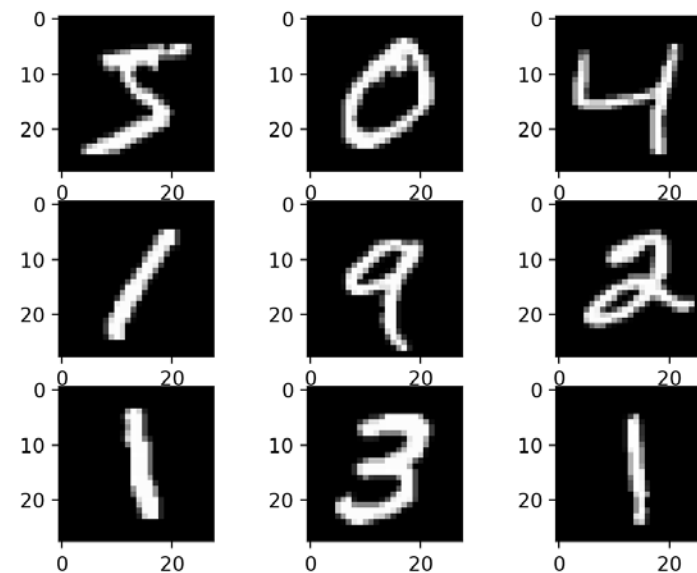


# MNIST

# MNIST

- MNIST
  - 手寫辨識
  - AI初步練習用dataset，必測的dataset
- Keras已內建dataset
  - 60000筆training
  - 10000筆testing

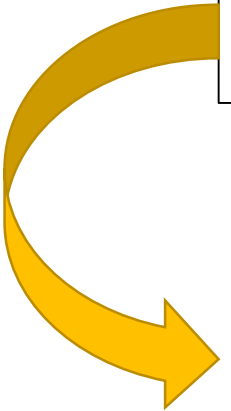


# Label



# 導入dataset

```
from keras.datasets import mnist ## 將dataset 導入進來  
(train_im, train_label), (test_im_ori, test_label) = mnist.load_data() ## 將dataset 分成test跟train
```

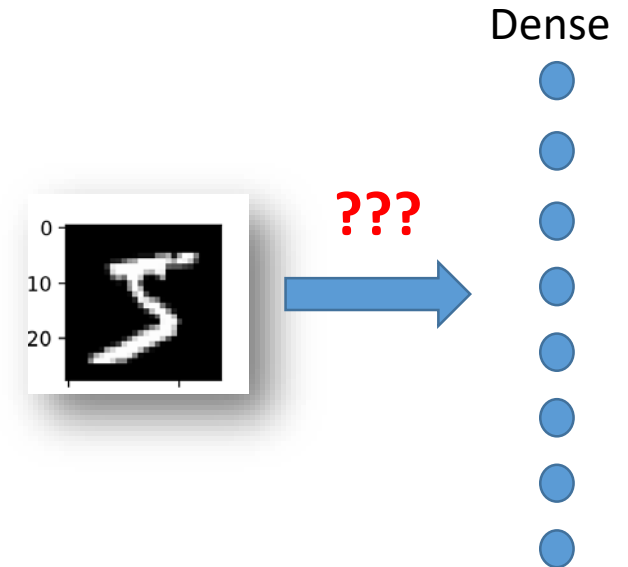


Keras 有內建幾個dataset，並且有提供讀取的方式

# 輸入前處理

```
train_im = train_im.reshape(train_im.shape[0], 28*28)
test_im = test_im_ori.reshape(test_im_ori.shape[0], 28*28)
train_label = np_utils.to_categorical(train_label, 10)
test_label = np_utils.to_categorical(test_label, 10)
```

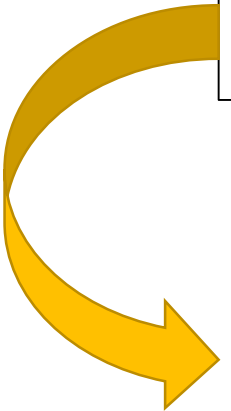
- 因為只用Dense(fully connect)，所以輸入必須一維(28\*28)
- 將進來的label轉換成[1,0,.....,0]的型態(one-hot)



# 建立模型並訓練

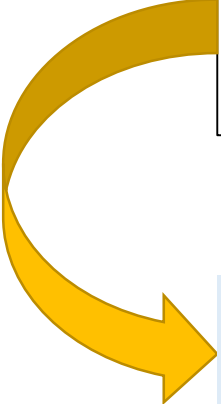
```
model = Sequential()
.... ##建立自己的模型
model.add(Dense(10, activation='softmax'))

model.compile(loss='categorical_crossentropy', optimizer='SGD', metrics=['accuracy'])
train_history = model.fit(train_im_input, train_label, epochs=Epoch, batch_size=32, validation_split=
0.1, callbacks=[TestCallback(train_im_input, train_label, test_im_input, test_label)])
```


- 
- 建立自己的模型，最後一層一定要softmax
  - 指定loss function跟訓練方式
  - 記錄下訓練的過程

# 評估結果

```
acc = model.evaluate(test_im, test_label)[1]
print ("Test Acc : ", acc*100, "%")
show_train_history(train_history, 'loss', 'val_loss')
predictions = model.predict(test_im)
```

- 
- 使用testing dataset看testing的準確率
  - 畫出訓練過程
  - 得到testing的預測結果

# 建立模型並訓練



```
draw_pic(test_im_ori[:25],predictions[:25],test_label[:25])
```

- 使用老師給的function – draw\_pic
- 並將testing dataset的 input、prediction、groundtruth 前25筆餵進去
- 可以得到如右圖，紅色代表預測錯誤



# 下載程式碼

- <https://reurl.cc/rlK2m4>

# 目標

- Testing Acc:90%
  - 非常簡單
- Testing Acc:95%
  - 簡單