دانشگاه تهران

پردیس دانشکده‌های فنی

دانشکده برق و کامپیوتر

یادگیری ماشین

تمرین شماره ۳

نام و نام خانوادگی

امیرحسین چمی

دی ۱۴۰۴

فهرست

فهرست شکل‌ها

فهرست جدول‌ها

چکیده

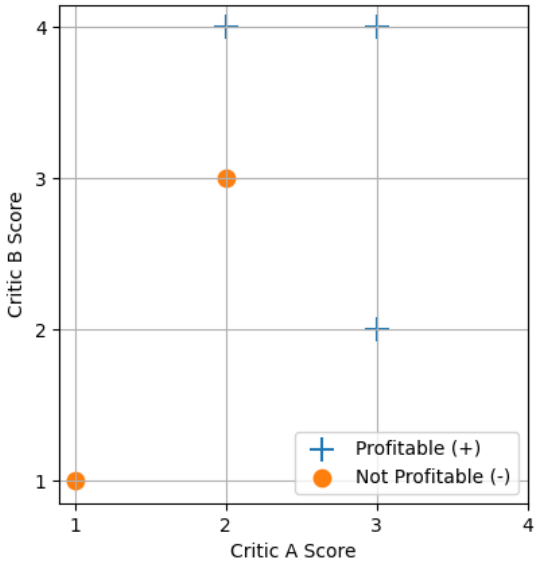
هدف از این مجموعه تمرین، بررسی توانایی، محدودیت‌ها و رفتار یادگیری مدل‌های مختلف یادگیری ماشین از دیدگاه تئوری و عملی است. در بخش‌های ابتدایی، با تحلیل مسئله‌ی پرسپترون، مفهوم جداسازی خطی و وابستگی قدرت مدل به نوع ویژگی‌ها مورد مطالعه قرار می‌گیرد و نشان داده می‌شود که حتی الگوریتم‌های یادگیری ساده نیز در صورت نامناسب بودن نمایش داده، قادر به حل برخی مسائل نخواهند بود. همچنین در ادامه، با بررسی شبکه‌های عصبی با توابع فعال‌سازی ReLU، ظرفیت نمایش (Representational Capacity) شبکه‌ها و نقش تعداد نورون‌ها و لایه‌های میانی در پیاده‌سازی توابع قطعه‌ای خطی تحلیل می‌شود.

در بخش‌های بعدی، تمرین به تحلیل دقیق‌تر شبکه‌های عصبی از منظر پس‌انتشار خطا، مشتق‌گیری و ساختار گرادیان و هسین می‌پردازد و در کنار آن، مفاهیم آماری مهمی مانند LDA و PCA برای درک بهتر نقش توزیع داده و کاهش بُعد بررسی می‌شوند. در نهایت، با اجرای عملی روش‌های Feature Selection شامل Forward Selection و Backward Elimination بر روی یک دسته‌بند Naive Bayes، تأثیر انتخاب ویژگی‌ها بر عملکرد مدل به‌صورت تجربی ارزیابی می‌شود. به نظرم این تمرین تقریبا دید جامعی از ارتباط میان پیچیدگی مدل، نمایش داده و عملکرد عملی در مسائل یادگیری ماشین ارائه می دهد.

پرسش ۱ – Perceptron

۱-۱. رسم نمودار امتیاز B بر حسب امتیاز A

با استفاده از یک کد پایتون نمودار آن را به صورت زیر رسم کردم. علامت مثبت برای فیلم های سودده و منفی برای فیلم های زیان‌ده هستند. (سوال کد استفاده شده را نخواسته به همین دلیل ارائه نمی شود.)



شکل ۱ . نمودار داده ها

واضحا این داده ها را نمی توان با استفاده از یک خط جداسازی کرد.

۱-۲. اجرای یک حلقه از الگوریتم پرسپترون

مرحله به مرحله پیش می رویم.

$$score = -۱.۱ + ۰.۱ + ۰.۱ = -۱ \quad (1)$$

پیش بینی منفی شد و فیلم اول واقعا زیانده بوده پس پیشبینی درست است و به روز رسانی وزن ها نیازی نیست.

$$score = -۱.۱ + ۰.۳ + ۰.۲ = -۱ \quad (2)$$

پیش بینی منفی شد و فیلم دوم سودده بوده پس پیشبینی درست نست و باید وزن ها به روز رسانی شود.

$$W_{new} = W_{old} + y\,X = [-۱,۰,۰] + [۱,۳,۲] = [۰,۳,۲] \quad (3)$$

حال به سراغ نمونه‌ی بعدی می رویم.

$$score = ۰ + ۳.۲ + ۲.۴ = ۱۴ \quad (4)$$

پیش بینی مثبت شد و فیلم سوم واقعا سودده بوده پس پیش بینی درست است و نیازی به آپدیت وزن ها نیست.

$$score = ۰ + ۳.۳ + ۲.۴ = ۱۷ \quad (5)$$

پیش بینی مثبت شد و فیلم چهارم واقعا سودده بوده پس پیش بینی درست است و نیازی به آپدیت وزن ها نیست.

$$score = ۰ + ۳.۲ + ۲.۳ = ۱۲ \quad (6)$$

پیش بینی مثبت شد و فیلم پنجم واقعا زیانده بوده پس پیش بینی درست نست و باید وزن ها به روز رسانی شود.

$$W_{new} = W_{old} + y\,X = [۰,۳,۲] - [۱,۲,۳] = [-۱,۱,-۱] \quad (7)$$

پس وزن نهایی بعد از یک دور $[-۱,۱,-۱]$ است.

۱-۳. وزن ها و جداسازی داده ها

در بخش اول دیدیم که داده ها جداپذیر خطی نیستند. وزن ها در این الگوریتم با یک دور اجرا نتوانستند نقاط را جدا کنند ولی حتی اگر بینهایت دور هم الگوریتم را اجرا کنیم، نمی توانند این داده‌ها را به درستی جدا کنند.

۱-۴. توانایی جداسازی الگوریتم پرسپترون

۱-۴-۱. حالت a

در این حالت مرز تصمیم برابر $A + B = ۸$ است. از آنجا که این مرز تصمیم با یک خط مشخص می شود پس این الگوریتم در این حالت موفق است.

۱-۴-۲. حالت b

این حالت یک ناحیه مربعی را با نقاط (۲و۲)، (۳و۲)، (۲و۳) و (۳و۳) مشخص می کند و یک خط نمی تواند داخل و بیرون مربط را جدا کند پس این الگوریتم در این حالت موفق نیست.

۱-۴-۳. حالت c

در این حالت مرز تصمیم برابر $A - B = ۰$ است. از آنجا که این مرز تصمیم با یک خط مشخص می شود پس الگوریتم پرسپترون در این حالت موفق است.

پرسش ۲– Neural Networks: Representation

منطق حل این سوال واضح و برای تمام شبکه ها و توابع داده شده یکسان است. هر تابع ReLU حداکثر یک شکست را در خط می تواند مدلسازی کند.

شبکه های G_1، G_2، H_1 و H_2 تابع ReLU ندارند پس فقط خط ها را می توانند مدلسازی کنند (یاد بگیرند).

شبکه های G_3، G_4 و G_5 یک ReLU دارند بنابراین حداکثر یک شکست را مدلسازی می کنند (یاد بگیرند).

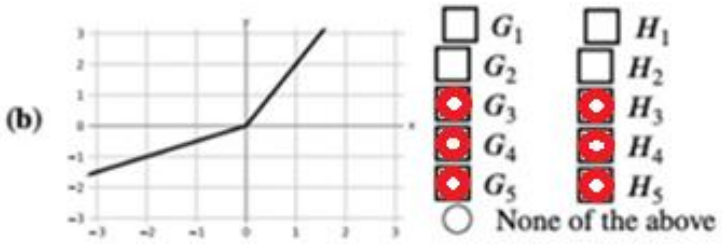
شبکه های H_3، H_4 و H_5 چند ReLU موازی دارند (به دلیل لایه‌های مخفی) بنابراین می توانند حداکثر چند شکست را مدلسازی کنند (یاد بگیرند). بر همین اساس می توانیم تکلیف تمامی نمودارها را مشخص کنیم.

۲-۱. نمودارهای a

با توجه به اینکه هر دوی این نمودارها خط صاف بدون شکستگی هستند، به راحتی می توان با استفاده از همه‌ی این شبکه‌ها این دو نمودار را یاد گرفت.

۲-۲. نمودارهای b

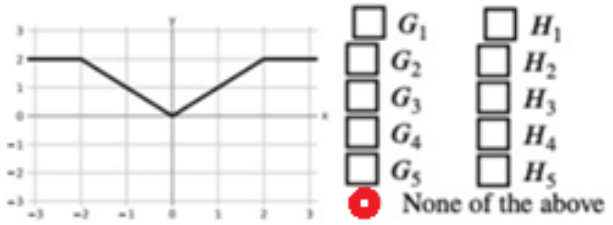
نمودار اول تنها شامل یک شکستگی است.



شکل ۲. نمودار اول b

این نمودار صرفا یک شکستگی دارد پس با استفاده از شبکه های مشخص شده در تصویر (۱) قابل یادگیری است.

نمودار دوم شامل سه شکستگی است.

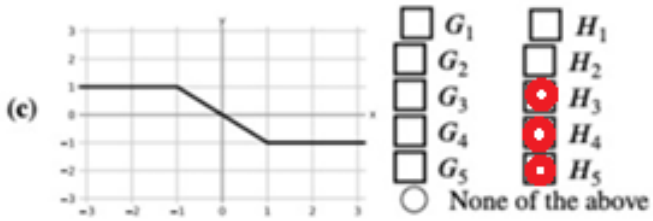


شکل ۳. نمودار دوم **b**

تمامی شبکه هایی که داریم نهایتا می توانند دو تا شکستگی را یاد بگیرند پس با استفاده از این شبکه ها نمی توان این نمودار را یاد گرفت.

۲-۳. نمودار c

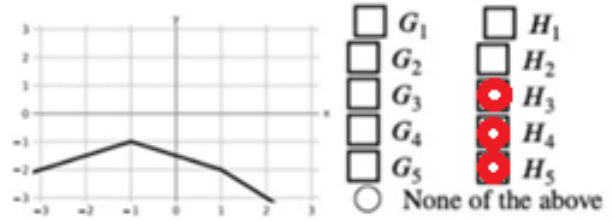
نمودار اول دو شکستگی دارد.



شکل ۴. نمودار اول **c**

از آنجا که دو شکستگی داریم، تنها با استفاده از شبکه‌های مشخص شده می توان در شکل (۳) می توان آن را یاد گرفت.

نمودار دوم دو شکستگی دارد.



شکل ۵. نمودار دوم **c**

از آنجا که دو شکستگی داریم، تنها با استفاده از شبکه‌های مشخص شده می توان در شکل (۳) می توان آن را یاد گرفت.

پرسش ۳– Neural Networks: Forward and Backward Propagation

۳-۱. ابعاد هر پارامتر و وکتورایز کردن شبکه

طبق صورت سوال می دانیم که ابعاد z_1 به صورت $D_{a_1} \times 1$ است. بر همین اساس طبق $z_1 = W_1 x^{(i)} + b_1$ می فهمیم که ابعاد $W_1 x^{(i)}$ باید برابر $D_{a_1} \times 1$ باشد و از آنجا که ابعاد $x^{(i)}$ برابر $D_x \times 1$ است پس ابعاد W_1 باید $D_{a_1} \times D_x$ باشد. در این صورت ابعاد در ضرب ماتریسی همخوانی خواهند داشت. بایاس این رابطه هم باید مطابق خروجی باشد در نتیجه b_1 بعد $D_{a_1} \times 1$ را دارد.

از آنجا که تابع $ReLU$ به صورت elementwise عمل می کند پس ابعاد a_1 هم $D_{a_1} \times 1$ می باشد.

برای پارامتر های بعدی ابتدا باید توجه کنیم که $\hat{y}^{(i)}$ یک اسکالر است (زیرا خروجی یک تابع لاجیستیک است) و همچنین تابع loss هم binary cross-Entropy می باشد. بنابراین تعداد نورون های خروجی صرفا یکی است. پس z_2 باید 1×1 باشد. پس در رابطه‌ی $z_2 = W_2 a_1 + b_2$ پارامتر b_2 و حاصل ضرب $W_2 a_1$ باید 1×1 باشد پس از آنجا که a_1 ابعاد $D_{a_1} \times 1$ دارد پس W_2 باید $1 \times D_{a_1}$ باشد.

حالا می خواهیم شبکه را وکتورایز کنیم. ابتدا ورودی های x و خروجی های y را بررسی می کنیم. در صورت سوال دیدیم که $x^{(i)}$ ابعاد $D_x \times 1$ دارد. حال ماتریس X را به صورت زیر تعریف می کنیم.

$$X = \left[x^{(1)}, x^{(2)}, \dots, x^{(m)}\right] \qquad (۸)$$

یعنی هر داده ی $x^{(i)}$ را به صورت ستونی در یک ماتریس چیدیم. ابعاد X برابر $D_x \times m$ می شود. (Y هم در انتها بررسی خواهد شد.)

در ادامه هر رابطه را بازنویسی کرده و ابعاد را مشخص می کنیم. در رابطه‌ی $Z_1 = W_1 X + B_1$ جمله‌ی $W_1 X$ و پارامتر Z_1 ابعاد $D_{a_1} \times m$ دارند و به صورت زیر هستند.

$$W_1 X = \begin{bmatrix} \vdots & \vdots & & \vdots \\ W_1 x^{(1)} & W_1 x^{(2)} & \dots & W_1 x^{(m)} \\ \vdots & \vdots & & \vdots \end{bmatrix} \qquad (۹)$$

$$Z_1 = \begin{bmatrix} z_1^{(1)} & z_1^{(2)} & \dots & z_1^{(m)} \end{bmatrix} \quad (10)$$

در واقع ضرایب W_1 همان ابعاد قبلی یعنی $D_{a_1} \times D_x$ را دارد و به همین شکل در تک تک ستون های X ضرب می شود. همچنین بایاس برای تمامی نمونه ها یکسان است. پس B_1 به صورت زیر است.

$$B_1 = \begin{bmatrix} b_1 & b_1 & \dots & b_1 \end{bmatrix} \quad (11)$$

ابعاد B_1 باید برابر با جمله‌ی اول باشد، در نتیجه برابر $D_{a_1} \times m$ است. البته توجه کنید این یک راه استاندارد برای نمایش B_1 است. در متلب و کتابخانه های پایتون مثل *numpy* برای جلوگیری از مصرف بی دلیل رم و افزایش بهینگی از روشی به اسم *broadcasting* استفاده می شود. یعنی به جای اینکه b_1 را در یک ماتریس تکرار کنیم، صرفا خود b_1 را به کار گرفته و در محاسبات *numpy*، مشکل ابعاد را به صورت خودکار حل می کند. یعنی تقریبا به صورت زیر عمل می کند. (در مورد *broadcasting* در *numpy* اینجا [1] بخوانید)

$$\mathbf{1}_m = \begin{bmatrix} 1 & 1 & \dots & 1 \end{bmatrix} \quad (12)$$

$$B_1 = b_1 \mathbf{1}_m \quad (8)$$

در این حالت ابعاد B_1 همان $D_{a_1} \times m$ و ابعاد b_1 همچنان $D_{a_1} \times 1$ است.

اتفاق مشابهی در مورد رابطه‌ی $Z_2 = W_2 A_1 + B_2$ می افتد. یعنی:

$$A_1 = \begin{bmatrix} a_1^{(1)} & a_1^{(2)} & \dots & a_1^{(m)} \end{bmatrix} \quad , \quad A_1 \in \mathbb{R}^{D_{a_1} \times m} \quad (9)$$

جمله‌ی $W_2 A_1$ ابعاد $m \times 1$ دارد که به صورت زیر است.

$$W_2 A_1 = \begin{bmatrix} \vdots & \vdots & & \vdots \\ W_2 a_1^{(1)} & W_2 a_1^{(2)} & \dots & W_2 a_1^{(m)} \\ \vdots & \vdots & & \vdots \end{bmatrix} \quad (10)$$

در واقع ضرایب W_2 همان ابعاد قبلی یعنی $1 \times D_{a_1}$ را دارد و به همین شکل در تک تک ستون های A_1 ضرب می شود. همچنین بایاس برای تمامی نمونه ها یکسان است. پس B_2 به صورت زیر است.

$$B_2 = \begin{bmatrix} b_2 & b_2 & \dots & b_2 \end{bmatrix} \quad (11)$$

ابعاد B_2 باید برابر با جمله‌ی اول باشد، در نتیجه برابر $m \times 1$ است و ابعاد b_2 همچنان 1×1 است. (می توان مشابه روابط (۱۲) و (۱۳) نیز B_2 را نشان داد.) نهایتا Z_2 و Y به صورت زیر هستند.

$$Z_2 = \begin{bmatrix} z_2^{(1)} & z_2^{(2)} & \dots & z_2^{(m)} \end{bmatrix} \quad , \quad Z_2 \in \mathbb{R}^{1 \times m} \quad (12)$$

$$Y = \begin{bmatrix} y^{(1)}, y^{(2)}, \dots, y^{(m)} \end{bmatrix} = \sigma(Z_2) \quad (13)$$
$$= \begin{bmatrix} \sigma\left(z_2^{(1)}\right) & \sigma\left(z_2^{(2)}\right) & \dots & \sigma\left(z_2^{(m)}\right) \end{bmatrix} \quad , \quad Y \in \mathbb{R}^{1 \times m}$$

بر این اساس Y هم $m \times ۱$ خواهد بود؛ زیرا $\hat{y}^{(i)}$ یک اسکالر است.

۳-۲. محاسبه‌ی $\frac{\partial J}{\partial \hat{y}^{(i)}}$ و $\frac{\partial J}{\partial \hat{y}}$

طبق صورت سوال میدانیم که:

$$L^{(i)} = y^{(i)} log\hat{y}^{(i)} + \left(۱ - y^{(i)}\right) \log\left(۱ - \hat{y}^{(i)}\right) \quad (14)$$

و تابع هزینه به صورت زیر است.

$$J = -\frac{۱}{m}\sum_{i=۱}^{m} L^{(i)} \quad (۲۰)$$

واضح است که صرفا جمله‌ی i-ام به $\hat{\boldsymbol{y}}^{(i)}$ وابسته بوده و بقیه کنار می روند.

$$\frac{\partial J}{\partial \hat{y}^{(i)}} = -\frac{۱}{m}\frac{\partial L^{(i)}}{\partial \hat{y}^{(i)}} \quad (۲۱)$$

حال از $L^{(i)}$ نسبت به $\hat{y}^{(i)}$ مشتق می گیریم.

$$\frac{\partial L^{(i)}}{\partial \hat{y}^{(i)}} = \frac{\partial}{\partial \hat{y}^{(i)}}\left(y^{(i)} log\hat{y}^{(i)}\right) + \frac{\partial}{\partial \hat{y}^{(i)}}\left(۱ - y^{(i)}\right) \log\left(۱ - \hat{y}^{(i)}\right) \quad (15)$$

$$= \frac{y^{(i)}}{\hat{y}^{(i)}} - \frac{۱ - y^{(i)}}{۱ - \hat{y}^{(i)}}$$

پس خواهیم داشت:

$$\delta_۱^{(i)} = \frac{\partial J}{\partial \hat{y}^{(i)}} = -\frac{۱}{m}\left(\frac{y^{(i)}}{\hat{y}^{(i)}} - \frac{۱ - y^{(i)}}{۱ - \hat{y}^{(i)}}\right) \quad (16)$$

حال می خواهیم $\frac{\partial \boldsymbol{J}}{\partial \hat{\boldsymbol{y}}}$ را، یعنی مشتق نسبت به معادل وکتورایز شده محاسبه کنیم.

$$\frac{\partial J}{\partial \hat{y}} = \left[\frac{\partial J}{\partial \hat{y}^{(۱)}} \quad \frac{\partial J}{\partial \hat{y}^{(۲)}} \quad \dots \quad \frac{\partial J}{\partial \hat{y}^{(m)}}\right] = [\delta^{(۱)} \quad \delta^{(۲)} \quad \dots \quad \delta^{(m)}] \quad (17)$$

با توجه به آنچه در محاسبه‌ی $\frac{\partial \boldsymbol{J}}{\partial \hat{\boldsymbol{y}}^{(i)}}$ دیدیم، $\frac{\partial \boldsymbol{J}}{\partial \hat{\boldsymbol{y}}}$ به صورت زیر خواهد بود.

$$\frac{\partial J}{\partial \hat{y}} = -\frac{۱}{m}\left(\frac{y}{\hat{y}} - \frac{۱ - y}{۱ - \hat{y}}\right) \quad (18)$$

۳-۳. محاسبه‌ی $\frac{\partial \hat{\boldsymbol{y}}^{(i)}}{\partial \boldsymbol{z}_{\boldsymbol{2}}^{(i)}}$

دیدیم که $\hat{y}^{(i)} = \sigma(z_۲)$ و $\sigma(z)$ تابع سیگموید است. ابتدا باید مشتق تابع سیگموید حساب شود. مشتق تابع سیگموید معروف است اما با این حال از ابتدا محاسبه می کنم. تابع سیگموید:

$$\sigma(z) = \frac{۱}{۱ + e^{-z}} \tag{19}$$

محاسبه‌ی مشتق:

$$\frac{d\sigma(z)}{dz} = -(۱ + e^{-z})^{-۲}.(-e^{-z}) = \frac{e^{-z}}{(۱ + e^{-z})^{۲}} \tag{20}$$

می توان به صورت زیر نیز این مشتق را بازنویسی کرد که مشابه فرمول معروف مشتق تابع سیگموید است.

$$\frac{d\sigma(z)}{dz} = \sigma(z)\big(۱ - \sigma(z)\big) = \frac{۱}{۱ + e^{-z}}\left(۱ - \frac{۱}{۱ + e^{-z}}\right) \tag{21}$$

حال از $\hat{y}^{(i)}$ و $z_۲^{(i)}$ استفاده می کنیم.

$$\frac{\partial \hat{y}^{(i)}}{\partial z_۲^{(i)}} = \sigma\left(z_۲^{(i)}\right)\left(۱ - \sigma\left(z_۲^{(i)}\right)\right) \tag{22}$$

از آنجاکه $\hat{y}^{(i)} = \sigma\left(z_۲^{(i)}\right)$ پس به صورت زیر بازنویسی می کنیم.

$$\delta_۲^{(i)} = \frac{\partial \hat{y}^{(i)}}{\partial z_۲^{(i)}} = \hat{y}^{(i)}\big(۱ - \hat{y}^{(i)}\big) \tag{۳۰}$$

۳-۴. محاسبه‌ی $\frac{\partial z_2}{\partial a_1}$

با توجه به $z_۲ = W_۲a_۱ + b_۲$ مشتق آن به صورت زیر است.

$$\delta_۳^{(i)} = \frac{\partial z_۲}{\partial a_۱} = W_۲ \tag{۳۱}$$

۳-۵. محاسبه‌ی $\frac{\partial a_1}{\partial z_1}$

می دانیم:

$$a_۱ = ReLU(z_۱) = \begin{cases} z_۱ & z_۱ > ۰ \\ ۰ & z_۱ \leq ۰ \end{cases} \tag{۳۲}$$

می دانیم که ابعاد $z_۱$ برابر $D_{a_۱} \times ۱$ است. بر این اساس می توان به صورت زیر هم آن را نوشت. $\left(k = ۱, \dots, D_{a_۱}\right)$

$$ReLU(z_{1,k}) = \begin{cases} z_{1,k} & z_{1,k} > 0 \\ 0 & z_{1,k} \leq 0 \end{cases} \quad (۳۳)$$

پس مشتق a_1 نسبت به z_1 به صورت *element-wise* به این شکل خواهد بود.

$$\frac{\partial a_{1,k}}{\partial z_{1,k}} = \begin{cases} 1 & z_{1,k} > 0 \\ 0 & z_{1,k} \leq 0 \end{cases} \quad (۳۴)$$

اگر بخواهیم این مشتق را به صورت برداری/ ماتریسی بنویسیم خواهیم داشت:

$$\delta_4^{(i)} = \frac{\partial a_1}{\partial z_1} = diag\left(1_{z_1^{(i)} > 0}\right) \quad (۳۵)$$

ولی در حالت کلی به این شکل است.

$$\delta_4^{(i)} = \frac{\partial a_1}{\partial z_1} = \frac{\partial}{\partial z_1} ReLU(z_1) = \mathbf{1}_{z>0}. \quad (۳۶)$$

۳-۶. محاسبه‌ی $\frac{\partial z_1}{\partial W_1}$

ابتدا توجه کنید که W_1 یک ماتریس و z_1 یک بردار است. پاسخ این مشتق سه بعد خواهد داشت یعنی انتظار یک تنسور داریم. با یک مثال می توان توضیح داد؛ خروجی k-ام و یک وزن رندوم مثلا $W_{1,ij}$ را در نظر بگیرید. (k اندیس هر درایه از z_1 و i,j اندیس های هر درایه از ماتریس W_1 هستند.)

مقدار $\frac{\partial z_{1,k}}{\partial W_{1,ij}}$ تنها در یک حالت شامل وزن $W_{1,ij}$ خواهد بود و آن هم در صورتی است که $k = i$ باشد در غیر این صورت مشتق صفر خواهد بود. پس در کل:

اگر $k = i$، مشتق $W_{1,kj}x_j$ نسبت به $W_{1,kj}$ برابر x_j خواهد بود و اگر $k \neq i$، مشتق صفر است. به همین دلیل است که این مشتق یک تنسور خواهد بود.

$$\delta_5^{(i)} = \frac{\partial z_{1,k}}{\partial W_{1,ij}} = \begin{cases} x_j & k = i \\ 0 & k \neq i \end{cases} \quad (۳۷)$$

این مشتق به همین شکل حساب شد؛ اما می توان آن را راحت تر و واضح تر با استفاده از تابع دلتای کرونکر نشان داد. این تابع به صورت زیر است:

$$\delta_{ij} = \begin{cases} 0 & i \neq j \\ 1 & i = j \end{cases} \quad (۳۸)$$

در مورد این تابع و خواص آن می توانید در [۲] بخوانید. با استفاده از این تابع می توان مشتق را به صورت زیر بازنویسی کرد.

$$\delta_{۵}^{(i)} = \left(\frac{\partial z_{۱}}{\partial W_{۱}}\right)_{kij} = \delta_{ki} x_j \qquad (۳۹)$$

توجه کنید؛ شاید در ابتدا اینجور به نظر برسد که صرفا در نظر گرفتن حالت $k = i$ کافی است و اگر بر این اساس پیش برویم به یک ماتریس (۲-D) برسیم. ولی چنین کاری اشتباه است. می توان بر اساس مفهوم مشتق آن را توضیح داد. وقتی می خواهیم $\frac{\partial z_{۱}}{\partial W_{۱}}$ را حساب کنیم یعنی می خواهیم ببینیم تغییر هر درایه از $W_{۱}$ چه اثری روی هر عنصر از $z_{۱}$ خواهد داشت، اگر صرفا $k = i$ را در نظر بگیریم یعنی تاثیر تعداد زیادی از درایه های $W_{۱}$ را در نظر نگرفته و بررسی نکرده ایم.

۳-۷. محاسبه‌ی $\frac{\partial J}{\partial W_1}$

با توجه به بخش های قبل به راحتی می توان این مشتق را به صورت زیر حساب کرد.

$$\frac{\partial J}{\partial W_{۱}} = \sum_{i=۱}^{m} \frac{\partial J}{\partial \hat{y}^{(i)}} \frac{\partial \hat{y}^{(i)}}{\partial z_{۲}^{(i)}} \frac{\partial z_{۲}^{(i)}}{\partial a_{۱}^{(i)}} \frac{\partial a_{۱}^{(i)}}{\partial z_{۱}^{(i)}} \frac{\partial z_{۱}^{(i)}}{\partial W_{۱}} = \sum_{i=۱}^{m} \delta_{1}^{(i)} \delta_{۲}^{(i)} \delta_{۳}^{(i)} \delta_{۴}^{(i)} \delta_{۵}^{(i)} \qquad (23)$$

از آنجا که هر کدام از این مشتق ها طولانی بوده و نوشتن اصل روابط صرفا باعث طولانی شدن رابطه شده و کمک خاصی به درک مطلب نمی کرد، صرفا با استفاده علامت گذاری این مشتق نوشته شد.

یک راه دیگر هم برای محاسبه‌ی این مشتق وجود دارد. با توجه به J مشتق آن به صورت زیر است.

$$\frac{\partial J}{\partial W_{۱}} = -\frac{۱}{m} \sum_{i=۱}^{m} \frac{\partial L^{(i)}}{\partial W_{۱}} \qquad (۴۱)$$

با استفاده از قاعده‌ی زنجیره‌ای می توان این مشتق را به صورت زیر نوشت.

$$\frac{\partial L^{(i)}}{\partial W_{۱}} = \frac{\partial L^{(i)}}{\partial \hat{y}^{(i)}} \frac{\partial \hat{y}^{(i)}}{\partial z_{۲}^{(i)}} \frac{\partial z_{۲}^{(i)}}{\partial a_{۱}^{(i)}} \frac{\partial a_{۱}^{(i)}}{\partial z_{۱}^{(i)}} \frac{\partial z_{۱}^{(i)}}{\partial W_{۱}} \qquad (۴۲)$$

توجه کنید که در این مشتق زنجیره‌ای از $\frac{\partial L^{(i)}}{\partial \hat{y}^{(i)}}$ استفاده شده و در بخش های قبلی این مشتق را حساب نکردیم. به همین دلیل صرفا به رابطه‌ی (۴۰) اکتفا کرده و روش دوم محاسبه را ادامه نمی دهیم.

پرسش ۴– Hessian of a Two-Layer Network

در این سوال می خواهیم ماتریس هسین یک شبکه دو لایه را محاسبه کنیم. روابط به صورت زیر هستند.

$$a_j = \sum_i w_{ji}^{(1)} x\_i \quad , \quad z_j = h(a_j) \tag{۴۳}$$

$$a_k = \sum_j w_{kj}^{(2)} z_j$$

و E_n خطای داده‌ی n است. بر این اساس مشتق های E_n را با علائم زیر نشان می دهیم.

$$\delta_k \equiv \frac{\partial E_n}{\partial a_k} \quad , \quad M_{kk'} \equiv \frac{\partial^2 E_n}{\partial a_k \partial a_{k'}} \tag{۴۴}$$

توجه کنید که در این اثبات از i' و i برای مشخص کردن ورودی ها، j' و j برای مشخص کردن لایه های مخفی و از k' و k برای مشخص کردن خروجی ها استفاده شده است.

۴-۱. هر دو وزن در لایه دوم

باید مشتق دوم زیر را حساب کرد که برای محاسبه آن از رابطه‌ی مشتق زنجیره ای استفاده می کنم.

$$\frac{\partial^2 E_n}{\partial w_{kj}^{(2)} \partial w_{k'j'}^{(2)}} = \frac{\partial}{\partial w_{kj}^{(2)}} \left(\frac{\partial E_n}{\partial w_{k'j'}^{(2)}} \right) \tag{۴۵}$$

با محاسبه مشتق داخلی شروع می کنیم.

$$\frac{\partial E_n}{\partial w_{k'j'}^{(2)}} = \frac{\partial E_n}{\partial a_{k'}} * \frac{\partial a_{k'}}{\partial w_{k'j'}^{(2)}} = \delta_{k'} * z_{j'} \tag{۴۶}$$

حالا مشتق دوم این عبارت را محاسبه می کنیم.

$$\frac{\partial}{\partial w_{kj}^{(2)}}\left(\frac{\partial E_n}{\partial w_{k'j'}^{(2)}}\right)=\frac{\partial\left(\delta_{k'}*z_{j'}\right)}{\partial w_{kj}^{(2)}}=\frac{\partial\left(\delta_{k'}*z_{j'}\right)}{\partial a_k}*\frac{\partial a_k}{\partial w_{kj}^{(2)}} \tag{47}$$

$$=z_j*\frac{\partial\left(\delta_{k'}*z_{j'}\right)}{\partial a_k}=z_j\left[z_{j'}*\frac{\partial\delta_{k'}}{\partial a_k}+\delta_{k'}*\frac{\partial z_{j'}}{\partial a_k}\right]$$

عبارت $\frac{\partial z_{j'}}{\partial a_k}$ برابر صفر است. پس

$$\frac{\partial^2 E_n}{\partial w_{kj}^{(2)}\partial w_{k'j'}^{(2)}}=z_j z_{j'}M_{kk'} \tag{48}$$

به این صورت مشتق ها با استفاده از قاعده‌ی زنجیره ای حساب شد.

۴-۲. هر دو وزن در لایه اول

باید مشتق دوم زیر را حساب کرد که برای محاسبه آن از رابطه‌ی مشتق زنجیره ای استفاده می کنم.

$$\frac{\partial^2 E_n}{\partial w_{ji}^{(1)}\partial w_{j'i'}^{(1)}}=\frac{\partial}{\partial w_{ji}^{(1)}}\left(\frac{\partial E_n}{\partial w_{j'i'}^{(1)}}\right) \tag{49}$$

با محاسبه مشتق داخلی شروع می کنیم. می توانیم این کار را با تقسیم به دو حالت $j=j'$ و $j\neq j'$ انجام دهیم. در این صورت اثبات واضح تر خواهد بود.

فرض کنید $j\neq j'$:

$$\frac{\partial E_n}{\partial w_{j'i'}^{(1)}}=\sum_{k'}\frac{\partial E_n}{\partial a_{k'}}\frac{\partial a_{k'}}{\partial w_{j'i'}^{(1)}}=\sum_{k'}\frac{\partial E_n}{\partial a_{k'}}\frac{\partial a_{k'}}{\partial z_{j'}}\frac{\partial z_{j'}}{\partial a_{j'}}\frac{\partial a_{j'}}{\partial w_{j'i'}^{(1)}}= \tag{50}$$

$$=\sum_{k'}\frac{\partial E_n}{\partial a_{k'}}\,w_{k'j'}^{(2)}\,h'\left(a_{j'}\right)x_{i'}$$

عبارات $w_{k'j'}^{(2)}\,h'\left(a_{j'}\right)x_{i'}$ به سادگی از وابستگی $a_{k'}$ به وزن های لایه‌ی دوم ($w_{k'j'}^{(2)}$)، رابطه‌ی z_j و نهایتا a_j با استفاده از مشتق زنجیره ای می آید. حالا مشتق دوم این عبارت را محاسبه می کنیم.

(۵۱)

$$\frac{\partial}{\partial w_{ji}^{(1)}}\left(\frac{\partial E_n}{\partial w_{j'i'}^{(1)}}\right)=\frac{\partial}{\partial w_{ji}^{(1)}}\left(\sum_{k'}\frac{\partial E_n}{\partial a_{k'}}\,w_{k'j'}^{(2)}\,h'(a_{j'})x_{i'}\right)$$

$$=\sum_{k'}h'(a_{j'})x_{i'}\,\frac{\partial}{\partial w_{ji}^{(1)}}\left(\frac{\partial E_n}{\partial a_{k'}}\,w_{k'j'}^{(2)}\right)$$

$$=\sum_{k'}h'(a_{j'})x_{i'}\,\sum_{k}\frac{\partial}{\partial a_k}\left(\frac{\partial E_n}{\partial a_{k'}}\,w_{k'j'}^{(2)}\right)\frac{\partial a_k}{\partial w_{ji}^{(1)}}$$

$$=\sum_{k'}h'(a_{j'})x_{i'}\,\sum_{k}\frac{\partial}{\partial a_k}\left(\frac{\partial E_n}{\partial a_{k'}}\,w_{k'j'}^{(2)}\right)\left(w_{kj}^{(2)}\,h'(a_j)x_i\right)$$

$$=\sum_{k'}h'(a_{j'})x_{i'}\,\sum_{k}M_{k\,k'}w_{k'j'}^{(2)}\left(w_{kj}^{(2)}\,h'(a_j)x_i\right)$$

حال با خارج کردن پارامتر هایی که به سامیشن و ایندکس آن ارتباطی ندارند می توانیم ساده سازی کنیم.

(۵۲)

$$\frac{\partial}{\partial w_{ji}^{(1)}}\left(\frac{\partial E_n}{\partial w_{j'i'}^{(1)}}\right)=x_{i'}x_i h'(a_{j'})h'(a_j)\sum_{k'}\sum_{k}w_{k'j'}^{(2)}\,w_{kj}^{(2)}M_{k\,k'}$$

حالا حالتی را فرض کنید که $j=j'$، فرایند تا حدودی شبیه به حالت قبلی خواهد بود. مشتق داخلی در این حالت به صورت زیر است.

(24)

$$\frac{\partial E_n}{\partial w_{ji'}^{(1)}}=\sum_{k'}\frac{\partial E_n}{\partial a_{k'}}\,w_{k'j}^{(2)}\,h'(a_j)x_{i'}$$

حالا مشتق دوم این عبارت را محاسبه می کنیم.

(25)

$$\frac{\partial}{\partial w_{ji}^{(1)}}\left(\frac{\partial E_n}{\partial w_{ji'}^{(1)}}\right)=\frac{\partial}{\partial w_{ji}^{(1)}}\left(\sum_{k'}\frac{\partial E_n}{\partial a_{k'}}\,w_{k'j}^{(2)}\,h'(a_j)x_{i'}\right)=\sum_{k'}\frac{\partial}{\partial w_{ji}^{(1)}}\left(\frac{\partial E_n}{\partial a_{k'}}\,w_{k'j}^{(2)}\,h'(a_j)x_{i'}\right)$$

$$=x_{i'}\sum_{k'}\frac{\partial}{\partial w_{ji}^{(1)}}\left(\frac{\partial E_n}{\partial a_{k'}}\,w_{k'j}^{(2)}\right)h'(a_j)+x_{i'}\sum_{k'}\left(\frac{\partial E_n}{\partial a_{k'}}\,w_{k'j}^{(2)}\right)\frac{\partial h'(a_j)}{\partial w_{ji}^{(1)}}$$

$$=x_{i'}x_i h'(a_j)h'(a_j)\sum_{k'}\sum_{k}w_{k'j}^{(2)}\,w_{kj}^{(2)}M_{k\,k'}+x_{i'}\sum_{k'}\left(\frac{\partial E_n}{\partial a_{k'}}\,w_{k'j}^{(2)}\right)\frac{\partial h'(a_j)}{\partial w_{ji}^{(1)}}$$

$$=x_{i'}x_i h'(a_j)h'(a_j)\sum_{k'}\sum_{k}w_{k'j}^{(2)}\,w_{kj}^{(2)}M_{k\,k'}+h''(a_j)x_i x_{i'}\sum_{k'}\delta_{k'}w_{k'j}^{(2)}$$

(توجه کنید برای اینکه هر مساوی در یک خط قرار گرفته و روابط شکل قابل درک تری داشته باشند مجبور شدم سایز فونت را کوچک تر کنم. لطفا زوم کنید.)

حالا که در هر دو حالت این مشتق را بررسی کردیم، با جمع روابط (۵۲) و (۵۴) به رابطه‌ی زیر خواهیم رسید.

$$\frac{\partial^2 E_n}{\partial w_{ji}^{(1)} \partial w_{j'i'}^{(1)}} = x_{i'} x_i \left(h'(a_{j'}) h'(a_j) \sum_{k'} \sum_{k} w_{k'j'}^{(2)} w_{kj}^{(2)} M_{k\,k'} + h''(a_j) I_{j\,j'} \sum_{k'} \delta_{k'} w_{k'j}^{(2)} \right) \quad (26)$$

به این صورت مشتق ها با استفاده از قاعده‌ی زنجیره ای حساب شد.

۴-۳. هر وزن از یکی از لایه ها

باید مشتق دوم زیر را حساب کرد که برای محاسبه آن از رابطه‌ی مشتق زنجیره ای استفاده می کنم.

$$\frac{\partial^2 E_n}{\partial w_{ji}^{(1)} \partial w_{kj'}^{(2)}} = \frac{\partial}{\partial w_{ji}^{(1)}} \left(\frac{\partial E_n}{\partial w_{kj'}^{(2)}} \right) \quad (۵۶)$$

با محاسبه مشتق داخلی شروع می کنیم. به طور مشابه قسمت قبل با تقسیم به دو حالت $j = j'$ و $j \neq j'$ پیش می رویم. در این صورت اثبات واضح تر خواهد بود. با حالت $j \neq j'$ شروع می کنیم.

$$\frac{\partial E_n}{\partial w_{kj'}^{(2)}} = \frac{\partial E_n}{\partial a_k} \frac{\partial a_k}{\partial w_{kj'}^{(2)}} = \delta_k \, z_{j'} \quad (۵۷)$$

حالا مشتق این عبارت را نسبت به وزن های لایه اول حساب می کنیم.

$$\begin{aligned} \frac{\partial}{\partial w_{ji}^{(1)}} \left(\frac{\partial E_n}{\partial w_{kj'}^{(2)}} \right) = \frac{\partial (\delta_k \, z_{j'})}{\partial w_{ji}^{(1)}} &= z_{j'} \sum_{k'} \frac{\partial \delta_k}{\partial a_{k'}} \frac{\partial a_{k'}}{\partial w_{ji}^{(1)}} \\ &= z_{j'} \sum_{k'} M_{kk'} \left(w_{k'j}^{(2)} h'(a_j) x_i \right) \\ &= x_i h'(a_j) z_{j'} \sum_{k'} M_{kk'} \, w_{k'j}^{(2)} \end{aligned} \quad (۵۸)$$

حالا حالتی را فرض کنید که $j = j'$، فرایند تا حدودی شبیه به حالت قبلی خواهد بود. مشتق داخلی در این حالت به صورت زیر است.

$$\frac{\partial E_n}{\partial w_{kj}^{(2)}} = \frac{\partial E_n}{\partial a_k} \frac{\partial a_k}{\partial w_{kj}^{(2)}} = \delta_k \, z_j \quad (۵۹)$$

حالا مشتق این عبارت را نسبت به وزن های لایه اول حساب می کنیم.

$$\frac{\partial}{\partial w_{ji}^{(1)}}\left(\frac{\partial E_n}{\partial w_{kj}^{(2)}}\right) = \frac{\partial(\delta_k z_j)}{\partial w_{ji}^{(1)}} = z_j \frac{\partial \delta_k}{\partial w_{ji}^{(1)}} + \delta_k \frac{\partial z_j}{\partial w_{ji}^{(1)}} \qquad (60)$$
$$= x_i h'(a_j) z_j \sum_{k'} M_{kk'} w_{k'j}^{(2)} + \delta_k h'(a_j) x_i$$

حال که هر دو حالت را بررسی کردیم، با جمع روابط (۵۸) و (۶۰) به رابطه‌ی زیر خواهیم رسید.

$$\frac{\partial^2 E_n}{\partial w_{ji}^{(1)} \partial w_{kj'}^{(2)}} = x_i h'(a_j)\left(\delta_k I_{jj'} + z_{j'} \sum_{k'} w_{k'j}^{(2)} M_{kk'}\right) \qquad (61)$$

به این صورت تمامی درایه های ماتریس هسین یک شبکه دو لایه محاسبه شد.

پرسش ۵– Linear Discriminant Analysis (LDA)

در این سوال، نمایشی دیگری از تابع هزینه‌ی LDA را اثبات خواهیم کرد.

فرض شده که دو دیتاست به صورت زیر داریم.

$$Y_1 = \{y_i\}_{i=1}^{n_1} \quad , \quad Y_2 = \{y_j\}_{j=1}^{n_2} \qquad (62)$$

حال از رابطه‌ی اصلی شروع می کنیم.

$$J = \frac{1}{n_1 n_2} \sum_{y_i} \sum_{y_j} (y_i - y_j)^2 = \frac{1}{n_1 n_2} \sum_{y_i} \sum_{y_j} (y_i^2 + y_j^2 - 2 y_i y_j) \qquad (63)$$
$$= \frac{1}{n_1 n_2}\left[\sum_{y_i}\sum_{y_j} y_i^2 + \sum_{y_i}\sum_{y_j} y_j^2 - 2\sum_{y_i}\sum_{y_j} y_i y_j\right]$$

در رابطه‌ی (۶۳)، جملات $\sum_{y_i}\sum_{y_j} y_j^2$ و $\sum_{y_i}\sum_{y_j} y_i^2$ به دلیل جمع شدن روی اندیسی که در جمع وجود ندارد، به ترتیب n_2 بار و n_1 بار تکرار می شوند. همچنین $\sum_{y_i}\sum_{y_j} y_i y_j$ برابر حاصل ضرب میانگین هاست. پس بر این اساس رابطه‌ی (۶۳) به صورت زیر در خواهد آمد.

$$J = \frac{1}{n_1 n_2}\left[n_2 \sum_{y_i} y_i^2 + n_1 \sum_{y_j} y_j^2 - 2 m_1 m_2\right] = \frac{1}{n_1}\sum_{y_i} y_i^2 + \frac{1}{n_2}\sum_{y_j} y_j^2 - 2 m_1 m_2 \qquad (27)$$

حالا مقادیر $۲m_۱^۲$ و $۲m_۲^۲$ را به رابطه‌ی بدست آمده اضافه و کم می کنیم.

$$J = \frac{۱}{n_۱}\sum_{y_i} y_i^۲ + \frac{۱}{n_۲}\sum_{y_j} y_j^۲ - ۲m_۱m_۲ + ۲m_۱^۲ + ۲m_۲^۲ - ۲m_۱^۲ \tag{٦٥}$$
$$= (m_۱ - m_۲)^۲ + \frac{۱}{n_۱}\sum_{y_i} y_i^۲ + \frac{۱}{n_۲}\sum_{y_j} y_j^۲ + m_۱^۲ + m_۲^۲$$
$$- ۲m_۱^۲ - ۲m_۲^۲$$

توجه کنید هر میانگین نمونه‌ای به صورت $m = \frac{۱}{n}\sum_i y_i$ بدست آمده است. پس روابط $۲m_۱^۲$ و $۲m_۲^۲$ را باز می کنیم و با استفاده از تعریف میانگین نمونه ای بازنویسی می کنیم.

$$J = (m_۱ - m_۲)^۲ + \frac{۱}{n_۱}\sum_{y_i} y_i^۲ + \frac{۱}{n_۲}\sum_{y_j} y_j^۲ + m_۱^۲ + m_۲^۲ - ۲m_۱\frac{۱}{n_۱}\sum_{y_i} y_i \tag{٦٦}$$
$$- ۲m_۲\frac{۱}{n_۲}\sum_{y_j} y_j$$

حال می توانیم $s_۱^۲$ و $s_۲^۲$ را بسازیم.

$$J = (m_۱ - m_۲)^۲ + \frac{۱}{n_۱}\sum_{y_i}(y_i^۲ + m_۱^۲ - ۲m_۱y_i) + \frac{۱}{n_۲}\sum_{y_j}\left(y_j^۲ + m_۲^۲ - ۲m_۲y_j\right) \tag{٦٧}$$
$$= (m_۱ - m_۲)^۲ + \frac{۱}{n_۱}\sum_{y_i}(y_i - m_۱)^۲ + \frac{۱}{n_۲}\sum_{y_j}\left(y_j - m_۲\right)^۲$$
$$= (m_۱ - m_۲)^۲ + \frac{۱}{n_۱}s_۱^۲ + \frac{۱}{n_۲}s_۲^۲$$

رابطه‌ی مورد نظر اثبات شد.

پرسش ۶– Feature Conditioning and PCA

در این سوال با استفاده از روش PCA یکی از روابط پایه ای و البته مهم را اثبات می کنیم.

فرض کنید دیتاست $\{x_n\}_{n=1}^{N}$ را داشته باشیم که در آن هر x_n یک بردار D بعدی است و می خواهیم با استفاده از روش PCA این داده ها را به فضای یک بعدی ببریم. بردار w را در این فضای D بعدی فرض کنید که جهت داده ها را در این فضا مشخص می کند. از آنجا که صرفا جهت این بردار برای ما اهمیت دارد و نه اندازه‌ی آن، پس فرض می کنیم این بردار واحد باشد. $(w^T w = 1)$ این فرض بدون از دست رفتن کلیت مسئله بوده و تاثیری بر آنچه می خواهیم نتیجه بگیریم ندارد.

هر داده‌ی x_n با استفاده از بردار w به یک نقطه به صورت $w^T x_n$ تبدیل می شود. میانگین داده های پراجکت شده (تبدیل شده) برابر $w^T \bar{X}$ است که در آن $\bar{X}$ میانگین نمونه ای به صورت زیر است.

$$\bar{X} = \frac{1}{N}\sum_{n=1}^{N} x_n \qquad (۶۸)$$

واریانس این داده های پراجکت شده به صورت زیر است.

$$\frac{1}{N}\sum_{n=1}^{N}(w^T x_n - w^T \bar{X})^2 = w^T S w \qquad (۶۹)$$

در رابطه‌ی (۶۹) پارامتر S ماتریس کوواریانس داده هاست.

$$S = \frac{1}{N}\sum_{n=1}^{N}(x_n - \bar{X})(x_n - \bar{X})^T \quad (۷۰)$$

حال یک مسئله بهینه سازی داریم که باید واریانس $w^T S w$ را ماکزیمم کنیم. اما باید توجه داشت برای اینکه در این ماکزیمم سازی اندازه w به سمت بینهایت میل نکند، $(\|w\| \longrightarrow \infty)$، باید فرض کنیم که بردار مورد نظر همان طور که قبلا هم اشاره شد حتما واحد باشد. $(w^T w = 1)$

حال با این شرط می توان با استفاده از روش لاگرانژ این ماکزیمم سازی را انجام داد. ضریب λ_1 را برای شرط در نظر می گیریم.

$$\mathcal{L} = w^T S w + \lambda_1(1 - w^T w) \quad (۷۱)$$

حال نسبت به w از این رابطه مشتق می گیریم و مساوی صفر قرار می دهیم.

$$Sw - \lambda_1 w = 0 \quad (۷۲)$$

و خواهیم داشت:

$$Sw = \lambda_1 w \quad (۷۳)$$

رابطه (۷۳) دقیقا همان معادله‌ی مقدار ویژه و بردار ویژه است.

پرسش ۷– Feature Conditioning

برای پیاده سازی کد این سوال از [۳] و [٤] استفاده کردم. الگوریتم های انتخاب پیشرونده و حذف پسرونده اجرا شد و نتیجه هر کدام به صورت زیر است.

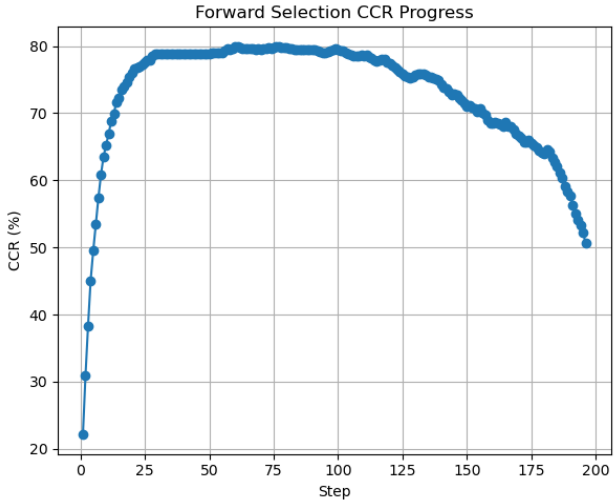
انتخاب پیشرونده:

===== BEST FORWARD SELECTION RESULT =====

Features: [۳۵, ۱۰۲, ۹۱, ۱۰٦, ۱۳۱, ۸۹, ٤۹, ۱۰٤, ٤۷, ٦٥, ۱۳۲, ٦۳, ۷٤, ۱۳۳, ٥۲, ۱٦۲, ۱۳٥, ۷٦, ۷۷, ۷۸, ۱٤٤, ۱٤۹, ۱۱٦, ٦۱, ۱۱۸, ٦٤, ۹۰, ۷٥, ۱٤۸, ۰, ۱, ۲, ۳, ۱۲, ۱۳, ۱٤, ۱٥, ۲۷, ۲۸, ۸٤, ۹۸, ۱٥٤, ۱٦۷, ۱٦۸, ۱۸۱, ۱۸۲, ۱۸۳, ۱۹٤, ۱۹٥, ۱۲۰, ٥۱, ۱۳٤, ۱٤۷, ۸۰, ۱۰۱, ۱۰۳, ٤٦, ٦۰, ۷۹, ٤۸, ۱٤٦]

Best CCR: ۷۹,۹٦٪

نمودار تغییر CCR نسبت به پیشروی الگوریتم (هر مرحله) به صورت زیر است.



شکل ۶. نمودار **CCR** الگوریتم انتخاب پیشرونده

همانگونه که انتظار داشتیم این نمودار تا جایی به صورت صعودی بوده و پس از آن انتخاب ویژگی های بیشتر باعث کاهش CCR شده است.

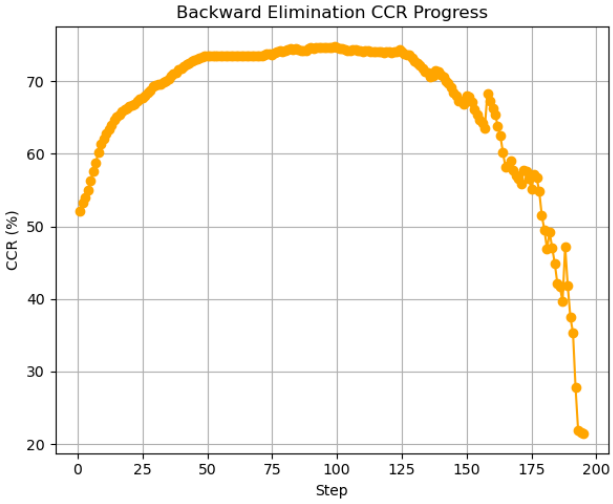
حذف پسرونده:

===== BEST BACKWARD ELIMINATION RESULT =====

Features: [۷, ۹, ۱۱, ۱۸, ۲۰, ۲۲, ۲۳, ۲۵, ۲۶, ۳۱, ۳٤, ۳٦, ۳۷, ۳۸, ۳۹, ٤۰, ٤٤, ٤٥, ٤٦, ٤۷, ٤۸, ٤۹, ٥۰, ۵۲, ۵۳, ۵۵, ۶۰, ۶۱, ۶۳, ۶۴, ۶۶, ۶۸, ۶۹, ۷۰, ۷۱, ۷۴, ۷۵, ۷۷, ۷۹, ۸۰, ۸۲, ۸۷, ۸۸, ۸۹, ۹۰, ۹۱, ۹۲, ۹۳, ۹۵, ۹۶, ۹۷, ۱۰۱, ۱۰۲, ۱۰۳, ۱۰۴, ۱۰۵, ۱۰۶, ۱۱۰, ۱۱۶, ۱۱۸, ۱۲۰, ۱۲۱, ۱۲۶, ۱۳۰, ۱۳۱, ۱۳۲, ۱۳۴, ۱۳۵, ۱۳۶, ۱۳۸, ۱۴۰, ۱۴۴, ۱۴۵, ۱۴۸, ۱۴۹, ۱۵۱, ۱۵۶, ۱۵۷, ۱۵۸, ۱۵۹, ۱۶۱, ۱۶۲, ۱۷۰, ۱۷۵, ۱۷۶, ۱۷۷, ۱۷۹, ۱۸۰, ۱۸۵, ۱۸۶, ۱۸۷, ۱۸۸, ۱۸۹, ۱۹۰, ۱۹۱, ۱۹۲, ۱۹۳]

Best CCR: ۷٤,۸۰٪

نمودار تغییر CCR نسبت به پیشروی الگوریتم (هر مرحله) به صورت زیر است.



شکل ۷. نمودار **CCR** الگوریتم حذف پسرونده

همانگونه که انتظار داشتیم این نمودار تا جایی به صورت صعودی بوده و پس از آن حذف ویژگی های بیشتر باعث کاهش CCR شده است.

پرسش ۸– Computational Exercise: Flower Image Classification

هدف این آزمایش، بررسی و مقایسه‌ی عملکرد مدل‌های مختلف یادگیری عمیق در مسئله‌ی طبقه‌بندی تصاویر گل‌ها است. در این تمرین، ابتدا چند مدل به صورت آموزش از صفر پیاده‌سازی و ارزیابی شدند و سپس از یادگیری انتقالی با استفاده از شبکه‌های از پیش آموزش دیده استفاده گردید.

۸–۱. پیش پردازش داده‌ها

تمام تصاویر به اندازه‌ی ۲۲۴ × ۲۲۴ تغییر مقیاس داده شدند. برای سازگاری با مدل‌های از پیش آموزش‌دیده، از نرمال‌سازی استاندارد ImageNet استفاده شد:

$$\text{Mean} = [۰,٤٨٥, ۰,٤٥٦, ۰,٤۰٦]$$
$$\text{Std} = [۰,۲۲۹, ۰,۲۲٤, ۰,۲۲٥]$$

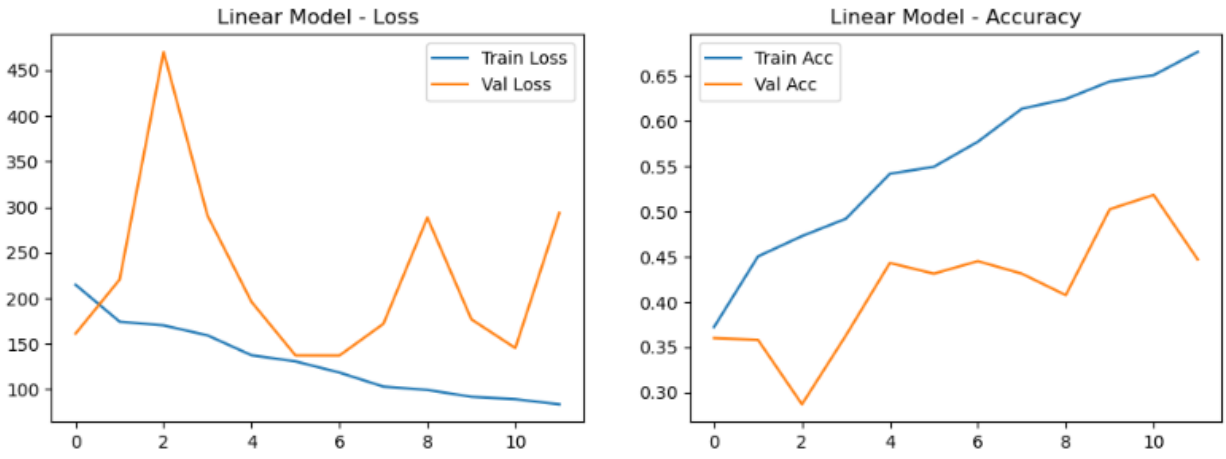
این پیش‌پردازش باعث پایداری بیشتر آموزش و بهبود همگرایی مدل‌ها شد.

۸–۲. بخش اول: آموزش از صفر

در این بخش، سه مدل مختلف که همگی بدون استفاده از وزن‌های از پیش آموزش دیده و به صورت کامل از ابتدا آموزش داده شدند، مورد بررسی قرار گرفتند. هدف از این بخش، تحلیل تاثیر افزایش ظرفیت مدل و استفاده از غیرخطی سازی و تکنیک‌های منظم سازی بر عملکرد نهایی شبکه بود. تمامی مدل‌ها با استفاده از حلقه‌ی آموزش دستی و مکانیزم Early Stopping آموزش داده شدند تا از بیش برازش جلوگیری شود.

۸–۲–۱. مدل خطی

اولین مدل مورد بررسی یک مدل خطی ساده بدون استفاده از هرگونه تابع فعال ساز غیرخطی بود. این مدل تنها قادر به یادگیری مرزهای تصمیم خطی است و در نتیجه توانایی محدودی در استخراج الگوهای پیچیده‌ی موجود در تصاویر دارد. نتایج نشان داد که این مدل به دقت اعتبارسنجی حدود ۵۲ درصد دست یافته است که بیانگر underfitting و ناکافی بودن ظرفیت مدل برای این مسئله است.



شکل ۸. نمودار دقت و **loss** مدل خطی

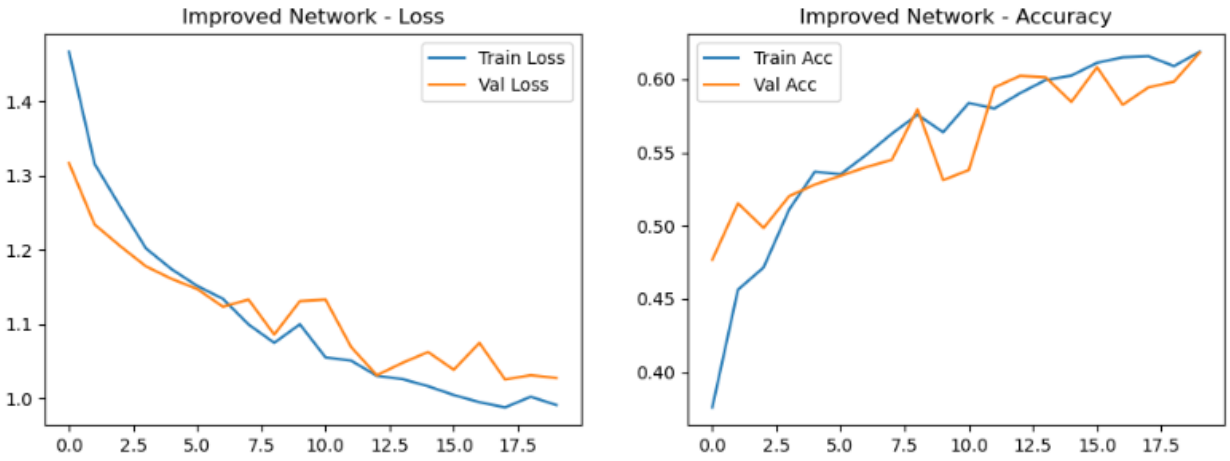
۸-۲-۲. شبکه با یک لایه پنهان و ReLU

در مدل دوم، با اضافه کردن یک لایه‌ی پنهان و استفاده از تابع فعال ساز ReLU، امکان یادگیری روابط غیرخطی فراهم شد. این تغییر منجر به بهبود قابل توجه عملکرد مدل گردید و دقت اعتبارسنجی به حدود ۶۸ درصد افزایش یافت. این مدل بهترین عملکرد را در میان مدل‌های آموزش از صفر ارائه داد و به وضوح نشان داد که استفاده از غیرخطی‌سازی نقش کلیدی در افزایش قدرت بیان شبکه‌های عصبی دارد.

شکل ۹. نمودار دقت و **loss** شبکه با یک لایه پنهان و **ReLU**

۸-۲-۳. شبکه‌ی بهبود یافته

مدل سوم به عنوان شبکه‌ی بهبود یافته شامل تکنیک‌هایی نظیر Batch Normalization، Dropout و regularization بود. این شبکه رفتار پایدارتری در طول آموزش از خود نشان داد و اختلاف بین دقت آموزش و اعتبارسنجی در آن کاهش یافت. با این حال، دقت اعتبارسنجی این مدل در حدود ۶۲ درصد باقی ماند که کمی کمتر از مدل قبلی است. این نتیجه نشان می‌دهد که اعمال regularization قوی در دیتاست‌های نسبتا کوچک می‌تواند منجر به محافظه کاری بیش از حد مدل و کاهش عملکرد نهایی شود.



شکل ۱۰. نمودار دقت و **loss** شبکه بهبود یافته

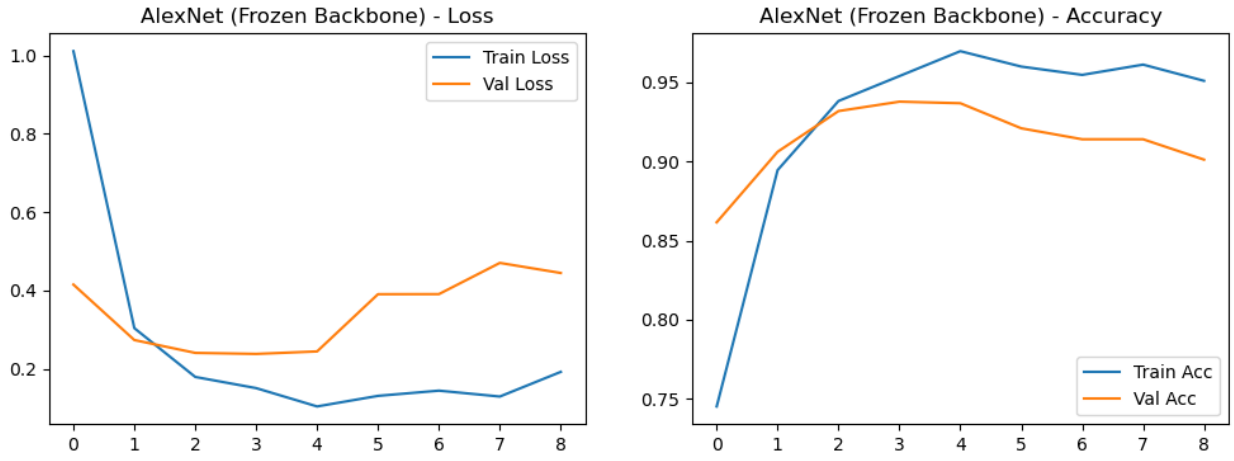
۸-۳. بخش دوم: یادگیری انتقالی

در این بخش از پروژه، از یادگیری انتقالی با استفاده از شبکه‌های AlexNet و VGG۱۶ که قبلا روی دیتاست ImageNet آموزش داده شده‌اند استفاده شد. در هر دو مدل، لایه‌های استخراج ویژگی فریز شدند و تنها لایه‌های نهایی طبقه بندی متناسب با تعداد کلاس‌های مسئله آموزش داده شدند. انتظار می‌رود که این رویکرد به دلیل بهره‌گیری از ویژگی‌های سطح بالای از پیش آموخته شده، عملکرد بهتری نسبت به آموزش کامل از صفر ارائه دهد.

۸-۳-۱. مدل AlexNet

مدل AlexNet با یک classifier جدید روی ویژگی‌های استخراج شده آموزش داده شد. نتایج نشان داد که این مدل از همان epoch های ابتدایی به دقت بالایی دست پیدا می‌کند. دقت اعتبارسنجی در epoch اول حدود ۸۶٪ بود و با ادامه‌ی آموزش افزایش یافت. بهترین دقت اعتبارسنجی این مدل در حدود ۹۳/۸٪ به دست آمد که نشان‌دهنده‌ی عملکرد بسیار قوی AlexNet در این مسئله است.

پس از این نقطه، با افزایش تعداد epoch ها، دقت اعتبارسنجی به تدریج کاهش یافت، در حالی که دقت آموزش همچنان بالا باقی ماند. این رفتار نشان دهنده‌ی شروع بیش برازش است و تایید می‌کند که استفاده از early stopping برای انتخاب بهترین مدل ضروری بوده است.

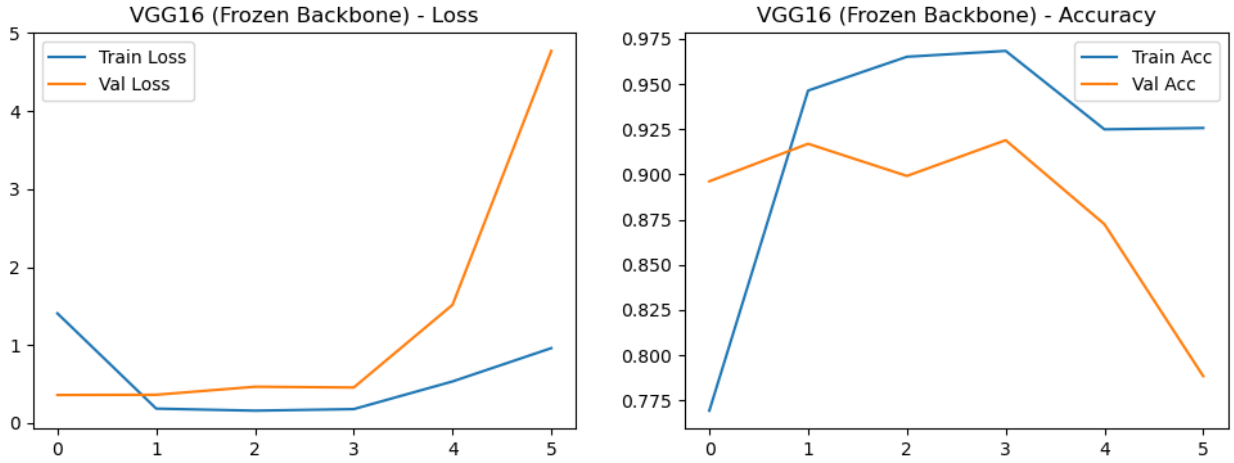


شکل ۱۱. نمودار دقت و loss مدل **AlexNet**

۸-۳-۲. مدل VGG۱۶

در ادامه، مدل VGG۱۶ با رویکرد مشابه مورد استفاده قرار گرفت. این شبکه به دلیل عمق بیشتر، ظرفیت بالاتری برای استخراج ویژگی‌های پیچیده دارد. نتایج نشان داد که VGG۱۶ نیز در epochهای ابتدایی به دقت بالایی دست می‌یابد و بهترین دقت اعتبارسنجی آن حدود ۹۱/۹٪ ثبت شد.

با این حال، با ادامه‌ی آموزش، کاهش محسوسی در دقت اعتبارسنجی مشاهده شد، به‌طوری که در epochهای پایانی افت شدیدی در عملکرد رخ داد. این موضوع نشان می‌دهد که VGG۱۶ نسبت به AlexNet سریع‌تر دچار overfitting شده است، که می‌تواند به دلیل تعداد پارامترهای بسیار بیشتر این شبکه و محدود بودن حجم دیتاست باشد.



شکل ۱۲. نمودار دقت و loss مدل **VGG۱۶**

۸-۴. مقایسه و تحلیل نهایی

مقایسه‌ی نتایج به دست آمده نشان می‌دهد که استفاده از یادگیری انتقالی منجر به بهبود بسیار چشمگیر عملکرد مدل‌ها نسبت به آموزش از صفر شده است. در حالی که بهترین مدل در Part I دقتی در حدود ۶۸٪ داشت، مدل AlexNet توانست دقت اعتبارسنجی نزدیک به ۹۴٪ را ارائه دهد.

در مقایسه‌ی دو مدل انتقالی، AlexNet عملکرد پایدارتر و بهتری نسبت به VGG۱۶ نشان داد. اگرچه VGG۱۶ شبکه‌ای عمیق‌تر است، اما در این مسئله خاص و با حجم داده‌ی محدود، دچار بیش برازش سریع‌تری شد. این نتایج نشان می‌دهد که انتخاب مدل مناسب در یادگیری انتقالی تنها به عمق شبکه وابسته نیست و باید با توجه به اندازه و ماهیت دیتاست انجام شود.

جدول ۱. جدول مقایسه مدل ها

مدل	روش آموزش	بهترین دقت اعتبار سنجی(درصد)
خطی	آموزش از ابتدا	۵۲
یک لایه پنهان با ReLU	آموزش از ابتدا	۶۷.۹
شبکه بهبود یافته	آموزش از ابتدا	۶۲
AlexNet	یادگیری انتقالی	۹۳.۸
VGG۱۶	یادگیری انتقالی	۹۱.۹

۸-۵. نتیجه گیری

در این تمرین نشان داده شد که مدل‌های ساده آموزش دیده از صفر، اگر چه قادر به یادگیری الگوهای اولیه هستند، اما عملکرد آن‌ها در مقایسه با مدل‌های مبتنی بر یادگیری انتقالی به مراتب ضعیف‌تر است. افزودن غیرخطی سازی باعث بهبود قابل توجه عملکرد شد، اما بیشترین جهش دقت با استفاده از شبکه‌های از پیش آموزش‌دیده به دست آمد.

به طور کلی، نتایج این آزمایش تایید می‌کند که یادگیری انتقالی یک راهکار بسیار موثر برای مسائل یادگیری ماشین، به ویژه در شرایطی با داده‌ی محدود، محسوب می‌شود و می‌تواند بدون نیاز به آموزش کامل شبکه از ابتدا، دقت بسیار بالاتری ارائه دهد.

مراجع

[۱] Available: [درون خطی]. "broadcasting," https://numpy.org/doc/stable/user/basics.broadcasting.html.

[۲] Available: [درون خطی]. "wikipedia," Kronecker delta, https://en.wikipedia.org/wiki/Kronecker_delta.

[۳] Available: [درون خطی]. "scikit-learn," GaussianNB, https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.GaussianNB.html.

[۴] Available: [درون خطی]. "scikit-learn," accuracy_score, https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html.