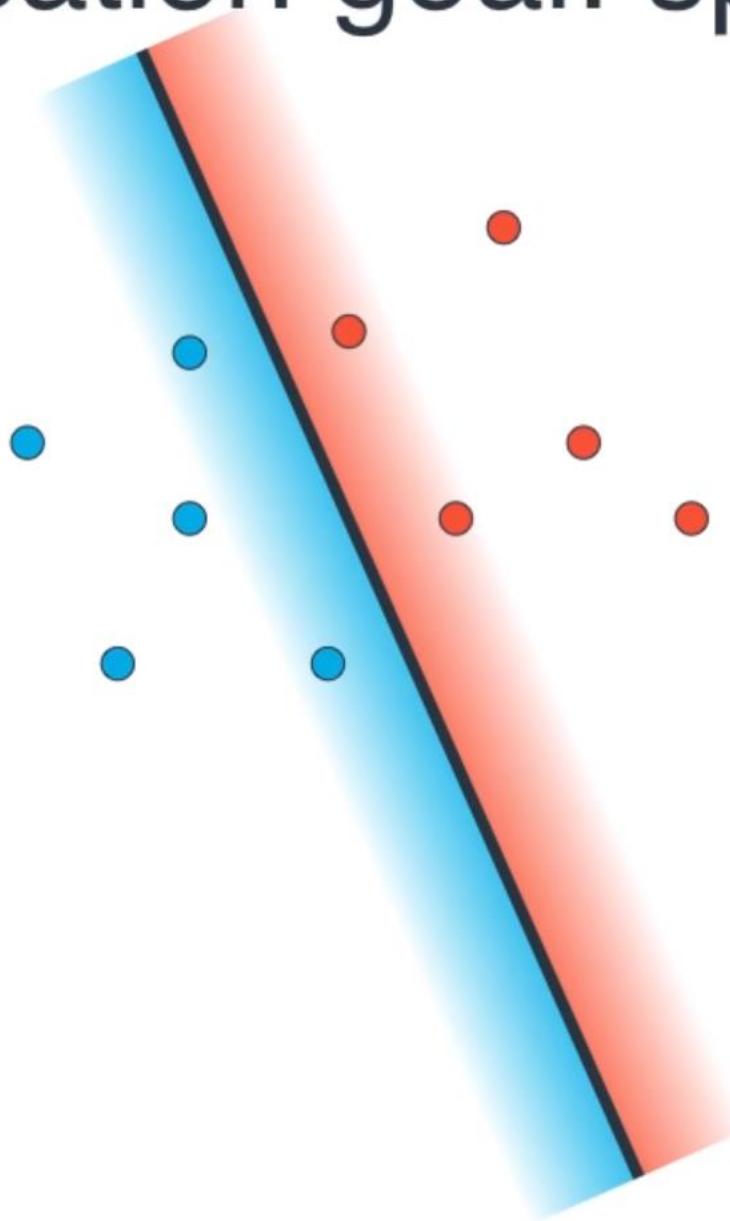


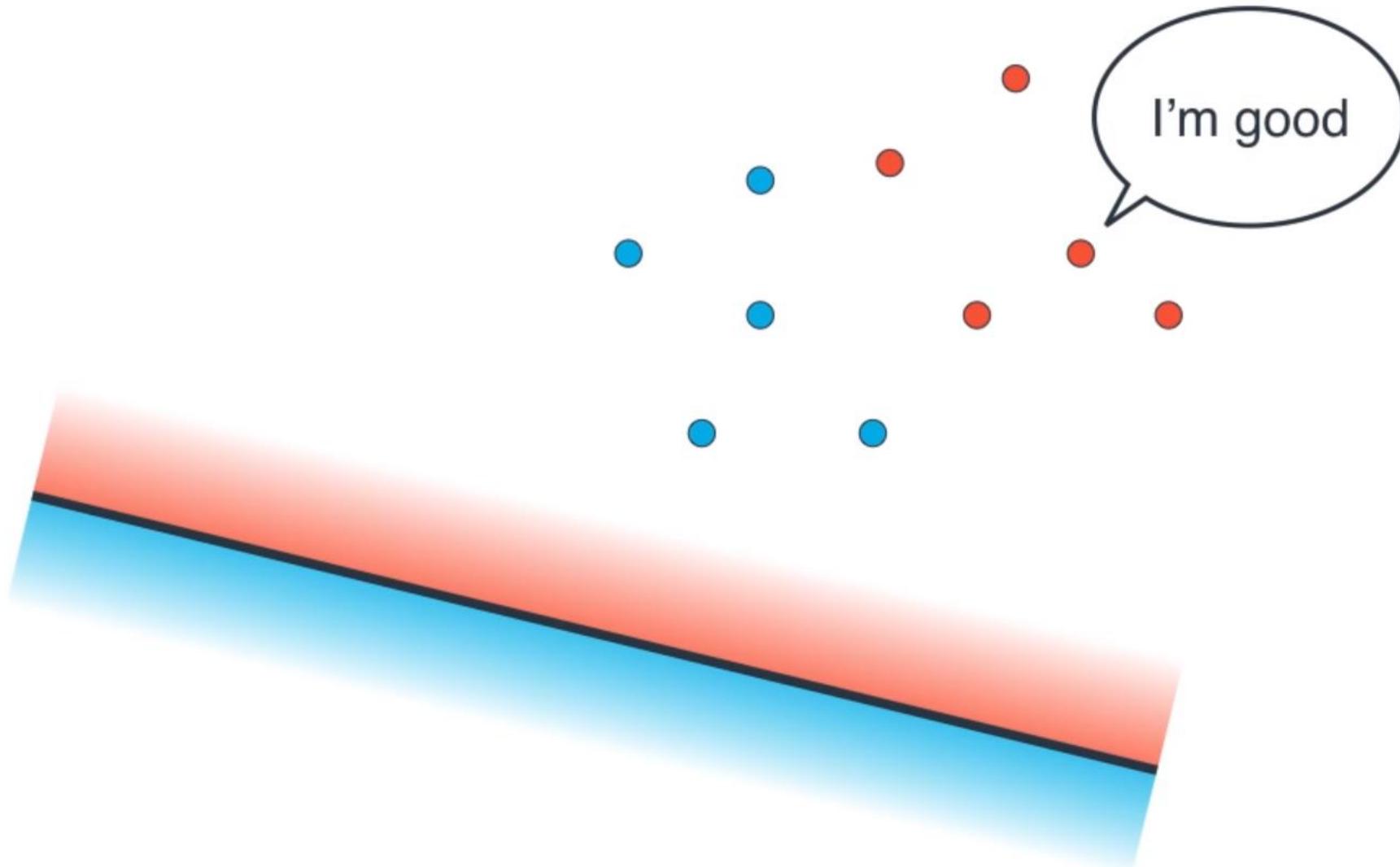
Support Vector Machine (SVM)

- Mohamed Imran

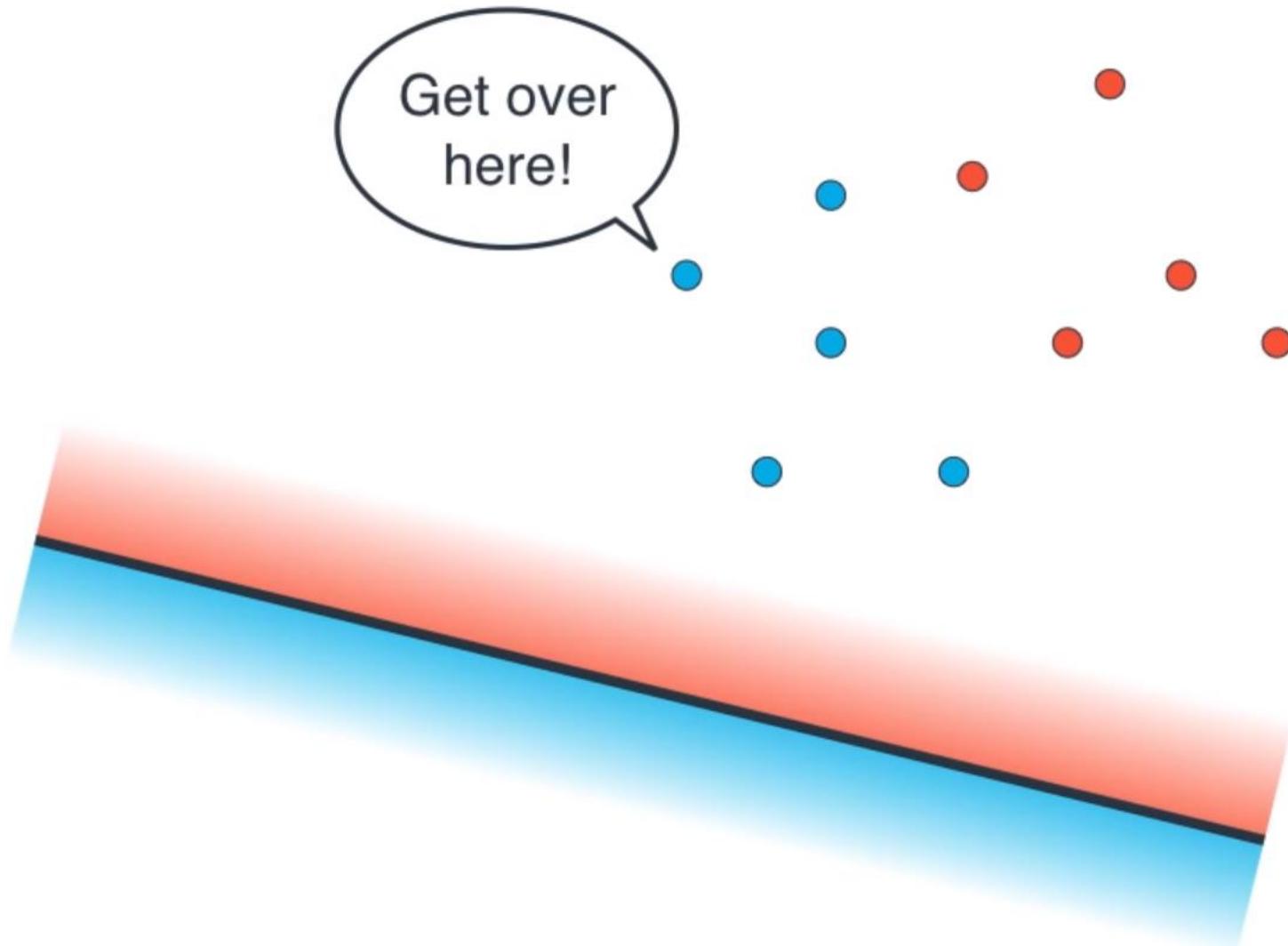
Classification goal: split data



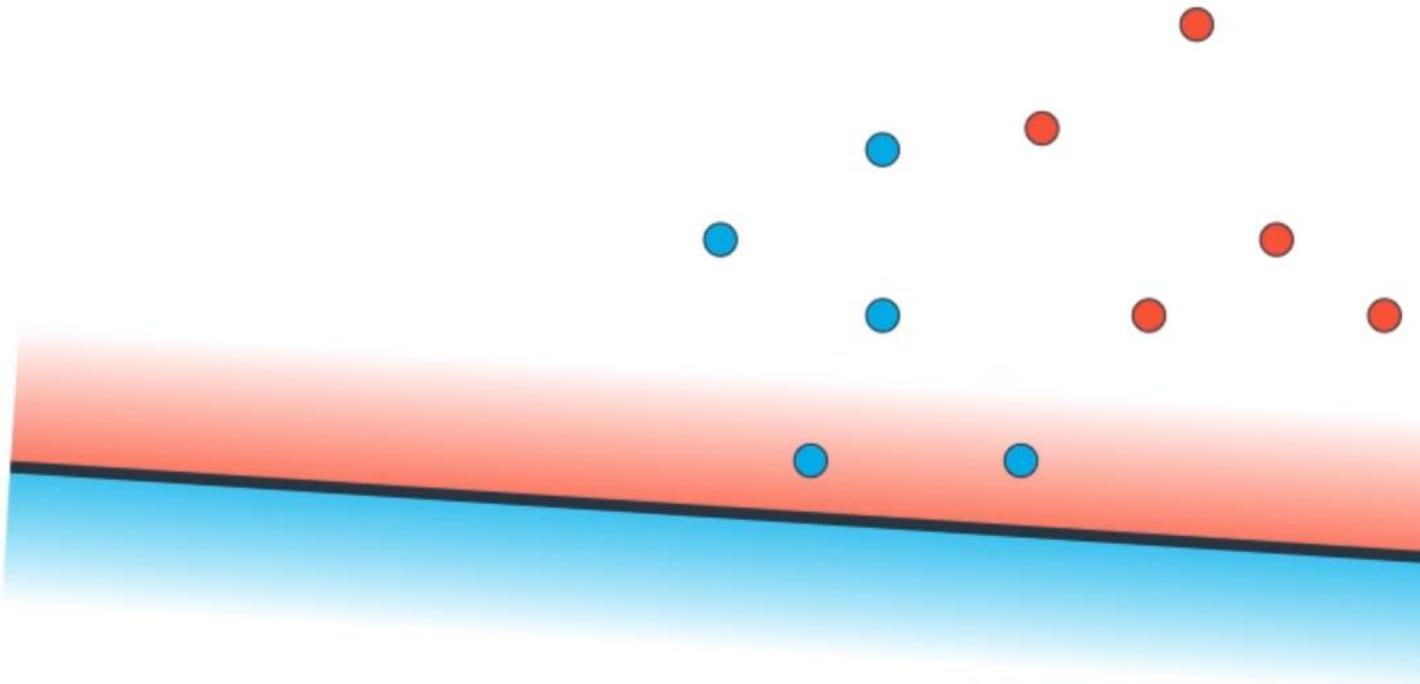
Perceptron Algorithm



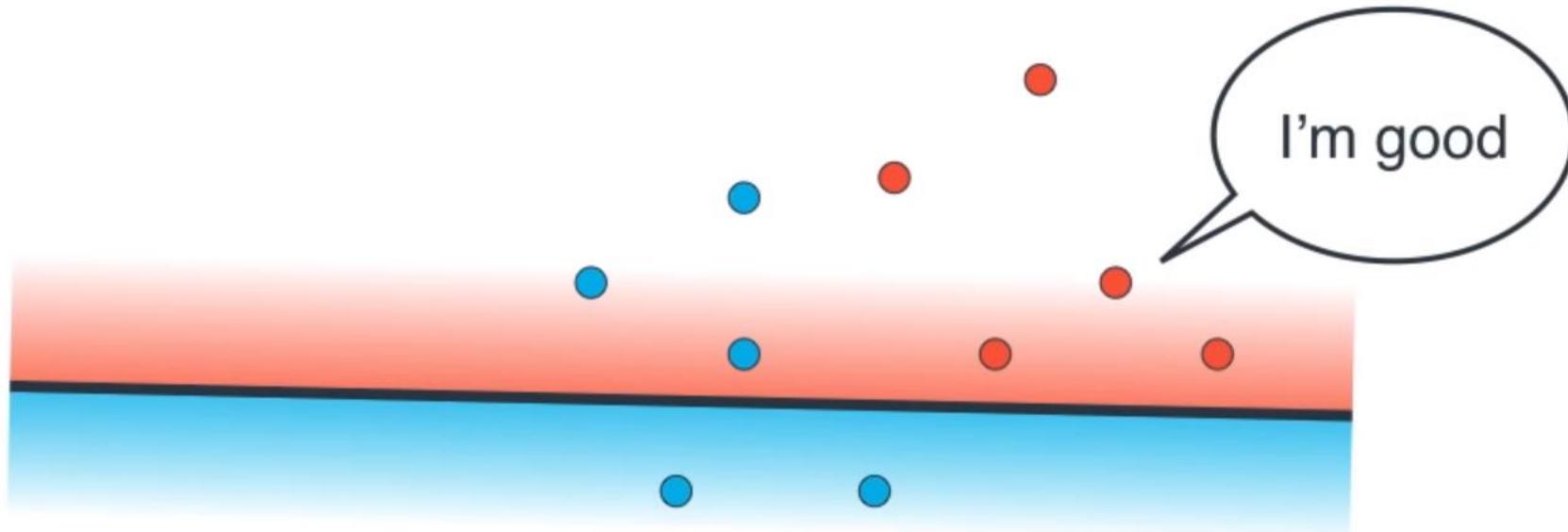
Perceptron Algorithm



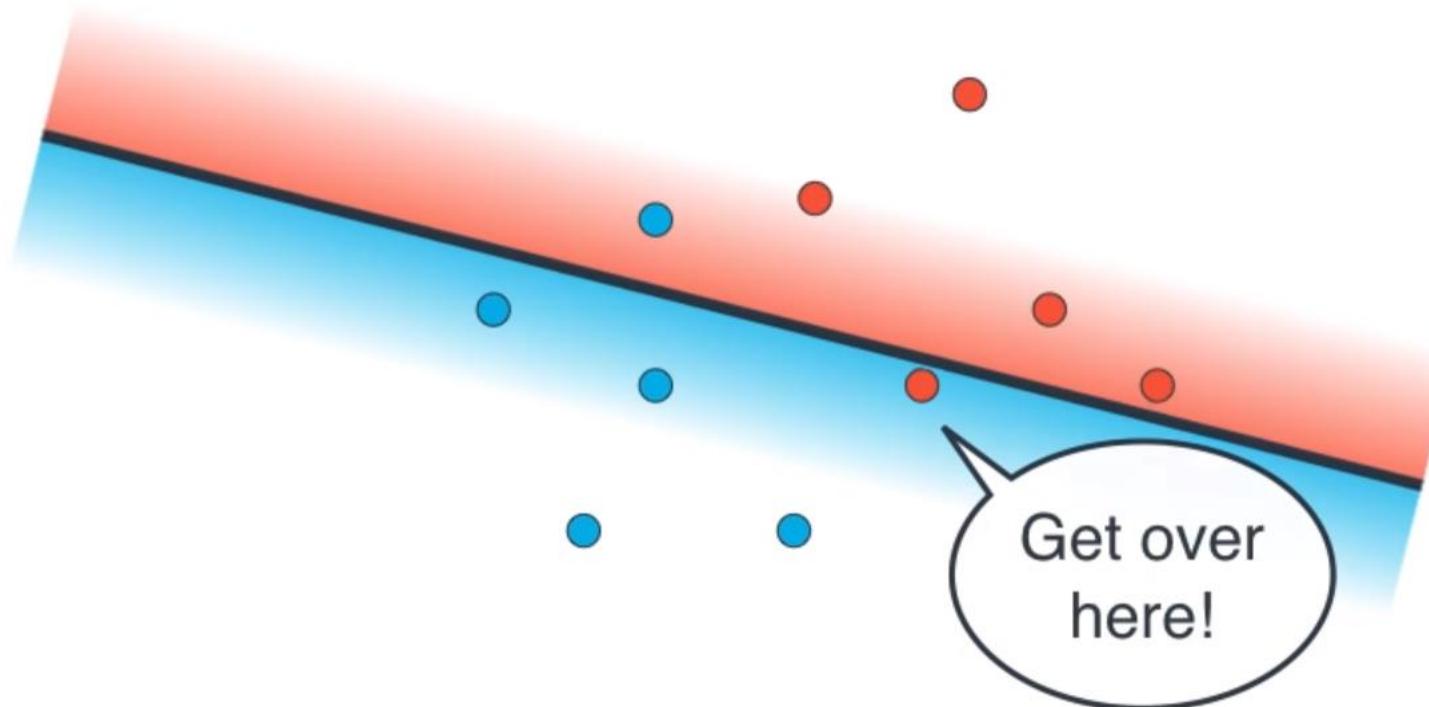
Perceptron Algorithm



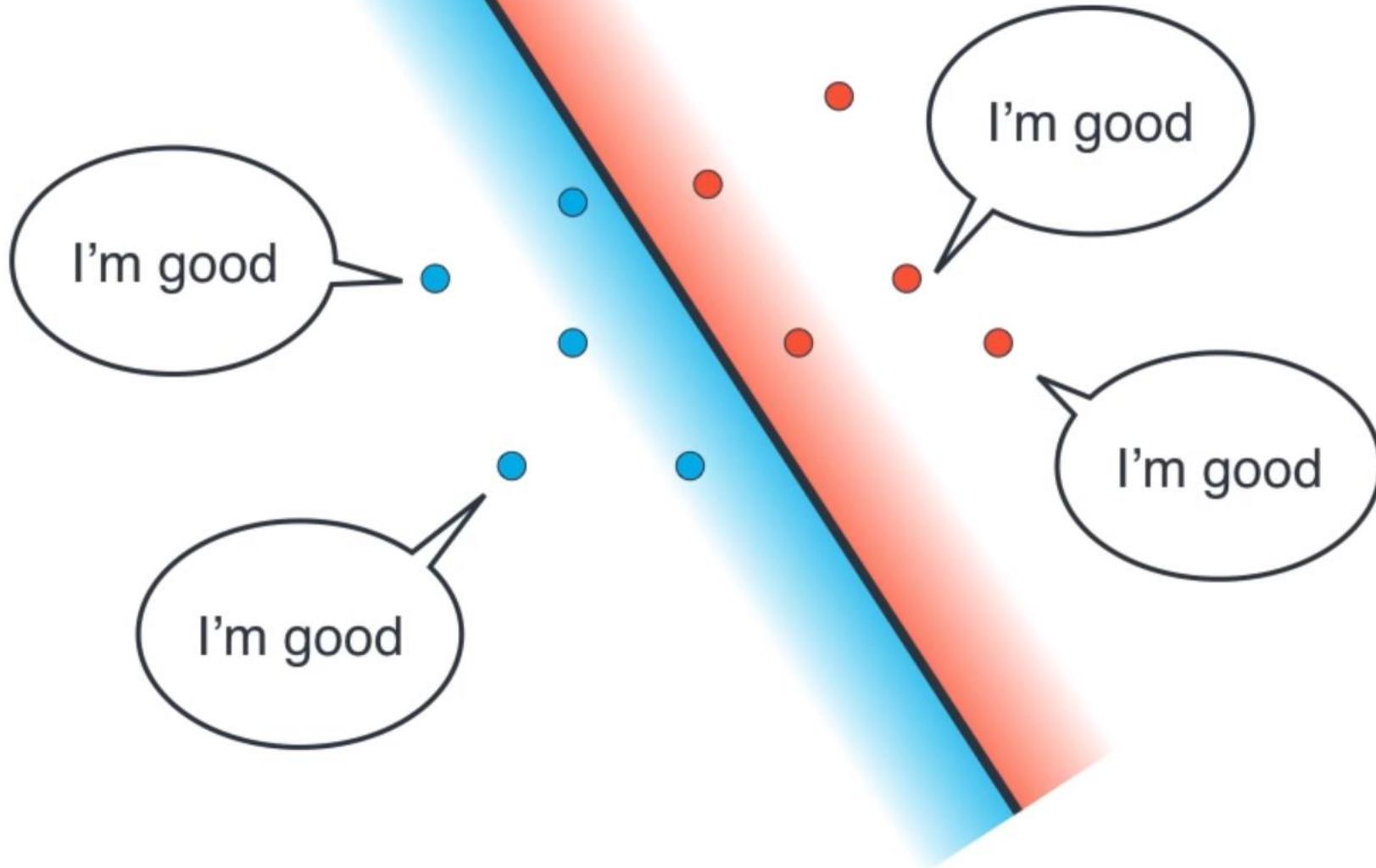
Perceptron Algorithm



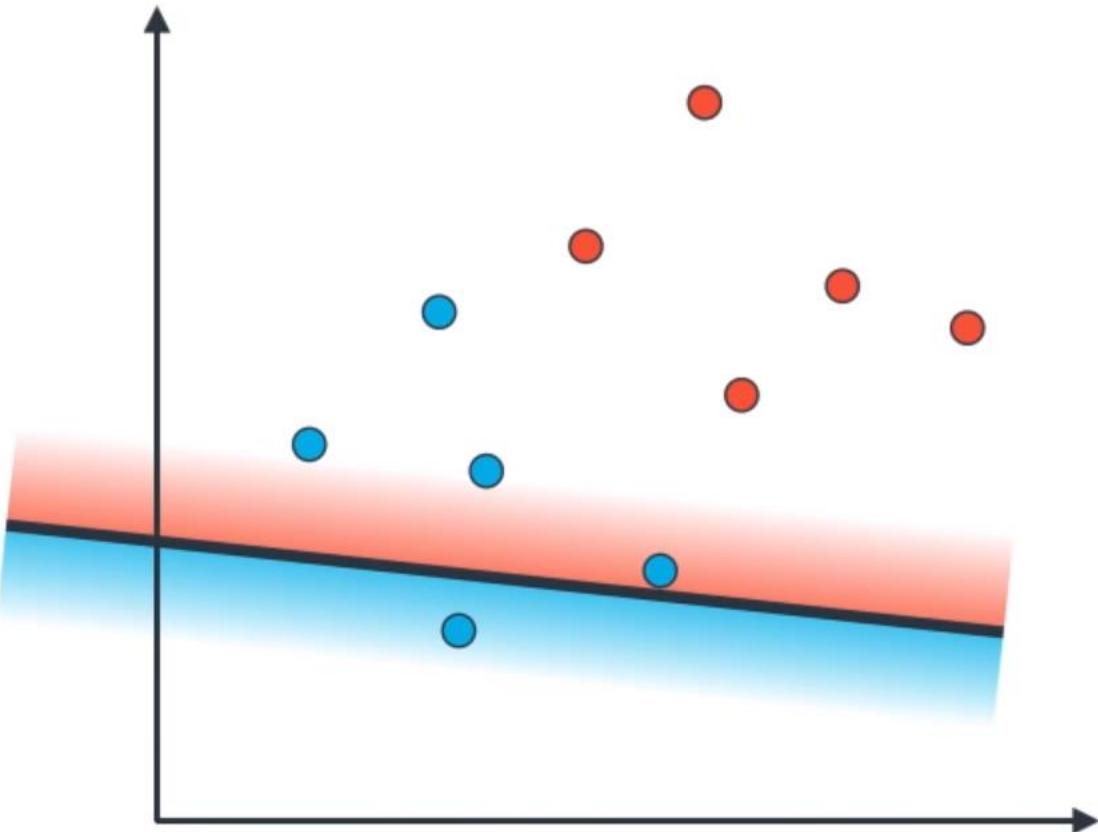
Perceptron Algorithm



Perceptron Algorithm



Perceptron algorithm



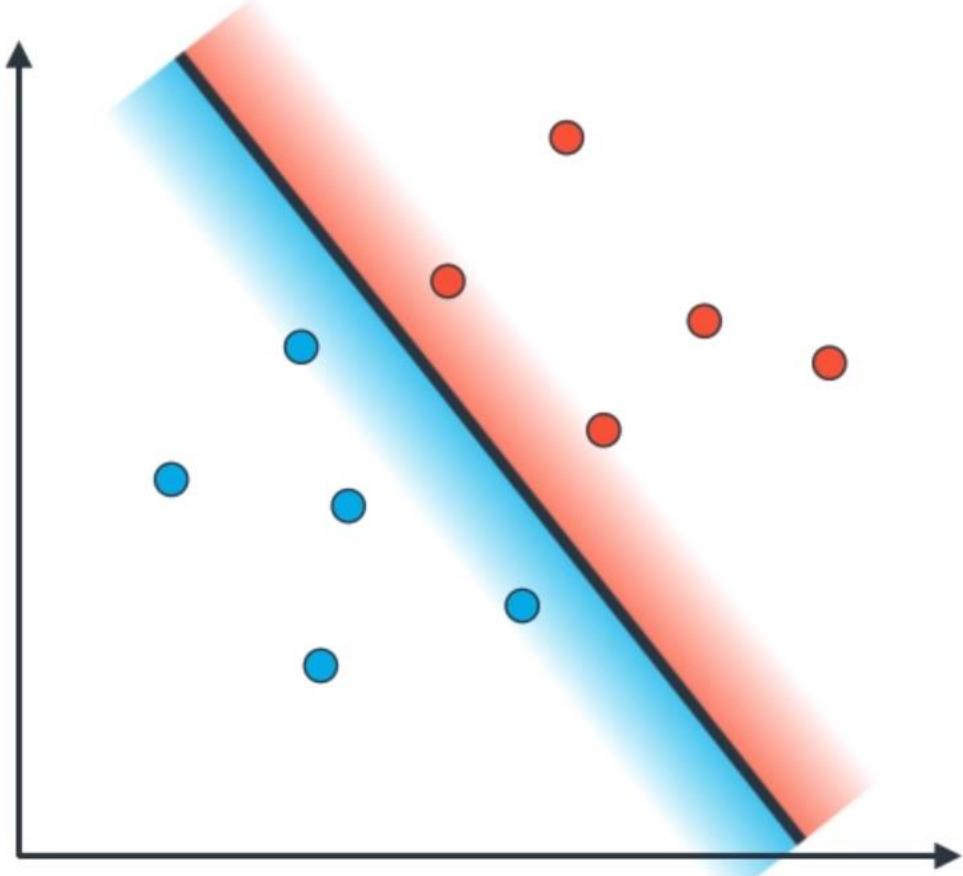
Step 1: Start with a random line with **blue** and **red** sides.

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

Perceptron algorithm



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

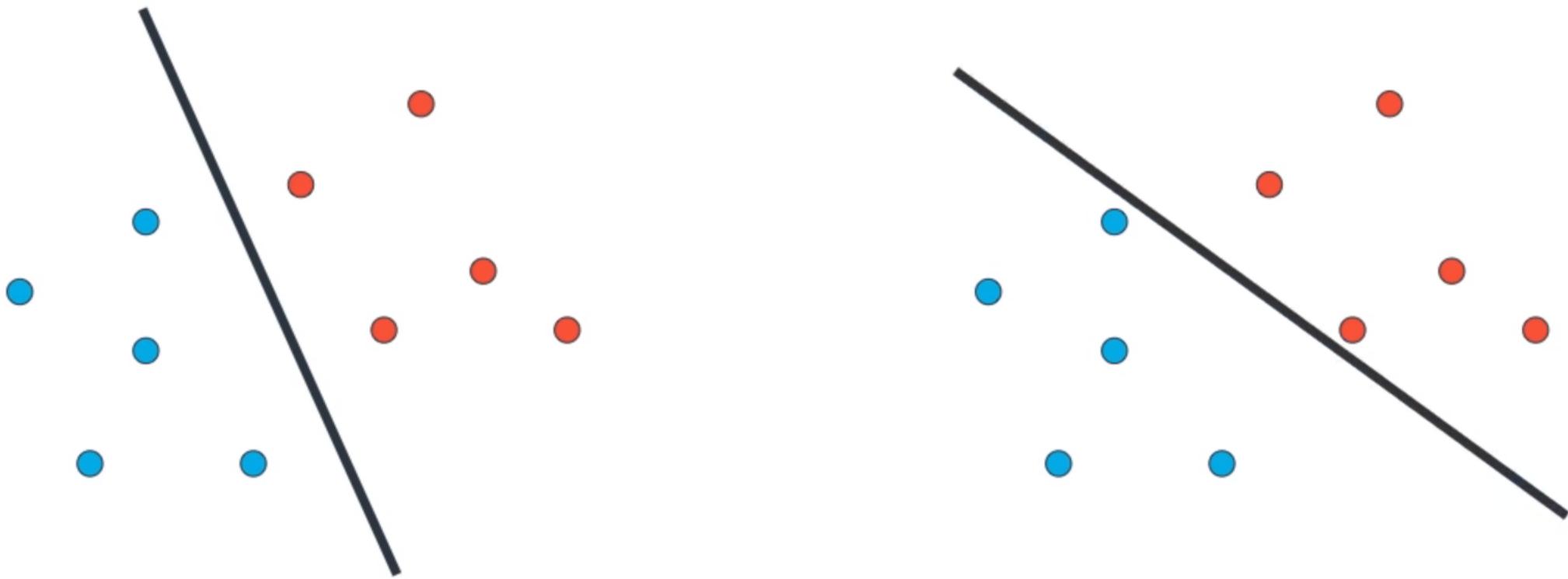
Step 2: Pick a large number. **1000** (number of repetitions, or epochs)

Step 3: (repeat **1000** times)

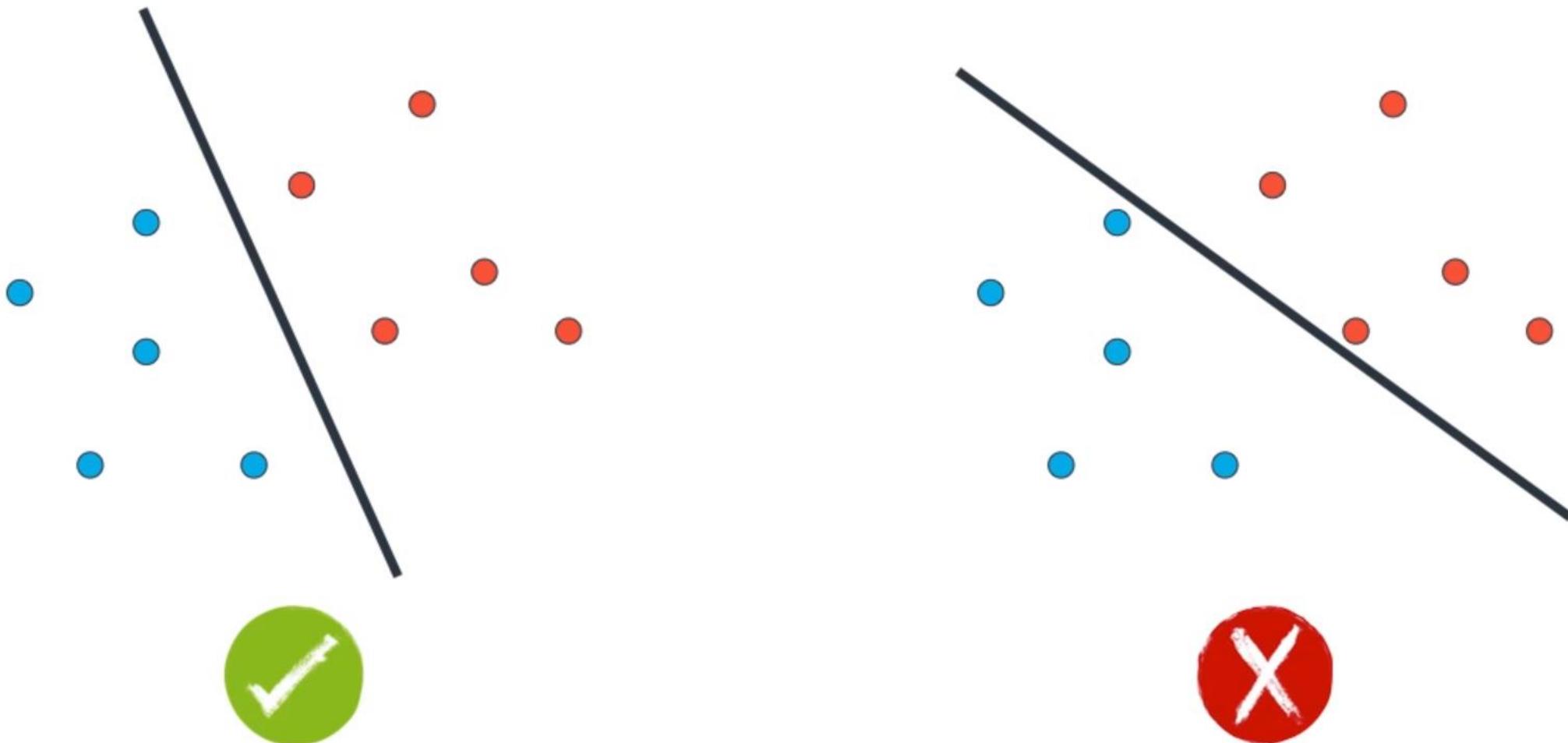
- Pick random point
- If point is correctly classified:
 - Do nothing
- If point is incorrectly classified
 - Move line towards point

Step 4: Enjoy your line that separates the data!

Which line is better?

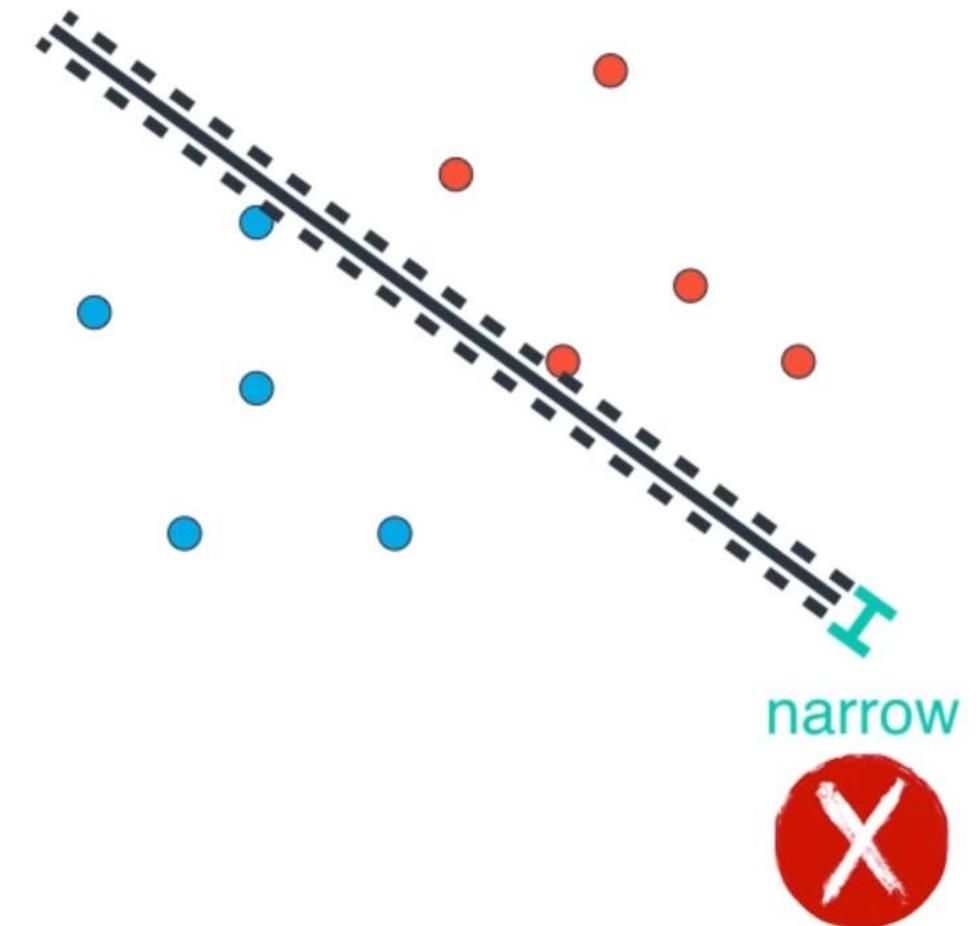
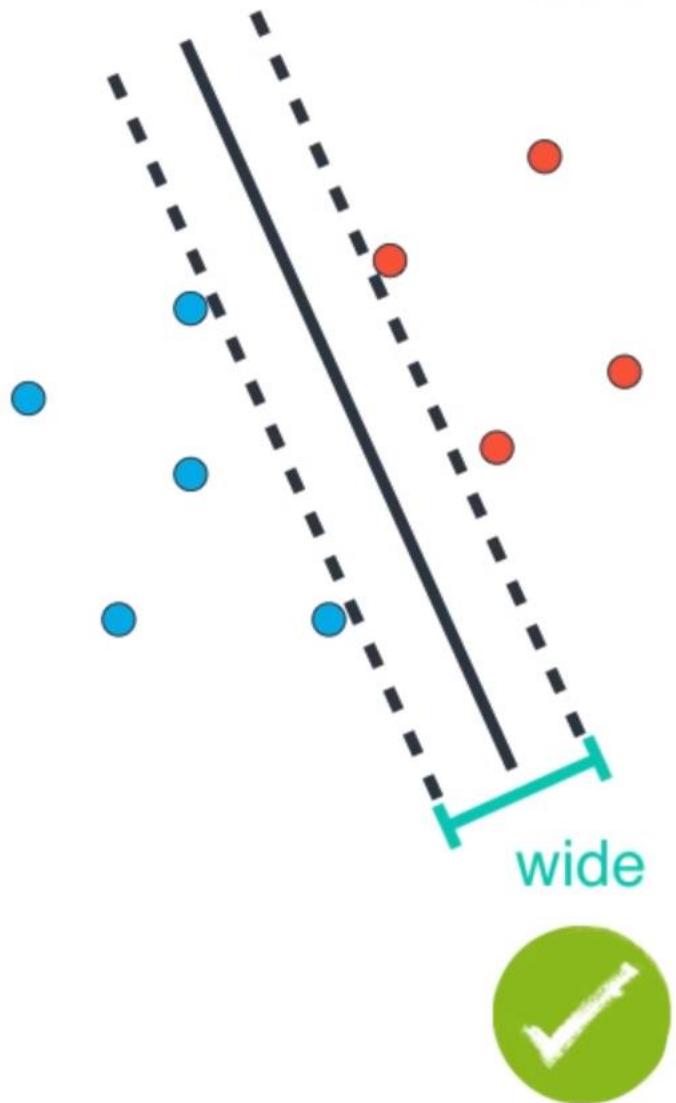


Which line is better?



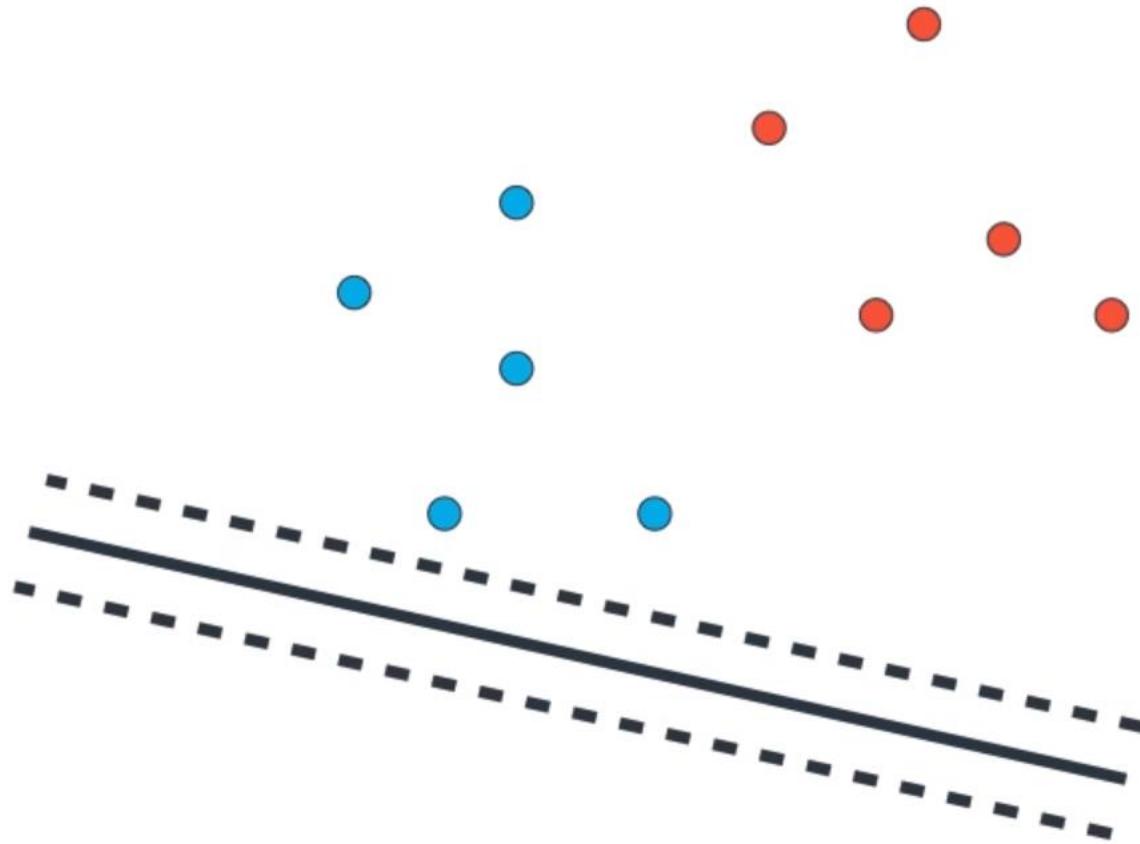
A perceptron algorithm just finds the line which separates the data, not necessary the best line

Which line is better?

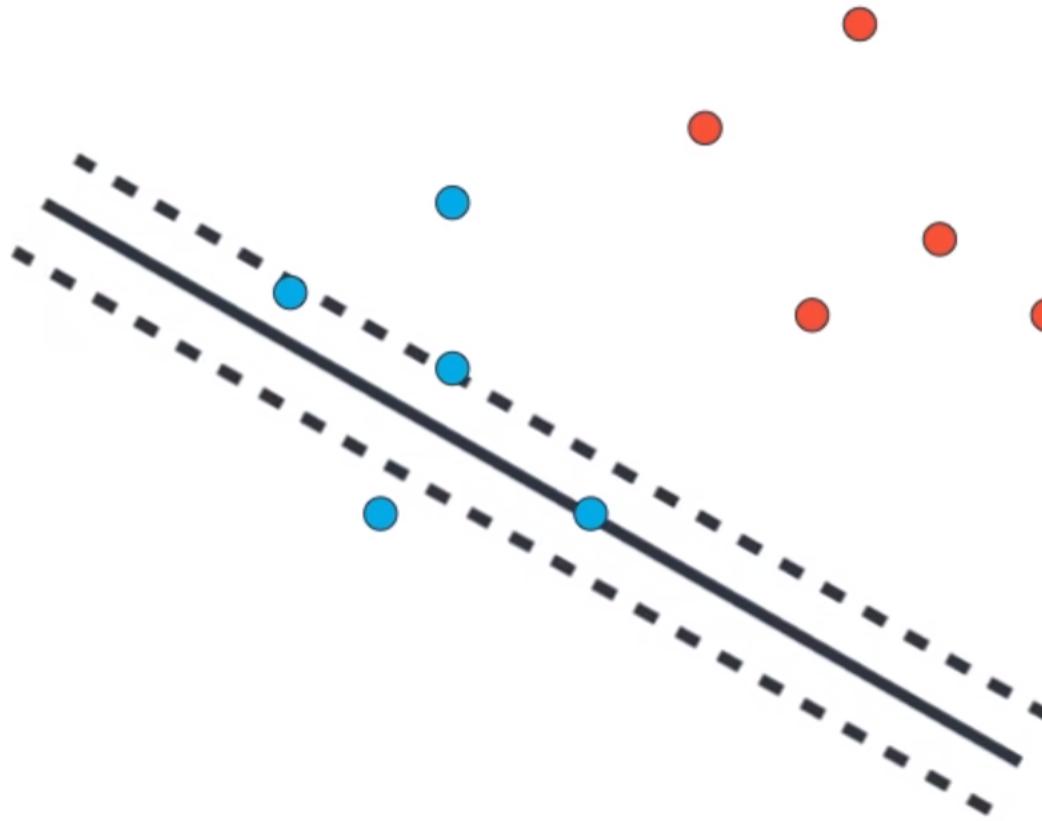


We tell the computer if you are going to find a wide margin, you're good. And if you're going to find narrow margin, you're not good.

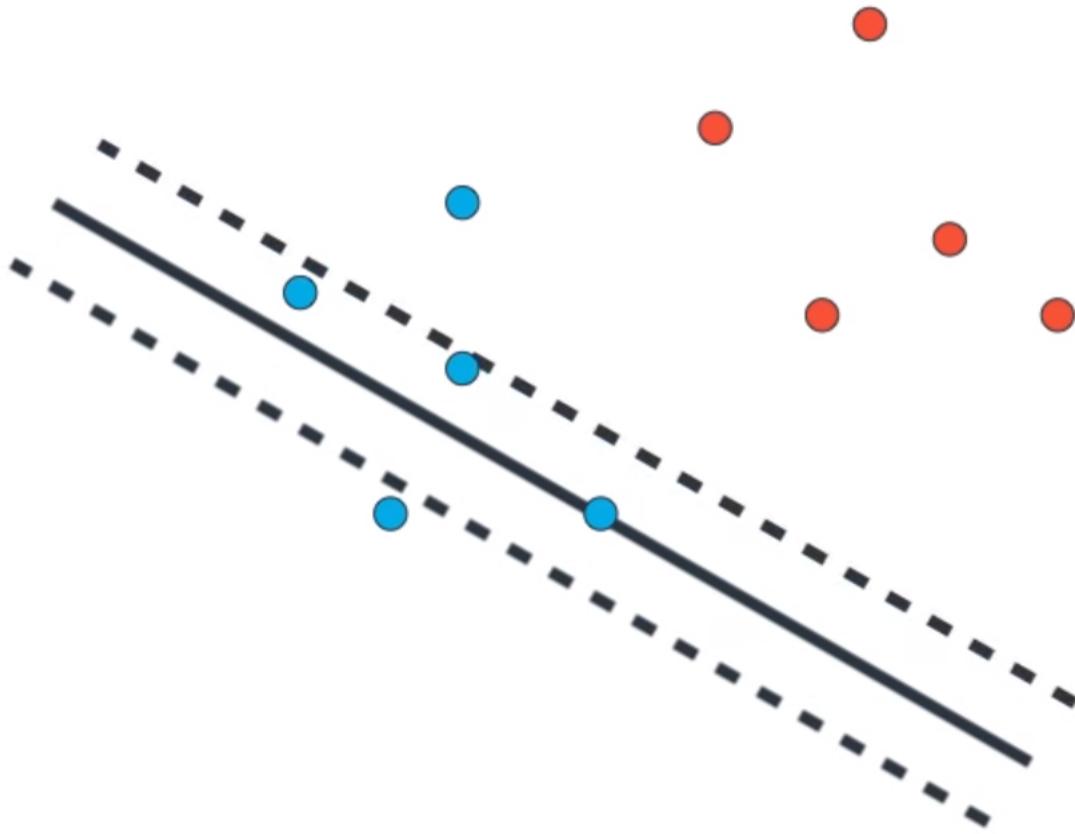
Split data - separate lines



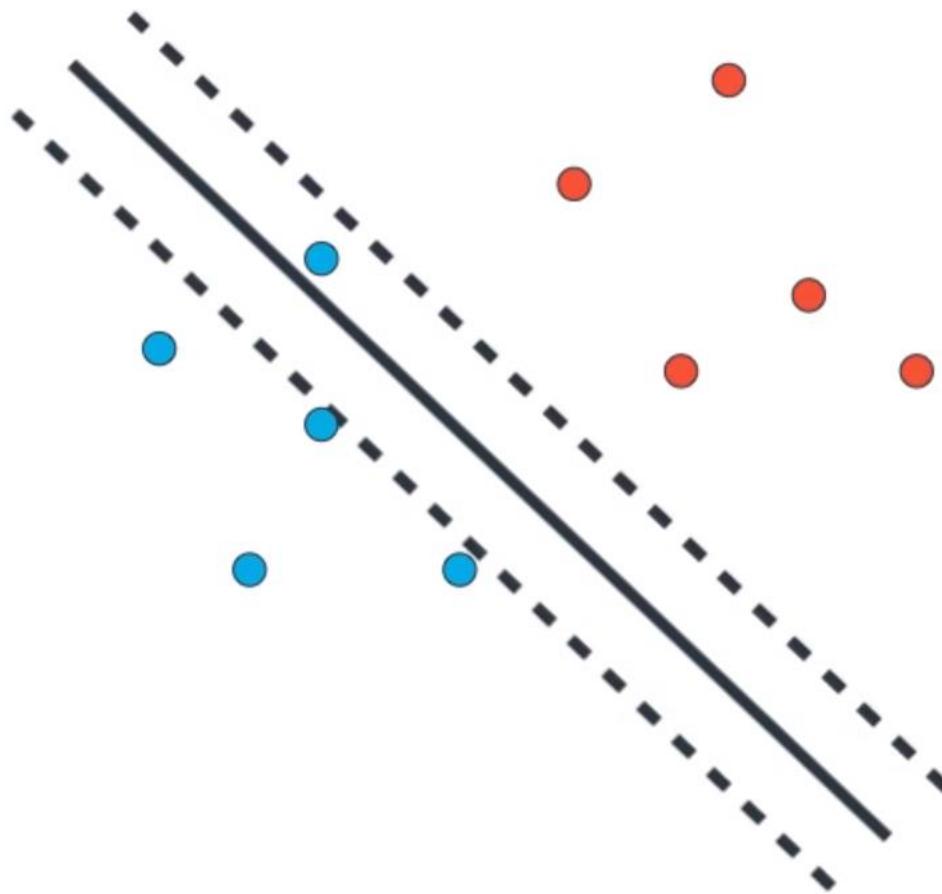
Split data - separate lines



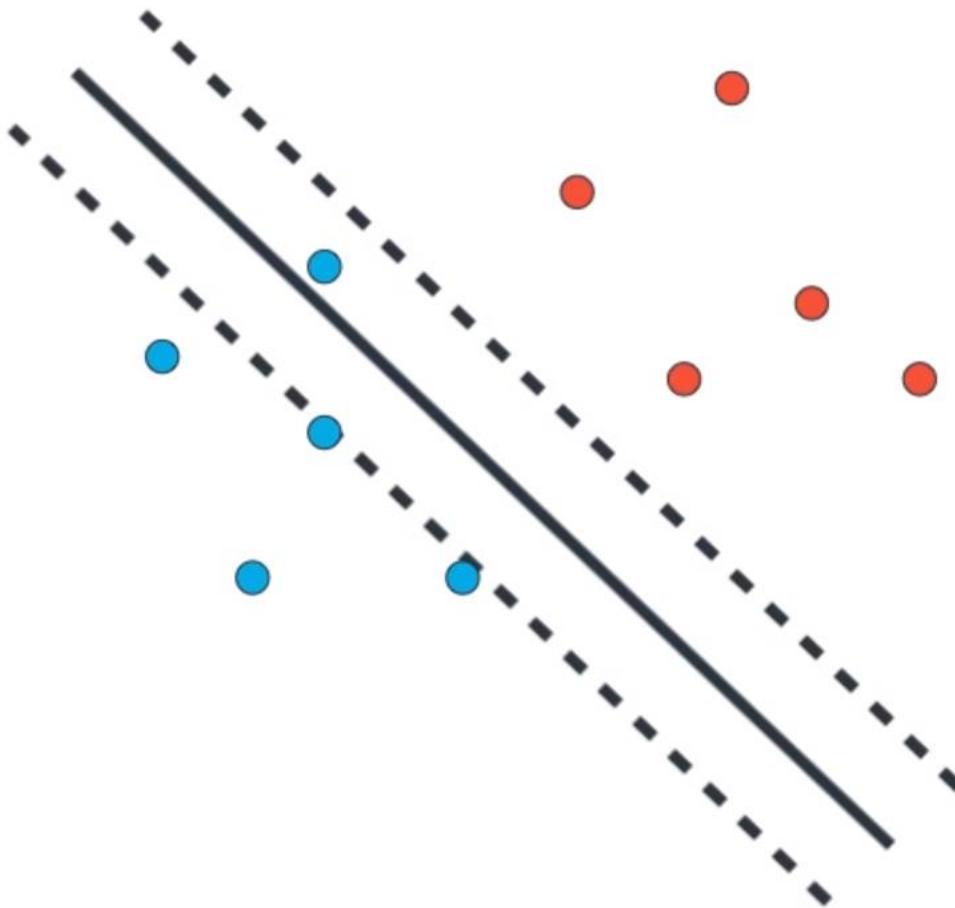
Split data - separate lines



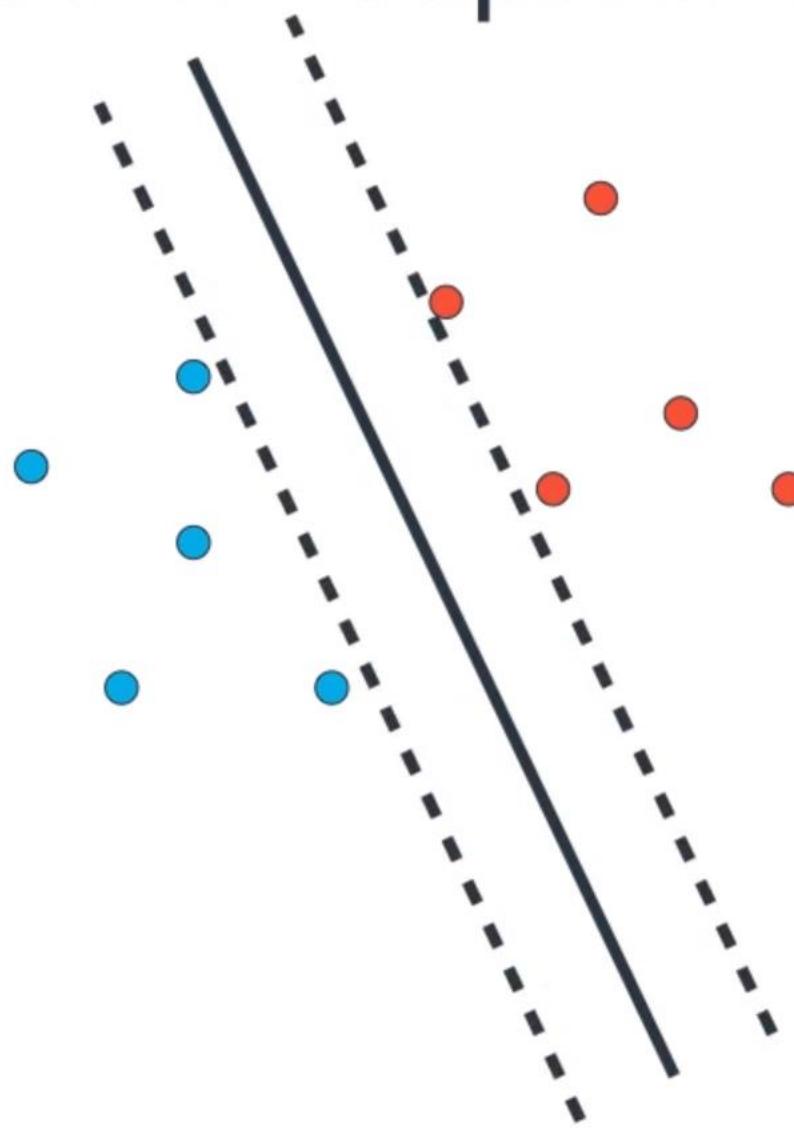
Split data - separate lines



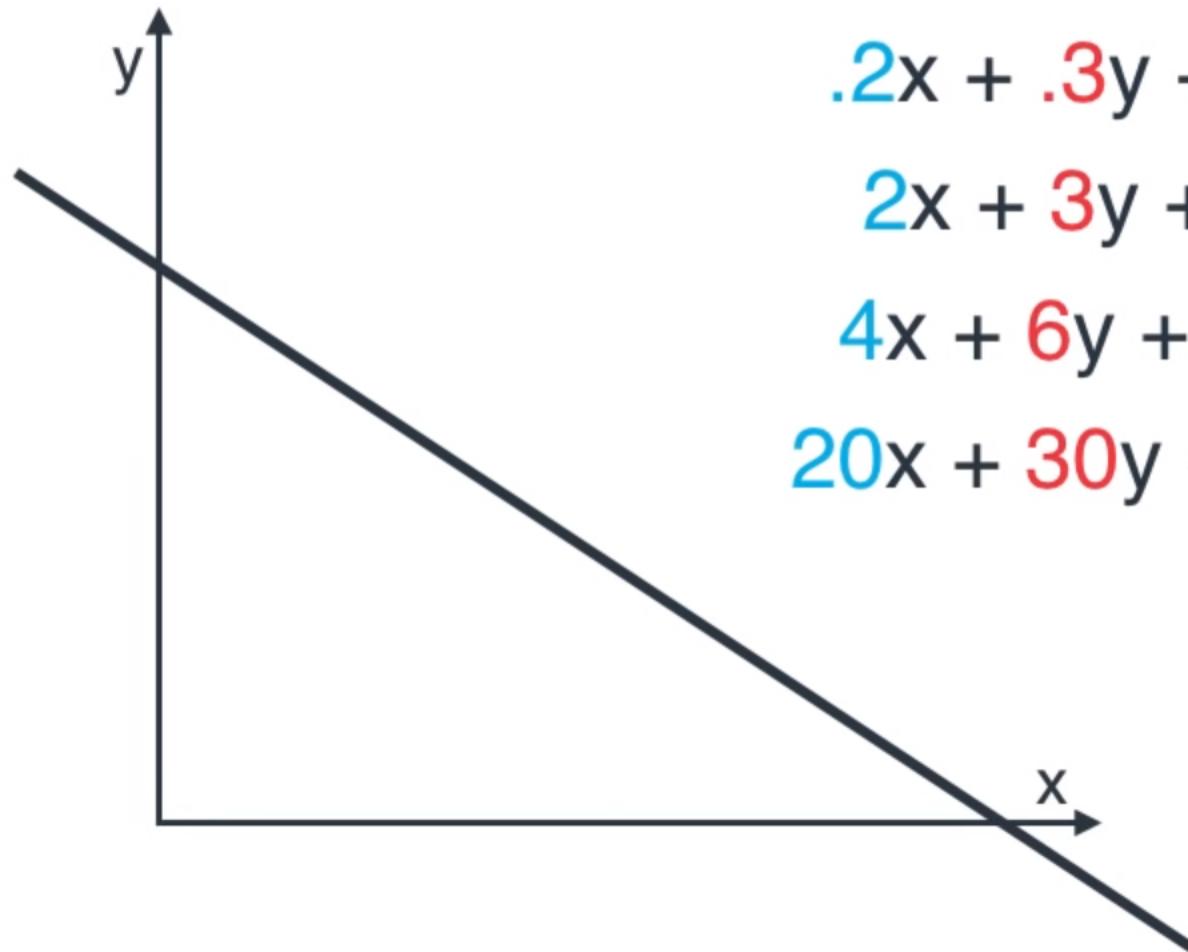
Split data - separate lines



Split data - separate lines



How to separate lines?



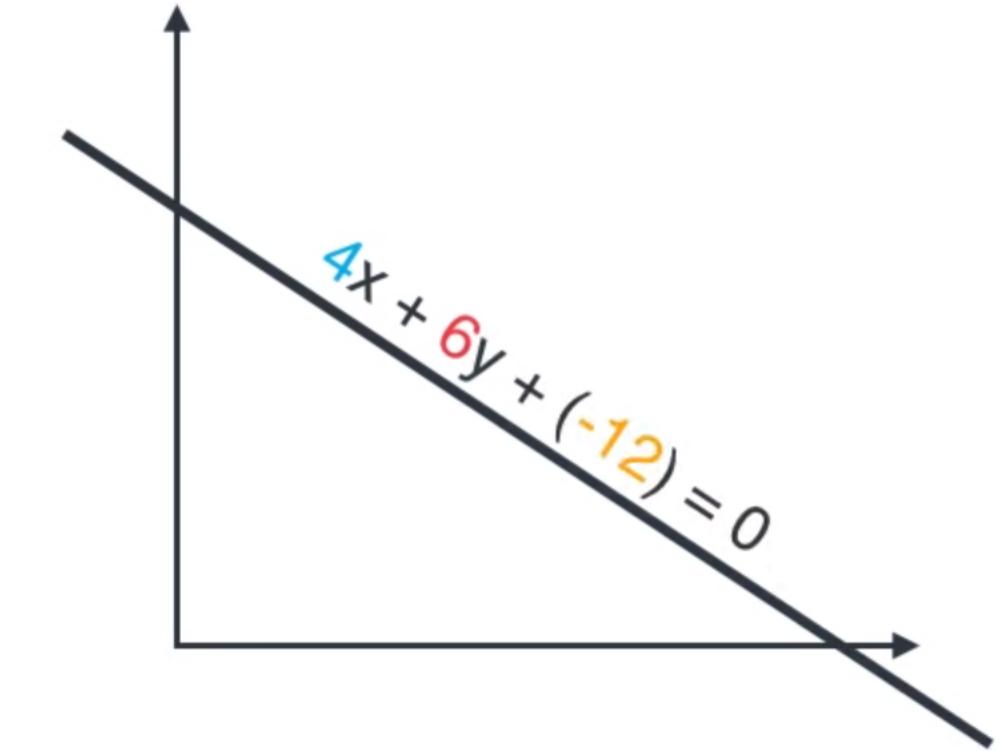
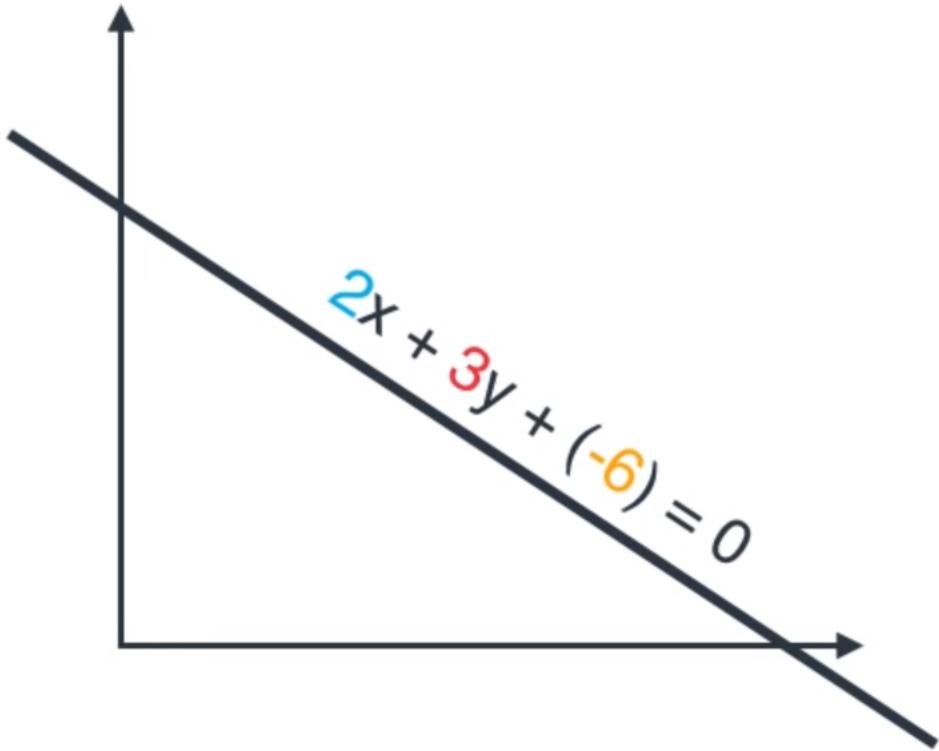
$$.2x + .3y + (-.6) = 0$$

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

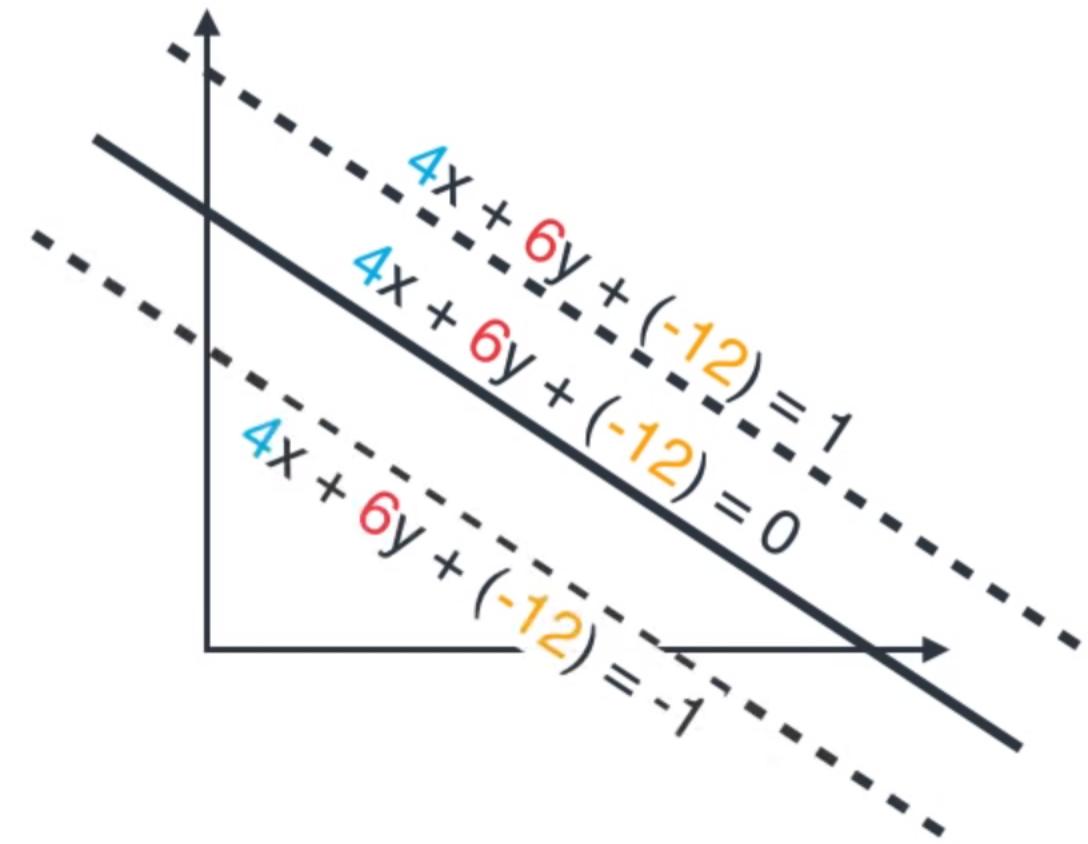
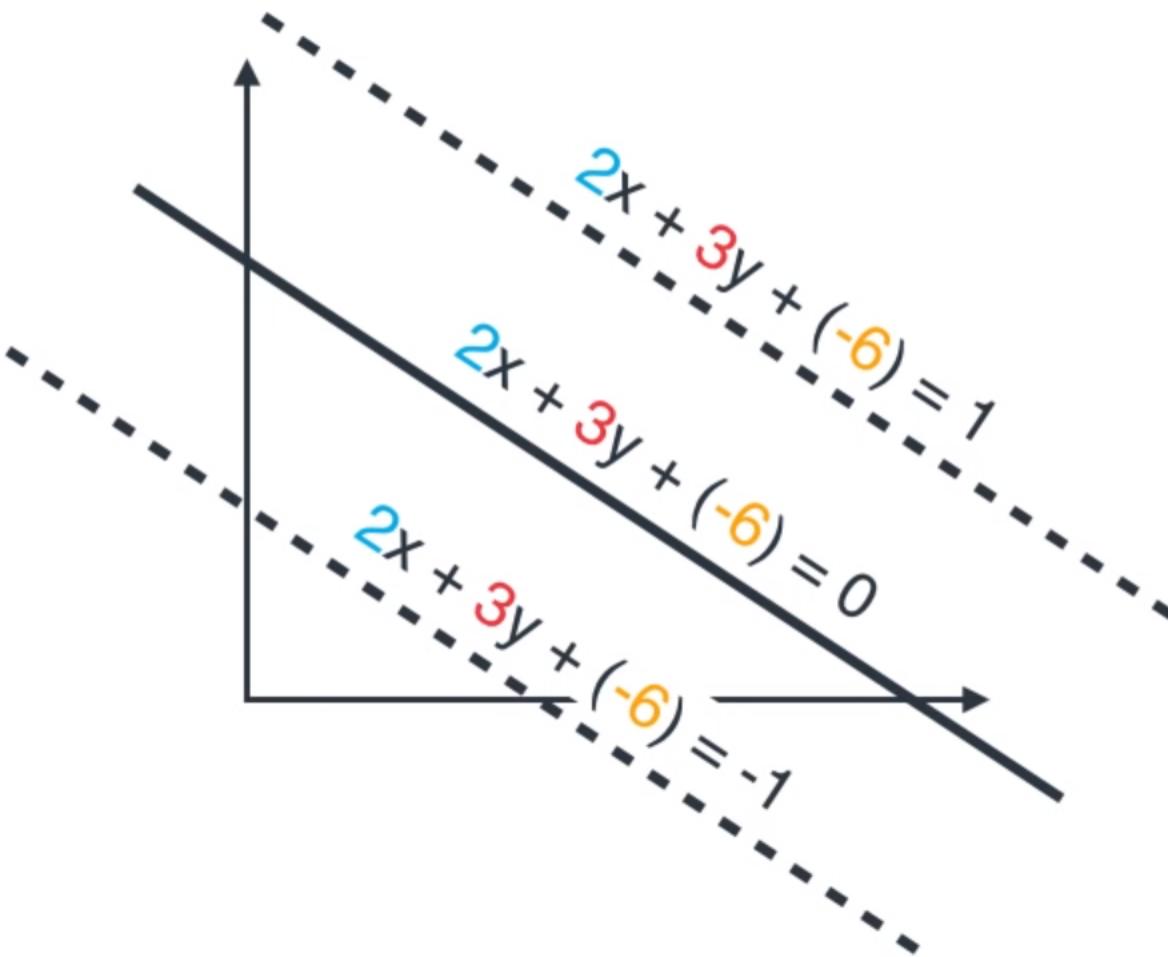
$$4x + 6y + (-12) = 0$$

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

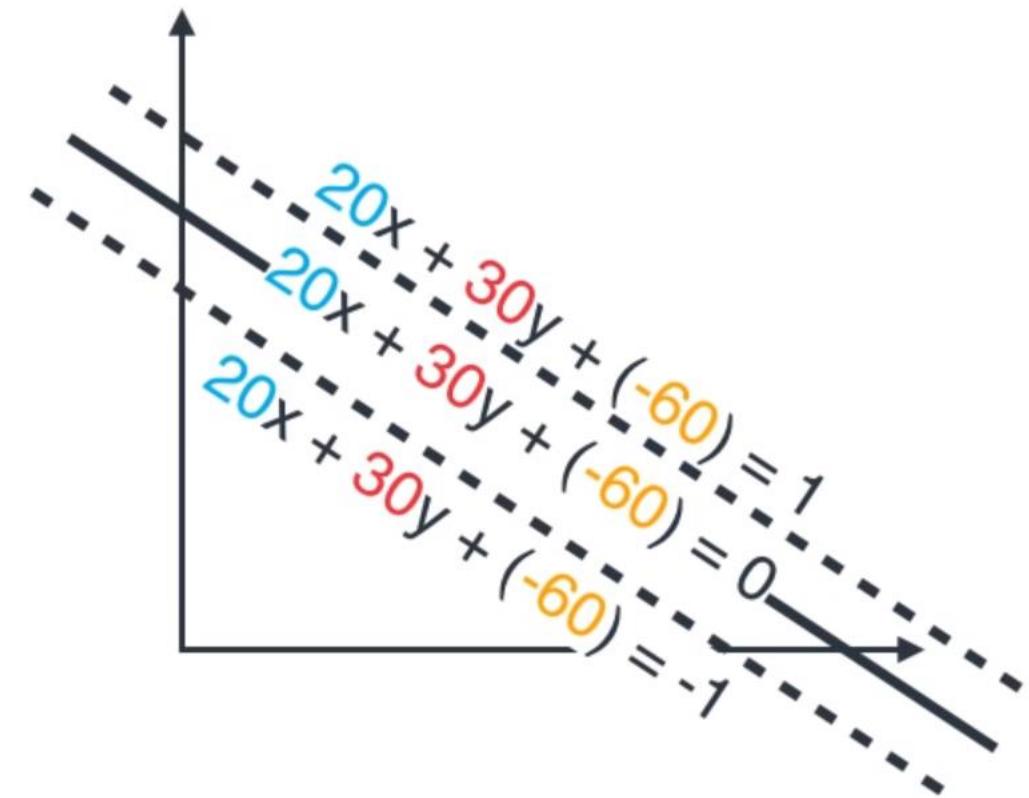
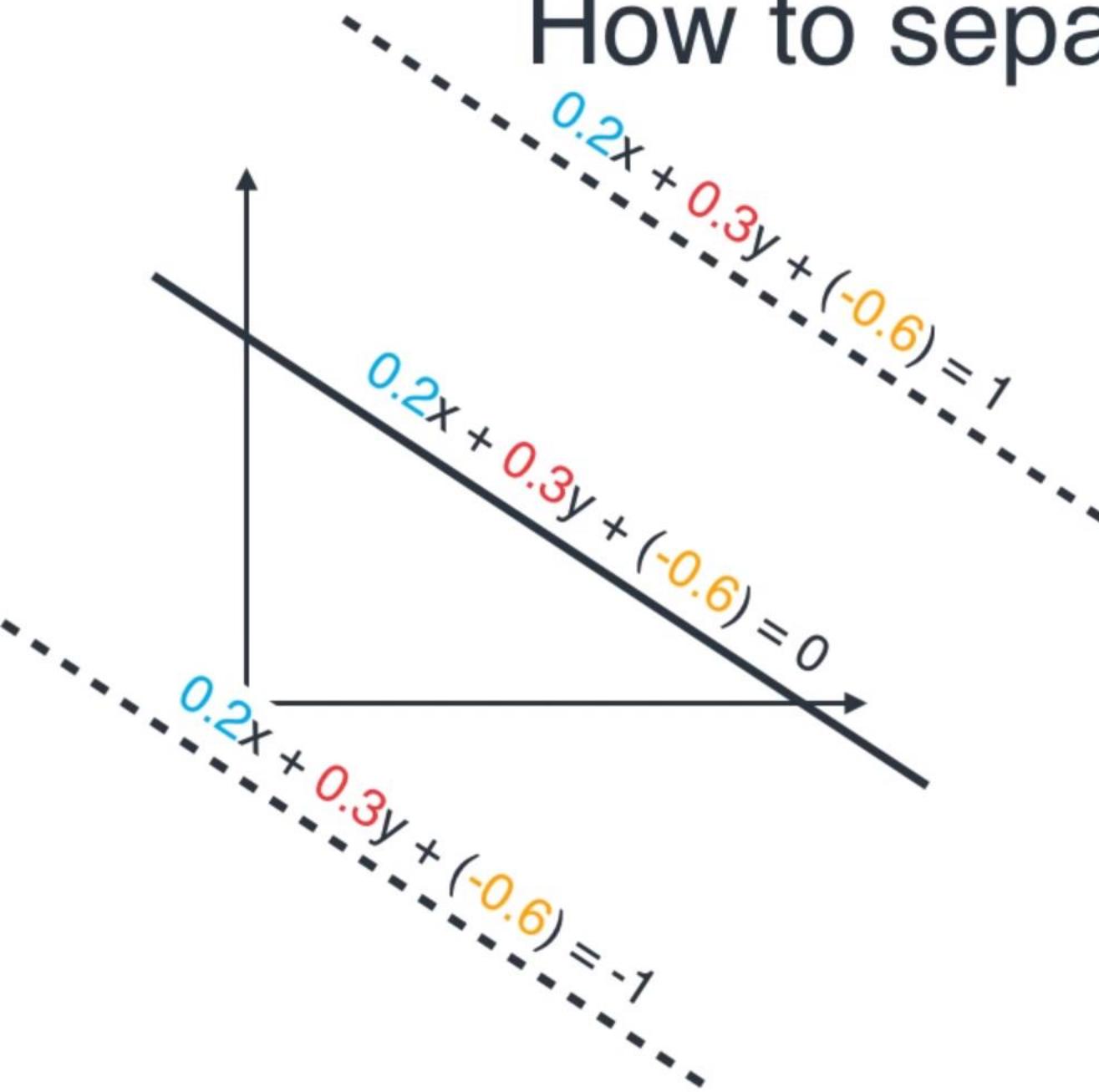
How to separate lines?



How to separate lines?



How to separate lines?



The original lines stays the same if multiplied by a constant. But the parallel lines move farther away or closer depending on the number multiplied is smaller or larger.

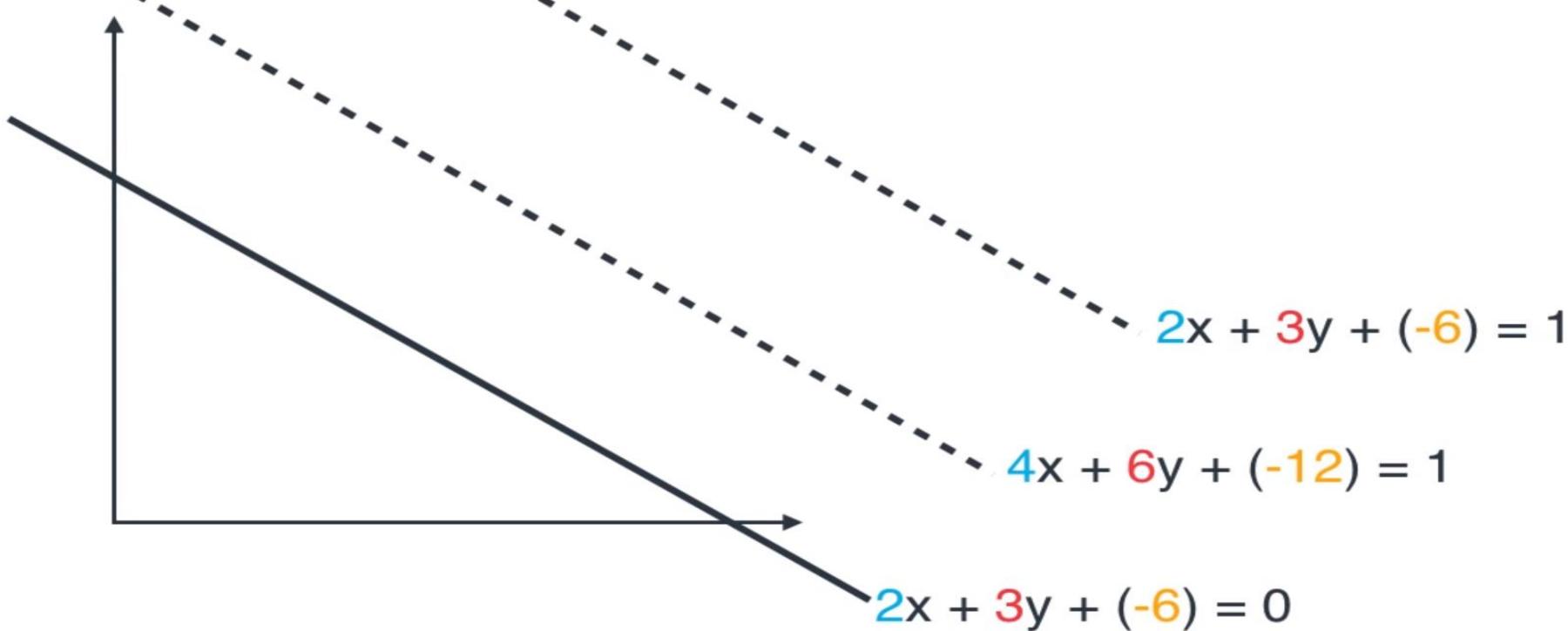
If the parallel lines multiplied by a negative numbers, the two lines switch. However it's not so important for this algorithm.

Well, we're going to multiply the lines with a smaller number.

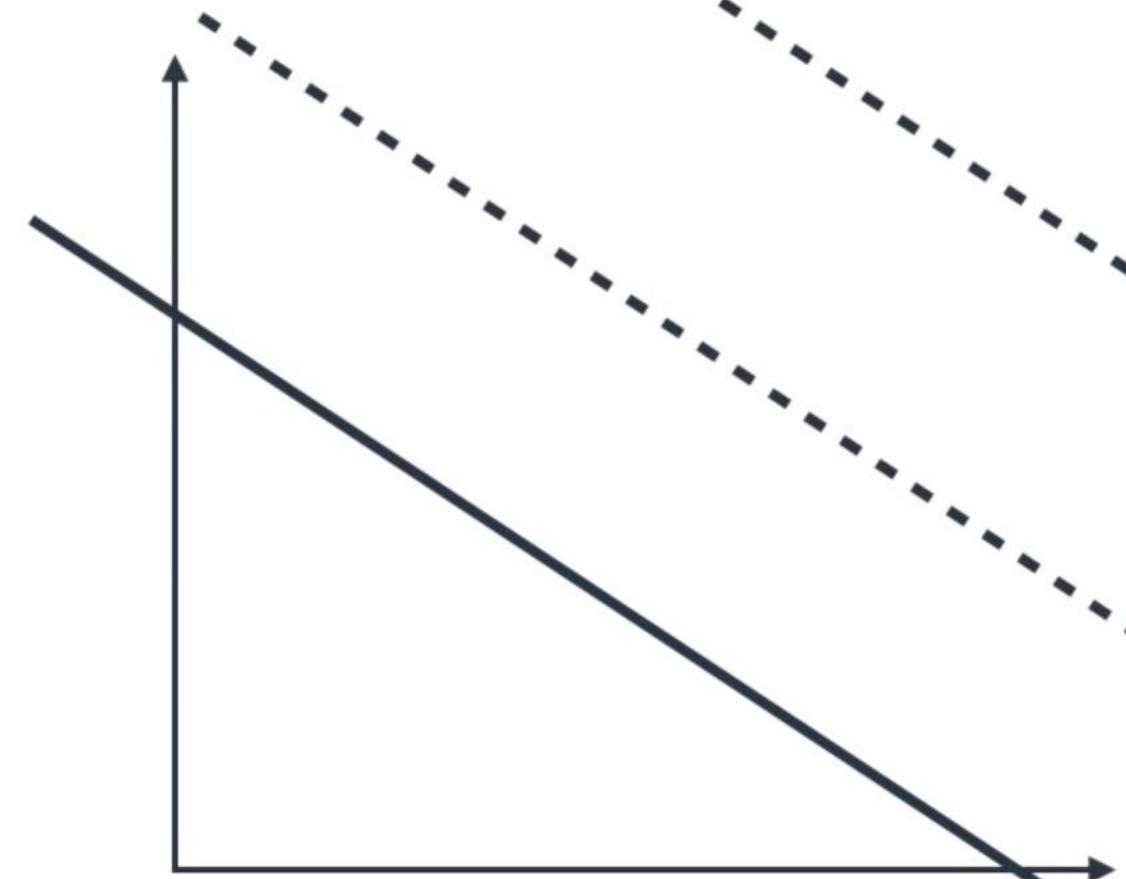
But before we do that, we need some justification why does this phenomenon happen.

Why is this line in the centre $4x+6y+(-12) = 1$?

How to separate lines?



How to separate lines?



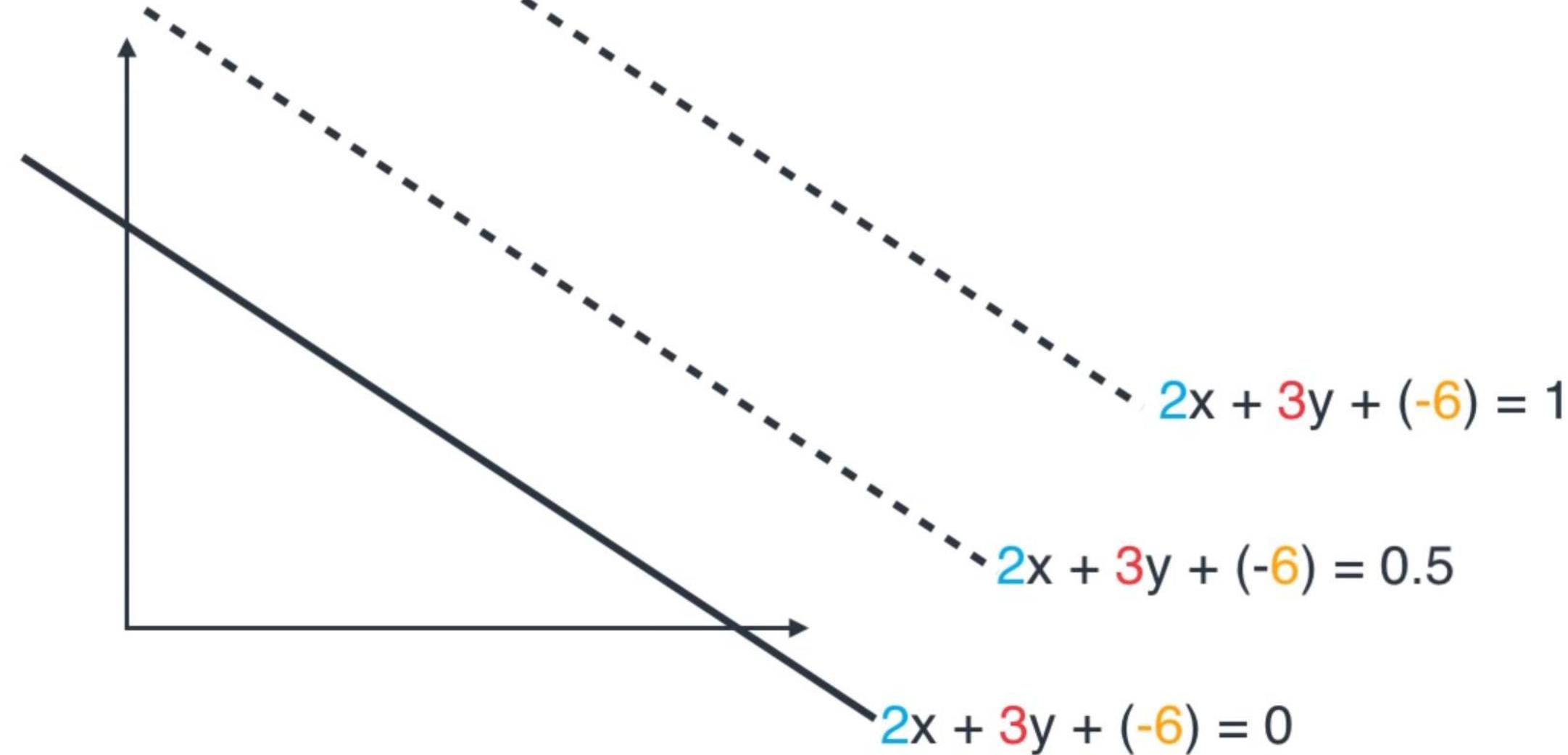
$$2x + 3y + (-6) = 0$$

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

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

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

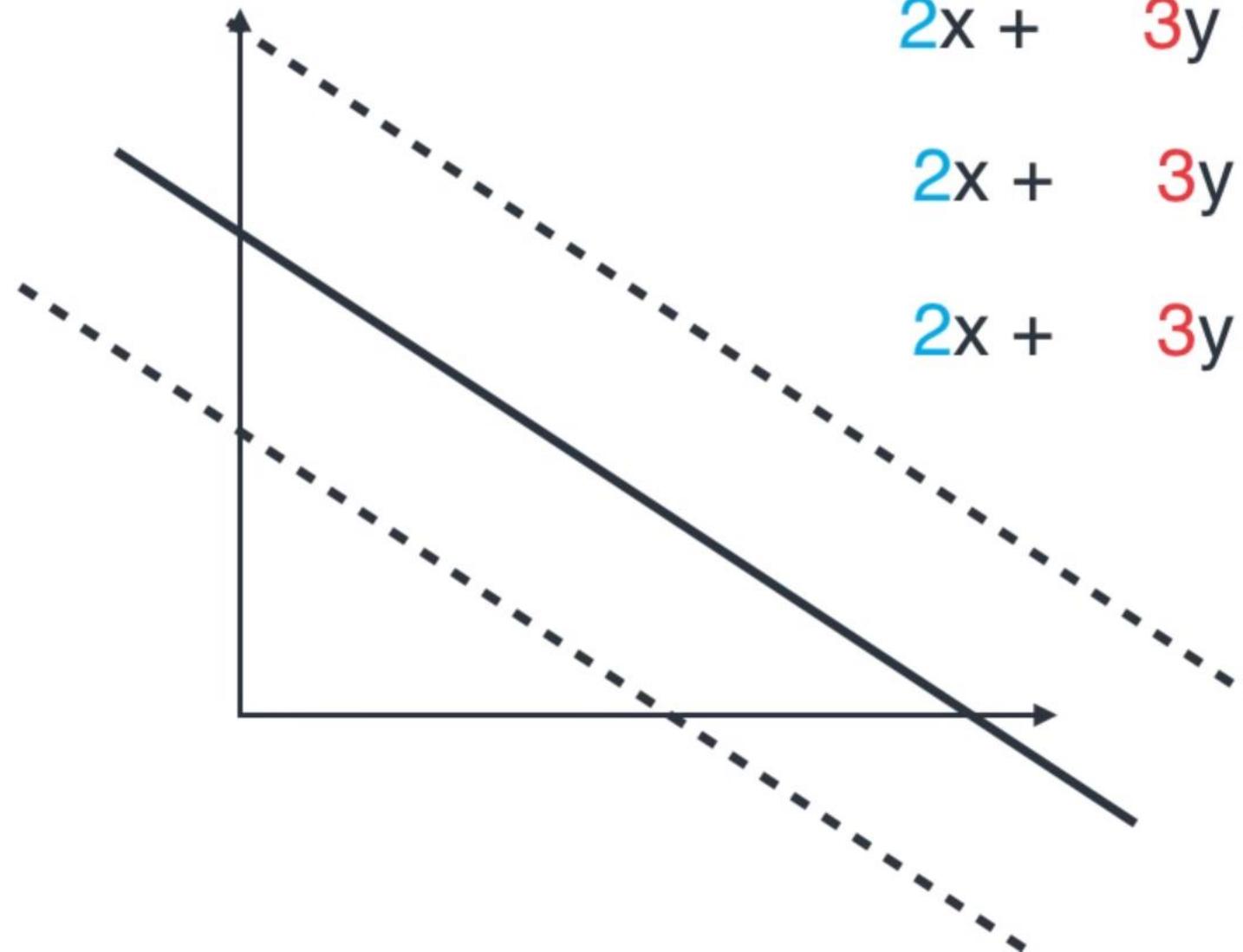
How to separate lines?



Expanding rate

Expanding rate

0.99



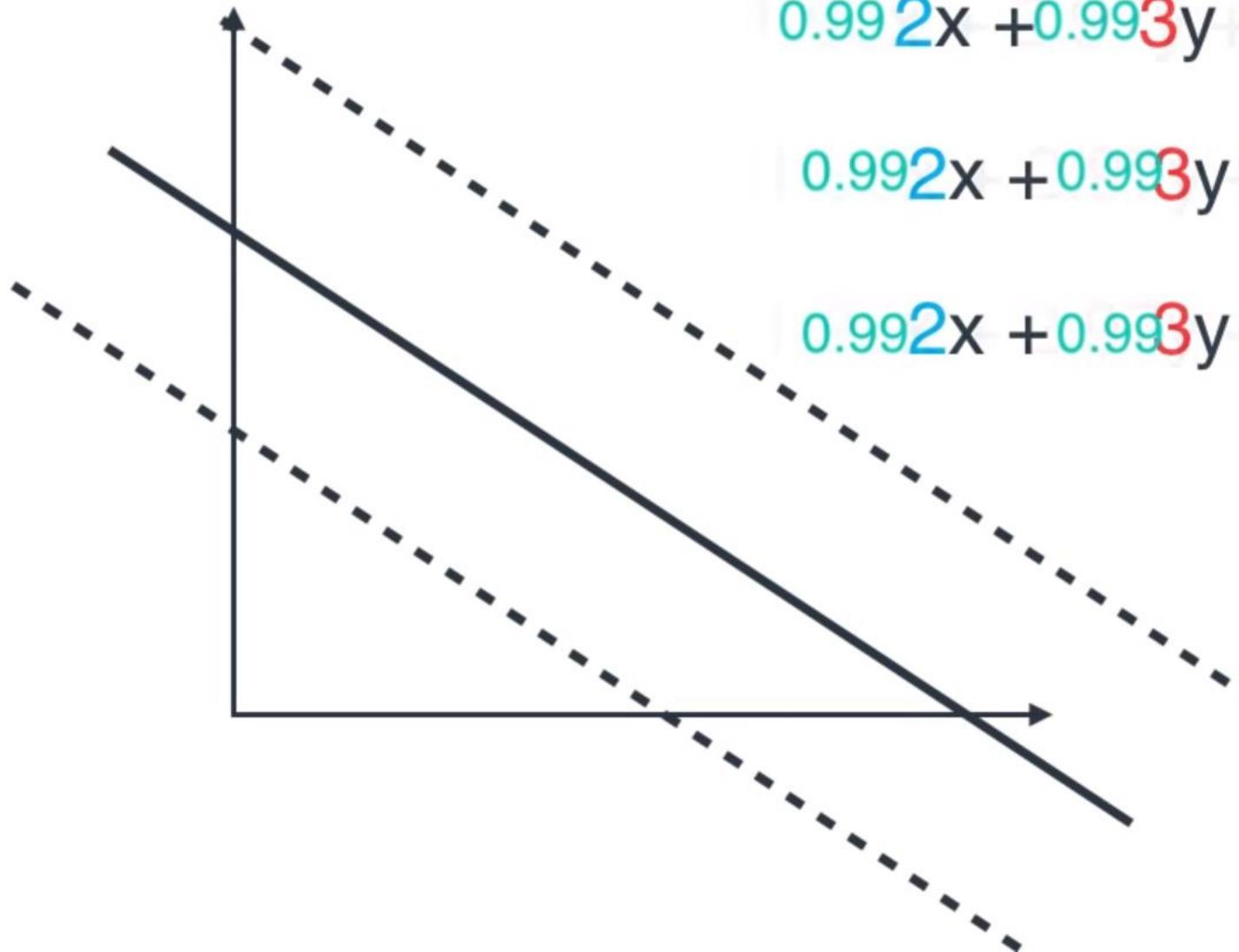
$$2x + 3y + (-6) = -1$$

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

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

Expanding rate

Expanding rate



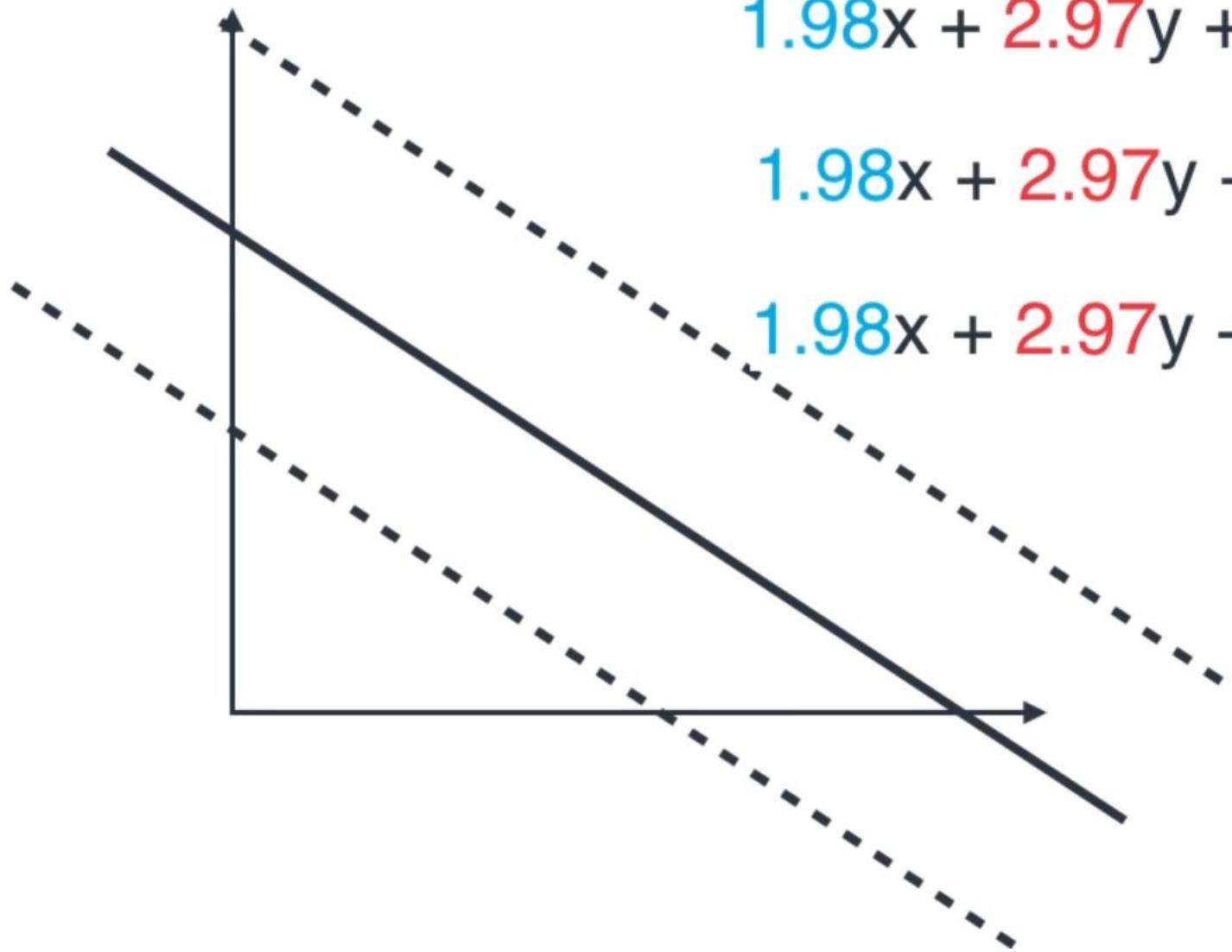
$$0.992x + 0.993y + 0.99(-6) = -1$$

$$0.992x + 0.993y + 0.99(-6) = 0$$

$$0.992x + 0.993y + 0.99(-6) = 1$$

Expanding rate

Expanding rate



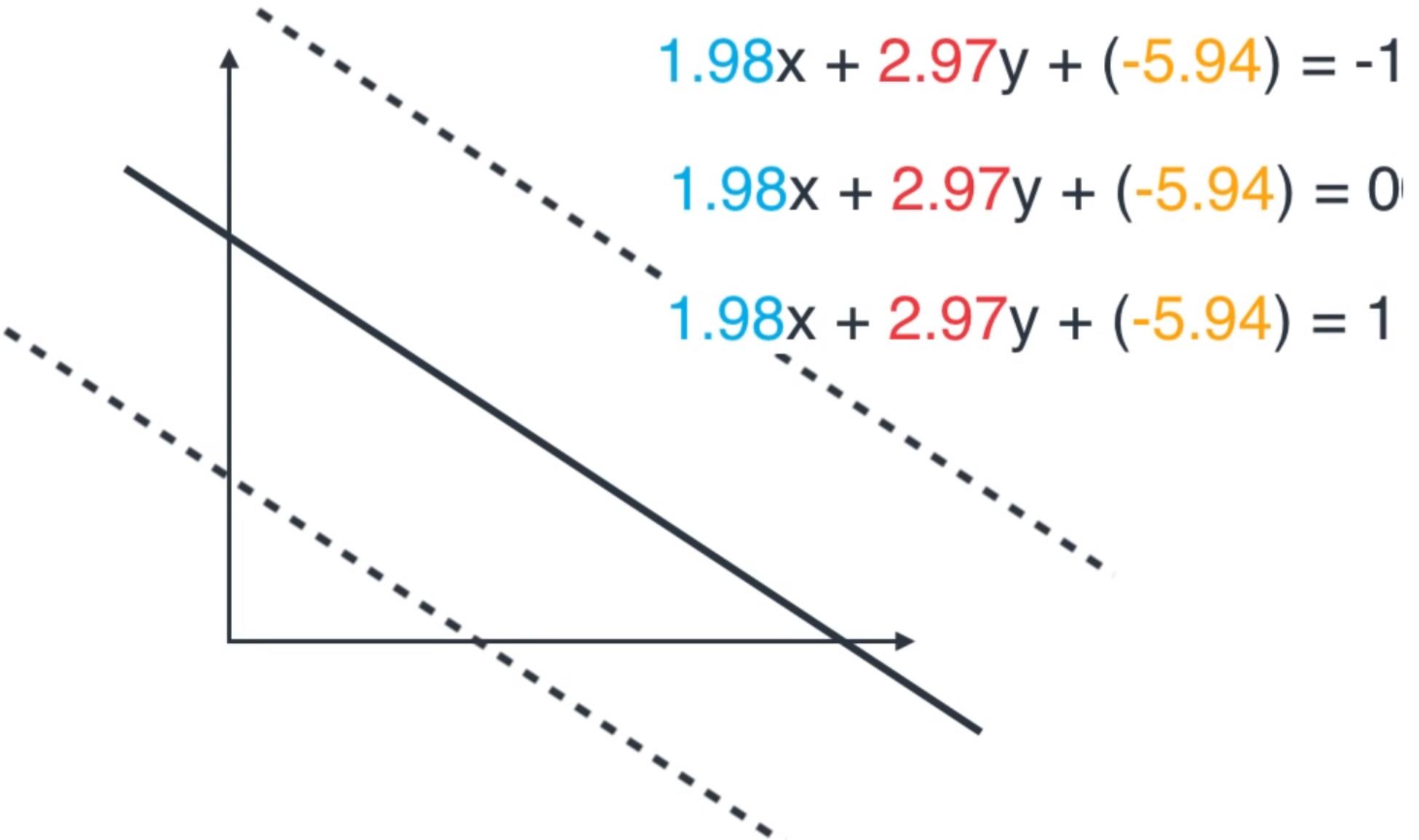
$$1.98x + 2.97y + (-5.94) = -1$$

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

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

Expanding rate

Expanding rate

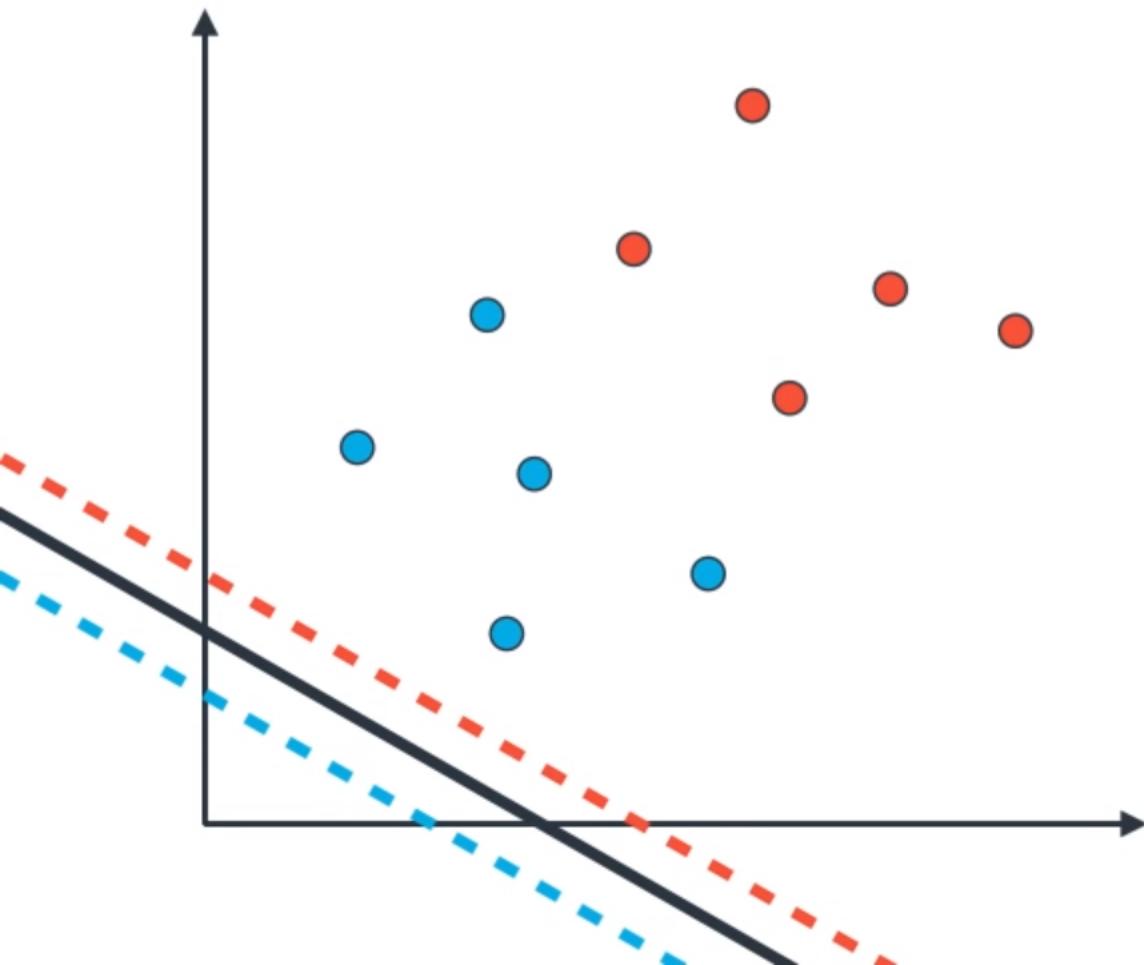


Now the lines spread a little bit.

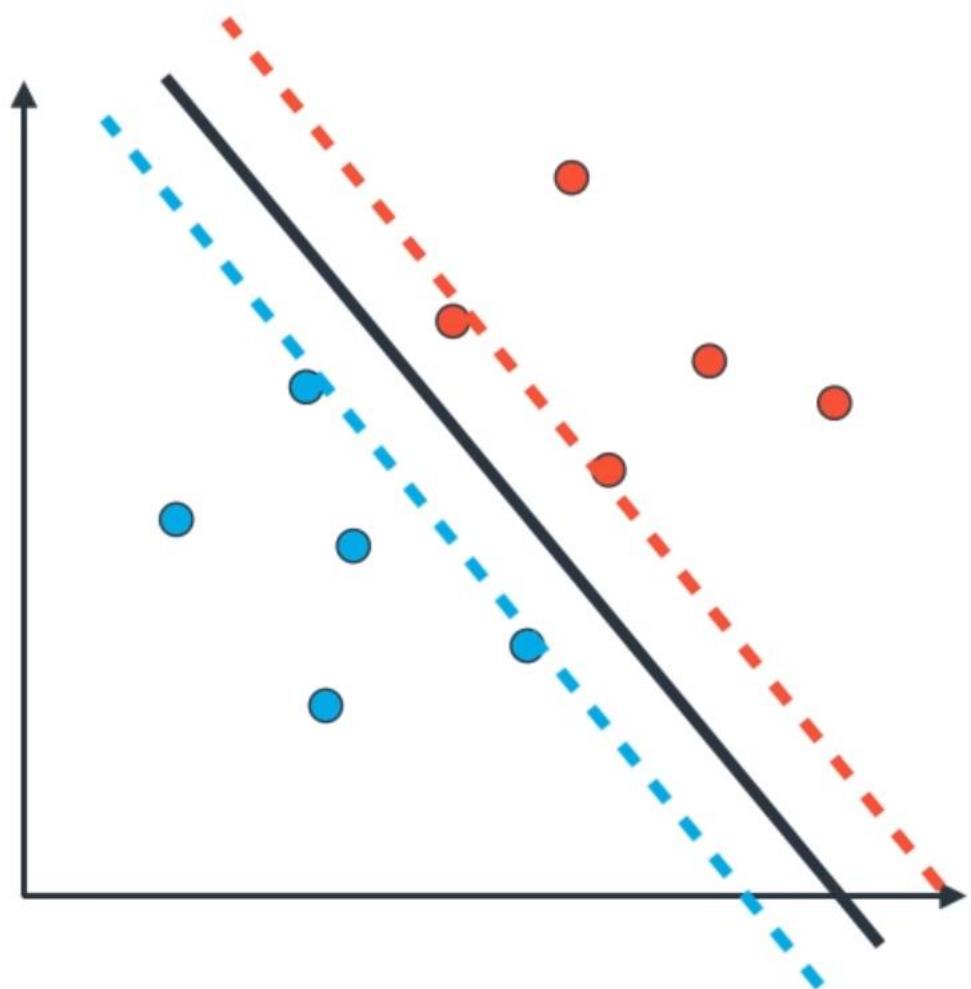
We're going to add this step to the perceptron algorithm which will spread the lines apart every time with error rate

SVM algorithm

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



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)

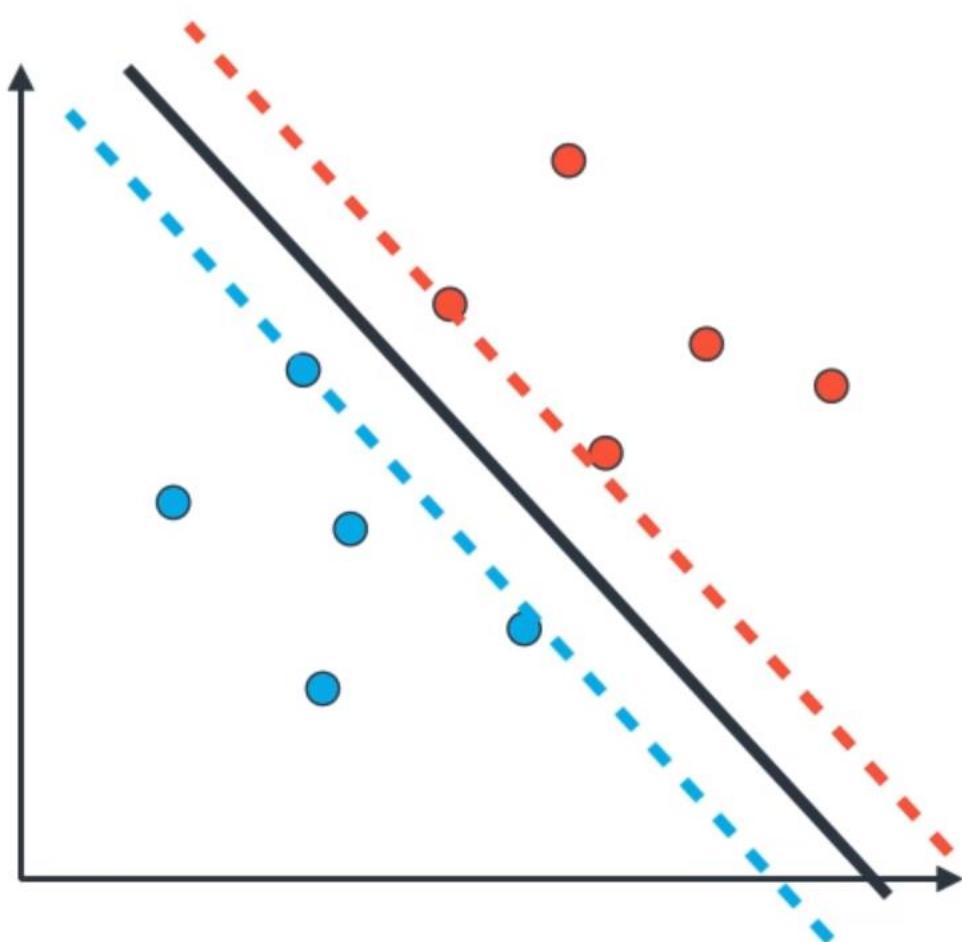
Step 3: Pick a number close to 1. (the expanding factor) **0.99**

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!

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**

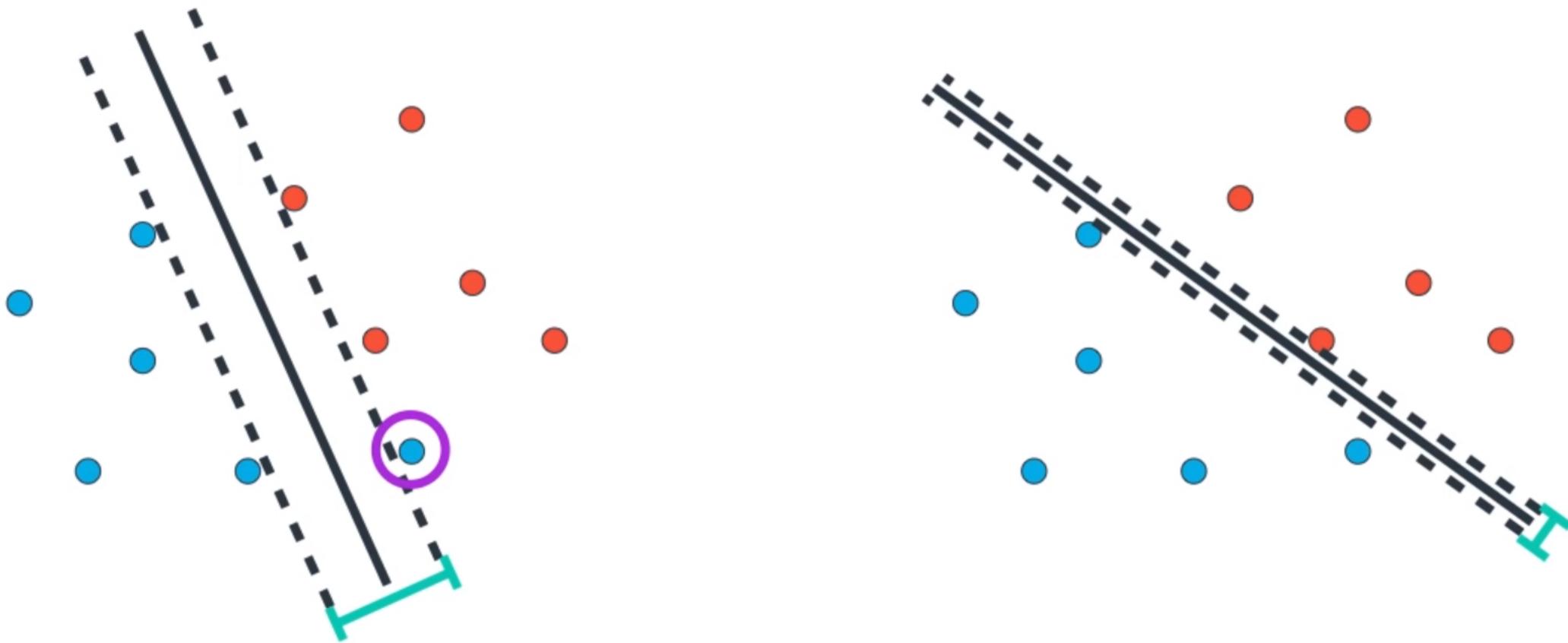
Step 4: Pick an expanding rate. **0.99**

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 $0.01p$ to a
 - Subtract $0.01q$ to b
 - Subtract 0.01 to c
- If point is, **red** and $ap+bq+c < 0$
 - Add $0.01p$ to a
 - Add $0.01q$ to b
 - Add 0.01 to c

Multiply a, b, c, by 0.99

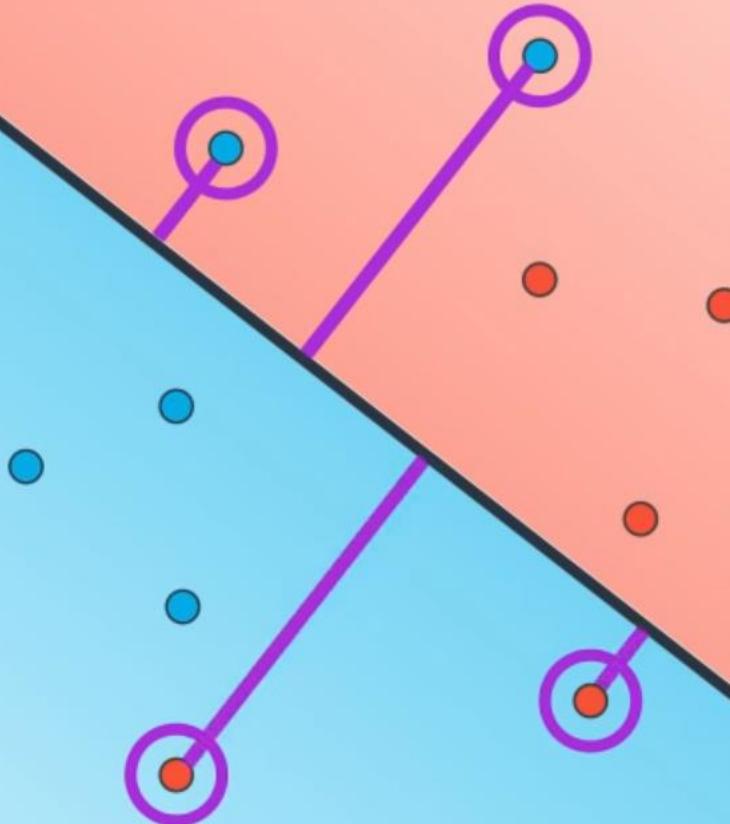
Which line is better?



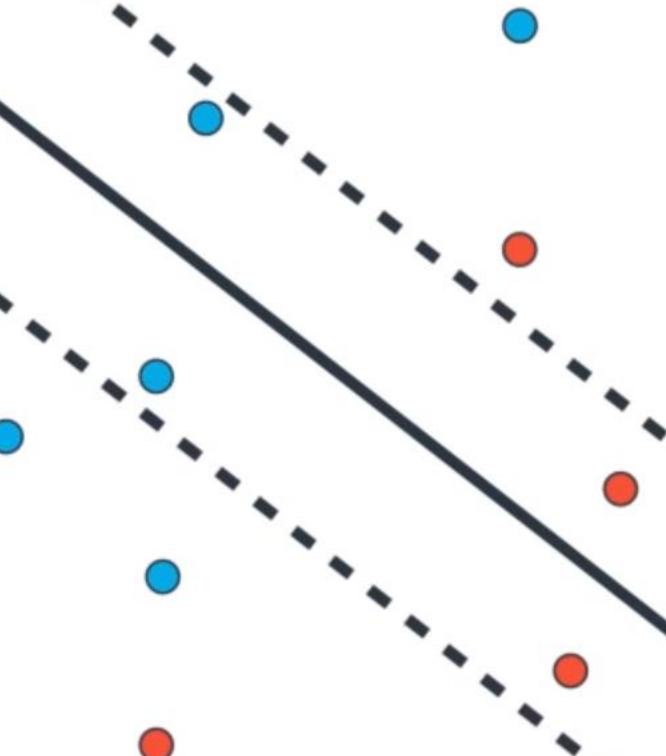
Which one is better?

We don't really know, but with the error function
we can probably analyse

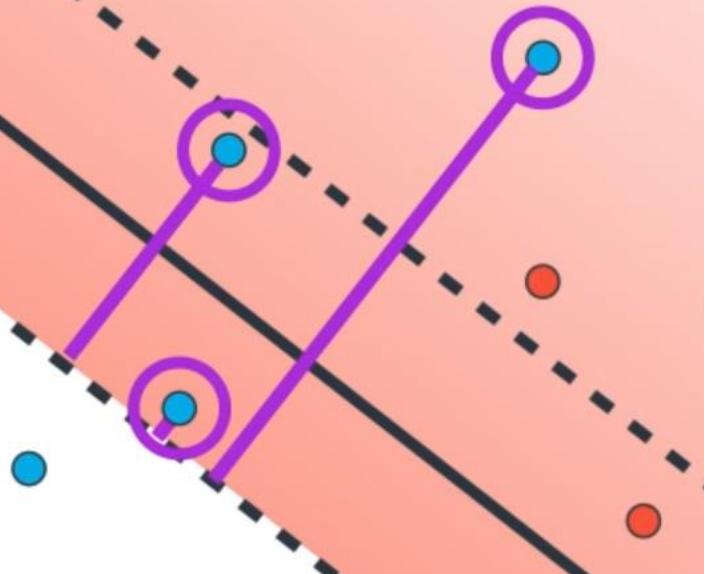
Perceptron Error



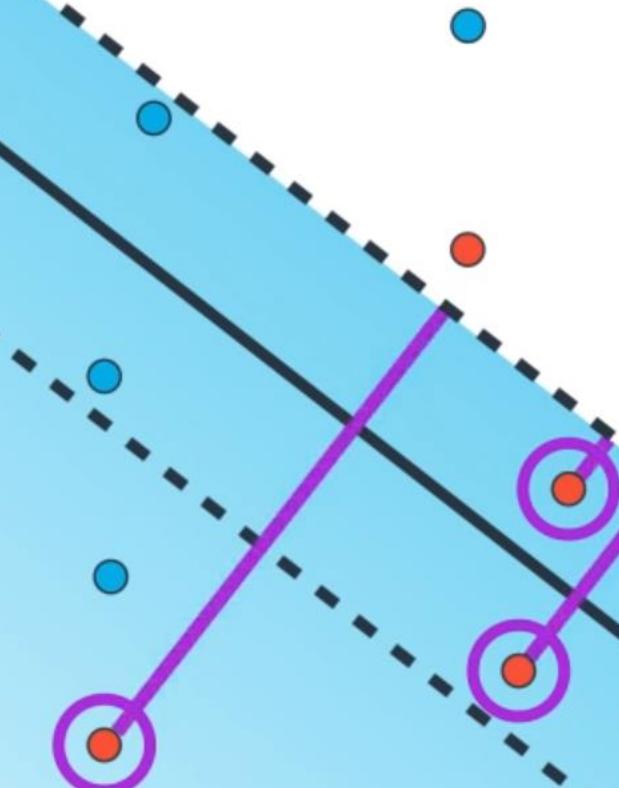
SVM Classification Error



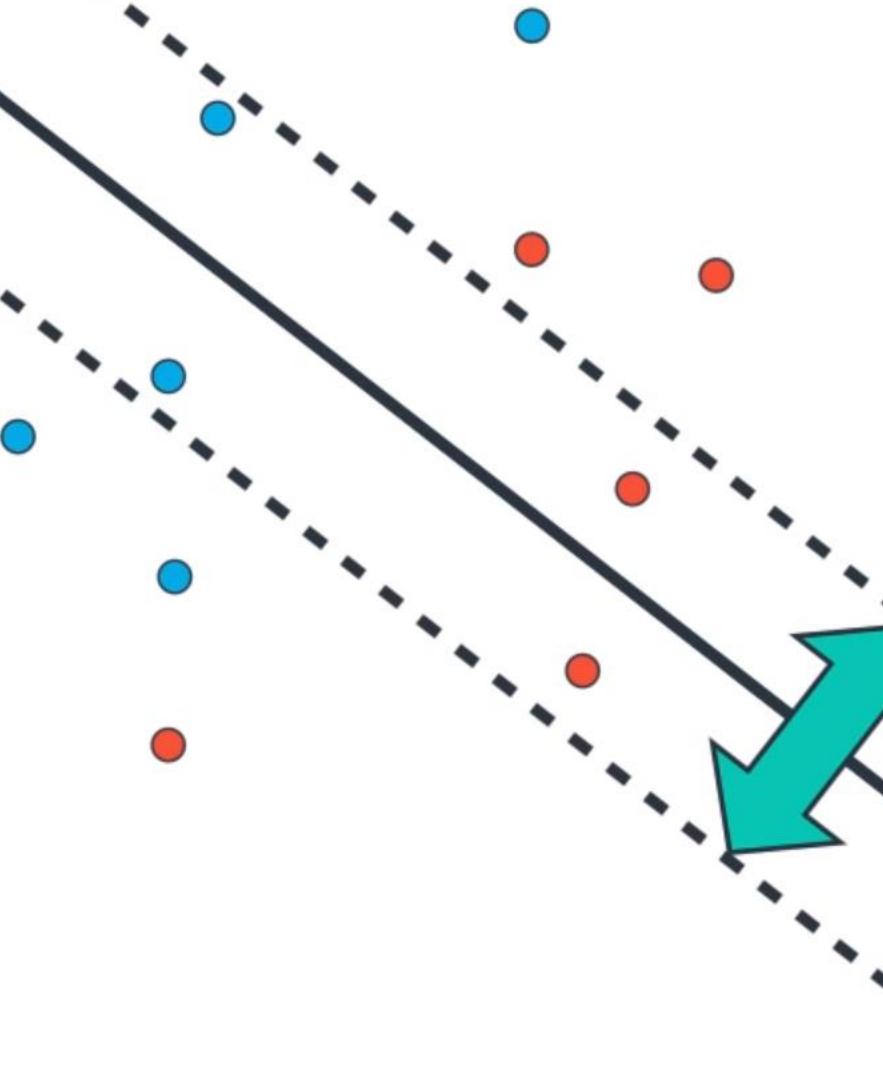
SVM Classification Error



SVM Classification Error



Margin Error

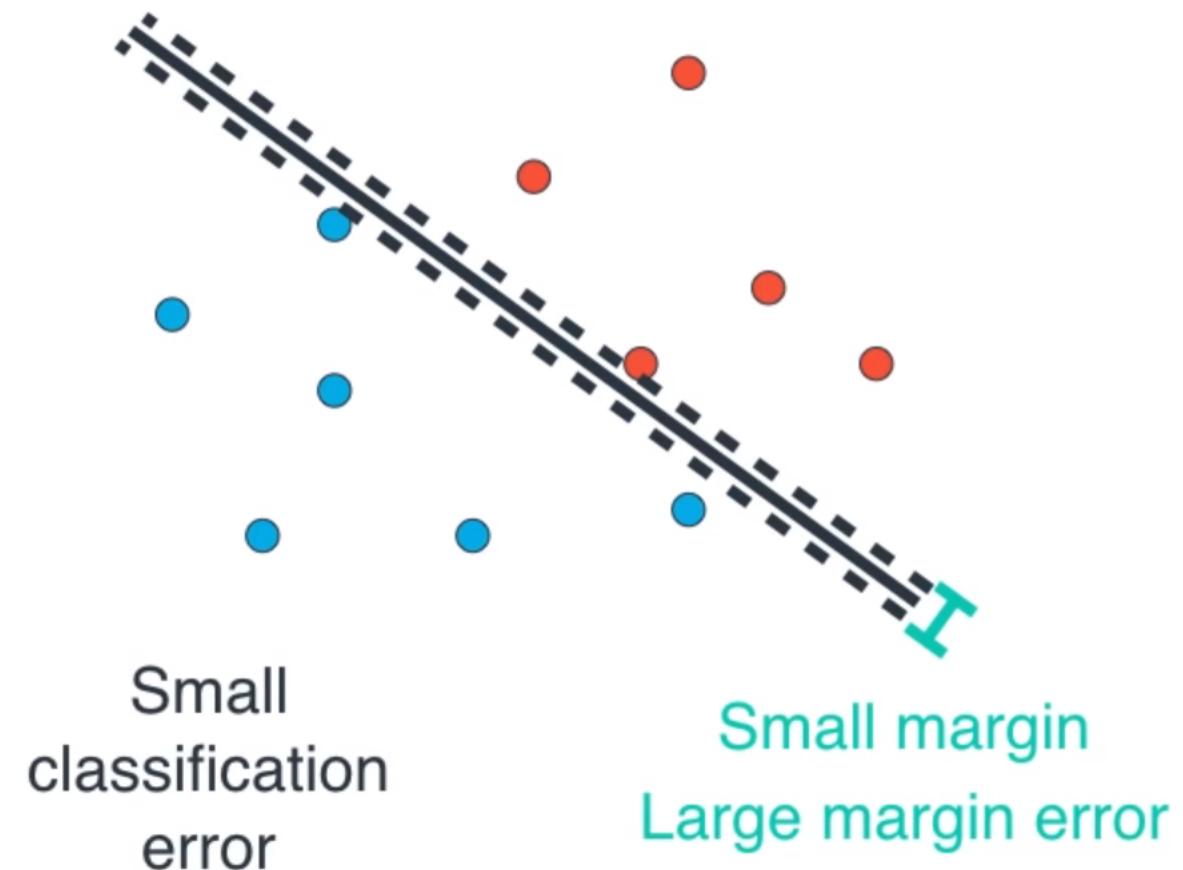
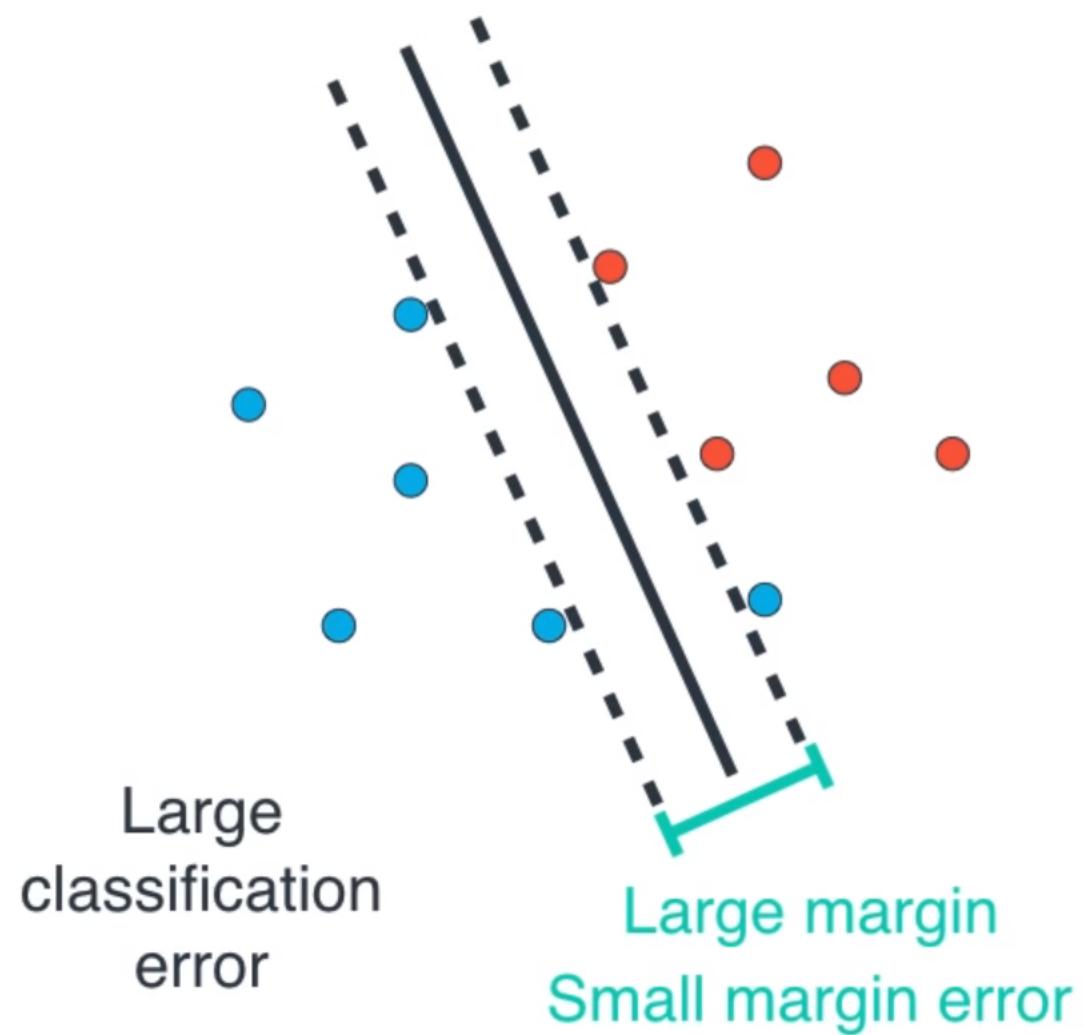


Margin error tells if these lines are close by or far apart.

Basically it's a number that's going to be big if the lines are close together. And small, if the lines are far apart because eventually it's an error.

Better the model, smaller the error and wider the lines are!

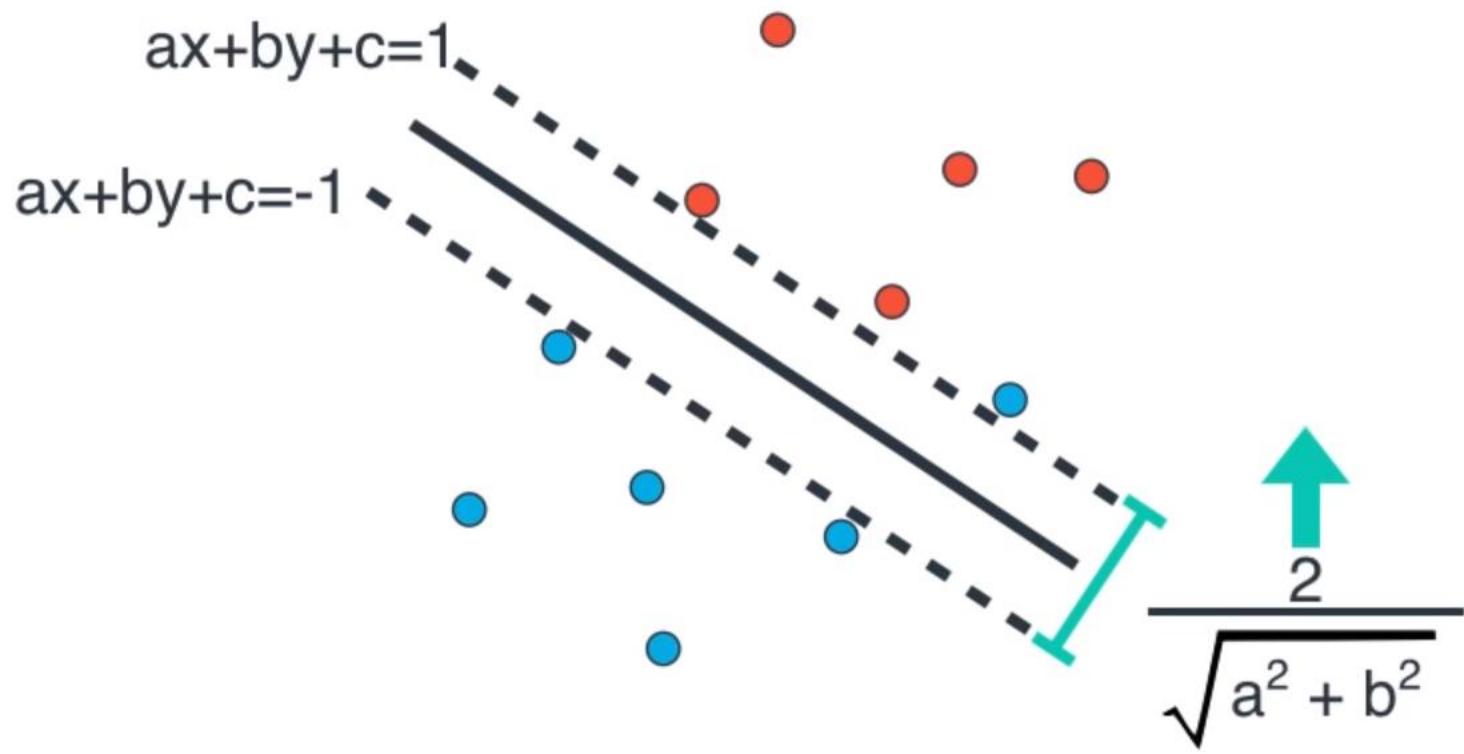
Margin Error



What can an error be?

We actually need a number if the distance is big, the number is small.
And number is small, if the distance is big.

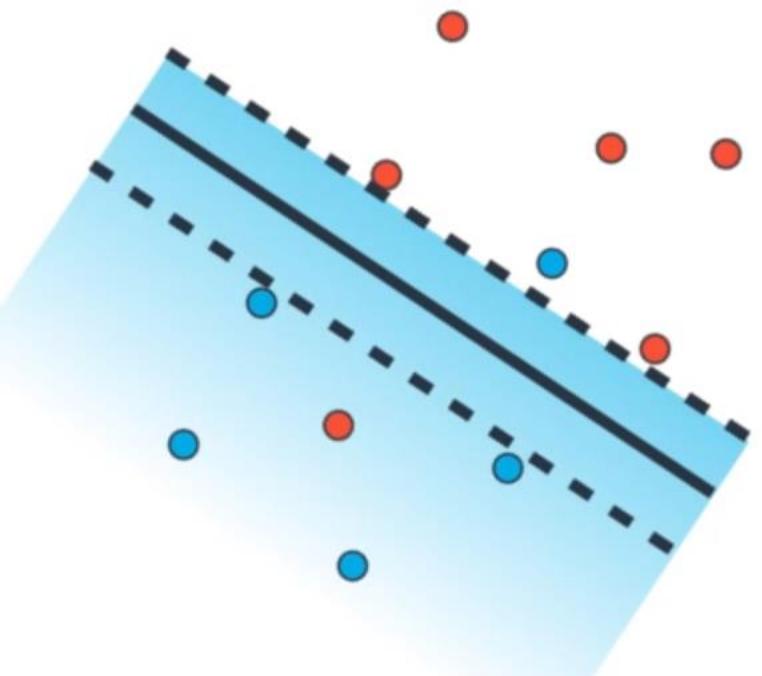
Margin Error



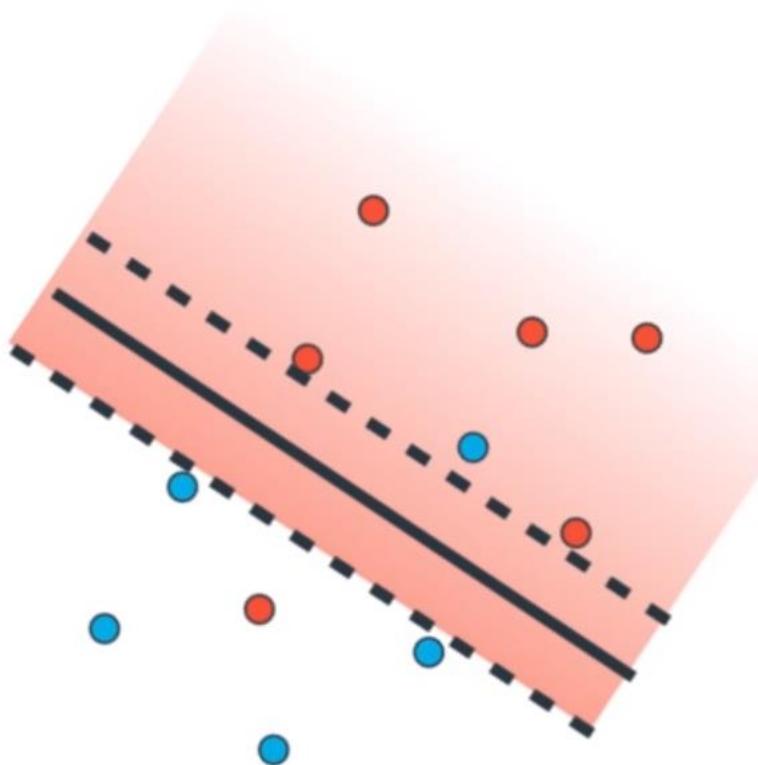
Regularization term!

$$\text{Margin error} = \frac{2}{\sqrt{a^2 + b^2}}$$

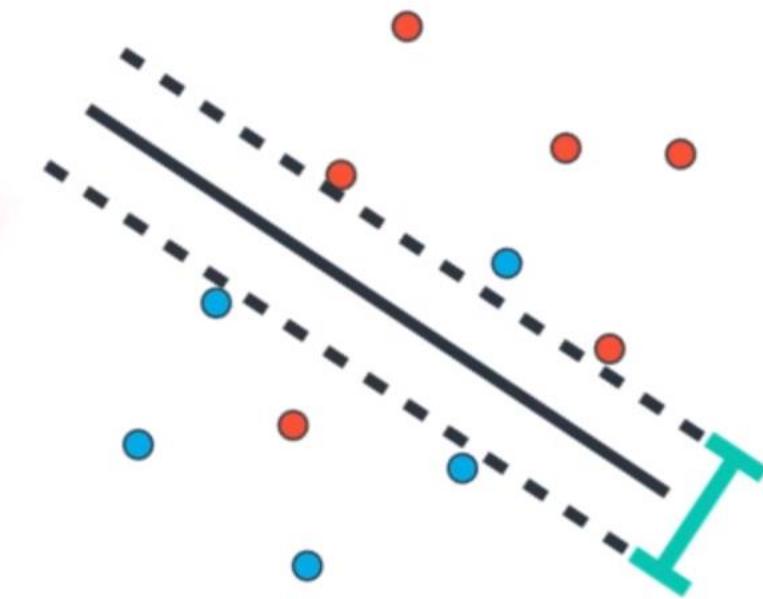
SVM Error



Blue Classification Error

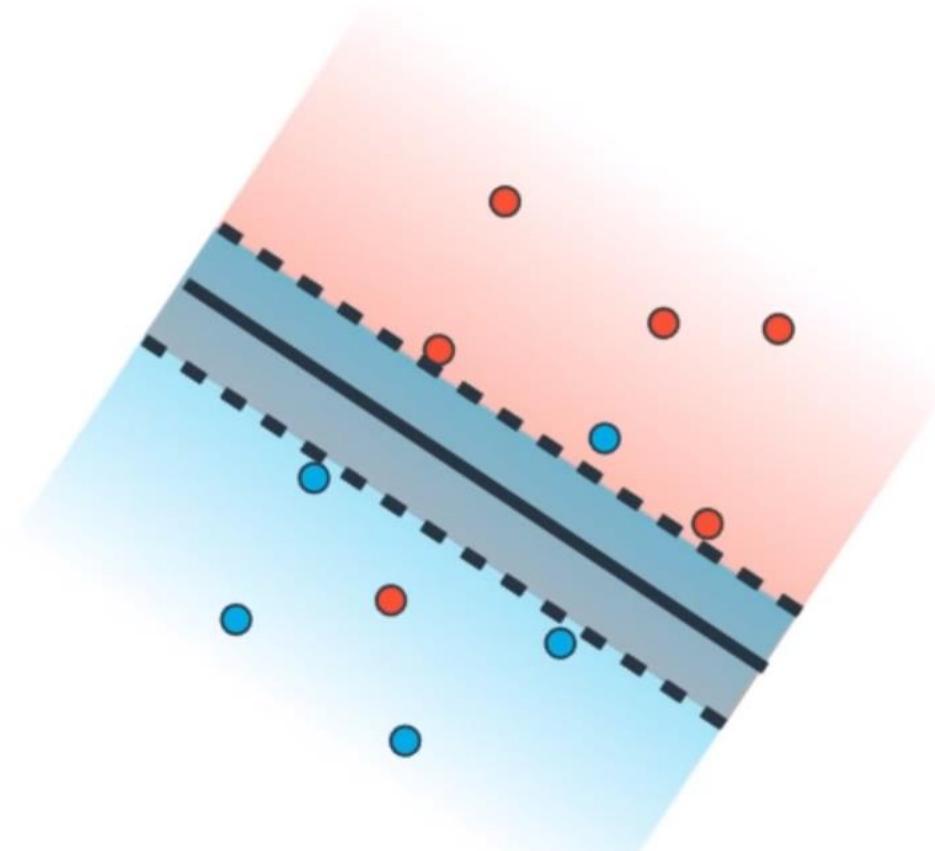


Red Classification Error

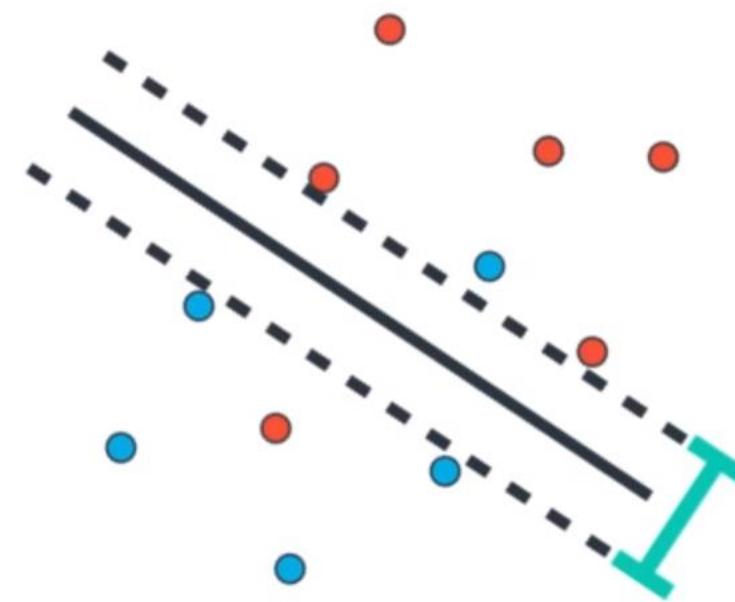


Margin Error

SVM Error

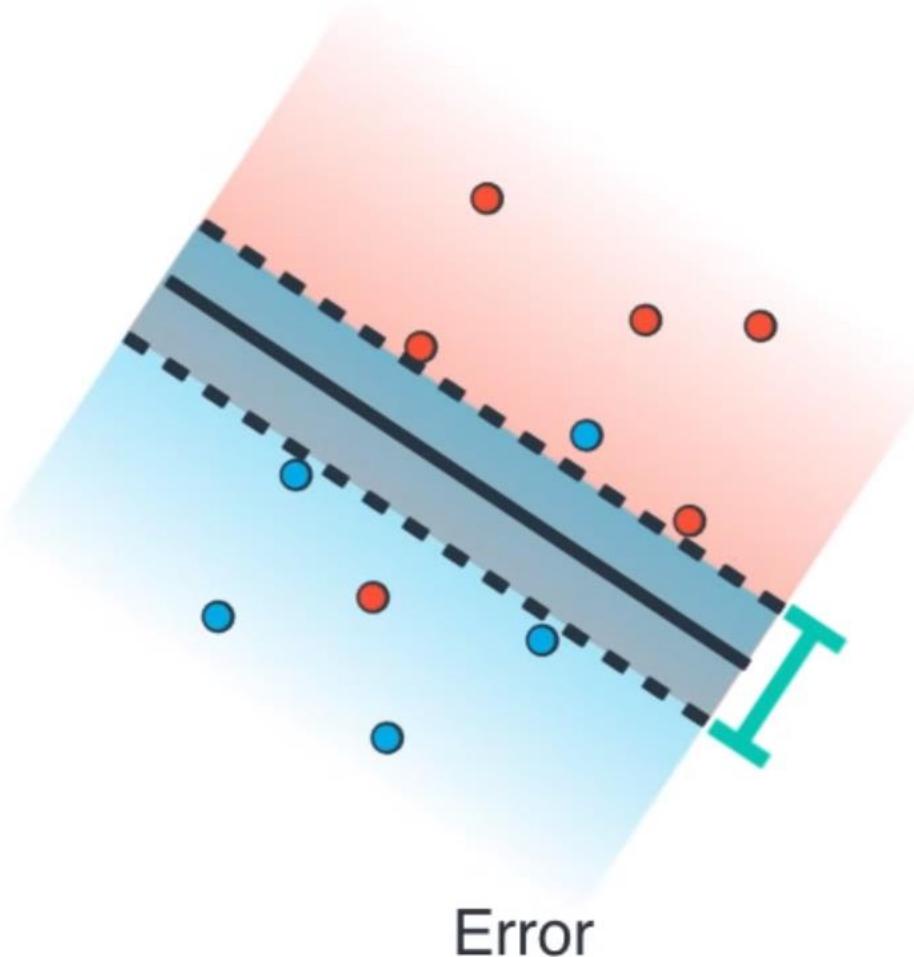


Classification Error



Margin Error

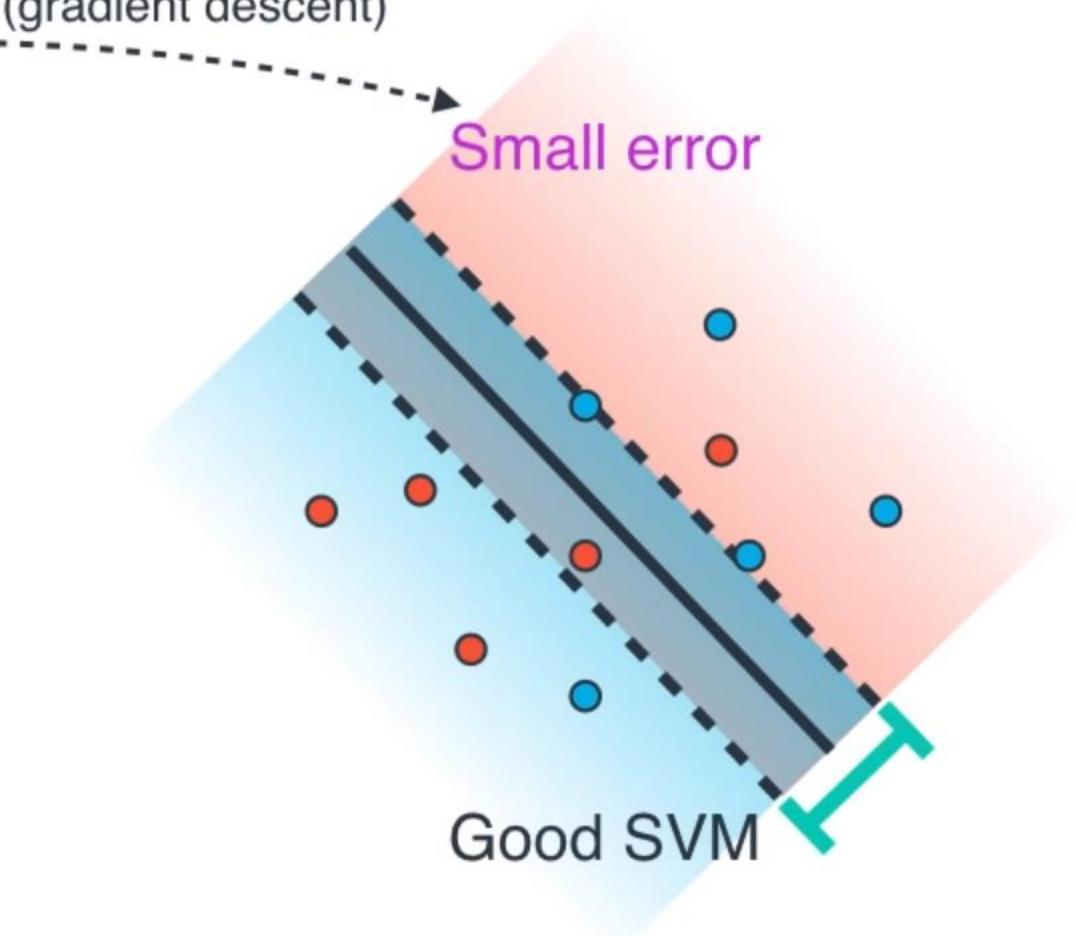
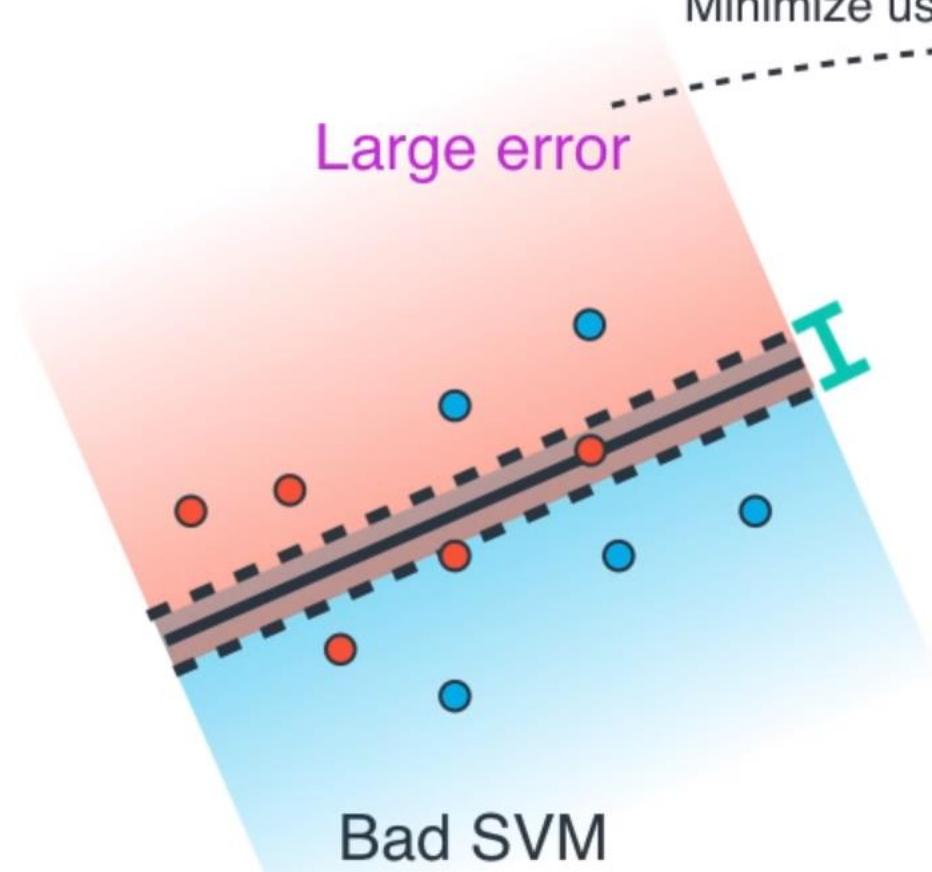
SVM Error



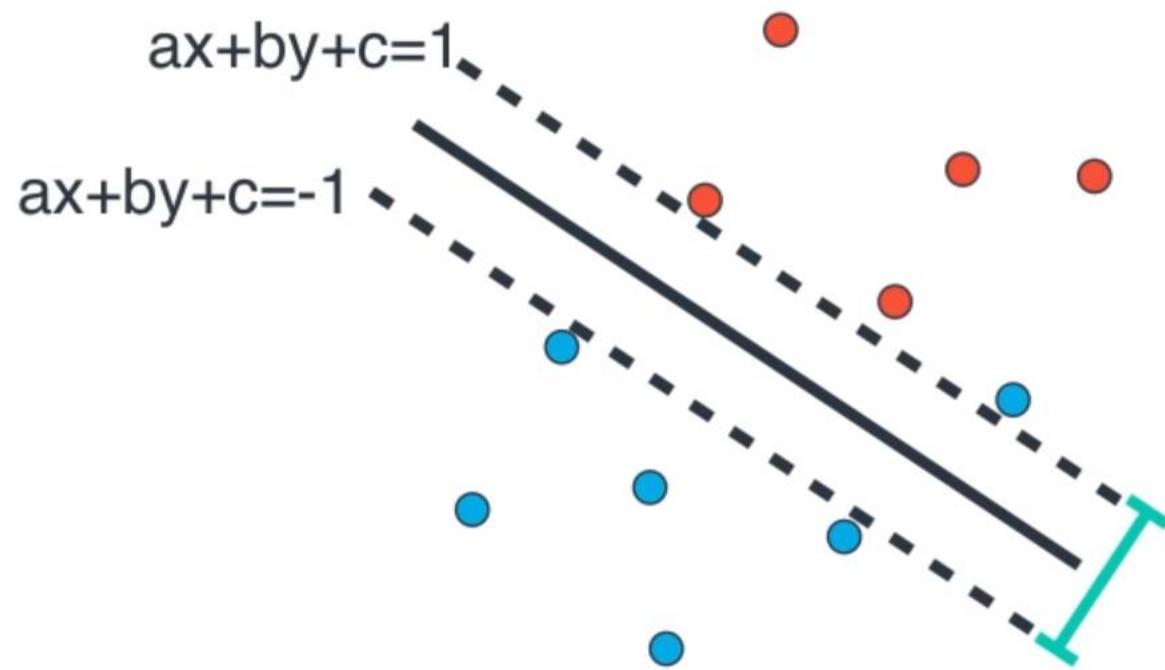
Gradient Descent

Same as the SVM trick!

Minimize using calculus (gradient descent)



Challenge - Gradient Descent



$$\text{Margin error} = a^2 + b^2$$

$$d\text{Error}/da = 2a$$

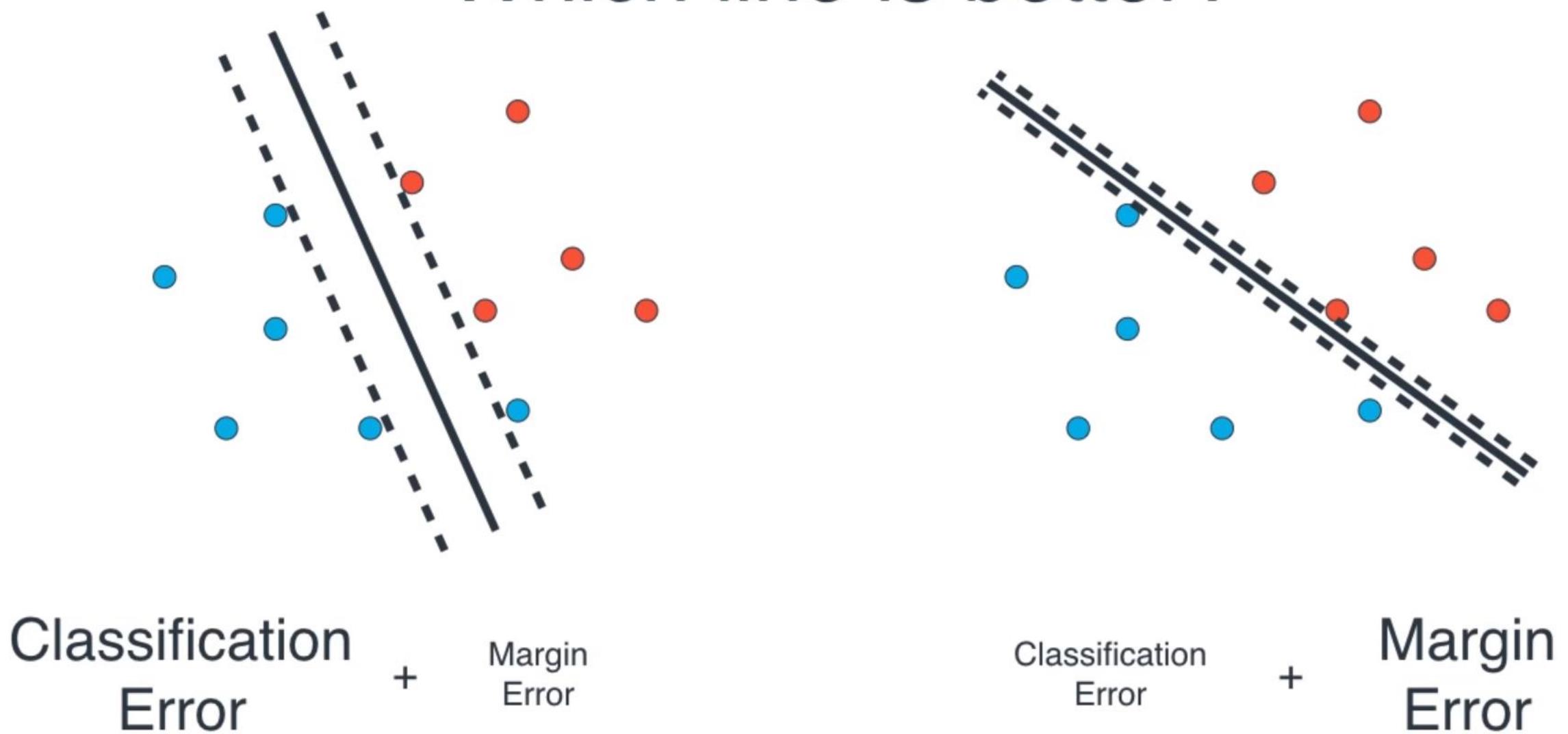
$$d\text{Error}/db = 2b$$

expanding factor!

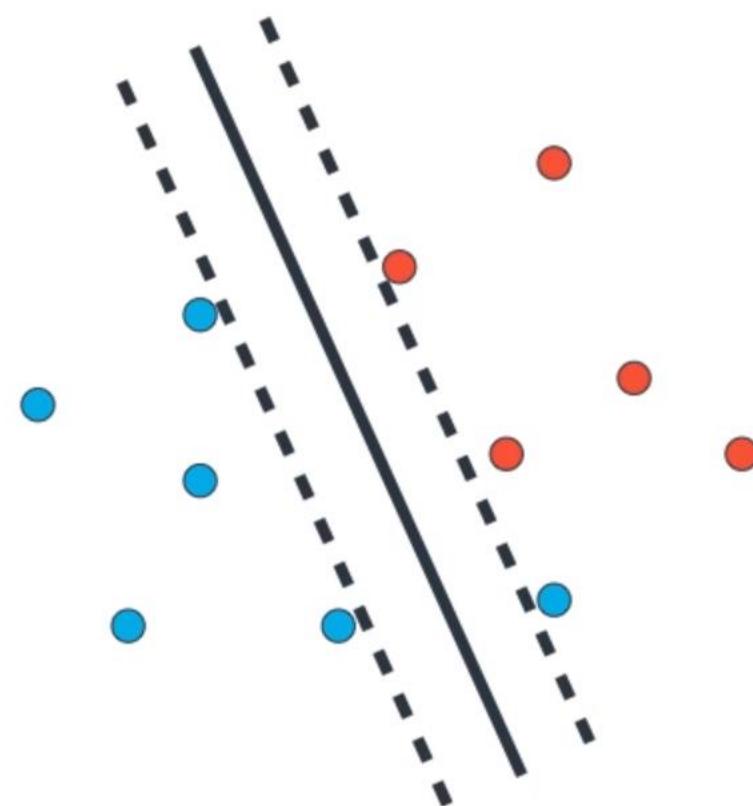
$$a \rightarrow a - \eta \quad 2a = a(1 - 2\eta)$$

$$b \rightarrow b - \eta \quad 2b = b(1 - 2\eta)$$

Which line is better?



The C parameter



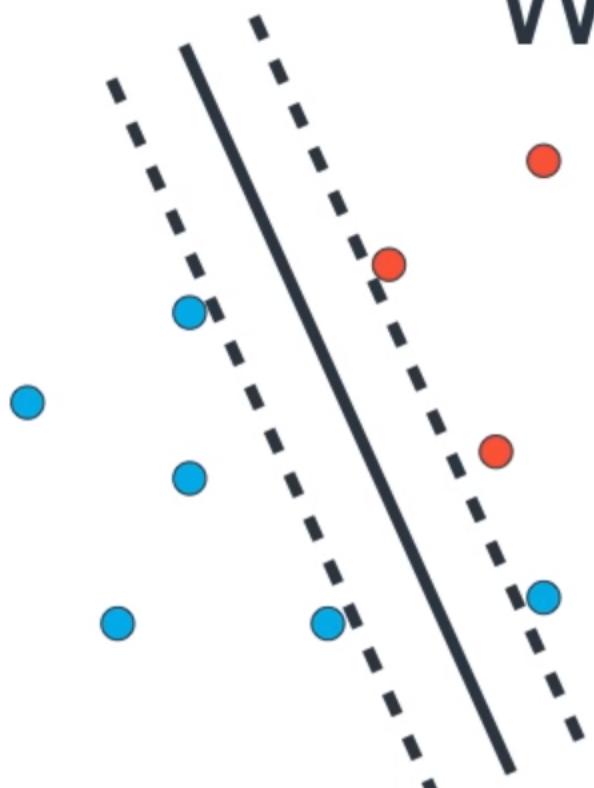
C

Classification
Error

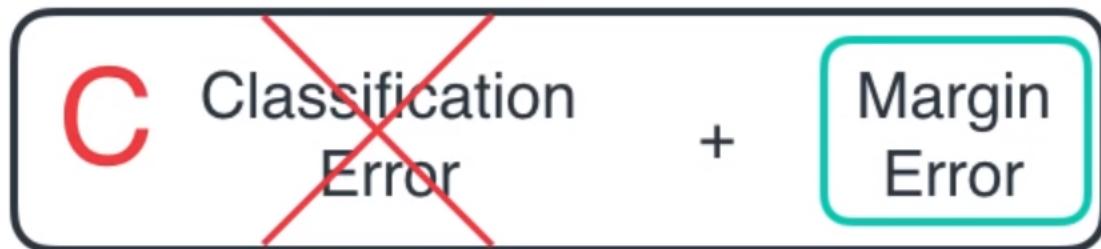
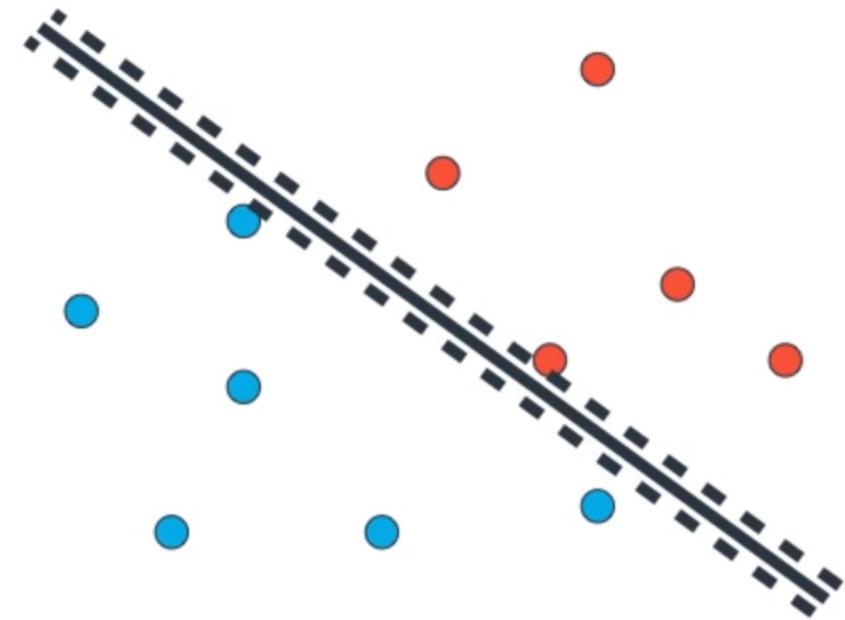
+

Margin
Error

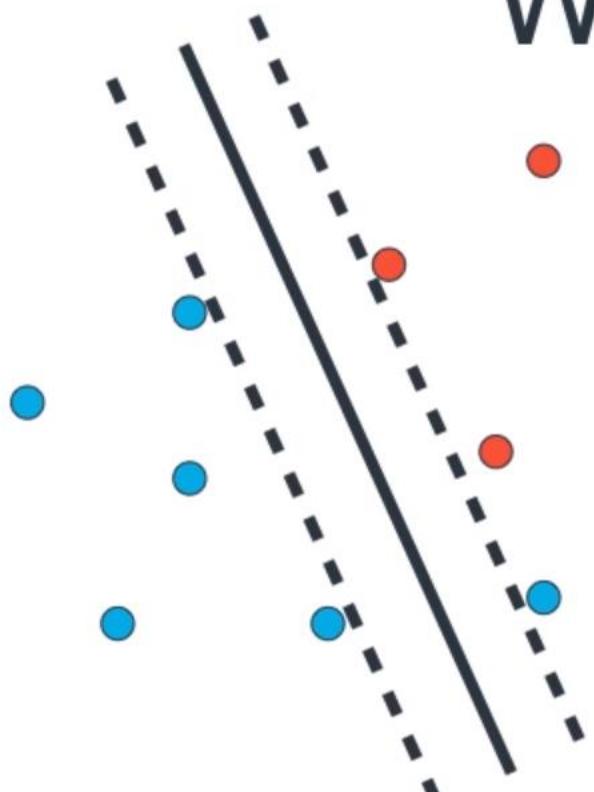
Which line is better?



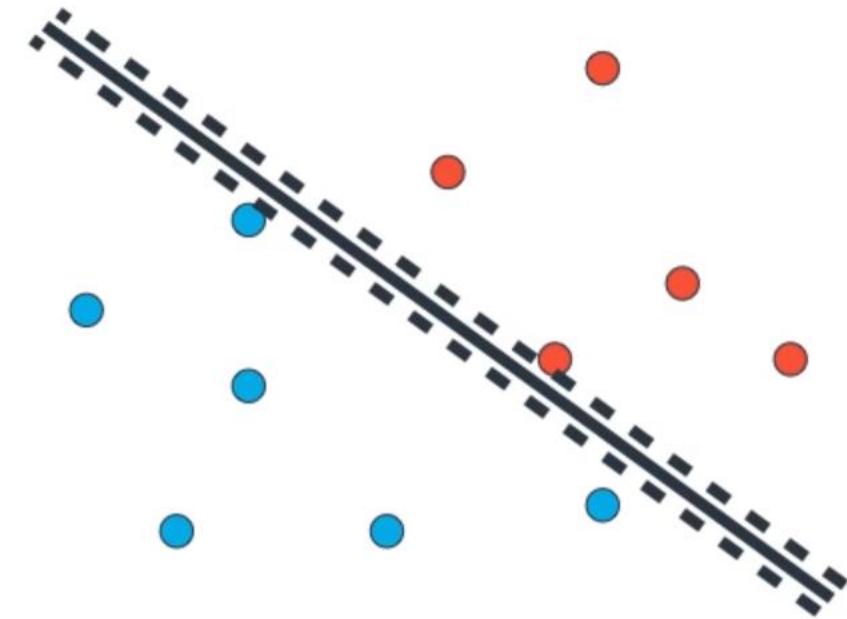
Small C
Focus on margin



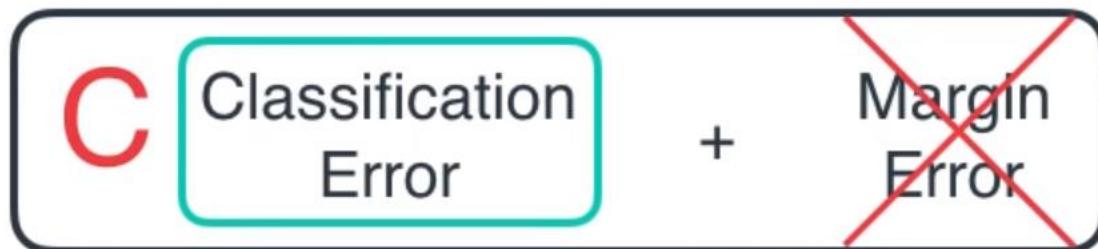
Which line is better?



Small C
Focus on margin



Large C
Focus on classification



Tune the hyper parameter and experiment which works best for your data!

Credits:

The presentation has been taken from the Luis Serrano's explanation on SVM.