... TO SOFTMAX REGRESSION

There is a straightforward generalization to the multiclass case:

- Instead of a single linear discriminant function we have g linear discriminant functions

$$f_k(\mathbf{x}) = \boldsymbol{\theta}_k^\top \mathbf{x}, \quad k = 1, 2, ..., g,$$

each indicating the confidence in class k.

- The g score functions are transformed into g probability functions by the **softmax** function $s : \mathbb{R}^g \to [0, 1]^g$

$$\pi_k(\mathbf{x}) = s(f(\mathbf{x}))_k = \frac{\exp(\boldsymbol{\theta}_k^\top \mathbf{x})}{\sum_{j=1}^{g} \exp(\boldsymbol{\theta}_j^\top \mathbf{x})},$$

instead of the **logistic** function for $g = 2$. The probabilities are well-defined: $\sum \pi_k(\mathbf{x}) = 1$ and $\pi_k(\mathbf{x}) \in [0, 1]$ for all k.

... TO SOFTMAX REGRESSION / 2

- The softmax function is a generalization of the logistic function. For $g = 2$, the logistic function and the softmax function are equivalent.
- Instead of the **Bernoulli** loss, we use the multiclass **logarithmic loss**

$$L(y, \pi(\mathbf{x})) = -\sum_{k=1}^{g} \mathbb{1}_{\{y=k\}} \log\left(\pi_k(\mathbf{x})\right).$$

- Note that the softmax function is a "smooth" approximation of the arg max operation, so $s((1, 1000, 2)^T) \approx (0, 1, 0)^T$ (picks out 2nd element!).
- Furthermore, it is invariant to constant offsets in the input:

$$s(f(\mathbf{x}) + \mathbf{c}) = \frac{\exp(\boldsymbol{\theta}_k^\top \mathbf{x} + c)}{\sum_{j=1}^{g} \exp(\boldsymbol{\theta}_j^\top \mathbf{x} + c)} = \frac{\exp(\boldsymbol{\theta}_k^\top \mathbf{x}) \cdot \exp(c)}{\sum_{j=1}^{g} \exp(\boldsymbol{\theta}_j^\top \mathbf{x}) \cdot \exp(c)} = s(f(\mathbf{x}))$$

SOFTMAX: LINEAR DISCRIMINANT FUNCTIONS

Softmax regression gives us a **linear classifier**.

- The softmax function $s(\mathbf{z})_k = \frac{\exp(\mathbf{z}_k)}{\sum_{j=1}^{g} \exp(\mathbf{z}_j)}$ is
  - a rank-preserving function, i.e. the ranks among the elements of the vector $\mathbf{z}$ are the same as among the elements of $s(\mathbf{z})$. This is because softmax transforms all scores by taking the $\exp(\cdot)$ (rank-preserving) and divides each element by **the same** normalizing constant.

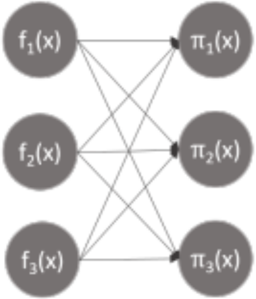
Thus, the softmax function has a unique inverse function $s^{-1} : \mathbb{R}^g \to \mathbb{R}^g$ that is also monotonic and rank-preserving.
Applying s_k^{-1} to $\pi_k(\mathbf{x}) = \frac{\exp(\boldsymbol{\theta}_k^\top \mathbf{x})}{\sum_{j=1}^{g} \exp(\boldsymbol{\theta}_j^\top \mathbf{x})}$ gives us $f_k(\mathbf{x}) = \boldsymbol{\theta}_k^\top \mathbf{x}$.
Thus, softmax regression is a linear classifier.

GENERALIZING SOFTMAX REGRESSION

Instead of simple linear discriminant functions we could use **any** model that outputs g scores

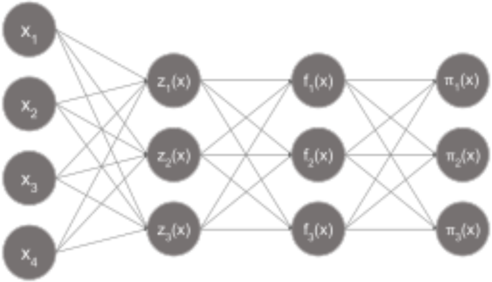
$$f_k(\mathbf{x}) \in \mathbb{R}, k = 1, 2, ..., g$$

We can choose a multiclass loss and optimize the score functions $f_k, k \in \{1, ..., g\}$ by multivariate minimization. The scores can be transformed to probabilities by the **softmax** function.



GENERALIZING SOFTMAX REGRESSION / 2

For example for a **neural network** (note that softmax regression is also a neural network with no hidden layers):



Remark: For more details about neural networks please refer to the lecture **Deep Learning**.