

SVMs, Duality and the Kernel Trick

Machine Learning – 10701/15781
 Carlos Guestrin
 Carnegie Mellon University

October 22nd, 2007

©2005-2007 Carlos Guestrin

1

Lagrange multipliers – Dual variables

Solving: $\min_x \max_{\alpha} x^2 - \alpha(x - b)$
 s.t. $\alpha \geq 0$

$\frac{\partial L}{\partial x} = 2x - \alpha = 0 \Rightarrow \alpha = 2x$

$\frac{\partial L}{\partial \alpha} = -(x - b)$

$x = 1 \Rightarrow \frac{\partial L}{\partial \alpha} = 0$
 $\alpha = 2 > 0$
 constraint relevant

if I set $x = -1$
 \downarrow
 $\frac{\partial L}{\partial \alpha} = 0$
 but $\alpha = -2 < 0$

$x = 0$ no way I can set $\frac{\partial L}{\partial \alpha} = 0$

\Rightarrow ignore $\alpha = 0$
 constraint irrelevant

©2005-2007 Carlos Guestrin

2

Dual SVM formulation – the non-separable case

$$\text{maximize}_{\alpha} \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j$$

α_i can't be too big
α_i very large "when you really constraint problem"
α_i - α_j target

$$\sum_i \alpha_i y_i = 0$$

$$C \geq \alpha_i \geq 0$$

only difference

derivation of dual for problem with slack penalty similar

$$\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$$

$$b = y_k - \mathbf{w} \cdot \mathbf{x}_k$$

for any k where $C > \alpha_k > 0$

©2005-2007 Carlos Guestrin

3

Why did we learn about the dual SVM?

- There are some quadratic programming algorithms that can solve the dual faster than the primal
- But, more importantly, the “kernel trick”!!!
 - Another little detour...

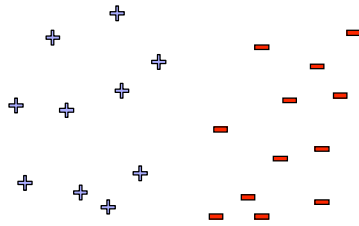
©2005-2007 Carlos Guestrin

4

Reminder from last time: What if the data is not linearly separable?

Use features of features of features of features....

$$\Phi(\mathbf{x}) : \mathbb{R}^m \mapsto F$$



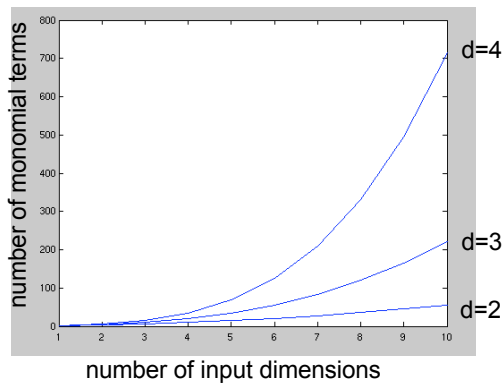
Feature space can get really large really quickly!

©2005-2007 Carlos Guestrin

Higher order polynomials

$$\text{num. terms} = \binom{d + m - 1}{d} = \frac{(d + m - 1)!}{d!(m - 1)!}$$

m – input features
d – degree of polynomial



grows fast!
d = 6, m = 100
about 1.6 billion terms

©2005-2007 Carlos Guestrin

6

Dual formulation only depends on dot-products, not on \mathbf{w} !

$$\begin{aligned} \text{maximize}_{\alpha} \quad & \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j \\ & \sum_i \alpha_i y_i = 0 \\ & C \geq \alpha_i \geq 0 \end{aligned}$$

$$\begin{aligned} \text{maximize}_{\alpha} \quad & \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \\ & K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j) \\ & \sum_i \alpha_i y_i = 0 \\ & C \geq \alpha_i \geq 0 \end{aligned}$$

©2005-2007 Carlos Guestrin

7

Dot-product of polynomials

$$\Phi(\mathbf{u}) \cdot \Phi(\mathbf{v}) = \text{polynomials of degree } d$$

©2005-2007 Carlos Guestrin

8

Finally: the “kernel trick”!

$$\text{maximize}_{\alpha} \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

$$K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$$

$$\sum_i \alpha_i y_i = 0$$

$$C \geq \alpha_i \geq 0$$

$$\mathbf{w} = \sum_i \alpha_i y_i \Phi(\mathbf{x}_i)$$

$$b = y_k - \mathbf{w} \cdot \Phi(\mathbf{x}_k)$$

for any k where $C > \alpha_k > 0$

- Never represent features explicitly
 - Compute dot products in closed form
- Constant-time high-dimensional dot-products for many classes of features
- Very interesting theory – Reproducing Kernel Hilbert Spaces
 - Not covered in detail in 10701/15781, more in 10702

©2005-2007 Carlos Guestrin

9

Polynomial kernels

- All monomials of degree d in $O(d)$ operations:
 $\Phi(\mathbf{u}) \cdot \Phi(\mathbf{v}) = (\mathbf{u} \cdot \mathbf{v})^d = \text{polynomials of degree } d$
- How about all monomials of degree up to d ?
 - Solution 0:
 - Better solution:

©2005-2007 Carlos Guestrin

10

Common kernels

- Polynomials of degree d $K(\mathbf{u}, \mathbf{v}) = (\mathbf{u} \cdot \mathbf{v})^d$
- Polynomials of degree up to d $K(\mathbf{u}, \mathbf{v}) = (\mathbf{u} \cdot \mathbf{v} + 1)^d$
- Gaussian kernels $K(\mathbf{u}, \mathbf{v}) = \exp\left(-\frac{\|\mathbf{u} - \mathbf{v}\|^2}{2\sigma^2}\right)$
- Sigmoid $K(\mathbf{u}, \mathbf{v}) = \tanh(\eta\mathbf{u} \cdot \mathbf{v} + \nu)$

Overfitting?

- Huge feature space with kernels, what about overfitting???
 - Maximizing margin leads to sparse set of support vectors
 - Some interesting theory says that SVMs search for simple hypothesis with large margin
 - Often robust to overfitting

What about at classification time

- For a new input \mathbf{x} , if we need to represent $\Phi(\mathbf{x})$, we are in trouble!
- Recall classifier: $\text{sign}(\mathbf{w} \cdot \Phi(\mathbf{x}) + b)$
- Using kernels we are cool!

$$K(\mathbf{u}, \mathbf{v}) = \Phi(\mathbf{u}) \cdot \Phi(\mathbf{v})$$

$$\mathbf{w} = \sum_i \alpha_i y_i \Phi(\mathbf{x}_i)$$

$$b = y_k - \mathbf{w} \cdot \Phi(\mathbf{x}_k)$$

for any k where $C > \alpha_k > 0$

©2005-2007 Carlos Guestrin

13

SVMs with kernels

- Choose a set of features and kernel function
- Solve dual problem to obtain support vectors α_i
- At classification time, compute:

$$\mathbf{w} \cdot \Phi(\mathbf{x}) = \sum_i \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i)$$

$$b = y_k - \sum_i \alpha_i y_i K(\mathbf{x}_k, \mathbf{x}_i)$$

for any k where $C > \alpha_k > 0$

Classify as

$$\text{sign}(\mathbf{w} \cdot \Phi(\mathbf{x}) + b)$$

©2005-2007 Carlos Guestrin

14

Remember kernel regression

Remember kernel regression???

1. $w_i = \exp(-D(x_i, query) / K_w^2)$
2. How to fit with the local points?

Predict the weighted average of the outputs:

$$\text{predict} = \Sigma w_i y_i / \Sigma w_i$$

SVMs v. Kernel Regression

SVMs

$$\text{sign}(w \cdot \Phi(x) + b)$$

or

$$\text{sign}\left(\sum_i \alpha_i y_i K(x, x_i) + b\right)$$

Kernel Regression

$$\text{sign}\left(\frac{\sum_i y_i K(x, x_i)}{\sum_j K(x, x_j)}\right)$$

SVMs v. Kernel Regression

SVMs

$$\text{sign}(\mathbf{w} \cdot \Phi(\mathbf{x}) + b)$$

or

sign

Kernel Regression

$$\text{sign}\left(\frac{\sum_i y_i K(\mathbf{x}, \mathbf{x}_i)}{\sum_j K(\mathbf{x}, \mathbf{x}_j)}\right)$$

Differences:

- SVMs:
 - Learn weights α_i (and bandwidth)
 - Often sparse solution
- KR:
 - Fixed “weights”, learn bandwidth
 - Solution may not be sparse
 - Much simpler to implement

What's the difference between SVMs and Logistic Regression?

	SVMs	Logistic Regression
Loss function		
High dimensional features with kernels		

Kernels in logistic regression

$$P(Y = 1 | x, \mathbf{w}) = \frac{1}{1 + e^{-(\mathbf{w} \cdot \Phi(x) + b)}}$$

- Define weights in terms of support vectors:

$$\mathbf{w} = \sum_i \alpha_i \Phi(\mathbf{x}_i)$$

$$\begin{aligned} P(Y = 1 | x, \mathbf{w}) &= \frac{1}{1 + e^{-(\sum_i \alpha_i \Phi(\mathbf{x}_i) \cdot \Phi(x) + b)}} \\ &= \frac{1}{1 + e^{-(\sum_i \alpha_i K(x, \mathbf{x}_i) + b)}} \end{aligned}$$

- Derive simple gradient descent rule on α_i

©2005-2007 Carlos Guestrin

19

What's the difference between SVMs and Logistic Regression? (Revisited)

	SVMs	Logistic Regression
Loss function	Hinge loss	Log-loss
High dimensional features with kernels	Yes!	Yes!

©2005-2007 Carlos Guestrin

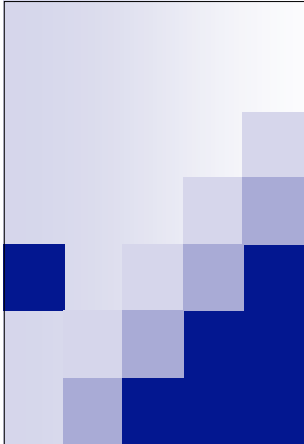
20

What you need to know

- Dual SVM formulation
 - How it's derived
- The kernel trick
- Derive polynomial kernel
- Common kernels
- Kernelized logistic regression
- Differences between SVMs and logistic regression

Announcements

- Midterm:
 - Thursday Oct. 25th, Thursday 5-6:30pm, MM A14
 - All content up to, and including SVMs and Kernels
 - Not learning theory
- Midterm review:
 - Tuesday, 5-6:30pm, location TBD
 - You should read midterms for Spring 2006 and 2007 **before** the review session
 - Then, you can ask about some of the questions in these midterms



PAC-learning, VC Dimension and Margin-based Bounds

Machine Learning – 10701/15781
Carlos Guestrin
Carnegie Mellon University

October 22nd, 2007

©2005-2007 Carlos Guestrin

23

What now...



- We have explored **many** ways of learning from data
- But...
 - How good is our classifier, really?
 - How much data do I need to make it “good enough”?

©2005-2007 Carlos Guestrin

24

A simple setting...

- Classification
 - m data points
 - **Finite** number of possible hypothesis (e.g., dec. trees of depth d)
- A learner finds a hypothesis h that is **consistent** with training data
 - Gets zero error in training – $\text{error}_{\text{train}}(h) = 0$
- What is the probability that h has more than ϵ true error?
 - $\text{error}_{\text{true}}(h) \geq \epsilon$

©2005-2007 Carlos Guestrin

25

How likely is a bad hypothesis to get m data points right?

- Hypothesis h that is **consistent** with training data → got m i.i.d. points right
 - h “bad” if it gets all this data right, but has high true error
- Prob. h with $\text{error}_{\text{true}}(h) \geq \epsilon$ gets one data point right

- Prob. h with $\text{error}_{\text{true}}(h) \geq \epsilon$ gets m data points right

©2005-2007 Carlos Guestrin

26

But there are many possible hypothesis that are consistent with training data

How likely is learner to pick a bad hypothesis

- Prob. h with $\text{error}_{\text{true}}(h) \geq \epsilon$ gets m data points right

- There are k hypothesis consistent with data
 - How likely is learner to pick a bad one?

Union bound

- $P(A \text{ or } B \text{ or } C \text{ or } D \text{ or } \dots)$

How likely is learner to pick a bad hypothesis

- Prob. h with $\text{error}_{\text{true}}(h) \geq \epsilon$ gets m data points right
- There are k hypothesis consistent with data
 - How likely is learner to pick a bad one?

Review: Generalization error in finite hypothesis spaces [Haussler '88]

- **Theorem:** Hypothesis space H finite, dataset D with m i.i.d. samples, $0 < \epsilon < 1$: for any learned hypothesis h that is consistent on the training data:

$$P(\text{error}_{\text{true}}(h) > \epsilon) \leq |H|e^{-m\epsilon}$$

Using a PAC bound

- Typically, 2 use cases: $P(\text{error}_{\text{true}}(h) > \epsilon) \leq |H|e^{-m\epsilon}$
 - 1: Pick ϵ and δ , give you m
 - 2: Pick m and δ , give you ϵ

Review: Generalization error in finite hypothesis spaces [Haussler '88]

- **Theorem:** Hypothesis space H finite, dataset D with m i.i.d. samples, $0 < \epsilon < 1$: for any learned hypothesis h that is consistent on the training data:

$$P(\text{error}_{\text{true}}(h) > \epsilon) \leq |H|e^{-m\epsilon}$$

Even if h makes zero errors in training data, may make errors in test

©2005-2007 Carlos Guestrin

33

Limitations of Haussler '88 bound

- $P(\text{error}_{\text{true}}(h) > \epsilon) \leq |H|e^{-m\epsilon}$

- Consistent classifier

- Size of hypothesis space

©2005-2007 Carlos Guestrin

34

What if our classifier does not have zero error on the training data?

- A learner with **zero** training errors may make mistakes in test set
- What about a learner with $error_{train}(h)$ in training set?

©2005-2007 Carlos Guestrin

35

Simpler question: What's the expected error of a hypothesis?

- The error of a hypothesis is like estimating the parameter of a coin!
- Chernoff bound: for m i.i.d. coin flips, x_1, \dots, x_m , where $x_i \in \{0, 1\}$. For $0 < \epsilon < 1$:

$$P\left(\theta - \frac{1}{m} \sum_i x_i > \epsilon\right) \leq e^{-2m\epsilon^2}$$

©2005-2007 Carlos Guestrin

36

Using Chernoff bound to estimate error of a single hypothesis

$$P\left(\theta - \frac{1}{m} \sum_i x_i > \epsilon\right) \leq e^{-2m\epsilon^2}$$

©2005-2007 Carlos Guestrin

37

But we are comparing many hypothesis: **Union bound**

For each hypothesis h_i :

$$P(\text{error}_{\text{true}}(h_i) - \text{error}_{\text{train}}(h_i) > \epsilon) \leq e^{-2m\epsilon^2}$$

What if I am comparing two hypothesis, h_1 and h_2 ?

©2005-2007 Carlos Guestrin

38

Generalization bound for $|H|$ hypothesis

- **Theorem:** Hypothesis space H finite, dataset D with m i.i.d. samples, $0 < \epsilon < 1$: for any learned hypothesis h :

$$P(\text{error}_{\text{true}}(h) - \text{error}_{\text{train}}(h) > \epsilon) \leq |H|e^{-2m\epsilon^2}$$

PAC bound and Bias-Variance tradeoff

$$P(\text{error}_{\text{true}}(h) - \text{error}_{\text{train}}(h) > \epsilon) \leq |H|e^{-2m\epsilon^2}$$

**or, after moving some terms around,
with probability at least $1-\delta$:**

$$\text{error}_{\text{true}}(h) \leq \text{error}_{\text{train}}(h) + \sqrt{\frac{\ln |H| + \ln \frac{1}{\delta}}{2m}}$$

- **Important: PAC bound holds for all h ,
but doesn't guarantee that algorithm finds best h !!!**

What about the size of the hypothesis space?

$$m \geq \frac{1}{2\epsilon^2} \left(\ln |H| + \ln \frac{1}{\delta} \right)$$

- How large is the hypothesis space?

Boolean formulas with n binary features

$$m \geq \frac{1}{2\epsilon^2} \left(\ln |H| + \ln \frac{1}{\delta} \right)$$



Number of decision trees of depth k

$$m \geq \frac{1}{2\epsilon^2} \left(\ln |H| + \ln \frac{1}{\delta} \right)$$

Recursive solution

Given n attributes

H_k = Number of decision trees of depth k

$H_0 = 2$

$$\begin{aligned} H_{k+1} &= (\text{\#choices of root attribute}) * \\ &\quad (\text{\# possible left subtrees}) * \\ &\quad (\text{\# possible right subtrees}) \\ &= n * H_k * H_k \end{aligned}$$

Write $L_k = \log_2 H_k$

$L_0 = 1$

$L_{k+1} = \log_2 n + 2L_k$

So $L_k = (2^k - 1)(1 + \log_2 n) + 1$

©2005-2007 Carlos Guestrin

43

PAC bound for decision trees of depth k

$$m \geq \frac{\ln 2}{2\epsilon^2} \left((2^k - 1)(1 + \log_2 n) + 1 + \ln \frac{1}{\delta} \right)$$

- Bad!!!
 - Number of points is exponential in depth!

- But, for m data points, decision tree can't get too big...

Number of leaves never more than number data points

©2005-2007 Carlos Guestrin

44

Number of decision trees with k leaves

$$m \geq \frac{1}{2\epsilon^2} \left(\ln |H| + \ln \frac{1}{\delta} \right)$$

H_k = Number of decision trees with k leaves

$$H_0 = 2$$

$$H_{k+1} = n \sum_{i=1}^k H_i H_{k+1-i}$$

Loose bound:

$$H_k = n^{k-1} (k+1)^{2k-1}$$

Reminder:

$$|\text{DTs depth } k| = 2 * (2n)^{2^k - 1}$$

©2005-2007 Carlos Guestrin

45

PAC bound for decision trees with k leaves – Bias-Variance revisited

$$H_k = n^{k-1} (k+1)^{2k-1} \quad \text{error}_{\text{true}}(h) \leq \text{error}_{\text{train}}(h) + \sqrt{\frac{\ln |H| + \ln \frac{1}{\delta}}{2m}}$$

$$\text{error}_{\text{true}}(h) \leq \text{error}_{\text{train}}(h) + \sqrt{\frac{(k-1) \ln n + (2k-1) \ln(k+1) + \ln \frac{1}{\delta}}{2m}}$$

©2005-2007 Carlos Guestrin

46

What did we learn from decision trees?

- Bias-Variance tradeoff formalized

$$\text{error}_{\text{true}}(h) \leq \text{error}_{\text{train}}(h) + \sqrt{\frac{(k-1) \ln n + (2k-1) \ln(k+1) + \ln \frac{1}{\delta}}{2m}}$$

- Moral of the story:
Complexity of learning not measured in terms of size hypothesis space, but in maximum *number of points* that allows consistent classification
 - Complexity m – no bias, lots of variance
 - Lower than m – some bias, less variance