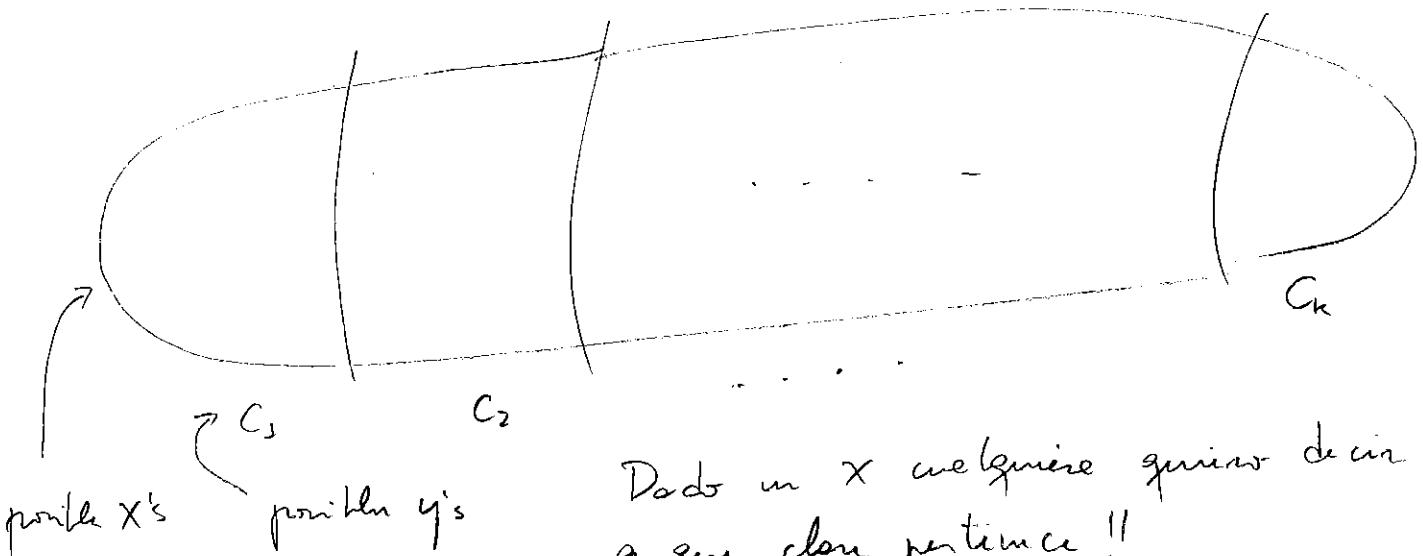
3a) Estimador de máxima verosimilitud, Información y Entropía.

Supongamos un universo de ejemplos, clasificados en k clases



Dado un x cualquiera queremos decir a qué clase pertenece !!

Si no tengo acceso a todos los datos, sólo me queda mirar el experimento como si los datos son generados en un proceso estocástico. Quiero aproximar entre otros a $P_{real}(C_i|x)$

Hemos usado redes neuronales para hacer esto

$$P_{NN}(C_j|x) = \hat{y}_j = \text{forward}_{NN}(x)_j$$

todo esto depende de los parámetros θ de la red

$\Rightarrow$ cuento con $P_\theta(C_j|x)$ computado por mi red para cada posible input.

Supongo que cuento con un conjunto de ejemplos

$$X = \{(x^{(1)}, s^{(1)}), (x^{(2)}, s^{(2)}), \dots, (x^{(N)}, s^{(N)})\} \quad s^{(i)} \in \{C_1, C_2, \dots, C_k\}$$

¿Cuál es la probabilidad que mi red le asigne a obtener exactamente estos datos? (si estamos suponiendo i.i.d) la probabilidad es

↓

$\prod_{i=1}^{N} P_\theta(s^{(i)} | x^{(i)})$ ← probabilidad total de que mi red explique todos los datos observados. Me gustaría que esta probabilidad sea tan grande como sea posible.

Un buen (el mejor) estimador para los parámetros, es el que maximiza la anterior expresión

$$\theta_{ML} = \arg\max_\theta \underbrace{\prod_{i=1}^{N} P_\theta(s^{(i)} | x^{(i)})}_{\text{likelihood}}$$ ← Estimador de máxima verosimilitud para mis parámetros, máximum likelihood

¿Por qué es bueno?

Suponiendo que existe un $\hat{\theta}$ tq $P_{\hat{\theta}} = P_{real}$ entonces

1) si $N \to \infty \Rightarrow \theta_{ML} \to \hat{\theta} \Rightarrow P_{\theta_{ML}} \to P_{real}$

$\lim_{N\to\infty} P_{\theta_{ML}} = P_{real}$ ← consistencia

2) dentro de todos los estimadores consistentes θ_{ML} es el que converge más rápido a $\hat{\theta}$ ← eficiencia

(1) y (2) hacen que θ_{ML} sea el estimador preferido en machine learning.

Log likelihood: las siguientes son formulaciones equivalentes para θ_{ML}

$$\theta_{ML} = \arg\max_\theta \prod_{i=1}^{N} P_\theta(s^{(i)} | x^{(i)}) = \arg\max_\theta \log\left(\prod_{i=1}^{N} P_\theta(s^{(i)} | x^{(i)})\right)$$

$$\theta_{ML} = \arg\max_\theta \sum_{i=1}^{N} \log\left(P_\theta(s^{(i)} | x^{(i)})\right)$$

$$\theta_{ML} = \arg\max_\theta \frac{1}{N} \sum_{i=1}^{N} \log\left(P_\theta(s^{(i)} | x^{(i)})\right)$$

$$\theta_{ML} = \arg\max_\theta \mathbb{E}_{(s,x)\in X}\left(\log\left(P_\theta(s^{(i)} | x^{(i)})\right)\right)$$

se le conoce como pérdida (aunque se está maximizando)

enfatiza después que esta formulación es el soporte teórico del entrenamiento por paquetes

Nos gustaría que la pérdida en nuestra red neuronal de alguna forma capturara esta forma de calcular los mejores parámetros. Para darle una interpretación usamos Teo. Información.

36) Teor. Información

Shannon (interesado en comunicación):

- La información de un evento debe ser mayor mientras menor probable es el evento (evento con probabilidad 1 entrega 0 información)
- La información debe ser siempre positiva
- La información de eventos independientes debiese ser aditiva

$$\Rightarrow \quad Inf(x) = \log\left(\frac{1}{P(x)}\right)$$ ← información de un evento x es el logaritmo del inverso de su probabilidad.

La "entropía" de una distribución de probabilidad (o de un experimento) es el promedio ponderado de la información entregada por todos los eventos.

$$\text{Entropía de } P: \quad H(P) = \sum_x P(x) \cdot Inf(x)$$

$$= \sum_x P(x) \cdot \log\left(\frac{1}{P(x)}\right)$$

usamos $\log_2$ si es en bits

$Inf(x)$ se puede interpretar como la cantidad mínima de bits que necesito para representar a x si quiero ser capaz de diferenciarlo de otros eventos $\Rightarrow H(P) = \mathbb{E}_{x \sim P}(Inf(x))$ cantidad promedio de bits que necesito para codificar los eventos de mi distribución

---

Supongamos que los datos distribuyen según P pero los codifico según una distribución errónea Q ¿Cuántos bits extras voy a predecir?

$$\underbrace{\sum P(x) \log\left(\frac{1}{Q(x)}\right)}_{\text{Bits con la distribución errónea}} - \underbrace{\sum P(x) \log\left(\frac{1}{P(x)}\right)}_{\text{Bits óptimos}} = D_{KL}(P, Q)$$

divergencia

(Kullback-Leibler)

$D_{KL}(P,Q)$ es una buena medida de error y vemos que está íntimamente relacionada a θ_{ML}

$$D_{KL}(P,Q) = \underbrace{\sum P(x) \log\left(\frac{1}{Q(x)}\right)}_{H(P,Q)} - \underbrace{\sum P(x) \log\left(\frac{1}{P(x)}\right)}_{H(P)}$$

↑ Entropía cruzada entre P y Q

← Si minimizamos para Q, $D_{KL}(P,Q)$ tiene un mínimo cuando $Q = P$

si tuviéramos P fija y queremos minimizar $D_{KL}(P, Q_\theta)$
$\Rightarrow$ Entonces con minimizar $H(P,Q_\theta) = \sum P(x) \log\left(\frac{1}{Q_\theta(x)}\right)$

Volviendo ahora a nuestro red con ejemplos $X = \{(x^{(i)}, s^{(i)})\}_{i=1}^{N}$

para el ejemplo $(x^{(i)}, s^{(i)})$ tengo dos distribuciones de probabilidad

$$P_\theta^{(i)}(c_j) = P_\theta(c_j \mid x^{(i)})$$

y la probabilidad empírica

$$P_{emp}^{(i)}(c_j) = P_{emp}(c_j \mid x^{(i)}) = \begin{cases} 1 & \text{si } c_j = s^{(i)} \\ 0 & \text{si } c_j \neq s^{(i)} \end{cases}$$

$$\mathcal{L}' = \frac{1}{N} \sum_{i=1}^{N} D_{KL}\left(P_{emp}^{(i)}, P_\theta^{(i)}\right)$$

queremos minimizar $\mathcal{L}'$ respecto a θ

$\Rightarrow$ Resulta exactamente lo mismo que minimizar ★

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} H\left(P_{emp}^{(i)}, P_\theta^{(i)}\right)$$

$$\theta_{ML} = \arg\min_\theta \mathcal{L} = \arg\min_\theta \frac{1}{N} \sum_{i=1}^{N} H(P_{emp}, P_\theta)$$

cross-entropy loss $\sum_j \underbrace{P_{emp}(c_j \mid x^{(i)})}_{\begin{array}{l}=0 \text{ si } c_j \neq s^{(i)} \\ =1 \text{ si } c_j = s^{(i)}\end{array}} \log\left(\frac{1}{P_\theta(c_j \mid x^{(i)})}\right)$

El estimador de máxima verosimilitud se obtiene de minimizar la cross-entropy loss

$$\Rightarrow \mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \log\left(\frac{1}{P_\theta(s^{(i)} \mid x^{(i)})}\right) = -\frac{1}{N} \sum_{i=1}^{N} \log\left(P_\theta(s^{(i)} \mid x^{(i)})\right)$$ loss