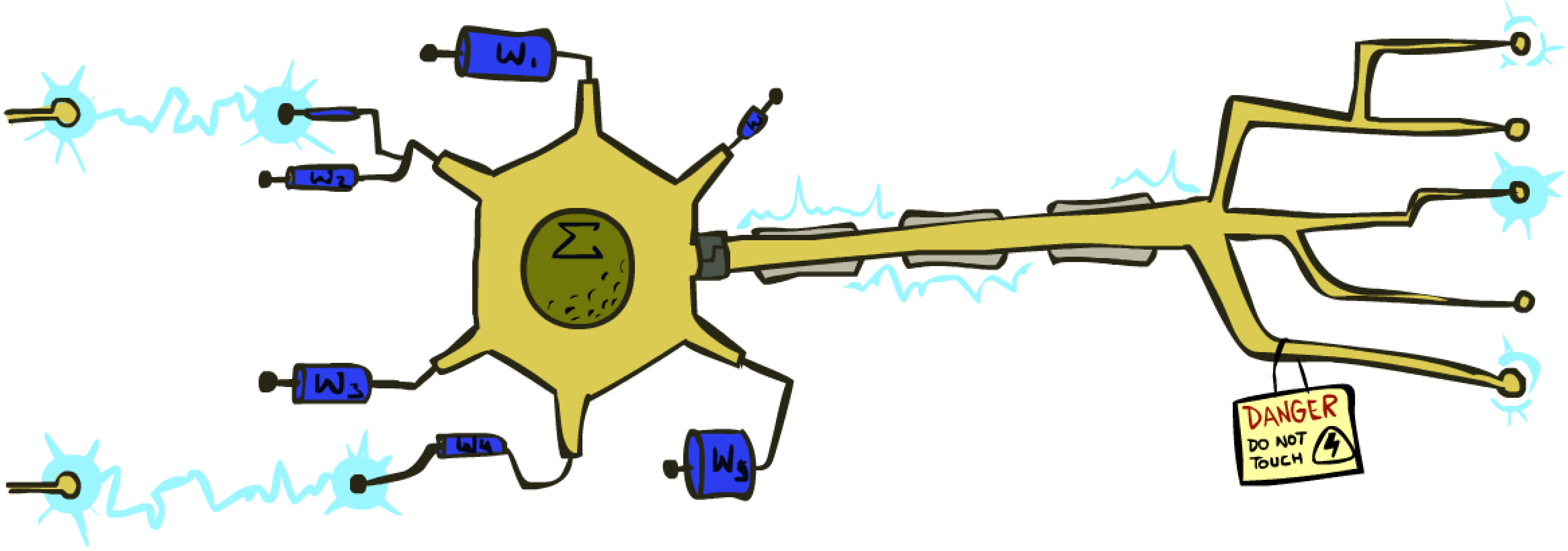
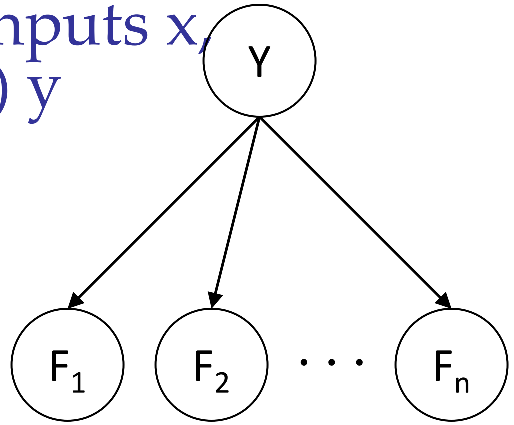
感知机和罗吉斯特回归

Perceptrons and Logistic Regression



上次的内容

- Classification: given inputs x, predict labels (classes) y



- Naïve Bayes

$$P(Y|F_{0,0} \dots F_{15,15}) \propto P(Y) \prod_{i,j} P(F_{i,j}|Y)$$

- Parameter estimation:
  - MLE, MAP, priors

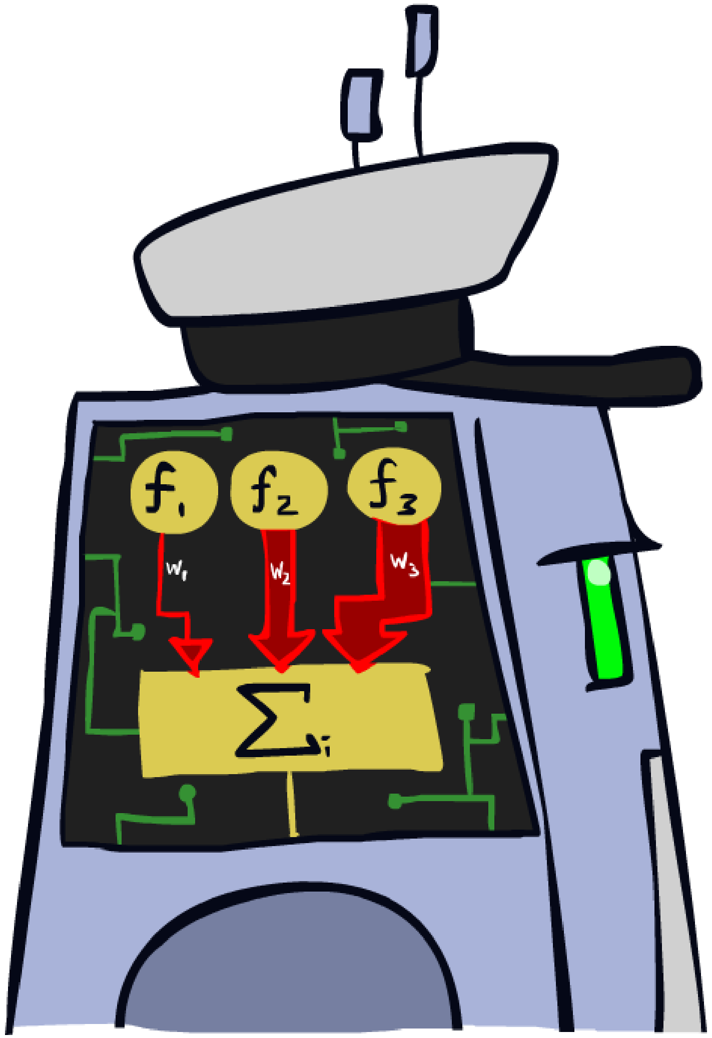
$$P_{\text{ML}}(x) = \frac{\text{count}(x)}{\text{total samples}}$$

- Laplace smoothing

$$P_{LAP,k}(x) = \frac{c(x)+k}{N+k|X|}$$

- Training set, held-out set, test set

线性判别器 Linear Classifiers



特征向量 Feature Vectors

x | $f(x)$ | y

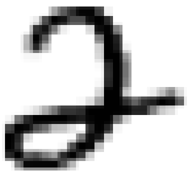
```
Hello,

Do you want free printr
cartriges?  Why pay more
when you can get them
ABSOLUTELY FREE!  Just
```

→

```
# free        : 2
YOUR_NAME     : 0
MISSPELLED    : 2
FROM_FRIEND   : 0
...
```

→ SPAM or +



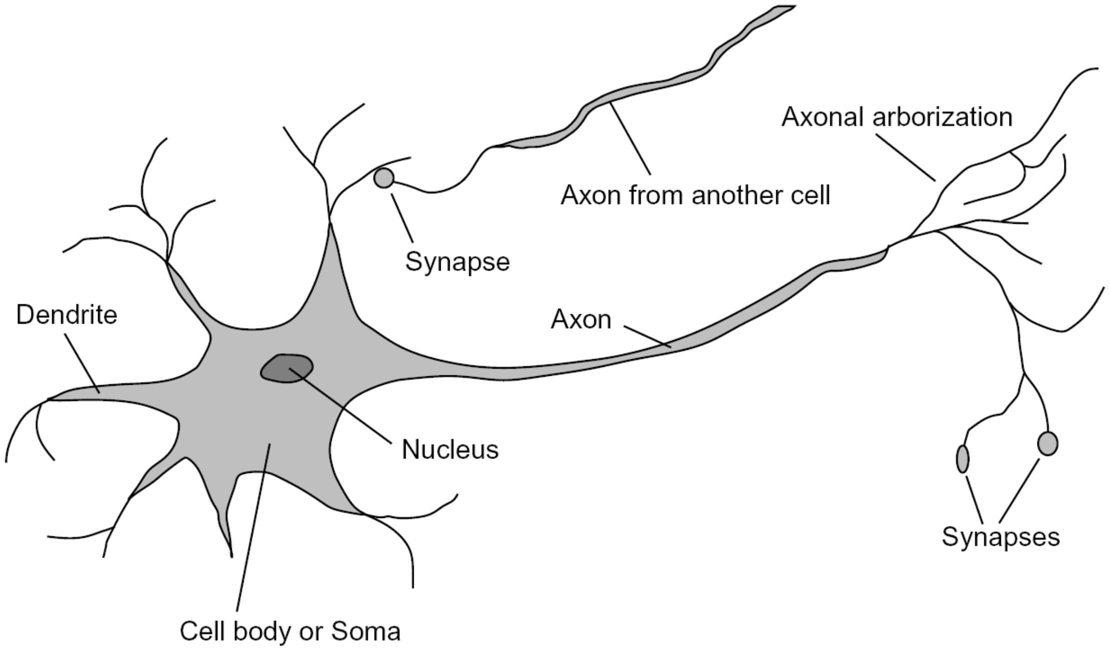
→

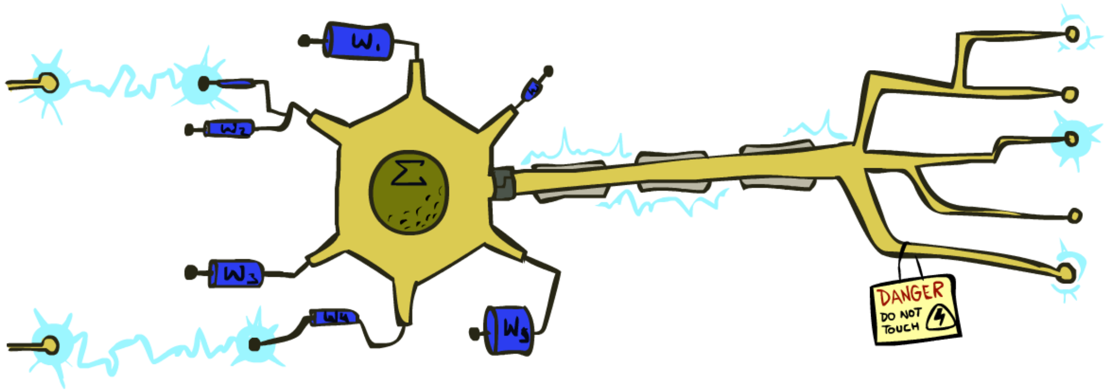
```
PIXEL-7,12  : 1
PIXEL-7,13  : 0
...
NUM_LOOPS   : 1
...
```

→ “2”

生物中的神经元

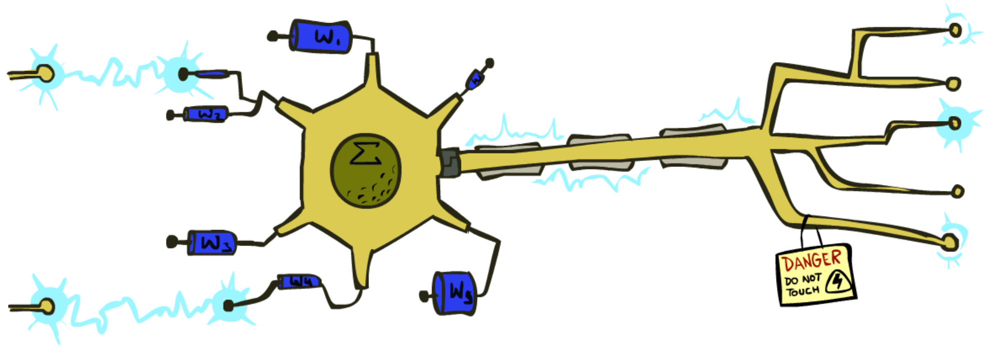
- Very loose inspiration: human neurons





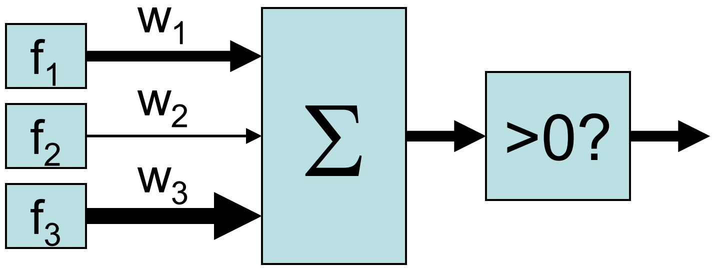
线性分类器 Linear Classifiers

- Inputs are feature values
- Each feature has a weight
- Sum is the activation （激活函数）



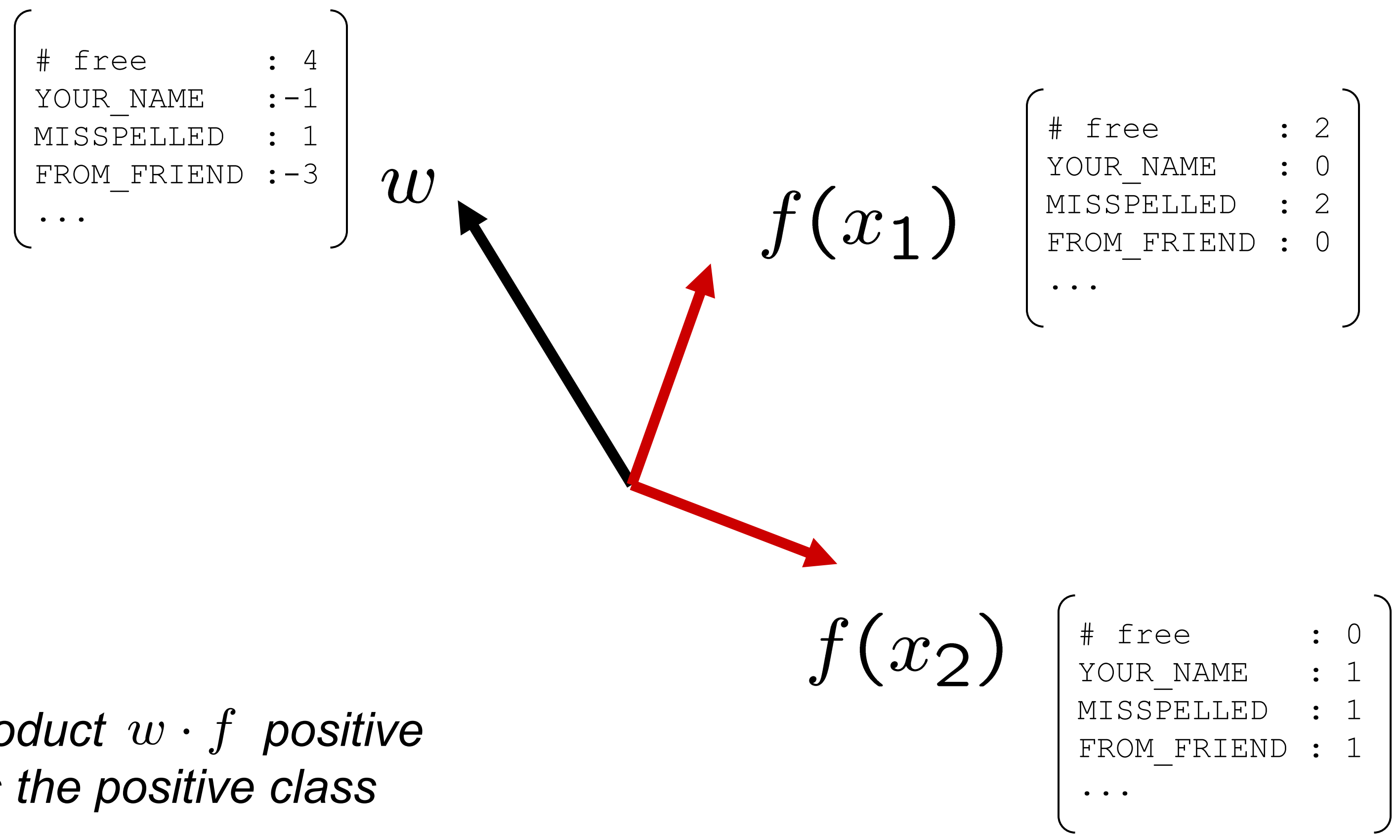
$$\text{activation}_w(x) = \sum_i w_i \cdot f_i(x) = w \cdot f(x)$$

- If the activation is:
  - Positive, output +1
  - Negative, output -1



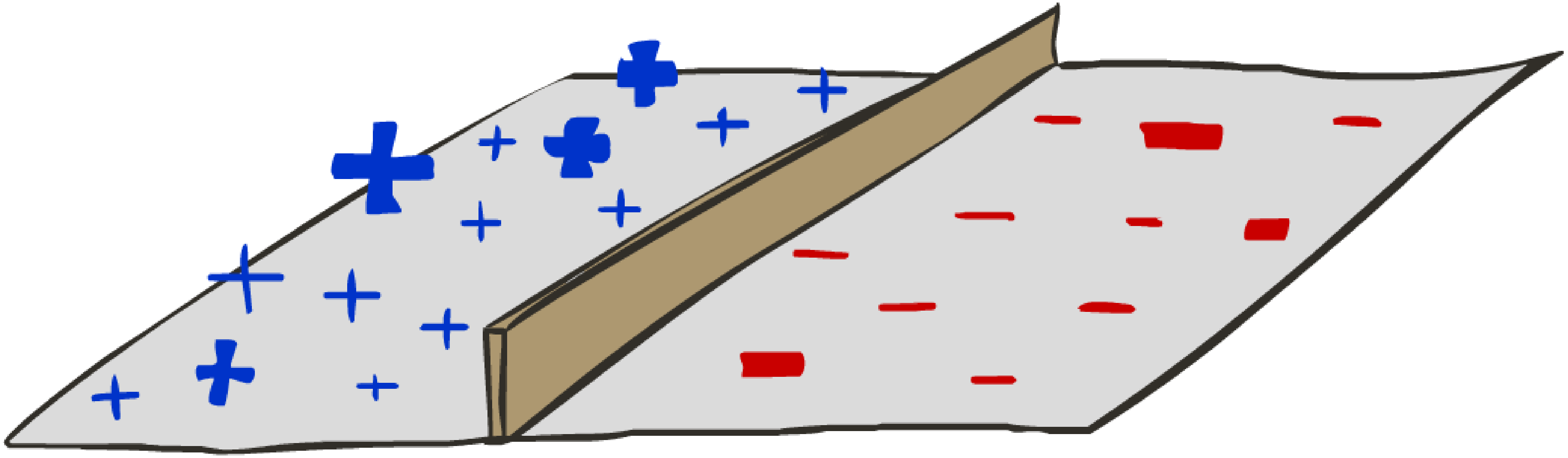
权重向量 Weights

- 二分类: compare features to a weight vector
- Learning: figure out the weight vector from examples



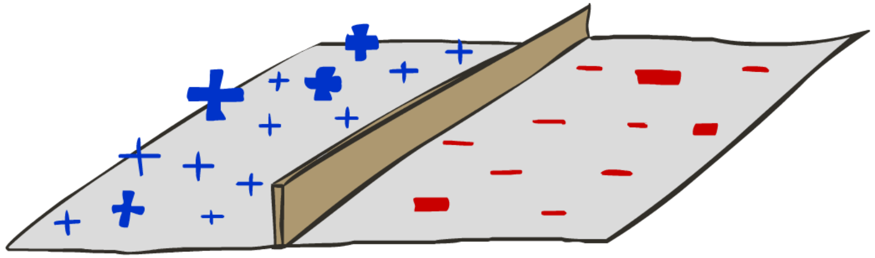
Dot product $w \cdot f$ positive means the positive class

Decision Rules



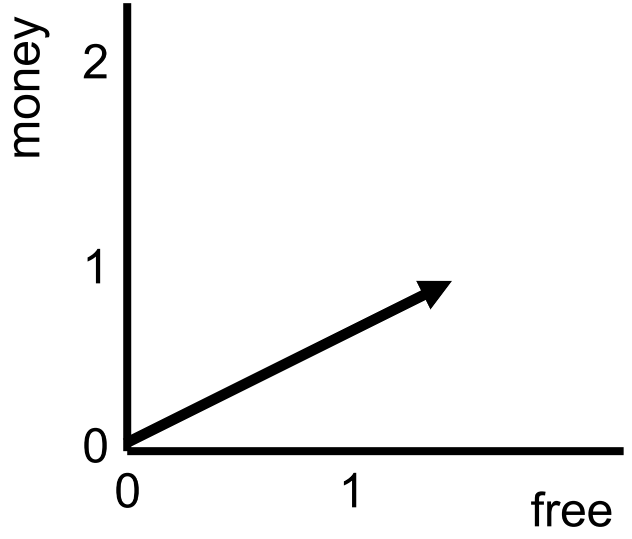
二分决策规则 Binary Decision Rule

- In the space of feature vectors
  - Examples are points
  - Any weight vector is a hyperplane（超平面）
  - One side corresponds to Y=+1
  - Other corresponds to Y=-1



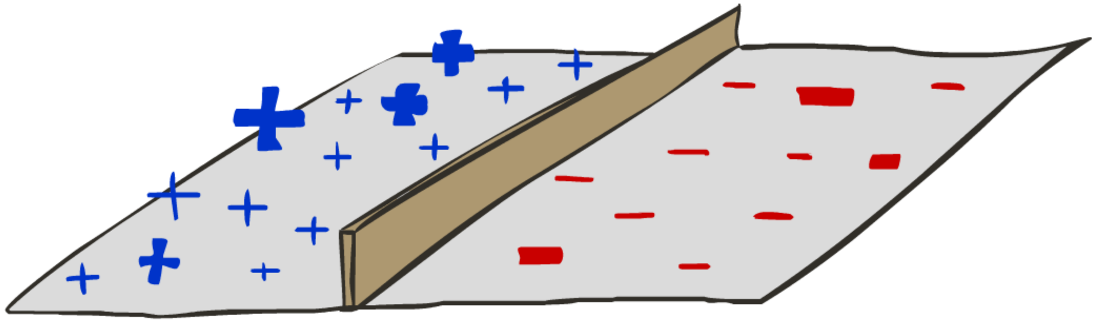
w

```
BIAS  : -3
free  :  4
money :  2
...
```



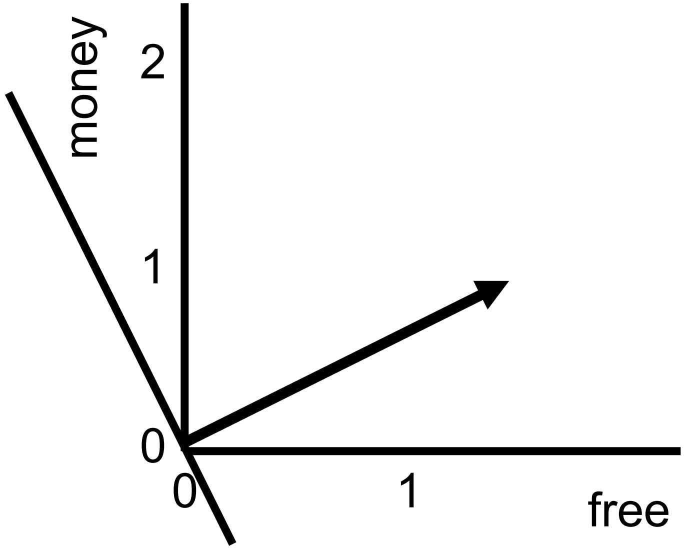
二分决策规则

- In the space of feature vectors
  - Examples are points
  - Any weight vector is a hyperplane
  - One side corresponds to Y=+1
  - Other corresponds to Y=-1



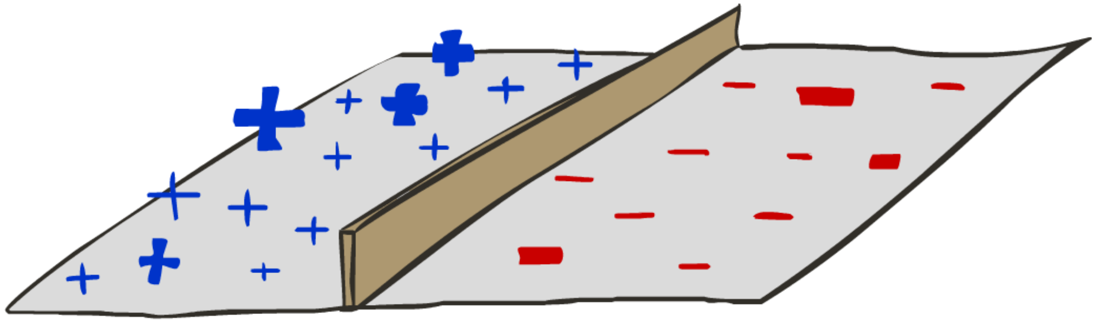
$$w$$

```
BIAS  : -3
free  :  4
money :  2
...
```



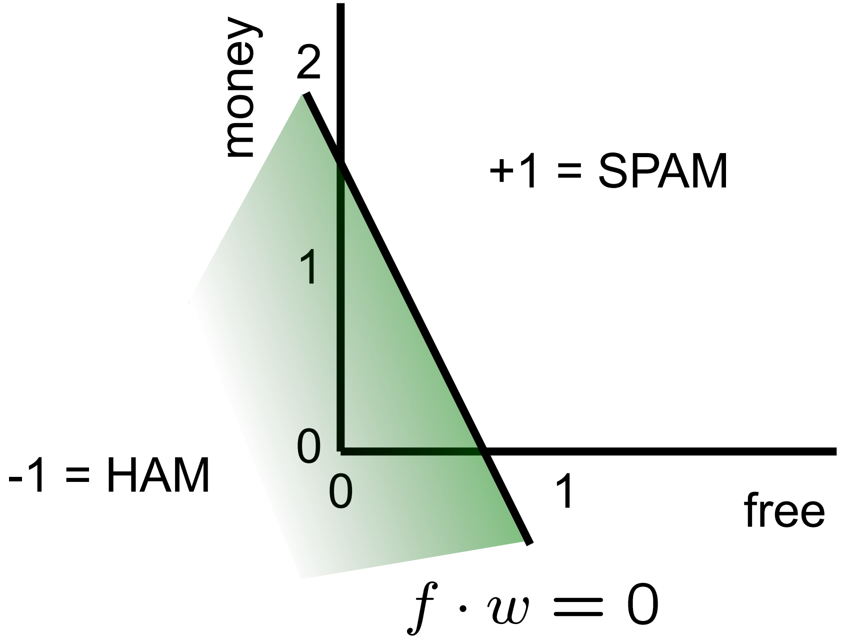
二分决策规则

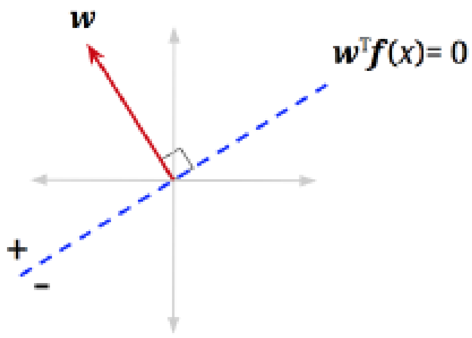
- In the space of feature vectors
  - Examples are points
  - Any weight vector is a hyperplane
  - One side corresponds to Y=+1
  - Other corresponds to Y=-1



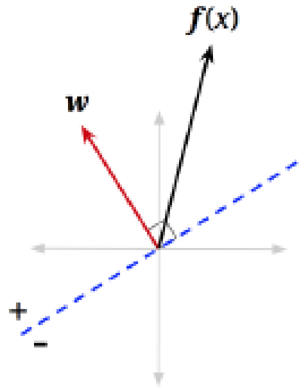
w

```
BIAS  : -3
free  :  4
money :  2
...
```

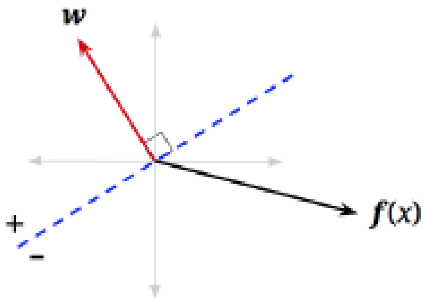




Decision Boundary



x **classified into positive class**

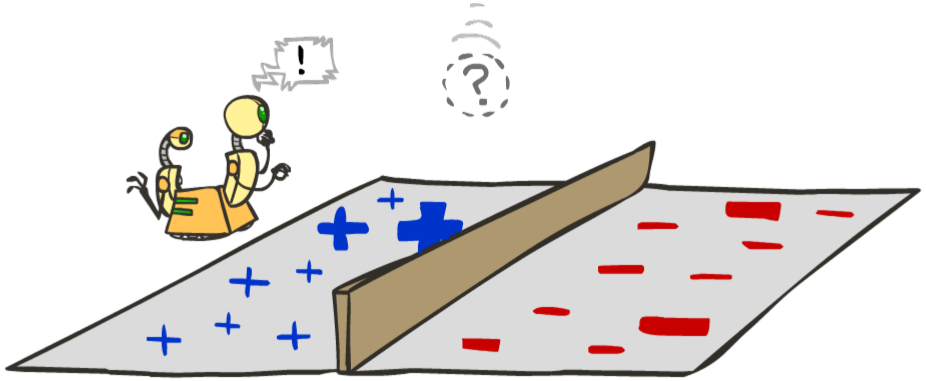


x **classified into negative class**

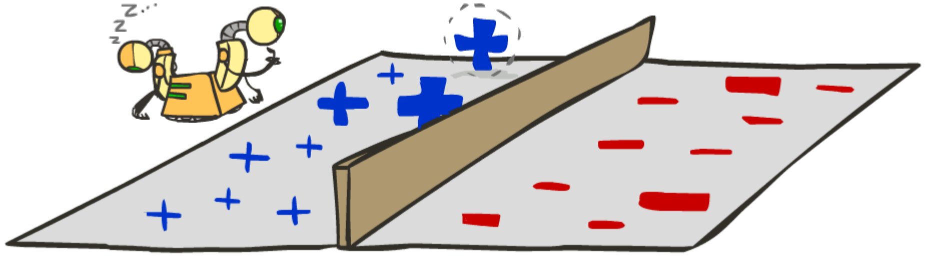
更新权重向量 Weight Updates

学习: 二分感知机

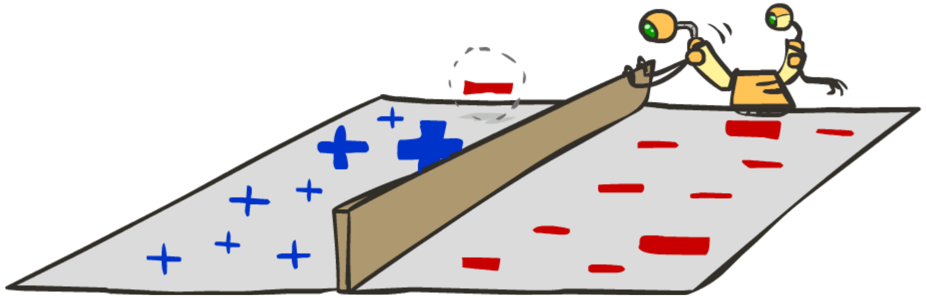
- Start with weights = 0
- For each training instance:
  - Classify with current weights



  - If correct (i.e., y=y*), no change!



  - If wrong: adjust the weight vector



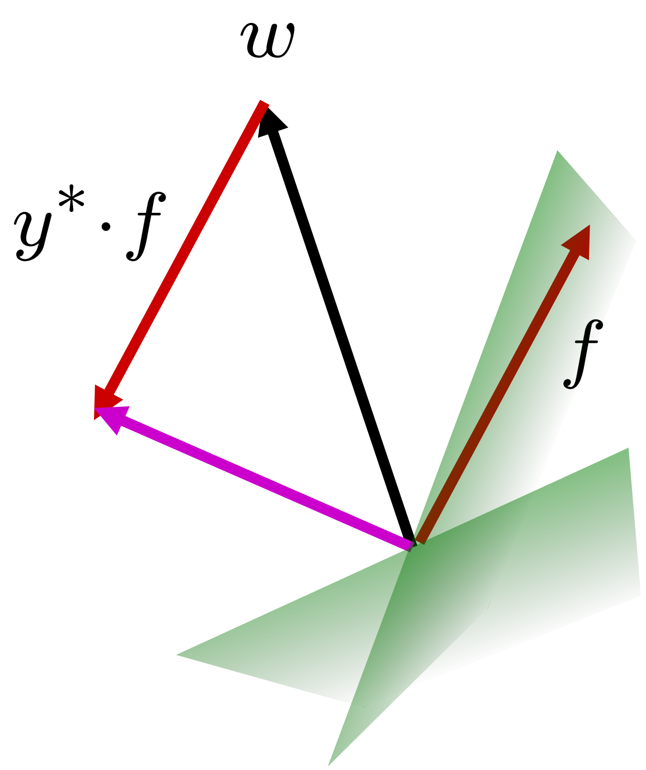
学习: 二分感知机

- Start with weights = 0
- For each training instance:
  - Classify with current weights

$$y = \begin{cases} +1 & \text{if } \ w \cdot f(x) \geq 0 \\ -1 & \text{if } \ w \cdot f(x) < 0 \end{cases}$$

  - If correct (i.e., y=y*), no change!
  - If wrong: adjust the weight vector by adding or subtracting the feature vector. Subtract if y* is -1.

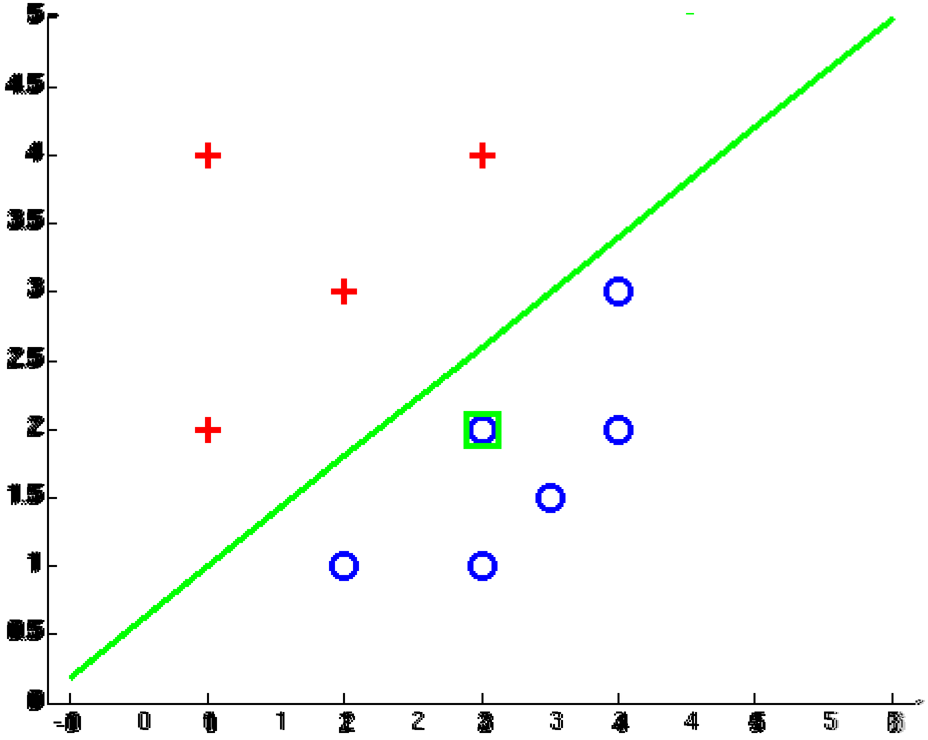
$$w = w + y^* \cdot f$$



Before: $w\,f$
After: $wf + y^*f\,f$
$f\,f$>=0

举例: 感知机

- Separable Case （可分割的情况下）



多类判别规则 Multiclass Decision Rule

- 多类别情况下:
  - A weight vector for each class:

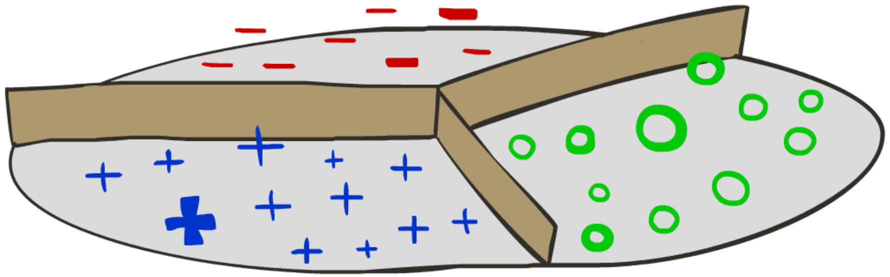
    $$w_y$$

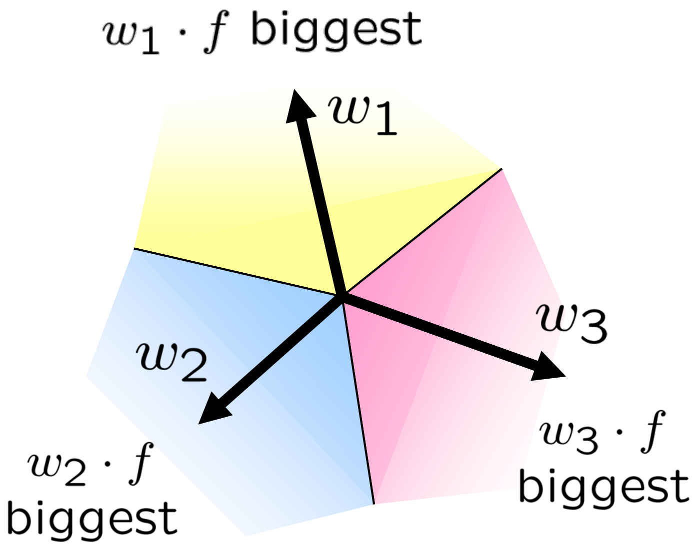
  - Score (activation) of a class y:

    $$w_y \cdot f(x)$$

  - Prediction highest score wins

    $$y = \arg\max_y \; w_y \cdot f(x)$$





Binary = multiclass where the negative class has weight zero

机器学习: 多分类感知机

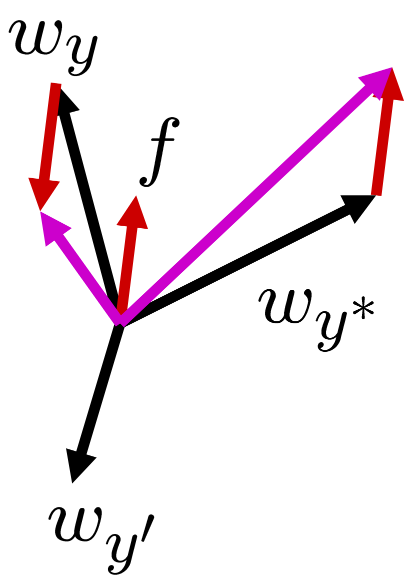
- Start with all weights = 0
- Pick up training examples one by one
- Predict with current weights

$$y = \arg\max_y \; w_y \cdot f(x)$$

- If correct, no change!
- If wrong: lower score of wrong answer, raise score of right answer

$$w_y = w_y - f(x)$$

$$w_{y^*} = w_{y^*} + f(x)$$



举例:多分类感知机

"win the vote" [1 1 0 1 1]

"win the election" [1 1 0 0 1]

"win the game" [1 1 1 0 1]

w_{SPORTS}

1 -2 -2

```
BIAS  : 1
win   : 0
game  : 0
vote  : 0
the   : 0
...
```

0	1
-1	0
0	1
-1	-1
-1	0

$w_{POLITICS}$

0 3 3

```
BIAS  : 0
win   : 0
game  : 0
vote  : 0
the   : 0
...
```

1	0
1	0
0	-1
1	1
1	0

w_{TECH}

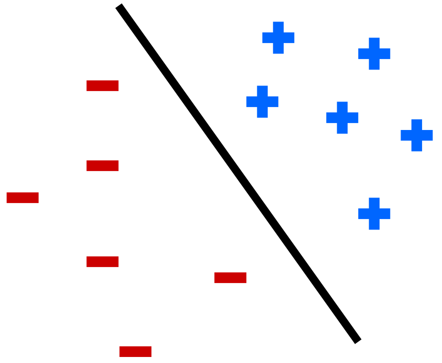
0 0

```
BIAS  : 0
win   : 0
game  : 0
vote  : 0
the   : 0
...
```

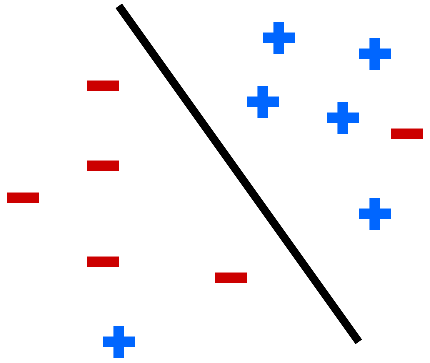
感知机的属性

- Separability 可分割: true if some parameters get the training set perfectly correct
- Convergence 收敛: if the training is separable, perceptron will eventually converge (binary case)
- Mistake Bound 误判率: the maximum number of mistakes (binary case) related to the *margin* or degree of separability

Separable

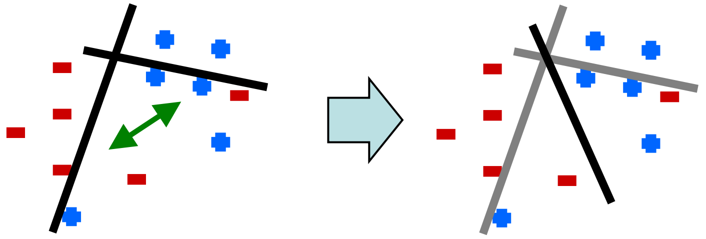


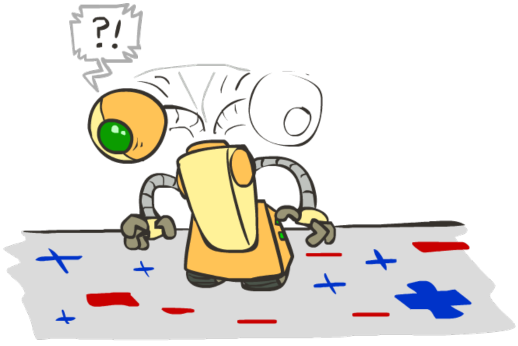
Non-Separable



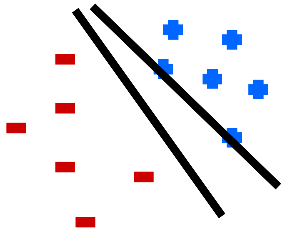
感知机存在的问题

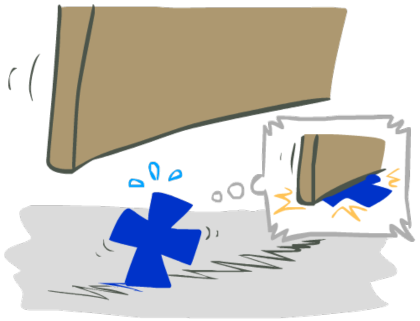
- Noise: if the data isn't separable, weights might thrash 难以收敛
  - Averaging weight vectors over time can help (averaged perceptron)



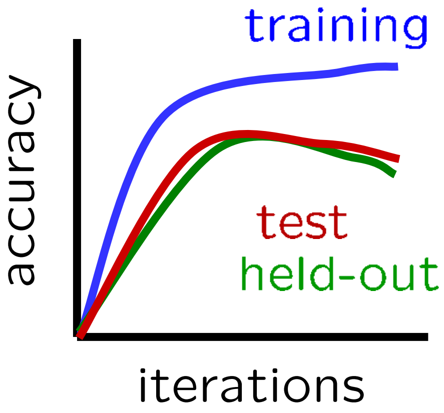


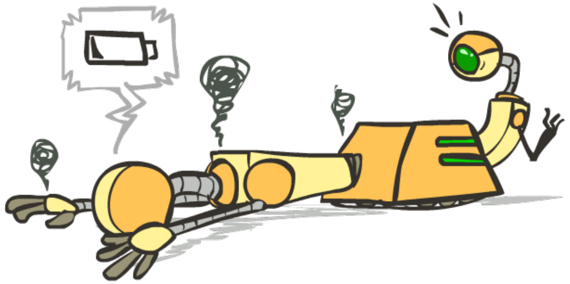
- Mediocre generalization 泛化学习性不强: finds a "barely" separating solution



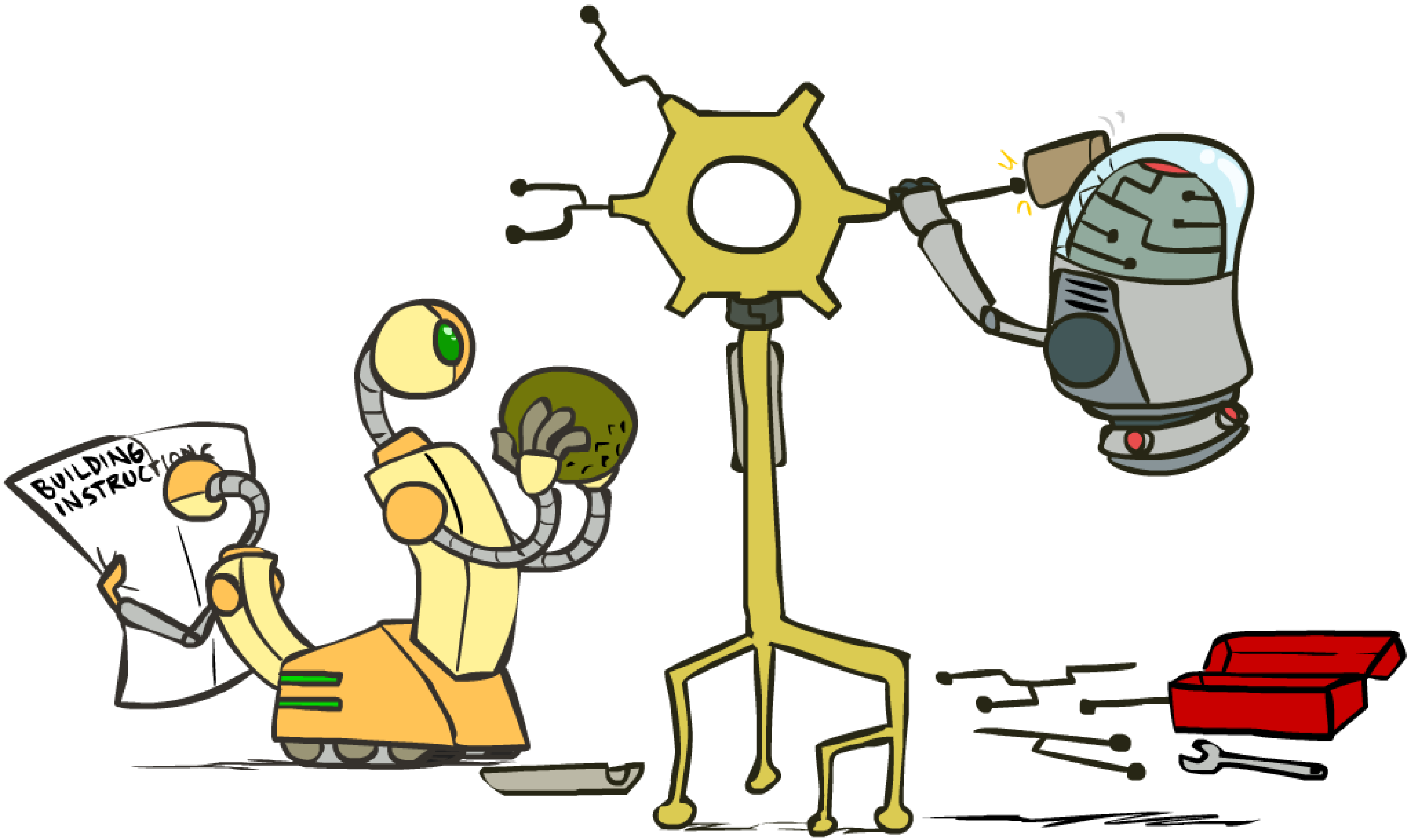


- Overtraining 易过拟合: test / held-out accuracy usually rises, then falls
  - Overtraining is a kind of overfitting

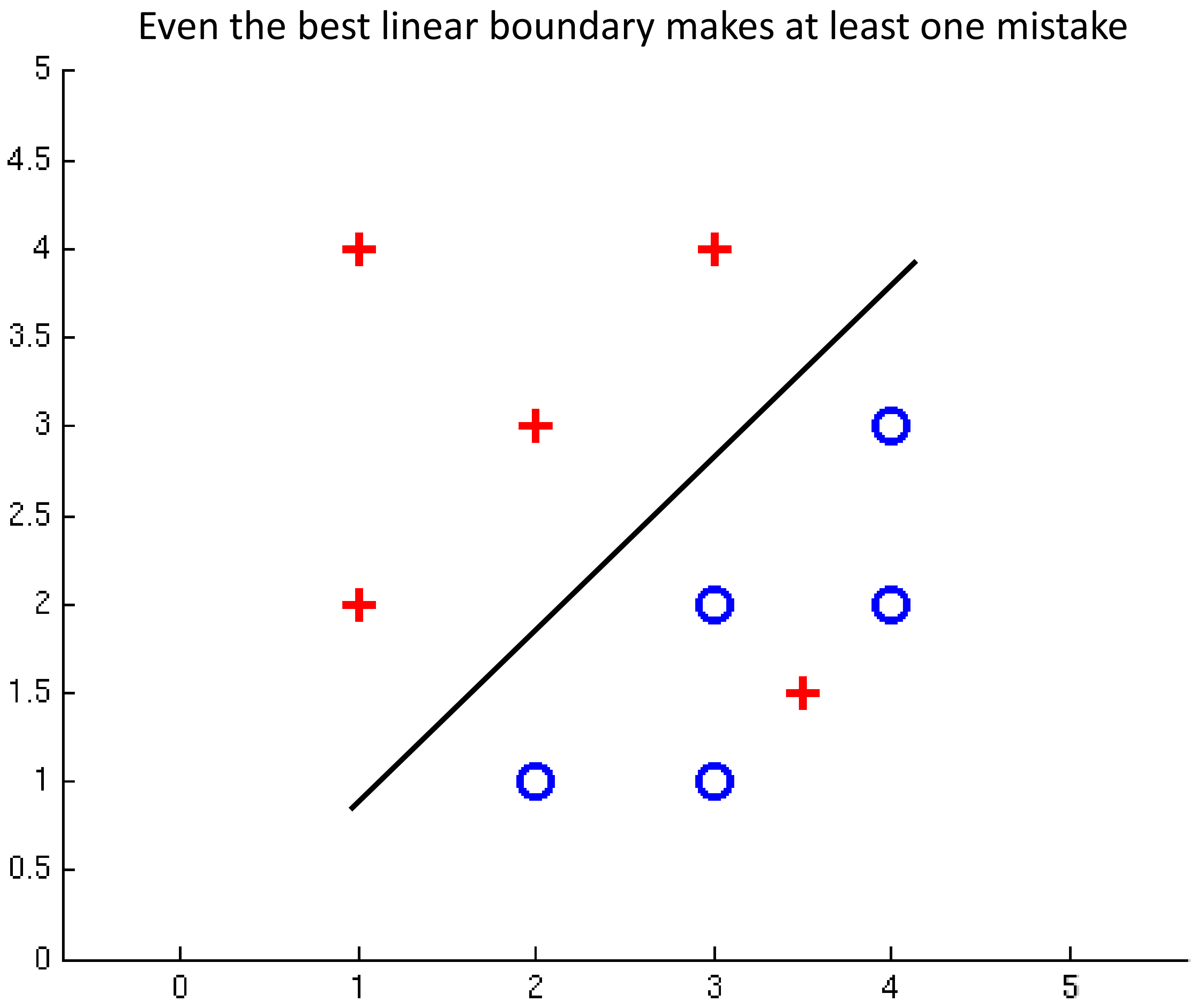




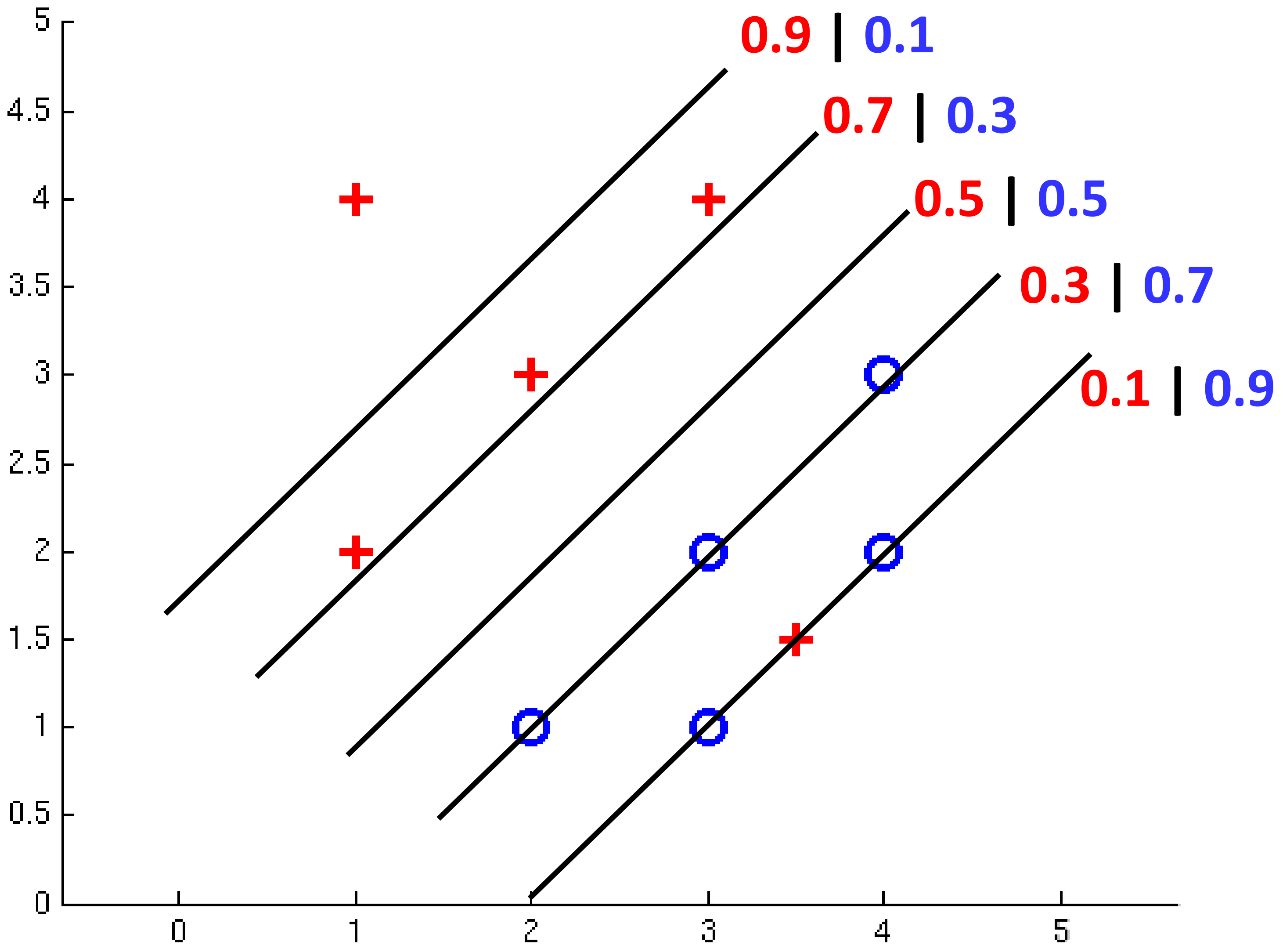
改进感知机模型



不可分割情况下: 肯定性的判别结果



不可分割情况下：概率化的判别决策

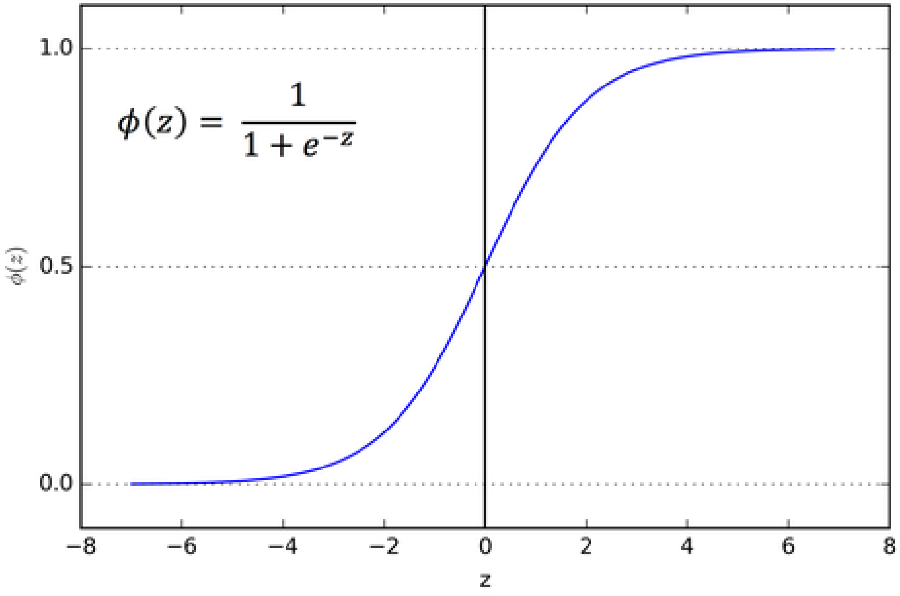


How to get probabilistic decisions?

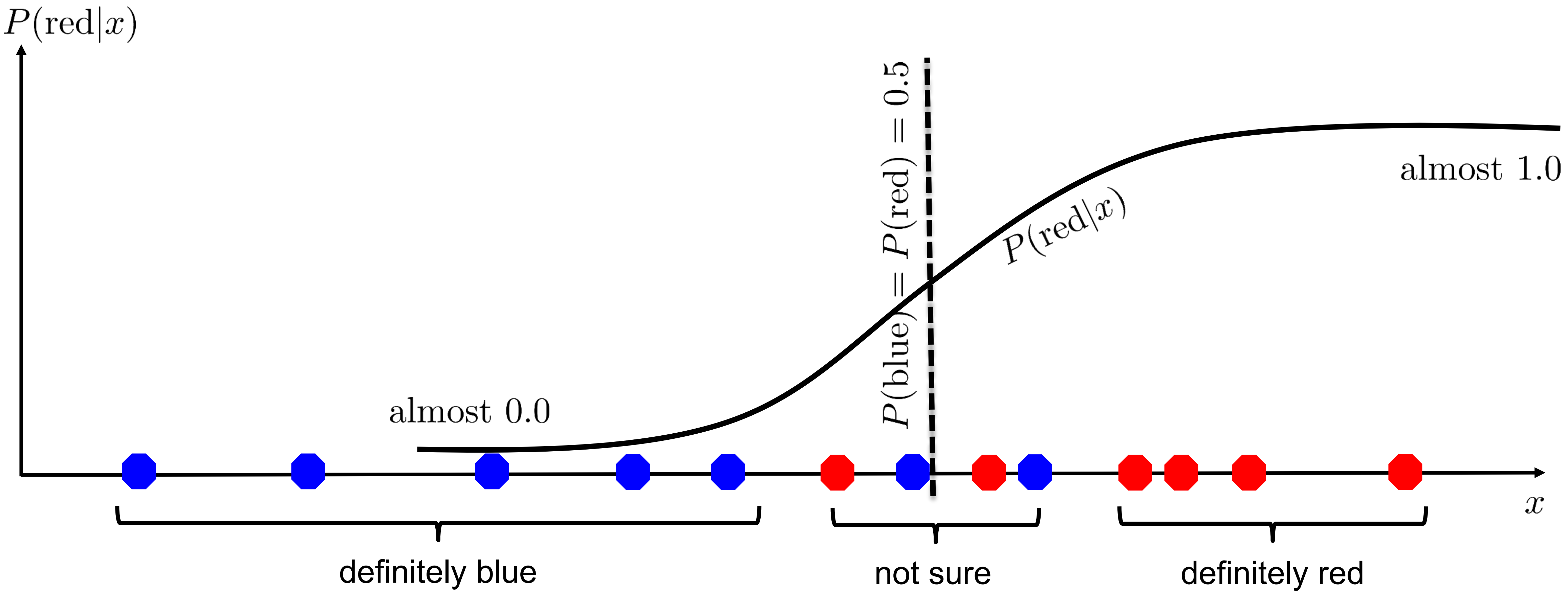
- Perceptron scoring: $z = w \cdot f(x)$
- If $z = w \cdot f(x)$ very positive → want probability going to 1
- If $z = w \cdot f(x)$ very negative → want probability going to 0

- Sigmoid function

$$\phi(z) = \frac{1}{1 + e^{-z}}$$



A 1D Example

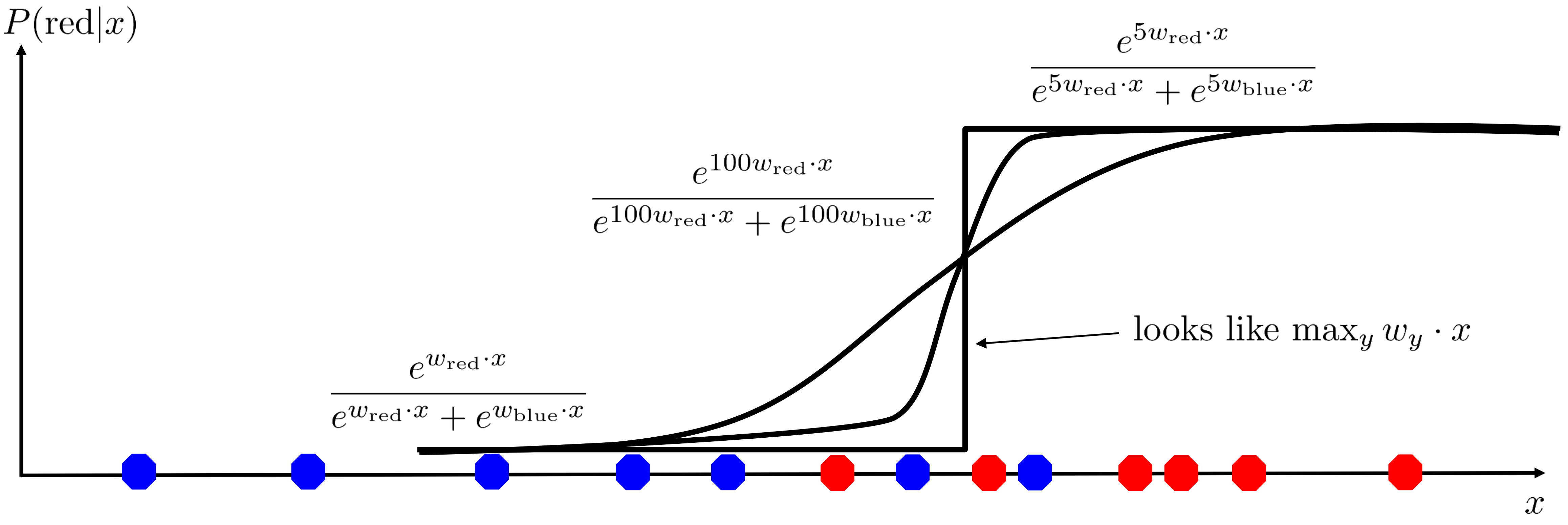


$$P(\text{red}|x) = \frac{e^{w_{\text{red}} \cdot x}}{e^{w_{\text{red}} \cdot x} + e^{w_{\text{blue}} \cdot x}}$$

probability increases exponentially as we move away from boundary

normalizer

The *Soft* Max



$$P(\text{red}|x) = \frac{e^{w_{\text{red}}\cdot x}}{e^{w_{\text{red}}\cdot x}+e^{w_{\text{blue}}\cdot x}}$$

Best w?

- Maximum likelihood estimation 最大似然估计:

$$\max_w \quad ll(w) = \max_w \quad \sum_i \log P(y^{(i)}|x^{(i)}; w)$$

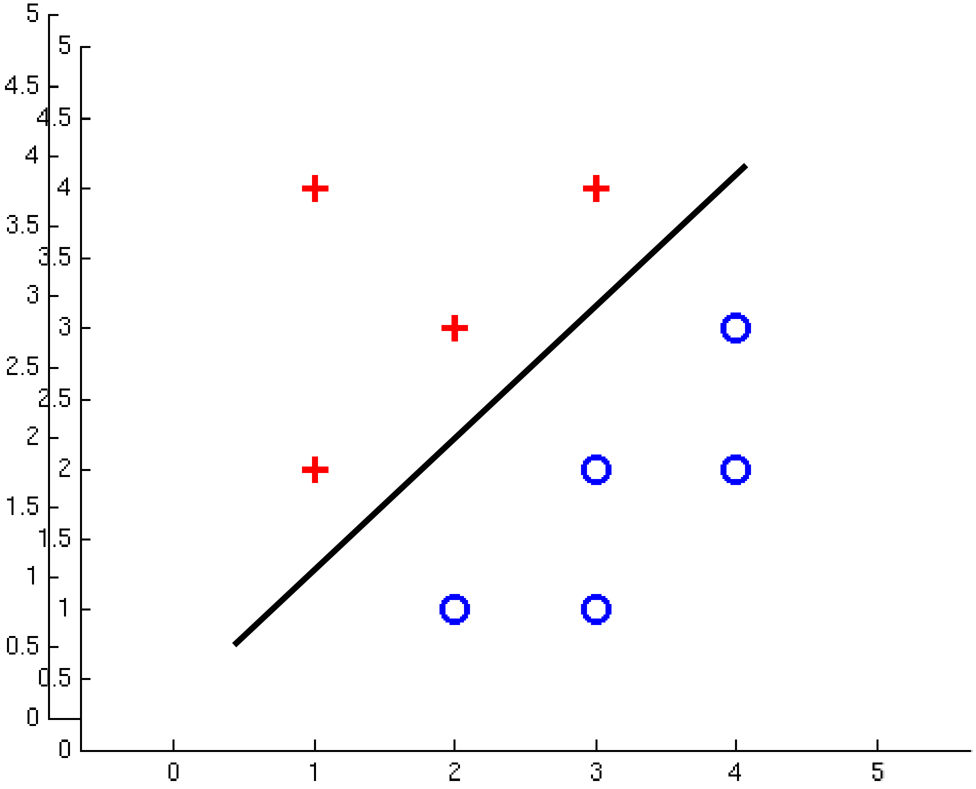
with:

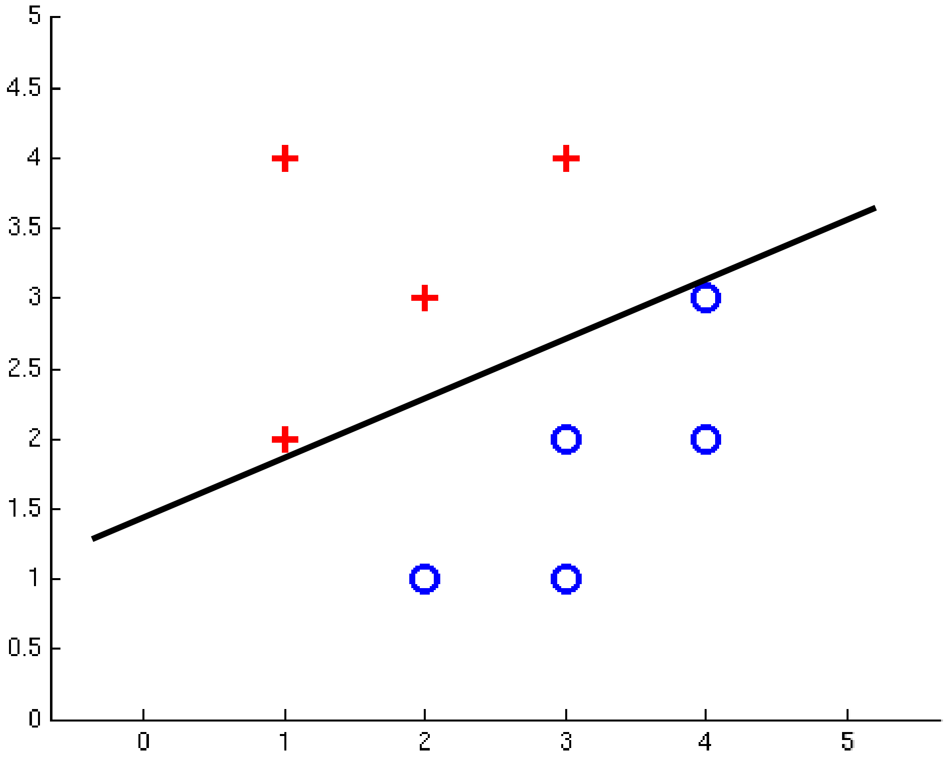
$$P(y^{(i)} = +1|x^{(i)}; w) = \frac{1}{1 + e^{-w \cdot f(x^{(i)})}}$$

$$P(y^{(i)} = -1|x^{(i)}; w) = 1 - \frac{1}{1 + e^{-w \cdot f(x^{(i)})}}$$

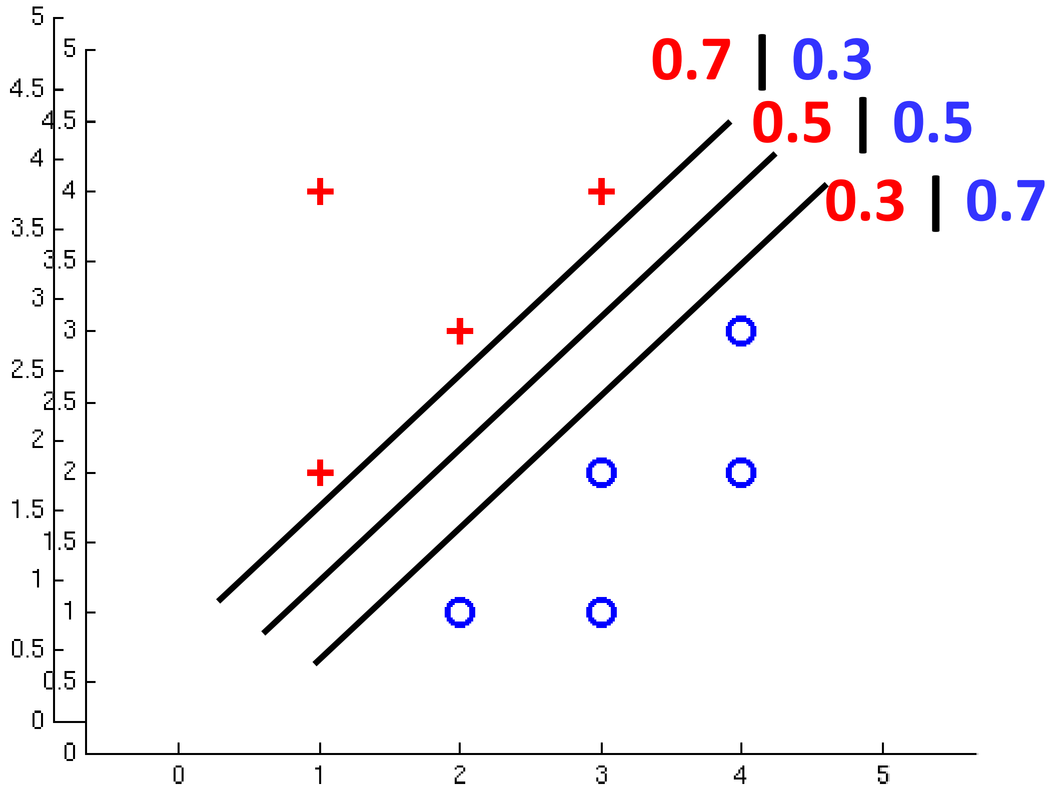
= Logistic Regression 罗吉斯特回归

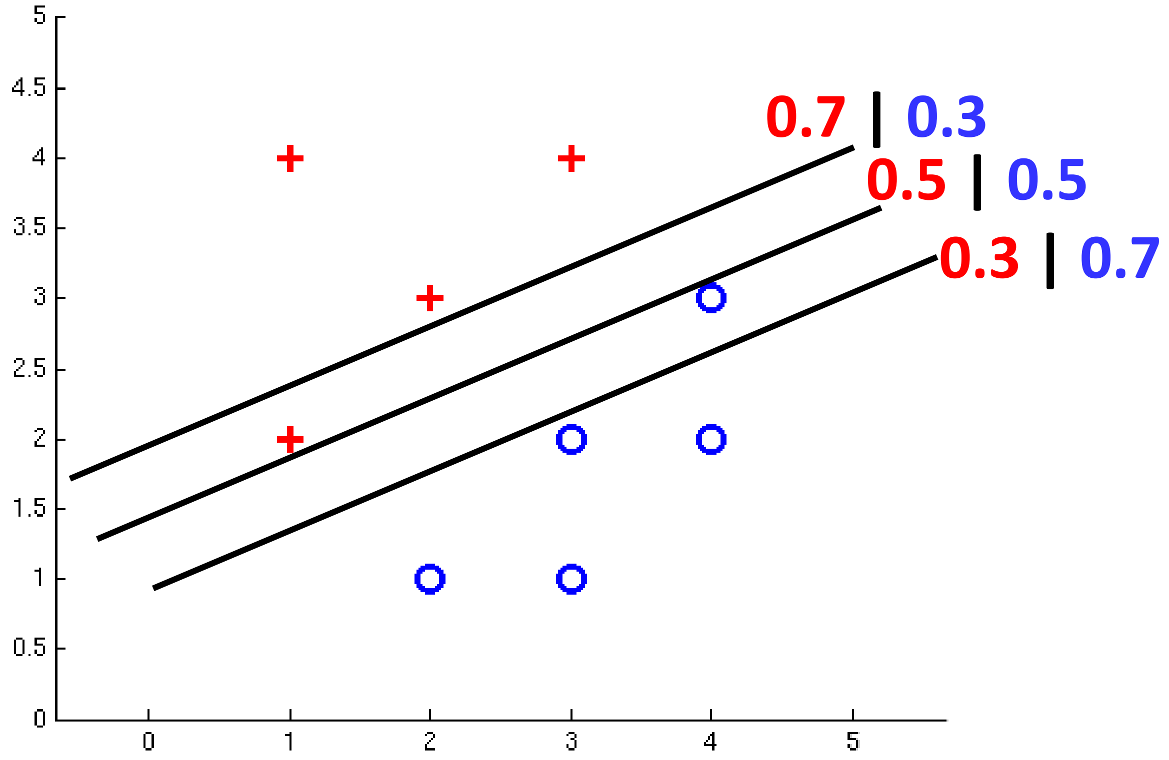
可分割情况: Deterministic Decision – Many Options





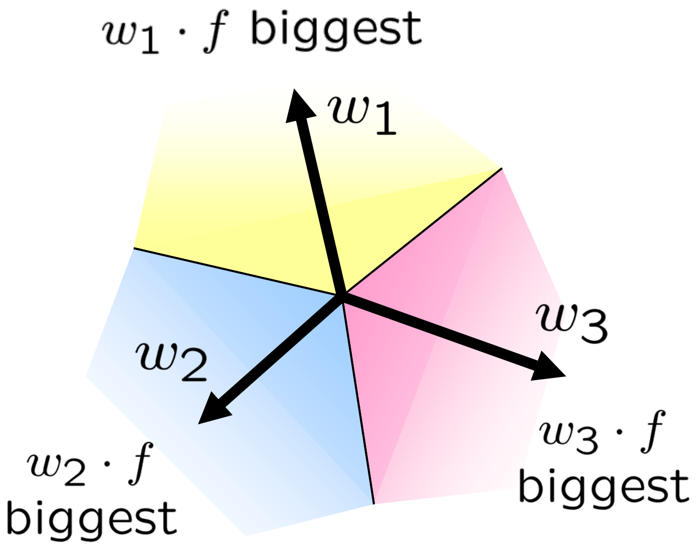
可分割情况: Probabilistic Decision – Clear Preference





多分类下的罗吉斯特回归

- Recall Perceptron:
  - A weight vector for each class: w_y
  - Score (activation) of a class y: $w_y \cdot f(x)$
  - Prediction highest score wins $y = \arg\max_y \; w_y \cdot f(x)$



- How to make the scores into probabilities?

$$\underbrace{z_1, z_2, z_3}_{\text{original activations}} \rightarrow \underbrace{\frac{e^{z_1}}{e^{z_1}+e^{z_2}+e^{z_3}}, \frac{e^{z_2}}{e^{z_1}+e^{z_2}+e^{z_3}}, \frac{e^{z_3}}{e^{z_1}+e^{z_2}+e^{z_3}}}_{\text{softmax activations}}$$

求解最优的 w?

- Maximum likelihood estimation最大似然估计:

$$\max_w \quad ll(w) = \max_w \quad \sum_i \log P(y^{(i)}|x^{(i)}; w)$$

with: $$P(y^{(i)}|x^{(i)}; w) = \frac{e^{w_{y^{(i)}} \cdot f(x^{(i)})}}{\sum_y e^{w_y \cdot f(x^{(i)})}}$$

= Multi-Class Logistic Regression 多分类情况下的罗吉斯特回归

下次课，关于如何求解这个优化问题

- Optimization
  - i.e., how do we solve:

$$\max_w \quad ll(w) = \max_w \quad \sum_i \log P(y^{(i)}|x^{(i)}; w)$$