Machine Learning - MT 2017
13 Support Vector Machines II

Christoph Haase

University of Oxford
November 6, 2017

Last Time

- Primal Formuation of SVM
- Slack variables for linearly non-separable data

SVM Formulation : Non-Separable Case

minimise: $\frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{N}\zeta_i$

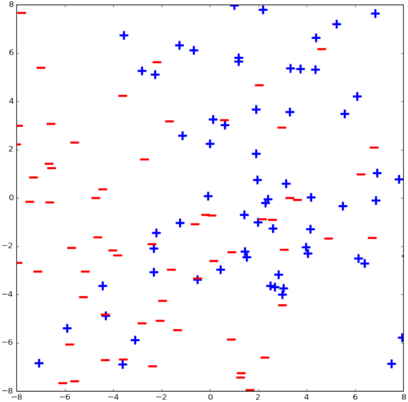
subject to:

$y_i(\mathbf{w}\cdot\mathbf{x}_i + w_0) \geq 1 - \zeta_i$

$\zeta_i \geq 0$

for $i = 1, \ldots, N$

Here $y_i \in \{-1, 1\}$



SVM Formulation : Loss Function

minimise: $$\underbrace{\frac{1}{2}\|\mathbf{w}\|_2^2}_{\text{Regularizer}} + \underbrace{C\sum_{i=1}^{N}\zeta_i}_{\text{Loss Function}}$$

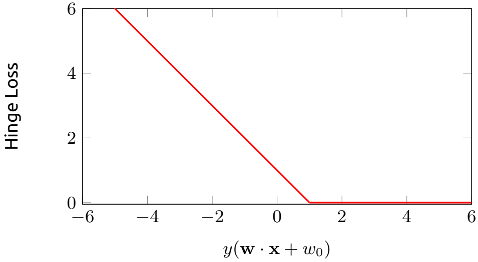
subject to:

$$y_i(\mathbf{w}\cdot\mathbf{x}_i + w_0) \geq 1 - \zeta_i$$

$$\zeta_i \geq 0$$

for $i = 1, \ldots, N$

Here $y_i \in \{-1, 1\}$



Note that for the optimal solution, $\zeta_i = \max\{0, 1 - y_i(\mathbf{w}\cdot\mathbf{x}_i + w_0)\}$

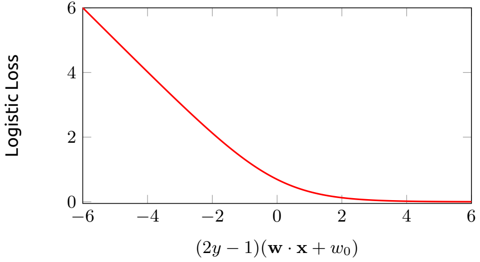
Thus, SVM can be viewed as minimizing the hinge loss with regularization

Logistic Regression: Loss Function

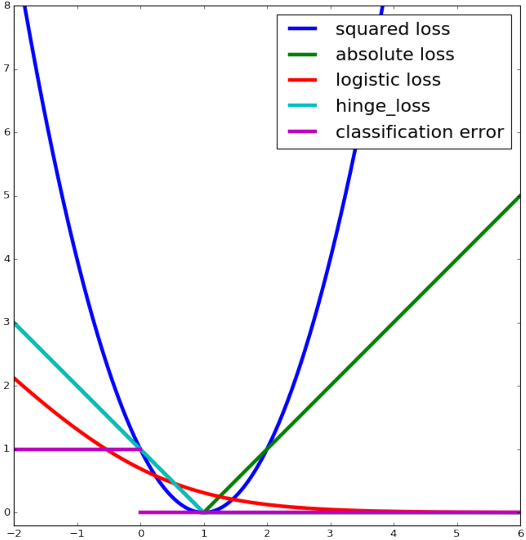
Here $y_i \in \{0, 1\}$, so to compare effectively to SVM, let $z_i = (2y_i - 1)$:

- $z_i = 1$ if $y_i = 1$
- $z_i = -1$ if $y_i = 0$

$$\mathrm{NLL}(y_i; \mathbf{w}, \mathbf{x}_i) = -\left(y_i \log\left(\frac{1}{1 + e^{-\mathbf{w}\cdot\mathbf{x}_i}}\right) + (1 - y_i)\log\left(\frac{1}{1 + e^{\mathbf{w}\cdot\mathbf{x}_i}}\right)\right)$$
$$= \log\left(1 + e^{-z_i(\mathbf{w}\cdot\mathbf{x}_i)}\right) = \log\left(1 + e^{-(2y_i - 1)(\mathbf{w}\cdot\mathbf{x}_i)}\right)$$



Loss Functions



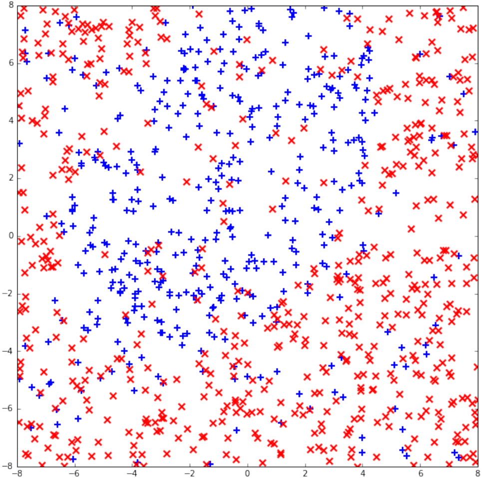
Outline

Dual Formulation of SVM

Kernels

SVM Formulation: Non-Separable Case

What if your data looks like this?



SVM Formulation : Constrained Minimisation

minimise: $\frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{N}\zeta_i$

subject to:

$$y_i(\mathbf{w}\cdot\mathbf{x}_i + w_0) - (1-\zeta_i) \geq 0$$

$$\zeta_i \geq 0$$

for $i = 1, \ldots, N$

Here $y_i \in \{-1, 1\}$

Contrained Optimisation with Inequalities

Primal Form

$$\begin{aligned} \text{minimise} \quad & F(\mathbf{z}) & \\ \text{subject to} \quad & g_i(\mathbf{z}) \geq 0 & i = 1, \ldots, m \\ & h_j(\mathbf{z}) = 0 & j = 1, \ldots, l \end{aligned}$$

Lagrange Function

$$\Lambda(\mathbf{z}; \alpha, \boldsymbol{\mu}) = F(\mathbf{z}) - \sum_{i=1}^{m} \alpha_i g_i(\mathbf{z}) - \sum_{j=1}^{l} \mu_j h_j(\mathbf{z})$$

For convex problems (as defined before), Karush-Kuhn-Tucker (KKT) conditions provide necessary and sufficient conditions for a critical point of Λ to be the minimum of the original constrained optimisation problem

For non-convex problems, they are necessary but not sufficient

KKT Conditions

Lagrange Function

$$\Lambda(\mathbf{z}; \boldsymbol{\alpha}, \boldsymbol{\mu}) = F(\mathbf{z}) - \sum_{i=1}^{m} \alpha_i g_i(\mathbf{z}) - \sum_{j=1}^{l} \mu_j h_j(\mathbf{z})$$

For convex problems, Karush-Kuhn-Tucker (KKT) conditions give necessary and sufficient conditions for a solution (critical point of Λ) to be optimal

Dual feasibility:	$\alpha_i \geq 0$	for $i = 1, \ldots m$
Primal feasibility:	$g_i(\mathbf{z}) \geq 0$	for $i = 1, \ldots m$
	$h_j(\mathbf{z}) = 0$	for $j = 1, \ldots l$
Complementary slackness:	$\alpha_i g_i(\mathbf{z}) = 0$	for $i = 1, \ldots m$

SVM Formulation

minimise: $\frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{N}\zeta_i$

subject to:

$$y_i(\mathbf{w}\cdot\mathbf{x}_i + w_0) - (1-\zeta_i) \geq 0$$

$$\zeta_i \geq 0$$

for $i = 1, \ldots, N$

Here $y_i \in \{-1, 1\}$

Lagrange Function

$$\Lambda(\mathbf{w}, w_0, \boldsymbol{\zeta}; \boldsymbol{\alpha}, \boldsymbol{\mu}) = \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{N}\zeta_i - \sum_{i=1}^{N}\alpha_i(y_i(\mathbf{w}\cdot\mathbf{x}_i + w_0) - (1-\zeta_i)) - \sum_{i=1}^{N}\mu_i\zeta_i$$

SVM Dual Formulation

Lagrange Function

$$\Lambda(\mathbf{w}, w_0, \boldsymbol{\zeta}; \boldsymbol{\alpha}, \boldsymbol{\mu}) = \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{N}\zeta_i - \sum_{i=1}^{N}\alpha_i(y_i(\mathbf{w}\cdot\mathbf{x}_i + w_0) - (1-\zeta_i)) - \sum_{i=1}^{N}\mu_i\zeta_i$$

We write derivatives with respect to $\mathbf{w}$, w_0 and ζ_i,

$$\frac{\partial\Lambda}{\partial w_0} = -\sum_{i=1}^{N}\alpha_i y_i$$

$$\frac{\partial\Lambda}{\partial\zeta_i} = C - \alpha_i - \mu_i$$

$$\nabla_{\mathbf{w}}\Lambda = \mathbf{w} - \sum_{i=1}^{N}\alpha_i y_i \mathbf{x}_i$$

For (KKT) dual feasibility constraints, we require $\alpha_i \geq 0$, $\mu_i \geq 0$

SVM Dual Formulation

Setting the derivatives to 0, substituting the resulting expressions in Λ (and simplifying), we get a function $g(\boldsymbol{\alpha})$ and some constraints

$$g(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j$$

Constraints

$$0 \leq \alpha_i \leq C \qquad\qquad i = 1, \ldots, N$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0$$

Finding critical points of Λ satisfying the KKT conditions corresponds to finding the maximum of $g(\boldsymbol{\alpha})$ subject to the above constraints

SVM: Primal and Dual Formulations

Primal Form

minimise: $\frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{N}\zeta_i$

subject to:

$$y_i(\mathbf{w}\cdot\mathbf{x}_i + w_0) \geq (1 - \zeta_i)$$

$$\zeta_i \geq 0$$

for $i = 1, \dots, N$

Dual Form

maximise $\sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \mathbf{x}_i\cdot\mathbf{x}_j$

subject to:

$$\textstyle\sum_{i=1}^{N}\alpha_i y_i = 0$$

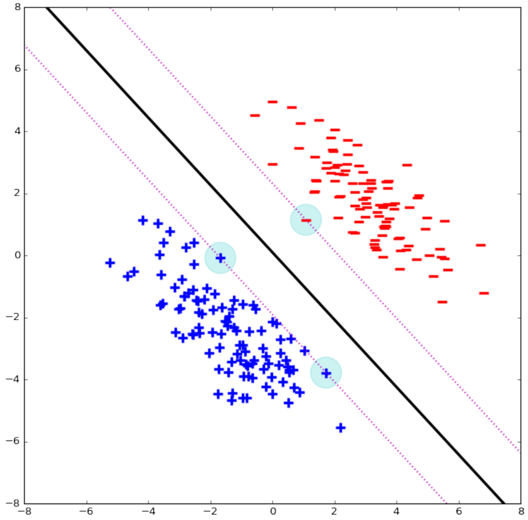
$$0 \leq \alpha_i \leq C$$

for $i = 1, \dots, N$

KKT Complementary Slackness Conditions

- For all i, $\alpha_i \left(y_i(\mathbf{w} \cdot \mathbf{x}_i + w_0) - (1 - \zeta_i) \right) = 0$
- If $\alpha_i > 0$, $y_i(\mathbf{w} \cdot \mathbf{x}_i + w_0) = 1 - \zeta_i$
- Recall the form of the solution: $\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$
- Thus, only those datapoints $\mathbf{x}_i$ for which $\alpha_i > 0$, determine the solution
- This is why they are called support vectors

Support Vectors



SVM Dual Formulation

maximise $\displaystyle\sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^{\mathsf{T}} \mathbf{x}_j$

subject to:

$$0 \leq \alpha_i \leq C \quad i = 1, \ldots, N$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0$$

- Objective depends only between dot products of training inputs
- Dual formulation particularly useful if inputs are high-dimensional
- Dual constraints are much simpler than primal ones
- To make a new prediction only need to know dot product with support vectors
  - Solution is of the form $\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$
  - And so $\mathbf{w} \cdot \mathbf{x}_{\text{new}} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x}_{\text{new}}$

Outline

Dual Formulation of SVM

Kernels

Gram Matrix

If we put the inputs in matrix $\mathbf{X}$, where the i^{th} row of $\mathbf{X}$ is $\mathbf{x}_i^\top$.

$$\mathbf{K} = \mathbf{X}\mathbf{X}^\top = \begin{bmatrix} \mathbf{x}_1^\top \mathbf{x}_1 & \mathbf{x}_1^\top \mathbf{x}_2 & \cdots & \mathbf{x}_1^\top \mathbf{x}_N \\ \mathbf{x}_2^\top \mathbf{x}_1 & \mathbf{x}_2^\top \mathbf{x}_2 & \cdots & \mathbf{x}_2^\top \mathbf{x}_N \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{x}_N^\top \mathbf{x}_1 & \mathbf{x}_N^\top \mathbf{x}_2 & \cdots & \mathbf{x}_N^\top \mathbf{x}_N \end{bmatrix}$$

- The matrix $\mathbf{K}$ is positive definite if $D > N$ and $\mathbf{x}_i$ are linearly independent
- If we perform basis expansion

$$\phi : \mathbb{R}^D \rightarrow \mathbb{R}^M$$

  then replace entries by $\phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j)$
- We only need the ability to compute inner products to use SVM

Kernel Trick

Suppose, $\mathbf{x} \in \mathbb{R}^2$ and we perform degree 2 polynomial expansion, we could use the map:

$$\psi(\mathbf{x}) = \left[1, x_1, x_2, x_1^2, x_2^2, x_1 x_2\right]^{\mathsf{T}}$$

But, we could also use the map:

$$\phi(\mathbf{x}) = \left[1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2, \sqrt{2}x_1 x_2\right]^{\mathsf{T}}$$

If $\mathbf{x} = [x_1, x_2]^{\mathsf{T}}$ and $\mathbf{x}' = [x_1', x_2']^{\mathsf{T}}$, then

$$\begin{aligned}\phi(\mathbf{x})^{\mathsf{T}}\phi(\mathbf{x}') &= 1 + 2x_1x_1' + 2x_2x_2' + x_1^2(x_1')^2 + x_2^2(x_2')^2 + 2x_1x_2x_1'x_2' \\ &= (1 + x_1x_1' + x_2x_2')^2 = (1 + \mathbf{x} \cdot \mathbf{x}')^2\end{aligned}$$

Instead of spending $\approx D^d$ time to compute inner products after degree d polynomial basis expansion, we only need $O(D)$ time

Kernel Trick

We can use a symmetric positive semi-definite matrix (Mercer Kernels)

$$\mathbf{K} = \begin{bmatrix} \kappa(\mathbf{x}_1, \mathbf{x}_1) & \kappa(\mathbf{x}_1, \mathbf{x}_2) & \cdots & \kappa(\mathbf{x}_1, \mathbf{x}_N) \\ \kappa(\mathbf{x}_2, \mathbf{x}_1) & \kappa(\mathbf{x}_2, \mathbf{x}_2) & \cdots & \kappa(\mathbf{x}_2, \mathbf{x}_N) \\ \vdots & \vdots & \ddots & \vdots \\ \kappa(\mathbf{x}_N, \mathbf{x}_1) & \kappa(\mathbf{x}_N, \mathbf{x}_2) & \cdots & \kappa(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix}$$

Here $\kappa(\mathbf{x}, \mathbf{x}')$ is some measure of similarity between $\mathbf{x}$ and $\mathbf{x}'$

The dual program becomes

maximise $\sum_{i=1}^{N} \alpha_i - \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j K_{i,j}$

subject to : $0 \leq \alpha_i \leq C$ and $\sum_{i=1}^{N} \alpha_i y_i = 0$

To make prediction on new $\mathbf{x}_{\text{new}}$, only need to compute $\kappa(\mathbf{x}_i, \mathbf{x}_{\text{new}})$ for support vectors $\mathbf{x}_i$ (for which $\alpha_i > 0$)

Polynomial Kernels

Rather than perform basis expansion,

$$\kappa(\mathbf{x}, \mathbf{x}') = (1 + \mathbf{x} \cdot \mathbf{x}')^d$$

This gives all terms of degree up to d

If we use $\kappa(\mathbf{x}, \mathbf{x}') = (\mathbf{x} \cdot \mathbf{x}')^d$, we get only degree d terms

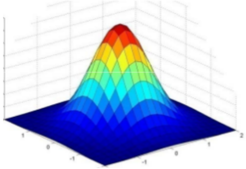
Linear Kernel: $\kappa(\mathbf{x}, \mathbf{x}') = \mathbf{x} \cdot \mathbf{x}'$

All of these satisfy the Mercer or positive-definite condition

Gaussian or RBF Kernel

Radial Basis Function (RBF) or Gaussian Kernel

$$\kappa(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\sigma^2}\right)$$



σ^2 is known as the bandwidth

We used this with $\gamma = \frac{1}{2\sigma^2}$ when we studied kernel basis expansion for regression

Can generalise to more general covariance matrices

Results in a Mercer kernel

Kernels on Discrete Data : Cosine Kernel

For text documents: let $\mathbf{x}$ denote bag of words

Cosine Similarity

$$\kappa(\mathbf{x}, \mathbf{x}') = \frac{\mathbf{x} \cdot \mathbf{x}'}{\|\mathbf{x}\|_2 \|\mathbf{x}'\|_2}$$

Term frequency $\text{tf}(c) = \log(1 + c)$, c word count for some word w

Inverse document frequency $\text{idf}(w) = \log\left(\frac{N}{1+N_w}\right)$, N_w #docs containing w

$\text{tf-idf}(\mathbf{x})_w = \text{tf}(x_w)\text{idf}(w)$

Kernels on Discrete Data : String Kernel

Let $\mathbf{x}$ and $\mathbf{x}'$ be strings over some alphabet $\mathcal{A}$

$$\mathcal{A} = \{A, R, N, D, C, E, Q, G, H, I, L, K, M, F, P, S, T, W, Y, V\}$$

IPTSALVKETLALLSTHRTLLIANETLRIPVPVHKNHQLCTEEIFQGIGTLESQTVQGGTV
ERLFKNLSLIKKYIDGQKKKCGEERRRVNQFLDYLQEFLGVMNTEWI

PHRRDLCSRSIWLARKIRSDLTALTESYVKHQGLWSELTEAERLQENLQAYRTFHVLLA
RLLEDQQVHFTPTEGDFHQAIHTLLLQVAAFAYQIEELMILLEYKIPRNEADGMLFEKK
LWGLKVLQELSQWTVRSIHDLRFISSHQTGIP

$$\kappa(\mathbf{x}, \mathbf{x}') = \sum_s w_s \phi_s(\mathbf{x}) \phi_s(\mathbf{x}')$$

$\phi_s(\mathbf{x})$ is the number of times s appears in $\mathbf{x}$ as substring

w_s is the weight associated with substring s

How to choose a good kernel?

Not always easy to tell whether a kernel function is a Mercer kernel

Mercer Condition: For any finite set of points, the Kernel matrix should be positive semi-definite

If the following hold:

- κ_1, κ_2 are Mercer kernels for points in $\mathbb{R}^D$
- $f : \mathbb{R}^D \to \mathbb{R}$
- $\phi : \mathbb{R}^D \to \mathbb{R}^M$
- κ_3 is a Mercer kernel on $\mathbb{R}^M$

the following are Mercer kernels

- $\kappa_1 + \kappa_2$, $\kappa_1 \cdot \kappa_2$, $\alpha\kappa_1$ for $\alpha \geq 0$
- $\kappa(\mathbf{x}, \mathbf{x}') = f(\mathbf{x})f(\mathbf{x}')$
- $\kappa_3(\phi(\mathbf{x}), \phi(\mathbf{x}'))$
- $\kappa(\mathbf{x}, \mathbf{x}') = \mathbf{x}^\mathsf{T}\mathbf{A}\mathbf{x}'$ for $\mathbf{A}$ positive definite

Kernel Trick in Linear Regression

Recall the least squares objective for linear regression

$$\mathcal{L}(\mathbf{w}) = \sum_{i=1}^{N} (\mathbf{w}^\mathsf{T} \mathbf{x}_i - y_i)^2$$

and the solution $\widehat{\mathbf{w}}_{\mathsf{LS}} = (\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}(\mathbf{X}^\mathsf{T}\mathbf{y})$.

We can express $\widehat{\mathbf{w}} = \sum_{i=1}^{m} \alpha_i \mathbf{x}_i$. Why?

You will give the answer in Problem Sheet 3

Concluding Remarks

- Revise and self-study multiclass classification and performance measures in lecture notes
- Next Time: Neural Networks
- Revise chain rule
- Online book by Michael Nielsen http://www.michaelnielsen.org