機器學習概論作業

範圍： Implementing DNN model using Keras

銘傳大學電腦與通訊工程系

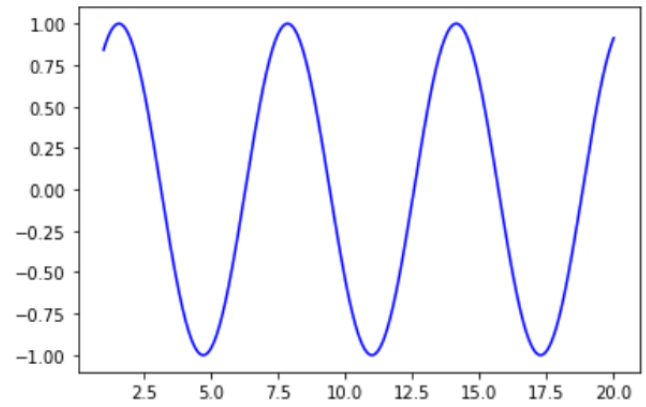
班　　級	電通三乙
姓　　名	李柏賢
學　　號	07050862
作業成果	應繳作業共 4 題，每題 25 分 我共完成 3.X 題，應得 90 分
授課教師	陳慶逸

■ 請確實填寫自己寫完成題數，填寫不實者(如上傳與作業明顯無關的答案，或是計算題數有誤者)，本次作業先扣 50 分。

一、 試建立一個 DNN 模型來擬合(鑑別) sin(x), $1 \leqq x \leqq 20$, 且資料點為 1000 點:

```
Points = 1000
X = np.linspace(1, 20, points)
Y = np.sin(X)
```

```
[<matplotlib.lines.Line2D at 0x7f9049ceae48>]
```



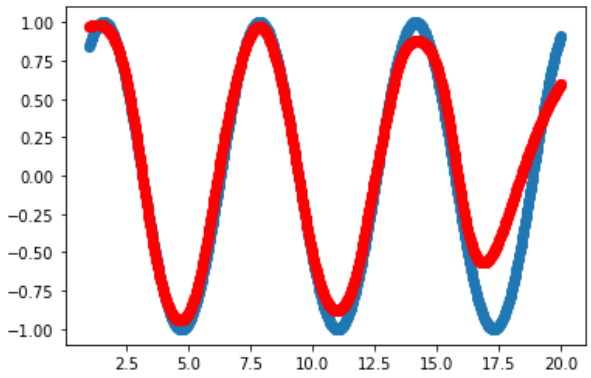
1. 訓練完成後，將訓練資料輸入 DNN 模型後，其輸出應要能完全擬合(fitting)上面波形。
2. 請用 model.summary()來產生模型的參數資料
3. 請用 plot_model()來產生 DNN 模型的架構圖

程式碼:

```
import numpy as np
import matplotlib.pyplot as plt
from keras.models import Sequential
from keras.layers import Dense
from keras.optimizers import SGD, Adam
from keras.utils import plot_model

points = 1000
X = np.linspace(1, 20, points)
y = np.sin(X)


model = Sequential()
model.add(Dense(50, input_dim =1, activation='sigmoid'))
model.add(Dense(30, activation='sigmoid'))
model.add(Dense(1))
```

```
sgd = SGD(lr=0.0553, decay=1e-6, momentum=0.9, nesterov=True)
model.compile(loss='mse', optimizer=sgd)
model.fit(X, y, epochs=560)
model.summary()

predictions = model.predict(X)
plt.scatter(X, y)
plt.plot(X, predictions, 'ro')
plt.show()
plot_model(model, to_file="Ch16_1.png", show_shapes=True)
```

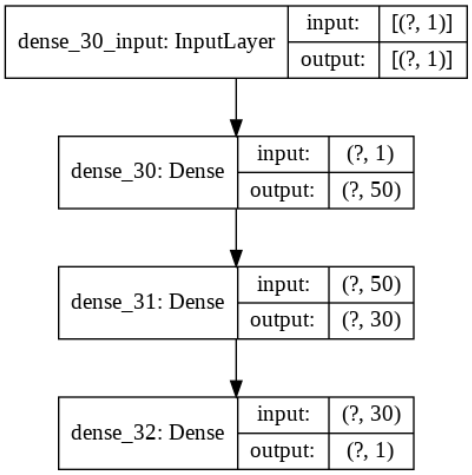
函數鑑別結果:



模型的參數資料:

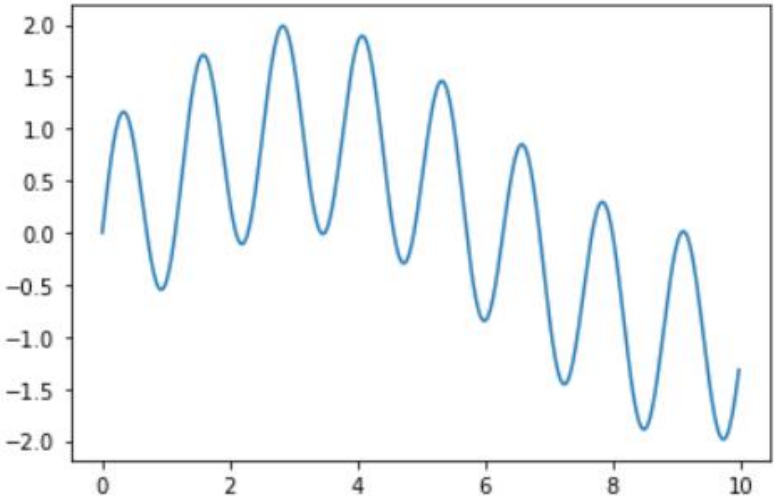
```
Model: "sequential_10"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_30 (Dense)             (None, 50)                100
_________________________________________________________________
dense_31 (Dense)             (None, 30)                1530
_________________________________________________________________
dense_32 (Dense)             (None, 1)                 31
=================================================================
Total params: 1,661
Trainable params: 1,661
Non-trainable params: 0
_________________________________________________________________
```

DNN 模型的架構圖:



二、 試建立一個 DNN 模型來擬合(鑑別)下面程式所產生的資料集形態:

```
X = np.arange(0, 10, 0.02)
y = (np.sin(5 * x) + np.sin(0.5 * x))
plt.plot(X,y)
plt.show()
```

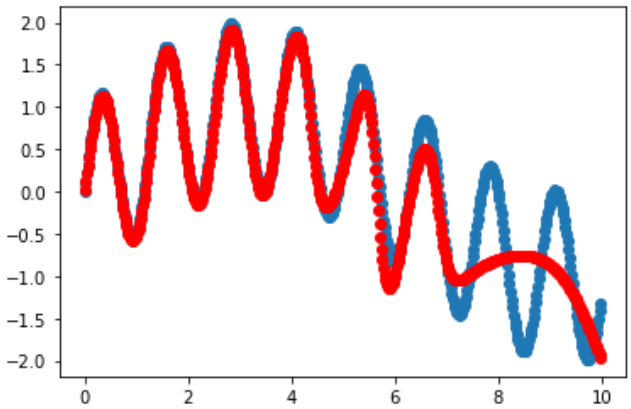


1. 訓練完成後，將訓練資料輸入 DNN 模型後，其輸出是否能夠擬合(fitting)上面波形? 請將結果貼上來(完全鑑別出來得滿分 25 分，每少涵蓋一個週期波形少 5 分)。
2. 請用 model.summary()來產生模型的參數資料
3. 請用 plot_model()來產生 DNN 模型的架構圖

程式碼:

```
import numpy as np
import matplotlib.pyplot as plt
from keras.models import Sequential
from keras.layers import Dense
from keras.optimizers import SGD, Adam

X = np.arange(0, 10, 0.02)
y = (np.sin(5 * X) + np.sin(0.5 * X))
plt.plot(X,y)
plt.show()


model = Sequential()
model.add(Dense(50, input_dim =1, activation='sigmoid'))
model.add(Dense(30, activation='sigmoid'))
model.add(Dense(1))
sgd = SGD(lr=0.04, decay=1e-6, momentum=0.9, nesterov=True)
model.compile(loss='mse', optimizer=sgd)
model.fit(X, y, epochs=2700)

predictions = model.predict(X)
plt.scatter(X, y)
plt.plot(X, predictions, 'ro')
plt.show()
model.summary()
plot_model(model, to_file="Ch16_1.png", show_shapes=True)
```

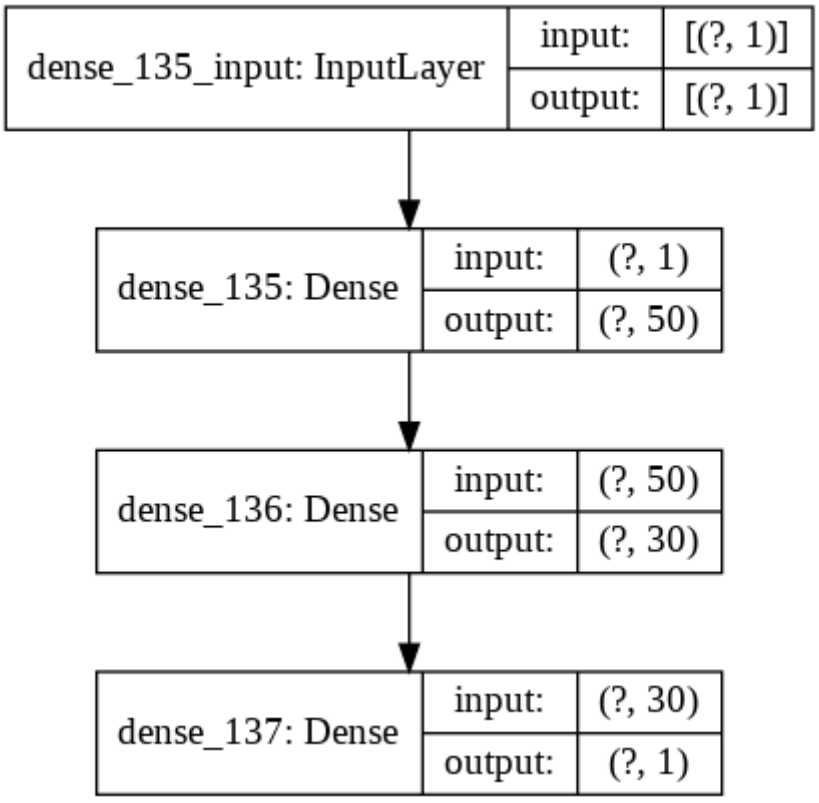
函數鑑別結果:



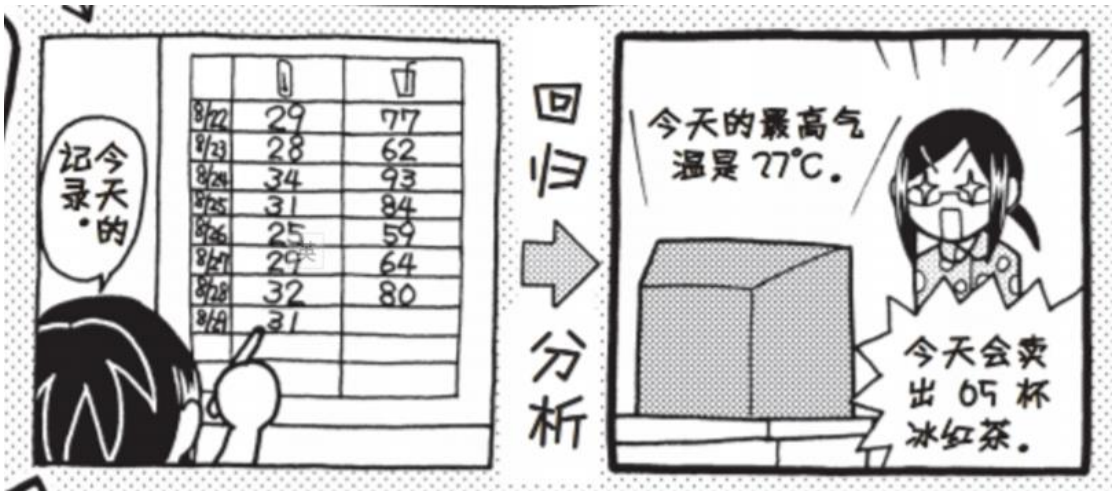
模型的參數資料:

```
Model: "sequential_61"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_135 (Dense)            (None, 50)                100
_________________________________________________________________
dense_136 (Dense)            (None, 30)                1530
_________________________________________________________________
dense_137 (Dense)            (None, 1)                 31
=================================================================
Total params: 1,661
Trainable params: 1,661
Non-trainable params: 0
```

DNN 模型的架構圖:



三、 試用 Keras 建立一個線性回歸模型來擬合下面的數據:



X = np.array([29, 28, 34, 31, 25, 29, 32, 31, 24, 33, 25, 31, 26, 30])

Y = np.array([77, 62, 93, 84, 59, 64, 80, 75, 58, 91, 51, 73, 65, 84])

提示： 不要用預設的 optimizers，可自行更改其中的參數:

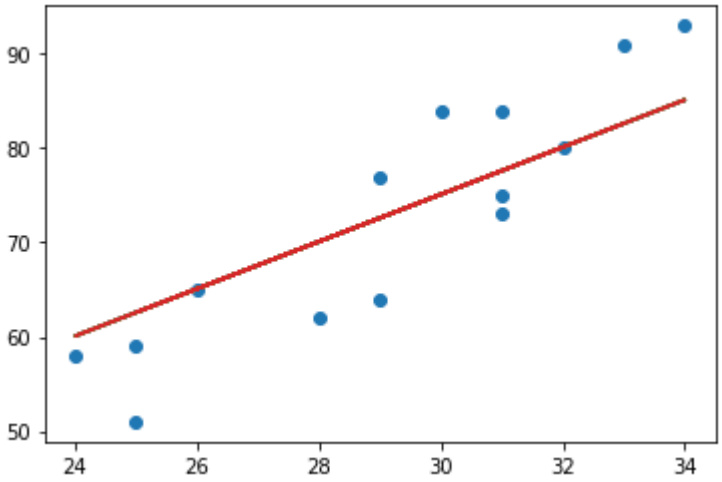
```
import numpy as np
from keras.models import Sequential
from keras.layers import Dense
from keras import optimizers
import matplotlib.pyplot as plt

# choose loss function and optimizing method
sgd = optimizers.SGD(lr=0.1, decay=1e-6, momentum=0.9, nesterov=True)
model.compile(loss='mse', optimizer= sgd)
```

1. 訓練完成後，將訓練資料輸入 DNN 模型後，其輸出應要能完全擬合(fitting)上面波形。
2. 請用 model.summary()來產生模型的參數資料
3. 請用 plot_model()來產生 DNN 模型的架構圖

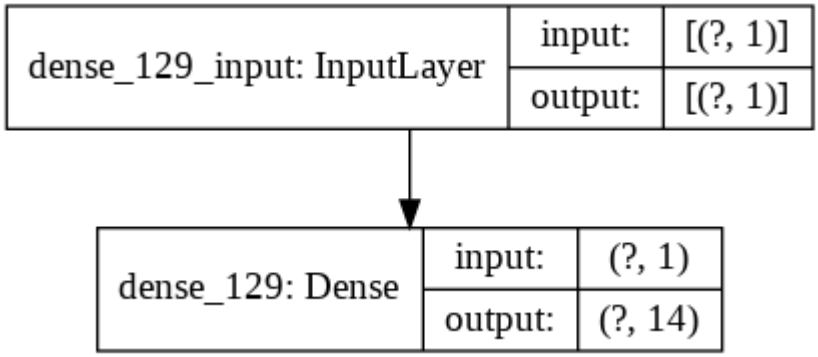
程式碼:

```
import numpy as np
from keras.models import Sequential
from keras.layers import Dense
from keras import optimizers
import matplotlib.pyplot as plt
from keras.utils import plot_model
X_train = np.array([29, 28, 34, 31, 25, 29, 32, 31, 24, 33, 25,
31, 26, 30])
Y_train = np.array([77, 62, 93, 84, 59, 64, 80, 75, 58, 91, 51,
73, 65, 84])
model = Sequential()
model.add(Dense(14, input_dim=1))
# choose loss function and optimizing method
model.compile(loss='mse', optimizer='sgd')
#訓練模型
model.fit(X_train, Y_train, epochs = 150)
# choose loss function and optimizing method
sgd = optimizers.SGD(lr=0.1, decay=1e6, momentum=0.9, nesterov=T
rue)
model.compile(loss='mse', optimizer= sgd)
model.summary()
#predictions = model.predict(X)
plt.scatter(X_train, Y_train)
plt.plot(X_train, model.predict(X_train))
plt.show()
plot_model(model, to_file="Ch16_1.png", show_shapes=True)
```

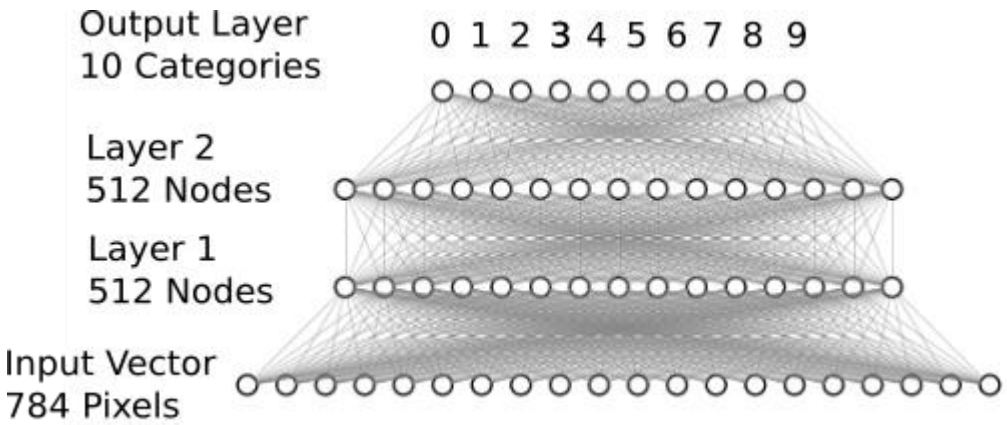
資料擬合結果：




```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_2 (Dense)              (None, 14)                28
=================================================================
Total params: 28
Trainable params: 28
Non-trainable params: 0
```



四、 試建立一個下面的 DNN 模型來實現 MNIST 的辨識:



程式碼:

```
import numpy as np
from keras.models import Sequential
from keras.datasets import mnist
from keras.layers import Dense, Dropout, Activation, Flatten
from keras.utils import np_utils
from matplotlib import pyplot as plt

# 載入 MNIST 資料庫的訓練資料，並自動分為『訓練組』及『測試組』
(X_train, y_train), (X_test, y_test) = mnist.load_data()

y_TrainOneHot = np_utils.to_categorical(y_train)
y_TestOneHot = np_utils.to_categorical(y_test)

#將 image 以 reshape 轉換為二維 ndarray 並進行 normalization
X_train_2D = X_train.reshape(60000, 28*28).astype('float32')
X_test_2D = X_test.reshape(10000, 28*28).astype('float32')
x_Train_norm = X_train_2D/255
x_Test_norm = X_test_2D/255

model = Sequential()
model.add(Dense(units=512, input_dim=784, activation='relu'))
model.add(Dense(units=10, activation='softmax'))
```

```
model.compile(loss='categorical_crossentropy', optimizer='adam',
 metrics=['accuracy'])
# 進行訓練, 訓練過程會存在 train_history 變數中
train_history = model.fit(x=x_Train_norm, y=y_TrainOneHot, valid
ation_split=0.2, epochs=10, batch_size=800, verbose=2)

scores = model.evaluate(x_Test_norm, y_TestOneHot)
print()
print("\t[Info] Accuracy of testing data = {:2.1f}%".format(scor
es[1]*100.0))

# 預測(prediction)
X = x_Test_norm[0:10,:]
predictions = model.predict_classes(X)
# get prediction result
print(predictions)
model.summary()
```

Accuracy of testing data :

```
Epoch 9/10
60/60 - 1s - loss: 0.0603 - accuracy: 0.9840 - val_loss: 0.0902 - val_accuracy: 0.9735
Epoch 10/10
60/60 - 1s - loss: 0.0517 - accuracy: 0.9863 - val_loss: 0.0857 - val_accuracy: 0.9738
313/313 [==============================] - 1s 2ms/step - loss: 0.0786 - accuracy: 0.9759

        [Info] Accuracy of testing data = 97.6%
```

模型的參數資料:

```
Model: "sequential_59"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_130 (Dense)            (None, 512)               401920
_________________________________________________________________
dense_131 (Dense)            (None, 10)                5130
=================================================================
Total params: 407,050
Trainable params: 407,050
                        Non-trainable params: 0
```