CS598: Machine Learning and Natural Language

Lecture 7: Introduction to Classification

Oct. 5,12 2004

Dan Roth

University of Illinois, Urbana-Champaign

danr@cs.uiuc.edu

http://L2R.cs.uiuc.edu/~danr

Context Sensitive Text Correction

Illinois' bored of education.	board
We took a walk it the park two.	in, too
We fill it need no be this way	feel, not
The amount of chairs in the room is…	number
I'd like a peace of cake for desert	piece, dessert

Disambiguation Problems

Middle Eastern _____ are known for their sweetness

Task: Decide which of { deserts , desserts } is more likely in the given context.

Ambiguity: modeled as confusion sets (class labels C)

C={ deserts, desserts}

C={ Noun,Adj.., Verb…}

C={ topic=Finance, topic=Computing}

C={ NE=Person, NE=location}

Disambiguation Problems

- Archetypical disambiguation problem

- Data is available (?)

- In principle, a solved problem

  Golding&Roth, Mangu&Brill,...

- But

  Many issues are involved in making an "in principle" solution a realistic one

Learning to Disambiguate

- Given
  - a confusion set C={ deserts, desserts}
  - sentence (s)

  Middle Eastern ____ are known for their sweetness
- Map into a feature based representation
- Learn a function F_C that determines which of C={ deserts, desserts} more likely in a given context.
- Evaluate the function on future C sentences

Learning Approach: Representation

S= I don't know whether to laugh or cry

[x x x x]

Consider words, pos tags, relative location in window

Generate binary features representing presence of:

a word/pos within window around target word

don't within +/-3 know within +/-3 Verb at -1

to within +/-3 laugh within +/-3 to a +1

conjunctions of size 2, within window of size 3

words: know __ to; ___ to laugh

pos+words: Verb __ to; ___ to Verb

Learning Approach: Representation

S= I don't know whether to laugh or cry

Is represented as a set of its active features

S= (don't at -2 , know within +/-3,… ____to Verb,…)

Label= the confusion set element that occurs in the text

Hope: S=I don't care whether to laugh or cry

has almost the same representation

This representation can be used by any propositional learning algorithm. (features, examples)

Previous works: TBL (Decision Lists) NB, SNoW, DT,…

Notes on Representation

- There is a huge number of potential features (~10^5).
- Out of these – only a small number is actually active in each example.

- The representation can be significantly smaller if we list only features that are active in each examples.

- Some algorithms can take this into account. Some cannot. (Later).

Notes on Representation (2)

- Formally:
  A feature =a characteristic function over sentences

$$\chi : \boldsymbol{S} \rightarrow \{0,1\}$$

- When the number of features is fixed, the collection of all examples is

$$\{(\chi_1, \chi_2, \ldots, \chi_{\boldsymbol{n}})\} \equiv \{0,1\}^{\boldsymbol{n}}$$

- When we do not want to fix the number of features (very large number, on-line algorithms,…) can work in the infinite attribute domain

$$\{(\chi_1, \chi_2, \ldots, \chi_{\boldsymbol{n}}, \ldots)\} \equiv \{0,1\}^{\infty}$$

An Algorithm

Consider all training data S: {(l, f, f,….)}

Represent as:

S={(f, #(l=0), #(l=1)} for all features

1. Choose best feature f* (and the label it suggests)
2. S ← S \ {Examples labeled in (1)}
3. GoTo 1

An Algorithm:Hypothesis

If f1 then label

Else, if f2 then label

Else…

Else default label

A decision list

Issues: How well will this do?

We train on the training data, what about new data?

Generalization

I saw the girl it the park

The from needs to be completed

I maybe there tomorrow

- New sentences you have not seen before. Can you recognize and correct it this time?
- Intuitively, there are some regularities in the language, "identified" from previous examples, which can be utilized on future examples.
- Two technical ways to formalize this intuition

1: Direct Learning

- Model the problem of text correction as a problem of learning from examples.
- Goal: learn directly how to make predictions.

PARADIGM

- Look at many examples.
- Discover some regularities in the data.
- Use these to construct a prediction policy.
- A policy (a function, a predictor) needs to be specific.

[it/in] rule: if **the** occurs after the target ⇒in

(in most cases, it won't be that simple, though)

2: Generative Model

- Model the problem of text correction as that of generating correct sentences.
- Goal: learn a model of the language; use it to predict.

PARADIGM

- Learn a probability distribution over all sentences
  - In practice: make assumptions on the distribution's type
- Use it to estimate which sentence is more likely.

Pr(I saw the girl it the park) <> Pr(I saw the girl in the park)

[In the same paradigm we sometimes learn a conditional probability distribution]

  - In practice: a decision policy depends on the assumptions

Example: Model of Language

- Model 1: There are 5 characters, A, B, C, D, E
- At any point can generate any of them, according to:

P(A)= 0.3;   P(B) =0.1;   P(C) =0.2;   P(D)= 0.2;   P(E)= 0.1   P(END)=0.1

- Graphical representation: A sunflower model
- A sentence in the language: AAACCCDEABB.
- A less likely sentence: DEEEEBBBBEEEEBBBBEEE

- Given the model, can compute the probability of a sentence, and decide which is more likely.

Example: Model of Language

- Model 2: A probabilistic finite state model.

Start:	$P_S(A)=0.4$; $P_S(B)=0.4$; $P_S(C)=0.2$
From A:	$P_A(A)=0.5$; $P_A(B)=0.3$; $P_A(C)=0.1$; $P_A(S)=0.1$
From B:	$P_B(A)=0.1$; $P_B(B)=0.4$; $P_B(C)=0.4$; $P_B(S)=0.1$
From C:	$P_C(A)=0.3$; $P_C(B)=0.4$; $P_C(C)=0.2$; $P_C(S)=0.1$

- Practical issues:
  - What is the space over which we define the model? Characters? Words? Ideas?
  - How do we acquire the model? Estimation; Smoothing

Learning Paradigms: Comments

- The difference in not along probabilistic/deterministic or statistical/symbolic Lines. Both paradigms can do both.
- The difference is in the basic assumptions underlying the paradigms, and why they work.
  - 1st: Distribution Free: uncover regularities in the past; hope they will be there in the future.
  - 2nd: Know the (type of) probabilistic model of the language (target phenomenon). Use it.
- Direct Learning vs. Generative: major philosophical debate in learning. Interesting computational issues too.

Direct Learning: Formalism

- Goal: discover some regularities from examples and generalize to previously unseen data.

What are the examples we learn from?

- Instance Space X: The space of all examples

$$\boldsymbol{X} = \{0,1\}^n \text{ or } \{0,1\}^\infty$$

How do we represent our hypothesis?

- Hypothesis Space H: Space of potential functions

$$\boldsymbol{h} : \boldsymbol{X} \rightarrow \{0,1\}$$

- Goal: given training data $S \subset X$, find a good $h \in H$

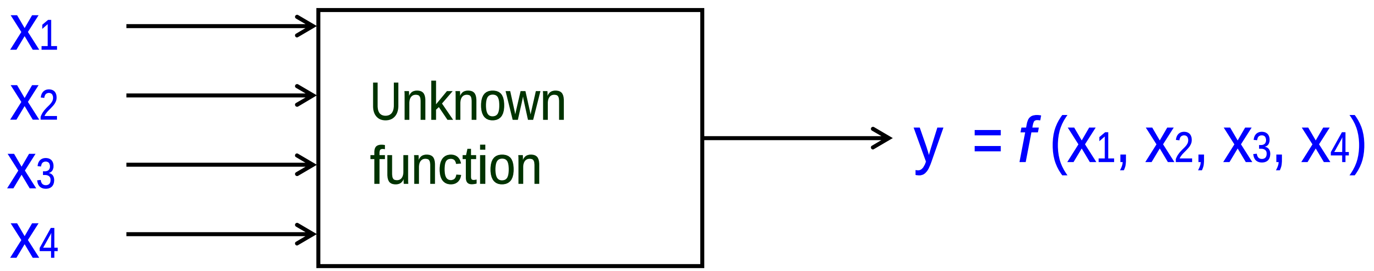
Why Does Learning Work?

- Learning is impossible, unless….
- Outcome of Learning cannot be trusted, unless,…
- How can we quantify the expected generalization?
- Assume h is good on the training data; what can be said on h's performance on previously unseen data?

- These are some of the topics studied in Computational Learning Theory (COLT)

- notice: mode of interaction is also important
- More on all of these in CS346 (CS440 now?)

Learning is impossible, unless...



Given:

Training examples $(x, f(x))$ of unknown function f

Find:

A good approximation to f

Example	x_1	x_2	x_3	x_4	y
1	0	0	1	0	0
2	0	1	0	0	0
3	0	0	1	1	1
4	1	0	0	1	1
5	0	1	1	0	0
6	1	1	0	0	0
7	0	1	0	1	0

Why Does Learning Work (2)?

- Complete Ignorance: There are 2^{16} = 56536 possible functions over four input features.
- We can't figure out which one is correct until we've seen every possible input-output pair.
- Even after seven examples we still have 2^9 possibilities for f
- Is Learning Possible?

Example	x_1	x_2	x_3	x_4	y
	0	0	0	0	?
	0	0	0	1	?
	0	0	1	0	0
	0	0	1	1	1
	0	1	0	0	0
	0	1	0	1	0
	0	1	1	0	0
	0	1	1	1	?
	1	0	0	0	?
	1	0	0	1	1
	1	0	1	0	?
	1	0	1	1	?
	1	1	0	0	0
	1	1	0	1	?
	1	1	1	0	?
	1	1	1	1	?

Hypothesis Space

- Simple Rules: There are only 16 simple conjunctive rules of the form

$$y=x_i \wedge x_j \wedge x_k$$

- Try to learn a function of this form that explains the data. (try it: there isn't).

- m-of-n rules: There are 29 possible rules of the form

"y = 1 if and only if at least m of the following n variables are 1"

(try it, there is).

Bias

- Learning requires guessing a good, small hypothesis class.
- We can start with a very small class and enlarge it until it contains an hypothesis that fits the data.
  - (model selection)
- We could be wrong !

Can We Trust the Hypothesis?

- There is a hidden conjunction the learner is to learn

$$f = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$$

- How many examples are needed to learn it ? How ?

- Protocol:
  - Some random source (e.g., Nature) provides training examples;
  - Teacher (Nature) provides the labels (f(x))
- Not the only possible protocol (membership query; teaching)

<(1,1,1,1,1,1,…,1,1), 1>	<(1,1,1,0,0,0,…,0,0), 0>
<(1,1,1,1,1,0,...0,1,1),1>	<(1,0,1,1,1,0,...0,1,1), 0>
<(1,1,1,1,1,0,...0,0,1),1>	<(1,0,1,0,0,0,...0,1,1), 0>
<(1,1,1,1,1,1,…,0,1), 1>	<(0,1,0,1,0,0,...0,1,1), 0>

Learning Conjunction

- Algorithm: Elimination
- Start with the set of all literals as candidates
- Eliminate a literal if not active (0) in a positive example

<(1,1,1,1,1,1,…,1,1), 1>	$f = x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$
<(1,1,1,0,0,0,…,0,0), 0>	learned nothing
<(1,1,1,1,1,0,...0,1,1),1>	$f = x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{99} \wedge x_{100}$
<(1,0,1,1,0,0,...0,0,1),0>	learned nothing
<(1,1,1,1,1,0,...0,0,1),1>	$f = x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$
<(1,0,1,0,0,0,...0,1,1),0>	
<(1,1,1,1,1,1,…,0,1), 1>	
<(0,1,0,1,0,0,...0,1,1),0>	Final: $f = x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$

Prototypical Learning Scenario

- Instance Space: X
- Hypothesis Space: H (set of possible hypotheses)
- Training instances S:
  - positive and negative examples of the target f
- S: sampled according to a fixed, unknown, probability distribution D over X
- Determine: A hypothesis $h \in H$ such that

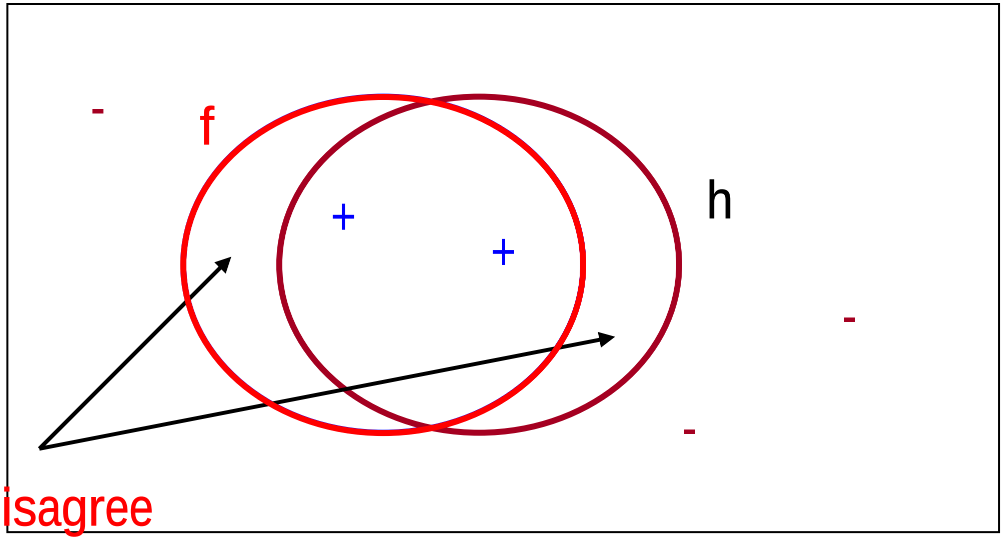
$$h(x) = f(x) \quad \text{for all } x \in S\ ?$$
$$h(x) = f(x) \quad \text{for all } x \in X\ ?$$

- Evaluated on future instances sampled according to D

$$f = x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$$

PAC Learning: Intuition

- Have seen many examples (drawn according to D)
- Since in all the positive examples x_1 was active, it is likely to be active in future positive examples
- If not, in any case, in D, x_1 is active only in relatively few examples, so our error will be small.



f and h disagree

$$\text{Error}_D = \mathbf{Pr}_{x \in D}[f(x) \neq h(x)]$$

Error can be bounded Via Chernoff bounds

A distribution free notion!

Generalization for Consistent Learners

- Claim: The probability that there exists a hypothesis $h \in H$ that:

  (1) is consistent with m examples and

  (2) satisfies $err(h) > \varepsilon$

  is less than $|H|(1-\varepsilon)^m$

Equivalently:

- For any distribution D governing the IID generation of training and test instances, for all $h \in H$, for all $0 < \varepsilon, \delta < 1$, if

$$m > \{\ln(|H|) + \ln(1/\delta)\}/\varepsilon$$

- Then, with probability at least $1-\delta$ (over the choice of the training set of size m),

$$err(h) < \varepsilon$$

Generalization for Consistent Learners

- Claim: The probability that there exists a hypothesis $h \in H$ that:

  (1) is consistent with m examples and

  (2) satisfies $err(h) > \varepsilon$

  is less than $|H|(1-\varepsilon)^m$

- Proof: Let h be such a bad hypothesis.
  - The probability that h is consistent with one example of f is

    $$P_{x \in D}[f(x)=h(x)] < (1-\varepsilon)$$

  - Since the m examples are drawn independently of each other, the probability that h is consistent with m examples is less than $(1-\varepsilon)^m$

  - The probability that *some* hypothesis in H is consistent with m examples is less than $|H|(1-\varepsilon)^m$

Generalization for Consistent Learners

- We want this probability to be smaller than δ, that is:

$$|H|(1-\varepsilon)^m < \delta$$

- And with $(1-x < e^{-x})$

$$\ln(|H|) - m\,\varepsilon < \ln(\delta)$$

What kind of hypothesis spaces do we want ? Large ? Small ?

To guarantee consistency we need $H \supseteq C$. But do we want the smallest H possible ?

- For any distribution D governing the IID generation of training and test instances, for all $h \in H$, for all $0 < \varepsilon, \delta < 1$, if

$$m > \{\ln(|H|) + \ln(1/\delta)\}/\varepsilon$$

- Then, with probability at least $1-\delta$ (over the choice of the training set of size m),

$$err(h) < \varepsilon$$

Generalization (Agnostic Learners)

- In general: we try to learn a concept f using hypotheses in H, but $f \notin H$
- Our goal should be to find a hypothesis $h \in H$, with a small training error:

$$Err_{TR}(h) = P_{x \in S} [f(x) \neq h(x)]$$

- We want a guarantee that a hypothesis with a small training error will have a good accuracy on unseen examples

$$Err_{D}(h) = P_{x \in D} [f(x) \neq h(x)]$$

- Hoeffding bounds characterize the deviation between the true probability of an event and its observed frequency over m independent trials.

$$Pr(p > E(p) + \varepsilon) < \exp\{-2m \varepsilon^2\}$$

(p is the underlying probability of the binary variable being 1)

Generalization (Agnostic Learners)

- Therefore, the probability that an element $h \in H$ will have training error which is off by more than ε can be bounded as follows:

$$\Pr(Err_D(h) > Err_{TR}(h) + \varepsilon) < \exp\{-2m\,\varepsilon^2\}$$

- As in the consistent case: use union bound to get a uniform bound on all H; to get $|H|\exp\{-2m\varepsilon^2\} < \delta$ we have the following generalization bound: a bound on how much will the true error deviate from the observed error.

- For any distribution D generating training and test instance, with probability at least $1-\delta$ over the choice of the training set of size m, (drawn IID), for all $h \in H$

$$\boxed{Err_D(h) < Err_{TR}(h) + \sqrt{\frac{\log|H| + \log(1/\delta)}{2m}}}$$

Summary: Generalization

- Learnability depends on the size of the hypothesis space.
- In the case of a finite hypothesis space:

$$Err_D(h) < Err_{TR}(h) + \sqrt{\frac{\log|H| + \log(1/\delta)}{2m}}$$

- In the case of an infinite hypothesis space

$$Err_D(h) < Err_{TR}(h) + \sqrt{\frac{kVC|H| + \log(1/\delta)}{2m}}$$

- Where VC(H) is the Vapnik-Chernvonenkis of the hypothesis class, a combinatorial measure of its comlexity.

Learning Theory: Summary (1)

- Labeled observations $S = \{ (x,l) \}_{i=1}^{m}$ sampled according to a distribution **D** on $X \times \{0,1\}$

- Goal: to compute a hypothesis $h \in H$ that performs well on future, unseen observations.

- Assumption: test examples are also sampled according to **D** (label is not observed)

Learning Theory:Summary(2) [Why does it work?]

- Look for $\mathbf{h \in H}$ that minimizes the true error

$$\mathbf{Err_D(h) = Pr_{(x,l)\in D}[h(x) \neq l]}$$

- All we get to see is the empirical error

$$\mathbf{Err_S(h) = |\{ x \in S \mid h(x) \neq l \}| / |S|}$$

- Basic theorem: With probability at least $\mathbf{(1\text{-}\delta)}$

$$\mathbf{Err_D(h) < Err_S(h) + \sqrt{[kVC(H) + \ln(1/\delta)]/m}}$$

Practical Lesson

- Use Hypothesis Space with small expressivity
- E.g. prefer to use a function that is linear in the feature space, over higher order functions

$$f(x) = \Sigma_I c_i \chi_i$$

- VC dimension of a linear function of dimension N: is N+1
- Sparsity: If there are a maximum of k active in each example then VC dimension is k+1
- Algorithmic issues: There are good algorithms for linear function; learning higher order functions is computationally hard.

Advances in Theory of Generalization

- VC dimension based bounds are unrealistic.
- The value is mostly in providing quantitative understanding of "why learning works" and what are the important complexity parameters.

- In recent years, this understanding has helped both to
  - drive new algorithms
  - Develop new methods that can actually provide somewhat realistic generalization bounds.

- PAC-Bayes Methods (McAlister, McAlister&Langford)

- Random Projection Methods (Garg, Har-Peled, Roth)

- This method can be shown to have some algorithmic implications.

2: Generative Model

- Model the problem of text correction as that of generating correct sentences.
- Goal: learn a model of the language; use it to predict.

PARADIGM

- Learn a probability distribution over all sentences
  - In practice: make assumptions on the distribution's type
- Use it to estimate which sentence is more likely.

Pr(I saw the girl it the park) <> Pr(I saw the girl in the park)

[In the same paradigm we sometimes learn a conditional probability distribution]

  - In practice: a decision policy depends on the assumptions

Before: Error Driven Learning

- Consider a distribution D over space $X \times Y$
- X - the instance space; Y - set of labels. (e.g. +/-1)
- Given a sample $\{(x,y)\}_1^m$, and a loss function $L(x,y)$
  Find $h \in H$ that minimizes

$$\Sigma_{i=1,m} L(h(x_i),y_i)$$

- L can be:

$L(a,b)=1,\ a \neq b$, o/w $L(a,b) = 0$	(0-1 loss)
$L(a,b)= (a-b)^2$,	(L_2)
$L(a,b)=\exp\{-\ y_i\, h(x_i)\}$	

- Find an algorithm that minimizes average loss; then, we know that things will be okay (as a function of H).

Basics of Bayesian Learning

- Goal: find the best hypothesis from some space H of hypotheses, given the observed data D.
- Define <u>best</u> to be: most <u>probable hypothesis</u> in H
- In order to do that, we need to assume a probability distribution over the class H.
- In addition, we need to know something about the relation between the data observed and the hypotheses (E.g., a coin problem.)
    - As we will see, we will be Bayesian about other things, e.g., the parameters of the model

Basics of Bayesian Learning

- P(h) - the prior probability of a hypothesis h
  Reflects background knowledge; before data is observed. If no information - uniform distribution.

- P(D) - The probability that this sample of the Data is observed. (No knowledge of the hypothesis)

- P(D|h): The probability of observing the sample D, given that the hypothesis h holds

- P(h|D): The posterior probability of h. The probability h holds, given that D has been observed.

Bayes Theorem

$$P(h \mid D) = P(D \mid h) \, {P(h)}/{P(D)}$$

- P(h|D) increases with P(h) and with P(D|h)
- P(h|D) decreases with P(D)

Learning Scenario

$$P(h \mid D) = P(D \mid h) \, {}^{P(h)}\!\big/_{P(D)}$$

- The learner considers a set of candidate hypotheses H (models), and attempts to find the most probable one $h \in H$, given the observed data.

- Such maximally probable hypothesis is called maximum a posteriori hypothesis (MAP); Bayes theorem is used to compute it:

$$\boxed{\begin{aligned} h_{MAP} &= \operatorname{argmax}_{h \in H} P(h \mid D) = \operatorname{argmax}_{h \in H} P(D \mid h) \, {}^{P(h)}\!\big/_{P(D)} \\ &= \operatorname{argmax}_{h \in H} P(D \mid h) P(h) \end{aligned}}$$

Learning Scenario (2)

$$\boxed{h_{MAP} = \operatorname{argmax}_{h \in H} P(h \mid D) = \operatorname{argmax}_{h \in H} P(D \mid h)P(h)}$$

- We may assume that a priori, hypotheses are equally probable

$$P(h_i) = P(h_j), \forall h_i, h_j \in H$$

- We get the Maximum Likelihood hypothesis:

$$\boxed{h_{ML} = \operatorname{argmax}_{h \in H} P(D \mid h)}$$

- Here we just look for the hypothesis that best explains the data

Bayes Optimal Classifier

- How should we use the general formalism?
- What should H be?

- H can be a collection of functions. Given the training data, choose an optimal function. Then, given new data, evaluate the selected function on it.
- H can be a collection of possible predictions. Given the data, try to directly choose the optimal prediction.
- H can be a collection of (conditional) probability distributions.

- Could be different!