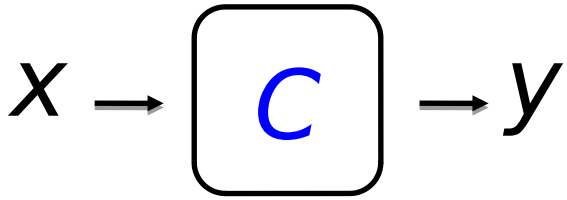
Natural Language Processing

Lecture 9: Classification

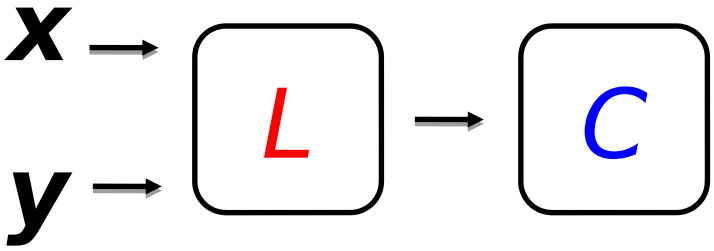
Classification

Notation

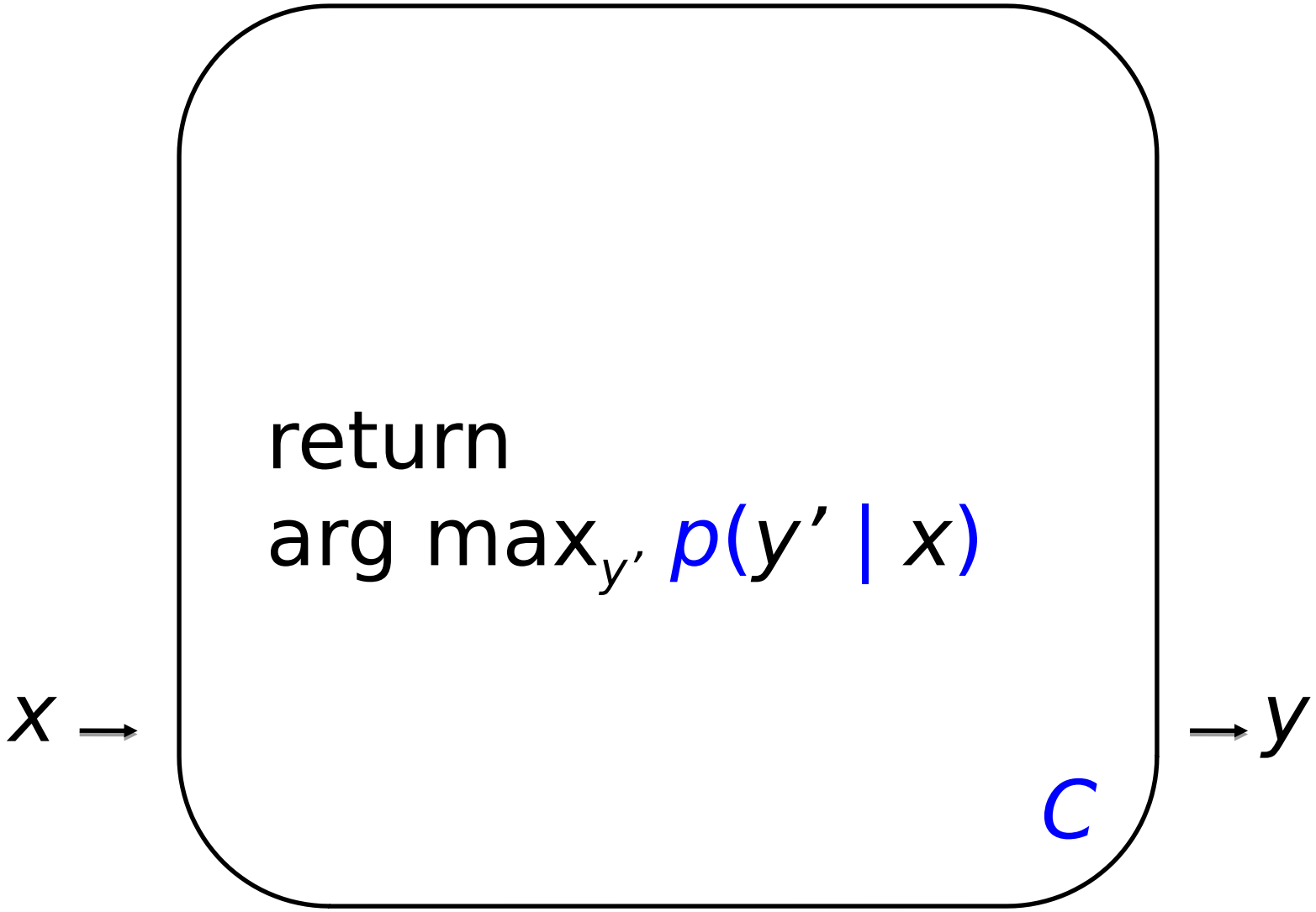
- Training examples: $\boldsymbol{x} = (x_1, x_2, \ldots, x_N)$
- Their categories: $\boldsymbol{y} = (y_1, y_2, \ldots, y_N)$
- A **classifier** C seeks to map x_i to y_i



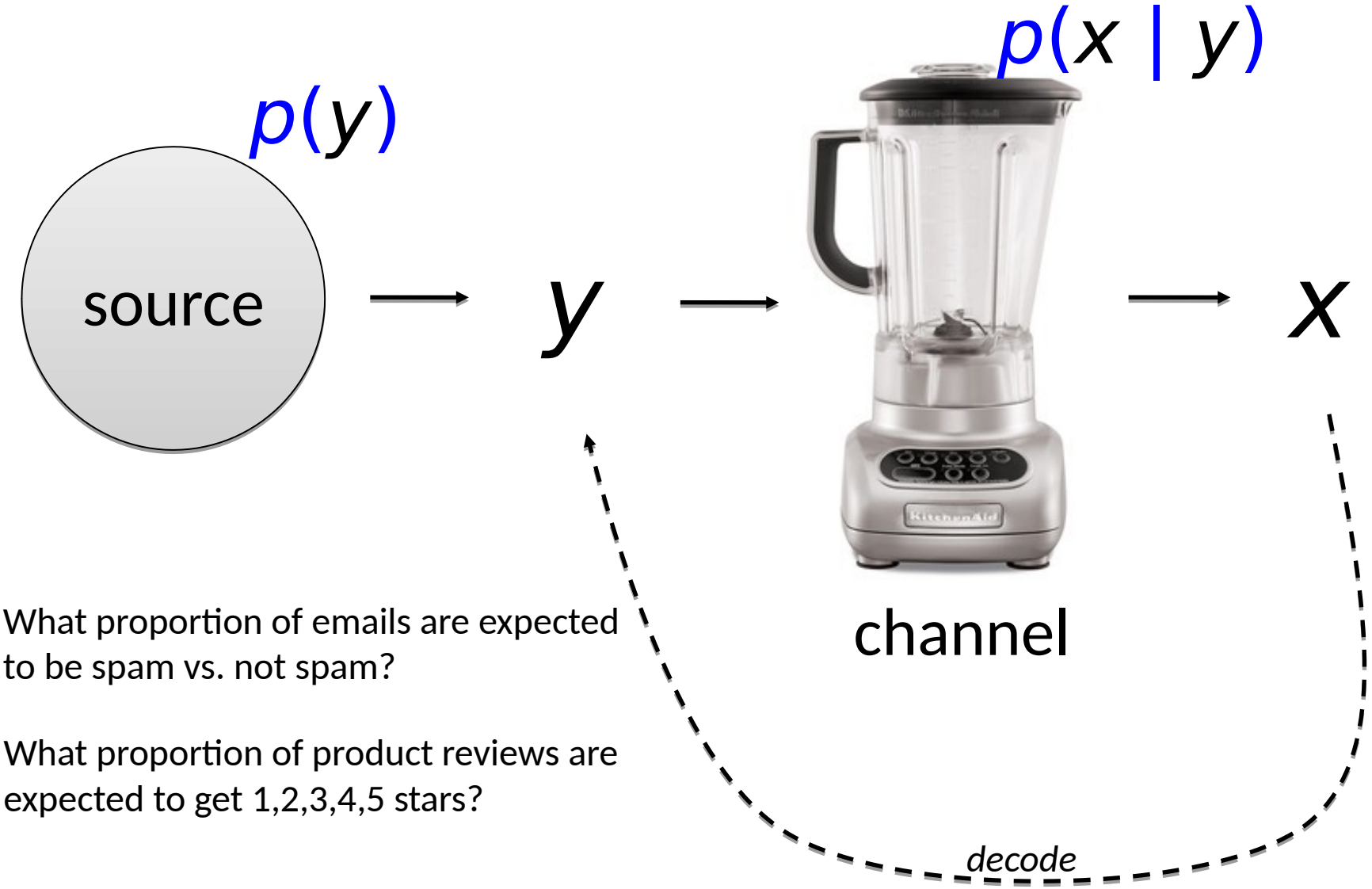
- A **learner** L infers C from $(\boldsymbol{x}, \boldsymbol{y})$



Probabilistic Classifiers



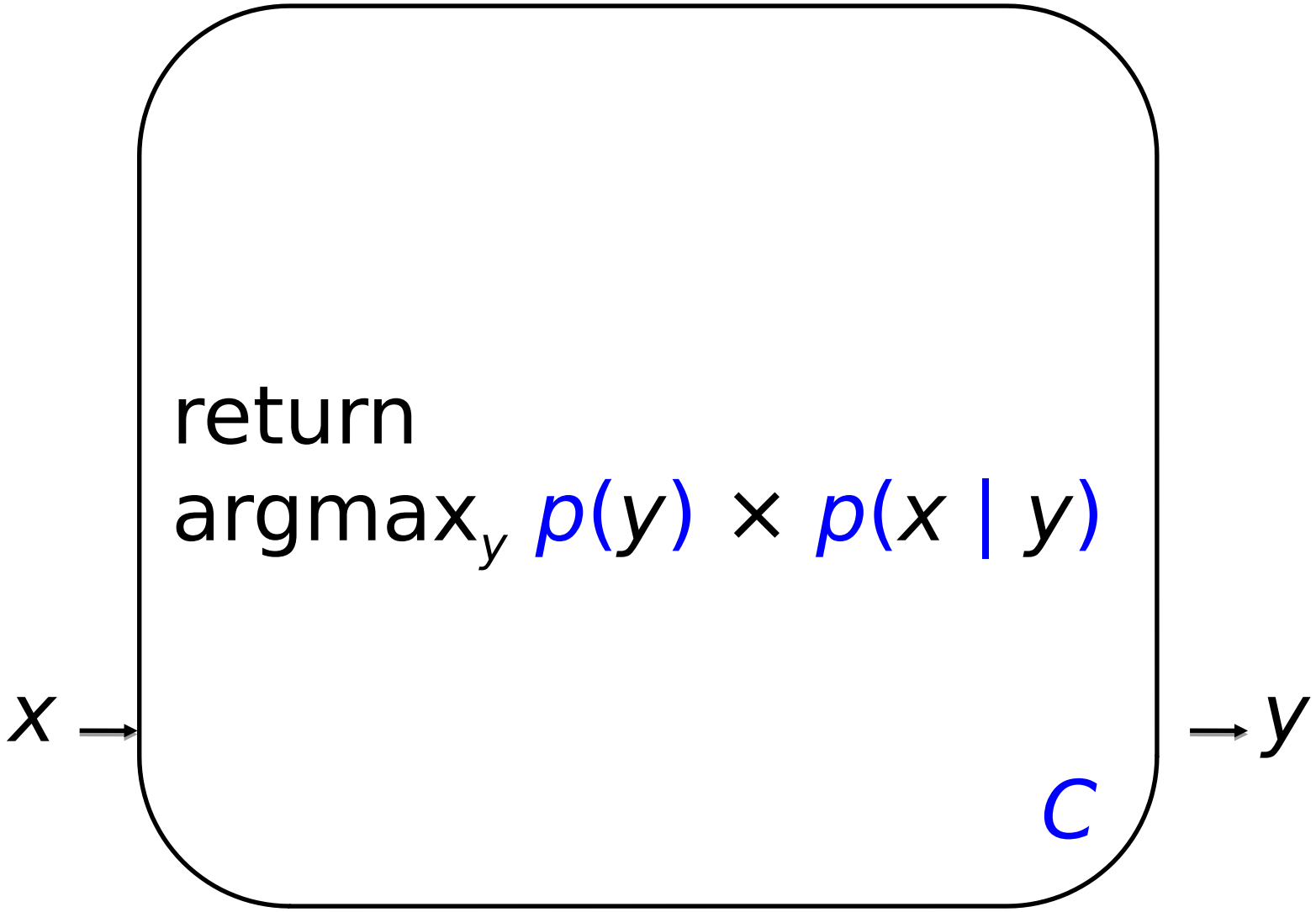
Noisy Channel Model (General)



What proportion of emails are expected to be spam vs. not spam?

What proportion of product reviews are expected to get 1,2,3,4,5 stars?

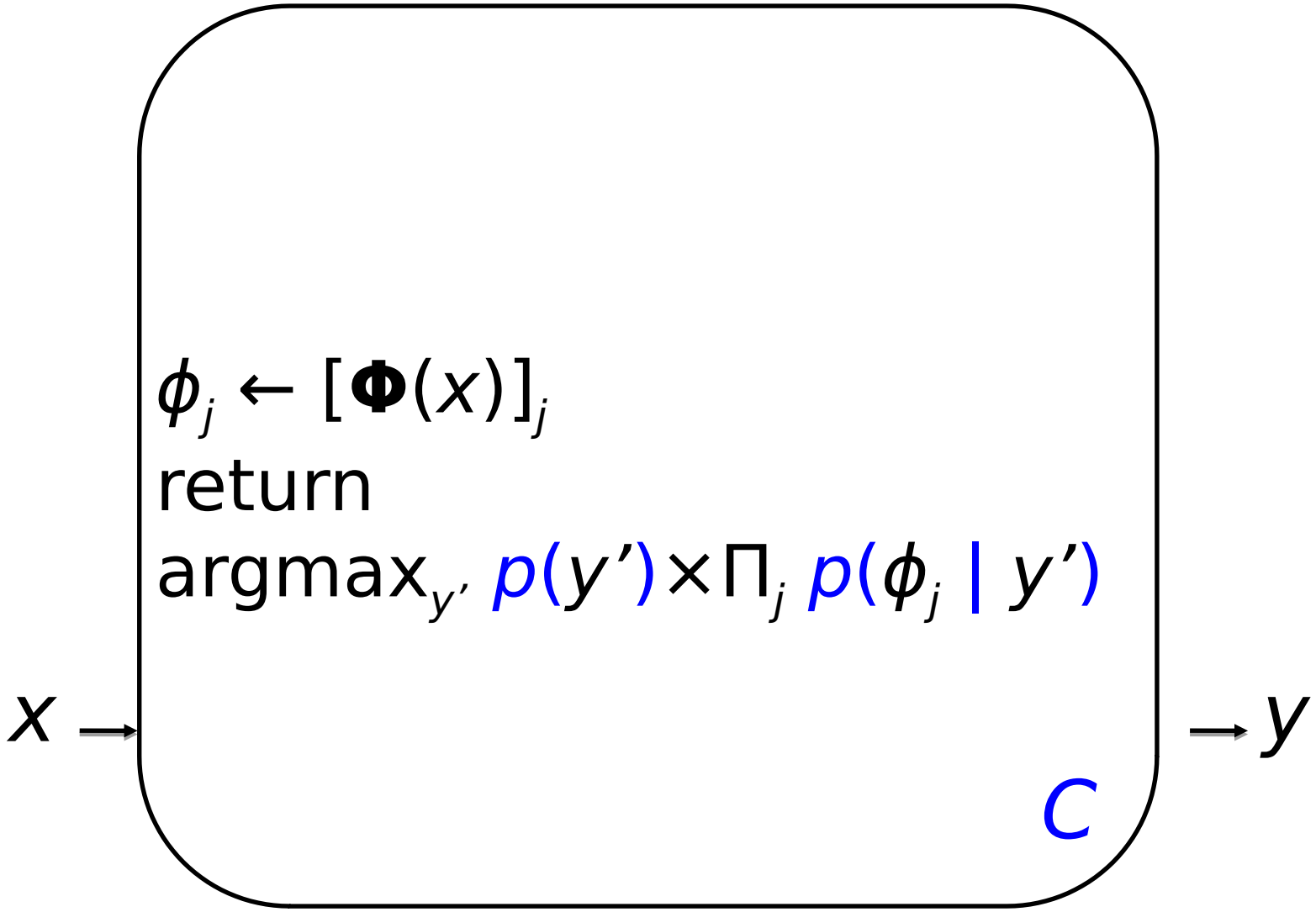
Noisy Channel Classifiers



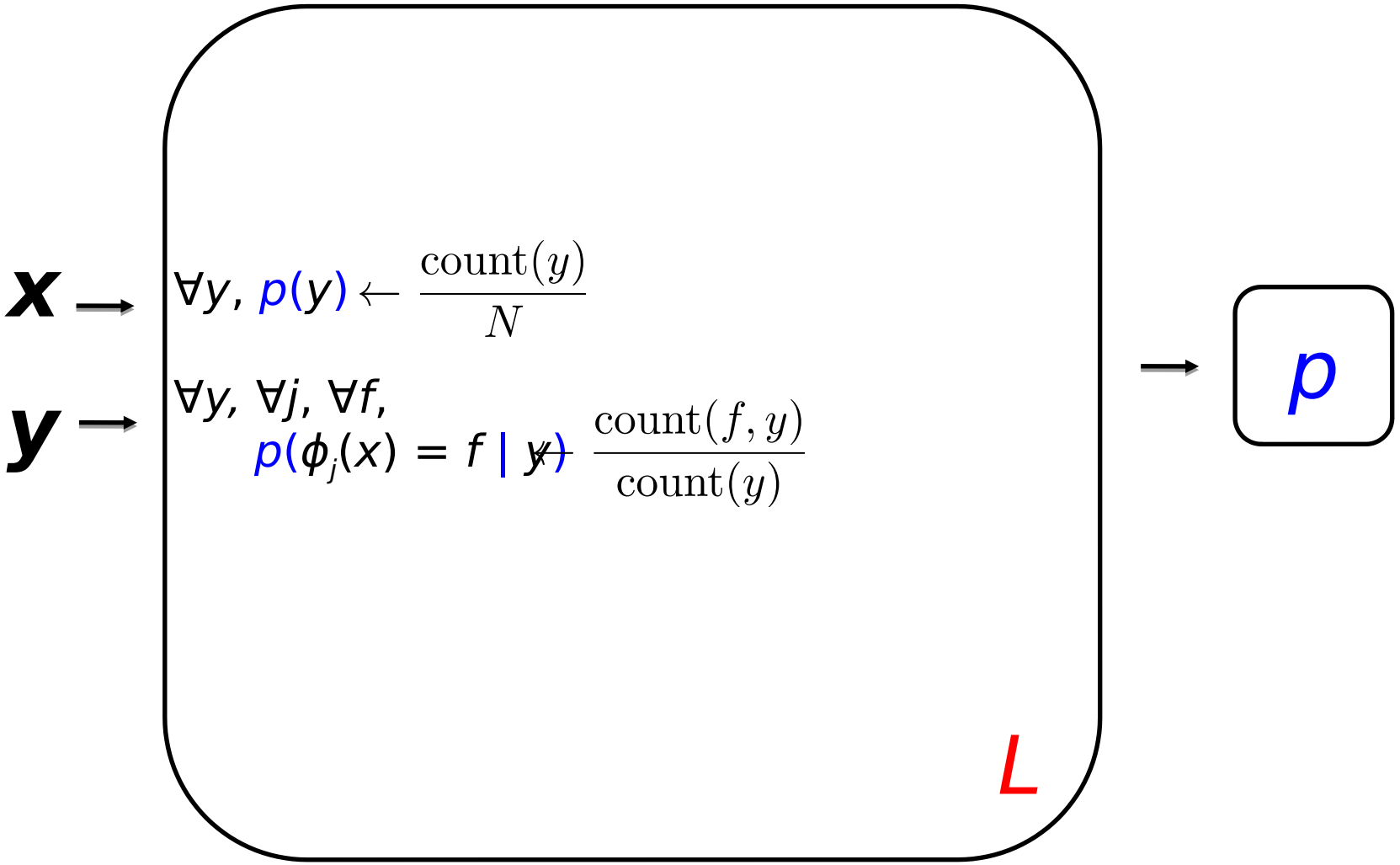
Representing Text: *Features*

- Any object $x \in \mathcal{X}$ you might be given to classify can be represented as a vector in a **vector space**
  - Vectors of representing text are often **sparse** and **high-dimensional**
- Designing Φ ("Feature engineering")
  - What information do you need to solve the problem?
  - What information do you need to avoid mistakes?
  - Very common: bag-of-words

Naïve Bayes Classifier



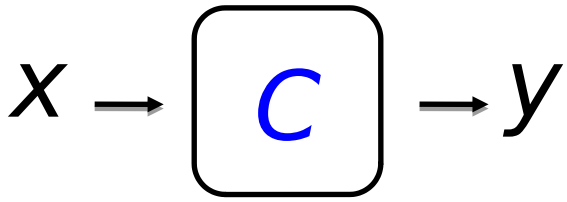
Naïve Bayes Learner



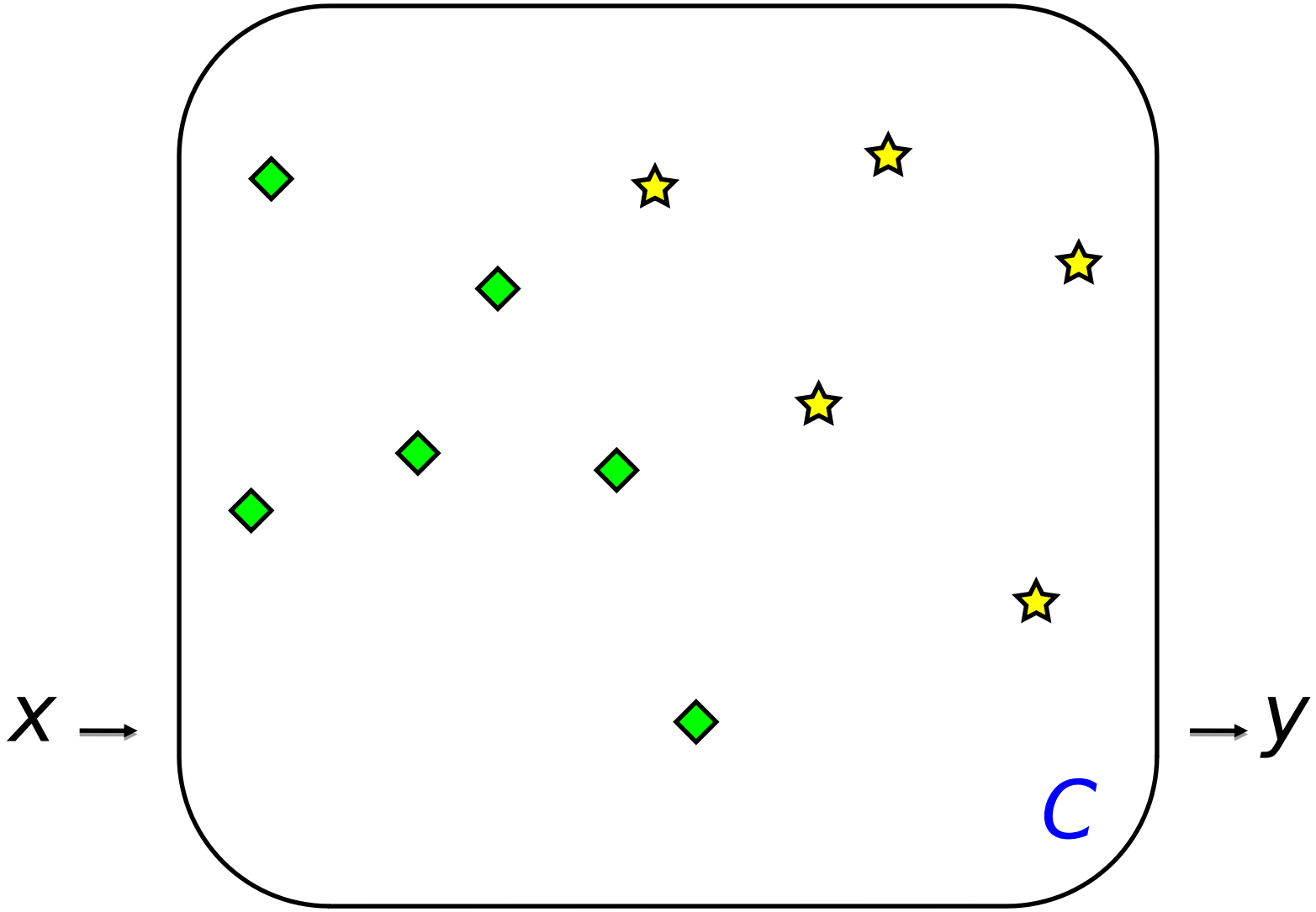
Linear Classifiers

C:

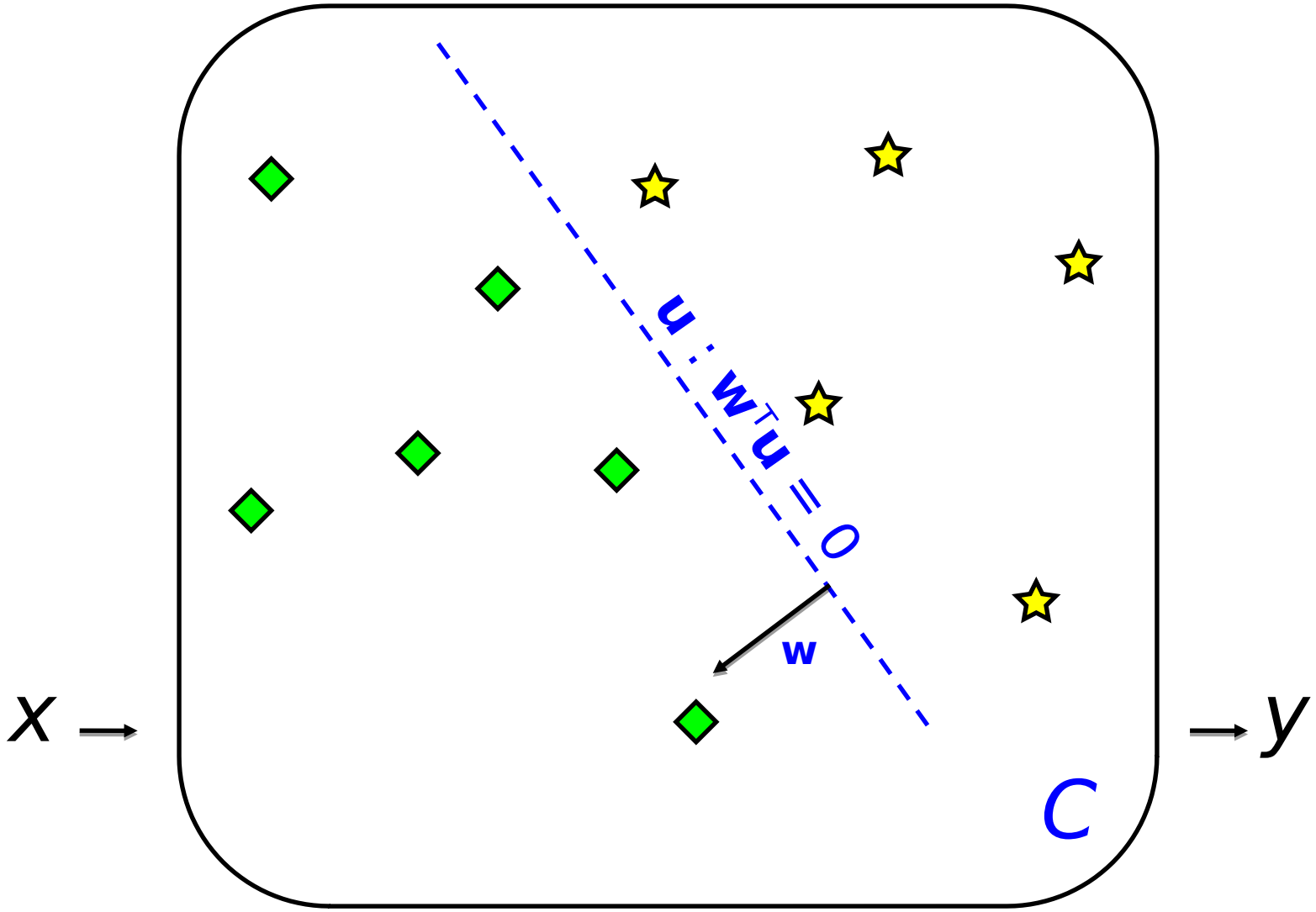
1. Use $\boldsymbol{\Phi}(x)$ to map x onto a **real-valued** feature space.
2. Calculate the linear score $z = \mathbf{w}^{\top} \boldsymbol{\Phi}(x)$.
3. If $z > 0$, then return y = **YES**, else y = **NO**.



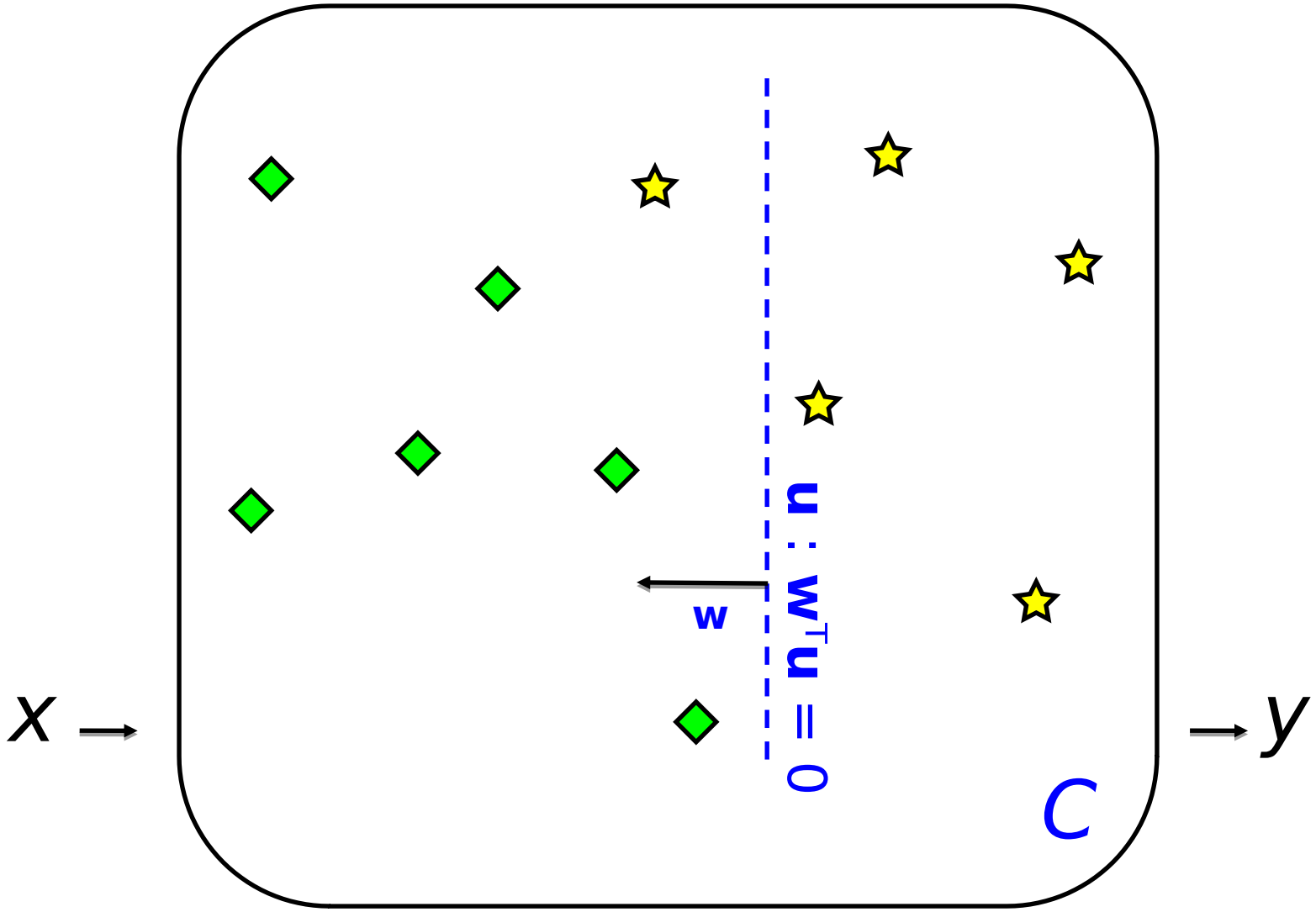
Linear Classifiers



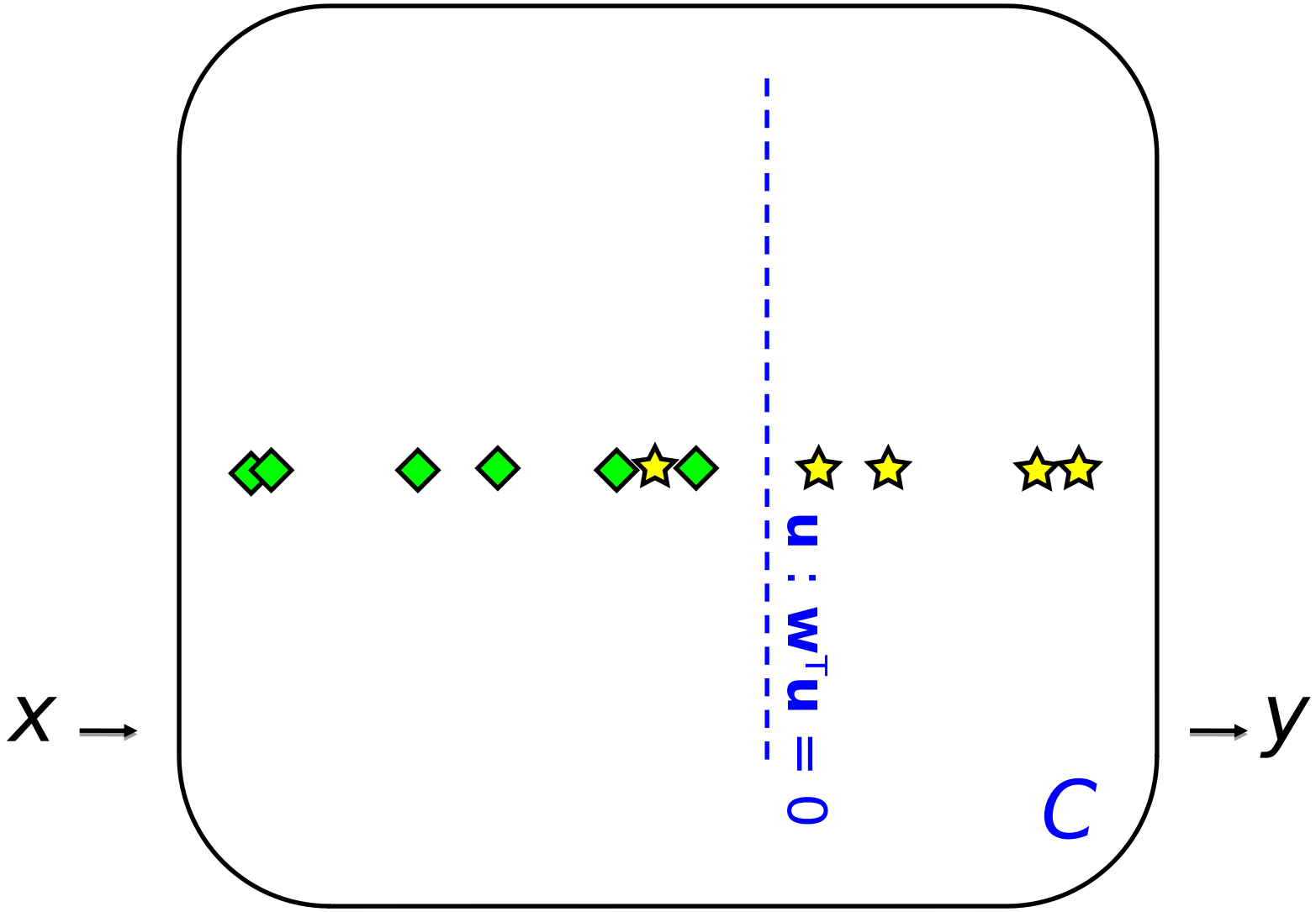
Linear Classifiers



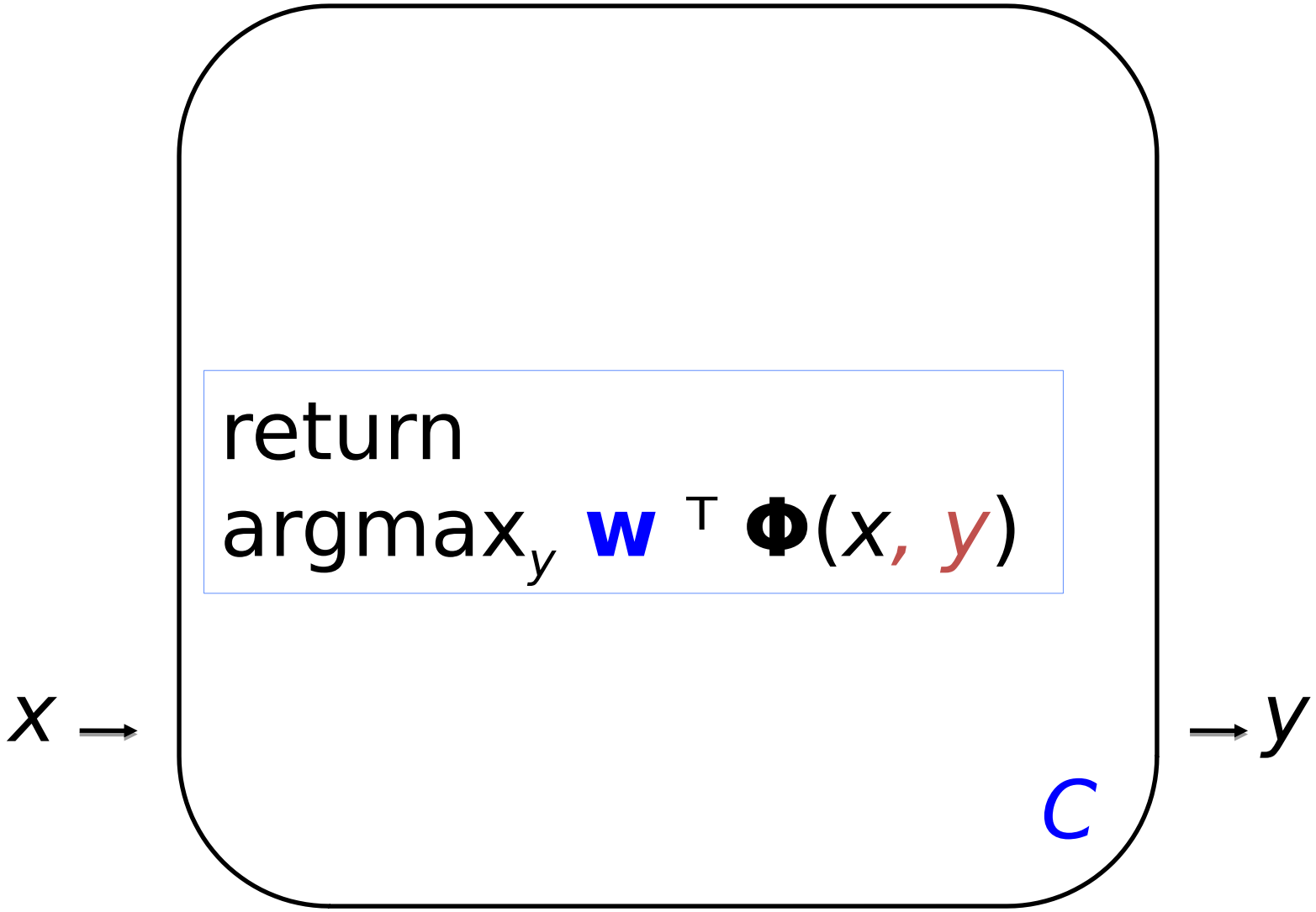
Linear Classifiers



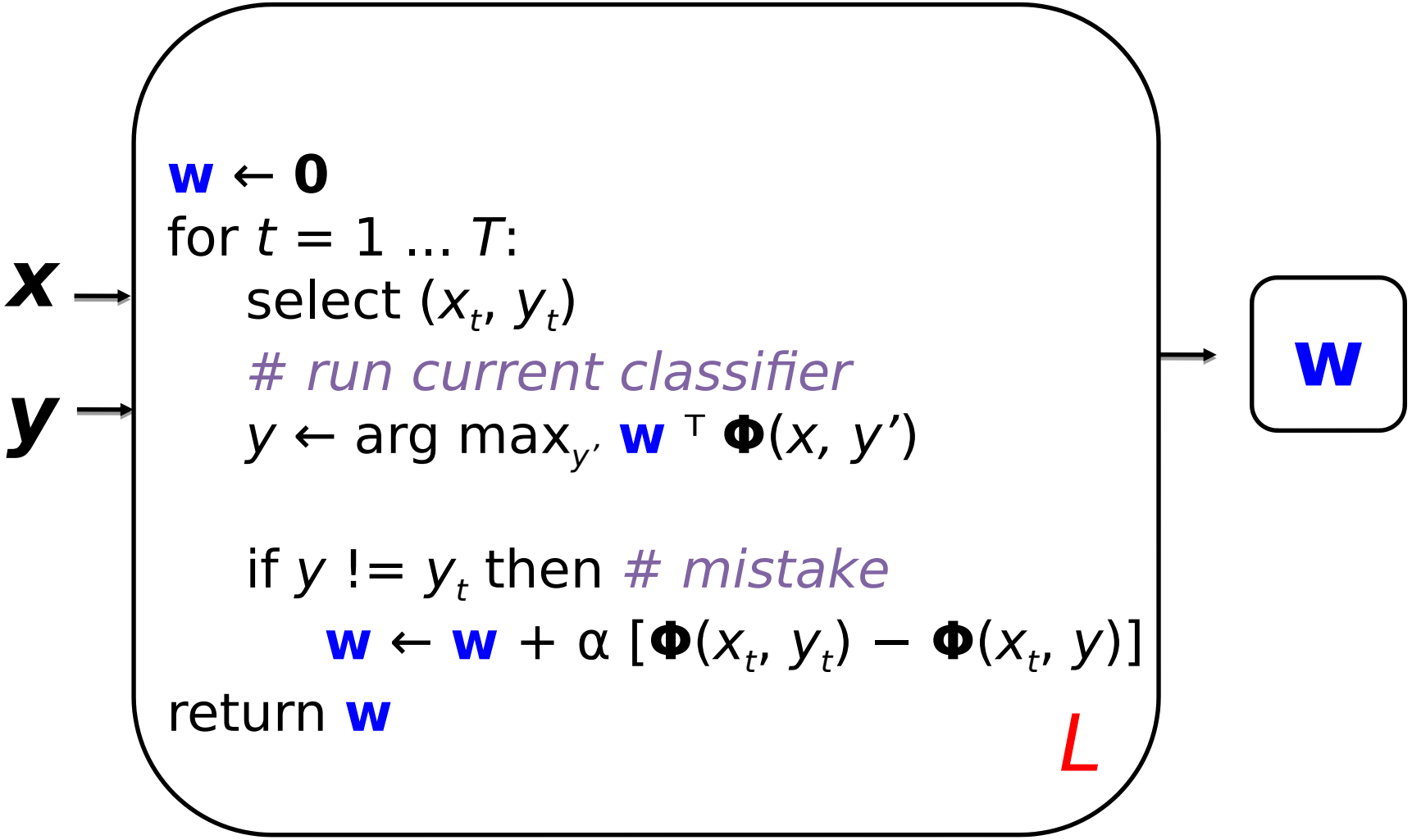
Linear Classifiers



Linear Classifiers (> 2 Classes)



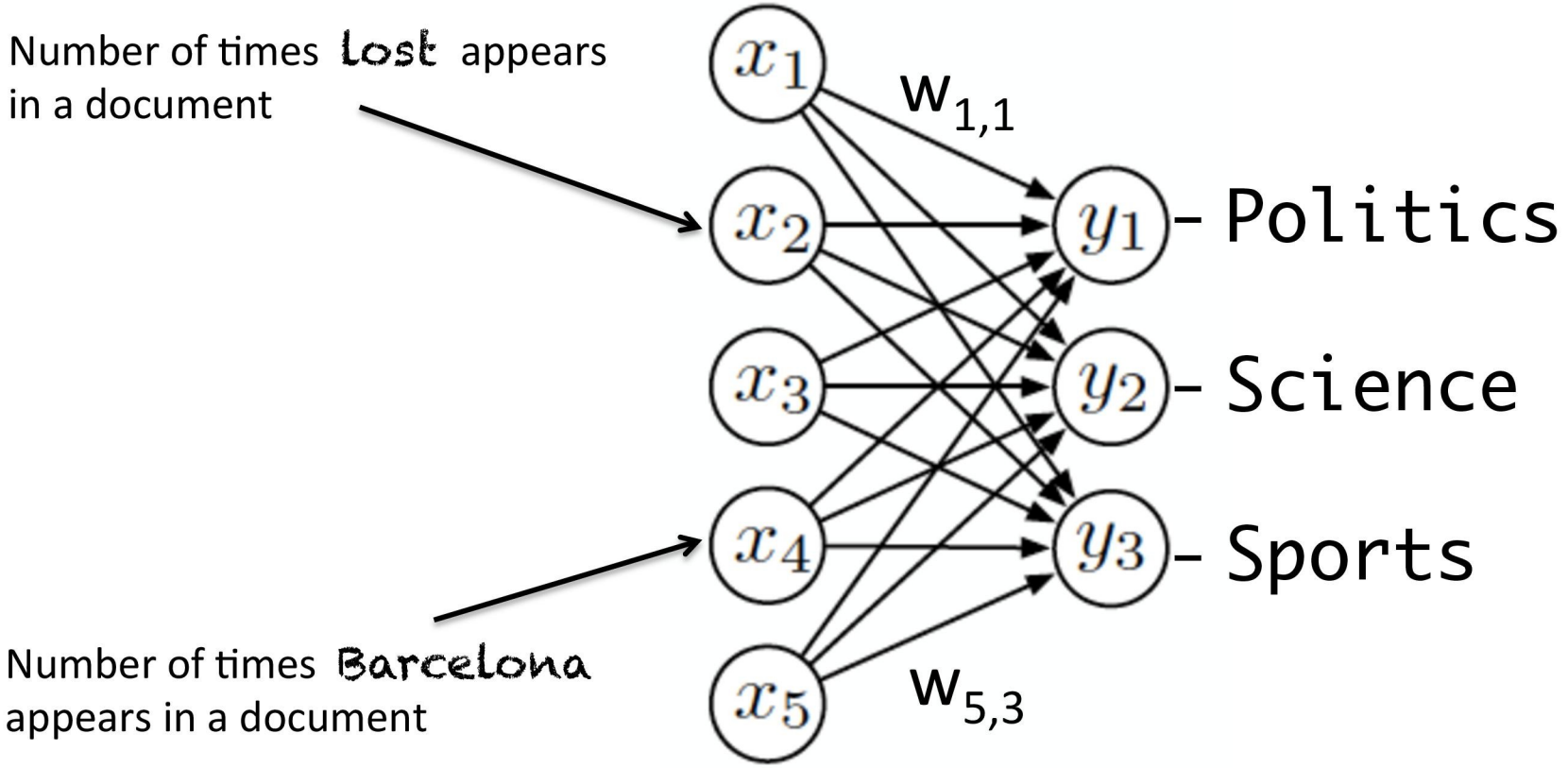
Perceptron Learner



neural network v1.0: linear model

Linear models: f(l, d) = w . g(l,d) = w(l) . x(d)

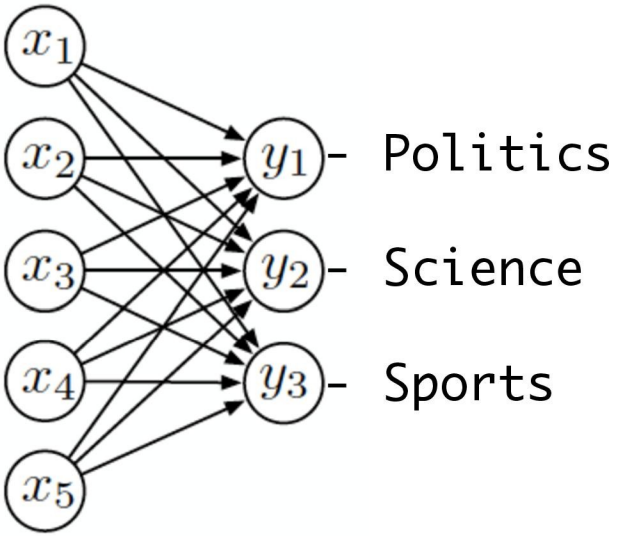
e.g., $y1 = x_1 w_{1,1} + x_2 w_{2,1} + x_3 w_{3,1} + x_4 w_{4,1} + x_5 w_{5,1} = w(1) . x(d)$



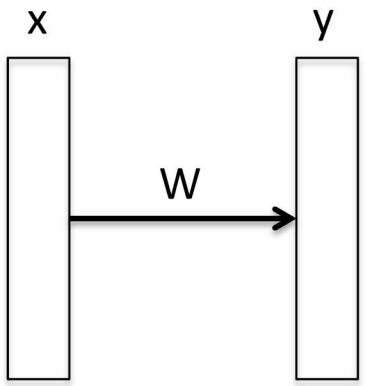
neural network v1.0: linear model

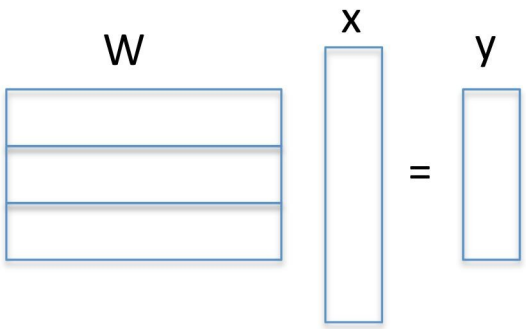
Linear models: f(l, d) = w . g(l,d) = w(l) . x(d)

e.g., $y1 = x_1 w_{1,1} + x_2 w_{2,1} + x_3 w_{3,1} + x_4 w_{4,1} + x_5 w_{5,1} = w(1) . x(d)$



same as

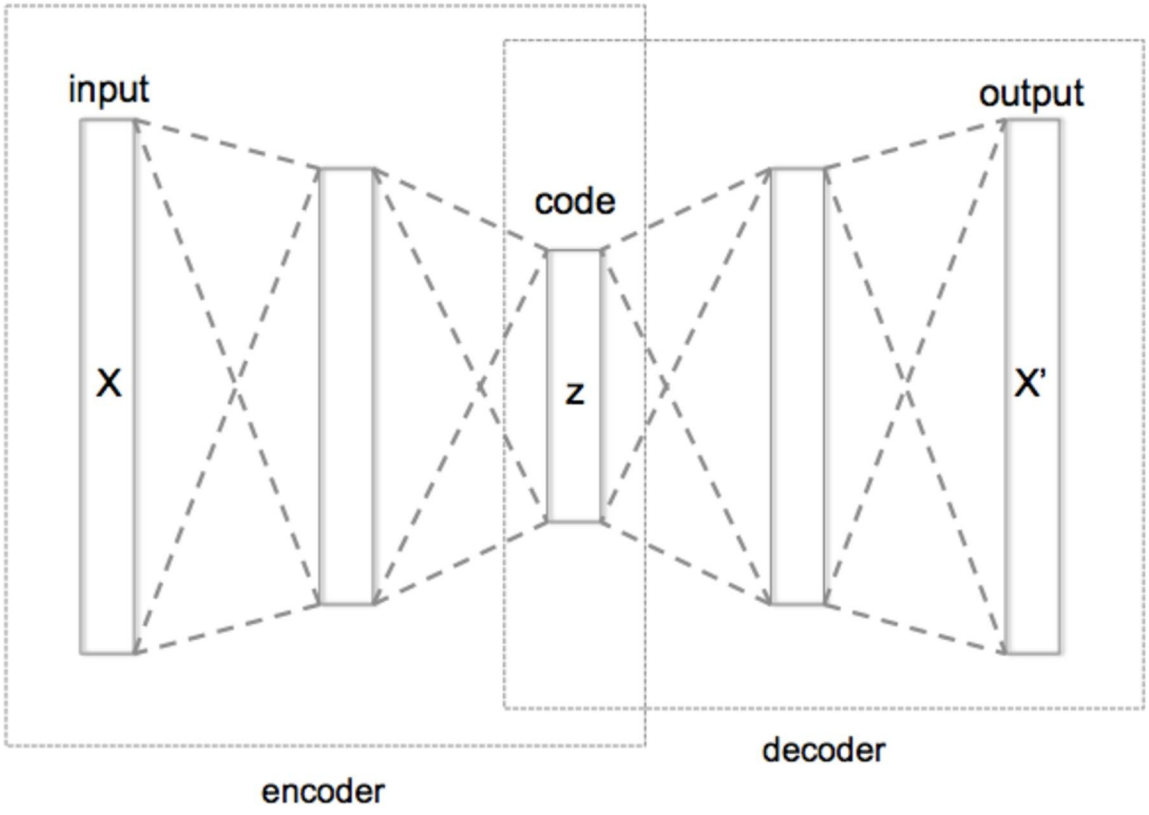




similar words still share no parameters!

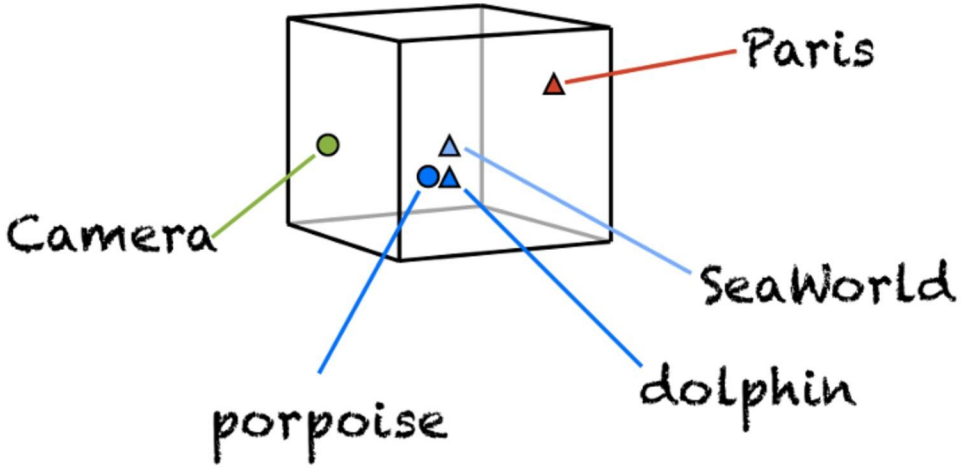
neural network v2.0: representation learning

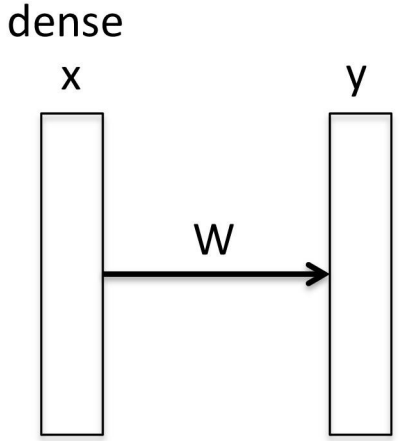
Big idea: induce low-dimensional dense feature representations of high-dimensional objects



neural network v2.1: representation learning

Big idea: embed words in a dense vector space and use the word embeddings as dense features

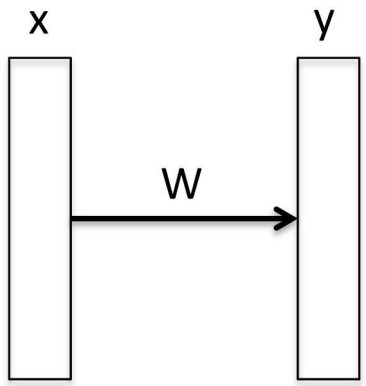




Did this really solve the problem?

neural network v3.0: complex functions

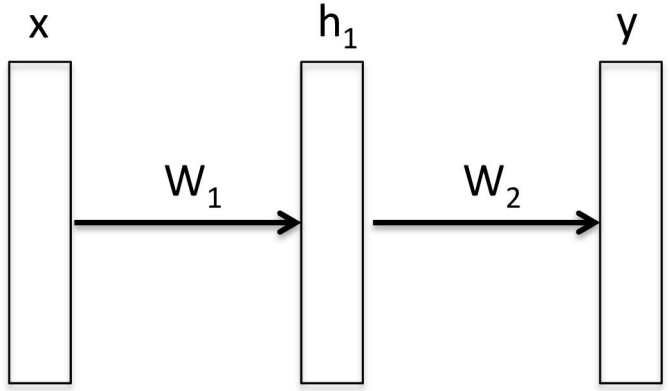
Big idea: define more complex functions by adding a hidden layer



$$y = W x$$

neural network v3.0: complex functions

Big idea: define more complex functions by adding a hidden layer

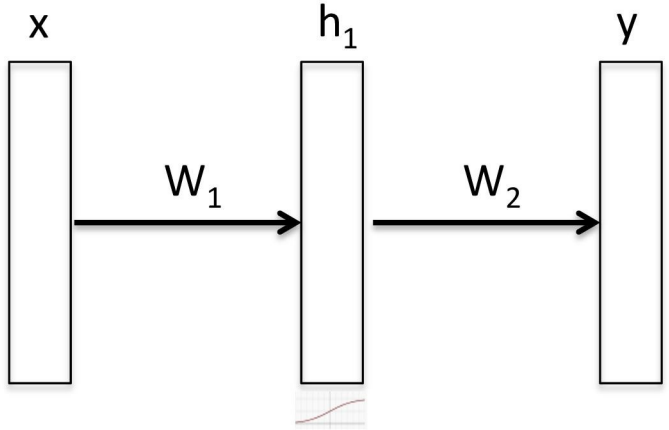


$$y = W_2\, h_1 = W_2\, (W_1\, x)$$
$$= W\, x$$

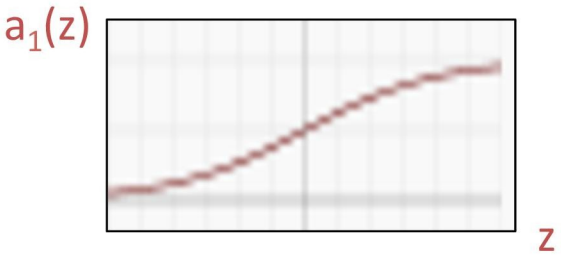
Wait! Is that true?!

neural network v3.0: complex functions

Big idea: define more complex functions by adding a hidden layer



non-linear functions,
e.g., logistic function
$a_1(z) = 1 / (1 + e^{-z})$



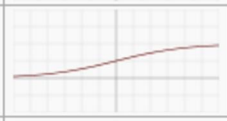
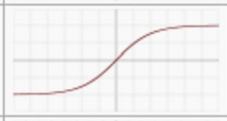
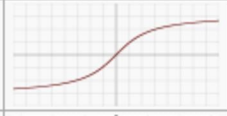
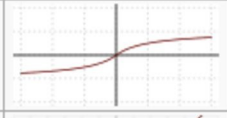
$$y = W_2 h_1 = W_2 \boxed{a_1(W_1 x)}$$

induced features

Universal approximation theorem
Cybenko., G. (1989)

neural network v3.0: complex functions

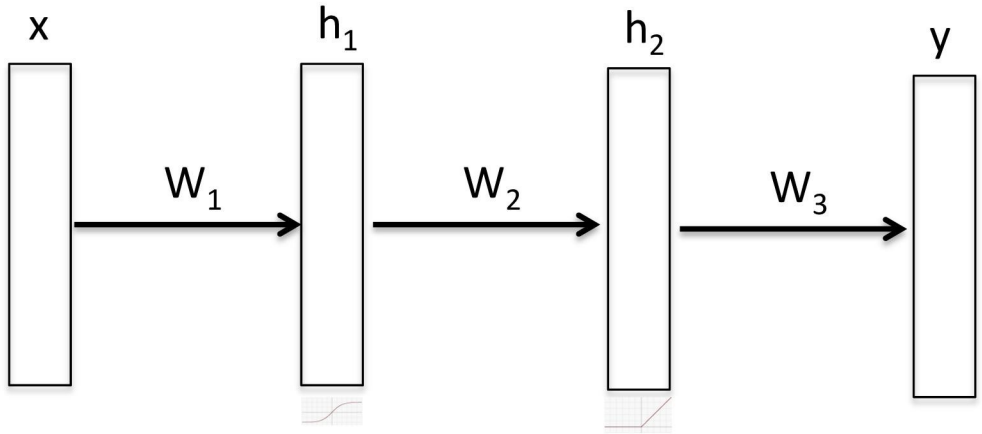
Popular activation/transfer/non-linear functions:

Name	Plot	Equation	Derivative	Range	Order continu
Identity		$f(x) = x$	$f'(x) = 1$	$(-\infty, \infty)$	C^∞
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$	$\{0, 1\}$	C^{-1}
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$	$f'(x) = f(x)(1 - f(x))$	$(0, 1)$	C^∞
TanH		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$	$(-1, 1)$	C^∞
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2 + 1}$	$(-\frac{\pi}{2}, \frac{\pi}{2})$	C^∞
Softsign [7]		$f(x) = \frac{x}{1 + \lvert x \rvert}$	$f'(x) = \frac{1}{(1 + \lvert x \rvert)^2}$	$(-1, 1)$	C^1
Rectified Linear Unit (ReLU)[8]		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$[0, \infty)$	C^0

https://en.wikipedia.org/wiki/Activation_function

neural network v3.5: “deeper” networks

Wait but why do we need more layers?



$$y = W_3 h_2 = W_3 a_2(W_2 a_1(W_1 x))$$

Classifiers

- Learners and Classifiers
- Noisy Channel
- Naive Bayes
- Linear Classifiers
- Perceptrons
- Neural Networks