CS 536 : Support Vector Machines

16:198:536

As noted previously, in the framework of binary classification, looking for linear separators on a data set admits two possible approaches so far:

- The Perceptron Learning Algorithm: simple learning algorithm, guaranteed convergence when a separator exists, but possibly in non-polynomial time. The solution $\underline{w}^*$ is structured as a simple linear combination of training data points. General sample complexity of
$$m \geq \frac{1}{\epsilon^2}\frac{1}{{\gamma^*}^2}\log\left(\frac{1}{\epsilon^2}\frac{1}{{\gamma^*}^2}\right)^2, \tag{1}$$
independent of the dimension of the feature space.
- Linear Programming: not so simple learning algorithms, but well studied and established, guaranteed convergence when separator exists, in polynomial time (for some LP algorithms). The structure of the solution $\underline{w}^*$ is not so clear. General sample complexity of
$$m \geq \frac{1}{\epsilon^2}k, \tag{2}$$
where k is the ambient dimension of the feature space.

Which is the better approach? Can we achieve the best of both worlds? Tangentially related - under either approach, there may be multiple possible linear separators admitted by the data; how can we determine what is the 'best' linear separator? This connects back to previous discussions of regularization and generalizability - what is the linear separator that will best generalize to new, unseen data? A reasonable argument can be made that the best separator will be the one that **is as far away from both classes of data points as possible**, i.e., has the maximum margin. Utilizing this separator minimizes the chance that new data points, while similar to old data points, will fall on the wrong side of the separator and be misclassified. Recalling the definition of the margin for a separator $\underline{w}$:

$$\gamma(\underline{w}) = \min_i \frac{|\underline{w}.\underline{x}^i|}{||\underline{w}||}, \tag{3}$$

we can specify the problem of finding the separator of *maximum margin* as

$$\begin{aligned} &\max_{\underline{w}} \ \min_i \frac{|\underline{w}.\underline{x}^i|}{||\underline{w}||} \\ &\text{(s.t.)} \ \ \forall i: \ y^i(\underline{w}.\underline{x}^i) > 0. \end{aligned} \tag{4}$$

It is convenient to equivalently express this problem in the following way:

$$\begin{aligned} &\max_{\underline{w},\gamma} \gamma \\ &\text{(s.t.)} \ \ \forall i: \ y^i(\underline{w}.\underline{x}^i) > 0 \\ &\qquad \forall i: \ \frac{|\underline{w}.\underline{x}^i|}{||\underline{w}||} \geq \gamma \\ &\qquad \gamma \geq 0. \end{aligned} \tag{5}$$

Observing that $y^i(\underline{w}.\underline{x}^i) = |\underline{w}.\underline{x}^i| > 0$ if and only if $(\underline{x}^i, y^i)$ is correctly classified, the above simplifies to

$$\begin{aligned} &\max_{\underline{w},\gamma} \gamma \\ &\text{(s.t.)} \ \ \forall i: \ y^i(\underline{w}.\underline{x}^i) \geq \gamma||\underline{w}|| \\ &\qquad \gamma \geq 0. \end{aligned} \tag{6}$$

As a final simplification, the constraint could be expressed as $y^i(\underline{w}'.\underline{x}^i) \geq 1$ where $\underline{w}' = \underline{w}/(\gamma||\underline{w}||)$, and $||\underline{w}'|| = 1/\gamma$. The final problem could then be re-expressed (dropping the $'$) as

$$\begin{aligned} &\max_{\underline{w}} \frac{1}{||\underline{w}||} \\ &(\text{s.t.}) \quad \forall i: \ y^i(\underline{w}.\underline{x}^i) \geq 1. \end{aligned} \tag{7}$$

This is frequently expressed in the following way, the classical formulation of the support vector machine:

The SVM Primal Problem: Find the weight vector $\underline{w}$ that solves the following optimization problem

$$\begin{aligned} &\min_{\underline{w}} \frac{1}{2}||\underline{w}||^2 \\ &(\text{s.t.}) \quad \forall i: \ y^i(\underline{w}.\underline{x}^i) \geq 1. \end{aligned} \tag{8}$$

Note that in this case the objective function is *quadratic* in the unknown variables, rather than linear - this is a quadratic programming problem rather than a linear programming problem, but in fact efficient solution algorithms still exist to generate a solution (should one exist) in polynomial time. These are found in a number of common numerical analysis packages. Additionally, it can be shown that the sample complexity of the SVM is

$$m \geq \frac{1}{\epsilon^2} \min \left[\frac{1}{\gamma^{*2}} \log \left(\frac{1}{\epsilon^2} \frac{1}{\gamma^{*2}} \right)^2, k \right], \tag{9}$$

i.e., this captures the 'best of both worlds' as requested.

Several questions remain, in particular:

- What is the structure of the solution, i.e., what does the optimal $\underline{w}^*$ 'look like' and how does it relate to the data?
- How can we apply or generalize this framework to data that is not linearly separable?

The Geometry of the Support Vector Machine

Given a solution $\underline{w}$ to the Primal SVM problem, we can say a couple of things about the structure and geometry of the solution:

- The linear separator dividing the two classes is the hyperplane defined by $\underline{w}.\underline{x} = 0$.
- The margin of this linear separator is given by $\gamma^* = 1/||\underline{w}||$. This can be seen from the derivation of the primal problem from the original problem of 'find a valid separator that maximizes the margin' as given above.
- The data points that lie closest to the linear separator or 'on the margin' satisfy $y^i(\underline{w}.\underline{x}^i) = 1$. To see this, note that any point that lies on the margin satisfies

  $$\gamma^* = \frac{|\underline{w}.\underline{x}^i}{||\underline{w}||} = \frac{y^i(\underline{w}.\underline{x}^i)}{1/\gamma^*}, \tag{10}$$

  which simplifies to the desired relation.

- The boundary of the margin is effectively defined by the hyperplanes $\underline{w}.\underline{x} = 1$ and $\underline{w}.\underline{x} = -1$. No data points lie between these two hyperplanes, since for all data points $y^i(\underline{w}.\underline{x}^i) \geq 1$, and data points *on* the margin lie exactly on these hyperplanes. The linear separator lies exactly half way between these two hyperplanes, parallel to both. Note - this last observation is simply that if the linear separator did not lie exactly between these two hyperplanes, the margin could be increased by moving the separator to the middle of these two boundaries.

Given the linear separator defined by $\underline{w}$, the **support vectors** of this solution are the data points $(\underline{x}^i, y^i)$ that lie exactly on $\underline{w}.\underline{x} = \pm 1$ - all other data points lie off this boundary, with $y^i(\underline{w}.\underline{x}^i) > 1$. It is interesting to note that the boundary is effectively defined exactly by these support vectors and no other data points - and the linear separator itself is defined in terms of the boundaries (half way between each) and therefore is defined in terms of the support vectors. These support vectors, as they are the data points 'closest' to the linear separator of maximum margin, effectively represent the data points that are *hardest to classify*. One way of interpreting the SVM and its solution then is that it identifies the hardest examples of data to classify, and defines the classifier purely in terms of these.

Another interesting observation here is that geometrically, the solution $\underline{w}$ to the SVM is stable to perturbations of non-support vector data points. Any point $(\underline{x}^i, y^i)$ that lies away from the boundary could be moved slightly without risk of misclassification by the linear separator. This suggests that we should be able to express or at least interpret the solution $\underline{w}$ *purely in terms of the support vectors*.

And this is in fact the case. Recall that for the output of the Perceptron Learning Algorithm, the solution $\underline{w}$ effectively had the form $\underline{w} = \sum_i \alpha_i y^i \underline{x}^i$ where α^i was a non-negative integer. It can be shown that if $\underline{w}$ is the solution to the Primal SVM problem, then $\underline{w}$ also has the form

$$\underline{w} = \sum_i \alpha_i y^i \underline{x}^i, \tag{11}$$

where each α_i is a non-negative *real* value, and $\alpha_i = 0$ if $(\underline{x}^i, y^i)$ is **not** a support vector, and $\alpha_i > 0$ if $(\underline{x}^i, y^i)$ is a support vector. Note, this gives an alternative expression for the final classifier,

$$f(\underline{x}) = \text{sign}\left(\sum_{\text{support } i} \alpha_i y^i \underline{x}^i . \underline{x} \right). \tag{12}$$

This captures the idea that the only data points that are really relevant are the support vectors themselves, and that a new data point $\underline{x}$ will be classified in terms of these support vectors.

One additional interesting observation about this geometric interpretation of the solution in terms of support vectors: while the initial data set may be defined in a very high dimensional space, and be quite large in itself, the set of support vectors will typically be much smaller, and of much lower dimension, effectively identifying the 'important' features of the problem, defined in terms of the data itself.

Theoretically and practically, it is frequently useful to consider the SVM not in terms of the Primal Problem, but rather in terms of the Dual - this is a frequent trick in constrained optimization problems, effectively amounting to a change of variables and restructuring of the constraints. The Dual SVM is given below:

The Dual SVM problem: Find the set of values $\{\alpha_i\}$ that solves the following:

$$\begin{aligned} \max_{\underline{\alpha}} \quad & \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i y^i \left(\underline{x}^i . \underline{x}^j \right) y^j \alpha_j \\ \text{(s.t.)} \quad & \sum_{i=1}^{m} \alpha_i y^i = 0 \\ & \forall i : \ \alpha_i \geq 0. \end{aligned} \tag{13}$$

The solution to the Dual problem effectively identifies the support vectors explicitly - $\alpha_i > 0$ for any $(\underline{x}^i, y^i)$ that is a support vector, otherwise $\alpha_i = 0$. These α_i also recover the original solution $\underline{w}$ in the way indicated above - these α_i are precisely the previously discussed coefficients on $y^i \underline{x}^i$. The fact that the Primal and Dual problem yield the same solution, albeit from different perspectives, is a result from deeper optimization theory - see the appendix for the relationship in this specific case.

One final note worth making about the Dual is the following - computationally, we have effectively exchanged m-many linear inequality constraints for a *single* linear equality constraint. This structural simplification frequently makes the Dual a more appealing problem to solve. Additionally, the Dual is effectively independent of the ambient dimension - needing to solve for m-many values α_i rather than $k+1$-many components of $\underline{w}$. This will be especially valuable when we exchange the 'raw' data or feature space for a much higher dimensional (potentially infinite dimensional) feature space while maintaining the same number of training data points.

Generalizing to Non-Linearly Separable Data

Support Vector Machines provide a nice unifying framework for developing linear separators, with a variety of good algorithmic solutions (linear, quadratic programming) and a lot of good geometric structure and intuition (support vectors). But they still rely on there being a linear separator between the training data classes. In reality, data is frequently noisy, training data may be misclassified, or there may simply not be a linear separator at all. The classic example of data lacking a linear separator is the **xor** function, which can be given in the following way:

$$(\underline{x}, y) = \begin{cases} ((0,0), -1) \\ ((1,0), 1) \\ ((0,1), 1) \\ ((1,1), -1) \end{cases} . \tag{14}$$

This can easily be seen as non-separable, graphing and labeling the data, but imagine looking for weights to satisfy the following:

$$\begin{aligned} w_0 + w_1 * 0 + w_2 * 0 < 0 \\ w_0 + w_1 * 1 + w_2 * 0 > 0 \\ w_0 + w_1 * 0 + w_2 * 1 > 0 \\ w_0 + w_1 * 1 + w_2 * 1 < 0 \end{aligned} \tag{15}$$

or

$$\begin{aligned} w_0 &< 0 \\ w_0 + w_1 &> 0 \\ w_0 + w_2 &> 0 \\ w_0 + w_1 + w_2 &< 0 \end{aligned} \tag{16}$$

These inequalities are actually inconsistent: the second and third combine to give $2w_0 + w_1 + w_2 > 0$, but since w_0 is negative we should have $2w_0 + w_1 + w_2 < w_0 + w_1 + w_2 < 0$, according to the last inequality. This is a contradiction, so there can be no satisfying assignment. The mechanics established so far are not applicable here.

Can this framework be adapted or generalized to this case? One way of considering this problem is to look at the notion of training error. Effectively, we have been forcing our classifying solution f to satisfy $\text{err}_{\text{train}}(f) = 0$ - all training data must be correctly classified. This is the so called hard margin SVM - the separator represents a hard cutoff between the classes. A 'soft' margin would potentially allow misclassified data on either side of the margin, assessing some penalty for misclassification of training data but attempting to minimize this penalty. Soft Margin SVMs are commonly specified in one of two ways, in terms of

The C-SVM: Given a value C, find $\underline{w}, \underline{\xi}$ to solve

$$\begin{aligned} &\min_{\underline{w},\underline{\xi}} \frac{1}{2}||\underline{w}||^2 + C\left(\frac{1}{m}\sum_{i=1}^{m}\xi_i\right) \\ \text{(s.t.)}\ &\forall i:\ y^i(\underline{w}.\underline{x}^i) \geq 1 - \xi_i, \\ &\forall i:\ \xi_i \geq 0. \end{aligned} \tag{17}$$

While the previous SVM specification enforced a 'hard' margin boundary of $\underline{w}.\underline{x}^i = \pm 1$, this specification allows some flexibility, allowing data points to cross the boundary up to ξ_i - but attempting to minimize the average crossing penalty over all data points. Note that given a solution $(\underline{w}, \underline{\xi})$, if $(\underline{x}^i, y^i)$ is misclassified, then we have that $0 > y^i(\underline{w}.\underline{x}^i) \geq 1 - \xi_i$ or $\xi_i \geq 1$. We can then estimate the total training error with $(1/m)\sum_i \xi_i$ - though this estimate can be quite poor depending on how 'non-separable' the data is. The objective function then attempts to minimize both the margin, and the training error - with the parameter C determining the tradeoff between the two.

or

The ν-SVM: Given a value ν, fine $\underline{w}, \underline{\xi}, \rho$ to solve

$$\begin{aligned} &\min_{\underline{w},\underline{\xi},\rho} \frac{1}{2}||\underline{w}||^2 - \rho\nu + \frac{1}{m}\sum_{i=1}^{m}\xi_i \\ \text{(s.t.)}\ &\forall i:\ y^i(\underline{w}.\underline{x}^i) \geq \rho - \xi_i, \\ &\forall i:\ \xi_i \geq 0 \\ &\rho \geq 0 \end{aligned} \tag{18}$$

This is very similar to the C-SVM, but relaxes the problem further - recall that the lower bound of 1, and then $1 - \xi_i$, was due to the reparameterization of the original maximal margin problem previously. This simply allows the scaling factor to be a parameter determined by the problem itself. One advantage this problem has over the C-SVM is the following result:

Theorem: If $(\underline{w}^*, \rho^*)$ with $\rho^* > 0$ represents an optimal solution to the ν-SVM problem, then $\text{err}_{\text{train}}(f) \leq \nu$. In this case, the parameter ν provides much more precise control over the training error than the rough estimated importance given by C in the C-SVM.

In general, C-SVMs are easier to train, but the parameter C is somewhat non-intuitive to control given that the ξ_i provide only a ballpark estimate for the training error. The ν-SVM provides much more control and greater precision - the corresponding cost is that these are typically harder to train.

An Example

Consider the following data:

$$\begin{aligned} &((-1,+1),+1) \\ &((+1,+1),+1) \\ &((-1,-1),-1) \\ &((+1,-1),-1) \\ &((0,+1/2),-1) \\ &((0,-1/2),+1). \end{aligned} \tag{19}$$

The first four points are linearly separable using the line $x_2 = 0$. However, the last two points wreck this separability - there is no linear separator that splits the whole data set. We can see this in that a linear separator would need to satisfy the following inequalities:

$$\begin{aligned} w_0 - w_1 + w_2 &> 0 \\ w_0 + w_1 + w_2 &> 0 \\ -w_0 + w_1 + w_2 &> 0 \\ -w_0 - w_1 + w_2 &> 0 \\ -w_0 - (1/2)w_2 &> 0 \\ w_0 - (1/2)w_2 &> 0. \end{aligned} \tag{20}$$

Adding the last two inequalities directly gives that $-w_2 > 0$, but adding all six inequalities gives $3w_2 > 0$. This requires that w_2 be both positive and negative, which is impossible - no linear separator exists.

However, we can recover the intuitive linear separator using the so called soft margin SVM, which assesses a penalty for misclassifying the last two points but allows it. In particular, we want to solve

$$\begin{aligned} \min_{\underline{w},\underline{\xi}} \ & \frac{1}{2}||\underline{w}||^2 + \frac{C}{6}\left(\xi_1 + \ldots + \xi_6\right) \\ \text{(s.t.)} \ & \xi_1 \geq 0, \ldots, \xi_6 \geq 0 \\ & w_0 - w_1 + w_2 \geq 1 - \xi_1 \\ & w_0 + w_1 + w_2 \geq 1 - \xi_2 \\ & -w_0 + w_1 + w_2 \geq 1 - \xi_3 \\ & -w_0 - w_1 + w_2 \geq 1 - \xi_4 \\ & -w_0 - (1/2)w_2 \geq 1 - \xi_5 \\ & w_0 - (1/2)w_2 \geq 1 - \xi_6. \end{aligned} \tag{21}$$

Re-arranging the bounds for the ξ_i, we can get the following results, that

$$\begin{aligned} \xi_1 &= \max(0, 1 - w_0 + w_1 - w_2) \\ \xi_2 &= \max(0, 1 - w_0 - w_1 - w_2) \\ \xi_3 &= \max(0, 1 + w_0 - w_1 - w_2) \\ \xi_4 &= \max(0, 1 + w_0 + w_1 - w_2) \\ \xi_5 &= \max(0, 1 - w_0 - (1/2)w_2) \\ \xi_6 &= \max(0, 1 - w_0 + (1/2)w_2). \end{aligned} \tag{22}$$

This reduces the original problem back to optimizing only over 3 variables, in an unconstrained fashion. Observing the symmetry around $w_0 = 0$ and $w_1 = 0$, some quick plots validate this as the minimum for any given value of w_2. It therefore remains to solve:

$$\begin{aligned}\min_{w_2} \frac{1}{2}w_2^2 + &\frac{C}{6}4\max(0, 1 - w_2)\\ &+ \frac{C}{6}\max(0, 1 - (1/2)w_2) + \frac{C}{6}\max(0, 1 + (1/2)w_2).\end{aligned} \tag{23}$$

It can be shown that for $0 \leq C \leq 3/2$, the above solves for $w_2 = (2/3)C$, and for $C > 3/2$, the above solves for $w_2 = 1$. Either way, this recovers the intuitive separator, which simplifies to $f(x_1, x_2) = \text{sign}(x_2)$.

A The Primal SVM vs the Dual SVM

We want to justify the following result, that

The SVM Primal Problem: Find the weight vector $\underline{w}$ that solves the following optimization problem

$$\begin{aligned} &\min_{\underline{w}} \frac{1}{2}||\underline{w}||^2 \\ &\text{(s.t.)}\ \ \forall i:\ y^i(\underline{w}.\underline{x}^i) \geq 1. \end{aligned} \tag{24}$$

and

The Dual SVM problem: Find the set of values $\{\alpha_i\}$ that solves the following:

$$\begin{aligned} &\max_{\underline{\alpha}} \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i y^i \left(\underline{x}^i.\underline{x}^j\right) y^j \alpha_j \\ &\text{(s.t.)}\ \ \sum_{i=1}^{m} \alpha_i y^i = 0 \\ &\quad \forall i:\ \alpha_i \geq 0. \end{aligned} \tag{25}$$

have the same solution (optimum values of the objective function) and that the link between the two problems is

$$\underline{w} = \sum_i \alpha_i y^i \underline{x}^i. \tag{26}$$

Formally, **Lagrangian Duality** says that the solution to a convex optimization problem with inequality constraints such as

$$\begin{aligned} &\min_{\underline{w}} f(\underline{w}) \\ &\text{(s.t.)}\ \ \forall i:\ 0 \geq g_i(\underline{w}) \end{aligned} \tag{27}$$

is equivalent to the solution to

$$\begin{aligned} &\max_{\underline{\lambda}} \min_{\underline{w}} \left(f(\underline{w}) + \sum_{i=1}^{m} \lambda_i g_i(\underline{w}) \right) \\ &\text{(s.t.)}\ \ \forall i: \lambda_i \geq 0. \end{aligned} \tag{28}$$

As everything is nicely differentiable here, taking $f(\underline{w}) = (1/2)||\underline{w}||^2$ and $g_i(\underline{w}) = 1 - y^i(\underline{w}.\underline{x}^i)$, we have that the interior minimum occurs when the gradient is zero, i.e.,

$$\nabla f(\underline{w}) + \sum_{i=1}^{m} \lambda_i \nabla g_i(\underline{w}) = 0, \tag{29}$$

or

$$\underline{w} + \sum_{i=1}^{m} \lambda_i(-y^i \underline{x}^i) = 0, \tag{30}$$

or

$$\underline{w} = \sum_{i=1}^{m} \lambda_i y^i \underline{x}^i. \tag{31}$$

Taking this $\underline{w}$, the objective function simplifies to

$$\begin{aligned} f(\underline{w}) + \sum_{i=1}^{m} \lambda_i g_i(\underline{w}) &= \frac{1}{2}\left(\sum_{i=1}^{m} \lambda_i y^i \underline{x}^i\right) . \left(\sum_{i=1}^{m} \lambda_i y^i \underline{x}^i\right) + \sum_{i=1}^{m} \lambda_i \left(1 - y^i \left(\sum_{j=1}^{m} \lambda_i y^j \underline{x}^j\right) .\underline{x}^i\right) \\ &= \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \lambda_i y^i \underline{x}^i . \underline{x}^j y^j \lambda_j + \sum_{i=1}^{m} \lambda_i - \sum_{i=1}^{m} \lambda_i \left(y^i \left(\sum_{j=1}^{m} \lambda_i y^j \underline{x}^j\right) .\underline{x}^i\right) \\ &= -\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \lambda_i y^i \underline{x}^i . \underline{x}^j y^j \lambda_j + \sum_{i=1}^{m} \lambda_i . \end{aligned} \tag{32}$$

Taking $\alpha_i = \lambda_i$ recovers the Dual SVM problem.