Let’s Meet!

by : Ryan Gin
GA Santa Monica
Data Science Immersive 2016

Problem Statement

Love and Dating can be hard. And as most of us would probably agree finding the right person can be a frustrating, often discouraging task as you go from person to person trying to find “the one”. In addition, people are more and more preoccupied with their careers and other responsibilities, and having a vibrant and active social life often requires an extensive amount of time and energy that many people, especially working professionals, simply do not have. Because of these reasons and many others, people are turning to online dating as a solution to resolve these issues.

Top 5 Online Dating Websites

List of Dating Sites and Unique Monthly Users

1) Match - 35,000,000
2) Plenty of Fish - 23,000,000
3) Zoosk - 11,500,000
4) Ok Cupid - 10,150,000
5) eHarmony - 7,100,000

Source: http://www.ebizmba.com/articles/dating-websites

Solution

My hope is to create an algorithm that can best match or pair any new user with their perfect match in a given dataset based on their similarity using Natural Language Processing.

Definitions

- **Natural language processing** (**NLP**) is a field of computer science, artificial intelligence, and computational linguistics concerned with the interactions between computers and human (**natural**) languages. As such, **NLP** is related to the area of human–computer interaction.
- **Latent Dirichlet allocation** (**LDA**) is a generative statistical model that allows sets of observations to be explained by unobserved groups that explain why some parts of the data are similar.
- **Cosine similarity** is a measure of **similarity** between two non zero vectors of an inner product space that measures the **cosine** of the angle between them.

Goals

1. **Filtering Function:** Using a filtering function, the user can filter out existing users in the database to pick candidates based on their preferences and indicate a features importance; any existing users who do not fit the criteria are filtered out
2. **Natural Language Processing:** Create topics based on words in user’s essay prompts
3. **Recommend by Similarity:** Using available data, determine a “Similarity Score” between any new incoming user and remaining users in the database
4. **Success Criteria:** The success of the algorithm will be how interested a new user is in their matches

Dataset

Dataset - OK Cupid Users

- Anonymous User information from a dataset of 59,946 OK Cupid Users
- Users within a 25 mile radius of San Francisco
- All users online from 06/30/2011-06/30/2012
- The data was scraped from public profiles on www.okcupid.com on 06/30/2012
- Permission to use this data was obtained from OkCupid president and co-founder Christian Rudder under the condition that the dataset remains public

Source: https://github.com/rudeboybert/JSE_OkCupid

Dataset (Features)

Profile User Information

- Age (mean: 32)
- Ethnicity (white: 31760; multi: 6652; asian: 5829; ???: 5228; hispanic / latin: 2652; black: 1893; other: 1641; indian: 1039; pacific islander: 398; middle eastern: 303; native american: 61)
- Height (in inches): 60 - 80
- Location (san francisco, california: 31064; oakland, california : 7214; berkeley, california : 4212; san mateo, california: 1331; palo alto, california: 1064)
- last online
- sex (gender) (m: 35829, f: 24117)
- Status (single: 55697; seeing someone: 2064; available: 1865; married: 310; unknown: 10)

Dataset (Features)

Lifestyle Variables

- Diet (no restrictions: 27881, restrictions: 5880, other: 1790)
- Drinking (socially: 41780, rarely: 5957, often: 5164, not at all: 3267, very often: 471, desperately: 322)
- Smokes (no: 43896; sometimes: 3787; when drinking: 3040; yes: 2231; trying to quit: 1480)
- Drugs (never: 37724, sometimes: 7732, often: 410)
- Body type (average: 14652, fit: 12711, athletic: 11819)
- Offspring (???: 35561, doesn't have kids: 7560, doesn't have kids, but might want them: 3875, doesn't have kids, but wants them: 3565, doesn't want kids: 2927, has kids: 1883, has a kid: 1881, doesn't have kids, and doesn't want any: 1132, has kids, but doesn't want more: 442, has a kid, but doesn't want more: 275, has a kid, and might want more: 231, wants kids: 225, might want kids: 182, has kids, and might want more: 115, has a kid, and wants more: 71, has kids, and wants more: 21

Dataset (Features)

Lifestyle Variables

- Orientation (straight: 51606; gay: 5573; bisexual: 2767)
- Pets (???: 19921; likes dogs and likes cats: 14814; likes dogs: 7224, likes dogs and has cats: 4313; has dogs: 4134; has dogs and likes cats: 2333; likes dogs and dislikes cats: 2029; has dogs and has cats: 1474; has cats: 1406; likes cats: 1063; has dogs and dislikes cats: 552; dislikes dogs and likes cats: 240, dislikes dogs and dislikes cats: 196, dislikes cats: 122; dislikes dogs and has cats: 81, dislikes dogs: 44)
- Religion (unknown: 19017; agnosticism: 8605; other: 7514; atheism: 6811; christianity: 5512; catholicism: 4522; judaism: 3024; buddhism: 1892; hinduism: 434; islam: 125)
- Sign (unknown: 0270; leo: 4241; gemini: 4144; libra: 4080; cancer: 4037; taurus: 4007; virgo: 3996; scorpio: 3975; aries: 3860; pisces: 3816; sagittarius: 3812; aquarius: 3773; capricorn: 3445)

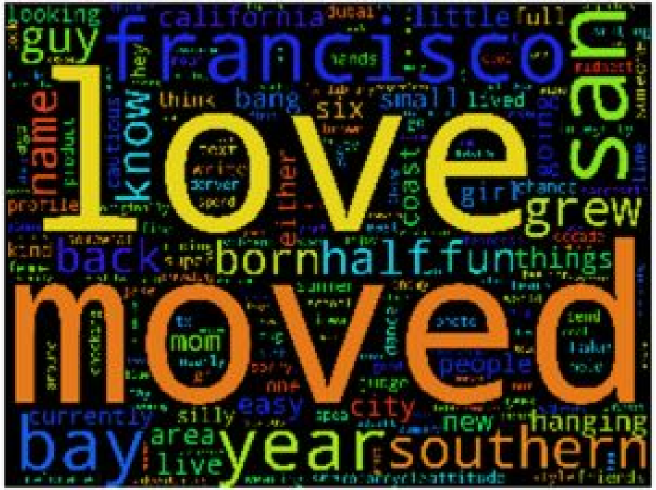
Dataset (Features)

Social Status

- Education (graduated from ph.d program: 1272, graduated from masters program: 8961, graduated from college/university: 23959, graduated from high school: 1428, other: 24326)
- Income (rather not say: 48442, others range from 200000 - 500000)
- Job (???: 8198, other: 7589, student: 4882, science / tech / engineering: 4848, computer / hardware / software: 4709, artistic / musical / writer: 4439, sales / marketing / biz dev: 4391, medicine / health: 3680, education / academia: 3513, executive / management: 2373, banking / financial / real estate: 2266, entertainment / media: 2250, law / legal services: 1381, hospitality / travel: 1364, construction / craftsmanship: 1021, clerical / administrative: 805, political / government: 708, rather not say: 436, transportation: 366, unemployed: 273, retired: 250, Military: 204)
- Speaks (English: 27531; Bilingual: 16622; Trilingual: 8488; Quadrilingual: 3180; Pentalingual: 1635)

Dataset (Essay Questions)

Essay0: My self summary

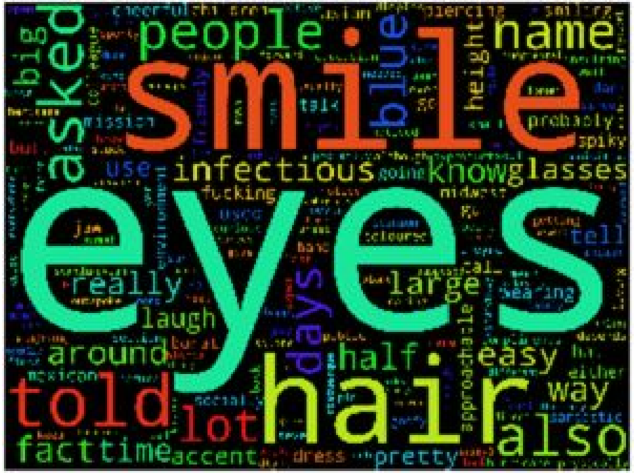


Essay1: What I’m doing with my life

Dataset (Essay Questions)

Essay2: I'm really good at

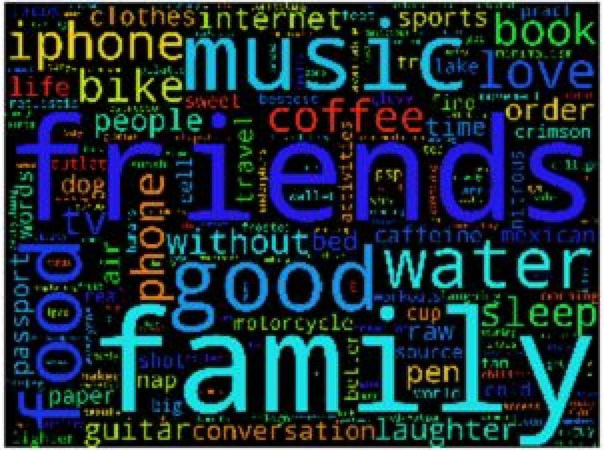
Essay3: The first thing people usually notice about me



Dataset (Essay Questions)

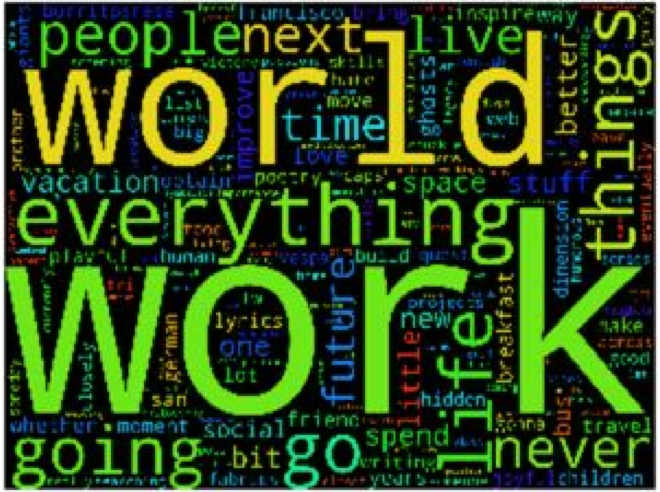
Essay4: Favorite books, movies, show, music, and food

Essay5: The six things I could never do without



Dataset (Essay Questions)

Essay6: I spend a lot of time thinking about



Essay7: On a typical Friday night I am

Dataset (Essay Questions)

Essay8: The most private thing I am willing to admit

Essay9: You should message me if...

Filtering Function

The Customized “Filtering Function” prompts the new user for their preference for each criteria as the user inputs their information. The user will also be prompted whether or not it was important. If so, existing users that do not fith that criteria will not be considered. If not, then the existing users will remain.

```
Body_Type = raw_input("What is your body type? (Possible Choices: in shape, average, not in shape, unknown): ")
Body_Type_Preference = raw_input("What is your ideal body type? (Possible Choices: 0: in shape, 1: average, 2: not in shape, 3: unknown, 4

def body_type(body):
    if Body_Type_Preference == 'in shape':
        return Age_df[Age_df['body_type'] == Body_Type_Preference]
    elif Body_Type_Preference == '0':
        return Age_df[Age_df['body_type'] == 'in shape']
    elif Body_Type_Preference == 'average':
        return Age_df[Age_df['body_type'] == Body_Type_Preference]
    elif Body_Type_Preference == '1':
        return Age_df[Age_df['body_type'] == 'average']
    elif Body_Type_Preference == 'not in shape':
        return Age_df[Age_df['body_type'] == Body_Type_Preference]
    elif Body_Type_Preference == '2':
        return Age_df[Age_df['body_type'] == 'not in shape']
    else:
        return Age_df

BT = raw_input("How important is this to you? (Possible Choices: Important: 1, Not Important: 0): ").lower()
```

Natural Language Processing

For natural language processing, I used NLTK, and Gensim Topic Modeling

```
# remove common words and tokenize
cachedStopWords = stoplist = set(stopwords.words("english"))
cachedStopWords.update(('and','i\'m', 'it\'s',
                            'a','and','so','arnt','this','when','it','many','so','cant','yes','no','these',
                           'i\'ve', 'i\'ve', 'i\'ll', 'love', 'like', ':', '&', '-', '*', '--', '~', 'im', '-i\'m', 'i\'d', 'de', 'al',
                       '(i.e.,', '1)', '2)', '3)', '4)', '5)', '6)', 'i', '(i', '3-4', '5', '.'))
# stoplist = set("for a of the and to in".split(' '))
texts = [[word for word in document.lower().split() if word not in cachedStopWords]
          for document in documents]

# remove words that appear only once
from collections import defaultdict
frequency = defaultdict(int)
for text in texts:
        for token in text:
            frequency[token] += 1

# texts = [[token for token in text if frequency[token] > 1]
#           for text in texts]

from pprint import pprint  # pretty-printer
pprint(texts)

dictionary = corpora.Dictionary(texts)
dictionary.save('/tmp/test.dict')  # store the dictionary, for future reference
#print(dictionary)

#print(dictionary.token2id)

new_doc = document
new_vec = dictionary.doc2bow(new_doc.lower().split())
#print(new_vec)  # the word "interaction" does not appear in the dictionary and is ignored
```

Latent Dirichlet Allocation (LDA)

For natural language processing, I used NLTK, and Gensim Topic Modeling

```
In [53]: lda = models.LdaModel(corpus, id2word=dictionary, num_topics=5) # initialize an LSI transformation
         corpus_lda = lda[corpus] # create a double wrapper over the original corpus: bow->tfidf->fold-in-lsi

         print lda.print_topics(5)

         for doc, i in enumerate(corpus_lda): # both bow->tfidf and tfidf->lsi transformations are actually executed here, on the fly
             print(doc, i)
```

```
[(0, u'0.008*good + 0.004*get + 0.004*new + 0.004*time + 0.004*friends + 0.004*people + 0.004*also + 0.003*enjoy + 0.003*things + 0.003*go')
 0.006*good + 0.004*friends + 0.004*people + 0.004*really + 0.003*time + 0.003*want + 0.003*go + 0.003*also + 0.003*things'), (2, u"0.005*ne
005*good + 0.005*friends + 0.004*also + 0.004*get + 0.004*enjoy + 0.004*family + 0.003*don't + 0.003*anything"), (3, u"0.006*good + 0.005*pe
 0.004*don't + 0.004*friends + 0.004*also + 0.004*really + 0.003*time + 0.003*life + 0.003*going"), (4, u"0.005*good + 0.005*get + 0.004*rea
 0.004*want + 0.004*new + 0.004*people + 0.004*friends + 0.004*life + 0.003*also")]
(0, [(4, 0.98541341272808403)])
(1, [(0, 0.9966107025764267)])
(2, [(2, 0.38297299757000453), (4, 0.60825122391020414)])
(3, [(2, 0.99043893272025896)])
(4, [(4, 0.99562928084889502)])
(5, [(1, 0.99215974910013338)])
(6, [(0, 0.011056597234123632), (3, 0.76662700099372927), (4, 0.22187950956638508)])
(7, [(2, 0.97450557653661718)])
(8, [(0, 0.96828682321299586), (3, 0.028095340462069511)])
(9, [(2, 0.98862046380419588)])
(10, [(4, 0.99625047605606398)])
(11, [(2, 0.99712736679506009)])
(12, [(3, 0.99664537107357376)])
(13, [(4, 0.99205111875273633)])
```

Similarity Score

Based on the lda topic modeling in Gensim, I calculated a similarity score based on the cosine similarity between users. An example output is below:

```
MmCorpus(556 documents, 25070 features, 89651 non-zero entries)
```

	Similarity	age	body_type	diet	drinks	drugs	education	ethnicity	height	income	...	offspring	orientation	pets	my_religion	sex	sign	smokes
555	1.000000	29	average	no restrictions	socially	no	graduated from college/university	asian	70	-1	...	none	straight	likes both	christianity	M	sagitttarius	no
395	1.000000	27	not in shape	restrictions	socially	no	graduated from two-year college	pacific islander	64	-1	...	???	straight	no answer	christianity	F	aries	no
522	1.000000	26	in shape	restrictions	socially	no	graduated from college/university	white	64	-1	...	???	straight	likes both	christianity	F	sagittarius	no
170	0.999998	26	not in	restrictions	socially	no	graduated from	black	57	-1		has a kid	straight	likes	christianity	F	taurus	no

Conclusions

- Similarity scores very high because of filtering by similarity and essay responses contain many similarities
- Natural Language Processing is pretty powerful but there are many limitations in finding pairs because most people rely on visuals
- Filters out options very quickly; might try collaborative filtering but most likely need more users
- Ran into a lot of processing issues even with this limited dataset; use AWS or other means to handle big data