PONTIFICIA
UNIVERSIDAD
CATÓLICA DE
VALPARAÍSO

Introducción al Deep Learning

Dr. Ing. Gabriel Hermosilla Vigneau

Introducción

- La inteligencia artificial (**IA** o **AI**) es un campo de investigación muy amplio, donde las máquinas muestran capacidades cognitivas como los comportamientos de aprendizaje, la interacción proactiva con el entorno, la inferencia y la deducción, la visión por ordenador, el reconocimiento del habla, la resolución de problemas, la representación del conocimiento, la percepción y muchos otros.

- Para obtener más información, consulte este artículo: Inteligencia artificial: un enfoque moderno, por S. Russell y P. Norvig, Prentice Hall, 2003 (Artificial Intelligence: A Modern Approach, by S. Russell and P. Norvig, Prentice Hall, 2003).

- Más coloquialmente, **AI** denota cualquier actividad en la que las máquinas imiten comportamientos inteligentes que suelen mostrar los humanos. La inteligencia artificial se inspira en elementos de informática, matemáticas y estadísticas.

Introducción

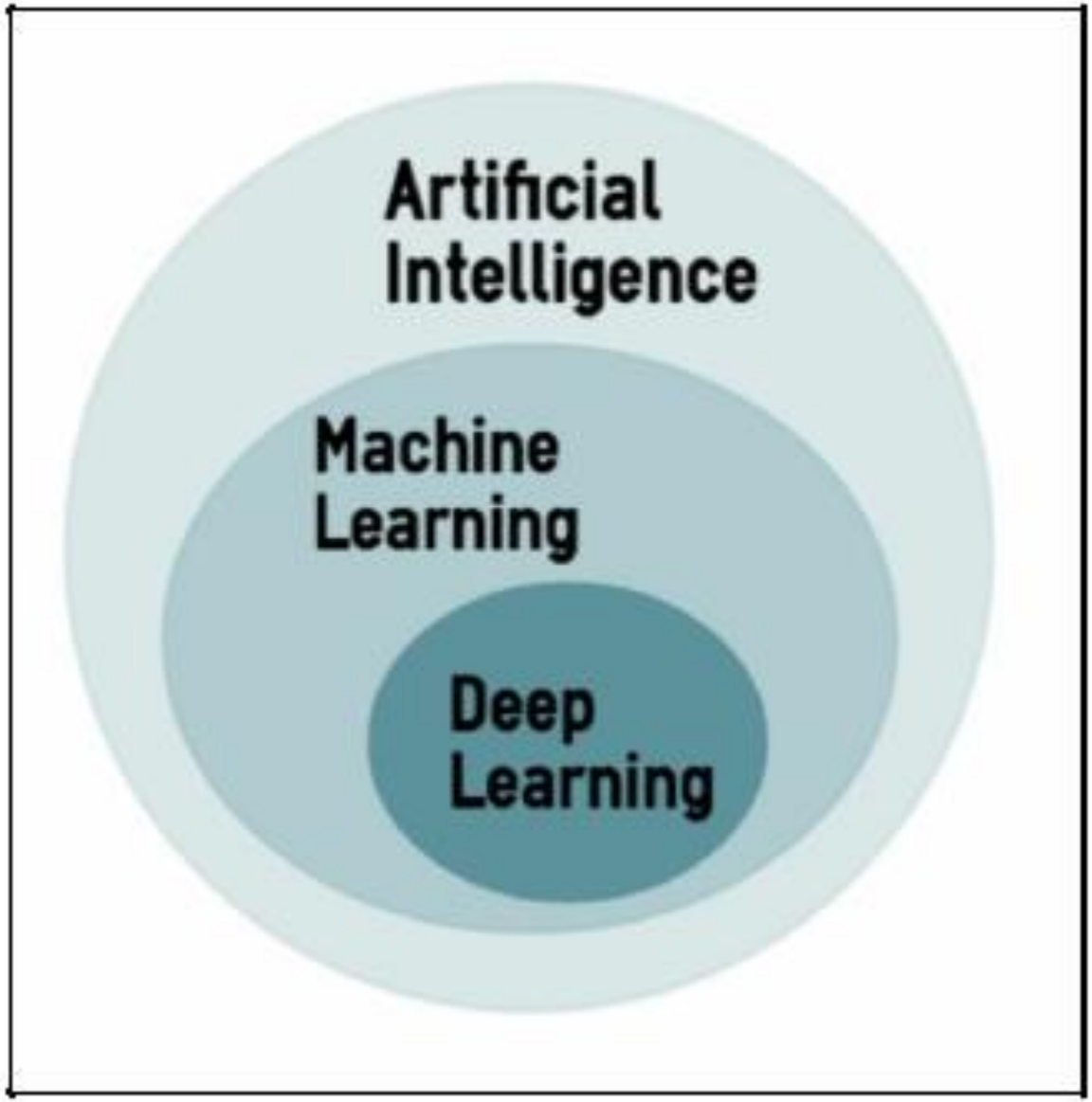
- El Machine Learning (**ML**) es un subgrupo de inteligencia artificial que se centra en enseñar a las computadoras cómo aprender sin la necesidad de programarse para tareas específicas (para obtener más información, consulte Reconocimiento de patrones y Aprendizaje automático, por C. M. Bishop, Springer, 2006). De hecho, la idea clave detrás de **ML** es que es posible crear algoritmos que aprendan y hagan predicciones sobre los datos.

- Hay tres categorías amplias diferentes de **ML**. En el **aprendizaje supervisado**, a la máquina se le presentan los datos de entrada y los resultados deseados, y el objetivo es aprender de esos ejemplos de entrenamiento de tal manera que se puedan hacer predicciones significativas para obtener datos nuevos e invisibles. En el **aprendizaje no supervisado**, la máquina se presenta solo con datos de entrada y la máquina tiene que encontrar alguna estructura significativa por sí misma sin supervisión externa. En el **aprendizaje por refuerzo**, la máquina actúa como un agente que interactúa con el entorno y aprende cuáles son los comportamientos que generan recompensas.

Introducción

- El Deep Learning (**DL**) es un subconjunto particular de metodologías de Machine Learning que utilizan **redes neuronales artificiales** (RNA, ANN), ligeramente inspiradas en la estructura de las neuronas ubicadas en el cerebro humano (para obtener más información, consulte el artículo Learning Deep Architectures for AI, de Y. Bengio , Found. Trends, vol. 2, 2009).

- Informalmente, la palabra profundo se refiere a la presencia de muchas capas en la red neuronal artificial, pero este significado ha cambiado con el tiempo. Mientras que hace 4 años, 10 capas ya eran suficientes para considerar una red tan profunda, hoy en día es más común considerar una red tan profunda cuando tiene cientos de capas.


Artificial
Intelligence
Machine
Learning
Deep
Learning

Introducción

- Deep Learning (DL) es un verdadero tsunami para el aprendizaje automático, ya que un número relativamente pequeño de metodologías inteligentes se han aplicado con gran éxito a muchos dominios diferentes (imagen, texto, video, habla y visión), lo que mejora significativamente los resultados anteriores de vanguardia, durante docenas de años.

- El éxito de **DL** también se debe a la disponibilidad de más datos de entrenamiento (como ImageNet para imágenes) y la disponibilidad de GPU de costo relativamente bajo para una computación numérica muy eficiente.

- Google, Microsoft, Amazon, Apple, Facebook y muchos otros utilizan esas técnicas de aprendizaje profundo todos los días para analizar grandes cantidades de datos. Sin embargo, este tipo de experiencia no se limita más al dominio de la investigación académica pura y a las grandes empresas industriales. Se ha convertido en una parte integral de la producción moderna de software y, por lo tanto, es algo que el estudiante debe dominar definitivamente.

Machine Learning

Machine Learning: Términos Básicos

- **Machine learning**: corresponde al subcampo de la inteligencia artificial que proporciona a las máquinas la capacidad de aprender **sin ser explícitamente programados**.

- Tipos de aprendizaje

  - **Supervisado**: Aprendizaje con muchos ejemplos etiquetados (Ejemplo: Clasificación de correo ya clasificado (spam, no deseado, etc.))

  - **No Supervisado**: Los datos de entrenamiento no incluyen las etiquetas (Ejemplo: clustering (K-means) o Principal Component Analysis (PCA))

  - **Aprendizaje por refuerzo**: Determinar las acciones a llevar a cabo mediante prueba y error (Ejemplo: Aprender a jugar ajedrez, recompensa: ganar o perder, hide and seek, atari-games, etc.)

Machine Learning: Términos Básicos

Los sistemas de machine learning aprenden cómo combinar entradas para producir predicciones útiles sobre datos nunca antes vistos.

Etiquetas:

Una etiqueta es el valor que estamos prediciendo, por ejemplo la variable *y* en la regresión lineal simple. La etiqueta podría ser el precio futuro del trigo, el tipo de animal que se muestra en una imagen, el significado de un clip de audio o simplemente cualquier cosa.

Atributos:

Un atributo es una variable de entrada, por ejemplo la variable **x** en la regresión lineal simple. Un proyecto de machine learning simple podría usar un solo atributo, mientras que otro más sofisticado podría usar millones de atributos, especificados como:

$$\{x_1, x_2, \ldots x_N\}$$

Machine Learning: Términos Básicos

Ejemplos:

Un **ejemplo** es una instancia de datos en particular, **x**. La **x** se coloca en negrita para indicar que es un vector.

Los ejemplos se dividen en dos categorías:

- ejemplos etiquetados
- ejemplos sin etiqueta

Un **ejemplo etiquetado** incluye tanto atributos como la etiqueta. Los ejemplos etiquetados se usan para **entrenar** el modelo.

Un **ejemplo sin etiqueta** contiene atributos, pero no la etiqueta.

Una vez que el modelo se entrena con ejemplos etiquetados, ese modelo se usa para predecir la etiqueta en ejemplos sin etiqueta.

Machine Learning: Términos Básicos

Modelos:

Un **modelo** define la relación entre los atributos y la etiqueta. Por ejemplo, un modelo de detección de spam en el correo electrónico o un modelo de reconocimiento de rostros. Destaquemos dos fases en el ciclo de un modelo:

Entrenamiento significa crear o **aprender** el modelo. Es decir, le muestras ejemplos etiquetados al modelo y permites que este aprenda gradualmente las relaciones entre los atributos y la etiqueta.

Inferencia significa aplicar el modelo entrenado a ejemplos sin etiqueta. Es decir, usar el modelo entrenado para realizar predicciones útiles.

Machine Learning: Términos Básicos

Regresión frente a clasificación

Un modelo de **regresión** predice valores continuos. Por ejemplo, los modelos de regresión hacen predicciones que responden a preguntas como las siguientes:

¿Cuál es el valor de una casa en California?
¿Cuál es la probabilidad de que un usuario haga clic en este anuncio?

Un modelo de **clasificación** predice valores discretos. Por ejemplo, los modelos de clasificación hacen predicciones que responden a preguntas como las siguientes:

¿Un mensaje de correo electrónico determinado es spam o no es spam?
¿Esta imagen es de un perro, un gato o un hámster?

Machine Learning: Regresión lineal

Regresión lineal

La regresión lineal es un método para encontrar la línea recta o el hiperplano que mejor se adapta a un conjunto de puntos.

Ejemplo: cantos de grillos vs temperatura

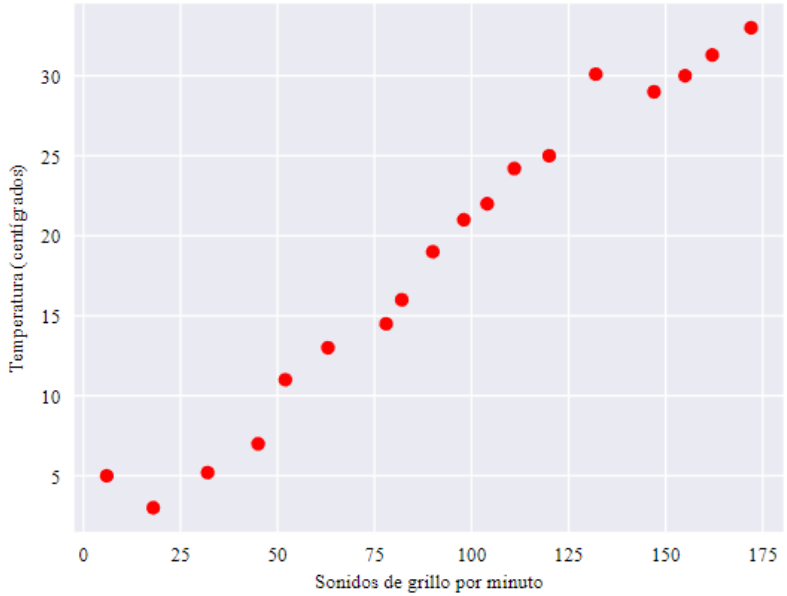


Figura 1. Cantos por minuto contra temperatura

Machine Learning: Regresión lineal

Regresión lineal

La representación muestra que la cantidad de cantos aumenta con la temperatura. ¿Es lineal la relación entre los cantos y la temperatura? Sí, ya que es posible dibujar una línea recta como la siguiente para representar dicha relación:

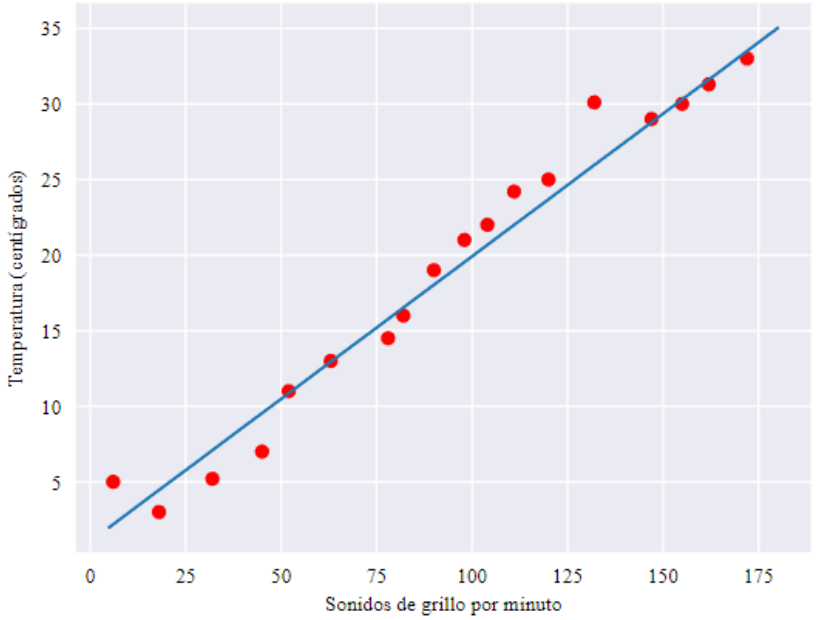


Figura 2. Una relación lineal

Machine Learning: Regresión lineal

Regresión lineal

Si bien la línea no pasa perfectamente por cada punto, demuestra con claridad la relación entre la temperatura y los cantos por minuto para dichos puntos. Si aplicamos un poco de álgebra, podemos determinar esta relación de la siguiente manera:

$$y = mx + b$$

donde:

y es la temperatura en grados centígrados, correspondiente al valor que intentamos predecir.

m es la pendiente de la línea.

x es la cantidad de cantos por minuto, correspondiente al valor de nuestro atributo de entrada.

b es la intersección en el eje y (eje de las ordenadas).

Machine Learning: Regresión lineal

Regresión lineal

Según las convenciones del machine learning, la ecuación para un modelo se escribirá de una forma un poco diferente:

$$y' = b + w_1 x_1$$

donde:

y' es la etiqueta predicha (un resultado deseado).

b es la ordenada al origen (la intersección en y). En la literatura se hace referencia a ella como w_0.

w_1 es la ponderación del atributo 1. La ponderación es el mismo concepto de la "pendiente" m, que se indicó anteriormente.

x_1 es un atributo (una entrada conocida).

Machine Learning: Regresión lineal

Regresión lineal

Para **inferir** (predecir) la temperatura y' para un valor nuevo de cantos de grillos por minuto x_1, solo agrega el valor x_1 a este modelo.

Para generar modelos más sofisticados, es posible agregar más atributos ponderados por su respectivo peso ($w_i \cdot x_i$). Por ejemplo, un modelo que se basa en tres atributos usaría la siguiente ecuación:

$$y' = b + w_1x_1 + w_2x_2 + w_3x_3$$

Machine Learning: Entrenamiento y pérdida

Entrenamiento y pérdida

Entrenar un modelo simplemente significa aprender (determinar) valores correctos para todas las ponderaciones (pesos w_i) y las ordenadas al origen (bias) de los ejemplos etiquetados. En un aprendizaje supervisado, el algoritmo de un aprendizaje automático construye un modelo al examinar varios ejemplos e intentar encontrar un modelo que minimice la pérdida. Este proceso se denomina **minimización del riesgo empírico**.

La pérdida es una penalidad por una predicción incorrecta. Esto quiere decir que la **pérdida** es un número que indica qué tan incorrecta fue la predicción del modelo en un solo ejemplo. Si la predicción del modelo es perfecta, la pérdida es cero; de lo contrario, la pérdida es mayor.

El objetivo de entrenar un modelo es encontrar un conjunto de ponderaciones (pesos) y ordenadas al origen (bias) que, en promedio, tengan pérdidas *bajas* en todos los ejemplos.

Machine Learning: Entrenamiento y pérdida

Entrenamiento y pérdida

Por ejemplo, la Figura 3 muestra un modelo al lado izquierdo con una pérdida alta, y al lado derecho un modelo con pérdida baja. Note que:
La flecha roja representa la pérdida.
La línea azul representa las predicciones.

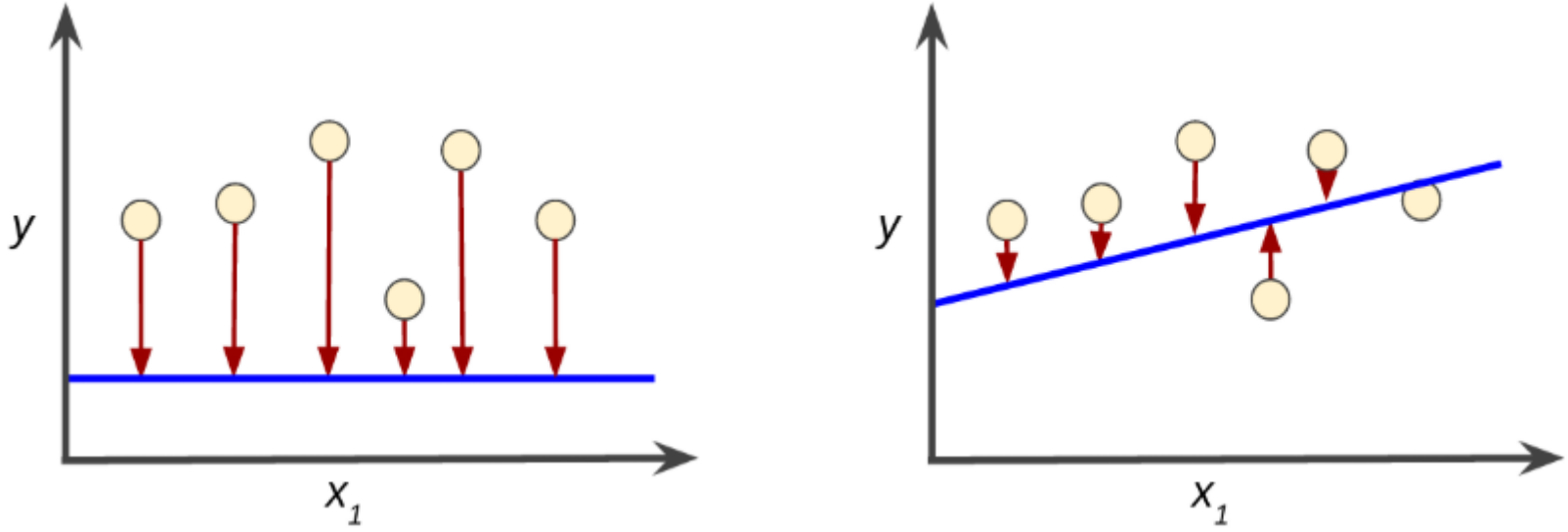


Figura 3. Pérdida alta en el modelo de la izquierda; pérdida baja en el modelo de la derecha.

Machine Learning: Entrenamiento y pérdida

Entrenamiento y pérdida

Pérdida al cuadrado:

Los modelos de regresión lineal que se examinan aquí usan una función de pérdida llamada **pérdida al cuadrado** (también conocida como pérdida **L_2**). A continuación, se muestra la pérdida al cuadrado para un único ejemplo:
= (observación − predicción(x))^2
= $(y - y')^2$

El **error cuadrático medio** (**ECM**) es el promedio de la pérdida al cuadrado de cada ejemplo. Para calcular el ECM, sumamos todas las pérdidas al cuadrado de los ejemplos individuales y, luego, lo dividimos por la cantidad de ejemplos:

$$ECM = \frac{1}{N} \sum_{(x,y)\in D} (y - predicción(x))^2$$

Machine Learning: Entrenamiento y pérdida

Entrenamiento y pérdida

Error cuadrático medio :

$$ECM = \frac{1}{N} \sum_{(x,y)\in D} (y - predicción(x))^2$$

donde:

(x, y) es un ejemplo en el que:

x es el conjunto de atributos (p. ej., cantos por minuto, edad) que el modelo usa para realizar las predicciones.

y es la etiqueta del ejemplo (p. ej., temperatura).

$predicción(x)$ es un atributo de las ponderaciones y las ordenadas al origen en combinación con el conjunto de atributos x.

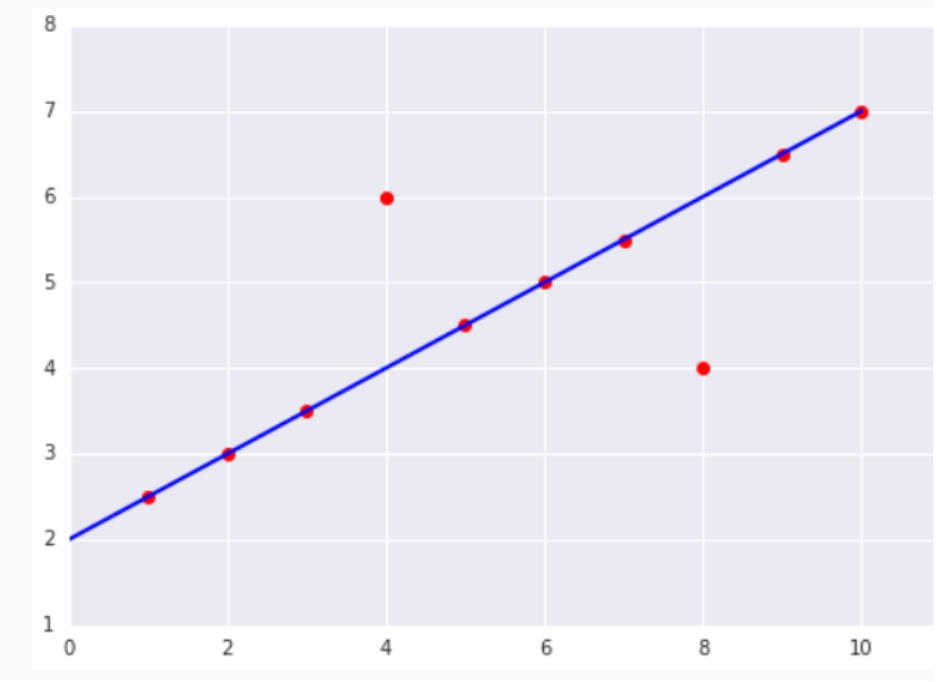
D es el conjunto de datos que contiene muchos ejemplos etiquetados, que son los pares (x, y).

N es la cantidad de ejemplos en D.

Machine Learning: Entrenamiento y pérdida

Entrenamiento y pérdida

Ejemplo:



$$ECM = \frac{0^2+0^2+0^2+2^2+0^2+0^2+0^2+2^2+0^2+0^2}{10} = 0{,}8$$

Machine Learning: Entrenamiento y pérdida

Reducción de la pérdida

La Figura 4 sugiere el proceso iterativo de prueba y error que usan los algoritmos de machine learning para entrenar un modelo:

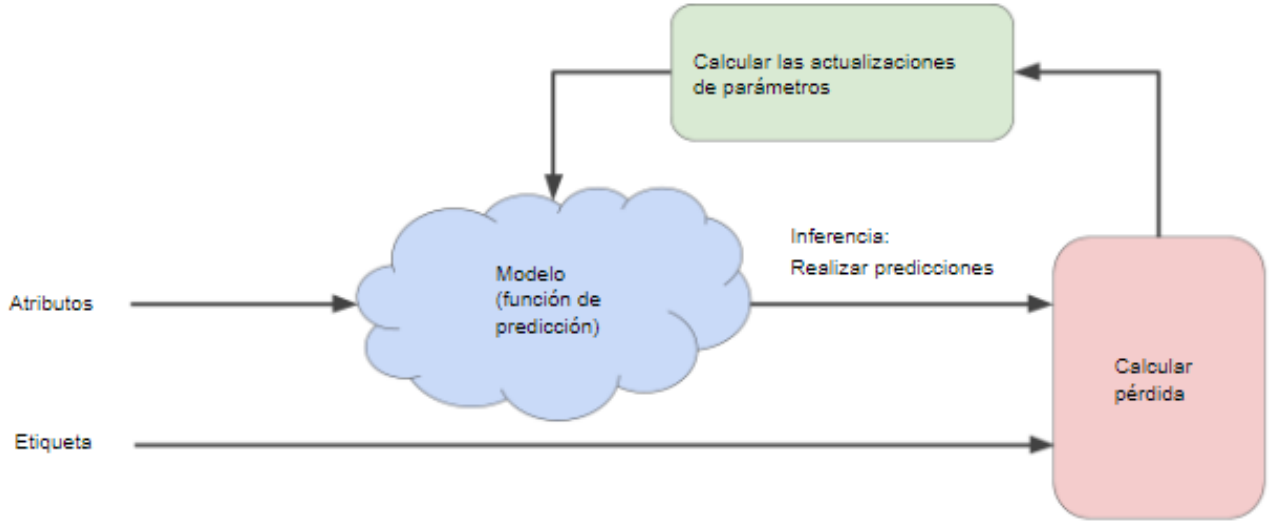


Figura 4. Un enfoque iterativo para entrenar un modelo.

Machine Learning: Entrenamiento y pérdida

Reducción de la pérdida

El "modelo" toma uno o más atributos como entrada y devuelve una predicción (y') como resultado. Para simplificar, considera un modelo que toma un atributo y devuelve una predicción:

$$y' = b + w_1 x_1$$

Para la regresión lineal, los valores de inicio no son importantes. Podríamos elegir valores al azar, pero tomaremos los siguientes valores triviales en su lugar:

$$b = 0 \qquad w_1 = 0$$

Supongamos que el primer valor del atributo es 10. Al vincular ese valor con el atributo de predicción, se obtiene lo siguiente:

```
y' = 0 + 0(10)
y' = 0
```

Machine Learning: Entrenamiento y pérdida

Reducción de la pérdida

La parte de “cálculo de pérdida" de la Figura 4 es la **función de pérdida** que usará el modelo. Suponga que usamos la función de pérdida al cuadrado. La función de pérdida incorpora dos valores de entrada:

y': la predicción del modelo para los atributos x
y: la etiqueta correcta correspondiente a los atributos x.

Finalmente, llegamos a la parte de “actualizar parámetros" de la Figura 4. Aquí, el sistema de aprendizaje automático examina el valor de la función de pérdida y genera valores nuevos para $[w_1]$, y $[b]$

Machine Learning: Entrenamiento y pérdida

Reducción de la pérdida

Note que el **cuadro verde** de la Figura 4, calcula valores nuevos, luego, el sistema de aprendizaje automático vuelve a evaluar todos esos atributos con todas las etiquetas y se obtiene un *nuevo valor para la función de pérdida*, que genera valores de parámetros nuevos.

El aprendizaje continúa iterando hasta que el algoritmo descubre los parámetros del modelo con la pérdida más baja posible. En general, iteras hasta que la pérdida general deja de cambiar o, al menos, cambia muy lentamente. Cuando eso ocurre, decimos que el modelo ha **convergido**.

Machine Learning: Descenso de gradiente

Descenso de gradiente

Para el tipo de problemas de regresión que hemos estado examinando, la representación resultante de pérdida frente a w_1 siempre será convexa. En otras palabras, la representación siempre se parecerá a una parábola o tazón.

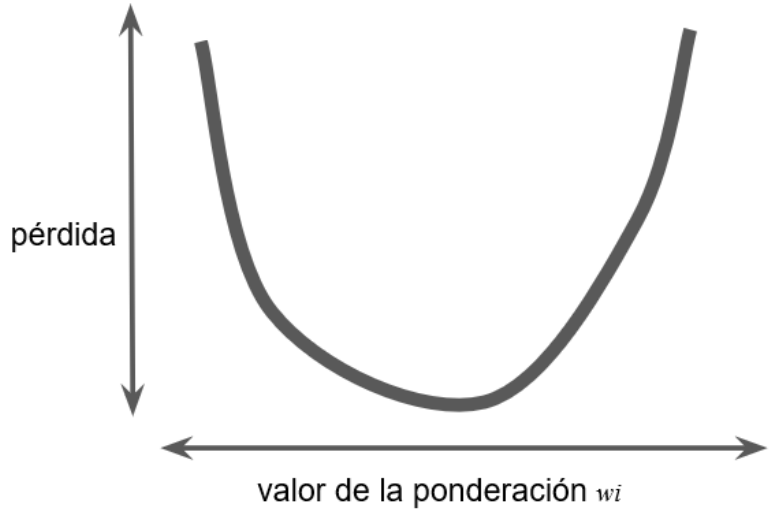


Figura 5. Los problemas de regresión producen gráficas de pérdida convexa vs. pesos.

Machine Learning: Descenso de gradiente

Descenso de gradiente

Los problemas convexos tienen un solo mínimo, es decir, un solo lugar en el que la pendiente es exactamente 0. Ese mínimo es donde converge la función de pérdida.

Calcular la función de pérdida para cada valor concebible de w_1 en todo el conjunto de datos sería una manera ineficaz de buscar el punto de convergencia. Un mecanismo más útil, muy popular en machine learning, es denominado **descenso de gradiente**.

La primera etapa en el descenso de gradiente es elegir un valor de inicio (un punto de partida) para w_1. El punto de partida no es muy importante; por lo tanto, muchos algoritmos simplemente establecen en 0 o eligen un valor al azar. En la Figura 6, se muestra que elegimos un punto de partida levemente mayor que 0.

Machine Learning: Descenso de gradiente

Descenso de gradiente

Luego, el algoritmo de descenso de gradientes calcula el gradiente de la curva de pérdida en el punto de partida. En resumen, un **gradiente** es un vector de derivadas parciales; indica por dónde es más cerca o más lejos acercarse a un óptimo. Note que que el gradiente de pérdida con respecto a un solo peso (como en la Figura 6) es equivalente a la derivada.

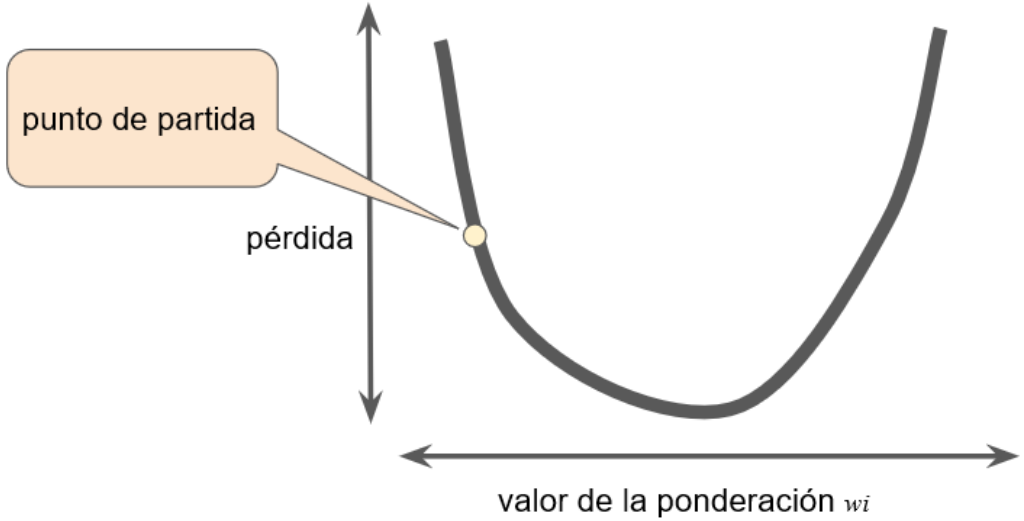


Figura 6. Un punto de partida para el descenso de gradientes.

Machine Learning: Descenso de gradiente

Descenso de gradiente

El gradiente de f (x, y) es un vector de dos dimensiones que te indica en qué dirección (x, y) debe moverse para acercarse a un óptimo.

En machine learning, las gradientes se usan en el descenso de gradiente. Con frecuencia tenemos una función de pérdida de muchas variables que intentamos minimizar, y tratamos de hacerlo al seguir el negativo de la gradiente de la función.

Note que la gradiente es un vector, de manera que tiene las dos características siguientes: una dirección y una magnitud.

Machine Learning: Descenso de gradiente

Descenso de gradiente

La gradiente siempre apunta en la dirección del aumento más empinado de la función de pérdida. El algoritmo de descenso de gradientes toma un paso en dirección de la gradiente negativa para reducir la pérdida lo más rápido posible.

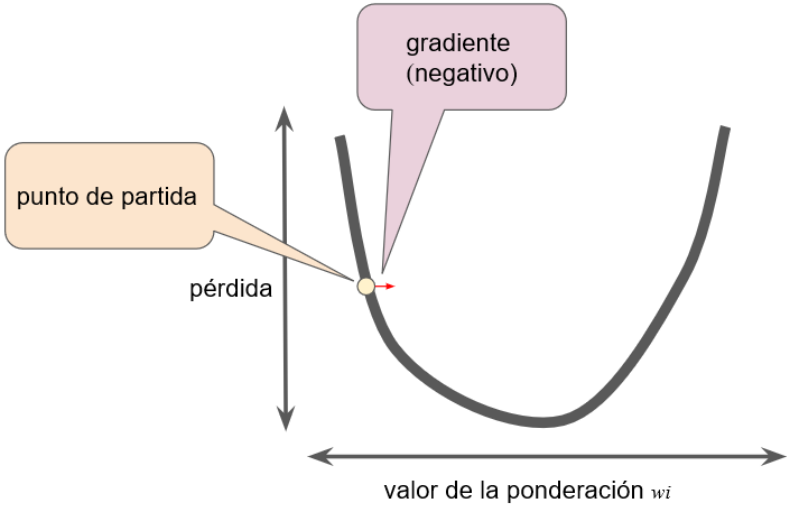


Figura 7. El descenso de gradientes se basa en gradientes negativas.

Machine Learning: Descenso de gradiente

Descenso de gradiente

Para determinar el siguiente punto a lo largo de la curva de la función de pérdida, el algoritmo de descenso de gradientes agrega alguna fracción de la magnitud de la gradiente al punto de partida, como se muestra en la Figura 8.
Luego, el descenso de gradientes repite este proceso y se acerca cada vez más al mínimo.

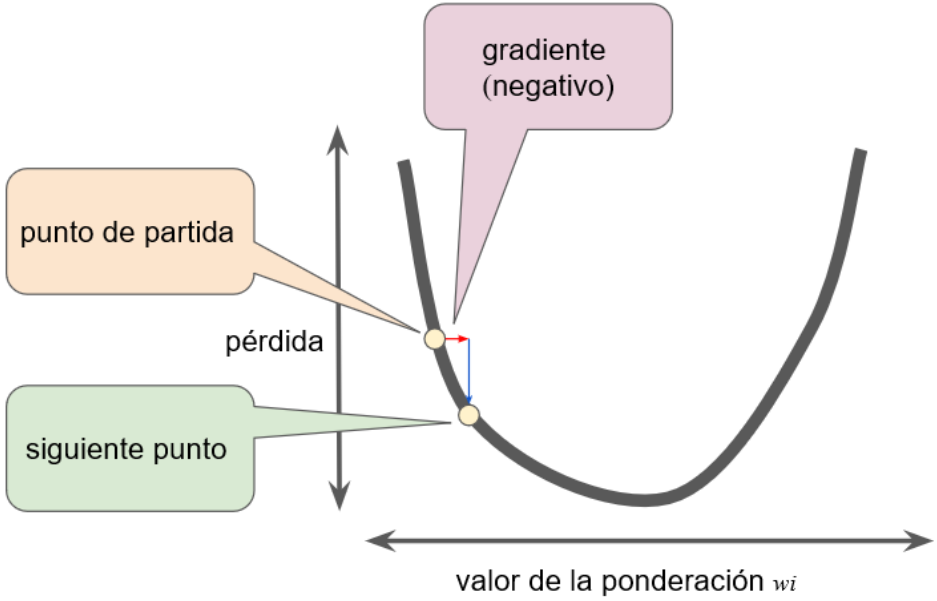


Figura 8. Un paso de la gradiente nos mueve hacia el siguiente punto en la curva de pérdida.

Machine Learning: Tasa de aprendizaje

Tasa de aprendizaje

Como se observó, el vector de gradiente tiene una dirección y una magnitud. Los algoritmos de descenso de gradiente multiplican el gradiente por un escalar conocido como **tasa de aprendizaje** (o **tamaño del paso** en algunas ocasiones) para determinar el siguiente punto.

Por ejemplo, si la magnitud del gradiente es 2.5 y la tasa de aprendizaje es 0.01, el algoritmo de descenso de gradientes tomará el siguiente punto 0.025 más alejado del punto anterior.

Machine Learning: Tasa de aprendizaje

Tasa de aprendizaje

Los **hiperparámetros** son los controles que los programadores ajustan en los algoritmos de machine learning. La mayoría de los programadores pasan gran parte de su tiempo ajustando la tasa de aprendizaje. Si eliges una tasa de aprendizaje muy pequeña, el aprendizaje llevará demasiado tiempo:

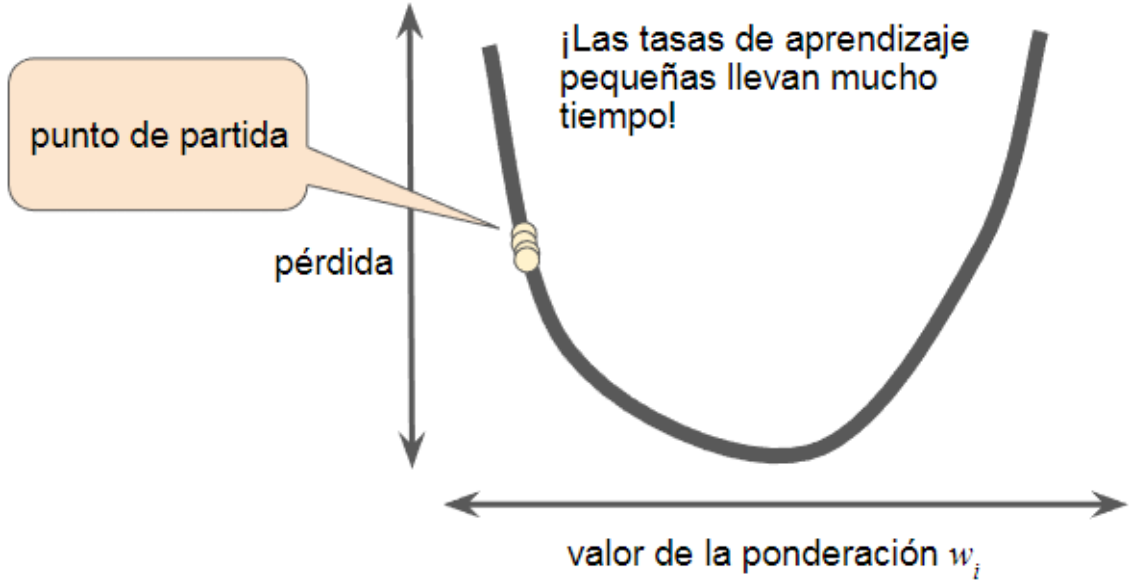


Figura 9. La tasa de aprendizaje es muy pequeña.

Machine Learning: Tasa de aprendizaje

Tasa de aprendizaje

A la inversa, si especificas una tasa de aprendizaje muy grande, el siguiente punto rebotará al azar eternamente en la parte inferior:

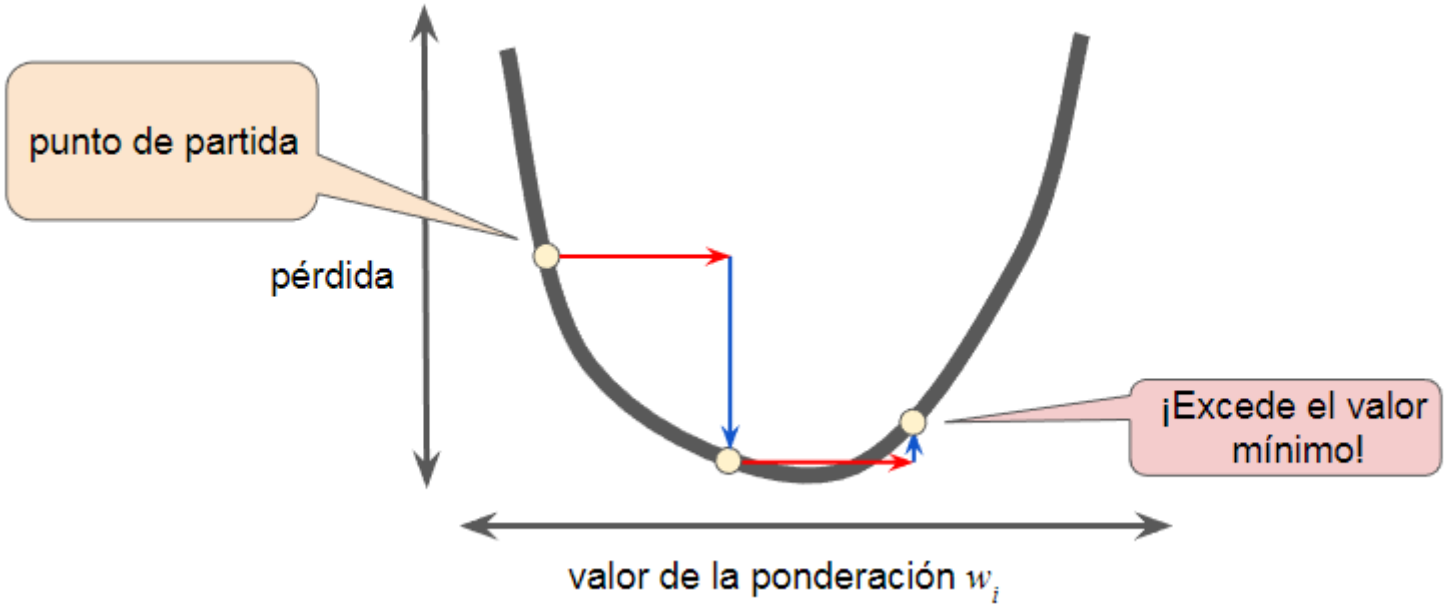


Figura 10. La tasa de aprendizaje es muy grande.

Machine Learning: Tasa de aprendizaje

Tasa de aprendizaje

Hay una tasa de aprendizaje óptima para cada problema de regresión. El valor optimo está relacionado con qué tan plana es la función de pérdida. Si sabes que el gradiente de la función de pérdida es pequeño, usa una tasa de aprendizaje mayor, que compensará el gradiente pequeño y dará como resultado un tamaño del paso más grande:

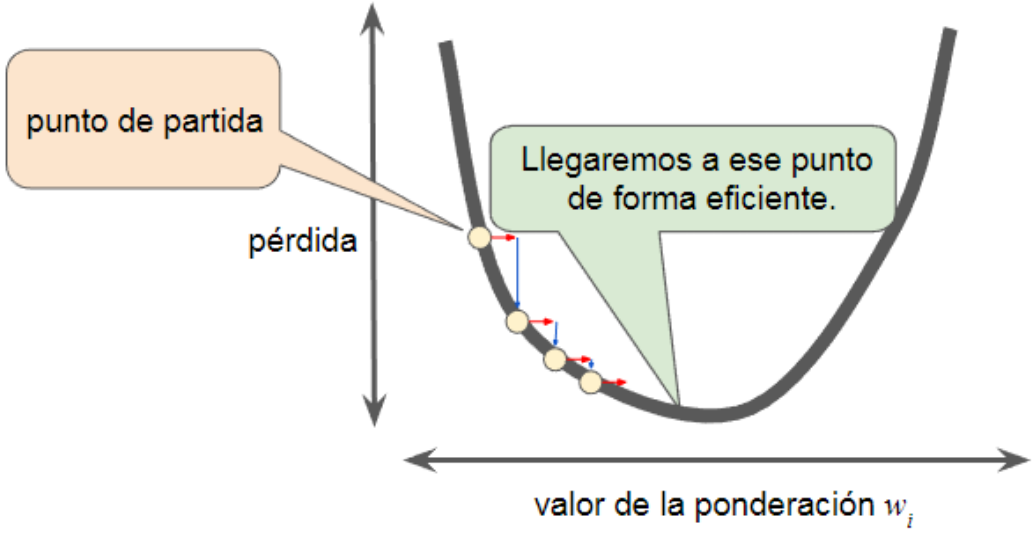


Figura 11. La tasa de aprendizaje es correcta.

Machine Learning: Descenso de gradiente estocástico

Descenso de gradiente estocástico

En el descenso del gradiente, un **lote** es la cantidad total de ejemplos que usas para calcular el gradiente en una sola iteración. Hasta ahora, hemos supuesto que el lote era el conjunto de datos completo.

Al trabajar a la escala de Google, los conjuntos de datos suelen tener miles de millones o incluso cientos de miles de millones de ejemplos. Además, los conjuntos de datos de Google con frecuencia contienen inmensas cantidades de atributos. En consecuencia, un lote puede ser enorme. Un lote muy grande puede causar que incluso una sola iteración tome un tiempo muy prolongado para calcularse.

Es probable que un conjunto de datos grande con ejemplos muestreados al azar contenga datos redundantes. De hecho, la redundancia se vuelve más probable a medida que aumenta el tamaño del lote. Un poco de redundancia puede ser útil para atenuar los gradientes inconsistentes, pero los lotes enormes tienden a no tener un valor mucho más predictivo que los lotes grandes.

Machine Learning: Descenso de gradiente estocástico

Descenso de gradiente estocástico

¿Cómo sería si pudiéramos obtener el gradiente correcto en promedio con mucho menos cómputo? Al elegir ejemplos al azar de nuestro conjunto de datos, podríamos estimar (si bien de manera inconsistente) un promedio grande de otro mucho más pequeño. El **descenso de gradiente estocástico (SGD)** lleva esta idea al extremo: usa un solo ejemplo (un tamaño del lote de 1) por iteración. Cuando se dan demasiadas iteraciones, el SGD funciona, pero es muy inconsistente. El término "estocástico" indica que el ejemplo único que compone cada lote se elige al azar.

El **descenso de gradiente estocástico de minilote (SGD de minilote)** es un equilibrio entre la iteración de lote completo y el SGD. Un minilote generalmente tiene entre *10 y 1,000 ejemplos*, elegidos al azar. El SGD de minilote reduce la cantidad de inconsistencia en el SGD, y sigue siendo más eficaz que el lote completo.