CSC 4780/6780
Fall 2022
Homework 0y

October 10, 2022

Softmax is given by:

$$p_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

So, the softmax of $[5, 3, 0, -1]$ is calculated by taking e^z for each of these

$$e^5 = 148.413159102576603$$

$$e^3 = 20.085536923187668$$

$$e^0 = 1$$

$$e^{-1} = 0.367879441171442$$

The sum of these is 169.86657546693573

Dividing each by that sum we get the vector $[0.87370431, 0.11824302, 0.00588697, 0.0021657]$

1 The Jacobian

For any function f, the jacobian of f is the matrix

$$\begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \dots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \dots & \frac{\partial f_2}{\partial x_n} \\ \dots & \dots & \dots & \dots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \dots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

As mentioned in the lecture,

- If $i = j$: $\frac{\partial p_i}{\partial z_j} = p_i\,(1 - p_i))$
- If $i \neq j$: $\frac{\partial p_i}{\partial z_j} = -p_i p_j$

So the jacobian of the softmax at $[5, 3, 0, -1]$ is

$$\begin{bmatrix} 0.1103451 & -0.1033094 & -0.0051435 & -0.0018922 \\ -0.1033094 & 0.1042616 & -0.0006961 & -0.0002561 \\ -0.0051435 & -0.0006961 & 0.0058523 & -0.0000127 \\ -0.0018922 & -0.0002561 & -0.0000127 & 0.0021610 \end{bmatrix}$$

(I know this question seemed arbitrary, but it is one of the things that allows nearly all neural nets that do classification to learn.)

Here is some code that gives you this result:

```
import numpy as np

z = np.array([5, 3, 0, -1])
k = z.shape[0]

ez = np.exp(z)
print(ez)
sum_ez = ez.sum()
print(sum_ez)
result = ez / sum_ez
print(result)

for row in range(k):
    for col in range(k):
        if row == col:
            v = result[row] * (1.0 -  result[row])
        else:
            v = -1.0 * result[row] * result[col]
        print(f"{v:.7f}  ", end="")
    print()
```