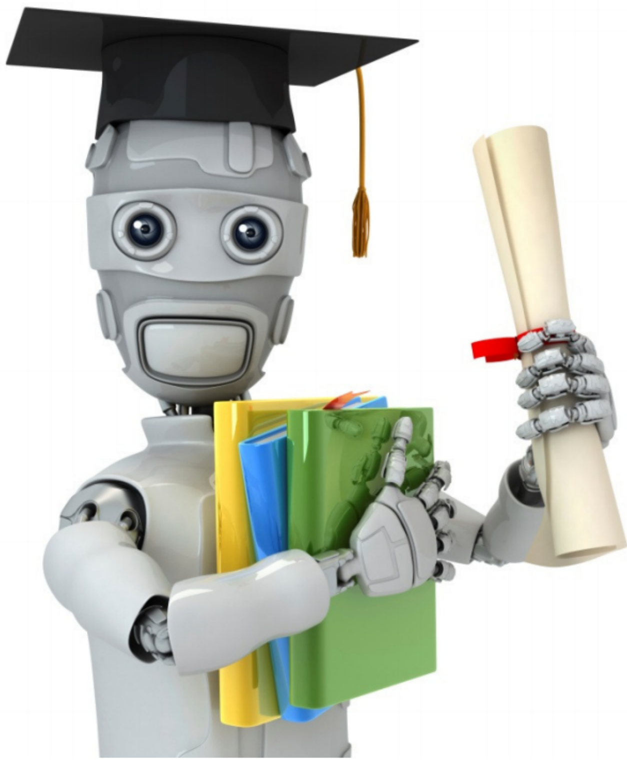
Introduction to Common Classifiers

Yinbin Ma 2017.4.21



Machine Learning

MSC in XDU

Outline

- Review Last Lecture
- Common Classifiers Algorithm
  - kNN - k Nearest Neighbor
  - Perceptron
  - Linear Support Vector Machine
- QA

MSC in XDU

Review

- Regression?

- Linear Regression?

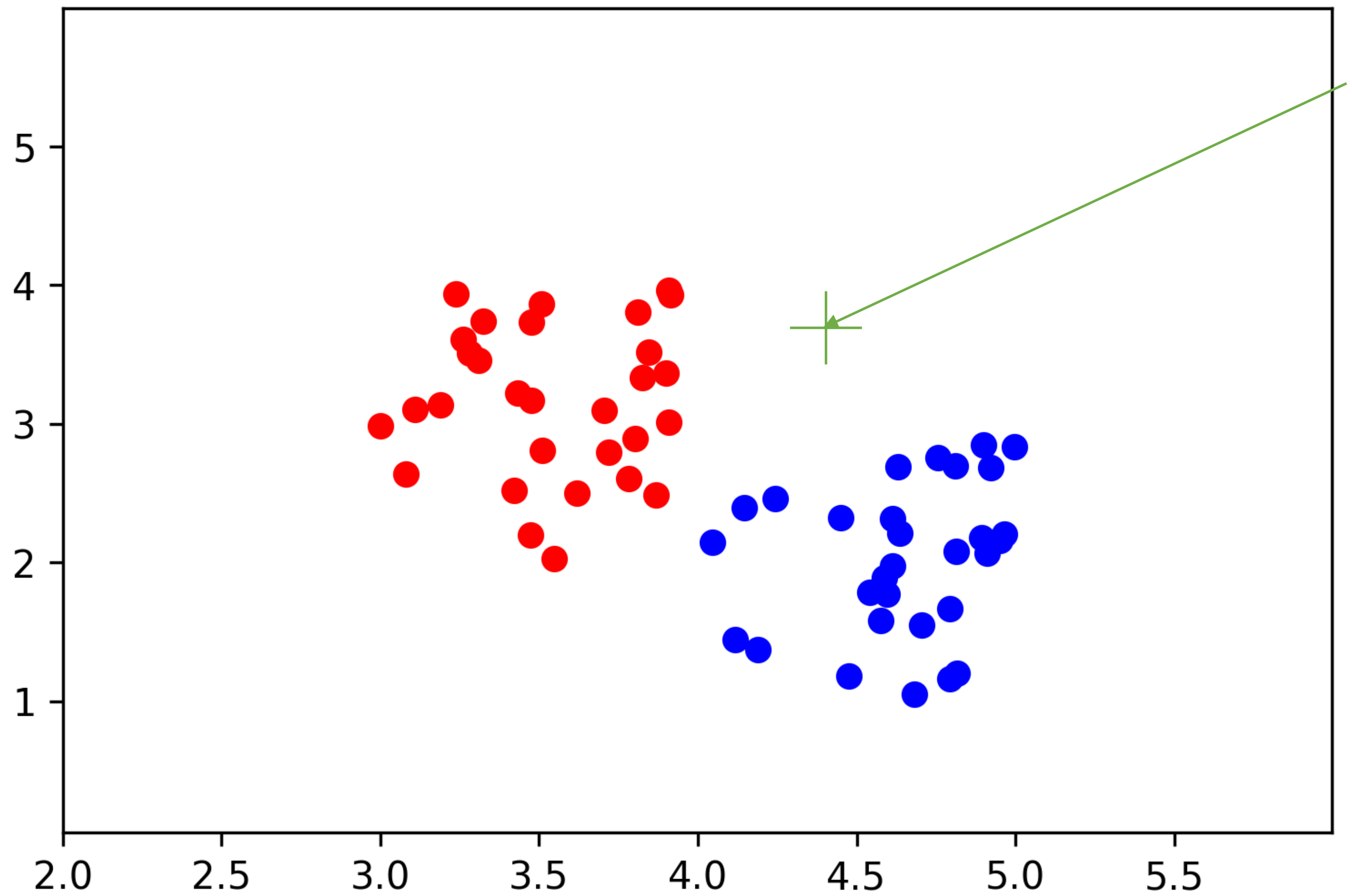
- Loss Function?

Classifiers

Classifiers

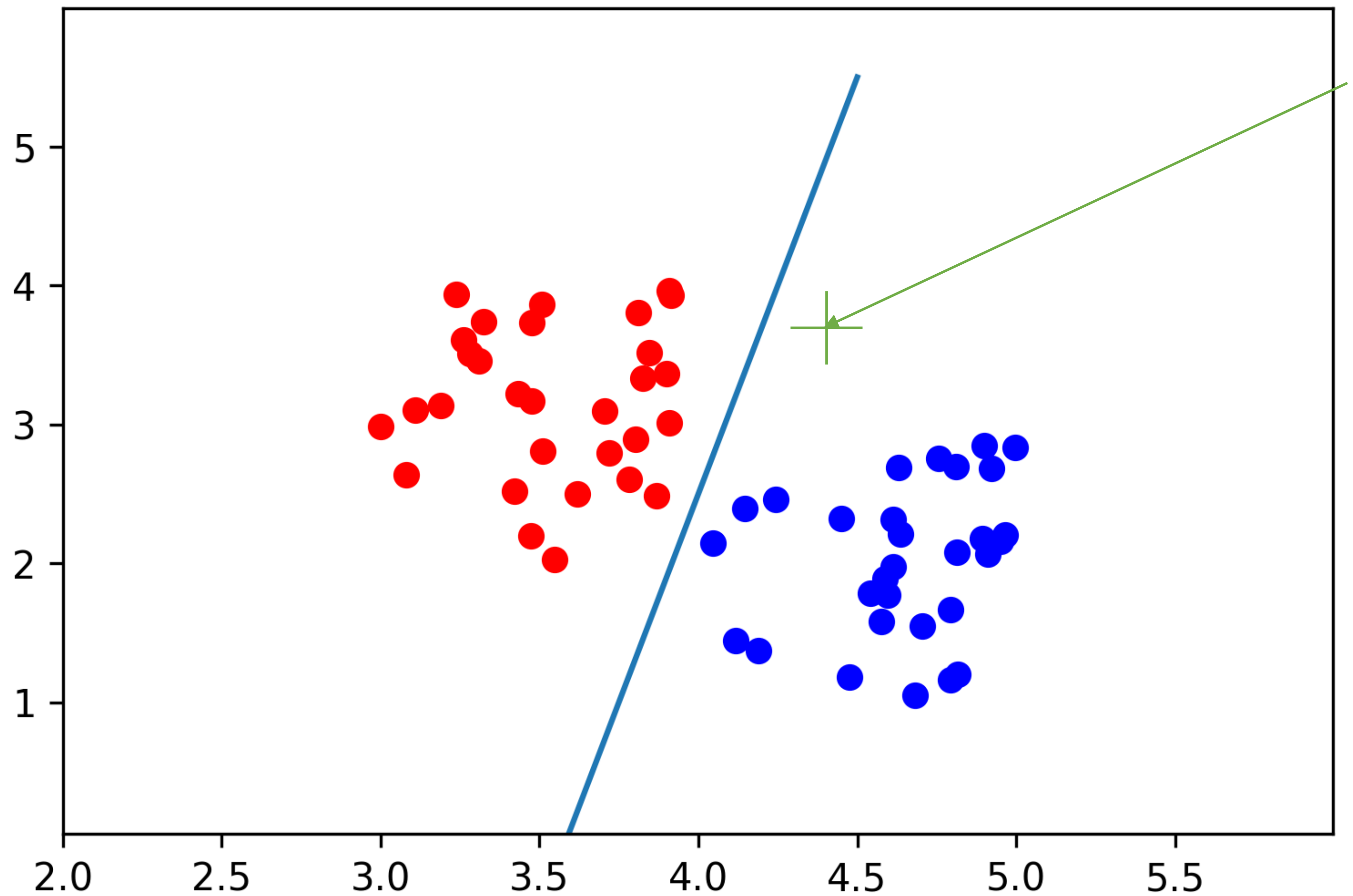
To solve the problem of identifying to which of a set of categories a new observation belongs

Classifiers



Giving X and Y, can you identity which class is this spot belongs?

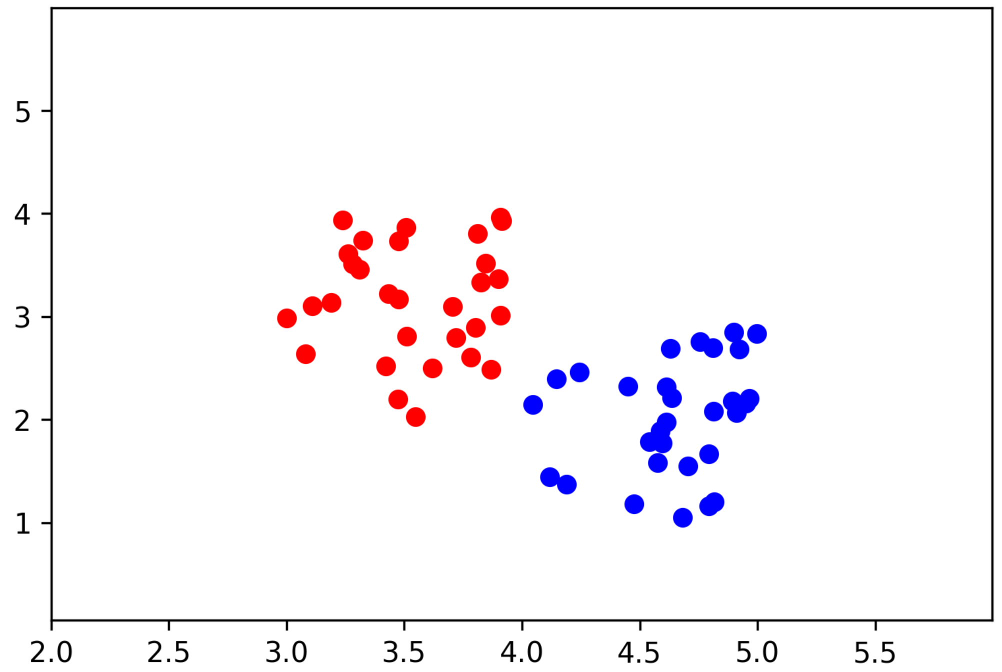
Classifiers



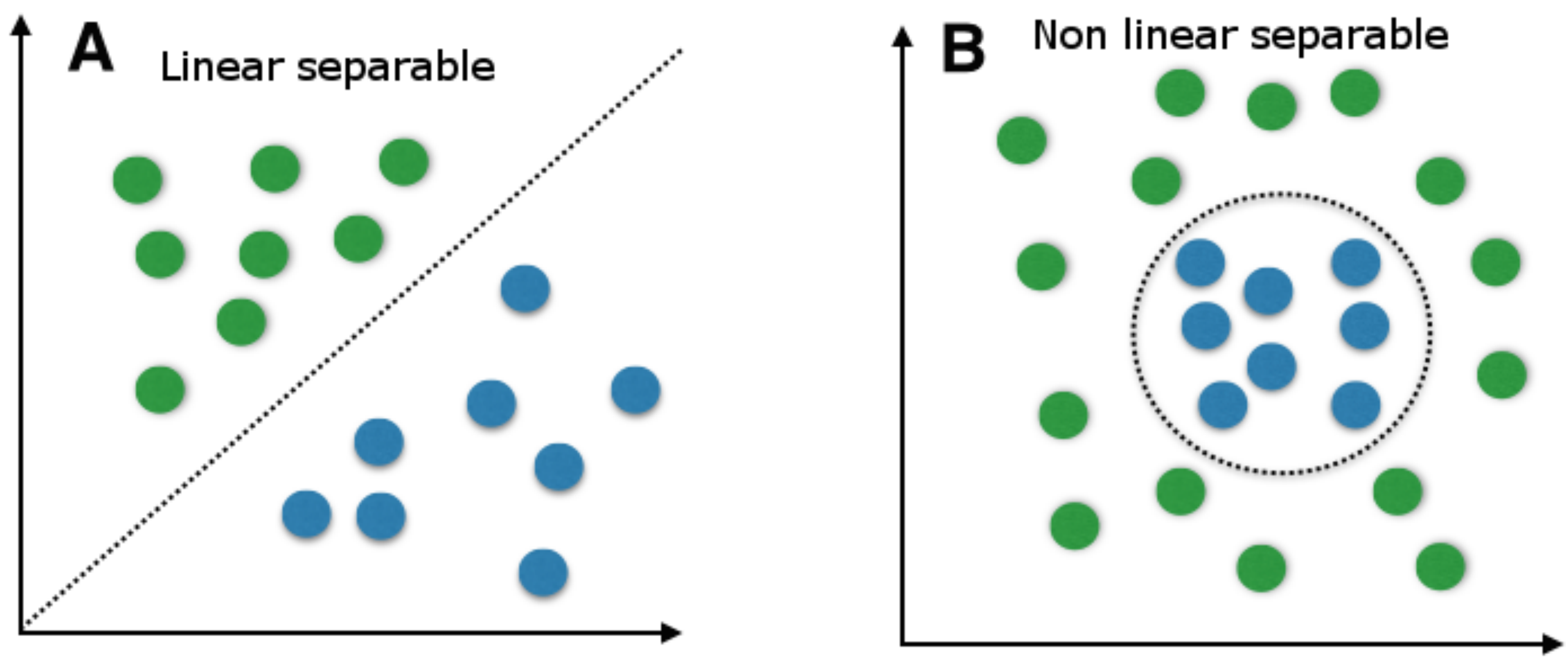
result:
this spot belongs
to blue

Classifiers

- first: find out a pattern
- building: train data
- after: benchmark



Linear VS non-Linear



- Hyperplane: $\sum_{i=1}^{n}(w_i \bullet x_i) = 0$

kNN

- X belongs to the “Nearest Neighbor” class.
- 远亲不如近邻
- Distance: $d(x,y)=\sqrt{x^2-y^2}$
- find out $\min(d(x,y))\longrightarrow$ x belongs y’s class

kNN

$X \leftarrow \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}$ $\quad Y \leftarrow \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$ $\quad X_1 = [1 \quad 2] \quad X_2 = [3 \quad 4] \quad X_3 = [5 \quad 6]$

X_1, X_3 belongs to 1 $\quad X_2$ belongs to 2

input: $X\_ \leftarrow \begin{bmatrix} 0 & 2 \\ 4 & 1 \end{bmatrix}$

output: $Y\_ \leftarrow \begin{bmatrix} ? \\ ? \end{bmatrix}$ the label of X_

kNN

- kNN: What happens when k = 2? k = 3?…

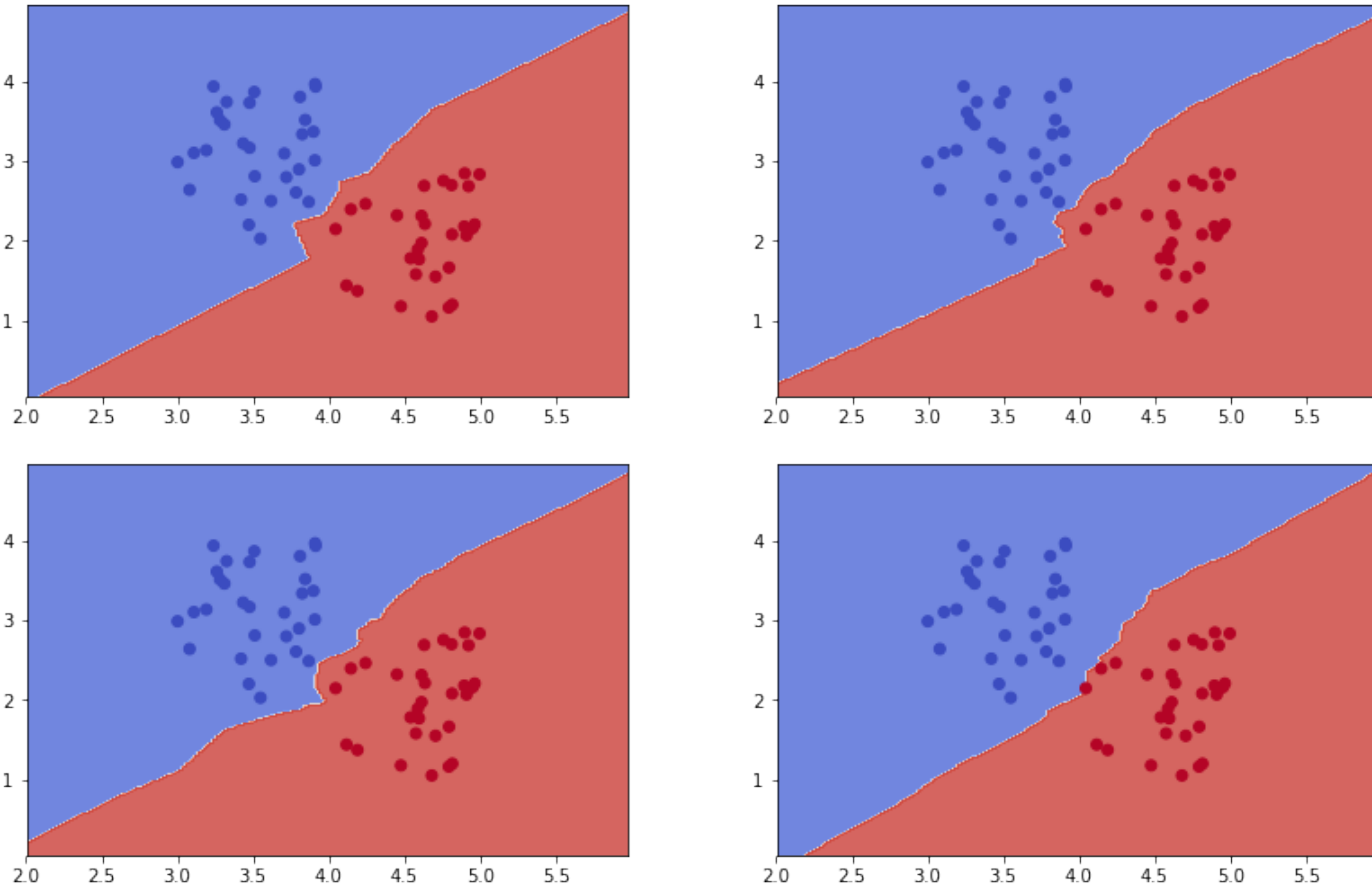
- Supposing that X is a m*c matrix, X_ is a n*c matrix, how many calculating operations can we get the Y_ ?

- In general, does kNN has training section? how could we fast it up?

- Can we redefine the “distance” concept?

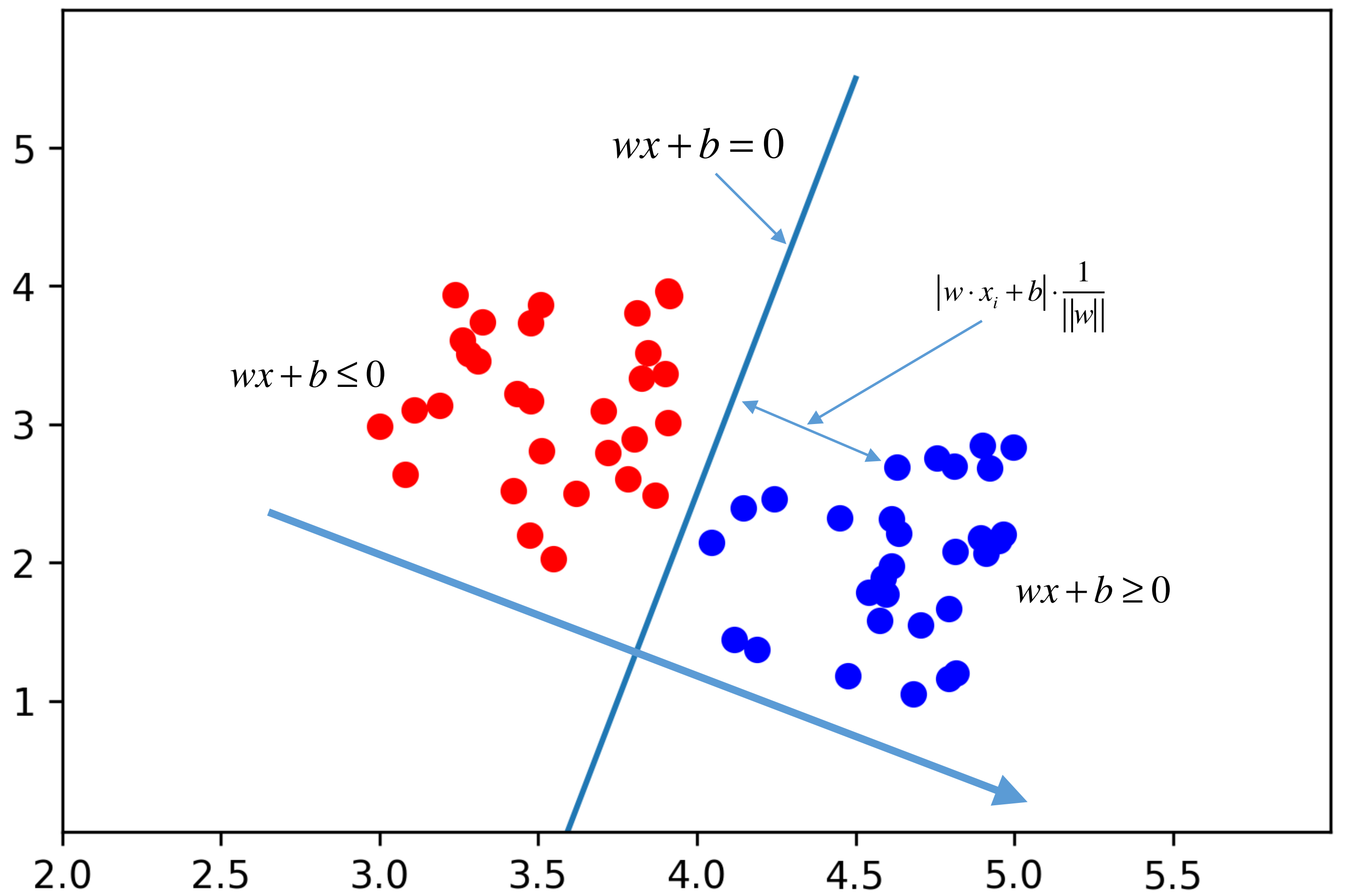
MSC in XDU

kNN



break time

Perceptron



$wx+b=0$
$|w \cdot x_i + b| \cdot \frac{1}{||w||}$
$wx+b \le 0$
$wx+b \ge 0$
5
4
3
2
1
2.0
2.5
3.0
3.5
4.0
4.5
5.0
5.5

Perceptron

- mapping 2D's data to 1D -> w is a 2*1 matrix
- find a threshold (normally use $wx + b = 0$)
- seems like we need a loss function

Perceptron

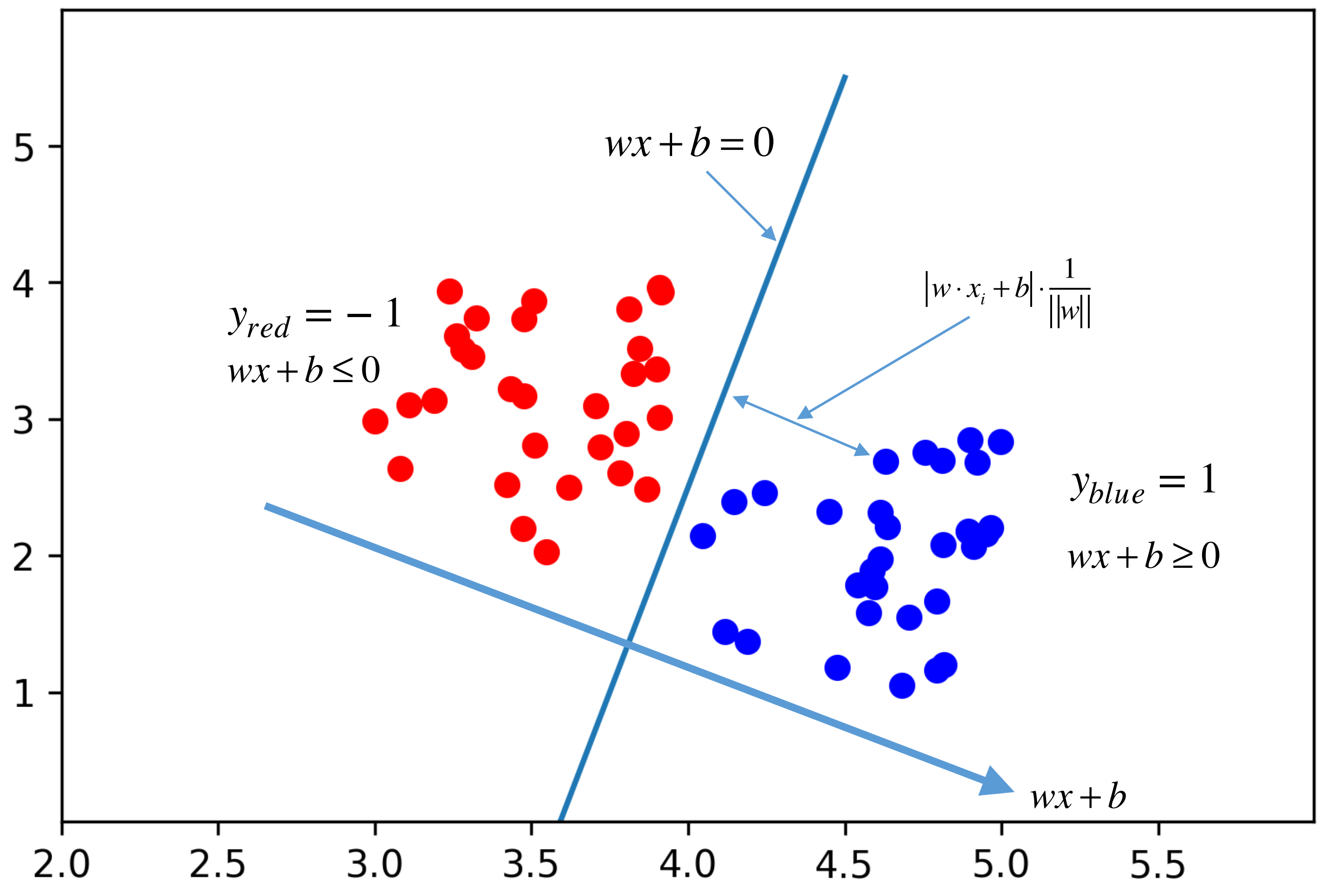
- Loss function:
    - score w and b if they fit data
    - better be continuity

- accuracy -> fault number

- less faults -> lower loss

MSC in XDU

Perceptron



$wx+b=0$
$|w \cdot x_i + b| \cdot \frac{1}{||w||}$
$y_{red} = -1$
$wx+b \leq 0$
$y_{blue} = 1$
$wx+b \geq 0$
$wx+b$
5
4
3
2
1
2.0
2.5
3.0
3.5
4.0
4.5
5.0
5.5

Perceptron

Supposing that M is fault set,$X_i \in M$

$$\text{Loss}(w,b) = -\cancel{\frac{1}{\|w\|}} \sum_{x_i \in M} y_i(w \cdot x_i + b) \quad \text{sum of margin}$$

$$\text{Loss}(w,b) = -\sum_{x_i \in M} y_i(w \cdot x_i + b)$$

Our goal is: minimize loss function -> use derivative (导数)

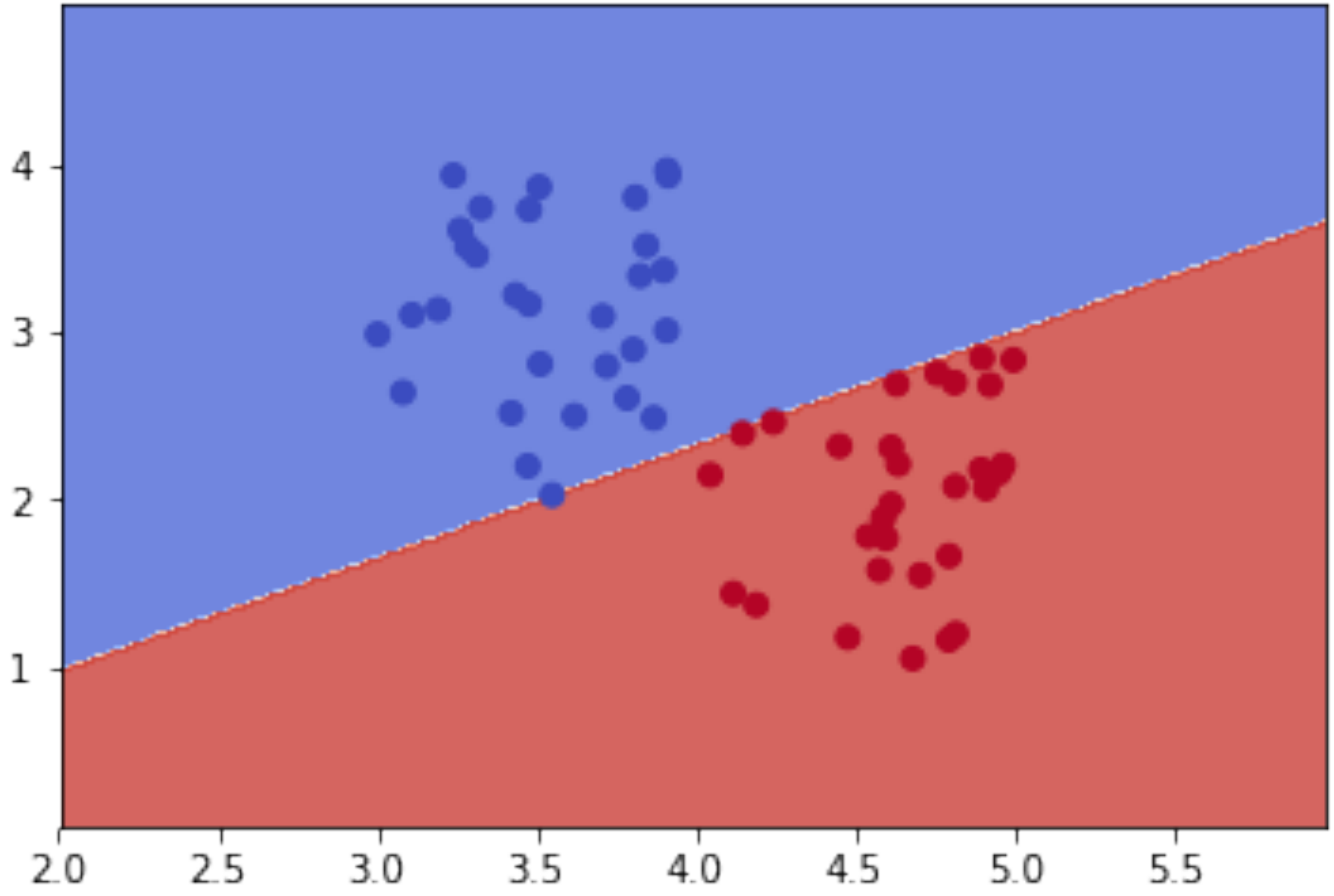
$$\nabla_w \text{Loss}(w,b) = -\sum_{x_i \in M} y_i \cdot x_i \qquad \nabla_b \text{Loss}(w,b) = -\sum_{x_i \in M} y_i$$

MSC in XDU

Perceptron

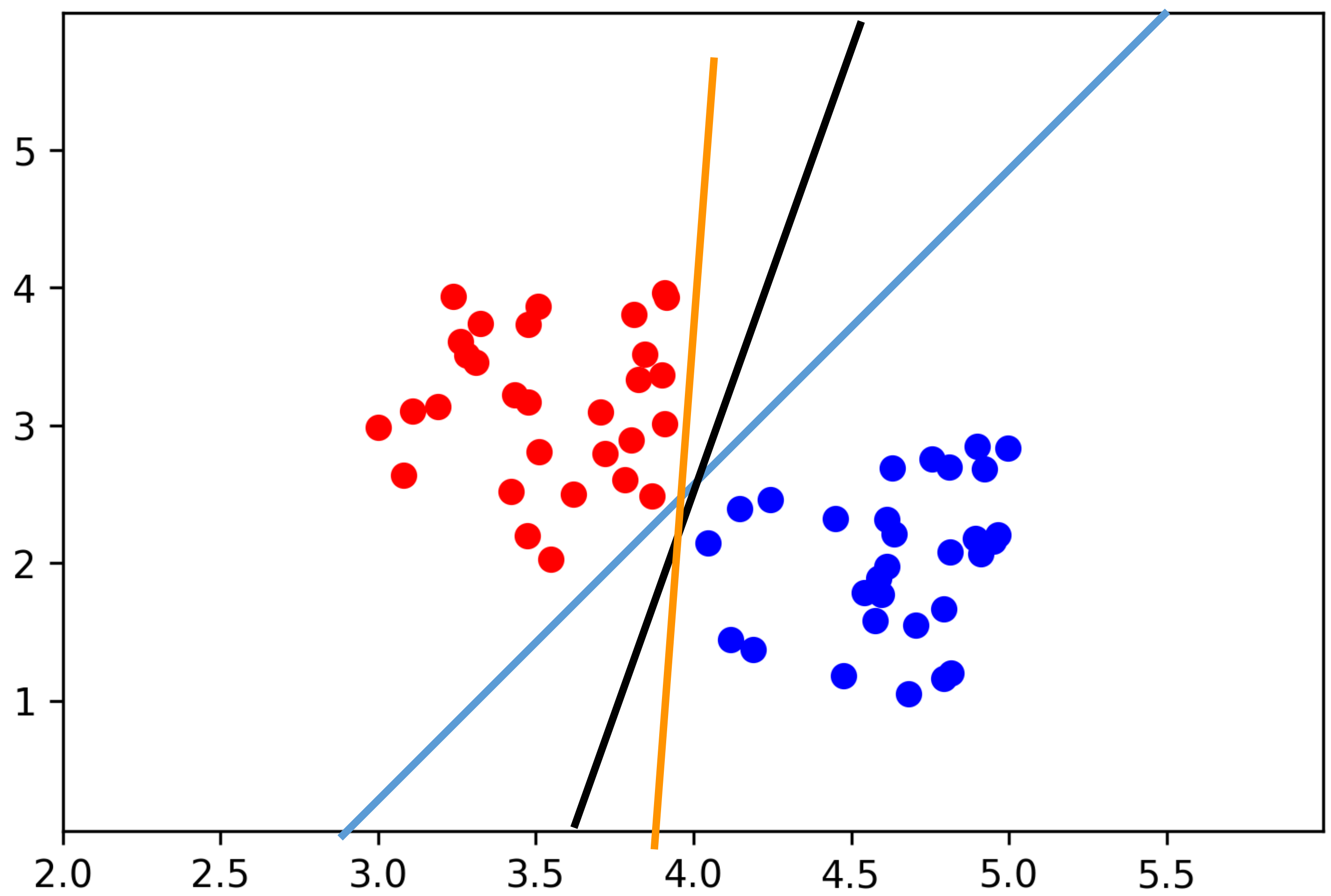
- Loss(w,b) = 0?
- Using once then throw out?
- If Loss(w,b) = 0, what does the wx+b = 0 looks like?

Perceptron



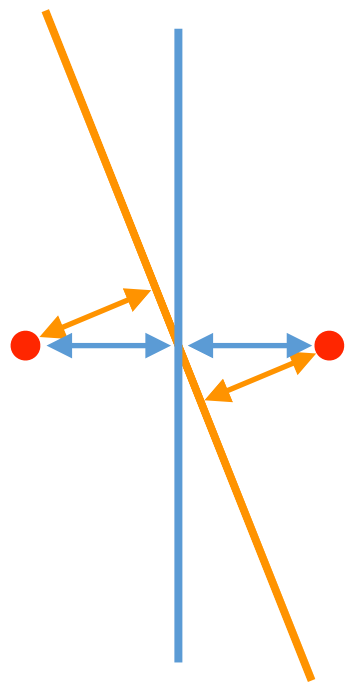
Linear SVM

Which hyperplane you will choose?



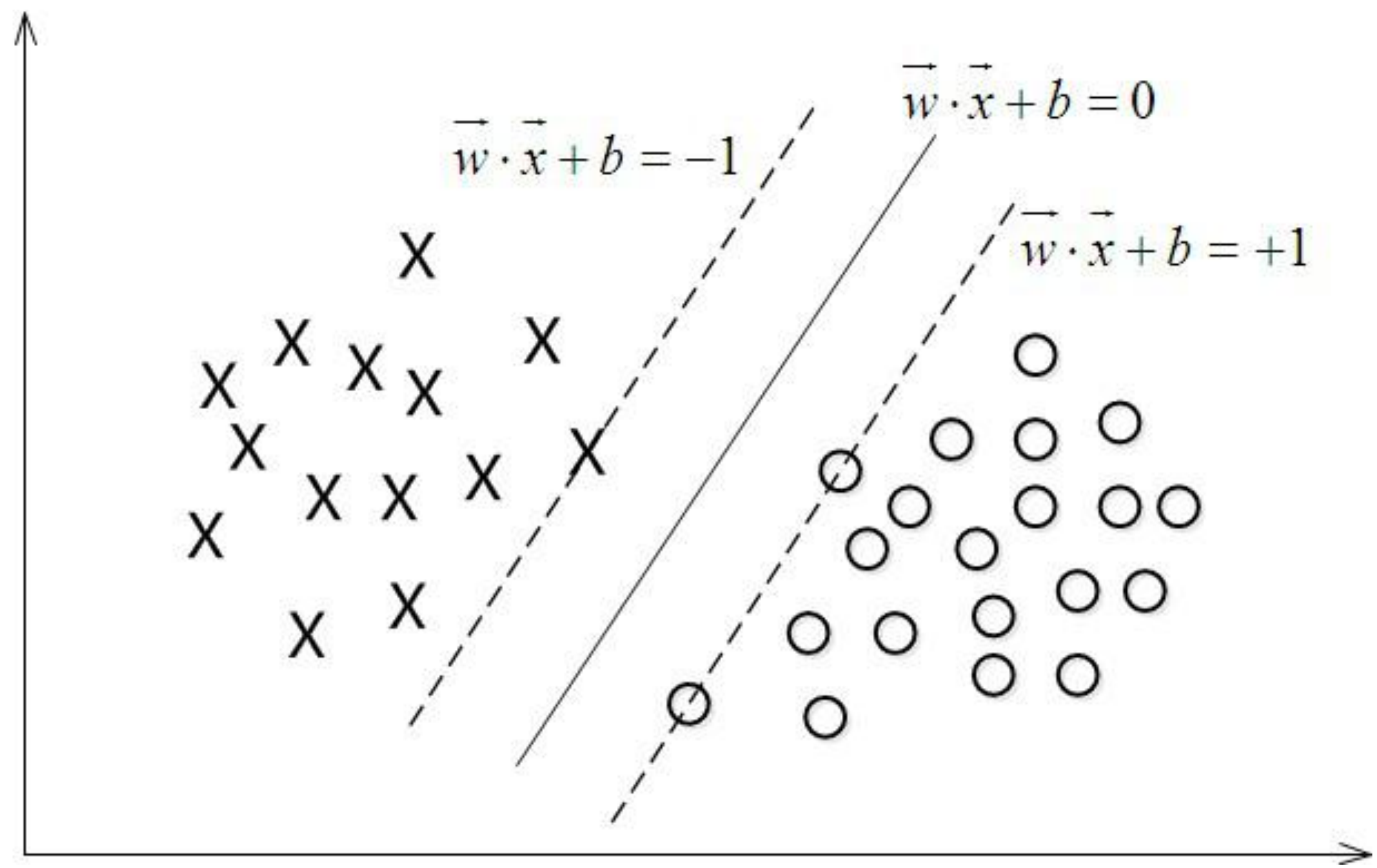
Linear SVM

- SVM: Support Vector Machine
- Maximize sum of the margin
- work like perceptron, but more stability.

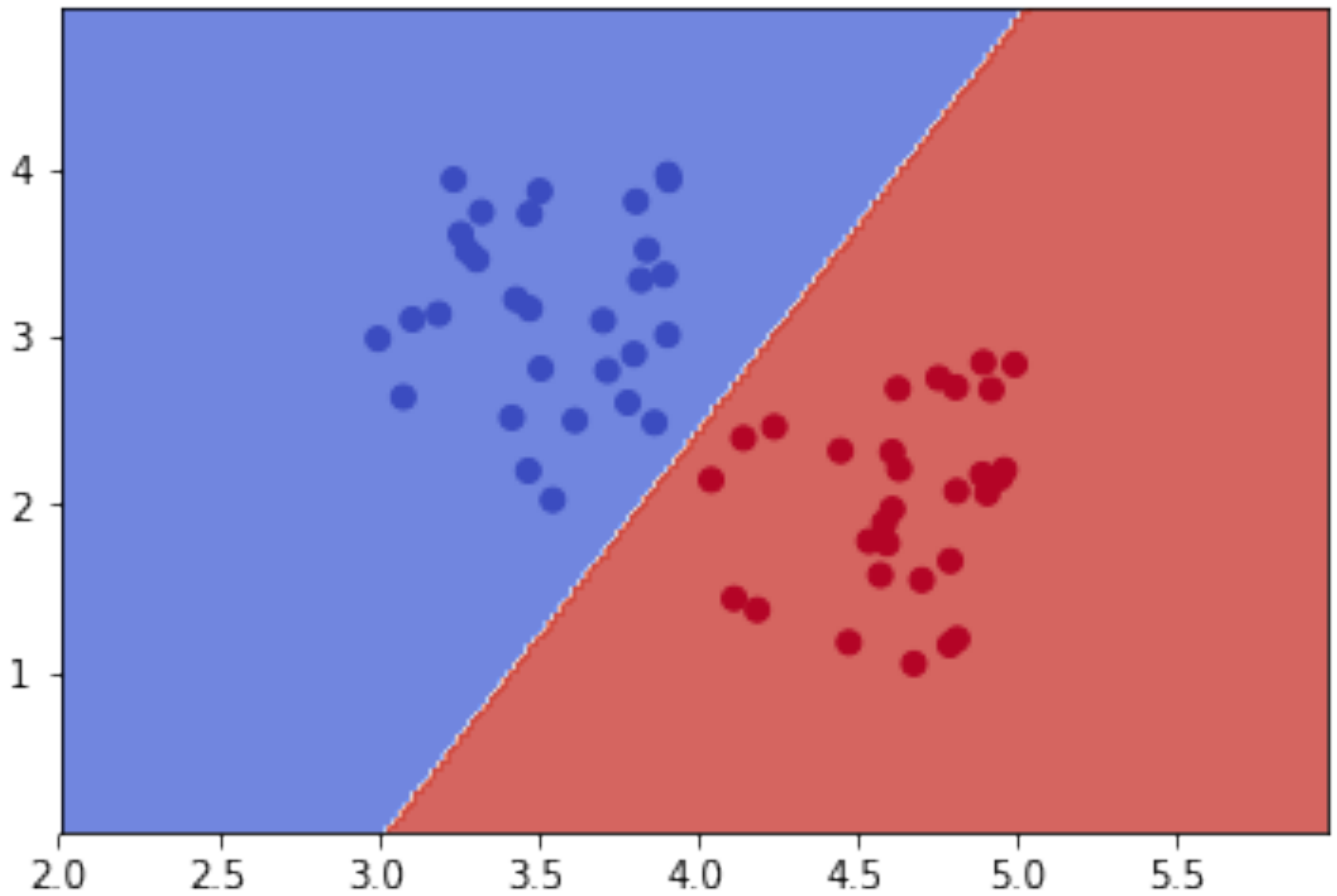


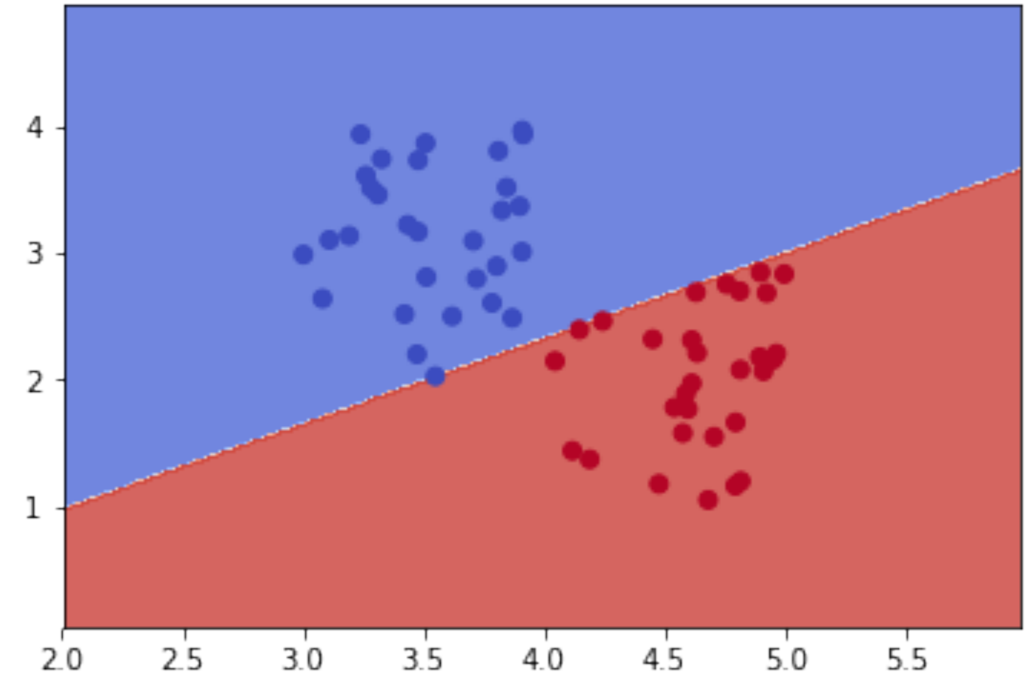
MSC in XDU

Linear SVM



Linear SVM





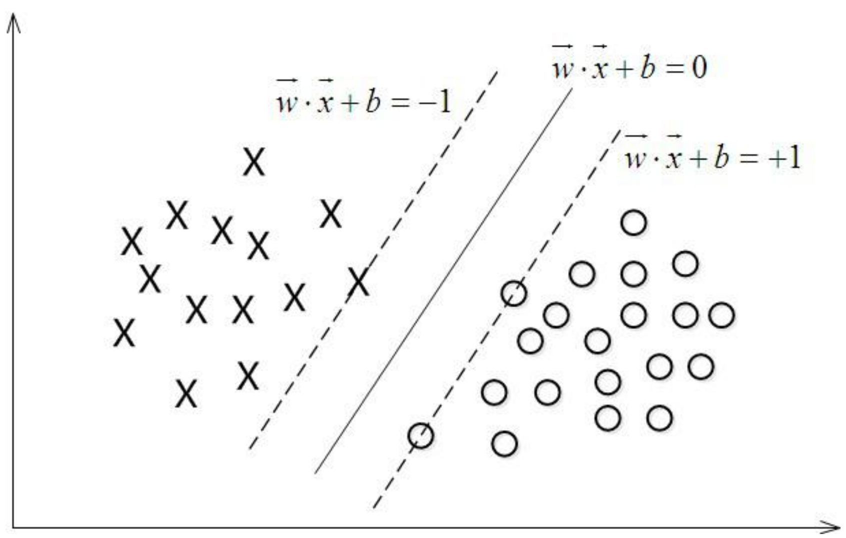
MSC in XDU

Linear SVM

$$\text{margin}(x_i) = \frac{1}{\|w\|} |wx + b|$$

$$\max \frac{1}{\|w\|} \quad s.t.,\ y_i(w \cdot x_i + b) \geq 1$$

$$\min \frac{1}{2}\|w\|^2 \quad s.t.,\ y_i(w \cdot x_i + b) \geq 1$$



$$\mathcal{L}(\mathrm{w},\mathrm{b},\alpha)=\frac{1}{2}\|w\|^2-\sum_{i=1}^{m}\alpha_i\left[y^{(i)}\left(w^Tx^{(i)}+b\right)-1\right]$$

$$=\frac{1}{2}\mathrm{w}^T\mathrm{w}-\sum_{i=1}^{m}\alpha_i y^{(i)}\mathrm{w}^T x^{(i)}-\sum_{i=1}^{m}\alpha_i y^{(i)} b+\sum_{i=1}^{m}\alpha_i$$

$$=\frac{1}{2}\mathrm{w}^T\sum_{i=1}^{m}\alpha_i y^{(i)}x^{(i)}-\sum_{i=1}^{m}\alpha_i y^{(i)}\mathrm{w}^T x^{(i)}-\sum_{i=1}^{m}\alpha_i y^{(i)} b+\sum_{i=1}^{m}\alpha_i$$

$$=\frac{1}{2}\mathrm{w}^T\sum_{i=1}^{m}\alpha_i y^{(i)}x^{(i)}-w^T\sum_{i=1}^{m}\alpha_i y^{(i)}x^{(i)}-\sum_{i=1}^{m}\alpha_i y^{(i)} b+\sum_{i=1}^{m}\alpha_i$$

$$=-\frac{1}{2}\mathrm{w}^T\sum_{i=1}^{m}\alpha_i y^{(i)}x^{(i)}-\sum_{i=1}^{m}\alpha_i y^{(i)} b+\sum_{i=1}^{m}\alpha_i$$

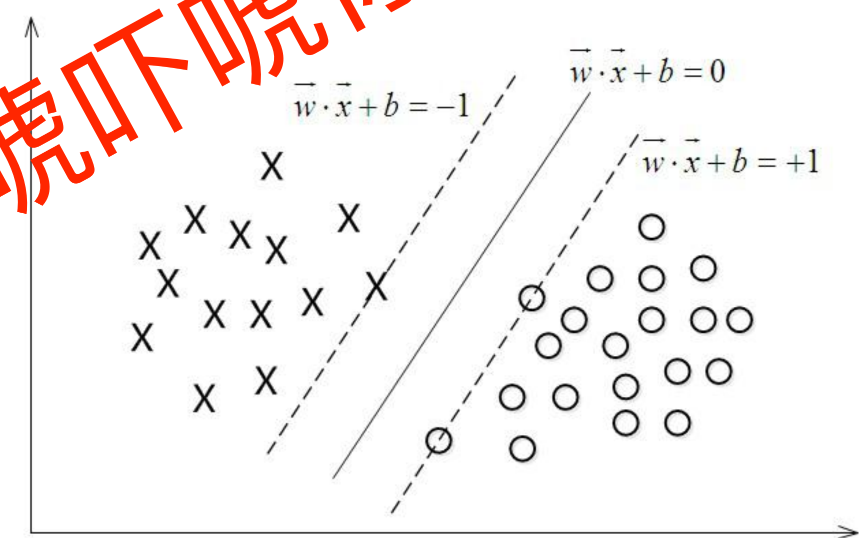
$$=-\frac{1}{2}\mathrm{w}^T\sum_{i=1}^{m}\alpha_i y^{(i)}x^{(i)}-b\sum_{i=1}^{m}\alpha_i y^{(i)}+\sum_{i=1}^{m}\alpha_i$$

$$=-\frac{1}{2}\left(\sum_{i=1}^{m}\alpha_i y^{(i)}x^{(i)}\right)^T\sum_{i=1}^{m}\alpha_i y^{(i)}x^{(i)}-b\sum_{i=1}^{m}\alpha_i y^{(i)}+\sum_{i=1}^{m}\alpha_i$$

$$=-\frac{1}{2}\sum_{i=1}^{m}\alpha_i y^{(i)}\left(x^{(i)}\right)^T\sum_{i=1}^{m}\alpha_i y^{(i)}x^{(i)}-b\sum_{i=1}^{m}\alpha_i y^{(i)}+\sum_{i=1}^{m}\alpha_i$$

$$=-\frac{1}{2}\sum_{i=1,j=1}^{m}\alpha_i y^{(i)}\left(x^{(i)}\right)^T\alpha_j y^{(j)}x^{(j)}-b\sum_{i=1}^{m}\alpha_i y^{(i)}+\sum_{i=1}^{m}\alpha_i$$

这一页只是用来吓唬吓唬你的

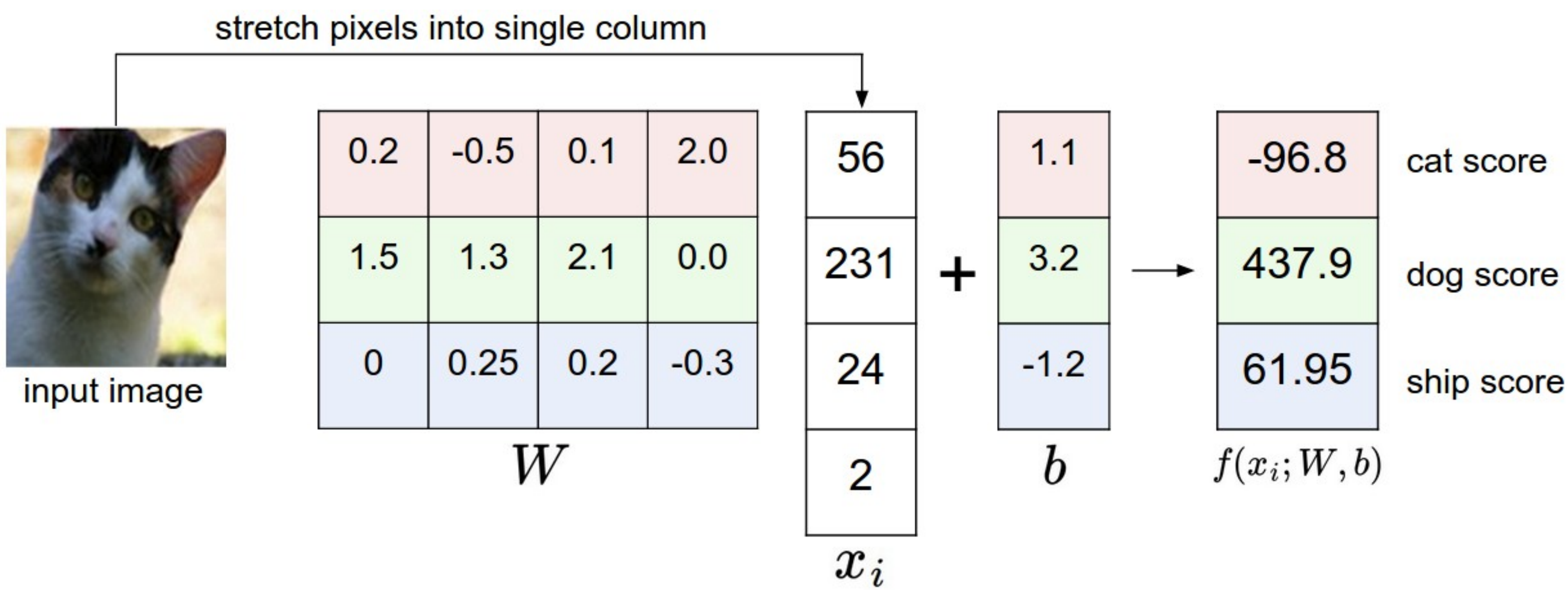


Linear SVM

- Complex loss function -> awful processing
- Why not change a simple loss function?
- Methods isn’t important but THE PURPOSE

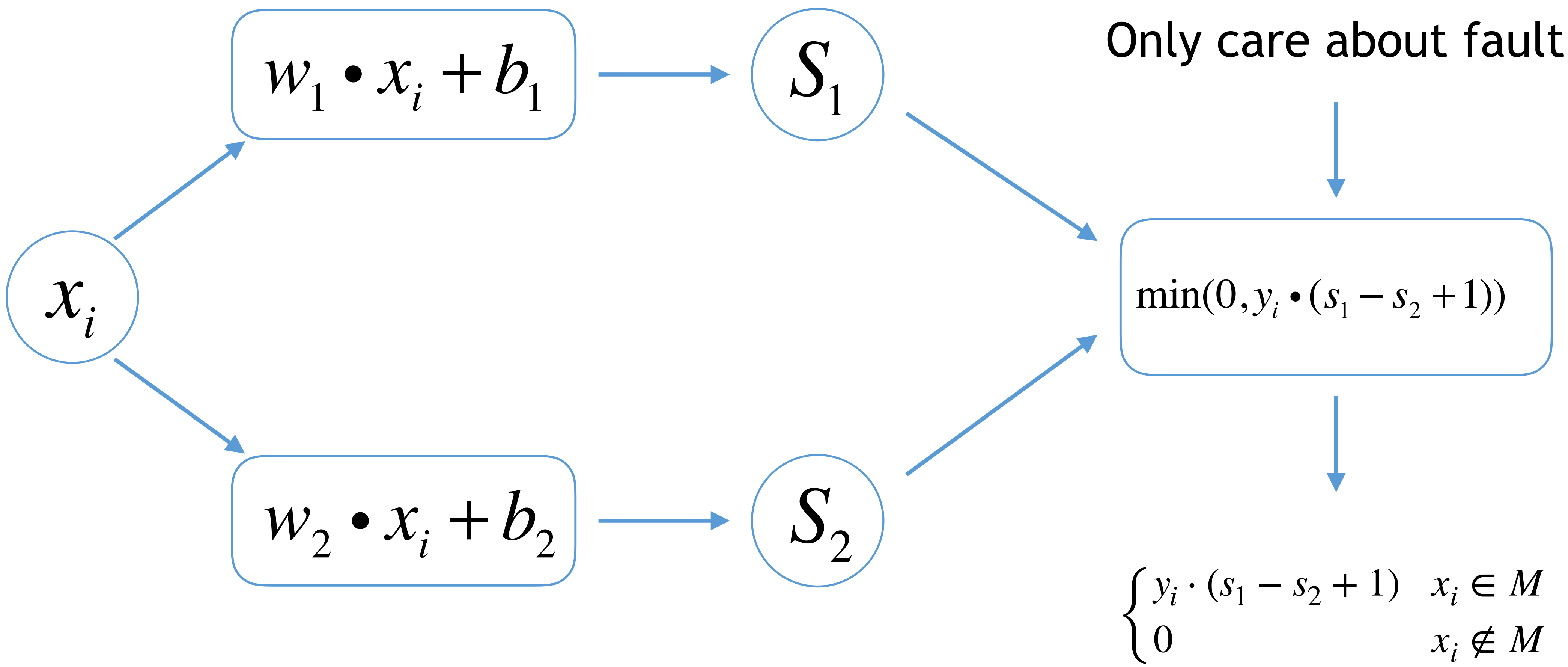
MSC
in XDU

Linear SVM



stretch pixels into single column
input image
0.2
-0.5
0.1
2.0
1.5
1.3
2.1
0.0
0
0.25
0.2
-0.3
W
56
231
24
2
x_i
+
1.1
3.2
-1.2
b
-96.8
437.9
61.95
$f(x_i; W, b)$
cat score
dog score
ship score

Linear SVM



x_i
$w_1 \bullet x_i + b_1$
$w_2 \bullet x_i + b_2$
S_1
S_2
Only care about fault
$\min(0, y_i \bullet (s_1 - s_2 + 1))$
$\begin{cases} y_i \cdot (s_1 - s_2 + 1) & x_i \in M \\ 0 & x_i \notin M \end{cases}$

Linear SVM

- Loss function:

$$L_i = \text{max}(0, -y_i \cdot (s_1 - s_2 + 1)) \quad L = \frac{1}{N}\sum L_i \geq 0 \quad w = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \quad b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

$$L_i = \text{max}(0, -y_i \cdot (w_1 x_i - w_2 x_i + b_1 - b_2 + 1))$$

$$\nabla_{w_1} L_i = \begin{cases} -y_i \cdot x_i & \text{if } -y_i \cdot (s_1 - s_2 + 1) > 0 \\ 0 & \text{else} \end{cases} \qquad \nabla_{w_2} L_i = \begin{cases} y_i \cdot x_i & \text{if } -y_i \cdot (s_1 - s_2 + 1) > 0 \\ 0 & \text{else} \end{cases}$$

$$\nabla_{b_1} L_i = \begin{cases} -1 & \text{if } -y_i \cdot (s_1 - s_2 + 1) > 0 \\ 0 & \text{else} \end{cases} \qquad \nabla_{b_1} L_i = \begin{cases} 1 & \text{if } -y_i \cdot (s_1 - s_2 + 1) > 0 \\ 0 & \text{else} \end{cases}$$

https://zhuanlan.zhihu.com/p/20945670?refer=intelligentunit
https://www.elen.ucl.ac.be/Proceedings/esann/esannpdf/es1999-461.pdf

Summary

- kNN: simple but calculating-complicated
- Perceptron: sounds perfect but needs improve
- Linear SVM: effective but hard to use

QA

email: bin@xdmsc.club

GitHub: https://github.com/MSC-XDU

Wiki: http://wiki.xdmsc.club