Learning Theory

Machine Learning – CSE546

Carlos Guestrin
University of Washington

October 27, 2013

A simple setting…

- Classification
  - N data points iid
  - **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 – $\text{error}_{train}(h) = 0$
- What is the probability that h has more than ε true error?
  - $\text{error}_{true}(h) \geq \varepsilon$ For some $\varepsilon > 0$

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

- Hypothesis h that is **consistent** with training data → got N i.i.d. points right
  - h "bad" if it gets all this data right, but has high true error

$\varepsilon > 0$

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

less than $1-\varepsilon$

if error $\varepsilon = 0.25$
75% points are correct $= 1-\varepsilon$

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

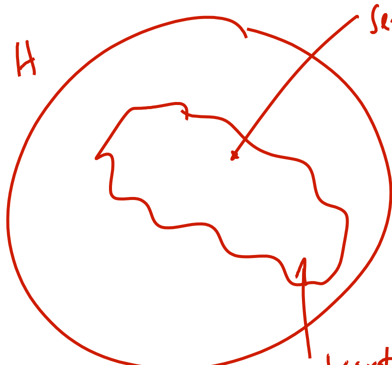
less than $(1-\varepsilon)^N$

prob bad h wins

decays exponentially in N

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

H



Set of hypothesis consistent with training data, don't want to make assumptions about which one was picked

worst-case bound

How likely is learner to pick a bad hypothesis

- Prob. h with $error_{true}(h) \geq \varepsilon$ gets N data points right

  less than $(1-\varepsilon)^N$

- There are k hypothesis consistent with data $\rightarrow h_1, \dots h_k$
  - How likely is learner to pick a bad one?

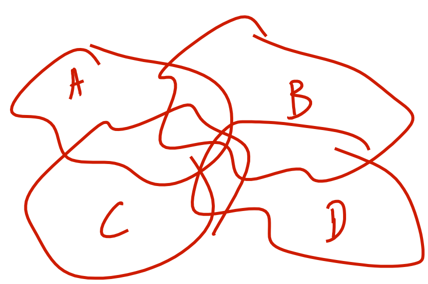
some bad, some good ⇒ deal with worst case

$P(\exists h$ consistent with train data, $error_{true}(h) \geq \varepsilon)$

$= \underbrace{P(error_{true}(h_1) \geq \varepsilon \text{ OR } error_{true}(h_2) \geq \varepsilon \text{ OR} \dots \text{ OR } error_{true}(h_k) \geq \varepsilon)}_{\text{Bound?}}$

Union bound

- P(A or B or C or D or …) $\leq P(A) + P(B) + P(C) + P(D) \dots$



How likely is learner to pick a bad hypothesis

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

less than $(1-\varepsilon)^N$

- There are k hypothesis consistent with data
  - How likely is it that learner will pick a bad one out of these k choices?

$P(\exists h \text{ consistent with train data}, \text{error}_{true}(h) \geq \varepsilon) \leq k(1-\varepsilon)^N$

$\leq |H|(1-\varepsilon)^N$

what's k?

$k \leq |H|$

total # hypotheses

crazy loose

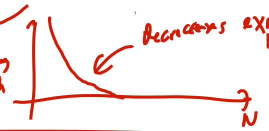
Generalization error in finite hypothesis spaces [Haussler '88]

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

$$P(error_{true}(h) \geq \epsilon) \leq |H|e^{-N\epsilon}$$

prob. that you'll be fired

prob picking bad h



decreases exponentially

$\leq |H|(1-\varepsilon)^N \leq |H|(e^{-\varepsilon})^N = |H|e^{-\varepsilon N}$

for $0 \leq \varepsilon \leq 1$

$1-\varepsilon \leq e^{-\varepsilon}$

Using a PAC bound

- Typically, 2 use cases:
  - 1: Pick ε and δ, give you N
  - 2: Pick N and δ, give you ε

$$P(error_{true}(h) > \epsilon) \leq |H|e^{-N\epsilon}$$

tolerance ← prob. fixed

$$\underbrace{P(error_{true}(h) \geq \varepsilon)}_{\text{bad event}} \leq \underbrace{|H| e^{-N\varepsilon}}_{\text{upper bound}} \leq \underbrace{\delta}_{\text{acceptable prob.}}$$

$\ln|H| - N\varepsilon \leq \ln \delta$

log of $|H|$; log dependence on δ

$$\Rightarrow \underbrace{N}_{\text{amount data needed}} \geq \frac{\ln|H| + \ln\frac{1}{\delta}}{\varepsilon}$$

linear in $1/\varepsilon$

$$\varepsilon \geq \frac{\ln|H| + \ln\frac{1}{\delta}}{N}$$

decrease $O\left(\frac{1}{N}\right)$

$\Rightarrow$ very good rate

More general settings: $O\left(\frac{1}{\sqrt{N}}\right)$

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

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

$$P(error_{true}(h) > \epsilon) \leq |H|e^{-N\epsilon}$$

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

Limitations of Haussler '88 bound

$$P(error_{true}(h) > \epsilon) \leq |H| e^{-N\epsilon}$$

- Consistent classifier

error train (h) = 0 → highly unrealistic

label noise, complex data, model bias fitting problems

Overfit

- Size of hypothesis space

ln |H| is bad

|H| very very large

|H| infinite (e.g. SVM)

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?

what happens when

error train (h) > 0 ?

:) error true (h) ?

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

- The error of a hypothesis is like estimating the parameter of a coin! $\theta \approx \hat{\theta} = \frac{3}{5}$
- Chernoff bound: for *N* i.i.d. coin flips, $x^1,\ldots,x^N$, where $x^j \in \{0,1\}$. For $0<\varepsilon<1$:

$$P\Bigg(\underset{\text{true}}{\theta} - \underbrace{\frac{1}{N}\sum_{j=1}^{N} x^j}_{\text{mean of train data } \hat{\theta}} > \underset{\varepsilon}{\epsilon}\Bigg) \leq \underbrace{e^{-2N\epsilon^2}}_{\text{something exp in } N}$$

Using Chernoff bound to estimate error of a single hypothesis

$$P\Bigg(\underset{\text{true } \mathrm{error}_{\mathrm{true}}(h)}{\theta} - \underbrace{\frac{1}{N}\sum_{j=1}^{N} x^j}_{\hat{\theta} = \text{error train}} > \epsilon\Bigg) \leq e^{-2N\epsilon^2}$$

$\theta = \int_x p(x)\, \mathbb{1}(h(x) \neq t(x))\, dx$ ← sample estimate of integral

$\frac{1}{N}\sum_{j=1}^{N} \mathbb{1}(h(x^j) \neq y^j) = \mathrm{error}_{\mathrm{train}}(h)$, $\quad y^j = t(x^j)$

if label noise: $\boxed{\theta = \int_x \int_{y_x} p(x)\, p(y_x|x)\, \mathbb{1}(h(x) \neq y_x)\, dx\, dy}$

$$P\left(\mathrm{error}_{\mathrm{true}}(h) - \mathrm{error}_{\mathrm{train}}(h) \geq \varepsilon\right) \leq e^{-2N\varepsilon^2}$$

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

over fit by more than ε

For each hypothesis h_i:

$$P\left(error_{true}(h_i) - error_{train}(h_i) > \epsilon\right) \leq e^{-2N\epsilon^2}$$

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

is h_1 better than h_2?

Danger: over-fit: $P\left(error_{train}(h_1) < error_{train}(h_2),\ \text{but } error_{true}(h_1) > error_{true}(h_2)\right)$

Bad event $P\left(\left[error_{true}(h_1) - error_{train}(h_1) \geq \varepsilon\right] \text{ OR } \left[error_{true}(h_2) - error_{train}(h_2) \geq \varepsilon\right]\right)$

$\leq P\left(error_{true}(h_1) - error_{train}(h_1) \geq \varepsilon\right) + P\left(error_{true}(h_2) - error_{train}(h_2) \geq \varepsilon\right)$

$\leq 2\, e^{-2N\varepsilon^2}$

Generalization bound for |H| hypothesis

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

$$P\left(error_{true}(h_i) - error_{train}(h_i) > \epsilon\right) \leq e^{-2N\epsilon^2} \leq \delta$$

hold $\forall h$:

$P\left(error_{true}(h) - error_{train}(h) > \varepsilon\right) \leq |H|\, e^{-2N\varepsilon^2}$

$\varepsilon \geq \sqrt{\dfrac{\ln|H| + \ln\frac{1}{\delta}}{2N}} \rightarrow O\left(\frac{1}{\sqrt{N}}\right)$ rate

with probability at least $1-\delta$: $error_{true}(h) - error_{train}(h) \leq \varepsilon$

PAC bound and Bias-Variance tradeoff

$$P\left(error_{true}(h) - error_{train}(h) > \epsilon\right) \leq e^{-2N\epsilon^2}$$

or, after moving some terms around, with probability at least 1-δ: bound on ε

If want small

$$error_{true}(h) \leq error_{train}(h) + \sqrt{\frac{\ln|H| + \ln\frac{1}{\delta}}{2N}}$$

	"bias"	"variance"
"Complex hypothesis space"	low	large ⇐ |H| is large
"simple H"	large	low ⇐ |H| is small

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

What about the size of the hypothesis space?

$$N \geq \frac{\ln|H| + \ln\frac{1}{\delta}}{2\epsilon^2}$$

- How large is the hypothesis space?

|H|?

|H| = really large ⇒ log |H| = only large ⇒ OK

but

|H| = really really large ⇒ ln |H| = really large ⇒ lots data

Boolean formulas with m binary features

$x_1 \wedge \neg x_2 \vee x_7 \wedge x_2 \cdots$

$$N \geq \frac{\ln|H| + \ln\frac{1}{\delta}}{2\epsilon^2}$$

H: any boolean formula : $|H|$?

$x_1, x_2 \ldots x_m$	y
0 0 0 0 0	0 or 1
0 0 0 0 0 1	0 or 1
⋮	

2^m rows

each row 2 possibilities

$|H| = 2^{2^m}$ = really really large

$\ln|H| = 2^m \ln 2$

exp. many data points

H: all conjunctions with negation

$x_1 \wedge \neg x_3 \wedge x_7$

each feature : 3 possibilities → positive, negated, absent

$|H| = 3^m$ ⇒ really large

$\ln|H| = m \ln 3$

linear in # of features

Number of decision trees of depth k

$$N \geq \frac{\ln|H| + \ln\frac{1}{\delta}}{2\epsilon^2}$$

Recursive solution
Given m attributes
H_k = Number of decision trees of depth k
H_0 =2
H_{k+1} = (#choices of root attribute) *
(# possible left subtrees) *
(# possible right subtrees)
$\leq$ m * H_k * H_k

Write $L_k = \log_2 H_k$
$L_0 = 1$
$L_{k+1} = \log_2 m + 2L_k$
So $\underbrace{L_k = (2^k-1)(1+\log_2 m) +1}$

simplify

$\ln|H| \leq 2^k \log m$

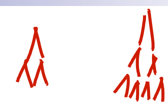
really really big in terms of depth

very nice in terms of num features

PAC bound for decision trees of depth k

exponential in depth

$$N \geq \frac{2^k \log m + \ln \frac{1}{\delta}}{\epsilon^2}$$



twice as much data

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

- But, for *N* data points, decision tree can't get too big…

no reason to have more than N leaves

Number of leaves never more than number data points

Number of Decision Trees with k Leaves

- Number of decision trees of depth k is really really big:
  - ln |H| is about $2^k \log m$
- Decision trees with up to k leaves:
  - |H| is about $m^k k^{2k}$ ← only really large
    - A very loose bound

$$\ln|H| \leq k \ln m + 2k \ln k$$

much better!

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

$\ln |H_{\text{DTs k leaves}}| \leq 2k(\ln m + \ln k)$

$error_{true}(h) \leq error_{train}(h) + \sqrt{\frac{\ln |H| + \ln \frac{1}{\delta}}{2N}}$

$$\boxed{error_{true}(h) \leq error_{train}(h) + \sqrt{\frac{2k(\ln m + \ln k) + \ln \frac{1}{\delta}}{2N}}}$$

max number of leaves k	"bias"	"variance"
$K \gtrsim N$	goes to zero	LARGE greater than 1
$K << N$	potentially larger	potentially small

What did we learn from decision trees?

- Bias-Variance tradeoff formalized

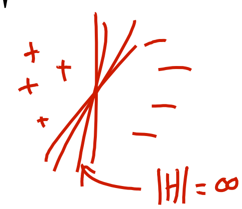
$$\boxed{error_{true}(h) \leq error_{train}(h) + \sqrt{\frac{2k(\ln m + \ln k) + \ln \frac{1}{\delta}}{2N}}}$$

- 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
  - $k \approx N$ Complexity *N* – no bias, lots of variance
  - Lower than *N* – some bias, less variance $k << N$

What about continuous hypothesis spaces?

$$error_{true}(h) \leq error_{train}(h) + \sqrt{\frac{\ln|H| + \ln\frac{1}{\delta}}{2N}}$$

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



- **As with decision trees, only care about the maximum number of points that can be classified exactly!**
  - **Called VC dimension… see readings for details**

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 ← # of leaves
  - Infinite case – VC dimension
- Bias-Variance tradeoff in learning theory
- Remember: will your algorithm find best classifier?

Markov Decision Processes (MDPs)

Machine Learning – CSE546
Carlos Guestrin
University of Washington

December 2, 2013

e.g. classification : supervised learning

X → Y

(GPA, grade) → {hire, not hire}

unsupervised case : e.g. clustering

just X (GPA, grade) ⇒ groups of people with similar Xs

Reinforcement Learning

training by feedback

weak feedback X₁ ← is good

X₂ ← is bad

Learning to act

- Reinforcement learning
- An agent
  - Makes sensor observations
  - Must select action
  - Receives rewards
    - positive for "good" states
    - negative for "bad" states

[Ng et al. '05]

Markov Decision Process (MDP) Representation

position of peasant

- State space: $x = (x_1, \ldots x_n)$
  gold
  - Joint state **x** of entire system
- Action space: build castle ...
  - Joint action **a**= {a_1,…, a_n} for all agents
- Reward function: positive reward when x is win game
  - Total reward R(**x**,**a**)
    - sometimes reward can depend on action
- Transition model: have gold no castle
  - Dynamics of the entire system P(**x**'|**x**,**a**)
    build castle
    have castle but no gold

want a policy
$\pi(x) \Rightarrow a$
what action at each state

Discount Factors

$\gamma \in [0,1)$

People in economics and probabilistic decision-making do this all the time.

The "Discounted sum of future rewards" using discount factor γ" is

(reward now) +

γ (reward in 1 time step) +

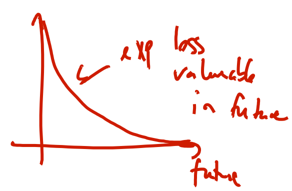
γ^2 (reward in 2 time steps) +

γ^3 (reward in 3 time steps) +

:

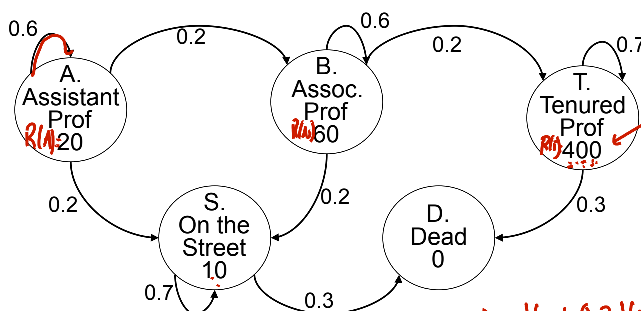
: (infinite sum)

maximize sum of discounted rewards



The Academic Life

Assume Discount Factor $\gamma = 0.9$



in cents

Define:

V_A = Expected discounted future rewards starting in state A $= 20 + \gamma(0.6\,V_A + 0.2\,V_B + 0.2\,V_S)$

V_B = Expected discounted future rewards starting in state B $= 60 + \gamma(0.6\,V_B + 0.2\,V_S + 0.2\,V_T)$

V_T = " " " " " " " T

V_S = " " " " " " " S

V_D = " " " " " " " D

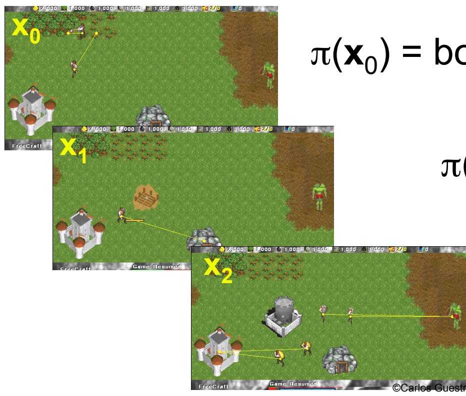
n states.
n unknowns
n equations
linear
⇒ e.g. matrix inversion

How do we compute V_A, V_B, V_T, V_S, V_D ?

Policy

Policy: $\pi(\mathbf{x}) = \mathbf{a}$

At state $\mathbf{x}$, action $\mathbf{a}$ for all agents



$\pi(\mathbf{x}_0)$ = both peasants get wood

$\pi(\mathbf{x}_1)$ = one peasant builds barrack, other gets gold

$\pi(\mathbf{x}_2)$ = peasants get gold, footmen attack

Value of Policy

Value: $V_\pi(\mathbf{x})$

Expected long-term reward starting from $\mathbf{x}$

formal view of recursion

and act according to π

$$V_\pi(\mathbf{x_0}) = \mathbf{E}_\pi[R(\mathbf{x}_0) + \gamma\, R(\mathbf{x}_1) + \gamma^2\, R(\mathbf{x}_2) + \gamma^3\, R(\mathbf{x}_3) + \gamma^4\, R(\mathbf{x}_4) + \ldots]$$

Start from $\mathbf{x_0}$ a =

Future rewards discounted by γ in [0,1)