Lecture 9: SVM

Lecturer: Yishay Mansour *Scribe: Yoav Cohen and Tomer Sachar Handelman*

9.1 Lecture Overview

In this lecture we present in detail one of the most theoretically well motivated and practically most effective classification algorithms in modern machine learning: Support Vector Machines (SVMs). We begin with building the intuition behind SVMs, continue to define SVM as an optimization problem and discuss how to efficiently solve it. We conclude with an analysis of the error rate of SVMs using two techniques: Leave One Out and VC-dimension.

9.2 Support Vector Machines

9.2.1 The binary classification problem

Support Vector Machine is a supervised learning algorithm that is used to learn a hyperplane that can solve the binary classification problem, which is among the most extensively studied problems in machine learning.

In the binary classification problem we consider an input space X which is a subset of $\mathbb{R}^n$ with $n \geq 1$. The output space Y is simply the set $\{+1, -1\}$, representing our two classes. Given a training set S of m points $S = \{(x_1, y_1), ..., (x_m, y_m)\}$ which are drawn from X i.i.d by an unknown distribution D, we would like to select a hypothesis $h \in H$ that best predicts the classification of other points which are also drawn by D from X.

For example, consider the problem of predicting whether a new drug will successfully treat a certain illness based on the patient's height and weight. The researchers select m people from the population who suffer from the illness, measure their heights and weights and begin treating them with the drug. After the clinical trial is completed, the researchers have m 2-dimensional points (vectors) that represent their patients' heights and weights and for each point a classification to $+1$ which indicates that the drug successfully treated the illness or -1 otherwise. These points can be used as a training set to learn a classification rule, which doctors can use to decide whether to prescribe the drug to the next patient they encounter who suffers from this illness.

There are infinitely many ways to generate a classification rule based on a training set. However, following the principle of Occam's Razor, simpler classification rules (with smaller VC-dimension or Rademacher complexity) provide better learning guarantees. One of the

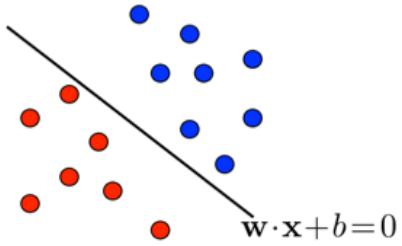

Figure 9.1: A linear classifier

simplest classes of classification rules are the class of *linear classifiers* or hyperplanes. A hypothesis $h \in H$ maps a sample $x \in X$ to $+1$ if $(w \cdot x + b) \geq 0$ or -1 otherwise. Figure 9.1 shows a linear classifier that separates a set of points to two classes, Red and Blue. For the remainder of this text we'll assume that the training set is linearly separable, e.g. there exists a hyperplane (w, b) that separates between the two classes completely.

Definition We define our hypothesis class H of linear classifiers as,

$$H = \{x \to sign(w \cdot x + b) | w \in \mathbb{R}^n, b \in \mathbb{R}\}. \tag{9.1}$$

9.2.2 Choosing a good hyperplane

In previous lectures we studied the Perceptron and Winnow algorithms that learn a hyperplane by continuously adjusting an existing one (iterating through the training set, adjusting whenever the existing hyperplane errors). Intuitively, consider two cases of positive classification by some linear classifier, where in one case $w \cdot x + b = 0.1$ and in the other case $w \cdot x + b = 100$. We are more confident in the decision made by the classifier for the latter point than the former. In the SVM algorithm we'll choose a hyperplane that maximizes the margin between the two classes. The simplest definition of the margin would be to consider the absolute value of $w \cdot x + b$ and is called the Functional Margin:

Definition We define the Functional Margin of S as,

$$\hat{\gamma}_s = \min_{i \in \{1,\ldots,m\}} \hat{\gamma^i}, \tag{9.2}$$

where

$$\hat{\gamma^i} = y^i(w \cdot x^i + b), \tag{9.3}$$

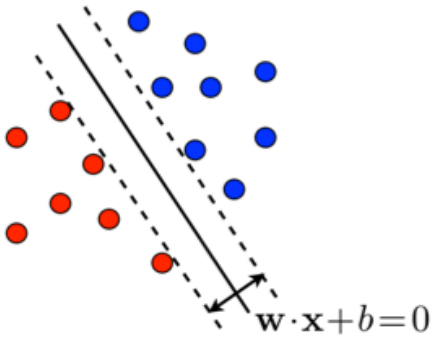

Figure 9.2: A maximal margin linear classifier

and y^i is the classification of x^i according to the hyperplane (w, b).

Figure 9.2 shows a linear classifier that maximizes the margin between the two classes.

Since our purpose is to find w and b that maximize the margin, we quickly notice that one could just scale w and b to increase the margin, but with no effect on the hyperplane. For example, $sign(w \cdot x + b) = sign(5w \cdot x + 5b)$ for all x however the functional margin of $(5w, 5b)$ is 5 times greater than that of (w, b). We can cope with this by adding an additional constraint of $||w|| = 1$. We'll come back to this point later.

Another approach to think about the margin would be to consider the geometric distance between the hyperplane and the points which are closest to it. This measure is called the Geometric Margin. To calculate it, let's take a look at Figure 9.3, which shows the separating hyperplane, its perpendicular vector $\vec{w}$ and the sample x^i. We are interested in calculating the length of AB, denoted as γ^i. As AB is also perpendicular to the hyperplane, it is parallel to $\vec{w}$. Since point A is x^i, point B would be $x^i - \gamma^i \cdot \frac{w}{||w||}$. We will now try to extract γ^i. Since point B is located on the hyperplane, we know that it satisfies the equation $w \cdot x + b = 0$. Hence:

$$w(x^i - \gamma^i \frac{w}{||w||}) + b = 0 \tag{9.4}$$

and solving for γ^i yields:

$$\gamma^i = \frac{w}{||w||} x^i + \frac{b}{||w||} \tag{9.5}$$

To make sure we get a positive length for the symmetrical case where x^i lies below the hyperplane, we multiply by y^i which give us:

$$\gamma^i = y^i(\frac{w}{||w||} x^i + \frac{b}{||w||}) \tag{9.6}$$

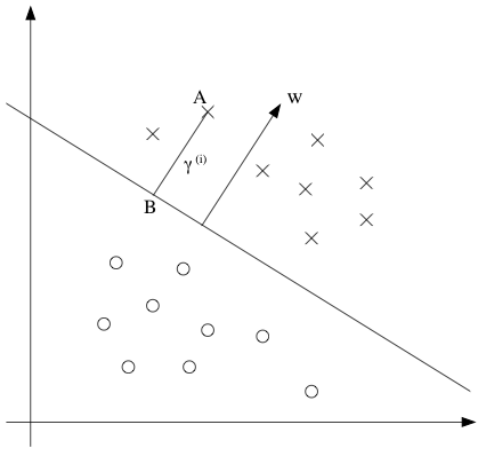

Figure 9.3: A maximal margin linear classifier

Definition We define the Geometric Margin of S as

$$\gamma_s = \min_{i \in \{1,\dots,m\}} \gamma^i, \tag{9.7}$$

where

$$\gamma^i = y^i(\frac{w}{||w||}x^i + \frac{b}{||w||}). \tag{9.8}$$

Note that from the functional margin and geometric margins are related, as follows:

$$\hat{\gamma^i} = ||w|| \cdot \gamma^i, \tag{9.9}$$

and are equal when $||w|| = 1$.

9.2.3 The Support Vector Machine Algorithm

In the previous section we discussed two definitions to the margin and presented the intuition behind seeking a hyperplane that maximizes it. In this section we will try to write an optimization program which finds such a hyperplane. Thus the process of learning an SVM (linear classifier with a maximal margin) is the process of solving an optimization problem based on the training set. In the following programs, we always look for (w, b) which maximizes the margin.

The first program we will write is:

$$\begin{gathered} \max \gamma \quad s.t. \\ y^i(w \cdot x^i + b) \geq \gamma, \quad i = 1, ..., m \\ ||w|| = 1 \end{gathered} \tag{9.10}$$

I.e., we want to maximize γ, subject to each training example having functional margin at least γ. The $||w|| = 1$ constraint moreover ensures that the functional margin equals to the geometric margin, so we are also guaranteed that all the geometric margins are at least γ. Thus, solving this problem will result in (w, b) with the largest possible geometric margin with respect to the training set.

The above program cannot be solved by any of-the-shelf optimization software since the $||w|| = 1$ constraint is non-linear, even non-convex. However, we can discard this constraint if we re-write the objective function using the geometric margin $\hat{\gamma}$ instead of the functional margin γ. Based on 9.9 we can write the following program:

$$\begin{gathered} \max \frac{\hat{\gamma}}{||w||} \quad s.t. \\ y^i(w \cdot x^i + b) \geq \hat{\gamma}, \quad i = 1, ..., m \end{gathered} \tag{9.11}$$

Although we gotten rid of the problematic constraint, we now have a non-convex objective function and the problem remains. Recall that we can scale (w, b) as we wish without changing anything - we will use this to add the scaling constraint that the functional margin of (w, b) with respect to the training set must be 1, i.e. $\hat{\gamma} = 1$. This gives us an objective function of $\max \frac{1}{||w||}$ which we can re-write as $\min \frac{1}{2}||w||^2$ (the factor of 0.5 and the power of 2 do not change the program but make future calculations easier). This gives us the final program:

$$\begin{gathered} \max \frac{1}{2}||w||^2 \quad s.t. \\ y^i(w \cdot x^i + b) \geq 1, \quad i = 1, ..., m \end{gathered} \tag{9.12}$$

Since the objective function is convex (quadratic) and all the constraints are linear, we can solve this problem efficiently using standard quadratic programming (QP) software.

9.3 Convex Optimization

In order to solve the optimization problem presented above more efficiently than generic QP algorithms we will use convex optimization techniques.

9.3.1 Introduction

Definition Let $f : X \to \mathbb{R}$. f is a convex function if

$$\forall x, y \in X, \lambda \in [0,1] \quad f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y). \tag{9.13}$$

Theorem 9.1 *Let $f : X \to \mathbb{R}$ be a differentiable convex function. Then $\forall x, y \in X$:* $f(y) - f(x) \geq \bigtriangledown f(x)(y-x)$.

Definition A convex optimization problem is defined as:
Let $f, g_i : X \to \mathbb{R}, \quad i = 1, ..., m$ be convex functions.
Find $min_{x \in X} \quad f(x) \quad$ s.t.:
$g_i(x) \leq 0, \quad i = 1, ..., m$

In a convex optimization problem we look for a value of $x \in X$ which minimizes $f(x)$ under the constraints $g_i(x) \leq 0, \quad i = 1, ..., m$.

9.3.2 Lagrange Multipliers

The method of Lagrange multipliers is used to find a maxima or minima of a function subject to constraints. We will use this method to solve our optimization problem.

Definition We define the Lagrangian L of function f subject to constraints $g_i, \quad i = 1, ..., m$ as:
$L(x, \alpha) = f(x) + \sum_{i=1}^{m} \alpha_i g_i(x) \quad \forall x \in X, \forall \alpha_i \geq 0$. Here the $\alpha_i' s$ are called the Lagrange Multipliers.

We will now use the Lagrangian to write a program called the Primal program which will be equal to $f(x)$ if all the constraints are met or ∞ otherwise:

Definition We define the Primal program as: $\Theta_P(x) = \max_{\alpha \geq 0} L(x, \alpha)$

Remember that the constraints are of the form $\forall i = 1, ..., m \quad g_i(x) \leq 0$. So, if all constraints are met, then $\sum_{i=1}^{m} \alpha_i g_i(x)$ is maximized when all α_i are 0 (otherwise the summation is negative). Since the summation is 0, we get that $\Theta_P(x) = f(x)$. If some constraint is not met, e.g., $\exists i$ s.t. $g_i(x) > 0$ then the summation is maximized when $\alpha_i \to \infty$ so we get that $\Theta_P(x) = \infty$.

Since the Primal program takes the value of $f(x)$ when all constraints are met, we can re-write our convex optimization problem from the previous section as:

$$\min_{x \in X} \Theta_P(x) = \min_{x \in X} \max_{\alpha \geq 0} L(x, \alpha) \tag{9.14}$$

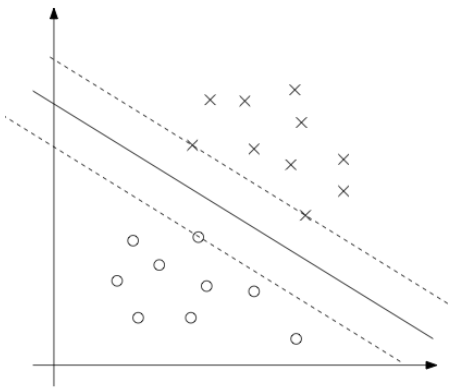

Figure 9.4: A maximal margin classifier and its support vectors

We define $p^* = \min_{x \in X} \Theta_P(x)$ as the value of primal program.

Definition We define the Dual program as: $\Theta_D(x) = \min_{x \in X} L(x, \alpha)$.

Let's now look at $\max_{\alpha \geq 0} \min_{x \in X} \Theta_D(x)$ which is $\max_{\alpha \geq 0} \min_{x \in X} L(x, \alpha)$. It is the same as our primal program only the order of the min and max is different. We also define $d^* = \max_{\alpha \geq 0} \min_{x \in X} L(x, \alpha)$ as the value of the dual program. We would like to show that $d^* = p^*$ which means that if we find a solution to one problem, we find a solution to the second problem.

We start by showing that $p^* \geq d^*$: since the "max min" of any function is always less than the "min max" of the function, we get that:

$$d^* = \max_{\alpha \geq 0} \min_{x \in X} L(x, \alpha) \leq \min_{x \in X} \max_{\alpha \geq 0} L(x, \alpha) = p^* \tag{9.15}$$

Claim 9.2 *If exists x^* and $\alpha^* \geq 0$ which are a saddle point and $\forall \alpha \geq 0$ and $\forall x$ which is feasible: $L(x^*, \alpha) \leq L(x^*, \alpha^*) \leq L(x, \alpha^*)$, then $p^* = d^*$ and x^* is a solution for $\Theta_P(x)$.*

Proof:

$$p^* = \inf_x \sup_{\alpha \geq 0} L(x, \alpha) \leq \sup_{\alpha \geq 0} L(x^*, \alpha) = L(x^*, \alpha^*) = \inf_x L(x, \alpha^*) \leq \sup_{\alpha \geq 0} \inf_x L(x, \alpha) = d^*$$

Since we showed before that $p^* \geq d^*$, and we have that $p^* \leq d^*$, we conclude that $p^* = d^*$. □

9.3.3 Karush-Kuhn-Tucker (KKT) conditions

The KKT conditions derive a characterization of an optimal solution to a convex problem.

Theorem 9.3 *Assume that f and g_i, $i = 1, ..., m$ are differentiable and convex. $\bar{x}$ is a solution to the optimization problem if and only if $\exists \bar{\alpha} \geq 0$ s.t.:*

1. $\bigtriangledown_x L(\bar{x}, \bar{\alpha}) = \bigtriangledown_x f(\bar{x}) + \bar{\alpha} \bigtriangledown_x g(\bar{x}) = 0$
2. $\bigtriangledown_\alpha L(\bar{x}, \bar{\alpha}) = g(\bar{x}) \leq 0$
3. $\bar{\alpha} g(\bar{x}) = \sum \alpha_i g_i(\bar{x}) = 0$

Proof: For every feasible x:

$$\begin{aligned} f(x) - f(\bar{x}) &\geq \bigtriangledown_x f(\bar{x}) \cdot (x - \bar{x}) \\ &\geq -\sum_{i=1}^{m} \bar{\alpha}_i \bigtriangledown_x g_i(\bar{x}) \cdot (x - \bar{x}) \\ &\geq -\sum_{i=1}^{m} \bar{\alpha}_i [g_i(x) - g_i(\bar{x})] \\ &\geq -\sum_{i=1}^{m} \bar{\alpha}_i g_i(x) \geq 0 \end{aligned}$$

The other direction holds as well (not shown here). □

For example, consider the following optimization problem: $\min \frac{1}{2}x^2$ s.t. $x \geq 2$.
We have $f(x) = \frac{1}{2}x^2$ and $g_1(x) = 2 - x$. The Lagrangian will be $L(x, \alpha) = \frac{1}{2}x^2 + \alpha(2 - x)$.
$\frac{\partial L}{\partial x} = x^* - \alpha = 0$ so $x^* = \alpha$
$L(x^*, \alpha) = \frac{1}{2}\alpha^2 + \alpha(2 - \alpha) = 2\alpha - \frac{1}{2}\alpha^2$
$\frac{\partial}{\partial \alpha} L(x^*, \alpha) = 2 - \alpha = 0$ so $\alpha = 2 = x^*$.

9.4 Optimal Margin Classifier

Let's go back to SVMs and re-write our optimization program:

$$\begin{gathered} \min \tfrac{1}{2}||w||^2 \quad s.t.: \\ y^i(w \cdot x^i + b) \geq 1, \quad i = 1, ..., m \\ g_i(w, b) = -y^i(w \cdot x^i + b) + 1 \leq 0 \end{gathered}$$

Following the KKT conditions, we get $\alpha_i \geq 0$ only for points in the training set which have a margin of exactly 1. These are the Support Vectors of the training set. Figure 9.4 shows a maximal margin classifier and its support vectors.

Let's construct the Lagrangian for this problem:
$L(w, b, \alpha) = \frac{1}{2}||w||^2 - \sum_{i=1}^{m} \alpha_i[y^i(w \cdot x^i + b) = 1]$.

Now we will find the dual form of the problem. To do so, we need to first minimize $L(w, b, \alpha)$ with respect to w and b (for fixed α), to get Θ_D, which well do by setting the derivatives of L with respect to w and b to zero. We have:

$$\bigtriangledown_x L(w, b, \alpha) = w - \sum_{i=1}^{m} \alpha_i y^i x^i = 0, \tag{9.16}$$

which implies that:

$$w^* = \sum_{i=1}^{m} \alpha_i y^i x^i. \tag{9.17}$$

When we take the derivative with respect to b we get:

$$\frac{\partial}{\partial b} L(w, b, \alpha) = \sum_{i=1}^{m} \alpha_i y^i = 0 \tag{9.18}$$

We'll take the definition of w^* we derived, plug it back into the Lagrangian, and we get:

$$L(w^*, b^*, \alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y^i y^j \alpha_i \alpha_j x^i x^j - b \sum_{1=1}^{m} \alpha_i y^i. \tag{9.19}$$

From (9.18) we get that the last term is zero so:

$$L(w^*, b^*, \alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y^i y^j \alpha_i \alpha_j x^i x^j = W(\alpha). \tag{9.20}$$

We end up with the following dual optimization problem:

$$\begin{aligned} \max W(\alpha) \qquad & s.t.: \\ \alpha_i \quad &\geq \quad 0, i = 1, ..., m \\ \sum_{i=1}^{m} \alpha_i y^i \quad &= \quad 0 \end{aligned}$$

The KKT conditions hold, so we can solve the dual problem, instead of solving the primal problem, by finding the α^*'s that maximize $W(\alpha)$ subject to the constraints. Assuming we found the optimal α^*'s we define:

$$w^* = \sum_{i=1}^{m} \alpha^* y^i x^i \tag{9.21}$$

which is the solution to the primal problem. We still need to find b^*. To do that, let's assume x^i is a support vector. We get:

$$1 = y^i(w^* \cdot x^i + b^*) \tag{9.22}$$
$$y^i = w^* \cdot x^i + b^* \tag{9.23}$$
$$b^* = y^i - w^* \cdot x^i \tag{9.24}$$

9.4.1 Error Analysis Using Leave-One-Out

In the Leave-One-Out (LOO) method we remove one point at a time from the training set, calculate an SVM for the remaining $m-1$ points and test our result using the removed point.

$$\hat{R}_{LOO} = \frac{1}{m} \sum_{i=1}^{m} I(h_{S-\{x^i\}}(x^i) \neq y^i), \tag{9.25}$$

where the indicator function $I(exp)$ is 1 if exp is true and 0 otherwise.

$$E_{S\sim D^m}[\hat{R}_{LOO}] = \frac{1}{m} \sum_{i=1}^{m} E[I(h_{S-\{x^i\}}(x^i) \neq y^i)] = E_{S,X}[h_{S-\{x^i\}}(x^i) \neq y^i] = E_{S'\sim D^{m-1}}[error(h_{S'})] \tag{9.26}$$

It follows that the expected error of LOO for a training set of size m is the same as for a training set with size $m-1$.

Theorem 9.4

$$E_{S\sim D^m}[error(h_S)] \leq E_{S\sim D^{m+1}}[\frac{N_{SV}(S)}{m+1}] \tag{9.27}$$

where $N_{SV}(S)$ is the number of support vectors in S

Proof: If h_S classifies a point incorrectly, the point must be a support vector. Hence:

$$\hat{R_{LOO}} \leq \frac{N_{SV}(S)}{m+1} \tag{9.28}$$

□

9.4.2 Generalization Bounds Using VC-dimension

Theorem 9.5 *Let $S = \{x : ||x|| \leq R\}$. Let d be the VC-dimension of the hyperplane set $\{sign(w \cdot x) : \min_{x \in S} |w \cdot x| = \lambda, \quad ||w|| \leq \Lambda\}$. Then $d \leq \frac{R^2\Lambda^2}{\lambda^2}$.*

Proof: Assume that the set $\{x^1, ..., x^d\}$ is shattered. So for every $y^i \in \{+1, -1\}$ exists w s.t. $\lambda \leq y^i(w \cdot x^i)$. $i = 1, ..., d$. Summing over d:

$$\lambda d \leq w \sum_{i=1}^{d} y^i x^i \leq ||w|| \cdot || \sum_{i=1}^{d} y^i x^i || \leq \Lambda || \sum_{i=1}^{d} y^i x^i || \tag{9.29}$$

Averaging over the y's with uniform distribution:

$$\lambda d \leq \Lambda E_y || \sum_{i=1}^{d} y^i x^i || \leq \Lambda E_y^{\frac{1}{2}} || \sum_{i=1}^{d} y^i x^i ||^2 = \Lambda \sqrt{E_y[\sum_{i,j} x^i x^j y^i y^j]} \tag{9.30}$$

Since $E_y[y^i y^j] = 0$ when $i \neq j$ and $E_y[y^i y^j] = 1$ when $i = j$, we can conclude that:

$$\lambda d \leq \Lambda \sqrt{E_y[\sum_{i,j} x^i x^j y^i y^j]} \leq \Lambda \sqrt{\sum_i ||x^i||^2} \leq \Lambda \sqrt{dR^2} \tag{9.31}$$

Therefore:

$$d \leq \frac{R^2\Lambda^2}{\lambda^2}. \tag{9.32}$$

□