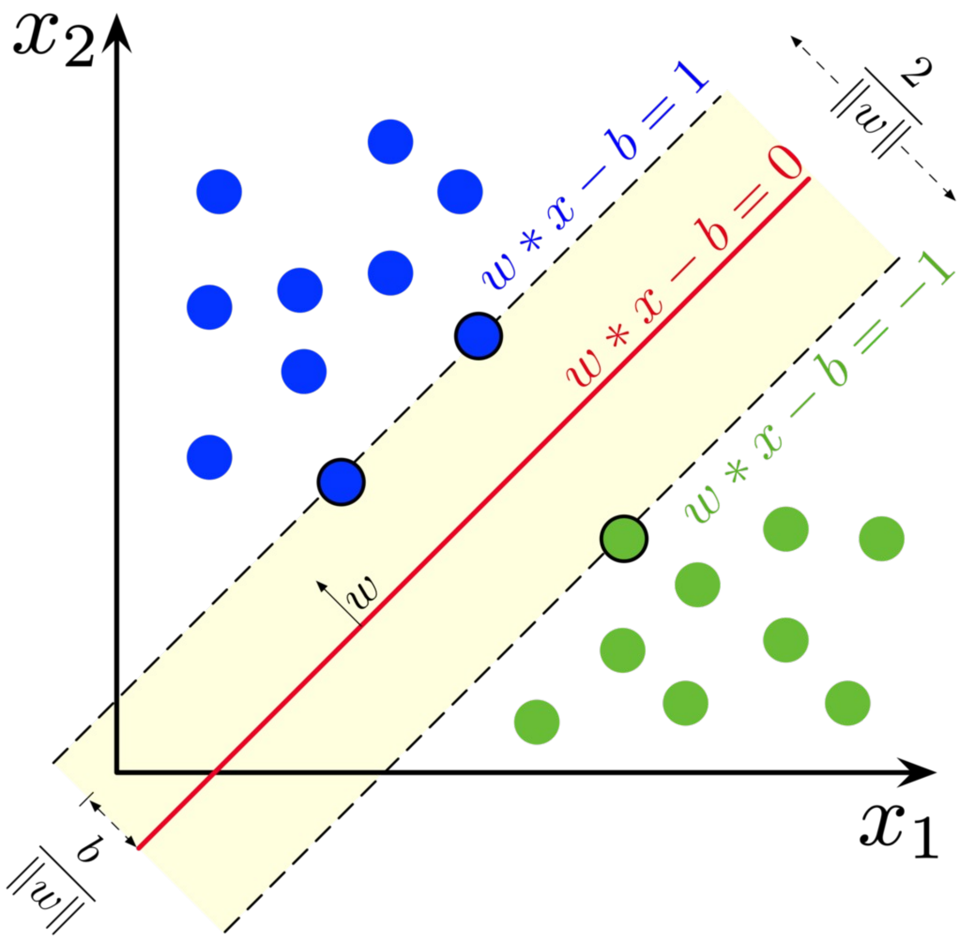
Part 7.9

An application: Support Vector Machines

Classification of data

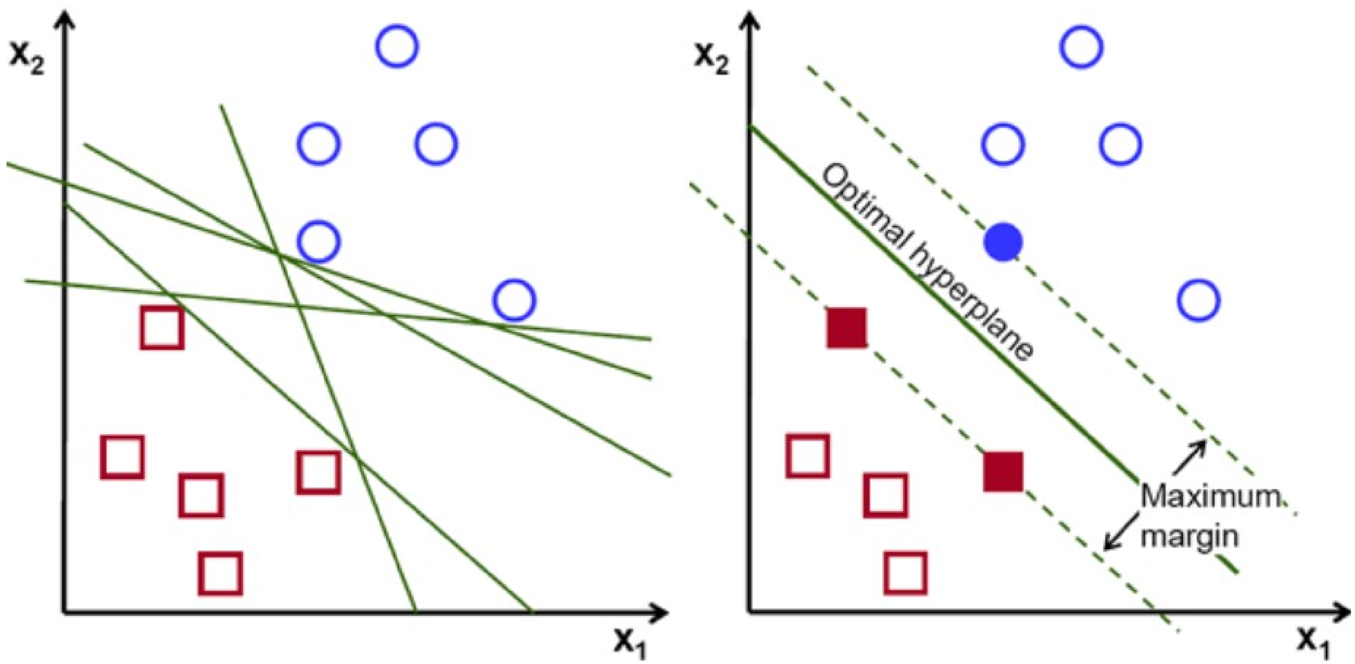
Goal: "Supervised learning" – given two data sets of different objects, find a way to "classify" any new points by finding a "separator".



Source: Wikipedia

Classification of data

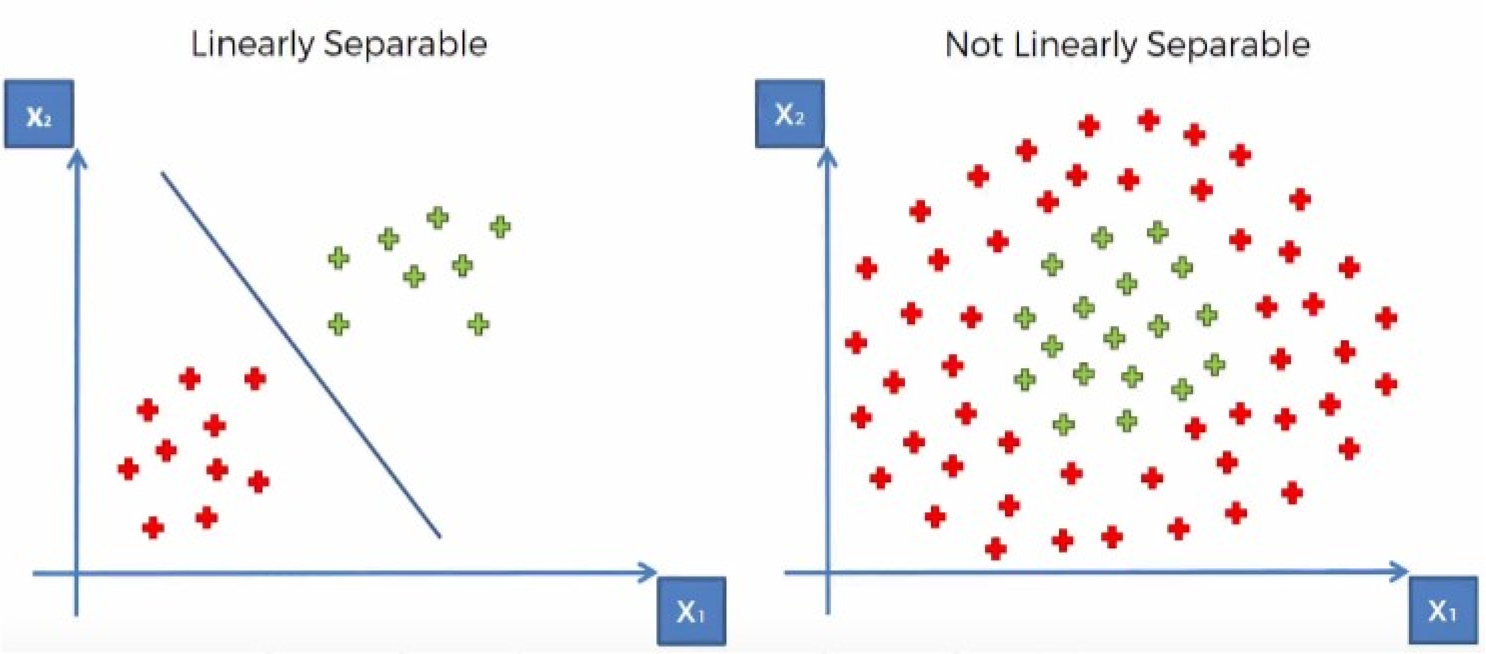
Issue 1: There may be many separators, even just among the linear ones.



Source: https://towardsdatascience.com/support-vector-machine-vs-logistic-regression-94cc2975433f

Classification of data

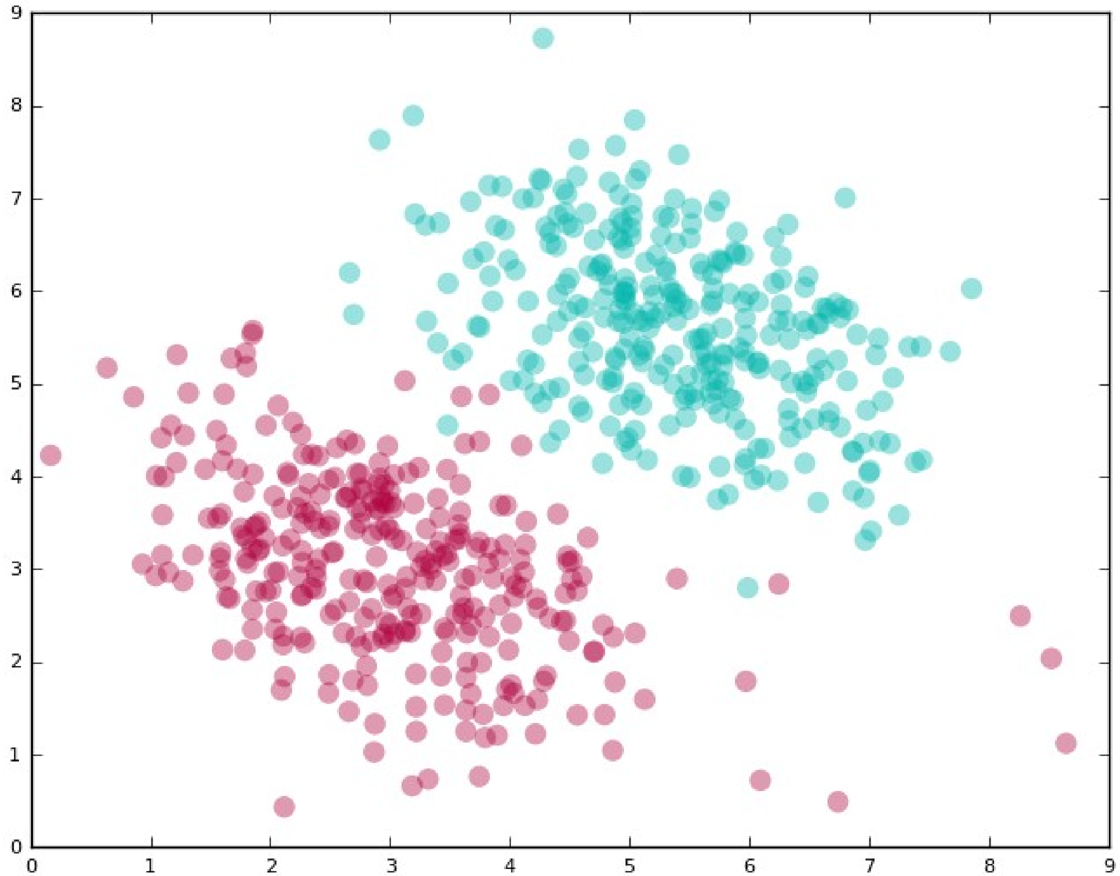
Issue 2: There may be no *linear* classifier, even though there may be nonlinear ones.



Source: Medium.com

Classification of data

Issue 3: It may not be reasonable to find a *perfect* classifier.



Source: blog.statsbot.co

Classification of data

Approach: Classification is an optimization problem!

In the simplest case: We are looking for a straight line that *minimizes the number of misclassifications.*

Classification of data

Goal: We are looking for a straight line that *minimizes the number of misclassifications.*

Formulation: Assume we have data points (x_i, y_i):

- x_i are the coordinates of the points.
- y_i is +1 if the point is part of data set 1
- y_i is -1 if the point is part of data set 2

Parameterization of the straight-line classifier:

- w is a direction vector
- b is a multiplier.

The straight line is given by $w.x - b = 0$.

Classification of data

Parameterization of the straight-line classifier:

- *w* is a direction vector
- *b* is a multiplier.

The straight line is given by $w.x - b = 0$.

Then:

- A point on the "near" side of the line has $w.x - b < 0$
- A point on the "far" side of the line has $w.x - b > 0$

Want:

- Data set 1 ($y=+1$) to be on the "near" side
- Data set 2 ($y=-1$) to be on the "far" side

Hard counting

Have:

- A point on the "near" side of the line has $w.x - b < 0$
- A point on the "far" side of the line has $w.x - b > 0$

Want:

- Data set 1 ($y=+1$) to be on the "near" side
- Data set 2 ($y=-1$) to be on the "far" side

Then optimize by counting misclassified points:

$$\text{minimize}_{w,b} \quad f(w,b) = \sum_i \chi\left(y_i\left(w\cdot x_i - b\right)\right)$$

$$\text{with } \chi(z) = 1 \quad \text{if } z > 0,$$
$$\chi(z) = 0 \quad \text{if } z \leq 0$$

Hard counting

Optimize by counting misclassified points:

$$\text{minimize}_{w,b} \quad f(w,b) = \sum_i \chi\left(y_i\left(w\cdot x_i - b\right)\right)$$

$$\text{with } \chi(z)=1 \quad \text{if } z>0, \qquad \chi(z)=0 \quad \text{if } z\leq 0$$

Problems:

- *f(w,b)* is integer valued → not smooth, not even continuous
- Typically many solutions (and maybe none with *f(w,b)=0*).

Hard counting

Observation: The formulation has too many parameters!

We could equally well have described the separating line via

$$w'.x - 1 = 0.$$

What was the purpose of introducing *b*?

Answer:

We actually want a whole separating region, i.e., we'd like it if

- A point on the "near" side of the line has $w'.x - (1-c) < 0$
- A point on the "far" side of the line has $w'.x - (1+c) > 0$

with c as large as possible. Equivalently: We want that

- A point on the "near" side of the line has $w.x - b < -1$
- A point on the "far" side of the line has $w.x - b > +1$

with $||w||$ as small as possible.

Hard counting

This leads to:

$$\text{minimize}_{w,b} \quad f(w,b) = \|w\|^2$$

$$\text{subject to} \qquad y_i(w \cdot x_i - b) \le -1$$

How is this now?

- *f(w,b)* is smooth and convex in *w*
- The constraints are linear in *w* and *b*
- There may or may not be a solution, depending on where data points lie

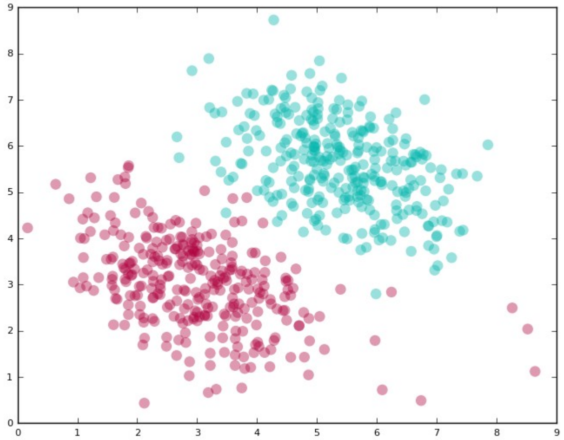
Soft counting

Hard counting used for the formulation:

$$\text{minimize}_{w,b} \quad f(w,b) = \|w\|^2$$

$$\text{subject to} \qquad y_i(w \cdot x_i - b) \leq -1$$

But what do we do in this situation:



Here, no line parameterized by *(w,b)* can satisfy all constraints!

Soft counting

Penalize how many points are on the wrong side and by how much:

Replace:

$$\text{minimize}_{w,b} \quad f(w,b) = \|w\|^2$$

$$\text{subject to} \quad y_i(w\cdot x_i - b) \leq -1$$

By:

$$\text{minimize}_{w,b} \quad f(w,b) = \frac{1}{N}\sum_i \max\left[0, 1 - y_i(w\cdot x_i - b)\right]$$

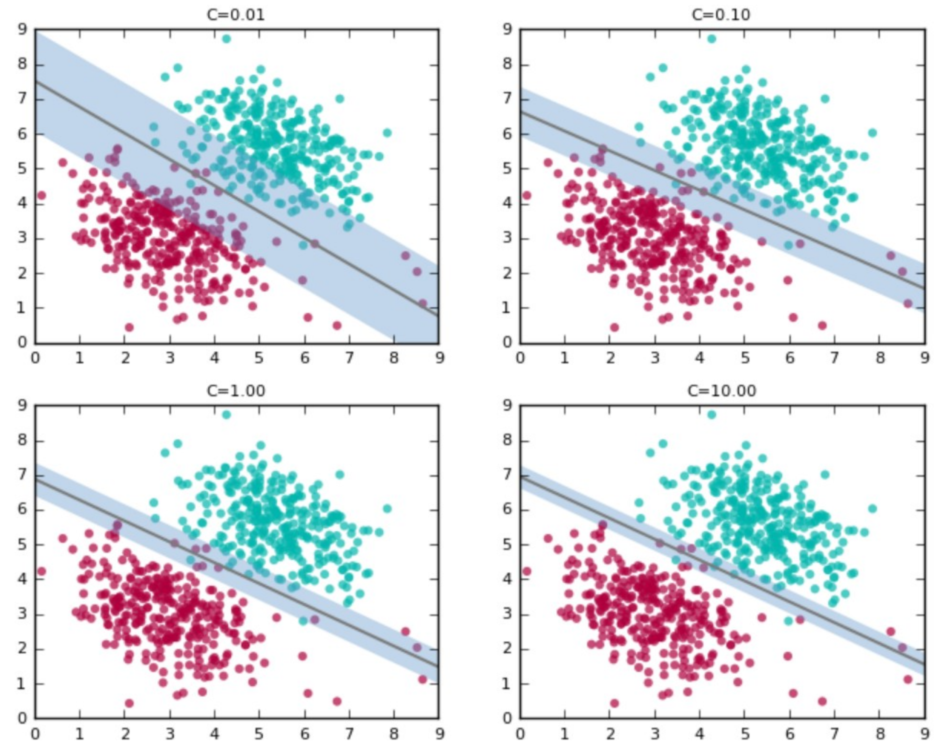
Soft counting

Make the gap big again:

$$\text{minimize}_{w,b} \quad f(w,b) = \left(\frac{1}{N} \sum_i \max\left[0, 1 - y_i (w \cdot x_i - b)\right] \right) + \lambda \|w\|^2$$

λ states what we value more:

- A big gap (lambda large)
- Fewer points on the "wrong" side (lambda small)



Soft counting

Make the gap big again:

$$\text{minimize}_{w,b} \quad f(w,b) = \left(\frac{1}{N} \sum_i \max\left[0, 1 - y_i (w \cdot x_i - b)\right] \right) + \lambda \|w\|^2$$

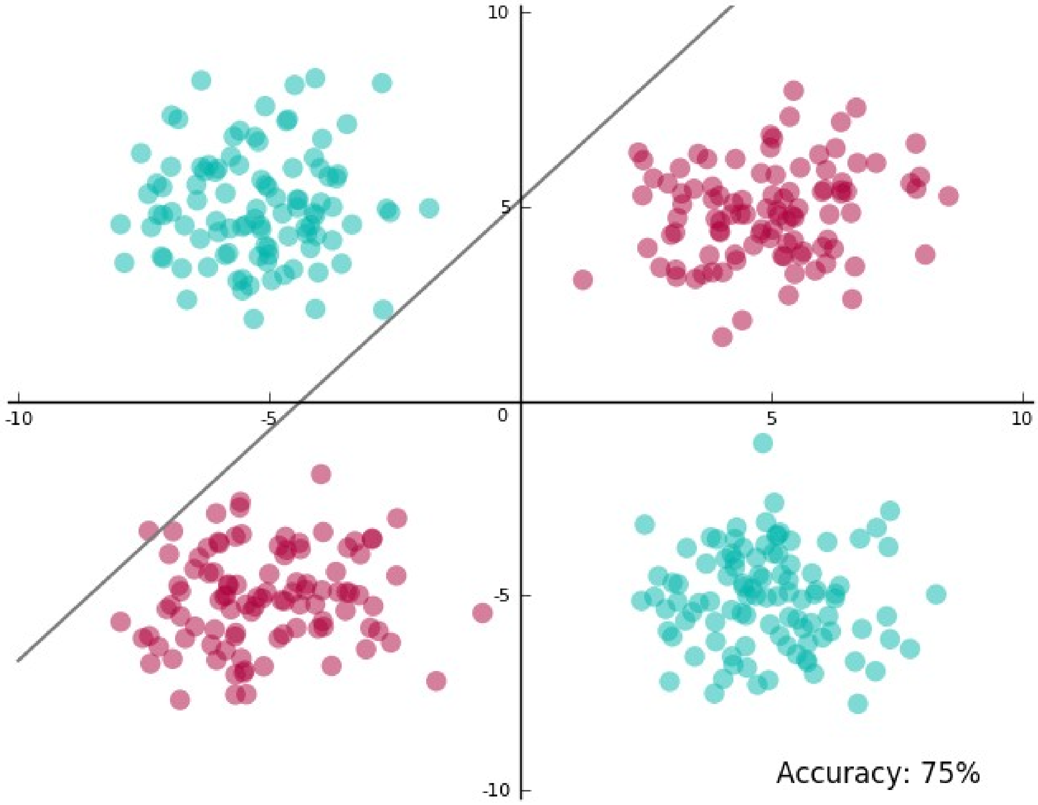
This formulation is non-smooth. It can be reformulated as a smooth, constrained problem using slack variables:

$$\text{minimize}_{w,b,s} \quad f(w,b,s) = \frac{1}{N} \sum_i s_i + \lambda \|w\|^2$$
$$s_i \geq 0,$$
$$s_i \geq 1 - y_i (w \cdot x_i - b)$$

This is called a *linear-quadratic problem*. They are easy to solve!

Nonlinear classifiers

In practice, data points can often not be separated by a straight line:



In such cases, one needs *nonlinear* classifiers. These are computed by *transforming* the data set $x \rightarrow g(x)$.