Support Vector Machines (SVM)

Dan Roth
danroth@seas.upenn.edu|http://www.cis.upenn.edu/~danroth/|461C, 3401 Walnut

Slides were created by Dan Roth (for CIS519/419 at Penn or CS446 at UIUC), and other authors who have made their ML slides available.

Penn Engineering

Administration (11/4/20)

Are we recording? YES!

Available on the web site

- Remember that all the lectures are available on the website before the class
  - Go over it and be prepared
  - A new set of written notes will accompany most lectures, with some more details, examples and, (when relevant) some code.
- HW 3: Due on 11/16/20
  - You cannot solve all the problems yet.
  - Less time consuming; no programming
- Projects

Projects

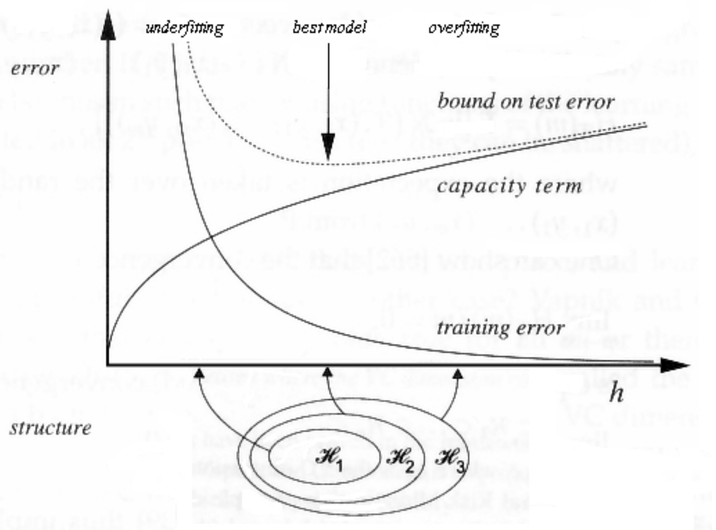
- CIS 519 students need to do a team project
  - Teams will be of size 2-4
  - We will help grouping if needed
- There will be 3 projects.
  - Natural Language Processing (Text)
  - Computer Vision (Images)
  - Speech (Audio)
- In all cases, we will give you datasets and initial ideas
  - The problem will be multiclass classification problems
  - You will get annotated data only for some of the labels, but will also have to predict other labels
  - 0-zero shot learning; few-shot learning; transfer learning
- A detailed note will come out today.
- Timeline:
  - 11/11 Choose a project and team up
  - 11/23 Initial proposal describing what your team plans to do
  - 12/2 Progress report
  - 12/15-20 (TBD) Final paper + short video
- Try to make it interesting!

COLT approach to explaining Learning

- No Distributional Assumption
- Training Distribution is the same as the Test Distribution
- Generalization bounds depend on this view and affects model selection.

$$Err_D(h) < Err_{TR}(h) + P(VC(H), \log(\frac{1}{\gamma}), \frac{1}{m})$$

- This is also called the "Structural Risk Minimization" principle.



COLT approach to explaining Learning

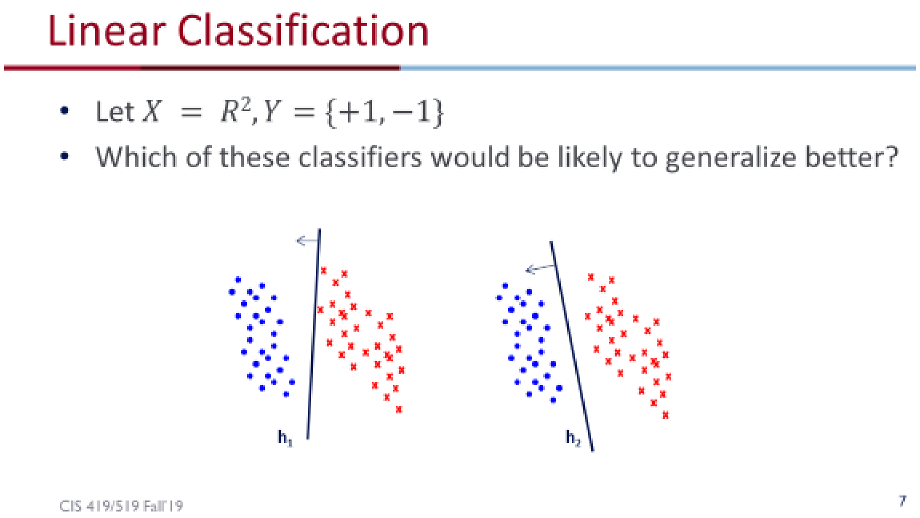
- No Distributional Assumption
- Training Distribution is the same as the Test Distribution
- Generalization bounds depend on this view and affect model selection.

$$Err_D(h) < Err_{TR}(h) + P(VC(H), \log(1/\Upsilon), 1/m)$$

  - As presented, the VC dimension is a combinatorial parameter that is associated with a class of functions.
- We know that the class of linear functions has a lower VC dimension than the class of quadratic functions.
  - But this notion can be refined to depend on a given data set, and this way directly affect the hypothesis chosen for a given data set.

Data Dependent VC dimension

- So far, we discussed VC dimension in the context of a <u>fixed</u> class of functions.
- We can also parameterize the class of functions in interesting ways.
- Consider the class of linear functions, parameterized by their margin. Note that this is a data dependent notion.



VC and Linear Classification

- Recall the VC based generalization bound:

$$Err(h) \leq err_{TR}(h) + Poly\{VC(H), \frac{1}{m}, \log(\frac{1}{\Upsilon})\}$$

- Here we get the same bound for both classifiers:

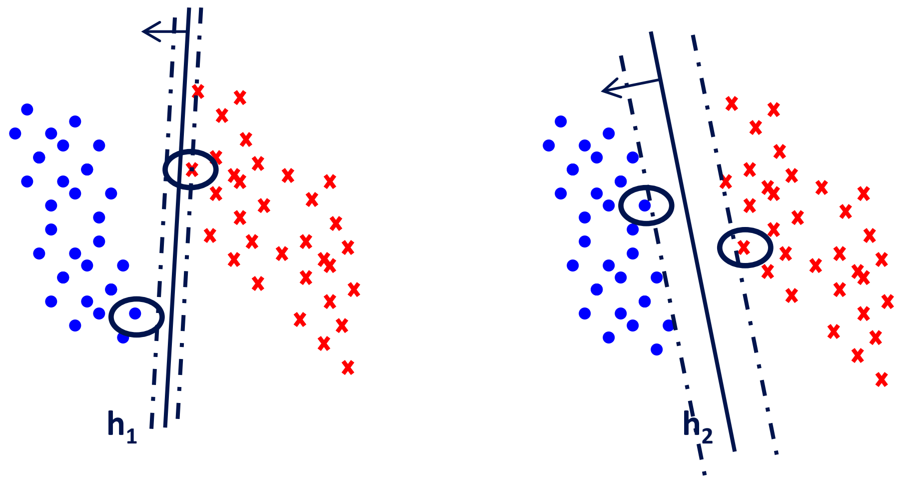
$$Err_{TR}(h_1) = Err_{TR}(h_2) = 0$$

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

- How, then, can we explain our intuition that h_2 should give better generalization than h_1?

Linear Classification

- Although both classifiers separate the data, the distance with which the separation is achieved is different:



Concept of Margin

- The margin Υ_i of a point $\boldsymbol{x}_i \in \boldsymbol{R}^n$ with respect to a linear classifier $h(\boldsymbol{x}) = sign(\boldsymbol{w}^T \cdot \boldsymbol{x} + b)$ is defined as the distance of $\boldsymbol{x}_i$ from the hyperplane $\boldsymbol{w}^T \cdot \boldsymbol{x} + b = 0$:

$$\Upsilon_i = \left| \frac{\boldsymbol{w}^T \cdot \boldsymbol{x}_i + b}{||\boldsymbol{w}||} \right|$$

- The margin of a set of points $\{\boldsymbol{x}_1, \ldots \boldsymbol{x}_m\}$ with respect to a hyperplane $\boldsymbol{w}$, is defined as the margin of the point closest to the hyperplane:

$$\Upsilon = \min_i \Upsilon_i = min_i \left| \frac{\boldsymbol{w}^T \cdot \boldsymbol{x}_i + b}{||\boldsymbol{w}||} \right|$$

VC and Linear Classification

- Theorem: If H_Υ is the space of all linear classifiers in $\boldsymbol{R}^n$ that separate the training data with margin at least Υ, then:

$$VC(H_\Upsilon) \leq \min(\frac{R^2}{\Upsilon^2}, n) + 1,$$

In particular, you see here that for "general" linear separators of dimensionality n, the VC is n+1

- Where R is the radius of the smallest sphere (in $\boldsymbol{R}^n$) that contains the data.
- Thus, for such classifiers, we have a bound of the form:

$$Err(h) \leq err_{TR}(h) + \left\{ \frac{O\left(\frac{R^2}{\Upsilon^2}\right) + \log\left(\frac{4}{\delta}\right)}{m} \right\}^{1/2}$$

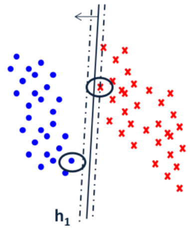
Towards Max Margin Classifiers

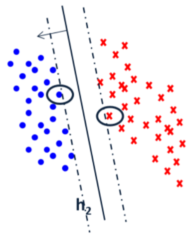
- First observation: When we consider the class H_Υ of linear hypotheses that separate a given data set with a margin Υ, we see that
  - Large Margin Υ → Small VC dimension of H_Υ

But, how can we do it algorithmically?

- Consequently, our goal could be to find a separating hyperplane $\boldsymbol{w}$ that maximizes the margin of the set S of examples.

- A second observation that drives an algorithmic approach is that:
  - Small $||\boldsymbol{w}||$ → Large Margin





- Together, this leads to an algorithm: from among all those $\boldsymbol{w}$'s that agree with the data, find the one with the minimal size $||\boldsymbol{w}||$
  - But, if $\boldsymbol{w}$ separates the data, so does $\boldsymbol{w}/7$....
  - We need to better understand the relations between $\boldsymbol{w}$ and the margin

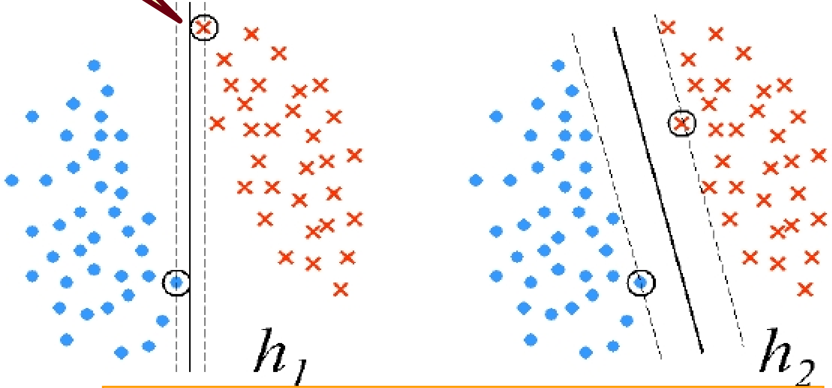
Maximal Margin

The distance between a point x and the hyperplane defined by w is: $|w^T x|/||w||$

- This discussion motivates the notion of a maximal margin.
- The maximal margin of a data set S is defined as:

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

A hypothesis (w) has many names



How does it help us to derive these h's?

1. For a given w: Find the closest point.
2. Then, across all w's (of size 1), find the point for which this closets point is the farthest (that gives the maximal margin).

Note: the selection of the point is in the min and therefore the max does not change if we scale w, so it's okay to only deal with normalized w's.

Interpretation 1: among all w's, choose the one that maximizes the margin.

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

Recap: Margin and VC dimension

Believe

Theorem (Vapnik): If H_Υ is the space of all linear classifiers in $\boldsymbol{R}^n$ that separate the training data with margin at least Υ, then

$$VC(H_\Upsilon) \leq R^2/\Upsilon^2$$

– where R is the radius of the smallest sphere (in $\boldsymbol{R}^n$) that contains the data.

- This is the first observation that will lead to an algorithmic approach.
- The second observation is that: Small $||\boldsymbol{w}||$ → Large Margin
- Consequently, the algorithm will be: from among all those $\boldsymbol{w}$'s that agree with the data, find the one with the minimal size $||\boldsymbol{w}||$

We'll show this

From Margin to $||w||$

- We want to choose the hyperplane that achieves the largest margin. That is, given a data set S, find:
  - $\boldsymbol{w}^* = \text{argmax}_{||\mathbf{w}||=1} \min_{(\mathbf{x},\mathbf{y}) \in S} |y\, \boldsymbol{w}^T \boldsymbol{x}|$

- How to find this $\boldsymbol{w}^*$?
- Claim: Define $\boldsymbol{w}_0$ to be the solution of the optimization problem:
  - $\boldsymbol{w}_0 = argmin \{||\boldsymbol{w}||^2 : \forall (\boldsymbol{x}, y) \in S, y\, \boldsymbol{w}^T \boldsymbol{x} \geq 1\}.$
  - Then:
  - $\mathbf{w}_0/||\mathbf{w}_0|| = argmax_{||\boldsymbol{w}||=1} \min_{(\boldsymbol{x},y)\in S} y\, \boldsymbol{w}^T \boldsymbol{x}$

Interpretation 2: among all $\boldsymbol{w}$'s that separate the data with margin 1, choose the one with minimal size.

- That is, the normalization of $\boldsymbol{w}_0$ corresponds to the largest margin separating hyperplane.

The next slide will show that the two interpretations are equivalent

From Margin to $||\boldsymbol{w}||$(2)

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

And, recall that $\Upsilon(S)$ is the maximal margin for the set S

- **Claim:** Define $\boldsymbol{w}_0$ to be the solution of the optimization problem:
  - $\boldsymbol{w}_0 = argmin\, \{||\boldsymbol{w}||^2 : \forall(\boldsymbol{x}, y) \in S, y\, \boldsymbol{w}^T \boldsymbol{x} \geq 1\}$(**)

  Then:
  - $\boldsymbol{w}_0/||\boldsymbol{w}_0|| = argmax_{||\boldsymbol{w}||=1} \min_{(\boldsymbol{x},y)\in S} y\, \boldsymbol{w}^T \boldsymbol{x}$

That is, the normalization of $\boldsymbol{w}_0$ corresponds to the largest margin separating hyperplane.

- **Proof:** Define $\boldsymbol{w}' = \boldsymbol{w}_0/||\boldsymbol{w}_0||$ and let $\boldsymbol{w}^*$ be the largest-margin separating hyperplane of size 1. We need to show that $\boldsymbol{w}' = \boldsymbol{w}^*$.

  Note first that $\frac{\boldsymbol{w}^*}{\Upsilon(S)}$ satisfies the constraints in (**);

  therefore: $||\boldsymbol{w}_0|| \leq ||\boldsymbol{w}^*/\Upsilon(S)|| = 1/\Upsilon(S)$.

- Consequently:

$$\forall\, (\boldsymbol{x}, y) \in S \quad y\, \boldsymbol{w}'^T \boldsymbol{x} = \frac{1}{||\boldsymbol{w}_0||} y \boldsymbol{w}_0^T \boldsymbol{x} \geq 1/||\boldsymbol{w}_0|| \geq \Upsilon(S)$$

But since $||\boldsymbol{w}'|| = 1$ this implies that $\boldsymbol{w}'$ corresponds to the largest margin, that is

$$\boldsymbol{w}' = \boldsymbol{w}^*$$

Def. of $\boldsymbol{w}_0$

Def. of $\boldsymbol{w}^*$

Def. of $\boldsymbol{w}'$

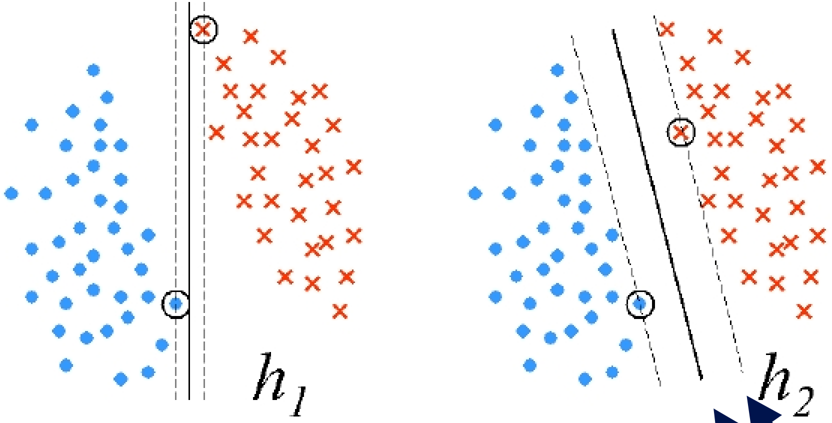
Def. of $\boldsymbol{w}_0$

Prev. ineq.

Def. of $\boldsymbol{w}^*$

Margin of a Separating Hyperplane

- A separating hyperplane: $\boldsymbol{w}^T \boldsymbol{x} + b = 0$



$$\boldsymbol{w}^T \boldsymbol{x}_i + b \geq 1 \quad if \quad y_i = 1$$
$$\boldsymbol{w}^T \boldsymbol{x}_i + b \leq -1 \quad if \quad y_i = -1$$

$$\rightarrow \; y_i(\boldsymbol{w}^T \boldsymbol{x}_i + b) \geq 1$$

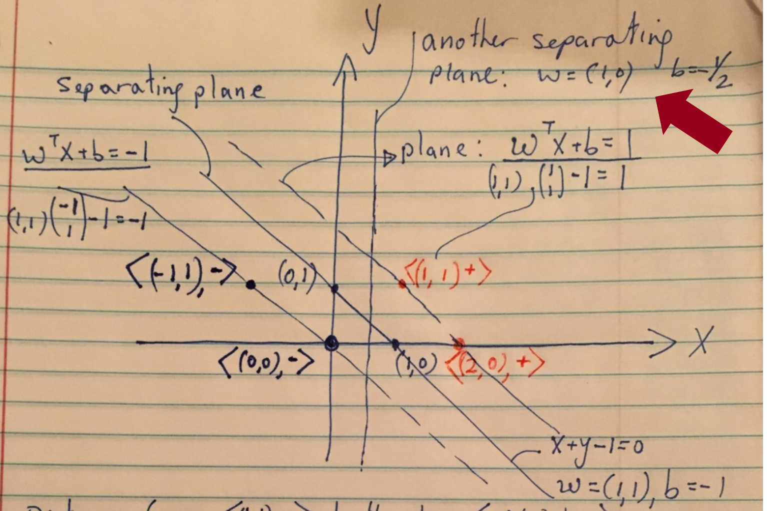
$\boldsymbol{w}^T \boldsymbol{x} + b = 0$

$\boldsymbol{w}^T \boldsymbol{x} + b = -1$

Distance between
$\boldsymbol{w}^T \boldsymbol{x} + b = +1$ *and* -1 is $2/||\boldsymbol{w}||$
What we did:

1. Consider all possible $\boldsymbol{w}$ with different angles
2. Scale $\boldsymbol{w}$ such that the <u>constraints are tight</u> (closets points are on the +/-1 line)
3. Pick the one with largest margin/minimal size

Assumption: data is linearly separable
Let $(\boldsymbol{x}_0 \;\; y_0)$ be a point on $\boldsymbol{w}^T\boldsymbol{x} + b = 1$
Then its distance to the separating plane $\boldsymbol{w}^T \boldsymbol{x} + b = 0$ is: $|\boldsymbol{w}^T \boldsymbol{x}_0 + b|/||\boldsymbol{w}|| = 1/||\boldsymbol{w}||$



Distance from $\langle(1,1)\ +\rangle$ to the plane $\langle w=(1,1),\ b=-1\rangle$

is : $\dfrac{(1,1)\binom{1}{1}-1}{\sqrt{2}} = \dfrac{1}{\sqrt{2}} = \dfrac{\sqrt{2}}{2} = \dfrac{1}{\|w\|}$

We could have represented $X+Y-1=0$ as

$\langle w=(2,2)\ b=-2\rangle$; $2X+2Y-2=0$

Then the ⊕ plane would be $w^TX+b=2$

$(2,2)\binom{1}{1}-2=2$

⊖ plane would be $(2,2)\binom{-1}{1}-2=-2$

$w^TX+b=-2$

For the second plane $w=(1,0)$, $b=-1/2$:

Check $\langle(1,1),+\rangle$: $(1,0)\binom{1}{1}-\frac{1}{2}=\frac{1}{2}$.

Not good, since we want to separate the positive points better, so we scale $\langle w,b\rangle$:

$(c,0)\binom{1}{1}-\frac{c}{2}=1$ ⇐ That's what we want.

$\Rightarrow$ $c-c/2=1$ $\quad c=2$.

$\Rightarrow$ We rename the plane to be $w=(2,0)$, $b=-1$

Now: $+$: $(2,0)\binom{1}{1}-1=1$

$+$: $(2,0)\binom{2}{0}-1=3$

$-$: $(2,0)\binom{-1}{1}-1=-3$

$-$: $(2,0)\binom{0}{0}-1=-1$

Good!

But, now $\|w\|=\|(2,0)\|=2$

Before we had $\|w\|=\|(1,1)\|=\sqrt{2}$, Better

Administration (11/9/20)

Are we recording? YES!

Available on the web site

- Remember that all the lectures are available on the website before the class
  - Go over it and be prepared
  - A new set of written notes will accompany most lectures, with some more details, examples and, (when relevant) some code.
- HW 3: Due on 11/16/20
  - You cannot solve all the problems yet.
  - Less time consuming; no programming
- Cheating
  - Several problems in HW1 and HW2

Projects

- CIS 519 students need to do a team project: Read the project descriptions
  - Teams will be of size 2-4
  - We will help grouping if needed

- There will be 3 projects.
  - Natural Language Processing (Text)
  - Computer Vision (Images)
  - Speech (Audio)

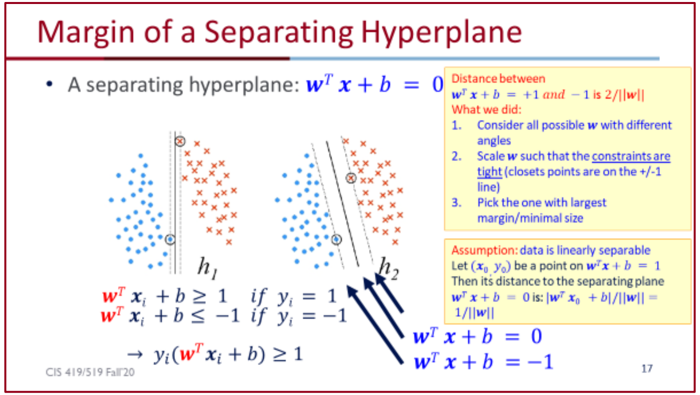
- In all cases, we will give you datasets and initial ideas
  - The problem will be multiclass classification problems
  - You will get annotated data only for some of the labels, but will also have to predict other labels
  - 0-zero shot learning; few-shot learning; transfer learning

- A detailed note will come out today.

- Timeline:
  - 11/11 Choose a project and team up
  - 11/23 Initial proposal describing what your team plans to do
  - 12/2 Progress report
  - 12/15-20 (TBD) Final paper + short video
- Try to make it interesting!

Hard SVM Optimization

- We have shown that the sought-after weight vector $\boldsymbol{w}$ is the solution of the following optimization problem:

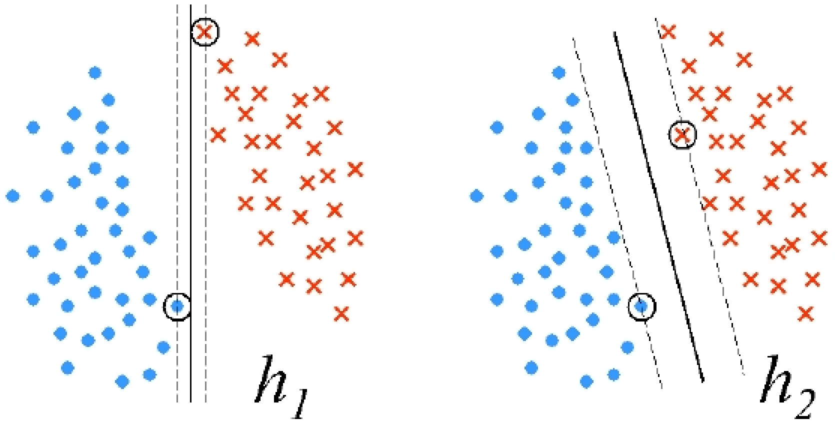


SVM Optimization: (***)

- Minimize: $½ ||\boldsymbol{w}||^2$
- Subject to: $\forall (\boldsymbol{x}, y) \in S$: $\quad y\, \boldsymbol{w}^T \boldsymbol{x} \geq 1$

- This is a quadratic optimization problem in $(n + 1)$ variables, with $|S| = m$ inequality constraints.
- It has a unique solution.

Maximal Margin



The margin of a linear separator $\mathbf{w}^T \mathbf{x} + b = 0$ is $\frac{1}{||\mathbf{w}||}$

$$\max \frac{1}{||\mathbf{w}||} = \min ||\mathbf{w}||$$
$$= \min \tfrac{1}{2}\, \mathbf{w}^T \mathbf{w}$$

$$\min_{w,b} \frac{1}{2} \mathbf{w}^T \mathbf{w}$$

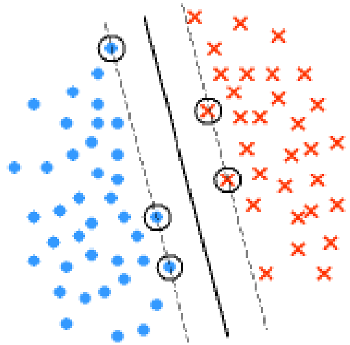
$$\text{s.t} \quad y_i(\mathbf{w}^{\mathrm{T}} \mathbf{x}_i + b) \geq 1, \forall (\mathbf{x}_i, y_i) \in S$$

Support Vector Machines

- The name "Support Vector Machine" stems from the fact that $\boldsymbol{w}^*$ is supported by (i.e. is the linear span of) the examples that are exactly at a distance $1/||\boldsymbol{w}^*||$ from the separating hyperplane. These vectors are therefore called support vectors.

- Theorem: Let $\boldsymbol{w}^*$ be the minimizer of the SVM optimization problem (***) for $S = \{(\boldsymbol{x}_i, y_i)\}$. Let $I = \{i\colon \boldsymbol{w}^{*T}\boldsymbol{x}_i = 1\}$.
  Then there exists coefficients $\alpha_i > 0$ such that:

$$\boldsymbol{w}^* = \sum_{i \in I} \alpha_i\, y_i\, \boldsymbol{x}_i$$



This representation should ring a bell…

How did we call this representation of w?

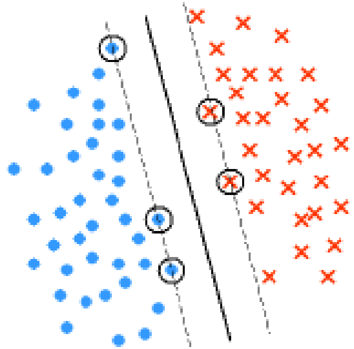
Start the presentation to see live content. For screen share software, share the entire screen. Get help at **pollev.com/app**

Duality

- This, and other properties of Support Vector Machines are shown by moving to the dual problem.

- Theorem: Let $\boldsymbol{w}^*$ be the minimizer of the SVM optimization problem (***) for $S = \{(\boldsymbol{x}_i, y_i)\}$.
  Let $I = \{ i: y_i(\boldsymbol{w}^{*T}\boldsymbol{x}_i + b) = 1\}$.
  Then there exists coefficients $\alpha_i > 0$ such that:

$$\boldsymbol{w}^* = \sum_{i \in I} \alpha_i y_i \boldsymbol{x}_i$$



Footnote about the threshold

- Similar to Perceptron, we can augment vectors to handle the bias term
  $\bar{\boldsymbol{x}} \Leftarrow (\boldsymbol{x}\,,\ 1);\ \ \bar{\boldsymbol{w}} \Leftarrow (\boldsymbol{w}\,,\ b)$ so that $\bar{\boldsymbol{w}}^T\bar{\boldsymbol{x}} = \boldsymbol{w}^T\boldsymbol{x} + b$
- Then consider the following formulation

$$\min_{\bar{\boldsymbol{w}}} \ \frac{1}{2}\bar{\boldsymbol{w}}^T\bar{\boldsymbol{w}} \quad \text{s.t} \ \ \mathrm{y_i}\bar{\boldsymbol{w}}^{\mathrm{T}}\bar{\boldsymbol{x}}_{\mathrm{i}} \ge 1, \forall(\boldsymbol{x}_i, y_i) \in \mathrm{S}$$

- However, this formulation is slightly different from (***), because it is equivalent to

$$\min_{\boldsymbol{w},b} \ \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + \underbrace{\frac{1}{2}b^2} \ \ \text{s.t} \ \ \mathrm{y_i}(\boldsymbol{w}^{\mathrm{T}}\mathbf{x}_{\mathrm{i}} + b) \ge 1, \forall(\boldsymbol{x}_i, y_i) \in \mathrm{S}$$

The bias term is included in the regularization.
This usually doesn't matter

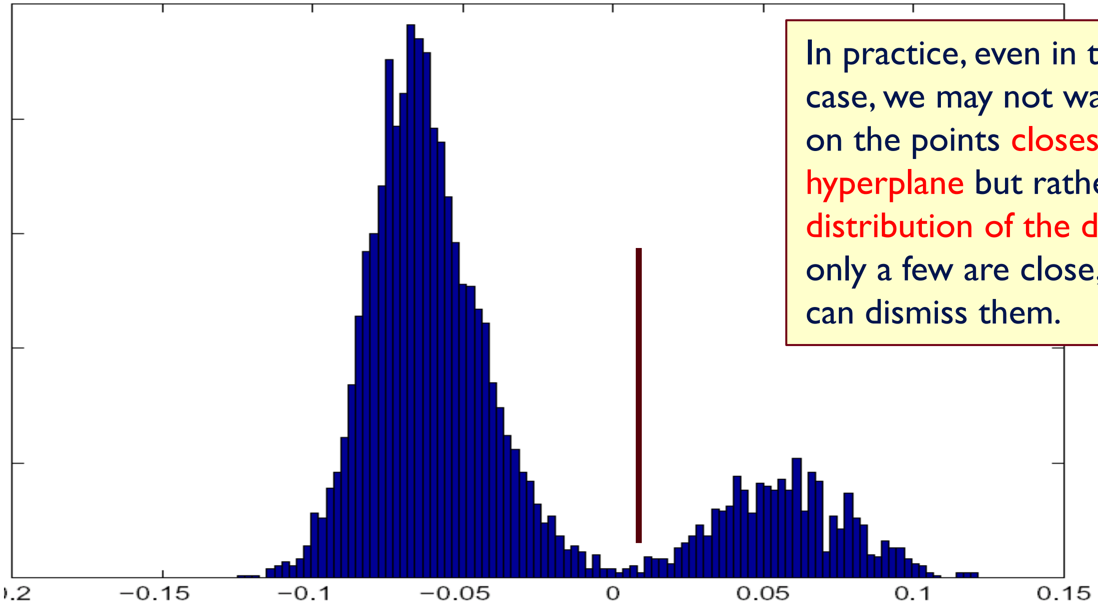
For simplicity, we ignore the bias term

Key Issues

- Computational Issues
  - Training of an SVM used to be is very time consuming – solving quadratic program.
  - Modern methods are based on Stochastic Gradient Descent and Coordinate Descent and are much faster.
- Is it really optimal?
  - Is the objective function we are optimizing the “right” one?

Real Data

- 17,000 dimensional context sensitive spelling
- Histogram of distance of points from the hyperplane



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.

Soft SVM

- The hard SVM formulation assumes linearly separable data.
- A natural relaxation:
  - maximize the margin while minimizing the # of examples that violate the margin (separability) constraints.
- However, this leads to non-convex problem that is hard to solve.
- Instead, we relax in a different way, that results in optimizing a surrogate loss function that is convex.

Soft SVM

- Notice that the relaxation of the constraint:

$$y_i \boldsymbol{w}^T \boldsymbol{x}_i \geq 1$$

- Can be done by introducing a slack variable ξ_i (per example) and requiring:

$$y_i \boldsymbol{w}^T \boldsymbol{x}_i \geq 1 - \xi_i \; ; \xi_i \geq 0$$

- Now, we want to solve:

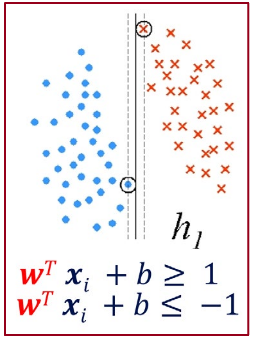
$$\min_{\boldsymbol{w}, \xi_i} \; \frac{1}{2} \boldsymbol{w}^T \boldsymbol{w} + C \sum_i \xi_i$$

$$\text{s.t} \quad y_i \boldsymbol{w}^T \boldsymbol{x}_i \geq 1 - \xi_i \; ; \xi_i \geq 0 \quad \forall i$$

> - **A large value of C** means that we want ξ_i to be small; that is, misclassifications are bad – we focus on a small training error (at the expense of margin).
> - **A small C** results in more training error, but hopefully better true error.

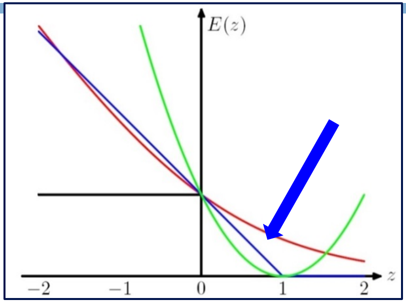
Soft SVM (2)

- Now, we want to solve:



$$\min_{\mathbf{w},\xi_i} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_i \xi_i$$

$$\text{s.t} \quad \xi_i \geq 1 - y_i \mathbf{w}^T x_i \qquad \xi_i \geq 0 \quad \forall i$$



In optimum, $\xi_i = \max(0, 1 - y_i\, \mathbf{w}^T \mathbf{x}_i)$

- Which can be written as:

$$\min_{\mathbf{w}} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_i \max(0, 1 - y_i \mathbf{w}^T \mathbf{x}_i).$$

- What is the interpretation of this?

SVM Objective Function

- The problem we solved is:

$$Min\ ½\ ||\boldsymbol{w}||^2\ +\ c\sum \xi_i$$

- Where $\xi_i > 0$ is called a slack variable, and is defined by:
  - $\xi_i = \max(0, 1 - y_i \boldsymbol{w}^T \boldsymbol{x}_i)$
  - Equivalently, we can say that: $y_i \boldsymbol{w}^T \boldsymbol{x}_i \geq 1 - \xi_i;\ \xi_i \geq 0$
- And this can be written as:

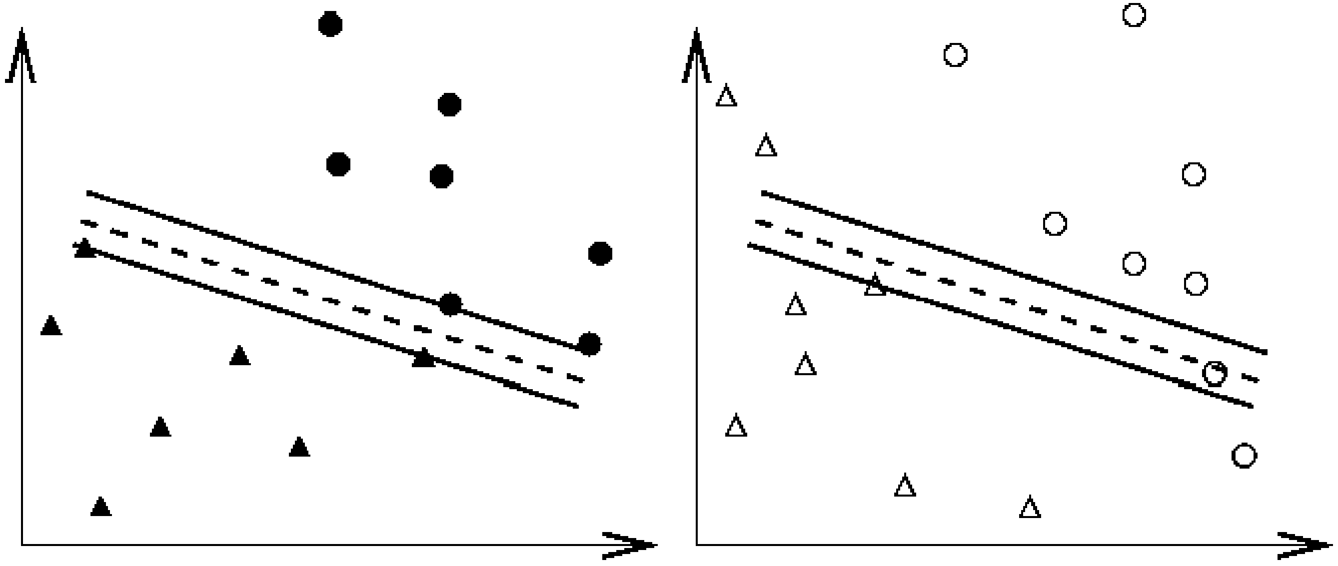
$$Min\ ½\ ||\boldsymbol{w}||^2 \quad + \quad c\sum\xi_i$$

Regularization term — Can be replaced by other **regularization functions**

Empirical loss — Can be replaced by other **loss functions**

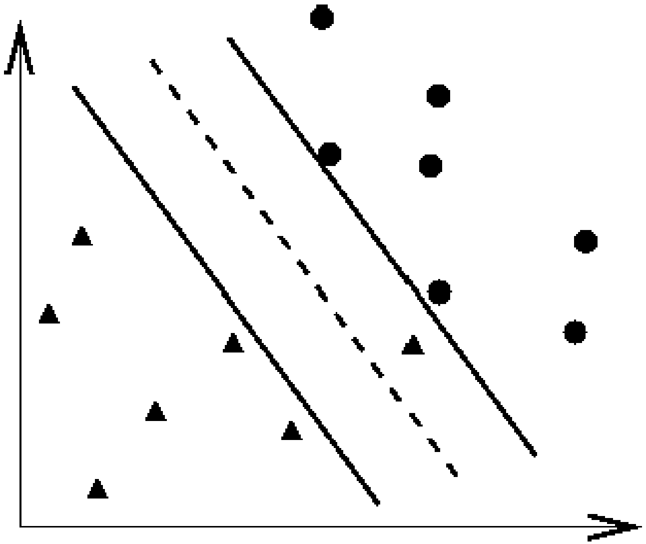
- General Form of a learning algorithm:
  - Minimize empirical loss, and Regularize (to avoid over fitting)
  - Theoretically motivated improvement over the original algorithm we've seen at the beginning of the semester.

Balance between regularization and empirical loss

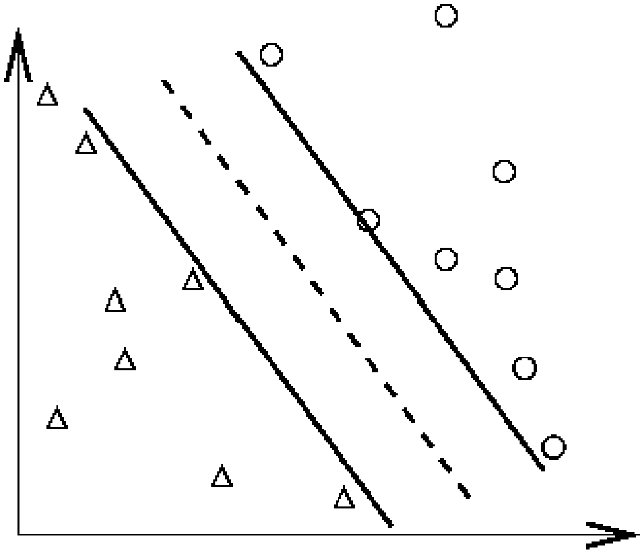


(a) Training data and an over-fitting classifier (b) Testing data and an over-fitting classifier

Balance between regularization and empirical loss



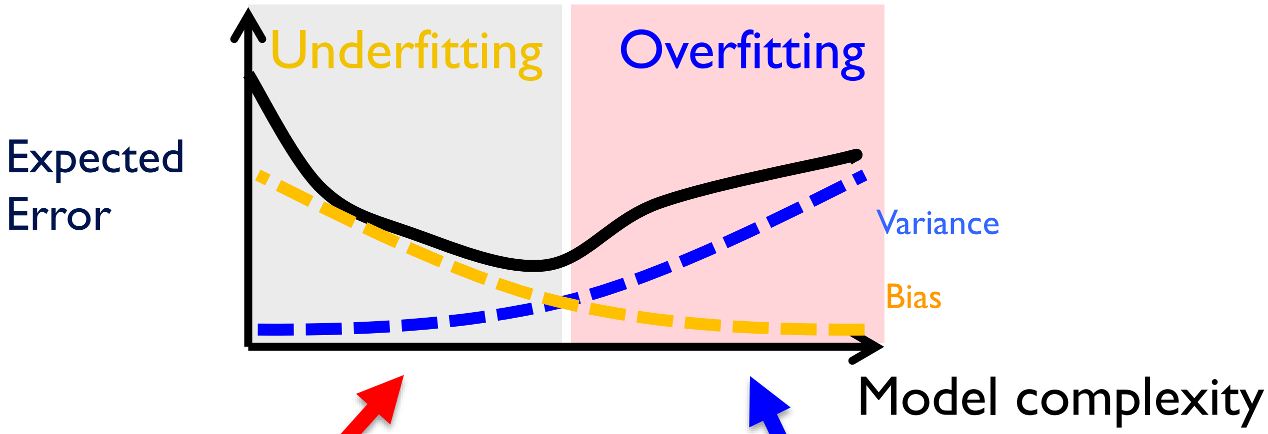
(c) Training data and a better classifier



(d) Testing data and a better classifier

(DEMO)

Underfitting and Overfitting



High Empirical Error

Simple models:
High bias and low variance
Smaller C

Complex models:
High variance and low bias
Larger C

Low Empirical Error

What Do We Optimize?

- L1-loss SVM

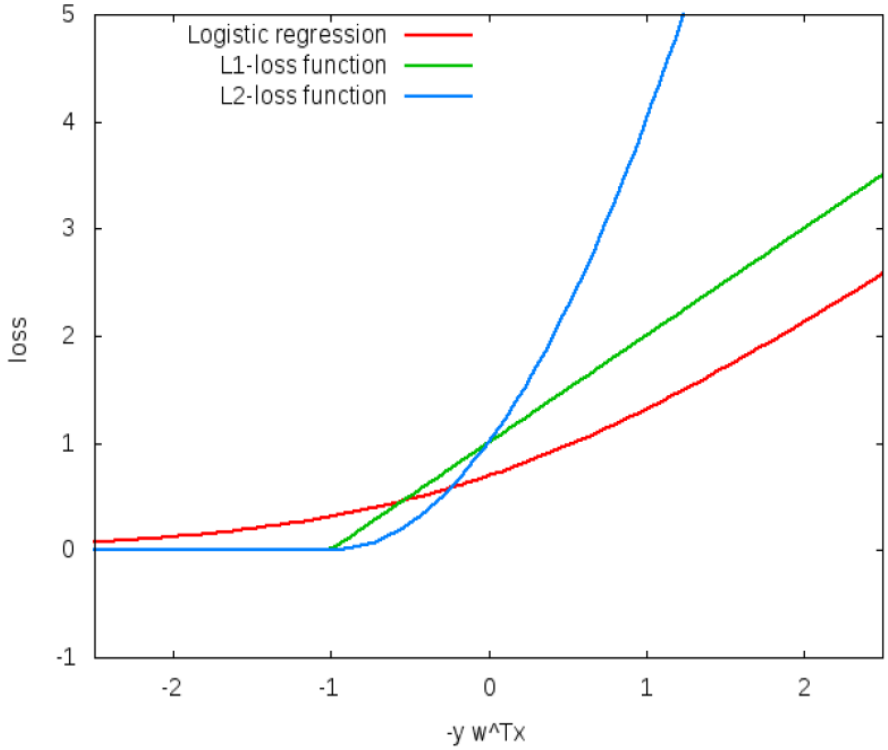
$$\min_{\boldsymbol{w}} \frac{1}{2} \boldsymbol{w}^T \boldsymbol{w} + C \sum_{i=1}^{l} \max(0, 1 - y_i \boldsymbol{w}^T \boldsymbol{x}_i)$$

- L2-loss SVM

$$\min_{\boldsymbol{w}} \frac{1}{2} \boldsymbol{w}^T \boldsymbol{w} + C \sum_{i=1}^{l} \max(0, 1 - y_i \boldsymbol{w}^T \boldsymbol{x}_i)^2$$

What Do We Optimize(2)?

- We get an unconstrained problem. We can use the (stochastic) gradient descent algorithm!
- Many other methods
  - Iterative scaling; non-linear conjugate gradient; quasi-Newton methods; truncated Newton methods; trust-region newton method.
  - All methods are iterative methods, that generate a sequence $\boldsymbol{w}_k$ that converges to the optimal solution of the optimization problem above.
- Currently: Limited memory BFGS is very popular



Optimization: How to Solve

1. Earlier methods used Quadratic Programming. Very slow.
2. The soft SVM problem is an unconstrained optimization problems. It is possible to use the gradient descent algorithm.

- Many options within this category:
  - Iterative scaling; non-linear conjugate gradient; quasi-Newton methods; truncated Newton methods; trust-region newton method.
  - All methods are iterative methods, that generate a sequence $\boldsymbol{w}_k$ that converges to the optimal solution of the optimization problem above.
  - Currently: Limited memory BFGS is very popular

3. 3rd generation algorithms are based on Stochastic Gradient Decent
   - The runtime does not depend on $n = \#(examples)$; advantage when n is very large.
   - Stopping criteria is a problem: method tends to be too aggressive at the beginning and reaches a moderate accuracy quite fast, but it's convergence becomes slow if we are interested in more accurate solutions.
4. Dual Coordinated Descent (& Stochastic Version)

SGD for SVM

- Goal: $\min_{\boldsymbol{w}} f(\boldsymbol{w}) \equiv \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + \frac{C}{m}\sum_i \max(0, 1 - y_i \boldsymbol{w}^T \boldsymbol{x}_i)$ m: data size

m is here for mathematical correctness, it doesn't matter in the view of modeling.

- Compute sub-gradient of $f(\boldsymbol{w})$:

$\nabla f(\boldsymbol{w}) = \boldsymbol{w} - C y_i \boldsymbol{x}_i$ if $1 - y_i \boldsymbol{w}^T \boldsymbol{x}_i \geq 0$; otherwise $\nabla f(\boldsymbol{w}) = \boldsymbol{w}$

1. Initialize $\boldsymbol{w} = \boldsymbol{0} \in \boldsymbol{R}^n$
2. For every example $(\boldsymbol{x}_i, y_i) \in D$

   If $y_i \boldsymbol{w}^T \boldsymbol{x}_i \leq 1$ update the weight vector to

   $\boldsymbol{w} \leftarrow \boldsymbol{w} - \gamma(\boldsymbol{w} - C y_i \boldsymbol{x}_i) = (1-\gamma)\boldsymbol{w} + \gamma C y_i \boldsymbol{x}_i$ (γ - learning rate)

   Otherwise $\boldsymbol{w} \leftarrow (1-\gamma)\boldsymbol{w}$
3. Continue until convergence is achieved

This algorithm should ring a bell…

Convergence can be proved for a slightly complicated version of SGD (e.g, Pegasos)

Nonlinear SVM

- We can map data to a high dimensional space: $\boldsymbol{x} \to \phi(\boldsymbol{x})$ (DEMO)
- Then use Kernel trick: $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \phi(\boldsymbol{x}_i)^T \phi(\boldsymbol{x}_j)$ (DEMO2)

Primal

$$\min_{\boldsymbol{w}} \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C\sum_i \xi_i$$

$$\text{s.t } y_i \boldsymbol{w}^T \phi(\boldsymbol{x}_i) \geq 1 - \xi_i$$

$$\xi_i \geq 0 \ \forall i$$

Dual

$$\min_{\alpha} \frac{1}{2}\alpha^T Q \alpha - e^T \alpha$$

$$\text{s.t } 0 \leq \alpha \leq C \ \ \forall i$$

$$Q_{ij} = y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

Theorem: Let $\boldsymbol{w}^*$ be the minimizer of the primal problem, α^* be the minimizer of the dual problem.
Then $\boldsymbol{w}^* = \sum_i \alpha^* y_i \boldsymbol{x}_i$

Nonlinear SVM

- Tradeoff between training time and accuracy
- Complex model vs. simple model

	Linear (LIBLINEAR)			RBF (LIBSVM)			
Data set	C	Time (s)	Accuracy	C	σ	Time (s)	Accuracy
a9a	32	5.4	84.98	8	0.03125	98.9	85.03
real-sim	1	0.3	97.51	8	0.5	973.7	97.90
ijcnn1	32	1.6	92.21	32	2	26.9	98.69
MNIST38	0.03125	0.1	96.82	2	0.03125	37.6	99.70
covtype	0.0625	1.4	76.35	32	32	54,968.1	96.08
webspam	32	25.5	93.15	8	32	15,571.1	99.20

From:
http://www.csie.ntu.edu.tw/~cjlin/papers/lowpoly_journal.pdf