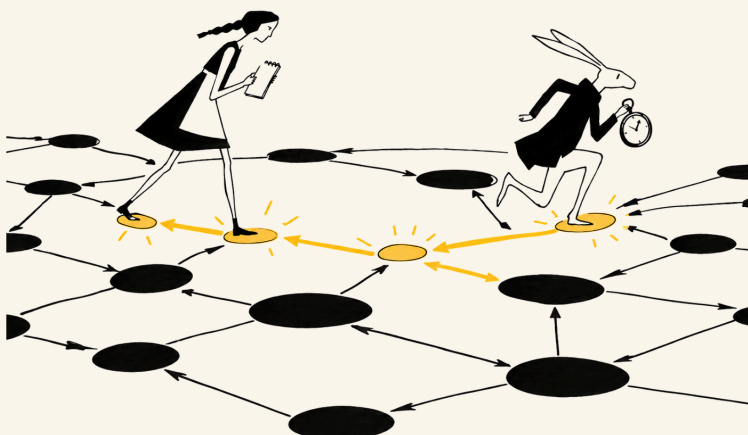


Las aventuras de Alicia en el País de las Maravillas Diferenciables



Simone Scardapane · 2024

Audiolecciones ilustradas

Lecturized

Las aventuras de Alicia en el País de las Maravillas Diferenciables

Audiolecciones ilustradas

Simone Scardapane · 2024

Contado como lecciones ilustradas por Lecturized

lecturized.com

Del original: Alice's Adventures in a Differentiable Wonderland, Volume I: A Tour of the
Land

Índice

Una receta sencilla, llevada a gran escala

Flechas que se reconocen entre sí

Cómo crecen los costes y hacia dónde es cuesta arriba

Bajando la cuesta en la niebla

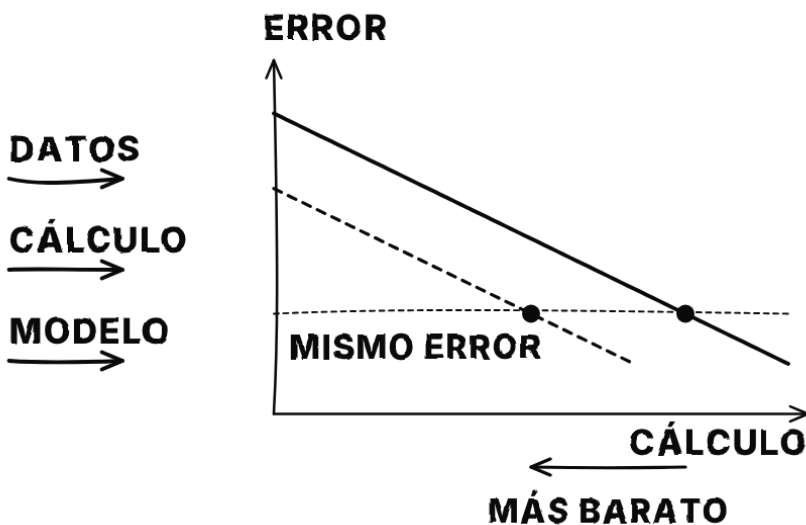
El momento, y de dónde salen los ejemplos

Una puntuación que bajar, y un examen que no ha visto

Muchos modelos posibles, y el más simple

Una receta sencilla, llevada a gran escala

Abra el asistente de chat de su teléfono y pregúntele algo que seguro nunca ha visto palabra por palabra, digamos, un limerick sobre el gato de su vecino y un formulario de impuestos. Probablemente espera un torpe remiendo de frases copiadas de algún sitio. Lo que vuelve es nuevo, acertado y a menudo gracioso. Algo dentro de ese programa ha aprendido mucho más que una lista de respuestas guardadas, y el libro empieza preguntando cómo se construye algo así. Su respuesta sorprende: a partir de un conjunto muy pequeño de principios y piezas, usados una y otra vez, y luego hechos enormes.



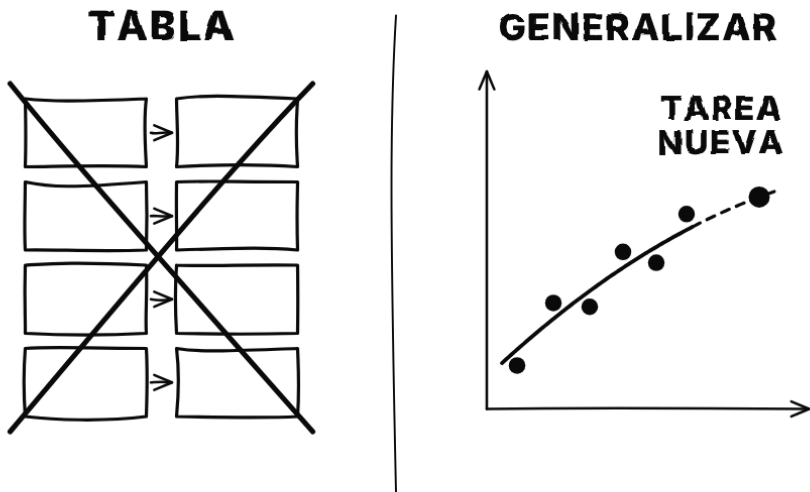
LECCIÓN AMARGA

Escalar baja el error

El libro lo llama el poder del escalado. Los investigadores descubrieron que, para casi cualquier tarea, añadir a la vez más datos, más potencia de cálculo y un modelo más grande hace que la precisión suba de forma predecible. Dicho al revés, el cálculo necesario para alcanzar una precisión dada sigue reduciéndose año tras año. El científico Richard Sutton llamó a esto la lección amarga: herramientas simples y generales más cada vez más cálculo vencen a los trucos especiales ingeniosos.

Aquí está el enigma que lo vuelve casi mágico. Estas redes son tan grandes que podrían simplemente memorizar todo lo que se les muestra y convertirse en una gigantesca tabla de consulta. En cambio, los modelos entrenados generalizan: resuelven tareas que nunca

estuvieron escritas en sus datos de entrenamiento, y cuanto más grandes son los datos, menos se apoyan en la pura memorización. Por eso su asistente puede escribir siquiera el limerick del gato.

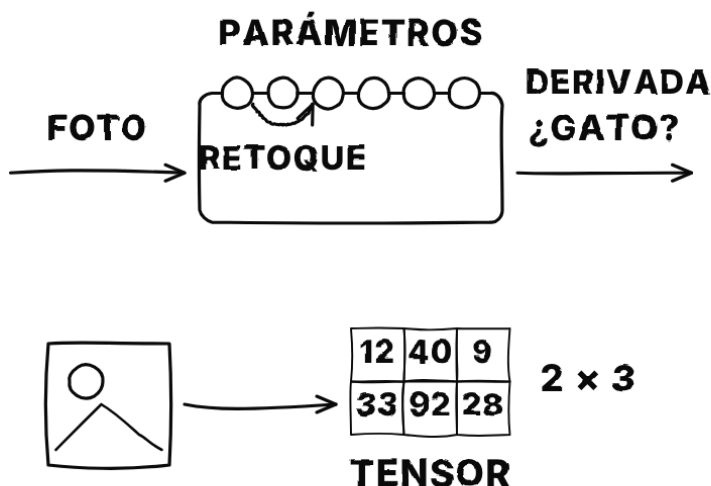


Memorizar o generalizar

El libro también señala cómo ha cambiado nuestra manera de usar estos modelos. Los enormes modelos preentrenados, llamados modelos fundacionales, pueden dirigirse simplemente dándoles una instrucción con texto o imágenes, mientras su interior sigue siendo una caja negra. Solo unos pocos equipos en el mundo pueden construir los más grandes, igual que solo unas pocas empresas pueden construir enormes programas de gestión empresarial. Aun así, con hardware corriente y algo de ingenio usted todavía puede ajustar,

fusionar, reducir o probar tales modelos, y para eso necesita mirar bajo el capó.

Mirar bajo el capó empieza con una idea: aprender es buscar. Usted escribe un programa con un número enorme de perillas ajustables, llamadas parámetros, y las gira hasta que el programa se comporta bien. Una aplicación de fotos que distingue gatos de perros se entrena exactamente así: se le muestran miles de imágenes etiquetadas y se retocan sus perillas tras cada error. Para buscar entre millones o miles de millones de perillas, el libro se limita a piezas cuyo efecto puede medirse con derivadas, que le dicen qué pasa si una perilla se mueve un poco. Por eso prefiere el nombre modelos diferenciables al de redes neuronales, y por eso construye todo a partir de formas automáticas de calcular derivadas y de bajar la pendiente paso a paso.



Aprender es buscar

Los modelos diferenciables no pueden trabajar directamente con letras o números enteros, que saltan en vez de variar suavemente, así que todos los datos se convierten en grandes matrices de números que el libro llama tensores. Una imagen, un sonido o una frase se convierten todos en tales matrices, y cada parte de un modelo se clasifica según qué forma de matriz recibe y devuelve. El resto del libro recorre luego las convoluciones para imágenes, sonido y texto, la atención y los transformadores, los grafos y los modelos recurrentes.

Ninguna de estas ideas es nueva. Entrenar modelos simples por descenso de gradiente se remonta al siglo diecinueve, y en 1958 el psicólogo Frank Rosenblatt ya levantaba revuelo en los periódicos con sus perceptrones. Lo que cambió fueron los datos y la potencia: los

concursos de imágenes de 2012 a 2017 y la llegada de los transformadores en 2017 mostraron lo bien que funcionan estas viejas ideas a gran escala. Quedan fuera aquí el plan del libro capítulo por capítulo y su notación para los tensores y sus formas; están en las secciones «Sobre este libro» y «Notación y símbolos».

Para recordar

Casi todo en la IA moderna se construye a partir de unos pocos principios y piezas simples, hechos poderosos al escalar juntos los datos, el cálculo y el tamaño del modelo. Los modelos grandes no solo memorizan; generalizan a tareas que nunca se les mostraron, y esa es la magia que el libro quiere explicar. Aprender es buscar: un programa con un número enorme de parámetros ajustables se afina hasta que se comporta bien. La búsqueda funciona porque cada pieza es diferenciable, así que las derivadas dicen cómo retocar cada parámetro, y por eso el libro habla de modelos diferenciables. Todos los datos, desde fotos hasta frases, se manejan como grandes matrices de números llamadas tensores.

Preguntas

1. Según las leyes de escalado, ¿qué suele ocurrir cuando los datos, la potencia de cálculo y el tamaño del modelo aumentan a la vez?

- a. La precisión sube de forma predecible, para casi cualquier tarea
 - b. La precisión sube solo en tareas de imágenes, no en lenguaje
 - c. El modelo empieza a memorizar y empeora en tareas nuevas
 - d. Nada cambia a menos que el diseño del modelo se repiense a mano
2. ¿Por qué sorprende que los modelos grandes entrenados resuelvan tareas que nunca se les mostraron?
- a. Porque son demasiado pequeños para guardar ejemplo alguno
 - b. Porque son lo bastante grandes para simplemente memorizar sus datos de entrenamiento como una tabla de consulta, y sin embargo generalizan
 - c. Porque se entrenaron solo con números, nunca con texto ni imágenes
 - d. Porque nadie puede darles instrucciones una vez entrenados
3. ¿Por qué el libro construye sus modelos solo con piezas diferenciables?
- a. Porque las piezas diferenciables son más baratas de guardar en memoria
 - b. Porque solo a las piezas diferenciables se les pueden dar instrucciones con texto
 - c. Porque las derivadas dicen cómo retocar cada uno de los millones de parámetros, lo que hace practicable la búsqueda de buenos ajustes
 - d. Porque las piezas diferenciables nunca necesitan datos para entrenarse

4. ¿Por qué una frase o una foto deben convertirse en grandes matrices de números antes de que un modelo diferenciable pueda usarlas?
- a. Porque las letras y los números enteros saltan en vez de variar suavemente, así que las derivadas no pueden trabajar con ellos directamente
 - b. Porque las matrices de números ocupan menos espacio que las letras
 - c. Porque la notación del libro prohíbe las letras
 - d. Porque solo las fotos pueden guardarse como letras

Respuestas

1. **a)** Los investigadores descubrieron que añadir a la vez más datos, más potencia de cálculo y un modelo más grande hace que la precisión suba de forma predecible para casi cualquier tarea.
2. **b)** Las redes son tan grandes que podrían memorizar todo lo que ven y convertirse en una gigantesca tabla de consulta; en cambio generalizan, y se apoyan menos en memorizar a medida que crecen los datos.
3. **c)** Aprender es una búsqueda entre un número enorme de parámetros, y las derivadas dicen qué pasa cuando una perilla se mueve un poco, así que guían la búsqueda automáticamente.
4. **a)** Los modelos diferenciables no pueden trabajar directamente con letras o números enteros, que saltan en vez de variar suavemente, así que todos los datos se convierten en matrices de números llamadas tensores.

Ejercicios

1. Piense en una aplicación de fotos que aprende a distinguir gatos de perros. Explique con sus propias palabras qué significa decir que

aprender es buscar, y cuáles son los parámetros en este ejemplo.

Pista: Imagine la aplicación como una máquina con un enorme panel de perillas y pregúntese qué les hace el entrenamiento.

Respuesta: La aplicación es un programa con un número enorme de perillas ajustables, sus parámetros. El entrenamiento le muestra imágenes etiquetadas y gira un poco las perillas tras cada error, buscando entre todas las posiciones posibles de las perillas una que haga que la aplicación separe bien gatos de perros.

2. Las ideas detrás de la IA de hoy, como el descenso de gradiente y el perceptrón, son antiguas. Explique, como lo hace esta parte, por qué solo se volvieron tan poderosas en los últimos quince años aproximadamente.

Pista: Pregúntese qué faltaba en 1958 que hoy abunda.

Respuesta: Los métodos se conocían desde hace mucho, pero no había suficientes datos ni potencia de cálculo para mostrar lo bien que funcionan. Los grandes concursos de imágenes desde 2012 y los transformadores desde 2017 mostraron que, con suficientes datos y parámetros, estas viejas ideas rinden notablemente bien.

3. Mucha gente usa ahora un modelo fundacional solo dándole instrucciones, como una caja negra. Explique por qué alguien podría querer entender, aun así, qué ocurre en su interior.

Pista: Piense en lo que podría hacer con un programa abierto más allá de simplemente ejecutarlo.

Respuesta: Dar instrucciones basta para usar el modelo, pero adaptarlo exige más: ajustarlo a su propia tarea, fusionar modelos, reducirlos para hardware débil o probar su robustez requiere mirar bajo el capó y ver cómo el modelo maneja los datos en su interior.

Flechas que se reconocen entre sí

Tome dos lápices y póngalos sobre la mesa, ambos partiendo del mismo punto. Apúntelos en la misma dirección, luego en ángulo recto, luego en direcciones opuestas, y pregúntese cada vez cuánto se parecen las dos direcciones. Puede sentir la respuesta sin ninguna aritmética: la misma dirección significa muy parecidas, ángulo recto significa sin relación, opuestas significa tan distintas como es posible. Esa simple sensación de parecido o distinto resulta ser la herramienta principal del capítulo de matemáticas, y la IA moderna la usa millones de veces por segundo.

PROD. ESCALAR

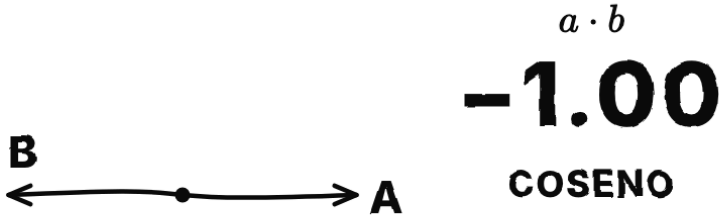


Iguals, en ángulo recto, opuestas

Antes de las matemáticas, el libro cuenta de dónde viene y qué no es. Nació de un curso sobre redes neuronales que el autor imparte en un máster en ciencia de datos en la Universidad Sapienza de Roma, y busca añadir su propio punto de vista a otros buenos libros sobre el tema. Subraya que teoría y código van de la mano, y que el auge de las redes neuronales desde 2012 aproximadamente debe mucho a bibliotecas de software como TensorFlow, PyTorch y JAX. No es un manual de programación, no intenta cubrirlo todo, y dice poco sobre hardware o sobre el oficio de hacer que modelos enormes funcionen en la práctica. Además, se publica bajo una licencia libre que permite a cualquiera remezclarlo y construir sobre él, que es justo lo que hace posible un curso como este.

El capítulo de matemáticas fija entonces el lenguaje para todo lo que sigue. Como vimos, todos los datos viven en tensores, matrices de números descritas por su forma, y se adaptan bien a los chips gráficos, que ejecutan muchas sumas simples en paralelo. Una sola fila de números es un vector, que puede imaginar como una flecha desde el origen, como sus lápices. Dos flechas pueden estirarse y sumarse para dar una tercera, y toda flecha tiene una longitud.

$$a \cdot b = a_1b_1 + a_2b_2$$



El producto escalar como un dial

La operación clave es el producto escalar: multiplicar dos vectores número a número y sumar los resultados, lo que da un solo número. Ese número es grande cuando las flechas apuntan en la misma dirección, cero cuando están en ángulo recto, y negativo cuando apuntan en direcciones opuestas, tal como mostraron sus lápices. Para flechas de longitud uno mide exactamente cuánto se parecen sus direcciones, y por eso se llama similitud coseno. El libro extrae la lección que más importa: el producto escalar es máximo cuando la segunda flecha coincide con la primera. En los capítulos siguientes, una flecha será la entrada y la otra los parámetros ajustables del modelo, así que el modelo responde con más fuerza cuando la entrada resuena con lo que sus parámetros han aprendido, una especie de coincidencia con una plantilla. Cuando su aplicación de fotos

encuentra todas las imágenes que se parecen a su perro, una comparación de este tipo es lo que mide cuánto se parecen dos imágenes.

Pruébalo

Tome dos lápices y sosténgalos desde el mismo punto de partida. Gire uno de ellos despacio desde apuntar a lo largo del otro hasta apuntar en sentido contrario. Debería sentir cómo la coincidencia cae de completa, pasando por nada en el ángulo recto, hasta un desacuerdo total al final, que es el mismo recorrido que hace el producto escalar.

Una tabla de números, una matriz, puede leerse como una pila de tales vectores, uno por fila, y el libro llama a esto un lote de datos. Multiplicar matrices no es entonces más que calcular muchos productos escalares a la vez, cada fila de una contra cada columna de la otra, en una sola operación. Cuando un asistente de chat procesa un lote entero de palabras de una vez, está haciendo exactamente esta clase de comparación masiva. Hay también un producto más simple, número a número, usado para enmascarar y escalar, y funciones como la exponencial suelen aplicarse a cada número de una tabla por separado, lo que difiere de las versiones matriciales especiales de los libros de texto. El libro advierte de una trampa común en el código: las bibliotecas estiran en silencio matrices de formas que no encajan para hacerlas coincidir, y dos vectores pueden convertirse por accidente en una tabla entera.

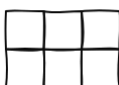
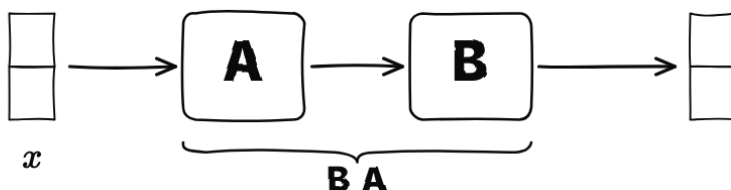
$$\begin{array}{c} \text{LOTE} \\ \hline \begin{array}{|c|c|c|} \hline 1 & 2 & 0 \\ \hline 0 & 1 & 3 \\ \hline \end{array} \end{array} \times \begin{array}{|c|c|} \hline 1 & 0 \\ \hline 1 & 1 \\ \hline 0 & 2 \\ \hline \end{array} = \begin{array}{|c|c|} \hline 3 & 2 \\ \hline 1 & 7 \\ \hline \end{array}$$

$$\begin{array}{|c|c|c|} \hline & & \\ \hline & & \\ \hline & & \\ \hline \end{array} + \begin{array}{|c|} \hline \\ \hline \\ \hline \\ \hline \end{array} = \begin{array}{|c|c|c|} \hline & & \\ \hline & & \\ \hline & & \\ \hline & & \\ \hline \end{array}$$

BROADCAST!

Muchos productos escalares a la vez

Por último, el libro explica por qué la multiplicación de matrices se define como se define. Toda función que respeta la suma y el estiramiento, llamada lineal, puede escribirse como una matriz, así que multiplicar una matriz por un vector significa aplicar la función, y multiplicar dos matrices significa aplicar una función tras otra. Quedan fuera aquí las fórmulas exactas de longitudes, productos escalares, productos de matrices e indexación; están en las secciones «Operaciones comunes con vectores» y «Operaciones comunes con matrices».



PILA DE FILAS

Matrices como funciones lineales

Para recordar

El libro nació de un curso universitario y trata la teoría y el código como socios, pensando en frameworks como PyTorch y JAX, mientras deja a un lado el hardware y la ingeniería. Los datos viven en tensores; un vector es una flecha, y una matriz es una pila de vectores, a menudo un lote de ejemplos. El producto escalar convierte dos flechas en un número que dice cuánto se parecen, y es máximo cuando coinciden, que es como un modelo reconoce los patrones que ha aprendido. La multiplicación de matrices son muchos productos escalares a la vez, y es también la manera de encadenar funciones lineales una tras otra.

Preguntas

1. Dos flechas de longitud uno apuntan en ángulo recto entre sí.
¿Qué dice su producto escalar?
 - a. Que son tan parecidas como es posible
 - b. Que no tienen relación: el producto escalar es cero
 - c. Que son tan distintas como es posible
 - d. Que una de ellas debe ser más larga que la otra
2. ¿Por qué un modelo responde con más fuerza cuando una entrada se parece a lo que sus parámetros han aprendido?
 - a. Porque el producto escalar entre la entrada y los parámetros es máximo cuando ambos coinciden
 - b. Porque los parámetros son siempre flechas más largas que las entradas
 - c. Porque el modelo copia la entrada en sus parámetros
 - d. Porque las matrices solo pueden multiplicarse por matrices que coincidan
3. ¿Qué calcula en realidad multiplicar dos matrices?
 - a. Las longitudes de todas las filas, una tras otra
 - b. Cada número de una tabla por el número en el mismo lugar de la otra
 - c. Muchos productos escalares a la vez, cada fila de la primera contra cada columna de la segunda
 - d. La exponencial de cada número de la tabla

4. En la visión del libro de las funciones lineales como matrices, ¿a qué corresponde multiplicar dos matrices?
- a. A sumar las dos funciones
 - b. A aplicar una función tras otra
 - c. A deshacer la primera función con la segunda
 - d. A aplicar ambas funciones a cada número por separado

Respuestas

1. **b)** El producto escalar es grande cuando las flechas apuntan en la misma dirección, cero en ángulo recto y negativo cuando apuntan en direcciones opuestas.
2. **a)** El producto escalar es máximo cuando la segunda flecha coincide con la primera, así que cuando la entrada resuena con los parámetros la respuesta llega a su pico: coincidencia con plantilla.
3. **c)** La multiplicación de matrices no es más que muchos productos escalares hechos en una sola operación, cada fila de una tabla contra cada columna de la otra.
4. **b)** Una matriz por un vector aplica la función, y una matriz por una matriz aplica una función tras otra.

Ejemplos resueltos

Un producto escalar a mano Halle el producto escalar de los dos vectores 0.1, 0, -0.3 y -4.0, 0.05, 0.1, el ejemplo que el propio libro resuelve.

Multiplique los números en los mismos lugares: 0.1 por -4.0 da -0.4.

Siguiente par: 0 por 0.05 da 0.

Último par: -0.3 por 0.1 da -0.03.

Sume los tres resultados: $-0.4 + 0 - 0.03 = -0.43$, un solo número; es negativo, así que las flechas apuntan más en direcciones distintas que en la misma.

Respuesta: -0.43

Ejercicios

1. Calcule el producto escalar de los vectores 1, 2 y 3, 4. Luego calcúlelo para 1, 0 y 0, 1, y diga qué le indica la segunda respuesta sobre las dos flechas.

Pista: Multiplique los números en los mismos lugares y sume los resultados.

Respuesta: $1 \times 3 + 2 \times 4 = 11$. Para 1, 0 y 0, 1 el resultado es 0, así que las dos flechas están en ángulo recto: direcciones sin relación.

2. El libro advierte que las bibliotecas estiran en silencio matrices de formas que no encajan. En su ejemplo, una columna de 4 números se suma a una lista simple de 4 números. ¿Qué forma sale, y por qué es una trampa?

Pista: Piense en qué pasa si cada número de la columna se combina con cada número de la lista.

Respuesta: El resultado es una tabla de 4 por 4 en vez de una lista de 4 números, porque la biblioteca estira ambas matrices para que encajen. Nada falla, así que el error puede pasar inadvertido.

3. Explique con sus propias palabras por qué una aplicación de fotos podría usar productos escalares para encontrar imágenes que se parezcan a su perro.

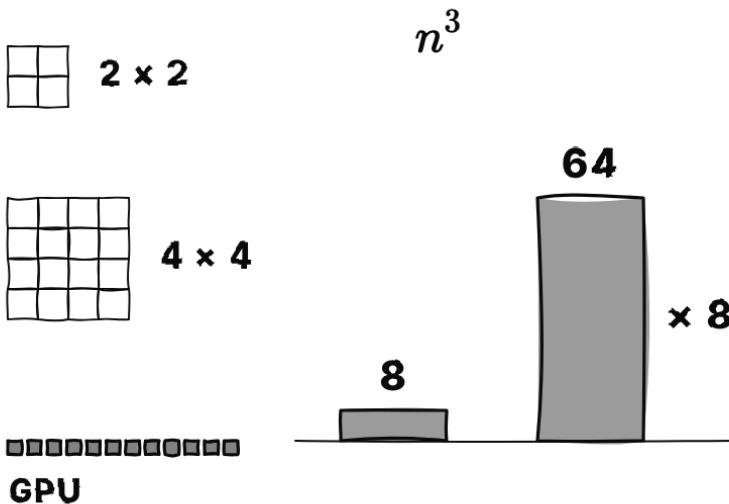
Pista: Piense en cada imagen convertida en una flecha de números.

Respuesta: Si cada imagen se convierte en una flecha de números, las imágenes parecidas dan flechas que apuntan en direcciones similares. Su

producto escalar es entonces grande, así que la aplicación puede ordenar las imágenes según lo grande que sea su producto escalar con la flecha de su perro.

Cómo crecen los costes y hacia dónde es cuesta arriba

Multiplique a mano dos tablas pequeñas de números, de dos filas por dos columnas cada una: necesita ocho multiplicaciones. Ahora duplique el tamaño a cuatro por cuatro, y quizá espere dieciséis. Cuéntelas y encontrará sesenta y cuatro, ocho veces más, porque duplicar cada lado de la tarea multiplica el trabajo tres veces seguidas. Esa sorpresa es la entrada del libro a una pregunta que se hace todo ingeniero de IA: ¿cómo crece el coste de una operación cuando crecen los datos?

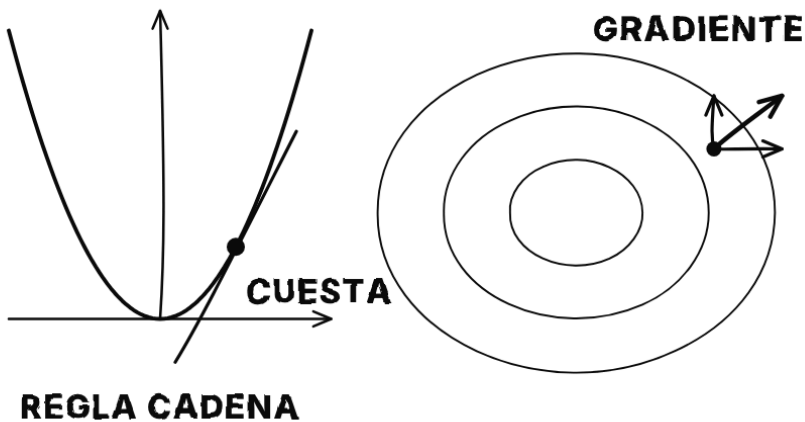


Doble tamaño, ocho veces el trabajo

El libro separa el tiempo, el número de pasos, del espacio, la memoria necesaria, y describe su crecimiento con una notación abreviada que usan los informáticos, que conserva solo lo que domina cuando los tamaños se hacen grandes. Con esa medida, la multiplicación de matrices corriente crece con el cubo del tamaño, que es el salto de ocho veces que acaba de contar. Existen métodos más ingeniosos con crecimiento más lento, pero rara vez se usan, porque son difíciles de repartir entre los miles de pequeños trabajadores de un chip gráfico. Tener muchos trabajadores solo divide el tiempo por una cantidad fija, y mover datos de un lado a otro puede costar más que la aritmética misma, así que en la práctica se mide el código real en vez de confiar solo en la fórmula. Por eso entrenar el modelo detrás de un asistente de chat se planifica tanto en torno a la memoria y los chips como en torno a las matemáticas.

Más allá de las tablas, el libro solo necesita dos extensiones: hacer el mismo producto de matrices para todo un lote de tablas a la vez, y un producto escalar sobre matrices de cualquier forma. También muestra una notación abreviada opcional llamada einsum, donde una cadena corta de letras nombra qué ejes se combinan y cuáles se suman, de modo que la misma línea funciona en distintas bibliotecas.

Luego llega el corazón del capítulo, la derivada. Para una función de un número, la derivada es la pendiente de la recta tangente en un punto: le dice si la función sube o baja si da un pasito a la derecha. Unas pocas reglas permiten hallar derivadas de sumas, productos y, lo más importante, de una función aplicada tras otra, donde las pendientes simplemente se multiplican; esta regla de la cadena sostendrá buena parte del libro.

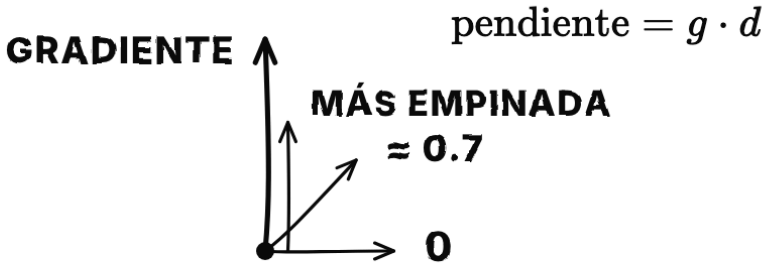


Pendiente, luego gradiente

Una función de muchos números es como un paisaje, y no hay una sola pendiente, porque puede dar pasos en infinitas direcciones. Dar pasos a lo largo de un eje a la vez da una pendiente por número, llamada derivada parcial, y apilarlas todas da el gradiente. El resultado clave del libro es que la pendiente en cualquier otra dirección es simplemente el producto escalar del gradiente con esa dirección, así que conocer el gradiente significa conocer todas las pendientes a la vez. Cuando se entrena una aplicación de fotos, el gradiente dice, para cada uno de sus millones de parámetros, cómo cambia el error si se retoca ese parámetro.

Pruébalo

Párese en un césped en pendiente o en una rampa y dé un pasito en varias direcciones: recto hacia arriba, de lado, en diagonal. Debería sentir la subida más empinada en una dirección, ninguna subida en ángulo recto con ella, y subidas parciales entre medias. El gradiente es la flecha de esa dirección más empinada, y la pendiente de cualquier otro paso se sigue de cuánto se alinea con esa flecha.



n^3 **REGLA CADENA**

Todas las pendientes de una flecha

Quedan fuera aquí la definición formal de la notación de crecimiento, la sintaxis de einsum y las reglas exactas de derivación; están en las

secciones «Complejidad computacional», «Notación de Einstein» y «Derivadas de funciones escalares».

Para recordar

El coste de una operación se juzga por cómo crece con el tamaño; la multiplicación de matrices corriente crece con el cubo del tamaño, así que duplicar las tablas supone ocho veces más trabajo. La velocidad real depende también del hardware, del trabajo en paralelo y del movimiento de datos, y por eso el código se mide, no solo se analiza. Una derivada es la pendiente de una función en un punto, y la regla de la cadena multiplica las pendientes cuando una función sigue a otra. El gradiente reúne la pendiente a lo largo de cada eje, y la pendiente en cualquier dirección es su producto escalar con el gradiente, así que el gradiente por sí solo dice cómo cambia una función en todas las direcciones.

Preguntas

1. ¿Por qué duplicar el tamaño de dos tablas cuadradas hace la multiplicación de matrices corriente unas ocho veces más lenta?
 - a. Porque el trabajo crece con el cubo del tamaño, y dos al cubo es ocho
 - b. Porque cada número tiene que copiarse ocho veces
 - c. Porque el chip gráfico se ralentiza con tablas más grandes
 - d. Porque la tabla resultado se vuelve ocho veces más ancha

2. ¿Por qué los métodos de multiplicación más ingeniosos, con crecimiento más lento, rara vez se usan en la práctica?
 - a. Porque dan respuestas erróneas para tablas grandes
 - b. Porque son difíciles de repartir entre los muchos pequeños trabajadores de un chip gráfico
 - c. Porque necesitan tablas de una forma especial
 - d. Porque nadie los ha escrito todavía
3. Una función de muchos números tiene pendiente en infinitas direcciones. ¿Qué basta para conocerlas todas?
 - a. Solo la pendiente a lo largo del primer eje
 - b. El valor de la función en muchos puntos al azar
 - c. El gradiente: la pendiente en cualquier dirección es su producto escalar con esa dirección
 - d. La segunda derivada de la función
4. En un césped en pendiente, ¿en qué dirección un pasito no da ninguna subida?
 - a. Recto a lo largo de la dirección más empinada
 - b. En ángulo recto con la dirección más empinada
 - c. Exactamente opuesta a la dirección más empinada
 - d. En diagonal entre arriba y de lado

Respuestas

1. a) La multiplicación de matrices corriente crece con el cubo del tamaño, así que duplicar cada lado multiplica el trabajo por dos tres veces seguidas: ocho.

2. b) Existen métodos con crecimiento más lento, pero son difíciles de repartir entre los miles de pequeños trabajadores de un chip gráfico, así que pierden en el uso real.

3. c) La pendiente en cualquier dirección es el producto escalar del gradiente con esa dirección, así que conocer el gradiente significa conocer todas las pendientes a la vez.

4. b) El gradiente apunta a lo largo de la subida más empinada, y un paso en ángulo recto con él no se alinea con él en absoluto, así que su pendiente es cero.

Ejemplos resueltos

Contar el trabajo de un producto de matrices Una tabla de 3 filas y 4 columnas se multiplica por una tabla de 4 filas y 5 columnas. ¿Cuántas multiplicaciones individuales necesita el método corriente?

El resultado tiene 3 filas y 5 columnas, así que hay que calcular 15 números.

Cada uno de ellos es un producto escalar de una fila de longitud 4 con una columna de longitud 4, que requiere 4 multiplicaciones.

Así que el total es 15 por 4, que es 60, el producto de los tres tamaños 3, 4 y 5.

Respuesta: 60 multiplicaciones

Ejercicios

1. El ejemplo de código del libro multiplica un lote de 4 tablas, cada una de 5 por 2, por un lote de 4 tablas, cada una de 2 por 3. ¿Cuál es la forma del resultado?

Pista: Cada uno de los 4 pares se multiplica por separado, como tablas corrientes.

Respuesta: Un lote de 4 tablas, cada una de 5 por 3: en cada par, 5 por 2 por 2 por 3 da 5 por 3.

2. El libro dibuja la función x al cuadrado menos $1.5x$. Su pendiente en un punto x es $2x$ menos 1.5 . ¿La función sube o baja en $x = 1$, y en $x = 0$?

Pista: Una pendiente positiva significa que sube hacia la derecha, una negativa que baja.

Respuesta: En $x = 1$ la pendiente es $2 - 1.5 = 0.5$, positiva, así que la función sube; en $x = 0$ la pendiente es -1.5 , negativa, así que baja.

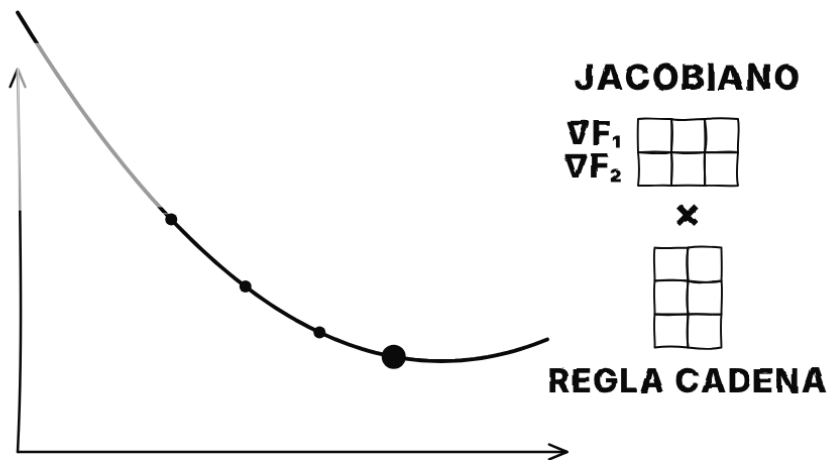
3. En cierto punto una función de dos números tiene el gradiente $2, -1$. ¿Cuál es la pendiente si da un paso en la dirección $1, 1$?

Pista: Tome el producto escalar del gradiente con la dirección.

Respuesta: $2 \times 1 + (-1) \times 1 = 1$, así que la función sube con pendiente 1 en esa dirección.

Bajando la cuesta en la niebla

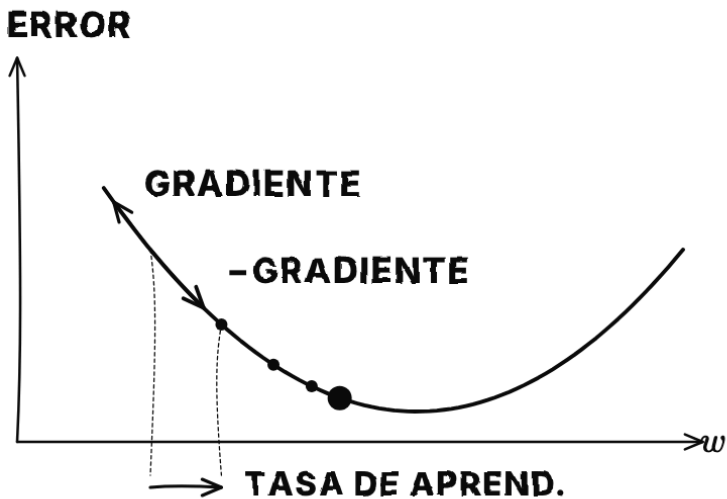
Imagine que está en la ladera de una colina con niebla espesa, sin ver más que sus propios pies, y quiere llegar al punto más bajo del valle. No puede ver el valle, pero sí sentir hacia dónde se inclina el suelo bajo sus zapatos. Así que da un pasito en la dirección en que cae más empinado, vuelve a sentir, y vuelve a dar un paso. Sorprendentemente a menudo, esta receta a ciegas le lleva abajo, y es exactamente como se entrena casi todo modelo de IA.



Sintiendo la pendiente en la niebla

Primero el libro amplía una vez más la idea de pendiente. Cuando una función devuelve varios números en vez de uno, cada salida tiene su propio gradiente, y apilarlos da una tabla llamada jacobiano. La regla de la cadena se mantiene: para una función aplicada tras otra, sus jacobianos simplemente se multiplican, como tablas. Como la pendiente de una sola curva, el jacobiano es la mejor aproximación en línea recta de cómo se comporta la función justo alrededor de un punto. El libro añade una advertencia para más adelante: para la capa más simple de un modelo, estas tablas pueden ser enormes y estar llenas de números repetidos, así que en la práctica nunca se guardan completas, un truco que explica el capítulo sobre diferenciación automática.

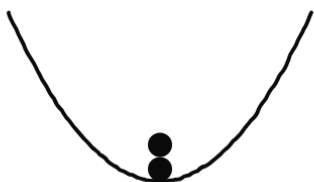
Ahora, la niebla. Los parámetros del modelo son su posición, y un número que mide lo mal que rinde el modelo es su altura; entrenar significa encontrar la posición donde ese número es mínimo. Salvo en casos raros, no hay fórmula que salte directamente a la respuesta, así que se avanza paso a paso, y cada paso tiene una longitud y una dirección. En aprendizaje automático la longitud se llama tasa de aprendizaje. Cualquier dirección que baje cuenta como dirección de descenso, y como la pendiente en una dirección es su producto escalar con el gradiente, el camino más empinado hacia abajo es justo el opuesto al gradiente. Dar pasos una y otra vez contra el gradiente, con pasos lo bastante pequeños, es el descenso de gradiente. Cuando se entrena un asistente de chat, cada uno de sus incontables pasos de entrenamiento es uno de esos pasitos cuesta abajo para todos sus parámetros a la vez.



Un paso contra el gradiente

¿Dónde termina la caminata? Se detiene donde el suelo es plano, un punto estacionario, pero el suelo plano puede ser el fondo de un valle, la cima de una colina, o una silla de montar como un paso de montaña que cae hacia un lado y sube hacia el otro. En general el descenso de gradiente solo promete llegar a algún punto plano, y a cuál depende de dónde empiece. Para un paisaje con forma de cuenco, llamado convexo, toda pendiente lleva al mismo punto más bajo, así que se alcanza el mínimo global desde cualquier sitio. Encontrar con certeza el punto más bajo de un paisaje lleno de baches significaría empezar desde todos los lugares posibles, lo cual no tiene esperanza.

CONVEXO



HUEVERA



Cuenco contra huevera

Pruébalo

Deje caer una canica en una ensaladera redonda desde cualquier lado: siempre acaba en el mismo punto del fondo. Ahora déjela caer sobre una huevera: se asienta en el hueco más cercano a donde cayó. El cuenco es un problema convexo; la huevera es el aspecto que tiene entrenar modelos reales.

El libro termina con el giro: toda red neuronal útil es una huevera, no un cuenco, a diferencia de algunos métodos de aprendizaje más antiguos que mantenían sus problemas convexos. Y sin embargo, partiendo de un lugar sensato, el descenso de gradiente en estas redes

aterriza en puntos que funcionan muy bien en la práctica. Quedan fuera aquí las definiciones formales del jacobiano, de los mínimos y de la convexidad, y la demostración de que el descenso más empinado es opuesto al gradiente; están en las secciones «Jacobianos», «Descenso de gradiente» y «Convergencia del descenso de gradiente».

Para recordar

Para una función con muchas salidas, los gradientes se apilan en un jacobiano, y los jacobianos de funciones encadenadas se multiplican como tablas. Entrenar significa encontrar los parámetros donde una medida del error es mínima, mediante pequeños pasos repetidos. El camino más empinado hacia abajo es opuesto al gradiente, y caminar en esa dirección con una tasa de aprendizaje pequeña es el descenso de gradiente. En un problema convexo, con forma de cuenco, llega al único punto más bajo desde cualquier sitio; en los paisajes llenos de baches de las redes neuronales solo llega a algún punto plano, que en la práctica funciona sorprendentemente bien.

Preguntas

1. ¿Por qué el camino más empinado hacia abajo en un paisaje es exactamente opuesto al gradiente?
 - a. Porque el gradiente siempre apunta al mínimo global

- b. Porque la pendiente en una dirección es su producto escalar con el gradiente, que es lo más negativo posible cuando la dirección es opuesta a él
 - c. Porque la tasa de aprendizaje invierte el gradiente automáticamente
 - d. Porque el gradiente es cero en todas partes salvo en el fondo
2. El descenso de gradiente se detiene en suelo plano. ¿Por qué eso no garantiza el punto más bajo?
- a. Porque el suelo plano también puede ser una cima o una silla, y qué punto plano se alcanza depende del inicio
 - b. Porque el descenso de gradiente nunca se detiene en suelo plano
 - c. Porque la tasa de aprendizaje crece cerca del fondo
 - d. Porque el suelo plano solo existe en problemas convexos
3. ¿Qué tiene de especial un problema convexo, con forma de cuenco?
- a. Tiene muchos huecos, así que el inicio decide la respuesta
 - b. El descenso de gradiente no puede usarse en él
 - c. Toda pendiente lleva al mismo punto más bajo, así que el mínimo global se alcanza desde cualquier sitio
 - d. Su gradiente es el mismo en todas partes
4. Las redes neuronales no son convexas. ¿Qué dice el libro sobre entrenarlas de todos modos?
- a. Nunca funciona sin una fórmula cerrada
 - b. Partiendo de un lugar sensato, el descenso de gradiente aterriza en puntos que funcionan muy bien en la práctica

- c. Solo funciona si la red se convierte primero en un problema convexo
- d. Siempre alcanza el mínimo global tras suficientes pasos

Respuestas

1. **b)** La pendiente en una dirección es el producto escalar de esa dirección con el gradiente, y es mínima cuando ambos apuntan en direcciones opuestas.
2. **a)** Los puntos planos pueden ser fondos de valle, cimas o sillas, y en general el descenso de gradiente solo llega a algún punto plano, según dónde empiece.
3. **c)** Para un paisaje convexo, con forma de cuenco, toda pendiente lleva al mismo punto más bajo, así que el descenso de gradiente alcanza el mínimo global desde cualquier inicio.
4. **b)** Toda red neuronal útil es una huevera, no un cuenco, y sin embargo desde un inicio sensato el descenso de gradiente aterriza en puntos que rinden muy bien.

Ejemplos resueltos

Una aproximación en línea recta cerca de un punto El libro toma la curva x al cuadrado menos $1.5x$, cuya pendiente es $2x$ menos 1.5 , y la reemplaza cerca de $x = 0.5$ por su recta tangente. Compare la curva y la recta en $x = 0.51$.

En $x = 0.5$ el valor de la curva es $0.25 - 0.75 = -0.5$, y su pendiente es $1 - 1.5 = -0.5$.

La recta tangente en 0.51 es el valor más la pendiente por el pasito: $-0.5 + (-0.5 \times 0.01) = -0.505$.

La curva misma en 0.51 da $0.2601 - 0.765 = -0.5049$.

Las dos difieren solo en una diezmilésima: muy cerca del punto, la recta es una aproximación excelente.

Respuesta: Curva -0.5049 , recta -0.5050

Ejercicios

1. Ejecute dos pasos de descenso de gradiente en la función x al cuadrado, cuya pendiente es $2x$, empezando en $x = 3$ con una tasa de aprendizaje de 0.1 .

Pista: Cada paso: el nuevo x es igual al x anterior menos la tasa de aprendizaje por la pendiente.

Respuesta: Paso 1: pendiente 6, así que $x = 3 - 0.6 = 2.4$. Paso 2: pendiente 4.8, así que $x = 2.4 - 0.48 = 1.92$. Cada paso acerca al fondo en 0.

2. Una función recibe 5 números y devuelve 3 números. ¿Qué forma tiene su jacobiano, y qué contiene cada fila?

Pista: Cada salida tiene su propio gradiente.

Respuesta: Es una tabla de 3 filas y 5 columnas: cada fila es el gradiente de una salida respecto a las 5 entradas.

3. Usando la imagen de la canica, explique por qué el punto de partida importa al entrenar una red neuronal pero no al minimizar una función con forma de cuenco.

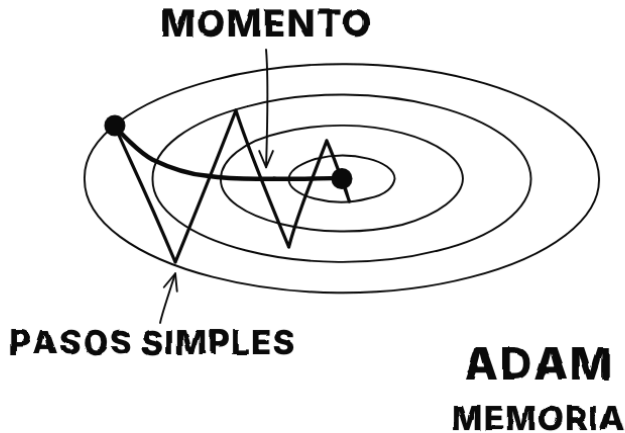
Pista: Compare la ensaladera con la huevera.

Respuesta: En un cuenco toda pendiente lleva a un solo fondo, así que la canica acaba allí desde cualquier sitio. El paisaje de error de una red neuronal es como una huevera con muchos huecos y puntos planos, así que el descenso de gradiente se asienta en el que esté cerca de su inicio.

El momento, y de dónde salen los ejemplos

Haga rodar una pelota de ping-pong y luego una canica pesada por una pendiente llena de baches y ondulaciones, como una hoja de cartón arrugada apoyada en un extremo. La pelota ligera tiembla, se engancha en cada ondulación y cambia de dirección en cada bache. La canica pesada conserva buena parte de su movimiento anterior, pasa por encima de las ondulaciones pequeñas y llega al fondo por un camino más suave. Esa diferencia es la primera idea de este tramo del libro, y la segunda trata del terreno mismo: los ejemplos de los que aprende un modelo.

DESDE ARRIBA



Pasos temblorosos contra momento

Como vimos, el descenso de gradiente da pasos contra el gradiente, pero el gradiente solo dice el camino más empinado hacia abajo justo donde usted está, y para modelos grandes puede ser muy ruidoso. El momento da algo de peso a la caminata: cada paso conserva parte de la dirección del paso anterior, mientras los pasos más antiguos se desvanecen. El camino se vuelve más suave y el entrenamiento suele llegar antes. Otro truco da a cada parámetro su propio tamaño de paso, según lo grandes que hayan sido sus gradientes, y un método popular llamado Adam combina tales ideas; es la opción por defecto en la mayoría de los frameworks y es bastante tolerante con sus ajustes. El precio es memoria, porque el método debe recordar los pasos anteriores. Los asistentes y modelos de fotos que usted usa casi seguro se entrenaron con Adam o uno de sus parientes cercanos.

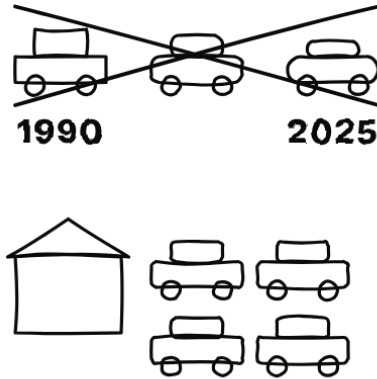
El libro hace entonces una pausa para practicar en vez de hacer ejercicios. Sugiere ganar fluidez en NumPy, la biblioteca básica para matrices, y luego en JAX y PyTorch, que se parecen mucho a NumPy pero además corren en chips gráficos y calculan los gradientes por usted. La primera tarea es escribir a mano el descenso de gradiente para un paisaje ondulado con muchas colinas y hondonadas, con y sin momento, y observar los caminos desde distintos puntos de partida. El libro señala, con una sonrisa, que se le pidió a ChatGPT que inventara esa función ondulada, y que nada más en el libro está generado por máquina.

El capítulo 3 pregunta entonces dónde empieza el aprendizaje: con un conjunto de datos. Algunas reglas son imposibles de escribir a mano, como reconocer objetos en fotos, pero los ejemplos del comportamiento deseado son fáciles de reunir, así que un conjunto de datos es simplemente muchos pares de una entrada y la salida correcta. El libro subraya una suposición que decide el éxito en silencio: los ejemplos deben venir de una sola fuente estable y recogerse sin sesgo. Fotos de coches reunidas a lo largo de varias décadas mezclan modelos que ya no existen, y fotos tomadas junto al concesionario de una marca sobrerrepresentan esa marca; un conjunto de datos de Italia puede servir bien a Roma y mal a Tokio. Incluso cuando un modelo ya está en marcha, el mundo se aleja de sus datos, y un recomendador cambia el mismo comportamiento del que aprende, así que la comprobación nunca termina.

UNA FUENTE, SIN SESGO PARES

 → **COCHE**
 → **PERRO**
 → **COCHE**
 → **GATO**

REVISAR ↻



Un conjunto de datos debe ser justo

No toda tarea trae respuestas correctas adjuntas. Sin ellas, en el aprendizaje no supervisado, se agrupan cosas parecidas o se buscan las coincidencias más cercanas, y ayuda un modelo ya entrenado en otra tarea, porque sus números internos colocan las cosas parecidas cerca unas de otras. El aprendizaje autosupervisado inventa sus propias respuestas a partir de datos en bruto, por ejemplo prediciendo cómo continúa un fragmento de texto, y así es como se entrenan primero los grandes modelos de lenguaje. A un modelo fundacional así se le puede pedir luego una tarea directamente, mostrarle unos pocos ejemplos en la instrucción, o ajustarlo por descenso de gradiente sobre un conjunto de datos pequeño, a veces cambiando solo una pequeña parte de sus parámetros. Cuando pega dos correos de muestra en un asistente de chat y pide un tercero en el

mismo estilo, lo está usando con unos pocos ejemplos, sin cambiar ninguno de sus parámetros. Quedan fuera aquí la fórmula del momento, los detalles del ejercicio de programación y la larga lista de otras modalidades de aprendizaje; están en las secciones «Acelerar el descenso de gradiente», «De la teoría a la práctica» y «Variantes del aprendizaje supervisado».

AUTOSUPERVISADO

EL SOL CAE EN

PREGUNTAR EJEMPLOS AJUSTE FINO

P EJ. EJ. P

MODELO

MODELO



Sin etiquetas, luego tres usos

Para recordar

El momento permite que cada paso del descenso de gradiente conserve parte de la dirección anterior, lo que suaviza el camino ruidoso y acelera el entrenamiento, a costa de algo de memoria; Adam, la opción habitual por defecto, añade un tamaño de paso para cada parámetro. Un conjunto de datos es una colección de pares de entrada y salida, usada cuando la regla misma no puede escribirse. Debe venir de una sola fuente estable y recogerse sin sesgo, y esto hay que comprobarlo mientras el modelo esté en uso. Sin respuestas etiquetadas, los modelos pueden agrupar, buscar o aprender de tareas que inventan ellos mismos, que es como empiezan los grandes modelos de lenguaje, y luego pueden usarse directamente, con ejemplos en la instrucción, o ajustarse.

Preguntas

1. ¿Por qué el momento hace más suave el camino del descenso de gradiente?
 - a. Porque hace cada paso más largo
 - b. Porque cada paso conserva parte de la dirección del paso anterior, así que los gradientes ruidosos aislados importan menos
 - c. Porque elimina el gradiente de la actualización
 - d. Porque solo se mueve cuando el gradiente es cero
2. ¿Cuál es el precio de usar momento o Adam?

- a. Más memoria, porque hay que recordar los pasos anteriores
 - b. Necesitan las segundas derivadas de la función
 - c. Solo pueden usarse en problemas convexos
 - d. Necesitan un conjunto de datos nuevo en cada paso
3. Un conjunto de datos de reconocimiento de coches se construye solo con fotos tomadas frente al concesionario de una marca. ¿Qué suposición rompe esto?
- a. Que el conjunto de datos es lo bastante pequeño para caber en memoria
 - b. Que todas las fotos tienen el mismo tamaño
 - c. Que los ejemplos se recogen sin sesgo de toda la fuente con la que se encontrará el modelo
 - d. Que el modelo usa momento
4. ¿Cómo se entrenan primero los grandes modelos de lenguaje, según esta parte?
- a. En una tarea autosupervisada inventada a partir de texto en bruto, como predecir cómo continúa un fragmento de texto
 - b. Solo con pares de preguntas y respuestas escritos a mano
 - c. Agrupando frases en clústeres sin ninguna tarea
 - d. Ajustando unos pocos parámetros sobre un conjunto de datos pequeño

Respuestas

1. **b)** El momento da algo de peso a la caminata: cada paso conserva parte de la dirección anterior mientras los pasos más antiguos se desvanecen, así que el camino es más suave.

2. a) El método debe recordar los pasos anteriores, así que necesita más memoria.

3. c) Los ejemplos deben venir de una sola fuente estable y recogerse sin sesgo; las fotos junto a un concesionario sobrerrepresentan esa marca.

4. a) El aprendizaje autosupervisado inventa sus propias respuestas a partir de datos en bruto, por ejemplo prediciendo cómo continúa el texto, y así empiezan los grandes modelos de lenguaje.

Ejemplos resueltos

Qué tan rápido se desvanecen los pasos antiguos Con momento, cada paso conserva una parte del anterior; la figura del libro usa una parte de 0.3. ¿Cuánto peso llevan en el paso actual los gradientes de uno, dos y tres pasos atrás?

El gradiente más nuevo entra en el paso completo: peso 1.

El paso anterior se conserva con una parte de 0.3: peso 0.3.

El anterior a ese se ha conservado dos veces: 0.3 por 0.3, que es 0.09.

Los gradientes más antiguos siguen encogiéndose por 0.3 cada vez, así que la caminata solo recuerda su dirección reciente.

Respuesta: 1, 0.3 y 0.09

Ejercicios

1. Un modelo que reconoce modelos de coches se entrenó con fotos recogidas en Italia. Explique, como hace el libro, por qué puede funcionar en Milán pero fallar en Tokio, y por qué la comprobación nunca termina.

Pista: Pregúntese si los ejemplos vienen de la misma fuente con la que se encontrará el modelo.

Respuesta: El conjunto de datos refleja los coches comunes en Italia, así que en Tokio el modelo se encuentra con una mezcla distinta de coches que rara vez vio. Incluso en Italia los coches de la calle cambian con los años, así que los datos se alejan de lo que el modelo aprendió y su rendimiento debe seguir comprobándose.

2. Usted da a un asistente de chat dos correos de muestra en su instrucción y pide un tercero en el mismo estilo. ¿Es esto ajuste fino? Explique la diferencia.

Pista: Pregúntese si cambia algún parámetro del modelo.

Respuesta: No, es una instrucción con unos pocos ejemplos: los ejemplos están en la instrucción y ningún parámetro cambia. El ajuste fino ejecutaría descenso de gradiente sobre un conjunto de datos pequeño y cambiaría de verdad los parámetros del modelo.

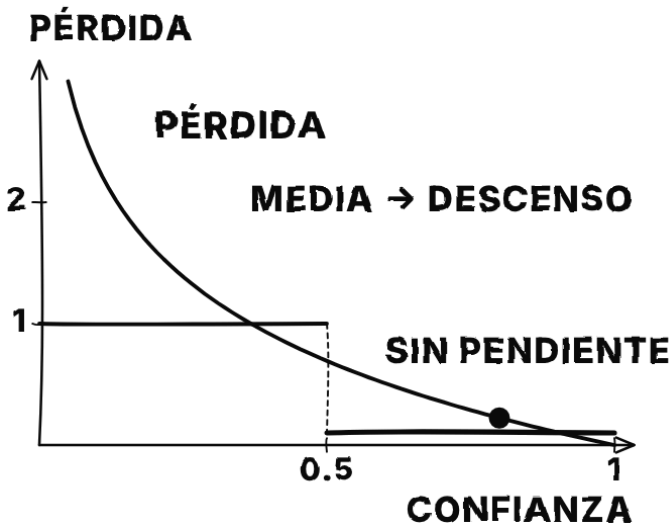
3. ¿Por qué un modelo entrenado para clasificar imágenes puede ayudarle a buscar imágenes parecidas en una colección de fotos, aunque nadie las haya etiquetado?

Pista: Piense en lo que hacen sus números internos con imágenes parecidas.

Respuesta: Los números internos del modelo entrenado colocan las imágenes parecidas cerca unas de otras, así que comparar esos números encuentra las imágenes más similares sin ninguna etiqueta.

Una puntuación que bajar, y un examen que no ha visto

Dele a un amigo el examen teórico de conducir del año pasado con las respuestas, y déjele estudiarlo todo lo que quiera. Luego entréguele exactamente el mismo examen: saca una nota perfecta. Ahora dele un examen nuevo con preguntas distintas, y su nota puede desplomarse, porque aprendió las respuestas, no las normas de circulación. Ese pequeño experimento contiene todo este tramo del libro: cómo el aprendizaje se convierte en un número que hacer pequeño, y por qué hacerlo pequeño en sus propios ejemplos no es la meta real.



Una puntuación que puede bajarse

El número es la pérdida: compara la predicción del modelo con la respuesta correcta y da una sola puntuación, donde más bajo significa mejor. Debe ser diferenciable, porque el plan es promediar la pérdida sobre todo el conjunto de datos y bajar ese promedio por descenso de gradiente sobre los parámetros, una receta que el libro llama minimización del riesgo empírico. La pérdida es donde usted escribe lo que le importa, y además debe poder usarse para optimizar. Una simple puntuación de acierto o error falla aquí, porque se mantiene plana casi en todas partes, así que su gradiente es cero y el descenso de gradiente nunca se mueve; en su lugar se usan puntuaciones más suaves que además premian acertar con confianza.

Luego viene una trampa. La pérdida media en los ejemplos de entrenamiento siempre puede hacerse perfecta con un truco: una tabla de consulta que guarda cada ejemplo y responde con él, y da algo por defecto para cualquier otra cosa. Esa tabla ha memorizado y no ha aprendido nada, exactamente como su amigo con el examen del año pasado. Lo que realmente queremos es una pérdida baja en entradas futuras que el modelo nunca ha visto, llamada riesgo esperado, que nunca puede calcularse porque no podemos reunir todos los ejemplos posibles. La diferencia entre ambos es la brecha de generalización, y de un modelo con una brecha grande se dice que sobreajusta. En la práctica se guarda un conjunto de prueba separado, nunca usado en el entrenamiento, y se compara la pérdida allí con la pérdida de entrenamiento. Por eso los creadores de una aplicación de fotos la juzgan con imágenes con las que nunca se entrenó, no con las que aprendió.

TABLA

x_1	y_1
x_2	y_2
x_3	y_3

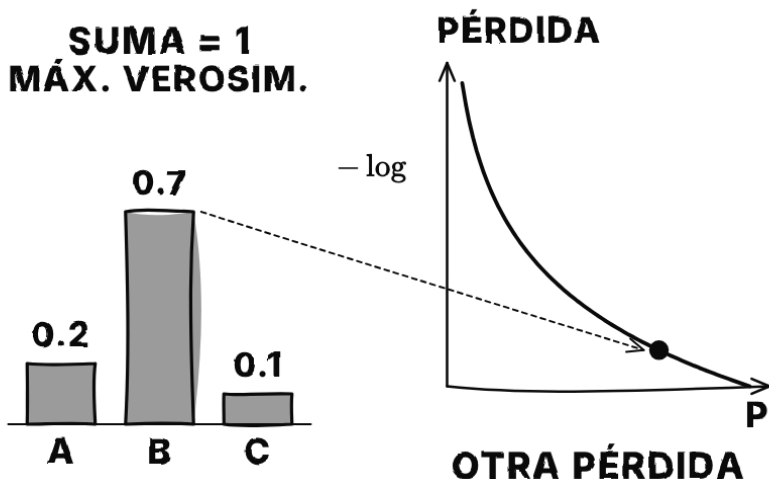
MEMORIZADO

SOBREAJUSTE



Memorizado contra aprendido

¿Cómo se elige una buena pérdida? El libro ofrece una segunda visión, más fundamentada: dejar que el modelo prediga no una respuesta sino una probabilidad para cada respuesta posible. Para una elección entre tres clases el modelo da tres probabilidades que suman uno, y para un número puede dar un valor más probable junto con lo incierto que es. Cuando un asistente de chat escribe, hace exactamente esto para cada palabra siguiente: una probabilidad para cada palabra que podría usar. El entrenamiento sigue entonces el principio de máxima verosimilitud: elegir el modelo bajo el cual el conjunto de datos que realmente observó es lo más probable posible.



Probabilidades, luego $-\log$

Las dos visiones se encuentran con elegancia. Tomar logaritmos y cambiar el signo convierte maximizar esa probabilidad en minimizar una suma de logaritmos negativos de probabilidades, que es de nuevo simplemente una pérdida media. Así que toda pérdida usada en la práctica puede verse como surgida de una elección de probabilidades, y el libro le pide tener presentes ambas visiones. Quedan fuera aquí las fórmulas de los riesgos, la pérdida hinge y los modelos gaussiano y categórico; están en las secciones «3.2 Funciones de pérdida», «Riesgo esperado y sobreajuste» y «Máxima verosimilitud».

Para recordar

Una pérdida es una puntuación diferenciable de cuán errónea es una predicción, y el entrenamiento baja su promedio sobre el conjunto de datos por descenso de gradiente. Memorizar los ejemplos de entrenamiento puede hacer perfecto ese promedio siendo a la vez inútil, así que la meta real es una pérdida baja en datos no vistos. Un conjunto de prueba separado mide la brecha entre ambos, y una brecha grande significa sobreajuste. Dejar que el modelo prediga probabilidades y elegir el modelo que hace más probables los datos observados, la máxima verosimilitud, da lo mismo que minimizar el logaritmo negativo de la probabilidad, una pérdida.

Preguntas

1. ¿Por qué una simple puntuación de acierto o error es inútil para entrenar por descenso de gradiente?
 - a. Porque es demasiado cara de calcular
 - b. Porque se mantiene plana casi en todas partes, así que su gradiente es cero y los parámetros nunca se mueven
 - c. Porque solo puede usarse en conjuntos de prueba
 - d. Porque da valores negativos para respuestas correctas
2. Una tabla de consulta que guarda cada ejemplo de entrenamiento obtiene en ellos una pérdida media perfecta. ¿Por qué sigue siendo inútil?

- a. Porque memoriza en vez de aprender, así que no dice nada útil sobre entradas que no ha visto
 - b. Porque las tablas de consulta no pueden guardar números
 - c. Porque su pérdida no es diferenciable
 - d. Porque hace más grande el conjunto de datos
3. ¿Cómo puede saber en la práctica si un modelo sobreajusta?
- a. Entrenándolo el doble de tiempo
 - b. Comprobando que su pérdida de entrenamiento es exactamente cero
 - c. Comparando su pérdida en un conjunto de prueba separado, nunca usado en el entrenamiento, con su pérdida de entrenamiento
 - d. Contando sus parámetros
4. ¿Qué elige el principio de máxima verosimilitud?
- a. El modelo con menos parámetros
 - b. El modelo bajo el cual el conjunto de datos realmente observado es lo más probable posible
 - c. El modelo cuyas predicciones son siempre la clase más probable
 - d. El modelo con el conjunto de prueba más grande

Respuestas

1. **b)** Una puntuación de acierto o error se mantiene plana casi en todas partes, así que su gradiente es cero y el descenso de gradiente nunca se mueve.
2. **a)** La tabla ha memorizado y no ha aprendido nada; la meta real es una pérdida baja en entradas futuras que el modelo nunca ha visto.

3. c) Se guarda un conjunto de prueba separado nunca usado en el entrenamiento y se compara la pérdida allí con la de entrenamiento; una brecha grande significa sobreajuste.

4. b) La máxima verosimilitud elige el modelo bajo el cual el conjunto de datos observado es lo más probable posible.

Ejemplos resueltos

La pérdida hinge en acción Para respuestas que son -1 o $+1$, el libro menciona la pérdida hinge: el mayor entre 0 y 1 menos la respuesta por la predicción. Calcúlela cuando la respuesta correcta es $+1$ y el modelo predice 0.3, luego 2, y cuando la respuesta correcta es -1 y el modelo predice 0.5.

Respuesta $+1$, predicción 0.3: $1 - 0.3 = 0.7$, mayor que 0, así que la pérdida es 0.7; lado correcto, pero sin la confianza suficiente.

Respuesta $+1$, predicción 2: $1 - 2 = -1$, así que la pérdida es 0; correcto y confiado, nada que corregir.

Respuesta -1 , predicción 0.5: la respuesta por la predicción es -0.5 , así que $1 + 0.5 = 1.5$; lado equivocado, pérdida mayor.

A diferencia de una puntuación de acierto o error, esta pérdida cambia suavemente con la predicción, así que su pendiente puede guiar el entrenamiento.

Respuesta: 0.7, 0 y 1.5

Ejercicios

1. Un modelo da a las respuestas correctas de tres ejemplos de entrenamiento las probabilidades 0.9, 0.8 y 0.5. ¿Cuál es la verosimilitud de este pequeño conjunto de datos, y cuál es la suma de los logaritmos naturales negativos de las tres probabilidades?

Pista: Los ejemplos son independientes, así que sus probabilidades se multiplican; el logaritmo convierte el producto en una suma.

Respuesta: La verosimilitud es $0.9 \times 0.8 \times 0.5 = 0.36$. Los logaritmos negativos son aproximadamente 0.105, 0.223 y 0.693, que suman alrededor de 1.02, lo mismo que menos el logaritmo de 0.36. Subir la verosimilitud significa bajar esta suma.

2. Usando el ejemplo del examen teórico de conducir, explique la diferencia entre el riesgo empírico y el riesgo esperado.

Pista: ¿Qué examen es el conjunto de entrenamiento, y cuál es la carretera real?

Respuesta: El riesgo empírico es la nota en el examen del año pasado, los ejemplos con los que se entrenó; el riesgo esperado es la nota sobre todas las preguntas que podrían hacerle alguna vez. Memorizar baja el primero sin bajar el segundo, y el segundo es lo que importa.

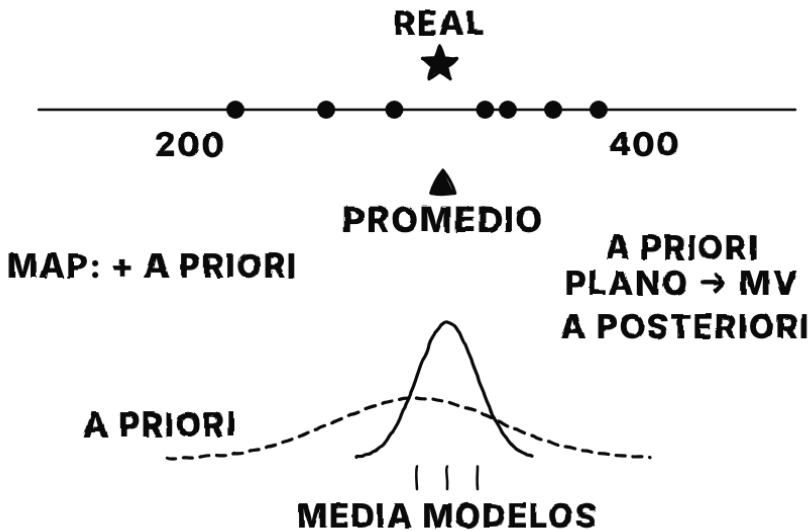
3. ¿Por qué es útil que un modelo pueda predecir un valor más probable junto con lo incierto que es, en vez de un solo número?

Pista: Piense en qué información extra da la segunda salida.

Respuesta: La segunda salida dice cuán confiada es la predicción, así que el modelo puede expresar incertidumbre, y elegir un modelo de probabilidad así puede ser más fácil que inventar una pérdida a mano; la máxima verosimilitud aporta entonces la pérdida.

Muchos modelos posibles, y el más simple

Llene un tarro de caramelos y pida a varios amigos que adivinen cuántos hay. Cada estimación fallará, unas por arriba y otras por abajo. Y sin embargo el promedio de todas las estimaciones suele estar más cerca que la mayoría de las estimaciones individuales, porque sus distintos errores se cancelan en parte. El libro usa la misma idea para dar un paso más allá de la máxima verosimilitud, y luego construye su primerísimo modelo.



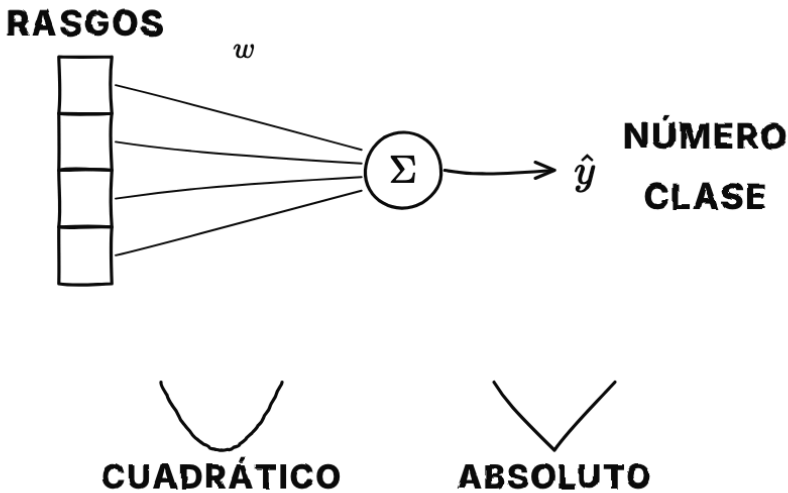
Promediar muchas estimaciones

La máxima verosimilitud elige un único mejor modelo, pero a menudo muchos ajustes distintos de los parámetros explican los datos más o menos igual de bien. La visión bayesiana los conserva todos: antes de ver dato alguno usted declara un a priori, una creencia sobre qué modelos son plausibles, por ejemplo que los modelos con parámetros más pequeños son más estables y por eso preferibles. Tras ver los datos, el teorema de Bayes combina este a priori con la verosimilitud en un a posteriori, una creencia actualizada sobre todos los modelos. Una predicción promedia entonces lo que dicen todos los modelos, cada uno ponderado por lo plausible que es ahora, igual que las estimaciones de los amigos. La pega es que el a posteriori casi nunca puede calcularse con exactitud, así que en la práctica se usan aproximaciones.

Dos resultados secundarios importan mucho. Si conserva solo el modelo más plausible, obtiene la verosimilitud más un término del a priori, y ese término extra tira suavemente de la solución hacia lo que el a priori favorece; esto se llama regularización y ayuda contra el sobreajuste. Con un a priori que no favorece nada, se reduce exactamente a la máxima verosimilitud. Y cuando llegan datos nuevos, el a posteriori de ayer se convierte simplemente en el a priori de hoy, lo que ayuda a un modelo a aprender más sin olvidar lo que ya sabía.

El capítulo 4 construye ahora el bloque más simple de todos, el modelo lineal, del que el libro dice que todo modelo diferenciable es una variación ingeniosa. La entrada es una lista de características, digamos los datos de un cliente de un banco, y la salida es o bien un número, una tarea de regresión, o bien una de unas pocas clases, una tarea de clasificación. Para predecir un número, la pérdida natural es el error cuadrático, y el libro muestra que no es arbitrario: es

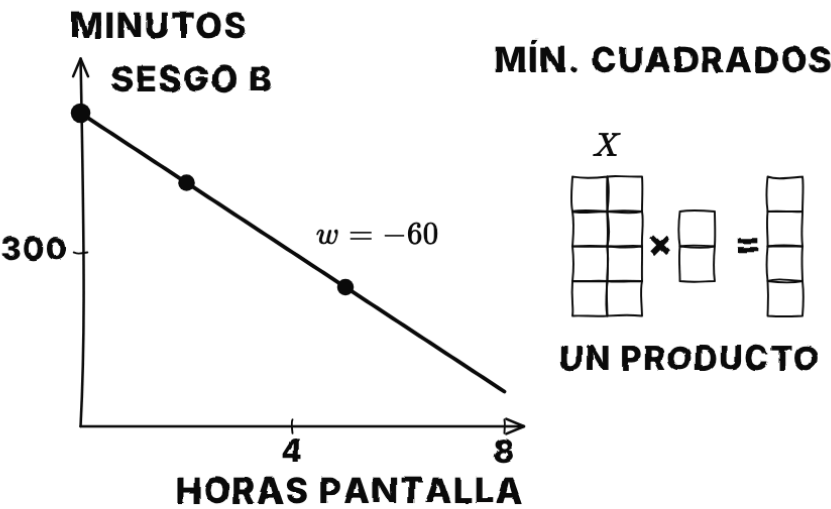
exactamente lo que da la máxima verosimilitud si se supone que los errores siguen una curva en forma de campana. Como elevar al cuadrado deja que unos pocos ejemplos muy equivocados dominen, hay variantes más suaves, como el error absoluto y una mezcla de los dos, y la única esquina aguda del error absoluto no es un obstáculo real para el descenso de gradiente.



El bloque más simple

El modelo lineal en sí da a cada característica un peso, suma todas las características ponderadas y añade un sesgo, la predicción cuando toda característica vale cero. Con una característica dibuja una recta, con dos un plano. Una estimación simple de cuánto durará la batería de su teléfono podría construirse justo así: tantos minutos perdidos por hora de pantalla, tantos por hora de navegación, más una

cantidad fija. Juntar el modelo lineal, el error cuadrático y el promedio sobre el conjunto de datos da el problema de mínimos cuadrados. El libro apila entonces todo el conjunto de datos en una sola tabla, un ejemplo por fila, de modo que todas las predicciones salen de un único producto de matrices, que es exactamente lo que mejor hacen los chips gráficos; barajar las filas solo baraja las predicciones de la misma manera. Quedan fuera aquí la fórmula de Bayes y sus aproximaciones, las fórmulas de las pérdidas y la expresión vectorizada de mínimos cuadrados; están en las secciones «3.3 Aprendizaje bayesiano», «La pérdida cuadrática y variantes» y «El modelo de mínimos cuadrados».



Una recta con sesgo

Para recordar

La visión bayesiana mantiene una creencia sobre todos los modelos, la actualiza con los datos mediante el teorema de Bayes y predice promediando los modelos, aunque en la práctica esto debe aproximarse. Conservar solo el modelo más plausible añade un término de regularización a la máxima verosimilitud, que tira hacia el a priori y combate el sobreajuste. El modelo lineal pondera cada característica, las suma con un sesgo, y es el bloque básico de todos los modelos diferenciables. El error cuadrático se deduce de la máxima verosimilitud con errores en forma de campana; junto con el modelo lineal da los mínimos cuadrados, calculados para todo el conjunto de datos a la vez con productos de matrices.

Preguntas

1. ¿Cómo hace una predicción la visión bayesiana, comparada con la máxima verosimilitud?
 - a. Usa solo el único modelo con la pérdida más baja
 - b. Promedia lo que dicen todos los modelos plausibles, cada uno ponderado por lo plausible que es tras ver los datos
 - c. Ignora los datos y usa solo el a priori
 - d. Elige un modelo al azar
2. ¿Qué añade el a priori cuando conserva solo el único modelo más plausible?

- a. Un término de regularización que tira de la solución hacia lo que el a priori favorece y ayuda contra el sobreajuste
 - b. Un segundo conjunto de datos con el que entrenar
 - c. Una tasa de aprendizaje mayor
 - d. Nada en absoluto, en todos los casos
3. ¿Por qué el error cuadrático es una pérdida natural para predecir números?
- a. Porque ignora los errores grandes
 - b. Porque es la única pérdida diferenciable
 - c. Porque es lo que da la máxima verosimilitud cuando los errores siguen una curva en forma de campana
 - d. Porque tiene una esquina aguda en cero
4. ¿Por qué el libro apila todo el conjunto de datos en una sola tabla?
- a. Para que toda predicción salga de un único producto de matrices, que los chips gráficos hacen muy rápido
 - b. Porque el orden de los ejemplos cambia las predicciones
 - c. Porque los modelos lineales no pueden manejar ejemplos individuales
 - d. Para quitar el sesgo del modelo

Respuestas

1. **b)** Una predicción promedia lo que dicen todos los modelos, cada uno ponderado por su plausibilidad tras los datos, como promediar las estimaciones de los amigos.
2. **a)** Conservar solo el modelo más plausible da la verosimilitud más un término del a priori, que tira hacia el a priori: regularización, útil contra el

sobreaajuste.

3. c) El error cuadrático es exactamente lo que da la máxima verosimilitud si se supone que los errores siguen una curva en forma de campana.

4. a) Con un ejemplo por fila, todas las predicciones salen de un único producto de matrices, exactamente lo que mejor hacen los chips gráficos.

Ejemplos resueltos

Tres pérdidas, un valor atípico Compare la pérdida cuadrática, la pérdida absoluta y la pérdida mixta de Huber para un error pequeño de 0.5 y un error grande de 3. La pérdida de Huber es la mitad del error cuadrático cuando el error es como mucho 1, y el error absoluto menos un medio en otro caso.

Error 0.5: cuadrática 0.25, absoluta 0.5, Huber la mitad de 0.25, que es 0.125.

Error 3: cuadrática 9, absoluta 3, Huber $3 - 0.5 = 2.5$.

La pérdida cuadrática crece nueve veces con el error grande, así que un ejemplo mal etiquetado puede dominar el promedio; las pérdidas absoluta y de Huber crecen solo al ritmo del error.

Respuesta: 0.25, 0.5, 0.125 para el error pequeño; 9, 3, 2.5 para el grande

Ejercicios

1. Un modelo lineal tiene los pesos 2 y -1 y el sesgo 0.5. ¿Qué predice para las características 3 y 4, y cuál es su error cuadrático si la respuesta correcta es 3?

Pista: Pondere cada característica, súmelas y luego añada el sesgo.

Respuesta: $2 \times 3 + (-1) \times 4 + 0.5 = 2.5$. El error es $2.5 - 3 = -0.5$, así que el error cuadrático es 0.25.

2. Un conjunto de datos tiene 100 ejemplos con 3 características cada uno, apilados en una tabla. ¿Qué forma tiene la tabla, y cuántas predicciones da un producto de matrices con los 3 pesos?

Pista: Un ejemplo por fila.

Respuesta: La tabla tiene 100 filas y 3 columnas; multiplicarla por los 3 pesos da 100 predicciones a la vez, a las que se añade el mismo sesgo.

3. Explique cómo la visión bayesiana ayuda a un modelo a aprender de datos nuevos sin olvidar lo que ya sabía.

Pista: ¿Qué papel juega hoy el a posteriori de ayer?

Respuesta: Tras los primeros datos, el a posteriori guarda lo que el modelo aprendió. Cuando llegan datos nuevos, ese a posteriori se usa como nuevo a priori y se actualiza otra vez, así que el conocimiento antiguo se lleva consigo en vez de sobrescribirse.