Information Privacy Tradeoff

Cole Becker and Amir Aliddd

September 2022

1 Related Works

1.1 Differential Privacy

1.2 Knapsack Problem

1.3 Complexity

2 The Problem

We consider the following decision problem, we denote Information Privacy Tradeoff Problem (IPT)

Input: Consider a dataset $\mathcal{D} = \{d_i\}_{i \leq m} \subseteq \mathbb{R}^n$ where $d_i \in \{0,1\}^n$, and denote $D \in \mathbb{R}^{m \times n}$ as the matrix containing this data. We can imagine for example that m is the number of people in the dataset, and $d_i \in \mathbb{R}^n$ represents a binary feature vector of information on n features for the ith person. In addition, you are given a non-negative weight vector $w \in \mathbb{R}^n$ representing the relative importance of each the n features, some number $r \in [0,1]$ and an index $k \in \{1, ..., m\}$. Finally we also take in a non-negative number $l \in \mathbb{R}$.

Question: Is there a way to select a subset of features of d_k, indexed by $x \in \{0,1\}^n$ such that $w^T x \geq l$ and the subset features of d_k are shared by at least rm people in the dataset (d_k included). **Note**: Throughout the entirety of this paper, we will assume $k = 0$ unless otherwise specified, and we will refer to d_k as Person of Interest (poi).

2.1 Motivation

Suppose you are a company holding such a dataset $\mathcal{D}$ about various attributes of a large quantity of individuals. Some advertising company Ad.co comes to you, and wants to purchase the data of some individual k to run some targeted advertising of their own. In order to convey which features they are most interested in knowing, or which features are most useful to their advertising campaign, they supply you with a weight vector w which represents how much Ad.co values knowing each feature of individual k in your dataset. However, as your company must comply with privacy regulations, you are not allowed to share too

much information about individual k which would distinguish them too much. Specifically, the features you reveal to Ad.co about individual k must also be shared by $100 * r\%$ of the members of your dataset. Your task is to maximize the useful information you can share with Ad.co while satisfying the imposed privacy constraints.

2.2 Formulation of IPT

Taking all the same inputs from the above problem, let us first define a helper matrix $K \in \mathbb{R}^{m \times n}$ as

$$K_{ij} = \begin{cases} 0 & \text{if } D_{ij} = D_{kj} = (d_k)_j \\ 1 & \text{otherwise} \end{cases}$$

We note that IPT can be reduced to the following non-convex optimization problem

$$\begin{aligned} \text{IPT} = \quad & \underset{x}{\text{maximize}} \quad w^T x \\ & \text{subject to} \quad ||Kx||_0 \leq m(1-r) \\ & \qquad\qquad\ \ x_i \in \{0,1\} \end{aligned} \tag{IPT}$$

where the answer to the decision question is given by whether IPT $\geq l$. To see why the constraint in this problem is equivalent to the requirement that the subset x of d_k's features is shared by at least rm people, note that

$$(Kx)_i = \begin{cases} 0 & \text{if } d_i \text{ shares the same } x \text{ features with } d_k \\ > 0 & \text{otherwise} \end{cases}$$

so the number of 0-elements in Kx must be greater than mr or equivalently, the number of non-zero elements must be less than $m(1-r)$

2.3 Related Problem

A closely related problem is the following problem is Generalized Information Privacy Tradeoff Problem (GIPT)

$$\text{GIPT} = \text{argmin}_k \ \text{IPT}_k$$

where IPT_k represents the information privacy tradeoff of the kth individual in the dataset. We can think of the minimum over all such individauls as the most distinguishable/unique, or least anonymous individual. This is the individual for whom you can reveal the least information before they begin to be distinguishable from $r\%$ of the dataset. In a closely related problem, we have

$$\text{argmax}_k \ \text{IPT}_k$$

which can be thought of as the most anonymous and least unique individual in the dataset.

20 Questions A simple way to think of this problem is as a game. Given a dataset of possible things to guess in a game of twenty questions (or people to learn information from), you would like to choose the least distinguishable individual to maximize the amount of guesses

NOTE: argmax_k IPT_k isn't quite identical to 20 questions. What is the goal of twenty questions? The goal as a guesser is to reveal the features that on average leave the fewest members remaining. Here we are interested in privacy, so it would be more of a word producing game. We would want to produce a word such that on average, it is very anonymous. In other words, no matter what features you reveal, it doesn't narrow down your search very much. argmax_k IPT_k, means that you can reveal lots of features without narrowing the search down much. However think of the worst case scenario, in which there is one feature that such an individual has that she shares with nobody. This would make them distinguishable in a way, even if they were very anonymous with respect to their other features. So perhaps it isn't entirely obvious that argmax_k IPT_k is the best metric for a 20 questions guessing game.

3 Complexity of IPT

Theorem 3.1. *IPT is NP-complete*

Proof. It is sufficient to show that IPT $\in$ NP and that Knapsack Problem (KNAP) $\longrightarrow$ IPT

1. IPT $\in$ NP: Given a certificate solution x^* to IPT, it is easy to check that the constraints are satisfied, and that $w^T x^* \geq l$

2. KNAP $\longrightarrow$ IPT: Consider the classic KNAP:
   Input: $w \in \mathbb{R}^n$ *a weight vector of n items,* $p \in \mathbb{R}^n$ *a price vector for the items,* $W \in \mathbb{R}$ *a weight capacity, and* $P \in \mathbb{R}$
   Question: *Is there a set of items of combined weight less than* W *but with combined price greater than* P*?*

   Using these inputs we will construct an instance of IPT (Note we will assume for convenience that $W \in [0, 1^T w]$. If not we can define a new W that is respective endpoint). To begin, set $m = 1 + 1^T w$ and construct the following data matrix $D \in \mathbb{R}^{m \times n}$ by the following procedure:

   (a) Set the first row of D to a row of all 0s

   (b) For each weight w_i, $i \in \{1, ..., n\}$ add w_i copies of the one hot encoded row vector

e_i. As an example:

$$w = [3, 1, 2, 1] \implies D = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

This is the data matrix D we will consider. Note that the auxiliary matrix K used in IPT is identical to D via the way we constructed it.

In addition, set $r = 1 - W/m$ (whereby assumption on W we have $r \in [0, 1]$), $l = P$, $k = 1$ the first row, and $\bar{w} = p$, the information weighting vector. We now have an instance of IPT. In terms of the KNAP variables, the equivalent IPT problem looks like the following:

$$\text{IPT(KNAP)} = \begin{array}{ll} \underset{x}{\text{maximize}} & p^T x \\ \text{subject to} & ||Dx||_0 \leq W \\ & x_i \in \{0, 1\} \end{array} \tag{1}$$

Notice the problem construction requires a polynomial amount of operations, as the size of $D = n * (1 + 1^T w)$, and everything else requires an affine amount of operations. We would now like to show the following equivalence between the problems.

$$\text{KNAP} \geq P \iff \text{IPT(KNAP)} \geq P$$

($\underline{\implies}$): Consider a solution $\hat{x} \in \{0, 1\}^n$ to KNAP satisfying the weight constraints and with $p^T x \geq P$. This same solution $\hat{x}$ will be feasible to IPT(KNAP). To see why, just note that $||Dx||_0 = w^T x$ is equivalent to taking the sum of the weights, because the construction of D and x mean that $||Dx||_0 = 1^T Dx$ and $1^T D = w$ by construction.
($\underline{\impliedby}$): The backwards argument is similar, but we need to build back the weight vector w from the matrix D. To do so we again note that $w = 1^T D$, and then the problem is identical to knapsack.

■

4 Bounds and Relaxations for IPT

4.1 Mixed Integer Program (MIP) Bounds

As written in its L-0 form, IPT is not even a mixed-integer program. Here we investigate affine relaxations of the L-0 norm and possible interpretations.

4.1.1 MIP Upper Bound

We write an IP relaxation of IPT as the following problem

$$\begin{aligned} \text{IPT}^{\text{IP}} = \quad & \underset{x\in\mathbb{R}^n}{\text{maximize}} && w^T x \\ & \text{subject to} && \mathbf{1}^T K x \leq m(1-r)\mathbf{1}^T x \\ & && x_i \in \{0,1\} \end{aligned} \tag{IPT$^{\text{IP}}$}$$

Where we have $\text{IPT} \leq \text{IPT}^{\text{IP}}$. To see why we would like to show

$$x \text{ feasible to IPT} \implies x \text{ feasible to IPT}^{\text{IP}}.$$

Consider x^* feasible to IPT. If $x^* = 0$ then x^* is clearly feasible to IPT^{IP} as both constraints are trivially satisfied. Now suppose $x^* \neq 0$, we want to show that

$$\frac{\mathbf{1}^T K x^*}{\mathbf{1}^T x^*} \leq m(1-r),$$

or equivalently, that $\frac{\mathbf{1}^T K x^*}{\mathbf{1}^T x^*} \leq ||Kx^*||_0$. The key is to notice is that $(Kx^*)_i \leq \mathbf{1}^T x^*$ for all $i \leq m$ due to the binary nature of the entries of K and x^*, which means that

$$\left(\frac{\mathbf{1}^T K x^*}{\mathbf{1}^T x^*}\right)_i = \begin{cases} 0 & \text{if } (Kx^*)_i = 0 \\ \leq 1 & \text{otherwise} \end{cases}$$

from the above statement we have

$$\frac{\mathbf{1}^T K x^*}{\mathbf{1}^T x^*} \leq ||Kx^*||_0$$

so x^* is feasible to IPT^{IP}, and $\text{IPT} \leq \text{IPT}^{\text{IP}}$.

4.1.2 MIP lower bound

Next consider the following linear integer program

$$\begin{aligned} \text{IPT}_{\text{IP}} \quad = & \underset{x}{\text{maximize}} && w^T x \\ & \text{subject to} && \mathbf{1}^T K x \leq m(1-r) \\ & && x_i \in \{0,1\} \end{aligned} \tag{IPT$_{\text{IP}}$}$$

To see why $\text{IPT}_{\text{IP}} \leq \text{IPT}$, we would like to show that

$$x \text{ feasible to IPT}_{\text{IP}} \implies x \text{ feasible to IPT}.$$

Fix x^* feasible to IPT_{IP}. Due to binary nature of K and x^*, we know Kx^* is a non-negative integer vector. This means $\mathbf{1}^T K x^* = ||Kx^*||_1$ and so we have

$$||Kx^*||_0 \leq ||Kx^*||_1 = \mathbf{1}^T K x^* \leq m(1-r)$$

so x^* is feasible to IPT, and $\text{IPT}_{\text{IP}} \leq \text{IPT}$

4.1.3 Relation to KNAP

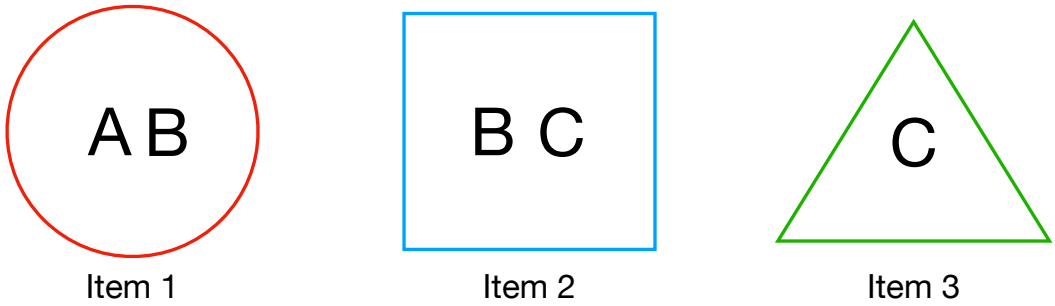
Succinctly, we have shown that $\text{IPT}_{\text{IP}} \leq \text{IPT} \leq \text{IPT}^{\text{IP}}$ due to the fact that for non-zero x, we have

$$\frac{\mathbf{1}^T K x}{\mathbf{1}^T x} \leq ||Kx||_0 \leq \mathbf{1}^T K x$$

However if we define $\alpha := \mathbf{1}^T K$, then we can interpret IPT_{IP} as a knapsack problem, where each item in the knapsack corresponds to a feature in our dataset. In this way, we can interpret the ith item α_i as a collection of people with feature i distinct from our choice user. In other words, the weight of each item α_i is the number of people with feature i distinct. This means that for a chosen subset of items (or features) x to put in our knapsack, we can interpret the total weight of each of these items $\alpha^T x = \mathbf{1}^T K x$, as the total number of people in our knapsack, where we count each person once for each distinct feature they have. This means if person i has 4 distinct features from poi, then they will be counted 4 times and provide a weight of 4 into the knapsack. As $(Kx)_i$ represents the number of unique features individual i has from poi, every person is added $(Kx)_i$ times to the bag. In other words, $(Kx)_i$ represents how many items individual i is in out of the subset x. In this way, counting the number of "unique" people in our knapsack corresponds to counting the number of people with ≥ 1 distinct feature from poi, which is equivalent to $||Kx||_0$. To illustrate this, consider an example K, and feature vector x

$$K = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}, \quad x = [1, 1, 0]^T, \quad Kx = \begin{bmatrix} 0 \\ 1 \\ 2 \\ 1 \end{bmatrix}, \quad \alpha = \begin{bmatrix} 2 & 2 & 1 \end{bmatrix}$$

Where we he have three individuals (call them A,B,C) aside from poi (row 1) and three features. We can view the items in our knapsack $\alpha = \mathbf{1}^T K$ as the following objects (one for each of the three features).



Here we only select the first two items $x = [1, 1, 0]^T$, so the weight of our knapsack $\mathbf{1}^T K x = \alpha^T x = 4$. However, there are only three unique individuals in our knapsack, as we double count B. In order to only count B once, we just take $||Kx||_0 = 3$ (i.e. count the number of non-zeros)

4.2 Convex Bounds

4.2.1 SDP Upper bound

Consider the following SDP:

$$\begin{aligned} \text{IPT}^{\text{SDP}} = \quad & \underset{x\in\mathbb{R}^n}{\text{maximize}} \quad w^T x \\ & \text{subject to} \quad \begin{bmatrix} X & Z & x \\ Z^T & Y & y \\ x^T & y^T & 1 \end{bmatrix} \succeq 0 \\ & \qquad\qquad x_i = X_{ii}, \quad \forall i \leq n \\ & \qquad\qquad y_j = Y_{jj}, \quad \forall j \leq m \\ & \qquad\qquad \mathbf{1}^T y \geq rm \end{aligned} \tag{IPT$^{\text{SDP}}$}$$

Imagine a vector $y \in \mathbb{R}^n$ with each entry define as

$$y_i = \begin{cases} 1 & \text{if} \end{cases}$$

Of course we can add on a the linear constraint from the MIP upper bound $\mathbf{1}^T K x \leq m(1-r)\mathbf{1}^T x$ to get a tighter SDP relaxation.

4.2.2 Greedy Lower Bound

Define $\bar{K} = 1 - K$ as the matrix with the entries in K flipped. Now for any column $\bar{K}_i$ of matrix $\bar{K}$, define its value

$$v(\bar{K}_i) = (1^T \bar{K}_i) w_i$$

and order the columns from largest to smallest value, then perform a greedy selection algorithm by selecting the columns with the largest value up until the privacy constraint is violated. The idea is we both want to select columns which are weighted highly and who share features with the kth column. Note that we can always rewrite $\mathbf{1}^T \bar{K} = m - \mathbf{1}^T K$, so this formulation need not involve a new $\overline{K}$.[1]

5 Methods to Solve IPT

5.0.1 Dynamic Programming Algorithm?

5.0.2 Reformulation of L0 norm (Oktay Gunluk)?

To Do:

1. SDP (tighter) relaxation: Think about SDP relaxation for entire problem, by considering the SDP relaxation for an integer constraint, and seeing if there is an interpretation in the 0-norm part of the problem (think SDP relatation from lecture 12 orf 523)

[1] Inspired by question 1 on the 2016 ORF 363 Final

2. Better Algorithm: Can well known "Dynamic Programming" algorithm for knapsack be generalized to IPT

3. Better LP relaxation/Bridge between relaxation & algorithm: Jeff's 2016 ORF 363 Final problem, is there a way to draw a connection between LP relaxation and the greedy algorithm (i.e. recall that the greedy algorithm for knapsack was equivalent to LP relaxation and pushing inequalities to equalities etc.) - YES they are identical, but need to prove

4. Solving original problem: Oktay Gunluk has written IP formulations of l_0 norm.

5. Looking at the Dual of the LP relaxation of the knapsack