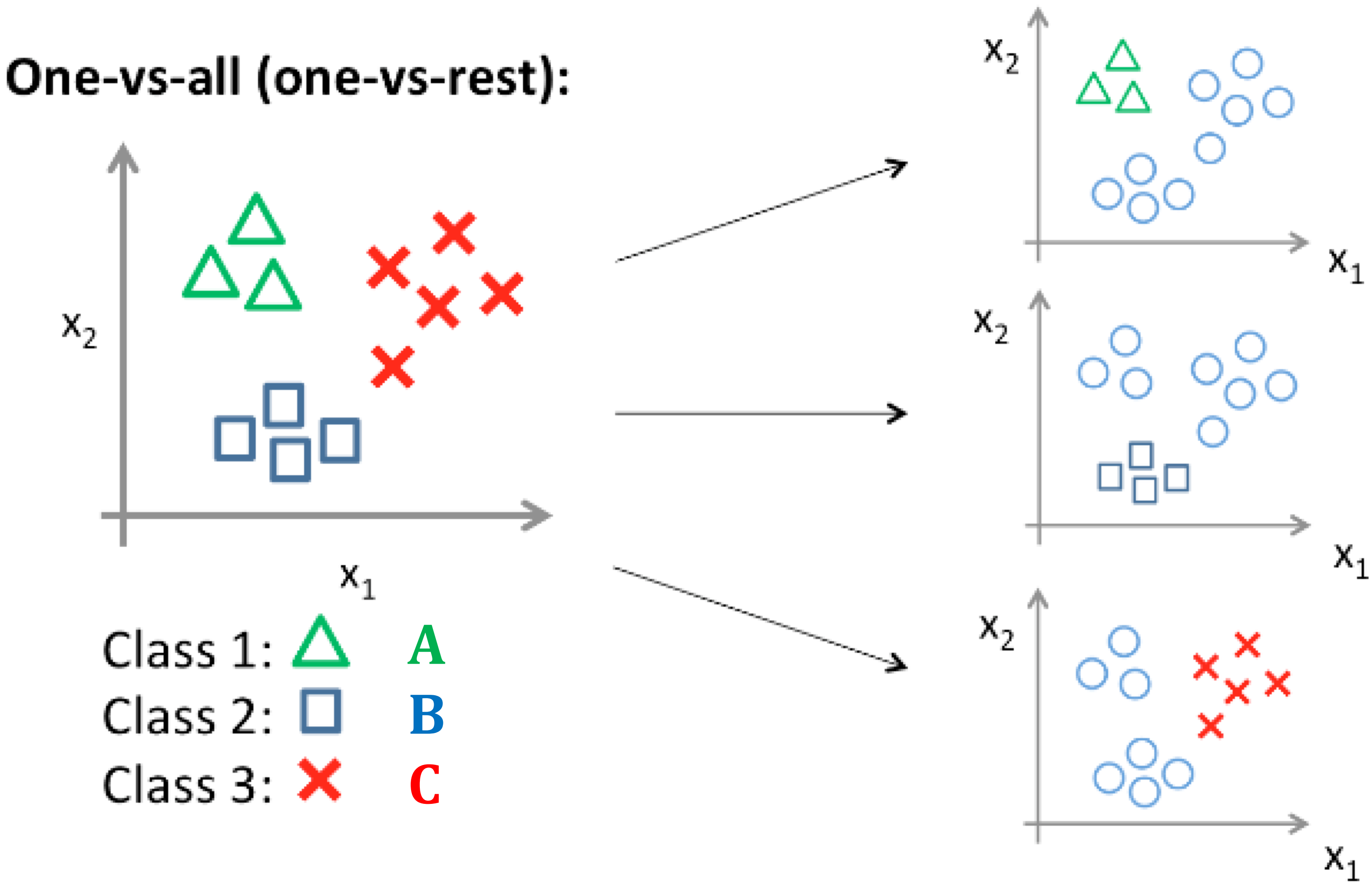
## Machine Learning

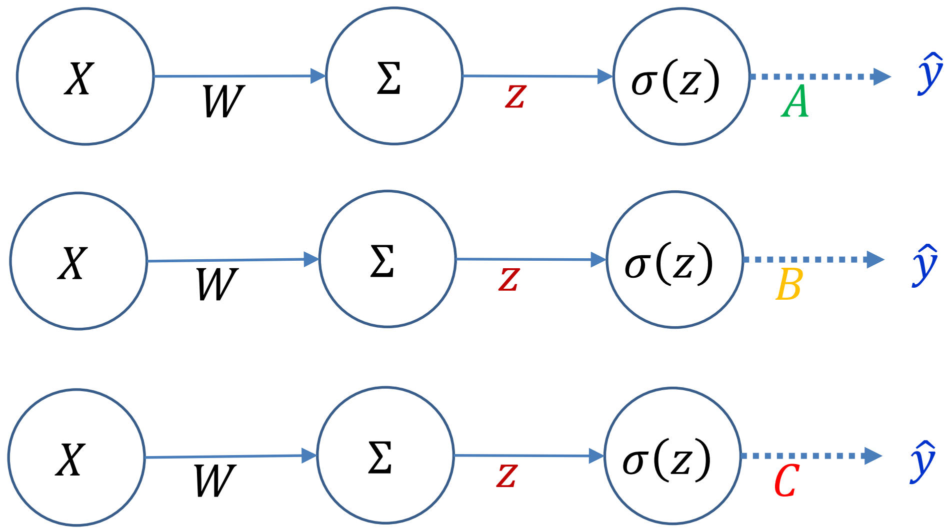
# Multinomial Classification

김선녕(ksycafe@gmail.com)

# 소프트맥스 회귀(다항 로지스틱 회귀)








$$\begin{bmatrix} x_{A1} & x_{A2} & x_{A3} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} = \begin{bmatrix} x_{A1}w_1 + x_{A2}w_2 + x_{A3}w_3 \end{bmatrix}$$

$$\begin{bmatrix} x_{B1} & x_{B2} & x_{B3} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} = \begin{bmatrix} x_{B1}w_1 + x_{B2}w_2 + x_{B3}w_3 \end{bmatrix}$$

$$\begin{bmatrix} x_{C1} & x_{C2} & x_{c3} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} = \begin{bmatrix} x_{C1}w_1 + x_{C2}w_2 + x_{C3}w_3 \end{bmatrix}$$

$$\begin{bmatrix} x_{A1} & x_{A2} & x_{A3} \\ x_{B1} & x_{B2} & x_{B3} \\ x_{C1} & x_{C2} & x_{C3} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} = [\,?\,]$$

$$\begin{bmatrix} x_{A1} & x_{A2} & x_{A3} \\ x_{B1} & x_{B2} & x_{B3} \\ x_{C1} & x_{C2} & x_{C3} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} = [\,?\,]$$

$$\begin{bmatrix} x_{A1} & x_{A2} & x_{A3} \\ x_{B1} & x_{B2} & x_{B3} \\ x_{C1} & x_{C2} & x_{C3} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} = \begin{bmatrix} x_{A1}w_1 + x_{A2}w_2 + x_{A3}w_3 \\ x_{B1}w_1 + x_{B2}w_2 + x_{B3}w_3 \\ x_{C1}w_1 + x_{C2}w_2 + x_{C3}w_3 \end{bmatrix} = \begin{bmatrix} \hat{y}_A \\ \hat{y}_B \\ \hat{y}_C \end{bmatrix} = \begin{bmatrix} \mathbf{2.0} \\ \mathbf{1.0} \\ \mathbf{0.1} \end{bmatrix}$$

$$\begin{bmatrix} \mathbf{2.0} \\ \mathbf{1.0} \\ \mathbf{0.1} \end{bmatrix} \begin{matrix} \rightarrow \; \boldsymbol{p} = \mathbf{0.7} \; A(\mathrm{O}) \\ \rightarrow \; \boldsymbol{p} = \mathbf{0.2} \; B(\times) \\ \rightarrow \; \boldsymbol{p} = \mathbf{0.1} \; C(\times) \end{matrix}$$

$$\therefore \hat{\boldsymbol{y}} = \boldsymbol{A}$$

# 소프트맥스 회귀



- 로지스틱 함수를 여러 개의 입력인 경우 로지스틱 함수를 사용할 수 있도록 일반화 한 것
- 이진 분류기를 훈련시켜 연결하지 않고 직접 다중 클래스를 지원할 수 있도록 일반화한 것
- 다항 로지스틱 회귀($multinomial\ logistic\ regression$)라고도 한다
- 샘플에 대해 각 클래스의 점수가 계산되면 소프트맥스 함수를 통과시켜 해당되는 클래스에 속할 **확률을 추정한다**

$$logistic\ function \xrightarrow{\text{일반화}(generalization)} softmax\ function$$

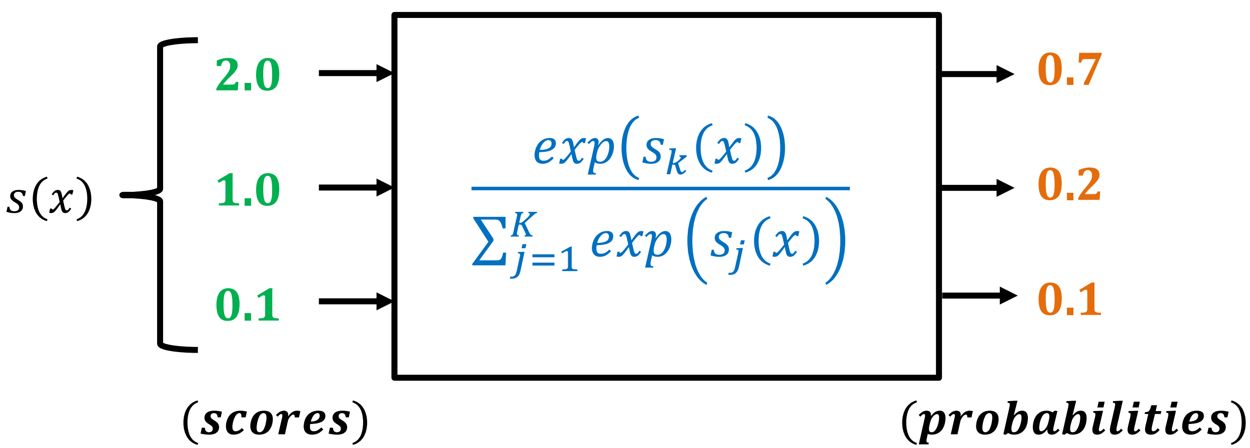
# 소프트맥스 함수(softmax function)



- 출력합계는 1. $n$차 실수 벡터를 0과 1 사이의 실수로 변환하여 출력

$$\sigma\big(s(x)\big)_k = \frac{exp\big(s_k(x)\big)}{\sum_{j=1}^{K} exp\left(s_j(x)\right)}$$

- $K$ : 클래스의 수
- $s(x)$ : 샘플 $x$에 대한 각 클래스의 점수를 담은 벡터
- $\sigma\big(s(x)\big)_k$ : 샘플 $x$ 에 대한 각 클래스의 점수가 주어졌을 때 이 샘플이 클래스 $k$ 에 속할 추정 **확률**



# CIFAR10



airplane
automobile
bird
cat
deer
dog
frog
horse
ship
truck

**50,000** training images
each image is **32x32x3**

**10,000** test images.

# Parameterized mapping from images to label scores



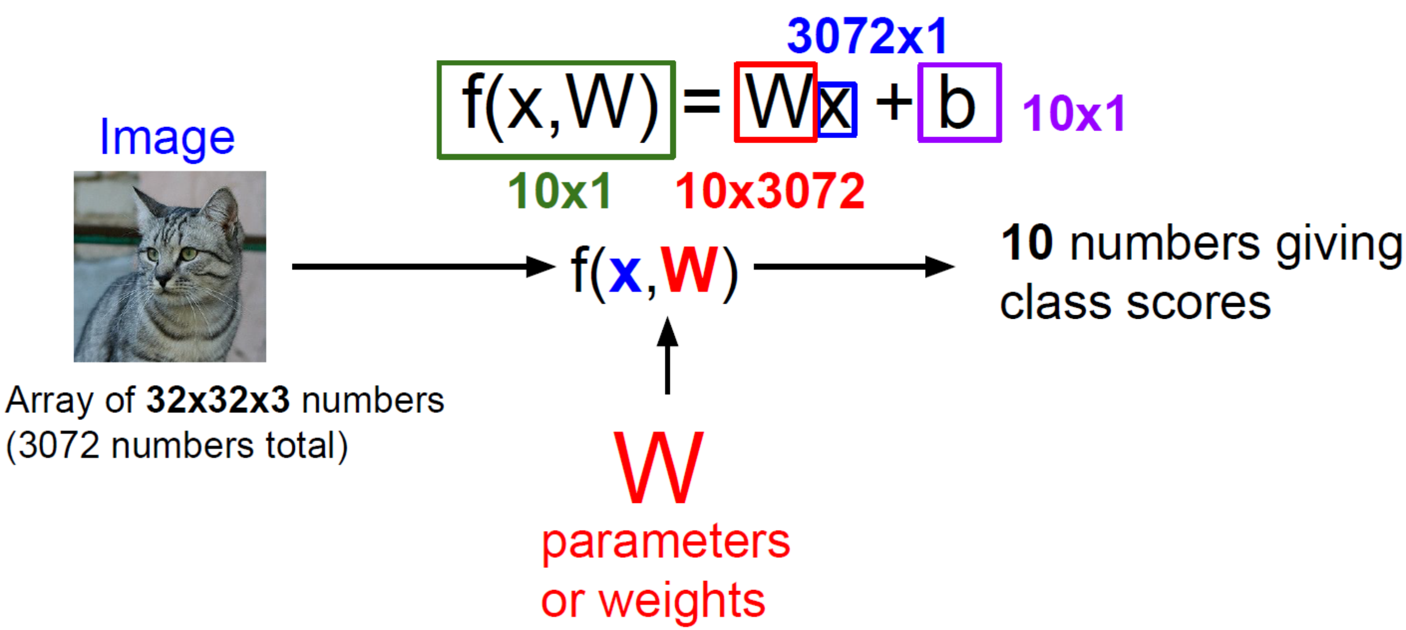
- training dataset of images : $x_i \in R^D, each\ associated\ with\ a\ label\ y_i$
$$(i = 1 \dots N\ \ and\ y_i \in 1 \dots K)$$
- we have ***N*** examples (each with a dimensionality ***D***) and ***K*** distinct categories.
- For example, in CIFAR-10
  - training set of ***N*** = 50,000 images, each with ***D*** = 32 x 32 x 3 = 3072 pixels, and ***K*** = 10, since there are 10 distinct classes (dog, cat, car, etc).
  - We will now define the **score function** $f: R^D \mapsto R^K$ that maps the raw image pixels to class scores.

http://cs231n.github.io/linear-classify/

# Linear classifier



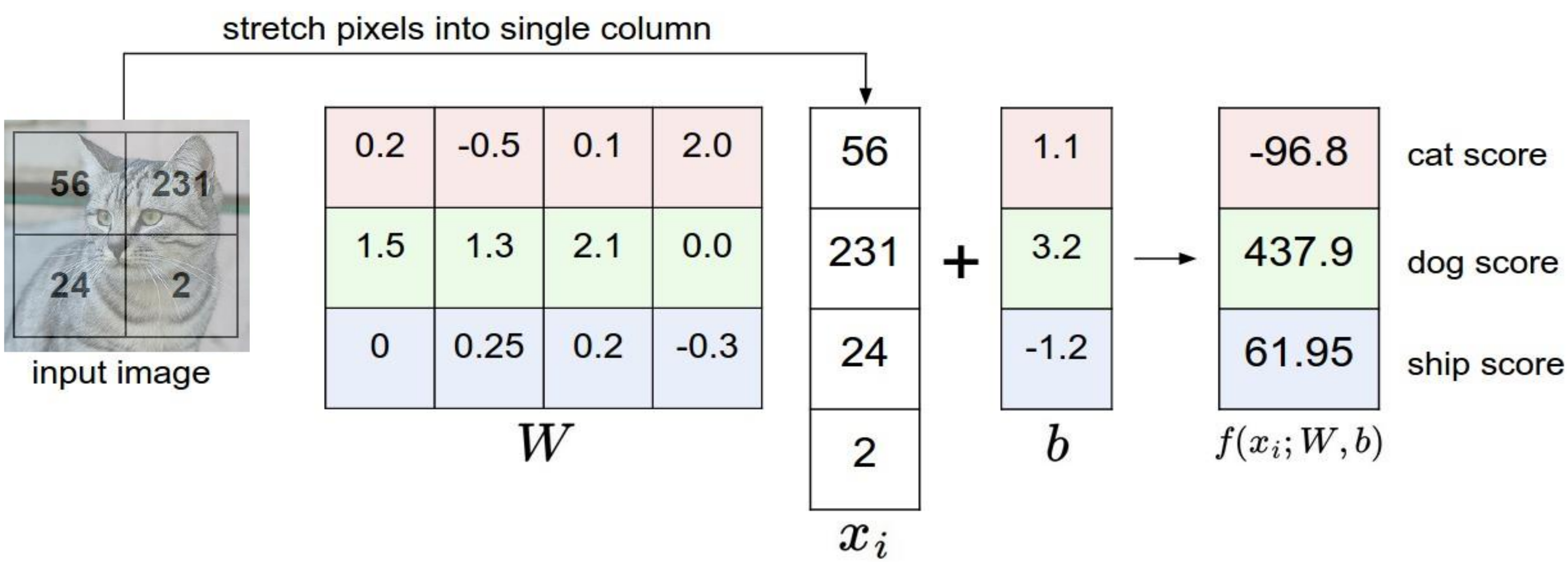
- $f(x_i, W, b) = Wx_i + b$
  - $x_i$ : image. has all of its pixels flattened out to a single column vector of shape [D x 1].
  - $W$ : matrix, of size [K × D], weights
  - $b$ : vector, of size [K × 1], bias vector
- In CIFAR-10
  - $x_i$ : $all\ pixels\ in\ the\ i-th\ image\ flattened\ into\ a\ single\ [3072 \times 1]\ column$
  - $W$ : $[10 \times 3072]$
  - $b$ : $[10 \times 1\ ]$



# Linear Classification



- 간결화를 위하여 4 pixel
- 3 class : red(cat), green(dog), blue(ship). (rgb채널과 관련없음)
- $f(x, W) = Wx + b$



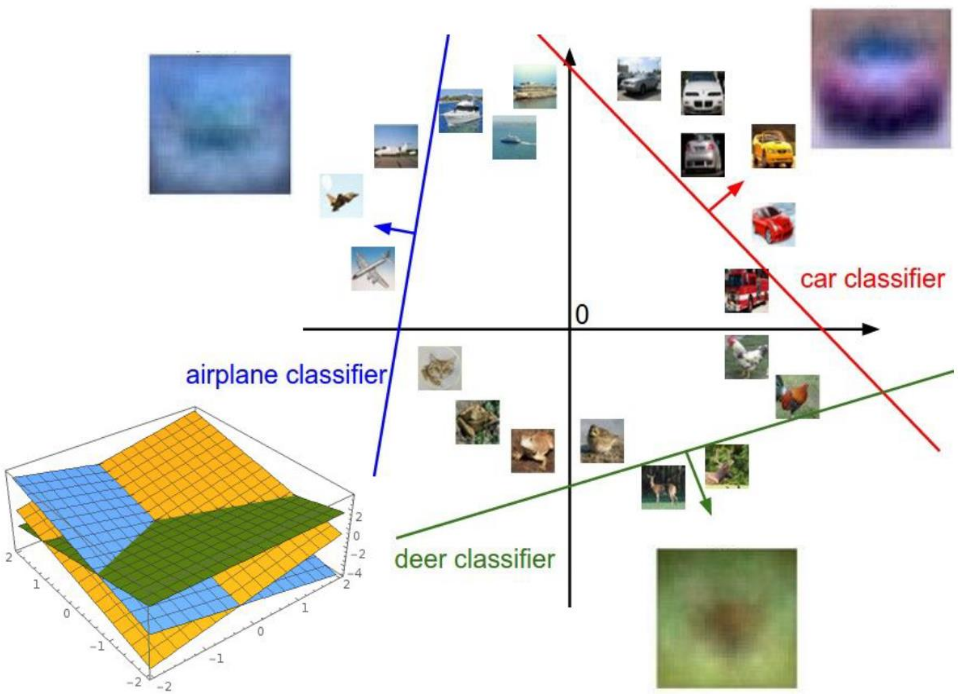

weight W는 좋지 않다
dog score가 가장 높다

http://cs231n.github.io/linear-classify/

# Analogy of images as high-dimensional points



- CIFAR-10의 각 이미지는 32 × 32 픽셀의 3072 차원 공간에 있는 점.
  - 마찬가지로 전체 데이터 세트는 (레이블이 지정된) 점 집합.
- 각 클래스의 점수를 모든 이미지 픽셀의 가중치로 정의 했으므로 각 클래스 점수는 이 공간에 대한 선형 함수(3072차원을 2차원으로 압축).




$$f(x,W) = Wx + b$$

Array of **32x32x3** numbers
(3072 numbers total)

Template Class


http://cs231n.github.io/linear-classify/

# How can we tell whether this W is good or bad?

- Defined a (linear) score function
- How can we tell whether this W is good or bad?



| | | | |
|---|---|---|---|
| airplane | -3.45 | -0.51 | 3.42 |
| automobile | -8.87 | **6.04** | 4.64 |
| bird | 0.09 | 5.31 | 2.65 |
| cat | **2.9** | -4.22 | 5.1 |
| deer | 4.48 | -4.19 | 2.64 |
| dog | 8.02 | 3.58 | 5.55 |
| frog | 3.78 | 4.49 | **-4.34** |
| horse | 1.06 | -4.37 | -1.5 |
| ship | -0.36 | -2.09 | -4.79 |
| truck | -0.72 | -2.93 | 6.14 |

http://cs231n.github.io/linear-classify/

# 정보이론(Information Theory)



- 섀넌($Claude\ Shannon$)의 정보이론
  - 잘 일어나지 않는 사건($unlikely\ event$)은 자주 발생하는 사건보다 정보량($informative$)이 많다
  - 확률 값이 0에 가까울수록 정보량은 무한대, 1에 가까울수록 정보량은 0이다
  - $P(x)$는 $x$가 발생될 확률. $I(x)$는 $x$의 정보량

$$I(x) = log_2 \frac{1}{P(x)} = -log_2$$

- 동전던지기
  - $I(x) = -log_2 P(x) = -\log(0.5) = 1bit$
- 8면체 주사위던지기
  - $I(x) = -log_2 P(x) = -\log\left(\frac{1}{8}\right) = 3bit$


➢ 클로드 섀넌(Claude Shannon, 1916년~2001년)
정보 이론의 아버지라고 불리며,디지털 회로 이론의 창시자

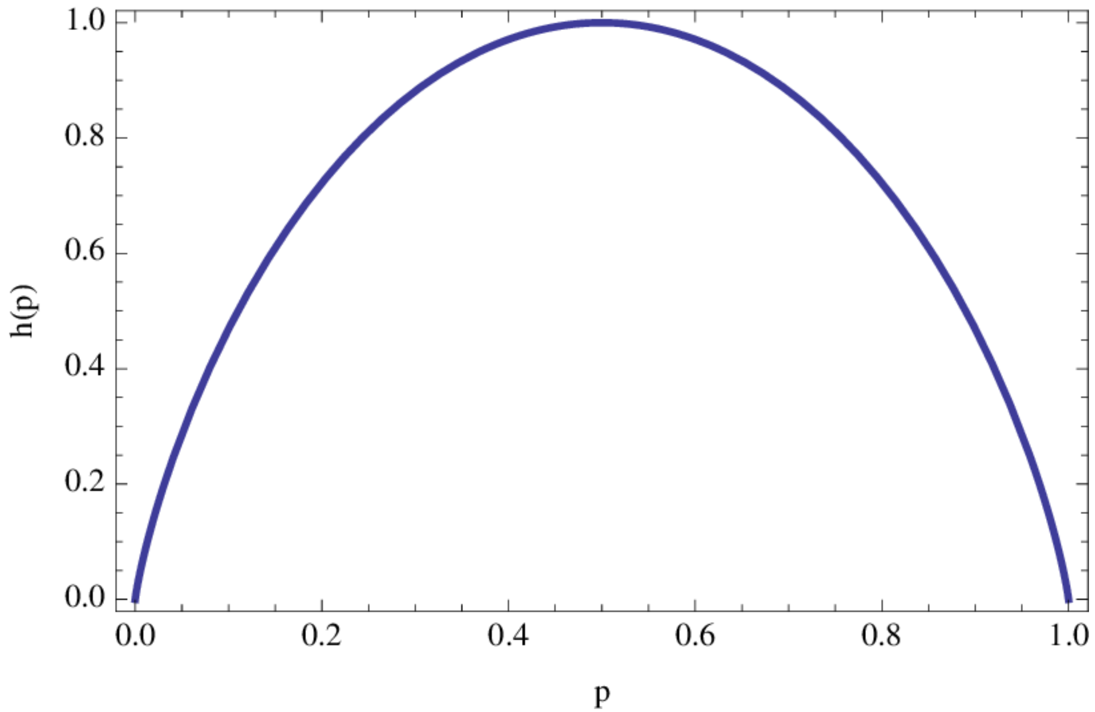
# Entropy



- 엔트로피($Entropy$) : 정보량($-log_2P(x)$)의 기대 값(정보량의 평균)

$$H(x) = -\sum_{i=1}^{n} p(x_i)\, log_2 p(x_i) = E_p[-log_2 p(x)]$$

- 동전을 던졌을 때 앞/뒷면이 나올 정보량의 기대 값(평균)
  - (앞/뒷면 = 50%/50%) $H(x) = -(0.5log_2 0.5 + 0.5log_2 0.5) = \mathbf{1}$
  - (앞/뒤면 = 100%/0%) $H(x) = -(1.0log_2 1.0 + 0.0log_2 0.0) = \mathbf{0}$
  - (앞/뒤면 = 90%/10%) $H(x) = -(0.9log_2 0.9 + 0.1log_2 0.1) = \mathbf{0.47}$



# Cost function : Cross Entropy

- 높은 확률을 추정하도록 만드는 것이 목적
  - 클래스가 2인경우는 로지스틱 회귀의 비용함수와 동일
- 실제확률 $p(x_i)$ 에 대하여 예측(학습)확률 $q(x_i)$를 통하여 예측하는 것

$$H(p,q) = \sum_{i=1}^{n} p(x_i)\, log_2 q\left(\frac{1}{x_i}\right) = -\sum_{i=1}^{n} p(x_i)\, log_2 q(x_i)$$

- $Cross\ Entropy\ \geq Entropy$
- $q(x_i)$가 학습하여 $p(x_i)$에 가까울 수록 $Cross\ Entropy$는 작아진다.
  즉, 최적은 $Cross\ Entropy$ 가 최소인 값을 찾는다 – 경사하강법

[예제] 가방에 0.8, 0.1, 0.1의 비율로 빨간,녹색,파란 공이 있다.
0.2, 0.2, 0.6의 비율로 있다고 예측하자.
이 때 $entropy$ 와 $cross\ entropy$ 는?

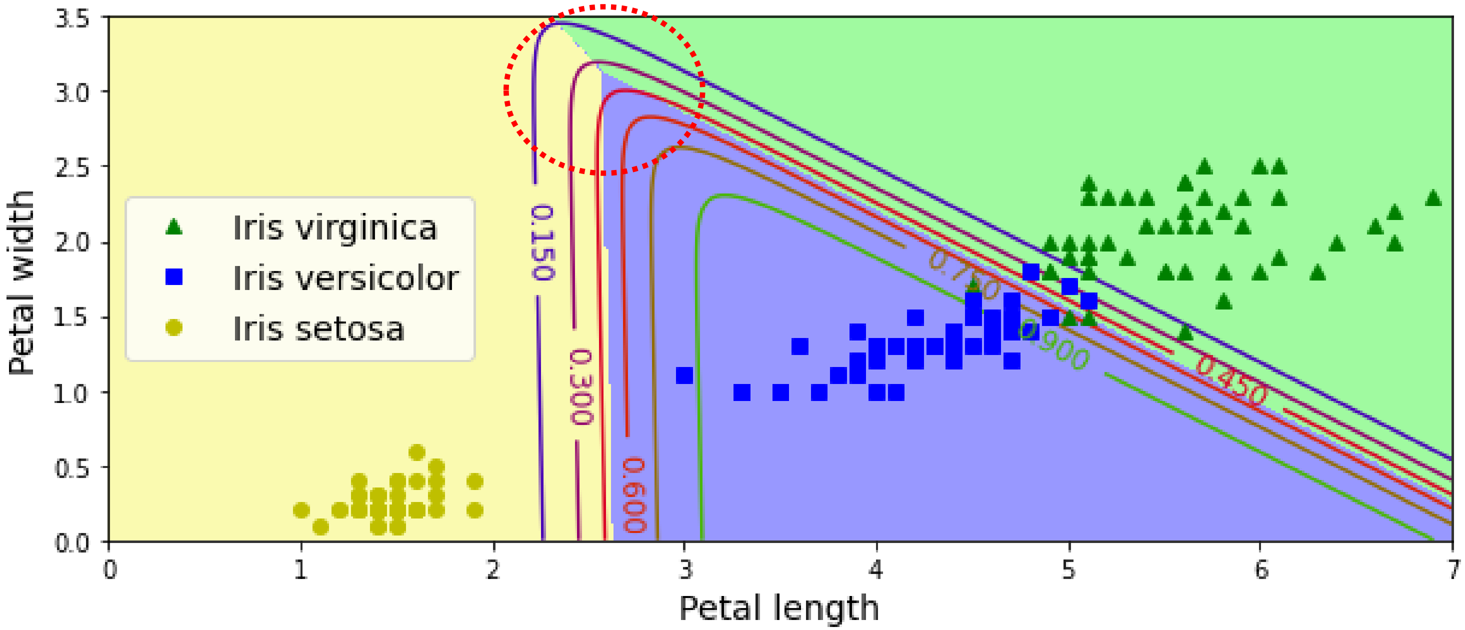
$$H(p) = -[(0.8 \log(0.8) + 0.1 \log(0.1) + 0.1 \log(0.1)] = 0.63$$
$$H(p,q) = -[(0.8 \log(0.2) + 0.1 \log(0.2) + 0.1 \log(0.6)] = 1.50$$

```
'''
소프트맥스 함수
로지스틱 회귀모델은 여러 개의 이진 분류기를 훈련시켜 연결하지 않고,
직접 다중 범주를 지원하도록 일반화될 수 있다.
이를 소프트맥스 회귀 또는 다항 로지스틱 회귀라고 한다.
'''

X = iris["data"][:, (2, 3)]  # 꽃잎 길이와 너비 변수
y = iris["target"]           # 3개의 범주 그대로 사용

'''
- 사이킷런의 LogisticRegression은 범주가 2이상이면 일대다(OvA) 전략을 사용
- 하지만 multi_class='multinomial'옵션: 소프트맥스 회귀를 사용
- 소프트맥스 회귀를 사용하려면 solver='lbfgs'옵션을 준다
- lbfgs:소프트맥스 회귀를 지원하는 알고리즘
'''
softmax_reg = LogisticRegression(multi_class="multinomial",
                solver="lbfgs", C=10, random_state=42)
softmax_reg.fit(X, y)
```

```python
# 훈련시킨 소프트맥스 분류기의 결정경계를 시각화. 새로운 샘플 생성
x0, x1 = np.meshgrid(
        np.linspace(0, 8, 500).reshape(-1, 1),
        np.linspace(0, 3.5, 200).reshape(-1, 1),
    )
X_new = np.c_[x0.ravel(), x1.ravel()]

y_proba = softmax_reg.predict_proba(X_new)
y_predict = softmax_reg.predict(X_new)

zz1 = y_proba[:, 1].reshape(x0.shape)
zz = y_predict.reshape(x0.shape)

plt.figure(figsize=(10, 4))
plt.plot(X[y==2, 0], X[y==2, 1], "g^", label="Iris virginica")
plt.plot(X[y==1, 0], X[y==1, 1], "bs", label="Iris versicolor")
plt.plot(X[y==0, 0], X[y==0, 1], "yo", label="Iris setosa")

from matplotlib.colors import ListedColormap
custom_cmap = ListedColormap(['#fafab0','#9898ff','#a0faa0'])

plt.contourf(x0, x1, zz, cmap=custom_cmap)
contour = plt.contour(x0, x1, zz1, cmap=plt.cm.brg)
plt.clabel(contour, inline=1, fontsize=12)
plt.xlabel("Petal length", fontsize=14)
plt.ylabel("Petal width", fontsize=14)
plt.legend(loc="center left", fontsize=14)
plt.axis([0, 7, 0, 3.5])

plt.show()
```

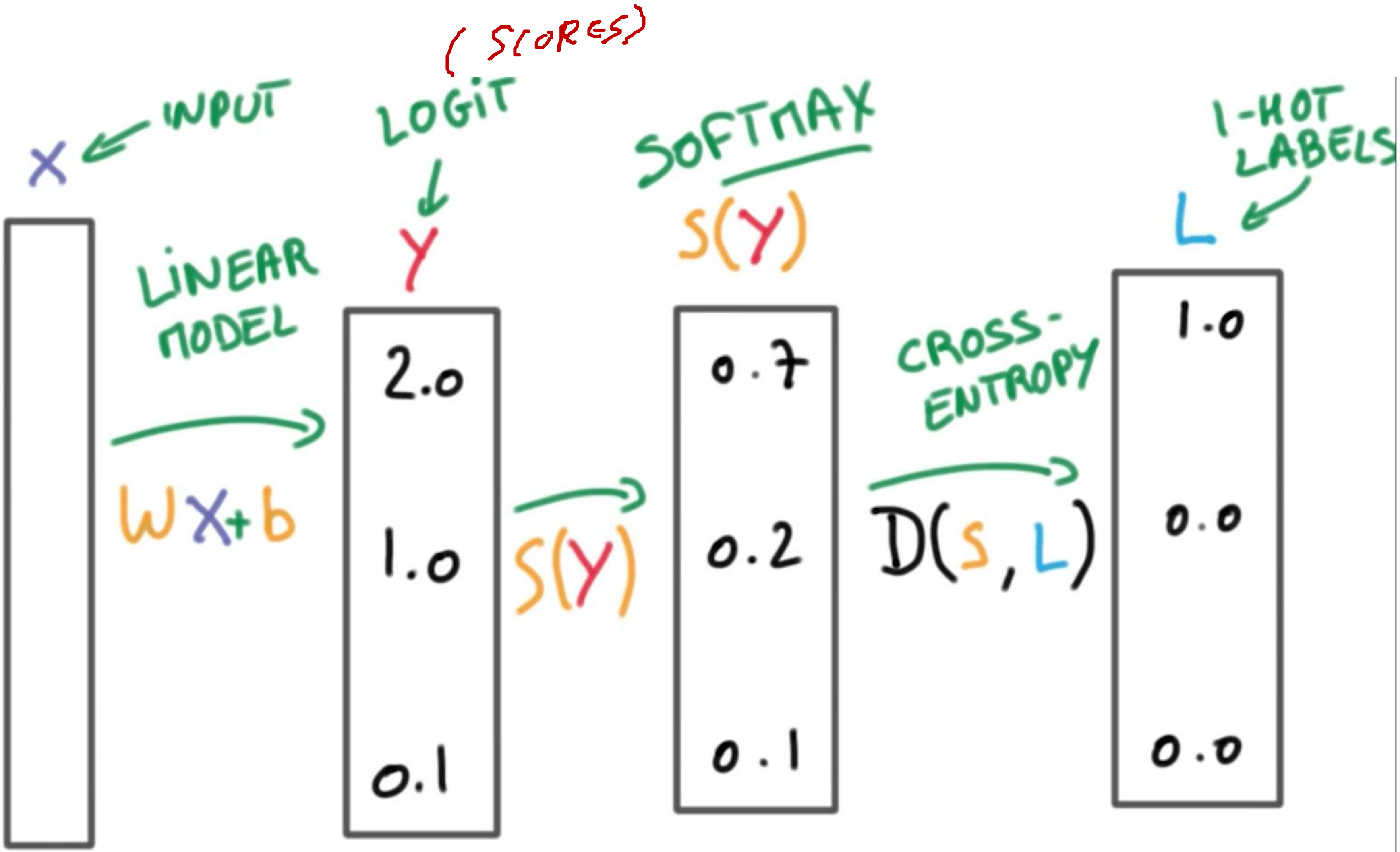
# 소프트맥스 회귀 결정경계






- **Iris versicolor클래스의 확률곡선**
- 이 모델은 이진분류와 달리 0.5의 경계가 아니라 범주에 속할 확률이 0.5 이하라도 분류한다는 점을 유의하자(범주가 2개보다 많으므로).
- 즉, 모든 결정경계가 만나는 지점은 동일하게 33.3%의 추정확률을 갖는다.

Ref : Hands-on Machine Learning with Scikit-Learn,keras&tensorflow - O'REILLY




https://www.udacity.com