example: email communication graphs, social interactions like in Facebook, Yahoo! Messenger, etc.
• Naïve anonymization
  – removes the label of each node and publish only the structure of the network

• Information Leaks
  – Nodes may still be re-identified based on network structure
Passive Attacks on an Anonymized Network

- Consider the above email communication graph
  - Each node represents an individual
  - Each edge between two individuals indicates that they have exchanged emails
Passive Attacks on an Anonymized Network

- Alice has sent emails to three individuals only
Passive Attacks on an Anonymized Network

- Alice has sent emails to three individuals only
- Only one node in the anonymized network has a degree three
- Hence, Alice can re-identify herself
Passive Attacks on an Anonymized Network

- Cathy has sent emails to five individuals
Passive Attacks on an Anonymized Network

- Cathy has sent emails to five individuals
- Only one node has a degree five
- Hence, Cathy can re-identify herself
Passive Attacks on an Anonymized Network

Now consider that Alice and Cathy share their knowledge about the anonymized network.

What can they learn about the other individuals?
Passive Attacks on an Anonymized Network

- First, Alice and Cathy know that only Bob have sent emails to both of them
Passive Attacks on an Anonymized Network

- First, Alice and Cathy know that only Bob have sent emails to both of them
- Bob can be identified
Passive Attacks on an Anonymized Network

- Alice has sent emails to Bob, Cathy, and Ed only
Passive Attacks on an Anonymized Network

- Alice has sent emails to Bob, Cathy, and Ed only
- Ed can be identified
Passive Attacks on an Anonymized Network

- Alice and Cathy can learn that Bob and Ed are connected
Passive Attacks on an Anonymized Network

- The above attack is based on knowledge about degrees of nodes. [Liu and Terzi, SIGMOD 2008]

- More sophisticated attacks can be launched given additional knowledge about the network structure, e.g., a subgraph of the network. [Zhou and Pei, ICDE 2008, Hay et al., VLDB 2008, ]

- Protecting privacy becomes even more challenging when the nodes in the anonymized network are labeled. [Pang et al., SIGCOMM CCR 2006]
Inferring Sensitive Values on a Network

- Each individual has a single sensitive attribute.
  - Some individuals share the sensitive attribute, while others keep it private

- **GOAL:** Infer the private sensitive attributes using
  - Links in the social network
  - Groups that the individuals belong to

- **Approach:** Learn a predictive model (*think classifier*) using public profiles as training data. [Zheleva and Getoor, WWW 2009]
Inferring Sensitive Values on a Network

- **Baseline**: Most commonly appearing sensitive value amongst all public profiles.
Inferring Sensitive Values on a Network

- **LINK**: Each node $x$ has a list of binary features $L_x$, one for every node in the social network.
  - Feature value $L_x[y] = 1$ if and only if $(x,y)$ is an edge.
  - Train a model on all pairs $(L_x, \text{sensitive value}(x))$, for $x$’s with public sensitive values.
  - Use learnt model to predict private sensitive values
Inferring Sensitive Values on a Network

- **GROUP**: Each node $x$ has a list of binary features $G_x$, one for every group in the social network.
  - Feature value $G_x[y] = 1$ if and only if $x$ belongs to group $y$.
  - Train a model on all pairs $(G_x, sensitive value(x))$, where $x$’s sensitive value is public.
  - Use model to predict private sensitive values
Inferring Sensitive Values on a Network

[Zheleva and Getoor, WWW 2009]

<table>
<thead>
<tr>
<th></th>
<th>Flickr (Location)</th>
<th>Facebook (Gender)</th>
<th>Facebook (Political View)</th>
<th>Dogster (Dog Breed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>27.7%</td>
<td>50%</td>
<td>56.5%</td>
<td>28.6%</td>
</tr>
<tr>
<td>LINK</td>
<td>56.5%</td>
<td>68.6%</td>
<td>58.1%</td>
<td>60.2%</td>
</tr>
<tr>
<td>GROUP</td>
<td>83.6%</td>
<td>77.2%</td>
<td>46.6%</td>
<td>82.0%</td>
</tr>
</tbody>
</table>
Active Attacks on Social Networks

[Backstrom et al., WWW 2007]

• Attacker may create a few nodes in the graph
  – Creates a few ‘fake’ Facebook user accounts.

• Attacker may add edges from the new nodes.
  – Create friends using ‘fake’ accounts.

• Goal: Discover an edge between two legitimate users.
High Level View of Attack

• Step 1: Create a graph structure with the ‘fake’ nodes such that it can be identified in the anonymous data.
High Level View of Attack

- Step 2: Add edges from the ‘fake’ nodes to real nodes.
High Level View of Attack

• Step 3: From the anonymized data, identify fake graph due to its special graph structure.
High Level View of Attack

- Step 4: Deduce edges by following links
Details of the Attack

• Choose $k$ real users $W = \{w_1, \ldots, w_k\}$
• Create $k$ fake users $X = \{x_1, \ldots, x_k\}$
• Creates edges $(x_i, w_i)$
• Create edges $(x_i, x_{i+1})$
• Create all other edges in $X$ with probability 0.5.
Why does it work?

- Given a graph $G$, and a set of nodes $S$, $G[S] = \text{graph induced by nodes in } S$.

- There is an *isomorphism* between two sets of nodes $S$, $S'$ if
  - There is a function mapping each node in $S$ to a node in $S'$
  - $(u,v)$ is an edge in $G[S]$ if and only if $(f(u), f(v))$ is an edge in $S'$

- Isomorphism from $S$ to $S$ is called an automorphism
  - Think: permuting the nodes
Why does it work?

• There is no S such that $G[S]$ is isomorphic to $G[X]$ (call it $H$).
• $H$ can be efficiently found from $G$.
• $H$ has no non-trivial automorphisms.
Recovery

Subgraph isomorphism is NP-hard
  – i.e., Finding X could be hard.

But since X has a path, with random edges, there is a simple brute force with pruning search algorithm.

Run Time: $O(N 2^{O(\log \log N)})$
Works in Real Life!

- LiveJournal – 4.4 million nodes, 77 million edges
- Success all but guaranteed by adding 10 nodes.
- Recovery typically takes a second.

[Backstrom et al., WWW 2007]
Summary of Social Networks

• Nodes in a graph can be re-identified using background knowledge of the structure of the graph.

• Link and group structure provide valuable information for accurately inferring private sensitive values.

• Active attacks that add nodes and edges are shown to be very successful.

• Guarding against these attacks is an open area for research!
Next Class

- K-Anonymity + Algorithms: How to limit de-anonymization?
References

A. Narayanan & V. Shmatikov, “Robust De-anonymization of Large Sparse Datasets”, SSP 2008
L. Backstrom, C. Dwork & J. Kleinberg, “Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography”, WWW 2007
E. Zheleva & L. Getoor, “To join or not to join: the illusion of privacy in social networks with mixed public and private user profiles”, WWW 2009