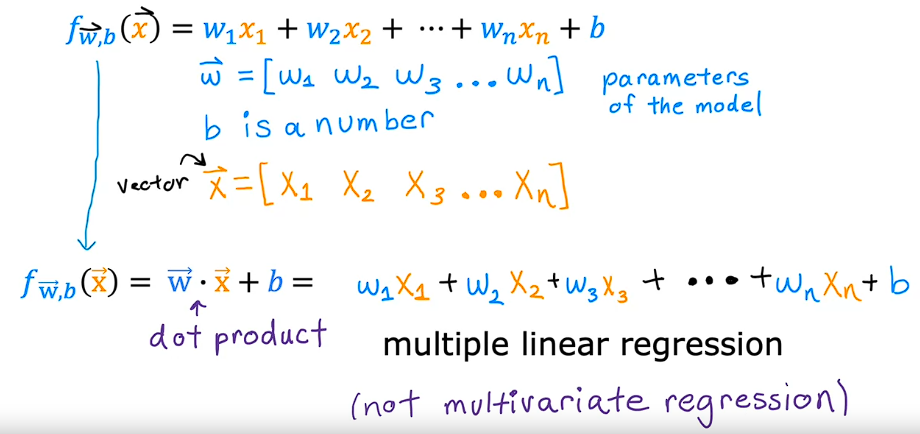
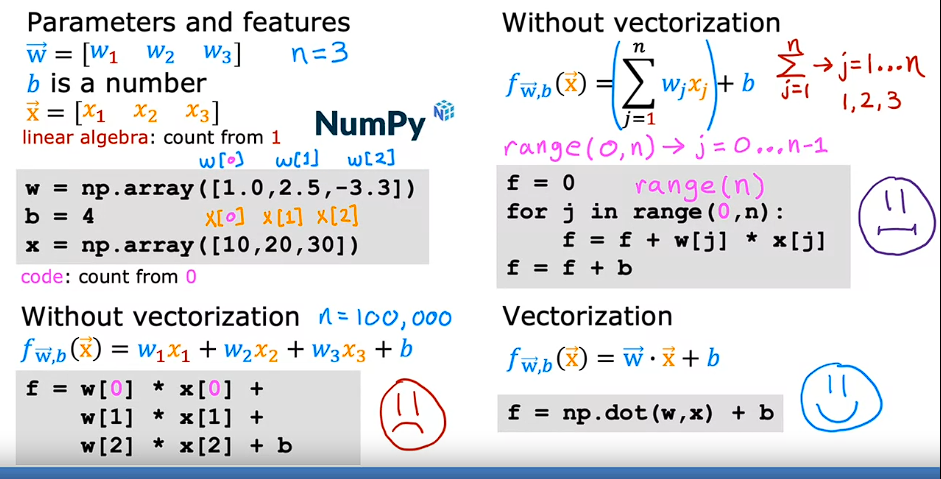
Linear regression with multiple variables

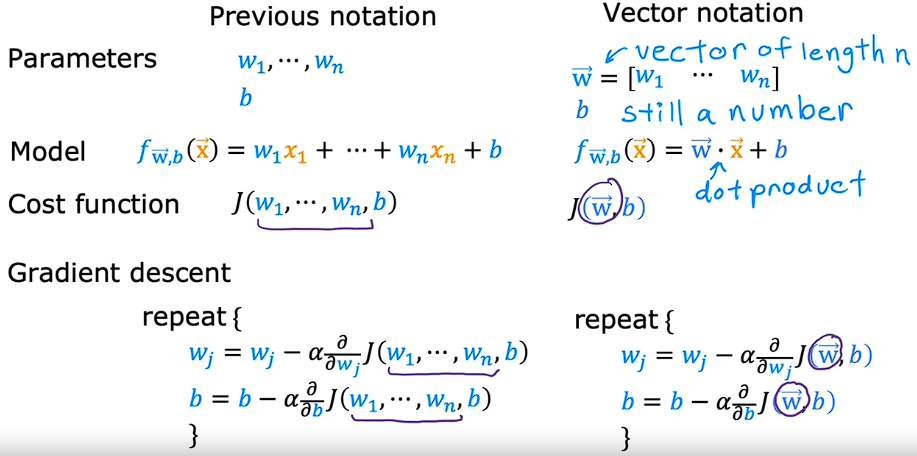


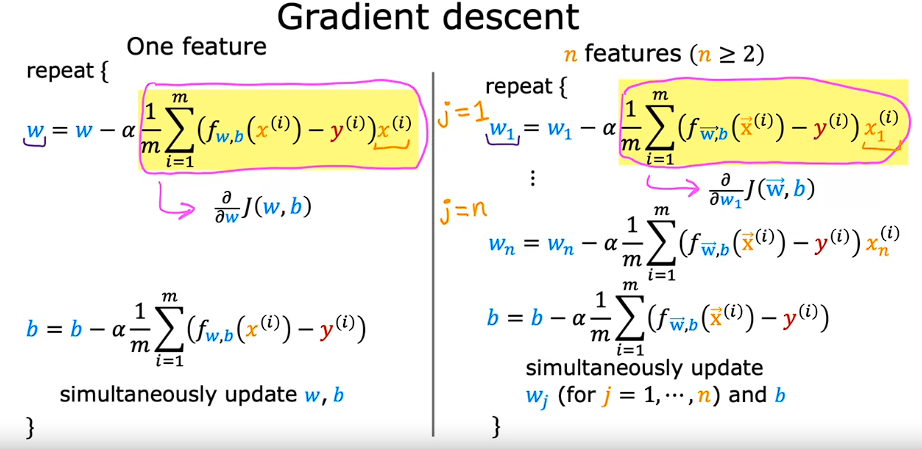
Vectorización



La función de producto punto (np.dot) de NumPy es capaz de usar hardware paralelo en tu ordenador, y esto es cierto tanto si lo ejecutas en un ordenador normal, es decir, en la CPU de un ordenador normal, como si utilizas una GPU, una unidad procesadora de gráficos, que a menudo se utiliza para acelerar los trabajos de aprendizaje automático. La función de producto punto (np.dot) de NumPy permite usar hardware paralelo hace que sea mucho más eficiente que el bucle for o el cálculo secuencial que vimos anteriormente. Ahora bien, esta versión es mucho más práctica cuando n es grande porque no se escribe w0 multiplicado por x0 más w1 por x1 más muchos términos adicionales como en la versión anterior. Pero si bien esto ahorra mucho en la escritura, todavía no es tan eficiente desde el punto de vista computacional porque todavía no usa la vectorización. En resumen, la vectorización hace que el código sea más corto, por lo que esperamos que sea más fácil de escribir y más fácil de leer para ti o para otras personas, y también hace que se ejecute mucho más rápido.

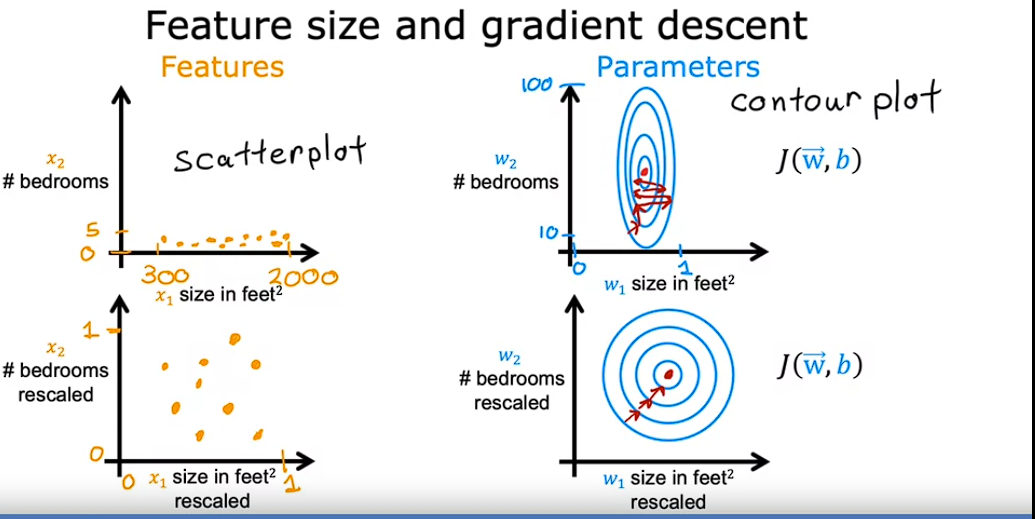
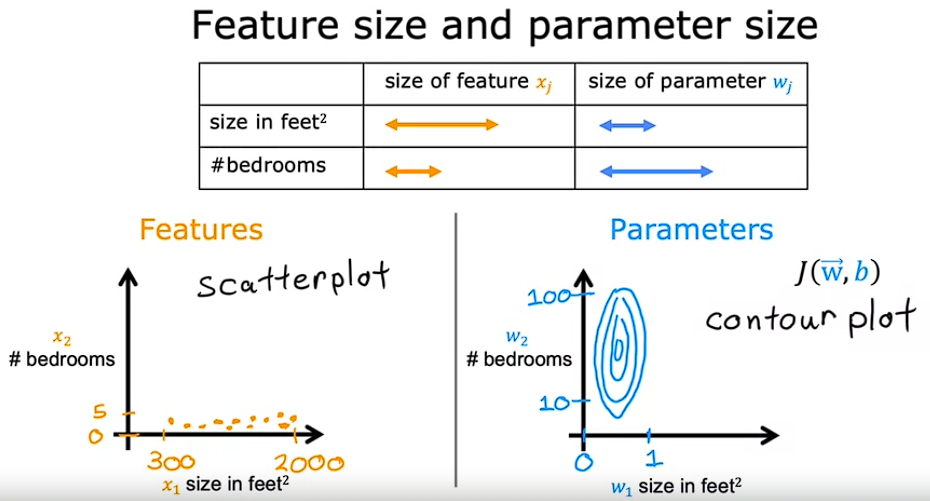
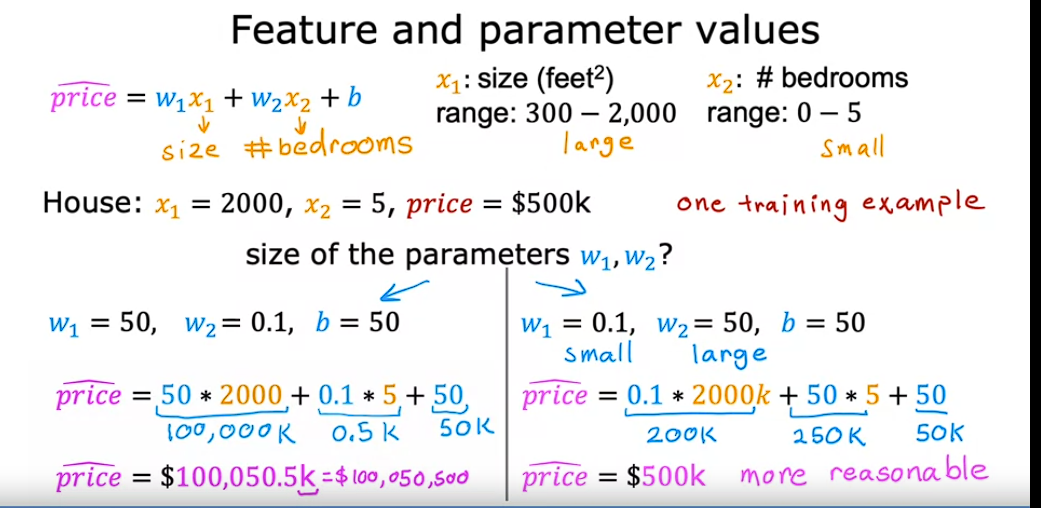
# **Descenso gradual para la regresión lineal múltiple**

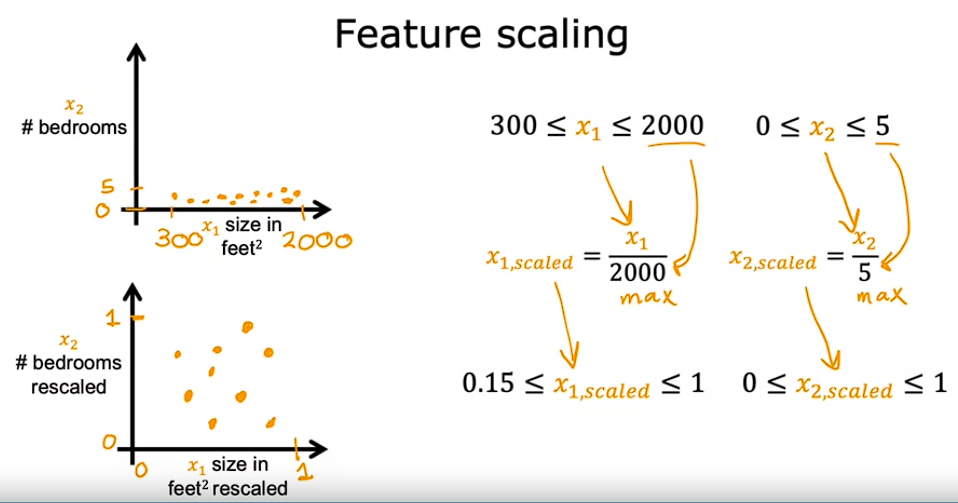


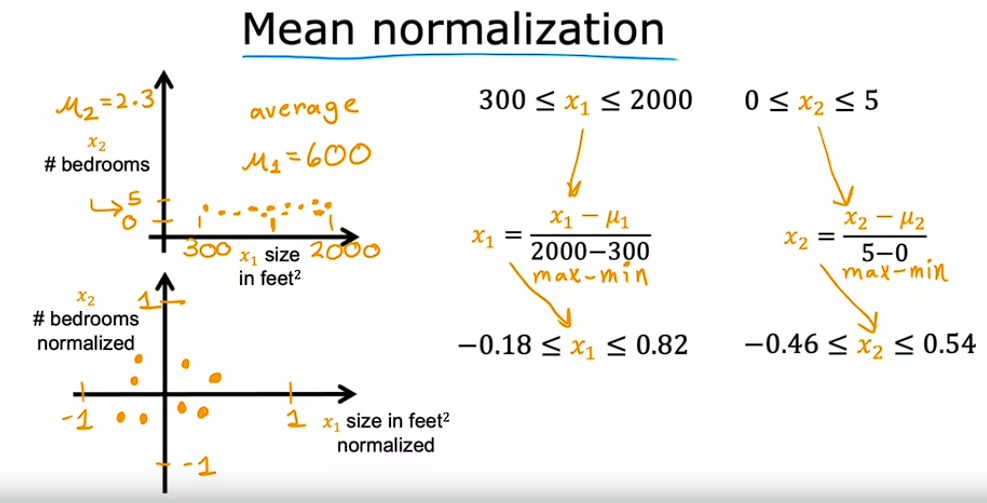


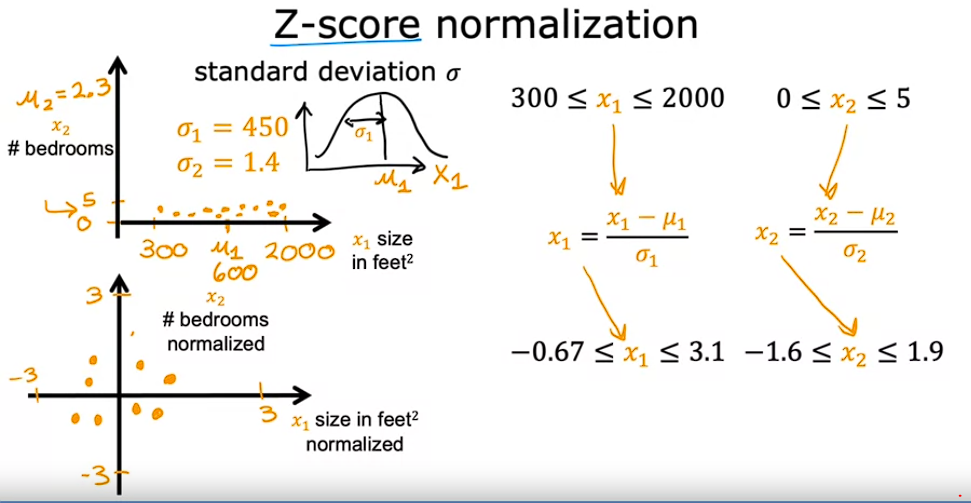
**Escalado de características**

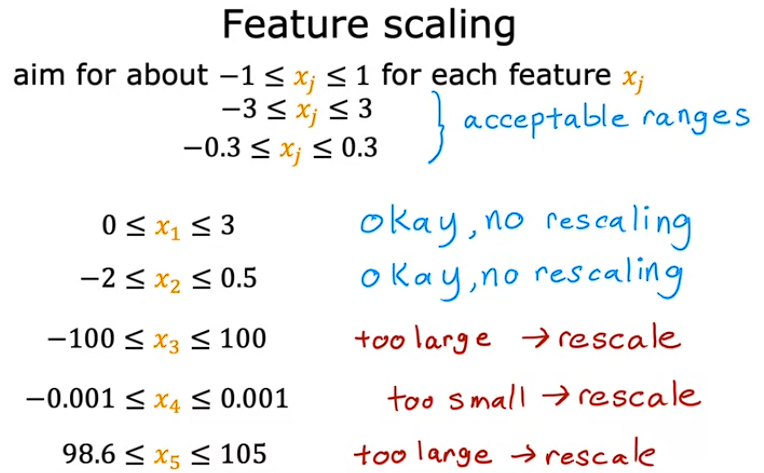
In this video you see a technique called feature scaling that will enable gradient descent to run much faster



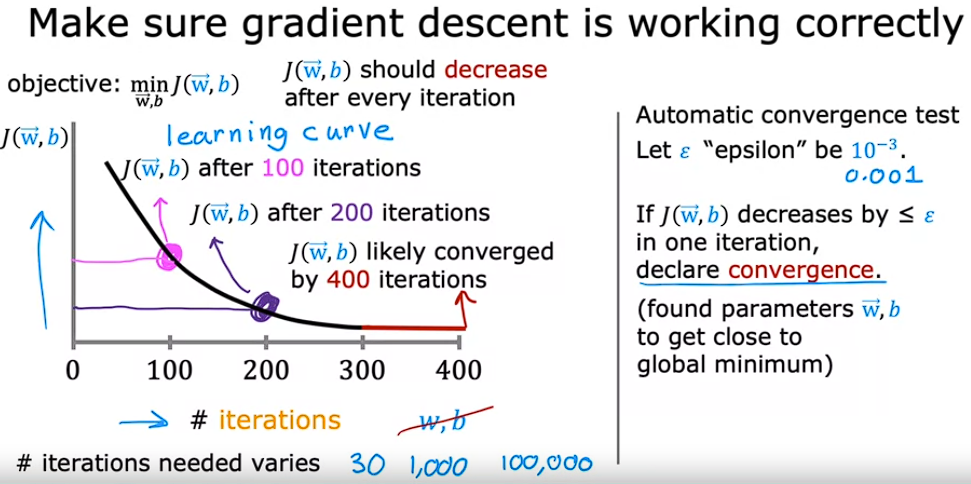






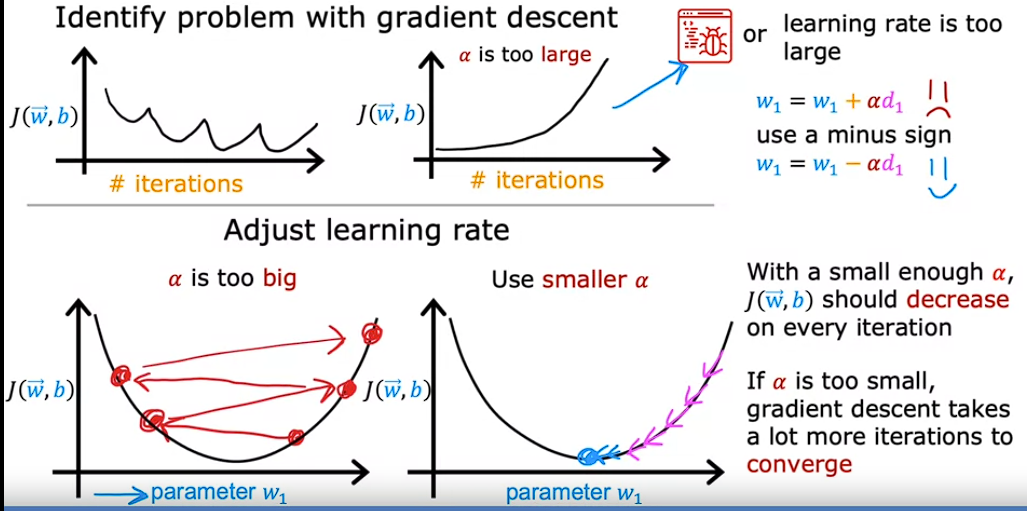


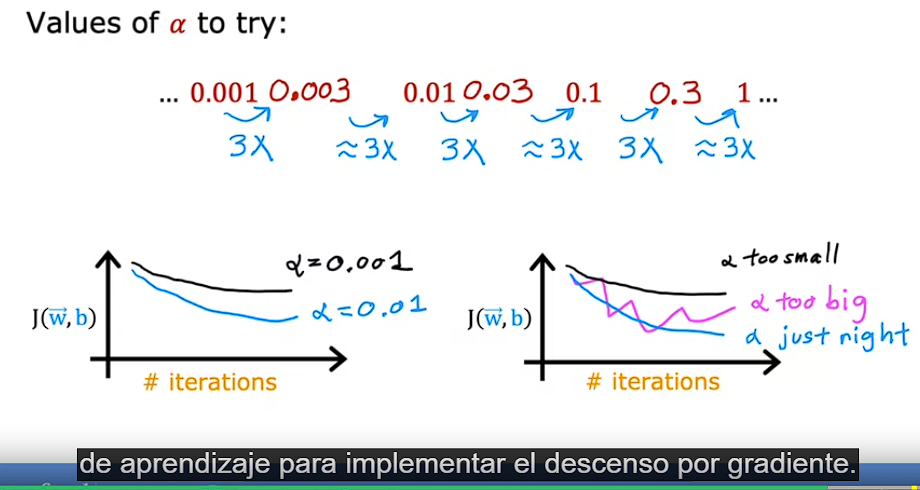
**Comprobación de la convergencia del descenso de gradiente**

****

# **Elección del ritmo de aprendizaje (elegir una buena tasa de aprendizaje para el modelo)**

Un consejo de depuración para implementar correctamente el gradiente de descenso es que, con una tasa de aprendizaje lo suficientemente pequeña, la función de coste debería disminuir en cada iteración. Así que si el gradiente de descenso no funciona, una cosa que suelo hacer y espero que este consejo también os resulte útil: otra cosa que suelo hacer es establecer Alpha en un número muy pequeño y comprobar si eso hace que el coste disminuya en cada iteración. Si incluso con Alpha establecido un número muy pequeño, J no disminuye en cada iteración, sino que a veces aumenta, eso normalmente significa que hay un error en alguna parte del código. Ten en cuenta que configurar Alpha para que sea muy pequeño es un paso de depuración y que un valor muy pequeño de Alpha no va a ser la opción más eficaz para entrenar realmente tu algoritmo de aprendizaje. Una compensación importante es que si tu ritmo de aprendizaje es demasiado bajo, los descensos de gradiente pueden tardar muchas iteraciones en converger. Por eso, cuando hago un descenso por gradiente, suelo probar un rango de valores para la tasa de aprendizaje alfa. Puedo empezar por probar una tasa de aprendizaje de 0,001 y también puedo probar una tasa de aprendizaje 10 veces mayor, digamos de 0,01 y 0,1, y así sucesivamente. Para cada opción de Alfa, puedes aplicar un gradiente descendente solo durante un puñado de iteraciones y trazar la función de coste J como una función del número de iteraciones. Después de probar algunos valores diferentes, puedes elegir el valor de Alpha, que parece disminuir la tasa de aprendizaje de forma rápida, pero también constante. De hecho, lo que hago en realidad es probar un rango de valores como este. Después de probar 0,001, triplicaré la tasa de aprendizaje hasta 0,003. Después de eso, probaré con 0.01, que nuevamente es aproximadamente tres veces más grande que 0.003. Así que se trata de probar a grandes rasgos los descensos de gradiente, en los que cada valor de Alpha es aproximadamente tres veces mayor que el valor anterior. Lo que haré es probar un rango de valores hasta encontrar el valor demasiado pequeño y luego asegurarme de haber encontrado un valor que sea demasiado grande. Intentaré elegir poco a poco la tasa de aprendizaje más alta posible, o simplemente algo un poco más pequeño que el valor razonable más alto que he encontrado. Cuando hago eso, normalmente obtengo una buena tasa de aprendizaje para mi modelo.

****

****

**Ingeniería de funciones**

En este video, se aborda el tema de la ingeniería de características en el aprendizaje automático supervisado. La elección de características adecuadas puede tener un gran impacto en el rendimiento del algoritmo de aprendizaje. A veces, es necesario diseñar nuevas características o combinar las características existentes para facilitar al algoritmo de aprendizaje hacer predicciones precisas.

A continuación, se presentan los puntos clave del video:

1. Importancia de elegir características adecuadas:

- La elección de características es un paso crítico para que el algoritmo funcione bien en aplicaciones prácticas.

- Puede ser necesario diseñar nuevas características o combinar las características existentes para mejorar el rendimiento del algoritmo.

2. Ejemplo de predicción del precio de una casa:

- Se utiliza el ejemplo de predecir el precio de una casa para ilustrar cómo se pueden elegir características adecuadas.

- Se tienen dos características: el ancho y la profundidad del terreno en el que se construye la casa.

- Se muestra un modelo inicial que utiliza estas dos características para predecir el precio de la casa.

3. Ingeniería de características:

- Se propone una nueva característica, el área del terreno, que se calcula multiplicando el ancho y la profundidad.

- Esta nueva característica puede ser más predictiva del precio de la casa que las características individuales de ancho y profundidad.

- Se muestra un nuevo modelo que utiliza las tres características (ancho, profundidad y área) para predecir el precio de la casa.

4. Ventajas de la ingeniería de características:

- La ingeniería de características permite ajustar no solo líneas rectas, sino también curvas y funciones no lineales a los datos.

- Al diseñar nuevas características, se pueden obtener modelos mucho mejores al tener en cuenta conocimientos o intuiciones sobre el problema.

5. Importancia de la intuición y el conocimiento del dominio:

- La ingeniería de características se basa en la intuición y el conocimiento del dominio del problema.

- Al comprender el problema y las características relevantes, se pueden diseñar características más efectivas.

6. Próximo video:

- El próximo video abordará cómo ajustar curvas y funciones no lineales a los datos utilizando la ingeniería de características.

Recuerda que la ingeniería de características es una herramienta poderosa para mejorar el rendimiento de los algoritmos de aprendizaje automático.

[Regresión polinómica](https://www.coursera.org/learn/machine-learning/lecture/OnGhN/polynomial-regression?trk_ref=coach_copy) 17 de ago. de 2024

En este contenido, se introduce el concepto de regresión polinómica como una extensión de la regresión lineal. La regresión polinómica permite ajustar curvas y funciones no lineales a los datos. Se presentan ejemplos de cómo ajustar una función cuadrática y cúbica a un conjunto de datos de viviendas.

Se destaca la importancia de la ingeniería de características al crear nuevas variables que son potencias de las variables originales. Se menciona que al utilizar características como el cuadrado o el cubo de una variable, es importante aplicar una escala a las características para que estén en rangos comparables.

Además, se menciona que existen diferentes opciones de características que se pueden utilizar, como la raíz cuadrada de una variable. Se enfatiza que la elección de características depende del conjunto de datos y que en cursos posteriores se aprenderá cómo seleccionar las mejores características y modelos.

También se menciona la importancia de la biblioteca Scikit-learn en el aprendizaje automático y se anima a los estudiantes a explorar los laboratorios opcionales que muestran cómo implementar la regresión polinómica y la regresión lineal utilizando Scikit-learn.

Finalmente, se invita a los estudiantes a participar en los cuestionarios y laboratorios de práctica para reforzar los conceptos aprendidos y se anticipa el próximo tema que se abordará en la siguiente semana: la clasificación.

<https://github.com/jagomezmat/machineLearning/blob/master/workWeek2/C1_W2_Lab04_FeatEng_PolyReg_Soln.ipynb>

**SEMANA 3**

Esta semana, aprenderá el otro tipo de aprendizaje supervisado, la clasificación. Aprenderá a predecir categorías utilizando el modelo de regresión logística. Aprenderá sobre el problema del sobreajuste, y cómo manejar este problema con un método llamado regularización. Al final de esta semana, ¡podrá practicar la aplicación de la regresión logística con regularización!

Objetivos de aprendizaje

Utilizar la regresión logística para la clasificación binaria

Implementar la regresión logística para la clasificación binaria

Abordar el sobreajuste mediante la regularización, para mejorar el rendimiento del modelo

[Motivaciones](https://www.coursera.org/learn/machine-learning/lecture/aoMt6/motivations?trk_ref=coach_copy) 17 de ago. de 2024

En esta semana del curso, aprenderás sobre clasificación en el aprendizaje automático. Mientras que la regresión lineal es útil para predecir valores numéricos, no es adecuada para problemas de clasificación donde la variable de salida solo puede tomar uno de dos valores posibles. Por ejemplo, determinar si un correo electrónico es spam o no, o si una transacción financiera en línea es fraudulenta. En estos casos, se utiliza la clasificación binaria, donde los valores de salida son cero o uno.

La regresión lineal no es la mejor opción para problemas de clasificación, ya que puede generar resultados imprecisos. En su lugar, se utiliza un algoritmo llamado regresión logística, que es uno de los algoritmos de aprendizaje más populares y ampliamente utilizados en la actualidad. La regresión logística produce valores de salida entre cero y uno, lo que evita los problemas que se presentan con la regresión lineal en problemas de clasificación.

En el próximo video, aprenderás más sobre el límite de decisión y cómo la regresión lineal puede generar resultados incorrectos cuando se utiliza para clasificación. También explorarás en detalle el algoritmo de regresión logística y cómo se utiliza para resolver problemas de clasificación binaria.

Recuerda que la clasificación es una parte fundamental del aprendizaje automático y es utilizada en una amplia variedad de aplicaciones, como la detección de spam, la detección de fraudes financieros y la clasificación de tumores. ¡Sigue adelante y continúa aprendiendo!

[Regresión logística](https://www.coursera.org/learn/machine-learning/lecture/zNxaw/logistic-regression?trk_ref=coach_copy) 18 de ago. de 2024

En este contenido se habla sobre la regresión logística, que es uno de los algoritmos de clasificación más utilizados en el mundo. Se explica cómo funciona este algoritmo y cómo se utiliza para clasificar tumores como malignos o benignos. Se introduce la función sigmoide, que es una función matemática importante en la regresión logística. También se explica cómo interpretar la salida de la regresión logística, que representa la probabilidad de que una muestra pertenezca a una clase determinada. En el próximo video se profundizará en los detalles de la regresión logística y se explorará el concepto de límite de decisión.

