Differentially Private Recommendation Systems

Giulia Fanti
Slides by Anupam Datta
Fall 2019
Administrative

- Proposals graded on Gradescope
  - Please look at comments
  - Even on questions where you got full points

- Piazza
  - StackOverflow guidelines
  - NEW: 5 students with the most answers endorsed by an instructor get a bonus point on final grade

- Recitation this week (Sruti): Friday @ 12.30 ET/9.30 am PT
  - Differential privacy practice

- James Office Hours
  - Wed (today) at 4pm ET/1 pm ET

- Giulia Office Hours
  - Friday at 5.30 pm ET/2.30 pm PT
Canvas Quiz

- 10 minutes
Randomized sanitization function $\kappa$ has $\varepsilon$-differential privacy if for all data sets $D_1$ and $D_2$ differing by at most one element and all subsets $S$ of the range of $\kappa$,

$$\Pr[\kappa(D_1) \in S] \leq e^\varepsilon \Pr[\kappa(D_2) \in S]$$
Laplace Mechanism

- Global Sensitivity: \[ \text{GS}_f = \max_{\text{neighbors } x, x'} \| f(x) - f(x') \|_1 \]

- Example: \[ \text{GS}_{\text{proportion}} = \frac{1}{n} \]

**Theorem:** If \( A(x) = f(x) + \text{Lap} \left( \frac{\text{GS}_f}{\epsilon} \right) \), then \( A \) is \( \epsilon \)-differentially private.
Example: Noise Addition

- **Example:** proportion of diabetics
  - $G_{\text{proportion}} = \frac{1}{n}$
  - Release $A(x) = \text{proportion} \pm \frac{1}{\epsilon n}$

- **Is this a lot?**
  - If $x$ is a random sample from a large underlying population,
    then **sampling noise** $\approx \frac{1}{\sqrt{n}}$
  - $A(x)$ “as good as” real proportion
Using Global Sensitivity

- Many natural functions have low global sensitivity
  - Histogram, covariance matrix, Lipschitz optimization problems

- Different mechanisms have different privacy-utility tradeoffs
  - Laplace noise can add more noise than necessary
Composition Theorem

Repeated querying degrades privacy; degradation is quantifiable

- **Theorem.** If $A_1$ is $\varepsilon_1$-differentially private and $A_2$ is $\varepsilon_2$-differentially private and they use independent random coins then the composition of $A_1$ and $A_2$ is $(\varepsilon_1 + \varepsilon_2)$-differentially private

- Work with your neighbor to prove this.
Applications

- Netflix data set [McSherry, Mironov 2009; MSR]
  - Accuracy of differentially private recommendations (wrt one movie rating) comparable to baseline set by Netflix
- Network trace data sets [McSherry, Mahajan 2010; MSR]

<table>
<thead>
<tr>
<th>Packet-level analyses</th>
<th>High accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet size and port dist.</td>
<td>(§5.1.1)</td>
</tr>
<tr>
<td>Worm fingerprinting [27]</td>
<td>strong privacy</td>
</tr>
<tr>
<td>Flow-level analyses</td>
<td></td>
</tr>
<tr>
<td>Common flow properties [30]</td>
<td>(§5.2.1)</td>
</tr>
<tr>
<td>Stepping stone detection [33]</td>
<td>strong privacy</td>
</tr>
<tr>
<td>Graph-level analyses</td>
<td></td>
</tr>
<tr>
<td>Anomaly detection [13]</td>
<td>(§5.3.1)</td>
</tr>
<tr>
<td>Passive topology mapping [9]</td>
<td>strong privacy</td>
</tr>
</tbody>
</table>
Challenge: High Sensitivity

- Approach: Add noise proportional to sensitivity to preserve $\varepsilon$-differential privacy

- Improvements:
  - Smooth sensitivity [Nissim, Raskhodnikova, Smith 2007; BGU-PSU]
  - Restricted sensitivity [Blocki, Blum, Datta, Sheffet 2013; CMU]
Differential Privacy: Summary

- An approach to releasing privacy-preserving statistics
- A rigorous privacy guarantee
  - Significant activity in theoretical CS community
- Several applications to real data sets
  - Recommendation systems, network trace data,
- Some challenges
  - High sensitivity -> high noise
  - Repeated querying
So far, you have seen:

- Definition of differential privacy
- How to make a scalar query differentially private
  - Laplacian noise

- What about training machine learning models on sensitive data?
  - How do we use differential privacy?
Differentially Private Recommender Systems: Building Privacy into the Netflix Prize Contenders

Frank McSherry and Ilya Mironov

KDD 2009
I want a predictor trained on your data.

How can we make this differentially private?
### Netflix $1,000,000 Prize Competition

<table>
<thead>
<tr>
<th>User/Movie</th>
<th>....</th>
<th>300</th>
<th>The Notebook</th>
<th>....</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>John</td>
<td>4</td>
<td>Unrated</td>
<td>Unrated</td>
<td></td>
</tr>
<tr>
<td>Mary</td>
<td>Unrated</td>
<td>Unrated</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sue</td>
<td>2</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Joe</td>
<td>5</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Queries: On a scale of 1 to 5 how would John rate “The Notebook” if he watched it?
## Netflix Prize Competition

**Note:** N x M table is very sparse (M = 17,770 movies, N = 500,000 users)

**To Protect Privacy:**
- Each user was randomly assigned to a globally unique ID
- Only 1/10 of the ratings were published
- The ratings that were published were perturbed a little bit

<table>
<thead>
<tr>
<th>User/Movie</th>
<th>13,537</th>
<th>13,538</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>258,964</td>
<td>(4, 10/11/2005)</td>
<td>Unrated</td>
<td></td>
</tr>
<tr>
<td>258,965</td>
<td>Unrated</td>
<td>Unrated</td>
<td></td>
</tr>
<tr>
<td>258,966</td>
<td>(2, 6/16/2005)</td>
<td>(5, 6/18/2005)</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>
Root Mean Square Error

\[ RMSE(P) = \sqrt{\frac{1}{k} \sum_{i=1}^{k} (p_i - a_i)^2} \]

- \( p_i \in [1,5] \) - predicted ratings
- \( a_i \in [1,5] \) - actual ratings
Netflix Prize Competition

Goal: Make accurate predictions as measured by Root Mean Squared Error (RMSE)

\[ RMSE(P) = \sqrt{\frac{1}{k} \sum_{i=1}^{k} (p_i - a_i)^2} \]

- \( p_i \in [1,5] \) - predicted ratings
- \( a_i \in [1,5] \) - actual ratings

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>BellKor's Pragmatic Chaos</td>
<td>0.8567 (&lt; 0.8572)</td>
</tr>
<tr>
<td>Challenge: 10% Improvement</td>
<td>0.8572</td>
</tr>
<tr>
<td>Netflix’s Cinematch (Baseline)</td>
<td>0.9525</td>
</tr>
</tbody>
</table>
## Leaderboard

Showing Test Score. [Click here to show quiz score](#)

Display top 20 leaders.

<table>
<thead>
<tr>
<th>Rank</th>
<th>Team Name</th>
<th>Best Test Score</th>
<th>% Improvement</th>
<th>Best Submit Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BellKor's Pragmatic Chaos</td>
<td>0.8567</td>
<td>10.06</td>
<td>2009-07-26 18:18:28</td>
</tr>
<tr>
<td>2</td>
<td>The Ensemble</td>
<td>0.8567</td>
<td>10.06</td>
<td>2009-07-26 18:38:22</td>
</tr>
<tr>
<td>3</td>
<td>Grand Prize Team</td>
<td>0.8582</td>
<td>9.90</td>
<td>2009-07-10 21:24:40</td>
</tr>
<tr>
<td>4</td>
<td>Opera Solutions and Vandelay United</td>
<td>0.8588</td>
<td>9.84</td>
<td>2009-07-10 01:12:31</td>
</tr>
<tr>
<td>5</td>
<td>Vandelay Industries I</td>
<td>0.8591</td>
<td>9.81</td>
<td>2009-07-10 00:32:20</td>
</tr>
<tr>
<td>6</td>
<td>PragmaticTheory</td>
<td>0.8594</td>
<td>9.77</td>
<td>2009-06-24 12:06:56</td>
</tr>
<tr>
<td>7</td>
<td>BellKor in BigChaos</td>
<td>0.8601</td>
<td>9.70</td>
<td>2009-05-13 08:14:09</td>
</tr>
<tr>
<td>8</td>
<td>Dace</td>
<td>0.8612</td>
<td>9.59</td>
<td>2009-07-24 17:18:43</td>
</tr>
<tr>
<td>9</td>
<td>Feeds2</td>
<td>0.8622</td>
<td>9.48</td>
<td>2009-07-12 13:11:51</td>
</tr>
<tr>
<td>10</td>
<td>BigChaos</td>
<td>0.8623</td>
<td>9.47</td>
<td>2009-04-07 12:33:59</td>
</tr>
<tr>
<td>11</td>
<td>Opera Solutions</td>
<td>0.8623</td>
<td>9.47</td>
<td>2009-07-24 00:34:07</td>
</tr>
<tr>
<td>12</td>
<td>BellKor</td>
<td>0.8624</td>
<td>9.46</td>
<td>2009-07-26 17:19:11</td>
</tr>
</tbody>
</table>

### Progress Prize 2008 - RMSE = 0.8627 - Winning Team: BellKor in BigChaos

<table>
<thead>
<tr>
<th>Rank</th>
<th>Team Name</th>
<th>Best Test Score</th>
<th>% Improvement</th>
<th>Best Submit Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>13</td>
<td>xiangliang</td>
<td>0.8642</td>
<td>9.27</td>
<td>2009-07-15 14:53:22</td>
</tr>
<tr>
<td>14</td>
<td>Gravity</td>
<td>0.8643</td>
<td>9.26</td>
<td>2009-04-22 18:31:32</td>
</tr>
<tr>
<td>15</td>
<td>Ces</td>
<td>0.8651</td>
<td>9.18</td>
<td>2009-06-21 19:24:53</td>
</tr>
<tr>
<td>16</td>
<td>Invisible Ideas</td>
<td>0.8653</td>
<td>9.15</td>
<td>2009-07-15 15:53:04</td>
</tr>
<tr>
<td>17</td>
<td>Just a guy in a garage</td>
<td>0.8662</td>
<td>9.06</td>
<td>2009-05-24 10:02:54</td>
</tr>
<tr>
<td>18</td>
<td>J Dennis Su</td>
<td>0.8666</td>
<td>9.02</td>
<td>2009-03-07 17:16:17</td>
</tr>
<tr>
<td>19</td>
<td>Craig Carmichael</td>
<td>0.8666</td>
<td>9.02</td>
<td>2009-07-25 16:00:54</td>
</tr>
<tr>
<td>20</td>
<td>acmehill</td>
<td>0.8668</td>
<td>9.00</td>
<td>2009-03-21 16:20:50</td>
</tr>
</tbody>
</table>

### Progress Prize 2007 - RMSE = 0.8723 - Winning Team: KorBell

### Cinematch score - RMSE = 0.9525
On Friday, Netflix announced on its corporate blog that it has settled a lawsuit related to its Netflix Prize, a $1 million contest that challenged machine learning experts to use Netflix’s data to produce better recommendations than the movie giant could serve up themselves.

The lawsuit called attention to academic research that suggests that Netflix indirectly exposed the movie preferences of its users by publishing anonymized customer data. In the suit, plaintiff Paul Navarro and others sought an injunction preventing Netflix from going through the so-called “Netflix Prize II,” a follow-up challenge that Netflix promised would offer up even more personal data such as genders and zipcodes.
Outline

- Approximate Differential Privacy
- Prediction Algorithms
- Privacy Preserving Prediction Algorithms
- Remaining Issues
Privacy in Recommender Systems

- Netflix might base its recommendation to me on both:
  - My own rating history
  - The rating history of other users

- Goal: not leak other users’ ratings to me

- Basic recommendation systems leak other users’ information
Recall Differential Privacy [Dwork et al 2006]

Dual-sided restatement: for all data sets $D_1$ and $D_2$ differing by at most one element and all outputs $s$ in the range of $\kappa$,

$$e^{-\epsilon} \leq \frac{\Pr[\kappa(D_1) = s]}{\Pr[\kappa(D_2) = s]} \leq e^{\epsilon}$$

and more generally, for all subsets $S$ of the range of $\kappa$

$$e^{-\epsilon} \leq \frac{\Pr[\kappa(D_1) \in S]}{\Pr[\kappa(D_2) \in S]} \leq e^{\epsilon}$$
Review: Laplacian Mechanism

$K(D) = \min_{\epsilon \geq 0} \mathbb{E}(GS_\epsilon(D))$

Thm: $K(D)$ is **\(\varepsilon\)-differentially private**

Picture Proof:

**Question:** The Gaussian (Normal) distribution is nicer because it is more tightly concentrated around its mean. Can we use that distribution instead?
Gaussian Mechanism

\[ \kappa(D) = f(D) + N\left(\frac{GS_f}{\varepsilon}\right) \]

Thm? \( K \) is \( \varepsilon \)-differentially private?

**Probability Density Function**

\[ N(x, 0, \sigma) \propto \frac{1}{\sigma \sqrt{2\pi}} \exp\left(-\frac{x^2}{2\sigma^2}\right) \]

---

**Problem:** The ratio can be huge at the tails!

\[ e^{-\varepsilon} \leq \text{Ratio} = \frac{B_{Black}}{B_{Red}} \leq e^{\varepsilon} \]

But these events are very unlikely…
Approximate Differential Privacy

Randomized sanitization function $\kappa$ has $(\epsilon, \delta)$-differential privacy if for all data sets $D_1$ and $D_2$ differing by at most one element and all subsets $S$ of the range of $\kappa$,

$$\Pr[\kappa(D_1) \in S] \leq e^\epsilon \Pr[\kappa(D_2) \in S] + \delta$$
Gaussian Mechanism

\[ K(D) = f(D) + N(\sigma^2) \]

**Thm** K is \((\varepsilon, \delta)\)-differentially private as long as \(\sigma \geq \frac{\sqrt{2 \ln(2/\delta)}}{\varepsilon} \times GS_f \)

**Idea** Use \(\delta\) to exclude the tails of the gaussian distribution
Multivariate Gaussian Mechanism

Suppose that f outputs a length d vector instead of a number

\[ K(D) = f(D) + N(\sigma^2)^d \]

**Thm** K is \((\varepsilon, \delta)\)-differentially private as long as

\[ \sigma \geq \frac{\sqrt{2 \ln(2/\delta)}}{\varepsilon} \times \max_{D_1 \approx D_2} \| f(D_1) - f(D_2) \|_2 \]

Remark: Similar results would hold with the Laplacian Mechanism, but we would need to add noise proportional to the larger \(L_1\) norm
Approximate Differential Privacy

- **Key Difference**
  - Approximate Differential Privacy does NOT require that:

\[ \text{Support}(\kappa(D_1)) = \text{Support}(\kappa(D_2)) \]

- The privacy guarantees made by \((\varepsilon,\delta)\)-differential privacy are not as strong as \(\varepsilon\)-differential privacy, but less noise is required to achieve \((\varepsilon,\delta)\)-differential privacy.
Approximate Differential Privacy

- Key similarity
  - Composition still holds!

- If \( M_1 \) and \( M_2 \) satisfy \((\varepsilon_1, \delta_1)\) and \((\varepsilon_2, \delta_2)\) differential privacy, respectively, then their linear composition satisfies \((\varepsilon_1 + \varepsilon_2, \delta_1 + \delta_2)\) differential privacy.
Differential Privacy for Netflix Queries

- What level of granularity to consider? What does it mean for databases $D_1$ and $D_2$ to differ on at most one element?
  - One user (column) is present in $D_1$ but not in $D_2$
  - One rating (cell) is present in $D_1$ but not in $D_2$

- Issue 1: Given a query “how would user $i$ rate movie $j$?”
  Consider: $\kappa(D - u[i])$ - how can it possibly be accurate?

- Issue 2: If the definition of differing in at most one element is taken over cells, then what privacy guarantees are made for a user with many data points?
Netflix Predictions – High Level

- **Q(i,j)** – “How would user i rate movie j?”

- Predicted rating may typically depend on
  - Global average rating over all movies and all users
  - Average movie rating of user i
  - Average rating of movie j
  - Ratings user i gave to similar movies
  - Ratings similar users gave to movie j

- Sensitivity may be small for many of these queries
Personal Rating Scale

- For Alice a rating of 3 might mean the movie was really terrible.
- For Bob the same rating might mean that the movie was excellent.
- How do we tell the difference?

Check the following:

\[ r_{im} - \bar{r}_i > 0? \]
Pearson’s Correlation is one metric for similarity of users i and j
• Consider all movies rated by both users
• Negative value whenever i likes a movie that j dislikes
• Positive value whenever i and j agree

\[ S(i, j) = \sum_{m \in L_i \cap L_j} (r_{im} - \bar{r}_i)(r_{jm} - \bar{r}_j) \]

We can use similar metrics to measure the similarity between two movies.
Netflix Predictions Example

- **Collaborative Filtering**
  - Find the k-nearest neighbors of user i who have rated movie j by Pearson’s Correlation:

    \[ S(i, \ell) \]  
    \[ N_i(k, j) = \{u_1, u_2, \ldots, u_k\} \]  
    \[ k \text{ most similar users who rated movie } j \]

- **Predicted Rating**

    \[ p_{ij} = \bar{r}_i + \frac{1}{k} \sum_{u \in N_i(k, j)} (r_{uj} - \bar{r}_u) \]
Netflix Prediction Sensitivity Example

\[ p_{ij} = \bar{r}_i + \frac{1}{k} \sum_{u \in N_i(k,j)} (r_{uj} - \bar{r}_u) \]

- Pretend the query \( Q(i,j) \) included user i’s rating history
- At most one of the neighbors ratings changes, and the range of ratings is 4 (since ratings are between 1 & 5). The L_1 sensitivity of the prediction is:

\[ \Delta p = \frac{4}{k} \]
Similarity of Two Movies

- Let $U$ be the set of all users who have rated both movies $i$ and $j$ then

$$S(i, j) = \sum_{u \in U} (r_{uj} - \bar{r}_u)(r_{ui} - \bar{r}_u)$$
K-Nearest Users or K-Nearest Movies?

- Find $k$ most similar users to $i$ that have also rated movie $j$?
- Find $k$ most similar movies to $j$ that user $i$ has rated?

Either way, after some pre-computation, we need to be able to find the $k$-nearest users/movies quickly!
Covariance Matrix

**Movie-Movie Covariance Matrix**
- $(M \times M)$ matrix
- $\text{Cov}[i][j]$ measures similarity between movies $i$ and $j$
- $M \approx 17,000$
- More accurate

**User-User Covariance Matrix?**
- $(N \times N)$ Matrix to measure similarity between users
- $N \approx 500,000$
- Less accurate
What do we need to make predictions?

For a large class of prediction algorithms it suffices to have:

- $G_{avg}$ – average rating for all movies by all users
- $M_{avg}$ – average rating for each movie by all users
- Average Movie Rating for each user
- Movie-Movie Covariance Matrix (COV)
Differentially Private Recommender Systems (High Level)

To respect approximate differential privacy publish

- $G_{avg} + \text{NOISE}$
- $M_{avg} + \text{NOISE}$
- $COV + \text{NOISE}$

- $GS(G_{avg}), GS(M_{avg})$ are very small so they can be published with little noise (e.g., Laplacian)
- $GS(COV)$ requires more care (our focus)

- Don’t publish average ratings for users (used in per-user prediction phase using k-NN or other algorithms)

Source: Differentially Private Recommender Systems (McSherry and Mironov)