Benemérita Universidad Autónoma de Puebla

Facultad de Ciencias Físico Matemáticas

Técnicas de Privacidad Diferencial Aplicadas a Aprendizaje de Máquina y a Modelos de Entrenamiento Descentralizado

Tesis presentada al

Posgrado en Matemáticas

como requisito para la obtención del grado de

Maestro en Ciencias Matemáticas

por

Lic. Víctor Manuel Ortiz Rosas

Asesorado por

Dr. Hugo Adán Cruz Suárez

Puebla Pue.
Enero de 2025

Benemérita Universidad Autónoma de Puebla

Facultad de Ciencias Físico Matemáticas

Técnicas de Privacidad Diferencial Aplicadas a Aprendizaje de Máquina y a Modelos de Entrenamiento Descentralizado

Tesis presentada al

Posgrado en Matemáticas

como requisito para la obtención del grado de

Maestro en Ciencias Matemáticas

por

Lic. Víctor Manuel Ortiz Rosas

Asesorado por

Dr. Hugo Adán Cruz Suárez

Puebla Pue.
Enero de 2025

Título: Técnicas de Privacidad Diferencial Aplicadas a Aprendizaje de Máquina y a Modelos de Entrenamiento Descentralizado

Estudiante: Lic. Víctor Manuel Ortiz Rosas

COMITÉ

Dr. Víctor Hugo Vázquez Guevara
Presidente

Dr. José Juan Castro Alva
Secretario

Dr. Carlos Camilo Garay
Vocal

Dr. Francisco Solano Tajonar Sanabria
Vocal

Dr. Hugo Adán Cruz Suárez
Asesor

DR. SEVERINO MUÑOZ AGUIRRE
SECRETARIO DE INVESTIGACIÓN Y
ESTUDIOS DE POSGRADO, FCFM-BUAP
P R E S E N T E:

Por este medio le informo que el C:

VÍCTOR MANUEL ORTIZ ROSAS

estudiante de la Maestría en Ciencias (Matemáticas), ha cumplido con las indicaciones que el Jurado le señaló en el Coloquio que se realizó el día 10 de diciembre de 2024, con la tesis titulada:

Técnicas de privacidad diferencial aplicadas a aprendizaje de máquina y a modelos de entrenamiento descentralizado

Por lo que se le autoriza a proceder con los trámites y realizar el examen de grado en la fecha que se le asigne.

A T E N T A M E N T E.
H. Puebla de Z. a 22 de enero de 2025

DR. RAÚL ESCOBEDO CONDE
COORDINADOR DEL POSGRADO
EN MATEMÁTICAS.

Facultad
de Ciencias
Físico Matemáticas

Av. San Claudio y 18 sur, edif. FM1
Ciudad Universitaria, Col. San
Manuel, Puebla, Pue. C.P. 72570
(222) 229 55 00 Ext. 7550 y 7552

Agradecimientos

El cierre de esta etapa en mi vida no habría sido posible sin el apoyo incondicional de mi familia. Agradezco profundamente a mis hermanas, Marina, Araceli y Miriam, por la confianza que siempre han depositado en mí. A mi padre, Manuel Ortiz Nieto, le expreso mi gratitud por las enseñanzas que me ha brindado a lo largo de mi vida. En especial, agradezco a mi madre, Ma. Antonia Rosas Jiménez, por su amor, apoyo y constante atención; su confianza en mí ha sido un pilar fundamental en cada momento, y sin ella, nada de esto habría sido posible.

A mis amigos y compañeros de FCFM y FacMed, gracias por los momentos compartidos y las experiencias vividas, por las horas en clases y en las instalaciones de la universidad. Su compañía hizo de esta experiencia algo único y especial.

A todas aquellas personas que conocí durante congresos, eventos académicos y otros encuentros a lo largo de este camino, muchas gracias. Algunos de ustedes se convirtieron en grandes amigos, y, donde sea que estén, quiero que sepan que guardo con cariño los momentos compartidos.

A mi jurado, Dr. Víctor Hugo Vázquez Guevara, Dr. José Juan Castro Alva, Dr. Carlos Camilo Garay y Dr. Francisco Solano Tajonar Sanabria, les agradezco por sus valiosas observaciones y retroalimentación, las cuales fueron fundamentales para mejorar este trabajo.

Al Consejo Nacional de Humanidades, Ciencia y Tecnología (CONAHCYT), mi sincero agradecimiento por el apoyo económico brindado durante este periodo.

Finalmente, expreso mi profunda gratitud al Dr. Hugo Adán Cruz Suárez, no solo por aceptar dirigir este trabajo, sino también por sus enseñanzas y consejos. Su guía ha sido esencial para la culminación de este proyecto.

A todos ustedes, muchas gracias.

Índice general

Índice de figuras

Índice de tablas

Resumen

El presente trabajo expone una serie de metodologías alrededor de la Privacidad Diferencial, se presentan mecanismos clásicos y se presentan aplicaciones tanto en el cálculo de estadísticas, así como en el desarrollo de modelos de Aprendizaje de Máquina tales como modelos predictivos o de clasificación.
Además de esto, se revisa el concepto de Aprendizaje Federado como un modelo de entrenamiento de modelos de Aprendizaje desde un enfoque descentralizado que complementado con Privacidad Diferencial presenta una opción bastante segura para el entrenamiento de modelos sin comprometer los datos de entrenamiento.

Palabras clave: *Aprendizaje de Máquina, Privacidad Diferencial, Aprendizaje Federado, Clasificación, Regresión.*

Introducción

En 1948, Shannon publicó una teoría matemática de la comunicación [35]. Su primera versión apareció en el Bell System Technical Journal de octubre de 1948, cuando Shannon habla de información, se trata para él de una unidad cuantificable que no tiene en cuenta el contenido del mensaje. Shannon trató de establecer a través de esta teoría una ecuación matemática para poder medir el valor informativo de los mensajes, tomando en consideración la información como un valor cuantificable en los procesos de comunicación. A partir de este avance, desde la década 1950 hasta inicio del siglo actual se ha venido trabajando en la comunicación, procesamiento y almacenamiento de datos, en particular datos asociados a personas.

Los algoritmos de privacidad más usados han sido k-anonimato [36], que consiste en que un individuo no debe ser identificable dentro de un grupo de tamaño menor que k. Otros modelos de privacidad existentes, han sido l-diversidad [25], t-cercanía [22] y δ-presencia [31]. Sin embargo, todos estos modelos pueden ser vulnerables bajo situaciones de acceso a información privilegiada.

En 2006 Dwork, McSherry, Nissim y Smith introdujeron el concepto de Privacidad ε-Diferencial (DP) [11], una definición matemática para la pérdida de privacidad asociada con cualquier publicación de datos extraídos de una base de datos estadística. Entendemos a una base de datos estadística como un conjunto de datos que se recopilan bajo compromiso de confidencialidad con el fin de producir estadísticas que, por su producción, no comprometan la privacidad de las personas que proporcionaron los datos. La privacidad diferencial garantiza que la inclusión o exclusión de un individuo en un conjunto de datos no afecte significativamente los resultados de un análisis. Esto se logra añadiendo ruido aleatorio a los datos, limitando la cantidad de información que se puede obtener sobre cualquier individuo.

El uso de privacidad diferencial actualmente ayudó a solucionar problemas de salud pública para prevenir la dispersión de enfermedades como el COVID-19 durante 2020, a través de datos recopilados por los teléfonos sin violar la privacidad de cada individuo [18]. También gigantes de la tecnología como Apple, Google y Microsoft han visto la necesidad de implementar privacidad diferencial al recopilar información, pues al recuperar datos biológicos a través de los dispositivos electrónicos de sus usuarios [27], si se da una violación que revele esta información resultaría en un grave problema.

La idea de privacidad diferencial se ha desarrollado al punto de ser aplicable al Internet de las cosas y modelos de Aprendizaje de Máquina [1, 44], evitando que los

modelos revelen información sobre los datos con los cuales fueron entrenados. Sin embargo, la privacidad diferencial sigue teniendo sus limitantes, pues en ocasiones se abusa de ella al grado de modificar tanto los datos que los hace inutilizables.

La relevancia de la Privacidad Diferencial está en que tiene una garantía teórica. Esto es fundamental, ya que una técnica de privacidad es más efectiva cuando cumple con garantías teóricas. Si publicamos los datos y posteriormente identificamos un problema, el daño ya estaría hecho, dado que los datos habrían sido difundidos.

Este trabajo aborda la aplicación de la Privacidad Diferencial en el entrenamiento de modelos de Aprendizaje Automático [2], con el objetivo de mejorar la protección de la información sensible durante el proceso de modelado. En primer lugar, se exploran diferentes técnicas de Aprendizaje Automático aplicadas a conjuntos de datos de Salud Pública, destacando cómo estos modelos pueden ser entrenados de manera efectiva mientras se asegura la privacidad de los datos utilizados. Posteriormente, se profundiza en el uso del Aprendizaje Federado [3], una metodología que permite entrenar modelos de manera distribuida sin la necesidad de centralizar los datos, lo que agrega una capa adicional de privacidad. En este contexto, se aplica el enfoque de Aprendizaje Federado con Privacidad Diferencial a una aplicación específica en la Ciudad de México, centrada en información de deuda pública, con el objetivo de entrenar un modelo de Regresión Lineal, pero sin la necesidad de compilar diferentes conjuntos de datos en uno solo. En resumen, este trabajo presenta diferentes metodologías donde se aplica la Privacidad Diferencial y en cada una de ellas se ejemplifica usando bases de datos reales, lo cual representa una contribución a la literatura existente sobre Privacidad Diferencial. Más aún, en lo referente a Aprendizaje Federado se revisaron 3 técnicas recientes y además se realizó un análisis comparativo de estas técnicas. La conclusión de este análisis es que en un entorno descentralizado aún se pueden entrenar modelos de aprendizaje sin comprometer su exactitud.

La tesis está dividida en cuatro capítulos del siguiente modo. En el Capítulo 1 se presenta la teoría básica de la Privacidad Diferencial, en éste se incluyen resultados básicos y conocidos referentes a esta teoría. En el Capítulo 2 se enuncian los Métodos Clásicos de Privacidad Diferencial y se dan ejemplos para el cálculo de estadísticas. Luego, en el Capítulo 3 se presenta cómo se aplica la Privacidad Diferencial al Aprendizaje de Máquina, del mismo modo se muestran resultados de aplicaciones de los modelos expuestos. Finalmente, en el Capítulo 4 se explora el Aprendizaje Federado mediante diferentes enfoques junto con la Privacidad Diferencial.

Capítulo 1

Preliminares

En este capítulo iniciaremos describiendo qué entendemos por una base de datos en términos de elementos aleatorios, además de la definición de mecanismo de aleatorización y lo que entendemos por privacidad diferencial.

Definición 1.1. *Sea $\mathbf{X}_1, \ldots, \mathbf{X}_n$ una muestra de elementos aleatorios de una distribución común y definidas en un espacio de probabilidad común $(\Omega, \mathcal{F}, P)$, esto es, para cada $i = 1, \ldots, n$, $\mathbf{X}_i : \Omega \to (B, \mathcal{F}_B)$ es una función medible y $(B, \mathcal{F}_B)$ es un espacio medible. Observe que las $\mathbf{X}_i$ pueden ser vectores del mismo tamaño $\mathbf{X}_i = (X_{i1}, \ldots X_{ir})$ con r un entero positivo fijo para $i = 1, \ldots, n$. En este caso llamamos a $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_n)$ una base de datos con n registros y r atributos. Denotaremos por $\mathcal{D}$ al conjunto de los posibles valores de $\mathbf{X}$, esto es, $\mathcal{D} = B^n$.*

El objetivo de la privacidad diferencial es proteger la información de cada individuo en una base de datos, pero permitir realizar un análisis estadístico de la base de datos en conjunto. Para entenderlo mejor, supongamos que queremos hacer público un dato estadístico acerca de una base de datos $D \in \mathcal{D}$ (promedio, mediana, cantidad de individuos que cumplen cierta condición, etcétera) al que llamaremos $f(D)$ y existe un individuo del cual sus datos están en D y está preocupado acerca de si $f(D)$ viola su privacidad. Entonces podríamos tomar una base alternativa D' que no contenga la información de este individuo (quitando o remplazando este registro) y hacer público $f(D')$. Ahora bien, este método es inviable, ya que, si todos los individuos no quieren que su información aparezca, nos quedaremos sin datos y sin poder publicar algo. Así, lo que pretende la privacidad diferencial es que podamos aproximar el efecto de tener o no tener un individuo en la base de datos asegurando que $f(D)$ y $f(D')$ sean aleatorios, pero tomarán el mismo valor con una probabilidad muy similar.

Definición 1.2. *Una petición (query) es una función f que asocia una base de datos $D \in \mathcal{D}$ a un elemento de un rango abstracto C.*

Ejemplo 1.1. *Las siguiente pueden ser consideradas peticiones sobre una base de datos:*

- $f_1(D) =$ *1er atributo de la base de datos.*
- $f_2(D) =$ *2do registro de la base de datos.*
- $f_3(D) =$ *Cantidad de individuos en D con cierta propiedad.*
- $f_4(D) =$ *Valor del 2do atributo del 3er registro en la base de datos.*
- $f_5(D) = D$ *(Solicitar la propia base de datos).*

  Ahora, suponiendo que la base de datos tiene únicamente un atributo numérico podemos considerar otro tipo de peticiones como:

- $f_6(D) = promedio(D)$.
- $f_7(D) = mediana(D)$.
- $f_8(D) = Q_1(D)$ *(Primer cuantil de los datos).*

En los ejemplos anteriores podemos identificar que C, el rango de la petición puede ser muy diverso, pues puede cambiar entre, $\mathbb{R}$, $\mathbb{Z}$, el conjunto de todas las bases de datos, la propia base de datos o el rango de cierto atributo (pensando en variables categóricas). De aquí que lo tomemos como un rango abstracto por ahora.

Nos centramos en los mecanismos que toman como entrada una base de datos $\mathbf{X}$ y dan como salida otra muestra aleatoria $\mathcal{M} = (\mathbf{Y}_1, \ldots, \mathbf{Y}_m)$ (no necesariamente con el mismo rango que las $\mathbf{X}_i$) donde similarmente cada $\mathbf{Y}_j$ es definida en el espacio de probabilidad común $(\Omega_1, \mathcal{F}_1, \mathbb{P})$, esto es, para cada $j = 1, \ldots, m$, se tiene que $\mathbf{Y}_j : \Omega_1 \to (C, \mathcal{F}_C)$ es una función medible y $(C, \mathcal{F}_C)$ es un espacio medible. Note que las $\mathbf{Y}_j$ pueden ser vectores del mismo tamaño $\mathbf{Y}_j = (Y_{j1}, \ldots Y_{js})$ con s un entero positivo fijo para $j = 1, \ldots, m$.

Por ejemplo, si nuestra base de datos consta de 2 atributos (peso y edad), entonces $B = [0, 200] \times \{0, 1, \ldots, 120\}$ y si nuestro mecanismo de interés es el cálculo del IMC, entonces $C = [0, 50]$, aunque si nos interesara determinar categoría del IMC, podríamos tomar $C = \{\text{Delgadez, Saludable, Sobrepeso}\}$.

Definición 1.3. *Diremos que dos bases de datos D y D' en $\mathcal{D}$ son adyacentes (o vecinas) si difieren en un registro. Denotamos este hecho como $D \simeq D'$.*

Definición 1.4. *Un mecanismo de aleatorización $Q(\cdot|X)$ es una distribución condicional para $\mathcal{M}$ dado $\mathbf{X}$. Así, $Q(S|\mathbf{X} = D)$ es la probabilidad de que la salida $\mathcal{M}$ esté en el conjunto $S \in \mathcal{F}_C$ dado que la entrada es $D \in \mathcal{D} = B^n$ [38].*

Definición 1.5. *Sea $\epsilon > 0$ decimos que Q satisface ϵ-privacidad diferencial si para cualquier par de bases adyacentes D y D' y cualquier $S \in \mathcal{F}_C$ tenemos:*

$$\frac{Q(S|\mathbf{X} = D)}{Q(S|\mathbf{X} = D')} \leq e^{\epsilon}. \tag{1.1}$$

En donde se asume que $\frac{0}{0} = 1$.

Observación 1.1. *$Q(S|\mathbf{X} = D)$ es la probabilidad de que la salida $\mathcal{M}$ esté en el conjunto $S \in \mathcal{F}_C$ dado que la entrada es $D \in B^n$, esto es, $Q(S|\mathbf{X} = D) = \mathbb{P}(\mathcal{M} \in S|\mathbf{X} = D)$ y denotaremos la probabilidad anterior por $\mathbb{P}(\mathcal{M}(D) \in S)$. Así, la condición en (1.1) es*

$$\frac{\mathbb{P}(\mathcal{M}(D) \in S)}{\mathbb{P}(\mathcal{M}(D') \in S)} \leq e^{\epsilon}. \tag{1.2}$$

Cuando un mecanismo satisfaga ϵ-privacidad diferencial diremos que es ϵ-DP o que satisface ϵ-DP. La Definición 1.5 nos indica que cambiar un registro en la base de datos no debería hacer un cambio muy grande en la distribución de probabilidad de la salida.

Anteriormente, se mencionó que el objetivo es evitar que la salida de un algoritmo dependa demasiado de un individuo en particular, haciendo así que la respuesta sea muy similar cuando se encuentre o no algún individuo. Esto queda controlado por ϵ, cuanto más cerca e^{ϵ} está de 1, la garantía de privacidad es más fuerte. Por lo tanto, normalmente se elige ϵ cercano a 0, aunque esto va a depender de cada problema en cuestión [11].

Cuando la distribución de $\mathcal{M}(D)$ tenga a q_D como función de densidad o de masa según sea el caso, podemos demostrar que se cumple la ϵ-DP en términos de q_D, pues si para todo $t \in C$ se tiene que $\frac{q_D(t)}{q_{D'}(t)} \leq e^{\epsilon}$ entonces se cumple la condición de (1.2) como se ve en el siguiente resultado.

Teorema 1.1. *Sea q_D la función de densidad (o masa) asociada a la distribución de $\mathcal{M}(D)$, si $\frac{q_D(t)}{q_{D'}(t)} \leq e^{\epsilon}$ para todo $t \in C$, entonces el algoritmo satisface ϵ-DP.*

Demostración. Sea $S \in \mathcal{F}_C$, si q_D es la función de densidad, entonces

$$\begin{aligned}
\mathbb{P}[\mathcal{M}(D) \in S] &= \int_S q_D(t)dt \\
&\leq \int_S e^{\epsilon} q_{D'}(t)dt \\
&\leq e^{\epsilon} \int_S q_{D'}(t)dt \\
&\leq e^{\epsilon} Pr[\mathcal{M}(D') \in S].
\end{aligned}$$

Similarmente cuando q_D es la función de masa, entonces

$$\begin{aligned}
\mathbb{P}[\mathcal{M}(D) \in S] &= \sum_{t \in S} q_D(t) \\
&\leq \sum_{t \in S} e^{\epsilon} q_{D'}(t) \\
&\leq e^{\epsilon} \sum_{t \in S} q_{D'}(t) \\
&\leq e^{\epsilon} Pr[\mathcal{M}(D') \in S].
\end{aligned}$$

Con lo que se demuestra el resultado. □

Ahora con la privacidad diferencial queremos que, al hacer una petición sobre dos bases de datos adyacentes, la diferencia sea lo suficientemente pequeña para ser aceptable. Visto así, lo que queremos es que dada una petición (o conjunto de peticiones) podamos encontrar un mecanismo de privacidad asociado que cumpla con un nivel de privacidad diferencial y que además proporcione información útil que responda a las peticiones.

Veamos el siguiente ejemplo en donde la salida es nuevamente una base de datos.

Ejemplo 1.2. *Mecanismo básico de respuestas: situémonos en un escenario en el que queremos recopilar respuestas sobre una pregunta comprometedora en la que únicamente podemos responder sí o no (1 y 0 respectivamente). Para no comprometer la privacidad de las personas encuestadas les pedimos que realicen el siguiente procedimiento:*

1. *Lanzar una moneda*
2. *Si cae cara, entonces responde de manera honesta.*
3. *De lo contrario, vuelve a lanzarla.*
4. *Si en este segundo lanzamiento sale cara, responde que sí, de lo contrario responde que no.*

A este proceso le llamaremos el algoritmo $\mathcal{A}$.
Nota: El encuestador no sabe cuántas veces el individuo lanzó la moneda. De este modo, si un individuo responde que sí, puede excusarse en que fue por las monedas.

Podemos representar el proceso anterior mediante el Algoritmo 1.

Algoritmo 1 Algoritmo básico

Entrada Base de datos $D = \{X_1, \ldots, X_n\}$
Salida Base de datos $\tilde{D} = \{Y_1, \ldots, Y_n\}$ modificada

for k=1 hasta n **do**
  $R = Ber\left(\frac{1}{2}\right)$ #generamos un valor aleatorio 0 o 1
  if R=1 **then**
    $Y_k = X_k$
  else
    $R_1 = Ber\left(\frac{1}{2}\right)$ #generamos otro valor aleatorio 0 o 1
    $Y_k = R_1$
  end if
end for
return $\tilde{D} = \{Y_1, \ldots, Y_n\}$

Teorema 1.2. *El Algoritmo 1 proporciona $ln(3)$-DP.*

Demostración. Sean $D = \{d_1, \ldots, d_n\}$, $D' = \{d'_1, \ldots, d'_n\} \in \{0,1\}^n$ bases de datos adyacentes, sin pérdida de generalidad podemos suponer que $d_1 \neq d'_1$. Sean $\mathcal{A}(D) = \{Y(d_1), \ldots, Y(d_n)\}$ y $\mathcal{A}(D') = \{Y(d'_1), \ldots, Y(d'_n)\}$. Tenemos que calcular $\frac{\mathbb{P}[\mathcal{A}(D)\in S]}{\mathbb{P}[\mathcal{A}(D')\in S]}$ pero por el tipo de dato basta con calcular $\frac{\mathbb{P}[\mathcal{A}(D)=s]}{\mathbb{P}[\mathcal{A}(D')=s]}$ esto para todo $s = (s_1, \ldots, s_n) \in Ran\mathcal{A} = \{0,1\}^n$. Entonces:

$$\begin{aligned}
\frac{\mathbb{P}[\mathcal{A}(D) = s]}{\mathbb{P}[\mathcal{A}(D') = s]} &= \frac{\mathbb{P}[(Y(d_1), Y(d_2), \ldots, Y(d_n)) = (s_1, s_2, \ldots, s_n)]}{\mathbb{P}[(Y(d'_1), Y(d'_2), \ldots, Y(d'_n)) = (s_1, s_2, \ldots, s_n)]} \\
&= \frac{\mathbb{P}[Y(d_1) = s_1, Y(d_2) = s_2, \ldots, Y(d_n) = s_n]}{\mathbb{P}[Y(d'_1) = s_1, Y(d'_2) = s_2, \ldots, Y(d'_n) = s_n]} \\
&= \frac{\mathbb{P}[Y(d_1) = s_1]\mathbb{P}[Y(d_2) = s_2] \ldots \mathbb{P}[Y(d_n) = s_n]}{\mathbb{P}[Y(d'_1) = s_1]\mathbb{P}[Y(d'_2) = s_2] \ldots \mathbb{P}[Y(d'_n) = s_n]} \\
&= \frac{\mathbb{P}[Y = s_1|X_1 = d_1]\mathbb{P}[Y = s_2|X_2 = d_2] \ldots \mathbb{P}[Y = s_n|X_n = d_n]}{\mathbb{P}[Y = s_1|X_1 = d'_1]\mathbb{P}[Y = s_2|X_2 = d'_2] \ldots \mathbb{P}[Y = s_n|X_n = d'_n]} \\
&= \frac{\mathbb{P}[Y = s_1|X_1 = d_1]\mathbb{P}[Y = s_2|X_2 = d_2] \ldots \mathbb{P}[Y = s_n|X_n = d_n]}{\mathbb{P}[Y = s_1|X_1 = d'_1]\mathbb{P}[Y = s_2|X_2 = d_2] \ldots \mathbb{P}[Y = s_n|X_n = d_n]} \\
&= \frac{\mathbb{P}[Y = s_1|X_1 = d_1]}{\mathbb{P}[Y = s_1|X_1 = d'_1]}.
\end{aligned}$$

Observe que las entradas $Y(d_i)$, $Y(d'_i)$ corresponde a la transformación de los datos reales d_i y d'_i por el Algoritmo 1, respectivamente.
Ahora, observe que tenemos dos casos:

- $s_1 = 1$:

$$\frac{\mathbb{P}[Y = 1|d_1 = 1]}{\mathbb{P}[Y = 1|d'_1 = 0]} = \frac{3/4}{1/4} = 3 \quad \text{y} \quad \frac{\mathbb{P}[Y = 1|d_1 = 0]}{\mathbb{P}[Y = 1|d'_1 = 1]} = \frac{1/4}{3/4} = \frac{1}{3}$$

- $s_1 = 0$:

$$\frac{\mathbb{P}[Y = 0|d_1 = 1]}{\mathbb{P}[Y = 0|d'_1 = 0]} = \frac{1/4}{3/4} = \frac{1}{3} \quad \text{y} \quad \frac{\mathbb{P}[Y = 0|d_1 = 0]}{\mathbb{P}[Y = 0|d'_1 = 1]} = \frac{3/4}{1/4} = 3$$

Luego, $\frac{\mathbb{P}[\mathcal{A}(D)=s]}{\mathbb{P}[\mathcal{A}(D')=s]} \leq 3 = e^{\ln(3)}$ y así $\frac{\mathbb{P}[\mathcal{A}(D)\in S]}{\mathbb{P}[\mathcal{A}(D')\in S]} \leq 3 = e^{\ln(3)}$. Por lo tanto, se cumple la Definición 1.5. □

Este mecanismo existe incluso antes de que estuviera bien definida la DP [32].

Observación 1.2. *Este mecanismo es bastante usado ya que podemos recuperar la proporción de datos reales de la siguiente forma:*
Nosotros quisiéramos estimar $\rho = \mathbb{P}(Y = 1|R = 1)$, Y es una variable aleatoria referente a la respuesta modificada de algún individuo y R es el resultado del primer

lanzamiento de la moneda, con 1 siendo sí y 0 siendo no.
Por teorema de la probabilidad total sabemos lo siguiente:

$$\mathbb{P}(Y=1)=\mathbb{P}(Y=1|R=1)\mathbb{P}(R=1)+\mathbb{P}(Y=1|R=0)\mathbb{P}(R=0). \tag{1.3}$$

Asumiendo que después de la encuesta, la proporción de respuestas positivas es s*, podemos tomar esta proporción como* $\mathbb{P}(Y=1)$*. Además, note que* $\mathbb{P}(Y=1|R=0)=1/2$ *y también* $\mathbb{P}(R=1)=\mathbb{P}(R=0)=1/2$*. Así que regresando a (1.3) tenemos*

$$s=\mathbb{P}(Y=1|R=1)1/2+(1/2)(1/2),$$

despejando, llegamos a que $\widehat{\rho}=2s-1/2$*, donde* $\widehat{\rho}$ *es la estimación de* $\rho=\mathbb{P}(Y=1|R=1)$ *que corresponde a la proporción de respuestas positivas reales.*

A veces por conveniencia se aplican varios mecanismos de manera secuencial, los siguientes teoremas nos dicen qué pasa con la ϵ-DP. El primer teorema de composición consiste en que al aplicar un primer algoritmo de aleatorización que satisface ϵ-DP a una base de datos D, si a ese resultado aplicamos un segundo algoritmo pero que no tenga acceso a la base de datos D, esa composición satisface ϵ-DP, esto queda enunciado en el siguiente resultado.

Teorema 1.3. ***Post-procesamiento*** *Dado un algoritmo de aleatorización* $\mathcal{A}_1(\cdot)$ *que satisface* ϵ*-DP, y otro mecanismo (no necesariamente aleatorio)* $\mathcal{A}_2$*, entonces la composición de* $\mathcal{A}_1$ *y* $\mathcal{A}_2$*, esto es,* $\mathcal{A}_2(\mathcal{A}_1(\cdot))$ *satisface* ϵ*-DP [23].*

Demostración. Sean D y D' bases de datos adyacentes. Sea $S\subseteq Ran(\mathcal{A}_2)$, observe que:

$$\begin{aligned}\mathbb{P}[\mathcal{A}_2(\mathcal{A}_1(D))\in S]&=\mathbb{P}[\mathcal{A}_1(D)\in\{r\in Ran(\mathcal{A}_1):\mathcal{A}_2(r)\in S\}]\\&\leq e^{\epsilon}\mathbb{P}[\mathcal{A}_1(D')\in\{r\in Ran(\mathcal{A}_1):\mathcal{A}_2(r)\in S\}]\\&=e^{\epsilon}\mathbb{P}[\mathcal{A}_2(\mathcal{A}_1(D'))\in S].\end{aligned}$$

□

Como se mencionó, en el resultado anterior, el algoritmo de posprocesamiento $\mathcal{A}_2$ accede solo a la salida de $\mathcal{A}_1$ y no al conjunto de datos de entrada D. La importancia de este teorema está en que, aunque hagamos un proceso cualquiera a un algoritmo que satisface ϵ-DP no se puede disminuir el parámetro de privacidad ϵ, esto significa en cierto modo que cualquier analista de datos sin conocimiento previo de D no puede hacer menos privado a un algoritmo que satisface ϵ-DP, haciendo así que no se vea vulnerada la información en D.
El siguiente teorema de composición, a diferencia del anterior, establece que uno de los algoritmos toma como entrada no sólo la respuesta del algoritmo anterior sino también la propia base de datos.

Teorema 1.4. *Dado un mecanismo de aleatorización $\mathcal{A}_1(\cdot) : \mathcal{D} \to O$ que satisface ϵ_1-DP, y otro mecanismo de aleatorización $\mathcal{A}_2(s, \cdot) : O \times \mathcal{D} \to R$ que satisface ϵ_2-DP para cualquier s, donde O y R son conjuntos no vacíos entonces $\mathcal{A}(D) = (\mathcal{A}_1(D), \mathcal{A}_2(\mathcal{A}_1(D), D))$ satisface $(\epsilon_1 + \epsilon_2)$-DP.*

Demostración. Sean D y D' bases de datos adyacentes. Sea $S \subseteq O \times R$, si $\mathcal{A}_1$ es discreta observe que usando el teorema de la probabilidad total tenemos:

$$\begin{aligned}
&\mathbb{P}[(\mathcal{A}_1(D), \mathcal{A}_2(\mathcal{A}_1(D), D)) \in S] \\
&= \sum_{r \in Ran(\mathcal{A}_1)} \mathbb{P}[(\mathcal{A}_1(D), \mathcal{A}_2(\mathcal{A}_1(D), D)) \in S | \mathcal{A}_1(D) = r] \mathbb{P}(\mathcal{A}_1(D) = r) \\
&= \sum_{r \in Ran(\mathcal{A}_1)} \mathbb{P}[(r, \mathcal{A}_2(r, D)) \in S] \mathbb{P}(\mathcal{A}_1(D) = r) \\
&\leq \sum_{r \in Ran(\mathcal{A}_1)} e^{\epsilon_2} \mathbb{P}[(r, \mathcal{A}_2(r, D')) \in S] e^{\epsilon_1} \mathbb{P}(\mathcal{A}_1(D') = r) \\
&= \sum_{r \in Ran(\mathcal{A}_1)} e^{\epsilon_1 + \epsilon_2} \mathbb{P}[(r, \mathcal{A}_2(r, D')) \in S] \mathbb{P}(\mathcal{A}_1(D') = r) \\
&= e^{\epsilon_1 + \epsilon_2} \sum_{r \in Ran(\mathcal{A}_1)} \mathbb{P}[(\mathcal{A}_1(D'), \mathcal{A}_2(\mathcal{A}_1(D'), D')) \in S | \mathcal{A}_1(D') = r] \mathbb{P}(\mathcal{A}_1(D') = r) \\
&= e^{\epsilon_1 + \epsilon_2} \mathbb{P}[(\mathcal{A}_1(D'), \mathcal{A}_2(\mathcal{A}_1(D'), D')) \in S].
\end{aligned}$$

Ahora, similarmente para el caso en que $\mathcal{A}_1(D)$ es continua usaremos su función de densidad q_D y de manera análoga usando el teorema de la probabilidad total para el caso continuo tenemos:

$$\begin{aligned}
&\mathbb{P}[(\mathcal{A}_1(D), \mathcal{A}_2(\mathcal{A}_1(D), D)) \in S] \\
&= \int_{Ran(\mathcal{A}_1)} \mathbb{P}[(\mathcal{A}_1(D), \mathcal{A}_2(\mathcal{A}_1(D), D)) \in S | \mathcal{A}_1(D) = r] q_D(r) dr \\
&= \int_{Ran(\mathcal{A}_1)} \mathbb{P}[(r, \mathcal{A}_2(r, D)) \in S] q_D(r) dr \\
&\leq \int_{Ran(\mathcal{A}_1)} e^{\epsilon_2} \mathbb{P}[(r, \mathcal{A}_2(r, D')) \in S] e^{\epsilon_1} q_{D'}(r) dr \\
&= \int_{Ran(\mathcal{A}_1)} e^{\epsilon_1 + \epsilon_2} \mathbb{P}[(r, \mathcal{A}_2(r, D')) \in S] q_{D'}(r) dr \\
&= e^{\epsilon_1 + \epsilon_2} \int_{Ran(\mathcal{A}_1)} \mathbb{P}[(\mathcal{A}_1(D'), \mathcal{A}_2(\mathcal{A}_1(D'), D')) \in S | \mathcal{A}_1(D') = r] q_{D'}(r) dr \\
&= e^{\epsilon_1 + \epsilon_2} \mathbb{P}[(\mathcal{A}_1(D'), \mathcal{A}_2(\mathcal{A}_1(D'), D')) \in S].
\end{aligned}$$

Por lo tanto, el resultado es válido.

□

El teorema anterior puede ser generalizado por inducción a la composición de varios mecanismos que pueden recibir como entrada tanto a la base de datos, como también las respuestas de mecanismos anteriores.

Teorema 1.5. *Sean $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_k$ algoritmos de aleatorización (que pueden tomar entradas auxiliares) que cumplen ϵ_1-DP, ϵ_2-DP,..., ϵ_k-DP respectivamente. Entonces el algoritmo $\mathcal{A}(D) = \mathbf{t} = (t_1, t_2, \ldots, t_k)$, donde $t_1 = \mathcal{A}_1(D)$, $t_2 = \mathcal{A}_2(t_1, D)$, ..., $t_k = \mathcal{A}_k(t_1, t_2, \ldots, t_{k-1}, D)$ satisface $(\epsilon_1 + \epsilon_2 + \cdots + \epsilon_k)$-DP.*

De este modo, se ha presentado la teoría básica de la Privacidad Diferencial. En el siguiente capítulo se introducirán los principales mecanismos que se utilizarán posteriormente, así como algunas de sus aplicaciones.

Capítulo 2

Métodos Clásicos

En esta sección se presentan los métodos más comunes dentro de la Privacidad Diferencial, estos métodos forman la base para algoritmos que se verán más adelante. Para la realización de los ejemplos de aplicaciones en este capítulo se consultó una base de datos del Instituto Nacional de Estadística y Geografía (INEGI) referente a Estadísticas de Defunciones Registradas en 2021 [17] la cual puede consultarse en `https://www.inegi.org.mx/programas/mortalidad/`. Además, las simulaciones y gráficas contenidas fueron obtenidas a través del uso del software RStudio.

2.1. Método de Laplace

El método de Laplace funciona cuando queremos responder a peticiones del tipo numérico, es decir, peticiones f que toman valores en $\mathbb{R}$, para este caso pareciera natural el agregar ruido al valor de $f(D)$, es decir, si $f(D)$ es un número real, entonces en lugar de publicar $f(D)$ que se trata del dato verdadero, podemos publicar $\widetilde{f}(D) = f(D) + X$ donde X es una variable aleatoria con cierto tipo de distribución, así, nuestro mecanismo de privacidad es $\mathcal{M}(D) = \widetilde{f}(D) = f(D)+X$. Nuestro trabajo ahora será saber qué distribución es la óptima para X. Bien, pues observemos que se requiere que para un par de bases adyacentes D, D' y $\forall t \in Ran(\mathcal{M})$:

$$\frac{q_D(t)}{q_{D'}(t)} = \frac{f_X(t - f(D))}{f_X(t - f(D'))} \leq e^{\epsilon}$$

en donde q_D y $q_{D'}$ representan la función de densidad para $\mathcal{M}(D)$ y $\mathcal{M}(D')$, respectivamente, y f_X representa la función de densidad de la variable aleatoria X. Si tomamos $d = f(D) - f(D')$ y $x = t - f(D)$ podemos transformar la condición anterior en

$$\forall x \in \mathbb{R} : \frac{f_X(x)}{f_X(x + d)} \leq e^{\varepsilon}. \tag{2.1}$$

Entonces (2.1) establece una condición necesaria para la variable aleatoria X en los posibles valores de d. Sin embargo, es importante destacar que d solo podrá

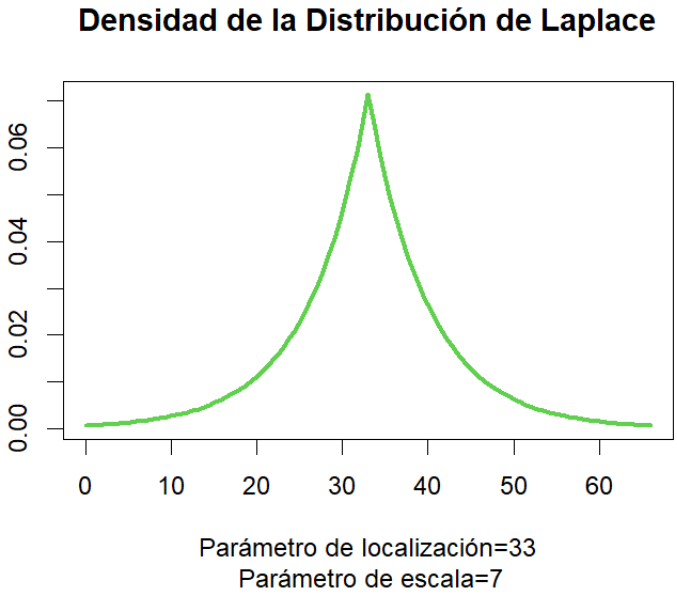


Figura 2.1: Densidad de Laplace.

asumir ciertos valores, lo que implica que el comportamiento de X estará restringido por estas condiciones. Por lo tanto, su análisis deberá enfocarse únicamente en estos casos específicos. Así, antes de analizar qué variable X satisface la condición mencionada, será útil considerar la siguiente definición.

Definición 2.1. *La máxima diferencia en los resultados de una petición sobre bases de datos adyacentes le llamaremos sensibilidad de f y se denota como Δf, es decir:*

$$\Delta f = \max_{D \simeq D'} |f(D) - f(D')|.$$

Así, queremos ver que la condición en (2.1) se cumpla para todo $d \leq \Delta f$, Esto significa que una condición que debe cumplir la variable aleatoria X es que $f_X(x) \leq e^{\epsilon} f_X(x+d)$ para toda x, es decir, si nos movemos una distancia no mayor a Δf, entonces la función de densidad de la variable aleatoria X crece o decrece por un factor no mayor a e^{ε}. Una distribución que satisface esto es la distribución de Laplace con parámetros $(0, \frac{\Delta f}{\epsilon})$ que tiene como función de densidad

$$f(x) = \frac{1}{2\frac{\Delta f}{\epsilon}} exp\left(-\frac{|x|}{\frac{\Delta f}{\epsilon}}\right).$$

En general, para una variable aleatoria con distribución $Lap(a, b)$, al parámetro a le llamamos de localización y a b parámetro de escala, cuando $a = 0$ escribiremos sólo $Lap(b)$. Un ejemplo de la gráfica de esta densidad puede verse en la Figura 2.1.

Ahora, observe que si $X \sim Lap(0, \frac{\Delta f}{\epsilon})$ entonces se puede ver que:

$$\frac{f_X(x)}{f_X(x+d)} \leq e^{\frac{\frac{d}{\Delta f}}{\epsilon}} \leq e^{\frac{\frac{\Delta f}{\Delta f}}{\epsilon}} = e^{\epsilon},$$

es decir que cumple la condición en (2.1).

Veamos formalmente en el siguiente teorema que el mecanismo $\mathcal{M}(D)$ antes mencionado satisface la ϵ-DP.

Teorema 2.1. *Para una petición f el mecanismo de Laplace definido como $\mathcal{M}(D) = f(D) + Lap(0, \frac{\Delta f}{\epsilon})$ satisface ϵ-DP.*

Demostración. Sean D y D' dos bases adyacentes y sea $t \in \mathbb{R}$, entonces

$$\begin{aligned} q_D(t) &= f_X(t - f(D)) \\ &= \frac{1}{2\frac{\Delta f}{\epsilon}} e^{-\frac{|t-f(D)|}{\frac{\Delta f}{\epsilon}}} \\ &= \frac{\epsilon}{2\Delta f} exp(-\frac{\epsilon|t-f(D)|}{\Delta f}). \end{aligned}$$

De manera análoga vamos a tener que $q_{D'}(t) = \frac{\epsilon}{2\Delta f} exp(-\frac{\epsilon|t-f(D')|}{\Delta f})$. Luego,

$$\begin{aligned} \frac{q_D(t)}{q_{D'}(t)} &= \frac{\frac{\epsilon}{2\Delta f} exp(-\frac{\epsilon|t-f(D)|}{\Delta f})}{\frac{\epsilon}{2\Delta f} exp(-\frac{\epsilon|t-f(D')|}{\Delta f})} \\ &= \frac{exp(-\frac{\epsilon|t-f(D)|}{\Delta f})}{exp(-\frac{\epsilon|t-f(D')|}{\Delta f})} \\ &= exp(-\frac{\epsilon|t-f(D)|}{\Delta f} - (-\frac{\epsilon|t-f(D')|}{\Delta f})). \end{aligned}$$

Ahora, usando la desigualdad del triángulo y la definición de Δf tenemos:

$$\begin{aligned} \frac{q_D(t)}{q_{D'}(t)} &= exp(\frac{\epsilon(|t-f(D')| - |t-f(D)|)}{\Delta f}) \\ &\leq exp(\frac{\epsilon(|t-f(D') - (t-f(D))|)}{\Delta f}) \\ &= exp(\frac{\epsilon(|f(D) - f(D'))|)}{\Delta f}) \\ &\leq exp(\frac{\epsilon\Delta f}{\Delta f}) \\ &= e^{\epsilon}. \end{aligned}$$

Lo que demuestra el resultado.

□

Ahora, consideremos una base de datos D y supongamos que nos interesa saber cuántos registros cumplen cierta condición. ¿Cómo podemos responder a esta pregunta satisfaciendo la ϵ-DP? Hay que notar que esta petición tiene sensibilidad 1 (remplazar un registro en la base de datos hace que el conteo sólo se modifique en máximo una unidad), entonces el ruido que tenemos que agregar para que se satisfaga ϵ-DP a la respuesta se trata de una variable aleatoria con distribución $Lap(0, \frac{1}{\epsilon})$. Así que para este tipo de peticiones no influye el tamaño de la base de datos para cumplir la $\epsilon-$DP, sino que sólo debemos fijarnos en cuánta privacidad queremos proveer, es decir, establecer el parámetro ϵ.

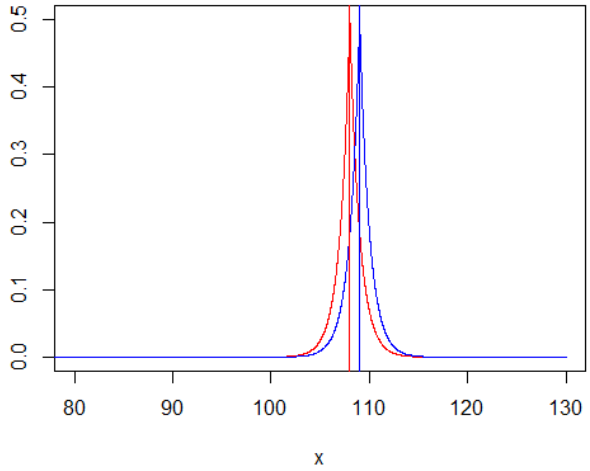
Ejemplo 2.1. *Considerando una base de datos de defunciones, suponiendo que estamos interesados en la cantidad de muertos menores de 18 años por causas relativas a Diabetes, tras la filtración de datos tenemos que esta cantidad es de 108, es decir,* $f(D) = 108$. *Aplicando el mecanismo de Laplace tenemos que el dato a publicar* $\widetilde{f}(D) = f(D) + Lap(0, \frac{1}{\epsilon})$ *debe seguir las distribuciones que se pueden ver en la Figura 2.2 según el nivel* ϵ *de privacidad que queramos proveer.*

Podemos observar en las gráficas de la Figura 2.2 entre más privacidad queramos mantener, haremos que la distribución de $\widetilde{f}(D)$ *se vuelva más platicúrtica abriendo posibilidades a que aparezcan valores muy alejados del valor real* $f(D)$ *como puede verse en las Figuras 2.2a y 2.2b, en donde las líneas verticales representan los valores reales* $f(D)$ *y* $f(D')$. *Esto nos lleva a notar que entre más privacidad queremos mantener, el valor que entregaremos podrá estar más alejado del real, pero a su vez, esto es lo que va a garantizar que las probabilidades de obtener el mismo valor en bases adyacentes sean muy similares. Concluimos que el costo de la privacidad es inverso a la utilidad de los resultados sobre las peticiones a entregar. Entonces, debemos encontrar el equilibrio óptimo entre utilidad de los datos y preservar la privacidad de cada individuo pues como en este caso, publicar información referente a las causas de muerte de algún familiar puede resultar controversial para cierto sector de la población.*

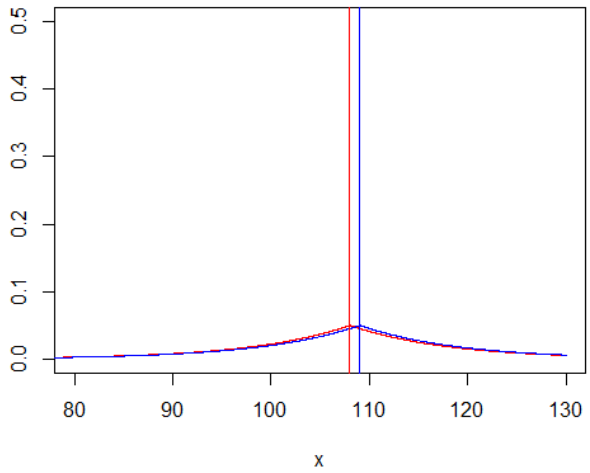
Ejemplo 2.2. *El mecanismo de Laplace es útil también para peticiones del tipo suma, por ejemplo, cuando* $f(D)$ *sea la suma de los valores de cierto atributo en la base de datos, para este caso se tiene que* $\Delta f = a_{\text{máx}} - a_{\text{mín}}$, *donde* $a_{\text{máx}}$ *y* $a_{\text{mín}}$ *son el valor máximo y mínimo que puede tomar uno de los sumandos, respectivamente. Así, nuestro mecanismo sería tomar* $\widetilde{f}(D) = f(D) + Lap(0, \frac{a_{\text{máx}} - a_{\text{mín}}}{\epsilon})$.

Para el caso del promedio, al tratarse de dos operaciones combinadas (conteo y suma) podemos aplicar DP en dos momentos y publicar la operación entre estas dos, así podemos usar los Teoremas 1.3 y 1.4 de composición como se verá en el siguiente ejemplo.

Ejemplo 2.3. ***Promedio vía mecanismo exponencial:*** *Podemos combinar los Ejemplos 2.1 y 2.2 para el cálculo del promedio, pues este cálculo involucra el conocimiento de dos peticiones sobre la base de datos: la suma de los datos, y la cantidad*



(a) $\epsilon = 1$.



(b) $\epsilon = 0.1$.

Figura 2.2: Densidades del Mecanismo de Laplace.

de datos para después operarlos en un cociente. Para satisfacer la ϵ-DP podemos usar el Algoritmo 2.

Las peticiones, contar y sumar tienen sensibilidades distintas, de ahí que su perturbación sea diferente. Ahora, según el Teorema 1.4 de composición , el algoritmo anterior provee $(\frac{\epsilon}{2} + \frac{\epsilon}{2})$-DP, es decir, ϵ-DP.

Algoritmo 2 Algoritmo para el cálculo del promedio perturbado[23]

Entrada Base de datos D, parámetro de privacidad ϵ, rango de datos $[a_{\text{mín}}, a_{\text{máx}}]$
Salida Promedio $Avg(D)$
$\widetilde{S} \leftarrow Sum(D) + Lap(\frac{2(a_{\text{máx}} - a_{\text{mín}})}{\epsilon})$
$\widetilde{C} \leftarrow Cont(D) + Lap(\frac{2}{\epsilon})$
return $\frac{\widetilde{S}}{\widetilde{C}}$

En la Figura 2.3 podemos ver implementado el Ejemplo 2.3 en el cálculo de la edad promedio de muerte de nuestra base de datos. Se presentan el promedio real y algunos resultados obtenidos por el algoritmo. Los valores obtenidos con ruido se encuentran muy próximos al valor real. Estos resultados pueden ser considerados aceptables, ya que reflejan con precisión el comportamiento esperado de ser diferentes del real para preservar privacidad y están dentro de un rango razonable de variación con respecto al promedio real.

```
Promedio real: 4065.57
Ejemplos de Promedio con ruido:  4064.37 ,  4066.05 y  4065.56
```

Figura 2.3: Cálculo del Promedio con $\epsilon = 1$.

El mecanismo de Laplace puede llevarse al caso vectorial simplemente sumando una muestra aleatoria i.i.d., pero antes de ello es necesario introducir una definición previa.

Definición 2.2. *Para el caso de petición vectorial, la sensibilidad de f es la máxima diferencia bajo la norma L_1 entre bases de datos adyacentes y se denota como Δf, es decir,*

$$\begin{aligned} \Delta f &= \max_{D \simeq D'} ||f(D) - f(D')||_1 \\ &= \sum_i |f(D)_i - f(D')_i|. \end{aligned}$$

Teorema 2.2. *Para una petición f que devuelve un vector de tamaño k, el mecanismo de Laplace definido como $\mathcal{M}(D) = f(D) + (X_1, \ldots, X_k)$ satisface ϵ-DP donde $X_1, \ldots, X_k$ son i.i.d. con distribución $Lap(0, \frac{\Delta f}{\epsilon})$ y en este caso $\Delta f = \max_{D \simeq D'} ||f(D) - f(D')||_1$.*

Demostración. Sean D y D' dos bases adyacentes, $f(D) = (a_1, \ldots, a_k)$ y sea $t =$

$(t_1, \ldots, t_k) \in \mathbb{R}^k$, entonces observe que:

$$\begin{aligned}
q_D(t) &= f_{X_1,\ldots,X_k}(t_1 - a_1, \ldots, t_k - a_k) \\
&= f_{X_1}(t_1 - a_1) \cdot \ldots \cdot f_{X_k}(t_k - a_k) \\
&= \prod_{i=1}^{k} f_{X_i}(t_i - a_i) \\
&= \prod_{i=1}^{k} \frac{1}{2\frac{\Delta f}{\epsilon}} exp\left(-\frac{|t_i - a_i|}{\frac{\Delta f}{\epsilon}}\right) \\
&= (\frac{\epsilon}{2\Delta f})^k exp\left(-\frac{\epsilon \sum_{i=1}^{k} |t_i - a_i|}{\Delta f}\right) \\
&= (\frac{\epsilon}{2\Delta f})^k exp\left(-\frac{\epsilon ||t - f(D)||_1}{\Delta f}\right).
\end{aligned}$$

De manera análoga vamos a tener que $q_{D'}(t) = (\frac{\epsilon}{2\Delta f})^k exp(-\frac{\epsilon||t-f(D')||_1}{\Delta f})$. Luego,

$$\begin{aligned}
\frac{q_D(t)}{q_{D'}(t)} &= \frac{(\frac{\epsilon}{2\Delta f})^k exp(-\frac{\epsilon||t-f(D)||_1}{\Delta f})}{(\frac{\epsilon}{2\Delta f})^k exp(-\frac{\epsilon||t-f(D')||_1}{\Delta f})} \\
&= \frac{exp(-\frac{\epsilon||t-f(D)||_1}{\Delta f})}{exp(-\frac{\epsilon||t-f(D')||_1}{\Delta f})} \\
&= exp\left(-\frac{\epsilon||t - f(D)||_1}{\Delta f} - \left(-\frac{\epsilon||t - f(D')||_1}{\Delta f}\right)\right).
\end{aligned}$$

Ahora, usando la desigualdad del triángulo para la norma L_1 y la propia definición de Δf tenemos:

$$\begin{aligned}
\frac{q_D(t)}{q_{D'}(t)} &= exp\left(\frac{\epsilon(||t - f(D')||_1 - ||t - f(D)||_1)}{\Delta f}\right) \\
&\leq exp\left(\frac{\epsilon(||t - f(D') - (t - f(D))||_1)}{\Delta f}\right) \\
&= exp\left(\frac{\epsilon(||f(D) - f(D'))||_1)}{\Delta f}\right) \\
&\leq exp\left(\frac{\epsilon\Delta f}{\Delta f}\right) \\
&= e^{\epsilon}.
\end{aligned}$$

Lo que demuestra el resultado. □

Un ejemplo de aplicación de este tipo de peticiones es en el cálculo de histogramas, pues podemos pensar un histograma como un vector de peticiones, en el que

cada una de las entradas es un conteo de la cantidad de registros que se encuentran en cierto intervalo.

Ejemplo 2.4. *Para generar un histograma de las edades de personas que murieron por causa de violencia podemos aplicar este mecanismo, así obtenemos los histogramas de la Figura 2.4, en la cual podemos ver que, pese a que se trata de histogramas distintos, ambos mantienen la información que podemos considerar esencial, como dejar claro cuál es el bloque más frecuente (población entre 20 y 30 años). Vale la pena notar que el histograma que satisface ϵ-DP hace que aparezcan datos donde no los había, como en los bloques de edad mayor a 100 años, esto no debería significar un problema en un primer momento ya que para un adversario esto complicaría su trabajo, sin embargo, según la intención de publicar el histograma podría provocar que se preste atención a un bloque erróneo. De aquí la importancia de elegir un valor adecuado para ϵ.*

2.2. Mecanismo Exponencial

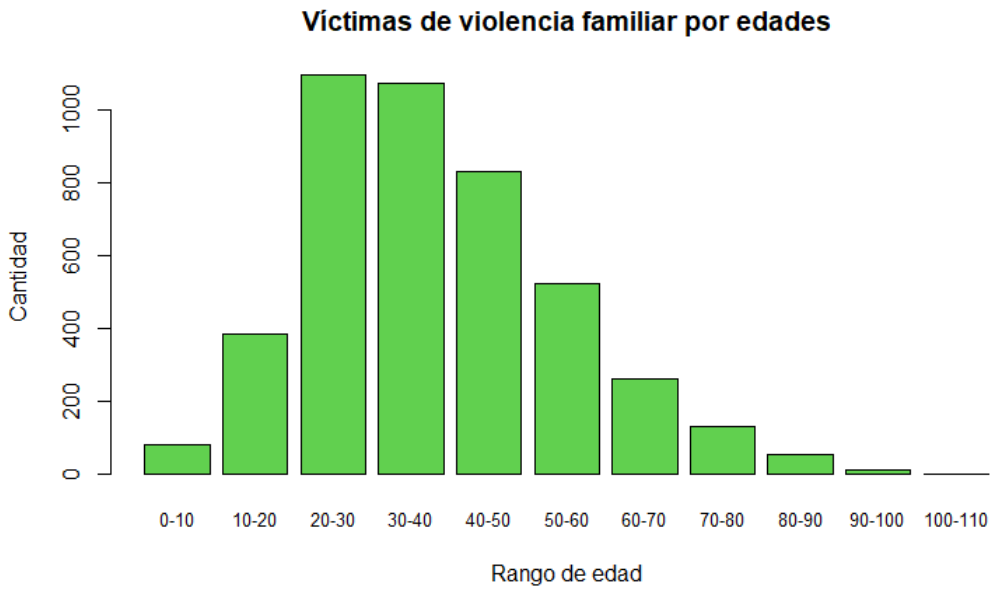
El mecanismo de Laplace resulta útil cuando se trata de peticiones que sean del tipo numérica, ya sea el caso escalar o vectorial, sin embargo, no puede adaptarse al caso en el que las peticiones sean del tipo categórico. En [28] se propone el mecanismo exponencial, este mecanismo puede aplicarse tanto a peticiones numéricas como a categóricas. La idea detrás de este mecanismo es la siguiente: Supongamos que queremos publicar $f(D) \in O$, siendo O el conjunto donde toma f sus valores. Para satisfacer la ϵ-DP el mecanismo debería devolver valores en O con cierta distribución de probabilidad, además nos gustaría que algunos valores en O fuesen más deseables que otros, tales como el valor real $f(D)$ o ciertos valores "cercanos". Esta preferencia se logra a través del uso de una función de utilidad $q : \mathcal{D} \times O \to \mathbb{R}$ tal que para cada $o \in O$ se asocia a cada pareja (D, o) un número real, en el entendido de que un valor alto indica una mejor utilidad del valor o para la base de datos D.

Definición 2.3. *Consideremos una función de utilidad $q : \mathcal{D} \times O \to \mathbb{R}$, y un parámetro de privacidad ϵ, el mecanismo exponencial, consiste en que dada una entrada $D \in \mathcal{D}$ se reporta $M_q^\epsilon(D) = o \in O$ con probabilidad proporcional a $\exp\left(\frac{\epsilon q(D,o)}{2\Delta q}\right)$ en donde $\Delta q = \max_{\forall o, D \cong D'} |q(D, o) - q(D', o)|$ es la sensibilidad de la función de utilidad. Esto significa que para el caso discreto se tiene la función de masa*

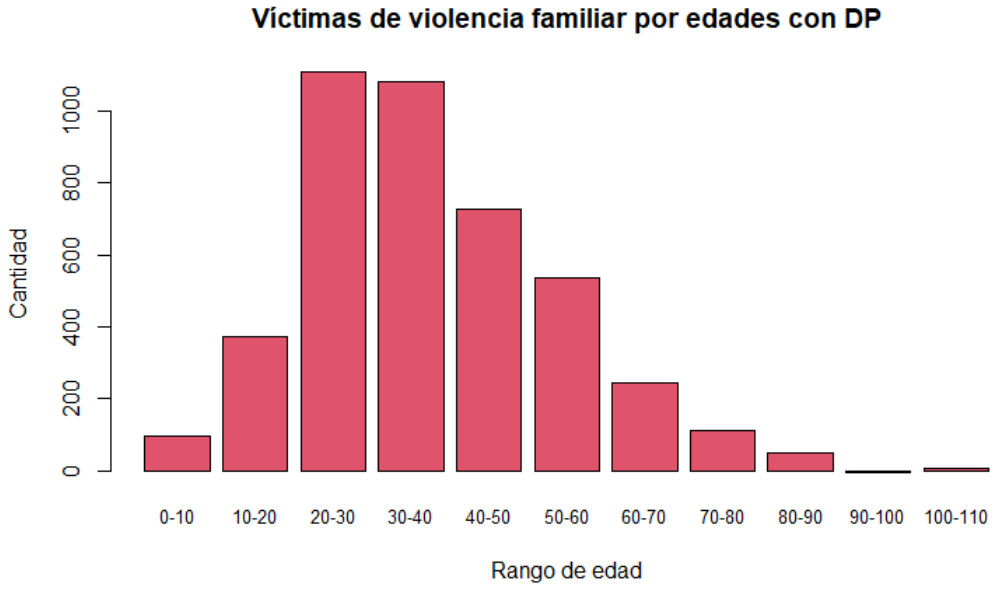
$$q_D(o) = \frac{\exp\left(\frac{\epsilon q(D,o)}{2\Delta q}\right)}{\sum_{o' \in O} \exp\left(\frac{\epsilon q(D,o')}{2\Delta q}\right)}, \tag{2.2}$$

y para el caso continuo se tiene la función de densidad

$$q_D(o) = \frac{\exp(\frac{\epsilon q(D,o)}{2\Delta q})}{\int_O \exp\left(\frac{\epsilon q(D,o')}{2\Delta q}\right) do'}. \tag{2.3}$$



(a) Histograma real.



(b) Histograma modificado con DP considerando $\epsilon = 1$.

Figura 2.4: Comparación de histogramas.

Teorema 2.3. *El mecanismo exponencial satisface ϵ-DP.*

Demostración. Sean D y D' bases de datos adyacentes y sea $o \in O$, observe que:

$$\begin{aligned}\frac{\exp(\frac{\epsilon q(D,o)}{2\Delta q})}{\exp(\frac{\epsilon q(D',o)}{2\Delta q})} &= \exp\left(\frac{\epsilon(q(D,o)-q(D',o))}{2\Delta q}\right)\\ &\leq \exp(\frac{\epsilon|q(D,o)-q(D',o)|}{2\Delta q})\\ &\leq \exp(\frac{\epsilon\Delta q}{2\Delta q})\\ &= \exp(\epsilon/2),\end{aligned}$$

es decir,

$$\frac{\exp(\frac{\epsilon q(D,o)}{2\Delta q})}{\exp(\frac{\epsilon q(D',o)}{2\Delta q})} \leq \exp(\epsilon/2). \tag{2.4}$$

De esto también se tiene que

$$\forall o' \in O: \quad \exp\left(\frac{\epsilon q(D',o')}{2\Delta q}\right) \leq \exp(\frac{\epsilon}{2})\exp\left(\frac{\epsilon q(D,o')}{2\Delta q}\right). \tag{2.5}$$

Así, para el caso discreto tenemos:

$$\begin{aligned}\frac{q_D(o)}{q_{D'}(o)} &= \frac{\frac{\exp\left(\frac{\epsilon q(D,o)}{2\Delta q}\right)}{\sum_{o'\in O}\exp\left(\frac{\epsilon q(D,o')}{2\Delta q}\right)}}{\frac{\exp\left(\frac{\epsilon q(D',o)}{2\Delta q}\right)}{\sum_{o'\in O}\exp\left(\frac{\epsilon q(D',o')}{2\Delta q}\right)}}\\ &= \left(\frac{\exp\left(\frac{\epsilon q(D,o)}{2\Delta q}\right)}{\exp\left(\frac{\epsilon q(D',o)}{2\Delta q}\right)}\right)\left(\frac{\sum_{o'\in O}\exp\left(\frac{\epsilon q(D',o')}{2\Delta q}\right)}{\sum_{o'\in O}\exp\left(\frac{\epsilon q(D,o')}{2\Delta q}\right)}\right).\end{aligned}$$

Ahora, usando las desigualdades (2.4) y (2.5) se tiene:

$$\begin{aligned}\frac{q_D(o)}{q_{D'}(o)} &\leq \exp(\frac{\epsilon}{2})\cdot\left(\frac{\sum_{o'\in O}\exp\left(\frac{\epsilon q(D',o')}{2\Delta q}\right)}{\sum_{o'\in O}\exp\left(\frac{\epsilon q(D,o')}{2\Delta q}\right)}\right)\\ &\leq \exp(\frac{\epsilon}{2})\cdot\left(\frac{\sum_{o'\in O}\exp(\frac{\epsilon}{2})\exp\left(\frac{\epsilon q(D,o')}{2\Delta q}\right)}{\sum_{o'\in O}\exp\left(\frac{\epsilon q(D,o')}{2\Delta q}\right)}\right)\\ &= \exp(\frac{\epsilon}{2})\cdot\exp(\frac{\epsilon}{2})\left(\frac{\sum_{o'\in O}\exp\left(\frac{\epsilon q(D,o')}{2\Delta q}\right)}{\sum_{o'\in O}\exp\left(\frac{\epsilon q(D,o')}{2\Delta q}\right)}\right)\\ &= \epsilon.\end{aligned}$$

Similarmente para el caso continuo tenemos:

$$\begin{aligned}\frac{q_D(o)}{q_{D'}(o)} &= \frac{\frac{\exp\left(\frac{\epsilon q(D,o)}{2\Delta q}\right)}{\int_O \exp\left(\frac{\epsilon q(D,o')}{2\Delta q}\right) do'}}{\frac{\exp\left(\frac{\epsilon q(D',o)}{2\Delta q}\right)}{\int_O \exp\left(\frac{\epsilon q(D',o')}{2\Delta q}\right) do'}} \\ &= \left(\frac{\exp\left(\frac{\epsilon q(D,o)}{2\Delta q}\right)}{\exp\left(\frac{\epsilon q(D',o)}{2\Delta q}\right)}\right) \cdot \left(\frac{\int_O \exp\left(\frac{\epsilon q(D',o')}{2\Delta q}\right) do'}{\int_O \exp\left(\frac{\epsilon q(D,o')}{2\Delta q}\right) do'}\right)\end{aligned}$$

usando las desigualdades (2.4) y (2.5) se concluye que:

$$\begin{aligned}\frac{q_D(o)}{q_{D'}(o)} &\leq \exp(\frac{\epsilon}{2}) \cdot \left(\frac{\int_O \exp\left(\frac{\epsilon q(D',o')}{2\Delta q}\right) do'}{\int_O \exp\left(\frac{\epsilon q(D,o')}{2\Delta q}\right) do'}\right) \\ &\leq \exp(\frac{\epsilon}{2}) \cdot \left(\frac{\int_O \exp(\frac{\epsilon}{2}) \exp\left(\frac{\epsilon q(D,o')}{2\Delta q}\right) do'}{\int_O \exp\left(\frac{\epsilon q(D,o')}{2\Delta q}\right) do'}\right) \\ &= \exp(\frac{\epsilon}{2}) \cdot \exp(\frac{\epsilon}{2}) \left(\frac{\int_O \exp\left(\frac{\epsilon q(D,o')}{2\Delta q}\right) do'}{\int_O \exp\left(\frac{\epsilon q(D,o')}{2\Delta q}\right) do'}\right) \\ &= e^{\epsilon}.\end{aligned}$$

Por lo tanto, en el caso discreto y en el continuo, se cumple la Definición 1.5 de ϵ-DP. □

En el siguiente ejemplo podemos ver, porqué el mecanismo de Laplace no es útil siempre, incluso cuando la petición sea numérica.

Ejemplo 2.5. *Supongamos que estamos interesados en hacer pública la estadística $f(D)$ siendo ésta la mediana de un conjunto de datos numéricos. Notemos que, según el rango de nuestro conjunto de datos, la sensibilidad de esta petición puede ser muy grande. Por ejemplo, si consideramos que nuestros datos toman valores entre 0 y 1000000 tenemos que:*

$$\Delta f = \max_{D \cong D'} |f(D) - f(D')| \geq 500000$$

pues si consideramos una base de datos con la misma cantidad de valores iguales a 1000000 *que de valores iguales a* 0*, tendremos que su mediana será* 500000 *y basta que modifiquemos un dato a* 0 *para que la mediana sea* 0*, de ahí que* $\Delta f = \max_{D \cong D'} |f(D) - f(D')| \geq |f(D) - f(D')| = |500000 - 0| = 500000$*. De lo cual podemos notar que usar el mecanismo de Laplace resultaría impráctico ya que*

para usarlo necesitamos una variable aleatoria $X \sim Lap(0, \frac{\Delta f}{\epsilon}) = Lap(0, \frac{500000}{\epsilon})$. *Un problema similar se obtiene si intentamos acomodar este mecanismo en el cálculo de la moda, pues el mecanismo no necesariamente nos devolverá un valor dentro del rango, ya que puede devolvernos un número que no sea entero, aunque los datos sean todos enteros, e incluso considerando un post-procesamiento de la salida (como tomar parte entera), esto no nos garantiza obtener un valor dentro del rango (imaginemos que todos los datos son pares o son categorías).*

Ejemplo 2.6. ***Moda vía mecanismo exponencial [23]:*** *Consideremos* O *como el rango de los posibles registros numéricos sobre los cuales queremos publicar la moda (en nuestro ejemplo anterior son los enteros entre 0 y 1000000) y tomemos como función de utilidad a* $n : \mathcal{D} \times O \to \mathbb{R}$ *tal que* $n(D, o) = n_o(D) = |\{d \in D | d = o\}|$, *dicho de otro modo,* $n_o(D)$ *cuenta los registros en* D *que sean iguales al valor* o. *Observe que para esta función de utilidad se tiene que* $\Delta n = 1$ *ya que modificar un registro modifica la función de contar* n *en una unidad. Así, el mecanismo exponencial que induce esta función de utilidad es devolver el valor* $\mathcal{M}(D) = o$ *con probabilidad*

$$\frac{\exp(\epsilon n_o(D))}{\sum_{y=0}^{1000000} \exp(\epsilon n_y(D))}.$$

Para ilustrar el Ejemplo 2.6, en la Figura 2.5 se muestra el resultado de implementar este algoritmo en el cálculo de la moda de las causas de muerte de nuestra base de datos. En esta figura se presenta la moda real, "U071" (COVID-19), y tres resultados obtenidos mediante el mecanismo. En la base de datos, "U071" (COVID-19) es la causa más común, seguida por "I219" (Infarto). Al comparar estos resultados, se observa que, aunque en algunos casos no se reporta la moda real, esta sigue siendo una causa con alta incidencia, lo cual resulta aceptable.

```
Moda real: U071
Ejemplos de Moda con ruido:  U071 ,  U071 y  I219
```

Figura 2.5: Cálculo de la Moda considerando $\epsilon = 1$.

Ejemplo 2.7. ***Mediana vía mecanismo exponencial [7]:*** *Consideremos* O *como el rango de los posibles registros numéricos sobre los cuales queremos publicar la mediana (en nuestro ejemplo anterior son los enteros entre 0 y 1000000). Definamos*

$$n_< : \mathcal{D} \times O \to \mathbb{R}$$

tal que $n_<(D, o) = n_{<o}(D) = |\{d \in D | d < o\}|$, *que dicho de otro modo,* $n_{<o}(D)$ *cuenta los registros en* D *que sean menores a* o *y tomemos como función de utilidad a* $q : \mathcal{D} \times O \to \mathbb{R}$ *tal que*

$$q(D, o) = -|n_{<o}(D) - \frac{|D|}{2}|.$$

La conveniencia que nos da esta función de utilidad, es que, se maximiza cuando se evalúa en la mediana real, que es lo que idealmente queremos.

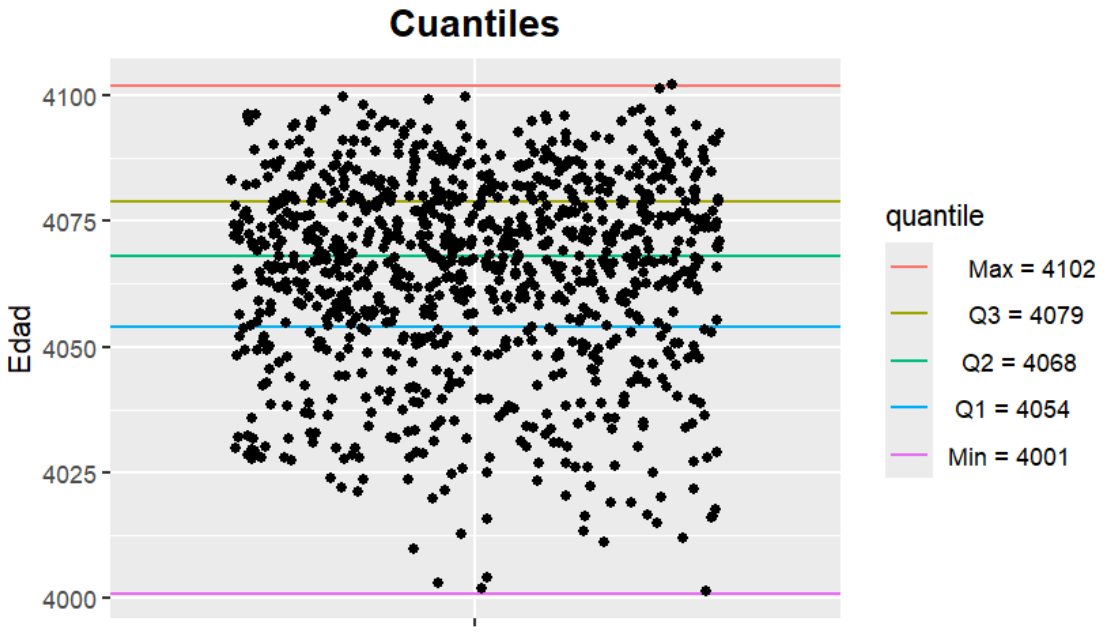
Ejemplo 2.8. ***Cuantiles vía mecanismo exponencial:*** *El caso de la mediana se puede generalizar al cálculo de los cuantiles de la siguiente forma, sólo debemos adaptar la función de utilidad convenientemente. Para el cuantil $\alpha \in (0,1)$ definimos la función de utilidad como:*

$$q(D,o) = -|n_{<o}(D) - |D| \cdot \alpha|.$$

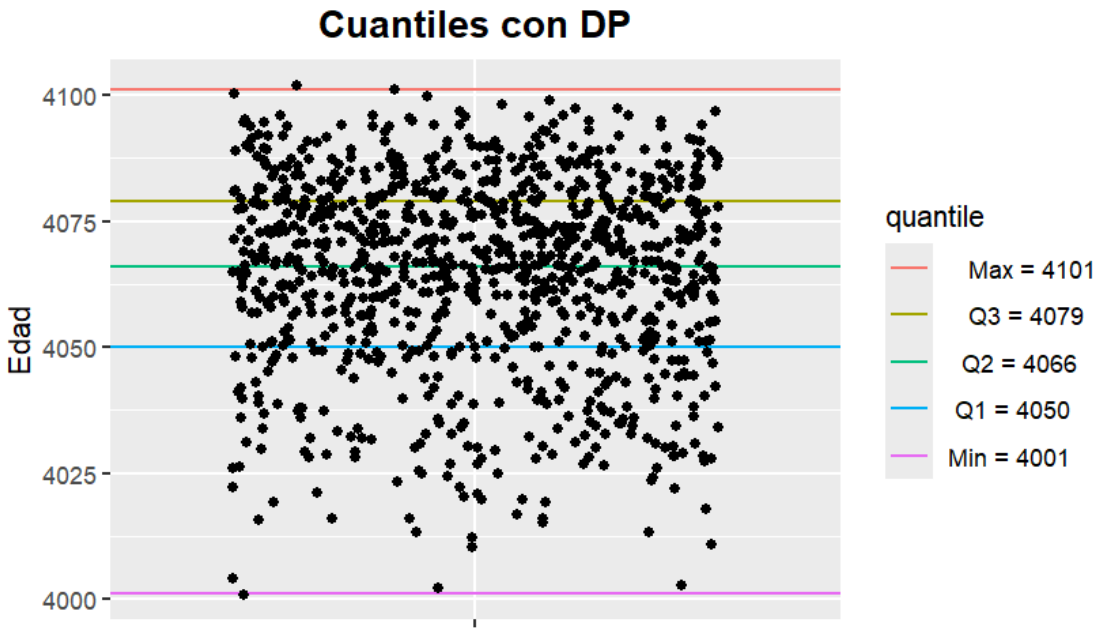
Del mismo modo, en la Figura 2.6 se muestra el resultado de implementar el Ejemplo 2.8 para el cálculo de diferentes cuantiles sobre las edades de muerte. Como puede verse, la versión con DP modifica el valor de algunos cuantiles mientras que otros los deja fijos, lo cual refleja el efecto del ruido introducido para garantizar la privacidad. Esta perturbación debe cuidarse especialmente en los cuantiles extremos, donde el ruido añadido puede generar desviaciones significativas. Por ello, es importante tener precaución al interpretar estos valores, ya que en aplicaciones sensibles podrían llevar a conclusiones erróneas o menos precisas.

Observación 2.1. ***El mecanismo de Laplace es un caso particular del mecanismo exponencial.*** *Consideremos una petición $f(D)$ con rango $\mathbb{R}$, aplicando el mecanismo de Laplace tenemos, $\mathcal{M}(D) = f(D) + Lap(\frac{\Delta f}{\epsilon})$. Si tomamos como O los números reales y función de utilidad a $q : \mathcal{D} \times O \to \mathbb{R}$ tal que $q(D,o) = -2|f(D) - o|$ se puede ver lo siguiente*

$$\begin{aligned}
\mathbb{P}(\mathcal{M}(D) = o) &= \frac{\epsilon}{2\Delta f} exp\left(-\frac{\epsilon|o - f(D)|}{\Delta f}\right) \\
&= \frac{\epsilon}{2\Delta f} exp\left(-\frac{\epsilon|f(D) - o|}{\Delta f}\right) \\
&= \frac{\frac{\epsilon}{2\Delta f} exp(-\frac{\epsilon|f(D)-o|}{\Delta f})}{1} \\
&= \frac{\frac{\epsilon}{2\Delta f} exp(-\frac{\epsilon|f(D)-o|}{\Delta f})}{\int_{\mathbb{R}} \frac{\epsilon}{2\Delta f} exp(-\frac{\epsilon|f(D)-t|}{\Delta f}) dt} \\
&= \frac{\frac{\epsilon}{2\Delta f} exp(-\frac{\epsilon|f(D)-o|}{\Delta f})}{\frac{\epsilon}{2\Delta f} \int_{\mathbb{R}} exp(-\frac{\epsilon|f(D)-t|}{\Delta f}) dt} \\
&= \frac{exp(-\frac{\epsilon|f(D)-o|}{\Delta f})}{\int_{\mathbb{R}} exp(-\frac{\epsilon|f(D)-t|}{\Delta f}) dt} \\
&= \frac{exp(\frac{\epsilon q(D,o)}{2\Delta f})}{\int_{\mathbb{R}} exp(\frac{\epsilon q(D,t)}{2\Delta f}) dt}.
\end{aligned}$$



(a) Cuantiles Reales.



(b) Cuantiles Modificados con DP considerando $\epsilon = 1$.

Figura 2.6: Diagrama de Cuantiles.

Como la última expresión coincide con (2.3) del Mecanismo Exponencial, se tiene el resultado. Más aún, según McSherry [28], cualquier mecanismo de aleatorización puede llevarse a un Mecanismo Exponencial usando la función de utilidad adecuada.

En este capítulo se han revisado las principales técnicas de Privacidad Diferencial y algunos ejemplos, en el siguiente capítulo se presentará cómo estas técnicas se aplican en algunos de los modelos más conocidos de Aprendizaje de Máquina.

Capítulo 3

Privacidad Diferencial en el Aprendizaje de Máquina

El aprendizaje de máquina se centra en el desarrollo de algoritmos y técnicas que permiten a las computadoras aprender y hacer predicciones o tomar decisiones basadas en datos. A través del aprendizaje de máquina, los sistemas pueden identificar patrones y hacer inferencias a partir de muestras de datos. La privacidad diferencial puede aplicarse a los algoritmos de aprendizaje de máquina con el objetivo de reducir el riesgo de reidentificación de un usuario que haya proporcionado sus datos para el entrenamiento del modelo. En esta sección estudiaremos cuatro modelos de aprendizaje de máquina: Regresión Lineal, Regresión Logística, Clasificación Difusa (*Fuzzy c-Means*) y Clasificación *Naive Bayes*, en cada uno de estos se proporciona una alternativa en la que se combine el uso de privacidad diferencial. Para el desarrollo de este capítulo se usó una base de datos referente a Diabetes tomada de Conjuntos de datos abiertos de Azure (Microsoft) y puede consultarse en `https://learn.microsoft.com/es-mx/azure/open-datasets/dataset-catalog` [29], además otra base usada es sobre información referente a vehículos del año 2000 tomada del sitio Kaggle y puede ser consultada en `https://www.kaggle.com/datasets/krupadharamshi/fuelconsumption/data` [21].

Para los modelos de Regresión Lineal, Regresión Logística y Clasificación Difusa, se asumirá que cada valor en la base de datos pertenece al intervalo $[0, 1]$ con el objetivo de simplificar los cálculos [45].

3.1. Regresión Lineal

La regresión lineal es un modelo matemático que tiene por objetivo predecir el valor de una variable desconocida a través de una relación lineal de otras variables conocidas. Este tipo de modelo se puede expresar en (3.1), en donde Y representa la variable a predecir (la cual es desconocida), $X_1, \ldots, X_d$ representan las variables conocidas, $\omega_0, \omega_1, \ldots, \omega_d$ son los parámetros del modelo (también llamados pesos) y

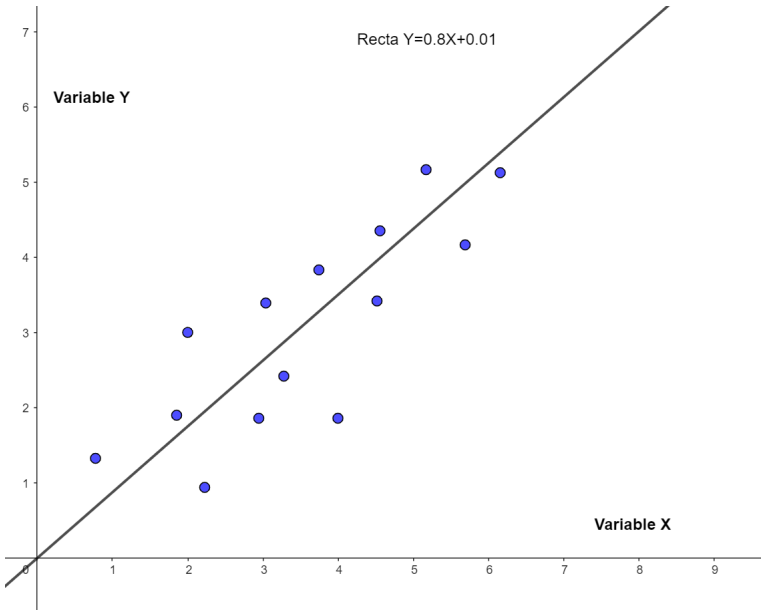


Figura 3.1: Regresión Lineal.

ϵ representa un factor de error.

$$Y = \omega_0 + \omega_1 X_1 + \cdots + \omega_d X_d + \epsilon. \tag{3.1}$$

El objetivo de la regresión lineal es construir un modelo de esta forma, para que así podamos hacer una predicción de Y conociendo los valores de $X_1, \ldots, X_d$. Para ello requerimos de un conjunto de datos de entrenamiento en los que explícitamente conozcamos tanto los valores de $X_1, \ldots, X_n$ como de Y y con ello ajustar los parámetros $\omega_0, \omega_1, \ldots, \omega_d$ a modo de minimizar la diferencia entre los datos de entrenamiento y la recta construida por estos parámetros, esta idea queda planteada en la Figura 3.1.

El proceso para ajustar un modelo de regresión lineal consiste en un problema de optimización. Para ello consideremos D una base de datos con n registros $d+1$-dimensional, tal que cada registro es de la forma $t_i = (x_{i1}, x_{i2}, \ldots, x_{id}, y_i)$ para $i = 1, \ldots, n$, por simplicidad denotamos $\mathbf{x_i} = (x_{i1}, x_{i2}, \ldots, x_{id}) \in \mathbb{R}^d$.

Definición 3.1. *Sea D una base de datos como antes, una regresión lineal en D devuelve una función de predicción $\rho(\mathbf{x_i}) = \mathbf{x_i} \cdot \boldsymbol{\omega}^*$, donde $\boldsymbol{\omega}^* \in \mathbb{R}^d$ es tal que minimiza alguna función de costo o error (función objetivo) $f_D(\boldsymbol{\omega}) = \sum_{i=1}^n f(t_i, \boldsymbol{\omega})$.*

En la definición anterior, normalmente se considera como función de costo a $f_D(\boldsymbol{\omega}) = \sum_{i=1}^n (y_i - \mathbf{x_i} \cdot \boldsymbol{\omega})^2$ (suma de los cuadrados de los residuos) ya que mide la cantidad de variabilidad que queda sin explicación después de realizar la regresión [19], así $\boldsymbol{\omega}^* \in \arg\min \sum_{i=1}^n (y_i - \mathbf{x_i} \cdot \boldsymbol{\omega})^2$.

Cuando se ajusta un modelo de regresión lineal, pese a que no se hacen públicos los valores en la base de datos D, sino sólo los pesos, cuando el modelo resulta ser

muy bueno que los datos se ajusten tan bien a la recta, (esto pasa cuando $\epsilon = 0$ en (3.1)) si se conocen los suficientes atributos de un registro t_i, se puede inferir el valor de un parámetro sensible.

Por esta razón, es importante realizar algún tipo de perturbación en el proceso de ajuste de estos modelos. El mecanismo funcional que propone Zhang [45] consiste en reescribir la función de costos a una forma polinómica y perturbar cada uno de los coeficientes a través de un ruido controlado, y optimizar esta nueva función a modo de obtener los parámetros que se necesitan.

Antes de continuar es útil introducir la siguiente notación. Recordemos que $\boldsymbol{\omega} = (\omega_0, \ldots, \omega_d)$. Para cada $j = 1, 2, \ldots$ denotemos por

$$\Phi_j = \{w_0^{c_0} w_1^{c_1} \ldots w_d^{c_d} | \sum_{l=1}^{d} c_l = j\},$$

al conjunto de todos los posibles productos de las potencias de $\omega_0, \ldots, \omega_d$ tales que la suma de los exponentes sume j. Por ejemplo, si $d = 2$, entonces $\Phi_0 = \{1\}$, $\Phi_1 = \{\omega_0, \omega_1, \omega_2\}$, $\Phi_2 = \{\omega_0^2, \omega_1^2, \omega_2^2, \omega_0\omega_1, \omega_0\omega_2, \omega_1\omega_2\}$, etc. Además de esto, también por conveniencia denotaremos por $\phi(\boldsymbol{\omega})$ a algún producto de $\omega_0, \ldots, \omega_d$.

Observe que la función $f(t_i, \boldsymbol{\omega})$ de la Definición 3.1 según el Teorema de Stone-Weierstrass [20] puede ser escrita como un polinomio de $\omega_0, \omega_1, \ldots, \omega_d$, esto es

$$f(t_i, \boldsymbol{\omega}) = \sum_{j=0}^{J} \sum_{\phi \in \Phi_j} \lambda_{\phi t_i} \phi(\boldsymbol{\omega}) \tag{3.2}$$

donde $J \in \mathbb{N}$ es el grado del polinomio y cada $\lambda_{\phi t_i}$ es el coeficiente del monomio $\phi(\boldsymbol{\omega})$. Para ilustrar mejor esto, veamos el siguiente ejemplo:

Ejemplo 3.1. *Considerando como función de costos $f(t_i, \boldsymbol{\omega}) = (y_i - \mathbf{x_i} \cdot \boldsymbol{\omega})^2$, podemos expresarla en su forma polinómica como sigue:*

$$\begin{aligned} f(t_i, \boldsymbol{\omega}) &= (y_i - \mathbf{x_i} \cdot \boldsymbol{\omega})^2 \\ &= y_i^2 - 2y_i x_i \boldsymbol{\omega} + (x_i \cdot \boldsymbol{\omega})^2 \\ &= y_i^2 - 2y_i \sum_{j=1}^{d} x_{ij}\omega_j + \left(\sum_{j=1}^{d} x_{ij}\omega_j\right)^2 \\ &= y_i^2 - 2y_i \sum_{j=1}^{d} x_{ij}\omega_j + \sum_{j=1}^{d}\sum_{l=1}^{d} x_{ij}\omega_j x_{il}\omega_l \\ &= y_i^2 + \sum_{j=1}^{d}(-2x_{ij}y_i)\omega_j + \sum_{1 \le j,l \le d} (x_{ij}x_{il})\omega_j\omega_l. \end{aligned}$$

Podemos identificar los elementos del polinomio en (3.2):

- *Para* $j = 0$

$$\sum_{\phi \in \Phi_0} \lambda_{\phi t_i} \phi(\boldsymbol{\omega}) = y_i^2$$

si $\phi \in \Phi_0 = \{1\}$, $\lambda_{\phi t_i} = y_i^2$ *y* $\phi(\boldsymbol{\omega}) = 1$.

- *Para* $j = 1$

$$\sum_{\phi \in \Phi_1} \lambda_{\phi t_i} \phi(\boldsymbol{\omega}) = \sum_{j=1}^{d} (-2x_{ij} y_i)\omega_j$$

si $\phi \in \Phi_1$, $\lambda_{\phi t_i} = -2x_{ij} y_i$ *y* $\phi(\boldsymbol{\omega}) = \omega_j$ *para* $j \in \{1, 2, \ldots, d\}$.

- *Para* $j = 2$

$$\sum_{\phi \in \Phi_2} \lambda_{\phi t_i} \phi(\boldsymbol{\omega}) = \sum_{1 \leq j,l \leq d} (x_{ij} x_{il})\omega_j \omega_l$$

si $\phi \in \Phi_2$ $\lambda_{\phi t_i} = x_{ij} x_{il}$ *y* $\phi(\boldsymbol{\omega}) = \omega_j \omega_l$ *para* $j, l \in \{1, 2, \ldots, d\}$.

Ahora, como $f_D(\boldsymbol{\omega}) = \sum_{i=1}^{n} f(t_i, \boldsymbol{\omega})$, f_D también puede llevarse a una forma polinómica:

$$\begin{aligned} f_D(\boldsymbol{\omega}) &= \sum_{i=1}^{n} f(t_i, \boldsymbol{\omega}) \\ &= \sum_{i=1}^{n} \sum_{j=0}^{J} \sum_{\phi \in \Phi_j} \lambda_{\phi t_i} \phi(\boldsymbol{\omega}) \\ &= \sum_{j=0}^{J} \sum_{\phi \in \Phi_j} \sum_{i=1}^{n} \lambda_{\phi t_i} \phi(\boldsymbol{\omega}) \\ &= \sum_{j=0}^{J} \sum_{\phi \in \Phi_j} \phi(\boldsymbol{\omega}) \sum_{i=1}^{n} \lambda_{\phi t_i} \\ &= \sum_{j=0}^{J} \sum_{\phi \in \Phi_j} \phi(\boldsymbol{\omega}) \lambda_{\phi} \end{aligned} \tag{3.3}$$

en donde $\lambda_{\phi} = \sum_{i=1}^{n} \lambda_{\phi t_i}$.

Continuando con el Ejemplo 3.1 se ilustra lo anterior en el Ejemplo 3.2.

Ejemplo 3.2.

$$\begin{aligned} f_D(\boldsymbol{\omega}) &= \sum_{i=1}^{n} f(t_i, \boldsymbol{\omega}) \\ &= \sum_{i=1}^{n} [y_i^2 + \sum_{j=1}^{d} (-2x_{ij}y_i)\omega_j + \sum_{1\leq j,l\leq d} (x_{ij}x_{il})\omega_j\omega_l] \\ &= \sum_{i=1}^{n} y_i^2 + \sum_{i=1}^{n}\sum_{j=1}^{d} (-2x_{ij}y_i)\omega_j + \sum_{i=1}^{n} \sum_{1\leq j,l\leq d} (x_{ij}x_{il})\omega_j\omega_l \\ &= \sum_{i=1}^{n} y_i^2 + \sum_{j=1}^{d} (-2\sum_{i=1}^{n} x_{ij}y_i)\omega_j + \sum_{1\leq j,l\leq d} (\sum_{i=1}^{n} x_{ij}x_{il})\omega_j\omega_l. \end{aligned}$$

De esto podemos identificar:

- *si* $\phi \in \Phi_0 = \{1\}$

$$\lambda_\phi = \sum_{i=1}^{n} \lambda_{\phi t_i} = \sum_{i=1}^{n} y_i^2$$

  para algún $j \in \{1, 2, \ldots, d\}$

- *si* $\phi \in \Phi_1$

$$\lambda_\phi = \sum_{i=1}^{n} \lambda_{\phi t_i} = -2\sum_{i=1}^{n} x_{ij}y_i$$

  para algún $j \in \{1, 2, \ldots, d\}$

- *si* $\phi \in \Phi_2$

$$\lambda_\phi = \sum_{i=1}^{n} \lambda_{\phi t_i} = \sum_{i=1}^{n} x_{ij}x_{il}$$

  para algunos $j, l \in \{1, 2, \ldots, d\}$.

Con la forma de (3.3), Zhang propone en [45] perturbar cada uno de los coeficientes λ_ϕ agregando un ruido $Lap(\frac{\Delta}{\epsilon})$, en donde $\Delta = 2\,\text{máx}_{t_i \in D} \sum_{j=1}^{J} \sum_{\phi\in\Phi} ||\lambda_{\phi t_i}||$. Esto queda planteado en el Algoritmo 3.

Ahora, veamos que este algoritmo cumple con la definición de DP.

Teorema 3.1. *El algoritmo 3 satisface* ϵ*-DP*

Algoritmo 3 Algoritmo Mecanismo Funcional [45]

Entrada Base de datos D, función objetivo $f_D(\boldsymbol{\omega}) = \sum_{j=1}^{J} \sum_{\phi \in \Phi} \lambda_\phi \phi(\boldsymbol{\omega})$, parámetro de privacidad ϵ
Salida Parámetros del modelo perturbado $\boldsymbol{\omega}^*$

1: $\Delta = 2\,\text{máx}_{t_i \in D} \sum_{j=1}^{J} \sum_{\phi \in \Phi} ||\lambda_{\phi t_i}||$
2: **for** j=1 hasta J **do**
3: **for** $\phi \in \Phi_j$ **do**
4: $\tilde{\lambda_\phi} = \lambda_\phi + Lap(\frac{\Delta}{\epsilon})$ #Se agrega un valor según la densidad de Laplace
5: **end for**
6: **end for**
7: $\tilde{f_D}(\boldsymbol{\omega}) = \sum_{j=1}^{J} \sum_{\phi \in \Phi} \tilde{\lambda_\phi} \phi(\boldsymbol{\omega})$
8: $\boldsymbol{\omega}^* = \arg\min \tilde{f_D}(\boldsymbol{\omega})$
9: **return** $\boldsymbol{\omega}^*$

Demostración. Sean D y D' bases de datos adyacentes, sin pérdida de generalidad vamos a suponer que difieren en el último registro, sea t_n y t'_n los últimos registros de D y D' respectivamente. Vamos a demostrar que la salida del paso 7, el polinomio $\tilde{f}_D(\boldsymbol{\omega})$ satisface ϵ-DP. Sea $p(\boldsymbol{\omega}) = \sum_{j=1}^{J} \sum_{\phi \in \Phi_j} r_\phi \phi(\boldsymbol{\omega})$ un polinomio de $\boldsymbol{\omega}$ de grado J se tiene lo siguiente:

$$\begin{aligned}\frac{\mathbb{P}(\tilde{f_D}(\boldsymbol{\omega}) = p(\boldsymbol{\omega}))}{\mathbb{P}(\tilde{f_{D'}}(\boldsymbol{\omega}) = p(\boldsymbol{\omega}))} &= \frac{\mathbb{P}(\sum_{j=1}^{J} \sum_{\phi \in \Phi_j} \tilde{\lambda_\phi} \phi(\boldsymbol{\omega}) = \sum_{j=1}^{J} \sum_{\phi \in \Phi_j} r_\phi \phi(\boldsymbol{\omega}))}{\mathbb{P}(\sum_{j=1}^{J} \sum_{\phi \in \Phi_j} \tilde{\lambda_\phi} \phi(\boldsymbol{\omega}) = \sum_{j=1}^{J} \sum_{\phi \in \Phi_j} r_\phi \phi(\boldsymbol{\omega}))} \\ &= \frac{\mathbb{P}(\sum_{j=1}^{J} \sum_{\phi \in \Phi_j} (\sum_{t_i \in D} \lambda_{\phi t_i}) + Lap(\frac{\Delta}{\epsilon})) = \sum_{j=1}^{J} \sum_{\phi \in \Phi_j} r_\phi \phi(\boldsymbol{\omega}))}{\mathbb{P}(\sum_{j=1}^{J} \sum_{\phi \in \Phi_j} (\sum_{t_i \in D'} \lambda_{\phi t_i}) + Lap(\frac{\Delta}{\epsilon})) = \sum_{j=1}^{J} \sum_{\phi \in \Phi_j} r_\phi \phi(\boldsymbol{\omega}))}.\end{aligned}$$

Ahora, sabemos que dos polinomios son iguales si y sólo si sus coeficientes son iguales término a término, así:

$$\begin{aligned}\frac{\mathbb{P}(\tilde{f_D}(\boldsymbol{\omega}) = p(\boldsymbol{\omega}))}{\mathbb{P}(\tilde{f_{D'}}(\boldsymbol{\omega}) = p(\boldsymbol{\omega}))} &= \frac{\mathbb{P}(\left\{\sum_{t_i \in D} \lambda_{\phi t_i} + Lap(\frac{\Delta}{\epsilon}) = r_\phi\right\}, \forall \phi \in \Phi_j, j = 1, \ldots, J)}{\mathbb{P}(\left\{\sum_{t_i \in D'} \lambda_{\phi t_i} + Lap(\frac{\Delta}{\epsilon}) = r_\phi\right\}, \forall \phi \in \Phi_j, j = 1, \ldots, J)} \\ &= \frac{\mathbb{P}(\bigcap_{j=1}^{J} \bigcap_{\phi \in \Phi_j} \left\{\sum_{t_i \in D} \lambda_{\phi t_i} + Lap(\frac{\Delta}{\epsilon}) = r_\phi\right\})}{\mathbb{P}(\bigcap_{j=1}^{J} \bigcap_{\phi \in \Phi_j} \left\{\sum_{t_i \in D'} \lambda_{\phi t_i} + Lap(\frac{\Delta}{\epsilon}) = r_\phi\right\})} \\ &= \frac{\mathbb{P}(\bigcap_{j=1}^{J} \bigcap_{\phi \in \Phi_j} \left\{Lap(\frac{\Delta}{\epsilon}) = r_\phi - \sum_{t_i \in D} \lambda_{\phi t_i}\right\})}{\mathbb{P}(\bigcap_{j=1}^{J} \bigcap_{\phi \in \Phi_j} \left\{Lap(\frac{\Delta}{\epsilon}) = r_\phi - \sum_{t_i \in D'} \lambda_{\phi t_i}\right\})}.\end{aligned}$$

Como todas las variables $Lap(\frac{\Delta}{\epsilon})$ son independientes entre sí, la probabilidad anterior se simplifica, además, en esta parte usamos la función de densidad de una

variable aleatoria $Lap(\frac{\Delta}{\epsilon})$.

$$\begin{aligned}\frac{\mathbb{P}(\tilde{f_D}(\boldsymbol{\omega}) = p(\boldsymbol{\omega}))}{\mathbb{P}(\tilde{f_{D'}}(\boldsymbol{\omega}) = p(\boldsymbol{\omega}))} &= \frac{\prod_{j=1}^{J}\prod_{\phi\in\Phi_j}\mathbb{P}(Lap(\frac{\Delta}{\epsilon}) = r_\phi - \sum_{t_i\in D}\lambda_{\phi t_i})}{\prod_{j=1}^{J}\prod_{\phi\in\Phi_j}\mathbb{P}(Lap(\frac{\Delta}{\epsilon}) = r_\phi - \sum_{t_i\in D'}\lambda_{\phi t_i})} \\ &= \frac{\prod_{j=1}^{J}\prod_{\phi\in\Phi_j}\frac{1}{2}(\frac{\Delta}{\epsilon})\exp(-\frac{|r_\phi - \sum_{t_i\in D}\lambda_{\phi t_i}|}{(\frac{\Delta}{\epsilon})})}{\prod_{j=1}^{J}\prod_{\phi\in\Phi_j}\frac{1}{2}(\frac{\Delta}{\epsilon})\exp(-\frac{|r_\phi - \sum_{t_i\in D'}\lambda_{\phi t_i}|}{(\frac{\Delta}{\epsilon})})}.\end{aligned}$$

Luego, aplicando la desigualdad del triángulo:

$$\begin{aligned}\frac{\mathbb{P}(\tilde{f_D}(\boldsymbol{\omega}) = p(\boldsymbol{\omega}))}{\mathbb{P}(\tilde{f_{D'}}(\boldsymbol{\omega}) = p(\boldsymbol{\omega}))} &= \prod_{j=1}^{J}\prod_{\phi\in\Phi_j}\frac{\exp(-\frac{\epsilon|r_\phi - \sum_{t_i\in D}\lambda_{\phi t_i}|}{\Delta})}{\exp(-\frac{\epsilon|r_\phi - \sum_{t_i\in D'}\lambda_{\phi t_i}|}{\Delta})} \\ &= \prod_{j=1}^{J}\prod_{\phi\in\Phi_j}\exp\left(-\frac{\epsilon|r_\phi - \sum_{t_i\in D}\lambda_{\phi t_i}|}{\Delta} + \frac{\epsilon|r_\phi - \sum_{t_i\in D'}\lambda_{\phi t_i}|}{\Delta}\right) \\ &= \prod_{j=1}^{J}\prod_{\phi\in\Phi_j}\exp\left(\frac{\epsilon[|r_\phi - \sum_{t_i\in D'}\lambda_{\phi t_i}| - |r_\phi - \sum_{t_i\in D}\lambda_{\phi t_i}|]}{\Delta}\right) \\ &\leq \prod_{j=1}^{J}\prod_{\phi\in\Phi_j}\exp\left(\frac{\epsilon|\sum_{t_i\in D}\lambda_{\phi t_i} - \sum_{t_i\in D'}\lambda_{\phi t_i}|}{\Delta}\right).\end{aligned}$$

Ahora, como hicimos el supuesto de que D y D' sólo son diferentes en el último registro, entonces la diferencia de las sumas dentro de la exponencial se reduce a:

$$\begin{aligned}\frac{\mathbb{P}(\tilde{f_D}(\boldsymbol{\omega}) = p(\boldsymbol{\omega}))}{\mathbb{P}(\tilde{f_{D'}}(\boldsymbol{\omega}) = p(\boldsymbol{\omega}))} &\leq \prod_{j=1}^{J}\prod_{\phi\in\Phi_j}\exp\left(\frac{\epsilon|\lambda_{\phi t_n} - \lambda_{\phi t'_n}|}{\Delta}\right) \\ &= \exp\left(\frac{\epsilon}{\Delta}\sum_{j=1}^{J}\sum_{\phi\in\Phi_j}|\lambda_{\phi t_n} - \lambda_{\phi t'_n}|\right) \\ &\leq \exp\left(\frac{\epsilon}{\Delta}\sum_{j=1}^{J}\sum_{\phi\in\Phi_j}(|\lambda_{\phi t_n}| + |\lambda_{\phi t'_n}|)\right) \\ &\leq \exp\left(\frac{\epsilon}{\Delta}2\max_t\sum_{j=1}^{J}\sum_{\phi\in\Phi_j}|\lambda_{\phi t}|\right) \\ &= \exp\left(\frac{\epsilon}{\Delta}\Delta\right) \\ &= e^\epsilon.\end{aligned}$$

□

Por lo tanto, el algoritmo efectivamente funciona, ahora, continuando con el Ejemplo 3.2 los nuevos coeficientes perturbados se desarrollan en el Ejemplo 3.3.

Ejemplo 3.3. *Primero determinemos Δ, esto se obtiene con ayuda de los resultados del Ejemplo 3.1 y recordando que cada valor en la base de datos está en el intervalo $[0,1]$, así tenemos:*

$$\begin{aligned}
\Delta &= 2 \max_t \sum_{j=1}^{J} \sum_{\phi \in \Phi_j} |\lambda_{\phi t}| \\
&= 2 \max_t \left(|y^2| + \sum_{j=1}^{d} | - 2x_j y| + \sum_{1 \leq j,l \leq d} |x_j x_l| \right) \\
&= 2 \max_t \left(y^2 + 2 \sum_{j=1}^{d} |x_j y| + \sum_{1 \leq j,l \leq d} |x_j x_l| \right) \\
&\leq 2 \max_t \left(y^2 + 2 \sum_{j=1}^{d} 1 + \sum_{1 \leq j,l \leq d} 1 \right) \\
&= 2(1 + 2d + d^2) \\
&= 2(d+1)^2.
\end{aligned}$$

Ahora, según el Algoritmo 3 se tiene $\tilde{\lambda_\phi} = \lambda_\phi + Lap(\frac{\Delta}{\epsilon})$, es decir:
Para $\phi \in \Phi_0 = \{1\}$

$$\tilde{\lambda_\phi} = \sum_{i=1}^{n} y_i^2 + Lap\left(\frac{\Delta}{\epsilon}\right).$$

Para $\phi \in \Phi_1$

$$\tilde{\lambda_\phi} = -2 \sum_{i=1}^{n} x_{ij} y_i + Lap\left(\frac{\Delta}{\epsilon}\right)$$

con algún $j \in \{1, 2, \ldots, d\}$.
Y para $\phi \in \Phi_2$

$$\tilde{\lambda_\phi} = \sum_{i=1}^{n} x_{ij} x_{il} + Lap\left(\frac{\Delta}{\epsilon}\right)$$

con algunos $j, l \in \{1, 2, \ldots, d\}$. Luego, nuestra nueva función objetivo con ruido es:

$$\begin{aligned}\tilde{f}_D(\boldsymbol{\omega}) &= \sum_{k=0}^{J} \sum_{\phi \in \Phi_k} \tilde{\lambda_\phi} \phi(\boldsymbol{\omega}) \\ &= \sum_{k=0}^{J} \sum_{\phi \in \Phi_k} \left(\lambda_{\phi t_i} + Lap\left(\frac{\Delta}{\epsilon}\right) \right) \phi(\boldsymbol{\omega}) \\ &= \left(\sum_{i=1}^{n} y_i^2 \right) + Lap\left(\frac{\Delta}{\epsilon}\right) + \sum_{j=1}^{d} \left(\left(-2 \sum_{i=1}^{n} x_{ij} y_i \right) + Lap\left(\frac{\Delta}{\epsilon}\right) \right) \omega_j \\ &\qquad + \sum_{1 \leq j,l \leq d} \left(\left(\sum_{i=1}^{n} x_{ij} x_{il} \right) + Lap\left(\frac{\Delta}{\epsilon}\right) \right) \omega_j \omega_l.\end{aligned}$$

Como sabemos que $\Delta = 2(d+1)^2$ *se tiene:*

$$\begin{aligned}\tilde{f}_D(\boldsymbol{\omega}) = \left(\sum_{i=1}^{n} y_i^2 \right) + Lap\left(\frac{2(d+1)^2}{\epsilon}\right) + \sum_{j=1}^{d} \left(\left(-2 \sum_{i=1}^{n} x_{ij} y_i \right) + Lap\left(\frac{2(d+1)^2}{\epsilon}\right) \right) \omega_j \\ + \sum_{1 \leq j,l \leq d} \left(\left(\sum_{i=1}^{n} x_{ij} x_{il} \right) + Lap\left(\frac{2(d+1)^2}{\epsilon}\right) \right) \omega_j \omega_l.\end{aligned}$$

Sobre esta última calculamos el valor donde alcanza su mínimo para así obtener $\boldsymbol{\omega}^* \in \arg\min_{\boldsymbol{\omega}} \tilde{f}_D(\boldsymbol{\omega})$.

Con esto, podemos aplicar el modelo de Regresión Lineal con Privacidad Diferencial (DP). En la Figura 3.2 se muestran las gráficas correspondientes a la relación entre el consumo de combustible de un motor y sus emisiones de CO_2, comparando un modelo tradicional de regresión lineal con su versión modificada para incorporar DP. Como puede observarse, existe una diferencia notable entre las rectas de regresión de ambos modelos. Sin embargo, es importante destacar que, en la región donde se concentra la mayor parte de los datos, ambas rectas se ajustan de manera similar. A medida que los datos se dispersan, la recta con DP muestra un ajuste menos preciso. No obstante, aunque el error es mayor en la versión con DP, su crecimiento no es significativo, lo que sugiere que este modelo modificado sigue siendo viable para su consideración.

3.2. Regresión Logística

Otro modelo de regresión muy usado es el de regresión logística, este modelo pertenece a una clase más grande llamada modelos de clasificación, en este tipo de modelos la variable a predecir Y no es cuantitativa como lo era en la regresión

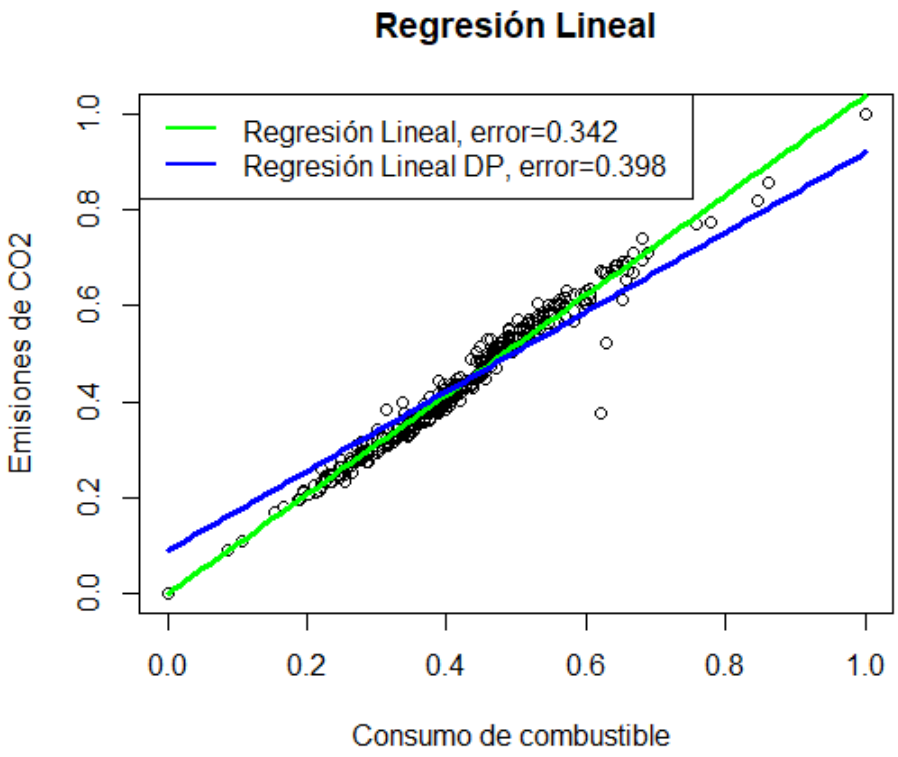


Figura 3.2: Regresión lineal en datos de consumo de combustible: comparación entre la versión normal y con privacidad diferencial para $\epsilon = 1$.

lineal, sino que en este caso se trata de una variable categórica, es decir, que estamos interesados en clasificar los registros en grupos según sus datos.

Para el caso de la regresión logística nos restringimos a que la variable Y tome los valores 0 y 1, es decir, que determinamos dos categorías y similarmente a como lo hicimos en la regresión lineal, queremos predecir el valor de Y (que para este caso es una categoría) con base en otras variables conocidas $X_1, X_2, \ldots, X_n$. A diferencia de la regresión lineal, no podemos tomar una combinación lineal de las variables conocidas para predecir, ya que, esto no necesariamente nos devolverá valores 0 o 1. Para resolver esto tomaremos a Y como una variable aleatoria y trataremos de aproximar $P(Y = 1)$ con los valores de $X_1, \ldots, X_n$, esto es, $P(Y = 1|X_1, \ldots, X_n)$ para que este valor se encuentre entre 0 y 1 se ajusta una curva sigmoide, esta queda representada por (3.4) y su gráfica en la Figura 3.3.

$$p(\mathbf{X}) = \frac{\exp(\omega_0 + \omega_1 X_1 + \cdots + \omega_d X_d)}{1 + \exp(\omega_0 + \omega_1 X_1 + \cdots + \omega_d X_d)} \tag{3.4}$$

Ahora, para estimar los parámetros $\boldsymbol{\omega}$ introducimos el concepto de odds que

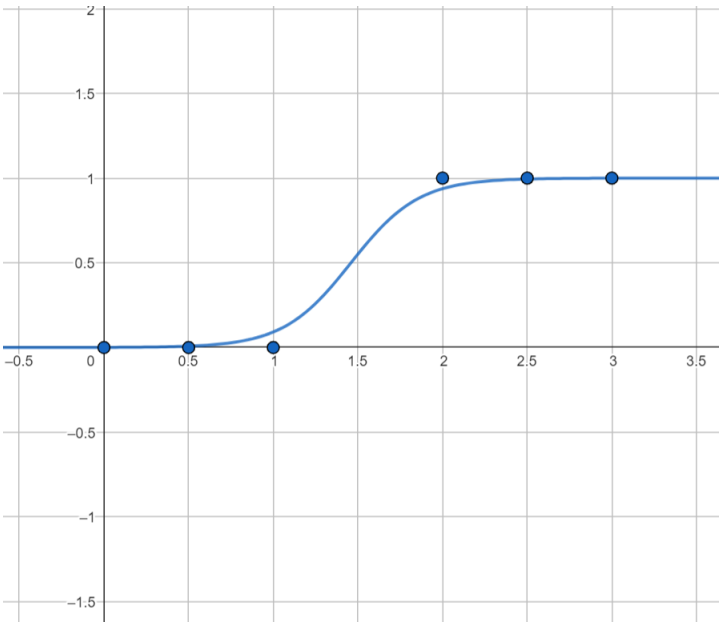


Figura 3.3: Regresión Logística.

representa la proporción entre casos favorables y no favorables de un evento.

$$odds = \frac{p}{1-p},$$

donde p representa la probabilidad de que ocurra el evento de interés. Con esto se puede llegar a que

$$p = \frac{e^{\log(odds)}}{1 + e^{\log(odds.)}}.$$

La forma anterior se deriva del siguiente modo:

$$\log(\frac{p}{1-p}) = \log(odds).$$

Tomando exponencial de ambos lados

$$\frac{p}{1-p} = e^{\log(odds)}.$$

Así, podemos despejar a p como sigue

$$\begin{aligned} p &= (1-p)e^{\log(odds)} \\ p &= e^{\log(odds)} - pe^{\log(odds)} \\ p + pe^{\log(odds)} &= e^{\log(odds)} \\ p(1 + e^{\log(odds)}) &= e^{\log(odds)} \\ p &= \frac{e^{\log(odds)}}{1 + e^{\log(odds)}}. \end{aligned}$$

En nuestro caso p será $P(Y = 1|X_1, \ldots, X_n)$, que para un registro $(x_1, \ldots, x_n, y)$ denotamos $p(\mathbf{x}) := P(Y = 1|X_1 = x_1, \ldots, X_n = x_n)$. Además, en la regresión logística el valor de $\log(odds)$ se aproxima mediante una combinación lineal de $X_1, \ldots, X_n$, esto es, $\log(odds) = \omega_0 + \omega_1 x_1 + \ldots \omega_d x_d$ (en esta relación a los valores $\boldsymbol{\omega} = (\omega_0, \ldots, \omega_d)$ les llamamos pesos similarmente a los modelos de regresión lineal). Así:

$$p(\mathbf{x}) = \frac{e^{\omega_0 + \omega_1 x_1 + \cdots + \omega_d x_d}}{1 + e^{\omega_0 + \omega_1 x_1 + \cdots + \omega_d x_d}}. \tag{3.5}$$

Para estimar los parámetros $\boldsymbol{\omega}$, supongamos que tenemos D una base de datos que con n registros $d + 1$-dimensional, tal que cada registro es de la forma $t_i = (x_{i1}, x_{i2}, \ldots, x_{id}, y_i)$ tal que $y_i \in \{0, 1\}$ para $i = 1, \ldots, n$. Como Y toma valores en $\{0, 1\}$ es ideal tomar $Y_i \sim Ber(p(\mathbf{x_i}))$. Así, podemos construir la función de verosimilitud y estimar $\boldsymbol{\omega}$ por máxima verosimilitud. La función de verosimilitud es la siguiente:

$$L(\boldsymbol{\omega}) = \prod_{t_i \in D} p(\mathbf{x}_i)^{y_i} \cdot (1 - p(\mathbf{x}_i))^{1-y_i}.$$

Luego, podemos obtener la función de log-verosimilitud:

$$l(\boldsymbol{\omega}) = \log(L(\boldsymbol{\omega})) = \sum_{i=1}^{n} y_i \log(p(\mathbf{x}_i)) + (1 - y_i) \log(1 - p(\mathbf{x}_i)).$$

Sustituyendo cada $p(\mathbf{x}_i)$ en su forma (3.5) tenemos:

$$\begin{aligned}
l(\boldsymbol{\omega}) &= \sum_{i=1}^{n} y_i \log\left(\frac{1}{1 + e^{-\boldsymbol{\omega}\mathbf{x}_i}}\right) + (1 - y_i) \log\left(\frac{e^{-\boldsymbol{\omega}\mathbf{x}_i}}{1 + e^{-\boldsymbol{\omega}\mathbf{x}_i}}\right) \\
&= \sum_{i=1}^{n} y_i \log\left(\frac{1}{1 + e^{-\boldsymbol{\omega}\mathbf{x}_i}}\right) + \log\left(\frac{e^{-\boldsymbol{\omega}\mathbf{x}_i}}{1 + e^{-\boldsymbol{\omega}\mathbf{x}_i}}\right) - y_i \log\left(\frac{e^{\boldsymbol{\omega}\mathbf{x}_i}}{1 + e^{-\boldsymbol{\omega}\mathbf{x}_i}}\right) \\
&= \sum_{i=1}^{n} y_i \left(\log\left(\frac{1}{1 + e^{-v\mathbf{x}_i}}\right) - \log\left(\frac{e^{-\boldsymbol{\omega}\mathbf{x}_i}}{1 + e^{-\boldsymbol{\omega}\mathbf{x}_i}}\right)\right) + \log\left(\frac{e^{-\boldsymbol{\omega}\mathbf{x}_i}}{1 + e^{-\boldsymbol{\omega}\mathbf{x}_i}}\right) \\
&= \sum_{i=1}^{n} y_i \log\left(\frac{\frac{1}{1+e^{-\boldsymbol{\omega}\mathbf{x}_i}}}{\frac{e^{-\boldsymbol{\omega}\mathbf{x}_i}}{1+e^{-\boldsymbol{\omega}\mathbf{x}_i}}}\right) + \log\left(\frac{e^{-\omega\mathbf{x}_i}}{1 + e^{-\boldsymbol{\omega}\mathbf{x}_i}}\right) \\
&= \sum_{i=1}^{n} y_i \log(e^{\boldsymbol{\omega}\mathbf{x}_i}) + \log\left(\frac{1}{1 + e^{\boldsymbol{\omega}\mathbf{x}_i}}\right) \\
&= \sum_{i=1}^{n} y_i \boldsymbol{\omega}\mathbf{x}_i + \log\left(\frac{1}{1 + e^{\boldsymbol{\omega}\mathbf{x}_i}}\right) \\
&= \sum_{i=1}^{n} y_i \boldsymbol{\omega}\mathbf{x}_i - \log(1 + e^{\boldsymbol{\omega}\mathbf{x}_i}).
\end{aligned}$$

Así, la estimación de $\boldsymbol{\omega}$ será tomar

$$\boldsymbol{\omega}^* \in \arg\max \sum_{i=1}^{n} y_i \boldsymbol{\omega}\mathbf{x}_i - \log(1 + e^{\boldsymbol{\omega}\mathbf{x}_i})$$

$$= \arg\min \sum_{i=1}^{n} \log(1 + e^{\boldsymbol{\omega}\mathbf{x}_i}) - y_i \boldsymbol{\omega}\mathbf{x}_i.$$

Similarmente, como se realizó en regresión lineal, podemos considerar como función objetivo

$$f(t_i, \boldsymbol{\omega}) = \log(1 + e^{\boldsymbol{\omega}\mathbf{x}_i}) - y_i \boldsymbol{\omega}\mathbf{x}_i$$

y

$$f_D(\boldsymbol{\omega}) = \sum_{i=1}^{n} f(t_i, \boldsymbol{\omega}) = \sum_{i=1}^{n} \log(1 + e^{\boldsymbol{\omega}x_i}) - y_i \boldsymbol{\omega}\mathbf{x}_i.$$

A diferencia de la función objetivo en regresión lineal, la forma de esta nueva función de costos no permite pasarla a una versión polinómica. Sin embargo, Zhang [45] propone considerar una aproximación de $f(t_i, \boldsymbol{\omega})$ como en (3.6) y esta pasarla a un polinomio para luego hacer la perturbación.

$$\hat{f}(t_i, \boldsymbol{\omega}) = \sum_{k=0}^{2} \frac{f_1^{(k)}(0)}{k!} (\mathbf{x}_i \boldsymbol{\omega})^k - y_i \mathbf{x}_i \boldsymbol{\omega} \quad \text{donde} \quad f_1(z) = \log(1 + \exp(z)). \tag{3.6}$$

Así, podemos hacer el desarrollo para $\hat{f}_D(\boldsymbol{\omega}) = \sum_{i=1}^{n} \hat{f}(t_i, \boldsymbol{\omega})$:

Ejemplo 3.4. *Considerando como función de costos* $\hat{f}(t_i, \boldsymbol{\omega}) = \sum_{k=0}^{2} \frac{f_1^{(k)}(0)}{k!} (x_i \boldsymbol{\omega})^k - y_i x_i \boldsymbol{\omega}$*, tal función puede ser expresada en su forma polinómica como sigue:*

$$\begin{aligned}
\hat{f}(t_i, \boldsymbol{\omega}) &= \sum_{k=0}^{2} \frac{f_1^{(k)}(0)}{k!} (\mathbf{x}_i \boldsymbol{\omega})^k - y_i \mathbf{x}_i \boldsymbol{\omega} \\
&= \left[\log(1 + \exp(0)) + \frac{e^0}{1 + e^0} (\mathbf{x}_i \boldsymbol{\omega}) + \frac{e^0}{(1 + e^0)^2} \cdot \frac{1}{2} (\mathbf{x}_i \boldsymbol{\omega})^2 - y_i \mathbf{x}_i \boldsymbol{\omega} \right] \\
&= \left[\log(2) + \frac{1}{2} (\mathbf{x}_i \boldsymbol{\omega}) + \frac{1}{8} (\mathbf{x}_i \boldsymbol{\omega})^2 \right] - y_i \mathbf{x}_i \boldsymbol{\omega} \\
&= \log(2) + \frac{1}{2} (\mathbf{x}_i \boldsymbol{\omega}) + \frac{1}{8} (\mathbf{x}_i \boldsymbol{\omega})^2 - y_i \mathbf{x}_i \boldsymbol{\omega} \\
&= \log(2) + (\frac{1}{2} \mathbf{x}_i - y_i \mathbf{x}_i) \boldsymbol{\omega} + \frac{1}{8} (x_i \boldsymbol{\omega})^2 \\
&= \log(2) + \sum_{j=1}^{d} (\frac{1}{2} x_{ij} - y_i x_{ij}) \omega_j + \sum_{1 \leq j,l \leq d} \frac{1}{8} x_{ij} x_{il} \omega_j \omega_l.
\end{aligned}$$

Identificamos los elementos del polinomio (3.2) como:

- $j = 0$

$$\sum_{\phi \in \Phi_0} \lambda_{\phi t_i} \phi(\boldsymbol{\omega}) = \log(2)$$

si $\phi \in \Phi_0 = \{1\}$, $\lambda_{\phi t_i} = \log(2)$ *y* $\phi(\boldsymbol{\omega}) = 1$

- $j = 1$

$$\sum_{\phi \in \Phi_1} \lambda_{\phi t_i} \phi(\boldsymbol{\omega}) = \sum_{j=1}^{d} (\frac{1}{2} x_{ij} - y_i x_{ij}) \omega_j$$

si $\phi \in \Phi_1$, $\lambda_{\phi t_i} = \frac{1}{2} x_{ij} - y_i x_{ij}$ *y* $\phi(\boldsymbol{\omega}) = \omega_j$ *para* $j \in \{1, 2, \ldots, d\}$

- $j = 2$

$$\sum_{\phi \in \Phi_2} \lambda_{\phi t_i} \phi(\boldsymbol{\omega}) = \sum_{1 \leq j,l \leq d} \frac{1}{8} x_{ij} x_{il} \omega_j \omega_l$$

si $\phi \in \Phi_2$, $\lambda_{\phi t_i} = \frac{1}{8} x_{ij} x_{il}$ *y* $\phi(\boldsymbol{\omega}) = \omega_j \omega_l$ *para* $j, l \in \{1, 2, \ldots, d\}$.

Luego

$$\begin{aligned}
\hat{f}_D(\boldsymbol{\omega}) &= \sum_{i=1}^{n} \hat{f}(t_i, \boldsymbol{\omega}) \\
&= \sum_{i=1}^{n} [\log(2) + \sum_{j=1}^{d} (\frac{1}{2} x_{ij} - y_i x_{ij}) \omega_j + \sum_{1 \leq j,l \leq d} \frac{1}{8} x_{ij} x_{il} \omega_j \omega_l] \\
&= \sum_{i=1}^{n} \log(2) + \sum_{i=1}^{n} \sum_{j=1}^{d} (\frac{1}{2} x_{ij} - y_i x_{ij}) \omega_j + \sum_{i=1}^{n} \sum_{1 \leq j,l \leq d} \frac{1}{8} x_{ij} x_{il} \omega_j \omega_l \\
&= \sum_{i=1}^{n} \log(2) + \sum_{j=1}^{d} \sum_{i=1}^{n} (\frac{1}{2} x_{ij} - y_i x_{ij}) \omega_j + \sum_{1 \leq j,l \leq d} \sum_{i=1}^{n} \frac{1}{8} x_{ij} x_{il} \omega_j \omega_l.
\end{aligned}$$

De aquí, se puede identificar que:

- *si* $\phi \in \Phi_0 = \{1\}$

$$\lambda_\phi = \sum_{i=1}^{n} \lambda_{\phi t_i} = \sum_{i=1}^{n} \log(2)$$

- *si* $\phi \in \Phi_1$

$$\lambda_\phi = \sum_{i=1}^{n} \lambda_{\phi t_i} = \sum_{j=1}^{d} \sum_{i=1}^{n} (\frac{1}{2} x_{ij} - y_i x_{ij})$$

para $j \in \{1, 2, \ldots, d\}$

- *si* $\phi \in \Phi_2$

$$\lambda_\phi = \sum_{i=1}^{n} \lambda_{\phi t_i} = \sum_{1 \le j,l \le d} \sum_{i=1}^{n} \frac{1}{8} x_{ij} x_{il}$$

para $j, l \in \{1, 2, \ldots, d\}$.

De manera similar a lo realizado en regresión lineal, es necesario perturbar los coeficientes. Sin embargo, para este caso, Zhang propone en [45] perturbar cada uno de los coeficientes λ_ϕ agregando un ruido $Lap(\frac{\Delta}{\epsilon})$ pero ahora siendo

$$\Delta = 2 \max_{t_i \in D} \left(\frac{f_1^{(1)}(0)}{1!} \sum_{j=1}^{d} x_j + \frac{f_1^{(2)}(0)}{2!} \sum_{1 \le j,l \le d} x_j x_l + y \sum_{j=1}^{d} x_j \right).$$

Esto queda resumido en el algoritmo 4.

Algoritmo 4 Algoritmo Mecanismo Funcional Polinomio Infinito [45]

Entrada Base de datos D, función objetivo $f_D(\boldsymbol{\omega}) = \sum_{i=1}^{n} f(t_i, \boldsymbol{\omega})$, parámetro de privacidad ϵ

Salida Parámetros del modelo perturbado $\boldsymbol{\omega}^*$

1: Descomponer la función en la forma $f(t_i, \boldsymbol{\omega}) = f_1(g_1(t_i, \omega))$
2: Considerar la aproximación $\hat{f}(t_i, \omega) = \sum_{k=0}^{2} \frac{f_1^{(k)}(0)}{k!} (\mathbf{x}_i \boldsymbol{\omega})^k - y_i \mathbf{x}_i \boldsymbol{\omega}$
3: Correr el Algoritmo 3 con las entradas $(D, \hat{f}, \epsilon)$
4: **return** $\boldsymbol{\omega}^*$ del Algoritmo 3

Ahora, veamos que este algoritmo cumple con la definición de DP.

Teorema 3.2. *El algoritmo 4 satisface ϵ-DP*

Demostración. La demostración se sigue de que las primeras dos líneas no dependen de la observación de la base de datos D, y el Algoritmo 3 satisface ϵ-DP según el Teorema 3.1, así según el Teorema 1.3 se sigue que el Algoritmo 4 satisface ϵ-DP. □

Ejemplo 3.5. *Ahora, recordando que cada valor en la base de datos está en el intervalo* $[0, 1]$*, determinemos* Δ *continuando el Ejemplo 3.4:*

$$\begin{aligned}
\Delta &= 2 \max_t \sum_{j=1}^{J} \sum_{\phi \in \Phi_j} |\lambda_{\phi t}| \\
&= 2 \max_t \left(\sum_{j=1}^{d} |(\frac{1}{2} x_{ij} - y_i x_{ij})| + \sum_{1 \le j,l \le d} |\frac{1}{8} x_j x_l| \right) \\
&\le 2(\frac{d}{2} + d + \frac{d^2}{8}) \\
&= \frac{d^2}{4} + 3d.
\end{aligned}$$

Luego nuestros coeficientes con ruido $\tilde{\lambda_\phi} = \lambda_\phi + Lap(\frac{\Delta}{\epsilon})$*, es decir:*

- *si* $\phi \in \Phi_0 = \{1\}$

$$\tilde{\lambda_\phi} = \sum_{i=1}^{n} \log(2) + Lap(\frac{\Delta}{\epsilon})$$

- *si* $\phi \in \Phi_1$

$$\tilde{\lambda_\phi} = \sum_{i=1}^{n} (\frac{1}{2} x_{ij} - y_i x_{ij}) + Lap(\frac{\Delta}{\epsilon})$$

  para $j \in \{1, 2, \ldots, d\}$

- *si* $\phi \in \Phi_2$

$$\tilde{\lambda_\phi} = \sum_{i=1}^{n} \frac{1}{8} x_{ij} x_{il} + Lap(\frac{\Delta}{\epsilon})$$

  para $j, l \in \{1, 2, \ldots, d\}$.

Luego, nuestra nueva función objetivo con ruido es:

$$\begin{aligned}
\tilde{\hat{f}}_D(\boldsymbol{\omega}) &= \sum_{k=0}^{J} \sum_{\phi \in \Phi_k} \tilde{\lambda_\phi} \phi(\boldsymbol{\omega}) \\
&= \sum_{k=0}^{J} \sum_{\phi \in \Phi_k} (\lambda_{\phi t_i} + Lap(\frac{\Delta}{\epsilon})) \phi(\boldsymbol{\omega}) \\
&= \sum_{i=1}^{n} \log(2) + Lap(\frac{\Delta}{\epsilon}) + \sum_{j=1}^{d} \left(\left(\sum_{i=1}^{n} (\frac{1}{2} x_{ij} - y_i x_{ij}) \right) + Lap(\frac{\Delta}{\epsilon}) \right) \omega_j \\
&\qquad + \sum_{1 \le j,l \le d} \left(\left(\sum_{i=1}^{n} \frac{1}{8} x_{ij} x_{il} \right) + Lap(\frac{\Delta}{\epsilon}) \right) \omega_j \omega_l.
\end{aligned}$$

Sobre esta última calculamos el valor donde alcanza su mínimo para así obtener $\tilde{\boldsymbol{\omega}} \in \arg\min_{\boldsymbol{\omega}} \overline{f}_D(\boldsymbol{\omega})$*, pero como sabemos que* $\Delta = \frac{d^2}{4} + 3d$*, entonces nuestra función objetivo es:*

$$\begin{aligned}
\tilde{\hat{f}}_D(\boldsymbol{\omega}) = \sum_{i=1}^{n} \log(2) + Lap\left(\frac{\frac{d^2}{4} + 3d}{\epsilon} \right) \\
+ \sum_{j=1}^{d} \left(\left(\sum_{i=1}^{n} \left(\frac{1}{2} x_{ij} - y_i x_{ij} \right) \right) + Lap\left(\frac{\frac{d^2}{4} + 3d}{\epsilon} \right) \right) \omega_j \\
+ \sum_{1 \le j,l \le d} \left(\left(\sum_{i=1}^{n} \frac{1}{8} x_{ij} x_{il} \right) + Lap\left(\frac{\frac{d^2}{4} + 3d}{\epsilon} \right) \right) \omega_j \omega_l.
\end{aligned}$$

Con esto hecho, podemos aplicar el modelo de regresión logística con DP. En la Figura 3.4 se muestran las gráficas al buscar una relación entre el nivel de glucosa de una persona y su predicción sobre si es posible que padezca de diabetes. En esta, se puede ver la comparativa de un modelo tradicional de regresión logística y uno considerando su versión con DP. Ambas curvas siguen una tendencia similar, ajustándose bien en la región donde se concentra la mayor parte de los datos. Aunque ambas regresiones siguen una tendencia similar, la versión con DP presenta un ajuste menos preciso en la región derecha, ya que incrementa significativamente las probabilidades en esta área. Además, al considerar la función objetivo definida en el Ejemplo 3.4, se observa que el error reportado es negativo para ambos casos, pero sigue siendo menor en el modelo de regresión tradicional, lo cual indica un mejor ajuste. Esto refleja un ajuste menos preciso cuando se introduce privacidad diferencial. Así, aunque el modelo con DP presenta un mayor error y pierde precisión, especialmente en los valores extremos, sigue manteniendo un ajuste razonable en la región central de los datos. Esto sugiere que, a pesar de la perturbación introducida por la privacidad diferencial, el modelo sigue siendo funcional.

3.3. Fuzzy c-Means

La teoría de conjuntos difusos propuesta por Zadeh [43] en 1965 captura la idea de la incertidumbre que se produce al querer capturar información del mundo real dentro de un lenguaje matemático. Mientras que en la teoría clásica se define la pertenencia de los distintos elementos a un conjunto haciéndoles corresponder el valor 1 si pertenecen y 0 en caso contrario, en un conjunto difuso se ha de definir una función que asocie el grado de pertenencia al conjunto. Así, si quisiéramos inducir una partición sobre un conjunto de datos, en lugar de determinar si un elemento pertenece o no a un subconjunto, daremos un grado o nivel de pertenencia a cada uno de estos. A este tipo de clasificación le llamamos clasificación difusa. Ahora bien, la libertad o la tolerancia a que un punto pueda pertenecer a varios grupos está controlada por el coeficiente de fuzzificación $m \in [1, +\infty)$ [8], normalmente se establece en $m = 2$ para mejores resultados [13].

En la teoría clásica de conjuntos para un conjunto no vacío X podemos construir un subconjunto a partir de una función $u : X \to \{0, 1\}$ de la siguiente forma:

$$A = \{x \in X | u(x) = 1\}.$$

La función anterior recibe el nombre de función característica asociada al conjunto A. Ahora bien, en el contexto de conjuntos difusos se extiende el rango de la función u al intervalo $[0, 1]$ de modo que un elemento puede "parcialmente" pertenecer al conjunto determinado por u, para ello se introduce la siguiente definición.

Definición 3.2. *Dado X un conjunto no vacío y $A \subseteq X$, definimos un subconjunto difuso de X como una función $u : X \to [0, 1]$, también llamada función de pertenencia.*

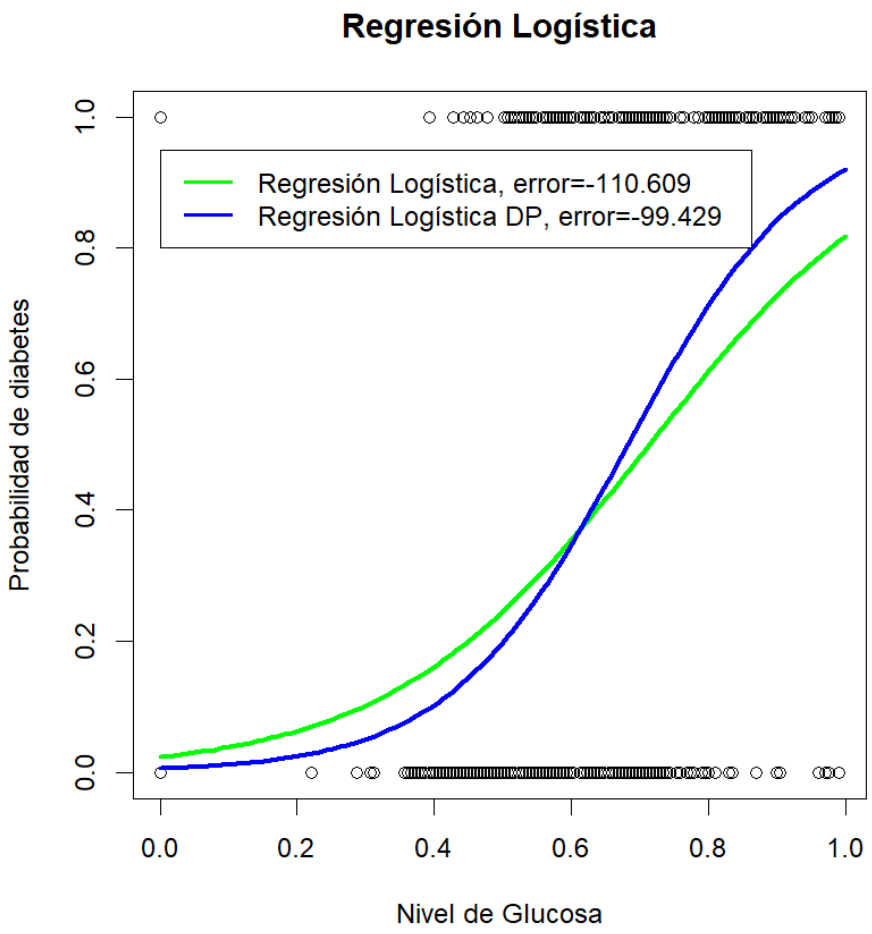


Figura 3.4: Regresión Logística en Datos Diabetes: comparación entre la versión normal y con privacidad diferencial para $\epsilon = 1$.

Definición 3.3. *Sea X un conjunto no vacío y $A \subseteq X$. Sean u_A y $\tilde{u}_A$ funciones de membresía asociadas al conjunto A. Diremos que $(u_A, \tilde{u}_A)$ forman una 2-partición difusa de X si para todo $x \in X$:*

$$u_A(x) + \tilde{u}_A(x) = 1, \tag{3.7}$$

$$0 < u_A(x) < 1. \tag{3.8}$$

Si quisiéramos extender la definición anterior a un tamaño digamos c un entero positivo, las condiciones (3.7) y (3.8) son equivalentes a que para todo $x \in X$:

$$\sum_{i=1}^{c} u_i(x) = 1, \tag{3.9}$$

$$0 < u_i(x) < 1. \tag{3.10}$$

Donde para cada $i = 1, 2, \ldots, n$, u_i son funciones de membresía asociadas a algún conjunto A_i.

En particular, si $X = \{x_1, \ldots, x_n\}$, es conveniente denotar $u_{ik} = u_i(x_k)$, con $i = 1, \ldots, c$ y $k = 1, \ldots, n$. De esta forma, las condiciones (3.9) y (3.10) se reescriben como:

$$\sum_{i=1}^{c} u_{ik} = 1 \quad \text{para } k = 1, \ldots, n \tag{3.11}$$

$$0 < \sum_{k=1}^{n} u_{ik} < n \quad \text{para } i = 1, \ldots, c. \tag{3.12}$$

Observación 3.1. *Cada renglón i de la matriz $U = [u_{ik}]_{n\times c}$ representa los valores de la i-ésima función de pertenencia u_i (esto es, el i-ésimo subconjunto difuso) de la c-partición difusa U de X.*

Los algoritmos de clasificación difusa consisten en asignar a un conjunto de datos una matriz U que represente una partición difusa, según existan similitudes entre los datos.

Uno de los algoritmos más conocidos para la clasificación difusa es el de *Fuzzy c-means*, propuesto por Dunn en 1973 [10]. En este algoritmo los grupos (o clusters) quedan determinados por unos puntos llamados centros (o centroides).

Supongamos que contamos con una base de datos $D = \{y_1, y_2, \ldots, y_n\} \subseteq \mathbb{R}^d$, un conjunto de k centroides iniciales $C = \{c_1, \ldots, c_k\} \subseteq \mathbb{R}^d$, $m \in [1, \infty)$ el coeficiente de fuzzificación y una matriz de pertenencia inicial $U = [u_{ij}]_{n\times k}$ en la que cada entrada u_{ij} representa la pertenencia del i-ésimo registro al j-ésimo cluster. El algoritmo *Fuzzy c-means* sigue un proceso iterativo, de este modo, según Bezdek en 1981 [5] propone que en cada iteración elijamos a C y U tal que se minimice la función objetivo (3.13) según la condición (3.14).

$$J(U, C) = \sum_{i=1}^{n}\sum_{j=1}^{k} u_{ij}^m ||y_i - c_j||^2 \tag{3.13}$$

$$\sum_{j=1}^{k} u_{ij} = 1, \forall i = 1, \ldots, n. \tag{3.14}$$

El cálculo de los valores para minimizar plantea un problema de optimización usando multiplicadores de Lagrange que según [5] tiene por solución:

$$u_{ij} = \frac{1}{\sum_{l=1}^{k} \left(\frac{||x_i - c_j||}{||x_i - c_l||}\right)^{\frac{2}{m-1}}} \tag{3.15}$$

$$c_j = \frac{\sum_{i=1}^{n} u_{ij}^m x_i}{\sum_{i=1}^{n} u_{ij}^m}. \tag{3.16}$$

Con estos valores óptimos, se construye el Algoritmo 5 de *Fuzzy c-means* (FcM).

Algoritmo 5 Algoritmo FcM [46]

Entrada Base de datos D, cantidad de clusters k, coeficiente de fuzzificación m, número de iteraciones t
Salida Conjunto de centroides C

1: Se inicializan aleatoriamente unos valores para U
2: **for** k=1 hasta t **do**
3: $(c_j)^k = \frac{\sum_{i=1}^{n} u_{ij}^m x_i}{\sum_{i=1}^{n} u_{ij}^m}$
4: $(u_{ij})^k = \frac{1}{\sum_{l=1}^{k} \left(\frac{||x_i - c_j||}{||x_i - c_l||} \right)^{\frac{2}{m-1}}}$
5: **end for**
6: **return** C

La parte vulnerable en este proceso es que, si para algún registro se conocen sus distancias hacia cada uno de los centroides en cada iteración, un atacante malicioso podría inferir el valor de un atributo específico de este registro.

En [34] se propone que en cada iteración se haga una perturbación tanto en las entradas de la matriz de pertenencia como también en los centroides de los cluster, esto según el Algoritmo 6.

Algoritmo 6 Algoritmo DPFCM [34]

Entrada Base de datos D, cantidad de clusters k, parámetro de privacidad ϵ, número de iteraciones t
Salida Conjunto de clusters C

1: Se inicializan aleatoriamente c centroides para conformar $C = (v_1, v_2, \ldots, v_c)$
2: $N = |D|$
3: $\varepsilon = \frac{\epsilon}{tc(2+d)}$
4: Se repiten los pasos 5 a 8 hasta alcanzar t iteraciones
5: $u_{ij} = \frac{1}{\sum_{l=1}^{k} \left(\frac{||x_i - c_j|| + Lap(\frac{3\sqrt{d}}{\varepsilon})}{||x_i - c_l|| + Lap(\frac{3\sqrt{d}}{\varepsilon})} \right)^{\frac{1}{m-1}}}$
6: Normalice cada renglón de U tal que sus entradas estén en $[0, 1]$ y $\sum_{i=1}^{c} u_{ki} = 1$ para cada $k = 1, \ldots, N$
7: $c_{ij} = \frac{\sum_{k=1}^{n} u_{ki}^m x_{kj}}{\sum_{k=1}^{n} u_{ki}^m} + Lap(\frac{3}{\varepsilon})$
8: Substituya 1 para valores $v_{ij} > 1$ y 0 para valores $v_{ij} < 0$.
return C

Con esto hecho, podemos aplicar el modelo de *Fuzzy c-means* con DP. En esta sección, primero observamos la Figura 3.5, en la que se presentan los casos positivos y negativos de diabetes y su relación con el índice de masa corporal (IMC) y el nivel de glucosa de cada individuo. Se puede notar que, en la parte central de la gráfica, existe una región donde los casos positivos y negativos se mezclan, lo que dificultaría

predecir a qué grupo pertenecería un punto situado en esa zona si no se conoce su etiqueta. Por esta razón, se opta por realizar una clasificación difusa, que asigna un nivel de pertenencia a cada grupo en lugar de una clasificación rígida.

En la Figura 3.6, se presenta el resultado de la clasificación difusa utilizando dos clusters. El color de cada punto representa la combinación de su pertenencia a cada cluster, y se puede observar su relación con la figura anterior. Finalmente, en la Figura 3.7, se muestra la clasificación difusa con privacidad diferencial (DP), donde se compara con la clasificación normal, permitiendo observar las diferencias en la ubicación de los centroides.

La razón de usar dos clusters es que se busca aproximar los grupos de positivos y negativos. Además, la clasificación difusa permite, en la región previamente incierta, asignar una pertenencia parcial a cada cluster. Como se puede apreciar en las gráficas, esto se cumple de manera efectiva tanto en la clasificación difusa como en la variante con DP. En esta última, al introducir el ruido por privacidad diferencial, se observa que los centroides cambian ligeramente de ubicación, pero la clasificación sigue siendo consistente con la estructura deseada.

Así, al tener un nuevo punto sin etiqueta, podríamos usar el modelo de *Fuzzy c-means* para obtener una clasificación difusa que no solo prediga si ese punto es diabético o no, sino también el grado de certeza asociado a esa predicción.

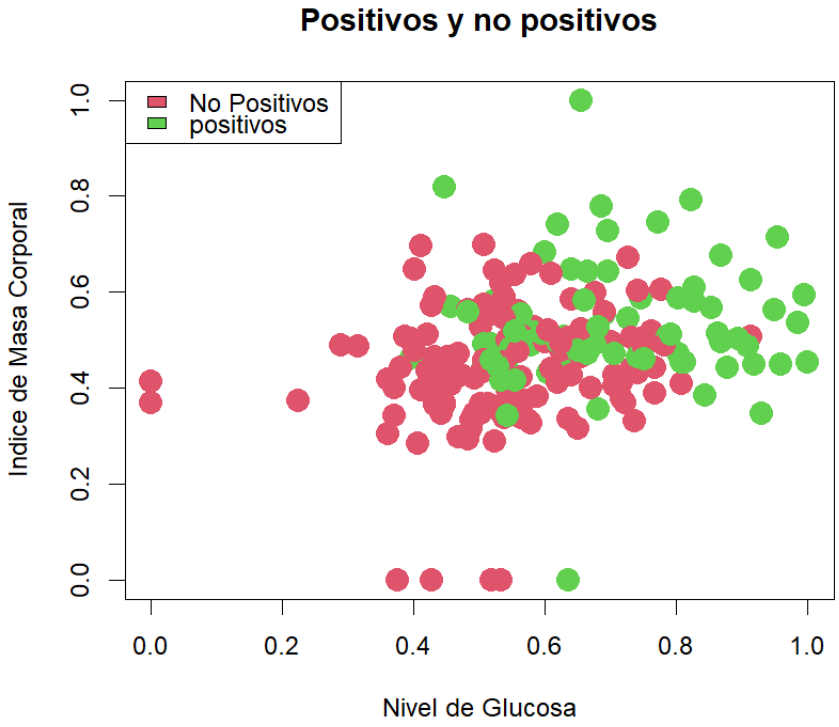


Figura 3.5: Positivos y Negativos de Diabetes.

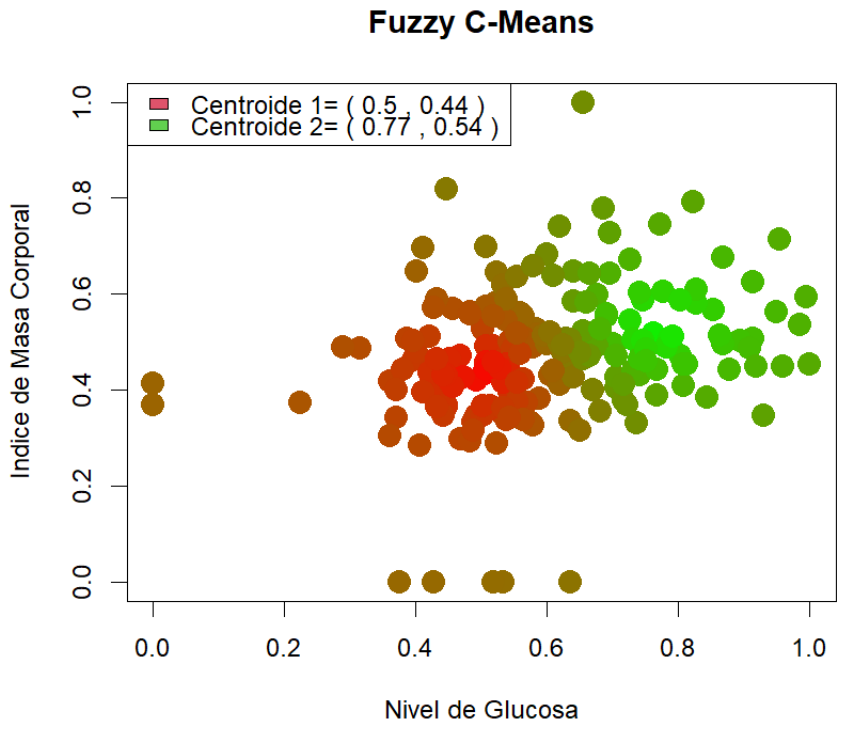


Figura 3.6: Fuzzy c-Means a Datos de Diabetes.

3.4. Clasificación Naive Bayes

El razonamiento bayesiano proporciona un enfoque probabilístico para la inferencia. Se basa en la suposición de que las cantidades de interés están regidas por distribuciones de probabilidad y que se pueden tomar decisiones óptimas razonando sobre estas probabilidades junto con los datos observados. Es importante para el aprendizaje automático porque proporciona un enfoque cuantitativo para evaluar la evidencia que respalda hipótesis alternativas [2].

Los métodos de aprendizaje bayesiano son relevantes en el estudio de aprendizaje de máquina ya que estos calculan explícitamente las probabilidades de hipótesis además se encuentran entre los enfoques más prácticos para ciertos tipos de problemas de aprendizaje [30].

Un ejemplo de estos métodos es el Clasificador *Naive Bayes*, tal método hace un supuesto de independencia condicional que casi siempre es incorrecto. Esta suposición incorrecta le otorga al clasificador la designación de "ingenuo". Tal supuesto simplifica enormemente los cálculos y la aplicación se vuelve muy rápida [33].

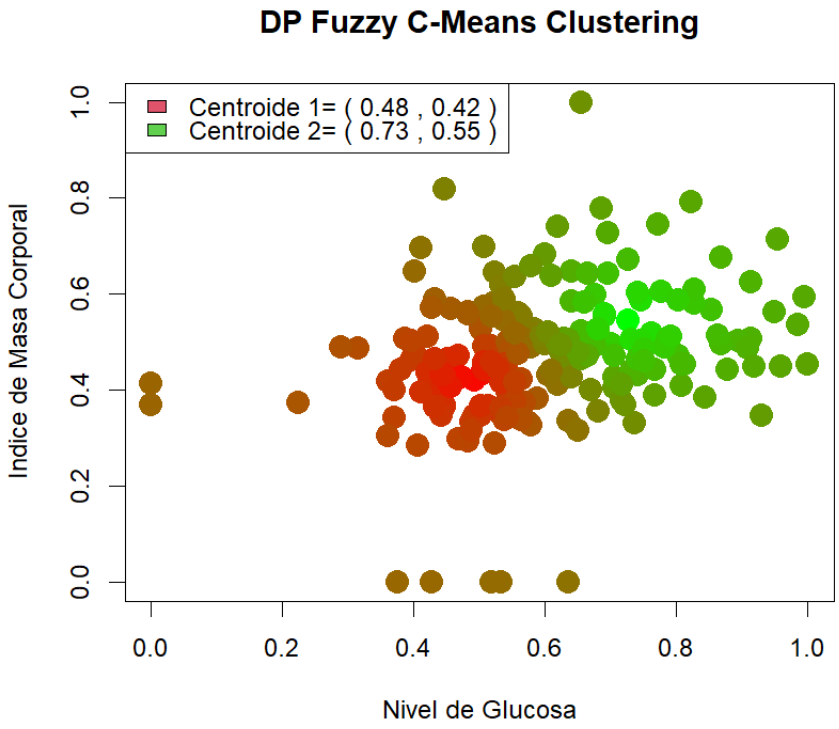


Figura 3.7: DP Fuzzy c-Means a Datos de Diabetes considerando $\epsilon = 1$.

Teorema de Bayes

Sea $(\Omega, \mathcal{F}, \mathbb{P})$ un espacio de probabilidad y sea $\mathcal{H} \subseteq \mathcal{F}$ el cual representa el conjunto de hipótesis. Escribiremos $\mathbb{P}(H)$ para denotar la probabilidad inicial de que la hipótesis $H \in \mathcal{H}$ sea cierta, antes de haber observado los datos de entrenamiento. $\mathbb{P}(H)$ a menudo se llama la probabilidad a priori de H y puede reflejar cualquier conocimiento de fondo que tengamos sobre la posibilidad de que H sea una hipótesis correcta. Si no tenemos tal conocimiento previo, entonces podríamos simplemente asignar la misma probabilidad a priori a cada hipótesis candidata. De manera similar, escribiremos $\mathbb{P}(E)$ con $E \in \mathcal{F}$ para denotar la probabilidad a priori de que la evidencia E sea observada (es decir, la probabilidad de E sin conocimiento sobre cuál hipótesis es cierta y representa la información observada que afecta nuestras creencias sobre las hipótesis). Luego, escribiremos $\mathbb{P}(E|H)$ para denotar la probabilidad de observar la evidencia E dado un escenario en el que la hipótesis H es cierta. En los problemas de aprendizaje automático, estamos interesados en la probabilidad $\mathbb{P}(H|E)$ de que H sea cierta dada la evidencia E. $\mathbb{P}(H|E)$ se llama la probabilidad a posteriori de H, porque refleja nuestra confianza en que H sea cierta después de haber visto la evidencia E. Nótese que la probabilidad a posteriori $\mathbb{P}(H|E)$ refleja la influencia de la evidencia E, en contraste con la probabilidad a priori $\mathbb{P}(H)$, que

es independiente de E.

El teorema de Bayes es importante para los métodos de aprendizaje bayesianos porque proporciona una manera de calcular la probabilidad a posteriori $\mathbb{P}(H|E)$, a partir de la probabilidad a priori $\mathbb{P}(H)$, junto con $\mathbb{P}(E)$ y $\mathbb{P}(E|H)$. Esta relación está dada en (3.17).

$$\mathbb{P}(H|E) = \frac{\mathbb{P}(E|H)\mathbb{P}(H)}{\mathbb{P}(E)}. \tag{3.17}$$

En muchos escenarios de aprendizaje se está interesado en encontrar la hipótesis más probable $H \in \mathcal{H}$ dada la evidencia E. Cualquier hipótesis máximamente probable se llama una hipótesis de máxima a posteriori (MAP). Podemos determinar las hipótesis MAP utilizando el Teorema de Bayes para calcular la probabilidad a posteriori de cada hipótesis candidata. Más precisamente, diremos que H_{MAP} es una hipótesis MAP siempre que:

$$\begin{aligned} H_{MAP} &\in \arg\max_{H\in\mathcal{H}} \mathbb{P}(H|E) \\ &= \arg\max_{H\in\mathcal{H}} \frac{\mathbb{P}(E|H)\mathbb{P}(H)}{\mathbb{P}(E)} \\ &= \arg\max_{H\in\mathcal{H}} \mathbb{P}(E|H)\mathbb{P}(H). \end{aligned}$$

El objetivo del clasificador *Naive Bayes* es encontrar la probabilidad a posteriori $\mathbb{P}(Y = y \mid \mathcal{X} = \mathbf{x})$ para cada clase y, y clasificar la muestra $\mathbf{x} = (a_1, \ldots, a_d) \in \mathbb{R}^d$ en la clase con la probabilidad a posteriori más alta. En este caso, nuestro conjunto de hipótesis es $\mathcal{H} = \{[Y = c_j] : c_j \in C\}$ y nuestra evidencia es el evento $[\mathcal{X} = \mathbf{x}]$. Así, queremos encontrar

$$c_{MAP} = \arg\max_{c_j\in C} \mathbb{P}(Y = c_j|\mathcal{X} = \mathbf{x}).$$

Utilizando el teorema de Bayes, podemos escribir:

$$\begin{aligned} c_{MAP} &= \arg\max_{c_j\in C} \mathbb{P}(Y = c_j|\mathcal{X} = \mathbf{x}) \\ &= \arg\max_{c_j\in C} \frac{\mathbb{P}(\mathcal{X} = \mathbf{x}|Y = c_j)\mathbb{P}(Y = c_j)}{\mathbb{P}(\mathcal{X} = \mathbf{x})} \\ &= \arg\max_{c_j\in C} \mathbb{P}(\mathcal{X} = \mathbf{x}|Y = c_j)\mathbb{P}(Y = c_j) \end{aligned}$$

El clasificador *Naive Bayes* se basa en la suposición simplificada de que los valores de los atributos son condicionalmente independientes dado el valor objetivo. En otras palabras, la suposición es que, dado el valor objetivo de alguna muestra, la probabilidad de observar la conjunción $(X_1, \ldots, X_d) = (a_1, \ldots, a_d)$ dada cierta una hipótesis, es simplemente el producto de las probabilidades de los atributos individuales condicionados a tal hipótesis: $\mathbb{P}(\mathcal{X} = \mathbf{x}|Y = c_j) = \prod_{i=1}^{d} \mathbb{P}(X_i = a_i|Y = c_j)$. Luego, tenemos c_{NB} el clasificador *Naive Bayes*.

$$c_{NB} \in \arg\max_{c_j \in C} \mathbb{P}(Y = c_j) \prod_{i=1}^{d} \mathbb{P}(X_i = a_i | Y = c_j) \tag{3.18}$$

Necesitamos estimar las distribuciones de Y y $X_i|[Y = c_j]$ para $i = 1, \ldots, d$ y cada $c_j \in C$ a partir de los datos del conjunto de entrenamiento $\mathcal{D}$.

Las probabilidades a priori $\mathbb{P}(Y = c_j)$ típicamente se estiman con las frecuencias del conjunto de entrenamiento, esto es, $\mathbb{P}(Y = c_j) = \frac{\#\{\text{registros en } \mathcal{D}: Y = c_j\}}{|\mathcal{D}|}$.

Ahora, para estimar la distribución de cada $X_i|[Y = c_j]$ podemos ajustar una distribución de probabilidad $f_{ij}(x : \theta)$ y estimaremos a θ con los registros en la base de entrenamiento para los cuales $Y = c_j$, es decir, pertenecen a la clase c_j.

Observación 3.1. *En el caso en que tengamos una variable aleatoria continua X_k, consideraremos la función de densidad $f_{X_k|[Y=c_j]}(a_k)$ en lugar de $\mathbb{P}(X_k = a_k|Y = c_j)$ en (3.18).*

Podemos usar la Privacidad Diferencial en el modelo *Naive Bayes* de la siguiente forma. Recordemos que para estimar la distribución de cada $X_i|[Y = c_j]$ intentamos ajustar una distribución de probabilidad $f_{ij}(x : \theta)$ y estimaremos a θ con los registros en la base de entrenamiento para los cuales $Y = c_j$, es decir, pertenecen a la clase c_j. Para satisfacer $\epsilon-$DP perturbaremos a cada θ según el mecanismo de Laplace antes descrito tomando

$$\theta' = \theta + Lap(\Delta\theta/\epsilon). \tag{3.19}$$

Nuestro mayor problema estará en determinar $\Delta\theta$ y este cálculo dependerá de la distribución que queramos considerar. Por ejemplo, si consideramos una distribución normal $X \sim N(\mu, \sigma)$, para estimar los parámetros tomamos $\hat{\mu} = \overline{x}$ y $\hat{\sigma} = S$, la media y desviación estándar muestral, respectivamente. En este caso se tiene que $\Delta\overline{x} = (u - l)/n + 1$ y $\Delta S = \sqrt{n}(u - l)/n + 1$, en donde n es el tamaño de la base de datos de entrenamiento y u, l representan el máximo y mínimo valor que puede tomar X. Para un desarrollo detallado de este cálculo puede consultarse [37].

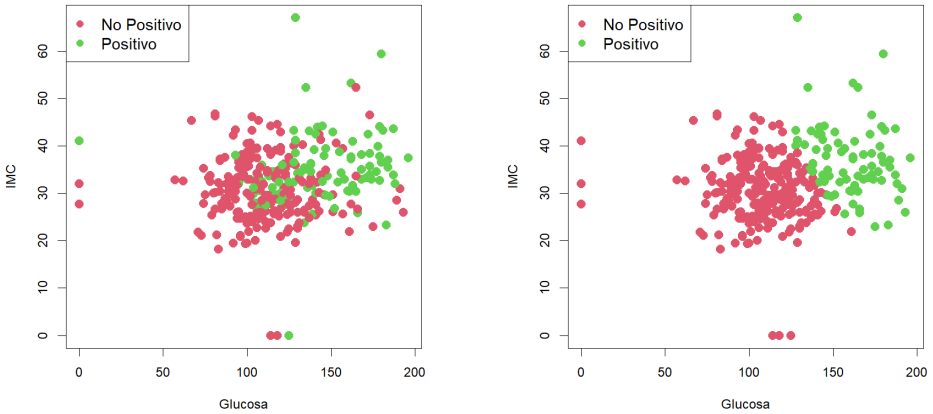
Se buscó crear un clasificador para determinar si una persona es diabética o no, $Y = 1$ y $Y = 0$, respectivamente, con base en dos variables que son el Índice de Masa Corporal (IMC) y su nivel de glucosa (variables X_1 y X_2, respectivamente), esto con los mismos datos usados anteriormente. Se ajustó cada una de las distribuciones condicionales, $X_1|[Y = 0]$, $X_1|[Y = 1]$, $X_2|[Y = 0]$ y $X_2|[Y = 1]$, a una distribución normal, esto basado en una prueba de normalidad de Kolmogorov-Smirnov [26] .

Al verificar el ajuste de las variables $X_1|[Y = 0]$, $X_1|[Y = 1]$, $X_2|[Y = 0]$ y $X_2|[Y = 1]$ a distribuciones normales se obtuvieron los p-valores de la Tabla 3.1 por lo que se acepta usar distribuciones normales usando como parámetros sus respectivas medias y desviaciones estándar muestrales.

Con estos parámetros construimos nuestro modelo clasificador como en (3.18). Al probarlo en nuestro conjunto de prueba se obtuvo una exactitud del 79 % y en la Figura 3.8 se presenta la clasificación estimada y real.

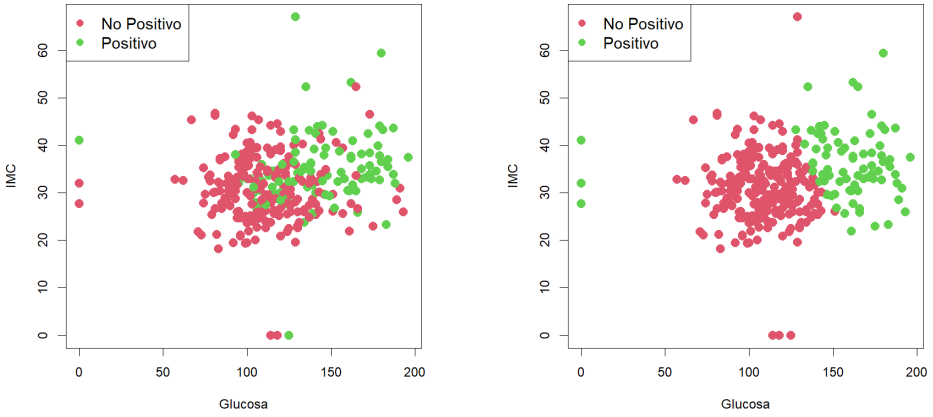
	mean	sd	p-value
$Glucosa\|[Y=0]$	108.67568	27.22728	0.21
$Glucosa\|[Y=1]$	137.8659	32.4038	0.93
$IMC\|[Y=0]$	30.603041	7.635663	0.33
$IMC\|[Y=1]$	35.139024	7.153218	0.13

Tabla 3.1: Estadísticas Descriptivas



(a) (b)

Figura 3.8: Resultado de Clasificación: Clase Verdadera (a), Predicción Naive Bayes (b).



(a) (b)

Figura 3.9: Resultado de Clasificación: Clase Verdadera (a), Predicción Naive Bayes con DP considerando $\epsilon = 1$ (b).

Ahora, como se mencionó anteriormente podemos optar por usar un mecanismo de Privacidad Diferencial que transforme los resultados de la Tabla 3.1 en una versión modificada a través de lo presentado en (3.19). Así, en la Tabla 3.2 se presenta una realización de la modificación DP *Naive Bayes* considerando $\epsilon = 0.5$.

	mean	sd
$Glucosa\|[Y=0]$	108.6949	26.9758
$Glucosa\|[Y=1]$	137.88405	31.26305
$IMC\|[Y=0]$	30.605681	8.632087
$IMC\|[Y=1]$	35.137254	7.828566

Tabla 3.2: Estadísticas Descriptivas con DP

Similarmente nuestro modelo clasificador y, al probarlo en el conjunto de prueba, se obtuvo una exactitud del 78 %. En la Figura 3.9 se presenta la clasificación estimada en su versión con privacidad diferencial (DP) y la clasificación real. Si bien la precisión del modelo se ve afectada por la adición de ruido, hemos logrado calibrar adecuadamente el parámetro ϵ, lo que permitió encontrar un equilibrio entre privacidad y exactitud. En este caso, la pérdida de precisión no es tan significativa, y el modelo sigue siendo funcional. Aunque la introducción de ruido no provocó un cambio importante en las gráficas, sí afectó las estadísticas presentadas en la Tabla 3.2. Sin embargo, este cambio no parece ser relevante desde el punto de vista práctico, ya que la precisión sigue siendo alta, similar a la del modelo original. Si el modelo tradicional tuviera una mayor exactitud, es muy probable que también lo tuviera su versión con privacidad diferencial. Del mismo modo, si el modelo tradicional fuera menos preciso, es razonable esperar que la versión con DP también lo fuera.

En esta sección hemos revisado cómo complementar algunos de los modelos de Aprendizaje Automático más conocidos, y como se ha visto, se obtienen buenos resultados incluso cuando son complementados con Privacidad Diferencial. En la aplicación de la privacidad diferencial en modelos de Aprendizaje hemos podido observar cómo se pueden combinar técnicas de aprendizaje automático con medidas de privacidad para proteger datos sensibles. Ahora bien, se debe reconocer la importancia de escoger un valor adecuado para el nivel de privacidad, pues si bien la precisión del modelo se ve afectada por la adición de ruido debemos encontrar una calibración adecuada del parámetro ϵ y lograr un equilibrio entre privacidad y exactitud.

En el siguiente capítulo se explorará cómo se puede entrenar modelos de aprendizaje, pero ahora sin la necesidad de tener un sólo conjunto de datos, sino abriendo la posibilidad de entrenar modelos con varios conjuntos de datos colaborando a la vez.

Capítulo 4

Aprendizaje Federado

A lo largo de este capítulo se desarrollará la teoría referente al Aprendizaje Federado (FL por sus siglas en inglés), se plantearán 3 técnicas y se complementará su uso con la introducción de Privacidad Diferencial. Primero revisaremos el enfoque tradicional de descenso por gradiente estocástico (SDG), una técnica ampliamente usada en el Aprendizaje de Máquina para optimizar funciones de costo. En una segunda sección se usará un enfoque por punto Proximal, el cual está relacionado con el método de regresión Ridge [16]. Por último, se dará un enfoque desde la teoría de grafos, en esta última parte se logra un aprendizaje descentralizado que complementado con la Privacidad Diferencial logra una seguridad muy alta para los usuarios.

El Aprendizaje Federado es un enfoque de aprendizaje automático donde los modelos son entrenados en dispositivos distribuidos, sin que los datos de entrenamiento abandonen los dispositivos locales. Esta configuración permite la colaboración entre múltiples dispositivos o instituciones sin compartir datos sensibles. Este modelo fue introducido por Google en 2016 [42] y funciona en esencia en una estructura como en la Figura 4.1. En esta configuración un servidor central inicializa los parámetros de un modelo que se quiere ajustar, luego cada cliente (aquellas personas o instituciones que cuentan con una base de datos) actualiza de manera local sus parámetros (parámetros locales) y devuelve estos al servidor central (sólo los parámetros, nunca los datos), luego este, habiendo recibido actualizaciones de todos los clientes actualiza los parámetros de manera general (parámetros globales) y el ciclo se repite de nuevo hasta alcanzar una cantidad predeterminada de iteraciones. Esta configuración permite que las bases de datos que posea cada cliente nunca salgan de ellos y así evitan compartir información sensible. Esta idea queda planteada en el Algoritmo 7, en este la función $Clientupdate(k, w_t)$ representa la forma en que el cliente k-ésimo actualizará sus parámetros a partir de los actuales, más adelante se abordarán las posibles opciones.

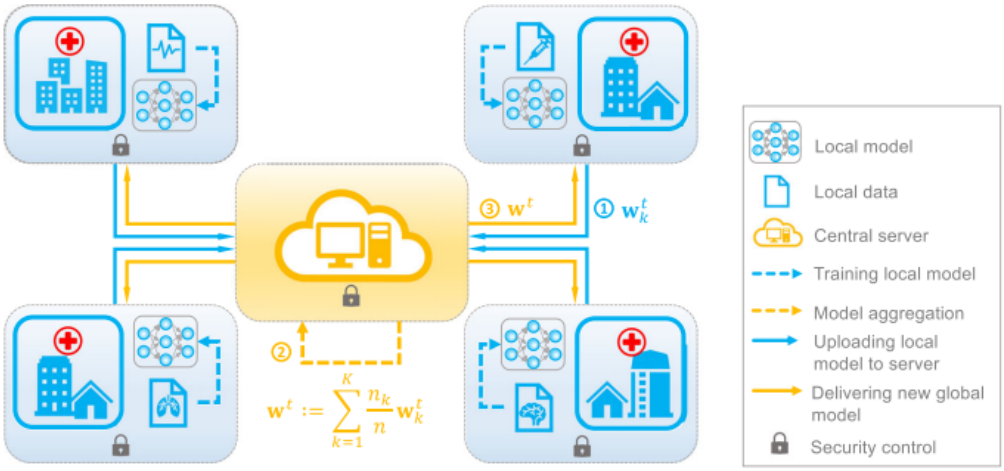


Figura 4.1: Arquitectura Básica del Aprendizaje Federado. Extraído de [41].

Algoritmo 7 Esquema General FL

Resultado: Modelo global entrenado
Inicializar: w_0
for cada ronda global $t = 0, 1, \ldots, T$ **do**
    for cada cliente $k \in \{1, \ldots, K\}$ **do**
        $w_{t+1}^k \leftarrow \text{ClientUpdate}(k, w_t)$
    end for
    $w_{t+1} \leftarrow \frac{1}{n} \sum_{k=1}^{K} n_k w_{t+1}^k$
end for
Salida Parámetros w_T

En donde K es la cantidad de clientes, n_k es el tamaño de la base de datos D_k del k-ésimo cliente, w_t son los parámetros globales del modelo en la ronda t, w_{t+1}^k son los parámetros locales del cliente k en la ronda t y $n = \sum_{k=1}^{K} n_k$.

El Aprendizaje Federado presenta una serie de ventajas:

- Privacidad: Los datos permanecen en los dispositivos locales.
- Reducción de Ancho de Banda: Solo se transfieren parámetros del modelo, no datos.
- Creación de muestras grandes: Cuando de manera local no se tienen suficientes datos para entrenar un modelo, se puede hacer en conjunto con otros clientes.

En contraste también presenta desventajas o debilidades como:

- Conflictos de comunicación: Requiere múltiples rondas de comunicación y por lo tanto, más cooperación por parte de los clientes.

- Seguridad: Riesgos de ataques como envenenamiento de parámetros (*model poisoning*) [47].

Incluso con esto, resulta ser una alternativa viable para ciertas tareas de aprendizaje como:

- Salud: Colaboración entre hospitales para mejorar diagnósticos sin compartir datos de pacientes.
- Tecnología Móvil: Personalización de modelos en dispositivos móviles (por ejemplo, teclados predictivos, sugerencias de contenido, etc.).

Pese a las ventajas y las múltiples aplicaciones que puede tener el FL, el hecho de tener un constante intercambio de parámetros lo hace vulnerable a que un adversario malicioso que tenga acceso a este intercambio de información pueda aprender algo sobre los datos de entrenamiento. Por esto, en los tres enfoques que seguiremos se usará la DP para perturbar las actualizaciones de parámetros y comparar la efectividad de este entrenamiento.

Para el desarrollo de este capítulo se utilizó una base de datos referente a deuda pública de la Ciudad de México (CDMX) tomada del Portal de datos abiertos de la Ciudad de México y puede consultarse es `https://datos.cdmx.gob.mx/dataset/deuda-de-la-ciudad` [12]. El objetivo en este capítulo es ajustar un modelo de regresión lineal para analizar los factores asociados a la deuda pública, utilizando un esquema de aprendizaje federado con privacidad diferencial usando los diferentes enfoques que se presentarán. Esto permitirá evaluar cómo diferentes características de la deuda, como el saldo del periodo, los días del contrato, los días restantes del contrato, las amortizaciones del periodo y los intereses del periodo influyen en el comportamiento de la deuda pública. Al mismo tiempo, se garantiza la privacidad de los datos involucrados. De esta manera, se logra un modelo predictivo efectivo que puede ser utilizado por múltiples entidades (acreedores) sin comprometer la confidencialidad de la información.

El proceso para poder utilizar los datos involucró las siguientes etapas:

1) Búsqueda y limpieza de datos

   La base de datos utilizada en este estudio contiene información sobre la deuda vigente de la Ciudad de México, desagregada por tipo de deuda, acreedor, fechas de inicio y fin de los créditos, así como los pagos de intereses y amortizaciones. La base de datos tiene registros del cuarto trimestre de 2018 al primer trimestre de 2024. Se realizó un proceso de limpieza para eliminar valores atípicos y registros incompletos, además de su normalización asegurando la coherencia y calidad de la información antes de su análisis.

2) Selección de variables

   A partir de la base de datos, se seleccionaron las siguientes variables relevantes para el modelo de regresión:

- Saldo periodo: saldo de la deuda en cada periodo,
- Días contrato: número de días totales del contrato de deuda,
- Días restantes contrato: número de días restantes para el vencimiento del contrato,
- Amortizaciones periodo: cantidad amortizada en cada periodo, e
- Intereses periodo: intereses pagados en cada periodo.

Estas variables fueron seleccionadas tras realizar un análisis de correlación que mostró su relevancia como se observa en la Tabla 4.1.

	saldo	**días**	**días res**	**amortizaciones**	**intereses**
saldo	1.000	0.628	0.675	-0.102	0.674
días	0.628	1.000	0.881	-0.184	0.456
días res	0.675	0.881	1.000	-0.185	0.481
amortizaciones	-0.102	-0.184	-0.185	1.000	0.148
intereses	0.674	0.456	0.481	0.148	1.000

Tabla 4.1: Correlaciones entre variables.

3) Particionamiento de datos

Para aplicar el aprendizaje federado, se particionó el conjunto de datos completo en subconjuntos por acreedor, simulando que cada acreedor dispone únicamente de su propio conjunto de datos. Este enfoque permite que las instituciones colaboren en el entrenamiento del modelo sin compartir sus datos sensibles directamente. Cada subconjunto de datos fue utilizado por un cliente en el esquema federado para ajustar localmente los parámetros del modelo, los cuales se enviaron posteriormente al servidor central para su agregación.

4.1. Enfoque por Descenso del Gradiente Estocástico

El Descenso por Gradiente Estocástico, es una variante del método por descenso del gradiente el cual es un método para optimizar una función [6].

La idea clave del descenso de gradiente estocástico es que el gradiente es un promedio, y podemos estimar ese promedio usando un pequeño subconjunto de muestras [14]. En cada paso del algoritmo, seleccionamos un lote de muestras de $\{x_1, \ldots, x_B\}$ al azar del conjunto de entrenamiento. El tamaño del lote B suele ser pequeño, lo importante es que el tamaño se mantiene constante, incluso si el conjunto de entrenamiento es muy grande. De este modo, podemos entrenar modelos con miles de millones de muestras utilizando actualizaciones basadas en solo unas pocas muestras a la vez.

El gradiente estimado se calcula como:

$$g = \frac{1}{B} \sum_{i=1}^{B} \nabla L(w, x_i),$$

usando las muestras del lote, donde L representa la función de costo a optimizar. Luego, el algoritmo de descenso de gradiente estocástico ajusta los parámetros siguiendo este gradiente:

$$w \leftarrow w - \eta g,$$

donde η representa la tasa de aprendizaje y se refiere a la cantidad en que se ajustan los parámetros de un modelo en cada iteración del algoritmo.

En el Algoritmo 8 podemos ver de mejor manera cómo se hace esta actualización, para una sola iteración y crear la función *Clientupdate* mencionada anteriormente.

Algoritmo 8 Clientupdate basada en SGD

Entrada: Parámetros globales w_t, número de cliente k, base de datos local D_k
Resultado: Parámetros locales del modelo entrenado w_{t+1}^k
for época $e = 1, \ldots, E$ **do**
    Muestrear al azar $\left\lfloor \frac{n}{B} \right\rfloor$ lotes de D_k
    for $j = 1, \ldots, \left\lfloor \frac{n}{B} \right\rfloor$ **do**
        for $i = 1, \ldots, B$ **do**
            Calcular gradiente:

$$g_j(x_i^j) \leftarrow \nabla L(w_t, x_i^j)$$

        end for
        Promedio del gradiente:

$$g_j \leftarrow \frac{1}{B} \sum_{i=1}^{B} g_j(x_i^j)$$

        Descenso:

$$w_{t+1}^k \leftarrow w_t - \eta g_j$$

    end for
end for
Salida: Parámetros optimizados w_{t+1}^k

η : tasa de aprendizaje, L : función a optimizar, B : tamaño de lote, E : número de épocas. Aquí, x_i^j se refiere al i-ésimo registro del j-ésimo lote.

Ahora, recordemos que nuestro objetivo es perturbar los parámetros de cada cliente para satisfacer la DP, en [3] se propone que cada cliente actualice sus parámetros de manera local siguiendo la idea del SDG pero haciendo una perturbación como puede verse en el Algoritmo 9.

Algoritmo 9 Clientupdate basada en SGD versión DP

Entrada: Parámetros globales w_t, número de cliente k, base de datos local D_k
Resultado: Parámetros locales del modelo entrenado w_{t+1}^k
for época $e = 1, \ldots, E$ **do**
    Muestrear al azar $\lfloor \frac{n}{B} \rfloor$ lotes de D_k
    for $j = 1, \ldots, \lfloor \frac{n}{B} \rfloor$ **do**
        for $i = 1, \ldots, B$ **do**
            Calcular gradiente:

$$g_j(x_i^j) \leftarrow \nabla L(w_t, x_i^j)$$

            Recortar gradiente:

$$\bar{g}_j(x_i^j) \leftarrow \frac{g_j(x_i^j)}{\text{máx}\left(1, \frac{\|g_j(x_i^j)\|_2}{C}\right)}$$

        end for
        Añadir ruido:

$$\tilde{g}_j \leftarrow \frac{1}{B}\left(\sum_{i=1}^{B} \bar{g}_j(x_i^j) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I})\right)$$

        Descenso:

$$w_{t+1}^k \leftarrow w_t - \eta \tilde{g}_j$$

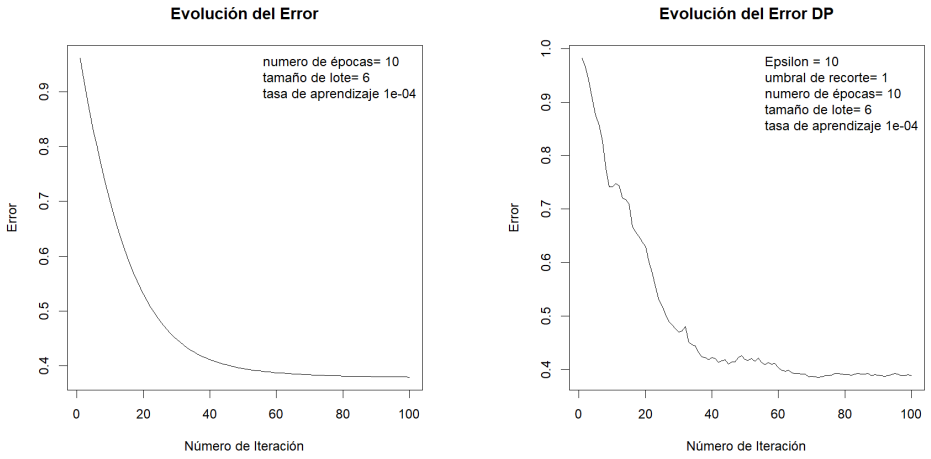
    end for
end for
Salida: Parámetros optimizados w_{t+1}

η : tasa de aprendizaje, L : función a optimizar, B : tamaño de lote, E : número de épocas, σ : escala de ruido (en este caso $\frac{1}{\epsilon}$), C : umbral de recorte. Aquí, x_i^j se refiere al i-ésimo registro del j-ésimo lote.

En esta versión se introduce el parámetro C llamado umbral de recorte, este factor funciona para limitar el tamaño de los gradientes y evitar que actualizaciones excesivamente grandes afecten negativamente el proceso de optimización. Esta técnica se conoce como *gradient clipping* [9].

En la Figura 4.2 observamos cómo se comporta el proceso de optimización entre el SGD normal y el que utiliza perturbaciones en las actualizaciones. Como se puede ver, ambos procesos convergen de manera similar, con la diferencia de que en la versión con DP se observan ciertas fluctuaciones a lo largo de las iteraciones, y la convergencia no es tan directa como en el método tradicional. Sin embargo, ambos métodos logran minimizar el error después de un número similar de iteraciones.

Para comparar cómo el valor de ϵ afecta la convergencia, en la Figura 4.3 se



(a) (b)

Figura 4.2: Error del Proceso de Optimización SDG: Optimización Normal (a), Optimización con DP (b).

grafica la evolución del error para diferentes valores de ϵ. En la gráfica se observa que, para valores pequeños de ϵ, el ruido introducido interfiere con la convergencia del proceso de optimización, e incluso, en algunos casos, en lugar de disminuir el error tras una iteración, este aumenta. Por otro lado, para valores grandes de ϵ, la convergencia es muy similar a la de un proceso sin ruido, aunque esto implica una menor privacidad.

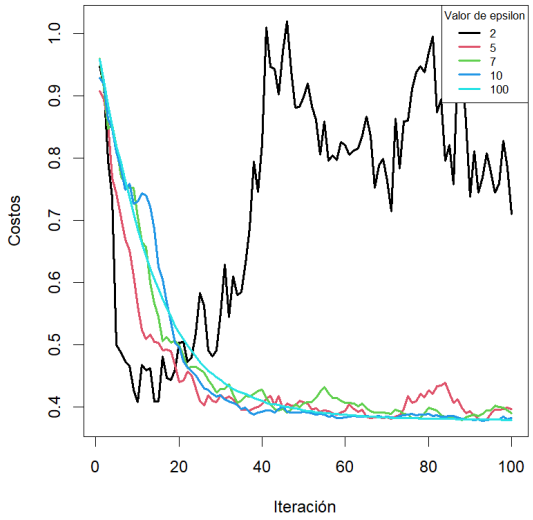


Figura 4.3: Error para Diferentes Valores de ϵ.

4.2. Enfoque por Aproximación del Punto Proximal

Otra variante del Aprendizaje federado es basada en el Algoritmo del Punto Proximal (PPA) [15]. Esta versión presenta una ventaja significativa y es que evita que los parámetros locales se alejen demasiado de los globales penalizando cuando esto ocurre. Para ello en la función ClientUpdate al momento de optimizar no sólo se busca reducir la función objetivo, sino además se busca reducir la distancia entre parámetros locales y globales. En [39] se propone una versión de esta técnica que queda planteada en el Algoritmo 10.

Algoritmo 10 Clientupdate basada en PPA

1: **Entrada:** Parámetros globales w_t, número de cliente k, base de datos local D_k
2: **Resultado:** Parámetros locales del modelo entrenado w_{t+1}^k
3: **Actualizar parámetros como:**

$$w_{t+1}^k = \arg\min_w \left(\sum_{x_i \in D_k} L(w, x_i) + \frac{\mu}{2} \|w - w_t\|^2 \right)$$

4: **Recortar:**

$$w_{t+1}^k \leftarrow \frac{w_{t+1}^k}{\max\left(1, \frac{\|w_{t+1}^k\|_2}{C}\right)}$$

5: **Salida:** Parámetros optimizados w_{t+1}^k

μ : coeficiente de penalización, L : función a optimizar, C : umbral de recorte.

Del mismo modo como lo hicimos en la sección anterior, buscamos perturbar las actualizaciones en cada iteración, esto se logra con la versión planteada en el Algoritmo 11.

Algoritmo 11 Clientupdate basada en PPA con DP

1: **Entrada:** Parámetros globales w_t, número de cliente k, base de datos local D_k
2: **Resultado:** Parámetros locales del modelo entrenado w_{t+1}^k
3: **Actualizar parámetros como:**

$$w_{t+1}^k = \arg\min_w \left(\sum_{x_i \in D_k} L(w, x_i) + \frac{\mu}{2} \|w - w_t\|^2 \right)$$

4: **Recortar:**

$$w_{t+1}^k \leftarrow \frac{w_{t+1}^k}{\max\left(1, \frac{\|w_{t+1}^k\|_2}{C}\right)}$$

5: **Añadir ruido y actualizar parámetros:**

$$w_{t+1}^k \leftarrow w_t + \mathcal{N}(0, \frac{2C}{m\epsilon})$$

6: **Salida:** Parámetros optimizados w_{t+1}^k

μ : coeficiente de penalización, L : función a optimizar, C : umbral de recorte, $m = \frac{n}{K}$

Ahora, debemos buscar la manera más eficiente para minimizar la función $\sum_{x_i \in D_k} L(w, x_i) + \frac{\mu}{2}\|w - w_t\|^2$ y así escoger a μ y w_{t+1}^k. Por ejemplo, si consideramos a $\sum_{x_i \in D_k} L(w, x_i)$ como el error cuadrático medio que ya hemos utilizado, podemos calcular el punto donde alcanza su mínimo como sigue:

Considerando

$$\begin{aligned} \sum_{x_i \in D_k} L(w, x_i) &= MSE(w) \\ &= \frac{1}{n_k} \sum_{i=1}^{n_k} (y_i - X_i w)^2 \\ &= \frac{1}{n_k} (y - Xw)^T (y - Xw). \end{aligned}$$

Tenemos que

$$\begin{aligned} \frac{\partial MSE(w)}{\partial w_i} &= \frac{2}{n}(y_i - X_i w)(-X_i) \\ &= \frac{2}{n} X_i (X_i w - y_i). \end{aligned}$$

Luego

$$\nabla MSE(w) = \tfrac{2}{n} X^T (Xw - y).$$

Por otro lado:

$$\begin{aligned}\frac{\partial}{\partial w_i}\frac{\mu}{2}\|w - w_{t-1}\|^2 &= \tfrac{\partial}{\partial w_i}\tfrac{\mu}{2}\textstyle\sum_{i=1}^{n}(w_i - w_{t-1,i})^2 \\ &= \tfrac{\mu}{2}2(w_i - w_{t-1,i}).\end{aligned}$$

Luego

$$\nabla\frac{\mu}{2}\|w - w_{t-1}\|^2 = \mu(w - w_{t-1}).$$

Así,

$$\nabla\left[\sum_{x_i\in D_k} L(w, x_i) + \frac{\mu}{2}\|w - w_{t-1}\|^2\right] = \frac{2}{n}X^T(Xw - y) + \mu(w - w_{t-1}).$$

Igualando a cero y resolviendo para w se tiene:

$$\begin{aligned}0 &= \frac{2}{n}X^T(Xw - y) + \mu(w - w_{t-1}) \\ 0 &= \frac{2}{n}X^TXw - \frac{2}{n}X^Ty + \mu w - \mu w_{t-1} \\ \frac{2}{n}X^Ty + \mu w_{t-1} &= \frac{2}{n}X^TXw + \mu w \\ \frac{2}{n}X^Ty + \mu w_{t-1} &= (\frac{2}{n}X^TX + \mu I^p)w \\ (\frac{2}{n}X^TX + \mu I^p)^{-1}(\frac{2}{n}X^Ty + \mu w_{t-1}) &= w.\end{aligned}$$

Con esto obtenemos el valor óptimo para w. Ahora, derivando respecto a μ:

$$\frac{\partial}{\partial\mu}\left[L_i(w) + \frac{\mu}{2}\|w - w_{t-1}\|^2\right] = \tfrac{1}{2}\|w - w_{t-1}\|^2.$$

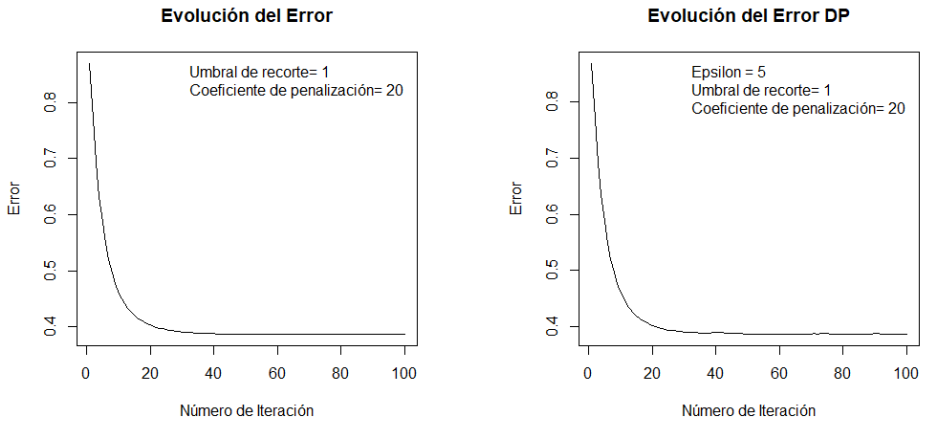
Lo anterior no nos indica cómo elegir a μ, así que lo ideal es utilizar validación cruzada (*cross validation*) [6] para elegir un valor ideal para μ, este proceso consiste en que el conjunto de datos se divide en k subconjuntos. Se entrena el modelo k veces, cada vez utilizando $k-1$ subconjuntos para entrenar y el subconjunto restante para validar. Al final, se promedia el rendimiento del modelo en las k iteraciones para obtener una estimación más fiable, esto se realizó para diferentes valores del coeficiente de penalización μ. En la Tabla 4.2 se presentan estos errores y de ahí se seleccionó el valor que presentó menor error.

En la Figura 4.4 observamos cómo se comporta el proceso de optimización entre el SGD tradicional y el que utiliza perturbaciones en las actualizaciones. Como se

Coeficiente de Penalización	Error Promedio Obtenido
0.001	0.453
0.01	0.442
0.1	0.415
1	0.401
2	0.397
5	0.393
10	0.387
20	0.398
100	0.448
200	0.607
500	0.749
1000	0.855
2000	0.855

Tabla 4.2: Error para Diferentes Valores de μ.

puede ver, ambos procesos convergen de manera similar, con la diferencia de que en la versión con DP se observan pequeñas fluctuaciones a lo largo de las iteraciones, y la convergencia no es tan directa como en el método tradicional. Sin embargo, ambos métodos logran minimizar el error después de un número similar de iteraciones. En comparación con la versión de SGD, donde las fluctuaciones son más evidentes, en este caso son menores debido a que la optimización es más eficiente.



(a) (b)

Figura 4.4: Error del Proceso de Optimización PPA: Optimización Normal (a), Optimización con DP (b).

Para comparar cómo el valor de ϵ afecta la convergencia, en la Figura 4.5 se grafica la evolución del error para diferentes valores de ϵ. En la gráfica se observa que,

para valores pequeños de ϵ, el ruido introducido interfiere más con la convergencia del proceso de optimización. Por otro lado, para valores grandes de ϵ, la convergencia es muy similar a la de un proceso sin ruido, aunque esto implica una menor privacidad. Un aspecto importante a destacar es que, al usar el modelo basado en PPA, podemos elegir valores de ϵ más pequeños e incluso así obtener buenos resultados. Esto demuestra que el modelo es más eficiente y menos sensible al ruido introducido por la privacidad diferencial.

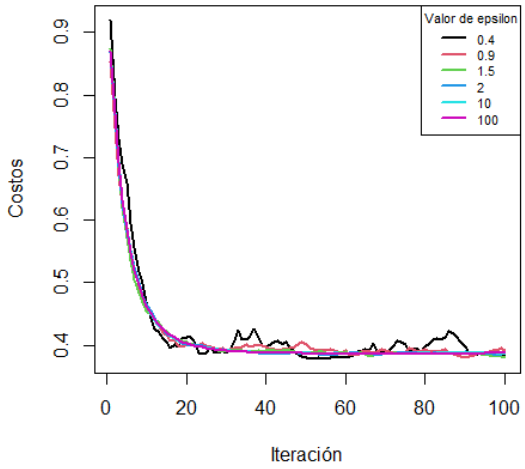


Figura 4.5: Error para Diferentes Valores de ϵ.

4.3. Enfoque usando Teoría de Grafos

En esta sección, usaremos un enfoque de la teoría de grafos, anteriormente se observó que con los otros enfoques se tenía un nodo en el que se concentraba toda la información, esto puede suponer un problema de comunicación para ese nodo en particular ya que ahí se concentra toda la información, más aún, cuando se tiene una cantidad excesiva de clientes esto también supone un problema. En [24] se propone el algoritmo descentralizado el cual busca evitar la concentración de información en un nodo. Antes de estudiar el algoritmo descentralizado se presentan una serie de definiciones de teoría de grafos necesarias para esta sección.

Definición 4.1. *Un grafo G es un par ordenado $G = (V, E)$, tal que V es un conjunto no vacío y E está conformado por subconjuntos de tamaño 2 de elementos de V. A los elementos de V les llamamos vértices o nodos y a los elementos de E les llamamos ejes o aristas.*

Usualmente para dibujar un grafo usamos un punto para representar un vértice y una línea entre un par de nodos si estos dos forman un eje.

Ejemplo 4.1. *Sea $G = (V, E)$ donde $V = \{1, 2, 3, 4, 5\}$ y*
$E = \{\{1, 2\}, \{1, 3\}, \{2, 3\}, \{3, 4\}, \{4, 5\}, \{5, 1\}\}$
Podemos representar este grafo en la Figura 4.6.

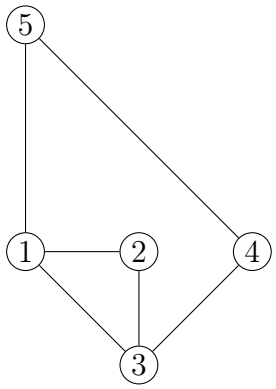


Figura 4.6: Grafo con 5 Vértices y Aristas.

Definición 4.2. *Decimos que dos vértices $v_1, v_2 \in V$ son adyacentes o vecinos si $\{v_1, v_2\} \in E$.*

Definición 4.3. *Dado un vértice $v \in V$, definimos su grado como el número de vecinos de v y lo denotamos como $grad(v)$, esto es*

$$grad(v) = |\{w \in V : \{v, w\} \in E\}|.$$

Definición 4.4. *Dos vértices $v_1, v_2 \in V$ están conectados por un camino si existe una cantidad finita de vértices $w_1, w_2, \ldots, w_k \in V$ tal que*

$$\{\{v_1, w_1\}, \{w_1, w_2\}, \{w_2, w_3\}, \ldots, \{w_{n-1}, w_n\}, \{w_n, v_2\}\} \subseteq E.$$

Definición 4.5. *Diremos que un grafo G es conexo si cualesquiera dos vértices están conectados por un camino.*

Observación 4.1. *En adelante todos los grafos que consideremos serán conexos.*

Con esta teoría básica podemos entender cómo funciona el algoritmo descentralizado. Primero notemos que los métodos mencionados anteriormente obedecen a una estructura como la que puede verse en la Figura 4.7 y ésta representa la comunicación entre clientes.

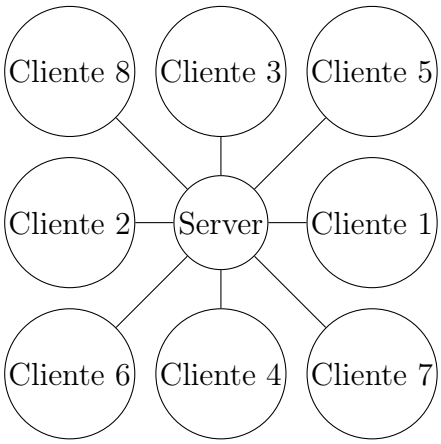


Figura 4.7: Estructura centralizada.

Sin embargo, en un problema concreto esta estructura no necesariamente puede verse de este modo, esto debido a falta de canales de comunicación, así como de la falta de confianza para concentrar toda la información en un sólo lugar. Luego, podría esperarse una estructura como la presentada en la Figura 4.8 en la cual ya no existe una figura central que concentre toda la información y en su lugar cada cliente se comunica con unos pocos.

Ahora bien, esta estructura indica que cada cliente entrenará el modelo de aprendizaje de manera local para después compartir sus actualizaciones únicamente con sus vecinos, después cada cliente actualiza sus parámetros obtenidos con la información recibida de sus vecinos y se repite este ciclo una cantidad suficiente de veces. Esta idea queda concentrada en el Algoritmo 12 en la cual se representa lo que cada cliente hace de manera local.

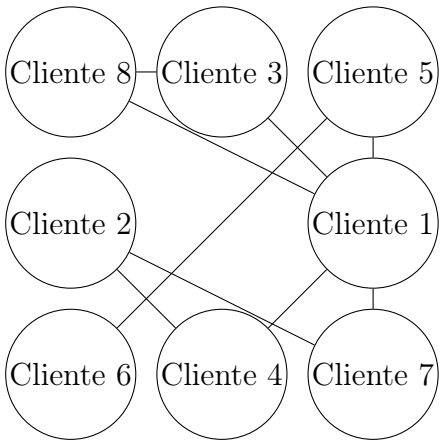


Figura 4.8: Estructura Descentralizada.

Algoritmo 12 Aprendizaje Federado usando Teoría de Grafos [4]

Entrada: Parámetros locales de sus vecinos w_t^j en cada $t = 1, \ldots, T$, base de datos local D_k
Resultado: Parámetros locales del modelo entrenado w_{t+1}^k
for $t = 1, \ldots, T$ **do**
    Muestrear al azar un lote de tamaño E de D_k
    for $l = 1, \ldots, E$ **do**
        Calcular gradiente:

$$g(x_l) \leftarrow \nabla L(w_t, x_l)$$

    end for
    Promedio del gradiente:

$$g \leftarrow \frac{1}{B} \sum_{l=1}^{B} g(x_l)$$

    Actualizar el modelo:

$$w_{t+\frac{1}{2}}^i \leftarrow w_t - \eta g$$

    Promedio ponderado de los parámetros:

$$w_{t+1}^i \leftarrow \sum_{j=1}^{K} W_{ij} w_{t+\frac{1}{2}}^j$$

end for
Salida: Parámetros optimizados w_{t+1}^k
tasa de aprendizaje η, tamaño del lote m, matriz de pesos W, L : función a optimizar.

En el Algoritmo 12 existen varios parámetros a seleccionar, algunos como la tasa de aprendizaje η, tamaño de los lotes E y el número de iteraciones T ya se han revisado en los métodos anteriores. Sin embargo, no es el caso para la mezcla de pesos W, esta es una matriz $W \in \mathbb{M}_{N\times N}(\mathbb{R}^+)$ la cual en cada entrada $W_{i,j}$ representa la influencia del nodo j sobre el nodo i.

Condición 4.1. *La matriz W cumple con las siguientes condiciones:*

a) W es no negativa, es decir, $W_{i,j} \geq 0$,

b) W es estocástica, esto es, $\sum_{j=1}^{N} W_{i,j} = 1$ para todo $i = 1, \ldots, N$ y

c) W es simétrica, i.e., $W = W^T$.

Observación 4.2. *Además, note que como W es estocástica y simétrica también es doblemente estocástica, esto es, al sumar por columnas también se obtiene* 1 *como resultado.*

La idea natural para obtener la matriz W es a partir del grafo que representa la estructura de comunicación, pero esta debe cumplir con las condiciones mencionadas en la Condición 4.1. En [40] se presentan dos propuestas para obtener una matriz W a partir de un grafo y que cumpla con tales condiciones para ello considere las siguientes definiciones.

Definición 4.6. *Dado un grafo $G = (V, E)$ con conjunto de vértices $V = \{1, 2, \ldots, n\}$ se definen las siguientes matrices asociadas al grafo:*

a) Matriz diagonal de grados D tal que:

$$D_{i,j} = \begin{cases} grad(i) & si\ i = j, \\ 0 & si\ i \neq j. \end{cases}$$

b) Matriz de adyacencia A tal que:

$$A_{i,j} = \begin{cases} 1 & si\ \{i, j\} \in E, \\ 0 & si\ \{i, j\} \notin E. \end{cases}$$

Con la definición de las matrices anteriores podemos construir la matriz Laplaciana de la siguiente forma

Definición 4.7. *Dado un grafo $G = (V, E)$ se define la matriz Laplaciana como $L = D - A$, es decir,*

$$L_{i,j} = \begin{cases} -1 & si\ \{i, j\} \in E, \\ grad(i) & si\ i = j, \\ 0 & en\ otro\ caso. \end{cases}$$

Notación 4.1. *Denotemos por I_n a la matriz identidad de orden n. Dado un grafo $G = (V, E)$ denotamos*

$$\Delta(G) = \max_{k \in V} grad(k).$$

Para construir una matriz W que cumpla la Condición 4.1 en [40] se presenta el siguiente resultado.

Teorema 4.1. *Sea un grafo $G = (V, E)$ con conjunto de vértices $V = \{1, 2, \ldots, n\}$, considere L su matriz Laplaciana, sea $0 < \alpha \leq \frac{1}{\Delta(G)}$, entonces la matriz $W = I_n - \alpha L$ cumple con la Condición 4.1.*

Demostración. Observe que como

$$L_{i,j} = \begin{cases} -1 & \text{si } \{i,j\} \in E, \\ grad(i) & \text{si } i = j, \\ 0 & \text{en otro caso,} \end{cases}$$

entonces cada entrada de $W = I_n - \alpha L$ puede verse como:

$$W_{i,j} = \begin{cases} \alpha & \text{si } \{i,j\} \in E, \\ 1 - \alpha \cdot grad(i) & \text{si } i = j, \\ 0 & \text{en otro caso.} \end{cases} \tag{4.1}$$

Ahora, note que como $0 < \alpha \leq \frac{1}{d_{max}}$, si $\{i,j\} \in E$, entonces $W_{i,j} = \alpha > 0$, por otro lado, si $i = j$, entonces $W_{i,j} = 1 - \alpha \cdot grad(i)$, pero como $\alpha \leq \frac{1}{d_{max}}$, se tiene que $W_{i,j} = 1 - \alpha \cdot grad(i) \geq 0$. Así, en cualquier caso $W_{i,j} \geq 0$, así W es no negativa. También se puede ver que por la construcción de W efectivamente es simétrica pues $W_{i,j} = W_{j,i}$ para todo $i, j = 1, \ldots, N$. Por lo que sólo faltaría verificar que sea estocástica, para ello, sea $k \in V$ fijo, entonces k tiene $grad(k)$ vecinos así que existen $j_1, \ldots, j_{grad(k)} \in V$ tal que $\{k, j_1\}, \ldots, \{k, j_{grad(k)}\} \in E$. Luego

$$\begin{aligned} \sum_{j=1}^{N} W_{k,j} &= W_{k,1} + W_{k,2} + \cdots + W_{k,N} \\ &= W_{k,j_1} + W_{k,j_2} + \cdots + W_{k,j_{grad(k)}} + W_{k,k} \\ &= \underbrace{\alpha + \cdots + \alpha}_{grad(k)\text{-veces}} + (1 - \alpha \cdot grad(k)) \\ &= \alpha \cdot grad(k) + (1 - \alpha \cdot grad(k)) \\ &= 1. \end{aligned}$$

En la segunda identidad se usó (4.1). Por lo tanto, W satisface la Condición 4.1.

□

Con esto, el Teorema 4.1 nos da varias alternativas para generar a la matriz W según escojamos al parámetro α el cual recibe el nombre de tasa de difusión ya que controla la velocidad con la que la información se propaga en el grafo.

Ejemplo 4.2. *Consideremos el siguiente grafo:*

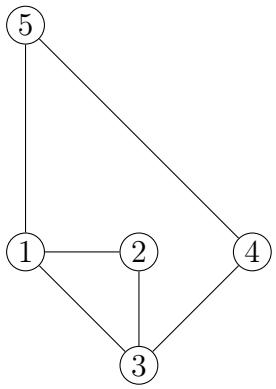


Figura 4.9: Grafo con 5 Vértices y Aristas.

Calculamos la matriz de pesos W usando la fórmula:

$$W_{i,j} = \begin{cases} \alpha & si\ \{i,j\} \in E, \\ 1 - \alpha \cdot grad(i) & si\ i = j, \\ 0 & en\ otro\ caso, \end{cases}$$

en donde $0 < \alpha \leq \frac{1}{d_{max}}$. Primero, calculamos los grados de los nodos:

- $grad(1) = 3$ *(conectado a 2, 3, 5)*
- $grad(2) = 2$ *(conectado a 1, 3)*
- $grad(3) = 3$ *(conectado a 1, 2, 4)*
- $grad(4) = 2$ *(conectado a 3, 5)*
- $grad(5) = 2$ *(conectado a 4, 1)*

Observe que $d_{max} = 3$. Luego, calculamos los pesos W_{ij} para cada par de vértices vecinos $(i,j) \in E$,

$$W_{12} = W_{13} = W_{15} = W_{23} = W_{34} = W_{45} = W_{51} = \alpha.$$

Finalmente, calculamos los valores en la diagonal de la matriz de pesos,

$$
\begin{aligned}
W_{11} &= 1 - \alpha \cdot grad(1) = 1 - 3\alpha, \\
W_{22} &= 1 - \alpha \cdot grad(1) = 1 - 2\alpha, \\
W_{33} &= 1 - \alpha \cdot grad(1) = 1 - 3\alpha,. \\
W_{44} &= 1 - \alpha \cdot grad(1) = 1 - 2\alpha, \\
W_{55} &= 1 - \alpha \cdot grad(1) = 1 - 2\alpha.
\end{aligned}
$$

La matriz de pesos W es

$$
W = \begin{bmatrix}
1-3\alpha & \alpha & \alpha & 0 & \alpha \\
\alpha & 1-2\alpha & \alpha & 0 & 0 \\
\alpha & \alpha & 1-3\alpha & \alpha & 0 \\
0 & 0 & \alpha & 1-2\alpha & \alpha \\
\alpha & 0 & 0 & \alpha & 1-2\alpha
\end{bmatrix}.
$$

En particular, si tomamos $\alpha = \frac{1}{6} \in (0, \frac{1}{\Delta(G)}]$ tenemos

$$
W = \begin{bmatrix}
\frac{3}{6} & \frac{1}{6} & \frac{1}{6} & 0 & \frac{1}{6} \\
\frac{1}{6} & \frac{4}{6} & \frac{1}{6} & 0 & 0 \\
\frac{1}{6} & \frac{1}{6} & \frac{3}{6} & \frac{1}{6} & 0 \\
0 & 0 & \frac{1}{6} & \frac{4}{6} & \frac{1}{6} \\
\frac{1}{6} & 0 & 0 & \frac{1}{6} & \frac{4}{6}
\end{bmatrix}.
$$

Otro camino diferente para generar una matriz W que cumpla con la Condición 4.1 queda enunciado en el siguiente teorema.

Teorema 4.2. *Sea un grafo $G = (V, E)$ con conjunto de vértices $V = \{1, 2, \ldots, n\}$ entonces la matriz W definida como*

$$
W_{i,j} = \begin{cases}
\frac{1}{\max\{grad(i), grad(j)\}+1} & si\ \{i,j\} \in E, \\
1 - \sum_{j \neq i} W_{i,j} & si\ i = j, \\
0 & en\ otro\ caso,
\end{cases}
$$

cumple con la Condición 4.1.

Demostración. Primero veamos que la matriz es no negativa, para $i \neq j$ es claro que por construcción $W_{i,j} \geq 0$, así que sólo debemos verificar que $W_{i,j} \geq 0$ para cuando $i = j$. Sea $i \in V$ fijo, note que $\frac{1}{\max\{grad(i), grad(j)\}+1} \leq \frac{1}{grad(i)+1}$ para cualquier $j \in V$.

Luego

$$\begin{aligned} W_{i,i} &= 1 - \sum_{j \neq i} W_{i,j} \\ &\geq 1 - \sum_{j \neq i} \frac{1}{grad(i)+1} \\ &= 1 - grad(i)\frac{1}{grad(i)+1} \\ &= \frac{1}{grad(i)+1} \\ &\geq 0. \end{aligned}$$

Así, W es no negativa. También se puede ver fácilmente que $W_{i,j} = W_{j,i}$ por lo que es simétrica, y trivialmente se cumple que es estocástica dada la forma en que está definida. Por lo tanto, W cumple con la Condición 4.1. □

Ejemplo 4.3. *Consideremos el siguiente grafo:*

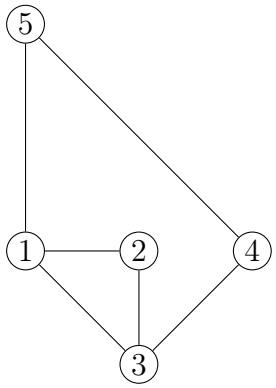


Figura 4.10: Grafo con 5 Vértices y Aristas.

Calculamos la matriz de pesos W usando la fórmula:

$$W_{ij} = \begin{cases} \frac{1}{\max(grad(i),grad(j))+1} & si\ i \neq j\ y\ \{i,j\} \in E, \\ 1 - \sum_{j \neq i} W_{ij} & si\ i = j, \\ 0 & de\ otro\ modo. \end{cases}$$

Primero, usando lo obtenido en el Ejemplo 4.2 tenemos que: $grad(1) = grad(3) = 3$, $grad(2) = grad(4) = grad(5) = 2$.

Luego, calculamos los pesos W_{ij} para cada par de vértices vecinos $(i,j) \in E$:

$$W_{12} = \frac{1}{\text{máx}(3,2)+1} = \frac{1}{4} = 0.25,$$
$$W_{13} = \frac{1}{\text{máx}(3,3)+1} = \frac{1}{4} = 0.25,$$
$$W_{15} = \frac{1}{\text{máx}(3,2)+1} = \frac{1}{4} = 0.25,$$
$$W_{23} = \frac{1}{\text{máx}(2,3)+1} = \frac{1}{4} = 0.25,$$
$$W_{34} = \frac{1}{\text{máx}(3,2)+1} = \frac{1}{4} = 0.25,$$
$$W_{45} = \frac{1}{\text{máx}(2,2)+1} = \frac{1}{3} \approx 0.333,$$
$$W_{51} = \frac{1}{\text{máx}(2,3)+1} = \frac{1}{4} = 0.25.$$

Finalmente, calculamos los valores en la diagonal de la matriz de pesos:

$$W_{11} = 1 - (W_{12} + W_{13} + W_{15}) = 1 - (0.25 + 0.25 + 0.25) = 1 - 0.75 = 0.25,$$
$$W_{22} = 1 - (W_{21} + W_{23}) = 1 - (0.25 + 0.25) = 1 - 0.50 = 0.50,$$
$$W_{33} = 1 - (W_{31} + W_{32} + W_{34}) = 1 - (0.25 + 0.25 + 0.25) = 1 - 0.75 = 0.25,$$
$$W_{44} = 1 - (W_{43} + W_{45}) = 1 - (0.25 + 0.333) = 1 - 0.583 = 0.417,$$
$$W_{55} = 1 - (W_{51} + W_{54}) = 1 - (0.25 + 0.333) = 1 - 0.583 = 0.417.$$

La matriz de pesos W es

$$W = \begin{bmatrix} 0.25 & 0.25 & 0.25 & 0 & 0.25 \\ 0.25 & 0.50 & 0.25 & 0 & 0 \\ 0.25 & 0.25 & 0.25 & 0.25 & 0 \\ 0 & 0 & 0.25 & 0.417 & 0.333 \\ 0.25 & 0 & 0 & 0.333 & 0.417 \end{bmatrix}.$$

Observación 4.3. *A la matriz del Teorema 4.1 se le llama de pesos de ejes constantes, ya que para dos nodos vecinos i, j, el cálculo de $W_{i,j}$ no involucra un cálculo entre los grados de los vértices i, j. En cambio, la matriz del Teorema 4.2 se llama de pesos de ejes basados en el grado local ya que para dos nodos vecinos i, j, el cálculo de $W_{i,j}$ involucra una operación entre los grados de los nodos i, j.*

Ahora, para aplicar DP al Algoritmo 12, del mismo modo como se ha hecho anteriormente, se perturbarán las actualizaciones que se comparten, esto se puede ver en el Algoritmo 13.

Algoritmo 13 Aprendizaje Federado con Teoría de Grafos con DP [4]

Entrada: Parámetros locales de sus vecinos w_t^j en cada $t = 1, \ldots, T$, base de datos local D_k
Resultado: Parámetros locales del modelo entrenado w_{t+1}^k
for $t = 1, \ldots, T$ **do**
    Muestrear al azar un lote de tamaño B de D_k
    for $l = 1, \ldots, B$ **do**
        Calcular gradiente:

$$g(x_l) \leftarrow \nabla L(w_t, x_l)$$

        Recortar gradiente:

$$\bar{g}(x_l) \leftarrow \frac{g(x_l)}{\text{máx}\left(1, \frac{\|g(x_l)\|_2}{C}\right)}$$

    end for
    Añadir ruido:

$$\tilde{g} \leftarrow \frac{1}{B}\left(\sum_{i=1}^{B} \bar{g}(x_l) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I})\right)$$

    Actualizar el modelo:

$$w_{t+\frac{1}{2}}^i \leftarrow w_t - \eta\tilde{g}$$

    Promedio ponderado de los parámetros:

$$w_{t+1}^i \leftarrow \sum_{j=1}^{K} W_{ij} w_{t+\frac{1}{2}}^j$$

end for
Salida: Parámetros optimizados w_{t+1}^k
tasa de aprendizaje η, tamaño del lote m, matriz de pesos W, L : función a optimizar, σ : escala de ruido (en este caso $\frac{1}{\epsilon}$), C : umbral de recorte.

Ahora, para nuestra aplicación, vamos a suponer que tenemos una estructura de grafo de 7 nodos como en la Figura 4.11.

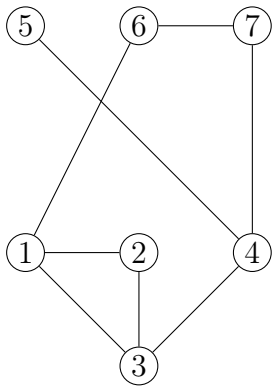


Figura 4.11: Grafo con 7 Vértices y Aristas.

Para tal grafo, su matriz de pesos según el Teorema 4.2 es:

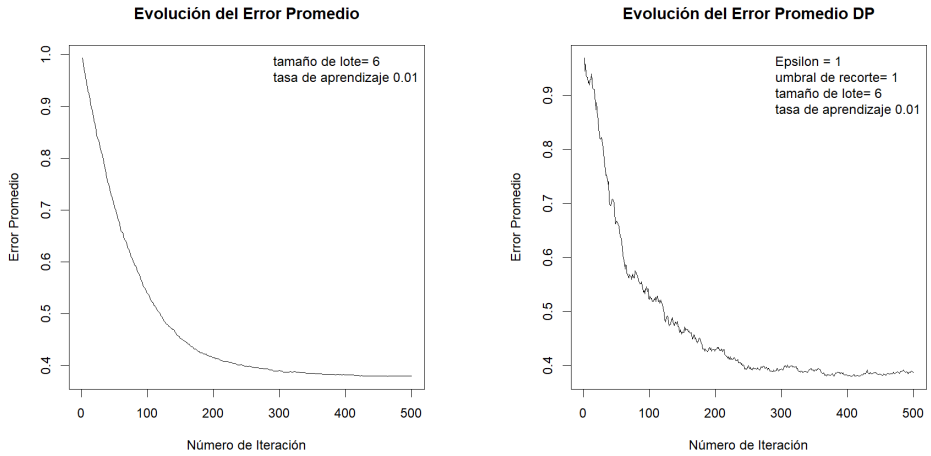
$$W = \begin{pmatrix} \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & 0 & 0 & \frac{1}{4} & 0 \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} & 0 & 0 & 0 & 0 \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & 0 & \frac{1}{4} \\ 0 & 0 & 0 & \frac{1}{4} & \frac{3}{4} & 0 & 0 \\ \frac{1}{4} & 0 & 0 & 0 & 0 & \frac{5}{12} & \frac{1}{3} \\ 0 & 0 & 0 & \frac{1}{4} & 0 & \frac{1}{3} & \frac{5}{12} \end{pmatrix}.$$

En la Figura 4.12 observamos cómo se comporta el proceso de optimización entre el proceso de grafos tradicional y el que utiliza perturbaciones en las actualizaciones. Como se puede ver, ambos procesos convergen de manera muy similar, a pesar del ruido introducido en el segundo caso. Aunque se presentan fluctuaciones a lo largo del proceso, estas no son lo suficientemente significativas como para afectar la convergencia del modelo.

Para comparar cómo el valor de ϵ afecta la convergencia, en la Figura 4.13 se grafica la evolución del error para diferentes valores de ϵ. Podemos notar que, para valores más pequeños de ϵ, las fluctuaciones provocadas por el ruido son mayores, pero incluso así el modelo logra reducir el error a lo largo del tiempo, lo cual es beneficioso para preservar la privacidad.

Ahora, considerando la misma estructura del grafo de la Figura 4.11, construyamos su matriz de pesos ahora según el Teorema 4.1, esta es:

$$W = \begin{pmatrix} 1-3\alpha & \alpha & \alpha & 0 & 0 & \alpha & 0 \\ \alpha & 1-2\alpha & \alpha & 0 & 0 & 0 & 0 \\ \alpha & \alpha & 1-3\alpha & \alpha & 0 & 0 & 0 \\ 0 & 0 & \alpha & 1-3\alpha & \alpha & 0 & \alpha \\ 0 & 0 & 0 & \alpha & 1-\alpha & 0 & 0 \\ \alpha & 0 & 0 & 0 & 0 & 1-2\alpha & \alpha \\ 0 & 0 & 0 & \alpha & 0 & \alpha & 1-2\alpha \end{pmatrix} \quad \text{con } 0 < \alpha \leq \frac{1}{3}.$$



(a) (b)

Figura 4.12: Error del Proceso de Optimización Grafos: Optimización Normal (a), Optimización con DP (b).

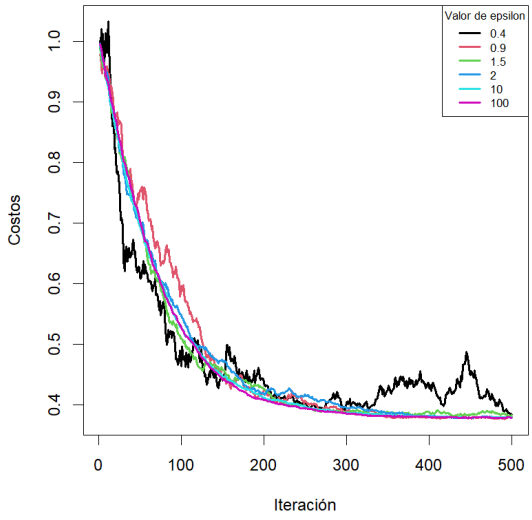


Figura 4.13: Error para Diferentes Valores de ϵ.

Para seleccionar un valor adecuado para la tasa de difusión α usamos validación cruzada como se ha hecho anteriormente. En la Tabla 4.3 se presentan los errores generados por cada valor y de ahí se seleccionó el valor que presentó menor error. Así, seleccionando $\alpha = \frac{1}{3}$ tenemos la matriz:

Tasa de difusión	**Error_Promedio_Obtenido**
0.01	0.408
0.05	0.384
0.1	0.381
0.15	0.380
0.18	0.380
0.2	0.379
0.22	0.380
0.25	0.380
0.27	0.379
0.3	0.379
0.33	0.379

Tabla 4.3: Error para Diferentes Valores de α.

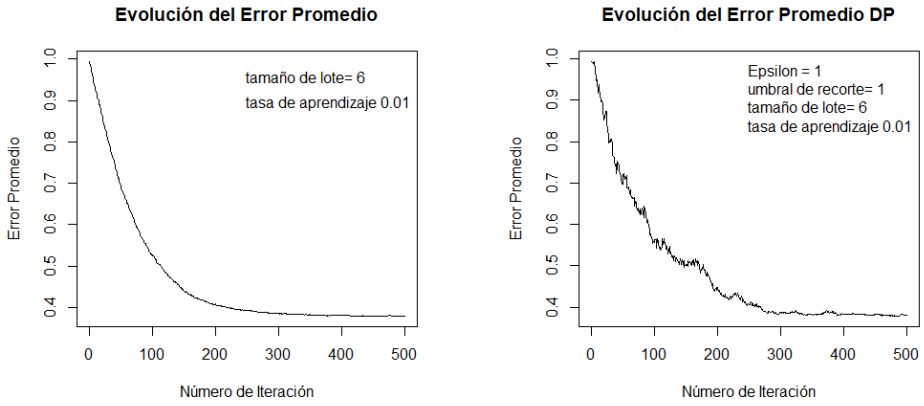
$$W = \begin{pmatrix} 0 & \frac{1}{3} & \frac{1}{3} & 0 & 0 & \frac{1}{3} & 0 \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{3} & 0 & \frac{1}{3} & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{3} & 0 & \frac{1}{3} & 0 & \frac{1}{3} \\ 0 & 0 & 0 & \frac{1}{3} & \frac{2}{3} & 0 & 0 \\ \frac{1}{3} & 0 & 0 & 0 & 0 & \frac{1}{3} & \frac{1}{3} \\ 0 & 0 & 0 & \frac{1}{3} & 0 & \frac{1}{3} & \frac{1}{3} \end{pmatrix}.$$

En la Figura 4.14 observamos cómo se comporta el proceso de optimización entre el proceso de grafos tradicional y el que utiliza perturbaciones en las actualizaciones. Como se puede ver, ambos procesos convergen de manera muy similar. A diferencia de lo que ocurre en la Figura 4.12, al considerar esta matriz, las fluctuaciones son más evidentes al inicio del entrenamiento, pero luego el comportamiento se asemeja mucho al de la versión sin perturbaciones.

Para comparar cómo el valor de ϵ afecta la convergencia, en la Figura 4.15 se grafica la evolución del error para diferentes valores de ϵ. Algo importante a destacar es que, en ciertos casos, el proceso de aprendizaje es más rápido en las primeras iteraciones cuando ϵ es pequeño. Esto significa que introducir ruido, en ocasiones, puede beneficiar la optimización, ya que el ruido se reduce más rápidamente en ciertos momentos.

Como hemos visto, ambas opciones de las matrices consideradas en los Teoremas 4.2 y 4.1 producen buenos resultados y ambas resultan viables de aplicar, por esta razón la matriz del Teorema 4.2 resulta más fácil de aplicar ya que involucra menos cálculos y evita la elección del parámetro de difusión α.

A manera de conclusión de este capítulo observamos que las tres alternativas estudiadas resultaron bastante efectivas a la hora de optimizar el proceso de aprendizaje del modelo, pero se resalta la relevancia de la última sección puesto que como se mencionó anteriormente, tener una estructura descentralizada contribuye más a



(a) (b)

Figura 4.14: Error del Proceso de Optimización Grafos con Matriz de Pesos Constante: Optimización Normal (a), Optimización con DP (b).

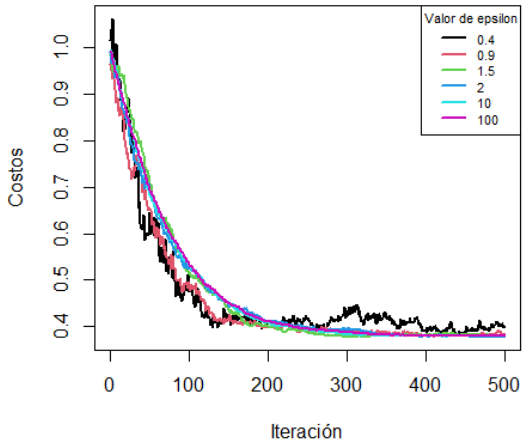


Figura 4.15: Error para Diferentes Valores de ϵ.

una mejor protección de privacidad, además de que puede obedecer a una estructura de comunicación del mundo real en la que no todos los clientes tienen la capacidad de comunicarse con todos a la vez, más aún, al no haber tantos canales de comunicación resulta ser un poco menos costoso este método en términos computacionales, es por este conjunto de razones que este método puede ser aplicable a gran escala resultado en un modelo bastante efectivo.

Conclusiones

A lo largo de este trabajo se han presentado diversos resultados de la aplicación de la Privacidad Diferencial a diferentes contextos, desde la publicación de estadísticas hasta el entrenamiento de modelos de aprendizaje y más aún, el entrenamiento de modelos en entornos no centralizados. La Privacidad Diferencial ha representado una alternativa muy poderosa cuando se quiere hacer análisis y/o ciencia de datos, pues como se ha mencionado, cosas como publicar estadísticas o modelos entrenados de aprendizaje automático, representan una posible vulnerabilidad para los datos y una oportunidad para un agente malicioso externo que quisiera hacer inferencia o aprender algo sobre los datos. Hay que notar que teoría de la Privacidad Diferencial se conecta fuertemente con diversas ramas de las matemáticas, especialmente con el análisis y la teoría de la probabilidad. Su desarrollo y aplicación requieren una comprensión profunda de estos campos, lo que hace que la teoría sea bastante compleja. Para lograr una comprensión adecuada y efectiva, es necesario tener en cuenta no solo los fundamentos matemáticos, sino también las técnicas avanzadas que se emplean para garantizar un equilibrio entre la privacidad de los datos y la precisión de los modelos. Este dominio requiere una integración de conocimientos teóricos y prácticos, y un enfoque detallado sobre cómo los parámetros, como el valor de ϵ, afectan el comportamiento y rendimiento de los modelos de aprendizaje.

Sobre este trabajo, se resalta la importancia y relevancia de lo desarrollado en el Capítulo 4 pues el Aprendizaje Federado resulta ser una muy buena alternativa para el entrenamiento de modelos de una forma no centralizada, pues la descentralización de datos representa una serie de ventajas como privacidad y seguridad, además permite que el modelo sea más resistente a algún fallo, pues si una parte falla, otras partes pueden seguir funcionando. Además de esto, se tiene una reducción de la censura y control, pues en este modelo ningún actor tiene control total sobre los datos, lo que permite más transparencia. Otra de las principales ventajas es que puede ser más sencillo hacerlo a gran escala, pues se pueden optimizar los recursos distribuyendo el procesamiento y almacenamiento. Se pueden realizar cálculos en varios dispositivos de forma paralela, lo que reduce la necesidad de transmitir grandes volúmenes de datos a una ubicación central.

Así, señalamos la importancia del Aprendizaje Federado complementado con Privacidad Diferencial para otorgar un modelo que provea de privacidad y sea eficiente. Pues más allá de la aplicación específica en el análisis de deuda pública, las contribuciones significativas de este trabajo radican en la potencialidad de colaboración

interinstitucional en múltiples sectores. La metodología utilizada implica que organizaciones diversas pueden colaborar para aprender de los datos sin comprometerlos a una posible violación de privacidad. Esto es crucial en campos como la salud, las finanzas o la investigación científica, donde los datos sensibles y privados son valiosos para generar modelos predictivos. Así, lo presentado en este trabajo no solo ha probado la viabilidad del aprendizaje federado en un contexto financiero, sino que también brinda la confianza para futuros proyectos colaborativos que prioricen tanto la privacidad como la precisión en el análisis de datos.

De manera general, se resalta la importancia de aplicar la privacidad diferencial en tareas o proyectos de análisis y ciencia de datos siempre que sea posible, ya que es fundamental para proteger la privacidad de los datos sensibles. Si bien es cierto que introducir privacidad diferencial puede afectar la precisión de los modelos en mayor o menor medida, este sacrificio resulta necesario para asegurar que los datos de los usuarios no se vean comprometidos. Este enfoque no solo es esencial para garantizar la protección de datos sensibles, sino que también contribuye a generar confianza y transparencia en el manejo de la información personal. Además, es una práctica recomendada en un mundo cada vez más enfocado en la protección de datos personales.

Trabajo Futuro: A modo de trabajo futuro por desarrollar se podría explorar las siguientes opciones:

- El uso de Privacidad Diferencial en otros modelos estadísticos o predictivos, tales como los basados en series de tiempo.
- Además, se quisiera explorar otro tipo de opciones para el Aprendizaje Federado, por ejemplo, desde el enfoque de teoría de grafos considerar diferentes tipos de configuraciones o explorar diferentes formas de generar las matrices de pesos.
- Otra alternativa está en explorar otros modelos emergentes de analítica de datos como lo es el análisis topológico de datos y considerar la forma de aplicar Privacidad Diferencial.

Se espera estudiar estas opciones en algún momento futuro.

Bibliografía

[1] Abadi, M., et al. (2016). Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security* (pp. 308-318).

[2] Alpaydin, E. (2020). *Introduction to machine learning*. Cambridge, MA: MIT Press.

[3] Banse, A., Kreischer, J., & *et al.* (2024). Federated learning with differential privacy. *arXiv preprint arXiv:2402.02230*.

[4] Bellet, A., Kermarrec, A.-M., & Lavoie, E. (2022). D-cliques: Compensating for data heterogeneity with topology in decentralized federated learning. In *2022 41st International Symposium on Reliable Distributed Systems (SRDS)* (pp. 1–11). IEEE.

[5] Bezdek, J. C. (2013). *Pattern recognition with fuzzy objective function algorithms*. Springer Science & Business Media.

[6] Bishop, C. M., & Nasrabadi, N. M. (2006). *Pattern recognition and machine learning* (Vol. 4, No. 4). New York: Springer.

[7] Böhler, J., & Kerschbaum, F. (2020). Secure multi-party computation of differentially private median. In *29th USENIX Security Symposium (USENIX Security 20)* (pp. 2147–2164).

[8] Castaneda, H., Zambrano, F., & Parra Ortega, C. (2010, August). Methods for least square estimation of nonlinear parameter in system identification. *Revista colombiana de tecnología avanzada*.

[9] Chen, X., Wu, S. Z., & Hong, M. (2020). Understanding gradient clipping in private SGD: A geometric perspective. *Advances in Neural Information Processing Systems, 33*, 13773–13782.

[10] Dunn, J. C. (1973). A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. Taylor & Francis.

[11] Dwork, C., McSherry, F., Nissim, K., & Smith, A. (2016). Calibrating noise to sensitivity in private data analysis. *Journal of Privacy and Confidentiality, 7*(3), 17–51.

[12] Gobierno de la Ciudad de México. (2024). *Deuda pública de la Ciudad de México*. Base de datos. Portal de Datos Abiertos de la Ciudad de México. Recuperado de `https://datos.cdmx.gob.mx`

[13] Gong, M., Liang, Y., Shi, J., Ma, W., & Ma, J. (2012). Fuzzy c-means clustering with local information and kernel metric for image segmentation. *IEEE Transactions on Image Processing, 22*(2), 573–584. IEEE.

[14] Goodfellow, I. (2016). *Deep learning*. Cambridge, MA: MIT Press.

[15] Güler, O. (1991). On the convergence of the proximal point algorithm for convex minimization. *SIAM Journal on Control and Optimization, 29*(2), 403–419.

[16] Hastie, T., Tibshirani, R., Friedman, J. H., & Friedman, J. H. (2009). *The elements of statistical learning: Data mining, inference, and prediction* (2nd ed.). New York: Springer.

[17] INEGI. (2021). *Estadísticas de Defunciones Registradas 2021*. Recuperado el 30 de mayo de 2023, de `https://www.inegi.org.mx/programas/mortalidad/`

[18] Jalabneh, R., Syed, H. Z., Pillai, S., et al. (2021). Use of mobile phone apps for contact tracing to control the COVID-19 pandemic: A literature review. *Applications of Artificial Intelligence in COVID-19*, 389-404.

[19] James, G., Witten, D., Hastie, T., Tibshirani, R. (2013). *An introduction to statistical learning* (Vol. 112). Springer.

[20] Kelley, J. L. (1955). *General topology*. D. Van Nastrand Company.

[21] Krupadharamshi, K. (2024). *Fuel Consumption Dataset - Vehículos del año 2000*. Base de datos. Kaggle. Recuperado de `https://www.kaggle.com/datasets/krupadharamshi/fuelconsumption/data`

[22] Li, N., Li, T., & Venkatasubramanian, S. (2006). t-closeness: Privacy beyond k-anonymity and l-diversity. In *2007 IEEE 23rd International Conference on Data Engineering* (pp. 106–115). IEEE.

[23] Li, N., Lyu, M., Su, D., & Yang, W. (2017). *Differential privacy: From theory to practice*. Springer.

[24] Lian, X., Zhang, C., Zhang, H., Hsieh, C.-J., Zhang, W., & Liu, J. (2017). Can decentralized algorithms outperform centralized algorithms? A case study for decentralized parallel stochastic gradient descent. *Advances in Neural Information Processing Systems, 30*, 1–12.

[25] Machanavajjhala, A., Kifer, D., Gehrke, J., & Venkitasubramaniam, M. (2007). l-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data (TKDD), 1*(1), 3–es.

[26] Massey Jr, F. J. (1951). The Kolmogorov-Smirnov test for goodness of fit. *Journal of the American Statistical Association, 46*(253), 68–78.

[27] Márquez, A. (2022). *La privacidad diferencial cómo solución a la amenaza de la privacidad en los datos biológicos*. Universitat Oberta de Catalunya.

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

[29] Microsoft Corporation. (2024). *Diabetes Dataset - Azure Open Datasets*. Base de datos. Conjuntos de Datos Abiertos de Azure (Microsoft). Recuperado de `https://learn.microsoft.com/es-mx/azure/open-datasets/dataset-catalog`

[30] Mitchell, T. M. (1997). *Machine learning*. New York: McGraw-Hill.

[31] Nergiz, M. E., Atzori, M., & Clifton, C. (2007). Hiding the presence of individuals from shared databases. In *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data* (pp. 665–676).

[32] Pitsch, W., & Emrich, E. (2012). The frequency of doping in elite sport: Results of a replication study. *International Review for the Sociology of Sport, 47*(5), 559–580. Sage Publications Sage UK: London, England.

[33] Schonlau, M. (2023). The Naive Bayes classifier. In *Applied Statistical Learning: With Case Studies in Stata* (pp. 143–160). Cham: Springer International Publishing.

[34] Shakiba, A. (2018). Differentially private fuzzy C-means clustering algorithms for fuzzy datasets. In *2018 6th Iranian Joint Congress on Fuzzy and Intelligent Systems (CFIS)* (pp. 91–93). IEEE.

[35] Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal, 27*(3), 379–423.

[36] Sweeney, L. (2002). k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 10*(05), 557–570.

[37] Vaidya, J., Basu, A., Shafiq, B., & Hong, Y. (2013). Differentially private naïve bayes classification. In *Proceedings - 2013 IEEE/WIC/ACM International Conference on Web Intelligence* (pp. 571–576). IEEE Computer Society.

[38] Wasserman, L., & Zhou, S. (2010). A statistical framework for differential privacy. *Journal of the American Statistical Association, 105*(489), 375–389. Taylor & Francis.

[39] Wei, K., Li, J., Ding, M., Ma, C., Yang, H., Faramarz, F., & Poor, H. V. (2020). Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security, 15*, 3454–3469.

[40] Xiao, L., & Boyd, S. (2004). Fast linear iterations for distributed averaging. *Systems & Control Letters, 53*(1), 65–78.

[41] Xu, J., Glicksberg, B. S., Su, C.-H., Walker, P., Bian, J., & Wang, F. Z. (2021). Federated learning for healthcare informatics. *Journal of Healthcare Informatics Research, 5*, 1–19.

[42] Yang, Q., Liu, Y., Chen, T., & Tong, Y. (2019). Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology, 10*(2), 1–19.

[43] Zadeh, L. A. (1965). Fuzzy sets. *Information and Control, 8*(3), 338–353. Elsevier.

[44] Zaman, Z., Xue, W., Gauravaram, P., Hu, W., & Jha, S. K. (2023). Privacy-preserving probabilistic data encoding for IoT data analysis.

[45] Zhang, J., Zhang, Z., Xiao, X., Yang, Y., & Winslett, M. (2012). Functional mechanism: Regression analysis under differential privacy. *Proceedings of the VLDB Endowment, 5*(11), 1364-1375.

[46] Zhang, Y., & Han, J. (2021). Differential privacy fuzzy C-means clustering algorithm based on Gaussian kernel function. *PLOS ONE, 16*(3), e0248737.

[47] Zhou, X., Xu, M., Wu, Y., & Zheng, N. (2021). Deep model poisoning attack on federated learning. *Future Internet, 13*(3), 73.