1 handout: slides

asst 5 milestone was due

Supervised Learning: Summary So Far

learning as function approximation

k-NN: distance function (any attributes), any labels
Neural network: numeric attributes, numeric or binary labels

> **Regression:** incremental training with LMS
> **3-Layer ANN:** train with BackProp

Inductive Logic Programming: logical concepts
Decision Trees: easier with discrete attributes and labels

Naive Bayes

Bayes' Theorem

$$P(H|D) = \frac{P(H)P(D|H)}{P(D)}$$

Bayes' Theorem

$$P(H|D) = \frac{P(H)P(D|H)}{P(D)}$$

$$\begin{aligned} P(H) &= 0.0001 \\ P(D|H) &= 0.99 \\ P(D) &= 0.01 \end{aligned}$$

$$P(H|D) =$$

Bayes' Theorem

$$P(H|D) = \frac{P(H)P(D|H)}{P(D)}$$

$$\begin{aligned} P(H) &= 0.0001 \\ P(D|H) &= 0.99 \\ P(D) &= 0.01 \end{aligned}$$

$$P(H|D) =$$

If you don't have P(D),

Bayes' Theorem

$$P(H|D) = \frac{P(H)P(D|H)}{P(D)}$$

$$\begin{aligned} P(H) &= 0.0001 \\ P(D|H) &= 0.99 \\ P(D) &= 0.01 \end{aligned}$$

$$P(H|D) =$$

If you don't have P(D), somtimes it helps to note that

$$P(D) = P(D|H)P(H) + P(D|\neg H)P(\neg H)$$

A Naive Bayesian Model

Bayes' Theorem:

$$P(H|D) = \frac{P(H)P(D|H)}{P(D)}$$

A Naive Bayesian Model

Bayes' Theorem:

$$P(H|D) = \frac{P(H)P(D|H)}{P(D)}$$

naive model:

$$P(D|H) = P(x_i, \ldots, x_n|H) = \prod_i P(x_i|H)$$

A Naive Bayesian Model

Bayes' Theorem:

$$P(H|D) = \frac{P(H)P(D|H)}{P(D)}$$

naive model:

$$P(D|H) = P(x_i, \ldots, x_n|H) = \prod_i P(x_i|H)$$

attributes independent, given class

A Naive Bayesian Model

Bayes' Theorem:

$$P(H|D) = \frac{P(H)P(D|H)}{P(D)}$$

naive model:

$$P(D|H) = P(x_i, \ldots, x_n|H) = \prod_i P(x_i|H)$$

attributes independent, given class

$$P(H|x_1, \ldots, x_n) = \alpha P(H) \prod_i P(x_i|H)$$

The 'Naive Bayes' Classifier

$$P(H|x_1, \ldots, x_n) = \alpha P(H) \prod_i P(x_i|H)$$

attributes independent, given class

maximum *a posteriori* = pick highest
maximum likelihood = ignore prior

watch for sparse data when learning!

learning as density estimation

Break

- asst 5
- exam 2
- projects

Boosting

Ensemble Learning

committees, ensembles

weak vs strong learners

reduce variance, expand hypothesis space (eg, half-spaces)

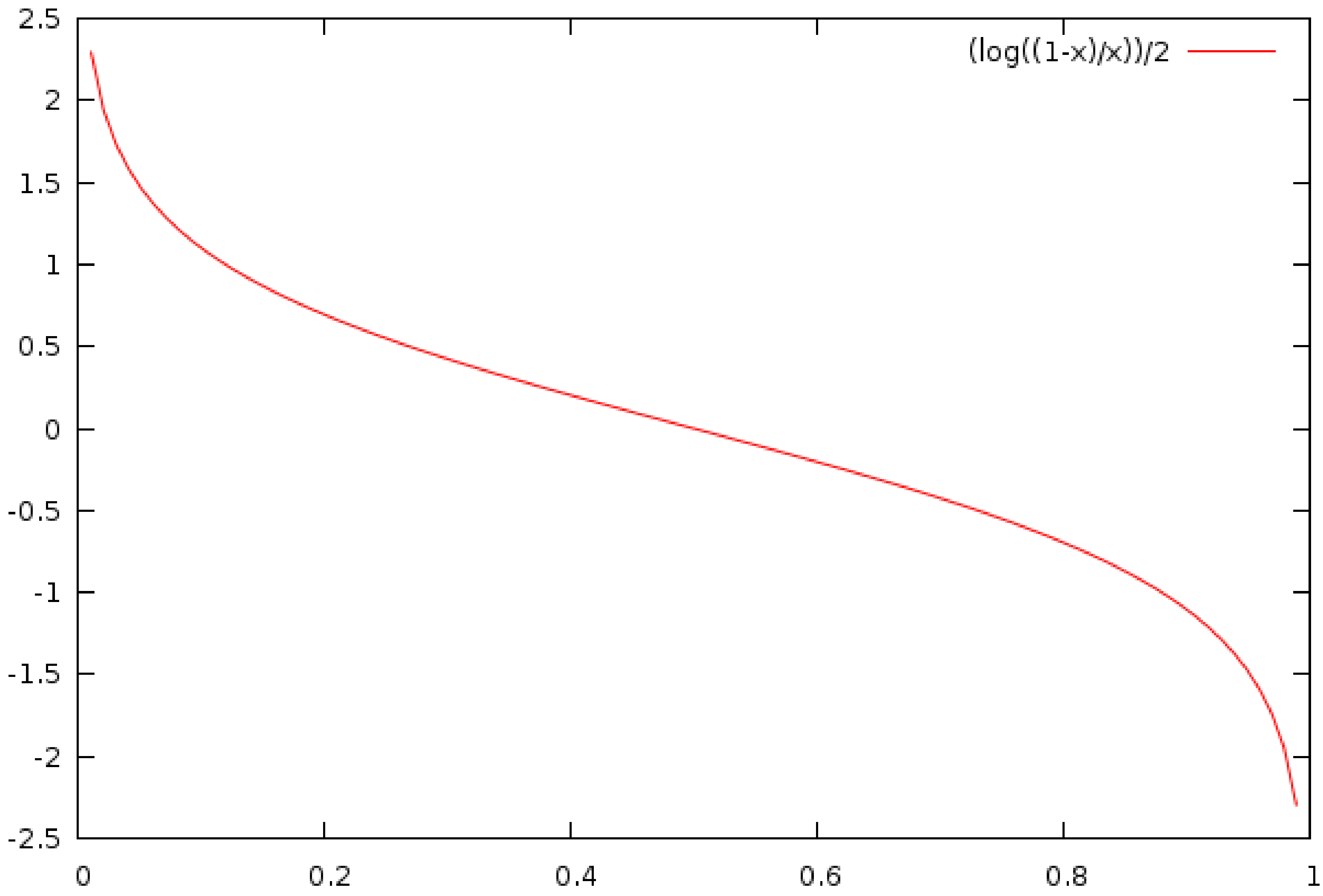
AdaBoost

N examples, T rounds, L a weak learner on weighted examples

$p \leftarrow$ uniform distribution over the N examples
for $t = 1$ to T do
    $h_t \leftarrow$ call L with weights p
    $\epsilon_t \leftarrow h_t$'s weighted misclassification probability
    if $\epsilon_t = 0$, return h_t
    $\alpha_t \leftarrow \frac{1}{2}\ln(\frac{1-\epsilon_t}{\epsilon_t})$
    for each example i
        if $h_t(i)$ is correct, $p_i \leftarrow p_i e^{-\alpha_t}$
        else, $p_i \leftarrow p_i e^{\alpha_t}$
    normalize p to sum to 1
return the h weighted by the α

to classify, choose label with highest sum of weighted votes

Boosting Function



doesn't overfit (maximizes margin even when no error)
outliers get high weight, can be inspected
problems:

- not enough data
- hypothesis class too small
- boosting: learner too weak, too strong

Supervised Learning: Summary

k-NN: distance function (any attributes), any labels
Neural network: numeric attributes, numeric or binary labels

> **Perceptron:** equivalent to linear regression
> **3-Layer ANN:** BackProp learning

Decision Trees: easier with discrete attributes and labels
Inductive Logic Programming: logical concepts
Naive Bayes: easier with discrete attributes and labels
Boosting: general wrapper to improve performance

Didn't cover: RBFs, EBL, SVMs

EOLQs

- What question didn't you get to ask today?
- What's still confusing?
- What would you like to hear more about?

Please write down your most pressing question about AI and put it in the box on your way out.
Thanks!