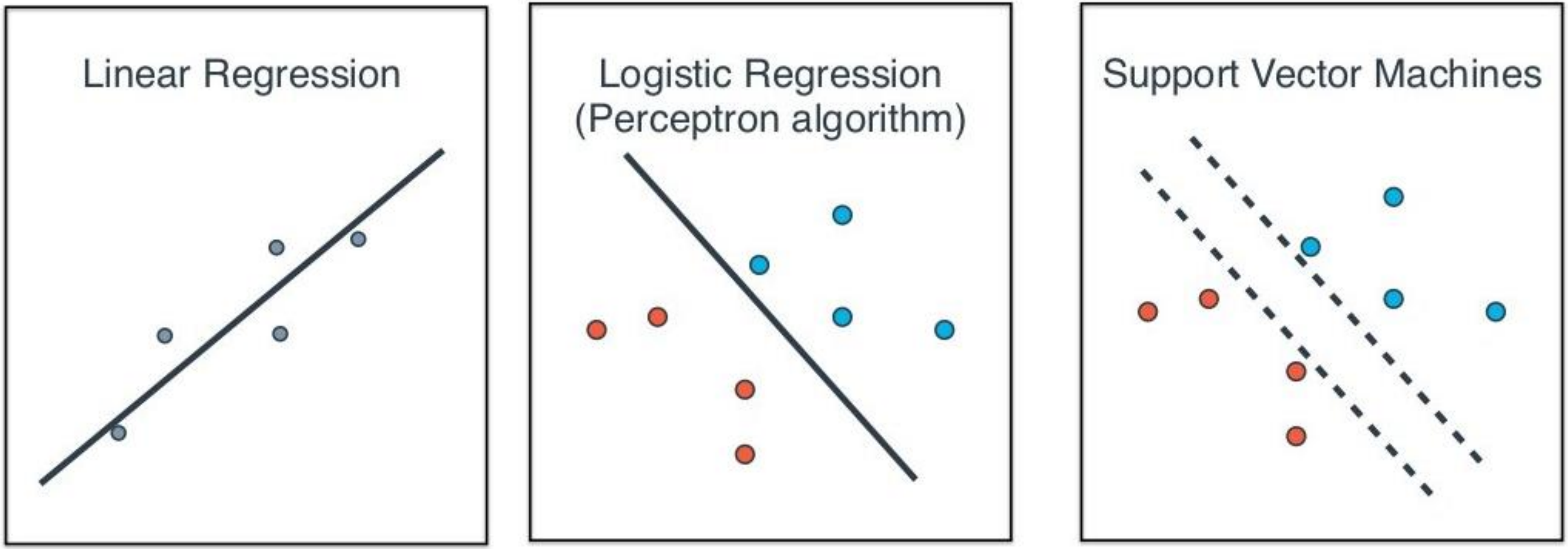
Support Vector Machines (SVM)



Linear Regression

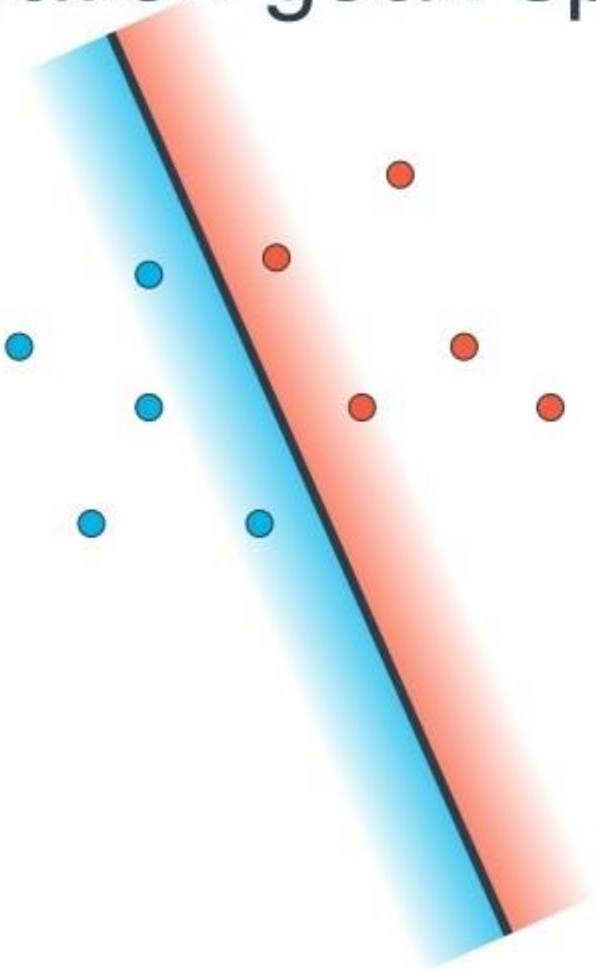
Logistic Regression (Perceptron algorithm)

Support Vector Machines

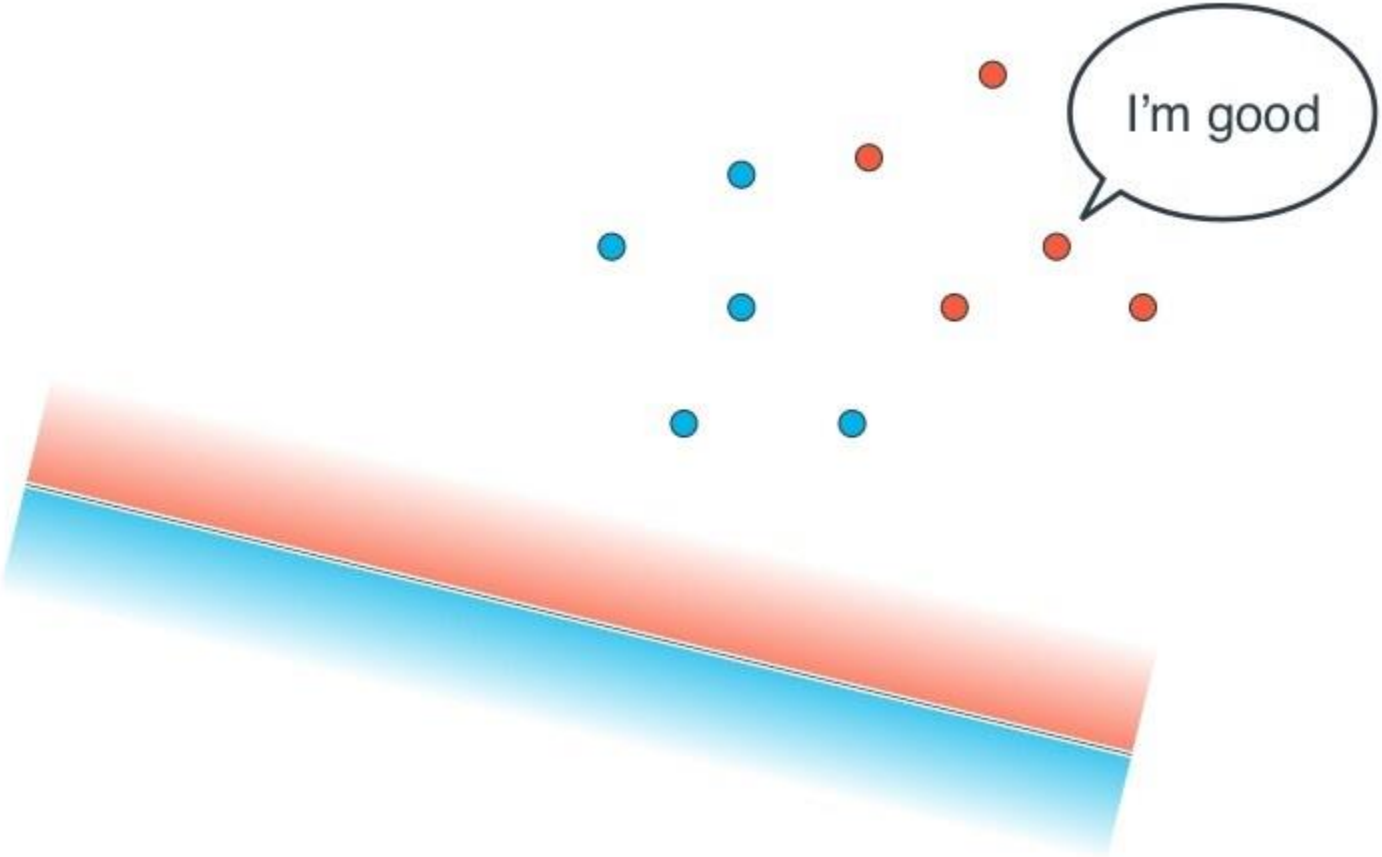
3. Recap on Logistic Regression

Recap on logistic regression and the perceptron algorithm

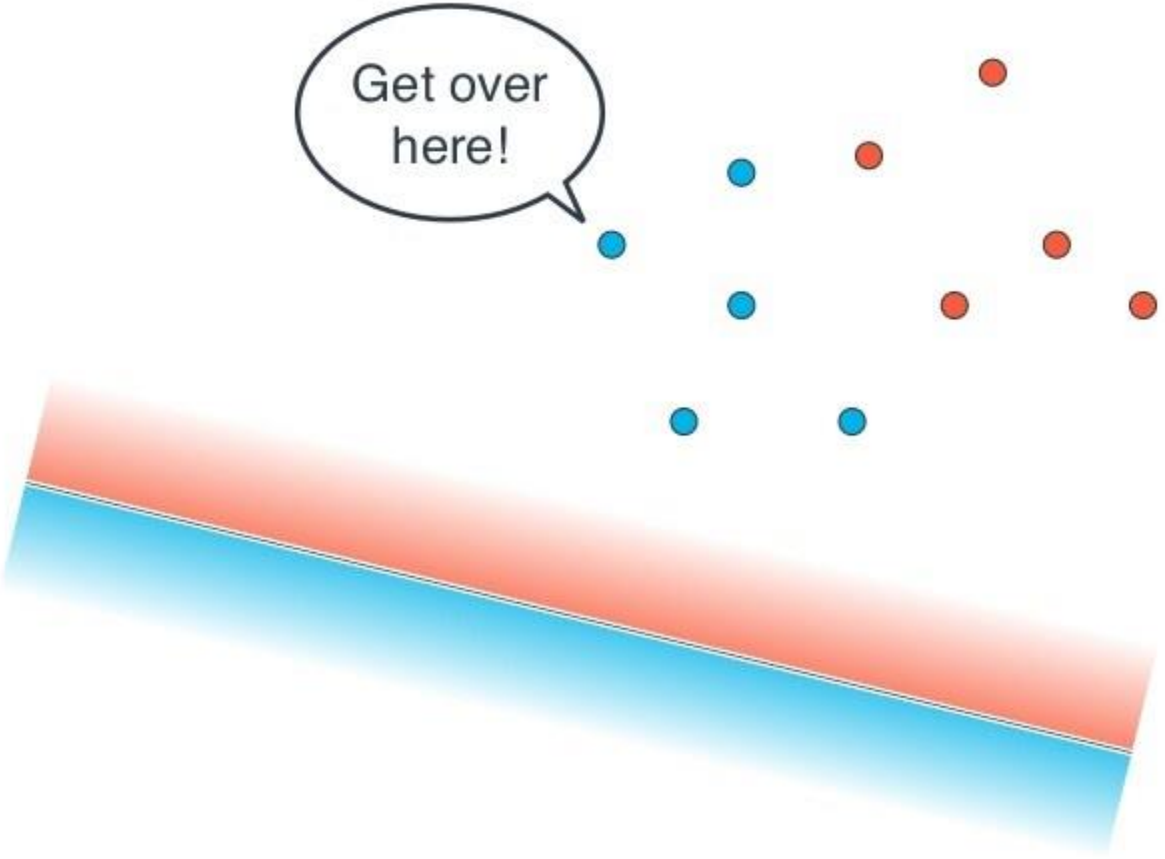
Classification goal: split data



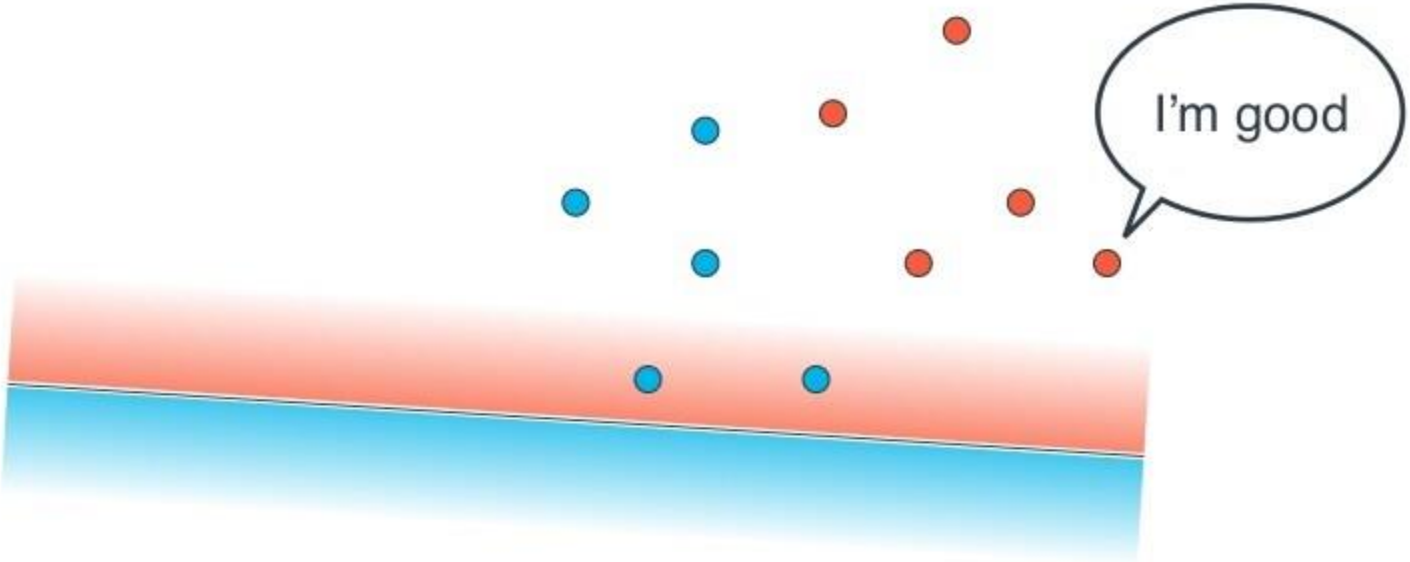
Perceptron Algorithm



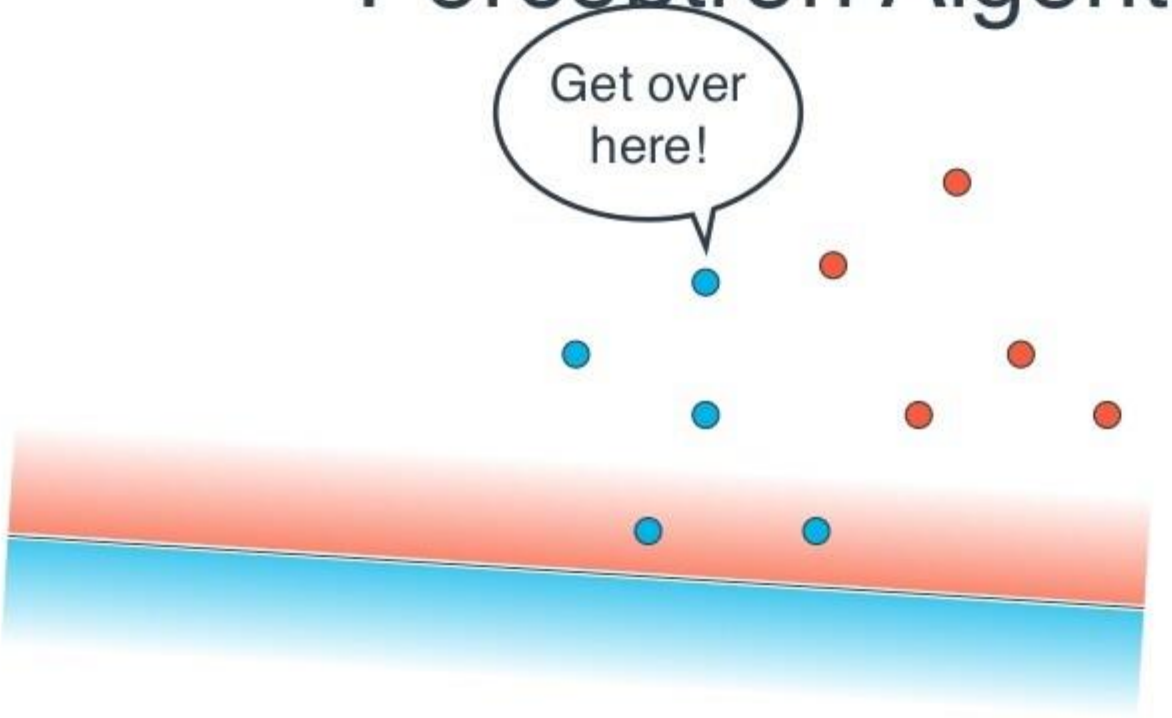
Perceptron Algorithm



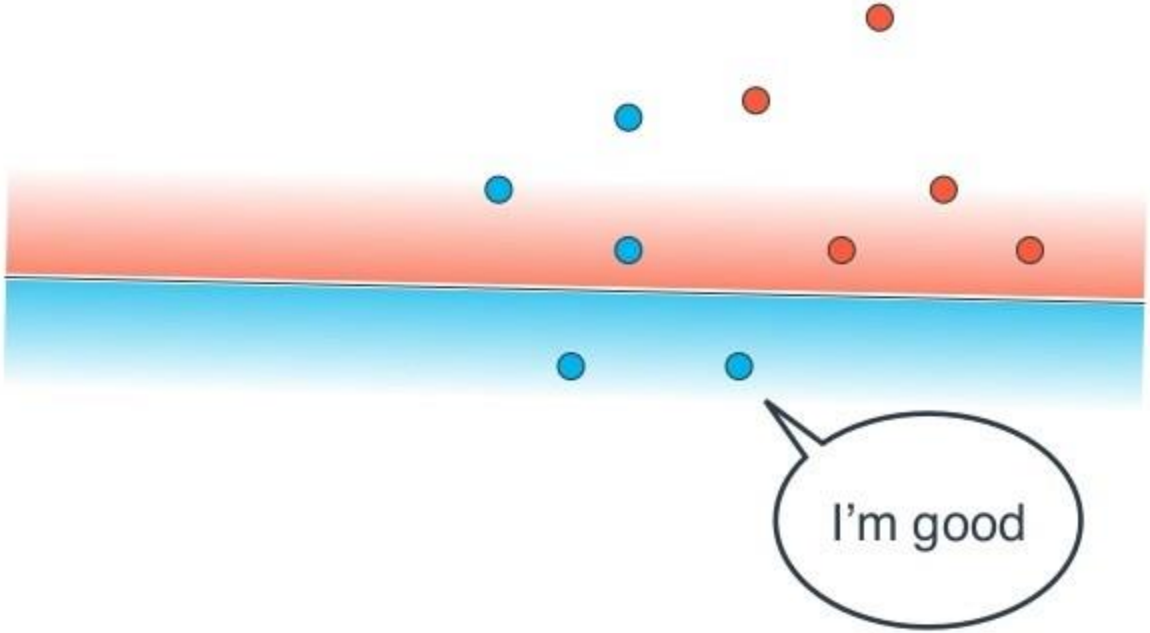
Perceptron Algorithm



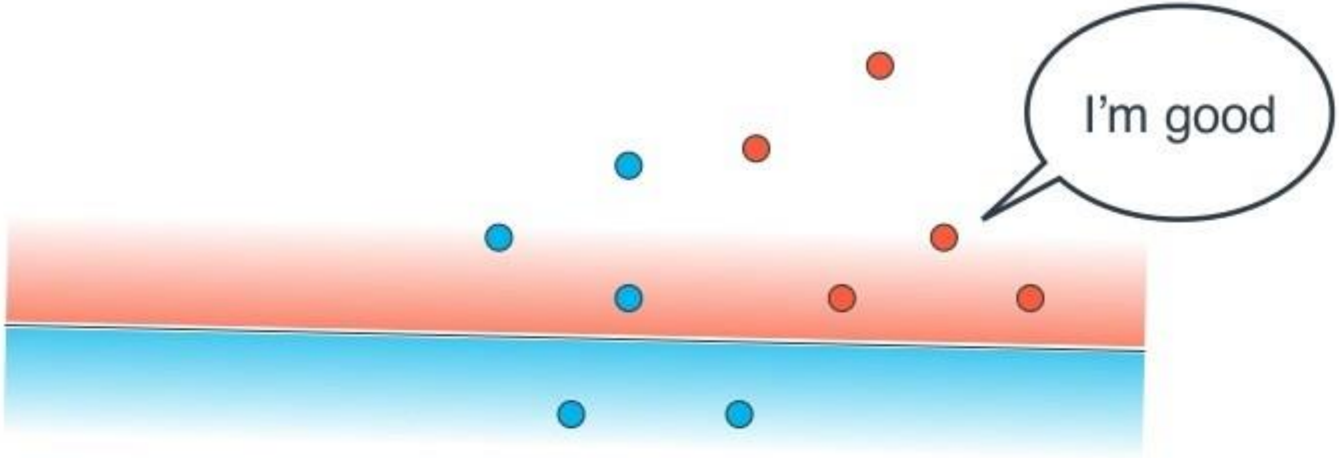
Perceptron Algorithm



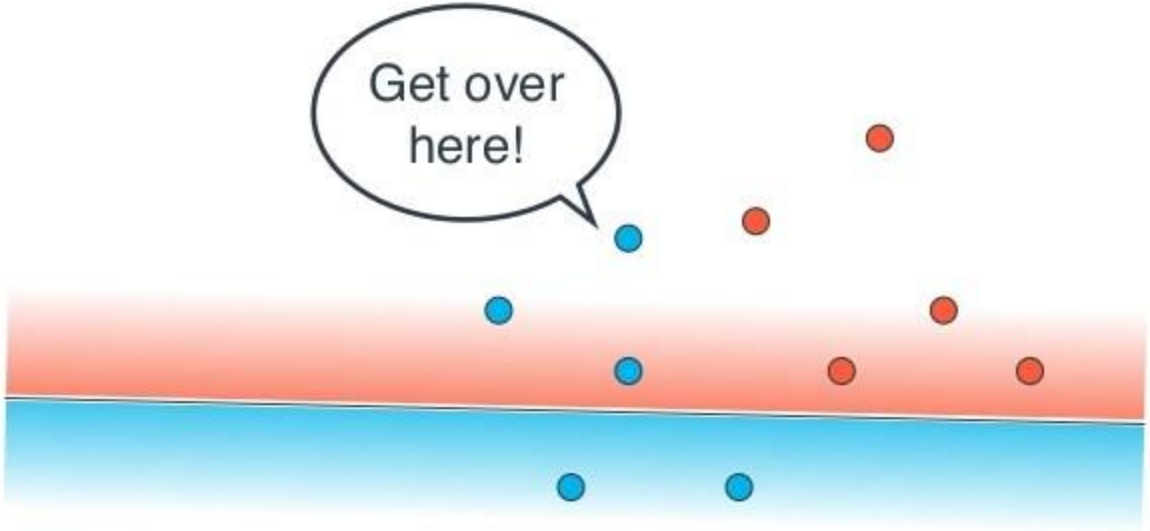
Perceptron Algorithm



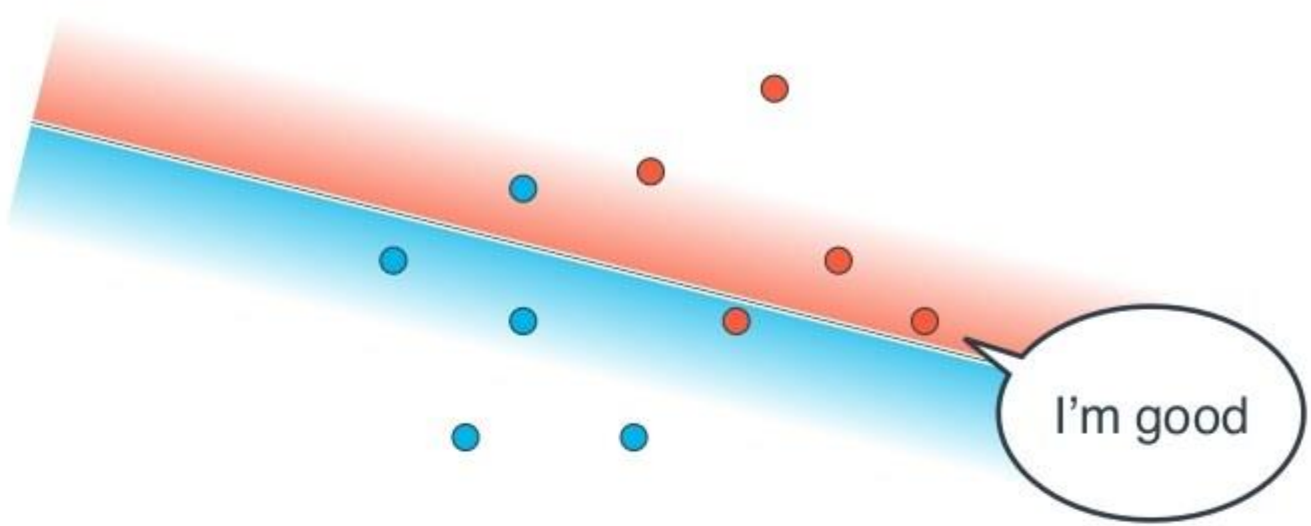
Perceptron Algorithm



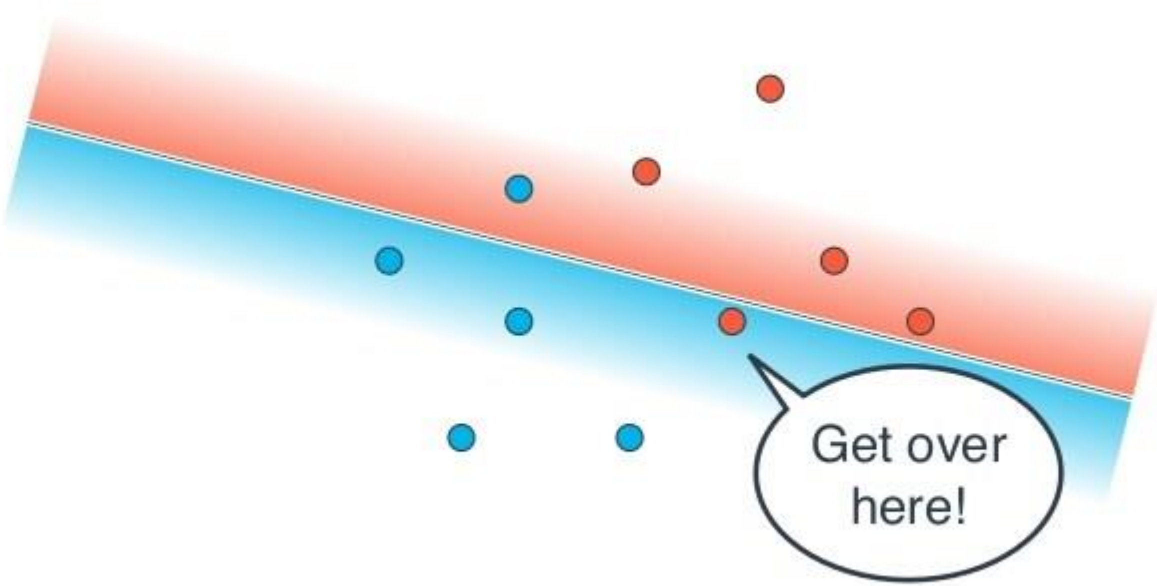
Perceptron Algorithm



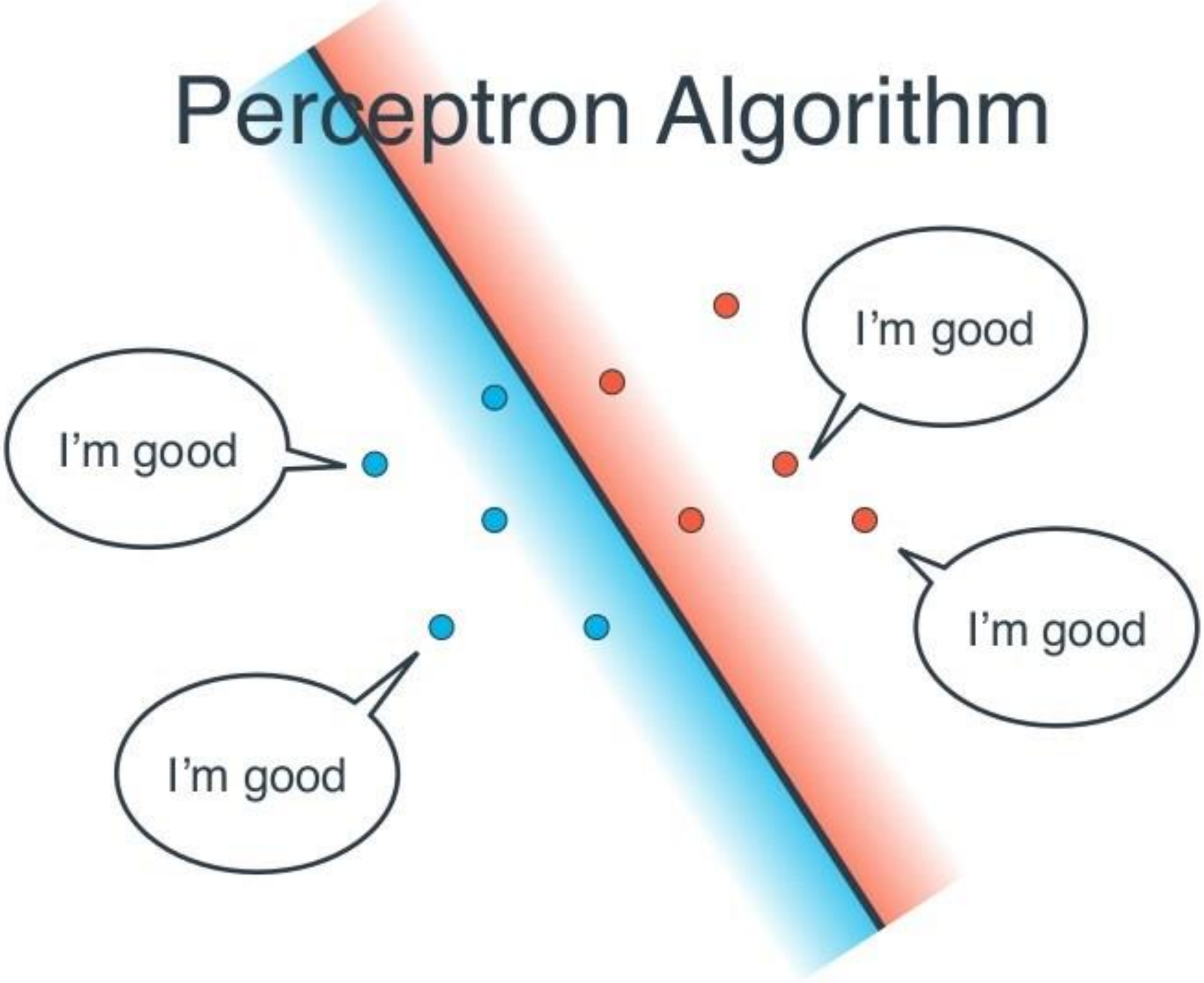
Perceptron Algorithm



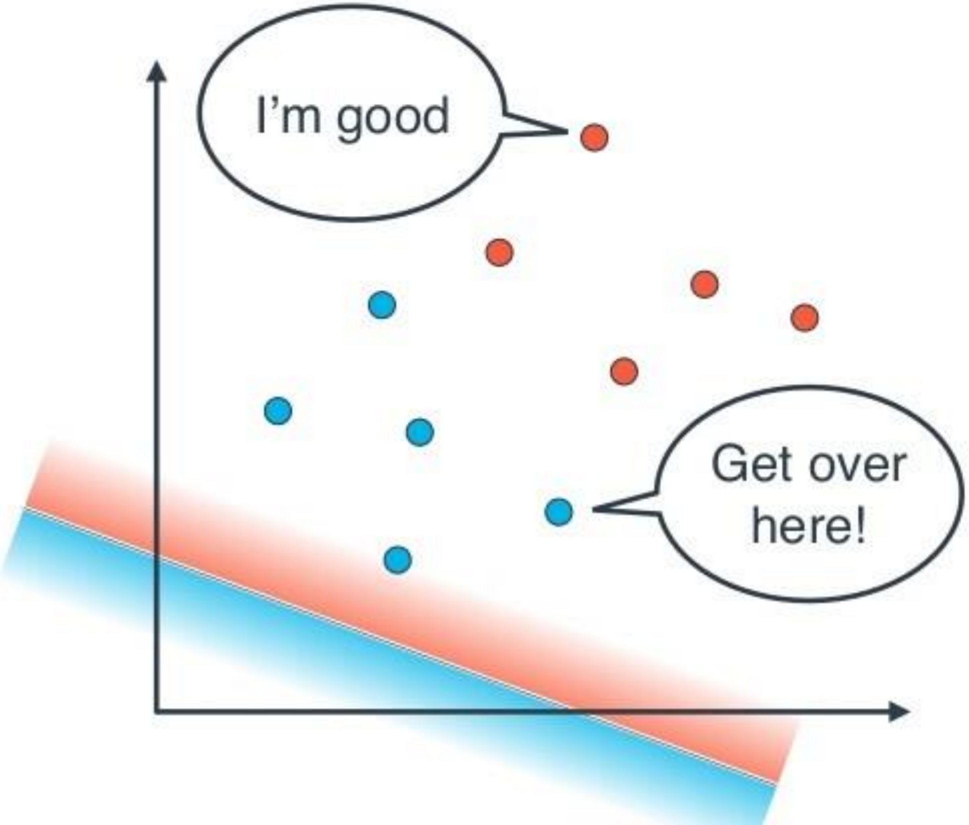
Perceptron Algorithm



Perceptron Algorithm



Perceptron algorithm



Step 1: Start with a random line with blue and red sides.

Step 2: Pick a large number. 1000 (number of repetitions, or epochs)

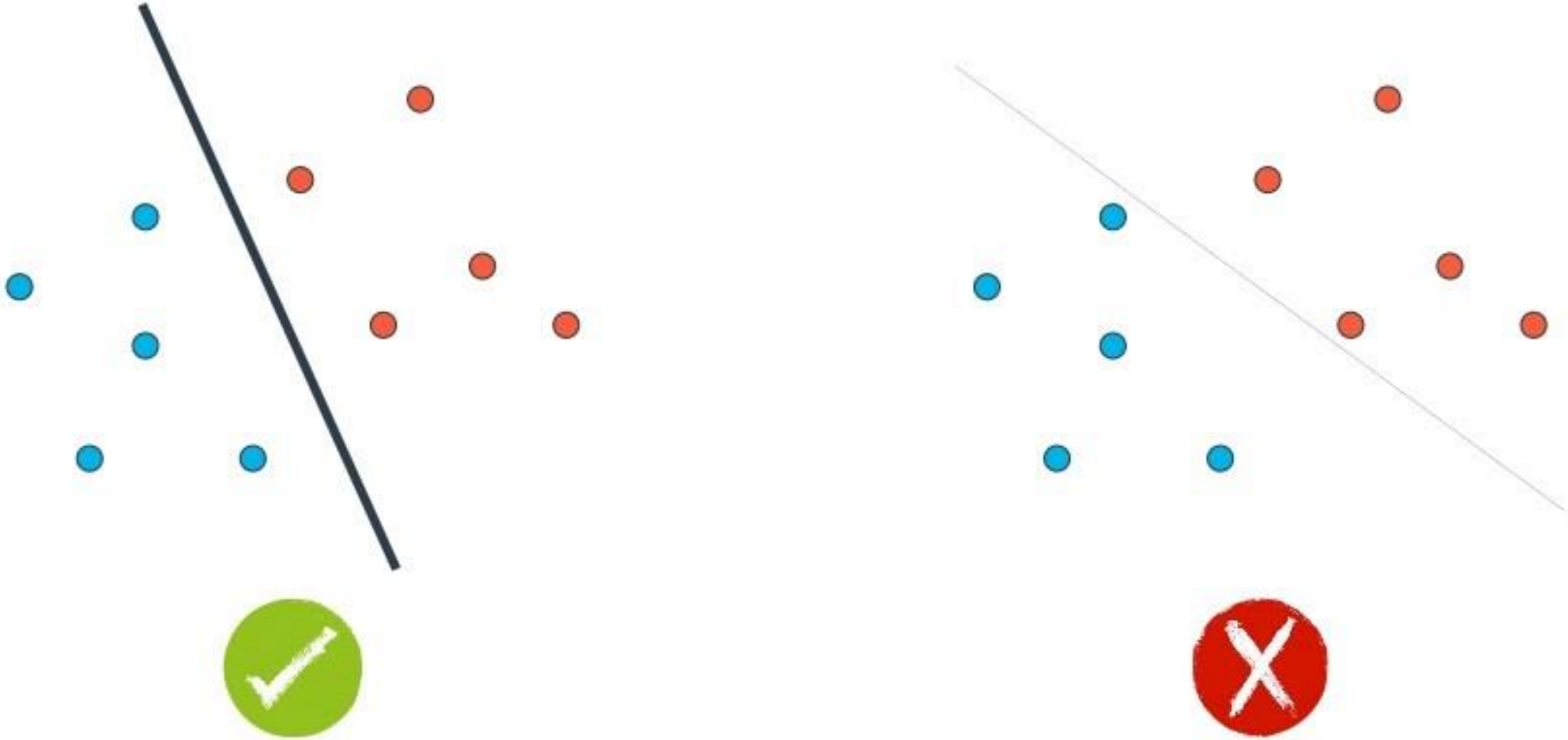
Step 3: (repeat 1000 times)

```
- Pick random point
- If point is correctly classified:
    - Do nothing
- If point is incorrectly classified
    - Move line towards point
```

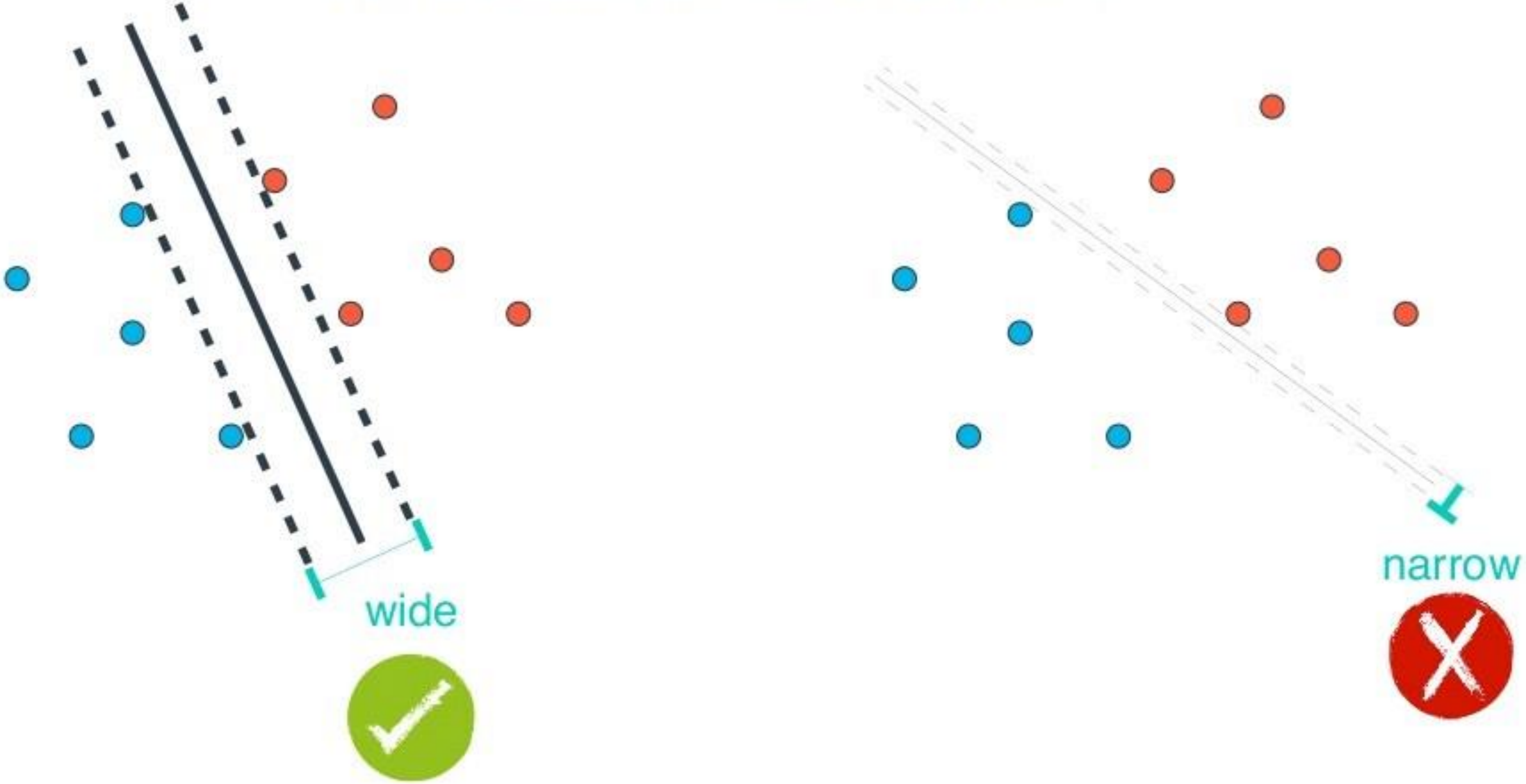
Step 4: Enjoy your line that separates the data!

4. Choice between two lines

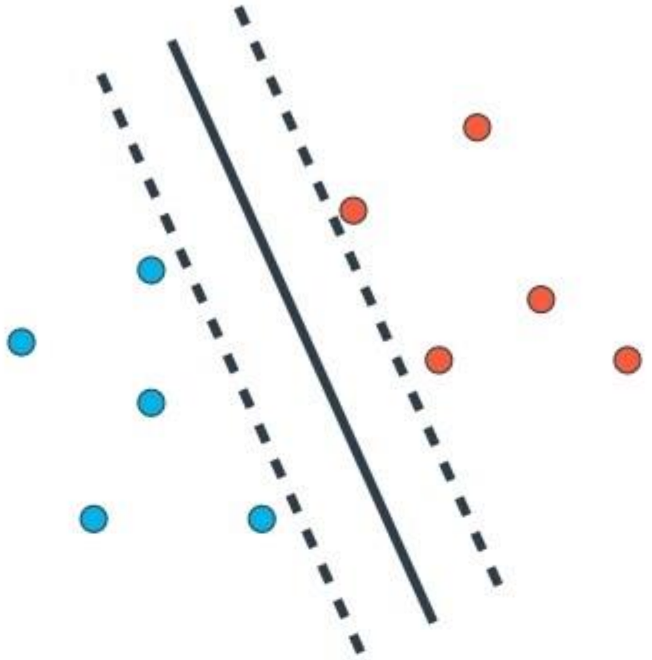
Which line is better?



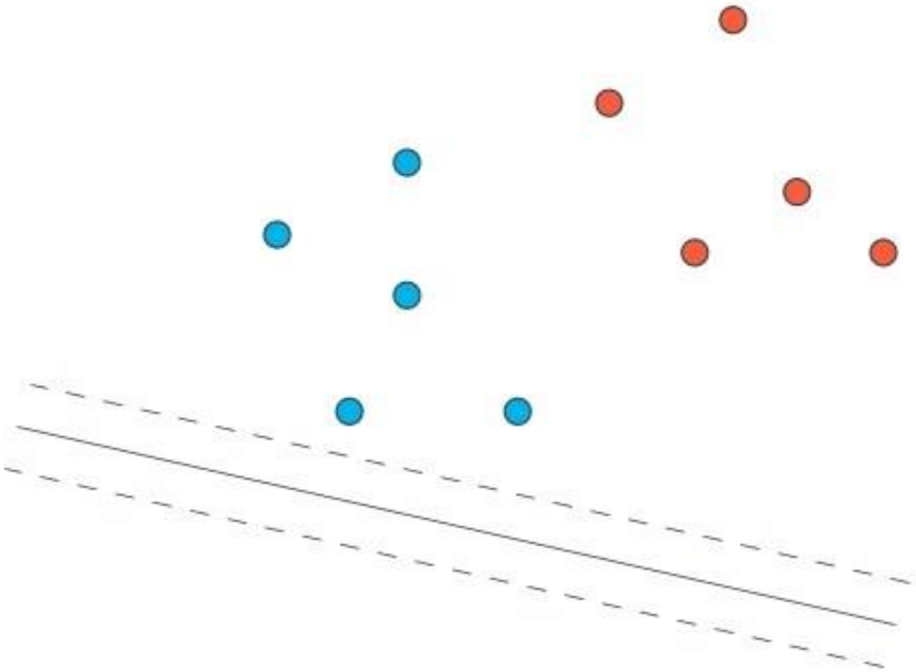
Which line is better?



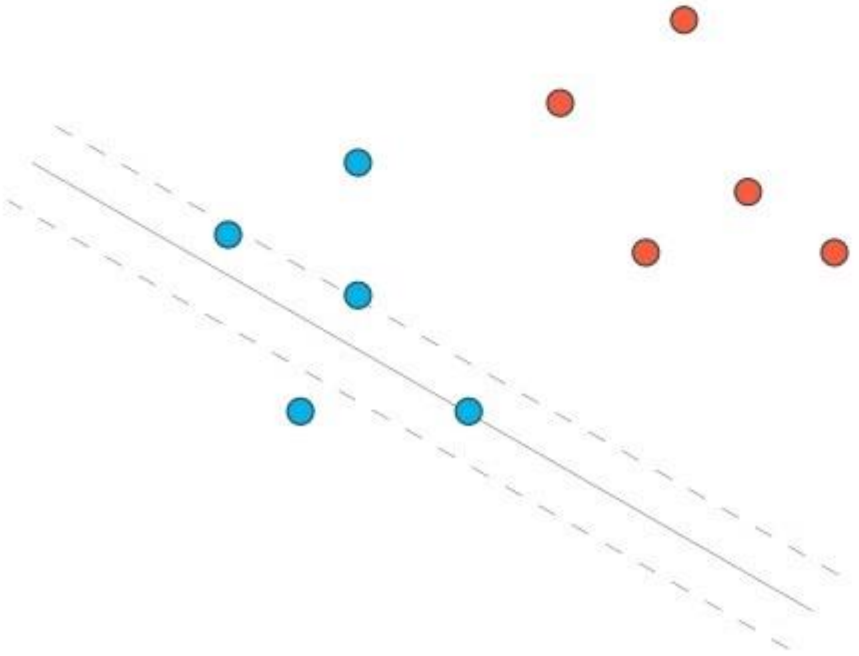
Split data - separate lines



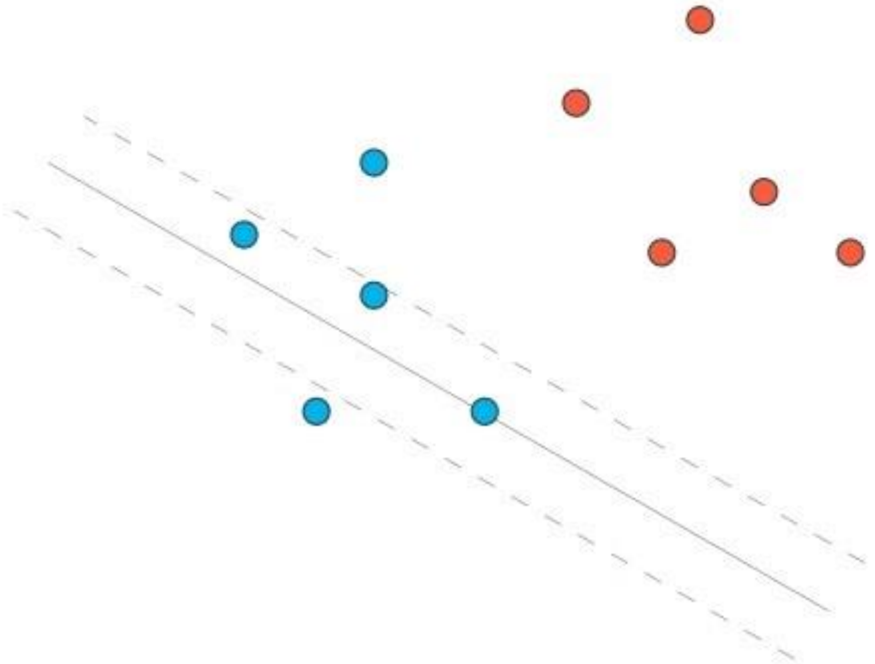
Split data - separate lines



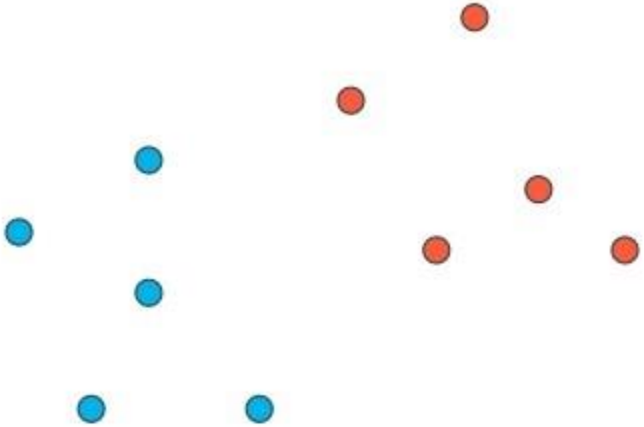
Split data - separate lines



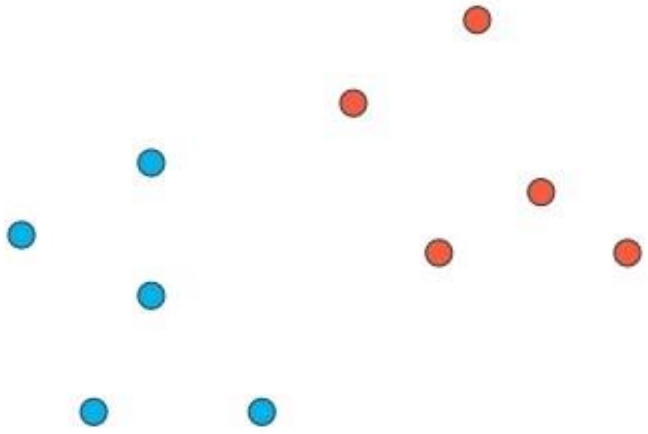
Split data - separate lines



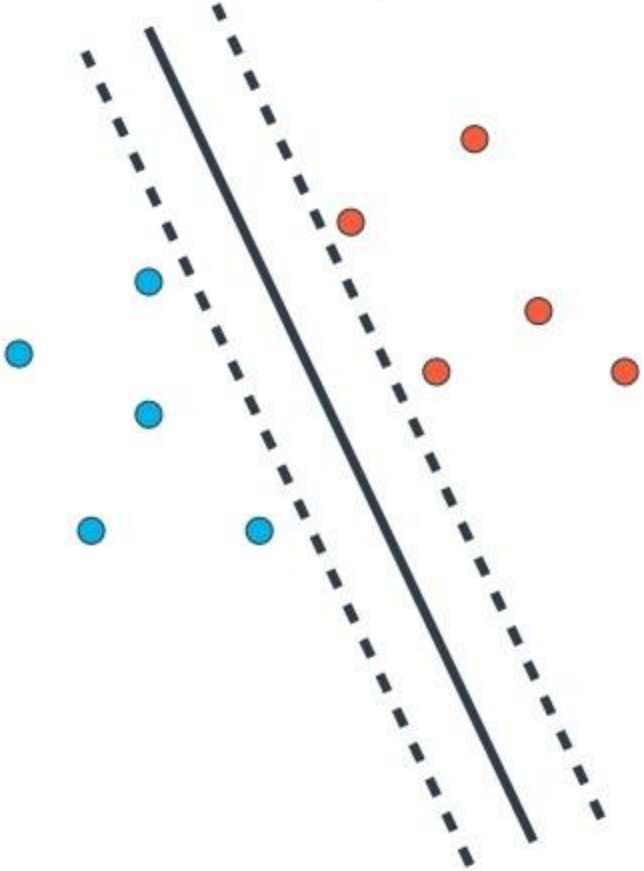
Split data - separate lines



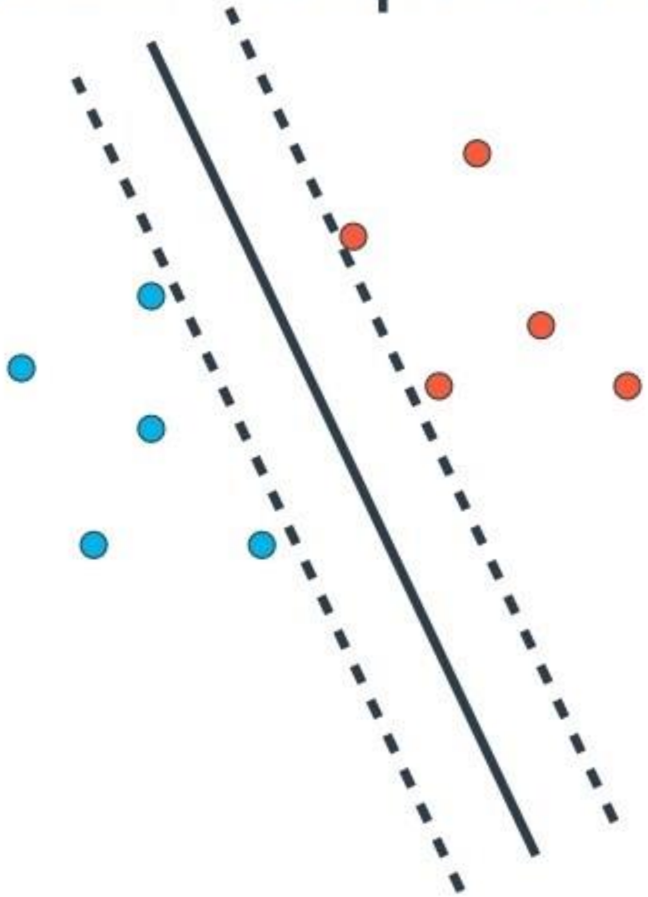
Split data - separate lines



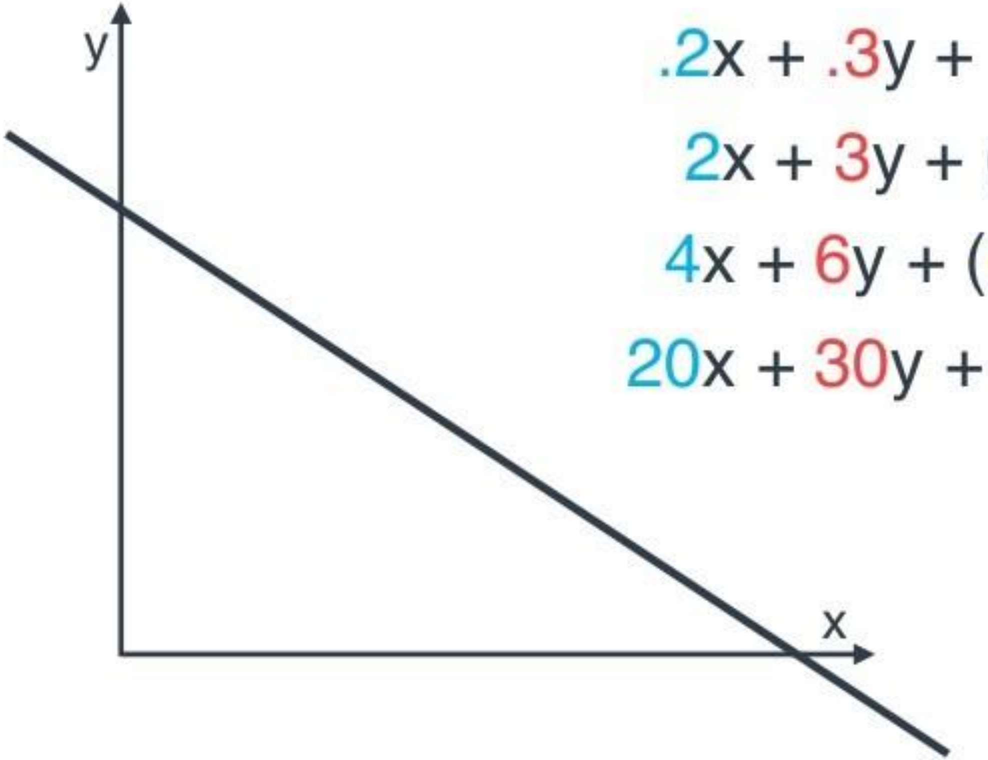
Split data - separate lines



Split data - separate lines



How to separate lines?



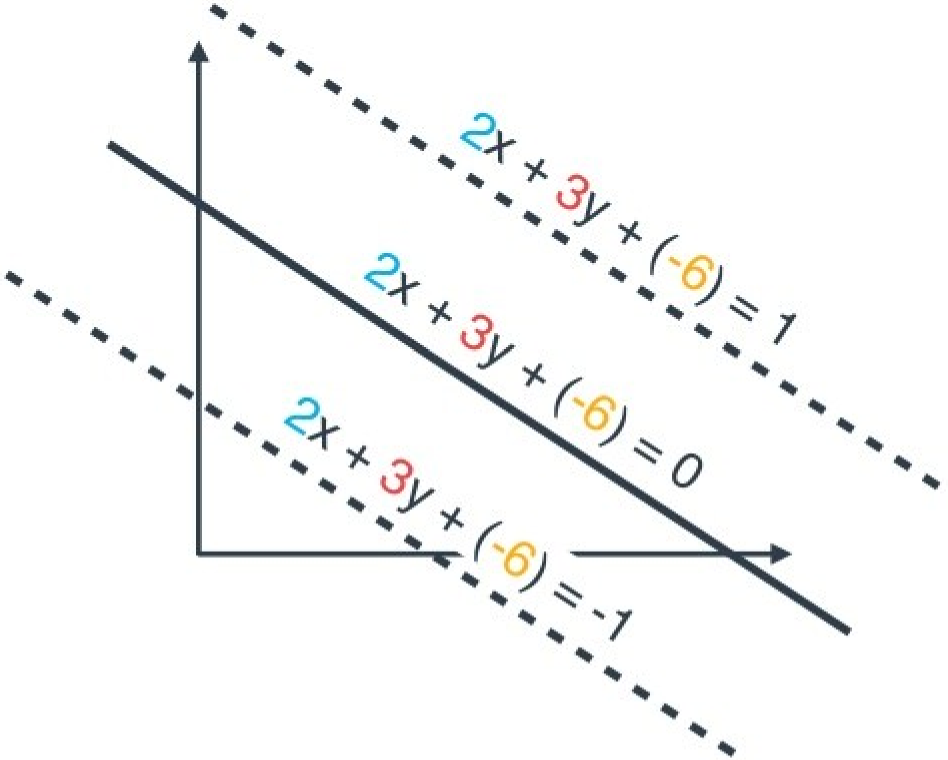
$$.2x + .3y + (-.6) = 0$$

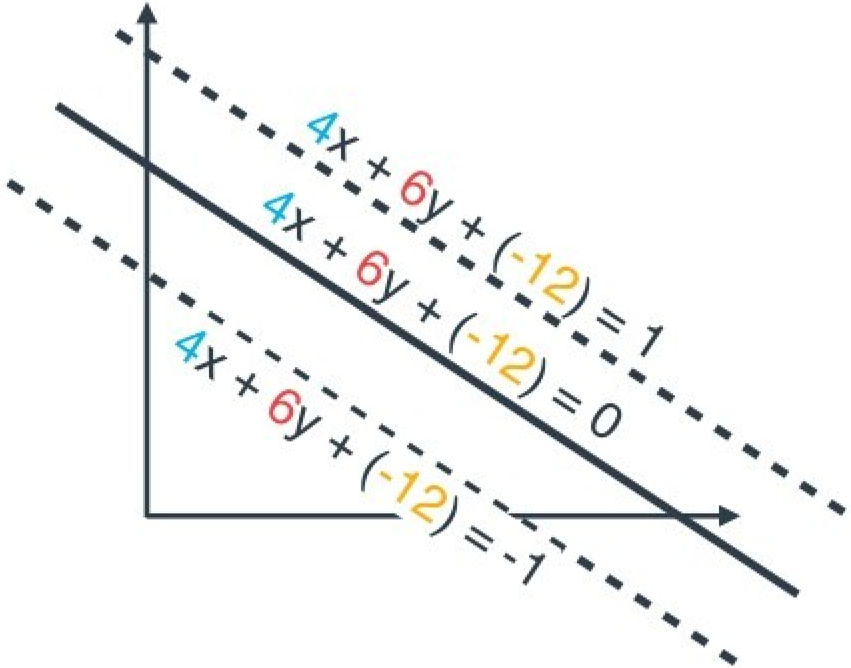
$$2x + 3y + (-6) = 0$$

$$4x + 6y + (-12) = 0$$

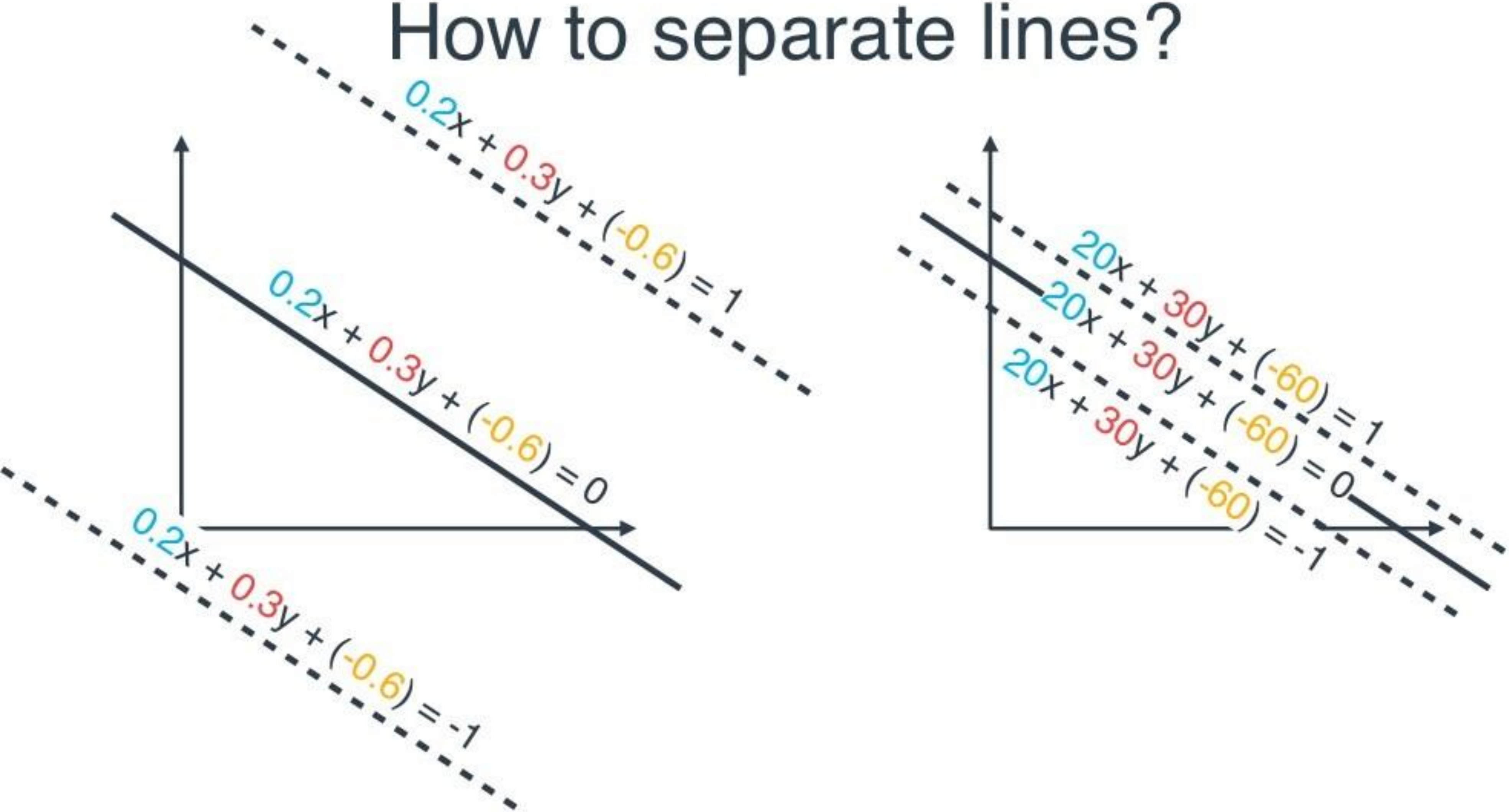
$$20x + 30y + (-60) = 0$$

How to separate lines?



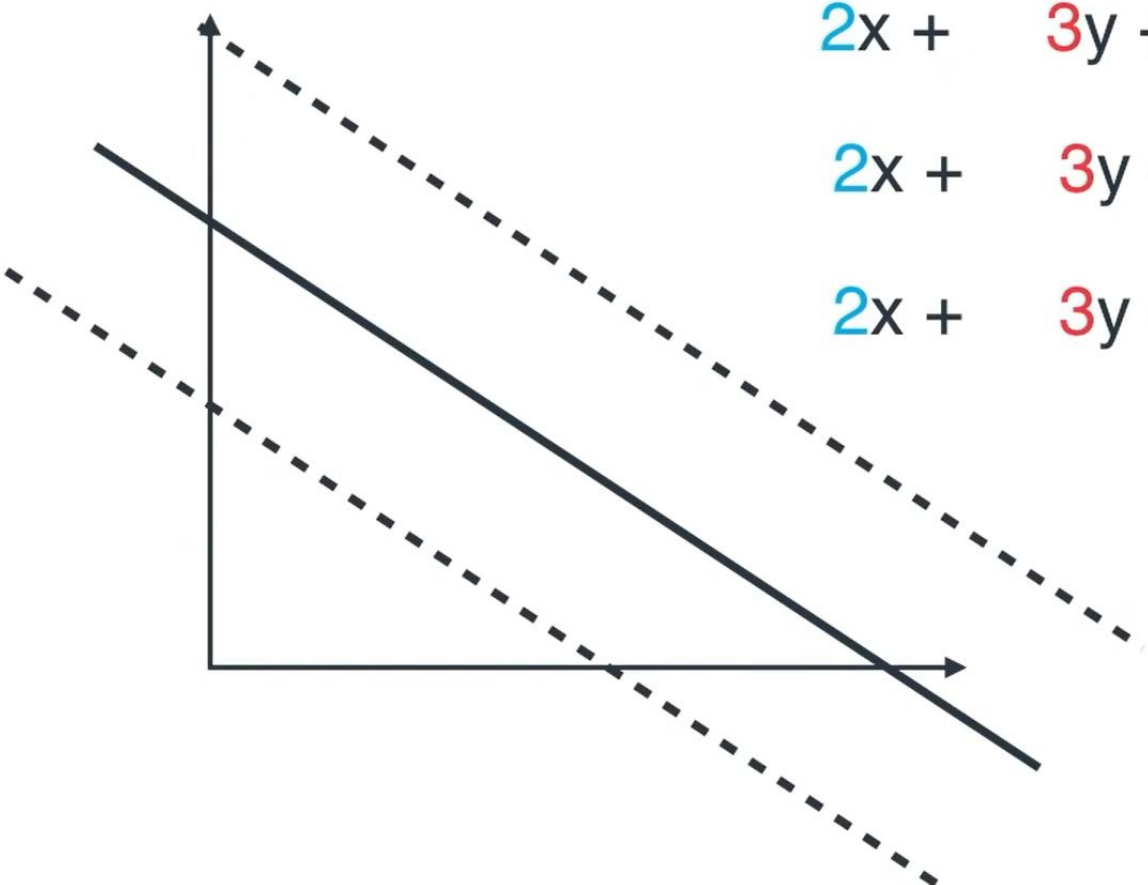


How to separate lines?



$0.2x + 0.3y + (-0.6) = 1$
$0.2x + 0.3y + (-0.6) = 0$
$0.2x + 0.3y + (-0.6) = -1$
$20x + 30y + (-60) = 1$
$20x + 30y + (-60) = 0$
$20x + 30y + (-60) = -1$

Expanding rate

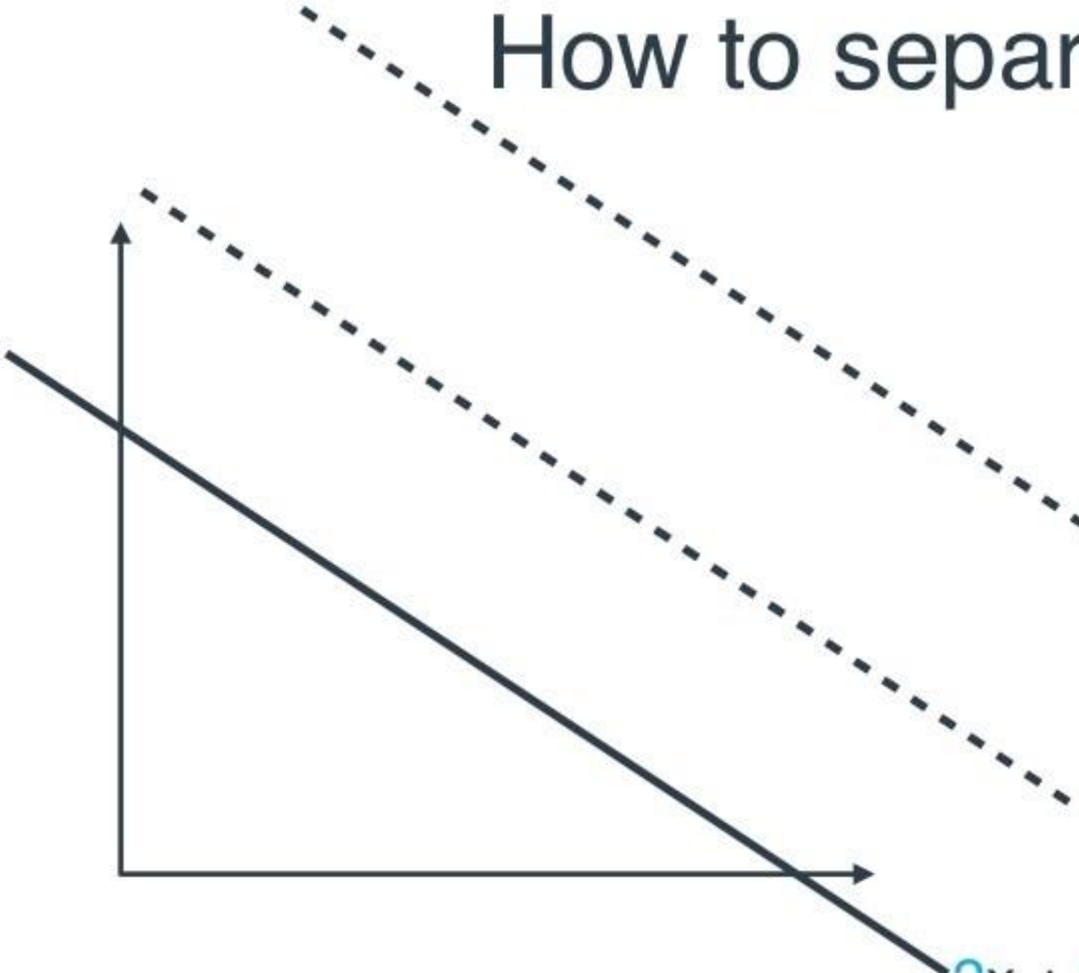


$$2x + 3y + (-6) = -1$$

$$2x + 3y + (-6) = 0$$

$$2x + 3y + (-6) = 1$$

How to separate lines?



$$2x + 3y + (-6) = 0.5$$

$$2x + 3y + (-6) = 1$$

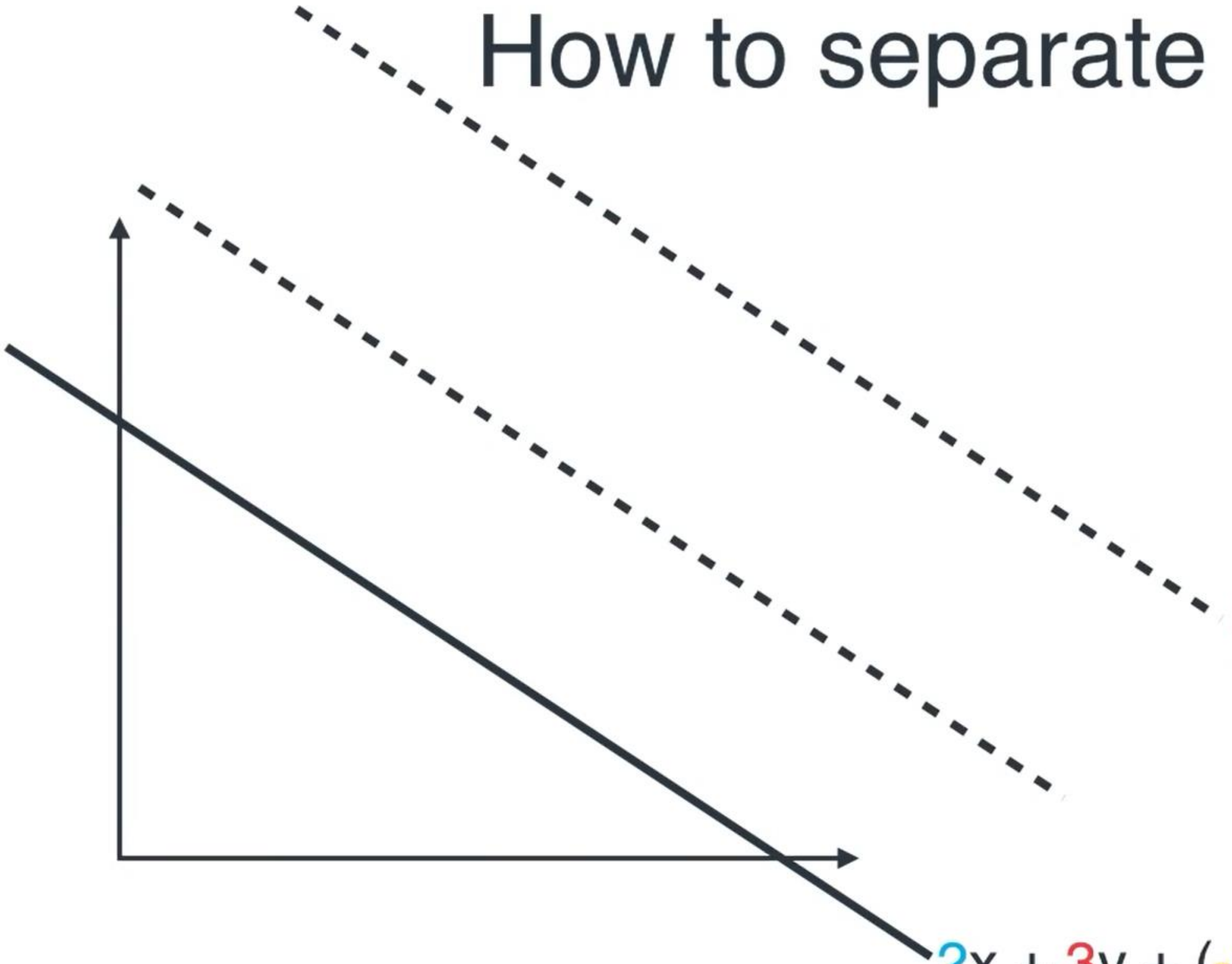
$$4x + 6y + (-12) = 1$$

$$2x + 3y + (-6) = 0$$

How to separate lines?

$$4x + 6y + (-12) = 1$$

$$2x + 3y + (-6) = 0.5$$



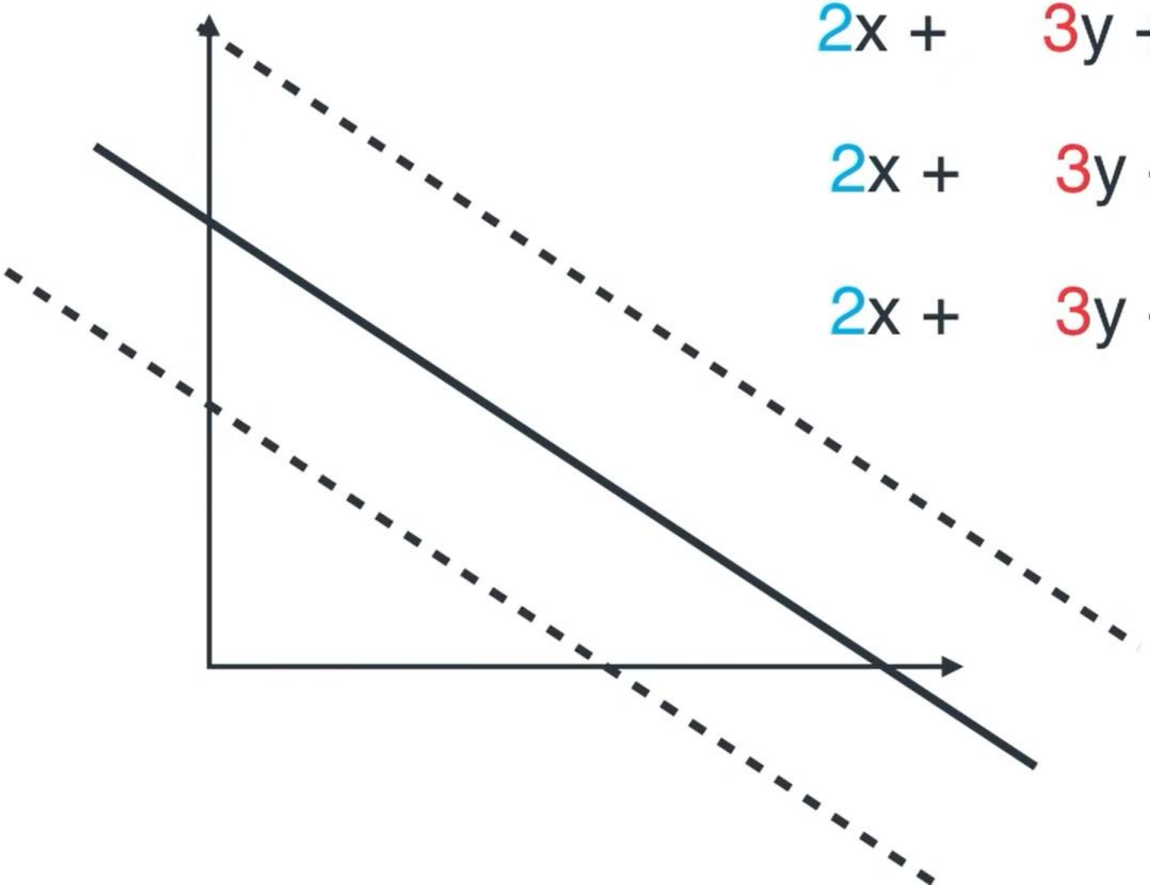
$$2x + 3y + (-6) = 1$$

$$2x + 3y + (-6) = 0$$

Expanding rate

Expanding rate

0.99



$$2x + 3y + (-6) = -1$$

$$2x + 3y + (-6) = 0$$

$$2x + 3y + (-6) = 1$$

Expanding rate

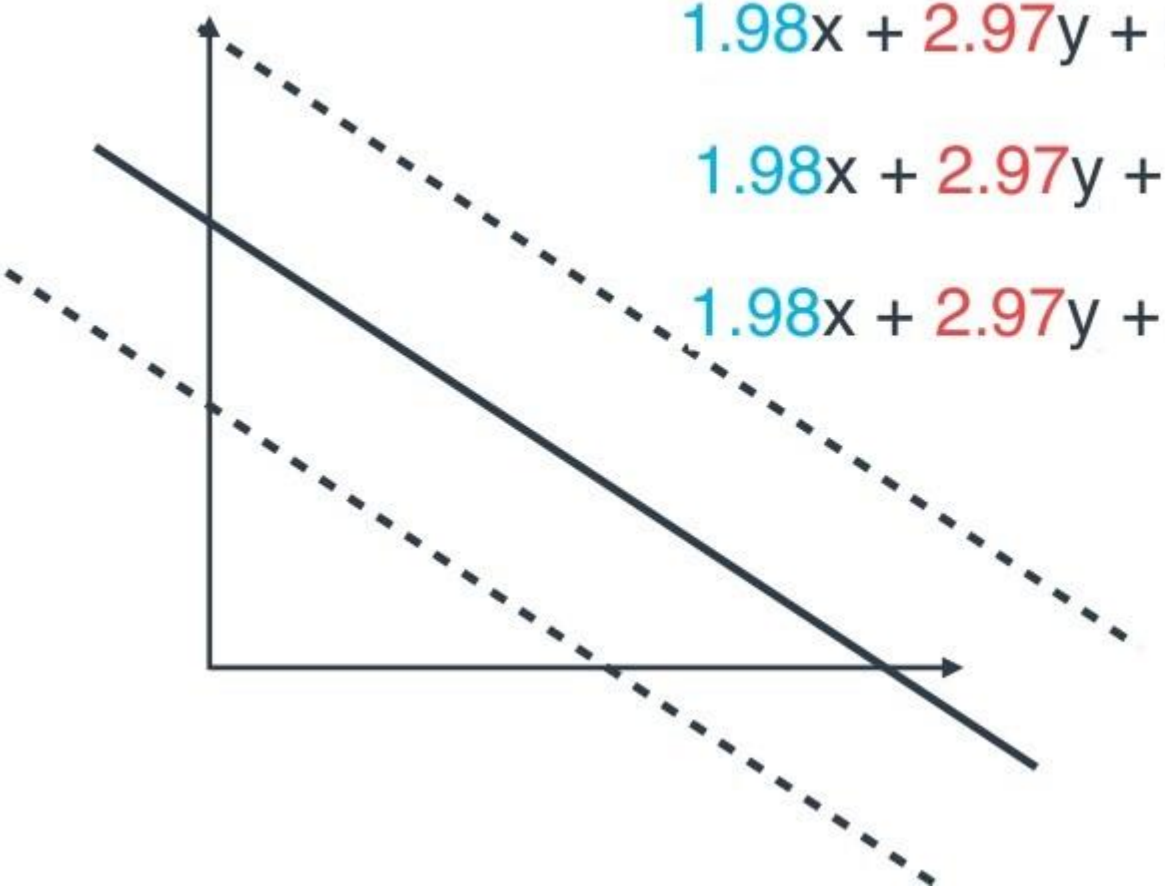
Expanding rate

0.99

$$1.98x + 2.97y + (-5.94) = -1$$

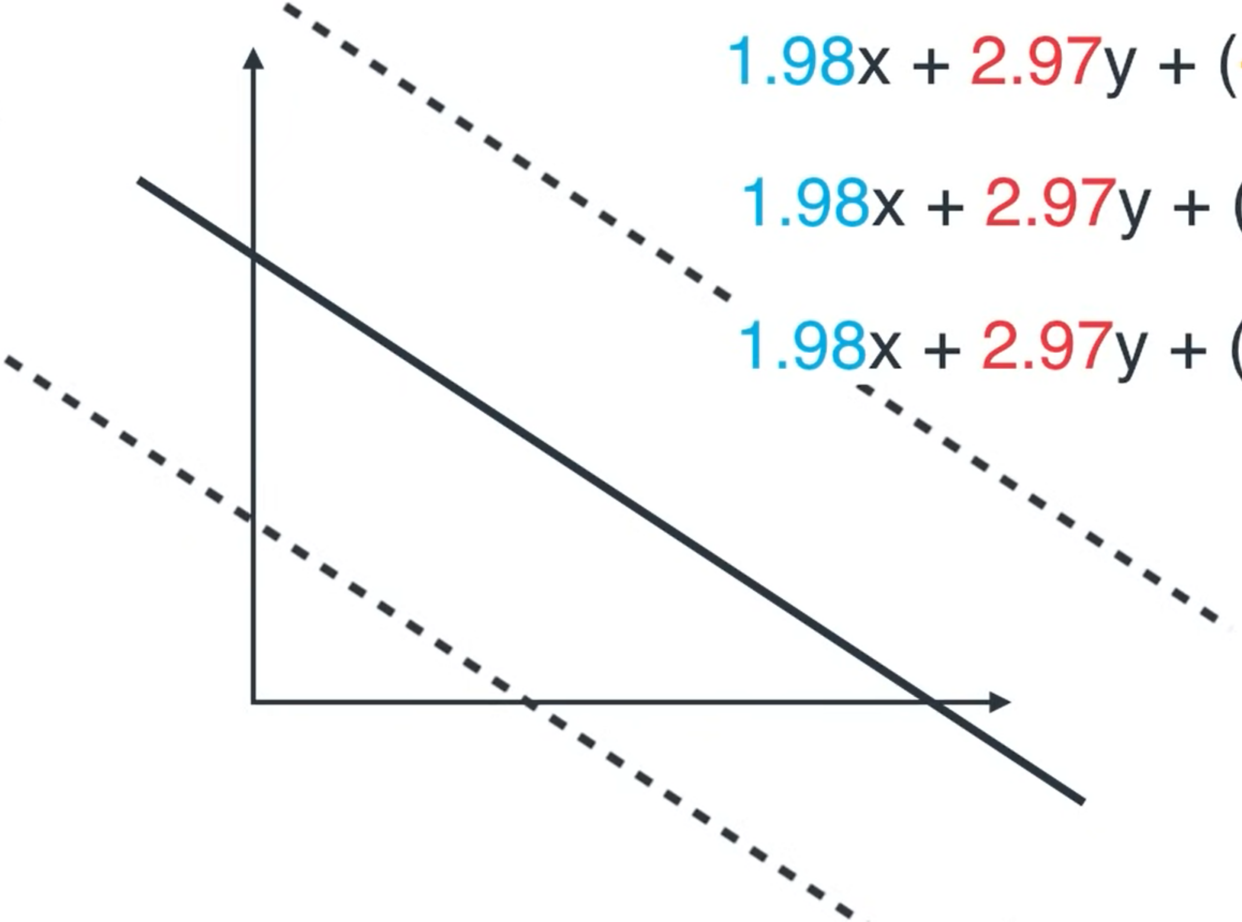
$$1.98x + 2.97y + (-5.94) = 0$$

$$1.98x + 2.97y + (-5.94) = 1$$



Expanding rate

Expanding rate



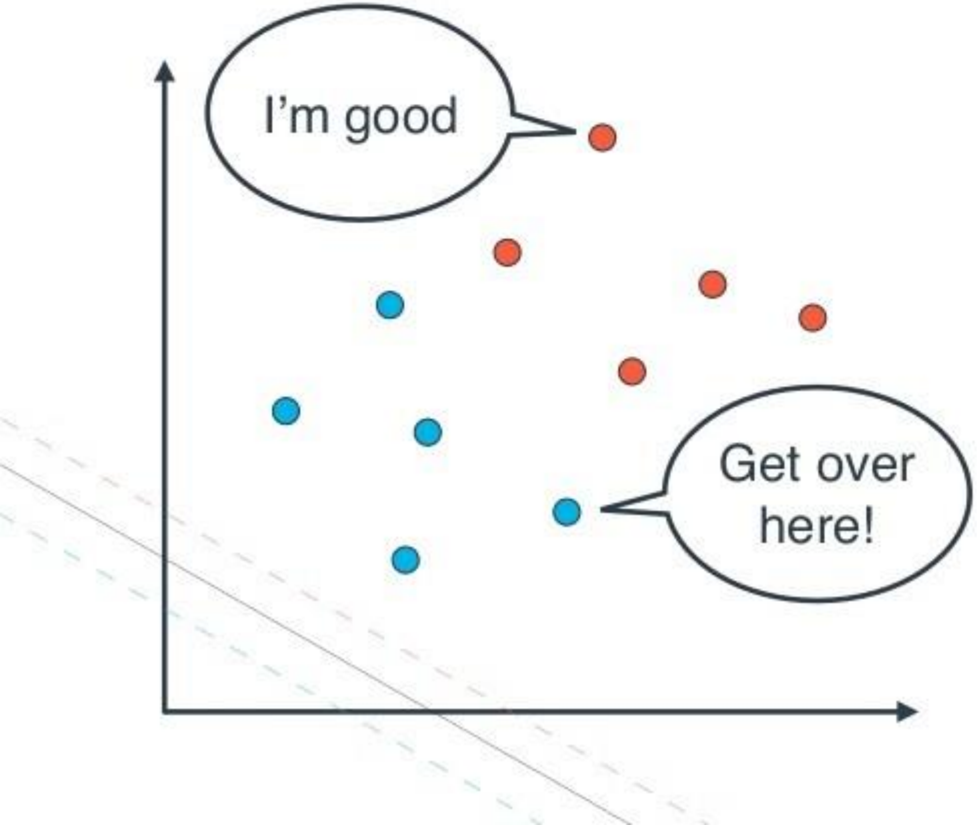
$$1.98x + 2.97y + (-5.94) = -1$$

$$1.98x + 2.97y + (-5.94) = 0$$

$$1.98x + 2.97y + (-5.94) = 1$$

5. SVM algorithm SC

SVM algorithm



Step 1: Start with a line, and two equidistant parallel lines to it.

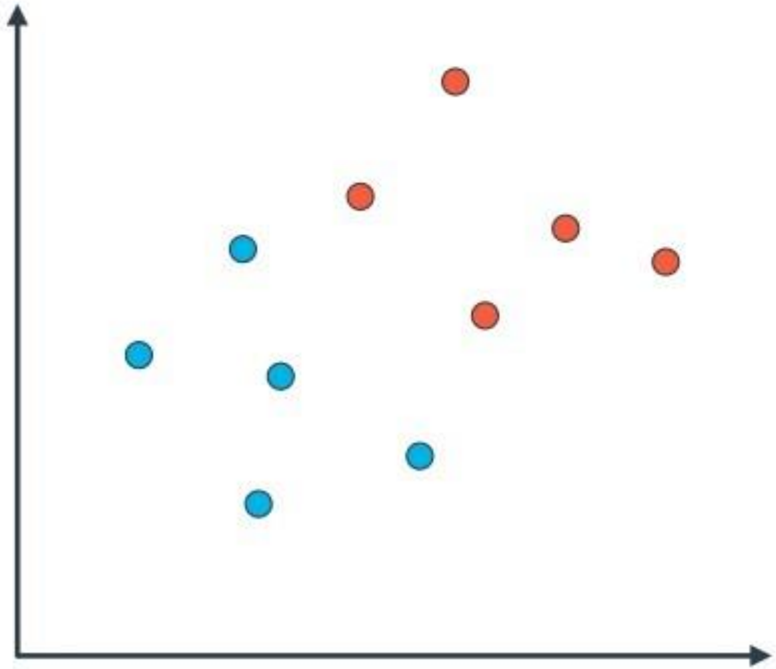
Step 2: Pick a large number. 1000 (number of repetitions, or epochs)

Step 3: Pick a number close to 1. (the expanding factor) 0.99

Step 4: (repeat 1000 times)

- Pick random point
- If point is correctly classified:
  - Do nothing
- If point is incorrectly classified
  - Move line towards point

SVM algorithm



Step 1: Start with a line, and two equidistant parallel lines to it.

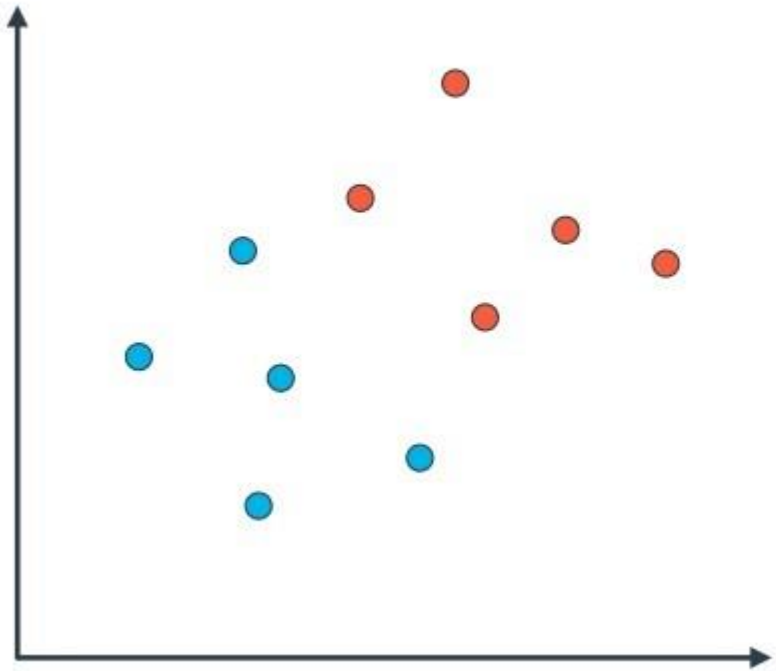
Step 2: Pick a large number. 1000 (number of repetitions, or epochs)

Step 3: Pick a number close to 1. (the expanding factor) 0.99

Step 4: (repeat 1000 times)

- Pick random point
- If point is correctly classified:
    - Do nothing
- If point is incorrectly classified
    - Move line towards point

SVM algorithm



Step 1: Start with a line, and two equidistant parallel lines to it.

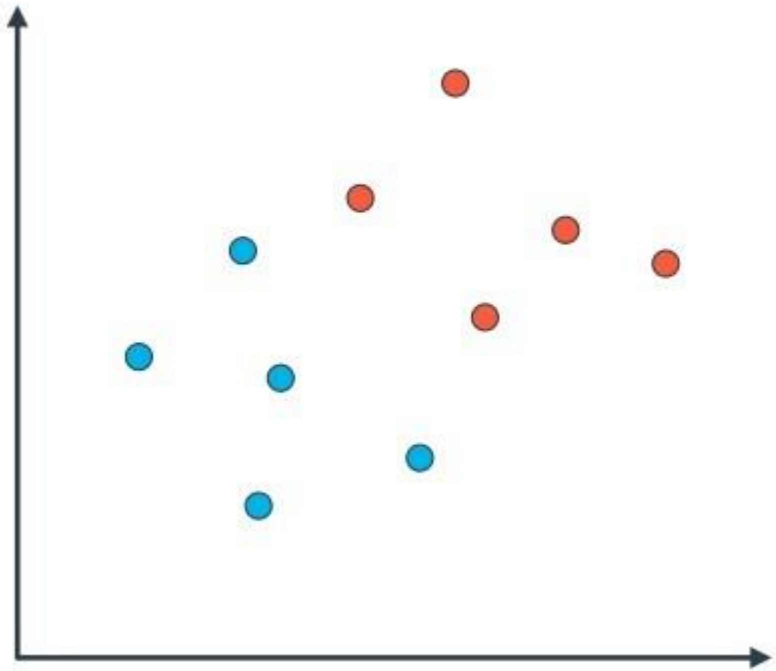
Step 2: Pick a large number. 1000 (number of repetitions, or epochs)

Step 3: Pick a number close to 1. (the expanding factor) 0.99

Step 4: (repeat 1000 times)

- Pick random point
- If point is correctly classified:
    - Do nothing
- If point is incorrectly classified
    - Move line towards point

SVM algorithm



Step 1: Start with a line, and two equidistant parallel lines to it.

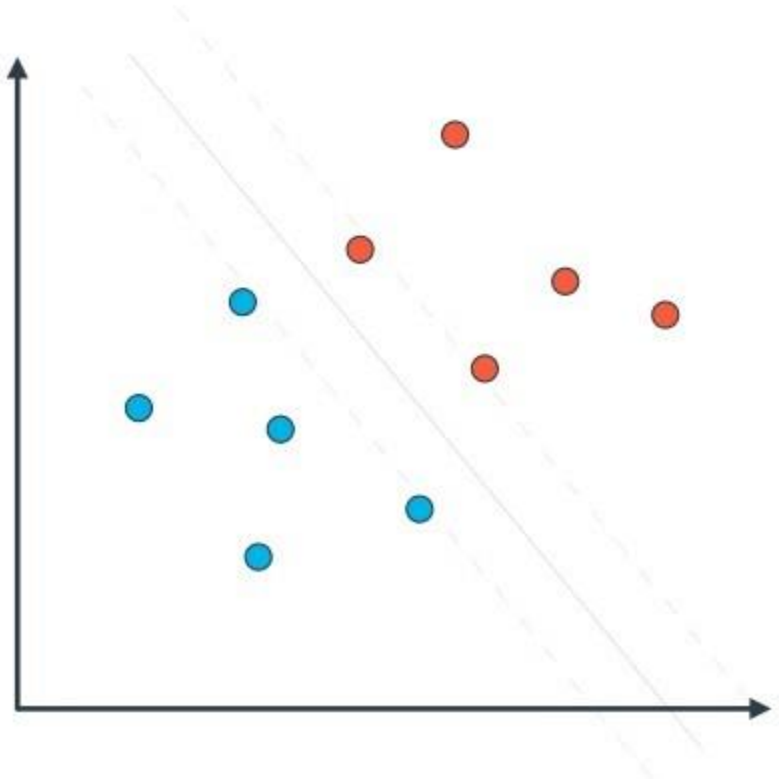
Step 2: Pick a large number. 1000 (number of repetitions, or epochs)

Step 3: Pick a number close to 1. (the expanding factor) 0.99

Step 4: (repeat 1000 times)

- Pick random point
- If point is correctly classified:
    - Do nothing
- If point is incorrectly classified
    - Move line towards point
- Separate the lines using the expanding factor

SVM algorithm



Step 1: Start with a line, and two equidistant parallel lines to it.

Step 2: Pick a large number. 1000 (number of repetitions, or epochs)

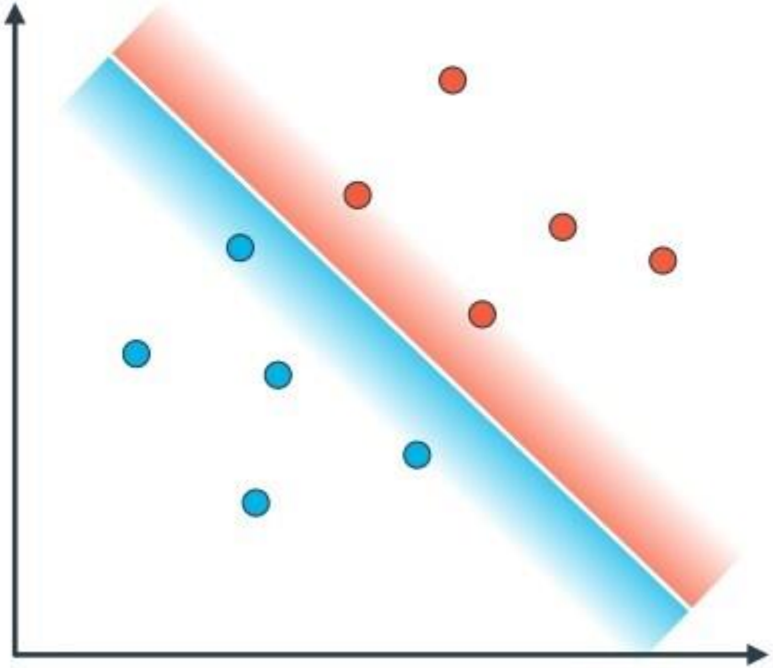
Step 3: Pick a number close to 1. (the expanding factor) 0.99

Step 4: (repeat 1000 times)

- Pick random point
- If point is correctly classified:
    - Do nothing
- If point is incorrectly classified
    - Move line towards point
- Separate the lines using the expanding factor

Step 5: Enjoy your lines that separate the data!

Perceptron algorithm



Step 1: Start with a random line of equation ax + by + c = 0

Step 2: Pick a large number. **1000** (number of repetitions, or epochs)

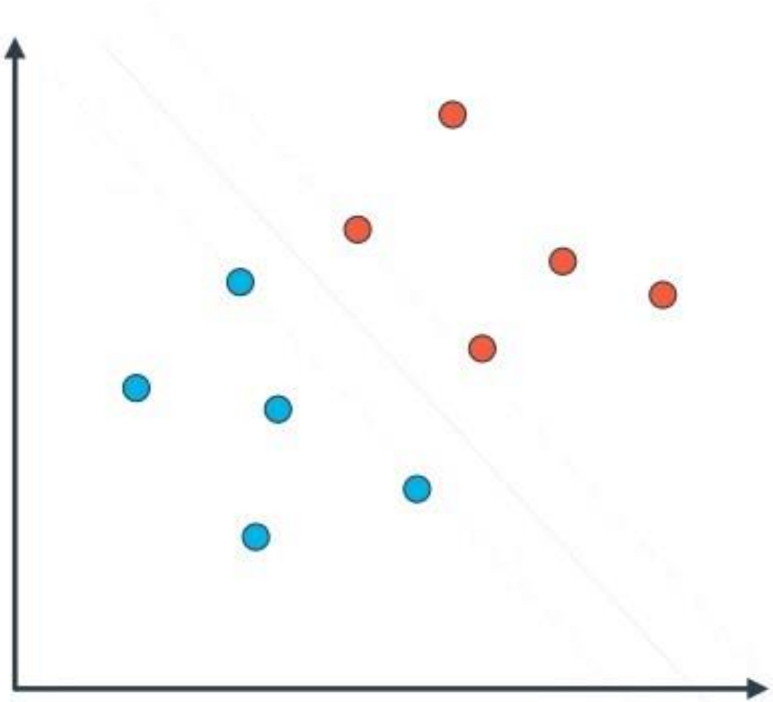
Step 3: Pick a small number. **0.01** (learning rate)

Step 4: (repeat **1000** times)

- Pick random point **(p,q)**
- If point is correctly classified
  - Do nothing
- If point is blue, and ap+bq+c > 0
  - Subtract 0.01**p** to a
  - Subtract 0.01**q** to b
  - Subtract 0.01 to c
- If point is, red and ap+bq+c < 0
  - Add 0.01**p** to a
  - Add 0.01**q** to b
  - Add 0.01 to c

Step 5: Enjoy your line!

SVM algorithm



Step 1: Start with a random line of equation ax + by + c = 0.
Draw parallel lines with equations:

- ax + by + c = 1, and
- ax + by + c = -1

Step 2: Pick a large number. **1000** (number of repetitions, or epochs)

Step 3: Pick a learning rate. **0.01**

Step 4: Pick an expanding rate. **0.99**

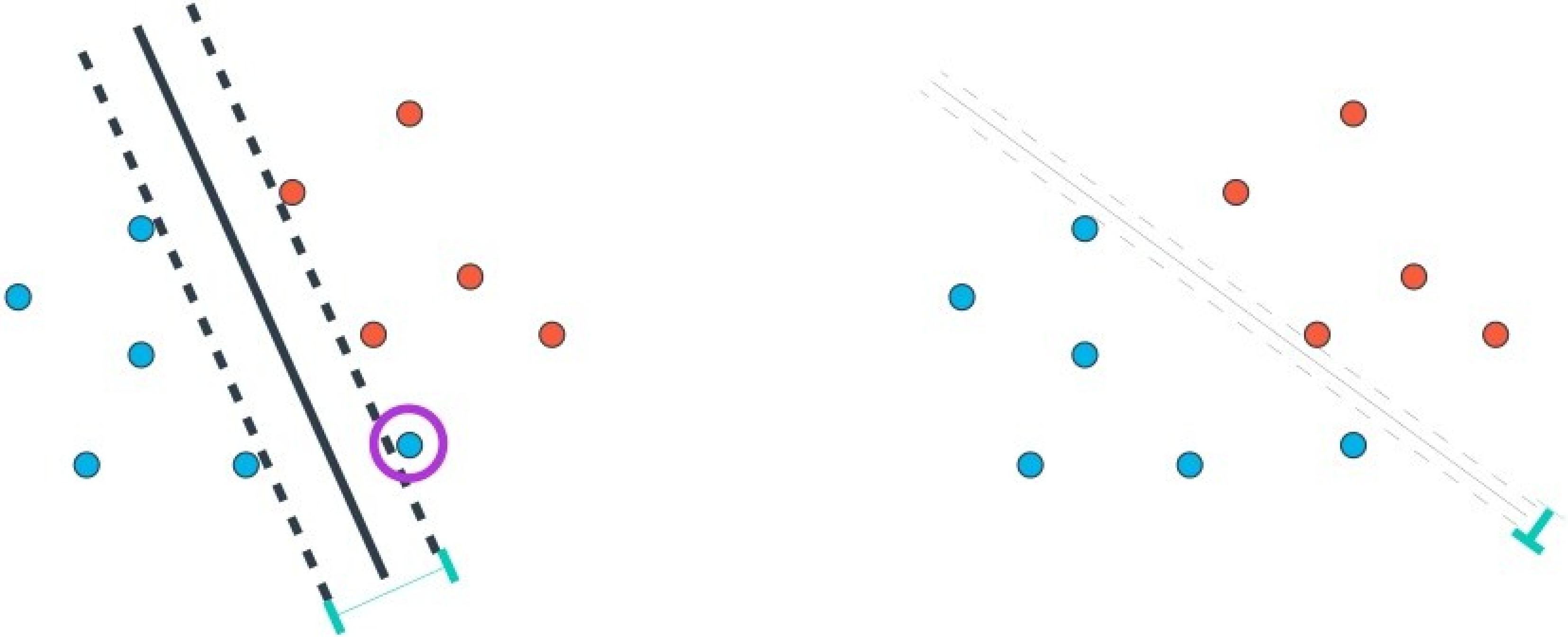
Step 5: (repeat **1000** times)

- Pick random point **(p,q)**
- If point is correctly classified
    - Do nothing
- If point is blue, and ap+bq+c > 0
    - Subtract 0.01**p** to a
    - Subtract 0.01**q** to b
    - Subtract 0.01 to c
- If point is, red and ap+bq+c < 0
    - Add 0.01**p** to a
    - Add 0.01**q** to b
    - Add 0.01 to c
- **Multiply a, b, c, by 0.99**

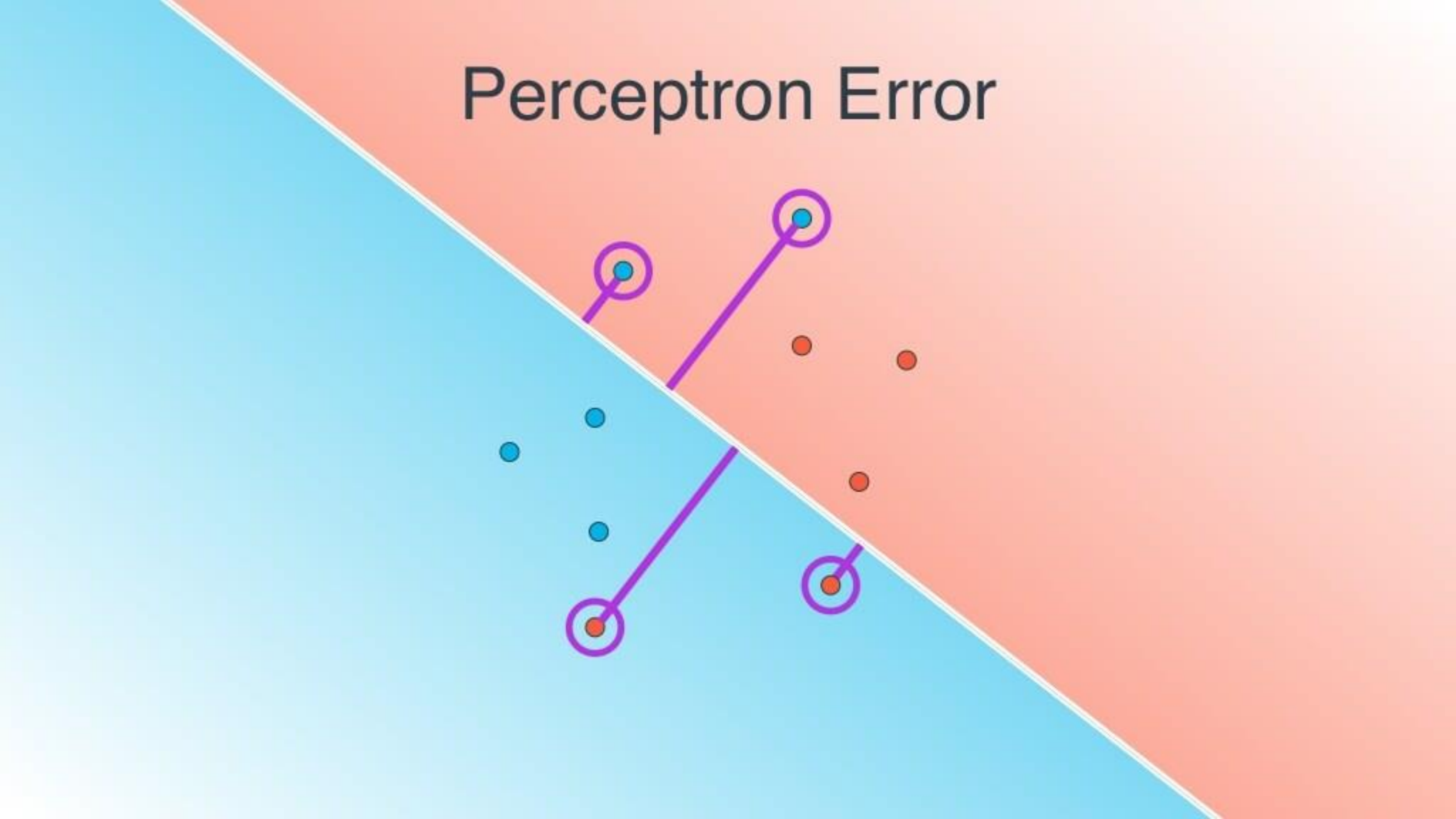
6. Error functions HS

7. Error function(s)

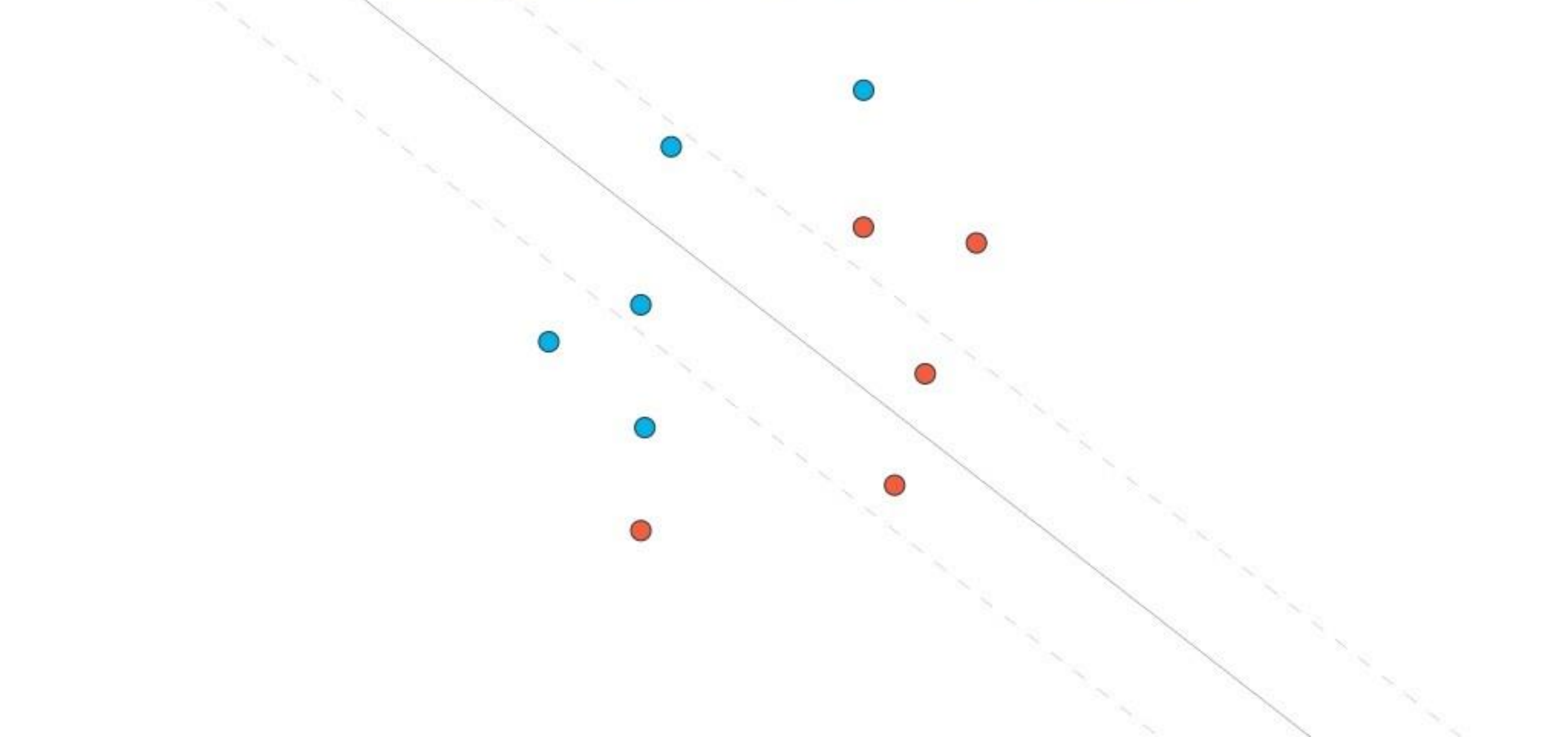
Which line is better?



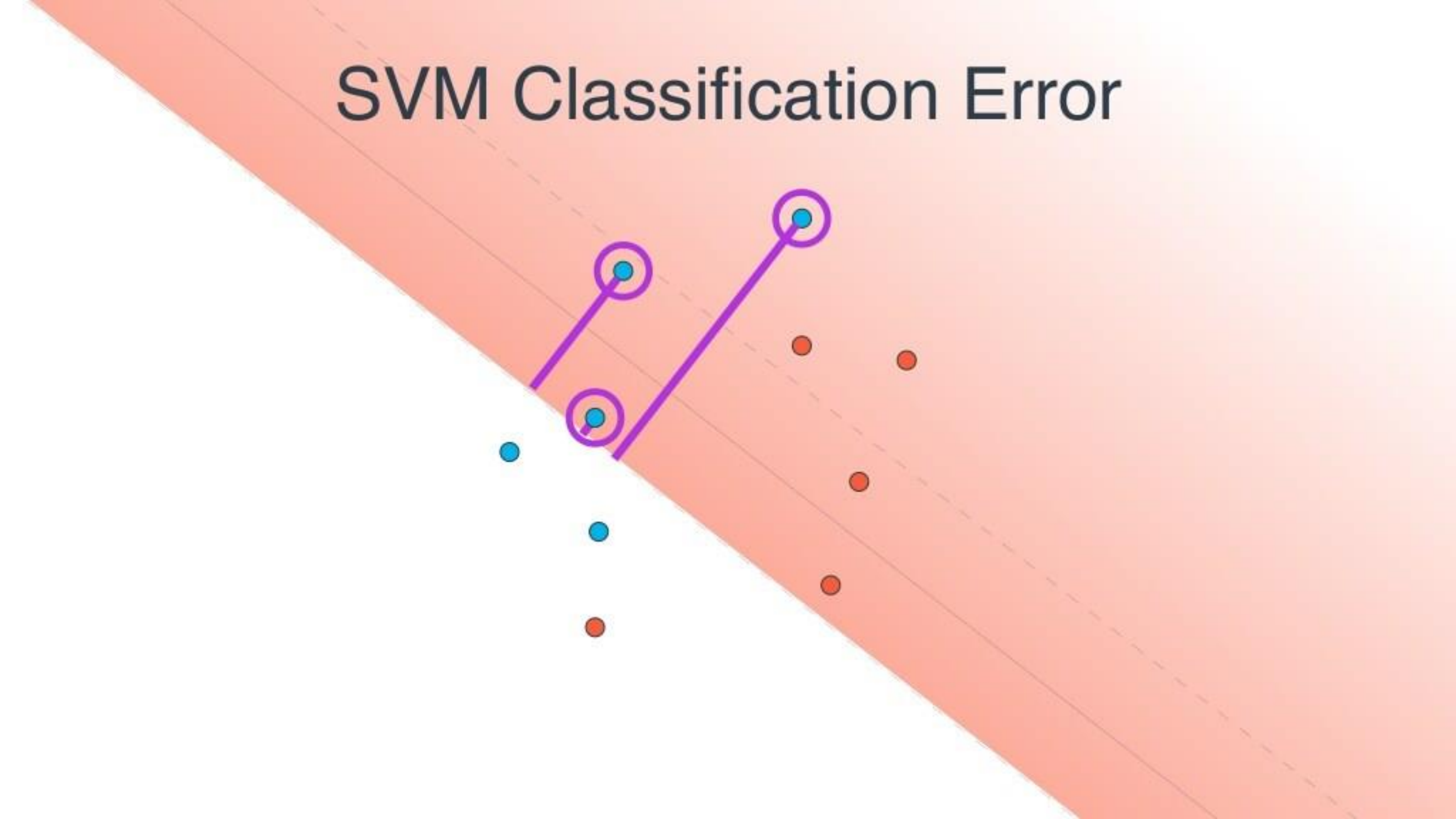
Perceptron Error



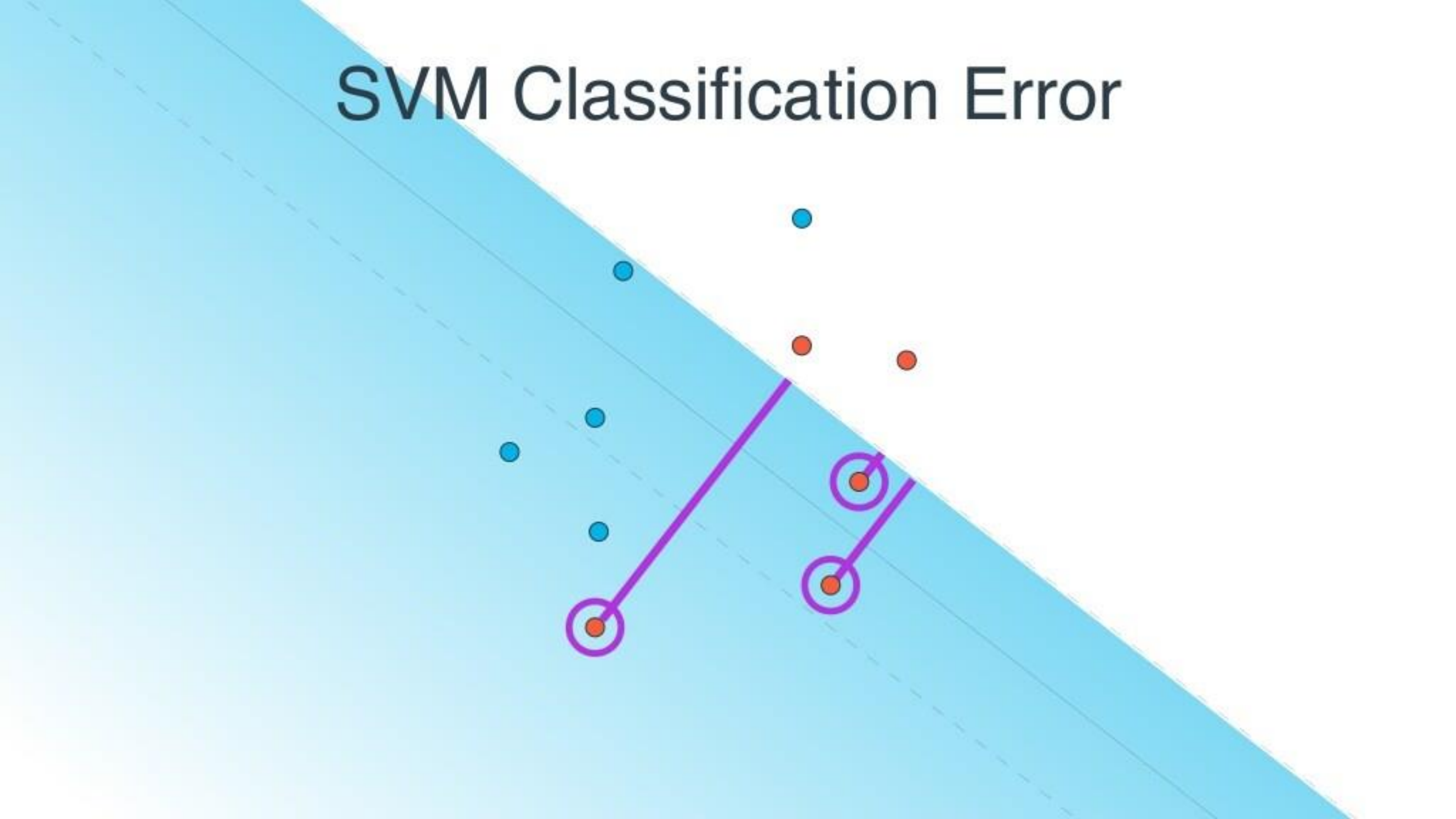
SVM Classification Error



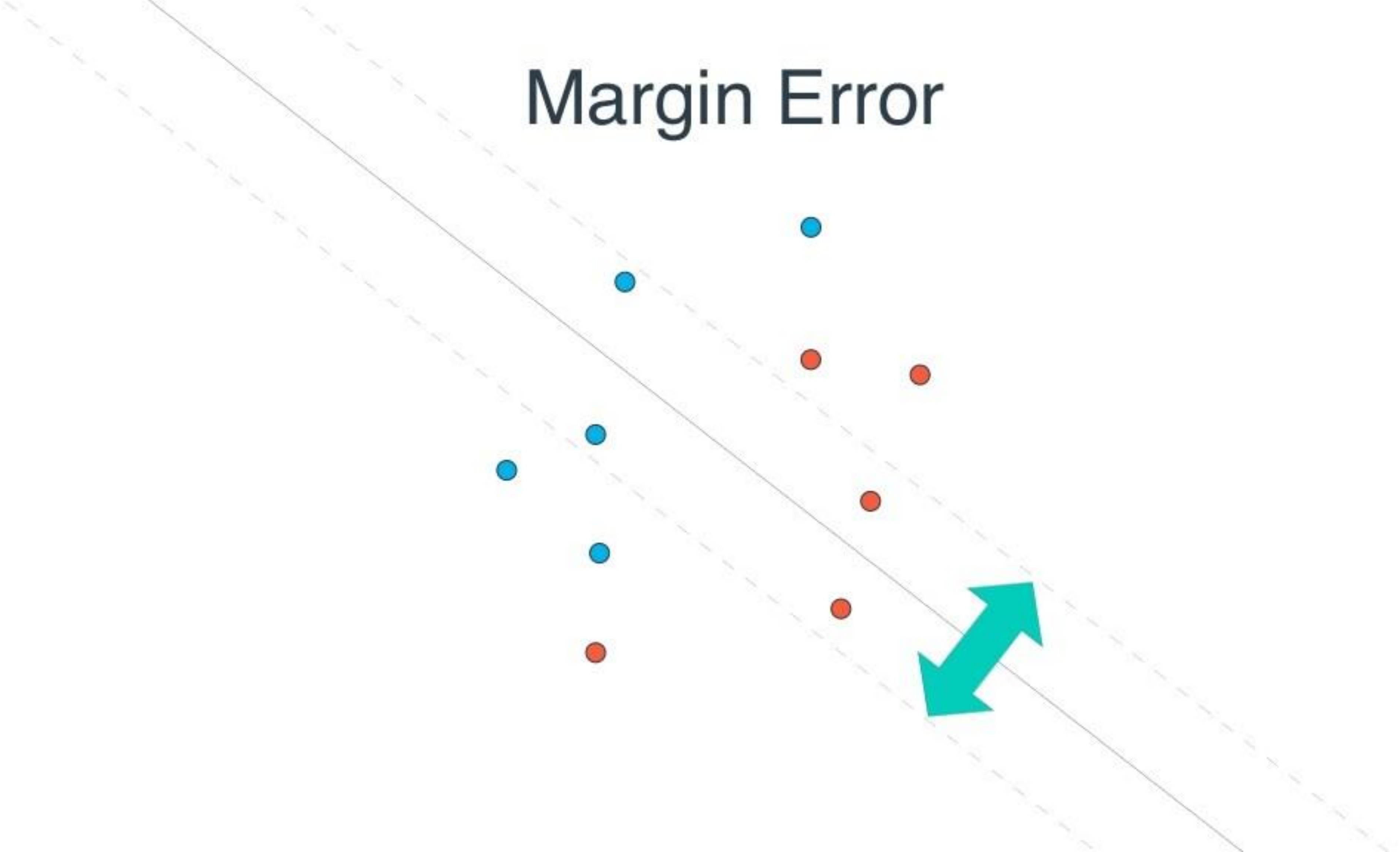
SVM Classification Error



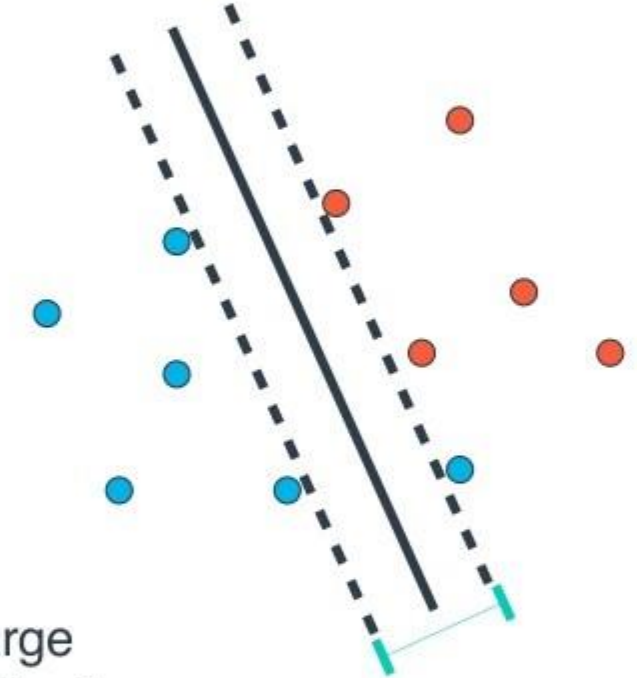
SVM Classification Error



Margin Error

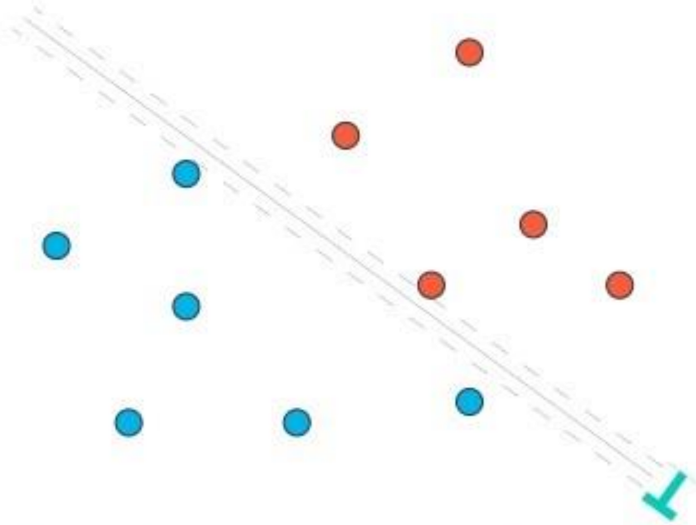


Margin Error



Large classification error

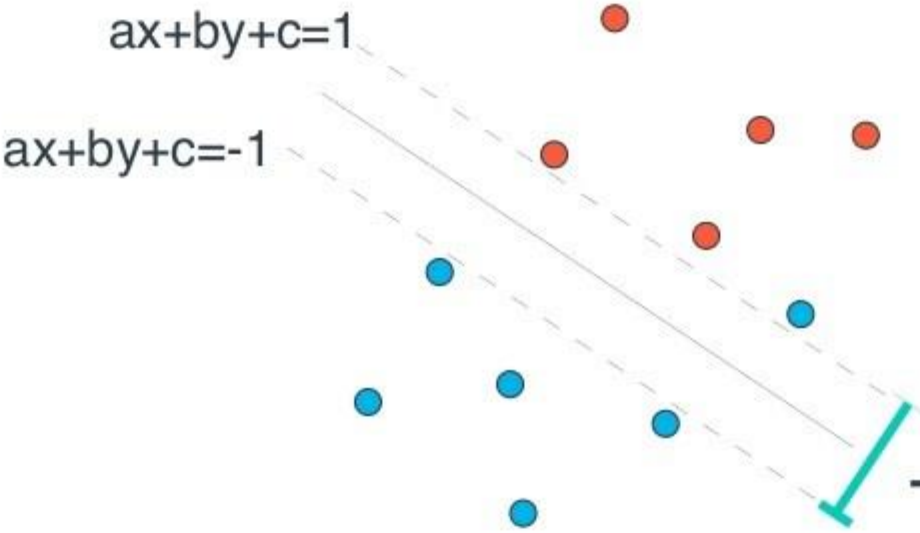
Large margin
Small margin error



Small classification error

Small margin
Large margin error

Margin Error



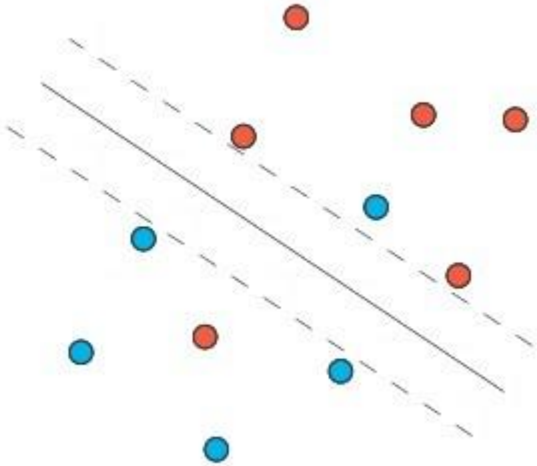
$$\frac{2}{\sqrt{a^2 + b^2}}$$

Challenge:

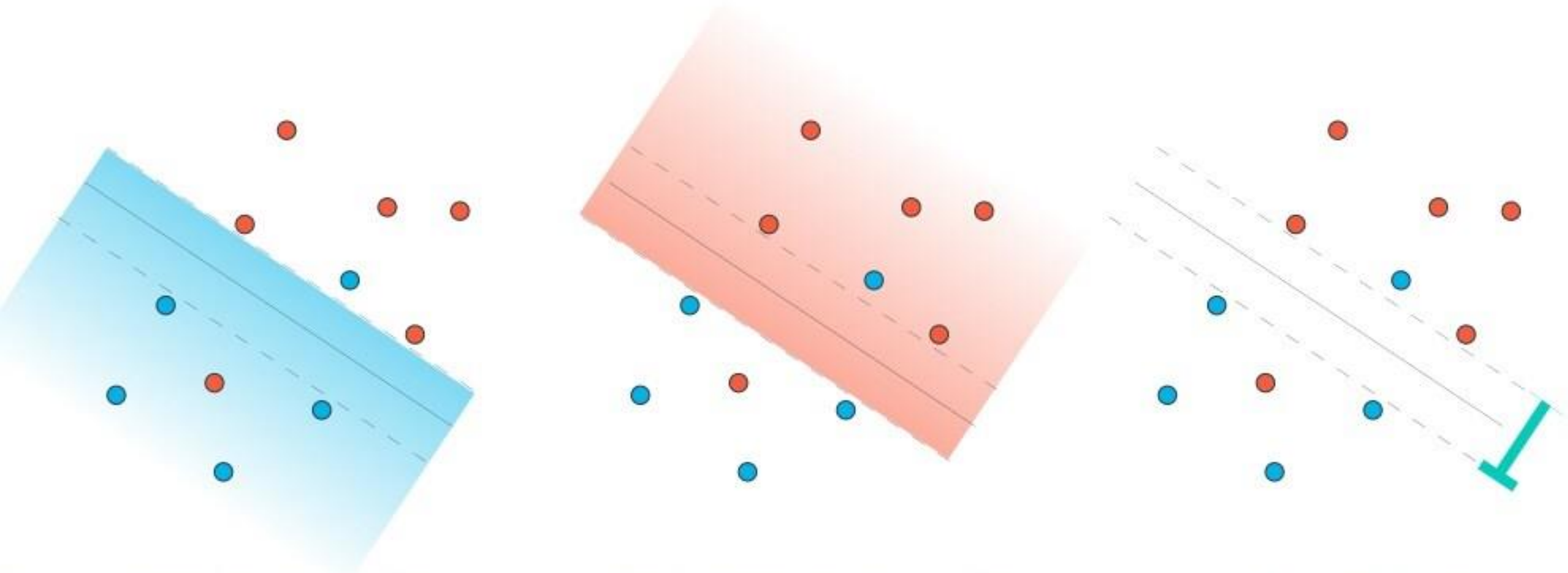
Regularization term!

Margin error = $a^2 + b^2$

SVM Error



SVM Error

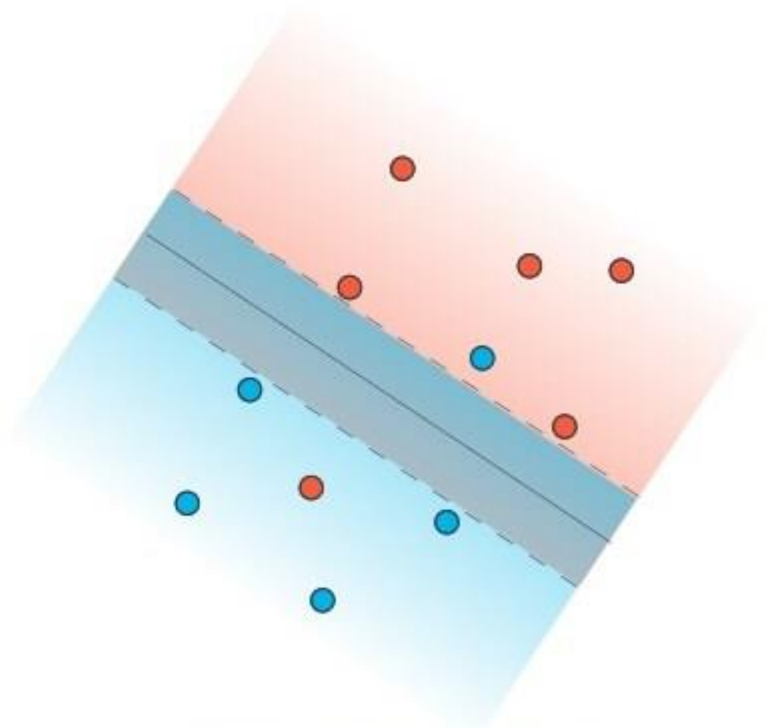


Blue Classification Error

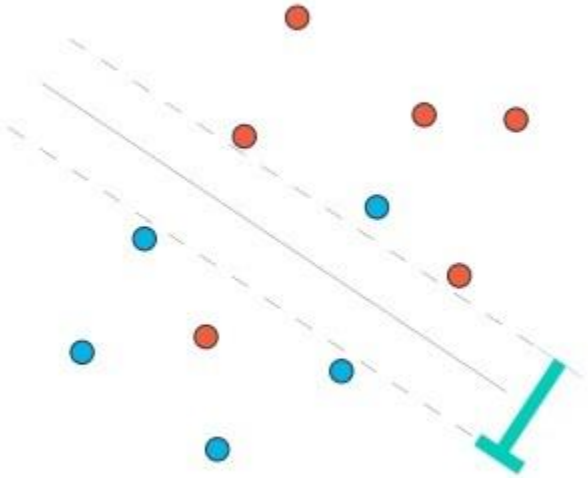
Red Classification Error

Margin Error

SVM Error

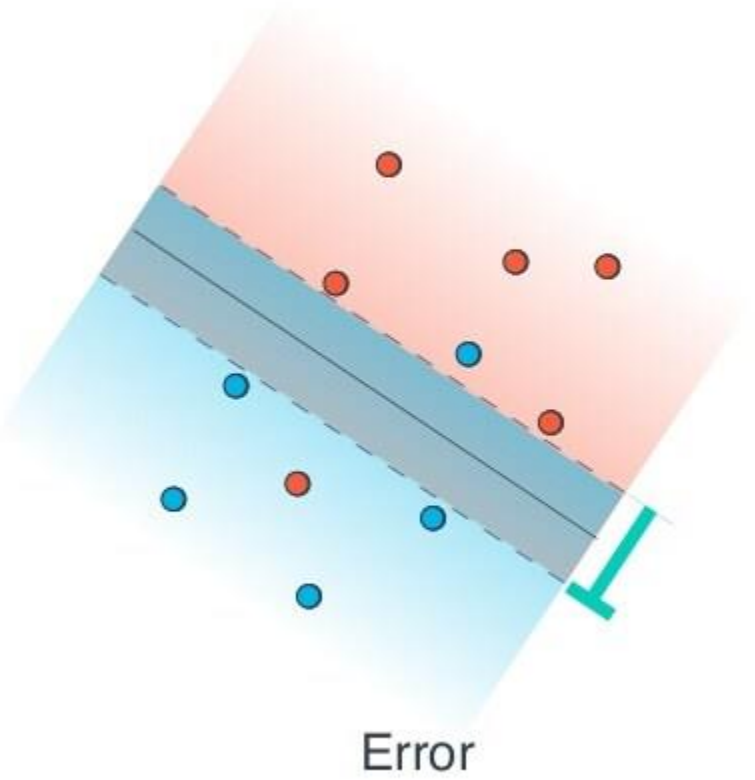


Classification Error



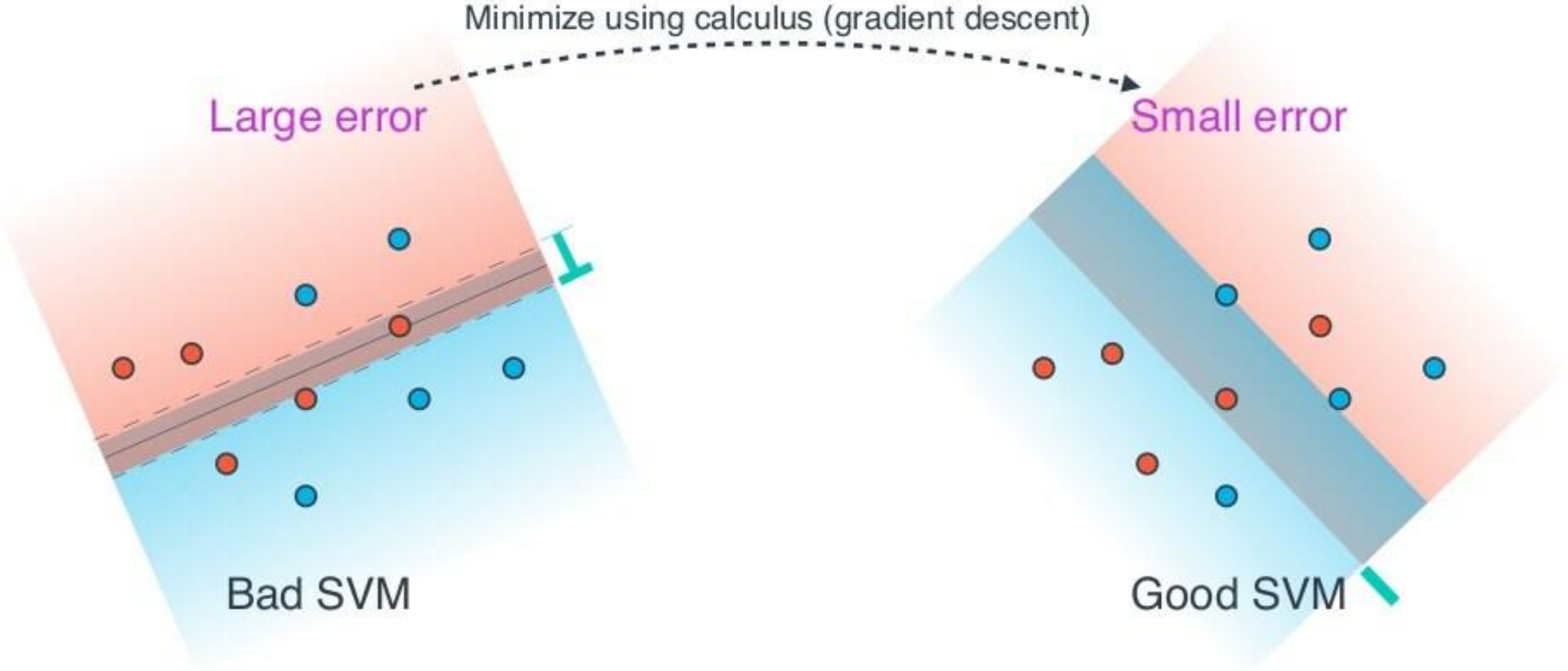
Margin Error

SVM Error

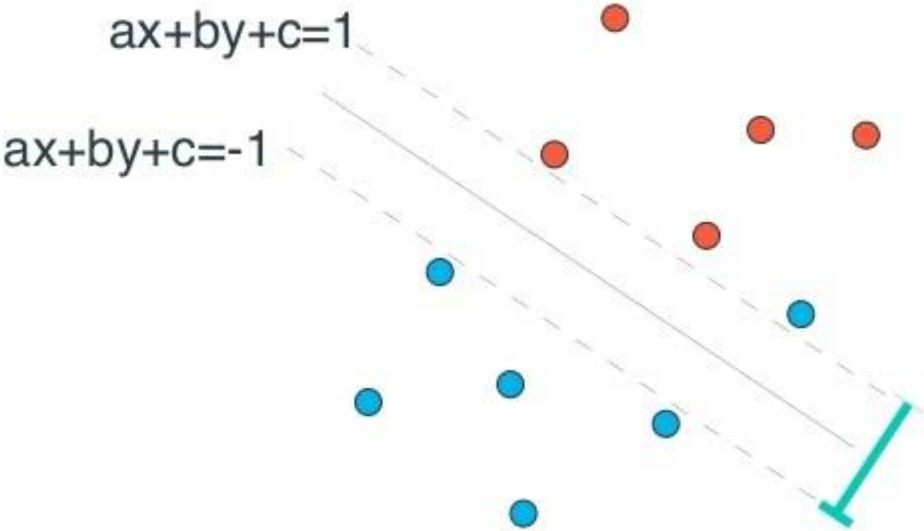


Gradient Descent

Same as the SVM trick!



Challenge - Gradient Descent



Margin error = $a^2 + b^2$

dError/da = 2a

dError/db = 2b

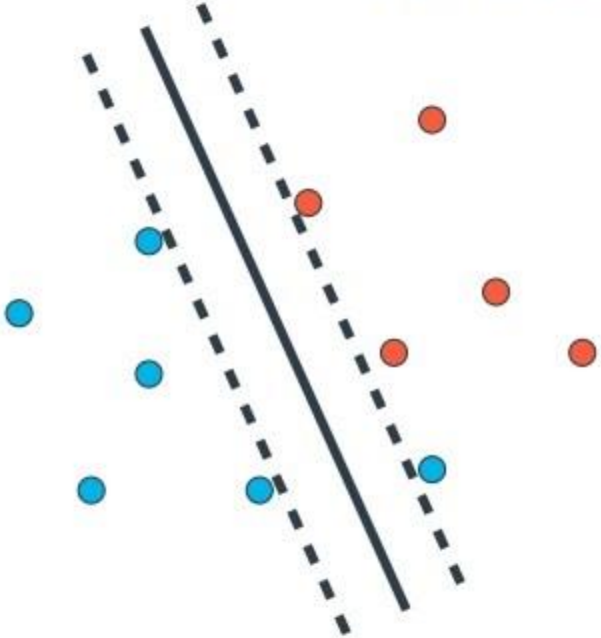
expanding factor!

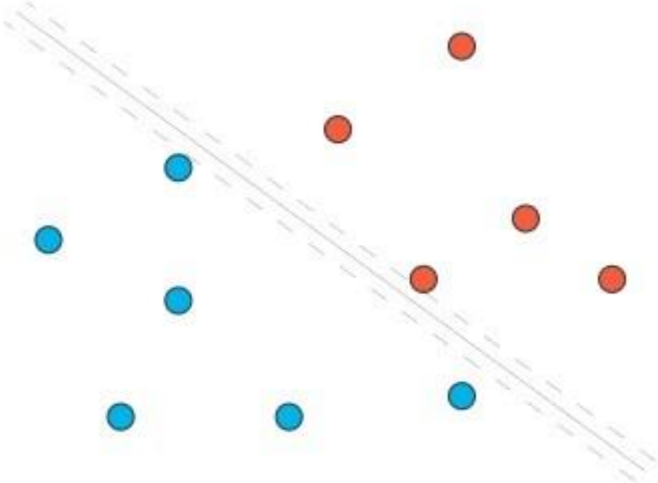
$$a \longrightarrow a - \eta\, 2a = a\boxed{(1-2\eta)}$$

$$b \longrightarrow b - \eta\, 2b = b\boxed{(1-2\eta)}$$

8. The C Parameter

Which line is better?

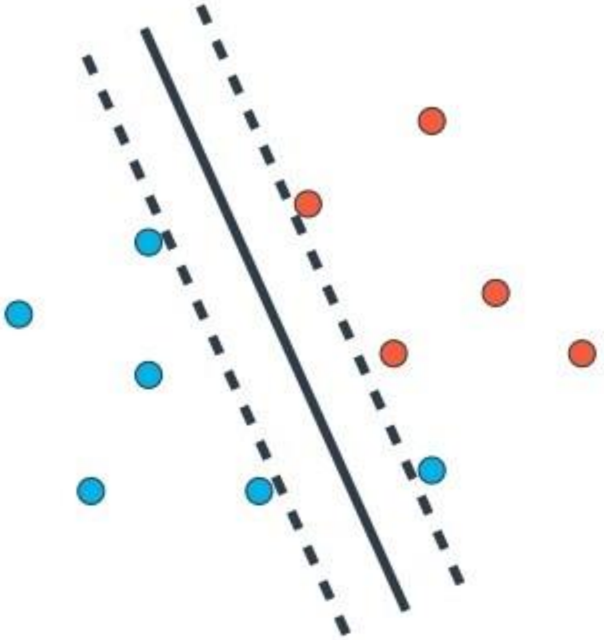




Classification Error + Margin Error

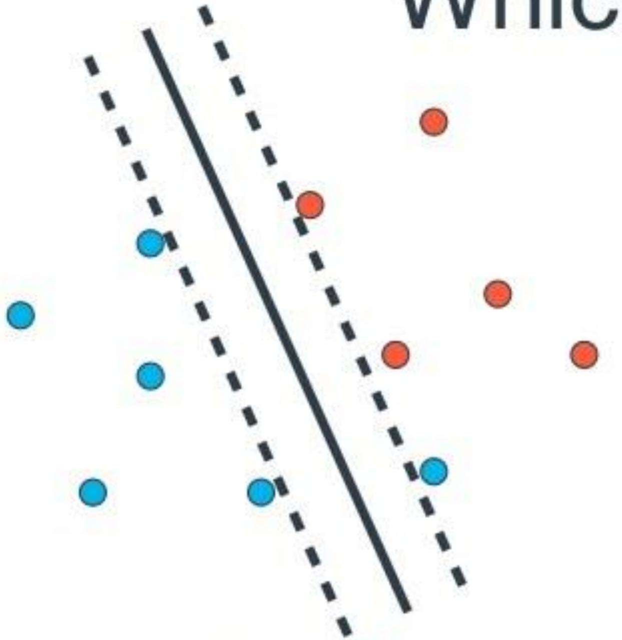
Classification Error + Margin Error

The C parameter

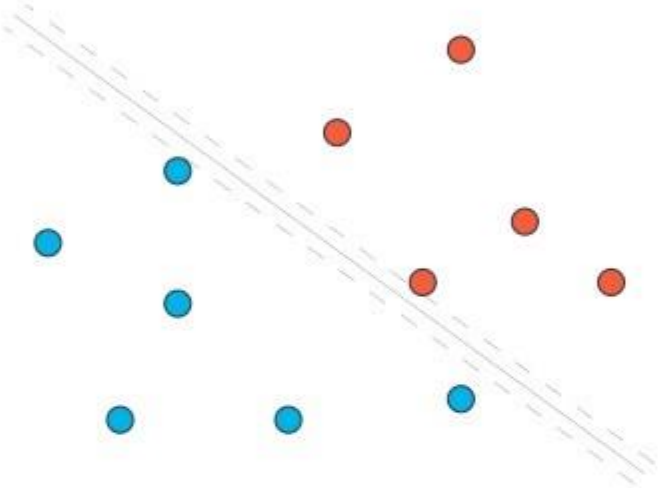


C Classification Error + Margin Error

Which line is better?



Small C
Focus on margin



Large C
Focus on classification

C ~~Classification Error~~ + ~~Margin Error~~