تکلیف اول

مجتبی ملائی
۴۰۱۳۱۳۸۳

۱

۱. اگر توابع فعال ساز را حذف کنیم، فرمول نهایی به شکل زیر خواهد بود:

$$\hat{y} = \sum_{j=1}^{4}(\sum_{i=1}^{3} x_i w_{ij}^{(1)} + b_j^{(1)}) w_i^{(2)} + b_j^{(2)}$$

که میتوان آن را به صورت زیر نوشت:

$$\hat{y} = \sum_{j=1}^{4}(\sum_{i=1}^{3} x_i w_{ij}^{(1)} w_j^{(2)} + b_j^{(1)} w_j^{(2)}) + b_j^{(2)}$$

$$\hat{y} = \sum_{i=1}^{3}(x_i(\sum_{j=1}^{4} w_{ij}^{(1)} w_j^{(2)}) + \sum_{j=1}^{4}(b_j^{(1)} w_j^{(2)})) + \sum_{j=1}^{4} b_j^{(2)}$$

حالا اگر قرار دهیم:

$$w_i' = \sum_{j=1}^{4} w_{ij}^{(1)} w_j^{(2)}, b' = \sum_{j=1}^{4} b_j^{(1)} w_j^{(2)} + b_j^{(2)}$$

خواهیم داشت:

$$\hat{y} = \sum_{i=1}^{3} x_i w_i' + b'$$

که در واقع همان ساختار یک شبکه عصبی بدون لایه پنهان است.

۲. آ) $\frac{\partial \mathcal{L}}{\partial \hat{y}} = -\frac{y}{\hat{y}} + \frac{(1-y)}{1-\hat{y}}$

ب) $\frac{\partial \hat{y}}{\partial z^{(2)}} = \sigma(z^{(2)})(1-\sigma(z^{(2)})) = \hat{y}(1-\hat{y})$

ج) $\frac{\partial a_j^{(1)}}{\partial z_j^{(1)}} = 1 - \tanh^2(z_j^{(1)}) = 1 - (a_j^{(1)})^2$

د) $\frac{\partial J}{\partial z^{(2)}} = \frac{\partial J}{\partial \mathcal{L}} \frac{\partial \mathcal{L}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z^{(2)}} = \frac{1}{m}(-\frac{y}{\hat{y}} + \frac{(1-y)}{1-\hat{y}})\hat{y}(1-\hat{y}) = \frac{1}{m}(\hat{y} - y)$

ه) $\frac{\partial \mathcal{L}}{\partial w_{ij}^{(1)}} = \frac{\partial \mathcal{L}}{\partial z^{(2)}} \frac{\partial z^{(2)}}{\partial a_j^{(1)}} \frac{\partial a_j^{(1)}}{\partial z_j^{(1)}} \frac{\partial z_j^{(1)}}{\partial w_{ij}^{(1)}} = (\hat{y} - y)(w_j^{(2)})(1 - (a_j^{(1)})^2)(x_i^{(1)})$

و) $\frac{\partial \mathcal{L}}{\partial b_j^{(1)}} = \frac{\partial \mathcal{L}}{\partial z^{(2)}} \frac{\partial z^{(2)}}{\partial a_j^{(1)}} \frac{\partial a_j^{(1)}}{\partial z_j^{(1)}} \frac{\partial z_j^{(1)}}{\partial b_j^{(1)}} = (\hat{y} - y)(w_j^{(2)})(1 - (a_j^{(1)})^2)$

ز) $\frac{\partial \mathcal{L}}{\partial w_j^{(2)}} = \frac{\partial \mathcal{L}}{\partial z^{(2)}} \frac{\partial z^{(2)}}{\partial w_j^{(2)}} = (\hat{y} - y) a_j^{(1)}$

ح) $\frac{\partial \mathcal{L}}{\partial b^{(2)}} = \frac{\partial \mathcal{L}}{\partial z^{(2)}} \frac{\partial z^{(2)}}{\partial b^{(2)}} = \hat{y} - y$

.۳

forward propagation:

$$z_j^{(1)} = [-0.1425, 0.1036, 0.7293, -0.1154]$$
$$a_j^{(1)} = [-0.14154322, 0.10323094, 0.62263691, -0.11489045]$$
$$z^{(2)} = 0.31613438, \hat{y} = 0.57838188, \mathcal{L} = 0.54752093, J = 0.54752093$$

back-propagation:

$$\frac{\partial J}{\partial w_j^{(2)}} = \begin{bmatrix} -0.05967719 \\ 0.04352403 \\ 0.262515 \\ -0.04843989 \end{bmatrix}, \frac{\partial J}{\partial b^{(2)}} = 0.42161812$$

$$\frac{\partial J}{\partial w_{ij}^{(1)}} = \begin{bmatrix} 0.02375734 & -0.03357857 & 0.00890675 & 0.00478461 \\ 0.04648176 & -0.0656972 & 0.01742624 & 0.00936119 \\ 0.06920618 & -0.09781583 & 0.02594574 & 0.01393777 \end{bmatrix}, \frac{\partial J}{\partial b_j^{(1)}} = \begin{bmatrix} 0.1032928 \\ -0.14599378 \\ 0.03872499 \\ 0.02080264 \end{bmatrix}$$

updated parameters:

$$w_j^{(2)} = w_j^{(2)} - 0.01 * \frac{\partial J}{\partial w_j^{(2)}} = \begin{bmatrix} 0.25059677 \\ -0.35043524 \\ 0.14737485 \\ 0.0504844 \end{bmatrix}, b^{(2)} = b^{(2)} - 0.01 * \frac{\partial J}{\partial b^{(2)}} = 0.29578382$$

$$w_{ij}^{(1)} = w_{ij}^{(1)} - 0.01 * \frac{\partial J}{\partial w_{ij}^{(1)}} = \begin{bmatrix} 0.44976243 & -0.11966421 & 0.77991093 & 0.71995215 \\ 0.04953518 & 0.35065697 & -0.22017426 & -0.85009361 \\ -0.55069206 & 0.11097816 & 0.66974054 & 0.44986062 \end{bmatrix}$$

$$b_j^{(1)} = b_j^{(1)} - 0.01 * \frac{\partial J}{\partial b_j^{(1)}} = \begin{bmatrix} 0.09896707 \\ -0.09854006 \\ 0.19961275 \\ -0.20020803 \end{bmatrix}$$

۲

میتوانیم از نمونه‌برداری مجدد داده‌ها (Resampling) به دو صورت زیر استفاده کنیم

- **Oversampling (افزایش داده‌های اقلیت):** می‌توان داده‌های دارای برچسب ۰ را با تکرار داده‌های موجود یا ایجاد داده‌های مصنوعی افزایش داد.
- **Undersampling (کاهش داده‌های اکثریت):** می‌توان تعداد نمونه‌های دارای برچسب ۱ را کاهش داد تا تعادل برقرار شود.

۳

.۱

step	x	$f(x)$	$f'(x)$
0	5.5	-7175	3867
1	9.367	-26994	-5876
2	15.244	-53615	-1345
3	16.589	-53654	1409
4	15.180	-53525	-1459
5	16.639	-53582	1525

Table 1: $\alpha = 0.001$

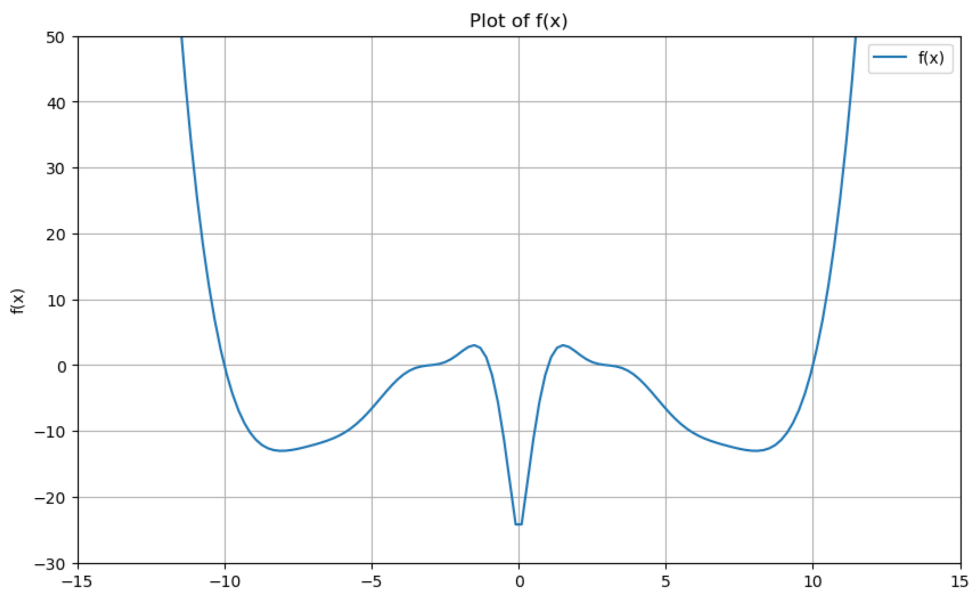
step	x	$f(x)$	$f'(x)$
0	5.5	-7175	-3867
1	15.556	-53946	-772
2	17.564	-51109	3880
3	7.476	-16320	-5272
4	21.185	-15350	16939
5	-22.858	-197967	-7377

Table 2: $\alpha = 0.0026$

.۲

- **نرخ یادگیری:** نرخ یادگیری بر یافتن کمینه جهانی (global minimum) و سرعت همگرایی تأثیر دارد. مقدار زیاد آن باعث همگرایی سریع‌تر و جستجوی گسترده‌تر می‌شود، اما ممکن است نوسان ایجاد کرده و از کمینه عبور کند. مقدار کم نیز سرعت همگرایی را کاهش داده و ممکن است مدل را در کمینه محلی گرفتار کند.
- **تابع هزینه:** تابع هزینه با داشتن مینیمم‌های محلی یا فلات‌های زیاد می‌تواند روند یادگیری را دشوار کند و بر همگرایی تأثیر منفی بگذارد.
- **نقطه شروع:** انتخاب مناسب نقطه شروع می‌تواند مسیر مدل را به سمت نقطه بهینه هدایت کند، در حالی که نقطه شروع نامناسب ممکن است باعث کاهش سرعت همگرایی و گیر افتادن در کمینه محلی شود.

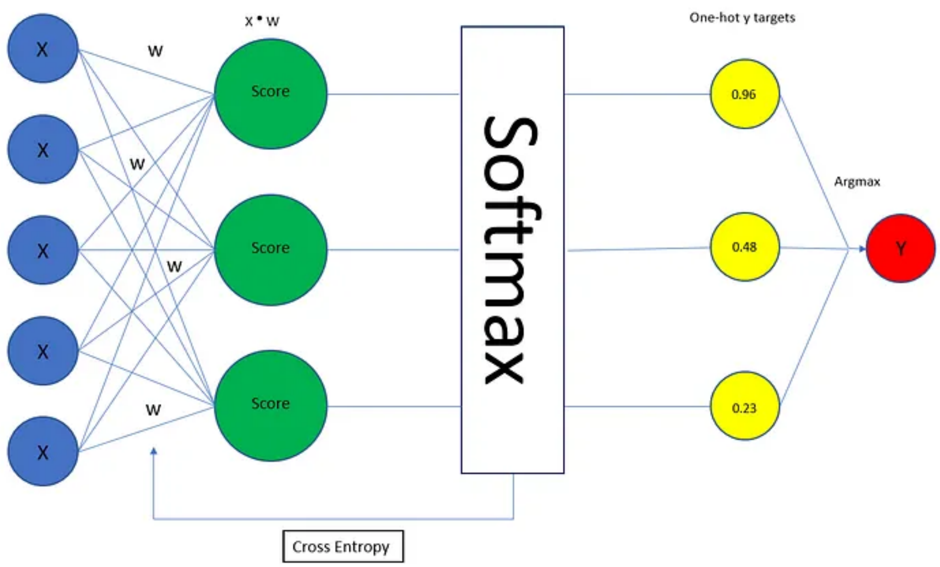
.۳ انتخاب یک نرخ یادگیری مناسب شرط **لازم** برای رسیدن به نقطه بهینه است، اما **به‌تنهایی کافی نیست**. رسیدن به نقطه بهینه علاوه بر نرخ یادگیری، به **تابع هزینه** و **مقادیر اولیه** پارامترها نیز بستگی دارد. به عنوان مثال تابع زیر را در نظر بگیرید. اگر نقطه شروع در نزدیکی نقطه بهینه نباشد، به سمت کمینه های محلی میرود. اگر بخواهیم که از کمینه محلی خارج شویم باید نرخ یادگیری را تا حد زیادی افزایش دهیم که این موضوع نیز باعث میشود تا با تغییر نقطه آِغازین این امکان ایجاد شود که هر بار از روی نقطه بهینه پرش کنیم.



۴

۱. **One-vs-Rest (OvR) Approach (Binary Logistic Regression for Each Class)** میتوان از سه مدل logistic regression استفاده کرد. برای اینکار برای هر کلاس یک مدل میسازیم که تشخیص دهد ار آن کلاس است یا از دو کلاس دیگر. به عنوان مثال خروجی مدل اول نشان میدهد که ورودی از نوع اول است یا نوع دوم و سوم است $(0, (1, 2))$. در نهایت سه مدل خواهیم داشت و برای پیشبینی، کلاسی که بیشترین امتیاز را دارد(خروجی مدل آن بزرگ تر است) را انتخاب می‌کنیم.

۲. **Multinomial Logistic Regression (Softmax Regression)** برای این کار نورون لایه خروجی را حذف میکنیم و بجای آن سه(در حالت کلی به تعداد کلاس های مورد نیاز) نورن قرار میدهیم. سپس خروجی هر نورون را به تابع softmax مدهیم و نهایتا کلاسی که بزرگترین خروجی را دارد را انتخاب میکنیم .



ضابطه تابع softmax برای کلاس i ام:

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

که در آن K تعداد کلاس ها است.

۳. **categorical cross-entropy** میتوانیم از تابع زیر استفاده کنیم:

$$J = -\frac{1}{m} \sum_{i=0}^{m} \sum_{c=0}^{C} y_{i,c} \log(\sigma(\vec{z_i})_c)$$

- اگر مقدار پیشبینی شده برای کلاس پاسخ ($y_{i,c} = 1$) نزدیک به یک باشد: $\log(\sigma(\vec{z_i})_c)$ به سمت ۰ میل میکند پس هزینه را کم میکند که درست است.
- اگر مقدار پیشبینی شده برای کلاس پاسخ ($y_{i,c} = 1$) نزدیک به صفر باشد: $\log(\sigma(\vec{z_i})_c)$ به سمت $-\infty$ میل میکند پس هزینه را زیاد میکند که درست است.
- تابع softmax باعث میشود تا با افزایش احتمال یک کلاس، شانس کلاس های دیگر کم بشود. پس برای وقتی که $y_{i,c} = 0$ اگر تابع عددی نزدیک به یک پیشبینی کرده باشد، هزینه به صورت غیر مستقیم زیاد می‌شود.
- این تابع smooth و convex است. پس برای گرادیان کاهشی و بهینه ساز ها مناسب است.

۵