Feature engineering in NLP

Mariana Romanyshyn
Computational Linguist at *Grammarly*

Contents

1. Presentation of text in NLP

Presentation of text in NLP

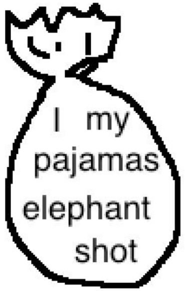
Trump beat Clinton in the election.

= or ≠

Clinton beat Trump in the election.

Presentation of text in NLP

- Bag of words

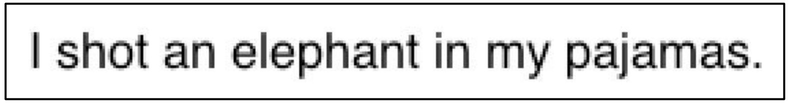


Presentation of text in NLP

- Bag of words

I
my
pajamas
elephant
shot

- Sequence



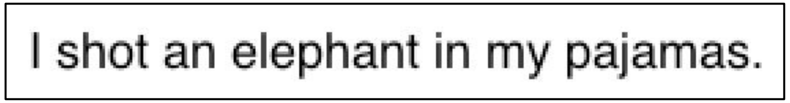
I shot an elephant in my pajamas.

Presentation of text in NLP

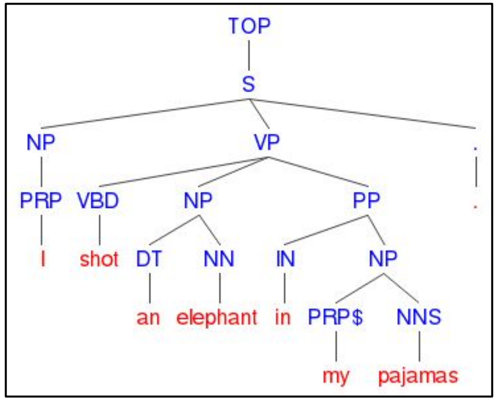
- Bag of words

I my
pajamas
elephant
shot

- Sequence


I shot an elephant in my pajamas.

- Tree

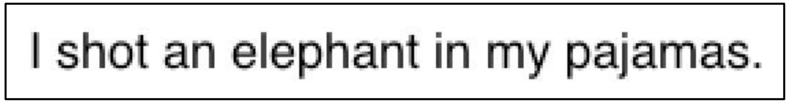


Presentation of text in NLP

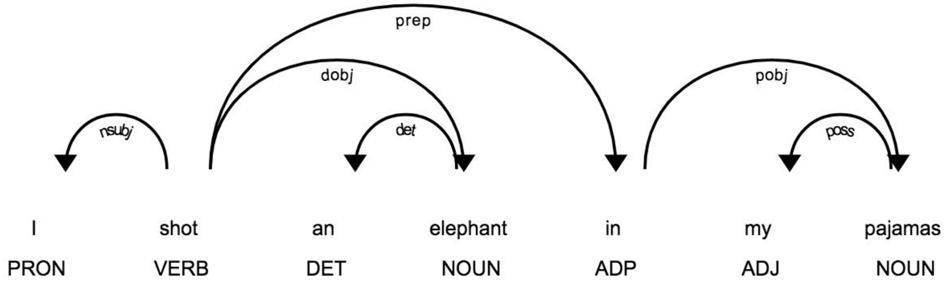
- Bag of words

I my
pajamas
elephant
shot

- Sequence


I shot an elephant in my pajamas.

- Tree



Presentation of text in NLP

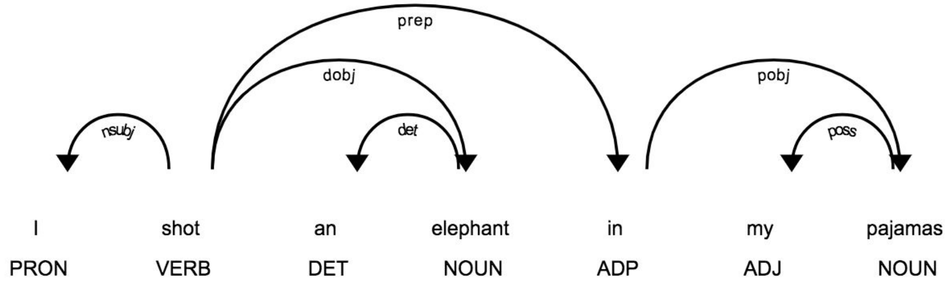
- Bag of words

I
my
pajamas
elephant
shot

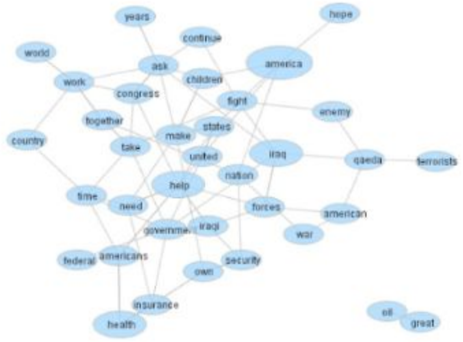
- Sequence

I shot an elephant in my pajamas.

- Tree



- Graph

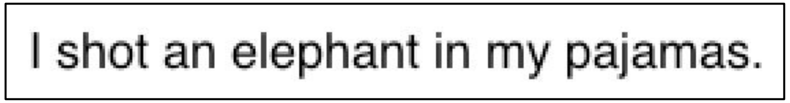


Presentation of text in NLP

- Bag of words

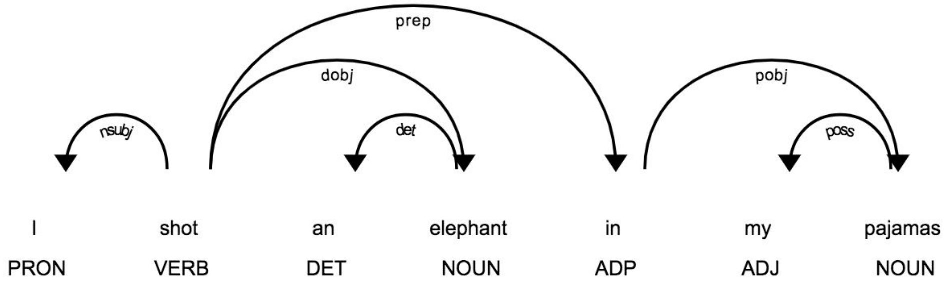
I my
pajamas
elephant
shot

- Sequence

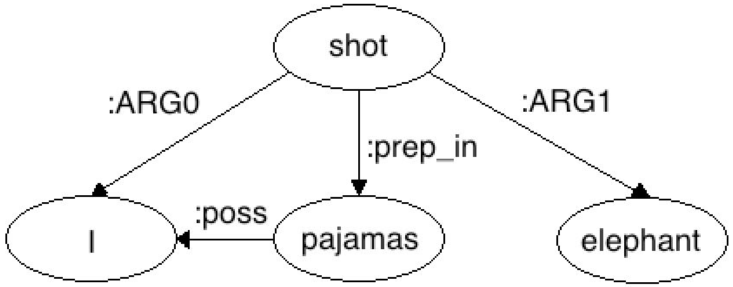


I shot an elephant in my pajamas.

- Tree



- Graph



Think of different NLP tasks

How would you view the text if you need...

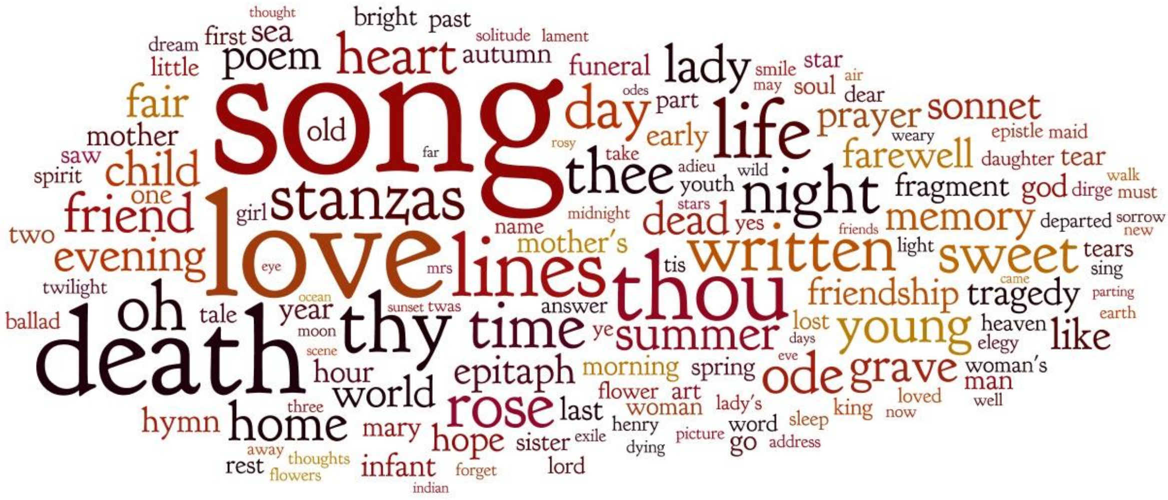
- classification of news articles by topic?
- named-entity recognition?
- sentiment assignment to objects in the text?
- abstractive text summarization?

2. Feature engineering

The Word

A word is its...

1. form
2. function
3. meaning



The Form

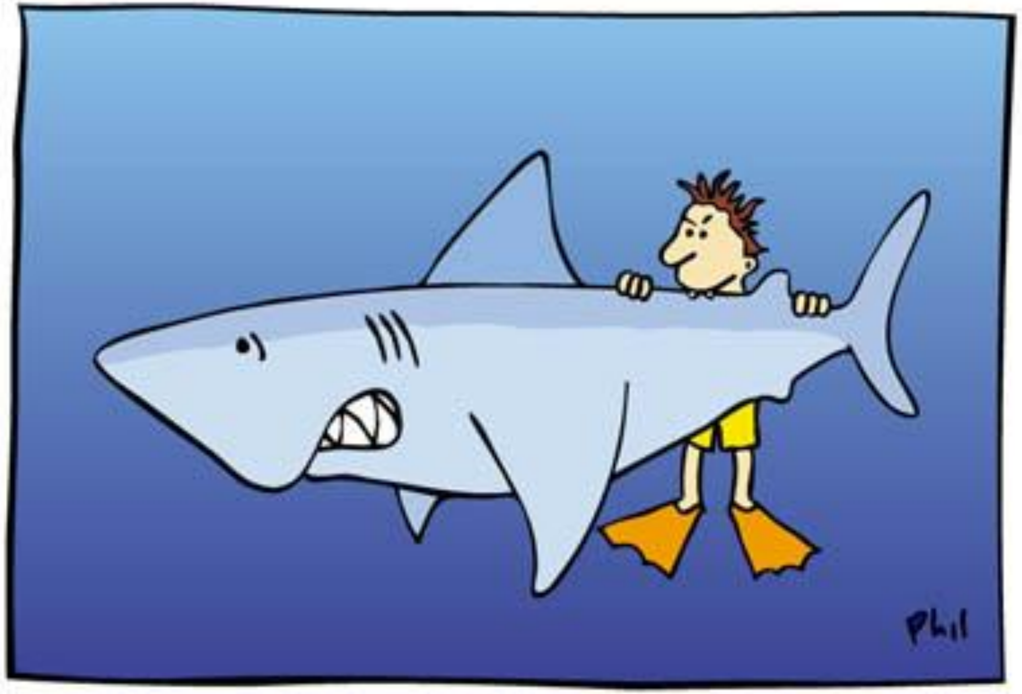
The form — how the word is written.

Form Features

- Capitalization, hyphenation, apostrophes
- Lemma or stem
- Number of stems
- Number and types of affixes
- Length of the word/lemma/stem
- Number of tokens, position of a token
- Number of syllables in a word
- Ratio of vowels vs. consonants
- Voiced vs. voiceless consonants
- All possible frequencies

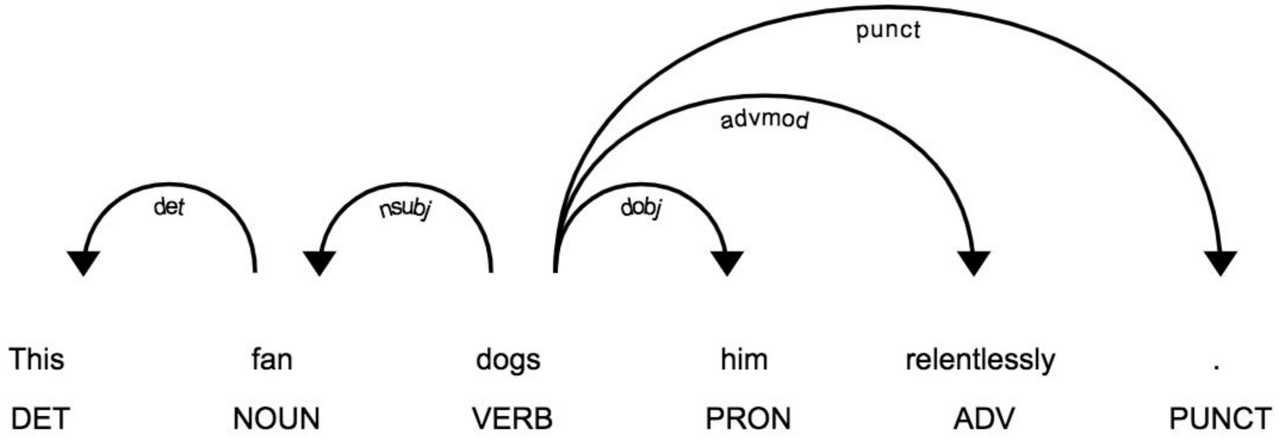
Form Features

a ***man-eating*** *shark* vs. a ***man eating*** *shark*



The Function

The function — what the word does and how it interacts with other words in the text.



Function Features

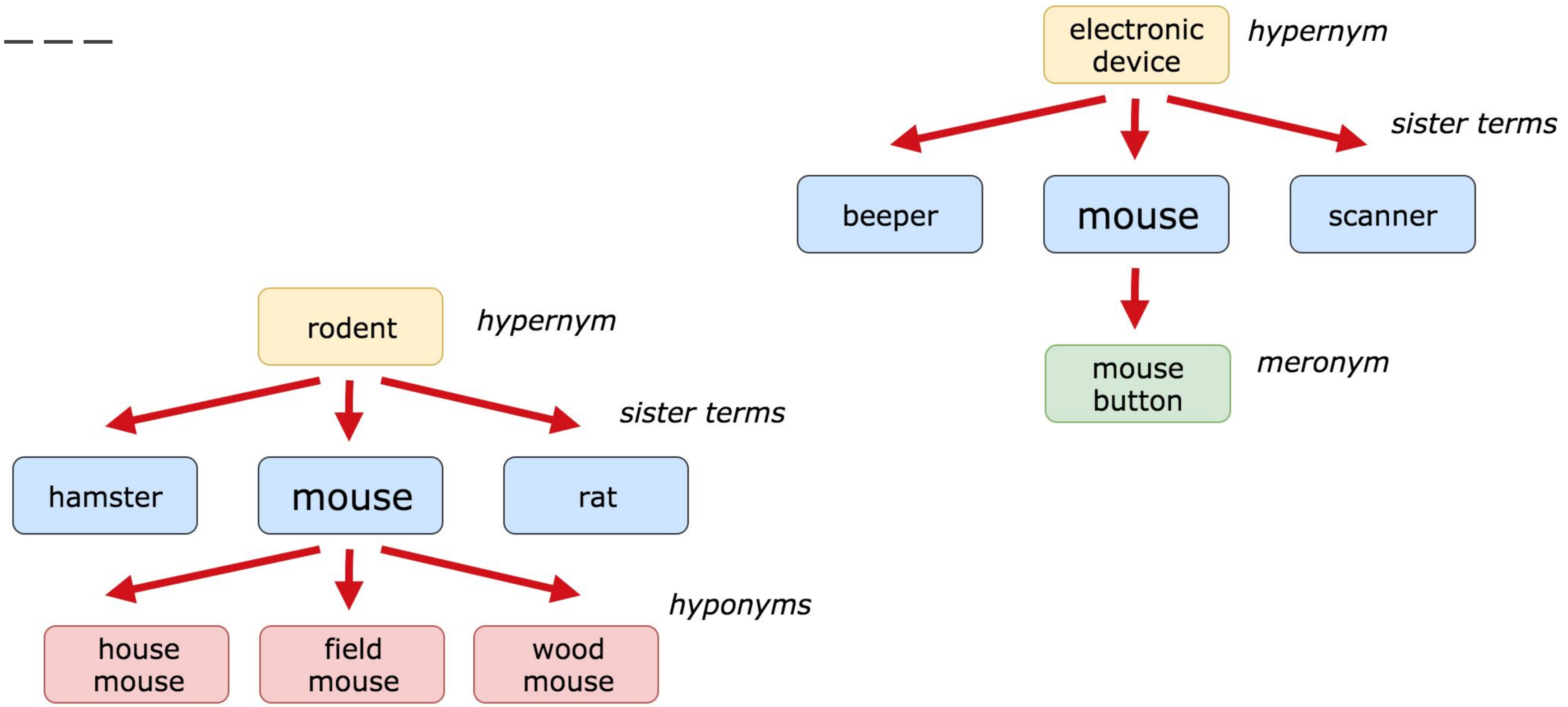
- Part of speech, part-of-speech tag
- Morphological properties:
  - *gender, animacy, number, person, case*
  - *aspect, voice, tense, degree of comparison*
- Constituents
  - *parents, children*
- Direct and indirect dependencies
  - *parents, children, type of relation*
- Depth of the syntactic tree
- Statistics: *POS+word, POS ngrams, syntactic ngrams*

The Meaning

The meaning — the particular sense the word obtains in the context and how it correlates with other words and senses.

DADDY, I WANT A PONY!
milk
SPOILED MILK

Lexical Semantics in WordNet



Entailment

Entailment examples:

- *Mom slices a cucumber. => A woman cuts a vegetable.*
- *A jogger was spotted. => Someone was jogging.*
- *Her husband was snoring. => Her husband was sleeping.*
- *The king of France is bald. => There exists a king of France.*

Meaning Features

- Word sense
- Number of senses
- Shortest path to another word sense
- Similarity to another sense
- Synonyms/antonyms, hyponyms/hypernyms, meronyms/holonyms
- Entailment
- Semantic role

3. Feature encoding

Encode as a bag of words: boolean

```
The pound extended losses against both the dollar and the euro .
```

[1,	1,	0,	0,	1,	0,	...]
the	*dollar*	*hello*	*pirate*	*losses*	*run*	

Encode as a bag of words: count

`The pound extended losses against both the dollar and the euro .`

[3,	*1,*	*0,*	*0,*	*1,*	*0,*	*...]*
the	*dollar*	*hello*	*pirate*	*losses*	*run*	

Encode as a bag of words: tf-idf

```
The pound extended losses against both the dollar and the euro .
```

[	*0.1,*	*0.7,*	*0,*	*0,*	*0.4,*	*0,*	*…]*
	the	*dollar*	*hello*	*pirate*	*losses*	*run*	

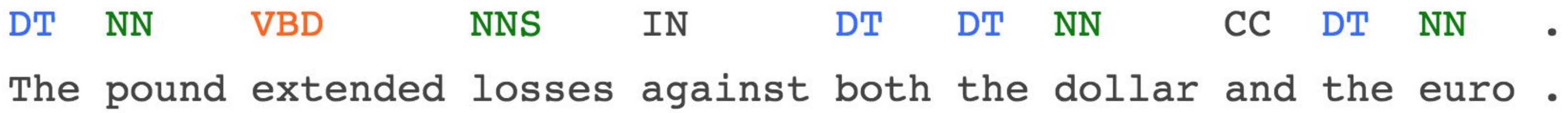
Encode as a bag of features

DT	NN	VBD	NNS	IN	DT	DT	NN	CC	DT	NN	.
The	pound	extended	losses	against	both	the	dollar	and	the	euro	.

“losses”/NNS:

{“word-2”: “pound”,
 “word-1”: “extended”,
 “word+1”: “against”,
 “word+2”: “both”,
“tag-2”: “NN”,
“tag-1”: “VBD”,
“tag+1”: “IN”,
“tag+2”: “DT”}

Encode as a bag of features



```
DT  NN    VBD      NNS    IN      DT   DT  NN     CC  DT  NN   .
The pound extended losses against both the dollar and the euro .
```

[1, 1, 0, …]

word-2=pound word-1=extended word+1=hello

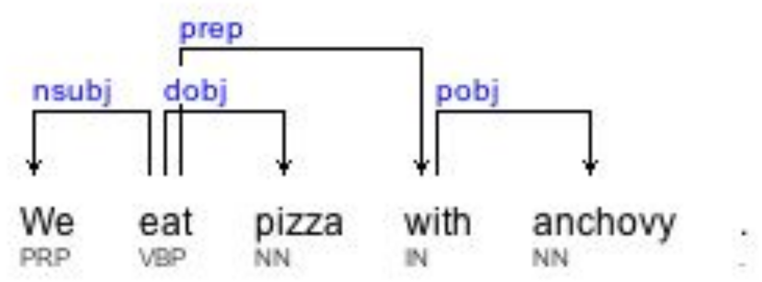
Encode as a bag of features

```
DT  NN    VBD      NNS    IN      DT   DT  NN     CC  DT  NN   .
The pound extended losses against both the dollar and the euro .
```

“losses”/NNS:

{“*left-bigram*”: “*pound extended*”,
“*right-bigram*”: “*against both*”,
“*context*”: “*extended losses against both*”}

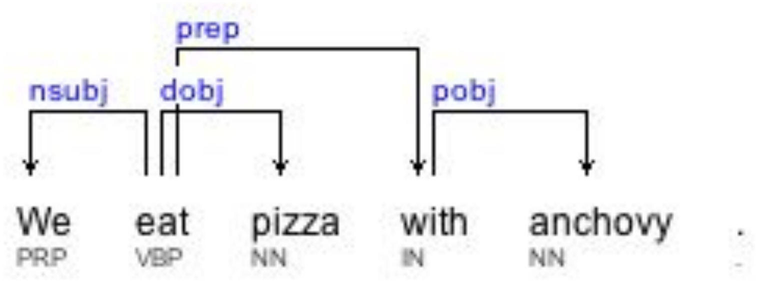
Encode as a bag of features



“eat”/VBP:

[1,	*0,*	*0,*	*1,*	*0,*	*1,*	*0,*	*0,*	*…]*
nsubj	*acl*	*relcl*	*dobj*	*pobj*	*prep*	*punct*	*xcomp*	

Encode as a bag of features



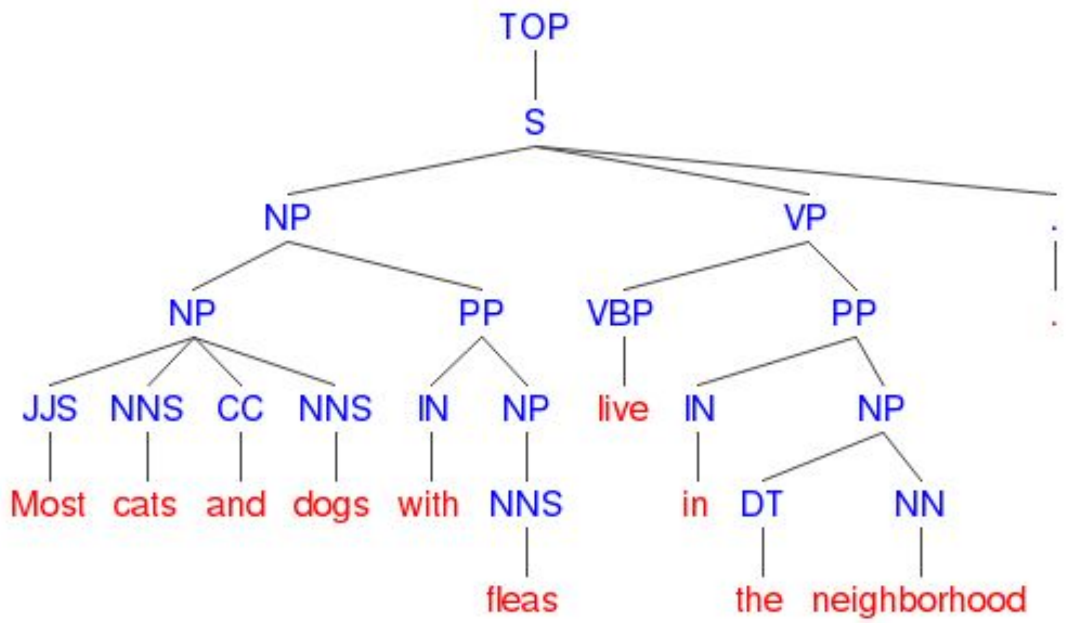
"eat"/VBP:

- *nsubj_We, dobj_pizza, prep_with*
- *nsubj_PRP, dobj_NN, prep_IN*
- *nsubj_We, dobj_pizza, prep_with_pobj_anchovy*

Encode as a bag of features

“fleas”/NNS:

{“label”: “NP”,
“anc-left”: ”PP”,
“anc-right”: ”S”,
“span-width”:1}



Example

https://bit.ly/2KNsiLJ

or

https://github.com/mariana-scorp/esscass-2019-nlp/blob/master/2-features/feature-encoding.ipynb

4. A few words about ngrams (optional)

What are ngrams

Ngram - a contiguous sequence of ***n*** items from a given text.

What are ngrams

Ngram - a contiguous sequence of ***n*** items from a given text.

So, if ***n* = 3**:

$\underbrace{\textit{<S>} \quad \textit{Why} \quad \textit{did}}$ *n't* *you* *listen* *to* *me* *?* *</S>*

What are ngrams

Ngram - a contiguous sequence of ***n*** items from a given text.

So, if ***n* = 3**:

$$\textit{<S>}\quad \underbrace{\textit{Why}\quad \textit{did}\quad \textit{n't}}\quad \textit{you}\quad \textit{listen}\quad \textit{to}\quad \textit{me}\quad \textit{?}\quad \textit{</S>}$$

What are ngrams

Ngram - a contiguous sequence of ***n*** items from a given text.

So, if ***n* = 3**:

<S> Why $\underbrace{\textit{did n't you}}$ *listen to me ? </S>*

What are ngrams

Ngram - a contiguous sequence of ***n*** items from a given text.

So, if ***n* = 3**:

$$\textit{<S>} \quad \textit{Why} \quad \textit{did} \quad \underbrace{\textit{n't} \quad \textit{you} \quad \textit{listen}} \quad \textit{to} \quad \textit{me} \quad \textit{?} \quad \textit{</S>}$$

What are ngrams

Ngram - a contiguous sequence of ***n*** items from a given text.

So, if ***n* = 3**:

$$\textit{<S>} \quad \textit{Why} \quad \textit{did} \quad \textit{n't} \quad \underbrace{\textit{you} \quad \textit{listen} \quad \textit{to}} \quad \textit{me} \quad \textit{?} \quad \textit{</S>}$$

What are ngrams

Ngram - a contiguous sequence of ***n*** items from a given text.

So, if ***n* = 3**:

<S> *Why did n’t you* $\underbrace{\textit{listen to me}}$? </S>

What are ngrams

Ngram - a contiguous sequence of ***n*** items from a given text.

So, if ***n* = 3**:

$$\textit{<S>} \quad \textit{Why} \quad \textit{did} \quad \textit{n't} \quad \textit{you} \quad \textit{listen} \quad \underbrace{\textit{to} \quad \textit{me} \quad \textit{?}} \quad \textit{</S>}$$

What are ngrams

Ngram - a contiguous sequence of ***n*** items from a given text.

So, if ***n* = 3**:

$\textit{<S> Why did n't you listen to } \underbrace{\textit{me ? </S>}}$

Token ngrams

Usually 1 ≥ n ≥ 5.

<S>   Why   did   n’t   you   listen   to   me   ?   </S>

n = 1: *(<S>), (Why), (did), (n’t), (you), (listen), (to), (me), (?)…*
n = 2: *(<S> Why), (Why did), (did n’t), (n’t you), (you listen), (listen to)…*
n = 3: *(<S> Why did), (Why did n’t), (did n’t you), (you listen to)…*
…

Character Ngrams

<S>   Why   did   n’t   you   listen   to   me   ?   </S>

For words:

n = 3: *(<w> W h), (W h y), (h y </w>), (<w> d i), (d i d), (i d n), (d n ‘)…*

For sentences:

n = 3: *(W h y), (h y _), (y _ d), (_ d i), (d i d), (i d n), (d n ‘), (n ‘ t)…*

POS Ngrams

<S>	*Why*	*did*	*n't*	*you*	*listen*	*to*	*me*	?	</S>
<S>	WDT	VDB	RB	PRP	VB	TO	PRP	.	</S>

POS:

***n* = 3**: *(<S>, WDT, VBD), (WDT, VBD, RB), (VBD, RB, PRP), (RB, PRP, VB)…*

Token+POS:

***n* = 2**: *(<S>_<S>, Why_WDT), (Why_WDT, did_VBD), (did_VBD, n't_RB)…*

Token or POS:

***n* = 3**: *(<S>, WDT, did), (WDT, did, RB), (did, RB, PRP), (RB, PRP, listen)…*

Tree Ngrams

Head+dependency:

listen_nsubj

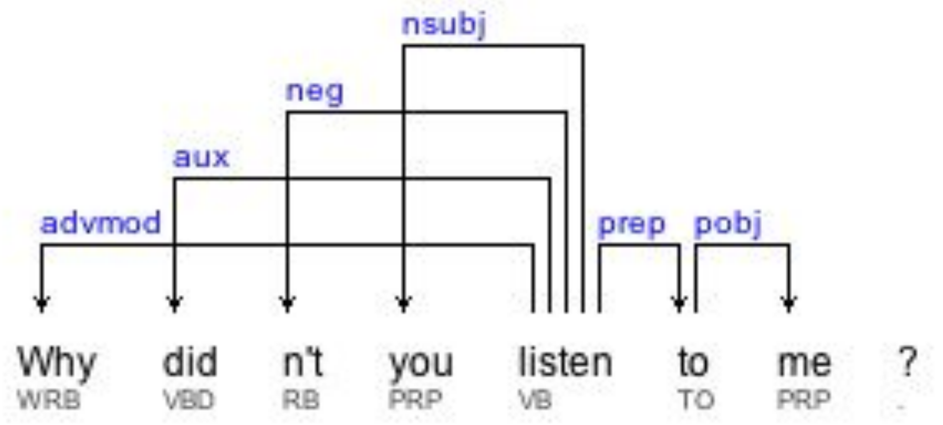
listen_nsubj_you

listen_prep_to_pobj_me

Head+POS+dependency:

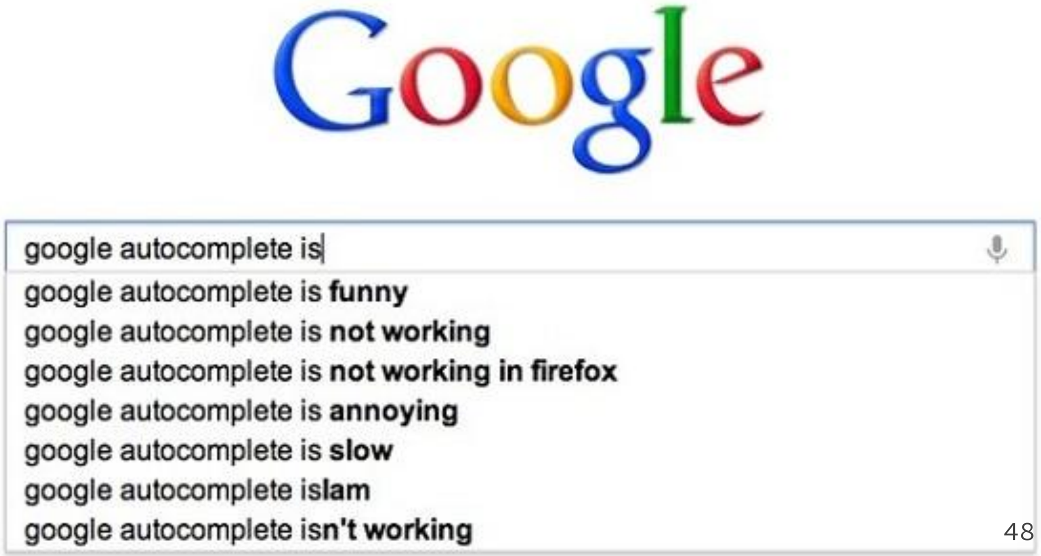
listen/VB_nsubj

listen/VB_nsubj_you/PRP



Ngrams usage

- Speech recognition
- Text generation
- Autocompletion


google autocomplete is
google autocomplete is **funny**
google autocomplete is **not working**
google autocomplete is **not working in firefox**
google autocomplete is **annoying**
google autocomplete is **slow**
google autocomplete is**lam**
google autocomplete is**n't working**

Ngrams usage

- Speech recognition
- Text generation
- Autocompletion
- Handwriting recognition
- Spelling correction
- (and GEC in general)

Borrowed from https://www.youtube.com/watch?v=l3pdGfKycz8

Ngrams as a feature

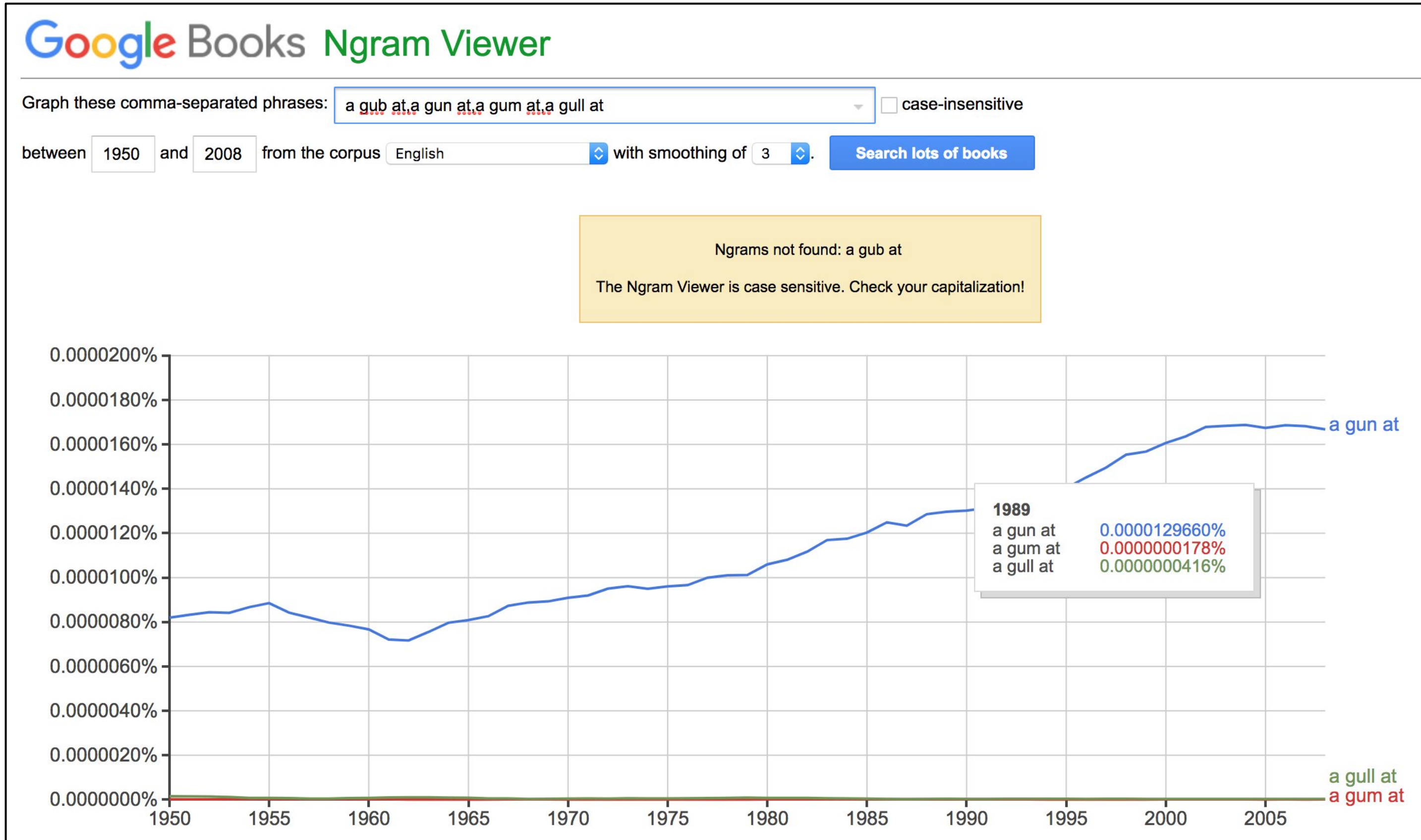
Frequency or probability:

a gub at

a gun at

a gum at

a gull at


Google Books Ngram Viewer

Graph these comma-separated phrases: a gub at,a gun at,a gum at,a gull at ☐ case-insensitive

between 1950 and 2008 from the corpus English with smoothing of 3. Search lots of books

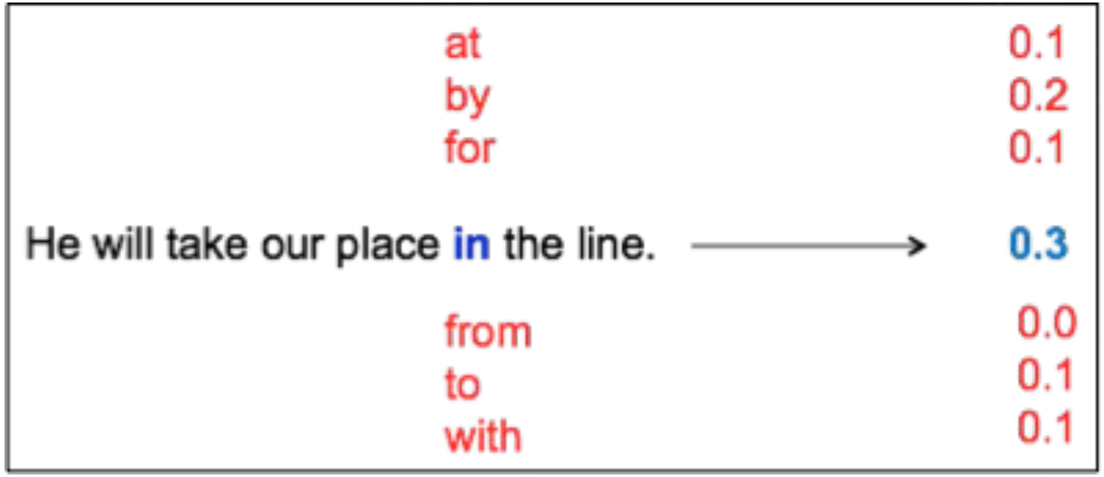
Ngrams not found: a gub at

The Ngram Viewer is case sensitive. Check your capitalization!

1989
a gun at 0.0000129660%
a gum at 0.0000000178%
a gull at 0.0000000416%

Ngrams as a feature

Frequency or probability



at	0.1
by	0.2
for	0.1
He will take our place **in** the line. $\longrightarrow$	0.3
from	0.0
to	0.1
with	0.1

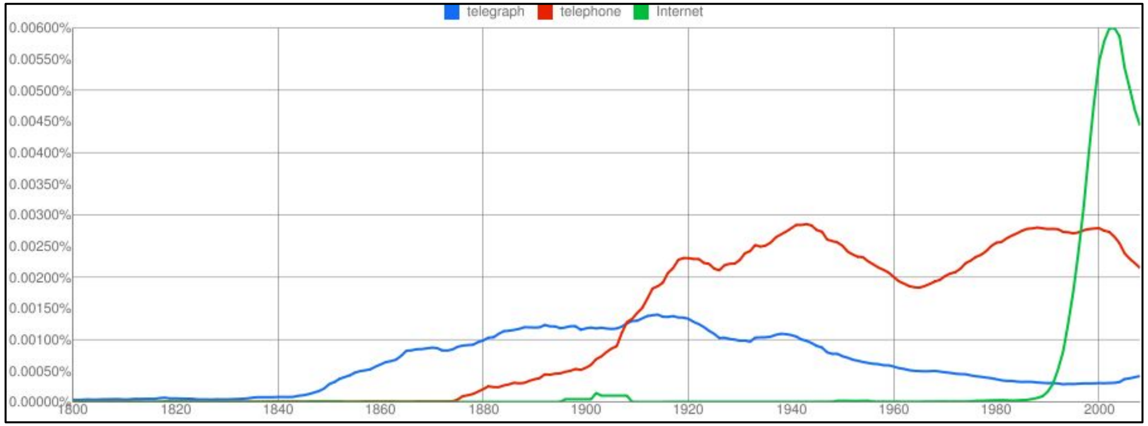
Ngrams as a feature

Conditional probability

$$P(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n)}{C(w_{n-1})}$$

Where to get ngrams

- 1 mln of 2/3/4/5-ngrams from COCA for free
- Google ngrams (and how to download)
- Google syntactic ngrams
- collect on your own



5. Logistic Regression (optional)

Logistic Regression

Logistic regression - a discriminative linear model used for binary classification.

- *like Perceptron*, it's linear
- *like Naive Bayes*, it extracts a set of weighted features, takes logs, and combines them linearly
- *unlike Naive Bayes*, it's discriminative

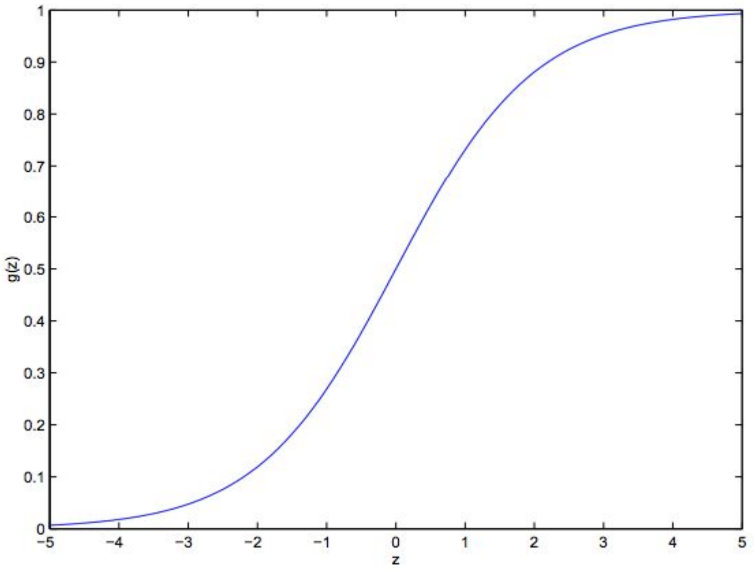
$$z = \left(\sum_{i=1}^{n} w_i x_i\right).$$

Logistic Regression

A [0; 1] function would be handy: y = 1 if p(y=1|x) > 0.5.

Sigmoid function:

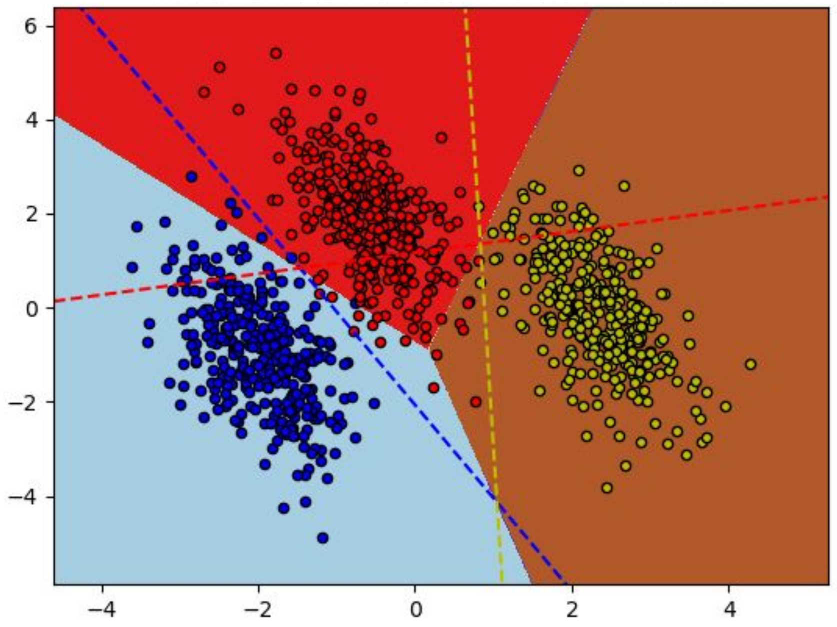
$$\begin{aligned} P(y=1) &= \sigma(w \cdot x + b) \\ &= \frac{1}{1+e^{-(w \cdot x + b)}} \end{aligned}$$



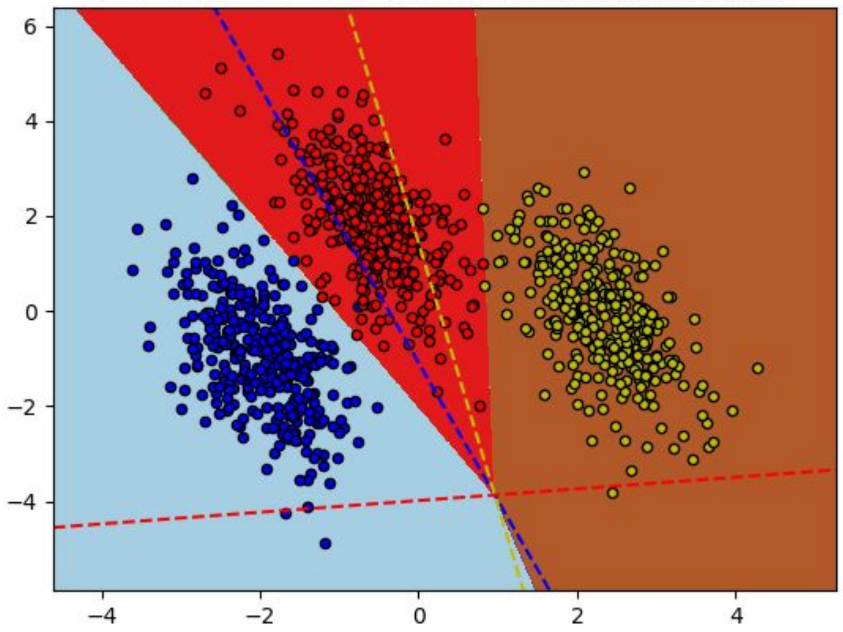
Borrowed from Andrew Ng: https://goo.gl/zGvb4W

Logistic Regression: multiclass

one vs. rest



multinomial (MaxEnt)



http://scikit-learn.org/stable/auto_examples/linear_model/plot_logistic_multinomial.html

Logistic Regression

For multinomial logistic regression, use softmax:

$$p(c|x) = \frac{\exp\left(\sum_{i=1}^{N} w_i f_i(c,x)\right)}{\sum_{c' \in C} \exp\left(\sum_{i=1}^{N} w_i f_i(c',x)\right)}$$

Logistic Regression: example

Welcome to St . Paul ‘s Cathedral !

[Is this period a sentence end?]

Logistic Regression: example

Welcome to St . *Paul* ‘s Cathedral !

[Is this period a sentence end?]

y: *{is-end, is-not-end}*
x: *{“word+1_is_cap”, “word-1=kittens”, “word-1=St”, “tag-1=PRP”, “tag+1=JJ”}*

Logistic Regression: example

Welcome to St . *Paul ‘s Cathedral !*

[Is this period a sentence end?]

y: *{is-end, is-not-end}*

x: *{“word+1_is_cap”, “word-1=kittens”, “word-1=St”, “tag-1=PRP”, “tag+1=JJ”}*

$\mathbf{x}_j$: *[1, 0, 1, 0, 0]*

Logistic Regression: example

Welcome to St . *Paul* ‘s Cathedral !

[Is this period a sentence end?]

y: *{is-end, is-not-end}*
x: *{“word+1_is_cap”, “word-1=kittens”, “word-1=St”, “tag-1=PRP”, “tag+1=JJ”}*

$\mathbf{x}_j$: *[1, 0, 1, 0, 0]*

$\mathbf{w}_{\textbf{is-end}}$: *[2.9, 2.5, -0.9, 0, 0]*
$\mathbf{w}_{\textbf{is-not-end}}$: *[0.5, -0.7, 2.9, 0, 0]*

Logistic Regression: example

Welcome to St . Paul ‘s Cathedral !

[Is this period a sentence end?]

y: *{is-end, is-not-end}*
x: {“word+1_is_cap”, “word-1=kittens”, “word-1=St”, *“tag-1=PRP”, “tag+1=JJ”}*

$\mathbf{x}_j$: *[1, 0, 1, 0, 0]*

$\mathbf{w}_{\textbf{is-end}}$: [2.9, 2.5, -0.9, 0, 0]
$\mathbf{w}_{\textbf{is-not-end}}$: [0.5, -0.7, 2.9, 0, 0]

$\mathbf{P(is\text{-}end|x_j)} = e^{2.9-0.9} / (e^{2.9-0.9} + e^{0.5+2.9}) = 0.2$
$\mathbf{P(is\text{-}not\text{-}end|x_j)} = e^{0.5+2.9} / (e^{2.9-0.9} + e^{0.5+2.9}) = 0.8$

Logistic Regression: weights

Learn weights:

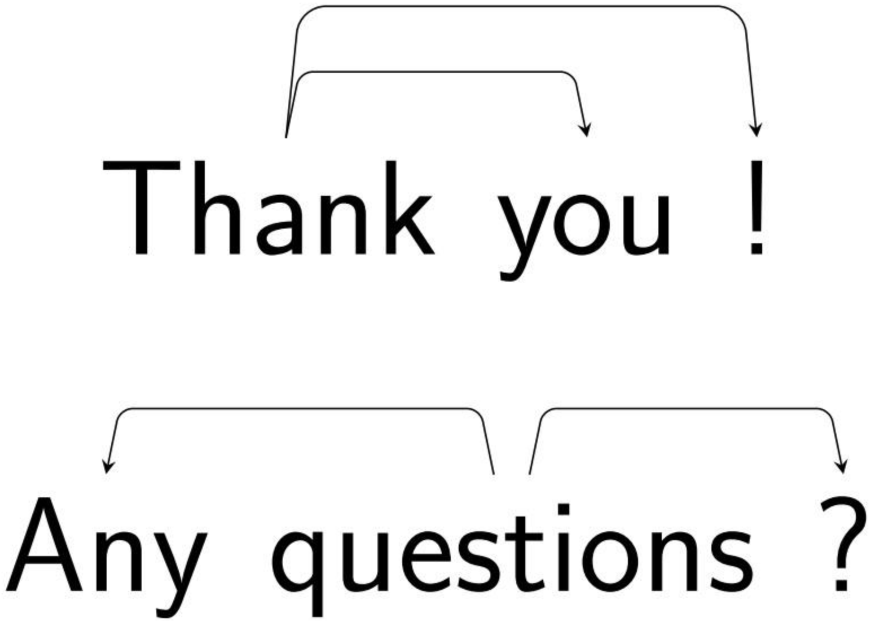
- start with a vector of zeros
- move towards the gradient
- to maximize the probability / minimize the loss function

$$\hat{w} = \underset{w}{\operatorname{argmax}} \sum_{j} \log P(y^{(j)} | x^{(j)})$$

6. Error correction use case

Error correction use case

https://bit.ly/31GFZmd
or
https://github.com/mariana-scorp/esscass-2019-nlp/blob/master/2-features/adjective-vs-adverb.ipynb



mariana.romanyshyn@grammarly.com