# Tema 1

## 1. Introducción

El crecimiento de Big data se ha visto favorecido por el desarrollo de frameworks (plataformas de código abierto).

**Hadop y spark**
- +facilidad de trabajo
- + barato de almacenar

Con la aparición de Internet of things (IoT), más objetos y dispositivos están conectados a internet, recopilando datos sobre los patrones de uso de los clientes y además la aparición del aprendizaje automático ha permitido el aprovechamiento de estos datos.

La computación en la nube ha ampliado la utilidad de Big Data.

**Big Data** se fundamenta en dos ideas → almacenamiento y análisis de datos.

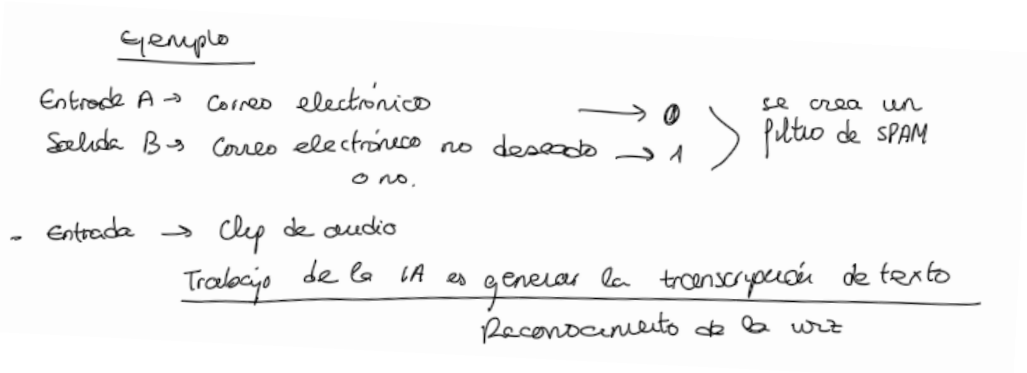
No se puede entender sin los nuevos desarrollos y algoritmos para entender los datos

*Así surgen términos como; Aprendizaje estadístico, inteligencia artificial, machine learning y Deep learning*

Debido a la aparición del aprendizaje automático y otras disciplinas el aprendizaje estadístico ha surgido como un nuevo subcampo en estadística centrado en el modelo y la predicción supervisada y no supervisada.

¿Cuáles son las fuerzas que han fomentado la era BIG DATA? Son 3 fuerzas;

1) Volumen y variedad de datos.
2) Computación en la nube.
3) Aplicación y desarrollo de algoritmos aplicables a situaciones reales para crear nuevo valor.




## 2. IA → Inteligencia Artificial.

Su aumento ha sido impulsado por una herramienta; **el aprendizaje automático.**

- El tipo más usado de aprendizaje automático es un tipo de IA que aprende de A o B, podemos definirlo como **Aprendizaje supervisado.**

La IA ha despegado recientemente debido al aumento de las redes neuronales y gracias al **aprendizaje profundo.**

**¿Cómo alcanzamos un nivel óptimo de rendimiento?** → Necesitamos dos cosas; Tener gran cantidad de datos y ser capaz de entrenar una red neuronal grande.

Muchas compañías pueden entrenar grandes redes neuronales con una cantidad suficientemente grande de datos, obteniendo un rendimiento bastante bueno gracias al crecimiento de la velocidad de computación y de los procesadores especializados (uds de procesamiento de gráficos o las GPU).

## 3. ¿Qué tipo de tareas podemos realizar?

Los tipos de tareas que pueden realizar los algoritmos teniendo los suficientes datos disponibles son:

- **Descripción** → cuando se combinan diferentes conjuntos de datos en un solo lugar o se muestran en un mapa, se puede obtener una mejor comprensión al verlo desde varias perspectivas. Esto puede ayudar a tomar mejores decisiones. Siempre hay nuevas formas de mejorar la forma en que se presentan los datos para mejorar la toma de decisiones.

Aprendizaje supervisado

- **Inferencias** → conocer las causas subyacentes de un fenómeno y la influencia de cada factor en el resultado es esencial para poder tomar medidas efectivas para cambiarlo.
- **Predicciones** → con los algoritmos, se pueden hacer predicciones.
- **Clasificaciones** → clasifica los elementos en los grupos a los que pertenecen. Esta tarea es especialmente adecuada

Aprendizaje no supervisado

- **Agrupamientos/Clustering** → Clustering es una técnica que consiste en agrupar objetos similares juntos en un grupo, de tal manera que los objetos en un mismo grupo sean más parecidos entre sí que con los objetos en otros grupos. Esta técnica es útil en marketing para segmentar a los clientes y entender mejor sus necesidades y comportamientos.
- **Recomendaciones** → son como una predicción, recomienda algo que quieres. Hay muchas técnicas; como el filtrado colaborativo o algoritmos de aprendizaje automático, el análisis de redes para crear motores de búsqueda como Google™.
- **Sistemas cognitivos** → un sistema cognitivo es un enfoque más avanzado y complejo para la automatización y la toma de decisiones en situaciones complejas.

## 4. Tipos de datos

¿Por qué existe Big Data? Existe gracias a la medida actual en la que la información puede generarse y ponerse a disposición.

**Diferencia entre digitalización y datificación**

- Digitalización son datos en formato digital, y su siguiente paso es la data, que consiste en describir o enunciar un fenómeno en un formato cuantificado para que pueda ser tabulado y analizado.

Conclusión y diferencia fundamental

- La digitalización permite que la información analógica se transfiera y almacene en un formato digital más conveniente.
- La datificación garantiza la versión digitalizada de las señales analógicas para generar información que no se había inferido con estas señales en su forma original.

El rendimiento depende del volumen y la variedad de datos.

- SQL → Elementos que podemos almacenar en 1 tabla.
- BIG DATA → Dato estructurado (tabla) o semiestructurado.

## 5. Fuentes de Datos

1) Máquinas
2) Gente
3) Corporaciones

## 6. Plataforma y programabilidad: Tecnología

El tamaño de los datos y la complejidad del procesamiento requieren un almacenamiento y rendimiento computacional adecuado. Según Google Trends, **"Hadoop"** es la consulta más asociada con "Big Data" y es la tecnología más destacada en este campo. Hadoop es un framework de código abierto que permite el procesamiento distribuido de grandes cantidades de datos utilizando un grupo de máquinas y modelos específicos de programación informática.

**Principales componentes de Hadoop →**

- **El sistema de archivos HDFS de Hadoop** facilita el acceso y la gestión de datos que están distribuidos en un entorno de almacenamiento complejo y disperso.
- **MapReduce** es un modelo de programación que se ha diseñado específicamente para implementar algoritmos distribuidos y paralelos de manera eficiente. Sin embargo, hay ciertas tareas que no pueden ser realizadas únicamente utilizando el algoritmo de computación MapReduce. Por ello, en los últimos años, se han agregado muchos otros complementos y módulos al ambiente de Hadoop para ampliar sus funcionalidades y mejorar su capacidad de procesamiento de datos.

## 7. Características de Big Data

Definición de Big Data → Conjunto de datos más grandes y complejos, especialmente de nuevas fuentes de datos. Tienen tanto volumen que el software tradicional no puede administrarlo.

**¿Cuáles son las Vs del Big Data?**

1) **Volumen** de los datos.
2) **Velocidad** a la que se reciben los datos y se actúa**.**
3) **Variedad** de los datos disponibles.
4) **Veracidad** (que cantidad de precisión puede ser un conjunto de datos)
5) **Valencia →** es la conexión entre los elementos de datos. Los gráficos de redes pueden ser útiles para visualizar y analizar estas conexiones de datos.
6) **Valor →** aspecto fundamental en Big Data. Todo debe ir enfocado a crear valor para el negocio, personas y/o sociedad.

## 8. Visión estratégica: las 5 P del Big Data y los proyectos de datos.

Distinguimos 5 elementos que determinan la creación de valor en una visión estratégica de un proyecto de datos.

1) **Propósito.**
2) **Personas.**
3) **Procesos →** captura el valor de las fuentes de datos y lo transforma en valor, hay dos tipos de procesos.
- Procesos técnicos
- Procesos organizativos.
4) **Plataformas** (arquitectura del sistema, el diseño e implementación de hardware…)
5) **Programabilidad** (herramientas y lenguajes de programación).

## 9. Definición de Big Data

Se refiere a **4 temas clave;** información, tecnologías, métodos e impacto o creación de valor.

La generación de valor mediante una propuesta que comienza con la recolección de datos provenientes de diversas fuentes, incluyendo máquinas, personas y corporaciones. Estos datos son voluminosos, variados, veloces, veraces y conectados, y son analizados utilizando una variedad de algoritmos y técnicas para describirlos, predecirlos, clasificarlos, agruparlos y hacer recomendaciones. La plataforma en la que se ejecutan estos algoritmos cuenta con un entorno de programación que culmina en una visualización, métrica, ranking, recomendación o alerta, que sirve para facilitar el proceso de toma de decisiones o para una acción automática.

**APRENDIZAJE SUPERVISADO, I: CONCEPTOS GENERALES Y LÍNEA DE REGRESIÓN**

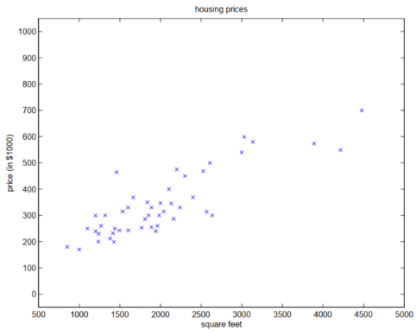
# Contenido

## 1.- Introducción y Nomenclatura

Imaginemos que tenemos un conjunto de datos que consiste en:

- Precio de algunas casas en una zona
- Pies cuadrados ( metros cuadrados) de cada uno de ellos

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




Si determinamos que existe una asociación entre pies cuadrados y precio, podremos predecir el precio de otras casas en esa zona (Ng, 2012).

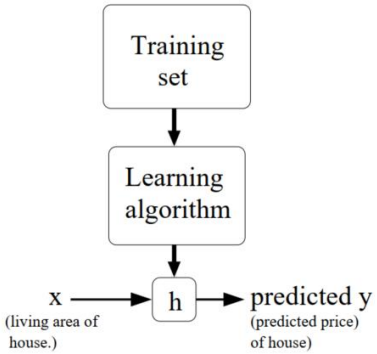
La variable de entrada son los pies cuadrados **(X)** (también conocida como **predictor, variable independiente, característica o simplemente variable**) . Cada una de las muestras de la entrada se denomina como **$x^{(i)}$**

La variable de salida **(Y)** (**también conocida como respuesta, variable dependiente o variable** objetivo) es el precio. Cada una de las muestras del output se denota como **$y^{(i)}$.**

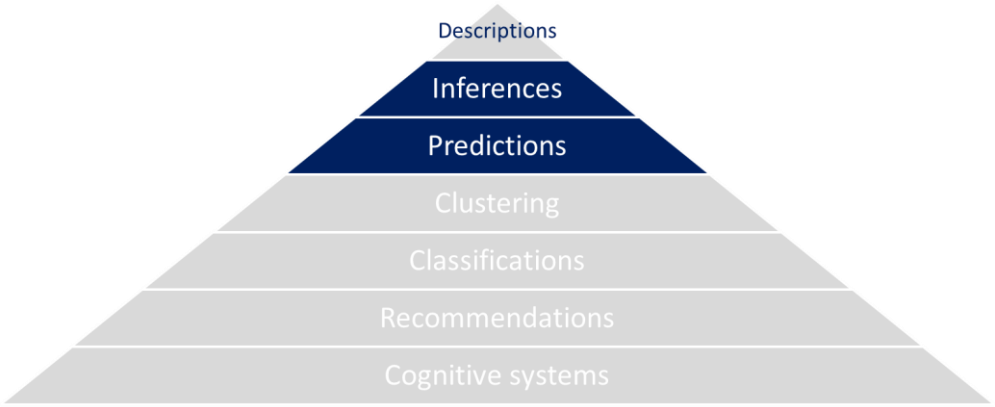
Un par **$(x^{(i)}, y^{(i)})$** se denomina **ejemplo de entrenamiento (training example),** y el conjunto de datos que usaremos para aprender: una lista de m ejemplos de entrenamiento **$\{(x^{(i)}, y^{(i)}); i = \{1, . . ., m\}$** se denomina **conjunto de entrenamiento ( training set).**

Téngase en cuenta que el **superíndice** **"(i)"** en la notación es simplemente un índice en el conjunto de entrenamiento y no tiene nada que ver con la exponenciación. También usaremos X para denotar el espacio de valores de entrada, e Y el espacio de valores de salida. En este ejemplo, **X = Y = R**

Para describir el problema de aprendizaje supervisado un poco más formalmente, nuestro objetivo es, dado un conjunto de entrenamiento, aprender una función h: X → Y para que **h(x)** sea un predictor "bueno" para el valor correspondiente de y. Por razones históricas, esta función h se llama hipótesis. Visto gráficamente, el proceso es, por lo tanto, así (Ng, 2012):



Cuando la variable objetivo que estamos tratando de predecir es continua, como en nuestro ejemplo de vivienda, llamamos al problema de aprendizaje un problema **de regresión**. Cuando y puede asumir solo un pequeño número de valores discretos (como si, dada la superficie habitable, quisiéramos predecir si una vivienda es una casa o un apartamento, por ejemplo), lo llamamos un **problema de clasificación.**




**El aprendizaje supervisado es un conjunto de algoritmos de aprendizaje automático que se pueden entrenar conociendo pares de entrada y salida.**

## 2.- Generalización del problema de la regresión

De manera más general, supongamos que observamos una respuesta cuantitativa Y y p predictores diferentes, $X_1$, $X_2$, . . ., Xp. Suponemos que existe alguna relación entre Y y X = ($X_1$, $X_2$, . . . , Xp), que se puede escribir de forma muy general como (James, Witten, et al., 2013):

$$Y = f(X) + \epsilon.$$

Aquí f es una función fija pero desconocida de **$X_1$, . . ., $X_p$**, y **ε** es un **término de error aleatorio**, que es independiente de **X** y tiene promedio cero. En esta formulación, **f** representa la información sistemática que **X** proporciona sobre **Y.**

En esencia, el aprendizaje estadístico paramétrico se refiere a un conjunto de enfoques para estimar **f.**

Hay dos razones principales por las que tal vez deseemos estimar f: **predicción** e **inferencia.**

## 3.- Predicción: error reducible e irreducible (James, Witten, et al., 2013, p. 34)

Dado que el término de error es cero en promedio, podemos predecir Y usando una aproximación

$$\hat{Y} = \hat{f}(X)$$

donde **f^** representa nuestra estimación para f, e **Y^** representa la predicción resultante para **Y**. En este contexto, **f^** a menudo se trata como una **caja negra**, en el sentido de que uno no se preocupa típicamente por la forma exacta de f^, siempre que produzca predicciones precisas para Y.

La precisión de **Y^** como predicción para Y depende de dos cantidades, que llamaremos el **error reducible** y el **error irreducible**

Este **error es reducible** porque potencialmente podemos mejorar la precisión de **f^** mediante el uso de la técnica de aprendizaje estadístico más adecuada para estimar f.

Sin embargo, incluso si fuera posible formar una estimación perfecta para f, de modo que nuestra respuesta estimada tomara la forma **Y^ = f(X)**, ¡nuestra predicción todavía tendría algún error!

Esto se debe a que, Y es también una función de **ε,** que, por definición, no se puede predecir usando **X**. Por lo tanto, la variabilidad asociada con **ε** también afecta la precisión de nuestras predicciones. Esto se conoce como el **error irreducible**, porque no importa lo bien que estimemos f, no podemos reducir el error introducido por **ε.**

La cantidad **ε** puede contener:

- Variables medidas que son útiles para predecir **Y:** ya que no las medimos, f no puede usarlas para su predicción.
- También puede contener variaciones inconmensurables, no medibles.

Consideremos una estimación dada **f^** y un conjunto de predictores **X**, que produce la predicción **Y^ = f^(X).** Supongamos por un momento que tanto f^ como X son fijos. Entonces, es fácil demostrar que

$$\begin{aligned} E(Y-\hat{Y})^2 &= E[f(X)+\epsilon-\hat{f}(X)]^2 \\ &= \underbrace{[f(X)-\hat{f}(X)]^2}_{\text{Reducible}} + \underbrace{\text{Var}(\epsilon)}_{\text{Irreducible}}, \end{aligned}$$

donde **E (Y – Y^)²** representa el promedio, o valor esperado, de la diferencia esperada al cuadrado entre el valor previsto y el valor real de Y, y Var(ε) representa la varianza asociada con el término de error **ε**

## 4.- Inferencia

A menudo estamos interesados en comprender la forma en que, **Y** se ve afectado por el cambio de $\mathbf{X_1, \ldots, X_p}$. En esta situación deseamos estimar **f**, pero nuestro objetivo no es necesariamente hacer predicciones para **Y**. En cambio, queremos entender la relación entre **X e Y,** o más específicamente, entender cómo **Y** cambia en función de $X_1, \ldots, X_p$. Ahora **f^ no puede ser tratado como una caja negra**, porque necesitamos saber su forma exacta.

**La inferencia se trata de comprender la relación causa-efecto. Queremos saber el por qué.**

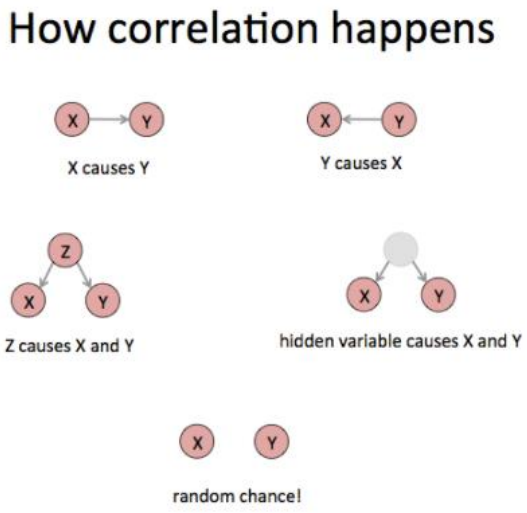
En una regresión lineal, por ejemplo, los parámetros que ponderan las entradas representan la relación estimada.

En cualquier caso, es importante distinguir la diferencia entre causalidad y correlación.

Si bien la causalidad y la correlación pueden existir al mismo tiempo, la correlación no implica causalidad. **La causalidad** se aplica explícitamente a los casos en que la acción **A** causa el resultado **B**. Por otro lado, la correlación es simplemente una relación. La acción A se relaciona con la acción B, pero un evento no necesariamente hace que ocurra el otro evento (Madhaven, 2020).

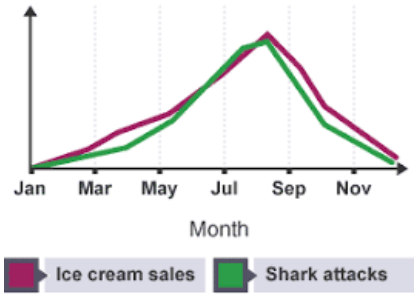
La correlación y la causalidad a menudo se confunden porque a la mente humana le gusta encontrar patrones incluso cuando no existen. A menudo fabricamos estos patrones cuando dos variables parecen estar tan estrechamente asociadas que una depende de la otra. Eso implicaría una relación de causa y efecto donde el evento dependiente es el resultado de un evento independiente. Hay muchas otras posibilidades para una asociación, incluyendo (Madhaven, 2020):

- Lo contrario es cierto: B causa A.
- Los dos están correlacionados, pero hay más: A y B están correlacionados, pero son causados por C.
- Hay otra variable involucrada: A causa B, siempre y cuando D suceda.
- Hay una reacción en cadena: A causa E, lo que lleva a E a causar B (pero sólo se vió que A causa B).




(*Por qué la correlación no es causalidad*, s.f.)

Por ejemplo (*Correlation Does Not Necessarily Mean Causation*, 2021), se realizó un estudio que encontró una fuerte correlación entre las ventas de helados y el número de ataques de tiburones para varias playas que fueron muestreadas.




(*Por qué la correlación no es causalidad*, s.f.)

**Conclusión**: El aumento de las ventas de helados causa más ataques de tiburones (a los tiburones les gusta comer personas que están llenas de helado).

**Mejor explicación**: La variable de confusión es la temperatura. Las temperaturas más cálidas hacen que las ventas de helados aumenten. Las temperaturas más cálidas también traen más personas a las playas, lo que aumenta las posibilidades de ataques de tiburones. Esto se conoce como respuesta común, donde dos variables (ventas de helados y ataques de tiburones) responden a los cambios en alguna tercera variable (temperatura).

**Las pruebas de hipótesis o la experimentación A / B / nos revelan la diferencia entre la correlación a la causalidad, si bien están fuera del ámbito de esta nota introductoria**

## 5.- Enfoques paramétricos y no paramétricos

Podemos distinguir dos enfoques diferentes: Paramétrico y No Paramétrico

- **Enfoques paramétricos**: reduce el problema a
    - Asumimos una forma de función dada. Por ejemplo, podemos suponer en nuestro ejemplo anterior que la relación entre el precio y los pies al cuadrado es lineal, siendo $h_\Theta$ nuestra salida estimada, **X** la entrada y $\Theta_0$ **y** $\Theta_1$ los parámetros a entrenar

      $$h_\Theta=\Theta_0+\Theta_1 X$$

    - Luego entrenamos el modelo con nuestros pares de muestras **$(x^{(i)}, y^{(i)})$[1] y calculamos el $\Theta_i$.** En este sentido vamos a ajustar la función asumida a los ejemplos de entrenamiento.
    - El problema surge la función asuminda. Si está lejos de lo real, la estimación será muy pobre.
    - La relación y por lo tanto la función estimada podría ser muy compleja, multivariante (varias entradas) y no lineal (no solo una recta sino cualquier otra función posible). Esto conducirá al concepto de sobreajuste y a la compensación sesgo-varianza que cubriremos más adelante.
- **Enfoques no paramétricos**:
    - No hacemos suposiciones sobre la función. Solo tratamos de encontrar una estimación que se acerque lo más posible a los puntos de datos sin ser demasiado áspero o ondulante (James, Witten, et al., 2013).
    - El problema es la falta de interpretabilidad del modelo. No conocemos las relaciones de causalidad
    - De todos modos, los mismos conceptos de sobreajuste y compensación sesgo-varianza son aplicables aquí.
    - Por ejemplo, las redes neuronales pueden considerarse no paramétricas.

[1] Recuerda que Y es la real salida y **$y^{(i)}$ en nuestro caso el precio real que corresponde al real $x^{(i)}$el pies** [2]

## 5.- Paramétrico: modelo para ajustar la función a los datos de entrenamiento (Ng, 2012, p. 4,5)

Vamos a describir la metodología para ajustar un modelo lineal con una sola entrada o variable a unos datos de entrenamiento, pero la metodología se puede generalizar. Volvamos al ejemplo del precio-tamaño

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

Aquí, las **X** son vectores unidimensionales en R. Por ejemplo, $\mathbf{x^{(i)}}_1$ es la superficie de la i-ésima casa en el conjunto de entrenamiento. En general, al diseñar un problema de aprendizaje, dependerá del científico de datos qué características elegir. Si se está recopilando datos de vivienda, también puede decidir incluir otras características como si cada casa tiene una chimenea, el número de baños, etc. Diremos más sobre la selección de características más adelante, pero por ahora tomemos la función como dada.

Para realizar un aprendizaje supervisado, debemos decidir cómo vamos a representar **funciones/hipótesis** $h_\theta$ en un ordenador. Como elección inicial, digamos que decidimos aproximarnos a **Y ( la relación real)** como una función lineal de x (la variable de entrada):

$$h_\theta(x) = \theta_0 + \theta_1 x_1$$

Aquí, los $\theta_i$ son los parámetros (también llamados pesos) que parametrizan el espacio de las funciones lineales que se mapean de X a Y.

Ahora, dado un conjunto de entrenamiento, ¿cómo elegimos, o aprendemos, los parámetros θ? Un método razonable parece ser hacer **h(x)** esté cerca de **Y**, al menos para las muestras de entrenamiento. Para formalizar esto, definiremos una función que mide, para cada valor de los **θ,** cómo de cerca están los $\mathbf{h(x^{(i)})}$ **de** los $\mathbf{y^{(i)}}$ correspondientes. Definimos pues la **función de coste**

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

Si se ha estudiado anteriormente este tema, se puede reconocer esta función de coste como la función de mínimos cuadrados que da lugar al **modelo de regresión de mínimos cuadrados ordinarios.**

Queremos elegir **θ** para minimizar **J(θ).** Para hacerlo, usaremos un algoritmo de búsqueda que comience con alguna "suposición inicial" para **θ**, y que cambie repetidamente θ para hacer **J(θ)** más pequeño, hasta que con suerte converjamos a un valor de θ que minimice J(θ). Específicamente, consideremos el algoritmo de **descenso de gradiente ( gradiente descent)**, que comienza con alguna **θ** inicial y realiza repetidamente la actualización.

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta).$$

(Esta actualización se realiza simultáneamente para todos los valores de **j = 0, . . . , n.)**. en la función del gradiente descendente, **α se llama tasa de aprendizaje ( learning paramenter**) . Este es un algoritmo muy natural que repetidamente da un paso en la dirección de la disminución más pronunciada de **J**.

Para implementar este algoritmo, debemos averiguar cuál es el término de la derivada parcial que indicamos en el lado derecho.

$$\begin{aligned}\frac{\partial}{\partial\theta_j}J(\theta) &= \frac{\partial}{\partial\theta_j}\frac{1}{2}\left(h_\theta(x)-y\right)^2 \\ &= 2\cdot\frac{1}{2}\left(h_\theta(x)-y\right)\cdot\frac{\partial}{\partial\theta_j}\left(h_\theta(x)-y\right) \\ &= \left(h_\theta(x)-y\right)\cdot\frac{\partial}{\partial\theta_j}\left(\sum_{i=0}^{n}\theta_i x_i - y\right) \\ &= \left(h_\theta(x)-y\right)x_j\end{aligned}$$

Para un solo ejemplo de entrenamiento, esto proporciona la regla de actualización:

$$\theta_j := \theta_j + \alpha\left(y^{(i)} - h_\theta(x^{(i)})\right)x_j^{(i)}.$$

La regla se llama regla de actualización de **LMS (LMS significa "mínimos cuadrados medios". "least square mean")** y también se conoce como la regla de aprendizaje de Widrow-Hoff.

Esta regla tiene varias propiedades que parecen naturales e intuitivas. Por ejemplo, la magnitud de la actualización es proporcional al término de error **($y^{(i)} - h\theta(x^{(i)})$;** así, por ejemplo, si nos encontramos con un ejemplo de entrenamiento en el que nuestra predicción casi coincide con el valor real de $y^{(i)}$, entonces encontramos que hay poca necesidad de cambiar los parámetros; en contraste, se realizará un cambio mayor en los parámetros si nuestra predicción $h_\theta(x^{(i)})$ tiene un gran error (es decir, si está muy lejos de $y^{(i)}$).

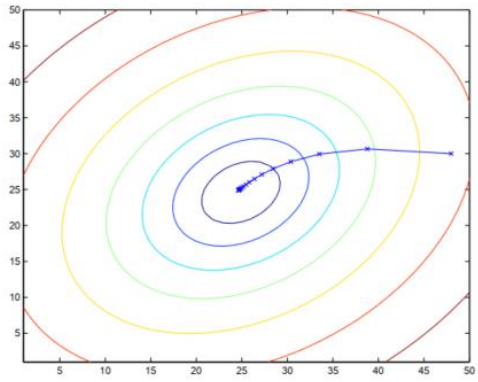
Habíamos derivado la regla LMS para cuando solo había un solo ejemplo de entrenamiento. Para más de un ejemplo lo reemplazamos por el siguiente algoritmo:

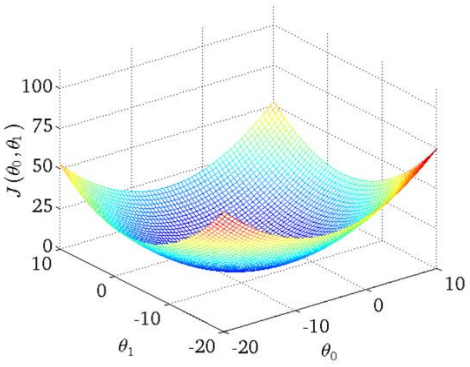
$$\begin{aligned}&\text{repeat until convergence } \{ \\ &\quad \theta_0 := \theta_0 - \alpha\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right) \\ &\quad \theta_1 := \theta_1 - \alpha\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)\cdot x^{(i)} \\ &\}\end{aligned}$$

Esto es simplemente el descenso de gradiente en la **función de coste original J**. Este método analiza cada ejemplo en todo el conjunto de entrenamiento en cada paso y se denomina **descenso de gradiente por lotes[2] ( "batch gradient descent")**. Tengase en cuenta que, si bien el descenso del gradiente puede ser susceptible a mínimos locales en general, el problema de optimización que hemos planteado aquí para la regresión lineal tiene solo un óptimo global y

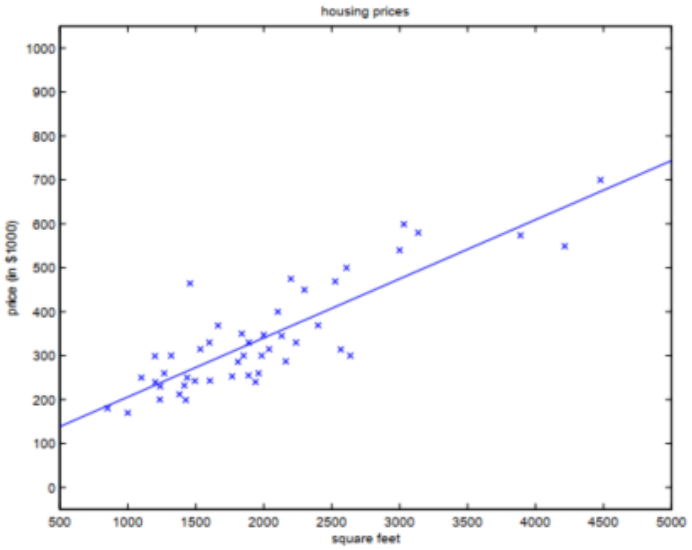
[2] "Batch": Cada paso de descenso de gradiente utiliza todos los ejemplos de entrenamiento.

ningún otro local; por lo tanto, el **descenso del gradiente siempre converge** (**suponiendo que la tasa de aprendizaje α no sea demasiado grande**) al mínimo global. De hecho, **J** es una función cuadrática convexa. En el gráfico se muestra el descenso de gradiente a medida que se ejecuta para minimizar una función cuadrática.






Cuando ejecutamos el descenso del gradiente por lotes para ajustar **θ** en nuestro conjunto de datos anterior, para aprender a predecir el precio de la vivienda en función del área habitable, obtenemos $\theta_0 = 71.27$, $\theta_1 = 0.1345$. Si trazamos **hθ(x)** en función de x (área), a lo largo con los datos de entrenamiento, obtenemos la siguiente figura:




## 6.- Flexibilidad e interpretabilidad

En el apartado anterior hemos ajustado una línea recta al conjunto de entrenamiento, pero por supuesto podemos añadir más características y podemos cambiar la forma de la curva para ajustarnos mejor a los datos y por tanto el error reducible será menor. Asimismo, podemos elegir una metodología no paramétrica que se ajuste muy bien a los datos sin asumir una función específica y finalmente consiguiendo una "superficie" compleja.

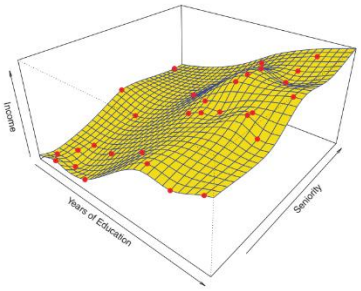



FIGURE 2.6. *A rough thin-plate spline fit to the* Income *data from Figure 2.3. This fit makes zero errors on the training data.*

Por ejemplo, abajo en la figura más a la izquierda se muestra el resultado de ajustar una $\mathbf{y = \theta_0 + \theta_1 x}$ a un conjunto de datos (**gráfico A)**. Vemos que los datos no se encuentran realmente en línea recta, por lo que el ajuste no es muy bueno (Ng, 2012, p. 14).

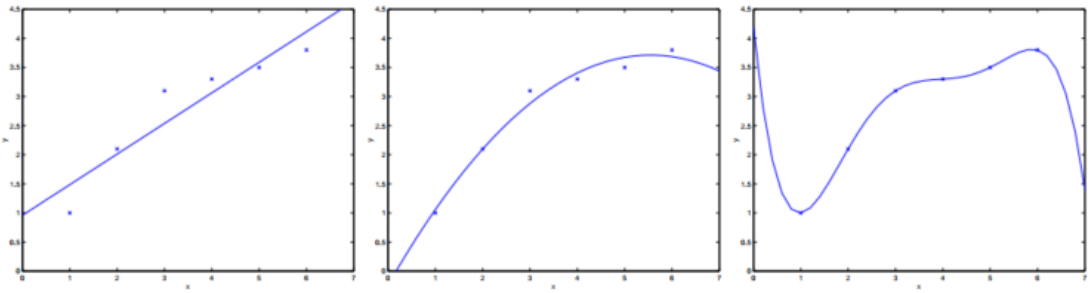


**Gráfico A** **Gráfico B** **Gráfico C**

Si hubiéramos añadido una característica extra $\mathbf{x_2}$, y ajustado (gráfico **B**) $\mathbf{y = \theta_0 + \theta_1 x + \theta_2 x^2}$, entonces obtenemos un ajuste ligeramente mejor a los datos. (Ver figura del medio B) Ingenuamente, podría parecer que cuantas más características agreguemos, mejor. Sin embargo, también existe el peligro de agregar demasiadas características: la figura más a la derecha es el resultado de encajar un polinomio de orden 5 (gráfico **C)** $\mathbf{y = \theta_0 + \theta_1 x + \theta_2 x^2 + \ldots\ldots + \Theta_5 x^5}$.

Vemos que a pesar de que la curva ajustada pasa a través de los datos perfectamente, no esperaríamos que esto sea un muy buen predictor de, digamos, los precios de la vivienda (y) para diferentes superficies (x). Sin definir formalmente lo que significan estos términos, diremos que la figura de la izquierda muestra un caso de **subajuste**, en el que los datos muestran claramente la estructura no capturada por el modelo, y la figura de la derecha es un ejemplo de **sobreajuste** (Ng, 2012, p. 14). Volveremos a estos conceptos más adelante.

De los muchos métodos existentes, algunos son menos flexibles, o más restrictivos, en el sentido de que pueden producir solo una gama relativamente pequeña de formas para estimar **f**. Por ejemplo, la regresión lineal es un enfoque relativamente inflexible, porque solo puede generar funciones lineales. Otros métodos, como las splines de placa delgada (no paramétricas), son considerablemente más flexibles porque pueden generar una gama mucho más amplia de formas posibles para estimar **f.** (James, Witten, et al., 2013, p. 25)

Se podría razonablemente hacer la siguiente pregunta: ¿por qué elegiríamos usar un método más restrictivo en lugar de un enfoque muy flexible? Hay varias razones por las que podríamos preferir un modelo más restrictivo pero la esencial es la interpretación. Si nos interesa principalmente **la inferencia**, entonces los modelos restrictivos son mucho más interpretables.

(James, Witten, et al., 2013, p. 25).. En el ejemplo dado es difícil interpretar el significado de la superficie a la potencia de 5. En el siguiente gráfico podemos ver la relación entre flexibilidad e interpretabilidad de algunos métodos.


(James, Witten, et al., 2013, p. 25)

En algunos entornos, sin embargo, solo nos interesa la predicción, y la interpretabilidad del modelo predictivo simplemente no es de interés. En este contexto, podríamos esperar que sea mejor utilizar el modelo más flexible disponible. Sorprendentemente, ¡este no es siempre el caso! A menudo obtendremos predicciones más precisas utilizando un método menos flexible. Este fenómeno, que puede parecer contradictorio a primera vista, tiene que ver con el potencial de sobreajuste en métodos altamente flexibles.

## 7.- Evaluación de la precisión del modelo: MSE para diferentes ajustes (James, Hastie, et al., 2013, pp. 29-33)

Para evaluar el rendimiento de un método de aprendizaje estadístico en un conjunto de datos dado, necesitamos alguna forma de medir cómo de buenas son sus predicciones por la coincidencia con los datos observados. Es decir, necesitamos cuantificar hasta qué punto el valor de respuesta predicho para una observación dada está cerca del valor de respuesta verdadero para esa observación. En el contexto de la regresión, la medida más utilizada es el **error cuadrático medio (MSE) que hemos visto**, dado por:

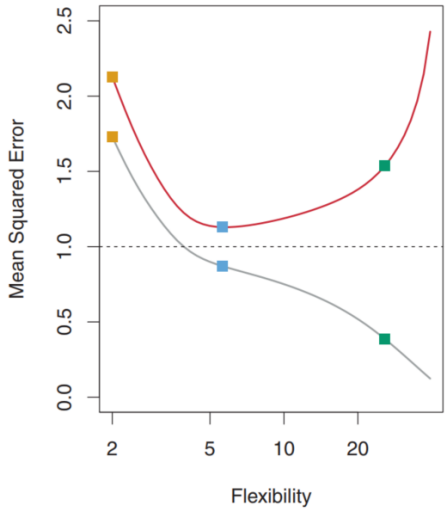
$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left( h_\theta(x^{(i)}) - y^{(i)} \right)^2$$

El MSE se calcula utilizando los datos de entrenamiento que se utilizaron para ajustarse al modelo, por lo que debe denominarse con mayor precisión el **MSE de entrenamiento**. Más bien, estamos interesados en la precisión de las predicciones que obtenemos cuando aplicamos nuestro método a datos de prueba nunca antes vistos. **Queremos elegir el método que da la prueba más baja MSE, en lugar de la MSE de entrenamiento más baja.**

En algunas configuraciones, podemos tener un conjunto de datos de prueba disponible, es decir, podemos tener acceso a un conjunto de observaciones que no se utilizaron para entrenar el método de aprendizaje estadístico.

Veamos cuál es la relación entre la MSE y la flexibilidad. La **curva gris** muestra el **MSE de entrenamiento promedio** en función de la flexibilidad, o más formalmente los grados de libertad, para una serie de splines de suavizado (un tipo de funciones no paramétricas). **Los grados de libertad** es una cantidad que resume la flexibilidad de una curva. Una curva más restringida y, por lo tanto, más suave tiene menos grados de libertad que una curva ondulada; tenga en cuenta que en la figura la regresión lineal está en el extremo más restrictivo, con dos grados de libertad. **El entrenamiento MSE disminuye monótonamente a medida que aumenta la flexibilidad.**

**La curva roja (conjunto de pruebas)** en comparación con **el gris (conjunto de entrenamiento)** muestra la evolución, pero ahora en el conjunto de pruebas. Observamos que el **MSE de entrenamiento disminuye** monótonamente a medida que aumenta la flexibilidad del modelo, y que hay una **forma de U en el MSE de prueba.** La línea discontinua es el error irreducible, el mínimo posible para el conjunto de prueba.




## 8.- The bias-Variance Trade-off (James, Hastie, et al., 2013, pp. 33-36)

La forma de U observada en las curvas MSE de prueba resulta ser el resultado de dos propiedades competitivas de los métodos de aprendizaje estadístico.

Es posible demostrar que la prueba esperada MSE, para un valor dado $x_0$, siempre se puede descomponer en la suma de tres cantidades fundamentales: **la varianza** de $\hat{f}(x_0)$, el **sesgo** de $\hat{f}(x_0)$ y **la varianza del ε de error.**

$$E\left(y_0 - \hat{f}(x_0)\right)^2 = \mathrm{Var}(\hat{f}(x_0)) + [\mathrm{Bias}(\hat{f}(x_0))]^2 + \mathrm{Var}(\epsilon).$$ [3]

[3] Está fuera del ámbito de esta nota técnica para explicarla en profundidad, así que no debe preocuparse si no comprende planamente la formulación. La intuición explicada es el elemento relevante

La ecuación nos dice que, para minimizar el error de prueba esperado, necesitamos seleccionar un método de aprendizaje estadístico que simultáneamente logre **una baja varianza y un bajo sesgo.**

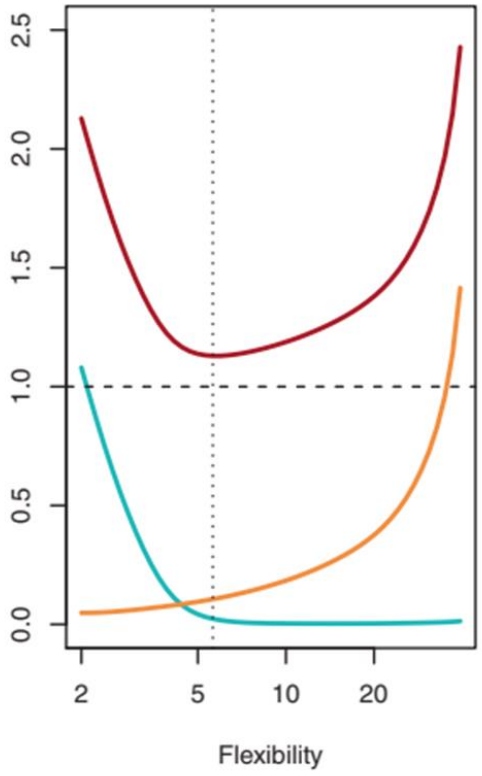
**La varianza (variance)** se refiere a la cantidad por la cual **f^** cambiaría si la estimamos **utilizando un conjunto de datos de entrenamiento diferente**. Dado que los datos de entrenamiento se utilizan para ajustarse al método de aprendizaje estadístico, diferentes conjuntos de datos de entrenamiento darán como resultado un **f^** diferente**. Pero lo ideal es que la estimación para f no varíe demasiado entre series de entrenamiento.** Si un método tiene una alta varianza, entonces pequeños cambios en los datos de entrenamiento pueden resultar en grandes cambios en **f^. En general, los métodos estadísticos más flexibles tienen una mayor varianza.**

**El sesgo (Bias)** se refiere al **error** que se introduce al **aproximar un problema de la vida real**, que puede ser extremadamente complicado, **por un modelo mucho más simple. En general, los métodos más flexibles resultan tener menos sesgo.**

En general, a medida que usamos métodos más flexibles, la varianza aumentará y el sesgo disminuirá. La tasa relativa de cambio de estas dos cantidades determina si la PRUEBA MSE aumenta o disminuye.

A medida que aumentamos la flexibilidad de una clase de métodos, el sesgo tiende a disminuir inicialmente más rápido de lo que aumenta la varianza. Sin embargo, en algún momento el aumento de la flexibilidad tiene poco impacto en el sesgo, pero comienza a aumentar significativamente la varianza. Cuando esto sucede, la prueba MSE aumenta.

En el siguiente gráfico, **la línea azul el sesgo** y la **línea naranja representan la varianza.** La línea discontinua horizontal representa Var(ε), el error irreducible. **El rojo representa el MSE**, la precisión del método.




La relación entre el sesgo, la varianza y el conjunto de prueba MSE dada en la ecuación anterior y mostrada en la figura se conoce como la **compensación sesgo-varianza ( bias-variance trade-off)**

## Bibliografía

*Correlación no significa necesariamente causalidad*. (2021). https://www.statsmedic.com/correlation-does-not-mean-causation

James, G., Hastie, T., Tibshirani, R., Witten, D., & Friedman, J. (2013). Una introducción al aprendizaje estadístico - con aplicaciones en R. En *Elementos* (Vol. 1). https://doi.org/10.1007/978-1-4614-7138-7

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

Madhaven, A. (2020). Correlación vs causalidad: Comprenda la diferencia para su producto. En *amplitud*. https://amplitude.com/blog/causation-correlation

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

*Por qué la correlación no es causalidad*. (s.f.). Consultado el 18 de septiembre de 2021 en https://www.ibpsychmatters.com/why-correlation-is-not-causation

**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]

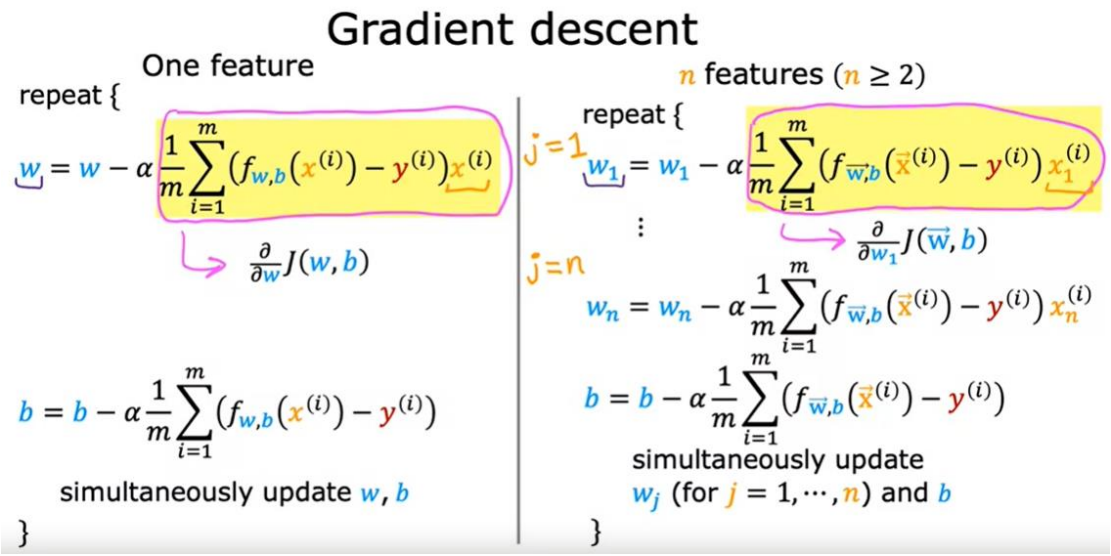
El MSE será: $J(\vec{w},b)$

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

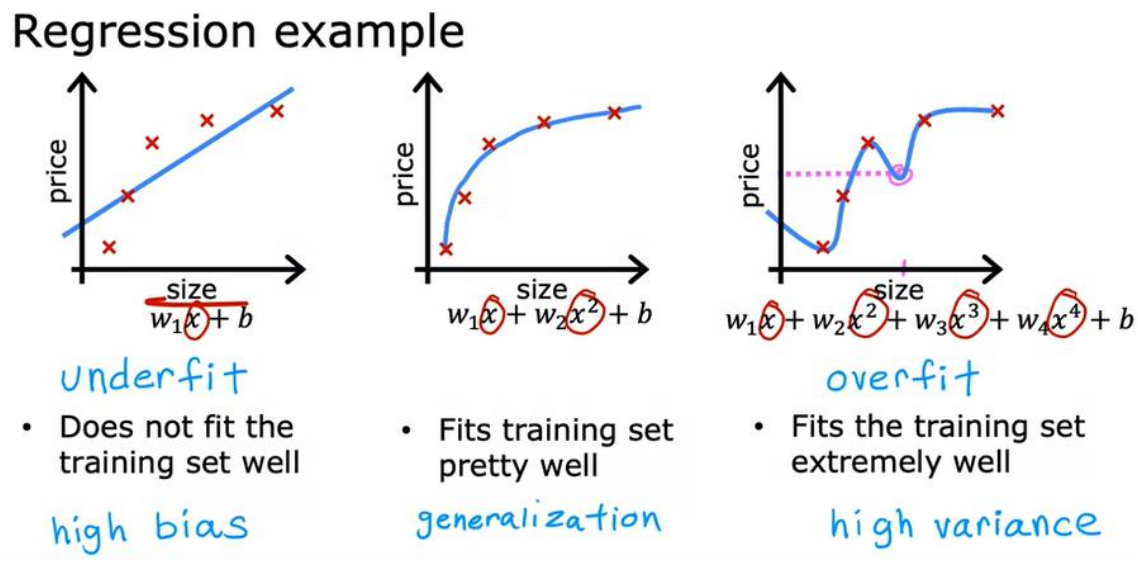
$$w_j := w_j - \alpha \frac{\delta}{\delta w_j} J(\vec{w},b) =$$

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

[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.

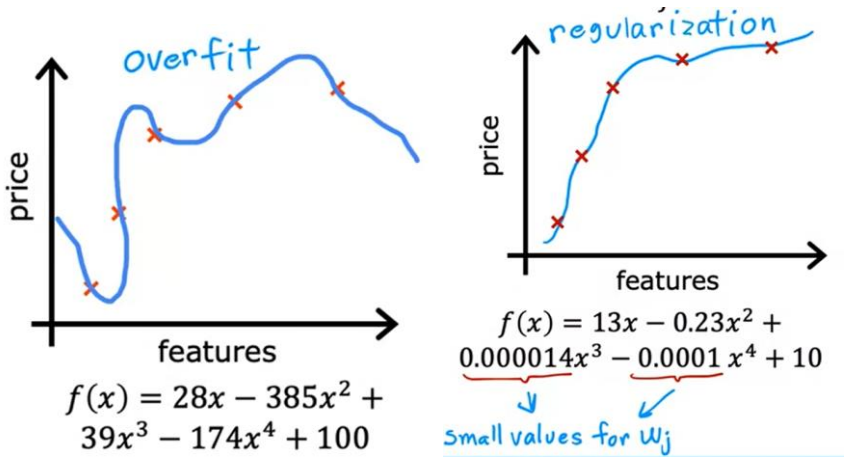
| size | bedrooms | floors | age | avg income | ... | distance to coffee shop | price |
|---|---|---|---|---|---|---|---|
| $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | | $x_{100}$ | $y$ |
| | | | | | | | |

**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*.

## 1. Tratamiento de datos

Se recopilan datos y se traducen en información usable

Pasos para este proceso;

- Descubrimiento de datos → es un proyecto en I+D.
- Adquisición o cobro
- Preparación
- Preprocesamiento
- Integración → proyecto de operación.
- Explotación (análisis, informe y visualización, acción)

  **Mugging de datos** → proceso inicial de refinar los datos para adecuarlos al usuario. Explotación, enriquecimiento, transformación e integración y validación de datos.

  - ☐ Front end → interfaz de usuario.
  - ☐ Back end → servidor, aplicación, base de datos…

  **Análisis - minería de datos:** proceso de predecir datos (resultados) a partir de grandes conjuntos de datos.

  **Informe y visualización:** diseñar la salida para que el usuario lo entienda y lo pueda usar.

## 2. Modelos de datos y tipos de datos.

Distinguimos:

- operaciones
- construcciones
- estructuras

Tipos de datos:

- Estructurado (20%) → longitud y formato definido (nº, fechas y grupos de palabras y números)

Modelo relacional → tecnología clave en la actualidad. Estructura y organiza grandes cantidades de datos de forma eficiente y efectiva. Usa SQL como lenguaje de consulta estructurado. Permite una gran flexibilidad en la gestión de datos y en la realización de consultas.

- Semiestructurado → datos con características consistentes y definidas. No se limita a una base rígida como los relacionales.
- Sin estructurar (80%) → no siguen un formato específico.

## 3. Gestión de datos

Recopila, mantiene y usa datos de forma segura, eficiente y rentable

Objetivo → ayudar a las personas/organizaciones a optimizar el uso de los datos y sacar el mayor provecho de ellos. También consiste en dar respuesta a los problemas que parecen hacer operativo un determinado proyecto de datos.

### Data management: Data storage

Persistencia → servicio más importantes que proporcionan las bases de datos operativos. Garantiza que los datos almacenados no serán modificados sin autorización y que estarán disponibles para el negocio.

Bases de datos relacionales → 1 o más relaciones representadas en tablas donde se almacenan los datos. Está formada por columnas y filas (la clave principal es la primera columna). → Esquema de base de datos.

SQL ha evolucionado en sintonía con la tecnología RDBMS.

CRUD: Crear, Recuperar, actualizar y eliminar son operaciones comunes y relacionadas que puede usar directamente una base de datos o a través de APIs.

NoSQL no se basa en el modelo tabla/clave de los RDBMS. Sus características son:

- Escalabilidad: capacidad de escribir datos en diferentes almacenes a la vez y sin limitaciones.
- Modelos de datos y consultas: usan marcos especializados para almacenar datos
- Diseño de persistencia: elemento más crítico de los NoSQL.
- Diversidad de interfaces
- Consistencia eventual: utilizan BASE en vez de ACID (RDBMS)

El almacenamiento de datos distribuidos consiste en una red donde los datos o información se almacenan en un nodo o computadora.

Las bases de datos distribuidas son las que recuperan rápidamente datos en muchos nodos.

Los almacenes de datos distribuidos tienen mayor disponibilidad y facilidad de acceso a escritura y lectura.

## Data management - Ingesta de datos.

Ingesta de datos: proceso de adquisición e importación de datos en un almacén de datos o una base de datos.

Si los datos se ingieren en tiempo real, cada registro se inserta en la base de datos a medida que se emite.

- Data in motion: Analizados a medida que se generan.
- Datos en reposo: recopilamos antes del análisis.
- Data Streaming: datos generados continuamente por miles de fuentes de datos que envían los registros simultáneamente en tamaños pequeños. En los data streaming systems, se computa en tiempo real un elemento de datos a la vez.

## Integración de datos

La integración de datos se ha centrado en el movimiento a través de middleware.

La integración de datos debe identificar:

- Origen de los datos
- Identificar las fuentes.
- Identificar formatos y lenguajes de scripting
- Recopilación de datos
- Datos de muestra

Las fuentes de datos no estructuradas necesitan moverse rápidamente a través de grandes distancias geográficas para su intercambio.

Para integrar datos en entornos de aplicaciones mixtas → obtener datos de un entorno de datos (origen) a otro entorno de datos (destino) (tecnologías de extracción, transformación y carga (ETL) se han usado para lograr esto en entornos de almacenamiento de datos tradicionales).

Las herramientas ETL se usan para transformar los datos en el formato requerido por el almacén de datos. La transformación se realiza en una ubicación intermedia antes de que los datos se carguen en un almacén de datos. ETL nos da la infraestructura subyacente para la integración mediante:

- Extraer: leer datos de bases de datos de origen
- Transformar: convertir en formato en uno que se ajuste
- Cargar: escribir los datos en la base de destino

La transmisión de datos (streaming data) y el procesamiento de eventos complejos son cada vez más importantes. La computación de transmisión está diseñada para manejar un flujo continuo de una gran cantidad de datos no estructurados.

## Recuperación de datos

Proceso de búsqueda, identificación y extracción de datos requeridos de una base de datos. Requieren escribir consultas o comandos de extracción de datos por parte de los usuarios en una base de datos.

Una base de datos está diseñada para hacer que los sistemas transaccionales se ejecuten de manera eficiente. Este tipo de bases de datos (OLTP) está diseñada para manejar transacciones pero no análisis.

Un almacén de datos (datawarehouse) es un tipo de base de datos que integra copias de datos de transacciones de sistemas de origen dispares y los aprovisiona para un uso analístico. Un datawarehouse es del tipo OLAP.

En un datawarehouse los datos se cargan en el almacén después de transformarlos en un formato bien definido y estructurado. Esto se llama esquema en la escritura.

Un data lake es un depósito de almacenamiento masivo con una enorme potencia de procesamiento y capacidad para manejar una gran cantidad de concurrencias, tareas… Un data lake garantiza que todos los datos se almacenen para un uso más adelante: esquema en escritura

Funciona de la siguiente manera: Los datos se cargan desde su fuente almacenada en su formato nativo hasta que se necesita, momento en el que las aplicaciones pueden leer libremente los datos y agregarle estructura.

Los datos se almacenan como un BLOB con identificador único.

**Infraestructura de almacenamiento y recuperación de datos.**
Jerarquía de memoria:

- El registro interno:
- La caché:
- RAM
- Disco duro
- Cinta magnética:

Las tecnologías emergentes de memoria principal no volátil (NVM) ofrecen una densidad de memoria mucho mayor, un costo por bit mucho menor y un consumo de energía en espera que la DRAM.8.

Sin embargo, la escalabilidad es una decisión entre hacer una máquina que hace que un servidor sea más potente versus agregar más máquinas.

El escalado vertical supone un mayor número de procesadores y RAM o el nuevo NVMM en el que cualquier operación funciona mejor con más memoria, pero su mantenimiento puede ser difícil y costoso y por supuesto posee limitaciones de crecimiento.
El escalado horizontal implica agregar más máquinas, posiblemente menos potentes, a una red relativamente más lenta. Las operaciones paralelas posiblemente serán más lentas, pero en la práctica es más probable añadir más máquinas.

## Calidad de los datos.

Calidad de los datos →
- Perfiles de datos.
- Análisis y estandarización de datos.
- Coincidencia de datos y limpieza de datos.

Generalización de perfiles de datos → proporcionan las métricas e informes que los propietarios de información necesitan.
Puede usar la generalización de perfiles de datos para:
- Analizar, clasificar e identificar los datos.

Análisis y estandarización de datos → capacidades de estandarización de datos.
Limpieza de datos → corregir los datos y que sean consistentes.
Coincidencia de datos → identificación de posibles duplicados para registro de cuentas, contactos…

## Seguridad de los datos

Seguridad → datos más sensibles → más seguridad.
Si un sistema de big data se implementa en la nube, necesitamos asegurar:
- Máquinas.
- Transferencia de datos mediante diferentes fases de la operación de datos.

**Técnicas de protección de datos**

- Cifrado
- Anonimización de datos
- Tokenización
- Controles de bases de datos.

## 4. Ejemplos de algunas arquitecturas posibles

La más simple es un sistema con RMDB. Se puede organizar de diferentes maneras;
- **1 nivel** → se colocan todos los componentes necesarios para una app o tecnología de software en un solo servidor o plataforma.
- **2 niveles** → arquitectura del servidor del cliente. Comunicación directa entre el cliente y el servidor. No existe intermediario.
- **3 niveles** → se separan sus niveles entre sí en función de la dificultad de los usuarios y cómo usan los datos presentes en la base de datos. Arquitectura más usada para diseñar un DBMS. Sus tres niveles son:
    - Base de datos
    - Aplicación: servidor de apps y programas que acceden a la base de datos. Media entre el usuario final y la base de datos.
    - Usuario: la aplicación proporciona varias vistas de la base de datos. Estas vistas son generadas por las apps que residen en el nivel de aplicación.

Para cosas más complejas tenemos que coordinar diferentes bases de datos y fuentes → DATAWAREHOUSE. Cuenta también con 3 niveles
- **Nivel inferior:** sistema de base de datos relacional. Se limpian los datos mediante herramientas de back - end.

- **Nivel intermedio:** Es un servidor OLAP, se presenta una vista abstracta de la base de datos. Media entre el usuario final y la base de datos.
- **Nivel superior**: capa de cliente front-end. Herramientas y API que conectan y se obtienen datos del almacén de datos (herramientas de consulta, de informes, de consulta administradas, de análisis y de minería de datos.

Modelo de Gartner:

- **Adquirir**: recoge todo tipo de datos útiles.
- **Organizar**: organiza de extremo a extremo, eso es LDW:

- Proporciona una arquitectura de gestión de datos moderna y escalable bien localizada para satisfacer las necesidades de datos y análisis de la empresa.
- Admite un enfoque de desarrollo que aprovecha la arquitectura y técnicas de almacenamiento de datos empresarial existentes en la organización.
- Establece una capa de acceso a datos compartidos que relaciona lógicamente los datos, independientemente del origen.

- **Análisis de la arquitectura de extremo a extremo** se puede ver dificultada por el aumento de la demanda.

## 1. Big data stack

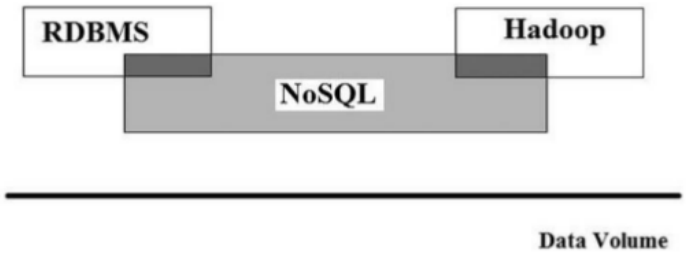
**Stack → grupo de herramientas e infraestructuras integradas.**
El entorno debe estar formado por:

- Consideraciones de hardware.
- software de infraestructura.
- software operativo.
- software de administración.
- interfaces de programación de apps (API) bien definidas e incluso herramientas de desarrollo de software.

La arquitectura tiene que ser capaz de realizar todos los **requisitos fundamentales:**

- Ingerir
- Integrar
- Organizar y almacenar
- Análisis
- Actuar o entregar o visualizar.
- Asegurar el sistema.

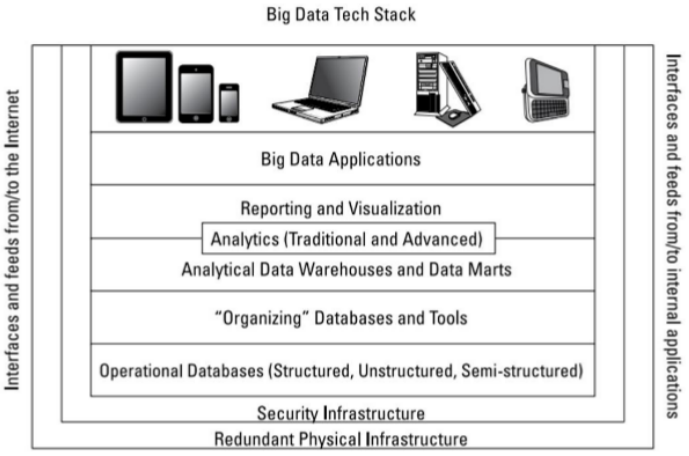
El big data produce algunos problemas de escala específicos para esta infraestructura.
**¿Donde se encuentran las bases de datos relacionales, las bases de datos NoSQL y el sistema de Big data Hadoop en la escala de datos?**




Se muestra el volumen de datos.

**RESUMEN BIG DATA STACK**

**Capa 0** → infraestructura física; hardware, red…
**Capa 1** → requisitos de seguridad y privacidad.
**Capa 2** → bases de datos operativos
**Capa 3** → organización de los servicios y herramientas de captura de datos. En esta capa se incluyen estas tecnologías.



- Un sistema de archivos distribuido: adapta la descomposición de los flujos de datos y para proporcionar escala y capacidad de almacenamiento.
- Servicios de serialización: convierte un objeto de datos en una serie de bytes que pueden ser fácilmente transmitidos a otro destino. Esto es útil para transferir objetos de datos entre diferentes sistemas o aplicaciones.
- Servicios de coordinación: para crear aplicaciones distribuidas.
- Herramientas de extracción, transformación y carga (ETL): para datos estructurados y no estructurados en Hadoop.
- Servicio de flujo de trabajo: para programar trabajos y proporcionar una estructura para sincronizar elementos de proceso entre capas.

**Capa 4** → los almacenes de datos analíticos y data marts.
*Data mart:* parte de un almacén de datos que se enfoca en una línea de negocio particular. Estos data marts contienen información resumida y específica para un área de la organización.
**Análisis de Big Data:** los algoritmos tienen que ser capaces de trabajar con muchos datos en tiempo real y con gran variedad. 3 clases de herramientas:

- Informes y paneles
- Visualización: evolución de los cuadros de mando.
- Analítica avanzada: predicciones, recomendaciones, inferencias, tendencias…

**Aplicaciones del Big Data:**

- horizontales, tratan problemas comunes en todas las industrias.
- verticales, ayudan a resolver problemas específicos.

## 2. Virtualización

- Proceso de crear una representación virtual o basada en software de algo, como aplicaciones virtuales, servidores, almacenamiento y red.
- Separa los recursos y servicios del entorno de entrega físico subyacente, permite crear muchos sistemas virtuales dentro de un solo sistema físico.
- Proporciona una base para la computación en la nube.
- Al optimizar todos los elementos de su infraestructura se obtiene la eficiencia necesaria para procesar y administrar grandes volúmenes de datos estructurados y no estructurados. Big Data → distribución.

Tiene **3 características que soportan la escalabilidad y eficiencia operativa necesaria para entornos de Big Data:**

- Particionamiento: separación de los recursos disponibles.
- Aislamiento: máquina virtual separada del host, un error no afecta a todo.
- Encapsulación: máquina virtual como un solo archivo.

**¿Qué podemos virtualizar?**

- Servidores
- Procesador
- Infraestructura de apps .
- Redes

## 3. Principios de DFS

Cuando los datos superan la capacidad de almacenamiento de una sola máquina, se dividen en varias máquinas. Esto se llama un sistema de archivos distribuidos y es más complejo debido a la programación de red. Hadoop tiene un sistema de archivos distribuidos llamado HDFS.
Un sistema de archivos distribuido (DFS), da la solución para los problemas que provoca una arquitectura de este tipo.

- Primer problema a resolver → **fallo del hardware.**

Se pueden evitar la pérdida de datos mediante la replicación (copia de los datos). Permite también una alta concurrencia (acceso a la vez de muchos usuarios) y estas dos cosas provocan la falta de consistencia.

- Segundo problema → **se necesitan combinar los datos de alguna forma en las tareas de análisis.**

**¿Qué es un paralelismo de datos?**
Dividir los datos entre varios procesadores en sistemas de computación paralela.
Se enfoca en distribuir los datos a través de diferentes nodos de computación para que cada procesador pueda realizar la misma tarea en diferentes partes de los datos.
El paralelismo de tareas se enfoca en dividir las operaciones que se deben realizar.

## 4. Entorno hadoop

**Hadoop**

- Plataforma confiable y escalable para almacenamiento y análisis. Se ejecuta en hardware básico.
- Diseñado para procesar grandes cantidades de datos estructurados y no estructurados.
- Es autoreparable → los servidores se pueden agregar o quitar del clúster dinámicamente.
- Se desarrolló porque representaba la forma más práctica para que las empresas puedan administrar grandes cantidades de datos mediante el paralelismo de datos. Esto facilita el manejo de grandes volúmenes de información.

**2 componentes de hadoop:**

- Hadoop distributed file system: clúster de almacenamiento de datos de bajo coste que facilita la administración de archivos relacionados en todas las máquinas.
- Motor mapreduce: implementación de procesamiento de datos paralelo / distribuido de alto rendimiento.

## 5. HDFS

- Es una parte de Hadoop que se encarga de almacenar los datos en múltiples computadoras.
- Datos más seguros y fáciles de acceder.
- Se dividen los archivos grandes en piezas más pequeñas llamadas bloques y los almacena en diferentes nodos de datos.
- NameNode es el encargado de controlar todo el acceso a los archivos y asegurarse de que los datos estén en su lugar correcto. Los metadatos son información sobre los datos, como el nombre y la ubicación del archivo. HDFS los utiliza para mantener un registro de todos los archivos y directorios almacenados en el clúster.

## 6. MAP REDUCE

Hadoop MapReduce es un algoritmo implementado en el proyecto Apache Hadoop que funciona como un motor. Convierte la entrada en salida de forma eficiente y rápida, lo que permite obtener las respuestas necesarias. Es como un motor que necesita combustible para funcionar.
El algoritmo funciona de la siguiente manera:

- Se sube un archivo y se divide en pedazos para procesarlo.
- Se asignan los datos: asignar pares de valores clave a los elementos de las piezas.
- Corto y aleatorio: organiza las piezas haciendo posible al mismo tiempo:
  Equilibrar el número de piezas en cada ordenador.
  Piezas homogéneas en cada ordenador.
- Reducir: realizar la tarea.

## 7. Plataforma de BIG DATA: arquitectura.

Tiene que consumir incontables fuentes de datos de una manera rápida y económica.

Tiene que tener las siguientes capas.

- Fuentes de datos
- Capa de ingestión
- Capa de visualización
- Capa de administración de la plataforma Hadoop
- Capa de almacenamiento de Hadoop
- Capa de infraestructura de Hadoop
- Capa de seguridad
- Capa de monitoreo

## 1. Aprendizaje supervisado VS no supervisado

Aprendizaje estadístico → dos categorías:

- supervisado:
- no supervisado:

Relación entre las variables:

- Clustering
- PCA: resumir varias variables en un pequeño nº de ellas.

## 2. Agrupamiento o clustering (muy útil para la segmentación en marketing).

Conjunto amplio de técnicas para encontrar subgrupos o clústeres en un conjunto de datos. La similitud o diferencia entre observaciones debe definirse según el dominio y el conocimiento de los datos.
La agrupación busca subgrupos homogéneos entre las observaciones.
Otra manera de buscar la homogeneidad: densidad de las relaciones mediante la maximización de modularidad.
Otras metodologías buscan grupos no superpuestos y particiones superpuestas existentes.

## 3. KMEANS

Enfoque simple y elegante para particionar un conjunto de datos en K Clusters distintos y no superpuestos.

¿Cómo especificamos el número deseado de cúmulos KMEANS?

- Especificamos el nº deseado de cúmulos K.
- El algoritmo KMEANS asignará cada observación exacta a cada uno de los cúmulos K.

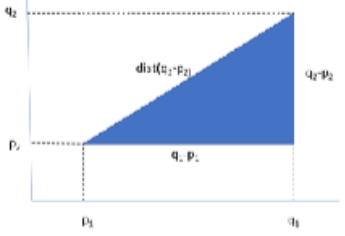
Buena agrupación → es aquella para la cual la variación dentro del clúster es lo más pequeña posible. La variación dentro del clúster para el grup oCk es una medida W(Ck) de la cantidad en la que las observaciones dentro de un clúster difieren entre sí.
Buscamos una división que haga que la variación interna sea la mínima → la definimos con la distancia euclidiana al cuadrado.

$$J(c,\mu) = \sum_{1=1}^{m} \|x^i - \mu_c(i)\|^2$$

Enfoque simple → permite encontrar esa división. Los centroides son la media de las observaciones en un cúmulo.

$$dist\ (p,q) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2}$$

Disminuye la variación interna y mejora hasta que no haya cambios, pero termina en un óptimo local.



## 4. Limitaciones del KMEANS

El modelo K-means clustering presenta dos limitaciones principales:

- la elección del número de grupos K.
- la posibilidad de que la solución encontrada sea un óptimo local en lugar de un óptimo global.

Para aplicar el modelo, es necesario estandarizar la escala de las diferentes características.

- La elección de K no es simple y existen metodologías para comparar y validar los resultados.
- El problema del óptimo local puede resolverse ejecutando el algoritmo varias veces con diferentes centroides y seleccionando el que tenga la medida de variación interna más baja.

Es una buena práctica estandarizar los datos usando la misma escala o al menos intentar que sean lo más homogéneos posible y hacerlos centrados en la media.

## 5. Ejemplos del clustering

Muchos de los ejemplos mezclan la clasificación y la agrupación. Para clasificar algo usamos algoritmos de aprendizaje supervisado.

- Segmentación de marketing.
- noticias falsas
- detección de fraudes
- las imágenes médicas segmentadas.

## 1. Analítica y análisis de textos, búsqueda y recuperación de información

Texto y multimedia → tipo de dato no estructurado → estructura impredecible.

``Datos no estructurados" son datos que no tienen una estructura clara, semánticamente abierta y fácil de usar. Por ello no se pueden almacenar en una base de datos tradicional o RDBMS. Casi ningún dato es realmente ``no estructurado´´.

Para poder analizar un texto tenemos que transformarlo en vectores, matrices o cosas parecidas (lenguaje que entienda el ordenador).

Podemos hacer algunas distinciones →

- La analítica de textos → proceso de análisis de texto no estructurado, se coge info importante, proporciona información cuantitativa.
- Minería o análisis de texto → se obtiene información cualitativa mediante texto no estructurado, proporciona información cualificada.

Búsqueda y recuperación de información; es encontrar material (datos no estructurados) para satisfacer unas necesidades.

**El proceso del lenguaje natural (NLP)**

Es una rama de la IA que ayuda a los ordenadores a entender, interpretar y manipular el lenguaje humano.

## 2. Tipos de tareas y aplicaciones

Podemos realizar varias acciones sobre un conjunto de documentos o textos.

- Indexación de documentos.
- Resumen de documentos o un grupo de ellos.
- Etiquetado.
- Recuperar documentos o textos de una base de datos.
- Clasificación de documentos en varios grupos.
- Búsqueda de los grupos o documentos de agrupación.
- Comprender el sentimiento de un texto y contexto.
- Procesamiento avanzado del lenguaje natural (NLP).

  Traducir a otro idioma, comprender y responder preguntas.

- Gestión de riesgos.
- Comentarios de los clientes, escucha social y aplicaciones centradas en las personas.
- Publicidad mediante medios digitales.
- Enriquecimiento de contenido.
- Filtrado de spam.
- Prevención de delitos cibernéticos.

## ③ Clasificación de textos, análisis de sentimiento y recuperación de información: Clasificador de Bayes

Tenemos un conjunto de clases → buscamos a qué clase pertenece el objeto.

· Temas → clases generales
Clasificador de texto → tarea de clasificar los textos.

* Usaremos el algoritmo de Bayes para clasificar texto.

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \rightarrow \text{Método de aprendizaje probabilístico.}$$

Probabilidad de que un documento (d) esté en la clase (c);

$$P(c|d) = P(c) \times \prod P(t_k|c) \cdot (1)$$

$P(c|d)$ → d o cma en c  $P(c)$ = probabilidad previa.

- Probabilidad previa

$$P(c) = \frac{Nc}{N\,docs} = \frac{N^o \text{ documentos clase } c}{N^o \text{ documentos en total}}$$

- Probabilidad condicional

$$P(c|d) = \frac{(Tct) + 1}{(T'ct) + (B)}$$

(Tct) → N° de veces que aparece t en los documentos de la clase C

(T'ct) → N° total de palabras (longitud) de cada clase.

(B) → N° de términos diferentes en el vocabulario.

Otras maneras de calcularlo.

1- Bernoulli
2- Máquina vectorial de soporte
3- Redes neuronales.

## 4. Recuperación de información: Modelo de espacio vectorial por puntuación.

Representación de un conjunto de documentos como vectores → modelo de espacio vectorial.

Proceso para puntuar y variar un conjunto de documentos: Pasos

- Caracterización de la base de datos de documentos
a) construcción de la matriz de frecuencias del conjunto de documentos.
- Ponderación de las palabras relevantes:
- Frecuencia inversa del documento.
- Término frecuencia del documento x matriz de frecuencia inversa del documento.
- Caracterización de la consulta: vector frecuencia.
- Clasificamos los documentos de la similitud más alta a la más baja.

**Caracterización de la base de datos de documentos.**

Frecuencia del documento → nº de veces que cada palabra se repite en estos documentos. Cuanto más frecuente más relevante. Como hay muchos elementos repetidos, necesitamos eliminar aquellos que no aporten. Para ello:

Frecuencia inversa del documento

$$idf_t = \log \frac{N}{df_t}$$

$N$ → Número de documentos en la base de datos.

$df_t$ → Frecuencia del término en todos los documentos

idf de un término raro → alto
idf de un término frecuente → bajo

Frecuencia inversa + Frecuencia del término

$tf \cdot idf$

Caracterización de la consulta → vector de frecuencias de término.

$tf \cdot idf$

matriz term frecuencia

Similitud $d_1$ y $d_2$ → calcular similitud del coseno

$$sim(d_1, d_2) = \frac{V(d_1) \cdot V(d_2)}{\|V(d_1)\| \cdot \|V(d_2)\|}$$

$V(d_1) \cdot V(d_2)$ → producto de puntos de los vectores $d_1$ y $d_2$

$\|V(d_1)\| \cdot \|V(d_2)\|$ → producto de las longitudes euclidianas de los vectores del documento. Normaliza la longitud del vector

a mayor ángulo → menor coseno = menos similitud entre el documento y la consulta