lab5

March 24, 2020

[1]:
```
%load_ext autoreload
%autoreload 2
```

1 Problem 1

[2]:
```
import numpy as np

X, Y = np.load('data/lab4.p', allow_pickle=True)
```

As in the previous lab, we convert the data samples to a canonical data structutre better suited for classification.

[3]:
```
from utils import normalize_data

x_train, x_test, y_train, y_test = normalize_data(X, Y)
```

1.1 Parzen classifier

[4]:
```
from utils import error_score, accuracy_score
from models import ParzenClassifier

for kernel in␣
 ↪['gaussian','tophat','epanechnikov','exponential','linear','cosine']:
    for h in [0.1, 5.0]:
        model = ParzenClassifier(h=h, kernel=kernel)
        model.fit(x_train, y_train)
        y_pred_train = model.predict(x_train)
        y_pred_test = model.predict(x_test)
        reclassification_error = error_score(y_pred_train, y_train)
        test_error = error_score(y_pred_test, y_test)

        print(f'[{h=}, {kernel=}] {reclassification_error = :.1%}  {test_error␣
 ↪= :.1%}')
```

```
[h=0.1, kernel='gaussian'] reclassification_error = 0.0%  test_error = 25.0%
[h=5.0, kernel='gaussian'] reclassification_error = 2.0%  test_error = 10.0%
[h=0.1, kernel='tophat'] reclassification_error = 0.0%  test_error = 40.0%
```

```
[h=5.0, kernel='tophat'] reclassification_error = 2.0%  test_error = 20.5%
[h=0.1, kernel='epanechnikov'] reclassification_error = 0.0%  test_error = 40.0%
[h=5.0, kernel='epanechnikov'] reclassification_error = 1.5%  test_error = 20.5%
[h=0.1, kernel='exponential'] reclassification_error = 0.0%  test_error = 11.0%
[h=5.0, kernel='exponential'] reclassification_error = 1.5%  test_error = 10.0%
[h=0.1, kernel='linear'] reclassification_error = 0.0%  test_error = 40.0%
[h=5.0, kernel='linear'] reclassification_error = 1.0%  test_error = 20.5%
[h=0.1, kernel='cosine'] reclassification_error = 0.0%  test_error = 40.0%
[h=5.0, kernel='cosine'] reclassification_error = 1.5%  test_error = 20.5%
```

We see that the best performing kernel functions for this particular classification task are the `gaussian` and the `exponential` ones. For these kernels, the results are similar to what we obtained in the previous lab.

1.2 Quadratic Discriminant Analysis classifier

[5]:
```
from sklearn.discriminant_analysis import QuadraticDiscriminantAnalysis as QDA

model = QDA()
model.fit(x_train, y_train)
y_pred_train = model.predict(x_train)
y_pred_test = model.predict(x_test)
reclassification_error = error_score(y_pred_train, y_train)
test_error = error_score(y_pred_test, y_test)

print(f'{reclassification_error = :.1%}  {test_error = :.1%}')
```

```
reclassification_error = 4.5%  test_error = 10.0%
```

[6]:
```
from sklearn.metrics import confusion_matrix
```

[7]:
```
cm_train = confusion_matrix(y_train, y_pred_train)
cm_test = confusion_matrix(y_test, y_pred_test)

print(f'Reclassification confusion matrix : \n {cm_train} \n')
print(f'Testing confusion matrix : \n {cm_test}')
```

```
Reclassification confusion matrix :
 [[116   4]
 [  5  75]]

Testing confusion matrix :
 [[112   8]
 [ 12  68]]
```

1.3 Nearest Neighbours classifier

[8]:
```
from sklearn.neighbors import KNeighborsClassifier

for kn in range(1, 11):
    model = KNeighborsClassifier(n_neighbors=kn)
    model.fit(x_train, y_train)
    y_pred_train = model.predict(x_train)
    y_pred_test = model.predict(x_test)
    reclassification_error = error_score(y_pred_train, y_train)
    test_error = error_score(y_pred_test, y_test)

    print(f'[{kn = }] {reclassification_error = :.1%}  {test_error = :.1%}')
```

```
[kn = 1] reclassification_error = 0.0%  test_error = 11.0%
[kn = 2] reclassification_error = 4.0%  test_error = 11.5%
[kn = 3] reclassification_error = 3.5%  test_error = 11.0%
[kn = 4] reclassification_error = 3.5%  test_error = 11.5%
[kn = 5] reclassification_error = 3.0%  test_error = 10.0%
[kn = 6] reclassification_error = 3.0%  test_error = 10.5%
[kn = 7] reclassification_error = 4.5%  test_error = 11.0%
[kn = 8] reclassification_error = 4.0%  test_error = 10.5%
[kn = 9] reclassification_error = 4.5%  test_error = 9.5%
[kn = 10] reclassification_error = 5.0%  test_error = 10.5%
```

As was already shown in the previous assignment, we get similar results with our implementation as with sklearn's one. I can't really compare this results with the previous assignment as this is exactly what I have done in the assignment, thinking that we had to implement majority-voting k-nn and not pdf estimate k-nn.

1.4 Cross validation

[9]:
```
from sklearn.model_selection import cross_validate

scores = cross_validate(QDA(), x_train, y_train, cv=5)
mean_accuracy = np.mean(scores["test_score"])
std_score = 2 * np.std(scores["test_score"])

print(f'{mean_accuracy = :.1%} {std_score = :.1%}')
```

```
mean_accuracy = 94.0% std_score = 6.8%
```

[10]:
```
for kn in range(1, 11):
    model = KNeighborsClassifier(n_neighbors=kn)
    scores = cross_validate(model, x_train, y_train, cv=5)
    mean_accuracy = np.mean(scores["test_score"])
    std_score = 2 * np.std(scores["test_score"])
    print(f'[{kn = }] {mean_accuracy = :.1%} {std_score = :.1%}')
```

```
[kn = 1] mean_accuracy = 94.5% std_score = 8.0%
[kn = 2] mean_accuracy = 92.0% std_score = 4.9%
[kn = 3] mean_accuracy = 96.0% std_score = 8.7%
[kn = 4] mean_accuracy = 95.5% std_score = 5.8%
[kn = 5] mean_accuracy = 94.5% std_score = 8.6%
[kn = 6] mean_accuracy = 93.0% std_score = 7.3%
[kn = 7] mean_accuracy = 94.0% std_score = 6.8%
[kn = 8] mean_accuracy = 94.5% std_score = 4.9%
[kn = 9] mean_accuracy = 94.0% std_score = 6.8%
[kn = 10] mean_accuracy = 93.5% std_score = 6.0%
```

There seems to be a tendency for the standard deviation to decrease as the number of neighbours increase.

1.5 Grid search

[11]:
```python
from sklearn.model_selection import GridSearchCV

params = {
    'n_neighbors' : range(1, 11),
    'algorithm' : ('ball_tree', 'kd_tree')
}

model = KNeighborsClassifier()
grid = GridSearchCV(model, params, cv=5)
grid.fit(x_train, y_train)

print(f'Best parameters : {grid.best_params_} \nCross validation score : {grid.
 ↪best_score_:.1%}')

model = KNeighborsClassifier(**grid.best_params_)
model.fit(x_train, y_train)
y_pred = model.predict(x_test)
score = accuracy_score(y_pred, y_test)

print(f'Test score : {score:.1%}')
```

```
Best parameters : {'algorithm': 'ball_tree', 'n_neighbors': 3}
Cross validation score : 96.0%
Test score : 89.0%
```

2 Problem 2

2.1 Using keras

With such a small dataset, neural networks are hard to tune. Since samples only have two features, we observe strong overfitting as soon as we add too much nodes or hidden layers. One solution would be to use dropout, but since we want as few layers as possible it is preferable to use kernel

regularization.

[12]:
```
import tensorflow as tf
import os
import random

# enable reproducibility
SEED = 42
os.environ['TF_DETERMINISTIC_OPS'] = '1'
os.environ['PYTHONHASHSEED'] = str(SEED)
random.seed(SEED)
np.random.seed(SEED)
tf.random.set_seed(SEED)

x_train_norm = tf.keras.utils.normalize(x_train)
x_test_norm = tf.keras.utils.normalize(x_test)

y_train_norm = y_train - 1
y_test_norm = y_test - 1

model = tf.keras.Sequential([
    tf.keras.layers.Dense(4, activation='relu', kernel_regularizer=tf.keras.
 ↪regularizers.l2(l=0.5)),
    tf.keras.layers.Dense(2, activation='softmax')
])

compile_params = {
    'optimizer': tf.keras.optimizers.Adam(learning_rate=1e-3),
    'loss' : 'sparse_categorical_crossentropy',
    'metrics' : ['sparse_categorical_accuracy'],
}

train_params = {
    'validation_data' : (x_test_norm, y_test_norm),
    'epochs' : 200,
    'batch_size': 10,
}

model.compile(**compile_params)
history = model.fit(x_train_norm, y_train_norm, **train_params)
model.evaluate(x_test_norm, y_test_norm)

_, test_acc = model.evaluate(x_test_norm, y_test_norm)
print(f'Test accuracy : {test_acc:.1%}')
```

```
Train on 200 samples, validate on 200 samples
Epoch 1/200
200/200 [==============================] - 0s 2ms/sample - loss: 1.6103 -
```

```
sparse_categorical_accuracy: 0.6000 - val_loss: 1.5673 -
val_sparse_categorical_accuracy: 0.6000
Epoch 2/200
200/200 [==============================] - 0s 204us/sample - loss: 1.5264 -
sparse_categorical_accuracy: 0.6000 - val_loss: 1.4896 -
val_sparse_categorical_accuracy: 0.6000
Epoch 3/200
200/200 [==============================] - 0s 187us/sample - loss: 1.4488 -
sparse_categorical_accuracy: 0.6000 - val_loss: 1.4177 -
val_sparse_categorical_accuracy: 0.6000
Epoch 4/200
200/200 [==============================] - 0s 224us/sample - loss: 1.3775 -
sparse_categorical_accuracy: 0.6000 - val_loss: 1.3531 -
val_sparse_categorical_accuracy: 0.6000
Epoch 5/200
200/200 [==============================] - 0s 244us/sample - loss: 1.3134 -
sparse_categorical_accuracy: 0.6000 - val_loss: 1.2946 -
val_sparse_categorical_accuracy: 0.6000
Epoch 6/200
200/200 [==============================] - 0s 210us/sample - loss: 1.2561 -
sparse_categorical_accuracy: 0.6000 - val_loss: 1.2424 -
val_sparse_categorical_accuracy: 0.6000
Epoch 7/200
200/200 [==============================] - 0s 227us/sample - loss: 1.2037 -
sparse_categorical_accuracy: 0.6000 - val_loss: 1.1952 -
val_sparse_categorical_accuracy: 0.6000
Epoch 8/200
200/200 [==============================] - 0s 250us/sample - loss: 1.1567 -
sparse_categorical_accuracy: 0.6000 - val_loss: 1.1520 -
val_sparse_categorical_accuracy: 0.6000
Epoch 9/200
200/200 [==============================] - 0s 226us/sample - loss: 1.1133 -
sparse_categorical_accuracy: 0.6000 - val_loss: 1.1125 -
val_sparse_categorical_accuracy: 0.6000
Epoch 10/200
200/200 [==============================] - 0s 233us/sample - loss: 1.0743 -
sparse_categorical_accuracy: 0.6000 - val_loss: 1.0763 -
val_sparse_categorical_accuracy: 0.6000
Epoch 11/200
200/200 [==============================] - 0s 229us/sample - loss: 1.0373 -
sparse_categorical_accuracy: 0.6200 - val_loss: 1.0424 -
val_sparse_categorical_accuracy: 0.6400
Epoch 12/200
200/200 [==============================] - 0s 256us/sample - loss: 1.0036 -
sparse_categorical_accuracy: 0.7350 - val_loss: 1.0109 -
val_sparse_categorical_accuracy: 0.6900
Epoch 13/200
200/200 [==============================] - 0s 250us/sample - loss: 0.9718 -
```

```
sparse_categorical_accuracy: 0.7700 - val_loss: 0.9814 -
val_sparse_categorical_accuracy: 0.7050
Epoch 14/200
200/200 [==============================] - 0s 258us/sample - loss: 0.9424 -
sparse_categorical_accuracy: 0.7900 - val_loss: 0.9539 -
val_sparse_categorical_accuracy: 0.7300
Epoch 15/200
200/200 [==============================] - 0s 265us/sample - loss: 0.9149 -
sparse_categorical_accuracy: 0.7900 - val_loss: 0.9286 -
val_sparse_categorical_accuracy: 0.7300
Epoch 16/200
200/200 [==============================] - 0s 274us/sample - loss: 0.8892 -
sparse_categorical_accuracy: 0.7900 - val_loss: 0.9044 -
val_sparse_categorical_accuracy: 0.7400
Epoch 17/200
200/200 [==============================] - 0s 266us/sample - loss: 0.8652 -
sparse_categorical_accuracy: 0.7950 - val_loss: 0.8820 -
val_sparse_categorical_accuracy: 0.7450
Epoch 18/200
200/200 [==============================] - 0s 261us/sample - loss: 0.8429 -
sparse_categorical_accuracy: 0.7950 - val_loss: 0.8614 -
val_sparse_categorical_accuracy: 0.7450
Epoch 19/200
200/200 [==============================] - 0s 262us/sample - loss: 0.8220 -
sparse_categorical_accuracy: 0.8000 - val_loss: 0.8414 -
val_sparse_categorical_accuracy: 0.7550
Epoch 20/200
200/200 [==============================] - 0s 264us/sample - loss: 0.8025 -
sparse_categorical_accuracy: 0.8000 - val_loss: 0.8232 -
val_sparse_categorical_accuracy: 0.7550
Epoch 21/200
200/200 [==============================] - 0s 217us/sample - loss: 0.7837 -
sparse_categorical_accuracy: 0.8000 - val_loss: 0.8058 -
val_sparse_categorical_accuracy: 0.7450
Epoch 22/200
200/200 [==============================] - 0s 198us/sample - loss: 0.7663 -
sparse_categorical_accuracy: 0.8000 - val_loss: 0.7899 -
val_sparse_categorical_accuracy: 0.7450
Epoch 23/200
200/200 [==============================] - 0s 261us/sample - loss: 0.7501 -
sparse_categorical_accuracy: 0.8000 - val_loss: 0.7749 -
val_sparse_categorical_accuracy: 0.7450
Epoch 24/200
200/200 [==============================] - 0s 226us/sample - loss: 0.7345 -
sparse_categorical_accuracy: 0.8000 - val_loss: 0.7604 -
val_sparse_categorical_accuracy: 0.7450
Epoch 25/200
200/200 [==============================] - 0s 176us/sample - loss: 0.7204 -
```

```
sparse_categorical_accuracy: 0.8050 - val_loss: 0.7461 -
val_sparse_categorical_accuracy: 0.7600
Epoch 26/200
200/200 [==============================] - 0s 224us/sample - loss: 0.7065 -
sparse_categorical_accuracy: 0.8050 - val_loss: 0.7338 -
val_sparse_categorical_accuracy: 0.7550
Epoch 27/200
200/200 [==============================] - 0s 251us/sample - loss: 0.6934 -
sparse_categorical_accuracy: 0.8050 - val_loss: 0.7214 -
val_sparse_categorical_accuracy: 0.7600
Epoch 28/200
200/200 [==============================] - 0s 263us/sample - loss: 0.6812 -
sparse_categorical_accuracy: 0.8150 - val_loss: 0.7098 -
val_sparse_categorical_accuracy: 0.7600
Epoch 29/200
200/200 [==============================] - 0s 215us/sample - loss: 0.6693 -
sparse_categorical_accuracy: 0.8250 - val_loss: 0.6988 -
val_sparse_categorical_accuracy: 0.7600
Epoch 30/200
200/200 [==============================] - 0s 246us/sample - loss: 0.6580 -
sparse_categorical_accuracy: 0.8250 - val_loss: 0.6885 -
val_sparse_categorical_accuracy: 0.7600
Epoch 31/200
200/200 [==============================] - 0s 242us/sample - loss: 0.6472 -
sparse_categorical_accuracy: 0.8250 - val_loss: 0.6784 -
val_sparse_categorical_accuracy: 0.7600
Epoch 32/200
200/200 [==============================] - 0s 260us/sample - loss: 0.6366 -
sparse_categorical_accuracy: 0.8250 - val_loss: 0.6685 -
val_sparse_categorical_accuracy: 0.7650
Epoch 33/200
200/200 [==============================] - 0s 246us/sample - loss: 0.6263 -
sparse_categorical_accuracy: 0.8200 - val_loss: 0.6587 -
val_sparse_categorical_accuracy: 0.7750
Epoch 34/200
200/200 [==============================] - 0s 219us/sample - loss: 0.6164 -
sparse_categorical_accuracy: 0.8200 - val_loss: 0.6493 -
val_sparse_categorical_accuracy: 0.7700
Epoch 35/200
200/200 [==============================] - 0s 241us/sample - loss: 0.6062 -
sparse_categorical_accuracy: 0.8300 - val_loss: 0.6401 -
val_sparse_categorical_accuracy: 0.7750
Epoch 36/200
200/200 [==============================] - 0s 250us/sample - loss: 0.5968 -
sparse_categorical_accuracy: 0.8350 - val_loss: 0.6313 -
val_sparse_categorical_accuracy: 0.7750
Epoch 37/200
200/200 [==============================] - 0s 286us/sample - loss: 0.5862 -
```

```
sparse_categorical_accuracy: 0.8350 - val_loss: 0.6212 -
val_sparse_categorical_accuracy: 0.8100
Epoch 38/200
200/200 [==============================] - 0s 225us/sample - loss: 0.5767 -
sparse_categorical_accuracy: 0.8550 - val_loss: 0.6122 -
val_sparse_categorical_accuracy: 0.8250
Epoch 39/200
200/200 [==============================] - 0s 214us/sample - loss: 0.5677 -
sparse_categorical_accuracy: 0.8550 - val_loss: 0.6052 -
val_sparse_categorical_accuracy: 0.8100
Epoch 40/200
200/200 [==============================] - 0s 253us/sample - loss: 0.5589 -
sparse_categorical_accuracy: 0.8650 - val_loss: 0.5972 -
val_sparse_categorical_accuracy: 0.8300
Epoch 41/200
200/200 [==============================] - 0s 262us/sample - loss: 0.5515 -
sparse_categorical_accuracy: 0.8700 - val_loss: 0.5906 -
val_sparse_categorical_accuracy: 0.8300
Epoch 42/200
200/200 [==============================] - 0s 265us/sample - loss: 0.5444 -
sparse_categorical_accuracy: 0.8700 - val_loss: 0.5852 -
val_sparse_categorical_accuracy: 0.8300
Epoch 43/200
200/200 [==============================] - 0s 256us/sample - loss: 0.5383 -
sparse_categorical_accuracy: 0.8700 - val_loss: 0.5800 -
val_sparse_categorical_accuracy: 0.8300
Epoch 44/200
200/200 [==============================] - 0s 238us/sample - loss: 0.5324 -
sparse_categorical_accuracy: 0.8700 - val_loss: 0.5747 -
val_sparse_categorical_accuracy: 0.8350
Epoch 45/200
200/200 [==============================] - 0s 262us/sample - loss: 0.5275 -
sparse_categorical_accuracy: 0.8750 - val_loss: 0.5697 -
val_sparse_categorical_accuracy: 0.8350
Epoch 46/200
200/200 [==============================] - 0s 291us/sample - loss: 0.5219 -
sparse_categorical_accuracy: 0.8750 - val_loss: 0.5656 -
val_sparse_categorical_accuracy: 0.8350
Epoch 47/200
200/200 [==============================] - 0s 237us/sample - loss: 0.5175 -
sparse_categorical_accuracy: 0.8700 - val_loss: 0.5622 -
val_sparse_categorical_accuracy: 0.8350
Epoch 48/200
200/200 [==============================] - 0s 217us/sample - loss: 0.5132 -
sparse_categorical_accuracy: 0.8700 - val_loss: 0.5583 -
val_sparse_categorical_accuracy: 0.8350
Epoch 49/200
200/200 [==============================] - 0s 262us/sample - loss: 0.5084 -
```

```
sparse_categorical_accuracy: 0.8800 - val_loss: 0.5540 -
val_sparse_categorical_accuracy: 0.8400
Epoch 50/200
200/200 [==============================] - 0s 215us/sample - loss: 0.5047 -
sparse_categorical_accuracy: 0.9000 - val_loss: 0.5506 -
val_sparse_categorical_accuracy: 0.8400
Epoch 51/200
200/200 [==============================] - 0s 203us/sample - loss: 0.5006 -
sparse_categorical_accuracy: 0.9000 - val_loss: 0.5470 -
val_sparse_categorical_accuracy: 0.8450
Epoch 52/200
200/200 [==============================] - 0s 251us/sample - loss: 0.4970 -
sparse_categorical_accuracy: 0.9000 - val_loss: 0.5447 -
val_sparse_categorical_accuracy: 0.8400
Epoch 53/200
200/200 [==============================] - 0s 268us/sample - loss: 0.4930 -
sparse_categorical_accuracy: 0.9000 - val_loss: 0.5412 -
val_sparse_categorical_accuracy: 0.8450
Epoch 54/200
200/200 [==============================] - 0s 289us/sample - loss: 0.4907 -
sparse_categorical_accuracy: 0.9000 - val_loss: 0.5376 -
val_sparse_categorical_accuracy: 0.8600
Epoch 55/200
200/200 [==============================] - 0s 268us/sample - loss: 0.4864 -
sparse_categorical_accuracy: 0.9050 - val_loss: 0.5355 -
val_sparse_categorical_accuracy: 0.8500
Epoch 56/200
200/200 [==============================] - 0s 192us/sample - loss: 0.4831 -
sparse_categorical_accuracy: 0.9050 - val_loss: 0.5329 -
val_sparse_categorical_accuracy: 0.8550
Epoch 57/200
200/200 [==============================] - 0s 265us/sample - loss: 0.4807 -
sparse_categorical_accuracy: 0.9000 - val_loss: 0.5310 -
val_sparse_categorical_accuracy: 0.8450
Epoch 58/200
200/200 [==============================] - 0s 271us/sample - loss: 0.4777 -
sparse_categorical_accuracy: 0.9050 - val_loss: 0.5279 -
val_sparse_categorical_accuracy: 0.8600
Epoch 59/200
200/200 [==============================] - 0s 243us/sample - loss: 0.4745 -
sparse_categorical_accuracy: 0.9100 - val_loss: 0.5256 -
val_sparse_categorical_accuracy: 0.8650
Epoch 60/200
200/200 [==============================] - 0s 259us/sample - loss: 0.4720 -
sparse_categorical_accuracy: 0.9150 - val_loss: 0.5236 -
val_sparse_categorical_accuracy: 0.8600
Epoch 61/200
200/200 [==============================] - 0s 318us/sample - loss: 0.4690 -
```

```
sparse_categorical_accuracy: 0.9100 - val_loss: 0.5212 -
val_sparse_categorical_accuracy: 0.8650
Epoch 62/200
200/200 [==============================] - 0s 292us/sample - loss: 0.4664 -
sparse_categorical_accuracy: 0.9150 - val_loss: 0.5191 -
val_sparse_categorical_accuracy: 0.8650
Epoch 63/200
200/200 [==============================] - 0s 278us/sample - loss: 0.4639 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.5173 -
val_sparse_categorical_accuracy: 0.8650
Epoch 64/200
200/200 [==============================] - 0s 265us/sample - loss: 0.4615 -
sparse_categorical_accuracy: 0.9150 - val_loss: 0.5150 -
val_sparse_categorical_accuracy: 0.8700
Epoch 65/200
200/200 [==============================] - 0s 262us/sample - loss: 0.4592 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.5133 -
val_sparse_categorical_accuracy: 0.8650
Epoch 66/200
200/200 [==============================] - 0s 257us/sample - loss: 0.4571 -
sparse_categorical_accuracy: 0.9150 - val_loss: 0.5119 -
val_sparse_categorical_accuracy: 0.8650
Epoch 67/200
200/200 [==============================] - 0s 224us/sample - loss: 0.4549 -
sparse_categorical_accuracy: 0.9150 - val_loss: 0.5097 -
val_sparse_categorical_accuracy: 0.8700
Epoch 68/200
200/200 [==============================] - 0s 305us/sample - loss: 0.4524 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.5080 -
val_sparse_categorical_accuracy: 0.8700
Epoch 69/200
200/200 [==============================] - 0s 236us/sample - loss: 0.4505 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.5061 -
val_sparse_categorical_accuracy: 0.8700
Epoch 70/200
200/200 [==============================] - 0s 260us/sample - loss: 0.4482 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.5044 -
val_sparse_categorical_accuracy: 0.8650
Epoch 71/200
200/200 [==============================] - 0s 236us/sample - loss: 0.4464 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.5033 -
val_sparse_categorical_accuracy: 0.8700
Epoch 72/200
200/200 [==============================] - 0s 227us/sample - loss: 0.4449 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.5013 -
val_sparse_categorical_accuracy: 0.8700
Epoch 73/200
200/200 [==============================] - 0s 282us/sample - loss: 0.4424 -
```

```
sparse_categorical_accuracy: 0.9200 - val_loss: 0.5006 -
val_sparse_categorical_accuracy: 0.8650
Epoch 74/200
200/200 [==============================] - 0s 327us/sample - loss: 0.4404 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4989 -
val_sparse_categorical_accuracy: 0.8650
Epoch 75/200
200/200 [==============================] - 0s 246us/sample - loss: 0.4384 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4971 -
val_sparse_categorical_accuracy: 0.8650
Epoch 76/200
200/200 [==============================] - 0s 183us/sample - loss: 0.4365 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4959 -
val_sparse_categorical_accuracy: 0.8650
Epoch 77/200
200/200 [==============================] - 0s 205us/sample - loss: 0.4348 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4942 -
val_sparse_categorical_accuracy: 0.8800
Epoch 78/200
200/200 [==============================] - 0s 267us/sample - loss: 0.4332 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4925 -
val_sparse_categorical_accuracy: 0.8850
Epoch 79/200
200/200 [==============================] - 0s 240us/sample - loss: 0.4314 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4918 -
val_sparse_categorical_accuracy: 0.8650
Epoch 80/200
200/200 [==============================] - 0s 251us/sample - loss: 0.4297 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4908 -
val_sparse_categorical_accuracy: 0.8650
Epoch 81/200
200/200 [==============================] - 0s 243us/sample - loss: 0.4285 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4892 -
val_sparse_categorical_accuracy: 0.8800
Epoch 82/200
200/200 [==============================] - 0s 282us/sample - loss: 0.4266 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4884 -
val_sparse_categorical_accuracy: 0.8650
Epoch 83/200
200/200 [==============================] - 0s 243us/sample - loss: 0.4247 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4861 -
val_sparse_categorical_accuracy: 0.8850
Epoch 84/200
200/200 [==============================] - 0s 226us/sample - loss: 0.4230 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4852 -
val_sparse_categorical_accuracy: 0.8850
Epoch 85/200
200/200 [==============================] - 0s 236us/sample - loss: 0.4214 -
```

```
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4842 -
val_sparse_categorical_accuracy: 0.8850
Epoch 86/200
200/200 [==============================] - 0s 250us/sample - loss: 0.4200 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4825 -
val_sparse_categorical_accuracy: 0.8850
Epoch 87/200
200/200 [==============================] - 0s 270us/sample - loss: 0.4190 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4818 -
val_sparse_categorical_accuracy: 0.8850
Epoch 88/200
200/200 [==============================] - 0s 240us/sample - loss: 0.4178 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4804 -
val_sparse_categorical_accuracy: 0.8850
Epoch 89/200
200/200 [==============================] - 0s 257us/sample - loss: 0.4155 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4794 -
val_sparse_categorical_accuracy: 0.8850
Epoch 90/200
200/200 [==============================] - 0s 275us/sample - loss: 0.4139 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4786 -
val_sparse_categorical_accuracy: 0.8850
Epoch 91/200
200/200 [==============================] - 0s 283us/sample - loss: 0.4124 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4776 -
val_sparse_categorical_accuracy: 0.8850
Epoch 92/200
200/200 [==============================] - 0s 224us/sample - loss: 0.4117 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4759 -
val_sparse_categorical_accuracy: 0.8900
Epoch 93/200
200/200 [==============================] - 0s 243us/sample - loss: 0.4098 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4758 -
val_sparse_categorical_accuracy: 0.8850
Epoch 94/200
200/200 [==============================] - 0s 203us/sample - loss: 0.4083 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4744 -
val_sparse_categorical_accuracy: 0.8850
Epoch 95/200
200/200 [==============================] - 0s 253us/sample - loss: 0.4071 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4737 -
val_sparse_categorical_accuracy: 0.8850
Epoch 96/200
200/200 [==============================] - 0s 277us/sample - loss: 0.4059 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4727 -
val_sparse_categorical_accuracy: 0.8850
Epoch 97/200
200/200 [==============================] - 0s 299us/sample - loss: 0.4046 -
```

```
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4709 -
val_sparse_categorical_accuracy: 0.8950
Epoch 98/200
200/200 [==============================] - 0s 268us/sample - loss: 0.4031 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4701 -
val_sparse_categorical_accuracy: 0.8900
Epoch 99/200
200/200 [==============================] - 0s 220us/sample - loss: 0.4022 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4702 -
val_sparse_categorical_accuracy: 0.8850
Epoch 100/200
200/200 [==============================] - 0s 273us/sample - loss: 0.4004 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4679 -
val_sparse_categorical_accuracy: 0.8950
Epoch 101/200
200/200 [==============================] - 0s 299us/sample - loss: 0.3995 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4679 -
val_sparse_categorical_accuracy: 0.8900
Epoch 102/200
200/200 [==============================] - 0s 250us/sample - loss: 0.4007 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4658 -
val_sparse_categorical_accuracy: 0.8950
Epoch 103/200
200/200 [==============================] - 0s 224us/sample - loss: 0.3984 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4660 -
val_sparse_categorical_accuracy: 0.8900
Epoch 104/200
200/200 [==============================] - 0s 198us/sample - loss: 0.3957 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4651 -
val_sparse_categorical_accuracy: 0.8900
Epoch 105/200
200/200 [==============================] - 0s 225us/sample - loss: 0.3945 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4641 -
val_sparse_categorical_accuracy: 0.8900
Epoch 106/200
200/200 [==============================] - 0s 234us/sample - loss: 0.3934 -
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4634 -
val_sparse_categorical_accuracy: 0.8900
Epoch 107/200
200/200 [==============================] - 0s 184us/sample - loss: 0.3921 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4617 -
val_sparse_categorical_accuracy: 0.8950
Epoch 108/200
200/200 [==============================] - 0s 249us/sample - loss: 0.3913 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4611 -
val_sparse_categorical_accuracy: 0.8950
Epoch 109/200
200/200 [==============================] - 0s 241us/sample - loss: 0.3898 -
```

```
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4607 -
val_sparse_categorical_accuracy: 0.8950
Epoch 110/200
200/200 [==============================] - 0s 266us/sample - loss: 0.3886 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4596 -
val_sparse_categorical_accuracy: 0.8950
Epoch 111/200
200/200 [==============================] - 0s 271us/sample - loss: 0.3876 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4584 -
val_sparse_categorical_accuracy: 0.8950
Epoch 112/200
200/200 [==============================] - 0s 256us/sample - loss: 0.3866 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4576 -
val_sparse_categorical_accuracy: 0.8950
Epoch 113/200
200/200 [==============================] - 0s 189us/sample - loss: 0.3854 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4572 -
val_sparse_categorical_accuracy: 0.8950
Epoch 114/200
200/200 [==============================] - 0s 214us/sample - loss: 0.3848 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4561 -
val_sparse_categorical_accuracy: 0.8950
Epoch 115/200
200/200 [==============================] - 0s 264us/sample - loss: 0.3833 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4561 -
val_sparse_categorical_accuracy: 0.8950
Epoch 116/200
200/200 [==============================] - 0s 272us/sample - loss: 0.3825 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4549 -
val_sparse_categorical_accuracy: 0.8950
Epoch 117/200
200/200 [==============================] - 0s 260us/sample - loss: 0.3812 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4541 -
val_sparse_categorical_accuracy: 0.8950
Epoch 118/200
200/200 [==============================] - 0s 247us/sample - loss: 0.3805 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4529 -
val_sparse_categorical_accuracy: 0.8950
Epoch 119/200
200/200 [==============================] - 0s 266us/sample - loss: 0.3795 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4525 -
val_sparse_categorical_accuracy: 0.8950
Epoch 120/200
200/200 [==============================] - 0s 228us/sample - loss: 0.3781 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4516 -
val_sparse_categorical_accuracy: 0.8950
Epoch 121/200
200/200 [==============================] - 0s 267us/sample - loss: 0.3776 -
```

```
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4517 -
val_sparse_categorical_accuracy: 0.8950
Epoch 122/200
200/200 [==============================] - 0s 257us/sample - loss: 0.3764 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4499 -
val_sparse_categorical_accuracy: 0.8950
Epoch 123/200
200/200 [==============================] - 0s 251us/sample - loss: 0.3752 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4494 -
val_sparse_categorical_accuracy: 0.8950
Epoch 124/200
200/200 [==============================] - 0s 262us/sample - loss: 0.3746 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4489 -
val_sparse_categorical_accuracy: 0.8950
Epoch 125/200
200/200 [==============================] - 0s 264us/sample - loss: 0.3737 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4482 -
val_sparse_categorical_accuracy: 0.8950
Epoch 126/200
200/200 [==============================] - 0s 276us/sample - loss: 0.3727 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4475 -
val_sparse_categorical_accuracy: 0.8950
Epoch 127/200
200/200 [==============================] - 0s 273us/sample - loss: 0.3714 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4473 -
val_sparse_categorical_accuracy: 0.8950
Epoch 128/200
200/200 [==============================] - 0s 290us/sample - loss: 0.3710 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4461 -
val_sparse_categorical_accuracy: 0.8950
Epoch 129/200
200/200 [==============================] - 0s 250us/sample - loss: 0.3696 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4461 -
val_sparse_categorical_accuracy: 0.8950
Epoch 130/200
200/200 [==============================] - 0s 260us/sample - loss: 0.3692 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4447 -
val_sparse_categorical_accuracy: 0.8950
Epoch 131/200
200/200 [==============================] - 0s 268us/sample - loss: 0.3678 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4448 -
val_sparse_categorical_accuracy: 0.8950
Epoch 132/200
200/200 [==============================] - 0s 289us/sample - loss: 0.3670 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4438 -
val_sparse_categorical_accuracy: 0.8950
Epoch 133/200
200/200 [==============================] - 0s 260us/sample - loss: 0.3669 -
```

```
sparse_categorical_accuracy: 0.9200 - val_loss: 0.4437 -
val_sparse_categorical_accuracy: 0.8950
Epoch 134/200
200/200 [==============================] - 0s 256us/sample - loss: 0.3657 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4416 -
val_sparse_categorical_accuracy: 0.8950
Epoch 135/200
200/200 [==============================] - 0s 240us/sample - loss: 0.3643 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4409 -
val_sparse_categorical_accuracy: 0.8950
Epoch 136/200
200/200 [==============================] - 0s 321us/sample - loss: 0.3635 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4410 -
val_sparse_categorical_accuracy: 0.8950
Epoch 137/200
200/200 [==============================] - 0s 283us/sample - loss: 0.3631 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4412 -
val_sparse_categorical_accuracy: 0.8950
Epoch 138/200
200/200 [==============================] - 0s 231us/sample - loss: 0.3619 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4395 -
val_sparse_categorical_accuracy: 0.8950
Epoch 139/200
200/200 [==============================] - 0s 225us/sample - loss: 0.3609 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4388 -
val_sparse_categorical_accuracy: 0.8950
Epoch 140/200
200/200 [==============================] - 0s 262us/sample - loss: 0.3602 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4379 -
val_sparse_categorical_accuracy: 0.8950
Epoch 141/200
200/200 [==============================] - 0s 218us/sample - loss: 0.3594 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4379 -
val_sparse_categorical_accuracy: 0.8950
Epoch 142/200
200/200 [==============================] - 0s 178us/sample - loss: 0.3589 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4369 -
val_sparse_categorical_accuracy: 0.8950
Epoch 143/200
200/200 [==============================] - 0s 232us/sample - loss: 0.3579 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4362 -
val_sparse_categorical_accuracy: 0.8950
Epoch 144/200
200/200 [==============================] - 0s 240us/sample - loss: 0.3581 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4379 -
val_sparse_categorical_accuracy: 0.8950
Epoch 145/200
200/200 [==============================] - 0s 275us/sample - loss: 0.3565 -
```

```
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4351 -
val_sparse_categorical_accuracy: 0.8950
Epoch 146/200
200/200 [==============================] - 0s 229us/sample - loss: 0.3556 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4344 -
val_sparse_categorical_accuracy: 0.8950
Epoch 147/200
200/200 [==============================] - 0s 219us/sample - loss: 0.3557 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4357 -
val_sparse_categorical_accuracy: 0.8950
Epoch 148/200
200/200 [==============================] - 0s 262us/sample - loss: 0.3537 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4341 -
val_sparse_categorical_accuracy: 0.8950
Epoch 149/200
200/200 [==============================] - 0s 274us/sample - loss: 0.3534 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4324 -
val_sparse_categorical_accuracy: 0.8900
Epoch 150/200
200/200 [==============================] - 0s 267us/sample - loss: 0.3523 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4325 -
val_sparse_categorical_accuracy: 0.8950
Epoch 151/200
200/200 [==============================] - 0s 254us/sample - loss: 0.3538 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4338 -
val_sparse_categorical_accuracy: 0.8950
Epoch 152/200
200/200 [==============================] - 0s 201us/sample - loss: 0.3520 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4310 -
val_sparse_categorical_accuracy: 0.8900
Epoch 153/200
200/200 [==============================] - 0s 235us/sample - loss: 0.3502 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4310 -
val_sparse_categorical_accuracy: 0.8900
Epoch 154/200
200/200 [==============================] - 0s 256us/sample - loss: 0.3494 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4308 -
val_sparse_categorical_accuracy: 0.8950
Epoch 155/200
200/200 [==============================] - 0s 258us/sample - loss: 0.3491 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4308 -
val_sparse_categorical_accuracy: 0.8950
Epoch 156/200
200/200 [==============================] - 0s 256us/sample - loss: 0.3483 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4303 -
val_sparse_categorical_accuracy: 0.8950
Epoch 157/200
200/200 [==============================] - 0s 226us/sample - loss: 0.3476 -
```

```
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4288 -
val_sparse_categorical_accuracy: 0.8950
Epoch 158/200
200/200 [==============================] - 0s 182us/sample - loss: 0.3469 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4281 -
val_sparse_categorical_accuracy: 0.8900
Epoch 159/200
200/200 [==============================] - 0s 217us/sample - loss: 0.3464 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4290 -
val_sparse_categorical_accuracy: 0.8950
Epoch 160/200
200/200 [==============================] - 0s 216us/sample - loss: 0.3455 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4277 -
val_sparse_categorical_accuracy: 0.8950
Epoch 161/200
200/200 [==============================] - 0s 225us/sample - loss: 0.3452 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4282 -
val_sparse_categorical_accuracy: 0.8950
Epoch 162/200
200/200 [==============================] - 0s 259us/sample - loss: 0.3444 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4265 -
val_sparse_categorical_accuracy: 0.8900
Epoch 163/200
200/200 [==============================] - 0s 260us/sample - loss: 0.3445 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4269 -
val_sparse_categorical_accuracy: 0.8950
Epoch 164/200
200/200 [==============================] - 0s 281us/sample - loss: 0.3433 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4251 -
val_sparse_categorical_accuracy: 0.8900
Epoch 165/200
200/200 [==============================] - 0s 269us/sample - loss: 0.3420 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4259 -
val_sparse_categorical_accuracy: 0.8950
Epoch 166/200
200/200 [==============================] - 0s 251us/sample - loss: 0.3417 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4241 -
val_sparse_categorical_accuracy: 0.8900
Epoch 167/200
200/200 [==============================] - 0s 255us/sample - loss: 0.3410 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4251 -
val_sparse_categorical_accuracy: 0.8950
Epoch 168/200
200/200 [==============================] - 0s 222us/sample - loss: 0.3403 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4241 -
val_sparse_categorical_accuracy: 0.8950
Epoch 169/200
200/200 [==============================] - 0s 185us/sample - loss: 0.3397 -
```

```
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4236 -
val_sparse_categorical_accuracy: 0.8900
Epoch 170/200
200/200 [==============================] - 0s 257us/sample - loss: 0.3388 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4235 -
val_sparse_categorical_accuracy: 0.8950
Epoch 171/200
200/200 [==============================] - 0s 260us/sample - loss: 0.3382 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4234 -
val_sparse_categorical_accuracy: 0.8950
Epoch 172/200
200/200 [==============================] - 0s 259us/sample - loss: 0.3381 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4225 -
val_sparse_categorical_accuracy: 0.8950
Epoch 173/200
200/200 [==============================] - 0s 257us/sample - loss: 0.3382 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4207 -
val_sparse_categorical_accuracy: 0.8850
Epoch 174/200
200/200 [==============================] - 0s 275us/sample - loss: 0.3364 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4222 -
val_sparse_categorical_accuracy: 0.8950
Epoch 175/200
200/200 [==============================] - 0s 242us/sample - loss: 0.3361 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4209 -
val_sparse_categorical_accuracy: 0.8950
Epoch 176/200
200/200 [==============================] - 0s 269us/sample - loss: 0.3351 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4203 -
val_sparse_categorical_accuracy: 0.8950
Epoch 177/200
200/200 [==============================] - 0s 276us/sample - loss: 0.3349 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4215 -
val_sparse_categorical_accuracy: 0.8950
Epoch 178/200
200/200 [==============================] - 0s 275us/sample - loss: 0.3362 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4187 -
val_sparse_categorical_accuracy: 0.8850
Epoch 179/200
200/200 [==============================] - 0s 263us/sample - loss: 0.3342 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4201 -
val_sparse_categorical_accuracy: 0.8950
Epoch 180/200
200/200 [==============================] - 0s 252us/sample - loss: 0.3331 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4189 -
val_sparse_categorical_accuracy: 0.8900
Epoch 181/200
200/200 [==============================] - 0s 257us/sample - loss: 0.3327 -
```

```
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4188 -
val_sparse_categorical_accuracy: 0.8950
Epoch 182/200
200/200 [==============================] - 0s 256us/sample - loss: 0.3317 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4175 -
val_sparse_categorical_accuracy: 0.8900
Epoch 183/200
200/200 [==============================] - 0s 229us/sample - loss: 0.3312 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4174 -
val_sparse_categorical_accuracy: 0.8900
Epoch 184/200
200/200 [==============================] - 0s 225us/sample - loss: 0.3315 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4167 -
val_sparse_categorical_accuracy: 0.8900
Epoch 185/200
200/200 [==============================] - 0s 274us/sample - loss: 0.3308 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4176 -
val_sparse_categorical_accuracy: 0.8950
Epoch 186/200
200/200 [==============================] - 0s 274us/sample - loss: 0.3298 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4159 -
val_sparse_categorical_accuracy: 0.8900
Epoch 187/200
200/200 [==============================] - 0s 269us/sample - loss: 0.3294 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4153 -
val_sparse_categorical_accuracy: 0.8900
Epoch 188/200
200/200 [==============================] - 0s 260us/sample - loss: 0.3288 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4171 -
val_sparse_categorical_accuracy: 0.8950
Epoch 189/200
200/200 [==============================] - 0s 267us/sample - loss: 0.3277 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4165 -
val_sparse_categorical_accuracy: 0.8950
Epoch 190/200
200/200 [==============================] - 0s 238us/sample - loss: 0.3286 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4143 -
val_sparse_categorical_accuracy: 0.8900
Epoch 191/200
200/200 [==============================] - 0s 274us/sample - loss: 0.3268 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4144 -
val_sparse_categorical_accuracy: 0.8900
Epoch 192/200
200/200 [==============================] - 0s 259us/sample - loss: 0.3263 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4145 -
val_sparse_categorical_accuracy: 0.8900
Epoch 193/200
200/200 [==============================] - 0s 251us/sample - loss: 0.3257 -
```

```
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4133 -
val_sparse_categorical_accuracy: 0.8900
Epoch 194/200
200/200 [==============================] - 0s 199us/sample - loss: 0.3257 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4131 -
val_sparse_categorical_accuracy: 0.8900
Epoch 195/200
200/200 [==============================] - 0s 238us/sample - loss: 0.3245 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4140 -
val_sparse_categorical_accuracy: 0.8900
Epoch 196/200
200/200 [==============================] - 0s 200us/sample - loss: 0.3244 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4135 -
val_sparse_categorical_accuracy: 0.8950
Epoch 197/200
200/200 [==============================] - 0s 194us/sample - loss: 0.3242 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4116 -
val_sparse_categorical_accuracy: 0.8900
Epoch 198/200
200/200 [==============================] - 0s 181us/sample - loss: 0.3235 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4121 -
val_sparse_categorical_accuracy: 0.8900
Epoch 199/200
200/200 [==============================] - 0s 189us/sample - loss: 0.3224 -
sparse_categorical_accuracy: 0.9300 - val_loss: 0.4113 -
val_sparse_categorical_accuracy: 0.8900
Epoch 200/200
200/200 [==============================] - 0s 233us/sample - loss: 0.3223 -
sparse_categorical_accuracy: 0.9250 - val_loss: 0.4113 -
val_sparse_categorical_accuracy: 0.8900
200/200 [==============================] - 0s 397us/sample - loss: 0.4113 -
sparse_categorical_accuracy: 0.8900
200/200 [==============================] - 0s 41us/sample - loss: 0.4113 -
sparse_categorical_accuracy: 0.8900
Test accuracy : 89.0%
```

[13]:
```
from matplotlib import pyplot as plt
plt.rcParams['figure.figsize'] = [16, 10]

records = history.history
fig, ax = plt.subplots(1, 2)

ax[0].plot(records["sparse_categorical_accuracy"], label="Train")
ax[0].plot(records["val_sparse_categorical_accuracy"], label="Test")
ax[0].set(xlabel="Epoch", ylabel="Accuracy", title="Model accuracy")
ax[0].legend(loc="upper left")
```

```
ax[1].plot(records["loss"], label="Train")
ax[1].plot(records["val_loss"], label="Test")
ax[1].set(xlabel="Epoch", ylabel="Loss", title="Model loss")
ax[1].legend(loc="upper right")
```

[13]:
```
<matplotlib.legend.Legend at 0x7fcdcc707880>
```

2.2 Using Tensorflow 1

[14]:
```
tf.compat.v1.disable_eager_execution()

def shuffle_batch(X, y, batch_size):
    rnd_idx = np.random.permutation(len(X))
    n_batches = len(X) // batch_size
    for batch_idx in np.array_split(rnd_idx, n_batches):
        X_batch, y_batch = X[batch_idx], y[batch_idx]
        yield X_batch, y_batch

def reset_graph(seed=42):
    tf.compat.v1.reset_default_graph()
    tf.random.set_seed(seed)
    np.random.seed(seed)

def neuron_layer(X, n_neurons, name, activation=None):
```

```
    with tf.name_scope(name):
        n_inputs = int(X.get_shape()[1])
        stddev = 2 / np.sqrt(n_inputs)
        init = tf.random.truncated_normal((n_inputs, n_neurons), stddev=stddev)
        W = tf.Variable(init, name="kernel")
        b = tf.Variable(tf.zeros([n_neurons]), name="bias")
        Z = tf.matmul(X, W) + b
        if activation is not None:
            return activation(Z)
        else:
            return Z

n_inputs = 2
n_hidden1 = 4
n_outputs = 2

reset_graph()

X = tf.compat.v1.placeholder(tf.float32, shape=(None, n_inputs), name="X")
y = tf.compat.v1.placeholder(tf.int32, shape=(None), name="y")

with tf.name_scope("dnn"):
    hidden1 = neuron_layer(X, n_hidden1, name="hidden1", activation=tf.nn.relu)
    logits = neuron_layer(hidden1, n_outputs, name="outputs")

with tf.name_scope("loss"):
    xentropy = tf.nn.sparse_softmax_cross_entropy_with_logits(labels=y,␣
 ↪logits=logits)
    loss = tf.reduce_mean(xentropy, name="loss")

learning_rate = 0.01
with tf.name_scope("train"):
    optimizer = tf.compat.v1.train.GradientDescentOptimizer(learning_rate)
    training_op = optimizer.minimize(loss)

with tf.name_scope("eval"):
    correct = tf.math.in_top_k(y, logits, 1)
    accuracy = tf.reduce_mean(tf.cast(correct, tf.float32))

n_epochs = 100
batch_size = 10

with tf.compat.v1.Session() as sess:
    tf.compat.v1.global_variables_initializer().run()
    for epoch in range(n_epochs):
        for x_batch, y_batch in shuffle_batch(x_train_norm, y_train_norm,␣
 ↪batch_size):
```

```
            sess.run(training_op, feed_dict={X: x_batch, y: y_batch})
        acc_batch = accuracy.eval(feed_dict={X: x_batch, y: y_batch})
        acc_val = accuracy.eval(feed_dict={X: x_test_norm, y: y_test_norm})
        print(epoch, "Batch accuracy:", acc_batch, "Val accuracy:", acc_val)


    Z = logits.eval(feed_dict={X: x_test_norm})
    y_pred = np.argmax(Z, axis=1)

    print("Predicted classes:", y_pred)
    print("Actual classes:   ", y_test_norm)
```

```
WARNING:tensorflow:From /usr/lib/python3.8/site-
packages/tensorflow_core/python/ops/resource_variable_ops.py:1628: calling
BaseResourceVariable.__init__ (from tensorflow.python.ops.resource_variable_ops)
with constraint is deprecated and will be removed in a future version.
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
0 Batch accuracy: 0.9 Val accuracy: 0.885
1 Batch accuracy: 1.0 Val accuracy: 0.895
2 Batch accuracy: 1.0 Val accuracy: 0.895
3 Batch accuracy: 1.0 Val accuracy: 0.895
4 Batch accuracy: 0.9 Val accuracy: 0.9
5 Batch accuracy: 0.9 Val accuracy: 0.9
6 Batch accuracy: 0.9 Val accuracy: 0.905
7 Batch accuracy: 0.9 Val accuracy: 0.9
8 Batch accuracy: 1.0 Val accuracy: 0.9
9 Batch accuracy: 1.0 Val accuracy: 0.9
10 Batch accuracy: 1.0 Val accuracy: 0.9
11 Batch accuracy: 1.0 Val accuracy: 0.895
12 Batch accuracy: 1.0 Val accuracy: 0.89
13 Batch accuracy: 0.9 Val accuracy: 0.89
14 Batch accuracy: 1.0 Val accuracy: 0.89
15 Batch accuracy: 1.0 Val accuracy: 0.89
16 Batch accuracy: 1.0 Val accuracy: 0.89
17 Batch accuracy: 0.9 Val accuracy: 0.89
18 Batch accuracy: 1.0 Val accuracy: 0.89
19 Batch accuracy: 0.9 Val accuracy: 0.885
20 Batch accuracy: 1.0 Val accuracy: 0.885
21 Batch accuracy: 0.9 Val accuracy: 0.885
22 Batch accuracy: 0.9 Val accuracy: 0.885
23 Batch accuracy: 0.8 Val accuracy: 0.885
24 Batch accuracy: 1.0 Val accuracy: 0.885
25 Batch accuracy: 0.9 Val accuracy: 0.885
26 Batch accuracy: 0.9 Val accuracy: 0.885
27 Batch accuracy: 0.8 Val accuracy: 0.885
28 Batch accuracy: 0.9 Val accuracy: 0.885
```

```
29 Batch accuracy: 1.0 Val accuracy: 0.885
30 Batch accuracy: 1.0 Val accuracy: 0.885
31 Batch accuracy: 1.0 Val accuracy: 0.885
32 Batch accuracy: 1.0 Val accuracy: 0.885
33 Batch accuracy: 1.0 Val accuracy: 0.885
34 Batch accuracy: 0.9 Val accuracy: 0.885
35 Batch accuracy: 1.0 Val accuracy: 0.885
36 Batch accuracy: 0.9 Val accuracy: 0.885
37 Batch accuracy: 1.0 Val accuracy: 0.885
38 Batch accuracy: 0.9 Val accuracy: 0.89
39 Batch accuracy: 0.9 Val accuracy: 0.89
40 Batch accuracy: 1.0 Val accuracy: 0.89
41 Batch accuracy: 1.0 Val accuracy: 0.89
42 Batch accuracy: 0.9 Val accuracy: 0.89
43 Batch accuracy: 1.0 Val accuracy: 0.89
44 Batch accuracy: 0.9 Val accuracy: 0.89
45 Batch accuracy: 1.0 Val accuracy: 0.89
46 Batch accuracy: 1.0 Val accuracy: 0.89
47 Batch accuracy: 0.8 Val accuracy: 0.89
48 Batch accuracy: 0.8 Val accuracy: 0.89
49 Batch accuracy: 0.9 Val accuracy: 0.89
50 Batch accuracy: 0.9 Val accuracy: 0.89
51 Batch accuracy: 0.8 Val accuracy: 0.89
52 Batch accuracy: 1.0 Val accuracy: 0.89
53 Batch accuracy: 1.0 Val accuracy: 0.89
54 Batch accuracy: 0.9 Val accuracy: 0.89
55 Batch accuracy: 0.9 Val accuracy: 0.89
56 Batch accuracy: 1.0 Val accuracy: 0.89
57 Batch accuracy: 0.9 Val accuracy: 0.89
58 Batch accuracy: 0.8 Val accuracy: 0.89
59 Batch accuracy: 0.9 Val accuracy: 0.89
60 Batch accuracy: 0.9 Val accuracy: 0.89
61 Batch accuracy: 1.0 Val accuracy: 0.89
62 Batch accuracy: 0.9 Val accuracy: 0.89
63 Batch accuracy: 0.9 Val accuracy: 0.89
64 Batch accuracy: 1.0 Val accuracy: 0.89
65 Batch accuracy: 1.0 Val accuracy: 0.89
66 Batch accuracy: 1.0 Val accuracy: 0.89
67 Batch accuracy: 1.0 Val accuracy: 0.89
68 Batch accuracy: 1.0 Val accuracy: 0.89
69 Batch accuracy: 1.0 Val accuracy: 0.89
70 Batch accuracy: 1.0 Val accuracy: 0.89
71 Batch accuracy: 0.8 Val accuracy: 0.89
72 Batch accuracy: 1.0 Val accuracy: 0.89
73 Batch accuracy: 1.0 Val accuracy: 0.89
74 Batch accuracy: 1.0 Val accuracy: 0.89
75 Batch accuracy: 1.0 Val accuracy: 0.89
76 Batch accuracy: 1.0 Val accuracy: 0.89
```

```
77 Batch accuracy: 1.0 Val accuracy: 0.89
78 Batch accuracy: 1.0 Val accuracy: 0.89
79 Batch accuracy: 1.0 Val accuracy: 0.89
80 Batch accuracy: 1.0 Val accuracy: 0.89
81 Batch accuracy: 1.0 Val accuracy: 0.89
82 Batch accuracy: 0.9 Val accuracy: 0.89
83 Batch accuracy: 0.8 Val accuracy: 0.89
84 Batch accuracy: 0.9 Val accuracy: 0.89
85 Batch accuracy: 1.0 Val accuracy: 0.89
86 Batch accuracy: 1.0 Val accuracy: 0.89
87 Batch accuracy: 1.0 Val accuracy: 0.89
88 Batch accuracy: 1.0 Val accuracy: 0.89
89 Batch accuracy: 1.0 Val accuracy: 0.89
90 Batch accuracy: 0.9 Val accuracy: 0.89
91 Batch accuracy: 0.9 Val accuracy: 0.89
92 Batch accuracy: 0.9 Val accuracy: 0.89
93 Batch accuracy: 1.0 Val accuracy: 0.89
94 Batch accuracy: 0.9 Val accuracy: 0.89
95 Batch accuracy: 0.9 Val accuracy: 0.89
96 Batch accuracy: 1.0 Val accuracy: 0.89
97 Batch accuracy: 0.9 Val accuracy: 0.89
98 Batch accuracy: 0.9 Val accuracy: 0.89
99 Batch accuracy: 1.0 Val accuracy: 0.89
Predicted classes: [0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0
0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 1 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0
 0 1 0 0 0 0 0 0 0 1 1 1 1 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 0 1 1 0 1 1 1 1 0 1 0 0 1 1 1 1 1 1 1 0 0 0 1 1 0 1 1 1 1 1 0 1 1 1 1 1 1
 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1]
Actual classes:    [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
```