def add\_day(corpus, day, member\_lists):

'''Gets the speeches for a given day'''

date\_string = day.strftime('%Y/%b/%d').lower()

url = 'http://hansard.millbanksystems.com/sittings/{}.js'.format(date\_string)

res = requests.get(url)

try:

obj = json.loads(res.text)

try:

sections = obj[0]['house\_of\_commons\_sitting']['top\_level\_sections']

for section in sections:

try:

sec = section['section']

add\_debate(corpus, 'http://hansard.millbanksystems.com/commons/{}/{}'

.format(date\_string, sec['slug']), day, sec['title'],

member\_lists)

except KeyError:

pass

except KeyError:

pass

except ValueError:

print('No data for {}'.format(date\_string))

def get\_aye\_members\_from\_term(self, term, division\_id):

''' Returns a list of member ids corresponding to members who voted aye

in a given division and spoke in a debate matching a given term '''

self.\_\_curs.execute('''SELECT DISTINCT MEMBER\_ID FROM SPEECH

WHERE DEBATE\_URL IN (SELECT URL FROM DEBATE

WHERE TITLE LIKE ? COLLATE NOCASE)

AND MEMBER\_ID IN (SELECT ID FROM MEMBER INNER JOIN VOTE ON

VOTE.MEMBER\_ID = MEMBER.ID

WHERE VOTE.DIVISION\_ID=? AND

(VOTE.VOTE='AyeVote')) ''',

('%{}%'.format(term), division\_id))

rows = self.\_\_curs.fetchall()

return [row[0] for row in rows]

def get\_paragraph\_text(paragraph):

'''Converts a paragraph tag to plain text'''

paragraph = re.sub(r'<p.\*?>', '', paragraph)

paragraph = re.sub(r'</p.\*?>', '', paragraph)

paragraph = re.sub(r'<a.\*?>.\*?</a>', '', paragraph)

paragraph = re.sub(r'<span.\*?>.\*?</span>', '', paragraph)

paragraph = re.sub(r'<q.\*?>.\*?</q>', '\'', paragraph)

return paragraph

def match\_full\_name(speaker, member\_lists):

''' Returns the member\_id for the given speaker where a match exists '''

if speaker in member\_lists['match\_list']:

mp\_id = member\_lists['match\_list'][speaker]

else:

max\_similarity = 0

no\_titles = remove\_titles(speaker)

for name in member\_lists['name\_list']:

similarity = Levenshtein.ratio(remove\_titles(name[1]), no\_titles)

if similarity > max\_similarity:

max\_similarity = similarity

best\_match = name

if max\_similarity > 0.85:

member\_lists['name\_list'].remove(best\_match)

mp\_id = best\_match[0]

else:

mp\_id = match\_first\_and\_family\_name(no\_titles, member\_lists['name\_list'],

speaker)

member\_lists['match\_list'][speaker] = mp\_id

return mp\_id

<p class='first-para'>

The right hon. Gentleman failed to answer my hon. Friend the Member for South

<a class='permalink column-permalink' id='column\_783'

title='Col. 783 &mdash; HC Deb 18 March 2003 vol 401 c783'

name='column\_783' href='iraq#column\_783' rel='bookmark'>783</a>

Staffordshire (Sir Patrick Cormack). Will he clear up an...

<q>legally, no new resolution is required for the use of force to implement

resolution 687."&#x2014;[<span class="italic">Official Report</span>

, 24 September 2002; Vol. 390, c. 43.]</q>

Lord Goodhart, however, has said...

</p>

def reduce\_features(complete\_train\_features, complete\_test\_features):

''' Performs the same principle component

analysis on given train and test features '''

\_, sigma, v\_transpose = svd(complete\_train\_features, full\_matrices=True,

compute\_uv=True)

rank = compute\_rank(sigma)

truncated\_v = v\_transpose[:rank].transpose()

return matmul(complete\_train\_features, truncated\_v), matmul(complete\_test\_features,

truncated\_v)