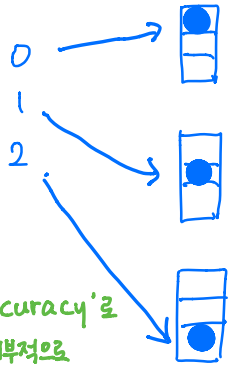
```
# Lab 6 Softmax Classifier
import tensorflow as tf
import numpy as np

x_raw = [[1, 2, 1, 1],
         [2, 1, 3, 2],
         [3, 1, 3, 4],
         [4, 1, 5, 5],
         [1, 7, 5, 5],
         [1, 2, 5, 6],
         [1, 6, 6, 6],
         [1, 7, 7, 7]]
y_raw = [[0, 0, 1],  → 2
         [0, 0, 1],  → 2
         [0, 1, 0],  → 1
         [0, 1, 0],  → 1
         [0, 1, 0],  → 1
         [1, 0, 0],  → 0
         [1, 0, 0]] → 0

x_data = np.array(x_raw, dtype=np.float32)
y_data = np.array(y_raw, dtype=np.float32)

nb_classes = 3

tf.model = tf.keras.Sequential()
tf.model.add(tf.keras.layers.Dense(input_dim=4, units=nb_classes, use_bias=True))  # use_bias is True,
by default

# use softmax activations: softmax = exp(logits) / reduce_sum(exp(logits), dim)
tf.model.add(tf.keras.layers.Activation('softmax'))

# use loss == categorical_crossentropy
tf.model.compile(loss='categorical_crossentropy', optimizer=tf.keras.optimizers.SGD(lr=0.1),
metrics=['accuracy'])
tf.model.summary()

history = tf.model.fit(x_data, y_data, epochs=2000)
print('--------------')
# Testing & One-hot encoding
a = tf.model.predict(np.array([[1, 11, 7, 9]]))
print(a, tf.keras.backend.eval(tf.argmax(a, axis=1)))

print('evaluates the value of a variable')
b = tf.model.predict(np.array([[1, 3, 4, 3]]))
print(b, tf.keras.backend.eval(tf.argmax(b, axis=1)))

print('--------------')
# or use argmax embedded method, predict_classes
c = tf.model.predict(np.array([[1, 1, 0, 1]]))
c_onehot = tf.model.predict_classes(np.array([[1, 1, 0, 1]]))
```

개까지 label을 가질 때, 하나만 Hot하게 만듦

: One-Hot Encoding



* tensorflow에서 신경망을 이해할 때 사용하는 모듈
: tf.keras.Dense

평가 기준을 'accuracy'로 지정했을 경우, 내부적으로 Categorial_accuracy() 함수를 이용하여 정확도가 계산됨

출력 값의 크기 (y)

Sequential 모델은 레이어를 선형으로 연결하여 구성하고, 레이어 인스턴스를 생성자에게 넘겨줌으로서 구성 가능

2D 레이어 / 입력형태 지정(x) / 편향(b)을 사용할지 여부

활성화 함수 / softmax를 이용

Stochastic Gradient Descent

learning rate

다중 분류 손실 함수
: 출력 값이 one-hot encoding된 결과로 나옴

평가기준

model을 요약해서 shape 등의 정보를 볼 수 있음

전체 데이터 셋에 대해 한 번 학습을 완료한 상태

loss function을 통해 구한 차이를 이용해 기울기를 구하고, network의 parameter(W, b)를 학습에 어떻게 반영할 것인지 결정하는 방법

백엔드 함수 중 하나로, 0, 1, 2번째 중 어디에 있는 값이 최대값인지 알려줌

```
print(c, c_onehot)

print('--------------')
all = tf.model.predict(np.array([[1, 11, 7, 9], [1, 3, 4, 3], [1, 1, 0, 1]]))
all_onehot = tf.model.predict_classes(np.array([[1, 11, 7, 9], [1, 3, 4, 3], [1, 1, 0, 1]]))
print(all, all_onehot)
```

* Keras에서 loss 함수

① categorial_crossentropy(다중 분류 손실 함수)
- 출력 값이 one-hot coding된 결과로 나옴
  → label(y)을 one-hot encoding 해서 넣어줘야 함
- 클래스가 상호 배타적일 경우 사용
  → 각 샘플이 정확히 하나의 클래스에 속하는 경우

② Sparse_categorial_crossentropy (다중 분류 손실 함수)
- integer type 클래스
  → one-hot encoding하지 않고 정수 형태로 label(y)을 넣어줌
- 한 샘플에 여러 클래스가 있거나 label이 soft 확률일 경우 사용

③ binary_crossentropy
- binary 다중 분류 손실 함수
- label들이 독립적일 때 사용

* epoch / batch

① epoch : 전체 dataset을 다 학습시킨 상태

② batch