Clase 9: Clasificación (parte 2)

MA5204: Aprendizaje de Máquinas

Felipe Tobar

Department of Mathematical Engineering &
Center for Mathematical Modelling
Universidad de Chile

20 de abril de 2021

UNIVERSIDAD
DE CHILE

El perceptrón - Introducción

Las nociones básicas que hemos visto hasta ahora para lidiar con el problema de clasificación tienen dos problemas conceptuales.

1. Falta de una métrica correcta
2. No existe una *función de verosimilitud* apropiada

La incorporación de un función que *conecte* al modelo lineal con la clase, resulta en un *modelo lineal generalizado*, es decir, un modelo lineal concatenado con una función no-lineal que llamaremos *función de enlace*.

Sin embargo, el desafío más importante en esta construcción es que el modelo resultante ya no es lineal, **ni en la entrada ni en los parámetros**, pues una verosimilitud (función de enlace) lineal nunca nos llevará desde un espacio de características (hemos asumido $\mathbb{R}^M$) al espacio de categorías $\{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_k\}$. Consecuentemente, necesitamos una no-linealidad **después** de la parte lineal

Una forma de resolver estas problemáticas es mediante el uso del **Perceptrón** (Rosenblatt, 1958), un modelo de clasificación binario que tuvo mucha importancia en el área de reconocimiento de patrones.

El Perceptrón

El Perceptrón es una función no lineal (fija) que recibe un vector de características[1] de x, $\phi(x) \in \mathbb{R}^D$, y le asigna un valor $\{+1, -1\}$ de la siguiente forma:

$$y(x) = f(\theta^\top \phi(x))$$

$$f(u) = \begin{cases} +1, & u \geq 0 \\ -1, & u < 0 \end{cases}$$

El Perceptrón entonces asigna x a la clase $\mathcal{C}_1$ si $y(x) = +1$ y asignará x a la clase $\mathcal{C}_2$ cuando $y(x) = -1$. Notemos que para el caso que ϕ es lineal, este es el mismo clasificador presentado en la Sección de clasificación lineal, pero en este caso el criterio para asignar la clase es **parte del modelo**.

Representando las etiquetas mediante la codificación $t \in \{+1, -1\}$, la condición de asignación puede ser cubierta por la expresión:

$$\theta^\top \phi(x_n) t_n > 0, \quad \forall (x_n, t_n) \in \mathcal{D}.$$

[1] En este caso consideramos no linealidad antes y después de la parte lineal, sin embargo, considerar la entrada como x o como $\phi(x)$ es equivalente en base a lo visto en los modelos lineales en los parámetros.

El Perceptrón

Podemos entonces satisfacer esta restricción mediante el "criterio del perceptrón", el cual se basa en examinar los elementos de $\mathcal{D}$ que fueron clasificados incorrectamente.

Este criterio asocia a los puntos clasificados correctamente error 0 y a los puntos mal clasificados error $-\theta^\top \phi(x)t > 0$. De esta forma, si denotamos $\mathcal{M}$ el conjunto de puntos mal clasificados, se debe minimizar la siguiente función objetivo:

$$J_{\mathrm{P}}(\theta, x) = \mathbb{E}\left(-\theta^\top \phi(x)t(x)\mathbb{1}_{\theta^\top \phi(x)t(x)\leq 0}\right)$$

$$\approx -\sum_{(x_i,t_i)\in\mathcal{D}} \theta^\top \phi(x_i)t_i\mathbb{1}_{\theta^\top \phi(x_i)t_i\leq 0} = -\sum_{(x_i,t_i)\in\mathcal{M}} \theta^\top \phi(x_i)t_i.$$

Para el problema de minimización del funcional del perceptrón, se puede utilizar el método del gradiente estocástico.

Método del gradiente estocástico

En aprendizaje de máquinas por lo general se busca un parámetro óptimo que minimice el error de ajuste de acuerdo a una función de pérdida J. Dicho problema puede ser escrito de la forma:

$$\hat{\theta} = \arg\min_{\theta} \sum_{i=1}^{n} J(y_i, \hat{y}_\theta(x_i)) = \arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} J(y_i, \hat{y}_\theta(x_i)).$$

Donde y_i corresponde a la salida de x_i mientras que $\hat{y}_\theta(x_i)$ representa la predicción de la salida de x_i mediante un modelo de parámetro(s) θ.

En el caso general, el óptimo no puede ser enconrado de forma analítica o bien, el algoritmo del gradiente (clásico) se queda atrapado en mínimos locales. Una forma distinta de ver el problema es considerar que $(x_i, y_i) \sim \mu$ iid para una distribución μ desconocida. Desde ese punto de vista, el problema se reduce a minimizar $\mathbb{E}\left(J(y, \hat{y}_\theta(x))\right)$. Este tipo de problemas puede ser escrito en general como

$$\min_{\theta} \mathbb{E}(f(\theta, X)), \quad X \sim \mu \text{ desconocida.}$$

Método del gradiente estocástico

Una alternativa al método del gradiente clásico $\theta^{\tau+1} = \theta^{\tau} - \beta_{\tau+1}\nabla_\theta \mathbb{E}(f(\theta^{\tau}, X))$ consiste en utilizar las observaciones iid $(x_i)_{i\geq 1} \sim \mu$ al momento de iterar, considerando una observación por iteración en vez del funcional $\mathbb{E}(f(\theta^{\tau}, X))$ completo, es decir:

$$\theta^{\tau+1} = \theta^{\tau} - \eta_{\tau+1}\nabla_\theta f(\theta^{\tau}, x_{\tau+1}).$$

Notar que en cada iteración se necesita evaluar una sola vez $\nabla_\theta f$ y no hace falta calcular su esperanza, más aún, este algoritmo permite entrenar modelos con datos a medida que van llegando (actualización a tiempo real)

Este algoritmo es conocido como método del **gradiente descendente estocástico (SGD)**, donde el término $\nabla_\theta f(\theta^{\tau}, x_{\tau+1})$ puede ser visto como un gradiente exacto perturbado o una realización del gradiente (que es una variable aleatoria):

$$\nabla_\theta f(\theta^{\tau}, x_{\tau+1}) = \nabla_\theta \mathbb{E}(f(\theta^{\tau}, X)) + \Delta_t$$

donde $\Delta_t = \nabla_\theta f(\theta^{\tau}, x_{\tau+1}) - \nabla_\theta \mathbb{E}(f(\theta^{\tau}, X))$ cumple que $\mathbb{E}(\Delta_t) = 0$ ya que de acuerdo a la regla integral de Leibniz, $\nabla_\theta \mathbb{E}(f(\theta^{\tau}, X)) = \mathbb{E}(\nabla_\theta f(\theta^{\tau}, X))$.

Nota: El gradiente estocástico provee robustez a mínimos locales **¿por qué?**

Método del gradiente estocástico

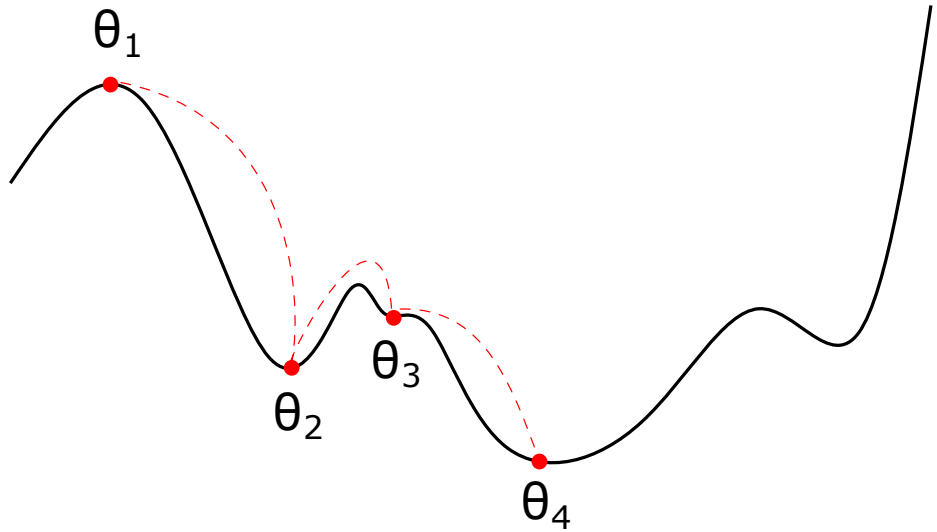

Fig.. Posibles iteraciones del algoritmo SGD. El algoritmo del gradiente clásico hubiese quedado atrapado en θ_2 ya que en dicho punto el gradiente es nulo por lo que no hay desplazamiento.

Theorem (Robbins-Siegmund)

Bajo hipótesis razonables sobre f (regularidad en θ, integrabilidad de $\nabla_\theta f$ y cotas) y tasas de aprendizaje suficientemente pequeñas (por ejemplo, $\eta_\tau = 1/\tau$), la sucesión $(\theta^\tau)_{\tau \geq 1}$ converge c.s. al conjunto de puntos críticos de $\mathbb{E}(f(\theta, X))$.

El perceptrón

En este caso, el algoritmo iterativo tiene la siguiente estructura:

$$\begin{aligned}\theta^{\tau+1} &= \theta^{\tau} - \eta_{\tau} \nabla_{\theta} J_{\mathrm{P}}(\theta^{\tau}, x_i) \\ &= \theta^{\tau} + \eta_{\tau} \phi(x_i) t_i.\end{aligned}$$

Es importante notar que al actualizar el vector θ, el conjunto de puntos mal clasificados $\mathcal{M}$ va a cambiar, pues (esperamos que) en cada iteración los elementos del conjunto de puntos mal clasificados vaya disminuyendo.
Por lo tanto, el algoritmo de entrenamiento para el perceptrón es el siguiente:

i) para cada punto en el conjunto de entrenamiento $\{x_i\}_{i=1}^{N}$,

ii) si el punto x_i fue clasificado correctamente el vector de pesos de mantiene igual

iii) si x_i fue clasificado incorrectamente, el vector θ^{τ} es actualizado según la ecuación anterior con $\eta = 1$ mediante

$$\theta^{\tau+1} = \theta^{\tau} + \phi(x_i) t_i.$$

Es decir, el parámetro θ está paso a paso modificado en la dirección de las características $\phi(x_i)$ con multiplicador ± 1 en base a la clase verdadera de x_i hasta que todos los puntos de $\mathcal{D}$ están bien clasificados.

Modelo generativo

Los modelos que hemos revisado hasta este punto son del tipo *discriminativo*, es decir, modelan directamente la función $f : x \mapsto c$. Con una interpretación probabilística, esto es equivalente a modelar la probabilidad condicional $\mathbb{P}(\mathcal{C}_k|x)$, es decir, dado que conozco el input (o características de) x, cuál es la distribución de probabilidad sobre las clases. Sin embargo, hemos considerado métodos determinísticos, que solo asignan probabilidad 1 a una sola clase.

Un paradigma alternativo es considerar es un enfoque *generativo*, en el cual modelamos dos objetos: en primer lugar la "probabilidad condicional de clase" la cual representa cómo distribuyen los valores de los inputs x cuando la clase es, por ejemplo, $\mathcal{C}_k$, denotada por $\mathbb{P}(x|\mathcal{C}_k)$. En segundo lugar las "probabilidades de clase", o el prior sobre clases, denotada $\mathbb{P}(\mathcal{C}_k)$. Luego, podemos calcular la densidad posterior sobre las clases dado un input x usando el Teorema de Bayes de acuerdo a

$$\mathbb{P}(\mathcal{C}_k|x) = \frac{\mathbb{P}(x|\mathcal{C}_k)\mathbb{P}(\mathcal{C}_k)}{\mathbb{P}(x)}.$$

Modelo generativo

Para el caso de 2 clases, se tiene el siguiente desarrollo:

$$\begin{aligned}
\mathbb{P}(\mathcal{C}_1|x) &= \frac{\mathbb{P}(x|\mathcal{C}_1)\mathbb{P}(\mathcal{C}_1)}{\mathbb{P}(x)} \\
&= \frac{\mathbb{P}(x|\mathcal{C}_1)\mathbb{P}(\mathcal{C}_1)}{\mathbb{P}(x|\mathcal{C}_1)\mathbb{P}(\mathcal{C}_1) + \mathbb{P}(x|\mathcal{C}_2)\mathbb{P}(\mathcal{C}_2)} \\
&= \frac{1}{1 + \frac{\mathbb{P}(x|\mathcal{C}_2)\mathbb{P}(\mathcal{C}_2)}{\mathbb{P}(x|\mathcal{C}_1)\mathbb{P}(\mathcal{C}_1)}} \\
&= \frac{1}{1+\exp(-r)} = \sigma(r).
\end{aligned}$$

Donde hemos introducido la notación $r = r(x) = \ln\left(\frac{\mathbb{P}(x|\mathcal{C}_1)\mathbb{P}(\mathcal{C}_1)}{\mathbb{P}(x|\mathcal{C}_2)\mathbb{P}(\mathcal{C}_2)}\right)$ y la función logística definida mediante $\sigma(r) = \frac{1}{1+e^{-r}}$, la cual tiene propiedades que serán útiles en el entrenamiento, en particular:

$$\text{reflejo: } \sigma(-r) = 1 - \sigma(r)$$

$$\text{derivada: } \frac{d}{dr}\sigma(r) = \sigma(r)(1-\sigma(r))$$

$$\text{inversa: } r(\sigma) = \ln\left(\frac{\sigma}{1-\sigma}\right).$$

Modelo generativo

Si bien la expresión de la distribución condicional en la ecuación anterior parece una presentación antojadiza para hacer aparecer la función logística (sigmoide), pues $r = r(x)$ puede ser cualquier cosa. Sin embargo, veremos que existe una elección particular de las distribuciones condicionales de clase que lleva a un r que es efectivamente lineal en x. En general, nos referiremos a este clasificador como **regresión logística** en dicho caso, es decir, cuando $r(x) = a^\top x + b$.
Podemos ahora considerar el caso de múltiples clases $\{\mathcal{C}_1, \ldots, \mathcal{C}_K\}$, donde un desarrollo similar al anterior resulta en:

$$\mathbb{P}(\mathcal{C}_i|x) = \frac{\mathbb{P}(x|\mathcal{C}_i)\mathbb{P}(\mathcal{C}_i)}{\sum_j \mathbb{P}(x|\mathcal{C}_j)\mathbb{P}(\mathcal{C}_j)} = \frac{\exp(s_i)}{\sum_j \exp(s_j)},$$

donde hemos denotado $s_i = \log\left(\mathbb{P}(x|\mathcal{C}_i)\mathbb{P}(\mathcal{C}_i)\right)$. La función que aparece al lado derecho de la ecuación se conoce como *exponencial normalizada* o *softmax*, y corresponde a una generalización de la función logística a múltiples clases. Además, esta función tiene la propiedad de ser una aproximación suave de la función máximo y convertir cualquier vector $s = [s_1, \ldots, s_k]$ en una distribución de probabilidad, donde podemos hablar de "la probabilidad de ser clase $\mathcal{C}_k$".

Clase 9: Clasificación (parte 2)

MA5204: Aprendizaje de Máquinas

Felipe Tobar

Department of Mathematical Engineering &
Center for Mathematical Modelling
Universidad de Chile

20 de abril de 2021

UNIVERSIDAD
DE CHILE

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, *65*(6), 386–408.