APRENDIZADO DE MÁQUINA

FUNÇÃO *SOFTMAX* / REGULARIZAÇÃO

Zenilton K. G. Patrocínio Jr
zenilton@pucminas.br

Função *Softmax*

A ***função softmax*** é uma função que recebe como entrada um vetor e transforma essa entrada em uma ***distribuição de probabilidade***

Função *Softmax*

A ***função softmax*** é uma função que recebe como entrada um vetor e transforma essa entrada em uma ***distribuição de probabilidade***

Seja $f_j(x)$ uma estimativa da probabilidade que x pertença a classe j

Função *Softmax*

A ***função softmax*** é uma função que recebe como entrada um vetor e transforma essa entrada em uma ***distribuição de probabilidade***

Seja $f_j(x)$ uma estimativa da probabilidade que x pertença a classe j

A ***função softmax*** sobre o vetor de "scores" $(s_1, \ldots, s_k)$ é dada por

$$f_j(x) = \frac{\exp(s_j)}{\exp(s_1) + \exp(s_2) + \cdots + \exp(s_k)} \quad \text{sendo que} \quad \sum_{j=1}^{k} f_j(x) = 1$$

Função *Softmax*

A ***função softmax*** é uma função que recebe como entrada um vetor e transforma essa entrada em uma ***distribuição de probabilidade***

Seja $f_j(x)$ uma estimativa da probabilidade que x pertença a classe j

A ***função softmax*** sobre o vetor de "scores" $(s_1, \ldots, s_k)$ é dada por

$$f_j(x) = \frac{\exp(s_j)}{\exp(s_1) + \exp(s_2) + \cdots + \exp(s_k)} \quad \text{sendo que} \quad \sum_{j=1}^{k} f_j(x) = 1$$

O nome "softmax" indica que se algum "score" s_j for razoavelmente maior que os demais, a saída será próxima de $(0, \ldots, 1, \ldots, 0)$ em que o 1 se encontra na j posição

Classificador *Softmax*

gato	**3,2**
carro	5,1
rã	-1,7

"Scores" ≡ probabilidades logarítmicas não normalizadas das classes

$$s = f(x, W)$$

Classificador *Softmax*

gato	**3,2**
carro	5,1
rã	-1,7

"Scores" ≡ probabilidades logarítmicas não normalizadas das classes

$$P(Y = k|X = x_i) = \frac{e^{s_{y_i}}}{\sum_j e^{s_j}}$$ em que $$s = f(x, W)$$

Classificador *Softmax*

gato	**3,2**
carro	5,1
rã	-1,7

"Scores" ≡ probabilidades logarítmicas não normalizadas das classes

$$P(Y = k|X = x_i) = \frac{e^{s_{y_i}}}{\sum_j e^{s_j}}$$ em que $$s = f(x, W)$$

Deseja-se maximizar a log-verossimilhança, ou ainda, **minimizar a função de log-verossimilhança negativa** da classe correta (considerando uma função de perda)

$$L_i = -\log P(Y = y_i \mid X = x_i)$$

Classificador *Softmax*

gato	**3,2**
carro	5,1
rã	-1,7

"Scores" ≡ probabilidades logarítmicas não normalizadas das classes

$$P(Y = k|X = x_i) = \frac{e^{s_{y_i}}}{\sum_j e^{s_j}} \quad \text{em que} \quad s = f(x, W)$$

Deseja-se maximizar a log-verossimilhança, ou ainda, **minimizar a função de log-verossimilhança negativa** da classe correta (considerando uma função de perda)

$$L_i = -\log P(Y = y_i \mid X = x_i)$$

Portanto

$$L_i = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$

Exemplo – Classificador *Softmax*

$$L_i = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$

gato	**3,2**
carro	5,1
rã	-1,7

probabilidades log
não normalizadas

Exemplo – Classificador *Softmax*

$$L_i = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$

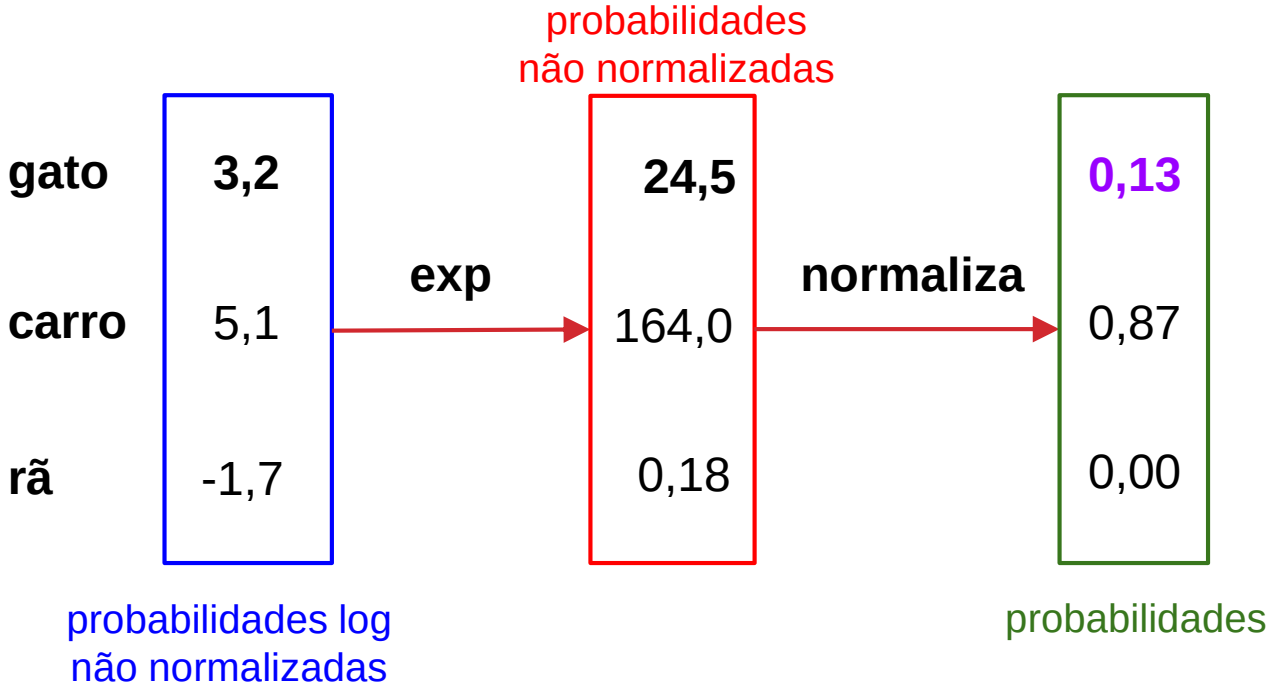
probabilidades não normalizadas

		exp	
gato	**3,2**		**24,5**
carro	5,1	→	164,0
rã	-1,7		0,18

probabilidades log não normalizadas

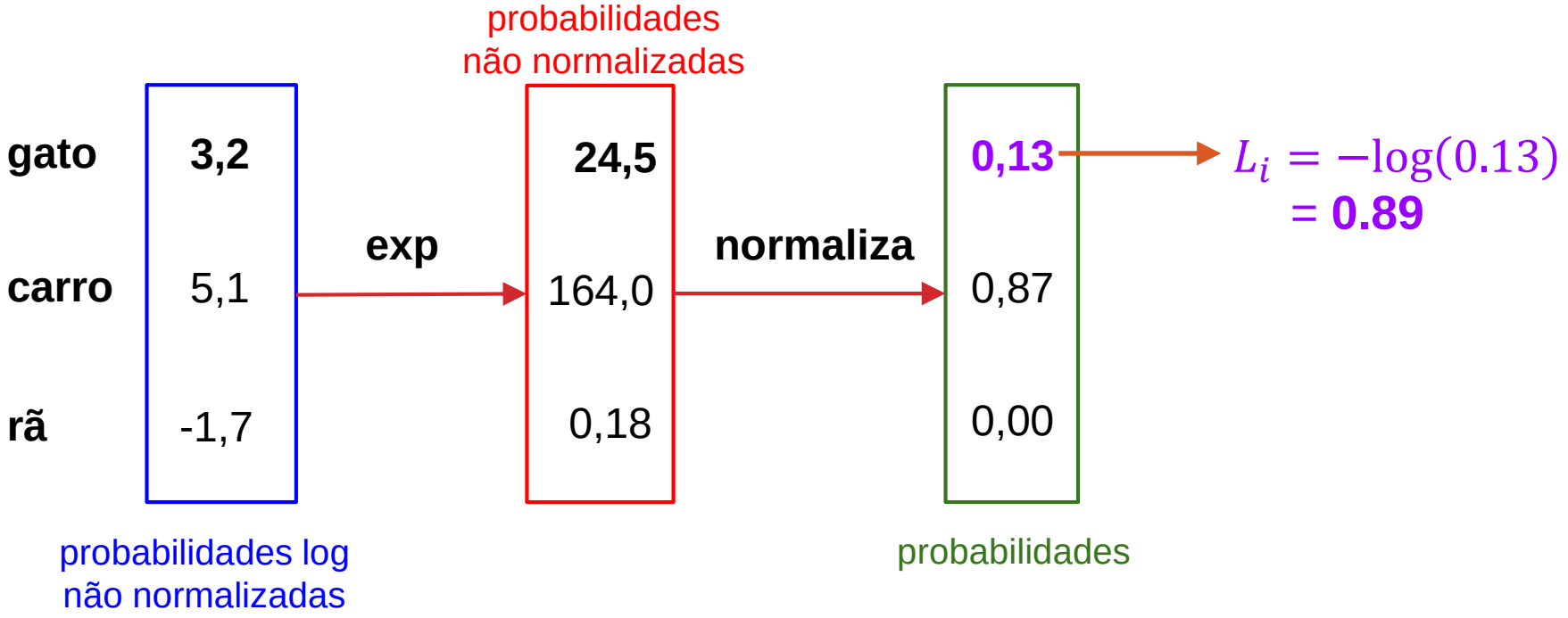
Exemplo – Classificador *Softmax*

$$L_i = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$



Exemplo – Classificador *Softmax*

$$L_i = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$



Exemplo de Código – Função de Perda de Articulação

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Código em Python (usando numpy)

```
def L_i_vectorized(x, y, W):
  scores = W.dot(x)
  margins = np.maximum(0, scores - scores[y] + 1)
  margins[y] = 0
  loss_i = np.sum(margins)
  return loss_i
```

Problema com Cálculo da Perda

$$f(x, W) = Wx$$

$$L = \frac{1}{N}\sum_{i=1}^{N}\sum_{j \neq y_i} \max\left(0, f(x_i, W)_j - f(x_i, W)_{y_i} + 1\right)$$

Problema com Cálculo da Perda

$$f(x, W) = Wx$$

$$L = \frac{1}{N}\sum_{i=1}^{N}\sum_{j \neq y_i} \max\left(0, f(x_i, W)_j - f(x_i, W)_{y_i} + 1\right)$$

Problema com Cálculo da Perda

Suponha: 3 imagens treino e 3 classes

Para algum W, “scores” $s = f(x, W) = Wx$ são



gato	**3,2**	1,3	2,2
carro	5,1	**4,9**	2,5
rã	-1,7	2,0	**-3,1**
Perda:	**2,9**	**0**	**12,9**

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Antes:

= max(0, 1,3 – 4,9 + 1) +
  max(0, 2,0 – 4,9 + 1)
= max(0, -2,6) + max(0, -1,9)
= 0 + 0
= 0

Problema com Cálculo da Perda

Suponha: 3 imagens treino e 3 classes

Para algum W, “scores” $s = f(x, W) = Wx$ são



gato	**3,2**	1,3	2,2
carro	5,1	**4,9**	2,5
rã	-1,7	2,0	**-3,1**
Perda:	**2,9**	**0**	**12,9**

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Antes:

= max(0, 1,3 – 4,9 + 1) +
  max(0, 2,0 – 4,9 + 1)
= max(0, -2,6) + max(0, -1,9)
= 0 + 0
= 0

Com W duas vezes maior:

= max(0, 2,6 – 9,8 + 1) +
  max(0, 4,0 – 9,8 + 1)
= max(0, -6,2) + max(0, -4,8)
= 0 + 0
= 0

Problema com Cálculo da Perda

Suponha: 3 imagens treino e 3 classes

Para algum W, “scores” $s = f(x, W) = Wx$ são



gato	**3,2**	1,3	2,2
carro	5,1	**4,9**	2,5
rã	-1,7	2,0	**-3,1**
Perda:	**2,9**	**0**	**12,9**

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Antes:

= max(0, 1,3 – 4,9 + 1) +
  max(0, 2,0 – 4,9 + 1)
= max(0, -2,6) + max(0, -1,9)
= 0 + 0
= 0

Com W duas vezes maior:

= max(0, 2,6 – 9,8 + 1) +
  max(0, 4,0 – 9,8 + 1)
= max(0, -6,2) + max(0, -4,8)
= 0 + 0
= 0

Problema com Cálculo da Perda

$$f(x, W) = Wx$$

$$L = \frac{1}{N}\sum_{i=1}^{N}\sum_{j \neq y_i} \max\left(0, f(x_i, W)_j - f(x_i, W)_{y_i} + 1\right)$$

Essa perda tem uma dependência linear de $||w^j||_2$ e essa dependência é geralmente negativa, portanto, a minimização do risco tende a fazer crescer $||w^j||_2$

Problema com Cálculo da Perda

$$f(x, W) = Wx$$

$$L = \frac{1}{N}\sum_{i=1}^{N}\sum_{j \neq y_i} \max\left(0, f(x_i, W)_j - f(x_i, W)_{y_i} + 1\right)$$

Essa perda tem uma dependência linear de $||w^j||_2$ e essa dependência é geralmente negativa, portanto, a minimização do risco tende a fazer crescer $||w^j||_2$

Pode-se tentar corrigir isso usando um termo de regularização $\lambda||W||_2$ em que a norma da matrix $||\cdot||_2$ representa soma dos quadrados de seus elementos

Problema com Cálculo da Perda

$$f(x, W) = Wx$$

$$L = \frac{1}{N}\sum_{i=1}^{N}\sum_{j \neq y_i} \max\left(0, f(x_i, W)_j - f(x_i, W)_{y_i} + 1\right)$$

Essa perda tem uma dependência linear de $||w^j||_2$ e essa dependência é geralmente negativa, portanto, a minimização do risco tende a fazer crescer $||w^j||_2$

Pode-se tentar corrigir isso usando um termo de regularização $\lambda||W||_2$ em que a norma da matrix $||\cdot||_2$ representa soma dos quadrados de seus elementos

$$L = \frac{1}{N}\sum_{i=1}^{N}\sum_{j \neq y_i} \max\left(0, f(x_i, W)_j - f(x_i, W)_{y_i} + 1\right) \; + \lambda||W||_2$$

Força ou Peso de Regularização

$$L = \frac{1}{N}\sum_{i=1}^{N}\sum_{j \neq y_i} \max\left(0, f(x_i, W)_j - f(x_i, W)_{y_i} + 1\right) \; + \; \lambda R(W)$$

Força ou Peso de Regularização

$$L = \frac{1}{N}\sum_{i=1}^{N}\sum_{j\neq y_i}\max\left(0, f(x_i, W)_j - f(x_i, W)_{y_i} + 1\right) + \lambda R(W)$$

λ= força de regularização (hiperparâmetro)

Força ou Peso de Regularização

$$L = \frac{1}{N}\sum_{i=1}^{N}\sum_{j\neq y_i}\max\left(0, f(x_i, W)_j - f(x_i, W)_{y_i} + 1\right) + \lambda R(W)$$

λ= força de regularização (hiperparâmetro)

Regularizações comuns:

- L2 (Ridge) $R(W) = \sum_k \sum_l W_{k,l}^2$
- L1 (Lasso) $R(W) = \sum_k \sum_l \left|W_{k,l}\right|$
- L1 + L2 (*Elastic net*) $R(W) = \sum_k \sum_l \beta W_{k,l}^2 + \left|W_{k,l}\right|$
- Técnicas mais recentes: ***Dropout***, ...