Universidad Popular del Cesar

Procesamiento de Señales II
Ciencia de Datos II

Aprendizaje Supervisado

Dr. Ing. José Ramón Iglesias
DSP-ASIC BUILDER GROUP
Director Semillero TRIAC
Ingenieria Electronica
Universidad Popular del Cesar

Temario del Curso

- Introducción al ML
- Etapas en la aplicación del ML
- Aprendizaje supervisado.
  - Repaso: Regresión Lineal y Polinomial, Regresión Logística, Naive Bayes.
- Support Vector Machines.
  - Repaso: Perceptrón.
  - SVC/SVR. Datos no linealmente separables. Función de costo.
- Ensemble learning.
  - Repaso: Decision Trees
  - Random Forest, Bagging, Boosting, Voting.
- Redes neuronales.
  - Perceptrón multicapa.
- Sistemas de recomendación.
  - Filtrado colaborativo.
- Prácticas

Primera Clase

Primera clase: 19/05/2023

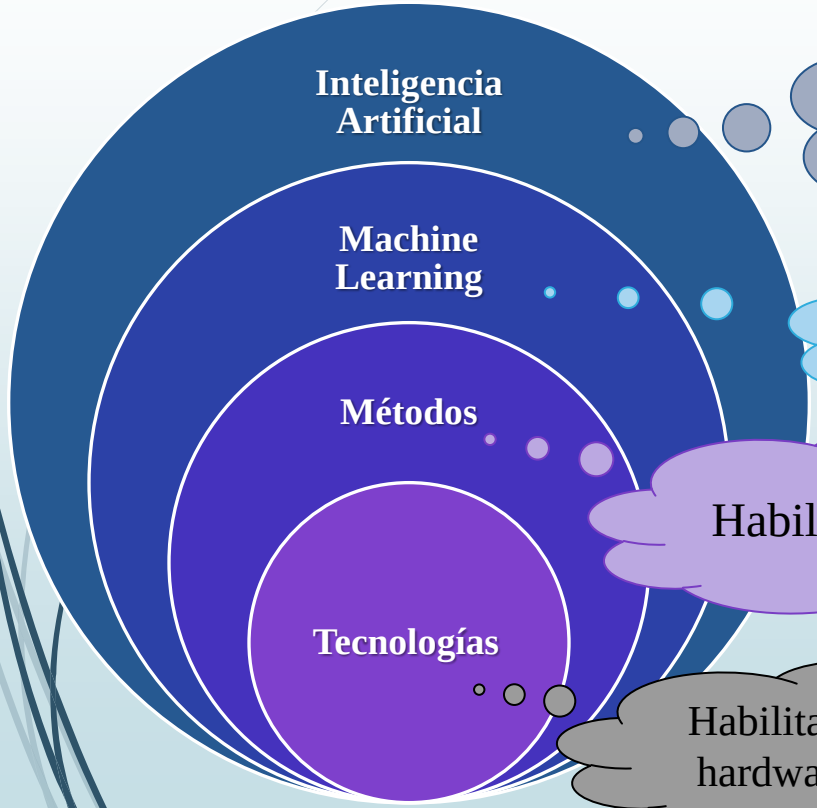
- Introducción al ML
- Etapas en la aplicación del ML
- Aprendizaje supervisado.
  - Repaso: Regresión Lineal y Polinomial, Regresión Logística, Naive Bayes.
- Support Vector Machines.
  - Repaso: Perceptrón.
  - SVC/SVR. Datos no linealmente separables. Función de costo.
- Ensemble learning.
  - Repaso: Decision Trees
  - Random Forest, Bagging, Boosting, Voting.
- Redes neuronales.
  - Perceptrón multicapa.
- Sistemas de recomendación.
  - Filtrado colaborativo.
- Prácticas

Introducción al Machine Learning

El aprendizaje automático es el proceso que le da a las computadoras la habilidad de aprender sin ser explícitamente programadas.
A.L Samuel

Se dice que un programa de computación aprende de la experiencia E con respecto a una tarea T y alguna medida de rendimiento P, si es que el rendimiento en T, medido por P, mejora con la experiencia
E.T.M Mitchell

¿Cómo se relacionan la IA y el ML?



Capacidad de censar, razonar, relacionar y aprender

- Visión por computadora
- Procesamiento del lenguaje natural
- Reconocimiento de voz
- Robótica y movimiento
- Planificación y optimización
- Simulación de conocimiento

Habilidad de aprender

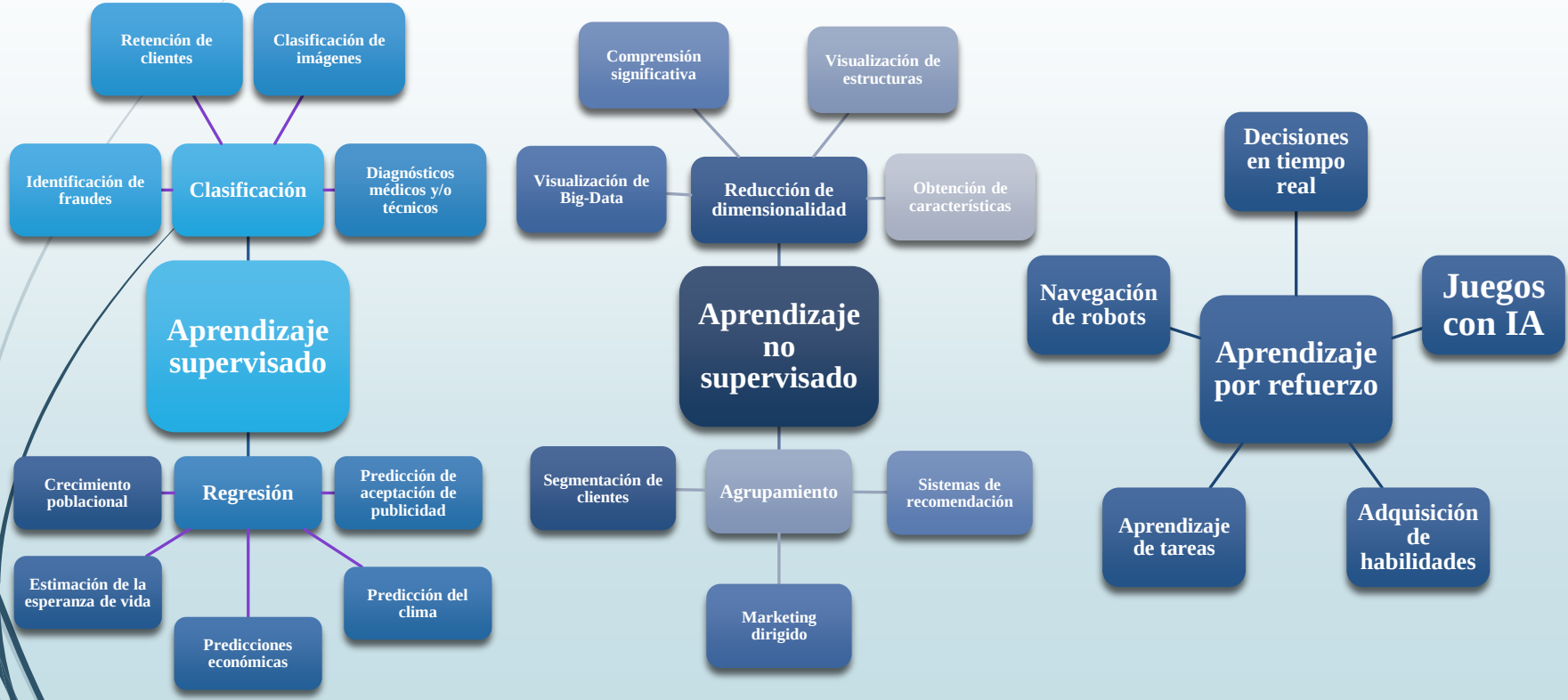
- Aprendizaje supervisado,
- Aprendizaje no supervisado
- A. por refuerzo

Habilidad de razonar

- Regresión
- Árboles de decisión
- ANN
- SVM
- Ensemble Learning

Habilitación física de hardware y software

¿Qué tipo de aprendizaje automático o Machine Learning hay?



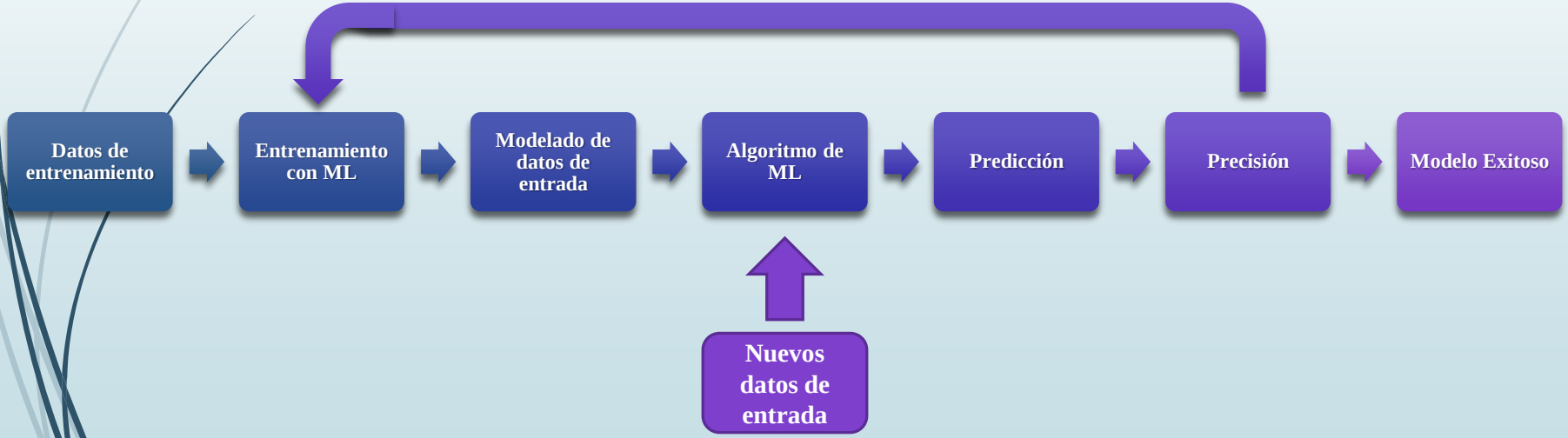
Etapas en la aplicación del Aprendizaje Supervisado

- Definición del problema
- Análisis detallado de los datos
- Definición de métricas de éxito y fracaso
- División del conjunto de datos (entrenamiento y test)
- Procesar los datos
- Diseñar un modelo "blando" para ver que ande
- Ajustar los parámetros del modelo a lo deseado
- Analizar la evolución del entrenamiento
- Entrenar con los requerimientos definidos
- Testear el sistema
- En caso de no obtener resultados aceptables, **volver al principio**

Descripción del problema: Aprendizaje supervisado

Datos: Se dispone de un conjunto de registros (o ejemplos, o instancias) descritos por *n* atributos: A_1, A_2, ... , A_n y cada instancia está anotada con una etiqueta, pudiendo ser una clase o un valor numérico.

Objetivo: Aprender un modelo (o función) a partir de los datos, buscando predecir sus etiquetas a partir de los atributos. Este modelo puede ser utilizado para predecir las etiquetas de nuevos registros sin anotar.



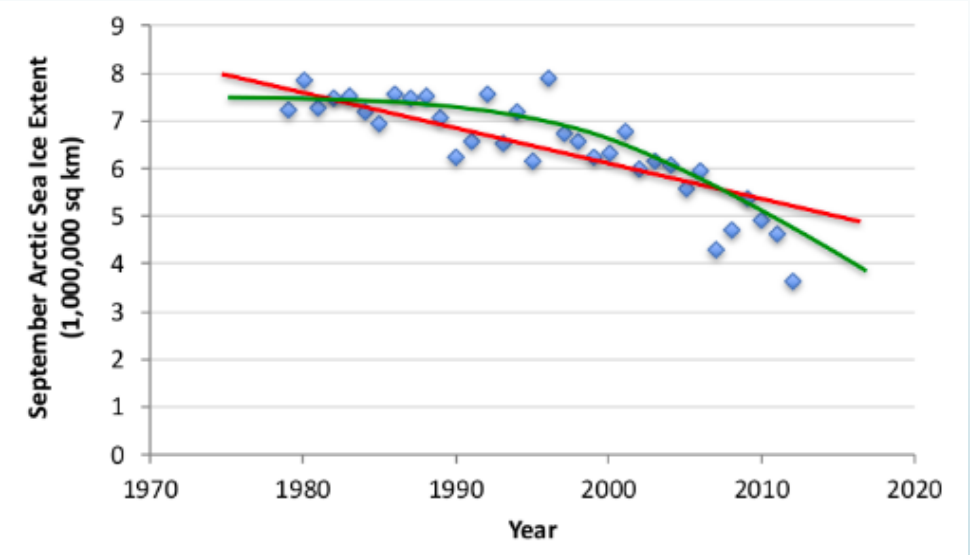
Aprendizaje Supervisado

Regresión

Dados (x_1,y_1), (x_2, y_2), …, (x_n, y_n)

Aprender una *f(x)* que permita predecir y a partir de *x*

Si $y \in \mathbb{R}^n$: Es un problema de **regresión**.



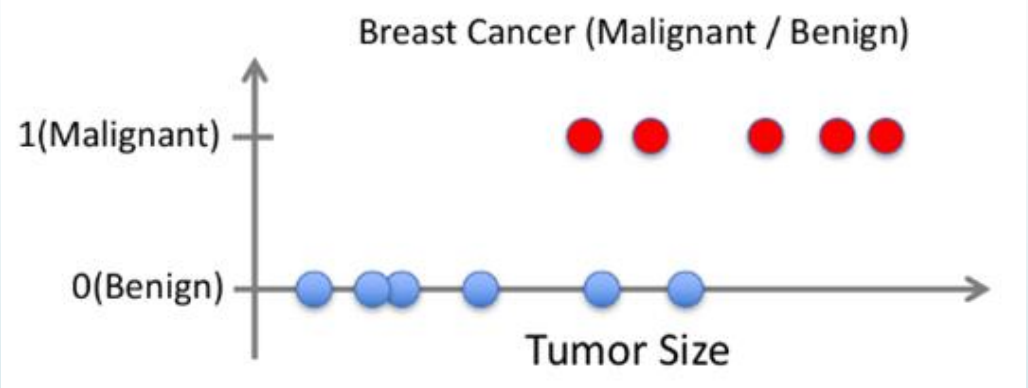
Filminas del curso previo

Clasificación

Dados (x_1,y_1), (x_2, y_2), …, (x_n, y_n)

Aprender una *f(x)* que permita predecir y a partir de *x*

Si *y* es categórica: Es un problema de **clasificación**.



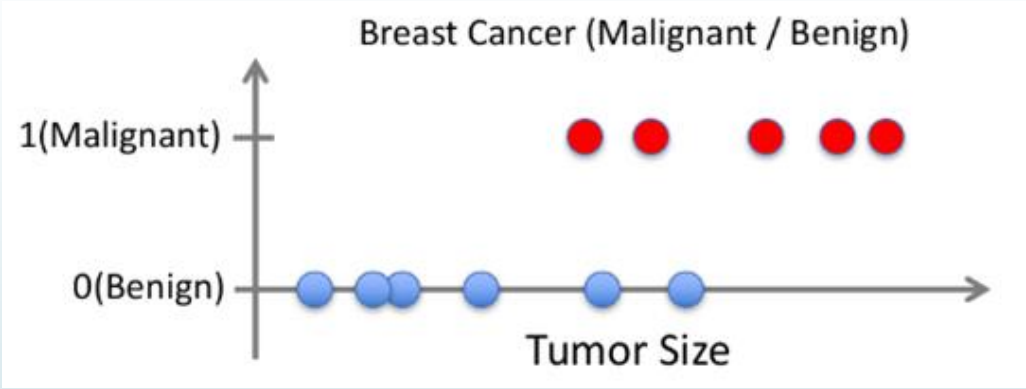
Filminas del curso previo

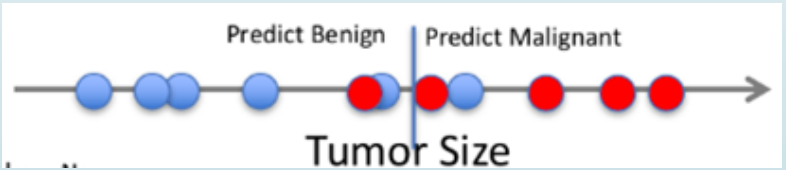
Clasificación

Dados (x_1,y_1), (x_2, y_2), …, (x_n, y_n)

Aprender una *f(x)* que permita predecir y a partir de *x*

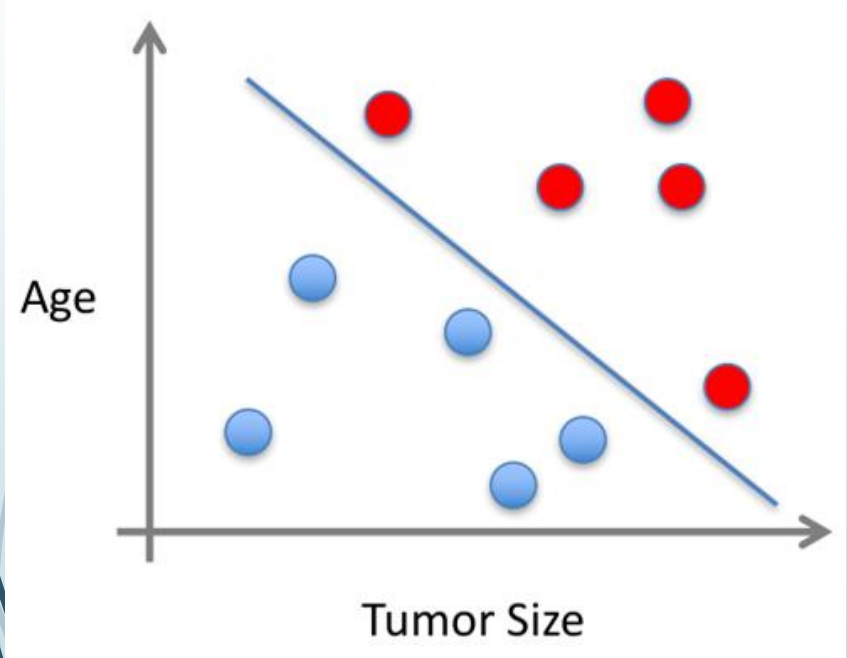
Si *y* es categórica: Es un problema de **clasificación**.





Filminas del curso previo

Aprendizaje Supervisado



- La variable ***x*** puede ser multidimensional.
- Cada dimensión corresponde a un atributo:
  - Edad del paciente
  - Tamaño del tumor
  - Uniformidad en la forma de la célula
  - Etcétera
- La regresión busca “acercar” los datos a una función (lineal, polinomial, etc.)
- La clasificación busca separar los datos mediante ciertos “bordes”.

Filminas del curso previo

Elección de *hiperparámetros*

Se tiene la base de datos con toda los datos. Dividir el conjunto total de ejemplos en tres subconjuntos

- **Entrenamiento**: aprendizaje de variables del modelo
- **Validación**: ajuste/elección de hiperparámetros
- **Evaluación**: estimación final del desempeño del modelo entrenado (y con hiperparámetros elegidos adecuadamente)

Regresión Lineal y Polinomial

Regresión Lineal

Busca ajustar los datos de entrenamiento mediante una función que sea un hiperplano.

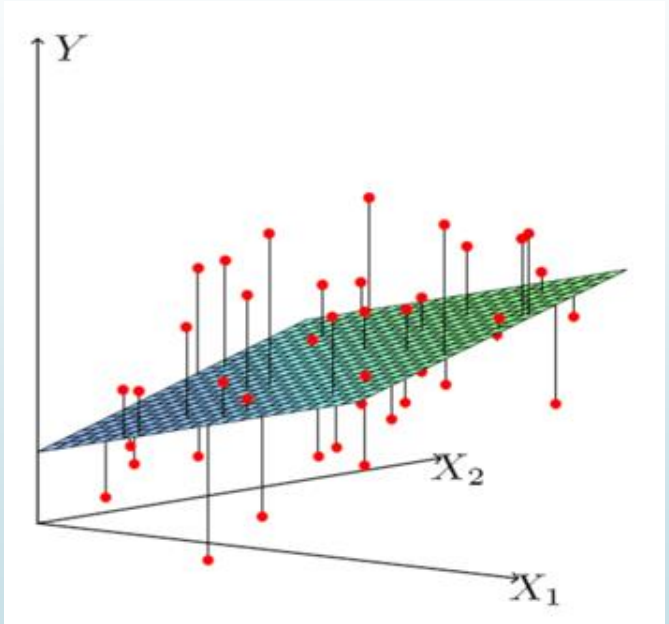
$$y = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ \ldots \ + \theta_d x_d = \sum_{j=0}^{d} \theta_j x_j$$

Los valores θ son los pesos de los atributos o *features*.

Se entrena minimizando la suma del error cuadrático.

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

Se resuelve mediante $\min_\theta J(\theta)$

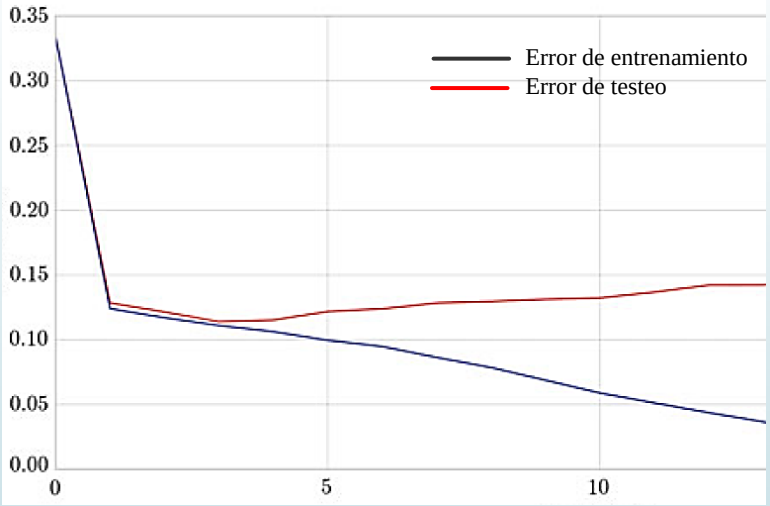


Regresión Polinomial

Busca ajustar los datos de entrenamiento mediante una función polinomial:

$$y(x, W) = w_0 + w_1 x + w_2 x^2 + \cdots + w_M x^M$$

$$= \sum_{j=0}^{M} w_j \, x^j$$



Mientras más alto el grado del polinomio, más se ajusta a los datos (pero se vuelve más complejo y tiende a sobreajustar).

Demo Time (demo_1_linear_regression)

Regresión Logística

Regresión Logística

Usa un enfoque probabilístico.

$h_\theta(x)$ debería devolver $p(y = 1|x; \theta)$

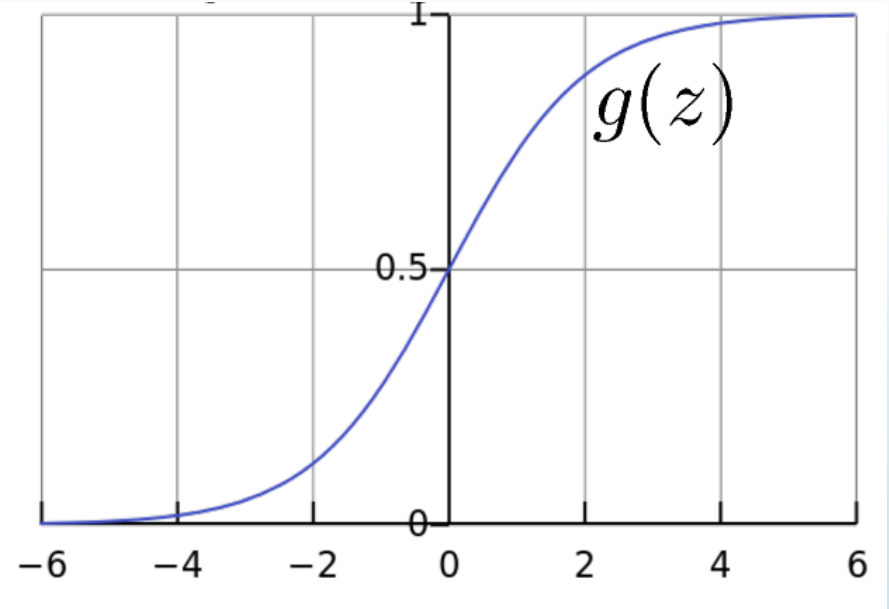
Como $0 \leq h_\theta(x) \leq 1$

Modelo de regresión logística

$$h_\theta(x) = g(\theta^T x)$$

$$g(z) = \frac{1}{1 + e^{-z}}$$

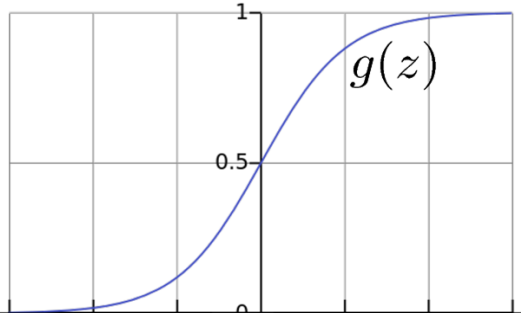
$$h_\theta(x) = \frac{1}{1 + e^{-\theta^t x}}$$



Regresión Logística

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = g\left(\boldsymbol{\theta}^{\mathsf{T}}\boldsymbol{x}\right)$$

$$g(z) = \frac{1}{1+e^{-z}}$$

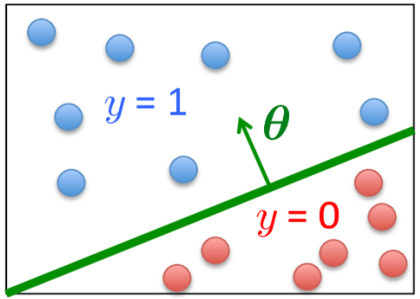


$\boldsymbol{\theta}^{\mathsf{T}}\boldsymbol{x}$ debería tener *valores* ***negativos*** *grandes para instancias negativas* y *valores* ***positivos*** *grandes para instancias positivas*.

Definir un umbral y...

Predecir : $y = 1$ si $h_{\boldsymbol{\theta}}(\boldsymbol{x}) \geq 0.5$

Predecir : $y = 0$ si $h_{\boldsymbol{\theta}}(\boldsymbol{x}) < 0.5$



Based on slide by Andrew Ng

Regresión Logística: Función de costo

$$J(\theta) = -\sum_{i=1}^{n} \left[y^i \log h_\theta\left(x^i\right) + \left(1 - y^i\right) \log\left(1 - h_\theta\left(x^i\right)\right) \right]$$

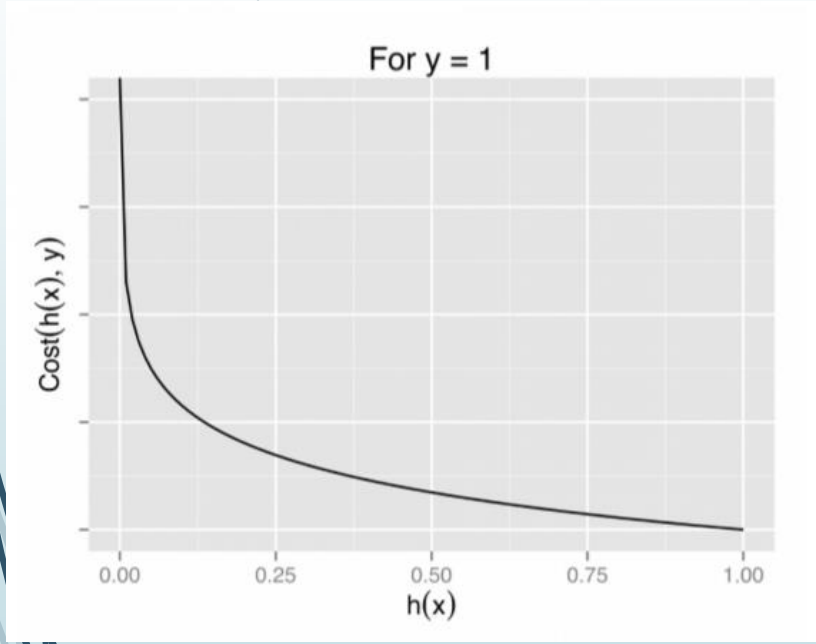
El costo de una sola instancia de los dato se define como:

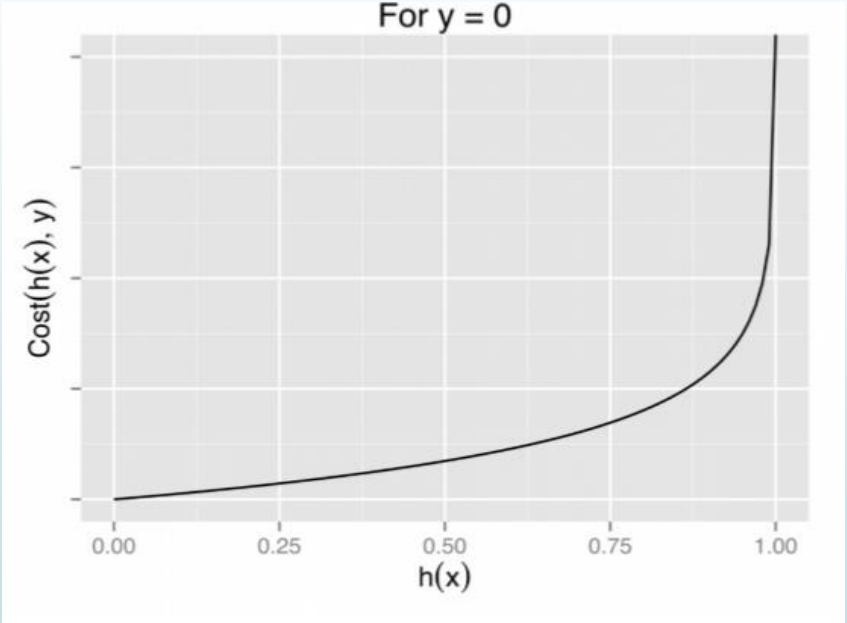
$$cost(h_\theta(x), \mathrm{y}) = \begin{cases} -\log\left(h_\theta(x)\right) & si\ y = 1 \\ -\log\left(1 - h_\theta(x)\right) & si\ y = 0 \end{cases}$$

Reescribimos la función de costo como:

$$J(\theta) = -\sum_{i=1}^{n} \left[cost\left(h_\theta\left(x^i\right), \mathrm{y}^i\right) \right]$$

Comportamiento de la función de costo





Demo Time (demo_2_logistic_regression)

Naive Bayes

Naive Bayes

- Es un clasificador basado en el teorema de Bayes, con una asunción "*inocente*" sobre los datos.
- Son "*naive*" porque se asume que las variables predictoras son independientes entre sí.
- Es muy sencillo de programar y entender.
- Sirve mucho como línea base de comparación, puede tener resultados que sobrepasan a algoritmos mucho más complejos.
- Es rápido de entrenar y funciona con datos de mucha dimensionalidad (e.g. es muy útil a la hora de clasificar documentos).

Teorema de Bayes

El algoritmo de "***Naive Bayes***" está fuertemente ligado al teorema de Bayes.

El Teorema de Bayes establece:

$$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)}$$

El teorema establece que se puede encontrar la probabilidad de **A** (e.g. una clase objetivo) dada la ocurrencia de **B** (e.g. un conjunto de features). Es decir, **B** es la evidencia y **A** es la hipótesis.

Naive Bayes: Asunciones

1. Bajo el teorema de Bayes, la principal asunción es que los atributos son independientes entre sí. Esto significa que la presencia de una característica no afecta a las otras. Esta asunción es "***naive***", por eso el nombre del algoritmo.
2. Una segunda asunción, es que todos los atributos tienen el mismo efecto en la salida del algoritmo.

Naive Bayes: Utilizando el Teorema de Bayes

En base a lo establecido, se puede utilizar el teorema de Bayes para calcular la probabilidad de una clase $\boldsymbol{y}$ de la siguiente manera:

$$\boldsymbol{P(y|X) = \frac{P(X|y)\,P(y)}{P(X)}}$$

Donde $\boldsymbol{y}$ representa la clase y $\boldsymbol{X}$ representa el vector de atributos:

$$\boldsymbol{X = (x_1, x_2, \ldots, x_n)}$$

Naive Bayes: Utilizando el Teorema de Bayes

Utilizando sustitución en la fórmula anterior obtenemos:

$$P(y|x_1, \dots, x_n) = \frac{P(x_1|y)\, P(x_2|y)\ \dots P(x_n|y)\, P(y)}{P(x_1)\, P(x_2)\ \dots P(x_n)}$$

Dado que el denominador es estático para todas las entradas del conjunto de datos se puede ignorar y se establece una proporcionalidad:

$$P(y|x_1, \dots, x_x)\ \propto P(y) \prod_{i=1}^{n} P(x_i|y)$$

En base a lo visto previamente, la predicción de la clase objetivo es sencillamente aquella con mayor probabilidad:

$$y = argmax_y\, P(y) \prod_{i=1}^{n} P(x_i|y)$$

Naive Bayes: Algoritmo

- Convertir el conjunto de datos en tablas de frecuencias.
- Crear una tabla de probabilidad calculando las correspondientes de cada evento.
- Calcular la probabilidad para cada clase

$$\boldsymbol{P(y)/\forall\, y \in Y}$$

- Calcular la probabilidad condicional de cada atributo dada cada clase:

$$\boldsymbol{P(x_i|y)/0 \leq i \leq n\,, \forall\, y \in Y}$$

- Calcular la clase de acuerdo a la que maximice la probabilidad (o, en este caso la proporción): $\boldsymbol{y = argmax_y\, P(y)\ \prod_{i=1}^{n} P(x_i|y)}$

Puntos fuertes y débiles de Naive Bayes

Ventajas:

- Un manera fácil y rápida de predecir clases, para problemas de clasificación binarios y multiclase.
- Se comporta mejor que otros modelos de clasificación, incluso con menos datos de entrenamiento cuando si son independientes las características.
- Es bueno para salvar problemas de dimensionalidad y rendimiento

Desventajas:

- Aunque son unos clasificadores bastante buenos, los algoritmos Naive Bayes son conocidos por ser pobres estimadores.
- La presunción de independencia Naive muy probablemente no reflejará cómo son los datos en el mundo real.
- Cuando el conjunto de datos de prueba tiene una característica que no ha sido observada en el conjunto de entrenamiento, el modelo le asignará una probabilidad de cero y no podrá realizar predicciones.

Naive Bayes: Tipos de algoritmos

Bernoulli Naive Bayes: Para casos donde los atributos son variables binarias (e.g. si una palabra ocurre o no en un documento).

Multinomial Naive Bayes: Para casos donde los atributos representan frecuencias (e.g. la cantidad de veces que una palabra ocurre en un documento).

Gaussian Naive Bayes: Para casos donde los atributos toman valores continuos, se asume que los valores son muestras de una distribución gaussiana (esto se usa para calcular las probabilidades condicionales en el algoritmo).

Naive Bayes: Relación con Regresión Logística

Naive Bayes es un modelo **generativo**, es decir, que intenta modelar la probabilidad $p(y,\boldsymbol{x})$ donde "y" representa la etiqueta y "$\boldsymbol{x}$" representa los atributos. A grandes rasgos, trata de generar los atributos del modelo dada la etiqueta.

La Regresión Logística es un modelo **discriminativo**, es decir, que intenta modelar la probabilidad $p(y|\boldsymbol{x})$. A grandes rasgos, determina la etiqueta "y" a partir del vector de atributos "$\boldsymbol{x}$". Esto quiere decir que no le hace falta hacer ninguna asunción sobre $p(\boldsymbol{x})$ ya que no es necesaria para modelar.

Demo Time (demo_3_naive_bayes)

Algoritmo del perceptrón

El algoritmo del "perceptrón"

- Propuesto por Frank Rosenblatt en 1958
- El objetivo es encontrar un hiperplano de separación, esto es, sólo encuentra la solución si los datos son linealmente separables
- Es un algoritmo online (procesa un ejemplo a la vez)
- Es la red neuronal más simple
- Tiene una capa de entrada y un único nodo de salida.

Filminas del curso previo

El algoritmo del "perceptrón"

Entrada
- Conjunto de entrenamiento
- Tasa de aprendizaje

Algoritmo
- Inicializar los pesos
- $\omega^0 \in \mathbb{R}$

Proceso
- Para cada ejemplo (x_i, y_i) predecir $y_i^t = signo(\omega^t x_i + \omega_0)$

Comparo salidas real y obtenida
- Si $y_i^t \neq y_i \therefore \omega^{t+1} \leftarrow \omega^t + r(y_i x_i)$

Sólo se actualiza cuando se comete un error

Si se han actualizado algunos pesos se vuelve a “Proceso”, si no, se finaliza

Link

Demo Time (demo_4_perceptron)

Fin de la primera clase