Text Processing and Classification

with Python

Fatih Erikli
Software Developer
@Adphorus

fatiherikli@gmail.com
http://fatiherikli.com

Text Processing

Analysis and manipulation of electronic text.

- Generating
- Parsing
- Similarity
- Translation
- Classification

Python?

Python have rich ecosystem for text processing.

- NLTK
- TextBlob
- Pyparsing
- Builtin Modules

TextBlob

It provides a simple API for diving into common natural language processing (NLP) tasks.

- Part of Speech Tagging
- Noun-phrase Extraction
- Wordnet Integration
- Sentiment Analysis

Simple API

2. ~ | python (python)

```
>>> from textblob import TextBlob
>>> blob = TextBlob('Python is a widely used high-level, general-purpose, interpreted, dynamic programming lan
guage. Its design philosophy emphasizes code readability, and its syntax allows programmers to express concept
s in fewer lines of codethan would be possible in languages such as C++ or Java.')
>>> blob.pos_tags
[(u'Python', u'NNP'), (u'is', u'VBZ'), (u'a', u'DT'), (u'widely', u'RB'), (u'used', u'VBN'), (u'high-level', u
'JJ'), (u'general-purpose', u'JJ'), (u'interpreted', u'VBN'), (u'dynamic', u'JJ'), (u'programming', u'NN'), (u
'language', u'NN'), (u'Its', u'PRP$'), (u'design', u'NN'), (u'philosophy', u'NN'), (u'emphasizes', u'VBZ'), (u
'code', u'NN'), (u'readability', u'NN'), (u'and', u'CC'), (u'its', u'PRP$'), (u'syntax', u'NN'), (u'allows', u
'VBZ'), (u'programmers', u'NNS'), (u'to', u'TO'), (u'express', u'VB'), (u'concepts', u'NNS'), (u'in', u'IN'),
(u'fewer', u'JJR'), (u'lines', u'NNS'), (u'of', u'IN'), (u'codethan', u'NN'), (u'would', u'MD'), (u'be', u'VB'
), (u'possible', u'JJ'), (u'in', u'IN'), (u'languages', u'NNS'), (u'such', u'JJ'), (u'as', u'IN'), (u'C', u'NN
'), (u'+', u'SYM'), (u'+', u'SYM'), (u'or', u'CC'), (u'Java', u'NNP')]
>>> blob.noun_phrases
WordList(['python', u'design philosophy', u'code readability', 'c++', 'java'])
>>>
```

```
>>> animals = TextBlob("cat dog octopus")
>>> animals.words
WordList(['cat', 'dog', 'octopus'])
>>> animals.words.pluralize()
WordList(['cats', 'dogs', 'octopodes'])
>>>
```

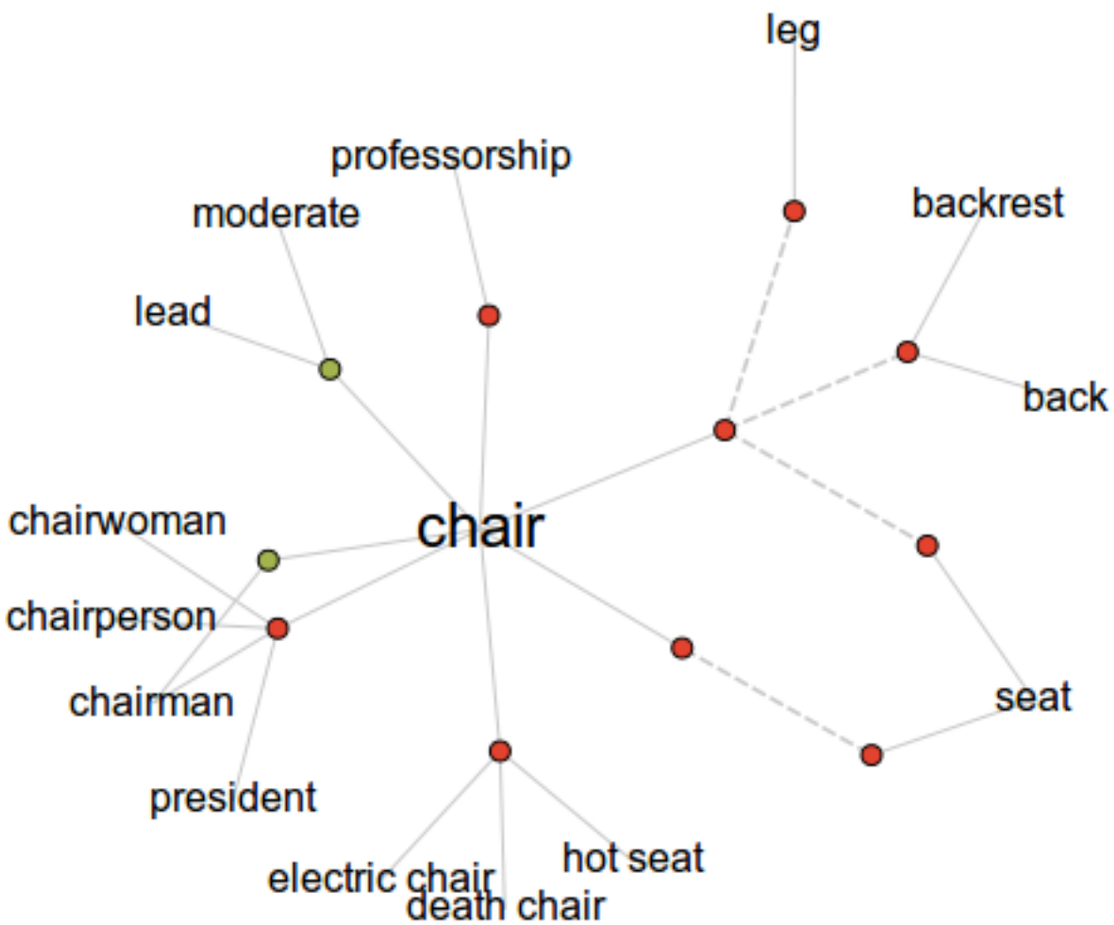
Sentences & Sentiment Analysis

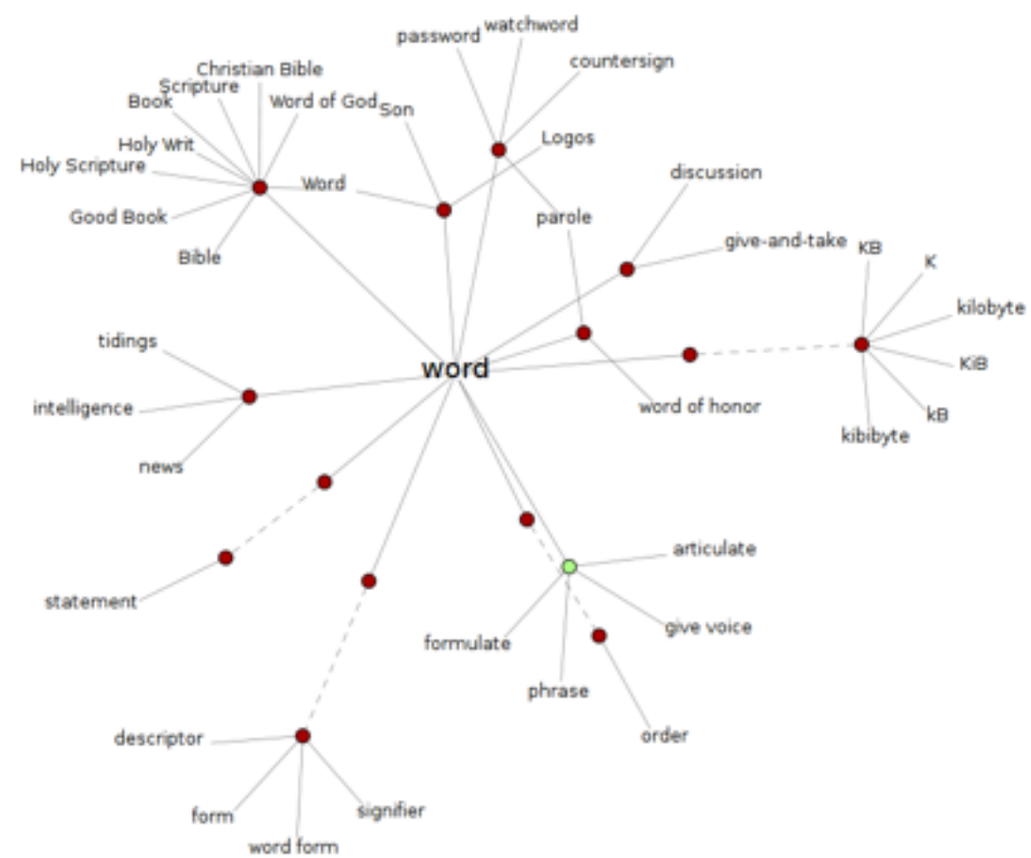
2. ~ | python (python)

```
>>> blob = TextBlob('I think python.org is awesome. What do you think?')
>>> blob.sentences
[Sentence("I think python.org is awesome."), Sentence("What do you think?")]
>>> blob.sentences[0].sentiment
Sentiment(polarity=1.0, subjectivity=1.0)
>>> blob.sentences[1].sentiment
Sentiment(polarity=0.0, subjectivity=0.0)
>>>
>>>
>>>
>>> TextBlob('I hate you').sentiment
Sentiment(polarity=-0.8, subjectivity=0.9)
>>> TextBlob('I love you').sentiment
Sentiment(polarity=0.5, subjectivity=0.6)
>>> TextBlob('Python is a programming language').sentiment
Sentiment(polarity=0.0, subjectivity=0.0)
>>>
```

Wordnet Integration

WordNet is a lexical database for the English language.





Paths

- **S:** (noun) **apple** (fruit with red or yellow or green skin and sweet to tart crisp whitish flesh)
  - *Hyponyms*
  - *Direct hypernyms*
  - *Indirect hypernyms*
    - edible_fruit
      - fruit
        - reproductive_structure
          - plant_organ
            - plant_part
              - natural_object
                - whole
                - object
                - physical_entity
                - entity

Relation Types

- **Is a (hypernym and hyponym)**
  A type-of relationship between two types. For example Organic Process is the hypernym of Evolution.

- **Part of (holonymy and meronymy)**
  A part/whole relationship between two types. For example Natural Selection is the part of Evolution

Interface

2. ~ | python (python)

~ | python (python) ~ | fish (fish)

```
>>> from textblob import Word
>>> computer = Word('computer')
>>> computer.synsets
[Synset('computer.n.01'), Synset('calculator.n.01')]
>>> synset = computer.synsets[0]
>>> synset.hypernym_paths()
[[Synset('entity.n.01'), Synset('physical_entity.n.01'), Synset('object.n.01'), Synset('whole.n.02'), Synset('
artifact.n.01'), Synset('instrumentality.n.03'), Synset('device.n.01'), Synset('machine.n.01'), Synset('comput
er.n.01')]]
>>>
```

Semantic Similarity

2. ~ | python (python)

~ | python (python) | ~ | python (Python) | ~ | redis-server (redis-server) | ~ | fish (fish)

```
>>> apple = Word('apple').synsets[0]
>>> orange = Word('orange').synsets[0]
>>> computer = Word('computer').synsets[0]
>>> apple.path_similarity(apple)
1.0
>>> apple.path_similarity(orange)
0.25
>>> apple.path_similarity(computer)
0.07692307692307693
>>>
```

Text Classification

Text Classification

The task is to assign a document to one or more classes or categories.

Use cases

- Spam filters
- Web page classification
- News and and topic categorization
- Sentiment Analysis

Techniques

- Neural Networks
- K-nearest neighbour algorithms
- Decision Trees
- Naive Bayes Classification

Training

Just count the words under a label.

Politics	
democrat	4
socialism	20
democrat	30
communism	20
politician	10
holacaust	20

Drugs	
smoke	30
weed	22
lsd	34
heroin	54
cannabis	23
marijuana	52

Stem, singularize and eliminate given text

Classification

Marijuana should be legalized nationally in the United States just as it is already in Colorado.

Classification

Calculate scores for each labels

drugs: 4

drugs: 2

Marijuana should be legalized nationally in the United States just as it is already in Colorado.

politics: 3

Drugs = 4 + 2 = 6 / TotalWordCountOfDrugs
Politics = 3 / TotalWordCountOfPolitics

Redis Implementation

Initial labels

initial labels on a Set which is called `labels`

```
SADD labels Politics
SADD labels Drugs
SADD labels Game
SADD labels Programming
```

Querying for training

"Brainfuck! Brainfuck is awesome" as Programming

```
HINCRBY Programming Brainfuck 2

`awesome` and `is` should not be
trained. they are fuzzy words for
training logic.
```

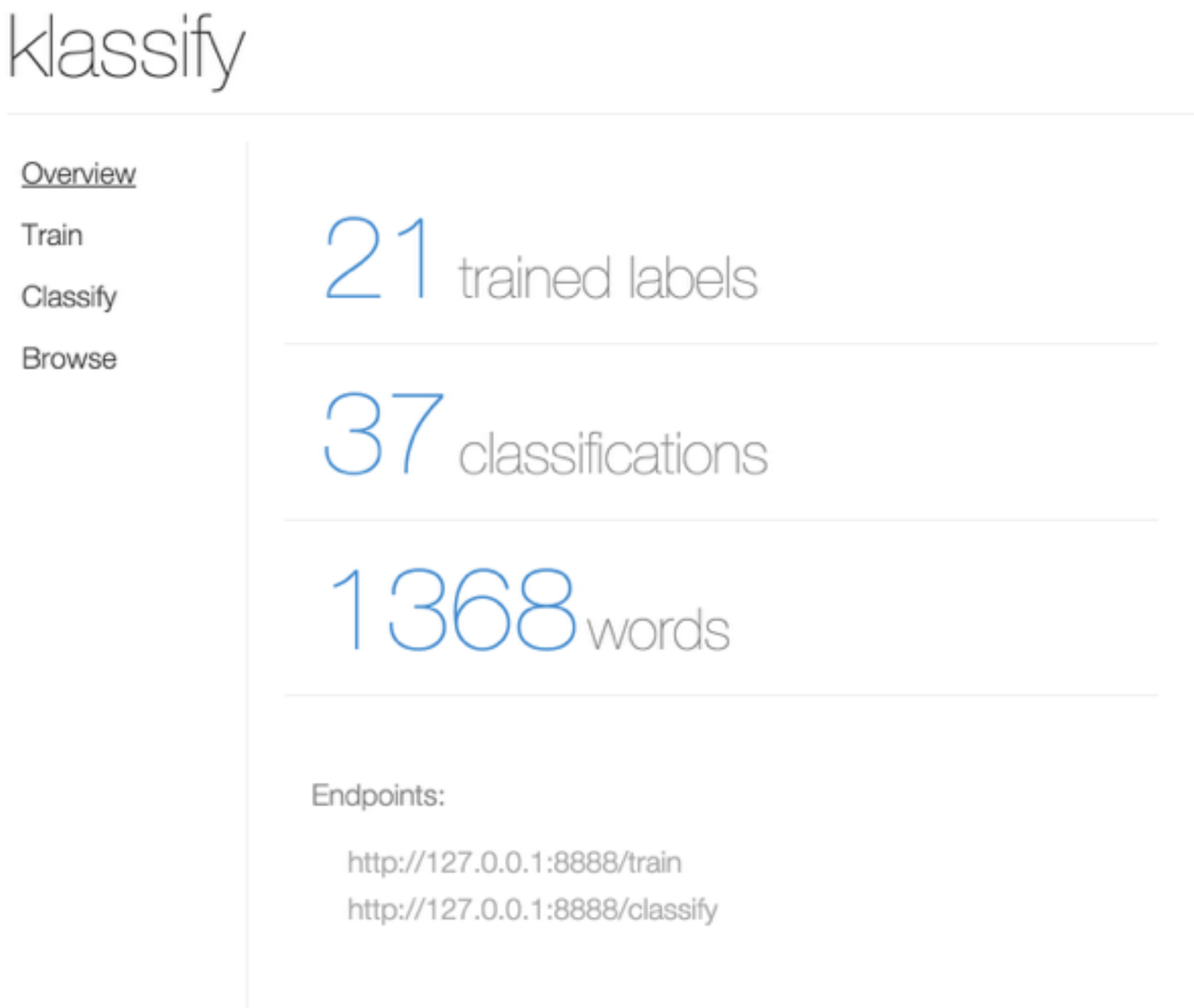
Queries for classification

```
HVALS Programming
# 3 3 2 3 5

HGET Programming Brainfuck
# 5
```

A protoype:

pip install klassify



Training

```
HTTP POST :8888/train

{
  ‘text’: ‘Brainfuck is awesome’,
  ‘label’: ‘Programming’
}
```

Classification

request

```
HTTP POST :8888/classify

{
  'text': 'Brainfuck is awesome'
}
```

response

```
{
    "label": "Programming",
    "scores": {
        "Aliens": 1.428,
        "Animals": 2.380,
        "Society": 6.4935,
        "Technology": 9.523
    }
}
```

Response


```
{
    "label": "Programming",
    "scores": {
        "Aliens": 1.4285714285714287e-05,
        "Animals": 2.380952380952381e-06,
        "Society": 6.493506493506494e-07,
        "Technology": 9.523809523809525e-07
    }
}
```

DEMO