3.5. Softmax Regression

Lecture based on “Dive into Deep Learning” **http://D2L.AI** (Zhang et al., 2020)

Prof. Dr. Christoph Lippert

Digital Health & Machine Learning

Label Representation

Example

- 4 features
  "height", "weight", "speed", "fur"
- 3 labels "cat", "chicken", "dog".

- $y \in \{1, 2, 3\}$, where the integers represent {dog, cat, chicken} respectively.
  integer encoding
  Not ideal unless there is a natural ordering.
- **One-hot** encoding.
  Vector with one component for every possible category.

$$y \in \{(1,0,0),(0,1,0),(0,0,1)\}$$

Network Architecture

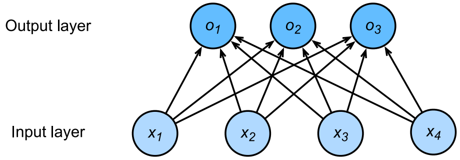
- Model requires one ouput per category.
- With linear models, we will need as many linear functions as we have outputs.

In our example:

$$
\begin{aligned}
o_1 &= x_1 w_{11} + x_2 w_{21} + x_3 w_{31} + x_4 w_{41} + b_1,\\
o_2 &= x_1 w_{12} + x_2 w_{22} + x_3 w_{32} + x_4 w_{42} + b_2,\\
o_3 &= x_1 w_{13} + x_2 w_{23} + x_3 w_{33} + x_4 w_{43} + b_3.
\end{aligned}
$$

More compactly:

$$\mathbf{o} = \mathbf{W}\mathbf{x} + \mathbf{b}$$



Predict probabilities

- Need to guarantee that the probabilities are *non-negative* and *sum up to 1*.
- Solution:

  The nonlinear **softmax** function:

$$\hat{\mathbf{y}} = \mathrm{softmax}(\mathbf{o}) \qquad \text{where} \quad \hat{y}_i = \frac{\exp(o_i)}{\sum_j \exp(o_j)}$$

The most likely class is obtained by by

$$\hat{\imath}(\mathbf{o}) = \operatorname*{argmax}_i o_i = \operatorname*{argmax}_i \hat{y}_i$$

Summarizing it all in vector notation we get

$$\mathbf{o}^{(i)} = \mathbf{W}\mathbf{x}^{(i)} + \mathbf{b}$$

where

$$\hat{\mathbf{y}}^{(i)} = \mathrm{softmax}(\mathbf{o}^{(i)})$$

.

Vectorization

Assume that we are given a mini-batch $\mathbf{X}$ of examples with dimensionality d and batch size n.

Assume that we have q categories (outputs).

Then the mini-batch features $\mathbf{X}$ are in $\mathbb{R}^{n \times d}$, weights $\mathbf{W} \in \mathbb{R}^{d \times q}$ and the bias satisfies $\mathbf{b} \in \mathbb{R}^q$.

$$\mathbf{O} = \mathbf{X}\mathbf{W} + \mathbf{b}$$
$$\hat{\mathbf{Y}} = \mathrm{softmax}(\mathbf{O})$$

Loss Function

$$p(Y|X) = \prod_{i=1}^{n} p(y^{(i)}|x^{(i)})$$

$$-\log p(Y|X) = \sum_{i=1}^{n} -\log p(y^{(i)}|x^{(i)})$$

This yields the **loss** function:

$$l = -\log \hat{p}(y|x) = -\sum_j y_j \log \hat{y}_j,$$

where $\hat{y} = \mathrm{softmax}(\mathbf{o})$.

By construction

- $\mathbf{y}$ consists of all zeroes but for the correct label, such as $(1, 0, 0)$.
- Since all $\hat{y}_j$ are probabilities $\log \hat{y}_j \leq 0$.
- The loss function is minimized if we correctly predict y with *certainty*.

  i.e. if $p(y|x) = 1$

Derivatives

$$
\begin{aligned}
l &= -\sum_j y_j \log \hat{y}_j \\
&= \sum_j y_j \log \sum_k \exp(o_k) - \sum_j y_j o_j \\
&= \log \sum_k \exp(o_k) - \sum_j y_j o_j
\end{aligned}
$$

$$\partial_{o_j} l = \frac{\exp(o_j)}{\sum_k \exp(o_k)} - y_j = \mathrm{softmax}(\mathbf{o})_j - y_j = \Pr(y =$$

Now consider the case where we don't just observe a single outcome but maybe, an entire **distribution over outcomes**.

- We can represent the multinomial distribution as a generic probability vector $(0.1, 0.2, 0.7)$.
- The loss l still works

$$
\begin{aligned}
l(\mathbf{y}, \hat{\mathbf{y}}) &= -\sum_j y_j \log \hat{y}_j \\
&= E_y[-\log \hat{p}(y|x)]
\end{aligned}
$$

This loss is the **cross-entropy loss**.

The **Entropy** of a distribution p is defined as

$$H[p] = \sum_j -p(j) \log p(j)$$

- A key concept in **information theory**
- measures the minimum number of 'nats' needed to encode a distribution.

KL-divergence is a distance measure between distributions

$$\begin{aligned} D(p \| q) &= \sum_j p(j) \log \frac{p(j)}{q(j)} \\ &= -\sum_j p(j) \log q(j) - H[p] \end{aligned}$$

Note:

$$\begin{aligned} l(\mathbf{y}, \hat{\mathbf{y}}) &= -\sum_j y_j \log \hat{y}_j \\ &= D(\mathbf{y} \| \hat{\mathbf{y}}) + \underbrace{H[p]}_{\text{const.}} \end{aligned}$$

Minimizing the **cross-entropy loss** is equivalent to minimizing $D(\mathbf{y} \| \hat{\mathbf{y}})$!

Summary

- We introduced the **softmax** which maps a vector into probabilities.
- Softmax regression applies to classification problems.
    - It uses the *probability distribution* of the output category in the softmax operation.
- Cross entropy is a good measure of the difference between two probability distributions.
    - It measures the number of bits needed to encode the data given our model.