APRENDIZAJE SUPERVISADO, II: MULTIVARIANTE Y SOBREAJUSTE Y SUBAJUSTE

Contenido

1.- Regresión multivariante

Comencemos revisando la versión de regresión lineal que analiza no solo una característica, sino muchas características diferentes.

En la versión original de la regresión lineal, teníamos una sola característica x, el tamaño de la casa y con ello somos capaces de predecir y, el precio de la casa.

Living area (feet^2)	Price (1000$s)
2104	400
1600	330
2400	369
1416	232
3000	540
⋮	⋮

El modelo era $\hat{f}(\mathbf{w}, \mathbf{b})$=b+$w_1$X

Pero ahora, ¿y si no solo tuvieramos el tamaño de la casa como una característica con la que tratar de predecir el precio, sino que también supiéramos el número de dormitorios, el número de pisos y la edad de la casa en años? Parece que esto le daría mucha más información con la que predecir el precio.

Multiple features (variables)

Size in feet²	Number of bedrooms	Number of floors	Age of home in years	Price ($) in $1000's
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	1	36	178
...	...	...	...	...

Para introducir un poco de notación nueva, vamos a usar las variables X $_1$, X $_2$, X $_3$ **y X_4,** para denominar las cuatro características. Escribiremos X subíndice X sub j **X_j,** para representar la lista de características. Aquí, j pasará de uno a cuatro, porque tenemos cuatro características. Usaremos n minúscula para denotar el número total de características, por lo que, en este ejemplo, n es igual a 4.

Como antes, usaremos **X superíndice** i x i para denotar el ejemplo de entrenamiento $^{\text{i-ésimo}}$. Aquí x i en realidad va a ser una lista de cuatro números, o a veces llamaremos a esto un vector que incluye todas las características del ejemplo de entrenamiento i-ésimo.

Como ejemplo concreto, x **superíndice entre paréntesis 2 $x^{(2)}$**, será un vector de las características para el segundo ejemplo de entrenamiento, por lo que será igual a este 1416, 3, 2 y 40 y técnicamente, a veces esto se llama vector fila en lugar de vector columna.

Para referirme a una característica específica en el ejemplo de entrenamiento i-ésimo, escribiremos X superíndice i, subíndice j, así que, por ejemplo, X superíndice 2 subíndice 3 **x_3**

Multiple features (variables)

Size in feet²	Number of bedrooms	Number of floors	Age of home in years	Price ($) in $1000's
X_1	X_2	X_3	X_4	
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	1	36	178
...	...	...	...	...

$j = 1 \ldots 4$, $n = 4$, $i = 2$

x_j = j^{th} feature
n = number of features
$\vec{x}^{(i)}$ = features of i^{th} training example

$\vec{x}^{(2)}$

2 será el valor de la tercera característica, que es el número de pisos en el segundo ejemplo de entrenamiento y por lo tanto será igual a 2.

Ahora actualicemos el modelo. Podemos escribir el modelo como

$$\hat{f}(\mathbf{w},\mathbf{b})=b+w_1X+ w_2X+ w_3X+ w_4X$$

En cambio, el modelo será el mismo que antes, pero ahora usaremos vectores,

$$\vec{w} =[w_1,w_2, w_3,w_4]$$

b= un número

$$\vec{x} =[X_1,X_2, X_3,X_4]$$

Entonces el modelo se puede escribir como,

$$\hat{f}(\vec{w},\mathbf{b})=b+\vec{w} * \vec{x}$$

Siendo * un producto de punto de dos vectores.[1]

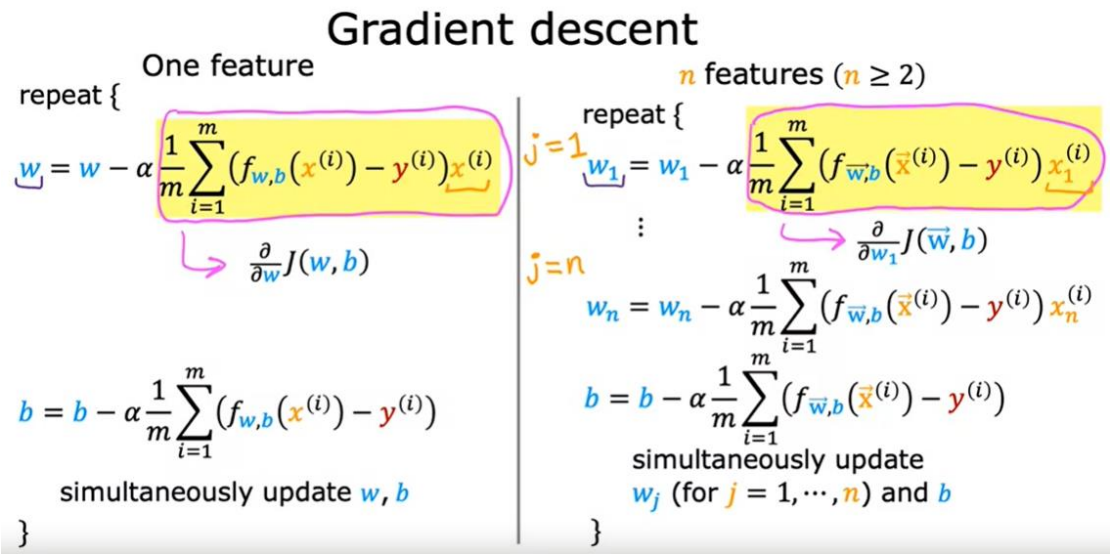
El MSE será: $J(\vec{w},b)$

Vamos a ponerlo todo junto para implementar el descenso de gradiente para la regresión lineal múltiple con vectorización.

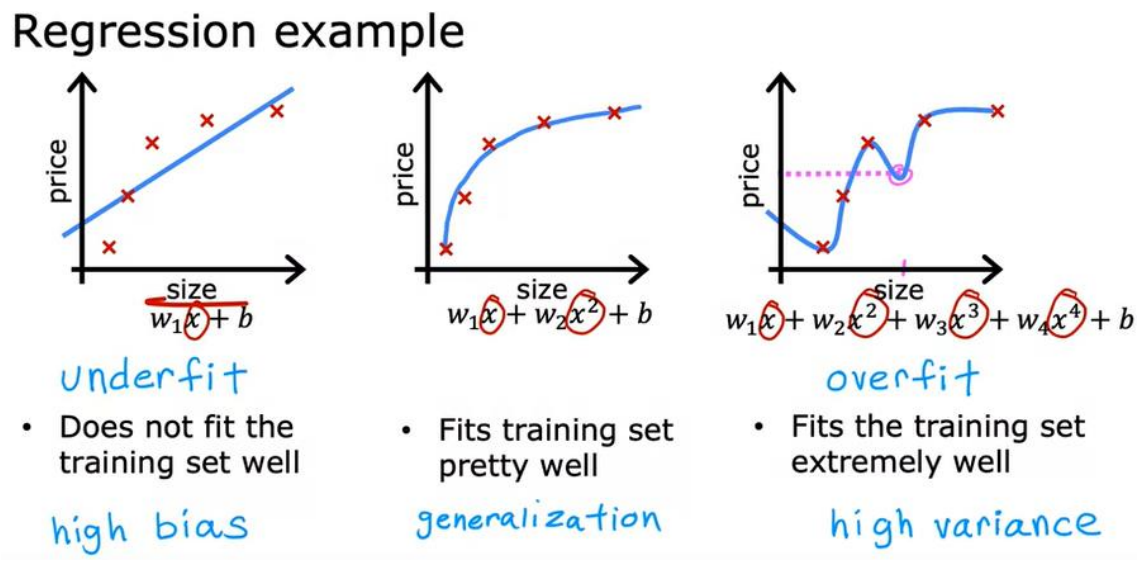
$$w_j := w_j - \alpha \frac{\delta}{\delta w_j} J(\vec{w},b) =$$

$$wj - \alpha \frac{1}{m} \sum_{i=1}^{m} (\hat{\vec{y}} - \vec{y})\, x_j^i$$

[1] En matemáticas multiplicamos Flecha By columnas para multiplicar dos vectores por lo que somos se supone que transponer El vector $\vec{x}$, pero en Python la biblioteca numpy usa la nomenclatura np.dot(w,x) para hacerlo. Esa es la razón No lo hacemos Escriba ahora la transposición of el vector



2.- Sobreajuste y subajuste (Ng, 2012)(Ng, 2020)



Ya hemos hablado de varianza y sesgo en notas técnicas anteriores, pero ahora, después de entender el modelo multivariante, es hora de tener una visión complementaria del problema.

Volvamos a nuestro ejemplo original de predecir los precios de la vivienda con regresión lineal, donde se predecir el precio en función del tamaño de una casa.

Para ayudarnos a entender qué es **el sobreajuste**, veamos un ejemplo de regresión lineal.

Supongamos que su conjunto de datos se ve como cruces en el gráfico, con la característica de entrada x siendo el tamaño de la casa, y el valor, y que está tratando de predecir el precio de la casa.

Una cosa que podría hacer es ajustar una **función lineal** a estos datos. Si hacemos eso, obtenemos un ajuste en línea recta a los datos. Pero este no es un muy buen modelo. Mirando

los datos, parece bastante claro que a medida que aumenta el tamaño de la casa, el proceso se aplana. Este algoritmo no se ajusta muy bien a los datos de entrenamiento. **El término técnico para esto es que el modelo no se ajusta a los datos de entrenamiento**. Otro término es que el algoritmo tiene **un alto sesgo**.

Es posible que hayas leído en las noticias sobre algunos algoritmos de aprendizaje que realmente demuestran prejuicios contra ciertas etnias o ciertos géneros. En el aprendizaje automático, el término sesgo tiene múltiples significados. Es fundamental verificar los algoritmos de aprendizaje en busca de sesgos basados en características como el género o el origen étnico. Pero el **término sesgo también tiene un segundo significado técnico**, que es el que estamos usando aquí, que es si el algoritmo ha subajustado los datos, **lo que significa que ni siquiera puede ajustarse bien al conjunto de entrenamiento**. Hay un patrón claro en los datos de entrenamiento que el algoritmo simplemente no puede capturar.

Otra forma de pensar en esta forma de sesgo es como si el algoritmo de aprendizaje tuviera una idea preconcebida muy fuerte, o decimos un sesgo muy fuerte, de que los precios de la vivienda van a ser una función completamente lineal del tamaño a pesar de los datos en contrario. Esta idea preconcebida de que los datos son lineales hace que se ajuste a una línea recta que se ajusta mal a los datos, lo que lleva a datos mal ajustados.

Ahora, veamos una **segunda variación de un modelo**, que es si inserta una función cuadrática en los datos con dos características, x y x ^ 2, entonces cuando ajuste los parámetros W1 y W2, puede obtener una curva que se ajuste a los datos algo mejor.

Además, si tuviera que obtener el precio de una casa nueva, que no está en este conjunto de cinco ejemplos de capacitación. Este modelo probablemente funcionaría bastante bien en esa nueva casa. Si eres agente de la propiedad, la idea de que quieres que tu algoritmo de aprendizaje funcione bien, incluso en ejemplos que no están en el conjunto de entrenamiento, **se llama generalización.**

Técnicamente decimos que desea que su algoritmo de aprendizaje se generalice bien, lo que significa hacer buenas predicciones incluso en ejemplos nuevos que nunca antes había visto. Estos modelos cuadráticos parecen ajustarse al conjunto de entrenamiento no perfectamente, pero bastante bien. Creo que generalizaría bien a nuevos ejemplos.

Ahora veamos el **otro extremo**. ¿Qué pasaría si ajustáramos un polinomio de cuarto orden a los datos? Tienes x, x^2, x^3 y x^4 como características. Con este polinomio, se puede ajustar exactamente la curva que pasa a través de los cinco ejemplos de entrenamiento. Esto, por un lado, parece hacer un trabajo extremadamente bueno ajustando los datos de entrenamiento porque pasa a través de todos los datos de entrenamiento perfectamente. De hecho, podría elegir parámetros que resultarán en que la función de coste sea exactamente igual a cero porque los errores son cero en los cinco ejemplos de entrenamiento.

Pero esta es una curva muy ondulada, sube y baja por todas partes. No parece que este sea un modelo particularmente bueno para predecir los precios de la vivienda. **El término técnico es que diremos que este modelo ha sobreajustado los datos,** o este modelo tiene un problema de sobreajuste.

Porque a pesar de que se ajusta muy bien al conjunto de entrenamiento, se ha ajustado demasiado bien a los datos, por lo tanto, está sobreajustado. No parece que este modelo se generalice a nuevos ejemplos que nunca antes se habían visto. **Otro término para esto es que**

el algoritmo tiene una alta varianza. En el aprendizaje automático, muchas personas usarán los términos sobreajuste y alta varianza casi indistintamente. Usaremos los términos underfit y high bias casi indistintamente.

La **intuición detrás del sobreajuste o la alta varianza** es que el algoritmo está tratando muy agresivamente de adaptarse a cada ejemplo de entrenamiento. Resulta que, si su conjunto de entrenamiento fuera un poco diferente, entonces la función que ajusta el algoritmo podría terminar siendo totalmente diferente. Por eso decimos que el algoritmo tiene una alta varianza.

Contrastando este modelo más a la derecha con el del medio para la misma casa, al parecer, el modelo medio les da una predicción mucho más razonable para el precio. Realmente no hay un nombre para este caso en el medio, pero vamos a llamar a esto correcto, porque no es ni inadecuado ni sobreadaptado.

Se puede decir que el objetivo del aprendizaje automático es encontrar un modelo que, con suerte, no sea ni insuficiente ni sobreajustado. En otras palabras, con suerte, un modelo que no tenga ni un alto sesgo ni una alta varianza.

3.- Abordar el sobreajuste (Ng, 2012)(Ng, 2020)

Veamos qué se puede hacer si creemos que puede haber sobreajuste. Digamos que se ajusta a un modelo y tiene una alta varianza, está sobreajustado. Aquí está nuestro modelo de predicción de precios de la vivienda sobre ajustado.

Una forma de abordar este problema es **recopilar más datos de entrenamiento**. Si puede obtener más datos, es decir, más ejemplos de capacitación sobre tamaños y precios de casas, **entonces con el conjunto de capacitación más grande, el algoritmo de aprendizaje aprenderá a ajustarse a una función que sea menos ondulada**. Puede continuar ajustando un polinomio de orden alto o algunas de las funciones con muchas características, y si tiene suficientes ejemplos de entrenamiento, lo hará correctamente.

Ahora, obtener más datos no siempre es una opción. Tal vez no se han vendido tantas casas en esta ubicación, por lo que tal vez no haya más datos para agregar. Una segunda opción para abordar el sobreajuste es ver si puede usar **menos características**.

En los ejemplos anteriores, las características de nuestros modelos podrían incluir el tamaño x, así como el tamaño al cuadrado, y este x al cuadrado, y x al cubo y x ^ 4 y así sucesivamente. Estas eran muchas características polinómicas. En ese caso, una forma de reducir el sobreajuste es simplemente no usar tantas de estas características polinómicas.

Pero ahora veamos un ejemplo diferente. Tal vez tenga muchas características diferentes de una casa de las cuales tratar de predecir su precio, que van desde el tamaño, el número de habitaciones, el número de pisos, la edad, el ingreso promedio del vecindario, etc., la distancia total a la cafetería más cercana. Resulta que si se tienen muchas características como estas pero no tiene suficientes datos de entrenamiento, entonces su algoritmo de aprendizaje también puede sobreajustarse a su conjunto de entrenamiento. Ahora, en lugar de usar las 100 funciones, si tuviéramos que elegir solo un subconjunto de las más útiles, tal vez el tamaño, las habitaciones y la edad de la casa. Si cree que esas son las características más relevantes, entonces usando solo ese subconjunto más pequeño de características, es posible que su modelo ya no se ajuste tan mal.

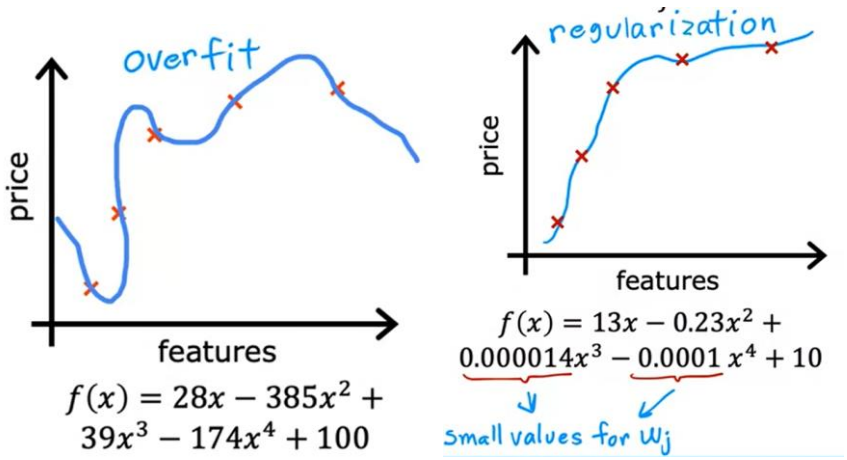
size	bedrooms	floors	age	avg income	...	distance to coffee shop	price
x_1	x_2	x_3	x_4	x_5		x_{100}	y

La elección del conjunto más apropiado de características para usar a veces también se denomina selección de características. Una forma de hacerlo es usar su intuición para elegir lo que cree que es el mejor conjunto de características, lo que es más relevante para predecir el precio. Ahora, una desventaja de la selección de características es que, al usar solo un subconjunto de las características, el algoritmo está desechando parte de la información que tiene sobre las casas.

Ahora, esto nos lleva a la tercera opción para reducir el sobreajuste. Esta técnica se denomina **regularización** (James et al., 2013, p. 176). Si observa un modelo sobreajustado, aquí hay un modelo que usa características polinómicas: x, x al cuadrado, x al cubo, etc. Nos encontraremos que los parámetros son a menudo relativamente grandes. Ahora, si tuviera que eliminar algunas de estas características, digamos, si eliminara la característica x ^ 4, eso corresponde a establecer este parámetro en 0. Por lo tanto, establecer un parámetro en 0 equivale a eliminar una característica. Resulta que la regularización es una forma de reducir los impactos de algunas características más suaves sin hacer algo tan duro como eliminarlo por completo. **Lo que hace la regularización es alentar al algoritmo de aprendizaje a reducir los valores de los parámetros sin exigir necesariamente que el parámetro se establezca exactamente en 0**. Resulta que incluso si se ajusta a un polinomio de orden superior como este, siempre que pueda obtener el algoritmo para usar valores de parámetros más pequeños: w1, w2, w3, w4. , se conlcuye con una curva que termina ajustándose mucho mejor a los datos de entrenamiento. Por lo tanto, lo que hace la regularización es permitir mantener todas sus características, pero solo evitan que las características tengan un efecto demasiado grande, que es lo que a veces puede causar sobreajuste.

Por cierto, por convención, normalmente solo reducimos el tamaño de los parámetros wj, es decir, w1 a wn. No hace una gran diferencia si se regulariza el parámetro b también, se puede hacer pero no es relevante.

Reducir el tamaño de los parámetros w_j



4.- Bibliografía

James, G., Witten, D., Hastie, T. y Tibshirani, R. (2013). *Una introducción al aprendizaje estadístico*.

Ng, A. (2012). 1. Aprendizaje supervisado. *Aprendizaje automático*, 1–30.

Ng, A. (2020). *IA para todos*.