Day 71 損失函數

損失函數的介紹與應用

出題教練 陳宇春

本日知識點目標

- 了解損失函數
- 針對不同的問題使用合適的損失函數

損失函數

CUPOY

- 機器學習中所有的算法都需要最大化或最小化一個函數，這個函數被稱為「目標函數」。其中，我們一般把最小化的一類函數，稱為「損失函數」。它能根據預測結果，衡量出模型預測能力的好壞
- 損失函數大致可分為：分類問題的損失函數和回歸問題的損失函數
  - Numerical Issues

損失函數為什麼是最小化

- 期望：希望模型預測出來的東西可以跟實際的值一樣
- 預測出來的東西基本上跟實際值都會有落差
  - 在回歸問題稱為「殘差(residual)」
  - 在分類問題稱為「錯誤率(error rate)」

- 損失函數中的損失就是「實際值和預測值的落差」
- y 表示實際值，ŷ 表示預測值

$$\overline{loss/residual} = y - \hat{y}$$

$$error\ \ rate = \frac{\sum_{i=1}^{n} sign(y_i \neq \hat{y}_i)}{n}, sign(y_i \neq \hat{y}_i) = \begin{cases} 1, & y_i \neq \hat{y}_i \\ 0, & y_i = \hat{y}_i \end{cases}$$

損失函數的分類介紹 - mean_squared_error

CUPOY

- 均方誤差(mean_squared_error)：就是最小平方法(Least Square) 的目標函數 -- 預測值與實際值的差距之平均值。還有其他變形的函數, 如 mean_absolute_error、mean_absolute_percentage_error、mean_squared_logarithmic_error。

$$\sum(\hat{y}^2 - y^2)/N$$

- 使用時機：
  - n 個樣本的預測值（y）與（y_）的差距
  - Numerical 相關
- Keras 上的調用方式：
  - from keras import losses
  - model.compile(loss= ‘mean_squared_error‘, optimizer='sgd’)
  - 其中，包含 y_true， y_pred 的傳遞，函數是表達如下：
  - keras.losses.mean_squared_error(y_true, y_pred)

損失函數的分類介紹 - Cross Entropy

- 當預測值與實際值愈相近，損失函數就愈小，反之差距很大，就會更影響損失函數的值
- 要用 Cross Entropy 取代 MSE，因為，在梯度下時，Cross Entropy 計算速度較快，
- 使用時機：
  - 整數目標：Sparse categorical_crossentropy
  - 分類目標：categorical_crossentropy
  - 二分類目標：binary_crossentropy。
- Keras 上的調用方式：
  - from keras import losses
  - model.compile(loss= ‘categorical_crossentropy ‘, optimizer='sgd’)
  - 其中, 包含y_true， y_pred的傳遞, 函數是表達如下：
  - keras.losses.categorical_crossentropy(y_true, y_pred)

損失函數的分類介紹: Hinge Error (hinge)

CUPOY

- 是一種單邊誤差，不考慮負值同樣也有多種變形，squared_hinge、categorical_hinge

$$\ell(y) = \max(0, 1 - t \cdot y)$$

- 使用時機：
  - 適用於『支援向量機』(SVM)的最大間隔分類法(maximum-margin classification)
- Keras 上的調用方式：
  - from keras import losses
  - model.compile(loss= ‘hinge‘, optimizer='sgd’)
  - 其中，包含 y_true，y_pred 的傳遞, 函數是表達如下:
  - keras.losses.hinge(y_true, y_pred)

特別的案例: 自定義損失函數

CUPOY

- 根據問題的實際情況，定制合理的損失函數
- 舉例：預測果汁日銷量問題，如果預測銷量大於實際銷量則會損失成本；如果預測銷量小於實際銷量則會損失利潤。
  - 考慮重點：製造一盒果汁的成本和銷售一盒果汁的利潤不是等價的
  - 需要使用符合該問題的自定義損失函數自定義損失函數為：

$$loss = \sum n\, f(y\_, y)$$

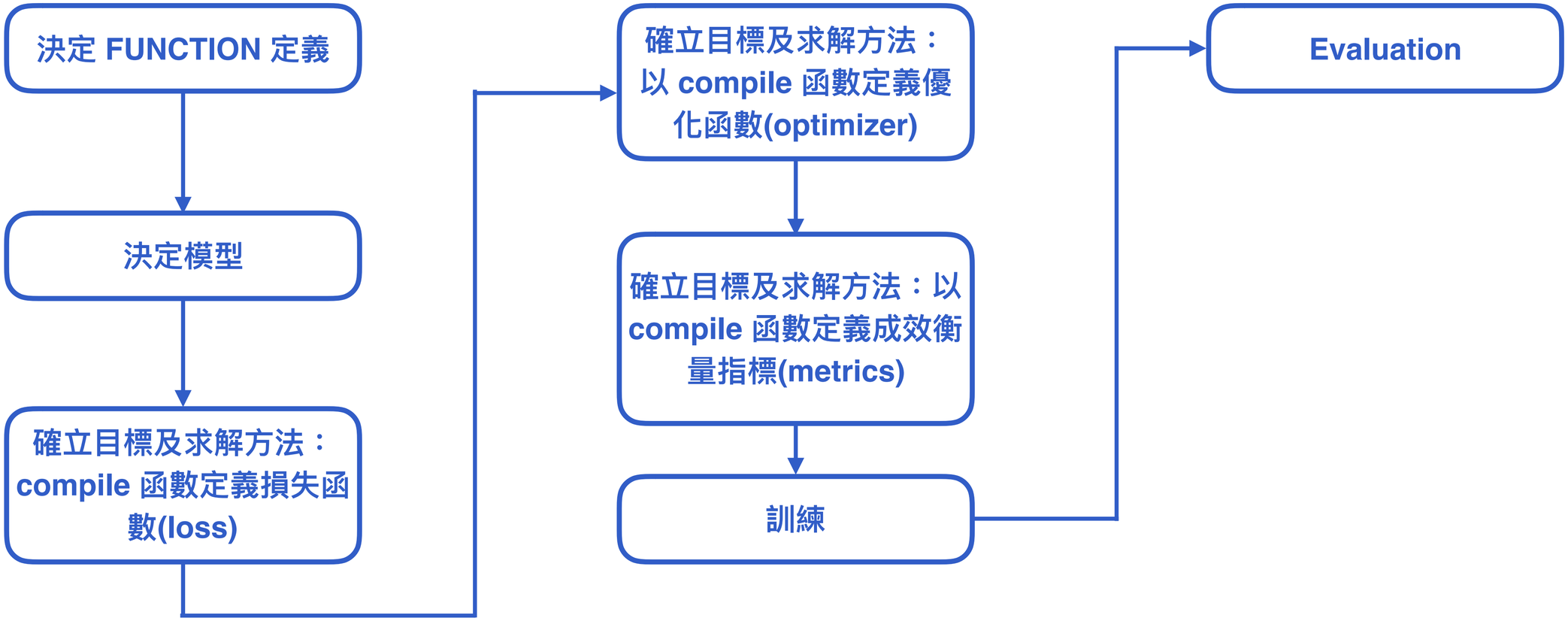
特別的案例: 自定義損失函數 (II)

CUPOY

- 接續上一頁
  - 損失函數表示若預測結果 y 小於標準答案 y_，損失函數為利潤乘以預測結果 y 與標準答案之差
  - 若預測結果 y 大於標準答案 y_，損失函數為成本乘以預測結果 y 與標準答案之差用
  - Tensorflow 函數表示為：

```
loss = tf.reduce_sum(tf.where(tf.greater(y, y_), COST*(y-y_), PROFIT*(y_-y)))
```

前述流程 / python程式 對照



前述流程 / python程式 對照

CUPOY

載入之前訓練的模型

In [21]:
```
try:
    model.load_weights("SaveModel/cifarCnnModel.h5")
    print("載入模型成功!繼續訓練模型")
except :
    print("載入模型失敗!開始訓練一個新模型")
```

```
載入模型失敗!開始訓練一個新模型
```

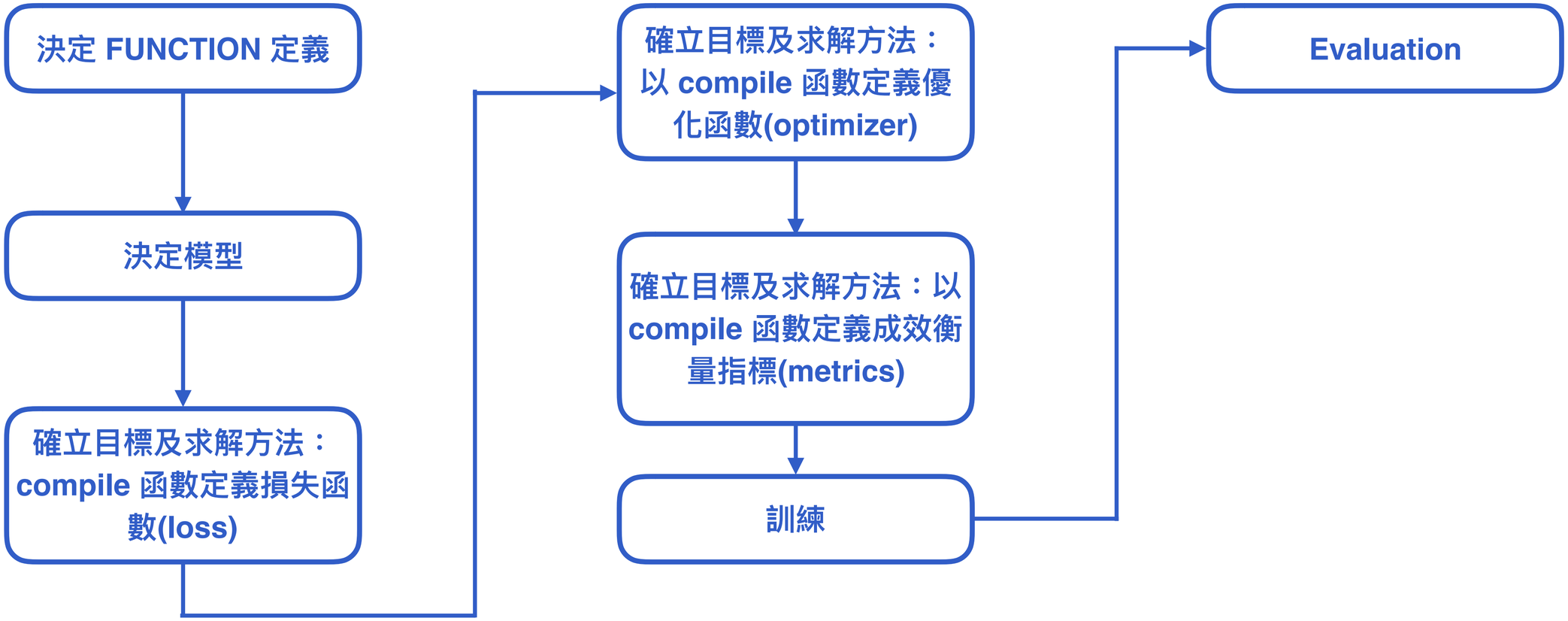
訓練模型

In [42]:
```
model.compile(loss='categorical_crossentropy', optimizer='sgd', metrics=['accuracy'])

'''
作業解答:
請分別選用 "MSE", "binary _crossentropy"
查看Train/test accurancy and loss rate
'''
```

In [43]:
```
train_history=model.fit(x_img_train_normalize, y_label_train_OneHot,
                        validation_split=0.2,
                        epochs=10, batch_size=16, verbose=1)
```

複習：流程


決定 FUNCTION 定義
決定模型
確立目標及求解方法：compile 函數定義損失函數(loss)
確立目標及求解方法：以 compile 函數定義優化函數(optimizer)
確立目標及求解方法：以 compile 函數定義成效衡量指標(metrics)
訓練
Evaluation

複習：流程

CUPOY

- 損失函數中的損失就是「實際值和預測值的落差」，損失函數是最小化
- 損失函數大致可分為：分類問題的損失函數和回歸問題的損失函數

解題時間
It’s Your Turn

請跳出PDF至官網Sample Code＆作業
開始解題