Boosting en el modelo de aprendizaje PAC

Ricardo Mendoza*

Institución Universitaria Politécnico Grancolombiano. Bogotá - Colombia

Fecha de entrega: 2 de diciembre de 2011
Fecha de aprobación: 21 de marzo de 2013

Resumen Una revisión de la idea de *Boosting* en el modelo de aprendizaje PAC es presentada. Adicionalmente se provee una revisión del primer método de *Boosting* práctico, el *Boosting* adaptativo (*Adaboost*), dando detalles respecto a las garantías teóricas en la convergencia del error y explorando el importante concepto de margen.

Abstract A review on the idea of Boosting in the PAC learning model is presented. Also a review of the first practical Boosting method, the adaptative boosting (Adaboost) is provided, giving details concerning theoretical garantees on error convergence and exploring the important concept of margin.

Palabras Clave: aprendizaje de máquina, modelo PAC, dimensión VC, entrenamiento supervisado, *Boosting*.

Keywords: machine learning, PAC model, VC dimension, supervised learning, Boosting.

* El proyecto de investigación del cual es producto este documento, ha sido financiado por la Fundación Politécnico Grancolombiano Institución Universitaria, mediante el contrato de investigación PG856_100511, aprobado el 17 de diciembre de 2010.
ramendoza@poli.edu.co

1. Introducción

Potencializar o *Boosting* [1,2,3,4,5], es una técnica en *Machine Learning* [6,7,8] para obtener clasificadores fuertes, mediante la combinación de clasificadores débiles en aprendizaje supervisado [9,10]. La idea de *Boosting* [11,5] está estrechamente relacionada con el modelo de aprendizaje Probablemente Aproximadamente Correcto (PAC, por sus siglas en inglés) de Valiant [12,13,14,15] y con el concepto de dimensión Vapnik-Chervonenkis [16,17,18] o dimensión VC.

El primer método práctico de *Boosting*, el *Boosting* Adaptativo o *Adaboost*, desarrollado por Freund y Shapire [19,20,21], generó un debate importante en la comunidad de *Machine Learning*, debido a que sus resultados superaban las predicciones basadas en la dimensión VC [4,22,23,24], y dando lugar al concepto de margen [2,4,25,26], que es de central importancia en la comprensión y comportamiento del error de generalización [11,27] en diversas clases de clasificadores como redes neuronales [28,29], arboles de decisión [30,31] y máquinas de vectores de soporte (SVM por sus siglas en inglés) [32,33].

La organización del artículo es la siguiente: la sección 2 y 3, están dedicadas a introducir la teoría del modelo de aprendizaje PAC y el concepto de dimensión VC. La sección 4, se enfoca en la idea de *Boosting* y sus orígenes en el modelo PAC. La sección 5, explora el método de *Boosting* adaptativo y explora en detalle resultados referentes a las garantías teóricas que este ofrece en la convergencia del error de prueba, terminando con una ilustración del concepto de margen y sus implicaciones en *Adaboost*. Finalmente, la sección conclusiones, cierra este documento, ofreciendo un recuento de los aspectos más relevantes de su contenido.

2. Modelo PAC

El modelo PAC de Valiant [12], fue el primer modelo de aprendizaje computacional. Al centrarse en el problema de clasificación binaria, provee condiciones necesarias para el aprendizaje de clases de conceptos bajo una clase de hipótesis particular, empleando algún mecanismo eficiente (algoritmo), que formula hipótesis sobre algún conjunto de datos etiquetados (instancias) dados.

En el modelo PAC, las clases de conceptos C y de hipótesis H, corresponden a subconjuntos sobre algún espacio de entrada X. En particular, las clases de hipótesis definen subconjuntos que están relacionados bajo algún tipo de parametrización; por ejemplo los puntos en $\mathbb{R}^2$, contenidos en rectángulos cuyos lados están alineados con los ejes de coordenadas x, y.

Un concepto particular $c \subseteq C$, define etiquetas $y \in \{0, 1\}$ para cada instancia $x \in X$ del espacio de entrada. Específicamente, a una instancia perteneciente al concepto $x \in c$, le es asignada por convención una etiqueta positiva $y = 1$; en caso contrario, le es asignada una etiqueta negativa $y = 0$; este mismo esquema, se replica con la evaluación de una instancia bajo una hipótesis.

El problema de aprendizaje en el modelo PAC se concibe de la siguiente forma:

- Existe un espacio de entrada X, que puede ser infinito.

- Se quiere aprender un concepto $c \in C$, que define a todas las instancias positivas en X.
- Se dispone de una clase de hipótesis H tal que $c \in H$.
- Existe un oráculo $EX(c, D)$, que genera instancias desde X empleando alguna distribución arbitraria (pero invariante) D y que en base a su conocimiento del concepto particular (que también es invariante y se asume su existencia), retorna el par (x, y), con $y = c(x)$.
- Se quiere encontrar alguna hipótesis $h \in H$, pueda identificar con precisión si $x \in c$.
- El criterio de error para las hipótesis es:

$$e(h) = P_D[c(x) \neq h(x)] \tag{1}$$

- Existe un algoritmo A, que puede encontrar a h si puede consultar un número suficiente de veces el oráculo.

Adicionalmente, se busca que el concepto se pueda aprender eficientemente, esto es, el número de consultas que requiere A sea pequeño.

Dado que la hipótesis propuesta por el algoritmo depende de la información provista por el oráculo, el error de la hipótesis en sí, es una variable aleatoria. En este sentido, el interés durante el aprendizaje es encontrar una hipótesis que con baja probabilidad δ, tenga error ε grande.

En referencia a lo anterior, el modelo PAC provee las condiciones para la viabilidad de aprender una clase de conceptos C, empleando una clase de hipótesis H. Esto motiva la siguiente definición:

Aprendizaje PAC. Si existe un algoritmo A, con acceso al oráculo $EX(c, D)$ tal que $\forall c \in C$, $\forall D$ y $\forall \varepsilon, \delta > 0$, puede encontrar, en un número de consultas al oráculo acotado por algún polinomio con respecto a ε^{-1} y δ^{-1}, una hipótesis h tal que

$$P_D[e(h) \leq \varepsilon] \leq 1 - \delta \tag{2}$$

Entonces decimos que la clase de conceptos C, se puede eficientemente PAC aprender, por la clase de hipótesis H.

Un resultado importante en el modelo PAC, está relacionado con la garantía de aprendizaje bajo clases de hipótesis finitas. En este caso se tiene:

$$|H|e^{-\varepsilon m} \leq \delta \tag{3}$$

$$e(h) \leq \frac{1}{m}\left(\ln|H| + \ln\frac{1}{\delta}\right) \tag{4}$$

$$m \geq \frac{1}{\varepsilon}\left(\ln|H| + \ln\frac{1}{\delta}\right) \tag{5}$$

Aquí, del resultado 5 se observa como el número de consultas requerido para obtener una hipótesis arbitrariamente buena, está acotado por un polinomio. Sin embargo este resultado no es aplicable sobre clases de hipótesis infinitas. En este caso, se requiere una medida diferente de complejidad para la clase de hipótesis. La dimensión VC es dicha medida.

3. Dimensión VC

En el contexto PAC, la dimensión Vapnik-Chervonenkis [16], explica hasta qué punto, una clase de conceptos representa de forma completa un conjunto de instancias de tamaño dado.

Por representar, nos referimos a la cantidad de conceptos no redundantes que podemos expresar sobre un conjunto de instancias S (Eq 6), empleando la clase de conceptos C.

$$\Pi_C(S) = \{c \cap S : c \in C\} \tag{6}$$

Cuando la cantidad de conceptos que se pueden representar es máxima, para cualquier conjunto de instancias de tamaño fijo $|S| = m$

$$|\Pi_C(S)| \equiv \Pi_C(m) = 2^m \tag{7}$$

Se dice que S es pulverizado por la clase de conceptos C. Con esto la dimensión VC, corresponde al máximo tamaño del conjunto de instancias S que es pulverizado por C

$$VC(C) = \max\{m : m = |S| \wedge \Pi_C(m) = 2^m\} \tag{8}$$

Teniendo en cuenta esta definición, la intuición podría hacer pensar que una clase de conceptos C infinita, tiene también dimensión VC infinita. Sin embargo, es bien conocido que ciertas clases de conceptos, aunque infinitas cuentan con dimensión VC finita. Lo anterior lleva al resultado más importante para el contexto PAC referente a la dimensión VC

$$\begin{array}{l} |C| \Rightarrow \infty \\ VC(C) \leq d < \infty \rightarrow \exists k : \Pi_C(m) \leq O(m^k) \end{array} \tag{9}$$

La expresión 9, indica, que para conceptos con dimensión VC finita, el número de conceptos no redundantes que se pueden describir en conjuntos de instancias de tamaño creciente, es una función acotada por algún un polinomio de grado k.

De esto y recordando la definición de aprendizaje PAC, se puede concluir que la clase de conceptos C, puede ser eficientemente PAC aprendida por la clase infinita de hipótesis H, si y solo si. la dimensión VC de la clase de hipótesis H, es finita.

4. *Boosting*

La idea base de *Boosting* se centra en la equivalencia del aprendizaje PAC débil y PAC fuerte de conceptos, demostrada por Shapire [1]. A continuación se proveen las definiciones de aprendizaje PAC débil y fuerte:

Aprendizaje PAC débil. Si existe un algoritmo A, con acceso al oráculo $EX(c, D)$ tal que $\forall c \in C$, $\forall D$ y $\forall \varepsilon, \delta > 0$, puede encontrar, en un número de consultas al oráculo acotado por algún polinomio, con respecto a ε^{-1} y δ^{-1}, una hipótesis h tal que

$$P_D[e(h) \leq \varepsilon] \leq 1 - \delta \tag{10}$$

Aprendizaje PAC Fuerte. Si existe un algoritmo A, con acceso al oráculo $EX(c, D)$ tal que $\forall c \in C$, $\forall D$ y $\exists \varepsilon, \delta : 0 < \varepsilon, \delta < 1/2$, puede encontrar, en un número de consultas al oráculo acotado por algún polinomio, con respecto a ε^{-1} y δ^{-1}, una hipótesis h tal que

$$P_D[e(h) \leq \varepsilon] \leq 1 - \delta \tag{11}$$

El método de *Boosting*, crea combinaciones [3] de clasificadores débiles (que cumplen la condición de aprendizaje PAC débil), para de esta manera obtener un clasificador que es fuerte (cumple la condición de aprendizaje fuerte).

La idea aquí está en observar que con suficientes llamados al oráculo, se puede garantizar con probabilidad arbitrariamente alta, que un clasificador débil realice una clasificación correcta [6].

De esta manera podemos crear una jerarquía de clasificadores, en el cual el algoritmo provee la mejor hipótesis débil sobre las instancias en donde los demás se equivocan, creando así una hipótesis conjunta que es fuerte (con suficientes consultas al oráculo, puedo obtener precisión ε y confianza δ arbitrarias).

5. *Boosting* adaptativo (*Adaboost*)

El método de *Boosting* adaptativo (*Adaboost*) de Freund y Schapire [19], emplea clasificadores débiles, para crear una hipótesis con error de prueba inferior —llamado clasificador fuerte— mediante combinación creciente de estos, agregando uno nuevo en cada iteración de entrenamiento.

5.1. Algoritmo canónico de *Adaboost*

En *Adaboost* a cada hipótesis débil está asociada una constante (α) que indica el peso, es decir la importancia, de la respuesta al interior del clasificador combinado (Eq. 12). La respuesta final del clasificador fuerte se obtiene mediante empleo de la función signo (Eq. 13).

$$f_T(x) = \sum_{i=1}^{T} \alpha_i h_i(x) \tag{12}$$

$$y = \text{signo}(f_T(x)) \qquad \text{signo}(x) = \begin{cases} 1 : x \geq 0 \\ -1 : x < 0 \end{cases} \tag{13}$$

La obtención de los coeficientes α_i, en *Adaboost* se estructura alrededor del error pesado para un clasificador débil (Eq. 14), donde D_i, es el peso dado a la respuesta del clasificador frente a una instancia x_i y y_i es su etiqueta correspondiente. Por su parte los pesos sobre el conjunto de datos S, son normalizados, definiendo así una distribución D sobre las instancias de entrenamiento (Eq. 15).

$$e_D = \sum_{(x_i, y_i) \in S,\, h(x_i) \neq y_i} D_i \tag{14}$$

$$\sum_{i=1}^{n} D_i = 1, \quad D_i \in D, \quad |S| = n \tag{15}$$

El valor de alfa (Eq. 16), corresponde al valor que minimiza la cota superior exponencial para el error del clasificador fuerte combinado (Eq. 17), para la iteración t, dado un clasificador débil h^* con mínimo (ínfimo) error pesado respecto a D (Eq. 18).

$$\alpha = \frac{1}{2}\ln\left(\frac{1 - e_D}{e_D}\right) \tag{16}$$

$$e(f) = \frac{1}{n}\sum_{i=1}^{n} e^{-y_i f(x_i)} \geq \frac{1}{n}\sum_{i=1}^{n} I_{\{y_i f(x_i) \leq 0\}} \tag{17}$$

$$e(h^*) = \sum_{i=1}^{n} D_i e^{-y_i \alpha h^*(x_i)} \tag{18}$$

En cada iteración se define una distribución D_t de pesos sobre todas las instancias de entrenamiento. Con esto se evalúa el error pesado de hipótesis débiles candidatas, para determinar la de error ínfimo h^*. Los pesos en distribución D_t, son ajustados frente al error de la hipótesis en $t-1$, reduciendo los pesos para los datos correctamente clasificados e incrementándolos para los datos incorrectamente clasificados. Para esto emplea una función exponencial en función del margen de cada dato (Eq. 19), donde Z_t es un factor que normaliza la suma de los pesos a 1.

$$D_{t+1}(i) = \frac{D_t(i) e^{-y_i \alpha_t h_t(x_i)}}{Z_t} \tag{19}$$

El algoritmo canónico para *Adaboost* (Algoritmo 5.1) inicia estableciendo una distribución D_1 uniforme para el error pesado de los datos en la primera hipótesis débil, e itera un número predefinido de veces T, reduciendo con alta probabilidad el error empírico en cada iteración; durante las iteraciones eventualmente puede alcanzar el error de prueba cero.

5.2. Convergencia

Un aspecto importante de *Adaboost* consiste en proveer garantías teóricas respecto a la convergencia del error de prueba. En particular, existe una cota inferior sobre el número de iteraciones que *Adaboost* requiere para retornar una hipótesis consistente con los datos de entrenamiento; esto siempre y cuando la clase de hipótesis utilizada garantice un error pesado inferior a un medio para cualquier D_1.

Para esto, requerimos una cota superior para el error de entrenamiento del clasificador fuerte (Eq 20), para ello iniciaremos con desarrollar en detalle, algunos de los resultados para *Adaboost*.

$$e(h) = \frac{1}{n}\sum_{i=1}^{n} I_{\{y_i f(x_i) \leq 0\}} \tag{20}$$

Algoritmo 5.1 *Adaboost*

```
Adaboost (S, T)
  D1[i]:= 1/|S| %para todo i=1,2,...,|S|
  para todo t= 1,2,...,T repetir
    h[t] := hipotesisPesadaInfima(S,Dt)
    e[t] := errorPesadoHipotesis(S, h[t], Dt)
    alfa[t] := 1/2 ln(1-e[t]/e[t])
    %para todo para todo i=1,2,...,|S|
    Dt+1[i] = Dt[i]*exp(-alfa[t]*Yi*h[t](Xi))/Zt
    %Donde Zt es tal que suma(Dt+1[i])=1 para todo i
  finalizar para todo t
  retornar h[], alfa[]
fin Adaboost
```

Iniciamos con la expresión para los pesos sobre los datos en $t+1$, con base en su definición recursiva (Eq. 19). Dando un paso atrás se obtiene

$$\begin{aligned} D_{t+1}(i) &= \frac{D_{t-1}(i)\exp\left(-\alpha_t y_i h_{t-1}(x_i)\right)}{Z_{t-1}} \frac{\exp\left(-\alpha_t y_i h_t(x_i)\right)}{Z_t} \\ &= \frac{D_{t-1}(i)\exp\left(-\alpha_t y_i h_{t-1}(x_i) - \alpha_t y_i h_t(x_i)\right)}{Z_{t-1}Z_t} \end{aligned} \tag{21}$$

Continuando hasta $t = 1$ y teniendo en cuenta que $D_1(i) = 1/n$ tenemos

$$D_{t+1}(i) = \frac{1}{n} \frac{\exp\left(\sum_{j=1}^{t} - \alpha_j y_i h_j(x_i)\right)}{\prod_{j=1}^{t} Z_{t-1}} \tag{22}$$

Notando la sumatoria de exponentes corresponde parcialmente a la cota exponencial para el error del clasificador fuerte f (Eq. 17), se obtiene una cota para el error de entrenamiento (Eq. 20) en términos de los valores de normalización Z_j para las distribuciones $D_j(i)$

$$1 = \sum_{i=1}^{n} D_{t+1}(i) = \sum_{i=1}^{n} \frac{\exp\left(\sum_{j=1}^{t} - \alpha_j y_i h_j(x_i)\right)}{n\prod_{j=1}^{t} Z_j} \tag{23}$$

$$e(f) \leq \frac{1}{n}\sum_{i=1}^{n} \exp\left(-y_j f_t(x_i)\right) = \prod_{j=1}^{t} Z_j \tag{24}$$

Los valores de normalización pueden ser expresados en términos de la definición de los pesos (Eq. 19) y los coeficientes α_t (Eq. 16):

$$\begin{aligned} D_{t+1}(i) &= \frac{D_t(i)\exp(-\alpha_t y_i h_t(x_i))}{Z_t} \\ &= \frac{D_t(i)}{Z_t}\left(\frac{1-e_t}{e_t}\right)^{-\frac{y_i h_t(x_i)}{2}} \end{aligned} \tag{25}$$

Así, los pesos, para los casos de clasificación positiva y negativa se representan recursivamente en términos de los valores de normalización y los errores pesados

$$D_{t+1}(i) = \begin{cases} \frac{D_t(i)}{Z_t}\sqrt{\left(\frac{1-e_t}{e_t}\right)} & : \quad y_i h_t(x_i) < 0 \\ \frac{D_t(i)}{Z_t}\sqrt{\left(\frac{e_t}{1-e_t}\right)} & : \quad y_i h_t(x_i) \geq 0 \end{cases} \tag{26}$$

Recordando la definición del error pesado (Eq. 14) y el resultado anterior (Eq. 24) expresamos el valor de normalización para la distribución de pesos en términos del error pesado

$$1 = \sum_{i:y_i \neq h_t(x_i)} D_t(i) + \sum_{i:y_i = h_t(x_i)} D_t(i) = e_t + (1-e_t) \tag{27}$$

$$e_{t+1} = \sum_{i:y_i \neq h_t(x_i)} D_{t+1}(i) \tag{28}$$

$$e_{t+1} = \frac{1}{Z_t}\sum_{i:y_i \neq h_t(x_i)} D_t(i)\sqrt{\left(\frac{1-e_t}{e_t}\right)} \tag{29}$$

$$Z_t = \sum_{i:y_i \neq h_t(x_i)} D_{t+1}(i) + \sum_{i:y_i = h_t(x_i)} D_{t+1}(i) \tag{30}$$

$$Z_t = e_t\sqrt{\left(\frac{1-e_t}{e_t}\right)} + (1-e_t)\sqrt{\left(\frac{e_t}{1-e_t}\right)} \tag{31}$$

$$Z_t = 2\sqrt{e_t\,(1-e_t)} \tag{32}$$

Empleando la anterior (Eq. 32), al resultado para la cota del error del clasificador fuerte (Eq 24), obtenemos

$$e(f) \leq \prod_{j=1}^{t} 2\sqrt{e_j\,(1-e_j)} \tag{33}$$

Esta desigualdad implica una reducción exponencial del error respecto al número de iteraciones aplicada de *Boosting*, siempre y cuando exista una cota superior sobre el error pesado tal que

$$\forall t : e_t < \frac{1}{2} \tag{34}$$

Ahora, supongamos que dicha garantía existe (e) —para alguna clase de clasificadores débiles— dada la existencia de una constante γ tal que

$$\forall j : e_j = e \leq \frac{1}{2} - \gamma, \gamma > 0 \tag{35}$$

Que lleva a coeficientes α constantes para todas las iteraciones de *Boosting*

$$\alpha = \frac{1}{2}\ln\left(\frac{1+2\gamma}{1-2\gamma}\right) \tag{36}$$

Empleando la garantía sobre el error pesado (Eq. 35) sobre el resultado para la cota de error del clasificador fuerte (Eq. 32), manipulando y aplicando la cota exponencial $(1-x) < e^{-x}$, obtenemos

$$e(f_T) \leq \prod_{j=1}^{T} 2\sqrt{(1/2-\gamma)\,(1/2+\gamma)} \tag{37}$$

$$e(f_T) \leq \prod_{j=1}^{T} \sqrt{1-4\gamma^2} \tag{38}$$

$$e(f_T) \leq (1-4\gamma^2)^{T/2} \leq e^{-2\gamma^2 T} \tag{39}$$

Es de notar que esta cota para el error del clasificador es independiente del número de instancias de entrenamiento, el impacto de la clase de hipótesis débil empleada y el número de instancias está contenido implícitamente en γ. Partiendo del hecho que el clasificador fuerte generado por *Adaboost* es consistente si su error es inferior al ocasionado por un único error de clasificación en $n = |S|$ datos. Empleando la cota para el clasificador fuerte (Eq. 39)

$$e(f) < \frac{1}{n} \leq \frac{1}{n}\sum_{i=1}^{n} \exp(-y_j f_t(x_i)) \leq e^{-2\gamma^2 T} \tag{40}$$

Tenemos que el número (estrictamente superior) de iteraciones requeridas para garantizar una hipótesis fuerte consistente sobre n datos es:

$$\frac{1}{n} < e^{-2\gamma^2 T} \quad \Rightarrow \quad \ln n < 2\gamma^2 T \quad \Rightarrow \quad T > \frac{\ln n}{2\gamma^2} \tag{41}$$

5.3. Márgenes y aprendizaje

Aunque *Adaboost* alcanza el error empírico de cero, el error sobre el conjunto de prueba pude seguir disminuyendo (Fig. 1). Lo anterior ocurre aunque la dimensión VC correspondiente al clasificador combinado siga incrementando.

Este fenómeno, denominado corrimiento del margen [2] (Fig. 2), se da por el incremento positivo de los márgenes $y_i f(x_i)$ en el clasificador fuerte. *Adaboost* tiene este comportamiento dado que los valores α_i son seleccionados buscando

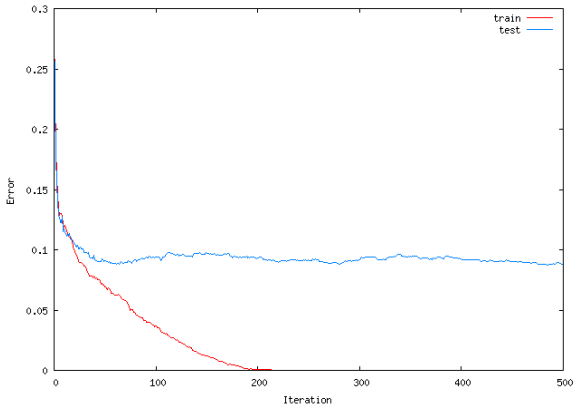

Figura 1. Error empírico y de prueba sobre 500 iteraciones de *Adaboost*. Es de notar que luego de que el error de entrenamiento alcanza un valor de 0, la hipótesis combinada tiende a mejorar sobre el error de prueba. Notar adicionalmente, que en el mínimo observable en la iteración 70, la dimensión VC de la hipótesis combinada es cercana a la dimensión VC de la clase con la cual se originaron los datos de entrenamiento y prueba.

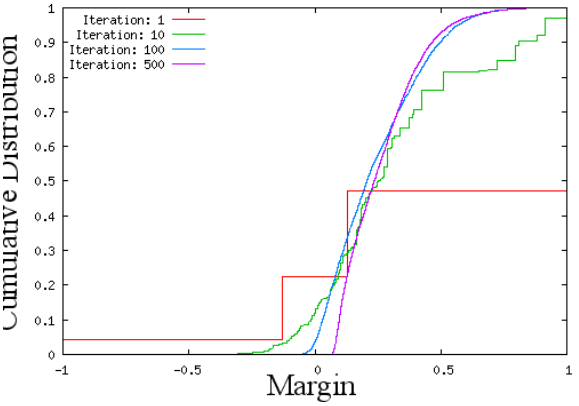

Figura 2. Distribución acumulada de márgenes para las iteraciones 1, 10, 100 y 500 de *Adaboost*. Se puede observar cómo la distribución de márgenes es empujada a la izquierda, así como la convergencia de la distribución.

minimizar la cota de error exponencial, incrementando a su vez el margen (Eq. 17).

Por último, es de resaltar que el concepto de margen es central en otros métodos de aprendizaje como en redes neuronales, árboles de decisión y en las

máquinas de vectores de soporte SVM [32], siendo de importancia central en su teoría.

6. Conclusiones

Se presentó la idea de *Boosting* bajo el marco del modelo de aprendizaje PAC, derivado de la equivalencia entre el aprendizaje PAC débil y fuerte, y su relación con la dimensión VC, como medida para la complejidad de clases de hipótesis infinitas, mostrando cómo en clases de hipótesis con dimensión VC finita, el número de instancias necesarias para obtener una hipótesis con precisión y confianza arbitraria está acotada por una función polinómica.

Así mismo, se exploró el método de *Boosting* Adaptativos *Adaboost*, presentando en forma detallada, un importante resultado sobre la convergencia del error de prueba en *Adaboost*, e ilustrando cómo la maximización local del margen, permite a *Adaboost* obtener resultados superiores a los esperados, bajo el análisis de la complejidad VC del clasificador fuerte combinado y el fenómeno del corrimiento de márgenes.

Referencias

1. Schapire, R.E.: The strength of weak learnability. Machine learning. 5, 197–227 (1990).
2. Schapire, R.E., Freund, Y., Bartlett, P., Lee, W.S.: Boosting the margin: A new explanation for the effectiveness of voting methods. The annals of statistics. 26, 1651–1686 (1998).
3. Maclin, R., Opitz, D.: Popular ensemble methods: An empirical study, (2011). Recuperado de: `http://arxiv.org/pdf/1106.0257`.
4. Rudin, C., Daubechies, I., Schapire, R.E.: On the dynamics of boosting. NIPS Proceedings. (2003).
5. Schapire, R.E.: The boosting approach to machine learning: An overview. Lecture Notes in Statistics. Springer Verlag. 149–172 (2003).
6. Freund, Y., Schapire, R., Abe, N.: A short introduction to boosting. Journal-Japanese Society For Artificial Intelligence. 14, 1612 (1999).
7. Sebastiani, F.: Machine learning in automated text categorization. ACM computing surveys (CSUR). 34, 1–47 (2002).
8. Long, P.M., Servedio, R.A., Anderson, R.N., Boulanger, A.: Systems and methods for martingale boosting in machine learning. Google Patents (2011).
9. Godec, M., Grabner, H., Leistner, C., Bischof, H.: Speeding Up Semi-Supervised On-line Boosting for Tracking. (2010).
10. Chen, K., Wang, S.: Semi-supervised learning via regularized boosting working on multiple semi-supervised assumptions. Pattern Analysis and Machine Intelligence, IEEE Transactions on. 33, 129–143 (2011).
11. Schapire, R.E., Freund, Y.: Boosting: Foundations and Algorithms. (2012).
12. Valiant, L.G.: A theory of the learnable. Communications of the ACM. 27, 1134–1142 (1984).
13. Pitt, L., Valiant, L.G.: Computational limitations on learning from examples. Journal of the ACM (JACM). 35, 965–984 (1988).

14. Kearns, M., Mansour, Y., Ng, A.Y., Ron, D.: An experimental and theoretical comparison of model selection methods. Machine Learning. 27, 7–50 (1997).
15. Haussler, D.: Probably Approximately Correct Learning. Proceedings of the Eighth National Conference on Artificial Intelligence (1990).
16. Vapnik, V., Chervonenkis, A.: Uniform convergence of frequencies of occurence of events to their probabilities. Dokl. Akad. Nauk SSSR. págs. 915–918 (1968).
17. Blumer, A., Ehrenfeucht, A., Haussler, D., Warmuth, M.K.: Learnability and the Vapnik-Chervonenkis dimension. Journal of the ACM (JACM). 36, 929–965 (1989).
18. Pestov, V.: PAC learnability versus VC dimension: a footnote to a basic result of statistical learning. Neural Networks (IJCNN), The 2011 International Joint Conference on. págs. 1141–1145 (2011).
19. Freund, Y., Schapire, R.: A desicion-theoretic generalization of on-line learning and an application to boosting. Computational learning theory. págs. 23–37 (1995).
20. Rudin, C., Daubechies, I., Schapire, R.E.: The dynamics of AdaBoost: Cyclic behavior and convergence of margins. The Journal of Machine Learning Research. 5, 1557–1595 (2004).
21. Merler, S., Caprile, B., Furlanello, C.: Parallelizing AdaBoost by weights dynamics. Computational statistics & data analysis. 51, 2487–2498 (2007).
22. Schapire, R.E.: The convergence rate of adaboost. The 23rd Conference on Learning Theory, open problem (2010).
23. Blanchard, G., Lugosi, G., Vayatis, N., others: On the rate of convergence of regularized boosting classifiers. The Journal of Machine Learning Research. 4, 861–894 (2003).
24. Reyzin, L., Schapire, R.E.: How boosting the margin can also boost classifier complexity. Proceedings of the 23rd international conference on Machine learning. págs. 753–760 (2006).
25. Rosset, S., Zhu, J., Hastie, T.: Boosting as a regularized path to a maximum margin classifier. The Journal of Machine Learning Research. 5, 941–973 (2004).
26. Rudin, C., Cortes, C., Mohri, M., Schapire, R.: Margin-based ranking meets boosting in the middle. Learning Theory. 63–78 (2005).
27. Koltchinskii, V., Panchenko, D.: Empirical Margin Distributions and Bounding the Generalization Error of Combined Classifiers. Annals of Statistics. 1–50 (2002).
28. Barak, O., Rigotti, M.: A simple derivation of a bound on the perceptron margin using singular value decomposition. Neural computation. 23, 1935–1943 (2011).
29. Shen, C., Li, H.: Boosting through optimization of margin distributions. Neural Networks, IEEE Transactions on. 21, 659–666 (2010).
30. Agapitos, A., O'Neill, M., Brabazon, A., Theodoridis, T.: Maximum margin decision surfaces for increased generalisation in evolutionary decision tree learning. Genetic Programming. 61–72 (2011).
31. Freund, Y., Mason, L.: The alternating decision tree learning algorithm. MACHINE LEARNING - INTERNATIONAL WORKSHOP THEN CONFERENCE. págs. 124–133 (1999).
32. Burges, C.J.C.: A tutorial on support vector machines for pattern recognition. Data mining and knowledge discovery. 2, 121–167 (1998).
33. Smola, A.J., Schölkopf, B.: Learning with kernels. MIT Press (1998).