

Hoja de referencia VIP: Aprendizaje Supervisado

Afshine AMIDI y Shervine AMIDI

6 de octubre de 2018

Traducido por Juan P. Chavat. Revisado por Fernando Gonzalez-Herrera, Alonso Melgar-Lopez y Fernando Diaz.

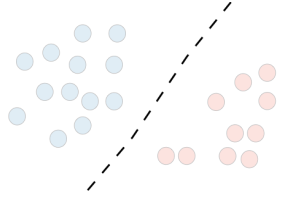
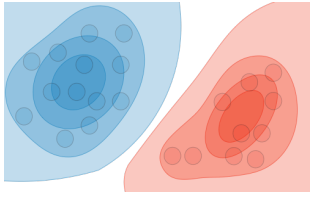
Introducción al aprendizaje supervisado

Dado un conjunto de puntos $\{x^{(1)}, \dots, x^{(m)}\}$ asociado a un conjunto de etiquetas $\{y^{(1)}, \dots, y^{(m)}\}$, queremos construir un clasificador que aprenda cómo predecir y dado x .

□ **Tipo de predicción** – Los diferentes tipos de modelos de predicción se resumen en la siguiente tabla:

	Regresión	Clasificador
Etiqueta	Continuo	Clase
Ejemplos	Regresión lineal	Regresión logística, SVM, Naive Bayes

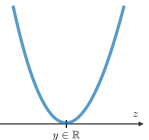
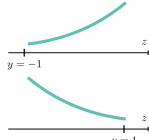
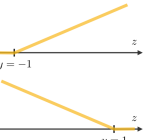
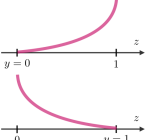
□ **Tipo de modelo** – Los diferentes tipos de modelos se resumen en la siguiente tabla:

	Modelo discriminatorio	Modelo generativo
Objetivo	Estima directamente $P(y x)$	Estima $P(x y)$ para deducir $P(y x)$
Qué se aprende	Límite de decisión	Distribución de los datos
Ilustración		
Ejemplos	Regresiones, SVMs	GDA, Naive Bayes

Notación y conceptos generales

□ **Hipótesis** – La hipótesis se representa con h_θ y es el modelo que elegimos. Para un dato de entrada $x^{(i)}$, la predicción dada por el modelo se representa como $h_\theta(x^{(i)})$.

□ **Función de pérdida** – Una función de pérdida es una función $L : (z, y) \in \mathbb{R} \times Y \mapsto L(z, y) \in \mathbb{R}$ que toma como entrada el valor predicho z y el valor real esperado y , dando como resultado qué tan diferentes son ambos. Las funciones de pérdida más comunes se detallan en la siguiente tabla:

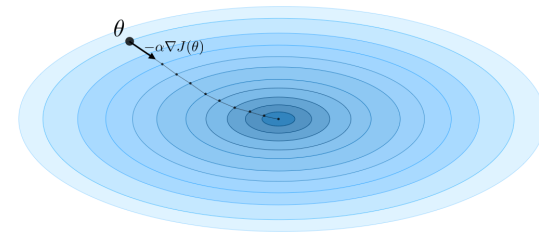
Error cuadrático	Logística	Bisagra	Entropía cruzada
$\frac{1}{2}(y - z)^2$	$\log(1 + \exp(-yz))$	$\max(0, 1 - yz)$	$-\left[y \log(z) + (1 - y) \log(1 - z)\right]$
			
Regresión lineal	Regresión logística	SVM	Red neuronal

□ **Función de costo** – La función de costo J es comúnmente utilizada para evaluar el rendimiento de un modelo y se define utilizando la función de pérdida L de la siguiente forma:

$$J(\theta) = \sum_{i=1}^m L(h_\theta(x^{(i)}), y^{(i)})$$

□ **Descenso de gradiente** – Siendo $\alpha \in \mathbb{R}$ la tasa de aprendizaje, la regla de actualización de descenso en gradiente se expresa junto a la tasa de aprendizaje y la función de costo J de la siguiente manera:

$$\theta \leftarrow \theta - \alpha \nabla J(\theta)$$



Observación: El descenso en gradiente estocástico (en inglés, *Stochastic Gradient Descent*) actualiza el parámetro basándose en cada ejemplo de entrenamiento, mientras que el descenso por lotes realiza la actualización del parámetro basándose en un conjunto (un lote) de ejemplos de entrenamiento.

□ **Verosimilitud** – La verosimilitud de un modelo $L(\theta)$ dados los parámetros θ es utilizada para hallar los valores óptimos de θ a través de la verosimilitud. En la práctica se utiliza la log-verosimilitud $\ell(\theta) = \log(L(\theta))$ la cual es fácil de optimizar. Tenemos:

$$\theta^{\text{opt}} = \arg \max_{\theta} L(\theta)$$

□ **Algoritmo de Newton** – El algoritmo de Newton es un método numérico para hallar θ tal que $\ell'(\theta) = 0$. Su regla de actualización es:

$$\theta \leftarrow \theta - \frac{\ell'(\theta)}{\ell''(\theta)}$$

Observación: la generalización multidimensional, también conocida como método de Newton-Raphson, tiene la siguiente regla de actualización:

$$\theta \leftarrow \theta - \left(\nabla_{\theta}^2 \ell(\theta) \right)^{-1} \nabla_{\theta} \ell(\theta)$$

Regresión lineal

Asumimos que $y|x; \theta \sim \mathcal{N}(\mu, \sigma^2)$

□ **Ecuaciones normales** – Sea X la matriz de diseño, el valor de θ que minimiza la función de costo es una solución en forma cerrada tal que:

$$\theta = (X^T X)^{-1} X^T y$$

□ **Algoritmo LMS** – Sea α la tasa de aprendizaje, la regla de actualización del algoritmo LMS (en inglés, *Least Mean Squares*) para el entrenando de m puntos, conocida también como tasa de aprendizaje de Widrow-Hoff, se define como:

$$\forall j, \quad \theta_j \leftarrow \theta_j + \alpha \sum_{i=1}^m [y^{(i)} - h_{\theta}(x^{(i)})] x_j^{(i)}$$

Observación: la regla de actualización es un caso particular del ascenso de gradiente.

□ **LWR** – Regresión local ponderada, LWR (en inglés, *Locally Weighted Regression*) es una variante de la regresión lineal que pondera cada ejemplo de entrenamiento en su función de costo utilizando $w^{(i)}(x)$, la cual se define con el parámetro $\tau \in \mathbb{R}$ as:

$$w^{(i)}(x) = \exp \left(- \frac{(x^{(i)} - x)^2}{2\tau^2} \right)$$

Clasificación y regresión logística

□ **Función sigmoide** – La función sigmoide g , también conocida como la función logística, se define de la siguiente forma:

$$\forall z \in \mathbb{R}, \quad g(z) = \frac{1}{1 + e^{-z}} \in]0, 1[$$

□ **Regresión logística** – Asumiendo que $y|x; \theta \sim \text{Bernoulli}(\phi)$, tenemos la siguiente forma:

$$\phi = p(y = 1|x; \theta) = \frac{1}{1 + \exp(-\theta^T x)} = g(\theta^T x)$$

Observación: no existe solución en forma cerrada para los casos de regresiones logísticas.

□ **Regresión softmax** – La regresión softmax, también llamada regresión logística multiclase, es utilizada para generalizar regresiones logísticas cuando hay más de dos clases resultantes. Por convención, se define $\theta_K = 0$, lo que hace al parámetro de Bernoulli ϕ_i de cada clase i igual a:

$$\phi_i = \frac{\exp(\theta_i^T x)}{\sum_{j=1}^K \exp(\theta_j^T x)}$$

Modelos lineales generalizados

□ **Familia exponencial** – Se dice que una clase de distribuciones está en una familia exponencial si es posible escribirla en términos de un parámetro natural, también llamado parámetro canónico o función de enlace, η , un estadístico suficiente $T(y)$ y una función de log-partición (log-partition function) $a(\eta)$ de la siguiente manera:

$$p(y; \eta) = b(y) \exp(\eta^T T(y) - a(\eta))$$

Observación: comúnmente se tiene $T(y) = y$. Además, $\exp(-a(\eta))$ puede ser visto como un parámetro de normalización que asegura que las probabilidades sumen uno.

La siguiente tabla presenta un resumen de las distribuciones exponenciales más comunes:

Distribución	η	$T(y)$	$a(\eta)$	$b(y)$
Bernoulli	$\log \left(\frac{\phi}{1-\phi} \right)$	y	$\log(1 + \exp(\eta))$	1
Gaussiana	μ	y	$\frac{\eta^2}{2}$	$\frac{1}{\sqrt{2\pi}} \exp \left(-\frac{y^2}{2} \right)$
Poisson	$\log(\lambda)$	y	e^{η}	$\frac{1}{y!}$
Geométrica	$\log(1 - \phi)$	y	$\log \left(\frac{e^{\eta}}{1 - e^{\eta}} \right)$	1

□ **Supuestos de los modelos GLM** – Los modelos lineales generalizados (en inglés, *Generalized Linear Models*) (GLM) tienen como objetivo la predicción de una variable aleatoria y como una función de $x \in \mathbb{R}^{n+1}$ bajo los siguientes tres supuestos:

$$(1) \quad y|x; \theta \sim \text{ExpFamily}(\eta) \quad (2) \quad h_{\theta}(x) = E[y|x; \theta] \quad (3) \quad \eta = \theta^T x$$

Observación: los métodos de mínimos cuadrados ordinarios y regresión logística son casos particulares de los modelos lineales generalizados.

Máquinas de vectores de soportes

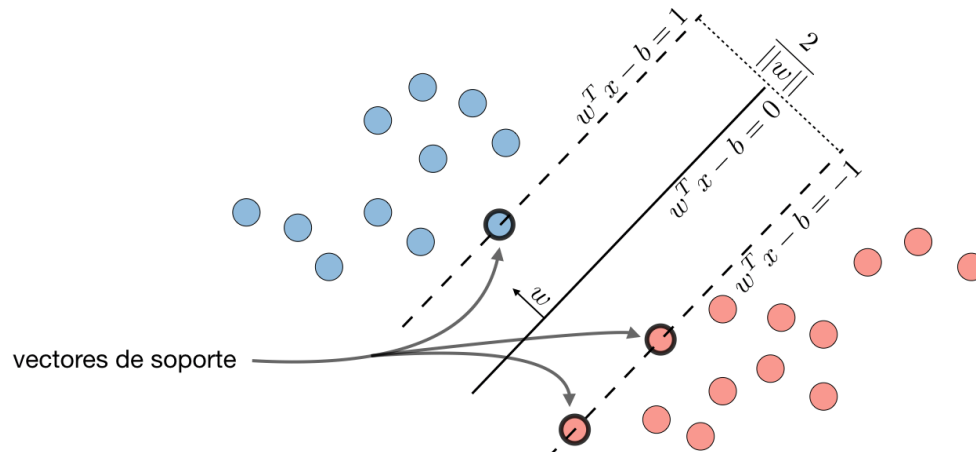
El objetivo de las máquinas de vectores de soportes (en inglés, *Support Vector Machines*) es hallar la línea que maximiza la mínima distancia a la línea.

□ **Clasificador de margen óptimo** – El clasificador de margen óptimo h se define de la siguiente manera:

$$h(x) = \text{sign}(w^T x - b)$$

donde $(w, b) \in \mathbb{R}^n \times \mathbb{R}$ es la solución del siguiente problema de optimización:

$$\min \frac{1}{2} \|w\|^2 \quad \text{tal que} \quad y^{(i)}(w^T x^{(i)} - b) \geq 1$$



Observación: la línea se define como $w^T x - b = 0$.

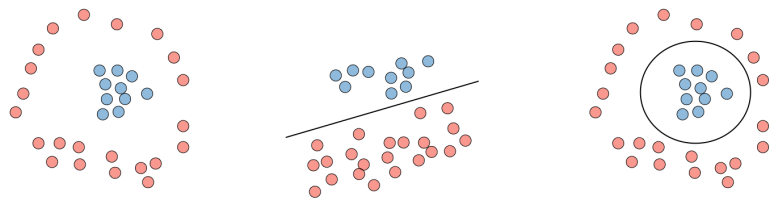
□ **Función de pérdida de tipo bisagra** – La función de pérdida de tipo bisagra (en inglés, *Hinge loss*) es utilizada en la configuración de SVMs y se define de la siguiente manera:

$$L(z, y) = [1 - yz]_+ = \max(0, 1 - yz)$$

□ **Núcleo** – Dado un mapeo de características ϕ , se define el núcleo K (en inglés, *Kernel*) como:

$$K(x, z) = \phi(x)^T \phi(z)$$

En la práctica, el núcleo K definido por $K(x, z) = \exp\left(-\frac{\|x - z\|^2}{2\sigma^2}\right)$ es conocido como núcleo Gaussiano y es comúnmente utilizado.



Separabilidad no lineal \longrightarrow Mapeo de núcleo ϕ \longrightarrow Límite de decisión en el espacio original

Observación: decimos que utilizamos el "truco del núcleo" (en inglés, *kernel trick*) para calcular la función de costo porque en realidad no necesitamos saber explícitamente el mapeo ϕ que generalmente es muy complicado. En cambio, solo se necesitan los valores $K(x, z)$.

□ **Lagrangiano** – Se define el Lagrangiano $\mathcal{L}(w, b)$ de la siguiente manera:

$$\mathcal{L}(w, b) = f(w) + \sum_{i=1}^l \beta_i h_i(w)$$

Observación: los coeficientes β_i son llamados multiplicadores de Lagrange.

Aprendizaje generativo

Un modelo generativo primero trata de aprender como se generan los datos estimando $P(x|y)$, lo que luego podemos utilizar para estimar $P(y|x)$ utilizando el Teorema de Bayes.

Análisis discriminante Gaussiano

□ **Marco** – El Análisis discriminante Gaussiano (en inglés, *GDA - Gaussian Discriminant Analysis*) asume que y , $x|y = 0$ y $x|y = 1$ son de la siguiente forma:

$$y \sim \text{Bernoulli}(\phi)$$

$$x|y = 0 \sim \mathcal{N}(\mu_0, \Sigma) \quad \text{et} \quad x|y = 1 \sim \mathcal{N}(\mu_1, \Sigma)$$

□ **Estimación** – La siguiente tabla resume las estimaciones encontradas al maximizar la probabilidad:

$\hat{\phi}$	$\hat{\mu}_j \quad (j = 0, 1)$	$\hat{\Sigma}$
$\frac{1}{m} \sum_{i=1}^m 1_{\{y^{(i)}=1\}}$	$\frac{\sum_{i=1}^m 1_{\{y^{(i)}=j\}} x^{(i)}}{\sum_{i=1}^m 1_{\{y^{(i)}=j\}}}$	$\frac{1}{m} \sum_{i=1}^m (x^{(i)} - \mu_{y^{(i)}})(x^{(i)} - \mu_{y^{(i)}})^T$

Naive Bayes

□ **Supuestos** – El modelo Naive Bayes supone que las características de cada punto de los datos son todas independientes:

$$P(x|y) = P(x_1, x_2, \dots | y) = P(x_1|y)P(x_2|y)\dots = \prod_{i=1}^n P(x_i|y)$$

□ **Soluciones** – Maximizar la log-probabilidad da las siguientes soluciones, con $k \in \{0, 1\}, l \in [1, L]$

$$P(y = k) = \frac{1}{m} \times \#\{j | y^{(j)} = k\} \quad \text{y} \quad P(x_i = l | y = k) = \frac{\#\{j | y^{(j)} = k \text{ y } x_i^{(j)} = l\}}{\#\{j | y^{(j)} = k\}}$$

Observación: Naive Bayes es comúnmente utilizado para la clasificación de texto y la detección de correo no deseado (*spam*).

Métodos basados en árboles y conjuntos

Estos métodos pueden ser utilizados tanto en problemas de regresión como de clasificación.

□ **CART** – Árboles de clasificación y regresión (en inglés, *Classification and Regression Trees*) (CART), comúnmente conocidos como árboles de decisión, pueden ser representados como árboles binarios. Presentan la ventaja de ser muy interpretables.

□ **Bosques aleatorios** – Es una técnica (en inglés, *Random Forest*) basada en árboles que utiliza una gran cantidad de árboles de decisión construidos a partir de conjuntos de características

seleccionadas al azar. A diferencia del árbol de decisión simple, la solución del método de bosques aleatorios es difícilmente interpretable aunque por su frecuente buen rendimiento es un algoritmo muy popular.

Observación: el método de bosques aleatorios es un tipo de método de conjuntos.

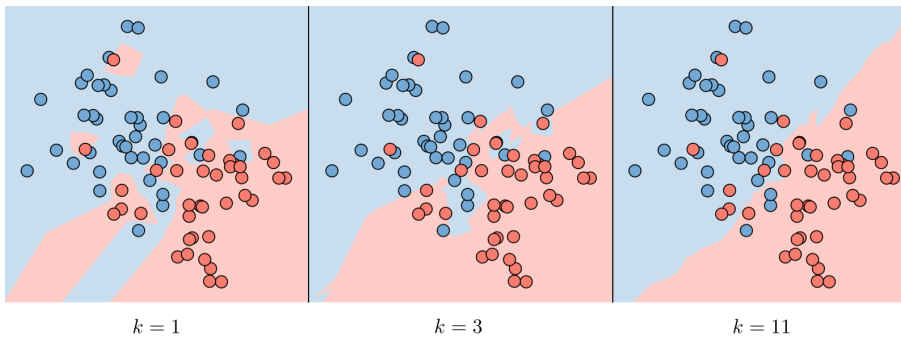
□ **Potenciación** – La idea de la potenciación (en inglés, *Boosting*) es combinar varios métodos de aprendizaje débiles para conformar uno más fuerte. La siguiente tabla resume los principales tipos de potenciación:

Potenciamiento adaptativo	Potenciamiento del gradiente
- Se pondera fuertemente en los errores para mejorar en el siguiente paso del potenciamiento	- Los métodos de aprendizaje débiles entrenan sobre los errores restantes

Otros métodos no paramétricos

□ **k vecinos más cercanos** – El algoritmo de k vecinos más cercanos (en inglés, *k-nearest neighbors algorithm*), comúnmente conocido como k -NN, es un método no paramétrico en el que la respuesta a un punto de los datos está determinada por la naturaleza de sus k vecinos del conjunto de datos. El método puede ser utilizado tanto en clasificaciones como regresiones.

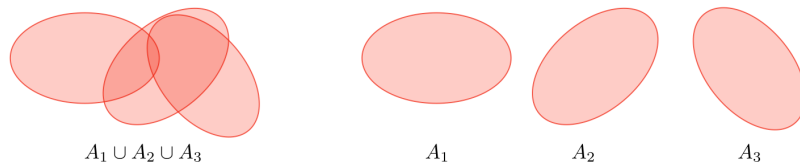
Observación: Cuanto mayor es el parámetro k , mayor es el sesgo, y cuanto menor es el parámetro k , mayor la varianza.



Teoría del aprendizaje

□ **Desigualdad de Boole** – Sean A_1, \dots, A_k k eventos, tenemos que:

$$P(A_1 \cup \dots \cup A_k) \leq P(A_1) + \dots + P(A_k)$$



□ **Desigualdad de Hoeffding** – Sean Z_1, \dots, Z_m m variables iid extraídas de una distribución de Bernoulli de parámetro ϕ . Sea $\hat{\phi}$ su media empírica y $\gamma > 0$ fija. Tenemos que:

$$P(|\phi - \hat{\phi}| > \gamma) \leq 2 \exp(-2\gamma^2 m)$$

Observación: esta desigualdad se conoce también como el límite de Chernoff.

□ **Error de entrenamiento** – Para un clasificador dado h , se define el error de entrenamiento $\hat{\epsilon}(h)$, también conocido como riesgo empírico o error empírico, de la siguiente forma:

$$\hat{\epsilon}(h) = \frac{1}{m} \sum_{i=1}^m 1_{\{h(x^{(i)}) \neq y^{(i)}\}}$$

□ **Aprendizaje correcto probablemente aproximado (PAC)** – PAC es un marco bajo el cual se probaron numerosos resultados en teoría de aprendizaje, y presenta los siguientes supuestos:

- los conjuntos de entrenamiento y de prueba siguen la misma distribución
- los ejemplos de entrenamiento son escogidos de forma independiente

□ **Shattering** – Dado un conjunto $S = \{x^{(1)}, \dots, x^{(d)}\}$, y un conjunto de clasificadores \mathcal{H} , decimos que \mathcal{H} destroza (shatters) S si para cualquier conjunto de etiquetas $\{y^{(1)}, \dots, y^{(d)}\}$, tenemos que:

$$\exists h \in \mathcal{H}, \quad \forall i \in [1, d], \quad h(x^{(i)}) = y^{(i)}$$

□ **Teorema de la frontera superior** – Sea \mathcal{H} una clase de hipótesis finita tal que $|\mathcal{H}| = k$ y sea δ y el tamaño de la muestra m fijo. Entonces, con probabilidad de al menos $1 - \delta$, tenemos:

$$\epsilon(\hat{h}) \leq \left(\min_{h \in \mathcal{H}} \epsilon(h) \right) + 2 \sqrt{\frac{1}{2m} \log \left(\frac{2k}{\delta} \right)}$$

□ **Dimensión VC** – La dimensión de Vapnik-Chervonenkis (VC) de una clase de hipótesis finita \mathcal{H} , denotada como $VC(\mathcal{H})$, es el tamaño del conjunto más grande destrozado (shattered) por \mathcal{H} .

Observación: la dimensión VC de $\mathcal{H} = \{\text{set of linear classifiers in 2 dimensions}\}$ es 3.



□ **Teorema (Vapnik)** – Dado \mathcal{H} , con $VC(\mathcal{H}) = d$ y m el número de ejemplos de entrenamiento. Con probabilidad de al menos $1 - \delta$, tenemos que:

$$\epsilon(\hat{h}) \leq \left(\min_{h \in \mathcal{H}} \epsilon(h) \right) + O \left(\sqrt{\frac{d}{m} \log \left(\frac{m}{d} \right)} + \frac{1}{m} \log \left(\frac{1}{\delta} \right) \right)$$