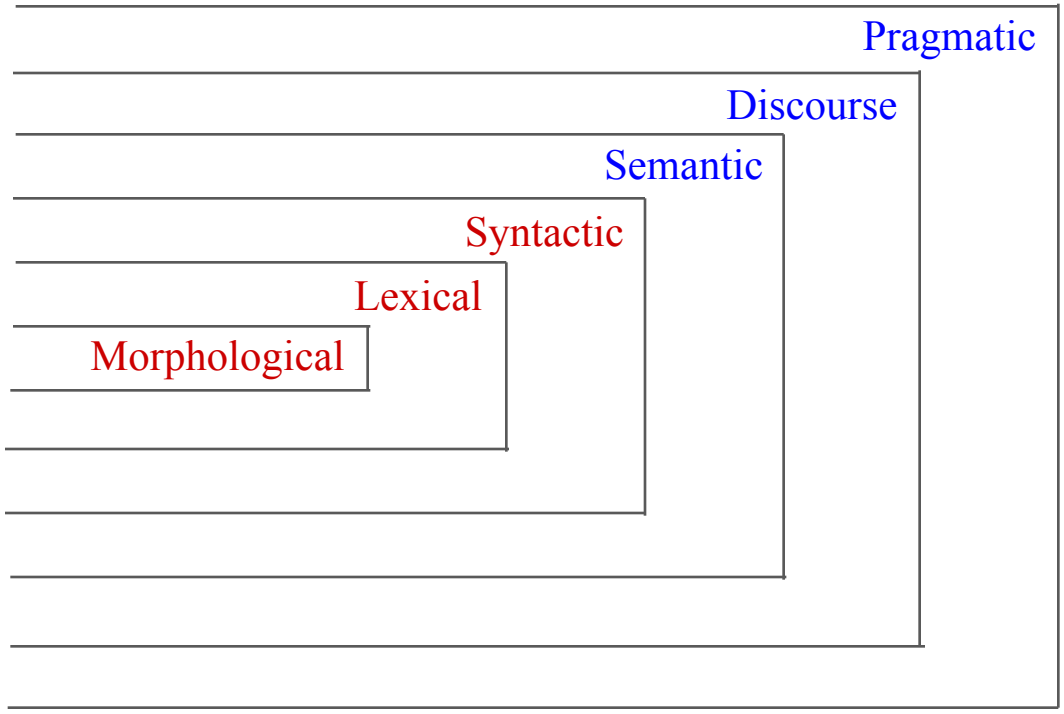
Part-of-Speech (POS) Tagging: Introduction

School of Information Studies
Syracuse University

Levels of Language

POS tags are assigned to words but may be determined by adjacent words.



What Is Part-of-Speech Tagging?

The general purpose of a part-of-speech tagger is to associate each word in a text with its correct lexical-syntactic category (represented by a tag).

- Example using tags from the Penn Treebank POS tag set

03/14/1999 (AFP)… the extremist Harkatul Jihad group, reportedly backed by Saudi dissident Osama bin Laden…

… the|DT extremist|JJ Harkatul|NNP Jihad|NNP group|NN,|, reportedly|RB backed|VBD by|IN Saudi|NNP dissident|NN Osama|NNP bin|NN Laden|NNP…

What Are Parts of Speech?

They are approximately eight traditional basic word classes, sometimes called syntactic classes or types.

These are the ones taught in grade school grammar.

- N noun *chair, bandwidth, pacing*
- V verb *study, debate, munch*
- ADJ adjective *purple, tall, ridiculous (includes articles)*
- ADV adverb *unfortunately, slowly*
- P preposition *of, by, to*
- CON conjunction *and, but*
- PRO pronoun *I, me, mine*
- INT interjection *um*

For example, see the shows from *Schoolhouse Rock* on grammar.

Open and Closed Classes of Words

Open classes—can add words to these basic word classes:

- Nouns, verbs, adjectives, adverbs
- Some kinds of numbers

- Every known human language has nouns and verbs
- Open class words are more often borrowed from one language to another

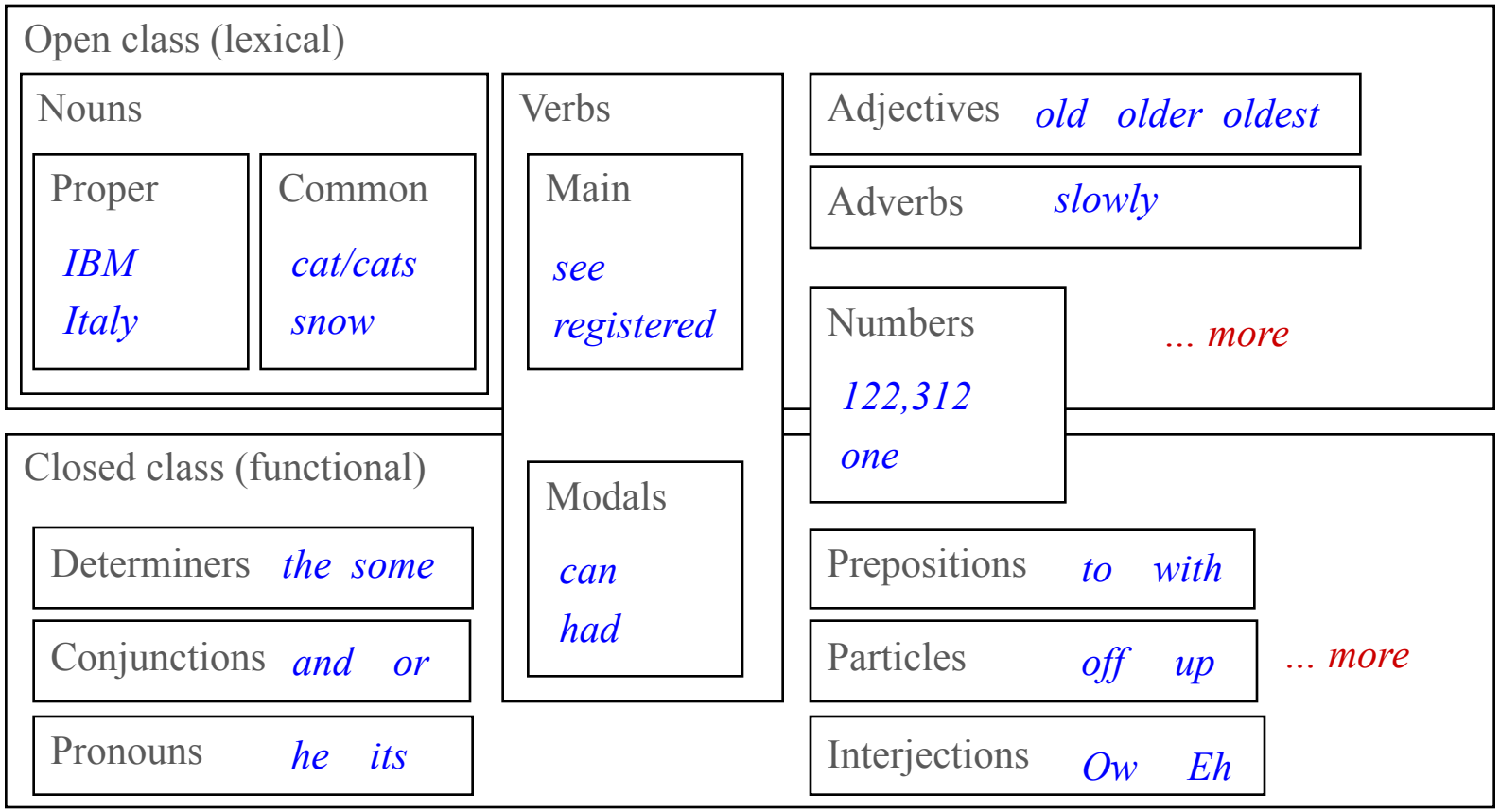
Closed classes—words are not added to these classes:

- Determiners, pronouns, prepositions, particles, auxiliary verbs

- Closed-class words are often function words that have structuring uses in grammar
- Differ more from language to language than open-class words

Open and Closed Classes

We may want to make more distinctions than eight classes.



POS Tag Sets

School of Information Studies
Syracuse University

Possible Tag Sets for English

Kucera and Francis (Brown Corpus)—87 POS tags

C5 (British National Corpus) —61 POS tags

- Tagged by Lancaster's UCREL project

Penn Treebank—45 POS tags

- Most widely used of the tag sets today

Penn Treebank

A corpus containing:

- Over 1.6 million words of hand-parsed material from the Dow Jones News Service, plus an additional 1 million words tagged for part of speech.
- The first fully parsed version of the Brown Corpus, which has also been completely retagged using the Penn Treebank tag set
- Source code for several software packages that permits the user to search for specific constituents in tree structures

Costs $1,250 to $2,500 for research use

Separate licensing needed for commercial use

Word Classes: Penn Treebank Tag Set

Tag	Description	Example	Tag	Description	Example
CC	coordinating conjunction	*and, but, or*	SYM	symbol	*+, %, &*
CD	cardinal number	*one, two, three*	TO	"to"	*to*
DT	determiner	*a, the*	UH	interjection	*ah, oops*
EX	existential "there"	*there*	VB	verb, base form	*eat*
FW	foreign word	*mea culpa*	VBD	verb, past tense	*ate*
IN	preposition/subordinating conjunction	*of, in, by*	VBG	verb, gerund	*eating*
JJ	adjective	*yellow*	VBN	verb, past participle	*eaten*
JJR	adjective, comparative	*bigger*	VBP	verb, non-3sg present	*eat*
JJS	adjective, superlative	*wildest*	VBZ	verb, 3sg present	*eats*
LS	list item marker	*1, 2, One*	WDT	wh- determiner	*which, that*
MD	modal	*can, should*	WP	wh- pronoun	*what, who*
NN	noun, singular or mass	*llamas*	WP$	possessive wh-	*whose*

Word Classes: Penn Treebank Tag Set

Tag	Description	Example	Tag	Description	Example
NNS	noun, plural	*llamas*	WRB	wh- adverb	*how, where*
NNP	proper noun, singular	*IBM*	$	dollar sign	*$*
NNPS	proper noun, plural	*Carolinas*	#	pound sign	#
PDT	predeterminer	*all, both*	“	left quote	‘ or “
POS	possessive ending	*’s*	”	right quote	’ or ”
PRP	personal pronoun	*I, you, he*	(	left parenthesis	[, (, {, <
PRP$	possessive pronoun	*your, one’s*	)	right parenthesis	],), }, >
RB	adverb	*quickly, never*	,	comma	,
RBR	adverb, comparative	*fastest*	.	sentence – final punctuation	. ! ?
RBS	adverb, superlative	*fastest*	:	mid-sentence punctuation	: ; … - -
RP	particle	*up, off*			

Examples of Penn Treebank Tagging

The/DT grand/JJ jury/NN commented/VBD
on/IN a/DT number/NN of/IN other/JJ topics/NNS ./.

Book/VB that/DT flight/NN ./.

Does/VBZ that/DT flight/NN serve/VB dinner/NN ?/?

POS Classes of English Words

School of Information Studies
Syracuse University

Nouns and Numerals

Nouns: people, places, things

- Nouns can be further subdivided
  - Proper nouns are usually capitalized in English (*Boulder, Granby, Eli Manning*)
  - Common nouns
    - Count (have plurals, get counted: *chair/chairs, one chair, two chairs*) vs.
    - Mass (don't get counted: *furniture, snow, salt, communism*)
- Properties of nouns
  - Can be preceded by a determiner, adjectives, etc.
    - (But there are differences for proper and common nouns)
  - Can take possessives (*IBM's annual review*)

Numerals, ordinals: *one, two, three, third, …*

Verbs and Adjectives

Verbs are words for actions and processes

- Run, vote, believe, etc.
- Verbs may have inflections for:
  - Third-person singular (*she eats*)
  - Non–third-person singular (*I eat, you eat*)
  - Progressive (*eating*)
  - Past participles (*eaten*)

Adjectives have words for descriptions, properties, and qualities; for concepts of color, age, value, etc.

- (*small, red, old, good*)
- include comparatives (*old, older, oldest*)

Adverbs

This class is more of a hodgepodge, both syntactically and semantically.

- Usually modify verbs but may also modify other adverbs and verb phrases
- Directional, locational: *downhill, home, here*
- Degree: *very, somewhat, extremely*
- Manner: *slowly, delicately*
- Temporal: *yesterday, Monday*

"Unfortunately, John walked home extremely slowly yesterday."

Determiners and Conjunctions

Determiners often occur with nouns, marking the beginning of the noun phrase.

- Articles: *a, an, the* are the most common of the determiners, often the most frequently occurring words in a corpus
- Other determiners include *this, that, those*

Conjunctions connect phrases, clauses, or sentences.

- Can indicate relations between them
- Coordinating (*and, or, but*) join elements of equal status
- Subordinating may indicate a lesser status
- Other examples: *like, though, than, when, before, that, which, while, heretofore, notwithstanding*

Prepositions

Prepositions occur before noun phrases and show relationships between that noun and other words.

- Spatial and temporal relations: *over the river and through the woods, before then, with gusto, at the house*
- May show other relations, such as marking the agent of a verb: *Hamlet was written by Shakespeare.*
- Others: *of, among, on, toward, from,* etc.

Charts from Jurafsky and Martin text

English Single-Word Particles

Definition of the term "particle" in linguistics varies

- Primarily words that used to provide shades of meaning to other words, particularly verbs
  - The use of "over" gives a slightly different meaning to "turned" in *she turned the paper over*
- Particles can resemble a preposition or an adverb in occurring with a verb
- Others: *up, down, on, off, alongside, between, together*, etc.

Pronouns

Pronouns are a kind of shorthand for referring to a noun phrase, entity, or event.

- Personal
  - *he, I, it, me*
- Possessive
  - *ours, mine, her, their*
- Demonstrative
  - *that, those*
- Indefinite
  - *one, neither, somebody, both*
- Wh- pronouns
  - *what, who, whom, whoever*
- Reflexive
  - *myself, ourselves*

Auxiliary Verbs

Auxiliary, or helping verbs, are used to express semantic features of the main verb, including tenses such as past or future, whether it is completed or negated.

- *He had run the race before.*
- *I should have been running.*

- The verbs "do" and "have" have particular roles in their various inflected forms.
- Copula verbs are usually forms of the verb "be" and link the subject with other forms.

Modal verbs are the auxiliary verbs that express likelihood or ability.

- *can, might, must, could, should, …*

Unique Function Words

Interjections: *oh, hey, alas, uh, um*

Negatives: *no, not*

Politeness markers: *please, thank you*

Existential there: *There are two on the table.*

POS Tagging: Introduction

School of Information Studies
Syracuse University

Why Is Part-of-Speech Tagging Hard?

The POS tagging task is to assign a sequence of tags to a sequence of words (usually a sentence).

- Or can be viewed as assigning a tag to a word in the context of a sequence

Words may be ambiguous in different ways.

- A word may have multiple meanings as the same part of speech
  - *file*: **noun**, a folder for storing papers
  - *file*: **noun**, instrument for smoothing rough edges
- A word may function as multiple parts of speech
  - A *round* table: **adjective**
  - A *round* of applause: **noun**
  - To *round* out your interests: **verb**
  - To work the year *round*: **adverb**

Why Is Part-of-Speech Tagging Needed?

May be useful to know what function the word plays, instead of depending on the word itself

Internally, next higher levels of NL processing:

- Phrase bracketing
  - Can write regexps like (Det) Adj* N+ over the output for phrases
- Parsing
  - Using the word class instead of the word can speed up parsing
- Semantics
  - Word classes can help determine word semantics

Why Is Part-of-Speech Tagging Needed?

Word classes (POS tags) are also useful in applications

- Speech synthesis: text-to-speech (how do we pronounce “lead” or “object”?)
- Information retrieval: selection of high-content words
- Word-sense disambiguation
- Sentiment detection: selection of high-opinion or emotion words

Overview of Approaches

Stochastic Approaches

- Refers to any approach which incorporates frequencies or probabilities
- Requires a tagged corpus to learn frequencies of words with POS tags

- ***N*-gram taggers:** uses the context of (a few) previous tags
- **Hidden Markov Model (HMM) taggers:** uses the context of the entire sequence of words and previous tags
  - This technique has been the most widely used of modern taggers but has the problem of unknown words

Classification Taggers

- Uses morphology of word and (a few) surrounding words
- Helps solve the problem of unknown words

Word Class Ambiguity (in the Brown Corpus)

Recall that words often have more than one word class: another example is the word *this*

- *This* is a nice day = PRP (pronoun)
- *This* day is nice = DT (determiner)
- You can go *this* far = RB (adverb)

Degree of ambiguity in English

- 40% of word tokens are ambiguous
- 11.5% of word types are ambiguous
  - Unambiguous (one tag): 35,340
  - Ambiguous (two to seven tags): 4,100

The word "still" has seven tags

2 tags	3,760
3 tags	264
4 tags	61
5 tags	12
6 tags	2
7 tags	1

(Derose, 1988)

N-Gram Approach

N-gram approach to probabilistic POS tagging

- Calculates the probability of a given sequence of tags occurring for a sequence of words
- The best tag for a given word is determined by the (already calculated) probability that it occurs with the n previous tags
- May be bigram, trigram, etc.

Presented here as an introduction to HMM tagging

- And given in more detail in the NLTK
- In practice, bigram and trigram probabilities have the problem that the combinations of words are sparse in the corpus
- Combine the taggers with a back-off approach

N-Gram Tagging

Initialize a tagger by learning probabilities from a tagged corpus.

$word_{n-1}$	…	$word_{-2}$	$word_{-1}$	word	
tag_{n-1}		…	tag_{-2}	tag_{-1}	XX

- Probability that the sequence … tag_{-2} tag_{-1} word gives tag XX
- Note that initial sequences will include a start marker as part of the sequence

Tagger algorithm: Use the tagger to tag word sequences (usually of length 2–3) with unknown tags.

- Sequence through the words
  - To determine the POS tag for the next word, use the previous *n*–1 tags and the word to look up probabilities and use the highest probability tag

POS Tagging: HMM Probabilities

School of Information Studies
Syracuse University

Need Longer Sequence Tagging

A more comprehensive approach to tagging considers the entire sequence of words

- *Secretariat is expected to race tomorrow*

What is the best sequence of tags that corresponds to this sequence of observations?

Probabilistic view

- Consider all possible sequences of tags
- Out of this universe of sequences, choose the tag sequence that is most probable given the observation sequence of *n* words $w_1 \ldots wn$

Thanks to Jim Martin's online class slides for the examples and equation typesetting in this section on HMMs.

Road to HMMs

We want, out of all sequences of n tags $t_1 \ldots t_n$, the single tag sequence such that $P(t_1 \ldots t_n | w_1 \ldots w_n)$ is highest.

- That is, find the tag sequence $t_1 \ldots t_n$ that maximizes the probability that sequence should tag the word sequence $w_1 \ldots w_n$
- We use this notation:

$$\hat{t}_1^n = \underset{t_1^n}{\operatorname{argmax}} P(t_1^n | w_1^n)$$

Hat ^ means "our estimate of the best one"

Argmax$_x$ $P(x)$ means "the x such that $P(x)$ is maximized"

Road to HMMs

This equation (from the previous slide) is guaranteed to give us the best tag sequence

$$\hat{t}_1^n = \underset{t_1^n}{\operatorname{argmax}} P(t_1^n | w_1^n)$$

But how to make it operational?

How to compute this value?

Intuition of Bayesian classification:

- Use Bayes' rule to transform into a set of other probabilities that are easier to compute

Thomas Bayes 1701–1761

Using Bayes' Rule

Bayes' rule: $$P(x|y) = \frac{P(y|x)P(x)}{P(y)}$$

Apply Bayes' rule: $$\hat{t}_1^n = \underset{t_1^n}{\text{argmax}} \frac{P(w_1^n|t_1^n)P(t_1^n)}{P(w_1^n)}$$

Note that this is using the conditional probability, given a tag sequence, what is the most likely word sequence with those tags.

- Eliminate the denominator, as it is the same for every sequence.

$$\hat{t}_1^n = \underset{t_1^n}{\text{argmax}} P(w_1^n|t_1^n)P(t_1^n)$$

Likelihood and Prior

- To further simplify. look at the two probabilities

$$\hat{t}_1^n = \underset{t_1^n}{\text{argmax}} \overbrace{P(w_1^n|t_1^n)}^{\text{likelihood}} \overbrace{P(t_1^n)}^{\text{prior}}$$

- Likelihood: Assume that the probability of the word depends only on its tag.

$$P(w_1^n|t_1^n) \approx \prod_{i=1} P(w_i|t_i)$$

- Prior: Make the bigram assumption that the tag only depends on the previous tag.

$$P(t_1^n) \approx \prod_{i=1}^{n} P(t_i|t_{i-1})$$

$$\hat{t}_1^n = \underset{t_1^n}{\text{argmax}}\, P(t_1^n|w_1^n) \approx \underset{t_1^n}{\text{argmax}} \prod_{i=1}^{n} P(w_i|t_i)P(t_i|t_{i-1})$$

Two Sets of Probabilities (1)

Tag transition probabilities $P(t_i|t_i-1)$ (priors)

Given a tag t_i-1, how likely is it that it is followed by t_i?

- Determiners likely to precede adjectives and nouns
  - That/DT flight/NN and The/DT yellow/JJ hat/NN
  - So we expect P(NN|DT) and P(JJ|DT) to be high
- Compute P(NN|DT) by counting in a labeled corpus:

$$P(t_i|t_{i-1}) = \frac{C(t_{i-1}, t_i)}{C(t_{i-1})}$$

Count of DT NN sequence

$$P(NN|DT) = \frac{C(DT, NN)}{C(DT)} = \frac{56,509}{116,454} = .49$$

Two Sets of Probabilities (2)

Word likelihood probabilities $P(w_i|t_i)$

Given a tag t_i, how likely is it that it is tagging word w_i

- VBZ (3sg present verb) likely to be "is"
- Compute $P(\text{is}|\text{VBZ})$ by counting in a labeled corpus:

$$P(w_i|t_i) = \frac{C(t_i, w_i)}{C(t_i)}$$

Count of "is" tagged with VBZ

$$P(is|VBZ) = \frac{C(VBZ, is)}{C(VBZ)} = \frac{10,073}{21,627} = .47$$

An Example: The Word "Race"

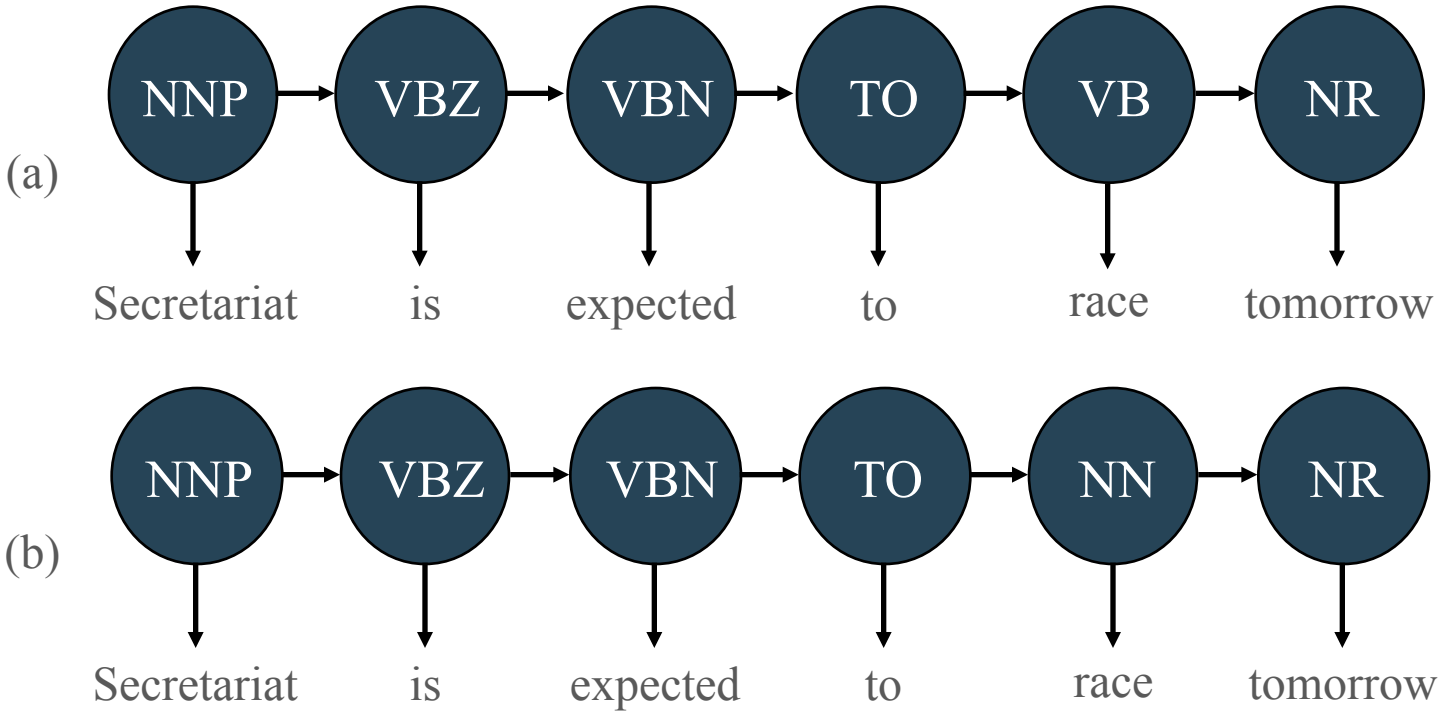
The word "race" can occur as a verb or as a noun.

- Secretariat/NNP is/VBZ expected/VBN to/TO **race**/VB tomorrow/NR
- People/NNS continue/VB to/TO inquire/VB the/DT reason/NN for/IN the/DT **race**/NN for/IN outer/JJ space/NN

How do we pick the right tag?

Disambiguating "Race"

Which tag sequence is most likely?



Example

The equations only differ in "to race tomorrow."

$P(\text{NN}|\text{TO}) = .00047$

$P(\text{VB}|\text{TO}) = .83$

$P(\text{race}|\text{NN}) = .00057$

$P(\text{race}|\text{VB}) = .00012$

$P(\text{NR}|\text{VB}) = .0027$

$P(\text{NR}|\text{NN}) = .0012$

The tag transition probabilities $P(\text{NN}|\text{TO})$ and $P(\text{VB}|\text{TO})$

Lexical likelihoods from the Brown Corpus for "race" given a POS tag NN or VB

Tag sequence probability for how likely an adverb NR occurs given the previous tag verb VB or noun NN

$P(\text{VB}|\text{TO})P(\text{NR}|\text{VB})P(\text{race}|\text{VB}) = .00000027$

$P(\text{NN}|\text{TO})P(\text{NR}|\text{NN})P(\text{race}|\text{NN}) = .00000000032$

So, we (correctly) choose the verb tag.

HMM Algorithm

We have seen an example of how the HMM algorithm can assign a tag, given the two sets of probabilities for the sequence of words.

We can compute the two sets of probabilities by counting frequencies in a corpus.

But what algorithm actually produces sequences of tags?

- The optional lecture explains how an optimization algorithm, the Viterbi algorithm, finds the best tag sequences

POS Tagging: HMM Algorithm

School of Information Studies
Syracuse University

Hidden Markov Models

Recall that we estimated the best probable tag sequence for a given sequence of words as:

$$\hat{t}_1^n = \underset{t_1^n}{\operatorname{argmax}} P(t_1^n | w_1^n) \approx \underset{t_1^n}{\operatorname{argmax}} \prod_{i=1}^{n} P(w_i | t_i) P(t_i | t_{i-1})$$

with the word likelihood x the tag transition (prior) probabilities.

When we evaluated the probabilities by hand for a sentence, we could pick the optimum tags.

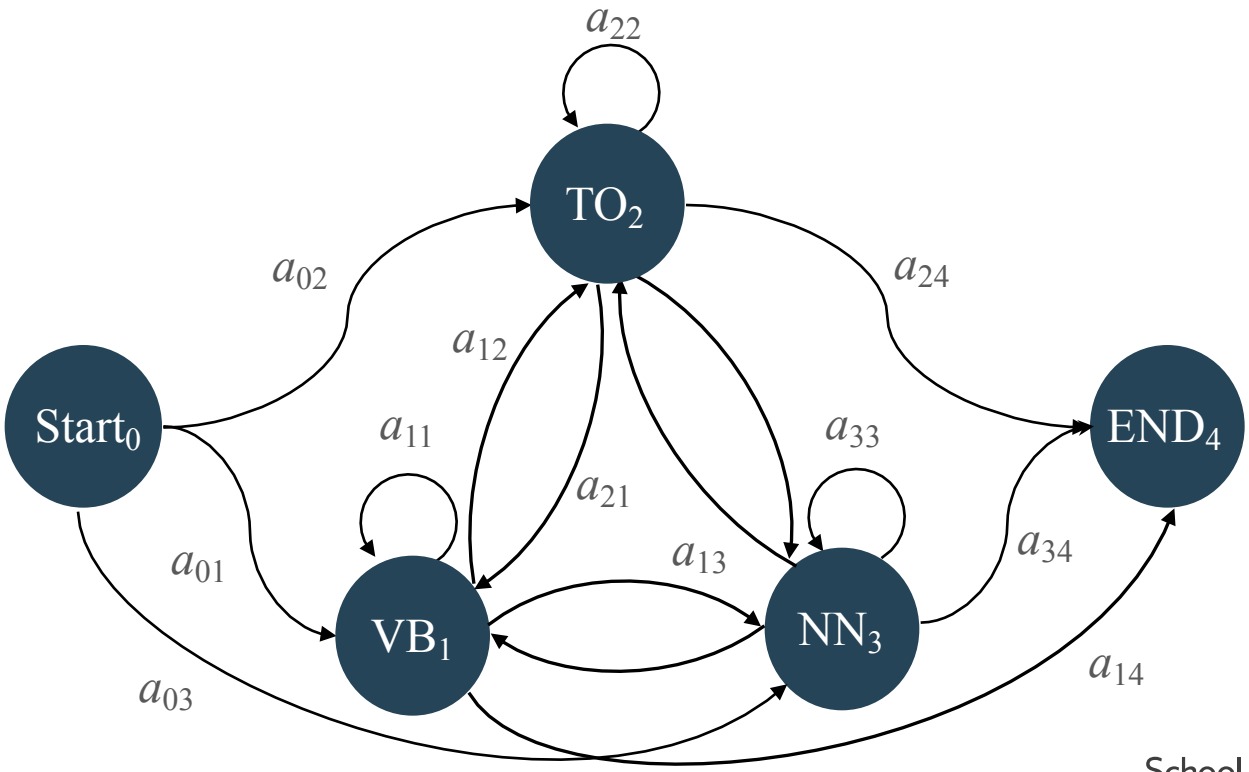
But in general, we need an optimization algorithm to most efficiently pick the best tag sequence without computing all possible combinations of tag sequence probabilities.

What we've described with these two kinds of probabilities is a Hidden Markov Model (HMM).

- The Markov Model is the sequence of words, and the hidden states are the POS tags for each word.

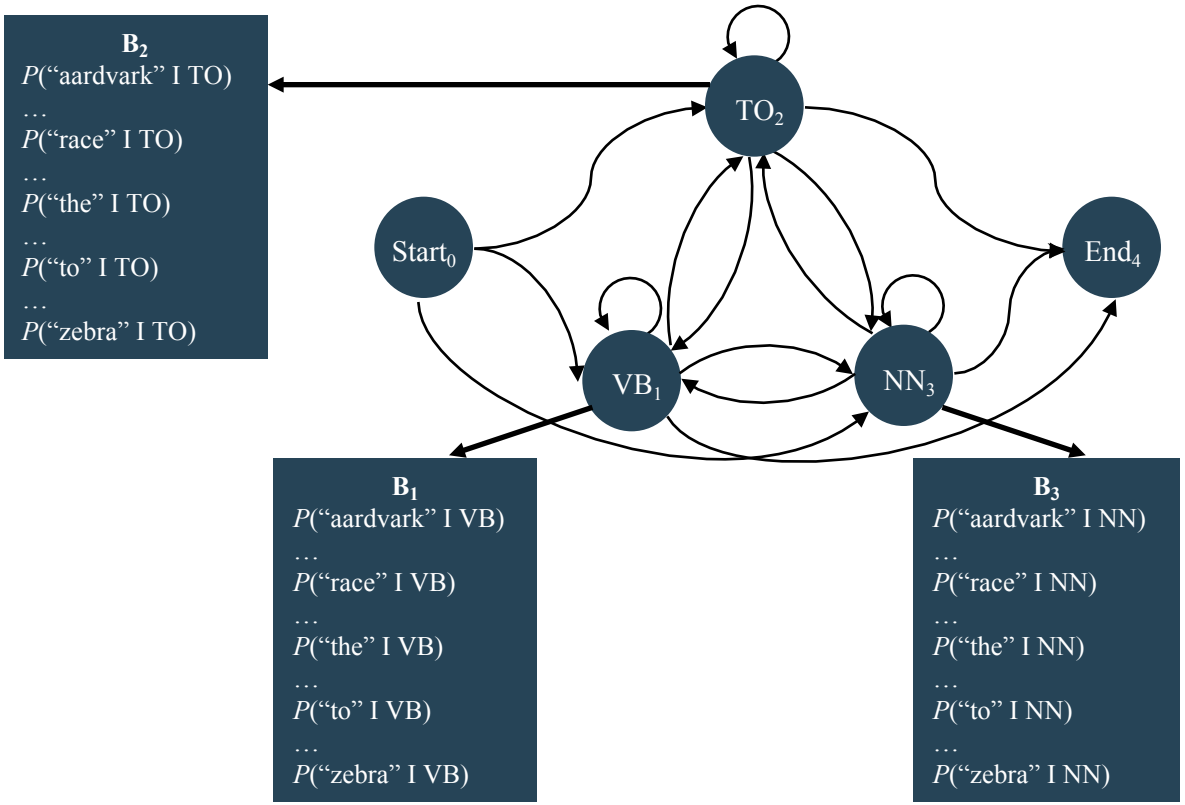
Tag Transition Probabilities for HMM

The HMM hidden states, the POS tags, can be represented in a graph where the edges are the transition probabilities between POS tags.



Word Likelihoods for HMM

For each POS tag, give words with probabilities.



The A Matrix for the POS HMM

Example of tag transition probabilities represented in a matrix, usually called the A matrix in an HMM:

- The probability that VB follows <s> is .019,…

	VB	TO	NN	PPSS
<s>	.019	.0043	.041	.067
VB	.0038	.035	.047	.0070
TO	.83	0	.00047	0
NN	.0040	.016	.087	.0045
PPSS	.23	.00079	.0012	.00014

Figure 4.15. Tag transition probabilities (the *a* array, $P(t_i|t_{i-1})$ computed from the 87-tag Brown Corpus without smoothing. The rows are labeled with the conditioning events; thus $P(PPSS|VB)$ is .0070. The symbol <s> is the start-of-sentence symbol.

From Jim Martin

The B Matrix for the POS HMM

Word likelihood probabilities are represented in a matrix, where for each tag, we show the probability that the tag on that word,

	I	want	to	race
VB	0	.0093	0	.00012
TO	0	0	.99	0
NN	0	.000054	0	.00057
PPSS	.37	0	0	0

Figure 4.16. Observation likelihood (the *b* array) computed from the 87-tag Brown Corpus without smoothing.

From Jim Martin

Using HMMs for POS Tagging

From the tagged corpus, create a tagger by computing the two matrices of probabilities, A and B.

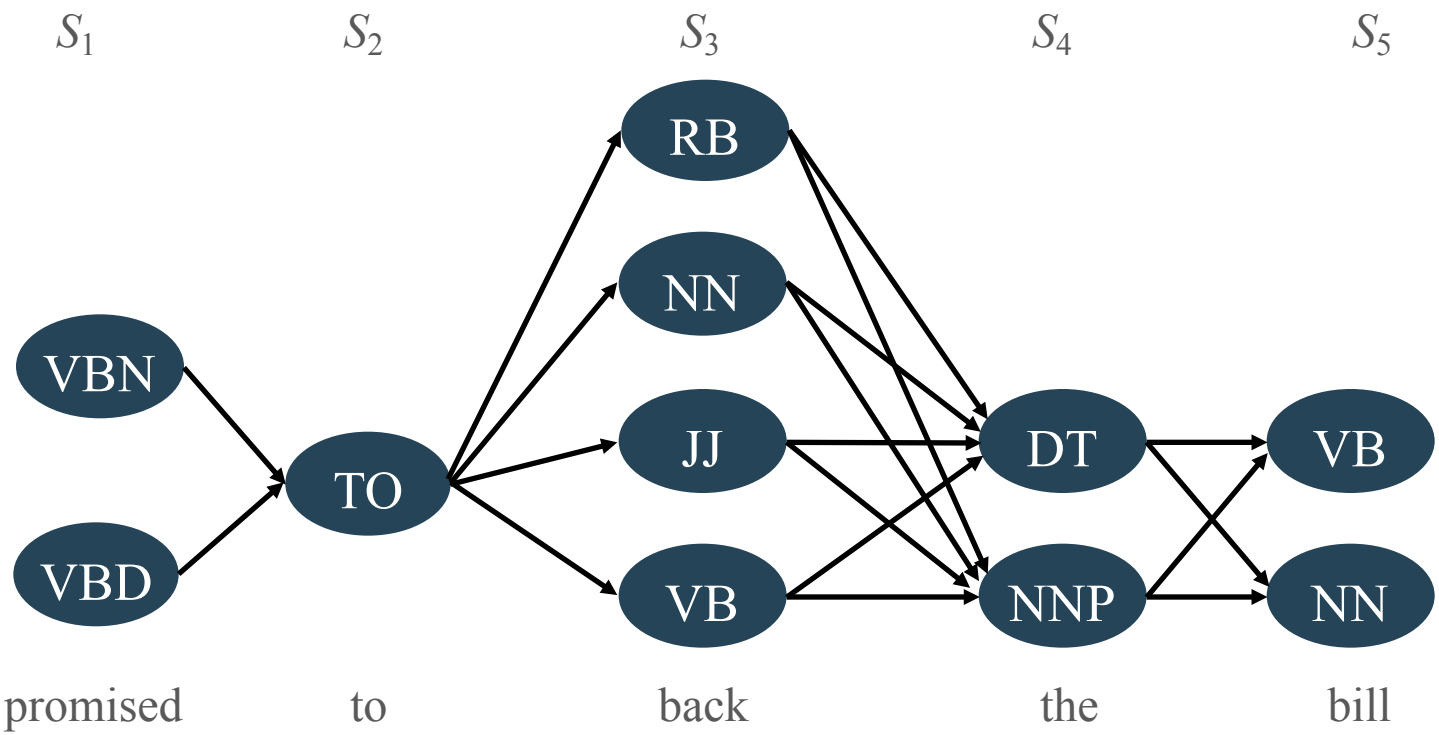
- It is straightforward for bigram HMM, done by counting.
- For higher-order HMMs, efficiently compute matrix by the forward–backward algorithm.

To apply the HMM tagger to unseen text, we must find the best sequence of transitions.

- Given a sequence of words, find the sequence of states (POS tags) with the highest probabilities along the path.
- This task is sometimes called “decoding.”
- Use the Viterbi algorithm.

Viterbi Intuition: We Are Looking for the Best "Path"

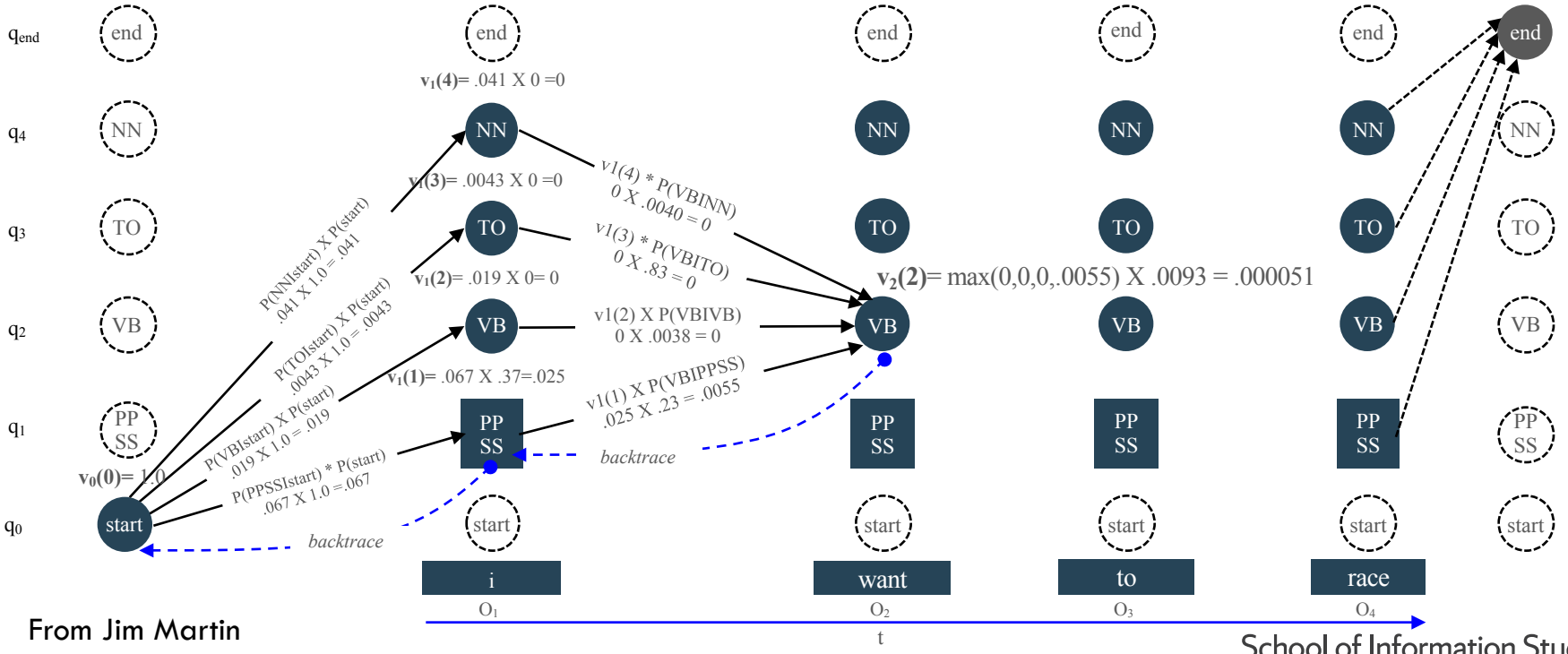
Each word has states representing the possible POS tags.



Viterbi Example

Each pair of tags labeled with an edge giving transition probability

Each tag in a state labeled with a Viterbi value giving maximum over states in previous word of: (its Viterbi value * transition probability * word likelihood), representing "best path to this node"



From Jim Martin

Viterbi Algorithm Sketch

This algorithm fills in the elements of the array Viterbi in the previous slide (columns are words, rows are states (POS tags))

function Viterbi

for each state *s,* compute the initial column

Viterbi[*s,* 1] = A[0, *s*] * B[*s,* word1]

for each word *w* from 2 to *N* (length of sequence)

for each state *s,* compute the column for *w*

Viterbi[*s, w*] =

maximum over *s'* (Viterbi[*s'*, *w* – 1] * A[*s'*, *s*] * B[*s, w*])

<save back pointer to trace final path>

return the trace of back pointers

where A is the matrix of state transitions and B is the matrix of state/word likelihoods

POS Tagging: Classifier

School of Information Studies
Syracuse University

Recall HMM

So, an HMM POS tagger computes the tag transition probabilities (the A matrix) and word likelihood probabilities for each tag (the B matrix) from a (training) corpus.

Then, for each sentence that we want to tag, it uses the Viterbi algorithm to find the path of the best sequence of tags to fit that sentence.

- Described in the optional lecture

This is an example of a sequential classifier. Let's look at how this is related to a more traditional classifier, which we might call a feature-based classifier.

- Classification task:

  Given a word in a sentence, what is its POS tag?

Comparison of HMM and Feature-Based Classifiers

Recall that **HMM (and *N*-gram) taggers** are sequential classifiers that use the previous sequence of tags as information.

$word_{n-1}$	…	$word_{-2}$	$word_{-1}$	word
tag_{n-1}	…	tag_{-2}	tag_{-1}	XX

- In the order from left to right, use information from previous tags (tag prior probabilities) and word (word likelihood probabilities) to predict the next tag in the sequence.

Instead, a **feature-based classifier** is looking just at the word and properties/features of the surrounding words.

$word_{-2}$	$word_{-1}$	word	$word_{+1}$	$word_{+2}$
		XX		

- Assign a tag XX to the word.

How Can We Improve Our Tagger?

What are the main sources of information for our HMM POS tagger?

- Knowledge of tags of neighboring words
- Knowledge of word tag probabilities
  - *man* is rarely used as a verb

Unknown words (words not occurring in the training corpus) can be a problem because we don't have this information.

And we are not including information about the features of the words themselves.

Feature-Based Classifiers

A feature-based classifier is an algorithm that will take a word and assign a POS tag based on features of the word in its context in the sentence.

Many algorithms are used for these traditional classifiers. Just to name a few:

- Naïve Bayes, Maximum entropy (MaxEnt), Support vector machines (SVM)
- We'll be covering a lot more about classifiers later in the course

Features of Words

Can do surprisingly well just looking at a word by itself:

- Word — the: the → DT (determiner)
- Lowercased word — Importantly: importantly → RB (adverb)
- Prefixes — unfathomable: un- → JJ (adjective)
- Suffixes — Importantly: -ly → RB
tangential: -al → JJ
- Capitalization — Meridian: CAP → NNP (proper noun)
- Word shapes — 35-year: d-x → JJ

These properties can include information about the previous or the next word(s)

- The word *be* appears to the left — pretty → JJ

But not information about tags of the previous or next words, unlike HMM

Development Process for Features

The tagged data should be separated into a training set and a test set.

- The classifier is trained on the training set, which produces a "tagger"
- And evaluated on the test set, by applying the tagger to every word and comparing the predicted tag with the answer in the test set
  - May also hold out some data for development
  - Evaluation numbers are not prejudiced by the training set

Development Process for Features

If our feature-based tagger has errors, then we improve the features.

- Suppose we incorrectly tag *as* as IN in the phrase *as soon as*, when it should be RB:

  PRP VBD IN RB IN PRP VBD.
  They left as soon as he arrived.

- We could fix this with a feature that includes the next word.

POS Tagging: Evaluation and Demos

School of Information Studies
Syracuse University

Evaluation: Is Our POS Tagger Any Good?

Answer: We use a manually tagged corpus, which we will call the "gold standard."

- We run our POS tagger on the gold standard and compare its predicted tags with the gold tags.
- We compute the accuracy (and other evaluation measures).

Important: 100% is impossible, even for human annotators.

- We estimate that humans can do POS tagging at about 98% accuracy (by comparing humans with each other).
- Some tagging decisions are very subtle and hard to do.
  - Mrs/NNP Shaefer/NNP never/RB got/VBD around/RP to/TO joining/VBG
  - All/DT we/PRP gotta/VBN do/VB is/VBZ go/VB around/IN the/DT corner/NN
  - Chateau/NNP Petrus/NNP costs/VBZ around/RB 250/CD
- The "gold standard" will have human mistakes; humans are subject to fatigue etc.

Overview of POS Tagger Accuracy

Stanford NLP group performed experiments with different tagging techniques and looked at the improvements.

Techniques	accuracies of all words/unknown words
▪ Most frequent tag:	~90% / ~50%
▪ Trigram HMM:	~95%/~55%
▪ HMM with trigrams	
▪ MaxEnt $P(t\|w)$:	93.7%/82.6%
▪ Feature-based tagger	
▪ MEMM tagger:	96.9%/86.9%
▪ Combines feature based and HMM tagger	
▪ Bidirectional dependencies:	97.2%/90.0%
▪ Upper bound:	~98% (human agreement)

Most errors on unknown words

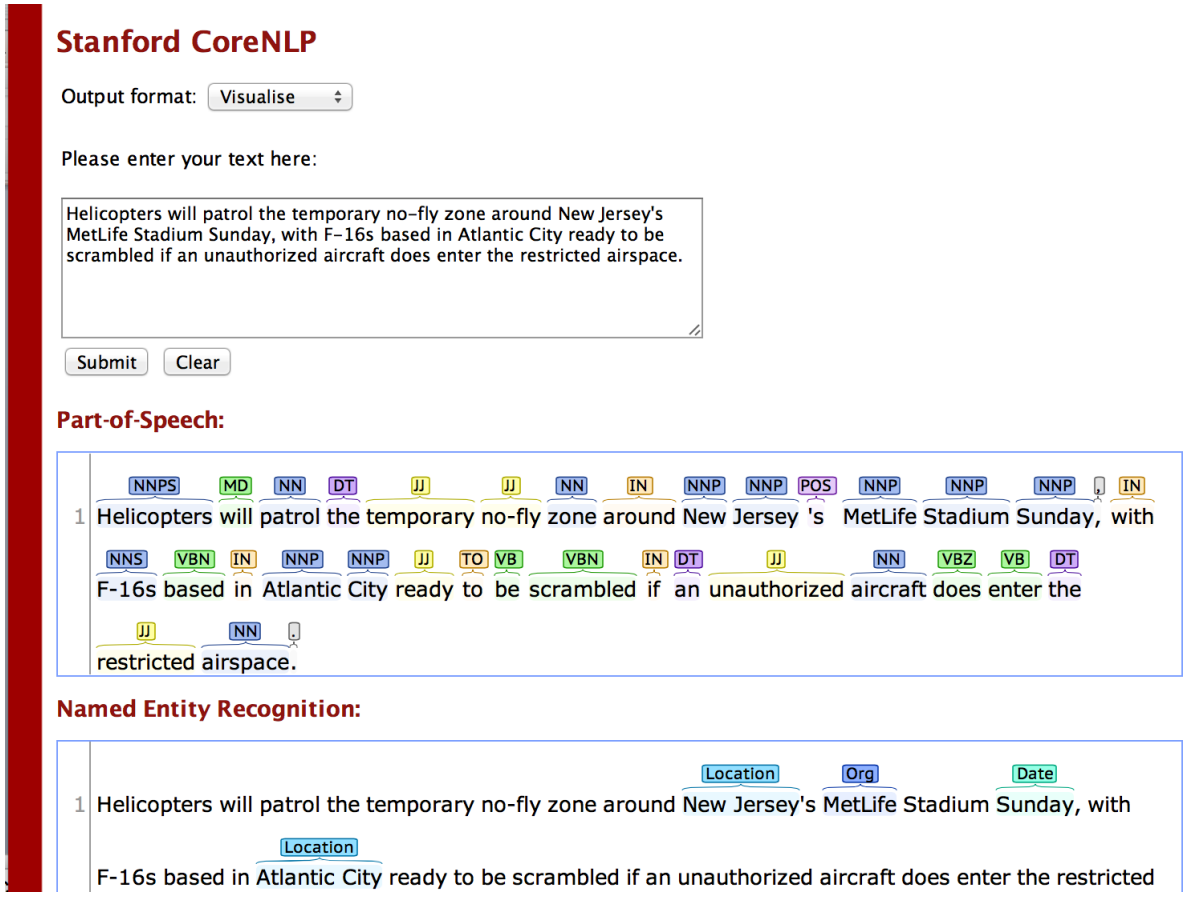
* Results from slides by Chris Manning

POS Taggers With Online Demos

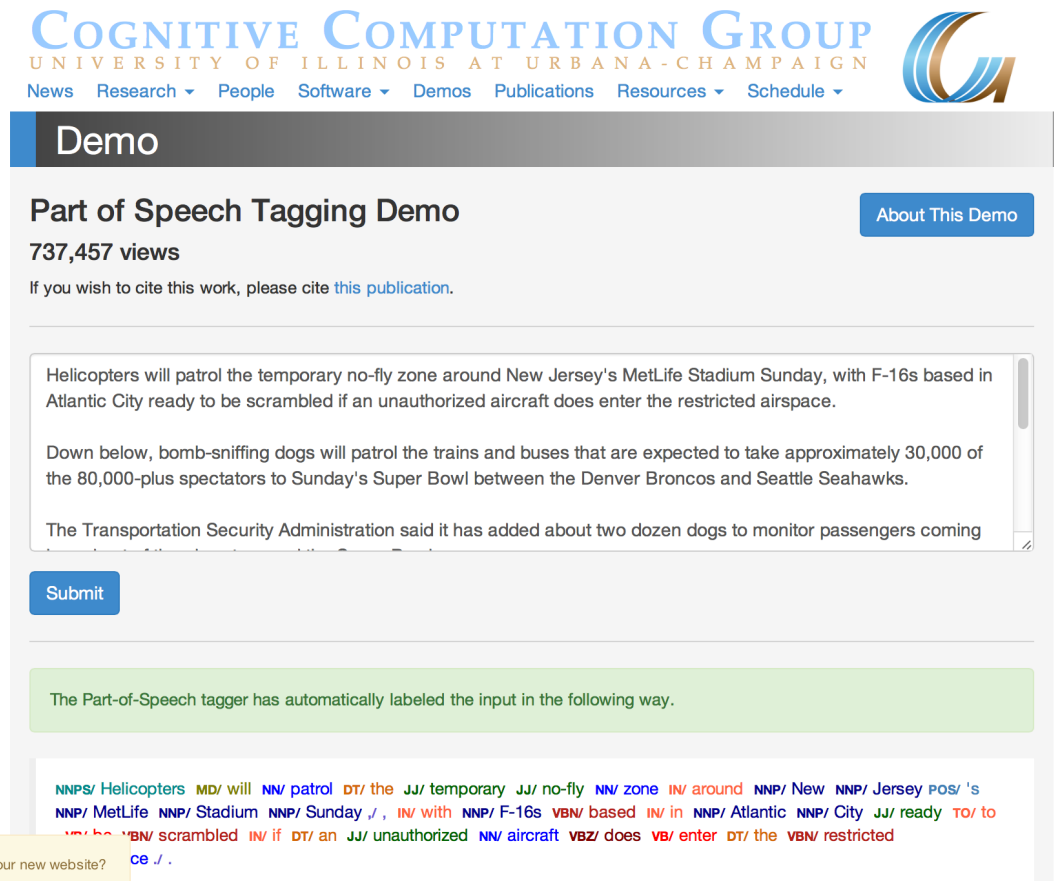
There are not too many online taggers available for demos, but here are some possibilities:

- The Stanford online parser demo includes POS tags: http://nlp.stanford.edu:8080/parser/ http://nlp.stanford.edu:8080/corenlp/
- Illinois (UIUC) tagger demo from the Cognitive Computation Group
- http://cogcomp.cs.illinois.edu/demo/pos/?id=4 (colors!)

Stanford NLP Demo



UIUC Demo



School of Information Studies
Syracuse University

Conclusions

Part-of-speech tagging is a doable task with high-performance results.

- In addition to the standard text POS taggers discussed here, there are now POS tag systems and taggers developed for social media text.

It contributes to many practical, real-world NLP applications and is now used as a pre-processing module in most systems.

Computational techniques learned at this level can be applied to NLP tasks at higher levels of language processing.

Classification Introduction

School of Information Studies
Syracuse University

Classification: Definition

Given a collection of examples (training set):

- Each example is represented by a set of features, sometimes called attributes
- Each example is to be given a label or class

Find a model for the label as a function of the values of features.

Goal: Use the model to assign labels to previously unseen examples as accurately as possible.

A test set is used to determine the accuracy of the model.

- Usually, the given data set is divided into training and test sets, with training set used to build the model and test set used to validate it.

Supervised vs. Unsupervised Learning

Supervised learning (classification and other tasks)

- Supervision: the training data (observations, measurements, etc.) are accompanied by labels indicating the class of the observations
- New data are classified based on the training set

Unsupervised learning (includes clustering)

- The class labels of training data are unknown
- Given a set of measurements, observations, etc. with the aim of establishing the existence of classes or clusters in the data

NLP Tasks

Many NLP tasks can be accomplished either through:

- Rule-based or symbolic techniques, such as using RegExp
- Supervised techniques, where the task is defined automatically from a training set

In both cases, the evaluation of the task will most likely use a training set to define the technique and a test set for evaluation.

- POS tagging uses Hidden Markov Models
- Parsing uses statistical lexicalized parsers
- Sentiment analysis uses classification

The evaluation of these tasks often uses ideas from the evaluation of classification.

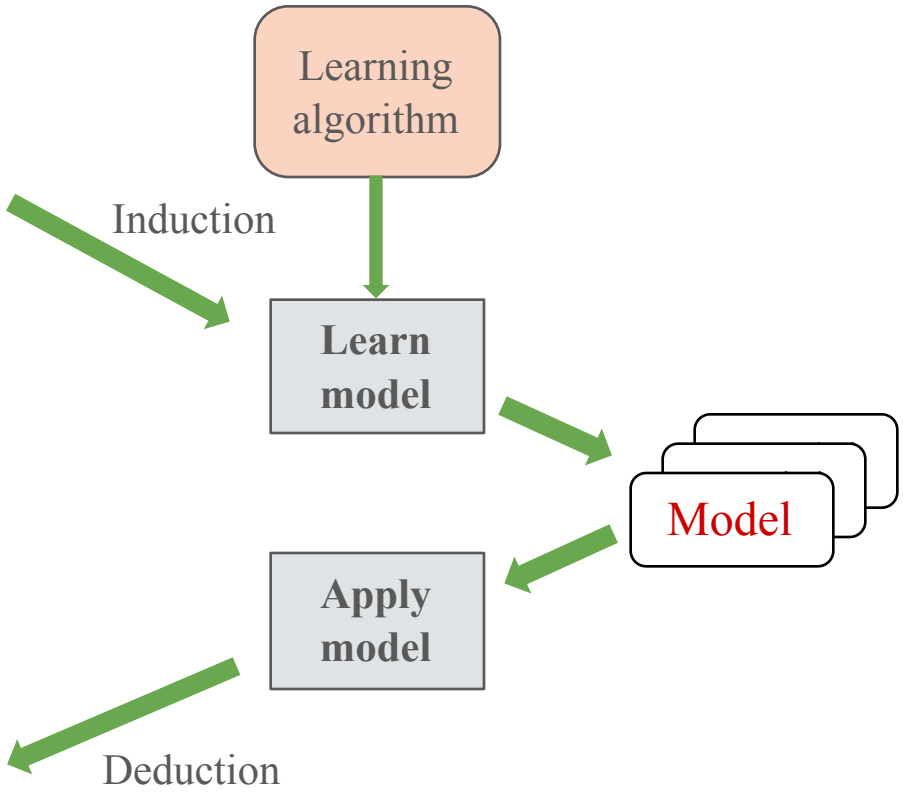
Illustrating Classification Tasks

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	**No**
2	No	Medium	100K	**No**
3	No	Small	70K	**No**
4	Yes	Medium	120K	**No**
5	No	Large	95K	**Yes**
6	No	Medium	60K	**No**
7	Yes	Large	220K	**No**
8	No	Small	85K	**Yes**
9	No	Medium	75K	**No**
10	No	Small	90K	**Yes**

Training Set

Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	**?**
12	Yes	Medium	80K	**?**
13	Yes	Large	110K	**?**
14	No	Small	95K	**?**
15	No	Large	67K	**?**

Test Set



Classification Techniques

There are a number of different classification algorithms to build a model for classification.

- Decision-tree methods, Rule-based methods, Memory-based reasoning, instance-based learning, Neural networks, Genetic algorithms, Naïve Bayes and Bayesian belief networks, Support vector machines

In this introduction, we illustrate classification tasks using decision-tree methods.

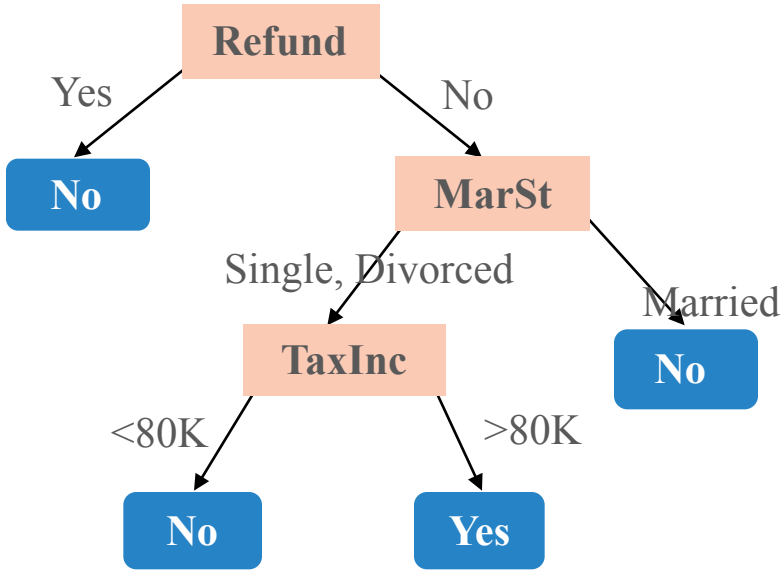
Features can have numeric values (continuous) or a finite set of values (categorical/nominal), including Boolean true/false.

Example of a Decision Tree

Boolean Categorical Continuous Class

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	**No**
2	No	Married	100K	**No**
3	No	Single	70K	**No**
4	Yes	Married	120K	**No**
5	No	Divorced	95K	**Yes**
6	No	Married	60K	**No**
7	Yes	Divorced	220K	**No**
8	No	Single	85K	**Yes**
9	No	Married	75K	**No**
10	No	Single	90K	**Yes**

Training Data

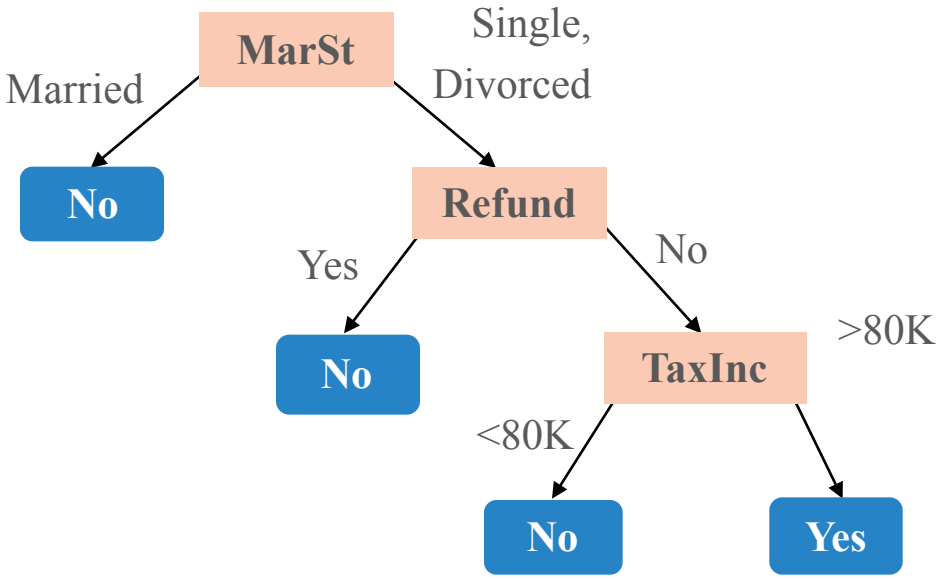


Model: Decision Tree

Example task: Given the marital status (MarSt), refund status, and taxable income (TaxInc) of a person, label them as to whether they will cheat on their income tax.

Another Example of Decision Tree

	Boolean	Categorical	Continuous	Class
Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes



There could be more than one tree that fits the same data!

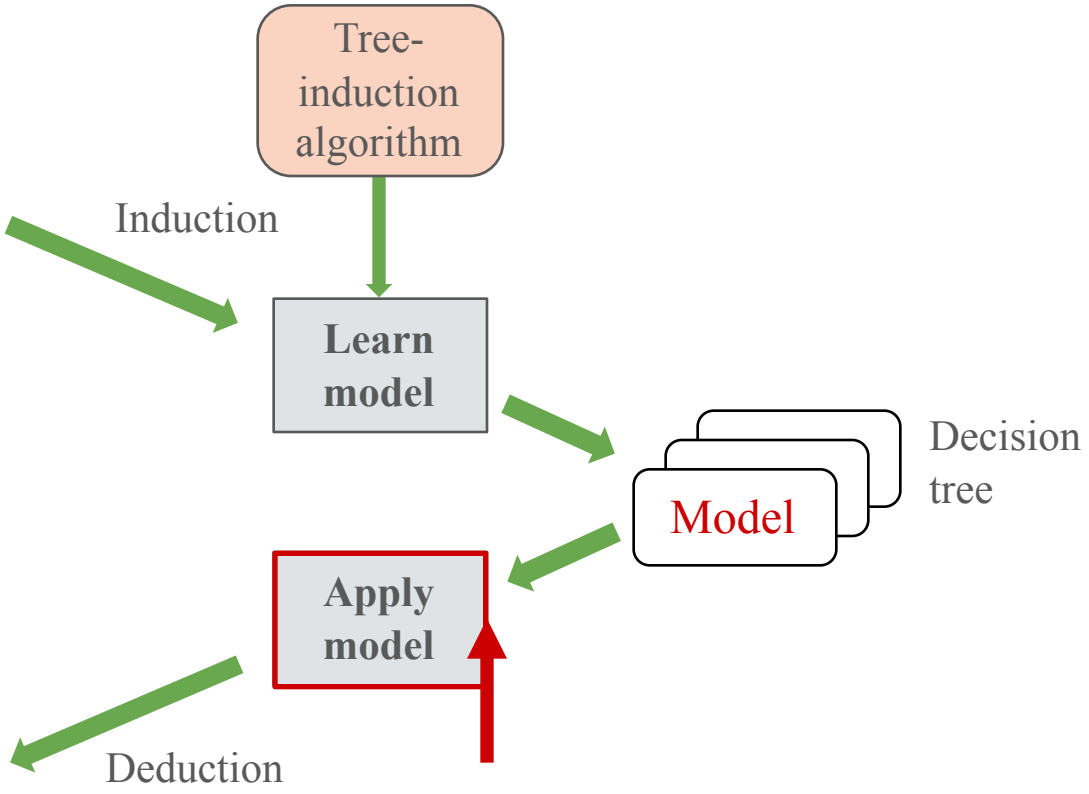
Decision-Tree Classification Task

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	**No**
2	No	Medium	100K	**No**
3	No	Small	70K	**No**
4	Yes	Medium	120K	**No**
5	No	Large	95K	**Yes**
6	No	Medium	60K	**No**
7	Yes	Large	220K	**No**
8	No	Small	85K	**Yes**
9	No	Medium	75K	**No**
10	No	Small	90K	**Yes**

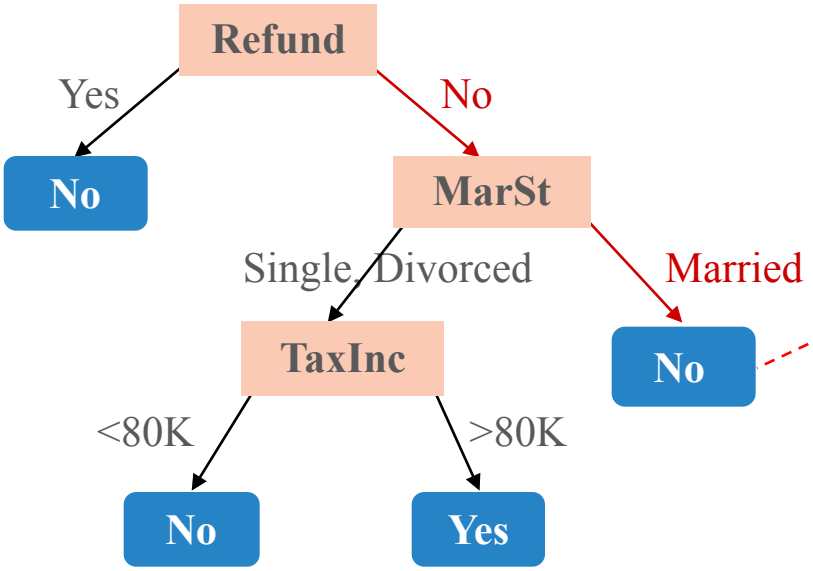
Training Set

Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	**?**
12	Yes	Medium	80K	**?**
13	Yes	Large	110K	**?**
14	No	Small	95K	**?**
15	No	Large	67K	**?**

Test Set



Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Assign cheat to "No"

Classification Evaluation

School of Information Studies
Syracuse University

Confusion Matrix

Focus on the predictive capability of a model on test set

- Compare the known actual class with the predicted class

Confusion matrix for a binary classifier (two labels) on test set:

		Predicted Class	
		Class = Yes	Class = No
Actual Class	Class = Yes	a	b
	Class = No	c	d

a: TP (true positive)

b: FN (false negative)

c: FP (false positive)

d: TN (true negative)

Classifier Evaluation Measures

Confusion matrix:

	Yes - C_1	No - C_2
Yes - C_1	a: True positive	b: False negative
No - C_2	c: False positive	d: True negative

$$\text{Accuracy} = \frac{a+d}{a+b+c+d} = \frac{TP+TN}{TP+TN+FP+FN}$$

Precision = TP/(TP + FP),

percent correct out of all predicted Yes

Recall = TP/(TP + FN),

percent correct out of all actual Yes

F-measure = 2 * (Recall * Precision)/(Recall + Precision)

Accuracy Example

Suppose that we are predicting whether or not someone will buy a computer.

- Evaluating accuracy with a test set of 10,000 people

Classes	buy_computer = yes	buy_computer = no	Total
buy_computer = yes	6,954	46	7000
buy_computer = no	412	2,588	3000
total	7,366	2,634	10000

$$\text{Accuracy} = \frac{a+d}{a+b+c+d} = \frac{TP+TN}{TP+TN+FP+FN}$$

Accuracy = 6,954 + 2,588/10,000 = 95.42%

Multi-Class Classification

Many classification algorithms solve binary classification tasks, while many tasks are naturally multi-class (i.e., there are more than two labels).

Multi-class problems are solved by training a number of binary classifiers and combining them to get a multi-class result.

The confusion matrix is extended to the multi-class case.

The accuracy definition is naturally extended to the multi-class case.

Precision and recall are defined for the binary classifiers trained for each label.

Issues With Imbalanced Classes

Consider a two-class problem with labels Yes and No.

- Number of No examples = 990
- Number of Yes examples = 10

If the model predicts everything to be No, accuracy is 990/1,000 = 99%.

- Accuracy is misleading because the model does not detect any Yes examples.
- Precision and recall will be better measures if you are training a classifier to find rare examples.

Classifier Evaluation Methods

Holdout method

- Given data are randomly partitioned into two independent sets:
  - Training set (e.g., 2/3) for model construction
  - Test set (e.g., 1/3) for accuracy estimation

Cross-validation (*k* – fold, where *k* = 10 is most popular)

- Randomly partition the data into *k* mutually exclusive subsets, each approximately equal size
- At *i –th* iteration, use Di as test set and others as training set