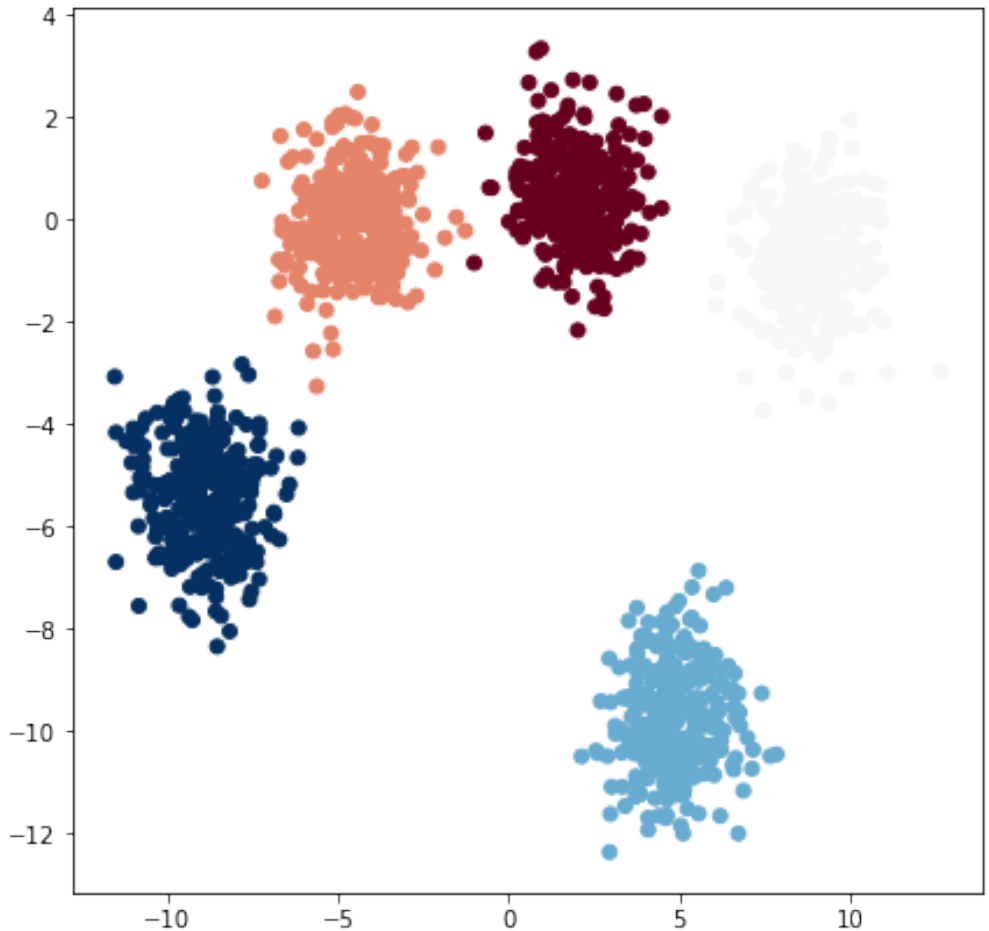
Softmax

Topics in Digital Media

Faisal Qureshi

Softmax

Our goal is to extend ideas first explored in logistic regression, which is a binary classifier, to multi-class problems.



Multinomial distribution

Multinomial distribution can be used to model a random variable X that takes values in $\{1, \cdots, k\}$.

$$\Pr(X = i) = \phi_i$$

Since probabilities all sum to 1, $\sum_{i=1}^{k} \phi_i = 1$. Therefore, $\phi_k = 1 - \sum_{i=1}^{k-1} \phi_i$.

Parameters of a multinomial distribution are $\phi_1, \cdots, \phi_{k-1}$.

Example

- Classification in 3 or more classes
- Which of the k diseases does a patient have?

Multinomial distribution

Using indicators variables introduced previously, we can write the the probability of a multinomial random variable X as follows:

$$\begin{aligned}\Pr(X) &= \phi_1^{\mathbb{I}_1(x)} \phi_2^{\mathbb{I}_2(x)} \cdots \phi_k^{\mathbb{I}_k(x)} \\ &= \prod_{i=1}^{K} \phi_i^{\mathbb{I}_i(x)}\end{aligned}$$

$\phi_k = 1 - \sum_{i=1}^{k-1} \phi_i$

$\mathbb{I}_k(x) = \sum_{i=1}^{k-1} 1 - \mathbb{I}_i(x)$

Indicator variable

$$\mathbb{I}_c(y^{(i)}) = \begin{cases} 1 & \text{if } y^{(i)} = c \\ 0 & \text{otherwise} \end{cases}$$

Multiclass classification

The goal of multiclass classification is to learn $h_\theta(\mathbf{x})$, which can be used to assign a label $y \in \{1, \cdots, K\}$ to the input $\mathbf{x}$. Label y takes values in $\{1, \cdots, K\}$, so we can use multinomial distribution to specify its probability distribution.

Under the assumption that data is i.i.d.

$$\Pr(y|\mathbf{X}, \theta) = \prod_{i=1}^{N} \left(\prod_{j=1}^{K} \left(h_{\theta_j}(\mathbf{x}^{(i)}) \right)^{\mathbb{I}_j(y^{(i)})} \right)$$

Change of notation: θ_i, where $i \in 1, \cdots, K$ now refers to an $(M+1)$-dimensional vector. Previously θ_i referred to the ith element of the $(M+1)$-dimensional vector θ.

Likelihood for multiclass classification

Likelihood for ith example

$$
\begin{aligned}
L(\theta) &= \Pr(y^{(i)} | \mathbf{X}, \theta) \\
&= \prod_{j=1}^{K} \left(h_{\theta_j}(\mathbf{x}^{(i)}) \right)^{\mathbb{I}_j(y^{(i)})}
\end{aligned}
$$

Negative log likelihood for ith example

Define $\mathbf{y}^{(i)}$, a K-dimensional vector as follows:

$$\mathbf{y}_j^{(i)} = \begin{cases} 1 & \text{if } \mathbb{I}_j(y^{(i)}) \\ 0 & \text{otherwise} \end{cases}$$

Here $i \in [1, N]$ and $j \in [1, K]$.

We can now write negative log likelihood as follows

$$l(\theta) = -\sum_{j=1}^{K} \mathbf{y}_j^{(i)} \log h_{\theta_j}(\mathbf{x}^{(i)})$$

Softmax function

Softmax function or *normalized exponential function* "squashes" a K-dimensional vector $\mathbf{z}$ of arbitrary real values to a K-dimensional vector $\mathrm{S}(\mathbf{z})$ of real values in the range $[0, 1]$ that add up to 1.

$$\mathrm{S}(\mathbf{z})_i = \frac{e^{z_i}}{\sum_k e^{z_k}}$$

Softmax function is often used to highlight the largest values and suppress values which are significantly below the maximum value.

Code example (from Wikipedia)

```
>>> import math
>>> z = [1.0, 2.0, 3.0, 4.0, 1.0, 2.0, 3.0]
>>> z_exp = [math.exp(i) for i in z]
>>> print([round(i, 2) for i in z_exp])
[2.72, 7.39, 20.09, 54.6, 2.72, 7.39, 20.09]
>>> sum_z_exp = sum(z_exp)
>>> print(round(sum_z_exp, 2))
114.98
>>> softmax = [round(i / sum_z_exp, 3) for i in z_exp]
>>> print(softmax)
[0.024, 0.064, 0.175, 0.475, 0.024, 0.064, 0.175]
```

Derivative of softmax function

Case 1

$$\begin{aligned}\frac{\partial}{\partial z_i}\mathrm{S}(\mathbf{z})_i &= \frac{e^{z_i}\left(\sum_k e^{z_k}\right) - e^{z_i}e^{z_i}}{\left(\sum_k e^{z_k}\right)^2} \\ &= \left(\frac{e^{z_i}}{\sum_k e^{z_k}}\right)\left(\frac{\sum_k e^{z_k} - e^{z_i}}{\sum_k e^{z_k}}\right) = \left(\frac{e^{z_i}}{\sum_k e^{z_k}}\right)\left(1 - \frac{e^{z_i}}{\sum_k e^{z_k}}\right) \\ &= \mathrm{S}(\mathbf{z})_i(1 - \mathrm{S}(\mathbf{z})_i)\end{aligned}$$

Case 2

$$\begin{aligned}\frac{\partial}{\partial z_j}\mathrm{S}(\mathbf{z})_i &= \frac{-e^{z_i}e^{z_j}}{\left(\sum_k e^{z_k}\right)^2} \\ &= -\left(\frac{e^{z_i}}{\sum_k e^{z_k}}\right)\left(\frac{e^{z_j}}{\sum_k e^{z_k}}\right) = -\left(\frac{e^{z_i}}{\sum_k e^{z_k}}\right)\left(\frac{e^{z_i}}{\sum_k e^{z_k}}\right) \\ &= -\mathrm{S}(\mathbf{z})_i\mathrm{S}(\mathbf{z})_j\end{aligned}$$

Derivative of softmax function

We can use Kronecker's delta function δ_{ij} to represent the derivative of a softmax function in terms of itself as follows

$$\frac{\partial}{\partial z_j}\mathrm{S}(\mathbf{z})_i = \mathrm{S}(\mathbf{z})_i(\delta_{ij} - \mathrm{S}(\mathbf{z})_j)$$

Here

$$\delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

Softmax classifier

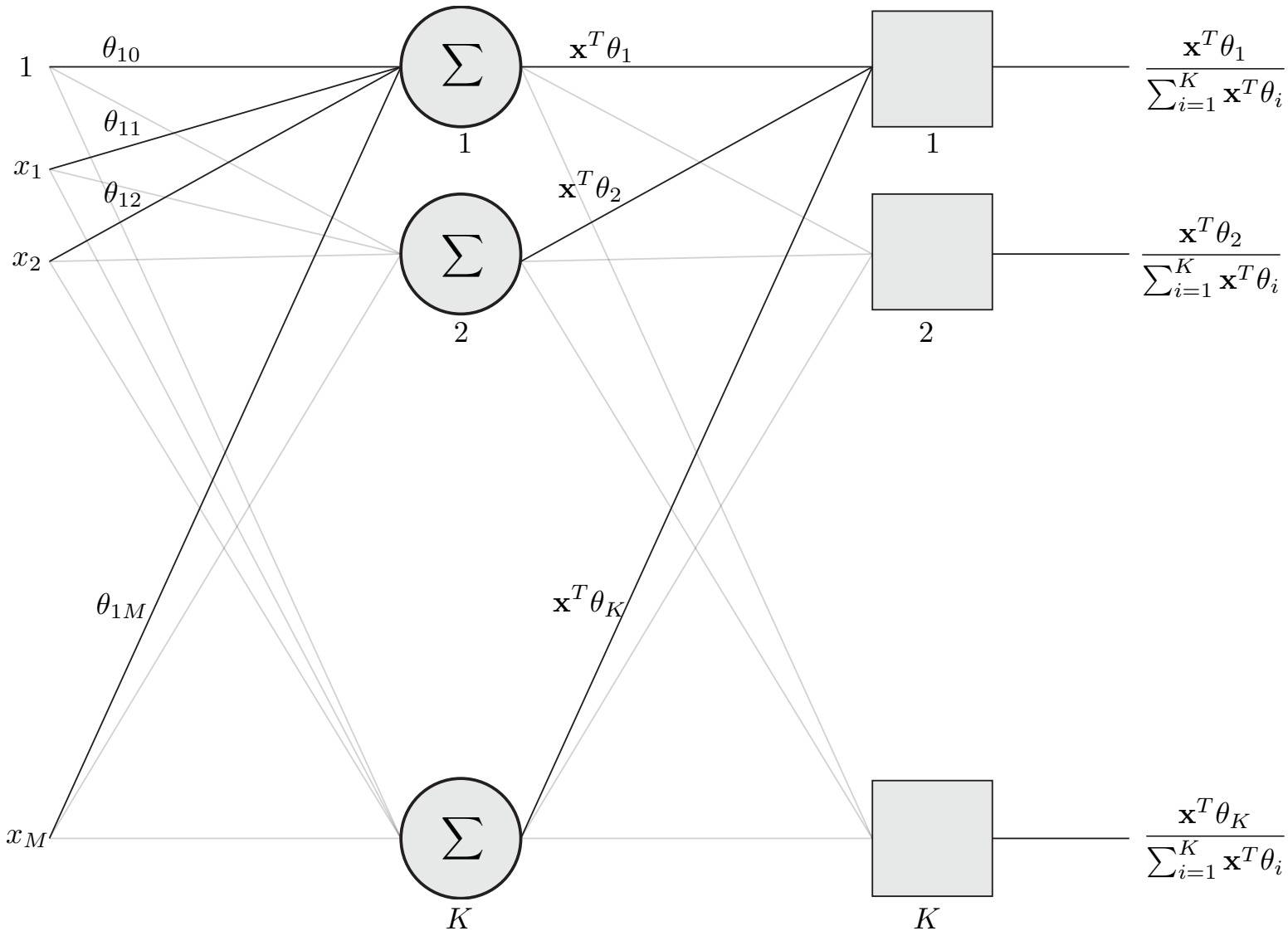
Probability distribution of label y is given by softmax function.

$$h_{\theta_i}(\mathbf{x}) = S(\mathbf{x}^{(i)})_j$$
$$= \frac{e^{\mathbf{x}^T \theta_i}}{\sum_{j=1}^{K} e^{\mathbf{x}^T \theta_j}}$$

Negative log likelihood for softmax classifier

$$l(\theta) = -\sum_{j=1}^{K} \mathbf{y}_j^{(i)} \log \mathrm{S}(\mathbf{x}^{(i)})_j \qquad \text{(For } i\text{th example)}$$

Softmax classifier (K-classes)



Softmax classifier derviation

Notation change: drop superscript (i) and let $\mathrm{S}(\mathbf{x}^{(i)})_j = \pi_j$ for simplicity.

$$\frac{\partial}{\partial \theta_l} = -\sum_{j=1}^{K} \mathbf{y}_j \frac{1}{\pi_j} \frac{\partial}{\partial \theta_l} \pi_j \qquad l \in \{1, \cdots, K\}$$

$$= -\frac{\mathbf{y}_l \left(\pi_l (1 - \pi_l) \mathbf{x}\right)}{\pi_l} - \sum_{j \neq l} \frac{\mathbf{y}_j \left(-\pi_l \pi_j \mathbf{x}\right)}{\pi_j}$$

$$= \left(-\mathbf{y}_l + \mathbf{y}_l \pi_l + \sum_{j \neq l} \mathbf{y}_j \pi_l \right) \mathbf{x}$$

$$= \left(-\mathbf{y}_l + \pi_l \sum_{j=1}^{K} \mathbf{y}_j \right) \mathbf{x}$$

$$= \left(-\mathbf{y}_l + \pi_l \right) \mathbf{x} \qquad \text{Because } \sum_{j=1}^{K} \mathbf{y}_j = 1$$

Softmax classifier gradient descent

Notation: k here refers to the iteration number for gradient descent. η is the learning rate. $l \in \{1, \cdots, K\}$, where K is the number of classes or distinct values labels can take.

Stochatic gradient descent

$$\begin{aligned}\theta_l^{(k+1)} &= \theta_l^{(k)} - \eta \nabla_l l(\theta) \\ &= \theta_l^{(k)} + \eta \left(\frac{e^{\mathbf{x}^T \theta_l}}{\sum_{j=1}^{K} e^{\mathbf{x}^T \theta_j}} - \mathbf{y}_l \right) \mathbf{x}\end{aligned}$$

Cross Entropy

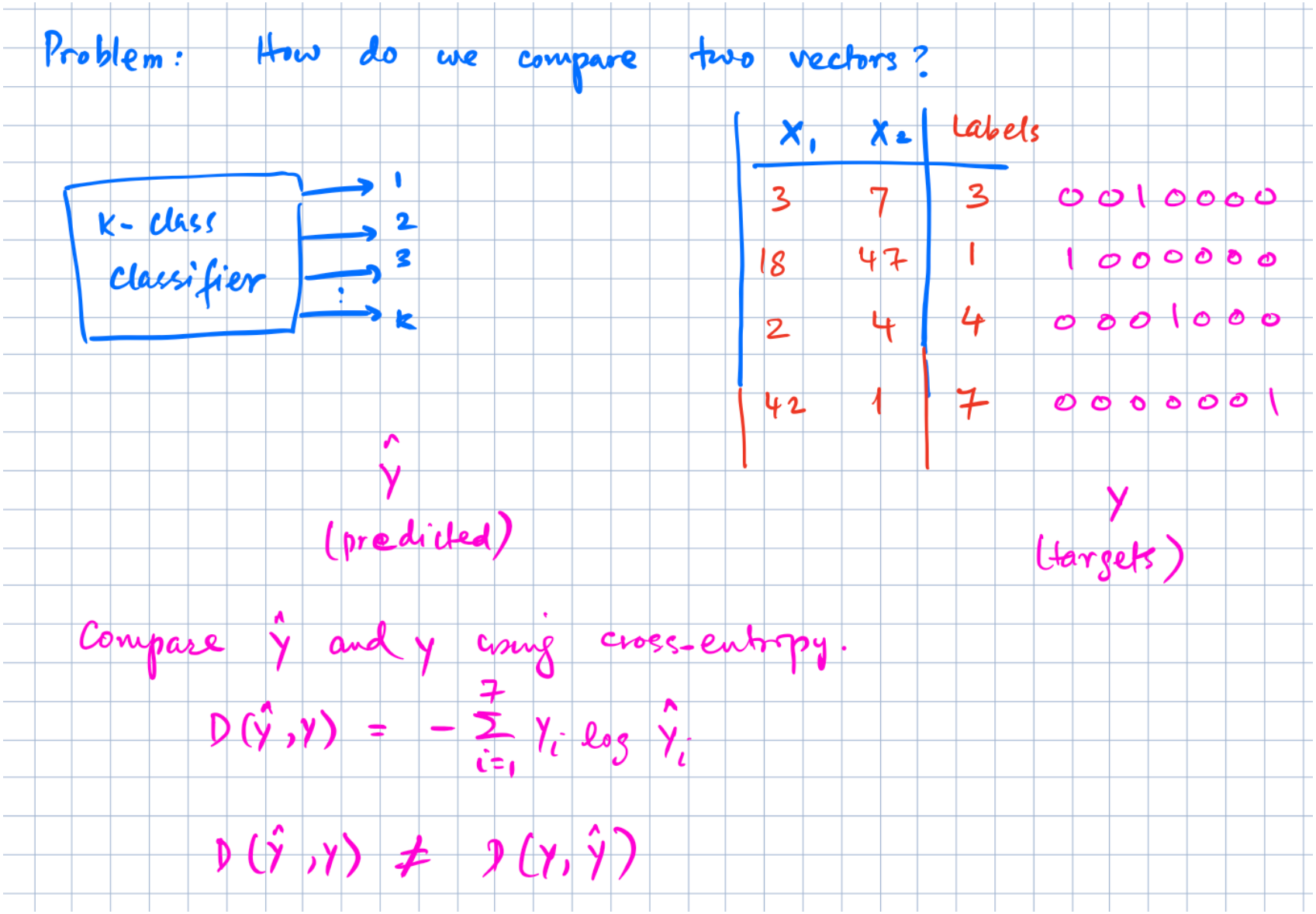


Figure 1:

Summary

- Softmax classifier
- Multinomial distribution