TCC

April 6, 2021

1 Análise de dados e construção de modelo de machine learning para análise de fake news

- importando libs para análise de dados

[1]:
```
import pandas as pd
import numpy as np
from unidecode import unidecode
import re
import collections
import nltk
import time
import warnings
warnings.filterwarnings("ignore", category=FutureWarning)
```

1.1 Importando o arquivo com todas as notícias

[2]:
```
todas_noticias = pd.read_csv('datasets/todos_os_dados.csv')
todas_noticias.target.unique()
```

[2]:
```
array(['falso', 'verdadeiro'], dtype=object)
```

1.2 Limpeza dos dados (Feature engineering)

- Para esta análise de dados não precisaremos do link da imagem nem da origem da notícia
- Vamos trabalhar apenas com a coluna `info` e a coluna `target`

[3]:
```
todas_noticias = todas_noticias[['info', 'target']]
todas_noticias.sample(5)
```

[3]:

	info	target
3527	Crise econômica: Empresário que demitiu 223 fu…	falso
7896	\nNota antigas de US$ 100 serão recolhidas\n	falso
3140	Alerta da OMS: " Situação é gravíssima. Quatro…	falso
1454	Camarotti: "Com a delação da Odebrecht, terem…	falso

```
3092  Dois caças da Marinha se colidem e um deles ca…  falso
```

Renomeando: Serão renomeadas o nome das colunas para português, estamos trabalhando com dados em português e o código também segue o padrão português, também para facilitar o entendimento de quem irá consumir este notebook

[4]:
```
colunas = {'info': 'texto', 'target': 'classificacao'}
todas_noticias.rename(columns=colunas, inplace=True)
```

1.2.1 Total de notícias verdadeiras:

[5]:
```
total_verdadeiras = len(todas_noticias[todas_noticias.classificacao ==␣
↪'verdadeiro'])
total_verdadeiras
```

[5]: 5014

1.2.2 Total de notícias falsas

[6]:
```
total_falsas = len(todas_noticias[todas_noticias.classificacao == 'falso'])
total_falsas
```

[6]: 7497

1.2.3 Total de notícias

[7]:
```
total = len(todas_noticias)
total
```

[7]: 12511

- Verificar se existem dados vazios ou inválidos nas colunas info e target

A função assert joga um erro se encontrar algum dado

[8]:
```
num_texto_null = len(todas_noticias[todas_noticias['texto'].isnull()])
num_texto_vazio = len(todas_noticias[todas_noticias['texto'].isna()])
assert  num_texto_null + num_texto_vazio == 0
```

[9]:
```
num_classificacao_null = len(todas_noticias[todas_noticias['classificacao'].
↪isnull()])
num_classificacao_vazio = len(todas_noticias[todas_noticias['classificacao'].
↪isna()])
assert num_classificacao_vazio + num_classificacao_null == 0
```

1.2.4 Normalizando os dados

- Passar todos os dados para lowercase para que não haja distinção entre maíusculo e minúsculo
- Remover acentuação, aspas e possíveis caracteres especiais

[10]:
```
todas_noticias.texto = todas_noticias.texto.map(lambda x: x.lower())
```

1.2.5 Removendo acentuação

- usando a lib `unidecode` remove a acentuação das palavras

[11]:
```
todas_noticias.texto = todas_noticias.texto.map(lambda x: unidecode(x))
```

1.2.6 Removendo caracteres especiais

- Expressões regulares podem ser usadas para remover caracteres especiais

[12]:
```
todas_noticias.texto = todas_noticias.texto.map(lambda x: re.sub('[\W_0-9]+', '␣
↪', x))
```

1.3 Contando as palavaras

- Montando um contador geral
- Montando um contador de palavras comuns para fake news
- Montando um contador de palavras comuns para notícias verdadeiras

[13]:
```
verdadeiro_count = collections.Counter()
falso_count = collections.Counter()
todas_count = collections.Counter()
```

1.4 Montando a contagem das palavras

- fazendo a contagem

[14]:
```
for index, row in todas_noticias.iterrows():
    noticia = row['texto'].split(' ')
    if row['classificacao'] == 'verdadeiro':
        verdadeiro_count.update(noticia)
    else:
        falso_count.update(noticia)
    todas_count.update(noticia)
```

1.4.1 Criando cópia dos contadores verdadeiros e falsos

- Vamos salvar uma cópia dos contadores de palavras para ambas as classes para podermos analisar se as stop words estão trazendo alguma vantagem ou desvantagem para o modelo

[15]:
```
verdadeiro_count_plot = verdadeiro_count.copy()
falso_count_plot = falso_count.copy()
```

1.5 Calculando a razão

- Com o cálculo da razão podemos ter a noção de quanto uma palavra ocorre no lado positivo em relação ao negativo.
- O calculo se dá por:

razao = ocorrencia_palavra_lado_positivo / (ocorrencia_da_palavra_lado_negativo +1)

- Palavras que ocorrem no muito mais no lado positivo do que no lado negativo tendem a apresentar uma razão com alto valor. Já palavras que ocorrem muito no lado negativo terão valor de razão bem pequeno. Já para stop words o que se espera é que o valor fique próximo de 1, visto que a quantidade de ocorrência da mesma pode ser próxima para ambos os lados.

[16]:
```
razao_falso_verdadeira = collections.Counter()
for word, count in list(todas_count.most_common()):
    if count > 10:
        razao = verdadeiro_count[word] / float(falso_count[word] + 1)
        razao_falso_verdadeira[word] = razao
```

1.5.1 Calculando log da razão

- Os valores de razão podem ficar muito dispersos, podendo palavras terem valores de razão muito distantes, desta maneira o cálculo do logarítimo vai aproximar os valores, facilitando a plotagem de distribuição
- Os valores de razão maiores devem ter um log variando mais próximo dos valores de 1 a 5
- Os valores mais próximos de 1 que pode ser onde as stop words, ficariam com 0 (pois o log de 1 é 0)
- E os valores muito pequenos de razão ficariam na ponta oposta, como valores negatívos entre -1 e -5
- Vale ressaltar que vamos manter as stopwords na distribuição para observar o comportamento

[17]:
```
np.seterr(divide = 'ignore')
razao_falso_verdadeira_log = {k: np.log(v) for k, v in razao_falso_verdadeira.
 ↪items()}
```

1.5.2 Obtendo a lista de stopwords a partir da biblioteca nltk

- A biblioteca nltk nos permite fazer o download de uma lista de palavras vazias (stop words) de várias linguas, no caso deste trabalho a lingua portuguesa.

[18]:
```
nltk.download('stopwords')
stopwords = nltk.corpus.stopwords.words('portuguese')
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     /home/willian/nltk_data…
[nltk_data]   Package stopwords is already up-to-date!
```

1.5.3 Removendo stopwords da contagem de palavras

- Vamos remover as stop words, da contagem das classes (lembrando que temos uma cópia destes dados)
- primeiramente serão padronizadas as stop words de acordo com o restante das outras palavaras

[19]:
```
palavras_comuns = [unidecode(palavra) for palavra in stopwords]
```

- Executando a remoção

[20]:
```
for palavra in palavras_comuns:
    if palavra in list(falso_count):
        del falso_count[palavra]
    if palavra in list(verdadeiro_count):
        del verdadeiro_count[palavra]
```

1.6 Criando o vocabulário

- O vocabulário é a lista de palavaras únicas que apareceram nas notícias verdadeiras e falsas.

[21]:
```
vocabulario = set(todas_count.keys())
len(vocabulario)
```

[21]:
```
82080
```

1.7 Visualização de dados

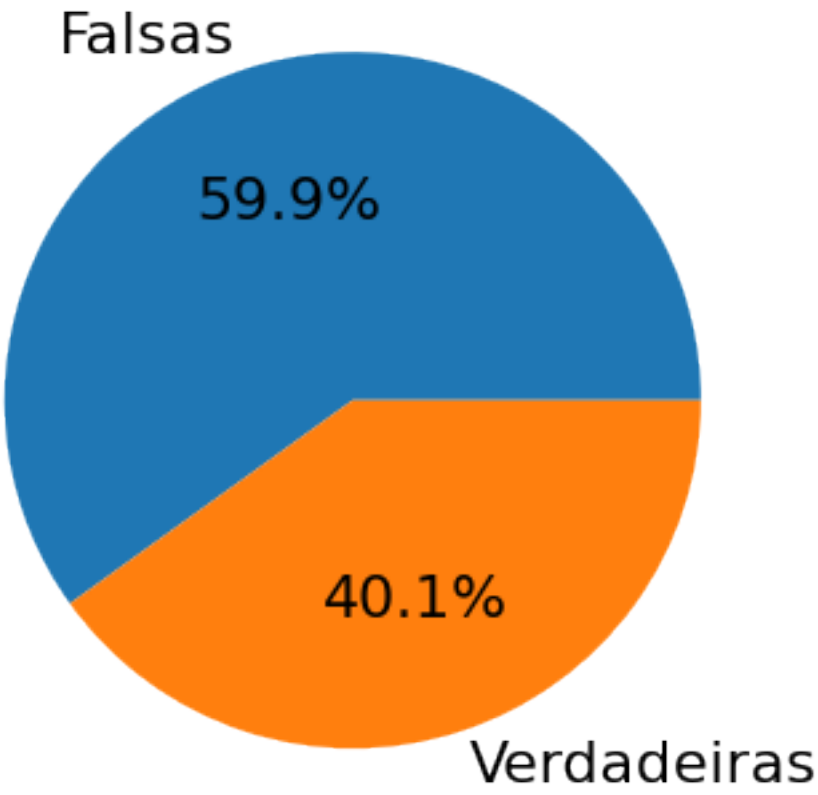
- Nesta parte faremos visualização dos dados e através das visualizações podemos formar hipóteses

[22]:
```
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

1.8 Proporção de notícias Falsas x Verdadeiras

- Qual a porcentagem que temos da relação entre notícias falsas e verdadeiras

[23]:
```
#Porcentagem falsa
plt.rcParams.update({'font.size': 22})
porcentagem_falsa = total_falsas / total * 100
porcentagem_verdadeira = total_verdadeiras / total * 100

fatias = [porcentagem_falsa, porcentagem_verdadeira]
labels = ['Falsas', 'Verdadeiras']
fig1 = plt.subplot()
ax = fig1.pie(fatias, labels=labels, autopct='%1.1f%%')
plt.gcf().set_size_inches(12,6)


plt.show()
```



1.8.1 Ponto de atenção:

- Temos uma proporção de quase 20% a mais para notícias falsas.

1.9 Estatísticas gerais do dataset

- vamos calcular a média e mediana para as notícias presentes no dataset

[24]:
```
## Media
falsas = todas_noticias[todas_noticias.classificacao == 'falso']
somatorio_falsas = falsas.texto.map(lambda x: len(x))
media_falsas = somatorio_falsas.mean()

## Media
verdadeiras = todas_noticias[todas_noticias.classificacao == 'verdadeiro']
somatorio_verdadeiras = verdadeiras.texto.map(lambda x: len(x))
media_verdadeiras = somatorio_verdadeiras.mean()

#Mediana
mediana_falsas = somatorio_falsas.median()

#Mediana
mediana_verdadeiras = somatorio_verdadeiras.median()

print(f'Média de tamanho (em cacateres) de notícias verdadeiras␣
 ↪{round(media_verdadeiras)}')
print(f'Média de tamanho (em cacateres) de notícias falsas␣
 ↪{round(media_falsas)}')
print(f'Mediana de tamanho (em caracteres) de notícias verdadeiras␣
 ↪{round(mediana_verdadeiras)}')
print(f'Mediana de tamanho (em caracteres) de notícias falsas␣
 ↪{round(mediana_falsas)}')
```

```
Média de tamanho (em cacateres) de notícias verdadeiras 4635
Média de tamanho (em cacateres) de notícias falsas 585
Mediana de tamanho (em caracteres) de notícias verdadeiras 4026
Mediana de tamanho (em caracteres) de notícias falsas 300
```

- Se observarmos as notícias verdadeiras tem maior tamanho, apesar de terem menor quantidade.
- Não é notada a presença de outliers modificando muito a média, visto que a o média e mediana estão bem próximas

1.10 Analisando ocorrência de palavras mais comuns

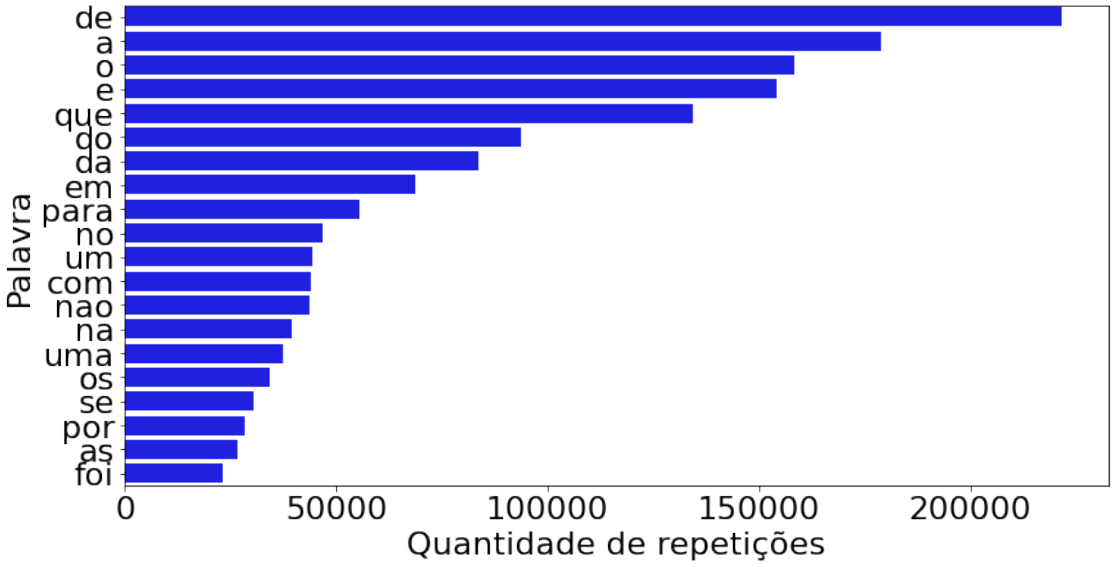
- Analisando as palavras mais comuns de forma geral

- Primeiramente para facilitar o trabalho criamos um dataset utilizando o resultado da função `most_common` provida pela lib `Counter` que já retorna o resultado do maior para o menor.

[25]:
```
df_most_commom = pd.DataFrame(todas_count.most_common(), columns=['word',␣
↪'count'])
```

- Agora vamos plotar um gráfico das palavras mais comuns em geral, temos o seguinte resultado:

[26]:
```
ax = sns.barplot(y='word', x='count', data=df_most_commom[:20],  color = 'blue')
plt.gcf().set_size_inches(12,6)
plt.rcParams.update({'font.size': 22})
ax.set(xlabel='Quantidade de repetições', ylabel='Palavra')
plt.show()
```



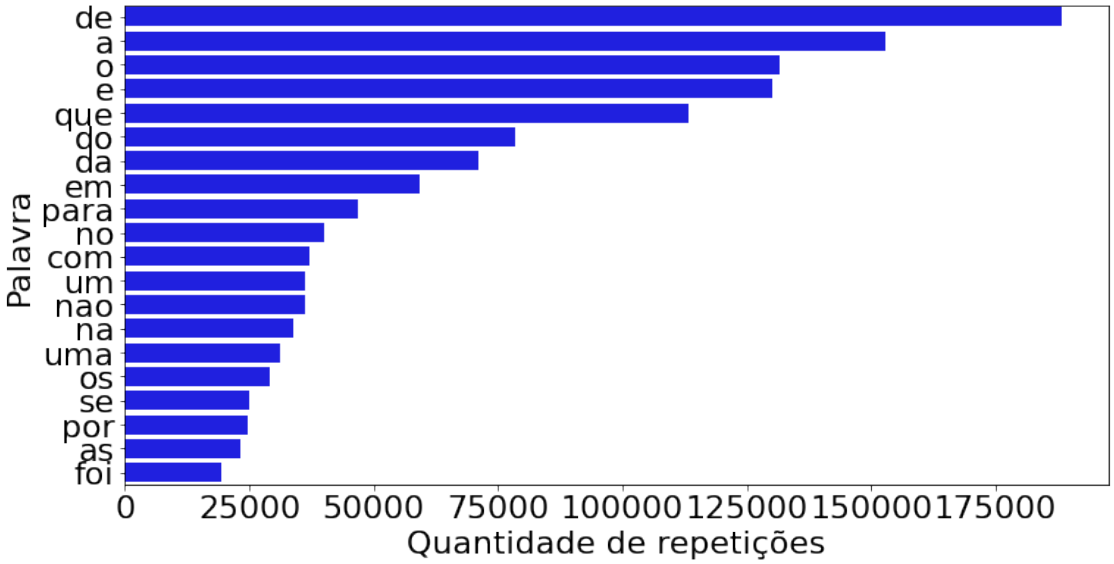
- Repare que o resultado são apenas palavras vazias (stop words)
- Elas fazem bastante volume no geral

1.10.1 Agora vamos observar por classe

- Vamos utilizar o cópia que temos da contagem de palavras em notícias, verdadeiras que ainda possuem a presença de stop words

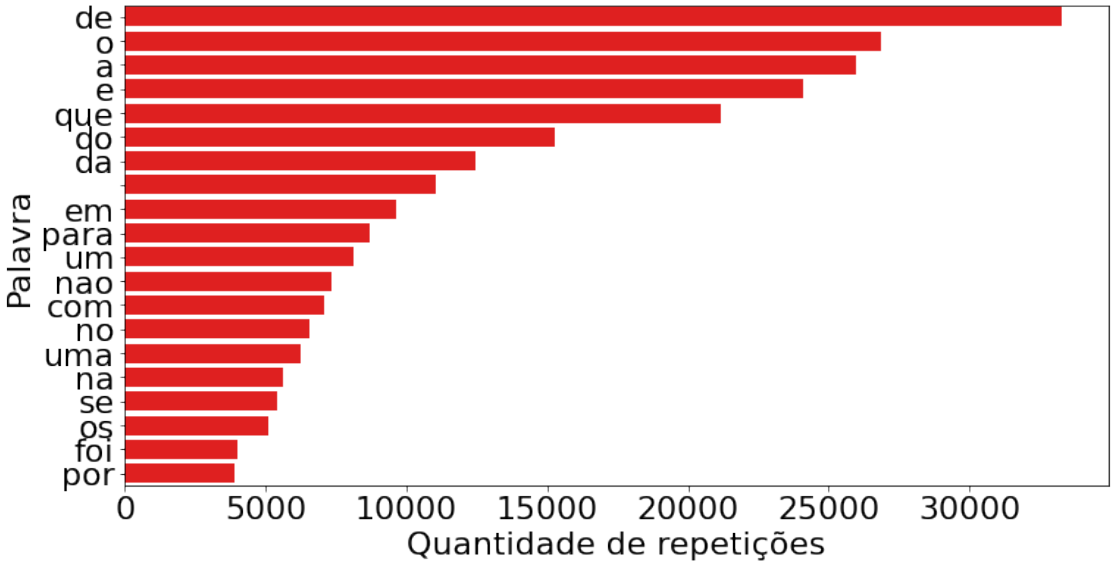
[27]:
```
df_verdadeiro_count = pd.DataFrame(verdadeiro_count_plot.
↪most_common(),columns=['word','count'])
ax = sns.barplot(y='word', x='count', data=df_verdadeiro_count[:20],  color =␣
↪'blue')
plt.gcf().set_size_inches(12,6)
plt.rcParams.update({'font.size': 22})
```

```
ax.set(xlabel='Quantidade de repetições', ylabel='Palavra')
plt.show()
```



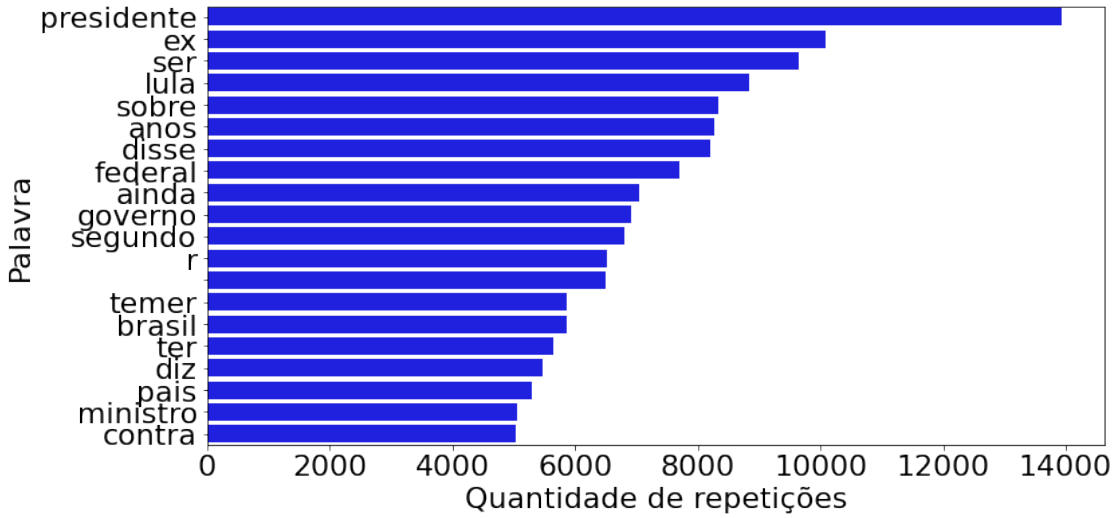
- Percebe-se que as stop words são predominantes no caso de notícias Verdadeiras
- Vamos observar se o mesmo acontece com notícias Falsas

[28]:
```
df_falso_count = pd.DataFrame(falso_count_plot.most_common(), columns=['word',␣
↪'count'])
ax = sns.barplot(y='word', x='count', data=df_falso_count[:20], color='red')
plt.gcf().set_size_inches(12,6)
ax.set(xlabel='Quantidade de repetições', ylabel='Palavra')
plt.rcParams.update({'font.size': 22})
```



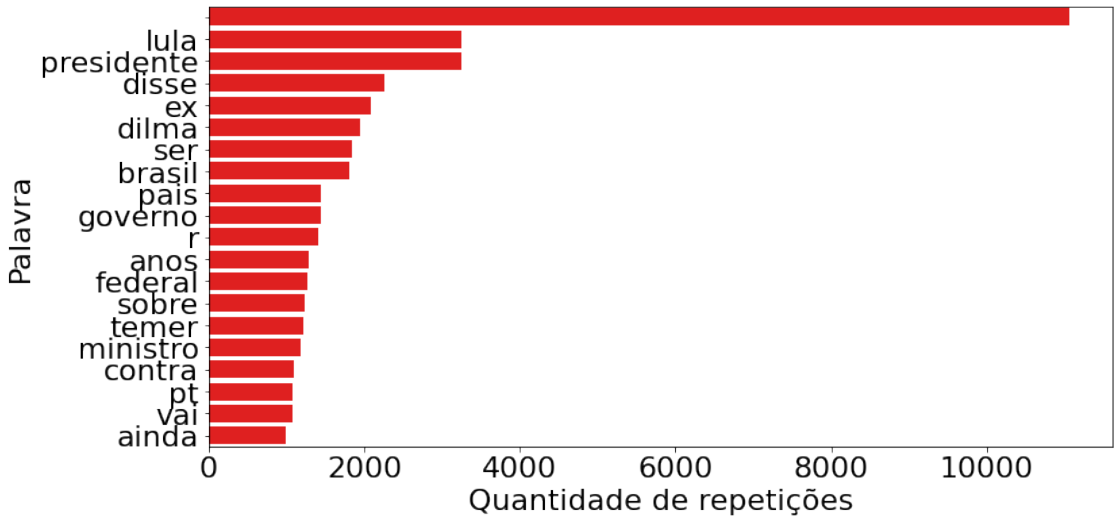
- Percebe-se que são praticamente as mesmas palavras dominam para ambas as classes
- Desta maneira fica difícil fazermos uma análise
- Para a análise seguinte vamos plotar os gráficos sem a presença das stop words, para analisarmos o comportamento
- Abaixo plotagem do mesmo gráfico agora sem a presença de stop words em notícias verdadeiras:

[29]:
```
df_verdadeiro_mc_whithout_sw = pd.DataFrame(verdadeiro_count.most_common(),␣
↪columns=['word', 'count'])
ax = sns.barplot(y='word', x='count', data=df_verdadeiro_mc_whithout_sw[:20],␣
↪color='blue')
plt.gcf().set_size_inches(12,6)
ax.set(xlabel='Quantidade de repetições', ylabel='Palavra')
plt.rcParams.update({'font.size': 22})
```



- Observe que agora as palavras que aparecem denotam um certo contexto para a notícia
- Agora vamos observar abaixo como fica o gráfico de notícias Falsas sem a presença de stop word:

[30]:
```
df_falso_mc_whithout_sw = pd.DataFrame(falso_count.most_common(),␣
↪columns=['word', 'count'])

ax = sns.barplot(y='word', x='count', data=df_falso_mc_whithout_sw[:20],␣
↪color='red')
plt.gcf().set_size_inches(12,6)
ax.set(xlabel='Quantidade de repetições', ylabel='Palavra')
plt.rcParams.update({'font.size': 22})
```



- Repare que também podemos ter um contexto do que se é mencionado nas notícias Falsas.
- Uma visualização muito comum em é a nuvem de palavras, é uma forma interessante de mostrar palavras que ocorrem bastante, mas que não cabem em um gráfico de barras.
- A nuvem de palavras da enfase nas palavras mais repetidas deixando-as maiores, vai reduzindo de tamanho as outras palavras que ocorreram bastante mas com menor intensidade.

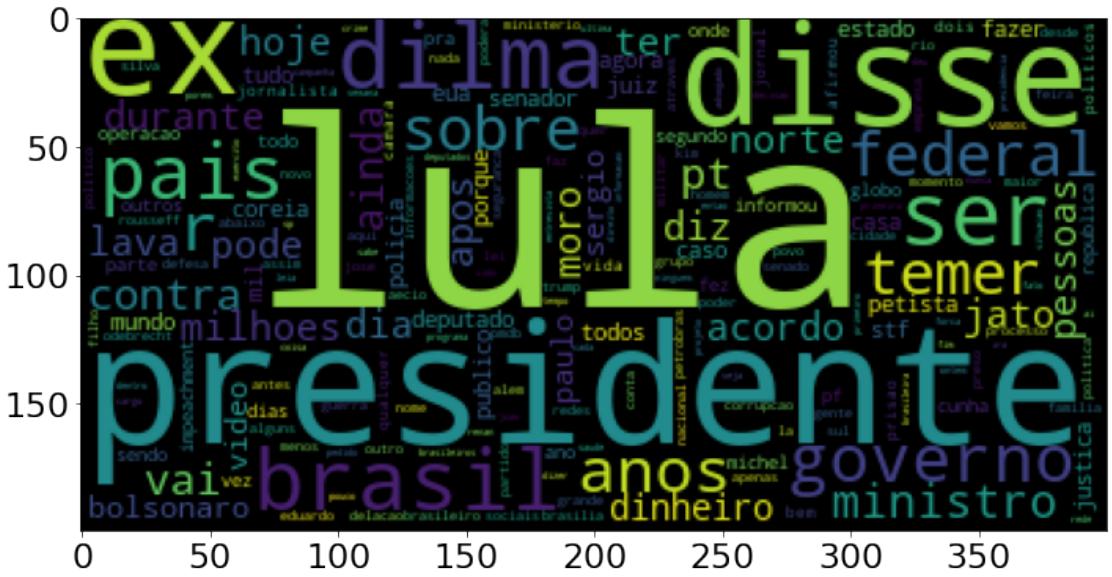
[31]:
```
from wordcloud import WordCloud, ImageColorGenerator
```

- Abaixo a nuvem de palavras para notícias verdadieras

[32]:
```
verdadeiro_cloud = WordCloud(stopwords=stopwords).
 ↪generate_from_frequencies(verdadeiro_count)

plt.imshow(verdadeiro_cloud)
plt.gcf().set_size_inches(12,6)
plt.rcParams.update({'font.size': 22})
```

- Abaixo a nuvem de palavras para notícias falsas.

[33]:
```
falso_count_cloud = WordCloud(stopwords=stopwords).
 ↪generate_from_frequencies(falso_count)

plt.imshow(falso_count_cloud)
plt.gcf().set_size_inches(12,6)
plt.rcParams.update({'font.size': 22})
```



- Agora que temos noção do contexto, podemos começar a avalidar a hipótese de que a ocorrência de uma determinadas palavras pode ter relação com a classificação da notícia.

1.11 Observando a densidade e concentração das palavras

- A partir de agora analisar a concentração das palavras
- Vamos utilizar os valores de razão calculados anteriormente neste documento.
- Testar a hipótese citada acima, verificar a tendência para cada classe.
- Vamos construir um histograma para avaliar a distribuição das palavras através da razão

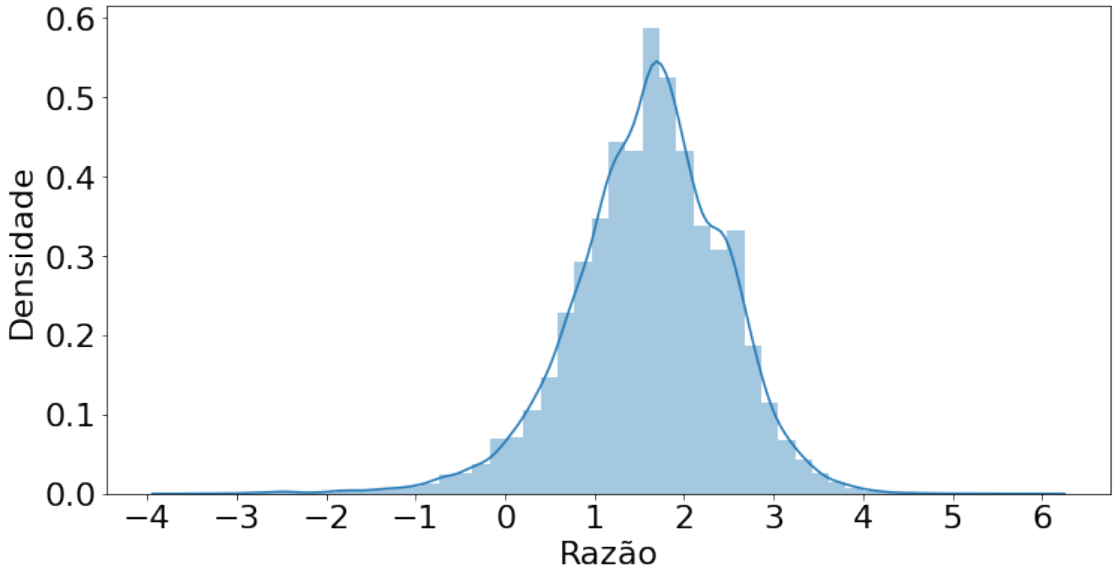
- Antes de criar a visualização vamos padronizar e organizar os valores
- Criaremos um DataFrame com os valores de razão.
- Renomearemos a coluna de `0` para razão, para ficar mais claro o dado que estamos trabalhando
- Substituiremos os valores infinitos para `not a number` e em seguidas fazemos a remoção dos mesmos.
- Convertemos todos os valores de texto para float64 para trabalhar com a plotagem dos valores.

[34]:
```
s  = pd.Series(razao_falso_verdadeira_log,index=razao_falso_verdadeira_log.
↪keys()).to_frame()
s.rename(columns = {0: 'razao'}, inplace=True)
s.head()

s.replace([np.inf, -np.inf], np.nan, inplace=True)
s.dropna(inplace=True)
s['razao'] = s['razao'].map(lambda x: np.float64(x))
```

- Agora vamos fazer a plotagem do histograma da razão

[35]:
```
raz = sns.distplot(s['razao'])
plt.gcf().set_size_inches(12,6)
plt.rcParams.update({'font.size': 22})
raz.figure.set_size_inches(12,6)
raz.set_xticks(range(-4,7))
raz.set(xlabel='Razão', ylabel='Densidade')
```

[35]:
```
[Text(0.5, 0, 'Razão'), Text(0, 0.5, 'Densidade')]
```



1.11.1 Analisando as distribuições

- Vamos analisar os intervalos presentes
- Encontrar o intervalo onde se encontram as stop words
- E quais os intervalos para os notícias verdadeiras e falsas
- Primeiramente separamos em dataframes específicos por classe pois até o momento todas as razões se encontram em apenas um DataFrame sendo ele `razao_falso_verdadeira_log`

Abaixo vamos criar para a classe verdadeira

[36]:
```
verdadeiras = []
for nome, cont in verdadeiro_count.items():
    if nome in razao_falso_verdadeira_log.keys():
        ratio = razao_falso_verdadeira_log[nome]
        if not np.isinf(ratio):
            verdadeiras.append({'palavra':nome, 'cont': cont, 'ratio': ratio})
verdadeiras = pd.DataFrame(verdadeiras)
```

Agora o DataFrame de razões das notícias falsas

[37]:
```
falsas = []
for nome, cont in falso_count.items():
    if nome in razao_falso_verdadeira_log.keys():
        ratio = razao_falso_verdadeira_log[nome]
        if not np.isinf(ratio):
            falsas.append({'palavra':nome, 'cont': cont, 'ratio': ratio})
falsas = pd.DataFrame(falsas)
```

E por ultimo vamos criar um DataFrame de classe neutra apenas com as razões das stop words

[38]:
```
stop = []
for nome in stopwords:
    if nome in razao_falso_verdadeira_log.keys():
        ratio = razao_falso_verdadeira_log[nome]
        cont = todas_count[nome]
        if not np.isinf(ratio):
            stop.append({'palavra':nome, 'cont': cont, 'ratio': ratio})
stop = pd.DataFrame(stop)
```

Agora vamos buscar os ranges, pegando a menor razão e a maior razão para cada classe

[39]:
```
print('Stop Words - razão mínima: {:.2f} e valor máximo de: {:.2f}'
      .format(min(stop['ratio']), max(stop['ratio'])))

print('Verdadeiras - razão mínima: {:.2f} e valor máximo de: {:.2f}'
      .format(min(verdadeiras['ratio']), max(verdadeiras['ratio'])))

print('Falsas - razão mínima: {:.2f} e valor máximo de: {:.2f}'
      .format(min(falsas['ratio']), max(falsas['ratio'])))
```

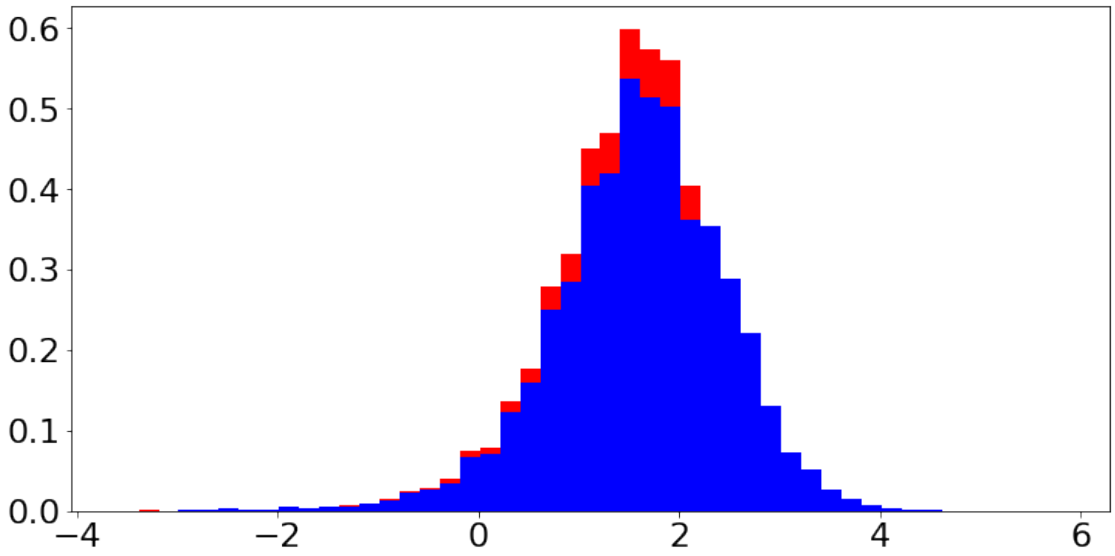
```
Stop Words - razão mínima: -0.18 e valor máximo de: 2.84
Verdadeiras - razão mínima: -3.58 e valor máximo de: 5.89
Falsas - razão mínima: -3.58 e valor máximo de: 4.72
```

- Como podemos observar as stop words ficam mais concentradas no centro da distrbuição
- As notícias Verdadeiras tem palavras variando dentro de toda a distrbuição e indo bastante para o lado direito do histograma
- As notícias Falsas tem palavras que também variam por toda a distrbuição, porém não chegam a ponta extrema a direita do histogramarama

- Podemos criar gráficos sobrepostos para podermos comparar onde estão as diferenças entre as classes verdadeiras e falsas

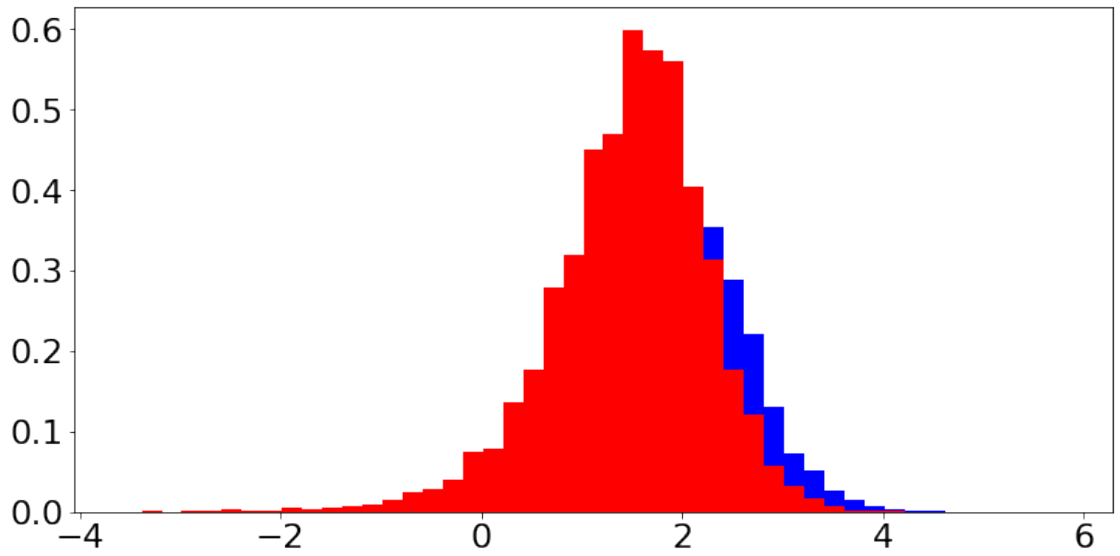
[40]:
```
razao_range = np.arange(min(s['razao']), max(s['razao']), 0.2)
fig = plt.figure(figsize=(12,6))
ax1 = fig.add_subplot(111)

ax1.hist(falsas['ratio'],density=True,bins=razao_range, color = 'red')
ax1.hist(verdadeiras['ratio'],density=True,bins=razao_range, color = 'blue')
```

```
plt.show()
```



- Se observarmos as distribuições parecem elas parecem bem parecidas
- Mas podemos notar uma leve tendência para a esquerda
- O que era esperado diante do cálculo da razão feito anteriormente.
- Abaixo vamos colocar a ditribuição de razões para notícias falsas na frente.

[41]:
```
fig = plt.figure(figsize=(12,6))
ax1 = fig.add_subplot(111)


ax1.hist(verdadeiras['ratio'],density=True,bins=razao_range, color = "blue")
ax1.hist(falsas['ratio'],density=True,bins=razao_range, color = "red")
plt.show()
```



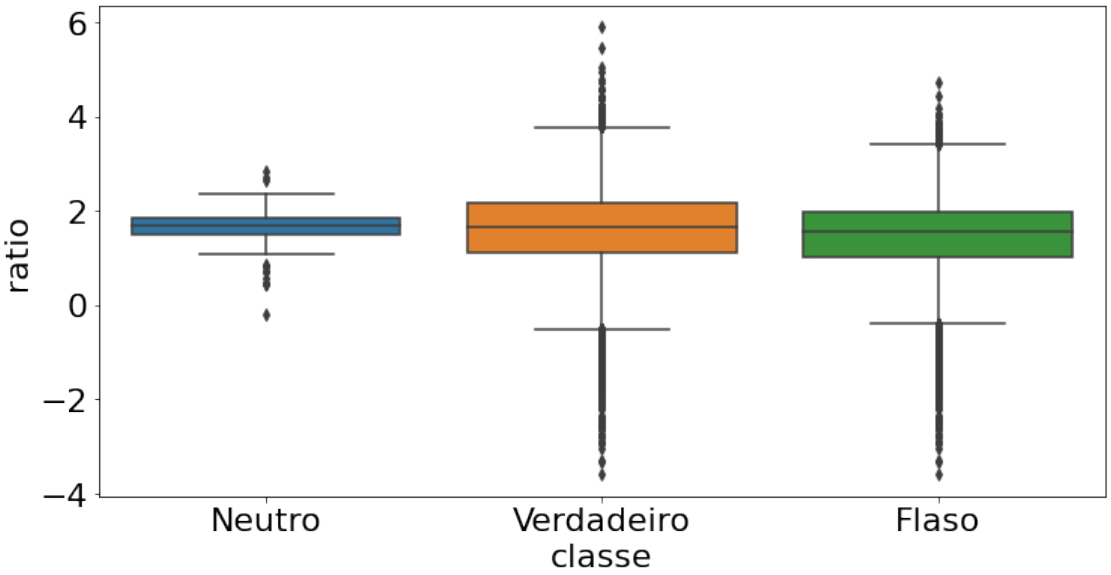
- Se vimos anteriormente que as notícias negativas tinham uma leve tendência a esquerda, agora temos a mesma visualização para notícias verdadeiras tendendo um pouco mais a direita.

Podemos usar de mais gráficos para enxergar como estão as distribuições por classe.

- Para criarmos as visualizções por classe seguintes, criamos primeiro um `DataFrame` onde:
  - temos um `DataFrame` de `stop words` com a classificação `Neutro`
  - padronizamos todas as colunas dos `DataFrames` de nomes `verdadeiras` e `falsas` criados anteriormente para colunas com nomes:
    - `word`, `count`, `ratio`, `classe`
  - concatenamos todos os dataframes.

[42]:
```
stop_df = []
for nome in stopwords:
    if nome in razao_falso_verdadeira_log.keys():
        ratio = razao_falso_verdadeira_log[nome]
        cont = todas_count[nome]
        if not np.isinf(ratio):
            stop_df.append({'palavra':nome, 'cont': cont, 'ratio': ratio,␣
 ↪'classe':'Neutro'})
stop_df = pd.DataFrame(stop_df)

stop_df.rename(columns={'palavra':'word', 'cont':'count'}, inplace=True)
falsas.rename(columns={'palavra': 'word', 'cont': 'count'}, inplace=True)
verdadeiras.rename(columns={'palavra': 'word', 'cont': 'count'}, inplace=True)

falsas['classe'] = 'Flaso'
verdadeiras['classe'] = 'Verdadeiro'

all_count_ratio = pd.concat([stop_df, verdadeiras, falsas])
```

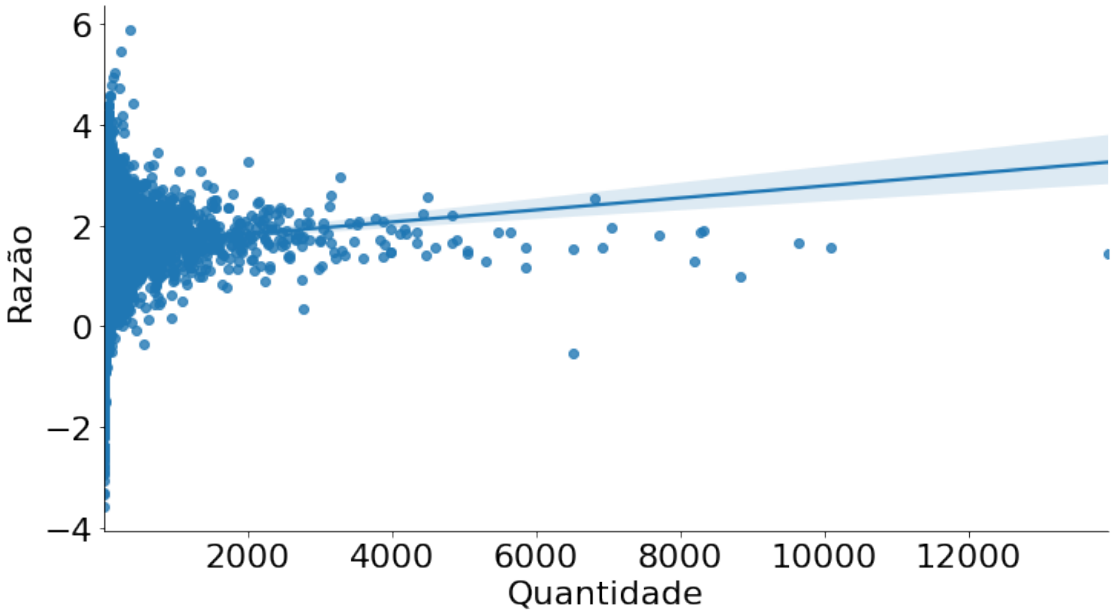
[43]:
```
ax = sns.boxplot(x='classe', y='ratio', data=all_count_ratio)
ax.figure.set_size_inches(12,6)
plt.rcParams.update({'font.size': 22})
```



- Podemos ver como a classe verdadeira está sutilmente acima da classe falsa.
- Mas por mais que possamos ver essa diferença bem sutíl entre as classes, ainda temos uma dificuldade de enxergar e dizer se faz sentido ou não, que palavras presentes em uma notícia podem denotar se a notícia é falsa ou não.
- Podemos então utilizar o lmplot, o mesmo além de plotar a distribuição tende também a mostrar para que lado a distribuição esta crescendo.

Vamos iniciar analisando como as notícias verdadeiras se comportam:

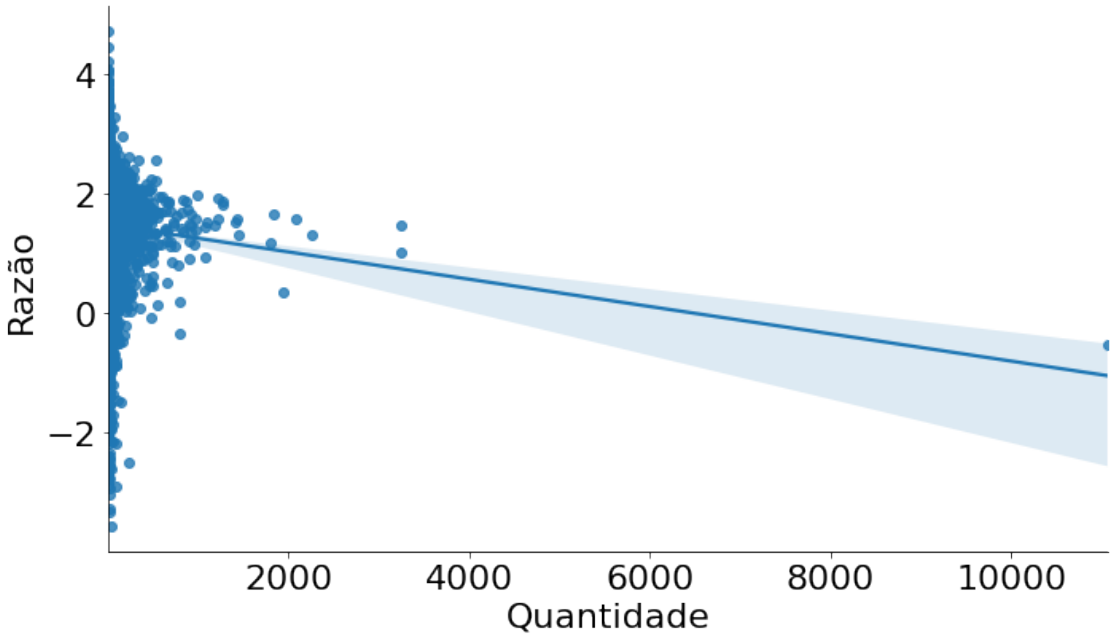
[44]:
```
ax = sns.lmplot(x='count', y='ratio', data=verdadeiras)
ax.set(xlabel='Quantidade', ylabel='Razão')
plt.gcf().set_size_inches(12, 6)
plt.rcParams.update({'font.size': 22})
```



- Como podemos observar a tendência é de que quanto mais ocorrem palavras em uma notícia verdadeira, ela tende a crescer para o lado positivo da razão, com valores acima de 2 se distanciando de forma mais lenta do centro da distribuição, em relação ao histograma que construímos, seria para o lado direito da distribuição.

Agora vamos olhar a tendência para notícias falsas:

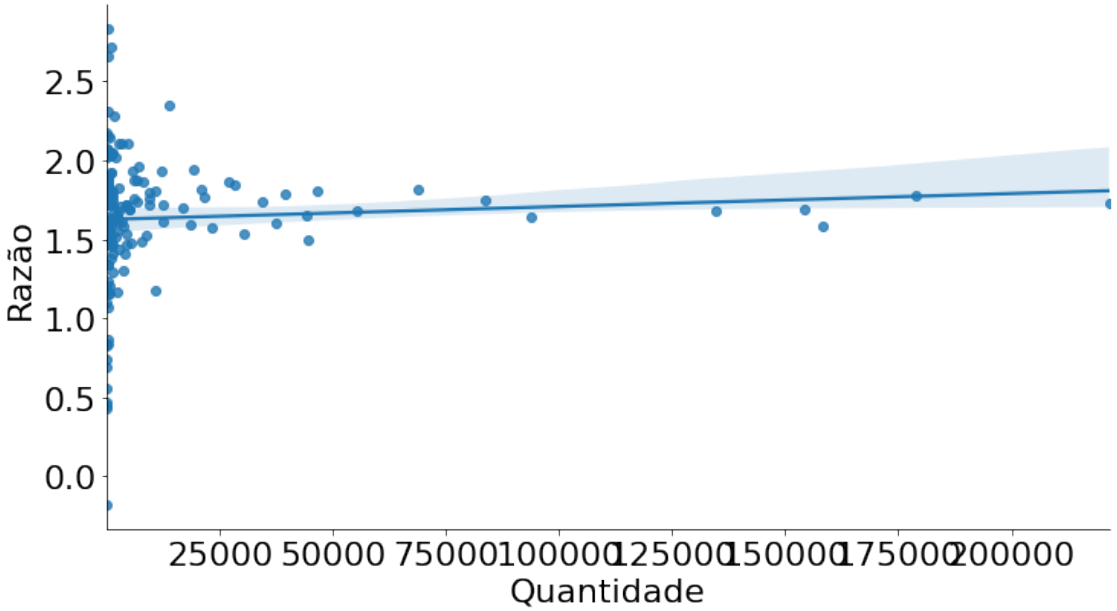
[45]:
```
len(falsas)
ax = sns.lmplot(x='count', y='ratio', data=falsas)
ax.set(xlabel='Quantidade', ylabel='Razão')
plt.gcf().set_size_inches(12, 6)
plt.rcParams.update({'font.size': 22})
```



- Caminha em direção oposta, inidicando que quanto mais palavras ocorrem em uma notícia falsa, mais palavras com menor valor de razão vão surgindo

Podemos também analisar o comportamento das stop words

[46]:
```
ax = sns.lmplot(x='cont', y='ratio', data=stop)
ax.set(xlabel='Quantidade', ylabel='Razão')
plt.gcf().set_size_inches(12, 6)

plt.rcParams.update({'font.size': 14})
```



- Como é esperado, as stop words vão continuar acontecendo em ambas as situações, apresentando uma linha quase reta de crescimento. O crescimento leve para cima indica-se mais pelo fator de notícias verdadeiras terem um tamanho mais elevado, mas o crescimento é muito pequeno.

Porem podemos estar sofrendo com algum viés de confirmação. As diferenças apresentadas, podem ser por coincidência, podem não ter relação como imaginamos, podem sofrer de creças iniciais que este trabalho propõe, já que o mesmo busca provar que através de palavras presentes em notícias, podemos identificar uma notícia falsa ou verdadeira.

Para acabar com a dúvida de uma vez por todas, usaremos um cálculo estatístico da biblioteca de estátitisca do `scipy.stats` com a função `ranksums`. Ela tem por intuito dizer se dois conjuntos de medidas (no nosso caso, das razões para notícias verdadeiras e falsas) são derivadas de uma mesma distribuição.

Vamos elaborar duas hipóteses:

Hnull > A distribuição da razão das palavras é a mesma para notícias verdadeiras e falsas

Halt > A distribuição da razão das palavras é a mesma para notícias verdadeiras e falsas

No nosso caso, estamos buscando validar a hipótese alternativa. O que a função ranksums propõe é que devemos aceitar a hipótese alternativa caso o valor do `p-value` seja menor que `0.05`, a diferença entre as duas distribuições seria significativa.

Link da documentação da função runksums

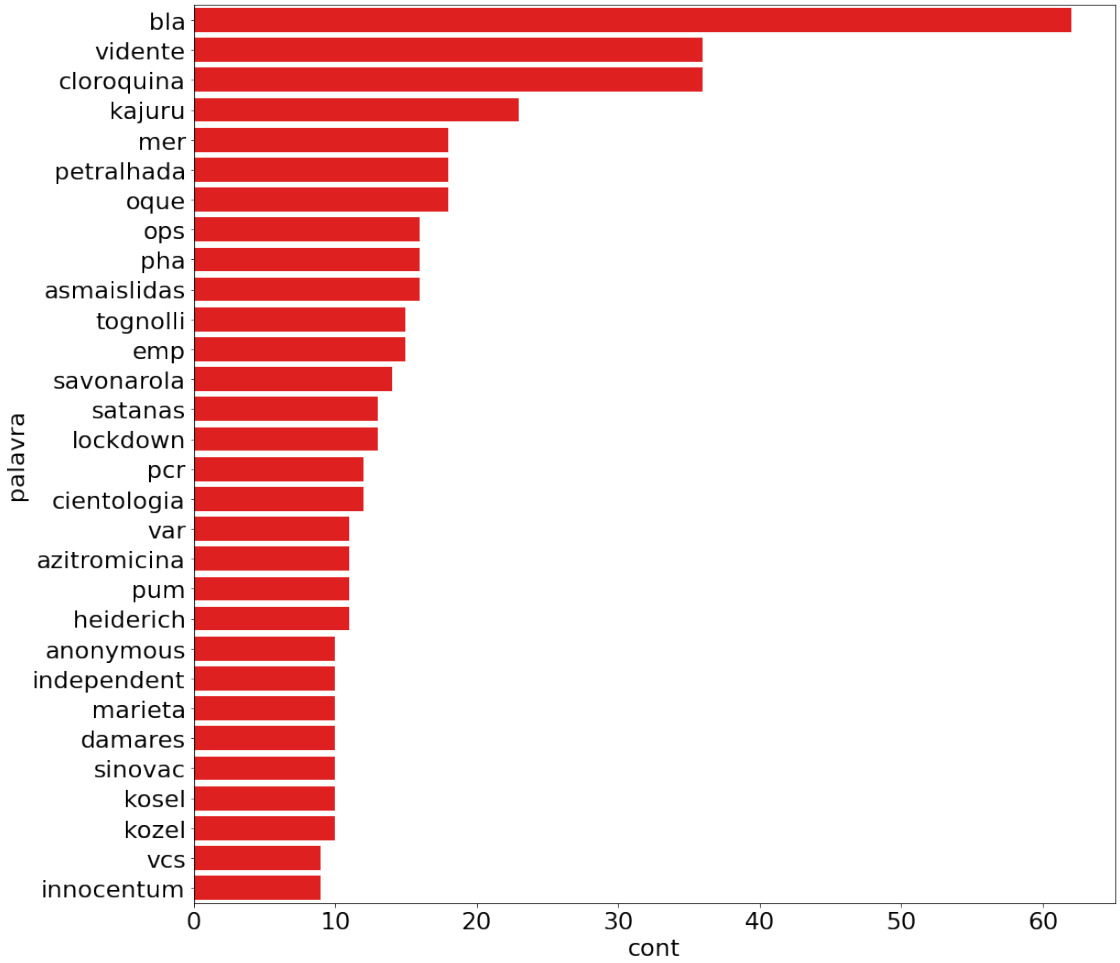
[47]:
```
from scipy.stats import ranksums
pvalue = ranksums(falsas['ratio'], verdadeiras['ratio']).pvalue

if pvalue < 0.05:
   print('Aceitar a hipótese alternativa')
else:
   print('Aceitar a hipótese nula')
```

```
Aceitar a hipótese alternativa
```

1.12 Analisando palavras que apresentaram ocorrência em apenas um dos lados

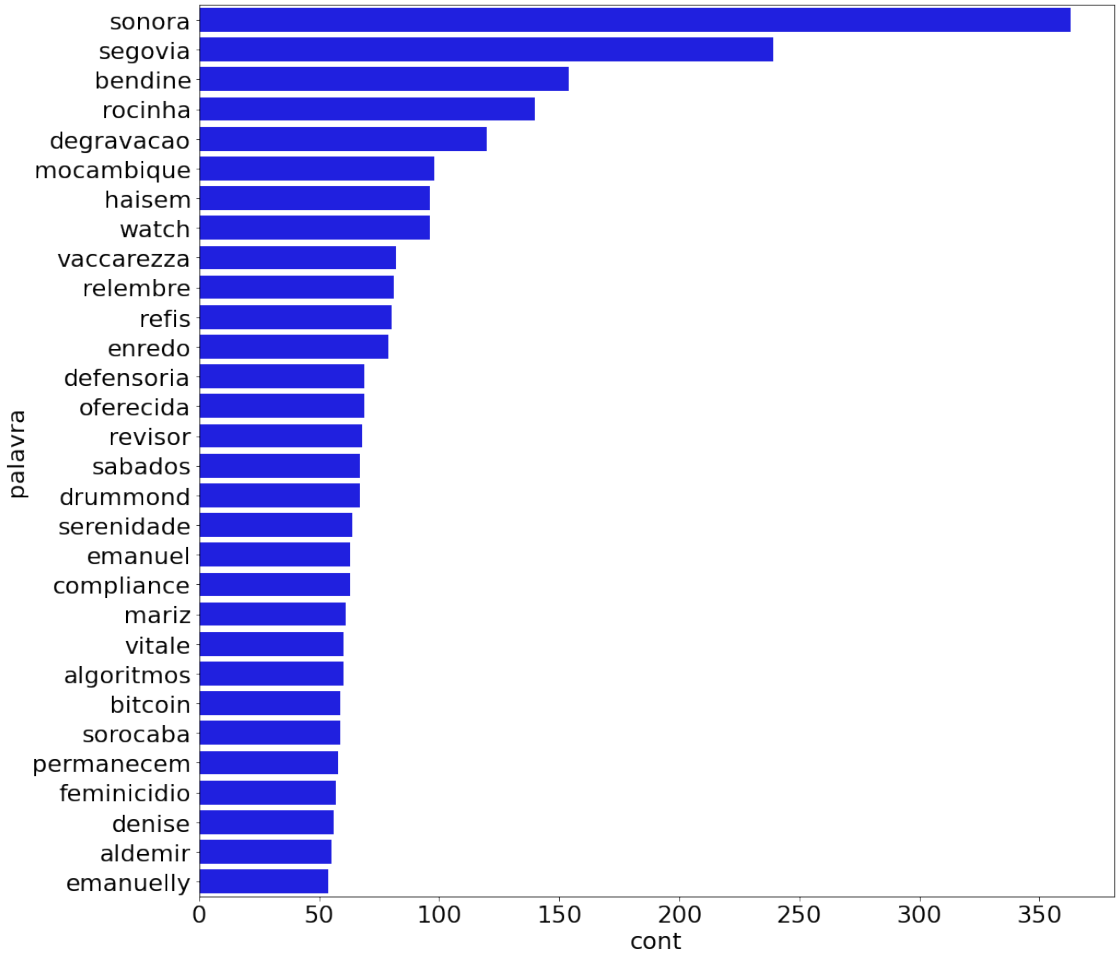
- Abaixo um gráfico com as palavras que mais apareceram em notícias falsas e não apareceram em notícias verdadeiras

[48]:
```
falso_para_verdadeiro = []
for palavra in falso_count.keys():
    if palavra not in verdadeiro_count.keys():
        falso_para_verdadeiro.append({'palavra': palavra, 'cont':␣
 ↪falso_count[palavra]})
falso_para_verdadeiro = pd.DataFrame(falso_para_verdadeiro)
falso_para_verdadeiro.sort_values(by='cont', ascending=False)[0:20]

falso_para_verdadeiro.sort_values(by='cont', ascending=False, inplace=True)
plt.rcParams.update({'font.size': 22})
ax = sns.barplot(y='palavra', x='cont', data=falso_para_verdadeiro[:30], ␣
 ↪color='red')
plt.gcf().set_size_inches(15, 15)
```



- É possível notar a presença de palavras escritas com grafia incorreta, palavras abreviadas típicas de redes sociais como "vcs" e palavras de linguagem mais coloquial

[55]:
```
verdadeiro_para_falso = []
for palavra in verdadeiro_count.keys():
    if palavra not in falso_count.keys():
        verdadeiro_para_falso.append({'palavra': palavra, 'cont':␣
 ↪verdadeiro_count[palavra]})
verdadeiro_para_falso = pd.DataFrame(verdadeiro_para_falso)
verdadeiro_para_falso.sort_values(by='cont', ascending=False)[0:20]

verdadeiro_para_falso.sort_values(by='cont', ascending=False, inplace=True)
ax = sns.barplot(y='palavra', x='cont', data=verdadeiro_para_falso[:30],␣
 ↪color='blue')
plt.gcf().set_size_inches(15, 15)
```



- Já para notícias verdadeiras não encontramos palavras com linguajar mais coloquial, e nem erros de grafia.

1.12.1 Conclusão da análise

- Dentro desta análise pudemos notar que notícias os seguintes fatores:

Notícias verdadeiras tem tamanho maior que notícias falsas, que notícias falsas tem linguajar podem utilizar de linguajar mais coloquial e abreviações alem de mais erros de grafia, e também podemos notar uma diferença na distribuição da razão entre palavras onde notícias verdadeiras e falsas tendem a ter razões tendendo a direções distintas.

2 Construindo um modelo

- Agora vamos construir alguns modelos de machine learning e ver se os mesmos conseguem aprender a identificar fake news, através apenas das palavras presentes dentro da notícia

- Vamos criar alugmas funções utilitárias que vão nos ajudar no treinameto dos modelos adiante

[50]:
```
def formatar_lote_de_noticias(noticias):
    lote = []
    for i, linha in noticias.iterrows():
        contagem = dict.fromkeys(vocabulario, 0)
        palavras = linha['texto'].split()

        for palavra in palavras:
            contagem[palavra] += 1
        lote.append(contagem)

    return pd.DataFrame(lote)
```

- A função acima tem por objetivo pegar um texto e transformar o mesmo em uma entrada para treinamento dos modelos. A função acima formata a entrada para o estilo **count** (contar) que seria criar uma entrada com a contagem da ocorrência de uma palavra em um dataset. Como mostra a tabela abaixo:

Por exemplo para a frase: "Trabalho de conclusão de curso" a tabela ficaria da seguinte maneira

de	trabalho	conclusao	curso	porem	presidente	covid	...
2	1	1	1	0	0	0	...

Repare que "de" aparece duas vezes na saída da entrada processada. E as palavras do vocabulário que não apareceram na frase ficam com o valor 0.

Vale ressaltar que a coluna do dataset são todas as palavras presentes no vocabulário, ou seja, temos 82.080

- Agora a função abaixo faz a transformação para o dataset **contains** (contém), que ao contrário da contagem, ele apenas sinaliza se a palavra apareceu ou não no dataset

[51]:
```
def formatar_lote_de_noticias_sim_nao(noticias):
    lote = []
    for i, linha in noticias.iterrows():
        contagem = dict.fromkeys(vocabulario, 0)
        palavras = linha['texto'].split()

        for palavra in palavras:
            contagem[palavra] = 1
        lote.append(contagem)

    return pd.DataFrame(lote)
```

O resultado para o mesmo exemplo citado acima, seria este:

de	trabalho	conclusao	curso	porem	presidente	covid	...
1	1	1	1	0	0	0	...

- Repare que agora a palavra "de" aparece apenas com o número 1, mesmo ocorrendo duas vezes na frase. A idéia é realmente dizer se a notícia contém a palavra não importa quantas vezes ela apareça.
- Usamos `1` para dizer que aparece e `0` para dizer que não aparece.

Desta forma podemos comparar e ver se o volume de uma palavra atrapalharia ou ajudaria o modelo a aprender, ou se seria indiferente.

- A função abaixo formata as classificações para as redes neurais que vamos construir.
- Os modelos de redes neurais que serão destacados mais a frente, espera que as classes sejam retornadas em formato de vetor. Por exemplo: enquanto utilizamos `1` para Verdadeiro e `0` para Falso, a rede neural espera uma classificação no estilo one hot encoding

Ficando desta maneira:

Verdadeiro	Falso
0	1
1	0

Ou seja quando for falso, será representado por um vetor [0,1] e para verdadeiro por um vetor [1,0]

[52]:
```
def format_labels(y_train):
    rows = []
    for value in y_train:
        if value == 0:
            rows.append([0,1])
        else:
            rows.append([1,0])
    return rows
```

- A função abaixo apenas formata o resultado do modelo de machine learning dizendo se ele classificou como Verdadeiro ou Falso no lugar de retornar números 0 ou 1

[53]:
```
def classificar_noticia(est, noticia):
    classe = est.predict(formatar_lote_de_noticias(pd.DataFrame([{'texto':
 ↪noticia}])))[0]
    return 'Verdadeiro' if classe == 1 else 'Falso'
```

- A função abaixo pega o resultado de uma rede neural e converte em um resultadoado simples de 0 ou 1 como os modelos de machine learning

[54]:
```
def classify(result):
    if(result[0] == 0 and result[1] == 1):
        return 0
```

```
    else:
        return 1
```

- Como o vocabulário tem 82.080 palavras, e cada palavra é uma coluna, o dataset fica grande demais para ser alocado na memória, pois o `DataFrame` resultante teria 12511 linhas por 82.080, o que demandaria muito espaço na memória. Desta forma vamos processar as informações em lotes.

[57]:
```
tamanho_lote = 128
```

- Vamos agora criar um `DataFrame` para treinamento com as notícias, e um `DataFrame` com as classes das notícias

[58]:
```
noticias, classes = todas_noticias['texto'].
↪to_frame(),todas_noticias['classificacao'].to_frame()
```

- Assim como temos as funções auxiliares para classificar transfomrar os textos em linhas de contains ou count. Para as classes precisamos transformar os textos em números para passar para o treinamento do modelo. Passaremos o valor de classe `falso` para `0` e `verdadeiro` para 1

[59]:
```
classes.classificacao = classes.classificacao.map(lambda x: 0 if x == 'falso'␣
↪else 1)
```

- Um processo importante no treinamento de modelos de machine learning e deep learning (redes neurais) é dividir os dados em dados de treinos e dados de teste, os dados de treino são usados pelos modelos para aprender a executar a tarefa. E para validar se o modelo aparendeu fazemos uso dos dados de teste.
- Seria um erro avaliar o modelo com apenas os dados de treino, o mesmo já aprendeu aquele caso, precisamos que ele saiba generalizar e executar para qualquer tipo de variação da informação.
- Também separamos uma parte dos dados para um dataset de validação, o mesmo pode ser usado durante o treino para garantir que o modelo está aprendendo e generalizando os casos e não apenas "decorando" aquele caso específico. A diferença do dataset de vilidação para o de testes, é que o de validação é utilizado durante o processo de treinamento apoiando o modelo no processo de otimização e generalização, porém ambos os datasets não podem fazer parte dos dados de treino.
- O dataset foi separado da seguinte maneira, 75% das notícias e suas respectivas classes foram separadas para treinamento do modelo. 25% dos dados foram separados para teste e validação. Sendo validação ficando apenas com 2 vezes o tamanho do lote que definimos, o dataset de validação tem apenas 256 registros, o restante fica com o dataset de teste.
- Utilizamos a função train_test_split para separação e sua documentação pode ser econtrada aqui

[60]:
```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(noticias,␣
↪classes['classificacao'].array, random_state=42)

validacao_treino, validacao_classe = X_test[:2*tamanho_lote], y_test[:
↪2*tamanho_lote]

X_test, y_test = X_test[2*tamanho_lote:], y_test[2*tamanho_lote:]
```

- Agora vamos criar o dataset de notícias, no formato em que as palavras do vocabulário se tornem as colunas, e que seja feita o **count** (contagem) de palavras. Guardaremos esta informação na variável `x_test`, para avaliarmos a performance dos modelos que foram treinados com dataset de **count**

[61]:
```
x_test = formatar_lote_de_noticias(X_test)
```

- Em segudia vamos criar também um dataset de teste mas agora para validar os modelos treiandos com o dataset **contains** (que apenas indicam a ocorrência da palavra e não a quantidade)

[62]:
```
x_test_sn = formatar_lote_de_noticias_sim_nao(X_test)
```

- Vamos montar um pequeno batch para uma avaliação rápida dos modelos de rede neural

[183]:
```
x_test_batch = formatar_lote_de_noticias(X_test[0:128]).to_numpy()
y_test_batch = np.asanyarray(format_labels(y_test[0:128].to_numpy()))
```

[330]:
```
x_test_batch_sn = formatar_lote_de_noticias_sim_nao(X_test[0:128]).to_numpy()
y_test_batch_sn = np.asanyarray(format_labels(y_test[0:128].to_numpy()))
```

- Vamos formatar a validação de acordo com os padrões de entrada do modelo usando as funções utilitárias definidas anteriormente. Primeiramente para dados com **count**

[63]:
```
val_formated = formatar_lote_de_noticias(validacao_treino).to_numpy()
val_formated_class = np.asanyarray(format_labels(validacao_classe.to_numpy()))
```

- Também vamos formatar as entradas para o dataset de **contains**

[65]:
```
val_formated_sn = formatar_lote_de_noticias_sim_nao(validacao_treino).to_numpy()
val_formated_class_sn = np.asanyarray(format_labels(validacao_classe.
↪to_numpy()))
```

- Abaixo importamos as funções para avaliação das métricas dos resultados dos modelos. Também definimos um vetor com o nome das classes pois um dos modelos de avaliação consegue avaliar os resultados por classe para termos uma análise ainda mais rica e decidirmos o melhor modelo.

[66]:
```
import joblib
from sklearn.metrics import confusion_matrix
from sklearn.metrics import precision_recall_fscore_support
```

```
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report

target_names = ['Falso', 'Verdadeiro']
```

2.0.1 Importando modelos utilizados

- Como estamos trabalhando com um dataset de tamanho elevado optei por utilizar dos modelos de machine do scikit learning aqueles que são escaláveis e possuem a função `partial_fit` para treinamento incremental do modelo, como vamos processar em lotes esta função vai ser estremamente útil para a tarefa. Além disso, em caso do modelo ter boa performance, podemos usar a função `partial_fit` para treinamento real para avaliação online das notícias que forem surgindo.
- Vamos treinar vários modelos para avaliação de performance, cada um tem sua particularidade e vamos avaliar qual se sai melhor dentre as condições de treino definidas.
- A documentação das funções importadas aqui que dão suporte a `partial_fit` podem ser encontradas dentro da documentação do scikit learning.
- Framework utilizado para construção de modelo de rede neural (deep learning) foi o keras. Cuja documentação pode ser encontrada neste link. O keras torna mais simples e abstrai a complexidade da sintaxe do tensorflow possibilitando escrever modelos de forma mais rápida e simples.

[258]:
```
from sklearn.linear_model import SGDClassifier
from sklearn.naive_bayes import MultinomialNB
from sklearn.linear_model import Perceptron
from sklearn.naive_bayes import BernoulliNB
from sklearn.linear_model import PassiveAggressiveClassifier
from keras.models import load_model
from keras.models import save_model
from keras.models import model_from_json
import tensorflow as tf
```

- Abaixo estamos apenas declarando uma lista para salar as estatísticas resultantes dos treinamentos dos modelos mais a frente.

[293]:
```
stats = []
```

- Definindo uma notícia teste para obtermos a avaliação do modelo.

[70]:
```
noticia = 'cloroquina salva milhoes em tratamento precoce'
```

2.1 SGD Classifier no dataset count

- Começando com os modelos lineares o primeiro testado foi o SGDClassifier configurado com o parâmetro loss como 'log' que da a regressão linear um classificador probabilistico. Como

estamos trabalhando a hipótese de que as palavras podem interferir no resultado da classificação, um classificador probabilístico é uma opção interessante para a situação. Os outros parâmetros do classificador foram mantidos padrões. O modelo foi treinado para ambos os datasets.

- O primeiro teste foi usando o dataset **counts**

2.1.1 Observação o modelo já foi treinado, caso queira usar o modelo já treinado ele pode ser carregado diretamente aqui

- A célula abaixo executa o treinamento

[ ]:
```
est = SGDClassifier(loss='log', penalty='l2', tol=1e-3)

inicio_lote = 0
fim_lote = tamanho_lote

total_lotes = str(int(len(X_train)/tamanho_lote))
ultimo_lote = False
i = 1
while True:
    x_train = formatar_lote_de_noticias(X_train[inicio_lote:fim_lote])
    est.partial_fit(x_train, y_train[inicio_lote:fim_lote], classes=[0,1])
    score = est.score(formatar_lote_de_noticias(validacao_treino),␣
 ↪validacao_classe)

    print(f'Lote {i} of {total_lotes} - score: {score}')
    time.sleep(0.10)

    inicio_lote += tamanho_lote

    if fim_lote + tamanho_lote > len(X_train):
        fim_lote = len(X_train)
        ultimo_lote = True
    else:
        fim_lote += tamanho_lote

    i += 1
    if ultimo_lote:
        break
```

- Avaliando o modelo

2.1.2 Exportando o modelo SGD

[ ]:
```
joblib.dump(est, 'pesos/sgd.sav')
```

2.1.3 Carregar modelo SGD

[73]:
```
sgd_model = joblib.load('pesos/sgd.sav')
```

- Classificar a notícia de teste usando o modelo SGD

[76]:
```
classificar_noticia(sgd_model, noticia)
```

[76]:
```
'Falso'
```

2.2 Avaliando as métricas do modelo SGD

- Avaliando a acurácia do modelo

[82]:
```
acuracia = sgd_model.score(x_test, y_test) * 100
print("A acurácia para o modelo SGD com o dataset counts foi de {:.2f}%".
 ↪format(acuracia))
```

```
A acurácia para o modelo SGD com o dataset counts foi de 61.04%
```

- obtendo as predições do modelo utilizando o dataset de treino com **counts**

[84]:
```
sgd_pred = sgd_model.predict(x_test)
```

2.3 Classification Report

- A função `classification_report` nos dá todas as métricas importantes que podemos obter de um modelo.
- Ela irá nos dar a precisão por classe.
- O quanto de revoação (recall) o modelo tem por classe e no geral
- O f1-score geral e por classe
- A acurária do modelo
- Todas estas métricas são úteis e nos ajudarão a escolher qual o modelo se saiu melhor e pode ser utilizado para esta tarefa.

[294]:
```
report = classification_report(y_test, sgd_pred, target_names=target_names, ␣
 ↪output_dict=True)
```

- Após obter o report do modelo vamos pegar as informações mais importantes que o relatório nos oferece para podermos comparar os modelos após todos os treinamentos.

[295]:
```
sgd_stats = {'model': 'sgd_count',
             'accuracy': report['accuracy'],
             'recall': report['macro avg']['recall'],
             'f1_score': report['macro avg']['f1-score'],
             'support': report['macro avg']['support'],
             'falso_precision': report['Falso']['precision'],
             'falso_recall': report['Falso']['recall'],
             'falso_f1_score': report['Falso']['f1-score'],
             'verdadeiro_precision': report['Verdadeiro']['precision'],
             'verdadeiro_recall': report['Verdadeiro']['recall'],
             'verdadeiro_f1_score': report['Verdadeiro']['f1-score']}
```

[296]:
```
stats.append(sgd_stats)
```

2.4 SGD Classifier no dataset contains

2.4.1 Observação o modelo já foi treinado, caso queira usar o modelo já treinado ele pode ser carregado diretamente aqui

- A célula abaixo executa o treinamento

[ ]:
```
est_sn = SGDClassifier(loss='log', penalty='l2', tol=1e-3)

inicio_lote = 0
fim_lote = tamanho_lote

total_lotes = str(int(len(X_train)/tamanho_lote))
ultimo_lote = False
i = 1
while True:
    x_train = formatar_lote_de_noticias_sim_nao(X_train[inicio_lote:fim_lote])
    est_sn.partial_fit(x_train, y_train[inicio_lote:fim_lote], classes=[0,1])
    score = est_sn.score(formatar_lote_de_noticias(validacao_treino),␣
 ↪validacao_classe)

    print(f'Lote {i} of {total_lotes} - score: {score}')
    time.sleep(0.10)

    inicio_lote += tamanho_lote

    if fim_lote + tamanho_lote > len(X_train):
        fim_lote = len(X_train)
        ultimo_lote = True
    else:
```

```
        fim_lote += tamanho_lote

    i += 1
    if ultimo_lote:
        break
```

- Avaliando o modelo

2.4.2 Exportando o modelo SGD (contains)

[ ]:
```
joblib.dump(est_sn, 'pesos/sgdsn.sav')
```

2.4.3 Carregar modelo SGD (contains)

[96]:
```
sgdsn_model_sn = joblib.load('pesos/sgdsn.sav')
```

- Classificar usando modelo SGD (contains)

[97]:
```
classificar_noticia(sgdsn_model_sn, noticia)
```

[97]:
```
'Verdadeiro'
```

2.5 Avaliando as métricas do modelo SGD contains

- Avaliando a acurácia do modelo

[100]:
```
acuracia = sgdsn_model_sn.score(x_test_sn, y_test) * 100
print("A acurácia para o modelo SGD com o dataset contains foi de {:.2f}%".
 ↪format(acuracia))
```

```
A acurácia para o modelo SGD com o dataset contains foi de 40.08%
```

- obtendo as predições do modelo utilizando o dataset de treino com **contains**

[102]:
```
sgdsn_pred = sgdsn_model_sn.predict(x_test_sn)
```

2.6 Classification Report

- A função `classification_report` nos dá todas as métricas importantes que podemos obter de um modelo.
- Ela irá nos dar a precisão por classe.
- O quanto de revoação (recall) o modelo tem por classe e no geral
- O f1-score geral e por classe

- A acurária do modelo
- Todas estas métricas são úteis e nos ajudarão a escolher qual o modelo se saiu melhor e pode ser utilizado para esta tarefa.

[297]:
```
report = classification_report(y_test, sgdsn_pred, target_names=target_names,␣
↪output_dict=True)
```

[298]:
```
sgd_stats = {'model': 'sgd_contains',
             'accuracy': report['accuracy'],
             'recall': report['macro avg']['recall'],
             'f1_score': report['macro avg']['f1-score'],
             'support': report['macro avg']['support'],
             'falso_precision': report['Falso']['precision'],
             'falso_recall': report['Falso']['recall'],
             'falso_f1_score': report['Falso']['f1-score'],
             'verdadeiro_precision': report['Verdadeiro']['precision'],
             'verdadeiro_recall': report['Verdadeiro']['recall'],
             'verdadeiro_f1_score': report['Verdadeiro']['f1-score']}
```

[299]:
```
stats.append(sgd_stats)
```

2.7 MultinomialNB com dataset counts

- O segundo modelo treinado foi MultinomialNB, em sua documentação consta que o modelo é aplicável para features discretas como contagem de palavras, o que também se encaixa bem para a hipótese que estamos validando. O modelo foi testado com os parâmetros padrões em ambos os datasets.
- O primeiro teste foi usando o dataset **contains**

2.7.1 Observação o modelo já foi treinado, caso queira usar o modelo já treinado ele pode ser carregado diretamente aqui

- A célula abaixo executa o treinamento

[ ]:
```
clf = MultinomialNB()
## Treinando
inicio_lote = 0
fim_lote = tamanho_lote

total_lotes = str(int(len(X_train)/tamanho_lote))
ultimo_lote = False
i = 1
while True:
    x_train = formatar_lote_de_noticias(X_train[inicio_lote:fim_lote])
    clf.partial_fit(x_train, y_train[inicio_lote:fim_lote], classes=[0,1])
```

```
    score = clf.score(formatar_lote_de_noticias(validacao_treino),␣
↪validacao_classe)

    print(f'Lote {i} of {total_lotes} - score: {score}')
    time.sleep(0.10)

    inicio_lote += tamanho_lote

    if fim_lote + tamanho_lote > len(X_train):
        fim_lote = len(X_train)
        ultimo_lote = True
    else:
        fim_lote += tamanho_lote

    i += 1
    if ultimo_lote:
        break
```

- Avaliando o modelo

2.7.2 Exportando o modelo MultinomialNB

[ ]:
```
joblib.dump(est_sn, 'pesos/MultinomialNB.sav')
```

2.7.3 Carregar modelo MultinomialNB

[108]:
```
multinb_model = joblib.load('pesos/MultinomialNB.sav')
```

- Classificar usando modelo MultinomialNB

[109]:
```
classificar_noticia(multinb_model, noticia)
```

[109]:
```
'Falso'
```

- Resultados do modelo

2.8 Avaliando as métricas do modelo MultinomialNB count

- Avaliando a acurácia do modelo

[127]:
```
acuracia = multinb_model.score(x_test, y_test) * 100
print("A acurácia para o modelo MultinomialNB com o dataset contains foi de {:.
↪2f}%".format(acuracia))
```

```
A acurácia para o modelo MultinomialNB com o dataset contains foi de 60.79%
```

- obtendo as predições do modelo utilizando o dataset de treino com **counts**

[111]:
```
multinb_model_pred = multinb_model.predict(x_test)
```

2.9 Classification Report

- A função `classification_report` nos dá todas as métricas importantes que podemos obter de um modelo.
- Ela irá nos dar a precisão por classe.
- O quanto de revoação (recall) o modelo tem por classe e no geral
- O f1-score geral e por classe
- A acurária do modelo
- Todas estas métricas são úteis e nos ajudarão a escolher qual o modelo se saiu melhor e pode ser utilizado para esta tarefa.

[300]:
```
report = classification_report(y_test, multinb_model_pred,␣
↪target_names=target_names, output_dict=True)
```

[301]:
```
sgd_stats = {'model': 'multinomial_count',
             'accuracy': report['accuracy'],
             'recall': report['macro avg']['recall'],
             'f1_score': report['macro avg']['f1-score'],
             'support': report['macro avg']['support'],
             'falso_precision': report['Falso']['precision'],
             'falso_recall': report['Falso']['recall'],
             'falso_f1_score': report['Falso']['f1-score'],
             'verdadeiro_precision': report['Verdadeiro']['precision'],
             'verdadeiro_recall': report['Verdadeiro']['recall'],
             'verdadeiro_f1_score': report['Verdadeiro']['f1-score']}
```

[302]:
```
stats.append(sgd_stats)
```

2.10 MultiomailNB com datastet contains

2.10.1 Observação o modelo já foi treinado, caso queira usar o modelo já treinado ele pode ser carregado diretamente aqui

- A célula abaixo executa o treinamento

[ ]:
```
clf_sn = MultinomialNB()

inicio_lote = 0
fim_lote = tamanho_lote
```

```
total_lotes = str(int(len(X_train)/tamanho_lote))
ultimo_lote = False
i = 1
while True:
    x_train = formatar_lote_de_noticias_sim_nao(X_train[inicio_lote:fim_lote])
    clf_sn.partial_fit(x_train, y_train[inicio_lote:fim_lote], classes=[0,1])
    score = clf_sn.score(formatar_lote_de_noticias(validacao_treino),␣
 ↪validacao_classe)

    print(f'Lote {i} of {total_lotes} - score: {score}')
    time.sleep(0.10)

    inicio_lote += tamanho_lote

    if fim_lote + tamanho_lote > len(X_train):
        fim_lote = len(X_train)
        ultimo_lote = True
    else:
        fim_lote += tamanho_lote

    i += 1
    if ultimo_lote:
        break
```

- Avaliando o modelo

2.10.2 Exportando o modelo MultinomialNB

[ ]:
```
joblib.dump(est_sn, 'pesos/MultinomialNBSN.sav')
```

2.10.3 Carregar modelo MultinomialNB

[115]:
```
multinb_model_sn = joblib.load('pesos/MultinomialNB.sav')
```

- Classificar usando modelo MultinomialNB

[116]:
```
classificar_noticia(multinb_model_sn, noticia)
```

[116]:
```
'Falso'
```

- Resultados do modelo

2.11 Avaliando as métricas do modelo MultinomialNB contains

- Avaliando a acurácia do modelo

[126]:
```
acuracia = multinb_model_sn.score(x_test_sn, y_test) * 100
print("A acurácia para o modelo MultinomialNB com o dataset contains foi de {:.
↪2f}%".format(acuracia))
```

```
A acurácia para o modelo MultinomialNB com o dataset contains foi de 60.79%
```

- obtendo as predições do modelo utilizando o dataset de treino com **contains**

[118]:
```
multinb_model_pred_sn = multinb_model_sn.predict(x_test_sn)
```

2.12 Classification Report

- A função `classification_report` nos dá todas as métricas importantes que podemos obter de um modelo.
- Ela irá nos dar a precisão por classe.
- O quanto de revoação (recall) o modelo tem por classe e no geral
- O f1-score geral e por classe
- A acurária do modelo
- Todas estas métricas são úteis e nos ajudarão a escolher qual o modelo se saiu melhor e pode ser utilizado para esta tarefa.

[303]:
```
report = classification_report(y_test, multinb_model_pred_sn,␣
↪target_names=target_names, output_dict=True)
```

[304]:
```
sgd_stats = {'model': 'multinomial_contains',
             'accuracy': report['accuracy'],
             'recall': report['macro avg']['recall'],
             'f1_score': report['macro avg']['f1-score'],
             'support': report['macro avg']['support'],
             'falso_precision': report['Falso']['precision'],
             'falso_recall': report['Falso']['recall'],
             'falso_f1_score': report['Falso']['f1-score'],
             'verdadeiro_precision': report['Verdadeiro']['precision'],
             'verdadeiro_recall': report['Verdadeiro']['recall'],
             'verdadeiro_f1_score': report['Verdadeiro']['f1-score']}
```

[305]:
```
stats.append(sgd_stats)
```

3 Perceptron com dataset count

- O terceiro modelo testado foi o Perceptron, base para construções de redes neurais de multiplas camadas, quando usado de maneira isolada ao invés de várias camadas o perceptron é apenas um modelo capaz apenas de aprender funções lineares. A parâmetrização do perceptron também não apresentou muitas opções para o problema em questão, então foi usado apenas o perceptron com seus parâmetros padrões testados sobre ambos os datasets.

3.0.1 Observação o modelo já foi treinado, caso queira usar o modelo já treinado ele pode ser carregado diretamente aqui

- A célula abaixo executa o treinamento

[ ]:
```
percep = Perceptron()

inicio_lote = 0
fim_lote = tamanho_lote

total_lotes = str(int(len(X_train)/tamanho_lote))
ultimo_lote = False
i = 1
while True:
    x_train = formatar_lote_de_noticias(X_train[inicio_lote:fim_lote])
    percep.partial_fit(x_train, y_train[inicio_lote:fim_lote], classes=[0,1])
    score = percep.score(formatar_lote_de_noticias(validacao_treino),␣
 ↪validacao_classe)

    print(f'Lote {i} of {total_lotes} - score: {score}')
    time.sleep(0.10)

    inicio_lote += tamanho_lote

    if fim_lote + tamanho_lote > len(X_train):
        fim_lote = len(X_train)
        ultimo_lote = True
    else:
        fim_lote += tamanho_lote

    i += 1
    if ultimo_lote:
        break
```

- Avaliando o modelo

3.0.2 Exportando o modelo Perceptron

[ ]:
```
joblib.dump(percep, 'pesos/Percep.sav')
```

3.0.3 Carregar modelo Perceptron

[123]:
```
perceb_model = joblib.load('pesos/Percep.sav')
```

- Classificar usando modelo Perceptron

[124]:
```
classificar_noticia(perceb_model, noticia)
```

[124]:
```
'Falso'
```

3.1 Avaliando as métricas do modelo Perceptron count

- Avaliando a acurácia do modelo

[139]:
```
acuracia = perceb_model.score(x_test, y_test) * 100
print("A acurácia para o modelo Perceptron com o dataset counts foi de {:.2f}%".
 ↪format(acuracia))
```

```
A acurácia para o modelo Perceptron com o dataset counts foi de 60.52%
```

- obtendo as predições do modelo utilizando o dataset de treino com **counts**

[129]:
```
perceb_model_pred = perceb_model.predict(x_test)
```

3.2 Classification Report

- A função `classification_report` nos dá todas as métricas importantes que podemos obter de um modelo.
- Ela irá nos dar a precisão por classe.
- O quanto de revoação (recall) o modelo tem por classe e no geral
- O f1-score geral e por classe
- A acurária do modelo
- Todas estas métricas são úteis e nos ajudarão a escolher qual o modelo se saiu melhor e pode ser utilizado para esta tarefa.

[306]:
```
report = classification_report(y_test, perceb_model_pred,␣
 ↪target_names=target_names, output_dict=True)
```

[307]:
```
sgd_stats = {'model': 'percep_count',
             'accuracy': report['accuracy'],
             'recall': report['macro avg']['recall'],
             'f1_score': report['macro avg']['f1-score'],
             'support': report['macro avg']['support'],
             'falso_precision': report['Falso']['precision'],
             'falso_recall': report['Falso']['recall'],
             'falso_f1_score': report['Falso']['f1-score'],
             'verdadeiro_precision': report['Verdadeiro']['precision'],
             'verdadeiro_recall': report['Verdadeiro']['recall'],
             'verdadeiro_f1_score': report['Verdadeiro']['f1-score']}
```

[308]:
```
stats.append(sgd_stats)
```

3.3 Perceptron com dataset contains

3.3.1 Observação o modelo já foi treinado, caso queira usar o modelo já treinado ele pode ser carregado diretamente aqui

- A célula abaixo executa o treinamento

[ ]:
```
percep_sn = Perceptron()
inicio_lote = 0
fim_lote = tamanho_lote

total_lotes = str(int(len(X_train)/tamanho_lote))
ultimo_lote = False
i = 1
while True:
    x_train = formatar_lote_de_noticias_sim_nao(X_train[inicio_lote:fim_lote])
    percep_sn.partial_fit(x_train, y_train[inicio_lote:fim_lote], classes=[0,1])
    score = percep_sn.score(formatar_lote_de_noticias(validacao_treino),␣
 ↪validacao_classe)

    print(f'Lote {i} of {total_lotes} - score: {score}')
    time.sleep(0.10)

    inicio_lote += tamanho_lote

    if fim_lote + tamanho_lote > len(X_train):
        fim_lote = len(X_train)
        ultimo_lote = True
    else:
        fim_lote += tamanho_lote

    i += 1
```

```
    if ultimo_lote:
        break
```

- Avaliando o modelo

3.3.2 Exportando o modelo Perceptron

[ ]:
```
joblib.dump(percep_sn, 'pesos/percep_sn.sav')
```

3.3.3 Carregar modelo Perceptron

[136]:
```
perceb_model_sn = joblib.load('pesos/percep_sn.sav')
```

- Classificar usando modelo Perceptron

[137]:
```
classificar_noticia(perceb_model_sn, noticia)
```

[137]:
```
'Verdadeiro'
```

3.4 Avaliando as métricas do modelo Perceptron contains

- Avaliando a acurácia do modelo

[138]:
```
acuracia = perceb_model_sn.score(x_test_sn, y_test) * 100
print("A acurácia para o modelo Perceptron com o dataset contains foi de {:.
 ↪2f}%".format(acuracia))
```

```
A acurácia para o modelo Perceptron com o dataset contains foi de 39.35%
```

- obtendo as predições do modelo utilizando o dataset de treino com **contains**

[140]:
```
perceb_model_pred_sn = perceb_model_sn.predict(x_test_sn)
```

3.5 Classification Report

- A função `classification_report` nos dá todas as métricas importantes que podemos obter de um modelo.
- Ela irá nos dar a precisão por classe.
- O quanto de revoação (recall) o modelo tem por classe e no geral
- O f1-score geral e por classe
- A acurária do modelo

- Todas estas métricas são úteis e nos ajudarão a escolher qual o modelo se saiu melhor e pode ser utilizado para esta tarefa.

[309]:
```
report = classification_report(y_test, perceb_model_pred_sn,␣
↪target_names=target_names, output_dict=True)from keras.models import␣
↪model_from_json
```

[310]:
```
sgd_stats = {'model': 'percep_contains',
             'accuracy': report['accuracy'],
             'recall': report['macro avg']['recall'],
             'f1_score': report['macro avg']['f1-score'],
             'support': report['macro avg']['support'],
             'falso_precision': report['Falso']['precision'],
             'falso_recall': report['Falso']['recall'],
             'falso_f1_score': report['Falso']['f1-score'],
             'verdadeiro_precision': report['Verdadeiro']['precision'],
             'verdadeiro_recall': report['Verdadeiro']['recall'],
             'verdadeiro_f1_score': report['Verdadeiro']['f1-score']}
```

[311]:
```
stats.append(sgd_stats)
```

3.6 Bernouli com dataset counts

- O quarto modelo testado foi BernoulliNB ao contrário do MultinomialNB bernouli foi feito para trabalhar bem com dados binários, ou seja a ocorrência ou não do registro, que é exatamente o que um dos datasets construído representa. Por se tratar de um modelo que por padrão já parece ser feito para o problema em questão foi mantida a configuração padrão do modelo e o mesmo foi testado em ambos os datasets.

3.6.1 Observação o modelo já foi treinado, caso queira usar o modelo já treinado ele pode ser carregado diretamente aqui

- A célula abaixo executa o treinamento

[ ]:
```
bern_count = BernoulliNB()
inicio_lote = 0
fim_lote = tamanho_lote

total_lotes = str(int(len(X_train)/tamanho_lote))
ultimo_lote = False
i = 1
while True:
    x_train = formatar_lote_de_noticias(X_train[inicio_lote:fim_lote])
    bern_count.partial_fit(x_train, y_train[inicio_lote:fim_lote],␣
↪classes=[0,1])
```

```
    score = bern_count.score(formatar_lote_de_noticias(validacao_treino),␣
↪validacao_classe)

    print(f'Lote {i} of {total_lotes} - score: {score}')
    time.sleep(0.10)

    inicio_lote += tamanho_lote

    if fim_lote + tamanho_lote > len(X_train):
        fim_lote = len(X_train)
        ultimo_lote = True
    else:
        fim_lote += tamanho_lote

    i += 1
    if ultimo_lote:
        break
```

- Avaliando o modelo

3.6.2 Exportando o modelo Bernouli

[ ]:
```
joblib.dump(percep_sn, 'pesos/bernouli.sav')
```

3.6.3 Carregar modelo Bernouli

[145]:
```
bern_model = joblib.load('pesos/bernouli.sav')
```

- Classificar usando modelo Bernouli

[146]:
```
classificar_noticia(bern_model, noticia)
```

[146]:
```
'Falso'
```

3.7 Avaliando as métricas do modelo Bernouli count

- Avaliando a acurácia do modelo

[147]:
```
acuracia = bern_model.score(x_test, y_test) * 100
print("A acurácia para o modelo Bernouli com o dataset count foi de {:.2f}%".
↪format(acuracia))
```

```
A acurácia para o modelo Bernouli com o dataset count foi de 83.77%
```

- obtendo as predições do modelo utilizando o dataset de treino com **count**

[148]:
```
bern_model_pred = bern_model.predict(x_test)
```

3.8 Classification Report

- A função `classification_report` nos dá todas as métricas importantes que podemos obter de um modelo.
- Ela irá nos dar a precisão por classe.
- O quanto de revoação (recall) o modelo tem por classe e no geral
- O f1-score geral e por classe
- A acurária do modelo
- Todas estas métricas são úteis e nos ajudarão a escolher qual o modelo se saiu melhor e pode ser utilizado para esta tarefa.

[312]:
```
report = classification_report(y_test, bern_model_pred,␣
↪target_names=target_names, output_dict=True)
```

[313]:
```
sgd_stats = {'model': 'bernouli_count',
             'accuracy': report['accuracy'],
             'recall': report['macro avg']['recall'],
             'f1_score': report['macro avg']['f1-score'],
             'support': report['macro avg']['support'],
             'falso_precision': report['Falso']['precision'],
             'falso_recall': report['Falso']['recall'],
             'falso_f1_score': report['Falso']['f1-score'],
             'verdadeiro_precision': report['Verdadeiro']['precision'],
             'verdadeiro_recall': report['Verdadeiro']['recall'],
             'verdadeiro_f1_score': report['Verdadeiro']['f1-score']}
```

[314]:
```
stats.append(sgd_stats)
```

3.9 Bernouli com dataset contains

3.9.1 Observação o modelo já foi treinado, caso queira usar o modelo já treinado ele pode ser carregado diretamente aqui

- A célula abaixo executa o treinamento

[ ]:
```
bern = BernoulliNB()
inicio_lote = 0
fim_lote = tamanho_lote

total_lotes = str(int(len(X_train)/tamanho_lote))
ultimo_lote = False
```

```
i = 1
while True:
    x_train = formatar_lote_de_noticias_sim_nao(X_train[inicio_lote:fim_lote])
    bern.partial_fit(x_train, y_train[inicio_lote:fim_lote], classes=[0,1])
    score = bern.score(formatar_lote_de_noticias(validacao_treino),␣
 ↪validacao_classe)

    print(f'Lote {i} of {total_lotes} - score: {score}')
    time.sleep(0.10)

    inicio_lote += tamanho_lote

    if fim_lote + tamanho_lote > len(X_train):
        fim_lote = len(X_train)
        ultimo_lote = True
    else:
        fim_lote += tamanho_lote

    i += 1
    if ultimo_lote:
        break
```

- Avaliando o modelo

3.9.2 Exportando o modelo Bernouli

[ ]:
```
joblib.dump(bern, 'pesos/bernoulisn.sav')
```

3.9.3 Carregar modelo Bernouli

[163]:
```
bern_model_sn = joblib.load('pesos/bernoulisn.sav')
```

- Classificar usando modelo Bernouli

[153]:
```
classificar_noticia(bern_model_sn, noticia)
```

[153]:
```
'Falso'
```

3.10 Avaliando as métricas do modelo Bernouli contains

- Avaliando a acurácia do modelo

[155]:
```
acuracia = bern_model_sn.score(x_test_sn, y_test) * 100
print("A acurácia para o modelo Bernouli com o dataset contains foi de {:.2f}%".
 ↪format(acuracia))
```

```
A acurácia para o modelo Bernouli com o dataset contains foi de 83.77%
```

- obtendo as predições do modelo utilizando o dataset de treino com **contains**

[156]:
```
bern_model_sn_pred = bern_model_sn.predict(x_test_sn)
```

3.11 Classification Report

- A função `classification_report` nos dá todas as métricas importantes que podemos obter de um modelo.
- Ela irá nos dar a precisão por classe.
- O quanto de revoação (recall) o modelo tem por classe e no geral
- O f1-score geral e por classe
- A acurária do modelo
- Todas estas métricas são úteis e nos ajudarão a escolher qual o modelo se saiu melhor e pode ser utilizado para esta tarefa.

[315]:
```
report = classification_report(y_test, bern_model_sn_pred,␣
↪target_names=target_names, output_dict=True)
```

[316]:
```
sgd_stats = {'model': 'benouli_contains',
             'accuracy': report['accuracy'],
             'recall': report['macro avg']['recall'],
             'f1_score': report['macro avg']['f1-score'],
             'support': report['macro avg']['support'],
             'falso_precision': report['Falso']['precision'],
             'falso_recall': report['Falso']['recall'],
             'falso_f1_score': report['Falso']['f1-score'],
             'verdadeiro_precision': report['Verdadeiro']['precision'],
             'verdadeiro_recall': report['Verdadeiro']['recall'],
             'verdadeiro_f1_score': report['Verdadeiro']['f1-score']}
```

[317]:
```
stats.append(sgd_stats)
```

3.12 Passive agressive classifier com dataset count

- O último modelo testado para os modelos do scikit learning foi o modelo Passive Aggressive Classifier, que trabalha de forma parecida com o perceptron, funciona de forma passiva quando o modelo acerta uma predição durante o treinamento e de forma agresiva alterando o modelo caso haja uma predição errada, como utilizamos o perceptron de forma padrão este modelo também foi utilizado com seus parâmetros padrões e treinado em ambos os datasets.

3.12.1 Observação o modelo já foi treinado, caso queira usar o modelo já treinado ele pode ser carregado diretamente aqui

- A célula abaixo executa o treinamento

[ ]:
```
passive = PassiveAggressiveClassifier()

inicio_lote = 0
fim_lote = tamanho_lote

total_lotes = str(int(len(X_train)/tamanho_lote))
ultimo_lote = False
i = 1
while True:
    x_train = formatar_lote_de_noticias(X_train[inicio_lote:fim_lote])
    passive.partial_fit(x_train, y_train[inicio_lote:fim_lote], classes=[0,1])
    score = passive.score(formatar_lote_de_noticias(validacao_treino),␣
 ↪validacao_classe)

    print(f'Lote {i} of {total_lotes} - score: {score}')
    time.sleep(0.10)

    inicio_lote += tamanho_lote

    if fim_lote + tamanho_lote > len(X_train):
        fim_lote = len(X_train)
        ultimo_lote = True
    else:
        fim_lote += tamanho_lote

    i += 1
    if ultimo_lote:
        break
```

- Avaliando o modelo

3.12.2 Exportando o modelo Passive Agressive

[ ]:
```
joblib.dump(passive, 'pesos/passive.sav')
```

3.12.3 Carregar modelo Passive Agressive

[165]:
```
passive_model = joblib.load('pesos/passive.sav')
```

- Classificar usando modelo Passive Agressive

[166]:
```
classificar_noticia(passive_model, noticia)
```

[166]:
```
'Falso'
```

3.13 Avaliando as métricas do modelo Passive Agressive count

- Avaliando a acurácia do modelo

[167]:
```
acuracia = passive_model.score(x_test, y_test) * 100
print("A acurácia para o modelo Passive Agressive com o dataset count foi de {:.
↪2f}%".format(acuracia))
```

```
A acurácia para o modelo Passive Agressive com o dataset count foi de 60.79%
```

- obtendo as predições do modelo utilizando o dataset de treino com **count**

[168]:
```
passive_model_pred = passive_model.predict(x_test)
```

3.14 Classification Report

- A função `classification_report` nos dá todas as métricas importantes que podemos obter de um modelo.
- Ela irá nos dar a precisão por classe.
- O quanto de revoação (recall) o modelo tem por classe e no geral
- O f1-score geral e por classe
- A acurária do modelo
- Todas estas métricas são úteis e nos ajudarão a escolher qual o modelo se saiu melhor e pode ser utilizado para esta tarefa.

[318]:
```
report = classification_report(y_test, passive_model_pred,␣
↪target_names=target_names, output_dict=True)
```

[319]:
```
sgd_stats = {'model': 'passive_count',
             'accuracy': report['accuracy'],
             'recall': report['macro avg']['recall'],
             'f1_score': report['macro avg']['f1-score'],
             'support': report['macro avg']['support'],
             'falso_precision': report['Falso']['precision'],
             'falso_recall': report['Falso']['recall'],
             'falso_f1_score': report['Falso']['f1-score'],
             'verdadeiro_precision': report['Verdadeiro']['precision'],
             'verdadeiro_recall': report['Verdadeiro']['recall'],
             'verdadeiro_f1_score': report['Verdadeiro']['f1-score']}
```

[320]:
```
stats.append(sgd_stats)
```

3.15 Passive agressive classifier com dataset contains

3.15.1 Observação o modelo já foi treinado, caso queira usar o modelo já treinado ele pode ser carregado diretamente aqui

- A célula abaixo executa o treinamento

[ ]:
```
passive_sn = PassiveAggressiveClassifier()

inicio_lote = 0
fim_lote = tamanho_lote

total_lotes = str(int(len(X_train)/tamanho_lote))
ultimo_lote = False
i = 1
while True:
    x_train = formatar_lote_de_noticias(X_train[inicio_lote:fim_lote])
    passive_sn.partial_fit(x_train, y_train[inicio_lote:fim_lote],␣
 ↪classes=[0,1])
    score = passive_sn.score(formatar_lote_de_noticias(validacao_treino),␣
 ↪validacao_classe)

    print(f'Lote {i} of {total_lotes} - score: {score}')
    time.sleep(0.10)

    inicio_lote += tamanho_lote

    if fim_lote + tamanho_lote > len(X_train):
        fim_lote = len(X_train)
        ultimo_lote = True
    else:
        fim_lote += tamanho_lote

    i += 1
    if ultimo_lote:
        break
```

- Avaliando o modelo

3.15.2 Exportando o modelo Passive Agressive

[ ]:
```
joblib.dump(passive_sn, 'pesos/passivesn.sav')
```

3.15.3 Carregar modelo Passive Agressive

[173]:
```
passive_model_sn = joblib.load('pesos/passivesn.sav')
```

- Classificar usando modelo Passive Agressive

[174]:
```
classificar_noticia(passive_model_sn, noticia)
```

[174]:
```
'Falso'
```

3.16 Avaliando as métricas do modelo Passive Agressive contains

- Avaliando a acurácia do modelo

[175]:
```
acuracia = passive_model_sn.score(x_test_sn, y_test) * 100
print("A acurácia para o modelo Passive Agressive com o dataset contains foi de␣
↪{:.2f}%".format(acuracia))
```

```
A acurácia para o modelo Passive Agressive com o dataset contains foi de 60.76%
```

- obtendo as predições do modelo utilizando o dataset de treino com **contains**

[176]:
```
passive_model_sn_pred = passive_model_sn.predict(x_test_sn)
```

3.17 Classification Report

- A função `classification_report` nos dá todas as métricas importantes que podemos obter de um modelo.
- Ela irá nos dar a precisão por classe.
- O quanto de revoação (recall) o modelo tem por classe e no geral
- O f1-score geral e por classe
- A acurária do modelo
- Todas estas métricas são úteis e nos ajudarão a escolher qual o modelo se saiu melhor e pode ser utilizado para esta tarefa.

[321]:
```
report = classification_report(y_test, passive_model_sn_pred,␣
↪target_names=target_names, output_dict=True)
```

[322]:
```
sgd_stats = {'model': 'passive_contains',
             'accuracy': report['accuracy'],
             'recall': report['macro avg']['recall'],
             'f1_score': report['macro avg']['f1-score'],
             'support': report['macro avg']['support'],
             'falso_precision': report['Falso']['precision'],
```

```
            'falso_recall': report['Falso']['recall'],
            'falso_f1_score': report['Falso']['f1-score'],
            'verdadeiro_precision': report['Verdadeiro']['precision'],
            'verdadeiro_recall': report['Verdadeiro']['recall'],
            'verdadeiro_f1_score': report['Verdadeiro']['f1-score']}
```

[323]:
```
stats.append(sgd_stats)
```

3.18 Redes Neurais

- As redes neurais foram construídas com o uso da biblioteca keras.
- Redes neurais são conhecidas por sua capacidade de aprender funções mais complexas e possuir uma alta capacidade de generalização. Com este intuito foram testadas redes neurais e validadas as performances em relação aos demais modelos de machine learning do scikit learning na tarefa de predição da classe das notícias.
- Para os modelos de rede neural, foram construídos 4 modelos, um modelo treiando sem conjunto de validação dos dados durante o treinamento e utilizando o dataset com a contagem de ocorencia de trabalho. Outro dataset também com sem conjunto de validação no treinamento utiliznado do dataset com a sinalização binária da presença da palavra no registro. Os outros dois modelos utilizaram conjunto de validação no treinamento o primeiro com dataset com a contagem de palavras e outro com dataset apenas com a sinalização da presença ou ausência da palavra.
- As redes neurais não possuem a função partial_fit, porém como possuem um modelo de aprendizado baseado na atualização de pesos, chamar o método fit após o mesmo já ter sido utilizado uma vez em um lote, somente atualiza os pesos da rede fazendo com que o aprendizado seja contínuo e não se inicie novamente o processo.
- As redes neurais seguiam a arquitetura da seguinte maneira, a camada de entrada possuí 82.020 neurônios, para a entrada do array com as informações da notícia, esta mesma camada é conectada em 12 neurônios sem função de ativação definida, que por sua vez se conecta em uma camada de 32 neurônios com ativação 'relu' que por sua vez se conecta a uma camada de 2 neurônios de saída com função de ativação softmax que nos dá a probabilidade da entrada ser de cada classe.

[276]:
```
def create_nn_model():
    nn = tf.keras.models.Sequential([
      tf.keras.layers.Dense(12, input_dim=82080),
      tf.keras.layers.Dense(32, activation='relu'),
      tf.keras.layers.Dense(2, activation='softmax')
    ])

    lossss = tf.keras.losses.BinaryCrossentropy(
        from_logits=False, label_smoothing=0, reduction="auto",␣
↪name="binary_crossentropy"
    )
```

```
    nn.compile(optimizer='adam',
                  loss=lossss,
                  metrics=['accuracy'])

    return nn
```

3.19 Rede Neural com dataset count e sem validação em treino

3.19.1 Observação o modelo já foi treinado, caso queira usar o modelo já treinado ele pode ser carregado diretamente aqui

- A célula cria o modelo de rede neural descrito

[287]:
```
model = create_nn_model()
```

- A célula abaixo treina a rede neural

[288]:
```
inicio_lote = 0
fim_lote = tamanho_lote

total_lotes = str(int(len(X_train)/tamanho_lote))
ultimo_lote = False
i = 1


while True:
    x_train = formatar_lote_de_noticias(X_train[inicio_lote:fim_lote])
    y_tr = np.asanyarray(format_labels(y_train[inicio_lote:fim_lote].
 ↪to_numpy()))
    model.fit(x_train.to_numpy(), y_tr, epochs=10)
    inicio_lote += tamanho_lote

    if fim_lote + tamanho_lote > len(X_train):
        fim_lote = len(X_train)
        ultimo_lote = True
    else:
        fim_lote += tamanho_lote

    i += 1
    if ultimo_lote:
        break
```

```
Epoch 1/10
4/4 [==============================] - 1s 47ms/step - loss: 0.6395 - accuracy:
0.5260
```

```
Epoch 2/10
4/4 [==============================] - 0s 48ms/step - loss: 0.5258 - accuracy:
0.4948
Epoch 3/10
4/4 [==============================] - 0s 32ms/step - loss: 0.5109 - accuracy:
0.7000
Epoch 4/10
4/4 [==============================] - 0s 41ms/step - loss: 0.4593 - accuracy:
0.9667
Epoch 5/10
4/4 [==============================] - 0s 37ms/step - loss: 0.3995 - accuracy:
0.9500
Epoch 6/10
4/4 [==============================] - 0s 41ms/step - loss: 0.3434 - accuracy:
0.9531
Epoch 7/10
4/4 [==============================] - 0s 35ms/step - loss: 0.2989 - accuracy:
0.9417
Epoch 8/10
4/4 [==============================] - 0s 43ms/step - loss: 0.2707 - accuracy:
0.9802
Epoch 9/10
4/4 [==============================] - 0s 36ms/step - loss: 0.2263 - accuracy:
0.9740
Epoch 10/10
4/4 [==============================] - 0s 36ms/step - loss: 0.2148 - accuracy:
0.9802
Epoch 1/10
4/4 [==============================] - 0s 49ms/step - loss: 0.5705 - accuracy:
0.7266
Epoch 2/10
4/4 [==============================] - 0s 29ms/step - loss: 0.4836 - accuracy:
0.7656
Epoch 3/10
4/4 [==============================] - 0s 30ms/step - loss: 0.3754 - accuracy:
0.8672
Epoch 4/10
4/4 [==============================] - 0s 30ms/step - loss: 0.3077 - accuracy:
0.8750
Epoch 5/10
4/4 [==============================] - 0s 58ms/step - loss: 0.2772 - accuracy:
0.8750
Epoch 6/10
4/4 [==============================] - 0s 60ms/step - loss: 0.2589 - accuracy:
0.8750
Epoch 7/10
4/4 [==============================] - 0s 44ms/step - loss: 0.2393 - accuracy:
0.8906
```

```
Epoch 8/10
4/4 [==============================] - 0s 34ms/step - loss: 0.2238 - accuracy:
0.9062
Epoch 9/10
4/4 [==============================] - 0s 41ms/step - loss: 0.2099 - accuracy:
0.9062
Epoch 10/10
4/4 [==============================] - 0s 42ms/step - loss: 0.1974 - accuracy:
0.9062
Epoch 1/10
4/4 [==============================] - 0s 42ms/step - loss: 0.6807 - accuracy:
0.7500
Epoch 2/10
4/4 [==============================] - 0s 38ms/step - loss: 0.5055 - accuracy:
0.7422
Epoch 3/10
4/4 [==============================] - 0s 31ms/step - loss: 0.4194 - accuracy:
0.7578
Epoch 4/10
4/4 [==============================] - 0s 45ms/step - loss: 0.3000 - accuracy:
0.8750
Epoch 5/10
4/4 [==============================] - 0s 35ms/step - loss: 0.2509 - accuracy:
0.8906
Epoch 6/10
4/4 [==============================] - 0s 32ms/step - loss: 0.2235 - accuracy:
0.9141
Epoch 7/10
4/4 [==============================] - 0s 40ms/step - loss: 0.2065 - accuracy:
0.9219
Epoch 8/10
4/4 [==============================] - 0s 66ms/step - loss: 0.1918 - accuracy:
0.9297
Epoch 9/10
4/4 [==============================] - 0s 45ms/step - loss: 0.1796 - accuracy:
0.9297
Epoch 10/10
4/4 [==============================] - 0s 39ms/step - loss: 0.1685 - accuracy:
0.9531
Epoch 1/10
4/4 [==============================] - 0s 30ms/step - loss: 0.4397 - accuracy:
0.8203
Epoch 2/10
4/4 [==============================] - 0s 41ms/step - loss: 0.2807 - accuracy:
0.8984
Epoch 3/10
4/4 [==============================] - 0s 44ms/step - loss: 0.2119 - accuracy:
0.9297
```

```
Epoch 4/10
4/4 [==============================] - 0s 51ms/step - loss: 0.1741 - accuracy:
0.9297
Epoch 5/10
4/4 [==============================] - 0s 50ms/step - loss: 0.1492 - accuracy:
0.9375
Epoch 6/10
4/4 [==============================] - 0s 56ms/step - loss: 0.1348 - accuracy:
0.9375
Epoch 7/10
4/4 [==============================] - 0s 43ms/step - loss: 0.1222 - accuracy:
0.9453
Epoch 8/10
4/4 [==============================] - 0s 55ms/step - loss: 0.1130 - accuracy:
0.9531
Epoch 9/10
4/4 [==============================] - 0s 49ms/step - loss: 0.1040 - accuracy:
0.9609
Epoch 10/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0973 - accuracy:
0.9609
Epoch 1/10
4/4 [==============================] - 0s 36ms/step - loss: 0.4103 - accuracy:
0.8125
Epoch 2/10
4/4 [==============================] - 0s 45ms/step - loss: 0.2896 - accuracy:
0.8750
Epoch 3/10
4/4 [==============================] - 0s 44ms/step - loss: 0.2206 - accuracy:
0.9219
Epoch 4/10
4/4 [==============================] - 0s 43ms/step - loss: 0.1809 - accuracy:
0.9219
Epoch 5/10
4/4 [==============================] - 0s 30ms/step - loss: 0.1538 - accuracy:
0.9297
Epoch 6/10
4/4 [==============================] - 0s 34ms/step - loss: 0.1380 - accuracy:
0.9297
Epoch 7/10
4/4 [==============================] - 0s 33ms/step - loss: 0.1265 - accuracy:
0.9453
Epoch 8/10
4/4 [==============================] - 0s 32ms/step - loss: 0.1168 - accuracy:
0.9453
Epoch 9/10
4/4 [==============================] - 0s 18ms/step - loss: 0.1080 - accuracy:
0.9531
```

```
Epoch 10/10
4/4 [==============================] - 0s 44ms/step - loss: 0.1000 - accuracy:
0.9531
Epoch 1/10
4/4 [==============================] - 0s 36ms/step - loss: 0.3048 - accuracy:
0.8672
Epoch 2/10
4/4 [==============================] - 0s 37ms/step - loss: 0.2348 - accuracy:
0.8984
Epoch 3/10
4/4 [==============================] - 0s 32ms/step - loss: 0.2025 - accuracy:
0.8984
Epoch 4/10
4/4 [==============================] - 0s 51ms/step - loss: 0.1816 - accuracy:
0.8984
Epoch 5/10
4/4 [==============================] - 0s 34ms/step - loss: 0.1636 - accuracy:
0.8984
Epoch 6/10
4/4 [==============================] - 0s 44ms/step - loss: 0.1476 - accuracy:
0.9062
Epoch 7/10
4/4 [==============================] - 0s 33ms/step - loss: 0.1347 - accuracy:
0.9141
Epoch 8/10
4/4 [==============================] - 0s 37ms/step - loss: 0.1221 - accuracy:
0.9219
Epoch 9/10
4/4 [==============================] - 0s 38ms/step - loss: 0.1115 - accuracy:
0.9453
Epoch 10/10
4/4 [==============================] - 0s 40ms/step - loss: 0.1024 - accuracy:
0.9531
Epoch 1/10
4/4 [==============================] - 0s 29ms/step - loss: 0.3428 - accuracy:
0.8516
Epoch 2/10
4/4 [==============================] - 0s 50ms/step - loss: 0.2390 - accuracy:
0.8906
Epoch 3/10
4/4 [==============================] - 0s 53ms/step - loss: 0.1848 - accuracy:
0.9062
Epoch 4/10
4/4 [==============================] - 0s 47ms/step - loss: 0.1530 - accuracy:
0.9141
Epoch 5/10
4/4 [==============================] - 0s 38ms/step - loss: 0.1345 - accuracy:
0.9297
```

```
Epoch 6/10
4/4 [==============================] - 0s 39ms/step - loss: 0.1198 - accuracy:
0.9375
Epoch 7/10
4/4 [==============================] - 0s 46ms/step - loss: 0.1081 - accuracy:
0.9531
Epoch 8/10
4/4 [==============================] - 0s 35ms/step - loss: 0.0989 - accuracy:
0.9609
Epoch 9/10
4/4 [==============================] - 0s 34ms/step - loss: 0.0900 - accuracy:
0.9766
Epoch 10/10
4/4 [==============================] - 0s 29ms/step - loss: 0.0832 - accuracy:
0.9844
Epoch 1/10
4/4 [==============================] - 0s 50ms/step - loss: 0.5393 - accuracy:
0.8125
Epoch 2/10
4/4 [==============================] - 0s 39ms/step - loss: 0.3159 - accuracy:
0.8594
Epoch 3/10
4/4 [==============================] - 0s 51ms/step - loss: 0.2329 - accuracy:
0.9062
Epoch 4/10
4/4 [==============================] - 0s 42ms/step - loss: 0.1826 - accuracy:
0.9141
Epoch 5/10
4/4 [==============================] - 0s 43ms/step - loss: 0.1593 - accuracy:
0.9219
Epoch 6/10
4/4 [==============================] - 0s 45ms/step - loss: 0.1385 - accuracy:
0.9531
Epoch 7/10
4/4 [==============================] - 0s 45ms/step - loss: 0.1270 - accuracy:
0.9531
Epoch 8/10
4/4 [==============================] - 0s 44ms/step - loss: 0.1167 - accuracy:
0.9531
Epoch 9/10
4/4 [==============================] - 0s 47ms/step - loss: 0.1081 - accuracy:
0.9609
Epoch 10/10
4/4 [==============================] - 0s 47ms/step - loss: 0.1007 - accuracy:
0.9688
Epoch 1/10
4/4 [==============================] - 0s 40ms/step - loss: 0.6414 - accuracy:
0.7266
```

```
Epoch 2/10
4/4 [==============================] - 0s 32ms/step - loss: 0.3350 - accuracy:
0.8281
Epoch 3/10
4/4 [==============================] - 0s 43ms/step - loss: 0.2402 - accuracy:
0.8516
Epoch 4/10
4/4 [==============================] - 0s 32ms/step - loss: 0.2036 - accuracy:
0.8516
Epoch 5/10
4/4 [==============================] - 0s 35ms/step - loss: 0.1710 - accuracy:
0.8984
Epoch 6/10
4/4 [==============================] - 0s 35ms/step - loss: 0.1480 - accuracy:
0.9531
Epoch 7/10
4/4 [==============================] - 0s 43ms/step - loss: 0.1283 - accuracy:
0.9688
Epoch 8/10
4/4 [==============================] - 0s 46ms/step - loss: 0.1168 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 36ms/step - loss: 0.1088 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 53ms/step - loss: 0.1011 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 43ms/step - loss: 0.5880 - accuracy:
0.7656
Epoch 2/10
4/4 [==============================] - 0s 28ms/step - loss: 0.3455 - accuracy:
0.8281
Epoch 3/10
4/4 [==============================] - 0s 41ms/step - loss: 0.1903 - accuracy:
0.9141
Epoch 4/10
4/4 [==============================] - 0s 42ms/step - loss: 0.1432 - accuracy:
0.9375
Epoch 5/10
4/4 [==============================] - 0s 47ms/step - loss: 0.1233 - accuracy:
0.9609
Epoch 6/10
4/4 [==============================] - 0s 45ms/step - loss: 0.1093 - accuracy:
0.9766
Epoch 7/10
4/4 [==============================] - 0s 55ms/step - loss: 0.0987 - accuracy:
0.9922
```

```
Epoch 8/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0902 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 34ms/step - loss: 0.0843 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0783 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 61ms/step - loss: 0.5126 - accuracy:
0.7969
Epoch 2/10
4/4 [==============================] - 0s 42ms/step - loss: 0.3433 - accuracy:
0.8594
Epoch 3/10
4/4 [==============================] - 0s 56ms/step - loss: 0.2363 - accuracy:
0.8906
Epoch 4/10
4/4 [==============================] - 0s 31ms/step - loss: 0.1608 - accuracy:
0.9453
Epoch 5/10
4/4 [==============================] - 0s 30ms/step - loss: 0.1284 - accuracy:
0.9766
Epoch 6/10
4/4 [==============================] - 0s 44ms/step - loss: 0.1084 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 57ms/step - loss: 0.0914 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0794 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 35ms/step - loss: 0.0706 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 39ms/step - loss: 0.0627 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 43ms/step - loss: 0.4989 - accuracy:
0.7812
Epoch 2/10
4/4 [==============================] - 0s 43ms/step - loss: 0.2293 - accuracy:
0.8906
Epoch 3/10
4/4 [==============================] - 0s 48ms/step - loss: 0.1696 - accuracy:
0.9297
```

```
Epoch 4/10
4/4 [==============================] - 0s 33ms/step - loss: 0.1396 - accuracy:
0.9297
Epoch 5/10
4/4 [==============================] - 0s 44ms/step - loss: 0.1174 - accuracy:
0.9609
Epoch 6/10
4/4 [==============================] - 0s 36ms/step - loss: 0.0990 - accuracy:
0.9688
Epoch 7/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0844 - accuracy:
0.9688
Epoch 8/10
4/4 [==============================] - 0s 34ms/step - loss: 0.0734 - accuracy:
0.9766
Epoch 9/10
4/4 [==============================] - 0s 39ms/step - loss: 0.0643 - accuracy:
0.9844
Epoch 10/10
4/4 [==============================] - 0s 32ms/step - loss: 0.0574 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 58ms/step - loss: 0.5911 - accuracy:
0.7812
Epoch 2/10
4/4 [==============================] - 0s 45ms/step - loss: 0.3353 - accuracy:
0.8281
Epoch 3/10
4/4 [==============================] - 0s 36ms/step - loss: 0.2496 - accuracy:
0.9141
Epoch 4/10
4/4 [==============================] - 0s 46ms/step - loss: 0.1820 - accuracy:
0.9453
Epoch 5/10
4/4 [==============================] - 0s 40ms/step - loss: 0.1428 - accuracy:
0.9688
Epoch 6/10
4/4 [==============================] - 0s 50ms/step - loss: 0.1143 - accuracy:
0.9766
Epoch 7/10
4/4 [==============================] - 0s 27ms/step - loss: 0.0967 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 55ms/step - loss: 0.0818 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0706 - accuracy:
1.0000
```

```
Epoch 10/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0622 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 49ms/step - loss: 0.5255 - accuracy:
0.7812
Epoch 2/10
4/4 [==============================] - 0s 39ms/step - loss: 0.2334 - accuracy:
0.8984
Epoch 3/10
4/4 [==============================] - 0s 40ms/step - loss: 0.1424 - accuracy:
0.9531
Epoch 4/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0988 - accuracy:
0.9844
Epoch 5/10
4/4 [==============================] - 0s 57ms/step - loss: 0.0751 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0612 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 43ms/step - loss: 0.0520 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0449 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 53ms/step - loss: 0.0401 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 35ms/step - loss: 0.0363 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 54ms/step - loss: 0.6091 - accuracy:
0.7969
Epoch 2/10
4/4 [==============================] - 0s 25ms/step - loss: 0.2686 - accuracy:
0.8594
Epoch 3/10
4/4 [==============================] - 0s 41ms/step - loss: 0.1698 - accuracy:
0.9219
Epoch 4/10
4/4 [==============================] - 0s 35ms/step - loss: 0.1298 - accuracy:
0.9375
Epoch 5/10
4/4 [==============================] - 0s 40ms/step - loss: 0.1019 - accuracy:
0.9609
```

```
Epoch 6/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0848 - accuracy:
0.9688
Epoch 7/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0699 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 39ms/step - loss: 0.0615 - accuracy:
0.9844
Epoch 9/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0546 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 43ms/step - loss: 0.0480 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 40ms/step - loss: 0.4421 - accuracy:
0.8516
Epoch 2/10
4/4 [==============================] - 0s 43ms/step - loss: 0.2442 - accuracy:
0.8984
Epoch 3/10
4/4 [==============================] - 0s 43ms/step - loss: 0.1312 - accuracy:
0.9453
Epoch 4/10
4/4 [==============================] - 0s 31ms/step - loss: 0.0876 - accuracy:
0.9688
Epoch 5/10
4/4 [==============================] - 0s 25ms/step - loss: 0.0671 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 58ms/step - loss: 0.0548 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 27ms/step - loss: 0.0455 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 27ms/step - loss: 0.0370 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 43ms/step - loss: 0.0325 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0276 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 34ms/step - loss: 0.4669 - accuracy:
0.8438
```

```
Epoch 2/10
4/4 [==============================] - 0s 45ms/step - loss: 0.1910 - accuracy:
0.9062
Epoch 3/10
4/4 [==============================] - 0s 45ms/step - loss: 0.1002 - accuracy:
0.9531
Epoch 4/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0717 - accuracy:
0.9844
Epoch 5/10
4/4 [==============================] - 0s 53ms/step - loss: 0.0530 - accuracy:
1.0000
Epoch 6/10
4/4 [==============================] - 0s 33ms/step - loss: 0.0426 - accuracy:
1.0000
Epoch 7/10
4/4 [==============================] - 0s 24ms/step - loss: 0.0345 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0296 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0260 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0235 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 48ms/step - loss: 0.4243 - accuracy:
0.8359
Epoch 2/10
4/4 [==============================] - 0s 43ms/step - loss: 0.2505 - accuracy:
0.8984
Epoch 3/10
4/4 [==============================] - 0s 42ms/step - loss: 0.1410 - accuracy:
0.9219
Epoch 4/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0961 - accuracy:
0.9375
Epoch 5/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0632 - accuracy:
1.0000
Epoch 6/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0472 - accuracy:
1.0000
Epoch 7/10
4/4 [==============================] - 0s 55ms/step - loss: 0.0378 - accuracy:
1.0000
```

```
Epoch 8/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0318 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0274 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 62ms/step - loss: 0.0245 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 45ms/step - loss: 0.6073 - accuracy:
0.7969
Epoch 2/10
4/4 [==============================] - 0s 40ms/step - loss: 0.3098 - accuracy:
0.8750
Epoch 3/10
4/4 [==============================] - 0s 54ms/step - loss: 0.1494 - accuracy:
0.9375
Epoch 4/10
4/4 [==============================] - 0s 31ms/step - loss: 0.0818 - accuracy:
0.9844
Epoch 5/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0556 - accuracy:
0.9922
Epoch 6/10
4/4 [==============================] - 0s 60ms/step - loss: 0.0409 - accuracy:
1.0000
Epoch 7/10
4/4 [==============================] - 0s 33ms/step - loss: 0.0312 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 22ms/step - loss: 0.0262 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 34ms/step - loss: 0.0224 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0195 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 40ms/step - loss: 0.4854 - accuracy:
0.8438
Epoch 2/10
4/4 [==============================] - 0s 38ms/step - loss: 0.2212 - accuracy:
0.9297
Epoch 3/10
4/4 [==============================] - 0s 47ms/step - loss: 0.1248 - accuracy:
0.9453
```

```
Epoch 4/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0655 - accuracy:
0.9844
Epoch 5/10
4/4 [==============================] - 0s 34ms/step - loss: 0.0421 - accuracy:
0.9922
Epoch 6/10
4/4 [==============================] - 0s 55ms/step - loss: 0.0317 - accuracy:
1.0000
Epoch 7/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0255 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0215 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0184 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0163 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 47ms/step - loss: 0.6914 - accuracy:
0.8125
Epoch 2/10
4/4 [==============================] - 0s 43ms/step - loss: 0.3320 - accuracy:
0.8828
Epoch 3/10
4/4 [==============================] - 0s 37ms/step - loss: 0.2158 - accuracy:
0.9062
Epoch 4/10
4/4 [==============================] - 0s 45ms/step - loss: 0.1368 - accuracy:
0.9375
Epoch 5/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0870 - accuracy:
0.9688
Epoch 6/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0603 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 56ms/step - loss: 0.0456 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0365 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0302 - accuracy:
1.0000
```

```
Epoch 10/10
4/4 [==============================] - 0s 36ms/step - loss: 0.0255 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 28ms/step - loss: 0.3388 - accuracy:
0.8594
Epoch 2/10
4/4 [==============================] - 0s 53ms/step - loss: 0.1768 - accuracy:
0.9297
Epoch 3/10
4/4 [==============================] - 0s 40ms/step - loss: 0.1182 - accuracy:
0.9688
Epoch 4/10
4/4 [==============================] - 0s 35ms/step - loss: 0.0789 - accuracy:
0.9766
Epoch 5/10
4/4 [==============================] - 0s 34ms/step - loss: 0.0618 - accuracy:
0.9766
Epoch 6/10
4/4 [==============================] - 0s 32ms/step - loss: 0.0486 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0355 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0280 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 55ms/step - loss: 0.0207 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0155 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 42ms/step - loss: 0.4667 - accuracy:
0.8281
Epoch 2/10
4/4 [==============================] - 0s 35ms/step - loss: 0.3083 - accuracy:
0.8828
Epoch 3/10
4/4 [==============================] - 0s 43ms/step - loss: 0.2319 - accuracy:
0.9062
Epoch 4/10
4/4 [==============================] - 0s 36ms/step - loss: 0.1708 - accuracy:
0.9219
Epoch 5/10
4/4 [==============================] - 0s 43ms/step - loss: 0.1305 - accuracy:
0.9453
```

```
Epoch 6/10
4/4 [==============================] - 0s 39ms/step - loss: 0.1006 - accuracy:
0.9531
Epoch 7/10
4/4 [==============================] - 0s 36ms/step - loss: 0.0807 - accuracy:
0.9609
Epoch 8/10
4/4 [==============================] - 0s 33ms/step - loss: 0.0651 - accuracy:
0.9609
Epoch 9/10
4/4 [==============================] - 0s 35ms/step - loss: 0.0508 - accuracy:
0.9844
Epoch 10/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0435 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 35ms/step - loss: 0.4824 - accuracy:
0.7891
Epoch 2/10
4/4 [==============================] - 0s 33ms/step - loss: 0.3069 - accuracy:
0.8281
Epoch 3/10
4/4 [==============================] - 0s 36ms/step - loss: 0.2030 - accuracy:
0.8984
Epoch 4/10
4/4 [==============================] - 0s 27ms/step - loss: 0.1365 - accuracy:
0.9531
Epoch 5/10
4/4 [==============================] - 0s 51ms/step - loss: 0.1001 - accuracy:
0.9531
Epoch 6/10
4/4 [==============================] - 0s 32ms/step - loss: 0.0794 - accuracy:
0.9609
Epoch 7/10
4/4 [==============================] - 0s 25ms/step - loss: 0.0617 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 34ms/step - loss: 0.0491 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0399 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 36ms/step - loss: 0.0338 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 43ms/step - loss: 0.4246 - accuracy:
0.7969
```

```
Epoch 2/10
4/4 [==============================] - 0s 37ms/step - loss: 0.2373 - accuracy:
0.8594
Epoch 3/10
4/4 [==============================] - 0s 46ms/step - loss: 0.1478 - accuracy:
0.9297
Epoch 4/10
4/4 [==============================] - 0s 38ms/step - loss: 0.1033 - accuracy:
0.9688
Epoch 5/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0756 - accuracy:
0.9766
Epoch 6/10
4/4 [==============================] - 0s 55ms/step - loss: 0.0584 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0458 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0385 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0321 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 32ms/step - loss: 0.0278 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 44ms/step - loss: 0.4807 - accuracy:
0.8359
Epoch 2/10
4/4 [==============================] - 0s 39ms/step - loss: 0.2560 - accuracy:
0.8828
Epoch 3/10
4/4 [==============================] - 0s 58ms/step - loss: 0.1487 - accuracy:
0.9141
Epoch 4/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0926 - accuracy:
0.9609
Epoch 5/10
4/4 [==============================] - 0s 51ms/step - loss: 0.0688 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0543 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0426 - accuracy:
0.9922
```

```
Epoch 8/10
4/4 [==============================] - 0s 39ms/step - loss: 0.0351 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0296 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 31ms/step - loss: 0.0241 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 52ms/step - loss: 0.5251 - accuracy:
0.7500
Epoch 2/10
4/4 [==============================] - 0s 38ms/step - loss: 0.3528 - accuracy:
0.8203
Epoch 3/10
4/4 [==============================] - 0s 32ms/step - loss: 0.2197 - accuracy:
0.8750
Epoch 4/10
4/4 [==============================] - 0s 37ms/step - loss: 0.1394 - accuracy:
0.9453
Epoch 5/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0914 - accuracy:
0.9766
Epoch 6/10
4/4 [==============================] - 0s 36ms/step - loss: 0.0608 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0434 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 39ms/step - loss: 0.0315 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0267 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 26ms/step - loss: 0.0217 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 62ms/step - loss: 0.3814 - accuracy:
0.8203
Epoch 2/10
4/4 [==============================] - 0s 34ms/step - loss: 0.1760 - accuracy:
0.9297
Epoch 3/10
4/4 [==============================] - 0s 40ms/step - loss: 0.1065 - accuracy:
0.9609
```

```
Epoch 4/10
4/4 [==============================] - 0s 31ms/step - loss: 0.0611 - accuracy:
0.9922
Epoch 5/10
4/4 [==============================] - 0s 22ms/step - loss: 0.0433 - accuracy:
1.0000
Epoch 6/10
4/4 [==============================] - 0s 23ms/step - loss: 0.0291 - accuracy:
1.0000
Epoch 7/10
4/4 [==============================] - 0s 33ms/step - loss: 0.0222 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 35ms/step - loss: 0.0184 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0157 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 26ms/step - loss: 0.0137 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 50ms/step - loss: 0.2468 - accuracy:
0.8828
Epoch 2/10
4/4 [==============================] - 0s 46ms/step - loss: 0.1621 - accuracy:
0.9453
Epoch 3/10
4/4 [==============================] - 0s 45ms/step - loss: 0.1079 - accuracy:
0.9688
Epoch 4/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0859 - accuracy:
0.9766
Epoch 5/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0660 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0538 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 51ms/step - loss: 0.0391 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 51ms/step - loss: 0.0295 - accuracy:
0.9844
Epoch 9/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0241 - accuracy:
0.9922
```

```
Epoch 10/10
4/4 [==============================] - 0s 32ms/step - loss: 0.0205 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 42ms/step - loss: 0.3853 - accuracy:
0.8438
Epoch 2/10
4/4 [==============================] - 0s 56ms/step - loss: 0.2212 - accuracy:
0.9062
Epoch 3/10
4/4 [==============================] - 0s 53ms/step - loss: 0.1534 - accuracy:
0.9375
Epoch 4/10
4/4 [==============================] - 0s 53ms/step - loss: 0.0935 - accuracy:
0.9688
Epoch 5/10
4/4 [==============================] - 0s 71ms/step - loss: 0.0612 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0429 - accuracy:
1.0000
Epoch 7/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0319 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 31ms/step - loss: 0.0252 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 32ms/step - loss: 0.0213 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 32ms/step - loss: 0.0174 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 26ms/step - loss: 0.2713 - accuracy:
0.8516
Epoch 2/10
4/4 [==============================] - 0s 35ms/step - loss: 0.1925 - accuracy:
0.9141
Epoch 3/10
4/4 [==============================] - 0s 47ms/step - loss: 0.1248 - accuracy:
0.9375
Epoch 4/10
4/4 [==============================] - 0s 39ms/step - loss: 0.0759 - accuracy:
0.9766
Epoch 5/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0517 - accuracy:
0.9922
```

```
Epoch 6/10
4/4 [==============================] - 0s 33ms/step - loss: 0.0347 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0224 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0172 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 39ms/step - loss: 0.0137 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0117 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 36ms/step - loss: 0.3239 - accuracy:
0.8672
Epoch 2/10
4/4 [==============================] - 0s 43ms/step - loss: 0.1453 - accuracy:
0.9219
Epoch 3/10
4/4 [==============================] - 0s 56ms/step - loss: 0.0902 - accuracy:
0.9453
Epoch 4/10
4/4 [==============================] - 0s 54ms/step - loss: 0.0469 - accuracy:
0.9922
Epoch 5/10
4/4 [==============================] - 0s 62ms/step - loss: 0.0299 - accuracy:
1.0000
Epoch 6/10
4/4 [==============================] - 0s 57ms/step - loss: 0.0210 - accuracy:
1.0000
Epoch 7/10
4/4 [==============================] - 0s 55ms/step - loss: 0.0156 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0121 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0104 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 36ms/step - loss: 0.0090 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 46ms/step - loss: 0.3765 - accuracy:
0.8594
```

```
Epoch 2/10
4/4 [==============================] - 0s 43ms/step - loss: 0.2793 - accuracy:
0.8906
Epoch 3/10
4/4 [==============================] - 0s 49ms/step - loss: 0.1697 - accuracy:
0.9219
Epoch 4/10
4/4 [==============================] - 0s 50ms/step - loss: 0.1042 - accuracy:
0.9609
Epoch 5/10
4/4 [==============================] - 0s 36ms/step - loss: 0.0514 - accuracy:
0.9922
Epoch 6/10
4/4 [==============================] - 0s 62ms/step - loss: 0.0329 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 59ms/step - loss: 0.0227 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0174 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0141 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0120 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 56ms/step - loss: 0.7301 - accuracy:
0.8906
Epoch 2/10
4/4 [==============================] - 0s 42ms/step - loss: 0.1709 - accuracy:
0.9453
Epoch 3/10
4/4 [==============================] - 0s 63ms/step - loss: 0.1473 - accuracy:
0.9531
Epoch 4/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0967 - accuracy:
0.9766
Epoch 5/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0642 - accuracy:
0.9922
Epoch 6/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0432 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0337 - accuracy:
0.9922
```

```
Epoch 8/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0266 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0228 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 39ms/step - loss: 0.0187 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 43ms/step - loss: 0.4287 - accuracy:
0.8203
Epoch 2/10
4/4 [==============================] - 0s 55ms/step - loss: 0.2326 - accuracy:
0.8906
Epoch 3/10
4/4 [==============================] - 0s 37ms/step - loss: 0.1518 - accuracy:
0.9375
Epoch 4/10
4/4 [==============================] - 0s 39ms/step - loss: 0.1054 - accuracy:
0.9609
Epoch 5/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0670 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0441 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 43ms/step - loss: 0.0319 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0263 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0215 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0183 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 53ms/step - loss: 0.4347 - accuracy:
0.8203
Epoch 2/10
4/4 [==============================] - 0s 43ms/step - loss: 0.2419 - accuracy:
0.8672
Epoch 3/10
4/4 [==============================] - 0s 48ms/step - loss: 0.1458 - accuracy:
0.9141
```

```
Epoch 4/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0873 - accuracy:
0.9766
Epoch 5/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0533 - accuracy:
1.0000
Epoch 6/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0353 - accuracy:
1.0000
Epoch 7/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0257 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 33ms/step - loss: 0.0193 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0155 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0131 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 38ms/step - loss: 0.3836 - accuracy:
0.8203
Epoch 2/10
4/4 [==============================] - 0s 44ms/step - loss: 0.2164 - accuracy:
0.8906
Epoch 3/10
4/4 [==============================] - 0s 31ms/step - loss: 0.1143 - accuracy:
0.9375
Epoch 4/10
4/4 [==============================] - 0s 32ms/step - loss: 0.0648 - accuracy:
0.9922
Epoch 5/10
4/4 [==============================] - 0s 53ms/step - loss: 0.0393 - accuracy:
1.0000
Epoch 6/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0283 - accuracy:
1.0000
Epoch 7/10
4/4 [==============================] - 0s 31ms/step - loss: 0.0198 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0162 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 35ms/step - loss: 0.0134 - accuracy:
1.0000
```

```
Epoch 10/10
4/4 [==============================] - 0s 27ms/step - loss: 0.0117 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 47ms/step - loss: 0.4543 - accuracy:
0.8828
Epoch 2/10
4/4 [==============================] - 0s 37ms/step - loss: 0.1971 - accuracy:
0.9297
Epoch 3/10
4/4 [==============================] - 0s 46ms/step - loss: 0.1099 - accuracy:
0.9688
Epoch 4/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0661 - accuracy:
0.9844
Epoch 5/10
4/4 [==============================] - 0s 34ms/step - loss: 0.0316 - accuracy:
1.0000
Epoch 6/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0226 - accuracy:
1.0000
Epoch 7/10
4/4 [==============================] - 0s 56ms/step - loss: 0.0155 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0122 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0104 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 31ms/step - loss: 0.0090 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 30ms/step - loss: 0.4218 - accuracy:
0.8203
Epoch 2/10
4/4 [==============================] - 0s 40ms/step - loss: 0.2770 - accuracy:
0.8750
Epoch 3/10
4/4 [==============================] - 0s 37ms/step - loss: 0.1697 - accuracy:
0.9297
Epoch 4/10
4/4 [==============================] - 0s 45ms/step - loss: 0.1012 - accuracy:
0.9531
Epoch 5/10
4/4 [==============================] - 0s 49ms/step - loss: 0.0682 - accuracy:
0.9844
```

```
Epoch 6/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0462 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0322 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0225 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0171 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 28ms/step - loss: 0.0144 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 47ms/step - loss: 0.3618 - accuracy:
0.8359
Epoch 2/10
4/4 [==============================] - 0s 40ms/step - loss: 0.2485 - accuracy:
0.8672
Epoch 3/10
4/4 [==============================] - 0s 43ms/step - loss: 0.1589 - accuracy:
0.9297
Epoch 4/10
4/4 [==============================] - 0s 50ms/step - loss: 0.1030 - accuracy:
0.9766
Epoch 5/10
4/4 [==============================] - 0s 36ms/step - loss: 0.0729 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0559 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0416 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 35ms/step - loss: 0.0340 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 28ms/step - loss: 0.0278 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0242 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 56ms/step - loss: 0.3763 - accuracy:
0.8359
```

```
Epoch 2/10
4/4 [==============================] - 0s 58ms/step - loss: 0.2557 - accuracy:
0.8594
Epoch 3/10
4/4 [==============================] - 0s 45ms/step - loss: 0.1534 - accuracy:
0.9219
Epoch 4/10
4/4 [==============================] - 0s 59ms/step - loss: 0.1029 - accuracy:
0.9609
Epoch 5/10
4/4 [==============================] - 0s 76ms/step - loss: 0.0678 - accuracy:
0.9766
Epoch 6/10
4/4 [==============================] - 0s 68ms/step - loss: 0.0481 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 54ms/step - loss: 0.0342 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 65ms/step - loss: 0.0275 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 71ms/step - loss: 0.0202 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 57ms/step - loss: 0.0159 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 42ms/step - loss: 0.3949 - accuracy:
0.8594
Epoch 2/10
4/4 [==============================] - 0s 46ms/step - loss: 0.2257 - accuracy:
0.9141
Epoch 3/10
4/4 [==============================] - 0s 40ms/step - loss: 0.1551 - accuracy:
0.9375
Epoch 4/10
4/4 [==============================] - 0s 65ms/step - loss: 0.0989 - accuracy:
0.9688
Epoch 5/10
4/4 [==============================] - 0s 57ms/step - loss: 0.0647 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 71ms/step - loss: 0.0465 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0343 - accuracy:
0.9844
```

```
Epoch 8/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0237 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0193 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0159 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 36ms/step - loss: 0.2847 - accuracy:
0.8828
Epoch 2/10
4/4 [==============================] - 0s 35ms/step - loss: 0.1835 - accuracy:
0.9297
Epoch 3/10
4/4 [==============================] - 0s 40ms/step - loss: 0.1232 - accuracy:
0.9531
Epoch 4/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0752 - accuracy:
0.9766
Epoch 5/10
4/4 [==============================] - 0s 34ms/step - loss: 0.0458 - accuracy:
0.9766
Epoch 6/10
4/4 [==============================] - 0s 35ms/step - loss: 0.0288 - accuracy:
1.0000
Epoch 7/10
4/4 [==============================] - 0s 51ms/step - loss: 0.0205 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0151 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 31ms/step - loss: 0.0122 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 39ms/step - loss: 0.0104 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 46ms/step - loss: 0.2832 - accuracy:
0.8828
Epoch 2/10
4/4 [==============================] - 0s 35ms/step - loss: 0.1407 - accuracy:
0.9531
Epoch 3/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0739 - accuracy:
0.9844
```

```
Epoch 4/10
4/4 [==============================] - 0s 55ms/step - loss: 0.0399 - accuracy:
1.0000
Epoch 5/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0234 - accuracy:
1.0000
Epoch 6/10
4/4 [==============================] - 0s 49ms/step - loss: 0.0166 - accuracy:
1.0000
Epoch 7/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0124 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0103 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0085 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 49ms/step - loss: 0.0076 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 57ms/step - loss: 0.3320 - accuracy:
0.8672
Epoch 2/10
4/4 [==============================] - 0s 49ms/step - loss: 0.1833 - accuracy:
0.9297
Epoch 3/10
4/4 [==============================] - 0s 47ms/step - loss: 0.1177 - accuracy:
0.9609
Epoch 4/10
4/4 [==============================] - 0s 60ms/step - loss: 0.0810 - accuracy:
0.9766
Epoch 5/10
4/4 [==============================] - 0s 76ms/step - loss: 0.0606 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 65ms/step - loss: 0.0464 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0352 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0242 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0200 - accuracy:
0.9922
```

```
Epoch 10/10
4/4 [==============================] - 0s 73ms/step - loss: 0.0161 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 55ms/step - loss: 0.3337 - accuracy:
0.8594
Epoch 2/10
4/4 [==============================] - 0s 45ms/step - loss: 0.1449 - accuracy:
0.9609
Epoch 3/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0798 - accuracy:
0.9766
Epoch 4/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0472 - accuracy:
0.9922
Epoch 5/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0328 - accuracy:
0.9922
Epoch 6/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0243 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 43ms/step - loss: 0.0195 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0157 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0136 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0115 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 47ms/step - loss: 0.2584 - accuracy:
0.8984
Epoch 2/10
4/4 [==============================] - 0s 39ms/step - loss: 0.1555 - accuracy:
0.9219
Epoch 3/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0796 - accuracy:
0.9688
Epoch 4/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0359 - accuracy:
0.9922
Epoch 5/10
4/4 [==============================] - 0s 63ms/step - loss: 0.0220 - accuracy:
1.0000
```

```
Epoch 6/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0156 - accuracy:
1.0000
Epoch 7/10
4/4 [==============================] - 0s 51ms/step - loss: 0.0111 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 66ms/step - loss: 0.0090 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 51ms/step - loss: 0.0075 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0067 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 42ms/step - loss: 0.1988 - accuracy:
0.8984
Epoch 2/10
4/4 [==============================] - 0s 45ms/step - loss: 0.1185 - accuracy:
0.9375
Epoch 3/10
4/4 [==============================] - 0s 35ms/step - loss: 0.0651 - accuracy:
0.9766
Epoch 4/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0367 - accuracy:
1.0000
Epoch 5/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0253 - accuracy:
1.0000
Epoch 6/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0173 - accuracy:
1.0000
Epoch 7/10
4/4 [==============================] - 0s 56ms/step - loss: 0.0133 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0100 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 51ms/step - loss: 0.0087 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0074 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 49ms/step - loss: 0.5424 - accuracy:
0.8750
```

```
Epoch 2/10
4/4 [==============================] - 0s 51ms/step - loss: 0.2109 - accuracy:
0.9375
Epoch 3/10
4/4 [==============================] - 0s 53ms/step - loss: 0.1119 - accuracy:
0.9609
Epoch 4/10
4/4 [==============================] - 0s 43ms/step - loss: 0.0751 - accuracy:
0.9922
Epoch 5/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0443 - accuracy:
0.9922
Epoch 6/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0259 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0152 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 51ms/step - loss: 0.0117 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0101 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0087 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 60ms/step - loss: 0.3925 - accuracy:
0.8438
Epoch 2/10
4/4 [==============================] - 0s 54ms/step - loss: 0.2802 - accuracy:
0.8672
Epoch 3/10
4/4 [==============================] - 0s 47ms/step - loss: 0.1787 - accuracy:
0.9219
Epoch 4/10
4/4 [==============================] - 0s 42ms/step - loss: 0.1139 - accuracy:
0.9688
Epoch 5/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0787 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0622 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0449 - accuracy:
0.9922
```

```
Epoch 8/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0337 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0284 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0249 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 50ms/step - loss: 0.3093 - accuracy:
0.8672
Epoch 2/10
4/4 [==============================] - 0s 55ms/step - loss: 0.1859 - accuracy:
0.9219
Epoch 3/10
4/4 [==============================] - 0s 50ms/step - loss: 0.1008 - accuracy:
0.9688
Epoch 4/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0666 - accuracy:
0.9766
Epoch 5/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0450 - accuracy:
0.9766
Epoch 6/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0305 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 63ms/step - loss: 0.0226 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0194 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 55ms/step - loss: 0.0169 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0141 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 50ms/step - loss: 0.5178 - accuracy:
0.8594
Epoch 2/10
4/4 [==============================] - 0s 52ms/step - loss: 0.2055 - accuracy:
0.9141
Epoch 3/10
4/4 [==============================] - 0s 35ms/step - loss: 0.1333 - accuracy:
0.9688
```

```
Epoch 4/10
4/4 [==============================] - 0s 34ms/step - loss: 0.0878 - accuracy:
0.9766
Epoch 5/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0582 - accuracy:
0.9766
Epoch 6/10
4/4 [==============================] - 0s 36ms/step - loss: 0.0396 - accuracy:
0.9766
Epoch 7/10
4/4 [==============================] - 0s 63ms/step - loss: 0.0272 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 54ms/step - loss: 0.0190 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0120 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 56ms/step - loss: 0.0105 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 33ms/step - loss: 0.3590 - accuracy:
0.8750
Epoch 2/10
4/4 [==============================] - 0s 48ms/step - loss: 0.2058 - accuracy:
0.9219
Epoch 3/10
4/4 [==============================] - 0s 49ms/step - loss: 0.1491 - accuracy:
0.9453
Epoch 4/10
4/4 [==============================] - 0s 44ms/step - loss: 0.1031 - accuracy:
0.9688
Epoch 5/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0796 - accuracy:
0.9766
Epoch 6/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0493 - accuracy:
0.9766
Epoch 7/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0364 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0268 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 39ms/step - loss: 0.0223 - accuracy:
0.9922
```

```
Epoch 10/10
4/4 [==============================] - 0s 43ms/step - loss: 0.0190 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 46ms/step - loss: 0.3762 - accuracy:
0.8125
Epoch 2/10
4/4 [==============================] - 0s 44ms/step - loss: 0.2140 - accuracy:
0.8906
Epoch 3/10
4/4 [==============================] - 0s 45ms/step - loss: 0.1135 - accuracy:
0.9688
Epoch 4/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0729 - accuracy:
0.9766
Epoch 5/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0478 - accuracy:
0.9922
Epoch 6/10
4/4 [==============================] - 0s 51ms/step - loss: 0.0335 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 72ms/step - loss: 0.0263 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0194 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 62ms/step - loss: 0.0163 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 36ms/step - loss: 0.0129 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 66ms/step - loss: 0.4195 - accuracy:
0.8125
Epoch 2/10
4/4 [==============================] - 0s 60ms/step - loss: 0.2710 - accuracy:
0.8672
Epoch 3/10
4/4 [==============================] - 0s 50ms/step - loss: 0.1407 - accuracy:
0.9375
Epoch 4/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0901 - accuracy:
0.9531
Epoch 5/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0512 - accuracy:
0.9844
```

```
Epoch 6/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0409 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0319 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 36ms/step - loss: 0.0260 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 56ms/step - loss: 0.0227 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0189 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 43ms/step - loss: 0.3592 - accuracy:
0.8594
Epoch 2/10
4/4 [==============================] - 0s 39ms/step - loss: 0.2368 - accuracy:
0.9297
Epoch 3/10
4/4 [==============================] - 0s 45ms/step - loss: 0.1594 - accuracy:
0.9531
Epoch 4/10
4/4 [==============================] - 0s 41ms/step - loss: 0.1126 - accuracy:
0.9609
Epoch 5/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0597 - accuracy:
0.9766
Epoch 6/10
4/4 [==============================] - 0s 54ms/step - loss: 0.0416 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0286 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0224 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0187 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0163 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 39ms/step - loss: 0.3183 - accuracy:
0.8516
```

```
Epoch 2/10
4/4 [==============================] - 0s 32ms/step - loss: 0.1783 - accuracy:
0.9141
Epoch 3/10
4/4 [==============================] - 0s 65ms/step - loss: 0.1135 - accuracy:
0.9453
Epoch 4/10
4/4 [==============================] - 0s 31ms/step - loss: 0.0657 - accuracy:
0.9844
Epoch 5/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0410 - accuracy:
1.0000
Epoch 6/10
4/4 [==============================] - 0s 43ms/step - loss: 0.0265 - accuracy:
1.0000
Epoch 7/10
4/4 [==============================] - 0s 43ms/step - loss: 0.0198 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0154 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 58ms/step - loss: 0.0126 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0105 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 52ms/step - loss: 0.2705 - accuracy:
0.8750
Epoch 2/10
4/4 [==============================] - 0s 40ms/step - loss: 0.1911 - accuracy:
0.9141
Epoch 3/10
4/4 [==============================] - 0s 54ms/step - loss: 0.1162 - accuracy:
0.9688
Epoch 4/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0829 - accuracy:
0.9688
Epoch 5/10
4/4 [==============================] - 0s 54ms/step - loss: 0.0505 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0310 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0201 - accuracy:
1.0000
```

```
Epoch 8/10
4/4 [==============================] - 0s 43ms/step - loss: 0.0119 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 62ms/step - loss: 0.0097 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0080 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 49ms/step - loss: 0.9650 - accuracy:
0.8438
Epoch 2/10
4/4 [==============================] - 0s 55ms/step - loss: 0.2889 - accuracy:
0.9062
Epoch 3/10
4/4 [==============================] - 0s 47ms/step - loss: 0.1584 - accuracy:
0.9531
Epoch 4/10
4/4 [==============================] - 0s 49ms/step - loss: 0.0959 - accuracy:
0.9688
Epoch 5/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0589 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 53ms/step - loss: 0.0398 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 58ms/step - loss: 0.0264 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0199 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0154 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 34ms/step - loss: 0.0128 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 49ms/step - loss: 0.3744 - accuracy:
0.8516
Epoch 2/10
4/4 [==============================] - 0s 29ms/step - loss: 0.2222 - accuracy:
0.9219
Epoch 3/10
4/4 [==============================] - 0s 33ms/step - loss: 0.1321 - accuracy:
0.9453
```

```
Epoch 4/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0848 - accuracy:
0.9688
Epoch 5/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0583 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0389 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 66ms/step - loss: 0.0303 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0250 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 49ms/step - loss: 0.0216 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0188 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 49ms/step - loss: 0.3235 - accuracy:
0.8906
Epoch 2/10
4/4 [==============================] - 0s 44ms/step - loss: 0.2302 - accuracy:
0.9219
Epoch 3/10
4/4 [==============================] - 0s 44ms/step - loss: 0.1420 - accuracy:
0.9297
Epoch 4/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0865 - accuracy:
0.9453
Epoch 5/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0571 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0391 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 35ms/step - loss: 0.0280 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 53ms/step - loss: 0.0220 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 34ms/step - loss: 0.0163 - accuracy:
1.0000
```

```
Epoch 10/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0131 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 52ms/step - loss: 0.2749 - accuracy:
0.8906
Epoch 2/10
4/4 [==============================] - 0s 49ms/step - loss: 0.1763 - accuracy:
0.9453
Epoch 3/10
4/4 [==============================] - 0s 61ms/step - loss: 0.1024 - accuracy:
0.9766
Epoch 4/10
4/4 [==============================] - 0s 51ms/step - loss: 0.0708 - accuracy:
0.9844
Epoch 5/10
4/4 [==============================] - 0s 51ms/step - loss: 0.0306 - accuracy:
0.9922
Epoch 6/10
4/4 [==============================] - 0s 43ms/step - loss: 0.0223 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0159 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0120 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 31ms/step - loss: 0.0099 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 32ms/step - loss: 0.0086 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 57ms/step - loss: 0.3106 - accuracy:
0.8984
Epoch 2/10
4/4 [==============================] - 0s 80ms/step - loss: 0.1706 - accuracy:
0.9375
Epoch 3/10
4/4 [==============================] - 0s 65ms/step - loss: 0.1128 - accuracy:
0.9453
Epoch 4/10
4/4 [==============================] - 0s 53ms/step - loss: 0.0778 - accuracy:
0.9766
Epoch 5/10
4/4 [==============================] - 0s 67ms/step - loss: 0.0495 - accuracy:
0.9766
```

```
Epoch 6/10
4/4 [==============================] - 0s 81ms/step - loss: 0.0270 - accuracy:
1.0000
Epoch 7/10
4/4 [==============================] - 0s 66ms/step - loss: 0.0193 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0155 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 49ms/step - loss: 0.0134 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0114 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 69ms/step - loss: 0.4197 - accuracy:
0.8594
Epoch 2/10
4/4 [==============================] - 0s 47ms/step - loss: 0.3025 - accuracy:
0.8984
Epoch 3/10
4/4 [==============================] - 0s 56ms/step - loss: 0.2081 - accuracy:
0.9297
Epoch 4/10
4/4 [==============================] - 0s 68ms/step - loss: 0.1280 - accuracy:
0.9531
Epoch 5/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0795 - accuracy:
0.9609
Epoch 6/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0536 - accuracy:
0.9766
Epoch 7/10
4/4 [==============================] - 0s 49ms/step - loss: 0.0360 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 79ms/step - loss: 0.0224 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 49ms/step - loss: 0.0161 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0113 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 57ms/step - loss: 0.3492 - accuracy:
0.8750
```

```
Epoch 2/10
4/4 [==============================] - 0s 50ms/step - loss: 0.1825 - accuracy:
0.9297
Epoch 3/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0884 - accuracy:
0.9766
Epoch 4/10
4/4 [==============================] - 0s 53ms/step - loss: 0.0527 - accuracy:
0.9844
Epoch 5/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0350 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 36ms/step - loss: 0.0260 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 55ms/step - loss: 0.0183 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0136 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 43ms/step - loss: 0.0113 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 53ms/step - loss: 0.0097 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 46ms/step - loss: 0.4771 - accuracy:
0.8359
Epoch 2/10
4/4 [==============================] - 0s 40ms/step - loss: 0.2062 - accuracy:
0.8984
Epoch 3/10
4/4 [==============================] - 0s 50ms/step - loss: 0.1291 - accuracy:
0.9531
Epoch 4/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0729 - accuracy:
0.9766
Epoch 5/10
4/4 [==============================] - 0s 53ms/step - loss: 0.0500 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0329 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0243 - accuracy:
0.9922
```

```
Epoch 8/10
4/4 [==============================] - 0s 74ms/step - loss: 0.0175 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 49ms/step - loss: 0.0146 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0127 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 60ms/step - loss: 0.4343 - accuracy:
0.8906
Epoch 2/10
4/4 [==============================] - 0s 57ms/step - loss: 0.2208 - accuracy:
0.9297
Epoch 3/10
4/4 [==============================] - 0s 39ms/step - loss: 0.1218 - accuracy:
0.9609
Epoch 4/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0779 - accuracy:
0.9844
Epoch 5/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0595 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 57ms/step - loss: 0.0394 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 32ms/step - loss: 0.0253 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 29ms/step - loss: 0.0185 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0117 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 54ms/step - loss: 0.0096 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 56ms/step - loss: 0.2955 - accuracy:
0.8984
Epoch 2/10
4/4 [==============================] - 0s 52ms/step - loss: 0.1662 - accuracy:
0.9531
Epoch 3/10
4/4 [==============================] - 0s 48ms/step - loss: 0.1141 - accuracy:
0.9766
```

```
Epoch 4/10
4/4 [==============================] - 0s 34ms/step - loss: 0.0527 - accuracy:
0.9844
Epoch 5/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0322 - accuracy:
0.9922
Epoch 6/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0160 - accuracy:
1.0000
Epoch 7/10
4/4 [==============================] - 0s 35ms/step - loss: 0.0102 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0085 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0071 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 27ms/step - loss: 0.0061 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 43ms/step - loss: 0.5363 - accuracy:
0.8438
Epoch 2/10
4/4 [==============================] - 0s 53ms/step - loss: 0.2445 - accuracy:
0.8984
Epoch 3/10
4/4 [==============================] - 0s 51ms/step - loss: 0.1680 - accuracy:
0.9219
Epoch 4/10
4/4 [==============================] - 0s 53ms/step - loss: 0.1103 - accuracy:
0.9609
Epoch 5/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0701 - accuracy:
0.9766
Epoch 6/10
4/4 [==============================] - 0s 36ms/step - loss: 0.0469 - accuracy:
0.9766
Epoch 7/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0291 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0223 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 51ms/step - loss: 0.0182 - accuracy:
1.0000
```

```
Epoch 10/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0145 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 53ms/step - loss: 0.4257 - accuracy:
0.8750
Epoch 2/10
4/4 [==============================] - 0s 48ms/step - loss: 0.2530 - accuracy:
0.9219
Epoch 3/10
4/4 [==============================] - 0s 50ms/step - loss: 0.1765 - accuracy:
0.9375
Epoch 4/10
4/4 [==============================] - 0s 53ms/step - loss: 0.1210 - accuracy:
0.9609
Epoch 5/10
4/4 [==============================] - 0s 49ms/step - loss: 0.0831 - accuracy:
0.9688
Epoch 6/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0580 - accuracy:
0.9766
Epoch 7/10
4/4 [==============================] - 0s 55ms/step - loss: 0.0380 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0266 - accuracy:
0.9844
Epoch 9/10
4/4 [==============================] - 0s 55ms/step - loss: 0.0216 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0181 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 43ms/step - loss: 0.1987 - accuracy:
0.9219
Epoch 2/10
4/4 [==============================] - 0s 36ms/step - loss: 0.1116 - accuracy:
0.9453
Epoch 3/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0585 - accuracy:
0.9844
Epoch 4/10
4/4 [==============================] - 0s 60ms/step - loss: 0.0338 - accuracy:
1.0000
Epoch 5/10
4/4 [==============================] - 0s 51ms/step - loss: 0.0223 - accuracy:
1.0000
```

```
Epoch 6/10
4/4 [==============================] - 0s 43ms/step - loss: 0.0151 - accuracy:
1.0000
Epoch 7/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0122 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 54ms/step - loss: 0.0099 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0086 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 61ms/step - loss: 0.0075 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 54ms/step - loss: 0.2780 - accuracy:
0.9297
Epoch 2/10
4/4 [==============================] - 0s 54ms/step - loss: 0.1580 - accuracy:
0.9609
Epoch 3/10
4/4 [==============================] - 0s 33ms/step - loss: 0.0999 - accuracy:
0.9688
Epoch 4/10
4/4 [==============================] - 0s 35ms/step - loss: 0.0563 - accuracy:
0.9688
Epoch 5/10
4/4 [==============================] - 0s 39ms/step - loss: 0.0348 - accuracy:
0.9922
Epoch 6/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0201 - accuracy:
1.0000
Epoch 7/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0141 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 54ms/step - loss: 0.0111 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0087 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0076 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 51ms/step - loss: 0.2838 - accuracy:
0.9062
```

```
Epoch 2/10
4/4 [==============================] - 0s 60ms/step - loss: 0.2006 - accuracy:
0.9375
Epoch 3/10
4/4 [==============================] - 0s 62ms/step - loss: 0.1184 - accuracy:
0.9609
Epoch 4/10
4/4 [==============================] - 0s 49ms/step - loss: 0.0819 - accuracy:
0.9688
Epoch 5/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0542 - accuracy:
0.9766
Epoch 6/10
4/4 [==============================] - 0s 31ms/step - loss: 0.0370 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 43ms/step - loss: 0.0253 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0198 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 35ms/step - loss: 0.0141 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0114 - accuracy:
1.0000
```

- Avaliando o modelo

3.19.2 Exportando o modelo Rede Neural

[289]:
```
model.save_weights('pesos/keras_finished_count_2.h5')
```

3.19.3 Carregar modelo Rede Neural

[290]:
```
# keras_model = load_model('pesos/keras_finished_count_2.h5')
keras_model = create_nn_model()
keras_model.load_weights('pesos/keras_finished_count_2.h5')
```

3.20 Avaliando as métricas do modelo Passive Agressive contains

- Avaliando a acurácia do modelo

[291]:
```
score, acc = keras_model.evaluate(x_test_batch, y_test_batch)
print("A acurácia para o modelo com Redes neurais com o dataset count sem␣
 ↪validação foi de {:.2f}%".format(acc*100))
```

```
4/4 [==============================] - 0s 24ms/step - loss: 0.3442 - accuracy:
0.8688
A acurácia para o modelo com Redes neurais com o dataset count sem validação foi
de 83.59%
```

- obtendo as predições do modelo utilizando o dataset de treino com **count**

[292]:
```
y_pred_keras = keras_model.predict(x_test.to_numpy()).round()
y_pred_keras = [classify(result) for result in y_pred_keras]
```

3.21 Classification Report

- A função `classification_report` nos dá todas as métricas importantes que podemos obter de um modelo.
- Ela irá nos dar a precisão por classe.
- O quanto de revoação (recall) o modelo tem por classe e no geral
- O f1-score geral e por classe
- A acurária do modelo
- Todas estas métricas são úteis e nos ajudarão a escolher qual o modelo se saiu melhor e pode ser utilizado para esta tarefa.

[324]:
```
report = classification_report(y_test, y_pred_keras, target_names=target_names,␣
 ↪output_dict=True)
```

[326]:
```
sgd_stats = {'model': 'keras_count',
             'accuracy': report['accuracy'],
             'recall': report['macro avg']['recall'],
             'f1_score': report['macro avg']['f1-score'],
             'support': report['macro avg']['support'],
             'falso_precision': report['Falso']['precision'],
             'falso_recall': report['Falso']['recall'],
             'falso_f1_score': report['Falso']['f1-score'],
             'verdadeiro_precision': report['Verdadeiro']['precision'],
             'verdadeiro_recall': report['Verdadeiro']['recall'],
             'verdadeiro_f1_score': report['Verdadeiro']['f1-score']}
```

[327]:
```
stats.append(sgd_stats)
```

3.22 Rede Neural com dataset contains e sem validação em treino

3.22.1 Observação o modelo já foi treinado, caso queira usar o modelo já treinado ele pode ser carregado diretamente aqui

- A célula cria o modelo de rede neural descrito

[328]:
```
model_contains = create_nn_model()
```

- A célula abaixo treina a rede neural

[329]:
```
inicio_lote = 0
fim_lote = tamanho_lote

total_lotes = str(int(len(X_train)/tamanho_lote))
ultimo_lote = False
i = 1

while True:
    x_train = formatar_lote_de_noticias_sim_nao(X_train[inicio_lote:fim_lote])
    y_tr = np.asanyarray(format_labels(y_train[inicio_lote:fim_lote].
 ↪to_numpy()))
    model_contains.fit(x_train.to_numpy(), y_tr, epochs=10)
    inicio_lote += tamanho_lote

    if fim_lote + tamanho_lote > len(X_train):
        fim_lote = len(X_train)
        ultimo_lote = True
    else:
        fim_lote += tamanho_lote

    i += 1
    if ultimo_lote:
        break
```

```
Epoch 1/10
4/4 [==============================] - 1s 41ms/step - loss: 0.6969 - accuracy:
0.6427
Epoch 2/10
4/4 [==============================] - 0s 34ms/step - loss: 0.6720 - accuracy:
0.7906
Epoch 3/10
4/4 [==============================] - 0s 22ms/step - loss: 0.6017 - accuracy:
0.9115
Epoch 4/10
4/4 [==============================] - 0s 46ms/step - loss: 0.5242 - accuracy:
0.9104
Epoch 5/10
```

```
4/4 [==============================] - 0s 48ms/step - loss: 0.4605 - accuracy:
0.8990
Epoch 6/10
4/4 [==============================] - 0s 43ms/step - loss: 0.4016 - accuracy:
0.9240
Epoch 7/10
4/4 [==============================] - 0s 40ms/step - loss: 0.3805 - accuracy:
0.9010
Epoch 8/10
4/4 [==============================] - 0s 38ms/step - loss: 0.2905 - accuracy:
0.9406
Epoch 9/10
4/4 [==============================] - 0s 37ms/step - loss: 0.2803 - accuracy:
0.9177
Epoch 10/10
4/4 [==============================] - 0s 67ms/step - loss: 0.2338 - accuracy:
0.9385
Epoch 1/10
4/4 [==============================] - 0s 23ms/step - loss: 0.4546 - accuracy:
0.8203
Epoch 2/10
4/4 [==============================] - 0s 29ms/step - loss: 0.3781 - accuracy:
0.8516
Epoch 3/10
4/4 [==============================] - 0s 25ms/step - loss: 0.3196 - accuracy:
0.8516
Epoch 4/10
4/4 [==============================] - 0s 37ms/step - loss: 0.2820 - accuracy:
0.8516
Epoch 5/10
4/4 [==============================] - 0s 44ms/step - loss: 0.2563 - accuracy:
0.8516
Epoch 6/10
4/4 [==============================] - 0s 51ms/step - loss: 0.2366 - accuracy:
0.8516
Epoch 7/10
4/4 [==============================] - 0s 49ms/step - loss: 0.2191 - accuracy:
0.8594
Epoch 8/10
4/4 [==============================] - 0s 50ms/step - loss: 0.2019 - accuracy:
0.8672
Epoch 9/10
4/4 [==============================] - 0s 53ms/step - loss: 0.1863 - accuracy:
0.8750
Epoch 10/10
4/4 [==============================] - 0s 43ms/step - loss: 0.1717 - accuracy:
0.8984
Epoch 1/10
```

```
4/4 [==============================] - 0s 32ms/step - loss: 0.4307 - accuracy:
0.8047
Epoch 2/10
4/4 [==============================] - 0s 37ms/step - loss: 0.3208 - accuracy:
0.8672
Epoch 3/10
4/4 [==============================] - 0s 56ms/step - loss: 0.2794 - accuracy:
0.8750
Epoch 4/10
4/4 [==============================] - 0s 44ms/step - loss: 0.2446 - accuracy:
0.8906
Epoch 5/10
4/4 [==============================] - 0s 65ms/step - loss: 0.2152 - accuracy:
0.8906
Epoch 6/10
4/4 [==============================] - 0s 40ms/step - loss: 0.1917 - accuracy:
0.9062
Epoch 7/10
4/4 [==============================] - 0s 45ms/step - loss: 0.1740 - accuracy:
0.9297
Epoch 8/10
4/4 [==============================] - 0s 47ms/step - loss: 0.1548 - accuracy:
0.9375
Epoch 9/10
4/4 [==============================] - 0s 39ms/step - loss: 0.1405 - accuracy:
0.9531
Epoch 10/10
4/4 [==============================] - 0s 34ms/step - loss: 0.1291 - accuracy:
0.9688
Epoch 1/10
4/4 [==============================] - 0s 47ms/step - loss: 0.2913 - accuracy:
0.8828
Epoch 2/10
4/4 [==============================] - 0s 38ms/step - loss: 0.2091 - accuracy:
0.9062
Epoch 3/10
4/4 [==============================] - 0s 37ms/step - loss: 0.1617 - accuracy:
0.9141
Epoch 4/10
4/4 [==============================] - 0s 66ms/step - loss: 0.1334 - accuracy:
0.9297
Epoch 5/10
4/4 [==============================] - 0s 45ms/step - loss: 0.1104 - accuracy:
0.9375
Epoch 6/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0961 - accuracy:
0.9688
Epoch 7/10
```

```
4/4 [==============================] - 0s 41ms/step - loss: 0.0836 - accuracy:
0.9688
Epoch 8/10
4/4 [==============================] - 0s 33ms/step - loss: 0.0745 - accuracy:
0.9844
Epoch 9/10
4/4 [==============================] - 0s 60ms/step - loss: 0.0676 - accuracy:
0.9844
Epoch 10/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0617 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 45ms/step - loss: 0.3729 - accuracy:
0.8438
Epoch 2/10
4/4 [==============================] - 0s 53ms/step - loss: 0.2224 - accuracy:
0.9141
Epoch 3/10
4/4 [==============================] - 0s 58ms/step - loss: 0.1787 - accuracy:
0.9375
Epoch 4/10
4/4 [==============================] - 0s 35ms/step - loss: 0.1462 - accuracy:
0.9453
Epoch 5/10
4/4 [==============================] - 0s 34ms/step - loss: 0.1222 - accuracy:
0.9531
Epoch 6/10
4/4 [==============================] - 0s 47ms/step - loss: 0.1050 - accuracy:
0.9609
Epoch 7/10
4/4 [==============================] - 0s 55ms/step - loss: 0.0914 - accuracy:
0.9688
Epoch 8/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0807 - accuracy:
0.9766
Epoch 9/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0727 - accuracy:
0.9844
Epoch 10/10
4/4 [==============================] - 0s 64ms/step - loss: 0.0657 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 60ms/step - loss: 0.2530 - accuracy:
0.8828
Epoch 2/10
4/4 [==============================] - 0s 33ms/step - loss: 0.1945 - accuracy:
0.9219
Epoch 3/10
```

```
4/4 [==============================] - 0s 42ms/step - loss: 0.1586 - accuracy:
0.9219
Epoch 4/10
4/4 [==============================] - 0s 35ms/step - loss: 0.1315 - accuracy:
0.9219
Epoch 5/10
4/4 [==============================] - 0s 63ms/step - loss: 0.1101 - accuracy:
0.9609
Epoch 6/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0921 - accuracy:
0.9766
Epoch 7/10
4/4 [==============================] - 0s 53ms/step - loss: 0.0795 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0703 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0631 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 66ms/step - loss: 0.0577 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 41ms/step - loss: 0.3719 - accuracy:
0.8594
Epoch 2/10
4/4 [==============================] - 0s 43ms/step - loss: 0.2286 - accuracy:
0.9062
Epoch 3/10
4/4 [==============================] - 0s 31ms/step - loss: 0.1899 - accuracy:
0.9062
Epoch 4/10
4/4 [==============================] - 0s 32ms/step - loss: 0.1544 - accuracy:
0.9141
Epoch 5/10
4/4 [==============================] - 0s 28ms/step - loss: 0.1221 - accuracy:
0.9375
Epoch 6/10
4/4 [==============================] - 0s 53ms/step - loss: 0.0926 - accuracy:
0.9766
Epoch 7/10
4/4 [==============================] - 0s 43ms/step - loss: 0.0776 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 31ms/step - loss: 0.0686 - accuracy:
1.0000
Epoch 9/10
```

```
4/4 [==============================] - 0s 32ms/step - loss: 0.0605 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 34ms/step - loss: 0.0538 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 38ms/step - loss: 0.3546 - accuracy:
0.8516
Epoch 2/10
4/4 [==============================] - 0s 42ms/step - loss: 0.2405 - accuracy:
0.8984
Epoch 3/10
4/4 [==============================] - 0s 53ms/step - loss: 0.1965 - accuracy:
0.9062
Epoch 4/10
4/4 [==============================] - 0s 47ms/step - loss: 0.1479 - accuracy:
0.9375
Epoch 5/10
4/4 [==============================] - 0s 44ms/step - loss: 0.1173 - accuracy:
0.9609
Epoch 6/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0942 - accuracy:
0.9766
Epoch 7/10
4/4 [==============================] - 0s 33ms/step - loss: 0.0788 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 58ms/step - loss: 0.0690 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 36ms/step - loss: 0.0611 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 60ms/step - loss: 0.0549 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 36ms/step - loss: 0.6668 - accuracy:
0.7812
Epoch 2/10
4/4 [==============================] - 0s 40ms/step - loss: 0.3503 - accuracy:
0.8516
Epoch 3/10
4/4 [==============================] - 0s 40ms/step - loss: 0.2263 - accuracy:
0.8984
Epoch 4/10
4/4 [==============================] - 0s 36ms/step - loss: 0.1584 - accuracy:
0.9219
Epoch 5/10
```

```
4/4 [==============================] - 0s 35ms/step - loss: 0.1192 - accuracy:
0.9688
Epoch 6/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0945 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 34ms/step - loss: 0.0832 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0753 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0690 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 53ms/step - loss: 0.0631 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 48ms/step - loss: 0.3909 - accuracy:
0.8281
Epoch 2/10
4/4 [==============================] - 0s 48ms/step - loss: 0.2069 - accuracy:
0.8906
Epoch 3/10
4/4 [==============================] - 0s 53ms/step - loss: 0.1533 - accuracy:
0.9219
Epoch 4/10
4/4 [==============================] - 0s 59ms/step - loss: 0.1177 - accuracy:
0.9531
Epoch 5/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0969 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 56ms/step - loss: 0.0819 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0731 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0669 - accuracy:
0.9844
Epoch 9/10
4/4 [==============================] - 0s 43ms/step - loss: 0.0619 - accuracy:
0.9844
Epoch 10/10
4/4 [==============================] - 0s 53ms/step - loss: 0.0580 - accuracy:
0.9844
Epoch 1/10
```

```
4/4 [==============================] - 0s 60ms/step - loss: 0.4171 - accuracy:
0.8359
Epoch 2/10
4/4 [==============================] - 0s 39ms/step - loss: 0.2753 - accuracy:
0.8750
Epoch 3/10
4/4 [==============================] - 0s 55ms/step - loss: 0.1991 - accuracy:
0.9297
Epoch 4/10
4/4 [==============================] - 0s 48ms/step - loss: 0.1320 - accuracy:
0.9531
Epoch 5/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0979 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 30ms/step - loss: 0.0779 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 65ms/step - loss: 0.0668 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0589 - accuracy:
0.9844
Epoch 9/10
4/4 [==============================] - 0s 53ms/step - loss: 0.0528 - accuracy:
0.9844
Epoch 10/10
4/4 [==============================] - 0s 43ms/step - loss: 0.0478 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 41ms/step - loss: 0.3817 - accuracy:
0.8281
Epoch 2/10
4/4 [==============================] - 0s 47ms/step - loss: 0.1988 - accuracy:
0.9297
Epoch 3/10
4/4 [==============================] - 0s 58ms/step - loss: 0.1480 - accuracy:
0.9297
Epoch 4/10
4/4 [==============================] - 0s 52ms/step - loss: 0.1135 - accuracy:
0.9531
Epoch 5/10
4/4 [==============================] - 0s 49ms/step - loss: 0.0894 - accuracy:
0.9609
Epoch 6/10
4/4 [==============================] - 0s 59ms/step - loss: 0.0708 - accuracy:
0.9688
Epoch 7/10
```

```
4/4 [==============================] - 0s 38ms/step - loss: 0.0585 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0485 - accuracy:
0.9844
Epoch 9/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0413 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0361 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 47ms/step - loss: 0.5544 - accuracy:
0.7500
Epoch 2/10
4/4 [==============================] - 0s 36ms/step - loss: 0.2795 - accuracy:
0.8438
Epoch 3/10
4/4 [==============================] - 0s 45ms/step - loss: 0.2075 - accuracy:
0.8984
Epoch 4/10
4/4 [==============================] - 0s 45ms/step - loss: 0.1597 - accuracy:
0.9531
Epoch 5/10
4/4 [==============================] - 0s 48ms/step - loss: 0.1228 - accuracy:
0.9609
Epoch 6/10
4/4 [==============================] - 0s 46ms/step - loss: 0.1013 - accuracy:
0.9766
Epoch 7/10
4/4 [==============================] - 0s 49ms/step - loss: 0.0837 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0718 - accuracy:
0.9844
Epoch 9/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0641 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0554 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 50ms/step - loss: 0.4058 - accuracy:
0.8516
Epoch 2/10
4/4 [==============================] - 0s 54ms/step - loss: 0.1822 - accuracy:
0.9219
Epoch 3/10
```

```
4/4 [==============================] - 0s 45ms/step - loss: 0.1186 - accuracy:
0.9531
Epoch 4/10
4/4 [==============================] - 0s 70ms/step - loss: 0.0885 - accuracy:
0.9844
Epoch 5/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0636 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0527 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0435 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0349 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 31ms/step - loss: 0.0304 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 39ms/step - loss: 0.0258 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 33ms/step - loss: 0.4583 - accuracy:
0.8281
Epoch 2/10
4/4 [==============================] - 0s 27ms/step - loss: 0.2749 - accuracy:
0.8750
Epoch 3/10
4/4 [==============================] - 0s 46ms/step - loss: 0.1876 - accuracy:
0.8906
Epoch 4/10
4/4 [==============================] - 0s 44ms/step - loss: 0.1254 - accuracy:
0.9531
Epoch 5/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0961 - accuracy:
0.9609
Epoch 6/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0777 - accuracy:
0.9766
Epoch 7/10
4/4 [==============================] - 0s 35ms/step - loss: 0.0626 - accuracy:
0.9766
Epoch 8/10
4/4 [==============================] - 0s 43ms/step - loss: 0.0515 - accuracy:
0.9844
Epoch 9/10
```

```
4/4 [==============================] - 0s 54ms/step - loss: 0.0445 - accuracy:
0.9844
Epoch 10/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0400 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 29ms/step - loss: 0.3055 - accuracy:
0.8984
Epoch 2/10
4/4 [==============================] - 0s 52ms/step - loss: 0.1470 - accuracy:
0.9375
Epoch 3/10
4/4 [==============================] - 0s 40ms/step - loss: 0.1001 - accuracy:
0.9531
Epoch 4/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0767 - accuracy:
0.9688
Epoch 5/10
4/4 [==============================] - 0s 65ms/step - loss: 0.0619 - accuracy:
0.9766
Epoch 6/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0488 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 58ms/step - loss: 0.0372 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 33ms/step - loss: 0.0315 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0258 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0226 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 52ms/step - loss: 0.4038 - accuracy:
0.8984
Epoch 2/10
4/4 [==============================] - 0s 48ms/step - loss: 0.1815 - accuracy:
0.9375
Epoch 3/10
4/4 [==============================] - 0s 46ms/step - loss: 0.1005 - accuracy:
0.9531
Epoch 4/10
4/4 [==============================] - 0s 35ms/step - loss: 0.0700 - accuracy:
0.9844
Epoch 5/10
```

```
4/4 [==============================] - 0s 44ms/step - loss: 0.0484 - accuracy:
0.9922
Epoch 6/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0380 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0296 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 30ms/step - loss: 0.0251 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 30ms/step - loss: 0.0217 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0191 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 40ms/step - loss: 0.5594 - accuracy:
0.8047
Epoch 2/10
4/4 [==============================] - 0s 53ms/step - loss: 0.3061 - accuracy:
0.8750
Epoch 3/10
4/4 [==============================] - 0s 46ms/step - loss: 0.1516 - accuracy:
0.9219
Epoch 4/10
4/4 [==============================] - 0s 31ms/step - loss: 0.1119 - accuracy:
0.9609
Epoch 5/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0725 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 35ms/step - loss: 0.0573 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0412 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 29ms/step - loss: 0.0346 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0308 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0277 - accuracy:
1.0000
Epoch 1/10
```

```
4/4 [==============================] - 0s 52ms/step - loss: 0.5121 - accuracy:
0.8281
Epoch 2/10
4/4 [==============================] - 0s 45ms/step - loss: 0.2525 - accuracy:
0.8984
Epoch 3/10
4/4 [==============================] - 0s 51ms/step - loss: 0.1309 - accuracy:
0.9453
Epoch 4/10
4/4 [==============================] - 0s 60ms/step - loss: 0.0884 - accuracy:
0.9688
Epoch 5/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0599 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0451 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 36ms/step - loss: 0.0325 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0265 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0221 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0195 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 50ms/step - loss: 0.4606 - accuracy:
0.8750
Epoch 2/10
4/4 [==============================] - 0s 47ms/step - loss: 0.2205 - accuracy:
0.9375
Epoch 3/10
4/4 [==============================] - 0s 48ms/step - loss: 0.1176 - accuracy:
0.9688
Epoch 4/10
4/4 [==============================] - 0s 35ms/step - loss: 0.0869 - accuracy:
0.9688
Epoch 5/10
4/4 [==============================] - 0s 51ms/step - loss: 0.0703 - accuracy:
0.9688
Epoch 6/10
4/4 [==============================] - 0s 43ms/step - loss: 0.0546 - accuracy:
0.9844
Epoch 7/10
```

```
4/4 [==============================] - 0s 42ms/step - loss: 0.0402 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 61ms/step - loss: 0.0318 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0256 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 68ms/step - loss: 0.0221 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 50ms/step - loss: 0.3872 - accuracy:
0.8672
Epoch 2/10
4/4 [==============================] - 0s 47ms/step - loss: 0.2808 - accuracy:
0.9062
Epoch 3/10
4/4 [==============================] - 0s 42ms/step - loss: 0.2213 - accuracy:
0.9219
Epoch 4/10
4/4 [==============================] - 0s 49ms/step - loss: 0.1682 - accuracy:
0.9375
Epoch 5/10
4/4 [==============================] - 0s 60ms/step - loss: 0.1258 - accuracy:
0.9688
Epoch 6/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0958 - accuracy:
0.9688
Epoch 7/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0799 - accuracy:
0.9688
Epoch 8/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0612 - accuracy:
0.9766
Epoch 9/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0534 - accuracy:
0.9766
Epoch 10/10
4/4 [==============================] - 0s 53ms/step - loss: 0.0438 - accuracy:
0.9844
Epoch 1/10
4/4 [==============================] - 0s 56ms/step - loss: 0.3195 - accuracy:
0.8672
Epoch 2/10
4/4 [==============================] - 0s 44ms/step - loss: 0.1799 - accuracy:
0.9219
Epoch 3/10
```

```
4/4 [==============================] - 0s 49ms/step - loss: 0.1155 - accuracy:
0.9609
Epoch 4/10
4/4 [==============================] - 0s 31ms/step - loss: 0.0883 - accuracy:
0.9688
Epoch 5/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0646 - accuracy:
0.9766
Epoch 6/10
4/4 [==============================] - 0s 67ms/step - loss: 0.0445 - accuracy:
0.9766
Epoch 7/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0312 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0233 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 43ms/step - loss: 0.0179 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0153 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 53ms/step - loss: 0.4982 - accuracy:
0.8047
Epoch 2/10
4/4 [==============================] - 0s 32ms/step - loss: 0.3100 - accuracy:
0.8516
Epoch 3/10
4/4 [==============================] - 0s 49ms/step - loss: 0.2378 - accuracy:
0.9062
Epoch 4/10
4/4 [==============================] - 0s 54ms/step - loss: 0.1804 - accuracy:
0.9219
Epoch 5/10
4/4 [==============================] - 0s 51ms/step - loss: 0.1436 - accuracy:
0.9297
Epoch 6/10
4/4 [==============================] - 0s 39ms/step - loss: 0.1148 - accuracy:
0.9453
Epoch 7/10
4/4 [==============================] - 0s 39ms/step - loss: 0.0926 - accuracy:
0.9688
Epoch 8/10
4/4 [==============================] - 0s 49ms/step - loss: 0.0773 - accuracy:
0.9766
Epoch 9/10
```

```
4/4 [==============================] - 0s 43ms/step - loss: 0.0628 - accuracy:
0.9844
Epoch 10/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0512 - accuracy:
0.9844
Epoch 1/10
4/4 [==============================] - 0s 52ms/step - loss: 0.6042 - accuracy:
0.7734
Epoch 2/10
4/4 [==============================] - 0s 58ms/step - loss: 0.3156 - accuracy:
0.8672
Epoch 3/10
4/4 [==============================] - 0s 49ms/step - loss: 0.2260 - accuracy:
0.8984
Epoch 4/10
4/4 [==============================] - 0s 53ms/step - loss: 0.1512 - accuracy:
0.9297
Epoch 5/10
4/4 [==============================] - 0s 56ms/step - loss: 0.1189 - accuracy:
0.9688
Epoch 6/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0972 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 51ms/step - loss: 0.0832 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0722 - accuracy:
0.9844
Epoch 9/10
4/4 [==============================] - 0s 55ms/step - loss: 0.0647 - accuracy:
0.9844
Epoch 10/10
4/4 [==============================] - 0s 54ms/step - loss: 0.0574 - accuracy:
0.9844
Epoch 1/10
4/4 [==============================] - 0s 48ms/step - loss: 0.4436 - accuracy:
0.7812
Epoch 2/10
4/4 [==============================] - 0s 42ms/step - loss: 0.2249 - accuracy:
0.8750
Epoch 3/10
4/4 [==============================] - 0s 41ms/step - loss: 0.1726 - accuracy:
0.8828
Epoch 4/10
4/4 [==============================] - 0s 45ms/step - loss: 0.1294 - accuracy:
0.9297
Epoch 5/10
```

```
4/4 [==============================] - 0s 45ms/step - loss: 0.1055 - accuracy:
0.9453
Epoch 6/10
4/4 [==============================] - 0s 63ms/step - loss: 0.0801 - accuracy:
0.9609
Epoch 7/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0672 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0583 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0511 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0455 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 60ms/step - loss: 0.2885 - accuracy:
0.8906
Epoch 2/10
4/4 [==============================] - 0s 58ms/step - loss: 0.1770 - accuracy:
0.9297
Epoch 3/10
4/4 [==============================] - 0s 37ms/step - loss: 0.1373 - accuracy:
0.9453
Epoch 4/10
4/4 [==============================] - 0s 49ms/step - loss: 0.1104 - accuracy:
0.9609
Epoch 5/10
4/4 [==============================] - 0s 53ms/step - loss: 0.0855 - accuracy:
0.9609
Epoch 6/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0672 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0549 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0475 - accuracy:
0.9844
Epoch 9/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0417 - accuracy:
0.9844
Epoch 10/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0367 - accuracy:
0.9844
Epoch 1/10
```

```
4/4 [==============================] - 0s 56ms/step - loss: 0.5033 - accuracy:
0.7656
Epoch 2/10
4/4 [==============================] - 0s 50ms/step - loss: 0.3326 - accuracy:
0.8125
Epoch 3/10
4/4 [==============================] - 0s 58ms/step - loss: 0.2170 - accuracy:
0.8672
Epoch 4/10
4/4 [==============================] - 0s 50ms/step - loss: 0.1637 - accuracy:
0.9062
Epoch 5/10
4/4 [==============================] - 0s 56ms/step - loss: 0.1284 - accuracy:
0.9453
Epoch 6/10
4/4 [==============================] - 0s 42ms/step - loss: 0.1053 - accuracy:
0.9531
Epoch 7/10
4/4 [==============================] - 0s 51ms/step - loss: 0.0881 - accuracy:
0.9688
Epoch 8/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0749 - accuracy:
0.9844
Epoch 9/10
4/4 [==============================] - 0s 43ms/step - loss: 0.0648 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 53ms/step - loss: 0.0568 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 41ms/step - loss: 0.4532 - accuracy:
0.8359
Epoch 2/10
4/4 [==============================] - 0s 59ms/step - loss: 0.1831 - accuracy:
0.9062
Epoch 3/10
4/4 [==============================] - 0s 42ms/step - loss: 0.1263 - accuracy:
0.9531
Epoch 4/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0934 - accuracy:
0.9688
Epoch 5/10
4/4 [==============================] - 0s 73ms/step - loss: 0.0753 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 51ms/step - loss: 0.0603 - accuracy:
0.9844
Epoch 7/10
```

```
4/4 [==============================] - 0s 37ms/step - loss: 0.0508 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0433 - accuracy:
0.9844
Epoch 9/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0374 - accuracy:
0.9844
Epoch 10/10
4/4 [==============================] - 0s 56ms/step - loss: 0.0331 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 35ms/step - loss: 0.3107 - accuracy:
0.8828
Epoch 2/10
4/4 [==============================] - 0s 67ms/step - loss: 0.1610 - accuracy:
0.9453
Epoch 3/10
4/4 [==============================] - 0s 34ms/step - loss: 0.1299 - accuracy:
0.9531
Epoch 4/10
4/4 [==============================] - 0s 50ms/step - loss: 0.1001 - accuracy:
0.9531
Epoch 5/10
4/4 [==============================] - 0s 33ms/step - loss: 0.0808 - accuracy:
0.9766
Epoch 6/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0661 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0571 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0484 - accuracy:
0.9844
Epoch 9/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0429 - accuracy:
0.9844
Epoch 10/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0372 - accuracy:
0.9844
Epoch 1/10
4/4 [==============================] - 0s 55ms/step - loss: 0.3612 - accuracy:
0.8438
Epoch 2/10
4/4 [==============================] - 0s 60ms/step - loss: 0.2270 - accuracy:
0.8906
Epoch 3/10
```

```
4/4 [==============================] - 0s 48ms/step - loss: 0.1650 - accuracy:
0.9219
Epoch 4/10
4/4 [==============================] - 0s 59ms/step - loss: 0.1252 - accuracy:
0.9375
Epoch 5/10
4/4 [==============================] - 0s 53ms/step - loss: 0.0929 - accuracy:
0.9531
Epoch 6/10
4/4 [==============================] - 0s 36ms/step - loss: 0.0753 - accuracy:
0.9688
Epoch 7/10
4/4 [==============================] - 0s 62ms/step - loss: 0.0609 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0518 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0454 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0386 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 39ms/step - loss: 0.3000 - accuracy:
0.8750
Epoch 2/10
4/4 [==============================] - 0s 46ms/step - loss: 0.1894 - accuracy:
0.9141
Epoch 3/10
4/4 [==============================] - 0s 55ms/step - loss: 0.1421 - accuracy:
0.9297
Epoch 4/10
4/4 [==============================] - 0s 67ms/step - loss: 0.1109 - accuracy:
0.9453
Epoch 5/10
4/4 [==============================] - 0s 49ms/step - loss: 0.0881 - accuracy:
0.9766
Epoch 6/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0696 - accuracy:
0.9766
Epoch 7/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0553 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0466 - accuracy:
0.9922
Epoch 9/10
```

```
4/4 [==============================] - 0s 42ms/step - loss: 0.0405 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 25ms/step - loss: 0.0367 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 48ms/step - loss: 0.3580 - accuracy:
0.8672
Epoch 2/10
4/4 [==============================] - 0s 59ms/step - loss: 0.2250 - accuracy:
0.9062
Epoch 3/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0917 - accuracy:
0.9766
Epoch 4/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0703 - accuracy:
0.9844
Epoch 5/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0567 - accuracy:
0.9922
Epoch 6/10
4/4 [==============================] - 0s 58ms/step - loss: 0.0453 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 58ms/step - loss: 0.0389 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0316 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 53ms/step - loss: 0.0284 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 63ms/step - loss: 0.0239 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 62ms/step - loss: 0.3874 - accuracy:
0.7969
Epoch 2/10
4/4 [==============================] - 0s 53ms/step - loss: 0.2266 - accuracy:
0.8750
Epoch 3/10
4/4 [==============================] - 0s 53ms/step - loss: 0.1571 - accuracy:
0.9297
Epoch 4/10
4/4 [==============================] - 0s 54ms/step - loss: 0.1126 - accuracy:
0.9531
Epoch 5/10
```

```
4/4 [==============================] - 0s 42ms/step - loss: 0.0811 - accuracy:
0.9688
Epoch 6/10
4/4 [==============================] - 0s 43ms/step - loss: 0.0590 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0485 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0419 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 31ms/step - loss: 0.0361 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 32ms/step - loss: 0.0312 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 33ms/step - loss: 0.4041 - accuracy:
0.8828
Epoch 2/10
4/4 [==============================] - 0s 50ms/step - loss: 0.1582 - accuracy:
0.9453
Epoch 3/10
4/4 [==============================] - 0s 55ms/step - loss: 0.1197 - accuracy:
0.9453
Epoch 4/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0892 - accuracy:
0.9844
Epoch 5/10
4/4 [==============================] - 0s 59ms/step - loss: 0.0639 - accuracy:
0.9922
Epoch 6/10
4/4 [==============================] - 0s 29ms/step - loss: 0.0478 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0398 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0314 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0262 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0233 - accuracy:
0.9922
Epoch 1/10
```

```
4/4 [==============================] - 0s 52ms/step - loss: 0.3728 - accuracy:
0.8281
Epoch 2/10
4/4 [==============================] - 0s 61ms/step - loss: 0.2159 - accuracy:
0.8828
Epoch 3/10
4/4 [==============================] - 0s 39ms/step - loss: 0.1621 - accuracy:
0.9141
Epoch 4/10
4/4 [==============================] - 0s 44ms/step - loss: 0.1276 - accuracy:
0.9453
Epoch 5/10
4/4 [==============================] - 0s 27ms/step - loss: 0.0940 - accuracy:
0.9609
Epoch 6/10
4/4 [==============================] - 0s 62ms/step - loss: 0.0668 - accuracy:
0.9766
Epoch 7/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0496 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 56ms/step - loss: 0.0411 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 51ms/step - loss: 0.0346 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 56ms/step - loss: 0.0307 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 58ms/step - loss: 0.3660 - accuracy:
0.8672
Epoch 2/10
4/4 [==============================] - 0s 56ms/step - loss: 0.2688 - accuracy:
0.8906
Epoch 3/10
4/4 [==============================] - 0s 52ms/step - loss: 0.1681 - accuracy:
0.9062
Epoch 4/10
4/4 [==============================] - 0s 49ms/step - loss: 0.1281 - accuracy:
0.9297
Epoch 5/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0899 - accuracy:
0.9688
Epoch 6/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0714 - accuracy:
0.9922
Epoch 7/10
```

```
4/4 [==============================] - 0s 44ms/step - loss: 0.0592 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 60ms/step - loss: 0.0516 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0454 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0408 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 53ms/step - loss: 0.4594 - accuracy:
0.8047
Epoch 2/10
4/4 [==============================] - 0s 65ms/step - loss: 0.2825 - accuracy:
0.8672
Epoch 3/10
4/4 [==============================] - 0s 33ms/step - loss: 0.1987 - accuracy:
0.9141
Epoch 4/10
4/4 [==============================] - 0s 42ms/step - loss: 0.1446 - accuracy:
0.9297
Epoch 5/10
4/4 [==============================] - 0s 52ms/step - loss: 0.1058 - accuracy:
0.9531
Epoch 6/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0786 - accuracy:
0.9688
Epoch 7/10
4/4 [==============================] - 0s 54ms/step - loss: 0.0617 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 33ms/step - loss: 0.0506 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 34ms/step - loss: 0.0438 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0385 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 47ms/step - loss: 0.4749 - accuracy:
0.7891
Epoch 2/10
4/4 [==============================] - 0s 53ms/step - loss: 0.2218 - accuracy:
0.8828
Epoch 3/10
```

```
4/4 [==============================] - 0s 47ms/step - loss: 0.1451 - accuracy:
0.9219
Epoch 4/10
4/4 [==============================] - 0s 56ms/step - loss: 0.0998 - accuracy:
0.9688
Epoch 5/10
4/4 [==============================] - 0s 66ms/step - loss: 0.0740 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0549 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 30ms/step - loss: 0.0446 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0374 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0317 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 34ms/step - loss: 0.0280 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 42ms/step - loss: 0.3000 - accuracy:
0.8438
Epoch 2/10
4/4 [==============================] - 0s 55ms/step - loss: 0.2371 - accuracy:
0.8906
Epoch 3/10
4/4 [==============================] - 0s 67ms/step - loss: 0.1815 - accuracy:
0.9297
Epoch 4/10
4/4 [==============================] - 0s 41ms/step - loss: 0.1265 - accuracy:
0.9297
Epoch 5/10
4/4 [==============================] - 0s 39ms/step - loss: 0.0906 - accuracy:
0.9766
Epoch 6/10
4/4 [==============================] - 0s 63ms/step - loss: 0.0695 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 64ms/step - loss: 0.0559 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 57ms/step - loss: 0.0465 - accuracy:
1.0000
Epoch 9/10
```

```
4/4 [==============================] - 0s 40ms/step - loss: 0.0394 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 53ms/step - loss: 0.0339 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 42ms/step - loss: 0.4639 - accuracy:
0.8047
Epoch 2/10
4/4 [==============================] - 0s 42ms/step - loss: 0.2680 - accuracy:
0.8750
Epoch 3/10
4/4 [==============================] - 0s 49ms/step - loss: 0.1873 - accuracy:
0.8984
Epoch 4/10
4/4 [==============================] - 0s 43ms/step - loss: 0.1330 - accuracy:
0.9375
Epoch 5/10
4/4 [==============================] - 0s 40ms/step - loss: 0.1024 - accuracy:
0.9609
Epoch 6/10
4/4 [==============================] - 0s 53ms/step - loss: 0.0808 - accuracy:
0.9609
Epoch 7/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0662 - accuracy:
0.9766
Epoch 8/10
4/4 [==============================] - 0s 34ms/step - loss: 0.0570 - accuracy:
0.9844
Epoch 9/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0515 - accuracy:
0.9844
Epoch 10/10
4/4 [==============================] - 0s 61ms/step - loss: 0.0467 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 47ms/step - loss: 0.6075 - accuracy:
0.7969
Epoch 2/10
4/4 [==============================] - 0s 42ms/step - loss: 0.2632 - accuracy:
0.8828
Epoch 3/10
4/4 [==============================] - 0s 54ms/step - loss: 0.1737 - accuracy:
0.9297
Epoch 4/10
4/4 [==============================] - 0s 49ms/step - loss: 0.1330 - accuracy:
0.9453
Epoch 5/10
```

```
4/4 [==============================] - 0s 48ms/step - loss: 0.1031 - accuracy:
0.9453
Epoch 6/10
4/4 [==============================] - 0s 58ms/step - loss: 0.0849 - accuracy:
0.9609
Epoch 7/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0725 - accuracy:
0.9766
Epoch 8/10
4/4 [==============================] - 0s 55ms/step - loss: 0.0615 - accuracy:
0.9766
Epoch 9/10
4/4 [==============================] - 0s 58ms/step - loss: 0.0544 - accuracy:
0.9844
Epoch 10/10
4/4 [==============================] - 0s 61ms/step - loss: 0.0475 - accuracy:
0.9844
Epoch 1/10
4/4 [==============================] - 0s 59ms/step - loss: 0.3892 - accuracy:
0.8125
Epoch 2/10
4/4 [==============================] - 0s 37ms/step - loss: 0.2728 - accuracy:
0.8828
Epoch 3/10
4/4 [==============================] - 0s 41ms/step - loss: 0.2068 - accuracy:
0.9141
Epoch 4/10
4/4 [==============================] - 0s 31ms/step - loss: 0.1729 - accuracy:
0.9297
Epoch 5/10
4/4 [==============================] - 0s 26ms/step - loss: 0.1323 - accuracy:
0.9531
Epoch 6/10
4/4 [==============================] - 0s 13ms/step - loss: 0.1055 - accuracy:
0.9609
Epoch 7/10
4/4 [==============================] - 0s 16ms/step - loss: 0.0835 - accuracy:
0.9609
Epoch 8/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0702 - accuracy:
0.9766
Epoch 9/10
4/4 [==============================] - 0s 39ms/step - loss: 0.0561 - accuracy:
0.9844
Epoch 10/10
4/4 [==============================] - 0s 36ms/step - loss: 0.0493 - accuracy:
1.0000
Epoch 1/10
```

```
4/4 [==============================] - 0s 34ms/step - loss: 0.2596 - accuracy:
0.8672
Epoch 2/10
4/4 [==============================] - 0s 52ms/step - loss: 0.1906 - accuracy:
0.8984
Epoch 3/10
4/4 [==============================] - 0s 47ms/step - loss: 0.1529 - accuracy:
0.9531
Epoch 4/10
4/4 [==============================] - 0s 44ms/step - loss: 0.1146 - accuracy:
0.9688
Epoch 5/10
4/4 [==============================] - 0s 35ms/step - loss: 0.0931 - accuracy:
0.9766
Epoch 6/10
4/4 [==============================] - 0s 36ms/step - loss: 0.0749 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0593 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 57ms/step - loss: 0.0511 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 31ms/step - loss: 0.0393 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0329 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 38ms/step - loss: 0.4817 - accuracy:
0.8516
Epoch 2/10
4/4 [==============================] - 0s 37ms/step - loss: 0.2109 - accuracy:
0.9297
Epoch 3/10
4/4 [==============================] - 0s 24ms/step - loss: 0.1311 - accuracy:
0.9766
Epoch 4/10
4/4 [==============================] - 0s 28ms/step - loss: 0.0672 - accuracy:
0.9922
Epoch 5/10
4/4 [==============================] - 0s 49ms/step - loss: 0.0437 - accuracy:
1.0000
Epoch 6/10
4/4 [==============================] - 0s 29ms/step - loss: 0.0336 - accuracy:
1.0000
Epoch 7/10
```

```
4/4 [==============================] - 0s 42ms/step - loss: 0.0270 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 32ms/step - loss: 0.0226 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 35ms/step - loss: 0.0193 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0171 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 35ms/step - loss: 0.4679 - accuracy:
0.8281
Epoch 2/10
4/4 [==============================] - 0s 41ms/step - loss: 0.2234 - accuracy:
0.8906
Epoch 3/10
4/4 [==============================] - 0s 37ms/step - loss: 0.1585 - accuracy:
0.9375
Epoch 4/10
4/4 [==============================] - 0s 67ms/step - loss: 0.1244 - accuracy:
0.9531
Epoch 5/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0970 - accuracy:
0.9688
Epoch 6/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0779 - accuracy:
0.9766
Epoch 7/10
4/4 [==============================] - 0s 49ms/step - loss: 0.0628 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 59ms/step - loss: 0.0550 - accuracy:
0.9844
Epoch 9/10
4/4 [==============================] - 0s 43ms/step - loss: 0.0474 - accuracy:
0.9844
Epoch 10/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0418 - accuracy:
0.9844
Epoch 1/10
4/4 [==============================] - 0s 28ms/step - loss: 0.3558 - accuracy:
0.8281
Epoch 2/10
4/4 [==============================] - 0s 41ms/step - loss: 0.2171 - accuracy:
0.8906
Epoch 3/10
```

```
4/4 [==============================] - 0s 48ms/step - loss: 0.1477 - accuracy:
0.9375
Epoch 4/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0988 - accuracy:
0.9531
Epoch 5/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0696 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0506 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0392 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 29ms/step - loss: 0.0328 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 39ms/step - loss: 0.0271 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0245 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 41ms/step - loss: 0.3032 - accuracy:
0.8750
Epoch 2/10
4/4 [==============================] - 0s 47ms/step - loss: 0.1618 - accuracy:
0.9375
Epoch 3/10
4/4 [==============================] - 0s 34ms/step - loss: 0.1003 - accuracy:
0.9609
Epoch 4/10
4/4 [==============================] - 0s 30ms/step - loss: 0.0721 - accuracy:
0.9688
Epoch 5/10
4/4 [==============================] - 0s 55ms/step - loss: 0.0552 - accuracy:
0.9922
Epoch 6/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0424 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0344 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0296 - accuracy:
0.9922
Epoch 9/10
```

```
4/4 [==============================] - 0s 27ms/step - loss: 0.0258 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0222 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 35ms/step - loss: 0.2829 - accuracy:
0.8906
Epoch 2/10
4/4 [==============================] - 0s 29ms/step - loss: 0.1297 - accuracy:
0.9453
Epoch 3/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0925 - accuracy:
0.9766
Epoch 4/10
4/4 [==============================] - 0s 51ms/step - loss: 0.0694 - accuracy:
0.9844
Epoch 5/10
4/4 [==============================] - 0s 25ms/step - loss: 0.0534 - accuracy:
0.9922
Epoch 6/10
4/4 [==============================] - 0s 51ms/step - loss: 0.0408 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0293 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0239 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0205 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 39ms/step - loss: 0.0180 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 34ms/step - loss: 0.4347 - accuracy:
0.8906
Epoch 2/10
4/4 [==============================] - 0s 40ms/step - loss: 0.1741 - accuracy:
0.9297
Epoch 3/10
4/4 [==============================] - 0s 35ms/step - loss: 0.0988 - accuracy:
0.9609
Epoch 4/10
4/4 [==============================] - 0s 43ms/step - loss: 0.0798 - accuracy:
0.9766
Epoch 5/10
```

```
4/4 [==============================] - 0s 92ms/step - loss: 0.0598 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 76ms/step - loss: 0.0511 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 67ms/step - loss: 0.0401 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 51ms/step - loss: 0.0336 - accuracy:
0.9844
Epoch 9/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0295 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0247 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 39ms/step - loss: 0.5124 - accuracy:
0.8125
Epoch 2/10
4/4 [==============================] - 0s 56ms/step - loss: 0.2745 - accuracy:
0.8828
Epoch 3/10
4/4 [==============================] - 0s 42ms/step - loss: 0.1932 - accuracy:
0.8828
Epoch 4/10
4/4 [==============================] - 0s 41ms/step - loss: 0.1333 - accuracy:
0.9375
Epoch 5/10
4/4 [==============================] - 0s 36ms/step - loss: 0.0956 - accuracy:
0.9531
Epoch 6/10
4/4 [==============================] - 0s 39ms/step - loss: 0.0749 - accuracy:
0.9766
Epoch 7/10
4/4 [==============================] - 0s 56ms/step - loss: 0.0587 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 36ms/step - loss: 0.0491 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 53ms/step - loss: 0.0410 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0351 - accuracy:
1.0000
Epoch 1/10
```

```
4/4 [==============================] - 0s 38ms/step - loss: 0.3089 - accuracy:
0.8750
Epoch 2/10
4/4 [==============================] - 0s 46ms/step - loss: 0.2210 - accuracy:
0.8906
Epoch 3/10
4/4 [==============================] - 0s 46ms/step - loss: 0.1495 - accuracy:
0.9375
Epoch 4/10
4/4 [==============================] - 0s 55ms/step - loss: 0.1082 - accuracy:
0.9453
Epoch 5/10
4/4 [==============================] - 0s 63ms/step - loss: 0.0780 - accuracy:
0.9609
Epoch 6/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0632 - accuracy:
0.9766
Epoch 7/10
4/4 [==============================] - 0s 53ms/step - loss: 0.0510 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0422 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0362 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0316 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 51ms/step - loss: 0.7035 - accuracy:
0.8906
Epoch 2/10
4/4 [==============================] - 0s 85ms/step - loss: 0.3863 - accuracy:
0.9219
Epoch 3/10
4/4 [==============================] - 0s 37ms/step - loss: 0.1587 - accuracy:
0.9453
Epoch 4/10
4/4 [==============================] - 0s 67ms/step - loss: 0.1136 - accuracy:
0.9609
Epoch 5/10
4/4 [==============================] - 0s 30ms/step - loss: 0.0909 - accuracy:
0.9766
Epoch 6/10
4/4 [==============================] - 0s 55ms/step - loss: 0.0727 - accuracy:
0.9766
Epoch 7/10
```

```
4/4 [==============================] - 0s 42ms/step - loss: 0.0630 - accuracy:
0.9766
Epoch 8/10
4/4 [==============================] - 0s 49ms/step - loss: 0.0513 - accuracy:
0.9766
Epoch 9/10
4/4 [==============================] - 0s 39ms/step - loss: 0.0455 - accuracy:
0.9844
Epoch 10/10
4/4 [==============================] - 0s 51ms/step - loss: 0.0395 - accuracy:
0.9844
Epoch 1/10
4/4 [==============================] - 0s 27ms/step - loss: 0.2720 - accuracy:
0.8438
Epoch 2/10
4/4 [==============================] - 0s 42ms/step - loss: 0.1788 - accuracy:
0.9141
Epoch 3/10
4/4 [==============================] - 0s 38ms/step - loss: 0.1277 - accuracy:
0.9375
Epoch 4/10
4/4 [==============================] - 0s 36ms/step - loss: 0.0932 - accuracy:
0.9531
Epoch 5/10
4/4 [==============================] - 0s 66ms/step - loss: 0.0682 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 60ms/step - loss: 0.0521 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 58ms/step - loss: 0.0412 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 92ms/step - loss: 0.0347 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 57ms/step - loss: 0.0300 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 75ms/step - loss: 0.0266 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 39ms/step - loss: 0.5170 - accuracy:
0.8359
Epoch 2/10
4/4 [==============================] - 0s 44ms/step - loss: 0.2715 - accuracy:
0.9141
Epoch 3/10
```

```
4/4 [==============================] - 0s 42ms/step - loss: 0.1284 - accuracy:
0.9766
Epoch 4/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0960 - accuracy:
0.9844
Epoch 5/10
4/4 [==============================] - 0s 49ms/step - loss: 0.0744 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0598 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 65ms/step - loss: 0.0469 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 54ms/step - loss: 0.0400 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0337 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0294 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 46ms/step - loss: 0.5963 - accuracy:
0.7656
Epoch 2/10
4/4 [==============================] - 0s 44ms/step - loss: 0.3611 - accuracy:
0.8281
Epoch 3/10
4/4 [==============================] - 0s 34ms/step - loss: 0.1763 - accuracy:
0.8984
Epoch 4/10
4/4 [==============================] - 0s 37ms/step - loss: 0.1115 - accuracy:
0.9453
Epoch 5/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0788 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 32ms/step - loss: 0.0608 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0495 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0423 - accuracy:
0.9922
Epoch 9/10
```

```
4/4 [==============================] - 0s 44ms/step - loss: 0.0372 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0329 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 33ms/step - loss: 0.4781 - accuracy:
0.8047
Epoch 2/10
4/4 [==============================] - 0s 46ms/step - loss: 0.2623 - accuracy:
0.8984
Epoch 3/10
4/4 [==============================] - 0s 43ms/step - loss: 0.2004 - accuracy:
0.9219
Epoch 4/10
4/4 [==============================] - 0s 56ms/step - loss: 0.1532 - accuracy:
0.9531
Epoch 5/10
4/4 [==============================] - 0s 41ms/step - loss: 0.1150 - accuracy:
0.9922
Epoch 6/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0948 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 35ms/step - loss: 0.0731 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 43ms/step - loss: 0.0616 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 26ms/step - loss: 0.0500 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0399 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 49ms/step - loss: 0.5071 - accuracy:
0.8281
Epoch 2/10
4/4 [==============================] - 0s 48ms/step - loss: 0.2040 - accuracy:
0.9062
Epoch 3/10
4/4 [==============================] - 0s 44ms/step - loss: 0.1356 - accuracy:
0.9609
Epoch 4/10
4/4 [==============================] - 0s 36ms/step - loss: 0.0974 - accuracy:
0.9766
Epoch 5/10
```

```
4/4 [==============================] - 0s 43ms/step - loss: 0.0709 - accuracy:
0.9922
Epoch 6/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0546 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0438 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0357 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 55ms/step - loss: 0.0302 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 53ms/step - loss: 0.0265 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 48ms/step - loss: 0.3314 - accuracy:
0.8438
Epoch 2/10
4/4 [==============================] - 0s 35ms/step - loss: 0.1908 - accuracy:
0.9219
Epoch 3/10
4/4 [==============================] - 0s 49ms/step - loss: 0.1316 - accuracy:
0.9375
Epoch 4/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0913 - accuracy:
0.9688
Epoch 5/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0670 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 60ms/step - loss: 0.0525 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0435 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0383 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 27ms/step - loss: 0.0337 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 32ms/step - loss: 0.0304 - accuracy:
0.9922
Epoch 1/10
```

```
4/4 [==============================] - 0s 46ms/step - loss: 0.6870 - accuracy:
0.7734
Epoch 2/10
4/4 [==============================] - 0s 48ms/step - loss: 0.2856 - accuracy:
0.8750
Epoch 3/10
4/4 [==============================] - 0s 42ms/step - loss: 0.1793 - accuracy:
0.9453
Epoch 4/10
4/4 [==============================] - 0s 41ms/step - loss: 0.1313 - accuracy:
0.9609
Epoch 5/10
4/4 [==============================] - 0s 54ms/step - loss: 0.0987 - accuracy:
0.9609
Epoch 6/10
4/4 [==============================] - 0s 39ms/step - loss: 0.0722 - accuracy:
0.9766
Epoch 7/10
4/4 [==============================] - 0s 51ms/step - loss: 0.0568 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 51ms/step - loss: 0.0433 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 39ms/step - loss: 0.0381 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0320 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 40ms/step - loss: 0.4794 - accuracy:
0.8125
Epoch 2/10
4/4 [==============================] - 0s 26ms/step - loss: 0.2473 - accuracy:
0.8828
Epoch 3/10
4/4 [==============================] - 0s 52ms/step - loss: 0.1591 - accuracy:
0.9453
Epoch 4/10
4/4 [==============================] - 0s 45ms/step - loss: 0.1129 - accuracy:
0.9531
Epoch 5/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0882 - accuracy:
0.9766
Epoch 6/10
4/4 [==============================] - 0s 58ms/step - loss: 0.0639 - accuracy:
0.9844
Epoch 7/10
```

```
4/4 [==============================] - 0s 47ms/step - loss: 0.0510 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 58ms/step - loss: 0.0421 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 24ms/step - loss: 0.0352 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0303 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 41ms/step - loss: 0.5917 - accuracy:
0.8516
Epoch 2/10
4/4 [==============================] - 0s 44ms/step - loss: 0.2693 - accuracy:
0.8984
Epoch 3/10
4/4 [==============================] - 0s 47ms/step - loss: 0.1473 - accuracy:
0.9375
Epoch 4/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0961 - accuracy:
0.9531
Epoch 5/10
4/4 [==============================] - 0s 43ms/step - loss: 0.0684 - accuracy:
0.9688
Epoch 6/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0499 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0357 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 34ms/step - loss: 0.0284 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0239 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 49ms/step - loss: 0.0206 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 36ms/step - loss: 0.3505 - accuracy:
0.8672
Epoch 2/10
4/4 [==============================] - 0s 37ms/step - loss: 0.1797 - accuracy:
0.9219
Epoch 3/10
```

```
4/4 [==============================] - 0s 47ms/step - loss: 0.1227 - accuracy:
0.9531
Epoch 4/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0888 - accuracy:
0.9766
Epoch 5/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0622 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0501 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0413 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0360 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0318 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0286 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 46ms/step - loss: 0.2921 - accuracy:
0.8828
Epoch 2/10
4/4 [==============================] - 0s 31ms/step - loss: 0.2034 - accuracy:
0.9141
Epoch 3/10
4/4 [==============================] - 0s 36ms/step - loss: 0.1479 - accuracy:
0.9609
Epoch 4/10
4/4 [==============================] - 0s 37ms/step - loss: 0.1129 - accuracy:
0.9609
Epoch 5/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0881 - accuracy:
0.9609
Epoch 6/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0652 - accuracy:
0.9688
Epoch 7/10
4/4 [==============================] - 0s 51ms/step - loss: 0.0524 - accuracy:
0.9766
Epoch 8/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0427 - accuracy:
0.9844
Epoch 9/10
```

```
4/4 [==============================] - 0s 44ms/step - loss: 0.0338 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0279 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 51ms/step - loss: 0.3861 - accuracy:
0.8203
Epoch 2/10
4/4 [==============================] - 0s 65ms/step - loss: 0.2628 - accuracy:
0.8750
Epoch 3/10
4/4 [==============================] - 0s 55ms/step - loss: 0.1868 - accuracy:
0.8984
Epoch 4/10
4/4 [==============================] - 0s 35ms/step - loss: 0.1282 - accuracy:
0.9531
Epoch 5/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0945 - accuracy:
0.9766
Epoch 6/10
4/4 [==============================] - 0s 35ms/step - loss: 0.0681 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0554 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0463 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 31ms/step - loss: 0.0388 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 35ms/step - loss: 0.0337 - accuracy:
0.9922
Epoch 1/10
4/4 [==============================] - 0s 47ms/step - loss: 0.5511 - accuracy:
0.8047
Epoch 2/10
4/4 [==============================] - 0s 48ms/step - loss: 0.2440 - accuracy:
0.8828
Epoch 3/10
4/4 [==============================] - 0s 46ms/step - loss: 0.1462 - accuracy:
0.9297
Epoch 4/10
4/4 [==============================] - 0s 32ms/step - loss: 0.1061 - accuracy:
0.9531
Epoch 5/10
```

```
4/4 [==============================] - 0s 52ms/step - loss: 0.0777 - accuracy:
0.9766
Epoch 6/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0552 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0453 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0396 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0326 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 35ms/step - loss: 0.0286 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 42ms/step - loss: 0.5188 - accuracy:
0.8125
Epoch 2/10
4/4 [==============================] - 0s 55ms/step - loss: 0.2335 - accuracy:
0.9141
Epoch 3/10
4/4 [==============================] - 0s 46ms/step - loss: 0.1690 - accuracy:
0.9297
Epoch 4/10
4/4 [==============================] - 0s 40ms/step - loss: 0.1196 - accuracy:
0.9453
Epoch 5/10
4/4 [==============================] - 0s 53ms/step - loss: 0.0901 - accuracy:
0.9531
Epoch 6/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0642 - accuracy:
0.9766
Epoch 7/10
4/4 [==============================] - 0s 50ms/step - loss: 0.0501 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0397 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0337 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0285 - accuracy:
1.0000
Epoch 1/10
```

```
4/4 [==============================] - 0s 43ms/step - loss: 0.2593 - accuracy:
0.8906
Epoch 2/10
4/4 [==============================] - 0s 35ms/step - loss: 0.1441 - accuracy:
0.9375
Epoch 3/10
4/4 [==============================] - 0s 40ms/step - loss: 0.1023 - accuracy:
0.9531
Epoch 4/10
4/4 [==============================] - 0s 36ms/step - loss: 0.0783 - accuracy:
0.9609
Epoch 5/10
4/4 [==============================] - 0s 22ms/step - loss: 0.0612 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 79ms/step - loss: 0.0517 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 52ms/step - loss: 0.0424 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 46ms/step - loss: 0.0365 - accuracy:
0.9922
Epoch 9/10
4/4 [==============================] - 0s 60ms/step - loss: 0.0318 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 42ms/step - loss: 0.0280 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 50ms/step - loss: 0.2373 - accuracy:
0.8984
Epoch 2/10
4/4 [==============================] - 0s 42ms/step - loss: 0.1340 - accuracy:
0.9453
Epoch 3/10
4/4 [==============================] - 0s 43ms/step - loss: 0.0927 - accuracy:
0.9531
Epoch 4/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0658 - accuracy:
0.9688
Epoch 5/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0456 - accuracy:
0.9922
Epoch 6/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0362 - accuracy:
1.0000
Epoch 7/10
```

```
4/4 [==============================] - 0s 36ms/step - loss: 0.0291 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 48ms/step - loss: 0.0241 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 39ms/step - loss: 0.0199 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0178 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 32ms/step - loss: 0.4430 - accuracy:
0.8516
Epoch 2/10
4/4 [==============================] - 0s 23ms/step - loss: 0.2074 - accuracy:
0.9297
Epoch 3/10
4/4 [==============================] - 0s 47ms/step - loss: 0.1380 - accuracy:
0.9297
Epoch 4/10
4/4 [==============================] - 0s 34ms/step - loss: 0.0937 - accuracy:
0.9609
Epoch 5/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0584 - accuracy:
0.9844
Epoch 6/10
4/4 [==============================] - 0s 30ms/step - loss: 0.0414 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0295 - accuracy:
1.0000
Epoch 8/10
4/4 [==============================] - 0s 51ms/step - loss: 0.0244 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 44ms/step - loss: 0.0201 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0173 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 46ms/step - loss: 0.4575 - accuracy:
0.8594
Epoch 2/10
4/4 [==============================] - 0s 48ms/step - loss: 0.2819 - accuracy:
0.8906
Epoch 3/10
```

```
4/4 [==============================] - 0s 40ms/step - loss: 0.1863 - accuracy:
0.9297
Epoch 4/10
4/4 [==============================] - 0s 36ms/step - loss: 0.1453 - accuracy:
0.9375
Epoch 5/10
4/4 [==============================] - 0s 24ms/step - loss: 0.1028 - accuracy:
0.9609
Epoch 6/10
4/4 [==============================] - 0s 40ms/step - loss: 0.0815 - accuracy:
0.9766
Epoch 7/10
4/4 [==============================] - 0s 33ms/step - loss: 0.0575 - accuracy:
0.9766
Epoch 8/10
4/4 [==============================] - 0s 53ms/step - loss: 0.0434 - accuracy:
0.9844
Epoch 9/10
4/4 [==============================] - 0s 26ms/step - loss: 0.0358 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0309 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 21ms/step - loss: 0.2305 - accuracy:
0.8984
Epoch 2/10
4/4 [==============================] - 0s 38ms/step - loss: 0.1229 - accuracy:
0.9531
Epoch 3/10
4/4 [==============================] - 0s 35ms/step - loss: 0.0947 - accuracy:
0.9531
Epoch 4/10
4/4 [==============================] - 0s 39ms/step - loss: 0.0668 - accuracy:
0.9688
Epoch 5/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0478 - accuracy:
0.9766
Epoch 6/10
4/4 [==============================] - 0s 27ms/step - loss: 0.0364 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 41ms/step - loss: 0.0273 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 29ms/step - loss: 0.0203 - accuracy:
0.9922
Epoch 9/10
```

```
4/4 [==============================] - 0s 21ms/step - loss: 0.0164 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 19ms/step - loss: 0.0135 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 22ms/step - loss: 0.3282 - accuracy:
0.9062
Epoch 2/10
4/4 [==============================] - 0s 39ms/step - loss: 0.1443 - accuracy:
0.9375
Epoch 3/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0935 - accuracy:
0.9766
Epoch 4/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0620 - accuracy:
0.9844
Epoch 5/10
4/4 [==============================] - 0s 39ms/step - loss: 0.0454 - accuracy:
0.9922
Epoch 6/10
4/4 [==============================] - 0s 32ms/step - loss: 0.0341 - accuracy:
0.9922
Epoch 7/10
4/4 [==============================] - 0s 47ms/step - loss: 0.0259 - accuracy:
0.9922
Epoch 8/10
4/4 [==============================] - 0s 36ms/step - loss: 0.0213 - accuracy:
1.0000
Epoch 9/10
4/4 [==============================] - 0s 28ms/step - loss: 0.0179 - accuracy:
1.0000
Epoch 10/10
4/4 [==============================] - 0s 37ms/step - loss: 0.0155 - accuracy:
1.0000
Epoch 1/10
4/4 [==============================] - 0s 54ms/step - loss: 0.2782 - accuracy:
0.8984
Epoch 2/10
4/4 [==============================] - 0s 52ms/step - loss: 0.1421 - accuracy:
0.9297
Epoch 3/10
4/4 [==============================] - 0s 38ms/step - loss: 0.1166 - accuracy:
0.9453
Epoch 4/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0940 - accuracy:
0.9688
Epoch 5/10
```

```
4/4 [==============================] - 0s 35ms/step - loss: 0.0728 - accuracy:
0.9688
Epoch 6/10
4/4 [==============================] - 0s 33ms/step - loss: 0.0573 - accuracy:
0.9844
Epoch 7/10
4/4 [==============================] - 0s 33ms/step - loss: 0.0480 - accuracy:
0.9844
Epoch 8/10
4/4 [==============================] - 0s 38ms/step - loss: 0.0384 - accuracy:
0.9844
Epoch 9/10
4/4 [==============================] - 0s 45ms/step - loss: 0.0327 - accuracy:
0.9922
Epoch 10/10
4/4 [==============================] - 0s 36ms/step - loss: 0.0266 - accuracy:
0.9922
```

3.22.2 Exportando o modelo Rede Neural

[331]:
```
model_contains.save_weights('pesos/keras_finished_contains_2.h5')
```

3.22.3 Carregar modelo Rede Neural

[332]:
```
keras_model_sn = create_nn_model()
keras_model_sn.load_weights('pesos/keras_finished_contains_2.h5')
```

3.23 Avaliando as métricas do modelo Rede Neural contains sem validação em treino

- Avaliando a acurácia do modelo

[333]:
```
score, acc = keras_model_sn.evaluate(x_test_batch_sn, y_test_batch_sn)
print("A acurácia para o modelo com Redes neurais com o dataset contains sem␣
 ↪validação foi de {:.2f}%".format(acc*100))
```

```
4/4 [==============================] - 1s 42ms/step - loss: 0.2759 - accuracy:
0.8792
A acurácia para o modelo com Redes neurais com o dataset contains sem validação
foi de 84.38%
```

- obtendo as predições do modelo utilizando o dataset de treino com **contains**

[334]:
```
y_pred_keras_sn = model_contains.predict(x_test_sn.to_numpy()).round()
y_pred_keras_sn = [classify(result) for result in y_pred_keras_sn]
```

3.24 Classification Report

- A função `classification_report` nos dá todas as métricas importantes que podemos obter de um modelo.
- Ela irá nos dar a precisão por classe.
- O quanto de revoação (recall) o modelo tem por classe e no geral
- O f1-score geral e por classe
- A acurária do modelo
- Todas estas métricas são úteis e nos ajudarão a escolher qual o modelo se saiu melhor e pode ser utilizado para esta tarefa.

[335]:
```
report = classification_report(y_test, y_pred_keras_sn,␣
↪target_names=target_names, output_dict=True)
```

[336]:
```
sgd_stats = {'model': 'keras_contains',
             'accuracy': report['accuracy'],
             'recall': report['macro avg']['recall'],
             'f1_score': report['macro avg']['f1-score'],
             'support': report['macro avg']['support'],
             'falso_precision': report['Falso']['precision'],
             'falso_recall': report['Falso']['recall'],
             'falso_f1_score': report['Falso']['f1-score'],
             'verdadeiro_precision': report['Verdadeiro']['precision'],
             'verdadeiro_recall': report['Verdadeiro']['recall'],
             'verdadeiro_f1_score': report['Verdadeiro']['f1-score']}
```

[337]:
```
stats.append(sgd_stats)
```

3.25 Rede Neural com dataset count e com validação em treino

3.25.1 Observação o modelo já foi treinado, caso queira usar o modelo já treinado ele pode ser carregado diretamente aqui

- A célula cria o modelo de rede neural descrito

[338]:
```
model_ = create_nn_model()
```

- A célula abaixo treina a rede neural

[339]:
```
inicio_lote = 0
fim_lote = tamanho_lote

total_lotes = str(int(len(X_train)/tamanho_lote))
ultimo_lote = False
i = 1
```

```
while True:
    x_train = formatar_lote_de_noticias(X_train[inicio_lote:fim_lote])
    y_tr = np.asanyarray(format_labels(y_train[inicio_lote:fim_lote].
 ↪to_numpy()))
    model_.fit(x=x_train.to_numpy(), y=y_tr, validation_data=(val_formated,␣
 ↪val_formated_class), epochs=10)
    inicio_lote += tamanho_lote

    if fim_lote + tamanho_lote > len(X_train):
        fim_lote = len(X_train)
        ultimo_lote = True
    else:
        fim_lote += tamanho_lote

    i += 1
    if ultimo_lote:
        break
```

```
Epoch 1/10
4/4 [==============================] - 13s 4s/step - loss: 0.7112 - accuracy:
0.4490 - val_loss: 0.6399 - val_accuracy: 0.5039
Epoch 2/10
4/4 [==============================] - 1s 176ms/step - loss: 0.5975 - accuracy:
0.6094 - val_loss: 0.5962 - val_accuracy: 0.4609
Epoch 3/10
4/4 [==============================] - 1s 199ms/step - loss: 0.5347 - accuracy:
0.6406 - val_loss: 0.5965 - val_accuracy: 0.4844
Epoch 4/10
4/4 [==============================] - 1s 184ms/step - loss: 0.4977 - accuracy:
0.7458 - val_loss: 0.5941 - val_accuracy: 0.5312
Epoch 5/10
4/4 [==============================] - 1s 162ms/step - loss: 0.4565 - accuracy:
0.8969 - val_loss: 0.5822 - val_accuracy: 0.5703
Epoch 6/10
4/4 [==============================] - 0s 142ms/step - loss: 0.4263 - accuracy:
0.9573 - val_loss: 0.5694 - val_accuracy: 0.5938
Epoch 7/10
4/4 [==============================] - 0s 138ms/step - loss: 0.4106 - accuracy:
0.9354 - val_loss: 0.5550 - val_accuracy: 0.6719
Epoch 8/10
4/4 [==============================] - 1s 159ms/step - loss: 0.3656 - accuracy:
0.9396 - val_loss: 0.5469 - val_accuracy: 0.7344
Epoch 9/10
4/4 [==============================] - 0s 153ms/step - loss: 0.3407 - accuracy:
0.9437 - val_loss: 0.5447 - val_accuracy: 0.7695
Epoch 10/10
```

```
4/4 [==============================] - 1s 225ms/step - loss: 0.2936 - accuracy:
0.9240 - val_loss: 0.5415 - val_accuracy: 0.7930
Epoch 1/10
4/4 [==============================] - 2s 621ms/step - loss: 0.5387 - accuracy:
0.7500 - val_loss: 0.5167 - val_accuracy: 0.7539
Epoch 2/10
4/4 [==============================] - 1s 163ms/step - loss: 0.4694 - accuracy:
0.7891 - val_loss: 0.5040 - val_accuracy: 0.8047
Epoch 3/10
4/4 [==============================] - 1s 200ms/step - loss: 0.4007 - accuracy:
0.8516 - val_loss: 0.5289 - val_accuracy: 0.8125
Epoch 4/10
4/4 [==============================] - 0s 144ms/step - loss: 0.3451 - accuracy:
0.8516 - val_loss: 0.5582 - val_accuracy: 0.7773
Epoch 5/10
4/4 [==============================] - 1s 222ms/step - loss: 0.3074 - accuracy:
0.8516 - val_loss: 0.5886 - val_accuracy: 0.7578
Epoch 6/10
4/4 [==============================] - 1s 157ms/step - loss: 0.2776 - accuracy:
0.8516 - val_loss: 0.6208 - val_accuracy: 0.7500
Epoch 7/10
4/4 [==============================] - 1s 171ms/step - loss: 0.2557 - accuracy:
0.8672 - val_loss: 0.6706 - val_accuracy: 0.7461
Epoch 8/10
4/4 [==============================] - 1s 173ms/step - loss: 0.2376 - accuracy:
0.8906 - val_loss: 0.7308 - val_accuracy: 0.7344
Epoch 9/10
4/4 [==============================] - 1s 166ms/step - loss: 0.2219 - accuracy:
0.8984 - val_loss: 0.8020 - val_accuracy: 0.7305
Epoch 10/10
4/4 [==============================] - 1s 178ms/step - loss: 0.2070 - accuracy:
0.9062 - val_loss: 0.8828 - val_accuracy: 0.7227
Epoch 1/10
4/4 [==============================] - 1s 204ms/step - loss: 0.7329 - accuracy:
0.7500 - val_loss: 0.5117 - val_accuracy: 0.7578
Epoch 2/10
4/4 [==============================] - 1s 182ms/step - loss: 0.5536 - accuracy:
0.7188 - val_loss: 0.6561 - val_accuracy: 0.6250
Epoch 3/10
4/4 [==============================] - 1s 189ms/step - loss: 0.5002 - accuracy:
0.7656 - val_loss: 0.6159 - val_accuracy: 0.6406
Epoch 4/10
4/4 [==============================] - 1s 176ms/step - loss: 0.3496 - accuracy:
0.8438 - val_loss: 0.5416 - val_accuracy: 0.7031
Epoch 5/10
4/4 [==============================] - 1s 192ms/step - loss: 0.2854 - accuracy:
0.8828 - val_loss: 0.4759 - val_accuracy: 0.7812
Epoch 6/10
```

```
4/4 [==============================] - 1s 167ms/step - loss: 0.2511 - accuracy:
0.8828 - val_loss: 0.4363 - val_accuracy: 0.8086
Epoch 7/10
4/4 [==============================] - 1s 186ms/step - loss: 0.2294 - accuracy:
0.8828 - val_loss: 0.4199 - val_accuracy: 0.8086
Epoch 8/10
4/4 [==============================] - 1s 161ms/step - loss: 0.2141 - accuracy:
0.8828 - val_loss: 0.4163 - val_accuracy: 0.8203
Epoch 9/10
4/4 [==============================] - 1s 192ms/step - loss: 0.2013 - accuracy:
0.8828 - val_loss: 0.4173 - val_accuracy: 0.8242
Epoch 10/10
4/4 [==============================] - 1s 206ms/step - loss: 0.1901 - accuracy:
0.8906 - val_loss: 0.4180 - val_accuracy: 0.8359
Epoch 1/10
4/4 [==============================] - 1s 257ms/step - loss: 0.3899 - accuracy:
0.8516 - val_loss: 0.3986 - val_accuracy: 0.8125
Epoch 2/10
4/4 [==============================] - 1s 203ms/step - loss: 0.2896 - accuracy:
0.8984 - val_loss: 0.3894 - val_accuracy: 0.8164
Epoch 3/10
4/4 [==============================] - 1s 151ms/step - loss: 0.2313 - accuracy:
0.9219 - val_loss: 0.3738 - val_accuracy: 0.8203
Epoch 4/10
4/4 [==============================] - 0s 118ms/step - loss: 0.1868 - accuracy:
0.9297 - val_loss: 0.3676 - val_accuracy: 0.8359
Epoch 5/10
4/4 [==============================] - 0s 111ms/step - loss: 0.1637 - accuracy:
0.9297 - val_loss: 0.3719 - val_accuracy: 0.8477
Epoch 6/10
4/4 [==============================] - 0s 129ms/step - loss: 0.1480 - accuracy:
0.9297 - val_loss: 0.3853 - val_accuracy: 0.8438
Epoch 7/10
4/4 [==============================] - 1s 162ms/step - loss: 0.1355 - accuracy:
0.9375 - val_loss: 0.3999 - val_accuracy: 0.8438
Epoch 8/10
4/4 [==============================] - 0s 150ms/step - loss: 0.1254 - accuracy:
0.9453 - val_loss: 0.4123 - val_accuracy: 0.8359
Epoch 9/10
4/4 [==============================] - 1s 176ms/step - loss: 0.1171 - accuracy:
0.9453 - val_loss: 0.4194 - val_accuracy: 0.8359
Epoch 10/10
4/4 [==============================] - 1s 190ms/step - loss: 0.1094 - accuracy:
0.9531 - val_loss: 0.4257 - val_accuracy: 0.8359
Epoch 1/10
4/4 [==============================] - 1s 193ms/step - loss: 0.4777 - accuracy:
0.7969 - val_loss: 0.3643 - val_accuracy: 0.8164
Epoch 2/10
```

```
4/4 [==============================] - 1s 153ms/step - loss: 0.3214 - accuracy:
0.8516 - val_loss: 0.4055 - val_accuracy: 0.8086
Epoch 3/10
4/4 [==============================] - 1s 170ms/step - loss: 0.2537 - accuracy:
0.8984 - val_loss: 0.3835 - val_accuracy: 0.8164
Epoch 4/10
4/4 [==============================] - 1s 192ms/step - loss: 0.1936 - accuracy:
0.9297 - val_loss: 0.3524 - val_accuracy: 0.8203
Epoch 5/10
4/4 [==============================] - 1s 174ms/step - loss: 0.1646 - accuracy:
0.9297 - val_loss: 0.3335 - val_accuracy: 0.8242
Epoch 6/10
4/4 [==============================] - 1s 202ms/step - loss: 0.1478 - accuracy:
0.9297 - val_loss: 0.3264 - val_accuracy: 0.8320
Epoch 7/10
4/4 [==============================] - 1s 154ms/step - loss: 0.1362 - accuracy:
0.9297 - val_loss: 0.3268 - val_accuracy: 0.8320
Epoch 8/10
4/4 [==============================] - 1s 191ms/step - loss: 0.1270 - accuracy:
0.9297 - val_loss: 0.3305 - val_accuracy: 0.8438
Epoch 9/10
4/4 [==============================] - 1s 176ms/step - loss: 0.1191 - accuracy:
0.9297 - val_loss: 0.3350 - val_accuracy: 0.8398
Epoch 10/10
4/4 [==============================] - 0s 138ms/step - loss: 0.1118 - accuracy:
0.9375 - val_loss: 0.3395 - val_accuracy: 0.8359
Epoch 1/10
4/4 [==============================] - 1s 194ms/step - loss: 0.2975 - accuracy:
0.8672 - val_loss: 0.3315 - val_accuracy: 0.8320
Epoch 2/10
4/4 [==============================] - 0s 145ms/step - loss: 0.2399 - accuracy:
0.8906 - val_loss: 0.3312 - val_accuracy: 0.8320
Epoch 3/10
4/4 [==============================] - 1s 182ms/step - loss: 0.2080 - accuracy:
0.8984 - val_loss: 0.3353 - val_accuracy: 0.8320
Epoch 4/10
4/4 [==============================] - 1s 224ms/step - loss: 0.1857 - accuracy:
0.8984 - val_loss: 0.3423 - val_accuracy: 0.8359
Epoch 5/10
4/4 [==============================] - 1s 194ms/step - loss: 0.1688 - accuracy:
0.8984 - val_loss: 0.3495 - val_accuracy: 0.8320
Epoch 6/10
4/4 [==============================] - 1s 165ms/step - loss: 0.1537 - accuracy:
0.8984 - val_loss: 0.3559 - val_accuracy: 0.8281
Epoch 7/10
4/4 [==============================] - 1s 191ms/step - loss: 0.1412 - accuracy:
0.8984 - val_loss: 0.3589 - val_accuracy: 0.8281
Epoch 8/10
```

```
4/4 [==============================] - 1s 191ms/step - loss: 0.1295 - accuracy:
0.9141 - val_loss: 0.3609 - val_accuracy: 0.8320
Epoch 9/10
4/4 [==============================] - 1s 189ms/step - loss: 0.1189 - accuracy:
0.9141 - val_loss: 0.3612 - val_accuracy: 0.8242
Epoch 10/10
4/4 [==============================] - 1s 175ms/step - loss: 0.1097 - accuracy:
0.9375 - val_loss: 0.3618 - val_accuracy: 0.8242
Epoch 1/10
4/4 [==============================] - 1s 257ms/step - loss: 0.3162 - accuracy:
0.8672 - val_loss: 0.3484 - val_accuracy: 0.8242
Epoch 2/10
4/4 [==============================] - 1s 183ms/step - loss: 0.2398 - accuracy:
0.8828 - val_loss: 0.4427 - val_accuracy: 0.8125
Epoch 3/10
4/4 [==============================] - 1s 169ms/step - loss: 0.1900 - accuracy:
0.9062 - val_loss: 0.6144 - val_accuracy: 0.8047
Epoch 4/10
4/4 [==============================] - 0s 135ms/step - loss: 0.1632 - accuracy:
0.9062 - val_loss: 0.7911 - val_accuracy: 0.7773
Epoch 5/10
4/4 [==============================] - 1s 214ms/step - loss: 0.1449 - accuracy:
0.9062 - val_loss: 0.9271 - val_accuracy: 0.7578
Epoch 6/10
4/4 [==============================] - 0s 141ms/step - loss: 0.1282 - accuracy:
0.9219 - val_loss: 1.0078 - val_accuracy: 0.7539
Epoch 7/10
4/4 [==============================] - 1s 171ms/step - loss: 0.1162 - accuracy:
0.9297 - val_loss: 1.0414 - val_accuracy: 0.7578
Epoch 8/10
4/4 [==============================] - 1s 178ms/step - loss: 0.1062 - accuracy:
0.9375 - val_loss: 1.0622 - val_accuracy: 0.7578
Epoch 9/10
4/4 [==============================] - 1s 171ms/step - loss: 0.0979 - accuracy:
0.9609 - val_loss: 1.0773 - val_accuracy: 0.7578
Epoch 10/10
4/4 [==============================] - 1s 168ms/step - loss: 0.0912 - accuracy:
0.9766 - val_loss: 1.0893 - val_accuracy: 0.7578
Epoch 1/10
4/4 [==============================] - 1s 247ms/step - loss: 0.5097 - accuracy:
0.8047 - val_loss: 0.3566 - val_accuracy: 0.8203
Epoch 2/10
4/4 [==============================] - 1s 164ms/step - loss: 0.3601 - accuracy:
0.8281 - val_loss: 0.4653 - val_accuracy: 0.7812
Epoch 3/10
4/4 [==============================] - 1s 168ms/step - loss: 0.2689 - accuracy:
0.8906 - val_loss: 0.4535 - val_accuracy: 0.7773
Epoch 4/10
```

```
4/4 [==============================] - 1s 207ms/step - loss: 0.1903 - accuracy:
0.9219 - val_loss: 0.4099 - val_accuracy: 0.8008
Epoch 5/10
4/4 [==============================] - 1s 169ms/step - loss: 0.1603 - accuracy:
0.9219 - val_loss: 0.3876 - val_accuracy: 0.8086
Epoch 6/10
4/4 [==============================] - 0s 149ms/step - loss: 0.1439 - accuracy:
0.9297 - val_loss: 0.3809 - val_accuracy: 0.8125
Epoch 7/10
4/4 [==============================] - 1s 171ms/step - loss: 0.1320 - accuracy:
0.9375 - val_loss: 0.3828 - val_accuracy: 0.8125
Epoch 8/10
4/4 [==============================] - 0s 135ms/step - loss: 0.1229 - accuracy:
0.9531 - val_loss: 0.3857 - val_accuracy: 0.8125
Epoch 9/10
4/4 [==============================] - 1s 167ms/step - loss: 0.1135 - accuracy:
0.9531 - val_loss: 0.3885 - val_accuracy: 0.8125
Epoch 10/10
4/4 [==============================] - 0s 149ms/step - loss: 0.1065 - accuracy:
0.9609 - val_loss: 0.3915 - val_accuracy: 0.8164
Epoch 1/10
4/4 [==============================] - 1s 241ms/step - loss: 0.5932 - accuracy:
0.7422 - val_loss: 0.3837 - val_accuracy: 0.8281
Epoch 2/10
4/4 [==============================] - 1s 186ms/step - loss: 0.3568 - accuracy:
0.7969 - val_loss: 0.3681 - val_accuracy: 0.8398
Epoch 3/10
4/4 [==============================] - 1s 177ms/step - loss: 0.2512 - accuracy:
0.8359 - val_loss: 0.3750 - val_accuracy: 0.8320
Epoch 4/10
4/4 [==============================] - 1s 177ms/step - loss: 0.2070 - accuracy:
0.8516 - val_loss: 0.3850 - val_accuracy: 0.8281
Epoch 5/10
4/4 [==============================] - 1s 204ms/step - loss: 0.1778 - accuracy:
0.8828 - val_loss: 0.3908 - val_accuracy: 0.8203
Epoch 6/10
4/4 [==============================] - 1s 184ms/step - loss: 0.1586 - accuracy:
0.9297 - val_loss: 0.3920 - val_accuracy: 0.8164
Epoch 7/10
4/4 [==============================] - 1s 177ms/step - loss: 0.1431 - accuracy:
0.9688 - val_loss: 0.3919 - val_accuracy: 0.8164
Epoch 8/10
4/4 [==============================] - 1s 166ms/step - loss: 0.1318 - accuracy:
0.9844 - val_loss: 0.3907 - val_accuracy: 0.8203
Epoch 9/10
4/4 [==============================] - 1s 160ms/step - loss: 0.1239 - accuracy:
0.9844 - val_loss: 0.3903 - val_accuracy: 0.8203
Epoch 10/10
```

```
4/4 [==============================] - 1s 184ms/step - loss: 0.1169 - accuracy:
0.9922 - val_loss: 0.3906 - val_accuracy: 0.8125
Epoch 1/10
4/4 [==============================] - 1s 242ms/step - loss: 0.4097 - accuracy:
0.7969 - val_loss: 0.4090 - val_accuracy: 0.8164
Epoch 2/10
4/4 [==============================] - 0s 149ms/step - loss: 0.2711 - accuracy:
0.8516 - val_loss: 0.4426 - val_accuracy: 0.8164
Epoch 3/10
4/4 [==============================] - 1s 210ms/step - loss: 0.2059 - accuracy:
0.8750 - val_loss: 0.4971 - val_accuracy: 0.8086
Epoch 4/10
4/4 [==============================] - 1s 197ms/step - loss: 0.1627 - accuracy:
0.9062 - val_loss: 0.5474 - val_accuracy: 0.7969
Epoch 5/10
4/4 [==============================] - 1s 174ms/step - loss: 0.1402 - accuracy:
0.9531 - val_loss: 0.5911 - val_accuracy: 0.7930
Epoch 6/10
4/4 [==============================] - 1s 166ms/step - loss: 0.1218 - accuracy:
0.9766 - val_loss: 0.6186 - val_accuracy: 0.7930
Epoch 7/10
4/4 [==============================] - 0s 143ms/step - loss: 0.1108 - accuracy:
0.9844 - val_loss: 0.6400 - val_accuracy: 0.7930
Epoch 8/10
4/4 [==============================] - 1s 156ms/step - loss: 0.1035 - accuracy:
0.9922 - val_loss: 0.6556 - val_accuracy: 0.7930
Epoch 9/10
4/4 [==============================] - 0s 141ms/step - loss: 0.0958 - accuracy:
0.9922 - val_loss: 0.6699 - val_accuracy: 0.7930
Epoch 10/10
4/4 [==============================] - 0s 147ms/step - loss: 0.0905 - accuracy:
1.0000 - val_loss: 0.6792 - val_accuracy: 0.7969
Epoch 1/10
4/4 [==============================] - 1s 228ms/step - loss: 0.4536 - accuracy:
0.7891 - val_loss: 0.3830 - val_accuracy: 0.8164
Epoch 2/10
4/4 [==============================] - 1s 196ms/step - loss: 0.3502 - accuracy:
0.8516 - val_loss: 0.4912 - val_accuracy: 0.7500
Epoch 3/10
4/4 [==============================] - 1s 173ms/step - loss: 0.2590 - accuracy:
0.8828 - val_loss: 0.5031 - val_accuracy: 0.7422
Epoch 4/10
4/4 [==============================] - 1s 176ms/step - loss: 0.1853 - accuracy:
0.9141 - val_loss: 0.4823 - val_accuracy: 0.7539
Epoch 5/10
4/4 [==============================] - 1s 178ms/step - loss: 0.1509 - accuracy:
0.9609 - val_loss: 0.4605 - val_accuracy: 0.7695
Epoch 6/10
```

```
4/4 [==============================] - 1s 178ms/step - loss: 0.1275 - accuracy:
0.9844 - val_loss: 0.4431 - val_accuracy: 0.7734
Epoch 7/10
4/4 [==============================] - 1s 162ms/step - loss: 0.1113 - accuracy:
0.9922 - val_loss: 0.4328 - val_accuracy: 0.7734
Epoch 8/10
4/4 [==============================] - 1s 173ms/step - loss: 0.1000 - accuracy:
1.0000 - val_loss: 0.4249 - val_accuracy: 0.7852
Epoch 9/10
4/4 [==============================] - 1s 192ms/step - loss: 0.0910 - accuracy:
1.0000 - val_loss: 0.4195 - val_accuracy: 0.7852
Epoch 10/10
4/4 [==============================] - 1s 218ms/step - loss: 0.0824 - accuracy:
1.0000 - val_loss: 0.4176 - val_accuracy: 0.7930
Epoch 1/10
4/4 [==============================] - 1s 240ms/step - loss: 0.5167 - accuracy:
0.8047 - val_loss: 0.3746 - val_accuracy: 0.8203
Epoch 2/10
4/4 [==============================] - 1s 175ms/step - loss: 0.2670 - accuracy:
0.8750 - val_loss: 0.3517 - val_accuracy: 0.8320
Epoch 3/10
4/4 [==============================] - 1s 162ms/step - loss: 0.1968 - accuracy:
0.9297 - val_loss: 0.3569 - val_accuracy: 0.8359
Epoch 4/10
4/4 [==============================] - 0s 148ms/step - loss: 0.1585 - accuracy:
0.9297 - val_loss: 0.3723 - val_accuracy: 0.8516
Epoch 5/10
4/4 [==============================] - 1s 158ms/step - loss: 0.1354 - accuracy:
0.9375 - val_loss: 0.3860 - val_accuracy: 0.8477
Epoch 6/10
4/4 [==============================] - 1s 178ms/step - loss: 0.1190 - accuracy:
0.9609 - val_loss: 0.3949 - val_accuracy: 0.8516
Epoch 7/10
4/4 [==============================] - 1s 185ms/step - loss: 0.1054 - accuracy:
0.9609 - val_loss: 0.3992 - val_accuracy: 0.8555
Epoch 8/10
4/4 [==============================] - 1s 192ms/step - loss: 0.0945 - accuracy:
0.9609 - val_loss: 0.4015 - val_accuracy: 0.8477
Epoch 9/10
4/4 [==============================] - 1s 188ms/step - loss: 0.0851 - accuracy:
0.9688 - val_loss: 0.4031 - val_accuracy: 0.8516
Epoch 10/10
4/4 [==============================] - 1s 170ms/step - loss: 0.0785 - accuracy:
0.9766 - val_loss: 0.4041 - val_accuracy: 0.8516
Epoch 1/10
4/4 [==============================] - 1s 257ms/step - loss: 0.6069 - accuracy:
0.7422 - val_loss: 0.3807 - val_accuracy: 0.8516
Epoch 2/10
```

```
4/4 [==============================] - 1s 201ms/step - loss: 0.3736 - accuracy:
0.7969 - val_loss: 0.3785 - val_accuracy: 0.8164
Epoch 3/10
4/4 [==============================] - 0s 151ms/step - loss: 0.2617 - accuracy:
0.8672 - val_loss: 0.4003 - val_accuracy: 0.7969
Epoch 4/10
4/4 [==============================] - 0s 153ms/step - loss: 0.2067 - accuracy:
0.9062 - val_loss: 0.4036 - val_accuracy: 0.7930
Epoch 5/10
4/4 [==============================] - 0s 141ms/step - loss: 0.1774 - accuracy:
0.9297 - val_loss: 0.3974 - val_accuracy: 0.8008
Epoch 6/10
4/4 [==============================] - 1s 160ms/step - loss: 0.1527 - accuracy:
0.9609 - val_loss: 0.3914 - val_accuracy: 0.7969
Epoch 7/10
4/4 [==============================] - 0s 149ms/step - loss: 0.1349 - accuracy:
0.9688 - val_loss: 0.3857 - val_accuracy: 0.7969
Epoch 8/10
4/4 [==============================] - 1s 182ms/step - loss: 0.1220 - accuracy:
0.9688 - val_loss: 0.3819 - val_accuracy: 0.7969
Epoch 9/10
4/4 [==============================] - 1s 175ms/step - loss: 0.1088 - accuracy:
0.9844 - val_loss: 0.3780 - val_accuracy: 0.7969
Epoch 10/10
4/4 [==============================] - 1s 161ms/step - loss: 0.0991 - accuracy:
1.0000 - val_loss: 0.3754 - val_accuracy: 0.7969
Epoch 1/10
4/4 [==============================] - 1s 235ms/step - loss: 0.3491 - accuracy:
0.8281 - val_loss: 0.3277 - val_accuracy: 0.8281
Epoch 2/10
4/4 [==============================] - 1s 183ms/step - loss: 0.2227 - accuracy:
0.8906 - val_loss: 0.3247 - val_accuracy: 0.8516
Epoch 3/10
4/4 [==============================] - 1s 140ms/step - loss: 0.1630 - accuracy:
0.9375 - val_loss: 0.3600 - val_accuracy: 0.8516
Epoch 4/10
4/4 [==============================] - 0s 140ms/step - loss: 0.1212 - accuracy:
0.9688 - val_loss: 0.3934 - val_accuracy: 0.8594
Epoch 5/10
4/4 [==============================] - 1s 156ms/step - loss: 0.0982 - accuracy:
0.9766 - val_loss: 0.4308 - val_accuracy: 0.8594
Epoch 6/10
4/4 [==============================] - 1s 172ms/step - loss: 0.0854 - accuracy:
0.9922 - val_loss: 0.4550 - val_accuracy: 0.8555
Epoch 7/10
4/4 [==============================] - 1s 156ms/step - loss: 0.0764 - accuracy:
0.9922 - val_loss: 0.4715 - val_accuracy: 0.8555
Epoch 8/10
```

```
4/4 [==============================] - 1s 154ms/step - loss: 0.0686 - accuracy:
0.9922 - val_loss: 0.4814 - val_accuracy: 0.8594
Epoch 9/10
4/4 [==============================] - 1s 183ms/step - loss: 0.0636 - accuracy:
0.9922 - val_loss: 0.4887 - val_accuracy: 0.8594
Epoch 10/10
4/4 [==============================] - 0s 155ms/step - loss: 0.0583 - accuracy:
0.9922 - val_loss: 0.4965 - val_accuracy: 0.8594
Epoch 1/10
4/4 [==============================] - 1s 216ms/step - loss: 0.3754 - accuracy:
0.8359 - val_loss: 0.3591 - val_accuracy: 0.8672
Epoch 2/10
4/4 [==============================] - 1s 160ms/step - loss: 0.2402 - accuracy:
0.8906 - val_loss: 0.3306 - val_accuracy: 0.8594
Epoch 3/10
4/4 [==============================] - 0s 152ms/step - loss: 0.1810 - accuracy:
0.9219 - val_loss: 0.3323 - val_accuracy: 0.8555
Epoch 4/10
4/4 [==============================] - 0s 143ms/step - loss: 0.1416 - accuracy:
0.9375 - val_loss: 0.3393 - val_accuracy: 0.8594
Epoch 5/10
4/4 [==============================] - 1s 168ms/step - loss: 0.1137 - accuracy:
0.9453 - val_loss: 0.3460 - val_accuracy: 0.8516
Epoch 6/10
4/4 [==============================] - 1s 162ms/step - loss: 0.0987 - accuracy:
0.9609 - val_loss: 0.3513 - val_accuracy: 0.8555
Epoch 7/10
4/4 [==============================] - 1s 167ms/step - loss: 0.0850 - accuracy:
0.9766 - val_loss: 0.3555 - val_accuracy: 0.8477
Epoch 8/10
4/4 [==============================] - 1s 180ms/step - loss: 0.0758 - accuracy:
0.9844 - val_loss: 0.3584 - val_accuracy: 0.8477
Epoch 9/10
4/4 [==============================] - 1s 171ms/step - loss: 0.0688 - accuracy:
0.9844 - val_loss: 0.3607 - val_accuracy: 0.8477
Epoch 10/10
4/4 [==============================] - 1s 178ms/step - loss: 0.0638 - accuracy:
0.9922 - val_loss: 0.3626 - val_accuracy: 0.8477
Epoch 1/10
4/4 [==============================] - 1s 206ms/step - loss: 0.4021 - accuracy:
0.8516 - val_loss: 0.3533 - val_accuracy: 0.8477
Epoch 2/10
4/4 [==============================] - 1s 157ms/step - loss: 0.2274 - accuracy:
0.8906 - val_loss: 0.3663 - val_accuracy: 0.8516
Epoch 3/10
4/4 [==============================] - 0s 117ms/step - loss: 0.1422 - accuracy:
0.9531 - val_loss: 0.4038 - val_accuracy: 0.8438
Epoch 4/10
```

```
4/4 [==============================] - 0s 133ms/step - loss: 0.1020 - accuracy:
0.9766 - val_loss: 0.4871 - val_accuracy: 0.8516
Epoch 5/10
4/4 [==============================] - 1s 160ms/step - loss: 0.0818 - accuracy:
0.9844 - val_loss: 0.5696 - val_accuracy: 0.8438
Epoch 6/10
4/4 [==============================] - 1s 194ms/step - loss: 0.0684 - accuracy:
0.9844 - val_loss: 0.6367 - val_accuracy: 0.8359
Epoch 7/10
4/4 [==============================] - 1s 155ms/step - loss: 0.0582 - accuracy:
0.9844 - val_loss: 0.6917 - val_accuracy: 0.8281
Epoch 8/10
4/4 [==============================] - 1s 176ms/step - loss: 0.0489 - accuracy:
0.9844 - val_loss: 0.7360 - val_accuracy: 0.8164
Epoch 9/10
4/4 [==============================] - 1s 155ms/step - loss: 0.0417 - accuracy:
0.9922 - val_loss: 0.7753 - val_accuracy: 0.8125
Epoch 10/10
4/4 [==============================] - 0s 136ms/step - loss: 0.0383 - accuracy:
0.9922 - val_loss: 0.8091 - val_accuracy: 0.8164
Epoch 1/10
4/4 [==============================] - 1s 295ms/step - loss: 0.4375 - accuracy:
0.8125 - val_loss: 0.3194 - val_accuracy: 0.8398
Epoch 2/10
4/4 [==============================] - 1s 155ms/step - loss: 0.2320 - accuracy:
0.8828 - val_loss: 0.3654 - val_accuracy: 0.8281
Epoch 3/10
4/4 [==============================] - 1s 153ms/step - loss: 0.1322 - accuracy:
0.9609 - val_loss: 0.3345 - val_accuracy: 0.8438
Epoch 4/10
4/4 [==============================] - 0s 144ms/step - loss: 0.0930 - accuracy:
0.9688 - val_loss: 0.3193 - val_accuracy: 0.8516
Epoch 5/10
4/4 [==============================] - 1s 170ms/step - loss: 0.0734 - accuracy:
0.9844 - val_loss: 0.3154 - val_accuracy: 0.8516
Epoch 6/10
4/4 [==============================] - 1s 159ms/step - loss: 0.0619 - accuracy:
0.9844 - val_loss: 0.3168 - val_accuracy: 0.8633
Epoch 7/10
4/4 [==============================] - 1s 161ms/step - loss: 0.0555 - accuracy:
0.9922 - val_loss: 0.3212 - val_accuracy: 0.8594
Epoch 8/10
4/4 [==============================] - 1s 182ms/step - loss: 0.0497 - accuracy:
1.0000 - val_loss: 0.3257 - val_accuracy: 0.8555
Epoch 9/10
4/4 [==============================] - 0s 153ms/step - loss: 0.0466 - accuracy:
1.0000 - val_loss: 0.3298 - val_accuracy: 0.8633
Epoch 10/10
```

```
4/4 [==============================] - 1s 179ms/step - loss: 0.0433 - accuracy:
1.0000 - val_loss: 0.3328 - val_accuracy: 0.8633
Epoch 1/10
4/4 [==============================] - 1s 237ms/step - loss: 0.5087 - accuracy:
0.8359 - val_loss: 0.3646 - val_accuracy: 0.8438
Epoch 2/10
4/4 [==============================] - 1s 189ms/step - loss: 0.2879 - accuracy:
0.8828 - val_loss: 0.4157 - val_accuracy: 0.8203
Epoch 3/10
4/4 [==============================] - 1s 188ms/step - loss: 0.1696 - accuracy:
0.9141 - val_loss: 0.4007 - val_accuracy: 0.8242
Epoch 4/10
4/4 [==============================] - 1s 174ms/step - loss: 0.1185 - accuracy:
0.9375 - val_loss: 0.3819 - val_accuracy: 0.8281
Epoch 5/10
4/4 [==============================] - 0s 150ms/step - loss: 0.0967 - accuracy:
0.9766 - val_loss: 0.3716 - val_accuracy: 0.8398
Epoch 6/10
4/4 [==============================] - 1s 192ms/step - loss: 0.0800 - accuracy:
0.9922 - val_loss: 0.3669 - val_accuracy: 0.8398
Epoch 7/10
4/4 [==============================] - 1s 198ms/step - loss: 0.0716 - accuracy:
1.0000 - val_loss: 0.3641 - val_accuracy: 0.8438
Epoch 8/10
4/4 [==============================] - 1s 172ms/step - loss: 0.0641 - accuracy:
1.0000 - val_loss: 0.3640 - val_accuracy: 0.8438
Epoch 9/10
4/4 [==============================] - 1s 160ms/step - loss: 0.0593 - accuracy:
1.0000 - val_loss: 0.3647 - val_accuracy: 0.8398
Epoch 10/10
4/4 [==============================] - 1s 211ms/step - loss: 0.0549 - accuracy:
1.0000 - val_loss: 0.3666 - val_accuracy: 0.8398
Epoch 1/10
4/4 [==============================] - 1s 265ms/step - loss: 0.4770 - accuracy:
0.7734 - val_loss: 0.3488 - val_accuracy: 0.8477
Epoch 2/10
4/4 [==============================] - 1s 177ms/step - loss: 0.2813 - accuracy:
0.8828 - val_loss: 0.3324 - val_accuracy: 0.8398
Epoch 3/10
4/4 [==============================] - 1s 185ms/step - loss: 0.1508 - accuracy:
0.9297 - val_loss: 0.4451 - val_accuracy: 0.8242
Epoch 4/10
4/4 [==============================] - 0s 139ms/step - loss: 0.1096 - accuracy:
0.9531 - val_loss: 0.6506 - val_accuracy: 0.7969
Epoch 5/10
4/4 [==============================] - 1s 160ms/step - loss: 0.0817 - accuracy:
0.9766 - val_loss: 0.8054 - val_accuracy: 0.7891
Epoch 6/10
```

```
4/4 [==============================] - 1s 184ms/step - loss: 0.0646 - accuracy:
0.9844 - val_loss: 0.8873 - val_accuracy: 0.7812
Epoch 7/10
4/4 [==============================] - 1s 167ms/step - loss: 0.0514 - accuracy:
0.9922 - val_loss: 0.9322 - val_accuracy: 0.7852
Epoch 8/10
4/4 [==============================] - 0s 148ms/step - loss: 0.0434 - accuracy:
1.0000 - val_loss: 0.9612 - val_accuracy: 0.7891
Epoch 9/10
4/4 [==============================] - 1s 185ms/step - loss: 0.0399 - accuracy:
1.0000 - val_loss: 0.9756 - val_accuracy: 0.7891
Epoch 10/10
4/4 [==============================] - 1s 171ms/step - loss: 0.0365 - accuracy:
1.0000 - val_loss: 0.9839 - val_accuracy: 0.7891
Epoch 1/10
4/4 [==============================] - 1s 238ms/step - loss: 0.8633 - accuracy:
0.8203 - val_loss: 0.5411 - val_accuracy: 0.7695
Epoch 2/10
4/4 [==============================] - 1s 179ms/step - loss: 0.5257 - accuracy:
0.7812 - val_loss: 0.8607 - val_accuracy: 0.6953
Epoch 3/10
4/4 [==============================] - 1s 172ms/step - loss: 0.4165 - accuracy:
0.8125 - val_loss: 0.8694 - val_accuracy: 0.6758
Epoch 4/10
4/4 [==============================] - 1s 208ms/step - loss: 0.1682 - accuracy:
0.9297 - val_loss: 0.7688 - val_accuracy: 0.7070
Epoch 5/10
4/4 [==============================] - 1s 174ms/step - loss: 0.1090 - accuracy:
0.9688 - val_loss: 0.6828 - val_accuracy: 0.7109
Epoch 6/10
4/4 [==============================] - 1s 223ms/step - loss: 0.0848 - accuracy:
0.9766 - val_loss: 0.6310 - val_accuracy: 0.7188
Epoch 7/10
4/4 [==============================] - 1s 166ms/step - loss: 0.0749 - accuracy:
0.9766 - val_loss: 0.5992 - val_accuracy: 0.7188
Epoch 8/10
4/4 [==============================] - 1s 200ms/step - loss: 0.0684 - accuracy:
0.9766 - val_loss: 0.5804 - val_accuracy: 0.7461
Epoch 9/10
4/4 [==============================] - 1s 171ms/step - loss: 0.0614 - accuracy:
0.9844 - val_loss: 0.5697 - val_accuracy: 0.7578
Epoch 10/10
4/4 [==============================] - 1s 202ms/step - loss: 0.0563 - accuracy:
1.0000 - val_loss: 0.5627 - val_accuracy: 0.7617
Epoch 1/10
4/4 [==============================] - 1s 257ms/step - loss: 0.6231 - accuracy:
0.7422 - val_loss: 0.5097 - val_accuracy: 0.7695
Epoch 2/10
```

```
4/4 [==============================] - 1s 181ms/step - loss: 0.3793 - accuracy:
0.8203 - val_loss: 0.4550 - val_accuracy: 0.7930
Epoch 3/10
4/4 [==============================] - 1s 230ms/step - loss: 0.2463 - accuracy:
0.8750 - val_loss: 0.4594 - val_accuracy: 0.7891
Epoch 4/10
4/4 [==============================] - 1s 180ms/step - loss: 0.1795 - accuracy:
0.9297 - val_loss: 0.4988 - val_accuracy: 0.7969
Epoch 5/10
4/4 [==============================] - 1s 180ms/step - loss: 0.1428 - accuracy:
0.9453 - val_loss: 0.5179 - val_accuracy: 0.7969
Epoch 6/10
4/4 [==============================] - 0s 153ms/step - loss: 0.1205 - accuracy:
0.9531 - val_loss: 0.5242 - val_accuracy: 0.8086
Epoch 7/10
4/4 [==============================] - 1s 181ms/step - loss: 0.1030 - accuracy:
0.9844 - val_loss: 0.5241 - val_accuracy: 0.8086
Epoch 8/10
4/4 [==============================] - 1s 209ms/step - loss: 0.0930 - accuracy:
0.9844 - val_loss: 0.5230 - val_accuracy: 0.8164
Epoch 9/10
4/4 [==============================] - 1s 166ms/step - loss: 0.0826 - accuracy:
0.9922 - val_loss: 0.5200 - val_accuracy: 0.8164
Epoch 10/10
4/4 [==============================] - 1s 203ms/step - loss: 0.0740 - accuracy:
0.9922 - val_loss: 0.5157 - val_accuracy: 0.8125
Epoch 1/10
4/4 [==============================] - 1s 227ms/step - loss: 0.3731 - accuracy:
0.8359 - val_loss: 0.3675 - val_accuracy: 0.8164
Epoch 2/10
4/4 [==============================] - 1s 191ms/step - loss: 0.1920 - accuracy:
0.9141 - val_loss: 0.3846 - val_accuracy: 0.8125
Epoch 3/10
4/4 [==============================] - 1s 196ms/step - loss: 0.1316 - accuracy:
0.9453 - val_loss: 0.4074 - val_accuracy: 0.7969
Epoch 4/10
4/4 [==============================] - 0s 128ms/step - loss: 0.0965 - accuracy:
0.9688 - val_loss: 0.4148 - val_accuracy: 0.8086
Epoch 5/10
4/4 [==============================] - 1s 190ms/step - loss: 0.0732 - accuracy:
0.9766 - val_loss: 0.4156 - val_accuracy: 0.8086
Epoch 6/10
4/4 [==============================] - 0s 156ms/step - loss: 0.0579 - accuracy:
0.9844 - val_loss: 0.4161 - val_accuracy: 0.8086
Epoch 7/10
4/4 [==============================] - 1s 185ms/step - loss: 0.0476 - accuracy:
0.9922 - val_loss: 0.4145 - val_accuracy: 0.8086
Epoch 8/10
```

```
4/4 [==============================] - 1s 167ms/step - loss: 0.0404 - accuracy:
0.9922 - val_loss: 0.4136 - val_accuracy: 0.8086
Epoch 9/10
4/4 [==============================] - 1s 177ms/step - loss: 0.0338 - accuracy:
0.9922 - val_loss: 0.4126 - val_accuracy: 0.8086
Epoch 10/10
4/4 [==============================] - 1s 198ms/step - loss: 0.0305 - accuracy:
1.0000 - val_loss: 0.4123 - val_accuracy: 0.8086
Epoch 1/10
4/4 [==============================] - 1s 244ms/step - loss: 0.5007 - accuracy:
0.8125 - val_loss: 0.3736 - val_accuracy: 0.8203
Epoch 2/10
4/4 [==============================] - 1s 195ms/step - loss: 0.3676 - accuracy:
0.8438 - val_loss: 0.3351 - val_accuracy: 0.8398
Epoch 3/10
4/4 [==============================] - 1s 175ms/step - loss: 0.2585 - accuracy:
0.9062 - val_loss: 0.3207 - val_accuracy: 0.8320
Epoch 4/10
4/4 [==============================] - 1s 196ms/step - loss: 0.2084 - accuracy:
0.9297 - val_loss: 0.3283 - val_accuracy: 0.8242
Epoch 5/10
4/4 [==============================] - 1s 171ms/step - loss: 0.1789 - accuracy:
0.9297 - val_loss: 0.3359 - val_accuracy: 0.8203
Epoch 6/10
4/4 [==============================] - 1s 150ms/step - loss: 0.1481 - accuracy:
0.9375 - val_loss: 0.3411 - val_accuracy: 0.8203
Epoch 7/10
4/4 [==============================] - 1s 165ms/step - loss: 0.1264 - accuracy:
0.9609 - val_loss: 0.3439 - val_accuracy: 0.8242
Epoch 8/10
4/4 [==============================] - 1s 161ms/step - loss: 0.1121 - accuracy:
0.9609 - val_loss: 0.3455 - val_accuracy: 0.8242
Epoch 9/10
4/4 [==============================] - 1s 167ms/step - loss: 0.0978 - accuracy:
0.9688 - val_loss: 0.3469 - val_accuracy: 0.8281
Epoch 10/10
4/4 [==============================] - 0s 146ms/step - loss: 0.0890 - accuracy:
0.9688 - val_loss: 0.3477 - val_accuracy: 0.8281
Epoch 1/10
4/4 [==============================] - 1s 231ms/step - loss: 0.5345 - accuracy:
0.7656 - val_loss: 0.3724 - val_accuracy: 0.8047
Epoch 2/10
4/4 [==============================] - 1s 173ms/step - loss: 0.3427 - accuracy:
0.8047 - val_loss: 0.4319 - val_accuracy: 0.7891
Epoch 3/10
4/4 [==============================] - 0s 130ms/step - loss: 0.2530 - accuracy:
0.8750 - val_loss: 0.4445 - val_accuracy: 0.7852
Epoch 4/10
```

```
4/4 [==============================] - 0s 136ms/step - loss: 0.1965 - accuracy:
0.9141 - val_loss: 0.4401 - val_accuracy: 0.7891
Epoch 5/10
4/4 [==============================] - 0s 144ms/step - loss: 0.1602 - accuracy:
0.9297 - val_loss: 0.4320 - val_accuracy: 0.7930
Epoch 6/10
4/4 [==============================] - 0s 156ms/step - loss: 0.1368 - accuracy:
0.9531 - val_loss: 0.4248 - val_accuracy: 0.7969
Epoch 7/10
4/4 [==============================] - 1s 170ms/step - loss: 0.1200 - accuracy:
0.9531 - val_loss: 0.4198 - val_accuracy: 0.7969
Epoch 8/10
4/4 [==============================] - 1s 159ms/step - loss: 0.1069 - accuracy:
0.9688 - val_loss: 0.4157 - val_accuracy: 0.8047
Epoch 9/10
4/4 [==============================] - 1s 187ms/step - loss: 0.0952 - accuracy:
0.9688 - val_loss: 0.4136 - val_accuracy: 0.8047
Epoch 10/10
4/4 [==============================] - 1s 180ms/step - loss: 0.0880 - accuracy:
0.9766 - val_loss: 0.4118 - val_accuracy: 0.8086
Epoch 1/10
4/4 [==============================] - 1s 201ms/step - loss: 0.4776 - accuracy:
0.8047 - val_loss: 0.3761 - val_accuracy: 0.8125
Epoch 2/10
4/4 [==============================] - 0s 139ms/step - loss: 0.3269 - accuracy:
0.8281 - val_loss: 0.3298 - val_accuracy: 0.8203
Epoch 3/10
4/4 [==============================] - 1s 160ms/step - loss: 0.2067 - accuracy:
0.8828 - val_loss: 0.3512 - val_accuracy: 0.8281
Epoch 4/10
4/4 [==============================] - 0s 147ms/step - loss: 0.1603 - accuracy:
0.9297 - val_loss: 0.3987 - val_accuracy: 0.8047
Epoch 5/10
4/4 [==============================] - 1s 159ms/step - loss: 0.1373 - accuracy:
0.9375 - val_loss: 0.4218 - val_accuracy: 0.8047
Epoch 6/10
4/4 [==============================] - 1s 178ms/step - loss: 0.1180 - accuracy:
0.9375 - val_loss: 0.4285 - val_accuracy: 0.7969
Epoch 7/10
4/4 [==============================] - 1s 206ms/step - loss: 0.1044 - accuracy:
0.9531 - val_loss: 0.4281 - val_accuracy: 0.8008
Epoch 8/10
4/4 [==============================] - 1s 176ms/step - loss: 0.0920 - accuracy:
0.9688 - val_loss: 0.4266 - val_accuracy: 0.8125
Epoch 9/10
4/4 [==============================] - 1s 190ms/step - loss: 0.0842 - accuracy:
0.9844 - val_loss: 0.4237 - val_accuracy: 0.8164
Epoch 10/10
```

```
4/4 [==============================] - 1s 205ms/step - loss: 0.0780 - accuracy:
0.9922 - val_loss: 0.4221 - val_accuracy: 0.8125
Epoch 1/10
4/4 [==============================] - 1s 187ms/step - loss: 0.4043 - accuracy:
0.8203 - val_loss: 0.3609 - val_accuracy: 0.8242
Epoch 2/10
4/4 [==============================] - 0s 146ms/step - loss: 0.2722 - accuracy:
0.8750 - val_loss: 0.3488 - val_accuracy: 0.8164
Epoch 3/10
4/4 [==============================] - 0s 158ms/step - loss: 0.1870 - accuracy:
0.8906 - val_loss: 0.3515 - val_accuracy: 0.8203
Epoch 4/10
4/4 [==============================] - 1s 173ms/step - loss: 0.1348 - accuracy:
0.9453 - val_loss: 0.3525 - val_accuracy: 0.8203
Epoch 5/10
4/4 [==============================] - 1s 169ms/step - loss: 0.1068 - accuracy:
0.9531 - val_loss: 0.3525 - val_accuracy: 0.8203
Epoch 6/10
4/4 [==============================] - 1s 160ms/step - loss: 0.0900 - accuracy:
0.9688 - val_loss: 0.3529 - val_accuracy: 0.8203
Epoch 7/10
4/4 [==============================] - 1s 161ms/step - loss: 0.0772 - accuracy:
0.9844 - val_loss: 0.3535 - val_accuracy: 0.8164
Epoch 8/10
4/4 [==============================] - 0s 142ms/step - loss: 0.0676 - accuracy:
0.9844 - val_loss: 0.3548 - val_accuracy: 0.8164
Epoch 9/10
4/4 [==============================] - 0s 151ms/step - loss: 0.0617 - accuracy:
0.9844 - val_loss: 0.3565 - val_accuracy: 0.8164
Epoch 10/10
4/4 [==============================] - 0s 139ms/step - loss: 0.0542 - accuracy:
0.9922 - val_loss: 0.3584 - val_accuracy: 0.8164
Epoch 1/10
4/4 [==============================] - 1s 206ms/step - loss: 0.5331 - accuracy:
0.7031 - val_loss: 0.3544 - val_accuracy: 0.8164
Epoch 2/10
4/4 [==============================] - 1s 226ms/step - loss: 0.3806 - accuracy:
0.7578 - val_loss: 0.3579 - val_accuracy: 0.8242
Epoch 3/10
4/4 [==============================] - 1s 172ms/step - loss: 0.2492 - accuracy:
0.8438 - val_loss: 0.3856 - val_accuracy: 0.8281
Epoch 4/10
4/4 [==============================] - 1s 176ms/step - loss: 0.1809 - accuracy:
0.9062 - val_loss: 0.4149 - val_accuracy: 0.8320
Epoch 5/10
4/4 [==============================] - 1s 173ms/step - loss: 0.1425 - accuracy:
0.9453 - val_loss: 0.4375 - val_accuracy: 0.8320
Epoch 6/10
```

```
4/4 [==============================] - 1s 164ms/step - loss: 0.1225 - accuracy:
0.9453 - val_loss: 0.4435 - val_accuracy: 0.8320
Epoch 7/10
4/4 [==============================] - 1s 150ms/step - loss: 0.1029 - accuracy:
0.9922 - val_loss: 0.4466 - val_accuracy: 0.8281
Epoch 8/10
4/4 [==============================] - 1s 177ms/step - loss: 0.0937 - accuracy:
0.9922 - val_loss: 0.4464 - val_accuracy: 0.8242
Epoch 9/10
4/4 [==============================] - 1s 165ms/step - loss: 0.0856 - accuracy:
1.0000 - val_loss: 0.4468 - val_accuracy: 0.8203
Epoch 10/10
4/4 [==============================] - 0s 152ms/step - loss: 0.0794 - accuracy:
1.0000 - val_loss: 0.4490 - val_accuracy: 0.8203
Epoch 1/10
4/4 [==============================] - 1s 262ms/step - loss: 0.3434 - accuracy:
0.8438 - val_loss: 0.4570 - val_accuracy: 0.8164
Epoch 2/10
4/4 [==============================] - 1s 217ms/step - loss: 0.1837 - accuracy:
0.9062 - val_loss: 0.4518 - val_accuracy: 0.8242
Epoch 3/10
4/4 [==============================] - 1s 165ms/step - loss: 0.1395 - accuracy:
0.9531 - val_loss: 0.4486 - val_accuracy: 0.8164
Epoch 4/10
4/4 [==============================] - 0s 149ms/step - loss: 0.1141 - accuracy:
0.9609 - val_loss: 0.4501 - val_accuracy: 0.8164
Epoch 5/10
4/4 [==============================] - 0s 158ms/step - loss: 0.0934 - accuracy:
0.9766 - val_loss: 0.4583 - val_accuracy: 0.8242
Epoch 6/10
4/4 [==============================] - 0s 120ms/step - loss: 0.0794 - accuracy:
0.9844 - val_loss: 0.4686 - val_accuracy: 0.8164
Epoch 7/10
4/4 [==============================] - 1s 153ms/step - loss: 0.0679 - accuracy:
0.9844 - val_loss: 0.4796 - val_accuracy: 0.8164
Epoch 8/10
4/4 [==============================] - 1s 154ms/step - loss: 0.0594 - accuracy:
0.9922 - val_loss: 0.4891 - val_accuracy: 0.8203
Epoch 9/10
4/4 [==============================] - 1s 151ms/step - loss: 0.0535 - accuracy:
0.9922 - val_loss: 0.4999 - val_accuracy: 0.8203
Epoch 10/10
4/4 [==============================] - 0s 141ms/step - loss: 0.0477 - accuracy:
0.9922 - val_loss: 0.5065 - val_accuracy: 0.8242
Epoch 1/10
4/4 [==============================] - 1s 235ms/step - loss: 0.3690 - accuracy:
0.8672 - val_loss: 0.3296 - val_accuracy: 0.8281
Epoch 2/10
```

```
4/4 [==============================] - 1s 172ms/step - loss: 0.2324 - accuracy:
0.9219 - val_loss: 0.3127 - val_accuracy: 0.8398
Epoch 3/10
4/4 [==============================] - 1s 190ms/step - loss: 0.1485 - accuracy:
0.9609 - val_loss: 0.3155 - val_accuracy: 0.8359
Epoch 4/10
4/4 [==============================] - 1s 163ms/step - loss: 0.1091 - accuracy:
0.9688 - val_loss: 0.3189 - val_accuracy: 0.8398
Epoch 5/10
4/4 [==============================] - 1s 199ms/step - loss: 0.0857 - accuracy:
0.9688 - val_loss: 0.3247 - val_accuracy: 0.8398
Epoch 6/10
4/4 [==============================] - 1s 171ms/step - loss: 0.0730 - accuracy:
0.9766 - val_loss: 0.3296 - val_accuracy: 0.8359
Epoch 7/10
4/4 [==============================] - 1s 202ms/step - loss: 0.0643 - accuracy:
0.9766 - val_loss: 0.3339 - val_accuracy: 0.8281
Epoch 8/10
4/4 [==============================] - 1s 178ms/step - loss: 0.0580 - accuracy:
0.9844 - val_loss: 0.3372 - val_accuracy: 0.8242
Epoch 9/10
4/4 [==============================] - 1s 185ms/step - loss: 0.0525 - accuracy:
0.9922 - val_loss: 0.3393 - val_accuracy: 0.8242
Epoch 10/10
4/4 [==============================] - 0s 146ms/step - loss: 0.0480 - accuracy:
0.9922 - val_loss: 0.3413 - val_accuracy: 0.8242
Epoch 1/10
4/4 [==============================] - 1s 263ms/step - loss: 0.3113 - accuracy:
0.8359 - val_loss: 0.3356 - val_accuracy: 0.8281
Epoch 2/10
4/4 [==============================] - 1s 190ms/step - loss: 0.2146 - accuracy:
0.9141 - val_loss: 0.3297 - val_accuracy: 0.8320
Epoch 3/10
4/4 [==============================] - 0s 142ms/step - loss: 0.1521 - accuracy:
0.9297 - val_loss: 0.3292 - val_accuracy: 0.8281
Epoch 4/10
4/4 [==============================] - 0s 142ms/step - loss: 0.1056 - accuracy:
0.9766 - val_loss: 0.3316 - val_accuracy: 0.8281
Epoch 5/10
4/4 [==============================] - 1s 165ms/step - loss: 0.0860 - accuracy:
0.9766 - val_loss: 0.3354 - val_accuracy: 0.8281
Epoch 6/10
4/4 [==============================] - 1s 171ms/step - loss: 0.0733 - accuracy:
0.9766 - val_loss: 0.3408 - val_accuracy: 0.8398
Epoch 7/10
4/4 [==============================] - 0s 160ms/step - loss: 0.0620 - accuracy:
1.0000 - val_loss: 0.3472 - val_accuracy: 0.8320
Epoch 8/10
```

```
4/4 [==============================] - 1s 177ms/step - loss: 0.0555 - accuracy:
1.0000 - val_loss: 0.3535 - val_accuracy: 0.8281
Epoch 9/10
4/4 [==============================] - 1s 168ms/step - loss: 0.0497 - accuracy:
1.0000 - val_loss: 0.3598 - val_accuracy: 0.8203
Epoch 10/10
4/4 [==============================] - 1s 189ms/step - loss: 0.0454 - accuracy:
1.0000 - val_loss: 0.3659 - val_accuracy: 0.8203
Epoch 1/10
4/4 [==============================] - 1s 236ms/step - loss: 0.3299 - accuracy:
0.8516 - val_loss: 0.3256 - val_accuracy: 0.8281
Epoch 2/10
4/4 [==============================] - 1s 212ms/step - loss: 0.1920 - accuracy:
0.8906 - val_loss: 0.3194 - val_accuracy: 0.8320
Epoch 3/10
4/4 [==============================] - 0s 146ms/step - loss: 0.1389 - accuracy:
0.9141 - val_loss: 0.3166 - val_accuracy: 0.8320
Epoch 4/10
4/4 [==============================] - 1s 173ms/step - loss: 0.1018 - accuracy:
0.9609 - val_loss: 0.3165 - val_accuracy: 0.8359
Epoch 5/10
4/4 [==============================] - 1s 194ms/step - loss: 0.0824 - accuracy:
0.9688 - val_loss: 0.3181 - val_accuracy: 0.8320
Epoch 6/10
4/4 [==============================] - 1s 175ms/step - loss: 0.0648 - accuracy:
0.9922 - val_loss: 0.3203 - val_accuracy: 0.8359
Epoch 7/10
4/4 [==============================] - 1s 168ms/step - loss: 0.0553 - accuracy:
1.0000 - val_loss: 0.3229 - val_accuracy: 0.8359
Epoch 8/10
4/4 [==============================] - 0s 94ms/step - loss: 0.0475 - accuracy:
1.0000 - val_loss: 0.3257 - val_accuracy: 0.8359
Epoch 9/10
4/4 [==============================] - 0s 146ms/step - loss: 0.0423 - accuracy:
1.0000 - val_loss: 0.3285 - val_accuracy: 0.8359
Epoch 10/10
4/4 [==============================] - 0s 160ms/step - loss: 0.0386 - accuracy:
1.0000 - val_loss: 0.3314 - val_accuracy: 0.8359
Epoch 1/10
4/4 [==============================] - 1s 233ms/step - loss: 0.2394 - accuracy:
0.8594 - val_loss: 0.3272 - val_accuracy: 0.8438
Epoch 2/10
4/4 [==============================] - 1s 171ms/step - loss: 0.1268 - accuracy:
0.9453 - val_loss: 0.3477 - val_accuracy: 0.8438
Epoch 3/10
4/4 [==============================] - 1s 180ms/step - loss: 0.0774 - accuracy:
0.9844 - val_loss: 0.4019 - val_accuracy: 0.8438
Epoch 4/10
```

```
4/4 [==============================] - 0s 152ms/step - loss: 0.0565 - accuracy:
1.0000 - val_loss: 0.4679 - val_accuracy: 0.8438
Epoch 5/10
4/4 [==============================] - 1s 192ms/step - loss: 0.0452 - accuracy:
1.0000 - val_loss: 0.5246 - val_accuracy: 0.8477
Epoch 6/10
4/4 [==============================] - 0s 123ms/step - loss: 0.0368 - accuracy:
1.0000 - val_loss: 0.5705 - val_accuracy: 0.8438
Epoch 7/10
4/4 [==============================] - 0s 147ms/step - loss: 0.0319 - accuracy:
1.0000 - val_loss: 0.6083 - val_accuracy: 0.8398
Epoch 8/10
4/4 [==============================] - 1s 167ms/step - loss: 0.0281 - accuracy:
1.0000 - val_loss: 0.6351 - val_accuracy: 0.8398
Epoch 9/10
4/4 [==============================] - 1s 162ms/step - loss: 0.0253 - accuracy:
1.0000 - val_loss: 0.6549 - val_accuracy: 0.8398
Epoch 10/10
4/4 [==============================] - 1s 153ms/step - loss: 0.0234 - accuracy:
1.0000 - val_loss: 0.6708 - val_accuracy: 0.8438
Epoch 1/10
4/4 [==============================] - 1s 215ms/step - loss: 0.3757 - accuracy:
0.8438 - val_loss: 0.4203 - val_accuracy: 0.8555
Epoch 2/10
4/4 [==============================] - 0s 153ms/step - loss: 0.2765 - accuracy:
0.8594 - val_loss: 0.3433 - val_accuracy: 0.8594
Epoch 3/10
4/4 [==============================] - 1s 157ms/step - loss: 0.1895 - accuracy:
0.9062 - val_loss: 0.3247 - val_accuracy: 0.8477
Epoch 4/10
4/4 [==============================] - 1s 182ms/step - loss: 0.1348 - accuracy:
0.9609 - val_loss: 0.3142 - val_accuracy: 0.8516
Epoch 5/10
4/4 [==============================] - 1s 187ms/step - loss: 0.0930 - accuracy:
0.9766 - val_loss: 0.3091 - val_accuracy: 0.8555
Epoch 6/10
4/4 [==============================] - 1s 222ms/step - loss: 0.0747 - accuracy:
0.9844 - val_loss: 0.3060 - val_accuracy: 0.8594
Epoch 7/10
4/4 [==============================] - 1s 152ms/step - loss: 0.0642 - accuracy:
0.9844 - val_loss: 0.3045 - val_accuracy: 0.8633
Epoch 8/10
4/4 [==============================] - 0s 154ms/step - loss: 0.0577 - accuracy:
0.9844 - val_loss: 0.3035 - val_accuracy: 0.8633
Epoch 9/10
4/4 [==============================] - 1s 160ms/step - loss: 0.0515 - accuracy:
0.9844 - val_loss: 0.3033 - val_accuracy: 0.8672
Epoch 10/10
```

```
4/4 [==============================] - 0s 156ms/step - loss: 0.0476 - accuracy:
0.9922 - val_loss: 0.3032 - val_accuracy: 0.8672
Epoch 1/10
4/4 [==============================] - 1s 237ms/step - loss: 0.4669 - accuracy:
0.8594 - val_loss: 0.3368 - val_accuracy: 0.8477
Epoch 2/10
4/4 [==============================] - 1s 197ms/step - loss: 0.1992 - accuracy:
0.9219 - val_loss: 0.3811 - val_accuracy: 0.8125
Epoch 3/10
4/4 [==============================] - 1s 166ms/step - loss: 0.1458 - accuracy:
0.9609 - val_loss: 0.3950 - val_accuracy: 0.8086
Epoch 4/10
4/4 [==============================] - 0s 143ms/step - loss: 0.1050 - accuracy:
0.9766 - val_loss: 0.3951 - val_accuracy: 0.8086
Epoch 5/10
4/4 [==============================] - 1s 169ms/step - loss: 0.0737 - accuracy:
0.9766 - val_loss: 0.3924 - val_accuracy: 0.8086
Epoch 6/10
4/4 [==============================] - 1s 215ms/step - loss: 0.0585 - accuracy:
0.9844 - val_loss: 0.3903 - val_accuracy: 0.8164
Epoch 7/10
4/4 [==============================] - 1s 168ms/step - loss: 0.0478 - accuracy:
0.9844 - val_loss: 0.3886 - val_accuracy: 0.8164
Epoch 8/10
4/4 [==============================] - 1s 176ms/step - loss: 0.0408 - accuracy:
0.9922 - val_loss: 0.3882 - val_accuracy: 0.8164
Epoch 9/10
4/4 [==============================] - 1s 180ms/step - loss: 0.0351 - accuracy:
0.9922 - val_loss: 0.3883 - val_accuracy: 0.8203
Epoch 10/10
4/4 [==============================] - 0s 124ms/step - loss: 0.0310 - accuracy:
0.9922 - val_loss: 0.3893 - val_accuracy: 0.8203
Epoch 1/10
4/4 [==============================] - 1s 230ms/step - loss: 0.4739 - accuracy:
0.7969 - val_loss: 0.3468 - val_accuracy: 0.8398
Epoch 2/10
4/4 [==============================] - 1s 195ms/step - loss: 0.2915 - accuracy:
0.8672 - val_loss: 0.3149 - val_accuracy: 0.8594
Epoch 3/10
4/4 [==============================] - 1s 170ms/step - loss: 0.2218 - accuracy:
0.8984 - val_loss: 0.3398 - val_accuracy: 0.8555
Epoch 4/10
4/4 [==============================] - 1s 202ms/step - loss: 0.1613 - accuracy:
0.9453 - val_loss: 0.3414 - val_accuracy: 0.8516
Epoch 5/10
4/4 [==============================] - 1s 198ms/step - loss: 0.1272 - accuracy:
0.9609 - val_loss: 0.3447 - val_accuracy: 0.8516
Epoch 6/10
```

```
4/4 [==============================] - 1s 165ms/step - loss: 0.0911 - accuracy:
0.9688 - val_loss: 0.3483 - val_accuracy: 0.8516
Epoch 7/10
4/4 [==============================] - 0s 155ms/step - loss: 0.0715 - accuracy:
0.9844 - val_loss: 0.3522 - val_accuracy: 0.8555
Epoch 8/10
4/4 [==============================] - 1s 173ms/step - loss: 0.0576 - accuracy:
0.9844 - val_loss: 0.3556 - val_accuracy: 0.8555
Epoch 9/10
4/4 [==============================] - 1s 192ms/step - loss: 0.0467 - accuracy:
0.9922 - val_loss: 0.3592 - val_accuracy: 0.8516
Epoch 10/10
4/4 [==============================] - 1s 149ms/step - loss: 0.0412 - accuracy:
0.9922 - val_loss: 0.3622 - val_accuracy: 0.8516
Epoch 1/10
4/4 [==============================] - 1s 199ms/step - loss: 0.4590 - accuracy:
0.7734 - val_loss: 0.3232 - val_accuracy: 0.8633
Epoch 2/10
4/4 [==============================] - 1s 188ms/step - loss: 0.2980 - accuracy:
0.8203 - val_loss: 0.3563 - val_accuracy: 0.8398
Epoch 3/10
4/4 [==============================] - 0s 145ms/step - loss: 0.2005 - accuracy:
0.8750 - val_loss: 0.3841 - val_accuracy: 0.8242
Epoch 4/10
4/4 [==============================] - 1s 188ms/step - loss: 0.1403 - accuracy:
0.9531 - val_loss: 0.3912 - val_accuracy: 0.8242
Epoch 5/10
4/4 [==============================] - 1s 193ms/step - loss: 0.0979 - accuracy:
0.9766 - val_loss: 0.3909 - val_accuracy: 0.8242
Epoch 6/10
4/4 [==============================] - 1s 180ms/step - loss: 0.0756 - accuracy:
0.9922 - val_loss: 0.3872 - val_accuracy: 0.8203
Epoch 7/10
4/4 [==============================] - 1s 187ms/step - loss: 0.0600 - accuracy:
1.0000 - val_loss: 0.3847 - val_accuracy: 0.8281
Epoch 8/10
4/4 [==============================] - 0s 137ms/step - loss: 0.0505 - accuracy:
1.0000 - val_loss: 0.3832 - val_accuracy: 0.8320
Epoch 9/10
4/4 [==============================] - 1s 178ms/step - loss: 0.0443 - accuracy:
1.0000 - val_loss: 0.3828 - val_accuracy: 0.8359
Epoch 10/10
4/4 [==============================] - 1s 158ms/step - loss: 0.0391 - accuracy:
1.0000 - val_loss: 0.3825 - val_accuracy: 0.8359
Epoch 1/10
4/4 [==============================] - 1s 234ms/step - loss: 0.4810 - accuracy:
0.7969 - val_loss: 0.3310 - val_accuracy: 0.8594
Epoch 2/10
```

```
4/4 [==============================] - 1s 165ms/step - loss: 0.2453 - accuracy:
0.8750 - val_loss: 0.3255 - val_accuracy: 0.8750
Epoch 3/10
4/4 [==============================] - 0s 147ms/step - loss: 0.1496 - accuracy:
0.9219 - val_loss: 0.4352 - val_accuracy: 0.8516
Epoch 4/10
4/4 [==============================] - 1s 167ms/step - loss: 0.0979 - accuracy:
0.9688 - val_loss: 0.4342 - val_accuracy: 0.8516
Epoch 5/10
4/4 [==============================] - 1s 181ms/step - loss: 0.0721 - accuracy:
0.9922 - val_loss: 0.4242 - val_accuracy: 0.8516
Epoch 6/10
4/4 [==============================] - 1s 185ms/step - loss: 0.0551 - accuracy:
0.9922 - val_loss: 0.4203 - val_accuracy: 0.8555
Epoch 7/10
4/4 [==============================] - 1s 190ms/step - loss: 0.0443 - accuracy:
1.0000 - val_loss: 0.4221 - val_accuracy: 0.8594
Epoch 8/10
4/4 [==============================] - 1s 180ms/step - loss: 0.0382 - accuracy:
1.0000 - val_loss: 0.4244 - val_accuracy: 0.8594
Epoch 9/10
4/4 [==============================] - 1s 166ms/step - loss: 0.0333 - accuracy:
1.0000 - val_loss: 0.4288 - val_accuracy: 0.8594
Epoch 10/10
4/4 [==============================] - 0s 149ms/step - loss: 0.0293 - accuracy:
1.0000 - val_loss: 0.4342 - val_accuracy: 0.8594
Epoch 1/10
4/4 [==============================] - 1s 245ms/step - loss: 0.4722 - accuracy:
0.8594 - val_loss: 0.3223 - val_accuracy: 0.8594
Epoch 2/10
4/4 [==============================] - 1s 160ms/step - loss: 0.2630 - accuracy:
0.9141 - val_loss: 0.3287 - val_accuracy: 0.8516
Epoch 3/10
4/4 [==============================] - 1s 172ms/step - loss: 0.1666 - accuracy:
0.9219 - val_loss: 0.3270 - val_accuracy: 0.8477
Epoch 4/10
4/4 [==============================] - 1s 156ms/step - loss: 0.1084 - accuracy:
0.9609 - val_loss: 0.3243 - val_accuracy: 0.8477
Epoch 5/10
4/4 [==============================] - 1s 156ms/step - loss: 0.0774 - accuracy:
0.9766 - val_loss: 0.3237 - val_accuracy: 0.8477
Epoch 6/10
4/4 [==============================] - 0s 149ms/step - loss: 0.0532 - accuracy:
1.0000 - val_loss: 0.3252 - val_accuracy: 0.8516
Epoch 7/10
4/4 [==============================] - 1s 191ms/step - loss: 0.0416 - accuracy:
1.0000 - val_loss: 0.3272 - val_accuracy: 0.8516
Epoch 8/10
```

```
4/4 [==============================] - 0s 148ms/step - loss: 0.0340 - accuracy:
1.0000 - val_loss: 0.3292 - val_accuracy: 0.8555
Epoch 9/10
4/4 [==============================] - 1s 157ms/step - loss: 0.0299 - accuracy:
1.0000 - val_loss: 0.3311 - val_accuracy: 0.8594
Epoch 10/10
4/4 [==============================] - 1s 183ms/step - loss: 0.0269 - accuracy:
1.0000 - val_loss: 0.3331 - val_accuracy: 0.8594
Epoch 1/10
4/4 [==============================] - 1s 248ms/step - loss: 0.4711 - accuracy:
0.7812 - val_loss: 0.3372 - val_accuracy: 0.8594
Epoch 2/10
4/4 [==============================] - 1s 183ms/step - loss: 0.3537 - accuracy:
0.8359 - val_loss: 0.3400 - val_accuracy: 0.8555
Epoch 3/10
4/4 [==============================] - 0s 151ms/step - loss: 0.2711 - accuracy:
0.8438 - val_loss: 0.3375 - val_accuracy: 0.8555
Epoch 4/10
4/4 [==============================] - 0s 138ms/step - loss: 0.1989 - accuracy:
0.8984 - val_loss: 0.3336 - val_accuracy: 0.8633
Epoch 5/10
4/4 [==============================] - 0s 130ms/step - loss: 0.1447 - accuracy:
0.9219 - val_loss: 0.3303 - val_accuracy: 0.8555
Epoch 6/10
4/4 [==============================] - 0s 116ms/step - loss: 0.1064 - accuracy:
0.9531 - val_loss: 0.3261 - val_accuracy: 0.8516
Epoch 7/10
4/4 [==============================] - 0s 120ms/step - loss: 0.0794 - accuracy:
0.9766 - val_loss: 0.3237 - val_accuracy: 0.8516
Epoch 8/10
4/4 [==============================] - 0s 132ms/step - loss: 0.0651 - accuracy:
0.9766 - val_loss: 0.3217 - val_accuracy: 0.8555
Epoch 9/10
4/4 [==============================] - 0s 123ms/step - loss: 0.0539 - accuracy:
0.9844 - val_loss: 0.3205 - val_accuracy: 0.8555
Epoch 10/10
4/4 [==============================] - 0s 114ms/step - loss: 0.0463 - accuracy:
0.9922 - val_loss: 0.3193 - val_accuracy: 0.8555
Epoch 1/10
4/4 [==============================] - 1s 235ms/step - loss: 0.4263 - accuracy:
0.7734 - val_loss: 0.3144 - val_accuracy: 0.8555
Epoch 2/10
4/4 [==============================] - 1s 162ms/step - loss: 0.2831 - accuracy:
0.8438 - val_loss: 0.3015 - val_accuracy: 0.8711
Epoch 3/10
4/4 [==============================] - 1s 181ms/step - loss: 0.1777 - accuracy:
0.8984 - val_loss: 0.2941 - val_accuracy: 0.8711
Epoch 4/10
```

```
4/4 [==============================] - 1s 159ms/step - loss: 0.1276 - accuracy:
0.9453 - val_loss: 0.2942 - val_accuracy: 0.8672
Epoch 5/10
4/4 [==============================] - 1s 169ms/step - loss: 0.0953 - accuracy:
0.9609 - val_loss: 0.2980 - val_accuracy: 0.8750
Epoch 6/10
4/4 [==============================] - 1s 171ms/step - loss: 0.0770 - accuracy:
0.9766 - val_loss: 0.3023 - val_accuracy: 0.8711
Epoch 7/10
4/4 [==============================] - 0s 130ms/step - loss: 0.0627 - accuracy:
0.9766 - val_loss: 0.3065 - val_accuracy: 0.8711
Epoch 8/10
4/4 [==============================] - 0s 147ms/step - loss: 0.0528 - accuracy:
0.9844 - val_loss: 0.3104 - val_accuracy: 0.8711
Epoch 9/10
4/4 [==============================] - 0s 153ms/step - loss: 0.0458 - accuracy:
0.9922 - val_loss: 0.3140 - val_accuracy: 0.8750
Epoch 10/10
4/4 [==============================] - 1s 188ms/step - loss: 0.0406 - accuracy:
0.9922 - val_loss: 0.3175 - val_accuracy: 0.8750
Epoch 1/10
4/4 [==============================] - 1s 204ms/step - loss: 0.3676 - accuracy:
0.8281 - val_loss: 0.2952 - val_accuracy: 0.8750
Epoch 2/10
4/4 [==============================] - 1s 194ms/step - loss: 0.2521 - accuracy:
0.8672 - val_loss: 0.2854 - val_accuracy: 0.8750
Epoch 3/10
4/4 [==============================] - 1s 191ms/step - loss: 0.1650 - accuracy:
0.9141 - val_loss: 0.2850 - val_accuracy: 0.8750
Epoch 4/10
4/4 [==============================] - 1s 194ms/step - loss: 0.1135 - accuracy:
0.9375 - val_loss: 0.2848 - val_accuracy: 0.8789
Epoch 5/10
4/4 [==============================] - 1s 156ms/step - loss: 0.0857 - accuracy:
0.9531 - val_loss: 0.2842 - val_accuracy: 0.8789
Epoch 6/10
4/4 [==============================] - 1s 219ms/step - loss: 0.0712 - accuracy:
0.9766 - val_loss: 0.2829 - val_accuracy: 0.8750
Epoch 7/10
4/4 [==============================] - 1s 197ms/step - loss: 0.0584 - accuracy:
0.9922 - val_loss: 0.2822 - val_accuracy: 0.8789
Epoch 8/10
4/4 [==============================] - 1s 186ms/step - loss: 0.0497 - accuracy:
0.9922 - val_loss: 0.2821 - val_accuracy: 0.8750
Epoch 9/10
4/4 [==============================] - 0s 144ms/step - loss: 0.0443 - accuracy:
0.9922 - val_loss: 0.2825 - val_accuracy: 0.8750
Epoch 10/10
```

```
4/4 [==============================] - 1s 188ms/step - loss: 0.0389 - accuracy:
0.9922 - val_loss: 0.2831 - val_accuracy: 0.8750
Epoch 1/10
4/4 [==============================] - 1s 227ms/step - loss: 0.5126 - accuracy:
0.8203 - val_loss: 0.2728 - val_accuracy: 0.8516
Epoch 2/10
4/4 [==============================] - 1s 149ms/step - loss: 0.2579 - accuracy:
0.8828 - val_loss: 0.2836 - val_accuracy: 0.8477
Epoch 3/10
4/4 [==============================] - 0s 138ms/step - loss: 0.1878 - accuracy:
0.9297 - val_loss: 0.2900 - val_accuracy: 0.8398
Epoch 4/10
4/4 [==============================] - 0s 158ms/step - loss: 0.1359 - accuracy:
0.9453 - val_loss: 0.2933 - val_accuracy: 0.8281
Epoch 5/10
4/4 [==============================] - 1s 155ms/step - loss: 0.1029 - accuracy:
0.9688 - val_loss: 0.2946 - val_accuracy: 0.8359
Epoch 6/10
4/4 [==============================] - 1s 200ms/step - loss: 0.0773 - accuracy:
0.9766 - val_loss: 0.2943 - val_accuracy: 0.8359
Epoch 7/10
4/4 [==============================] - 1s 156ms/step - loss: 0.0630 - accuracy:
0.9844 - val_loss: 0.2945 - val_accuracy: 0.8359
Epoch 8/10
4/4 [==============================] - 1s 167ms/step - loss: 0.0511 - accuracy:
0.9844 - val_loss: 0.2948 - val_accuracy: 0.8359
Epoch 9/10
4/4 [==============================] - 0s 158ms/step - loss: 0.0428 - accuracy:
1.0000 - val_loss: 0.2959 - val_accuracy: 0.8359
Epoch 10/10
4/4 [==============================] - 1s 170ms/step - loss: 0.0358 - accuracy:
1.0000 - val_loss: 0.2967 - val_accuracy: 0.8320
Epoch 1/10
4/4 [==============================] - 1s 240ms/step - loss: 0.2742 - accuracy:
0.8750 - val_loss: 0.2876 - val_accuracy: 0.8438
Epoch 2/10
4/4 [==============================] - 1s 187ms/step - loss: 0.2064 - accuracy:
0.9062 - val_loss: 0.2778 - val_accuracy: 0.8438
Epoch 3/10
4/4 [==============================] - 1s 174ms/step - loss: 0.1561 - accuracy:
0.9375 - val_loss: 0.2728 - val_accuracy: 0.8594
Epoch 4/10
4/4 [==============================] - 1s 213ms/step - loss: 0.1111 - accuracy:
0.9531 - val_loss: 0.2709 - val_accuracy: 0.8672
Epoch 5/10
4/4 [==============================] - 1s 166ms/step - loss: 0.0809 - accuracy:
0.9688 - val_loss: 0.2712 - val_accuracy: 0.8672
Epoch 6/10
```

```
4/4 [==============================] - 1s 165ms/step - loss: 0.0607 - accuracy:
0.9766 - val_loss: 0.2725 - val_accuracy: 0.8672
Epoch 7/10
4/4 [==============================] - 0s 146ms/step - loss: 0.0466 - accuracy:
0.9922 - val_loss: 0.2740 - val_accuracy: 0.8711
Epoch 8/10
4/4 [==============================] - 0s 151ms/step - loss: 0.0365 - accuracy:
0.9922 - val_loss: 0.2752 - val_accuracy: 0.8711
Epoch 9/10
4/4 [==============================] - 1s 186ms/step - loss: 0.0302 - accuracy:
1.0000 - val_loss: 0.2765 - val_accuracy: 0.8789
Epoch 10/10
4/4 [==============================] - 1s 161ms/step - loss: 0.0263 - accuracy:
1.0000 - val_loss: 0.2778 - val_accuracy: 0.8789
Epoch 1/10
4/4 [==============================] - 1s 228ms/step - loss: 0.4637 - accuracy:
0.8750 - val_loss: 0.2733 - val_accuracy: 0.8750
Epoch 2/10
4/4 [==============================] - 1s 191ms/step - loss: 0.1851 - accuracy:
0.9297 - val_loss: 0.2835 - val_accuracy: 0.8555
Epoch 3/10
4/4 [==============================] - 1s 175ms/step - loss: 0.0999 - accuracy:
0.9766 - val_loss: 0.2966 - val_accuracy: 0.8516
Epoch 4/10
4/4 [==============================] - 1s 213ms/step - loss: 0.0582 - accuracy:
0.9922 - val_loss: 0.3011 - val_accuracy: 0.8516
Epoch 5/10
4/4 [==============================] - 1s 261ms/step - loss: 0.0394 - accuracy:
1.0000 - val_loss: 0.3028 - val_accuracy: 0.8516
Epoch 6/10
4/4 [==============================] - 1s 202ms/step - loss: 0.0293 - accuracy:
1.0000 - val_loss: 0.3038 - val_accuracy: 0.8516
Epoch 7/10
4/4 [==============================] - 1s 199ms/step - loss: 0.0225 - accuracy:
1.0000 - val_loss: 0.3045 - val_accuracy: 0.8594
Epoch 8/10
4/4 [==============================] - 1s 171ms/step - loss: 0.0182 - accuracy:
1.0000 - val_loss: 0.3051 - val_accuracy: 0.8594
Epoch 9/10
4/4 [==============================] - 1s 200ms/step - loss: 0.0154 - accuracy:
1.0000 - val_loss: 0.3059 - val_accuracy: 0.8594
Epoch 10/10
4/4 [==============================] - 1s 193ms/step - loss: 0.0134 - accuracy:
1.0000 - val_loss: 0.3065 - val_accuracy: 0.8594
Epoch 1/10
4/4 [==============================] - 1s 275ms/step - loss: 0.4296 - accuracy:
0.7969 - val_loss: 0.3005 - val_accuracy: 0.8672
Epoch 2/10
```

```
4/4 [==============================] - 1s 202ms/step - loss: 0.2282 - accuracy:
0.9375 - val_loss: 0.3403 - val_accuracy: 0.8398
Epoch 3/10
4/4 [==============================] - 1s 200ms/step - loss: 0.1651 - accuracy:
0.9453 - val_loss: 0.3704 - val_accuracy: 0.8477
Epoch 4/10
4/4 [==============================] - 1s 202ms/step - loss: 0.1299 - accuracy:
0.9688 - val_loss: 0.3791 - val_accuracy: 0.8477
Epoch 5/10
4/4 [==============================] - 1s 218ms/step - loss: 0.1009 - accuracy:
0.9688 - val_loss: 0.3765 - val_accuracy: 0.8516
Epoch 6/10
4/4 [==============================] - 1s 172ms/step - loss: 0.0785 - accuracy:
0.9688 - val_loss: 0.3729 - val_accuracy: 0.8477
Epoch 7/10
4/4 [==============================] - 1s 229ms/step - loss: 0.0645 - accuracy:
0.9766 - val_loss: 0.3719 - val_accuracy: 0.8359
Epoch 8/10
4/4 [==============================] - 1s 200ms/step - loss: 0.0539 - accuracy:
0.9766 - val_loss: 0.3733 - val_accuracy: 0.8320
Epoch 9/10
4/4 [==============================] - 1s 215ms/step - loss: 0.0439 - accuracy:
0.9844 - val_loss: 0.3752 - val_accuracy: 0.8320
Epoch 10/10
4/4 [==============================] - 1s 183ms/step - loss: 0.0370 - accuracy:
0.9922 - val_loss: 0.3778 - val_accuracy: 0.8281
Epoch 1/10
4/4 [==============================] - 1s 274ms/step - loss: 0.4318 - accuracy:
0.7812 - val_loss: 0.3022 - val_accuracy: 0.8477
Epoch 2/10
4/4 [==============================] - 1s 193ms/step - loss: 0.2319 - accuracy:
0.8594 - val_loss: 0.3207 - val_accuracy: 0.8359
Epoch 3/10
4/4 [==============================] - 1s 207ms/step - loss: 0.1510 - accuracy:
0.9297 - val_loss: 0.3279 - val_accuracy: 0.8320
Epoch 4/10
4/4 [==============================] - 1s 231ms/step - loss: 0.0987 - accuracy:
0.9688 - val_loss: 0.3269 - val_accuracy: 0.8438
Epoch 5/10
4/4 [==============================] - 1s 249ms/step - loss: 0.0650 - accuracy:
1.0000 - val_loss: 0.3234 - val_accuracy: 0.8477
Epoch 6/10
4/4 [==============================] - 1s 186ms/step - loss: 0.0485 - accuracy:
1.0000 - val_loss: 0.3196 - val_accuracy: 0.8555
Epoch 7/10
4/4 [==============================] - 1s 213ms/step - loss: 0.0379 - accuracy:
1.0000 - val_loss: 0.3168 - val_accuracy: 0.8594
Epoch 8/10
```

```
4/4 [==============================] - 0s 151ms/step - loss: 0.0307 - accuracy:
1.0000 - val_loss: 0.3153 - val_accuracy: 0.8594
Epoch 9/10
4/4 [==============================] - 1s 192ms/step - loss: 0.0262 - accuracy:
1.0000 - val_loss: 0.3146 - val_accuracy: 0.8672
Epoch 10/10
4/4 [==============================] - 1s 189ms/step - loss: 0.0233 - accuracy:
1.0000 - val_loss: 0.3141 - val_accuracy: 0.8672
Epoch 1/10
4/4 [==============================] - 1s 210ms/step - loss: 0.3230 - accuracy:
0.8516 - val_loss: 0.3399 - val_accuracy: 0.8438
Epoch 2/10
4/4 [==============================] - 1s 156ms/step - loss: 0.1965 - accuracy:
0.8828 - val_loss: 0.3333 - val_accuracy: 0.8398
Epoch 3/10
4/4 [==============================] - 0s 147ms/step - loss: 0.1099 - accuracy:
0.9453 - val_loss: 0.3206 - val_accuracy: 0.8438
Epoch 4/10
4/4 [==============================] - 1s 194ms/step - loss: 0.0689 - accuracy:
0.9688 - val_loss: 0.3101 - val_accuracy: 0.8438
Epoch 5/10
4/4 [==============================] - 1s 177ms/step - loss: 0.0477 - accuracy:
0.9844 - val_loss: 0.3027 - val_accuracy: 0.8477
Epoch 6/10
4/4 [==============================] - 0s 126ms/step - loss: 0.0360 - accuracy:
0.9922 - val_loss: 0.2975 - val_accuracy: 0.8516
Epoch 7/10
4/4 [==============================] - 1s 156ms/step - loss: 0.0264 - accuracy:
1.0000 - val_loss: 0.2947 - val_accuracy: 0.8594
Epoch 8/10
4/4 [==============================] - 0s 116ms/step - loss: 0.0218 - accuracy:
1.0000 - val_loss: 0.2932 - val_accuracy: 0.8672
Epoch 9/10
4/4 [==============================] - 0s 101ms/step - loss: 0.0184 - accuracy:
1.0000 - val_loss: 0.2930 - val_accuracy: 0.8672
Epoch 10/10
4/4 [==============================] - 0s 141ms/step - loss: 0.0156 - accuracy:
1.0000 - val_loss: 0.2930 - val_accuracy: 0.8672
Epoch 1/10
4/4 [==============================] - 1s 238ms/step - loss: 0.2299 - accuracy:
0.8984 - val_loss: 0.2743 - val_accuracy: 0.8828
Epoch 2/10
4/4 [==============================] - 1s 170ms/step - loss: 0.1367 - accuracy:
0.9297 - val_loss: 0.2592 - val_accuracy: 0.8984
Epoch 3/10
4/4 [==============================] - 1s 160ms/step - loss: 0.0889 - accuracy:
0.9688 - val_loss: 0.2613 - val_accuracy: 0.8828
Epoch 4/10
```

```
4/4 [==============================] - 1s 221ms/step - loss: 0.0620 - accuracy:
0.9844 - val_loss: 0.2738 - val_accuracy: 0.8945
Epoch 5/10
4/4 [==============================] - 1s 159ms/step - loss: 0.0465 - accuracy:
1.0000 - val_loss: 0.2851 - val_accuracy: 0.8867
Epoch 6/10
4/4 [==============================] - 1s 209ms/step - loss: 0.0332 - accuracy:
1.0000 - val_loss: 0.2927 - val_accuracy: 0.8867
Epoch 7/10
4/4 [==============================] - 1s 179ms/step - loss: 0.0253 - accuracy:
1.0000 - val_loss: 0.2981 - val_accuracy: 0.8789
Epoch 8/10
4/4 [==============================] - 1s 167ms/step - loss: 0.0200 - accuracy:
1.0000 - val_loss: 0.3023 - val_accuracy: 0.8750
Epoch 9/10
4/4 [==============================] - 1s 160ms/step - loss: 0.0166 - accuracy:
1.0000 - val_loss: 0.3055 - val_accuracy: 0.8711
Epoch 10/10
4/4 [==============================] - 1s 192ms/step - loss: 0.0144 - accuracy:
1.0000 - val_loss: 0.3084 - val_accuracy: 0.8750
Epoch 1/10
4/4 [==============================] - 1s 234ms/step - loss: 0.4938 - accuracy:
0.8594 - val_loss: 0.2918 - val_accuracy: 0.8789
Epoch 2/10
4/4 [==============================] - 1s 183ms/step - loss: 0.1932 - accuracy:
0.9297 - val_loss: 0.2943 - val_accuracy: 0.8711
Epoch 3/10
4/4 [==============================] - 1s 158ms/step - loss: 0.1089 - accuracy:
0.9688 - val_loss: 0.3076 - val_accuracy: 0.8594
Epoch 4/10
4/4 [==============================] - 1s 174ms/step - loss: 0.0827 - accuracy:
0.9766 - val_loss: 0.3199 - val_accuracy: 0.8750
Epoch 5/10
4/4 [==============================] - 1s 199ms/step - loss: 0.0618 - accuracy:
0.9766 - val_loss: 0.3286 - val_accuracy: 0.8711
Epoch 6/10
4/4 [==============================] - 1s 225ms/step - loss: 0.0464 - accuracy:
0.9844 - val_loss: 0.3338 - val_accuracy: 0.8672
Epoch 7/10
4/4 [==============================] - 1s 218ms/step - loss: 0.0375 - accuracy:
0.9922 - val_loss: 0.3371 - val_accuracy: 0.8633
Epoch 8/10
4/4 [==============================] - 1s 159ms/step - loss: 0.0307 - accuracy:
0.9922 - val_loss: 0.3389 - val_accuracy: 0.8594
Epoch 9/10
4/4 [==============================] - 1s 170ms/step - loss: 0.0230 - accuracy:
0.9922 - val_loss: 0.3405 - val_accuracy: 0.8555
Epoch 10/10
```

```
4/4 [==============================] - 1s 185ms/step - loss: 0.0194 - accuracy:
1.0000 - val_loss: 0.3419 - val_accuracy: 0.8555
Epoch 1/10
4/4 [==============================] - 1s 272ms/step - loss: 0.4534 - accuracy:
0.7891 - val_loss: 0.3178 - val_accuracy: 0.8594
Epoch 2/10
4/4 [==============================] - 1s 193ms/step - loss: 0.2983 - accuracy:
0.8594 - val_loss: 0.3137 - val_accuracy: 0.8633
Epoch 3/10
4/4 [==============================] - 0s 121ms/step - loss: 0.1858 - accuracy:
0.9453 - val_loss: 0.3193 - val_accuracy: 0.8516
Epoch 4/10
4/4 [==============================] - 1s 227ms/step - loss: 0.1256 - accuracy:
0.9453 - val_loss: 0.3253 - val_accuracy: 0.8477
Epoch 5/10
4/4 [==============================] - 1s 195ms/step - loss: 0.0909 - accuracy:
0.9688 - val_loss: 0.3301 - val_accuracy: 0.8438
Epoch 6/10
4/4 [==============================] - 0s 132ms/step - loss: 0.0583 - accuracy:
0.9922 - val_loss: 0.3333 - val_accuracy: 0.8477
Epoch 7/10
4/4 [==============================] - 0s 134ms/step - loss: 0.0429 - accuracy:
0.9922 - val_loss: 0.3355 - val_accuracy: 0.8477
Epoch 8/10
4/4 [==============================] - 1s 152ms/step - loss: 0.0323 - accuracy:
1.0000 - val_loss: 0.3380 - val_accuracy: 0.8477
Epoch 9/10
4/4 [==============================] - 1s 188ms/step - loss: 0.0276 - accuracy:
1.0000 - val_loss: 0.3401 - val_accuracy: 0.8477
Epoch 10/10
4/4 [==============================] - 1s 211ms/step - loss: 0.0222 - accuracy:
1.0000 - val_loss: 0.3417 - val_accuracy: 0.8477
Epoch 1/10
4/4 [==============================] - 1s 197ms/step - loss: 0.2634 - accuracy:
0.8672 - val_loss: 0.3288 - val_accuracy: 0.8516
Epoch 2/10
4/4 [==============================] - 1s 197ms/step - loss: 0.1666 - accuracy:
0.9219 - val_loss: 0.3173 - val_accuracy: 0.8555
Epoch 3/10
4/4 [==============================] - 1s 191ms/step - loss: 0.1007 - accuracy:
0.9609 - val_loss: 0.3105 - val_accuracy: 0.8594
Epoch 4/10
4/4 [==============================] - 1s 190ms/step - loss: 0.0698 - accuracy:
0.9844 - val_loss: 0.3084 - val_accuracy: 0.8633
Epoch 5/10
4/4 [==============================] - 1s 189ms/step - loss: 0.0498 - accuracy:
0.9844 - val_loss: 0.3093 - val_accuracy: 0.8672
Epoch 6/10
```

```
4/4 [==============================] - 1s 191ms/step - loss: 0.0382 - accuracy:
0.9844 - val_loss: 0.3113 - val_accuracy: 0.8633
Epoch 7/10
4/4 [==============================] - 1s 168ms/step - loss: 0.0302 - accuracy:
0.9922 - val_loss: 0.3138 - val_accuracy: 0.8633
Epoch 8/10
4/4 [==============================] - 1s 253ms/step - loss: 0.0247 - accuracy:
1.0000 - val_loss: 0.3165 - val_accuracy: 0.8555
Epoch 9/10
4/4 [==============================] - 1s 207ms/step - loss: 0.0214 - accuracy:
1.0000 - val_loss: 0.3195 - val_accuracy: 0.8516
Epoch 10/10
4/4 [==============================] - 1s 225ms/step - loss: 0.0184 - accuracy:
1.0000 - val_loss: 0.3224 - val_accuracy: 0.8516
Epoch 1/10
4/4 [==============================] - 1s 226ms/step - loss: 0.4639 - accuracy:
0.8984 - val_loss: 0.3150 - val_accuracy: 0.8633
Epoch 2/10
4/4 [==============================] - 1s 179ms/step - loss: 0.2796 - accuracy:
0.9297 - val_loss: 0.3426 - val_accuracy: 0.8750
Epoch 3/10
4/4 [==============================] - 1s 178ms/step - loss: 0.1275 - accuracy:
0.9609 - val_loss: 0.3722 - val_accuracy: 0.8633
Epoch 4/10
4/4 [==============================] - 1s 177ms/step - loss: 0.0981 - accuracy:
0.9688 - val_loss: 0.3682 - val_accuracy: 0.8633
Epoch 5/10
4/4 [==============================] - 1s 219ms/step - loss: 0.0656 - accuracy:
0.9766 - val_loss: 0.3544 - val_accuracy: 0.8711
Epoch 6/10
4/4 [==============================] - 1s 244ms/step - loss: 0.0504 - accuracy:
0.9844 - val_loss: 0.3401 - val_accuracy: 0.8789
Epoch 7/10
4/4 [==============================] - 1s 250ms/step - loss: 0.0345 - accuracy:
0.9922 - val_loss: 0.3294 - val_accuracy: 0.8750
Epoch 8/10
4/4 [==============================] - 1s 216ms/step - loss: 0.0255 - accuracy:
1.0000 - val_loss: 0.3219 - val_accuracy: 0.8789
Epoch 9/10
4/4 [==============================] - 1s 207ms/step - loss: 0.0200 - accuracy:
1.0000 - val_loss: 0.3169 - val_accuracy: 0.8828
Epoch 10/10
4/4 [==============================] - 1s 205ms/step - loss: 0.0174 - accuracy:
1.0000 - val_loss: 0.3135 - val_accuracy: 0.8867
Epoch 1/10
4/4 [==============================] - 1s 230ms/step - loss: 0.4927 - accuracy:
0.8672 - val_loss: 0.2824 - val_accuracy: 0.8789
Epoch 2/10
```

```
4/4 [==============================] - 1s 169ms/step - loss: 0.2114 - accuracy:
0.9141 - val_loss: 0.2827 - val_accuracy: 0.8750
Epoch 3/10
4/4 [==============================] - 1s 198ms/step - loss: 0.1559 - accuracy:
0.9453 - val_loss: 0.2924 - val_accuracy: 0.8711
Epoch 4/10
4/4 [==============================] - 1s 221ms/step - loss: 0.1104 - accuracy:
0.9609 - val_loss: 0.2961 - val_accuracy: 0.8750
Epoch 5/10
4/4 [==============================] - 1s 201ms/step - loss: 0.0757 - accuracy:
0.9766 - val_loss: 0.2994 - val_accuracy: 0.8750
Epoch 6/10
4/4 [==============================] - 1s 170ms/step - loss: 0.0540 - accuracy:
0.9844 - val_loss: 0.3010 - val_accuracy: 0.8711
Epoch 7/10
4/4 [==============================] - 1s 202ms/step - loss: 0.0446 - accuracy:
0.9844 - val_loss: 0.3024 - val_accuracy: 0.8711
Epoch 8/10
4/4 [==============================] - 1s 184ms/step - loss: 0.0352 - accuracy:
0.9922 - val_loss: 0.3037 - val_accuracy: 0.8789
Epoch 9/10
4/4 [==============================] - 1s 179ms/step - loss: 0.0304 - accuracy:
0.9922 - val_loss: 0.3048 - val_accuracy: 0.8789
Epoch 10/10
4/4 [==============================] - 1s 158ms/step - loss: 0.0237 - accuracy:
1.0000 - val_loss: 0.3059 - val_accuracy: 0.8789
Epoch 1/10
4/4 [==============================] - 1s 184ms/step - loss: 0.3579 - accuracy:
0.8203 - val_loss: 0.3086 - val_accuracy: 0.8750
Epoch 2/10
4/4 [==============================] - 1s 164ms/step - loss: 0.2137 - accuracy:
0.9062 - val_loss: 0.3049 - val_accuracy: 0.8750
Epoch 3/10
4/4 [==============================] - 1s 209ms/step - loss: 0.1363 - accuracy:
0.9375 - val_loss: 0.2996 - val_accuracy: 0.8750
Epoch 4/10
4/4 [==============================] - 1s 195ms/step - loss: 0.0882 - accuracy:
0.9688 - val_loss: 0.2933 - val_accuracy: 0.8750
Epoch 5/10
4/4 [==============================] - 1s 188ms/step - loss: 0.0532 - accuracy:
0.9844 - val_loss: 0.2891 - val_accuracy: 0.8789
Epoch 6/10
4/4 [==============================] - 1s 157ms/step - loss: 0.0358 - accuracy:
1.0000 - val_loss: 0.2869 - val_accuracy: 0.8789
Epoch 7/10
4/4 [==============================] - 1s 166ms/step - loss: 0.0260 - accuracy:
1.0000 - val_loss: 0.2859 - val_accuracy: 0.8789
Epoch 8/10
```

```
4/4 [==============================] - 1s 184ms/step - loss: 0.0198 - accuracy:
1.0000 - val_loss: 0.2856 - val_accuracy: 0.8750
Epoch 9/10
4/4 [==============================] - 1s 159ms/step - loss: 0.0164 - accuracy:
1.0000 - val_loss: 0.2858 - val_accuracy: 0.8750
Epoch 10/10
4/4 [==============================] - 1s 160ms/step - loss: 0.0139 - accuracy:
1.0000 - val_loss: 0.2864 - val_accuracy: 0.8750
Epoch 1/10
4/4 [==============================] - 1s 208ms/step - loss: 0.4798 - accuracy:
0.8203 - val_loss: 0.2730 - val_accuracy: 0.8711
Epoch 2/10
4/4 [==============================] - 1s 157ms/step - loss: 0.2738 - accuracy:
0.8828 - val_loss: 0.2629 - val_accuracy: 0.8828
Epoch 3/10
4/4 [==============================] - 1s 209ms/step - loss: 0.1595 - accuracy:
0.9141 - val_loss: 0.2593 - val_accuracy: 0.8945
Epoch 4/10
4/4 [==============================] - 1s 160ms/step - loss: 0.0880 - accuracy:
0.9609 - val_loss: 0.2588 - val_accuracy: 0.8984
Epoch 5/10
4/4 [==============================] - 1s 213ms/step - loss: 0.0600 - accuracy:
0.9922 - val_loss: 0.2606 - val_accuracy: 0.9062
Epoch 6/10
4/4 [==============================] - 1s 187ms/step - loss: 0.0460 - accuracy:
0.9922 - val_loss: 0.2624 - val_accuracy: 0.9062
Epoch 7/10
4/4 [==============================] - 1s 166ms/step - loss: 0.0370 - accuracy:
0.9922 - val_loss: 0.2635 - val_accuracy: 0.9062
Epoch 8/10
4/4 [==============================] - 1s 155ms/step - loss: 0.0302 - accuracy:
0.9922 - val_loss: 0.2642 - val_accuracy: 0.9062
Epoch 9/10
4/4 [==============================] - 1s 158ms/step - loss: 0.0257 - accuracy:
0.9922 - val_loss: 0.2647 - val_accuracy: 0.9062
Epoch 10/10
4/4 [==============================] - 1s 215ms/step - loss: 0.0223 - accuracy:
1.0000 - val_loss: 0.2651 - val_accuracy: 0.9062
Epoch 1/10
4/4 [==============================] - 1s 249ms/step - loss: 0.3925 - accuracy:
0.8516 - val_loss: 0.2607 - val_accuracy: 0.9062
Epoch 2/10
4/4 [==============================] - 1s 197ms/step - loss: 0.2625 - accuracy:
0.8984 - val_loss: 0.2574 - val_accuracy: 0.8984
Epoch 3/10
4/4 [==============================] - 1s 188ms/step - loss: 0.1813 - accuracy:
0.9297 - val_loss: 0.2561 - val_accuracy: 0.8906
Epoch 4/10
```

```
4/4 [==============================] - 1s 166ms/step - loss: 0.1288 - accuracy:
0.9609 - val_loss: 0.2558 - val_accuracy: 0.8906
Epoch 5/10
4/4 [==============================] - 1s 213ms/step - loss: 0.0885 - accuracy:
0.9844 - val_loss: 0.2565 - val_accuracy: 0.8828
Epoch 6/10
4/4 [==============================] - 1s 204ms/step - loss: 0.0685 - accuracy:
0.9844 - val_loss: 0.2570 - val_accuracy: 0.8828
Epoch 7/10
4/4 [==============================] - 1s 201ms/step - loss: 0.0554 - accuracy:
0.9844 - val_loss: 0.2577 - val_accuracy: 0.8828
Epoch 8/10
4/4 [==============================] - 1s 196ms/step - loss: 0.0385 - accuracy:
0.9922 - val_loss: 0.2584 - val_accuracy: 0.8828
Epoch 9/10
4/4 [==============================] - 1s 195ms/step - loss: 0.0294 - accuracy:
1.0000 - val_loss: 0.2593 - val_accuracy: 0.8828
Epoch 10/10
4/4 [==============================] - 1s 198ms/step - loss: 0.0261 - accuracy:
1.0000 - val_loss: 0.2602 - val_accuracy: 0.8867
Epoch 1/10
4/4 [==============================] - 1s 329ms/step - loss: 0.6798 - accuracy:
0.8125 - val_loss: 0.2663 - val_accuracy: 0.8789
Epoch 2/10
4/4 [==============================] - 1s 210ms/step - loss: 0.2147 - accuracy:
0.8906 - val_loss: 0.2889 - val_accuracy: 0.8594
Epoch 3/10
4/4 [==============================] - 1s 162ms/step - loss: 0.1466 - accuracy:
0.9297 - val_loss: 0.2992 - val_accuracy: 0.8477
Epoch 4/10
4/4 [==============================] - 1s 201ms/step - loss: 0.0932 - accuracy:
0.9766 - val_loss: 0.2997 - val_accuracy: 0.8438
Epoch 5/10
4/4 [==============================] - 1s 166ms/step - loss: 0.0622 - accuracy:
0.9922 - val_loss: 0.2963 - val_accuracy: 0.8438
Epoch 6/10
4/4 [==============================] - 1s 175ms/step - loss: 0.0441 - accuracy:
1.0000 - val_loss: 0.2921 - val_accuracy: 0.8477
Epoch 7/10
4/4 [==============================] - 1s 190ms/step - loss: 0.0340 - accuracy:
1.0000 - val_loss: 0.2880 - val_accuracy: 0.8516
Epoch 8/10
4/4 [==============================] - 1s 200ms/step - loss: 0.0264 - accuracy:
1.0000 - val_loss: 0.2851 - val_accuracy: 0.8594
Epoch 9/10
4/4 [==============================] - 1s 150ms/step - loss: 0.0213 - accuracy:
1.0000 - val_loss: 0.2829 - val_accuracy: 0.8594
Epoch 10/10
```

```
4/4 [==============================] - 0s 140ms/step - loss: 0.0181 - accuracy:
1.0000 - val_loss: 0.2814 - val_accuracy: 0.8633
Epoch 1/10
4/4 [==============================] - 1s 344ms/step - loss: 0.2748 - accuracy:
0.8828 - val_loss: 0.2725 - val_accuracy: 0.8711
Epoch 2/10
4/4 [==============================] - 1s 203ms/step - loss: 0.1993 - accuracy:
0.9219 - val_loss: 0.2583 - val_accuracy: 0.8867
Epoch 3/10
4/4 [==============================] - 1s 251ms/step - loss: 0.1295 - accuracy:
0.9453 - val_loss: 0.2505 - val_accuracy: 0.8906
Epoch 4/10
4/4 [==============================] - 1s 253ms/step - loss: 0.0938 - accuracy:
0.9609 - val_loss: 0.2475 - val_accuracy: 0.8945
Epoch 5/10
4/4 [==============================] - 1s 199ms/step - loss: 0.0665 - accuracy:
0.9922 - val_loss: 0.2465 - val_accuracy: 0.8945
Epoch 6/10
4/4 [==============================] - 1s 203ms/step - loss: 0.0519 - accuracy:
0.9922 - val_loss: 0.2467 - val_accuracy: 0.8945
Epoch 7/10
4/4 [==============================] - 1s 247ms/step - loss: 0.0434 - accuracy:
0.9922 - val_loss: 0.2468 - val_accuracy: 0.8945
Epoch 8/10
4/4 [==============================] - 1s 173ms/step - loss: 0.0349 - accuracy:
0.9922 - val_loss: 0.2471 - val_accuracy: 0.8945
Epoch 9/10
4/4 [==============================] - 1s 262ms/step - loss: 0.0288 - accuracy:
0.9922 - val_loss: 0.2474 - val_accuracy: 0.8906
Epoch 10/10
4/4 [==============================] - 0s 151ms/step - loss: 0.0259 - accuracy:
0.9922 - val_loss: 0.2474 - val_accuracy: 0.8906
Epoch 1/10
4/4 [==============================] - 1s 379ms/step - loss: 0.6577 - accuracy:
0.7969 - val_loss: 0.2504 - val_accuracy: 0.8945
Epoch 2/10
4/4 [==============================] - 1s 239ms/step - loss: 0.2985 - accuracy:
0.8750 - val_loss: 0.2476 - val_accuracy: 0.9219
Epoch 3/10
4/4 [==============================] - 1s 222ms/step - loss: 0.1923 - accuracy:
0.9453 - val_loss: 0.2628 - val_accuracy: 0.9062
Epoch 4/10
4/4 [==============================] - 1s 194ms/step - loss: 0.1199 - accuracy:
0.9688 - val_loss: 0.2664 - val_accuracy: 0.9023
Epoch 5/10
4/4 [==============================] - 1s 209ms/step - loss: 0.0897 - accuracy:
0.9766 - val_loss: 0.2684 - val_accuracy: 0.9023
Epoch 6/10
```

```
4/4 [==============================] - 1s 192ms/step - loss: 0.0720 - accuracy:
0.9844 - val_loss: 0.2692 - val_accuracy: 0.9023
Epoch 7/10
4/4 [==============================] - 1s 192ms/step - loss: 0.0606 - accuracy:
0.9922 - val_loss: 0.2691 - val_accuracy: 0.9023
Epoch 8/10
4/4 [==============================] - 1s 176ms/step - loss: 0.0504 - accuracy:
0.9922 - val_loss: 0.2688 - val_accuracy: 0.9062
Epoch 9/10
4/4 [==============================] - 1s 224ms/step - loss: 0.0441 - accuracy:
0.9922 - val_loss: 0.2680 - val_accuracy: 0.9102
Epoch 10/10
4/4 [==============================] - 1s 191ms/step - loss: 0.0337 - accuracy:
0.9922 - val_loss: 0.2674 - val_accuracy: 0.9141
Epoch 1/10
4/4 [==============================] - 1s 275ms/step - loss: 0.4482 - accuracy:
0.8125 - val_loss: 0.2549 - val_accuracy: 0.9141
Epoch 2/10
4/4 [==============================] - 1s 164ms/step - loss: 0.2822 - accuracy:
0.8672 - val_loss: 0.2492 - val_accuracy: 0.9180
Epoch 3/10
4/4 [==============================] - 1s 188ms/step - loss: 0.1932 - accuracy:
0.9141 - val_loss: 0.2471 - val_accuracy: 0.9180
Epoch 4/10
4/4 [==============================] - 1s 175ms/step - loss: 0.1312 - accuracy:
0.9531 - val_loss: 0.2455 - val_accuracy: 0.9102
Epoch 5/10
4/4 [==============================] - 1s 162ms/step - loss: 0.0897 - accuracy:
0.9844 - val_loss: 0.2448 - val_accuracy: 0.9023
Epoch 6/10
4/4 [==============================] - 0s 144ms/step - loss: 0.0606 - accuracy:
0.9922 - val_loss: 0.2447 - val_accuracy: 0.8945
Epoch 7/10
4/4 [==============================] - 1s 214ms/step - loss: 0.0439 - accuracy:
1.0000 - val_loss: 0.2452 - val_accuracy: 0.8906
Epoch 8/10
4/4 [==============================] - 1s 163ms/step - loss: 0.0353 - accuracy:
1.0000 - val_loss: 0.2458 - val_accuracy: 0.8906
Epoch 9/10
4/4 [==============================] - 1s 171ms/step - loss: 0.0287 - accuracy:
1.0000 - val_loss: 0.2464 - val_accuracy: 0.8906
Epoch 10/10
4/4 [==============================] - 1s 199ms/step - loss: 0.0246 - accuracy:
1.0000 - val_loss: 0.2470 - val_accuracy: 0.8906
Epoch 1/10
4/4 [==============================] - 1s 297ms/step - loss: 0.3783 - accuracy:
0.8828 - val_loss: 0.2457 - val_accuracy: 0.8906
Epoch 2/10
```

```
4/4 [==============================] - 1s 179ms/step - loss: 0.2380 - accuracy:
0.9141 - val_loss: 0.2470 - val_accuracy: 0.8867
Epoch 3/10
4/4 [==============================] - 1s 183ms/step - loss: 0.1498 - accuracy:
0.9297 - val_loss: 0.2481 - val_accuracy: 0.8867
Epoch 4/10
4/4 [==============================] - 1s 179ms/step - loss: 0.0998 - accuracy:
0.9609 - val_loss: 0.2488 - val_accuracy: 0.8867
Epoch 5/10
4/4 [==============================] - 0s 130ms/step - loss: 0.0668 - accuracy:
0.9766 - val_loss: 0.2488 - val_accuracy: 0.8867
Epoch 6/10
4/4 [==============================] - 1s 196ms/step - loss: 0.0392 - accuracy:
1.0000 - val_loss: 0.2485 - val_accuracy: 0.8906
Epoch 7/10
4/4 [==============================] - 1s 157ms/step - loss: 0.0250 - accuracy:
1.0000 - val_loss: 0.2481 - val_accuracy: 0.8906
Epoch 8/10
4/4 [==============================] - 1s 207ms/step - loss: 0.0195 - accuracy:
1.0000 - val_loss: 0.2478 - val_accuracy: 0.8906
Epoch 9/10
4/4 [==============================] - 1s 165ms/step - loss: 0.0160 - accuracy:
1.0000 - val_loss: 0.2476 - val_accuracy: 0.8906
Epoch 10/10
4/4 [==============================] - 1s 210ms/step - loss: 0.0132 - accuracy:
1.0000 - val_loss: 0.2473 - val_accuracy: 0.8906
Epoch 1/10
4/4 [==============================] - 1s 295ms/step - loss: 0.2882 - accuracy:
0.8828 - val_loss: 0.2459 - val_accuracy: 0.8906
Epoch 2/10
4/4 [==============================] - 1s 194ms/step - loss: 0.2070 - accuracy:
0.9141 - val_loss: 0.2445 - val_accuracy: 0.8945
Epoch 3/10
4/4 [==============================] - 1s 235ms/step - loss: 0.1448 - accuracy:
0.9453 - val_loss: 0.2443 - val_accuracy: 0.8945
Epoch 4/10
4/4 [==============================] - 1s 178ms/step - loss: 0.0957 - accuracy:
0.9766 - val_loss: 0.2449 - val_accuracy: 0.8945
Epoch 5/10
4/4 [==============================] - 1s 193ms/step - loss: 0.0578 - accuracy:
0.9844 - val_loss: 0.2465 - val_accuracy: 0.8945
Epoch 6/10
4/4 [==============================] - 1s 168ms/step - loss: 0.0352 - accuracy:
1.0000 - val_loss: 0.2481 - val_accuracy: 0.8945
Epoch 7/10
4/4 [==============================] - 1s 175ms/step - loss: 0.0236 - accuracy:
1.0000 - val_loss: 0.2495 - val_accuracy: 0.8945
Epoch 8/10
```

```
4/4 [==============================] - 1s 210ms/step - loss: 0.0194 - accuracy:
1.0000 - val_loss: 0.2506 - val_accuracy: 0.8945
Epoch 9/10
4/4 [==============================] - 1s 202ms/step - loss: 0.0161 - accuracy:
1.0000 - val_loss: 0.2516 - val_accuracy: 0.8945
Epoch 10/10
4/4 [==============================] - 1s 244ms/step - loss: 0.0142 - accuracy:
1.0000 - val_loss: 0.2525 - val_accuracy: 0.8945
Epoch 1/10
4/4 [==============================] - 1s 272ms/step - loss: 0.2657 - accuracy:
0.8672 - val_loss: 0.2502 - val_accuracy: 0.8984
Epoch 2/10
4/4 [==============================] - 1s 214ms/step - loss: 0.1720 - accuracy:
0.9297 - val_loss: 0.2472 - val_accuracy: 0.9023
Epoch 3/10
4/4 [==============================] - 1s 218ms/step - loss: 0.1098 - accuracy:
0.9453 - val_loss: 0.2471 - val_accuracy: 0.8984
Epoch 4/10
4/4 [==============================] - 1s 214ms/step - loss: 0.0690 - accuracy:
0.9688 - val_loss: 0.2489 - val_accuracy: 0.8984
Epoch 5/10
4/4 [==============================] - 1s 219ms/step - loss: 0.0476 - accuracy:
0.9844 - val_loss: 0.2508 - val_accuracy: 0.9023
Epoch 6/10
4/4 [==============================] - 1s 196ms/step - loss: 0.0329 - accuracy:
1.0000 - val_loss: 0.2530 - val_accuracy: 0.8984
Epoch 7/10
4/4 [==============================] - 1s 222ms/step - loss: 0.0258 - accuracy:
1.0000 - val_loss: 0.2551 - val_accuracy: 0.9023
Epoch 8/10
4/4 [==============================] - 1s 188ms/step - loss: 0.0204 - accuracy:
1.0000 - val_loss: 0.2566 - val_accuracy: 0.9062
Epoch 9/10
4/4 [==============================] - 1s 233ms/step - loss: 0.0168 - accuracy:
1.0000 - val_loss: 0.2579 - val_accuracy: 0.9062
Epoch 10/10
4/4 [==============================] - 1s 180ms/step - loss: 0.0151 - accuracy:
1.0000 - val_loss: 0.2592 - val_accuracy: 0.9062
Epoch 1/10
4/4 [==============================] - 1s 258ms/step - loss: 0.3945 - accuracy:
0.8359 - val_loss: 0.2456 - val_accuracy: 0.9102
Epoch 2/10
4/4 [==============================] - 1s 195ms/step - loss: 0.2727 - accuracy:
0.8750 - val_loss: 0.2415 - val_accuracy: 0.8984
Epoch 3/10
4/4 [==============================] - 1s 211ms/step - loss: 0.1799 - accuracy:
0.9141 - val_loss: 0.2401 - val_accuracy: 0.8984
Epoch 4/10
```

```
4/4 [==============================] - 1s 166ms/step - loss: 0.0957 - accuracy:
0.9531 - val_loss: 0.2404 - val_accuracy: 0.8984
Epoch 5/10
4/4 [==============================] - 1s 192ms/step - loss: 0.0580 - accuracy:
0.9922 - val_loss: 0.2411 - val_accuracy: 0.8984
Epoch 6/10
4/4 [==============================] - 1s 183ms/step - loss: 0.0373 - accuracy:
1.0000 - val_loss: 0.2422 - val_accuracy: 0.8984
Epoch 7/10
4/4 [==============================] - 1s 155ms/step - loss: 0.0278 - accuracy:
1.0000 - val_loss: 0.2434 - val_accuracy: 0.8945
Epoch 8/10
4/4 [==============================] - 1s 159ms/step - loss: 0.0209 - accuracy:
1.0000 - val_loss: 0.2444 - val_accuracy: 0.8945
Epoch 9/10
4/4 [==============================] - 1s 182ms/step - loss: 0.0171 - accuracy:
1.0000 - val_loss: 0.2453 - val_accuracy: 0.8945
Epoch 10/10
4/4 [==============================] - 1s 186ms/step - loss: 0.0146 - accuracy:
1.0000 - val_loss: 0.2462 - val_accuracy: 0.8906
Epoch 1/10
4/4 [==============================] - 1s 192ms/step - loss: 0.3717 - accuracy:
0.8438 - val_loss: 0.2451 - val_accuracy: 0.9023
Epoch 2/10
4/4 [==============================] - 1s 175ms/step - loss: 0.2164 - accuracy:
0.8984 - val_loss: 0.2444 - val_accuracy: 0.9023
Epoch 3/10
4/4 [==============================] - 1s 165ms/step - loss: 0.1309 - accuracy:
0.9531 - val_loss: 0.2531 - val_accuracy: 0.8984
Epoch 4/10
4/4 [==============================] - 1s 197ms/step - loss: 0.0815 - accuracy:
0.9609 - val_loss: 0.2720 - val_accuracy: 0.8711
Epoch 5/10
4/4 [==============================] - 1s 177ms/step - loss: 0.0455 - accuracy:
0.9922 - val_loss: 0.2928 - val_accuracy: 0.8672
Epoch 6/10
4/4 [==============================] - 1s 187ms/step - loss: 0.0329 - accuracy:
1.0000 - val_loss: 0.3057 - val_accuracy: 0.8555
Epoch 7/10
4/4 [==============================] - 1s 193ms/step - loss: 0.0248 - accuracy:
1.0000 - val_loss: 0.3148 - val_accuracy: 0.8594
Epoch 8/10
4/4 [==============================] - 1s 211ms/step - loss: 0.0196 - accuracy:
1.0000 - val_loss: 0.3195 - val_accuracy: 0.8672
Epoch 9/10
4/4 [==============================] - 1s 191ms/step - loss: 0.0161 - accuracy:
1.0000 - val_loss: 0.3226 - val_accuracy: 0.8672
Epoch 10/10
```

```
4/4 [==============================] - 1s 197ms/step - loss: 0.0139 - accuracy:
1.0000 - val_loss: 0.3244 - val_accuracy: 0.8672
Epoch 1/10
4/4 [==============================] - 1s 270ms/step - loss: 0.3492 - accuracy:
0.8359 - val_loss: 0.2504 - val_accuracy: 0.8828
Epoch 2/10
4/4 [==============================] - 1s 229ms/step - loss: 0.1717 - accuracy:
0.9062 - val_loss: 0.2711 - val_accuracy: 0.8672
Epoch 3/10
4/4 [==============================] - 1s 174ms/step - loss: 0.1063 - accuracy:
0.9531 - val_loss: 0.2898 - val_accuracy: 0.8594
Epoch 4/10
4/4 [==============================] - 1s 232ms/step - loss: 0.0734 - accuracy:
0.9688 - val_loss: 0.3015 - val_accuracy: 0.8438
Epoch 5/10
4/4 [==============================] - 1s 188ms/step - loss: 0.0564 - accuracy:
0.9844 - val_loss: 0.3088 - val_accuracy: 0.8477
Epoch 6/10
4/4 [==============================] - 1s 205ms/step - loss: 0.0435 - accuracy:
0.9844 - val_loss: 0.3129 - val_accuracy: 0.8516
Epoch 7/10
4/4 [==============================] - 1s 201ms/step - loss: 0.0347 - accuracy:
0.9844 - val_loss: 0.3154 - val_accuracy: 0.8477
Epoch 8/10
4/4 [==============================] - 1s 195ms/step - loss: 0.0290 - accuracy:
0.9844 - val_loss: 0.3163 - val_accuracy: 0.8516
Epoch 9/10
4/4 [==============================] - 1s 175ms/step - loss: 0.0240 - accuracy:
1.0000 - val_loss: 0.3165 - val_accuracy: 0.8555
Epoch 10/10
4/4 [==============================] - 1s 190ms/step - loss: 0.0213 - accuracy:
1.0000 - val_loss: 0.3163 - val_accuracy: 0.8516
Epoch 1/10
4/4 [==============================] - 1s 268ms/step - loss: 0.4283 - accuracy:
0.8516 - val_loss: 0.3940 - val_accuracy: 0.8242
Epoch 2/10
4/4 [==============================] - 1s 195ms/step - loss: 0.2313 - accuracy:
0.8984 - val_loss: 0.4091 - val_accuracy: 0.8203
Epoch 3/10
4/4 [==============================] - 1s 206ms/step - loss: 0.1393 - accuracy:
0.9531 - val_loss: 0.3970 - val_accuracy: 0.8203
Epoch 4/10
4/4 [==============================] - 1s 204ms/step - loss: 0.0895 - accuracy:
0.9766 - val_loss: 0.3859 - val_accuracy: 0.8242
Epoch 5/10
4/4 [==============================] - 1s 216ms/step - loss: 0.0659 - accuracy:
0.9766 - val_loss: 0.3767 - val_accuracy: 0.8242
Epoch 6/10
```

```
4/4 [==============================] - 1s 180ms/step - loss: 0.0498 - accuracy:
0.9922 - val_loss: 0.3700 - val_accuracy: 0.8242
Epoch 7/10
4/4 [==============================] - 1s 226ms/step - loss: 0.0354 - accuracy:
0.9922 - val_loss: 0.3655 - val_accuracy: 0.8242
Epoch 8/10
4/4 [==============================] - 1s 221ms/step - loss: 0.0252 - accuracy:
1.0000 - val_loss: 0.3620 - val_accuracy: 0.8242
Epoch 9/10
4/4 [==============================] - 1s 202ms/step - loss: 0.0196 - accuracy:
1.0000 - val_loss: 0.3594 - val_accuracy: 0.8242
Epoch 10/10
4/4 [==============================] - 1s 209ms/step - loss: 0.0170 - accuracy:
1.0000 - val_loss: 0.3576 - val_accuracy: 0.8242
Epoch 1/10
4/4 [==============================] - 1s 304ms/step - loss: 0.3214 - accuracy:
0.8750 - val_loss: 0.3339 - val_accuracy: 0.8438
Epoch 2/10
4/4 [==============================] - 1s 193ms/step - loss: 0.2159 - accuracy:
0.9062 - val_loss: 0.3067 - val_accuracy: 0.8477
Epoch 3/10
4/4 [==============================] - 1s 172ms/step - loss: 0.1271 - accuracy:
0.9531 - val_loss: 0.2923 - val_accuracy: 0.8672
Epoch 4/10
4/4 [==============================] - 1s 178ms/step - loss: 0.0767 - accuracy:
0.9688 - val_loss: 0.2978 - val_accuracy: 0.8867
Epoch 5/10
4/4 [==============================] - 1s 172ms/step - loss: 0.0517 - accuracy:
0.9766 - val_loss: 0.3218 - val_accuracy: 0.8789
Epoch 6/10
4/4 [==============================] - 1s 176ms/step - loss: 0.0371 - accuracy:
0.9844 - val_loss: 0.3476 - val_accuracy: 0.8750
Epoch 7/10
4/4 [==============================] - 1s 205ms/step - loss: 0.0275 - accuracy:
0.9922 - val_loss: 0.3655 - val_accuracy: 0.8750
Epoch 8/10
4/4 [==============================] - 1s 192ms/step - loss: 0.0239 - accuracy:
0.9922 - val_loss: 0.3796 - val_accuracy: 0.8750
Epoch 9/10
4/4 [==============================] - 1s 185ms/step - loss: 0.0212 - accuracy:
0.9922 - val_loss: 0.3896 - val_accuracy: 0.8633
Epoch 10/10
4/4 [==============================] - 1s 208ms/step - loss: 0.0190 - accuracy:
1.0000 - val_loss: 0.3972 - val_accuracy: 0.8633
Epoch 1/10
4/4 [==============================] - 1s 257ms/step - loss: 0.3541 - accuracy:
0.8594 - val_loss: 0.3114 - val_accuracy: 0.8828
Epoch 2/10
```

```
4/4 [==============================] - 1s 206ms/step - loss: 0.2270 - accuracy:
0.9062 - val_loss: 0.2915 - val_accuracy: 0.8945
Epoch 3/10
4/4 [==============================] - 1s 171ms/step - loss: 0.1609 - accuracy:
0.9219 - val_loss: 0.2807 - val_accuracy: 0.8906
Epoch 4/10
4/4 [==============================] - 1s 217ms/step - loss: 0.1043 - accuracy:
0.9453 - val_loss: 0.2756 - val_accuracy: 0.8906
Epoch 5/10
4/4 [==============================] - 1s 223ms/step - loss: 0.0722 - accuracy:
0.9609 - val_loss: 0.2732 - val_accuracy: 0.8906
Epoch 6/10
4/4 [==============================] - 1s 243ms/step - loss: 0.0518 - accuracy:
0.9922 - val_loss: 0.2726 - val_accuracy: 0.8984
Epoch 7/10
4/4 [==============================] - 1s 233ms/step - loss: 0.0384 - accuracy:
1.0000 - val_loss: 0.2729 - val_accuracy: 0.9023
Epoch 8/10
4/4 [==============================] - 1s 227ms/step - loss: 0.0320 - accuracy:
1.0000 - val_loss: 0.2734 - val_accuracy: 0.9062
Epoch 9/10
4/4 [==============================] - 1s 208ms/step - loss: 0.0254 - accuracy:
1.0000 - val_loss: 0.2739 - val_accuracy: 0.9062
Epoch 10/10
4/4 [==============================] - 1s 162ms/step - loss: 0.0215 - accuracy:
1.0000 - val_loss: 0.2746 - val_accuracy: 0.9062
Epoch 1/10
4/4 [==============================] - 1s 250ms/step - loss: 0.3178 - accuracy:
0.8984 - val_loss: 0.2803 - val_accuracy: 0.8984
Epoch 2/10
4/4 [==============================] - 1s 197ms/step - loss: 0.2266 - accuracy:
0.9297 - val_loss: 0.2769 - val_accuracy: 0.9023
Epoch 3/10
4/4 [==============================] - 1s 224ms/step - loss: 0.1636 - accuracy:
0.9531 - val_loss: 0.2714 - val_accuracy: 0.9062
Epoch 4/10
4/4 [==============================] - 1s 214ms/step - loss: 0.1253 - accuracy:
0.9688 - val_loss: 0.2668 - val_accuracy: 0.9062
Epoch 5/10
4/4 [==============================] - 1s 180ms/step - loss: 0.0922 - accuracy:
0.9688 - val_loss: 0.2638 - val_accuracy: 0.9062
Epoch 6/10
4/4 [==============================] - 1s 172ms/step - loss: 0.0634 - accuracy:
0.9766 - val_loss: 0.2619 - val_accuracy: 0.9023
Epoch 7/10
4/4 [==============================] - 1s 181ms/step - loss: 0.0415 - accuracy:
0.9922 - val_loss: 0.2606 - val_accuracy: 0.9023
Epoch 8/10
```

```
4/4 [==============================] - 1s 174ms/step - loss: 0.0343 - accuracy:
0.9922 - val_loss: 0.2599 - val_accuracy: 0.9023
Epoch 9/10
4/4 [==============================] - 1s 153ms/step - loss: 0.0243 - accuracy:
0.9922 - val_loss: 0.2594 - val_accuracy: 0.9062
Epoch 10/10
4/4 [==============================] - 1s 185ms/step - loss: 0.0193 - accuracy:
0.9922 - val_loss: 0.2592 - val_accuracy: 0.9062
Epoch 1/10
4/4 [==============================] - 1s 207ms/step - loss: 0.2209 - accuracy:
0.9141 - val_loss: 0.2558 - val_accuracy: 0.9102
Epoch 2/10
4/4 [==============================] - 1s 241ms/step - loss: 0.1228 - accuracy:
0.9531 - val_loss: 0.2516 - val_accuracy: 0.9062
Epoch 3/10
4/4 [==============================] - 1s 208ms/step - loss: 0.0769 - accuracy:
0.9844 - val_loss: 0.2482 - val_accuracy: 0.9102
Epoch 4/10
4/4 [==============================] - 1s 214ms/step - loss: 0.0514 - accuracy:
0.9844 - val_loss: 0.2451 - val_accuracy: 0.9102
Epoch 5/10
4/4 [==============================] - 1s 223ms/step - loss: 0.0315 - accuracy:
0.9922 - val_loss: 0.2437 - val_accuracy: 0.9141
Epoch 6/10
4/4 [==============================] - 1s 202ms/step - loss: 0.0272 - accuracy:
0.9922 - val_loss: 0.2434 - val_accuracy: 0.9141
Epoch 7/10
4/4 [==============================] - 1s 164ms/step - loss: 0.0193 - accuracy:
1.0000 - val_loss: 0.2434 - val_accuracy: 0.9141
Epoch 8/10
4/4 [==============================] - 1s 177ms/step - loss: 0.0164 - accuracy:
1.0000 - val_loss: 0.2438 - val_accuracy: 0.9141
Epoch 9/10
4/4 [==============================] - 1s 182ms/step - loss: 0.0139 - accuracy:
1.0000 - val_loss: 0.2443 - val_accuracy: 0.9141
Epoch 10/10
4/4 [==============================] - 1s 168ms/step - loss: 0.0124 - accuracy:
1.0000 - val_loss: 0.2448 - val_accuracy: 0.9062
Epoch 1/10
4/4 [==============================] - 1s 261ms/step - loss: 0.3409 - accuracy:
0.9219 - val_loss: 0.2377 - val_accuracy: 0.9180
Epoch 2/10
4/4 [==============================] - 1s 165ms/step - loss: 0.1557 - accuracy:
0.9688 - val_loss: 0.2380 - val_accuracy: 0.9102
Epoch 3/10
4/4 [==============================] - 1s 154ms/step - loss: 0.1103 - accuracy:
0.9766 - val_loss: 0.2402 - val_accuracy: 0.9102
Epoch 4/10
```

```
4/4 [==============================] - 0s 139ms/step - loss: 0.0782 - accuracy:
0.9766 - val_loss: 0.2406 - val_accuracy: 0.9102
Epoch 5/10
4/4 [==============================] - 1s 171ms/step - loss: 0.0566 - accuracy:
0.9766 - val_loss: 0.2401 - val_accuracy: 0.9102
Epoch 6/10
4/4 [==============================] - 1s 198ms/step - loss: 0.0343 - accuracy:
0.9844 - val_loss: 0.2394 - val_accuracy: 0.9102
Epoch 7/10
4/4 [==============================] - 1s 206ms/step - loss: 0.0228 - accuracy:
1.0000 - val_loss: 0.2387 - val_accuracy: 0.9102
Epoch 8/10
4/4 [==============================] - 1s 240ms/step - loss: 0.0177 - accuracy:
1.0000 - val_loss: 0.2381 - val_accuracy: 0.9102
Epoch 9/10
4/4 [==============================] - 1s 172ms/step - loss: 0.0144 - accuracy:
1.0000 - val_loss: 0.2377 - val_accuracy: 0.9141
Epoch 10/10
4/4 [==============================] - 1s 202ms/step - loss: 0.0122 - accuracy:
1.0000 - val_loss: 0.2374 - val_accuracy: 0.9141
Epoch 1/10
4/4 [==============================] - 1s 343ms/step - loss: 0.2827 - accuracy:
0.9141 - val_loss: 0.2371 - val_accuracy: 0.9219
Epoch 2/10
4/4 [==============================] - 1s 231ms/step - loss: 0.1850 - accuracy:
0.9375 - val_loss: 0.2714 - val_accuracy: 0.8867
Epoch 3/10
4/4 [==============================] - 1s 241ms/step - loss: 0.1262 - accuracy:
0.9609 - val_loss: 0.3267 - val_accuracy: 0.8672
Epoch 4/10
4/4 [==============================] - 1s 265ms/step - loss: 0.0892 - accuracy:
0.9688 - val_loss: 0.3357 - val_accuracy: 0.8672
Epoch 5/10
4/4 [==============================] - 1s 219ms/step - loss: 0.0635 - accuracy:
0.9766 - val_loss: 0.3244 - val_accuracy: 0.8633
Epoch 6/10
4/4 [==============================] - 1s 239ms/step - loss: 0.0466 - accuracy:
0.9844 - val_loss: 0.3083 - val_accuracy: 0.8672
Epoch 7/10
4/4 [==============================] - 1s 202ms/step - loss: 0.0333 - accuracy:
0.9922 - val_loss: 0.2987 - val_accuracy: 0.8789
Epoch 8/10
4/4 [==============================] - 1s 188ms/step - loss: 0.0254 - accuracy:
0.9922 - val_loss: 0.2922 - val_accuracy: 0.8828
Epoch 9/10
4/4 [==============================] - 1s 152ms/step - loss: 0.0199 - accuracy:
1.0000 - val_loss: 0.2891 - val_accuracy: 0.8828
Epoch 10/10
```

```
4/4 [==============================] - 0s 131ms/step - loss: 0.0166 - accuracy:
1.0000 - val_loss: 0.2877 - val_accuracy: 0.8828
```

3.25.2 Exportando o modelo Rede Neural

[340]:
```
model_.save_weights('pesos/keras_finished_count_val_2.h5')
```

3.25.3 Carregar modelo Rede Neural

[342]:
```
keras_model_val = create_nn_model()
keras_model_val.load_weights('pesos/keras_finished_count_val_2.h5')
```

3.26 Avaliando as métricas do modelo Rede Neural count com validação em treino

- Avaliando a acurácia do modelo

[343]:
```
score, acc = keras_model_val.evaluate(x_test_batch, y_test_batch)
print("A acurácia para o modelo com Redes neurais com o dataset count com␣
 ↪validação foi de {:.2f}%".format(acc*100))
```

```
4/4 [==============================] - 0s 19ms/step - loss: 0.3021 - accuracy:
0.8604
A acurácia para o modelo com Redes neurais com o dataset count com validação foi
de 82.03%
```

- obtendo as predições do modelo utilizando o dataset de treino com **count**

[344]:
```
y_pred_keras_val = keras_model_val.predict(x_test.to_numpy()).round()
y_pred_keras_val = [classify(result) for result in y_pred_keras_val]
```

3.27 Classification Report

- A função `classification_report` nos dá todas as métricas importantes que podemos obter de um modelo.
- Ela irá nos dar a precisão por classe.
- O quanto de revoação (recall) o modelo tem por classe e no geral
- O f1-score geral e por classe
- A acurária do modelo
- Todas estas métricas são úteis e nos ajudarão a escolher qual o modelo se saiu melhor e pode ser utilizado para esta tarefa.

[346]:
```
report = classification_report(y_test, y_pred_keras_val,␣
↪target_names=target_names, output_dict=True)
```

[347]:
```
sgd_stats = {'model': 'keras_count_val',
             'accuracy': report['accuracy'],
             'recall': report['macro avg']['recall'],
             'f1_score': report['macro avg']['f1-score'],
             'support': report['macro avg']['support'],
             'falso_precision': report['Falso']['precision'],
             'falso_recall': report['Falso']['recall'],
             'falso_f1_score': report['Falso']['f1-score'],
             'verdadeiro_precision': report['Verdadeiro']['precision'],
             'verdadeiro_recall': report['Verdadeiro']['recall'],
             'verdadeiro_f1_score': report['Verdadeiro']['f1-score']}
```

[348]:
```
stats.append(sgd_stats)
```

3.28 Rede Neural com dataset contains e com validação em treino

3.28.1 Observação o modelo já foi treinado, caso queira usar o modelo já treinado ele pode ser carregado diretamente aqui

- A célula cria o modelo de rede neural descrito

[349]:
```
model_sn = create_nn_model()
```

- A célula abaixo treina a rede neural

[350]:
```
inicio_lote = 0
fim_lote = tamanho_lote

total_lotes = str(int(len(X_train)/tamanho_lote))
ultimo_lote = False
i = 1

while True:
    x_train = formatar_lote_de_noticias_sim_nao(X_train[inicio_lote:fim_lote])
    y_tr = np.asanyarray(format_labels(y_train[inicio_lote:fim_lote].
↪to_numpy()))
    model_sn.fit(x=x_train.to_numpy(), y=y_tr,␣
↪validation_data=(val_formated_sn, val_formated_class_sn), epochs=10)
    inicio_lote += tamanho_lote

    if fim_lote + tamanho_lote > len(X_train):
        fim_lote = len(X_train)
        ultimo_lote = True
    else:
```

```
        fim_lote += tamanho_lote

    i += 1
    if ultimo_lote:
        break
```

```
Epoch 1/10
4/4 [==============================] - 12s 3s/step - loss: 0.6876 - accuracy:
0.7146 - val_loss: 0.6652 - val_accuracy: 0.7695
Epoch 2/10
4/4 [==============================] - 1s 156ms/step - loss: 0.6319 - accuracy:
0.9198 - val_loss: 0.6253 - val_accuracy: 0.8086
Epoch 3/10
4/4 [==============================] - 1s 179ms/step - loss: 0.5398 - accuracy:
0.9385 - val_loss: 0.5805 - val_accuracy: 0.8516
Epoch 4/10
4/4 [==============================] - 1s 193ms/step - loss: 0.4492 - accuracy:
0.9104 - val_loss: 0.5428 - val_accuracy: 0.8594
Epoch 5/10
4/4 [==============================] - 1s 171ms/step - loss: 0.3669 - accuracy:
0.9219 - val_loss: 0.5118 - val_accuracy: 0.8555
Epoch 6/10
4/4 [==============================] - 1s 182ms/step - loss: 0.3142 - accuracy:
0.8906 - val_loss: 0.4816 - val_accuracy: 0.8477
Epoch 7/10
4/4 [==============================] - 1s 215ms/step - loss: 0.2441 - accuracy:
0.9313 - val_loss: 0.4551 - val_accuracy: 0.8516
Epoch 8/10
4/4 [==============================] - 1s 198ms/step - loss: 0.2214 - accuracy:
0.9219 - val_loss: 0.4339 - val_accuracy: 0.8516
Epoch 9/10
4/4 [==============================] - 1s 155ms/step - loss: 0.1802 - accuracy:
0.9521 - val_loss: 0.4154 - val_accuracy: 0.8516
Epoch 10/10
4/4 [==============================] - 1s 166ms/step - loss: 0.1926 - accuracy:
0.9396 - val_loss: 0.4020 - val_accuracy: 0.8477
Epoch 1/10
4/4 [==============================] - 1s 228ms/step - loss: 0.4166 - accuracy:
0.8203 - val_loss: 0.3735 - val_accuracy: 0.8594
Epoch 2/10
4/4 [==============================] - 1s 180ms/step - loss: 0.3385 - accuracy:
0.8516 - val_loss: 0.3524 - val_accuracy: 0.8594
Epoch 3/10
4/4 [==============================] - 1s 197ms/step - loss: 0.2858 - accuracy:
0.8516 - val_loss: 0.3395 - val_accuracy: 0.8633
Epoch 4/10
4/4 [==============================] - 1s 185ms/step - loss: 0.2554 - accuracy:
```

```
0.8516 - val_loss: 0.3349 - val_accuracy: 0.8516
Epoch 5/10
4/4 [==============================] - 1s 171ms/step - loss: 0.2305 - accuracy:
0.8516 - val_loss: 0.3338 - val_accuracy: 0.8516
Epoch 6/10
4/4 [==============================] - 1s 177ms/step - loss: 0.2088 - accuracy:
0.8672 - val_loss: 0.3334 - val_accuracy: 0.8438
Epoch 7/10
4/4 [==============================] - 1s 207ms/step - loss: 0.1904 - accuracy:
0.8750 - val_loss: 0.3304 - val_accuracy: 0.8438
Epoch 8/10
4/4 [==============================] - 1s 169ms/step - loss: 0.1729 - accuracy:
0.8906 - val_loss: 0.3282 - val_accuracy: 0.8516
Epoch 9/10
4/4 [==============================] - 1s 160ms/step - loss: 0.1567 - accuracy:
0.9219 - val_loss: 0.3259 - val_accuracy: 0.8594
Epoch 10/10
4/4 [==============================] - 1s 190ms/step - loss: 0.1452 - accuracy:
0.9375 - val_loss: 0.3233 - val_accuracy: 0.8594
Epoch 1/10
4/4 [==============================] - 1s 246ms/step - loss: 0.4096 - accuracy:
0.8047 - val_loss: 0.3049 - val_accuracy: 0.8633
Epoch 2/10
4/4 [==============================] - 0s 156ms/step - loss: 0.3018 - accuracy:
0.8750 - val_loss: 0.3030 - val_accuracy: 0.8672
Epoch 3/10
4/4 [==============================] - 1s 161ms/step - loss: 0.2534 - accuracy:
0.8750 - val_loss: 0.2995 - val_accuracy: 0.8711
Epoch 4/10
4/4 [==============================] - 0s 152ms/step - loss: 0.2200 - accuracy:
0.8906 - val_loss: 0.2965 - val_accuracy: 0.8750
Epoch 5/10
4/4 [==============================] - 1s 185ms/step - loss: 0.1908 - accuracy:
0.8906 - val_loss: 0.2918 - val_accuracy: 0.8750
Epoch 6/10
4/4 [==============================] - 1s 168ms/step - loss: 0.1677 - accuracy:
0.9219 - val_loss: 0.2900 - val_accuracy: 0.8867
Epoch 7/10
4/4 [==============================] - 1s 187ms/step - loss: 0.1467 - accuracy:
0.9453 - val_loss: 0.2874 - val_accuracy: 0.8828
Epoch 8/10
4/4 [==============================] - 1s 187ms/step - loss: 0.1304 - accuracy:
0.9688 - val_loss: 0.2864 - val_accuracy: 0.8828
Epoch 9/10
4/4 [==============================] - 1s 201ms/step - loss: 0.1183 - accuracy:
0.9922 - val_loss: 0.2852 - val_accuracy: 0.8828
Epoch 10/10
4/4 [==============================] - 1s 212ms/step - loss: 0.1066 - accuracy:
```

```
0.9922 - val_loss: 0.2839 - val_accuracy: 0.8789
Epoch 1/10
4/4 [==============================] - 1s 316ms/step - loss: 0.2925 - accuracy:
0.8906 - val_loss: 0.2789 - val_accuracy: 0.8711
Epoch 2/10
4/4 [==============================] - 1s 223ms/step - loss: 0.2069 - accuracy:
0.9062 - val_loss: 0.2791 - val_accuracy: 0.8711
Epoch 3/10
4/4 [==============================] - 1s 212ms/step - loss: 0.1511 - accuracy:
0.9141 - val_loss: 0.2775 - val_accuracy: 0.8711
Epoch 4/10
4/4 [==============================] - 1s 204ms/step - loss: 0.1159 - accuracy:
0.9375 - val_loss: 0.2780 - val_accuracy: 0.8711
Epoch 5/10
4/4 [==============================] - 1s 185ms/step - loss: 0.0948 - accuracy:
0.9609 - val_loss: 0.2812 - val_accuracy: 0.8711
Epoch 6/10
4/4 [==============================] - 1s 218ms/step - loss: 0.0797 - accuracy:
0.9844 - val_loss: 0.2873 - val_accuracy: 0.8594
Epoch 7/10
4/4 [==============================] - 1s 213ms/step - loss: 0.0688 - accuracy:
0.9922 - val_loss: 0.2945 - val_accuracy: 0.8594
Epoch 8/10
4/4 [==============================] - 1s 198ms/step - loss: 0.0605 - accuracy:
0.9922 - val_loss: 0.3037 - val_accuracy: 0.8477
Epoch 9/10
4/4 [==============================] - 1s 185ms/step - loss: 0.0546 - accuracy:
1.0000 - val_loss: 0.3139 - val_accuracy: 0.8398
Epoch 10/10
4/4 [==============================] - 1s 191ms/step - loss: 0.0500 - accuracy:
1.0000 - val_loss: 0.3239 - val_accuracy: 0.8359
Epoch 1/10
4/4 [==============================] - 1s 260ms/step - loss: 0.3702 - accuracy:
0.8594 - val_loss: 0.2667 - val_accuracy: 0.8750
Epoch 2/10
4/4 [==============================] - 1s 247ms/step - loss: 0.2236 - accuracy:
0.9297 - val_loss: 0.2927 - val_accuracy: 0.8359
Epoch 3/10
4/4 [==============================] - 1s 247ms/step - loss: 0.1628 - accuracy:
0.9453 - val_loss: 0.2946 - val_accuracy: 0.8398
Epoch 4/10
4/4 [==============================] - 1s 193ms/step - loss: 0.1295 - accuracy:
0.9531 - val_loss: 0.2889 - val_accuracy: 0.8398
Epoch 5/10
4/4 [==============================] - 1s 184ms/step - loss: 0.1056 - accuracy:
0.9609 - val_loss: 0.2826 - val_accuracy: 0.8555
Epoch 6/10
4/4 [==============================] - 1s 236ms/step - loss: 0.0901 - accuracy:
```

```
0.9609 - val_loss: 0.2802 - val_accuracy: 0.8555
Epoch 7/10
4/4 [==============================] - 1s 217ms/step - loss: 0.0773 - accuracy:
0.9844 - val_loss: 0.2779 - val_accuracy: 0.8555
Epoch 8/10
4/4 [==============================] - 1s 238ms/step - loss: 0.0667 - accuracy:
0.9922 - val_loss: 0.2742 - val_accuracy: 0.8555
Epoch 9/10
4/4 [==============================] - 1s 216ms/step - loss: 0.0599 - accuracy:
0.9922 - val_loss: 0.2723 - val_accuracy: 0.8633
Epoch 10/10
4/4 [==============================] - 1s 179ms/step - loss: 0.0534 - accuracy:
1.0000 - val_loss: 0.2712 - val_accuracy: 0.8633
Epoch 1/10
4/4 [==============================] - 1s 315ms/step - loss: 0.2661 - accuracy:
0.8828 - val_loss: 0.2741 - val_accuracy: 0.8594
Epoch 2/10
4/4 [==============================] - 1s 213ms/step - loss: 0.1799 - accuracy:
0.9219 - val_loss: 0.3208 - val_accuracy: 0.8398
Epoch 3/10
4/4 [==============================] - 1s 183ms/step - loss: 0.1473 - accuracy:
0.9453 - val_loss: 0.3695 - val_accuracy: 0.8008
Epoch 4/10
4/4 [==============================] - 1s 202ms/step - loss: 0.1127 - accuracy:
0.9688 - val_loss: 0.3958 - val_accuracy: 0.7930
Epoch 5/10
4/4 [==============================] - 1s 228ms/step - loss: 0.0924 - accuracy:
0.9766 - val_loss: 0.4154 - val_accuracy: 0.7930
Epoch 6/10
4/4 [==============================] - 1s 222ms/step - loss: 0.0744 - accuracy:
1.0000 - val_loss: 0.4240 - val_accuracy: 0.7891
Epoch 7/10
4/4 [==============================] - 1s 253ms/step - loss: 0.0641 - accuracy:
1.0000 - val_loss: 0.4262 - val_accuracy: 0.7891
Epoch 8/10
4/4 [==============================] - 1s 220ms/step - loss: 0.0561 - accuracy:
1.0000 - val_loss: 0.4250 - val_accuracy: 0.7891
Epoch 9/10
4/4 [==============================] - 1s 223ms/step - loss: 0.0508 - accuracy:
1.0000 - val_loss: 0.4207 - val_accuracy: 0.7930
Epoch 10/10
4/4 [==============================] - 1s 166ms/step - loss: 0.0459 - accuracy:
1.0000 - val_loss: 0.4178 - val_accuracy: 0.7969
Epoch 1/10
4/4 [==============================] - 1s 251ms/step - loss: 0.4287 - accuracy:
0.7891 - val_loss: 0.2832 - val_accuracy: 0.8633
Epoch 2/10
4/4 [==============================] - 0s 150ms/step - loss: 0.2364 - accuracy:
```

```
0.8906 - val_loss: 0.4319 - val_accuracy: 0.8398
Epoch 3/10
4/4 [==============================] - 1s 173ms/step - loss: 0.1949 - accuracy:
0.9062 - val_loss: 0.6607 - val_accuracy: 0.8047
Epoch 4/10
4/4 [==============================] - 1s 158ms/step - loss: 0.1561 - accuracy:
0.9141 - val_loss: 0.7509 - val_accuracy: 0.8008
Epoch 5/10
4/4 [==============================] - 1s 236ms/step - loss: 0.1174 - accuracy:
0.9297 - val_loss: 0.7640 - val_accuracy: 0.7969
Epoch 6/10
4/4 [==============================] - 1s 188ms/step - loss: 0.0876 - accuracy:
0.9688 - val_loss: 0.7427 - val_accuracy: 0.8008
Epoch 7/10
4/4 [==============================] - 1s 197ms/step - loss: 0.0668 - accuracy:
0.9922 - val_loss: 0.7267 - val_accuracy: 0.8008
Epoch 8/10
4/4 [==============================] - 1s 198ms/step - loss: 0.0569 - accuracy:
1.0000 - val_loss: 0.7221 - val_accuracy: 0.8008
Epoch 9/10
4/4 [==============================] - 1s 206ms/step - loss: 0.0508 - accuracy:
1.0000 - val_loss: 0.7255 - val_accuracy: 0.8008
Epoch 10/10
4/4 [==============================] - 1s 201ms/step - loss: 0.0457 - accuracy:
1.0000 - val_loss: 0.7387 - val_accuracy: 0.7969
Epoch 1/10
4/4 [==============================] - 1s 240ms/step - loss: 0.4948 - accuracy:
0.8672 - val_loss: 0.3464 - val_accuracy: 0.8477
Epoch 2/10
4/4 [==============================] - 1s 204ms/step - loss: 0.2406 - accuracy:
0.8984 - val_loss: 0.3645 - val_accuracy: 0.8281
Epoch 3/10
4/4 [==============================] - 0s 142ms/step - loss: 0.1884 - accuracy:
0.9219 - val_loss: 0.4202 - val_accuracy: 0.8086
Epoch 4/10
4/4 [==============================] - 1s 212ms/step - loss: 0.1443 - accuracy:
0.9531 - val_loss: 0.4394 - val_accuracy: 0.8047
Epoch 5/10
4/4 [==============================] - 0s 135ms/step - loss: 0.1117 - accuracy:
0.9688 - val_loss: 0.4428 - val_accuracy: 0.8008
Epoch 6/10
4/4 [==============================] - 1s 178ms/step - loss: 0.0882 - accuracy:
0.9922 - val_loss: 0.4410 - val_accuracy: 0.8008
Epoch 7/10
4/4 [==============================] - 1s 179ms/step - loss: 0.0738 - accuracy:
0.9922 - val_loss: 0.4375 - val_accuracy: 0.8008
Epoch 8/10
4/4 [==============================] - 1s 194ms/step - loss: 0.0625 - accuracy:
```

```
0.9922 - val_loss: 0.4372 - val_accuracy: 0.8125
Epoch 9/10
4/4 [==============================] - 0s 149ms/step - loss: 0.0551 - accuracy:
0.9922 - val_loss: 0.4365 - val_accuracy: 0.8125
Epoch 10/10
4/4 [==============================] - 1s 176ms/step - loss: 0.0495 - accuracy:
0.9922 - val_loss: 0.4350 - val_accuracy: 0.8125
Epoch 1/10
4/4 [==============================] - 1s 246ms/step - loss: 0.7154 - accuracy:
0.7500 - val_loss: 0.3712 - val_accuracy: 0.8242
Epoch 2/10
4/4 [==============================] - 0s 120ms/step - loss: 0.3418 - accuracy:
0.8281 - val_loss: 0.3379 - val_accuracy: 0.8516
Epoch 3/10
4/4 [==============================] - 1s 170ms/step - loss: 0.2181 - accuracy:
0.8984 - val_loss: 0.3911 - val_accuracy: 0.8398
Epoch 4/10
4/4 [==============================] - 0s 145ms/step - loss: 0.1530 - accuracy:
0.9297 - val_loss: 0.4281 - val_accuracy: 0.8398
Epoch 5/10
4/4 [==============================] - 1s 184ms/step - loss: 0.1180 - accuracy:
0.9609 - val_loss: 0.4370 - val_accuracy: 0.8203
Epoch 6/10
4/4 [==============================] - 1s 159ms/step - loss: 0.0872 - accuracy:
0.9844 - val_loss: 0.4402 - val_accuracy: 0.8242
Epoch 7/10
4/4 [==============================] - 1s 175ms/step - loss: 0.0749 - accuracy:
0.9922 - val_loss: 0.4437 - val_accuracy: 0.8281
Epoch 8/10
4/4 [==============================] - 1s 176ms/step - loss: 0.0654 - accuracy:
0.9922 - val_loss: 0.4500 - val_accuracy: 0.8203
Epoch 9/10
4/4 [==============================] - 1s 217ms/step - loss: 0.0590 - accuracy:
0.9922 - val_loss: 0.4577 - val_accuracy: 0.8203
Epoch 10/10
4/4 [==============================] - 1s 189ms/step - loss: 0.0538 - accuracy:
0.9922 - val_loss: 0.4669 - val_accuracy: 0.8164
Epoch 1/10
4/4 [==============================] - 1s 250ms/step - loss: 0.4655 - accuracy:
0.8203 - val_loss: 0.3925 - val_accuracy: 0.8281
Epoch 2/10
4/4 [==============================] - 1s 217ms/step - loss: 0.2340 - accuracy:
0.8906 - val_loss: 0.4000 - val_accuracy: 0.8281
Epoch 3/10
4/4 [==============================] - 1s 260ms/step - loss: 0.1555 - accuracy:
0.9219 - val_loss: 0.4162 - val_accuracy: 0.8203
Epoch 4/10
4/4 [==============================] - 1s 220ms/step - loss: 0.1194 - accuracy:
```

```
0.9453 - val_loss: 0.4284 - val_accuracy: 0.8203
Epoch 5/10
4/4 [==============================] - 1s 152ms/step - loss: 0.0960 - accuracy:
0.9766 - val_loss: 0.4333 - val_accuracy: 0.8047
Epoch 6/10
4/4 [==============================] - 1s 186ms/step - loss: 0.0806 - accuracy:
0.9844 - val_loss: 0.4370 - val_accuracy: 0.8008
Epoch 7/10
4/4 [==============================] - 1s 227ms/step - loss: 0.0710 - accuracy:
0.9844 - val_loss: 0.4401 - val_accuracy: 0.7969
Epoch 8/10
4/4 [==============================] - 0s 127ms/step - loss: 0.0648 - accuracy:
0.9844 - val_loss: 0.4432 - val_accuracy: 0.7969
Epoch 9/10
4/4 [==============================] - 0s 136ms/step - loss: 0.0594 - accuracy:
0.9844 - val_loss: 0.4460 - val_accuracy: 0.8008
Epoch 10/10
4/4 [==============================] - 1s 172ms/step - loss: 0.0556 - accuracy:
0.9844 - val_loss: 0.4491 - val_accuracy: 0.8086
Epoch 1/10
4/4 [==============================] - 1s 287ms/step - loss: 0.4668 - accuracy:
0.7891 - val_loss: 0.4023 - val_accuracy: 0.8125
Epoch 2/10
4/4 [==============================] - 0s 148ms/step - loss: 0.2911 - accuracy:
0.8750 - val_loss: 0.3827 - val_accuracy: 0.8203
Epoch 3/10
4/4 [==============================] - 1s 236ms/step - loss: 0.1950 - accuracy:
0.9219 - val_loss: 0.3782 - val_accuracy: 0.8242
Epoch 4/10
4/4 [==============================] - 1s 209ms/step - loss: 0.1349 - accuracy:
0.9531 - val_loss: 0.3750 - val_accuracy: 0.8203
Epoch 5/10
4/4 [==============================] - 1s 206ms/step - loss: 0.0995 - accuracy:
0.9688 - val_loss: 0.3681 - val_accuracy: 0.8281
Epoch 6/10
4/4 [==============================] - 0s 146ms/step - loss: 0.0802 - accuracy:
0.9844 - val_loss: 0.3598 - val_accuracy: 0.8359
Epoch 7/10
4/4 [==============================] - 0s 157ms/step - loss: 0.0663 - accuracy:
0.9844 - val_loss: 0.3557 - val_accuracy: 0.8398
Epoch 8/10
4/4 [==============================] - 1s 212ms/step - loss: 0.0584 - accuracy:
0.9844 - val_loss: 0.3525 - val_accuracy: 0.8398
Epoch 9/10
4/4 [==============================] - 1s 179ms/step - loss: 0.0533 - accuracy:
0.9844 - val_loss: 0.3496 - val_accuracy: 0.8398
Epoch 10/10
4/4 [==============================] - 1s 183ms/step - loss: 0.0479 - accuracy:
```

```
0.9844 - val_loss: 0.3485 - val_accuracy: 0.8398
Epoch 1/10
4/4 [==============================] - 1s 302ms/step - loss: 0.3797 - accuracy:
0.8203 - val_loss: 0.3352 - val_accuracy: 0.8438
Epoch 2/10
4/4 [==============================] - 1s 153ms/step - loss: 0.1897 - accuracy:
0.9219 - val_loss: 0.3192 - val_accuracy: 0.8594
Epoch 3/10
4/4 [==============================] - 1s 171ms/step - loss: 0.1393 - accuracy:
0.9531 - val_loss: 0.3133 - val_accuracy: 0.8672
Epoch 4/10
4/4 [==============================] - 1s 169ms/step - loss: 0.1081 - accuracy:
0.9609 - val_loss: 0.3116 - val_accuracy: 0.8594
Epoch 5/10
4/4 [==============================] - 1s 179ms/step - loss: 0.0860 - accuracy:
0.9609 - val_loss: 0.3110 - val_accuracy: 0.8594
Epoch 6/10
4/4 [==============================] - 1s 165ms/step - loss: 0.0704 - accuracy:
0.9688 - val_loss: 0.3110 - val_accuracy: 0.8594
Epoch 7/10
4/4 [==============================] - 1s 187ms/step - loss: 0.0579 - accuracy:
0.9844 - val_loss: 0.3115 - val_accuracy: 0.8594
Epoch 8/10
4/4 [==============================] - 1s 255ms/step - loss: 0.0482 - accuracy:
0.9922 - val_loss: 0.3131 - val_accuracy: 0.8594
Epoch 9/10
4/4 [==============================] - 1s 187ms/step - loss: 0.0410 - accuracy:
1.0000 - val_loss: 0.3150 - val_accuracy: 0.8594
Epoch 10/10
4/4 [==============================] - 1s 219ms/step - loss: 0.0362 - accuracy:
1.0000 - val_loss: 0.3172 - val_accuracy: 0.8594
Epoch 1/10
4/4 [==============================] - 1s 281ms/step - loss: 0.5454 - accuracy:
0.7969 - val_loss: 0.3187 - val_accuracy: 0.8594
Epoch 2/10
4/4 [==============================] - 1s 176ms/step - loss: 0.2797 - accuracy:
0.8594 - val_loss: 0.3148 - val_accuracy: 0.8594
Epoch 3/10
4/4 [==============================] - 1s 194ms/step - loss: 0.2094 - accuracy:
0.9062 - val_loss: 0.3139 - val_accuracy: 0.8633
Epoch 4/10
4/4 [==============================] - 1s 267ms/step - loss: 0.1553 - accuracy:
0.9375 - val_loss: 0.3162 - val_accuracy: 0.8633
Epoch 5/10
4/4 [==============================] - 1s 188ms/step - loss: 0.1221 - accuracy:
0.9609 - val_loss: 0.3185 - val_accuracy: 0.8594
Epoch 6/10
4/4 [==============================] - 1s 156ms/step - loss: 0.0961 - accuracy:
```

```
0.9766 - val_loss: 0.3187 - val_accuracy: 0.8594
Epoch 7/10
4/4 [==============================] - 1s 189ms/step - loss: 0.0814 - accuracy:
0.9844 - val_loss: 0.3182 - val_accuracy: 0.8594
Epoch 8/10
4/4 [==============================] - 1s 207ms/step - loss: 0.0713 - accuracy:
0.9844 - val_loss: 0.3184 - val_accuracy: 0.8516
Epoch 9/10
4/4 [==============================] - 1s 234ms/step - loss: 0.0622 - accuracy:
0.9844 - val_loss: 0.3189 - val_accuracy: 0.8477
Epoch 10/10
4/4 [==============================] - 1s 255ms/step - loss: 0.0555 - accuracy:
0.9922 - val_loss: 0.3195 - val_accuracy: 0.8477
Epoch 1/10
4/4 [==============================] - 1s 239ms/step - loss: 0.4352 - accuracy:
0.8281 - val_loss: 0.3145 - val_accuracy: 0.8516
Epoch 2/10
4/4 [==============================] - 1s 170ms/step - loss: 0.1961 - accuracy:
0.9219 - val_loss: 0.3401 - val_accuracy: 0.8438
Epoch 3/10
4/4 [==============================] - 1s 177ms/step - loss: 0.1122 - accuracy:
0.9609 - val_loss: 0.3761 - val_accuracy: 0.8281
Epoch 4/10
4/4 [==============================] - 1s 193ms/step - loss: 0.0815 - accuracy:
0.9844 - val_loss: 0.4013 - val_accuracy: 0.8086
Epoch 5/10
4/4 [==============================] - 1s 187ms/step - loss: 0.0622 - accuracy:
0.9844 - val_loss: 0.4159 - val_accuracy: 0.8008
Epoch 6/10
4/4 [==============================] - 1s 184ms/step - loss: 0.0524 - accuracy:
0.9844 - val_loss: 0.4188 - val_accuracy: 0.8008
Epoch 7/10
4/4 [==============================] - 1s 163ms/step - loss: 0.0450 - accuracy:
0.9922 - val_loss: 0.4174 - val_accuracy: 0.8086
Epoch 8/10
4/4 [==============================] - 1s 208ms/step - loss: 0.0396 - accuracy:
0.9922 - val_loss: 0.4117 - val_accuracy: 0.8203
Epoch 9/10
4/4 [==============================] - 1s 197ms/step - loss: 0.0361 - accuracy:
0.9922 - val_loss: 0.4030 - val_accuracy: 0.8398
Epoch 10/10
4/4 [==============================] - 1s 174ms/step - loss: 0.0316 - accuracy:
0.9922 - val_loss: 0.3964 - val_accuracy: 0.8477
Epoch 1/10
4/4 [==============================] - 1s 271ms/step - loss: 0.4554 - accuracy:
0.8281 - val_loss: 0.3320 - val_accuracy: 0.8633
Epoch 2/10
4/4 [==============================] - 1s 235ms/step - loss: 0.2795 - accuracy:
```

```
0.8828 - val_loss: 0.3044 - val_accuracy: 0.8750
Epoch 3/10
4/4 [==============================] - 1s 207ms/step - loss: 0.1960 - accuracy:
0.8984 - val_loss: 0.3515 - val_accuracy: 0.8555
Epoch 4/10
4/4 [==============================] - 1s 189ms/step - loss: 0.1342 - accuracy:
0.9375 - val_loss: 0.4203 - val_accuracy: 0.8633
Epoch 5/10
4/4 [==============================] - 1s 223ms/step - loss: 0.1059 - accuracy:
0.9609 - val_loss: 0.4507 - val_accuracy: 0.8672
Epoch 6/10
4/4 [==============================] - 1s 188ms/step - loss: 0.0827 - accuracy:
0.9688 - val_loss: 0.4644 - val_accuracy: 0.8633
Epoch 7/10
4/4 [==============================] - 1s 159ms/step - loss: 0.0676 - accuracy:
0.9766 - val_loss: 0.4687 - val_accuracy: 0.8594
Epoch 8/10
4/4 [==============================] - 1s 215ms/step - loss: 0.0563 - accuracy:
0.9766 - val_loss: 0.4670 - val_accuracy: 0.8594
Epoch 9/10
4/4 [==============================] - 1s 224ms/step - loss: 0.0486 - accuracy:
0.9844 - val_loss: 0.4636 - val_accuracy: 0.8594
Epoch 10/10
4/4 [==============================] - 1s 194ms/step - loss: 0.0423 - accuracy:
0.9922 - val_loss: 0.4627 - val_accuracy: 0.8555
Epoch 1/10
4/4 [==============================] - 1s 235ms/step - loss: 0.3313 - accuracy:
0.8984 - val_loss: 0.4083 - val_accuracy: 0.8633
Epoch 2/10
4/4 [==============================] - 1s 184ms/step - loss: 0.1384 - accuracy:
0.9375 - val_loss: 0.3417 - val_accuracy: 0.8594
Epoch 3/10
4/4 [==============================] - 1s 180ms/step - loss: 0.0958 - accuracy:
0.9531 - val_loss: 0.3184 - val_accuracy: 0.8750
Epoch 4/10
4/4 [==============================] - 1s 173ms/step - loss: 0.0727 - accuracy:
0.9766 - val_loss: 0.3171 - val_accuracy: 0.8711
Epoch 5/10
4/4 [==============================] - 0s 151ms/step - loss: 0.0528 - accuracy:
0.9922 - val_loss: 0.3195 - val_accuracy: 0.8711
Epoch 6/10
4/4 [==============================] - 0s 156ms/step - loss: 0.0424 - accuracy:
0.9922 - val_loss: 0.3223 - val_accuracy: 0.8750
Epoch 7/10
4/4 [==============================] - 1s 159ms/step - loss: 0.0326 - accuracy:
1.0000 - val_loss: 0.3253 - val_accuracy: 0.8750
Epoch 8/10
4/4 [==============================] - 1s 177ms/step - loss: 0.0276 - accuracy:
```

```
1.0000 - val_loss: 0.3287 - val_accuracy: 0.8750
Epoch 9/10
4/4 [==============================] - 1s 241ms/step - loss: 0.0233 - accuracy:
1.0000 - val_loss: 0.3320 - val_accuracy: 0.8750
Epoch 10/10
4/4 [==============================] - 1s 191ms/step - loss: 0.0210 - accuracy:
1.0000 - val_loss: 0.3355 - val_accuracy: 0.8750
Epoch 1/10
4/4 [==============================] - 1s 272ms/step - loss: 0.3506 - accuracy:
0.8906 - val_loss: 0.3331 - val_accuracy: 0.8789
Epoch 2/10
4/4 [==============================] - 1s 198ms/step - loss: 0.1385 - accuracy:
0.9375 - val_loss: 0.3373 - val_accuracy: 0.8789
Epoch 3/10
4/4 [==============================] - 1s 231ms/step - loss: 0.1037 - accuracy:
0.9609 - val_loss: 0.3346 - val_accuracy: 0.8789
Epoch 4/10
4/4 [==============================] - 1s 196ms/step - loss: 0.0608 - accuracy:
0.9766 - val_loss: 0.3323 - val_accuracy: 0.8750
Epoch 5/10
4/4 [==============================] - 1s 242ms/step - loss: 0.0412 - accuracy:
1.0000 - val_loss: 0.3318 - val_accuracy: 0.8711
Epoch 6/10
4/4 [==============================] - 1s 185ms/step - loss: 0.0319 - accuracy:
1.0000 - val_loss: 0.3322 - val_accuracy: 0.8711
Epoch 7/10
4/4 [==============================] - 1s 244ms/step - loss: 0.0271 - accuracy:
1.0000 - val_loss: 0.3330 - val_accuracy: 0.8711
Epoch 8/10
4/4 [==============================] - 1s 199ms/step - loss: 0.0236 - accuracy:
1.0000 - val_loss: 0.3340 - val_accuracy: 0.8672
Epoch 9/10
4/4 [==============================] - 1s 280ms/step - loss: 0.0208 - accuracy:
1.0000 - val_loss: 0.3350 - val_accuracy: 0.8672
Epoch 10/10
4/4 [==============================] - 0s 139ms/step - loss: 0.0190 - accuracy:
1.0000 - val_loss: 0.3361 - val_accuracy: 0.8672
Epoch 1/10
4/4 [==============================] - 1s 267ms/step - loss: 0.5262 - accuracy:
0.8359 - val_loss: 0.3316 - val_accuracy: 0.8672
Epoch 2/10
4/4 [==============================] - 0s 145ms/step - loss: 0.2660 - accuracy:
0.8828 - val_loss: 0.3280 - val_accuracy: 0.8711
Epoch 3/10
4/4 [==============================] - 1s 174ms/step - loss: 0.1397 - accuracy:
0.9453 - val_loss: 0.3289 - val_accuracy: 0.8672
Epoch 4/10
4/4 [==============================] - 1s 176ms/step - loss: 0.0900 - accuracy:
```

```
0.9844 - val_loss: 0.3327 - val_accuracy: 0.8672
Epoch 5/10
4/4 [==============================] - 1s 196ms/step - loss: 0.0588 - accuracy:
0.9922 - val_loss: 0.3373 - val_accuracy: 0.8633
Epoch 6/10
4/4 [==============================] - 1s 174ms/step - loss: 0.0468 - accuracy:
0.9922 - val_loss: 0.3428 - val_accuracy: 0.8633
Epoch 7/10
4/4 [==============================] - 1s 203ms/step - loss: 0.0373 - accuracy:
1.0000 - val_loss: 0.3476 - val_accuracy: 0.8594
Epoch 8/10
4/4 [==============================] - 1s 210ms/step - loss: 0.0327 - accuracy:
1.0000 - val_loss: 0.3519 - val_accuracy: 0.8594
Epoch 9/10
4/4 [==============================] - 1s 232ms/step - loss: 0.0290 - accuracy:
1.0000 - val_loss: 0.3555 - val_accuracy: 0.8555
Epoch 10/10
4/4 [==============================] - 1s 169ms/step - loss: 0.0257 - accuracy:
1.0000 - val_loss: 0.3587 - val_accuracy: 0.8555
Epoch 1/10
4/4 [==============================] - 1s 258ms/step - loss: 0.3807 - accuracy:
0.8516 - val_loss: 0.3541 - val_accuracy: 0.8477
Epoch 2/10
4/4 [==============================] - 1s 195ms/step - loss: 0.2258 - accuracy:
0.9219 - val_loss: 0.3879 - val_accuracy: 0.8555
Epoch 3/10
4/4 [==============================] - 1s 177ms/step - loss: 0.1171 - accuracy:
0.9531 - val_loss: 0.4837 - val_accuracy: 0.8555
Epoch 4/10
4/4 [==============================] - 1s 208ms/step - loss: 0.0778 - accuracy:
0.9766 - val_loss: 0.5736 - val_accuracy: 0.8438
Epoch 5/10
4/4 [==============================] - 1s 193ms/step - loss: 0.0503 - accuracy:
0.9766 - val_loss: 0.6410 - val_accuracy: 0.8438
Epoch 6/10
4/4 [==============================] - 1s 171ms/step - loss: 0.0354 - accuracy:
1.0000 - val_loss: 0.6830 - val_accuracy: 0.8398
Epoch 7/10
4/4 [==============================] - 1s 184ms/step - loss: 0.0275 - accuracy:
1.0000 - val_loss: 0.7026 - val_accuracy: 0.8281
Epoch 8/10
4/4 [==============================] - 1s 195ms/step - loss: 0.0208 - accuracy:
1.0000 - val_loss: 0.7169 - val_accuracy: 0.8281
Epoch 9/10
4/4 [==============================] - 1s 190ms/step - loss: 0.0189 - accuracy:
1.0000 - val_loss: 0.7245 - val_accuracy: 0.8281
Epoch 10/10
4/4 [==============================] - 1s 189ms/step - loss: 0.0170 - accuracy:
```

```
1.0000 - val_loss: 0.7315 - val_accuracy: 0.8281
Epoch 1/10
4/4 [==============================] - 1s 286ms/step - loss: 0.7519 - accuracy:
0.8438 - val_loss: 0.3725 - val_accuracy: 0.8594
Epoch 2/10
4/4 [==============================] - 1s 187ms/step - loss: 0.1837 - accuracy:
0.9453 - val_loss: 0.4775 - val_accuracy: 0.8164
Epoch 3/10
4/4 [==============================] - 1s 183ms/step - loss: 0.1396 - accuracy:
0.9453 - val_loss: 0.6038 - val_accuracy: 0.7773
Epoch 4/10
4/4 [==============================] - 1s 225ms/step - loss: 0.0950 - accuracy:
0.9688 - val_loss: 0.6473 - val_accuracy: 0.7773
Epoch 5/10
4/4 [==============================] - 1s 223ms/step - loss: 0.0682 - accuracy:
0.9766 - val_loss: 0.6572 - val_accuracy: 0.7656
Epoch 6/10
4/4 [==============================] - 1s 181ms/step - loss: 0.0537 - accuracy:
0.9844 - val_loss: 0.6569 - val_accuracy: 0.7656
Epoch 7/10
4/4 [==============================] - 1s 243ms/step - loss: 0.0438 - accuracy:
0.9844 - val_loss: 0.6510 - val_accuracy: 0.7656
Epoch 8/10
4/4 [==============================] - 1s 243ms/step - loss: 0.0375 - accuracy:
0.9844 - val_loss: 0.6413 - val_accuracy: 0.7734
Epoch 9/10
4/4 [==============================] - 1s 267ms/step - loss: 0.0288 - accuracy:
1.0000 - val_loss: 0.6315 - val_accuracy: 0.7734
Epoch 10/10
4/4 [==============================] - 1s 196ms/step - loss: 0.0239 - accuracy:
1.0000 - val_loss: 0.6224 - val_accuracy: 0.7891
Epoch 1/10
4/4 [==============================] - 1s 322ms/step - loss: 0.6339 - accuracy:
0.8047 - val_loss: 0.4476 - val_accuracy: 0.8320
Epoch 2/10
4/4 [==============================] - 1s 237ms/step - loss: 0.3078 - accuracy:
0.8828 - val_loss: 0.3562 - val_accuracy: 0.8633
Epoch 3/10
4/4 [==============================] - 1s 198ms/step - loss: 0.2287 - accuracy:
0.9219 - val_loss: 0.3820 - val_accuracy: 0.8750
Epoch 4/10
4/4 [==============================] - 1s 229ms/step - loss: 0.1806 - accuracy:
0.9219 - val_loss: 0.4178 - val_accuracy: 0.8633
Epoch 5/10
4/4 [==============================] - 1s 233ms/step - loss: 0.1414 - accuracy:
0.9453 - val_loss: 0.4295 - val_accuracy: 0.8555
Epoch 6/10
4/4 [==============================] - 1s 192ms/step - loss: 0.1018 - accuracy:
```

```
0.9609 - val_loss: 0.4298 - val_accuracy: 0.8555
Epoch 7/10
4/4 [==============================] - 1s 221ms/step - loss: 0.0766 - accuracy:
0.9688 - val_loss: 0.4277 - val_accuracy: 0.8594
Epoch 8/10
4/4 [==============================] - 1s 192ms/step - loss: 0.0592 - accuracy:
0.9766 - val_loss: 0.4238 - val_accuracy: 0.8633
Epoch 9/10
4/4 [==============================] - 1s 200ms/step - loss: 0.0487 - accuracy:
0.9844 - val_loss: 0.4207 - val_accuracy: 0.8633
Epoch 10/10
4/4 [==============================] - 1s 164ms/step - loss: 0.0424 - accuracy:
0.9922 - val_loss: 0.4187 - val_accuracy: 0.8633
Epoch 1/10
4/4 [==============================] - 1s 357ms/step - loss: 0.3178 - accuracy:
0.8828 - val_loss: 0.3383 - val_accuracy: 0.8594
Epoch 2/10
4/4 [==============================] - 1s 183ms/step - loss: 0.1726 - accuracy:
0.9297 - val_loss: 0.3426 - val_accuracy: 0.8516
Epoch 3/10
4/4 [==============================] - 1s 161ms/step - loss: 0.1242 - accuracy:
0.9609 - val_loss: 0.3610 - val_accuracy: 0.8477
Epoch 4/10
4/4 [==============================] - 1s 169ms/step - loss: 0.0829 - accuracy:
0.9766 - val_loss: 0.3739 - val_accuracy: 0.8477
Epoch 5/10
4/4 [==============================] - 1s 192ms/step - loss: 0.0596 - accuracy:
0.9844 - val_loss: 0.3832 - val_accuracy: 0.8398
Epoch 6/10
4/4 [==============================] - 1s 208ms/step - loss: 0.0418 - accuracy:
0.9922 - val_loss: 0.3906 - val_accuracy: 0.8359
Epoch 7/10
4/4 [==============================] - 1s 157ms/step - loss: 0.0305 - accuracy:
0.9922 - val_loss: 0.3975 - val_accuracy: 0.8242
Epoch 8/10
4/4 [==============================] - 1s 155ms/step - loss: 0.0224 - accuracy:
1.0000 - val_loss: 0.4022 - val_accuracy: 0.8242
Epoch 9/10
4/4 [==============================] - 1s 164ms/step - loss: 0.0172 - accuracy:
1.0000 - val_loss: 0.4045 - val_accuracy: 0.8242
Epoch 10/10
4/4 [==============================] - 1s 222ms/step - loss: 0.0150 - accuracy:
1.0000 - val_loss: 0.4053 - val_accuracy: 0.8242
Epoch 1/10
4/4 [==============================] - 1s 311ms/step - loss: 0.5130 - accuracy:
0.8125 - val_loss: 0.3769 - val_accuracy: 0.8438
Epoch 2/10
4/4 [==============================] - 1s 220ms/step - loss: 0.2965 - accuracy:
```

```
0.8516 - val_loss: 0.3467 - val_accuracy: 0.8594
Epoch 3/10
4/4 [==============================] - 1s 202ms/step - loss: 0.2101 - accuracy:
0.9062 - val_loss: 0.3319 - val_accuracy: 0.8672
Epoch 4/10
4/4 [==============================] - 1s 219ms/step - loss: 0.1690 - accuracy:
0.9219 - val_loss: 0.3283 - val_accuracy: 0.8672
Epoch 5/10
4/4 [==============================] - 1s 198ms/step - loss: 0.1318 - accuracy:
0.9297 - val_loss: 0.3277 - val_accuracy: 0.8750
Epoch 6/10
4/4 [==============================] - 1s 234ms/step - loss: 0.1043 - accuracy:
0.9609 - val_loss: 0.3289 - val_accuracy: 0.8750
Epoch 7/10
4/4 [==============================] - 1s 237ms/step - loss: 0.0831 - accuracy:
0.9688 - val_loss: 0.3309 - val_accuracy: 0.8750
Epoch 8/10
4/4 [==============================] - 1s 238ms/step - loss: 0.0681 - accuracy:
0.9766 - val_loss: 0.3333 - val_accuracy: 0.8711
Epoch 9/10
4/4 [==============================] - 1s 201ms/step - loss: 0.0586 - accuracy:
0.9844 - val_loss: 0.3355 - val_accuracy: 0.8711
Epoch 10/10
4/4 [==============================] - 1s 176ms/step - loss: 0.0488 - accuracy:
0.9922 - val_loss: 0.3375 - val_accuracy: 0.8633
Epoch 1/10
4/4 [==============================] - 1s 326ms/step - loss: 0.4639 - accuracy:
0.7969 - val_loss: 0.3426 - val_accuracy: 0.8672
Epoch 2/10
4/4 [==============================] - 1s 224ms/step - loss: 0.2489 - accuracy:
0.8750 - val_loss: 0.3416 - val_accuracy: 0.8750
Epoch 3/10
4/4 [==============================] - 1s 189ms/step - loss: 0.1814 - accuracy:
0.9062 - val_loss: 0.3372 - val_accuracy: 0.8750
Epoch 4/10
4/4 [==============================] - 1s 204ms/step - loss: 0.1319 - accuracy:
0.9453 - val_loss: 0.3364 - val_accuracy: 0.8672
Epoch 5/10
4/4 [==============================] - 1s 199ms/step - loss: 0.1048 - accuracy:
0.9609 - val_loss: 0.3373 - val_accuracy: 0.8633
Epoch 6/10
4/4 [==============================] - 1s 218ms/step - loss: 0.0863 - accuracy:
0.9766 - val_loss: 0.3395 - val_accuracy: 0.8633
Epoch 7/10
4/4 [==============================] - 1s 188ms/step - loss: 0.0732 - accuracy:
0.9844 - val_loss: 0.3422 - val_accuracy: 0.8672
Epoch 8/10
4/4 [==============================] - 1s 193ms/step - loss: 0.0607 - accuracy:
```

```
0.9844 - val_loss: 0.3452 - val_accuracy: 0.8672
Epoch 9/10
4/4 [==============================] - 1s 171ms/step - loss: 0.0526 - accuracy:
0.9844 - val_loss: 0.3486 - val_accuracy: 0.8672
Epoch 10/10
4/4 [==============================] - 1s 199ms/step - loss: 0.0433 - accuracy:
1.0000 - val_loss: 0.3520 - val_accuracy: 0.8672
Epoch 1/10
4/4 [==============================] - 1s 340ms/step - loss: 0.3525 - accuracy:
0.8125 - val_loss: 0.3612 - val_accuracy: 0.8672
Epoch 2/10
4/4 [==============================] - 1s 199ms/step - loss: 0.2192 - accuracy:
0.8828 - val_loss: 0.3679 - val_accuracy: 0.8711
Epoch 3/10
4/4 [==============================] - 1s 178ms/step - loss: 0.1534 - accuracy:
0.9219 - val_loss: 0.3696 - val_accuracy: 0.8594
Epoch 4/10
4/4 [==============================] - 1s 192ms/step - loss: 0.1163 - accuracy:
0.9453 - val_loss: 0.3683 - val_accuracy: 0.8555
Epoch 5/10
4/4 [==============================] - 1s 166ms/step - loss: 0.0930 - accuracy:
0.9609 - val_loss: 0.3675 - val_accuracy: 0.8555
Epoch 6/10
4/4 [==============================] - 1s 165ms/step - loss: 0.0738 - accuracy:
0.9766 - val_loss: 0.3671 - val_accuracy: 0.8555
Epoch 7/10
4/4 [==============================] - 1s 189ms/step - loss: 0.0582 - accuracy:
0.9844 - val_loss: 0.3681 - val_accuracy: 0.8555
Epoch 8/10
4/4 [==============================] - 1s 206ms/step - loss: 0.0502 - accuracy:
0.9922 - val_loss: 0.3696 - val_accuracy: 0.8516
Epoch 9/10
4/4 [==============================] - 0s 151ms/step - loss: 0.0433 - accuracy:
0.9922 - val_loss: 0.3715 - val_accuracy: 0.8516
Epoch 10/10
4/4 [==============================] - 1s 209ms/step - loss: 0.0382 - accuracy:
1.0000 - val_loss: 0.3737 - val_accuracy: 0.8477
Epoch 1/10
4/4 [==============================] - 1s 252ms/step - loss: 0.3065 - accuracy:
0.8828 - val_loss: 0.3814 - val_accuracy: 0.8477
Epoch 2/10
4/4 [==============================] - 1s 231ms/step - loss: 0.1792 - accuracy:
0.9219 - val_loss: 0.3918 - val_accuracy: 0.8477
Epoch 3/10
4/4 [==============================] - 1s 248ms/step - loss: 0.1319 - accuracy:
0.9531 - val_loss: 0.4046 - val_accuracy: 0.8555
Epoch 4/10
4/4 [==============================] - 1s 199ms/step - loss: 0.0967 - accuracy:
```

```
0.9609 - val_loss: 0.4147 - val_accuracy: 0.8594
Epoch 5/10
4/4 [==============================] - 1s 256ms/step - loss: 0.0779 - accuracy:
0.9688 - val_loss: 0.4189 - val_accuracy: 0.8594
Epoch 6/10
4/4 [==============================] - 1s 245ms/step - loss: 0.0609 - accuracy:
0.9844 - val_loss: 0.4214 - val_accuracy: 0.8555
Epoch 7/10
4/4 [==============================] - 1s 198ms/step - loss: 0.0494 - accuracy:
0.9844 - val_loss: 0.4233 - val_accuracy: 0.8594
Epoch 8/10
4/4 [==============================] - 1s 209ms/step - loss: 0.0421 - accuracy:
0.9844 - val_loss: 0.4254 - val_accuracy: 0.8555
Epoch 9/10
4/4 [==============================] - 1s 190ms/step - loss: 0.0358 - accuracy:
0.9844 - val_loss: 0.4277 - val_accuracy: 0.8516
Epoch 10/10
4/4 [==============================] - 1s 150ms/step - loss: 0.0320 - accuracy:
0.9844 - val_loss: 0.4295 - val_accuracy: 0.8555
Epoch 1/10
4/4 [==============================] - 1s 225ms/step - loss: 0.5123 - accuracy:
0.7969 - val_loss: 0.3897 - val_accuracy: 0.8516
Epoch 2/10
4/4 [==============================] - 1s 175ms/step - loss: 0.3149 - accuracy:
0.8438 - val_loss: 0.3646 - val_accuracy: 0.8516
Epoch 3/10
4/4 [==============================] - 0s 152ms/step - loss: 0.2122 - accuracy:
0.8828 - val_loss: 0.3591 - val_accuracy: 0.8398
Epoch 4/10
4/4 [==============================] - 1s 214ms/step - loss: 0.1482 - accuracy:
0.9297 - val_loss: 0.3572 - val_accuracy: 0.8438
Epoch 5/10
4/4 [==============================] - 1s 218ms/step - loss: 0.1208 - accuracy:
0.9531 - val_loss: 0.3565 - val_accuracy: 0.8438
Epoch 6/10
4/4 [==============================] - 1s 157ms/step - loss: 0.0991 - accuracy:
0.9609 - val_loss: 0.3559 - val_accuracy: 0.8438
Epoch 7/10
4/4 [==============================] - 1s 217ms/step - loss: 0.0821 - accuracy:
0.9766 - val_loss: 0.3559 - val_accuracy: 0.8477
Epoch 8/10
4/4 [==============================] - 1s 192ms/step - loss: 0.0721 - accuracy:
0.9766 - val_loss: 0.3559 - val_accuracy: 0.8555
Epoch 9/10
4/4 [==============================] - 1s 208ms/step - loss: 0.0648 - accuracy:
0.9766 - val_loss: 0.3564 - val_accuracy: 0.8555
Epoch 10/10
4/4 [==============================] - 1s 191ms/step - loss: 0.0598 - accuracy:
```

```
0.9766 - val_loss: 0.3567 - val_accuracy: 0.8555
Epoch 1/10
4/4 [==============================] - 1s 270ms/step - loss: 0.5201 - accuracy:
0.8281 - val_loss: 0.3620 - val_accuracy: 0.8633
Epoch 2/10
4/4 [==============================] - 1s 226ms/step - loss: 0.2009 - accuracy:
0.9062 - val_loss: 0.4015 - val_accuracy: 0.8594
Epoch 3/10
4/4 [==============================] - 1s 179ms/step - loss: 0.1283 - accuracy:
0.9375 - val_loss: 0.4750 - val_accuracy: 0.8477
Epoch 4/10
4/4 [==============================] - 1s 186ms/step - loss: 0.0918 - accuracy:
0.9609 - val_loss: 0.5376 - val_accuracy: 0.8281
Epoch 5/10
4/4 [==============================] - 1s 208ms/step - loss: 0.0687 - accuracy:
0.9844 - val_loss: 0.5834 - val_accuracy: 0.8242
Epoch 6/10
4/4 [==============================] - 1s 159ms/step - loss: 0.0571 - accuracy:
0.9844 - val_loss: 0.6147 - val_accuracy: 0.8164
Epoch 7/10
4/4 [==============================] - 1s 163ms/step - loss: 0.0474 - accuracy:
0.9844 - val_loss: 0.6383 - val_accuracy: 0.8086
Epoch 8/10
4/4 [==============================] - 1s 171ms/step - loss: 0.0400 - accuracy:
0.9844 - val_loss: 0.6554 - val_accuracy: 0.8086
Epoch 9/10
4/4 [==============================] - 0s 141ms/step - loss: 0.0361 - accuracy:
0.9844 - val_loss: 0.6702 - val_accuracy: 0.8125
Epoch 10/10
4/4 [==============================] - 1s 193ms/step - loss: 0.0308 - accuracy:
0.9922 - val_loss: 0.6808 - val_accuracy: 0.8125
Epoch 1/10
4/4 [==============================] - 1s 239ms/step - loss: 0.5422 - accuracy:
0.8594 - val_loss: 0.3616 - val_accuracy: 0.8594
Epoch 2/10
4/4 [==============================] - 1s 158ms/step - loss: 0.1945 - accuracy:
0.9062 - val_loss: 0.3528 - val_accuracy: 0.8477
Epoch 3/10
4/4 [==============================] - 1s 209ms/step - loss: 0.1479 - accuracy:
0.9375 - val_loss: 0.3882 - val_accuracy: 0.8398
Epoch 4/10
4/4 [==============================] - 1s 180ms/step - loss: 0.1126 - accuracy:
0.9531 - val_loss: 0.4137 - val_accuracy: 0.8320
Epoch 5/10
4/4 [==============================] - 1s 185ms/step - loss: 0.0906 - accuracy:
0.9688 - val_loss: 0.4287 - val_accuracy: 0.8203
Epoch 6/10
4/4 [==============================] - 1s 166ms/step - loss: 0.0724 - accuracy:
```

```
0.9844 - val_loss: 0.4352 - val_accuracy: 0.8203
Epoch 7/10
4/4 [==============================] - 1s 230ms/step - loss: 0.0609 - accuracy:
0.9844 - val_loss: 0.4376 - val_accuracy: 0.8203
Epoch 8/10
4/4 [==============================] - 1s 167ms/step - loss: 0.0528 - accuracy:
0.9844 - val_loss: 0.4371 - val_accuracy: 0.8203
Epoch 9/10
4/4 [==============================] - 1s 199ms/step - loss: 0.0460 - accuracy:
0.9844 - val_loss: 0.4354 - val_accuracy: 0.8203
Epoch 10/10
4/4 [==============================] - 1s 193ms/step - loss: 0.0407 - accuracy:
0.9844 - val_loss: 0.4342 - val_accuracy: 0.8242
Epoch 1/10
4/4 [==============================] - 1s 242ms/step - loss: 0.4111 - accuracy:
0.8281 - val_loss: 0.3854 - val_accuracy: 0.8398
Epoch 2/10
4/4 [==============================] - 1s 168ms/step - loss: 0.2527 - accuracy:
0.8828 - val_loss: 0.3471 - val_accuracy: 0.8555
Epoch 3/10
4/4 [==============================] - 1s 174ms/step - loss: 0.1701 - accuracy:
0.9141 - val_loss: 0.3290 - val_accuracy: 0.8633
Epoch 4/10
4/4 [==============================] - 1s 157ms/step - loss: 0.1222 - accuracy:
0.9297 - val_loss: 0.3204 - val_accuracy: 0.8789
Epoch 5/10
4/4 [==============================] - 1s 180ms/step - loss: 0.0884 - accuracy:
0.9688 - val_loss: 0.3168 - val_accuracy: 0.8789
Epoch 6/10
4/4 [==============================] - 1s 182ms/step - loss: 0.0714 - accuracy:
0.9766 - val_loss: 0.3160 - val_accuracy: 0.8789
Epoch 7/10
4/4 [==============================] - 1s 153ms/step - loss: 0.0585 - accuracy:
0.9922 - val_loss: 0.3165 - val_accuracy: 0.8828
Epoch 8/10
4/4 [==============================] - 1s 189ms/step - loss: 0.0470 - accuracy:
1.0000 - val_loss: 0.3178 - val_accuracy: 0.8828
Epoch 9/10
4/4 [==============================] - 1s 157ms/step - loss: 0.0413 - accuracy:
1.0000 - val_loss: 0.3195 - val_accuracy: 0.8828
Epoch 10/10
4/4 [==============================] - 1s 170ms/step - loss: 0.0360 - accuracy:
1.0000 - val_loss: 0.3213 - val_accuracy: 0.8828
Epoch 1/10
4/4 [==============================] - 1s 215ms/step - loss: 0.3053 - accuracy:
0.8594 - val_loss: 0.3242 - val_accuracy: 0.8867
Epoch 2/10
4/4 [==============================] - 1s 173ms/step - loss: 0.1932 - accuracy:
```

```
0.9141 - val_loss: 0.3273 - val_accuracy: 0.8867
Epoch 3/10
4/4 [==============================] - 1s 194ms/step - loss: 0.1440 - accuracy:
0.9297 - val_loss: 0.3300 - val_accuracy: 0.8828
Epoch 4/10
4/4 [==============================] - 1s 152ms/step - loss: 0.1118 - accuracy:
0.9531 - val_loss: 0.3324 - val_accuracy: 0.8828
Epoch 5/10
4/4 [==============================] - 1s 213ms/step - loss: 0.0870 - accuracy:
0.9688 - val_loss: 0.3344 - val_accuracy: 0.8867
Epoch 6/10
4/4 [==============================] - 1s 187ms/step - loss: 0.0675 - accuracy:
0.9766 - val_loss: 0.3363 - val_accuracy: 0.8867
Epoch 7/10
4/4 [==============================] - 1s 188ms/step - loss: 0.0537 - accuracy:
0.9844 - val_loss: 0.3386 - val_accuracy: 0.8867
Epoch 8/10
4/4 [==============================] - 1s 196ms/step - loss: 0.0481 - accuracy:
0.9844 - val_loss: 0.3404 - val_accuracy: 0.8867
Epoch 9/10
4/4 [==============================] - 0s 138ms/step - loss: 0.0428 - accuracy:
0.9922 - val_loss: 0.3425 - val_accuracy: 0.8828
Epoch 10/10
4/4 [==============================] - 1s 156ms/step - loss: 0.0391 - accuracy:
0.9922 - val_loss: 0.3447 - val_accuracy: 0.8828
Epoch 1/10
4/4 [==============================] - 1s 193ms/step - loss: 0.3606 - accuracy:
0.8750 - val_loss: 0.3416 - val_accuracy: 0.8828
Epoch 2/10
4/4 [==============================] - 1s 228ms/step - loss: 0.1984 - accuracy:
0.9062 - val_loss: 0.3400 - val_accuracy: 0.8828
Epoch 3/10
4/4 [==============================] - 1s 182ms/step - loss: 0.0908 - accuracy:
0.9453 - val_loss: 0.3367 - val_accuracy: 0.8828
Epoch 4/10
4/4 [==============================] - 0s 151ms/step - loss: 0.0635 - accuracy:
0.9844 - val_loss: 0.3363 - val_accuracy: 0.8828
Epoch 5/10
4/4 [==============================] - 1s 157ms/step - loss: 0.0517 - accuracy:
0.9844 - val_loss: 0.3376 - val_accuracy: 0.8789
Epoch 6/10
4/4 [==============================] - 1s 184ms/step - loss: 0.0416 - accuracy:
0.9922 - val_loss: 0.3399 - val_accuracy: 0.8789
Epoch 7/10
4/4 [==============================] - 1s 177ms/step - loss: 0.0358 - accuracy:
1.0000 - val_loss: 0.3431 - val_accuracy: 0.8789
Epoch 8/10
4/4 [==============================] - 1s 177ms/step - loss: 0.0304 - accuracy:
```

```
1.0000 - val_loss: 0.3467 - val_accuracy: 0.8789
Epoch 9/10
4/4 [==============================] - 0s 141ms/step - loss: 0.0260 - accuracy:
1.0000 - val_loss: 0.3502 - val_accuracy: 0.8789
Epoch 10/10
4/4 [==============================] - 1s 202ms/step - loss: 0.0234 - accuracy:
1.0000 - val_loss: 0.3537 - val_accuracy: 0.8789
Epoch 1/10
4/4 [==============================] - 1s 230ms/step - loss: 0.4636 - accuracy:
0.8203 - val_loss: 0.3572 - val_accuracy: 0.8828
Epoch 2/10
4/4 [==============================] - 1s 221ms/step - loss: 0.2424 - accuracy:
0.9062 - val_loss: 0.3521 - val_accuracy: 0.8867
Epoch 3/10
4/4 [==============================] - 1s 214ms/step - loss: 0.1626 - accuracy:
0.9375 - val_loss: 0.3451 - val_accuracy: 0.8867
Epoch 4/10
4/4 [==============================] - 1s 215ms/step - loss: 0.1100 - accuracy:
0.9609 - val_loss: 0.3400 - val_accuracy: 0.8828
Epoch 5/10
4/4 [==============================] - 1s 163ms/step - loss: 0.0835 - accuracy:
0.9609 - val_loss: 0.3347 - val_accuracy: 0.8867
Epoch 6/10
4/4 [==============================] - 1s 176ms/step - loss: 0.0657 - accuracy:
0.9844 - val_loss: 0.3324 - val_accuracy: 0.8867
Epoch 7/10
4/4 [==============================] - 1s 166ms/step - loss: 0.0550 - accuracy:
0.9844 - val_loss: 0.3331 - val_accuracy: 0.8867
Epoch 8/10
4/4 [==============================] - 1s 175ms/step - loss: 0.0470 - accuracy:
0.9922 - val_loss: 0.3342 - val_accuracy: 0.8828
Epoch 9/10
4/4 [==============================] - 1s 168ms/step - loss: 0.0419 - accuracy:
0.9922 - val_loss: 0.3359 - val_accuracy: 0.8867
Epoch 10/10
4/4 [==============================] - 1s 180ms/step - loss: 0.0365 - accuracy:
0.9922 - val_loss: 0.3378 - val_accuracy: 0.8867
Epoch 1/10
4/4 [==============================] - 1s 219ms/step - loss: 0.3306 - accuracy:
0.8906 - val_loss: 0.3117 - val_accuracy: 0.8984
Epoch 2/10
4/4 [==============================] - 1s 208ms/step - loss: 0.1859 - accuracy:
0.9219 - val_loss: 0.3013 - val_accuracy: 0.8984
Epoch 3/10
4/4 [==============================] - 1s 179ms/step - loss: 0.1279 - accuracy:
0.9609 - val_loss: 0.3019 - val_accuracy: 0.8984
Epoch 4/10
4/4 [==============================] - 1s 178ms/step - loss: 0.0920 - accuracy:
```

```
0.9766 - val_loss: 0.3044 - val_accuracy: 0.8906
Epoch 5/10
4/4 [==============================] - 1s 197ms/step - loss: 0.0706 - accuracy:
0.9844 - val_loss: 0.3069 - val_accuracy: 0.8945
Epoch 6/10
4/4 [==============================] - 1s 186ms/step - loss: 0.0528 - accuracy:
0.9922 - val_loss: 0.3090 - val_accuracy: 0.8945
Epoch 7/10
4/4 [==============================] - 1s 169ms/step - loss: 0.0445 - accuracy:
0.9922 - val_loss: 0.3106 - val_accuracy: 0.8984
Epoch 8/10
4/4 [==============================] - 1s 170ms/step - loss: 0.0369 - accuracy:
0.9922 - val_loss: 0.3119 - val_accuracy: 0.8984
Epoch 9/10
4/4 [==============================] - 1s 186ms/step - loss: 0.0325 - accuracy:
0.9922 - val_loss: 0.3130 - val_accuracy: 0.8984
Epoch 10/10
4/4 [==============================] - 1s 158ms/step - loss: 0.0284 - accuracy:
0.9922 - val_loss: 0.3142 - val_accuracy: 0.8984
Epoch 1/10
4/4 [==============================] - 1s 328ms/step - loss: 0.3098 - accuracy:
0.8438 - val_loss: 0.3067 - val_accuracy: 0.9062
Epoch 2/10
4/4 [==============================] - 1s 196ms/step - loss: 0.2121 - accuracy:
0.8828 - val_loss: 0.3068 - val_accuracy: 0.8984
Epoch 3/10
4/4 [==============================] - 1s 180ms/step - loss: 0.1591 - accuracy:
0.9062 - val_loss: 0.3101 - val_accuracy: 0.8906
Epoch 4/10
4/4 [==============================] - 1s 248ms/step - loss: 0.1119 - accuracy:
0.9453 - val_loss: 0.3136 - val_accuracy: 0.8906
Epoch 5/10
4/4 [==============================] - 1s 252ms/step - loss: 0.0855 - accuracy:
0.9531 - val_loss: 0.3151 - val_accuracy: 0.8906
Epoch 6/10
4/4 [==============================] - 1s 210ms/step - loss: 0.0635 - accuracy:
0.9844 - val_loss: 0.3164 - val_accuracy: 0.8867
Epoch 7/10
4/4 [==============================] - 1s 219ms/step - loss: 0.0506 - accuracy:
1.0000 - val_loss: 0.3180 - val_accuracy: 0.8906
Epoch 8/10
4/4 [==============================] - 1s 247ms/step - loss: 0.0412 - accuracy:
1.0000 - val_loss: 0.3198 - val_accuracy: 0.8867
Epoch 9/10
4/4 [==============================] - 1s 176ms/step - loss: 0.0346 - accuracy:
1.0000 - val_loss: 0.3217 - val_accuracy: 0.8867
Epoch 10/10
4/4 [==============================] - 1s 207ms/step - loss: 0.0308 - accuracy:
```

```
1.0000 - val_loss: 0.3239 - val_accuracy: 0.8867
Epoch 1/10
4/4 [==============================] - 1s 257ms/step - loss: 0.3580 - accuracy:
0.8672 - val_loss: 0.3172 - val_accuracy: 0.8945
Epoch 2/10
4/4 [==============================] - 1s 164ms/step - loss: 0.2664 - accuracy:
0.8750 - val_loss: 0.3070 - val_accuracy: 0.9023
Epoch 3/10
4/4 [==============================] - 1s 174ms/step - loss: 0.1806 - accuracy:
0.8906 - val_loss: 0.3001 - val_accuracy: 0.9062
Epoch 4/10
4/4 [==============================] - 1s 161ms/step - loss: 0.1242 - accuracy:
0.9297 - val_loss: 0.2957 - val_accuracy: 0.9023
Epoch 5/10
4/4 [==============================] - 1s 199ms/step - loss: 0.0928 - accuracy:
0.9766 - val_loss: 0.2927 - val_accuracy: 0.8984
Epoch 6/10
4/4 [==============================] - 1s 174ms/step - loss: 0.0724 - accuracy:
0.9922 - val_loss: 0.2914 - val_accuracy: 0.8984
Epoch 7/10
4/4 [==============================] - 0s 140ms/step - loss: 0.0617 - accuracy:
0.9922 - val_loss: 0.2912 - val_accuracy: 0.8945
Epoch 8/10
4/4 [==============================] - 1s 165ms/step - loss: 0.0539 - accuracy:
0.9922 - val_loss: 0.2918 - val_accuracy: 0.8945
Epoch 9/10
4/4 [==============================] - 1s 175ms/step - loss: 0.0489 - accuracy:
0.9922 - val_loss: 0.2929 - val_accuracy: 0.8945
Epoch 10/10
4/4 [==============================] - 1s 158ms/step - loss: 0.0439 - accuracy:
0.9922 - val_loss: 0.2942 - val_accuracy: 0.8945
Epoch 1/10
4/4 [==============================] - 1s 254ms/step - loss: 0.5162 - accuracy:
0.8203 - val_loss: 0.3005 - val_accuracy: 0.8945
Epoch 2/10
4/4 [==============================] - 1s 183ms/step - loss: 0.3055 - accuracy:
0.8750 - val_loss: 0.3120 - val_accuracy: 0.8945
Epoch 3/10
4/4 [==============================] - 1s 178ms/step - loss: 0.2092 - accuracy:
0.9141 - val_loss: 0.3194 - val_accuracy: 0.8945
Epoch 4/10
4/4 [==============================] - 0s 139ms/step - loss: 0.1514 - accuracy:
0.9219 - val_loss: 0.3270 - val_accuracy: 0.8906
Epoch 5/10
4/4 [==============================] - 1s 160ms/step - loss: 0.1106 - accuracy:
0.9531 - val_loss: 0.3331 - val_accuracy: 0.8828
Epoch 6/10
4/4 [==============================] - 0s 138ms/step - loss: 0.0780 - accuracy:
```

```
0.9766 - val_loss: 0.3358 - val_accuracy: 0.8828
Epoch 7/10
4/4 [==============================] - 1s 161ms/step - loss: 0.0631 - accuracy:
0.9844 - val_loss: 0.3380 - val_accuracy: 0.8828
Epoch 8/10
4/4 [==============================] - 1s 215ms/step - loss: 0.0521 - accuracy:
0.9922 - val_loss: 0.3388 - val_accuracy: 0.8828
Epoch 9/10
4/4 [==============================] - 1s 213ms/step - loss: 0.0461 - accuracy:
0.9922 - val_loss: 0.3407 - val_accuracy: 0.8828
Epoch 10/10
4/4 [==============================] - 1s 176ms/step - loss: 0.0405 - accuracy:
0.9922 - val_loss: 0.3421 - val_accuracy: 0.8828
Epoch 1/10
4/4 [==============================] - 1s 272ms/step - loss: 0.4285 - accuracy:
0.7812 - val_loss: 0.2951 - val_accuracy: 0.8984
Epoch 2/10
4/4 [==============================] - 0s 152ms/step - loss: 0.2145 - accuracy:
0.8828 - val_loss: 0.2666 - val_accuracy: 0.9062
Epoch 3/10
4/4 [==============================] - 1s 174ms/step - loss: 0.1424 - accuracy:
0.9297 - val_loss: 0.2655 - val_accuracy: 0.9062
Epoch 4/10
4/4 [==============================] - 0s 144ms/step - loss: 0.0984 - accuracy:
0.9922 - val_loss: 0.2686 - val_accuracy: 0.8984
Epoch 5/10
4/4 [==============================] - 0s 140ms/step - loss: 0.0776 - accuracy:
0.9922 - val_loss: 0.2709 - val_accuracy: 0.8906
Epoch 6/10
4/4 [==============================] - 1s 165ms/step - loss: 0.0591 - accuracy:
0.9922 - val_loss: 0.2723 - val_accuracy: 0.8867
Epoch 7/10
4/4 [==============================] - 1s 166ms/step - loss: 0.0478 - accuracy:
0.9922 - val_loss: 0.2738 - val_accuracy: 0.8906
Epoch 8/10
4/4 [==============================] - 1s 158ms/step - loss: 0.0400 - accuracy:
1.0000 - val_loss: 0.2751 - val_accuracy: 0.8906
Epoch 9/10
4/4 [==============================] - 1s 190ms/step - loss: 0.0342 - accuracy:
1.0000 - val_loss: 0.2764 - val_accuracy: 0.8906
Epoch 10/10
4/4 [==============================] - 1s 173ms/step - loss: 0.0300 - accuracy:
1.0000 - val_loss: 0.2776 - val_accuracy: 0.8906
Epoch 1/10
4/4 [==============================] - 1s 252ms/step - loss: 0.3616 - accuracy:
0.8594 - val_loss: 0.2806 - val_accuracy: 0.8750
Epoch 2/10
4/4 [==============================] - 1s 235ms/step - loss: 0.2490 - accuracy:
```

```
0.8906 - val_loss: 0.2779 - val_accuracy: 0.8789
Epoch 3/10
4/4 [==============================] - 1s 185ms/step - loss: 0.1906 - accuracy:
0.9141 - val_loss: 0.2765 - val_accuracy: 0.8867
Epoch 4/10
4/4 [==============================] - 1s 199ms/step - loss: 0.1321 - accuracy:
0.9375 - val_loss: 0.2729 - val_accuracy: 0.8867
Epoch 5/10
4/4 [==============================] - 1s 154ms/step - loss: 0.0988 - accuracy:
0.9609 - val_loss: 0.2713 - val_accuracy: 0.8906
Epoch 6/10
4/4 [==============================] - 1s 184ms/step - loss: 0.0778 - accuracy:
0.9766 - val_loss: 0.2712 - val_accuracy: 0.8945
Epoch 7/10
4/4 [==============================] - 1s 171ms/step - loss: 0.0660 - accuracy:
0.9844 - val_loss: 0.2718 - val_accuracy: 0.8945
Epoch 8/10
4/4 [==============================] - 1s 178ms/step - loss: 0.0545 - accuracy:
0.9844 - val_loss: 0.2732 - val_accuracy: 0.8945
Epoch 9/10
4/4 [==============================] - 1s 192ms/step - loss: 0.0474 - accuracy:
0.9844 - val_loss: 0.2745 - val_accuracy: 0.8945
Epoch 10/10
4/4 [==============================] - 1s 188ms/step - loss: 0.0416 - accuracy:
0.9922 - val_loss: 0.2758 - val_accuracy: 0.8945
Epoch 1/10
4/4 [==============================] - 1s 296ms/step - loss: 0.3601 - accuracy:
0.8281 - val_loss: 0.2695 - val_accuracy: 0.9062
Epoch 2/10
4/4 [==============================] - 1s 205ms/step - loss: 0.2302 - accuracy:
0.8594 - val_loss: 0.2709 - val_accuracy: 0.9023
Epoch 3/10
4/4 [==============================] - 1s 186ms/step - loss: 0.1637 - accuracy:
0.8984 - val_loss: 0.2788 - val_accuracy: 0.9062
Epoch 4/10
4/4 [==============================] - 1s 188ms/step - loss: 0.1181 - accuracy:
0.9375 - val_loss: 0.2863 - val_accuracy: 0.9062
Epoch 5/10
4/4 [==============================] - 1s 211ms/step - loss: 0.0923 - accuracy:
0.9609 - val_loss: 0.2914 - val_accuracy: 0.9023
Epoch 6/10
4/4 [==============================] - 1s 248ms/step - loss: 0.0713 - accuracy:
0.9844 - val_loss: 0.2954 - val_accuracy: 0.8945
Epoch 7/10
4/4 [==============================] - 1s 178ms/step - loss: 0.0621 - accuracy:
0.9844 - val_loss: 0.2990 - val_accuracy: 0.8906
Epoch 8/10
4/4 [==============================] - 1s 178ms/step - loss: 0.0563 - accuracy:
```

```
0.9844 - val_loss: 0.3032 - val_accuracy: 0.8906
Epoch 9/10
4/4 [==============================] - 1s 169ms/step - loss: 0.0499 - accuracy:
0.9844 - val_loss: 0.3068 - val_accuracy: 0.8906
Epoch 10/10
4/4 [==============================] - 1s 231ms/step - loss: 0.0449 - accuracy:
0.9844 - val_loss: 0.3099 - val_accuracy: 0.8906
Epoch 1/10
4/4 [==============================] - 1s 236ms/step - loss: 0.4885 - accuracy:
0.8125 - val_loss: 0.2719 - val_accuracy: 0.9062
Epoch 2/10
4/4 [==============================] - 1s 170ms/step - loss: 0.2538 - accuracy:
0.8594 - val_loss: 0.2625 - val_accuracy: 0.9023
Epoch 3/10
4/4 [==============================] - 1s 210ms/step - loss: 0.1652 - accuracy:
0.9375 - val_loss: 0.2617 - val_accuracy: 0.8984
Epoch 4/10
4/4 [==============================] - 1s 159ms/step - loss: 0.1169 - accuracy:
0.9453 - val_loss: 0.2629 - val_accuracy: 0.8945
Epoch 5/10
4/4 [==============================] - 1s 160ms/step - loss: 0.0916 - accuracy:
0.9531 - val_loss: 0.2652 - val_accuracy: 0.8867
Epoch 6/10
4/4 [==============================] - 1s 237ms/step - loss: 0.0755 - accuracy:
0.9766 - val_loss: 0.2662 - val_accuracy: 0.8867
Epoch 7/10
4/4 [==============================] - 1s 241ms/step - loss: 0.0643 - accuracy:
0.9844 - val_loss: 0.2669 - val_accuracy: 0.8906
Epoch 8/10
4/4 [==============================] - 1s 220ms/step - loss: 0.0554 - accuracy:
0.9844 - val_loss: 0.2674 - val_accuracy: 0.8906
Epoch 9/10
4/4 [==============================] - 1s 219ms/step - loss: 0.0502 - accuracy:
0.9844 - val_loss: 0.2677 - val_accuracy: 0.8906
Epoch 10/10
4/4 [==============================] - 1s 199ms/step - loss: 0.0457 - accuracy:
0.9844 - val_loss: 0.2682 - val_accuracy: 0.8906
Epoch 1/10
4/4 [==============================] - 1s 228ms/step - loss: 0.3493 - accuracy:
0.8359 - val_loss: 0.2677 - val_accuracy: 0.8906
Epoch 2/10
4/4 [==============================] - 1s 178ms/step - loss: 0.2497 - accuracy:
0.9062 - val_loss: 0.2657 - val_accuracy: 0.8906
Epoch 3/10
4/4 [==============================] - 0s 149ms/step - loss: 0.1968 - accuracy:
0.9219 - val_loss: 0.2644 - val_accuracy: 0.8945
Epoch 4/10
4/4 [==============================] - 1s 207ms/step - loss: 0.1536 - accuracy:
```

```
0.9375 - val_loss: 0.2633 - val_accuracy: 0.8945
Epoch 5/10
4/4 [==============================] - 1s 196ms/step - loss: 0.1104 - accuracy:
0.9531 - val_loss: 0.2621 - val_accuracy: 0.8945
Epoch 6/10
4/4 [==============================] - 0s 152ms/step - loss: 0.0913 - accuracy:
0.9609 - val_loss: 0.2629 - val_accuracy: 0.8906
Epoch 7/10
4/4 [==============================] - 1s 172ms/step - loss: 0.0700 - accuracy:
0.9766 - val_loss: 0.2638 - val_accuracy: 0.8906
Epoch 8/10
4/4 [==============================] - 1s 191ms/step - loss: 0.0573 - accuracy:
0.9844 - val_loss: 0.2650 - val_accuracy: 0.8906
Epoch 9/10
4/4 [==============================] - 1s 167ms/step - loss: 0.0478 - accuracy:
0.9922 - val_loss: 0.2659 - val_accuracy: 0.8906
Epoch 10/10
4/4 [==============================] - 0s 122ms/step - loss: 0.0407 - accuracy:
0.9922 - val_loss: 0.2672 - val_accuracy: 0.8984
Epoch 1/10
4/4 [==============================] - 1s 304ms/step - loss: 0.3014 - accuracy:
0.8672 - val_loss: 0.2667 - val_accuracy: 0.8906
Epoch 2/10
4/4 [==============================] - 1s 221ms/step - loss: 0.2235 - accuracy:
0.8984 - val_loss: 0.2654 - val_accuracy: 0.8984
Epoch 3/10
4/4 [==============================] - 1s 206ms/step - loss: 0.1657 - accuracy:
0.9219 - val_loss: 0.2668 - val_accuracy: 0.8984
Epoch 4/10
4/4 [==============================] - 1s 178ms/step - loss: 0.1187 - accuracy:
0.9531 - val_loss: 0.2689 - val_accuracy: 0.9023
Epoch 5/10
4/4 [==============================] - 1s 219ms/step - loss: 0.0898 - accuracy:
0.9766 - val_loss: 0.2715 - val_accuracy: 0.9062
Epoch 6/10
4/4 [==============================] - 1s 206ms/step - loss: 0.0688 - accuracy:
0.9922 - val_loss: 0.2735 - val_accuracy: 0.9062
Epoch 7/10
4/4 [==============================] - 1s 196ms/step - loss: 0.0559 - accuracy:
0.9922 - val_loss: 0.2745 - val_accuracy: 0.9062
Epoch 8/10
4/4 [==============================] - 1s 185ms/step - loss: 0.0443 - accuracy:
0.9922 - val_loss: 0.2753 - val_accuracy: 0.9062
Epoch 9/10
4/4 [==============================] - 1s 241ms/step - loss: 0.0362 - accuracy:
0.9922 - val_loss: 0.2762 - val_accuracy: 0.9062
Epoch 10/10
4/4 [==============================] - 1s 253ms/step - loss: 0.0309 - accuracy:
```

```
0.9922 - val_loss: 0.2771 - val_accuracy: 0.9062
Epoch 1/10
4/4 [==============================] - 1s 258ms/step - loss: 0.4961 - accuracy:
0.8438 - val_loss: 0.2650 - val_accuracy: 0.9062
Epoch 2/10
4/4 [==============================] - 1s 176ms/step - loss: 0.1721 - accuracy:
0.9531 - val_loss: 0.2621 - val_accuracy: 0.8984
Epoch 3/10
4/4 [==============================] - 1s 175ms/step - loss: 0.1138 - accuracy:
0.9688 - val_loss: 0.2660 - val_accuracy: 0.8906
Epoch 4/10
4/4 [==============================] - 0s 128ms/step - loss: 0.0564 - accuracy:
0.9844 - val_loss: 0.2699 - val_accuracy: 0.8828
Epoch 5/10
4/4 [==============================] - 1s 162ms/step - loss: 0.0420 - accuracy:
1.0000 - val_loss: 0.2723 - val_accuracy: 0.8789
Epoch 6/10
4/4 [==============================] - 1s 185ms/step - loss: 0.0317 - accuracy:
1.0000 - val_loss: 0.2738 - val_accuracy: 0.8828
Epoch 7/10
4/4 [==============================] - 1s 153ms/step - loss: 0.0253 - accuracy:
1.0000 - val_loss: 0.2746 - val_accuracy: 0.8828
Epoch 8/10
4/4 [==============================] - 1s 173ms/step - loss: 0.0209 - accuracy:
1.0000 - val_loss: 0.2751 - val_accuracy: 0.8867
Epoch 9/10
4/4 [==============================] - 1s 168ms/step - loss: 0.0179 - accuracy:
1.0000 - val_loss: 0.2755 - val_accuracy: 0.8867
Epoch 10/10
4/4 [==============================] - 1s 179ms/step - loss: 0.0160 - accuracy:
1.0000 - val_loss: 0.2761 - val_accuracy: 0.8867
Epoch 1/10
4/4 [==============================] - 1s 240ms/step - loss: 0.4359 - accuracy:
0.8281 - val_loss: 0.2606 - val_accuracy: 0.8945
Epoch 2/10
4/4 [==============================] - 1s 186ms/step - loss: 0.2156 - accuracy:
0.8906 - val_loss: 0.2689 - val_accuracy: 0.8945
Epoch 3/10
4/4 [==============================] - 1s 168ms/step - loss: 0.1618 - accuracy:
0.9453 - val_loss: 0.2822 - val_accuracy: 0.8945
Epoch 4/10
4/4 [==============================] - 1s 156ms/step - loss: 0.1226 - accuracy:
0.9531 - val_loss: 0.2892 - val_accuracy: 0.8789
Epoch 5/10
4/4 [==============================] - 1s 207ms/step - loss: 0.0925 - accuracy:
0.9688 - val_loss: 0.2926 - val_accuracy: 0.8789
Epoch 6/10
4/4 [==============================] - 1s 203ms/step - loss: 0.0750 - accuracy:
```

```
0.9766 - val_loss: 0.2940 - val_accuracy: 0.8750
Epoch 7/10
4/4 [==============================] - 1s 174ms/step - loss: 0.0634 - accuracy:
0.9844 - val_loss: 0.2938 - val_accuracy: 0.8750
Epoch 8/10
4/4 [==============================] - 0s 146ms/step - loss: 0.0529 - accuracy:
0.9844 - val_loss: 0.2937 - val_accuracy: 0.8750
Epoch 9/10
4/4 [==============================] - 1s 176ms/step - loss: 0.0464 - accuracy:
0.9844 - val_loss: 0.2930 - val_accuracy: 0.8789
Epoch 10/10
4/4 [==============================] - 1s 194ms/step - loss: 0.0399 - accuracy:
0.9844 - val_loss: 0.2929 - val_accuracy: 0.8789
Epoch 1/10
4/4 [==============================] - 1s 263ms/step - loss: 0.3506 - accuracy:
0.8438 - val_loss: 0.2739 - val_accuracy: 0.8945
Epoch 2/10
4/4 [==============================] - 1s 173ms/step - loss: 0.2148 - accuracy:
0.8906 - val_loss: 0.2699 - val_accuracy: 0.8906
Epoch 3/10
4/4 [==============================] - 1s 197ms/step - loss: 0.1442 - accuracy:
0.9219 - val_loss: 0.2706 - val_accuracy: 0.8867
Epoch 4/10
4/4 [==============================] - 1s 202ms/step - loss: 0.1046 - accuracy:
0.9688 - val_loss: 0.2744 - val_accuracy: 0.8789
Epoch 5/10
4/4 [==============================] - 1s 168ms/step - loss: 0.0745 - accuracy:
0.9766 - val_loss: 0.2770 - val_accuracy: 0.8828
Epoch 6/10
4/4 [==============================] - 1s 166ms/step - loss: 0.0568 - accuracy:
0.9844 - val_loss: 0.2790 - val_accuracy: 0.8828
Epoch 7/10
4/4 [==============================] - 1s 224ms/step - loss: 0.0448 - accuracy:
0.9844 - val_loss: 0.2807 - val_accuracy: 0.8828
Epoch 8/10
4/4 [==============================] - 1s 220ms/step - loss: 0.0371 - accuracy:
0.9922 - val_loss: 0.2820 - val_accuracy: 0.8828
Epoch 9/10
4/4 [==============================] - 1s 169ms/step - loss: 0.0311 - accuracy:
1.0000 - val_loss: 0.2836 - val_accuracy: 0.8828
Epoch 10/10
4/4 [==============================] - 1s 191ms/step - loss: 0.0269 - accuracy:
1.0000 - val_loss: 0.2852 - val_accuracy: 0.8828
Epoch 1/10
4/4 [==============================] - 1s 248ms/step - loss: 0.3356 - accuracy:
0.8750 - val_loss: 0.2921 - val_accuracy: 0.8789
Epoch 2/10
4/4 [==============================] - 1s 169ms/step - loss: 0.1613 - accuracy:
```

```
0.9297 - val_loss: 0.2865 - val_accuracy: 0.8828
Epoch 3/10
4/4 [==============================] - 1s 189ms/step - loss: 0.1109 - accuracy:
0.9531 - val_loss: 0.2819 - val_accuracy: 0.8945
Epoch 4/10
4/4 [==============================] - 1s 169ms/step - loss: 0.0771 - accuracy:
0.9688 - val_loss: 0.2793 - val_accuracy: 0.8906
Epoch 5/10
4/4 [==============================] - 1s 218ms/step - loss: 0.0556 - accuracy:
0.9766 - val_loss: 0.2781 - val_accuracy: 0.8906
Epoch 6/10
4/4 [==============================] - 1s 188ms/step - loss: 0.0417 - accuracy:
0.9922 - val_loss: 0.2775 - val_accuracy: 0.8906
Epoch 7/10
4/4 [==============================] - 1s 202ms/step - loss: 0.0339 - accuracy:
0.9922 - val_loss: 0.2773 - val_accuracy: 0.8906
Epoch 8/10
4/4 [==============================] - 0s 139ms/step - loss: 0.0268 - accuracy:
0.9922 - val_loss: 0.2774 - val_accuracy: 0.8906
Epoch 9/10
4/4 [==============================] - 0s 106ms/step - loss: 0.0217 - accuracy:
0.9922 - val_loss: 0.2779 - val_accuracy: 0.8906
Epoch 10/10
4/4 [==============================] - 1s 185ms/step - loss: 0.0190 - accuracy:
1.0000 - val_loss: 0.2785 - val_accuracy: 0.8906
Epoch 1/10
4/4 [==============================] - 1s 243ms/step - loss: 0.2812 - accuracy:
0.8828 - val_loss: 0.2744 - val_accuracy: 0.8945
Epoch 2/10
4/4 [==============================] - 1s 185ms/step - loss: 0.1333 - accuracy:
0.9375 - val_loss: 0.2756 - val_accuracy: 0.8906
Epoch 3/10
4/4 [==============================] - 0s 153ms/step - loss: 0.0944 - accuracy:
0.9688 - val_loss: 0.2790 - val_accuracy: 0.8867
Epoch 4/10
4/4 [==============================] - 0s 131ms/step - loss: 0.0699 - accuracy:
0.9922 - val_loss: 0.2814 - val_accuracy: 0.8906
Epoch 5/10
4/4 [==============================] - 1s 166ms/step - loss: 0.0525 - accuracy:
0.9922 - val_loss: 0.2820 - val_accuracy: 0.8906
Epoch 6/10
4/4 [==============================] - 1s 183ms/step - loss: 0.0374 - accuracy:
0.9922 - val_loss: 0.2823 - val_accuracy: 0.8906
Epoch 7/10
4/4 [==============================] - 1s 187ms/step - loss: 0.0294 - accuracy:
1.0000 - val_loss: 0.2823 - val_accuracy: 0.8945
Epoch 8/10
4/4 [==============================] - 0s 121ms/step - loss: 0.0238 - accuracy:
```

```
1.0000 - val_loss: 0.2829 - val_accuracy: 0.8945
Epoch 9/10
4/4 [==============================] - 0s 133ms/step - loss: 0.0203 - accuracy:
1.0000 - val_loss: 0.2839 - val_accuracy: 0.8945
Epoch 10/10
4/4 [==============================] - 1s 156ms/step - loss: 0.0178 - accuracy:
1.0000 - val_loss: 0.2851 - val_accuracy: 0.8945
Epoch 1/10
4/4 [==============================] - 1s 290ms/step - loss: 0.4182 - accuracy:
0.8672 - val_loss: 0.2945 - val_accuracy: 0.8867
Epoch 2/10
4/4 [==============================] - 1s 209ms/step - loss: 0.1730 - accuracy:
0.9219 - val_loss: 0.3035 - val_accuracy: 0.8867
Epoch 3/10
4/4 [==============================] - 1s 209ms/step - loss: 0.1161 - accuracy:
0.9609 - val_loss: 0.3015 - val_accuracy: 0.8789
Epoch 4/10
4/4 [==============================] - 1s 208ms/step - loss: 0.0752 - accuracy:
0.9766 - val_loss: 0.3021 - val_accuracy: 0.8789
Epoch 5/10
4/4 [==============================] - 1s 211ms/step - loss: 0.0590 - accuracy:
0.9844 - val_loss: 0.3024 - val_accuracy: 0.8789
Epoch 6/10
4/4 [==============================] - 1s 154ms/step - loss: 0.0448 - accuracy:
0.9844 - val_loss: 0.3033 - val_accuracy: 0.8789
Epoch 7/10
4/4 [==============================] - 1s 185ms/step - loss: 0.0355 - accuracy:
0.9922 - val_loss: 0.3050 - val_accuracy: 0.8789
Epoch 8/10
4/4 [==============================] - 1s 154ms/step - loss: 0.0307 - accuracy:
0.9922 - val_loss: 0.3064 - val_accuracy: 0.8789
Epoch 9/10
4/4 [==============================] - 1s 194ms/step - loss: 0.0259 - accuracy:
1.0000 - val_loss: 0.3080 - val_accuracy: 0.8750
Epoch 10/10
4/4 [==============================] - 1s 210ms/step - loss: 0.0233 - accuracy:
1.0000 - val_loss: 0.3099 - val_accuracy: 0.8750
Epoch 1/10
4/4 [==============================] - 1s 282ms/step - loss: 0.4986 - accuracy:
0.7891 - val_loss: 0.2818 - val_accuracy: 0.8867
Epoch 2/10
4/4 [==============================] - 1s 175ms/step - loss: 0.3166 - accuracy:
0.8672 - val_loss: 0.2840 - val_accuracy: 0.8711
Epoch 3/10
4/4 [==============================] - 1s 194ms/step - loss: 0.2100 - accuracy:
0.9062 - val_loss: 0.2976 - val_accuracy: 0.8672
Epoch 4/10
4/4 [==============================] - 1s 176ms/step - loss: 0.1423 - accuracy:
```

```
0.9453 - val_loss: 0.3098 - val_accuracy: 0.8594
Epoch 5/10
4/4 [==============================] - 1s 212ms/step - loss: 0.1083 - accuracy:
0.9453 - val_loss: 0.3179 - val_accuracy: 0.8555
Epoch 6/10
4/4 [==============================] - 1s 177ms/step - loss: 0.0834 - accuracy:
0.9609 - val_loss: 0.3223 - val_accuracy: 0.8555
Epoch 7/10
4/4 [==============================] - 1s 228ms/step - loss: 0.0640 - accuracy:
0.9766 - val_loss: 0.3209 - val_accuracy: 0.8594
Epoch 8/10
4/4 [==============================] - 1s 166ms/step - loss: 0.0523 - accuracy:
0.9844 - val_loss: 0.3172 - val_accuracy: 0.8594
Epoch 9/10
4/4 [==============================] - 1s 197ms/step - loss: 0.0427 - accuracy:
1.0000 - val_loss: 0.3139 - val_accuracy: 0.8633
Epoch 10/10
4/4 [==============================] - 1s 189ms/step - loss: 0.0361 - accuracy:
1.0000 - val_loss: 0.3110 - val_accuracy: 0.8711
Epoch 1/10
4/4 [==============================] - 1s 251ms/step - loss: 0.2935 - accuracy:
0.8438 - val_loss: 0.2969 - val_accuracy: 0.8867
Epoch 2/10
4/4 [==============================] - 1s 197ms/step - loss: 0.2127 - accuracy:
0.8828 - val_loss: 0.2823 - val_accuracy: 0.8867
Epoch 3/10
4/4 [==============================] - 1s 196ms/step - loss: 0.1418 - accuracy:
0.9375 - val_loss: 0.2756 - val_accuracy: 0.8906
Epoch 4/10
4/4 [==============================] - 1s 201ms/step - loss: 0.0953 - accuracy:
0.9688 - val_loss: 0.2720 - val_accuracy: 0.8906
Epoch 5/10
4/4 [==============================] - 1s 189ms/step - loss: 0.0708 - accuracy:
0.9766 - val_loss: 0.2707 - val_accuracy: 0.8867
Epoch 6/10
4/4 [==============================] - 1s 172ms/step - loss: 0.0533 - accuracy:
0.9922 - val_loss: 0.2708 - val_accuracy: 0.8867
Epoch 7/10
4/4 [==============================] - 1s 248ms/step - loss: 0.0437 - accuracy:
0.9922 - val_loss: 0.2716 - val_accuracy: 0.8828
Epoch 8/10
4/4 [==============================] - 1s 184ms/step - loss: 0.0373 - accuracy:
0.9922 - val_loss: 0.2730 - val_accuracy: 0.8789
Epoch 9/10
4/4 [==============================] - 1s 168ms/step - loss: 0.0315 - accuracy:
0.9922 - val_loss: 0.2749 - val_accuracy: 0.8789
Epoch 10/10
4/4 [==============================] - 1s 206ms/step - loss: 0.0270 - accuracy:
```

```
1.0000 - val_loss: 0.2766 - val_accuracy: 0.8789
Epoch 1/10
4/4 [==============================] - 1s 267ms/step - loss: 0.5686 - accuracy:
0.8672 - val_loss: 0.2726 - val_accuracy: 0.8789
Epoch 2/10
4/4 [==============================] - 1s 164ms/step - loss: 0.3042 - accuracy:
0.9062 - val_loss: 0.2723 - val_accuracy: 0.8828
Epoch 3/10
4/4 [==============================] - 1s 157ms/step - loss: 0.1371 - accuracy:
0.9531 - val_loss: 0.2799 - val_accuracy: 0.8867
Epoch 4/10
4/4 [==============================] - 1s 226ms/step - loss: 0.1042 - accuracy:
0.9766 - val_loss: 0.2842 - val_accuracy: 0.8867
Epoch 5/10
4/4 [==============================] - 1s 198ms/step - loss: 0.0847 - accuracy:
0.9766 - val_loss: 0.2854 - val_accuracy: 0.8828
Epoch 6/10
4/4 [==============================] - 1s 219ms/step - loss: 0.0701 - accuracy:
0.9766 - val_loss: 0.2857 - val_accuracy: 0.8828
Epoch 7/10
4/4 [==============================] - 1s 221ms/step - loss: 0.0586 - accuracy:
0.9766 - val_loss: 0.2858 - val_accuracy: 0.8789
Epoch 8/10
4/4 [==============================] - 1s 164ms/step - loss: 0.0517 - accuracy:
0.9844 - val_loss: 0.2858 - val_accuracy: 0.8867
Epoch 9/10
4/4 [==============================] - 1s 157ms/step - loss: 0.0442 - accuracy:
0.9844 - val_loss: 0.2862 - val_accuracy: 0.8867
Epoch 10/10
4/4 [==============================] - 0s 135ms/step - loss: 0.0380 - accuracy:
0.9844 - val_loss: 0.2869 - val_accuracy: 0.8867
Epoch 1/10
4/4 [==============================] - 1s 218ms/step - loss: 0.2484 - accuracy:
0.8906 - val_loss: 0.2848 - val_accuracy: 0.8867
Epoch 2/10
4/4 [==============================] - 1s 218ms/step - loss: 0.1801 - accuracy:
0.9219 - val_loss: 0.2847 - val_accuracy: 0.8906
Epoch 3/10
4/4 [==============================] - 1s 195ms/step - loss: 0.1350 - accuracy:
0.9297 - val_loss: 0.2854 - val_accuracy: 0.8750
Epoch 4/10
4/4 [==============================] - 1s 206ms/step - loss: 0.0911 - accuracy:
0.9531 - val_loss: 0.2874 - val_accuracy: 0.8750
Epoch 5/10
4/4 [==============================] - 1s 170ms/step - loss: 0.0617 - accuracy:
0.9844 - val_loss: 0.2918 - val_accuracy: 0.8750
Epoch 6/10
4/4 [==============================] - 1s 183ms/step - loss: 0.0484 - accuracy:
```

```
0.9844 - val_loss: 0.2965 - val_accuracy: 0.8750
Epoch 7/10
4/4 [==============================] - 1s 197ms/step - loss: 0.0376 - accuracy:
0.9922 - val_loss: 0.3010 - val_accuracy: 0.8750
Epoch 8/10
4/4 [==============================] - 1s 207ms/step - loss: 0.0314 - accuracy:
1.0000 - val_loss: 0.3060 - val_accuracy: 0.8711
Epoch 9/10
4/4 [==============================] - 1s 179ms/step - loss: 0.0267 - accuracy:
1.0000 - val_loss: 0.3113 - val_accuracy: 0.8711
Epoch 10/10
4/4 [==============================] - 1s 168ms/step - loss: 0.0245 - accuracy:
1.0000 - val_loss: 0.3167 - val_accuracy: 0.8711
Epoch 1/10
4/4 [==============================] - 1s 215ms/step - loss: 0.6189 - accuracy:
0.8281 - val_loss: 0.2767 - val_accuracy: 0.8945
Epoch 2/10
4/4 [==============================] - 1s 177ms/step - loss: 0.2335 - accuracy:
0.8906 - val_loss: 0.2684 - val_accuracy: 0.8789
Epoch 3/10
4/4 [==============================] - 0s 151ms/step - loss: 0.1318 - accuracy:
0.9453 - val_loss: 0.2853 - val_accuracy: 0.8711
Epoch 4/10
4/4 [==============================] - 1s 185ms/step - loss: 0.0918 - accuracy:
0.9844 - val_loss: 0.3047 - val_accuracy: 0.8477
Epoch 5/10
4/4 [==============================] - 1s 166ms/step - loss: 0.0713 - accuracy:
0.9844 - val_loss: 0.3202 - val_accuracy: 0.8320
Epoch 6/10
4/4 [==============================] - 1s 256ms/step - loss: 0.0544 - accuracy:
0.9922 - val_loss: 0.3293 - val_accuracy: 0.8320
Epoch 7/10
4/4 [==============================] - 1s 200ms/step - loss: 0.0449 - accuracy:
0.9922 - val_loss: 0.3356 - val_accuracy: 0.8320
Epoch 8/10
4/4 [==============================] - 1s 191ms/step - loss: 0.0390 - accuracy:
0.9922 - val_loss: 0.3402 - val_accuracy: 0.8320
Epoch 9/10
4/4 [==============================] - 1s 172ms/step - loss: 0.0317 - accuracy:
0.9922 - val_loss: 0.3440 - val_accuracy: 0.8359
Epoch 10/10
4/4 [==============================] - 0s 152ms/step - loss: 0.0273 - accuracy:
0.9922 - val_loss: 0.3474 - val_accuracy: 0.8359
Epoch 1/10
4/4 [==============================] - 1s 265ms/step - loss: 0.4826 - accuracy:
0.7891 - val_loss: 0.3208 - val_accuracy: 0.8398
Epoch 2/10
4/4 [==============================] - 1s 159ms/step - loss: 0.2811 - accuracy:
```

```
0.8594 - val_loss: 0.2826 - val_accuracy: 0.8555
Epoch 3/10
4/4 [==============================] - 0s 152ms/step - loss: 0.1453 - accuracy:
0.9297 - val_loss: 0.2657 - val_accuracy: 0.8672
Epoch 4/10
4/4 [==============================] - 1s 182ms/step - loss: 0.0894 - accuracy:
0.9766 - val_loss: 0.2676 - val_accuracy: 0.8750
Epoch 5/10
4/4 [==============================] - 1s 176ms/step - loss: 0.0624 - accuracy:
0.9844 - val_loss: 0.2745 - val_accuracy: 0.8711
Epoch 6/10
4/4 [==============================] - 1s 216ms/step - loss: 0.0477 - accuracy:
0.9922 - val_loss: 0.2805 - val_accuracy: 0.8711
Epoch 7/10
4/4 [==============================] - 1s 198ms/step - loss: 0.0368 - accuracy:
0.9922 - val_loss: 0.2841 - val_accuracy: 0.8711
Epoch 8/10
4/4 [==============================] - 1s 182ms/step - loss: 0.0315 - accuracy:
0.9922 - val_loss: 0.2863 - val_accuracy: 0.8711
Epoch 9/10
4/4 [==============================] - 1s 211ms/step - loss: 0.0267 - accuracy:
0.9922 - val_loss: 0.2875 - val_accuracy: 0.8672
Epoch 10/10
4/4 [==============================] - 1s 189ms/step - loss: 0.0233 - accuracy:
1.0000 - val_loss: 0.2884 - val_accuracy: 0.8672
Epoch 1/10
4/4 [==============================] - 1s 347ms/step - loss: 0.4519 - accuracy:
0.8047 - val_loss: 0.2824 - val_accuracy: 0.8672
Epoch 2/10
4/4 [==============================] - 1s 219ms/step - loss: 0.2756 - accuracy:
0.8906 - val_loss: 0.2796 - val_accuracy: 0.8750
Epoch 3/10
4/4 [==============================] - 1s 200ms/step - loss: 0.1940 - accuracy:
0.9297 - val_loss: 0.2821 - val_accuracy: 0.8750
Epoch 4/10
4/4 [==============================] - 1s 181ms/step - loss: 0.1372 - accuracy:
0.9688 - val_loss: 0.2858 - val_accuracy: 0.8672
Epoch 5/10
4/4 [==============================] - 1s 179ms/step - loss: 0.1001 - accuracy:
0.9844 - val_loss: 0.2892 - val_accuracy: 0.8594
Epoch 6/10
4/4 [==============================] - 1s 200ms/step - loss: 0.0784 - accuracy:
0.9922 - val_loss: 0.2923 - val_accuracy: 0.8594
Epoch 7/10
4/4 [==============================] - 1s 174ms/step - loss: 0.0615 - accuracy:
0.9922 - val_loss: 0.2951 - val_accuracy: 0.8594
Epoch 8/10
4/4 [==============================] - 1s 185ms/step - loss: 0.0520 - accuracy:
```

```
0.9922 - val_loss: 0.2976 - val_accuracy: 0.8594
Epoch 9/10
4/4 [==============================] - 1s 243ms/step - loss: 0.0396 - accuracy:
0.9922 - val_loss: 0.3000 - val_accuracy: 0.8594
Epoch 10/10
4/4 [==============================] - 1s 189ms/step - loss: 0.0344 - accuracy:
0.9922 - val_loss: 0.3022 - val_accuracy: 0.8594
Epoch 1/10
4/4 [==============================] - 1s 217ms/step - loss: 0.3037 - accuracy:
0.8438 - val_loss: 0.3013 - val_accuracy: 0.8594
Epoch 2/10
4/4 [==============================] - 0s 133ms/step - loss: 0.2044 - accuracy:
0.8906 - val_loss: 0.2996 - val_accuracy: 0.8633
Epoch 3/10
4/4 [==============================] - 1s 169ms/step - loss: 0.1334 - accuracy:
0.9453 - val_loss: 0.2988 - val_accuracy: 0.8711
Epoch 4/10
4/4 [==============================] - 1s 207ms/step - loss: 0.0864 - accuracy:
0.9766 - val_loss: 0.2985 - val_accuracy: 0.8633
Epoch 5/10
4/4 [==============================] - 1s 173ms/step - loss: 0.0642 - accuracy:
0.9844 - val_loss: 0.2987 - val_accuracy: 0.8672
Epoch 6/10
4/4 [==============================] - 1s 256ms/step - loss: 0.0505 - accuracy:
0.9844 - val_loss: 0.2992 - val_accuracy: 0.8672
Epoch 7/10
4/4 [==============================] - 1s 237ms/step - loss: 0.0396 - accuracy:
0.9922 - val_loss: 0.3001 - val_accuracy: 0.8711
Epoch 8/10
4/4 [==============================] - 1s 229ms/step - loss: 0.0323 - accuracy:
0.9922 - val_loss: 0.3013 - val_accuracy: 0.8711
Epoch 9/10
4/4 [==============================] - 1s 206ms/step - loss: 0.0256 - accuracy:
1.0000 - val_loss: 0.3025 - val_accuracy: 0.8711
Epoch 10/10
4/4 [==============================] - 1s 204ms/step - loss: 0.0223 - accuracy:
1.0000 - val_loss: 0.3038 - val_accuracy: 0.8711
Epoch 1/10
4/4 [==============================] - 1s 271ms/step - loss: 0.3343 - accuracy:
0.8672 - val_loss: 0.3037 - val_accuracy: 0.8633
Epoch 2/10
4/4 [==============================] - 1s 198ms/step - loss: 0.2157 - accuracy:
0.9219 - val_loss: 0.3008 - val_accuracy: 0.8711
Epoch 3/10
4/4 [==============================] - 1s 215ms/step - loss: 0.1562 - accuracy:
0.9297 - val_loss: 0.2996 - val_accuracy: 0.8750
Epoch 4/10
4/4 [==============================] - 1s 212ms/step - loss: 0.1046 - accuracy:
```

```
0.9609 - val_loss: 0.2993 - val_accuracy: 0.8750
Epoch 5/10
4/4 [==============================] - 1s 201ms/step - loss: 0.0808 - accuracy:
0.9688 - val_loss: 0.2999 - val_accuracy: 0.8750
Epoch 6/10
4/4 [==============================] - 1s 291ms/step - loss: 0.0538 - accuracy:
0.9844 - val_loss: 0.3006 - val_accuracy: 0.8711
Epoch 7/10
4/4 [==============================] - 1s 246ms/step - loss: 0.0394 - accuracy:
0.9922 - val_loss: 0.3018 - val_accuracy: 0.8711
Epoch 8/10
4/4 [==============================] - 1s 170ms/step - loss: 0.0327 - accuracy:
0.9922 - val_loss: 0.3032 - val_accuracy: 0.8711
Epoch 9/10
4/4 [==============================] - 1s 193ms/step - loss: 0.0285 - accuracy:
0.9922 - val_loss: 0.3045 - val_accuracy: 0.8711
Epoch 10/10
4/4 [==============================] - 1s 211ms/step - loss: 0.0252 - accuracy:
0.9922 - val_loss: 0.3057 - val_accuracy: 0.8711
Epoch 1/10
4/4 [==============================] - 1s 402ms/step - loss: 0.5936 - accuracy:
0.7891 - val_loss: 0.3041 - val_accuracy: 0.8750
Epoch 2/10
4/4 [==============================] - 1s 199ms/step - loss: 0.3116 - accuracy:
0.8906 - val_loss: 0.3480 - val_accuracy: 0.8594
Epoch 3/10
4/4 [==============================] - 1s 253ms/step - loss: 0.1845 - accuracy:
0.9297 - val_loss: 0.4279 - val_accuracy: 0.8359
Epoch 4/10
4/4 [==============================] - 1s 209ms/step - loss: 0.1339 - accuracy:
0.9531 - val_loss: 0.4838 - val_accuracy: 0.8281
Epoch 5/10
4/4 [==============================] - 1s 192ms/step - loss: 0.1001 - accuracy:
0.9609 - val_loss: 0.5060 - val_accuracy: 0.8203
Epoch 6/10
4/4 [==============================] - 1s 183ms/step - loss: 0.0730 - accuracy:
0.9688 - val_loss: 0.5100 - val_accuracy: 0.8164
Epoch 7/10
4/4 [==============================] - 1s 258ms/step - loss: 0.0575 - accuracy:
0.9844 - val_loss: 0.5056 - val_accuracy: 0.8203
Epoch 8/10
4/4 [==============================] - 1s 222ms/step - loss: 0.0446 - accuracy:
1.0000 - val_loss: 0.4998 - val_accuracy: 0.8242
Epoch 9/10
4/4 [==============================] - 1s 229ms/step - loss: 0.0353 - accuracy:
1.0000 - val_loss: 0.4921 - val_accuracy: 0.8242
Epoch 10/10
4/4 [==============================] - 1s 237ms/step - loss: 0.0301 - accuracy:
```

```
1.0000 - val_loss: 0.4867 - val_accuracy: 0.8281
Epoch 1/10
4/4 [==============================] - 1s 262ms/step - loss: 0.4804 - accuracy:
0.8281 - val_loss: 0.3797 - val_accuracy: 0.8438
Epoch 2/10
4/4 [==============================] - 1s 174ms/step - loss: 0.2832 - accuracy:
0.8906 - val_loss: 0.3183 - val_accuracy: 0.8672
Epoch 3/10
4/4 [==============================] - 1s 165ms/step - loss: 0.1638 - accuracy:
0.9297 - val_loss: 0.3153 - val_accuracy: 0.8633
Epoch 4/10
4/4 [==============================] - 0s 141ms/step - loss: 0.1150 - accuracy:
0.9609 - val_loss: 0.3223 - val_accuracy: 0.8633
Epoch 5/10
4/4 [==============================] - 1s 194ms/step - loss: 0.0916 - accuracy:
0.9844 - val_loss: 0.3296 - val_accuracy: 0.8633
Epoch 6/10
4/4 [==============================] - 1s 190ms/step - loss: 0.0686 - accuracy:
0.9844 - val_loss: 0.3357 - val_accuracy: 0.8594
Epoch 7/10
4/4 [==============================] - 1s 192ms/step - loss: 0.0568 - accuracy:
0.9922 - val_loss: 0.3409 - val_accuracy: 0.8555
Epoch 8/10
4/4 [==============================] - 1s 165ms/step - loss: 0.0434 - accuracy:
1.0000 - val_loss: 0.3455 - val_accuracy: 0.8555
Epoch 9/10
4/4 [==============================] - 1s 156ms/step - loss: 0.0362 - accuracy:
1.0000 - val_loss: 0.3497 - val_accuracy: 0.8555
Epoch 10/10
4/4 [==============================] - 1s 243ms/step - loss: 0.0315 - accuracy:
1.0000 - val_loss: 0.3542 - val_accuracy: 0.8516
Epoch 1/10
4/4 [==============================] - 1s 233ms/step - loss: 0.5494 - accuracy:
0.8594 - val_loss: 0.3501 - val_accuracy: 0.8594
Epoch 2/10
4/4 [==============================] - 1s 189ms/step - loss: 0.2821 - accuracy:
0.8984 - val_loss: 0.3408 - val_accuracy: 0.8633
Epoch 3/10
4/4 [==============================] - 1s 252ms/step - loss: 0.1547 - accuracy:
0.9297 - val_loss: 0.3338 - val_accuracy: 0.8516
Epoch 4/10
4/4 [==============================] - 1s 218ms/step - loss: 0.1005 - accuracy:
0.9609 - val_loss: 0.3297 - val_accuracy: 0.8555
Epoch 5/10
4/4 [==============================] - 1s 206ms/step - loss: 0.0721 - accuracy:
0.9688 - val_loss: 0.3274 - val_accuracy: 0.8555
Epoch 6/10
4/4 [==============================] - 1s 158ms/step - loss: 0.0543 - accuracy:
```

```
0.9844 - val_loss: 0.3266 - val_accuracy: 0.8555
Epoch 7/10
4/4 [==============================] - 1s 186ms/step - loss: 0.0400 - accuracy:
0.9922 - val_loss: 0.3267 - val_accuracy: 0.8555
Epoch 8/10
4/4 [==============================] - 1s 156ms/step - loss: 0.0331 - accuracy:
0.9922 - val_loss: 0.3275 - val_accuracy: 0.8516
Epoch 9/10
4/4 [==============================] - 0s 146ms/step - loss: 0.0273 - accuracy:
1.0000 - val_loss: 0.3285 - val_accuracy: 0.8516
Epoch 10/10
4/4 [==============================] - 1s 188ms/step - loss: 0.0233 - accuracy:
1.0000 - val_loss: 0.3294 - val_accuracy: 0.8555
Epoch 1/10
4/4 [==============================] - 1s 306ms/step - loss: 0.2873 - accuracy:
0.8828 - val_loss: 0.3248 - val_accuracy: 0.8516
Epoch 2/10
4/4 [==============================] - 1s 210ms/step - loss: 0.1970 - accuracy:
0.9219 - val_loss: 0.3196 - val_accuracy: 0.8555
Epoch 3/10
4/4 [==============================] - 1s 198ms/step - loss: 0.1316 - accuracy:
0.9531 - val_loss: 0.3186 - val_accuracy: 0.8555
Epoch 4/10
4/4 [==============================] - 1s 218ms/step - loss: 0.0855 - accuracy:
0.9688 - val_loss: 0.3207 - val_accuracy: 0.8594
Epoch 5/10
4/4 [==============================] - 1s 219ms/step - loss: 0.0585 - accuracy:
0.9844 - val_loss: 0.3239 - val_accuracy: 0.8672
Epoch 6/10
4/4 [==============================] - 1s 219ms/step - loss: 0.0453 - accuracy:
0.9844 - val_loss: 0.3276 - val_accuracy: 0.8672
Epoch 7/10
4/4 [==============================] - 1s 191ms/step - loss: 0.0363 - accuracy:
1.0000 - val_loss: 0.3313 - val_accuracy: 0.8633
Epoch 8/10
4/4 [==============================] - 1s 166ms/step - loss: 0.0307 - accuracy:
1.0000 - val_loss: 0.3348 - val_accuracy: 0.8633
Epoch 9/10
4/4 [==============================] - 1s 262ms/step - loss: 0.0279 - accuracy:
1.0000 - val_loss: 0.3381 - val_accuracy: 0.8633
Epoch 10/10
4/4 [==============================] - 1s 215ms/step - loss: 0.0243 - accuracy:
1.0000 - val_loss: 0.3410 - val_accuracy: 0.8594
Epoch 1/10
4/4 [==============================] - 1s 262ms/step - loss: 0.3139 - accuracy:
0.8672 - val_loss: 0.3392 - val_accuracy: 0.8633
Epoch 2/10
4/4 [==============================] - 1s 167ms/step - loss: 0.2185 - accuracy:
```

```
0.9141 - val_loss: 0.3384 - val_accuracy: 0.8711
Epoch 3/10
4/4 [==============================] - 1s 181ms/step - loss: 0.1457 - accuracy:
0.9453 - val_loss: 0.3404 - val_accuracy: 0.8711
Epoch 4/10
4/4 [==============================] - 1s 206ms/step - loss: 0.1029 - accuracy:
0.9609 - val_loss: 0.3433 - val_accuracy: 0.8672
Epoch 5/10
4/4 [==============================] - 1s 217ms/step - loss: 0.0735 - accuracy:
0.9609 - val_loss: 0.3455 - val_accuracy: 0.8594
Epoch 6/10
4/4 [==============================] - 1s 175ms/step - loss: 0.0500 - accuracy:
0.9922 - val_loss: 0.3473 - val_accuracy: 0.8594
Epoch 7/10
4/4 [==============================] - 1s 181ms/step - loss: 0.0380 - accuracy:
0.9922 - val_loss: 0.3491 - val_accuracy: 0.8594
Epoch 8/10
4/4 [==============================] - 1s 178ms/step - loss: 0.0295 - accuracy:
1.0000 - val_loss: 0.3510 - val_accuracy: 0.8594
Epoch 9/10
4/4 [==============================] - 1s 184ms/step - loss: 0.0240 - accuracy:
1.0000 - val_loss: 0.3530 - val_accuracy: 0.8594
Epoch 10/10
4/4 [==============================] - 1s 208ms/step - loss: 0.0208 - accuracy:
1.0000 - val_loss: 0.3550 - val_accuracy: 0.8594
Epoch 1/10
4/4 [==============================] - 1s 235ms/step - loss: 0.4067 - accuracy:
0.7969 - val_loss: 0.3396 - val_accuracy: 0.8633
Epoch 2/10
4/4 [==============================] - 1s 200ms/step - loss: 0.2784 - accuracy:
0.8516 - val_loss: 0.3298 - val_accuracy: 0.8594
Epoch 3/10
4/4 [==============================] - 1s 255ms/step - loss: 0.1877 - accuracy:
0.9062 - val_loss: 0.3305 - val_accuracy: 0.8594
Epoch 4/10
4/4 [==============================] - 1s 200ms/step - loss: 0.1184 - accuracy:
0.9688 - val_loss: 0.3351 - val_accuracy: 0.8516
Epoch 5/10
4/4 [==============================] - 1s 204ms/step - loss: 0.0864 - accuracy:
0.9766 - val_loss: 0.3400 - val_accuracy: 0.8438
Epoch 6/10
4/4 [==============================] - 0s 138ms/step - loss: 0.0644 - accuracy:
0.9766 - val_loss: 0.3438 - val_accuracy: 0.8438
Epoch 7/10
4/4 [==============================] - 1s 210ms/step - loss: 0.0510 - accuracy:
0.9844 - val_loss: 0.3456 - val_accuracy: 0.8398
Epoch 8/10
4/4 [==============================] - 1s 196ms/step - loss: 0.0414 - accuracy:
```

```
0.9922 - val_loss: 0.3474 - val_accuracy: 0.8398
Epoch 9/10
4/4 [==============================] - 1s 184ms/step - loss: 0.0351 - accuracy:
0.9922 - val_loss: 0.3487 - val_accuracy: 0.8359
Epoch 10/10
4/4 [==============================] - 1s 191ms/step - loss: 0.0303 - accuracy:
0.9922 - val_loss: 0.3499 - val_accuracy: 0.8398
Epoch 1/10
4/4 [==============================] - 1s 247ms/step - loss: 0.5729 - accuracy:
0.7969 - val_loss: 0.3263 - val_accuracy: 0.8555
Epoch 2/10
4/4 [==============================] - 1s 192ms/step - loss: 0.3110 - accuracy:
0.8594 - val_loss: 0.3328 - val_accuracy: 0.8672
Epoch 3/10
4/4 [==============================] - 1s 269ms/step - loss: 0.1482 - accuracy:
0.9297 - val_loss: 0.3576 - val_accuracy: 0.8711
Epoch 4/10
4/4 [==============================] - 1s 163ms/step - loss: 0.1036 - accuracy:
0.9688 - val_loss: 0.3757 - val_accuracy: 0.8633
Epoch 5/10
4/4 [==============================] - 1s 181ms/step - loss: 0.0712 - accuracy:
0.9766 - val_loss: 0.3855 - val_accuracy: 0.8594
Epoch 6/10
4/4 [==============================] - 1s 232ms/step - loss: 0.0584 - accuracy:
0.9766 - val_loss: 0.3887 - val_accuracy: 0.8555
Epoch 7/10
4/4 [==============================] - 1s 180ms/step - loss: 0.0478 - accuracy:
0.9922 - val_loss: 0.3909 - val_accuracy: 0.8555
Epoch 8/10
4/4 [==============================] - 1s 171ms/step - loss: 0.0418 - accuracy:
0.9922 - val_loss: 0.3930 - val_accuracy: 0.8555
Epoch 9/10
4/4 [==============================] - 1s 189ms/step - loss: 0.0351 - accuracy:
0.9922 - val_loss: 0.3950 - val_accuracy: 0.8633
Epoch 10/10
4/4 [==============================] - 0s 158ms/step - loss: 0.0301 - accuracy:
1.0000 - val_loss: 0.3971 - val_accuracy: 0.8633
Epoch 1/10
4/4 [==============================] - 1s 223ms/step - loss: 0.4224 - accuracy:
0.8359 - val_loss: 0.3119 - val_accuracy: 0.8750
Epoch 2/10
4/4 [==============================] - 1s 171ms/step - loss: 0.2234 - accuracy:
0.8906 - val_loss: 0.2924 - val_accuracy: 0.8594
Epoch 3/10
4/4 [==============================] - 1s 175ms/step - loss: 0.1624 - accuracy:
0.9375 - val_loss: 0.3003 - val_accuracy: 0.8594
Epoch 4/10
4/4 [==============================] - 0s 143ms/step - loss: 0.1177 - accuracy:
```

```
0.9609 - val_loss: 0.3079 - val_accuracy: 0.8555
Epoch 5/10
4/4 [==============================] - 1s 217ms/step - loss: 0.0813 - accuracy:
0.9688 - val_loss: 0.3117 - val_accuracy: 0.8438
Epoch 6/10
4/4 [==============================] - 1s 154ms/step - loss: 0.0620 - accuracy:
0.9844 - val_loss: 0.3152 - val_accuracy: 0.8398
Epoch 7/10
4/4 [==============================] - 1s 185ms/step - loss: 0.0475 - accuracy:
0.9844 - val_loss: 0.3173 - val_accuracy: 0.8398
Epoch 8/10
4/4 [==============================] - 1s 179ms/step - loss: 0.0384 - accuracy:
0.9922 - val_loss: 0.3189 - val_accuracy: 0.8398
Epoch 9/10
4/4 [==============================] - 1s 199ms/step - loss: 0.0320 - accuracy:
0.9922 - val_loss: 0.3195 - val_accuracy: 0.8438
Epoch 10/10
4/4 [==============================] - 1s 182ms/step - loss: 0.0282 - accuracy:
0.9922 - val_loss: 0.3206 - val_accuracy: 0.8477
Epoch 1/10
4/4 [==============================] - 1s 281ms/step - loss: 0.2905 - accuracy:
0.8750 - val_loss: 0.3019 - val_accuracy: 0.8555
Epoch 2/10
4/4 [==============================] - 1s 189ms/step - loss: 0.1867 - accuracy:
0.9297 - val_loss: 0.2909 - val_accuracy: 0.8711
Epoch 3/10
4/4 [==============================] - 1s 241ms/step - loss: 0.1134 - accuracy:
0.9531 - val_loss: 0.2896 - val_accuracy: 0.8750
Epoch 4/10
4/4 [==============================] - 1s 269ms/step - loss: 0.0835 - accuracy:
0.9609 - val_loss: 0.2926 - val_accuracy: 0.8711
Epoch 5/10
4/4 [==============================] - 1s 191ms/step - loss: 0.0581 - accuracy:
0.9766 - val_loss: 0.2966 - val_accuracy: 0.8711
Epoch 6/10
4/4 [==============================] - 1s 181ms/step - loss: 0.0448 - accuracy:
0.9922 - val_loss: 0.3004 - val_accuracy: 0.8789
Epoch 7/10
4/4 [==============================] - 1s 208ms/step - loss: 0.0376 - accuracy:
0.9922 - val_loss: 0.3035 - val_accuracy: 0.8789
Epoch 8/10
4/4 [==============================] - 1s 171ms/step - loss: 0.0310 - accuracy:
1.0000 - val_loss: 0.3063 - val_accuracy: 0.8828
Epoch 9/10
4/4 [==============================] - 1s 209ms/step - loss: 0.0265 - accuracy:
1.0000 - val_loss: 0.3088 - val_accuracy: 0.8828
Epoch 10/10
4/4 [==============================] - 1s 185ms/step - loss: 0.0239 - accuracy:
```

```
1.0000 - val_loss: 0.3111 - val_accuracy: 0.8828
Epoch 1/10
4/4 [==============================] - 1s 263ms/step - loss: 0.2507 - accuracy:
0.8984 - val_loss: 0.2986 - val_accuracy: 0.8789
Epoch 2/10
4/4 [==============================] - 1s 181ms/step - loss: 0.1498 - accuracy:
0.9297 - val_loss: 0.2999 - val_accuracy: 0.8750
Epoch 3/10
4/4 [==============================] - 1s 195ms/step - loss: 0.0853 - accuracy:
0.9766 - val_loss: 0.3053 - val_accuracy: 0.8828
Epoch 4/10
4/4 [==============================] - 1s 197ms/step - loss: 0.0586 - accuracy:
0.9844 - val_loss: 0.3126 - val_accuracy: 0.8867
Epoch 5/10
4/4 [==============================] - 1s 203ms/step - loss: 0.0431 - accuracy:
0.9844 - val_loss: 0.3203 - val_accuracy: 0.8828
Epoch 6/10
4/4 [==============================] - 1s 167ms/step - loss: 0.0293 - accuracy:
1.0000 - val_loss: 0.3270 - val_accuracy: 0.8867
Epoch 7/10
4/4 [==============================] - 1s 160ms/step - loss: 0.0238 - accuracy:
1.0000 - val_loss: 0.3326 - val_accuracy: 0.8867
Epoch 8/10
4/4 [==============================] - 1s 163ms/step - loss: 0.0193 - accuracy:
1.0000 - val_loss: 0.3373 - val_accuracy: 0.8867
Epoch 9/10
4/4 [==============================] - 1s 181ms/step - loss: 0.0163 - accuracy:
1.0000 - val_loss: 0.3410 - val_accuracy: 0.8867
Epoch 10/10
4/4 [==============================] - 1s 167ms/step - loss: 0.0146 - accuracy:
1.0000 - val_loss: 0.3444 - val_accuracy: 0.8828
Epoch 1/10
4/4 [==============================] - 1s 266ms/step - loss: 0.3528 - accuracy:
0.8594 - val_loss: 0.3347 - val_accuracy: 0.8828
Epoch 2/10
4/4 [==============================] - 1s 190ms/step - loss: 0.2462 - accuracy:
0.8984 - val_loss: 0.3229 - val_accuracy: 0.8828
Epoch 3/10
4/4 [==============================] - 1s 178ms/step - loss: 0.1613 - accuracy:
0.9375 - val_loss: 0.3126 - val_accuracy: 0.8750
Epoch 4/10
4/4 [==============================] - 1s 216ms/step - loss: 0.0990 - accuracy:
0.9531 - val_loss: 0.3029 - val_accuracy: 0.8750
Epoch 5/10
4/4 [==============================] - 1s 166ms/step - loss: 0.0650 - accuracy:
0.9844 - val_loss: 0.2969 - val_accuracy: 0.8711
Epoch 6/10
4/4 [==============================] - 1s 180ms/step - loss: 0.0468 - accuracy:
```

```
0.9922 - val_loss: 0.2940 - val_accuracy: 0.8750
Epoch 7/10
4/4 [==============================] - 1s 199ms/step - loss: 0.0350 - accuracy:
0.9922 - val_loss: 0.2935 - val_accuracy: 0.8789
Epoch 8/10
4/4 [==============================] - 1s 167ms/step - loss: 0.0283 - accuracy:
1.0000 - val_loss: 0.2941 - val_accuracy: 0.8750
Epoch 9/10
4/4 [==============================] - 1s 159ms/step - loss: 0.0235 - accuracy:
1.0000 - val_loss: 0.2953 - val_accuracy: 0.8750
Epoch 10/10
4/4 [==============================] - 0s 141ms/step - loss: 0.0209 - accuracy:
1.0000 - val_loss: 0.2967 - val_accuracy: 0.8750
Epoch 1/10
4/4 [==============================] - 1s 299ms/step - loss: 0.3900 - accuracy:
0.8672 - val_loss: 0.2943 - val_accuracy: 0.8750
Epoch 2/10
4/4 [==============================] - 1s 164ms/step - loss: 0.2563 - accuracy:
0.9062 - val_loss: 0.2901 - val_accuracy: 0.8750
Epoch 3/10
4/4 [==============================] - 1s 152ms/step - loss: 0.1808 - accuracy:
0.9375 - val_loss: 0.2886 - val_accuracy: 0.8750
Epoch 4/10
4/4 [==============================] - 1s 183ms/step - loss: 0.1367 - accuracy:
0.9609 - val_loss: 0.2872 - val_accuracy: 0.8750
Epoch 5/10
4/4 [==============================] - 1s 234ms/step - loss: 0.1058 - accuracy:
0.9609 - val_loss: 0.2859 - val_accuracy: 0.8750
Epoch 6/10
4/4 [==============================] - 1s 200ms/step - loss: 0.0750 - accuracy:
0.9766 - val_loss: 0.2854 - val_accuracy: 0.8711
Epoch 7/10
4/4 [==============================] - 1s 188ms/step - loss: 0.0599 - accuracy:
0.9844 - val_loss: 0.2853 - val_accuracy: 0.8672
Epoch 8/10
4/4 [==============================] - 0s 144ms/step - loss: 0.0495 - accuracy:
0.9844 - val_loss: 0.2861 - val_accuracy: 0.8633
Epoch 9/10
4/4 [==============================] - 1s 164ms/step - loss: 0.0371 - accuracy:
0.9922 - val_loss: 0.2867 - val_accuracy: 0.8633
Epoch 10/10
4/4 [==============================] - 1s 164ms/step - loss: 0.0304 - accuracy:
0.9922 - val_loss: 0.2870 - val_accuracy: 0.8633
Epoch 1/10
4/4 [==============================] - 1s 231ms/step - loss: 0.2299 - accuracy:
0.8906 - val_loss: 0.2798 - val_accuracy: 0.8789
Epoch 2/10
4/4 [==============================] - 1s 191ms/step - loss: 0.1396 - accuracy:
```

```
0.9375 - val_loss: 0.2794 - val_accuracy: 0.8789
Epoch 3/10
4/4 [==============================] - 1s 154ms/step - loss: 0.0888 - accuracy:
0.9531 - val_loss: 0.2922 - val_accuracy: 0.8672
Epoch 4/10
4/4 [==============================] - 1s 157ms/step - loss: 0.0592 - accuracy:
0.9844 - val_loss: 0.3110 - val_accuracy: 0.8789
Epoch 5/10
4/4 [==============================] - 1s 175ms/step - loss: 0.0417 - accuracy:
0.9922 - val_loss: 0.3267 - val_accuracy: 0.8789
Epoch 6/10
4/4 [==============================] - 1s 206ms/step - loss: 0.0322 - accuracy:
0.9922 - val_loss: 0.3389 - val_accuracy: 0.8750
Epoch 7/10
4/4 [==============================] - 1s 162ms/step - loss: 0.0240 - accuracy:
0.9922 - val_loss: 0.3476 - val_accuracy: 0.8711
Epoch 8/10
4/4 [==============================] - 1s 199ms/step - loss: 0.0208 - accuracy:
0.9922 - val_loss: 0.3527 - val_accuracy: 0.8711
Epoch 9/10
4/4 [==============================] - 1s 152ms/step - loss: 0.0158 - accuracy:
1.0000 - val_loss: 0.3560 - val_accuracy: 0.8711
Epoch 10/10
4/4 [==============================] - 1s 167ms/step - loss: 0.0133 - accuracy:
1.0000 - val_loss: 0.3583 - val_accuracy: 0.8711
Epoch 1/10
4/4 [==============================] - 1s 284ms/step - loss: 0.2749 - accuracy:
0.8750 - val_loss: 0.3260 - val_accuracy: 0.8828
Epoch 2/10
4/4 [==============================] - 1s 160ms/step - loss: 0.1505 - accuracy:
0.9453 - val_loss: 0.3001 - val_accuracy: 0.8828
Epoch 3/10
4/4 [==============================] - 1s 169ms/step - loss: 0.0983 - accuracy:
0.9766 - val_loss: 0.2879 - val_accuracy: 0.8867
Epoch 4/10
4/4 [==============================] - 1s 223ms/step - loss: 0.0656 - accuracy:
0.9766 - val_loss: 0.2826 - val_accuracy: 0.8906
Epoch 5/10
4/4 [==============================] - 1s 203ms/step - loss: 0.0480 - accuracy:
0.9844 - val_loss: 0.2808 - val_accuracy: 0.8945
Epoch 6/10
4/4 [==============================] - 1s 202ms/step - loss: 0.0357 - accuracy:
0.9922 - val_loss: 0.2801 - val_accuracy: 0.8984
Epoch 7/10
4/4 [==============================] - 1s 186ms/step - loss: 0.0263 - accuracy:
0.9922 - val_loss: 0.2801 - val_accuracy: 0.8984
Epoch 8/10
4/4 [==============================] - 1s 154ms/step - loss: 0.0203 - accuracy:
```

```
1.0000 - val_loss: 0.2808 - val_accuracy: 0.8945
Epoch 9/10
4/4 [==============================] - 1s 208ms/step - loss: 0.0177 - accuracy:
1.0000 - val_loss: 0.2817 - val_accuracy: 0.8906
Epoch 10/10
4/4 [==============================] - 1s 198ms/step - loss: 0.0153 - accuracy:
1.0000 - val_loss: 0.2829 - val_accuracy: 0.8906
Epoch 1/10
4/4 [==============================] - 1s 250ms/step - loss: 0.2540 - accuracy:
0.8984 - val_loss: 0.2830 - val_accuracy: 0.8945
Epoch 2/10
4/4 [==============================] - 1s 203ms/step - loss: 0.1551 - accuracy:
0.9219 - val_loss: 0.2842 - val_accuracy: 0.8984
Epoch 3/10
4/4 [==============================] - 1s 179ms/step - loss: 0.1037 - accuracy:
0.9766 - val_loss: 0.2869 - val_accuracy: 0.8867
Epoch 4/10
4/4 [==============================] - 1s 182ms/step - loss: 0.0786 - accuracy:
0.9766 - val_loss: 0.2891 - val_accuracy: 0.8828
Epoch 5/10
4/4 [==============================] - 1s 150ms/step - loss: 0.0619 - accuracy:
0.9766 - val_loss: 0.2904 - val_accuracy: 0.8828
Epoch 6/10
4/4 [==============================] - 1s 189ms/step - loss: 0.0482 - accuracy:
0.9844 - val_loss: 0.2920 - val_accuracy: 0.8828
Epoch 7/10
4/4 [==============================] - 1s 192ms/step - loss: 0.0377 - accuracy:
0.9844 - val_loss: 0.2944 - val_accuracy: 0.8828
Epoch 8/10
4/4 [==============================] - 1s 179ms/step - loss: 0.0291 - accuracy:
0.9922 - val_loss: 0.2967 - val_accuracy: 0.8828
Epoch 9/10
4/4 [==============================] - 1s 199ms/step - loss: 0.0248 - accuracy:
0.9922 - val_loss: 0.3005 - val_accuracy: 0.8828
Epoch 10/10
4/4 [==============================] - 1s 171ms/step - loss: 0.0183 - accuracy:
1.0000 - val_loss: 0.3038 - val_accuracy: 0.8828
```

3.28.2 Exportando o modelo Rede Neural

[351]:
```
model_sn.save_weights('pesos/keras_model_sn_val_2.h5')
```

3.28.3 Carregar modelo Rede Neural

[353]:
```
model_sn = create_nn_model()
model_sn.load_weights('pesos/keras_model_sn_val_2.h5')
```

3.29 Avaliando as métricas do modelo Rede Neural count com validação em treino

- Avaliando a acurácia do modelo

[354]:
```
score, acc = model_sn.evaluate(x_test_batch_sn, y_test_batch_sn)
print("A acurácia para o modelo com Redes neurais com o dataset contains com␣
↪validação foi de {:.2f}%".format(acc*100))
```

```
4/4 [==============================] - 0s 51ms/step - loss: 0.2637 - accuracy:
0.8792
A acurácia para o modelo com Redes neurais com o dataset contains com validação
foi de 85.94%
```

- obtendo as predições do modelo utilizando o dataset de treino com **contains**

[355]:
```
y_pred_keras_sn_val = model_sn.predict(x_test_sn.to_numpy()).round()
y_pred_keras_sn_val = [classify(result) for result in y_pred_keras_sn_val]
```

3.30 Classification Report

- A função `classification_report` nos dá todas as métricas importantes que podemos obter de um modelo.
- Ela irá nos dar a precisão por classe.
- O quanto de revoação (recall) o modelo tem por classe e no geral
- O f1-score geral e por classe
- A acurária do modelo
- Todas estas métricas são úteis e nos ajudarão a escolher qual o modelo se saiu melhor e pode ser utilizado para esta tarefa.

[356]:
```
report = classification_report(y_test, y_pred_keras_sn_val,␣
↪target_names=target_names, output_dict=True)
```

[357]:
```
sgd_stats = {'model': 'keras_contains_val',
             'accuracy': report['accuracy'],
             'recall': report['macro avg']['recall'],
             'f1_score': report['macro avg']['f1-score'],
             'support': report['macro avg']['support'],
             'falso_precision': report['Falso']['precision'],
```

```
            'falso_recall': report['Falso']['recall'],
            'falso_f1_score': report['Falso']['f1-score'],
            'verdadeiro_precision': report['Verdadeiro']['precision'],
            'verdadeiro_recall': report['Verdadeiro']['recall'],
            'verdadeiro_f1_score': report['Verdadeiro']['f1-score']}
```

[358]:
```
stats.append(sgd_stats)
```

3.30.1 Formatando resultados

- Vamos adicionar todas as estatícas dentro de um `DataFrame` do pandas.

[359]:
```
stats_df = pd.DataFrame(stats)
```

- Agora para facilitar a visualização dos dados vamos formata-los para que fiquem em valores de porcentagem, multiplicando os mesmos por 100.

[361]:
```
stats_df.accuracy = stats_df.accuracy.map(lambda i:i*100)
stats_df.recall = stats_df.recall.map(lambda i:i*100)
stats_df.f1_score = stats_df.f1_score.map(lambda i:i*100)
stats_df.support = stats_df.support.map(lambda i:i*100)
stats_df.falso_precision = stats_df.falso_precision.map(lambda i:i*100)
stats_df.falso_recall = stats_df.falso_recall.map(lambda i:i*100)
stats_df.falso_f1_score = stats_df.falso_f1_score.map(lambda i:i*100)
stats_df.verdadeiro_precision = stats_df.verdadeiro_precision.map(lambda i:
↪i*100)
stats_df.verdadeiro_recall = stats_df.verdadeiro_recall.map(lambda i:i*100)
stats_df.verdadeiro_f1_score = stats_df.verdadeiro_f1_score.map(lambda i:i*100)
```

- Agora salvamos o dataset formatado, para avaliarmos as estatísticas, sem precisar refazer todo o treinamento e coletar os resultados dos modelos.

[381]:
```
stats_df.to_csv('estatisticas/model_stats_formatted2.csv')
```

4 Comparando resultados dos modelos.

Acurácia: performance geral do modelo. Porcentagem de acertos do modelo dentre todas as amostras.

Precisão: Dentre as predições de uma clase quantas o modelo acertou. Como neste trabalho temos duas classes, teremos a precisão para dados verdadeiros e falsos respecitivamente. Tem um olhar mais voltado para falsos positivos.

Revocação (Recall): Medida de sensibilidade do modelo, dentre todas as situações de fake news quantos por cento foram classificadas corretamente. É uma métrica mais adequada quando estamos olhando mais para os falsos negativos.

F1-Score é a média harmônica entre revocação e precisão: Normalmente quando seu valor é baixo representa uma baixa precisão ou revocação do modelo.

Observação: Caso queira começar deste ponto e apenas avalir os resultados, podemos partir deste ponto

- Podemos carregar as métricas já salvas caso queiramos iniciar a análise a partir daqui.

[382]:
```
stats_df = pd.read_csv('estatisticas/model_stats_formatted2.csv')
```

- `DataFrame` completo com os resultados

[376]:
```
stats_df
```

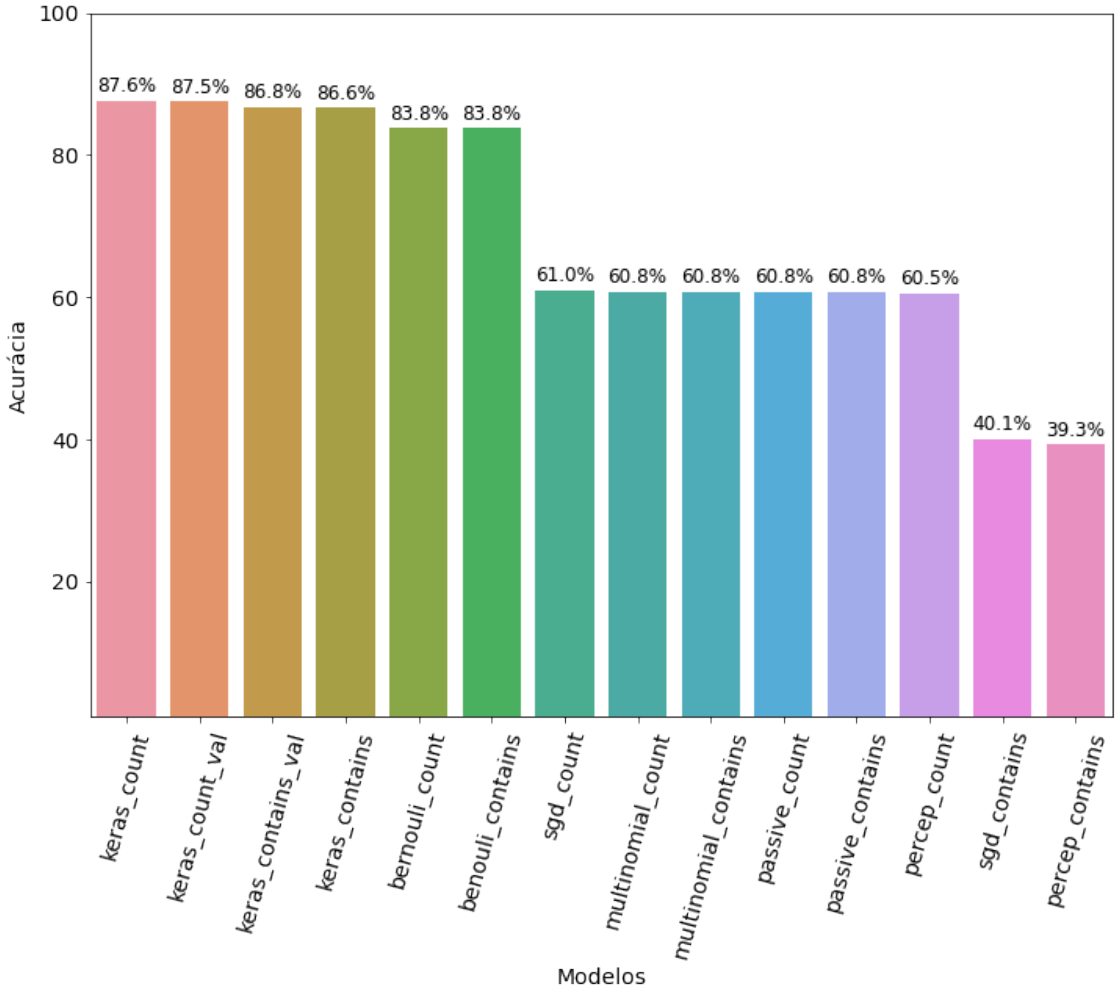
[376]:

	Unnamed: 0	model	accuracy	recall	f1_score	support	falso_precision	falso_recall	falso_f1_score	verdadeiro_precision	verdadeiro_recall	verdadeiro_f1_score
0	0	sgd_count	61.037604	50.222420	38.319360	287200	60.969655	100.000000	75.752979	100.000000	0.444840	0.885740
1	1	sgd_contains	40.076602	49.819519	32.234573	287200	59.183673	4.977117	9.182058	39.045872	94.661922	55.287088
2	2	multinomial_count	60.793872	49.942792	37.808575	287200	60.836237	99.885584	75.617150			
3	3	multinomial_contains	60.793872	49.942792	37.808575	287200	60.836237	99.885584	75.617150			
4	4	percep_count	60.515320	49.936277	38.830310	287200	60.832745	98.627002	75.250982			
5	5	percep_contains	39.345404	48.742437	32.102913	287200	51.612903	5.491991	9.927611			
6	6	bernouli_count	83.774373	79.540420	81.237832	287200	79.403670	99.027460	88.136456			
7	7	benouli_contains	83.774373	79.540420	81.237832	287200	79.403670	99.027460	88.136456			
8	8	passive_count	60.793872	50.800305	41.892643	287200	61.259957	96.796339	75.033259			
9	9	passive_contains	60.759053	50.168266	39.090424	287200	60.944660	98.913043	75.419847			
10	10	keras_count	87.639276	87.034801	87.028920	287200	89.868346	89.816934	89.842632			
11	11	keras_contains	86.629526	85.411289	85.800389	287200	87.513751	91.018307	89.231632			
12	12	keras_count_val	87.500000	86.777466	86.853263	287200	89.437819	90.102975	89.769165			
13	13	keras_contains_val	86.768802	85.970341	86.072877	287200	88.731597	89.645309	89.186113			

2	0.000000	0.000000	0.000000
3	0.000000	0.000000	0.000000
4	36.842105	1.245552	2.409639
5	38.495905	91.992883	54.278215
6	97.543353	60.053381	74.339207
7	97.543353	60.053381	74.339207
8	49.090909	4.804270	8.752026
9	45.714286	1.423488	2.761001
10	84.177778	84.252669	84.215207
11	85.104364	79.804270	82.369146
12	84.428443	83.451957	83.937360
13	83.634720	82.295374	82.959641

- Primeiro vamos dar uma olhada na **Acurácia** dos modelos treinados

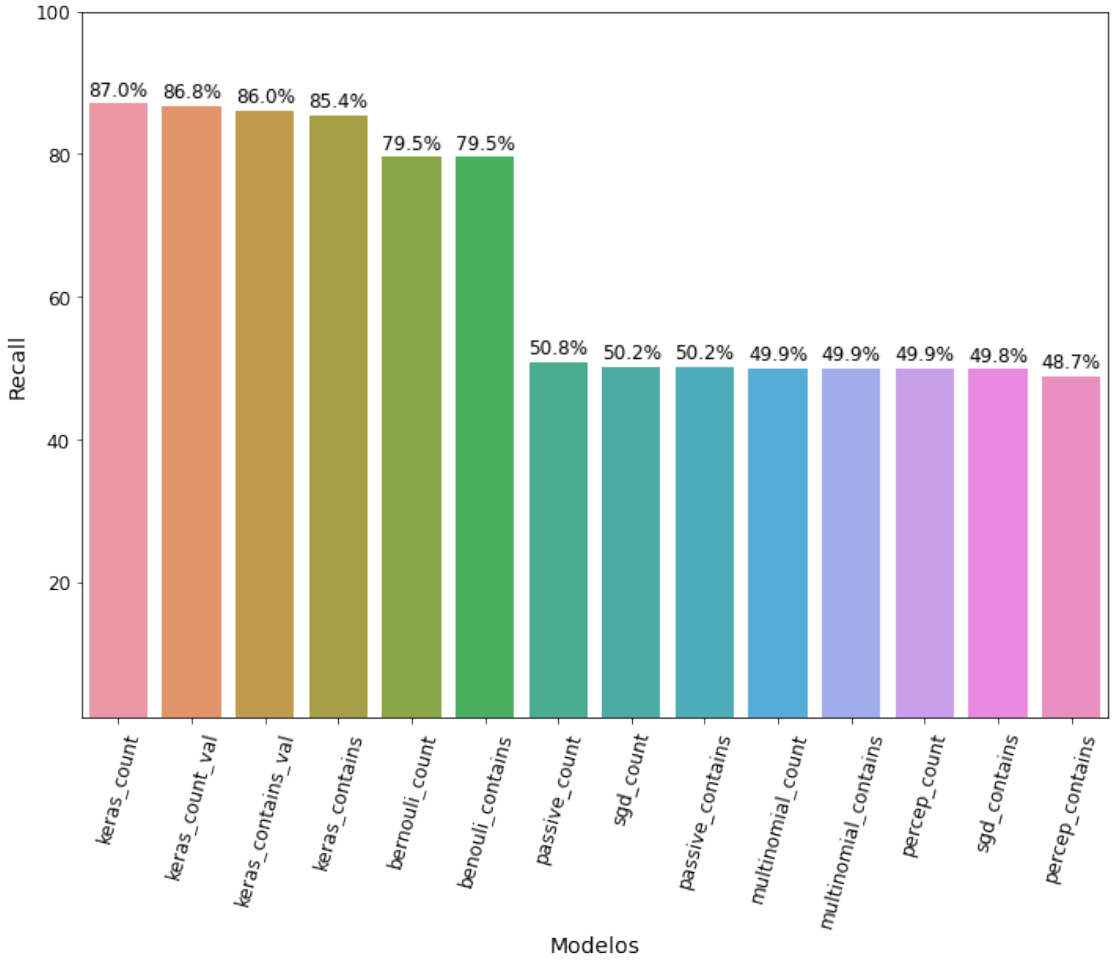
[387]:
```
plt.clf()
plt.xticks(rotation=75)
plt.rcParams.update({'font.size': 12})
g = sns.barplot(x='model', y = 'accuracy', data=stats_df.
 ↪sort_values(['accuracy'], ascending=False))
for p in g.patches:
    g.annotate(str(format(p.get_height(), '.1f')) + '%',
                   (p.get_x() + p.get_width() / 2., p.get_height()),
                   ha = 'center', va = 'center',
                   xytext = (0, 9),
                   textcoords = 'offset points')
plt.xlabel("Modelos", size=14)
plt.ylabel("Acurácia", size=14)
plt.savefig("acurácia modelos.png")
plt.gcf().set_size_inches(11.7, 8.27)
plt.ylim(1,100)
plt.show()
```



- Como podemos observar as redes neurais (deep learning), dominam no quesito acurácia.
- Também podemos observar que o modelo de machine learning `BernoulliNB` foi o que se saiu melhor dentre os modelos do scikit learning, se aproximando bastante da performance dos modelos de deep learning.

[388]:
```
plt.xticks(rotation=75)
g = sns.barplot(x='model', y = 'recall', data=stats_df.sort_values(['recall'],␣
 ↪ascending=False))
for p in g.patches:
    g.annotate(str(format(p.get_height(), '.1f')) + '%',
                   (p.get_x() + p.get_width() / 2., p.get_height()),
                   ha = 'center', va = 'center',
                   xytext = (0, 9),
                   textcoords = 'offset points')
plt.xlabel("Modelos", size=14)
plt.ylabel("Recall", size=14)
```

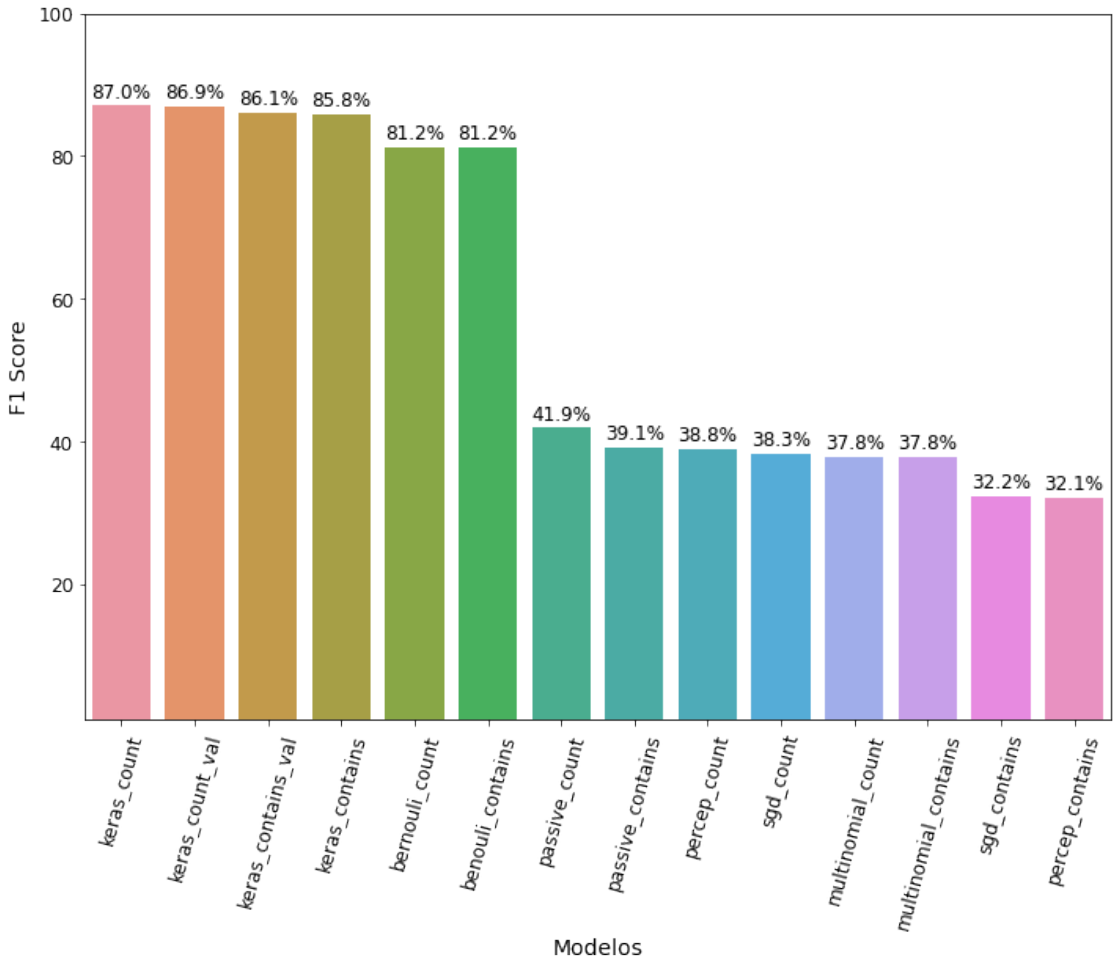
```
plt.savefig("acurácia modelos.png")
plt.gcf().set_size_inches(11.7, 8.27)
plt.ylim(1,100)
plt.show()
```



- Podemos observar que o modelo `keras_count` continua sendo o de melhor performance, seguindo pelas outras redes neurais.
- Vale destacar que devido a característica da métrica recall ela é a melhor opção de avaliação para o caso estudado, visto que é melhor classificar uma notícia verdadeira como falsa e mandá-la para revisão, do que classificar uma falsa como verdadiera e deixá-la passar adiante.

[389]:
```
plt.xticks(rotation=75)
g = sns.barplot(x='model', y = 'f1_score', data=stats_df.
 ↪sort_values(['f1_score'], ascending=False))
for p in g.patches:
    g.annotate(str(format(p.get_height(), '.1f')) + '%',
                    (p.get_x() + p.get_width() / 2., p.get_height()),
```

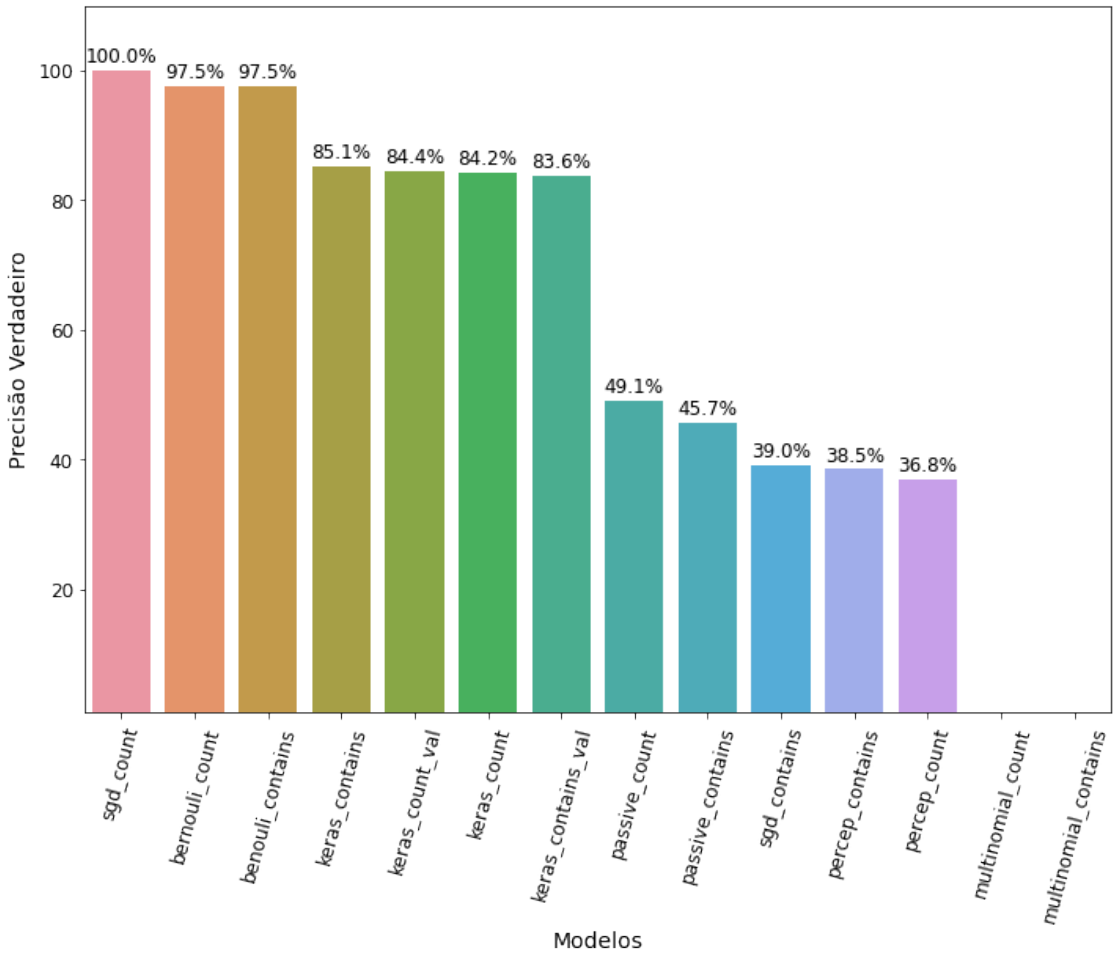
```
                    ha = 'center', va = 'center',
                    xytext = (0, 9),
                    textcoords = 'offset points')
plt.xlabel("Modelos", size=14)
plt.ylabel("F1 Score", size=14)
plt.savefig("acurácia modelos.png")
plt.gcf().set_size_inches(11.7, 8.27)
plt.ylim(1,100)
plt.show()
```



- A rede neural continua `keras_count` continua a frente dos outros modelos, com um bom f1 score representando que a mesma tem um bom nível de precisão e revocação.

[390]:
```
plt.xticks(rotation=75)
g = sns.barplot(x='model', y = 'verdadeiro_precision', data=stats_df.
 ↪sort_values(['verdadeiro_precision'], ascending=False))
for p in g.patches:
```

```
    g.annotate(str(format(p.get_height(), '.1f')) + '%',
                   (p.get_x() + p.get_width() / 2., p.get_height()),
                   ha = 'center', va = 'center',
                   xytext = (0, 9),
                   textcoords = 'offset points')
plt.xlabel("Modelos", size=14)
plt.ylabel("Precisão Verdadeiro", size=14)
plt.savefig("acurácia modelos.png")
plt.gcf().set_size_inches(11.7, 8.27)
plt.ylim(1,110)
plt.show()
```

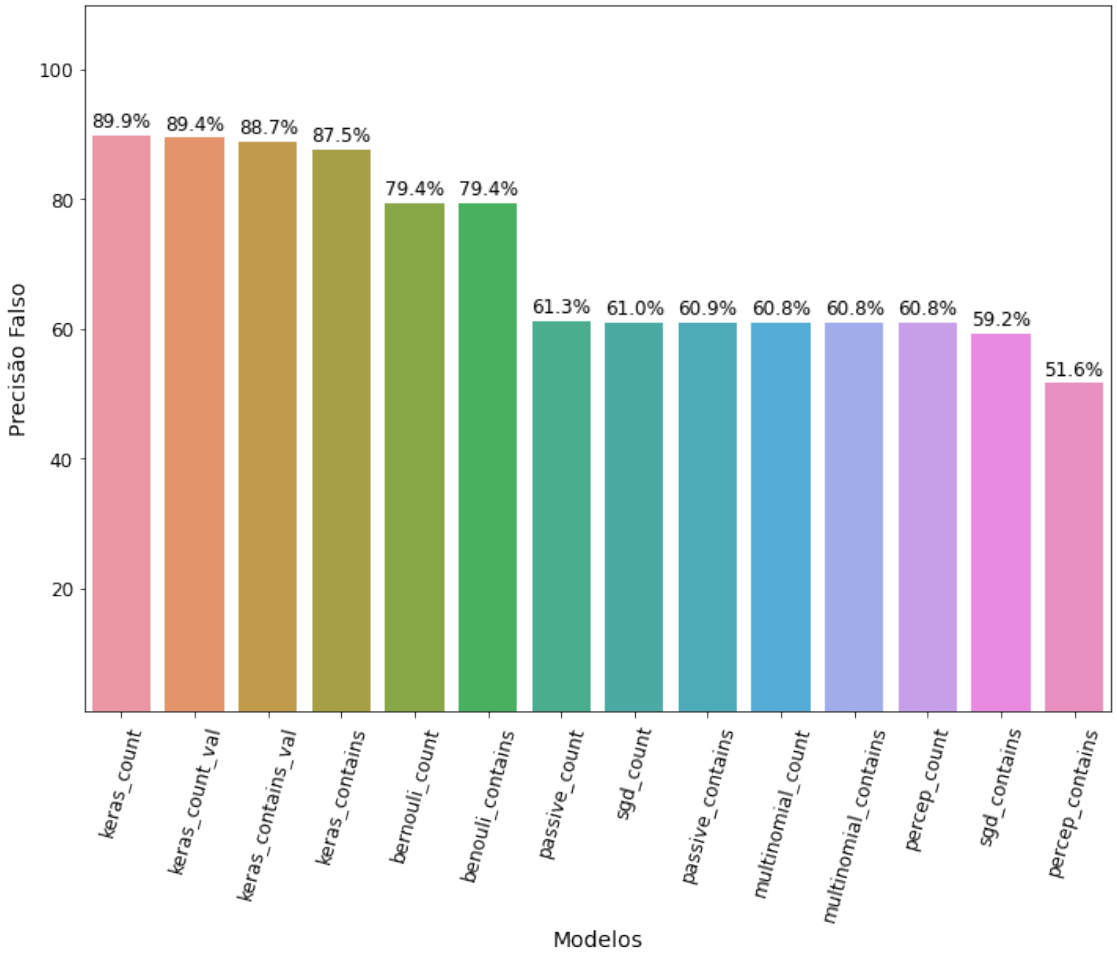


- Agora vemos um comportamento interessante quando se trata de precisão, viemos que a rede neural `keras_count`não aparece em primeiro lugar. Curiosamente o modelo de machine learning `sgd`está com a melhor performance. O grande problema é que o mesmo não esta generalizando bem. O modelo pode estar classificando muitas notícias como verdadeiras, ou seja ele pode ter dado como verdadeiras todas as notícias que eram verdadeiras apresentando

100% de precisão, mas ao mesmo tempo ele pode ter dado como verdadeiras várias notícias falsas.

- Vejamos o comportamento quando observamos a precisção para casos de notícias falsas.

[391]:
```
plt.xticks(rotation=75)
g = sns.barplot(x='model', y = 'falso_precision', data=stats_df.
 ↪sort_values(['falso_precision'], ascending=False))
for p in g.patches:
    g.annotate(str(format(p.get_height(), '.1f')) + '%',
                    (p.get_x() + p.get_width() / 2., p.get_height()),
                    ha = 'center', va = 'center',
                    xytext = (0, 9),
                    textcoords = 'offset points')
plt.xlabel("Modelos", size=14)
plt.ylabel("Precisão Falso", size=14)
plt.savefig("Precisão Falso.png")
plt.gcf().set_size_inches(11.7, 8.27)
plt.ylim(1,110)
plt.show()
```



- Observe que agora o modelo `sgd_count` já não apresenta um bom resultado, ou seja ele realmente está classificando a maioria das notícias como verdadeiras não tendo aprendido muito bem a identificar quando as memas são falsas.
- O modelo `keras_count` também apresentou a melhor performance em classificação de notícias falsas.

5 Conclusão

- Dadas os resultados obtidos podemos concluir que o modelo `keras_count` foi o que apresentou melhor resultado apresentando 87.6% de acurácia, tendo boa generalização, e tendo o melhor nível de revocação (com 87%) e precisção para notícias falsas (89.9%) dando mais segurança no uso do modelo como auxílio no combate às fake news na internet.
- Apesar o modelo `keras_count`ter se saído melhor, o uso dos outros modelos de deep learning, não seriam equivocados, visto que apresentaram poquíssima diferença de performance entre os mesmos.
- É possível utilizar deste modelo, para solicitar revisões de notícias que foram classificadas como falsas antes do compartilhamento da mesma, ou pelo menos com uma sinalização de que o conteúdo pode conter informações inverídicas.

5.0.1 Importante: caso tenha retreinado os modelos deste trabalho, é possível que os modelos de rede neural possam estar em ordem diferente, pois como os mesmos começam com pesos aleatórios, o treinamento pode ser diferente todas as vezes gerando performance diferentes, dando divergências com o resultado desta conclusão. Porém em todos os testes feitos, as performances se aproximavam dos resultados aqui apresentados e as redes neurais sempre ficavam bem próximas em quesito de performance.

[ ]: