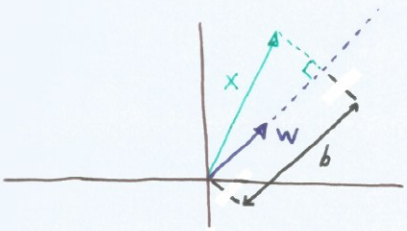
WEEK 5.3

SUPPORT-VECTOR MACHINES

→ SVMs are a linear classifier (like logistic regression & perceptron)

→ SVMs are the combination of 2 ideas → Maximum margin classification
→ The 'kernel trick'

BASICS

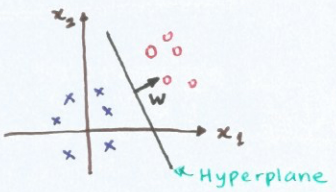


$w^T x$ is length of the projection of x onto w (if w is a unit vector)
i.e. $b = w^T x$

For any linear classifier, we have

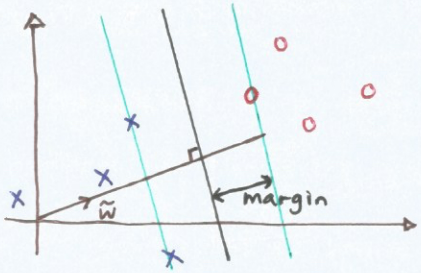
→ Training Instances (x_i, y_i), $i = 1 \dots n$
↳ $y_i \in [-1, +1]$ → we now use -1 rather than 0 in this lect.

→ Hyperplane $w^T x + w_0 = 0$



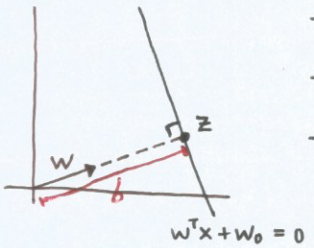
MARGIN

→ The margin is the distance between the decision boundary (or the hyperplane) and the closest training pt.



HOW TO GET EQUATION OF THE MARGIN?

1) Get distance from the origin to the hyperplane



→ Define z as the point on the hyperplane closest to origin
→ z must be proportional to w, bc. w is normal to hyperplane
→ By defn of b, we have the norm of z given by: $\|z\| = b$

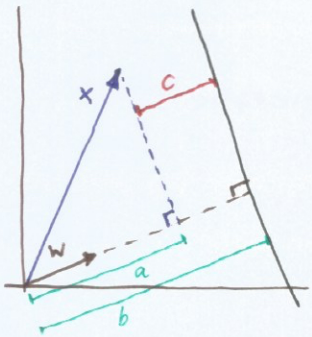
So, $b \frac{w}{\|w\|} = z$ where $\|w\| = \sqrt{w^T w}$

→ Since z is in the hyperplane, $w^T z + w_0 = 0$

Substituting we get $w^T \frac{bw}{\|w\|} + w_0 = 0$

$$b = -\frac{w_0}{\|w\|}$$

2) Compute distance from x to hyperplane



→ Now we want c, which is clearly $c = |b - a|$

where a is the length of the projection of x onto w

$$a = \frac{w^T x}{\|w\|}$$

→ Hence $$c = \left| -\frac{w_0}{\|w\|} - \frac{w^T x}{\|w\|} \right|$$

$$= \frac{1}{\|w\|} |w^T x + w_0|$$

→ The margin is the distance from the closest training pt. to the hyperplane

$$\text{margin} = \min_i \frac{1}{\|\vec{w}\|} |\vec{w}^T \vec{x}_i + w_0|$$

REMOVING THE SCALE

→ Notice that $(\vec{w}, w_0)$ and $(c\vec{w}, cw_0)$ defines the same hyperplane (The scale is arbitrary)

→ This is because we predict class $y=1$ if $w^T x + w_0 \geq 0$

which is the same thing as $cw^T x + cw_0 \geq 0$

→ To remove this freedom, we will put a constraint on $(\vec{w}, w_0)$ such that

$$\min_i |\vec{w}^T \vec{x}_i + w_0| = 1$$

→ With this constraint, the margin is always $\frac{1}{\|\vec{w}\|}$

MAX MARGIN OPTIMISATION PROBLEM

$$\max_w \frac{1}{\|\vec{w}\|}$$

subject to $w^T x_i + w_0 \geq 0 \quad \forall i \text{ w/ } y_i = +1$

$w^T x_i + w_0 \leq 0 \quad \forall i \text{ w/ } y_i = -1$

$\left(\min_i |w^T x_i + w_0| = 1\right)$ — becomes redundant

→

$$\max_w \frac{1}{\|\vec{w}\|}$$

s.t. $w^T x_i + w_0 \geq +1 \quad \forall i \text{ w/ } y_i = +1$

$w^T x_i + w_0 \leq -1 \quad \forall i \text{ w/ } y_i = -1$

$y_i (w^T x_i + w_0) \geq 1$ for all i

because $y_i \in \{-1, +1\}$

→ Note that maximizing $\frac{1}{\|\vec{w}\|}$ is the same as minimizing $\|\vec{w}\|^2$

→ For a linear SVM, the form of the soln for the model parameters is:

$$\vec{w} = \sum_i \alpha_i y_i \vec{x}_i$$

→ This soln is sparse, meaning that the optimal hyperplane is determined by just a small no. of the training examples (called the support vectors)

↳ WHY? The points that are on the margin are the ones that determine the margin. Moving the points not on the marginal hyperplane do not change the solution.

↳ Hence this means that most of the $\alpha_i = 0$ for non-support vectors

→ The optimization problem to find α_i has a unique global soln w/ no local minima

→ Doing a prediction on new data point x, this is how we compute it:

$$f(x) = \text{sign}\left((w^T x) + w_0\right)$$

$$= \text{sign}\left(\sum_{i=1}^{n} \alpha_i y_i (x_i^T x) + w_0\right)$$

↳ The prediction is based on the dot pdt. of the new data pt. w/ the support vectors

NON-SEPARABLE TRAINING SETS

→ If at least one pt. is misclassified → Data is not linearly separable and the optimization problem tht. we have given has no solution.

→ SOLN
- Don't require tht. we classify all points correctly; ignore some points
- This is obviously dangerous so we need to give a penalty for doing so

↳ Add a 'slack' variable $\xi_i \geq 0$ for each training example

- $\xi_i = 0$ if data point is correctly classified.
- New optimization problem is to minimize

$$\|\vec{w}\|^2 + C\left(\sum_{i=1}^{n} \xi_i^{(k)}\right)$$ — usually set k=1

$$\text{s.t. to } w^T x_i + w_0 \geq 1 - \xi_i \quad \text{for } y_i = +1$$

$$w^T x_i + w_0 \leq -1 + \xi_i \quad \text{for } y_i = -1$$

↳ C is a trade-off parameter. Large C gives a large penalty to errors.

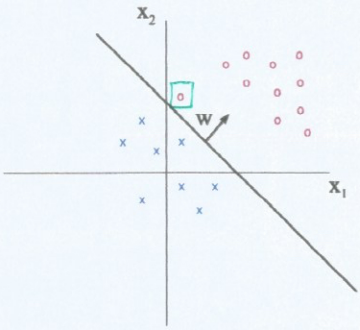
↳ Soln to this has the same form as before, but in this case the support vectors also include all where $\xi_i \neq 0$ → We moved misclassified data pts. to the margin

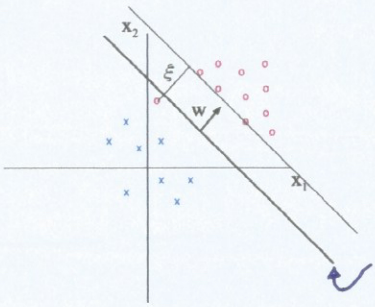
→ Now, our problem looks even more like a ridge regression:

- $C(\sum_{i=1}^{n} \xi_i)$ measures how well we fit the data
- $\|\vec{w}\|^2$ penalizes weight vectors w/ a large norm

↳ can be viewed as a regularization param. (like λ)

→ We are allowed to do this tradeoff even when data is linearly separable. Why might we want to do this?





Might be preferable to have a DB tht. is more evenly spaced between the mass of the 2 classes

NON-LINEAR SVMs

→ SVMs can be made nonlinear just like any other linear algo we've seen (i.e. basis expansion)

→ But in SVM, the basis expansion is implemented using something called a kernel

↳ Kernels can be faster to compute w/ if the expanded feature space is very high dimensional

KERNELIZED SVMs

→ A kernel is in some sense an alternate API for specifying to the classifier what your expanded feature space is

→ Linear algo. depends only on $x^T x_i$. Hence transformed algo. depends only on $\phi(x)^T \phi(x_i)$

→ So we use a kernel fn $k(x_i, x_j)$ to compute the dot pdt. of transformed $\vec{x}$ w/ each of the transformed support vectors, ...

$$k(x_i, x_j) = \phi(x_i)^T \phi(x_j)$$

... then perform a linear weighted sum of the results to see if it is (+)ve/(−)ve

SVMs

→ A SVM is a kernelized maximum margin classifier

→ For max margin we had the property

$$\vec{w} = \sum_i a_i y_i \vec{x}_i$$

This means we would predict the label of a test sample $\vec{x}$ as

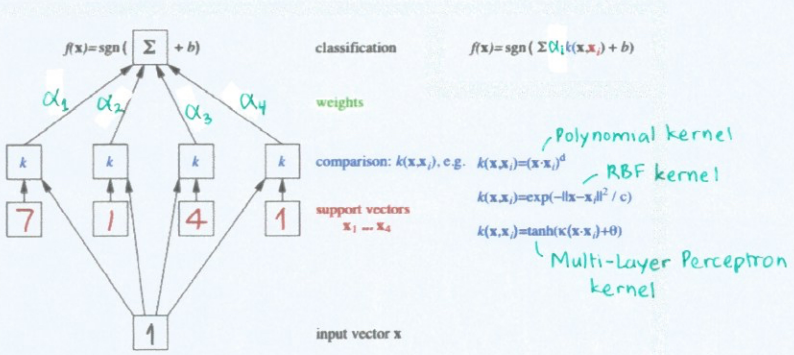
$$\hat{y} = \text{sign}\left[w^T x + w_0\right]$$

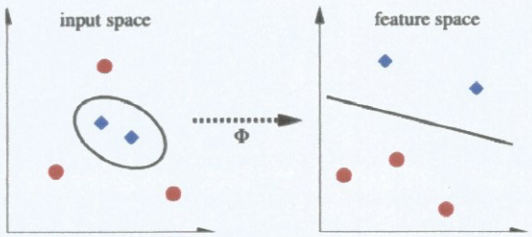
$$= \text{sign}\left[\sum_i \alpha_i y_i (x_i^T x) + w_0\right]$$

Kernelizing this we get

$$\hat{y} = \text{sign}\left[\sum_i \alpha_i y_i \, k(x_i, x) + b\right]$$

i.e.





DIFFERENCE BETWEEN LOGISTIC REGRESSION & SVM

1) SVM tries to find the 'best' margin that separates the classes while logistic regression can have diff. DB w/ diff. weights tht. are near the optimal pt.

2) SVM works well w/ unstructured & semi-struct. data like text & images while logistic regression works well w/ already identified independent variables

3) The risk of overfitting is less in SVM

4) SVM gives sparse soln while LR gives non-sparse soln