Learning a logistic regression from Aggregated Data

Alexandre Gilotte
David Rohde
Click prediction models and Privacy sandbox

<table>
<thead>
<tr>
<th>Partner</th>
<th>Publisher</th>
<th>Ad Size</th>
<th>More features...</th>
<th>Click ?</th>
</tr>
</thead>
<tbody>
<tr>
<td>42</td>
<td>A</td>
<td>Small</td>
<td>...</td>
<td>0</td>
</tr>
<tr>
<td>43</td>
<td>A</td>
<td>Large</td>
<td>...</td>
<td>0</td>
</tr>
<tr>
<td>42</td>
<td>B</td>
<td>Large</td>
<td>...</td>
<td>1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>0</td>
</tr>
</tbody>
</table>

Learning model of $P(Y=1|X=x)$

Loss := $\Sigma_i Loss(f(x_i, y_i))$

- Eg logistic regression
- Gradient descent, ...

Privacy sandbox:
- Dataset no longer available!
- Instead, "Aggregated data"
Aggregated data?

Tables counting displays and clicks
- On subsets of variables
- Tables may be overlapping
- Also, noise may be added to get differential privacy guarantees.

Learning $P(Y | X)$ from these tables??

<table>
<thead>
<tr>
<th>Partner</th>
<th>Publisher</th>
<th>Nb Displays</th>
<th>Nb Clicks</th>
</tr>
</thead>
<tbody>
<tr>
<td>42</td>
<td>A</td>
<td>10000</td>
<td>600</td>
</tr>
<tr>
<td>42</td>
<td>B</td>
<td>55000</td>
<td>1000</td>
</tr>
<tr>
<td>43</td>
<td>A</td>
<td>20000</td>
<td>500</td>
</tr>
<tr>
<td>43</td>
<td>B</td>
<td>8000</td>
<td>300</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Partner</th>
<th>Ad size</th>
<th>Nb Displays</th>
<th>Nb Clicks</th>
</tr>
</thead>
<tbody>
<tr>
<td>42</td>
<td>Small</td>
<td>100000</td>
<td>5500</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Publisher</th>
<th>Ad size</th>
<th>Nb Displays</th>
<th>Nb Clicks</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Small</td>
<td>30000</td>
<td>700</td>
</tr>
</tbody>
</table>

... ... ... ... ...

... ... ... ... ...

... ... ... ... ...

... ... ... ... ...
Formalizing the aggregated data

• Unobserved dataset \((x_i, y_i)\) iid

• Quadratic kernel \(K\) mapping \(x\) to \(\{0;1\}^D\)

Observed aggregated data:

\[
d := \sum_i K(x_i)
\]

\[
c := \sum_i y_i \cdot K(x_i)
\]
Proposed approach

**Modeling**
Choose a parametric model on the **joined** distribution of features $X$ and labels $Y$

$$\mathbb{P}_\theta(X = x, Y = y)$$

**Training**
Select theta maximizing the likelihood of observed event:

$$\text{Argmax}_\theta \mathbb{P}_\theta(D = d, C = c)$$

**Predicting**
With the conditional law:

$$\mathbb{P}_\theta(Y = 1 | X = x) = \frac{\mathbb{P}_\theta(Y = 1, X = x)}{\mathbb{P}_\theta(Y = 1, X = x) + \mathbb{P}_\theta(Y = 0, X = x)}$$

Aggregated data are a realization of the random variables:

$$D := \sum_i K(X_i)$$
$$C := \sum_i Y_i \cdot K(X_i)$$

Intractable for most models ?!
Loglinear model

Modeling
Parametric model loglinear in $\mathcal{K}(X)$:

$$P_{\mu, \theta}(X = x, Y = y) := \frac{\exp (\mathcal{K}(x) \cdot \mu + y \cdot \mathcal{K}(x) \cdot \theta)}{Z_{\mu, \theta}}$$

- « Random Markov Field »

Predicting

$$P_{\mu, \theta}(Y = y | X = x) = \sigma(\mathcal{K}(x) \cdot \theta)$$

- No $Z$, and no $\mu$
- Looks like a logistic regression with kernel $\mathcal{K}$

Normalization constant. Intractable!?
Training

Gradient of the log-likelihood

\[ \nabla_\mu \log \mathbb{P}_{\mu,\theta}(D = d, C = c) = d - \mathbb{E}_{\mu,\theta}(D) \]

\[ \nabla_\theta \log \mathbb{P}_{\mu,\theta}(D = d, C = c) = c - \mathbb{E}_{\mu,\theta}(C) \]

- Exponential family
- Aggregated data are sufficient statistics!

Aggregated data

- Depends only on the model (no data)
- Estimated by Monte Carlo on Gibbs samples.
Experimental Results

On a medium size public Criteo dataset

- Public Criteo dataset
- Quite « small »
  - 11 features
  - 16M examples, 33% clicks

- Not far from skyline!

On Criteo Adkdd challenge

- Larger dataset
- 18 features
- ~100M aggregated samples

<table>
<thead>
<tr>
<th>Model</th>
<th>NLLH</th>
<th>Training time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logistic, 2 order kernel, full dataset</td>
<td>0.091</td>
<td>2h</td>
</tr>
<tr>
<td>Markov Random Field (ours)</td>
<td>0.089</td>
<td>120h</td>
</tr>
<tr>
<td>Logistic, no kernel, full dataset</td>
<td>0.076</td>
<td>0.2h</td>
</tr>
<tr>
<td>Logistic, 2 order kernel, small trainset</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Still quite far from logistic with full data
- But using *only* aggregated data
Limitations and next steps

Optimization is difficult!

- Gibbs sampling: inefficient with strongly correlated features

Modelling error on P(X)

- Compared to logistic regression, we have to model P(X) instead of using train samples
- With many correlated features, our model on P(X) may be very wrong, leading to worse P(Y|X)
- Higher order aggregation tables?

Validating with only aggregated data?

- Choice of parameters and monitoring model quality by cross validation … on granular (ie non aggregated) data.
- How to avoid this?
Also in the paper

**L2 Regularization**
- Strong regularization on $\theta$
- Low regularization on $\mu$

**Monte Carlo on Gibbs samples**
- Marginalize on $Y$ to lower the noise on the gradients of $\theta$

**Re-using Gibbs samples between gradient steps**
- « Persistent contrastive divergence »

**Modelling also the noise**
- If aggregated data are noisy
- $\text{Argmax}( P( D+\text{Noise} = d, C+\text{Noise} = c) )$

Thank you!