Softmax (Multiclass classifier)

Advanced Topics in High-Performance Computing

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 probability of a multinomial random variable $X$ as follows:

\[
\text{Pr}(X) = \prod_{i=1}^{K} \phi_i \cdot \phi_2(x) \cdots \phi_k(x)
\]

\[
= \prod_{i=1}^{k} \phi_i \cdot \phi_k(x)
\]

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

\[
\Pi_k(x) = \sum_{i=1}^{k-1} 1 - \Pi_i(x)
\]
Indicator variable

\[ I_c(y^{(i)}) = \begin{cases} 
1 & \text{if } y^{(i)} = c \\
0 & \text{otherwise}
\end{cases} \]
Multiclass classification

The goal of multiclass classification is to learn $h_\theta(x)$, which can be used to assign a label $y \in \{1, \cdots, K\}$ to the input $x$. 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.

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

Change of notation: $\theta_i$, where $i \in 1, \cdots, K$ now refers to an $(M + 1)$-dimensional vector. Previously $\theta_i$ referred to the $i$th element of the $(M + 1)$-dimensional vector $\theta$. 
Likelihood for multiclass classification

Likelihood for $i$th example

\[ L(\theta) = \Pr(y^{(i)}|X, \theta) = \prod_{j=1}^{K} \left( h_{\theta_j}(x^{(i)}) \right) I_j(y^{(i)}) \]

Negative log likelihood for $i$th example

\[ l(\theta) = - \sum_{j=1}^{K} I_j(y^{(i)}) \log h_{\theta_j}(x^{(i)}) \]
Negative log likelihood for $i$th example

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

$$y_j^{(i)} = \begin{cases} 1 & \text{if } I_j(y^{(i)}) \\ 0 & \text{otherwise} \end{cases}$$

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

We can now write negative log likelihood as follows

$$l(\theta) = - \sum_{j=1}^{K} y_j^{(i)} \log h_{\theta_j}(x^{(i)})$$
Softmax function

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

\[
S(z)_i = \frac{e^{z_i}}{\sum_k e^{z_k}}
\]

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

**Code example (from Wikipedia)**

```python
>>> 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
\[
\frac{\partial}{\partial z_i} S(z)_i = \frac{e^{z_i} \left( \sum_k e^{z_k} \right) - e^{z_i} e^{z_i}}{(\sum_k e^{z_k})^2} = \left( \frac{e^{z_i}}{\sum_k e^{z_k}} \right) \left( \frac{\sum_k e^{z_k} - e^{z_i}}{\sum_k e^{z_k}} \right) = \left( \frac{e^{z_i}}{\sum_k e^{z_k}} \right) \left( 1 - \frac{e^{z_i}}{\sum_k e^{z_k}} \right) = S(z)_i (1 - S(z)_i)
\]

Case 2
\[
\frac{\partial}{\partial z_j} S(z)_i = -\frac{e^{z_i} e^{z_j}}{(\sum_k e^{z_k})^2} = -\left( \frac{e^{z_i}}{\sum_k e^{z_k}} \right) \left( \frac{e^{z_j}}{\sum_k e^{z_k}} \right) = -\left( \frac{e^{z_i}}{\sum_k e^{z_k}} \right) \left( \frac{e^{z_i}}{\sum_k e^{z_k}} \right) = -S(z)_i S(z)_j
\]
Derivative of softmax function

We can use Kronecker’s delta function $\delta_{ij}$ to represent the derivative of a softmax function in terms of itself as follows

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

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.

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

Negative log likelihood for softmax classifier

$$ l(\theta) = - \sum_{j=1}^K y_j^{(i)} \log S(x^{(i)})_j \quad (\text{For } i\text{th example}) $$
Softmax classifier (K-classes)
Softmax classifier derivation

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

\[
\frac{\partial}{\partial \theta_l} = -\sum_{j=1}^{K} y_j \frac{1}{\pi_j} \frac{\partial}{\partial \theta_l} \pi_j \\
= -\frac{y_l (\pi_l (1 - \pi_l) x)}{\pi_l} - \sum_{j \neq l} y_j (-\pi_l \pi_j x) \\
= \left( -y_l + y_l \pi_l + \sum_{j \neq l} y_j \pi_l \right) x \\
= \left( -y_l + \pi_l \sum_{j=1}^{K} y_j \right) x \\
= (-y_l + \pi_l) x \quad \text{Because} \sum_{j=1}^{K} y_j = 1
\]
Softmax classifier gradient descent

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

**Stochastic gradient descent**

\[
\theta_l^{(k+1)} = \theta_l^{(k)} - \eta \nabla l(\theta)
\]

\[
= \theta_l^{(k)} + \eta \left( \frac{e^{x^T \theta_l}}{\sum_{j=1}^{K} e^{x^T \theta_j}} - y_l \right) x
\]
Cross Entropy

Problem: How do we compare two vectors?

\[
\begin{array}{c}
\text{k-class classifier} \\
\vdots \\
\text{k}
\end{array}
\]

\[
\begin{array}{c|c|c}
X_1 & X_2 & \text{Labels} \\
3 & 7 & 3 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0 \quad 0 \quad 0 \\
18 & 47 & 1 \quad 1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \\
2 & 4 & 4 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0 \quad 0 \\
42 & 1 & 7 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \\
\end{array}
\]

\[
\hat{Y} \quad \text{predicted}
\]

\[
Y \quad \text{targets}
\]

Compare \( \hat{Y} \) and \( Y \) using cross-entropy:

\[
D(\hat{Y}, Y) = -\sum_{i=1}^{k} Y_i \log \hat{Y}_i
\]

\[
D(\hat{Y}, Y) \neq D(Y, \hat{Y})
\]

Figure 1:
Summary

- Softmax classifier
- Multinomial distribution