Support Vector Machines

Instructor Max Welling
ICS273A
UCIrvine
Philosophy

• First formulate a classification problem as finding a separating hyper-plane that maximizes “the margin”.

• Allow for errors in classification using “slack-variables”.

• Convert problem to the “dual problem”.

• This problem only depends on inner products between feature vectors which can be replaced with kernels.

• A kernel is like using an *infinite* number of features.
The Margin

- Large margins are good for generalization performance (on future data).
- Note: this is very similar to logistic regression (but not identical).
• We would like to find an expression for the margin (a distance).

• Points on the decision line satisfy: $w^T x - b = 0$

• First imagine the line goes through the origin: $w^T x = 0$
  Then shift origin: $w^T (x - a) = 0$
  Choose $a \parallel w \Rightarrow b = w^T a = \| a \| \times \| w \| \Rightarrow \| a \| = \frac{b}{\| w \|}$
Primal Problem

- Points on support vector lines (dashed) are given by: \( w^T x = b + \delta \) and \( w^T x = b - \delta \).
- If I change: \( w \rightarrow \lambda w, \ b \rightarrow \lambda b, \ \delta \rightarrow \lambda \delta \) the equations are still valid. Thus we can choose \( \delta = 1 \) without loss of generality.
Primal Problem

- We can express the margin as: $2\left(\frac{b+1}{||w||} - \frac{b}{||w||}\right) = \frac{2}{||w||}$
- $2/||w||$ is always true. Check this also for $b$ in $(-1,0)$.
- Recall: we want to maximize the margin, such that all data-cases end up on the correct side of the support vector lines.

$$\underset{w,b}{\min} \ ||w||^2 \ subject \ to \ \begin{cases} w^T x_n \geq b + 1 \ if \ y_n = +1 \\ w^T x_n \leq b - 1 \ if \ y_n = -1 \end{cases} \ \forall n$$
Primal problem (QP)

\[
\begin{align*}
\min_{w,b} & \quad \frac{1}{2} \|w\|^2 \\
\text{s.t.} & \quad y_n(w^T x_n - b) - 1 \geq 0 \quad \forall n
\end{align*}
\]
Slack Variables

• It is not very realistic to assume that the data are perfectly separable.

• Solution: add slack variables to allow violations of constraints:

\[
\begin{align*}
   w^T x_n & \geq b + 1 - \xi_n \quad \text{if } y_n = +1 \\
   w^T x_n & \leq b - 1 + \xi_n \quad \text{if } y_n = -1
\end{align*}
\quad \forall n
\]

• However, we should try to minimize the number of violations. We do this by adding a term to the objective:

\[
\min_{w,b,\xi} \frac{1}{2} \| w \|^2 + C \sum_{n=1}^{N} \xi_n
\]

s.t. \( y_n (w^T x_n - b) - 1 + \xi_n \geq 0 \quad \forall n \)

s.t. \( \xi_n \geq 0 \quad \forall n \)
Let’s say we wanted to define new features: $\phi(x) = [x, y, x^2, y^2, xy, ...]$

The problem would then transform to:

$$\min_{w,b,\xi} \frac{1}{2} ||w||^2 + C \sum_{n=1}^{N} \xi_n$$

s.t. $y_n (w^T \phi(x_n) - b) - 1 + \xi_n \geq 0 \quad \forall n$

s.t. $\xi_n \geq 0 \quad \forall n$

• Rationale: data that is linearly non-separable in low dimensions may become linearly separable in high dimensions (provided sensible features are chosen).
Let’s say we wanted very many features (F>>N), or perhaps *infinitely many features.*

In this case we have very many parameters $w$ to fit.

By converting to the *dual problem,* we have to deal with exactly N parameters.

This is a change of basis, where we recognize that we only need dimensions inside the space spanned by the data-cases.

The transformation to the dual is rooted in the theory of *constrained convex optimization.* For a convex problem (no local minima) the dual problem is equivalent to the primal problem (i.e. we can switch between them).
Dual Problem (QP)

\[
\begin{align*}
\max_\alpha & \quad \sum_n \alpha_n - \frac{1}{2}\sum_{n,m} \alpha_n \alpha_m y_n y_m \phi_n^T \phi_m \\
\text{s.t.} & \quad \sum_n \alpha_n y_n = 0, \quad \alpha_n \in [0, C] \quad \forall n
\end{align*}
\]

• The \( \alpha_n \) should be interpreted as forces acting on the data-items. Think of a ball running down a hill (optimizing \( w \) over \( ||w||^2 \)). When it hits a wall, the wall start pushing back, i.e. the force is active.

If data-item is on the correct side of the margin: no force active: \( \alpha_n = 0 \)

If data-item is on the support-vector line (i.e. it is a support vector!) The force becomes active: \( \alpha_n \in [0, C] \)

If data-item is on the wrong side of the support vector line, the force is fully engaged: \( \alpha_n = C \)
Complementary Slackness

• The complementary slackness conditions come from the KKT conditions in convex optimization theory.

\[ \alpha_n (\gamma_n (w^T \phi_n - b) - 1 + \xi_n) = 0 \]

• From these conditions you can derive the conditions on alpha (previous slide)

• The fact that many alpha’s are 0 is important for reasons of efficiency.
Kernel Trick

• Note that the dual problem only depends on $\phi_n^T \phi_m$

• We can now move to infinite number of features by replacing:

$$\phi(x_n)^T \phi(x_m) \rightarrow K(x_n, x_m)$$

• As long as the kernel satisfies 2 important conditions you can forget about the features

$$v^T K v \geq 0 \quad \forall v \quad (\text{positive semi definite, positive eigenvalues})$$

$$K = K^T \quad (\text{symmetric})$$

• Examples: $K_{pol}(x, y) = (r + x^T y)^d$

$$K_{rbf}(x, y) = c \exp(-\beta \|x - y\|^2)$$
Prediction

• If we work in high dimensional feature spaces or with kernels, b has almost no impact on the final solution. In the following we set b=0 for convenience.

• One can derive a relation between the primal and dual variables (like the primal dual transformation, it requires Lagrange multipliers which we will avoid here. But see notes for background reading).

• Using this we can derive the prediction equation:

\[ y_{test} = \text{sign}\left[ w^T x_{test} \right] = \text{sign}\left[ \sum_{n \in SV} \alpha_n y_n K(x_{test}, x_n) \right] \]

• Note that it depends on the features only through their inner product (or kernel).

• Note: prediction only involves support vectors (i.e. those vectors close to or on wrong side of the boundary). This is also efficient.
Conclusions

- kernel-SVMs are non-parametric classifiers: It keeps all the data around in the kernel-matrix.

- Still we often have parameters to tune (C, kernel parameters). This is done using X-validation or by minimizing a bound on the generalization error.

- SVMs are state-of-the-art (given a good kernel).

- SVMs are also slow (at least $O(N^2)$). However approximations are available to elevate that problem (i.e. $O(N)$).