import pandas as pd

import numpy as np

from keras.models import Sequential

from keras.layers import Dense

from sklearn.model\_selection import train\_test\_split

#資料預處理

df=pd.read\_csv("iris\_data.csv")

df\_x=df.drop('target',axis=1)

df\_y=pd.get\_dummies(df['target'])

X\_train, X\_test, y\_train, y\_test = train\_test\_split(df\_x, df\_y, test\_size=0.3)

#定義模型

model = Sequential()

model.add(Dense(6,input\_shape=(4,),activation="relu"))

model.add(Dense(6, activation="relu"))

model.add(Dense(3, activation="sigmoid"))

model.summary() #顯示摘要資訊

#編譯模型

model.compile(loss="categorical\_crossentropy", optimizer="adam", metrics=["accuracy"])

#訓練模型

model.fit(X\_train,y\_train, epochs=100, batch\_size=5)

loss, accuracy = model.evaluate(X\_test,y\_test)

print("Accuracy = {:.2f}".format(accuracy))

model.save('iris.h5')

def convert\_to\_labels(df):

arr=np.array(df)

out=[]

for data in arr:

out.append(list(data).index(max(data)))

return np.array(out)

Y\_pred= model.predict(X\_test)

Y\_pred=convert\_to\_labels(Y\_pred)

ytest=convert\_to\_labels(y\_test)

tb=pd.crosstab(ytest, Y\_pred,rownames= ["label"], colnames=["predict"])

print(tb)

import pandas as pd

import numpy as np

from keras.models import Sequential

from keras.layers import Dense

from sklearn.model\_selection import train\_test\_split

#資料預處理

df=pd.read\_csv("boston\_housing.csv")

df\_x=df.drop('medv',axis=1)

df\_y=df['medv']

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(df\_x, df\_y, test\_size=0.3)

#定義模型

def build\_model():

model = Sequential()

model.add(Dense(32, input\_shape=(X\_train.shape[1],), activation="relu"))

model.add(Dense(1))

model.compile(loss="mse", optimizer="adam", metrics=["mae"])

model.summary() #顯示摘要資訊

return model

k = 4

nb\_val\_samples = len(X\_train) // k

nb\_epochs = 80

mse\_scores = []

mae\_scores = []

for i in range(k):

print("Processing Fold #" + str(i))

X\_val = X\_train[i\*nb\_val\_samples: (i+1)\*nb\_val\_samples]

Y\_val = Y\_train[i\*nb\_val\_samples: (i+1)\*nb\_val\_samples]

X\_train\_p = np.concatenate([X\_train[:i\*nb\_val\_samples], X\_train[(i+1)\*nb\_val\_samples:]], axis=0)

Y\_train\_p = np.concatenate([Y\_train[:i\*nb\_val\_samples], Y\_train[(i+1)\*nb\_val\_samples:]], axis=0)

model = build\_model()

model.fit(X\_train\_p, Y\_train\_p, epochs=nb\_epochs, batch\_size=16, verbose=0)

mse, mae = model.evaluate(X\_val, Y\_val)

mse\_scores.append(mse)

mae\_scores.append(mae)