Machine learning - HT 2016
7. Classification: Support Vector Machines

Varun Kanade

University of Oxford
February 12, 2016
Today we’ll discuss classification using support vector machines.

- No clear probabilistic interpretation
- Maximum Margin Formulation
- Optimisation problem using Hinge Loss
- Dual Formulation
Binary Classification

Goal: Find a linear separator

Data is linearly separable if there exists a linear separator with no classification error
Maximum Margin Principle

Maximise the distance of the closest point from the decision boundary
Geometry

Given a hyperplane: \( H \equiv \mathbf{w} \cdot \mathbf{x} + w_0 = 0 \) and a point \( \mathbf{x} \in \mathbb{R}^n \)

How far is \( \mathbf{x} \) from \( H \)?
Geometry

For a hyperplane: $H = \mathbf{w} \cdot \mathbf{x} + w_0 = 0$ with $\|\mathbf{w}\|_2 = 1$

The distance of point $\mathbf{x}$ from $H$ is $|\mathbf{w} \cdot \mathbf{x} + w_0|$

If we don’t restrict $\|\mathbf{w}\|_2$ to be 1, then the distance is:

$$\frac{|\mathbf{w} \cdot \mathbf{x} + w_0|}{\|\mathbf{w}\|_2}$$
SVM Formulation: Separable Case

With margin:

Fix \( \|w\|_2 = 1 \)
and then

\[
\max \alpha
\]

s.t. \( y_i (w \cdot x_i + w_0) \geq \alpha \) \( \forall i \)

margin

Alternatively, fix margin = 1
and minimize \( \|w\|_2^2 \).

\[
\min \frac{1}{2} \|w\|_2^2
\]

s.t. \( y_i (w \cdot x_i + w_0) \geq 1 \) \( \forall i \)

QUADRATIC PROGRAM

Without margin condition

\[
y_i (w \cdot x_i + w_0) \geq 1
\]

(linear program)

No objective, just need to find a feasible point
SVM Formulation: Separable Case

minimise: $\frac{1}{2} \| w \|^2$

subject to:

$$y_i (w \cdot x_i + w_0) \geq 1$$

for $i = 1, \ldots, m$

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

If data is separable, then we find a classifier with no **classification error** on the training set.
Non-separable Data

Quadratic program on previous slide has no feasible solution

Which linear separator should we try to find?

Possible choices

(i) Least # misclassified point (NP-hard)

(ii) Sum of distance from boundary
SVM Formulation: Non-Separable Case

minimise: \( \frac{1}{2} ||w||^2 + C \sum_{i=1}^{m} \xi_i \)

subject to:

\( y_i (w \cdot x_i + w_0) \geq 1 - \xi_i \)

for \( i = 1, \ldots, m \)

Here \( y_i \in \{-1, 1\} \)

\( \xi_i \geq 0 \)
SVM Formulation: Non-Separable Case

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

subject to:

$y_i(w \cdot x_i + w_0) \geq 1 - \zeta_i$

$\zeta_i \geq 0$

for $i = 1, \ldots, m$

Here $y_i \in \{-1, 1\}$
SVM Formulation: Loss Function

Regularizer

Loss

minimise: \( \frac{1}{2} \| w \|^2 + C \sum_{i=1}^{m} \zeta_i \)

subject to:

\( y_i (w \cdot x_i + w_0) \geq 1 - \zeta_i \)

\( \zeta_i \geq 0 \)

for \( i = 1, \ldots, m \)

Here \( y_i \in \{-1, 1\} \)

Either \( \zeta_i = 0 \)

correctly classified

\( w, y_i (w \cdot x_i + w_0) = 1 - \zeta_i \)

\( \zeta_i = \max \{ 0, 1 - y_i (w \cdot x_i + w_0) \} \)
Logistic Regression: Loss Function

Here \( y_i \in \{0, 1\} \)

\[
\ell(w; x_i, y_i) = - \left( y_i \log \left( \frac{1}{1 + e^{-w \cdot x_i}} \right) + (1 - y_i) \log \left( \frac{1}{1 + e^{w \cdot x_i}} \right) \right)
\]

\[
= \log (1 + e^{-\xi_i \cdot (w \cdot x_i)})
\]
Loss Functions
minimise: $\frac{1}{2} \| w \|_2^2 + C \sum_{i=1}^{m} \zeta_i$

subject to:

$y_i (w \cdot x_i + w_0) \geq 1 - \zeta_i$

$\zeta_i \geq 0$

for $i = 1, \ldots, m$

Here $y_i \in \{-1, 1\}$
SVM Formulation: Non-Separable Case

What if your data looks like this?

Use basis expansion.

e.g. Quadratic
SVM Formulation: Constrained Minimisation

minimise: \( \frac{1}{2} \| \mathbf{w} \|^2 + C \sum_{i=1}^{m} \zeta_i \)

subject to:

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

\[ \zeta_i \geq 0 \]

for \( i = 1, \ldots, m \)

Here \( y_i \in \{-1, 1\} \)
SVM Formulation

minimise: \( \frac{1}{2} \| \mathbf{w} \|_2^2 + C \sum_{i=1}^{m} \zeta_i \)

subject to:
\[
\begin{align*}
\alpha_i: & \quad y_i (\mathbf{w} \cdot \mathbf{x}_i + w_0) - (1 - \zeta_i) \geq 0 \\
\mu_i: & \quad \zeta_i \geq 0
\end{align*}
\]
for \( i = 1, \ldots, m \)

Here \( y_i \in \{-1, 1\} \)

Lagrange Function

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

Constrained optima are stationary points of \( L \) for suitably chosen \( \alpha_i, \mu_i \)
(Idealised) Lagrange Function

\[ L_{\text{ideal}}(w, w_0, \zeta) = \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{m} \zeta_i - \sum_{i=1}^{m} \mathbb{I}_{\geq 0}(y_i(w \cdot x_i + w_0) - (1 - \zeta_i)) - \sum_{i=1}^{m} \mathbb{I}_{\geq 0}(\zeta_i), \]

where \( \mathbb{I}_{\geq 0}(x) = 0 \) if \( x \geq 0 \) and \( -\infty \) otherwise.

Lagrange Function

\[ L(w, w_0, \zeta; \alpha, \mu) = \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{m} \zeta_i - \sum_{i=1}^{m} \alpha_i(y_i(w \cdot x_i + w_0) - (1 - \zeta_i)) - \sum_{i=1}^{m} \mu_i \zeta_i \]

For any \( \alpha, \mu \geq 0 \), we have for all \( w, w_0, \zeta \)

\[ L(w, w_0, \zeta; \alpha, \mu) \leq L_{\text{ideal}}(w, w_0, \zeta) \]
Dual Formulation

Lagrange Function

\[ L(w, w_0, \zeta; \alpha, \mu) = \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{m} \zeta_i - \sum_{i=1}^{m} \alpha_i (y_i (w \cdot x_i + w_0) - (1 - \zeta_i)) - \sum_{i=1}^{m} \mu_i \zeta_i \]

\[ g(\alpha, \mu) = \inf_{w, w_0} L(w, w_0, \zeta; \alpha, \mu) \leq L(w^*, w_0^*, \zeta^*; \alpha, \mu) \]

Maximising \( g(\alpha, \mu) \) will give us the “best possible” lower bound

\[ \inf \text{ value of objective} \]

In our case STRONG DUALITY holds
Dual Formulation

Lagrange Function

\[ L(w, w_0, \zeta; \alpha, \mu) = \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{m} \zeta_i - \sum_{i=1}^{m} \alpha_i (y_i (w \cdot x_i + w_0) - (1 - \zeta_i)) - \sum_{i=1}^{m} \mu_i \zeta_i \]

For \( \alpha, \mu > 0 \)

minimize \( L \) wrt \( w, w_0, \zeta \)

\[ \nabla_w L = w - \sum_{i=1}^{m} \alpha_i y_i x_i = 0 \]

\[ \Rightarrow w = \sum_{i=1}^{m} \alpha_i y_i x_i \]

\[ \nabla_{w_0} L = - \sum_{i=1}^{m} \alpha_i y_i = 0 \]

\[ \Rightarrow \sum_{i=1}^{m} \alpha_i y_i = 0 \]

\[ \frac{\partial L}{\partial \zeta_i} = C - \alpha_i - \mu_i = 0 \]

\[ \Rightarrow \alpha_i = C - \mu_i \]

\[ \Rightarrow 0 \leq \alpha_i \leq C \]

[DUAL PROBLEM] \[ \max: - \frac{1}{2} \sum_{i=3}^{m} \alpha_i \sum_{j=3}^{m} y_j y_j x_j^T x_i + \sum_{i=1}^{m} \zeta_i \]

s.t. \[ \sum_{i=3}^{m} y_i \alpha_i = 0 \] and \( 0 \leq \alpha_i \leq C \) \forall i
Dual Formulation

**Primal Form**

minimise: \( \frac{1}{2} \| w \|_2^2 \)

subject to:

\[ y_i(w \cdot x_i + w_0) \geq 1 - \zeta_i \]
\[ \zeta_i \geq 0 \]

for \( i = 1, \ldots, m \)

**Lagrange Function**

\[ L(w, w_0, \zeta; \alpha, \mu) = \frac{1}{2} \| w \|_2^2 + C \sum_{i=1}^{m} \zeta_i - \sum_{i=1}^{m} \alpha_i (y_i(w \cdot x_i + w_0) - (1 - \zeta_i)) - \sum_{i=1}^{m} \mu_i \zeta_i \]

**KKT Conditions**

\[ \alpha_i^* = 0 \quad \text{or} \quad y_i(w^* \cdot x_i + w_0^*) - 1 = 0, \quad \text{where} \quad w^* = \sum_{i=1}^{m} \alpha_i^* y_i x_i \]

**Dual Form**

maximise: \( \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j x_i^T x_j \)

subject to:

\[ C \geq \alpha_i \geq 0 \]
\[ \sum_{i=1}^{m} \alpha_i y_i = 0 \]

\( \beta^* \) dual variables = \( m \)

Objective only depends on \( x_i^T \cdot x_j \)

\( w^* = \sum \alpha_i^* y_i x_i \)

Relating primal & dual solutions

sparse if most \( \alpha_i = 0 \)
Support Vectors

For all other points: \( \alpha_i = 0 \)

Support vectors: \( \alpha_i \neq 0 \)
SVM Dual Program

maximise: \[ \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j x_i^T x_j \]

subject to:
\[ C \geq \alpha_i \geq 0 \]
\[ \sum_{i=1}^{m} \alpha_i y_i = 0 \]

- Objective depends only between dot products of training inputs
- Particularly useful if inputs are high-dimensional
- To make a new prediction only need to know dot product with support vectors
Gram Matrix

If we put the inputs in matrix $X$, where the $i^{th}$ row of $X$ is $x_i^T$.

$$K = XX^T = \begin{bmatrix} \langle x_1, x_1 \rangle & \langle x_1, x_2 \rangle & \cdots & \langle x_1, x_m \rangle \\ \langle x_2, x_1 \rangle & \langle x_2, x_2 \rangle & \cdots & \langle x_2, x_m \rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle x_m, x_1 \rangle & \langle x_m, x_2 \rangle & \cdots & \langle x_m, x_m \rangle \end{bmatrix}$$

- The matrix $K$ is positive definite (if $n > m$ and $x_i$ are linearly independent)
- If we do feature expansion first

$$\phi : \mathbb{R}^n \rightarrow \mathbb{R}^N$$

then replace entries by $\langle \phi(x_i), \phi(x_j) \rangle$
Kernel Trick

Suppose we do basis expansion with up to degree two terms (say \( n = 2 \))

\[
\phi((x_1, x_2)) = (1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2, \sqrt{2}x_1x_2)
\]

If \( x = (x_1, x_2) \) and \( x' = (x'_1, x'_2) \) then what is \( \langle \phi(x), \phi(x') \rangle \)?

\[
\langle \phi(x), \phi(x') \rangle = 1 + 2x_1x'_1 + 2x_2x'_2 + x_1^2(x'_1)^2 + x_2^2(x'_2)^2 + 2x_1x_2x'_1x'_2
\]

\[
= (1 + x_1x'_1 + x_2x'_2)^2
\]

\[
= (1 + \langle x, x' \rangle)^2
\]

To compute matrix entries, only need \( O(n) \) time!
Kernel Trick

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

\[
K = \begin{bmatrix}
\kappa(x_1, x_1) & \kappa(x_1, x_2) & \cdots & \kappa(x_1, x_m) \\
\kappa(x_2, x_1) & \kappa(x_2, x_2) & \cdots & \kappa(x_2, x_m) \\
\vdots & \vdots & \ddots & \vdots \\
\kappa(x_m, x_1) & \kappa(x_m, x_2) & \cdots & \kappa(x_m, x_m)
\end{bmatrix}
\]

Here \(\kappa(x, x')\) is some measure of similarity between \(x\) and \(x'\)

The dual program becomes

maximise \[\sum_{i=1}^{m} \alpha_i - \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j K_{i,j} \quad \text{subject to : } 0 \leq \alpha_i \leq C\]

To make prediction on new \(x_{\text{new}}\), need to compute \(\kappa(x_i, x_{\text{new}})\) for support vectors \(x_i\)
Polynomial Kernels

Rather than perform basis expansion,

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

This gives all terms of degree up to $d$

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

**Linear Kernel:** $\kappa(x, x') = x \cdot x'$

All of these satisfy the Mercer or positive-definite condition
Gaussian or RBF Kernel

Radial Basis Function (RBF) or Gaussian Kernel

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

\( \sigma^2 \) is known as the \textbf{bandwidth}

Can generalise to more general covariance matrices

Results in a Mercer kernel
Kernels on Discrete Data: Cosine Kernel

For text documents: let $x$ denote bag of words

Cosine Similarity

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

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

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

$tf-idf(x)_w = tf(x_w)idf(w)$
Let $x$ and $x'$ be strings over some alphabet $\mathcal{A}$


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

$\phi_s(x)$ is the number of times $s$ appears in $x$ as substring
How to choose a good kernel?

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

For any finite set of inputs, the Kernel matrix should be positive definite.

If the following hold:

1. $\kappa_1$, $\kappa_2$ are Mercer kernels for points in $\mathbb{R}^n$.
2. $f : \mathbb{R}^n \rightarrow \mathbb{R}$
3. $\phi : \mathbb{R}^n \rightarrow \mathbb{R}^N$
4. $\kappa_3$ is a Mercer kernel on $\mathbb{R}^N$

the following are Mercer kernels:

1. $\kappa_1 + \kappa_2$, $\kappa_1 \cdot \kappa_2$, $\alpha \kappa_1$ for $\alpha \in \mathbb{R}^+$
2. $\kappa(x, x') = f(x)f(x')$
3. $\kappa_3(\phi(x), \phi(x'))$
4. $\kappa(x, x') = x^T A x'$ for $A$ positive definite.
Books

 Kernel Methods for Pattern Analysis

 Advances in Kernel Methods Support Vector Learning
Kernel Trick in Linear Regression

Recall the loss function for linear regression (least squares)

\[ L(w) = \sum_{i=1}^{m} (x_i^T w - y_i)^2 \]

and the solution \( \hat{w} = (X^T X)^{-1} (X^T y) \).

Can write \( \hat{w} = \sum_{i=1}^{m} \alpha_i x_i \). Why?

Suppose \( \hat{w} = \tilde{w} + u \). where \( u \perp x_i \) \& \( \tilde{w} \).

Then \( \langle \hat{w}, x_i \rangle = \langle \tilde{w}, x_i \rangle \).

So \( \tilde{w} \) is identical solution.

If we had regularized, then \( u \) is forced to be 0, since \( ||\hat{w}||^2 = ||\tilde{w}||^2 + ||u||^2 \).
Support Vector Regression

$\epsilon$-sensitive loss function

$$\ell_\epsilon(y, \hat{y}) = \begin{cases} 0 & \text{if } |y - \hat{y}| < \epsilon \\ |y - \hat{y}| - \epsilon & \text{otherwise} \end{cases}$$

Loss function:

$$L(w) = \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{m} \ell_\epsilon(x_i^T w, y_i)$$