Support Vector Machines

Intro to ML 466/566
Fall 2022

Many of these slides are derived from Seyong Kim and Alex Thomo. Thanks!

Administrivia

- As 2 will be out today
  - Due Nov 17 (was originally supposed to be Nov 15)

Linear classifier (E.g., Perceptron)

$$h(\mathbf{x}) = sign(\mathbf{w} \cdot \mathbf{x} + b)$$

Which outputs $+1$ or -1.

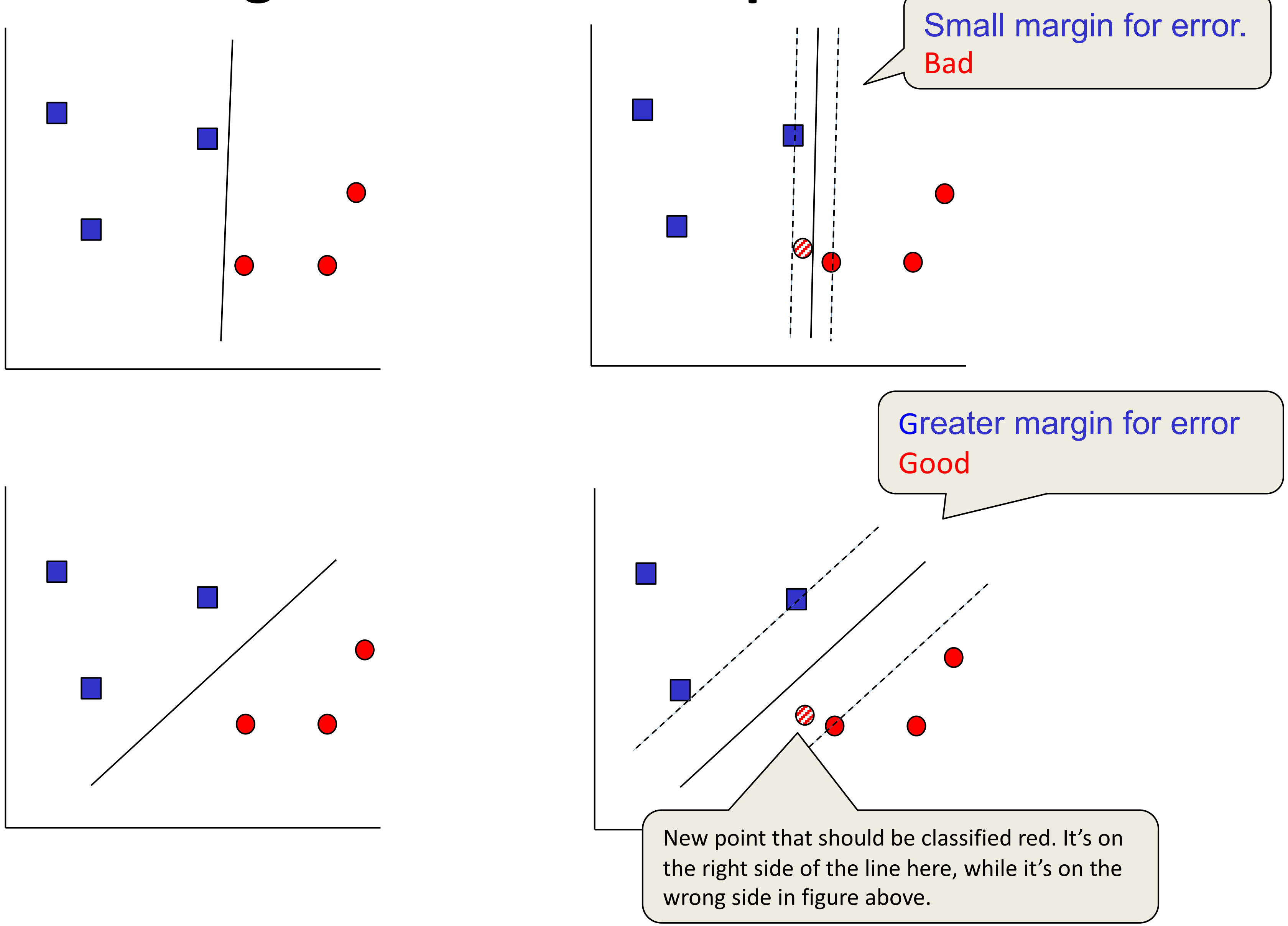
Say:

$+1$ corresponds to blue, and

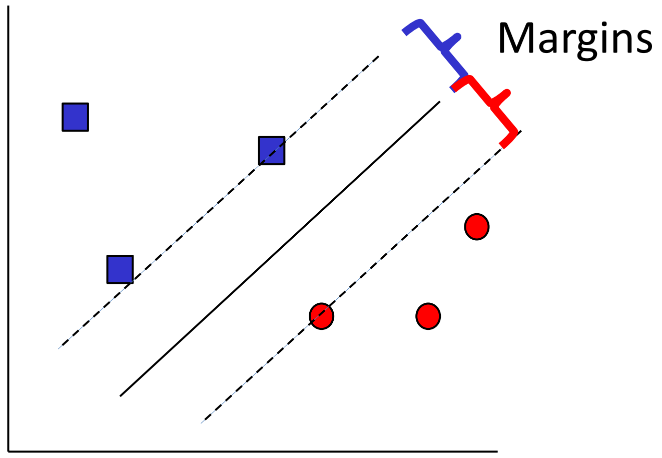
-1 to red, or vice versa.

Many lines do the job.
Which one to choose?

Margin for different separators



What is a margin?



Margin is the distance between:

- a point (■, ●)
- and the decision plane

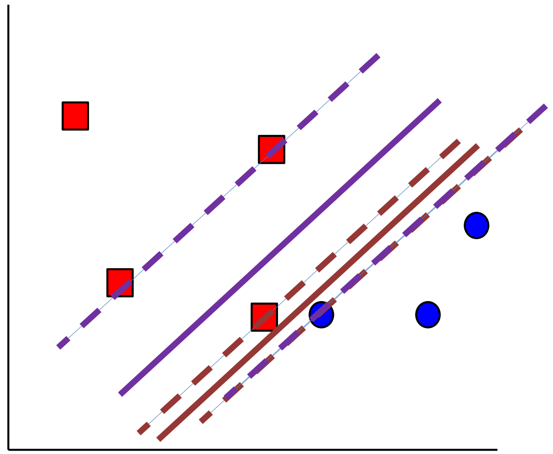
We want to find parameters of a line that: ■ ●

1. Maximizes the margin for the closest of both point types
2. Correctly classify all points

What about outliers?

We want to find parameters of a line that:

1. Maximizes the margin for both point types
2. Correctly classify all points



There is a trade off to be made!

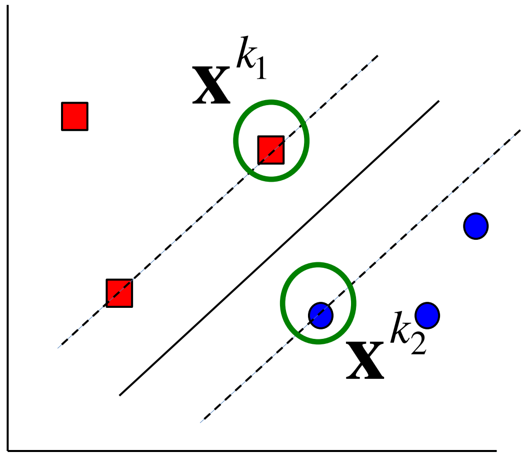
Support Vectors

We want to find parameters of a line that:

1. **Maximizes the margin for the closest of both point types**
2. Correctly classify all points

Because of point 1, we will always have (at least) one point on each of the +1 and -1 lines

Closest points are called "support vectors".



Scale Invariance

$$h(\mathbf{x}) = sign(\mathbf{w} \cdot \mathbf{x} + b)$$

- We rescale **w** and *b* (without changing the line) such that:

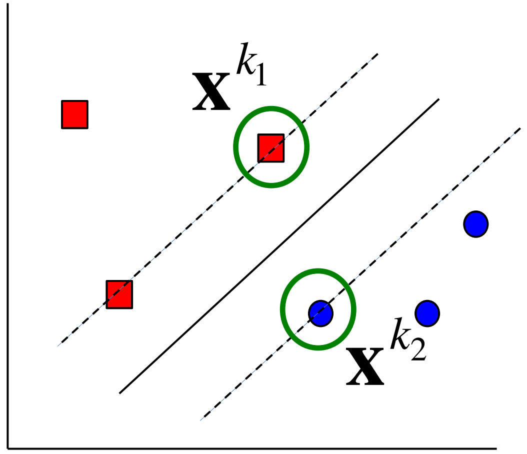
$$\mathbf{w} \cdot \mathbf{x}^{k_1} + b = 1$$

for the closest point(s) to the line on the +1 side, and

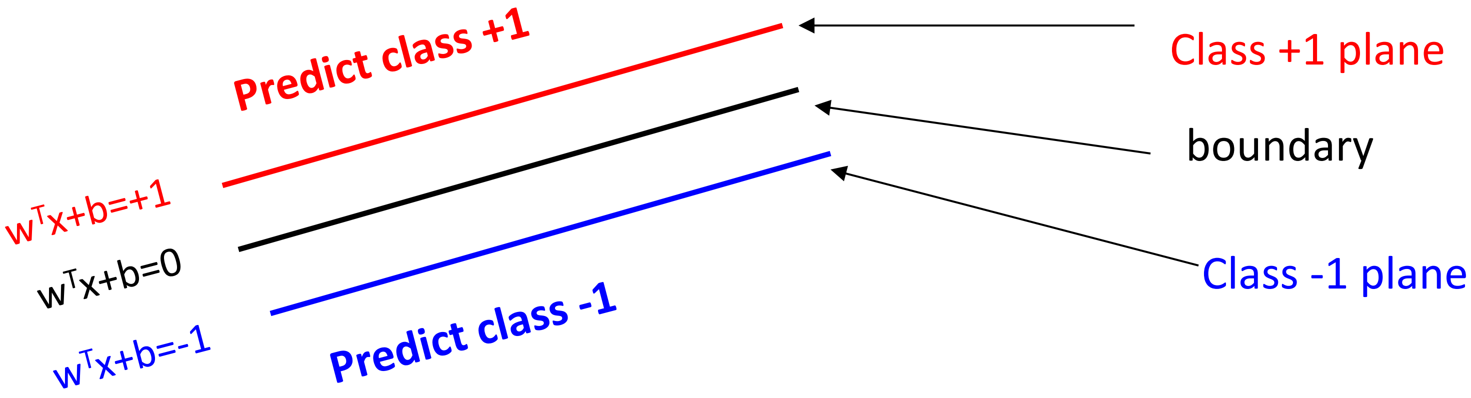
$$\mathbf{w} \cdot \mathbf{x}^{k_2} + b = -1$$

for the closest point(s) to the line on the -1 side.

Closest points are called "support vectors".



Specifying a max margin classifier

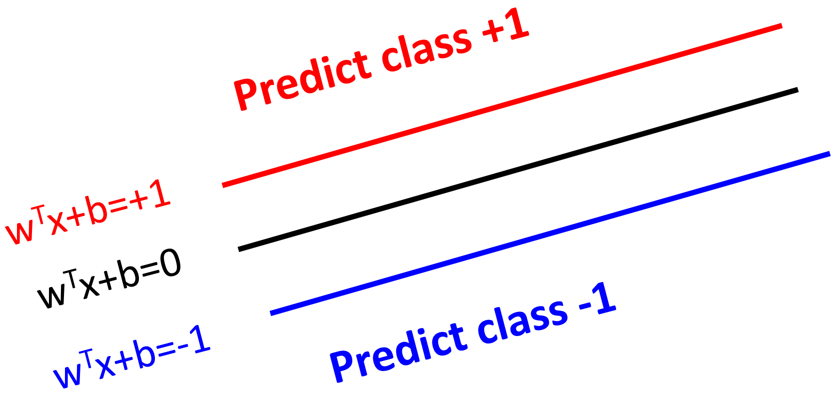


Classify as +1	if	$w^Tx+b \geq 1$
Classify as -1	if	$w^Tx+b \leq -1$
Undefined	if	$-1 < w^Tx+b < 1$

Learn one w and b to satisfy these equations

In practice, at test time we would still give +1, -1 classes to the undefined points based on the sign. At train time, we are setting up the problem so no training points fall into this intermediate zone.

Specifying a max margin classifier

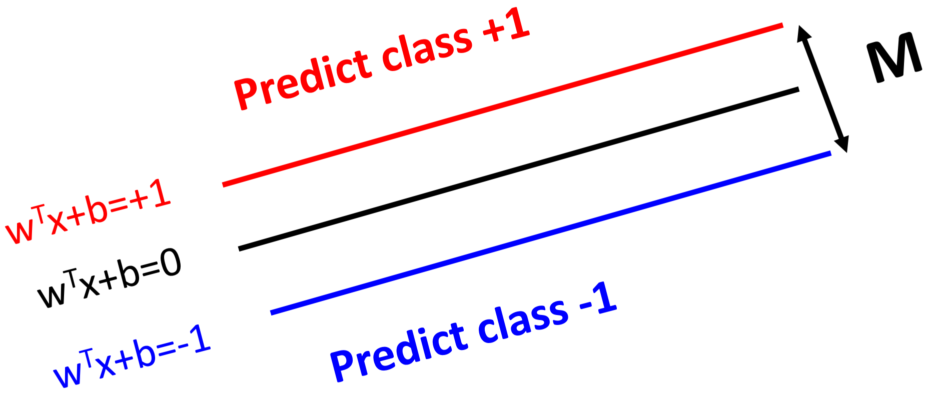


Is the linear separation assumption realistic?

We will deal with this shortly, but lets assume it is for now

Classify as +1	if	$w^Tx+b \geq 1$
Classify as -1	if	$w^Tx+b \leq -1$
Undefined	if	$-1 < w^Tx+b < 1$

Maximizing the margin



Classify as +1 if $w^Tx+b \geq 1$

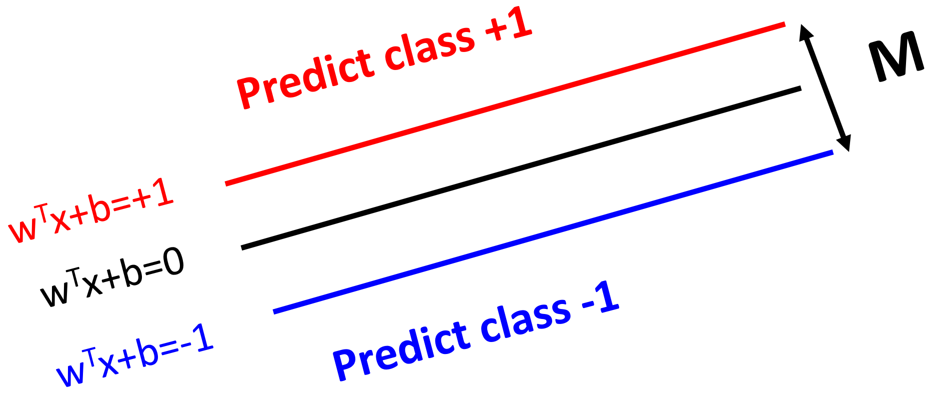
Classify as -1 if $w^Tx+b \leq -1$

Undefined if $-1 < w^Tx+b < 1$

- Lets define the width of the margin by M
- How can we encode our goal of maximizing M in terms of our parameters (w and b)?
- Lets start with a few observations

To the handwritten notes!

Maximizing the margin



Classify as +1 if $w^Tx+b \geq 1$

Classify as -1 if $w^Tx+b \leq -1$

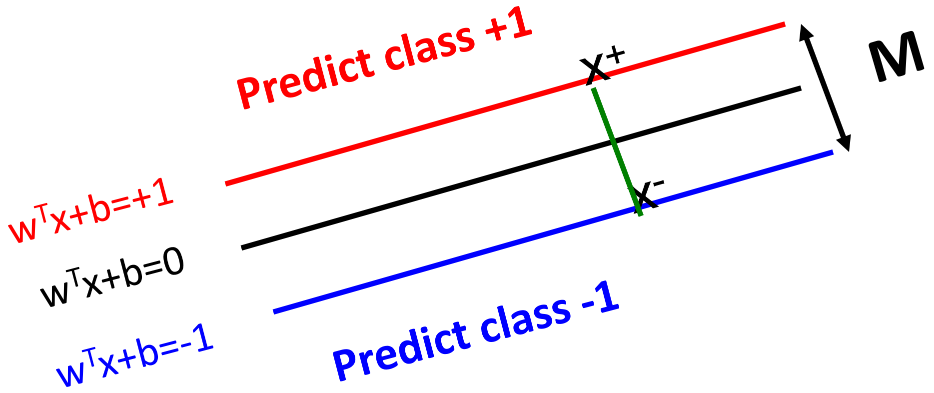
Undefined if $-1 < w^Tx+b < 1$

- Observation 1: the vector w is orthogonal to the +1 plane
- Why?

> Let u and v be two points on the +1 plane, then for the vector defined by u and v we have $w^T(u-v) = 0$

> Corollary: the vector w is orthogonal to the -1 plane

Maximizing the margin



Classify as +1 if $w^Tx+b \geq 1$

Classify as -1 if $w^Tx+b \leq -1$

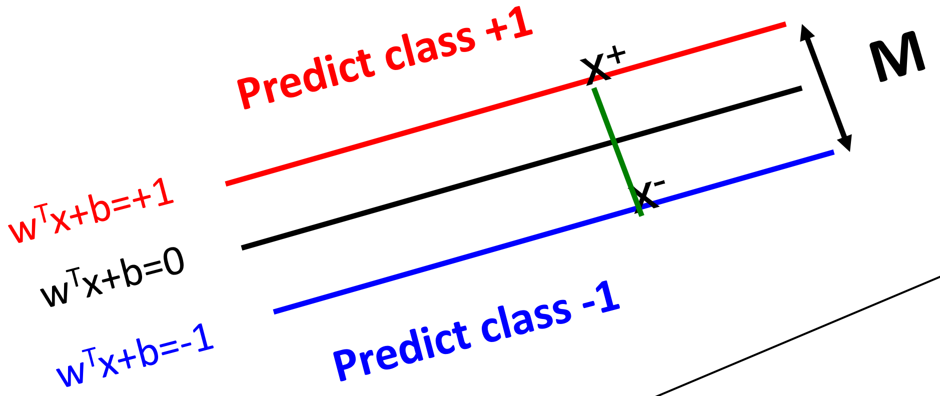
Undefined if $-1 < w^Tx+b < 1$

- Observation 1: the vector w is orthogonal to the +1 and -1 planes
- Observation 2: if x^+ is a point on the +1 plane and x^- is the closest point to x^+ on the -1 plane then

$$x^+ = \lambda w + x^-$$

Since w is orthogonal to both planes we need to 'travel' some distance along w to get from x^+ to x^-

Putting it together



- $w^T x^+ + b = +1$
- $w^T x^- + b = -1$
- $x^+ = \lambda w + x^-$
- $| x^+ - x^- | = M$

We can now define M in terms of w and b

$w^T x^+ + b = +1$

$\Rightarrow$

$w^T (\lambda w + x^-) + b = +1$

$\Rightarrow$

$w^T x^- + b + \lambda w^T w = +1$

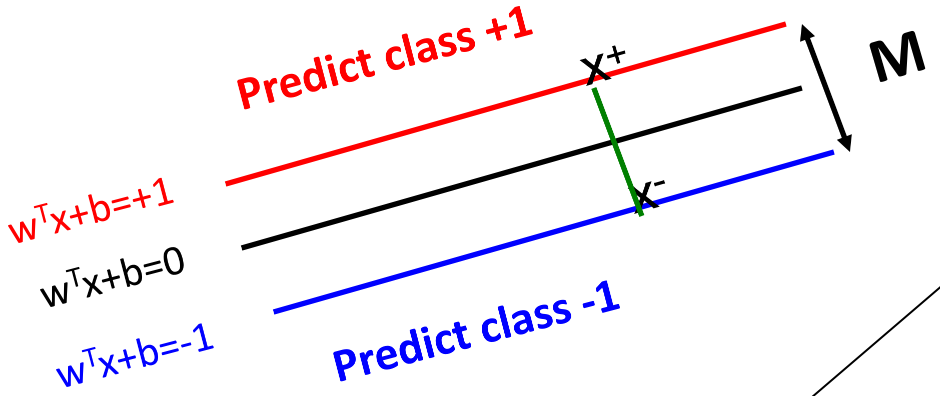
$\Rightarrow$

$-1 + \lambda w^T w = +1$

$\Rightarrow$

$\lambda = 2/w^T w$

Putting it together



- $w^T x^+ + b = +1$
- $w^T x^- + b = -1$
- $x^+ = \lambda w + x^-$
- $| x^+ - x^- | = M$
- $\lambda = 2/w^Tw$

We can now define M in terms of w and b

$$M = |x^+ - x^-|$$

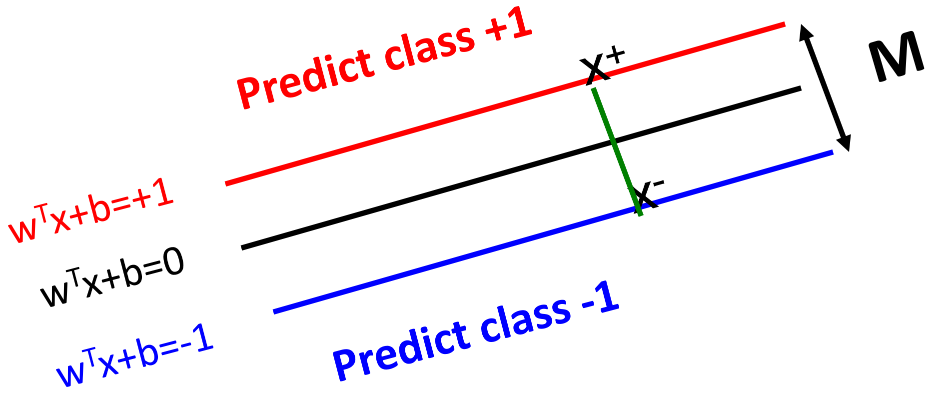
$$\Rightarrow$$

$$M = |\lambda w| = \lambda |w| = \lambda \sqrt{w^T w}$$

$$\Rightarrow$$

$$M = 2\frac{\sqrt{w^T w}}{w^T w} = \frac{2}{\sqrt{w^T w}}$$

Finding the optimal parameters



$$M = \frac{2}{\sqrt{w^T w}}$$

We can now search for the optimal parameters by finding a solution that:

1. Correctly classifies all points
2. Maximizes the margin (or equivalently minimizes w^Tw)

Several optimization methods can be used:
Gradient descent, simulated annealing, EM etc.

Aside: Quadratic programming (QP)

Quadratic programming solves optimization problems of the following form:

$$\min_U \frac{u^T R u}{2} + d^T u + c$$

subject to n inequality constraints:

$$a_{11}u_1 + a_{12}u_2 + \ldots \leq b_1$$

$$\vdots \qquad \vdots \qquad \vdots$$

$$a_{n1}u_1 + a_{n2}u_2 + \ldots \leq b_n$$

and k equivalency constraints:

$$a_{n+1,1}u_1 + a_{n+1,2}u_2 + \ldots = b_{n+1}$$

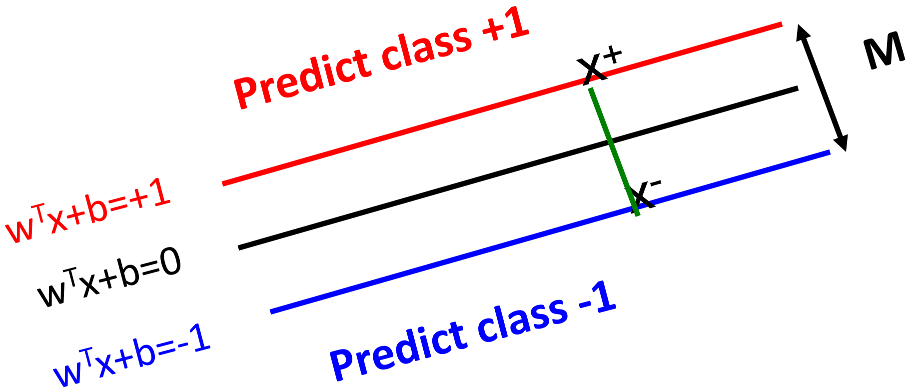
$$\vdots \qquad \vdots \qquad \vdots$$

$$a_{n+k,1}u_1 + a_{n+k,2}u_2 + \ldots = b_{n+k}$$

Quadratic term

When a problem can be specified as a QP problem we can use solvers that are better than gradient descent or simulated annealing

SVM as a QP problem



$$M = \frac{2}{\sqrt{w^T w}}$$

Min $(w^Tw)/2$

subject to the following inequality constraints:

For all x in class + 1

$w^Tx+b \geq 1$

For all x in class - 1

$w^Tx+b \leq -1$

A total of n constraints if we have n input samples

$$\min_U \frac{u^T R u}{2} + d^T u + c$$

subject to n inequality constraints:

$$\begin{aligned} a_{11}u_1 + a_{12}u_2 + \ldots &\leq b_1 \\ \vdots \qquad \vdots \qquad &\vdots \\ a_{n1}u_1 + a_{n2}u_2 + \ldots &\leq b_n \end{aligned}$$

and k equivalency constraints:

$$\begin{aligned} a_{n+1,1}u_1 + a_{n+1,2}u_2 + \ldots &= b_{n+1} \\ \vdots \qquad \vdots \qquad &\vdots \\ a_{n+k,1}u_1 + a_{n+k,2}u_2 + \ldots &= b_{n+k} \end{aligned}$$

SVM as a QP problem: a simplification

Min $(w^Tw)/2$

subject to the following inequality constraints:

For all x in class + 1

$w^Tx+b \geq 1$

For all x in class - 1

$w^Tx+b \leq -1$

Min $(w^Tw)/2$

subject to the following inequality constraints:

For all x in class + 1

$y(w^Tx+b) \geq 1$

For all x in class - 1

$y(w^Tx+b) \geq 1$

The same constraint!!
So much easier to handle!

Example

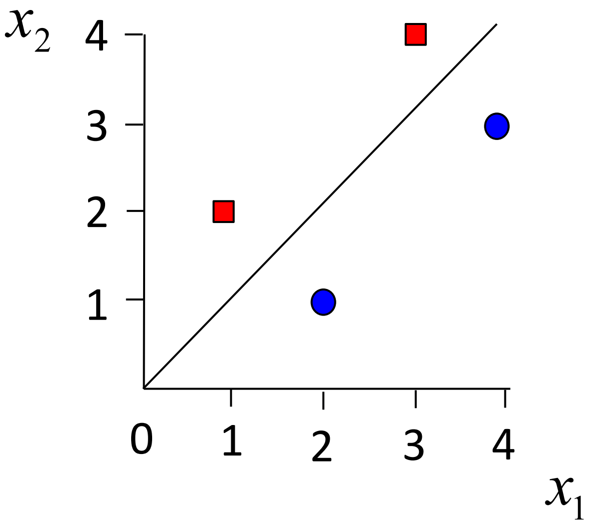
Training tuples:

([1, 2], +1)

([2, 1], −1)

([3, 4], +1)

([4, 3], −1)



$$\min_{w_1, w_2} \frac{1}{2}\left(w_1^2 + w_2^2\right)$$

subject to

$$(+1)(w_1 + 2w_2) \geq 1$$
$$(-1)(2w_1 + w_2) \geq 1$$
$$(+1)(3w_1 + 4w_2) \geq 1$$
$$(-1)(4w_1 + 3w_2) \geq 1$$

For this example, solution is easy to see:

b = 0 and w = [−1, +1], i.e. the line is:

$-x_1+x_2=0$

All conditions are satisfied.

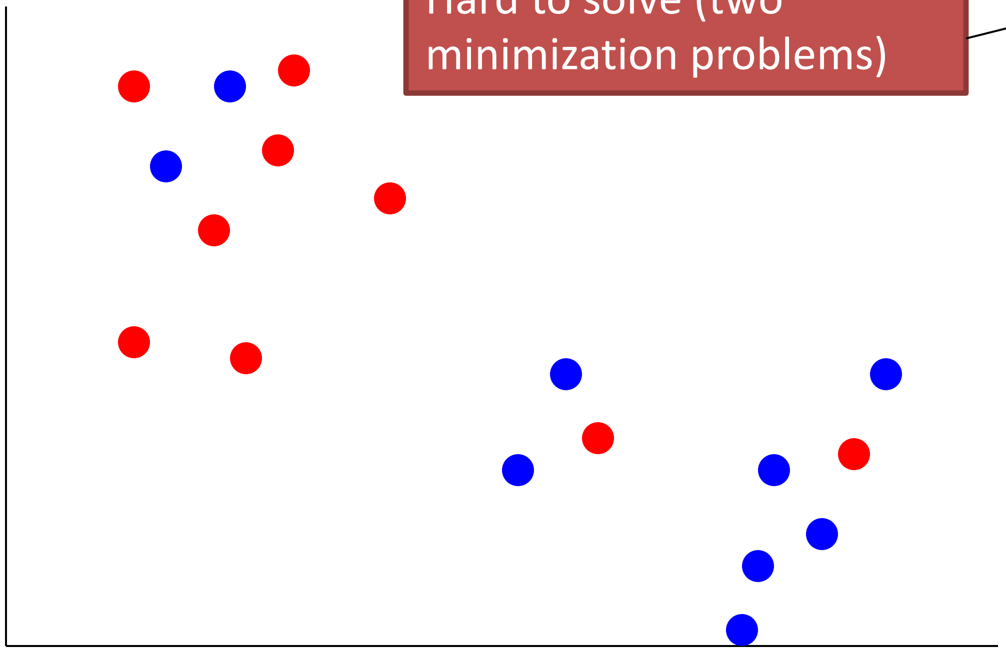
$$\mathrm{M} = \frac{2}{\|\mathbf{w}\|} = \frac{2}{\sqrt{1+1}} = \frac{2}{\sqrt{2}} = \sqrt{2}$$

Non linearly separable case

- So far we assumed that a linear plane can perfectly separate the points
- But this is not usally the case

- noise, outliers

Hard to solve (two minimization problems)



How can we convert this to a QP problem?

- Minimize training errors?

  min $w^T w$

  min #errors

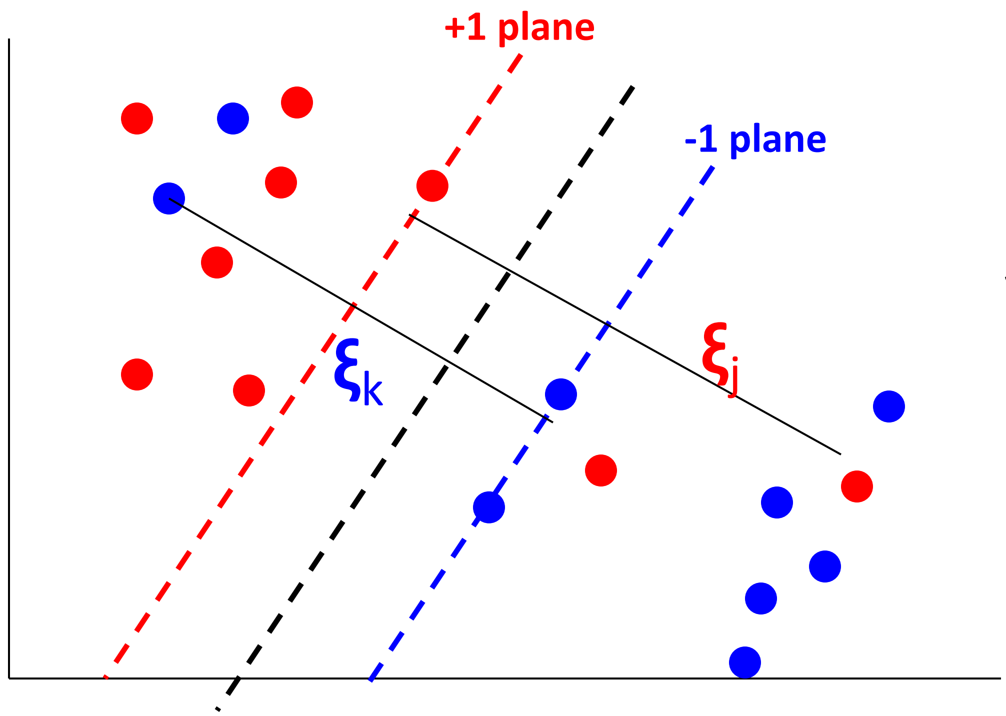
- Penalize training errors:

min $w^T w + C*(\#errors)$

Hard to encode in a QP problem

Non linearly separable case

- Instead of minimizing the number of misclassified points we can minimize the *distance* between these points and their correct plane



The new optimization problem is:

$$\min_w \frac{w^T w}{2} + \sum_{i=1}^{n} C\xi_i$$

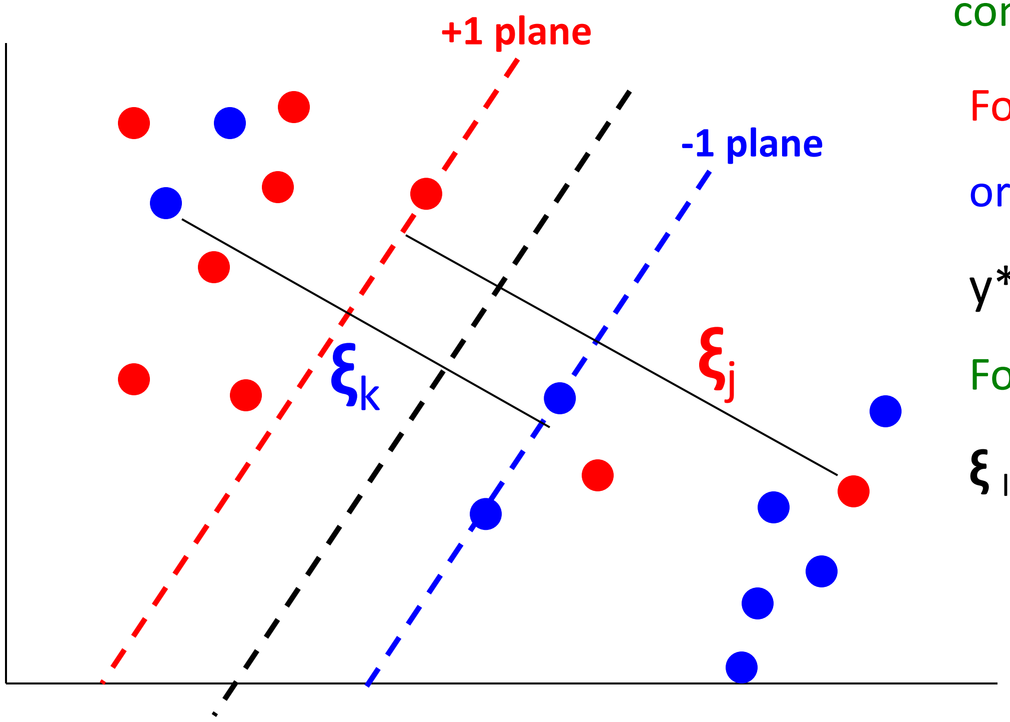
subject to the following inequality constraints:

For all x_i in class +1 or class -1

$$y^*(w^T x+b) \geq 1- \xi_i$$

Wait. Are we missing something?

Final optimization for non linearly separable case



The new optimization problem is:

$$\min_w \frac{w^T w}{2} + \sum_{i=1}^{n} C\xi_i$$

subject to the following inequality constraints:

For all x_i in class +1

or class -1

$y^*(w^Tx+b) \geq 1 - \xi_i$

n constraints

For all i

$\xi_i \geq 0$

Another n constraints

Slack variables (ε)

Where we are

Two optimization problems: For the separable and non separable cases

$$\min_w \frac{w^T w}{2}$$

For all x in class + 1

$w^T x + b \geq 1$

For all x in class - 1

$w^T x + b \leq -1$

$$\min_w \frac{w^T w}{2} + \sum_{i=1}^{n} C\xi_i$$

For all x_i in class + 1

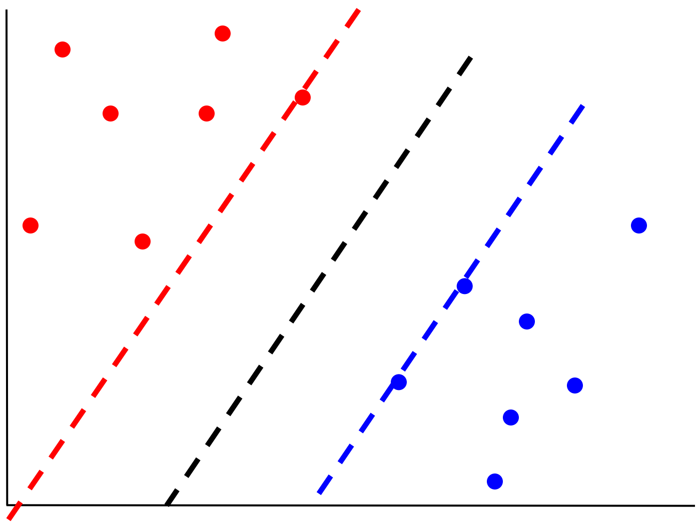
$w^T x + b \geq 1 - \xi_i$

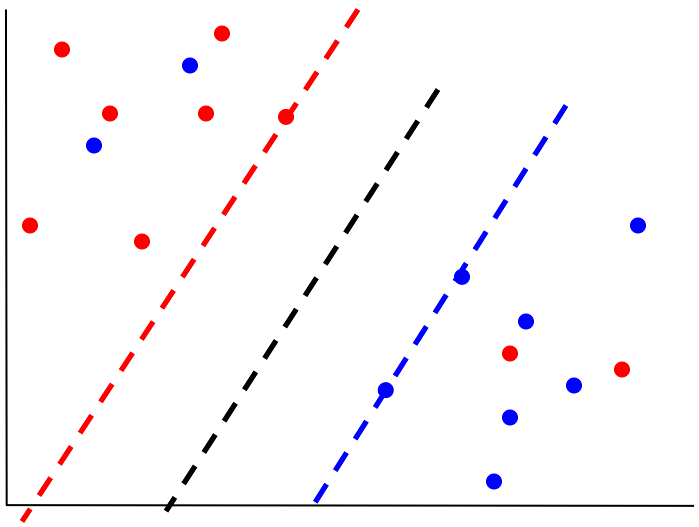
For all x_i in class - 1

$w^T x + b \leq -1 + \xi_i$

For all I

$\xi_I \geq 0$





Where we are

Two optimization problems: For the separable and non separable cases

Hard margin

$$\min_w \frac{w^T w}{2}$$

For all x in class + 1

$w^T x + b \geq 1$

For all x in class - 1

$w^T x + b \leq -1$

Soft margin

$$\min_w \frac{w^T w}{2} + \sum_{i=1}^{n} C\xi_i$$

For all x_i in class + 1

$w^T x + b \geq 1 - \xi_i$

For all x_i in class - 1

$w^T x + b \leq -1 + \xi_i$

For all I

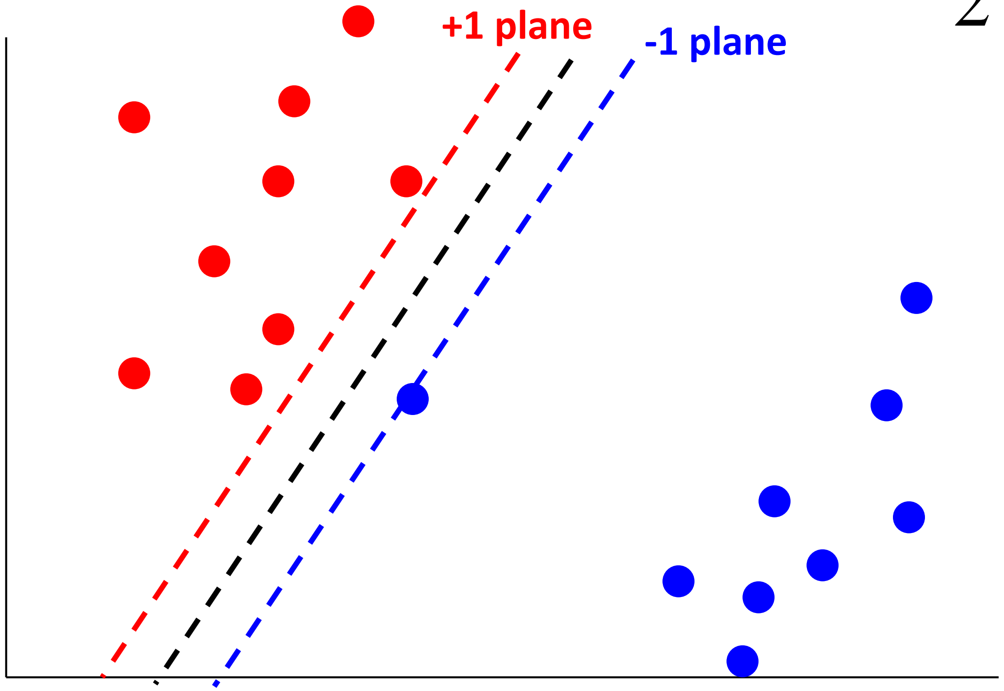
$\xi_I \geq 0$

- Instead of solving these QPs directly we will solve a **dual** formulation of the SVM optimization problem
- One reason for switching to this type of representation (the dual formulation) is that it allows us to use a neat trick that will make our lives easier (and the run time faster)

Can you think of a case where the data is linearly separable, but a slack variable will be non-zero?

Can you think of a case where the data is linearly separable, but a slack variable will be non-zero?

$$\min_w \frac{w^T w}{2} + \sum_{i=1}^{n} C\xi_i$$



Some example code

https://colab.research.google.com/drive/1QRxaGk_YZaKeO0Q3S42NKdi4g2dGuTD1?usp=sharing

Tuesday* we learned the primal form
Today we will derive the dual form

What is the dual?
It is a reformulation of an optimization problem
It provides a lower bound on the primal minimization problem
When the solution to the primal is different from the solution to the dual, there is a duality gap

*or whenever you watched the video

To the handwritten notes!

Kernels

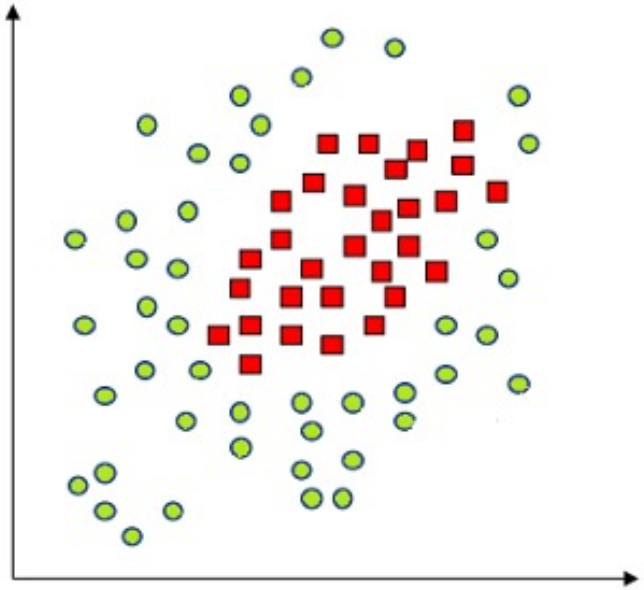
- In real life, data is often not linearly separable
  - We saw that slack variables can help when it’s *nearly* separable

- Is there anything else we can do when the data is not linearly separable?

- Yes! Use a **kernel**

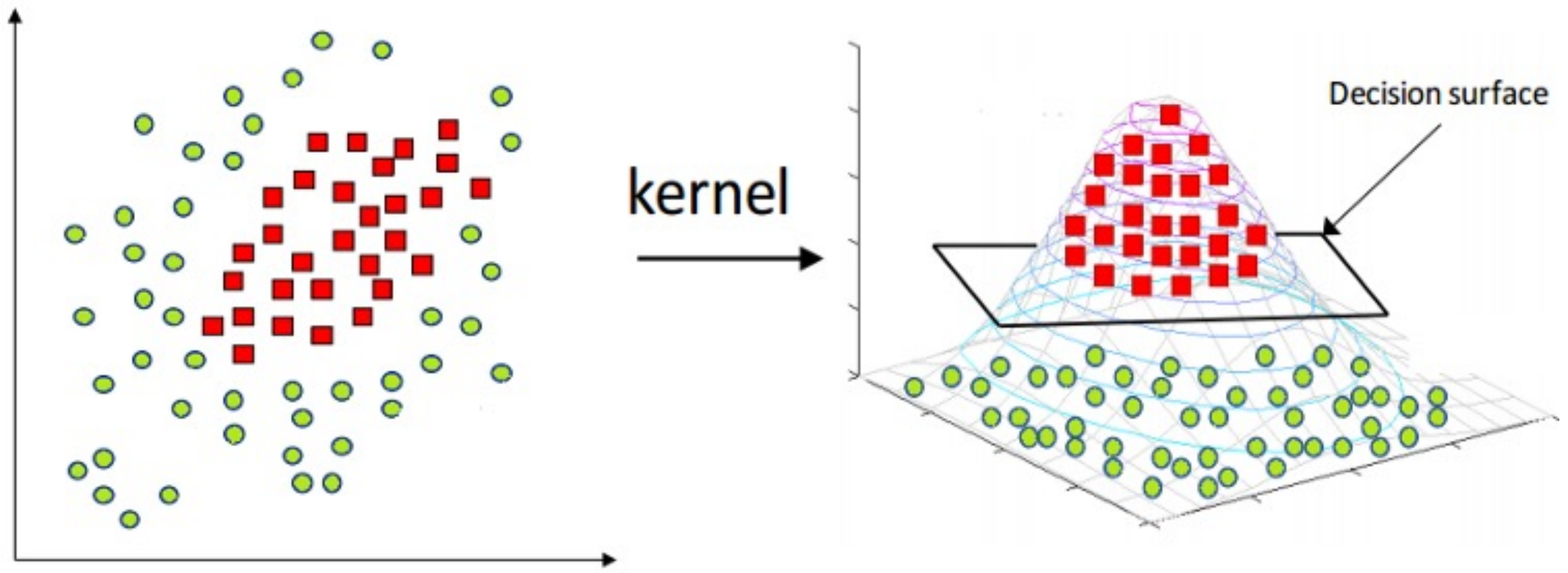
- First we start with some intuition

Help, my data is non-linearly separable!!



Data may be separable if we do a little trick

Let's make a third dimension: $x_3 = x_1^2 + x_2^2$



https://medium.com/@zxr.nju/what-is-the-kernel-trick-why-is-it-important-98a98db0961d

Code example

- Kernel.py
  - Degree = 1
  - Degree = 2

Kernels

- Kernels can find separating lines in data that is not linearly separable
- They do it by *projecting* the data into another dimension, essentially transforming the data

- Is it computationally expensive to generate these new features?
  - The *kernel trick* makes it so it's actually not too bad!

Interpretation

- If you use a linear SVM, you can interpret the weights as you would any linear classifier
  - Here's a nice demo https://medium.com/@aneesha/visualising-top-features-in-linear-svm-with-scikit-learn-and-matplotlib-3454ab18a14d

- If you use poly/RBF it's much harder to interpret the features…

SVM Resources

- Nice tutorials
  - http://cs229.stanford.edu/notes/cs229-notes3.pdf
  - https://www.youtube.com/watch?v=_PwhiWxHK8o
- Visualize the Kernel Trick
  - https://www.youtube.com/watch?v=3liCbRZPrZA
- Statsquest
  - https://www.youtube.com/watch?v=efR1C6CvhmE