Deep Learning

Softmax(Multinominal) classification:

Yoon Joong Kim
Department of Computer Engineering, Hanbat National University
yjkim@hanbat.ac.kr

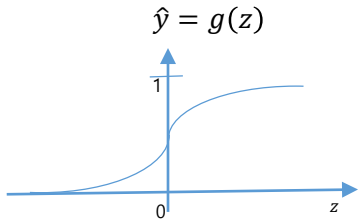
Contents

Multinomial(multiclass) classification

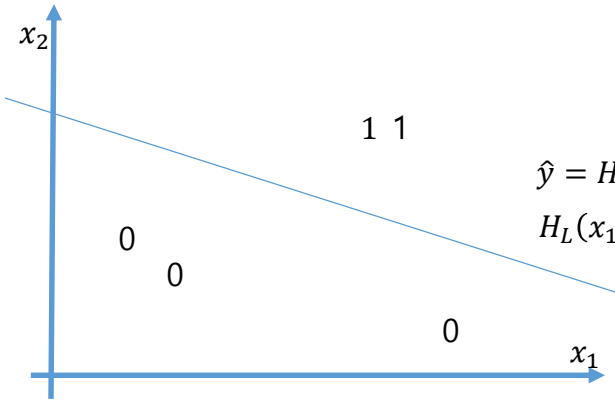
- Binary classification
  - Instance가 2 인 classes 중에서 하나를 선택하는 분류기법
  - Logistic classification(regression)
  - model
    - $\hat{y}_{lo} = H_{lo}(x) = \sigma(H_L(x))$, $H_L(x) = wx + b$, $\sigma(\mathrm{z}) = \frac{1}{1+e^{-z}}$,sigmoid
    - $loss(X, \mathrm{Y}) = ce(\mathrm{H}_{lo}(\mathrm{X}), \mathrm{Y}) = \frac{1}{N}\sum ce(\hat{y}_{i,lo}, y_i)$
- Softmax classification
  - Instance가 2이상의 인 classes중에서 하나를 선택하는 분류기법
    - $\hat{y}_{so} = H_{so}(x) = \sigma(H_L(x))$, $H_L(x) = wx + b$, $\sigma(\mathrm{z_j}) = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$ $for\ j = 1, \ldots, K$,$softmax$
    - $loss(X, Y) = \mathrm{ce}(\mathrm{H_{so}}(\mathrm{X}), \mathrm{Y}) = \frac{1}{N}\sum cross\_entropy(\hat{y}_{i,so}, y_{i\_\mathrm{ohe}})$

1. How to classify 2 classes with logistic classifier?

- How to classify 2 classes with logistic classifier?
  - Logistic regression
    - $H_{Lo}(X) = g(H_L(X))$
      - $H_L(X) = XW + b \ \ [-\infty \sim +\infty]$
      - $g(z) = \frac{1}{1+e^{-z}} \qquad [0 \sim 1]$
    - $loss(\mathrm{X}, \mathrm{Y}) = \frac{1}{N}\sum cross\_entropy(H_{Lo}(X_i), Y_i))$



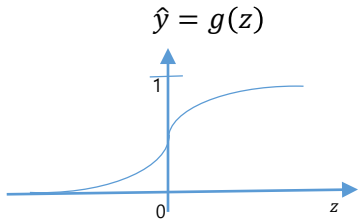
X		Y
x1 (hours)	x2 (attendance)	y (grade)
2	4	0
3	3	0
9	5	1
10	5	1
11	1	0



$$\hat{y} = H_{Lo}(x_1, x_2) = g(H_L(x_1, x_2))$$
$$H_L(x_1, x_2) = (x_1, x_2)\begin{pmatrix} w_1 \\ w_2 \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

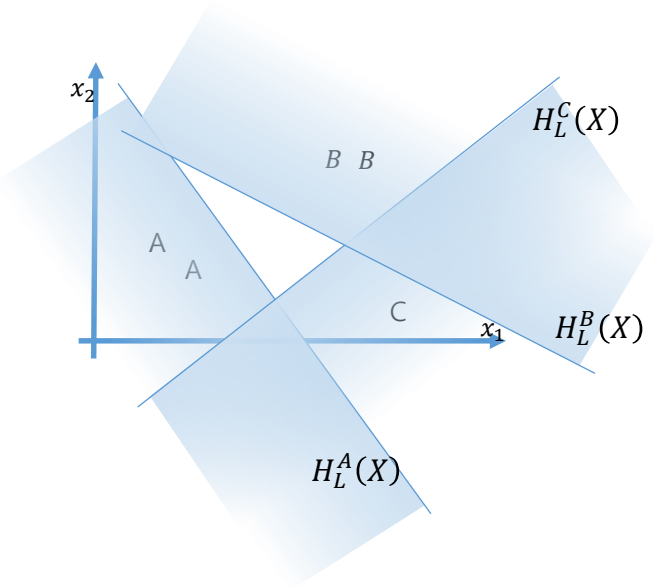
1. How to classify 2 classes with logistic classifier?(cont.)

- How to classify 3 classes with logistic classifier?
  - Logistic regression
    - $H_{Lo}(X) = g(H_L(X))$
      - $H_L(X) = XW + b \ \ [-\infty \sim +\infty]$
      - $g(z) = \frac{1}{1+e^{-z}} \qquad [0 \sim 1]$
    - $loss(\mathrm{X}, \mathrm{Y}) = \frac{1}{N}\sum cross\_entropy(H_{Lo}(X_i), Y_i))$



  - How to classify X ?

X		Y
x1 (hours)	x2 (attendance)	y (grade)
10	5	A
9	5	A
3	2	B
2	4	B
11	1	C



Let X=(10,5)

$$X \Rightarrow H_{LO}^A(X) \Rightarrow \bar{Y}^A \equiv 1.0$$
$$X \Rightarrow H_{LO}^B(X) \Rightarrow \bar{Y}^B \equiv 0.0$$
$$X \Rightarrow H_{LO}^C(X) \Rightarrow \bar{Y}^C \equiv 0.0$$

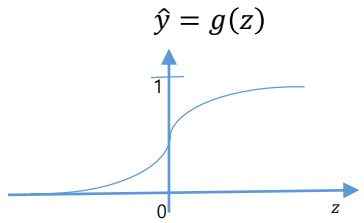
$\Rightarrow A$

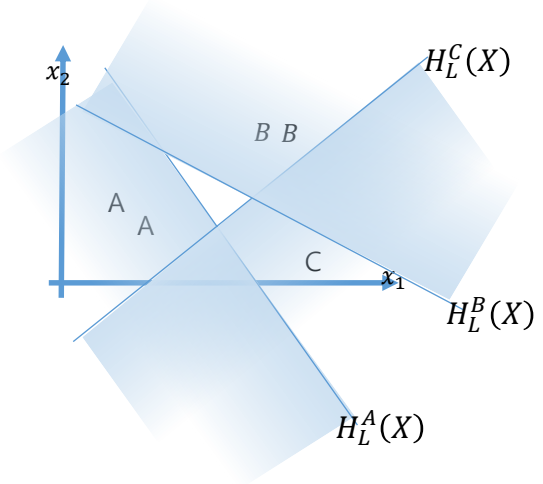
1. How to classify 2 classes with logistic classifier? (cont.)

- How to classify 3 classes with logistic classifier?
  - Logistic regression
    - $H_{Lo}(X) = g(H_L(X))$

      $H_L(X) = XW + b \ [-\infty \sim +\infty]$

      $g(z) = \frac{1}{1+e^{-z}} \quad [0 \sim 1]$
    - $loss(\{X_i\}) = \frac{1}{N} cross\_entropy(H_{Lo}(X_i), Y_i)$
  - How to classify X ?
    - Dataset 확장
      - $(X, Y_A), (X, Y_B), (X, Y_C)$
    - 3개의 가설함수(모델) 정의
      - $H^A_{LO}(X), H^B_{LO}(X), H^C_{LO}(X)$
    - 각 dataset으로 모델학습
    - 인식
      - 입력 X에 대하여 3개의 예측 값 계산 $\hat{y}_A, \hat{y}_B, \hat{y}_C$
      - $loss(X, Y) = ce(\hat{Y}_i, \mathrm{Y})$ 중 최대값 채택





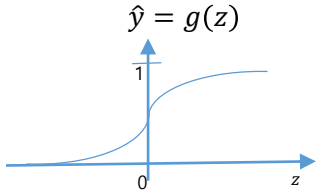
X		Y_A	Y_C	Y_C	Y
x1	x2				
2	4	1	0	0	A
3	3	1	0	0	A
9	5	0	1	0	B
10	5	0	1	0	B
11	1	0	0	1	C

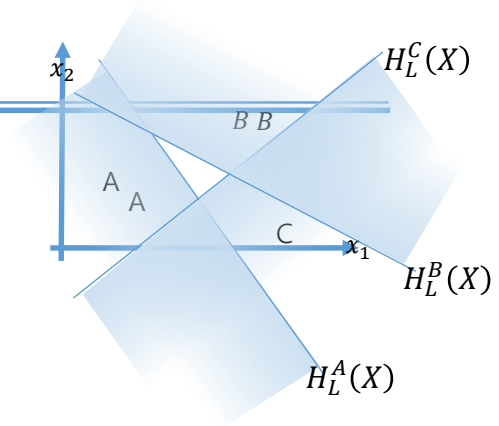
Let X=(10,5)

$$X \Rightarrow H^A_{LO}(X) \Rightarrow \bar{Y}^A \equiv 1.0$$
$$X \Rightarrow H^B_{LO}(X) \Rightarrow \bar{Y}^B \equiv 0.0$$
$$X \Rightarrow H^C_{LO}(X) \Rightarrow \bar{Y}^C \equiv 0.0$$

$\Rightarrow A$

1. How to classify 2 classes with logistic classifier?





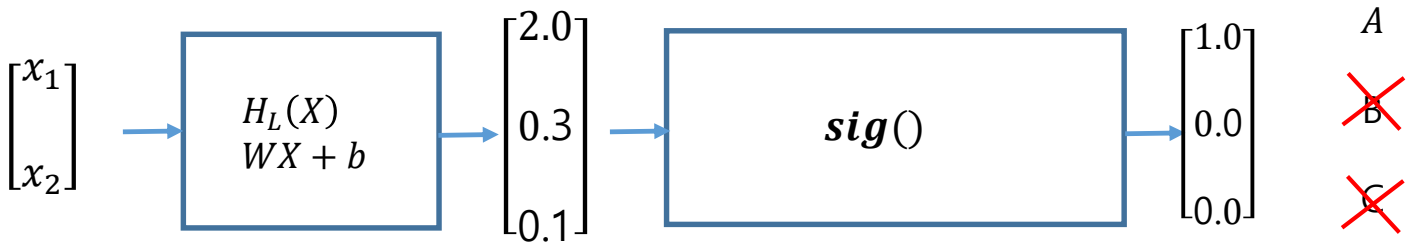
- How to classify 3 classes with logistic classifier?
  - How to predict(calculate Hypothesis function) ?
    - $WX + B = H_L(X)$
    - $sig\big(H(X)\big) = H_{Lo}\,(X) = \bar{Y}$

$$\begin{bmatrix} w_{A1}\ w_{A2} \\ w_{B1}\ w_{B2} \\ w_{C1}\ w_{C2} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + b = \begin{bmatrix} w_{A1}x_1 + w_{A2}x_2 + b \\ w_{B1}x_1 + w_{B2}x_2 + b \\ w_{C1}x_1 + w_{C2}x_2 + b \end{bmatrix} = \begin{bmatrix} H_L^A(X) \\ H_L^B(X) \\ H_L^C(X) \end{bmatrix} = \begin{bmatrix} 2.0 \\ 0.3 \\ 0.1 \end{bmatrix} => sig\left(\begin{bmatrix} H_L^A(X) \\ H_L^B(X) \\ H_L^C(X) \end{bmatrix}\right) = \begin{bmatrix} \overline{Y_A} \\ \overline{Y_B} \\ \overline{Y_C} \end{bmatrix} \equiv \begin{bmatrix} 1.0 \\ 0.0 \\ 0.0 \end{bmatrix} \Rightarrow \begin{matrix} A \\ B \\ C \end{matrix}$$

X
X

Linear Regression

sigmoid function



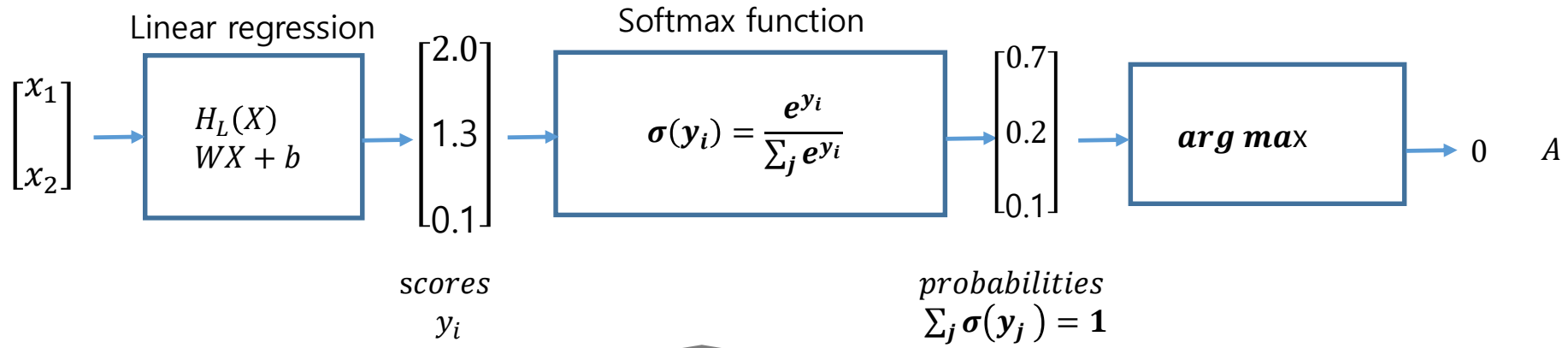
$H_L(X)$
$WX + b$

$LinearSVC()$ in sklearn

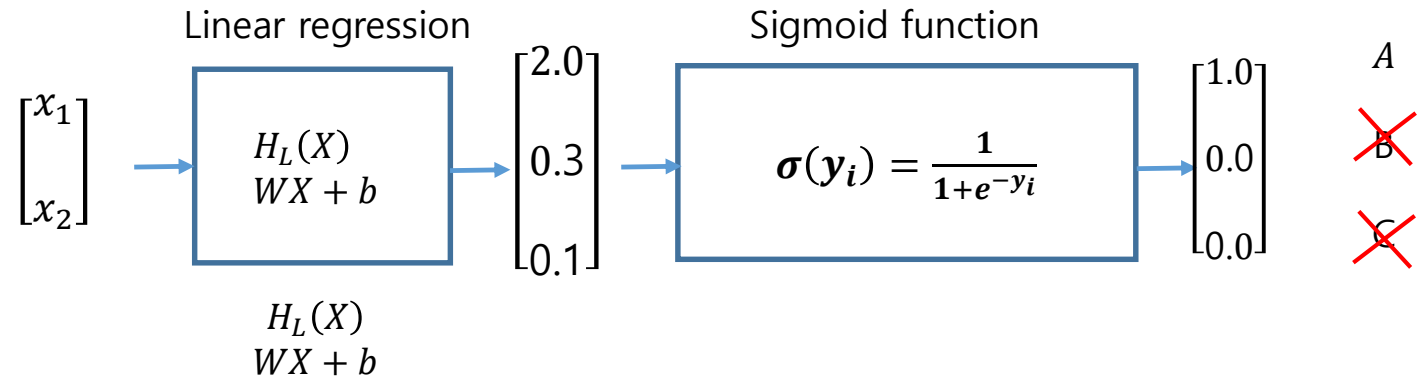
Is the sigmoid nessary in multinoomial classification ?

2. How to classify 3 classes with softmax classifier?

- How to classify 3 classes with softmax classifier?
  - How to predict on softmax prediction function ?



  - How to predict on logistic(sigmoid) prediction function ?



3. Mathematical interpretation of three models

Linear Regression

Linear Hypothesis function

$\mathrm{H_L(X)} = WX + b \equiv \bar{Y}$

$\bar{Y} = H_L(X)$

Loss function : mean square error function(mse)

$$\mathrm{loss}(X,Y) = \frac{1}{m}\sum_i (\bar{y_i} - y_i)^2$$

Gradient descent algorithm

$$W := W - \alpha \frac{\partial}{\partial W} loss(W|X,Y)$$

Logistic regression(Binary classification)

Sigmoid Hypothesis function

$\mathrm{H_L(X)} = WX + b$

$\mathrm{H_{Lo}(X)} = sig(H_L(x)), sig(x) = \frac{1}{a+e^{-x}}$

$\bar{Y} = H_{Lo}(X)$

Loss function : binary_crossentropy function

$$loss(X,Y) = -\frac{1}{m}\sum_{i=1}^{m}(ylog(\bar{y_i}) + (1-y)\log(1-\bar{y_i}))$$

Gradient descent algorithm

$$W := W - \alpha \frac{\partial}{\partial W} loss(W|X,Y)$$

Softmax classification

Softmax Hypothesis function

$\mathrm{H_L(X)} = WX + b$

$H_{So}(\mathrm{X}) = softmax(\mathrm{H_L(X)}), s(h_i) = \frac{e^{h_i}}{\sum_k e^{h_k}}$

$\bar{Y} = H_{So}(X)$

Loss function : catagorial_crossentropy

$$\mathrm{loss}(X,Y) = \frac{1}{N}\sum_{i=1}^{N} D\left(H\left(X^{(i)}\right), Y^{(i)}\right)$$

$$D(\bar{Y},Y) = -\sum_j y_j log(\bar{y_j})$$

Gradient descent algorithm

$$W := W - \alpha \frac{\partial}{\partial W} loss(W|X,Y)$$

4. Three classification models in Keras

```
#Linear Regression Model
X      =np.array([[1.1, 2.3], [2.0, 3.6]]) ;Y    =np.array([[10.25], [11.2]])
X_val=np.array([[1.3, 1.8]]);          ;        y_val=np.array([[10.2]])

model=Sequential()
model.add(Dense(1
                activation='linear'
                input_dim=2)
model.compile(
                loss='mse'
                optimizer='adam')

p=model.predict(np.array([[1.1, 1.7]]))#[[11.33]]
```

```
#Logistic regression(Binary classification) Model
X    =np.array([[1, 2], [2, 3]]) ;Y    =np.array([[0],[1]])
X_val=np.array([[1, 1]]);          ; y_val=np.array([[0]])

model=Sequential()
model.add(Dense(1,
   activation='sigmoid',
   input_dim=2
model.compile(
   loss='binary_crossentropy',
   optimizer='adam',
   metrics=['accuracy'])

y_hat     =model.predict(X)                                  #[[0.3] [0.4]]
Acc       =model.evaluate(X_val, y_val, verbose=0)[1]  #0.7
```

```
#Multinomial(softmax) classification
X          = np.array([[1, 2, 1, 4],  [2, 1, 3, 5],  [3, 1, 3, 1]])
Y_ohe    = np.array([[0, 0, 1],  [0, 1, 0],  [1, 0, 0]])

model=Sequential()
model.add(Dense(3,
   activation='softmax',
   input_dim=4)
model.compile(
   loss='categorial_crossentropy',
   optimizer='adam',
   metrics=['accuracy'])

Y_hat     = model.predict(np.array([[1,2,1,3]]) #[[0.0188113  0.15894766 0.822241  ]]
Acc   =model.evaluate(X_val, y_val, verbose=0)[1]  #0.7
```

Examples

- Example 1, Simple softmax classifier
- Example 2, Fancy animal classification

Example 1. Simple softmax classifier

- Simple softmax classifier
  - Simple dataset으로
    모델을 개발하고
    분류 성능을 평가하시오.

  - 데이터셋생성
  - 모델 구성
  - 모델 학습 방법 설정
  - 모델 학습하기
  - 모델 평가하기
  - 모델 사용하기

Example 1. Simple softmax classifier

- 문제
  - 다음의 data를 이용하여 3 class classifier 를 작성하라.
- 데이터셋 생성
  - Y(label) 데이터를 one_hot_encoding한다.
    - nb_class =3
    - 0 => 1 0 0
      1 => 0 1 0
      2 => 0 0 1
  - 행렬표현과 코딩

x1	x2	x3	x4	Y
1	2	1	1	A
2	1	3	2	A
3	1	3	4	A
4	1	5	5	B
1	7	5	5	B
1	2	5	6	B
1	6	6	6	C
1	7	7	7	C

x1	x2	x3	x4	Y	Y1	Y_ohe
1	2	1	1	A	0	1 0 0
2	1	3	2	A	0	1 0 0
3	1	3	4	A	0	1 0 0
4	1	5	5	B	1	0 1 0
1	7	5	5	B	1	0 1 0
1	2	5	6	B	1	0 1 0
1	6	6	6	C	2	0 0 1
1	7	7	7	C	2	0 0 1

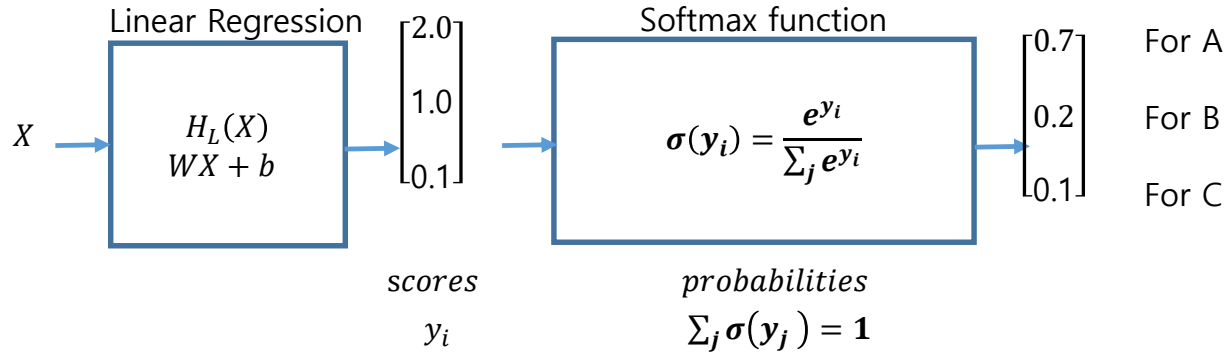
$$X=\begin{bmatrix}1,&2,&1,&1\\2,&1,&3,&2\\3,&1,&3,&4\\4,&1,&5,&5\\1,&7,&5,&5\\1,&2,&5,&6\\1,&6,&6,&6\\1,&7,&7,&7\end{bmatrix}\quad Y=\begin{bmatrix}0,&0,&1\\0,&0,&1\\0,&0,&1\\0,&1,&0\\0,&1,&0\\0,&1,&0\\1,&0,&0\\1,&0,&0\end{bmatrix}\quad \text{class}=\begin{bmatrix}A\\A\\A\\B\\B\\B\\C\\C\end{bmatrix}$$

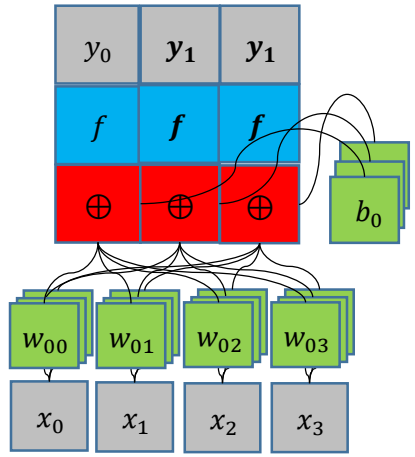
$$W=\begin{bmatrix}w_{11},w_{12},w_{13}\\w_{21},w_{22},w_{23}\\w_{31},w_{32},w_{33}\\w_{41},w_{42},w_{43}\end{bmatrix}\qquad B=[b_1,b_2,b_3]$$

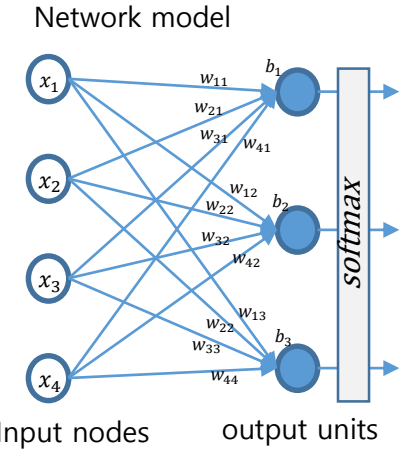
```
# 1. 데이터셋 생성하기
X = np.array([[1, 2, 1, 1],  [2, 1, 3, 2],  [3, 1, 3, 4],  [4, 1, 5, 5],  [1, 7, 5, 5], [1, 2, 5, 6],   [1, 6, 6, 6],  [1, 7, 7, 7]])
Y = np.array([[0, 0, 1],   [0, 0, 1],  [0, 0, 1],   [0, 1, 0],   [0, 1, 0],   [0, 1, 0],   [0, 0, 1],  [0, 0, 1]])
nb_classes=3
X,X_val,Y,Y_val=train_test_split(X,Y,random_state=0) #(6,4) (2,4)  (6,3) (2,3)
```

Example 1. Simple softmax classifier(cont.)

- 모델의 개념과 구성







$H_L(X) = WX + b$
$H_{so}(X) = softmax(H_L(X))$

X.shape[1]

```
# 2. 모델 구성하기
model=Sequential(name='Softmax_Sample_Classification')
model.add(Dense(units=3,input_dim=4,activation='softmax'))
```

```
X = np.array(                Y = np.array(
    [[1, 2, 1, 1],               [[0, 0, 1],
     [2, 1, 3, 2],                [0, 0, 1],
       …                             …
    ])                           ])
```

Example 1. Simple softmax classifier(cont.)

- 모델 학습방법 설정
  - Gradient Descent Optimization Algorithms [link]
  - loss function
    - Categorial cross entropy function
      - $\text{loss}(X,Y) = \frac{1}{N}\sum D(H(X),Y)$

        $= \frac{1}{N}\sum D(\bar{Y},Y) = -\frac{1}{N} Y log(\bar{Y}) = -\frac{1}{N}\sum_i y_i \log(\bar{y_i})$
  - optimizer
    - $W \coloneqq W - \alpha \frac{\partial}{\partial W}\mathcal{L}(W)$
    - Stochastic Gradient Descent (sgd)
  - Metrics : 학습과정에서 매 epoch마다 모델의 학습정도를 평가하는 방법으로 설정
    - Accuracy : 정확도

```
# 2. 모델 구성하기
model=Sequential(name='Softmax_Sample_Classification')
model.add(Dense(units=nb_classes,input_dim=X.shape[1],activation='softmax'))

# 3. 모델 학습방법 설정하기
model.compile(loss='categorical_crossentropy',optimizer='sgd', metrics=['accuracy'])
model.summary()
```

Example 1. Simple softmax classifier(cont.)

- 모델학습 시키기
  - 학습데이터 설정 : X,Y
  - 데이터의 반복학습 횟수 : epochs=1000
  - 학습정보 출력 량 verbose = 0,1,2
  - 매회 데이터 학습을 마치고 모델을 평가하여 학습되 정도를 확인하기 위하여 평가방법을 지정한다.
    validation=['accuracy']
- 모델평가하기
  - model.evaulate(X,Y)
    모델학습과정에서 지정한 평가방법으로 평가 결과를 반환한다.
- 모델사용하기
  - model.evaulate(X[0:1])
    데이터를 입력하여 모델이 예측한 값을 반환 한다.

```
# 4. 모델 학습시키기
hist=model.fit(X,Y,epochs=200,verbose=1,validation_data=(X_val, Y_val))#,batch_size=len(X))

# 5. 학습과정 살펴보기
print('\nfitted-history :')
print('\tacc_max:{:.2f},\tval_acc_max:{:.2f},\tloss_min:{:.6f},\tval_loss_min:{:.6f}'.format(
max(hist.history['accuracy']),max(hist.history['val_accuracy']),
min(hist.history['loss']),max(hist.history['val_loss'])))

# 6. 모델 평가하기
print('\nmodel.evaluate(X_val, Y_val): ')
loss_and_acc = model.evaluate(X_val, Y_val,verbose=0)
print('\tloss and_acc :', loss_and_acc)

# 7. 모델 사용하기
y_hat = model.predict(X[0:1])
print('\ny_hat=model.predict(X[0:1])')
print('\tX[0:1] : ',X[0:1])
print('\ty_hat:   ',y_hat)
print('\targamx(yhat) : ',np.argmax(y_hat,axis=1))

yhat = model.predict(X[1:4])
print('\ny_hat=nmodel.predict(X[1:4])')
print('\tX[1:4] : ',X[1:4])
print('\ty_hat       : ',y_hat)
print('\targamx(yhat) : ',np.argmax(y_hat,axis=1))
```

Example 1. Simple softmax classifier(cont.)

```
from keras.models import Sequential
from keras.layers import Dense
from sklearn.model_selection import train_test_split
import numpy as np

# 1. 데이터셋 생성하기
X = np.array([[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 = np.array([[0, 0, 1],   [0, 0, 1],  [0, 0, 1],   [0, 1, 0],   [0, 1, 0],   [0, 1, 0],   [0, 0, 1],  [0, 0, 1]])
nb_classes=3
X,X_val,Y,Y_val=train_test_split(X,Y,random_state=0)
#(6,4) (2,4)  (6,3) (2,3)

# 2. 모델 구성하기
model=Sequential(name='Softmax_Sample_Classification')
model.add(Dense(units=nb_classes,input_dim=X.shape[1],activation='softmax'))

# 3. 모델 학습방법 설정하기
model.compile(loss='categorical_crossentropy',optimizer='sgd', metrics=['accuracy'])
model.summary()

# 4. 모델 학습시키기
hist=model.fit(X,Y,epochs=200,verbose=1,validation_data=(X_val, Y_val))#,batch_size=len(X))

# 5. 학습과정 살펴보기
print('\nfitted-history :')
print('\tacc_max:{:.2f},\tval_acc_max:{:.2f},\tloss_min:{:.6f},\tval_loss_min:{:.6f}'.format(
max(hist.history['accuracy']),max(hist.history['val_accuracy']),
min(hist.history['loss']),max(hist.history['val_loss'])))
```

```
# 6. 모델 평가하기
print('\nmodel.evaluate(X_val, Y_val): ')
loss_and_acc = model.evaluate(X_val, Y_val,verbose=0)
print('\tloss and_acc :', loss_and_acc)

# 7. 모델 사용하기
y_hat = model.predict(X[0:1])
print('\ny_hat=model.predict(X[0:1])')
print('\tX[0:1] : ',X[0:1])
print('\ty_hat:   ',y_hat)
print('\targamx(yhat) : ',np.argmax(y_hat,axis=1))

yhat = model.predict(X[1:4])
print('\ny_hat=nmodel.predict(X[1:4])')
print('\tX[1:4] : ',X[1:4])
print('\ty_hat       : ',y_hat)
print('\targamx(yhat) : ',np.argmax(y_hat,axis=1))
```

Example 1. Simple softmax clas

```
from keras.models import Sequential
from keras.layers import Dense
from sklearn.model_selection import train_test_split
import numpy as np

# 1. 데이터셋 생성하기
X = np.array([[1, 2, 1, 1],  [2, 1, 3, 2],  [3, 1, 3, 4],  [4, 1, 5, 5],  [1,
Y = np.array([[0, 0, 1],   [0, 0, 1],  [0, 0, 1],   [0, 1, 0],   [0, 1, 0],   [
nb_classes=3
X,X_val,Y,Y_val=train_test_split(X,Y,random_state=0)
#(6,4) (2,4)  (6,3) (2,3)
```

```
# 6. 모델 평가하기
print('\nmodel.evaluate(X_val, Y_val): ')
loss_and_acc = model.evaluate(X_val, Y_val,verbose=0)
print('\tloss and_acc :', loss_and_acc)

# 7. 모델 사용하기
y_hat = model.predict(X[0:1])
print('\ny_hat=model.predict(X[0:1])')
print('\tX[0:1] : ',X[0:1])
print('\ty_hat:   ',y_hat)
print('\targamx(yhat) : ',np.argmax(y_hat,axis=1))

yhat = model.predict(X[1:4])
print('\ny_hat=nmodel.predict(X[1:4])')
print('\tX[1:4] : ',X[1:4])
print('\ty_hat       : ',y_hat)
print('\targamx(yhat) : ',np.argmax(y_hat,axis=1))
```

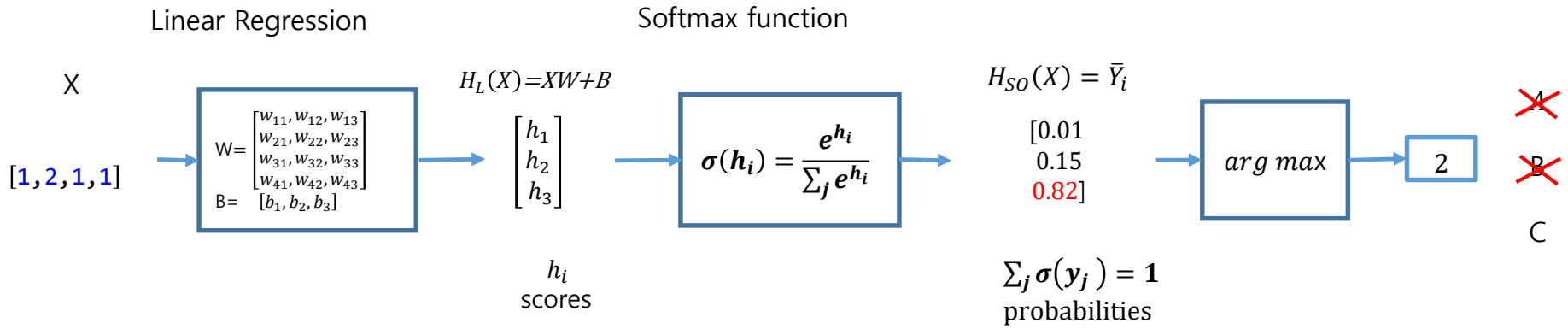
```
Model: "Softmax_Sample_Classification"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 3)                 15
=================================================================
Total params: 15
Trainable params: 15
Non-trainable params: 0
_________________________________________________________________

Train on 6 samples, validate on 2 samples
Epoch 1/200
6/6 [==============================] - 0s 65ms/step - loss: 10.4271 - accuracy: 0.0000e+00 - val_loss: 4.5422 - val_accuracy:
0.0000e+00
Epoch 2/200
6/6 [==============================] - 0s 8ms/step - loss: 9.6818 - accuracy: 0.0000e+00 - val_loss: 3.8110 - val_accuracy:
0.0000e+00
Epoch 3/200

Epoch 199/200
6/6 [==============================] - 0s 1ms/step - loss: 0.5661 - accuracy: 0.6667 - val_loss: 1.4009 - val_accuracy: 0.5000
Epoch 200/200
6/6 [==============================] - 0s 28ms/step - loss: 0.5656 - accuracy: 0.6667 - val_loss: 1.4003 - val_accuracy: 0.5000

fitted-history :
      acc_max:0.67,   val_acc_max:1.00,      loss_min:0.565597,     val_loss_min:4.542212

model.evaluate(X_val, Y_val):
      loss and_acc : [1.4003145694732666, 0.5]

y_hat=model.predict(X[0:1])
      X[0:1] :  [[2 1 3 2]]
      y_hat   :  [[0.01586246 0.46529737 0.5188402 ]]
      argamx(yhat) :  [2]

y_hat=nmodel.predict(X[1:4])
      X[1:4] :  [[1 7 7 7]
 [4 1 5 5]
 [1 2 1 1]]
 y_hat   :     [[1.0841307e-03       5.3002125e-01        4.6889463e-01]
                [1.1496942e-04       1.8826017e-01        8.1082493e-01]
                [1.2-05         5.1698810e-01 4.8297709e-01]]
 argamx(yhat) :  [1 2  1]
```

```python
# 1. 데이터셋 생성하기
X = np.array([[1, 2, 1, 1],  [2, 1, 3, 2],  [3, 1, 3, 4],  [4, 1, 5, 5],  [1, 7, 5, 5], [1, 2, 5, 6],
Y = np.array([[0, 0, 1],   [0, 0, 1],  [0, 0, 1],   [0, 1, 0],   [0, 1, 0],   [0, 1, 0],   [0, 0, 1],  [0
nb_classes=3
X,X_val,Y,Y_val=train_test_split(X,Y,random_state=0)

# 2. 모델 구성하기
model=Sequential(name='Softmax_Sample_Classification')
model.add(Dense(units=nb_classes,input_dim=X.shape[1],activation='softmax'))

# 3. 모델 학습방법 설정하기
model.compile(loss='categorical_crossentropy',optimizer='sgd', metrics=['accuracy'])
model.summary()

# 4. 모델 학습시키기
hist=model.fit(X,Y,epochs=200,verbose=1,validation_data=(X_val, Y_val))#,batch_siz

# 5. 학습과정 살펴보기
print('\nfitted-history :')
print('\tacc_max:{:.2f},\tval_acc_max:{:.2f},\tloss_min:{:.6f},\tval_loss_min:{:.6f}'.for
max(hist.history['accuracy']),max(hist.history['val_accuracy']),
min(hist.history['loss']),max(hist.history['val_loss'])))

# 6. 모델 평가하기
print('\nmodel.evaluate(X_val, Y_val): ')
loss_and_acc = model.evaluate(X_val, Y_val,verbose=0)
print('\tloss and_acc :', loss_and_acc)

# 7. 모델 사용하기
y_hat = model.predict(X[0:1])
print('\ny_hat=model.predict(X[0:1])')
print('\tX[0:1]    : ',X[0:1])
print('\ty_hat:     ',y_hat)
print('\targamx(yhat) : ',np.argmax(y_hat,axis=1))

yhat = model.predict(X[1:4])
print('\ny_hat=nmodel.predict(X[1:4])')
print('\tX[1:4]    : ',X[1:4])
print('\ty_hat     : ',y_hat)
print('\targamx(yhat) : ',np.argmax(y_hat,axis=1))
```

```
Model: "Softmax_Sample_Classification"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 3)                 15
=================================================================
Total params: 15
Trainable params: 15
Non-trainable params: 0
_________________________________________________________________

Train on 6 samples, validate on 2 samples
Epoch 1/200
6/6 [==============================] - 0s 65ms/step - loss: 10.4271 - accuracy: 0.0000e+00 - val_loss: 4.5422 - val_accuracy:
0.0000e+00
Epoch 2/200
6/6 [==============================] - 0s 8ms/step - loss: 9.6818 - accuracy: 0.0000e+00 - val_loss: 3.8110 - val_accuracy:
0.0000e+00
Epoch 3/200

Epoch 199/200
6/6 [==============================] - 0s 1ms/step - loss: 0.5661 - accuracy: 0.6667 - val_loss: 1.4009 - val_accuracy: 0.5000
Epoch 200/200
6/6 [==============================] - 0s 28ms/step - loss: 0.5656 - accuracy: 0.6667 - val_loss: 1.4003 - val_accuracy: 0.5000

fitted-history :
	acc_max:0.67,	val_acc_max:1.00,	loss_min:0.565597,	val_loss_min:4.542212

model.evaluate(X_val, Y_val):
	loss and_acc : [1.4003145694732666, 0.5]

y_hat=model.predict(X[0:1])
	X[0:1]  :  [[2 1 3 2]]
	y_hat   :  [[0.01586246 0.46529737 0.5188402 ]]
	argamx(yhat) :  [2]

y_hat=nmodel.predict(X[1:4])
	X[1:4]  :  [[1 7 7 7]
 [4 1 5 5]
 [1 2 1 1]]
 y_hat   :     [[1.0841307e-03       5.3002125e-01        4.6889463e-01]
                [1.1496942e-04       1.8826017e-01        8.1082493e-01]
                [1.2-05         5.1698810e-01 4.8297709e-01]]
 argamx(yhat) :  [1 2  1]
```

Example 1. Simple softmax classifier(cont.)

- 모델의 예측처리 개념
  - $\bar{Y}_{SO}$ = model.predict([1, 2, 1, 1])?



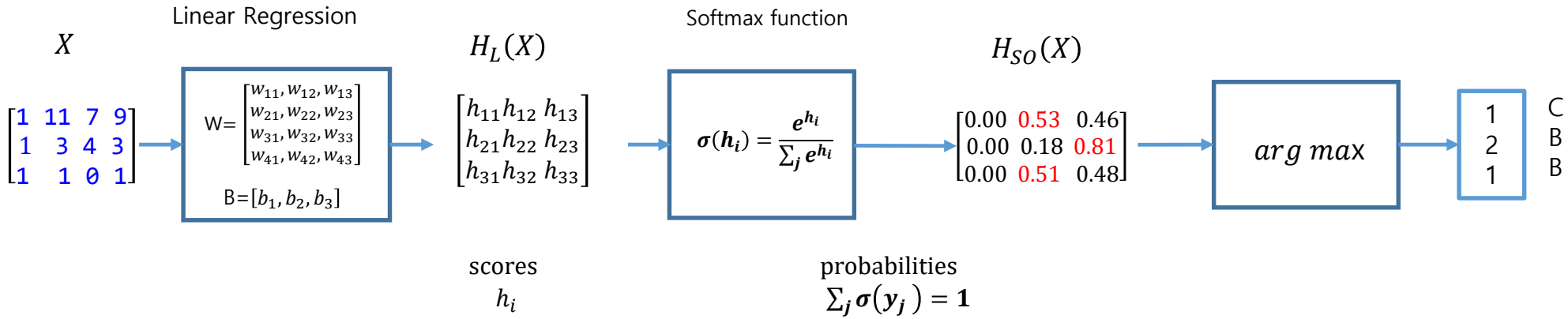
$\bar{Y}_{SO} = H_{SO}([1, 2, 1, 1])$
=model.predict(X[:1])
=model.predict([1, 2, 1, 1])
= Sofatmax(H_{SO}([1, 2, 1, 1]))
=Sofatmax(H_{SO}([[h_1 h_2 h_3])
=[0.0188113 0.15894766 0.822241]
$\bar{Y}$ = np.argmax($\bar{Y}_{SO}$,axis=1)=[2]

```
# 7. 모델 사용하기
yhat = model.predict(X[0:1])  #[[1, 2, 1, 1]]
print('X[0:1] : ',X[0:1])
print('yhat  = H(X[:1])     = {}'.format(yhat))
print('yhat1 = argamx(yhat) = {}'.format(np.argmax(yhat)))
```

```
X[0:1] :  [[1 2 1 1]]
yhat     = H(X[0:1])     = [[0.0188113  0.15894766 0.822241  ]]
yhat1    = argamx(yhat,1) = 2
```

Example 1. Simple softmax classifier(cont.)

- 모델의 예측처리 개념
  - $\bar{Y}_{So}$ =model.predict(X[:3])= ?



$\bar{Y} = H_{SO}(X[:3])$
=model.predict(X[:3])
=model.oredict [[2 1 3 2]
[3 1 3 4]
[4 1 5 5]])
= [[1.0841307e-03 5.3002125e-01 4.6889463e-01]
[3.4792713e-05 1.8826017e-01 8.1082493e-01]
[1.2-05 5.1698810e-01 4.8297709e-01]]

np.argmax($\bar{Y}$,axis=1)=[1,2,1]

```
yhat = model.predict(X[1:4])
print('X[1:4] : ',X[1:4])
print('yhat  = H(X[1:4])    = {}'.format(yhat))
print('yhat1 = argamx(yhat) = {}'.format(np.argmax(yhat,axis=1)))
```

```
X[1:4]  :     [[2 1 3 2]
               [3 1 3 4]
               [4 1 5 5]]
yhat  = H(X[1:4])    =    [[1.0841307e-03 5.3002125e-01 4.6889463e-01]
                           [3.4792713e-05 .1496942e-04 1.8826017e-01 8.1082493e-01]
                           [1.2-05 5.1698810e-01 4.8297709e-01]]
yhat1 = argamx(yhat ,axis=1)= [2 1 1]
```

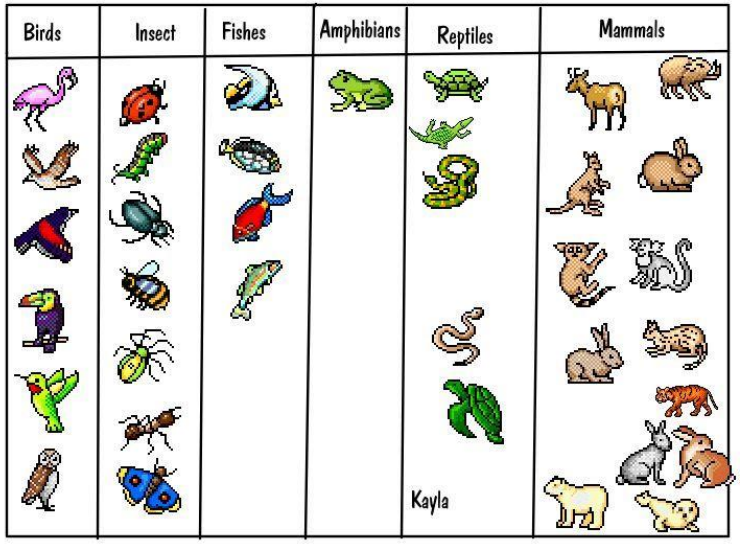
Example 2. Fancy animal classification

- 문제
  - 동물의 17가지 특성을 분석하여 종을 판별하는 softmax classifier를 작성하기

  - 데이터(data-04-zoo.csv)를 다운로드하고
  - 모델을 개발하고
  - 분류실험을 하여 개발한 모델의 분류성능을 분석하시오.

- Dataset 생성
  - 데이터셋 다운로드
    - DeepLearningZeroToAll
      - data-04-zoo.csv

Example 2. Fancy animal classification



- Data-04-zoo.txt의 분석

```
# https://archive.ics.uci.edu/ml/machine-learning-databases/zoo/zoo.data,,,,,,,,,,,,,,,
#   1. animal name:      (deleted),,,,,,,,,,,,,,,
#   2. hair      Boolean",,,,,,,,,,,,,,,
#   3. feathers      Boolean",,,,,,,,,,,,,,,
#   4. eggs      Boolean",,,,,,,,,,,,,,,
#   5. milk      Boolean",,,,,,,,,,,,,,,
#   6. airborne      Boolean",,,,,,,,,,,,,,,
#   7. aquatic      Boolean",,,,,,,,,,,,,,,
#   8. predator      Boolean",,,,,,,,,,,,,,,
#   9. toothed      Boolean",,,,,,,,,,,,,,,
#  10. backbone      Boolean",,,,,,,,,,,,,,,
#  11. breathes      Boolean",,,,,,,,,,,,,,,
#  12. venomous      Boolean",,,,,,,,,,,,,,,
#  13. fins      Boolean",,,,,,,,,,,,,,,
#  14. legs      Numeric (set of values: {0",2,4,5,6,8}),,,,,,,,,,
#  15. tail      Boolean",,,,,,,,,,,,,,,
#  16. domestic      Boolean",,,,,,,,,,,,,,,
#  17. catsize      Boolean",,,,,,,,,,,,,,,
#  18. type      Numeric (integer values in range [0",6]),,,,,,,,,,,,,,,
1,0,0,1,0,0,1,1,1,1,0,0,4,0,0,1,0
1,0,0,1,0,0,0,1,1,1,0,0,4,1,0,1,0
0,0,1,0,0,1,1,1,1,0,0,1,0,1,0,0,3
1,0,0,1,0,0,1,1,1,1,0,0,4,0,0,1,0
1,0,0,1,0,0,1,1,1,1,0,0,4,1,0,1,0
...
```

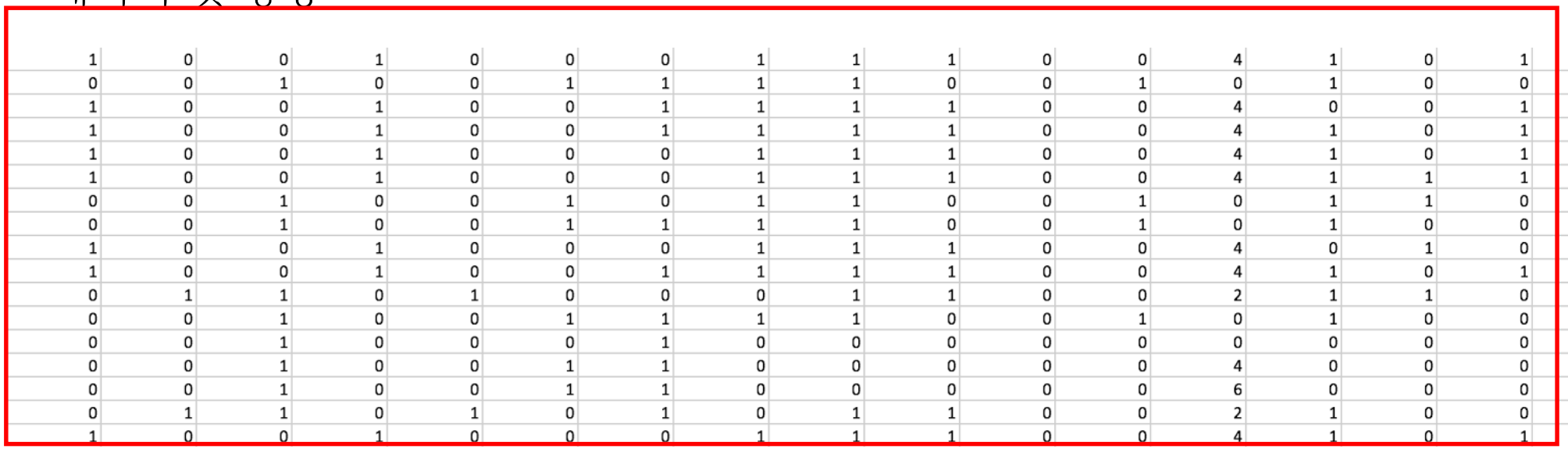
1	0	0	1	0	0	0	1	1	1	0	0	4	1	0	1	0
0	0	1	0	0	1	1	1	1	1	0	0	1	0	1	0	0	3
1	0	0	1	0	0	0	1	1	1	1	0	0	4	0	0	1	0
1	0	0	1	0	0	0	1	1	1	1	0	0	4	1	0	1	0
1	0	0	1	0	0	0	0	1	1	1	0	0	4	1	0	1	0
1	0	0	1	0	0	0	0	1	1	1	0	0	4	1	1	1	0
0	0	1	0	0	1	0	1	1	1	0	0	1	0	1	1	0	3
0	0	1	0	0	1	1	1	1	1	0	0	1	0	1	0	0	3
1	0	0	1	0	0	0	1	1	1	1	0	0	4	0	1	0	0
1	0	0	1	0	0	1	1	1	1	1	0	0	4	1	0	1	0
0	1	1	0	1	0	0	0	1	1	1	0	0	2	1	1	0	1
0	0	1	0	0	1	1	1	1	1	1	0	0	1	0	1	0	0	3
0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	6
0	0	1	0	0	1	1	0	0	0	0	0	4	0	0	0	0	6
0	0	1	0	0	1	1	0	0	0	0	0	6	0	0	0	0	6
0	1	1	0	1	0	1	0	1	1	1	0	0	2	1	0	0	1
1	0	0	1	0	0	0	1	1	1	0	0	4	1	0	1	0

X Y

```
XY=np.loadtxt('data/data-04-zoo-1.csv',delimiter=',',dtype='float')  #(101,17)
X=XY[:,:-1]                  #(101,16)
Y=XY[:,[-1]].astype(int) #(101,1)
```

Example 2. Fancy animal classification

- 데이터 셋 생성



1	0	0	1	0	0	0	1	1	1	0	0	4	1	0	1
0	0	1	0	0	1	1	1	1	0	0	1	0	1	0	0
1	0	0	1	0	0	1	1	1	1	0	0	4	0	0	1
1	0	0	1	0	0	1	1	1	1	0	0	4	1	0	1
1	0	0	1	0	0	0	1	1	1	0	0	4	1	0	1
1	0	0	1	0	0	0	1	1	1	0	0	4	1	1	1
0	0	1	0	0	1	0	1	1	0	0	1	0	1	1	0
0	0	1	0	0	1	1	1	1	0	0	1	0	1	0	0
1	0	0	1	0	0	0	1	1	1	0	0	4	0	1	0
1	0	0	1	0	0	1	1	1	1	0	0	4	1	0	1
0	1	1	0	1	0	0	0	1	1	0	0	2	1	1	0
0	0	1	0	0	1	1	1	1	0	0	1	0	1	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	1	0	0	1	1	0	0	0	0	0	4	0	0	0
0	0	1	0	0	1	1	0	0	0	0	0	6	0	0	0
0	1	1	0	1	0	1	0	1	1	0	0	2	1	0	0
1	0	0	1	0	0	0	1	1	1	0	0	4	1	0	1

Y
0
3
0
0
0
0
3
3
0
0
1
3
6
6
6
1
0

Y_ohe

```
1. 0. 0. 0. 0. 0. 0
0. 0. 0. 1  0. 0. 0.
1. 0. 0. 0. 0. 0. 0.
1. 0. 0. 0. 0. 0. 0.
1. 0. 0. 0. 0. 0. 0.
1. 0. 0. 0. 0. 0. 0.
0. 0. 0. 1. 0. 0. 0.
0. 0. 0. 1. 0. 0. 0.
1. 0. 0. 0. 0. 0. 0.
1. 0. 0. 0. 0. 0. 0.
0. 1. 0. 0. 0. 0. 0.
0. 0. 0. 1. 0. 0. 0.
0. 0. 0. 0. 0. 0. 1.
0. 0. 0. 0. 0. 0. 1.
0. 0. 0. 0. 0. 0. 1.
.....
```

```
# 1. 데이터셋 생성하기
XY=np.loadtxt('data/data-04-zoo-1.csv',delimiter=',',dtype='float')  #(101,17)
X=XY[:,:-1]                  #(101,16)
Y=XY[:,[-1]].astype(int)  #(101,1)

nb_classes = np.unique(Y).size      #7
Y_ohe=np_utils.to_categorical(Y,nb_classes)    #(101,7)
#Y_ohe=np.eye(nb_classes)[Y.flatten()]          #(101,7)

X,X_val,Y,Y_val=train_test_split(X,Y_ohe,random_state=0) #(75,16) (26,16) (75,7) (26,7)
```

```
from keras.models import Sequential
from keras.layers import Dense
from sklearn.model_selection import train_test_split
from keras.utils import np_utils
import numpy as np
```

```
Y=[[0],[3],[0],...]
Y_ohe
  =[[1,0,0,0,0,0,0],
    [0,0,0,1,0,0,0]
    [1,0,0,0,0,0,0]
     ....]
```

Example 2. Fancy anima

- 데이터셋 생성
- 모델 구성
- 모델학습방법 설정
- 모델학습
- 학습과정 분석
- 모델평가하기
- 모델사용하기

```
Model: "Softmax_Fancy_Animal_Classification"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 7)                 119
=================================================================
Total params: 119
Trainable params: 119
Non-trainable params: 0
_________________________________________________________________

Train on 75 samples, validate on 26 samples
Epoch 1/200
75/75 [==============================] - 0s 4ms/step - loss: 1.8031 - accuracy: 0.4267 - val_loss: 1.7999 - val_accuracy: 0.4231
Epoch 2/200
75/75 [==============================] - 0s 2ms/step - loss: 1.7656 - accuracy: 0.4000 - val_loss: 1.7715 - val_accuracy: 0.4231
Epoch 3/200
75/75 [==============================] - 0s 292us/step - loss: 1.7372 - accuracy: 0.4000 - val_loss: 1.7407 - val_accuracy: 0.4231

Epoch 199/200
75/75 [==============================] - 0s 3ms/step - loss: 0.6282 - accuracy: 0.8667 - val_loss: 0.6113 - val_accuracy: 0.8846
Epoch 200/200
75/75 [==============================] - 0s 1ms/step - loss: 0.6268 - accuracy: 0.8667 - val_loss: 0.6098 - val_accuracy: 0.8846

fitted-history :
      acc_max:0.87,   val_acc_max:0.88,      loss_min:0.626761,      val_loss_min:1.799907

model.evaluate(X_val, Y_val):
      loss and_acc : [0.6097744703292847, 0.8846153616905212]

y_hat=model.predict(X[0:1])
      X[0:1] :  [[1. 0. 0. 1. 0. 0. 0. 1. 1. 1. 0. 0. 4. 1. 1. 1.]]
      y_hat  :  [[0.9215614  0.02619649 0.0063446  0.00196954 0.00482013 0.02501752  0.01409036]]
      argamx(yhat) :  [0]

y_hat=nmodel.predict(X[1:4])
      X[1:4] :  [[0. 1. 1. 0. 1. 0. 0. 0. 1. 1. 0. 0. 2. 1. 0. 0.]
                 [0. 0. 1. 0. 0. 1. 0. 1. 1. 0. 0. 1. 0. 1. 0. 0.]
                 [0. 0. 0. 0. 0. 1. 1. 1. 1. 0. 1. 0. 0. 1. 0. 0.]]
      y_hat  :  [[0.9215614  0.02619649 0.0063446  0.00196954 0.00482013 0.02501752   0.01409036]]
      argamx(yhat) :  [0]
```

The End