PAC-learning, VC Dimension and Margin-based Bounds

Machine Learning – 10701/15781
Carlos Guestrin
Carnegie Mellon University
March 6th, 2006
Announcements 1

- Midterm on Wednesday
  - open book, texts, notes,…
  - no laptops
  - bring a calculator

Review session today at 5pm
NSH 3305
Announcements 2

Final project details are out!!!
- http://www.cs.cmu.edu/~guestrin/Class/10701/projects.html
- Great opportunity to apply ideas from class and learn more
- Example project:
  - Take a dataset
  - Define learning task
  - Apply learning algorithms
  - Design your own extension
  - Evaluate your ideas
- many of suggestions on the webpage, but you can also do your own

Boring stuff:
- Individually or groups of two students
- It’s worth 20% of your final grade
- You need to submit a one page proposal on Wed. 3/22 (just after the break)
- A 5-page initial write-up (milestone) is due on 4/12 (20% of project grade)
- An 8-page final write-up due 5/8 (60% of the grade)
- A poster session for all students will be held on Friday 5/5 2-5pm in NSH atrium (20% of the grade)
- You can use late days on write-ups, each student in team will be charged a late day per day.

MOST IMPORTANT:

Have some Fun!!
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”?

Learning Theory
How likely is learner to pick a bad hypothesis

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

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

\[
P(\text{at least one of the } k \text{ was bad} \text{ and it got lucky})?
\]

\[
P(\text{one got lucky}) \leq (1-\varepsilon)^m \binom{k}{1}
\]
**Union bound**

- P(A or B or C or D or \(\ldots\)) \leq P(A) + P(B) + P(C) + \ldots
How likely is learner to pick a bad hypothesis

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

- There are $k$ hypothesis consistent with data
  - How likely is learner to pick a bad one?
    $$P(\text{h}_1 \text{ bad } \& \text{ got lucky } \text{ or } \text{h}_2 \text{ bad } \& \text{ got lucky } \text{ or } \text{h}_3 \text{ ...})$$
    $$\leq P(\text{h}_1 \text{ bad } \& \text{ lucky}) + P(\text{h}_2 \text{ bad } \& \text{ lucky}) + P(\text{h}_3 \text{ ...}) + \ldots$$
    $$\leq (3-\varepsilon)^m$$

- How big is $K$?
  - $K \leq |H|$ (loose bound!)
  - $1 - 3\varepsilon \leq e^{-\varepsilon m}$ (make eq. simpler)
Review: Generalization error in finite hypothesis spaces [Haussler ’88]

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

$$P(\text{error}_\text{true}(h) > \varepsilon) \leq |H|e^{-m\varepsilon} < 0.01$$

The probability of error decreases exponentially fast with $m$.
Using a PAC bound

Typically, 2 use cases:

1: Pick \( \varepsilon \) and \( \delta \), give you \( m \)

2: Pick \( m \) and \( \delta \), give you \( \varepsilon \)

\[
P(\text{error}_{\text{true}}(h) > \varepsilon) \leq |H|e^{-m\varepsilon}
\]
Review: Generalization error in finite hypothesis spaces [Haussler ’88]

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

\[
P(\text{error}_D(h) > \varepsilon) \leq |H| e^{-m\varepsilon}
\]

If I can always learn a consistent classifier then

Even if \( h \) makes zero errors in training data, may make errors in test
Limitations of Haussler ‘88 bound

1. Consistent classifier

\[ P(\text{error}_x(h) > \varepsilon) \leq |H|e^{-m\varepsilon} \]

There may not be such \( h \) in \( \text{class} \)!

2. Size of hypothesis space

Bound depends on \( |H| \)

Really really large?

Infinite?

Continues
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, \( x_1, \ldots, x_m \), where \( x_i \in \{0, 1\} \). For \( 0 < \varepsilon < 1 \):

\[
P \left( \theta - \frac{1}{m} \sum_{i} x_i > \varepsilon \right) \leq e^{-2m\varepsilon^2}
\]
But we are comparing many hypothesis: Union bound

For each hypothesis $h_i$:

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

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

$$P(\exists i, \text{error}_{\text{true}}(h_i) - \text{error}_{\text{train}}(h_i) > \varepsilon) \leq |H| e^{-2m\varepsilon^2}$$
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 $h$:

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

2m$\epsilon^2$ = 20 not as good 😞

Side note: Haussler's bound for consistent $h$:

$$P \leq |H|e^{-m\epsilon}$$

$\epsilon = 0.1$  
$m = 1000$  
$\implies m \cdot \epsilon = 166$
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-\(\delta\):

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

\[ \text{“bias”} \quad \downarrow \text{smaller} \quad |H| \text{small} \quad \uparrow \text{larger} \quad |H| \text{large} \]

\[ \text{“variance”} \quad \downarrow \text{smaller} \quad |H| \text{small} \quad \uparrow \text{larger} \quad |H| \text{large} \]

- Important: PAC bound holds for all \(h\), but doesn’t guarantee that algorithm finds best \(h\)!!!
What about the size of the hypothesis space?

\[ m \geq \frac{1}{2\varepsilon^2} \left( \ln |H| + \ln \frac{1}{\delta} \right) \]

How large is the hypothesis space? \(|H|\)

\[ \ln |H| \, ? \]
Boolean formulas with $n$ binary features

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

what's $\ln |H|$?

Conjunctions:

| $h_1 = x_1 \land x_2 \land x_7$ |
| $h_2 = x_2 \land x_5 \land x_8 \ldots$ |
| $\vdots$ |
| $\leq \{\emptyset, \{1,7\}, \{\emptyset, 1,7\}, \ldots\}$ |

$|H| = 3^n$ (really large)

$\ln |H| = n \ln 3$

"Small"

| $|H| = 2^n$ (really really large) |
| $\ln |H| = 2^n \ln 2$ |

too large
Number of decision trees of depth $k$

Recursive solution

Given $n$ attributes

$H_k = \text{Number of decision trees of depth } k$

$H_0 = 2$

$H_{k+1} = (\# \text{choices of root attribute}) \times (\# \text{possible left subtrees}) \times (\# \text{possible right subtrees})$

$= n \times H_k \times H_k$

Write $L_k = \log_2 H_k$

$L_0 = 1$

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

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

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

```
DT:

after $m$ leaves
not worth
splitting any more!
```

Number of leaves never more than number data points
Number of decision trees with $k$ leaves

\[ H_k = \text{Number of decision trees with } k \text{ leaves} \]

\[ H_0 = 2 \]

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

***Reminder:***

**Loose bound:**

\[ H_k \leq n^{k-1}(k + 1)2^{k-1} \]

**Reminder:**

\[ |\text{DTs depth } k| = 2 \times (2n)^{2^k-1} \]
PAC bound for decision trees with $k$ leaves – Bias-Variance revisited

\[ H_k = n^{k-1}(k + 1)^{2k-1} \]

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

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

\[
\begin{array}{c|c|c|c}
K = m & 0 & > 1 \text{ (bad)} \\
K = \alpha m & \uparrow & \downarrow \text{ down} \\
\alpha < 1 & \alpha < 1 & \alpha < 1 & \alpha < 1
\end{array}
\]
What did we learn from decision trees?

- Bias-Variance tradeoff formalized

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

- 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
What about continuous hypothesis spaces?

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

- Continuous hypothesis space: linear classifiers
  - $|H| = \infty$
  - Infinite variance???

- As with decision trees, only care about the maximum number of points that can be classified exactly!
How many points can a linear boundary classify exactly? (1-D)
How many points can a linear boundary classify exactly? (2-D)
How many points can a linear boundary classify exactly? (d-D)
Shattering a set of points

**Definition:** a **dichotomy** of a set \( S \) is a partition of \( S \) into two disjoint subsets.

**Definition:** a set of instances \( S \) is **shattered** by hypothesis space \( H \) if and only if for every dichotomy of \( S \) there exists some hypothesis in \( H \) consistent with this dichotomy.

\[ \forall \text{ dichotomies } \exists h \text{ labels correctly: } \]
\[ \{ x_1, x_2 \} \rightarrow + \quad \{ x_3, x_4 \} \rightarrow - \quad \text{choose } h_7 \]
\[ \{ x_3 \} \rightarrow + \quad \{ x_1, x_2, x_4 \} \rightarrow - \quad \text{choose } h_{52} \]

\( \forall \text{ splits } \exists h \text{ shattered } \]
Definition: The Vapnik-Chervonenkis dimension, $VC(H)$, of hypothesis space $H$ defined over instance space $X$ is the size of the largest finite subset of $X$ shattered by $H$. If arbitrarily large finite sets of $X$ can be shattered by $H$, then $VC(H) \equiv \infty$. 

[Handwritten notes: hyperplane in 2d, can’t shatter, but I don’t care... I get to pick locations, I pick: ]
PAC bound using VC dimension

- Number of training points that can be classified exactly is VC dimension!!!
  - Measures relevant size of hypothesis space, as with decision trees with k leaves
  - Bound for infinite dimension hypothesis spaces:

\[
\text{error}_{true}(h) \leq \text{error}_{train}(h) + \sqrt{\frac{VC(H) \left( \ln \frac{2m}{VC(H)} + 1 \right) + \ln \frac{4}{\delta}}{m}}
\]

\[
\begin{array}{c|c|c}
\text{high VC dim.} & \uparrow & \downarrow \\
\text{low VC dim.} & \downarrow & \uparrow \\
\end{array}
\]
Examples of VC dimension

- Linear classifiers:
  - $\text{VC}(H) = d + 1$, for $d$ features plus constant term $b$

- Neural networks
  - $\text{VC}(H) = \#\text{parameters}$
  - Local minima means NNs will probably not find best parameters

- 1-Nearest neighbor?
Another VC dim. example

What’s the VC dim. of decision stumps in 2d?

VC at least 3

prove \( \exists \) no 4 points that can be labeled correctly

\((x_1, x_2, x_3, x_4)\)

\(x_1 \text{ min}_{xy} \text{ coord.} \rightarrow t\)

\(x_2 \text{ max}_{xy} \rightarrow +\)

\(x_3, x_4 \rightarrow -\)

\(x_1 \geq x\) or \(x_1 \leq x\)

\(x_2 \geq y\) or \(x_2 \leq y\)

\(x_3 > x\) or \(x_3 < x\)

\(x_4 > y\) or \(x_4 < y\)
PAC bound for SVMs

- SVMs use a linear classifier
- For $d$ features, $VC(H) = d+1$:

$$error_{true}(h) \leq error_{train}(h) + \sqrt{\frac{(d + 1) \left( \ln \frac{2m}{d+1} + 1 \right) + \ln \frac{4}{\delta}}{m}}$$
VC dimension and SVMs: Problems!!

Doesn’t take margin into account

\[
\text{error}_{true}(h) \leq \text{error}_{train}(h) + \sqrt{\frac{(d+1) \left( \ln \frac{2m}{d+1} + 1 \right) + \ln \frac{4}{\delta}}{m}}
\]

- What about kernels?

- Polynomials: num. features grows really fast = Bad bound
  \[
  \text{num. terms} = \binom{p + n - 1}{p} = \frac{(p + n - 1)!}{p!(n - 1)!}
  \]
  \[n \text{ – input features}
  \]
  \[p \text{ – degree of polynomial}
  \]

- Gaussian kernels can classify any set of points exactly

\[\text{bound bad!!} \quad \text{"VC} \geq \infty\]

©2006 Carlos Guestrin
Margin-based VC dimension

- H: Class of linear classifiers: \( w \cdot \Phi(x) \) (b=0)
  - Canonical form: \( \min_j |w \cdot \Phi(x_j)| = 1 \)
- VC(H) = \( R^2 \frac{w \cdot w}{\text{margin}^2} \)
  - Doesn’t depend on number of features!!!
  - \( R^2 = \max_j \Phi(x_j) \cdot \Phi(x_j) \) – magnitude of data
  - \( R^2 \) is bounded even for Gaussian kernels \( \rightarrow \) bounded VC dimension

- Large margin, low \( w \cdot w \), low VC dimension – Very cool!
Applying margin VC to SVMs?

\[
\text{error}_{\text{true}}(h) \leq \text{error}_{\text{train}}(h) + \sqrt{\frac{VC(H) \left( \ln \frac{2m}{VC(H)} + 1 \right) + \ln \frac{4}{\delta}}{m}}
\]

- **VC(H) = \overline{R^2 \ w \cdot w}**
  - R^2 = \max_j \Phi(x_j) \cdot \Phi(x_j) – magnitude of data, doesn’t depend on choice of \(w\)

- SVMs minimize \overline{w \cdot w}

- SVMs minimize VC dimension to get best bound?
  - Not quite right: 😞
    - Bound assumes VC dimension chosen before looking at data
    - Would require union bound over infinite number of possible VC dimensions...
    - But, it can be fixed!
Structural risk minimization theorem

\[
\text{error}_{\text{true}}(h) \leq \text{error}_{\text{train}}^{\gamma}(h) + C\sqrt{\frac{R^2}{\gamma^2} \ln m + \ln \frac{1}{\delta}}
\]

\[
\text{error}_{\text{train}}^{\gamma}(h) = \text{num. points with margin} < \gamma
\]

- For a family of hyperplanes with margin \( \gamma > 0 \)
  - \( w \cdot w \leq 1 \)
- SVMs maximize margin \( \gamma \) + hinge loss
  - Optimize tradeoff training error (bias) versus margin \( \gamma \)
    (variance)
Reality check – Bounds are loose

\[ \text{error}_{\text{true}}(h) \leq \text{error}_{\text{train}}(h) + \sqrt{\frac{(d + 1) \left( \ln \frac{2m}{d+1} + 1 \right) + \ln \frac{4}{\delta}}{m}} \]

- Bound can be very loose, why should you care?
  - There are tighter, albeit more complicated, bounds
  - Bounds gives us formal guarantees that empirical studies can’t provide
  - Bounds give us intuition about complexity of problems and convergence rate of algorithms

©2006 Carlos Guestrin
What you need to know

- Finite hypothesis space
  - Derive results
  - Counting number of hypothesis
  - Mistakes on Training data

- Complexity of the classifier depends on number of points that can be classified exactly
  - Finite case – decision trees
  - Infinite case – VC dimension

- Bias-Variance tradeoff in learning theory

- Margin-based bound for SVM

- Remember: will your algorithm find best classifier?
Big Picture

Machine Learning – 10701/15781
Carlos Guestrin
Carnegie Mellon University
March 6th, 2006
What you have learned thus far

- Learning is function approximation
- Point estimation
- Regression
- Naïve Bayes
- Logistic regression
- Bias-Variance tradeoff
- Neural nets
- Decision trees
- Cross validation
- Boosting
- Instance-based learning
- SVMs
- Kernel trick
- PAC learning
- VC dimension
- Margin bounds
- Mistake bounds
Review material in terms of…

- Types of learning problems
- Hypothesis spaces
- Loss functions
- Optimization algorithms
Text Classification

Company home page vs Personal home page vs Univeristy home page

Our energy exploration, production, and distribution operations span the globe, with activities in more than 100 countries.

At TOTAL, we draw our greatest strength from our fast-growing oil and gas reserves. Our strategic emphasis on natural gas provides a strong position in a rapidly expanding market.

Our expanding refining and marketing operations in Asia and the Mediterranean Rim complement already solid positions in Europe, Africa, and the U.S.

Our growing specialty chemicals sector adds balance and profit to the core energy business.
Function fitting

Temperature data
Monitoring a complex system

- Reverse water gas shift system (RWGS)
- Learn model of system from data
- Use model to predict behavior and detect faults
Types of learning problems

- Classification
- Regression
- Density estimation

Input – Features

Output?
The learning problem

- Data: $\langle x_1, \ldots, x_n, y \rangle$
- Learning task
- Features/Function approximator
- Loss function
- Optimization algorithm
- Learned function
Comparing learning algorithms

- Hypothesis space
- Loss function
- Optimization algorithm
Naïve Bayes versus Logistic regression

Naïve Bayes

\[ P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)} \]

\[ P(X|Y) = \prod_i P(X_i|Y) \]

Logistic regression

\[ P(Y = 1|x) = \frac{1}{1 + \exp(w_0 + \sum_i w_i x_i)} \]
Naïve Bayes versus Logistic regression – Classification as density estimation

\[ P(Y \mid X) \]

- Choose class with highest probability

- In addition to class, we get certainty measure
Logistic regression versus Boosting

**Logistic regression**

\[
P(Y = y_i | x) = \frac{1}{1 + \exp(-y_i (w \cdot x + b))}
\]

Log-loss

\[
\sum_{j=1}^{m} \log \left[ 1 + \exp(-y_i (w \cdot x_j + b)) \right]
\]

**Boosting**

**Classifier**

\[
\text{sign} \left( \sum_{t=1}^{T} \alpha_t h_t(x) \right)
\]

**Exponential-loss**

\[
\frac{1}{m} \sum_{j=1}^{m} \exp \left( -y_j \sum_{t=1}^{T} \alpha_t h_t(x_j) \right)
\]
Linear classifiers – Logistic regression versus SVMs

\[ w \cdot x + b = 0 \]
What’s the difference between SVMs and Logistic Regression? (Revisited again)

<table>
<thead>
<tr>
<th></th>
<th>SVMs</th>
<th>Logistic Regression</th>
</tr>
</thead>
<tbody>
<tr>
<td>Loss function</td>
<td>Hinge loss</td>
<td>Log-loss</td>
</tr>
<tr>
<td>High dimensional features with kernels</td>
<td>Yes!</td>
<td>Yes!</td>
</tr>
<tr>
<td>Solution sparse</td>
<td>Often yes!</td>
<td>Almost always no!</td>
</tr>
<tr>
<td>Type of learning</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
SVMs and instance-based learning

**SVMs**

\[ w \cdot \Phi(x) = \sum_i \alpha_i y_i K(x, x_i) \]
\[ b = y_k - \sum_i \alpha_i y_i K(x_k, x_i) \]
for any \( k \) where \( C > \alpha_k > 0 \)

**Instance based learning**

\[ P(y | x) = \frac{\sum_i y_i K(x, x_i)}{\sum_i K(x, x_i)} > 0.5? \]

\[ \text{sign} \left( \sum_i y_i K(x, x_i) - 0.5 \sum_i K(x, x_i) \right) \]
Instance-based learning versus Decision trees

1-Nearest neighbor

Decision trees
Logistic regression versus Neural nets

\[ g(w_0 + \sum_{i} w_i x_i) = \frac{1}{1 + e^{-(w_0 + \sum_{i} w_i x_i)}} \]

Logistic regression          Neural Nets
Linear regression versus Kernel regression

Linear Regression

Kernel regression

Kernel-weighted linear regression
Kernel-weighted linear regression

Local basis functions for each region

Kernels average between regions
SVM regression

$$\min_{w,\xi,\bar{\xi}} \quad \frac{1}{2} w \cdot w + C \sum_{j=1}^{m} (\xi_j + \bar{\xi}_j)$$

s.t. $$y_j - (w \cdot x_j + b) \leq \epsilon + \xi_j$$

$$ (w \cdot x_j + b) - y_j \leq \epsilon + \bar{\xi}_j$$

$$\xi_j \geq 0, \quad \bar{\xi}_j \geq 0, \quad \forall j$$
BIG PICTURE
(a few points of comparison)

Naïve Bayes
DE, LL

Logistic regression
DE, LL

SVMs
CI, Mrg

Instance-based Learning
DE, CI, Reg

Neural Nets
DE, CI, Reg, RMS

Decision trees
DE, CI, Reg

Boosting
CI, exp-loss

SVM regression
Reg, Mrg

Kernel regression
Reg, RMS

Linear regression
Reg, RMS

DE
density estimation

CI
Classification

Reg
Regression

LL
Log-loss/MLE

Mrg
Margin-based

RMS
Squared error

This is a very incomplete view!!!