CS140: Feature Selection

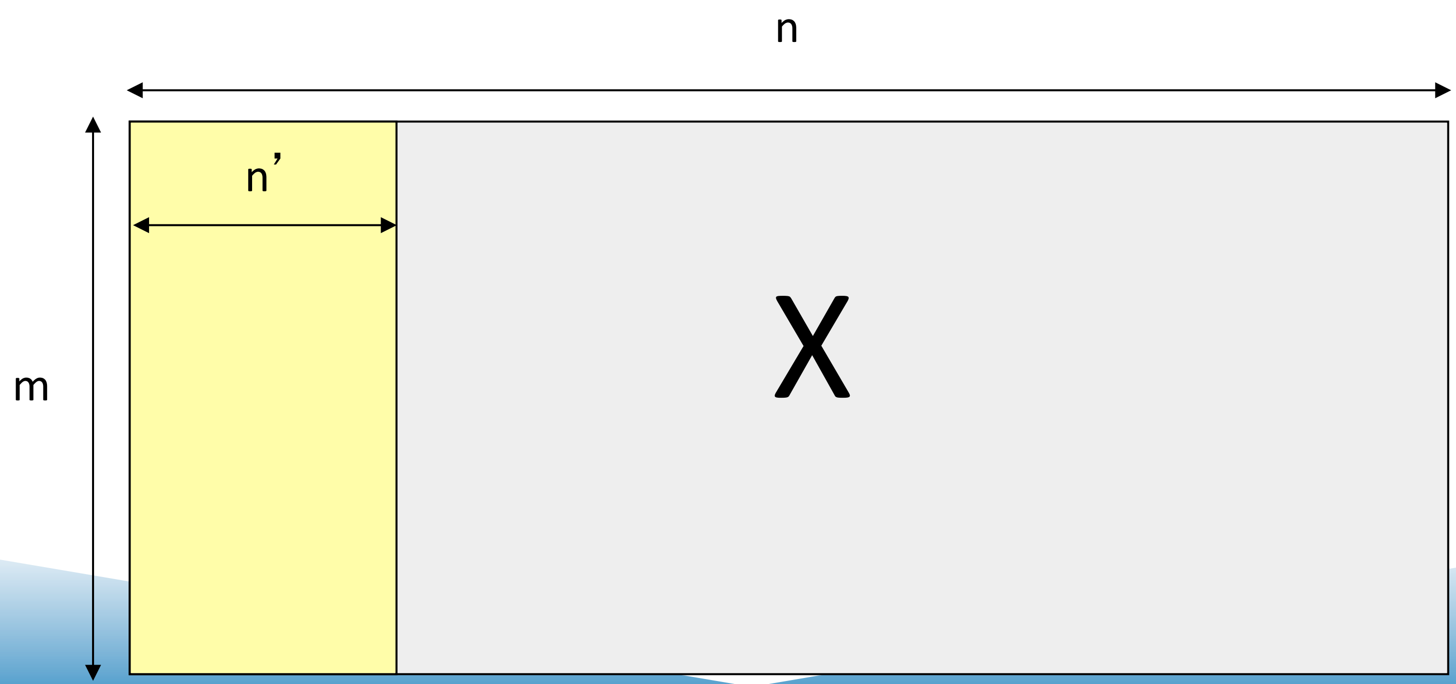
March 22, 2016

James Pustejovsky
Brandeis University

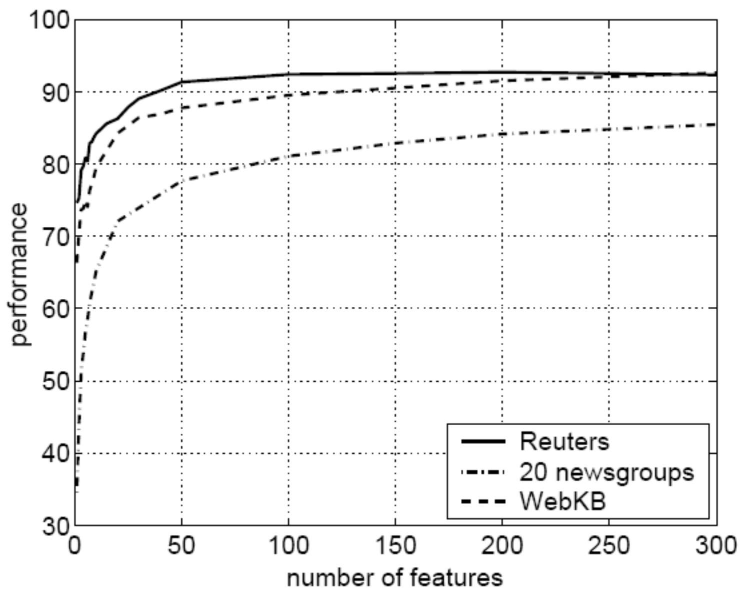
Slides from Fei Xia.

Feature Selection

- **Thousands to millions of low level features**: select the most relevant one to build **better, faster, and easier to understand** learning machines.



Text Filtering



Reuters: 21578 news wire, 114 semantic categories.

20 newsgroups: 19997 articles, 20 categories.

WebKB: 8282 web pages, 7 categories.

Bag-of-words: >100000 features.

Top 3 words of some categories:

- **Alt.atheism**: atheism, atheists, morality
- **Comp.graphics**: image, jpeg, graphics
- **Sci.space**: space, nasa, orbit
- **Soc.religion.christian**: god, church, sin
- **Talk.politics.mideast**: israel, armenian, turkish
- **Talk.religion.misc**: jesus, god, jehovah

Bekkerman et al, JMLR, 2003

The Cycle of Computational Linguistics

- We can study anything about language ...

1. Formalize some insights
2. Study the formalism mathematically
3. Develop & implement algorithms

   Select the features!

4. Test on real data

Feature types

- Target
  - What you are trying to learn
  - Consider complexity
    - 43 parts of speech or 118?
- “Features”
  - Selected knowledge that is used to train the model
  - Must be something I can measure/count!
  - Some are more obvious than others

Which features to use?

Most crucial decision you’ll make!

1. Topic
   - Words, phrases, ?
2. Author
   - Stylistic features
3. Sentiment
   - Adjectives, ?
4. Spam
   - Specialized vocabulary

How to choose features

- Consider cost
  - Words vs. POS vs parse tree
- Observable/countable
- Differentiating
  - Remove "non-informative" terms from documents
- Questions to consider
  - Stemmed or surface form?
  - Single words or phrases?
  - Words or word classes?

Word Sense Disambiguation

- Supervised machine learning approach:
  - A training corpus of words tagged in context with their sense
  - Corpus is used to train a classifier that can tag words in new text
- Summary of what we need:
  - the **tag set** ("sense inventory")
  - the **training corpus**
  - A set of **features** extracted from the training corpus
  - A **classifier**

Feature vectors

- A simple representation for each observation (each instance of a target word)
  - Vectors of sets of feature/value pairs
    - I.e. files of comma-separated values
  - These vectors should represent the window of words around the target

Collocational

- Position-specific information about the words in the window
- guitar and bass player stand
  - [guitar, NN, and, CC, player, NN, stand, VB]
  - $\text{Word}_{n-2,}$ $\text{POS}_{n-2,}$ $\text{word}_{n-1,}$ $\text{POS}_{n-1,}$ Word_{n+1} POS_{n+1}…
  - In other words, a vector consisting of
  - [position n word, position n part-of-speech…]

Word Similarity: Context vector

- Consider a target word *w*
- Suppose we had one binary feature f_i for each of the *N* words in the lexicon v_i
- Which means "word v_i occurs in the neighborhood of *w*"
- w=(f1,f2,f3,…,fN)
- If w=tezguino, v1 = bottle, v2 = drunk, v3 = matrix:
- w = (1,1,0,…)

Co-occurrence vectors based on dependencies

- For the word "cell": vector of NxR features
  - R is the number of dependency relations
- What do I need for this?

	subj-of, absorb	subj-of, adapt	subj-of, behave	...	pobj-of, inside	pobj-of, into	...	nmod-of, abnormality	nmod-of, anemia	nmod-of, architecture	...	obj-of, attack	obj-of, call	obj-of, come from	obj-of, decorate	...	nmod, bacteria	nmod, body	nmod, bone marrow
cell	1	1	1		16	30		3	8	1		6	11	3	2		3	2	2

Semantic Role Labeling

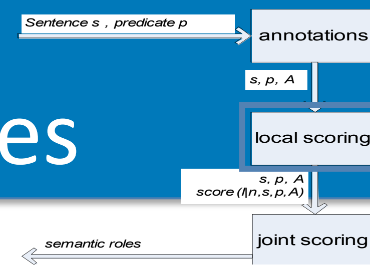
- What's the target? What am I trying to learn?
  - Traditional thematic roles
    - Agent, patient, theme, goal, instrument
  - FrameNet
    - Seller, buyer
  - "Agnostic" Propbank
    - A0, A1, A2
- What features are available that would help to model the distinctions?

Steps in SRL

From Xue & Palmer EMLNP 2004

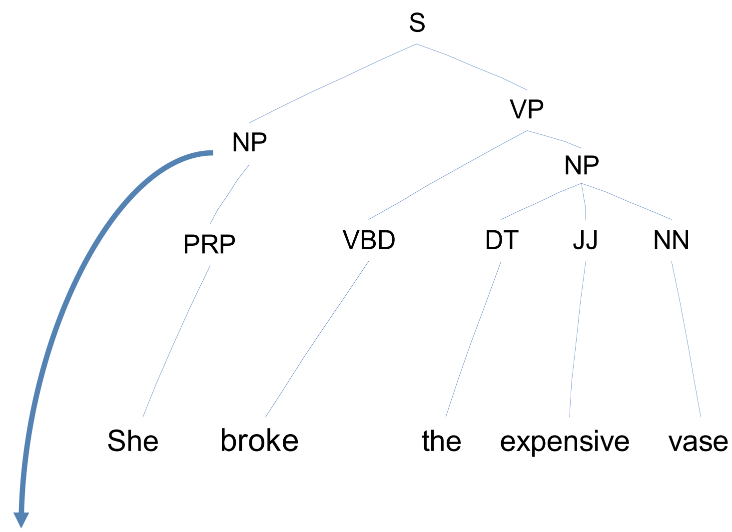
- Stage 1: Filter out constituents that are clearly not semantic arguments to the predicate in question (saves time)
- Stage 2: Classify the candidates derived from the first stage as either semantic arguments or non-arguments.
- Stage 3: Run a multi-category classifier to classify the constituents that are labeled as arguments into one of the classes plus NULL.

Gildea & Jurafsky (2002) Features



- Key early work
  - Future systems use these features as a baseline

- Constituent Independent
  - Target predicate (lemma)
  - Voice
  - Subcategorization
- Constituent Specific
  - Path
  - Position (*left, right*)
  - Phrase Type
  - Governing Category (*S* or *VP*)
  - Head Word

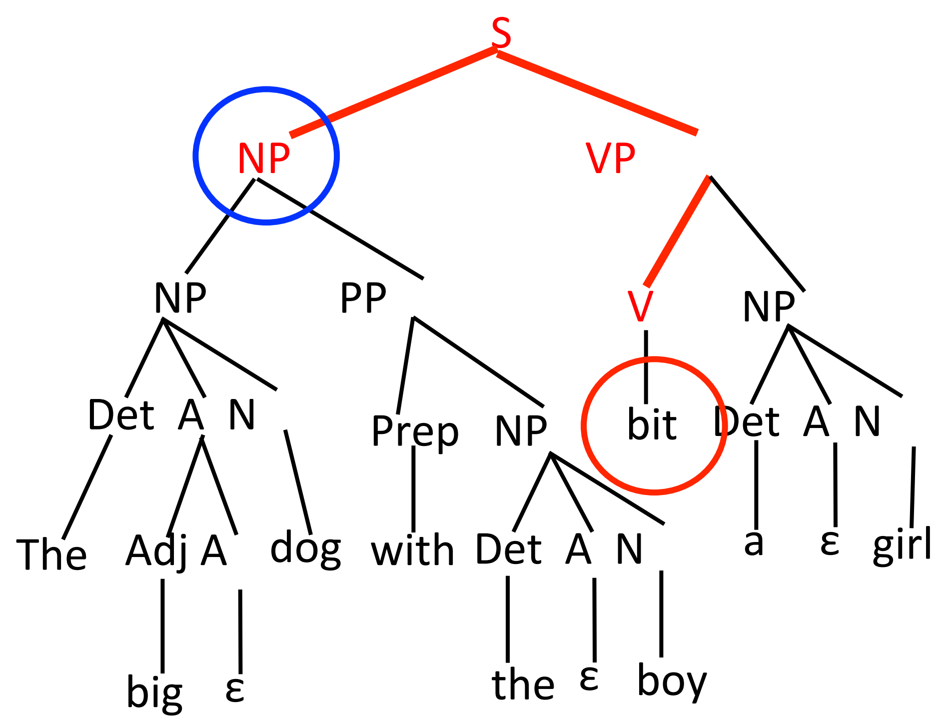


Target	*broke*
Voice	*active*
Subcategorization	*VP→VBD NP*
Path	*VBD↑VP↑S↓NP*
Position	*left*
Phrase Type	*NP*
Gov Cat	*S*
Head Word	*She*

Parse Tree Path Feature: Example 1

Path Feature Value:

V ↑ VP ↑ S ↓ NP

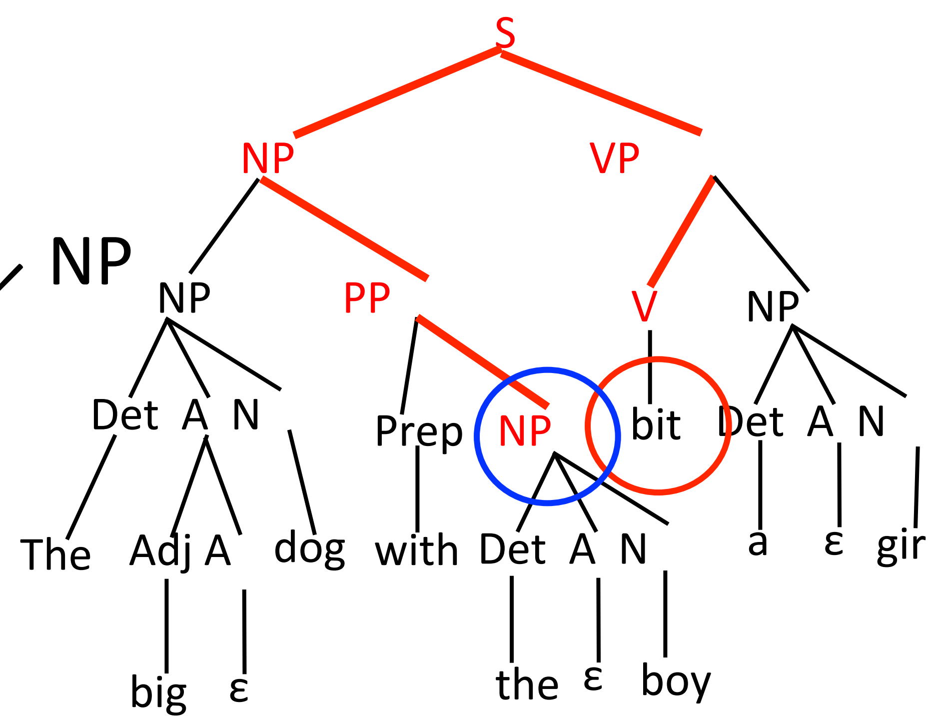


From Randy Mooney, UT Austin

Parse Tree Path Feature: Example 2

Path Feature Value:

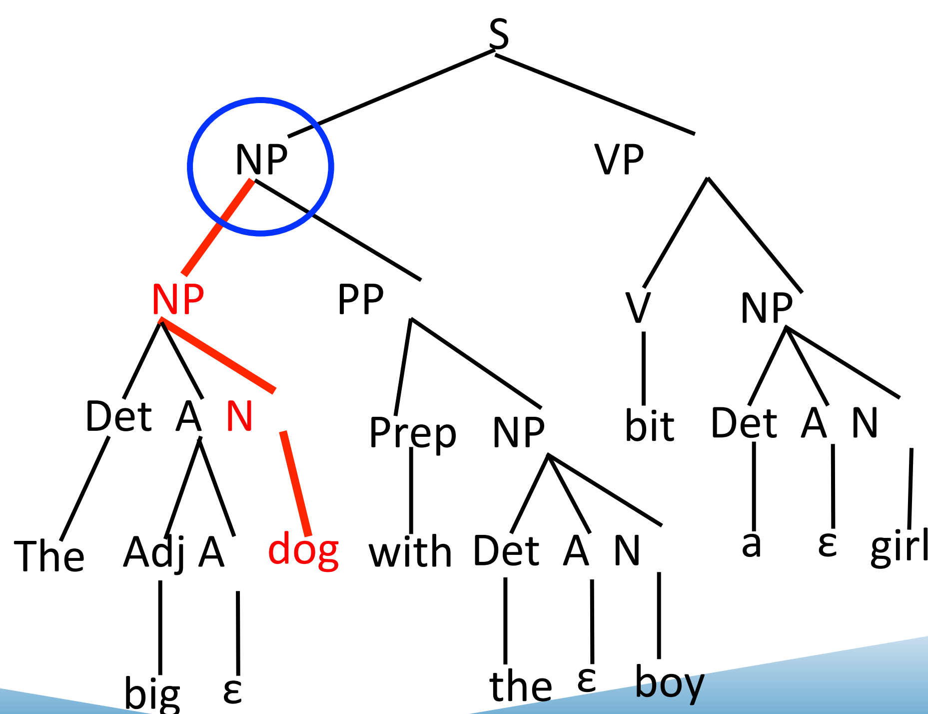
V ↑ VP ↑ S ↓ NP ↓ PP ↓ NP



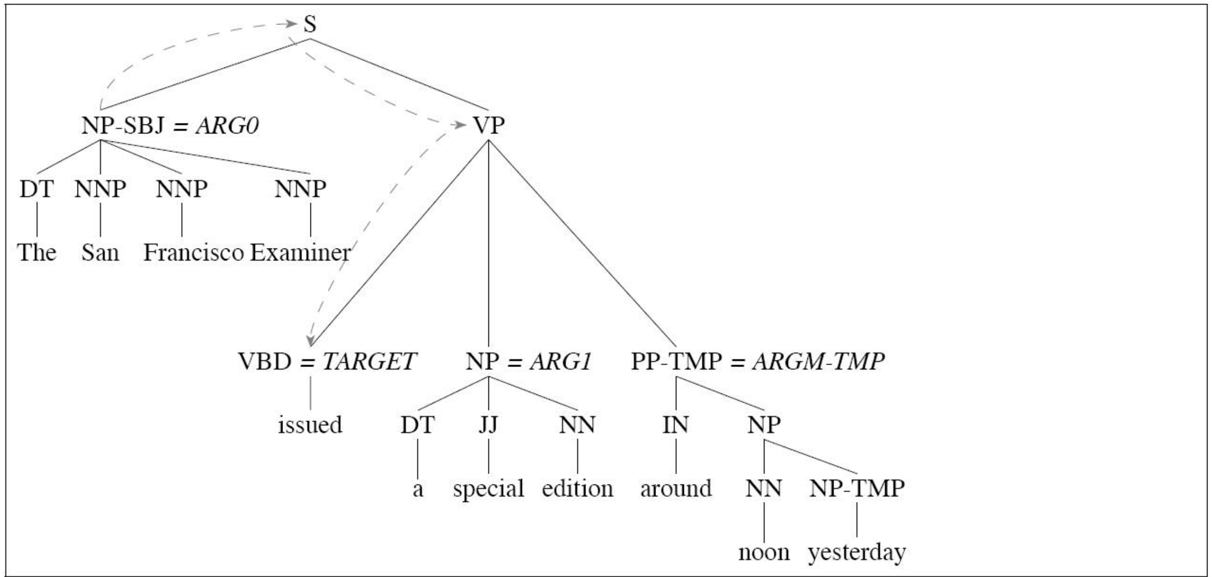
Head Word Feature Example

- There are standard syntactic rules for determining which word in a phrase is the **head**.

Head Word:
dog



Another example



Target	*issued*
Voice	*active*
Subcategorization	*VP→VBD NP PP*
Path	*VBD↑VP↑S↓NP*
Position	*left*
Phrase Type	*NP*
Gov Cat	*S*
Head Word	*Examiner*

Target	*issued*
Voice	*active*
Subcategorization	*VP→VBD NP PP*
Path	*VBD↑VP↓NP*
Position	*right*
Phrase Type	*NP*
Gov Cat	*VP*
Head Word	*edition*

Summary "Standard" features

- Predicate The predicate itself.
- Path The minimal path from the constituent being classified to the predicate.
- Phrase Type The syntactic category (NP, PP, etc.) of the constituent being classified.
- Position The relative position of the constituent being classified with regard to the predicate (before or after)
- Voice Whether the predicate is active or passive.
- Head Word The head word of the constituent being classified.
- Sub-categorization The phrase structure rule expanding the parent of the predicate.

Argument Identification

- A subset of features and their combination contribute most to argument identification
  - path,
  - head word, head word part-of-speech,
  - predicate - phrase type combination,
  - predicate- head word combination,
  - distance between constituent and predicate, with the predicate specified.

Argument identification

- Some features do not help discriminate argument identification
  - path: Can't distinguish between sisters
    - Direct object & indirect object not distinct
  - Subcategorization: Shared by all of the arguments
  - Voice: Same for all args, mabey combine with arg/ label
  - phrase type: Does help but would be stronger if p ared with the predicate
  - head word: Also should be paired with predicate

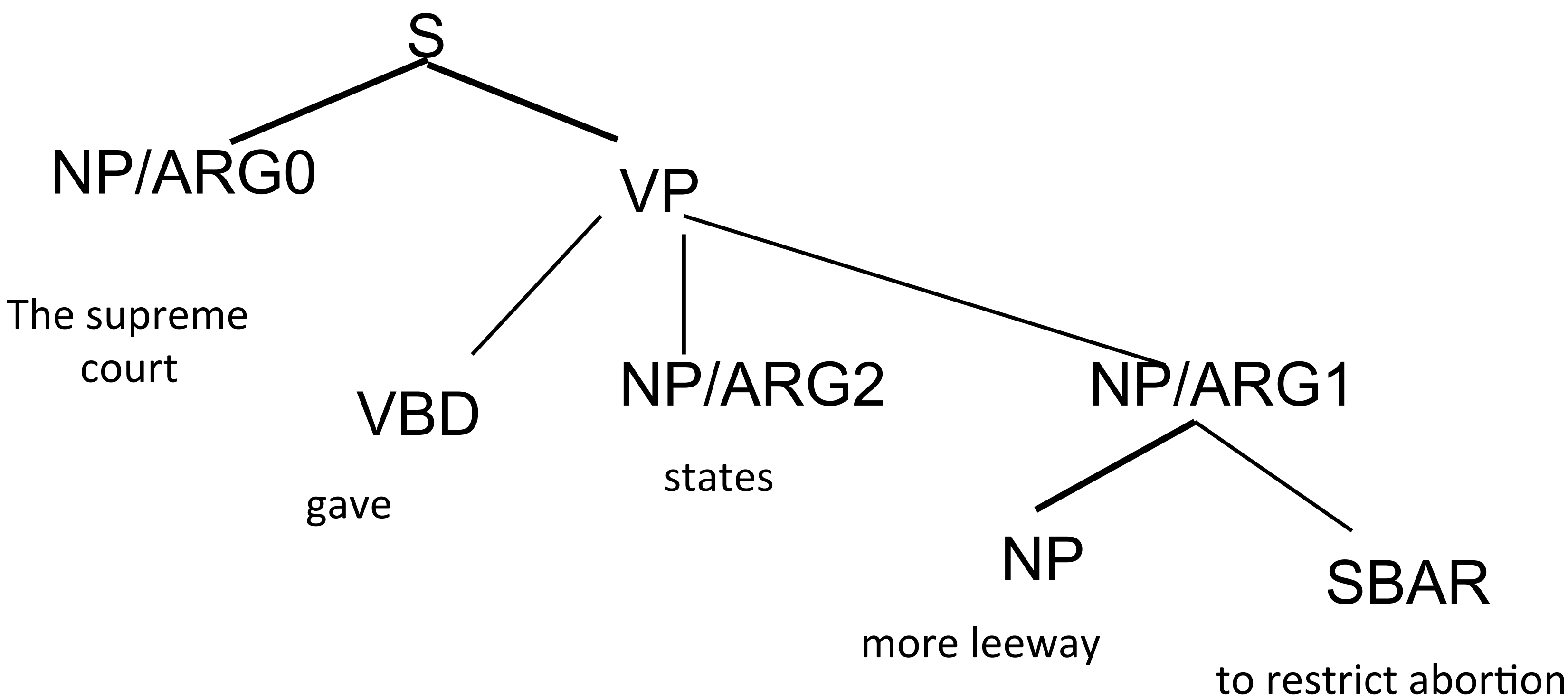
New features for Argument Identification

- Syntactic frame: varies with the constituent being classified to complement the path and subcat features
- Lexicalized constituent type: combination of the predicate lemma and the phrase type, rather than the phrase type itself, e.g. give np.
- Lexicalized head : predicate lemma and the head word combination as a feature, e.g. give states.
- Voice position combination: voice position combination as a feature, e.g. passive before.
- Head of PP: parent If the parent of the current constituent is a PP, then the head of this PP, the preposition is also used as a feature.

Performance per feature

Features	Accuracy	Gold(f)
Baseline	88.09	82.89
Syntactic frame	89.82	84.64
Pred-Head	88.69	83.77
Pred-POS	89.12	83.81
Voice position	88.44	82.57
PP parent	89.53	84.34
First word	88.60	83.01
Last word	88.64	83.51
Left sister	89.20	83.74
all	92.95	88.51

Syntactic Frames



Syntactic frame for “states”: np_give_NP_np

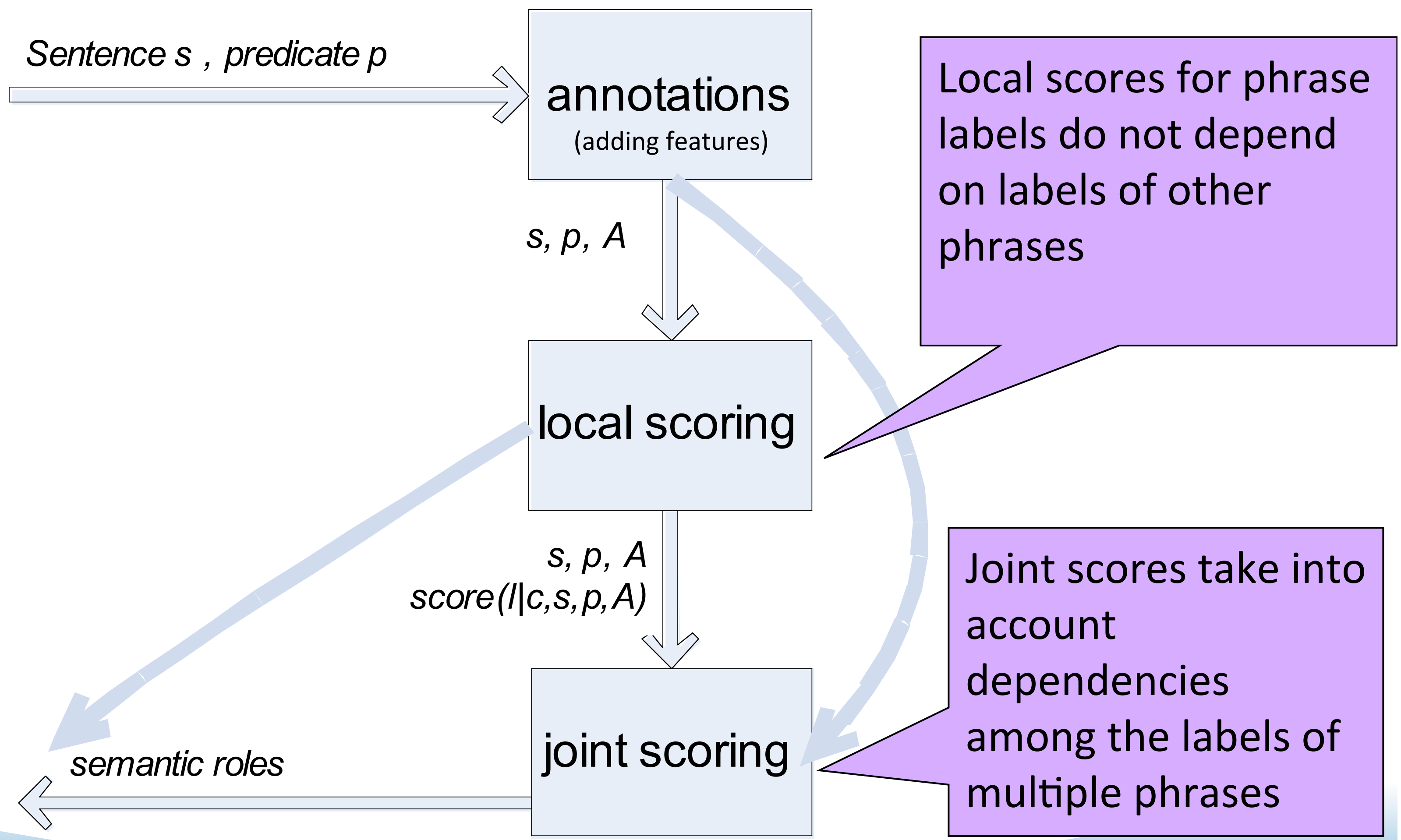
Syntactic from for “more leeway…”: np_give_np_NP

Pradhan et al. 2004 features

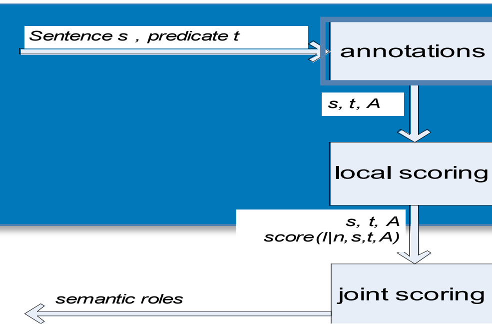
- Predicate cluster
- Noun head and POS of PP constituent
- Verb sense
- Partial path
- Named entities in constituent (7) [Surdeanu et al., 2003]
- Head word POS [Surdeanu et al., 2003]
- First and last word in constituent and their POS
- Parent and sibling features
- Constituent tree distance
- Ordinal constituent position
- Temporal cue words in constituent
- Previous 2 classifications

Slide from Sameer Pradhan

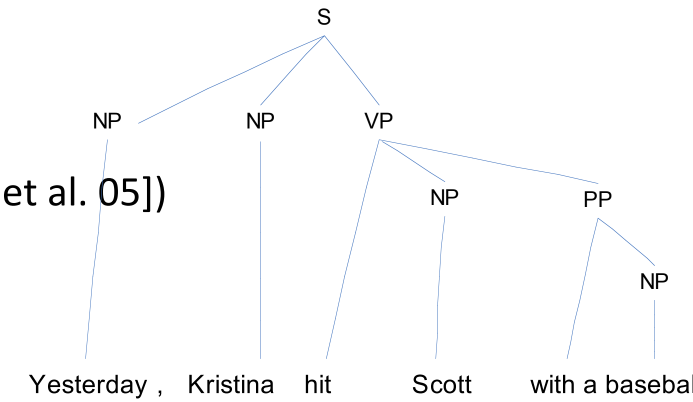
Basic Architecture of a Generic SRL System



Annotations Used



- Syntactic Parsers
  - Collins', Charniak's (most systems)
  - CCG parses ([Gildea & Hockenmaier 03],[Pradhan et al. 05])
  - TAG parses ([Chen & Rambow 03])



- Shallow parsers

  [$_{NP}$Yesterday] , [$_{NP}$Kristina] [$_{VP}$hit] [$_{NP}$Scott] [$_{PP}$with] [$_{NP}$a baseball].

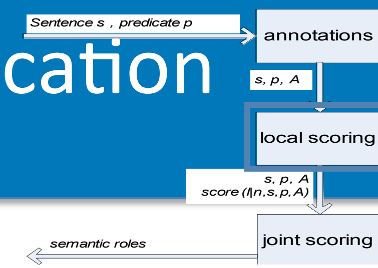
- Semantic ontologies (WordNet, automatically derived), and named entity classes

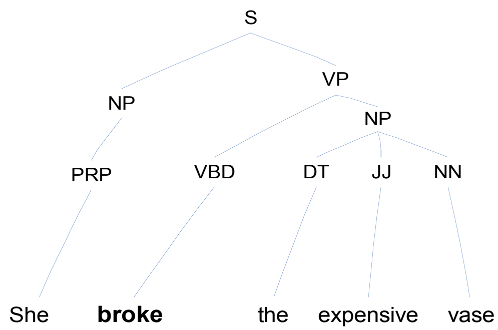
  (v) **hit** (cause to move by striking)

  WordNet hypernym

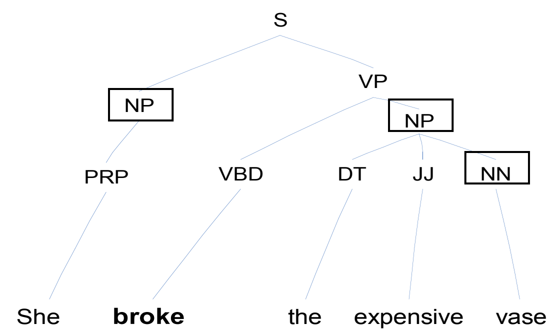
  propel, impel *(*cause to move forward with force*)*

Combining Identification and Classification Models

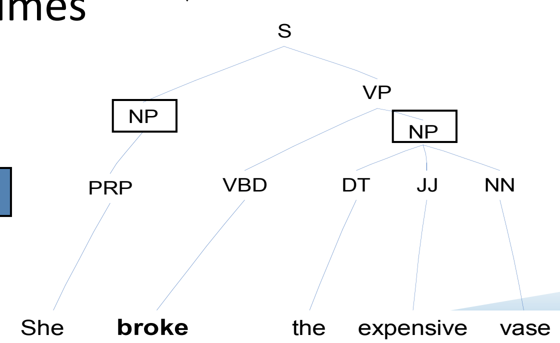




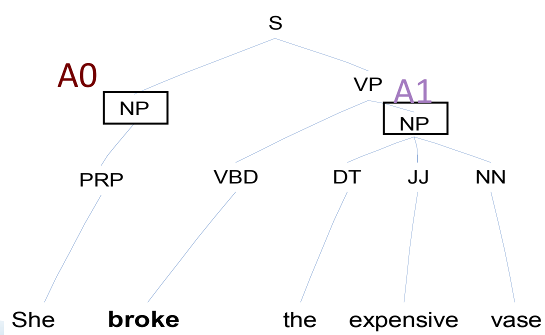
Step 1. *Pruning.* Using a hand-specified filter.



Step 2. *Identification.* Identification model (filters out candidates with high probability of NONE)

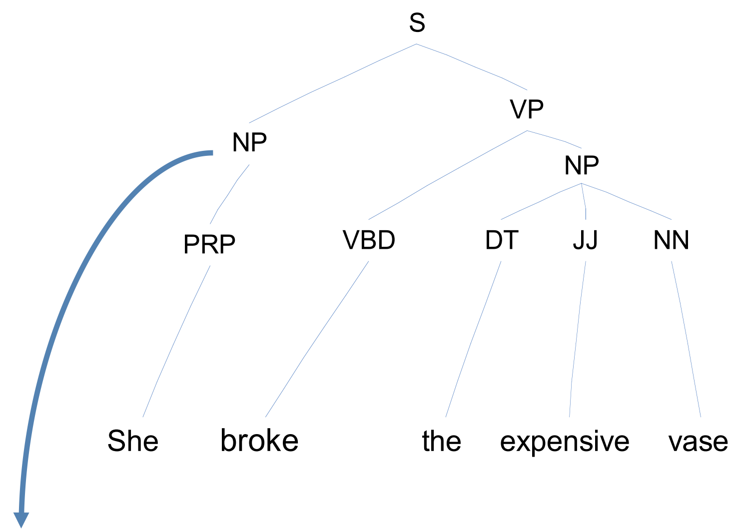


Step 3. *Classification.* Classification model assigns one of the argument labels to selected nodes (or sometimes possibly NONE)



Gildea & Jurafsky (2002) Features

- Key early work
  - Future systems use these features as a baseline
- Constituent Independent
  - Target predicate (lemma)
  - Voice
  - Subcategorization
- Constituent Specific
  - Path
  - Position (*left, right*)
  - Phrase Type
  - Governing Category (*S* or *VP*)
  - Head Word

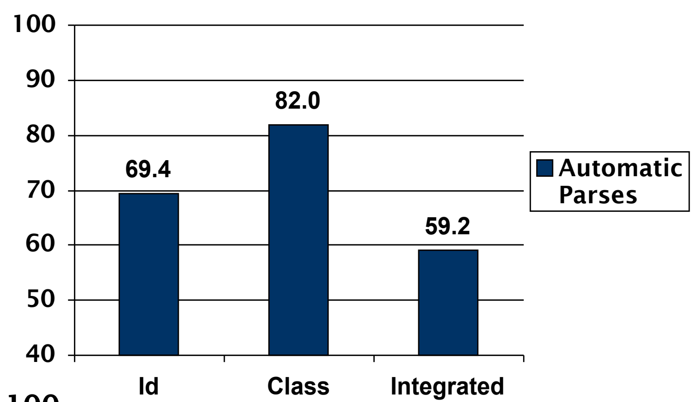


Target	*broke*
Voice	*active*
Subcategorization	*VP→VBD NP*
Path	*VBD↑VP↑S↓NP*
Position	*left*
Phrase Type	*NP*
Gov Cat	*S*
Head Word	*She*

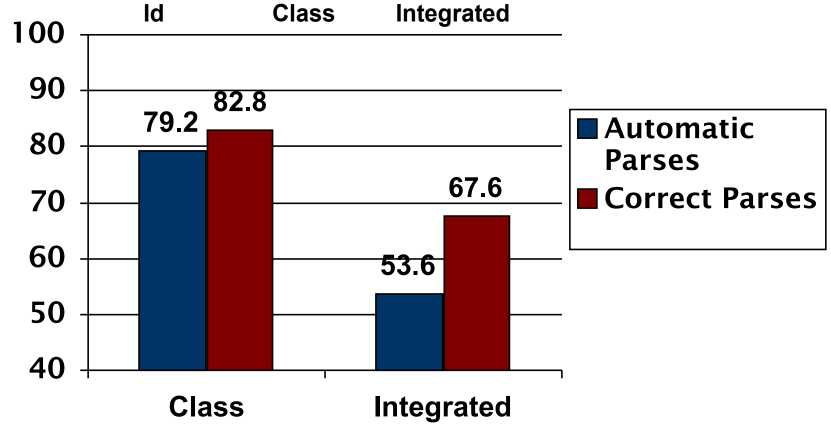
Performance with Baseline Features using the G&J Model

- **Machine learning algorithm:** interpolation of relative frequency estimates based on subsets of the 7 features introduced earlier

FrameNet Results



Propbank Results



Per Argument Performance

CoNLL-05 Results on WSJ-Test

- Core Arguments (Freq. ~70%)

	Best F_1	Freq.
A0	88.31	25.58%
A1	79.91	35.36%
A2	70.26	8.26%
A3	65.26	1.39%
A4	77.25	1.09%

- Adjuncts (Freq. ~30%)

	Best F_1	Freq.
TMP	78.21	6.86%
ADV	59.73	3.46%
DIS	80.45	2.05%
MNR	59.22	2.67%
LOC	60.99	2.48%
MOD	98.47	3.83%
CAU	64.62	0.50%
NEG	98.91	1.36%

Arguments that need to be improved

Data from Carreras&Màrquez's slides (CoNLL 2005)

What is Feature selection ?

- Feature selection:
  Problem of selecting some subset of a learning algorithm's input variables upon which it should focus attention, while ignoring the rest
  (DIMENSIONALITY REDUCTION)

- Humans/animals do this constantly

Nomenclature

- **Univariate method**: considers one variable (feature) at a time.
- **Multivariate method:** considers subsets of variables (features) together.
- **Filter method:** ranks features or feature subsets independently of the predictor (classifier).
- **Wrapper method:** uses a classifier to assess features or feature subsets.

Feature Selection in ML ?

Why even think about Feature Selection in ML?

- The information about the target class is **inherent in the variables**!

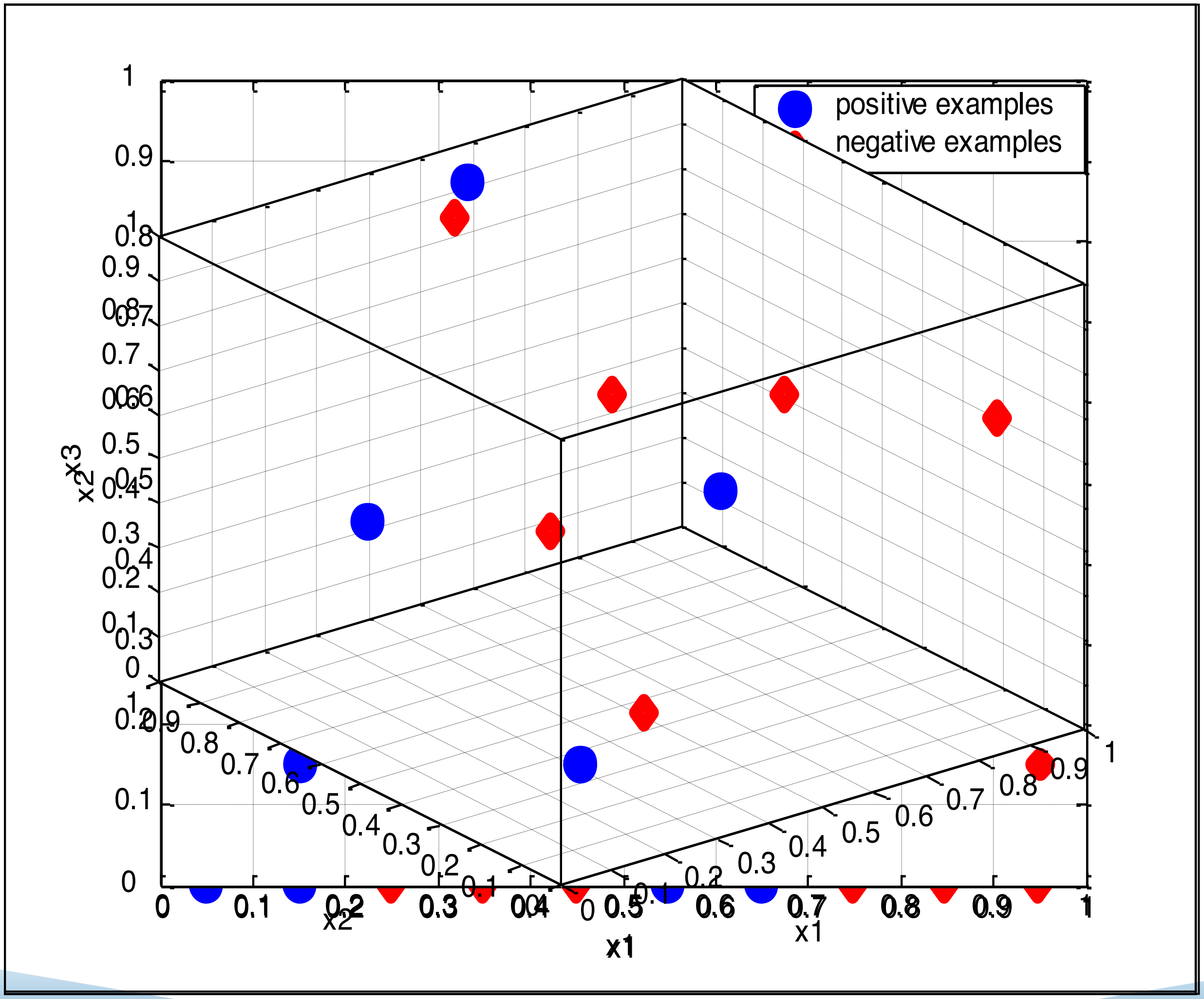
- Naive theoretical view:
  More features
  => More information
  => More discrimination power.

- In practice:
  many reasons why this is not the case!
- Also:
  Optimization is (usually) good, so why not try to optimize the input-coding ?

Feature Selection in ML

- Many explored domains have hundreds to tens of thousands of variables/features with many irrelevant and redundant ones
- In domains with many features the underlying probability distribution can be very complex and very hard to estimate (e.g. dependencies between variables)
- Irrelevant and redundant features can confuse learners
- Limited training data
- Limited computational resources
- **Curse of dimensionality**

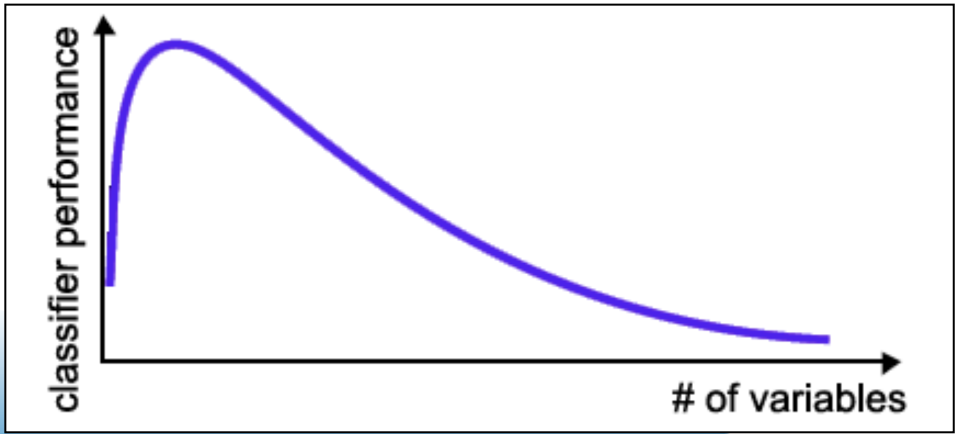
Curse of dimensionality



Curse of dimensionality

- The required number of samples (to achieve the same accuracy) grows exponentially with the number of variables!
- In practice: number of training examples is fixed!

  => the classifier's performance usually will degrade for a large number of features!



In many cases the information that is lost by discarding variables is made up for by a more accurate mapping/sampling in the lower-dimensional space !

Example for ML-Problem

Gene selection from microarray data

- Variables:
  gene expression coefficients corresponding to the amount of mRNA in a patient 's sample (e.g. tissue biopsy)
- Task: Separate healthy patients from cancer patients
- Usually there are only about 100 examples (patients) available for training and testing (!!!)
- Number of variables in the raw data: 6.000 – 60.000
- Does this work ?

Example for ML-Problem

Text-Categorization

- Documents are represented by a vector of dimension the size of the vocabulary containing word frequency counts

- Vocabulary ~ 15,000 words (i.e. each document is represented by a 15,000-dimensional vector)

- Typical tasks:
  - Automatic sorting of documents into web-directories
  - Detection of spam-email

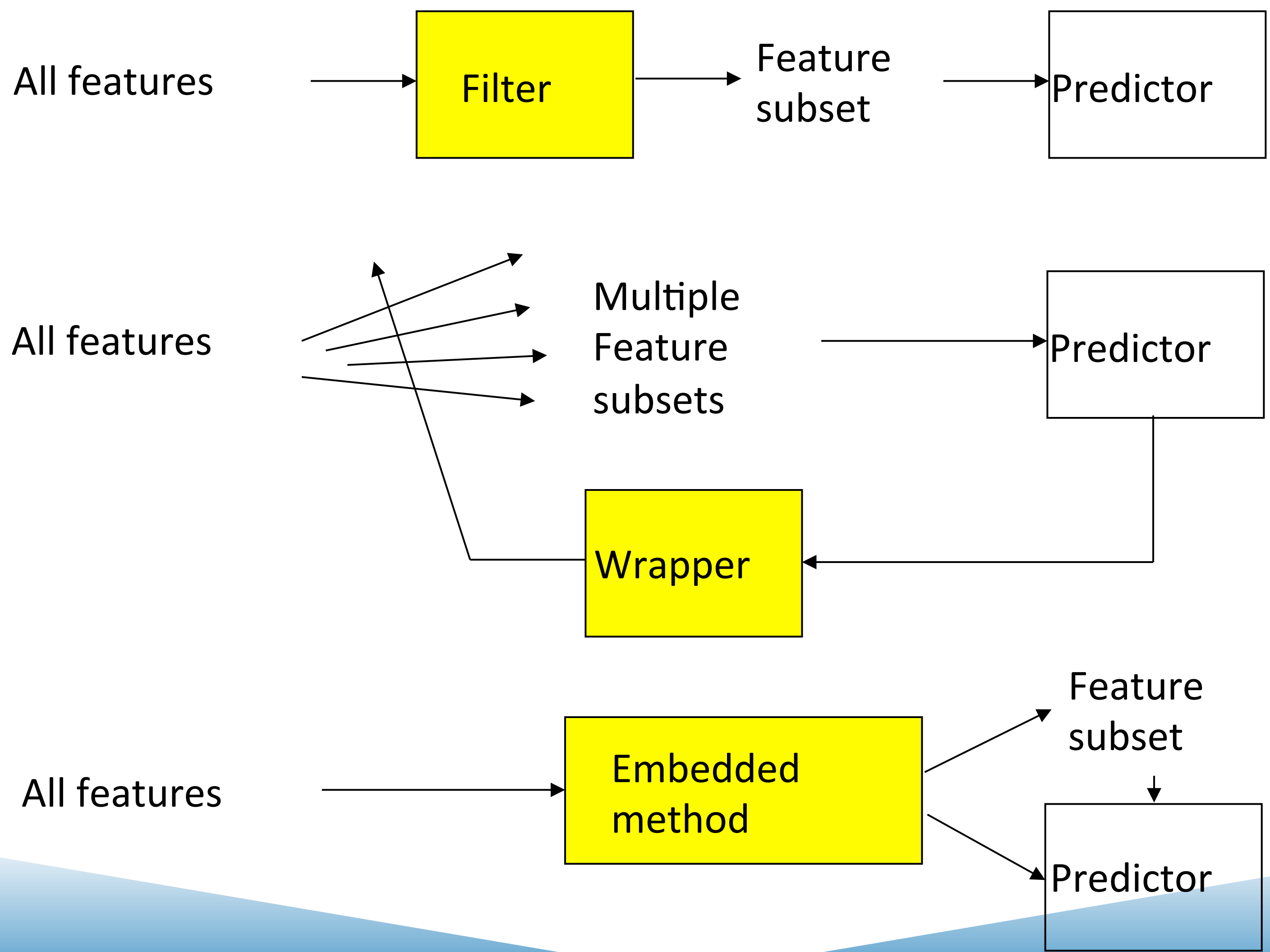
Motivation

- Especially when dealing with a large number of variables there is a need for **dimensionality reduction**
- Feature Selection can significantly improve a learning algorithm's performance

Approaches

- Wrapper
  - feature selection takes into account the contribution to the performance of a given type of classifier
- Filter
  - feature selection is based on an evaluation criterion for quantifying how well feature (subsets) discriminate the two classes
- Embedded
  - feature selection is part of the training procedure of a classifier (e.g. decision trees)

Filters, Wrappers, and Embedded methods



Filters

Methods:

- Criterion: Measure feature/feature subset "relevance"
- Search: Usually order features (individual feature ranking or nested subsets of features)
- Assessment: Use statistical tests

Results:

- Are (relatively) robust against overfitting
- May fail to select the most "useful" features

Wrappers

Methods:

- Criterion: Measure feature subset "usefulness"
- Search: Search the space of all feature subsets
- Assessment: Use cross-validation

Results:

- Can in principle find the most "useful" features, but
- Are prone to overfitting

Embedded Methods

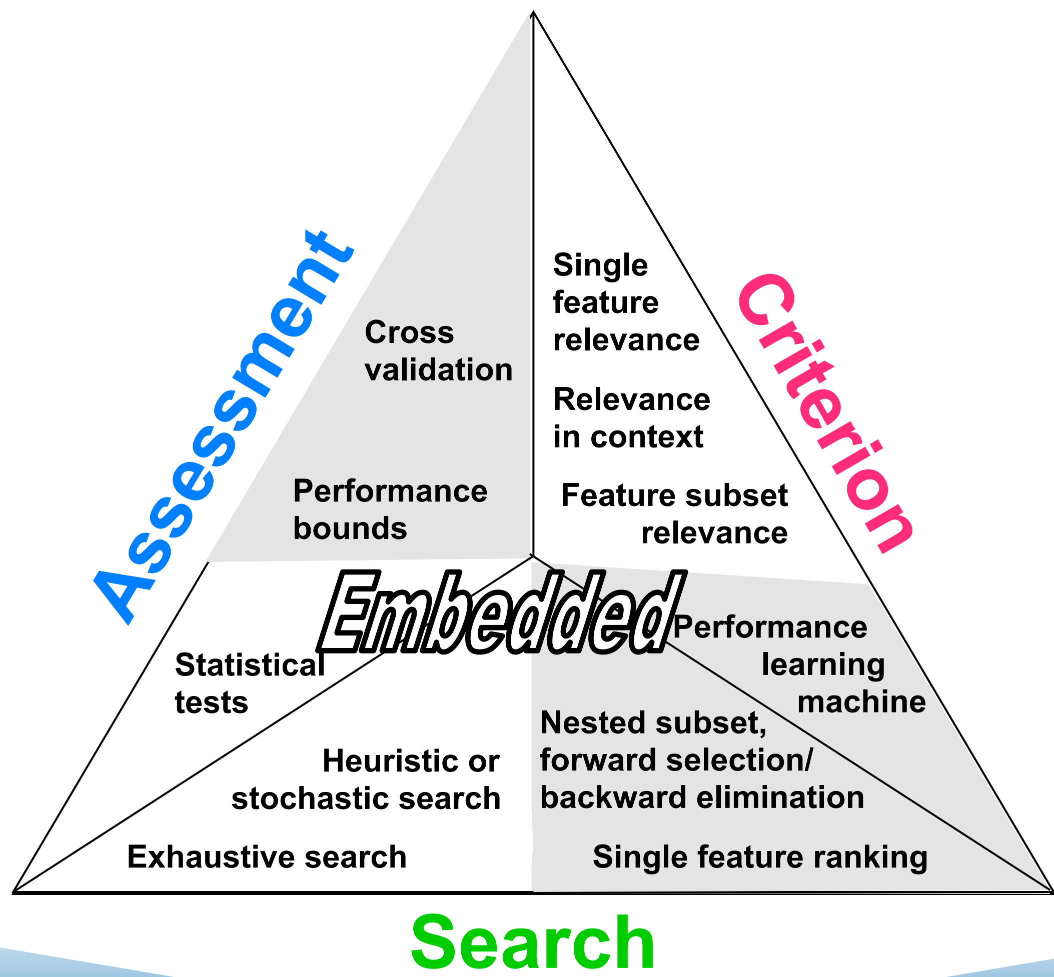
Methods:

- Criterion: Measure feature subset "usefulness"
- Search: **Search guided by the learning process**
- Assessment: Use cross-validation

Results:

- Similar to wrappers, but
- Less computationally expensive
- Less prone to overfitting

Three "Ingredients"

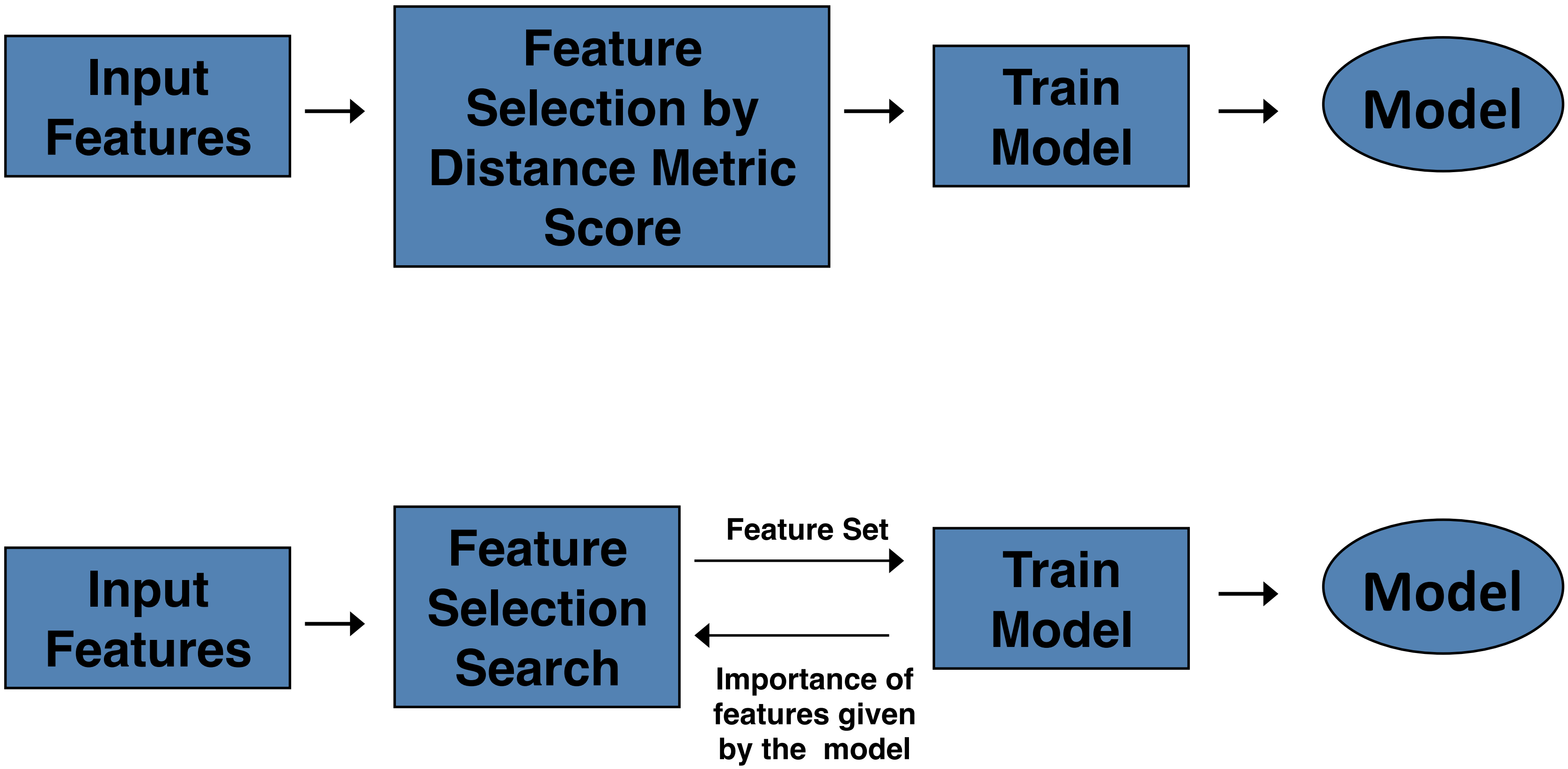


Embedded methods

- Attempt to jointly or simultaneously train both a classifier and a feature subset
- Often optimize an objective function that jointly rewards accuracy of classification and penalizes use of more features.
- Intuitively appealing

Example: tree-building algorithms

Approaches to Feature Selection



Adapted from Shin and Jasso

Filter methods

$\mathbf{R}^p \longrightarrow$ Feature selection $\longrightarrow \mathbf{R}^s \longrightarrow$ Classifier design

$s \ll p$

•Features are scored independently and the top s are used by the classifier

•Score: correlation, mutual information, t-statistic, F-statistic, p-value, tree importance statistic etc

Easy to interpret. Can provide some insight into the class markers.

Adapted from J. Fridlyand

Problems with filter method

- Redundancy in selected features: features are considered independently and not measured on the basis of whether they contribute new information
- Interactions among features generally can not be explicitly incorporated (some filter methods are smarter than others)
- Classifier has no say in what features should be used: some scores may be more appropriates in conjuction with some classifiers than others.

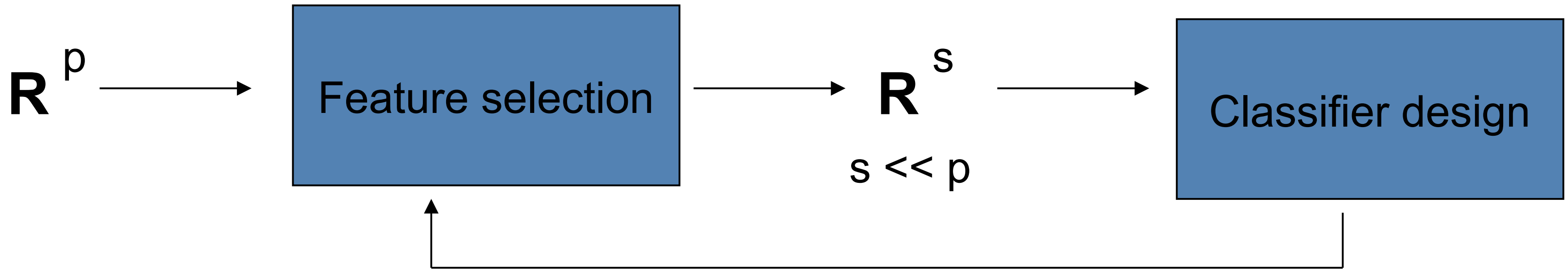
Adapted from J. Fridlyand

Dimension reduction: a variant on a filter method

- Rather than retain a subset of s features, perform dimension reduction by projecting features onto s principal components of variation (e.g. PCA etc)
- Problem is that we are no longer dealing with one feature at a time but rather a linear or possibly more complicated combination of all features.
  - It may be good enough for a black box but how does one build a diagnostic chip on a "supergene"? (even though we don't want to confuse the tasks)
- Those methods tend not to work better than simple filter methods.

Adapted from J. Fridlyand

Wrapper methods



•Iterative approach: many feature subsets are scored based on classification performance and best is used.

•Selection of subsets: forward selection, backward selection, Forward-backward selection, tree harvesting etc

Adapted from J. Fridlyand

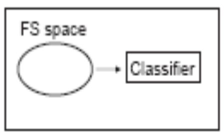
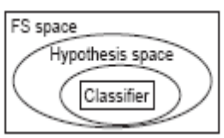
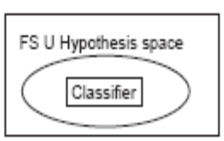
Problems with wrapper methods

- Computationally expensive: for each feature subset to be considered, a classifier must be built and evaluated
- No exhaustive search is possible: generally greedy algorithms only.
- Easy to overfit.

Adapted from J. Fridlyand

Feature Selection techniques in a nutshell

Table 1. A taxonomy of feature selection techniques. For each feature selection type, we highlight a set of characteristics which can guide the choice for a technique suited to the goals and resources of practitioners in the field.

	Model search	Advantages		Disadvantages	Examples
Filter		Univariate	Fast Scalable Independent of the classifier	Ignores feature dependencies Ignores interaction with the classifier	Chi-square Euclidean distance t-test Information gain, Gain ratio [6]
		Multivariate	Models feature dependencies Independent of the classifier Better computational complexity than wrapper methods	Slower than univariate techniques Less scalable than univariate techniques Ignores interaction with the classifier	Correlation based feature selection (CFS) [45] Markov blanket filter (MBF) [62] Fast correlation based feature selection (FCBF) [136]
Wrapper		Deterministic	Simple Interacts with the classifier Models feature dependencies Less computationally intensive than randomized methods	Risk of over fitting More prone than randomized algorithms to getting stuck in a local optimum (greedy search) Classifier dependent selection	Sequential forward selection (SFS) [60] Sequential backward elimination (SBE) [60] Plus q take-away r [33] Beam search [106]
		Randomized	Less prone to local optima Interacts with the classifier Models feature dependencies	Computationally intensive Classifier dependent selection Higher risk of overfitting than deterministic algorithms	Simulated annealing Randomized hill climbing [110] Genetic algorithms [50] Estimation of distribution algorithms [52]
Embedded		Interacts with the classifier Better computational complexity than wrapper methods Models feature dependencies		Classifier dependent selection	Decision trees Weighted naive Bayes [28] Feature selection using the weight vector of SVM [44, 125]

Saeys Y, Inza I, Larrañaga P. A review of feature selection techniques in bioinformatics Bioinformatics. 2007 Oct 1;23(19):2507-17

Creating attribute-value table

	f_1	f_2	…	f_K	y
x_1					
x_2					
…					

- Choose features:
  - Define feature templates
  - Instantiate the feature templates
  - Dimensionality reduction: feature selection

- Feature weighting
  - The weight for f_k: the whole column
  - The weight for f_k in d_i: a cell

An example: text classification task

- Define feature templates:
  - One template only: word
- Instantiate the feature templates
  - All the words appeared in the training (and test) data
- Dimensionality reduction: feature selection
  - Remove stop words
- Feature weighting
  - Feature value: term frequency (tf), or tf-idf

Dimensionality reduction (DR)

- What is DR?
  - Given a feature set r, create a new set r', s.t.
    - r' is much smaller than r, and
    - the classification performance does not suffer too much.

- Why DR?
  - ML algorithms do not scale well.
  - DR can reduce overfitting.

Types of DR

- r is the original feature set, r' is the one after DR.

- Local DR vs. Global DR
  - Global DR: r' is the same for every category
  - Local DR: a different r' for each category

- Term extraction vs. term selection

Term selection vs. extraction

- Term selection: r' is a subset of r
  - Wrapping methods: score terms by training and evaluating classifiers.

    ➔ expensive and classifier-dependent

  - Filtering methods

- Term extraction: terms in r' are obtained by combinations or transformation of r terms.
  - Term clustering:
  - Latent semantic indexing (LSI)

Term selection by filtering

- Main idea: scoring terms according to predetermined numerical functions that measure the "importance" of the terms.
- It is fast and classifier-independent.
- Scoring functions:
  - Information Gain
  - Mutual information
  - chi square
  - …

Basic distributions (treating features as binary)

Probability distributions on the event space of documents:

$P(t_k)$: The % of docs where t_k occurs

$P(\bar{t_k})$, $P(c_i)$, $P(\bar{c_i})$

$$P(t_k, c_i),\ P(t_k, \bar{c_i}),\ P(\bar{t_k}, c_i),\ P(\bar{t_k}, \bar{c_i}).$$

$$P(t_k|c_i),\ P(t_k|\bar{c_i}),\ P(\bar{t_k}|c_i),\ P(\bar{t_k}|\bar{c_i}).$$

Calculating basic distributions

	$\bar{c}_i$	c_i
$\bar{t}_k$	a	b
t_k	c	d

$$P(t_k, c_i) = d/N$$

$$P(t_k) = (c + d)/N, P(c_i) = (b + d)/N$$

$$P(t_k|c_i) = d/(b + d)$$

where $N = a + b + c + d$

Term selection functions

- Intuition: for a category c_i , the most valuable terms are those that are distributed most <u>differently</u> in the sets of possible and negative examples of c_i.

Term selection functions

Document frequency:

the num of docs in which t_k occurs

Pointwise mutual information:

$$MI(t_k, c_i) = log\frac{P(t_k,c_i)}{P(c_i)P(t_k)}$$

Information gain: $IG(t_k, c_i) =$

$$P(t_k, c_i)log\frac{P(t_k,c_i)}{P(c_i)P(t_k)} + P(\bar{t_k}, c_i)log\frac{P(\bar{t_k},c_i)}{P(c_i)P(\bar{t_k})}$$

Information gain

- IG(Y|X): We must transmit Y. How many bits on average would it save us if both ends of the line knew X?

- Definition:

  $IG(Y, X) = H(Y) - H(Y|X)$

Information gain**

$$\boxed{\sum_i IG(t_k, c_i)}$$
$$= \sum_{c \in C} \sum_{t \in \{t_k, \bar{t_k}\}} P(t, c) log \frac{P(t,c)}{P(c)P(t)}$$
$$= \sum_{c \in C} \sum_t P(t, c) log P(c|t)$$
$$- \sum_c \sum_t P(t, c) log P(c)$$
$$= -H(C|T) - \sum_c ((log P(c)) \sum_t P(t, c))$$
$$= -H(C|T) + H(C) = \boxed{IG(C|T)}$$

More term selection functions**

GSS coefficient:

$$GSS(t_k, c_i) = P(t_k, c_i)P(\bar{t_k}, \bar{c_i}) - P(t_k, \bar{c_i})P(\bar{t_k}, c_i)$$

NGL coefficient: N is the total number of docs

$$NGL(t_k, c_i) = \frac{\sqrt{N}\ \ GSS(t_k, c_i)}{\sqrt{P(t_k)P(\bar{t_k})P(c_i)P(\bar{c_i})}}$$

Chi-square: (one of the definitions)

$$\chi^2(t_k, c_i) = NGL(t_k, c_i)^2 = \frac{(ad-bc)^2 N}{(a+b)(a+c)(b+d)(c+d)}$$

More term selection functions**

Relevancy score:

$$RS(t_k, c_i) = log\frac{P(t_k|c_i)+d}{P(\bar{t_k}|\bar{c_i})+d}$$

Odds Ratio:

$$OR(t_k, c_i) = \frac{P(t_k|c_i)P(\bar{t_k}|\bar{c_i})}{P(\bar{t_k}|c_i)P(t_k|\bar{c_i})}$$

Global DR

- For local DR, calculate $f(t_k, c_i)$.

- For global DR, calculate one of the following:

Sum: $f_{sum}(t_k) = \sum_{i=1}^{|C|} f(t_k, c_i)$

Average: $f_{avg}(t_k) = \sum_{i=1}^{|C|} f(t_k, c_i) P(c_i)$

Max: $f_{max}(t_k) = \max_{i=1}^{|C|} f(t_k, c_i)$

|C| is the number of classes

Which function works the best?

- It depends on
  - Classifiers
  - Data
  - …

- According to (Yang and Pedersen 1997):

$$\{OR, NGL, GSS\} > \{\chi^2_{max}, IG_{sum}\} > \{\#_{avg}\} >> \{MI\}$$

Alternative feature values

- Binary features: 0 or 1.
- Term frequency (TF): the number of times that t_k appears in d_i.
- Inversed document frequency (IDF): log |D| /$d_{k,}$ where d_k is the number of documents that contain t_k.
- TFIDF = TF * IDF
- Normalized TFIDF: $$w_{ik} = \frac{tfidf(d_i, t_k)}{Z}$$

Feature weights

- Feature weight 2 {0,1}: same as DR
- Feature weight 2 R: iterative approach:
  - Ex: MaxEnt

➔ Feature selection is a special case of feature weighting.

Summary so far

- Curse of dimensionality ➔ dimensionality reduction (DR)

- DR:
  - Term extraction
  - Term selection
    - Wrapping method
    - Filtering method: different functions

Summary (cont)

- Functions:
  - Document frequency
  - Mutual information
  - Information gain
  - Gain ratio
  - Chi square
  - …