One-Hot Encoding

\[
\begin{bmatrix}
0 \\
1 \\
0 \\
0 \\
0
\end{bmatrix} \quad \rightarrow \quad \begin{bmatrix}
0 \\
0 \\
0 \\
0 \\
1
\end{bmatrix}
\]
One-Hot Encoding

• Data: \((x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(n)}, y^{(n)})\)
• E.g., \(y^{(i)} \in \{"person", "hamster", "capybara"\}\)
• Encode as \(y^{(i)} \in \{1, 2, 3\}\)?
  • Shouldn’t be running something like linear regression, since “hamster” is not really the average of “person” and “capybara,” so things are not likely to work well (Explanation on the board)

• Solution: one-hot encoding
  • “person” \(\Rightarrow [1, 0, 0]\)
  • “hamster” \(\Rightarrow [0, 1, 0]\)
  • “capybara” \(\Rightarrow [0, 0, 1]\)
Multilayer Neural Network for Classification

- $o_i$ is large if the probability that the correct class is $i$ is high

A possible cost function:

$$ \sum_{i=1}^{m} (o^{(i)} - y^{(i)})^2 $$

$y^{(i)}$'s encoded using one-hot encoding
Softmax

- Want to estimate the probability $P(y = y'|x, \theta)$
  - $\theta$: network parameters

$$p_i = \frac{\exp(o_i)}{\sum_j \exp(o_j)}$$
Softmax

- $p_i = \frac{\exp(o_i)}{\sum_j \exp(o_j)}$ can be thought of as probabilities
- $0 < p_i < 1$
- $\sum_j p_j = 1$
- This is a generalization of logistic regression
  - (For two outputs, $p_1 = \frac{\exp(o_1)}{\exp(o_1)+\exp(o_2)} = \frac{1}{1+\exp(o_2-o_1)}$)
Cost Function: $-\sum_j y_j \log p_j$

- Likelihood (single training case): $P(y_j = 1; x|w)$
  - The probability for $y_j = 1$ that the network outputs with weights $w$
- The likelihood of $y = (0, ..., 0, 1, 0, 0, ..., 0)$ is $p_j$, where $j$ is the index of the non-zero entry in $y$
  - Same as $\prod_j p_j^{y_j}$
- Negative log-likelihood (single training case)
  - $-\sum_j y_j \log p_j$
Cost Function Gradient

\[
p_i = \frac{e^{o_i}}{\sum_j e^{o_j}}
\]

\[
\frac{\partial p_i}{\partial o_i} = p_i \ (1 - p_i)
\]

\[
C = - \sum_j y_j \log p_j
\]

\[
\frac{\partial C}{\partial o_i} = \sum_j \frac{\partial C}{\partial p_j} \frac{\partial p_j}{\partial o_i} = p_i - y_i
\]