Análisis Matemático

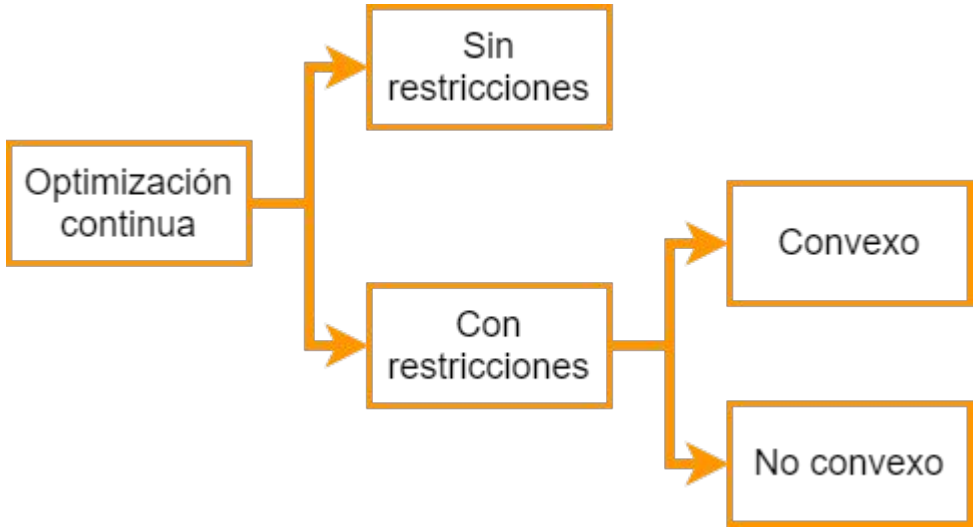
Clase 6

Optimización

Optimización

Dado que los algoritmos de aprendizaje automático se implementan en una computadora, las formulaciones matemáticas se expresan como métodos de optimización numérica.

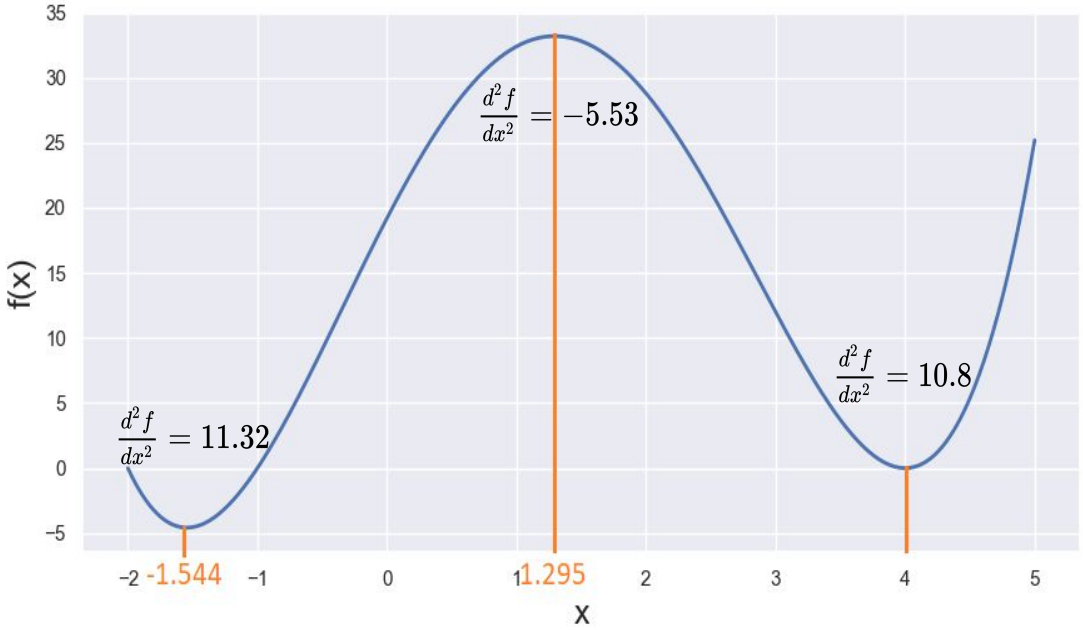
Dada una función objetivo, la búsqueda de los mejores parámetros se realiza mediante algoritmos de optimización.



Optimización sin restricciones

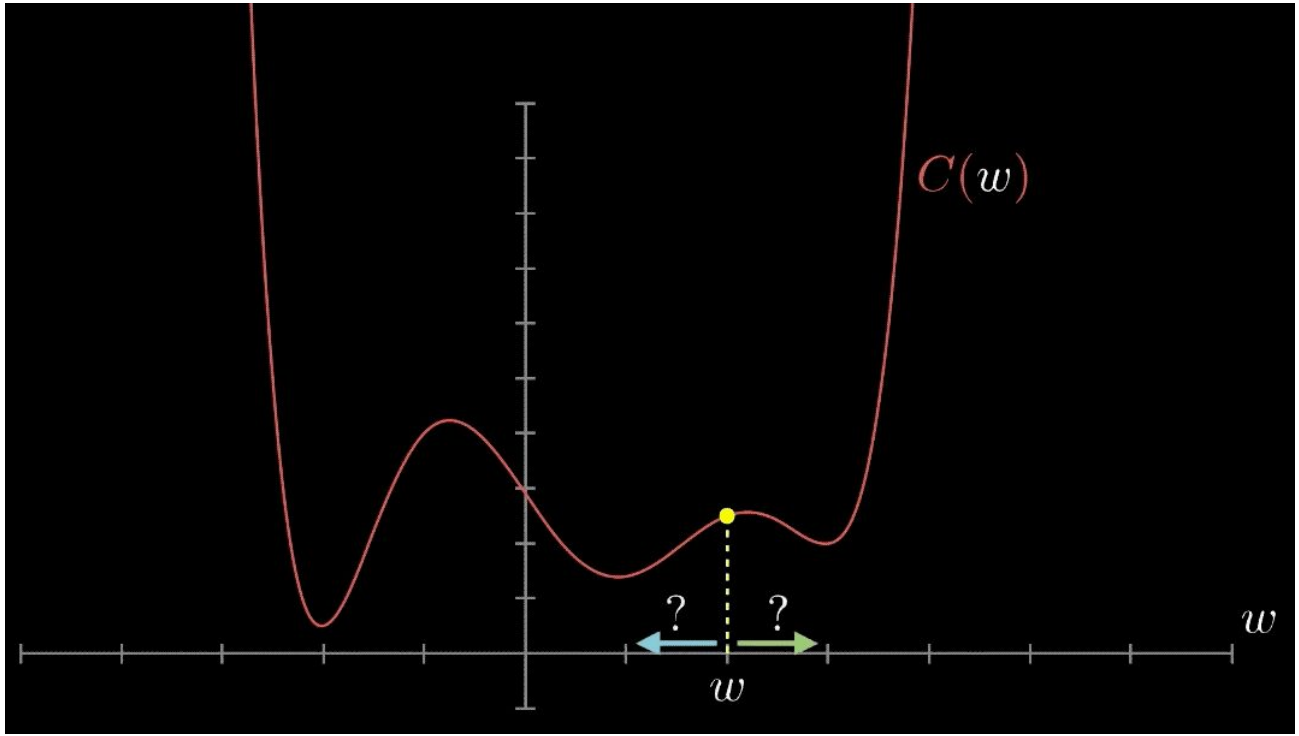
Gradiente Descendiente

Motivación



En este caso, como es un polinomio sencillo (grado 4) se puede calcular a mano las derivadas para analizar máximos mínimos, pero ¿qué hago en el caso general?

Motivación



Fuente

La idea es moverse en dirección opuesta al gradiente.
Algunos temas sin resolver:

- ¿Cuánto me muevo? (step size)
- ¿Qué pasa si no inicializo mal el algoritmo?

Gradiente descendiente

Optimización usando gradiente descendiente

Consideremos el problema de hallar el mínimo de una función a valores reales $f : \mathbb{R}^n \to \mathbb{R}$:

$$\min_x f(x)$$

Asumimos que f es diferenciable, y que no podemos hallar la solución cerrada de forma analítica.

Para hallar un mínimo (local) usando gradiente descendiente, vamos tomando pasos proporcionales a ∇_f.

Optimización usando gradiente descendiente

Si comenzamos desde una semilla x_0, y para $\gamma_n > 0$ (pequeño),

$$x_{n+1} = x_n - \gamma_n \nabla_f (x_n)^T$$

de forma que $f(x_{n+1}) \leq f(x_n) \leq \ldots \leq f(x_0)$.

Para una sucesión de pasos γ_i apropiada , la secuencia $f(x_0) \geq f(x_1) \geq \ldots$ converge a un mínimo local.

Obs: Puede resultar lento para converger al mínimo. También es crucial la elección de los γ_i

Gradiente descendiente: sobre la inicialización

Obs: El mínimo al que converge depende del punto de partida x_0

Fuente

Gradiente descendiente: tamaño del paso

Obs: Si el paso γ es muy pequeño, el algoritmo puede tardar mucho en converger. Por el contrario, si γ es muy grande el algoritmo puede no converger.

Los métodos adaptativos van modificando el valor de γ_i en cada paso del algoritmo. Dos simples heurísticas:

- Si $f(x_n) = f(x_{n-1} - \gamma \nabla_f (x_{n-1})^T) > f(x_{n-1})$ volvé a x_{n-1} y reducí γ
- Si $f(x_n) = f(x_{n-1} - \gamma \nabla_f (x_{n-1})^T) < f(x_{n-1})$ intentá aumentar γ.

Gradiente descendiente con momentum

El método de gradiente descendiente con momentum le da un poco de "memoria" al gradiente para acelerar la convergencia.

En cada paso, se agrega un término a la actualización que recuerda que ocurrió en la iteración anterior:

$$x_{i+1} = x_i - \gamma_i \nabla_f (x_i)^T + \alpha \Delta x_i$$
$$\Delta x_i = x_i - x_{i-1} = \alpha \Delta x_{i-1} - \gamma_{i-1} \nabla_f (x_{i-1})^T$$

con $0 \leq \alpha \leq 1$.

Gradiente descendiente con momentum

El término extra reduce las oscilaciones y suaviza las actualizaciones del gradiente.

Los métodos de gradiente descendiente descendiente con y sin momentum se los llama "*batch methods*" porque trabajan sobre todo el dataset completo.

Gradiente descendiente estocástico

Para cuando calcular el gradiente es muy costoso, se busca aproximar el gradiente de una forma más "barata". Si esta aproximación apunta en dirección parecida al gradiente sirve para acercarse al mínimo.

El gradiente descendiente estocástico (SGD) es una aproximación estocástica del método de gradiente descendiente. Es **estocástica** porque obtenemos una aproximación ruidosa del valor verdadero del gradiente.

La idea detrás de SGD es que si la función de costo $L(\theta)$ se puede escribir como suma pérdidas $L(\theta) = \sum_{n=1}^{N} L_n(\theta)$, cada una dependiente de una muestra, entonces en cada iteración puedo optimizar sólo respecto de un subconjunto de $\{L_1(\theta), \ldots, L_N(\theta)\}$. Esto se lo conoce como "*minibatch methods*".

Gradiente descendiente estocástico

Algunos comentarios para elegir el tamaño el *batch*:

- Si el *batch* es más grande (tomo más muestras) da una estimación más precisa del gradiente, pero es más costos
- Las arquitecturas multicore quedan subutilizadas si el tamaño el batch es muy pequeño
- Considerar los límites del hardware a la hora de elegir el tamaño del *batch*. En general la cantidad de memoria escala con el tamaño del *batch*.

Optimización con restricciones

Lagrangiano

Restricciones

¿Qué ocurre cuando la función que queremos optimizar se encuentra sujeta a restricciones?

Sean $f : \mathbb{R}^n \to \mathbb{R}$, $g_i : \mathbb{R}^n \to \mathbb{R}\ i = 1, \ldots, m$ y $h_i : \mathbb{R}^n \to \mathbb{R}$.
Consideremos el problema de optimización con restricciones

$$\begin{aligned} \min_x \quad & f(x) \\ s.t. \quad & g_i(x) \leq 0 \quad \forall\, i = 1, \ldots, m. \\ & h_i(x) = 0 \quad \forall\, i = 1, \ldots k \end{aligned}$$

Para resolver este problema recurrimos a los multiplicadores de Lagrange.

Lagrangiano: definición

Asociamos al problema de la diapositiva anterior el Lagrangiano introduciendo los multiplicadores de Lagrange $\lambda_i \geq 0$ correspondientes a cada g_i tal que

$$\mathcal{L}(x, \boldsymbol{\lambda}) = f(x) + \sum_{i=1}^{m} \lambda_i g_i(x) + \sum_{i=1}^{k} \mu_i h_i(x) = f(x) + \boldsymbol{\lambda} \boldsymbol{g}(x) + \boldsymbol{\mu} \boldsymbol{h}(x)$$

donde $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_m]^T$, $\boldsymbol{g}(x) = [g_1(x), \ldots, g_m(x)]^T$ y $\boldsymbol{h}(x) = [h_1(x), \ldots, h_k(x)]^T$.

Al conjunto de punto que satisface las restricciones, i.e. $\{x \in \mathbb{R}^n : g_1(x) \leq 0, \ldots, g_m(x) \geq 0, h_1(x) = 0, \ldots, h_k(x) = 0\}$, se lo llama conjunto factible.

Problema dual

Def: Al problema

$$\begin{aligned} \min_{x} \quad & f(x) \\ s.t. \quad & g_i(x) \leq 0 \quad \forall\, i = 1, \ldots, m. \\ & h_i(x) = 0 \quad \forall\, i = 1, \ldots k \end{aligned}$$

se lo conoce como problema primal. El problema dual asociado está dado por

$$\max_{\boldsymbol{\lambda} > 0, \boldsymbol{\mu}} \mathcal{D}(\boldsymbol{\lambda}, \boldsymbol{\mu})$$

donde $\boldsymbol{\lambda}$ y $\boldsymbol{\mu}$ son las variables duales y $\mathcal{D}(\boldsymbol{\lambda}, \boldsymbol{\mu}) = \min_{x} \mathcal{L}(x, \boldsymbol{\lambda}, \boldsymbol{\mu})$.
Si f y las g_i son derivables, el problema dual se resuelve calculando $\frac{\partial \mathcal{L}}{\partial x} = 0$, $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}} = 0$ y $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\lambda}} = 0$

Problema dual

En la definición anterior entran en juago dos conceptos:

1. Desigualdad *minimax*: para cualquier función de dos argumentos $\varphi(x, y)$ vale que el maximin es menor que el minimax:

   $$\max_y \min_x \varphi(x, y) \leq \min_x \max_y \varphi(x, y)$$

2. Dualidad débil: Los valores primales son siempre mayores o iguales a los duales

Bibliografía

Bigliografía

- Multiplicadores de Lagrange
- "Deep Learning", de Ian Goodfellow, Yoshua Bengio y Aaron Courville. Secciones 4.3 (GD) y 4.4 (optimización con restricciones)