```
{"nbformat":4,"nbformat_minor":0,"metadata":{"colab":{"provenance":[]},"kernelspec":
{"name":"python3","display_name":"Python 3"},"language_info":{"name":"python"}},"cells":
[{"cell_type":"code","metadata":{"id":"mCGU3AhSlp2W","executionInfo":
{"status":"ok","timestamp":1769770079790,"user_tz":180,"elapsed":24864,"user":
{"displayName":"Luciano de Andrade Barbosa","userId":"01495378349148092861"}}},"source":
["# mlp for multiclass classification\n","from numpy import argmax\n","from pandas import
read_csv\n","from sklearn.model_selection import train_test_split\n","from sklearn.preprocessing
import LabelEncoder\n","from tensorflow.keras import Sequential\n","from
tensorflow.keras.layers import Dense"],"execution_count":1,"outputs":[]},
{"cell_type":"code","metadata":{"colab":
{"base_uri":"https://localhost:8080/"},"id":"71tLw6DVmeo3","outputId":"cbf2ba1a-3a69-4a4e-
aa11-337de33dab39","executionInfo":
{"status":"ok","timestamp":1769770092129,"user_tz":180,"elapsed":169,"user":
{"displayName":"Luciano de Andrade Barbosa","userId":"01495378349148092861"}}},"source":
["# load the dataset\n","path =
'https://raw.githubusercontent.com/jbrownlee/Datasets/master/iris.csv'\n","df = read_csv(path,
header=None)\n","# split into input and output columns\n","X, y = df.values[:, :-1], df.values[:,
-1]\n","# ensure all data are floating point values\n","X = X.astype('float32')\n","# encode strings
to integer\n","y = LabelEncoder().fit_transform(y)\n","# split into train and test
datasets\n","X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.33)\n","print(X_train.shape, X_test.shape, y_train.shape, y_test.shape)\n","#
determine the number of input features\n","n_features =
X_train.shape[1]"],"execution_count":2,"outputs":[{"output_type":"stream","name":"stdout","text":
["(100, 4) (50, 4) (100,) (50,)\n"]}]},{"cell_type":"code","metadata":
{"id":"KEzTT8ogmmoG","colab":{"base_uri":"https://localhost:8080/"},"executionInfo":
{"status":"ok","timestamp":1769770210248,"user_tz":180,"elapsed":350,"user":
{"displayName":"Luciano de Andrade
Barbosa","userId":"01495378349148092861"}},"outputId":"522d12d1-e39c-46e4-f74f-
79af00157f3e"},"source":["# define model\n","model = Sequential()\n","model.add(Dense(10,
activation='relu', kernel_initializer='he_normal', input_shape=
(n_features,)))\n","model.add(Dense(8, activation='relu',
kernel_initializer='he_normal'))\n","model.add(Dense(3,
activation='softmax'))"],"execution_count":3,"outputs":
[{"output_type":"stream","name":"stderr","text":["/usr/local/lib/python3.12/dist-
packages/keras/src/layers/core/dense.py:93: UserWarning: Do not pass an
`input_shape`/`input_dim` argument to a layer. When using Sequential models, prefer using an
`Input(shape)` object as the first layer in the model instead.\n","
super().__init__(activity_regularizer=activity_regularizer, **kwargs)\n"]}]},
{"cell_type":"code","metadata":{"colab":
{"base_uri":"https://localhost:8080/"},"id":"haGlv1Nzmorl","outputId":"e827eccc-e3da-436e-b54e-
2bc1f5d63d4c","executionInfo":
{"status":"ok","timestamp":1769770248708,"user_tz":180,"elapsed":9830,"user":
{"displayName":"Luciano de Andrade Barbosa","userId":"01495378349148092861"}}},"source":
["# compile the model\n","model.compile(optimizer='adam',
loss='sparse_categorical_crossentropy', metrics=['accuracy'])\n","# fit the
model\n","model.fit(X_train, y_train, epochs=150, batch_size=32,
verbose=0)"],"execution_count":4,"outputs":[{"output_type":"execute_result","data":{"text/plain":
[""]},"metadata":{},"execution_count":4}]},{"cell_type":"code","metadata":{"colab":
{"base_uri":"https://localhost:8080/"},"id":"K0oDNi36m0ns","outputId":"25d1acf2-2dda-4a66-
ad74-f8f60f2fdc3c","executionInfo":
{"status":"ok","timestamp":1769770318966,"user_tz":180,"elapsed":210,"user":
{"displayName":"Luciano de Andrade Barbosa","userId":"01495378349148092861"}}},"source":
["# evaluate the model\n","loss, acc = model.evaluate(X_test, y_test, verbose=0)\n","print('Test
Accuracy: %.3f' % acc)\n","# make a prediction\n","row = [5.1,3.5,1.4,0.2]\n","import numpy as
np\n","yhat = model.predict(np.array([row]))\n","print('Predicted: %s (class=%d)' % (yhat,
argmax(yhat)))"],"execution_count":6,"outputs":[{"output_type":"stream","name":"stdout","text":
```

["Test Accuracy: 0.900\n","\u001b[1m1/1\u001b[0m
\u001b[32m━━━━━━━━━━━━━━━━━━━━\u001b[0m\u001b[37m\u001b[0m
\u001b[1m0s\u001b[0m 70ms/step\n","Predicted: [[0.9263848 0.07129735 0.00231787]]
(class=0)\n"]}]},{"cell_type":"code","metadata":{"colab":
{"base_uri":"https://localhost:8080/"},"id":"3DR7JYe4xDrC","outputId":"99126356-7c00-4a9f-
eb91-765ac8710fe4"},"source":["# mlp for regression\n","from numpy import sqrt\n","from
pandas import read_csv\n","from sklearn.model_selection import train_test_split\n","from
tensorflow.keras import Sequential\n","from tensorflow.keras.layers import Dense\n","# load the
dataset\n","path =
'https://raw.githubusercontent.com/jbrownlee/Datasets/master/housing.csv'\n","df =
read_csv(path, header=None)\n","# split into input and output columns\n","X, y = df.values[:, :-1],
df.values[:, -1]\n","# split into train and test datasets\n","X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.33)\n","print(X_train.shape, X_test.shape, y_train.shape,
y_test.shape)\n","# determine the number of input features\n","n_features =
X_train.shape[1]\n","# define model\n","model = Sequential()\n","model.add(Dense(10,
activation='relu', kernel_initializer='he_normal', input_shape=
(n_features,)))\n","model.add(Dense(8, activation='relu',
kernel_initializer='he_normal'))\n","model.add(Dense(1))\n","# compile the
model\n","model.compile(optimizer='adam', loss='mse')\n","# fit the model\n","model.fit(X_train,
y_train, epochs=150, batch_size=32, verbose=0)\n","# evaluate the model\n","error =
model.evaluate(X_test, y_test, verbose=0)\n","print('MSE: %.3f, RMSE: %.3f' % (error,
sqrt(error)))\n","# make a prediction\n","row =
[0.00632,18.00,2.310,0,0.5380,6.5750,65.20,4.0900,1,296.0,15.30,396.90,4.98]\n","yhat =
model.predict([row])\n","print('Predicted: %.3f' % yhat)"],"execution_count":null,"outputs":
[{"output_type":"stream","text":["(339, 13) (167, 13) (339,) (167,)\n","MSE: 47.145, RMSE:
6.866\n","Predicted: 27.508\n"],"name":"stdout"}]},{"cell_type":"code","metadata":{"colab":
{"base_uri":"https://localhost:8080/"},"id":"3hRyaZCryHri","outputId":"1015f67d-bdc2-48ee-fa95-
b5e3b90ef173"},"source":["#https://machinelearningmastery.com/how-to-stop-training-deep-
neural-networks-at-the-right-time-using-early-stopping/\n","# mlp overfit on the moons dataset
with simple early stopping\n","from sklearn.datasets import make_moons\n","from keras.models
import Sequential\n","from keras.layers import Dense\n","from keras.callbacks import
EarlyStopping\n","from matplotlib import pyplot\n","# generate 2d classification dataset\n","X, y =
make_moons(n_samples=100, noise=0.2, random_state=1)\n","# split into train and
test\n","n_train = 30\n","trainX, testX = X[:n_train, :], X[n_train:, :]\n","trainy, testy = y[:n_train],
y[n_train:]\n","# define model\n","model = Sequential()\n","model.add(Dense(500, input_dim=2,
activation='relu'))\n","model.add(Dense(1,
activation='sigmoid'))\n","model.compile(loss='binary_crossentropy', optimizer='adam', metrics=
['accuracy'])\n","# simple early stopping\n","es = EarlyStopping(monitor='val_loss', mode='min',
verbose=1)\n","# fit model\n","history = model.fit(trainX, trainy, validation_data=(testX, testy),
epochs=4000, verbose=0, callbacks=[es])\n","# evaluate the model\n","_, train_acc =
model.evaluate(trainX, trainy, verbose=0)\n","_, test_acc = model.evaluate(testX, testy,
verbose=0)\n","print('Train: %.3f, Test: %.3f' % (train_acc,
test_acc))"],"execution_count":null,"outputs":[{"output_type":"stream","text":["Epoch 00224: early
stopping\n","Train: 0.967, Test: 0.814\n"],"name":"stdout"}]}]}