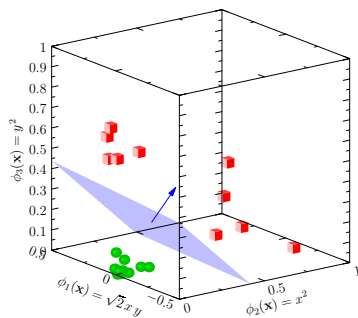


## Advanced Machine Learning

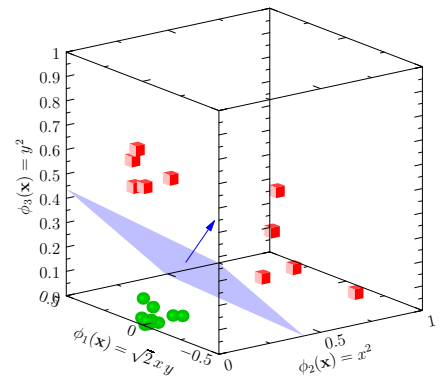
### Support Vector Machines



*Support Vector Machines, maximum margins*

## Outline

1. The Big Picture
2. Maximum Margins
3. Duality
4. Practice

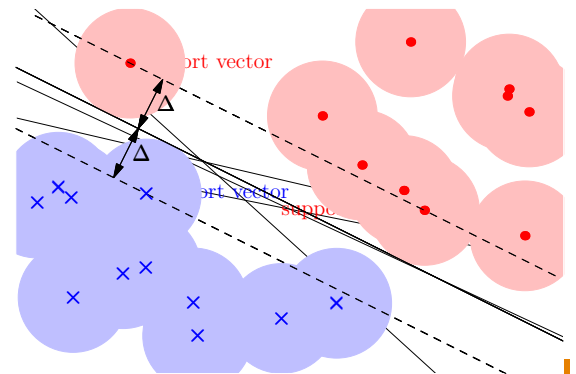


## Support Vector Machines

- Support vector machines, when used right, often have the best generalisation results
- They are typically used on numerical data, but can and have been adapted to text, sequences, etc.
- Although not as trendy as deep learning, they will often be the method of choice on small data sets
- They subtly regularise themselves, choosing a solution that generalises well from a host of different solutions

## Linear Separation of Data

- SVMs classify linearly separable data



- Finds maximum-margin separating plane

## Extended Feature Space

- To increase the likelihood of linear-separability we often use a high-dimensional mapping

$$\mathbf{x} = (x_1, x_2, \dots, x_p)^T \rightarrow \phi(\mathbf{x}) = (\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \dots, \phi_r(\mathbf{x}))^T$$

$$r \gg p$$

- Finding the maximum margin hyper-plane is time consuming in “primal” form if  $r$  is large
- We can work in the “dual” space of patterns, then we only need to compute inner-products

$$\langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j) = \sum_{k=1}^r \phi_k(\mathbf{x}_i) \phi_k(\mathbf{x}_j)$$

## Kernel Trick

- If we choose a **positive semi-definite** kernel function  $K(\mathbf{x}, \mathbf{y})$  then there exists functions  $\phi(\mathbf{x}) = (\phi_k(\mathbf{x}) | k = 1, 2, \dots, r)$ , such that

$$K(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle$$

(like an eigenvector decomposition of a matrix)

- Never need to compute  $\phi_k(\mathbf{x}_i)$  explicitly as we only need the inner-product  $\langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle = K(\mathbf{x}_i, \mathbf{x}_j)$  to compute maximum margin separating hyper-plane
- Sometimes  $\phi(\mathbf{x}_i)$  is an infinite dimensional vector so it is good we don't have to compute all the elements!

## Kernel Functions

- Kernel functions are symmetric functions of two variable
- Strong restriction: *positive semi-definite*
- Examples

Quadratic kernel:  $K(\mathbf{x}_1, \mathbf{x}_2) = (\mathbf{x}_1^T \mathbf{x}_2)^2$

Gaussian (RBF) kernel:  $K(\mathbf{x}_1, \mathbf{x}_2) = e^{-\gamma \|\mathbf{x}_1 - \mathbf{x}_2\|^2}$

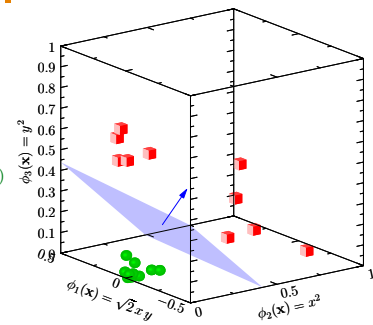
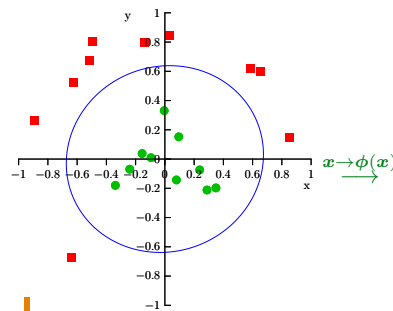
- Consider the mapping

$$\mathbf{x}_i = \begin{pmatrix} x_i \\ y_i \end{pmatrix} \rightarrow \phi(\mathbf{x}_i) = \begin{pmatrix} x_i^2 \\ y_i^2 \\ \sqrt{2}x_i y_i \end{pmatrix}$$

## Non-linear Separation of Data

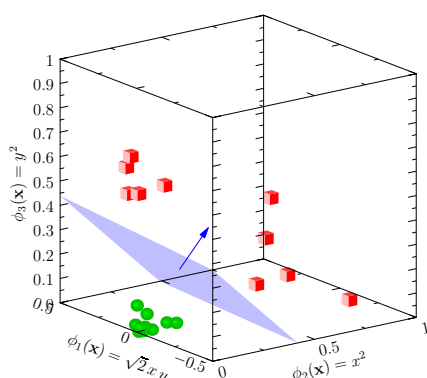
$$K(\mathbf{x}_1, \mathbf{x}_2) = \begin{pmatrix} x_1^2 & y_1^2 & \sqrt{2}x_1 y_1 \end{pmatrix} \begin{pmatrix} x_2^2 \\ y_2^2 \\ \sqrt{2}x_2 y_2 \end{pmatrix} = x_1^2 x_2^2 + y_1^2 y_2^2 + 2x_1 y_1 x_2 y_2$$

$$= (x_1 x_2 + y_1 y_2)^2 = (\mathbf{x}_1^T \mathbf{x}_2)^2$$



## Outline

1. The Big Picture
2. **Maximum Margins**
3. Duality
4. Practice

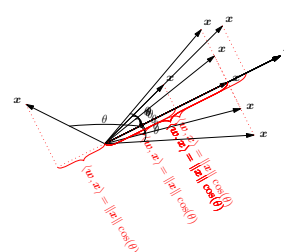


## Inner Product

- Recall the inner or dot product in  $\mathbb{R}^n$

$$\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x} \cdot \mathbf{y} = \mathbf{x}^T \mathbf{y} = \sum_{i=1}^n x_i y_i = \|\mathbf{x}\| \|\mathbf{y}\| \cos(\theta)$$

- If  $\|\mathbf{w}\| = 1$  then  $\langle \mathbf{x}, \mathbf{w} \rangle = \|\mathbf{x}\| \cos(\theta)$



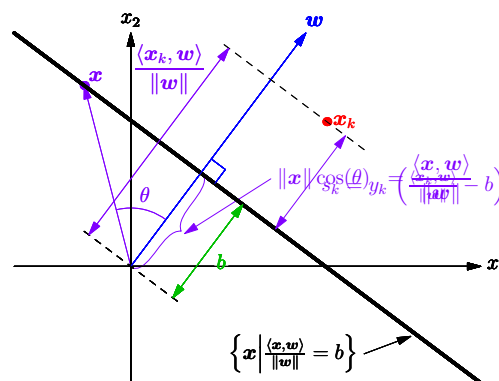
## Maximise Margin

- Consider a linearly separable set of data
  - ★  $\mathcal{D} = \{(\mathbf{x}_k, y_k)\}_{k=1}^m$
  - ★  $y_k \in \{-1, 1\}$
- Our task is to find a separating plane defined by the orthogonal vector  $\mathbf{w}$  and a threshold  $b$  such that

$$y_k \left( \frac{\langle \mathbf{w}, \mathbf{x}_k \rangle}{\|\mathbf{w}\|} - b \right) \geq \Delta$$

where  $\Delta$  is the margin

## Distance to hyperplanes



## Constrained Optimisation

- Wish to find  $\mathbf{w}$  and  $b$  to maximise  $\Delta$  subject to constraints

$$y_k \left( \frac{\langle \mathbf{w}, \mathbf{x}_k \rangle}{\|\mathbf{w}\|} - b \right) \geq \Delta \quad \text{for all } k = 1, 2, \dots, m$$

- If we divide through by  $\Delta$

$$y_k \left( \frac{\langle \mathbf{w}, \mathbf{x}_k \rangle}{\Delta \|\mathbf{w}\|} - \frac{b}{\Delta} \right) \geq 1 \quad \text{for all } k = 1, 2, \dots, m$$

- Define  $\hat{\mathbf{w}} = \mathbf{w}/(\Delta \|\mathbf{w}\|)$  and  $\hat{b} = b/\Delta$

$$y_k \left( \langle \hat{\mathbf{w}}, \mathbf{x}_k \rangle - \hat{b} \right) \geq 1$$

## Quadratic Programming Problem

- Note that as  $\hat{\mathbf{w}} = \mathbf{w}/(\Delta \|\mathbf{w}\|)$

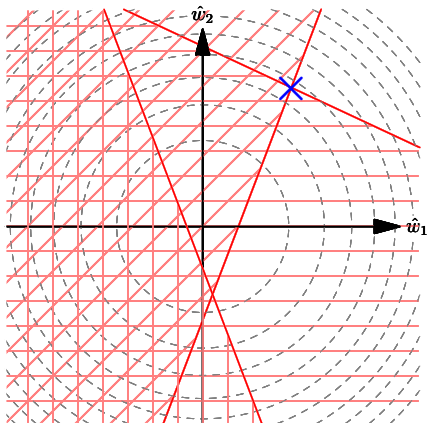
$$\|\hat{\mathbf{w}}\| = \left\| \frac{\mathbf{w}}{\Delta \|\mathbf{w}\|} \right\| = \frac{1}{\Delta \|\mathbf{w}\|} \|\mathbf{w}\| = \frac{1}{\Delta}$$

- Minimising  $\|\hat{\mathbf{w}}\|^2$  is equivalent to maximising the margin  $\Delta$

- Can write the optimisation problem as a *quadratic programming problem*

$$\min_{\hat{\mathbf{w}}, \hat{b}} \frac{\|\hat{\mathbf{w}}\|^2}{2} \quad \text{subject to } y_k \left( \langle \hat{\mathbf{w}}, \mathbf{x}_k \rangle - \hat{b} \right) \geq 1 \quad \text{for all } k = 1, 2, \dots, m$$

## Quadratic Programming in SVMs

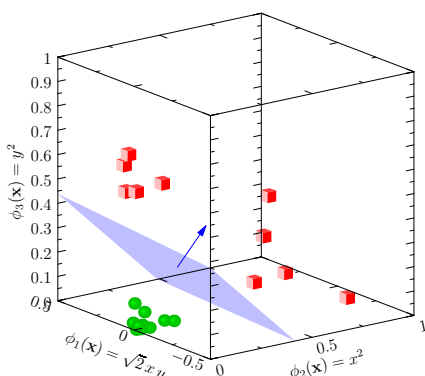


## Quadratic Programming

- We have a quadratic programming problem for the weights  $\hat{\mathbf{w}} = (\hat{w}_1, \hat{w}_2, \dots, \hat{w}_p)$  and bias  $\hat{b}$  and  $m$  constraints
- This is a classic but fiddly optimisation problems
- It can be solved in  $O(p^3)$  time (it involves inverting matrices) (phew it is not NP-complete!)
- We will see that there is an equivalent dual problem which allows us to use the kernel trick with time complexity  $O(m^3)$

## Outline

1. The Big Picture
2. Maximum Margins
3. **Duality**
4. Practice



## Extended Feature Space

- We can generalise the SVM if we map all our features vectors to an extended feature space

$$x \rightarrow \phi(x)$$

- The components of  $\phi(x)$  will typically be (non-linear) functions of  $x$  (e.g.  $\phi_1(x) = x_1^2, \phi_2(x) = x_2^2, \phi_3(x) = \sqrt{2}x_1x_2$ )
- We are free to choose whatever mappings we like
- There may be many more components of  $\phi(x)$  than of  $x$  making it easier to find a linear separation of the two classes
- But in the extended feature space (involving  $\phi(x) = (\phi_1(x), \phi_2(x), \dots, \phi_r(x))$ ) the time complexity is  $O(r^3)$

## Lagrangian

- In the extended feature space we can find a separating plane (given by  $w$  and  $b$ ) with maximum margin by solving the problem

$$\min_{w,b} \frac{\|w\|^2}{2} \quad \text{subject to } y_k(\langle w, \phi(x_k) \rangle - b) \geq 1 \text{ for all } k = 1, 2, \dots, m$$

- We can write this as a Lagrange problem

$$\min_{w,b} \max_{\alpha \geq 0} \mathcal{L}(w, b, \alpha)$$

where

$$\mathcal{L}(w, b, \alpha) = \frac{\|w\|^2}{2} - \sum_{k=1}^m \alpha_k \left( y_k (\langle w, \phi(x_k) \rangle - b) - 1 \right)$$

subject to  $\alpha_k \geq 0$

## Obtaining the Dual Form of the Problem

- Differentiating the Lagrangian with respect to  $w$

$$\mathcal{L}(w, b, \alpha) = \frac{1}{2} \|w\|^2 - \sum_{k=1}^m \alpha_k \left( y_k (\langle w, \phi(x_k) \rangle - b) - 1 \right)$$

- $\nabla_w \mathcal{L} = w - \sum_{k=1}^m \alpha_k y_k \phi(x_k) = 0$  implies that  $w^* = \sum_{k=1}^m \alpha_k y_k \phi(x_k)$

- $\frac{\partial \mathcal{L}}{\partial b} = \sum_{k=1}^m \alpha_k y_k = 0$  implies  $\sum_{k=1}^m \alpha_k y_k = 0$

- Substituting back into the Lagrangian

$$\max_{\alpha \geq 0} \sum_{k=1}^m \alpha_k - \frac{1}{2} \sum_{k,l=1}^m \alpha_k \alpha_l y_k y_l \langle \phi(x_k), \phi(x_l) \rangle$$

## The Dual Problem

- The dual problem is now to find  $\alpha_k$ 's that maximise

$$\mathcal{L}(\alpha) = \sum_{k=1}^m \alpha_k - \frac{1}{2} \sum_{k,l=1}^m \alpha_k \alpha_l y_k y_l \langle \phi(x_k), \phi(x_l) \rangle$$

- subject to constraints

$$\sum_{k=1}^m \alpha_k y_k = 0 \quad \forall k = 1, 2, \dots, m \quad \alpha_k \geq 0$$

- The Hessian of  $\mathcal{L}(\alpha)$  has elements  $H_{kl} = -y_k y_l \langle \phi(x_k), \phi(x_l) \rangle$  so  $v^T H v = -\|\sum_k v_k y_k \phi(x_k)\|^2 \leq 0$  (note this is negative semi-definite so there is a unique maximum)

## Sequential Minimal Optimisation

- One of the most efficient techniques for training SVMs is *Sequential Minimal Optimisation* or SMO
- This takes two Lagrange multipliers  $\alpha_i$  and  $\alpha_j$  and adjusts them to maximise the dual objective function
- This is very quick as it can be done in closed form
- Note that because  $\sum_{k=1}^m y_k \alpha_k = 0$  we have to change at least two variables at the same time
- A heuristic is used to choose good pairs of  $\alpha$ 's to optimise
- Run until close to the optimum

## Kernel Trick

- We will show in the next lecture that if  $K(x, y)$  is a positive semi-definite function then it can always be written as

$$K(x, y) = \langle \phi(x), \phi(y) \rangle$$

- As  $\langle \phi(x_k), \phi(x_l) \rangle$  appears in the dual problem we can express the dual problem as finding  $\alpha_k$ 's that maximise

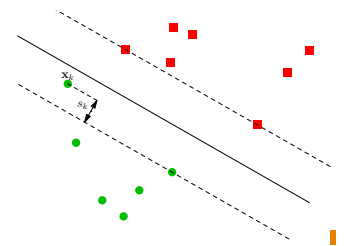
$$\mathcal{L}(\alpha) = \sum_{k=1}^m \alpha_k - \frac{1}{2} \sum_{k,l=1}^m \alpha_k \alpha_l y_k y_l K(x_k, x_l)$$

- We therefore never have to compute  $\phi(x)$

## Soft Margins

- We can relax the margin constraints by introducing *slack variables*,  $s_k \geq 0$

$$y_k(\langle x_k, w \rangle - b) \geq 1 - s_k$$



- Minimise  $\frac{\|w\|^2}{2} + C \sum_{k=1}^n s_k$
- Larger  $C$  punishes slack variables more

## Dual Problem with Slack Variables

- The Lagrangian with slack variables is

$$\mathcal{L} = \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{k=1}^m s_k - \sum_{k=1}^m \alpha_k \left( y_k (\langle \mathbf{w}, \phi(\mathbf{x}_k) \rangle - b) - 1 + s_k \right) - \sum_{k=1}^m \beta_k s_k$$

where  $\beta_k$  are Lagrange multipliers that ensure  $s_k \geq 0$  (note that  $\beta_k \geq 0$ —this is the KKT condition)

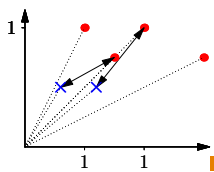
- Now minimising with respect to  $s_i$

$$\frac{\partial \mathcal{L}}{\partial s_i} = C - \alpha_i - \beta_i = 0$$

- Or  $\alpha_i = C - \beta_i$ . Since  $\beta_i \geq 0$  the constraint is  $\alpha_i \leq C$  (recall also  $\alpha_i \geq 0$ )

## Getting SVMs to Work Well

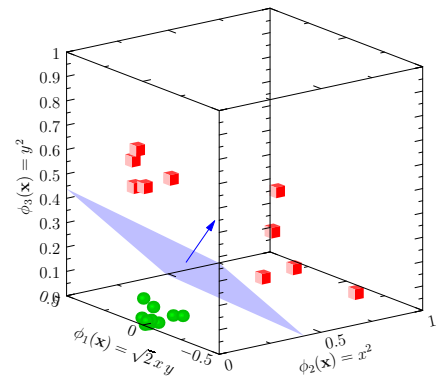
- SVMs rely on distances between data points
- These will change relative to each other if we rescale some features but not other—giving different maximum-margin hyper-planes



- If we don't know what features are important (most often the case), then it is worth scaling each feature (for example, so their range is between 0 and 1 or their variance is 1)

## Outline

- The Big Picture
- Maximum Margins
- Duality
- Practice



## Optimising C

- Recall that we can introduce soft-margins using slack variables where we minimise  $\frac{\|\hat{\mathbf{w}}\|^2}{2} + C \sum_{k=1}^m s_k$  subject to constraints
- In practice it can make a huge difference to the performance if we change  $C$
- Optimal  $C$  values changes by many orders of magnitude e.g.  $2^{-5}$ – $2^{15}$
- Typically optimised by a grid search (start from  $2^{-5}$  say and double until you reach  $2^{15}$ )
- Measure performance on a validation set

## Choosing the Right Kernel Function

- There are kernels design for particular data types (e.g. string kernels for text or biological sequences)
- For numerical data, people tend to look at using no kernel (linear SVM), a radial basis function (Gaussian) kernel or polynomial kernels
- Kernels often come with parameters, e.g. the popular radial basis function kernel

$$K(\mathbf{x}, \mathbf{y}) = e^{-\gamma \|\mathbf{x} - \mathbf{y}\|^2}$$

- Optimal  $\gamma$  values range over  $2^{-15}$ – $2^3$

## SVM Libraries

- Although SVMs have unique solutions, they require very well written optimisers
- If you have a large data set they can be very slow
- There are good libraries out there: svmlib, svm-lite, (now old), scikit-learn, etc.
- These will often automate normalisation of data and grid search for parameters

## Multi-Class Problems

- By construction SVMs separate only two classes
- If we have a multi-class problem we have to use multiple SVMs
- There are two major ways practitioners do this
  - One-versus-all:** for each class, train a separate classifier to determine that class versus all others
  - All-pairs:** train a classifier for all pairs of classes
- In both cases choose the class which the classifier is most certain about
- Beware SVMs don't like imbalanced datasets

## Conclusions

- We've seen how SVMs work
- We've learnt how to use them
- We've seen that we can find the maximum margin hyper-plane by solving a quadratic programming problem (with a unique solution)
- This is a convex optimisation problem with a unique optimum
- The **dual problem** of an SVM is particularly simple, especially if we use a positive semi-definite kernel (we explore these in the next lecture)