Sentiment Use

Practical 11: Sentiments

Simple Word Eg

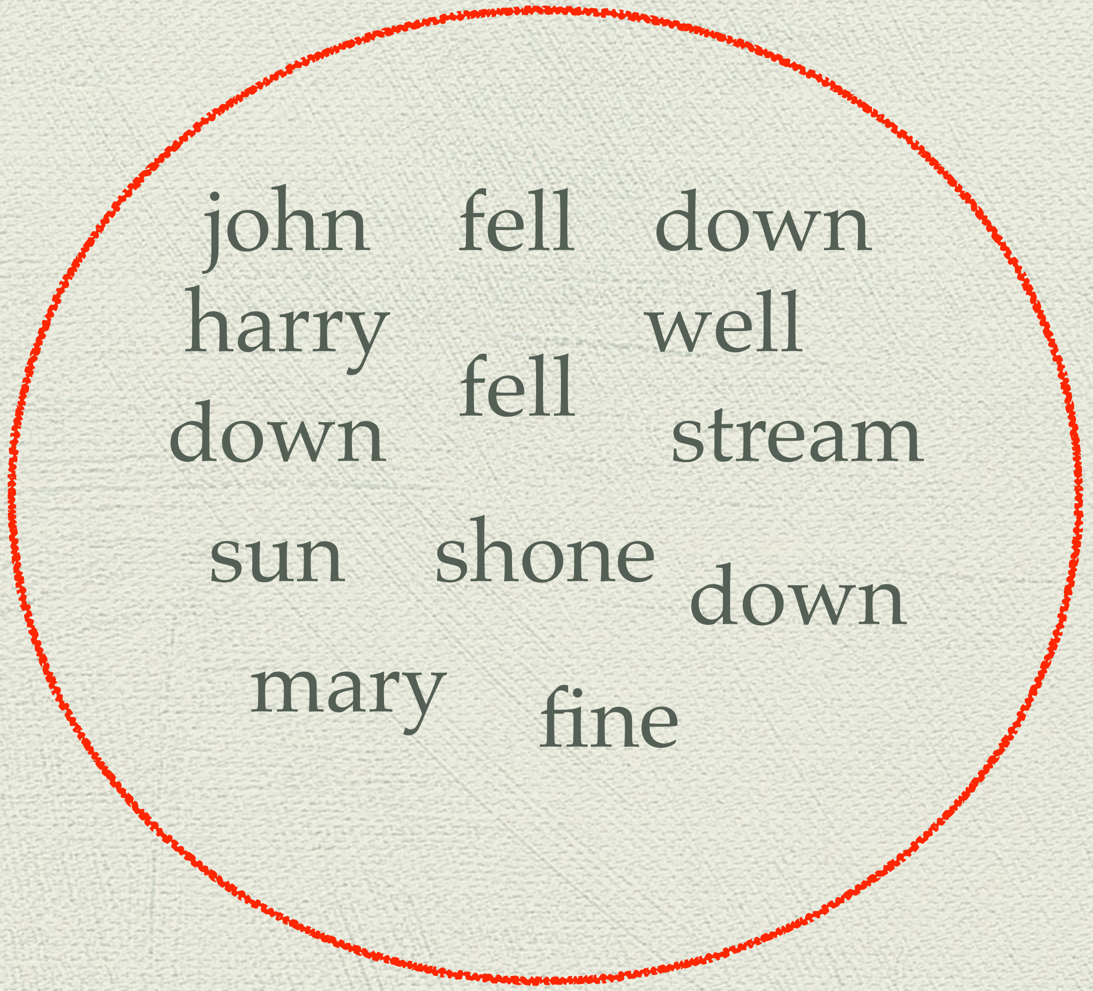
john fell down
harry fell as well
down by the stream
the sun shone before
it went down
mary was fine

{john-1, fell-2,down-3,
harry-1, too-1…}

bill fell down
jeff fell too
down by the river
the sun shone until
it sunk down
belinda was ill

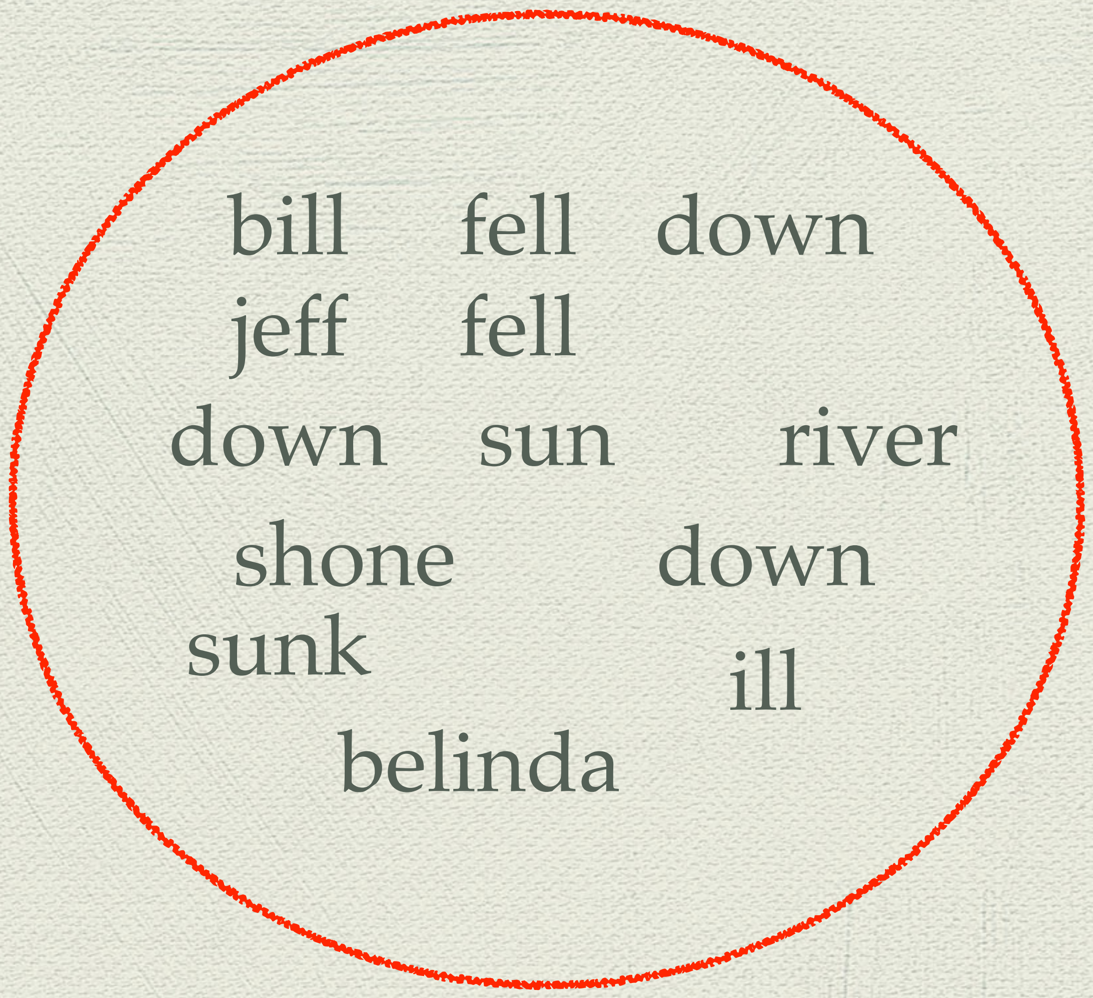
{john-1, fell-2,down-3,
harry-1, too-1…}

Simple Word Eg: Stops-Out



john fell down
harry well
fell
down stream
sun shone down
mary fine

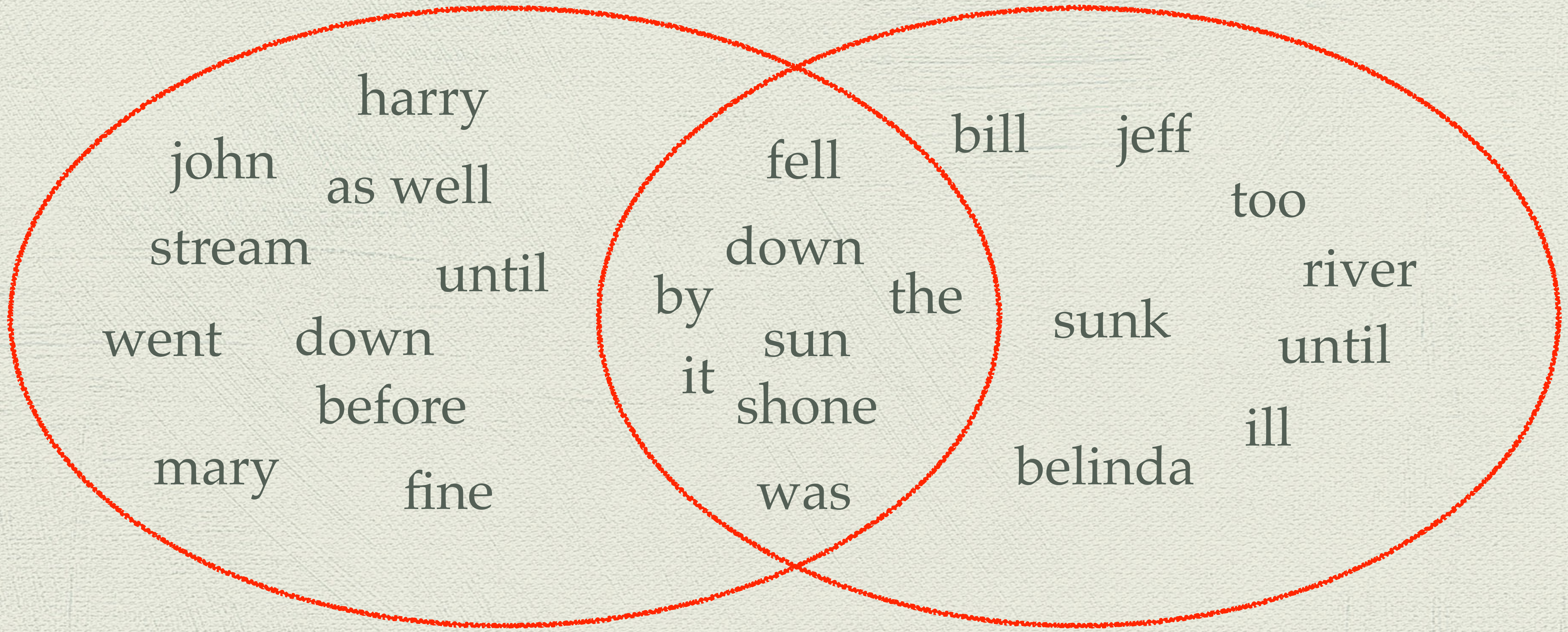
{john-1, fell-2,down-3,
harry-1, too-1…}



bill fell down
jeff fell
down sun river
shone down
sunk
ill
belinda

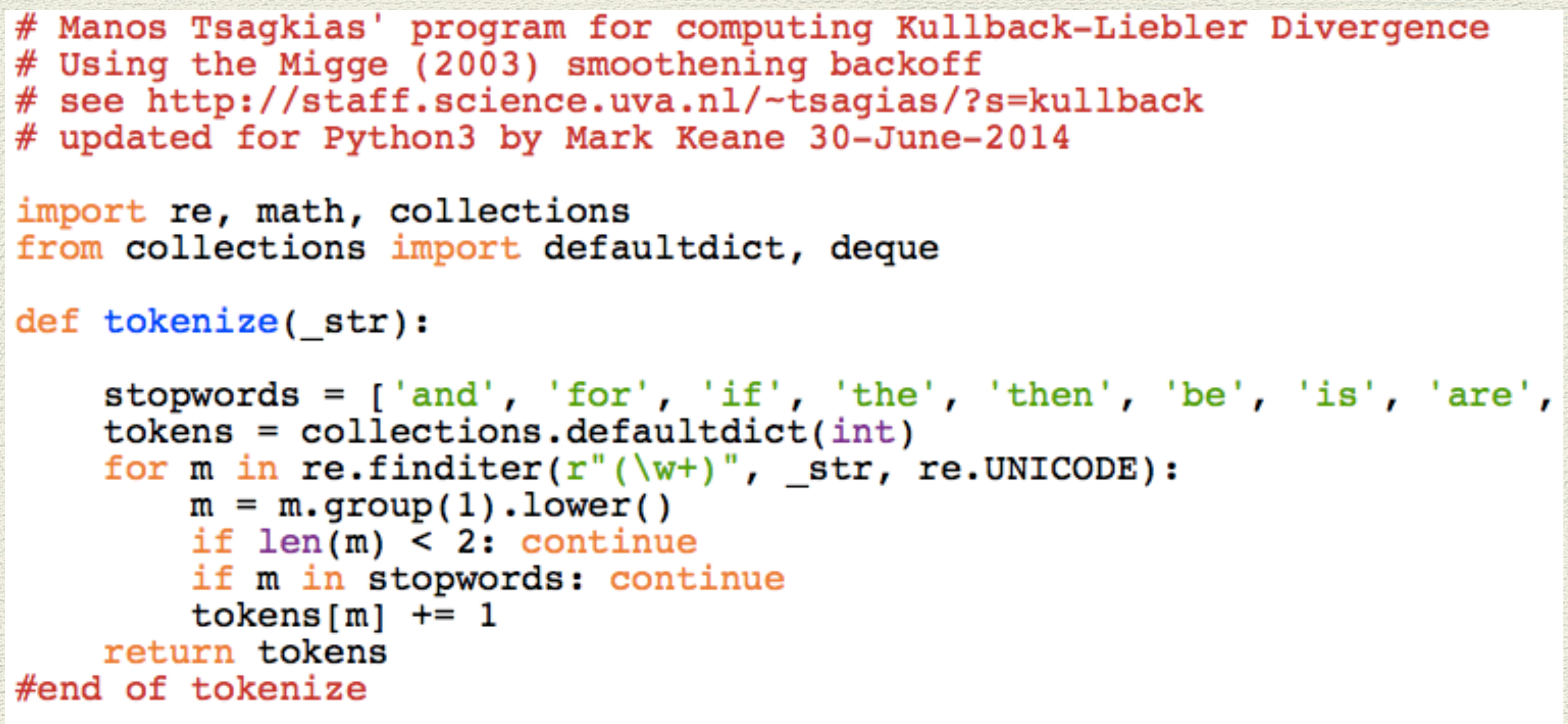
{john-1, fell-2,down-3,
harry-1, too-1…}

Simple Word Eg



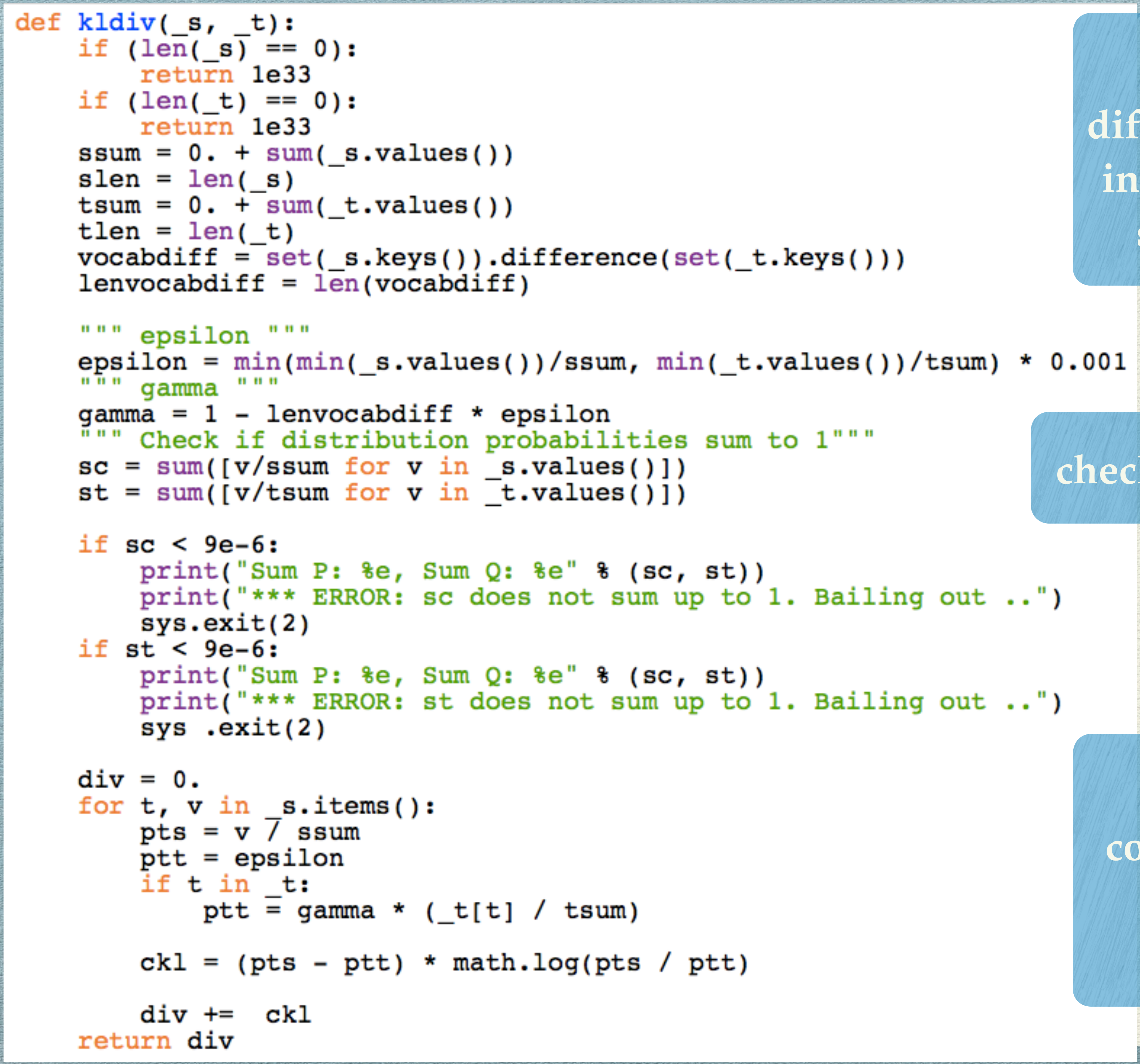
{john-1, fell-2,down-3, harry-1, too-1…}

{john-1, fell-2,down-3, harry-1, too-1…}



```
# Manos Tsagkias' program for computing Kullback-Liebler Divergence
# Using the Migge (2003) smoothening backoff
# see http://staff.science.uva.nl/~tsagias/?s=kullback
# updated for Python3 by Mark Keane 30-June-2014

import re, math, collections
from collections import defaultdict, deque

def tokenize(_str):

    stopwords = ['and', 'for', 'if', 'the', 'then', 'be', 'is', 'are',
    tokens = collections.defaultdict(int)
    for m in re.finditer(r"(\w+)", _str, re.UNICODE):
        m = m.group(1).lower()
        if len(m) < 2: continue
        if m in stopwords: continue
        tokens[m] += 1
    return tokens
#end of tokenize
```

tokenize words from sentences while removing stop words

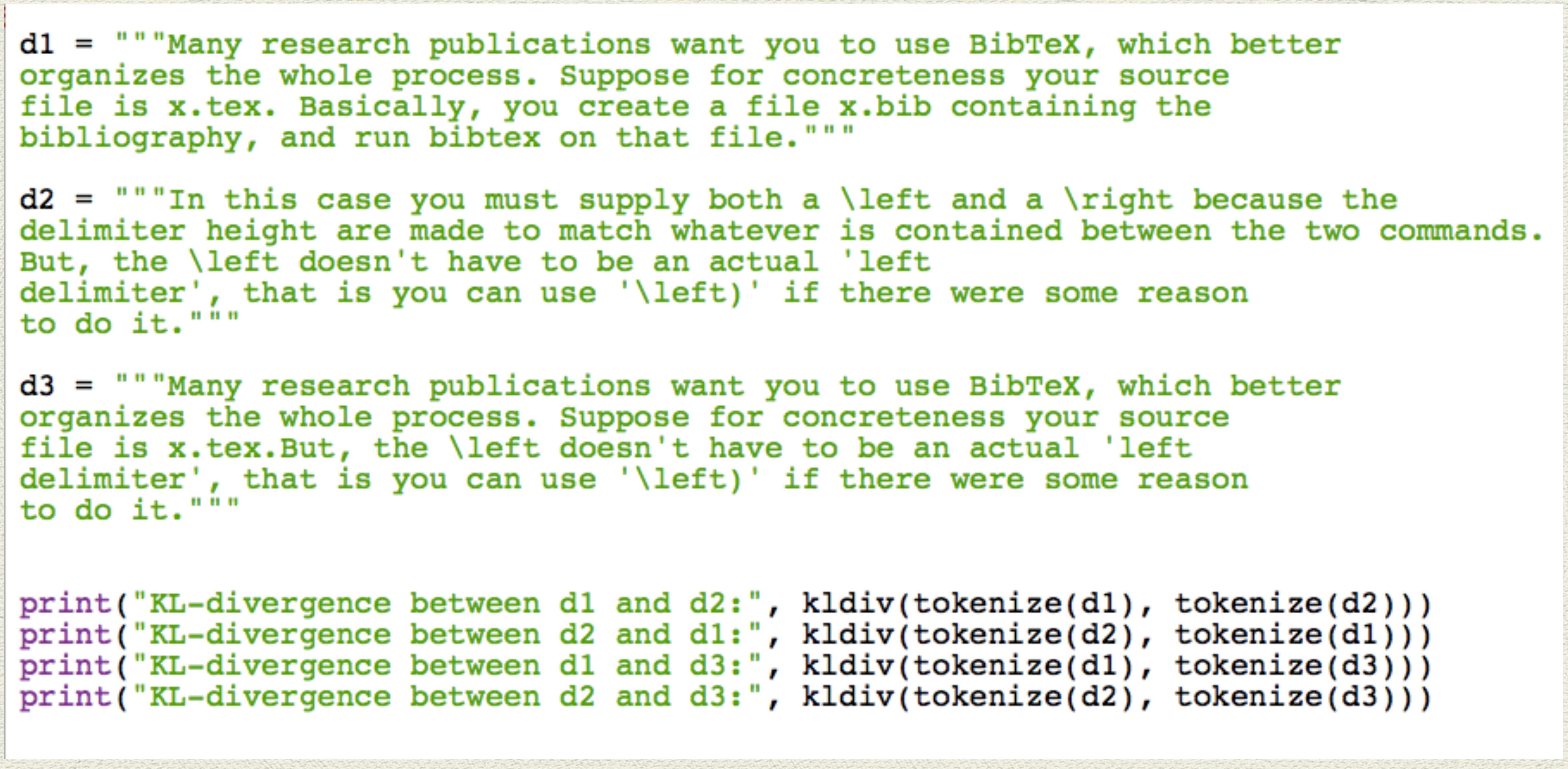


```
def kldiv(_s, _t):
    if (len(_s) == 0):
        return 1e33
    if (len(_t) == 0):
        return 1e33
    ssum = 0. + sum(_s.values())
    slen = len(_s)
    tsum = 0. + sum(_t.values())
    tlen = len(_t)
    vocabdiff = set(_s.keys()).difference(set(_t.keys()))
    lenvocabdiff = len(vocabdiff)

    """ epsilon """
    epsilon = min(min(_s.values())/ssum, min(_t.values())/tsum) * 0.001
    """ gamma """
    gamma = 1 - lenvocabdiff * epsilon
    """ Check if distribution probabilities sum to 1"""
    sc = sum([v/ssum for v in _s.values()])
    st = sum([v/tsum for v in _t.values()])

    if sc < 9e-6:
        print("Sum P: %e, Sum Q: %e" % (sc, st))
        print("*** ERROR: sc does not sum up to 1. Bailing out ..")
        sys.exit(2)
    if st < 9e-6:
        print("Sum P: %e, Sum Q: %e" % (sc, st))
        print("*** ERROR: st does not sum up to 1. Bailing out ..")
        sys .exit(2)

    div = 0.
    for t, v in _s.items():
        pts = v / ssum
        ptt = epsilon
        if t in _t:
            ptt = gamma * (_t[t] / tsum)

        ckl = (pts - ptt) * math.log(pts / ptt)

        div +=  ckl
    return div
```

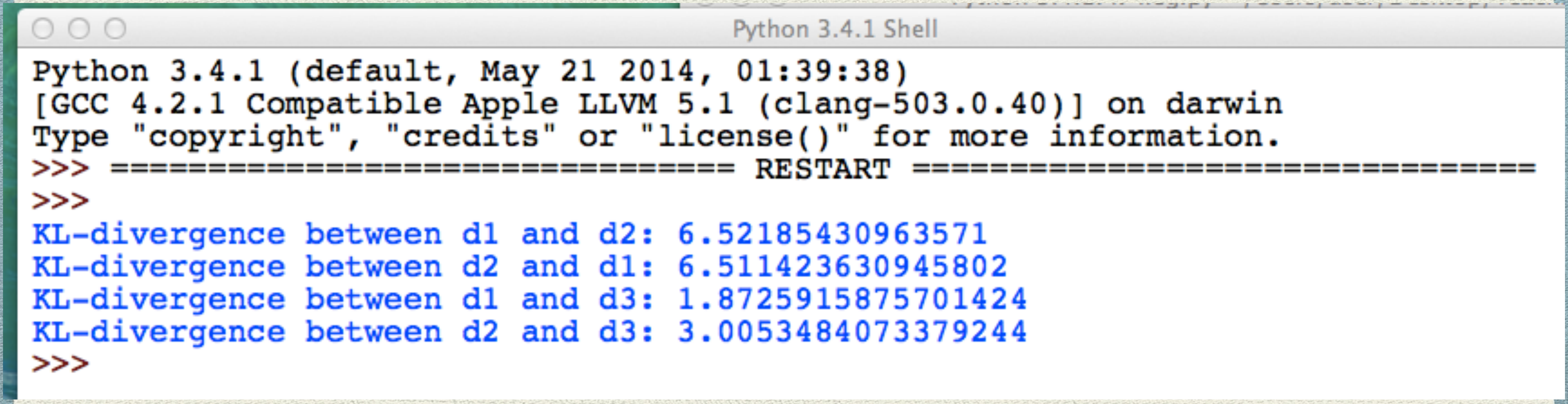
get set of different terms in two sets of sentences

check distribution

compute K-L formula



```
d1 = """Many research publications want you to use BibTeX, which better
organizes the whole process. Suppose for concreteness your source
file is x.tex. Basically, you create a file x.bib containing the
bibliography, and run bibtex on that file."""

d2 = """In this case you must supply both a \left and a \right because the
delimiter height are made to match whatever is contained between the two commands.
But, the \left doesn't have to be an actual 'left
delimiter', that is you can use '\left)' if there were some reason
to do it."""

d3 = """Many research publications want you to use BibTeX, which better
organizes the whole process. Suppose for concreteness your source
file is x.tex.But, the \left doesn't have to be an actual 'left
delimiter', that is you can use '\left)' if there were some reason
to do it."""


print("KL-divergence between d1 and d2:", kldiv(tokenize(d1), tokenize(d2)))
print("KL-divergence between d2 and d1:", kldiv(tokenize(d2), tokenize(d1)))
print("KL-divergence between d1 and d3:", kldiv(tokenize(d1), tokenize(d3)))
print("KL-divergence between d2 and d3:", kldiv(tokenize(d2), tokenize(d3)))
```



```
Python 3.4.1 (default, May 21 2014, 01:39:38)
[GCC 4.2.1 Compatible Apple LLVM 5.1 (clang-503.0.40)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
KL-divergence between d1 and d2: 6.52185430963571
KL-divergence between d2 and d1: 6.511423630945802
KL-divergence between d1 and d3: 1.8725915875701424
KL-divergence between d2 and d3: 3.0053484073379244
>>>
```

Practical

- Take the simple sentences we used in our word example
- Put these into the program and compute the K-L divergence scores for them, in both directions
- Now create a third story that is very different to the other two, add it to the program and report how its score changes relative to the first two.
- Comment on whether the score makes sense.
- Explain what role ***epsilon*** and ***gamma*** play in the computation of K-L