Lecture 11: July 26 (Part II)

Lecturer: Yishay Mansour *Scribe: Vladimir Koushnir and Ben Sandbank*

11.1 Lecture Overview

We develop the concept of support vector machines, and give some bounds to the errors they achieve. Finally we present a document classification problem and show how SVM's can be used to solve it.

11.2 Support Vector Machines

11.2.1 Some Preliminaries

In the first part of this lecture we presented a dual representation the perceptron algorithm. This representation allows to use a kernel to embed the (non-separable)data into a higher-dimension space in which it is linearly separable. In this kernel-induced feature-space, all we need to do is to select a separating hyperplane, and the classification is given by the sign of the function -

$$f(x) = \sum a_i t_i K(x_i, x) + b, \tag{11.1}$$

where the x_i's are the vectors in the training set. Thus, if y is the correct classification of x, then $yf(x) > 0$ if and only if f classifies x correctly (The problem that remains is - since there may be many hyperplanes which classify the train set correctly, which one should we use)?

11.2.2 Choice of Hyperplane

Naturally, we would like to pick a hyperplane which minimizes the generalization error of the SVM. According to PAC theory, the error rate of a classifier is $O(Vcdim/m)$, where m is the size of the train set. However, in the case of hyperplanes, $Vcdim >> m$, so this bound is not very useful for us. Moreover, PAC theory provides us with no way of choosing between consistent hyperplanes. In the next section we will bring an alternative criterion for selecting an optimal hyperplane.

11.2.3 Margin Based Bounds

Definition For every function f(x), define its *margin* as - $\Theta = \min_x(yf(x))$, where y is the classification of x.

For a given x, it seems intuitively clear that the larger $yf(x)$ is, so x is classified *with more certainty*. Therefore, achieving the largest margin on the training set will ensure that our generalization error is minimized. This intuition is proven in the following theorem, but first we will need some definitions.

Definition We define our basic hypothesis class H as the the following set -

$$H \doteq \{K(x_i, x) | x_i \in TrainSet\} \cup \{u(x) | \forall x u(x) = 1\}. \tag{11.2}$$

Obviously, our SVM classifier can be presented as a linear combination of the elements of H. We now define the *convex hull* of a set of functions -
Definition Let F be a set of functions. Then the *convex hull* of F is defined as:

$$conv(F) \doteq \{g : x \to \sum_{f \in F} a_f f(x) | a_f \geq 0; \sum_f a_f = 1\} \tag{11.3}$$

Note that we can normalize any SVM classifier by dividing it by $\sum a_i + b$ to get a second classifier which classifies exactly like as the first one that belongs to conv(H).
We prove our claim that the generalization error is minimized when the margin is maximized on conv(H), rather than on the space of all SVM classifiers. Later on we will show how the result is extended to the full space.

Theorem 11.1 *Let H be our basic hypothesis class. Let D be a distribution over $X \times \{-1, 1\}$, and let S be a sample of m examples chosen independently at random according to D. Let $\delta > 0$. Then with probability at least $1 - \delta$ over the random choice of the training set S, every $f \in conv(H)$ satisfies the following bound for all $\theta > 0$:*

$$\boldsymbol{P}_D[yf(x) \leq 0] \leq \boldsymbol{P}_S[yf(x) \leq 0] + O(\frac{1}{\sqrt{m}}(\frac{\lg m \lg |H|}{\theta^2} + log(1/\delta))^{1/2}). \tag{11.4}$$

(We denote by $\boldsymbol{P}_D[A]$ the probability of event A when x is chosen according to distribution D. $\boldsymbol{P}_S[A]$ is the probability of event A when x is chosen at random from S).

Proof: First define C_N to be the set of *unweighted* averages over N elements from H:

$$C_N \doteq \{f : x \to \frac{1}{N} \sum_{i=1}^{N} h_i(x) | h_i \in H\}. \tag{11.5}$$

Note that we allow the same $h \in H$ to appear multiple times in the sum.

Every SVM classifier $f \in conv(H)$ can be associated with a distribution over H as defined by the coefficients a_h (and b, which we will disregard for the remainder of this proof for notational clarity - its inclusion changes nothing). This is so because the sum of the $a_h's$ is 1, and can be viewed as the probability for picking the function $h \in H$. By choosing N elements of H independently at random according to this distribution we can generate an element of C_N. Thus we can define for each $h \in H$ a distribution Q over C_N - a function $g \in C_N$ is selected (when distributed according to Q) by choosing $h_1, ..., h_N$ independently at random according to the coefficients a_h:

$$g(x) = (1/N) \sum_{i=1}^{N} h_i(x). \tag{11.6}$$

Our goal is to find an upper bound on the generalization error of $f \in C$. For any $g \in C_N$ and $\theta > 0$ we can separate this probability into two terms:

$$\mathbf{P}_D[yf(x) \leq 0] \leq \mathbf{P}_D[yg(x) \leq \theta/2] + \mathbf{P}_D[yg(x) > \theta/2, yf(x) \leq 0]. \tag{11.7}$$

This holds because for every two events, A and B,

$$\mathbf{P}[A] = \mathbf{P}[B \cap A] + \mathbf{P}[\overline{B} \cap A] \leq \mathbf{P}[B] + \mathbf{P}[\overline{B} \cap A]. \tag{11.8}$$

Since equation (11.7) holds for any $g \in C_N$, we can take the expected value of the right hand side with respect to the distribution Q and get:

$$\begin{aligned} \mathbf{P}_D[yf(x) \leq 0] &\leq \mathbf{P}_{D,g\sim Q}[ygq \leq \theta/2] + \mathbf{P}_{D,g\sim Q}[yg(x) > \theta/2, yf(x) \leq 0] \\ &= E_{q\sim Q}[\mathbf{P}_D[yg(x) \leq \theta/2]] + E_D[\mathbf{P}_{g\sim Q}[yg(x) > \theta/2, yf(x) \leq 0]] \\ &\leq E_D[\mathbf{P}_{g\sim Q}[yg(x) > \theta/2 | yf(x) \leq 0]]. \end{aligned} \tag{11.9}$$

We will bound both terms in equation (11.9) separately, starting with the second term. Consider a fixed example (x, y) and take the probability inside the expectation with respect to the random choice of g. From the way we constructed g it is clear that

$$f(x) = E_{g\sim Q}[g(x)] \tag{11.10}$$

so the probability inside the expectation is equal to the probability that the average over N random draws from a distribution over $-1, 1$ is larger than its expected value by more than $\theta/2$. From the Chernoff bound we deduce that:

$$\mathbf{P}_{g\sim Q}[yg(x) > \theta/2 | yf(x) \leq 0] \leq e^{N\theta^2/8}. \tag{11.11}$$

So we will want to choose $N = O(\frac{1}{\theta^2})$

To bound the first term in 11.7 from above we use the fact that the probability over the choice of S that there exists *any* $g \in C_N$ and $\theta > 0$ such that

$$\mathbf{P}_D[yg(x) \leq \theta/2] > \mathbf{P}_S[yg(x) \leq \theta/2] + \epsilon_N \tag{11.12}$$

is at most $(N+1)|C_N|e^{2m\epsilon^(2)_N}$. The exponential term $e^{2m\epsilon^(2)_N}$ comes from the Chernoff bound which holds for any single choice of g and θ. The term $(N+1)|C_N|$ is an upper bound to the number of such choices where we have used the fact that, because of the form of the functions in C_N, we need to consider only values of θ of the form $2i/N$ for $i = 0, ..., N$. Also note that $|C_N| \leq |H|^N$.

Let us take then $e_N = \sqrt{(1/2m)\ln((N+1)|H|^N/\delta_N)}$, where δ_N will be set later. If we now take expectation with respect to Q, we get for every choice of δ and for every distribution Q, that with probability at least $1 - \delta_N$ -

$$\mathbf{P}_{D,g\sim Q}[yg(x) \leq \theta/2] \leq \mathbf{P}_{S,g\sim Q}[yg(x) \leq \theta/2] + \epsilon_N \tag{11.13}$$

To finish our argument we just need to relate the fraction of the training set on which $yg(x) \leq \theta/2$ to the fraction on which $yf(x) \leq \theta/2$, since this is the quantity we are measuring. Using equation 11.8 we get -

$$\begin{aligned}
\mathbf{P}_{S,g\sim Q}[yg(x) \leq \theta/2] &\leq \mathbf{P}_{S,g\sim Q}[yf(x) \leq \theta] + \mathbf{P}_{S,g\sim Q}[yg(x) \leq \theta/2, yf(x) > \theta] \\
&= \mathbf{P}_S[yf(x) \leq \theta] + \mathbf{E}_S[\mathbf{P}_{g\sim Q}[yg(x) \leq \theta/2, yf(x) > \theta] \\
&\leq \mathbf{P}_S[yf(x) \leq \theta] + \mathbf{E}_S[\mathbf{P}_{g\sim Q}[yg(x) \leq \theta/2 | yf(x) > \theta]]. \quad (11.14) \\
& \quad (11.15)
\end{aligned}$$

To bound the expression inside the expectation we once again use the Chernoff bound to get -

$$\mathbf{P}_{g\sim Q}[yg(x) \leq \theta/2 | yf(x) > \theta]] \leq e^{-N\theta^2/8}. \tag{11.16}$$

Let's set $\delta_N = \delta/(N(N+1))$ so that the probability of failure for *any* N will be at most $\sum_{N\geq 1} \delta_N \leq \delta$. Adding up our results we get that for every $\theta > 0$ and every $N \geq 1$, with probability at least $1-\delta$:

$$\mathbf{P}_D[yf(x)\le 0]\le \mathbf{P}_S[yf(x)\le\theta]+2e^{-N\theta^2/8}+sqrt\frac{1}{2m}\ln(\frac{N(N+1)^2|H|^N}{\delta}). \tag{11.17}$$

The statement of the theorem (finally) follows if we take N to be - $N=[(4/\theta^2)\ln(m/\ln|H|)]$.
$\square$

11.2.4 Extending The Result

The result of the last section holds for the space of *normalized SVM's*. We still need to show how to extend it to the general space of SVM's. As it turns out, there's quite a simple way to do that. Let us define the *Functional Margin* of an SVM in the same way we defined margins to begin with, i.e. $min_i y_i f(x_i)$. Let us further define the *Geometric Margin* of an SVM as $min_i y_i f(x_i)/|\sum_i a_i + b|$. The geometric margins of SVM's are just those margins of the normalized SVM's. Since the theorem from the last section holds for normalized SVM's, and since an SVM and its normalized analogue classify in exactly the same way, we get that the generalization error for an SVM is inversely proportional to the square of it *geometric margins*. The main trick in Support Vector Machines is to insist on a functional margin of at least 1. In that case, the geometric margin will obviously be greater than $1/|\sum_i a_i + b|$. In fact, since we can add $-b$ to all the points in our space, we may conclude that the geometric margin will be greater than $1/|\sum_i a_i|$. Following is a proof of this claim:

$$\begin{aligned}\gamma_g &= \frac{1}{2}(\frac{w}{||w||}x^+ + b - \frac{w}{||w||}x^- - b)\\ &= \frac{1}{2||w||}(wx^+ - wx^-)\\ &= \frac{1}{2||w||}(1-(-1))\\ &= \frac{1}{||w||}\end{aligned}$$

where $w=\sum_i a_i$ and x^+ and x^- are the positively and negatively classified points, respectively.

11.3 The SVM Algorithm

ow we are finally ready to bring the SVM algorithm itself. This is simply -
Find hyperplane (w, b) That

- Minimizes $||W||$
- Subject to $y_i(< w, x_i > +b) \geq 1$ for all i.

This means that the SVM algorithm finds a decision boundary (w, b) that maximizes its margin γ such that all points in the training set are correctly classified, and are at least γ distance away from the boundary (because of the relation between the functional and geometric margins, minimizing $||W|| + b$ while maintaining the functional margin at 1 maximizes the geometric margin - thus minimizing the generalization error).
The SVM algorithm can be cast as a quadratic programming problem. QP problems are problems that have -

- A quadratic goal function,
- Linear constraints, and
- A unique maximum.

There are polynomial time algorithms for solving QP problems.
The SVM QP problem is actually solved using what's called its *dual problem*. In the case of SVM, this is -

- Maximize $W(a) = \sum a_i - 1/2 \sum_{i,j} a_i y_i a_j y_j K(x_i, x_j) + b$
- Subject to
  - $\sum_i a_i y_i = 0$
  - $a_i \geq 0$

11.4 An SVM Application: Text

In this section we will show a possible application of an SVM. The problem is taken from the text domain - given a document, we would like to classify it to one of several categories (sports, news, business, etc'). We will represent each document as a vector in which each coordinate corresponds to a given word. The value in a given coordinate will be the number of times the corresponding word appeared in the text. Obviously, this gives us a huge and sparse vector. Some necessary definitions -

- ft_w - text frequency of the word w - $ft_w = \frac{\sharp documentscontainingw}{no.ofdocuments}$.
- idf_w - inverse document frequency. $idf_w = \frac{no.ofdocuments}{no.ofdocumentscontainingw}$.

- $M_w(x) = ft_w log(idf_w)/K$, K being a normalization constant.

Using the algorithm described, SVM will find a separating hyperplane in this 'document space'. Note that we did not mention the possibility of multiple categories, but they can be handled in the same way as two categories are handled - the classification of a given point will be determined by the hyperspace it is the furthest away from.