Facultad de Matemáticas

Grado en Matemáticas

Trabajo Fin de Grado

Privacidad diferencial en Ciencia de los Datos

Realizado por:
Carlos Pinto Pérez

Dirigido por:
D. Joaquín Borrego Díaz

Departamento:
Ciencias de la Computación e Inteligencia Artificial

Sevilla, Junio 2018

Abstract

This final project shows the main concepts related to privacy in Data Science. We present the definition of Differential Privacy (DP), which has emerged as the standard privacy notion for research in this topic, and focus on some of its key properties. Then, we explore the Laplacian Mechanism and the Exponential Mechanism as two fundamental tools for achieving DP, i.e. strong guarantee against an open world environment. We also give a case study about these questions and provide a theoretical analysis of the Sparse Vector Technique, presenting some experimental libraries to see how its works in detail.

Resumen

En este trabajo presentamos los conceptos principales sobre privacidad en Ciencia de Datos. Introducimos la definición de Privacidad Diferencial, que se ha desarrollado en los últimos años como una noción fundamental para la investigación en esta materia. Desarrollamos sus propiedades y hacemos un estudio introductorio y autocontenido de dos herramientas clave: el mecanismo de Laplace y el mecanismo exponencial. Ofrecemos también un caso práctico donde vemos la teoría en ejecución; exponemos la Técnica del Vector Disperso como un método con abundantes usos en el análisis de datos y terminamos mencionando dos librerías con el código informático oportuno para aproximarnos a estas cuestiones.

Índice general

Índice de figuras

Introducción

Motivación

La privacidad en Ciencia de Datos es una necesidad que ha ido ganando importancia en los últimos tiempos. A la vez que aumentan el volumen, la variedad y la masificación de los datos (debido entre otros factores al bajo coste de almacenamiento y la utilidad de técnicas predictivas sobre ellos), se abre un riesgo para el individuo sobre la exposición a la que se somete. En particular, es muy importante estudiar la relación entre los permisos que el usuario pueda ofrecer para el uso de la información que genera diariamente y la seguridad de no verse directa o indirectamente comprometido hacia algún tercero.

En este trabajo pretendemos trazar un camino sobre los problemas que surgen de esta cuestión y los principales métodos y técnicas para proteger nuestra privacidad, teniendo en cuenta que una *anonimización* completa de los datos choca frontalmente con el uso que podamos obtener en muchas ocasiones a través de ellos. Buscamos, por tanto, un equilibrio entre las propiedades o resultados que podamos extraer de grandes grupos de datos y la protección de los elementos particulares que los componen.

La privacidad diferencial surge como la herramienta principal para hacer frente a esta materia. Después de considerar otras aproximaciones relativamente recientes, como la *k-anonimización* (en inglés, *k-anonimity*[1][16]), o el estándar «puerto seguro» (*Standard Safe Harbor*[2]), se presenta como la forma más potente de «blindar» datos sensibles expuestos en un *mundo abierto* (es decir,

[1]Dividimos los datos en *cuasi-identificadores* y en *atributos sensitivos*, donde se supone que el adversario solo puede conocer los primeros. La *k-anonimity* consiste en liberar una tabla de resultados donde cada cuasi-identificador aparece al menos k veces, para prevenir la reidentificación.

[2]Una especificación de 18 tipos de elementos (nombres, números de teléfono, ciertas subdivisiones geográficas...) que deben ser eliminadas o distorsionadas para cumplir los estándares de *desidentificación* de la HIPAA (Health Insurance Portability and Accountability Act).

existe la posibilidad de que se usen fuentes de datos ajenas a las que conocemos en ámbito local, lo que ofrece una posibilidad adicional de *reidentificar* a los usuarios a través del cruce y la combinación de esta información).

Riesgos generales

Se deben considerar dos interpretaciones de Privacidad: la *privacidad como secreto* y la *privacidad como control* [12]. Veremos que la primera es imposible: En [5], T. Dalenius propone la siguiente definición: *El acceso a una base de datos estadística no debería permitir que alguien pueda aprender algo sobre un individuo que no pueda ser descubierto sin el acceso a ella.* Sin embargo, tal propuesta es inviable [7]: si imaginamos que no se sabe que fumar causa cáncer de pulmón, y un estudio demuestra que esto es cierto, un fumador cualquiera x, no envuelto en la investigación, verá afectada su privacidad: se podrá deducir que tiene más riesgo que otras personas de padecer esa enfermedad y, por ejemplo, su seguro médico podría subir de precio. Por tanto, la privacidad *ad omnia* es inalcanzable.

Para estudiar la otra opción, la privacidad como control, se introduce el Principio de Datos Personales (PDP, *Personal Data Principle*):

PDP: La privacidad de datos proporciona un control individual sobre nuestra propia información personal. La privacidad de una persona no se ve alterada si sus datos no son utilizados. Esto no significa que no se pueda aprender información sobre el individuo o no se le pueda provocar ningún daño; tener esta pretensión no es realista.

El enfoque que aborda la privacidad diferencial (DP, *Differential Privacy*) es el de conservar, con una medida ε-δ, las mismas conclusiones a partir de unos datos independientemente de si la información de un individuo particular está contenida en ellos o no. Una forma de alcanzarla consiste en introducir cierto ruido en los datos revelados de forma que la probabilidad de obtener algún resultado de ellos sea parecida (en función de ε y δ) a la que se podría obtener de otro conjunto de datos que difiera del original en un elemento.

Enumeramos a continuación una serie de aspectos técnicos que se dan en la práctica sobre el problema de la privacidad:

1. **Demasiado ruido inutiliza los datos**: yéndonos al caso más extremo, podemos *anonimizar* totalmente un conjunto de datos devolviendo un resultado completamente aleatorio. Por otro lado, devolver los datos

sin alterar (en su versión más *enriquecida*) puede abrir la posibilidad de que recibamos ataques mediante *reidentificación* vía comparación con los datos de otras bases de datos. Incluso esquivando la reidentificación total, la exposición de ciertos datos puede ser dañina[3].

2. **Revelar datos *inofensivos* no es viable**. Publicar cualquier tipo de información, incluso la ordinaria, puede acabar siendo problemático. Imaginemos que sabemos cuándo una persona compra algún tipo de alimento; si de repente deja de comprar algo, podemos suponer que ha habido algún cambio en su vida. Sea cierta la hipótesis o no, el usuario está comprometido.

3. **Debemos proteger la información de todos los sujetos**. A veces se abre la posibilidad de revelar completamente los datos de solo unos pocos elementos de la base de datos para permitir preservar la privacidad de la mayoría. Esto no nos resulta aceptable.

4. **En el contexto de peticiones (*queries*) a una base de datos, filtrarlas bajo algún criterio no es una opción**. Nos enfrentaríamos a dos problemas principales: primero, rechazar ciertas peticiones puede ofrecer en sí mismo pistas de lo que estamos ocultando. Segundo, un algoritmo que compruebe qué peticiones se están haciendo intentando descubrir algún dato particular a través de varias respuestas podría no ser eficiente; a más complejidad, las peticiones se pueden maquillar para pasar por alto estas barreras.

   Por ejemplo, no responder a las peticiones que se realicen sobre conjuntos pequeños o específicos (para proteger la individualidad, respetando por ejemplo el protocolo *Safe Harbor*) puede comprometerse de la siguiente forma: supongamos que tenemos un elemento llamado x en un conjunto $\mathcal{S}$, y queremos saber si tiene la propiedad $\mathcal{P}$. Podemos preguntar: ¿cuántos elementos de $\mathcal{S}$ cumplen $\mathcal{P}$? y seguidamente ¿cuántos elementos de $\mathcal{S}$, que no sean x, cumplen $\mathcal{P}$?

5. **Los resúmenes estadísticos no conservan la privacidad**. Se puede deducir del punto anterior que se puede lograr un *ataque diferencial* para reconstruir el conjunto total de datos a partir de las muestras obtenidas. En [8] se evidencian las dificultades que supone proteger una base de datos incluso para una cantidad lineal de peticiones[4].

[3] Por ejemplo, el registro de facturas en un restaurante durante un día puede servir para acotar la información que tenemos para saber qué plato ha podido consumir alguno de los clientes.

[4] Responder con cierta precisión un número grande de peticiones lleva a que la privacidad se consuma gradualmente o incluso a que se agote. A esto se le ha llamado Ley Fundamental

6. **Cuando existe correlación entre los datos, el nivel de la protección de la privacidad puede verse muy reducido**. Tomemos por ejemplo un grupo de personas del que sabemos que es probable que todos pertenezcan a una misma asociación, sin saber cuál es. Aunque los datos extraídos tengan ruido y los puedan proteger a cada uno individualmente, es fácil ver que una vez comprometida la *privacidad del grupo*, la privacidad individual (incluso en su versión de privacidad como control) queda fundamentalmente afectada. Sobre estos casos, como el control de un tutor sobre los datos de un menor o una familia, o el de cualquier directivo sobre una asociación completa, entran en juego cuestiones morales, éticas y legales más delicadas.

7. **Aplicar modelos de aprendizaje sobre datos privados requiere al menos dos capas de privacidad**. La primera, la que tratamos, es la de extraer información útil sobre un grupo mientras los datos de cada individuo están protegidos. La segunda, y que habría que considerar, es la del uso de las predicciones sobre el grupo. En [6] se cuenta que una familia descubrió que su hija estaba embarazada porque les empezó a llegar publicidad de productos de bebés a partir de un modelo predictivo aplicado sobre la familia completa. Esto pone de manifiesto que aplicar DP en los datos sobre los que se aprende el modelo no es suficiente.

Incidentes notables

El problema de la privacidad ha alcanzado su estatus por sus antecedentes. A modo de ilustración, describimos algunos casos donde no se tuvieron en cuenta los riesgos enumerados en el apartado anterior:

- En 1997 se identificó el registro médico del gobernador de Massachusetts, a partir de la relación de varias bases de datos de registros públicos. Al ser una personalidad importante, al caer enfermo se sabía que sus datos estaban registrados en la factura médica de algún hospital. A través del uso de los censos de voto, se pudo *reidentificar* (esto es, un *linkage attack* en el ámbito del mundo abierto) su información [3].

- En 2006 Netflix liberó los datos de 500.000 usuarios para un concurso público que consistía en intentar mejorar su sistema de recomendaciones,

de Recuperación de Información (*Fundamental Law of Information Recovery*) [1]. Con más precisión, cuando el ruido añadido es de $o(\sqrt{n})$, puede hacerse un ataque eficiente usando n peticiones. Una de las ideas que evolucionó hasta lo que hoy es la (ε, δ)-DP es la de estudiar protecciones frente a una cantidad de peticiones de orden sublineal.

tomando como principal medida de privacidad la eliminación de datos sensibles (nombres de usuario, etc). En [14] se hizo un estudio sobre cómo otro *linkage attack*, utilizando los datos públicos de The Internet Movie Database (IMDb), permitió reidentificar a usuarios que aparecieron en las muestras que publicó Netflix.

- AOL (una empresa de servicios de Internet) liberó los historiales de búsqueda, en un espacio de tres meses, de 650.000 de sus usuarios. La medida más destacable de protección fue sustituir las identificaciones de los usuarios por números aleatorios. Dos periodistas del New York Times reidentificaron a una señora de 62 años, Thelma Arnold, usando los datos de agendas telefónicas y observando que varias de sus búsquedas eran sobre información cercana a su entorno [2].

- Durante un tiempo, instituciones como U.S. National Institute of Health (NIH) permitían el acceso público a datos como las frecuencias agregadas de SNPs (*single-nucleotide polymorphisms*, mutaciones de ADN en localizaciones específicas). Eran estudios que comparaban las secuencias de ADN de dos grupos de participantes, unos con la mutación que se estudia y un grupo de control. En [9] propusieron ataques que podían localizar con una alta probabilidad en qué grupo se encontraba alguna persona suponiendo que disponíamos de la información de su ADN. Tras esto, se restringió el acceso público a estos datos.

Aproximación a la Privacidad Diferencial

Antes de presentar la definición de DP necesitamos aclarar tres cuestiones previas:

- La Privacidad Diferencial no es una propiedad sobre los datos finales una vez que los hayamos procesado (*privacidad sintáctica*), ni un algoritmo o conjunto de ellos: es una propiedad que se estudia sobre el algoritmo o el método que utilizamos por el que liberamos los datos.

- Otras aproximaciones de privacidad involucran la relación «conocimiento a priori» frente al «conocimiento a posteriori» del adversario, usando la regla de Bayes. En un mundo abierto donde no conocemos de qué información dispone el atacante antes de acceder a los datos revelados, y dada esta imposibilidad de acotar lo que sabe o puede saber, no podemos pretender aplicar esta idea a la DP.

- La introducción de ruido no determinista (aleatoriedad) es esencial. Concretamente, cualquier garantía de privacidad «no trivial» que se mantenga independientemente del resto de fuentes o información auxiliar, incluyendo otras bases de datos, estudios, rumores, noticias, estadísticas... requiere aleatoriedad. Para justificarlo, supongamos que utilizamos un algoritmo determinista como protección. Entonces existirían dos bases de datos que difieran solo en los datos de un individuo que frente a la misma pregunta darían información distinta. Haciendo los ajustes necesarios podríamos atacarlo convenientemente.

En [10], C. Dwork propone una caracterización de la DP como garantía: *Independientemente de la información externa, un adversario con acceso a la base de datos «protegida» extrae las mismas conclusiones estén los datos del sujeto en ella o no.*

Mecanismo básico de respuesta aleatoria

Veamos un caso práctico donde se aplica todo lo mencionado anteriormente [7]. Supongamos que queremos hacer un estudio entre un conjunto de personas (digamos un aula de alumnos) sobre alguna materia de la que queremos extraer datos generales respetando la privacidad individual de cada participante, como por ejemplo el consumo de tabaco. Lo planteamos de la siguiente forma:

Cada alumno tiene que responder a la pregunta con un sí o un no. Para ello, les decimos que tiren una moneda con estas condiciones:

$$\begin{cases} \text{Responden la verdad} & \text{si sale cara} \\ \text{Tiran otra moneda} & \text{si sale cruz} \longrightarrow \begin{cases} \text{Responden sí} & \text{si sale cara} \\ \text{Responden no} & \text{si sale cruz} \end{cases} \end{cases}$$

De esta forma, la introducción de ruido nos permite sacar conclusiones generales del estudio a la vez que podemos proteger la privacidad de los participantes. A este caso, variando los parámetros si es necesario, lo llamaremos **mecanismo básico** de respuesta aleatoria.

Objetivo de la memoria

El objetivo de este trabajo es estudiar, de una forma autocontenida, los fundamentos de la (ε, δ)-DP desde una perspectiva matemática y rigurosa: su

definición, los aspectos que abarca, las técnicas más comunes y las garantías que nos puede ofrecer.

Somos conscientes de que desde un estudio completamente teórico normalmente no se aprecia una diferencia entre los problemas y la teoría. Esto sí ocurre en la práctica. Buscamos combinar, en la medida de lo posible y entendiendo lo amplio que es este campo, aún siendo reciente, la base fundamentada de los conceptos y las herramientas y cómo todo este corpus se traduce en los algoritmos que se aplican en los casos reales.

Estructura del trabajo

El contenido de la memoria se divide en cinco capítulos y otro final dedicado a las conclusiones de este trabajo. El primero sirve de introducción a las bases de la Privacidad Diferencial, la modelización matemática y el significado de las variables más importantes: el presupuesto y la pérdida de privacidad, sus propiedades de composición, tanto de forma secuencial como en paralelo, y un ejemplo final.

En los capítulos 2 y 3 veremos las dos técnicas más comunes para asegurar el cumplimiento de la (ε, δ)-DP: el mecanismo de Laplace y el mecanismo exponencial, que añaden ruido de forma conveniente y precisa respetando la utilidad de los datos.

En el capítulo 4 aplicaremos, a través de un problema de cálculo de la media, la teoría vista hasta entonces: será un recorrido entre varios algoritmos y unas conclusiones sobre sus ventajas y diferencias.

Por último, en el capítulo 5 estudiaremos la Técnica del Vector Disperso (*Sparse Vector Technique*), que permite identificar las *queries* más importantes de entre un conjunto de ellas. Incluimos estudios sobre la optimización de estas técnicas y software para aplicarlas. Un resultado llamativo es que añadiendo un «bajo coste» de privacidad podremos no solo filtrar las peticiones sino responderlas.

Capítulo 1

Privacidad Diferencial

En este capítulo introducimos la definición de la DP y sus términos y propiedades más notables. Para situar al lector, nos moveremos en un contexto *offline* donde en cada problema nuestro objetivo es liberar unos datos *saneados* (que han pasado por los filtros de privacidad) al público en general.

1.1. Definición de Privacidad Diferencial

Primero necesitamos introducir los conceptos principales sobre los que trabajaremos:

Definición 1.1.1 *Llamamos* ***base de datos*** *D a un conjunto de datos, normalmente dispuestos en n filas, donde cada una de ellas representa los datos de algún individuo.*

Definición 1.1.2 *Una* ***petición*** *(query) es una función que se aplica a una base de datos D.*

Definición 1.1.3 *Un* ***mecanismo*** *$\mathcal{M}$ (con rango B) es un algoritmo que actúa sobre una base de datos D, un conjunto de datos $\mathcal{X}$ llamado* ***universo*** *compuesto por todas las filas posibles de D, un conjunto de peticiones (opcional), y devuelve una respuesta aleatoria:*

$$\begin{aligned} \mathcal{M} : A &\to B \\ a &\longmapsto b \text{ con probabilidad } (\mathcal{M}(a))_b. \end{aligned}$$

Con $A = D \times \mathcal{X} \times F$, llamando F al conjunto de peticiones. A veces, por comodidad de notación, se puede suprimir $\mathcal{X}$ y F del conjunto de entrada.

Ejemplo 1.1.4 *Tomando de referencia el mecanismo básico de la introducción, siendo n el número de alumnos que participan, escribiendo $\top$ como «sí» y $\bot$ como «no», tenemos que:*

$$\begin{cases} \mathcal{X} & = \{\top, \bot\}^n \\ D & = (a_1, ..., a_n) \in \{\top, \bot\}^n \text{ (conjunto de datos verdaderos)} \\ F & = \{f_i, i = 1, ..., n : f_i(D) = a_i\} \end{cases}$$

Donde cada $\mathcal{M}(D, \mathcal{X}, F) = \mathcal{M}(D)$ es un experimento aleatorio descrito como vimos.

Notemos que podemos representar cualquier base de datos D como un conjunto de determinados $x \in \mathcal{X}$. Durante los capítulos 2 y 3, por simplificación (y asumiendo que podemos caracterizar los elementos de una base de datos como elementos de $\mathbb{N}$), hablaremos de bases de datos como variables $x \in \mathbb{N}^{|\mathcal{X}|}$.

Definición 1.1.5 *Al número de filas de una base de datos D lo llamaremos **tamaño** de D y lo notaremos $|D|$.*

Definición 1.1.6 *Llamaremos **distancia entre bases de datos** al número de filas en que difieran dos bases de datos (es decir, es la **distancia de Hamming** adaptada para bases de datos).*

Definición 1.1.7 *A dos bases de datos D y D' que difieran en a lo sumo una fila, las llamaremos bases de datos **adyacentes**.*

Observación 1.1.8 *En las dos definiciones anteriores, las bases de datos pueden tener el mismo tamaño o no.*

A partir del ejemplo 1.1.4 podemos hacernos una idea de lo que nos interesa estudiar principalmente:

Definición 1.1.9 *Llamamos **pérdida de privacidad** a la cantidad:*

$$\mathcal{L}^{(\xi)}_{\mathcal{M}(D)||\mathcal{M}(D')} = \ln\left(\frac{\Pr[\mathcal{M}(D) = \xi]}{\Pr[\mathcal{M}(D') = \xi]}\right)$$

La pérdida de privacidad $\mathcal{L}$ indica, si es positiva, que ξ es más probable de ocurrir sobre D que sobre D', y recíprocamente si es negativa. La motivación de la Privacidad Diferencial es asegurar que para D y D' adyacentes, esta pérdida de privacidad está acotada por ε con probabilidad de al menos $1 - \delta$. Formalmente, generalizando el conjunto de valores aceptados T (en vez de ξ), y tomando la exponencial en la expresión anterior:

Definición 1.1.10 *Un mecanismo $\mathcal{M}$ cumple* ***(ε, δ)-Privacidad Diferencial*** *(que denotaremos (ε, δ)-DP), con $\varepsilon \geq 0$, si y solo si para dos bases de datos adyacentes cualesquiera D y D', se tiene que:*

$$\forall T \subseteq Rango(\mathcal{M}) \qquad \Pr[\mathcal{M}(D) \in T] \leq e^{\varepsilon} \Pr[\mathcal{M}(D') \in T] + \delta$$

Donde parametrizamos la pérdida de privacidad desconocida $\mathcal{L}$ a través de ε y añadiendo una variable adicional δ.

Una interpretación de la DP es la siguiente: ε y δ estiman la variación de la privacidad de algún individuo según este decida que sus datos pertenezcan a algún estudio (o base de datos) o no. Es decir, podemos considerar D como una base de datos que incluye la información del sujeto x, y D' como la base de datos donde, o se ha eliminado la fila que contiene los datos de x, o se ha sustituido por otra. La DP, siguiendo a Dwork, ofrece la garantía de «ocultar» los datos de cada individuo minimizando el perjuicio sobre la utilidad que se pueda extraer de ellos. La variable ε, según hemos visto, está muy relacionada con una cota superior de $\mathcal{L}$, y δ es un margen que nos permitimos debido a los cálculos.

Normalmente estaremos interesados en valores de δ menores que las inversas de cualquier polinomio respecto a $|D|$. En particular, para δ del orden de $1/|D|$ la definición permitiría la publicación de los registros de un porcentaje pequeño de los individuos sin ningún ruido, es decir, sin *sanear* [7]. Como vimos en la introducción, es uno de los casos que hay que evitar.

Una cuestión importante a considerar es qué valores de ε consideraremos que son suficientemente pequeños o deseables. Si podemos conseguir ε muy cercanos al 0, podríamos aspirar a alcanzar, por ejemplo, $(2\varepsilon, 0)$-DP sin que sea demasiado alarmante.

Pero ¿qué medida tiene que tener un ε para ser indeseable? Un valor alto (digamos, $\varepsilon = 10$) indica que existen dos bases de datos adyacentes donde existe una misma publicación t para la que la probabilidad de que se hubiera publicado t bajo una de ellas (y no en la otra) es muy alta. Por otro lado, dicha publicación t podría ser muy improbable de ocurrir frente a todas las del rango del mecanismo, o si el adversario no tiene la suficiente información auxiliar, podría no tener que reconocer que t es una respuesta sensible.

En [12] se ofrece una tabla (figura 1.1) que concreta, en términos generales, los intervalos de seguridad que implican cada elección de un ε. Para una hipótesis que el adversario tome con probabilidad $p = \Pr[\mathcal{M}(D) = \xi]$, indica el rango de probabilidad p' al que esta varía al aplicar ε-DP. Por ejemplo, para un $\varepsilon = 1$,

ϵ	0.01	0.1	1	5	10
$\lambda = e^{\epsilon}$	1.01	1.11	2.72	148	22026
$p = 0.001$	(0.0010, 0.0010)	(0.0009, 0.0011)	(0.0004, 0.0027)	(0.0000, 0.1484)	(0.0000, 1.0000)
$p = 0.01$	(0.0099, 0.0101)	(0.0090, 0.0111)	(0.0037, 0.0272)	(0.0001, 0.9933)	(0.0000, 1.0000)
$p = 0.1$	(0.0990, 0.1010)	(0.0905, 0.1105)	(0.0368, 0.2718)	(0.0007, 0.9939)	(0.0000, 1.0000)
$p = 0.5$	(0.4950, 0.5050)	(0.4524, 0.5476)	(0.1839, 0.8161)	(0.0034, 0.9966)	(0.0000, 1.0000)
$p = 0.75$	(0.7475, 0.7525)	(0.7237, 0.7738)	(0.3204, 0.9080)	(0.0051, 0.9983)	(0.0000, 1.0000)
$p = 0.99$	(0.9899, 0.9901)	(0.9889, 0.9910)	(0.9728, 0.9963)	(0.0067, 0.9999)	(0.0000, 1.0000)

Figura 1.1: Tabla sobre la elección de ε. Extraído de [12].

una creencia que el adversario toma con posibilidad $p = 0{,}5$ podría llegar a variar a algún valor dentro de $(0{,}1839, 0{,}8161)$.

En algunas partes de la teoría suavizaremos el concepto que estudiaremos, tomando $\delta = 0$:

Definición 1.1.11 *Un mecanismo $\mathcal{M}$ cumple ε-**Privacidad Diferencial** (ε-DP) si cumple $(\varepsilon, 0)$-DP.*

Destaquemos que bajo ε-DP, cualquier salida ξ es (casi siempre) igualmente probable de resultar en cualquier base de datos adyacente a D. Con la (ε, δ)-DP introducimos ventajas o desventajas sobre estas otras bases de datos.

Lema 1.1.12 *Con las condiciones anteriores, un mecanismo $\mathcal{M}$ cumple ε-DP si:*

$$\forall t \in Rango(\mathcal{M}) \qquad \frac{Pr[\mathcal{M}(D) = t]}{Pr[\mathcal{M}(D') = t]} \leq e^{\varepsilon}$$

donde asumimos que $\frac{0}{0} = 1$

Es decir, cuando $\delta = 0$:

$$\exp\left(\mathcal{L}^{(t)}_{\mathcal{M}(D)||\mathcal{M}(D')}\right) = \frac{Pr[\mathcal{M}(D) = t]}{Pr[\mathcal{M}(D') = t]} \leq \exp(\varepsilon)$$
$$\implies \mathcal{L}^{(t)}_{\mathcal{M}(D)||\mathcal{M}(D')} \leq \varepsilon$$

Según la relación entre D y D' (la forma en la que difieran en un elemento) tenemos dos tipos de privacidad diferencial:

Definición 1.1.13 *Si $|D| = |D'|$ en las condiciones de la definición anterior, hablaremos de **DP acotada**, es decir, D y D' solo difieren en los elementos de una fila. Si $|D| \neq |D'|$ (añadiendo o eliminando una fila) lo llamaremos **DP no acotada**.*

Proposición 1.1.14 *Notamos que si un mecanismo $\mathcal{M}$ cumple ε-DP acotada, entonces cumple 2ε-DP no acotada.*

Demostración: Basta observar que reemplazar una fila por una distinta es lo mismo que eliminar una fila y luego añadir otra. □

1.2. Propiedades

Una de las ventajas que tiene la definición matemática de DP es que, una vez retirados los datos de un individuo respetando la utilidad (que en general no nos sirve como objetivo, pues deseamos retirar los datos de *todos* los participantes), podemos repetir el proceso hasta abarcar toda la muestra. Para describir esta idea necesitamos formalizar la noción de composición de mecanismos.

1.2.1. Composición en serie

Vamos a ver dos tipos de composición en este sentido, una de *post-procesado* donde aplicamos cada mecanismo sobre el resultado del anterior, y otra *secuencial* en la que aplicamos los mecanismos tanto a los resultados de los anteriores como a la base de datos original.

Proposición 1.2.1 (Composición, post-procesado) *Sean $\mathcal{M}_1$ y $\mathcal{M}_2$ dos mecanismos tales que $\mathcal{M}_1$ cumple (ε, δ)-DP. Entonces $\mathcal{M}_2 \circ \mathcal{M}_1$ cumple (ε, δ)-DP.*

Demostración: Tomamos dos bases de datos adyacentes cualesquiera D y D'. Sea $S \subseteq Rango(\mathcal{M}_2)$ fijo, y $T = \{r \in Rango(\mathcal{M}_1) : \mathcal{M}_2(r) \in S\}$:

$$D \xrightarrow{\mathcal{M}_1} T \xrightarrow{\mathcal{M}_2} S$$

Entonces:

$$\begin{aligned} Pr[\mathcal{M}_2 \circ \mathcal{M}_1(D) \in S] &= Pr[\mathcal{M}_1(D) \in T] \\ &\leq e^{\varepsilon} Pr[\mathcal{M}_1(D') \in T] + \delta \\ &= e^{\varepsilon} Pr[\mathcal{M}_2 \circ \mathcal{M}_1(D') \in S] + \delta \end{aligned}$$

□

Este resultado no solo permite una composición cómoda de mecanismos, sino que indica que cualquier analista de datos, sin conocimiento previo sobre D, no

puede tratar un respuesta saneada a través de un $\mathcal{M}$ que cumpla (ε, δ)-DP y hacerlo *menos* privado.

Proposición 1.2.2 (Composición, secuencial) *Sean $\mathcal{M}_1(D)$ y $\mathcal{M}_2(s, D)$ dos mecanismos que cumplen ε_1-DP y ε_2-DP respectivamente. Entonces $\mathcal{M}(D) = \mathcal{M}_2(\mathcal{M}_1(D), D)$ cumple $(\varepsilon_1 + \varepsilon_2)$-DP.*

Demostración: Tomamos dos bases de datos adyacentes cualesquiera D y D'. Sea $S = Rango(\mathcal{M}_1)$. Entonces, $\forall t \in Rango(\mathcal{M}_2)$:

$$\begin{aligned} Pr[\mathcal{M}_2(\mathcal{M}_1(D), D) = t] &= \sum_{s \in S} Pr[\mathcal{M}_1(D) = s] Pr[\mathcal{M}_2(s, D) = t] \\ &\leq \sum_{s \in S} e_1^{\varepsilon} Pr[\mathcal{M}_1(D') = s] e_2^{\varepsilon} Pr[\mathcal{M}_2(s, D') = t] \\ &= e^{\varepsilon_1 + \varepsilon_2} Pr[\mathcal{M}_2(\mathcal{M}_1(D'), D') = t] \end{aligned}$$

Hemos hecho los cálculos en el caso discreto. El razonamiento es análogo para el caso continuo. □

Notemos que si en la proposición 1.2.1 consideramos solo la ε-DP, esta se convierte en un caso particular de la proposición 1.2.2, interpretando que $\mathcal{M}_2$ cumple 0-DP. Esto podemos ampliarlo a un resultado más general.

Lema 1.2.3 (Composición general secuencial) *Sean $\mathcal{M}_i$ un conjunto de mecanismos (que puedan recibir de entrada no solo necesariamente la base de datos) que cumplen, respectivamente, ε_i-DP para $i = 1, ..., k$. Entonces, $\mathcal{M}(D) = \boldsymbol{t}$, donde $\boldsymbol{t} = \langle t_1, t_2, ..., t_k \rangle$ y*

$$t_1 = \mathcal{M}_1(D), t_2 = \mathcal{M}_2(t_1, D), ..., t_k = \mathcal{M}_k(\langle t_1, t_2, ..., t_k - 1 \rangle, D)$$

cumple $(\sum_{i=1}^{k} \varepsilon_i)$-DP.

Debido a estos resultados, al parámetro ε se le llama **presupuesto de privacidad** (*privacy budget*), ya que lo dividimos en una suma de composiciones que se van consumiendo durante el procesado.

Nota 1.2.4 *En las dos proposiciones anteriores se pueden obtener los mismos resultados para (ε, δ)-DP, de forma que se cumple que la composición general secuencial de $\mathcal{M}_i$ para $i = 1, ..., k$, donde cada uno de ellos cumple $(\varepsilon_i, \delta_i)$-DP, cumple $(\sum_{i=1}^{k} \varepsilon_i, \sum_{i=1}^{k} \delta_i)$-DP. La demostración puede verse en [7].*

1.2.2. Convexidad

La DP también satisface propiedades interesantes que nos servirán más adelante para otros tipos de composición.

Proposición 1.2.5 *Sean $\mathcal{M}_1$ y $\mathcal{M}_2$ dos mecanismos que cumplen ε-DP, y $p \in [0,1]$. Consideremos $\mathcal{M}$ al mecanismo que aplica $\mathcal{M}_1$ con probabilidad p y $\mathcal{M}_2$ en el caso contrario. Entonces $\mathcal{M}$ cumple ε-DP.*

DEMOSTRACIÓN: Tomamos dos bases de datos adyacentes cualesquiera D y D'. Tenemos que, $\forall t \in Rango(\mathcal{M})$:

$$\begin{aligned} Pr[\mathcal{M}(D) = t] &= pPr[\mathcal{M}_1(D) = t] + (1-p)Pr[\mathcal{M}_2(D) = t] \\ &\leq pe^{\varepsilon}Pr[\mathcal{M}_1(D') = t] + (1-p)e^{\varepsilon}Pr[\mathcal{M}_2(D') = t] \\ &= e^{\varepsilon}(pPr[\mathcal{M}_1(D') = t] + (1-p)Pr[\mathcal{M}_2(D') = t]) \\ &= e^{\varepsilon}Pr[\mathcal{M}(D') = t] \end{aligned}$$

□

Como en el caso anterior, este resultado lo podemos generalizar:

Proposición 1.2.6 (Convexidad en el caso general) *Sean $\mathcal{M}_i$ mecanismos que cumplen ε-DP y $p_i \in [0,1]$ tales que $\sum_{i=1}^{k} p_i = 1$, con $i = 1, ..., k$. Sea $\mathcal{M}$ el mecanismo que aplica $\mathcal{M}_i$ con probabilidad p_i. Entonces $\mathcal{M}$ cumple ε-DP.*

1.2.3. Composición en paralelo

En vez de componer utilizando una sucesión de mecanismos sobre la misma base de datos, ahora vamos a estudiar cómo aplicar la técnica de *divide y vencerás*, es decir, subdividir el conjunto D en otros más pequeños para aplicar la DP «en paralelo» a lo largo de D.

Definición 1.2.7 *Definimos una **función de partición** f sobre D (con rango k) como sigue: sea g una función auxiliar que se aplica sobre cada fila de D devolviendo para cada una un entero entre 1 y k. Entonces el resultado de aplicar f sobre D son las particiones $D_1, D_2, ..., D_k$ donde cada D_i contiene a las filas $d \in D$ tales que $g(d) = i$. Es decir, f proporciona las clases de equivalencia de la relación:*

$$x \sim x' \Leftrightarrow g(x) = g(x')$$

Definición 1.2.8 *Diremos que un algoritmo es* ***no determinista*** *si dos ejecuciones sobre la misma entrada pueden diferir. Una* ***función no determinista*** *f es la función que calcula el resultado de un algoritmo no determinista. En caso contrario diremos que f es* ***determinista****.*

Ahora estamos en condiciones de formular y demostrar los dos resultados principales de esta subsección.

Proposición 1.2.9 (Composición en paralelo, DP no acotada) *Sean $\mathcal{M}_i$ un conjunto de k mecanismos que cumplen, respectivamente, ε_i-DP no acotada para todo $i = 1, ..., k$, y una función de partición determinista f que divide a D en $D_1, ..., D_k$. La composición:*

$$\mathcal{M}_1(D_1), \mathcal{M}_2(D_2), ..., \mathcal{M}_k(D_k)$$

satisface $(\text{máx}_{i\in[1..k]}\, \varepsilon_i)$*-DP.*

Demostración: Tomamos dos bases de datos cualesquiera D y D' donde asumimos sin pérdida de generalidad que D contiene una fila más que D'. Sean $\{D_i\}_{i=1}^{i=k}$ y $\{D'_i\}_{i=1}^{i=k}$ las particiones de D y D' bajo f, respectivamente. Observamos que existe un índice j tal que D_j tiene una fila más que D'_j y que $\forall i \neq j$, $D_i = D'_i$. Llamemos $\mathcal{M}(D)$ a la composición

$$\mathcal{M}_1(D_1), \mathcal{M}_2(D_2), ..., \mathcal{M}_k(D_k)$$

Como cada mecanismo se ejecuta independientemente en cada D_i, podemos construir $t = (t_1, ..., t_k)$ con cada $t_i \in Rango(\mathcal{M}_i)$ y:

$$\begin{aligned} Pr[\mathcal{M}(D) = t] &= Pr[(\mathcal{M}_1(D_1) = t_1) \wedge ... \wedge (\mathcal{M}_k(D_k) = t_k)] \\ &= Pr[\mathcal{M}_j(D_j) = t_j] \prod_{i\neq j} Pr[\mathcal{M}_i(D_i) = t_i] \\ &\leq e^{\varepsilon_j} Pr[\mathcal{M}_j(D'_j) = t_j] \prod_{i\neq j} Pr[\mathcal{M}_i(D'_i) = t_i] \\ &\leq e^{\text{máx}_{i\in[1..k]}\, \varepsilon_i} Pr[\mathcal{M}(D') = t] \end{aligned}$$

□

Remarcamos que la composición en paralelo se da solo bajo DP no acotada. De lo contrario, en la demostración anterior no habríamos podido escoger un único índice j con las propiedades adecuadas, sino que al ser $|D| = |D'|$ podría existir otro $i \neq j$ de modo que D_i contiene una fila más que D'_i a la vez que D'_j es una fila mayor que D_j. Esto llevaría a que $\frac{Pr[\mathcal{M}_i(D_i)]}{Pr[\mathcal{M}'_i(D_i)]}$ podría no estar acotado por ser $|D_i| \neq |D'_i|$.

Podemos extender la composición en paralelo usando funciones de partición f no deterministas:

Proposición 1.2.10 (Composición en paralelo, particiones aleatorias) *Sean $\mathcal{M}_i$ un conjunto de k mecanismos que cumplen, respectivamente, ε_i-DP no acotada para todo $i = 1, ..., k$, y una función de partición no determinista f que divide a D en $D_1, ..., D_k$. La composición:*

$$\mathcal{M}_1(D_1), \mathcal{M}_2(D_2), ..., \mathcal{M}_k(D_k)$$

satisface $(\text{máx}_{i\in[1..k]}\, \varepsilon_i)$*-DP.*

Demostración: Sea $\varepsilon = \text{máx}_{i\in[1..k]}\, \varepsilon_i$. Consideremos todas las posibles particiones que pueden resultar de aplicar f a D, que son un número finito. Podemos enumerar todos los resultados y considerar la sucesión de funciones deterministas f_i, donde cada una de ellas produce la partición i-ésima de D. En cada uno de estos casos, por la proposición 1.2.9, sabemos que se cumple la composición en paralelo. Por último, sabemos que el efecto no determinista de f se puede reducir a tomar alguna de las ciertas f_i gracias al resultado de convexidad de la proposición 1.2.6. □

1.2.4. Composición avanzada

Además de conseguir que los parámetros de privacidad se puedan degradar más lentamente al ejecutar composiciones de mecanismos, buscamos también la posibilidad de poder manejar formas más complejas de composición. En particular queremos cubrir los dos siguientes escenarios:

1. Uso repetido de diferentes mecanismos sobre la misma base de datos. Incluimos el caso de la repetición de un mismo mecanismo varias veces, y la separación de los datos en bloques privados para tratarlos modularmente. Aquí consideramos, por tanto, la composición secuencial y la composición en paralelo que hemos visto anteriormente.

2. Uso repetido de diferentes mecanismos en *diferentes* bases de datos, que podrían o no contener la información de un mismo individuo. Esto nos permite razonar sobre la acumulación de la pérdida de privacidad de una persona cuyos datos pueden estar dispersos en distintas fuentes. Frente a un mundo abierto, este es un problema fundamentalmente distinto al de estar atacando a una base de datos fija.

Para estudiar este segundo caso, vamos a suponer que el adversario puede cambiar no solo las peticiones que le hace a los mecanismos, sino las bases de datos con las que trata. Sea $\mathcal{F}$ una familia de mecanismos con acceso a bases de datos cualesquiera, por ejemplo, el conjunto de mecanismos que cumplen ε-DP. Imaginemos a un adversario A que dispone de $\mathcal{F}$. Veamos el siguiente caso:

Experimento de composición adaptativa

Para $i = 1, ..., k$:

1. A ataca dos bases de datos adyacentes x_i^0 y x_i^1, con un mecanismo $\mathcal{M}_i \in \mathcal{F}$ y unos parámetros w_i.
2. A recibe $y_i \in \mathcal{M}_i(w_i, x_i^j)$.

Asumimos que el adversario tiene libertad y conocimientos para elegir las bases de datos, los mecanismos y los parámetros que utilizará en función de los resultados que consiga de sus pasos anteriores.

Definición 1.2.11 *Definimos la* ***vista de A*** *del experimento como las elecciones que ha hecho de las bases de datos adyacentes (los superíndices j) y todas las respuestas de los mecanismos con los que ha atacado, $(y_1, ..., y_k)$. (Los x_i^j, $\mathcal{M}_i$ y los w_i pueden reidentificarse a partir de aquí)*

Consideremos que las elecciones x_i^0 mantienen los datos de un individuo B en la base de datos y que difieren de las x_i^1 en que en estas últimas no aparecen. Es decir, pueden darse dos casos extremos, donde en el primero ($j = 0$) el individuo B permite que sus datos sean liberados públicamente en distintos sitios («mundo real») y en el caso de $j = 1$ («mundo ideal») los datos que se liberan no dependen de los parámetros de B. El propósito es que estos dos casos estén «cercanos» en el sentido que requiere la DP: que el adversario no pueda «saber», dadas sus vistas del experimento, si los datos reales de B se han usado o no.

Para plantear formalmente la cuestión necesitamos unos conceptos preliminares:

Definición 1.2.12 *Notaremos V^0 a la vista de A en el «mundo real», es decir, cuando todas las elecciones del individuo B han sido acceder a liberar sus datos. Análogamente, notaremos V^1 a la vista de A en el «mundo ideal» descrito anteriormente.*

Definición 1.2.13 *Llamamos* ***soporte*** *de una función al conjunto de puntos de su dominio donde esta no se anula:*

$$Sop(f) = \{x \in dom(f) : f(x) \neq 0\}$$

Definición 1.2.14 *Se define la* ***máxima divergencia*** *entre dos variables aleatorias Y y Z que tienen el mismo dominio como:*

$$D_\infty(Y||Z) = \max_{S \subseteq Sop(Y) \cap Sop(Z)} \left[\ln \frac{\Pr[Y \in S]}{\Pr[Z \in S]} \right]$$

Notemos su relación con la pérdida de privacidad vista en la definición 1.1.9.

Definición 1.2.15 *La* ***máxima divergencia*** δ***-aproximada*** *entre Y y Z es:*

$$D_\infty^\delta(Y||Z) = \max_{\substack{S \subseteq Sop(Y) \cap Sop(Z) \\ \Pr[Y \in S] \geq \delta}} \left[\ln \frac{\Pr[Y \in S] - \delta}{\Pr[Z \in S]} \right]$$

Definición 1.2.16 *Diremos que la familia $\mathcal{F}$ de mecanismos con acceso a bases de datos cualesquiera cumple* ε***-DP bajo composición k-adaptativa***[1] *si para cada adversario A, tenemos que $D_\infty(V^0||V^1) \leq \varepsilon$, donde V^j indica la vista de A en el experimento de composición adaptativa. Hablaremos de (ε, δ)-DP en el mismo sentido si se cumple que $D_\infty^\delta(V^0||V^1) \leq \varepsilon$*

Teorema 1.2.17 (Composición avanzada) *Para todo $\varepsilon, \delta, \delta' \geq 0$, la clase de mecanismos que cumplen (ε, δ)-DP, satisfacen también $(\varepsilon', k\delta + \delta')$-DP bajo composición k-adaptativa para:*

$$\varepsilon' = \sqrt{2k \ln(1/\delta')}\varepsilon + k\varepsilon(e^\varepsilon - 1)$$

La demostración de este resultado puede verse en [7].

Ejemplo 1.2.18 *Supongamos que a lo largo de toda su vida una persona acaba siendo parte de $k = 10{,}000$ bases de datos donde todas cumplen $(\varepsilon_0, 0)$-DP. Asumiendo que no hay ninguna coordinación entre ellas, ¿cuál debería ser el valor de ε_0 que debemos asegurar para que la pérdida de privacidad acumulativa de esta persona esté acotada por $\varepsilon = 1$ con probabilidad de al menos $1 - e^{-32}$? A partir del teorema 1.2.17 podemos decir, escogiendo $\delta' = e^{-32}$, que es suficiente con tener $\varepsilon_0 \leq 1/801$. Siendo una medida contra cualquier adversario aleatorio en un mundo abierto, es un resultado destacable.*

[1] Differential privacy under k-fold adaptative composition.

1.3. Aplicaciones

Después de haber visto los términos principales de la DP, podemos aplicarla a casos particulares. Veamos algunos:

1.3.1. Aplicación del mecanismo básico

Recordemos el ejemplo de la introducción. Ante los datos *puros*, verdaderos, introdujimos el siguiente sistema para añadir ruido. Frente a cada dato individual, tiramos una moneda con estas opciones:

$$\begin{cases} \text{Se responde la verdad} & \text{si sale cara} \\ \text{Tiramos otra moneda} & \text{si sale cruz} \longrightarrow \begin{cases} \text{Respuesta: sí} & \text{si sale cara} \\ \text{Respuesta: no} & \text{si sale cruz} \end{cases} \end{cases}$$

Se puede interpretar que en los casos donde sale cruz en la primera tirada es como si retiráramos al individuo correspondiente del estudio, de forma a priori desconocida para algún adversario que pretenda la reidentificación. Digamos que un sujeto responde «sí» con probabilidad p, y «no» en caso contrario. Conociendo el procedimiento por el que los hemos hecho pasar, podemos estimar p como sigue:

La probabilidad de obtener un «sí» es (regla de la probabilidad total):

$$p * (1/2) + (1/2)^2 = p/2 + 1/4$$

Considerando que tras el experimento la proporción de «síes» ha sido s, como:

$$2(p/2 + 1/4) - 1/2 = p$$

Basta hacer:

$$\hat{p} = 2s - 1/2$$

Con lo que podemos rescatar la información que buscábamos sin exponer (en teoría) directamente a ningún individuo. Veamos ahora el presupuesto de privacidad que ofrece este mecanismo, es decir, su relación con la DP:

Proposición 1.3.1 *El mecanismo $\mathcal{M}$ descrito anteriormente cumple (ln 3)-DP.*

Demostración: Partimos de una base de datos cualquiera D, de tamaño n y otra adyacente D', que difiere de ella en la fila de un $x \in D$ que podemos considerar fijo. Sea, respectivamente, $x' \in D'$. Notemos que la diferencia entre

x y x' es lo que responderían de verdad dos individuos que tendrían respuestas distintas. Tenemos que:

$$Rango(\mathcal{M}(x)) = Rango(\mathcal{M}(x')) = \{\text{No, Sí}\}$$

que se extiende trivialmente a:

$$Rango(\mathcal{M}(D)) = Rango(\mathcal{M}(D')) = \{\text{No, Sí}\}^n \equiv \{\bot, \top\}^n$$

Estamos forzados a restringirnos a la DP acotada, de lo contrario no podría darse la igualdad anterior. Para cada participante hay cuatro casos posibles:

$$\Pr[\text{Respuesta} = \text{No} \mid \text{Verdad} = \text{No}] = \Pr[\text{Respuesta} = \text{Sí} \mid \text{Verdad} = \text{Sí}] = 3/4$$
$$\Pr[\text{Respuesta} = \text{No} \mid \text{Verdad} = \text{Sí}] = \Pr[\text{Respuesta} = \text{Sí} \mid \text{Verdad} = \text{No}] = 1/4$$

Para cada cambio $x \to x'$ tal y como hemos definido (con un rango fijo t, digamos $t = \{\text{Sí}\}$ y análogo para el «No») ocurre que:

$$\frac{\Pr[\text{Respuesta} = \text{Sí} \mid \text{Verdad} = \text{Sí}]}{\Pr[\text{Respuesta} = \text{Sí} \mid \text{Verdad} = \text{No}]} = \frac{3/4}{1/4} = 3$$
$$\frac{\Pr[\text{Respuesta} = \text{Sí} \mid \text{Verdad} = \text{No}]}{\Pr[\text{Respuesta} = \text{Sí} \mid \text{Verdad} = \text{Sí}]} = \frac{1/4}{3/4} = 1/3$$

Ahora podemos aplicar el lema 1.1.12:

$$\forall t \in Rango(M), t = (t_1, t_2, ..., t_n) :$$
$$\frac{\Pr[\mathcal{M}(D) = t]}{\Pr[\mathcal{M}(D') = t]} = \frac{\Pr[\mathcal{M}(x) = t_i]}{\Pr[\mathcal{M}(x') = t_i]} \leq 3$$

□

1.3.2. Publicación de histogramas

Se puede dar el caso de que deseemos un método para publicar el número de registros (o coincidencias dentro de alguna categoría) dentro de otro grupo de datos, mientras utilizamos DP. Podemos usar la composición en paralelo para, por ejemplo, publicar un histograma.

Un histograma divide el rango de valores en una serie de intervalos y luego operamos para medir a qué intervalo pertenece cada valor. Existen técnicas específicas para esto que pueden verse tanto en [12] como en [7].

Capítulo 2

Mecanismo de Laplace

En este capítulo veremos uno de los métodos principales y más comunes para satisfacer la DP: el mecanismo de Laplace. Siempre que tengamos un problema donde tratemos con alguna función $f : \mathbb{N}^{|\mathcal{X}|} \to \mathbb{R}^n$ (donde podemos restringir $\mathbb{R}$ a $\mathbb{Z}$ o $\mathbb{N}$) esta será una de las opciones inmediatas. Hablamos de problemas de conteo de datos o valores (como los histogramas) o de, por ejemplo, resúmenes estadísticos.

2.1. Introducción

La idea del mecanismo de Laplace es buscar alguna función de distribución que introduzca el ruido (aleatorio) deseado a los datos de forma cómoda y que además cumpla ε-DP. La distribución idónea para el caso (y de donde el mecanismo toma el nombre) es la distribución de Laplace:

Definición 2.1.1 *La **distribución de Laplace** (figura 2.1) centrada en μ y con escala $b > 0$ es la distribución con función de densidad:*

$$Lap(z|\mu, b) = \frac{1}{2b} exp\left(-\frac{|z-\mu|}{b}\right)$$

La distribución de Laplace es una versión simétrica de la distribución exponencial. Tiene esperanza μ y varianza $\sigma^2 = 2b^2$. Normalmente nos referiremos a ella como $Lap(b)$ (asumiendo que $\mu = 0$) para indicar que hablamos de alguna distribución $X \sim Lap(0, b)$.

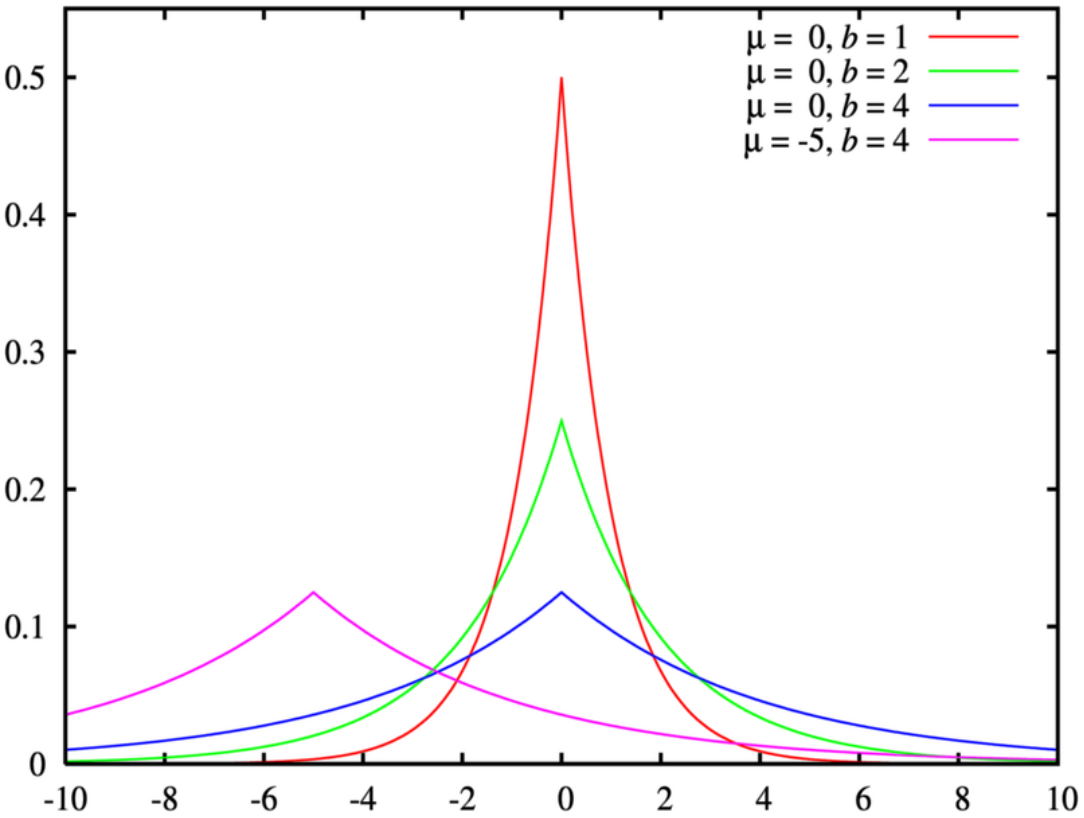

Figura 2.1: Distribución de Laplace, con varios valores para los parámetros μ y b.

Por otro lado, necesitamos medir la escala del ruido según nuestra situación. Para ello usaremos la ℓ_1-sensibilidad:

Definición 2.1.2 *La ℓ_1-**sensibilidad** de una función $f : \mathbb{N}^{|\mathcal{X}|} \to \mathbb{R}^n$, respecto de dos bases de datos adyacentes x e y, es:*

$$\Delta f = \max_{\substack{x,y \in \mathbb{N}^{|\mathcal{X}|} \\ \|x-y\|_1 = 1}} ||f(x) - f(y)||_1$$

Recuérdese que se define la norma-1 $||x - y||_1 = |x_1 - y_1| + ... + |x_k - y_k|$ como uno de los casos particulares de la norma-p[1].

Con esto podemos tener una acotación sobre la forma en la que un cambio en dos bases de datos adyacentes (es decir, de un único sujeto) afecta a los resultados de la función f.

[1]La norma-p de un vector $\mathbf{x}$ se define como $||\mathbf{x}||_p = \sqrt[p]{|x_1|^p + ... + |x_k|^p}$. Haciendo tender p a infinito, se tiene que $||\mathbf{x}||_\infty = \max_{i=1,...,k} |x_i|$.

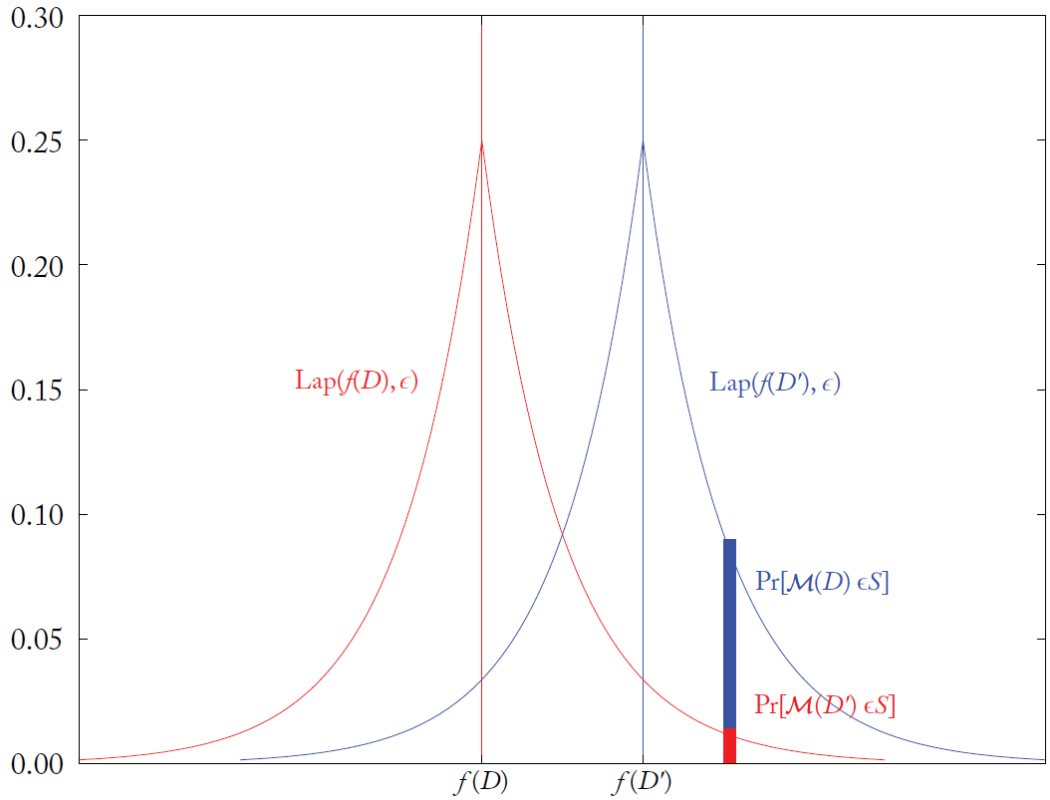

Figura 2.2: Mecanismo de Laplace, extraído de [12].

En estas condiciones ya podemos definir el mecanismo de Laplace. La idea es introducir perturbaciones en los resultados del mecanismo aprovechando las propiedades de la distribución de Laplace, que tendrá una esperanza 0 y una escala en función de la ℓ_1-sensibilidad que deseemos:

Definición 2.1.3 *Dada una función* $f : \mathbb{N}^{|\mathcal{X}|} \to \mathbb{R}^n$ *y una base de datos* x*, el* ***mecanismo de Laplace*** *se define como:*

$$\mathcal{M}_L(x, f(\cdot), \varepsilon) = f(x) + (Y_1, .., Y_n)$$

donde cada Y_i *es una variable aleatoria independiente e idénticamente distribuida a una* $Lap(\Delta f/\varepsilon)$*.*

Nota 2.1.1 *La función de densidad de la distribución* $Lap(\Delta f/\varepsilon)$ *es:*

$$Lap(z|0, \Delta f/\varepsilon) = \frac{\varepsilon}{2\Delta f} exp\left(\frac{-\varepsilon|z|}{\Delta f}\right)$$

Teorema 2.1.2 *El mecanismo de Laplace conserva la ε-DP.*

Demostración: Fijemos $x \in \mathbb{N}^{|\mathcal{X}|}$ y $y \in \mathbb{N}^{|\mathcal{X}|}$ tales que $||x-y||_1 \leq 1$. Sea p_x la función de densidad de $\mathcal{M}_L(x, f, \varepsilon)$ y, análogamente, p_y la función de densidad de $\mathcal{M}_L(y, f, \varepsilon)$. Comparamos las dos en un punto arbitrario z:

$$\frac{p_x(z)}{p_y(z)} = \prod_{i=1}^{n} \frac{\exp\left(-\dfrac{\varepsilon|f(x)_i - z_i|}{\Delta f}\right)}{\exp\left(-\dfrac{\varepsilon|f(y)_i - z_i|}{\Delta f}\right)} = \prod_{i=1}^{n} \exp\left(\frac{\varepsilon(|f(y)_i - z_i| - |f(x)_i - z_i|)}{\Delta f}\right)$$
$$\leq \prod_{i=1}^{n} \exp\left(\frac{\varepsilon|f(x)_i - f(y)_i|}{\Delta f}\right) = \exp\left(\frac{\varepsilon \cdot ||f(x) - f(y)||_1}{\Delta f}\right) \leq e^{\varepsilon}$$

Esto prueba que $p_x(z) \leq e^{\varepsilon} p_y(z)$. Para ver que es análogo para $p_y(x) \leq e^{\varepsilon} p_x(z)$, basta repetir el razonamiento anterior con $\frac{p_x(z)}{p_y(z)} \geq e^{-\varepsilon}$. □

Nota 2.1.3 *Existe un mecanismo similar, llamado* ***mecanismo gaussiano****, que toma una ℓ_2-sensibilidad[2] y un parámetro σ dependiente de ella, para añadir ruido mediante una distribución $\mathcal{N}(0, \sigma)$. Puede verse más en profundidad en el apéndice A de [7]. El mecanismo gaussiano cumple (ε, δ)-DP y tiene un comportamiento similar al de Laplace; sin embargo nos decantamos por el que estudiamos en este tema por ser más simple (y popular) manteniendo resultados casi equivalentes.*

Veremos ahora una cota para calcular la precisión que se puede alcanzar con este mecanismo. Primero necesitamos el siguiente resultado:

Lema 2.1.4 *Sea $Y \sim Lap(b)$. Se cumple, siempre que $t \geq 0$:*

$$Pr\big[|Y| \geq t \cdot b\big] = e^{-t}$$

Demostración: Operando directamente:

$$\Pr\big[|Y| \geq t \cdot b\big] = 1 - \Pr\big[|Y| \leq t \cdot b\big] = 1 - \Pr\big[Y \leq t \cdot b\big] - \Pr\big[-Y \leq t \cdot b\big]$$
$$= 2 - \Pr\big[Y \leq t \cdot b\big] - \Pr\big[Y \leq t \cdot b\big] = 2\Big[1 - \Pr\big[Y \leq t \cdot b\big]\Big]$$

Como $t \geq 0$, $\Pr\big[Y \leq t \cdot b\big] = 1 - \frac{1}{2}e^{-tb/b} = 1 - \frac{1}{2}e^{-t}$, luego

$$\Pr\big[|Y| \geq t \cdot b\big] = 2\Big[1 - \Pr\big[Y \leq t \cdot b\big]\Big] = 2\Big[1 - 1 + \frac{1}{2}e^{-t}\Big] = e^{-t}$$

□

[2] $\Delta_2 f = \max_{\substack{x,y \in \mathbb{N}^{|\mathcal{X}|} \\ \|x-y\|_1 = 1}} ||f(x) - f(y)||_2$

Teorema 2.1.5 *Sean* $f : \mathbb{N}^{|\mathcal{X}|} \to \mathbb{R}^n$, $y = \mathcal{M}_L(x, f(\cdot), \varepsilon)$. *Entonces,* $\forall \alpha \in (0, 1]$*:*

$$\Pr\left[||f(x) - y||_\infty \geq \ln\left(\frac{n}{\alpha}\right) \cdot \left(\frac{\Delta f}{\varepsilon}\right)\right] \leq \alpha$$

Demostración: Con las condiciones anteriores:

$$\begin{aligned}\Pr\left[||f(x) - y||_\infty \geq \ln\left(\frac{n}{\alpha}\right) \cdot \left(\frac{\Delta f}{\varepsilon}\right)\right] &= \Pr\left[\max_{i\in[1..n]} |Y_i| \geq \ln\left(\frac{n}{\alpha}\right) \cdot \left(\frac{\Delta f}{\varepsilon}\right)\right] \\ &\leq n \cdot \Pr\left[|Y_1| \geq \ln\left(\frac{n}{\alpha}\right) \cdot \left(\frac{\Delta f}{\varepsilon}\right)\right] \\ &= n \cdot \frac{\alpha}{n} = \alpha\end{aligned}$$

Donde en la primera igualdad hemos usado la definición de la norma del máximo y las distribuciones $Y \sim Lap(\Delta f/\varepsilon)$ son las que aparecen en el mecanismo de Laplace. En la penúltima igualdad hemos usado el lema 2.1.4, ya que $\ln(n/\alpha) \geq 0$. □

2.2. Aplicaciones

Ser capaces de aplicar un algoritmo que garantice que los resultados cumplan ε-DP y que además es rico en propiedades de composición proporciona muchas ventajas. Veamos una serie de ejemplos sobre problemas reales:

Ejemplo 2.2.1 *Una de las aplicaciones principales del mecanismo de Laplace es la del* ***conteo de peticiones***[3] ***(counting queries)****, es decir, dar una respuesta (numérica) a una pregunta del tipo «¿Cuántos elementos de* X *tienen la propiedad* P*?» Esta formulación es muy sencilla porque sobre una pregunta de conteo, la* ℓ_1*-sensibilidad es* 1*: basta notar que es lo máximo que puede alterar cualquier pregunta el hecho de añadir o eliminar los datos de un solo individuo. Esto también significa que el error introducido, que viene de* $Lap(1/\varepsilon)$*, es independiente del tamaño de la base de datos.*

Si en vez de contar una sola propiedad P lo hacemos con varias, digamos una cantidad m, podemos modelar el problema hacia el caso vectorial teniendo en

[3]Aunque trabajando con funciones del tipo $f : \mathbb{N}^N \to \mathbb{R}^n$, podemos ampliar estas cuestiones a otras relativas al porcentaje de elementos de una base de datos que cumplen alguna propiedad, o hacer combinaciones lineales sobre el conteo de resultados estableciendo un sistema de pesos, por ejemplo.

cuenta que en este caso la ℓ_1-sensibilidad se convierte en m, y la privacidad proviene de una $Lap(m/\varepsilon)$.

Al problema de responder un gran número de preguntas se le llama Problema de Liberar Peticiones (*Query Release Problem*). En el capítulo 5 veremos que el mecanismo de Laplace es una de las herramientas fundamentales para abordarlo.

Ejemplo 2.2.2 (Histogramas) *En el conteo de varias peticiones, si se da el caso de que estas puedan ser disjuntas (como en los histogramas, donde dividimos en rangos los resultados posibles), podemos mejorar la ℓ_1-sensibilidad del problema para que vuelva a ser* 1*, ya que como antes, añadir o eliminar los datos de un individuo solo afecta al rango al que este pertenece.*

Veamos ahora una aplicación del teorema 2.1.5:

Ejemplo 2.2.3 *Supongamos que queremos calcular los nombres más comunes entre algún censo de una población (cuyo tamaño puede llegar a ser todo lo grande que queramos, pongamos 300.000 habitantes), de entre una lista de unos* 10,000*. Esto puede entenderse como una petición (query) de la forma $f : \mathbb{N}^{300,000} \to \mathbb{N}^{10,000}$, que además puede formar un histograma. Por tanto, tenemos que la sensibilidad es $\Delta f = 1$ y podemos alcanzar una $(1, 0)$-DP con una probabilidad de error de un* 5 % *para un ruido de* $\ln(10000/0{,}5) \approx 12{,}2$*. Teniendo en cuenta la cantidad de la población o el número de nombres potenciales, es una cantidad pequeña.*

2.2.1. Informe del máximo con ruido[4]

Veremos ahora un método utilizado para responder a las peticiones de conteo que implican varias propiedades a la vez, donde igual que con los histogramas (donde las propiedades son únicas y disjuntas), lo que buscamos es reducir la sensibilidad del error: aunque tengamos m propiedades (o peticiones, conteos) que nos llevarían a una $Lap(m/\varepsilon)$, si es posible aplicaremos una $Lap(1/\varepsilon)$. Para ello, la idea es aplicar esta $Lap(1/\varepsilon)$ a cada petición y revelar únicamente el nombre (o el índice) de la que haya tenido el resultado más alto.

Observemos que aquí aplicamos el principio de *minimizar la información* que abrimos al público: no decimos ni en qué cantidad se alcanza en el máximo ni damos ninguna información sobre el resto de peticiones, incluso ignoramos el caso de empate en dicho máximo al revelar solo «la etiqueta» de una única propiedad.

[4]En inglés, *report noisy max*.

Teorema 2.2.4 *El informe del máximo con ruido cumple ε-DP.*

DEMOSTRACIÓN: Fijemos $D = D' \cup \{a\}$. Sea c (y respectivamente c') el vector de los conteos sobre D (respectivamente D'). Se cumplen dos propiedades:

$$\begin{aligned} c_j &\geq c'_j \qquad \forall j = 1, ..., m \\ 1 + c'_j &\geq c_j \qquad \forall j = 1, ..., m \end{aligned}$$

Vamos a fijar un índice $i \in [1..m]$ y acotar superior e inferiormente las probabilidades con lo que lo escogeremos.

Notemos por r_{-i} al resultado del experimento aleatorio de una distribución $[Lap(1/\varepsilon)]^{m-1}$, usado para añadir ruido a todos los componentes de c menos a la coordenada i-ésima. También denotaremos $\Pr[i|\xi]$ a la probabilidad de que el resultado del método del informe máximo sea i condicionada a ξ. Argumentamos para cada r_{-i} independientemente en dos partes: primero veremos que $\Pr[i|D, r_{-i}] \leq e^{\varepsilon} \Pr[i|D', r_{-i}]$ y luego que $\Pr[i|D', r_{-i}] \leq e^{\varepsilon} \Pr[i|D, r_{-i}]$.

En el primer caso, definimos:

$$r^* = \min_{r_i} : c_i + r_i > c_j + r_j \ \forall j \neq i.$$

Para un r_{-i} fijo, el resultado que devuelve el método será i (cuando la base de datos es D) si y solo si $r_i \geq r^*$. Tenemos, con $1 \leq j \neq i \leq m$:

$$\begin{aligned} c_i + r^* &> c_j + r_j \\ \Rightarrow (1 + c'_i) + r^* \geq c_i + r^* &> c_j + r_j \geq c'_j + r_j \\ \Rightarrow c'_i + (r^* + 1) &> c'_j + r_j \end{aligned}$$

Esto es, si $r_i \geq r^* + 1$, entonces el máximo estará en la i-ésima coordenada cuando la base de datos sea D' y el vector de ruido sea (r_i, r_{-i}). Tomando cada $r_i \sim Lap(1/\varepsilon)$:

$$\begin{aligned} \Pr[r_i \geq 1 + r^*] \geq e^{-\varepsilon} \Pr[r_i \geq r^*] = e^{-\varepsilon} \Pr[i|D, r^{-i}] \Rightarrow \\ \Pr[i|D', r^{-i}] \geq \Pr[r_i \geq 1 + r^*] \geq e^{-\varepsilon} \Pr[i|D, r^{-i}] \Rightarrow \\ \Pr[i|D, r_{-i}] \leq e^{\varepsilon} \Pr[i|D', r_{-i}] \end{aligned}$$

Análogamente, escogiendo:

$$r^* = \min_{r_i} : c'_i + r_i > c'_j + r_j \ \forall j \neq i.$$

Y como antes, para un r_{-i} fijo, el resultado que devuelve el método será i (cuando la base de datos es D') si y solo si $r_i \geq r^*$. Para $1 \leq j \neq i \leq m$:

$$\begin{aligned} c'_i + r^* &> c'_j + r_j \\ \Rightarrow c'_i + (r^* + 1) &> (1 + c'_j) + r_j \\ \Rightarrow c_i + (r^* + 1) \geq c'_i + (r^* + 1) &> (1 + c'_j) + r_j \geq c_j + r_j \end{aligned}$$

De donde se sigue el mismo razonamiento anterior: si $r_i \geq r^* + 1$, el máximo estará en la i-ésima coordenada en la base de datos sea D con el vector de ruido (r_i, r_{-i}). Por último,

$$\begin{aligned} \Pr[i|D, r^{-i}] \geq \Pr[r_i \geq 1 + r^*] \geq \Pr[r_i \geq r^*] = e^{-\varepsilon} \Pr[i|D', r^{-i}] \\ \Rightarrow e^{\varepsilon} \Pr[i|D', r_{-i}] \leq \Pr[i|D, r_{-i}] \end{aligned}$$

□

Ejemplo 2.2.5 *Supongamos que queremos saber cuál es la enfermedad más común que ha padecido algún conjunto de personas. Para ello debemos saber qué historial médico ha tenido cada individuo, lo que implica una gran cantidad de peticiones; en este caso, por cuántas enfermedades («propiedades») ha pasado cada paciente. Esto dispararía la sensibilidad de la función que aplicaríamos en caso de usar directamente el mecanismo de Laplace. Sin embargo, el informe del máximo con ruido proporciona exactamente lo que queremos reduciendo notablemente la variación necesaria del ruido.*

Capítulo 3

Mecanismo exponencial

El mecanismo exponencial tiene dos principales diferencias con el de Laplace. Por un lado podemos ampliar el rango de los resultados posibles (no tenemos que restringirnos a funciones $f : \mathbb{N}^{|\mathcal{X}|} \to \mathbb{R}^n$). Por otro, está diseñado para casos donde una mínima cantidad de ruido en la respuesta puede desestabilizar la utilidad de los datos.

Para asegurar la privacidad en este último caso usaremos una función u (de utilidad) que, intuitivamente, asignará a cada respuesta de la base de datos algún número que represente su valor, y entonces operaremos con ellos. Será lo que veremos en este capítulo.

3.1. Introducción

Dado un rango arbitrario $\mathcal{R}$, el mecanismo exponencial se define respecto a una función de utilidad $u : \mathbb{N}^{|\mathcal{X}|} \times \mathcal{R} \to \mathbb{R}$, donde normalmente buscamos obtener el elemento $r \in \mathcal{R}$ que maximice el resultado de esa función. Ponemos a continuación un caso particular (extraído de [7]) con estas condiciones:

Ejemplo 3.1.1 *Supongamos que tenemos una subasta de calabazas y cuatro pujantes por ellas. El primero ofrece* $3,01$\$ *y los otros tres,* 1\$ *cada uno. ¿Cuál es el precio óptimo? Poniéndolas a* 3\$ *o a* 1\$*, el beneficio sería de* 3\$*. Si ponemos* $3,01$\$*, ganaríamos* $3,01$\$*. Pero con* $3,02$\$ *como precio, la ganancia sería nula.*

Veamos los primeros pasos para añadir privacidad diferencial a este tipo de problemas.

Definición 3.1.1 *Llamamos* ***sensibilidad de la función de utilidad*** $u : \mathbb{N}^{|\mathcal{X}|} \times \mathcal{R} \to \mathbb{R}$ *a la cantidad:*

$$\Delta u = \max_{r \in \mathcal{R}} \max_{\substack{x,y \in \mathbb{N}^{|\mathcal{X}|} \\ \|x-y\|_1 = 1}} |u(x,r) - u(y,r)|$$

Es importante tener en cuenta que esta sensibilidad de u es respecto del argumento r que le puede llegar de la base de datos, fuera de ese rango puede tener cualquier comportamiento arbitrario.

Definición 3.1.2 *El* ***mecanismo exponencial*** $\mathcal{M}_E(x, u, \mathcal{R})$ *se basa en escoger (y devolver) un elemento* $r \in \mathcal{R}$ *con probabilidad proporcional a*

$$\exp\left(\frac{\varepsilon u(x,r)}{2\Delta u}\right)$$

Ejemplo 3.1.2 *Siguiendo el ejemplo 3.1.1, la utilidad de un precio p dada una base de datos D es el beneficio obtenido según refleje la curva de demanda que se desarrolle. Habrá una posibilidad exponencialmente más grande frente al resto de precios de escoger el óptimo.*

Observación 3.1.3 *La intuición es que la probabilidad con que debería funcionar el mecanismo exponencial fuera proporcional a* $\exp(\varepsilon u(x,r)/\Delta u)$*. De esta forma tendríamos que la pérdida de privacidad sería:*

$$\mathcal{L}_{\mathcal{M}_{E(x,u,\mathcal{R})} || \mathcal{M}_{E(y,u,\mathcal{R})}} = \ln\left(\frac{\exp(\varepsilon u(x,r)/\Delta u)}{\exp(\varepsilon u(y,r)/\Delta u)}\right) = \frac{\varepsilon[u(x,r) - u(y,r)]}{\Delta u} \leq \varepsilon$$

Sin embargo, se puede dar el caso de que al añadir (o eliminar) los datos de algún individuo de la base de datos, la utilidad de algunos elementos $r \in \mathcal{R}$ *se incremente o se vea perjudicada. Por este motivo se reserva la mitad del presupuesto de privacidad, como se ve en la definición.*

El mecanismo exponencial puede llegar a definir una distribución compleja sobre un dominio arbitrariamente grande, de modo que su puesta en práctica puede no ser eficiente, especialmente si el rango de u es super-polinómico[1] respecto a los parámetros naturales del problema.

Teorema 3.1.4 *El mecanismo exponencial cumple* ε*-DP.*

[1]Es un concepto que se usa para hacer referencia a la complejidad en tiempo (*time complexity*), es decir, la medición o estimación el tiempo de ejecución de un algoritmo. Decimos que un algoritmo es de tiempo super-polinómico si no puede ser acotado superiormente por ningún polinomio. Esto sucede, por ejemplo, para funciones que son de tipo exponencial (como, digamos, 2^n).

DEMOSTRACIÓN: Vamos a asumir por comodidad que el rango $\mathcal{R}$ es finito, aunque no es necesario. Para dos bases de datos adyacentes cualesquiera x e y ($||x-y||_1 \leq 1$) y algún $r \in \mathcal{R}$ fijo:

$$\begin{aligned}
\frac{\Pr[\mathcal{M}_E(x,u,\mathcal{R})=r]}{\Pr[\mathcal{M}_E(y,u,\mathcal{R})=r]} &= \frac{\left(\frac{\exp\left[\varepsilon u(x,r)/(2\Delta u)\right]}{\sum_{r'\in\mathcal{R}}\exp\left[\varepsilon u(x,r')/(2\Delta u)\right]}\right)}{\left(\frac{\exp\left[\varepsilon u(y,r)/(2\Delta u)\right]}{\sum_{r'\in\mathcal{R}}\exp\left[\varepsilon u(y,r')/(2\Delta u)\right]}\right)} \\
&= \frac{\exp\left(\frac{\varepsilon u(x,r)}{2\Delta u}\right)}{\exp\left(\frac{\varepsilon u(y,r)}{2\Delta u}\right)} \cdot \frac{\sum_{r'\in\mathcal{R}}\exp\left(\frac{\varepsilon u(y,r')}{2\Delta u}\right)}{\sum_{r'\in\mathcal{R}}\exp\left(\frac{\varepsilon u(x,r')}{2\Delta u}\right)} \\
&= \exp\left(\frac{\varepsilon[u(x,r)-u(y,r)]}{2\Delta u}\right) \cdot \frac{\sum_{r'\in\mathcal{R}}\exp\left(\frac{\varepsilon u(y,r')}{2\Delta u}\right)}{\sum_{r'\in\mathcal{R}}\exp\left(\frac{\varepsilon u(x,r')}{2\Delta u}\right)} \\
&\leq \exp\left(\frac{\varepsilon}{2}\right) \cdot \exp\left(\frac{\varepsilon}{2}\right) \cdot \frac{\sum_{r'\in\mathcal{R}}\exp\left(\frac{\varepsilon u(x,r')}{2\Delta u}\right)}{\sum_{r'\in\mathcal{R}}\exp\left(\frac{\varepsilon u(x,r')}{2\Delta u}\right)} = e^{\varepsilon}
\end{aligned} \tag{3.1}$$

En la última desigualdad hemos usado que, por la simetría de las bases de datos adyacentes:

$$\exp\left(\frac{\varepsilon u(y,r')}{2\Delta u}\right) \leq \exp\left(\frac{\varepsilon}{2}\right)\left(\frac{\varepsilon u(x,r')}{\Delta u}\right) \tag{3.2}$$

Para ver que $\Pr[\mathcal{M}_E(y,u,\mathcal{R})=r] \leq e^{\varepsilon}\Pr[\mathcal{M}_E(x,u,\mathcal{R})=r]$, basta repetir el mismo proceso con:

$$\frac{\Pr[\mathcal{M}_E(x,u,\mathcal{R})=r]}{\Pr[\mathcal{M}_E(y,u,\mathcal{R})=r]} \geq e^{-\varepsilon}$$

□

Una de las garantías de este mecanismo es que proporciona respuestas cercanas a la óptima, ya que descarta con una probabilidad exponencialmente alta a las que se van alejando de ese resultado.

Definición 3.1.3 *Dada una base de datos x y una función de utilidad u, llamaremos* ***valor máximo*** *de u en x a:*

$$OPT_u(x) = \max_{r\in\mathcal{R}} u(x,r)$$

Vamos a acotar la probabilidad con la que el mecanismo exponencial devuelve un «buen» elemento de $\mathcal{R}$, y para ello tendremos que hacerlo en relación a $OPT_u(x)$. El resultado será que la probabilidad de devolver un elemento con una utilidad menor que la del valor máximo va a ser del orden de $(\Delta u/\varepsilon)\ln|\mathcal{R}|$.

Teorema 3.1.5 *Sea una base de datos $x \in \mathbb{N}^{|\mathcal{X}|}$, y*

$$\mathcal{R}_{OPT} = \{r \in \mathcal{R} : u(x,r) = OPT_u(x)\}$$

Entonces:

$$\Pr\left[u\big(x, \mathcal{M}_E(x,u,\mathcal{R})\big) \leq OPT_u(x) - \frac{2\Delta u}{\varepsilon}\left(\ln\left(\frac{|\mathcal{R}|}{|\mathcal{R}_{OPT}|}\right) + t\right)\right] \leq e^{-t}$$

Demostración:

$$\begin{aligned}\Pr\left[u\big(x, \mathcal{M}_E(x,u,\mathcal{R})\big) \leq c\right] &\leq \frac{|\mathcal{R}|\exp\left[\varepsilon c/(2\Delta u)\right]}{|\mathcal{R}_{OPT}|\exp\left[\varepsilon OPT_u(x)/(2\Delta u)\right]} \\ &= \frac{|\mathcal{R}|}{|\mathcal{R}_{OPT}|}\exp\left(\frac{\varepsilon(c - OPT_u(x))}{2\Delta u}\right)\end{aligned}$$

La desigualdad se explica como sigue: cada $r \in \mathcal{R}$ con $u(x,r) \leq c$ tiene, mediante el mecanismo exponencial, un valor en su función de probabilidad de como mucho $\exp[\varepsilon c/(2\Delta u)]$, y por tanto podemos acotar el conjunto completo por $|\mathcal{R}|\exp[\varepsilon c/(2\Delta u)]$. Además, como hay al menos $|\mathcal{R}_{OPT}| \geq 1$ elementos con $u(x,r) = OPT_u(x)$, la probabilidad del conjunto total (en el denominador) es de $|\mathcal{R}_{OPT}|\exp[(\varepsilon OPT_u(x)/(2\Delta u)]$.

El resultado del teorema procede de usar este razonamiento escogiendo el c que se muestra en el enunciado. □

Como siempre se cumple que $|\mathcal{R}_{OPT}| \geq 1$, podemos usar directamente este corolario:

Corolario 3.1.6 *Para cualquier base de datos $x \in \mathbb{N}^{|\mathcal{X}|}$:*

$$\Pr\left[u\big(x, \mathcal{M}_E(x,u,\mathcal{R})\big) \leq OPT_u(x) - \frac{2\Delta u}{\varepsilon}\big(\ln\big(|\mathcal{R}|\big) + t\big)\right] \leq e^{-t}$$

Veamos una aplicación directa:

Ejemplo 3.1.7 *Supongamos que queremos saber qué condición médica es más común de entre dos posibilidades, A y B. Tenemos* 0 *casos de A y $c > 0$ para B. La noción de utilidad se basa en este caso (como en el mecanismo de Laplace)*

en el conteo de los datos: a más casos pertenecientes a alguna condición, más utilidad, y $\Delta u = 1$. *Esto implica que la utilidad de* A *es* 0 *y la de* B *es* c. *Aplicando el corolario 3.1.6 concluimos que la probabilidad de devolver como resultado (erróneo) la condición médica* A *está acotada por* $2e^{-c(\varepsilon/(2\Delta u))} = 2e^{-c\varepsilon/2}$. *Para ello estudiamos la probabilidad de que la utilidad del resultado del mecanismo sea menor o igual a* 0, *es decir,*

$$OPT_u(x) - \frac{2\Delta u}{\varepsilon}\big(\ln\big(|\mathcal{R}|\big) + t\big) = 0$$

y sustituyendo con los datos del problema:

$$c - \frac{2}{\varepsilon}(\ln 2 + t) = 0$$

de donde despejamos la variable t *de la función exponencial que acota a la probabilidad. Es decir,*

$$t = \frac{c\varepsilon}{2} - \ln 2$$

Observación 3.1.8 *Aplicando el mecanismo exponencial, hemos visto que la probabilidad de que salga un resultado de baja utilidad es exponencialmente pequeña. Ahora bien, si el número de posibles resultados potenciales es exponencialmente grande comparado al número de respuestas con buena utilidad, la precisión del mecanismo puede verse reducida.*

3.2. Funciones monótonas

Existen casos en los que si se dan las condiciones indicadas, se puede mejorar la precisión del mecanismo exponencial.

Definición 3.2.1 *Una función de utilidad* u *es* ***monótona*** *si para dos bases de datos adyacentes cualesquiera* x *e* y, *ocurre que:*

$$u(x, r) \geq u(y, r) \qquad \forall r \in \mathcal{R}$$

o bien,

$$u(x, r) \leq u(y, r) \qquad \forall r \in \mathcal{R}$$

En los problemas de conteo, por ejemplo, la función de utilidad es monótona.

Proposición 3.2.1 *El mecanismo exponencial* $\mathcal{M}_E(x, u, \mathcal{R})$, *aplicado en funciones de utilidad monótonas, puede mejorar el presupuesto de privacidad devolviendo una respuesta con probabilidad proporcional a* $\exp\left(\dfrac{\varepsilon u(x,r)}{\Delta u}\right)$, *en lugar de* $\exp\left(\dfrac{\varepsilon u(x,r)}{2\Delta u}\right)$, *manteniendo la* ε*-DP.*

Demostración: Es análoga a la demostración del teorema 3.1.4. La ecuación 3.1 pasa a ser:

$$\frac{\Pr[\mathcal{M}_E(x,u,\mathcal{R})=r]}{\Pr[\mathcal{M}_E(y,u,\mathcal{R})=r]} = \frac{\left(\frac{\exp(\varepsilon u(x,r)/\Delta u)}{\sum_{r'\in\mathcal{R}}\exp(\varepsilon u(x,r')/\Delta u)}\right)}{\left(\frac{\exp(\varepsilon u(y,r)/\Delta u)}{\sum_{r'\in\mathcal{R}}\exp(\varepsilon u(y,r')/\Delta u)}\right)}$$
$$= \frac{\exp\left(\frac{\varepsilon u(x,r)}{\Delta u}\right)}{\exp\left(\frac{\varepsilon u(y,r)}{\Delta u}\right)} \cdot \frac{\sum_{r'\in\mathcal{R}}\exp\left(\frac{\varepsilon u(y,r')}{\Delta u}\right)}{\sum_{r'\in\mathcal{R}}\exp\left(\frac{\varepsilon u(x,r')}{\Delta u}\right)}$$
$$\leq e^{\varepsilon}$$

La última desigualdad se da porque al ser u monótona, se puede ver que en el paso anterior, cuando el primer término crece, el segundo decrece (o al contrario), con lo que se puede hacer la acotación sobre el primero:

$$\frac{\exp\left(\frac{\varepsilon u(x,r)}{\Delta u}\right)}{\exp\left(\frac{\varepsilon u(y,r)}{\Delta u}\right)} = \exp\left(\frac{\varepsilon[u(x,r)-u(y,r)]}{\Delta u}\right) \leq e^{\varepsilon}$$

O sobre el segundo, como en (3.2):

$$\exp\left(\frac{\varepsilon u(y,r')}{\Delta u}\right) \leq e^{\varepsilon}\exp\left(\frac{\varepsilon u(x,r')}{\Delta u}\right)$$

□

Corolario 3.2.2 *Para cualquier base de datos $x \in \mathbb{N}^{|\mathcal{X}|}$, con u monótona:*

$$\Pr\left[u\big(x,\mathcal{M}_E(x,u,\mathcal{R})\big) \leq OPT_u(x) - \frac{\Delta u}{\varepsilon}\left(\ln\left(|\mathcal{R}|\right)+t\right)\right] \leq e^{-t}$$

Una de las consecuencias se puede ver en el ejemplo 3.1.7: cambiando adecuadamente el ruido que se introduce con el mecanismo exponencial, la probabilidad de observar una respuesta errónea pasa a ser $2\exp(-c\varepsilon)$, que mejora la cota que teníamos.

Definición 3.2.2 *Llamamos **informe del argumento máximo con ruido unilateral** (Report One-Sided Noisy Arg-Max) al algoritmo que consiste en añadir ruido a la utilidad de cada respuesta con el mecanismo exponencial, con parámetro $\varepsilon/\Delta u$ si la función de utilidad es monótona, o con parámetro $\varepsilon/(2\Delta u)$ en caso contrario, y devolver el índice de la categoría que haya tenido el resultado más alto.*

Este algoritmo es semejante al del informe del máximo con ruido, estudiado en 2.2.1. Podríamos compararlos en el caso de que $\Delta f = \Delta u = 1$, con función monótona o no, pero la preferencias sobre su uso dependen de parámetros más concretos de cada problema.

3.3. Relación con el mecanismo de Laplace

En [13] se argumenta que el mecanismo de Laplace se puede considerar como un caso particular del mecanismo exponencial, usando una función de utilidad adecuada. Incluso que, técnicamente, afinando con el valor de la función de utilidad, el mecanismo exponencial podría «capturar a la clase completa de los mecanismos de la DP».

Teorema 3.3.1 *Dada una función $f : \mathbb{N}^{|\mathcal{X}|} \to \mathbb{R}^n$, tomando*

$$\begin{aligned} u : \mathbb{N}^{|\mathcal{X}|} \times \mathcal{R} &\to \mathbb{R} \\ u(x, r) &\longmapsto - |f(x) - r| \end{aligned}$$

la sensibilidad de u es igual a la ℓ_1-sensibilidad de f.

Demostración:

$$\begin{aligned} \Delta u &= \max_{r \in \mathcal{R}} \max_{\substack{x,y \in \mathbb{N}^{|\mathcal{X}|} \\ \|x-y\|_1 = 1}} |u(x, r) - u(y, r)| \\ &= \max_{r \in \mathcal{R}} \max_{\substack{x,y \in \mathbb{N}^{|\mathcal{X}|} \\ \|x-y\|_1 = 1}} ||f(y) - r| - |f(x) - r|| \\ &= \max_{\substack{x,y \in \mathbb{N}^{|\mathcal{X}|} \\ \|x-y\|_1 = 1}} ||f(x) - f(y)||_1 = \Delta f \end{aligned}$$

□

Sin embargo, en [12] se contraargumenta esta tesis: no tenemos garantía de que la función de utilidad resultante sea monótona, y por tanto el mecanismo exponencial devuelve un resultado r con probabilidad proporcional a $\exp\left(\frac{-\varepsilon|f(x) - r|}{2\Delta f}\right)$, en lugar de lo que resultaría de usar el mecanismo de Laplace, esto es, una probabilidad proporcional a $\exp\left(\frac{-\varepsilon|f(x) - r|}{\Delta f}\right)$.

Observación 3.3.2 *Del mismo modo, dado un mecanismo cualquiera $\mathcal{M}$ que cumpla ε-DP, se puede escoger $u(x, r)$ como el logaritmo de la función de densi-*

dad de $\mathcal{M}$ *en el punto* r*. Como antes, nos encontramos con un problema: el mecanismo exponencial devuelve* r *con probabilidad proporcional a* $\exp\left(\frac{u(x,r)}{2}\right)$*, no a* $\exp\left(u(x,r)\right)$*.*

Capítulo 4

Casos prácticos

En este capítulo vamos a ver varias formas prácticas de aplicar la DP satisfactoriamente, así como algunas complicaciones que se pueden dar. Veremos que algún cambio en la implementación del algoritmo puede comprometer la privacidad, y compararemos algunos acercamientos distintos a un mismo problema.

El caso que trataremos será el de hallar el valor de la media de un atributo sobre un subconjunto de los datos totales que cumpla alguna condición, o pertenezca a alguna categoría. Nos referimos a responder preguntas de la forma «¿Cuántos paquetes de cigarros fuman, de media, lo sujetos de la base de datos que llevan fumando más de 15 años?», o «¿cuántas ventas producen, de media, los productos de una marca que llevan más de 6 meses en el mercado?»

Formalmente, tenemos la base de datos D, una condición de selección φ, un conjunto $\sigma_\varphi(D)$ que representa a todos los elementos de D que cumplen φ, un atributo a que toma valores entre $[a_{min}, a_{max}]$. Sea también $\Delta_a = a_{max} - a_{min}$. y x_a el valor (en el atributo a) del elemento $x \in D$. Vamos a estudiar la función:

$$Avg^a_\varphi(D) = \begin{cases} \dfrac{\sum_{x \in \sigma_\varphi(D)} x_a}{|\sigma_\varphi(D)|} & \text{si } \sigma_\varphi(D) \neq \emptyset \\ \dfrac{a_{min} + a_{max}}{2} & \text{si } \sigma_\varphi(D) = \emptyset \end{cases}$$

Que devuelve la media de los valores x_a de los elementos x que cumplen φ en el caso de que haya más de un elemento que cumpla dicha condición en D, o la media de los valores máximo y mínimo del atributo en caso contrario.

4.1. Procedimiento general

Veamos cómo sería el primer acercamiento natural, aplicando la teoría que hemos visto en los capítulos anteriores.

4.1.1. Mecanismo exponencial

Primero necesitamos seleccionar la función de utilidad que nos resulte más conveniente. Lo usual en este caso es elegir:

$$u(D, r) = -|\sum_{x \in \sigma_\varphi(D)} (x_a - r)|$$

Esto es, la utilidad de la media «verdadera» ($Avg^a_\varphi = \mu$) es 0 y para cualquier otro valor r, tendremos como resultado $-|\sigma_\varphi(D)| \cdot |r - \mu|$. La sensibilidad de la función u es $\Delta_a = a_{max} - a_{min}$, ya que en el peor de los casos ($r = a_{min}$) añadir un nuevo valor a_{max} cambiaría la utilidad en Δ_a (o simétricamente, intercambiando a_{min} y a_{max}).

Destacamos que que esta función no es monótona. Si $|\sigma_\varphi(D)|$ es grande, cualquier valor que se aleje de la media tendrá una probabilidad exponencialmente pequeña de ser seleccionado.

4.1.2. Mecanismo de Laplace

La primera opción es aplicar un ruido proporcional a la sensibilidad global de la función de la media, es decir, $\frac{|a_{max} - a_{min}|}{2}$, que es demasiado grande. Una aplicación directa en este sentido tiene poca precisión, al añadir ruido suficiente para los peores casos, donde hay pocos puntos con esas condiciones. Esto es, la respuesta del mecanismo se puede ver muy alterada añadiendo o retirando el valor de un único elemento.

4.2. Mecanismo de Laplace y composición

En este punto veremos cómo mejorar las aplicaciones prácticas del mecanismo de Laplace a través de las propiedades de composición de mecanismos que vimos en el capítulo 1. Por simplificación, tomaremos la base de datos D, sobre la que actúa el algoritmo, como el conjunto de los atributos $a \in \sigma_\varphi(D)$ de todos

los elementos que cumplen la condición φ. Además tendremos en cuenta otras dos funciones: una de *Suma* que devuelve la suma de los valores de los $a \in D$, y otra, *Conteo*, que cuenta el número de elementos a. Sobre estas dos *queries* (peticiones a la base de datos D que estamos utilizando) tendremos que dividir el presupuesto de privacidad ε.

Algoritmo 1: Media con ruido, Laplace simple

Datos: D, $[a_{min}, a_{max}]$, ε.
$\widetilde{S} \leftarrow D.Suma + Lap(2(a_{max} - a_{min})/\varepsilon)$
$\widetilde{C} \leftarrow D.Conteo + Lap(2/\varepsilon)$
si $\widetilde{C} \leq 1$ **entonces**
  devolver $\dfrac{a_{min} + a_{max}}{2}$
en otro caso
  devolver $\dfrac{\widetilde{S}}{\widetilde{C}}$
fin

Para ver que el algoritmo 1 cumple ε-DP, observemos que tanto $\widetilde{S}$ como $\widetilde{C}$ cumplen $(\varepsilon/2)$-DP, y en los datos que devolvemos aplicamos la composición secuencial de la proposición 1.2.2.

4.2.1. Un algoritmo no privado

Podemos intentar mejorar el algoritmo 1: antes añadimos ruido a la función de *conteo*, pero ¿es necesario, si solo queremos devolver con ruido el valor de la media? Consideremos la siguiente función (*Noisy Average*):

$$NAvg^a_\varphi(D) = \begin{cases} \dfrac{\left(\sum_{x \in \sigma_\varphi(D)} x_a\right) + Lap\left(\Delta_a/\varepsilon\right)}{|\sigma_\varphi(D)|} & \text{si } \sigma_\varphi(D) \neq \emptyset \\ U([a_{min}, a_{max}]) & \text{si } \sigma_\varphi(D) = \emptyset \end{cases}$$

Con $U([c,d])$ la distribución uniforme.

Teorema 4.2.1 *El mecanismo que usa la función $NAvg^a_\varphi(D)$ no cumple ε-DP.*

Demostración: Sean D y D' dos bases de datos adyacentes, e Y una variable aleatoria que sigue la función de distribución de $Lap\left(\Delta_a/\varepsilon\right)$. Sin pérdida de generalidad, podemos asumir que $\sigma_\varphi(D)$ contiene un elemento más que $\sigma_\varphi(D')$, y

que este elemento tiene un valor a'. Sea entonces $|\sigma_\varphi(D')| = n$ y $|\sigma_\varphi(D)| = n+1$. Para cualquier z en el rango del mecanismo:

$$\begin{aligned}\frac{\Pr\left[NAvg^a_\varphi(D) = z\right]}{\Pr\left[NAvg^a_\varphi(D') = z\right]} &= \frac{\Pr\left[\left(Y + \sum_{x\in\sigma_\varphi(D)} x_a\right)/(n+1) = z\right]}{\Pr\left[\left(Y + \sum_{x\in\sigma_\varphi(D')} x_a\right)/n = z\right]} \\ &= \frac{\Pr\left[Y = zn - \left(+\sum_{x\in\sigma_\varphi(D')} x_a\right) + z - a'\right]}{\Pr\left[Y = zn - \left(+\sum_{x\in\sigma_\varphi(D')} x_a\right)\right]} \\ &= e^{z-a'}\end{aligned}$$

Como z no está acotado, $z - a'$ puede ser arbitrariamente grande. □

Sin embargo, podemos restringir el rango de la respuesta del mecanismo. Esto es, imponiendo que el resultado pertenezca al intervalo $[a_{min}, a_{max}]$, podemos asegurar que $|z - a'| \leq \Delta_a$ y sí podríamos aproximarnos a la ε-DP.

Algoritmo 2: Media con ruido, re-muestreo (*Noisy Average Resampling*)

Datos: D, $[a_{min}, a_{max}]$, ε.
$S \leftarrow D.Suma$
$C \leftarrow D.Conteo$
si $C = 1$ **entonces**
  devolver $U([a_{min}, a_{max}])$
fin
$A \leftarrow \dfrac{S + Lap((a_{max} - a_{min})/\varepsilon)}{C}$
mientras $A < a_{min}$ ***o*** $A > a_{max}$ **hacer**
  $A \leftarrow \dfrac{S + Lap((a_{max} - a_{min})/\varepsilon)}{C}$
fin
devolver A

En este algoritmo utilizamos la función de *conteo* sin ruido, y repetimos la distorsión al resultado que le da el mecanismo de Laplace hasta que cae en el rango que buscamos. Esto, por otro lado, tiene el efecto de escalar las probabilidades para cada valor dentro de $[a_{min}, a_{max}]$. Cuando dos bases de datos adyacentes tienen una media distinta, este factor de escala se ve alterado, afectando a la ε-DP.

Teorema 4.2.2 *El mecanismo que resulta de aplicar el algoritmo 2 no cumple ε-DP.*

DEMOSTRACIÓN: Vamos a hacerlo con un contraejemeplo. Consideremos $D = \{-1\}$, $D' = \{-1{,}1\}$ y $[a_{min}, a_{max}] = [-1, 1]$. Utilizamos, por tanto, $Y \sim Lap(2/\varepsilon)$, y llamaremos $\mathcal{M}$ al mecanismo del algoritmo 2.

Para la base de datos D, la media (verdadera) es -1. Este resultado se obtendrá cuando $Y = 0$, y el recorrido que admitimos para los resultados de Y, para que la respuesta no salga de $[a_{min}, a_{max}] = [-1, 1]$, debe ser $[0, 2]$. Con esto en cuenta:

$$\Pr\Big[\mathcal{M}(D) = -1\Big] = \Pr\Big[Y = 0 | Y \in [0,2]\Big] = \frac{\frac{\varepsilon}{4}\exp\left(-\frac{\varepsilon}{2}|0|\right)}{\int_0^2 \frac{\varepsilon}{4}\exp\left(-\frac{\varepsilon}{2}|t|\right)dt} =$$

$$= \frac{\frac{\varepsilon}{4}}{\frac{1-\exp(-\varepsilon)}{2}} = \frac{\varepsilon}{2(1-\exp(-\varepsilon))}$$

Para D', la media es 0 y el ruido añadido es $\frac{Y}{2}$, ya que aquí la función *conteo* vale 2. Por tanto, tendremos el resultado -1 cuando $Y = -2$, y ahora Y recorre el intervalo $[-2, 2]$:

$$\Pr\Big[\mathcal{M}(D') = -1\Big] = \Pr\Big[Y = -2 | Y \in [-2,2]\Big] = \frac{\frac{\varepsilon}{4}\exp\left(-\frac{\varepsilon}{2}|-2|\right)}{\int_{-2}^2 \frac{\varepsilon}{4}\exp\left(-\frac{\varepsilon}{2}|t|\right)dt} =$$

$$= \frac{\frac{\varepsilon}{4}\exp(-\varepsilon)}{1-\exp(-\varepsilon)} = \frac{\varepsilon\exp(-\varepsilon)}{4(1-\exp(-\varepsilon))}$$

De modo que:

$$\frac{\Pr\Big[\mathcal{M}(D) = -1\Big]}{\Pr\Big[\mathcal{M}(D') = -1\Big]} = \frac{\frac{\varepsilon}{2(1-\exp(-\varepsilon))}}{\frac{\varepsilon\exp(-\varepsilon)}{4(1-\exp(-\varepsilon))}} = 2e^{\varepsilon} > e^{\varepsilon}$$

□

4.2.2. Un algoritmo privado

La pérdida de las propiedades de la privacidad se deben al método usado en el algoritmo 2 para ajustar el resultado dentro del rango que damos por

válido. En vez de repetir la muestra aleatoria de la distribución de Laplace (remuestreo) hasta que caiga donde queramos, podemos aplicar una función que haga de limitación:

$$\Pi_{[a_{min},a_{max}]}(z) = \begin{cases} a_{min} & \text{si } z < a_{min} \\ z & \text{si } a_{min} \leq z \leq a_{max} \\ a_{max} & \text{si } z > a_{max} \end{cases}$$

Esto hace que la distribución del mecanismo sea discreta en los puntos a_{min} y a_{max}, y continua entre ellos. Podemos asegurar la ε-DP mediante esta función, añadiendo unos cambios convenientes em el algoritmo:

Algoritmo 3: Media con ruido, restringida (*Noisy Average Clamping-Down*)

Datos: D, $[a_{min}, a_{max}]$, ε.
$S \leftarrow D.Suma$
$C \leftarrow D.Conteo$
si $C = 0$ **entonces**
  devolver $\Psi([a_{min}, a_{max}], \varepsilon)$
fin
$A \leftarrow \dfrac{S + Lap((a_{max} - a_{min})/\varepsilon)}{C}$
si $A < a_{min}$ **entonces**
  devolver a_{min}
si no, si $A > a_{max}$ **entonces**
  devolver a_{maz}
en otro caso
  devolver A
fin

Donde

$$\Psi([a_{min}, a_{max}], \varepsilon) = \begin{cases} a_{min} & \text{con probabilidad } (e^{-\varepsilon/2})/2 \\ a_{max} & \text{con probabilidad } (e^{-\varepsilon/2})/2 \\ U([a_{min}, a_{max}]) & \text{con probabilidad } 1 - e^{-\varepsilon/2} \end{cases}$$

Teorema 4.2.3 *El mecanismo que funciona aplicando el algoritmo 3 satisface ε-DP.*

La demostración se basa en analizar los casos posibles, y no añade ninguna idea que no se haya visto en las páginas anteriores, además de ser algo extensa. Puede verse completa en [12].

4.2.3. Normalización

Lo que hemos conseguido con el algoritmo 3 es la mejora que pretendíamos sobre el algoritmo 1 al no tener que introducir ruido en la función de *conteo*, de modo que podemos usar el presupuesto de privacidad ε completo en vez de tener que dividirlo en dos.

Sin embargo, desde el principio tomamos a la sensibilidad Δ_a como $a_{max} - a_{min}$, pero podemos mejorarla cambiando el enfoque: en vez de sumar cada valor y dividirlo por el conteo, partimos de un resultado esperado, $(a_{min} + a_{max})/2$, y por cada elemento x añadimos $x_a - (a_{min} + a_{max})/2$ al total. Esto hace que pasemos a tener una sensibilidad de $\Delta_a = (a_{max} - a_{min})/2$, mejorando considerablemente la anterior. Esto nos lleva a proponer el siguiente algoritmo:

Algoritmo 4: Media con ruido, normalizada (*Noisy Average Normalization*)

Datos: D, $[a_{min}, a_{max}]$, ε.
$\widetilde{S} \leftarrow D.Suma - D.Conteo \cdot (a_{min} + a_{max})/2 + Lap((a_{max} - a_{min})/\varepsilon)$
$\widetilde{C} \leftarrow D.Conteo + Lap(\varepsilon/2)$
si $\widetilde{C} \leq 1$ **entonces**
  devolver $(a_{min} + a_{max})/2$
en otro caso
  devolver $\dfrac{\widetilde{S}}{\widetilde{C}} + (a_{min} + a_{max})/2$
fin

Que sabemos que satisface ε-DP por el mismo argumento de composición que usamos en el algoritmo 1.

4.3. Conclusiones

Tomando como medida el uso del presupuesto de privacidad ε, tanto el algoritmo 3 como el 4 duplican la eficacia del algoritmo 1. Aún así, podemos preguntarnos cuál es el mejor. En [12], de modo empírico (aunque no se ofrecen las medidas computacionales pertinentes) se concluye que el algoritmo 3 es preferible al 4, seguido del método que aplica el mecanismo exponencial, y luego el 1. Cuando ε es grande, 3 y 4 son similares, y sus errores bajo la norma ℓ_1 son aproximadamente la mitad que los de 1 y 2. Si ε es pequeño, la ventaja de 3 sobre 4 se hace considerable.

Capítulo 5

Técnica del vector disperso[1]

Frente a una gran cantidad de peticiones (*queries*) a la base de datos, la garantía de privacidad puede no ser del todo alcanzable. Una aproximación para resolver este problema consiste en filtrar, de un conjunto normalmente grande de peticiones, las que sobrepasen algún umbral que fijemos. Esto es, podemos descartar las peticiones que no sean relevantes para responder de forma adaptativa y versátil, y así no gastar presupuesto de privacidad (ε) con ellas.

Esta técnica se usa en dos entornos de formas diferentes: uno *offline*, no interactivo, donde conocemos el conjunto de peticiones de antemano, y otro *online*, o dinámico, donde no tendremos este conocimiento previo. Notemos que en el caso *offline* el problema es similar al mecanismo del informe del máximo con ruido visto en 2.2.1, y que también se puede abordar mediante el mecanismo exponencial.

Para un contexto *online* debemos recordar la Ley Fundamental de Recuperación de la Información que vimos en la introducción de este trabajo: responder una cantidad lineal de peticiones, aún añadiendo un pequeño ruido, puede poner en riesgo gran parte de la privacidad de los datos. Uno de los modos de enfrentar esto es precisamente a partir de las posibilidades que nos ofrece la SVT.

[1] En inglés, *Sparse Vector Technique* (SVT).

5.1. Introducción

Empezaremos mostrando los primeros pasos hacia un algoritmo SVT y a la vez definiendo las propiedades y los términos más importantes.

Sea m el número de peticiones totales que tendremos, que exigiremos que tengan sensibilidad 1. Sin pérdida de generalidad podemos elegir un umbral T, público, fijado de antemano[2]. Lo que haremos será añadir ruido a las peticiones y compararlas una a una con el umbral: un resultado positivo ($\top$) indicará que lo ha sobrepasado, y por tanto será una petición que nos interese, y un resultado negativo ($\bot$) nos hará descartarla. Normalmente usaremos un límite c para las peticiones que aceptaremos por encima de dicho umbral. Solo consumiremos presupuesto de privacidad ε con las peticiones aceptadas.

Algo importante a tener en cuenta es que al comparar las peticiones con el umbral, no usaremos T sino una versión $\hat{T}$ con ruido, que será privada. Esto se debe a razones técnicas para el cumplimiento de (ε, δ)-DP que aparecerán en la demostración del teorema 5.1.1.

5.1.1. Primeros pasos

El primer algoritmo que presentamos se detiene al encontrar la primera petición que sobrepasa el umbral, es decir, usando $c = 1$, y veremos que es (ε, δ)-DP sea cual sea la cantidad m de peticiones recibidas. Estudiaremos el caso de $c > 1$ a partir de los teoremas de composición.

Teorema 5.1.1 *El algoritmo 5 satisface (ε, δ)-DP.*

Demostración: Fijemos dos bases de datos adyacentes D y D'. Sea A el resultado aleatorio del algoritmo 5 con los datos de entrada (D, $\{f_i\}$, T, ε) y A' el resultado análogo para (D', $\{f_i\}$, T, ε). Lo que devuelve el algoritmo es una secuencia de k elementos $a \in \{\top, \bot\}^k$ donde para cada $i < k$, $a_i = \bot$ y $a_k = \top$.

Hay dos realizaciones de variables aleatorias dentro del mecanismo: el umbral con ruido $\hat{T}$ y las perturbaciones que sufre cada petición, $\{\nu_i\}_{i=1}^k$. Fijaremos los valores (aleatorios) de $\nu_1, ..., \nu_{k-1}$ y tomaremos probabilidades en ν_k y en $\hat{T}$. Sea $g(D) = \text{máx}_{i<k}(f_i(D) + \nu_i)$ el máximo valor con ruido de las peticiones que se hacen en D antes de sobrepasar el umbral con la k-ésima. Escribiremos,

[2]Si queremos aplicar un umbral distinto para cada petición, se puede tomar $\mathbf{T} = \{T_1, T_2, ..., T_m\}$ y, dado el conjunto de peticiones $\mathbf{f} = \{f_1, f_2, ..., f_m\}$, definir una nueva secuencia $q_i = f_i - T_i$ de peticiones sobre la que aplicar la SVT, poniendo como umbral $T = 0$.

Algoritmo 5: SobrepasaUmbral (*AboveThreshold*)

Datos: D, $\{f_i\}$, T, ε.
$\hat{T} \leftarrow T + Lap\,(2/\varepsilon)$
para cada *petición* f_i **hacer**
  $\nu_i \leftarrow Lap\,(4/\varepsilon)$
  si $f_i(D) + \nu_i \geq \hat{T}$ **entonces**
    devolver $a_i = \top$
    detener
  en otro caso
    devolver $a_i = \bot$
  fin
fin

abusando de la notación, $Pr[\hat{T} = t]$ como el valor de la función de densidad de $\hat{T}$ en t, y análogamente para ν_k. También usaremos $\mathbf{1}[x]$ como la función indicador del evento x. Entonces:

$$\begin{aligned}
\Pr_{\hat{T},\nu_k}[A = a] &= \Pr_{\hat{T},\nu_k}[(\hat{T} > g(D)) \wedge (f_k(D) + \nu_k \geq \hat{T})] \\
&= \Pr_{\hat{T},\nu_k}\left[\hat{T} \in (g(D), f_k(D) + \nu_k)]\right] \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \Pr[\nu_k = v] \cdot \Pr[\hat{T} = t] \cdot \mathbf{1}[t \in (g(D), f_k(D) + \nu_k]] dv dt \\
&= *
\end{aligned}$$

Haciendo el cambio de variables:

$$\begin{aligned}
\hat{v} &= v + g(D) - g(D') + f_k(D') - f_k(D) \\
\hat{t} &= t + g(D) - g(D')
\end{aligned}$$

Y teniendo en cuenta que, para cualesquiera D y D' adyacentes, como cada petición $f_i(D)$ tiene sensibilidad 1 (y por tanto $g(D)$ también), se tiene que $|\hat{v} - v| \leq 2$ y $|\hat{t} - t| \leq 1$. Aplicando esto:

$$\begin{aligned}
* &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \Pr[\nu_k = \hat{v}] \cdot \Pr[\hat{T} = \hat{t}] \cdot \\
&\qquad \cdot \mathbf{1}[t + g(D) - g(D') \in (g(D), f_k(D') + v + g(D) - g(D')]] dv dt \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \Pr[\nu_k = \hat{v}] \cdot \Pr[\hat{T} = \hat{t}] \cdot \mathbf{1}[t \in (g(D'), f_k(D') + v]] dv dt \\
&\leq \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{\varepsilon/2} \Pr[\nu_k = v] e^{\varepsilon/2} \cdot \Pr[\hat{T} = t] \cdot \mathbf{1}[t \in (g(D'), f_k(D') + v]] dv dt \\
&= e^{\varepsilon} \Pr_{\hat{T},\nu_k}[(\hat{T} > g(D')) \wedge (f_k(D') + \nu_k \geq \hat{T})] = e^{\varepsilon} \Pr_{\hat{T},\nu_k}[A' = a]
\end{aligned}$$

□

Para garantizar y medir adecuadamente la funcionalidad de este tipo de algoritmos necesitaríamos un conocimiento exhaustivo de los errores posibles. Como añadimos ruido tanto al umbral como a las peticiones, se abre las posibilidad (aunque es exponencialmente pequeña) de que una petición con un valor pequeño nos salga como aceptada o que otra con un valor grande resulte rechazada.

Notación 5.1.2 *Usaremos * como exponente para referirnos a un número indeterminado a priori.*

Definición 5.1.3 *Diremos que un algoritmo que devuelve $a_1, a_2, ... \in \{\top, \bot\}^*$ en respuesta a un conjunto de peticiones $f_1, ..., f_k$ es (α, β)-**preciso** respecto al umbral T si, con probabilidad de error de como mucho β, el algoritmo no se detiene antes de procesar f_k y,*

$$\begin{cases} f_i(D) \geq T - \alpha & \text{si } a_i = \top \\ f_i(D) \leq T + \alpha & \text{si } a_i = \bot \end{cases}$$

Notemos que en la variable α se incluye tanto la perturbación en $\hat{T}$ como en los ν_i.

Teorema 5.1.4 *Para cualquier secuencia de peticiones $f_1, ..., f_k$ tales que se cumpla $|\{i < k : f_i(D) \geq T - \alpha\}| = 0$ (es decir, que la única petición que se aproxima al umbral o lo sobrepasa es la última), el algoritmo 5 es (α, β)-preciso para:*

$$\alpha = \frac{8(\log k + \log(2/\beta))}{\varepsilon}$$

Demostración: Equivalentemente al enunciado, vamos a probar que, excepto con probabilidad de a lo sumo β:

$$\max_{i=1,...,k} |\nu_i| + |T - \hat{T}| \leq \alpha$$

O lo que es lo mismo, para todo $a_i = \top$ se tiene:

$$f_i(D) + \nu_i \geq \hat{T} \geq T - |T - \hat{T}| \implies f_i(D) \geq T - |T - \hat{T}| - |\nu_i| \geq T - \alpha$$

Y para todo $a_i = \bot$:

$$f_i(D) < \hat{T} \leq T + |T - \hat{T}| + |\nu_i| \leq T + \alpha$$

También, para los $i < k$,

$$f_i(D) < T - \alpha < T - |\nu_i| - |T - \hat{T}| \implies f_i(D) + \nu_i \leq \hat{T}$$

Así, como condición para que el algoritmo no pare antes de responder la k peticiones, será $a_i = \perp$. Dividiremos el margen de error α en $\alpha/2$ para el error en $\hat{T}$, con probabilidad de como mucho $\beta/2$, y haremos lo análogo para el error en ν_i. Aplicando el lema 2.1.4:

$$\Pr\left[|T - \hat{T}| \geq \frac{\alpha}{2}\right] = \exp\left(-\frac{\varepsilon\alpha}{4}\right)$$

Sustituyendo para que la probabilidad sea como mucho $\beta/2$, debe ser

$$\alpha \geq \frac{4\log(2/\beta)}{\varepsilon}$$

Por otro lado:

$$\Pr\left[\max_{i\in[k]} |\nu_i| \geq \frac{\alpha}{2}\right] \leq k \cdot \exp\left(-\frac{\varepsilon\alpha}{8}\right)$$

Estableciendo otra vez que sea como mucho $\beta/2$, resulta que

$$\alpha \geq \frac{8(\log(2/\beta) + \log k)}{\varepsilon}$$

Con estos dos resultados se completa la prueba. □

5.1.2. Algoritmo de dispersión

El estudio del caso anterior, donde $c = 1$, nos sirve de base para establecer cotas mayores. En estos casos, frente a un flujo de peticiones que vayan llegando, usaremos el algoritmo anterior hasta encontrar el primer positivo, y lo reiniciaremos c veces a medida que consumimos las peticiones entrantes. Cada uso del algoritmo 5 es $(\varepsilon, 0)$-DP, con lo que podemos aplicar los teoremas de composición.

Teorema 5.1.5 *El algoritmo 6 satisface (ε, δ)-DP.*

Demostración: Notemos que el algoritmo 6 es equivalente a aplicar el algoritmo 5 con los parámetros $(D, \{f_i\}, T, \varepsilon')$ en la misma secuencia de peticiones $\{f_i\}$ estableciendo

$$\varepsilon' = \begin{cases} \varepsilon/c & \text{si } \delta = 0 \\ \varepsilon/\sqrt{8c\ln(1/\delta)} & \text{en otro caso} \end{cases}$$

Algoritmo 6: Dispersión (*Sparse*)

Datos: D, $\{f_i\}$, T, c, ε, δ.
si $\delta = 0$ **entonces**
  $\sigma \leftarrow 2c/\varepsilon$
en otro caso
  $\sigma \leftarrow \left(\sqrt{32c\ln(1/\delta)}\right)/\varepsilon$
fin
$\hat{T}_0 \leftarrow T + Lap(\sigma)$
contador $\leftarrow 0$
para cada *petición* f_i **hacer**
  $\nu_i \leftarrow Lap(2\sigma)$
  si $f_i(D) + \nu_i \geq \hat{T}_{contador}$ **entonces**
    devolver $a_i = \top$
    contador $\leftarrow$ contador $+1$
    $\hat{T}_{contador} \leftarrow T + Lap(\sigma)$
  en otro caso
    devolver $a_i = \perp$
  fin
  si *contador* $\geq c$ **entonces**
    detener
  fin
fin

Cuando el algoritmo 5 se detiene, lo volvemos a aplicar sobre el siguiente conjunto de peticiones de la secuencia de entrada, hasta agotar estos reinicios c veces.

Ya hemos probado que el algoritmo 5 cumple ε-DP, de modo que aplicando el teorema 1.2.17 (de composición avanzada) c veces con la elección de ε' que hemos tomado, se tiene el resultado. □

Queda probar la precisión de este algoritmo, que podemos hacer de nuevo por composición. Basta ver que si cada ejecución de 5 es $(\alpha, \beta/c)$-precisa, entonces 6 es (α, β)-preciso.

Teorema 5.1.6 *Para cualquier secuencia de k peticiones $f_1, ..., f_k$ tales que $|\{i : f_i(D) \geq T - \alpha\}| \geq c$, si $\delta > 0$, el algoritmo 6 es (α, β)-preciso para:*

$$\alpha = \frac{(\ln k + \ln \frac{2c}{\beta})\sqrt{512c \ln \frac{1}{\delta}}}{\varepsilon}$$

Y si $\delta = 0$, es (α, β)-preciso para:

$$\alpha = \frac{8c(\ln k + \ln(2c/\beta)}{\varepsilon}$$

Demostración: Basta aplicar el teorema 5.1.4 escogiendo β/c como β y ε como el ε' de la demostración del teorema 5.1.5. □

5.1.3. Dispersión numérica

En esta sección se presenta una modificación del algoritmo de dispersión que hemos visto: cuando aceptamos una petición (es decir, supera el umbral que hayamos fijado) devolveremos, en vez de un indicador, su valor numérico. Veremos que esto se puede conseguir a costa de solo un factor constante de pérdida de privacidad.

Teorema 5.1.7 *El algoritmo 7 cumple (ε, δ)-DP.*

Demostración: Se prueba por composición. Si $\delta = 0$, aplicar el algoritmo con los parámetros $(D, \{f_i\}, T, c, \varepsilon, 0)$ es ejecutar Dispersión$(D, \{f_i\}, T, c, \frac{8}{9}\varepsilon, 0)$ y luego un mecanismo de Laplace con $(\varepsilon', \delta') = (\frac{1}{9}\varepsilon, 0)$.
Si $\delta > 0$, entonces DispersiónNumérica$(D, \{f_i\}, T, c, \varepsilon, \delta)$ es componer Dispersión$(D, \{f_i\}, T, c, \varepsilon\sqrt{512}/(\sqrt{512}+1), \delta/2)$ con un mecanismo de Laplace de $(\varepsilon', \delta') = (\varepsilon/(\sqrt{512}+1), \delta/2)$. □

Algoritmo 7: DispersiónNumérica (*NumericSparse*)

Datos: D, $\{f_i\}$, T, c, ε, δ.
si $\delta = 0$ **entonces**
  $\varepsilon_1 \leftarrow 8\varepsilon/9$
  $\varepsilon_2 \leftarrow 2\varepsilon/9$
en otro caso
  $\varepsilon_1 \leftarrow \varepsilon\sqrt{512}/(\sqrt{512}+1)$
  $\varepsilon_2 \leftarrow 2\varepsilon/(\sqrt{512}+1)$
fin
si $\delta = 0$ **entonces**
  $\sigma(\varepsilon) \leftarrow 2c/\varepsilon$
en otro caso
  $\sigma(\varepsilon) \leftarrow \left(\sqrt{32c\ln(2/\delta)}\right)/\varepsilon$
fin
$\hat{T}_0 \leftarrow T + Lap(\sigma(\varepsilon_1))$
contador $\leftarrow 0$
para cada *petición* f_i **hacer**
  $\nu_i \leftarrow Lap(2\sigma(\varepsilon_1))$
  si $f_i(D) + \nu_i \geq \hat{T}_{contador}$ **entonces**
    $\nu_i \leftarrow Lap(\sigma(\varepsilon_2))$
    devolver $a_i = f_i(D) + \nu_i$
    contador $\leftarrow$ contador $+1$
    $\hat{T}_{contador} \leftarrow T + Lap(\sigma(\varepsilon_1))$
  en otro caso
    devolver $a_i = \perp$
  fin
  si *contador* $\geq c$ **entonces**
    detener
  fin
fin

Para hablar de la precisión en este caso, que cambia el conjunto de salida, tenemos que redefinir el concepto.

Definición 5.1.8 *Diremos que un algoritmo que recibe un conjunto de k peticiones $f_1, ..., f_k$ y devuelve una respuesta $a_1, a_2, ... \in (\mathbb{R} \cup \{\perp\})^*$ es (α, β)-**preciso** respecto a un umbral T si, con una posibilidad de error de como mucho β, el algoritmo no se detiene antes de procesar todas las peticiones (hasta f_k) y:*

$$\begin{cases} |f_i(D) - a_i| \leq \alpha & \text{si } a_i \in \mathbb{R} \\ f_i(D) \leq T + \alpha & \text{si } a_i = \perp \end{cases}$$

Teorema 5.1.9 *Dada una secuencia de k peticiones $f_1, ..., f_k$ tales que*

$$|\{i : f_i(D) \geq T - \alpha\}| \geq c$$

si $\delta > 0$, el algoritmo 7 es (α, β)-preciso para:

$$\alpha = \frac{(\ln k + \ln \frac{4c}{\beta})(\sqrt{512} + 1)\sqrt{c \ln \frac{2}{\delta}}}{\varepsilon}$$

Y si $\delta = 0$, es (α, β)-preciso para:

$$\alpha = \frac{9c(\ln k + \ln(4c/\beta))}{\varepsilon}$$

Demostración: La precisión requiere dos condiciones. La primera, que para todo $i \leq k$ tal que $a_i = \perp$, debe ser $f_i(D) \leq T + \alpha$. Esto se cumple con probabilidad de $1 - \beta/2$ por el teorema 5.1.6. La otra condición es que para todo $i \leq k$ tal que $a_i \in \mathbb{R}$, es $|f_i(D) - a_i| \leq \alpha$, que se cumple igualmente con probabilidad de $1 - \beta/2$ por la acción del mecanismo de Laplace. □

Este resultado es de gran interés: esta precisión es igual, con una diferencia de constantes y de un factor de $\ln k$, a la que podemos obtener con el mismo presupuesto de privacidad si conociéramos las peticiones de antemano y las respondiéramos con el mecanismo de Laplace. Esto es, la SVT nos permite identificar el valor de las peticiones que sobrepasan un umbral a bajo coste: de orden logarítmico respecto al número de peticiones irrelevantes. Es similar a lo que podríamos conseguir seleccionando primero las peticiones con el mecanismo exponencial y luego respondiéndolas con el mecanismo de Laplace.

5.2. Otros algoritmos

La Privacidad Diferencial, y concretamente la Técnica del Vector Disperso, son herramientas que han tenido su aparición en los últimos años. En esta

Algoritmo A.1: SVT en [12]

Datos: D, $\{f_i\}$, Δ, $\{T_i\}$, c, ε.
$\varepsilon_1 \leftarrow \varepsilon/2, \quad \sigma \leftarrow Lap(\Delta/\varepsilon_1)$
$\varepsilon_2 \leftarrow \varepsilon - \varepsilon_1, \quad$ contador $\leftarrow 0$
para cada *petición* f_i **hacer**
  $\nu_i \leftarrow Lap(2c\Delta/\varepsilon_2)$
  si $f_i(D) + \nu_i \geq T_i + \sigma$ **entonces**
    devolver $a_i = \top$
    contador $\leftarrow$ contador $+1$
    detener si contador $\geq c$
  en otro caso
    devolver $a_i = \bot$
  fin
fin

Algoritmo A.2: SVT en [7]

Datos: D, $\{f_i\}$, Δ, T, c, ε.
$\varepsilon_1 \leftarrow \varepsilon/2, \quad \sigma \leftarrow Lap(c\Delta/\varepsilon_1)$
$\varepsilon_2 \leftarrow \varepsilon - \varepsilon_1, \quad$ contador $\leftarrow 0$
para cada *petición* f_i **hacer**
  $\nu_i \leftarrow Lap(2c\Delta/\varepsilon_1)$
  si $f_i(D) + \nu_i \geq T + \sigma$ **entonces**
    devolver $a_i = \top, \quad \sigma \leftarrow Lap(c\Delta/\varepsilon_2)$
    contador $\leftarrow$ contador $+1$
    detener si contador $\geq c$
  en otro caso
    devolver $a_i = \bot$
  fin
fin

Figura 5.1: Una selección de distintos mecanismos de SVT (1).

sección veremos algunas variantes de SVT que se han dado a lo largo de varios autores, aunque no estudiaremos los algoritmos tan a fondo como hemos hecho con los de la sección anterior. Entre los algoritmos que aparecen en la figuras 5.1 y 5.2 resulta conveniente tener en cuenta a los que se demostró que no satisfacían ε-DP: aunque esto les quite la utilidad (el propósito de la SVT es mantener la privacidad) el análisis de las diferencias que presentan nos parece interesante.

La estructura, en general, es la misma:

1. Se genera ruido σ para perturbar el umbral T, que se aplicará luego en la comparación con las peticiones.
2. Se genera ruido ν_i para cada petición f_i.
3. Realizamos la comparación entre las peticiones y el umbral, considerando la distorsión añadida, y devolvemos el resultado.
4. Mantenemos la cuenta de positivos y paramos cuando llegamos a c.

Podemos ver una tabla con las propiedades y diferencias entre cada algoritmo de la figuras 5.1 y 5.2 en la figura 5.3. Hemos introducido otro parámetro Δ que indica la sensibilidad (máxima) que puede tener el conjunto de peticiones. En el estudio anterior, por comodidad, utilizamos $\Delta = 1$.

El mecanismo A.2 de la figura es el algoritmo 6 que vimos en la sección anterior, que probamos que cumple ε-DP. El algoritmo A.1 ofrece una ventaja

Algoritmo A.3: SVT en [15]

Datos: D, $\{f_i\}$, Δ, T, c, ε.
$\varepsilon_1 \leftarrow \varepsilon/2, \quad \sigma \leftarrow Lap(\Delta/\varepsilon_1)$
$\varepsilon_2 \leftarrow \varepsilon - \varepsilon_1, \quad$ contador $\leftarrow 0$
para cada *petición* f_i **hacer**
  $\nu_i \leftarrow Lap(c\Delta/\varepsilon_2)$
  si $f_i(D) + \nu_i \geq T + \sigma$ **entonces**
    devolver $a_i = f_i(D) + \nu_i$
    contador $\leftarrow$ contador $+1$
    detener si contador $\geq c$
  en otro caso
    devolver $a_i = \perp$
  fin
fin

Algoritmo A.4: SVT en [11]

Datos: D, $\{f_i\}$, Δ, T, c, ε.
$\varepsilon_1 \leftarrow \varepsilon/4, \quad \sigma \leftarrow Lap(\Delta/\varepsilon_1)$
$\varepsilon_2 \leftarrow \varepsilon - \varepsilon_1, \quad$ contador $\leftarrow 0$
para cada *petición* f_i **hacer**
  $\nu_i \leftarrow Lap(c\Delta/\varepsilon_2)$
  si $f_i(D) + \nu_i \geq T + \sigma$ **entonces**
    devolver $a_i = \top$
    contador $\leftarrow$ contador $+1$
    detener si contador $\geq c$
  en otro caso
    devolver $a_i = \perp$
  fin
fin

Algoritmo A.5: SVT en [17]

Datos: D, $\{f_i\}$, Δ, T, ε.
$\varepsilon_1 \leftarrow \varepsilon/2, \quad \sigma \leftarrow Lap(\Delta/\varepsilon_1)$
$\varepsilon_2 \leftarrow \varepsilon - \varepsilon_1$
para cada *petición* f_i **hacer**
  $\nu_i \leftarrow 0$
  si $f_i(D) + \nu_i \geq T + \sigma$ **entonces**
    devolver $a_i = \top$
  en otro caso
    devolver $a_i = \perp$
  fin
fin

Algoritmo A.6: SVT en [4]

Datos: D, $\{f_i\}$, Δ, $\{T_i\}$, ε.
$\varepsilon_1 \leftarrow \varepsilon/2, \quad \sigma \leftarrow Lap(\Delta/\varepsilon_1)$
$\varepsilon_2 \leftarrow \varepsilon - \varepsilon_1$
para cada *petición* f_i **hacer**
  $\nu_i \leftarrow Lap(\Delta/\varepsilon_2)$
  si $f_i(D) + \nu_i \geq T_i + \sigma$ **entonces**
    devolver $a_i = \top$
  en otro caso
    devolver $a_i = \perp$
  fin
fin

Figura 5.2: Una selección de distintos mecanismos de SVT (2).

frente a este: no es necesario hacer a σ dependiente de c, ni alterar σ después de cada positivo, para mantener la ε-DP. Esto lo hace más eficaz.

El mecanismo A.3 se probó ineficiente por el siguiente hecho: al revelar el valor de las peticiones aceptadas, aún con ruido, se proporciona también información sobre $\hat{T}$, a costa de la garantía de privacidad. Esto se soluciona en el algoritmo 7 que ya hemos visto, actualizando el ruido contenido en $\hat{T}$ después de cada positivo.

Las demostraciones que se ofrecieron de la privacidad de los mecanismos A.4-A.6 resultaron ser erróneas a partir de un error de integración que se describe someramente en [12], aunque no vemos necesario entrar en los detalles. Sin embargo, el mecanismo A.4 se utilizó en [11] como método para encontrar

	Alg. A.1	**Alg. A.2**	**Alg. A.3**	**Alg. A.4**	**Alg. A.5**	**Alg. A.6**
ε_1	$\varepsilon/2$	$\varepsilon/2$	$\varepsilon/2$	$\varepsilon/4$	$\varepsilon/2$	$\varepsilon/2$
Escala del ruido del umbral σ	Δ/ε_1	$c\Delta/\varepsilon_1$	Δ/ε_1	Δ/ε_1	Δ/ε_1	Δ/ε_1
Restablece σ después de cada positivo		**Sí**				
Escala del ruido ν_i	$2c\Delta/\varepsilon_2$	$2c\Delta/\varepsilon_2$	$c\Delta/\varepsilon_2$	Δ/ε_2	0	Δ/ε_2
Devuelve $f_i + \nu_i$ en vez de $\top$ **(no privado)**			**Sí**			
Informa sobre los $\top$s **(no privado)**					**Sí**	**Sí**
Tipo de privacidad	ε-DP	ε-DP		$(\frac{1+6c}{4}\varepsilon)$-DP		

Figura 5.3: Diferencias entre los algoritmos A.1-A.6. Traducido de [12].

artículos frecuentes (*finding frequent itemsets*) en un contexto donde las peticiones eran monótonas, de modo que el presupuesto de privacidad puede variar según se cumpla esta condición o no, y se puede alcanzar alguna ε-DP con él. Veremos la relación de la SVT y este tipo de funciones en la sección 5.3.2.

Notemos que en los mecanismos A.1 y A.6 se han usado umbrales vectoriales $\mathbf{T} = T_1, T_2, ...$ en contraposición a los usados en el resto.

Aunque existan técnicas como las que hemos visto en el teorema 5.1.7 para estudiar la (ε, δ)-DP, cabe destacar que en las aplicaciones prácticas que se usan para minería de datos se ha preferido fijar la atención en el cumplimiento de «solo» ε-DP.

5.3. Optimización

En esta sección vamos a estudiar las técnicas principales con las que podemos alterar los parámetros de las SVT para hacerlos más eficientes. Lo dividiremos en tres partes: un estudio de cómo repartir el presupuesto de privacidad entre las fases de cada algoritmo, la influencia que puede tener operar sobre peticiones monótonas, y una comparación en el caso *offline* de SVT y el mecanismo exponencial.

5.3.1. Presupuesto de privacidad

Dado un presupuesto de privacidad ε, vamos a estudiar cómo repartirlo entre la cantidad que vaya dirigida a perturbar el umbral T y la que se use para las peticiones. Por ejemplo, el mecanismo A.1 usa una relación de $1:1$ mientras que en A.4 vemos una de $1:3$.

Algoritmo 8: SVT Estándar (*Standard SVT*)

Datos: D, $\{f_i\}$, Δ, T, c, $\varepsilon_1, \varepsilon_2, \varepsilon_3$.
$\sigma \leftarrow Lap(\Delta/\varepsilon_1)$
contador $\leftarrow 0$
para cada *petición* f_i **hacer**
  $\nu_i \leftarrow Lap(2c\Delta/\varepsilon_2)$
  si $f_i(D) + \nu_i \geq T_i + \sigma$ **entonces**
    si $\varepsilon_3 > 0$ **entonces**
      devolver $a_i = f_i(D) + Lap(c\Delta/\varepsilon_3)$
    en otro caso
      devolver $a_i = \top$
    fin
    contador $\leftarrow$ contador $+1$
    si *contador* $\geq c$ **entonces**
      detener
    fin
  en otro caso
    devolver $a_i = \bot$
  fin
fin

En general, usamos $\varepsilon = \varepsilon_1 + \varepsilon_2 + \varepsilon_3$ como presupuesto de privacidad. ε_1 indica el ruido que añadimos al umbral, ε_2 el de las peticiones, y ε_3, en el caso de que usemos la dispersión numérica, el que se le añade al valor de las peticiones.

Teorema 5.3.1 *El algoritmo 8 es* $(\varepsilon_1 + \varepsilon_2 + \varepsilon_3)$*-DP.*

Demostración: La prueba se puede dividir en dos partes: en la primera se detecta si una petición sobrepasa el umbral, y en la segunda se usa el mecanismo de Laplace. Este último cumple ε_3-DP, y es el que revela, con ruido, las peticiones aceptadas. Como esto es una composición que ya hemos visto, basta ver que la primera parte cumple $(\varepsilon_1 + \varepsilon_2)$-DP. Sea $\mathcal{M}$ el mecanismo que estudiamos y $a \in \{\top, \bot\}^\ell$:

$$\Pr\left[\mathcal{M}(D) = a\right] = \int_{-\infty}^{\infty} \Pr[\sigma = z] g_D(z) h_D(z) dz$$

Notando:

$$\begin{aligned} g_D(z) &= \prod_{i \in I_\perp} \Pr[f_i(D) + \nu_i < T_i + z] \\ h_D(z) &= \prod_{i \in I_\top} \Pr[f_i(D) + \nu_i \geq T_i + z] \\ I_\perp &= \{j = 1, ..., \ell : a_j = \perp\} \\ I_\top &= \{j = 1, ..., \ell : a_j = \top\} \end{aligned}$$

Veamos que:

$$g_D(z) \leq g_{D'}(z + \Delta) \tag{5.1}$$
$$h_D(z) \leq e^{\varepsilon_2} h_{D'}(z + \Delta) \tag{5.2}$$
$$\Pr[\sigma = z] \leq e^{\varepsilon_1} \Pr[\sigma = z + \Delta] \tag{5.3}$$

Donde hemos supuesto, sin pérdida de generalidad, que en las bases de datos adyacentes D y D' se tiene que $f_i(D) \leq f_i(D') - \Delta$ para toda petición f_i.

Por tanto, para (5.1):

$$\begin{aligned} \Pr[f_i(D) + \nu_i < T_i + z] &= \Pr[\nu_i < T_i - f_i(D) + z] \\ &\leq \Pr[\nu_i < T_i + \Delta - f_i(D') + z] \qquad (5.4) \\ &= \Pr[f_i(D') + \nu_i < T_i + (z + \Delta)] \end{aligned}$$

La desigualdad (5.4) es una consecuencia de que $|f_i(D) - f_i(D')| \leq \Delta$.

En (5.2):

$$\begin{aligned} h_D(z) &= \prod_{i \in I_\top} \Pr[\nu_i \geq T_i + z - f_i(D)] \\ &\leq \prod_{i \in I_\top} \Pr[\nu_i \geq T_i + z - \Delta - f_i(D')] \qquad (5.5) \\ &\leq \prod_{i \in I_\top} e^{\varepsilon_2/c} \Pr[\nu_i \geq T_i + z - \Delta - f_i(D') + 2\Delta] \qquad (5.6) \\ &\leq e^{\varepsilon_2} \prod_{i \in I_\top} \Pr[f_i(D') + \nu_i \geq T_i + z + \Delta] \qquad (5.7) \\ &= e^{\varepsilon_2} h_{D'}(z + \Delta) \end{aligned}$$

(5.5) se justifica de la misma forma que (5.4). (5.7) se sigue de que $|I_\top| \leq c$. Las ecuaciones (5.6) y (5.3) se demuestran al operar con la distribución de Laplace, con parámetros $2c\Delta/\varepsilon_2$ y Δ/ε_1 respectivamente.

Utilizando (5.1), (5.2) y (5.3):

$$\begin{aligned}\Pr\left[\mathcal{M}(D)=a\right] &= \int_{-\infty}^{\infty} \Pr[\sigma=z] g_D(z) h_D(z) dz \\ &\leq \int_{-\infty}^{\infty} e^{\varepsilon_1+\varepsilon_2} \Pr[\sigma=z+\Delta] g_{D'}(z+\Delta) h_{D'}(z+\Delta) dz \\ &= e^{\varepsilon_1+\varepsilon_2} \Pr\left[\mathcal{M}(D')=a\right]\end{aligned}$$

□

Mientras usamos $\varepsilon_1+\varepsilon_2$ para aceptar las peticiones, ε_3 depende del contexto, el dominio con el que liberamos los resultados. Es decir, la proporción $(\varepsilon_1+\varepsilon_2) : \varepsilon_3$ dependerá de cada caso particular.

Por otra parte, la proporción $\varepsilon_1 : \varepsilon_2$ afecta a la precisión de la SVT. Tenemos:

$$f_i(D) + Lap\left(\frac{2c\Delta}{\varepsilon_2}\right) \geq T + Lap\left(\frac{\Delta}{\varepsilon_1}\right)$$

Si minimizamos la varianza de la comparación, esto es, de $Lap\left(\frac{\Delta}{\varepsilon_1}\right) - Lap\left(\frac{2c\Delta}{\varepsilon_2}\right)$, que es:

$$2\left(\frac{\Delta}{\varepsilon_1}\right)^2 + 2\left(\frac{2c\Delta}{\varepsilon_2}\right)^2$$

Fijando $\varepsilon_1+\varepsilon_2$, queda que se debe cumplir:

$$\varepsilon_1 : \varepsilon_2 = 1 : (2c)^{2/3}$$

En un contexto dinámico, *online*, el algoritmo 8 con estos parámetros es el que mejor funciona de todos los que hemos visto.

5.3.2. Peticiones monótonas

El resultado principal en el caso de que las peticiones sean monótonas es que podemos usar, para añadir ruido a las peticiones, $Lap\left(c\Delta/\varepsilon_2\right)$ en vez de $Lap\left(2c\Delta/\varepsilon_2\right)$.

Teorema 5.3.2 *Si todas las peticiones son monótonas, sustituyendo en el algoritmo 8 el ruido $\nu_i = Lap\left(2c\Delta/\varepsilon_2\right)$ por $\nu_i = Lap\left(c\Delta/\varepsilon_2\right)$, se sigue cumpliendo la $(\varepsilon_1+\varepsilon_2+\varepsilon_3)$-DP.*

Demostración: Como en la demostración del teorema 5.3.1, el ruido que introduce ε_3 es una composición con el mecanismo de Laplace en la última etapa. Vamos a probar que, para todo $a \in \{\top, \bot\}^{\ell}$:

$$\begin{aligned} \Pr\left[\mathcal{M}(D) = a\right] &= \int_{-\infty}^{\infty} \Pr[\sigma = z] g_D(z) h_D(z) dz \\ &\leq e^{\varepsilon_1 + \varepsilon_2} \Pr\left[\mathcal{M}(D') = a\right] \end{aligned} \tag{5.8}$$

Donde, como antes:

$$\begin{aligned} g_D(z) &= \prod_{i \in I_\bot} \Pr[f_i(D) + \nu_i < T_i + z] \\ h_D(z) &= \prod_{i \in I_\top} \Pr[f_i(D) + \nu_i \geq T_i + z] \end{aligned}$$

Se pueden dar dos casos distintos en los que se cumple 5.8. Por un lado:

$$\Pr[\sigma = z] g_D(z) h_D(z) \leq e^{\varepsilon_1 + \varepsilon_2} \Pr\left[\sigma = z\right] g_{D'}(z) h_{D'}(z) \tag{5.9}$$

O bien:

$$\Pr[\sigma = z] g_D(z) h_D(z) \leq e^{\varepsilon_1 + \varepsilon_2} \Pr\left[\sigma = z + \Delta\right] g_{D'}(z + \Delta) h_{D'}(z + \Delta) \tag{5.10}$$

Consideraremos primero el caso (5.9) en el que $f_i(D) \geq f_i(D')$ para todas las peticiones f_i. Entonces:

$$\Pr[f_i(D) + \nu_i < T_i + z] \leq \Pr[f_i(D') + \nu_i < T_i + z] \implies g_D(z) \leq g_{D'}(z)$$

Además,

$$h_D(z) \leq e^{\varepsilon_2} h_{D'}(z)$$

Sin que sea necesario incrementar cada umbral T_i en Δ, porque por el mismo argumento que el usado en la ecuación (5.2):

$$\begin{aligned} \Pr[f_i + \nu_i \geq T_i + z] &\leq \Pr[f_i(D') + \nu_i \geq T_i + z - \Delta] \\ &\leq e^{\varepsilon_2 / c} \Pr[f_i(D') + \nu_i \geq T_i + z] \end{aligned} \tag{5.11}$$

Con la diferencia entre (5.11) y (5.6) de que ahora podemos usar $\nu_i \sim Lap\left(\dfrac{c\Delta}{\varepsilon_2}\right)$.

La ecuación (5.10) ocurre cuando $f_i(D) \leq f_i(D')$. Como antes, se cumple que:

$$\begin{aligned} g_D(z) &\leq g_{D'}(z + \Delta) \\ \Pr[\sigma = z] &\leq e^{\varepsilon_1} \Pr[\sigma = z + \Delta] \end{aligned}$$

Y para la condición:

$$h_D(z) \leq e^{\varepsilon_2} h_{D'}(z)$$

Basta usar, como antes, $\nu_i \sim Lap\left(\frac{c\Delta}{\varepsilon_2}\right)$ para asegurar:

$$\Pr[f_i + \nu_i \geq T_i + z] \leq e^{\varepsilon_2/c} \Pr[f_i(D') + \nu_i \geq T_i + \Delta + z]$$

□

Proposición 5.3.3 *Para peticiones monótonas, la distribución óptima del presupuesto de privacidad entre ε_1 y ε_2 es:*

$$\varepsilon_1 : \varepsilon_2 = 1 : c^{2/3}$$

5.3.3. Relación con el mecanismo exponencial

La forma en la que abordemos las SVT es distinta según el caso *offline*, en el que las peticiones se conozcan de antemano, o el dinámico, *online*, donde no ocurra así.

En [4, 11, 17], esto es, una parte de la bibliografía reciente en la que se propusieron los algoritmos A.4, A.5 y A.6 respectivamente, algunas de las motivaciones fueron:

- Encontrar los c artículos más frecuentes de una colección.
- Determinar la estructura de una red bayesiana[3] que preserve tanta información como sea posible respecto de los datos originales.
- Elegir qué parámetros compartir al intentar desarrollar un modelo de red neuronal privado sobre los datos

En todos estos casos (notemos, además, que son contextos *offline*) el objetivo era seleccionar grupos de datos que tengan el valor de algún atributo más alto que el resto.

Como vimos en el capítulo 3, este problema también se puede abordar con el mecanismo exponencial. Específicamente, bastaría con aplicarlo c veces, cada

[3]Una red bayesiana es un modelo gráfico probabilístico que representa un conjunto de variables y las relaciones condicionadas de dependencia entre ellas. Por ejemplo, las relaciones entre distintas enfermedades y síntomas.

una de ellas con presupuesto de privacidad ε/c, y en cada paso eliminar de la lista de peticiones a la que hayamos obtenido como resultado. Relacionando la función de utilidad con el valor del parámetro que queremos filtrar, se tiene que la probabilidad de escoger alguna petición es proporcional a $\exp(\varepsilon/2c\Delta)$ y, en el caso de peticiones monótonas, a $\exp(\varepsilon/c\Delta)$.

Entonces, de entre SVT o el mecanismo exponencial, ¿cuál ofrece más garantías?

Definición 5.3.4 *Diremos que un algoritmo de SVT es* (α, β)-***correcto*** *si, al recibir una lista de* k *peticiones* $f_1, ..., f_k$*, rechaza las* $k-1$ *primeras y elige* f_k *con un valor de como mínimo* $T + \alpha$*, con probabilidad de al menos* $1 - \beta$*.*

Análogamente,

Definición 5.3.5 *Un algoritmo de selección de peticiones que se base en el mecanismo exponencial es* (α, β)-***correcto*** *si, al recibir una lista de* k *peticiones* $f_1, ..., f_k$*, rechaza una cantidad* $k-1$ *de ellas con valor menor o igual a* $T - \alpha$ *y elige la restante con un valor de como mínimo* $T + \alpha$*, con probabilidad de al menos* $1 - \beta$*.*

Estos conceptos son similares a la (α, β)-precisión que ya hemos estudiado. Mientras tenemos que, para las SVT:

$$\alpha_{SVT} = \frac{8(\ln k + \ln(2/\beta))}{\varepsilon}$$

Para el mecanismo exponencial, aplicando su definición resulta que la probabilidad de seleccionar una petición con una respuesta mayor o igual que $T + \alpha$, debe ser, como mínimo:

$$\frac{\exp(\varepsilon(T+\alpha)/2)}{(k-1)\exp(\varepsilon(T+\alpha)/2) + \exp(\varepsilon(T+\alpha)/2)}$$

Ajustándolo para que la probabilidad de este caso sea de al menos $1 - \beta$, de la misma forma que en la demostración del teorema 5.1.4, se obtiene:

$$\alpha_{ME} = \frac{(\ln(k-1) + \ln((1-\beta)/\beta))}{\varepsilon}$$

Que es menos que 8 veces α_{SVT}, lo que nos indica que el mecanismo exponencial es más «preciso» en los términos en los que hemos hablado.

Con todo, con la definición de (α, β)-correcto hemos supuesto que las primeras $k-1$ peticiones tienen un valor menor que $T - \alpha$. Si esto no fuera así,

los métodos que se proporcionan para el análisis de la comparación entre SVT y el mecanismo exponencial son los de mediciones experimentales.

En [12] se ofrece una evaluación de estos resultados: para el caso *offline*, el mecanismo exponencial supera las garantías de todas las variantes de SVT que hemos visto hasta ahora. Por otro lado, en el caso *online*, el algoritmo 8 es el más efectivo.

5.4. Software

Una versión de SVT escrita en Java dirigida al aprendizaje automático puede encontrarse en:

`https://github.com/mimno/Mallet/blob/master/src/cc/mallet/types/SparseVector.java`

Existe una librería muy reciente escrita en Python que incluye no solo las SVT, sino algunos de los mecanismos que hemos visto a lo largo del trabajo (como el informe del máximo con ruido de la sección 2.2.1), que está disponible en:

`https://github.com/fricklerhandwerk/differential-privacy`

Conclusiones

Aunque la Privacidad Diferencial es un campo reciente de la Ciencia de Datos, dispone de una bibliografía matemática rigurosamente formalizada que ofrece teoremas, propiedades y mecanismos que garantizan una correcta protección sobre el control de la información particular que puede ejercer cada usuario sobre sus datos. A su vez, mientras *anonimiza* la información de cada individuo, permite la extracción de conclusiones útiles sobre un conjunto grande de datos.

Destacamos que esta perspectiva aún flaquea en al menos dos cuestiones principales que no queda claro cómo abordar: cuando los datos que podemos distorsionar de los particulares están correlados, y cuando entra en juego el concepto de *privacidad de grupo*.

En el primer caso hablamos de protección contra un atacante que posea información, no ya de otras bases de datos, sino de las relaciones existentes entre los datos que pretende inferir. En el segundo, aunque entran en juego otras cuestiones (éticas, morales, judiciales), la Privacidad Diferencial no tendría mucho que poder hacer tal y como está planteada aquí.

Trabajo futuro

Los aspectos a estudiar del concepto de privacidad son muy numerosos. Ofrecemos una lista con varios temas en los que la DP debe ser tratada matemáticamente y no ha sido posible abarcar en este trabajo:

- Un recorrido por los ataques posibles a los mecanismos de privacidad y las contramedidas que se podrían ofrecer frente a ellos, enumerando las principales técnicas existentes. También cabría hablar de la relación entre las protecciones sobre la privacidad y la criptografía o la seguridad informática.

- La aplicación efectiva de los mecanismos de la Privacidad Diferencial necesita una medición clara del tiempo computacional que requieren. En algunos casos esto podría ser un impedimento frente a la exigencia de una velocidad adecuada en Ciencia de Datos.

- Existen técnicas más específicas para la aplicación de la Privacidad Diferencial en la minería de datos, aprendizaje automático, publicación de histogramas, y un amplio rango de casos prácticos sobre el análisis o la inferencia de datos donde entran en juego las particularidades de cada problema.

- En la subsección 1.2.3 estudiamos las propiedades de composición en paralelo cuando separábamos los datos en clases de equivalencia disjuntas. Sin embargo, los procedimientos de *Big Data* implican de forma inevitable situaciones donde lo corriente es el solapamiento de los datos. Una cuestión pendiente sería la ampliación de los teoremas de composición en paralelo a estos casos.

Bibliografía

[1] Abowd, J. M., Alvisi, L., Dwork, C., Kannan, S., Machanavajjhala, A., and Reiter, J. P. (2017). Privacy-preserving data analysis for the federal statistical agencies. *CoRR*, abs/1701.00752.

[2] Barbaro, M. and Zeller, T. (2006). A face is exposed for aol searcher no. 4417749.

[3] Barth-Jones, D. (2012). The 're-identificationóf governor william weld's medical information: A critical re-examination of health data identification risks and privacy protections, then and now.

[4] Chen K., C. K. (2016). A note on rank reduction in sparse multivariate regression. *Journal of statistical theory and practice*, pages 100–120.

[5] Dalenius, T. (1977). Towards a methodology for statistical disclosure control. *Statistik Tidskrift*, 15:2–1.

[6] Duhigg, C. (2012). How companies learn your secrets.

[7] Dwork, C. and Roth, A. (2014). The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9:211–407.

[8] Dwork, C. and Yekhanin, S. (2008). New efficient attacks on statistical disclosure control mechanisms. In *Advances in Cryptology—CRYPTO 2008*, volume 5157, pages 469–480. Springer Verlag.

[9] Homer, N., Szelinger, S., Redman, M., Duggan, D., Tembe, W., Muehling, J., Pearson, J. V., Stephan, D. A., Nelson, S. F., and Craig, D. W. (2008). Resolving individuals contributing trace amounts of dna to highly complex mixtures using high-density snp genotyping microarrays. *PLOS Genetics*, 4(8):1–9.

[10] Kasiviswanathan, S. P. and Smith, A. D. (2008). A note on differential privacy: Defining resistance to arbitrary side information. *CoRR*, abs/0803.3946.

[11] Lee, J. and Clifton, C. W. (2014). Top-k frequent itemsets via differentially private fp-trees. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 931–940, New York, NY, USA. ACM.

[12] Li, N., Lyu, M., Su, D., and Yang, W. (2016). Differential privacy: From theory to practice. *Synthesis Lectures on Information Security, Privacy, and Trust*, 8(4):1–138.

[13] McSherry, F. and Talwar, K. (2007). Mechanism design via differential privacy. In *Foundations of Computer Science, 2007. FOCS '07. 48th Annual IEEE Symposium on*, pages 94–103.

[14] Narayanan, A. and Shmatikov, V. (2006). How to break anonymity of the netflix prize dataset. *CoRR*, abs/cs/0610105.

[15] Roth, A. (2011). Lecture notes for "the algorithmic foundations of data privacy".

[16] Samarati, P. (2001). Protecting respondents identities in microdata release. *IEEE Transactions on Knowledge and Data Engineering*, 13(6):1010–1027.

[17] Stoddard, B., Chen, Y., and Machanavajjhala, A. (2014). Differentially private algorithms for empirical machine learning. *CoRR*, abs/1411.5428.