SOFTMAX

GENERALIZATION OF LOGISTIC REGRESSION

Multi-Class Classification

LOGISTIC REGRESSION → LEARNING FRAMEWORK TO DISTINGUISH AMONG TWO CLASSES.

$x^{(i)} \in \mathbb{R}^m$ ($x_0^{(i)} = 1$) $\quad y^{(i)} \in \{0, 1\}$

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

$\theta^T = [\theta_0, \theta_1, \theta_2, \ldots, \theta_m] \in \mathbb{R}^{m+1}$

$$x = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_m \end{bmatrix}$$

TRAINING SET $= \{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\}$ ← m SAMPLES

GOAL: MINIMIZE THE COST FUNCTION $J(\theta)$

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} \log\left(h_\theta(x^{(i)})\right) + (1 - y^{(i)}) \log\left(1 - h_\theta(x^{(i)})\right)\right]$$

CROSS-ENTROPY LOSS

REPEAT {

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial J(\theta)}{\partial \theta_j}$$

} FOR ALL j SIMULTANEOUSLY

GRADIENT DESCENT IS USED TO FIND PARAMETERS θ

LEARNING RATE: 0.1, 0.03, 0.01, 0.003, 0.001, ...

TO PREVENT OVERFITTING REGULARIZATION IS USED

$$J(\theta) = -\frac{1}{m}\left[\cdots\right] + \frac{\lambda}{2m}\sum_{j=1}^{m} \theta_j^2$$

1 VS ALL APPROACH TO PERFORM MULTI-CLASS CLASSIFICATION

SOFTMAX → LEARNING FRAMEWORK TO PERFORM MULTI-CLASS CLASSIFICATION.

IT IS AN EXTENSION OF THE LOGISTIC REGRESSION.

$x^{(i)} \in \mathbb{R}^m$ $\quad x_0^{(i)} = 1$

$y^{(i)} \in \{0, 1, \dots K-1\}$

↳ K CLASSES

ONE HOT ENCODING $\begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix} \leftarrow \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}$

TRAINING SET $= \{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \dots, (x^{(m)}, y^{(m)})\}$

↳ m SAMPLES

WE WANT TO ESTIMATE $P_v(y = k \mid x)$ $\forall k = 0, 1, \dots K-1$

⇒ OUR HYPOTHESIS HAVE TO ESTIMATE A K VECTORS

SOFTMAX MODEL

$$h_v(x) = \begin{bmatrix} P_{v^{(0)}}(y=0 \mid x) \\ P_{v^{(1)}}(y=1 \mid x) \\ P_{v^{(2)}}(y=2 \mid x) \\ \vdots \\ P_{v^{(K-1)}}(y=K-1 \mid x) \end{bmatrix} = \frac{1}{\sum_{k=0}^{K-1} e^{v^{(k)T}x}} \begin{bmatrix} e^{v^{(0)T}x} \\ e^{v^{(1)T}x} \\ e^{v^{(2)T}x} \\ \vdots \\ e^{v^{(K-1)T}x} \end{bmatrix}$$

$h_v(x)$ IS A MATRIX

$v \in \mathbb{R}^{(m+1) \times K}$

$$v = \begin{bmatrix} v_0^{(0)} & v_0^{(1)} & \cdots & v_0^{(K-1)} \\ v_1^{(0)} & v_1^{(1)} & \cdots & v_1^{(K-1)} \\ \vdots & \vdots & & \vdots \\ v_m^{(0)} & v_m^{(1)} & \cdots & v_m^{(K-1)} \end{bmatrix}$$

NORMALIZE THE DISTRIBUTION

NOTE THAT WHEN PARAMETERS ARE FIXED (LEARNED)

$$\sum_{k=0}^{K-1} P(y=k \mid x) = 1$$

Softmax Cost Function

- It uses cross-entropy loss function
- Let be $1\{\cdot\}$ the indicator function

$$1\{\text{proposition}\} = \begin{cases} 1 & \text{if proposition is true} \\ 0 & \text{if proposition is false} \end{cases}$$

- Let be $h_\theta^{(c)}(x) = P(y=c|x) = \dfrac{e^{\theta^{(c)T}x}}{\sum_{k=0}^{K-1} e^{\theta^{(k)T}x}}$

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{c=0}^{K-1} 1\{y^{(i)}=c\}\log\left(h_\theta^{(c)}(x^{(i)})\right)\right] + \frac{\lambda}{2m}\sum_{c=0}^{K-1}\sum_{j=1}^{n}\theta_j^{(c)2}$$

Regularization term

LEARNING GOAL IS TO FIND THE PARAMETERS

$$\theta = \begin{bmatrix} \theta_0^{(0)} & \theta_0^{(1)} & \theta_0^{(2)} & \cdots & \theta_0^{(K-1)} \\ \theta_1^{(0)} & \theta_1^{(1)} & \theta_1^{(2)} & \cdots & \theta_1^{(K-1)} \\ \vdots & \vdots & \vdots & & \vdots \\ \theta_m^{(0)} & \theta_m^{(1)} & \theta_m^{(2)} & \cdots & \theta_m^{(K-1)} \end{bmatrix}$$

THIS IS DONE BY FINDING THE MINIMUM OF THE DERIVATIVE OF THE COST FUNCTION WITH GRADIENT DESCENT (OR WITH STOCASTIC GRADIENT DESCENT)

REPEAT {

$$\theta_j^{(c)} \leftarrow \theta_j^{(c)} - \alpha \frac{\partial J(\theta)}{\partial \theta_j^{(c)}}$$

FOR ALL $j = 0, \ldots, m$ AND $c = 0, \ldots, K-1$

SIMULTANEOUSLY UNTIL CONVERGENCE }

$$\frac{\partial J(\theta)}{\partial \theta_j^{(c)}} = -\frac{1}{m} \sum_{i=1}^{m} \left[\left(\mathbb{1}\{y^{(i)} = c\} - h_\theta^{(c)}(x^{(i)}) \right) x_j^{(i)} \right] + \underbrace{\frac{\lambda}{m} \theta_j^{(c)}}_{\text{WITH REGULARIZATION}}$$

TO PREVENT OVERFITTING

PROPERTY OF SOFTMAX

SOFTMAX MODEL IS "OVERPARAMETRIZED", WHICH MEANS THAT FOR ANY HYPOTESIS WHICH MIGHT FIT THE DATA, THERE ARE MULTIPLE PARAMETERS SETTINGS THAT GIVE EXACTLY THE SAME SOLUTION TO MAP DATA $x \in \mathbb{R}^m$ TO PREDICTIONS $y \in \{0, \dots, K-1\}$

$$h_\theta^{(c)}(x^{(i)}) = P_{\theta^{(c)}}(y^{(i)} = c \mid x^{(i)}) = \frac{e^{(\theta^{(c)} - \psi)^T x^{(i)}}}{\sum_{k=0}^{K-1} e^{(\theta^{(k)} - \psi)^T x^{(i)}}}$$

$$= \frac{e^{\theta^{(c)T} x^{(i)}} \cdot e^{-\psi^T x^{(i)}}}{e^{-\psi^T x^{(i)}} \cdot \sum_{k=0}^{K-1} e^{\theta^{(k)T} x^{(i)}}} = \boxed{\frac{e^{\theta^{(c)T} x^{(i)}}}{\sum_{k=0}^{K-1} e^{\theta^{(k)T} x^{(i)}}}}$$

SUBTRACTING $\psi \in \mathbb{R}^{m+1}$ TO EACH PARAMETER $\theta^{(c)}$ $c = 0 \dots K-1$ DO NOT AFFECT THE HYPOTESIS h_θ

$$\psi = \begin{bmatrix} \psi_0 \\ \vdots \\ \psi_m \end{bmatrix}$$

SOFTMAX COST FUNCTION GENERALIZES THE LOGISTIC REGRESSION COST FUNCTION:

FOR $K=2$ (NOT CONSIDERING REGULARIZATION)

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{c=0}^{1} 1\{y^{(i)}=c\}\log\left(h_\theta^{(c)}(x^{(i)})\right)\right] =$$

$$= -\frac{1}{m}\left[\sum_{i=1}^{m}(1-y^{(i)})\log\left(h_\theta^{(0)}(x^{(i)})\right)+y^{(i)}\log\left(h_\theta^{(1)}(x^{(i)})\right)\right] =$$

$$= -\frac{1}{m}\left[\sum_{i=1}^{m}(1-y^{(i)})\log\left(1-h_\theta^{(1)}(x^{(i)})\right)+y^{(i)}\log\left(h_\theta^{(1)}(x^{(i)})\right)\right]$$

$h_\theta^{(0)} + h_\theta^{(1)} = 1 \Rightarrow h_\theta^{(0)} = 1 - h_\theta^{(1)}$

LOGISTIC REGRESSION IS A SPECIAL CASE OF SOFTMAX

RELATIONSHIP BETWEEN LOGISTIC REGRESSION AND SOFTMAX

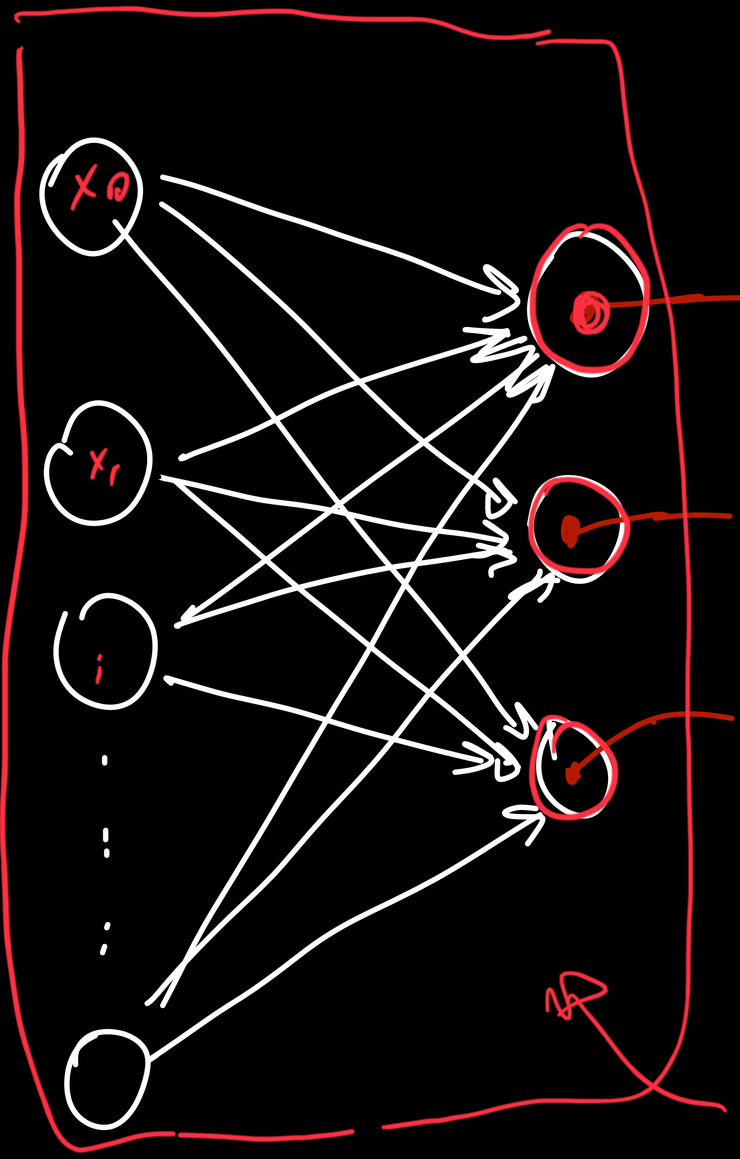
BY USING THE PROPERTY OF "OVERPARAMETRIZATION" IT IS SIMPLE TO SHOW THAT IN THE SPECIAL CASE OF $K=2$ THE SOFTMAX REGRESSION REDUCES TO LOGISTIC REGRESSION

$$h_\theta(x) = \begin{bmatrix} \frac{e^{\theta^{(0)T}x}}{e^{\theta^{(0)T}x} + e^{\theta^{(1)T}x}} \\ \frac{e^{\theta^{(1)T}x}}{e^{\theta^{(0)T}x} + e^{\theta^{(1)T}x}} \end{bmatrix} = \begin{bmatrix} \frac{e^{(\theta^{(0)}-\theta^{(1)})^T x}}{e^{(\theta^{(0)}-\theta^{(1)})^T x} + e^{\vec{(0)}^T x}} \\ \frac{e^{(\theta^{(1)}-\theta^{(1)})^T x}}{e^{(\theta^{(0)}-\theta^{(1)})^T x} + e^{\vec{(0)}^T x}} \end{bmatrix} = e^{\vec{(0)}^T} = 1$$

$$= \begin{bmatrix} \frac{e^{(\theta^{(0)}-\theta^{(1)})^T x}}{1 + e^{(\theta^{(0)}-\theta^{(1)})^T x}} \\ \frac{1}{1 + e^{(\theta^{(0)}-\theta^{(1)})^T x}} \end{bmatrix}^T = \begin{bmatrix} 1 - \phi \\ \phi \end{bmatrix}^T$$

$\theta^{(0)} - \theta^{(1)}$ CAN BE REPLACED BY A SINGLE PARAMETER θ

SOFTMAX "NETWORK"



LINEAR MODEL $v^T x$

+

LOGIT $e^{v^T x}$

+

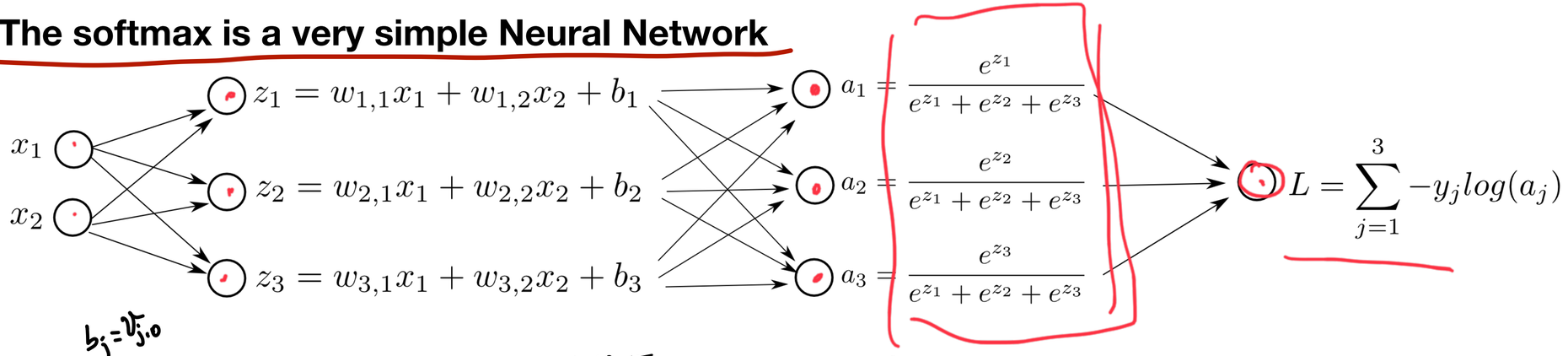
NORMALIZATION $\frac{e^{v^T x}}{\sum e^{v^T x}}$

$\hat{y}_1$ $\hat{y}_2$ $\hat{y}_3$

SOFTMAX "INGREDIENTS"

$$\sum_{i=1}^{3} \hat{y}_i = 1$$

The softmax is a very simple Neural Network



$b_j = w_{j,0}$

MODEL PARAMETERS

$$\theta^T = \begin{bmatrix} b_1 & w_{1,1} & w_{1,2} \\ b_2 & w_{2,1} & w_{2,2} \\ b_3 & w_{3,1} & w_{3,2} \end{bmatrix} \qquad X = \begin{bmatrix} x_0 = 1 \\ x_1 \\ x_2 \end{bmatrix}$$

INPUT

$$Y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$$

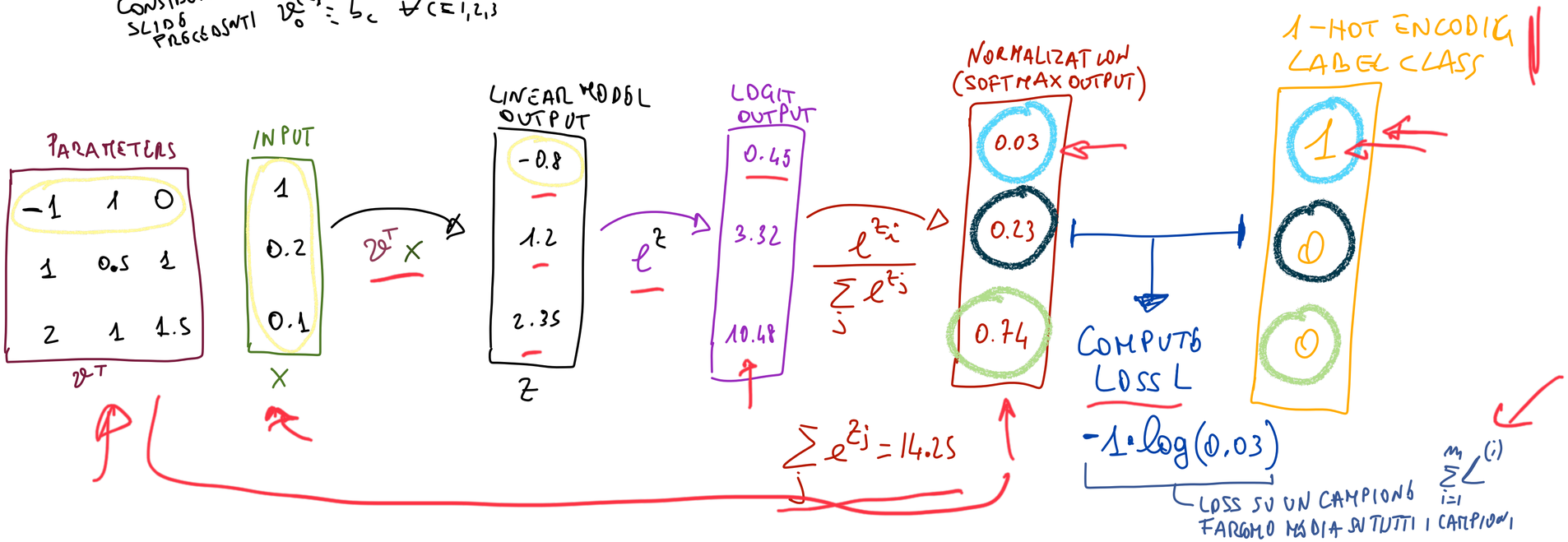
OUTPUT ≡ CLASS LABEL

$y_j \in \{0,1\} \quad \forall j = 1,2,3$

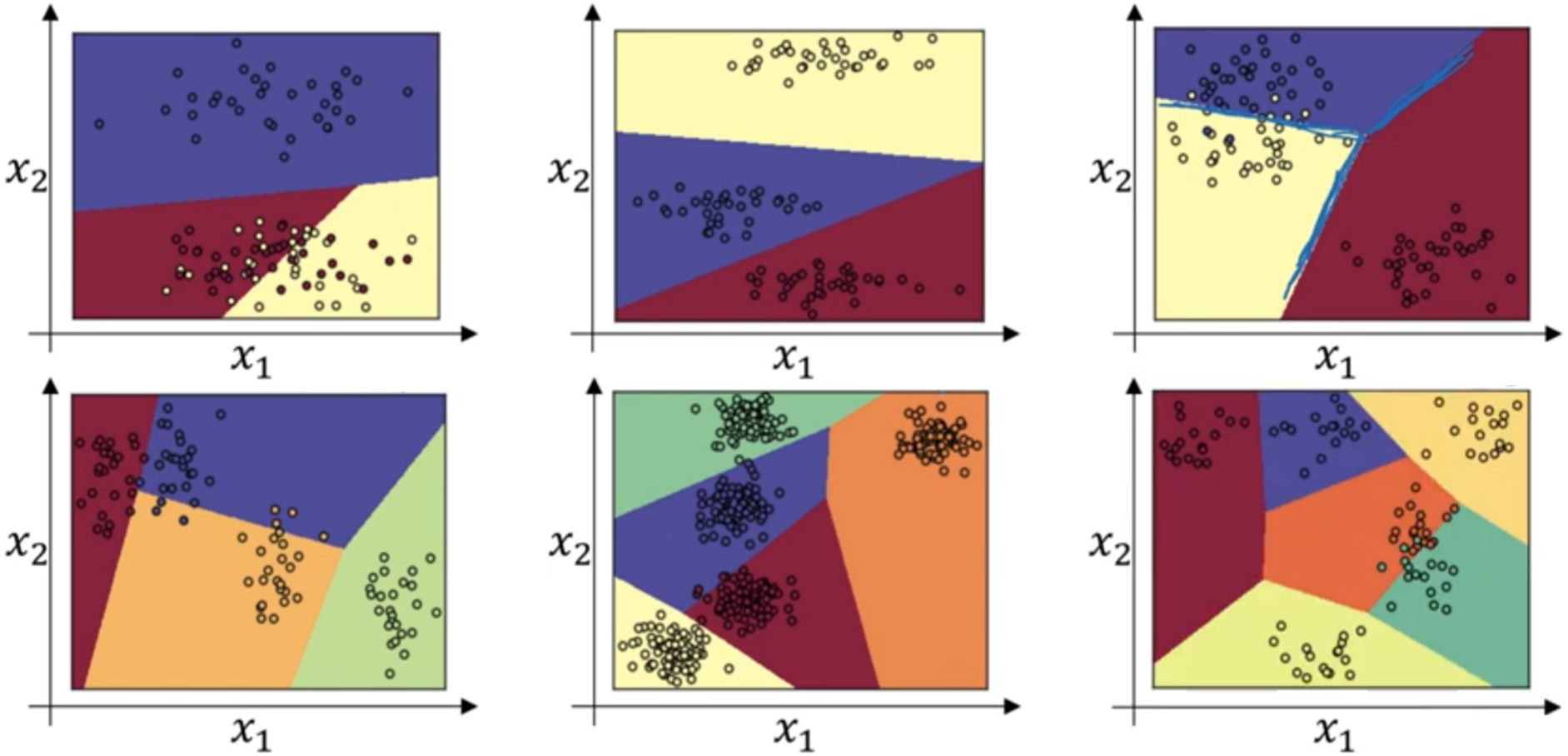
$\sum_j y_j = 1$

NOTAZIONE CONSIDERANDO SLIDE PRECEDENTI $\theta_j^{(c)} \equiv w_{cj} \quad \forall j = 1,2 \; \forall c = 1,2,3$

$\theta_0^{(c)} = b_c \quad \forall c = 1,2,3$



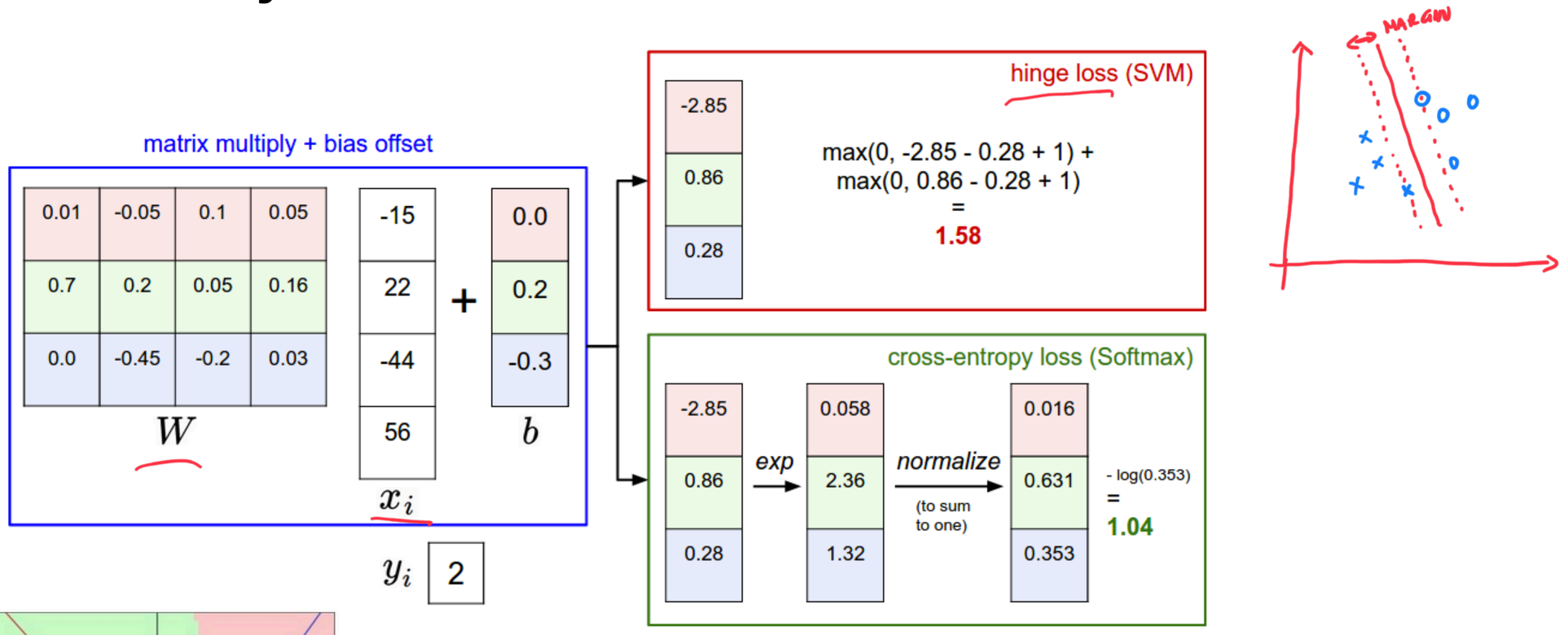
Softmax Classifier - Esempi

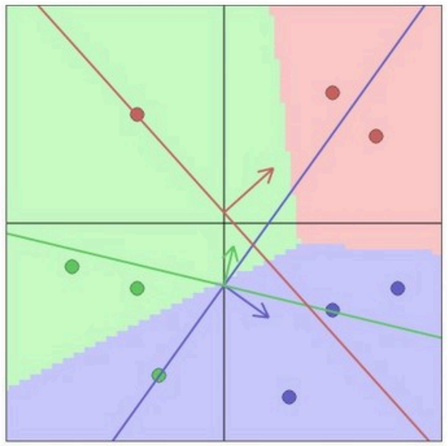


LINEAR BOUNDARIES — BUT WE CAN USE $\phi(x)$ KERNEL (OUTPUT OF ANOTHER NN) TO REPRESENT DATA AND HAVE NON-LINEAR BOUNDARIES

POSSIAMO USARE AD ESEMPIO UN MODELLO POLINOMIALE

Changing the Loss Function in this linear machine framework you can obtan the Multi-Class SVM

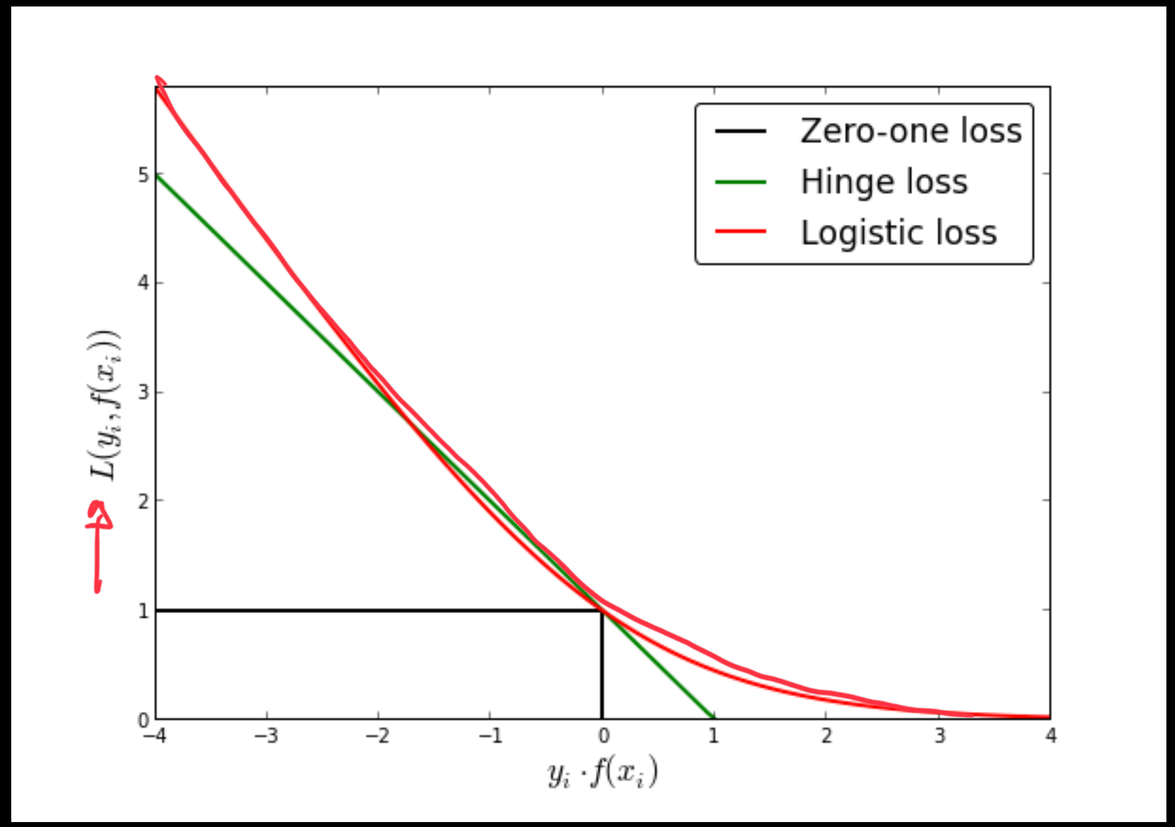




$$L = \underbrace{\frac{1}{N}\sum_i \sum_{j \neq y_i} \max(0, f_j - f_{y_i} + 1)}_{\text{data loss}} + \underbrace{\lambda \sum_k \sum_l W_{k,l}^2}_{\text{regularization loss}}$$

SAME FRAMEWORK OF SOFTMAX BUT A DIFFERENT LOSS FUNCTION IS USED

Loss Functions for Classification (binary)
Some examples



$$L(x,y) = \sum_{i=1}^{m} \text{loss}\left(f(x^{(i)}), y^{(i)}\right)$$

Zero-one loss

$$1\{f(x^{(i)}) = y^{(i)}\}$$

↑ indicator function

Hinge loss

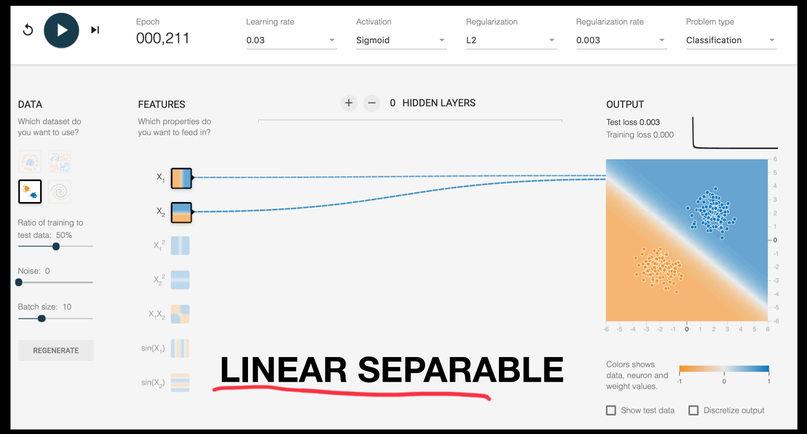
$$\max\left(0, 1 - f(x^{(i)}) y^{(i)}\right)$$

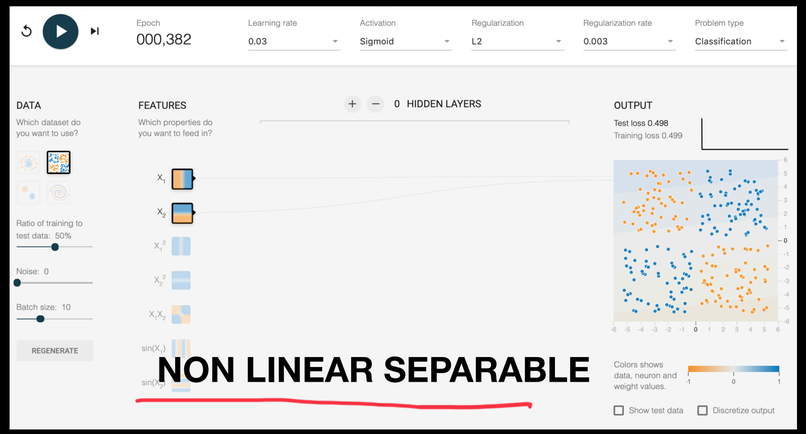
Logistic

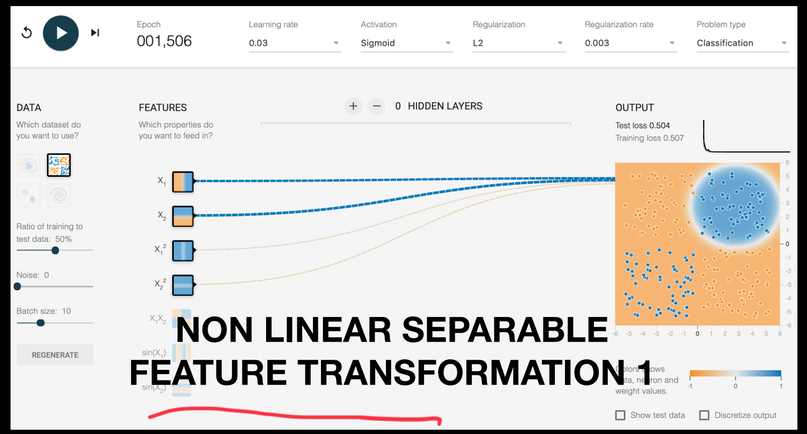
$$\log\left(1 + \exp\left(f(x^{(i)}) y^{(i)}\right)\right)$$

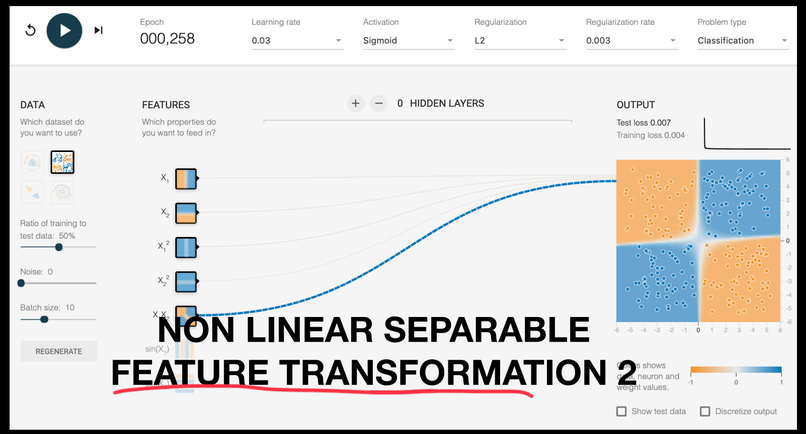
BINARY CLASSIFICATION

https://playground.tensorflow.org



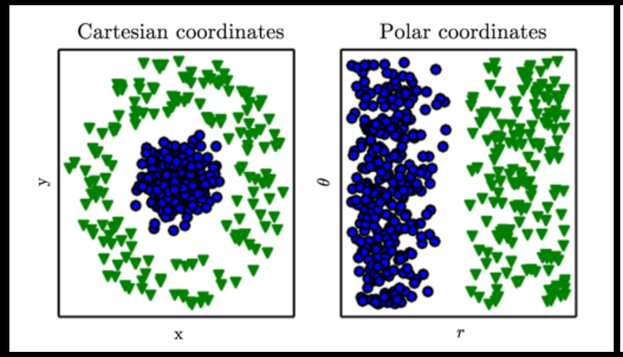


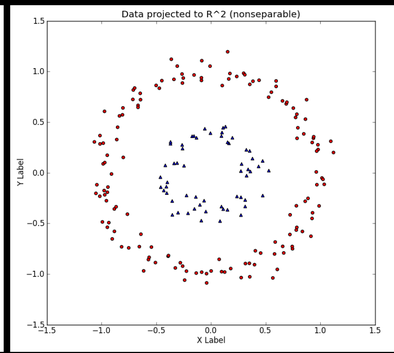


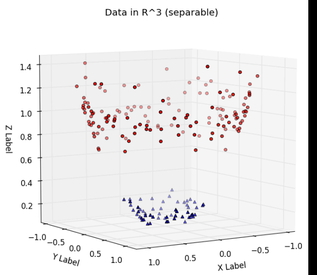


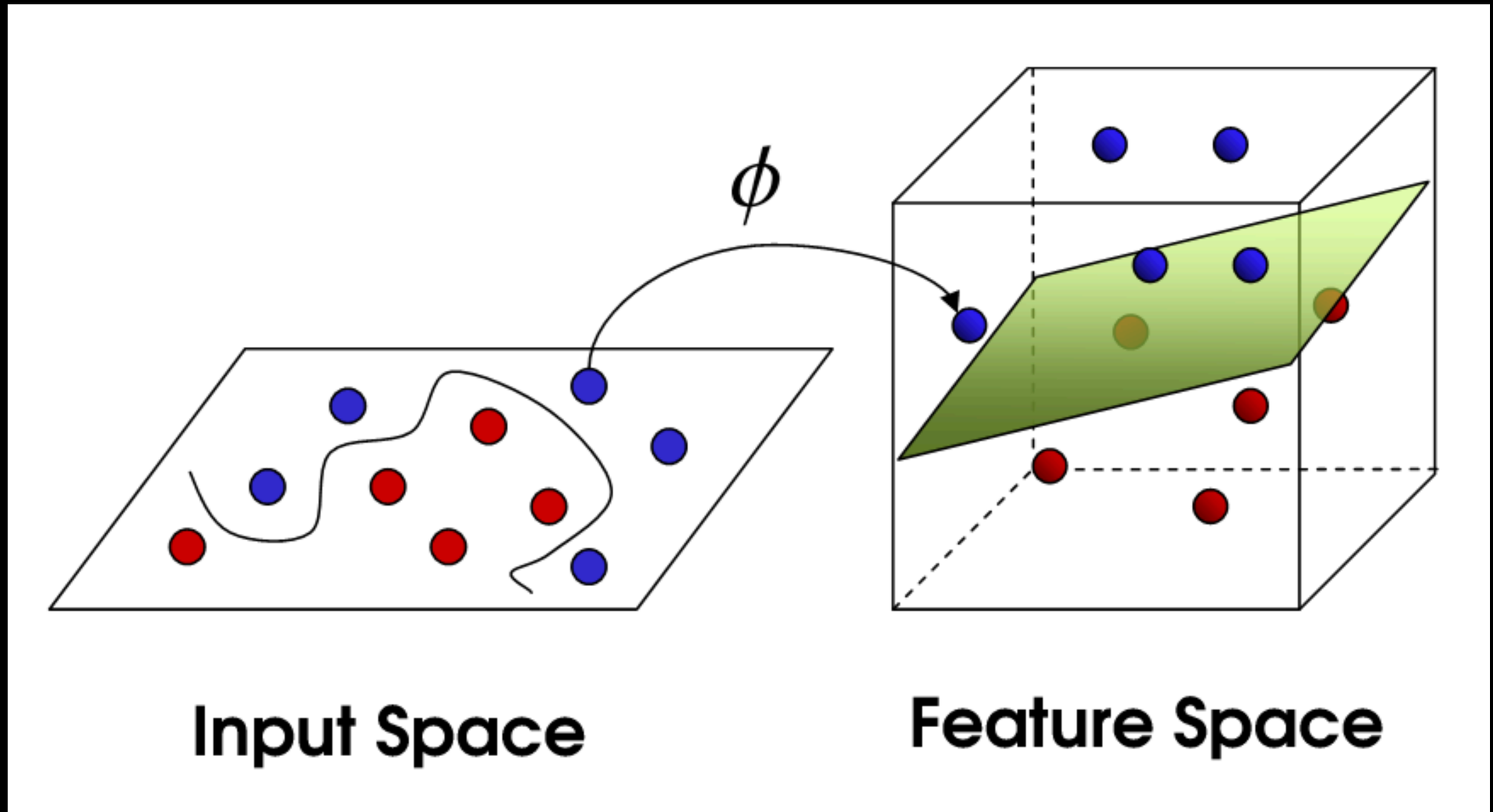
Role of Representation

How to define?









- REPRESENTATION LEARNING INTUITION
  - WE HAVE A NICE FRAMEWORK FOR MULTI-CLASS CLASSIFICATION = SOFTMAX
  - WE KNOW THAT IF WE HAVE A GOOD KERNEL FUNCTION $\phi(x)$ TO MAP FEATURE IN HIGH DIMENSIONAL NON LINEAR FEATURE SPACE WE CAN SOLVE COMPLEX PROBLEM

- HOW TO CHOOSE $\phi(x)$ IS NOT SIMPLE
  CAN WE LEARN A FUNCTION $\phi(x)$ FOR REPRESENTATION?
  $\Rightarrow$ NEURAL NET & DEEP LEARNING.

Network of Neurons - Learning Representation

https://playground.tensorflow.org