Word Sense Disambiguation: Sense Tagging using Machine Learning

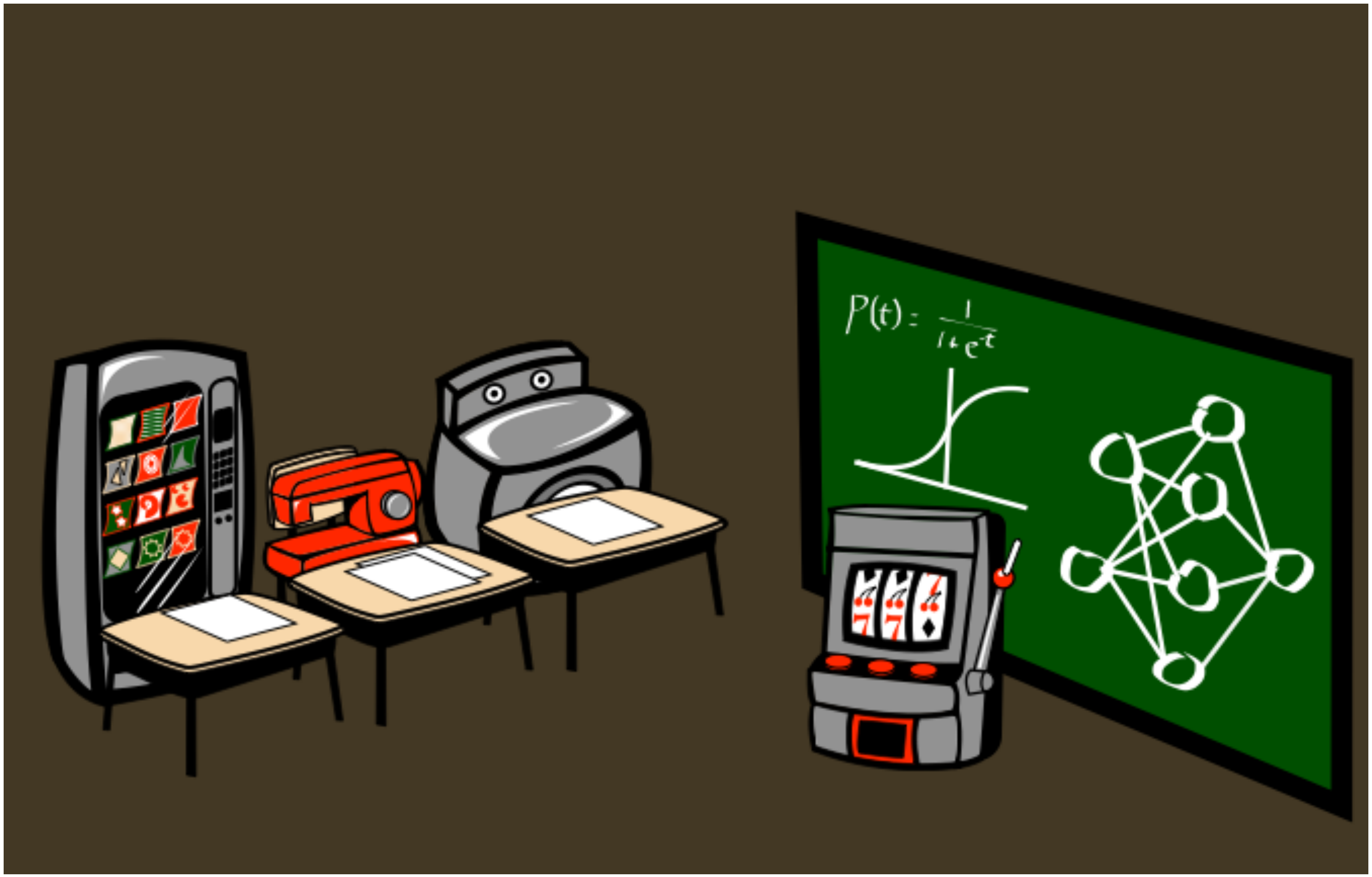
LING 7800/ CSCI 7000

September 25, 2014

Outline

- Supervised Machine Learning
- Probabilities
- Statistical Parsing
- Word Sense Disambiguation

What is Machine Learning?



What is Machine Learning?

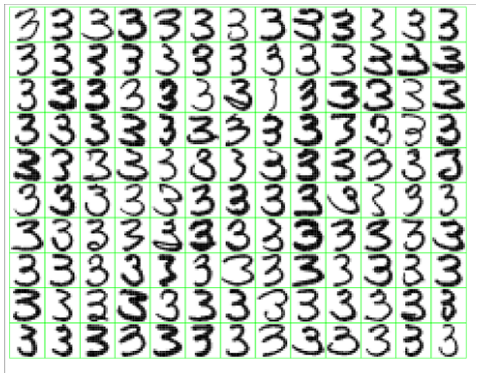
- AKA
  - Pattern Recognition
  - Data Mining
- An application of statistics

What is Machine Learning?

- Programming computers to do tasks that are (often) easy for humans to do, but hard to describe algorithmically.
- Learning from observation
- Creating models that can predict outcomes for unseen data
- Analyzing large amounts of data to discover new patterns

Problems / Application Areas

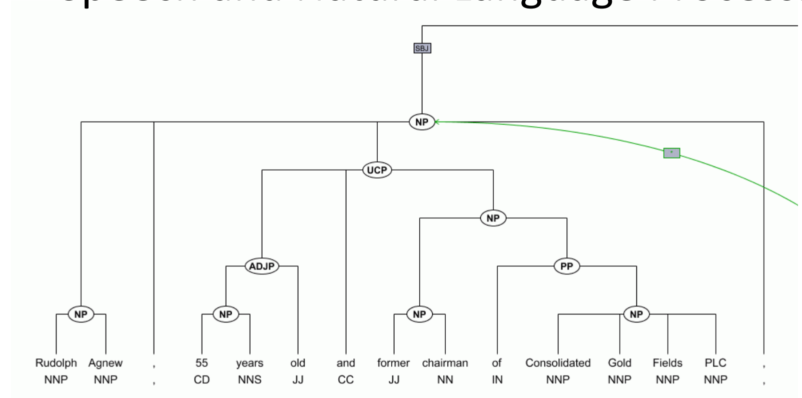
Optical Character Recognition



Face Recognition

Movie Recommendation

Speech and Natural Language Processing



Ok, so where do we start?

- Observations
  - Data! The more the merrier (usually)
- Representations
  - Often raw data is unusable, especially in natural language processing
  - Need a way to represent observations in terms of its properties (features)
- Feature Vector

f_0	f_1								f_n

Feedback to the Learner

- Supervised learning: Learner told immediately whether response behavior was appropriate (training set)
- Unsupervised learning: No classifications are given; the learner has to discover regularities and categories in the data for itself.
- Reinforcement learning: Feedback occurs after a sequence of actions

Supervised Learning

- Given a set of instances, each with a set of features, and their class labels, deduce a function that maps from feature values to labels:

Given:

$$\begin{pmatrix} x_{11}, & x_{12}, x_{13} \dots x_{1m} \\ x_{21}, & x_{22}, x_{23} \dots x_{2m} \\ \dots & \\ x_{n1}, & x_{n2}, x_{n3} \dots x_{nm} \end{pmatrix} \quad \begin{pmatrix} Y_1 \\ y_2 \\ \dots \\ y_n \end{pmatrix}$$

Find:

$$f(\mathbf{x}) = \hat{y}$$

f(**x**) is called a classifier.

The way and/or parameters of f(**x**) is chosen is called a classification model.

Supervised Learning

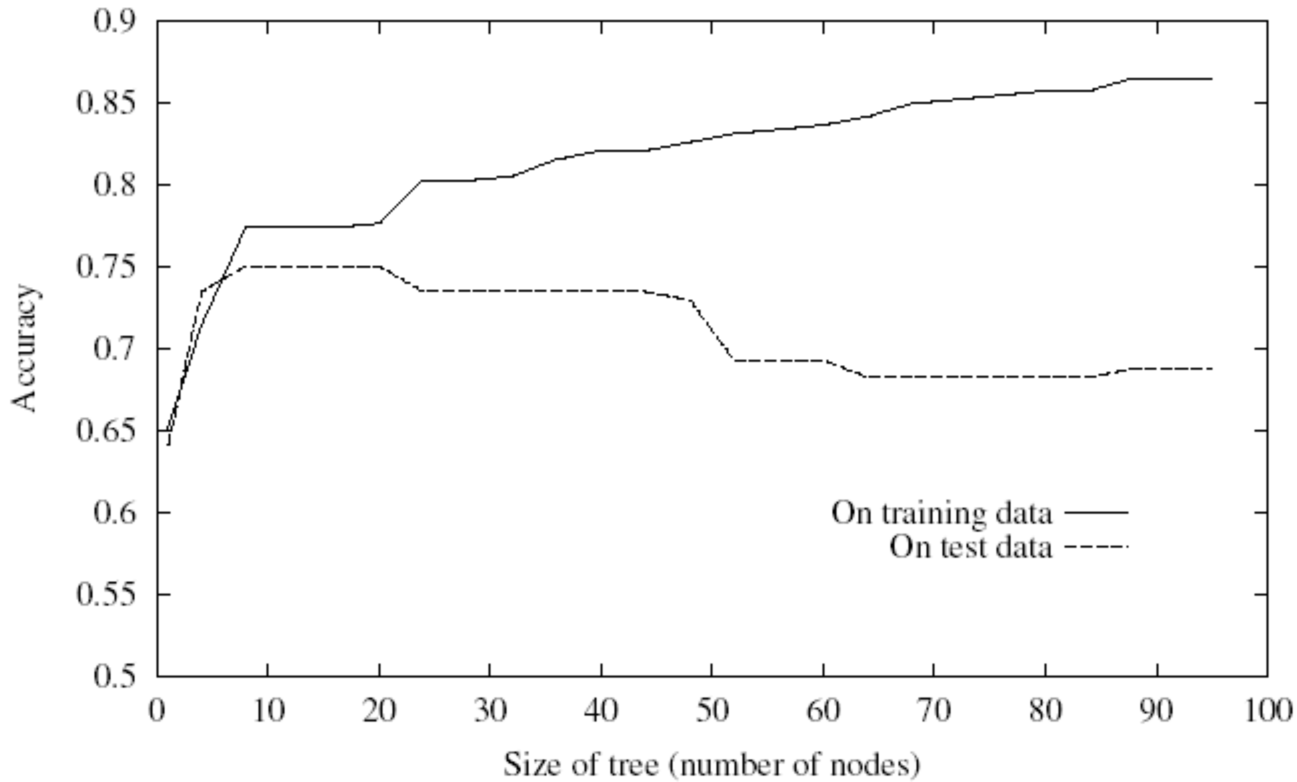
- Stages
  - Train model on data
  - Tune parameters of the model
  - Select best model
  - Evaluate

Measuring Success

- Training set, test set
- The measure of success is not how well the agent performs on the training examples, but how well it performs for new examples.

Evaluation

- Overfitting

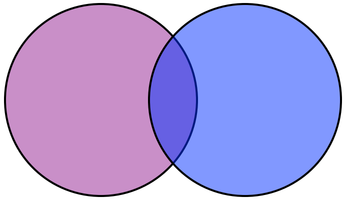


Calculating Probabilities

- When there's a fire, there's a 99% chance that the alarm will go off.
- On any given day, the chance of a fire starting in your house is 1 in 1000.
- What's the chance of there being a fire and your alarm going off tomorrow?

Axioms of Probability

- All probabilities are between 0 and 1
- $P(True) = 1$, $P(False) = 0$
  - P(cavity=true)=.05, P(cavity=false)=.95
- $P(A \lor B) = P(A) + P(B) - P(A \land B)$



derive $P(\neg A) = 1 - P(A)$

Random Variables

- A term whose value isn't necessarily known
  - Discrete r.v – values from a finite set
    - [to, with, from, by, of, for, on, at, …]
  - Boolean r.v. – values from {true,false}
  - Continuous r.v. – numerical values

Probability Calculations

- What do these notations mean?

A	← Boolean Random Variable
$P(A)$	← Unconditional Probability. The notation P(A) is a shortcut for P(A=true).
$P(\neg A)$	shortcut for P(A=false).
$P(A \vee B)$	← Probability of A or B: $P(A) + P(B) - P(A \wedge B)$
$P(A \wedge B)$	← Joint Probability. Probability of A and B together.
$P(A \mid B)$	← Probability of A given that we know B is true.
H	← Non-Boolean Random Variable
$P(H = h)$	← Probability H has some value

Product Rule

$$P(A \wedge B) = P(A|B) * P(B)$$

$$P(A|B) = \frac{P(A \wedge B)}{P(B)}$$

If we can find two of these values someplace (in a chart, from a word problem), then we can calculate the third one.

Using the Product Rule

- When there's a fire, there's a 99% chance that the alarm will go off.

  $P(A \mid F)$

- On any given day, the chance of a fire starting in your house is 1 in 1000.

  $P(F)$

- What's the chance of there being a fire and your alarm going off tomorrow?

  $P(A \wedge F) = P(A \mid F) * P(F)$

- .99 x .001 = .00099

Conditioning

- Sometimes we call the 2nd form of the product rule the "conditioning rule" because we can use it to calculate a conditional probability from a joint probability and an unconditional one.

$$P(A|B) = \frac{P(A \wedge B)}{P(B)}$$

Word Problem

- Out of the 1 million words in some corpus, we know that 9100 of those words are “to” being used as a PREPOSITION.

  P(PREP ∧ “to”)

- Further, we know that 2.53% of all the words that appear in the whole corpus are the word “to”.

  P(“to”)

- If we are told that some particular word in a sentence is “to” but we need to guess what part of speech it is, what is the probability the word is a PREPOSITION?

  What is P(PREP | “to”) ?

  Just calculate: P(PREP|“to”) = P(PREP∧“to”) / P(“to”)

Calculations

- 9100/1,000,000 = .0091 = P(PREP∧"to")
- .0253 = P("to")
- .0091/.0253 = .36 = = P(PREP∧"to") / P("to")

- OR 1M * 2.53% = 25,300

- 9100/25,300 = 36%

Statistical Parsing

- Probabilistic Context Free Grammars
- Finding probable parses
- Lexicalizing probabilities

Simple Context Free Grammar in BNF

S → NP VP
S → Aux NP VP
S → VP
NP → Pronoun
NP → Proper-Noun
NP → Det Nominal
NP → Nominal
Nominal → Noun
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb
VP → Verb NP
VP → Verb NP PP
VP → Verb PP
VP → Verb NP NP
VP → VP PP
PP → Prep NP

Simple Context Free Grammar in BNF

S	→	NP VP	[.80]
S	→	Aux NP VP	[.15]
S	→	VP	[.05]
NP	→	Pronoun	[.35]
NP	→	Proper-Noun	[.30]
NP	→	Det Nominal	[.20]
NP	→	Nominal	[.15]
Nominal	→	Noun	[.75]
Nominal	→	Nominal Noun	[.20]
Nominal	→	Nominal PP	[.05]
VP	→	Verb	[.35]
VP	→	Verb NP	[.20]
VP	→	Verb NP PP	[.10]
VP	→	Verb PP	[.15]
VP	→	Verb NP NP	[.05]
VP	→	VP PP	[.15]
PP	→	Prep NP	[1.0]

Computing Probabilities

S → NP VP [.80]

LHS → RHS

$$P(T,S) = \prod P(RHS_i|LHS_i)$$

Simple Context Free Grammar in BNF

S → NP VP
S → Aux NP VP
S → VP
NP → Pronoun
NP → Proper-Noun
NP → Det Nominal
NP → Nominal
Nominal → Noun
Nominal→ Nominal Noun
Nominal→ Nominal PP
VP → Verb
VP → Verb NP
VP → Verb NP PP
VP → Verb PP
VP → Verb NP NP
VP → VP PP
PP → Prep NP

Stop!

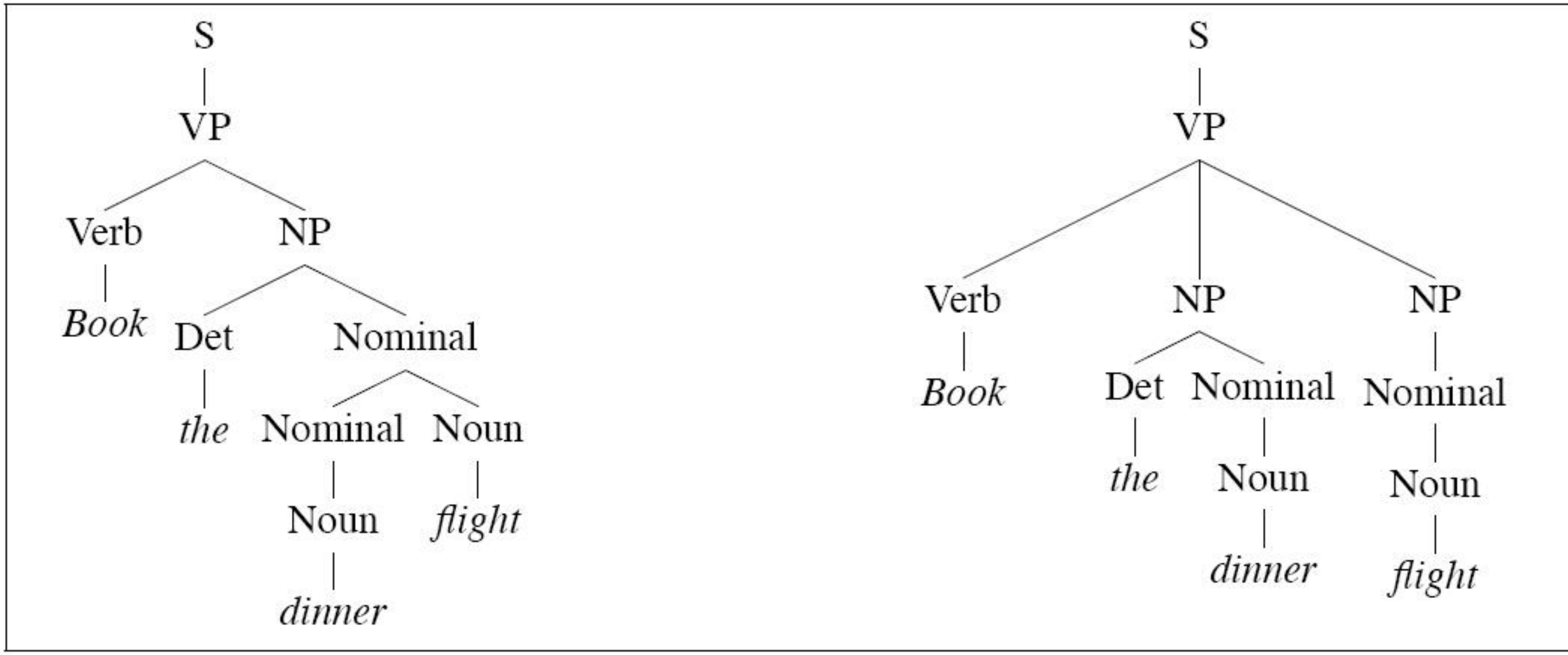
Computing Probabilities

S → NP, VP [.80]
S → VP [.05]
VP → Verb [.35]
Verb → *stop* [.02]
LHS → RHS

Stop!

$$P(T,S) = \prod_{i=1}^{n} P(RHS_i \mid LHS_i)$$

$$P(T,S) = .05 * .35 * .02 = .00035$$



Rules			P	Rules			P
S	→	VP	.05	S	→	VP	.05
VP	→	Verb NP	.20	VP	→	Verb NP NP	.10
NP	→	Det Nominal	.20	NP	→	Det Nominal	.20
Nominal	→	Nominal Noun	.20	NP	→	Nominal	.15
Nominal	→	Noun	.75	Nominal	→	Noun	.75
				Nominal	→	Noun	.75
Verb	→	book	.30	Verb	→	book	.30
Det	→	the	.60	Det	→	the	.60
Noun	→	dinner	.10	Noun	→	dinner	.10
Noun	→	flights	.40	Noun	→	flights	.40

Computing Probabilities

$$P(T_{left}) = .05 * .20 * .20 * .20 * .75 * .30 * .60 * .10 * .40 = 2.2x10^{-6}$$

$$P(T_{right}) = .05 * .10 * .20 * .15 * .75 * .75 * .30 * .60 * .10 * .40 = 6.1x10^{-7}$$

Subcategorization Frequencies

- The women kept the dogs on the beach.
  - Where keep? Keep on beach 95%
    - NP XP 81%
  - Which dogs? Dogs on beach 5%
    - NP 19%
- The women discussed the dogs on the beach.
  - Where discuss? Discuss on beach 10%
    - NP PP 24%
  - Which dogs? Dogs on beach 90%
    - NP 76%

Ford, Bresnan, Kaplan 82, Jurafsky 98, Roland,Jurafsky 99

Conditioning on lexical items

S → NP VP [.80]
S → Aux NP VP [.15]
S → VP [.05]
NP → Pronoun [.35]
NP → Proper-Noun [.30]
NP → Det Nominal [.20]
NP → Nominal [.15]
Nominal → Noun [.75]
Nominal → Nominal Noun [.20]
Nominal → Nominal PP [.05]
VP → Verb [.87] *{sleep, cry, laugh}*
VP → Verb NP [.03]
VP → Verb NP PP [.00]
VP → Verb PP [.05]
VP → Verb NP NP [.00]
VP → VP PP [.05]
PP → Prep NP [1.0]

Lexicalizing Probabilities

S → NP VP [.80]
S → Aux NP VP [.15]
S → VP [.05]
NP → Pronoun [.35]
NP → Proper-Noun [.30]
NP → Det Nominal [.20]
NP → Nominal [.15]
Nominal → Noun [.75]
Nominal → Nominal Noun [.20]
Nominal → Nominal PP [.05]
VP → Verb [.30]
VP → Verb NP [.55] *{break,split,crack..}*
VP → Verb NP PP [.05]
VP → Verb PP [.05]
VP → Verb NP NP [.00]
VP → VP PP [.05]
PP → Prep NP [1.0]

Training data for Statistical Parsers

- How does the computer learn the probabilities?
- Lots and lots of parsed sentences
- 50K WSJ sentences

Outline

- Supervised Machine Learning
- Probabilities
- Statistical Parsing
- Word Sense Disambiguation

Naïve Bayes

- Assumes that when class label is known the features are independent:

$$f(\mathbf{x}) = \underset{y}{\arg\max}\, p(y) \prod_{i=1}^{m} p(x_i | y)$$

$$P(T, S) = \prod_{i=1}^{n} P(RHS_i \mid LHS_i)$$

Naïve Bayes Dog vs Cat Classifier

- 2 features: weight & how frequently it chases a mouse

mouse chase	weight	label
0.7	55	dog
0.05	15	dog
0.2	100	dog
0.25	42	dog
0.2	32	dog
0.6	25	cat
0.2	15	cat
0.55	8	cat
0.15	12	cat
0.4	15	cat

Given an animal that weighs no more than 20 lbs and chases a mouse at least 21% of time, is it a cat or dog?

$$f(dog, w \leq 20, m \geq .21) =$$
$$p(dog)p(w \leq 20 \mid dog)p(m \geq 0.21 \mid dog) =$$
$$0.5 \times 0.2 \times 0.4 = 0.04$$

$$f(cat, w \leq 20, m \geq .21) =$$
$$p(cat)p(w \leq 20 \mid cat)p(m \geq 0.21 \mid cat) =$$
$$0.5 \times 0.8 \times 0.6 = 0.24$$

So, it’s a cat! In fact, naïve Bayes is 83.3% certain it’s a cat over a dog.

Word Sense Disambiguation

- Given an occurrence of a word, decide which sense, or meaning, was intended.
- Example, *run*
  - *run1:* move swiftly (*I ran to the store.*)
  - *run2:* operate (*I run a store*.)
  - *run3:* flow (*A river runs through the farm.)*
  - *run4:* length of torn stitches (*Her stockings had a run*.)

Word Sense Disambiguation

- Categories
  - Use word sense labels (*run1, run2, etc*.)

- Features – describe context of word
  - *near(w)* : is the given word near word w?
  - *pos*: word's part of speech
  - *left(w):* is word immediately preceded by w?
  - etc.

Word Sense Disambiguation

- Categories
  - Use word sense labels (*run1, run2, etc*.)

- Features – describe context of word
  - *near(w)* : is the given word near word [***race, river, stocking***]?
  - *pos*: word's part of speech [**noun or verb**]
  - *left(w):* is word immediately preceded by w?
  - etc.

WSD: Sample Training Data Features

POS	near(race)	near(river)	near(stockings)	Sense#
Verb	No	No	No	run1
Verb	No	No	No	run2
Verb	No	Yes	No	run3
Noun	No	No	Yes	run4

run1: move swiftly (*I ran to the store.*)
run2: operate (*I run a store*.)
run3: flow (*A river runs through the farm.)*
run4: length of torn stitches (*Her stockings had a run.*)

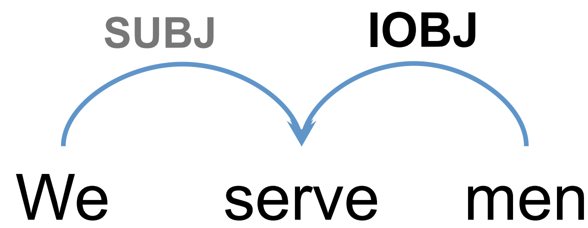
Word Sense Disambiguation

- Given an occurrence of a word, decide which sense, or meaning, was intended.
- Example, *run*
  - *run1:* move swiftly *(I ran to the store. John ran in the race by the river. She's running in heels and stockings!*)
  - *run2:* operate (*I run a store. He runs a river rafting guide service that has an annual race*)
  - *run3:* flow (*A river runs through the farm.)*
  - *run4:* length of torn stitches (*Her stockings had a run. Her sweater had a run.*)

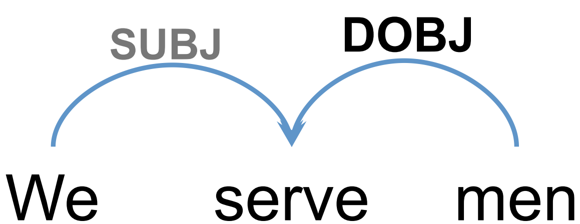
WSD: More instances

POS	near(race)	near(river)	near(stockings)	Sense#
Noun	No	No	No	run4
Verb	No	No	No	run1
Verb	No	Yes	No	run3
Noun	Yes	Yes	Yes	run4
Verb	No	No	Yes	run1
Verb	Yes	Yes	No	run2
Verb	No	No	No	run2
Noun	No	No	Yes	run4

Maybe more kinds of features would help?



We serve food to men.
We serve our community.
serve —IndirectObject→ men



We serve organic food.
We serve coffee to connoiseurs.
serve —DirectObject→ men

More Features for WSD

Dang & Palmer, SIGLEX-02

- Maximum entropy framework, $p(sense|context)$
- Contextual Linguistic Features
  - Topical feature for W, keywords
    - (determined automatically)
  - Local syntactic features for W:
    - presence of subject, complements, passive?
    - words in subject, complement positions, particles, preps,..
  - Local semantic features for W:
    - Semantic class info from WordNet (synsets, etc.)
    - Named Entity tag (PERSON, LOCATION,..) for proper Ns
    - Words within +/- 2 word window

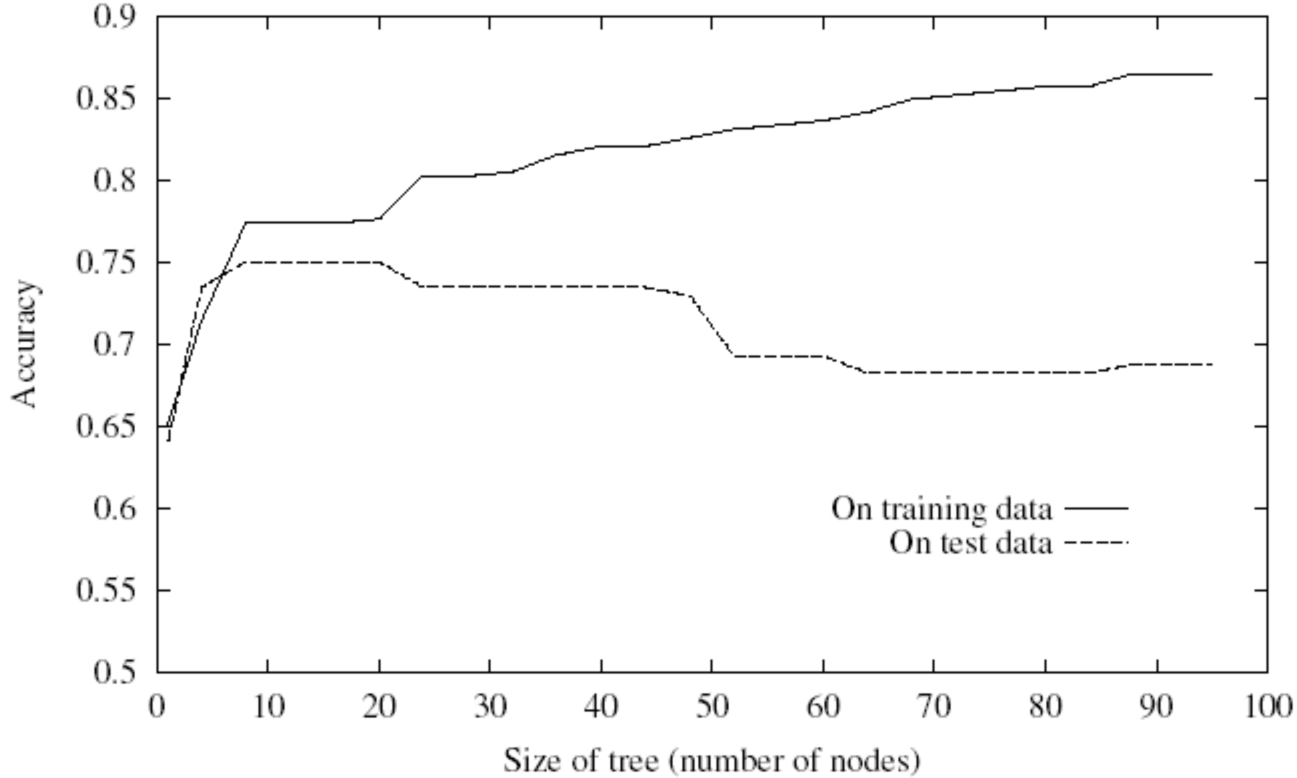
Contribution of features to result

Dang & Palmer, SIGLEX-02

- Maximum entropy framework, $p(sense|context)$
- Contextual Linguistic Features
  - Topical feature for W, keywords : **+2.5%,**
    - (determined automatically)
  - Local syntactic features for W: **+1.5 to +5%,**
    - presence of subject, complements, passive?
    - words in subject, complement positions, particles, preps,..
  - Local semantic features for W: **+6%**
    - Semantic class info from WordNet (synsets, etc.)
    - Named Entity tag (PERSON, LOCATION,..) for proper Ns
    - Words within +/- 2 word window

Evaluation

- Overfitting



WordNet: - call, 28 senses, Senseval2 groups (engineering!)

WN5, WN16,WN12
Loud cry

WN15 WN26
Bird or animal cry

WN3 WN19
WN1 WN22
Label

WN4 WN 7 WN8 WN9
Request

WN20 WN25
Call a loan/bond

WN18 WN27
Challenge

WN2 WN 13
Phone/radio
WN28

WN6 WN23
Visit

WN17 , WN 11

WN10, WN14, WN21, WN24,
Bid

Grouping improved scores: ITA 82%, MaxEnt WSD 69%

Palmer, Dang, Fellbaum,, NLE07

- *call*: 31% of errors due to confusion between senses within same group 1:
  - name, call -- (assign a specified, proper name to; *They named their son David*)
  - call -- (ascribe a quality to or give a name of a common noun that reflects a quality; *He called me a bastard*)
  - call -- (consider or regard as being;*I would not call her beautiful)*
  - *75% with training and testing on grouped senses vs.*
  - *43% with training and testing on fine-grained senses*

Automatic sense tagging

- Where does the sense tagger get the information it needs to apply all these criteria?