Intro to Natural Language Processing

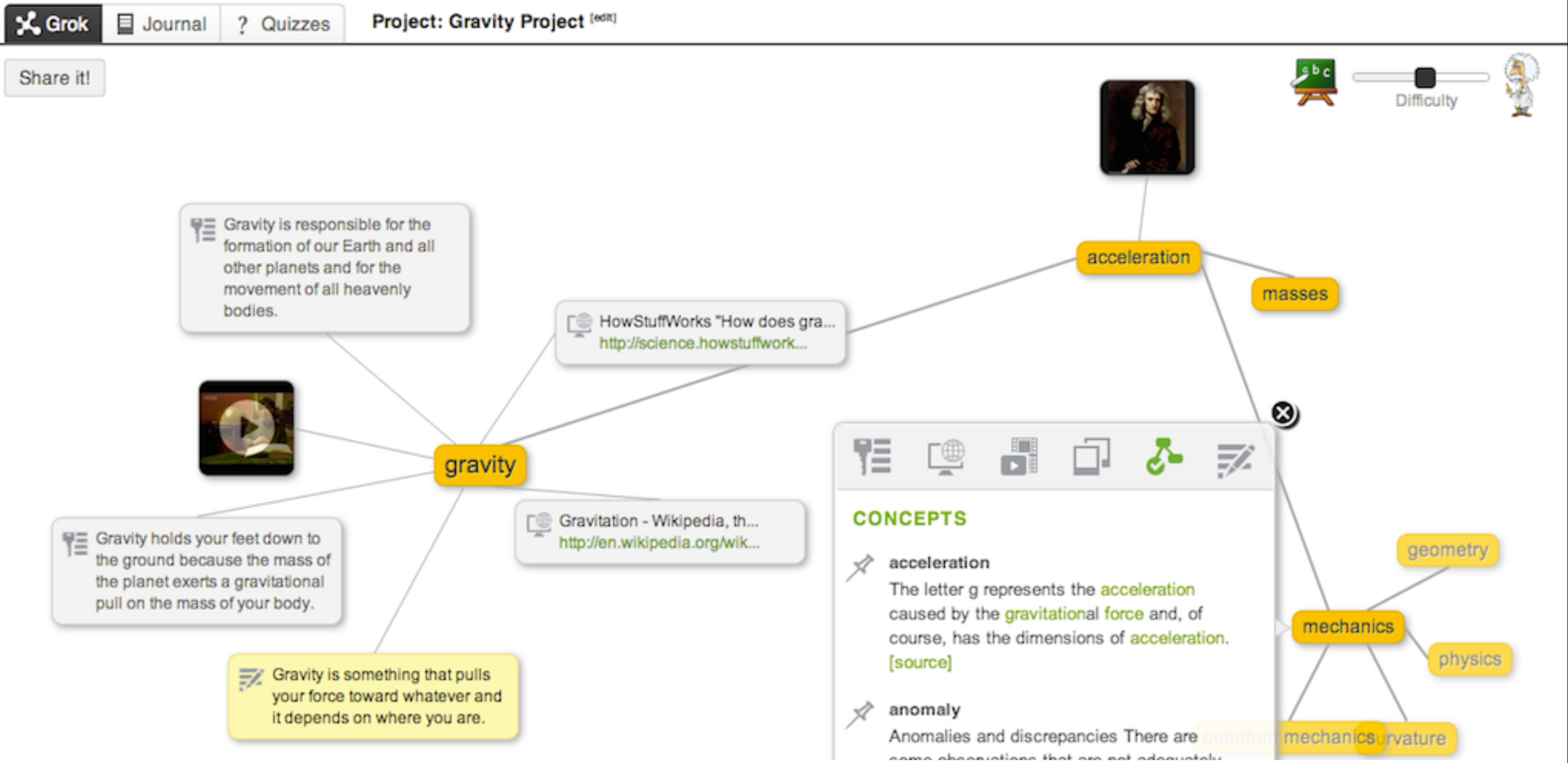
Kirill Kireyev

General Assembly

My Background

- PhD Computer / Cognitive Science
  - University of Colorado
- R&D @ Pearson Education
  - Adaptive vocabulary instruction
  - Essay scoring
  - Grammar/writing feedback

instaGrok



Project: Gravity Project [edit]

Gravity is responsible for the formation of our Earth and all other planets and for the movement of all heavenly bodies.

HowStuffWorks "How does gra...
http://science.howstuffwork...

Gravity holds your feet down to the ground because the mass of the planet exerts a gravitational pull on the mass of your body.

Gravitation - Wikipedia, th...
http://en.wikipedia.org/wik...

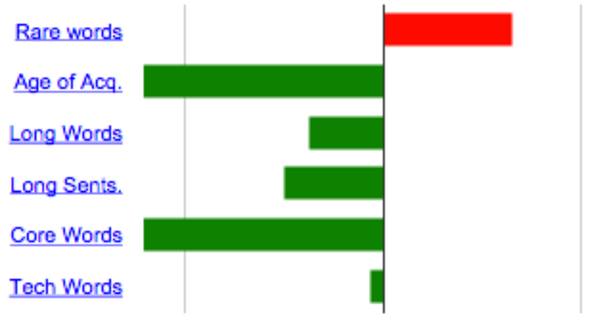
Gravity is something that pulls your force toward whatever and it depends on where you are.

CONCEPTS

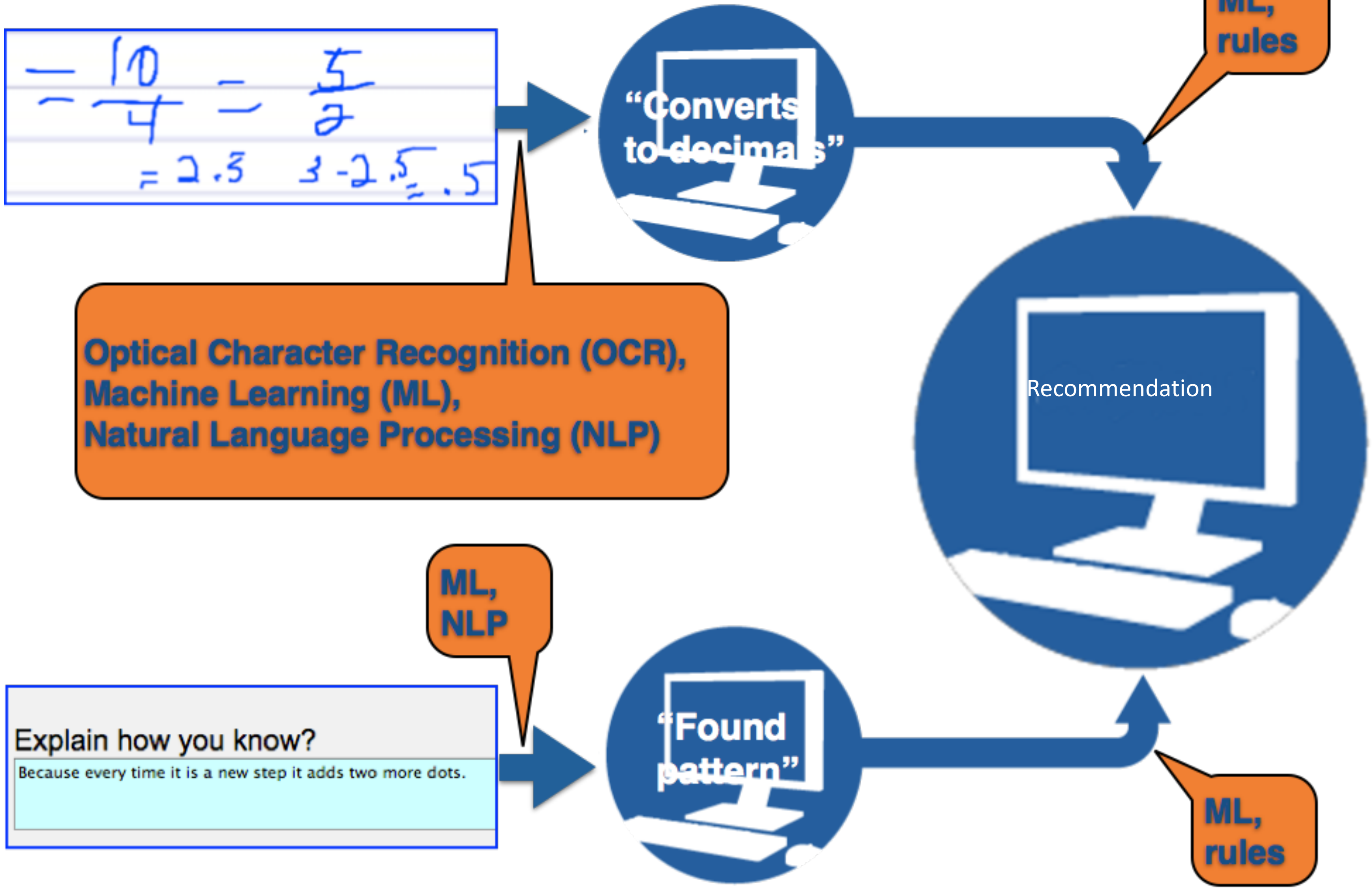
acceleration
The letter g represents the acceleration caused by the gravitational force and, of course, has the dimensions of acceleration. [source]

anomaly
Anomalies and discrepancies There are

TEXT GENOME

Title	Length	Lexile	Grade	Vocab Words	Themes	Other
The Zoo Crew	582	763	5	healthy ranger volunteer	Animals (45) Types of people (14) Ownership/possession (13) Disease/health (11) Destructive/helpful (10)	

CatZilla™



$= \frac{10}{4} = \frac{5}{2}$

$= 2.5 \quad 3 - 2.5 = .5$

Explain how you know?

Because every time it is a new step it adds two more dots.

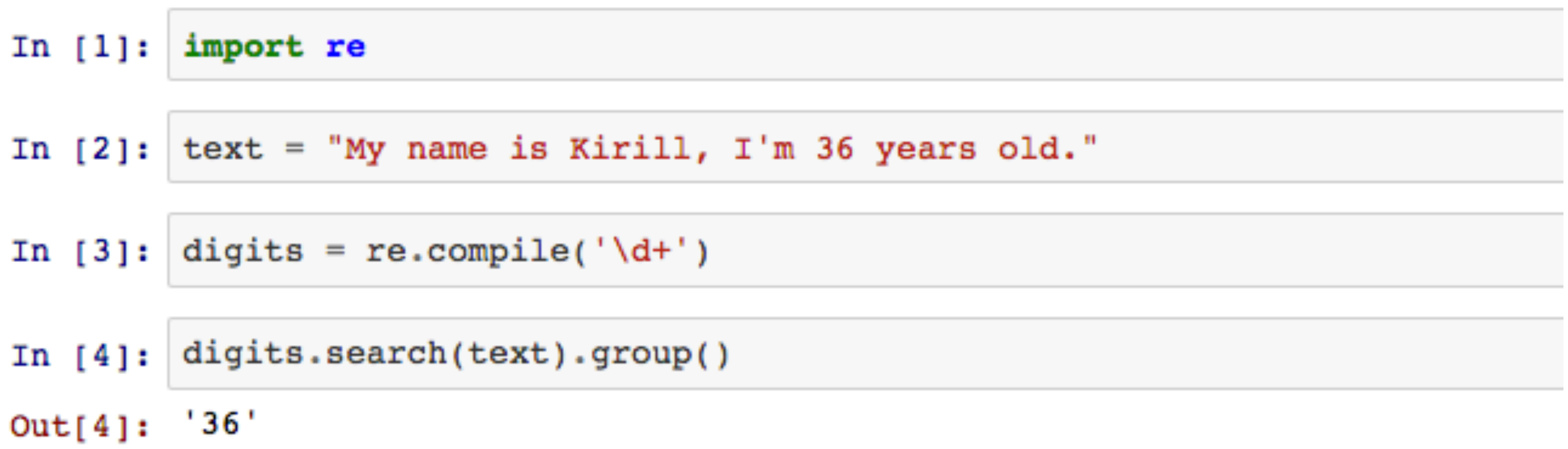
“Simple” Problem

Count the number of sentences in a text.

How would you do it?

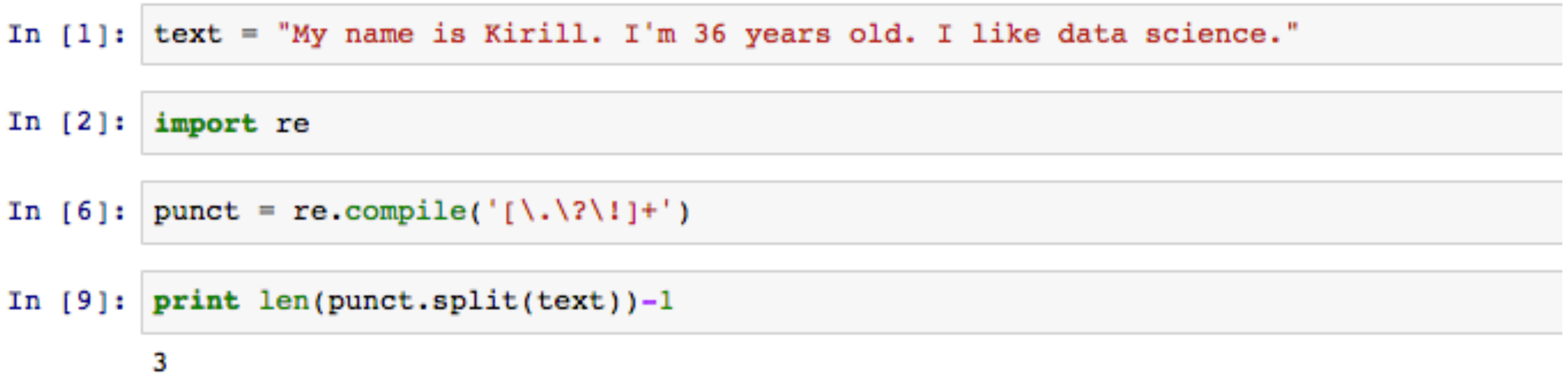
Count the punctuation?

Regular expressions



```
In [1]: import re

In [2]: text = "My name is Kirill, I'm 36 years old."

In [3]: digits = re.compile('\d+')

In [4]: digits.search(text).group()
Out[4]: '36'
```

Sentence segmentation with regular expressions



```
In [1]: text = "My name is Kirill. I'm 36 years old. I like data science."

In [2]: import re

In [6]: punct = re.compile('[\.\?\!]+')

In [9]: print len(punct.split(text))-1

        3
```

Not so simple!

“The incident took place at 2 p.m. on Monday.”

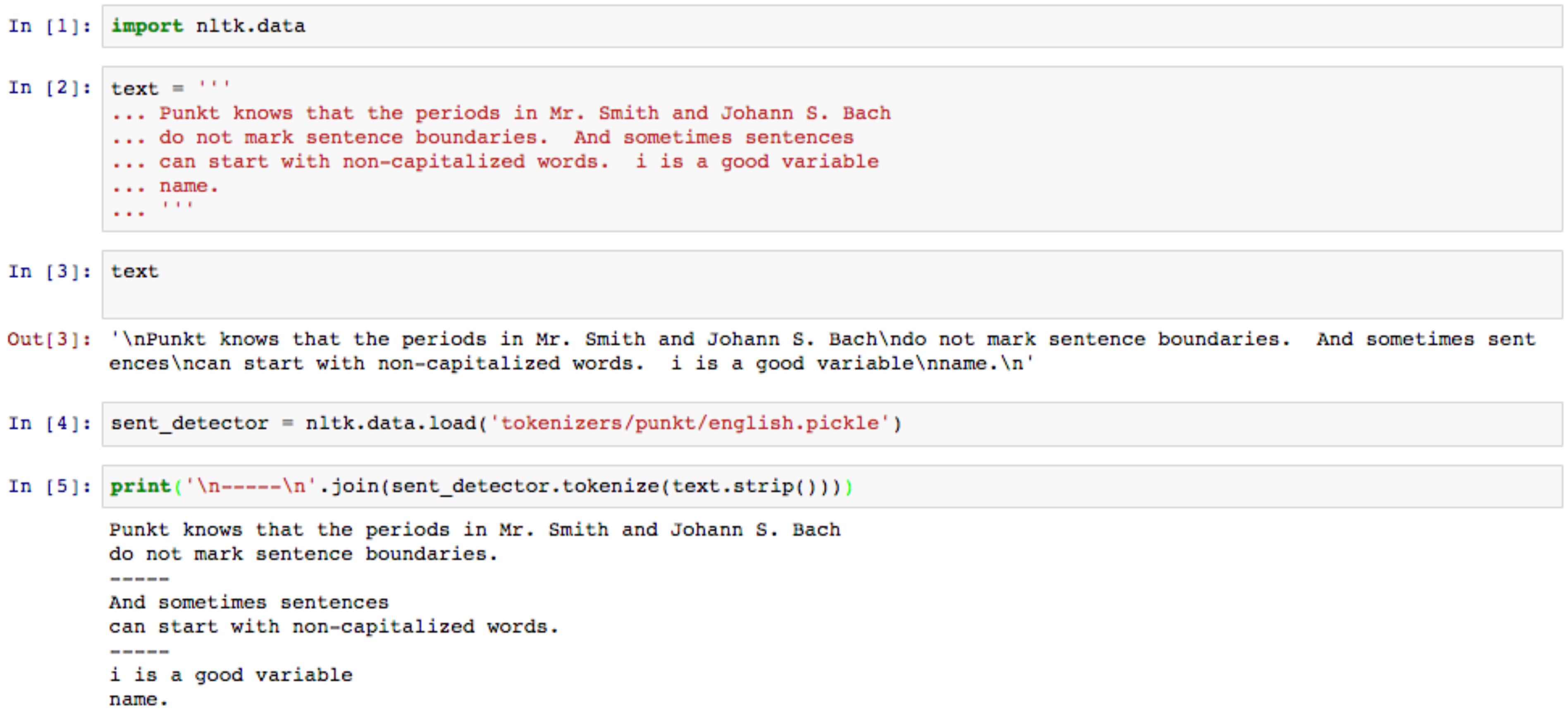
“The federal government has begun a civil rights investigation into how Freddie Gray died of a spinal injury after he was arrested in Baltimore, the U.S. Justice Department said Tuesday.”

“Mayor Stephanie Rawlings-Blake and Police Commissioner Anthony W. Batts called for calm to allow police to complete their investigation.”

Using Machine Learning

- Annotate a lot of sentences
- Train a classifier
- Features?
  - Preceding word(s)
  - Following word(s)
  - Capitalization
  - Length of word

NLTK



```
In [1]: import nltk.data
```

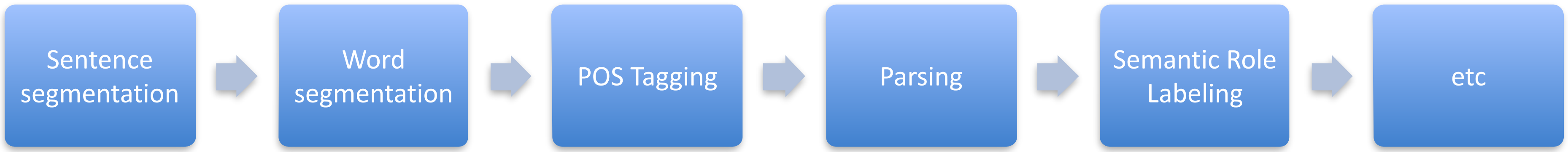
```
In [2]: text = '''
        ... Punkt knows that the periods in Mr. Smith and Johann S. Bach
        ... do not mark sentence boundaries.  And sometimes sentences
        ... can start with non-capitalized words.  i is a good variable
        ... name.
        ... '''
```

```
In [3]: text

Out[3]: '\nPunkt knows that the periods in Mr. Smith and Johann S. Bach\ndo not mark sentence boundaries.  And sometimes sent
        ences\ncan start with non-capitalized words.  i is a good variable\nname.\n'
```

```
In [4]: sent_detector = nltk.data.load('tokenizers/punkt/english.pickle')
```

```
In [5]: print('\n-----\n'.join(sent_detector.tokenize(text.strip())))

        Punkt knows that the periods in Mr. Smith and Johann S. Bach
        do not mark sentence boundaries.
        -----
        And sometimes sentences
        can start with non-capitalized words.
        -----
        i is a good variable
        name.
```

Understanding Language



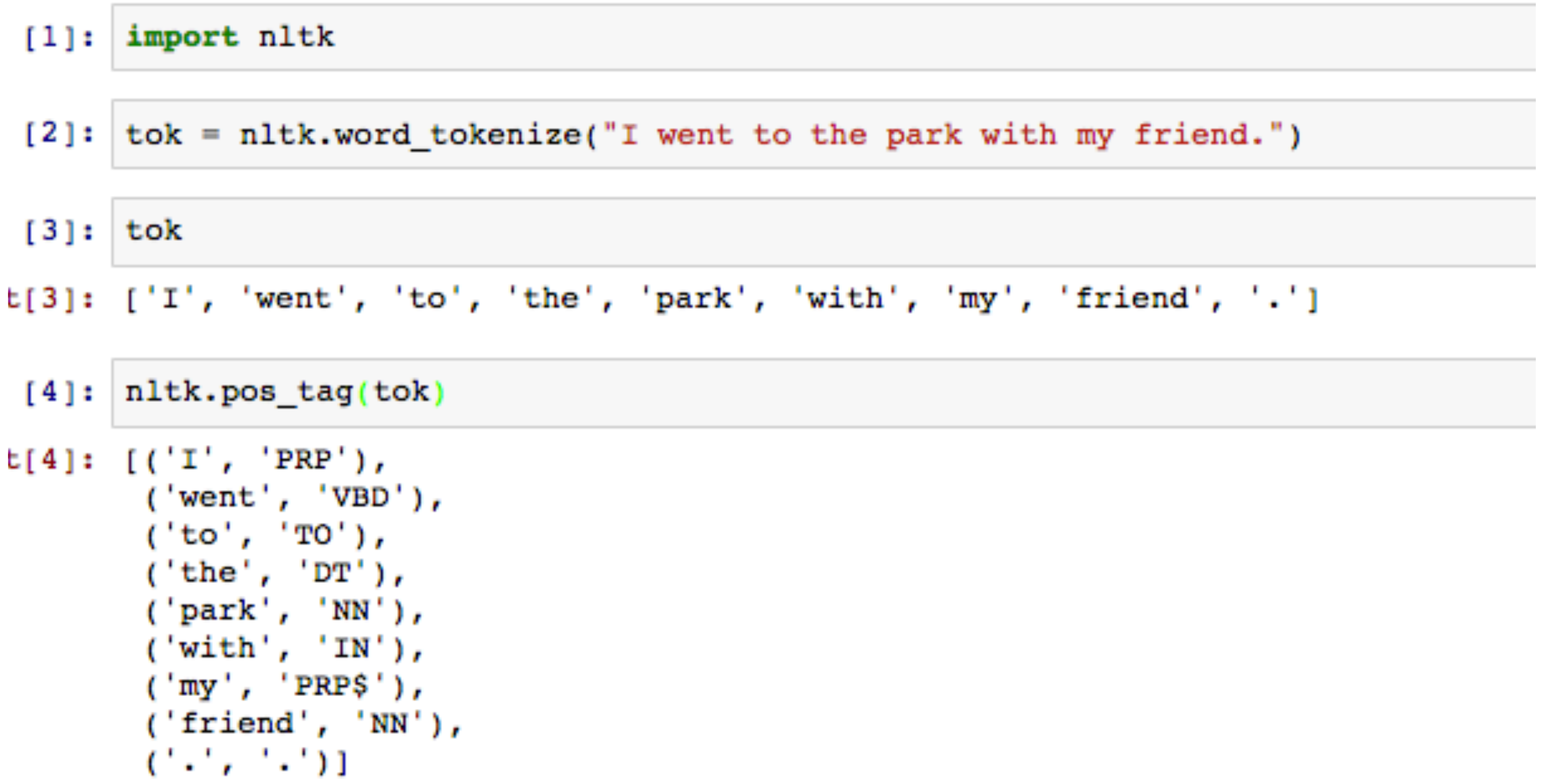
Part of Speech Tagging

I	went	to	the	park	with	my	friend.
PRP	VBD	TO	DT	NN	IN	PRP$	NN

I went to park my car.

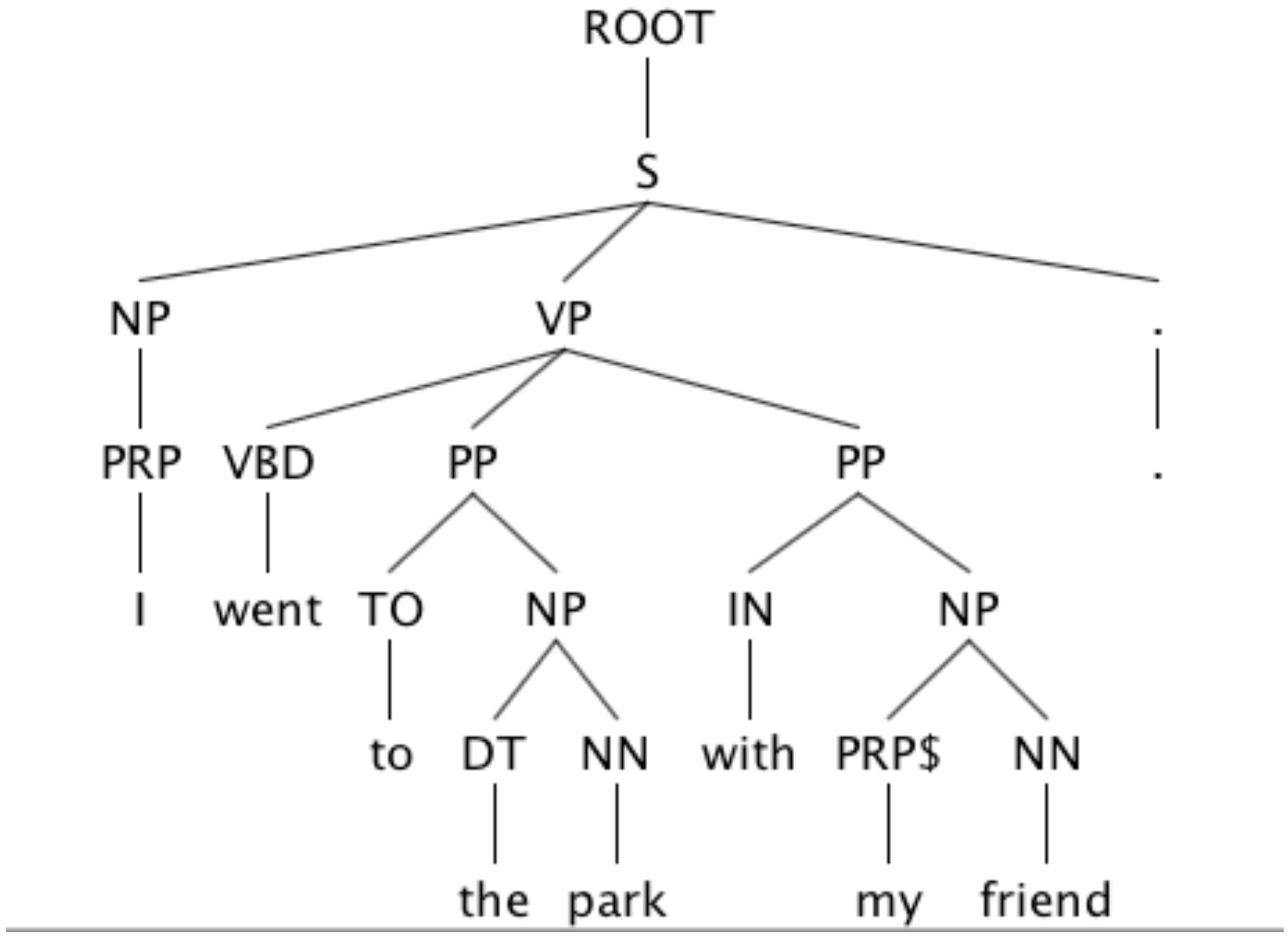
Will you friend me on Facebook?

NLTK POS Tagging



```
 [1]: import nltk

 [2]: tok = nltk.word_tokenize("I went to the park with my friend.")

 [3]: tok

t[3]: ['I', 'went', 'to', 'the', 'park', 'with', 'my', 'friend', '.']

 [4]: nltk.pos_tag(tok)

t[4]: [('I', 'PRP'),
       ('went', 'VBD'),
       ('to', 'TO'),
       ('the', 'DT'),
       ('park', 'NN'),
       ('with', 'IN'),
       ('my', 'PRP$'),
       ('friend', 'NN'),
       ('.', '.')]
```

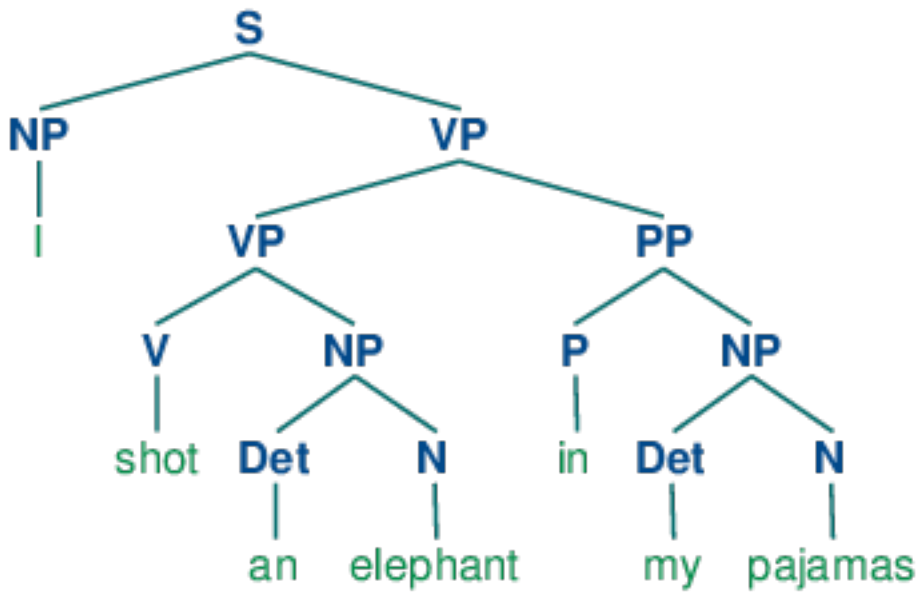
Parsing

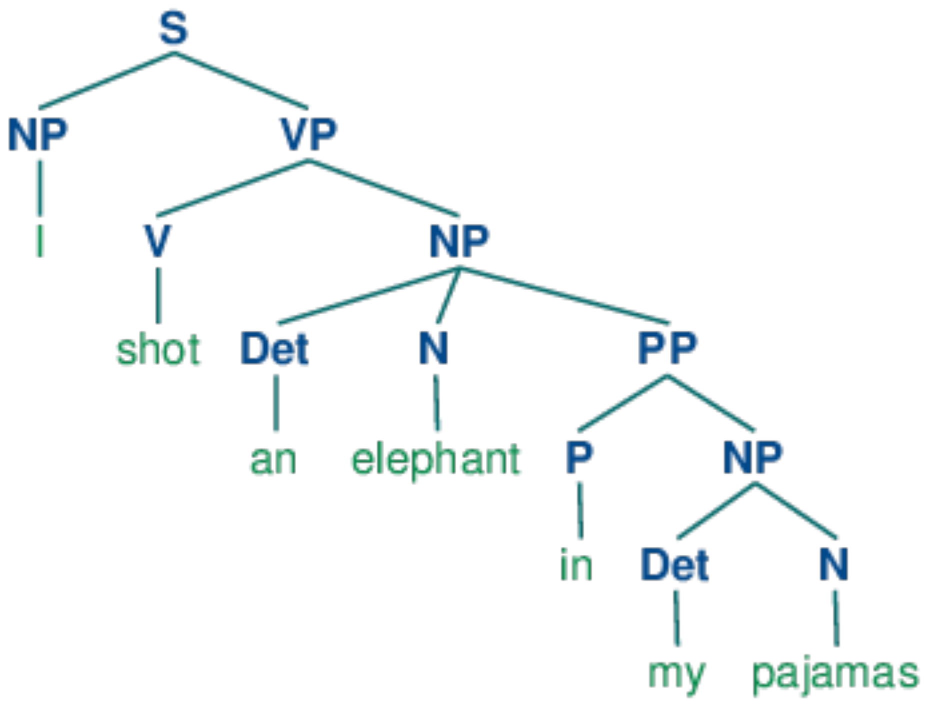

ROOT
S
NP
VP
.
PRP
VBD
PP
PP
.
I
went
TO
NP
IN
NP
to
DT
NN
with
PRP$
NN
the
park
my
friend

Parsing Ambiguity

While hunting in Africa, I shot an elephant in my pajamas.

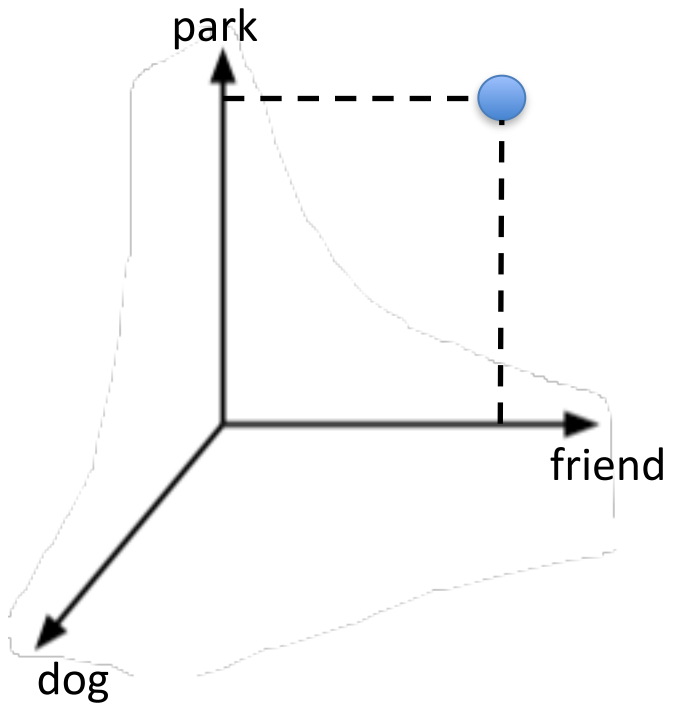
How he got into my pajamas, I don't know.





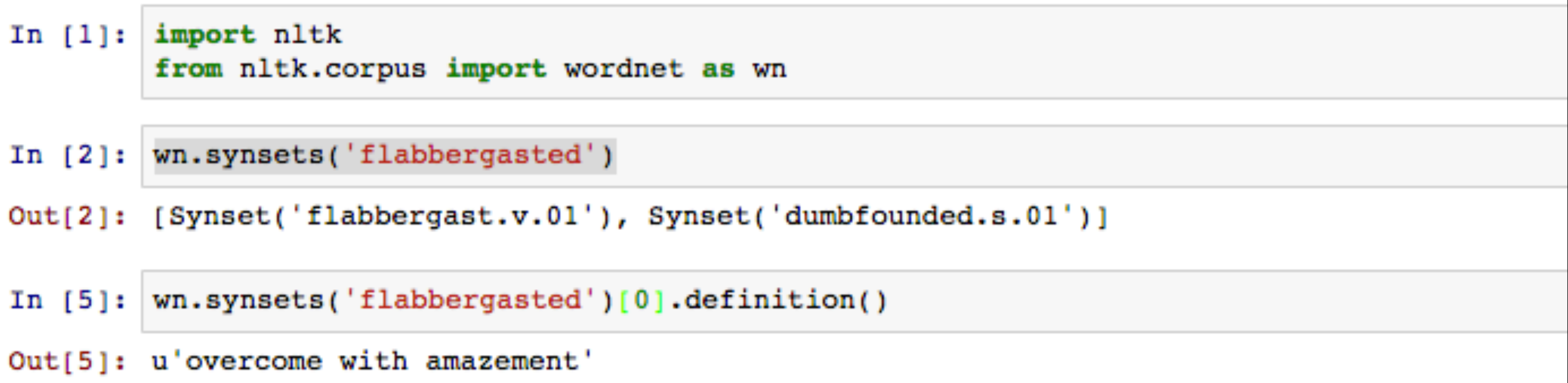
Vector Representation

- Instead of deep semantic representation, represent a text with a vector (of words)
- "I went with my friend to the park" -> { "go", "friend", "park" }
- Remove *stopwords* ("to")
- Dimensionality = { # words in lexicon }
  - Each word = independent dimension
- Ignores word order ("bag of words")
- Can do similarity comparisons using linear algebra methods (e.g. cosine)
- Future: dimensionality reduction (SVD)



WordNet

- http://wordnet.princeton.edu
- Example: Finding synonyms NLTK Wordnet



```
In [1]: import nltk
        from nltk.corpus import wordnet as wn

In [2]: wn.synsets('flabbergasted')

Out[2]: [Synset('flabbergast.v.01'), Synset('dumbfounded.s.01')]

In [5]: wn.synsets('flabbergasted')[0].definition()

Out[5]: u'overcome with amazement'
```

Sentiment Analysis

- “iPhone 6 is a fantastic product”
- “iPhone is difficult to use”
- Words with *sentiment polarity*:
  - Simple: fantastic: +1 / difficult: -1
- How to obtain polarity values?
  - Use annotated texts
  - Reviews
- Retain words (features) with high information gain:
  - E.g. Chi squared

Sentiment Analysis Complications

- “iPhone 6 is a fantastic product”
- “iPhone is difficult to use”
- “iPhone 6 is not great”
- “iPhone 6 is not necessarily the most amazing product I’ve used”
- Domain-dependent