سوال یک

الف)پاسخ به صورت زیر است:

سوال ①

الف)

$P(0) = \frac{1}{2}$, $P(1) = \frac{1}{2}$

$P(\text{مناوری} \mid label=0) = \frac{2}{9}$ $\quad$ $P(\text{مناوری} \mid label=1) = \frac{1}{8}$

$P(\text{فرهنگی} \mid label=0) = \frac{2}{9}$ $\quad$ $P(\text{فرهنگی} \mid label=1) = \frac{1}{8}$

$P(\text{علمی} \mid label=0) = \frac{2}{9}$ $\quad$ $P(\text{علمی} \mid label=1) = \frac{1}{8}$

$P(\text{اقتصادی} \mid label=0) = \frac{1}{9}$ $\quad$ $P(\text{اقتصادی} \mid label=1) = \frac{1}{8}$

$P(\text{اجتماعی} \mid label=0) = \frac{1}{9}$ $\quad$ $P(\text{اجتماعی} \mid label=1) = \frac{2}{8}$

$P(\text{سیاسی} \mid label=0) = \frac{1}{9}$ $\quad$ $P(\text{سیاسی} \mid label=1) = \frac{2}{8}$

داده تست اول

داده تست اول

$P(label=0, \text{مناوری, فرهنگی, علمی, اجتماعی}) = \frac{1}{2} \times \frac{2}{9} \times \frac{2}{9} \times \frac{2}{9} \times \frac{1}{9} = \frac{4}{9^4}$

$P(label=1, \text{فرهنگی, فرهنگی, علمی, اجتماعی}) = \frac{1}{2} \times \frac{1}{8} \times \frac{1}{8} \times \frac{1}{8} \times \frac{2}{8} = \frac{1}{8^4}$

$\frac{4}{9^4} \not\bigcirc \frac{1}{8^4} \rightarrow 8^5 \bigcirc 2 \times 9^4$

$32768 \not\bigcirc 13122$

با توجه اینکه $\frac{4}{9^4}$ بزرگتر است پس داده تست اول متعلق به لیبل ۰ تعلق دارد.

برای داده تست دوم با توجه به اینکه یکی از ویژگی‌ها هیچ کدام از داده‌های آموزشی نیست پس احتمال آن هم برای کلاس ۰ و هم کلاس ۱ برابر با $\boxed{0}$ است.

ب)پاسخ این قسمت به صورت زیر است:

ب)

$$P_{LAP,k}(x|y) = \frac{c(x,y)+k}{c(y)+k|x|}$$

" این فرمول باید بزنیم "

$P(\text{شادی} \mid label=0) = \frac{2+1}{9+6} = \boxed{\frac{1}{5}}$ $\quad P(\text{شادی} \mid label=1) = \frac{1+1}{8+6} = \boxed{\frac{1}{7}}$

$P(\text{زمستان} \mid label=0) = \frac{2+1}{9+6} = \boxed{\frac{1}{5}}$ $\quad P(\text{زمستان} \mid label=1) = \frac{1+1}{8+6} = \frac{1}{7}$

$P(\text{علی} \mid label=0) = \frac{2+1}{9+6} = \boxed{\frac{1}{5}}$ $\quad P(\text{علی} \mid label=1) = \frac{1+1}{8+6} = \frac{1}{7}$

$P(\text{استقلال} \mid label=0) = \frac{1+1}{9+6} = \frac{2}{15}$ $\quad P(\text{استقلال} \mid label=1) = \frac{1+1}{8+6} = \frac{1}{7}$

$P(\text{امتحان} \mid label=0) = \frac{1+1}{9+6} = \frac{2}{15}$ $\quad P(\text{امتحان} \mid label=1) = \frac{2+1}{8+6} = \frac{3}{14}$

$P(\text{سیب} \mid label=0) = \frac{1+1}{9+6} = \frac{2}{15}$ $\quad P(\text{سیب} \mid label=1) = \frac{2+1}{8+6} = \frac{3}{14}$

$P(\text{نشی} \mid label=0) = \frac{0+1}{9+6} = \boxed{\frac{1}{15}}$ $\quad P(\text{نشی} \mid label=1) = \frac{0+1}{8+6} = \boxed{\frac{1}{14}}$

$\xrightarrow[\text{تست}]{\text{داده}}$ $P(label=0, \text{شادی, علی, زمستان, امتحان, نشی}) = \frac{1}{2} \times \frac{1}{15} \times \frac{2}{15} \times \frac{1}{5} \times \frac{1}{5} \times \frac{1}{5}$

$$= \boxed{\frac{1}{(15)^2 \times 5^3}} = \frac{1}{28125}$$

$\xrightarrow[\text{تست}]{\text{داده}}$ $P(label=1, \text{شادی, زمستان, علی, امتحان, نشی}) = \frac{1}{2} \times \frac{1}{14} \times \frac{3}{14} \times \frac{1}{7} \times \frac{1}{7} \times \frac{1}{7}$

$$= \boxed{\frac{3}{2 \times (14)^2 \times 7^3}} = \boxed{\frac{3}{134456}}$$

$\frac{1}{28125} \bigcirc \frac{3}{134456} \Rightarrow 134456 \bigcirc 3 \times 28125$

84375

با توجه به اینکه $P(label=0, \text{شادی, علی, زمستان, امتحان, نشی})$ بزرگتر است پس با توجه به احتمالات پس برچسب

داده تست در کلاس 0 قرار می‌گیرد

سوال دوم

نوتبوک ها اجرا شدند.

سوال سوم

سوال ③:

likelihood

$$L(\theta) = \prod P(y_i|x_i)^{y_i} \times \left(1 - P(y_i|x_i)\right)^{(1-y_i)}$$

که x_i بردارهای ما ، y_i برچسب یا همان label های L هستند

$$\log L(\theta) = \sum y_i \log P(y_i|x_i) + (1-y_i)\log\left(1 - P(y_i|x_i)\right)$$

حالا از تابع probit استفاده می‌کنیم که آن چیست

$$P(y_i|x_i) = \Phi(x_i)$$

که آن چیست log - برابر است با:

$$-\log L(\theta) = \sum -y_i \log(\Phi(x_i)) - (1-y_i)\log(1-\Phi(x_i))$$

که در متن زیر $\Phi(a) = \int_{-\infty}^{a} N(\theta|0,1)\,d\theta$ جایگزین می‌کنیم که آن چیست

$$-\log L(\theta) = \sum -y_i \log\left(\int_{-\infty}^{x_i} N(\theta|0,1)\,d\theta\right) - (1-y_i)\log\left(1 - \int_{-\infty}^{x_i} N(\theta|0,1)\,d\theta\right)$$

سوال چهارم

الف) مهم ترین دلیل آن جلوگیری از خطی شدن شبکه هست دلایل دیگری نیز دارد که به شرح زیر است:

۱. غیرخطی بودن: توابع فعال‌سازی غیرخطی هستند، به این معنی که خروجی شبکه به طور غیرخطی با ورودی‌ها مرتبط است. این غیرخطیت اجازه می‌دهد تا شبکه قادر به مدل‌سازی الگوها و روابط پیچیده‌تری باشد که در صورت استفاده از توابع خطی میسر نبوده است.

۲. تقسیم‌پذیری: استفاده از توابع فعال‌سازی به شبکه‌های MLP امکان تقسیم‌پذیری (separability) بین داده‌ها را می‌دهد. این به معنی این است که شبکه قادر است بین داده‌های مختلف در فضای ویژگی، مرزهای تصمیم‌گیری متفاوتی ایجاد کند و الگوهای مختلف را به خوبی تمیز دهد.

۳. آموزش موثرتر: توابع فعال‌سازی بهبود آموزش شبکه را تسهیل می‌کنند. به طور کلی، توابع فعال‌سازی بهبود پایداری آموزش، جلوگیری از دچار شدن شبکه در مشکل برهم‌کنش بین لایه‌ها (vanishing/exploding gradients)، و به افزایش سرعت همگرایی شبکه کمک می‌کنند.

۴. تعامل با توزیع‌های احتمالاتی: توابع فعال‌سازی می‌توانند با توزیع‌های احتمالاتی مرتبط با ورودی‌ها تعامل کنند. مثلاً، تابع سیگموئید به طور معمول در مسائل دسته‌بندی برای تبدیل خروجی شبکه به احتمالات کلاس‌ها استفاده می‌شود.

ب) تابع غیرخطی که استفاده می کنیم بسیار بستگی به نوع مسئله ما دارد در تئوری بله میشه از هر تابع غیر خطی استفاده کرد ولی در عمل اینکه کی به جواب همگرا بشه یا مقیاس خروجی،نقطه شروع یا عوامل دیگر نوع تابع خطی بسیار مهم میکند چون مسئله به مسئله فرق دارد.

۱. زمینه مسئله: توابع فعال‌سازی باید با زمینه مسئله سازگار باشند. به عنوان مثال، در مسائل دسته‌بندی دودویی، استفاده از تابع فعال‌سازی سیگموئید مناسب است زیرا خروجی آن بین ۰ و ۱ قرار می‌گیرد و می‌تواند به عنوان احتمال یک کلاس مورد استفاده قرار گیرد. اما در مسائل رگرسیون، ممکن است تابع فعال‌سازی دیگری مثل تابع خطی (identity) مناسب‌تر باشد.

۲. مقیاس خروجی: تابع فعال‌سازی انتخاب شده باید با مقیاس خروجی مورد نظرتان سازگار باشد. برای مثال، اگر مقیاس خروجی بین ۰ و ۱ باشد، استفاده از تابع فعال‌سازی سیگموئید یا تانژانت هایپربولیک مناسب است. اما اگر مقیاس خروجی بین -۱ و ۱ باشد، ممکن است تابع هایپربولیک تانژانت یا تابع ReLUمناسب‌تر باشند.

۳. نقطه شروع: برخی توابع فعال‌سازی نیاز به تعریف نقطه شروع (bias) دارند. برای مثال، تابع ReLU نقطه شروع صفر دارد و برای ورودی‌های منفی صفر خروجی می‌دهد. در صورت استفاده از تابع فعال‌سازی دیگری، نیاز است تا نقطه شروع مناسب برای آن تعیین شود.

در نهایت، مهم است که اثرات تابع فعال‌سازی جدید را در شبکه‌های MLP آزمایش کنید و عملکرد آن را با توابع فعال‌سازی معمول مقایسه کنید.

سوال پنجم

الف)

تابع sigmoid :این تابع خروجی اش بین ۰و۱ هست و به صورت زیر تعریف میشود:

$$S(x) = \frac{1}{1 + e^{-x}}$$

مزایا:

- نقطه میانگین غیرخطی: تابع سیگموئید، با توجه به مقدار خروجی‌اش که بین ۰ و ۱ است، به صورت غیرخطی عمل می‌کند. این خاصیت می‌تواند به شبکه‌های عصبی کمک کند تا الگوها و روابط غیرخطی را بهتر مدل کنند.
- مشتق‌پذیری: تابع سیگموئید در همه نقاط قابل مشتق است، که میتواند در فرآیند آموزش شبکه مورد استفاده قرار گیرد.

معایب:

- خروجی محدود: خروجی تابع سیگموئید بین ۰ و ۱ محدود است. این محدودیت می‌تواند در مواردی که مقیاس خروجی دیگری مورد نیاز است، محدودیت‌هایی ایجاد کند.
- مشکل ناپدید شدن گرادیان: در شبکه‌های عمیق با استفاده از تابع سیگموئید، ممکن است مشکل ناپدید شدن گرادیان (vanishing gradient) به وجود آید که باعث کاهش سرعت آموزش و توانایی شبکه در یادگیری الگوهای پیچیده می‌شود.

تابع softmax :

تابع softmax یک تابع فعال‌سازی غیرخطی است که برای تبدیل یک بردار ورودی به یک بردار احتمالاتی با مقادیر مثبت استفاده می‌شود. این تابع معمولاً در انتهای یک شبکه عصبی که باید احتمالات خروجی را محاسبه کند، استفاده می‌شود. رابطه تابع softmax به صورت زیر است:

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

مزایا:

- تولید بردار احتمالاتی: تابعsoftmax ، ورودی را به یک بردار احتمالاتی تبدیل می‌کند که مقادیر آن مثبت و مجموع آن برابر با ۱ است. این خاصیت می‌تواند در مسائل دسته‌بندی چندکلاسه و تشخیص الگوها کمک کند.
- مشتق‌پذیری: تابع softmax در همه نقاط قابل مشتق است، که میتواند در فرآیند آموزش شبکه مورد استفاده قرار گیرد.

معایب:

- آسان برای اشباع شدن: تابع softmax در صورتی که یکی از عناصر ورودی به آن بسیار بزرگ باشد، می‌تواند به سرعت به سمت مشکل اشباع شدن (saturation) بروید. این مشکل می‌تواند باعث کاهش تفاوت بین احتمالات خروجی شبکه شود و توانایی تمایز بین دسته‌ها را کاهش دهد.

تابع Relu :

ReLUیا Rectified Linear Unit یک تابع غیرخطی است که برای ورودی‌های مثبت خطی عمل می‌کند و برای ورودی‌های منفی صفر می‌شود. رابطه تابع ReLU به صورت زیر است:

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

مزایا:

- سادگی و سرعت محاسباتی: تابع ReLU بسیار ساده است و محاسبات آن نیز بسیار سریع انجام می‌شود.
- حل مشکل ناپدید شدن گرادیان: تابع ReLU از مشکل ناپدید شدن گرادیان در آموزش شبکه‌های عمیق جلوگیری می‌کند. این تابع در حالتی که ورودی مثبت است، گرادیان را به صورت ثابت انتقال می‌دهد و این باعث می‌شود شبکه بتواند به طور مؤثری الگوها را یاد بگیرد.

معایب:

- خروجی منفی: تابع ReLU برای ورودی‌های منفی صفر خروجی می‌دهد ممکن است اطلاعاتی مفید درباره ورودی‌های منفی را از دست بدهیم.
- مشکل ازاد شدن (Dying ReLU) : یک مشکل ممکن در استفاده از تابعReLU ، مشکل ازاد شدن نورون‌ها است. این مشکل در صورتی رخ می‌دهد که وزن‌ها به گونه‌ای تنظیم شوند که ورودی نگاتیوی

به نورون‌ها برسد و خروجی آنها همیشه صفر باشد. در این صورت، گرادیان برای آپدیت وزن‌ها در مرحله به‌روزرسانی شبکه صفر می‌شود و این نورون‌ها دیگر یادگیری نمی‌کنند.

تابع tanh :

تابع tanh یک تابع فعال‌سازی غیرخطی است که ورودی را به مقادیری بین -۱ و ۱ می‌برد. این تابع از رابطه زیر تعریف می‌شود:

$$f(x) = \frac{\left(e^{x} - e^{-x}\right)}{\left(e^{x} + e^{-x}\right)}$$

مزایا:

- تولید خروجی با مقادیر منفی و مثبت: تابع tanh ورودی را به مقادیری بین -۱ و ۱ تبدیل می‌کند که شامل اعداد مثبت و منفی است. این خاصیت می‌تواند در مسائلی که نیاز به خروجی با مقادیر منفی و مثبت استفاده شود، مفید باشد.
- مشتق‌پذیری: تابع tanh در همه نقاط قابل مشتق است، که میتواند در فرآیند آموزش شبکه مورد استفاده قرار گیرد.

معایب:

- خروجی محدود: مانند تابع سیگموئید، خروجی تابع tanh نیز محدود به مقادیری بین -۱ و ۱ میشود.

مقایسه:

- تابع softmax برای مسائل دسته‌بندی چند دسته‌ای و تولید توزیع احتمالاتی از خروجی‌ها استفاده می‌شود.
- تابع tanh می‌تواند در مسائلی که مقادیر خروجی به مقادیر منفی و مثبت نیاز دارند مفید باشد.
- تابع sigmoid نیز برای مسائل دسته‌بندی دودویی و تبدیل ورودی به مقادیر احتمالاتی استفاده می‌شود.

- تابع ReLU به عنوان تابع فعال‌سازی پر کاربرد است و معمولاً در شبکه‌های عصبی عمیق استفاده می‌شود. این تابع به خوبی در آموزش شبکه و جلوگیری از ناپدید شدن گرادیان مورد استفاده قرار می‌گیرد.

ب)این قسمت در نوتبوک مطابق فرمول هایی که در بالا ذکر شد پیاده سازی شده است و قابل مشاهده است.

ج)

۱. **تعداد لایه‌ها و علت انتخاب تعداد :** من سه لایه قرار دارم علت این سه لایه با توجه به پیچیدگی مسئله هست که تقریبا مسئله اسانی هست پس نیاز نیست از یک شبکه با تعداد لایه زیاد استفاده بکنیم.

۲. **تعداد نورون های هر لایه و علت انتخاب این تعداد:**

لایه ورودی:من تصویر را به ۱۰ در ۱۰ ریسایز کردم به همین علت تعداد ورودی من یک لایه صدتایی هست

لایه مخفی: تعداد نورون‌ها در این لایه می‌تواند متغیر باشد و به عنوان یک پارامتر طراحی در نظر گرفته شود. به طور کلی، معمولاً افزایش تعداد نورون‌ها در لایه مخفی به افزایش قدرت مدل کمک می‌کند، اما باید مواظب برازش زیاد به داده‌های آموزشی باشیم تا از بروز بیش‌برازش (overfitting) جلوگیری شود منتعداد نورون ها در این لایه را برابر با ۶۴ گرفتم.

- لایه خروجی: تعداد نورون‌ها برابر با تعداد کلاس‌ها یا دسته‌هایی که می‌خواهیم تصاویر را جدا کنیم. در این حالت، ۳ نورون برای سه کلاس "ب"، "ک" و "ص" مناسب است.

۳. تابع فعال‌سازی و علت انتخاب آن:

- برای لایه‌های مخفی، می‌توان از تابع فعال‌سازی ReLU استفاده کرد. این تابع به صورت غیرخطی عمل می‌کند و می‌تواند به طور موثر اطلاعات غیرخطی را در داده‌ها نمایش دهد.
- برای لایه خروجی، می‌توان از تابع فعال‌سازی softmax استفاده کرد. این تابع برای تبدیل خروجی شبکه به توزیع احتمالاتی استفاده می‌شود، به این صورت که مقادیر خروجی را به فضای احتمال تبدیل می‌کند و مقدار احتمال هر دسته را مشخص می‌کند.

۴. تابع ضرر و علت انتخاب آن:

- برای مسئله دسته‌بندی چند دسته‌ای می‌توان از تابع ضرر "Cross Entropy" استفاده کرد. این تابع مناسب است زیرا به طور مستقیم با احتمالات خروجی شبکه و برچسب‌های واقعی مقایسه می‌کند و میزان مطابقت بین آن دو را اندازه‌گیری می‌کند.

د)شبکه ای که در بالا گفتم را در پایتورچ پیاده سازی کردم در ابتدا مدل MLP را تعریف کردم که کد ان به صورت زیر است:

```
class MLP(nn.Module):
  def __init__(self,input_size,hidden_size,output_size):
    super(MLP,self).__init__()
    self.fc1=nn.Linear(input_size,hidden_size)
    self.relu=nn.ReLU()
    self.fc2=nn.Linear(hidden_size,output_size)
    self.softmax=nn.Softmax(dim=1)


  def forward(self,x):
    x=self.fc1(x)
    x=self.relu(x)
    x=self.fc2(x)
    x = self.softmax(x)
    return x
```

- self.fc1یک لایه خطی (nn.Linear) است که ورودی‌ها به تعداد input_sizeرا به تعداد hidden_sizeخروجی تبدیل می‌کند.
- self.reluتابع فعال‌سازی ReLU است که برای فعال‌سازی خروجی لایه fc1استفاده می‌شود.
- self.fc2یک لایه خطی دیگر است که ورودی‌ها به تعداد hidden_sizeرا به تعداد output_sizeخروجی تبدیل می‌کند.
- self.softmaxتابع فعال‌سازی Softmax است که برای تولید احتمالات خروجی لایه fc2استفاده می‌شود.

سپس، در متد forward، جریان اطلاعات در شبکه تعریف شده است:

- ورودی xاز لایه fc1عبور می‌کند و با استفاده از تابع فعال‌سازی ReLU در self.reluفعال‌سازی می‌شود.
- خروجی این لایه به عنوان ورودی به لایه fc2داده می‌شود.

- در نهایت، با استفاده از تابع فعال‌سازی Softmax در self.softmax، احتمالات خروجی تولید می‌شوند.

سپس یک کلاس به نام *CustomDataset* تعریف کردم که دیتاست سه عکس و لیبل ها را اماده می کند سپس هایپرپارامترها را به صورت زیر تنظیم کردم:

```
# hyperparameter
input_size = 100
inputimg=10
hidden_size = 64
output_size = 3

learning_rate = 0.001
num_epochs = 4000
batch_size = 1
```

و بعد optimizer و تابع loss را تنظیم کردم optimizer را SGD و تابع loss را cross entropy گرفتم.و در ادامه در یک for عملیات forward و محاسبه lossو backward را انجام دادم که کد ان به صورت زیر است:

```
import matplotlib.pyplot as plt
import torchvision.transforms.functional as TF

for epoch in range(num_epochs):
  running_loss = 0.0
  for images, labels in dataset:
    images = images.view(-1, input_size)
    optimizer.zero_grad()
    outputs = model(images)
    outputs  = torch.squeeze(outputs)
    loss = criterion(outputs, labels)
    loss.backward()
    optimizer.step()
    running_loss += loss.item()
  epoch_loss = running_loss / len(dataset)
  print(f"Epoch {epoch+1}/{num_epochs}, Loss: {epoch_loss}")
```

میزان ضرر ما به صورت کاهشی بود که در نوتبوک Q5 نیز موجود است:

```
Epoch 16/4000, Loss: 1.1033800840377808
Epoch 17/4000, Loss: 1.1033018827438354
Epoch 18/4000, Loss: 1.1032267014185588
Epoch 19/4000, Loss: 1.1031514803568523
Epoch 20/4000, Loss: 1.1030847628911336
Epoch 21/4000, Loss: 1.1030079921086628
Epoch 22/4000, Loss: 1.1029271682103474
Epoch 23/4000, Loss: 1.102854569753011
Epoch 24/4000, Loss: 1.1027893622716267
Epoch 25/4000, Loss: 1.1027168035507202
...
Epoch 3997/4000, Loss: 0.6195782423019409
Epoch 3998/4000, Loss: 0.6194448669751486
Epoch 3999/4000, Loss: 0.6194724639256796
Epoch 4000/4000, Loss: 0.619433065255483
```

و بعد از ان نیز سه تصویر را به مدل دادیم و هر سه را نیز درست تشخیص داد که میتوانید در نوتبوک مشاهده کنید.

سوال ششم

با توجه به ترکیب توابع فعال‌ساز و آستانه ۰.۵ در شبکه عصبی برای دسته‌بندی دو کلاسه، ممکن است مشکلات یا چالش‌های زیر وجود داشته باشد:

۱. اشتباه در تصمیم‌گیری: با استفاده از تابع فعال‌ساز سیگموئید و آستانه ۰.۵، خروجی‌های بزرگتر یا مساوی ۰.۵ به عنوان کلاس ۱ در نظر گرفته می‌شوند و خروجی‌های کمتر از ۰.۵ به عنوان کلاس ۰ دسته‌بندی می‌شوند. اما این تصمیم‌گیری ممکن است منطقی نباشد و با داده‌های واقعی همخوانی نداشته باشد. در برخی موارد، توزیع داده‌ها و نحوه تفکیک کلاس‌ها ممکن است به گونه‌ای باشد که استفاده از آستانه ۰.۵ و تابع فعال‌ساز سیگموئید باعث ایجاد خطاها و نتایج نادرست شودمثلا بعضی اوقات از معیار ROC استفاده می کنند.

۲. مشکل اشباع سیگموئید: تابع فعال‌ساز سیگموئید در مقادیر بسیار بزرگ یا بسیار کوچک به سمت ۰ یا ۱ اشباع می‌شود. این اشباع می‌تواند منجر به کاهش گرادیان در فرآیند آموزش شود و باعث کاهش سرعت یادگیری شود.

۳. یک مشکل دیگه اینکه خروجی relu برای اعداد منفی همیشه ۰ هست و بعد از sigmoid در تابع مورد نظر مقدار یک میشود این یعنی همیشه وقتی ورودی نورون پایانی منفی باشه مدل کلاس را یک پیش بینی میکند ان هم با احتمال ۱۰۰ درصد .

سوال هفتم

الف)

۱. ساختار شبکه عصبی: یادگیری ماشین معمولاً بر اساس الگوریتم‌های سنتی مانند درخت تصمیم، ماشین بردار پشتیبان و رگرسیون لجستیک صورت می‌گیرد. این الگوریتم‌ها عموماً بر پایه ویژگی‌های دستی، انتخاب شده و یا استخراج شده از داده‌ها عمل می‌کنند. در مقابل، در یادگیری عمیق، شبکه‌های عصبی چندلایه با تعداد زیادی لایه مخفی استفاده می‌شوند که بتوانند خود ویژگی‌های مورد نیاز را از داده‌ها استخراج کنند. این قابلیت استخراج ویژگی‌های خودکار شبکه‌های عمیق باعث افزایش قدرت و عمق یادگیری می‌شود.

۲. نیاز به حجم بالای داده: یادگیری عمیق، به عنوان یک زیرمجموعه از یادگیری ماشین، بیشتر به داده‌های بزرگ و متنوع نیاز دارد تا بتواند ویژگی‌های مناسب را استخراج کند و مدل‌های کارآمدی را ساختاردهی کند. در حالی که در یادگیری ماشین، ممکن است با استفاده از حجم کمتری از داده‌ها، مدل‌های قابل قبولی بسازید.

۳. پیچیدگی مدل: شبکه‌های عمیق دارای معماری پیچیده‌تری هستند و تعداد پارامترهای قابل آموزش در آنها بسیار بیشتر است. این پیچیدگی بیشتر می‌تواند منجر به عملکرد بهتر در مسائل پیچیده‌تر شود، اما همچنین نیازمند توانایی محاسباتی بیشتر و مجموعه داده‌های بزرگتر است.

۴. نیاز به منابع محاسباتی: به دلیل پیچیدگی بیشتر شبکه‌های عمیق، نیاز به منابع محاسباتی بیشتری نسبت به روش‌های سنتی یادگیری ماشین دارند. آموزش و آزمون شبکه‌های عمیق ممکن است نیازمند استفاده از سخت‌افزارهای گرافیکی قوی (GPU) و یا منابع محاسباتی توزیع شده باشد.

از طرفی، هر دو روش یادگیری ماشین و یادگیری عمیق قابلیت‌ها و کاربردهای خود را دارند و بسته به مسئله و داده‌های مورد استفاده، مناسبی می‌توانند باشند. اما یادگیری عمیق به دلیل قدرت بالایش در استخراج ویژگی‌ها و پردازش داده‌های بزرگ، در بسیاری از مسائل پیچیده مانند تشخیص تصویر، ترجمه ماشینی، تشخیص گفتار و غیره، عملکرد بهتری از خود نشان می‌دهد.

ب)

پاسخ این سوال مربوط به نوع مسئله و شبکه استفاده شده است ولی معمولا لایه های ابتدایی ویژگی های abstract را در میارن و لایه های انتهایی ویژگی های سطح بالاتر را به دست می اورند مثلا اگر ورودی یک تصویر در نظر بگیریم لایه های ابتدایی ویژگی هایی همچون لبه در میارن ولی لایه انتهایی ویژگی هایی در میارن مثلا اینجا لب هست یا گوش هست پس به طوری کلی میتوان گفت لایه ۱۱ برای طبقه بندی مناسب تر از لایه ۷ هست.

ج)

در کل، استفاده از شبکه‌های عمیق تر برای تقریب توابع معمولاً به دلیل ویژگی‌های زیر از شبکه‌های عریض تر پیشنهاد می‌شود:

۱. قدرت انعطاف‌پذیری: شبکه‌های عمیق تر قادرند ویژگی‌های پیچیده‌تر و ساختارهای عمیق‌تر را در داده‌ها استخراج کنند. این انعطاف‌پذیری بیشتر می‌تواند منجر به تقریب دقیق‌تر توابع مورد نظر شود.

۲. تعمیم‌پذیری به داده‌های ناشناخته: شبکه‌های عمیق توانایی تعمیم‌پذیری به داده‌هایی که در مرحله آموزش موجود نبوده‌اند را دارند. این به این معناست که می‌توانند بر روی داده‌های جدید و ناشناخته به خوبی عمل کنند و توابع را به خوبی تقریب بزنند.

۳. جلوگیری از بیش‌برازش :(Overfitting) استفاده از شبکه‌های عمیق باعث کاهش احتمال بیش‌برازش می‌شود. شبکه‌های عمیق توانایی یادگیری ویژگی‌های عمومی و جامع را دارند که برای تقریب توابع بهتر استفاده می‌شوند و از تقریب توابع خاص و ویژگی‌های محدود جلوگیری می‌کنند.

د)

۱. قدرت نمایش قابلیت‌های پیچیده‌تر: با افزودن لایه‌های بیشتر، شبکه عصبی قادر به نمایش قابلیت‌های پیچیده‌تری می‌شود. این به معنای قدرت بیشتر مدل در یادگیری و تشخیص الگوهای پیچیده است.

۲. افزایش دقت و عملکرد: با افزودن لایه‌های بیشتر، میزان دقت و عملکرد شبکه عصبی می‌تواند بهبود یابد. این امر به دلیل توانایی شبکه در یادگیری و استخراج ویژگی‌های بیشتر از داده‌ها و تجزیه و تحلیل بهتر اطلاعات است.

۳. انعطاف‌پذیری بیشتر: افزودن لایه‌های بیشتر به شبکه عصبی، امکان انعطاف‌پذیری در معماری شبکه را افزایش می‌دهد. می‌توان با افزودن لایه‌های مختلف و ترکیب آن‌ها، معماری‌های متنوعتری برای شبکه طراحی کرد و بهبودهای مختلف را اعمال کرد.

۴. انتقال یادگیری (Transfer Learning) :با افزودن لایه‌های بیشتر، می‌توان از شبکه عصبی پیش‌آموزش دیده شده (pre-trained) در یک وظیفه مشابه استفاده کرده و آن را به وظیفه دیگری انتقال داد. این به معنای استفاده از یادگیری قبلی شبکه برای حل مسئله جدید است که به داده‌های کمتری نیاز دارد و زمان و هزینه آموزش را کاهش می‌دهد.

معایب:

۱. بیش‌برازش :(Overfitting) با افزایش تعداد لایه‌ها، احتمال بیش‌برازش در شبکه عصبی افزایش می‌یابد. بیش‌برازش به معنای بیش‌برازش به داده‌های آموزش و کاهش تعمیم‌پذیری شبکه است. این مشکل می‌تواند با استفاده از روش‌هایی مانند Dropout و Regularization مدیریت شود.

۲. پیچیدگی محاسباتی: با افزودن لایه‌های بیشتر، پیچیدگی محاسباتی شبکه نیز افزایش می‌یابد. این می‌تواند به معنای زمان و منابع مورد نیازبرای محاسبات بیشتر و نیاز به سخت‌افزار قدرتمندتر باشد.

۳. مشکل گرادیان محو :در شبکه‌های عصبی عمیق، گرادیان ممکن است در طول عملیات پس‌انتشار (backpropagation)محو شوند. این به معنای این است که گرادیان‌ها به لایه‌های اولیه یا نهایی شبکه نمی‌رسند و باعث کاهش سرعت و کیفیت یادگیری می‌شود. برخی روش‌های بهبود گرادیان مانند استفاده از تابع فعال‌سازی متفاوت، تکنیک‌های نرمال‌سازی و استفاده از شبکه‌های بازگشتی می‌توانند این مشکل را حل کنند.

۴. هزینه محاسباتی بالا: با افزودن لایه‌های بیشتر، هزینه محاسباتی شبکه نیز افزایش می‌یابد. آموزش و استنتاج شبکه‌های عصبی عمیق با تعداد بالای لایه‌ها ممکن است زمان‌بر و محاسباتی پرهزینه باشد.

به طور کلی، افزودن لایه‌های بیشتر به شبکه عصبی عمیق می‌تواند باعث افزایش قدرت و دقت مدل شود، اما همچنین ممکن است با مشکلاتی همچون بیش‌برازش و محاسبات پیچیده همراه باشد. برای طراحی و آموزش

شبکه عصبی، باید توجه شود که بر اساس وظیفه و داده‌های مورد استفاده، تعداد و ساختار لایه‌ها بهینه‌سازی شود.