Computational Methods in Astrophysics

Dr Rob Thacker (AT319E)

thacker@ap.smu.ca

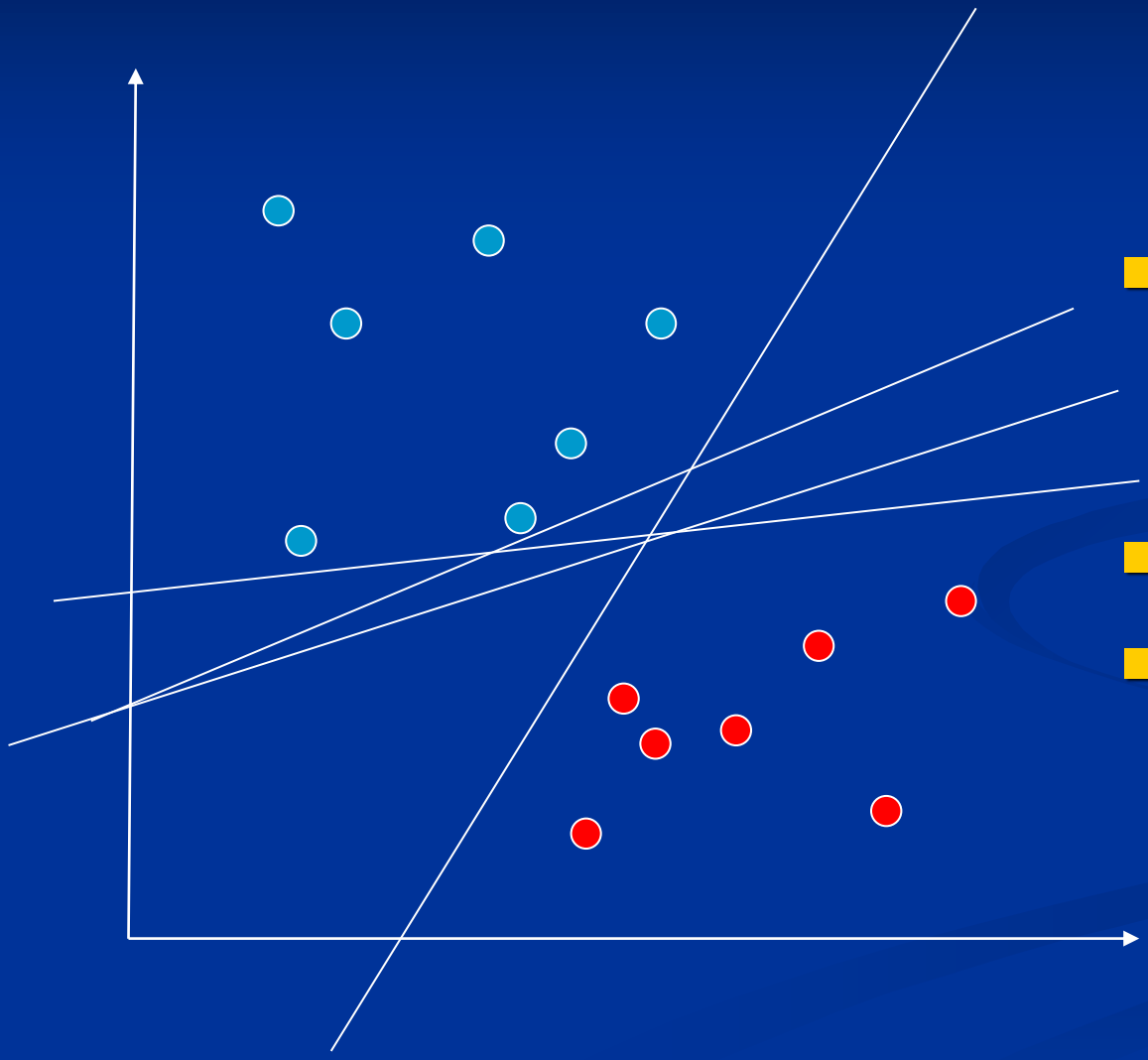
Support Vector Machines: Why so popular?

- 1998 it was used in character recognition and proved to be extremely accurate compared to tuned neural nets
- Although traditionally first taught in classification setting they can also be used for regression
  - Performance is widely viewed as "almost unbeatable"
- Can handle non-linear classification reasonably well
- SVM are designed as convex problems so that the solution is unique
  - No concerns about whether a local minima has been found
- Drawbacks:
  - they don't work well with discrete data
  - Slightly black box approach – no model is built
  - Issues with multiple classes

Very useful for highly multidimensional data

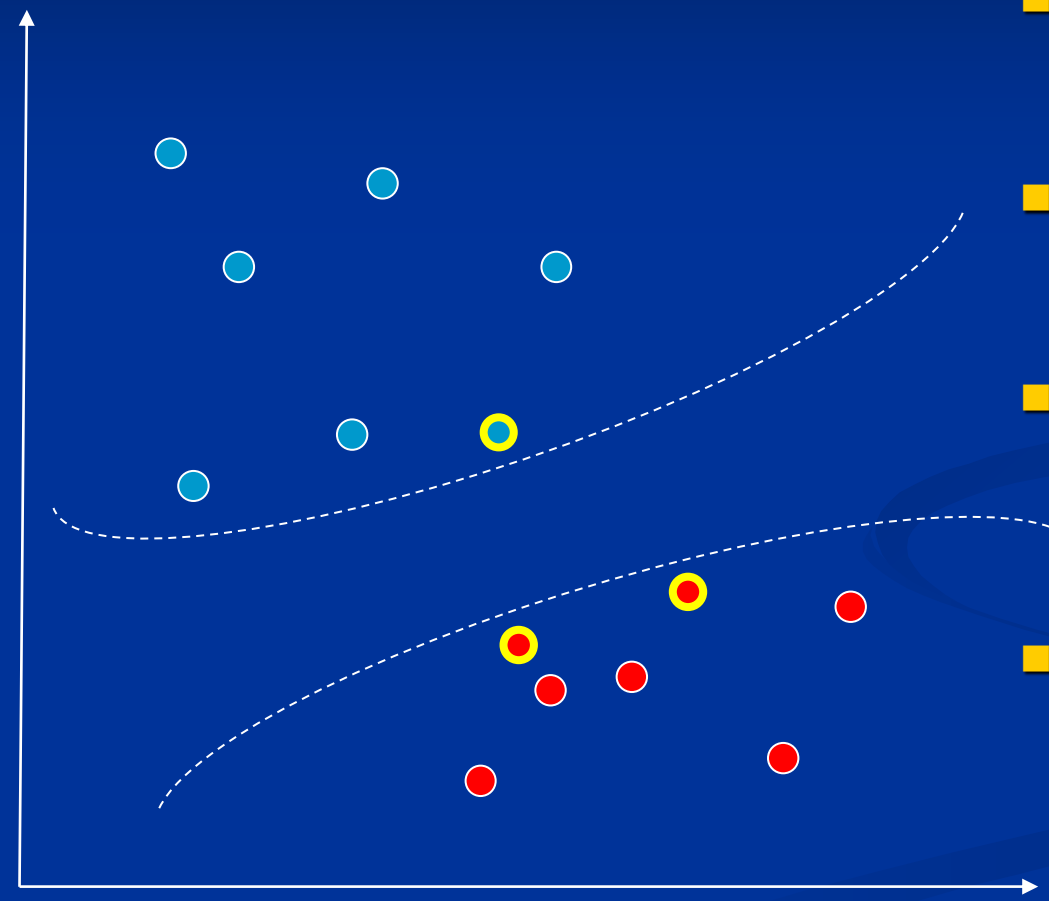
- Suppose you have 20,000 genes and 100 patients
- Number of parameters vastly exceeds number of samples
- Computations that scale as the number of genes will be 2 orders of magnitude worse (at least) than those which scale as the number of patients
- SVMs are written in such a way to be dependent on the number of samples
  - Caveat: even in this formalism there is still a hidden dependence on the lengths of vectors.

Classification in a plane



- Anyone of these lines separates the data
- Which is best?
- Notice how a plane is a binary classifier

Support vectors



- Between the two sets there is a decision surface
- The elements closest to that are known as the support vectors
- Clearly they have the largest impact on the position of the decision surface
- In practice you don't know which are SVs until you have the margin

Trivial math recap: equation of hyperplane

- Eqn of line in 2d is trivial
- Alternative form
- Taking $\vec{w}$=(x,y)
- In n-dimensions just increase vector size
  - Describes n-1 dimensional plane
- Form remains the same
- $\vec{w}$ is the normal vector to the plane

$$ax + by + c = 0$$

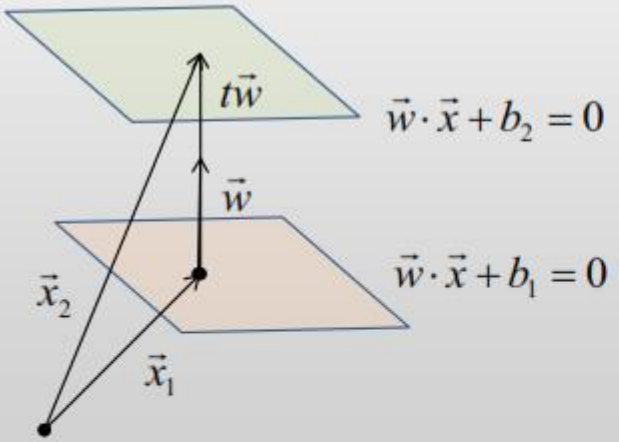
$$y = -\frac{a}{b}x - c$$

$$\vec{w}.\vec{x} + c = 0$$

c is an offset, can think of formula as

$$\vec{w}.(\vec{x} - \overrightarrow{x_0}) = 0$$

Distance between 2 parallel hyperplanes



$\vec{w}\cdot\vec{x}+b_2=0$

$\vec{w}\cdot\vec{x}+b_1=0$

$$\vec{x}_2=\vec{x}_1+t\vec{w}$$

$$D=\|t\vec{w}\|=|t|\|\vec{w}\|$$

$$\vec{w}\cdot\vec{x}_2+b_2=0$$

$$\vec{w}\cdot(\vec{x}_1+t\vec{w})+b_2=0$$

$$\vec{w}\cdot\vec{x}_1+t\|\vec{w}\|^2+b_2=0$$

$$(\vec{w}\cdot\vec{x}_1+b_1)-b_1+t\|\vec{w}\|^2+b_2=0$$

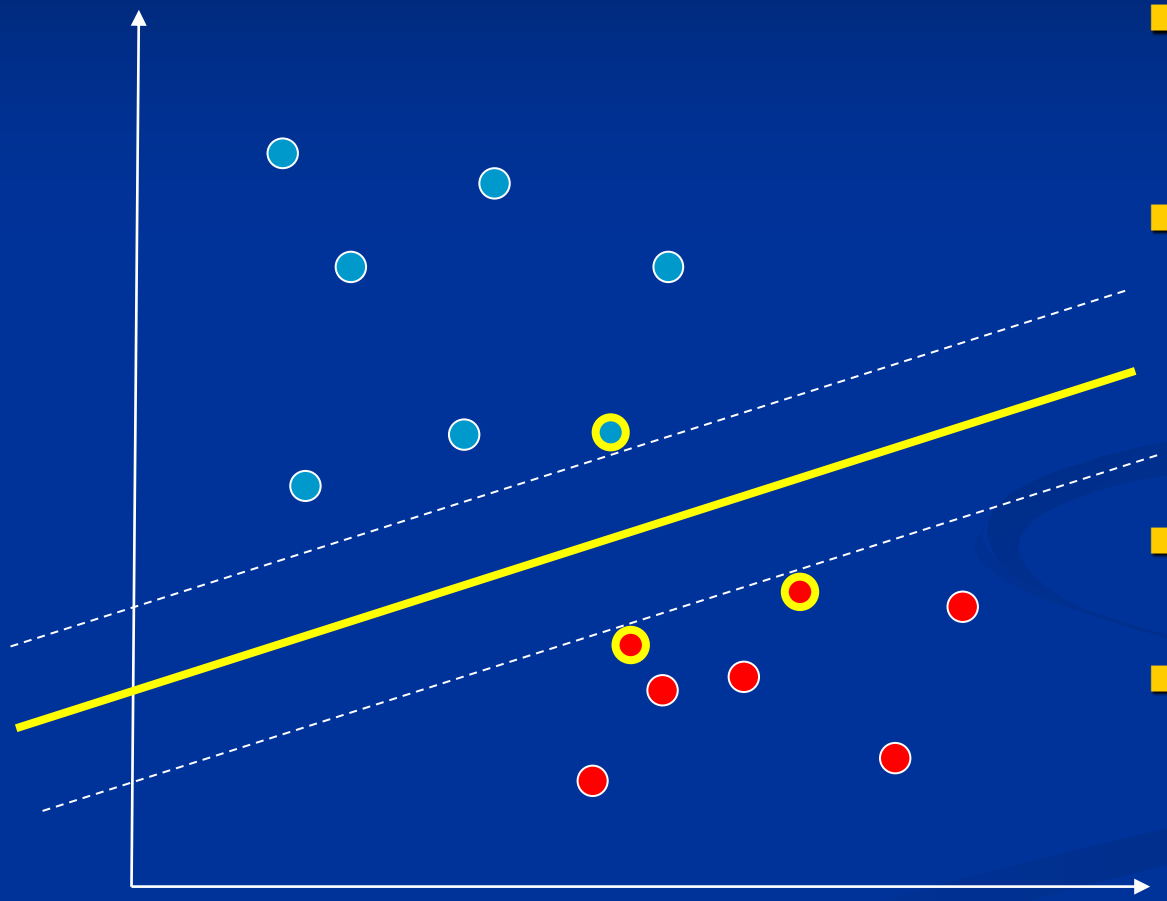
$$-b_1+t\|\vec{w}\|^2+b_2=0$$

$$t=(b_1-b_2)/\|\vec{w}\|^2$$

$$\Rightarrow D=|t|\|\vec{w}\|=|b_1-b_2|/\|\vec{w}\|$$

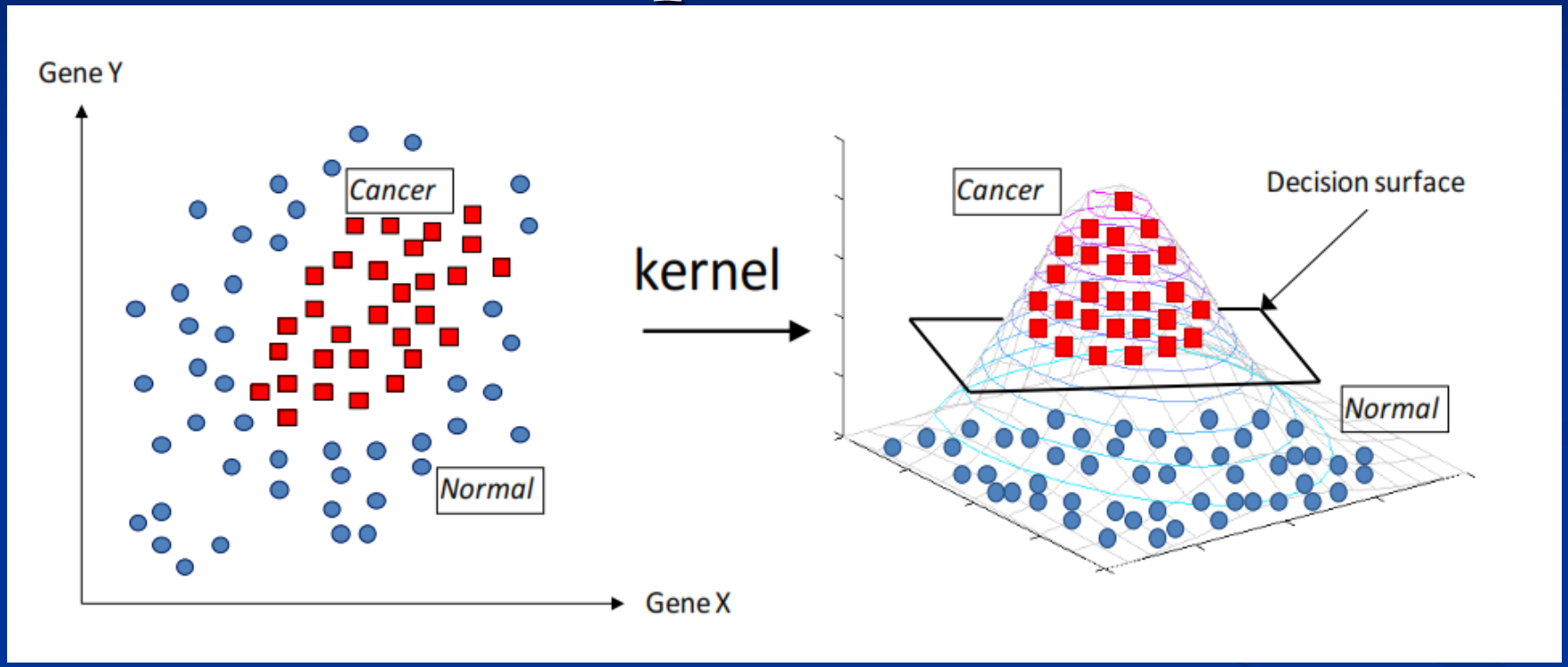
From "A Gentle Introduction to SVM" Statnikov et al

Optimizing the "gap"



- We want a plane that creates the largest "gap" around it
- This gap should be measured directly perpendicular to the plane, not along y or x
- This is called the "margin"
- In practice only a small number of points will contribute

Extending beyond linearly separable



- Using an appropriately defined kernel we can project into a higher dimensional space
- A new plane defined in that space can separate the data

From "A Gentle Introduction to SVM" Statnikov et al

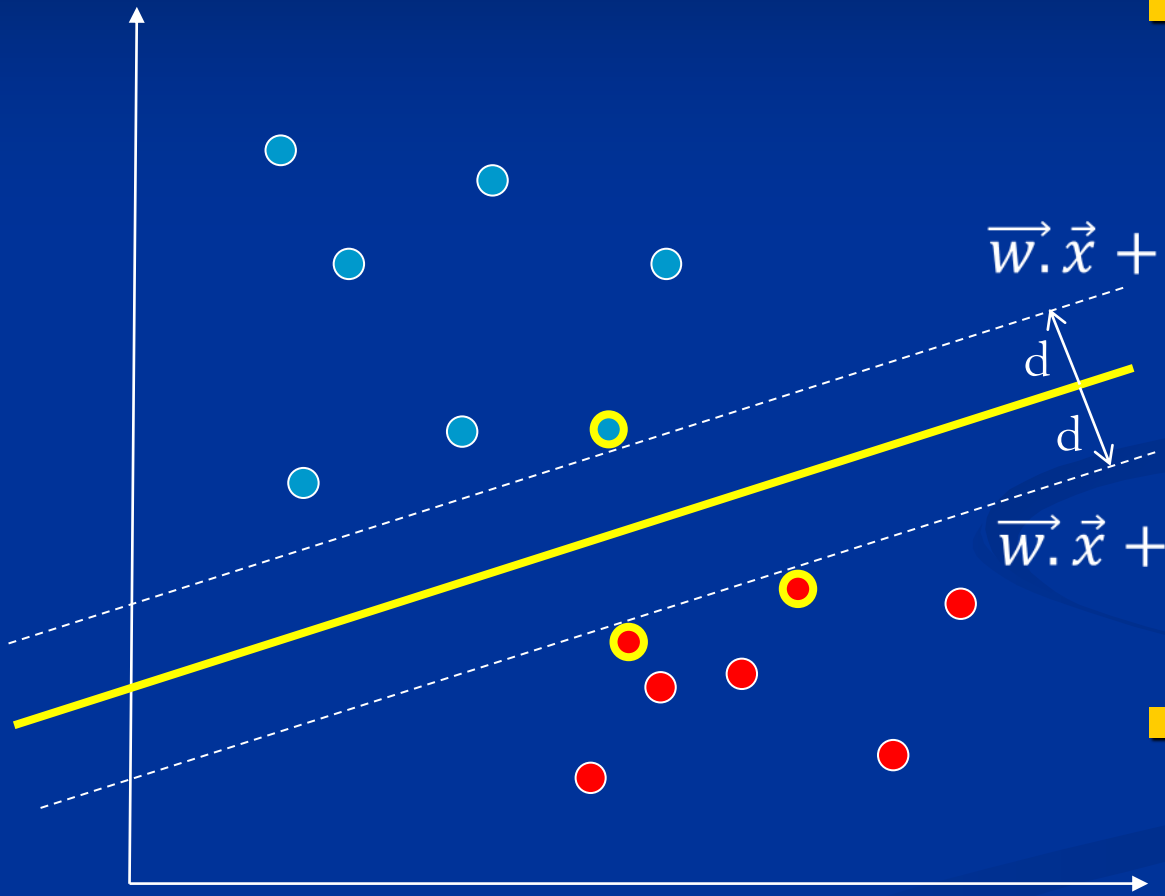
Which plane?

- Support vectors are those elements in the training set that would change the position of the dividing plane if they were removed
  - Think of them as being the "critical elements"
- Given a separating plane then the support vectors are defined as those elements for which

$$\vec{w}.\vec{x} + b = 1 \quad or \quad \vec{w}.\vec{x} + b = -1$$

- Points further away from these planes have successively more positive RHS or more negative

Maximizing the Margin

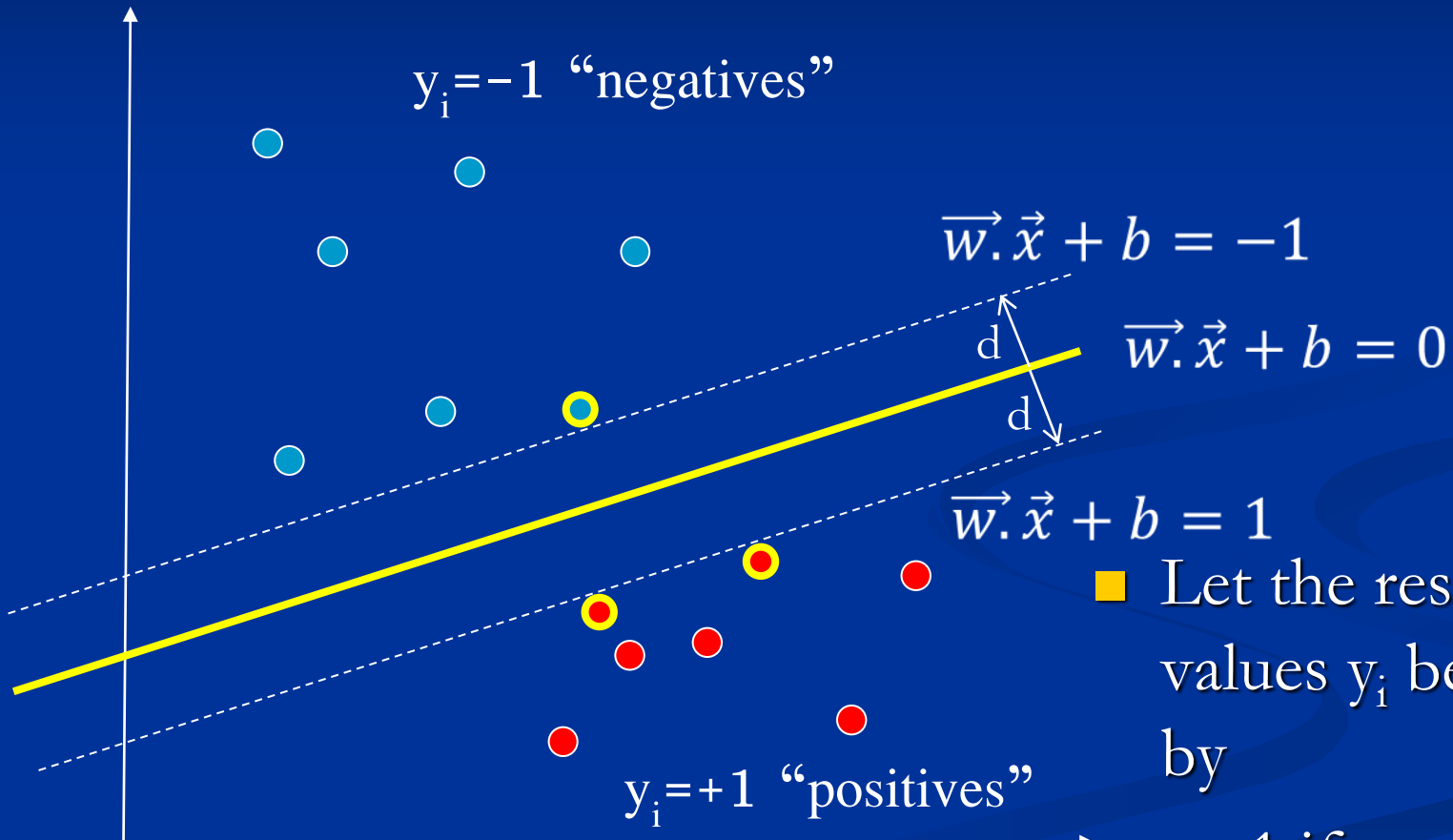


- Points on the dotted planes are the SVs
- The margin is 2d

$$2d = 2/|\vec{w}|$$

This is clearly a robust procedure relative to outliers!

Separation of classes



- Let the response values y_i be given by
- $y_i=1$ if $\mathbf{w}.\mathbf{x}_i+b\geq 1$, $y_i=-1$ if $\mathbf{w}.\mathbf{x}_i+b\leq -1$

In a nutshell

- Since the distance is dependent on $1/|\mathbf{w}|$ if we minimize $|\mathbf{w}|$ we maximize the margin
  - It's best to minimize $|w|^2/2$ – well behaved
- Other key points:
  - only the support vectors determine the slope
  - There are no points between $\mathbf{w}.\mathbf{x}+b=1$ & $\mathbf{w}.\mathbf{x}+b=-1$
  - Since we set $y_i=1$ if $\mathbf{w}.\mathbf{x}_i+b\geq 1$, $y_i=-1$ if $\mathbf{w}.\mathbf{x}_i+b\leq -1$ these two conditions can be combined into one formula:

$$y_i(\vec{w}.\vec{x}_i + b) \geq 1 \quad \forall \quad i = 1, \dots, N$$

- So we have a constraint on the minimization we want to do

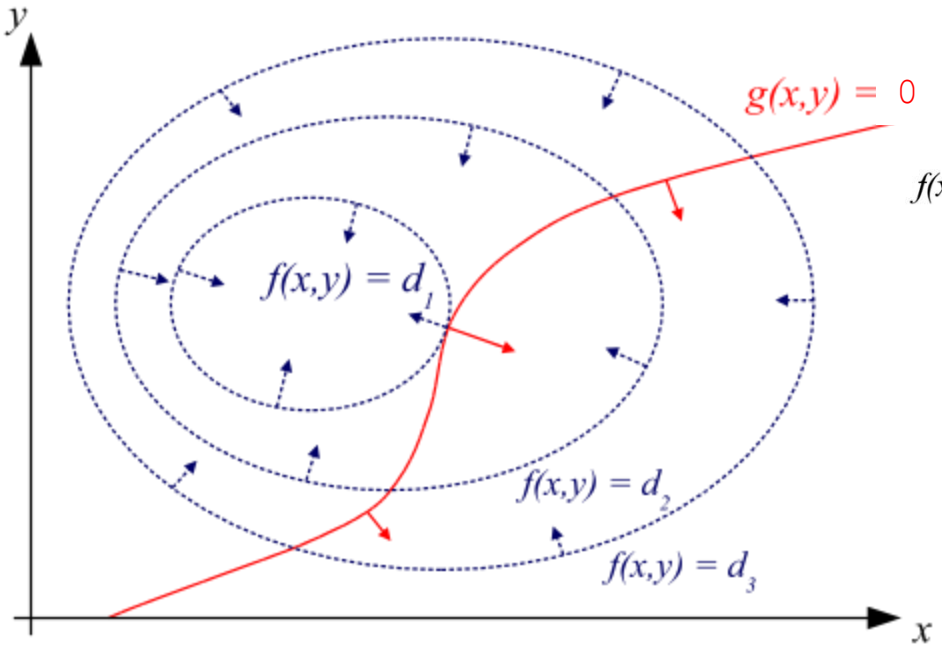
What is the classifier?

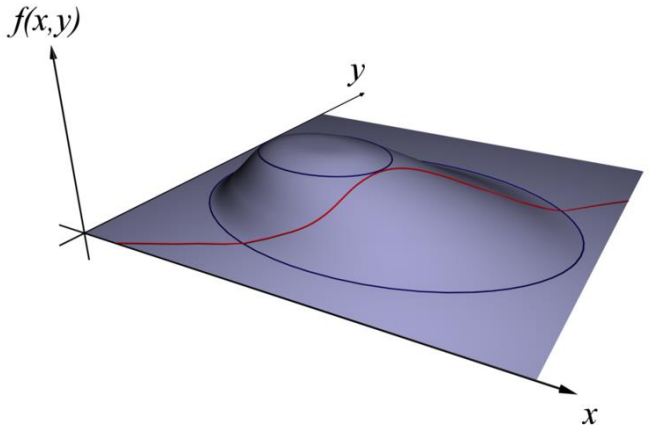
- Recall f($\mathbf{x}_i$) must produce y_i=1 or y_i=-1
- How can we get that form?
  - HINT: think about the form of the two boundary planes
- The answer is very simple:

$$f(\vec{x}) = sign(\vec{w}.\vec{x} + b)$$

Lagrange multiplier approach to optimization

- Quick recap of L.M. approaches –
- Suppose you have two funcs: maximize f(x,y) & subject to g(x,y)=0





Images from wikipedia

Lagrange Multipliers II

- If $f(x_0,y_0)$ is a max then there is no other point along $g(x,y)=0$ that we can move to that is higher
- So finding the max amounts to walking along the constraint $g(x,y)=0$ on the surface $f(x,y)$ until we find the maximum
- At the maximum $f(x,y)$ is stationary (if it is increasing we could carry on along $g(x,y)=0$ to a higher value)
- That means you are on a contour of constant $f(x,y)$
  - If so, the contours of $f(x,y)$ & $g(x,y)$ match here
- Or $f(x,y)$ is completely flat

Lagrange Multipliers III

- If contours match, normals must be parallel too
  - i.e. constraint line is tangent to f(x,y)=constant
- Normals are equal up to a constant, the Lagrange multiplier, $\boldsymbol{\lambda}$

$$\nabla f(x,y) = \boldsymbol{\lambda} \nabla g(x,y)$$

- So define a new function

$$L(x,y,\boldsymbol{\lambda}) = f(x,y) - \boldsymbol{\lambda} g(x,y)$$

- And solve gradient is zero FOR ALL variables

$$\nabla_{x,y,\boldsymbol{\lambda}} L(x,y,\boldsymbol{\lambda}) = 0$$

$$g(x,y) = 0 \; is \; required \; by \; the \; \boldsymbol{\lambda} \; derivative$$

Lagrange Multipliers IV

- To solve you rely on

$$\frac{\partial L(x,y,\boldsymbol{\lambda})}{\partial x} = 0, \quad \frac{\partial L(x,y,\boldsymbol{\lambda})}{\partial y} = 0, \quad \frac{\partial L(x,y,\boldsymbol{\lambda})}{\partial \boldsymbol{\lambda}} = 0$$

- First and second equtions usually give a relationship between x, y and $\boldsymbol{\lambda}$
  - Can usually be rewritten to give $\boldsymbol{\lambda}$=func$_1$(x,y) $\boldsymbol{\lambda}$=func$_2$(x,y)
  - From func$_1$(x,y) = func$_2$(x,y) a relationship between x and y can be derived to substitute for x or y
- The g(x,y)=0 relationship gives another between x & y
  - Substitute into this using previous relation to solve for x & y

Lagrange Multipliers V

- Example: find extrema of f(x,y)=xy+14 subject to $g(x,y)=x^2+y^2-18=0$
- Form Lagrange function:

$$L(x, y, \boldsymbol{\lambda}) = xy + 14 - \boldsymbol{\lambda}(x^2 + y^2 - 18)$$

$$\frac{\partial L(x, y, \boldsymbol{\lambda})}{\partial x} = y - 2x\boldsymbol{\lambda} = 0 \Rightarrow \boldsymbol{\lambda} = y/2x$$

$$\frac{\partial L(x,y,\boldsymbol{\lambda})}{\partial y} = x - 2y\boldsymbol{\lambda} = 0 \Rightarrow \boldsymbol{\lambda} = x/2y$$

- Thus $x^2=y^2$ and we can sub into the constraint:

$$x^2+y^2\text{-}18=2x^2\text{-}18=0 \Rightarrow x=+\text{-}3$$

Applying to maximizing the marginal

- Constraint(S) ("g") are given by $y_i(\vec{w}.\vec{x}_i + b) - 1 \geq 0$
  - Note there are as many constraints as samples - we can add more Lag. Multipliers for each
- Function ("f") to maximize is $\frac{1}{2}\sum_{i=1}^{n} w_i^2$
- Traditional to use $\boldsymbol{\alpha}$ rather than $\boldsymbol{\lambda}$ for Lag. multipliers
  - So the "Lagrange function" is written

$$\Lambda(\vec{w}, b, \vec{\boldsymbol{\alpha}}) = \frac{1}{2}\sum_{i=1}^{n} w_i^2 - \sum_{i=1}^{N} \boldsymbol{\alpha}_i[y_i(\vec{w}.\vec{x}_i + b) - 1]$$

Expand a bit

$$\Lambda(\vec{w}, b, \vec{\alpha}) = \frac{1}{2}\vec{w}.\vec{w} - \vec{w}.\sum_{i=1}^{N} \alpha_i y_i \vec{x}_i - b\sum_{i=1}^{N} \alpha_i y_i + \sum_{i=1}^{N} \alpha_i$$

- While formally appearing complex, this is actually a fairly straightforward formula
- If we consider derviatives wrt to $\vec{w}$ and $\vec{\alpha}$ then we get some simplifying assumptions

Derivation continued

If we set the derivatives with respect to $\vec{w}, b$ to 0, we obtain:

$$\frac{\partial \Lambda_P(\vec{w}, b, \vec{\alpha})}{\partial b} = 0 \Rightarrow \sum_{i=1}^{N} \alpha_i y_i = 0$$

$$\frac{\partial \Lambda_P(\vec{w}, b, \vec{\alpha})}{\partial \vec{w}} = 0 \Rightarrow \vec{w} = \sum_{i=1}^{N} \alpha_i y_i \vec{x}_i$$

We substitute the above into the equation for $\Lambda_P(\vec{w}, b, \vec{\alpha})$ and obtain "*dual formulation of linear SVMs*":

$$\Lambda_D(\vec{\alpha}) = \sum_{i=1}^{N} \alpha_i - \tfrac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j \vec{x}_i \cdot \vec{x}_j$$

We seek to maximize the above Lagrangian with respect to $\vec{\alpha}$, subject to the constraints that $\alpha_i \geq 0$ and $\sum_{i=1}^{N} \alpha_i y_i = 0$.

From "A Gentle Introduction to SVM" Statnikov et al

Why is that helpful?

- Why is the

$$\Lambda(\vec{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j \vec{x}_i . \vec{x}_j$$

form interesting? It's because there are only dot products in the function + reduced problem to that containing N unknowns

Solving for the α_j is a "quadratic programming" problem

- Dot products can be replaced by alternative "kernel" functions – this is used for data that is not linearly separable
- Constraints are $\alpha_i \geq 0 \ and \sum_{i=1}^{N} \alpha_i y_i = 0$
- Classifier $f(\vec{x}) = sign(\sum_{i=1}^{N} \alpha_i y_i \vec{x_i} . \vec{x} + b)$

Karush Kuhn Tucker Condition

$$\forall i \quad \boldsymbol{\alpha}_i [y_i (\vec{w}.\vec{x}_i + b) - 1] = 0$$

- Once $\boldsymbol{\alpha}_i$ are solved for, using the "KKT" condition given above we can get b ($\vec{w}$ formula below)
- Amazingly, only the support vectors will have non-zero $\boldsymbol{\alpha}_i$
- So pick one, then

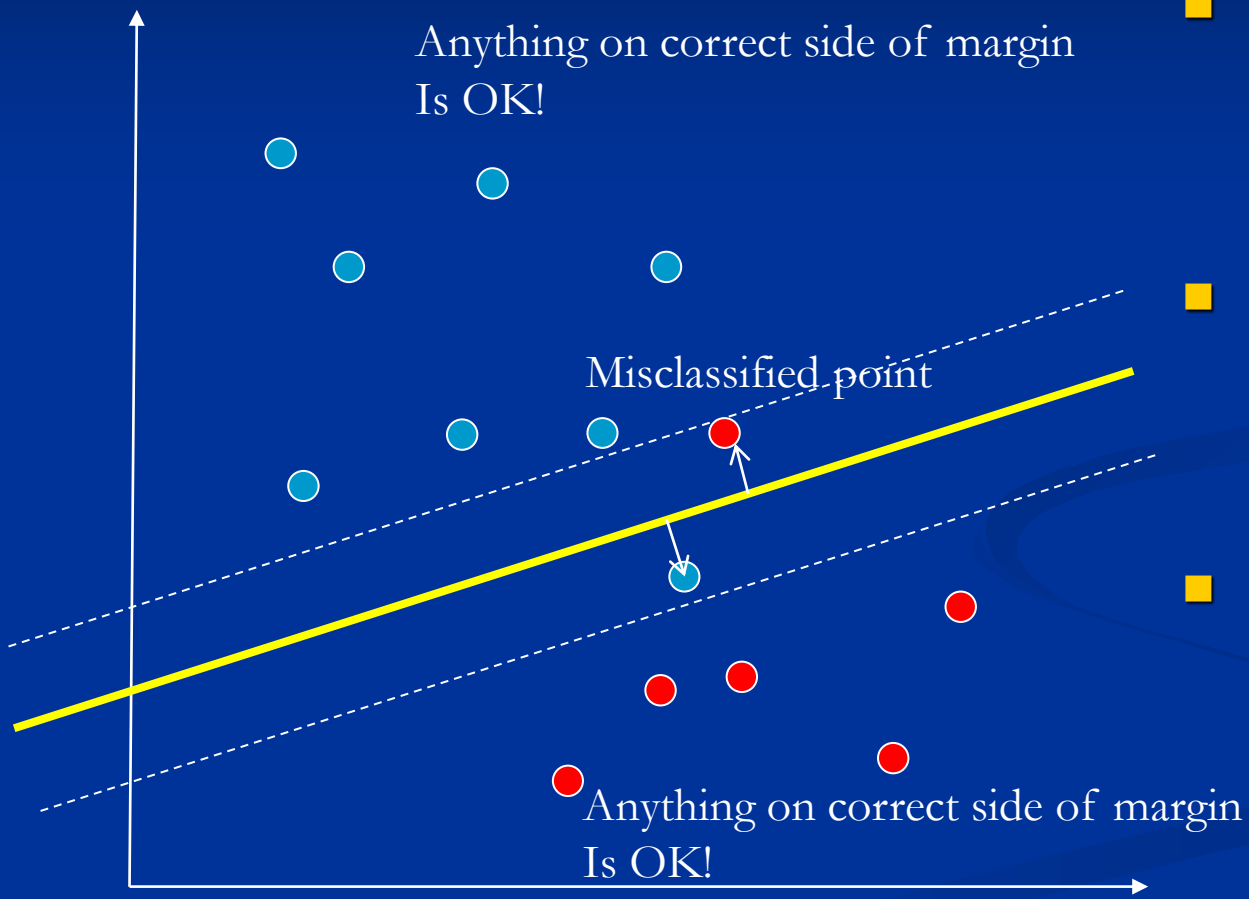
$$y_i (\vec{w}.\vec{x}_i + b) - 1 = 0$$

$$b = \frac{1}{y_i} - \vec{w}.\vec{x}_i$$

$$\vec{w} = \sum_{i=1}^{N} \boldsymbol{\alpha}_i y_i \vec{x}_i$$

- In practice better to average

How can we handle overlap?



- Distances now less than margin – concept is "lost"
- Need to account for these "noisy" results
- Add some kind of variable to allow for "negative" distances

Soft margin

- Introduce a "slack variables" (which are ≥ 0)

$$\overrightarrow{w}.\overrightarrow{x_i} + b \geq -1 + \boldsymbol{\xi}_i \text{ for } y_i = -1$$

$$\overrightarrow{w}.\overrightarrow{x_i} + b \geq 1 - \boldsymbol{\xi}_i \text{ for } y_i = +1$$

- Intrepretation – points outside margin are fine and can have $\boldsymbol{\xi}_i$=0 while the misclassified points need non-zero $\boldsymbol{\xi}_i$
- Can no longer just minimize $|\mathbf{w}|^2/2$ subject to above constraints, also need to account for slack variable, so we make it proportional, times some scaling factor, C

$$\frac{1}{2}\sum_{i=1}^{n} w_i^2 + C\sum_{i=1}^{N} \boldsymbol{\xi}_i$$

- With the constraint

$$y_i(\overrightarrow{w}.\overrightarrow{x_i} + b) - 1 + \boldsymbol{\xi}_i \geq 0 \text{ for i} = 1, \ldots, \text{N}$$

What is C?

- C is not specified a priori – finding the right value is a key issue in SVM
- C is essentially a trade off between the margin and the misclassification penalty
- Small C takes us back to looking for a large margin which suggests more training samples will be in bad positions => poorer classification and potentially poorer fit
- Large C means fewer training samples will be in bad positions but the margin will be smaller. Too large a C and you may wind-up with overfitting in non-linear problems

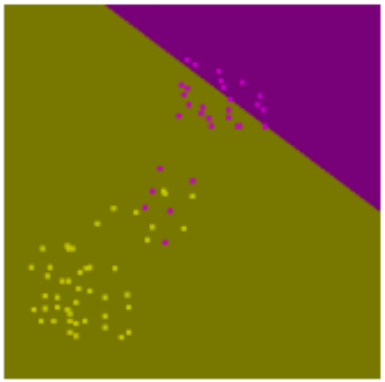
Parameter C in soft-margin SVM

Minimize $\frac{1}{2}\|\vec{w}\|^2 + C\sum_{i=1}^{N}\xi_i$ subject to $y_i(\vec{w}\cdot\vec{x}_i + b) \geq 1 - \xi_i$ for $i = 1,\ldots,N$

C=100

C=1

C=0.15



C=0.1

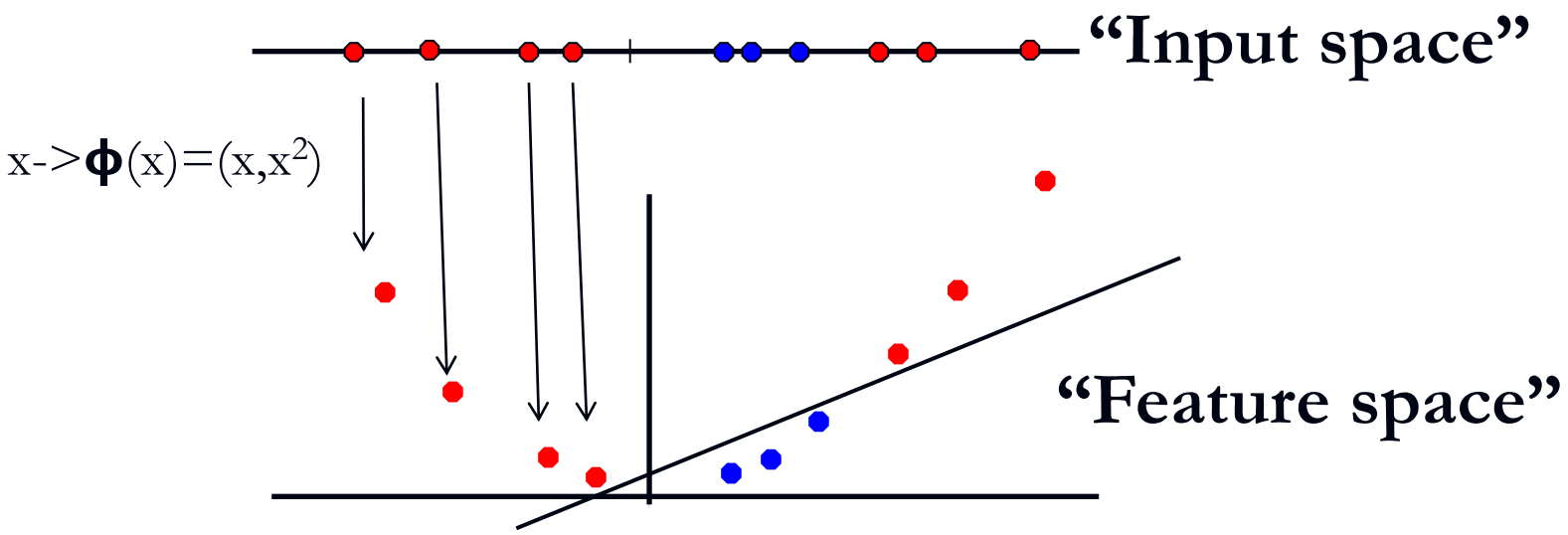
- When *C* is very large, the soft-margin SVM is equivalent to hard-margin SVM;
- When *C* is very small, we admit misclassifications in the training data at the expense of having w-vector with small norm;
- *C* has to be selected for the distribution at hand as it will be discussed later in this tutorial.

From "A Gentle Introduction to SVM" Statnikov et al

Going non-linear: Kernels

- The beauty of SVM is the idea of mapping into a higher-dimensional space to allow linear separations

In one dimension the data does not separate, but projected into 2d it is possible – there are of course many possible projection choices this one is parabolic $y=x^2$.



Credit: Moataz Al-Haj

High dimensions = more costly?

- Consider the dual formalism from earlier

$$\Lambda(\vec{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j \vec{x}_i . \vec{x}_j$$

- Dot product $\vec{x}_i . \vec{x}_j$ is used – so we don't need to define the map explicitly – we can just define a new kernel, K

$$\vec{x}_i . \vec{x}_j \rightarrow \mathrm{K}(\vec{x}_i, \vec{x}_j)$$

- This is known as the "kernel trick" as it avoids having to calculate complicated higher-d dot products
  - Although you obviously still have to do a little more computation
  - But it's insignificant compared to potential expense of higher-d

Kernels

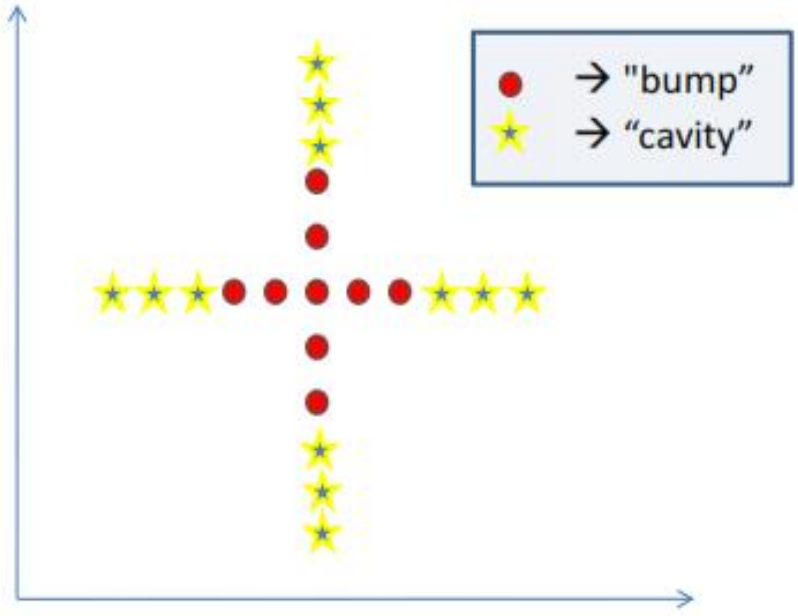
- This is really a very detailed analysis subject
- Not all functions can be kernels, must obey "Mercer Conditions"
- Common choices:

- Binomial $\mathrm{K}(\vec{x}_i, \vec{x}_j) = (\vec{x}_i . \vec{x}_j + 1)^p$
- Gaussian $\mathrm{K}(\vec{x}_i, \vec{x}_j) = exp(-\frac{1}{2\sigma^2}(\vec{x}_i - \vec{x}_j)^2)$
- Exponential $\mathrm{K}(\vec{x}_i, \vec{x}_j) = exp(-\frac{1}{2\sigma^2}|\vec{x}_i - \vec{x}_j|)$

Understanding the Gaussian kernel

Consider Gaussian kernel: $K(\vec{x}, \vec{x}_j) = \exp(-\gamma \left\| \vec{x} - \vec{x}_j \right\|^2)$

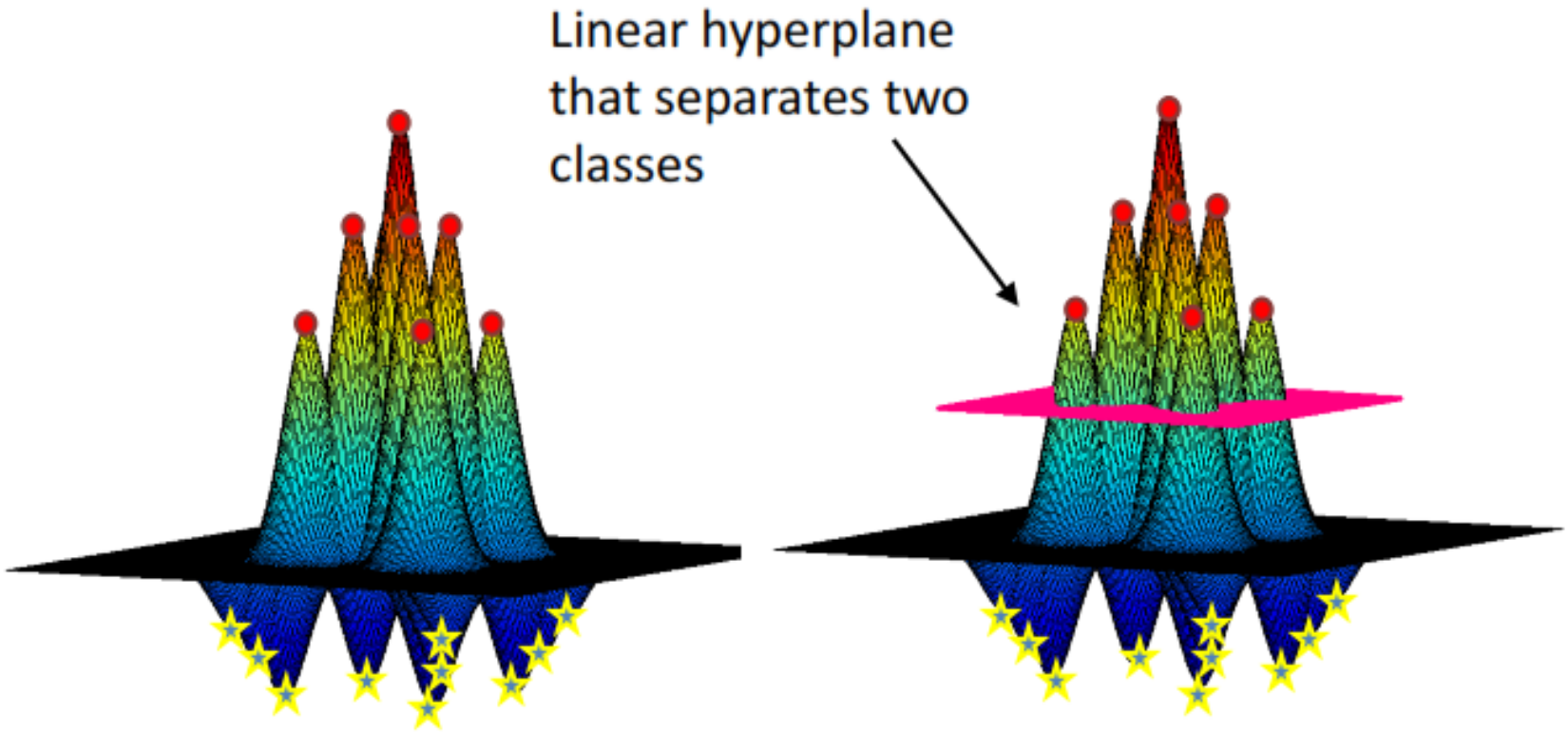
Geometrically, this is a "bump" or "cavity" centered at the training data point $\vec{x}_j$:



The resulting mapping functior is a **combination** of bumps and cavities.

From "A Gentle Introduction to SVM" Statnikov et al

Understanding the Gaussian kernel



From "A Gentle Introduction to SVM" Statnikov et al

How do you know which kernel to choose?

- You don't!
  - Many view this as a weakness of SVM
- Here's an issue – when you choose a kernel function you don't always know what dimensionality you are projecting in to
  - No guarantee you will actually be able to separate the data just because you choose a higher dimension
- Technically, because the Gaussian projection is an infinite series the true map is into an infinite dimensional space! (Think about how you would do a cross product to get all the terms)
- In practice, soft margins are also needed

Usual procedure

- Pick a kernel and C value
  - Common advice is to start with gaussian or low degree polynomial
- Check values of C via cross-validation (essentially you sub-sample)
  - Divide training data into K subsets
  - Train on union of K-1 subsets
  - Use unused subset to see how well classification works
  - Do this for all possible choices of the test subset
- Vary C, vary kernel until you get the smallest error

Summary

- SVMs are highly flexible ways to categorize data
- Key ideas:
  - Hyperplane formula and the sign of result creates a binary classifier
  - Lag. Multiplier approach gives dual formalism that depends on the # of samples
  - Only the support vectors contribute to placement of the decision surface
- Non-linear problems are handled via the `kernel trick' avoids higher dimensional problems
- Main issue: how to handle soft margins and which kernel