Multiclass Classification

Dr. Dongchul Kim

Binary Class

- There are only two classes in binary class data

x_1	x_2	...	x_m	y
				0
				1
				0

Multiclass

- More than two classes.

x_1	x_2	...	x_m	y
				0
				1
				2
				0
				2

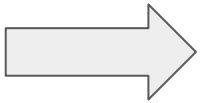
For examples,

- Iris data
  - https://archive.ics.uci.edu/ml/machine-learning-databases/iris/
  - **3 classes**
  - 4 features
  - 150 samples (50 samples per class)
- MNIST data
  - http://yann.lecun.com/exdb/mnist/
  - **10 classes**
  - 28x28 features
  - 60K training samples and 10K test samples

One-Hot Encoding

- Convert the class column to binary code (One-Hot label) like below.

y
0
1
2
0
2
1

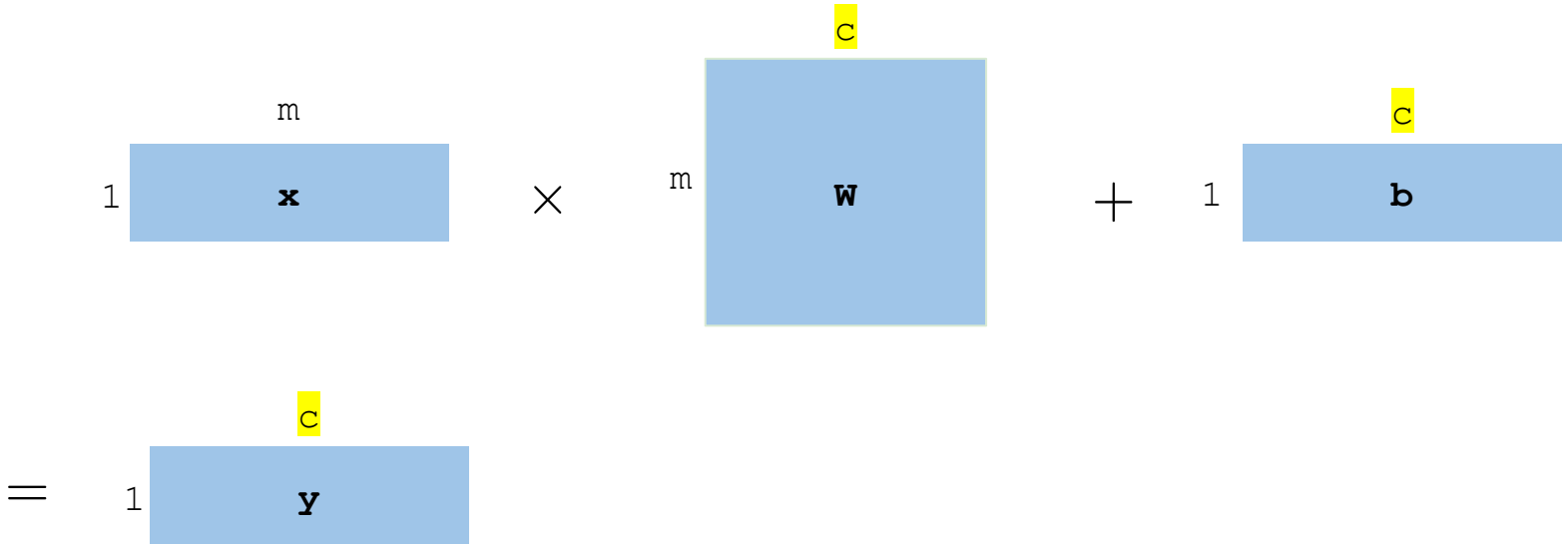


Class 0	Class 1	Class 2
1	0	0
0	1	0
0	0	1
1	0	0
0	0	1
0	1	0

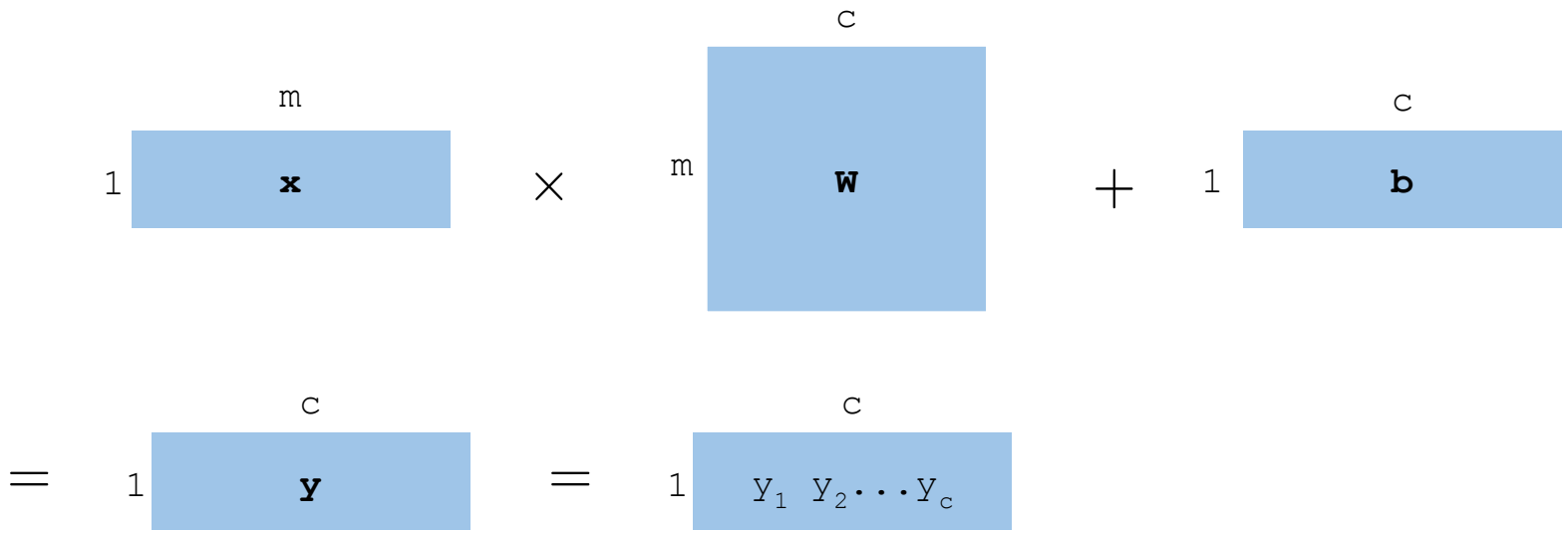
Linear Classifier

Linear Multiclass Classification

- `c` is # of classes.
- `m` is # of features.
- `maxIndex()` is a function that returns a index of element which is the max element given a vector.



Linear Multiclass Classification



Linear_Classifier(**x**) = maxIndex(**y**)

Linear Multiclass Classification

```
trainX = np.array([[0.1, 0.2, 0.3, 0.2],
                   [0.5, 0.4, 0.3, 0.7],
                   [0.3, 0.7, 0.4, 0.1],
                   [0.2, 0.8, 0.9, 0.3],
                   [1.1, 0.5, 0.2, 0.9],
                   [4.3, 5.3, 4.7, 4.2],
                   [4.5, 5.1, 5.3, 4.4],
                   [5.1, 4.8, 5.1, 4.6],
                   [4.9, 4.6, 4.9, 4.3],
                   [5.4, 5.5, 4.3, 4.7],
                   [10.1, 10.2, 10.3, 11.3],
                   [11.3, 11.2, 11.1, 10.3],
                   [12.5, 12.3, 12.1, 11.4],
                   [11.7, 11.8, 11.2, 12.8],
                   [13.1, 10.2, 12.4, 11.7]])
trainY = np.array([[1, 0, 0],
                   [1, 0, 0],
                   [1, 0, 0],
                   [1, 0, 0],
                   [1, 0, 0],
                   [0, 1, 0],
                   [0, 1, 0],
                   [0, 1, 0],
                   [0, 1, 0],
                   [0, 1, 0],
                   [0, 0, 1],
                   [0, 0, 1],
                   [0, 0, 1],
                   [0, 0, 1],
                   [0, 0, 1]])
```

```
testX = np.array([[0.5, 0.4, 0.6, 0.5],
                  [5.4, 5.6, 5.5, 5.2],
                  [11.7, 11.6, 11.5, 11.4]])
testY = np.array([[1, 0, 0],
                  [0, 1, 0],
                  [0, 0, 1]])
W = np.zeros((4, 3))
B = np.zeros(3)
alpha = 0.005
N = trainX.shape[0]
c = W.shape[1]
```

```
for i in range(1000):
    for j in range(c):
        W[:, j] = W[:, j] - alpha * (1/N) * np.dot(np.transpose(np.dot(trainX, W[:, j])+B[j] - trainY[:, j]), trainX)
        B[j] = B[j] - alpha * (1/N) * sum(np.dot(trainX, W[:, j])+B[j] - trainY[:, j])
print(W, B)
print(sum(np.argmax(np.dot(testX, W) + B, axis=1) == np.argmax(testY, axis=1))/testX.shape[0])
```

```
[[ 0.00136612 -0.03102624  0.01597588]
 [-0.07523478  0.2177921  -0.06253688]
 [-0.05620483  0.05216062  0.00881165]
 [ 0.05690523 -0.23662129  0.12713622]] [ 0.69839576  0.2862392  -0.14555836]
1.0
```

Linear classifier for multiclass

Please be aware that this multiclass classifier is founded on a linear function and MSE (cost function). In the realm of multiclass classification, we typically use the softmax function and cross-entropy. We'll delve into this topic and learn more about them in our upcoming lecture on neural networks.