Sentiment Analysis by NLTK

Wei-Ting Kuo
PyconApac2015

http://goo.gl/wJelD4

Sentiment Analysis?

Aim to determine the attitude of a speaker/writer with respect to some text.

Amazon's Customer Review

Customer Reviews

1,651

4.2 out of 5 stars

5 star	65%
4 star	13%
3 star	9%
2 star	5%
1 star	8%

Share your thoughts with other customers

Write a customer review

See all 1,651 customer reviews ›

Most Helpful Customer Reviews

1,011 of 1,031 people found the following review helpful

WARNING: Not All Extras Included!!!

By E-Transitions on March 21, 2010

Format: DVD

WARNING: This edition DOES NOT contain all the special features available with the New Moon release. Summit did an evil, evil, manipulative thing with this DVD release and divided up the special features among multiple retailers.

On Amazon you have just the standard discs with a limited number of extras.

If you buy your version at Target, you get an extra disc with Deleted Scenes, Interview with the Volturi, Fandimonium, The Beat Goes On: The Music of Twilight, and Frame by Frame: Storyboards to Screen.

If you buy at Borders, you get extras including Extended Scenes.

And if you buy at Walmart, you get a Sneak Peek at Eclipse (which includes an Eclipse scene), Team Edward v. Team Jacob, Becoming Jacob, Introducing the Wolfpack, Jacob Fast Forward, Edward Fast Forward, and Shooting in Italy.

Summit's hoping you buy THREE copies so that you can get to see all the special features they divided up. Don't give them the satisfaction! Buy one and call it a day!

Customer Images

See all customer images

Most Recent Customer Reviews

Five Stars

I love the saga it is very entertaining n a great love story.

Published 3 hours ago by Ruby T. Ward

Five Stars

buenisimas

Published 1 day ago by Rosa M. Fernández Mora

Five Stars

Great Product!!

Published 2 days ago by Cynthia

If you have to fold laundry, watch this movie...

If you need to read a review to determine whether this is a good use of your time...then feel free to watch the movie - read all the books - then go watch the rest of the movies. Read more

Published 2 days ago by Leah R. Guis

Four Stars

:)

Published 2 days ago by Hayde Longoria

Four Stars

Always a good movie

Twitter Follower's replies

Positive or Negative?

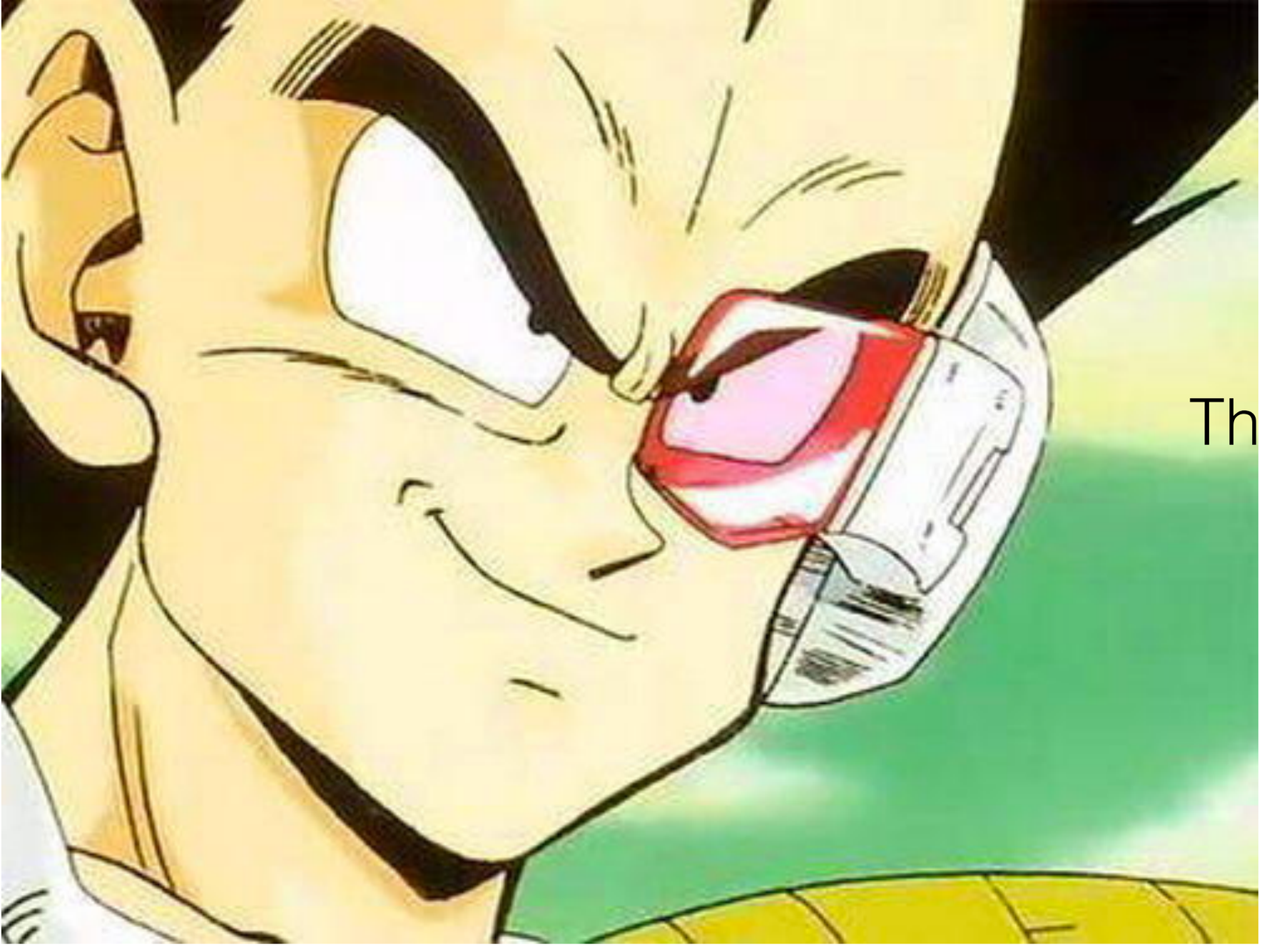
This is a **good** book! → Positive

This is a **good** book!
I **like** it! → (more) Positive

This is a **bad** book! → Negative

The first chapter is **good**,
but the rest is **terrible** → Negative

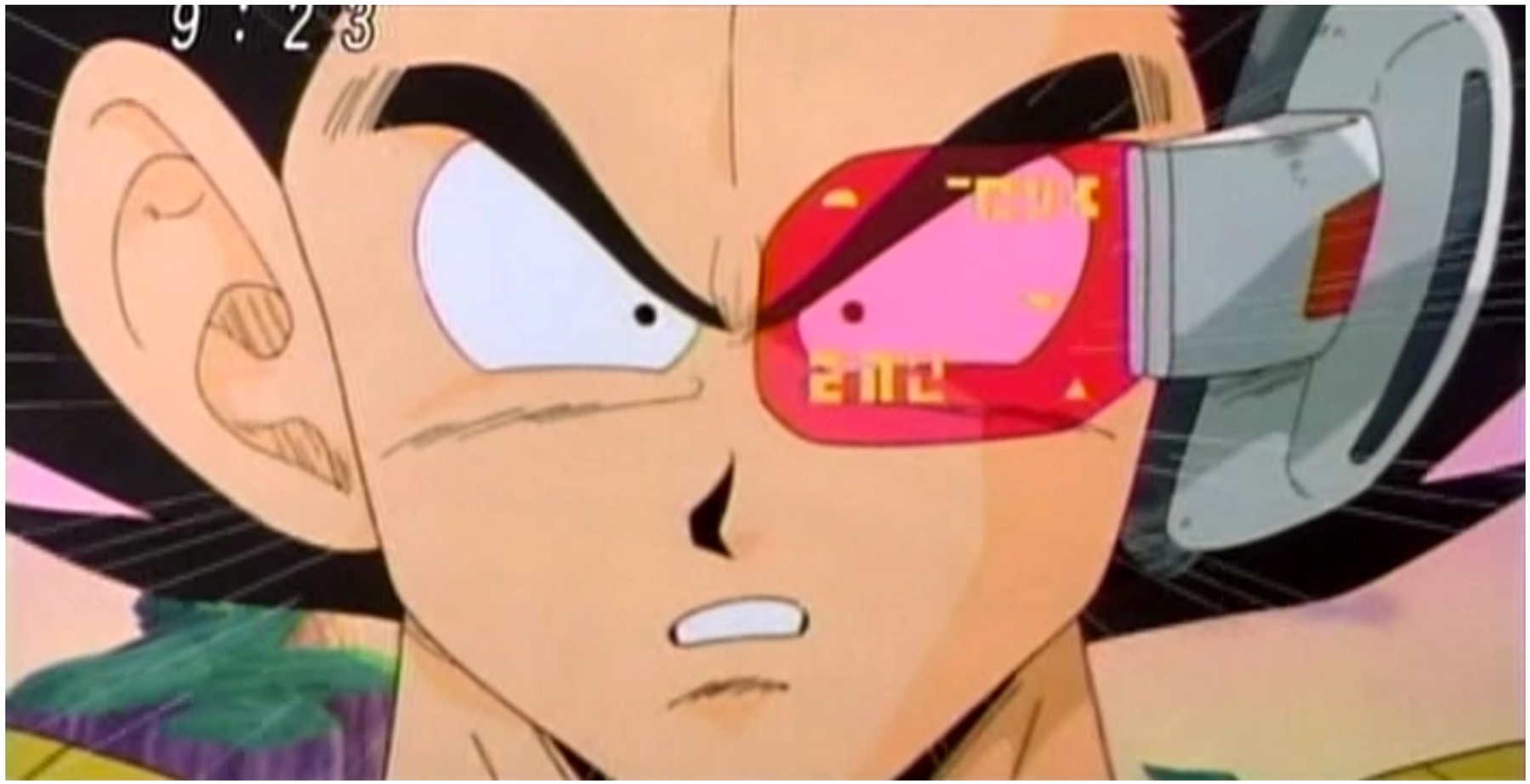
How to compute it?



This is a good book!

From scouter: This is a postive review

Why is sentiment analysis useful?



- This is a terrible book. Because it’s important, so I mention three times, terrible, terrible, terrible!!!

Let’s begin with the easiest way!

Sentiment Dictionary

like 1

good 2

bad -2

terrible -3

This is a **good** book! → 2 → Positive

Dictionary
like 1
good 2
bad -2
terrible -3

This is a **good** book!
I **like** it! → 3 → (more) Positive

This is a **bad** book! → -2 → Negative

The first chapter is **good**,
but the rest is **terrible** → -1 → Negative

AFINN-111

- http://www2.imm.dtu.dk/pubdb/views/publication_details.php?id=6010

- A list of words rated between -5 (neg) to 5 (pos)

Let's build the dictionary in Python

```
sentiment_dictionary = {}

for line in open('data/AFINN-111.txt'):
    word, score = line.split('\t')
    sentiment_dictionary[word] = int(score)
```

Let’s split the sentence first

```
words = 'This is a good book'.lower().split()
print(words)
```

```
['this', 'is', 'a', 'good', 'book']
```

And compute the score

```
sum( sentiment_dictionary.get(word, 0) for word in words )
```

```
3
```

Recap

```
sentiment_dictionary = {}

for line in open('data/AFINN-111.txt'):
    word, score = line.split('\t')
    sentiment_dictionary[word] = int(score)
```

```
words = 'this is a good book'.lower().split()
print(words)
```

```
['this', 'is', 'a', 'good', 'book']
```

```
sum( sentiment_dictionary.get(word, 0) for word in words )
```

```
3
```

What if the text is long?
And have many punctuation?

- Nice book! Though it is lack of advanced topics. It’s still good for beginners.

Doesn’t work!

Slide Type -

```
words = '''Nice book! Though it is lack of advanced topics.
        It’s still good for beginners.'''.lower().split()
print(words)
```

```
['nice', 'book!', 'though', 'it', 'is', 'lack', 'of', 'advanced', 'topic
s.', 'it’s', 'still', 'good', 'for', 'beginners.']
```

Big Data Borat
@BigDataBorat

Following

In Data Science, 80% of time spent prepare data, 20% of time spent complain about need for prepare data.

RETWEETS 409

FAVORITES 155

6:47 PM - 26 Feb 2013

NLTK to the rescue

- Natural Language ToolKit
- Works with Python3!

Tokenization

the process of breaking a stream of text up into words, phrases, symbols, or other meaningful elements called tokens

Word tokenize

Slide Type -

```
from nltk.tokenize import word_tokenize

words = word_tokenize('''Nice book! Though it is lack of advanced topics.
                         it's still good for beginners.'''.lower())
print(words)
```

```
['nice', 'book', '!', 'though', 'it', 'is', 'lack', 'of', 'advanced', 'top
ics', '.', 'it's', 'still', 'good', 'for', 'beginners', '.']
```

Slide Type -

```
sum( sentiment_dictionary.get(word, 0) for word in words )
```

```
3
```

It seems we lose some information

- **Nice** book! Though it is **lack** of advanced topics. It's still **good** for beginners.

Positive: 3

Separate to multiple sentences first

- **Nice** book! **Positive**
- Though it is **lack** of advanced topics. **Negative**
- It’s still **good** for beginners. **Positive**

How to split the text to sentences?

Sentence tokenize

Slide Type -

```
from nltk.tokenize import sent_tokenize

sentences = sent_tokenize('''Nice book! Though it is lack of advanced topic

for s in sentences: print(s)
```

```
nice book!
though it is lack of advanced topics.
it’s still good for beginners.
```

Compute score for each sentence

Slide Type -

```
from nltk.tokenize import sent_tokenize

sentences = sent_tokenize('''Nice book! Though it is lack of advanced topic

for s in sentences: print(s)
```

```
nice book!
though it is lack of advanced topics.
it’s still good for beginners.
```

Slide Type -

```
for sentence in sentences:
    words = word_tokenize(sentence)
    print( sum( sentiment_dictionary.get(word, 0) for word in words) )
```

```
3
-1
3
```

But we still miss some information in another case

The first chapter is **good**,
but the rest is **terrible** and **confusing**

Negative

It's a **bad** idea to buy this book.

Negative

At least the customer mentioned something good,
but it's not recorded

Let’s count Pos & Neg separately

The first chapter is **good**,
but the rest is **terrible** and **confusing**

Pos:3 **Neg: -5**

It’s a **bad** idea to buy this book.

Neg: -3

In Python

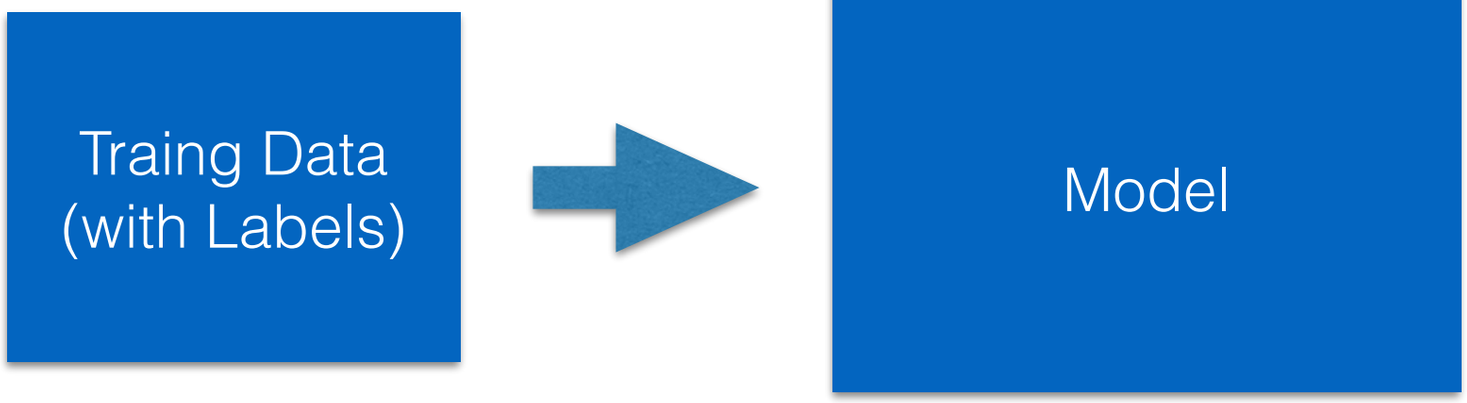
```
text = '''The first chapter is good, but the rest is terrible and confusing.
        It's a bad idea to buy this book.'''

result = []
for setence in sent_tokenize(text):
    pos = 0
    neg = 0
    for word in word_tokenize(setence):
        score = sentiment_dictionary.get(word, 0)
        if score > 0:
            pos += score
        if score < 0:
            neg += score
    result.append([pos, neg])

for s in result: print(s)
```

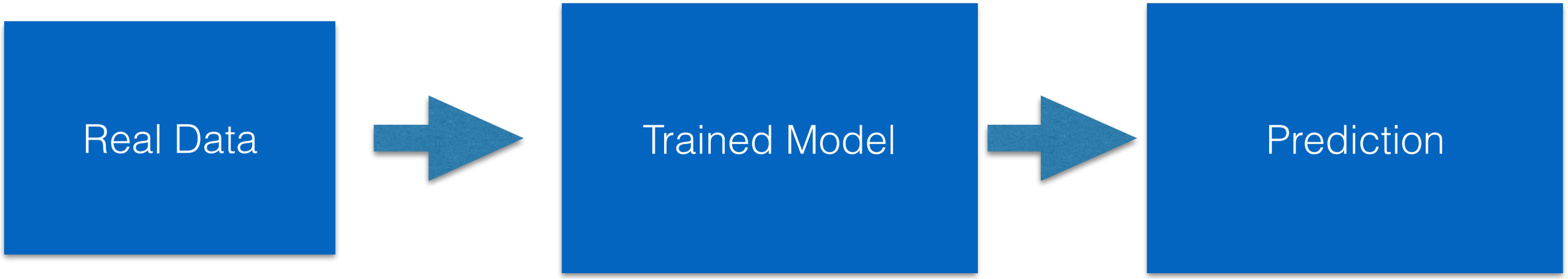
```
[3, -5]
[0, -3]
```

how about new words?

how about domain specific term?

Machine Learning!!!


Traing Data
(with Labels)
Model


Real Data
Trained Model
Prediction

Training Data

- This is a good book! Postive
- This is a awesome book! Postive
- This is a bad book! Negative
- This is a terrible book! Negative

- This is a good book! Postive
- This is a awesome book! Postive
- This is a bad book! Negative
- This is a terrible book Negative

The format NLTK use

```
def format_sentence(sentence):
    return {word: True for word in word_tokenize(sentence) }

format_sentence('this is a good book')
```

```
{'a': True, 'book': True, 'good': True, 'is': True, 'this': True}
```

Prepare the Training Set

Slide

```
s1 = 'this is a good book'
s2 = 'this is a awesome book'
s3 = 'this is a bad book'
s4 = 'this is a terrible book'
traing_data = [[format_sentence(s1), 'pos'],
               [format_sentence(s2), 'pos'],
               [format_sentence(s3), 'neg'],
               [format_sentence(s4), 'neg']]

for t in traing_data: print(t)
```

```
[{'book': True, 'a': True, 'is': True, 'this': True, 'good': True}, 'pos']
[{'awesome': True, 'a': True, 'is': True, 'this': True, 'book': True}, 'pos']
[{'book': True, 'a': True, 'is': True, 'this': True, 'bad': True}, 'neg']
[{'terrible': True, 'a': True, 'is': True, 'this': True, 'book': True}, 'neg']
```

Build the model, and train it!

```
from nltk.classify import NaiveBayesClassifier

model = NaiveBayesClassifier.train(traing_data)

model.classify(format_sentence('this is a good article'))
```

```
'pos'
```

Real Case

- Movie Review Data
  http://www.cs.cornell.edu/people/pabo/movie-review-data/

- 5331 positive reviews & 5331 negative reviews labelled by human.

Positive Reviews

```
ipython  vim vi...  ipython  ..15...FINN  ..15...FINN  ..we...ent  vim vi...  ipython  vim  ipython ...  ipython  ipython >>  +
  1 the rock is destined to be the 21st century's new " conan " and that he's going to make
     a splash even greater than arnold schwarzenegger , jean-claud van damme or steven sega
    l .
  2 the gorgeously elaborate continuation of " the lord of the rings " trilogy is so huge t
    hat a column of words cannot adequately describe co-writer/director peter jackson's exp
    anded vision of j . r . r . tolkien's middle-earth .
  3 effective but too-tepid biopic
  4 if you sometimes like to go to the movies to have fun , wasabi is a good place to start
     .
  5 emerges as something rare , an issue movie that's so honest and keenly observed that it
     doesn't feel like one .
  6 the film provides some great insight into the neurotic mindset of all comics -- even th
    ose who have reached the absolute top of the game .
  7 offers that rare combination of entertainment and education .
  8 perhaps no picture ever made has more literally showed that the road to hell is paved w
    ith good intentions .
  9 steers turns in a snappy screenplay that curls at the edges ; it's so clever you want t
    o hate it . but he somehow pulls it off .
 10 take care of my cat offers a refreshingly different slice of asian cinema .
 11 this is a film well worth seeing , talking and singing heads and all .
 12 what really surprises about wisegirls is its low-key quality and genuine tenderness .
 13  ( wendigo is ) why we go to the cinema : to be fed through the eye , the heart , the m
    ind .
"rt-polarity-pos.txt" [converted] 5331L, 626349C
```

Negative Reviews

```
1 simplistic , silly and tedious .
2 it's so laddish and juvenile , only teenage boys could possibly find it funny .
3 exploitative and largely devoid of the depth or sophistication that would make watching
   such a graphic treatment of the crimes bearable .
4 [garbus] discards the potential for pathological study , exhuming instead , the skewed
  melodrama of the circumstantial situation .
5 a visually flashy but narratively opaque and emotionally vapid exercise in style and my
  stification .
6 the story is also as unoriginal as they come , already having been recycled more times
  than i'd care to count .
7 about the only thing to give the movie points for is bravado -- to take an entirely sta
  le concept and push it through the audience's meat grinder one more time .
8 not so much farcical as sour .
9 unfortunately the story and the actors are served with a hack script .
10 all the more disquieting for its relatively gore-free allusions to the serial murders ,
    but it falls down in its attempts to humanize its subject .
11 a sentimental mess that never rings true .
12 while the performances are often engaging , this loose collection of largely improvised
    numbers would probably have worked better as a one-hour tv documentary .
13 interesting , but not compelling .
14 on a cutting room floor somewhere lies . . . footage that might have made no such thing
    a trenchant , ironic cultural satire instead of a frustrating misfire .
@
```

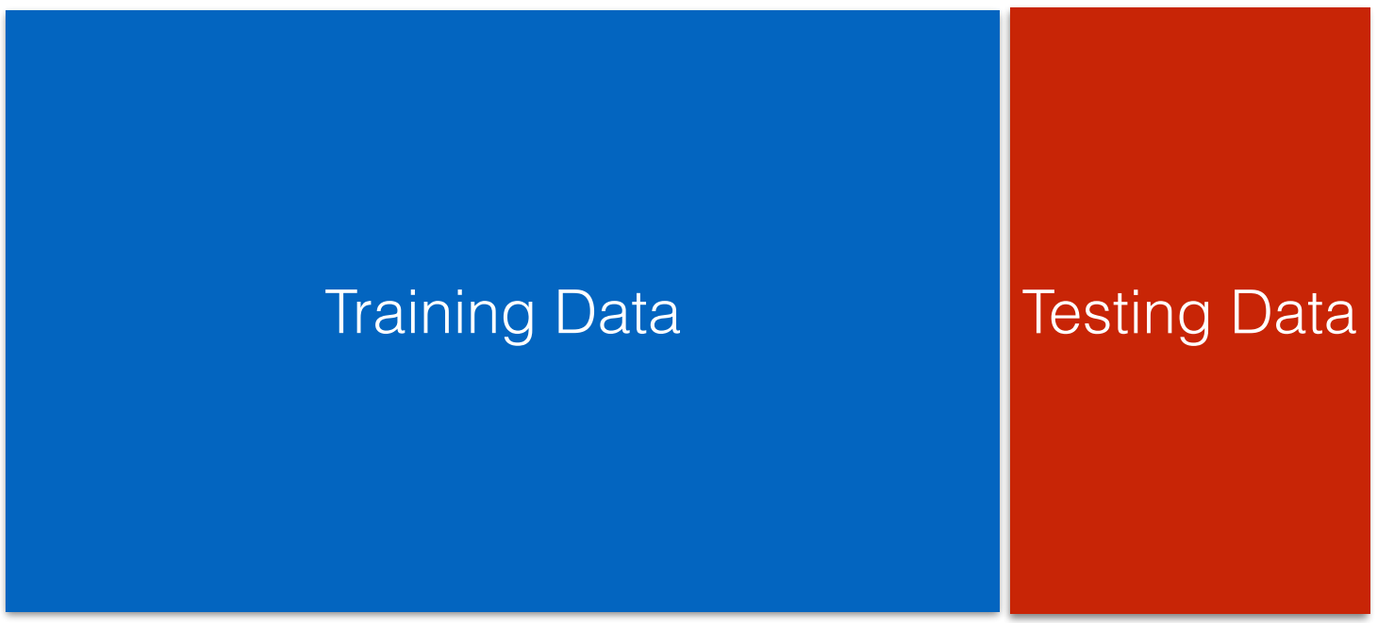
Read our data

Slide Type

```
pos_data = []
with open('data/rt-polaritydata/rt-polarity-pos.txt', encoding='latin-1') as f:
    for line in f:
        pos_data.append([format_sentence(line), 'pos'])


neg_data = []
with open('data/rt-polaritydata/rt-polarity-neg.txt', encoding='latin-1') as f:
    for line in f:
        neg_data.append([format_sentence(line), 'neg'])
```

Separate the data

Training data to train the model
Testing data to compute the accuracy



Separate our data

```
training_data = pos_data[:4000] + neg_data[:4000]
testing_data = pos_data[4000:] + neg_data[4000:]
```

Train the data

```
model = NaiveBayesClassifier.train(training_data)

print( model.classify(format_sentence('this is a nice article')) )
print( model.classify(format_sentence('this is a bad article')) )
```

```
pos
neg
```

Compute the accuracy

accuracy = number of correct / total

```
from nltk.classify.util import accuracy
accuracy(model, testing_data)
```

```
0.7772351615326822
```

How to enhance?

- Use the most frequent 1000 words only
- Use different model, maybe SVC
- Read more paper about the latest research

Q & A