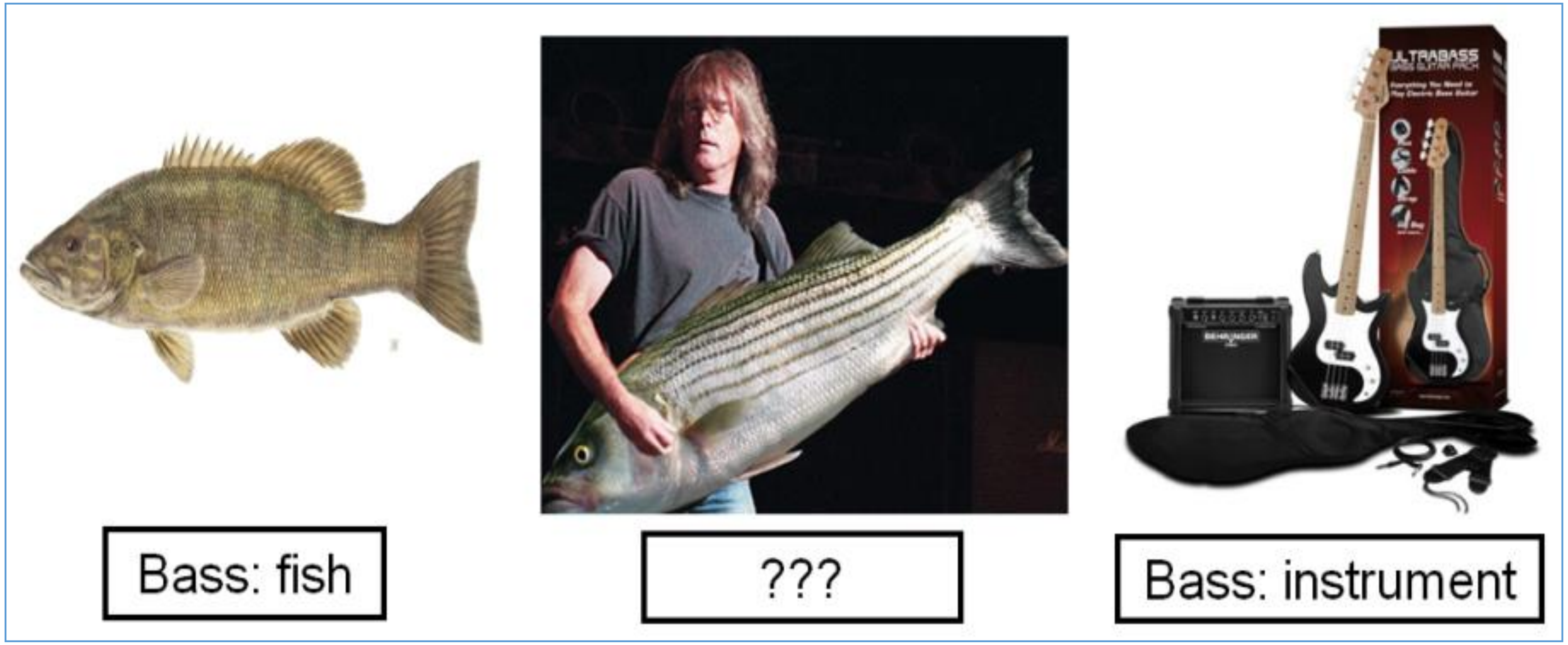
Word Sense Disambiguation

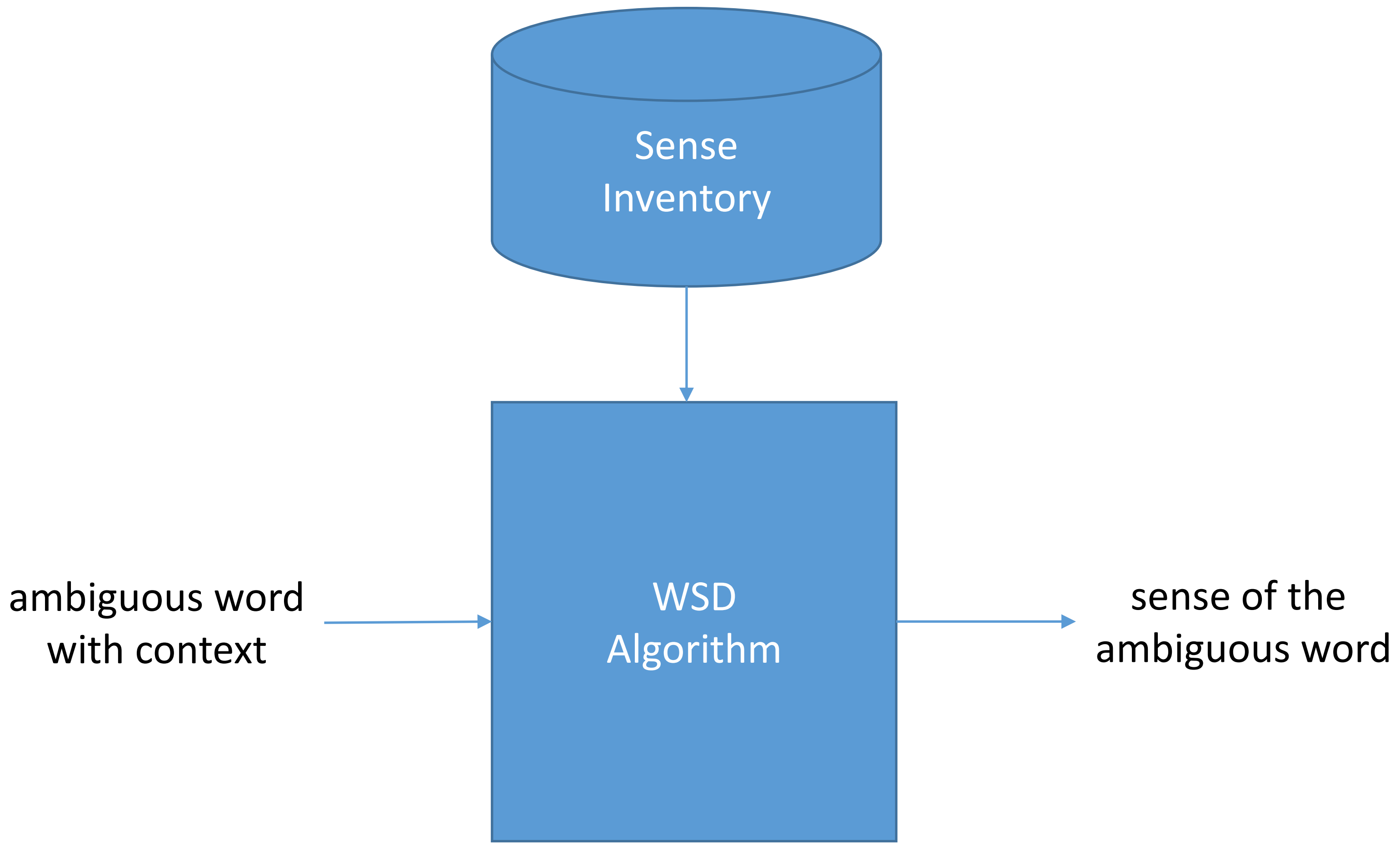
(aka the most fun of all NLP problems)

Word Sense Disambiguation

He played the bass.



Basic WSD Algorithm



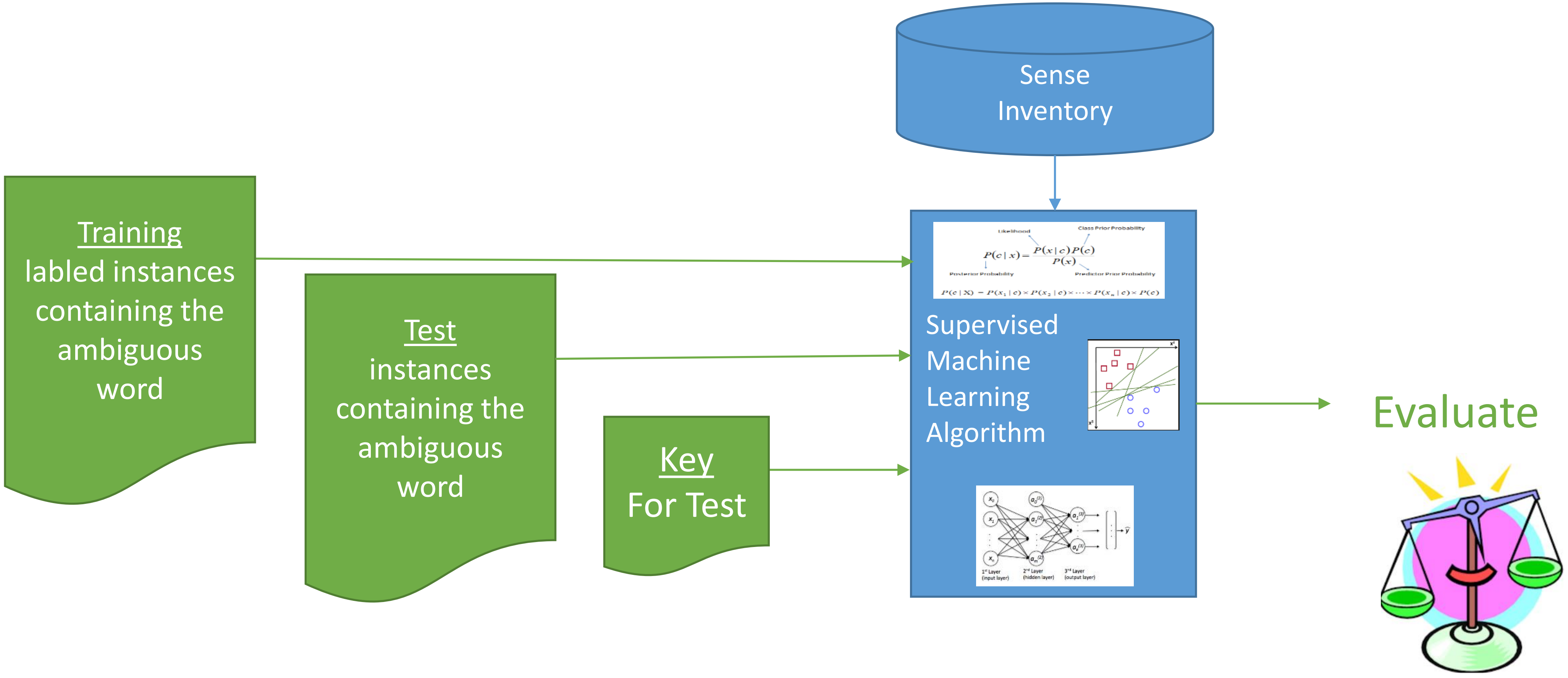
Types of WSD systems

- Supervised
- Unsupervised
- Knowledge-based

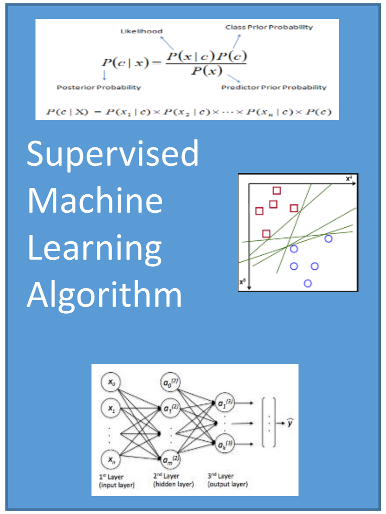
WSD
Algorithm

Supervised WSD

Learns patterns from manually annotated training data



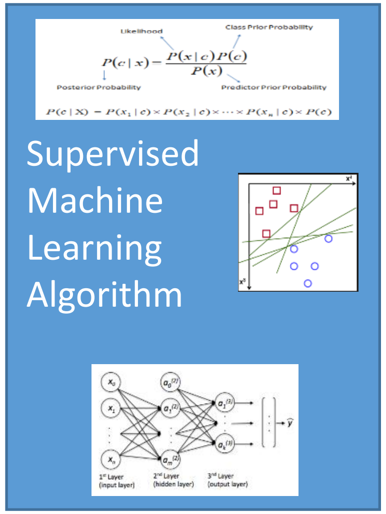
Supervised WSD



- Two components
  - Machine learning algorithm
  - Vector representation

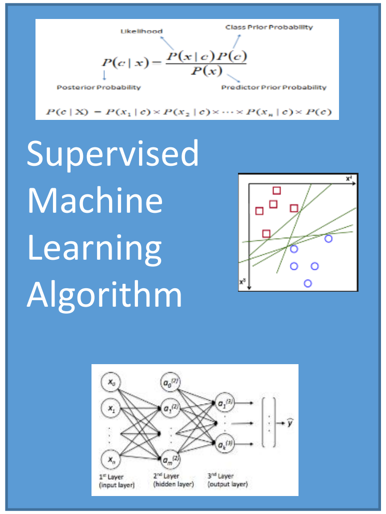
Machine learning

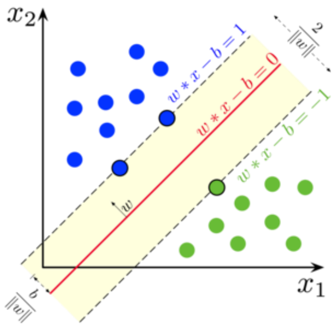
- Two ML types
  - Feature-based learning algorithms
  - Featureless learning algorithms

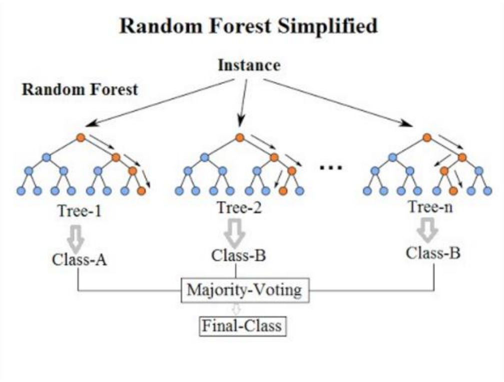


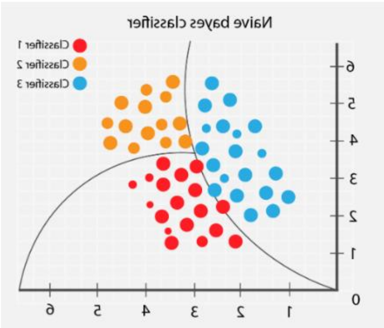
Machine learning

- Two ML types
  - **Feature-based learning algorithms**
  - Featureless learning algorithms



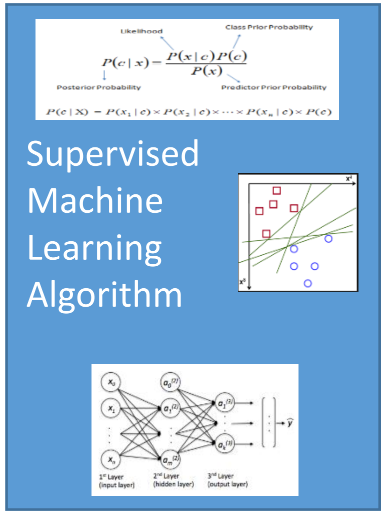


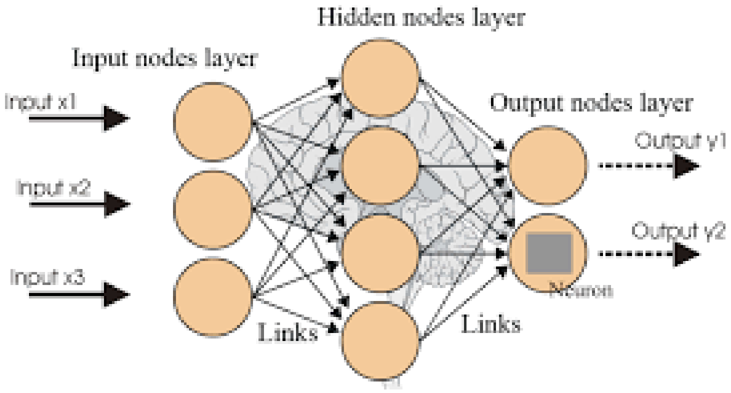


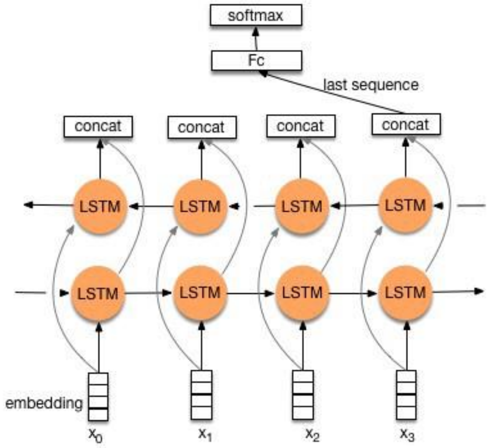


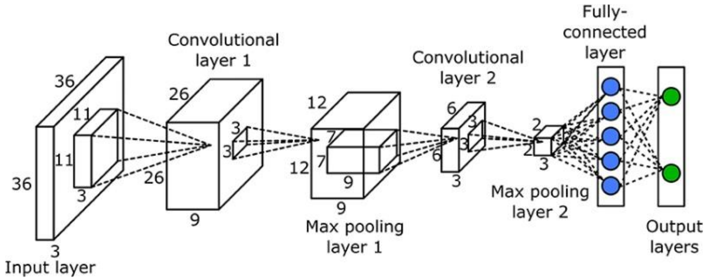
Machine learning

- Two ML types
  - Feature-based learning algorithms
  - **Featureless learning algorithms**









Vector representation

<lexelt item="line-n">
<instance id="line-n.w9_10:6830:">
<answer instance="line-n.w9_10:6830:" senseid="phone"/>
<context>
<s> In contrast, the California economy is booming, with 4.5% access
<head>line</head> growth in the past year. </s>
</context>
</instance>

Specifically: how do we represent line in the instance “line-n.w9_10:6830:”

Feature vector

- Feature vector
  - Consists of numeric or nominal values that encode linguistic information

- Example features:
  - Lexical Information
    - Bag-of-words -- words surrounding the target word
    - N-grams - Extension of bag-of-words (which is just unigrams)
    - Collocation - the information about the words located to the left or right of the target word
  - Syntactic Information
    - Part of speech of the target word
    - Part of speech of the previous word
  - Semantic information
    - Concept of the surrounding words

Feature vector

- Feature vector
  - Consists of numeric or nominal values that encode linguistic information

- Example features:
  - Lexical Information
    - Bag-of-words -- words surrounding the target word
    - N-grams - Extension of bag-of-words (which is just unigrams)
    - Collocation - the information about the words located to the left or right of the target word
  - Syntactic Information
    - Part of speech of the target word
    - Part of speech of the previous word
  - Semantic information
    - Concept of the surrounding words

Algorithm "learns" what information is useful for classifying the sense of an ambiguous word

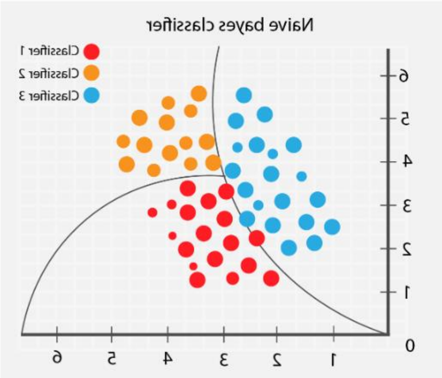
Feature vector

- Feature vector
  - Consists of numeric or nominal values that encode linguistic information

Algorithm "learns" what information is useful for classifying the sense of an ambiguous word

- Example features:
  - Lexical Information
    - Bag-of-words -- words surrounding the target word
    - N-grams - Extension of bag-of-words (which is just unigrams)
    - Collocation - the information about the words located to the left or right of the target word
  - Syntactic Information
    - Part of speech of the target word
    - Part of speech of the previous word
  - Semantic information
    - Concept of the surrounding words

$$\hat{s} = argmax_{s \in S} P(s) \prod_{j=1}^{n} P(f_j|s)$$

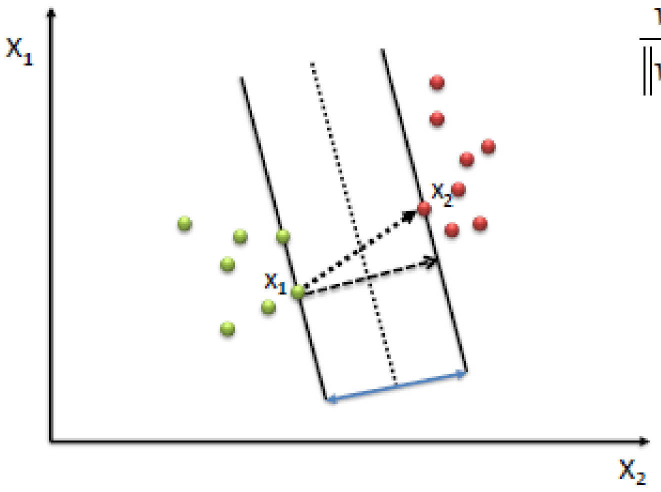


Feature vector

- Feature vector
    - Consists of numeric or nominal values that encode linguistic information

- Example features:
    - Lexical Information
        - Bag-of-words -- words surrounding the target word
        - N-grams - Extension of bag-of-words (which is just unigrams)
        - Collocation - the information about the words located to the left or right of the target word
    - Syntactic Information
        - Part of speech of the target word
        - Part of speech of the previous word
    - Semantic information
        - Concept of the surrounding words

Algorithm "learns" what information is useful for classifying the sense of an ambiguous word



$$\frac{w}{\|w\|}\cdot(x_2 - x_1) = \text{width} = \frac{2}{\|w\|}$$

$$w\cdot x_2 + b = 1$$
$$w\cdot x_1 + b = -1$$
$$w\cdot x_2 + b - w\cdot x_1 - b = 1-(-1)$$
$$w\cdot x_2 - w\cdot x_1 = 2$$
$$\frac{w}{\|w\|}(x_2 - x_1) = \frac{2}{\|w\|}$$

Feature-less representation

Still a vector but the representation is learned versus extracted

Feature-less representations:

- word embeddings
- character embeddings

Feature-less representation

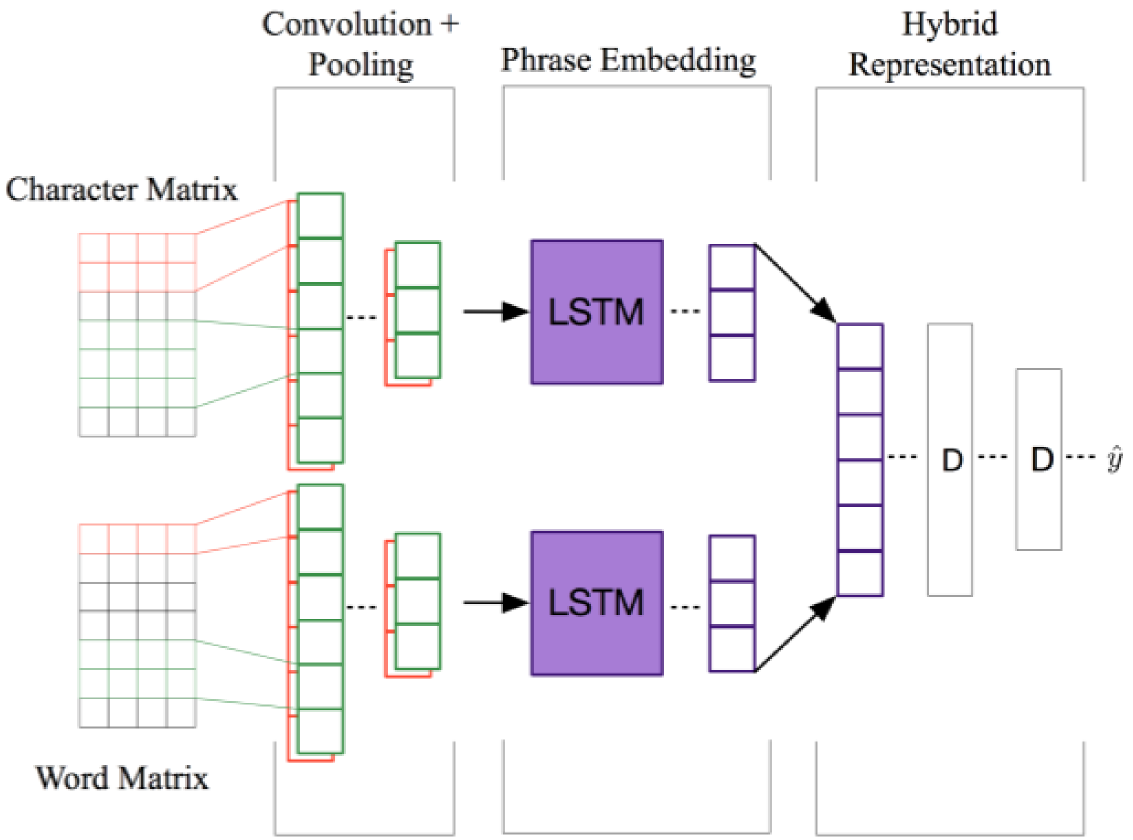
Still a vector but the representation is learned versus extracted

Feature-less representation

Still a vector but the representation is learned versus extracted

Feature-less representations:

- word embeddings
- character embeddings



Feature-less representation

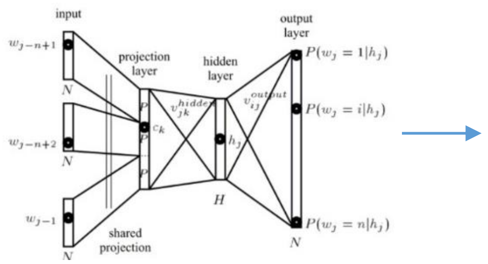
Still a vector but the representation is learned versus extracted

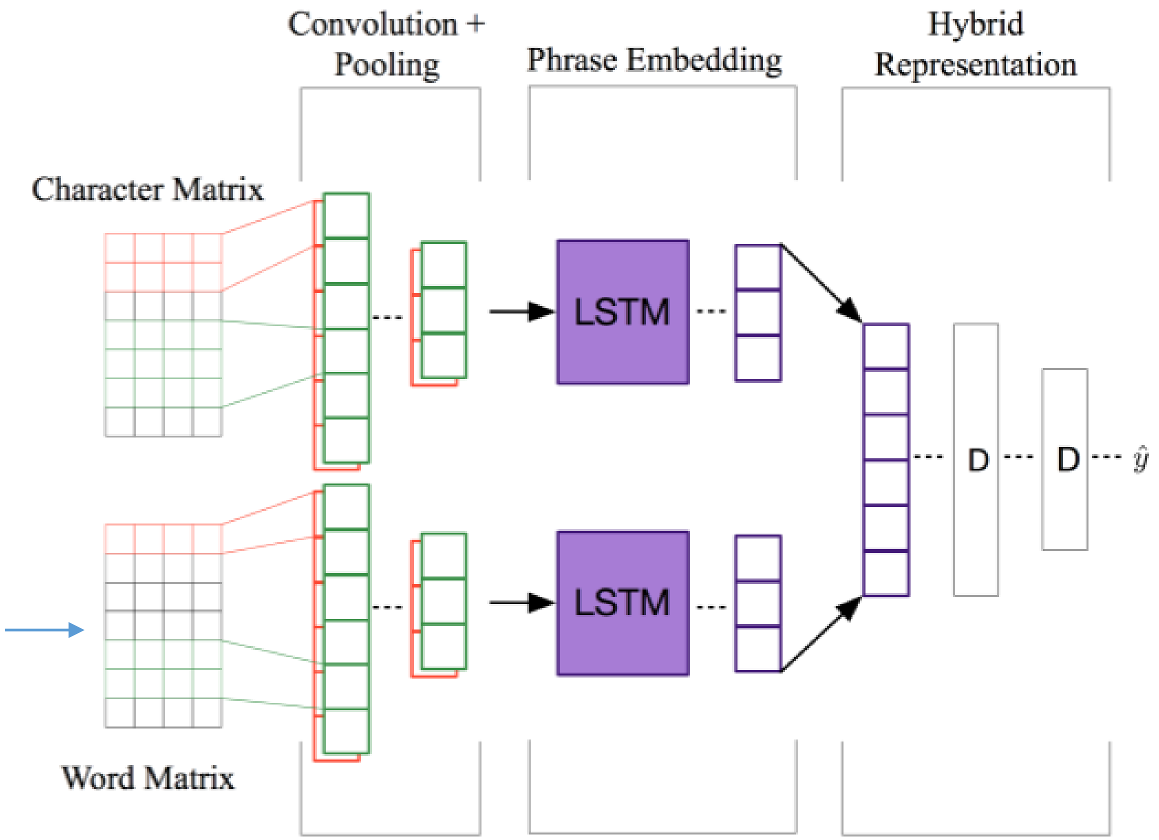
Feature-less representations:

- word embeddings
- character embeddings

Word embeddings

- *word2vec*
- glove
- BERT
- ELMO





Feature-less representation

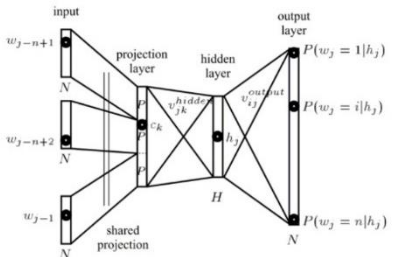
Still a vector but the representation is learned versus extracted

Feature-less representations:

- word embeddings
- character embeddings

Word embeddings

- *word2vec*
- glove
- BERT
- ELMO



Feature-less representation

Still a vector but the representation is learned versus extracted

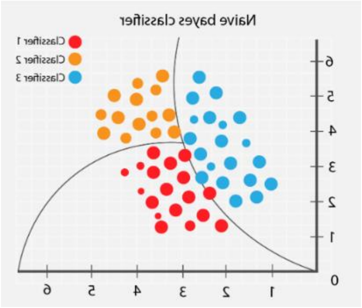
Feature-less representations:

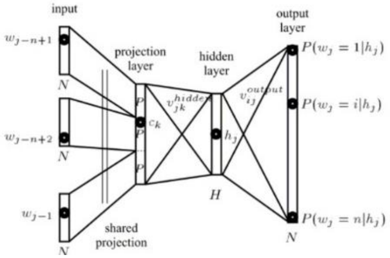
- word embeddings
- character embeddings

Word embeddings

- *word2vec*
- glove
- BERT
- ELMO

Could we use feature-less representation in a traditional machine learning algorithm?





Feature-less representation

Still a vector but the representation is learned versus extracted

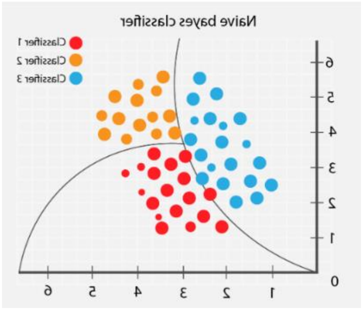
Feature-less representations:

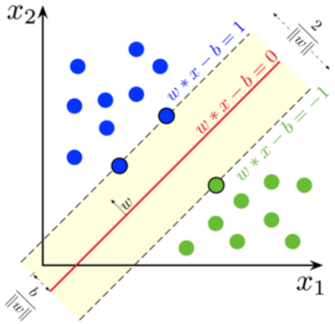
- word embeddings
- character embeddings

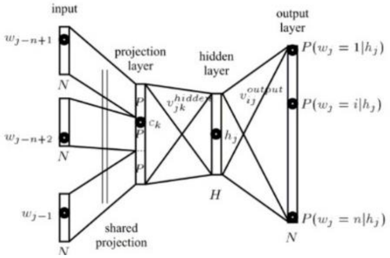
Word embeddings

- *word2vec*
- glove
- BERT
- ELMO

Could we use feature-less representation in a traditional machine learning algorithm?





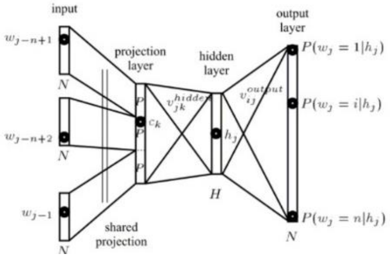


Feature-less representation

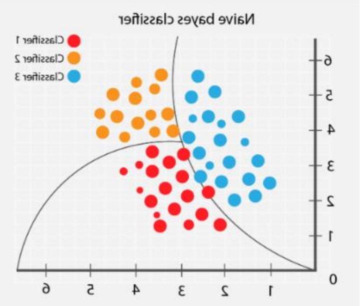
Still a vector but the representation is learned versus extracted

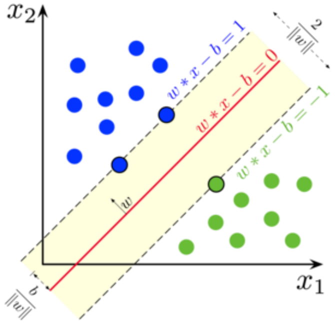
Feature-less representations:
- word embeddings
- character embeddings

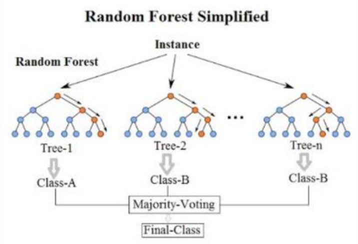
Word embeddings
- *word2vec*
- glove
- BERT
- ELMO



Could we use feature-less representation in a traditional machine learning algorithm?



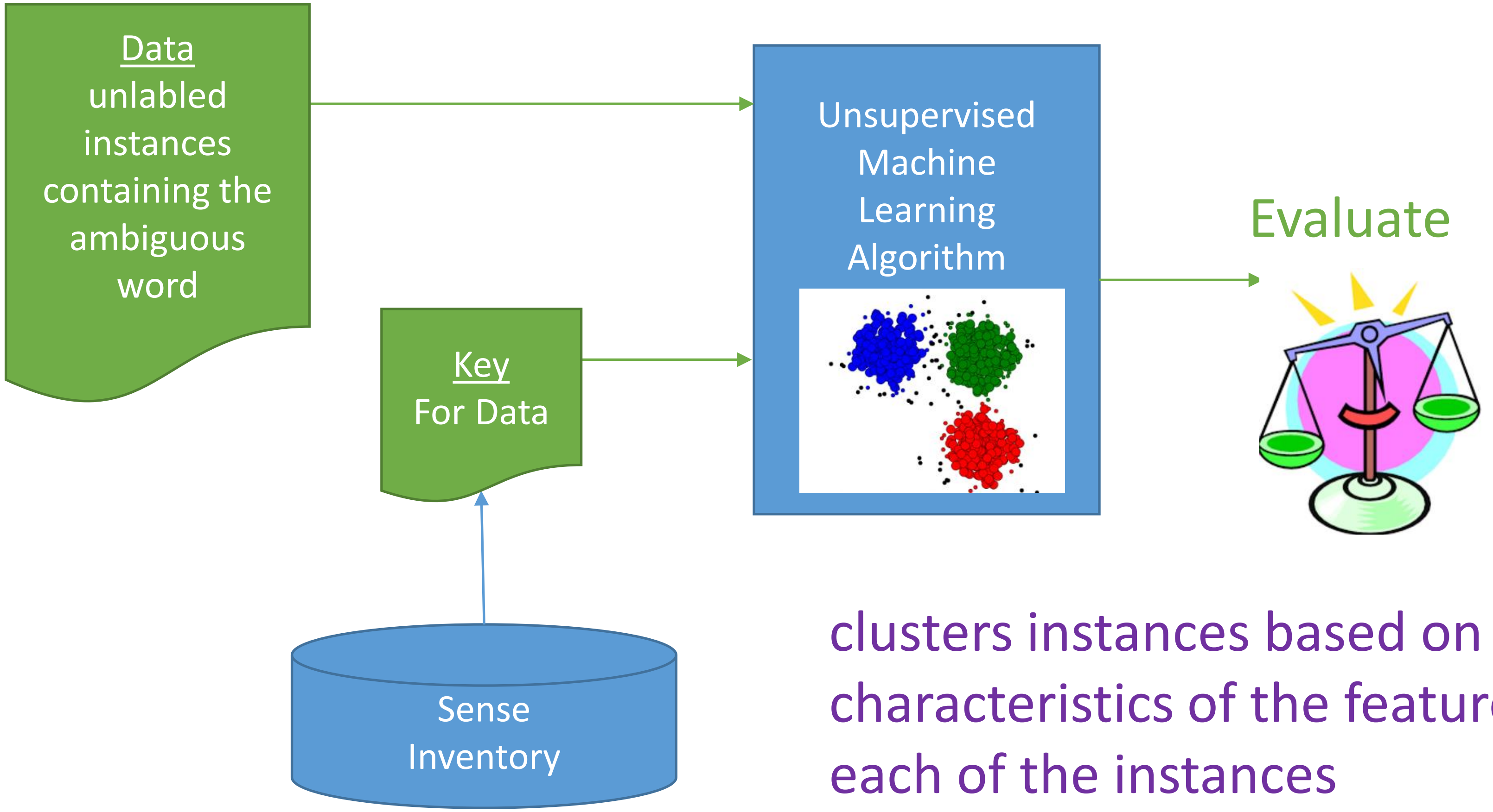




Disadvantage of Supervised Approaches

Need training data for each word that we want to disambiguate

Unsupervised WSD



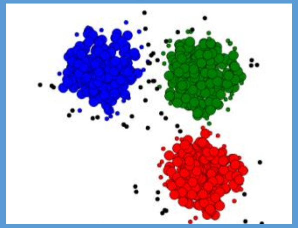
clusters instances based on the distributional characteristics of the features associated with each of the instances

Feature vector are often the same

- Feature vector
  - Consists of numeric or nominal values that encode linguistic information

- Example features:
  - Lexical Information
    - Bag-of-words -- words surrounding the target word
    - N-grams - Extension of bag-of-words (which is just unigrams)
    - Collocation - the information about the words located to the left or right of the target word
  - Syntactic Information
    - Part of speech of the target word
    - Part of speech of the previous word
  - Semantic information
    - Concept of the surrounding words

Unsupervised Machine Learning Algorithm

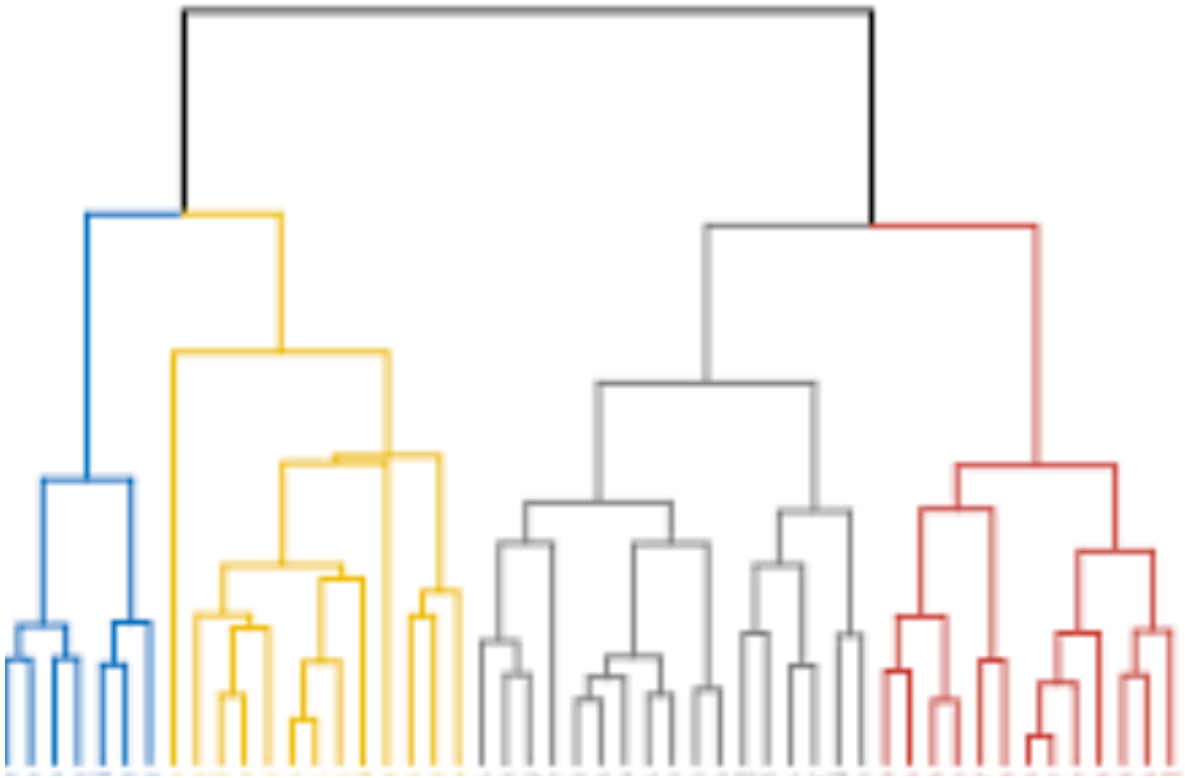


Unsupervised Algorithms

- Exclusive Clustering
- Overlapping Clustering
- **Hierarchical Clustering**
- Probabilistic Clustering

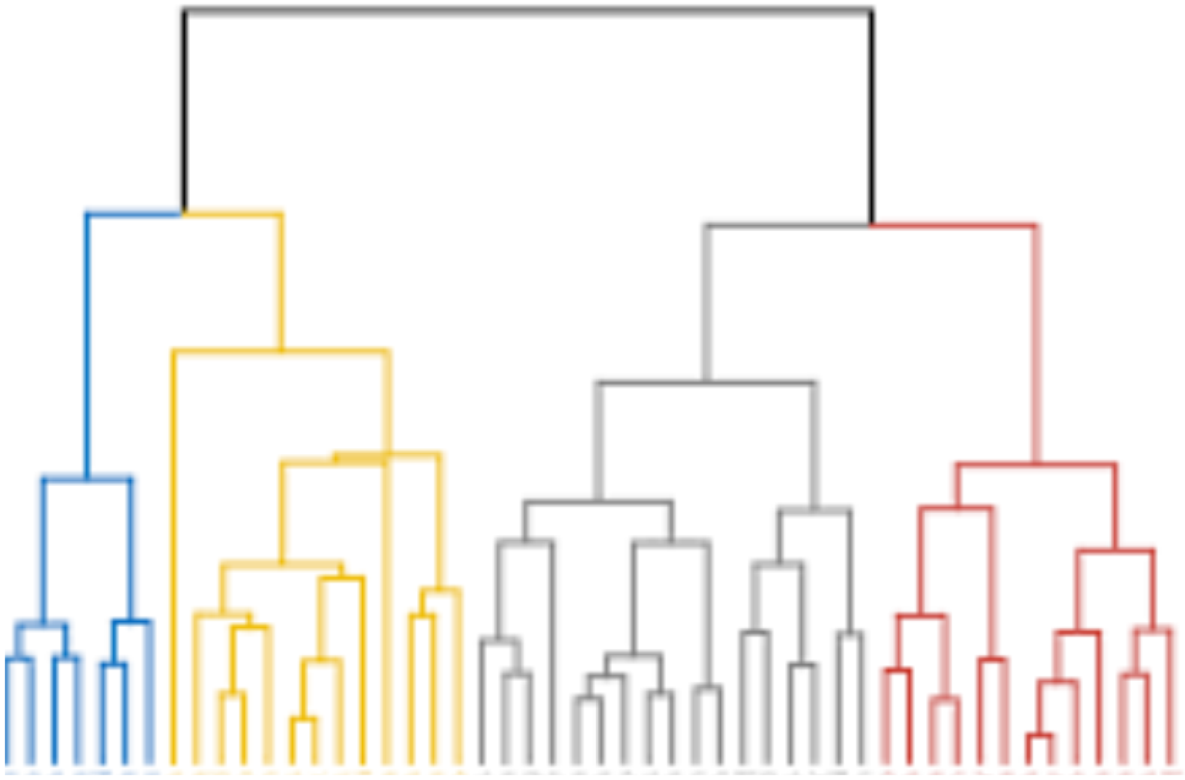
Hierarchical Clustering

- Idea: ensure nearby points end up in the same cluster
- Start with a collection C of n singleton clusters
  - each cluster contains one data point: $c_i=\{x_i\}$
- Repeat until only one cluster is left:
  - find a pair of clusters that is closest: $\min_{i,j} D(c_i, c_j)$
  - merge the clusters c_i, c_j into a new cluster c_{i+j}
  - remove c_i, c_j from the collection C, add c_{i+j}



Heirarchical Clustering

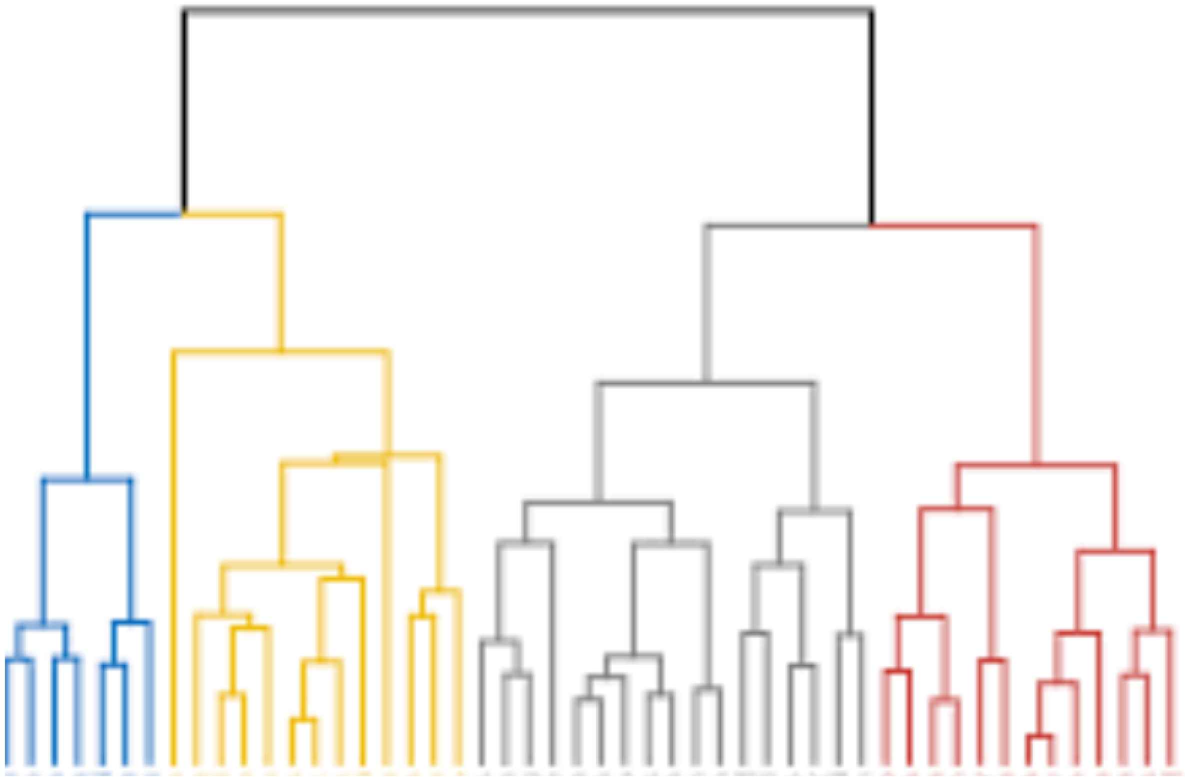
- Idea: ensure nearby points end up in the same cluster
- Start with a collection C of n singleton clusters
  - each cluster contains one data point: $c_i=\{x_i\}$
- Repeat until only one cluster is left:
  - find a pair of clusters that is closest: $\min_{i,j} D(c_i, c_j)$
  - merge the clusters c_i, c_j into a new cluster c_{i+j}
  - remove c_i,c_j from the collection C, add c_{i+j}



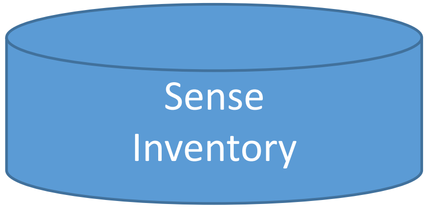
Question: how do we determine *k*?

Heirarchical Clustering

- Idea: ensure nearby points end up in the same cluster
- Start with a collection C of n singleton clusters
  - each cluster contains one data point: $c_i=\{x_i\}$
- Repeat until only one cluster is left:
  - find a pair of clusters that is closest: $\min_{i,j} D(c_i, c_j)$
  - merge the clusters c_i, c_j into a new cluster c_{i+j}
  - remove c_i, c_j from the collection C, add c_{i+j}



Question: how do we determine *k*?



Disadvantage of unsupervised algorithms

Historically they do not perform as well as supervised methods

Knowledge-based WSD

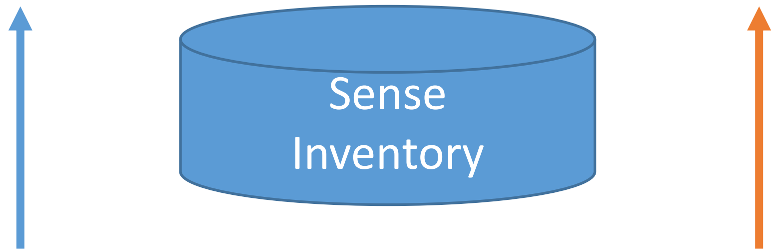
Use information from an external knowledge base and or corpora

Knowledge-based Algorithms

- MRD: uses machine readable dictionary approach
- SenseRelate: uses our similarity and relatedness measures

MRD Algorithm

A vector is created for each content word in the definition and averaged to create a single vector for that sense

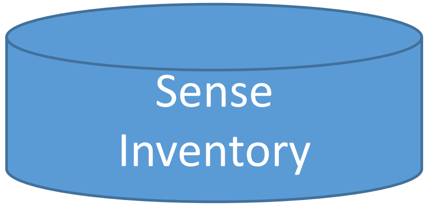


sense 1: definition

sense 2: definition

MRD Algorithm

A vector is created for each content word in the definition and averaged to create a single vector for that sense



sense 1: definition

sense 2: definition

The same is done to create a single vector for the instance containing the target word

instance containing the target word

MRD Algorithm

A vector is created for each content word in the definition and averaged to create a single vector for that sense

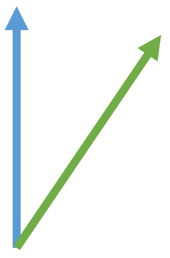
Sense Inventory

sense 1: definition

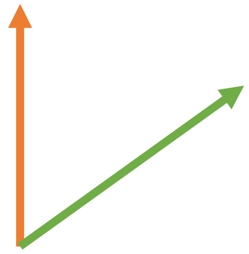
sense 2: definition

The same is done to create a single vector for the instance containing the target word

instance containing the target word



sense vector closest to the target word vector is assigned that sense



MRD Algorithm

A vector is created for each content word in the definition and averaged to create a single vector for that sense

bat 1: an implement with a handle and a solid surface, usually of wood, used for hitting the ball in games such as baseball, cricket, and table tennis.

Sense Inventory

sense 2: a mainly nocturnal mammal capable of sustained flight, with membranous wings that extend between the fingers and connecting the forelimbs to the body and the hind limbs to the tail.

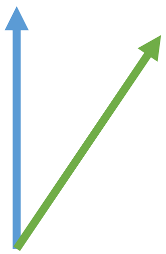
sense 1: definition

sense 2: definition

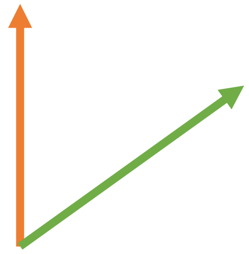
The same is done to create a single vector for the instance containing the target word

instance containing the target word

The **bat** flew through the air



sense vector closest to the target word vector is assigned that sense



SenseRelate algorithm

- Each possible sense of a **target word** is assigned a score

  [sum similarity between it and its surrounding terms]

- Assign target word the sense with highest score

SenseRelate example

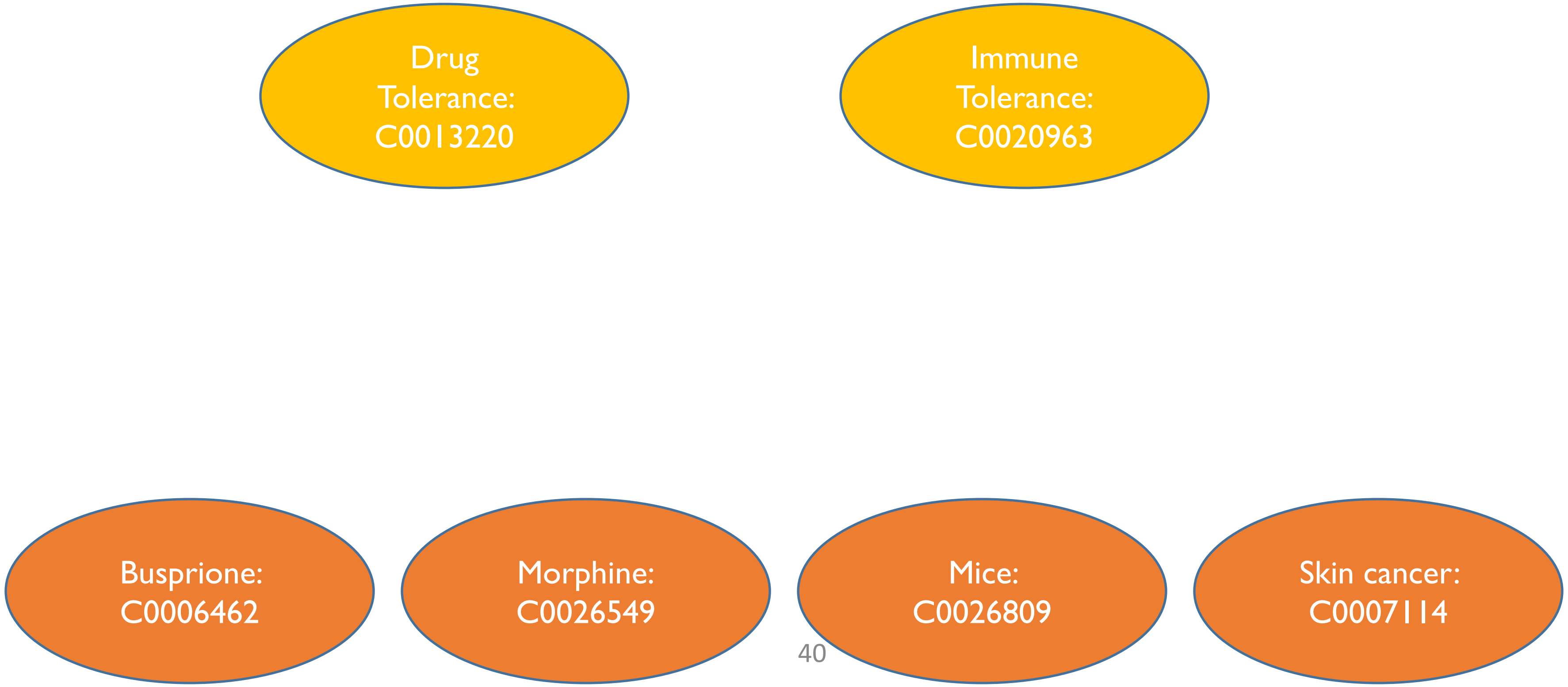
Busprione attenuates tolerance to morphine in mice with skin cancer

SenseRelate example

Busprione attenuates tolerance to morphine in mice with skin cancer

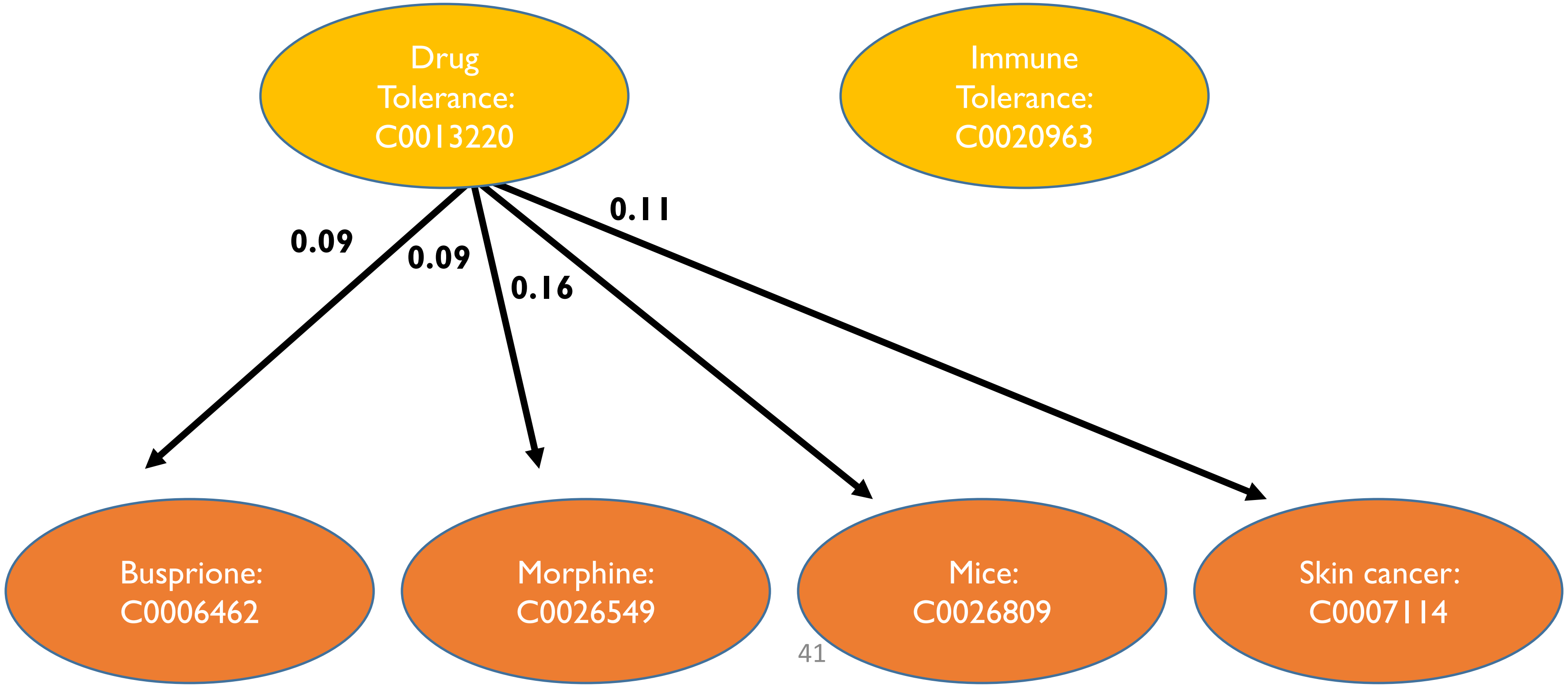
SenseRelate example

Busprione attenuates tolerance to morphine in mice with skin cancer



SenseRelate example

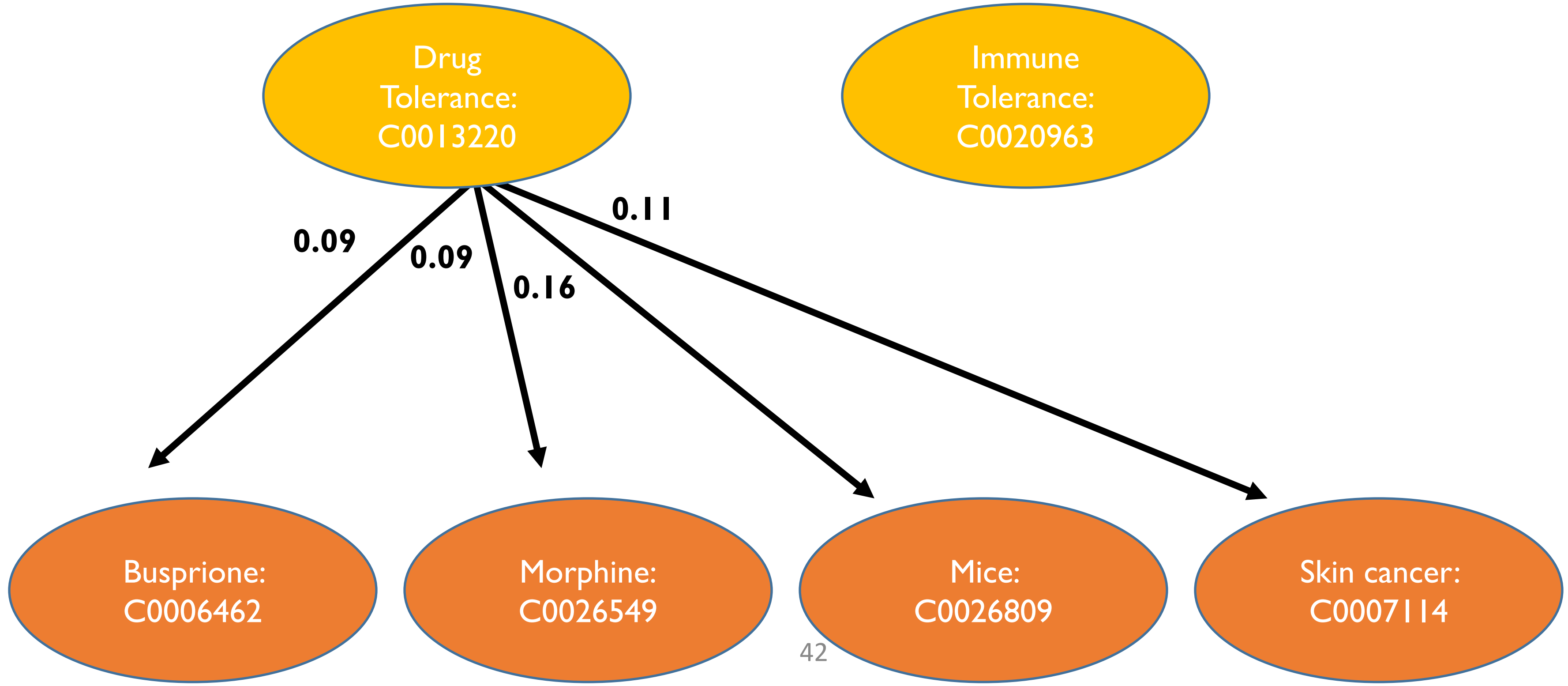
Busprione attenuates tolerance to morphine in mice with skin cancer



SenseRelate example

Busprione attenuates tolerance to morphine in mice with skin cancer

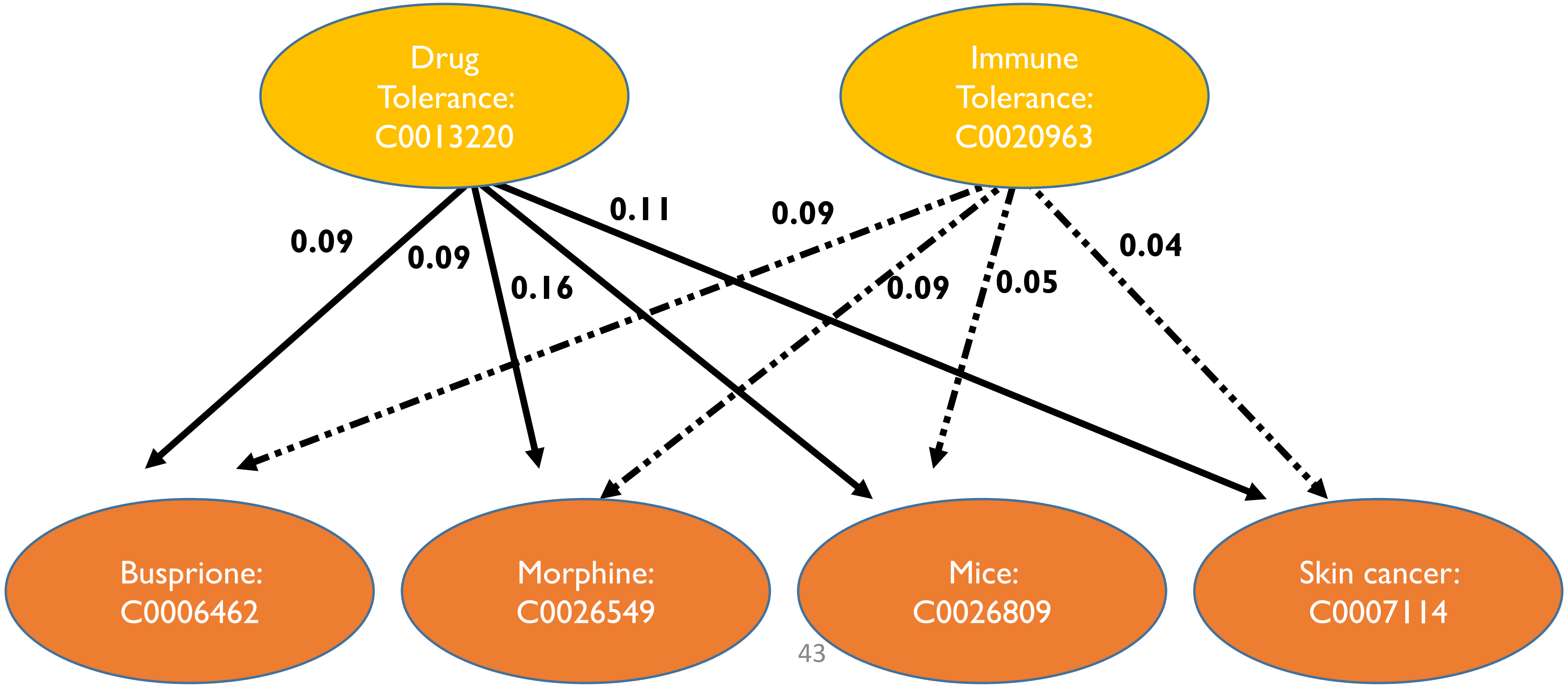
Drug Tolerance
Score = 0.09 + 0.09 + 0.16 + 0.11 = 0.45



SenseRelate example

Busprione attenuates tolerance to morphine in mice with skin cancer

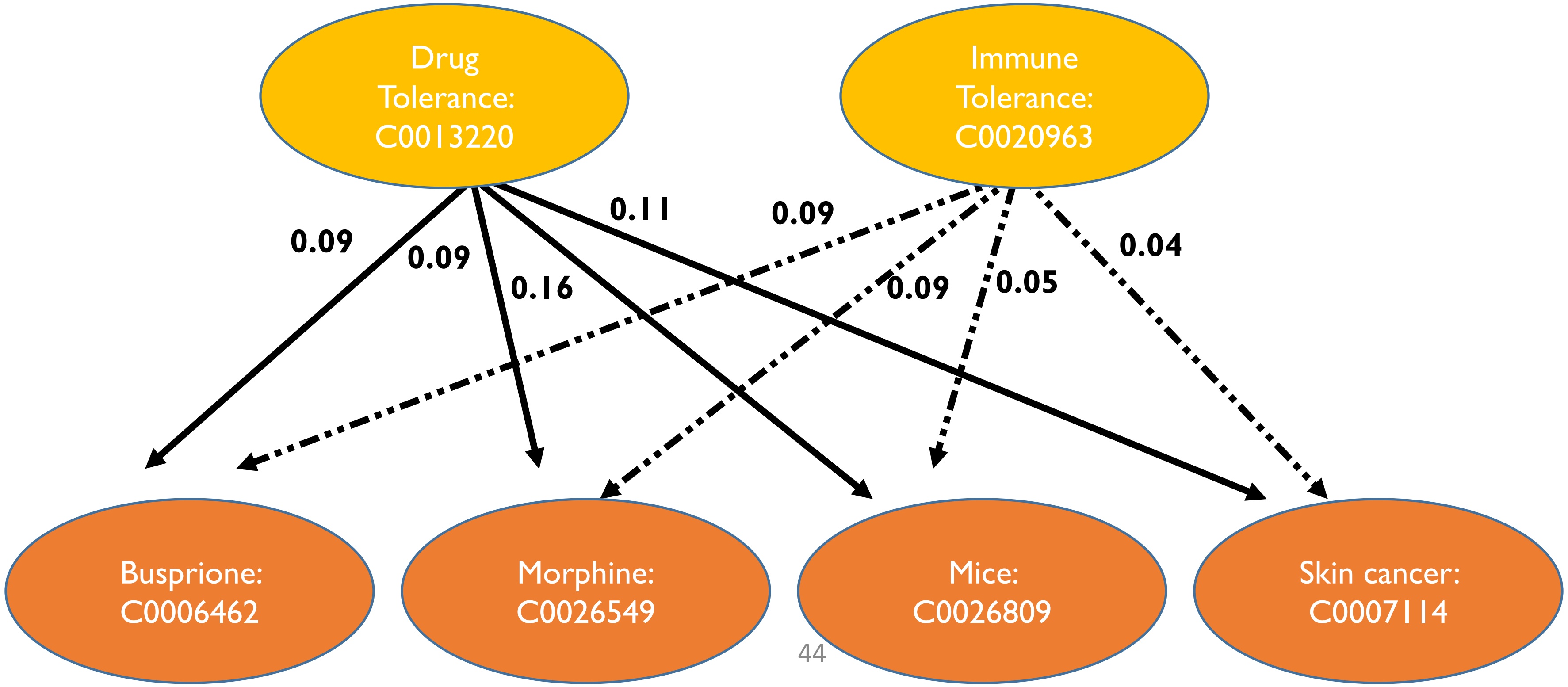
Drug Tolerance
Score = 0.09 + 0.09 + 0.16 + 0.11 = 0.45



SenseRelate example

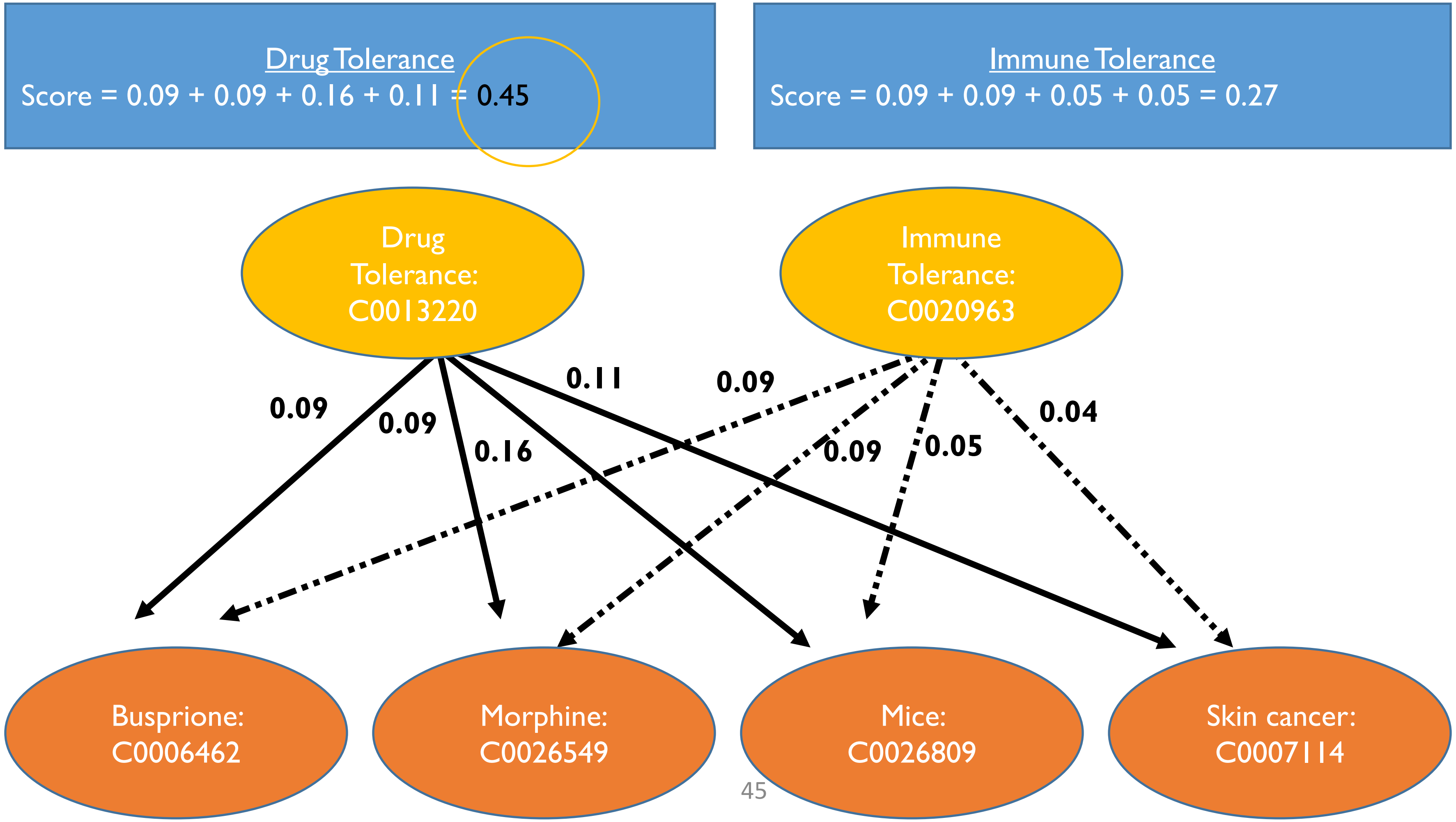
Busprione attenuates tolerance to morphine in mice with skin cancer

Drug Tolerance	Immune Tolerance
Score = 0.09 + 0.09 + 0.16 + 0.11 = 0.45	Score = 0.09 + 0.09 + 0.05 + 0.05 = 0.27



SenseRelate example

Busprione attenuates tolerance to morphine in mice with skin cancer



Sense Relate Assumption

An ambiguous word is often used in the sense that is most similar to the sense of the terms that surround it

[sum similarity between it and its surrounding terms]

Similarity and Relatedness Measures:

- Path-based similarity measures
- IC-based similarity measures
- Relatedness measures

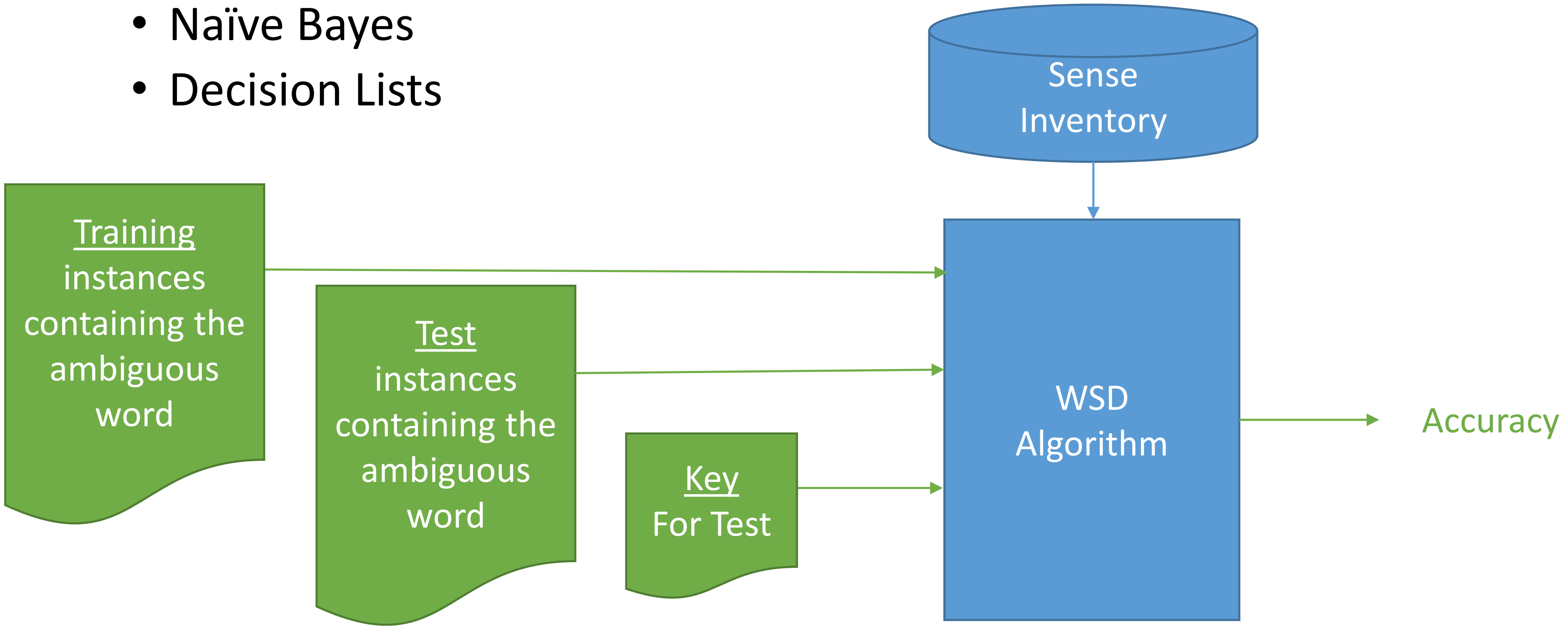
So for a quick recap: what is it that I wanted you to remember about each of the measures?

Our Focus: Supervised Machine Learning

Supervised Machine Learning

target words =
words to be
disambiguated

- Sense Inventory: small pre-selected set of target words
- Algorithms: Supervised machine learning
  - Naïve Bayes
  - Decision Lists



Training instances containing the ambiguous word

```
<lexelt item="line-n">
<instance id="line-n.w9_10:6830:">
<answer instance="line-n.w9_10:6830:" senseid="phone"/>
<context>
 <s> The New York plan froze basic rates, offered no protection to Nynex against an economic downturn that sharply cut demand and didn't offer flexible pricing. </s> <@> <s> In contrast, the California economy is booming, with 4.5% access <head>line</head> growth in the past year. </s>
</context>
</instance>
```

- Training data consists of instances containing the target word
  - Above is an example instance of the target word line

- The training data is annotated with the correct sense of line by human annotators

- Notes:
  - <head>line</head> : indicates the target word
  - senseid="phone"/> : indicates the sense annotated by the monk
  - instance="line-n.w9_10:6830:“ : indicates the instance id

Test
instances
containing the
ambiguous
word

<instance id="line-n.w8_059:8174:">
<context>
<s> Advanced Micro Devices Inc., Sunnyvale, Calif., and Siemens AG of West Germany said they agreed to jointly develop, manufacture and market microchips for data communications and telecommunications with an emphasis on the integrated services digital network. </s> <@> </p> <@> <p> <@> <s> The integrated services digital network, or ISDN, is an international standard used to transmit voice, data, graphics and video images over telephone <head>lines</head> . </s>
</context>

- Test data also consists of instances containing the target word
  - Above is an example instance of the target word line
- But it does not contain the answer – that is provided in the key file
  - senseid="phone“ : is not provided
- Which of course were annotated by the same annotators

Key
For Test

<answer instance="line-n.w8_059:8174:" senseid="phone"/>

Represent an ambiguous word

```
<lexelt item="line-n">
<instance id="line-n.w9_10:6830:">
<answer instance="line-n.w9_10:6830:" senseid="phone"/>
<context>
 <s> In contrast, the California economy is booming, with 4.5% access <head>line</head>
growth in the past year. </s>
</context>
</instance>
```

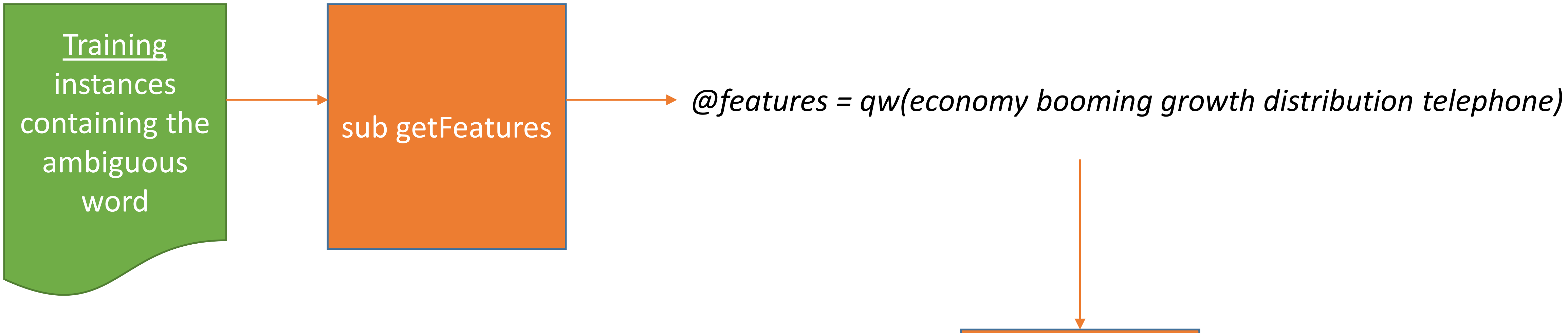
Specifically: how do we represent line in the instance “line-n.w9_10:6830:”

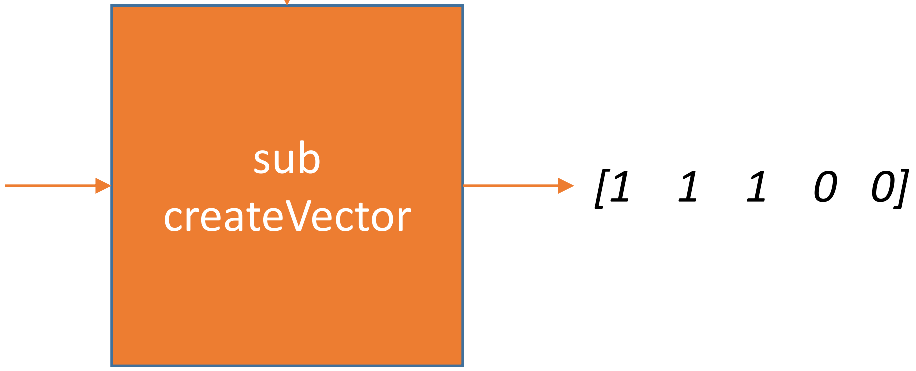
Feature vector

- Feature vector
  - Consists of numeric or nominal values that encode linguistic information

- Example features:
  - Bag-of-words
    - Word surrounding the target word
  - POS
    - Part of speech of the target word
  - Collocation features
    - The information about the words located to the left or right of the target word
  - N-grams
    - Extension of bag-of-words (which is just unigrams)

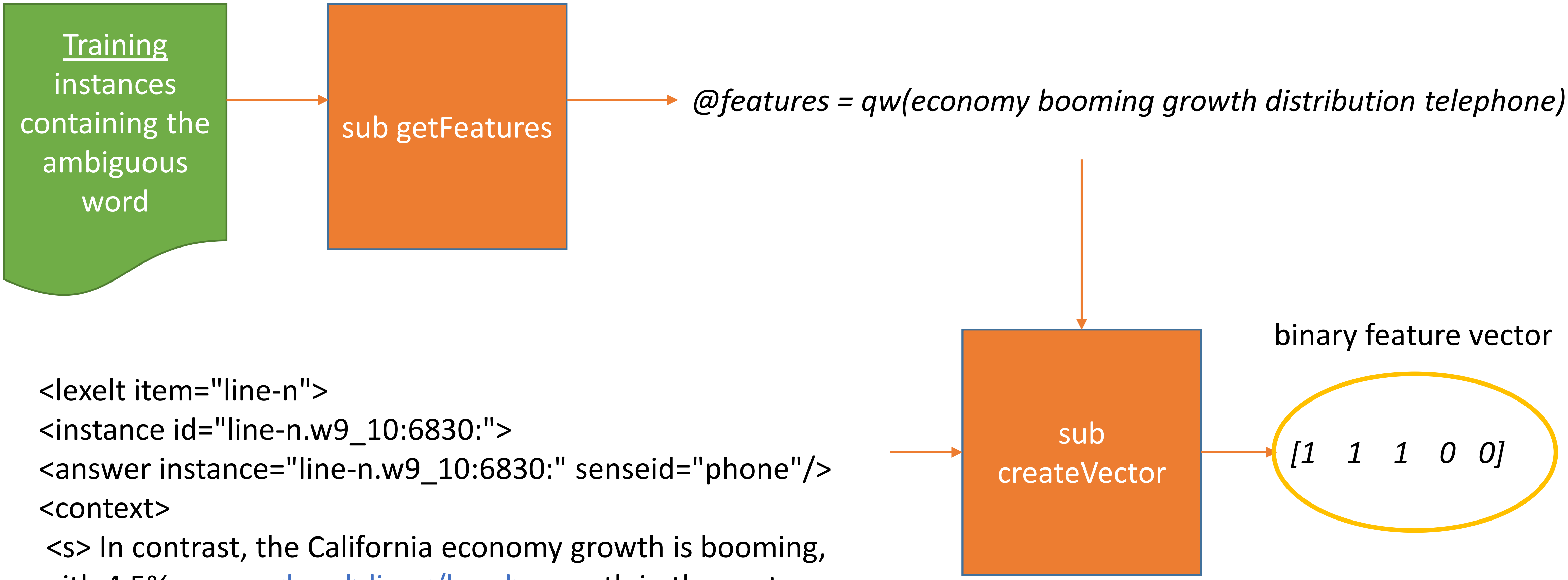
Bag-of-words (aka unigrams)





<lexelt item="line-n">
<instance id="line-n.w9_10:6830:">
<answer instance="line-n.w9_10:6830:" senseid="phone"/>
<context>
<s> In contrast, the California economy growth is booming, with 4.5% access <head>line</head> growth in the past year.</s>
</context>
</instance>

Bag-of-words (aka unigrams)




```
<lexelt item="line-n">
<instance id="line-n.w9_10:6830:">
<answer instance="line-n.w9_10:6830:" senseid="phone"/>
<context>
 <s> In contrast, the California economy growth is booming,
with 4.5% access <head>line</head> growth in the past
year.</s>
</context>
</instance>
```

Bag-of-words (aka unigrams)

<u>Training</u> instances containing the ambiguous word

sub getFeatures

@features = qw(economy booming growth distribution telephone)

frequency feature vector

sub createVector

[1 1 2 0 0]

<lexelt item="line-n">
<instance id="line-n.w9_10:6830:">
<answer instance="line-n.w9_10:6830:" senseid="phone"/>
<context>
<s> In contrast, the California economy growth is booming, with 4.5% access <head>line</head> growth in the past year.</s>
</context>
</instance>

Naïve Bayes

- Common machine learning algorithm
- Premise:
    - choose the best sense $\hat{s}$ given feature vector $\vec{f}$

- You see where this is going correct?

$$\hat{s} = argmax_{s \in S} P(s|\vec{f})$$

Difficult to calculate

$$\hat{s} = argmax_{s \in S} P(s|\vec{f})$$

So what do we do?

$$\hat{s} = \ argmax_{s \in S} P(s|\vec{f})$$

?

$$\hat{s} = \ argmax_{s \in S} \frac{P(\vec{f}|s)P(s)}{P(\vec{f})}$$

$$\hat{s} = argmax_{s \in S} P(s|\vec{f})$$

Bayes Rule

$$\hat{s} = argmax_{s \in S} \frac{P(\vec{f}|s)P(s)}{P(\vec{f})}$$

$$\hat{s} = argmax_{s \in S} P(s|\vec{f})$$

Bayes Rule

$$\hat{s} = argmax_{s \in S} \frac{P(\vec{f}|s)P(s)}{P(\vec{f})}$$

?

$$\hat{s} = argmax_{s \in S} P(\vec{f}|s)P(s)$$

$$\hat{s} = argmax_{s \in S} P(s|\vec{f})$$

Bayes Rule

$$\hat{s} = argmax_{s \in S} \frac{P(\vec{f}|s)P(s)}{P(\vec{f})}$$

Denominator same

$$\hat{s} = argmax_{s \in S} P(\vec{f}|s)P(s)$$

Naïve Assumption:

The features are conditionally independent given the word sense

$$\hat{s} = argmax_{s \in S} P(s) \prod_{j=1}^{n} P(f_j|s)$$

$$\hat{s} = argmax_{s \in S} P(s) \prod_{j=1}^{n} P(f_j | s)$$

$P(s_i)$ is the maximum likelihood estimate of how likely is these word to refer to the sense overall instances of the word

In other words,

How likely is *bank* referring to a *financial institution* over all instances of *bank*

$$\hat{s} = \ argmax_{s \in S} \, P(s) \prod_{j=1}^{n} P(f_j|s)$$

$$P(f_j|s) = \frac{count(f_j, s)}{count(s)}$$

So if we have a feature: [f_j = guitar]

- [f_j = guitar] occurred 3 times for sense $bass^1$
- sense $bass^1$ occurred 60 times in the training data

$P([guitar]|bass^1) = ?$

$$\hat{s} = \ argmax_{s \in S}\, P(s) \prod_{j=1}^{n} P(f_j|s)$$

$$P(f_j|s) = \frac{count(f_j, s)}{count(s)}$$

So if we have a feature: [f_j = guitar]

- [f_j = guitar] occurred 3 times for sense $bass^1$
- sense $bass^1$ occurred 60 times in the training data

$$P([guitar]|bass^1) = \frac{count(([guitar], bass^1)}{count(bass^1)} = \frac{3}{60} = 0.05$$

$$\hat{s} = argmax_{s \in S} P(s) \prod_{j=1}^{n} P(f_j|s)$$

- Probabilities are typically very low
  - Map everything to log-space and instead perform addition

$$Remember: \log(xy) = \log(x) + \log(y)$$

$$\hat{s} = argmax_{s \in S} P(s) \prod_{j=1}^{n} P(f_j|s)$$

- Probabilities are typically very low
  - Map everything to log-space and instead perform addition

$$\hat{s} = argmax_{s \in S} \log\big(P(s)\big) + \sum_{j=1}^{n} \log(P(f_j|s))$$

$$Remember: \log(xy) = \log(x) + \log(y)$$

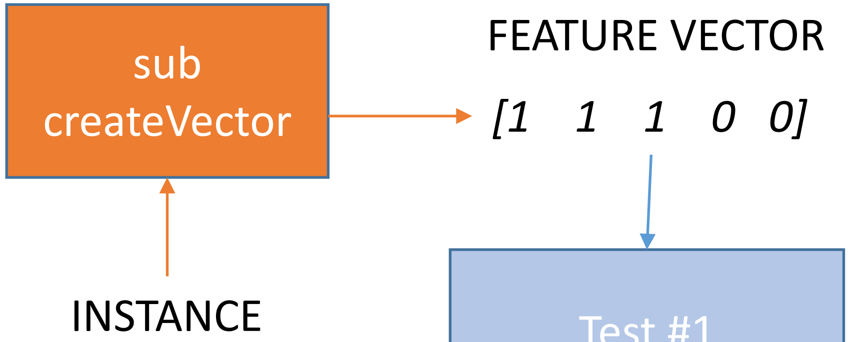
Decision Lists

- Equivalent to simple case statements (if-else statements)
    - A sequence of tests are applied to each target-word feature vector

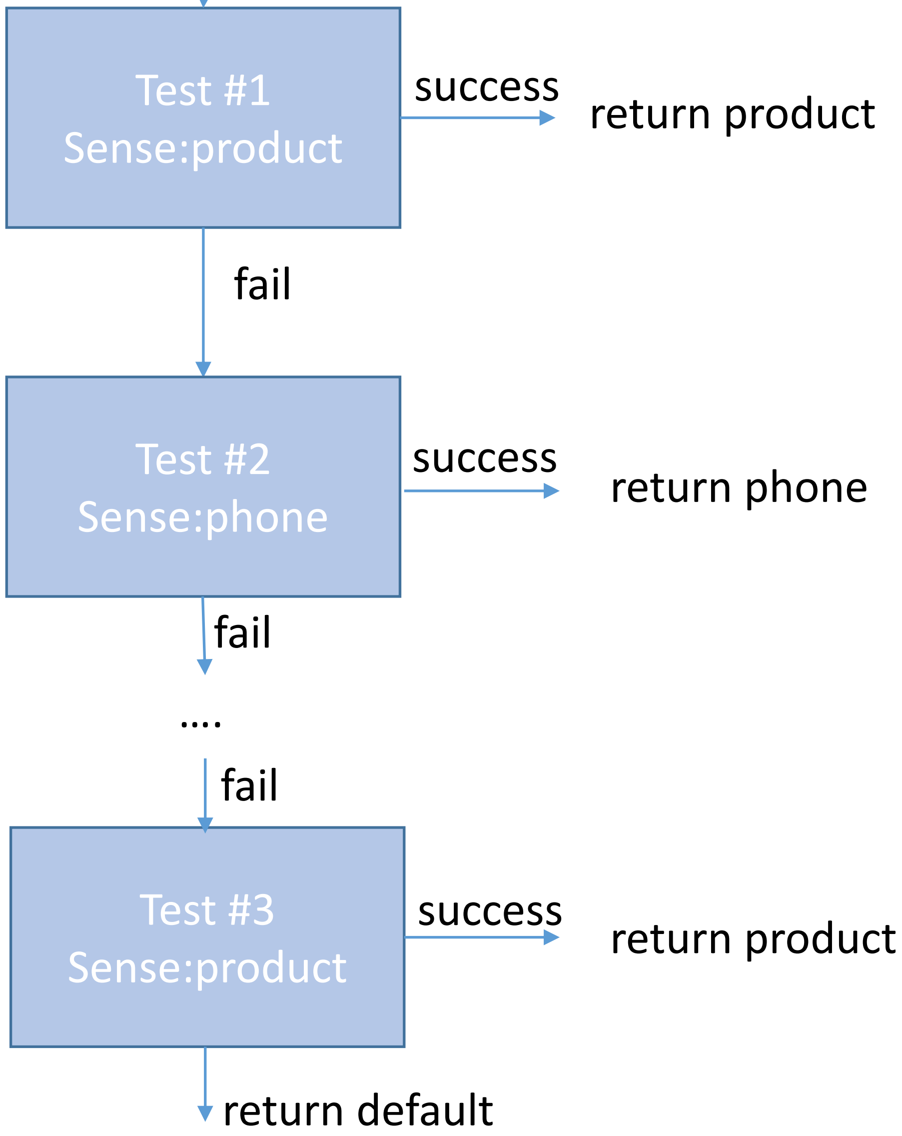
- Each test is indicative of a particular sense

    - If a test succeeds, then the sense associated with that test is returned
    - Otherwise, the next test in the sequence is applied

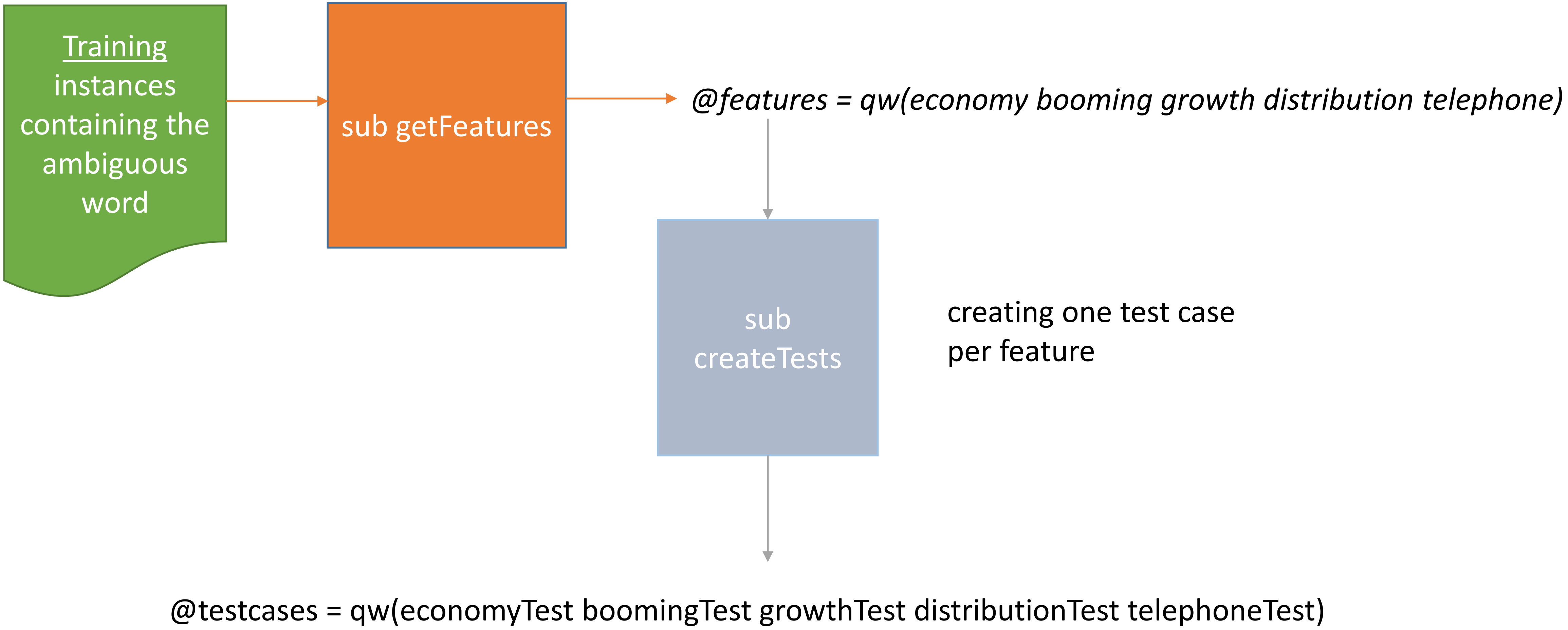
Generic example




```
<lexelt item="line-n">
<instance id="line-n.w9_10:6830:">
<answer instance="line-n.w9_10:6830:" senseid="phone"/>
<context>
 <s> In contrast, the California economy growth is booming,
with 4.5% access <head>line</head> growth in the past
year.</s>
</context>
</instance>
```



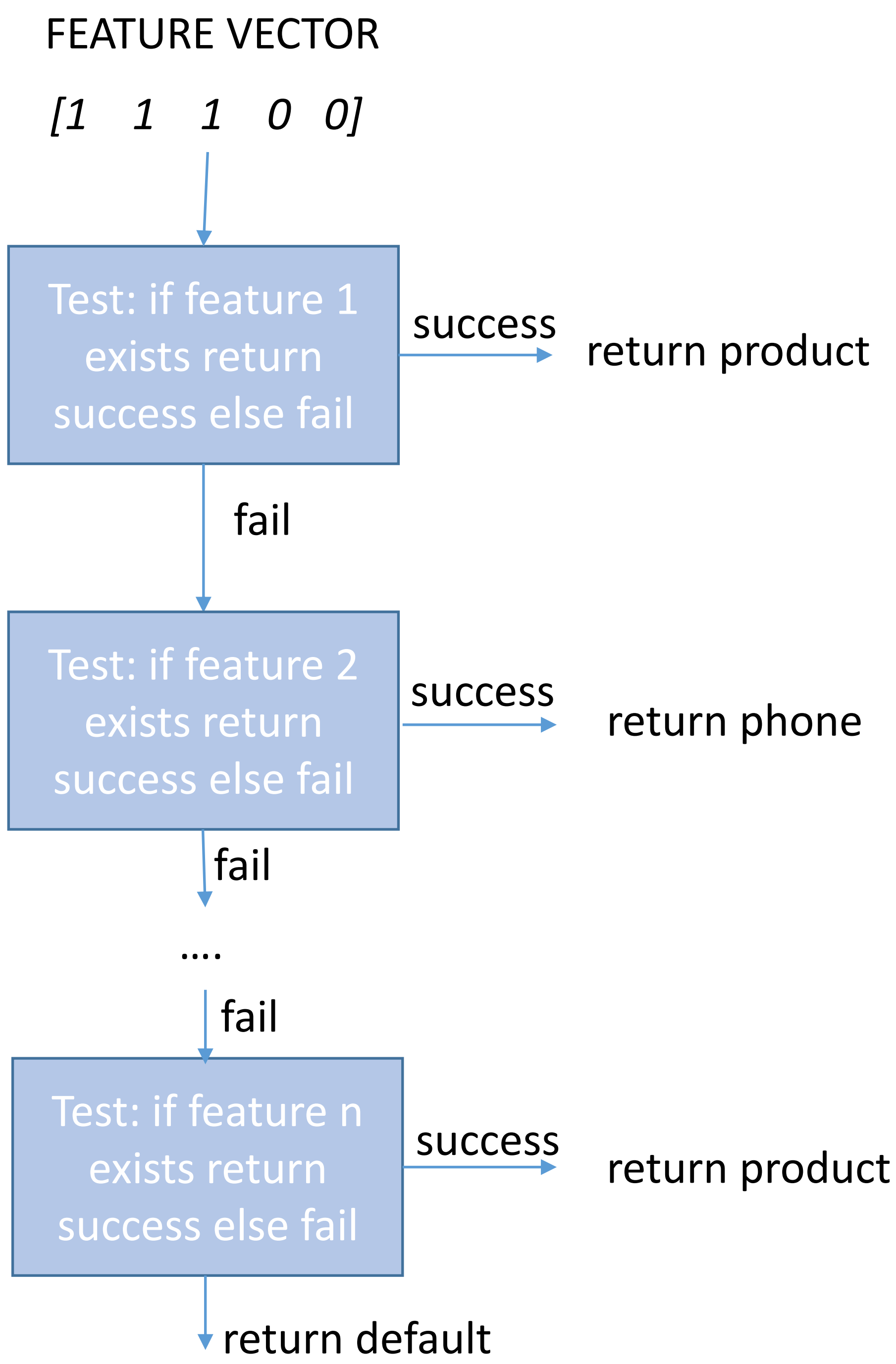
How to create the Tests



Individual Tests

The test are just simple if statements based on the occurrence of the feature.

if(feature exits in instance)
    return success
else
    return fail

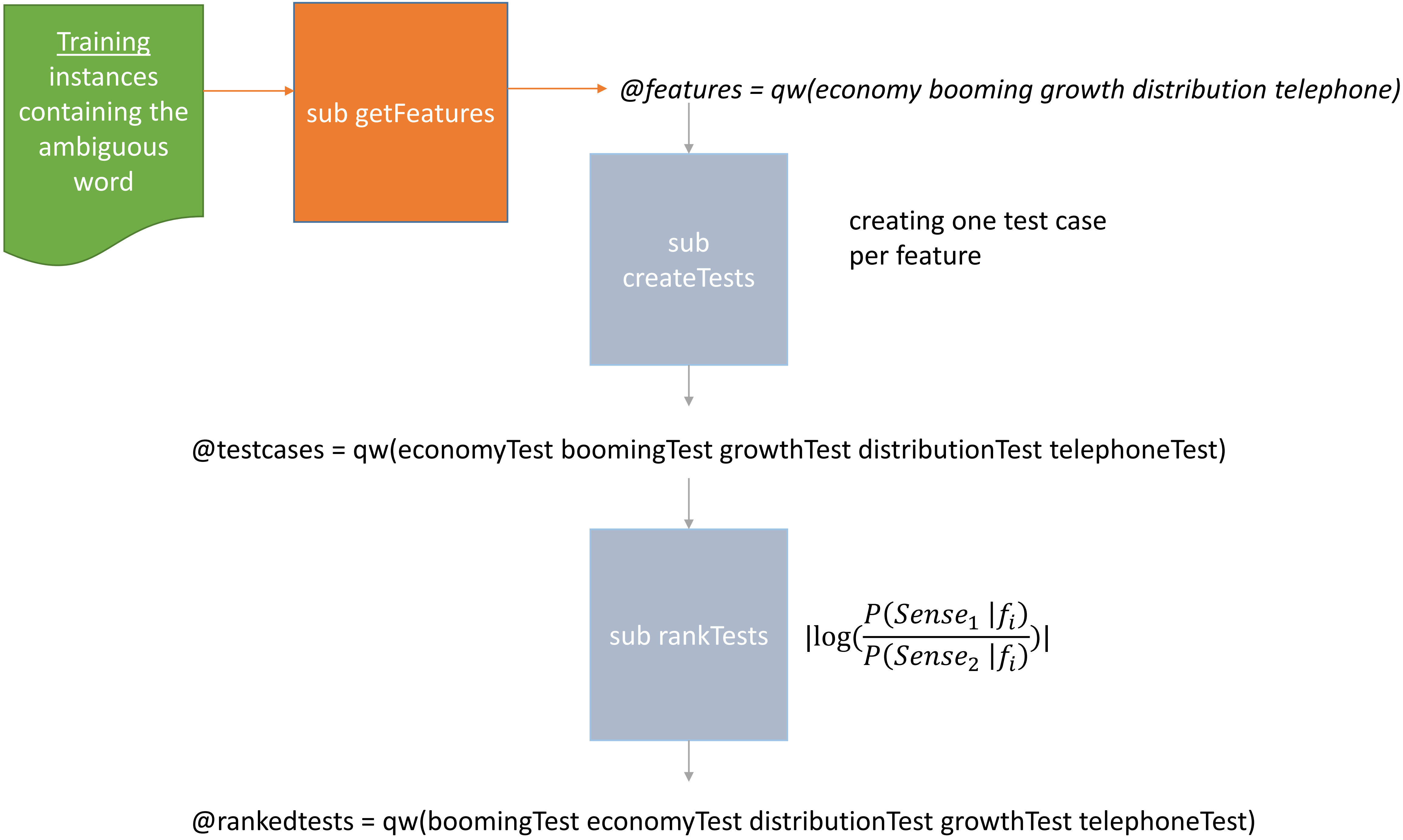


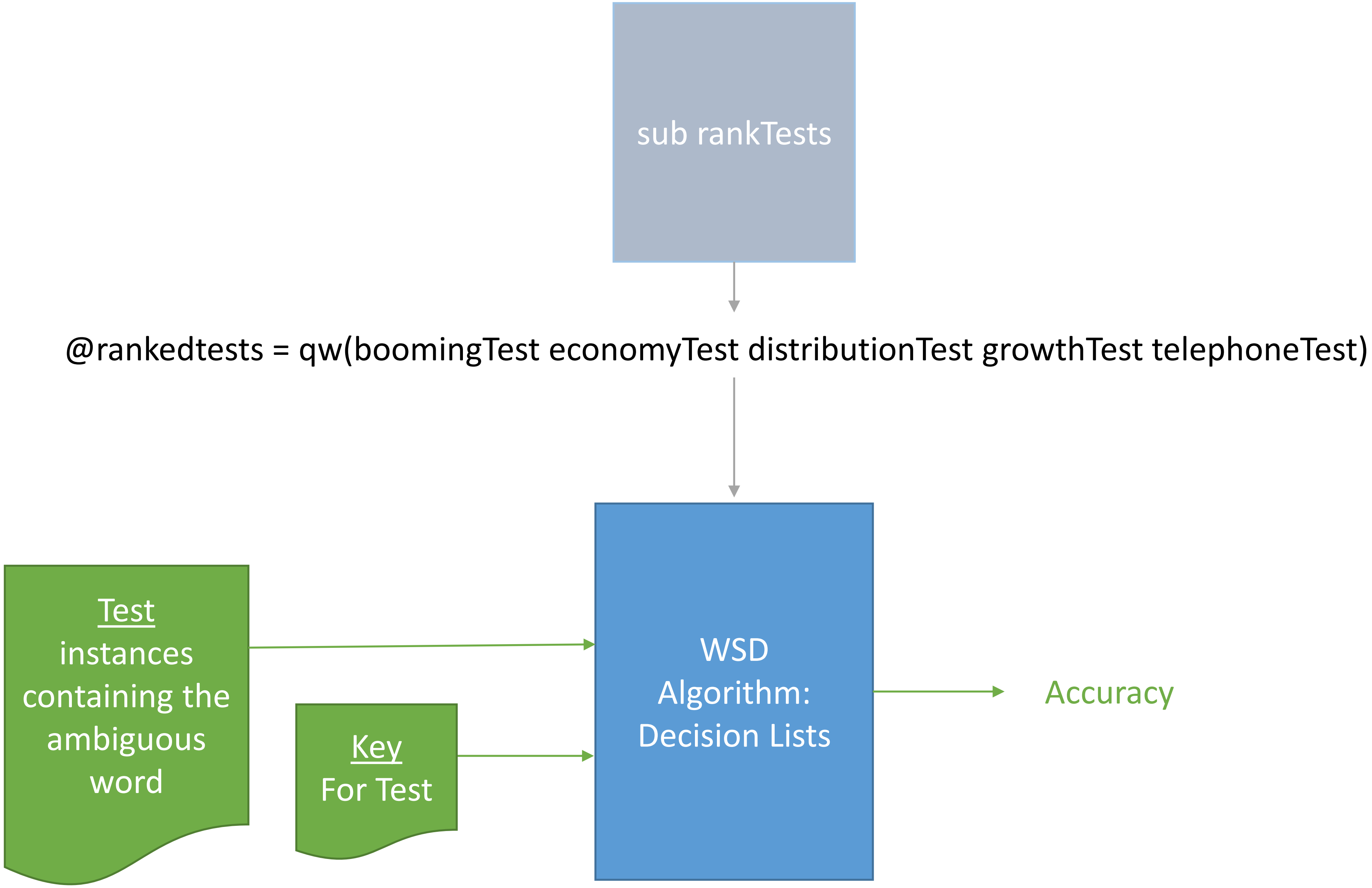
Ranking of the tests

- The individual tests are ranked by taking the ratio between the probabilities of the two senses
  - This tells us how discriminative a feature is (between senses)

$$|\log(\frac{P(Sense_1\ |f_i)}{P(Sense_2\ |f_i)})|$$

$$P(Sense_1\ |f_i) = \frac{Count(Sense_1\ f_i)}{Count(f_i)}$$


Training instances containing the ambiguous word
sub getFeatures
@features = qw(economy booming growth distribution telephone)
sub createTests
creating one test case per feature
@testcases = qw(economyTest boomingTest growthTest distributionTest telephoneTest)
sub rankTests
$|\log(\frac{P(Sense_1 \mid f_i)}{P(Sense_2 \mid f_i)})|$
@rankedtests = qw(boomingTest economyTest distributionTest growthTest telephoneTest)


sub rankTests
@rankedtests = qw(boomingTest economyTest distributionTest growthTest telephoneTest)
Test
instances containing the ambiguous word
Key
For Test
WSD Algorithm: Decision Lists
Accuracy

$$|\log(\frac{P(Sense_1\,|f_i)}{P(Sense_2\,|f_i)})|$$

$$P(Sense_1\,|f_i) = \frac{Count(Sense_1\ f_i)}{Count(f_i)}$$

bat^1 = flying mammal

bat^2 = stick one hits a baseball with

@features = qw(fly popcorn vampire)

freq{bat^1}{fly} = 5

freq{bat^1}{$popcorn$} = 1

freq{bat^1}{$vampire$} = 8

freq{bat^2}{fly} = 4

freq{bat^2}{$popcorn$} = 6

freq{bat^2}{$vampire$} = 1

freq{fly} = 12

freq{$popcorn$} = 8

freq{$vampire$} = 15

How do we evaluate the WSD algorithms.

Intrinsic Evaluation

- Accuracy:
  - The percentage of words tagged identically with the hand-labeled sense tags in the test set

Test instances containing the ambiguous word

<instance id="line-n.w8_059:8174:">
<context>
<s> The integrated services digital network, or ISDN, is an international standard used to transmit voice, data, graphics and video images over telephone <head>lines</head> . </s>
</context>

WSD Algorithm

assigned sense

check if they are the same

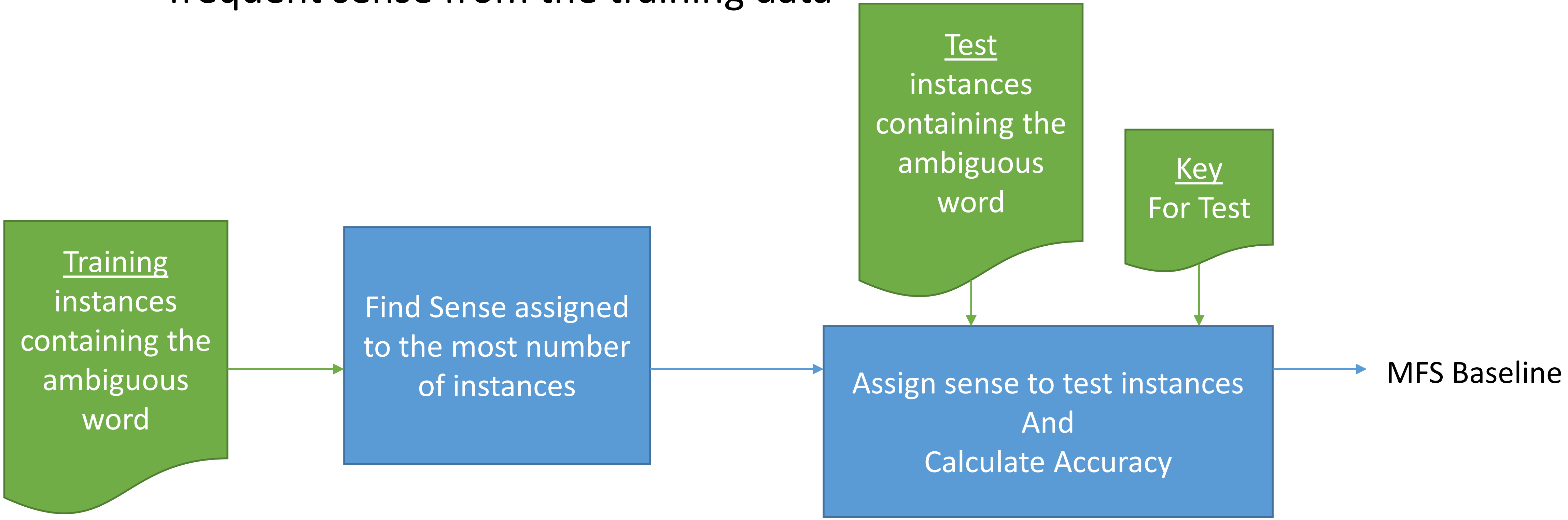
Key For Test

<answer instance="line-n.w8_059:8174:" senseid="phone"/>

MFS baseline

MFS Baseline	Your algorithm
Accuracy (%)	Accuracy (%)

- Most frequent sense
  - Assign each instance in the test data the most frequent sense from the training data



Programming assignment 4

Yarowsky’s Decision Lists for Lexical Ambiguity

- Link : https://arxiv.org/pdf/cmp-lg/9406034.pdf

DECISION LISTS FOR LEXICAL AMBIGUITY RESOLUTION:
Application to Accent Restoration in Spanish and French

David Yarowsky*
Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104
yarowsky@unagi.cis.upenn.edu

Features

- Word immediately to the right (+1 W)
- Word immediately to the left (-1 W)
- Word found in $\pm k$ word window[5] ($\pm$k W)
- Pair of words at offsets -2 and -1
- Pair of words at offsets -1 and +1
- Pair of words at offsets +1 and +2

Position	Collocation	côte	côté
-1 w	du *cote*	0	536
	la *cote*	766	1
	un *cote*	0	216
	notre *cote*	10	70
+1 w	*cote* ouest	288	1
	cote est	174	3
	cote du	55	156
+1w,+2w	*cote* du gouvernement	0	62
-2w,-1w	cote a *cote*	23	0
$\pm$k w	poisson (in $\pm k$ words)	20	0
$\pm$k w	ports (in $\pm k$ words)	22	0
$\pm$k w	opposition (in $\pm k$ words)	0	39

Features

- Example features we talked about:
  - Bag-of-words
    - Word surrounding the target word
  - POS
    - Part of speech of the target word
  - Collocation features
    - The information about the words located to the left or right of the target word
  - N-grams
    - Extension of bag-of-words (which is just unigrams)

- Yarowsky’s features
  - Word immediately to the right (+1 w)
  - Word immediately to the left (-1 w)
  - Word found in +=k word window
  - Pair of words at offsets -2 and -1
  - Pair of words at offsets -1 and +1
  - Pair of words at offsets +1 and +2

Questions?