Milestone_2_Part_1

April 12, 2017

1 Word Cloud and PCA Prep

1.1 Here we accomplished several tasks: 1) Collecting horror movies and romance movies from top 10000 movie data base, analyzing the contents of the movie title and movie overview, creating the wordclouds that show the most common words in two different genres; 2) Creating a corpus from the above-mentioned movies, filter the most frequent words, using a long boolean vector to indicate each word's appearance in each movie's title or overview; 3) Conducting PCA and choosing first PCs that explain 80% of variance in the data, cleaning the data format and outputting to .csv files for further PCA and SVM study in R.

In [1]:
```
import pandas as pd
import string
```

In [2]:
```
### Read in Top 10000 movies ###

movies = pd.read_csv("movies.csv", index_col=0)
movies = pd.DataFrame(movies)
```

In [3]:
```
### Filter out movies with invalid information format ###

valid_genre_filter = [type(i) is str for i in movies["genre_ids"]]
movies = movies[valid_genre_filter]
valid_title_filter = [type(i) is str for i in movies["title"]]
movies = movies[valid_title_filter]
```

In [4]:
```
### Remaining number of movies ###

len(movies)
```

Out[4]:
```
9814
```

In [5]:
```
### Collecting Romance movies and Horror movies from the data ###

Romance_movies =[]
Horror_movies = []
for key, movie in movies.iterrows():
    if "10749" in movie["genre_ids"]:
```

```
            Romance_movies.append(movie)
        elif "27" in movie["genre_ids"]:
            Horror_movies.append(movie)
```

In [6]:
```
### Number of romance movies ###

len(Romance_movies)
```

Out[6]: 1358

In [7]:
```
### Number of horror movies ###

len(Horror_movies)
```

Out[7]: 1327

In [8]:
```
### Combined text of overview information from horror movies ###

H_text = ""
for movie in Horror_movies:
    if isinstance(movie["overview"],str):
        H_text+=movie["overview"]
```

In [9]:
```
### Comined text of overview information from romance movies ###

R_text = ""
for movie in Romance_movies:
    if isinstance(movie["overview"],str):
        R_text+=movie["overview"]
```

In [10]:
```
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
import numpy as np
from PIL import Image
from os import path
import matplotlib.pyplot as plt
```

In [47]:
```
### The list of stopwords ###

stopwords = ['a', 'about', 'above', 'across', 'after', 'afterwards']
stopwords += ['again', 'against', 'all', 'almost', 'alone', 'along']
stopwords += ['already', 'also', 'although', 'always', 'am', 'among']
stopwords += ['amongst', 'amoungst', 'amount', 'an', 'and', 'another']
stopwords += ['any', 'anyhow', 'anyone', 'anything', 'anyway', 'anywhere']
stopwords += ['are', 'around', 'as', 'at', 'back', 'be', 'became']
stopwords += ['because', 'become', 'becomes', 'becoming', 'been']
stopwords += ['before', 'beforehand', 'behind', 'being', 'below']
stopwords += ['beside', 'besides', 'between', 'beyond', 'bill', 'both']
stopwords += ['bottom', 'but', 'by', 'call', 'can', 'cannot', 'cant']
stopwords += ['co', 'computer', 'con', 'could', 'couldnt', 'cry', 'de']
```

```
stopwords += ['describe', 'detail', 'did', 'do', 'done', 'down', 'due']
stopwords += ['during', 'each', 'eg', 'eight', 'either', 'eleven', 'else']
stopwords += ['elsewhere', 'empty', 'enough', 'etc', 'even', 'ever']
stopwords += ['every', 'everyone', 'everything', 'everywhere', 'except']
stopwords += ['few', 'fifteen', 'fifty', 'fill', 'find', 'fire', 'first']
stopwords += ['five', 'for', 'former', 'formerly', 'forty', 'found']
stopwords += ['four', 'from', 'front', 'full', 'further', 'get', 'give']
stopwords += ['go', 'had', 'has', 'hasnt', 'have', 'he', 'hence', 'her']
stopwords += ['here', 'hereafter', 'hereby', 'herein', 'hereupon', 'hers']
stopwords += ['herself', 'him', 'himself', 'his', 'how', 'however']
stopwords += ['hundred', 'i', 'ie', 'if', 'in', 'inc', 'indeed']
stopwords += ['interest', 'into', 'is', 'it', 'its', 'itself', 'keep']
stopwords += ['last', 'latter', 'latterly', 'least', 'less', 'ltd', 'made']
stopwords += ['many', 'may', 'me', 'meanwhile', 'might', 'mill', 'mine']
stopwords += ['more', 'moreover', 'most', 'mostly', 'move', 'much']
stopwords += ['must', 'my', 'myself', 'name', 'namely', 'neither', 'never']
stopwords += ['nevertheless', 'next', 'nine', 'no', 'nobody', 'none']
stopwords += ['noone', 'nor', 'not', 'nothing', 'now', 'nowhere', 'of']
stopwords += ['off', 'often', 'on','once', 'one', 'only', 'onto', 'or']
stopwords += ['other', 'others', 'otherwise', 'our', 'ours', 'ourselves']
stopwords += ['out', 'over', 'own', 'part', 'per', 'perhaps', 'please']
stopwords += ['put', 'rather', 're', 's', 'same', 'see', 'seem', 'seemed']
stopwords += ['seeming', 'seems', 'serious', 'several', 'she', 'should']
stopwords += ['show', 'side', 'since', 'sincere', 'six', 'sixty', 'so']
stopwords += ['some', 'somehow', 'someone', 'something', 'sometime']
stopwords += ['sometimes', 'somewhere', 'still', 'such', 'system', 'take']
stopwords += ['ten', 'than', 'that', 'the', 'their', 'them', 'themselves']
stopwords += ['then', 'thence', 'there', 'thereafter', 'thereby']
stopwords += ['therefore', 'therein', 'thereupon', 'these', 'they']
stopwords += ['thick', 'thin', 'third', 'this', 'those', 'though', 'three']
stopwords += ['three', 'through', 'throughout', 'thru', 'thus', 'to']
stopwords += ['together', 'too', 'top', 'toward', 'towards', 'twelve']
stopwords += ['twenty', 'two', 'un', 'under', 'until', 'up', 'upon']
stopwords += ['us', 'very', 'via', 'was', 'we', 'well', 'were', 'what']
stopwords += ['whatever', 'when', 'whence', 'whenever', 'where']
stopwords += ['whereafter', 'whereas', 'whereby', 'wherein', 'whereupon']
stopwords += ['wherever', 'whether', 'which', 'while', 'whither', 'who']
stopwords += ['whoever', 'whole', 'whom', 'whose', 'why', 'will', 'with']
stopwords += ['within', 'without', 'would', 'yet', 'you', 'your']
stopwords += ['yours', 'yourself', 'yourselves']

stopwords_set=set(stopwords)
```

In [48]:
```
### Generating wordcloud from horror movies ###

skull_mask = np.array(Image.open("skull.png"))
skull_wc = WordCloud(background_color = "white", max_words=100, mask = sku
                     stopwords=stopwords, max_font_size=30 )
```

```
skull_wc.generate(H_text)

h_poster = np.array(Image.open("horror.png"))
h_color = ImageColorGenerator(h_poster)

skull_wc = skull_wc.recolor(color_func=h_color)

skull_wc.to_file("skull_wc.png")

%matplotlib inline
plt.imshow(skull_wc, interpolation="bilinear")
plt.axis("off")
plt.figure()
```

Out[48]: <matplotlib.figure.Figure at 0x9fb70b8>

<matplotlib.figure.Figure at 0x9fb70b8>

In [49]:
```
### Generating wordcloud from romance movies ###
heart_mask = np.array(Image.open("heart.png"))
heart_wc = WordCloud(background_color = "white", max_words=100, mask = hea
                     stopwords=stopwords,max_font_size=30)
heart_wc.generate(R_text)

r_poster= np.array(Image.open("romance.png"))
```

```
r_color = ImageColorGenerator(r_poster)

heart_wc=heart_wc.recolor(color_func=r_color)
heart_wc.to_file("heart_wc.png")

%matplotlib inline
plt.imshow(heart_wc, interpolation="bilinear")
plt.axis("off")
plt.figure()
```

Out[49]: <matplotlib.figure.Figure at 0x9e12cc0>

<matplotlib.figure.Figure at 0x9e12cc0>

In [14]: *### Combined information of horror and romance movies ###*

```
Combined = Romance_movies+Horror_movies
```

In [15]: *### Creating the corpus ###*

```
wordlist = []
translator = str.maketrans('', '', string.punctuation)
for movie in Combined:
    if isinstance(movie["overview"],str):
        wordstring=movie["overview"].lower()
```

```
            wordstring=wordstring.translate(translator)
            wordlist.extend(wordstring.split())
        if isinstance(movie["title"],str):
            wordstring=movie["title"].lower()
            wordstring=wordstring.translate(translator)
            wordlist.extend(wordstring.split())
```

In [16]:
```
### Length of the wordlist ###

len(wordlist)
```

Out[16]: 153678

In [18]:
```
### Filter out stopwords in wordlist ###

wordlist = [w for w in wordlist if w not in stopwords]
```

In [19]:
```
### Remaining number of words in wordlist ###

len(wordlist)
```

Out[19]: 82380

In [20]:
```
### Creating a word dictionary with word frequency ###

worddict = {}
for i in wordlist:
    if i not in worddict:
        worddict[i]=1
    else:
        worddict[i]+=1

wordfreq = [(worddict[key], key) for key in worddict]
```

In [21]:
```
### Only keep the words with frequency more than 30 times ###

wordfreq = [(freq,word) for (freq,word) in wordfreq if freq>30]
```

In [22]:
```
### Number of unique words ###

len(wordfreq)
```

Out[22]: 390

In [43]:
```
### How does this word_freq list look like ###

wordfreq.sort()
wordfreq.reverse()
wordfreq[:20]
```

```
Out[43]: [(621, 'love'),
          (488, 'life'),
          (480, 'young'),
          (383, 'new'),
          (325, 'man'),
          (293, 'woman'),
          (283, 'family'),
          (269, 'friends'),
          (249, 'story'),
          (233, 'years'),
          (229, 'group'),
          (225, 'time'),
          (222, 'world'),
          (221, 'day'),
          (216, 'girl'),
          (214, 'night'),
          (212, 'school'),
          (209, 'home'),
          (200, 'finds'),
          (191, 'house')]
```

```
In [24]: ### The unique word list ###

         overview_dictionary = set()
         for (freq,word) in wordfreq:
             overview_dictionary.add(word)
```

```
In [25]: ### Creating a dataframe with the word frequency as a vector ###

         movie_word_freq = {}
         for movie in Combined:
             info ={}
             freq=[]
             for word in overview_dictionary:
                 if type(movie["overview"]) is str and word in movie["overview"]:
                     freq.append(1)
                 else:
                     freq.append(0)
             if sum(freq)>15:
                 info["freq"]=freq
                 if "27" in movie["genre_ids"]:
                     info["genre"]="Horror"
                 elif "10749" in movie["genre_ids"]:
                     info["genre"]="Romance"
                 info["title"]=movie["title"]
                 movie_word_freq[movie["title"]]=info

         df = pd.DataFrame.transpose(pd.DataFrame(movie_word_freq))
         df.index=range(len(df))
```

In [44]:
```
### How's it look like ###

df.iloc[:20]
```

Out[44]:

	genre	title	0	1	2	3	4	5	6	7	...	380	38
0	Romance	(500) Days of Summer	0	0	0	0	0	0	0	0	...	0	
1	Horror	13 Cameras	0	1	0	0	0	0	0	0	...	0	
2	Horror	13 Hours in a Warehouse	0	0	0	0	0	0	0	0	...	0	
3	Horror	13Hrs	0	0	0	0	0	0	0	0	...	0	
4	Horror	1972 Yellow House	0	0	0	0	0	0	0	0	...	0	
5	Romance	2 Days in Paris	1	0	0	0	0	0	0	0	...	0	
6	Horror	247°F	0	0	0	0	0	0	0	0	...	1	
7	Romance	28 Hotel Rooms	0	0	0	0	0	0	0	0	...	0	
8	Horror	28 Weeks Later	0	0	0	0	0	0	0	0	...	0	
9	Romance	3	0	1	0	0	0	0	0	0	...	0	
10	Horror	3 A.M.	0	0	0	0	0	0	0	0	...	0	
11	Romance	3 Idiots	1	1	0	0	0	0	0	0	...	0	
12	Horror	30 Days of Night	0	0	0	0	0	0	0	0	...	0	
13	Romance	35 and Ticking	0	0	0	0	0	0	0	0	...	0	
14	Romance	360	0	0	0	0	0	0	0	0	...	0	
15	Horror	4bia	0	1	0	0	0	0	0	0	...	0	
16	Horror	4th Period Mystery	0	0	0	0	0	0	0	0	...	0	
17	Horror	5150 Elm's Way	0	0	0	0	0	0	0	0	...	0	
18	Horror	7eventy 5ive	0	0	0	0	0	0	0	0	...	0	
19	Horror	9 Days	0	0	0	0	0	0	0	0	...	0	

	382	383	384	385	386	387	388	389
0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0
2	0	1	0	0	0	1	0	1
3	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	1
5	0	0	0	0	0	1	0	0
6	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	1	0
9	0	0	0	0	0	1	0	0
10	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0
15	0	0	0	0	0	1	0	1
16	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0

```
[20 rows x 392 columns]
```

In [27]:
```
### Transform the dataframe to have each word as a single feature ###

vector = pd.DataFrame(df["freq"].tolist())
df = pd.concat([df,vector], axis=1)
del df["freq"]
```

In [45]:
```
### How's this dataframe look like now ###

df.iloc[:20]
```

Out[45]:

	genre	title	0	1	2	3	4	5	6	7	...	380	38
0	Romance	(500) Days of Summer	0	0	0	0	0	0	0	0	...	0	
1	Horror	13 Cameras	0	1	0	0	0	0	0	0	...	0	
2	Horror	13 Hours in a Warehouse	0	0	0	0	0	0	0	0	...	0	
3	Horror	13Hrs	0	0	0	0	0	0	0	0	...	0	
4	Horror	1972 Yellow House	0	0	0	0	0	0	0	0	...	0	
5	Romance	2 Days in Paris	1	0	0	0	0	0	0	0	...	0	
6	Horror	247°F	0	0	0	0	0	0	0	0	...	1	
7	Romance	28 Hotel Rooms	0	0	0	0	0	0	0	0	...	0	
8	Horror	28 Weeks Later	0	0	0	0	0	0	0	0	...	0	
9	Romance	3	0	1	0	0	0	0	0	0	...	0	
10	Horror	3 A.M.	0	0	0	0	0	0	0	0	...	0	
11	Romance	3 Idiots	1	1	0	0	0	0	0	0	...	0	
12	Horror	30 Days of Night	0	0	0	0	0	0	0	0	...	0	
13	Romance	35 and Ticking	0	0	0	0	0	0	0	0	...	0	
14	Romance	360	0	0	0	0	0	0	0	0	...	0	
15	Horror	4bia	0	1	0	0	0	0	0	0	...	0	
16	Horror	4th Period Mystery	0	0	0	0	0	0	0	0	...	0	
17	Horror	5150 Elm's Way	0	0	0	0	0	0	0	0	...	0	
18	Horror	7eventy 5ive	0	0	0	0	0	0	0	0	...	0	
19	Horror	9 Days	0	0	0	0	0	0	0	0	...	0	

	382	383	384	385	386	387	388	389
0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0
2	0	1	0	0	0	1	0	1
3	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	1
5	0	0	0	0	0	1	0	0
6	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	1	0
9	0	0	0	0	0	1	0	0
10	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0

```
13    0    0    0    0    0    0    0    0
14    0    0    0    0    0    0    0    0
15    0    0    0    0    0    1    0    1
16    0    0    0    0    0    0    0    0
17    0    0    0    0    0    0    0    0
18    0    0    0    0    0    0    0    0
19    0    0    0    0    0    0    0    0

[20 rows x 392 columns]
```

In [29]:
```
### Number of horror movies after word-freq filtering ###

sum(df["genre"]=="Horror")
```

Out[29]: 527

In [30]:
```
### Number of romance movies after word-freq filtering ###

sum(df["genre"]=="Romance")
```

Out[30]: 570

In [46]:
```
### Separating the feature matrix for PCA ###
feature = df.ix[:,2:2+len(wordfreq)]
feature.iloc[:20]
```

Out[46]:

	0	1	2	3	4	5	6	7	8	9	...	380	381	382
0	0	0	0	0	0	0	0	0	0	0	...	0	1	0
1	0	1	0	0	0	0	0	0	0	0	...	0	0	0
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0
3	0	0	0	0	0	0	0	0	0	0	...	0	1	0
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0
5	1	0	0	0	0	0	0	0	0	0	...	0	1	0
6	0	0	0	0	0	0	0	0	0	0	...	1	1	0
7	0	0	0	0	0	0	0	0	0	0	...	0	0	0
8	0	0	0	0	0	0	0	0	0	0	...	0	0	0
9	0	1	0	0	0	0	0	0	0	0	...	0	0	0
10	0	0	0	0	0	0	0	0	0	0	...	0	0	0
11	1	1	0	0	0	0	0	0	0	0	...	0	1	0
12	0	0	0	0	0	0	0	0	0	0	...	0	0	0
13	0	0	0	0	0	0	0	0	0	0	...	0	1	0
14	0	0	0	0	0	0	0	0	0	0	...	0	0	0
15	0	1	0	0	0	0	0	0	0	0	...	0	1	0
16	0	0	0	0	0	0	0	0	0	0	...	0	1	0
17	0	0	0	0	0	0	0	0	0	0	...	0	1	0
18	0	0	0	0	0	0	0	0	0	0	...	0	0	0
19	0	0	0	0	0	0	0	0	0	0	...	0	0	0

	384	385	386	387	388	389

```
0     0    0    0    0    0    1
1     0    0    0    0    0    0
2     0    0    0    1    0    1
3     0    0    0    0    0    0
4     0    0    0    0    0    1
5     0    0    0    1    0    0
6     0    0    0    0    0    0
7     0    0    0    0    0    0
8     0    0    0    0    1    0
9     0    0    0    1    0    0
10    0    0    0    0    0    0
11    0    0    0    0    0    0
12    0    0    0    0    0    0
13    0    0    0    0    0    0
14    0    0    0    0    0    0
15    0    0    0    1    0    1
16    0    0    0    0    0    0
17    0    0    0    0    0    0
18    0    0    0    0    0    0
19    0    0    0    0    0    0

[20 rows x 390 columns]
```

In [32]:
```
from sklearn.decomposition import PCA
```

In [33]:
```
### Initial step in PCA ###

pca=PCA(n_components=len(feature.columns))
pca.fit(feature)
```

Out[33]:
```
PCA(copy=True, n_components=390, whiten=False)
```

In [34]:
```
### First 150 PCs explain above 80% of variance in data ###

sum(pca.explained_variance_ratio_[:150])
```

Out[34]:
```
0.8091573050580031
```

In [35]:
```
### Output .csv files for further analysis in R ###

feature.to_csv("feature.csv")
genre = df["genre"]
genre.to_csv("genre.csv")
title = df["title"]
title.to_csv("title.csv")
```