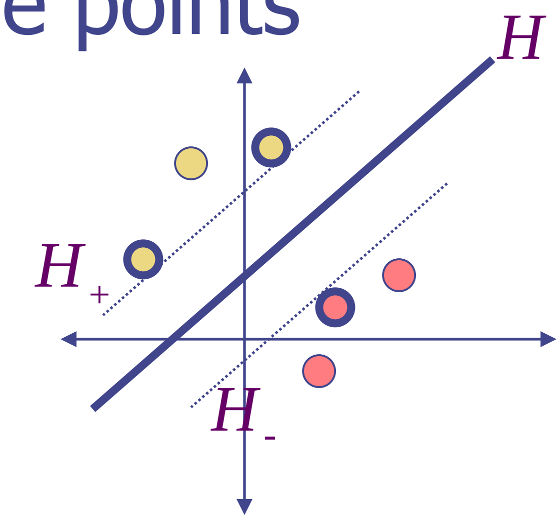
Machine Learning

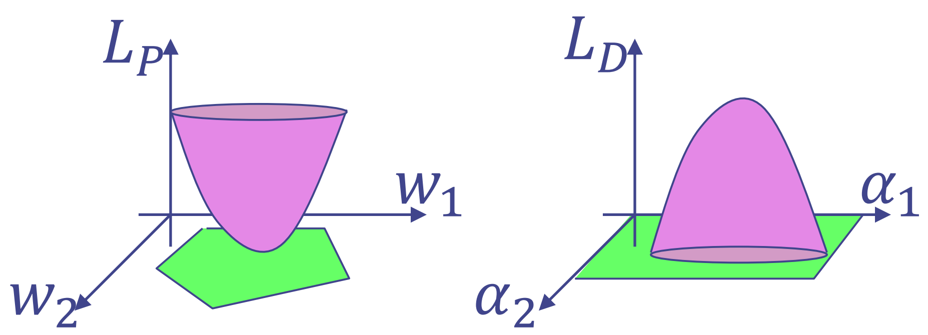
4771

Instructor: Itsik Pe'er

Reminder: SVM

- Linear classifier of separable points
- Maximizes margin
- QP + dual
- Has few support vectors





Duality

Duality

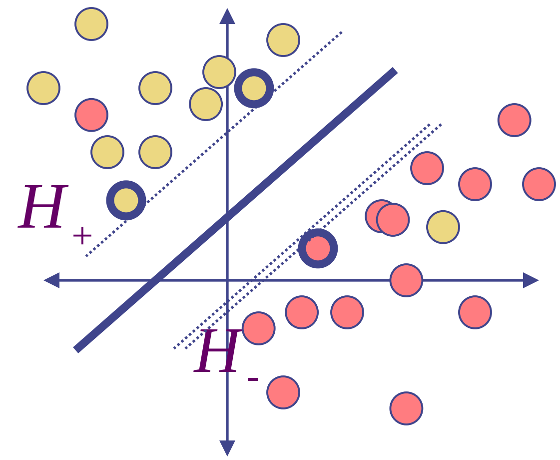
- Primal SVM problem L_P:
  minimize $\frac{1}{2}\|w\|^2$ s.t. $y_i(w^T x_i + b) - 1 \geq 0$
- KKT ~~Lagrange~~ multipliers α_i:
  $$\min_{w,b} \max_{\alpha \geq 0} \frac{1}{2}\|w\|^2 - \sum_i \alpha_i \left(y_i\left(w^T x_i + b\right) - 1\right)$$
- $w = \sum_i \alpha_i y_i x_i$; for $\alpha_i > 0$: $w^T \boldsymbol{x}_i + b = y_i$
- Dual: $L_D = max \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j x_i^T x_j$
  s.t. $\sum_i \alpha_i y_i = 0$, $\alpha_i \geq 0$
- Classifier: $sign(\sum_i \alpha_i y_i x_i \, x + b)$

Class 9 – More SVMs

- Review
- Generalizations
  - Non-separable
  - Non-linear

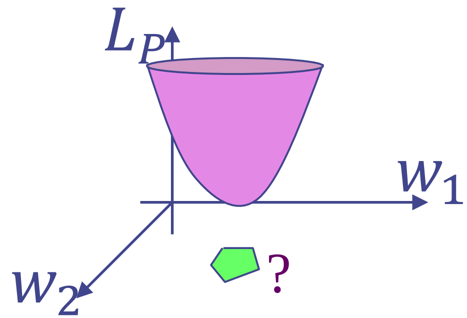
Non-Separable SVMs

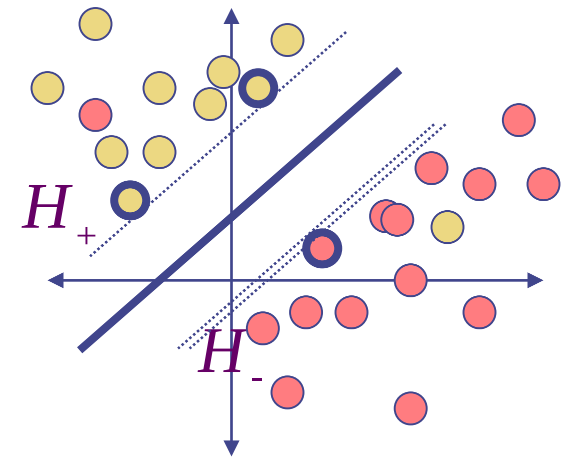
- What happens when non-separable?



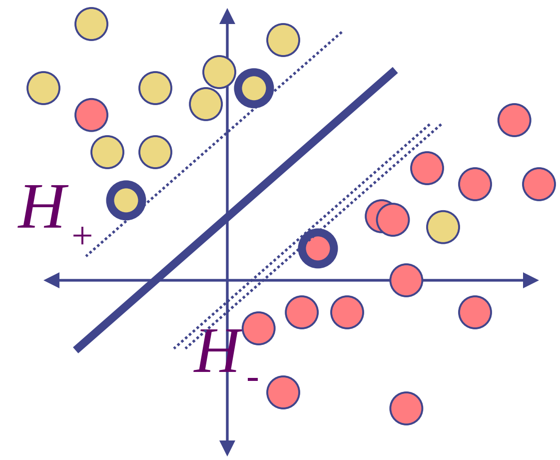
Non-Separable SVMs

- What happens when non-separable?
- There is no solution and convex hull shrinks to nothing

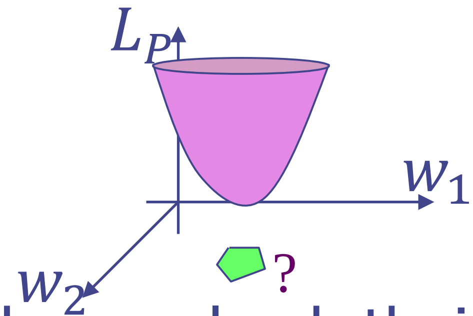




Non-Separable SVMs

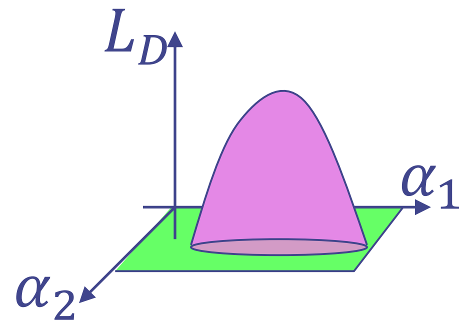


- What happens when non-separable?
- There is no solution and convex hull shrinks to nothing

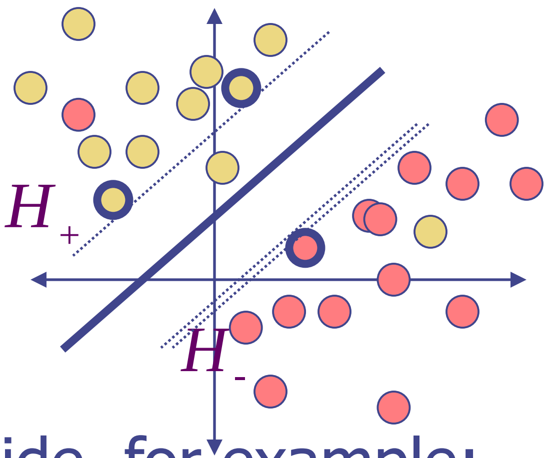


- Not all constraints can be resolved, their alphas go to ∞

$L_D = max \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j x_i^T x_j$ subject to $\sum_i \alpha_i y_i = 0$, $\alpha_i \geq 0$



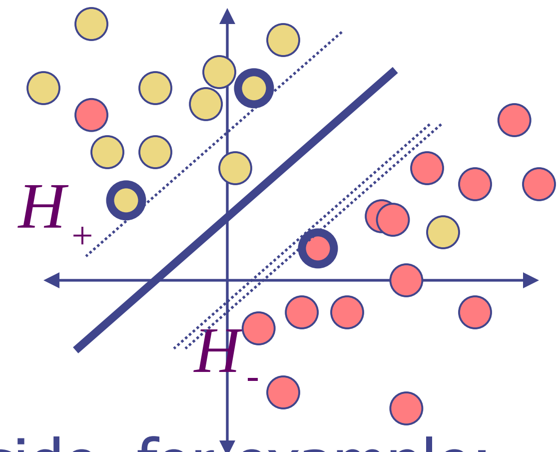
Non-Separable SVMs



- Instead of perfectly classifying each point: $y_i(w^T x_i + b) \geq 1$ we "Relax" the problem with (positive) slack variables ξ's allow data to (sometimes) fall on wrong side, for example:

$$(w^T x_i + b) \geq 1 - 0.03 \text{ if } y_i = +1$$

Non-Separable SVMs



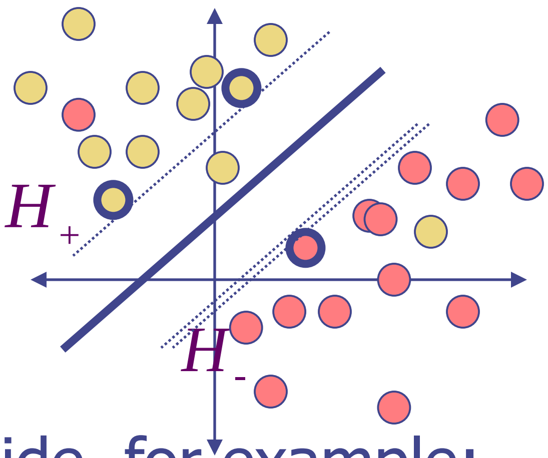
- Instead of perfectly classifying each point: $y_i(w^T x_i + b) \geq 1$ we "Relax" the problem with (positive) slack variables ξ's allow data to (sometimes) fall on wrong side, for example:

$$(w^T x_i + b) \geq 1 - 0.03 \text{ if } y_i = +1$$

- New constraints: $w^T x_i + b \geq +1 - \xi_i \; if \; y_i = +1 \; where \; \xi_i \geq 0$

$$w^T x_i + b \leq -1 + \xi_i \; if \; y_i = -1 \; where \; \xi_i \geq 0$$

Non-Separable SVMs



- Instead of perfectly classifying each point: $y_i(w^T x_i + b) \geq 1$ we "Relax" the problem with (positive) slack variables ξ's allow data to (sometimes) fall on wrong side, for example:

$$(w^T x_i + b) \geq 1 - 0.03 \text{ if } y_i = +1$$

- New constraints: $w^T x_i + b \geq +1 - \xi_i \; if \; y_i = +1 \; where \; \xi_i \geq 0$

$$w^T x_i + b \leq -1 + \xi_i \; if \; y_i = -1 \; where \; \xi_i \geq 0$$

- But too much ξ's means too much slack, so penalize them

$$L_p\colon \min \frac{1}{2}\|w\|^2 + C\sum_i \xi_i \; subject \; to \; y_i(w^T x_i + b) - 1 + \xi_i \geq 0$$

Non-Separable SVMs

- This new problem is still convex, still qp()!
- User chooses scalar C (or cross-validates) which controls how much slack ξ to use (how non-separable) and how robust to outliers or bad points on the wrong side

L_p:

Non-Separable SVMs

- This new problem is still convex, still qp()!
- User chooses scalar C (or cross-validates) which controls how much slack ξ to use (how non-separable) and how robust to outliers or bad points on the wrong side

Low slack **On right side** **For ξ positivity**

Large margin

$$L_p: \min \frac{1}{2}||w||^2 + C\sum_i \xi_i - \sum_i \alpha_i \left(y_i(w^T x_i + b) - 1 + \xi_i\right) - \sum_i \beta_i \xi_i$$

$$\frac{\partial}{\partial w} L_p =$$

$$\frac{\partial}{\partial b} L_p =$$

$$\frac{\partial}{\partial \xi_i} L_p =$$

Non-Separable SVMs

- This new problem is still convex, still qp()!
- User chooses scalar C (or cross-validates) which controls how much slack ξ to use (how non-separable) and how robust to outliers or bad points on the wrong side

Low slack **On right side** **For ξ positivity**

Large margin

$$L_p\colon \min \frac{1}{2}\left|\left|w\right|\right|^2 + C\sum_i \xi_i - \sum_i \alpha_i \left(y_i(w^T x_i + b\ ) - 1 + \xi_i\right) - \sum_i \beta_i \xi_i$$

$$\frac{\partial}{\partial w} L_p \ and \frac{\partial}{\partial b} L_p \ \ as\ before \ldots$$

$$\frac{\partial}{\partial \xi_i} L_p = C - \alpha_i - \beta_i = 0 \qquad \Rightarrow \alpha_i = C - \beta_i$$

$$but\ \alpha_i\ \&\ \beta_i \geq 0 \Rightarrow 0 \leq\ \alpha_i \leq C$$

Non-Separable SVMs

$$L_p : \min \frac{1}{2} ||w||^2 + C \sum_i \xi_i - \sum_i \alpha_i \left(y_i (w^T x_i + b) - 1 + \xi_i \right) - \sum_i \beta_i \xi_i$$

$$\frac{\partial}{\partial w} L_p \ and \ \frac{\partial}{\partial b} L_p \ \ as \ before \ldots$$

$$\frac{\partial}{\partial \xi_i} L_p = C - \alpha_i - \beta_i = 0 \quad \Rightarrow \alpha_i = C - \beta_i$$

$$but \ \alpha_i \ \& \ \beta_i \geq 0 \Rightarrow 0 \leq \ \alpha_i \leq C$$

- Can now write dual problem (to maximize):

L_D:

Non-Separable SVMs

$$L_p: \min \frac{1}{2}\left||w|\right|^2 + C\sum_i \xi_i - \sum_i \alpha_i \left(y_i(w^T x_i + b\ ) - 1 + \xi_i\right) - \sum_i \beta_i \xi_i$$

$$\frac{\partial}{\partial w} L_p \ and \frac{\partial}{\partial b} L_p \ \ as\ before \ldots$$

$$\frac{\partial}{\partial \xi_i} L_p = C - \alpha_i - \beta_i = 0 \quad \Rightarrow \alpha_i = C - \beta_i$$

$$but\ \alpha_i \ \& \ \beta_i \geq 0 \Rightarrow 0 \leq \ \alpha_i \leq C$$

- Can now write dual problem (to maximize):

$$L_D: \max \sum_i \alpha_i - \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i{}^T x_j$$

$$subject\ to \sum_i \alpha_i y_i = 0 \ \ and\ \alpha_i \in [0, C]$$

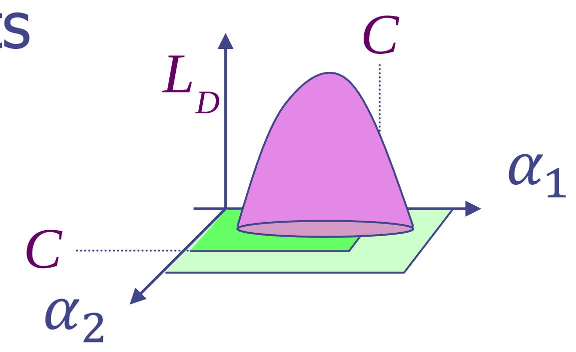
- Same dual as before but alphas can't grow beyond C

Non-Separable SVMs

- As we try to enforce a classification for a data point its KKT multiplier alpha keeps growing endlessly
- Clamping alpha to stop growing at C makes SVM "give up" on those non-separable points

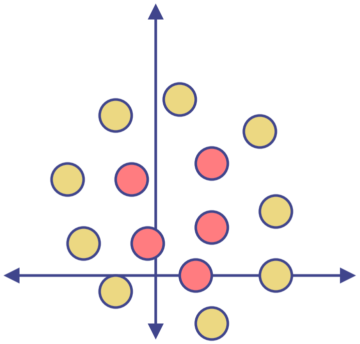
Non-Separable SVMs

- As we try to enforce a classification for a data point its KKT multiplier alpha keeps growing endlessly
- Clamping alpha to stop growing at C makes SVM "give up" on those non-separable points
- The dual program is now:
- Solve as before with extra constraints that alphas positive AND less than C... gives alphas... from alphas get $w = \sum_i \alpha_i y_i x_i$



Nonlinear SVMs

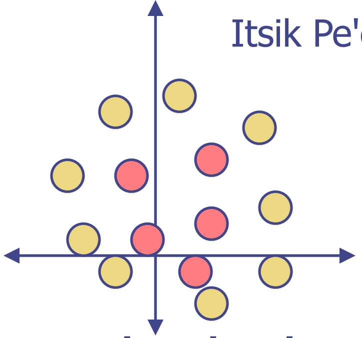
- What if the problem is not linear?



Classifying Antarctica

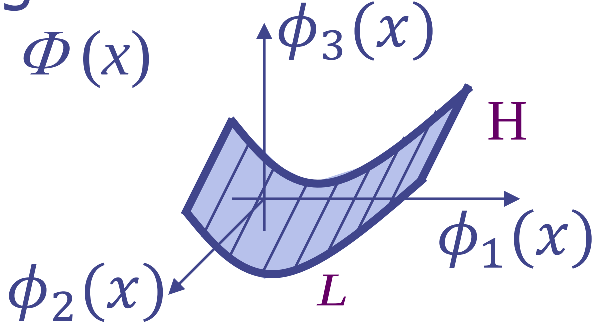
Itsik Pe'er, Columbia University

Nonlinear SVMs

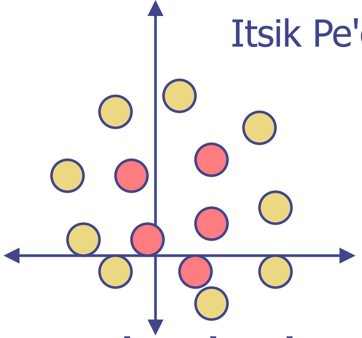


- What if the problem is not linear?
- We can use our old trick…
- Map d-dimensional x data from L-space to high dimensional H (Hilbert) feature-space via basis functions $\Phi(x)$
- For example, quadratic classifier:

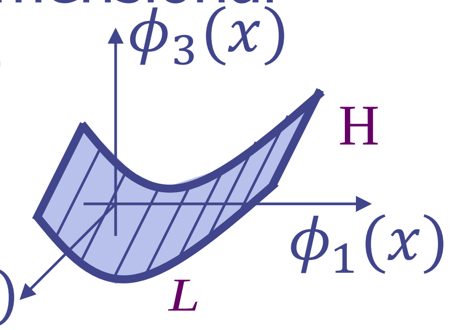
$$x_i \rightarrow \phi(x_i) \;\; via \;\; \phi(\vec{x}) = \begin{bmatrix} \vec{x} \\ vec(\vec{x}\vec{x}^T) \end{bmatrix}$$



Nonlinear SVMs



- What if the problem is not linear?
- We can use our old trick…
- Map d-dimensional x data from L-space to high dimensional H (Hilbert) feature-space via basis functions $\Phi(x)$
- For example, quadratic classifier:

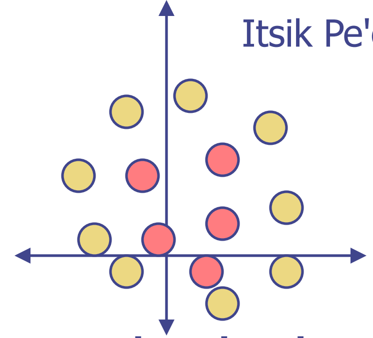


$$x_i \rightarrow \phi(x_i) \;\; via \;\; \phi(\vec{x}) = \begin{bmatrix} \vec{x} \\ vec(\vec{x}\vec{x}^T) \end{bmatrix}$$

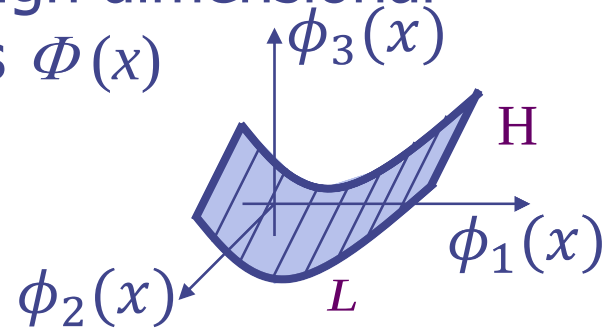
- Call ϕ's feature vectors computed from original x inputs

Itsik Pe'er, Columbia University

Nonlinear SVMs



- What if the problem is not linear?
- Map d-dimensional x data from L-space to high dimensional H (Hilbert) feature-space via basis functions $\Phi(x)$



- For example, quadratic classifier:

$$x_i \to \phi(x_i) \;\; via \;\; \phi(\vec{x}) = \begin{bmatrix} \vec{x} \\ vec(\vec{x}\vec{x}^T) \end{bmatrix}$$

- Call ϕ's feature vectors computed from original x inputs

- Dual qp used to be:

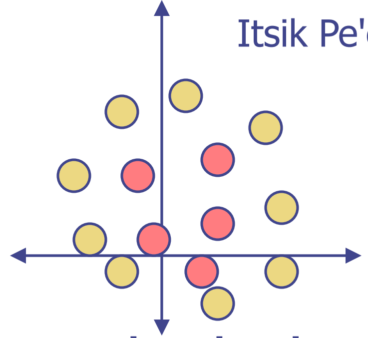
$$L_D\colon \max \sum_i \alpha_i - \sum_{i,j} \alpha_i \alpha_j y_i y_j {x_i}^T x_j \;\; s.t.\, \alpha_i \geq 0\,, \sum_i y_i \alpha_i = 0$$

- With linear classifier in original space:

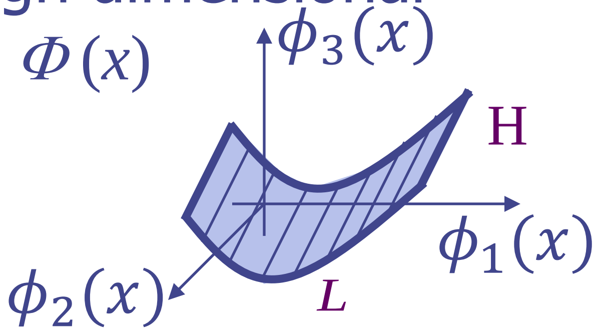
$$f(x) = sign\left(\sum_i \alpha_i y_i {x_i}^T x + b \right)$$

Itsik Pe'er, Columbia University

Nonlinear SVMs



- What if the problem is not linear?
- Map d-dimensional x data from L-space to high dimensional H (Hilbert) feature-space via basis functions $\Phi(x)$
- For example, quadratic classifier:



$$x_i \rightarrow \phi(x_i) \;\; via \;\; \phi(\vec{x}) = \begin{bmatrix} \vec{x} \\ vec(\vec{x}\vec{x}^T) \end{bmatrix}$$

- Call ϕ's feature vectors computed from original x inputs
- Replace all x's in the SVM equations with ϕ's
- Now solve the following learning problem:

$$L_D: \max \sum_i \alpha_i - \sum_{i,j} \alpha_i \alpha_j y_i y_j \phi(x_i)^T \phi(x_j) \;\; s.t. \, \alpha_i \geq 0 \, , \sum_i y_i \alpha_i = 0$$

- Which gives a nonlinear classifier in original space:

$$f(x) = sign\left(\sum_i \alpha_i y_i \phi(x_i)^T \phi(x) + b \right)$$

Kernels (see http://www.youtube.com/watch?v=3liCbRZPrZA)

- One important aspect of SVMs: all math involves only the *inner products* between the ϕ features!

$$f(x) = sign\left(\sum_i \alpha_i y_i \phi(x_i)^T \phi(x_i\ ) + b\right)$$

- Replace all inner products with a general kernel function
- Mercer kernel: accepts 2 inputs and outputs a scalar via:

$$k(x, \tilde{x}) = \langle \phi(x), \phi(\tilde{x}) \rangle = \begin{cases} \phi(x)^T \phi(\tilde{x}) & if\ \phi\ is\ finite \\ \int_t \phi(x,t)\phi(\tilde{x},t)dt & otherwise \end{cases}$$

Kernels (see http://www.youtube.com/watch?v=3liCbRZPrZA)

- Mercer kernel: accepts 2 inputs and outputs a scalar via:

$$k(x,\tilde{x}) = \langle \phi(x), \phi(\tilde{x}) \rangle = \begin{cases} \phi(x)^T \phi(\tilde{x}) & if\ \phi\ is\ finite \\ \int_i \phi(x,t)\phi(\tilde{x},t)dt & otherwise \end{cases}$$

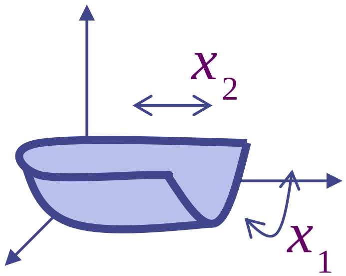
- Example: quadratic polynomial $\phi(x) = \begin{bmatrix} x_1^2 & \sqrt{2}x_1x_2 & x_2^2 \end{bmatrix}$

Kernels (see http://www.youtube.com/watch?v=3liCbRZPrZA)

- Mercer kernel: accepts 2 inputs and outputs a scalar via:

$$k(x,\tilde{x}) = \langle \phi(x), \phi(\tilde{x}) \rangle = \begin{cases} \phi(x)^T \phi(\tilde{x}) & if\ \phi\ is\ finite \\ \int_t \phi(x,t)\phi(\tilde{x},t)dt & otherwise \end{cases}$$

- Example: quadratic polynomial $\phi(x) = \begin{bmatrix} x_1^2 & \sqrt{2}x_1x_2 & x_2^2 \end{bmatrix}$



$$\begin{aligned} k(x,\tilde{x}) &= \phi(x)^T \phi(\tilde{x}) \\ &= x_1^2\tilde{x}_1^2 + 2x_1x_2\tilde{x}_1\tilde{x}_2 + x_2^2\tilde{x}_2^2 \\ &= (x_1\tilde{x}_1 + x_2\tilde{x}_2)^2 \end{aligned}$$

Kernels

- Sometimes, many $\Phi(x)$ will produce the same $k(x,x')$
- Sometimes $k(x,x')$ computable but features huge or infinite!
- Example: polynomials

If explicit polynomial mapping, feature space $\Phi(x)$ is huge

d-dimensional data, p-th order polynomial, $\dim(H) = \binom{d+p-1}{p}$

images of size 16x16 with p=4 have $dim(H)$=183million

Kernels

but can equivalently just use kernel: $k(x, y) = (x^T y)^p$

$k(x, \tilde{x}) =$

Kernels

but can equivalently just use kernel: $k(x,y) = (x^T y)^p$

$$k(x,\tilde{x}) = (x\tilde{x})^p = \left(\sum_i x_i \tilde{x}_i\right)^p$$

Multinomial Theorem

$$\propto \sum_r \frac{p!}{r_1!\, r_2!\, r_3!\ \dots (p - \sum_i r_i)!} x_1^{r_1} x_2^{r_2} \dots x_d^{r_d} \tilde{x}_1^{r_1} \tilde{x}_2^{r_2} \dots \tilde{x}_d^{r_d}$$

w=weight on term

$$\propto \sum_r \left(\sqrt{w_r} x_1^{r_1} x_2^{r_2} \dots x_d^{r_d}\right)\left(\sqrt{w_r} \tilde{x}_1^{r_1} \tilde{x}_2^{r_2} \dots \tilde{x}_d^{r_d}\right)$$

$$\propto \phi(x)\ \phi(\tilde{x})$$

Equivalent!

Kernels

- Replace each $x_i^T x_j \to k(x_i, x_j)$, for example:

  P-th Order Polynomial Kernel: $k(x, \tilde{x}) = (x^T \tilde{x} + 1)^P$

  RBF Kernel (infinite!): $k(x, \tilde{x}) = \exp\left(-\frac{1}{2\sigma^2}\|x - \tilde{x}\|^2\right)$

  Sigmoid (hyperbolic tan) Kernel: $k(x, \tilde{x}) = \tanh(\kappa x^T \tilde{x} - \delta)$

- Using kernels we get generalized inner product SVM:

  L_D: $\max \sum_i \alpha_i - \sum_{i,j} \alpha_i \alpha_j y_i y_j \phi(x_i)^T \phi(x_j)$ $s.t.$ $\alpha_i \in [0, C], \sum_i \alpha_i y_i = 0$

$$f(x) = sign\left(\sum_i \alpha_i y_i k(x_i, x) + b\right)$$

Kernels

- Using kernels we get generalized inner product SVM:

$L_D\colon \max \sum_i \alpha_i - \sum_{i,j} \alpha_i \alpha_j y_i y_j \phi(x_i)^T \phi(x_j) \;\; s.t. \; \alpha_i \in [0, C], \sum_i \alpha_i y_i = 0$

$$f(x) = sign\left(\sum_i \alpha_i y_i k(x_i, x) + b\right)$$

- Still qp solver, just use Gram matrix K (positive definite)

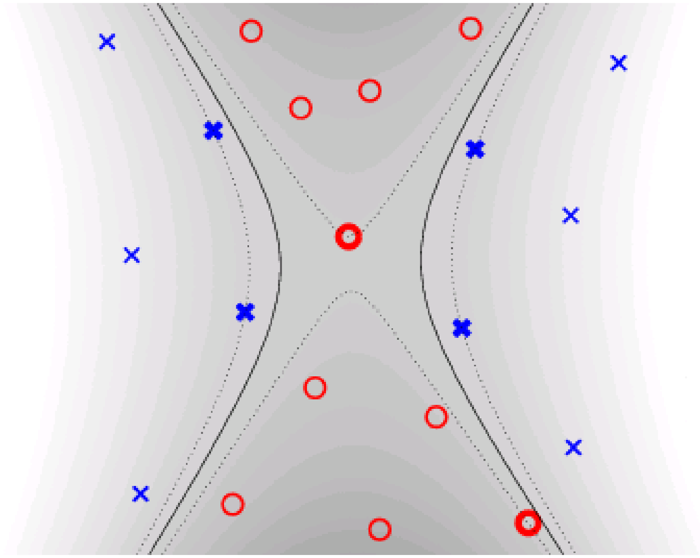
$$K_{i,j} = k(x_i, x_j)$$

$$K = \begin{bmatrix} k(x_1, x_1) & k(x_2, x_1) & k(x_3, x_1) \\ k(x_1, x_2) & k(x_2, x_2) & k(x_3, x_2) \\ k(x_1, x_3) & k(x_2, x_3) & k(x_3, x_3) \end{bmatrix}$$

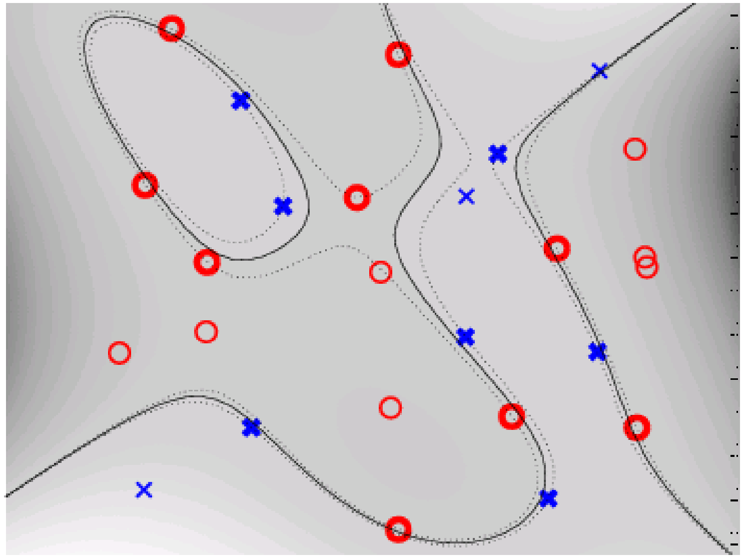
Kernelized SVMs

- Polynomial kernel: $k(x, \tilde{x}) = (x^T \tilde{x} + 1)^P$
- Radial basis function kernel: $k(x, \tilde{x}) = \exp\left(-\frac{1}{2\sigma^2} \|x - \tilde{x}\|^2\right)$

Polynomial Kernel



RBF kernel



- Least-squares, logistic-regression, perceptron: also kernelizable

Summary

- Nonseparable SVM
- Kernels