CS 446 Machine Learning Fall 2016 Oct 20, 2016

Support Vector Machines

Professor: Dan Roth *Scribe: C. Cheng*

Overview

- COLT approach to explaining learning
- Data-dependent VC dimension
- SVM and optimization
- Soft SVM

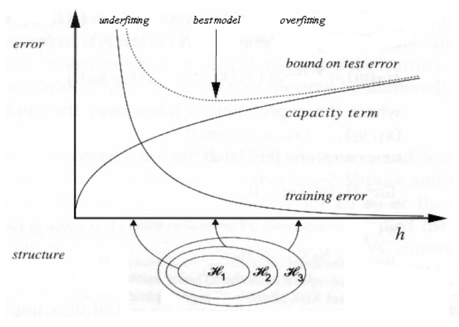
1 Computational Learning Theory Approach

When discussing computational learning theory, we made the key assumptions that we don't know the distribution that governs the generation of examples, but that the training and test distributions are the same.

Given this assumption, we were able to derive the generalization bound as

$$\text{Err}_D(h) < \text{Err}_{TR}(h) + P(\text{VC}(H), \log(1/\delta), 1/m)$$

We showed that the true error over the distribution is bounded by the training error plus something that depends on the complexity parameters. Figure 1



expressive

Figure 1: Trade-off between expressivity and generalization

shows two important curves. As the complexity of the hypothesis goes up, the training error goes down (an expressive enough hypothesis can always fit the data). However, given the generalization bound above, the capacity increases; in effect, test error increases as training error decreases. This is the trade-off between the expressivity of the hypothesis class and the ability to generalize. Ideally, we want to be somewhere in the middle, balancing between small empirical error and generalizable hypothesis. This is also referred to as *structural risk minimization.*

1.1 VC Dimension and Linear Classification

So far, we've discussed VC dimension to help with model selection; classes of functions with smaller VC dimensions are better (eg. linear is better than quadratic). However, this does not help us to choose between functions; when we have two possible linear hypotheses, VC dimension alone doesn't enable us to distinguish between them.

Let $X = \mathbb{R}^2$, $Y = \{+1, -1\}$. In Figure 2, both hypotheses have zero training error, but intuitively we might think h_2 is the better classifier. We cannot use

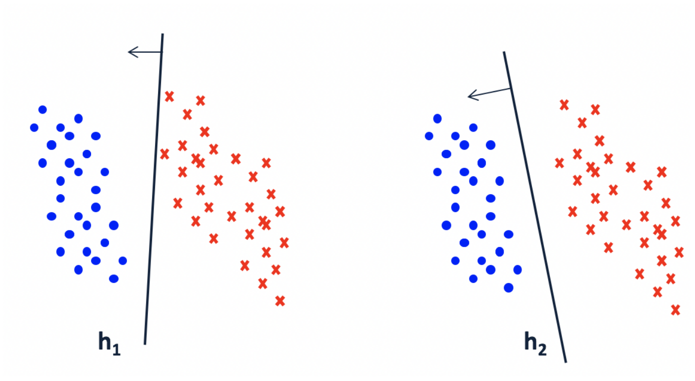


Figure 2: Linear Hypotheses

VC dimension to quantify this intuition, as the generalization bound is the same for both

$$\text{Err}(h) \le \text{err}_{TR}(h) + \text{Poly}\{\text{VC}(H), 1/m, \log(1/\delta)\}$$

$$\text{Err}_{TR}(h_1) = \text{Err}_{TR}(h_2) = 0$$

$$h_1, h_2 \in H_{lin(2)}, \text{VC}(H_{lin(2)}) = 3$$

In order to distinguish between these two hypotheses, we must develop the notion of a *data-dependent VC dimension*: which – rather than simply being dependent on a class of functions – takes the data into account. In the linear classification setting, we must first define the notion of a margin.

1.2 Margin and Data-Dependent VC Dimension

The *margin* γ_i of a point $x_i \in \mathbb{R}^n$ with respect to a linear classifier $h(x) = \text{sign}(w \cdot x + b)$ is defined as the distance of x_i from the hyperplane $w \cdot x + b$,

$$\gamma_i = |(w \cdot x + b)/\|w\||$$

or the size of the dot product divided by the size of the hyperplane.

For a set of points, the margin is the distance of the point closest to the hyperplane

$$\gamma = \min_i \gamma_i = \min_i |(w \cdot x + b)/\|w\||$$

With this in mind, we can develop the definition of data-dependent VC dimension

If H_γ is the space of all linear classifiers in $\mathbb{R}^n$ that separate the training data with margin at least γ, then,

$$\text{VC}(H_\gamma) \leq \min(R^2/\gamma^2, n) + 1$$

where R is the radius of the smallest sphere in $\mathbb{R}^n$ that contains the data. Recall that this is the mistake bound for Perceptron.

For such classifiers, then, we have a generalization bound in the form

$$\text{Err}(h) \leq \text{err}_{TR} + [(O(R^2/\gamma^2) + \log(4/\delta))/m]^{\frac{1}{2}}$$

Thus, if we use such a hypothesis class, and our training error is given by err_{TR}, then we have guarantees that our true error will be bounded as above, meaning that we want as large of a margin as possible. Simply, large γ values gives small data-dependent VC dimension of H_γ. In order to maximize this γ, we must then minimize the size of w.

The algorithm that behaves in this way is referred to as a *support vector machine.*

1.3 Maximal Margin

Given dataset S, we define a *maximal margin* as

$$\gamma(S) = \max_{\|w\|=1} \min_{(x,y) \in S} |y w^T x|$$

For a given w, first find the closest point. Then, find the w of size 1 that gives the maximal margin value. Note that the selection of the point is in the min and therefore the max does not change if we scale w, so its okay to only deal with normalized w values.

To derive the best hypothesis from these hypotheses, we look only at w values that have a fixed size, and choose the one that gives minimum distance, such that

$$\text{argmax}_{\|w\|=1} \min_{(x,y)\in S} |yw^T x|$$

1.4 Margin and VC dimension

There are two principles that will help us derive the algorithm. The first one is the Vapnik Theorem, and it is also the reason we want to maximized the margin. It basically says that learnability is determined by the size of γ. Specifically, if H_γ is the space of all linear classifiers in $\mathbb{R}^n$ that separate the training data with margin at least γ, then

$$\text{VC}(H_\gamma) \leq R^2/\gamma^2$$

where R is the radius of the smallest sphere in $\mathbb{R}^n$ that contains the data. The VC dimension is inversely proportional to γ, which means that we would want to maximize γ. This is the first observatio that will lead to an algorithmic approach.
The second observation gives us a way to maximize the margin. It says that the margin can be maximized if a small size w is retained. Consequently, the algorithm will be: from among all those ws that agree with the data, find the one with the minimal size $\|w\|$.

1.5 Hard SVM

We want to choose the hyperplane that achieves the largest margin. That is, given a data set S, find

$$w^* = \text{argmax}_{\|w\|=1} \min_{(x,y)\in S} |yw^T x|$$

This is basically the same as before. For a given w, first find the closest point. Then, among all ws of size 1, find the w that maximizes this points margin.
To find this w^*, define w_0 to be the solution to the following optimization problem

$$w_0 = \text{argmin}\|w\|^2 : \forall(x,y) \in S, yw^T x \geq 1$$

Then, the normalization of w_0

$$w_0/\|w_0\| = \text{argmax}_{\|w\|=1} \min_{(x,y)\in S} yw^T x$$

is returned as the solution. We claim that it corresponds to the largest margin separating hyperplane.
To prove the claim, define $w' = w_0/\|w_0\|$ and let w^* be the largest-margin

separating hyperplane of size 1. We need to show that $w = w^*$.
First note that $w^*/\gamma(S)$ satisfies the constraints in the optimization problem. Therefore, the size of w_0 is

$$\|w_0\| \leq \|w^*/\gamma(S)\| = 1/\gamma(S)$$

where $\|w^*\| = 1$.
Consequently, for any example (x, y) in the dataset S, from the definition of w', we have

$$yw'^T x = 1/\|w_0\|$$

By the definition of w_0, we have

$$yw_0^T x \geq 1/\|w_0\| \geq \gamma(S)$$

but since $\|w\| = 1$ this implies that w corresponds to the largest margin, that is $w = w^*$.
Essentially, what we've shown is that if we are solving this optimization problem, the solution is the one that maximizes the margin. Immediately, it gives us an algorithm. The sought after weight vector w is the solution of the following optimization problem:

$$\begin{aligned} \min \quad & \frac{1}{2}\|w\|^2 \\ \text{Subject to} \quad & \forall (x, y) \in S, yw^T x \geq 1 \end{aligned}$$

This is an optimization problem in $(n + 1)$ variables, with —$S| = m$ inequality constraints, since all data points have to satisfy the $yw^T x \geq 1$ constraints.

2 Support vector machines

2.1 Dual representation

The name Support Vector Machine stems from the fact that w^* is supported by (i.e. is the linear span of) the examples that are exactly at a distance $1/\|w^*\|$ from the separating hyperplane. These vectors are therefore called support vectors. In the figure below, the circles points are the ones we really care about.

To see it more clearly, we can move to what we call the dual representation. Let w^* be the minimizer of the SVM optimization problem that we formulated before. Consider the set of all the examples $S = \{(x_i, y_i)\}$, let I be the set of all the points that have distance 1 from the hyperplane, such that $I = \{i : w^{*T} x = 1\}$. Then there exists coefficients $\alpha_i > 0$ such that w^* can be written as a linear combination of examples in I

$$w^* = \sum_{i \in I} y_i x_i$$

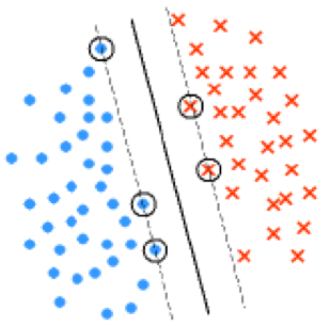


Figure 3: Support vectors

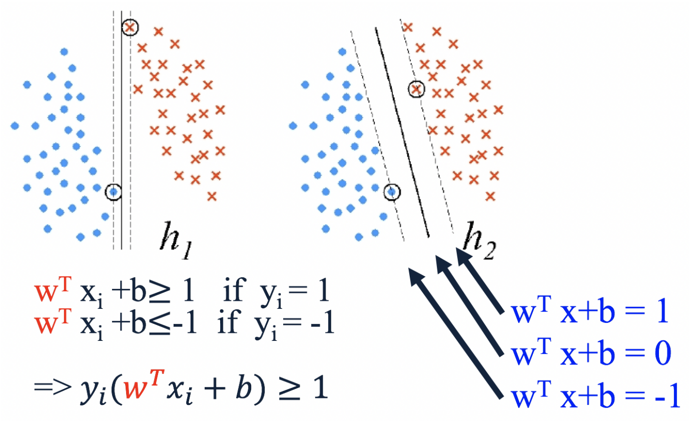
This representation should ring a bell. The solution is a linear combination of important examples only, because the other examples can be moved around without changing the margin. The examples that can change w are those that are closest to the hyperplane. This is really the same representation we have seen before in dual Perceptron.

2.2 Margin of a separating hyperplane

With the threshold unit included, a separating hyperplane is represented as

$$w^T x + b = 0$$

The above formulations are then represented in the figure below.



h_1
h_2
$w^T x_i + b \geq 1$ if $y_i = 1$
$w^T x_i + b \leq -1$ if $y_i = -1$
$\Rightarrow y_i(w^T x_i + b) \geq 1$
$w^T x + b = 1$
$w^T x + b = 0$
$w^T x + b = -1$

Figure 4: Margin of a separating hyperplane

With the assumption that data is linearly separable, the distance between $w^T x +$

$b = 1$ and $w^T x + b = -1$ is $2/\|w\|$. Then, the margin of a linear separator $w^T x + b = 0$ is $2/\|w\|$. Since maximizing $2/\|w\|$ is equivalent to minimizing $\|w\|$, thus minimizing $\frac{1}{2}w^T w$, the optimization problem can be formalized as

$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2}w^T w \\ \text{s.t.} \quad & y_i(w^T x_i + b) \geq 1, \forall (x,y) \in S \end{aligned}$$

2.3 Footnote about the threshold

We were cheating a little bit on the ease of transfer from the homogeneous representation that goes through the origin to the one that does not, but the difference is actually minimal. It does not matter so much. Here is how you can observe this.
Similar to Perceptron, we can augment vectors to handle the bias term. Let $\overrightarrow{x} = (x, 1)$ and $\overrightarrow{w} = (w, b)$, so that $\overrightarrow{w}^T \overrightarrow{x} = w^T x + b$.
Then, consider the optimization formulation we derived

$$\begin{aligned} \min_{\overrightarrow{w}} \quad & \frac{1}{2}\overrightarrow{w}^T \overrightarrow{w} \\ \text{s.t.} \quad & y_i \overrightarrow{w}^T x_i \geq 1, \forall (x,y) \in S \end{aligned}$$

which is what we would write when $\overrightarrow{w}$ goes through the origin.
However, if the bias term b is explicitly written, the formulation is slightly different from the above one, because it is equivalent to

$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2}w^T w + \frac{1}{2}b^2 \\ \text{s.t.} \quad & y_i(w^T x_i + b) \geq 1, \forall (x,y) \in S \end{aligned}$$

and we got $\frac{1}{2}b^2$ as an additional term of the regularizer. This bias term usually doesnt matter. For simplicity, we ignore the bias term, although it is regularized mathematically.

3 Soft SVM

3.1 Key issues

Training of an SVM used to be very time consuming, because people had to solve quadratic programs. Nowadays, stochastic gradient descent (SGD) algorithm can be used to learn SVM. It is kind of interesting, because even though SGD method has been with us since Newton was around, people haven't made a connection. It took time for people to realize that it is the way to do it, due to some technical difficulties.

We were kind of convinced that what we want is to maximize the margin. Is it really what we want to do? Is the objective function we are optimizing the right one?
The figure below shows the data of 17,000 dimensional context sensitive spelling. The histogram illustrate the distances of points from the hyperplane.

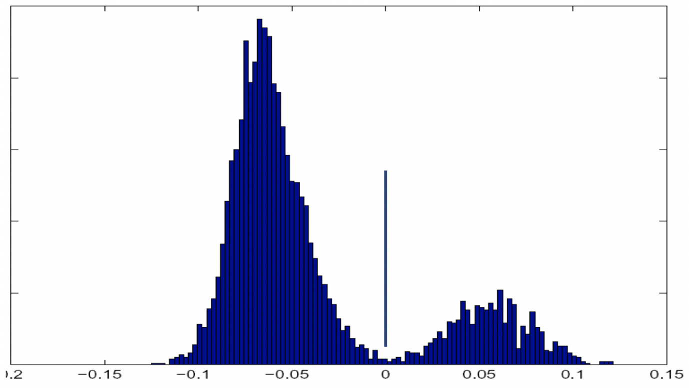


Figure 5: Histogram of distances of points from the hyperplane

In this case, the margin cannot be maximized, because there is a continuum of examples. It is also not clear whether eliminating some points around the origin and maximizing the margin is the right thing to do. The data might not be symmetric. We should take the distribution of points into account, rather than taking the extreme positives and extreme negatives into account. In principle, taking the extremes could make the decision very unstable, because adding en extreme point could bias the linear separator significantly.
In practice, even in the separable case, we may not want to depend on the points closest to the hyperplane but rather on the distribution of the distance. If only a few are close, maybe we can dismiss them. This applies both to generalization bounds and to the algorithm.

3.2 Formulation of soft SVM

In practice, people often need to soften the SVM formulation. By softening, it means taking into account the fact that perhaps learning of the training data cannot be consistent. The data might not be linearly separable, so we might not be able to find the linear separator that leaves all the examples outside some path which we define to be of width 1.
Therefore, the constraint

$$yw^Tx \geq 1$$

typically has to be relaxed. It can be done by introducing a per-example slack variable ξ_i and requiring

$$yw^T x \geq 1 - \xi_i, \xi_i \geq 0$$

The ξ_i can be really small, which means the point actually goes into the fixed the separator. If it is larger, the point could go to the other side of the separator. Rather than solving the standard SVM optimization problem as before, now we have to solve

$$\begin{aligned} \min_{w,\xi_i} \quad & \frac{1}{2}w^T w + C\sum_i \xi_i \\ \text{s.t.} \quad & y_i w^T x_i \geq 1 - \xi_i; \xi_i \geq 0 \quad \forall i \end{aligned}$$

Notice that coefficient C is introduced and it allows us to weigh which component of the objective function is more important to us.
The constraints can also be written as

$$\xi_i \geq 1 - y_i w^T x_i; \xi_i \geq 0 \quad \forall i$$

which is very similar with the context of hinge loss.
In optimum,

$$\xi_i = \max(0, 1 - y_i w^T x_i)$$

Using this notation, basically what we are optimizing is

$$\min_w \quad \frac{1}{2}w^T w + C\sum_i \max(0, 1 - y_i w^T x_i)$$

What is the interpretation of this?
The second term is really the empirical loss, which is what happens on the data. The first term is the regularization term. For sample complexity reasons, we want to make sure that the norm of w is small.
The hard SVM formulation assumes linearly separable data, so the loss is zero. In the general case, the relaxation attempts to minimize the empirical error. It is similar with the move we made from consistent learner to agnostic learner when talking about COLT. Computationally, a natural relaxation is to maximize the margin while minimizing the number of examples that violate the margin (separability) constraints. However, this leads to a non-convex problem that is hard to solve. Instead, we move to a surrogate loss function that is convex, and minimizing the surrogate loss led us to the optimization problem. SVM relies on the hinge loss function, such that we need to optimize

$$\min_w \quad \frac{1}{2}\|w\|^2 + C\sum_{(x,y)\in S} \max(0, 1 - yw^T x)$$

where the parameter C controls the tradeoff between large margin (small $\|w\|$) and small hinge-loss.

3.3 SVM Objective Function

The problem we solved is

$$\min_w \quad \frac{1}{2}\|w\|^2 + C\sum \xi_i$$

where $\xi_i > 0$ is called a slack variable (loss on the ith example), and is defined by

$$\xi_i = \max(0, 1 - y_i w^T x_i)$$

Equivalently, we can say that

$$y_i w^T x_i \geq 1 - \xi_i; \xi_i \geq 0$$

And this can be written as

$$\min_w \quad \frac{1}{2}\|w\|^2 + C\sum \xi_i$$

The regularization term $\frac{1}{2}\|w\|^2$ can be replaced by other regularization functions. The empirical loss term $C\sum \xi_i$ can be replaced by other loss functions. What we have been talking about is the L1-loss SVM. The regularization term be changed in multiple ways. One way is to square the hinge loss to obtain

$$\min_w \quad \frac{1}{2}w^T w + C\sum_{i=1}^{I} \max(0, 1 - y_i w^T x_i)$$

which gives the L2-loss SVM.
Changing the loss function to the following one

$$\min_w \quad \frac{1}{2}w^T w + C\sum_{i=1}^{I} \log(1 + e^{-y_i w^T x_i})$$

gives the logistic regression.
Graphically, the three loss functions are shown in the figure below

Conceptually, it is the same thing, but each replacement gives us a slightly different algorithm. The general form of a learning algorithm is still minimizing the empirical loss, while regularizing to avoid over fitting. Theoretically, adding a regularization term motivated improvement over the original algorithm we have seen at the beginning of the semester.

3.4 Overfitting and underfitting

We have played with the notion of overfitting and underfitting, or bias and variance, or empirical error and regularization. They are basically the same

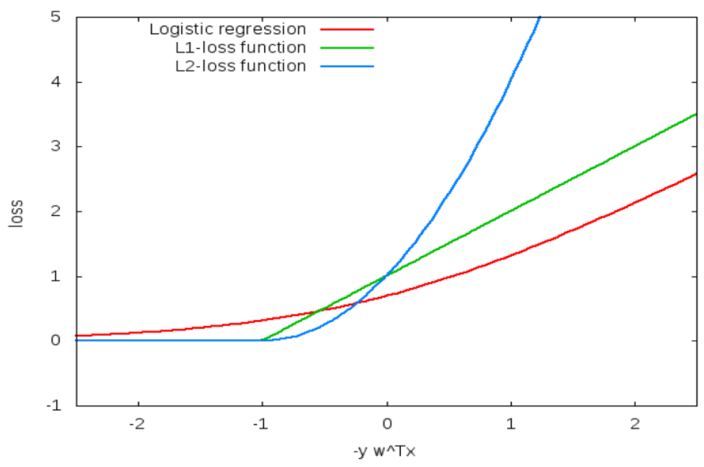


Figure 6: Different loss functions with no significant difference

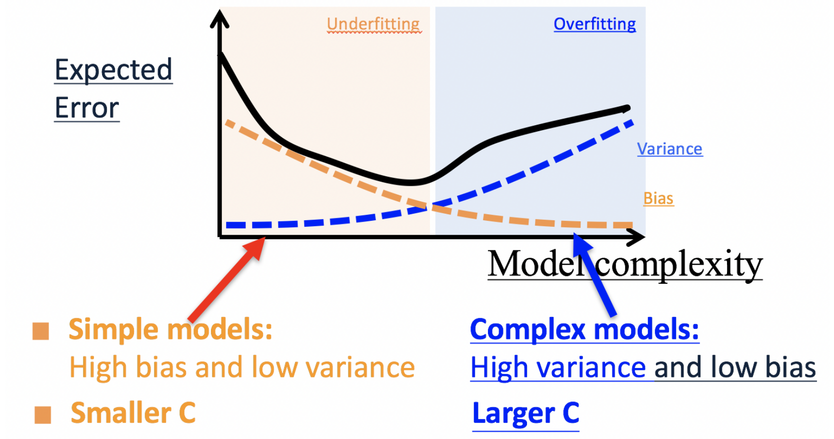


Figure 7

thing. The following figure shows the relationship of expected error versus model complexity.

When the model is simple, you don't have many options and have to make a lot of errors on the training data, thus the bias will be large. However, it will be a stable model with low variance, because the model is simple and changing the data a little bit does not affect much. On the other hand, an expressive model can fit the training data well, but the model will be very sensitive to change of the data. This also guides you to choose a small C or a large C in the tradeoff between regularization and empirical error.

4 Optimization

See slides for details. Additional slides on optimization are available.