Support Vector Machines

Pontus Giselsson

Outline

- **Classification**
- Support vector machines
- Nonlinear features
- Overfitting and regularization
- Dual problem
- Kernel SVM
- Training problem properties

Binary classification

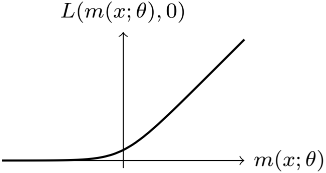
- Labels $y = 0$ or $y = 1$ (alternatively $y = -1$ or $y = 1$)
- Training problem

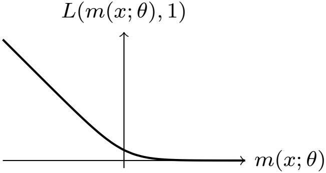
$$\underset{\theta}{\text{minimize}} \sum_{i=1}^{N} L(m(x_i; \theta), y_i)$$

- Design loss L to train model parameters θ such that:
  - $m(x_i; \theta) < 0$ for pairs (x_i, y_i) where $y_i = 0$
  - $m(x_i; \theta) > 0$ for pairs (x_i, y_i) where $y_i = 1$
- Predict class belonging for new data points x with trained $\bar{\theta}$:
  - $m(x; \bar{\theta}) < 0$ predict class $y = 0$
  - $m(x; \bar{\theta}) > 0$ predict class $y = 1$

Binary classification – Cost functions

- Different cost functions L can be used:
  - $y = 0$: Small cost for $m(x;\theta) \ll 0$ large for $m(x;\theta) \gg 0$
  - $y = 1$: Small cost for $m(x;\theta) \gg 0$ large for $m(x;\theta) \ll 0$

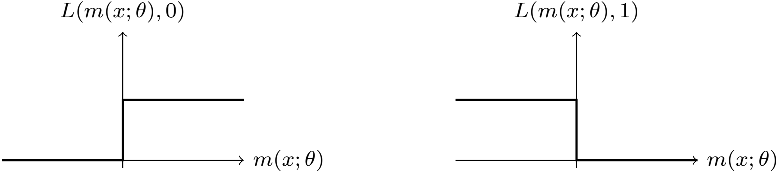




$L(u,y) = \log(1+e^u) - yu$ (logistic loss)

Binary classification – Cost functions

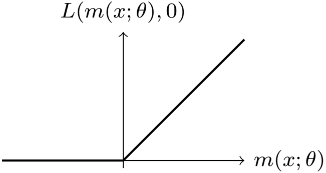
- Different cost functions L can be used:
  - $y = 0$: Small cost for $m(x;\theta) \ll 0$ large for $m(x;\theta) \gg 0$
  - $y = 1$: Small cost for $m(x;\theta) \gg 0$ large for $m(x;\theta) \ll 0$

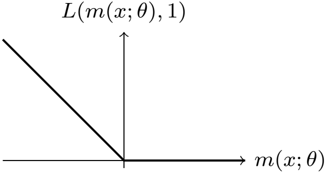


nonconvex (Neyman Pearson loss)

Binary classification – Cost functions

- Different cost functions L can be used:
  - $y = 0$: Small cost for $m(x;\theta) \ll 0$ large for $m(x;\theta) \gg 0$
  - $y = 1$: Small cost for $m(x;\theta) \gg 0$ large for $m(x;\theta) \ll 0$

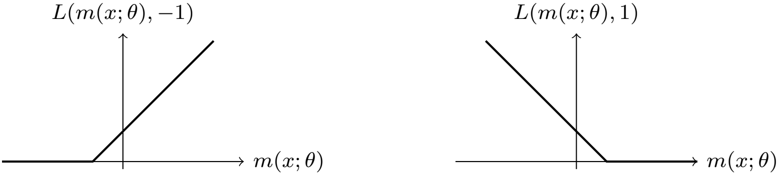




$$L(u, y) = \max(0, u) - yu$$

Binary classification – Cost functions

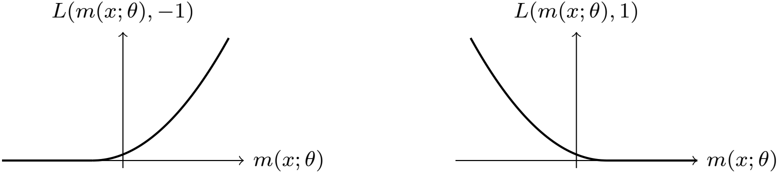
- Different cost functions L can be used:
  - $y = -1$: Small cost for $m(x;\theta) \ll 0$ large for $m(x;\theta) \gg 0$
  - $y = 1$: Small cost for $m(x;\theta) \gg 0$ large for $m(x;\theta) \ll 0$



$L(u, y) = \max(0, 1 - yu)$ (hinge loss used in SVM)

Binary classification – Cost functions

- Different cost functions L can be used:
  - $y = -1$: Small cost for $m(x;\theta) \ll 0$ large for $m(x;\theta) \gg 0$
  - $y = 1$: Small cost for $m(x;\theta) \gg 0$ large for $m(x;\theta) \ll 0$



$L(u, y) = \max(0, 1 - yu)^2$ (squared hinge loss)

Outline

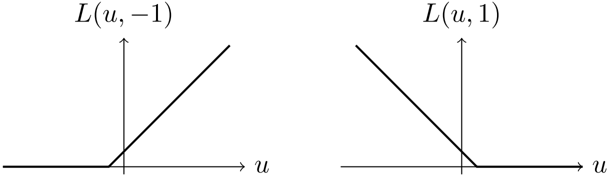
- Classification
- **Support vector machines**
- Nonlinear features
- Overfitting and regularization
- Dual problem
- Kernel SVM
- Training problem properties

Support vector machine

- SVM uses:
  - affine parameterized model $m(x;\theta) = w^T x + b$ (where $\theta = (w, b)$)
  - loss function $L(u, y) = \max(0, 1 - yu)$ (if labels $y = -1$, $y = 1$)
- Training problem, find model parameters by solving:

$$\underset{\theta}{\text{minimize}} \sum_{i=1}^{N} L(m(x_i;\theta), y_i) = \sum_{i=1}^{N} \max(0, 1 - y_i(w^T x_i + b))$$

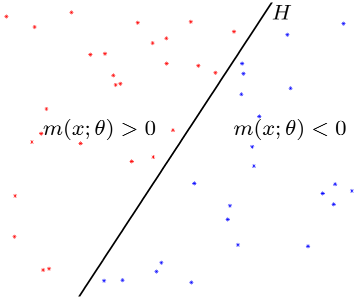
- Training problem convex in $\theta = (w, b)$ since:
  - model $m(x;\theta)$ is affine in θ
  - loss function $L(u, y)$ is convex in u



Prediction

- Use trained model m to predict label y for unseen data point x
- Since affine model $m(x; \theta) = w^T x + b$, prediction for x becomes:
  - If $w^T x + b < 0$, predict corresponding label $y = -1$
  - If $w^T x + b > 0$, predict corresponding label $y = 1$
  - If $w^T x + b = 0$, predict either $y = -1$ or $y = 1$
- A hyperplane (decision boundary) separates class predictions:

$$H := \{x : w^T x + b = 0\}$$

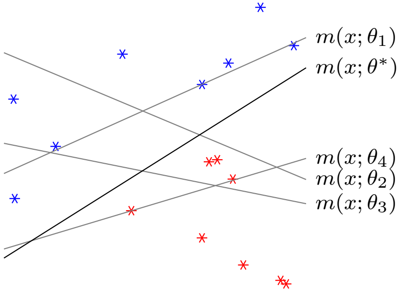


Training problem interpretation

- Every parameter choice $\theta = (w, b)$ gives hyperplane in data space:

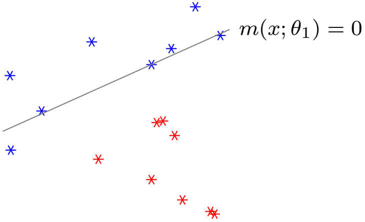
$$H := \{x : w^T x + b = 0\} = \{x : m(x;\theta) = 0\}$$

- Training problem searches hyperplane to "best" separates classes
- Example – models with different parameters θ:

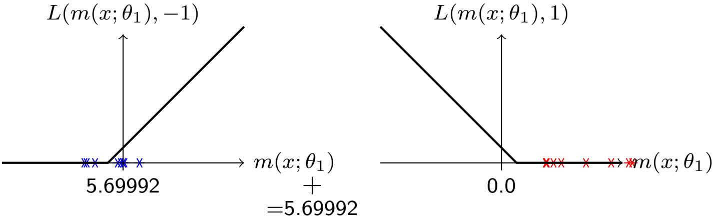


What is "best" separation?

- The "best" separation is the one that minimizes the loss function
- Hyperplane for model $m(\cdot;\theta)$ with parameter $\theta = \theta_1$:

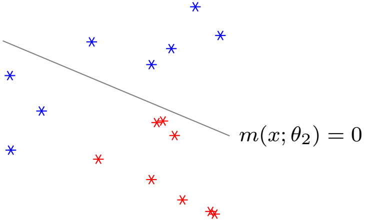


- Training loss:

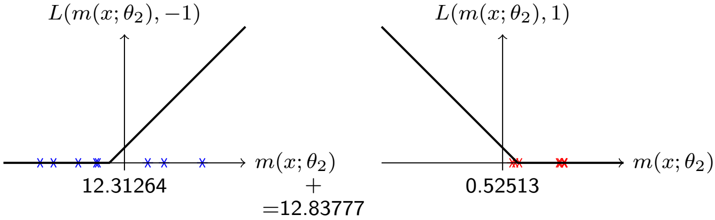


What is "best" separation?

- The "best" separation is the one that minimizes the loss function
- Hyperplane for model $m(\cdot;\theta)$ with parameter $\theta = \theta_2$:

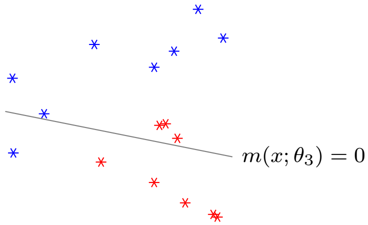


- Training loss:

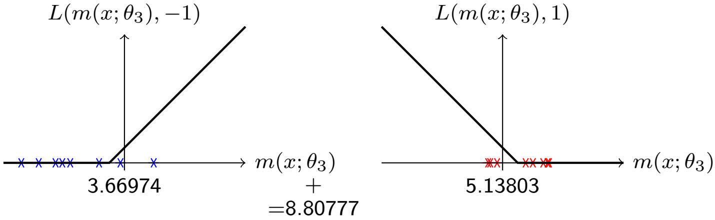


What is "best" separation?

- The "best" separation is the one that minimizes the loss function
- Hyperplane for model $m(\cdot;\theta)$ with parameter $\theta = \theta_3$:

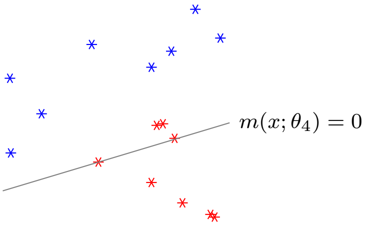


- Training loss:

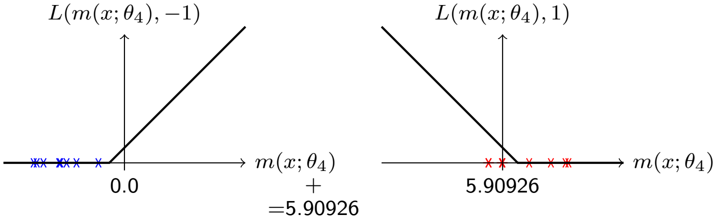


What is “best” separation?

- The “best” separation is the one that minimizes the loss function
- Hyperplane for model $m(\cdot;\theta)$ with parameter $\theta = \theta_4$:

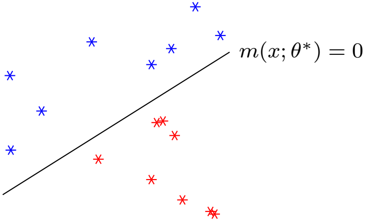


- Training loss:

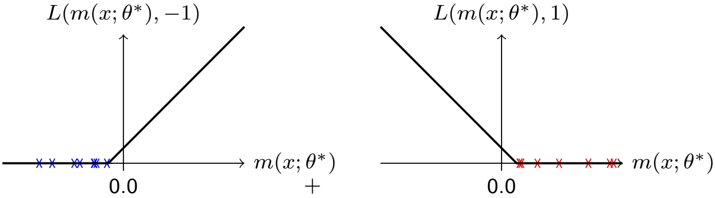


What is "best" separation?

- The "best" separation is the one that minimizes the loss function
- Hyperplane for model $m(\cdot;\theta)$ with parameter $\theta = \theta^*$:



- Training loss:

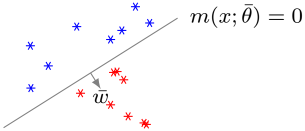


$+$

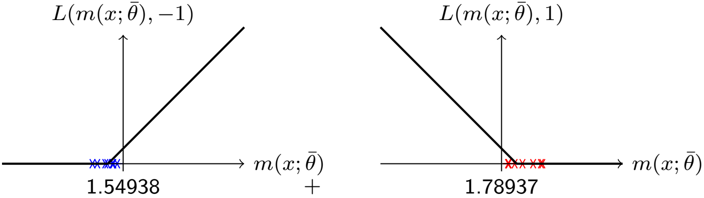
$=0.0$

Fully separable data – Solution

- Let $\bar{\theta} = (\bar{w}, \bar{b})$ give model that separates data:



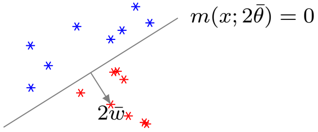
- Let $H_{\bar{\theta}} := \{x : m(x;\bar{\theta}) = \bar{w}^T x + \bar{b} = 0\}$ be hyperplane separates
- Training loss:



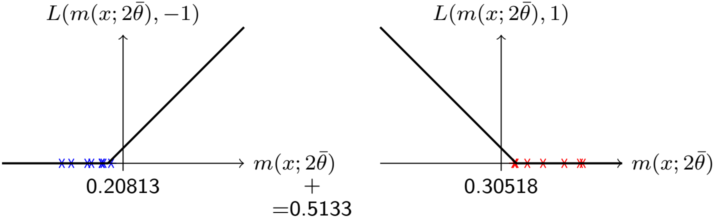
$+$
$=3.33875$

Fully separable data – Solution

- Also $2\bar{\theta} = (2\bar{w}, 2\bar{b})$ separates data:

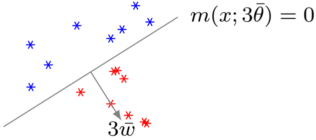


- Hyperplane $H_{2\bar{\theta}} := \{x : m(x; 2\bar{\theta}) = 2(\bar{w}^T x + \bar{b}) = 0\} = H_{\bar{\theta}}$ same
- Training loss reduced since input $m(x; 2\bar{\theta}) = 2m(x; \bar{\theta})$ further out:

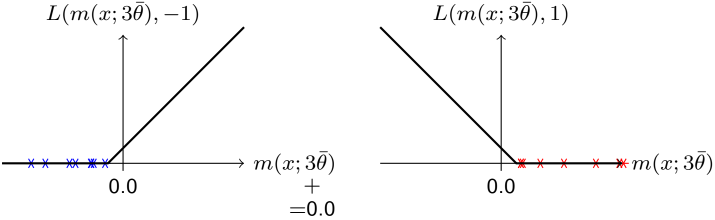


Fully separable data – Solution

- And $3\bar{\theta} = (3\bar{w}, 3\bar{b})$ also separates data:

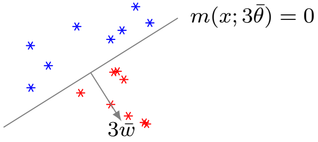


- Hyperplane $H_{3\bar{\theta}} := \{x : m(x; 3\bar{\theta}) = 3(\bar{w}^T x + \bar{b}) = 0\} = H_{\bar{\theta}}$ same
- Training loss further reduced since input $m(x; 3\bar{\theta}) = 3m(x; \bar{\theta})$:

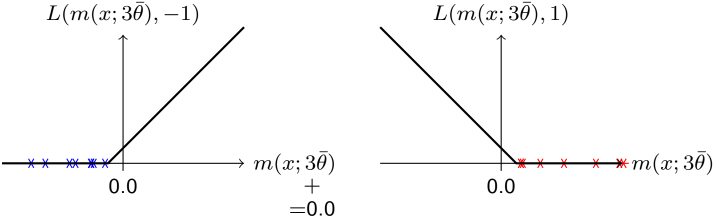


Fully separable data – Solution

- And $3\bar{\theta} = (3\bar{w}, 3\bar{b})$ also separates data:



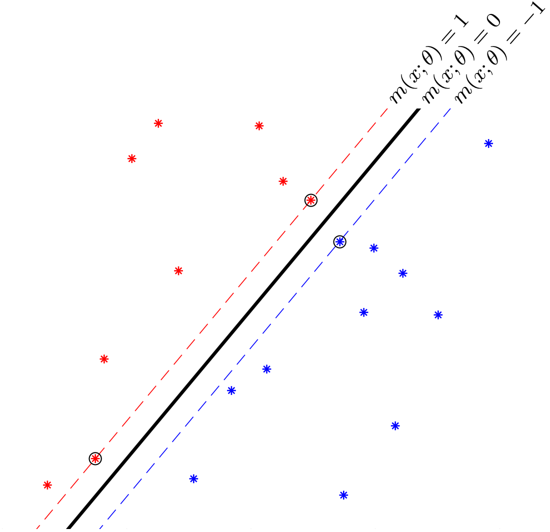
- Hyperplane $H_{3\bar{\theta}} := \{x : m(x; 3\bar{\theta}) = 3(\bar{w}^T x + \bar{b}) = 0\} = H_{\bar{\theta}}$ same
- Training loss



- As soon as $|m(x_i; \theta)| \geq 1$ (with correct sign) for all x_i, cost is 0

Margin classification and support vectors

- Support vector machine classifiers for separable data
- Classes separated with margin, $\circ$ marks *support vectors*

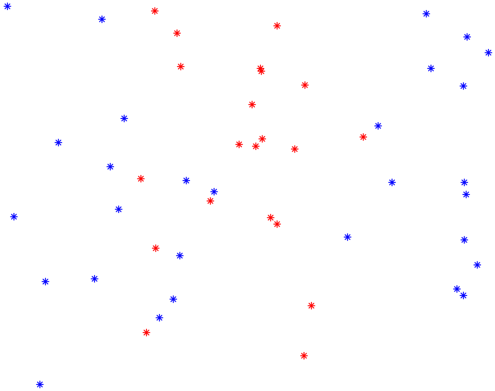


Outline

- Classification
- Support vector machines
- **Nonlinear features**
- Overfitting and regularization
- Dual problem
- Kernel SVM
- Training problem properties

Nonlinear example

- Can classify nonlinearly separable data using lifting



Adding features

- Create feature map $\phi : \mathbb{R}^n \to \mathbb{R}^p$ of training data
- Data points $x_i \in \mathbb{R}^n$ replaced by featured data points $\phi(x_i) \in \mathbb{R}^p$
- Example: Polynomial feature map with $n = 2$ and degree $d = 3$

$$\phi(x) = (x_1, x_2, x_1^2, x_1 x_2, x_2^2, x_1^3, x_1^2 x_2, x_1 x_2^2, x_2^3)$$

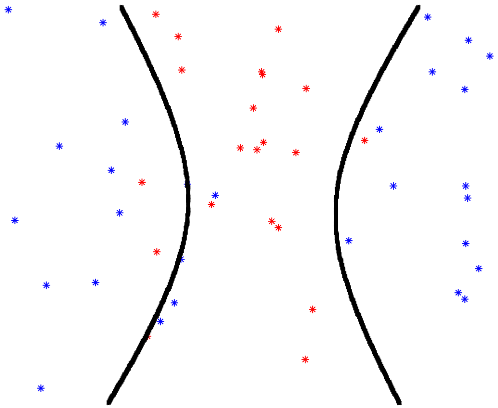
- Number of features $p + 1 = \binom{n+d}{d} = \frac{(n+d)!}{d!n!}$ grows fast!
- SVM training problem

$$\underset{\theta}{\text{minimize}} \sum_{i=1}^{N} \max(0, 1 - y_i(w^T \phi(x_i) + b))$$

still convex since features fixed

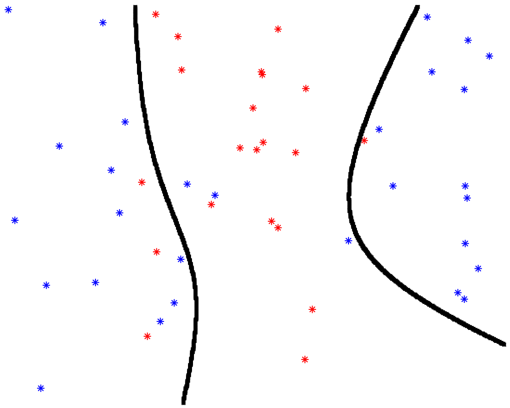
Nonlinear example – Polynomial features

- SVM and polynomial features of degree 2



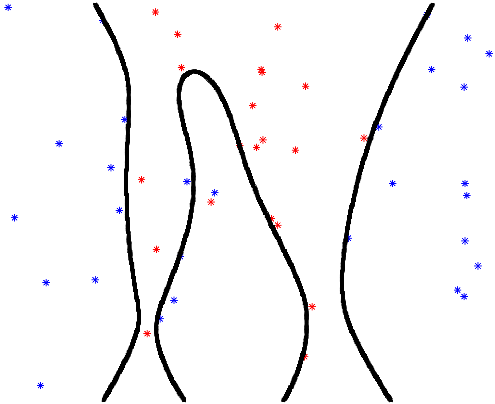
Nonlinear example – Polynomial features

- SVM and polynomial features of degree 3



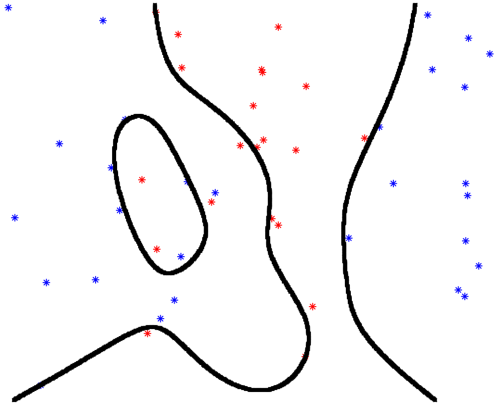
Nonlinear example – Polynomial features

- SVM and polynomial features of degree 4



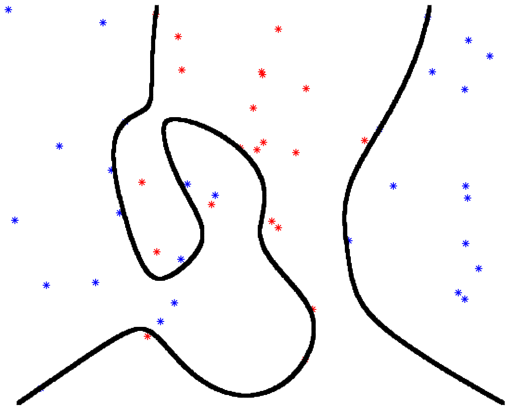
Nonlinear example – Polynomial features

- SVM and polynomial features of degree 5



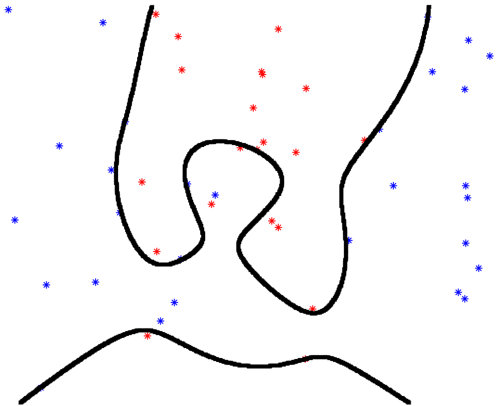
Nonlinear example – Polynomial features

- SVM and polynomial features of degree 6



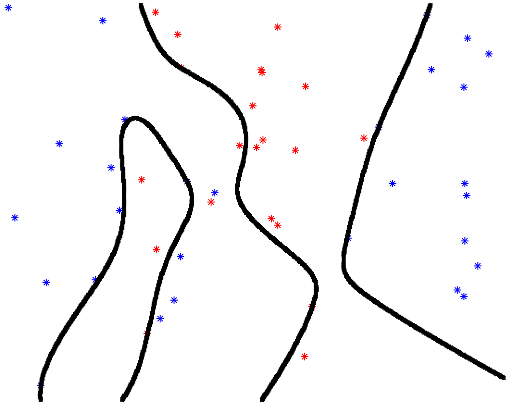
Nonlinear example – Polynomial features

- SVM and polynomial features of degree 7



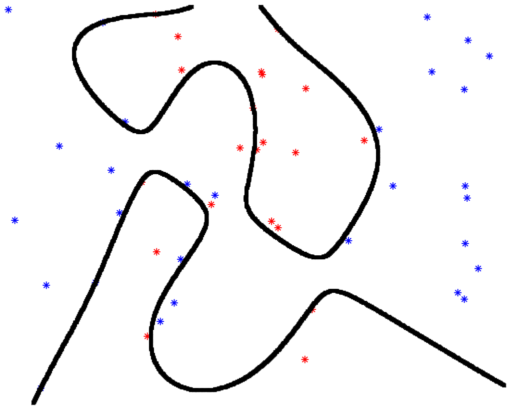
Nonlinear example – Polynomial features

- SVM and polynomial features of degree 8



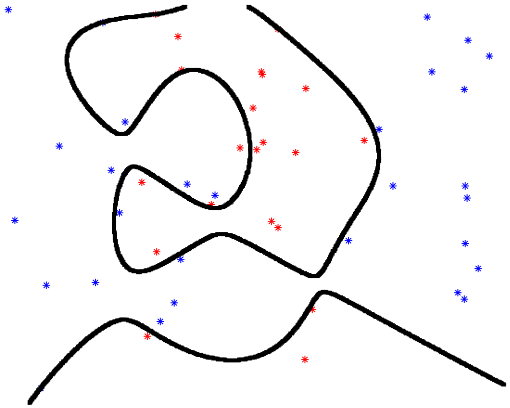
Nonlinear example – Polynomial features

- SVM and polynomial features of degree 9



Nonlinear example – Polynomial features

- SVM and polynomial features of degree 10



Outline

- Classification
- Support vector machines
- Nonlinear features
- **Overfitting and regularization**
- Dual problem
- Kernel SVM
- Training problem properties

Overfitting and regularization

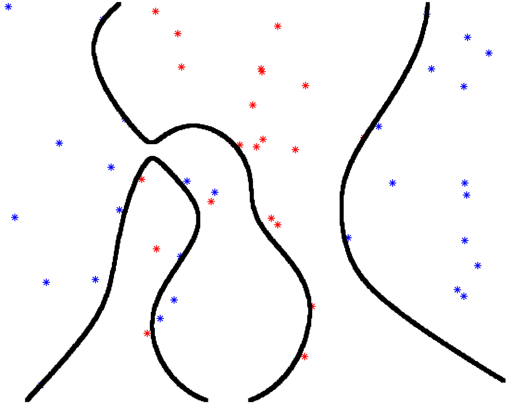
- SVM is prone to overfitting if model too expressive
- Regularization using $\|\cdot\|_1$ (for sparsity) or $\|\cdot\|_2^2$
- Tikhonov regularization with $\|\cdot\|_2^2$ especially important for SVM
- Regularize only linear terms w, not bias b
- Training problem with Tikhonov regularization of w

$$\underset{\theta}{\text{minimize}} \sum_{i=1}^{N} \max(0, 1 - y_i(w^T\phi(x_i) + b)) + \frac{\lambda}{2}\|w\|_2^2$$

(note that features are used $\phi(x_i)$)

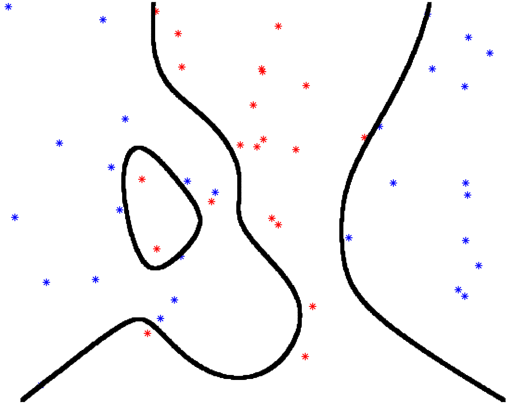
Nonlinear example revisited

- Regularized SVM and polynomial features of degree 6
- Regularization parameter: $\lambda = 0.00001$



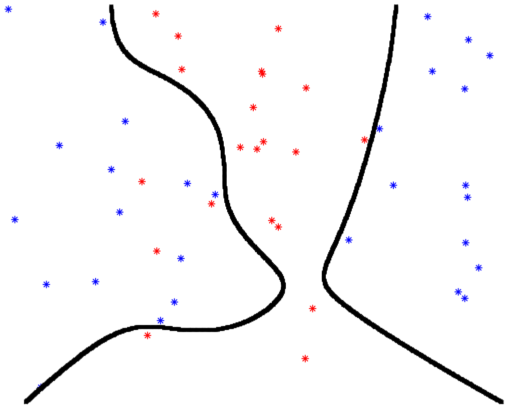
Nonlinear example revisited

- Regularized SVM and polynomial features of degree 6
- Regularization parameter: $\lambda = 0.00006$



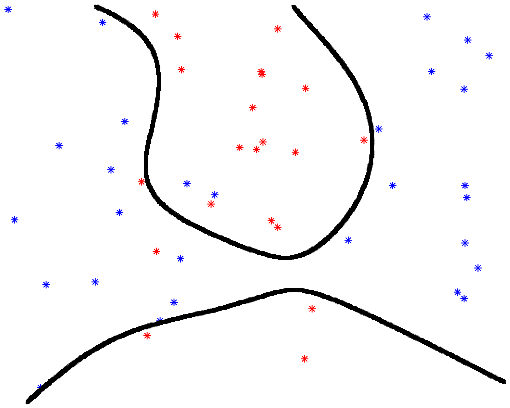
Nonlinear example revisited

- Regularized SVM and polynomial features of degree 6
- Regularization parameter: $\lambda = 0.00036$



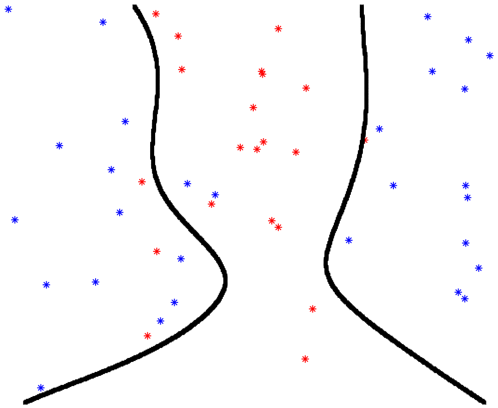
Nonlinear example revisited

- Regularized SVM and polynomial features of degree 6
- Regularization parameter: $\lambda = 0.0021$



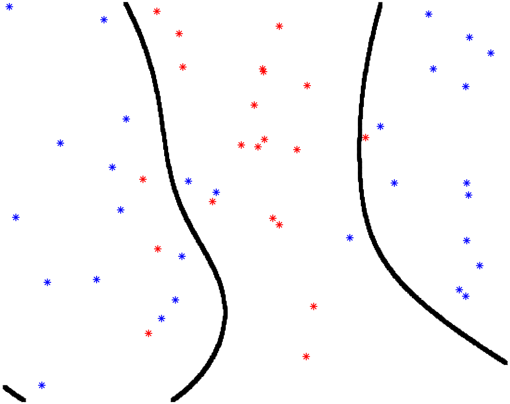
Nonlinear example revisited

- Regularized SVM and polynomial features of degree 6
- Regularization parameter: $\lambda = 0.013$



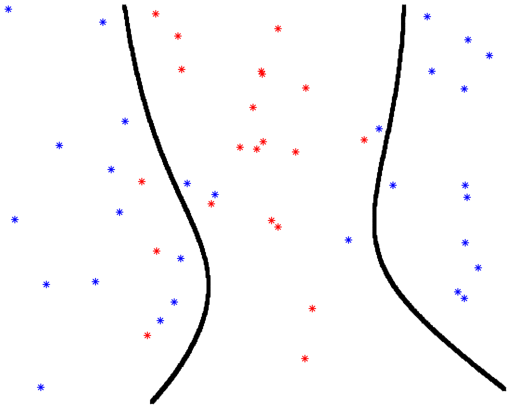
Nonlinear example revisited

- Regularized SVM and polynomial features of degree 6
- Regularization parameter: $\lambda = 0.077$



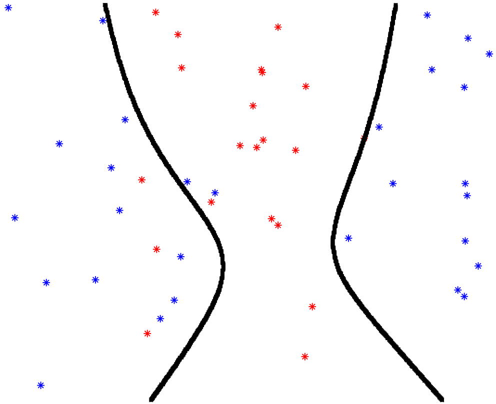
Nonlinear example revisited

- Regularized SVM and polynomial features of degree 6
- Regularization parameter: $\lambda = 0.46$



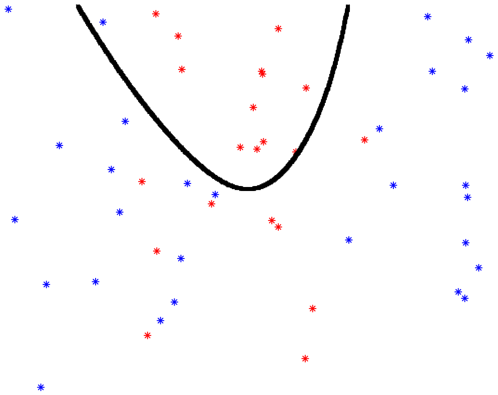
Nonlinear example revisited

- Regularized SVM and polynomial features of degree 6
- Regularization parameter: $\lambda = 2.78$



Nonlinear example revisited

- Regularized SVM and polynomial features of degree 6
- Regularization parameter: $\lambda = 16.7$



- λ and polynomial degree chosen using cross validation/holdout

Outline

- Classification
- Support vector machines
- Nonlinear features
- Overfitting and regularization
- **Dual problem**
- Kernel SVM
- Training problem properties

SVM problem reformulation

- Consider Tikhonov regularized SVM:

$$\underset{w,b}{\text{minimize}} \sum_{i=1}^{N} \max(0, 1 - y_i(w^T\phi(x_i) + b)) + \tfrac{\lambda}{2}\|w\|_2^2$$

- Derive dual from reformulation of SVM:

$$\underset{w,b}{\text{minimize}}\, \mathbf{1}^T \max(\mathbf{0}, \mathbf{1} - (X_{\phi,Y}w + Yb)) + \tfrac{\lambda}{2}\|w\|_2^2$$

where $\max$ is vector valued and

$$X_{\phi,Y} = \begin{bmatrix} y_1\phi(x_1)^T \\ \vdots \\ y_N\phi(x_N)^T \end{bmatrix}, \qquad Y = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix}$$

Dual problem

- Let $L = [X_{\phi,Y}, Y]$ and write problem as

$$\underset{w,b}{\text{minimize}} \underbrace{\mathbf{1}^T \max(\mathbf{0}, \mathbf{1} - (X_{\phi,Y} w + Y b))}_{f(L(w,b))} + \underbrace{\tfrac{\lambda}{2}\|w\|_2^2}_{g(w,b)}$$

  where

  - $f(\psi) = \sum_{i=1}^N f_i(\psi_i)$ and $f_i(\psi_i) = \max(0, 1 - \psi_i)$ (hinge loss)
  - $g(w, b) = \frac{\lambda}{2}\|w\|_2^2$, i.e., does not depend on b

- Dual problem

$$\underset{\nu}{\text{minimize}}\, f^*(\nu) + g^*(-L^T \nu)$$

Conjugate of g

- Conjugate of $g(w,b) = \frac{\lambda}{2}\|w\|_2^2 =: g_1(w) + g_2(b)$ is

$$g^*(\mu_w, \mu_b) = g_1^*(\mu_w) + g_2^*(\mu_b) = \tfrac{1}{2\lambda}\|\mu_w\|_2^2 + \iota_{\{0\}}(\mu_b)$$

- Evaluated at $-L^T\nu = -[X_{\phi,Y}, Y]^T\nu$:

$$\begin{aligned} g^*(-L^T\nu) &= g^*\left(-\begin{bmatrix} X_{\phi,Y}^T \\ Y^T \end{bmatrix}\nu\right) = \tfrac{1}{2\lambda}\|-X_{\phi,Y}^T\nu\|_2^2 + \iota_{\{0\}}(-Y^T\nu) \\ &= \tfrac{1}{2\lambda}\nu^T X_{\phi,Y} X_{\phi,Y}^T \nu + \iota_{\{0\}}(Y^T\nu) \end{aligned}$$

Conjugate of f

- Conjugate of $f_i(\psi_i) = \max(0, 1 - \psi_i)$ (hinge-loss):

$$f_i^*(\nu_i) = \begin{cases} \nu_i & \text{if } -1 \leq \nu_i \leq 0 \\ \infty & \text{else} \end{cases}$$

- Conjugate of $f(\psi) = \sum_{i=1}^N f_i(\psi_i)$ is sum of individual conjugates:

$$f^*(\nu) = \sum_{i=1}^N f_i^*(\nu_i) = \mathbf{1}^T \nu + \iota_{[-\mathbf{1},\mathbf{0}]}(\nu)$$

SVM dual

- The SVM dual is

$$\underset{\nu}{\text{minimize}}\, f^*(\nu) + g^*(-L^T\nu)$$

- Inserting the above computed conjugates gives dual problem

$$\begin{array}{ll} \underset{\nu}{\text{minimize}} & \sum_{i=1}^N \nu_i + \frac{1}{2\lambda}\nu^T X_{\phi,Y} X_{\phi,Y}^T \nu \\ \text{subject to} & -\mathbf{1} \leq \nu \leq \mathbf{0} \\ & Y^T\nu = 0 \end{array}$$

- Since $Y \in \mathbb{R}^N$, $Y^T\nu = 0$ is a hyperplane constraint
- If no bias term b; dual same but without hyperplane constraint

Primal solution recovery

- Meaningless to solve dual if we cannot recover primal
- Necessary and sufficient primal-dual optimality conditions

$$0 \in \begin{cases} \partial f^*(\nu) - L(w,b) \\ \partial g^*(-L^T\nu) - (w,b) \end{cases}$$

- From dual solution ν, find (w,b) that satisfies both of the above
- For SVM, second condition is

$$\partial g^*(-L^T\nu) = \begin{bmatrix} \frac{1}{\lambda}(-X_{\phi,Y}^T\nu) \\ \partial\iota_{\{0\}}(-Y^T\nu) \end{bmatrix} \ni \begin{bmatrix} w \\ b \end{bmatrix}$$

  which gives optimal $w = -\frac{1}{\lambda}X_{\Phi,Y}^T\nu$ (since unique)
- Cannot recover b from this condition

Primal solution recovery – Bias term

- Necessary and sufficient primal-dual optimality conditions

$$0 \in \begin{cases} \partial f^*(\nu) - L(w,b) \\ \partial g^*(-L^T\nu) - (w,b) \end{cases}$$

- For SVM, row i of first condition is $0 \in \partial f_i^*(\nu_i) - L_i(w,b)$ where

$$\partial f_i^*(\nu_i) = \begin{cases} [-\infty, 1] & \text{if } \nu_i = -1 \\ \{1\} & \text{if } -1 < \nu_i < 0 \\ [1, \infty] & \text{if } \nu_i = 0 \\ \emptyset & \text{else} \end{cases}, \quad L_i = y_i[\phi(x_i)^T \ 1]$$

- Pick i with $\nu_i \in (-1, 0)$, then unique subgradient $\partial f_i(\nu_i)$ is 1 and

$$0 = 1 - y_i(w^T\phi(x_i) + b)$$

and optimal b must satisfy $b = y_i - w^T\phi(x_i)$ for such i

Outline

- Classification
- Support vector machines
- Nonlinear features
- Overfitting and regularization
- Dual problem
- **Kernel SVM**
- Training problem properties

SVM dual – A reformulation

- Dual problem

$$\begin{array}{ll} \underset{\nu}{\text{minimize}} & \sum_{i=1}^{N} \nu_i + \frac{1}{2\lambda}\nu^T X_{\phi,Y} X_{\phi,Y}^T \nu \\ \text{subject to} & -\mathbf{1} \leq \nu \leq \mathbf{0} \\ & Y^T \nu = 0 \end{array}$$

- Let $\kappa_{ij} := \phi(x_i)^T \phi(x_j)$ and rewrite quadratic term:

$$\begin{aligned} \nu^T X_{\phi,Y} X_{\phi,Y}^T \nu &= \nu \, \mathbf{diag}(Y) \begin{bmatrix} \phi(x_1)^T \\ \vdots \\ \phi(x_N)^T \end{bmatrix} \begin{bmatrix} \phi(x_1) & \cdots & \phi(x_N) \end{bmatrix} \mathbf{diag}(Y)\nu \\ &= \nu \, \mathbf{diag}(Y) \underbrace{\begin{bmatrix} \kappa_{11} & \cdots & \kappa_{1N} \\ \vdots & \ddots & \vdots \\ \kappa_{N1} & \cdots & \kappa_{NN} \end{bmatrix}}_{K} \mathbf{diag}(Y)\nu \end{aligned}$$

where K is called *Kernel matrix*

SVM dual – Kernel formulation

- Dual problem with Kernel matrix

$$\begin{array}{ll} \underset{\nu}{\text{minimize}} & \sum_{i=1}^{N} \nu_i + \frac{1}{2\lambda}\nu^T \,\mathbf{diag}(Y) K \,\mathbf{diag}(Y)\nu \\ \text{subject to} & -\mathbf{1} \leq \nu \leq \mathbf{0} \\ & Y^T\nu = 0 \end{array}$$

- Solved without evaluating features, only scalar products:

$$\kappa_{ij} := \phi(x_i)^T \phi(x_j)$$

Kernel methods

- We explicitly defined features and created Kernel matrix
- We can instead create Kernel that implicitly defines features

Kernel operators

- Define:
  - Kernel operator $\kappa(x, y) : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$
  - Kernel shortcut $\kappa_{ij} = \kappa(x_i, x_j)$
  - A Kernel matrix

$$K = \begin{bmatrix} \kappa_{11} & \cdots & \kappa_{1N} \\ \vdots & \ddots & \vdots \\ \kappa_{N1} & \cdots & \kappa_{NN} \end{bmatrix}$$

- A Kernel operator $\kappa : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$ is:
  - *symmetric* if $\kappa(x, y) = \kappa(y, x)$
  - *positive semidefinite* (PSD) if symmetric and

$$\sum_{i,j}^{m} a_i a_j \kappa(x_i, x_j) \geq 0$$

  for all $m \in \mathbb{N}$, $\alpha_i, \alpha_j \in \mathbb{R}$, and $x_i, x_j \in \mathbb{R}^n$

- All Kernel matrices PSD if Kernel operator PSD

Mercer's theorem

- Assume κ is a positive semidefinite Kernel operator
- Mercer's theorem:

There exists continuous functions $\{e_j\}_{j=1}^{\infty}$ and nonnegative $\{\lambda_j\}_{j=1}^{\infty}$ such that

$$\kappa(x,y) = \sum_{j=1}^{\infty} \lambda_j e_j(x) e_j(y)$$

- Let $\phi(x) = (\sqrt{\lambda_1} e_1(x), \sqrt{\lambda_2} e_2(x), \ldots)$ be a feature map, then

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

  where scalar product in ℓ_2 (space of square summable sequences)
- A PSD kernel operator implicitly defines features

Kernel SVM dual and corresponding primal

- SVM dual from Kernel κ with Kernel matrix $K_{ij} = \kappa(x_i, x_j)$

$$\begin{array}{ll} \underset{\nu}{\text{minimize}} & \sum_{i=1}^{N} \nu_i + \frac{1}{2\lambda}\nu\, \mathbf{diag}(Y) K\, \mathbf{diag}(Y)\nu \\ \text{subject to} & -\mathbf{1} \leq \nu \leq \mathbf{0} \\ & Y^T \nu = 0 \end{array}$$

- Due to Mercer's theorem, this is dual to primal problem

$$\underset{\theta}{\text{minimize}} \sum_{i=1}^{N} \max(0, 1 - y_i(\langle w, \phi(x_i)\rangle + b)) + \tfrac{\lambda}{2}\|w\|^2$$

with potentially an infinite number of features ϕ and variables w

Primal recovery and class prediction

- Assume we know Kernel operator, dual solution, but not features
  - Can recover: Label prediction and primal solution b
  - Cannot recover: Primal solution w (might be infinite dimensional)
- Primal solution $b = y_i - w^T\phi(x_i)$:

$$w^T\phi(x_i) = -\tfrac{1}{\lambda}\nu^T X_{\phi,Y}\phi(x_i) = -\tfrac{1}{\lambda}\nu^T \begin{bmatrix} y_1\phi(x_1)^T \\ \vdots \\ y_N\phi(x_N)^T \end{bmatrix} \phi(x_i) = -\tfrac{1}{\lambda}\nu^T \begin{bmatrix} y_1\kappa_{1i} \\ \vdots \\ y_N\kappa_{Ni} \end{bmatrix}$$

- Label prediction for new data x (sign of $w^T\phi(x) + b$):

$$w^T\phi(x) + b = -\tfrac{1}{\lambda}\nu^T \begin{bmatrix} y_1\phi(x_1)^T\phi(x) \\ \vdots \\ y_N\phi(x_N)^T\phi(x) \end{bmatrix} + b = -\tfrac{1}{\lambda}\nu^T \begin{bmatrix} y_1\kappa(x_1, x) \\ \vdots \\ y_N\kappa(x_N, x) \end{bmatrix} + b$$

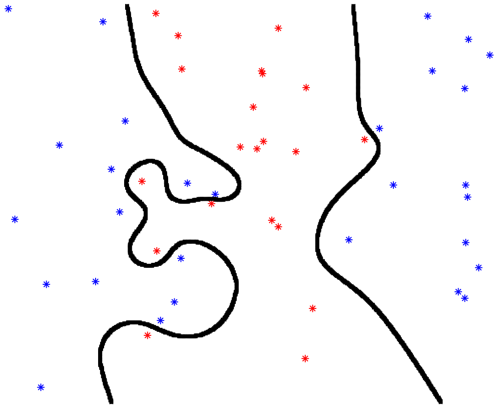
- We are really interested in label prediction, not primal solution

Valid kernels

- Polynomial kernel of degree d: $\kappa(x, y) = (1 + x^T y)^d$
- Radial basis function kernels:
  - Gaussian kernel: $\kappa(x, y) = e^{-\frac{\|x-y\|_2^2}{2\sigma^2}}$
  - Laplacian kernel: $\kappa(x, y) = e^{-\frac{\|x-y\|_2}{\sigma}}$
- Bias term b often not needed with Kernel methods

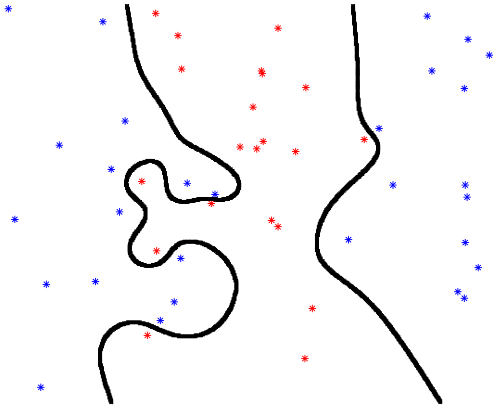
Example – Laplacian Kernel

- Regularized SVM with Laplacian Kernel with $\sigma = 1$
- Regularization parameter: $\lambda = 0.01$



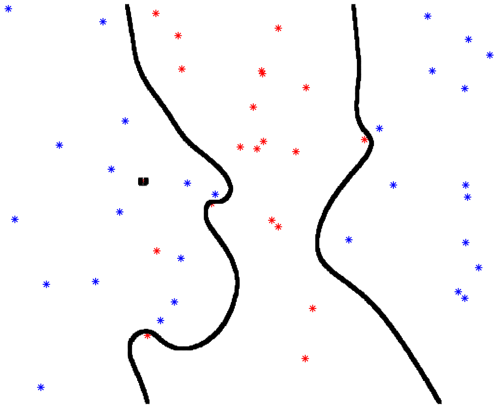
Example – Laplacian Kernel

- Regularized SVM with Laplacian Kernel with $\sigma = 1$
- Regularization parameter: $\lambda = 0.035938$



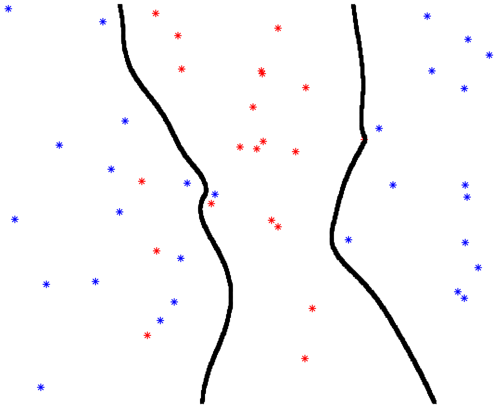
Example – Laplacian Kernel

- Regularized SVM with Laplacian Kernel with $\sigma = 1$
- Regularization parameter: $\lambda = 0.12915$



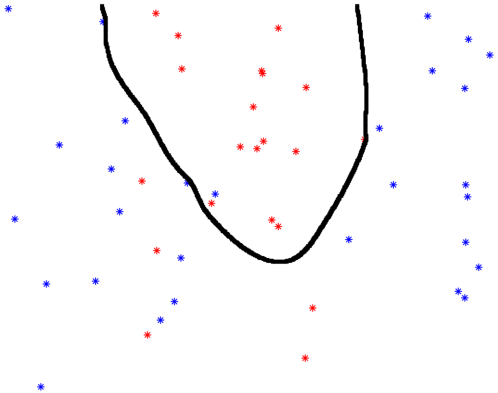
Example – Laplacian Kernel

- Regularized SVM with Laplacian Kernel with $\sigma = 1$
- Regularization parameter: $\lambda =$ 0.46416



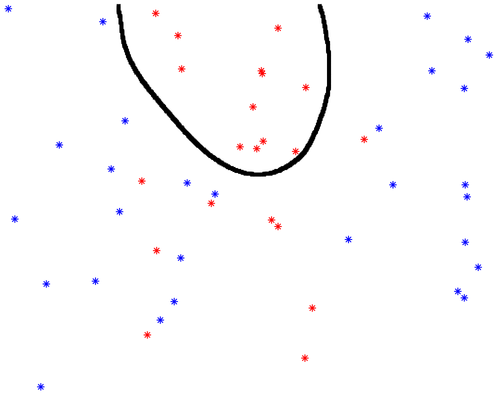
Example – Laplacian Kernel

- Regularized SVM with Laplacian Kernel with $\sigma = 1$
- Regularization parameter: $\lambda = 1.6681$



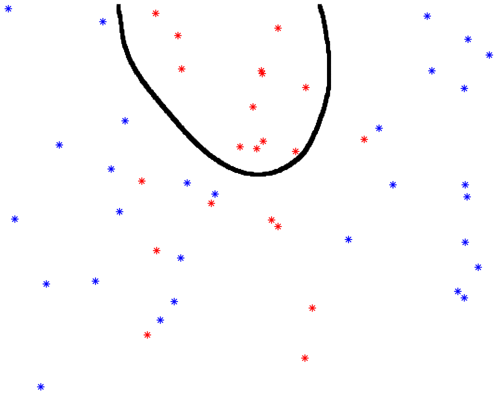
Example – Laplacian Kernel

- Regularized SVM with Laplacian Kernel with $\sigma = 1$
- Regularization parameter: $\lambda = 5.9948$



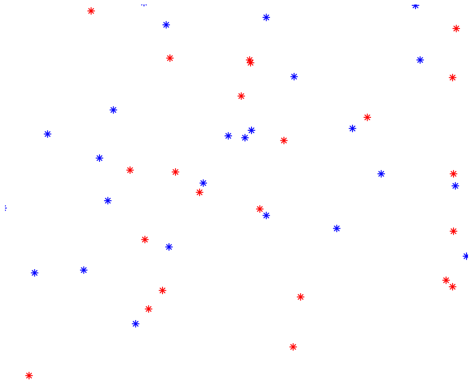
Example – Laplacian Kernel

- Regularized SVM with Laplacian Kernel with $\sigma = 1$
- Regularization parameter: $\lambda = 21.5443$



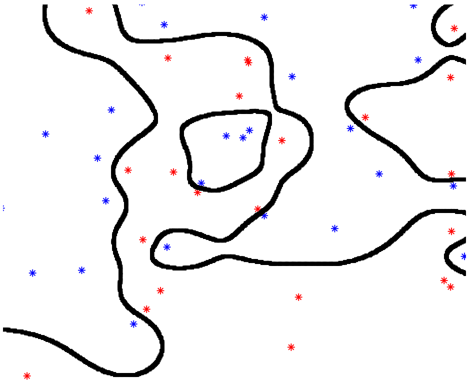
Example – Laplacian Kernel

- What if there is no structure in data? (Labels are randomly set)



Example – Laplacian Kernel

- What if there is no structure in data? (Labels are randomly set)
- Regularized SVM Laplacian Kernel, regularization parameter: $\lambda = 0.01$



- Linearly separable in high dimensional feature space
- Can be prone to overfitting $\Rightarrow$ Regularize and use cross validation

Outline

- Classification
- Support vector machines
- Nonlinear features
- Overfitting and regularization
- Dual problem
- Kernel SVM
- **Training problem properties**

Composite optimization – Dual SVM

Dual SVM problems

$$\begin{array}{ll} \underset{\nu}{\text{minimize}} & \sum_{i=1}^N \nu_i + \frac{1}{2\lambda}\nu^T X_{\phi,Y} X_{\phi,Y}^T \nu \\ \text{subject to} & -\mathbf{1} \leq \nu \leq \mathbf{0} \\ & Y^T\nu = 0 \end{array}$$

can be written on the form

$$\underset{\nu}{\text{minimize}}\, h_1(\nu) + h_2(-X_{\phi,Y}^T\nu),$$

where

- $h_1(\nu) = \mathbf{1}^T\nu + \iota_{[-\mathbf{1},\mathbf{0}]}(\nu) + \iota_{\{0\}}(Y^T\nu)$
  - First part $\mathbf{1}^T\nu + \iota_{[-\mathbf{1},\mathbf{0}]}(\nu)$ is conjugate of sum of hinge losses
  - Second part $\iota_{\{0\}}(Y^T\nu)$ comes from that bias b not regularized
- $h_2(\mu) = \frac{1}{2\lambda}\|\mu\|_2^2$ is conjugate to Tikhonov regularization $\frac{\lambda}{2}\|w\|_2^2$

Gradient and function properties

- Gradient of $(h_2 \circ -X_{\phi,Y}^T)$ satisfies:

$$\begin{aligned}\nabla(h_2 \circ -X_{\phi,Y}^T)(\nu) &= \nabla\left(\tfrac{1}{2\lambda}\nu^T X_{\phi,Y} X_{\phi,Y}^T \nu\right) = \tfrac{1}{\lambda} X_{\phi,Y} X_{\phi,Y}^T \nu \\ &= \tfrac{1}{\lambda}\,\mathbf{diag}(Y) K\,\mathbf{diag}(Y)\nu\end{aligned}$$

  where K is Kernel matrix
- Function properties
  - h_2 is convex and λ^{-1}-smooth, $h_2 \circ -X_{\phi,Y}^T$ is $\frac{\|X_{\phi,Y}\|_2^2}{\lambda}$-smooth
  - h_1 is convex and nondifferentiable, use prox in algorithms