به نام خدا

دانشگاه تهران

دانشکدگان علوم و فناوری های میان رشته ای

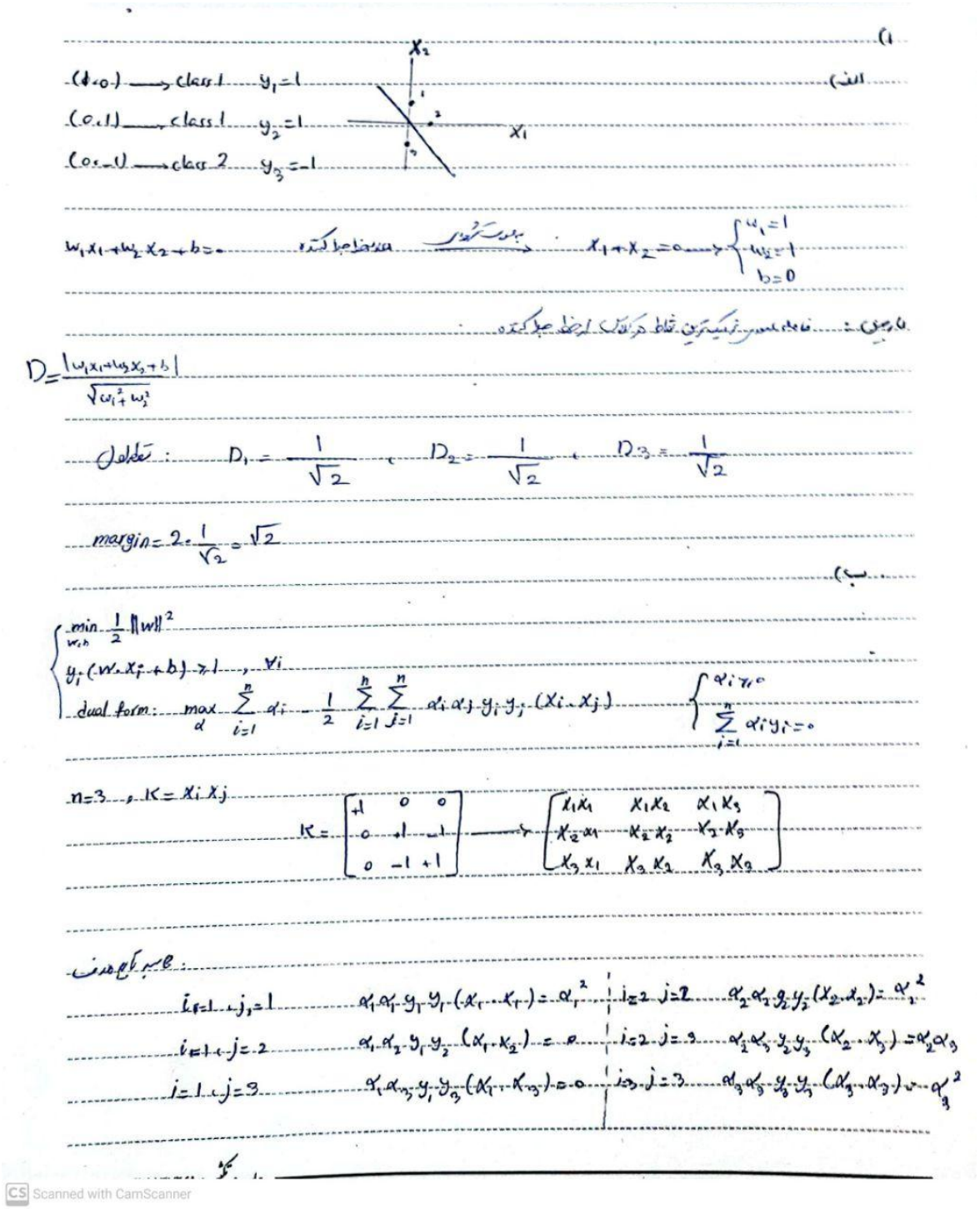
Neural Networks

تمرین دوم

نام و نام خانوادگی	OMID MORADI
شماره دانشجویی	
تاریخ ارسال گزارش	

سوال اول

الف)



$$\longrightarrow \frac{1}{2}\sum_{i=1}^{3}\sum_{j=1}^{3}\alpha_i\alpha_j y_i y_j (x_i \cdot x_j) = \frac{1}{2}\left(\alpha_1^2+\alpha_2^2+\alpha_3^2+2\alpha_2\alpha_3\right)$$

مسئله ←
$$\max_{\alpha_1,\alpha_2,\alpha_3} \alpha_1+\alpha_2+\alpha_3-\frac{1}{2}\left(\alpha_1^2+\alpha_2^2+\alpha_3^2+2\alpha_2\alpha_3\right)$$

قیود: $\alpha_1,\alpha_2,\alpha_3 \geq 0$ ، $\alpha_1 y_1+\alpha_2 y_2+\alpha_3 y_3 = 0 \rightarrow \boxed{\alpha_1+\alpha_2=\alpha_3}$

KKT:

$$\frac{\partial L}{\partial \alpha_1} = 1-\alpha_1 = 0 \rightarrow \alpha_1 = 1$$

$$\frac{\partial L}{\partial \alpha_2} = 1-\alpha_2+\alpha_3 = 0 \rightarrow \alpha_2+\alpha_3 = 1 \longrightarrow \begin{cases}\alpha_1=1\\ \alpha_2=0\\ \alpha_3=1\end{cases}$$

$$\frac{\partial L}{\partial \alpha_3} = 1-\alpha_3+\alpha_2 = 0 \rightarrow \alpha_2-\alpha_3 = -1$$

قید خط: $\alpha_1+\alpha_2=\alpha_3 \rightarrow 1+0=1$ ✓✓

معادله خط:

$$w = \sum_{i=1}^{n}\alpha_i y_i x_i = (1)(1)(1,0)+(0)(1)(0,1)+(1)(-1)(0,-1) = (1,1)$$

نتیجه: معادله خط جداکننده بهینه $x_1+x_2=1$ است.

ب) برای اینکه بفهمیم کدام نقاط بردار پشتیبان هستند باید بررسی کنیم که کدام نقاط شرایط KKT را برآورده کنند و در مرز حاشیه

1) $y_i(wx_i+b)=1$ ← به ازای تمام نقاط ۳ نقطه بردار پشتیبان هستند

2) $\alpha_i \geq 0$

سوال دوم

الف)

$x_{in1} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$

$w_1 = \begin{bmatrix} 0 & 1 & 2 \\ 0 & -1 & 0 \end{bmatrix}$ $\qquad z_1 = w^T x + b = \begin{bmatrix} 0 & 1 \\ 1 & -1 \\ 2 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} + 1 = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$

$b = 1$

$$z_1' = \sigma(z_1) = \begin{bmatrix} \frac{1}{1+e^{-1}} \\ \frac{1}{1+e^{-2}} \\ \frac{1}{1+e^{-3}} \end{bmatrix} = \begin{bmatrix} 0.73 \\ 0.88 \\ 0.95 \end{bmatrix}$$

خروجی:

$$z_{out} = w_2^T z_1' + b = \begin{bmatrix} 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.73 \\ 0.88 \\ 0.95 \end{bmatrix} + 1 = 2.68$$

$$z_{out}' = \sigma(z_{out}) = \frac{1}{1+e^{-2.68}} = 0.936$$

Error:

$$MSE = \frac{1}{2}(y_{true} - z_{out}')^2 = \frac{1}{2}(1-0.936)^2 = 0.002$$

backpropagation:

$$\delta_{out} = \frac{\partial E}{\partial z_{out}} = (a_{out} - y_{true}) \cdot \sigma'(z_{out})$$ ← فرمول اصلی

$$\sigma'(z_{out}) = a_{out}(1 - a_{out})$$

$$\delta_{out} = (a_{out} - y_{true}) \cdot a_{out}(1 - a_{out})$$

$$= (0.936 - 1)\, 0.936\, (1 - 0.936) = -0.0038$$

به‌روزرسانی وزن‌ها در لایه‌ی خروجی:

$$w_2 = w_2 - \eta \cdot \delta_{out} \cdot a_{hidden} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} - 0.5\,(-0.0038) \begin{bmatrix} 0.73 \\ 0.88 \\ 0.95 \end{bmatrix}$$

$$= \begin{bmatrix} 1.0014 \\ 0.0017 \\ 1.0018 \end{bmatrix}$$

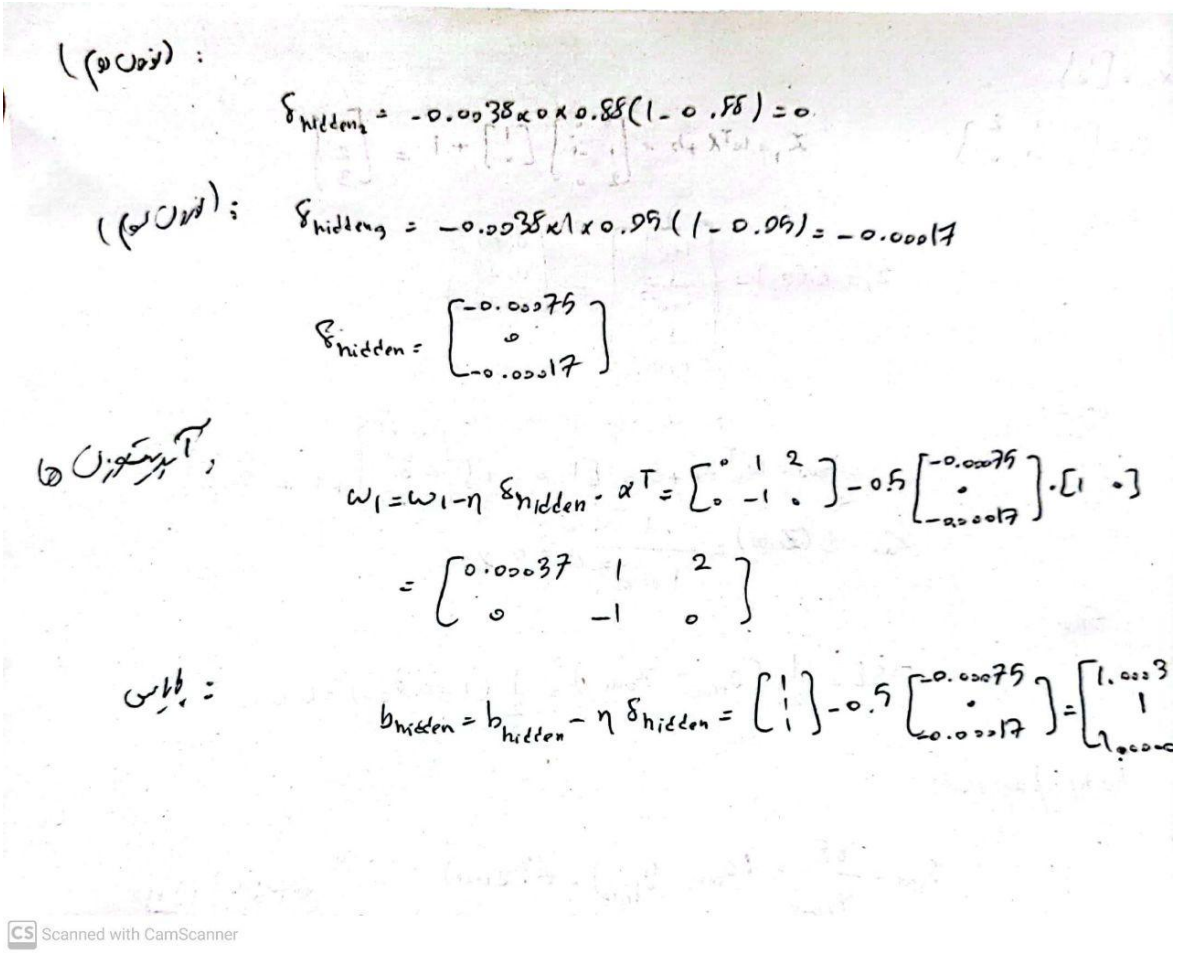
بایاس لایه‌ی خروجی:

$$b_{out} = b_{out} - \eta \cdot \delta_{out} = 1 - 0.5(-0.0038) = 1.0019$$

لایه‌ی مخفی:

(نورون اول) $\qquad \delta_{hidden_1} = \delta_{out} \cdot w_2 \cdot a_{hidden}(1 - a_{hidden})$

$$= -0.0038 \times 1 \times 0.73(1 - 0.73) = -0.00075$$



ب)

1. غیرخطی کردن خروجی‌ها
2. ایجاد مرزهای تصمیم‌گیری پیچیده
3. افزایش توانایی شبکه در یادگیری الگوهای پیچیده
4. کنترل و محدود کردن خروجی‌ها
5. مدل‌سازی مفاهیم غیرخطی در داده‌ها

پ)

1. **محو شدن گرادیان**

یکی از مشکلات سیگموئید، پدیده محو شدن گرادیان است. در این حالت، اگر ورودی بسیار بزرگ یا بسیار کوچک باشد، مشتق سیگموئید تقریباً به صفر نزدیک می‌شود. این موضوع در شبکه‌های عمیق باعث می‌شود گرادیان در طی پس‌انتشار کاهش یابد و لایه‌های ابتدایی نتوانند به خوبی به‌روزرسانی شوند. برای

رفع این مشکل، استفاده از ReLU پیشنهاد می‌شود، زیرا در ناحیه مثبت، مشتق آن برابر با 1 است و از محو شدن گرادیان جلوگیری می‌کند. همچنین نرمال‌سازی ورودی‌ها (مثلاً استاندارد کردن داده‌ها) می‌تواند از ورود مقادیر بیش از حد بزرگ یا کوچک به تابع سیگموئید جلوگیری کند.

2. خروجی‌های غیرصفر مرکزی(Non-Zero Centered Outputs)

مشکل دیگر سیگموئید این است که خروجی‌های آن **غیرصفر مرکزی** هستند، یعنی مقادیر خروجی همیشه در بازه [0,1] قرار دارند. این موضوع باعث می‌شود گرادیان‌ها همیشه یا مثبت یا صفر باشند، که الگوریتم گرادیان نزولی را کند می‌کند. برای حل این مشکل، می‌توان از تابع تانژانت هایپربولیک (Tanh) استفاده کرد که خروجی آن در بازه [−1,1]قرار دارد و گرادیان‌ها را متقارن‌تر می‌کند. استفاده از روش‌های بهینه‌سازی پیشرفته مانند Adam نیز می‌تواند تأثیر این مشکل را کاهش دهد.

برتری RELU نسبت به Sigmoid:

ReLUنسبت به سیگموئید مزایای زیادی دارد. این تابع محاسبات بسیار ساده‌تری دارد، زیرا فقط مقدار حداکثر (0,x)را محاسبه می‌کند، در حالی که سیگموئید به محاسبات نمایی نیاز دارد. علاوه بر این، ReLU به دلیل صفر کردن خروجی برای مقادیر منفی، یک خروجی پراکنده تولید می‌کند که باعث ساده‌تر شدن ساختار شبکه می‌شود. همچنین، در شبکه‌های عمیق، ReLUعملکرد بهتری دارد زیرا از محو شدن گرادیان جلوگیری کرده و سرعت یادگیری را افزایش می‌دهد.

بنابراین، ReLUبه دلیل سادگی، کارایی بالا و توانایی حل مشکلاتی که سیگموئید ایجاد می‌کند، انتخاب بهتری برای شبکه‌های عمیق است.

ت)

در مسیر پیشرو(Forward Pass) ، تابع فعال‌ساز ReLU به صورت f(x)=max(0,x) عمل می‌کند. اگر ورودی (x) مثبت باشد، خروجی همان مقدار ورودی است، و اگر ورودی منفی یا صفر باشد، خروجی صفر می‌شود. این رفتار باعث می‌شود که مقادیر مثبت بدون تغییر به لایه بعدی منتقل شوند و مقادیر منفی به صفر تبدیل شوند. بنابراین، برخی از نورون‌ها خاموش شده و در محاسبات بعدی تأثیر ندارند.

مثال عددی:

ورودی 2:

$$f(x) = \max(0, x) = (0, 2) = 2$$

ورودی 3-:

$$f(x) = \max(0, x) = (0, -3) = 0$$

در مسیر پس‌رو(Backward Pass) ، مشتق تابع ReLU نسبت به ورودی محاسبه می‌شود.

برای ورودی‌های مثبت($x>0$) ، مشتق برابر با 1 است، که به این معنی است که گرادیان به طور کامل به لایه‌های قبلی منتقل می‌شود و یادگیری ادامه پیدا می‌کند. اما برای ورودی‌های غیرمثبت($x\leq0$) ، مشتق برابر صفر است، و گرادیانی به وزن‌های مرتبط با این نورون‌ها منتقل نمی‌شود. این باعث می‌شود که این نورون‌ها به اصطلاح "خاموش" شوند و دیگر یاد نگیرند.

این رفتار مزایایی دارد، مانند جلوگیری از محو شدن گرادیان و تسریع یادگیری، زیرا مشتق ReLU برای مقادیر مثبت ثابت است. اما مشکل اصلی این است که نورون‌هایی که ورودی غیرمثبت دریافت می‌کنند ممکن است برای همیشه خاموش باقی بمانند (مشکل نورون‌های مرده). برای رفع این مشکل می‌توان از نسخه‌های اصلاح‌شده ReLU مانند Leaky ReLU استفاده کرد، که برای مقادیر منفی یک گرادیان کوچک (مثلاً$0.01x$) در نظر می‌گیرد. این کار باعث می‌شود نورون‌های خاموش همچنان امکان یادگیری داشته باشند. گزینه دیگر ReLU6است که خروجی ReLU را در بازه [0,6]محدود می‌کند و کنترل بیشتری روی گرادیان فراهم می‌کند.

به طور کلی، ReLUدر مسیر پیشرو و پس‌رو برای مقادیر مثبت بسیار مؤثر عمل می‌کند و در شبکه‌های عمیق به دلیل جلوگیری از محو شدن گرادیان محبوب است.

برای ورودی 2 گرادیان:

$$f'(x) = 1$$

برای ورودی 3- گرادیان:

$$f'(x) = 0$$

سوال سوم

ب)

بررسی مقادیر گمشده:(Missing Values)

- **دلیل :**مقادیر گمشده می‌توانند باعث خطا در مدل‌سازی شوند. روش‌های مختلفی مانند حذف ردیف‌ها، پر کردن مقادیر با میانگین، میانه، یا مدل‌سازی آماری وجود دارند.

حذف یا مدیریت مقادیر پرت:(Outliers)

- **دلیل :**مقادیر پرت می‌توانند تأثیر زیادی بر الگوریتم‌های حساس به مقیاس داشته باشند. حذف یا تعدیل این مقادیر باعث می‌شود مدل پایدارتر شود.

مقیاس‌بندی داده‌ها:(Scaling)

- **دلیل :**در الگوریتم‌هایی که از فاصله برای محاسبات استفاده می‌کنند (مانند رگرسیون یا الگوریتم‌های مبتنی بر درخت)، مقیاس متغیرها اهمیت زیادی دارد.

رمزگذاری متغیرهای دسته‌ای:(Encoding)

- **دلیل :**داده‌های دسته‌ای باید به فرم عددی تبدیل شوند تا الگوریتم‌های یادگیری ماشین بتوانند با آن‌ها کار کنند.

حذف یا ترکیب ویژگی‌های غیرمفید:(Feature Selection/Engineering)

- **دلیل :**برخی ویژگی‌ها ممکن است همبستگی بالایی با یکدیگر داشته باشند یا اطلاعات کمی برای مدل‌سازی ارائه دهند.

تبدیل داده‌های نامتقارن یا غیرخطی:

- **دلیل :**داده‌های نامتقارن می‌توانند نتایج مدل را تحت تأثیر قرار دهند. استفاده از تبدیلات لگاریتمی یا جذر می‌تواند توزیع داده‌ها را متعادل کند.

پ)

1)

- **Grid Search:** جستجوی سازمان‌یافته روی تمام ترکیب‌های ممکن از مقادیر پارامترهای مشخص شده. دقیق است اما زمان‌بر.
- **Random Search:** نمونه‌برداری تصادفی از ترکیبات پارامترها در فضای جستجو. سریع‌تر است و برای فضاهای بزرگ‌تر مؤثرتر.

2)

- **Linear Kernel:** برای داده‌های خطی؛ **پارامتر مهم :**نیاز به پارامتر اضافی خاصی ندارد.
- **RBF Kernel:** مناسب برای داده‌های غیرخطی؛ **پارامترها:**
  - Gamma: تأثیرگذاری نقاط دور و نزدیک.
  - C: جریمه خطای دسته‌بندی.
- **Polynomial Kernel:** برای مسائل با روابط چندجمله‌ای؛ **پارامترها:**
  - degree: درجه چندجمله‌ای.
  - C: میزان تحمل خطا.

3)

- **One-vs-Rest (OvR):** هر کلاس را در مقابل سایر کلاس‌ها مقایسه می‌کند .**مزیت :**ساده‌تر و سریع‌تر برای تعداد کلاس زیاد.
- **One-vs-One (OvO):** تمام جفت کلاس‌ها را مقایسه می‌کند .**مزیت :**معمولاً برای داده‌های کوچک‌تر بهتر عمل می‌کند.

در این مسئله :اگر دسته‌بندی چندکلاسی باشد، بسته به تعداد کلاس‌ها یکی از این روش‌ها لازم است. OvR برای تعداد کلاس‌های زیاد مناسب‌تر است. در نتیجه برای این مثال ovr مناسب نیست.

ث)

مدل SVM با استفاده از روش GridSearch برای بهینه‌سازی پارامترها به نتایج قابل‌توجهی دست یافت. بهترین پارامترهای به‌دست‌آمده شامل C برابر با 1، degreeبرابر با 2، gammaبا مقدار "scale" و کرنل "rbf"هستند. امتیاز اعتبارسنجی متقاطع (Cross-Validation) مدل 77.89 درصد گزارش شده است. دقت مدل روی داده‌های تست برابر با 82.23 درصد است که عملکرد مطلوب آن را نشان می‌دهد.

ماتریس درهم‌ریختگی نشان می‌دهد که مدل توانسته است 56.58 درصد از نمونه‌های منفی واقعی را به‌درستی منفی پیش‌بینی کند و 25.66 درصد از نمونه‌های مثبت واقعی را نیز به‌درستی به‌عنوان مثبت شناسایی کند. خطاهای False Positive و False Negative به ترتیب برابر با 8.55 درصد و 9.21

درصد هستند. این مقادیر نشان می‌دهند که مدل در ایجاد تعادل بین پیش‌بینی‌های مثبت و منفی عملکرد مناسبی داشته است.

به‌طور کلی، مدل عملکرد خوبی روی داده‌ها ارائه داده است و برای بهبود بیشتر می‌توان روش‌های پیش‌پردازش داده را تقویت کرد یا از داده‌های آموزشی بیشتری استفاده کرد.

سوال چهارم

ب)

جلوگیری از بیش‌برازش:
مدل باید روی داده‌های دیده‌نشده آزموده شود تا بتوان گفت عملکرد آن به داده‌های خاص وابسته نیست.

تنظیم هایپرپارامترها:
استفاده از مجموعه اعتبارسنجی کمک می‌کند بهترین تنظیمات را برای مدل انتخاب کنیم.

ارزیابی بی‌طرفانه:
مجموعه آزمون به‌عنوان داده‌های "کاملاً نادیده" استفاده می‌شود تا ارزیابی منصفانه‌ای از مدل ارائه دهد.

ت)

روند بهبود :افزایش تعداد نورون‌ها و اضافه کردن لایه‌های بیشتر باعث بهبود دقت مدل شده است.

مشکل احتمالی :دقت کلی هنوز پایین است. این موضوع ممکن است ناشی از:

1. نیاز به تنظیم بهتر هایپرپارامترها (مانند نرخ یادگیری یا تعداد دوره‌ها).
2. حجم کم داده‌های آموزشی یا کیفیت پایین داده‌ها.
3. مدل ساده‌ای که هنوز برای این داده‌ها کافی نیست

	Model	Test Accuracy
0	1 Hidden Layer (16 Neurons)	0.16
1	1 Hidden Layer (128 Neurons)	0.12
2	2 Hidden Layers (64, 128 Neurons)	0.08

ث)

نرمال‌سازی داده‌ها را به یک بازه ثابت، معمولاً [0, 1] یا [-1, 1]، مقیاس‌بندی می‌کند. این روش در مواقعی که مقیاس داده‌ها اهمیت دارد یا الگوریتم از فاصله‌محور استفاده می‌کند) مانند KNN و (SVM بسیار مفید است. با این حال، نرمال‌سازی به مقادیر پرت حساس است، زیرا این مقادیر می‌توانند به شدت بر محدوده داده‌ها تاثیر بگذارند.

استانداردسازی، برعکس، داده‌ها را به گونه‌ای تغییر می‌دهد که میانگین آن‌ها صفر و واریانس یک شود. این روش برای داده‌هایی با توزیع گاوسی مناسب است و تاثیر مقادیر پرت را کاهش می‌دهد.
استانداردسازی در الگوریتم‌هایی که به توزیع داده‌ها حساس هستند (مانند رگرسیون خطی و شبکه‌های عصبی) اغلب بهتر عمل می‌کند.

نتایج کد:

```
Test Accuracy with Normalization: 0.20499999821186066

Test Accuracy with Standardization: 0.41499999165534973
```

استانداردسازی معمولاً برای داده‌های تصویری مانند miniMNIST که شامل مقادیر عددی گسترده‌ای است، عملکرد بهتری ارائه می‌دهد. دلیل این موضوع این است که استانداردسازی ویژگی‌های هر نمونه را حول میانگین صفر و انحراف معیار یک متمرکز می‌کند، که باعث می‌شود شبکه عصبی بهتر بتواند اطلاعات مهم را استخراج کند و از اثرات مقادیر پرت جلوگیری کند.

در مقابل، نرمال‌سازی ممکن است برای داده‌هایی که به طور طبیعی در یک بازه محدود نیستند یا دارای مقادیر پرت قابل توجهی هستند، کارایی کمتری داشته باشد، زیرا بازه ثابت [0, 1] نمی‌تواند تفاوت‌های پیچیده بین مقادیر داده را به خوبی مدیریت کند.

ج)

با نرخ یادگیری 1e-9:

	Model	Test Accuracy
0	1 Hidden Layer (16 Neurons)	0.155
1	1 Hidden Layer (128 Neurons)	0.090
2	2 Hidden Layers (64, 128 Neurons)	0.135

با نرخ یادگیری 1e-5:

	Model	Test Accuracy
0	1 Hidden Layer (16 Neurons)	0.045
1	1 Hidden Layer (128 Neurons)	0.135
2	2 Hidden Layers (64, 128 Neurons)	0.045

نتایج با نرخ یادگیری $1e{-5}$:

1. **مدل با 16 نورون در یک لایه مخفی** :دقت بسیار پایین (4.5٪). این نشان می‌دهد که با نرخ یادگیری $1e{-5}$، مدل نتوانسته به خوبی به حداقل خطا نزدیک شود و عملاً یادگیری موثری رخ نداده است.
2. **مدل با 128 نورون در یک لایه مخفی** :دقت کمی بهتر (13.5٪)، که نشان می‌دهد تعداد بیشتر نورون‌ها به مدل کمک کرده است ویژگی‌های بیشتری از داده را بیاموزد، اما نرخ یادگیری هنوز نتوانسته مدل را بهینه کند.
3. **مدل با دو لایه مخفی (64 و 128 نورون)** :دقت بسیار پایین (4.5٪). این نتیجه نشان می‌دهد که با وجود معماری پیچیده‌تر، نرخ یادگیری ناکافی بوده و مدل عملاً به درستی آموزش ندیده است.

نتایج با نرخ یادگیری 1e−9:

1. **مدل با 16 نورون در یک لایه مخفی** :دقت 14.5٪. این مدل با نرخ یادگیری بسیار پایین عملکرد بهتری نسبت به 1e−5داشته است، زیرا احتمالاً توانسته تغییرات بسیار کوچک در وزن‌ها را به دقت مدیریت کند.
2. **مدل با 128 نورون در یک لایه مخفی** :دقت 10٪. در اینجا عملکرد ضعیف‌تر نسبت به مدل اول نشان می‌دهد که نرخ یادگیری بسیار پایین ممکن است از رسیدن به یک حداقل مطلوب جلوگیری کرده باشد.
3. **مدل با دو لایه مخفی (64 و 128 نورون)** :دقت 8.5٪. معماری پیچیده‌تر در اینجا حتی با نرخ یادگیری بسیار پایین عملکرد ضعیف‌تری داشته است، احتمالاً به دلیل نیاز به نرخ یادگیری مناسب‌تر برای یادگیری موثر در لایه‌های عمیق‌تر.

چ)

بیش‌برازش:(Overfitting)

زمانی رخ می‌دهد که مدل به‌طور دقیق داده‌های آموزشی را یاد می‌گیرد اما در تعمیم‌دهی به داده‌های جدید و نادیده (مانند داده‌های آزمون) عملکرد ضعیفی دارد. این معمولاً زمانی رخ می‌دهد که مدل بیش از حد پیچیده باشد یا داده‌های آموزشی ناکافی باشند.

نشانه‌ها:

- دقت زیاد روی مجموعه آموزش، اما دقت پایین روی مجموعه اعتبارسنجی یا آزمون.
- افزایش مقدار Validation Lossبا کاهش مقدار .Training Loss

- **راهکارها برای پیشگیری و حل:**

1. **رگولاراسیون:(Regularization)**
   - اضافه کردن جریمه‌هایی به تابع هزینه، مانند L1 یا.L2
2. **Dropout:**
   - به صورت تصادفی غیرفعال کردن درصدی از نورون‌ها در طول آموزش برای کاهش پیچیدگی مدل.

کم‌برازش:(Underfitting)

زمانی رخ می‌دهد که مدل حتی روی داده‌های آموزشی نیز عملکرد ضعیفی دارد و نتوانسته الگوهای موجود در داده‌ها را به درستی یاد بگیرد. این معمولاً زمانی رخ می‌دهد که مدل بیش از حد ساده باشد یا زمان کافی برای آموزش نداشته باشد.

نشانه‌ها:

- دقت پایین روی مجموعه آموزش.
- عدم کاهش مقدار Training Lossحتی پس از تعداد زیادی اپوک.

- **راهکارها برای پیشگیری و حل:**

1. **افزایش پیچیدگی مدل:**
   - اضافه کردن تعداد نورون‌ها یا لایه‌ها.
2. **کاهش نرخ یادگیری:(Learning Rate)**
   - کاهش نرخ یادگیری برای بهبود جزئیات یادگیری مدل.

تحلیل نتایج مراحل قبل با این مفاهیم:

بیش برازش برای دو حالت با 128 نورون و دولایه

کم برازش برای حالت با 16 نورون

سوال پنجم

الف)

برای یافتن مرز تصمیم باید نقاطی که توابع با هم اشتراک دارند را پیدا کنیم :

class 1 = class 2 :

$$x - y + 1 = -x - y + 2 \Rightarrow x = \frac{1}{2}$$

class 1 = class 3 :

$$x - y + 1 = -3 + y - x \Rightarrow x - y = -2$$

class 1 = class 4 : $x - y + 1 = x + y \longrightarrow y = 1/2$

class 2 = class 3 :

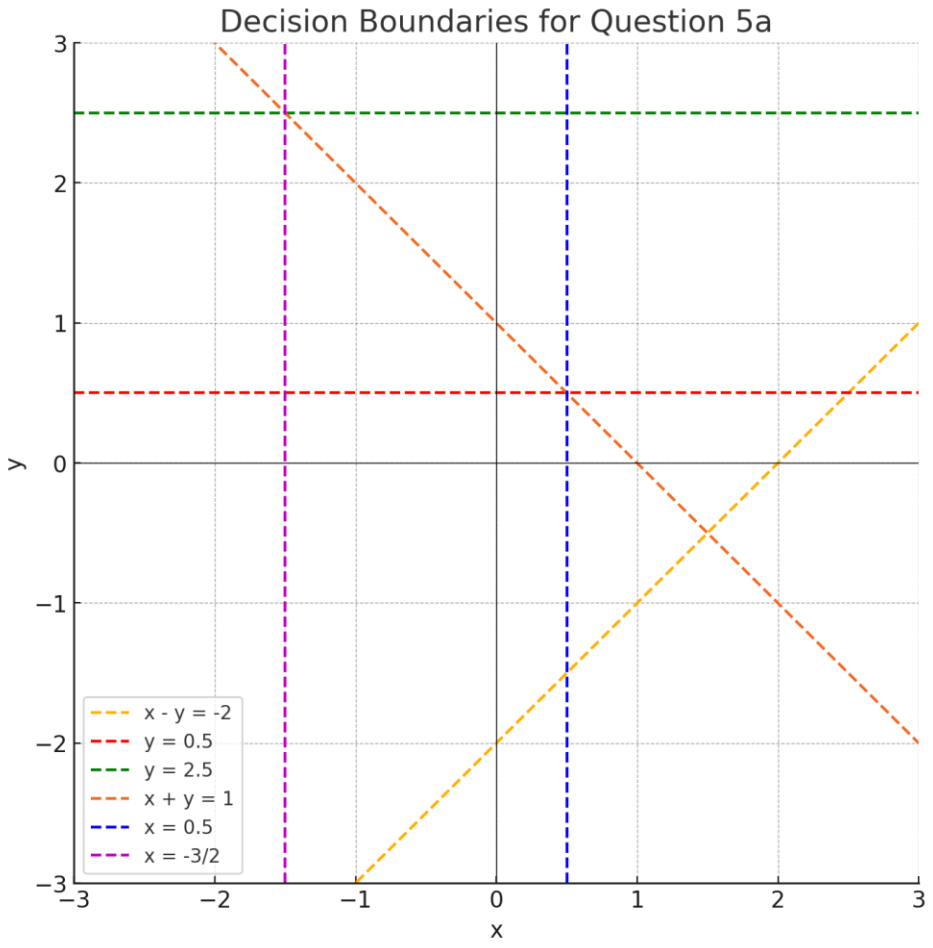
$$-x - y + 2 = -x + y - 3 \longrightarrow y = 5/2$$

class 2 = class 4 :

$$-x - y + 2 = x + y \longrightarrow x + y = 1$$

class 3 = class 4 :

$$-x + y + 3 = x + y \longrightarrow x = 3/2$$



ب)

- اگر داده‌ها به صورت خطی جداشدنی باشند، یک پرسپترون با **دو گره ورودی** کفایت می‌کند.
- اگر داده‌ها به صورت غیرخطی جداشدنی باشند، نیاز به افزودن ویژگی‌هایی است تا داده‌ها در فضای ویژگی جدید جداشدنی شوند.

پ)

1)

برای حل این بخش، باید شبکه‌ای طراحی شود که ورودی‌های X1,X2,..., Xn را دریافت کرده و تابع اکثریت را محاسبه کند. تابع اکثریت به این صورت است که اگر تعداد ورودی‌هایی که مقدار 1 دارند از نصف تعداد کل ورودی‌ها (n/2) بیشتر باشد، خروجی برابر با 1 است. در غیر این صورت، خروجی برابر با 0 خواهد بود.

این شبکه یک لایه ورودی دارد که شامل n نورون است، و وظیفه آن دریافت ورودی‌های Xi است. وزن هر ورودی برابر با 1 در نظر گرفته می‌شود تا مقدار هر ورودی مستقیماً به جمع نهایی اضافه شود. برای تصمیم‌گیری، در لایه خروجی یک بایاس (b) با مقدار n/2− تعریف می‌شود. خروجی شبکه از معادله

$$f\left(\sum_{i=1}^{n} WXi\right) + b$$

به دست می‌آید، که f یک تابع فعال (مانند سیگموئید یاHeaviside) است.

2)

تمامی مفادیر با توجه به فرمول $\sum w_i x_i + b$ مجاسبه شده.

گره α

مقدار	وزن
0	w1α
1	w2α
0	wα

گره β

مقدار	وزن

1	w1β
0	w2β
-4	wβ

گره γ

مقدار	وزن
0	w1γ
1	w2γ
-4	wγ

گره δ

مقدار	وزن
1	wαδ
1/2	wβδ
-1/2	wδ

گره A

مقدار	وزن
1	wδA
1	wγA
-1	wA

گره B

مقدار	وزن
1	wαB
1	wβB
-1	wγB
-1.5	wB

گره C

مقدار	وزن
-1	wαC
1	wβC
-1	wC

منابع