Universidad de Valladolid

FACULTAD DE CIENCIAS

TRABAJO FIN DE GRADO

Grado en Estadística

INFERENCIA VARIACIONAL GAUSSIANA Y SUS APLICACIONES

Autor/a: Mikel Izkue Urdaniz
Tutor/es: Eustasio del Barrio Tellado
Año: 2025

Índice general

1. Introducción **4**

2. Marco teórico: Inferencia Bayesiana y Variacional **6**
2.1. Inferencia Bayesiana 6
2.2. Métodos de Inferencia Aproximada 8
2.2.1. El método MCMC. 10
2.2.2. Métodos basados en optimización 12
2.3. Inferencia variacional 12
2.3.1. ELBO: Evidence Lower Bound 13
2.4. Familia Variacional y Aproximaciones 15
2.4.1. Inferencia variacional mediante CAVI para un modelo generativo simple 16

3. La Inferencia Variacional Gaussiana (IVG) **19**
3.1. Definición y Motivación 19
3.2. Formulación Matemática 19
3.3. Versiones del Modelo 20
3.3.1. Algoritmo CAVI con covarianza diagonal 20
3.4. Optimización mediante Gradiente Natural 21
3.5. Ventajas de la IVG 21
3.6. Limitaciones 21
3.7. Derivación de la ELBO paso a paso 22

4. Aplicaciones de la Inferencia Variacional Gaussiana (IVG) **24**
4.1. Modelos de mezcla de gaussianas (GMM) 24
4.2. Latent Dirichlet Allocation (LDA) 25
4.3. Variational Autoencoders (VAEs) 26
4.4. Regresión Logística Bayesiana 28
4.5. Modelos Jerárquicos en Biomedicina 28
4.6. Series Temporales y Modelos de Estado Latente 29
4.7. Sistemas de Recomendación 30
4.8. Resumen comparativo 32

5. Optimización en Inferencia Variacional Gaussiana (IVG) **33**
5.1. Problema de optimización en IVG 33
5.2. Reparametrización de Monte Carlo 34
5.3. Descenso por Gradiente Natural 34

5.4. Optimización práctica con Adam 34
5.5. Consideraciones numéricas 34
5.6. Resumen de técnicas empleadas en IVG 35

6. Aplicación: Inferencia Variacional Gaussiana en un Modelo de Mezcla de Normales 36
6.1. Formulación del modelo 36
6.2. Inferencia variacional 36
6.3. Implementación práctica 37
6.4. Ejemplo simulado 37
6.5. Discusión 38

7. Conclusión 39

Resumen

La inferencia bayesiana proporciona un marco teórico sólido para el análisis de datos inciertos, pero su aplicación práctica se ve limitada por la dificultad de calcular distribuciones posteriores complejas. La inferencia variacional surge como una alternativa eficiente al reformular el problema de inferencia como uno de optimización. Este trabajo se centra en la Inferencia Variacional Gaussiana (IVG), una técnica que restringe la familia de distribuciones aproximantes a las gaussianas multivariadas, permitiendo simplificar el proceso de inferencia sin renunciar a una buena capacidad de aproximación.

A lo largo del documento se analizan los fundamentos teóricos de IVG, sus propiedades computacionales y su implementación mediante técnicas modernas como el truco de reparametrización, el gradiente natural y el uso de optimizadores adaptativos. Además, se presentan ejemplos prácticos que ilustran la eficacia de IVG en diversos contextos de modelado probabilístico. El objetivo es ofrecer una visión clara y aplicada de una herramienta clave en la inferencia bayesiana aproximada.

Abstract

Bayesian inference provides a solid theoretical framework for modeling uncertainty in data analysis. However, its practical application is often hindered by the intractability of computing complex posterior distributions. Variational inference offers an efficient alternative by reformulating the inference task as an optimization problem. This work focuses on Gaussian Variational Inference (GVI), a method that restricts the approximating family to multivariate Gaussian distributions, simplifying the inference process while retaining good approximation capabilities.

Throughout this document, we analyze the theoretical foundations of GVI, its computational properties, and its implementation using modern techniques such as the reparameterization trick, natural gradient, and adaptive optimizers. Practical examples are also provided to demonstrate the effectiveness of GVI in various probabilistic modeling contexts. The aim is to present a clear and applied overview of a key tool in approximate Bayesian inference.

Capítulo 1

Introducción

La inferencia estadística es una herramienta esencial para modelar fenómenos inciertos a partir de datos. En el marco bayesiano, los parámetros desconocidos se tratan como variables aleatorias, y el objetivo principal es caracterizar su distribución posterior condicionada a las observaciones disponibles. Sin embargo, calcular esta distribución posterior de forma exacta implica resolver integrales de alta dimensión, lo cual rara vez es factible en modelos reales, especialmente en aquellos con estructuras complejas o con un número elevado de parámetros. Ante esta dificultad, se han desarrollado diversas aproximaciones. En las últimas décadas, dos enfoques principales han dominado el campo de la inferencia bayesiana aproximada: los métodos de Monte Carlo por cadenas de Markov (MCMC) y la inferencia variacional (VI). Mientras que los métodos MCMC ofrecen garantías asintóticas de convergencia a la distribución posterior verdadera, suelen ser costosos computacionalmente. Por el contrario, la inferencia variacional transforma el problema de inferencia en uno de optimización, buscando dentro de una familia de distribuciones una aproximación que minimice la divergencia frente a la verdadera posterior.

Dentro del marco de la inferencia variacional, una de las aproximaciones más extendidas es la Inferencia Variacional Gaussiana (IVG). Este enfoque asume que la distribución posterior puede aproximarse por una gaussiana multivariada, lo que permite simplificar el cálculo, aprovechar propiedades analíticas conocidas y aplicar técnicas de optimización eficientes y escalables. A pesar de su simplicidad, IVG ha demostrado ser sorprendentemente efectiva en múltiples contextos prácticos.

El objetivo principal de este trabajo es ofrecer una exposición clara, estructurada y rigurosa de los fundamentos teóricos y computacionales de la inferencia variacional gaussiana, así como ilustrar su aplicación en distintos tipos de modelos probabilísticos.

Para ello, el documento se organiza de la siguiente manera:

- En el **Capítulo 1**, se introduce el contexto de la inferencia bayesiana y las motivaciones detrás del uso de aproximaciones.
- El **Capítulo 2** presenta los fundamentos de la inferencia variacional, incluyendo la derivación de la ELBO y los principios de optimización subyacentes.
- En el **Capítulo 3**, se describe en detalle la formulación de la IVG, sus ventajas, limitaciones y su relación con otras técnicas.

- El **Capítulo 4** muestra aplicaciones prácticas de IVG en distintos modelos, lo que permite ilustrar su utilidad en problemas reales.
- El **Capítulo 5** se enfoca en los métodos de optimización utilizados para implementar IVG de forma eficiente, incluyendo el truco de reparametrización, el uso del gradiente natural y algoritmos como Adam.
- Finalmente, en la **Conclusión**, se sintetizan los aportes del trabajo y se proponen posibles extensiones y líneas de investigación futura.

Este recorrido permitirá al lector adquirir una comprensión profunda del papel que juega la Inferencia Variacional Gaussiana dentro del conjunto de técnicas modernas para el modelado probabilístico.

Capítulo 2

Marco teórico: Inferencia Bayesiana y Variacional

2.1. Inferencia Bayesiana

La inferencia bayesiana es un enfoque estadístico en el que las creencias sobre parámetros desconocidos se representan mediante distribuciones de probabilidad. Estas creencias se actualizan de forma sistemática al observar datos nuevos, utilizando el llamado *teorema de Bayes*.

El teorema de Bayes constituye el núcleo matemático de la inferencia estadística bayesiana. Fue formulado originalmente por Thomas Bayes en el siglo XVIII y publicado póstumamente en 1763. Su relevancia ha crecido enormemente en las últimas décadas gracias al auge del aprendizaje automático, donde proporciona un marco natural para la actualización de creencias.

Matemáticamente, el teorema de Bayes se expresa de la siguiente forma:

$$p(\theta|x) = \frac{p(\theta)p(x|\theta)}{p(x)}, \tag{2.1}$$

donde,

- θ representa los parámetros latentes del modelo,
- x son los datos observados,
- $p(\theta)$ es la distribución a priori,
- $p(x|\theta)$ es la verosimilitud
- $p(\theta|x)$ es la distribución a posteriori.

Este resultado permite combinar de manera formal el conocimiento previo con la información extraída de los datos observados, obteniendo una descripción probabilística completa del problema de inferencia.

Ejemplo clásico: Inferencia sobre una moneda sesgada

Supongamos que lanzamos una moneda diez veces y observamos ocho caras. Queremos inferir el sesgo θ, es decir, la probabilidad de obtener cara. Usamos como modelo una distribución binomial:

$$p(x|\theta) = \binom{10}{8}\theta^8(1-\theta)^2.$$

Si asumimos una distribución a priori uniforme, es decir $p(\theta) = 1$ para $\theta \in [0, 1]$, la distribución a posteriori es proporcional a la verosimilitud.
Esta distribución posterior es una Beta$(9, 3)$. Así, el teorema de Bayes nos permite actualizar la incertidumbre sobre θ de forma explícita.

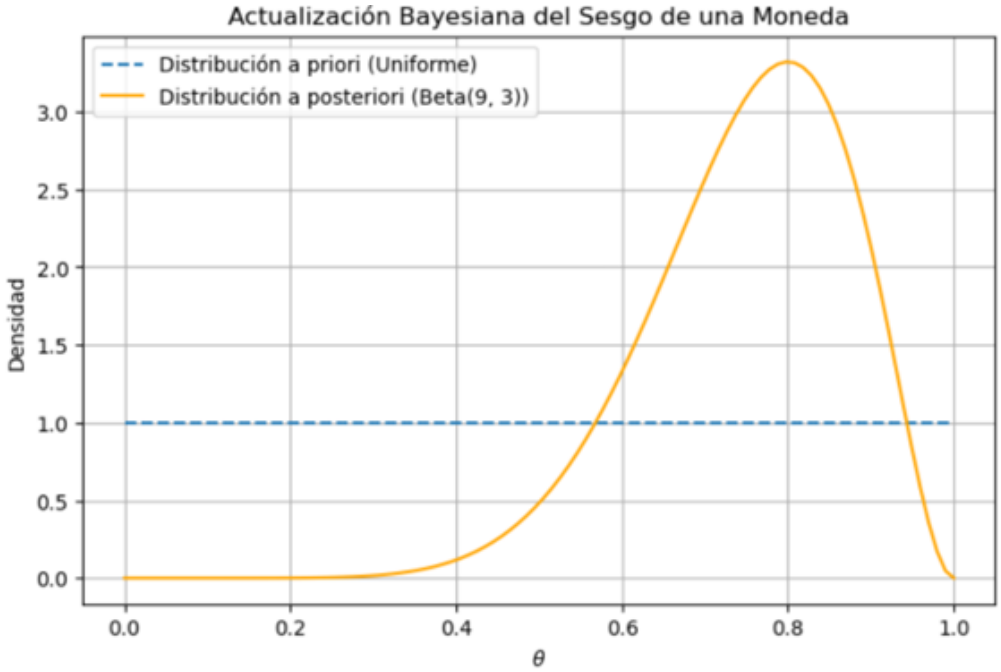

Figura 2.1: Comparación entre la distribución a priori uniforme y la distribución a posteriori

Como se observa en la Figura 2.1, la distribución a priori uniforme refleja una total incertidumbre sobre el valor de θ antes de observar los datos. Sin embargo, tras observar ocho caras en diez lanzamientos, la distribución a posteriori Beta$(9, 3)$ se concentra alrededor de $\theta \approx 0.8$, lo cual refleja una actualización de nuestra creencia: ahora consideramos mucho más probable que la moneda esté sesgada hacia cara.
Esta es una ventaja fundamental del enfoque bayesiano: la capacidad de incorporar de forma coherente la evidencia observada y actualizar nuestras creencias de manera explícita.
Esta es una ventaja fundamental del enfoque bayesiano: la capacidad de incorporar de forma coherente la evidencia observada y actualizar nuestras creencias de manera explícita.

Dificultades computacionales

En modelos simples como el anterior, la evidencia $p(x)$ puede calcularse analíticamente. Sin embargo, en la mayoría de los modelos reales, esta constante de normalización implica una

integral de alta dimensión. De hecho, la principal dificultad en la inferencia bayesiana radica en el cálculo de la constante de normalización

$$p(x) = \int p(x|\theta)p(\theta)d\theta, \tag{2.2}$$

también conocida como la *evidencia*. Esta integral es frecuentemente intratable en modelos complejos o de alta dimensión, lo que hace que el cálculo de la posterior exacta no sea viable en la práctica.

2.2. Métodos de Inferencia Aproximada

Para superar esta dificultad, se han desarrollado varios métodos de inferencia aproximada. Entre los más conocidos está el método de cadena de Markov Monte Carlo (abreviado como MCMC). Este método es una de las aplicaciones más interesantes del Teoréma Ergódico para CMTDH, que se estudia en la asignatura de Procesos Estocásticos del Grado en Estadística. El procedimiento es una herramienta fundamental en Estadística Bayesiana y se considera uno de los algoritmos más importantes del siglo XX. A continuación se describen los detalles principales del método.

El método de Monte Carlo

El objetivo de la simulación MCMC es la generación artificial de observaciones con una distribución dada. Esto permitiría calcular valores esperados de funciones de dicha distribución. Supongamos que Z es un vector aleatorio k-dimensional con función de densidad $f(x)$. La evaluación de valores esperados de funciones de Z mediante la expresión

$$E(\phi(Z)) = \int_{\mathbb{R}^k} \phi(x)f(x)dx$$

sólo es posible en algunos casos especiales y lo habitual es que sea necesario emplear algún método de integración numérica. Especialmente si k es grande resulta competitivo emplear el *método de Monte Carlo*: si es posible generar v.a.i.i.d. $Z_1, Z_2, \ldots$ con la misma distribución que Z entonces la Ley de los Grandes Números implica

$$\frac{1}{n}\sum_{i=1}^{n} \phi(Z_i) \to_{\text{c.s.}} E(\phi(Z)).$$

Generando suficientes réplicas Z_i, el promedio en el lado izquierdo anterior será una buena aproximación a la integral. Más aún, con el método de los intervalos de confianza se puede dar una cota para el error de aproximación (con un determinado nivel de confianza).

La *simulación* o generación de réplicas i.i.d. no es siempre una tarea fácil. En el caso en el que Z es una variable aleatoria unidimensional con función de distribución F se puede tratar

de usar el método de la inversa. Se define

$$F^{-1}(t) = \text{ínf}\left\{x \in \mathbb{R} : F(x) \geq t\right\}, \quad t \in (0,1).$$

F^{-1} es la *inversa cuantil*. Es inmediato que $F^{-1}(t) \leq x$ si y sólo si $t \leq F(x)$ y, a partir de aquí, que si $U \sim U(0,1)$ entonces $F^{-1}(U)$ tiene función de distribución F. Dando por hecho que podemos generar réplicas i.i.d. con distribución $U(0,1)$ (lo que es posible con cualquier ordenador actual) el problema de la simulación estaría resuelto. Claro está que esto sólo es posible para ditribuciones unidimensionales e incluso en ese caso la evaluación de F^{-1} puede ser problemática.

El método de aceptación rechazo

Una alternativa al método de la inversa es el *método de aceptación-rechazo*. Aunque el objetivo sigue siendo generar vectores aleatorios con densidad f, supongamos que somos capaces de generar $\{Y_n\}_{n\geq 1}$, i.i.d. con densidad g, de forma que

$$f(x) \leq cg(x), \quad x \in \mathbb{R}^k,$$

para cierta constante $c > 1$. Contamos además con $\{U_n\}_{n\geq 1}$ i.i.d. $U(0,1)$, independientes de las Y_n. Definimos

$$\tau = \text{ínf}\left\{ n \geq 1 : U_n \leq \frac{f(Y_n)}{cg(Y_n)} \right\} \tag{2.3}$$

y $Z = Y_\tau$. Se puede comprobar que Z es un vector aleatorio con densidad f (ver [10]).

La construcción anterior se utiliza para generar vectores aleatorios con densidad f de la siguiente manera. Se genera un *candidato*: un vector aleatorio Y_1 con densidad g. Este candidato se puede *aceptar* o *rechazar*. Esto se decide en función de un sorteo aleatorio: se genera $U_1 \sim U(0,1)$. Si $U_1 \leq \frac{f(Y_1)}{cg(Y_1)}$ entonces se acepta y $X_1 = Y_1$; en caso contrario se rechaza Y_1, se genera otro candidato y se repite el proceso anterior hasta que un candidato es aceptado y tomamos X_1 igual al candidato aceptado. Se itera el procedimiento para obtener más réplicas $X_2, X_3, \ldots$ De esta forma $X_1, X_2, \ldots$ son v.a.i.i.d. con densidad f. Se puede comprobar que el número medio de candidatos a generar hasta conseguir una observación aceptable es $\mathbb{E}(\tau) = c$. El procedimiento funciona igual si las distribuciones distribuciones de interés son discretas y f y g son las funciones de masa de probabilidad asociadas.
El algoritmo asociado al argumento anterior se escribiría en pseudo-código de la siguiente forma

- $i = 1$

(a.1) se genera candidato $Y_i \sim g$

(a.2) se genera $U_i \sim U(0,1)$

- si $U_i > \frac{f(Y_i)}{cg(Y_i)}$ rechazo: $i = i + 1$, vuelta a (a.1)

- si $U_i \leq \frac{f(Y_i)}{cg(Y_i)}$ aceptación: $Z = Y_i$; stop

Iterando este procedimiento se obtienen réplicas $Z_1, Z_2 \ldots$ i.i.d. f.

El método de aceptación-rechazo (en adelante AR) se puede emplear, en principio, para generar vectores aleatorios de dimensión general. Puede ocurrir, sin embargo, que en la práctica f sea conocida a falta de una constante de proporcionalidad. Teóricamente esto no es ningún problema. Si $f = K\tilde{f}$, con $\tilde{f} \geq 0$, entonces $K = \left(\int_{\mathbb{R}^k} \tilde{f}(x)dx\right)^{-1}$ para que f sea una densidad. Pero el cálculo de esta integral puede ser de una complejidad equivalente a la del valor objetivo, $\int_{\mathbb{R}^k} \phi(x)f(x)dx$, y la igualdad anterior es de escasa utilidad en la práctica.

2.2.1. El método MCMC.

La simulación de cadena de Markov Monte Carlo (en adelante MCMC) es un método inspirado en el método AR que permite generar sequencias que son una cadena de Markov ergódica (irreducible, persistente positiva y aperiódica) con distribución estacionaria prefijada. Desarrollamos la idea en el caso de distribuciones discretas, aunque se pueden emplear esencialmente los mismos métodos en el caso de distribuciones continuas. Supongamos entonces que $\pi = (\pi_i)_{i\in E}$ es la distribución objetivo, es decir, que estamos interesados en aproximar expresiones del tipo

$$E_\pi(g(Z)) = \sum_{i\in E} \pi_i \phi(i).$$

Sin pérdida de generalidad se puede asumir que $\pi_i > 0$ para todo $i \in E$. Si $\{Z_n\}_{n\geq 0}$ fuese una CMTDH irreducible y persistente con distribución estacionaria π entonces el Teorema Ergódico garantiza que

$$\frac{1}{N}\sum_{n=1}^{N} \phi(Z_i) \to_{\text{c.s.}} \sum_{i\in E} \pi_i \phi(i)$$

independientemente de la distribución inicial. Esta es la idea fundamental del método MCMC. El problema es, entonces, saber cómo generar una cadena $\{Z_n\}_{n\geq 0}$ con distribución estacionaria π.

En general hay muchas formas de elegir una matriz de transición $\mathbb{P} = [p_{i,j}]_{i,j\in E}$ de forma que π es una distribución estacionaria para $\mathbb{P}$. Se suele buscar $\mathbb{P}$ de forma que se satisfagan las condiciones de reversibilidad,

$$\pi_i p_{i,j} = \pi_j p_{j,i}, \quad i, j \in E. \tag{2.4}$$

El conjunto de métodos MCMC conocido como *algoritmo de Metropolis-Hastings* considera matrices de transición de la forma

$$p_{i,j} = q_{i,j}\alpha_{i,j}$$

para $j \neq i$, $p_{i,i} = 1 - \sum_{j\neq i} p_{i,j}$. La matriz $[q_{i,j}]_{i,j\in E}$ es la *matriz generadora de candidatos* y $[\alpha_{i,j}]_{i,j\in E}$ es la *matriz de aceptación*. La interpretación es que si la cadena está en el estado i, el siguiente candidato será el j con probabilidad $q_{i,j}$. Si $j \neq i$ es el candidato seleccionado, se

acepta con probabilidad $\alpha_{i,j}$; de lo contrario se permanece en i. Una elección posible para las probabilidades de aceptación es

$$\alpha_{i,j} = \frac{s_{i,j}}{1 + t_{i,j}},$$

donde $\Sigma = [s_{i,j}]_{i,j \in E}$ es una matriz simétrica y

$$t_{i,j} = \frac{\pi_i q_{i,j}}{\pi_j q_{j,i}}.$$

La matriz Σ se debe elegir de forma que $\alpha_{i,j} \in [0, 1]$, es decir, satisfaciendo

$$s_{i,j} \leq 1 + \text{mín}(t_{i,j}, t_{j,i}).$$

Se puede probar que si $[q_{i,j}]_{i,j \in E}$ es una matriz de transición irreducible entonces la matriz $\mathbb{P} = [p_{i,j}]_{i,j \in E}$ anteriormente definida es una matriz de transición irreducible que satisface las ecuaciones (2.4). Además, si $\alpha_{i,j} > 0$ siempre que $q_{i,j} > 0$, $i \neq j$, entonces $\mathbb{P}$ es irreducible y persistente positiva y su distribución estacionaria es π.

La construcción anterior deja cierta flexibilidad en la elección de las probabilidades de aceptación. El *algoritmo de Metropolis* corresponde al caso $s_{i,j} = 1 + \text{mín}(t_{i,j}, t_{j,i})$, es decir,

$$\alpha_{i,j} = \text{mín}\left(1, \frac{\pi_j q_{j,i}}{\pi_i q_{i,j}}\right).$$

En el caso particular en el que la generación de candidatos se hace puramente al azar ($q_{i,j}$ =constante) las probabilidades de aceptación son

$$\alpha_{i,j} = \text{mín}\left(1, \frac{\pi_j}{\pi_i}\right).$$

Otra elección frecuente es el *algoritmo de Barker*. Ahora $s_{i,j} = 1$ y

$$\alpha_{i,j} = \frac{\pi_j q_{j,i}}{\pi_i q_{i,j} + \pi_j q_{j,i}}.$$

Con generación de candidatos puramente al azar las probabilidades de aceptación son

$$\alpha_{i,j} = \frac{\pi_j}{\pi_i + \pi_j}.$$

Una observación importante es que si la matriz generadora de candidatos es conocida, los algoritmos de Metropolis-Hastings se pueden implementar sin conocimiento completo de π. En realidad es suficiente conocer π a falta de una constante de proporcionalidad. Esto es así porque las probabilidades de aceptación dependen únicamente de los cocientes $\frac{\pi_j}{\pi_i}$.

Tal como se ha comentado, los algoritmos tipo Metropolis-Hastings son adaptables al caso de distribuciones con densidad y tienen la ventaja de que se pueden implementar sin conocer completamente la densidad objetivo. Basta con conocerla a falta de una constante de proporcionalidad. Esta es la situación típica en inferencia bayesiana, en la que la distribu-

ción a posteriori se conoce salvo la constante $p(x)$. Esto explica que los métodos MCMC se convirtiesen en una herramienta fundamental en inferencia bayesiana.
Sin embargo, los métodos MCMC también tienen limitaciones. Aunque son asintóticamente exactos, su coste computacional es alto, especialmente en grandes volúmenes de datos o en modelos con estructuras jerárquicas.

2.2.2. Métodos basados en optimización

En situaciones como las mencionadas anteriormente, en las que la simulación MCMC resulta demasiado costosa, se puede plantear la alternativa de buscar una distribución $q(\theta)$ que sea lo más parecida posible a la distribución objetivo (en el contexto de la inferencia bayesiana ésta será la distribución a posteriori, $p(\theta|x)$). La distribución $q(\theta)$ se buscará dentro de una clase manejable, $\mathcal{Q}$. El objetivo se convierte en resolver el problema de minimización

$$q^*(\theta) = \arg\min_{q \in \mathcal{Q}} d(q(\theta), p(\theta|x)), \tag{2.5}$$

donde d es una medida de discrepancia entre distribuciones. Este enfoque convierte el problema de inferencia en uno de optimización funcional. La idea es que puede ser más tratable generar observaciones de q y que posiblemente vale la pena sacrificar la exactitud asintótica del método MCMC por ventajas computacionales. Esto, sin embargo, tiene algunas dificultades. En primer lugar hay que elegir una medida de discrepancia adecuada. Hay muchas métricas o divergencias entre probabilidades y no está claro a priori cuál de ellas puede ofrecer mejores resultados. Pero un problema más importante es que, en realidad, para resolver el problema (2.5) parece necesario conocer $p(\theta|x)$, porque de otra forma parece imposible determinar qué representante en $\mathcal{Q}$ está más cerca de $p(\theta|x)$. Y esto es, en principio, un problema insalvable, porque, precisamente, lo que queremos es aproximar $p(\theta|x)$ dado que no la conocemos completamente.

2.3. Inferencia variacional

La inferencia variacional (VI) es una solución a las consideraciones anteriores. En la inferencia variacional se busca una distribución aproximada $q(\theta)$ dentro de una familia simple (por ejemplo, gaussianas factorizadas) que minimice la divergencia de Kullback- Leibler (KL) respecto a la posterior:

$$q^*(\theta) = \operatorname*{argmin}_{q \in \mathcal{Q}} \mathrm{KL}(q(\theta) \| p(\theta|x)). \tag{2.6}$$

Esta es la formulación (2.5) en el caso particular en el que elegimos como discrepancia d la divergencia de Kullback-Leibler (KL). Esta es una medida fundamental en teoría de la información y estadística. En el contexto de la inferencia variacional, cumple un papel central, ya que toda la formulación de VI se basa en minimizar esta divergencia entre la distribución aproximada y la verdadera posterior.
En el caso de distribuciones con densidad la divergencia de Kullback-Leibler se define como

sigue:

$$\mathrm{KL}(q(\theta)\|p(\theta)) = \int \log \frac{q(\theta)}{p(\theta)} q(\theta) d\theta. \tag{2.7}$$

Es importante notar que:

- La divergencia KL no es simétrica: $\mathrm{KL}(q\|p) \neq \mathrm{KL}(p\|q)$,
- $\mathrm{KL}(q\|p) \geq 0$ y se da la igualdad si y sólo si $q = p$.

La última propiedad nos dice que KL es una medida de "distancia informativa" que cuantifica cuánto se aleja q de p.

2.3.1. ELBO: Evidence Lower Bound

El problema de optimización (2.6) sigue presentando, en principio, el problema de que para resolverlo es necesario conocer $p(\theta|x)$, lo que supondría una dificultad insalvable. Pero esto es sólo aparente, porque el problema se puede reformular de una forma equivalente en la que es suficiente conocer la verosimilitud del modelo. La conexión es a través de la llamada *Evidence Lower Bound* (ELBO), definida como:

$$\mathrm{ELBO}(q) = \mathbb{E}_q[\log p(x, \theta)] - \mathbb{E}_q[\log q(\theta)]. \tag{2.8}$$

Maximizar esta cantidad es equivalente a minimizar la divergencia KL entre la distribución aproximada y la posterior exacta. Para comprobar esta equivalencia recurrimos a la siguiente igualdad, que se comprueba fácilmente:

$$\mathrm{KL}(q(\theta)\|p(\theta|x)) = \mathbb{E}_q[\log q(\theta)] - \mathbb{E}_q[\log p(x, \theta)] + \log p(x).$$

Reordenando, obtenemos

$$\log p(x) = \mathrm{ELBO}(q) + \mathrm{KL}(q(\theta)\|p(\theta|x)). \tag{2.9}$$

De esta expresión se obtienen dos consecuencias importantes. Por un lado, dado que $\log p(x)$ es constante (no depende de q), maximizar la ELBO es equivalente a minimizar la divergencia $\mathrm{KL}(q(\theta)\|p(\theta|x))$. Por otra parte, como $\mathrm{KL}(q(\theta)\|p(\theta|x)) \geq 0$ concluimos que

$$\log p(x) \geq \mathrm{ELBO}(q),$$

lo que justifica el nombre del término.
Intuitivamente, el primer término en (2.9) recompensa distribuciones q que explican bien los datos, mientras que el segundo penaliza aquellas que se desvían mucho de la verdadera distribución a posteriori.
La ELBO tiene dos ventajas clave:

1. Proporciona un criterio claro de optimización para seleccionar la mejor distribución aproximante.

2. Actúa como cota inferior del logaritmo de la evidencia, $\log p(x)$, Figura 2.2 lo que permite evaluar modelos comparativamente.

Hay que insistir en que minimizar $\mathrm{KL}(q(\theta)\|p(\theta|x))$ implica penalizar aquellas regiones donde $q(\theta)$ asigna masa y $p(\theta|x)$ no. Sin embargo, no penaliza de forma fuerte el caso opuesto: si q omite regiones de alta probabilidad bajo la distribución a posteriori, la penalización puede ser pequeña. Esto tiene una consecuencia importante. La inferencia variacional puede subestimar la incertidumbre, especialmente si la posterior verdadera es multimodal o con colas pesadas, ya que $q^*(\theta)$, al minimizar $\mathrm{KL}(q(\theta)\|p(\theta|x))$, tiende a concentrarse en una sola moda. Esto se ilustra más adelante con un ejemplo.

Representación gráfica

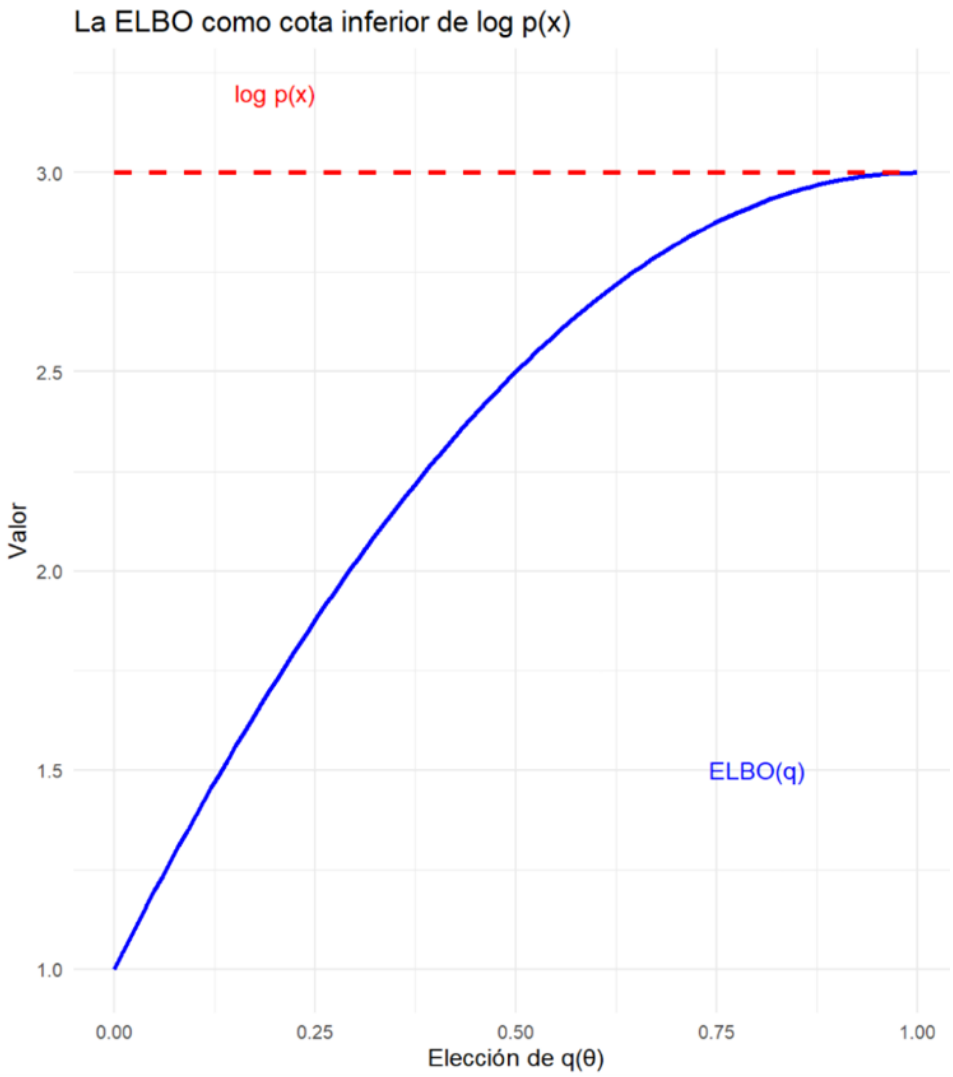

Figura 2.2: Interpretación gráfica de la ELBO como cota inferior

La Figura 2.2 representa gráficamente la relación entre la Evidence Lower Bound (ELBO) y la evidencia marginal $\log p(x)$. En este esquema, cada punto sobre el eje horizontal simboliza una elección distinta de la distribución variacional $q(\theta)$ dentro de la familia aproximante. La línea

roja discontinua indica el valor constante de $\log p(x)$, el cual es generalmente desconocido, pero actúa como una cota superior. La curva azul representa el valor de la ELBO en función de la calidad de la aproximación $q(\theta)$.
Como se observa:

- La ELBO siempre se encuentra por debajo o igual a $\log p(x)$. Esta propiedad es una consecuencia directa de la no negatividad de la divergencia KL.
- La ELBO alcanza su valor máximo cuando $q(\theta) = p(\theta \mid x)$, lo que implica una divergencia KL nula.
- Este gráfico enfatiza la idea de que maximizar la ELBO equivale a minimizar la KL entre la aproximación y la posterior verdadera.

Este tipo de visualización resulta útil no solo para entender el comportamiento de los métodos variacionales, sino también para interpretar los valores numéricos de la ELBO durante el entrenamiento de modelos bayesianos aproximados.

2.4. Familia Variacional y Aproximaciones

Un aspecto crucial en VI es la elección de la familia de distribuciones $\mathcal{Q}$. La elección más común es la aproximación *mean-field*, que asume independencia entre los parámetros:

$$q(\theta_1, \ldots, \theta_d) = \prod_{i=1}^{d} q_i(\theta_i). \tag{2.10}$$

Esta simplificación permite derivar actualizaciones explícitas mediante algoritmos como Coordinate Ascent Variational Inference (CAVI), donde se optimiza cada factor q_i en iteraciones sucesivas.
Para entender la importancia de elegir familias del tipo (2.10) y el fundamento del algoritmo CAVI, vamos a reescribir el término ELBO(q) como función del factor q_j, dejando fijo las demás marginales q_i, $i \neq j$. Obtenemos entonces

$$\text{ELBO}(q) = \mathbb{E}_j[\mathbb{E}_{-j} \log p(\theta_j, \theta_{-\mathbf{j}}, x)] - \mathbb{E}_j \log q_j(\theta_j) + \text{C},$$

donde $\mathbb{E}_j$ denota esperanza con respecto a $q_j(\theta_j)$, $\mathbb{E}_{-j}$ es la esperanza condicionalmente dada θ_j (esperanza respecto a las demás variables) y C es un término que no depende de q_j. Si denotamos $\text{ELBO}(q_j) = \mathbb{E}_j[\mathbb{E}_{-j} \log p(\theta_j, \theta_{-\mathbf{j}}, x)] - \mathbb{E}_j \log q_j(\theta_j)$ entonces vemos que $\text{ELBO}(q_j)$ es (salvo un factor de proporcionalidad) menos la divergencia de Kullback-Leibler de $q_j(\theta_j)$ a la probabilidad proporcional a $\exp(\log p(\theta_j, \theta_{-\mathbf{j}}, x))$. Dicho de otra forma, si, manteniendo fijos los factores $q_i(\theta_i)$, $i \neq j$, tratamos de encontrar el factor q_j tal que el producto (2.10) maximiza el término ELBO(q), entonces, ese máximo se alcanza con la elección

$$q_j^*(\theta_j) \propto \exp(\log p(\theta_j, \theta_{-\mathbf{j}}, x)). \tag{2.11}$$

En algunos casos esta ecuación conduce a un término q_j^* que está dentro de la familia $\mathcal{Q}$. Es en esos casos en los que se tiene más sentido implementar el algoritmo CAVI, que en resumen hace lo siguiente.

- partimos de $q(\theta_1, \ldots, \theta_d) = \prod_{i=1}^{d} q_i(\theta_i)$
- para $j = 1 \ldots, n$ actualizamos q_j cambiando a q_j^* como en (2.11)
- iteramos las actualizaciones anteriores hasta alcanzar convergencia

Claramente, si la familia $\mathcal{Q}$ es la familia de todas las distribuciones posibles con marginales independientes entonces la actualización (2.11) conduce a un nuevo elemento en $\mathcal{Q}$. En otros casos, tal como veremos en el siguiente capítulo, la actualización tiene que hacerse de forma que no nos salgamos de la familia $\mathcal{Q}$.

2.4.1. Inferencia variacional mediante CAVI para un modelo generativo simple

Se ha implementado una versión simplificada del algoritmo CAVI para ilustrar el funcionamiento del enfoque variacional en un modelo probabilístico generativo simulado $p(x, z)$. El código correspondiente puede consultarse en el Anexo **??**.

Descripción del modelo. El modelo considera una variable latente $z \in \mathbb{R}^m$ con distribución a priori gaussiana estándar $p(z) = \mathcal{N}(0, I)$, donde I es la matriz identidad. La variable observada $x \in \mathbb{R}^n$ se genera a partir de una distribución también gaussiana, cuya media depende linealmente de las variables latentes: $p(x \mid z) = \mathcal{N}(\sum_j z_j, I)$. Es decir, cada componente de x está centrado alrededor de la suma de los componentes de z, lo que introduce una dependencia global del dato observado respecto a todas las variables latentes.
Este modelo, aunque sencillo, presenta una estructura que permite aplicar inferencia variacional con actualización coordenada. En particular, asumimos que la aproximación variacional $q(z)$ factoriza completamente, es decir, $q(z) = \prod_{j=1}^{m} q_j(z_j)$. Esta es una suposición común en métodos variacionales, ya que simplifica el cálculo de las actualizaciones.

Implementación y procedimiento. El algoritmo comienza con una inicialización aleatoria de $q(z)$, y posteriormente actualiza cada componente de forma iterativa. Para cada coordenada z_j, se mantiene fija la estimación actual de los demás componentes z_{-j}, y se calcula una nueva distribución que maximiza el ELBO. En este caso, la actualización se hace evaluando la distribución conjunta $\log p(x, z)$ condicionada, lo que nos permite construir una nueva estimación de $q_j(z_j)$ dentro de la familia de distribuciones independientes asumida.

Resultados. Para ilustrar el funcionamiento del algoritmo, se simuló un vector de datos $x = (2{,}0,\ 3{,}0)$, y se ejecutó el algoritmo con $m = 3$ variables latentes. Como resultado, se obtuvo una trayectoria creciente del ELBO, lo que indica que el algoritmo converge hacia una

solución estable. La distribución variacional final $q(z)$ refleja un equilibrio entre la información contenida en los datos y la estructura probabilística del modelo.

Interpretación. El gráfico de la evolución del ELBO (Figura 2.3) muestra cómo el algoritmo mejora sucesivamente la calidad de la aproximación variacional. Por otro lado, el gráfico de la distribución final $q(z)$ (Figura 2.4) ilustra las probabilidades asignadas a cada componente, lo cual puede interpretarse como el resultado de ajustar las creencias iniciales en función de la observación de los datos.

Este experimento, aunque teórico, permite entender de forma concreta el mecanismo de inferencia variacional por coordenadas, así como su capacidad para aproximar distribuciones complejas en modelos probabilísticos latentes.

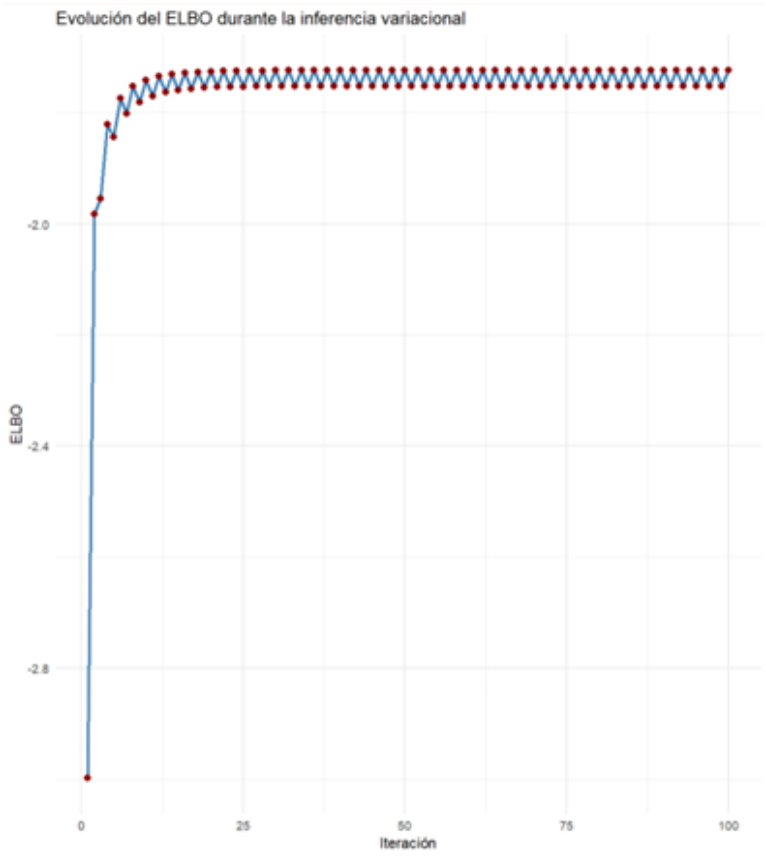

Figura 2.3: Evolución del ELBO

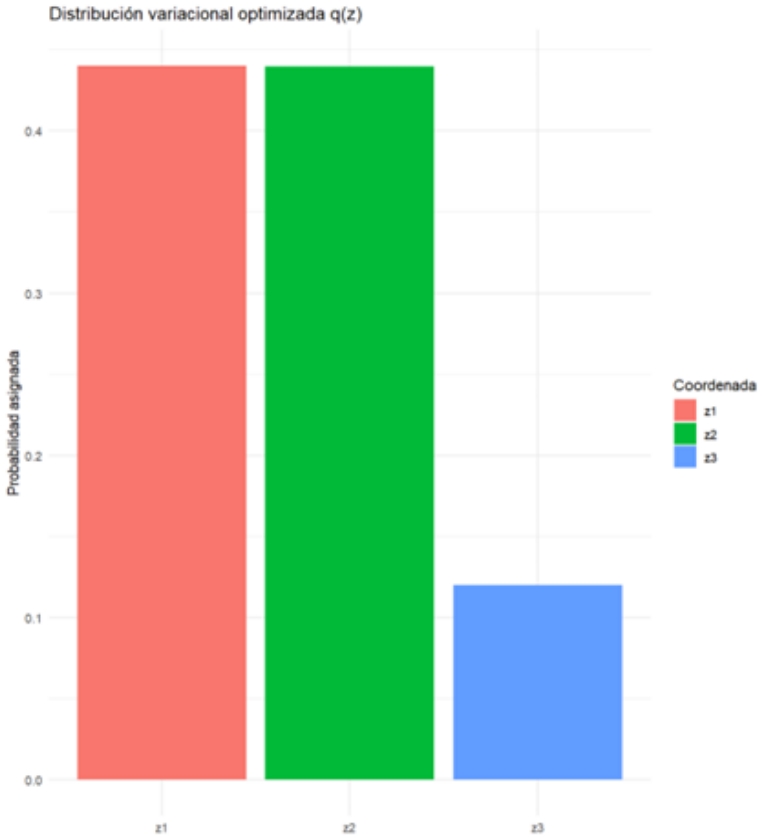

Figura 2.4: Probabilidades de cada componente

Capítulo 3

La Inferencia Variacional Gaussiana (IVG)

3.1. Definición y Motivación

La Inferencia Variacional Gaussiana (IVG) es una técnica específica dentro del marco de la inferencia variacional que restringe la familia de distribuciones aproximantes Q a las distribuciones normales multivariadas. En otras palabras, se aproxima la posterior $p(\theta|x)$ mediante una distribución gaussiana $q(z) = N(\mu, \Sigma)$, donde μ es el vector de medias y Σ la matriz de covarianzas.
Esta aproximación resulta especialmente útil por varias razones:

- Las distribuciones normales permiten un tratamiento matemático y computacional eficiente.
- El gradiente del ELBO se puede calcular analíticamente.
- Es posible aplicar técnicas de optimización avanzadas, como el descenso por gradiente natural.

3.2. Formulación Matemática

El objetivo de la IVG es encontrar los parámetros μ y Σ que maximizan la ELBO:

$$\text{ELBO}(q) = \mathbb{E}_q[\log p(x, \theta)] - \mathbb{E}_q[\log q(\theta)].$$

En la expresión anterior es útil saber que

- El primer término representa la esperanza del logaritmo del modelo conjunto, que impulsa $q(\theta)$ a ajustar bien los datos.
- El segundo término es la entropía direfencial de la distribución gaussiana, $\mathcal{H}$, que penaliza aproximaciones demasiado concentradas.

En el caso Gaussiano se tiene

$$\mathbb{E}_q[\log q(\theta)] = \mathcal{H}(q) = \frac{1}{2}\log((2\pi e)^d|\Sigma|),$$

donde d es la dimensión de θ y $|\Sigma|$ denota el determinante de Σ (ver [5]).

3.3. Versiones del Modelo

En inferencia variacional, cuando se aproxima la distribución posterior $p(\theta|x)$ mediante una distribución normal multivariada $q(\theta) = \mathcal{N}(\mu, \Sigma)$, se pueden imponer distintas restricciones sobre la matriz de covarianzas Σ según el compromiso entre expresividad y eficiencia computacional. Las tres variantes principales son:

- **Covarianza completa:** se permite que Σ sea una matriz simétrica y definida positiva general, lo que permite capturar todas las correlaciones entre los parámetros latentes. Esta opción es más expresiva, pero su optimización es costosa, ya que requiere estimar $d(d+1)/2$ parámetros en dimensión d.

- **Covarianza diagonal:** se restringe Σ a ser una matriz diagonal, es decir, $\Sigma = \text{diag}(\sigma_1^2, \ldots, \sigma_d^2)$. Esta opción sigue dentro de la familia gaussiana, pero impone independencia entre las variables latentes. Reduce significativamente la complejidad, ya que solo hay que estimar $2d$ parámetros: las medias μ y las varianzas marginales σ_j^2.

- **Estructura factorizada (mean-field):** consiste en asumir que la distribución variacional se factoriza como $q(\theta) = \prod_{j=1}^{d} q_j(\theta_j)$. Este enfoque no requiere que q_j sea gaussiana, pero si lo es y además se toma Σ diagonal, ambos enfoques coinciden. Por tanto, en el caso gaussiano, asumir una estructura factorizada equivale a suponer una covarianza diagonal. Sin embargo, la noción de mean-field es más general.

3.3.1. Algoritmo CAVI con covarianza diagonal

Cuando se utiliza una distribución gaussiana con covarianza diagonal en un enfoque mean-field, el algoritmo **CAVI (Coordinate Ascent Variational Inference)** puede expresarse de forma explícita.
En este caso, la distribución aproximante es:

$$q(\theta) = \mathcal{N}(\mu, \Sigma), \quad \text{con } \Sigma = \text{diag}(\sigma_1^2, \ldots, \sigma_d^2),$$

y se asume independencia entre coordenadas. El objetivo es maximizar la ELBO respecto a los parámetros variacionales μ y Σ. El algoritmo CAVI realiza actualizaciones iterativas de cada coordenada j, condicionando sobre las demás.
Las actualizaciones óptimas de cada parámetro (μ_j, σ_j^2) vienen dadas por:

$$\sigma_j^2 = \left(-\mathbb{E}_{q_{-j}} \left[\frac{\partial^2}{\partial \theta_j^2} \log p(x, \theta) \right] \right)^{-1}, \tag{3.1}$$

$$\mu_j = \sigma_j^2 \cdot \mathbb{E}_{q_{-j}} \left[\frac{\partial}{\partial \theta_j} \log p(x, \theta) \right]. \tag{3.2}$$

Estas expresiones se obtienen al maximizar la ELBO con respecto a $q_j(\theta_j)$ mientras se mantienen fijos los otros factores $q_{-j}(\theta_{-j})$. En la práctica, las esperanzas anteriores pueden calcularse de forma analítica o mediante muestreo dependiendo del modelo.

Este enfoque muestra cómo la independencia entre coordenadas impuesta por la covarianza diagonal permite descomponer la optimización en actualizaciones locales, lo que hace que el algoritmo sea computacionalmente eficiente.

3.4. Optimización mediante Gradiente Natural

La optimización de la ELBO puede realizarse mediante métodos de descenso por gradiente. En IVG, es especialmente efectivo el uso del gradiente natural, que ajusta la dirección de descenso teniendo en cuenta la geometría del espacio de distribuciones. Esta técnica mejora la convergencia y la estabilidad numérica, especialmente en entornos de alta dimensión o con funciones no convexas.

3.5. Ventajas de la IVG

- **Escalabilidad:** se adapta bien a grandes volúmenes de datos.
- **Flexibilidad:** permite introducir regularización mediante la entropía.
- **Eficiencia:** el uso de distribuciones gaussianas facilita el cómputo de expectativas.

3.6. Limitaciones

- **Baja fidelidad en distribuciones multimodales:** una única gaussiana no puede capturar múltiples modos.
- **Subestimación de incertidumbre:** tiende a producir varianzas menores que las reales, especialmente en la aproximación mean-field.
- **Dependencia del modelo:** en algunos contextos, la elección de la familia aproximante puede sesgar fuertemente los resultados.

3.7. Derivación de la ELBO paso a paso

La *Evidence Lower Bound* (ELBO) es la función objetivo fundamental en inferencia variacional. Su maximización permite aproximar la distribución posterior sin necesidad de calcular directamente la evidencia $p(x)$, que suele ser intratable.
A continuación, se presenta la derivación de la ELBO paso a paso a partir del teorema de Bayes y la definición de la divergencia de Kullback-Leibler.

Paso 1: Punto de partida. KL entre $q(\theta)$ y $p(\theta \mid x)$

El objetivo de la inferencia variacional es encontrar una distribución $q(\theta)$ dentro de una familia tractable $\mathcal{Q}$ que minimice:

$$\mathrm{KL}(q(\theta)\|p(\theta \mid x)) = \int q(\theta) \log \frac{q(\theta)}{p(\theta \mid x)}\, d\theta$$

Como esta divergencia es difícil de calcular directamente debido a la presencia de $p(\theta \mid x)$, aplicamos el teorema de Bayes:

$$p(\theta \mid x) = \frac{p(x, \theta)}{p(x)}$$

Entonces:

$$\mathrm{KL}(q(\theta)\|p(\theta \mid x)) = \int q(\theta) \log \frac{q(\theta)}{p(x, \theta)}\, d\theta + \log p(x)$$

Paso 2: Simplificación del logaritmo

Separando términos:

$$\mathrm{KL}(q(\theta)\|p(\theta \mid x)) = \int q(\theta) \log q(\theta)\, d\theta - \int q(\theta) \log p(x, \theta)\, d\theta + \log p(x)$$

Paso 3: Aislar $\log p(x)$

Reordenando:

$$\log p(x) = \mathrm{KL}(q(\theta)\|p(\theta \mid x)) + \underbrace{\mathbb{E}_{q(\theta)}[\log p(x, \theta)] - \mathbb{E}_{q(\theta)}[\log q(\theta)]}_{\mathrm{ELBO}(q)}$$

Por tanto:

$$\mathrm{ELBO}(q) = \mathbb{E}_{q(\theta)}[\log p(x, \theta)] - \mathbb{E}_{q(\theta)}[\log q(\theta)]$$

Paso 4: Interpretación de la ELBO

La ELBO tiene una interpretación clara:

- El término $\mathbb{E}_q[\log p(x, \theta)]$ mide qué tan bien $q(\theta)$ explica los datos y el modelo.
- El término $\mathbb{E}_q[\log q(\theta)]$ es la entropía negativa de q, que actúa como regularizador.

Al maximizar la ELBO, buscamos distribuciones q que:

1. Expliquen bien los datos.
2. No estén excesivamente concentradas (evitando sobreajuste).

Resumen

Término	**Interpretación**
$\mathbb{E}_q[\log p(x, \theta)]$	Ajuste al modelo y los datos
$\mathbb{E}_q[\log q(\theta)]$	Regularización vía entropía negativa
ELBO	Función a maximizar para encontrar q
$\log p(x)$	Cota superior de la ELBO

Cuadro 3.1: Interpretación de los términos de la ELBO

Capítulo 4

Aplicaciones de la Inferencia Variacional Gaussiana (IVG)

La Inferencia Variacional Gaussiana (IVG) es una técnica de inferencia bayesiana aproximada que ha ganado popularidad debido a su eficiencia computacional y escalabilidad. A diferencia de métodos como MCMC, que pueden ser costosos y lentos, IVG aproxima la distribución posterior mediante una familia tractable —habitualmente gaussianas— que se ajusta mediante optimización. En este capítulo se examinan varios modelos en los que la IVG resulta especialmente eficaz, explicando no solo su uso, sino también las razones estructurales y computacionales que la hacen adecuada.

4.1. Modelos de mezcla de gaussianas (GMM)

Los Gaussian Mixture Models (GMM) permiten modelar datos provenientes de subpoblaciones no observables directamente. Se asume que los datos han sido generados por una mezcla de distribuciones normales multivariadas:

$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x \mid \mu_k, \Sigma_k) \tag{4.1}$$

donde π_k son los pesos de mezcla y μ_k, Σ_k los parámetros de cada componente.

Desafío: La inferencia bayesiana exacta requiere estimar las asignaciones latentes y los parámetros de cada componente, lo cual es computacionalmente costoso con métodos como EM o MCMC, especialmente cuando el número de componentes o la dimensionalidad de los datos es alto.

Por qué IVG funciona: La estructura de mezcla permite que IVG modele las asignaciones latentes de forma tractable usando distribuciones categóricas, y que los parámetros se aproximen mediante gaussianas. Esto facilita inferencia paralela y escalable, ideal para problemas de clustering y segmentación en grandes conjuntos de datos.

Ejemplo práctico: Inferencia con IVG en una mezcla de dos gaussianas

Para ilustrar el uso de IVG, simulamos un conjunto de datos bidimensionales generados a partir de una mezcla de dos distribuciones normales. Cada componente tiene una media y varianza distintas, y las observaciones no incluyen las etiquetas de componente (latentes).
Se estima una mezcla de dos gaussianas usando Inferencia Variacional Gaussiana, aproximando la posterior sobre los parámetros μ_k, Σ_k y las asignaciones latentes z_n. La inferencia se realiza mediante optimización de la ELBO con el truco de reparametrización y el uso de distribuciones gaussianas para los parámetros.

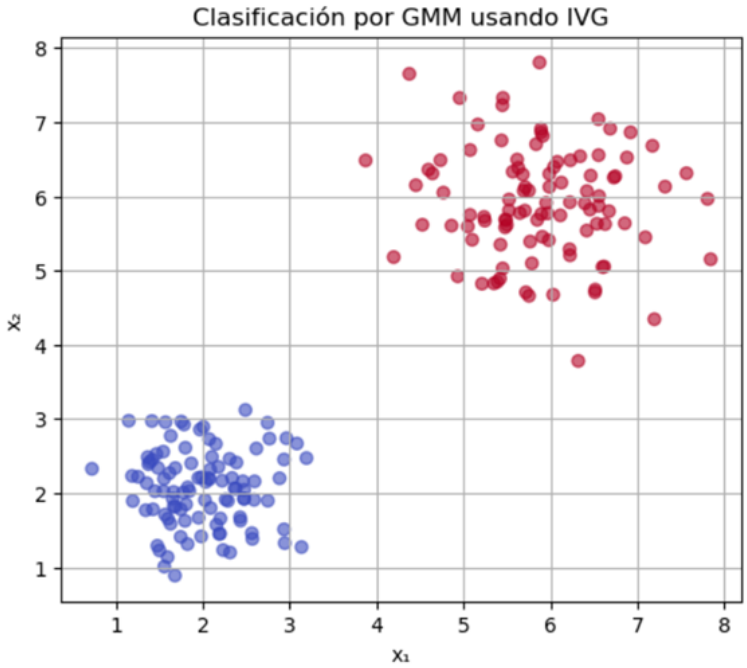

Figura 4.1: Clasificación variacional en una mezcla de dos gaussianas.

La Figura 4.2 muestra cómo IVG clasifica correctamente los puntos en dos grupos, asignando a cada observación una probabilidad posterior de pertenencia a cada componente. El modelo también estima la incertidumbre en los parámetros de cada componente (medias y covarianzas) mediante distribuciones gaussianas, lo cual no es posible con métodos como EM.
Este ejemplo demuestra que IVG no solo permite clasificación, sino también estimación bayesiana eficiente en modelos de mezcla, sin recurrir a muestreo MCMC ni asumir posteriori puntuales.

4.2. Latent Dirichlet Allocation (LDA)

LDA es un modelo generativo de temas para colecciones de texto. Se asume que cada documento está compuesto por una combinación de temas latentes, y cada tema está caracterizado por una distribución sobre palabras.

$$p(\theta_d, z_{d,n}, w_{d,n}) = p(\theta_d) \prod_{n=1}^{N_d} p(z_{d,n} \mid \theta_d)\, p(w_{d,n} \mid z_{d,n}, \beta) \tag{4.2}$$

Desafío: La inferencia exacta es intratable debido al gran número de variables latentes por documento y la estructura jerárquica del modelo.
Por qué IVG funciona: IVG, bajo una aproximación mean-field, descompone el problema de inferencia en factores independientes. Esta descomposición permite inferencia paralela documento por documento y habilita versiones online (como Online LDA) que funcionan incluso en colecciones con millones de textos.
Ejemplo: el algoritmo Online LDA (Hoffman et al., 2010) usa IVG para clasificar artículos de prensa en temas como política, deportes o economía, en tiempo real.

Visualización de la distribución de temas en documentos

El algoritmo Online LDA (Hoffman et al., 2010) usa IVG para clasificar artículos de prensa en temas como política, deportes o economía, en tiempo real.

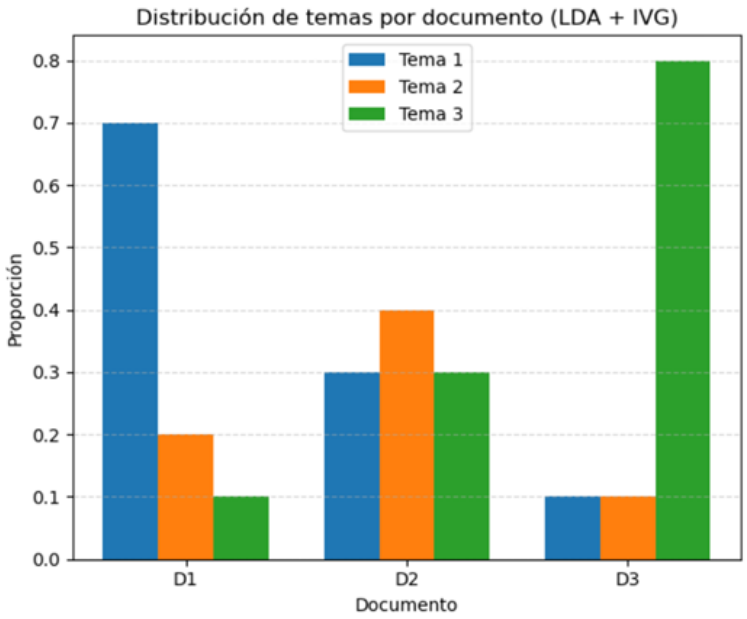

Figura 4.2: Distribución de temas en documentos según LDA.

En la Figura 4.2 cada barra representa la proporción de un tema en un documento específico. Por ejemplo, el documento 1 está compuesto mayoritariamente por el Tema 1 (70 %), mientras que el documento 3 está dominado por el Tema 3 (80 %). Esto ejemplifica cómo LDA con IVG permite descomponer documentos complejos en combinaciones interpretables de temas latentes.

4.3. Variational Autoencoders (VAEs)

Los VAEs combinan inferencia bayesiana con aprendizaje profundo para construir modelos generativos capaces de aprender representaciones latentes. Se define:

$$p(x, z) = p(z) \cdot p(x \mid z), \quad q(z \mid x) \approx p(z \mid x)$$

donde $q(z \mid x) = \mathcal{N}(\mu(x), \text{diag}(\sigma^2(x)))$ se entrena junto al modelo generativo.
La función de coste se basa en la ELBO.
Desafío: La posterior sobre las variables latentes es desconocida y depende de parámetros aprendibles, lo que impide usar inferencia clásica.
Por qué IVG funciona: Usando el truco de reparametrización:

$$z = \mu(x) + \sigma(x) \odot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I)$$

IVG permite entrenar el modelo mediante descenso por gradiente. Esta propiedad lo hace compatible con backpropagation, facilitando su integración en arquitecturas neuronales profundas para generación de imágenes, audio y otros.

Visualización de la representación latente aprendida por un VAE

El siguiente gráfico (Figura 4.3) muestra una representación bidimensional del espacio latente aprendido por un VAE a partir de datos simulados. Cada punto representa una muestra codificada en este espacio, coloreada según su clase latente simulada.
Se observan dos clusters claramente definidos, lo que ilustra cómo el VAE agrupa datos similares en regiones próximas del espacio latente. Esto demuestra que el modelo puede capturar patrones subyacentes y representar la variabilidad de manera continua y diferenciada.
Esta estructura en el espacio latente es fundamental para que el VAE pueda generar o reconstruir datos a partir de dichas representaciones, aprovechando la inferencia variacional y el truco de reparametrización para un entrenamiento eficiente.

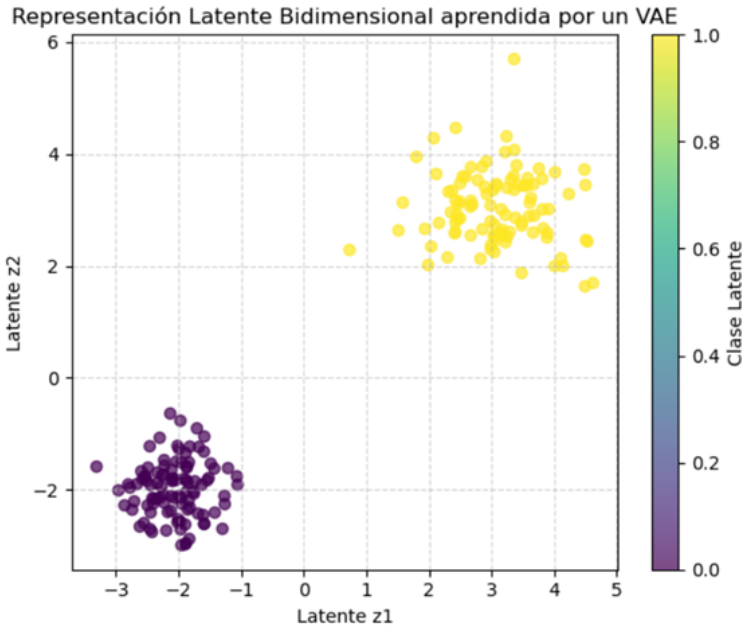

Figura 4.3: Representación latente aprendida por un VAE.

4.4. Regresión Logística Bayesiana

La regresión logística modela la probabilidad binaria como:

$$p(y_i = 1 \mid x_i, \beta) = \sigma(x_i^\top \beta), \quad \text{donde} \quad \beta \sim \mathcal{N}(0, \tau^2 I)$$

Desafío: La posterior sobre β no tiene forma cerrada, y métodos como Laplace pueden ser imprecisos en altas dimensiones.
Por qué IVG funciona: Aproximando $p(\beta \mid X, y)$ con una gaussiana $q(\beta) = \mathcal{N}(\mu, \Sigma)$ y usando el truco de reparametrización, IVG proporciona una inferencia eficiente y diferenciable, adecuada para clasificación con grandes volúmenes de datos y con incertidumbre cuantificada.
Ejemplo: Un ejemplo típico es la predicción médica basada en datos de pacientes, donde la rapidez y confiabilidad de la inferencia son esenciales para la toma de decisiones clínicas.

Visualización de una frontera de decisión y distribución de parámetros

El siguiente gráfico (Figura 4.4) simula una frontera de decisión para un problema binario en dos dimensiones, donde el modelo bayesiano estima no solo una frontera fija sino una distribución sobre las posibles fronteras, reflejando la incertidumbre en los parámetros β.

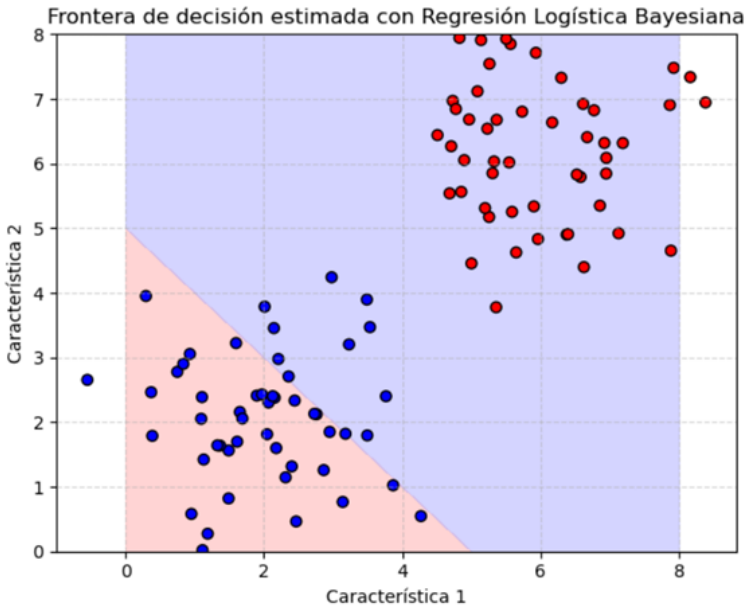

Figura 4.4: Frontera de decisión y distribución de parámetros en regresión logística bayesiana.

4.5. Modelos Jerárquicos en Biomedicina

Un ejemplo típico es:

$$\theta_i \sim \mathcal{N}(\mu, \sigma^2), \quad y_i \sim \text{Bernoulli}(\sigma(x_i^\top \theta_i))$$

Desafío: La presencia de niveles jerárquicos introduce múltiples dependencias latentes que dificultan la inferencia exacta, especialmente con miles de grupos (ej. pacientes, genes).

Por qué IVG funciona: IVG permite aproximar las distribuciones posteriores de efectos aleatorios θ_i de forma eficiente sin MCMC, lo que la hace ideal para estudios clínicos, genómica y educación, donde la estructura jerárquica es común.
Ejemplo: Por ejemplo, en análisis genómicos, cada gen o individuo puede ser considerado como un grupo latente distinto, permitiendo captar heterogeneidad biológica de manera efectiva.

Visualización de la variabilidad entre grupos

Este gráfico (Figura 4.5) simula la media y la dispersión de un parámetro θ_i estimado para diferentes grupos (por ejemplo, pacientes o genes), ilustrando cómo cada grupo tiene su propia distribución aproximada.

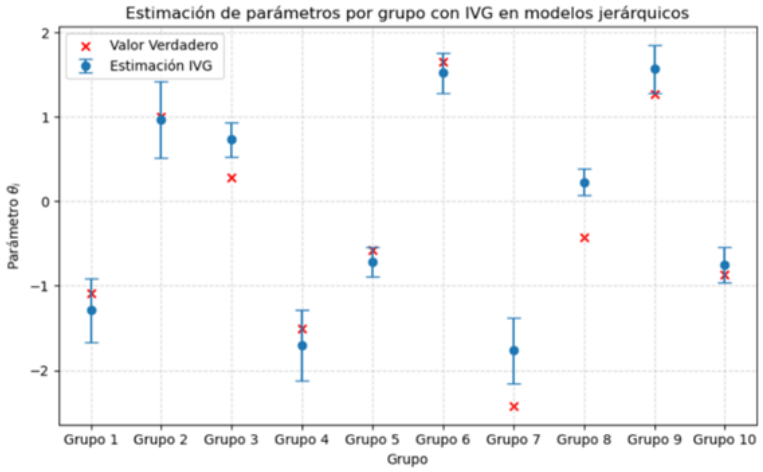

Figura 4.5: Variabilidad entre grupos estimada con IVG

En la Figura 4.5, cada punto azul representa la estimación media aproximada del parámetro latente θ_i para cada grupo, y las barras de error muestran la incertidumbre asociada estimada mediante IVG. Los cruces rojos indican los valores verdaderos simulados para referencia.
Este gráfico ejemplifica cómo los modelos jerárquicos permiten capturar diferencias específicas entre grupos, mientras que IVG facilita la estimación eficiente de dichas distribuciones sin necesidad de muestreos costosos. Esto es crucial en biomedicina para analizar variabilidad entre sujetos, genes o tratamientos con grandes volúmenes de datos.

4.6. Series Temporales y Modelos de Estado Latente

En modelos dinámicos como los filtros de Kalman o los modelos ocultos de Markov continuos (HMMs continuos), se modela una secuencia de estados latentes que evolucionan en el tiempo, condicionados por observaciones parciales y ruidosas.

$$x_t \sim p(x_t \mid x_{t-1}), \quad y_t \sim p(y_t \mid x_t)$$

Desafío: La dependencia temporal entre estados hace que la posterior sobre la trayectoria completa sea compleja y de alta dimensión.

Por qué IVG funciona: IVG puede modelar la trayectoria $z_{1:T}$ con una gaussiana estructurada o mediante redes neuronales recurrentes, permitiendo inferencia eficiente por gradiente. Es útil en seguimiento de objetos, análisis financiero y más.

Visualización de estados latentes estimados en series temporales

El siguiente gráfico (Figura 4.5) ilustra cómo IVG puede aproximar la trayectoria de un estado latente en una serie temporal, a partir de observaciones ruidosas.

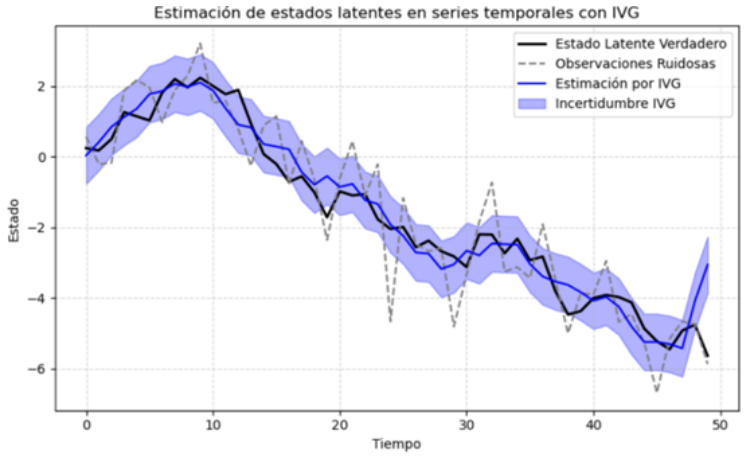

Figura 4.6: Trayectoria de un estado latente estimado con IVG.

En la Figura 4.5 la curva negra representa la trayectoria verdadera de un estado latente a lo largo del tiempo (por ejemplo, el estado de salud de un paciente). Las observaciones ruidosas (línea gris punteada) dificultan una estimación directa.
La línea azul muestra una estimación aproximada realizada por IVG, junto con una banda de incertidumbre (en azul claro) que refleja la distribución aproximada $q(zt)$ para cada instante t. Este enfoque permite capturar tanto la dinámica como la incertidumbre, sin recurrir a métodos como MCMC.

4.7. Sistemas de Recomendación

Los sistemas de recomendación basados en factorización matricial probabilística modelan las interacciones usuario-producto como:

$$r_{ij} \sim \mathcal{N}(u_i^\top v_j, \sigma^2)$$

donde u_i y v_j son vectores latentes de usuarios e ítems.
Desafío: La matriz de observaciones es dispersa y de gran escala, dificultando la inferencia exacta sobre todos los vectores latentes.
Por qué IVG funciona: IVG permite inferencia local y distribuida sobre cada u_i y v_j, lo que facilita la personalización, maneja la incertidumbre y escala a millones de usuarios y productos.

Visualización de vectores latentes y su incertidumbre

El siguiente gráfico (Figura 4.5) muestra la distribución estimada de los vectores latentes para usuarios y productos en un espacio bidimensional. Cada punto representa un usuario o producto, y las elipses reflejan la incertidumbre (varianza) aproximada de IVG.

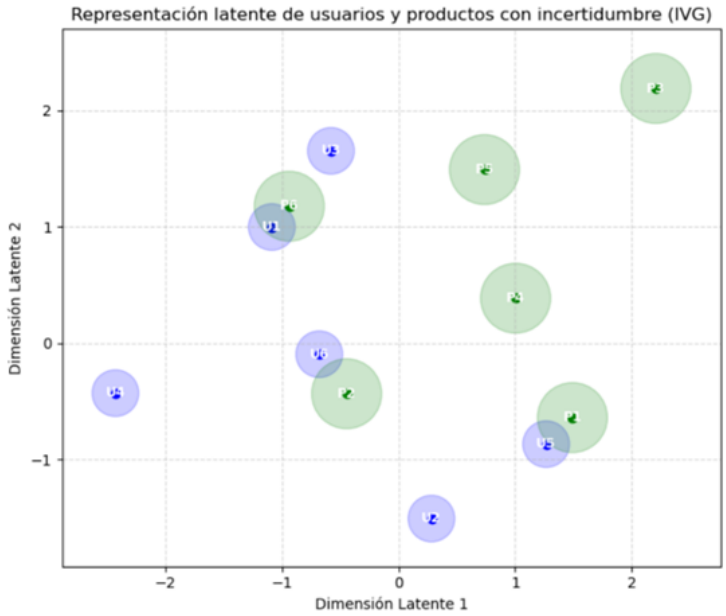

Figura 4.7: Vectores latentes e incertidumbre en un sistema de recomendación.

En la Figura 4.5 cada usuario (en azul) y producto (en verde) se representa mediante un vector latente en un espacio 2D. Las elipses indican la incertidumbre estimada mediante IVG sobre estos vectores latentes: áreas más grandes implican más incertidumbre. Esta información permite, por ejemplo, seleccionar productos con alta incertidumbre para mejorar la personalización en usuarios nuevos (exploración), o reforzar recomendaciones en zonas de alta confianza (explotación).

4.8. Resumen comparativo

Aplicación	**Desafío de inferencia**	**Ventaja de IVG**
GMM	Inferencia sobre múltiples asignaciones latentes	Posterior tractable y paralelizabilidad
LDA	Jerarquía y gran volumen de documentos	Mean-field escalable, inferencia online
VAE	Latentes dependientes de entrada	Reparametrización compatible con SGD
Regresión logística	Posterior no conjugada	Inferencia rápida y diferenciable sobre parámetros
Modelos jerárquicos	Estructura multinivel compleja	Estimación simultánea de niveles latentes
Series temporales	Dependencia entre estados latentes	Inferencia estructurada con redes o gaussianas
Recomendadores	Datos dispersos y masivos	Inferencia local eficiente sobre vectores latentes

Cuadro 4.1: Resumen de aplicaciones y fortalezas de IVG

Conclusión

La Inferencia Variacional Gaussiana se adapta de forma natural a modelos complejos donde la inferencia exacta es inviable, ya sea por la dimensionalidad, la jerarquía o la naturaleza dinámica de los datos. Su combinación de flexibilidad, velocidad y capacidad de integración con técnicas modernas de optimización la convierte en una herramienta clave en la inferencia bayesiana aplicada.

Capítulo 5

Optimización en Inferencia Variacional Gaussiana (IVG)

La optimización es un componente clave de la inferencia variacional. Dado que el objetivo es maximizar la *ELBO* con respecto a los parámetros de la distribución $q(\theta)$, se requieren herramientas eficientes, estables y escalables. En la Inferencia Variacional Gaussiana (IVG), la elección del algoritmo de optimización afecta directamente la calidad de la aproximación posterior [4].

5.1. Problema de optimización en IVG

En IVG, la distribución $q(\theta)$ se parametriza como una gaussiana multivariada $\mathcal{N}(\mu, \Sigma)$. Por tanto, el problema se reduce a:

$$\max_{\mu,\Sigma} \text{ELBO}(\mu, \Sigma)$$

donde:

- μ: vector de medias.
- Σ: matriz de covarianzas, que puede ser completa o diagonal según la variante utilizada.

La *ELBO* se evalúa como:

$$\mathcal{L}(\mu, \Sigma) = \mathbb{E}_{q(\theta)}[\log p(x, \theta)] - \mathbb{E}_{q(\theta)}[\log q(\theta)]$$

Este objetivo se optimiza mediante métodos de gradiente, enfrentando dos retos principales:

- La esperanza no suele tener forma cerrada.
- La matriz de covarianzas debe mantenerse semidefinida positiva.

5.2. Reparametrización de Monte Carlo

Para estimar la ELBO y sus gradientes, se emplea el truco de reparametrización propuesto por [9]:

$$\theta = \mu + L\epsilon, \quad \epsilon \sim \mathcal{N}(0, I)$$

donde L es la descomposición de Cholesky de Σ. Así, la esperanza se estima como:

$$\mathbb{E}_{q(\theta)}[\log p(x, \theta)] \approx \frac{1}{K} \sum_{k=1}^{K} \log p(x, \mu + L\epsilon^{(k)})$$

Este truco permite propagar el gradiente a través de las muestras, haciendo viable el uso de optimización por gradiente estocástico.

5.3. Descenso por Gradiente Natural

Una mejora importante sobre el gradiente clásico es el uso del *gradiente natural*, propuesto por [1], que tiene en cuenta la geometría del espacio de distribuciones probabilísticas:

$$\tilde{\nabla}_\lambda \mathcal{L} = F^{-1} \nabla_\lambda \mathcal{L}$$

donde F es la matriz de Fisher asociada a la familia $q(\theta)$. Esto mejora la convergencia y estabilidad numérica en espacios mal condicionados.

5.4. Optimización práctica con Adam

En contextos prácticos, se utilizan optimizadores adaptativos como Adam, introducido por [8], que combina momentum y escalado adaptativo del paso:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) \nabla \mathcal{L}$$
$$v_t = \beta_2 v_{t-1} + (1 - \beta_2)(\nabla \mathcal{L})^2$$

Este tipo de optimizador es útil en tareas de alta dimensionalidad con gradientes ruidosos.

5.5. Consideraciones numéricas

Mantenimiento de la positividad de Σ

Para garantizar que Σ sea semidefinida positiva, se utiliza [9, 11]:

- Una matriz diagonal con log-parametrización para mantener la positividad.
- La descomposición de Cholesky: $\Sigma = LL^\top$, optimizando directamente L.

Control de la varianza del estimador

La estimación por muestreo introduce varianza. Se puede mitigar con:

- Promediado sobre múltiples muestras.
- Técnicas de reducción de varianza como control variates o baselines.

5.6. Resumen de técnicas empleadas en IVG

Técnica	**Uso principal**
Reparametrización	Estimar gradientes diferenciables
Gradiente natural	Convergencia más rápida y estable
Adam	Optimización robusta con gradientes ruidosos
Cholesky	Garantía de positividad de Σ
Control de varianza	Estabilidad del entrenamiento

Cuadro 5.1: Resumen de técnicas utilizadas en IVG

Capítulo 6

Aplicación: Inferencia Variacional Gaussiana en un Modelo de Mezcla de Normales

Los modelos de mezcla de normales (GMM) son ampliamente utilizados para modelar datos que provienen de múltiples subpoblaciones. En este capítulo se describe cómo aplicar Inferencia Variacional Gaussiana (IVG) al problema de estimar los parámetros de una mezcla gaussiana, evitando métodos como EM o MCMC.

6.1. Formulación del modelo

Consideramos un modelo de mezcla de K componentes gaussianos:

$$p(x \mid \theta) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x \mid \mu_k, \Sigma_k)$$

donde $\theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}$ son los parámetros del modelo, y las variables latentes $z_n \in \{1, \ldots, K\}$ indican el componente del que proviene cada observación x_n.
El objetivo es inferir una distribución posterior sobre θ y sobre las asignaciones $z_1, \ldots, z_N$, dado un conjunto de datos $\{x_n\}_{n=1}^{N}$.

6.2. Inferencia variacional

La distribución posterior $p(\theta, \mathbf{z} \mid \mathbf{x})$ es intractable. Se introduce una distribución variacional:

$$q(\theta, \mathbf{z}) = q(\theta) \prod_{n=1}^{N} q(z_n)$$

donde $q(\theta)$ se aproxima mediante una distribución gaussiana multivariada (IVG). Para los z_n, se usa una distribución categórica con parámetros variacionales.
El objetivo es maximizar la ELBO:

$$\mathcal{L}(q) = \mathbb{E}_q[\log p(\mathbf{x}, \theta, \mathbf{z})] - \mathbb{E}_q[\log q(\theta, \mathbf{z})]$$

6.3. Implementación práctica

En la práctica, se usan los siguientes elementos:

- **Parametrización gaussiana** de $q(\theta) = \mathcal{N}(\mu, \Sigma)$ con reparametrización para muestreo eficiente.
- **Estimación estocástica de la ELBO** mediante muestreo Monte Carlo.
- **Optimización con Adam** y control de la varianza en gradientes.
- **Visualización**: se puede comparar la clasificación de puntos obtenida por IVG con la obtenida por EM.

6.4. Ejemplo simulado

Ejemplo práctico: Inferencia con IVG en una mezcla de tres gaussianas

Para ilustrar el procedimiento, se considera un conjunto de datos bidimensionales generado a partir de tres componentes gaussianas con diferentes medias y covarianzas. Aplicando Inferencia Variacional Gaussiana (IVG) mediante un modelo de mezcla, se obtiene una aproximación a la distribución posterior de los parámetros de cada componente y de las asignaciones latentes $q(z_n)$.

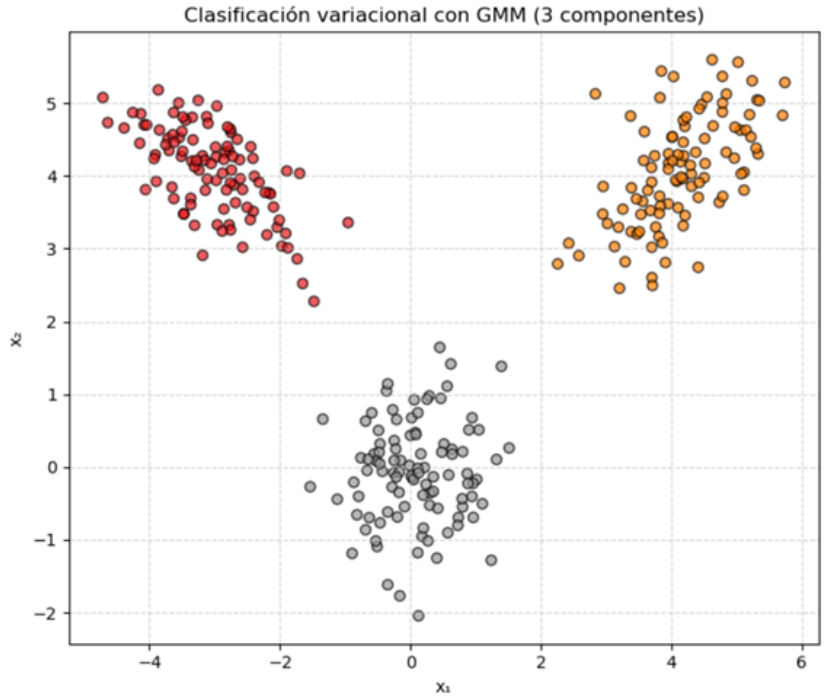

Figura 6.1: Clasificación variacional en una mezcla de tres gaussianas con IVG.

La Figura 6.1 muestra la clasificación resultante. IVG identifica correctamente los tres clústeres, incluso cuando los datos presentan cierta superposición entre grupos. La aproximación variacional proporciona no solo una asignación de clases, sino también una estimación de la incertidumbre sobre los parámetros del modelo, lo cual es fundamental para tareas de análisis exploratorio o segmentación probabilística.

6.5. Discusión

Este ejemplo muestra que IVG puede aplicarse a modelos clásicos como las mezclas de gaussianas, proporcionando estimaciones eficientes sin necesidad de muestreo MCMC. La aproximación gaussiana resulta suficiente para capturar la posterior de los parámetros en este caso simple, y el enfoque es escalable a problemas de mayor dimensión.

Capítulo 7

Conclusión

En este Trabajo Fin de Grado se ha abordado la Inferencia Variacional Gaussiana (IVG) como una técnica eficiente para aproximar distribuciones posteriores en modelos probabilísticos complejos. A lo largo del trabajo se ha revisado el marco teórico de la inferencia variacional, se han detallado las propiedades particulares de la aproximación gaussiana y se han expuesto los principales métodos de optimización que permiten llevar a cabo esta inferencia de forma práctica.

Uno de los aspectos clave ha sido el estudio de la función objetivo ELBO, cuya maximización permite ajustar los parámetros de la distribución variacional. En este contexto, técnicas como el truco de reparametrización, el uso del gradiente natural y la aplicación de optimizadores como Adam resultan fundamentales para obtener resultados precisos y estables, especialmente en entornos de alta dimensionalidad.

Además, se han revisado consideraciones numéricas importantes, como la necesidad de garantizar la positividad de la matriz de covarianzas y de controlar la varianza en los estimadores Monte Carlo. Estas cuestiones, aunque técnicas, tienen un impacto directo en la eficacia y robustez de los métodos de inferencia utilizados.

Por último, se han mostrado aplicaciones de IVG en diferentes contextos, lo que ha permitido ilustrar su flexibilidad y potencial en problemas reales de aprendizaje automático y estadística bayesiana.

Como trabajo futuro, sería interesante explorar aproximaciones más expresivas que superen las limitaciones de la gaussiana multivariada, como los flujos normalizadores (normalizing flows), así como analizar la integración de IVG en arquitecturas más complejas como los modelos generativos profundos.

En conjunto, este trabajo ha contribuido a consolidar una base sólida sobre inferencia variacional moderna, al mismo tiempo que ha proporcionado herramientas y reflexiones útiles para su aplicación práctica en problemas actuales de modelado probabilístico.

Bibliografía

[1] Shun-ichi Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10 (2):251–276, 1998.

[2] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006. URL https://www.academia.edu/17851990/Bishop_Pattern_Recognition_and_Machine_Learning.

[3] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003. URL https://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf.

[4] David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational inference: A review. *Journal of the American Statistical Association*, 112(518):859–877, 2017. doi: 10.1080/01621459.2017.1285773. URL https://arxiv.org/pdf/1601.00670.

[5] Thomas M. Cover and Joy A. Thomas. *Elements of information theory*. Wiley-Interscience [John Wiley & Sons], Hoboken, NJ, second edition, 2006. ISBN 978-0-471-24195-9; 0-471-24195-4. URL https://cs-114.org/wp-content/uploads/2015/01/Elements_of_Information_Theory_Elements.pdf.

[6] Andrew Gelman, John B. Carlin, Hal S. Stern, David B. Dunson, Aki Vehtari, and Donald B. Rubin. *Bayesian Data Analysis*. Chapman and Hall/CRC, 3rd edition, 2013. URL https://sites.stat.columbia.edu/gelman/book/BDA3.pdf.

[7] Matthew D. Hoffman, David M. Blei, Chong Wang, and John Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14(1):1303–1347, 2013. URL https://jmlr.org/papers/volume14/hoffman13a/hoffman13a.pdf.

[8] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[9] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2014. URL https://arxiv.org/abs/1312.6114.

[10] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012. URL https://www.academia.edu/35856835/Machine_Learning_A_Probabilistic_Perspective.

[11] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. *arXiv preprint arXiv:1505.05770*, 2015. URL `https://arxiv.org/abs/1505.05770`.

Anexo: Código fuente para la generación de imágenes

Código en python para la Figura 2.1

```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import beta

theta = np.linspace(0, 1, 100)
prior = np.ones_like(theta)  # Distribucion uniforme
posterior = beta.pdf(theta, 9, 3)  # Distribucion Beta(9, 3)

plt.figure(figsize=(8, 5))
plt.plot(theta, prior, label='Distribucion a priori (Uniforme)',
    linestyle='--')
plt.plot(theta, posterior, label='Distribucion a posteriori (Beta(9,
    3))', color='orange')
plt.title('Actualizacion Bayesiana del Sesgo de una Moneda')
plt.xlabel(r'$\theta$')
plt.ylabel('Densidad')
plt.legend()
plt.grid(True)
plt.show()
```

Listing 7.1: Comparación entre distribución a priori uniforme y posterior Beta(9,3)

Código en R para la Figura 2.2

```r
library(ggplot2)

# Crear un rango de valores que representen diferentes elecciones de
    q
q_vals <- seq(0, 1, length.out = 100)

```

```r
# Supongamos log p(x) es constante
log_px <- 3

# ELBO es siempre menor o igual a log p(x), y se maximiza cuando KL = 0
# Vamos a modelarla como una funcion arbitraria creciente
elbo <- log_px - (1 - q_vals)^2 * 2  # concava, maxima en q = 1

# Data frame para graficar
df <- data.frame(
  q_choice = q_vals,
  ELBO = elbo,
  log_px = rep(log_px, length(q_vals))
)

# Graficar
ggplot(df, aes(x = q_choice)) +
  geom_line(aes(y = ELBO), color = "blue", size = 1.3) +
  geom_line(aes(y = log_px), color = "red", linetype = "dashed", size = 1.2) +
  annotate("text", x = 0.2, y = log_px + 0.2, label = "log p(x)", color = "red", size = 5) +
  annotate("text", x = 0.8, y = min(elbo) + 0.5, label = "ELBO(q)", color = "blue", size = 5) +
  labs(
    title = "La ELBO como cota inferior de log p(x)",
    x = "Eleccion de q(θ)",
    y = "Valor"
  ) +
  theme_minimal(base_size = 14)
```

Listing 7.2: Visualización de la ELBO como cota inferior de $\log p(x)$

Código en R: CAVI para un modelo generativo simple (Figuras 2.3 y 2.4)

```r
# CAVI en R: Inferencia Variacional Coordinada para un modelo generativo simple

library(dplyr)
library(ggplot2)

# ----------------------------
```

```r
# Logaritmo de la distribucion conjunta log p(x, z)
# ----------------------------
log_joint <- function(z, x) {
  return(-0.5 * sum(z^2) - 0.5 * sum((x - sum(z))^2))
}

# ----------------------------
# Inicializacion aleatoria de q(z)
# ----------------------------
initialize_q <- function(m) {
  return(rnorm(m, 0, 1))  # Inicializa con N(0,1)
}

# ----------------------------
# Actualizacion de q(z) por coordenadas (Coordinate Ascent)
# ----------------------------
update_q <- function(q, x) {
  m <- length(q)
  new_q <- q
  for (j in 1:m) {
    z_minus_j <- q[-j]
    new_q[j] <- exp(log_joint(z_minus_j, x))  # Estimacion
       condicional
  }
  return(new_q / sum(new_q))  # Normalizacion para asegurar que q es
     valida
}

# ----------------------------
# Calculo de ELBO
# ----------------------------
compute_ELBO <- function(q, x) {
  entropy <- -sum(q * log(q + 1e-10))  # para evitar log(0)
  return(log_joint(q, x) + entropy)
}

# ----------------------------
# Algoritmo principal de inferencia variacional
# ----------------------------
variational_inference <- function(x, m = 3, tol = 1e-3, max_iters =
   100) {
  q <- initialize_q(m)
  elbo_values <- numeric()

```

```r
  for (i in 1:max_iters) {
    q <- update_q(q, x)
    elbo <- compute_ELBO(q, x)
    elbo_values <- c(elbo_values, elbo)

    if (length(elbo_values) > 1 && abs(diff(tail(elbo_values, 2))) < tol) {
      break
    }
  }

  return(list(q_opt = q, elbo_traj = elbo_values))
}

# ----------------------------
# Datos simulados y ejecucion del algoritmo
# ----------------------------
x <- c(2.0, 3.0)
result <- variational_inference(x)

# ----------------------------
# Mostrar resultados
# ----------------------------
cat("Vector q(z) optimizado:\n")
print(round(result$q_opt, 4))

cat("\nValor final de ELBO:", round(tail(result$elbo_traj, 1), 4), "\n")

# ----------------------------
# Graficar trayectoria del ELBO
# ----------------------------
df_elbo <- data.frame(iter = seq_along(result$elbo_traj), ELBO = result$elbo_traj)

ggplot(df_elbo, aes(x = iter, y = ELBO)) +
  geom_line(color = "steelblue", size = 1) +
  geom_point(color = "darkred", size = 2) +
  labs(title = "Evolucion del ELBO durante la inferencia variacional",
       x = "Iteracion", y = "ELBO") +
  theme_minimal()

# ----------------------------
```

```r
# Graficar distribucion final q(z)
# ----------------------------
df_q <- data.frame(Coordenada = paste0("z", 1:length(result$q_opt)),
                   Valor = result$q_opt)

ggplot(df_q, aes(x = Coordenada, y = Valor, fill = Coordenada)) +
  geom_bar(stat = "identity") +
  labs(title = "Distribucion variacional optimizada q(z)", y = "
     Probabilidad asignada", x = "") +
  theme_minimal()
```

Listing 7.3: CAVI en R: Inferencia Variacional Coordinada

Código en python para la Figura 4.1

```python
import numpy as np
import matplotlib.pyplot as plt
from sklearn.mixture import GaussianMixture

# Simulacion de datos de dos grupos
np.random.seed(0)
X1 = np.random.normal(loc=[2, 2], scale=0.5, size=(100, 2))
X2 = np.random.normal(loc=[6, 6], scale=0.8, size=(100, 2))
X = np.vstack([X1, X2])

# Aplicar GMM
gmm = GaussianMixture(n_components=2).fit(X)
labels = gmm.predict(X)

# Grafico
plt.figure(figsize=(6, 5))
plt.scatter(X[:, 0], X[:, 1], c=labels, cmap='coolwarm', alpha=0.6)
plt.title("Clasificacion por GMM usando IVG")
plt.xlabel("x1")
plt.ylabel("x2")
plt.grid(True)
plt.show()
```

Listing 7.4: Clasificación con Gaussian Mixture Models simulando IVG

Código en python para la Figura 4.2

```python
import matplotlib.pyplot as plt
import numpy as np

# Simulacion de distribucion de temas en 3 documentos
doc_topics = np.array([
    [0.7, 0.2, 0.1],
    [0.3, 0.4, 0.3],
    [0.1, 0.1, 0.8]
])

bar_width = 0.25
indices = np.arange(doc_topics.shape[0])

# Graficar proporcion por tema
plt.figure(figsize=(6, 5))
for i in range(doc_topics.shape[1]):
    plt.bar(indices + i * bar_width, doc_topics[:, i], width=bar_
        width, label=f"Tema {i+1}")

plt.title("Distribucion de temas por documento (LDA + IVG)")
plt.xlabel("Documento")
plt.ylabel("Proporcion")
plt.xticks(indices + bar_width, [f"D{i+1}" for i in range(3)])
plt.legend()
plt.grid(True, axis='y', linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()
```

Listing 7.5: Visualización de distribución de temas por documento (LDA + IVG)

Código en python para la Figura 4.3

```python
import matplotlib.pyplot as plt
import numpy as np

# Simulacion de puntos en espacio latente 2D para un VAE
np.random.seed(42)
# Dos clusters simulando dos clases latentes
cluster1 = np.random.normal(loc=[-2, -2], scale=0.5, size=(100, 2))
cluster2 = np.random.normal(loc=[3, 3], scale=0.7, size=(100, 2))

latent_points = np.vstack([cluster1, cluster2])
labels = np.array([0]*100 + [1]*100)
```

```
12
13 # Graficar representacion latente
14 plt.figure(figsize=(6,5))
15 plt.scatter(latent_points[:,0], latent_points[:,1], c=labels, cmap='
   viridis', alpha=0.7)
16 plt.colorbar(label='Clase Latente')
17 plt.title('Representacion Latente Bidimensional aprendida por un VAE'
   )
18 plt.xlabel('Latente z1')
19 plt.ylabel('Latente z2')
20 plt.grid(True, linestyle='--', alpha=0.5)
21 plt.tight_layout()
22 plt.show()
```

Listing 7.6: Representacion latente 2D simulada para un VAE

Código en python para la Figura 4.4

```
1  import numpy as np
2  import matplotlib.pyplot as plt
3
4  # Simulacion de datos para clasificacion binaria
5  np.random.seed(0)
6  N = 100
7  X = np.vstack((
8      np.random.normal(loc=[2, 2], scale=1, size=(N//2, 2)),
9      np.random.normal(loc=[6, 6], scale=1, size=(N//2, 2))
10 ))
11 y = np.array([0]*(N//2) + [1]*(N//2))
12
13 # Parametros "verdaderos" (simulados)
14 beta_mean = np.array([-5, 1, 1])  # intercepto y coeficientes
15
16 # Generamos un grid para visualizar la frontera de decision
17 xx, yy = np.meshgrid(np.linspace(0, 8, 100), np.linspace(0, 8, 100))
18 grid = np.c_[np.ones(xx.size), xx.ravel(), yy.ravel()]
19
20 # Funcion sigmoide
21 def sigmoid(z):
22     return 1 / (1 + np.exp(-z))
23
24 # Probabilidades con beta promedio
25 probs = sigmoid(grid @ beta_mean).reshape(xx.shape)
26
```

```
# Graficar datos y frontera de decision
plt.figure(figsize=(6,5))
plt.contourf(xx, yy, probs, levels=[0,0.5,1], colors=['#FFAAAA','#AAAAFF'], alpha=0.5)
plt.scatter(X[:,0], X[:,1], c=y, cmap='bwr', edgecolors='k')
plt.title('Frontera de decision estimada con Regresion Logistica Bayesiana')
plt.xlabel('Caracteritica 1')
plt.ylabel('Caracteristica 2')
plt.grid(True, linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()
```

Listing 7.7: Clasificacion binaria con regresion logistica bayesiana simulada

Código en python para la Figura 4.5

```
import numpy as np
import matplotlib.pyplot as plt

np.random.seed(123)

# Numero de grupos (ej. pacientes o genes)
n_groups = 10

# Media global y desviacion estandar de la distribucion jerarquica
global_mu = 0.0
global_sigma = 1.0

# Simulacion de parametros verdaderos por grupo (latentes)
true_thetas = np.random.normal(global_mu, global_sigma, n_groups)

# Estimaciones aproximadas de media y desviacion (IVG)
estimated_mus = true_thetas + np.random.normal(0, 0.3, n_groups)
estimated_sigmas = np.abs(np.random.normal(0.3, 0.1, n_groups))

# Visualizacion: medias y barras de incertidumbre por grupo
plt.figure(figsize=(8,5))
plt.errorbar(range(n_groups), estimated_mus, yerr=estimated_sigmas, fmt='o', capsize=5, label='Estimacion IVG')
plt.scatter(range(n_groups), true_thetas, color='red', marker='x', label='Valor Verdadero')
plt.xticks(range(n_groups), [f'Grupo {i+1}' for i in range(n_groups)])
```

```
plt.xlabel('Grupo')
plt.ylabel(r'Parametro $\theta_i$')
plt.title('Estimacion de parametros por grupo con IVG en modelos
    jerarquicos')
plt.legend()
plt.grid(True, linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()
```

Listing 7.8: Estimacion de parámetros jerarquicos con IVG

Código en python para la Figura 4.6

```
import numpy as np
import matplotlib.pyplot as plt

np.random.seed(42)
T = 50  # numero de pasos de tiempo

# Generar una trayectoria latente (por ejemplo, estado de salud de un
    paciente)
true_state = np.cumsum(np.random.normal(0, 0.5, T))

# Observaciones ruidosas del estado latente
observed = true_state + np.random.normal(0, 1.0, T)

# Estimacion aproximada con IVG (simulada como una version suavizada
    + incertidumbre)
estimated_mean = np.convolve(observed, np.ones(5)/5, mode='same')  #
    Suavizado
estimated_std = np.full(T, 0.8)  # Supongamos incertidumbre constante

# Graficar
plt.figure(figsize=(8,5))
plt.plot(true_state, label='Estado Latente Verdadero', color='black',
    linewidth=2)
plt.plot(observed, label='Observaciones Ruidosas', linestyle='--',
    color='gray')
plt.plot(estimated_mean, label='Estimacion por IVG', color='blue')
plt.fill_between(np.arange(T),
                 estimated_mean - estimated_std,
                 estimated_mean + estimated_std,
                 alpha=0.3, color='blue', label='Incertidumbre IVG')
plt.xlabel('Tiempo')
```

```
plt.ylabel('Estado')
plt.title('Estimacion de estados latentes en series temporales con IVG')
plt.legend()
plt.grid(True, linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()
```

Listing 7.9: Estimacion de estados latentes con IVG en series temporales

Código en python para la Figura 4.7

```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.patches import Ellipse

np.random.seed(123)

# Simulamos 6 usuarios y 6 productos en espacio latente 2D
n = 6
user_means = np.random.randn(n, 2)
product_means = np.random.randn(n, 2)
user_cov = 0.2 * np.ones((n, 2))  # desviaciones estandar simuladas
product_cov = 0.3 * np.ones((n, 2))

fig, ax = plt.subplots(figsize=(7, 6))

# Dibujar usuarios
for i in range(n):
    ax.scatter(*user_means[i], color='blue')
    ell = Ellipse(xy=user_means[i], width=2*user_cov[i,0], height=2*user_cov[i,1],
                  edgecolor='blue', facecolor='blue', alpha=0.2)
    ax.add_patch(ell)
    ax.text(*user_means[i], f'U{i+1}', fontsize=9, ha='center', va='center', color='white', weight='bold')

# Dibujar productos
for i in range(n):
    ax.scatter(*product_means[i], color='green')
    ell = Ellipse(xy=product_means[i], width=2*product_cov[i,0], height=2*product_cov[i,1],
                  edgecolor='green', facecolor='green', alpha=0.2)
    ax.add_patch(ell)
```

```
    ax.text(*product_means[i], f'P{i+1}', fontsize=9, ha='center', va
       ='center', color='white', weight='bold')

ax.set_title('Representacion latente de usuarios y productos con
   incertidumbre (IVG)')
ax.set_xlabel('Dimension Latente 1')
ax.set_ylabel('Dimension Latente 2')
ax.grid(True, linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()
```

Listing 7.10: Espacio latente de usuarios y productos con IVG

Código en python para la Figura 6.1

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.mixture import GaussianMixture

# Semilla
np.random.seed(123)

# Generar datos de tres gaussianas con distinta forma
X1 = np.random.multivariate_normal(mean=[0, 0], cov=[[0.4, 0], [0,
   0.4]], size=100)
X2 = np.random.multivariate_normal(mean=[4, 4], cov=[[0.6, 0.3],
   [0.3, 0.6]], size=100)
X3 = np.random.multivariate_normal(mean=[-3, 4], cov=[[0.5, -0.2],
   [-0.2, 0.3]], size=100)
X = np.vstack((X1, X2, X3))

# GMM como aproximaci n variacional
gmm = GaussianMixture(n_components=3, covariance_type='full', random_
   state=123)
gmm.fit(X)
labels = gmm.predict(X)

# Visualizaci n
plt.figure(figsize=(7, 6))
plt.scatter(X[:, 0], X[:, 1], c=labels, cmap='Set1', edgecolors='k',
   alpha=0.7)
plt.title("Clasificaci n variacional con GMM (3 componentes)")
plt.xlabel(" x  ")
plt.ylabel(" x  ")
```

```
plt.grid(True, linestyle='--', alpha=0.5)
plt.tight_layout()

# Guardar imagen
plt.savefig("gmm_ivg_three_components.png", dpi=300)
plt.show()
```

Listing 7.11: Clasificación variacional con GMM de tres componentes