Administration

- HW4 is due on Saturday 3/11
  - No extensions!
  - We will release solutions on Saturday night, so there is enough time for you to look at it before the exam.
- Midterm exam on Thursday 3/16
  - Closed books; in class; ~4 questions
  - All the material covered before the midterm
  - Practice midterms will be released over the weekend
  - Next Tuesday 3/14: Review

Questions?

- Additional Office hours:
  - 8:30-9:30 Tomorrow (Wednesday)

Projects

Scale of Projects:
25% of the grade

- Projects proposals are due on March 10 2017
- Within a week we will give you an approval to continue with your project along with comments and/or a request to modify/augment/do a different project. There will also be a mechanism for peer comments.

- We encourage team projects – a team can be up to 3 people.

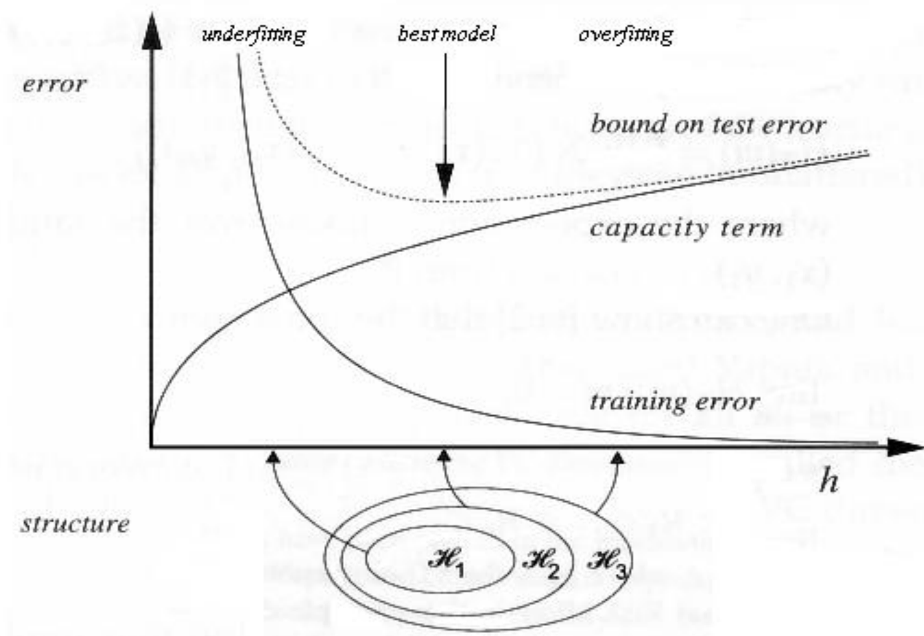
- Please start thinking and working on the project now.
- Your proposal is limited to 1-2 pages, but needs to include references and, ideally, some of the ideas you have developed in the direction of the project (maybe even some preliminary results).
- Any project that has a significant Machine Learning component is good.
- You can do experimental work, theoretical work, a combination of both or a critical survey of results in some specialized topic.
- The work *has* to include some reading. Even if you do not do a survey, you must read (at least) two related papers or book chapters and relate your work to it.
- Originality is not mandatory but is encouraged.
- Try to make it interesting!

Examples

- Fake News Challenge :- http://www.fakenewschallenge.org/
- KDD Cup 2013:
  - "Author-Paper Identification": given an author and a small set of papers, we are asked to identify which papers are really written by the author.
    - https://www.kaggle.com/c/kdd-cup-2013-author-paper-identification-challenge
  - "Author Profiling": given a set of document, profile the author: identification, gender, native language, ….
- Caption Control: Is it gibberish? Spam? High quality text?
  - Adapt an NLP program to a new domain
- Work on making learned hypothesis (e.g., linear threshold functions, NN) more comprehensible
  - Explain the prediction
- Develop a (multi-modal) People Identifier
- Compare Regularization methods: e.g., Winnow vs. L1 Regularization
- Large scale clustering of documents + name the cluster
- Deep Networks: convert a state of the art NLP program to a deep network, efficient, architecture.
- Try to prove something

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.

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

- 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.

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

- 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.

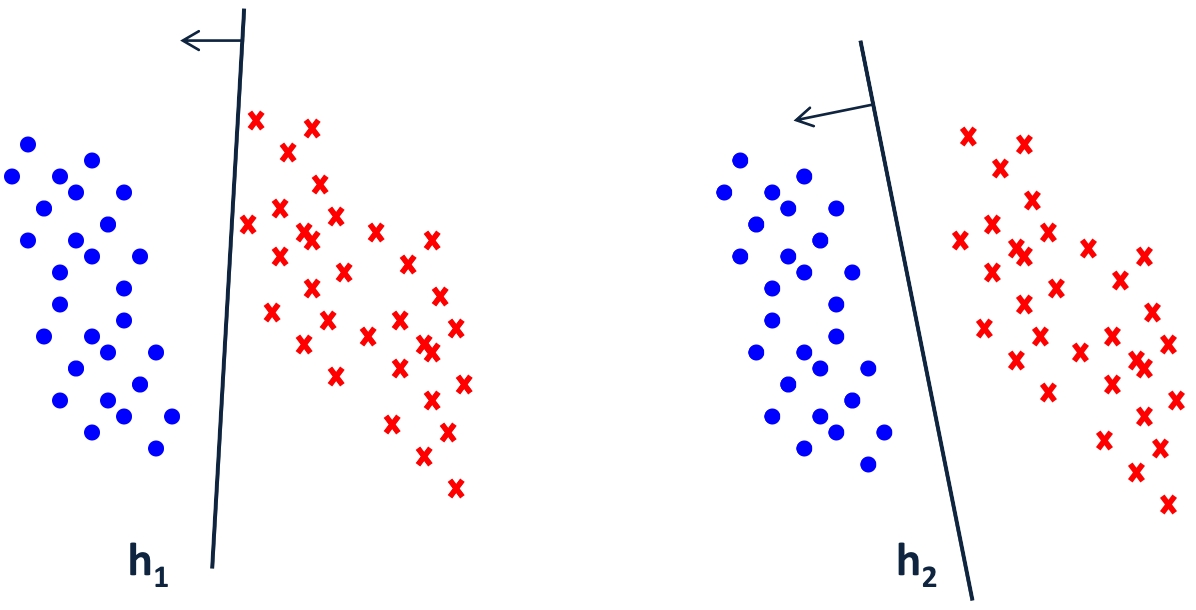
Data Dependent VC dimension

- So far we discussed VC dimension in the context of a fixed 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.

Linear Classification

- Let $X = R^2$, $Y = \{+1, -1\}$
- Which of these classifiers would be likely to generalize better?



VC and Linear Classification

- Recall the VC based generalization bound:

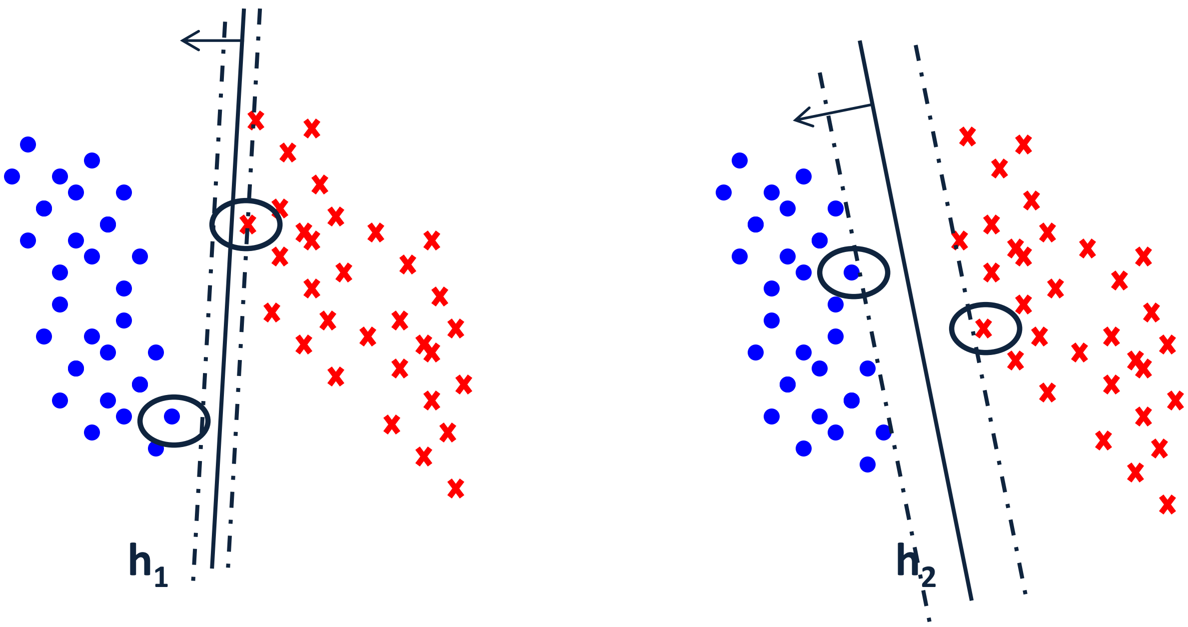
$$Err(h) \leq err_{TR}(h) + Poly\{VC(H), 1/m, \log(1/\delta)\}$$

- Here we get the same bound for both classifier:
- $Err_{TR}(h_1) = Err_{TR}(h_2) = 0$
- $h_1, h_2 \in H_{lin(2)}, VC(H_{lin(2)}) = 3$

- 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 γ_i of a point $x_i \in R^n$ with respect to a linear classifier $h(x) = sign(w \cdot x + b)$ is defined as the distance of x_i from the hyperplane $w \cdot x + b = 0$:
- $\gamma_i = |(w \cdot x_i + b)/||w|||$
- The margin of a set of points $\{x_1,\ldots x_m\}$ with respect to a hyperplane w, is defined as the margin of the point closest to the hyperplane:
- $$\gamma = \min_i \gamma_i = \min_i |(w \cdot x_i + b)/||w|||$$

VC and Linear Classification

- If H_γ is the space of all linear classifiers in $\Re^n$ that separate the training data with margin at least γ, then:

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

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

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

Data Dependent VC dimension

The benefit of VC dimension

- Namely, when we consider the class H_γ of linear hypotheses that separate a given data set with a margin γ,
- We see that
  - Large Margin γ → Small VC dimension of H_γ
- Consequently, our goal could be to find a separating hyperplane w that maximizes the margin of the set S of examples.
- A second observation that drives an algorithmic approach is that:

  Small ||w||→ Large Margin

- This leads to an algorithm: from among all those w's that agree with the data, find the one with the minimal size ||w||

(PS0, PS1): The distance between a point x and the hyperplane defined by (w; b) is: $|w^T x + b|/||w||$

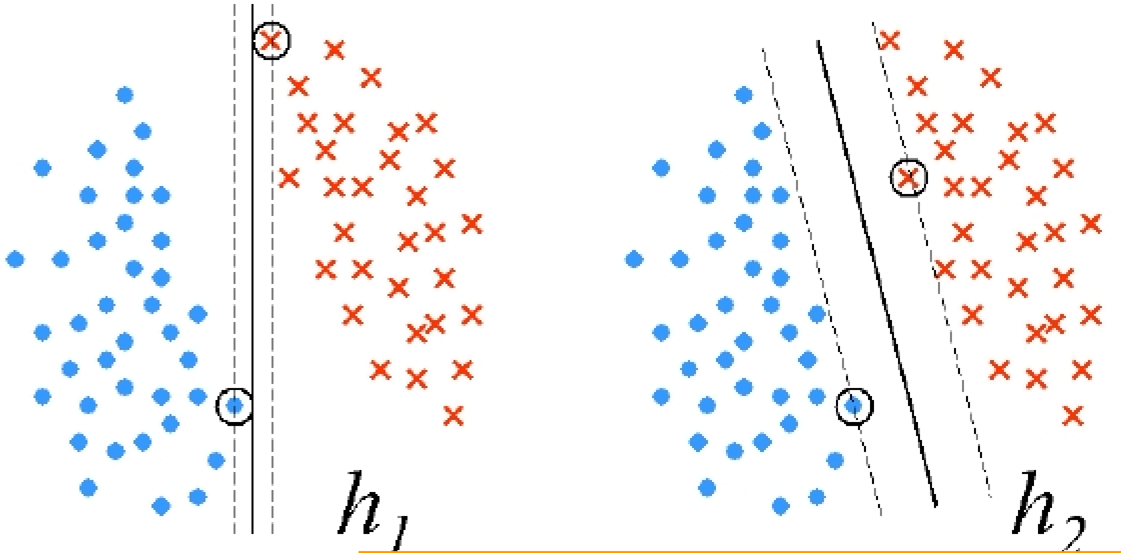
Maximal Margin

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

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

1. For a given w: Find the closest point.
2. Then, find the one the gives the maximal margin value across all w's (of size 1).

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.



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

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

Margin and VC dimension

Believe

- Theorem (Vapnik): If H_γ is the space of all linear classifiers in $\Re^n$ that separate the training data with margin at least γ, then

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

- where R is the radius of the smallest sphere (in $\Re^n$) that contains the data.
- This is the first observation that will lead to an algorithmic approach.

We'll Prove

- The second observation is that:

Small $||w||$ → Large Margin

- Consequently: the algorithm will be: from among all those w's that agree with the data, find the one with the minimal size $||w||$

Hard SVM

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

1. For a given w: Find the closest point.
2. Among all w's (of size 1) find the w the maximizes this point's margin.

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

- How to find this w^*?

- Claim: Define w_0 to be the solution of the optimization problem:

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

  Then:

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

1. Consider the set of "good" w's (those that separate the data).
2. Among those, choose the one with minimal size.

That is, the normalization of w_0 corresponds to the largest margin separating hyperplane.

Hard SVM (2)

- Claim: Define w_0 to be the solution of the optimization problem:

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

  Then:

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

That is, the normalization of w_0 corresponds to the largest margin separating hyperplane.

- Proof: Define $w' = w_0/||w_0||$ and let w^* be the largest-margin separating hyperplane of size 1. We need to show that $w' = w^*$.

  Def. of w_0

  Note first that $w^*/\gamma(S)$ satisfies the constraints in (**);

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

- Consequently:

  Def. of w' Def. of w_0 Prev. ineq.

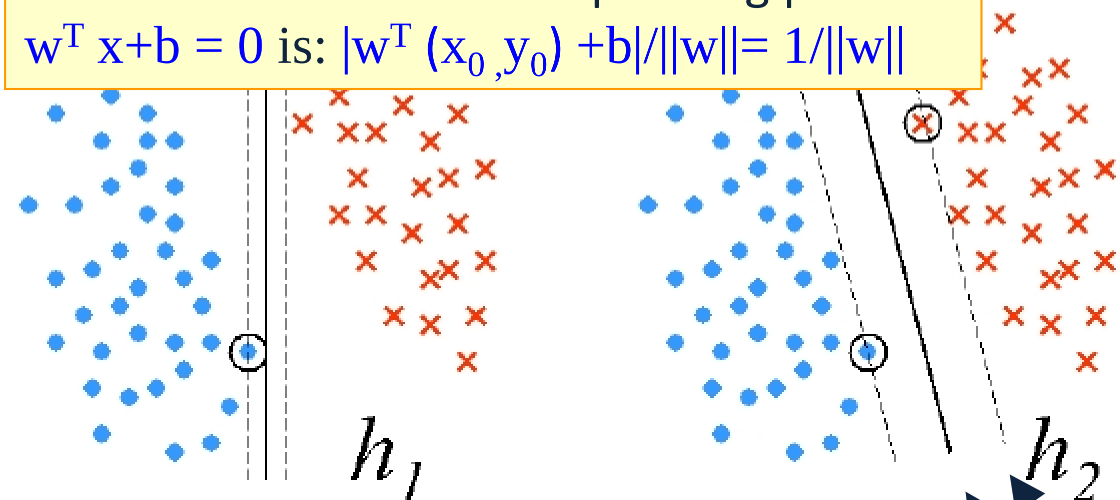
  $\forall (x,y) \in S \quad y\, w'^T x = 1/||w_0||\, y\, w_0^T x \geq 1/||w_0|| \geq \gamma(S)$

But since $||w'|| = 1$ this implies that w' corresponds to the largest margin, that is $w' = w^*$

Margin of a Separating Hyperplane

- A separating hyperplane: $w^T x+b = 0$

Assumption: data is linearly separable
Let (x_0, y_0) be a point on $w^Tx+b = 1$
Then its distance to the separating plane
$w^T x+b = 0$ is: $|w^T (x_0, y_0) +b|/||w||= 1/||w||$



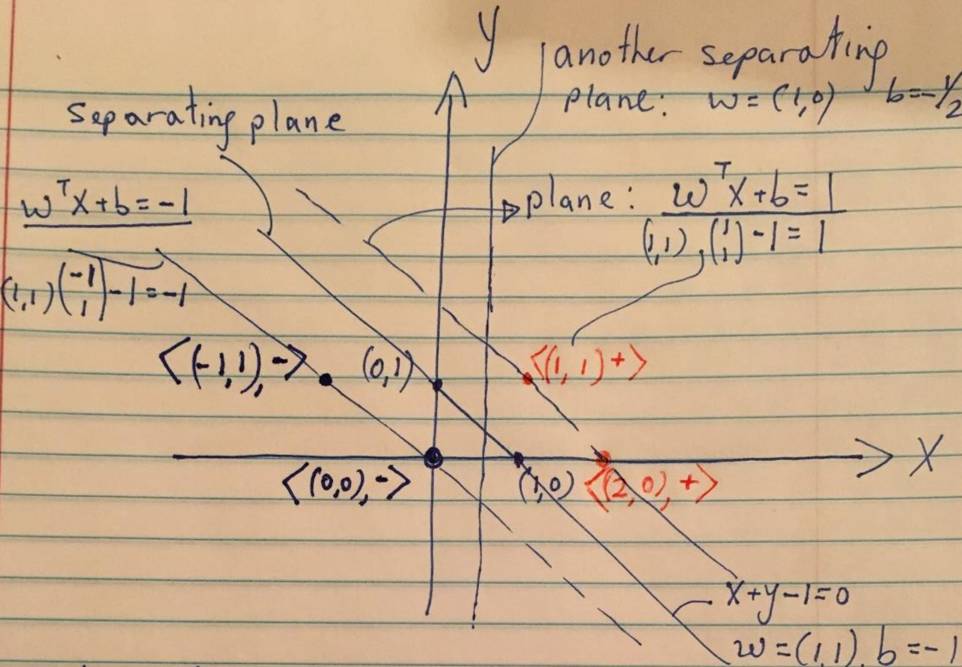
Distance between
$w^T x+b = +1$ and -1 is $2 / ||w||$
What we did:

1. Consider all possible w with different angles
2. Scale w such that the constraints are tight
3. Pick the one with largest margin/minimal size

$$w^T x_i + b \geq 1 \quad \text{if} \quad y_i = 1$$
$$w^T x_i + b \leq -1 \quad \text{if} \quad y_i = -1$$

$$=> y_i(w^T x_i + b) \geq 1$$

$w^T x+b = 0$
$w^T x+b = -1$



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

is : $\dfrac{(1,1)\begin{pmatrix}1\\1\end{pmatrix}-1}{\sqrt{2}} = \dfrac{1}{\sqrt{2}} = \dfrac{\sqrt{2}}{2} = \dfrac{1}{\|w\|}$ ~~Not 1, but we can make it 1~~

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 $w^T x + b = 2$

$(2,2)\begin{pmatrix}1\\1\end{pmatrix}-2=2$

⊖ plane would be $(2,2)\begin{pmatrix}-1\\1\end{pmatrix}-2=-2$

$w^T x + b = -2$

For the second plane $w=(1,0),\ b=-\frac{1}{2}$:

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

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

$(c,0)\begin{pmatrix}1\\1\end{pmatrix}-\frac{c}{2}=1$ ⟸ That's what we wa

$\Rightarrow\quad c-\frac{c}{2}=1 \quad \underline{\underline{c=2}}$

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

Now:

$+:\ (2,0)\begin{pmatrix}1\\1\end{pmatrix}-1=1$

$+:\ (2,0)\begin{pmatrix}2\\0\end{pmatrix}-1=3$

$-:\ (2,0)\begin{pmatrix}-1\\1\end{pmatrix}-1=-3$

$-:\ (2,0)\begin{pmatrix}0\\0\end{pmatrix}-1=-1$

Good!

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

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

Hard SVM Optimization

- We have shown that the sought after weight vector w is the solution of the following optimization problem:

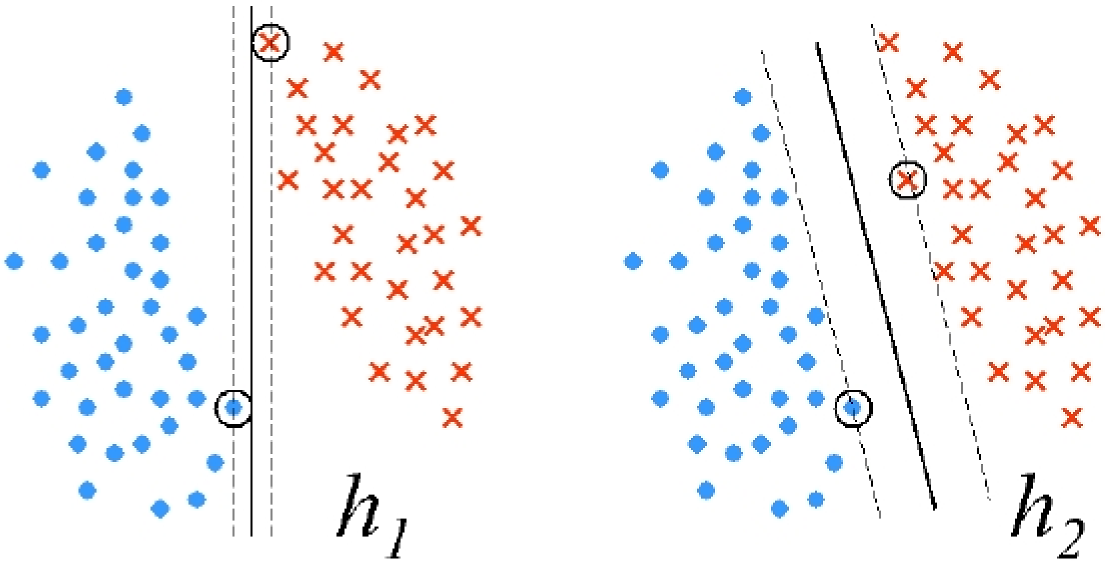
SVM Optimization: (***)

- Minimize: $\frac{1}{2} ||w||^2$

  Subject to: $\forall$ (x,y) $\in$ S: $y\, w^T x \geq 1$

- 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 $w^T x + b = 0$ is $2 / ||w||$

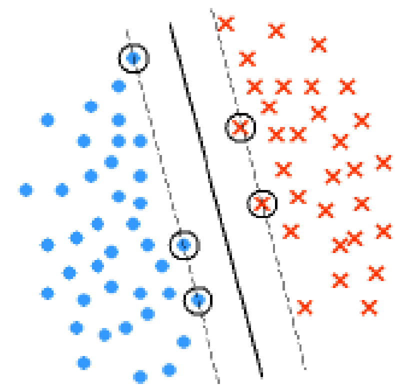
max $2 / ||w||$ = min $||w||$ = min $\frac{1}{2} w^T w$

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

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

Support Vector Machines

- 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.



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

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

This representation should ring a bell…

(recap) Kernel Perceptron

$$\textbf{Examples}: \mathbf{x} \in \{0,1\}^n; \quad \textbf{Nonlinear mapping}: \mathbf{x} \rightarrow \mathbf{t(x)}, \mathbf{t(x)} \in \mathbf{R}^{n'}$$

$$\textbf{Hypothesis}: \mathbf{w} \in \mathbf{R}^{n'}; \textbf{Decision function}: \mathbf{f(x)} = \mathbf{sgn}(\sum_{i=1}^{n'} \mathbf{w}_i \mathbf{t(x)}_i) = \mathbf{sgn}(\mathbf{w} \bullet \mathbf{t(x)})$$

$$\boxed{\textbf{If } \mathbf{f(x^{(k)})} \neq \mathbf{y^{(k)}}, \quad \mathbf{w} \leftarrow \mathbf{w} + \mathbf{r}\, \mathbf{y^{(k)}} t(\mathbf{x^{(k)}})}$$

- If n’ is large, we cannot represent **w** explicitly. However, the weight vector **w** can be written as a linear combination of examples:

$$\boxed{\mathbf{w} = \sum_{j=1}^{m} \mathbf{r}\alpha_j \mathbf{y^{(j)}} \mathbf{t(x^{(j)})}}$$

- Where α_j is the number of mistakes made on $x^{(j)}$
- Then we can compute f(x) based on $\{x^{(j)}\}$ and $\boldsymbol{\alpha}$

$$\boxed{\mathbf{f(x)} = \mathbf{sgn}(\mathbf{w} \bullet \mathbf{t(x)}) = \mathbf{sgn}(\sum_{j=1}^{m} \mathbf{r}\alpha_j \mathbf{y^{(j)}} \mathbf{t(x^{(j)})} \bullet \mathbf{t(x)}) = \mathbf{sgn}(\sum_{j=1}^{m} \mathbf{r}\alpha_j \mathbf{y^{(j)}} K(\mathbf{x^{(j)}}, \mathbf{x}))}$$

(recap) Kernel Perceptron

$$\textbf{Examples}: \mathbf{x} \in \{0,1\}^n; \quad \textbf{Nonlinear mapping}: \mathbf{x} \rightarrow \mathbf{t(x)}, \mathbf{t(x)} \in \mathbf{R}^{n'}$$

$$\textbf{Hypothesis}: \mathbf{w} \in \mathbf{R}^{n'}; \textbf{Decision function}: \mathbf{f(x)} = \mathbf{sgn(w \bullet t(x))}$$

- In the training phase, we initialize $\boldsymbol{\alpha}$ to be an all-zeros vector.
- For training sample $(x^{(k)}, y^{(k)})$, instead of using the original Perceptron update rule in the $R^{n'}$ space

$$\boxed{\textbf{If } \mathbf{f(x^{(k)})} \neq \mathbf{y^{(k)}}, \quad \mathbf{w} \leftarrow \mathbf{w} + \mathbf{r}\, \mathbf{y^{(k)}} t(\mathbf{x^{(k)}})}$$

we maintain $\boldsymbol{\alpha}$ by

$$\boxed{\textbf{if } \mathbf{f(x^{(k)})} = \mathbf{sgn}\Big(\sum_{j=1}^{m} \mathbf{r}\,\alpha_j \mathbf{y^{(j)}} K(\mathbf{x^{(j)}}, \mathbf{x^{(k)}})\Big) \neq \mathbf{y^{(k)}} \quad \textbf{then } \alpha_k \leftarrow \alpha_k + 1}$$

based on the relationship between w and $\boldsymbol{\alpha}$:

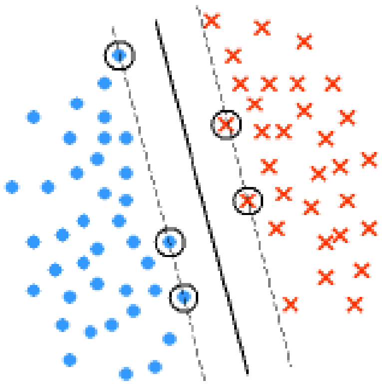
$$\boxed{\mathbf{w} = \sum_{j=1}^{m} \mathbf{r}\,\alpha_j \mathbf{y^{(j)}} \mathbf{t(x^{(j)})}}$$

Duality

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

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

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



Footnote about the threshold

- Similar to Perceptron, we can augment vectors to handle the bias term

  $\bar{x} \Leftarrow (x\ ,\ 1);\ \ \bar{w} \Leftarrow (w\ ,\ b)$ so that $\bar{w}^T \bar{x} = w^T x + b$
- Then consider the following formulation

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

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

$$\min_{w,b} \ \frac{1}{2}w^Tw + \underbrace{\frac{1}{2}b^2} \quad \text{s.t} \ \ \mathrm{y_i}(w^{\mathrm{T}}\mathrm{x_i} + b) \geq 1, \forall (x_i, y_i) \in \mathrm{S}$$

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

For simplicity, we ignore the bias term

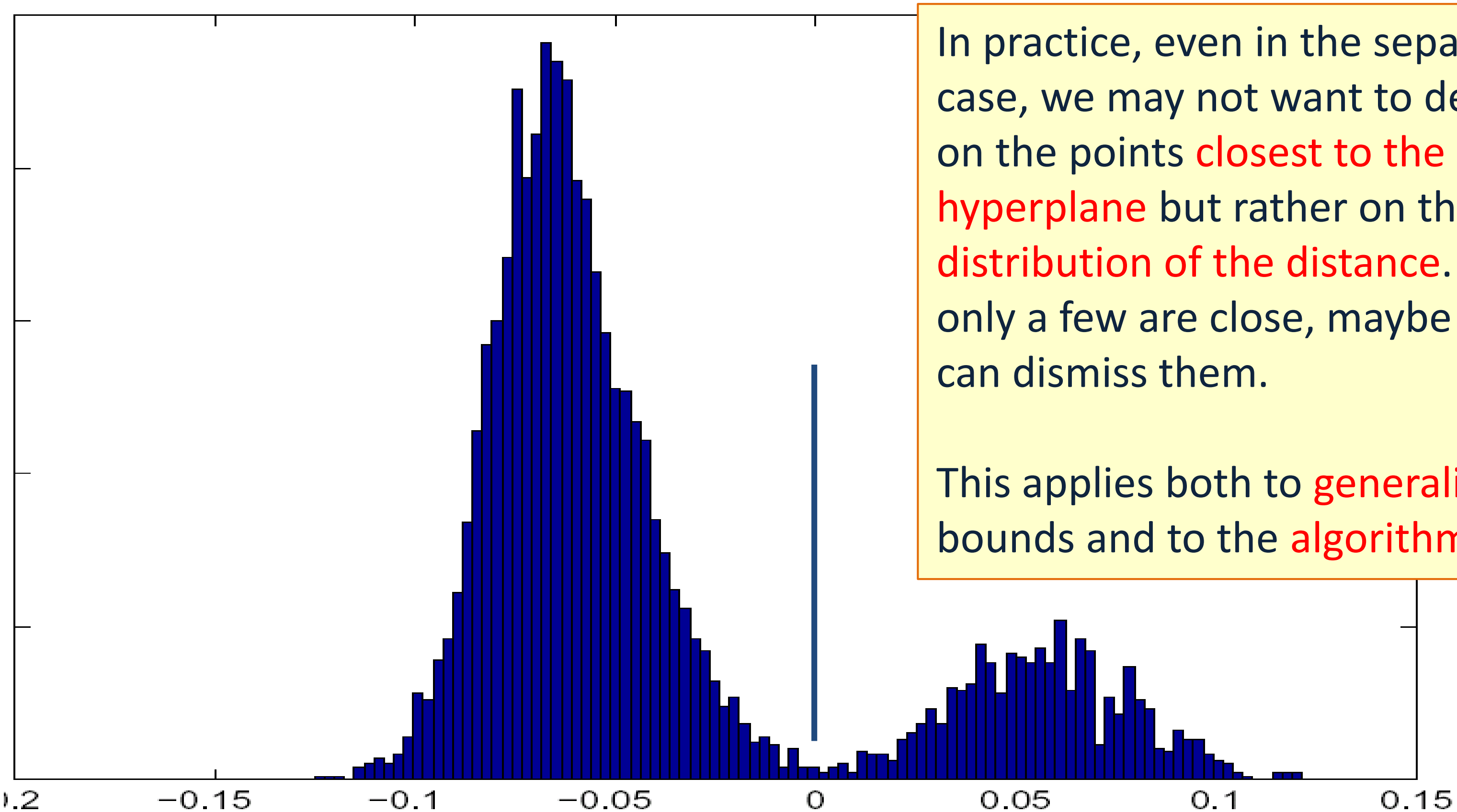
Key Issues

- 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.

- 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.

This applies both to generalization bounds and to the algorithm.

Soft SVM

- Notice that the relaxation of the constraint:

$$y_i w^T x_i \geq 1$$

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

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

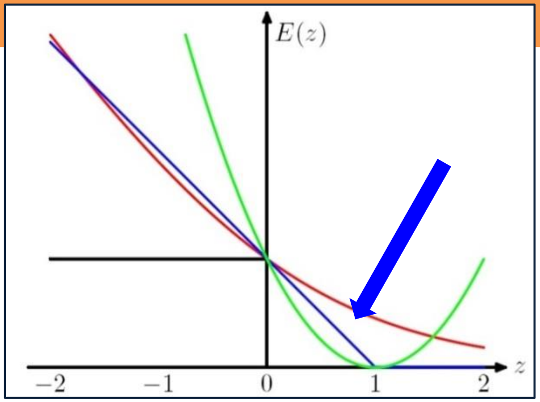
- Now, we want to solve:

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

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

A large value of C means that misclassifications are bad - resulting in smaller margins and less training error (but more expected true error). A small C results in more training error, hopefully better true error.

Soft SVM (2)



- Now, we want to solve:

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

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

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

- Which can be written as:

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

- What is the interpretation of this?

Soft SVM (3)

- 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, move to a surrogate loss function that is convex.
- SVM relies on the hinge loss function (note that the dual formulation can give some intuition for that too).

$$\mathrm{Min}_w \tfrac{1}{2} ||w||^2 + C \sum_{(x,y) \in S} \max(0, 1 - y\, w^T x)$$

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

SVM Objective Function

- The problem we solved is:

$$\text{Min } \tfrac{1}{2} ||w||^2 + c \sum \xi_i$$

- Where $\xi_i > 0$ is called a slack variable, and is defined by:
  - $\xi_i = \max(0, 1 - y_i w^t x_i)$
  - Equivalently, we can say that: $y_i w^t x_i \geq 1 - \xi; \quad \xi \geq 0$
- And this can be written as:

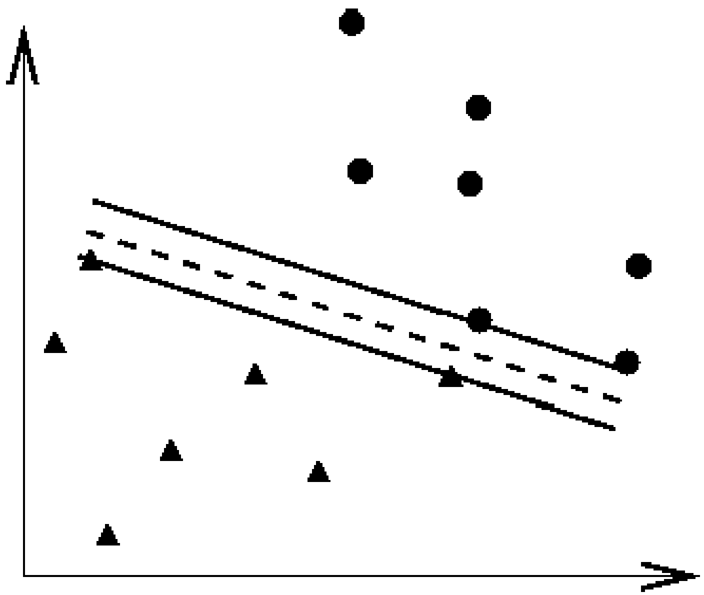
$$\text{Min } \underbrace{\tfrac{1}{2} ||w||^2}_{\text{Regularization term}} + \underbrace{c \sum \xi_i}_{\text{Empirical loss}}$$

Can be replaced by other **regularization functions**

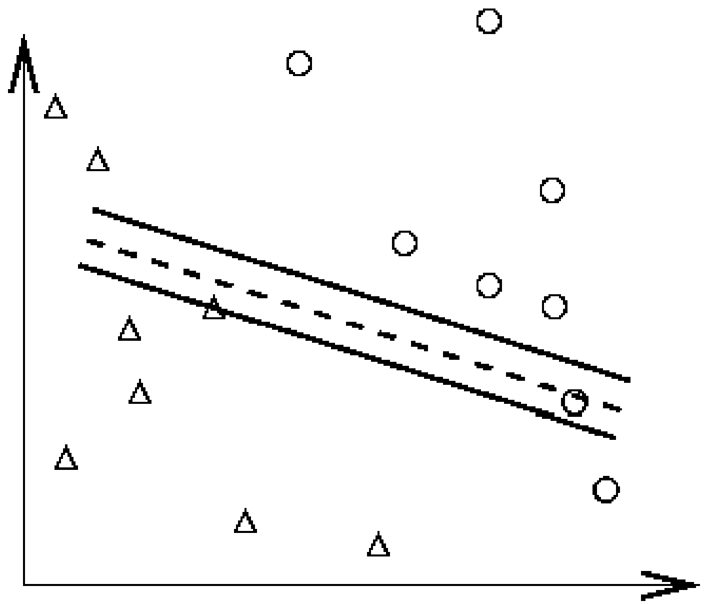
Can be replaced by other **loss functions**

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

Balance between regularization and empirical loss

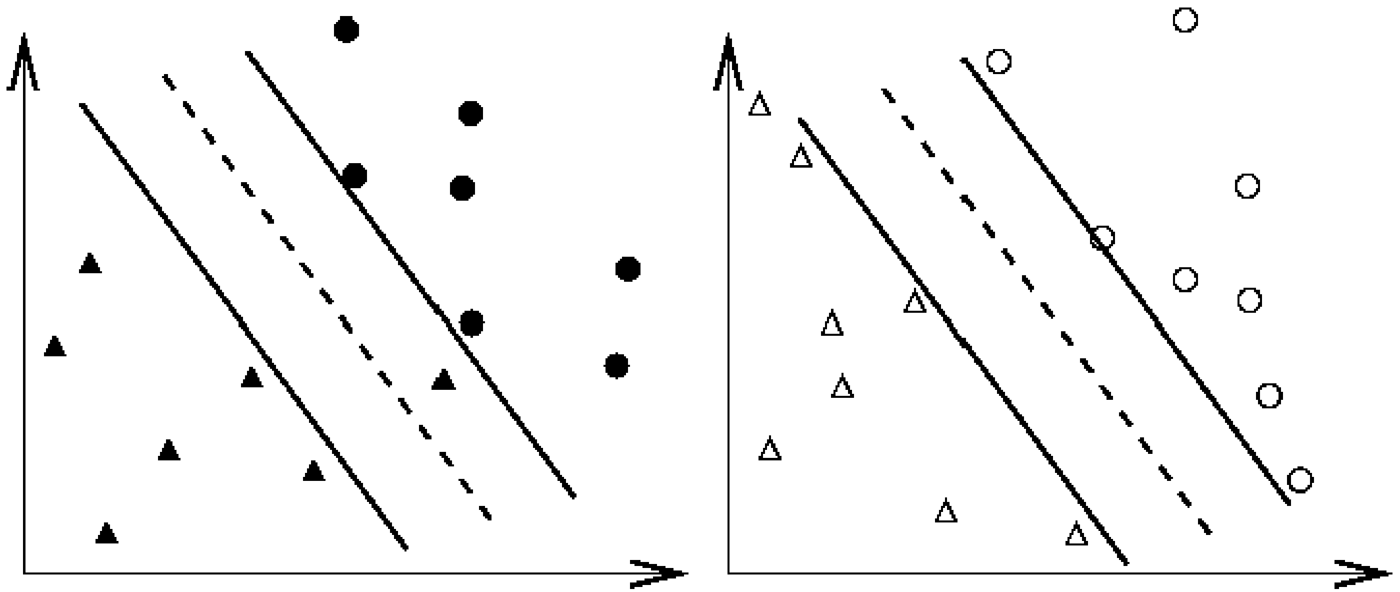


(a) Training data and an over-fitting classifier



(b) Testing data and an over-fitting classifier

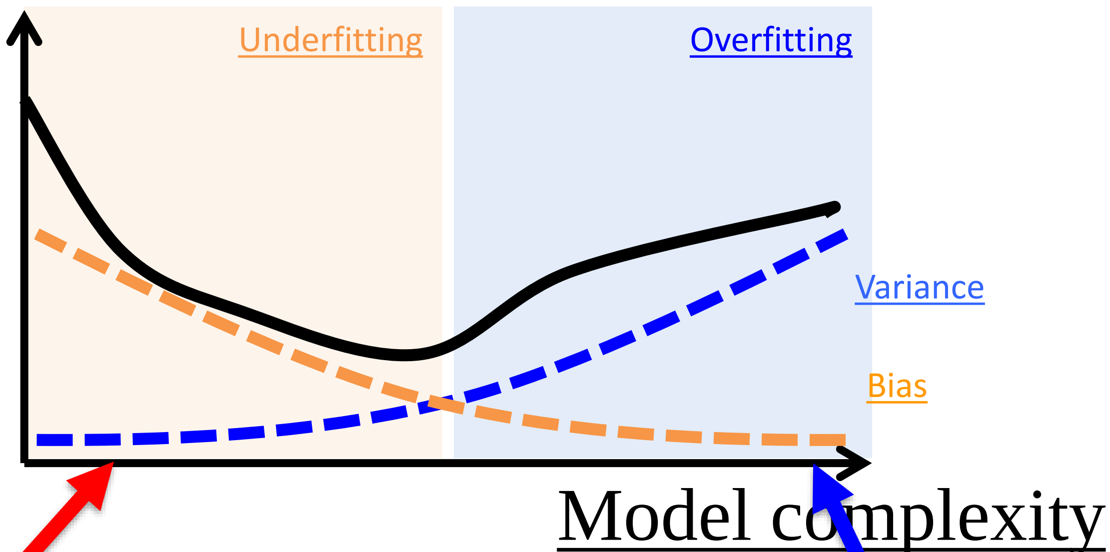
Balance between regularization and empirical loss



(c) Training data and a better classifier (d) Testing data and a better classifier

[(DEMO)]

Underfitting and Overfitting



- **Simple models:**
  High bias and low variance
- **Smaller C**

Complex models:
High variance and low bias

Larger C

What Do We Optimize?

- Logistic Regression

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

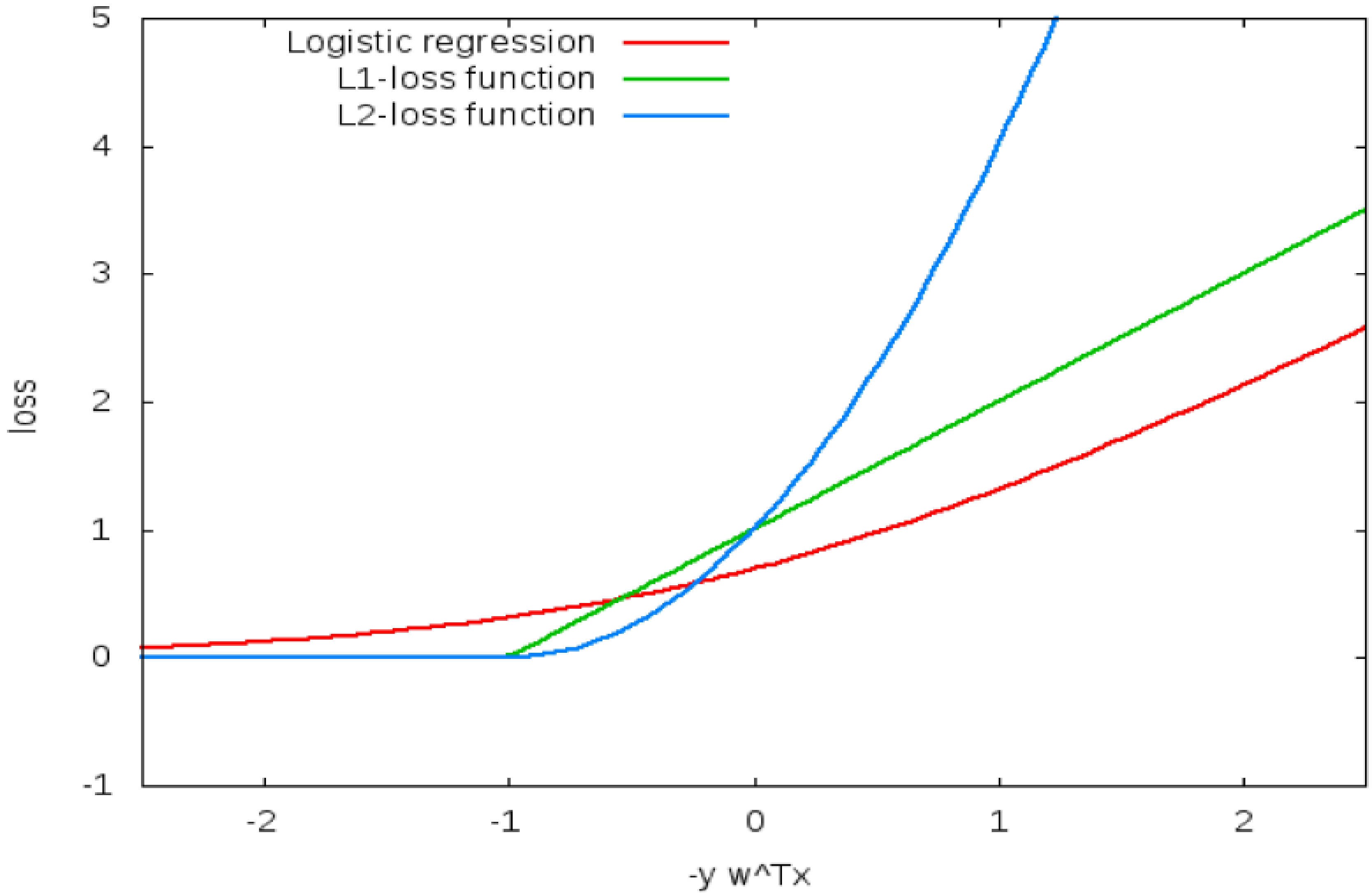
- L1-loss SVM

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

- L2-loss SVM

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

What Do We Optimize(2)?



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! Still, it is quite slow.
- 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 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: $\min_{w} f(w) \equiv \frac{1}{2} w^T w + \frac{C}{m} \sum_i \max(0, 1 - y_i w^T x_i)$.   m: data size

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

- Compute sub-gradient of $f(w)$:

$\nabla f(w) = w - C y_i x_i$ if $1 - y_i w^T x_i \geq 0$; otherwise $\nabla f(w) = w$

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

   If $y_i w^T x_i \leq 1$ update the weight vector to

   $w \leftarrow (1 - \gamma) w + \gamma C y_i x_i$ (γ - learning rate)

   Otherwise $w \leftarrow (1 - \gamma) w$
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: $\mathrm{x} \to \phi(x)$ (DEMO)
- Then use Kernel trick: $K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$ (DEMO2)

Primal:

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

$$\text{s.t} \quad \mathrm{y_i w^T} \phi(x_i) \geq 1 - \xi_i$$

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

Dual:

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

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

$$\mathrm{Q}_{ij} = y_i\, y_j K(x_i, x_j)$$

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

Nonlinear SVM

- Tradeoff between training time and accuracy
- Complex model v.s. 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