Final Project: Information Recommendation System for Students

LINNA LI

nicolelee005@gmail.com

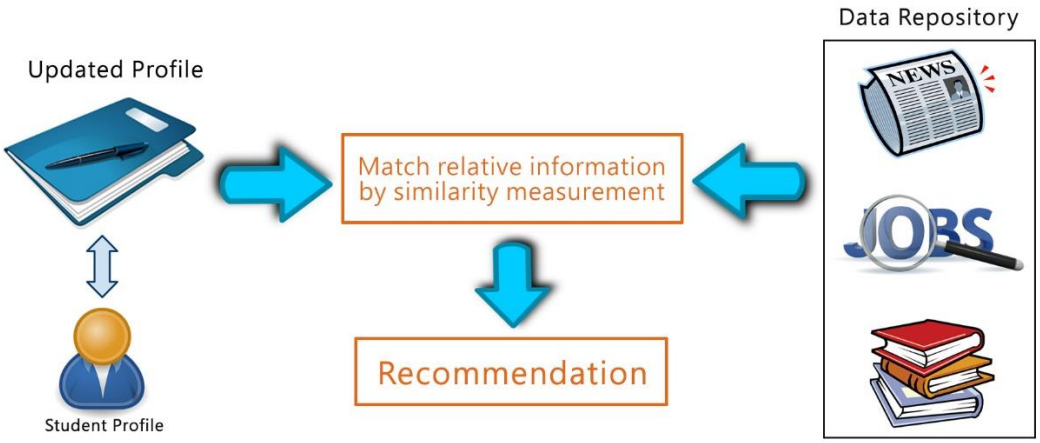
Abstract

In this project, I built a content-based information recommendation system to provide different information for CDM students with their input and stored personal profile as queries. The output information includes the most recent news, jobs and books. This implementation project involves website parsing, information retrieval, text mining and machine learning techniques. The system is built with Python and various modules.

Introduction

The Information Recommendation System means to provide updated information to students and assists them on both academic and career aspects by recommending news, job positions and books related to their majors and career interests. This content-based filtering, also referred to as cognitive filtering, recommends items based on a comparison between the content of the items and a student profile. The user profile is represented by the terms it contains. And the content of each item, like news, jobs or books, will be represented by same set of descriptors or terms with their occurrence in a document.

The general structure of the recommendation system includes a profile input collection, a similarity measurement system and a database. In the initial user interface, the student will be asked to enter his/her personal information, including name, major, courses, career interests and hobbies. With these new inputs, the user database will find and update the student's profile. The other database collects the most recent information, such as news, jobs and books, from real websites. After transforming the information in both database, I designed a system to calculate the similarity between the user's profile and each document in the data repository. In this way, the most relative documents will be retrieve as the recommendation results.



Technique explanation

I. Data repository preparation

To support the recommendation system, an information data repository is needed to build and store the most recent news, job position information and new released books' names from websites. Here are the websites I obtained the data from. Since the target students are mainly from technology field, I used the technology section of The Washington Post and the books only related to computer science and technology.

Item	Website	Website address
News	Reddit	http://www.reddit.com/r/news/
	The Washington Post	http://www.washingtonpost.com/business/technology/
Job	Craigslist	http://chicago.craigslist.org/search/jjj
Book	Barnes&Noble	http://www.barnesandnoble.com/s/computer-science?csrftoken=LPLAuMCh0iKHUoGMhLm5GTDBtQmcuEDp&dref=838&size=90&sort=R

To get the data in the real websites, I applied Beautiful Soup as the website crawler. Beautiful Soup is a Python package for parsing HTML documents. It is able to create a parse tree for parsed pages that can be used to extract data from HTML.

I will explain how to capture the news from Reddit and The Washington Post for example. Here are how the two target websites look like:

As you can see, there are lots of hyperlinks with different functions in both of them. Reddit.com is an entertainment, social networking and news website where registered community members can submit content. And the resources of the news on the websites are various, including all the possible websites that provide news. Thus for Reddit, the HTML code rules can be various as well. In contrary, the news on The Washington Post are only from the website itself, so that the HTML code rules are uniform, which allows me grab data more accurately. So in news part, I defined two functions separately for Reddit and Washington Post with Beautiful Soup.

The first step is to obtain the entire HTML of the main news page where the most recent news is listed. I complete this with Urllib2.urlopen() and BeautifulSoup().

```
html = urllib2.urlopen(Address).read()
soup = BeautifulSoup(html)
```

Then, for Reddit, I grabbed all the code beginning with <a> and then get the hyperlinks with 'herf'. For Washington Post, I conducted this more detailed by seeking the exact code referring to the news title, but not unrelated contents like advertising or other sections.

Reddit:
```
links = soup.findAll("a")
t = set([ link["href"] for link in links if link.has_attr('href')])
```

Washington Post:
```
soup2 = soup1.findAll('h2')
#links = soup1.findAll('a', attrs={'class':''})
links = []
for p in soup2:
    a = p.findAll('a')
    links += a

t = set([ link["href"] for link in links if link.has_attr('href')])
```

Next, I will select and deal with the links I just captured. For Reddit, since almost all the news are from other sites, if a link contains 'www.reddit.com', it means the link is not about news but the website itself. For Washington Post, some content following 'herf' is not the complete website address. I will add the main hostname to them.

Reddit:
```
l = []
for s in t:
    a = s.split('/')
    if ('www.reddit.com' not in a) and ('http:' in a):
        #and a[1]=='2015'
        l.append(s)
return l
```

Washington Post:
```
l = []
for s in t:
    if s[0] == '/':
        s = 'http://www.washingtonpost.com' + s
    l.append(s)
return l
```

At this point, I have finished the function of URL retrieval and able to stored these links.

```
l_web1 = RedditUrls(web1)
```
```
l_web2 = washingtonUrls(web2)
```

With these URLs, I will have a deeper explore to get the news content in each of them. For this getNews() function, I just applied on all the news links from both Reddit and Washington Post. With this function, I will obtain the title and the content. In my experiment, some websites are forbidden to grab information from. To solve this problem, I imported HTTPError and the function will skip these websites automatically.

I used Python dictionary as the container of the data repository. The collected news title is the key; the URL address and the news content will be the values in the two dictionaries respectively.

news_link = {news title : URL}

news_content = {news title : news content}

```
def getNews(articleUrl):
    clean = lambda s: str(re.sub('[\W_]+', ' ', s))

    try:
        html = urllib2.urlopen(articleUrl).read()
        soup = BeautifulSoup(html)

        title = list(set(soup.body.findAll('h1')))
        title_clean = ' '.join(clean(s.text) for s in title).strip()

        article = soup.body.findAll('p')
        article_clean = ' '.join(clean(s.text) for s in article).strip()

        return title_clean, article_clean

    except HTTPError:
        pass
```

```
news_link = {}
news_content = {}

for i in l_web1:
    try:
        title_i, article_i = getNews(i)
        news_link[title_i] = i
        news_content[title_i] = article_i
    except TypeError:
        pass

for j in l_web2:
    try:
        title_j, article_j = getNews(j)
        news_link[title_j] = j
        news_content[title_j] = article_j
    except TypeError:
        pass
```

```
news_link.keys()[15]
```

```
'The State Department s cyber infrastructure is in need of an overhaul'
```

```
news_link.values()[15]
```

```
'http://www.washingtonpost.com/blogs/the-switch/wp/2015/03/13/the-state-departments-cyber-infrastructure-is-in-need-of-an-overhaul/'
```

```
news_content.values()[15]
```

```
'washingtonpost com 1996 2015 The Washington Post  Help and Contact Us Terms of Service Privacy Policy Submissions and Discussion Policy R
SS Terms of Service Ad Choices Hillary Rodham Clinton may have been right to snub the State Department s e mail system  Clinton s use of a
 private e mail account while secretary of state has raised concerns that her communications may not have been secure But the State Depart
ment s cyber infrastructure could use an upgrade of its own in fact many of the agency s links to the online world were down Friday includ
ing parts of the unclassified e mail system  As a part of the Department of State s ongoing effort to ensure the integrity of our unclass
ified networks against cyber attacks the Department is implementing improvements to the security of its main unclassified network during a
 short planned outage of some internet linked systems spokesperson Jen Psaki said in a statement  But it s unclear just how planned the ou
```

The above is the outcome by this step. Similarly, I also got the result for job (job_link and job_content) and book (book_link and book_content). “job_link” contains the position title with the URL, and “job_content” contains the position title with the job description. “book_link” includes book name with the URL, and “book_content” includes book name with the book overview.

To fully prepare for the further calculation, I transformed these news result with stop-words and Porter Stemming System first. Based the online resource (http://facweb.cs.depaul.edu/mobasher/classes/csc575/porter.html), I built a function used to take stemming for the input text as following:

```
def stemming(mytext):
    import httplib, urllib
    params = urllib.urlencode({'input_text': mytext})
    conn = urllib.urlopen('http://www.bmobasher.com/cgi-bin/porter-online-stop.pl?'+params)

    words = []
    for line in conn:
        if line.startswith('<td align=center>'):
            words.append(line[17:-6])

    words = words[2:]
    stem_words = []
    for i in range(len(words)):
        if i%2 == 1:
            stem_words.append(words[i])

    return stem_words
```

In the function to stem information content, I combined title with content first, since the title often include important information. For instant, in the dictionary book_content, it’s possible that the overview only have writer information, but the long book name could outline the basic content of the book.

```
def ContentStem(content): #content is a dictionary --> {title : stemmed title+content}
    title_doc = {}
    for i in range(len(content)):
        key = content.keys()[i]
        view = content.values()[i]
        doc = key +' '+ view
        title_doc[key] = doc
```

Then, for each “content” part, I will put it into the stemming() function and get a list of tokens (content_stem) which can have duplicated items.

```
content_stem = {}
for i in range(len(title_doc)):
    key = title_doc.keys()[i]
    stem = stemming(title_doc.values()[i])
    if len(stem) != 0:
        content_stem[key] = stem

return content_stem
```

```
# {title : stemmed title+content} including duplicated terms
news_stem = ContentStem(news_content)
job_stem = ContentStem(job_content)
book_stem = ContentStem(book_content)
```

So far, I have finish processing the raw data from real websites and building up a data repository of news, jobs and books. The following variables will be used in the further calculation when a student inputs his/her information:

News: news_link, news_content, news_stem

Jobs: job_link, job_content, job_stem

Books: book_link, book_content, book_stem

I also store these results with JSON. If you cannot get this data repository successfully, please load the files in JSON.

In this way, every time I run these functions, the database will be updated with the newest information. However, in this process, I encountered some problems. Every time when I capture the URL links from The Washington Post, I have to run the function for more than 5 times, otherwise it will fail to get any links. But for Reddit, only single time run can get the links. This problem may be caused by the complicated HTML structure of The Washington Post and Python need more time to locate the right place.

II. Student profile storage

In this section, I will explain how the system store and update student profile. The Student Profile database is built up with Python dictionary, and more detailed, it’s a dictionary of dictionaries. The values of each sub-section are strings of stemmed words.

```
student_profile = {'nicole li':{'major':'', 'city':'beijing', 'course':'', 'career':'', 'hobby':'read'},
                   'amy rubin':{'major':'human computer interaction', 'city':'los angeles', 'course':'', 'c
                   'rand paul':{'major':'digit commun media art', 'city':'new york', 'course':'', 'career':
```

The system will collect new information from students by an input interface:

```
print "Your Name:"
name_n = raw_input()
print ""
print "Your Major:"
major_n =  raw_input()
print ""
print "Where are you living:"
city_n = raw_input()
print ""
print "What courses you have taken:"
course_n = raw_input()
print ""
print "What are your career interests:"
career_n = raw_input()
print ""
print "What are your hobbies:"
hobby_n = raw_input()
```

```
Your Name:
Amy Rubin

Your Major:
Human-Computer Interaction

Where are you living:
Chicago

What courses you have taken:
```

Then, this new input values will be added into the existed database according to the student's name (the first step in InfoRec() function).

```
new_pro = [name_n, major_n, city_n, course_n, career_n, hobby_n]
pro_item = ['name','major', 'city', 'course', 'career', 'hobby']

student = student_profile[str(name_n).lower()]

for n in range(1,6):
    record = stemming(student[pro_item[n]]) # record = ['human', 'computer', 'interaction']
    record += stemming(new_pro[n])
    record_u = list(set(record))
    student[pro_item[n]] = ' '.join(record_u)

major = student['major']
city = student['city']
course = student['course']
career = student['career']
hobby = student['hobby']
```

By this step, I'm able to update the Student Profile by adding this new inputs into the database. Taking the stemming on the stored data will allow students to input duplicated info, as well as keep the database neat and efficient. And at the end of this step, I extract related information by name from the Profile database for the further functions.

III. Similarity measurement

1. Prepare student information

   Since this system will provide news, jobs and books based on students' profile, I take different parts of profile information for these three sections. For new, I took all the information of a student. For jobs, I only took his/her career related info. And for books, I assume students want some about both major and interests. Combing those, I used Porter's Stemming to convert them into lists of tokens.

```
def stemInput(major, city, course, career, hobby):
    input_news_raw = major+' '+city+' '+course+' '+career+' '+hobby
    input_job_raw = major+' '+city+' '+course+' '+career+' '+career
    input_book_raw = major+' '+course+' '+hobby

    input_news = re.sub('[\W_]+', ' ', input_news_raw).lower()
    input_job = re.sub('[\W_]+', ' ', input_job_raw).lower()
    input_book = re.sub('[\W_]+', ' ', input_book_raw).lower()

    stem_news_user = stemming(input_news)  # list of terms
    stem_job_user = stemming(input_job)
    stem_book_user = stemming(input_book)

    return stem_news_user, stem_job_user, stem_book_user
```

2. Indexing

So far, the system is ready to generate indexing of terms for documents of news, jobs, books and queries (corresponding student's information).

This indexing() function will take target_stem and stem_target_user, in which "target" can be replaced by "news", "job" and "book". In the first step of the function, I will collect all the tokens appearing in both documents and the query. The outcome "allterm" would be a list of tokens.

```
def indexing(target_stem, stem_target_user):

    # combine terms in all news and query
    allterm_raw = []

    for t in target_stem.values():
        if len(t) != 0:
            allterm_raw.extend(t)
    allterm_raw.extend(stem_target_user)
    allterm = list(set(allterm_raw)) # allterm include all unique terms combining all news
```

Next step is to get the indexing for the query. First, I created a list of 0 whose length equals to the length of allterm. Then, to count the number of tokens in the original query, I created a function named "querycount". For the tokens in "allterm", if the tokens also appears in query, I will change 0 to the count by the token's index.

```
# get index of the query
q_target_index = [0]*len(allterm)
freq_target_user = querycount(stem_target_user)

for z in freq_target_user.keys():
    for p,q in enumerate(allterm):
        if q == z:
            q_target_index[p] = freq_target_user[q]
```

```
def querycount(stem_target_user):
    unique_list = list(set(stem_target_user))
    freq_target_user = {}
    for term in unique_list:
        freq_target_user[term] = stem_target_user.count(term)
    return freq_target_user
```

By the same way, I also get the indexing for the documents as following. And this function will return the indexing of query and documents.

```
# get index of each document
target_index = {}
for x in target_stem.keys():  # x is news title

    doc_index = [0]*len(allterm)  # array for each document

    for ti in range(len(allterm)): # ti is the index of terms in allterm

        term = allterm[ti]

        if term in target_stem[x]:
            count = target_stem[x].count(term)
            doc_index[ti] = count

    target_index[x] = doc_index

return q_target_index, target_index
```

3. Convert data with “tfidf”
   With the indexing, I will convert them with tfxidf weights.

```
def tfidf(q_target_index, target_index):

    doc_matrix = np.array(target_index.values())
    querydata = np.array(q_target_index)

    N = float(len(doc_matrix)) # number of documents
    newlist = []

    for i in range(len(doc_matrix.T)):
        nk = 0
        for freq in doc_matrix.T[i]:
            if freq != 0:
                nk += 1

        if nk != 0:
            idf = math.log(N/nk, 2)
            newlist.append(doc_matrix.T[i]*idf)
            querydata[i] = querydata[i]*idf
        else:
            newlist.append(doc_matrix.T[i])
            querydata[i] = 0

    tfidf_doc = array(newlist).T

    return tfidf_doc, querydata
```

4. Cosine similarity
   In this step, I will calculate the Cosine Similarity between the query with each document in the section, such as news section, job section and book section.

```
def ranking_result(tfidf_doc, querydata, target_link): # news_matrix_tfidf, news_user_tfidf

    data = tfidf_doc
    Q = querydata

    from numpy import linalg as la
    norm_Q = la.norm(Q)

    cosine = []
    for i in data:
        norm_rank = la.norm(i)
        sim = np.dot(Q,i)/(norm_Q*norm_rank)
        cosine.append(sim)
```

And I will also rank the result by their similarity values, and the function will return the top ranking results for the student, including news/job/book name, their similarity values and URL to the page.

```
    t = []
    for x, y in enumerate(cosine):
        name = target_link.keys()[x]
        t.append([y, name])
    ranking = np.array(sorted(t, key=lambda tu: tu[0], reverse=True))[0:5]

    recommendation = {}
    for ind, nm in enumerate(ranking[:,1]):
        recommendation[nm] = [ranking[ind,0],target_link[nm]]

    return recommendation
```

For all the four steps described above, I combined these steps into one functions as below. I highlighted the functions I explained. This function is able to return the recommendation from all the three sections.

```
def InfoRec(name_n, major_n, city_n, course_n, career_n, hobby_n, news_stem, job_stem, book_stem, news_link, job_link, book_link):

    new_pro = [name_n, major_n, city_n, course_n, career_n, hobby_n]
    pro_item = ['name','major', 'city', 'course', 'career', 'hobby']

    student = student_profile[str(name_n).lower()]

    for n in range(1,6):
        record = stemming(student[pro_item[n]]) # record = ['human', 'computer', 'interaction']
        record += stemming(new_pro[n])
        record_u = list(set(record))
        student[pro_item[n]] = ' '.join(record_u)

    major = student['major']
    city = student['city']
    course = student['course']
    career = student['career']
    hobby = student['hobby']

    stem_news_user, stem_job_user, stem_book_user = stemInput(major, city, course, career, hobby)

    q_news_index, news_index = indexing(news_stem, stem_news_user)
    q_job_index, job_index = indexing(job_stem, stem_job_user)
    q_book_index, book_index = indexing(book_stem, stem_book_user)

    news_matrix_tfidf, news_user_tfidf = tfidf(q_news_index, news_index)
    job_matrix_tfidf, job_user_tfidf = tfidf(q_job_index, job_index)
    book_matrix_tfidf, book_user_tfidf = tfidf(q_book_index, book_index)

    news_ranking = ranking_result(news_matrix_tfidf, news_user_tfidf, news_link)
    job_ranking = ranking_result(job_matrix_tfidf, job_user_tfidf, job_link)
    book_ranking = ranking_result(book_matrix_tfidf, book_user_tfidf, book_link)

    return news_ranking, job_ranking, book_ranking
```

IV. Example recommendations

In this part, I would like to show three examples.

Student 1:

Name	Nicole Li
Major	Predictive Analytics / Data analysis and data mining
City	Chicago
Courses	Computer Science, Data Analysis and Regression, Knowledge discovery technology, Programming Data Mining Apps, Intelligent Information Retrieval, Advanced Data Analysis
Career interests	data analysis or database management. I want a more technical position related to data in Twitter or Facebook or any other social media companies. Consulting companies are also ideal targets.
Hobbies	reading books, sports, Japanese, designing, new technology, cooking

Before these input, her profile was like this:

```
'nicole li':{'major':'', 'city':'beijing', 'course':'', 'career':'', 'hobby':'read'}
```

Getting these new information inputs, the system is run. After that, her profile changes into this:

```
student_profile['nicole li']
```

```
{'career': 'analysi media target databas twitter relat ideal technic consult facebook social posit compani data manag more',
 'city': 'beij chicago',
 'course': 'analysi comput advanc intellig app knowledg technolog mine discoveri inform program regress retriev data scienc',
 'hobby': 'japanes read technolog desig book cook new photographi sport',
 'major': 'analysi predict mine / analyt data'}
```

Here are the news recommended to Nicole Li (who is actually me), which includes some very interesting articles to me.

```
news_rec_nicole
```

```
{'5 sneaky ways to harness clean energy': ['0.0547216853796',
  'http://www.washingtonpost.com/blogs/innovations/wp/2015/03/18/5-sneaky-ways-to-harness-clean-energy/'],
 'Elon Musk Human driven cars may be outlawed because they re too dangerous': ['0.106606866021',
  'http://www.washingtonpost.com/blogs/the-switch/wp/2015/03/18/elon-musk-human-driven-cars-may-be-outlawed-because-theyre-too-dangerous/'
],
 'Law Disorder Civilization Discontents  13 inch Broadwell MacBook showdown Should you go Pro or get an Air  Atari to indie dev Stop rippi
ng off your own work on Tempest 2000 Updated  New York county sheriff must give up stingray records judge orders ArsTechnica Hands on with
 the new Retina MacBook TAG Heuer Intel Google announce smartwatch that feels like a normal watch  Former Facebook employee retains Ellen
Pao s lawyer in new discrimination case NASA s MAVEN spots auroras dust at high altitudes above Mars Gaye family asks judge to ban all sal
es performances of Blurred Lines  Congressman to FCC Open Internet rules jeopardize the open Internet': ['0.100447576511',
  'http://arstechnica.com/tech-policy/2015/03/new-york-county-sheriff-must-give-up-stingray-records-judge-orders/'],
 'Target hiking minimum wage to 9 hour in April Report': ['0.0848714246446',
  'http://www.cnbc.com/id/102508233'],
 'r news': ['0.0677201446028', 'http://goo.gl/gBldE?ri']}
```

Here are the jobs for Nicole Li. There is no position related to data analyst, but there is a Photographer job in it, which can match to Nicole's interests in photography.

```
job_rec_nicole
```

```
{'Apartment Maintenance Office Personnel Needed Part Time Available Aurora Naperville Geneva': ['0.107978899454',
  'http://chicago.craigslist.org/wcl/rej/4936272041.html'],
 'FULL TIME UX Designer CREATIVE CIRCLE': ['0.145597086897',
  'http://chicago.craigslist.org/nwc/web/4936318740.html'],
 'Now hiring No experience necessary': ['0.191393039996',
  'http://chicago.craigslist.org/nwc/tfr/4920355251.html'],
 'WANTED Photographers for Retail Position STUDENTS WELCOME Chicago': ['0.200962673944',
  'http://chicago.craigslist.org/nch/trd/4928150807.html'],
 'Wedding Videographer Wanted Chicagoland': ['0.24208754878',
  'http://chicago.craigslist.org/nwc/med/4925507909.html']}
```

Here are the books for Nicole, most of which are related to data analysis and Nicole's major.

```
book_rec_nicole
```

```
{'Applied Multiway Data Analysis Edition 1': ['0.126543167488',
  'http://www.barnesandnoble.com/w/applied-multiway-data-analysis-pieter-m-kroonenberg/1101194425?ean=9780470164976&itm=23&usri=data+analysis'],
 'Bayesian Data Analysis Third Edition Edition 3': ['0.121166955653',
  'http://www.barnesandnoble.com/w/bayesian-data-analysis-third-edition-andrew-gelman/1118725091?ean=9781439840955&itm=5&usri=data+analysis'],
 'Contemporary Theatre Film and Television A Biographical Guide Featuring Performers Directors Writers Producers Designers Managers Choreographers Technicians Composers Executives Dancers and Critics in the United States Canada Great Britain': ['0.118092734601',
  'http://www.barnesandnoble.com/w/contemporary-theatre-film-and-television-gale/1119941822?ean=9780787690403&itm=29&usri=film+produce'],
 'Developing Web Pages with jQuery': ['0.192746356666',
  'http://www.barnesandnoble.com/w/developing-web-pages-with-jquery-don-gosselin/1105175295?ean=9781435460799&itm=106&usri=web+design'],
 'The Independent Film Producer s Survival Guide A Business and Legal Sourcebook Edition 2': ['0.189641149119',
  'http://www.barnesandnoble.com/w/independent-film-producers-survival-guide-j-gunnar-erickson/1117853163?ean=9780825673184&itm=4&usri=film+produce']}
```

Student 2:

Name	Amy Rubin
Major	Human-Computer Interaction
City	Chicago
Courses	Interaction Design and Information Architecture, Digital Design, Topics in Human-Computer Interaction, Information Visualization and Infographics, Intranets and Portals, User Experience Design Practicum
Career interests	Web designer, Web graphic designer, Multimedia Web designers, The UI designer in Internet service providers (ISPs) and Internet consulting firms, or traditional advertising, marketing, and PR companies.
Hobbies	golfing, reading novels, and watching movies, cooking

Recommendation results:

```
student_profile['amy rubin']
```

```
{'career': 'firm web graphic tradit isp provid servic multimedia pr consult ui internet design advertis compani market',
 'city': 'lo angel chicago',
 'course': 'digit infograph interact portal humancomput topic inform visual design user intranet experi practicum architectur',
 'hobby': 'novel golf movi read technolog watch cook new',
 'major': 'interact comput humancomput human'}
```

```
news_rec_amy
```

```
{'Aramark worker ordered prisoner to feed inmates at Michigan prison cake partially eaten by rodents Get the latest Lansing updates': ['0.
0584647912712',
  'http://www.mlive.com/lansing-news/index.ssf/2015/03/inmates_at_mid-michigan_prison.html'],
 'Elon Musk Human driven cars may be outlawed because they re too dangerous': ['0.112947479837',
  'http://www.washingtonpost.com/blogs/the-switch/wp/2015/03/18/elon-musk-human-driven-cars-may-be-outlawed-because-theyre-too-dangerous/'
],
 'Love in the time of bots': ['0.111022696695',
  'http://www.washingtonpost.com/blogs/innovations/wp/2015/03/17/love-in-the-time-of-bots/'],
 'Obama Maybe it s time for mandatory voting': ['0.046134652853',
  'http://www.cnn.com/2015/03/19/politics/obama-mandatory-voting/'],
 'Texas Bar Charges Willingham Prosecutor with Misconduct': ['0.0362745342716',
  'http://www.pbs.org/wgbh/pages/frontline/criminal-justice/death-by-fire/texas-bar-charges-willingham-prosecutor-with-misconduct/']}
```

```
job_rec_amy
```

```
{'ARCLIGHT CINEMAS NOW HIRING GLENVIEW Glenview': ['0.132162963186',
  'http://chicago.craigslist.org/nwc/tfr/4930839908.html'],
 'CASTING NOW JUST FRIENDS who might want to take read more Nationwide': ['0.121838990346',
  'http://chicago.craigslist.org/nwi/tfr/4934751823.html'],
 'Extraordinary Personal Assistant of superior character and integrity West Palm Beach': ['0.10308376851',
  'http://chicago.craigslist.org/chc/etc/4938067614.html'],
 'Interior Designer Burr Ridge': ['0.105390220629',
  'http://chicago.craigslist.org/sox/egr/4938218631.html'],
 'WANTED Photographers for Retail Position STUDENTS WELCOME Chicago': ['0.107480047981',
  'http://chicago.craigslist.org/nch/trd/4928150807.html']}
```

```
book_rec_amy
```

```
{'Data Analysis for Database Design Edition 3': ['0.106480379941',
  'http://www.barnesandnoble.com/w/data-analysis-for-database-design-david-howe/1100665402?ean=9780750650861&itm=93&usri=data+analysis'],
 'Designing Effective Web Surveys': ['0.0745647586085',
  'http://www.barnesandnoble.com/w/designing-effective-web-surveys-mick-p-couper/1100295300?ean=9780521889452&itm=92&usri=web+design'],
 'Exploring the Art and Technology of Web Design Edition 1': ['0.137711780253',
  'http://www.barnesandnoble.com/w/exploring-the-art-and-technology-of-web-design-ruth-ann-anderson/1100963717?ean=9781401871079&itm=27&us
ri=web+design'],
 'Getting Started with WordPress Design Your Own Blog or Website Edition 1': ['0.0709826339555',
  'http://www.barnesandnoble.com/w/getting-started-with-wordpress-todd-kelsey/1102078147?ean=9781435460065&itm=122&usri=web+design'],
 'Web Design Concepts and Best Practices Edition 1': ['0.0936704175687',
  'http://www.barnesandnoble.com/w/web-design-carolee-cameron/1101448082?ean=9780763816544&itm=105&usri=web+design']}
```

Student 3:

Name	Rand Paul
Major	Film / Cinema Production
City	New York, Boston
Courses	NEW DIRECTONS IN FILM AND PHILOSOPHY, INTRO TO CINEMA AND NEW MEDIA, SOUND & IMAGE THEORY, WRITING FILM CRITICISM, TOPICS IN WORLD CINEMA, TOPICS IN AMERICAN CINEMA, AMERICAN FILM HISTORY 1960-1990, ANALYSIS OF FILM LANGUAGE
Career interests	The associate producer for film and TV works closely with a producer to develop appropriate film and TV programming and formats; however, they may be required to work on more specific elements of a production than the producer. The position includes a wide range of responsibilities, including writing and creating content, communicating between supervisors and subordinates, coordinating between multiple departments, editing, and directing.
Hobbies	sports, snowboarding, video games, hiking

Recommendation results:

```
student_profile['rand paul']
```

```
{'career': 'rang supervisor creat direct respons close film content depart develop commun tv write program includ more product format coor
din multipl associ requir wide appropri specif edit work element posit produc subordin',
 'city': 'new york boston',
 'course': 'sound topic 19601990 cinema media imag directon american histori write critic intro analysi theori world new languag philosoph
i film',
 'hobby': 'movi woodwork hike game snowboard video photographi sport',
 'major': 'digit art commun media cinema product / film'}
```

```
news_rec_rand
```

```
{'Administration sets record for withholding government files': ['0.0547846441294',
  'http://news.yahoo.com/us-sets-record-denying-censoring-government-files-071519617--politics.html'],
 'Aramark worker ordered prisoner to feed inmates at Michigan prison cake partially eaten by rodents Get the latest Lansing updates': ['0.
0632815311388',
  'http://www.mlive.com/lansing-news/index.ssf/2015/03/inmates_at_mid-michigan_prison.html'],
 'GOP The FCC s inspector general has launched an investigation into net neutrality': ['0.0578450879543',
  'http://www.washingtonpost.com/blogs/the-switch/wp/2015/03/17/gop-the-fccs-inspector-general-has-launched-an-investigation-into-net-neut
rality/'],
 'Law Disorder Civilization Discontents  13 inch Broadwell MacBook showdown Should you go Pro or get an Air  Atari to indie dev Stop rippi
ng off your own work on Tempest 2000 Updated  New York county sheriff must give up stingray records judge orders ArsTechnica Hands on with
 the new Retina MacBook TAG Heuer Intel Google announce smartwatch that feels like a normal watch  Former Facebook employee retains Ellen
Pao s lawyer in new discrimination case NASA s MAVEN spots auroras dust at high altitudes above Mars Gaye family asks judge to ban all sal
es performances of Blurred Lines  Congressman to FCC Open Internet rules jeopardize the open Internet': ['0.110701065942',
  'http://arstechnica.com/tech-policy/2015/03/new-york-county-sheriff-must-give-up-stingray-records-judge-orders/'],
 'Love in the time of bots': ['0.0656448502559',
  'http://www.washingtonpost.com/blogs/innovations/wp/2015/03/17/love-in-the-time-of-bots/']}
```

```
job_rec_rand
```

```
{'Full Time Front End Developer Agency North Burbs CREATIVE CIRCLE': ['0.158712358936',
  'http://chicago.craigslist.org/nch/web/4936811803.html'],
 'Graphic Designer Automotive Industry Oakbrook Downers Grove': ['0.208005323135',
  'http://chicago.craigslist.org/wcl/web/4938202855.html'],
 'Immediate Need for Homecare Caregivers for Seniors 10 hr Norridge Edgewater North Park': ['0.147881109515',
  'http://chicago.craigslist.org/chc/hea/4936861923.html'],
 'User Interaction Designer eCommerce Web Standards CREATIVE CIRCLE': ['0.158999889052',
  'http://chicago.craigslist.org/chc/med/4936465330.html'],
 'Wedding Videographer Wanted Chicagoland': ['0.101406017332',
  'http://chicago.craigslist.org/nwc/med/4925507909.html']}
```

```
book_rec_rand
```

```
{'Goodbye Cinema Hello Cinephilia Film Culture in Transition': ['0.148588825348',
  'http://www.barnesandnoble.com/w/goodbye-cinema-hello-cinephilia-jonathan-rosenbaum/1101613807?ean=9780226726649&itm=132&usri=film+produ
ce'],
 'Indie Film Producing The Craft of Low Budget Filmmaking': ['0.149169608995',
  'http://www.barnesandnoble.com/w/indie-film-producing-suzanne-lyons/1111091389?ean=9780240817637&itm=1&usri=film+produce'],
 'Media Law for Producers Edition 3': ['0.197196559641',
  'http://www.barnesandnoble.com/w/media-law-for-producers-philip-miller/1119374310?ean=9780240803036&itm=34&usri=film+produce'],
 'Media Law for Producers Edition 4': ['0.2034019209',
  'http://www.barnesandnoble.com/w/media-law-for-producers-philip-miller/1100697760?ean=9780240804781&itm=11&usri=film+produce'],
 'Statistical Techniques for Data Analysis Edition 2': ['0.248284246222',
  'http://www.barnesandnoble.com/w/statistical-techniques-for-data-analysis-john-k-taylor/1101530184?ean=9781584883852&itm=102&usri=data+a
nalysis']}
```

V. Future improvement

From the result, we can see that the system runs well. But there are still some problems with it. First of all, the data base is not big enough and this can cause inaccuracy of the result. Thus, I will keep collecting data for both information and student profile database. Additionally, the current user interface is using Python, which is hard to control by user. It would be better if a webpage form can be created for data input, as well as a page to display the recommendations.