Similar Movies Project

May 1, 2024

[ ]:
```
#This project is to show how to use natural language processing and␣
↪unsupervised learning in Python.
```

[48]:
```
import numpy as np
import pandas as pd
import os
import nltk

nltk.download('punkt')

np.random.seed(5)

#import data

movies_df=pd.read_csv("movies.txt")

print("Number of movies loaded: %s " % (len(movies_df)))

movies_df
```

```
Number of movies loaded: 100
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\benru\AppData\Roaming\nltk_data…
[nltk_data]   Package punkt is already up-to-date!
```

[48]:

	rank	title
0	0	The Godfather
1	1	The Shawshank Redemption
2	2	Schindler's List
3	3	Raging Bull
4	4	Casablanca
..	…	…
95	95	Rebel Without a Cause
96	96	Rear Window
97	97	The Third Man
98	98	North by Northwest
99	99	Yankee Doodle Dandy

```
                                        genre  \
0                        [u' Crime', u' Drama']
1                        [u' Crime', u' Drama']
2       [u' Biography', u' Drama', u' History']
3         [u' Biography', u' Drama', u' Sport']
4             [u' Drama', u' Romance', u' War']
..                                          …
95                                  [u' Drama']
96                 [u' Mystery', u' Thriller']
97  [u' Film-Noir', u' Mystery', u' Thriller']
98                 [u' Mystery', u' Thriller']
99     [u' Biography', u' Drama', u' Musical']

                                           wiki_plot  \
0   On the day of his only daughter's wedding, Vit…
1   In 1947, banker Andy Dufresne is convicted of …
2   In 1939, the Germans move Polish Jews into the…
3   In a brief scene in 1964, an aging, overweight…
4   It is early December 1941. American expatriate…
..                                                …
95  \r\n\r\n\r\n\r\nJim Stark is in police custody…
96  \r\n\r\n\r\n\r\nJames Stewart as L.B. Jefferie…
97  \r\n\r\n\r\n\r\nSocial network mapping all maj…
98  Advertising executive Roger O. Thornhill is mi…
99   \r\n  In the early days of World War II, Coha…

                                           imdb_plot
0   In late summer 1945, guests are gathered for t…
1   In 1947, Andy Dufresne (Tim Robbins), a banker…
2   The relocation of Polish Jews from surrounding…
3   The film opens in 1964, where an older and fat…
4   In the early years of World War II, December 1…
..                                                …
95  Shortly after moving to Los Angeles with his p…
96  L.B. "Jeff" Jeffries (James Stewart) recuperat…
97  Sights of Vienna, Austria, flash across the sc…
98  At the end of an ordinary work day, advertisin…
99                                              NaN

[100 rows x 5 columns]
```

[49]:
```python
# Combine wiki_plot and imdb_plot into a single column
movies_df["plot"] = movies_df["wiki_plot"].astype(str) + "\n" + \
                    movies_df["imdb_plot"].astype(str)

# Inspect the new DataFrame
movies_df.head()
```

[49]:
```
   rank                    title                                     genre  \
0     0             The Godfather                   [u' Crime', u' Drama']
1     1  The Shawshank Redemption                   [u' Crime', u' Drama']
2     2          Schindler's List  [u' Biography', u' Drama', u' History']
3     3               Raging Bull    [u' Biography', u' Drama', u' Sport']
4     4                Casablanca        [u' Drama', u' Romance', u' War']

                                          wiki_plot  \
0  On the day of his only daughter's wedding, Vit…
1  In 1947, banker Andy Dufresne is convicted of …
2  In 1939, the Germans move Polish Jews into the…
3  In a brief scene in 1964, an aging, overweight…
4  It is early December 1941. American expatriate…

                                          imdb_plot  \
0  In late summer 1945, guests are gathered for t…
1  In 1947, Andy Dufresne (Tim Robbins), a banker…
2  The relocation of Polish Jews from surrounding…
3  The film opens in 1964, where an older and fat…
4  In the early years of World War II, December 1…

                                             plot
0  On the day of his only daughter's wedding, Vit…
1  In 1947, banker Andy Dufresne is convicted of …
2  In 1939, the Germans move Polish Jews into the…
3  In a brief scene in 1964, an aging, overweight…
4  It is early December 1941. American expatriate…
```

[50]:
```
# Tokenize a paragraph into sentences and store in sent_tokenized
sent_tokenized = [sent for sent in nltk.sent_tokenize("""
                        Today (May 19, 2016) is his only daughter's wedding.
                        Vito Corleone is the Godfather.
                        """)]

# Word Tokenize first sentence from sent_tokenized, save as words_tokenized
words_tokenized= [word for word in nltk.word_tokenize(sent_tokenized[0])]

# Remove tokens that do not contain any letters from words_tokenized
import re

filtered = [word for word in words_tokenized if re.search('[a-zA-Z]', word)]

# Display filtered words to observe words after tokenization
filtered
```

[50]:
```
['Today', 'May', 'is', 'his', 'only', 'daughter', "'s", 'wedding']
```

[51]:
```
# Import the SnowballStemmer to perform stemming
from nltk.stem.snowball import SnowballStemmer

# Create an English language SnowballStemmer object
stemmer = SnowballStemmer("english")

# Print filtered to observe words without stemming
print("Without stemming: ", filtered)

# Stem the words from filtered and store in stemmed_words
stemmed_words = [stemmer.stem(word) for word in filtered]

# Print the stemmed_words to observe words after stemming
print("After stemming:   ", stemmed_words)
```

```
Without stemming:  ['Today', 'May', 'is', 'his', 'only', 'daughter', "'s",
'wedding']
After stemming:    ['today', 'may', 'is', 'his', 'onli', 'daughter', "'s",
'wed']
```

[52]:
```
# Define a function to perform both stemming and tokenization
def tokenize_and_stem(text):

    # Tokenize by sentence, then by word
    tokens = [word for sent in nltk.sent_tokenize(text) for word in nltk.
 ↪word_tokenize(sent)]

    # Filter out raw tokens to remove noise
    filtered_tokens = [token for token in tokens if re.search('[a-zA-Z]',␣
 ↪token)]

    # Stem the filtered_tokens
    stems = [stemmer.stem(t) for t in filtered_tokens]

    return stems

words_stemmed = tokenize_and_stem("Today (May 19, 2016) is his only daughter's␣
 ↪wedding.")
print(words_stemmed)
```

```
['today', 'may', 'is', 'his', 'onli', 'daughter', "'s", 'wed']
```

[53]:
```
# Import TfidfVectorizer to create TF-IDF vectors
from sklearn.feature_extraction.text import TfidfVectorizer

# Instantiate TfidfVectorizer object with stopwords and tokenizer
# parameters for efficient processing of text
tfidf_vectorizer = TfidfVectorizer(max_df=0.8, max_features=200000,
```

```
                                    min_df=0.2, stop_words='english',
                                    use_idf=True, tokenizer=tokenize_and_stem,
                                    ngram_range=(1,3))
```

[56]:
```
# Fit and transform the tfidf_vectorizer with the "plot" of each movie
# to create a vector representation of the plot summaries
tfidf_matrix = tfidf_vectorizer.fit_transform([x for x in movies_df["plot"]])

print(tfidf_matrix.shape)
```

```
(100, 564)
```

[58]:
```
# Import k-means to perform clusters
from sklearn.cluster import KMeans

# Create a KMeans object with 5 clusters and save as km
km = KMeans(n_clusters=5)

# Fit the k-means object with tfidf_matrix
km.fit(tfidf_matrix)

clusters = km.labels_.tolist()

# Create a column cluster to denote the generated cluster for each movie
movies_df["cluster"] = clusters

# Display number of films per cluster (clusters from 0 to 4)
movies_df['cluster'].value_counts()
```

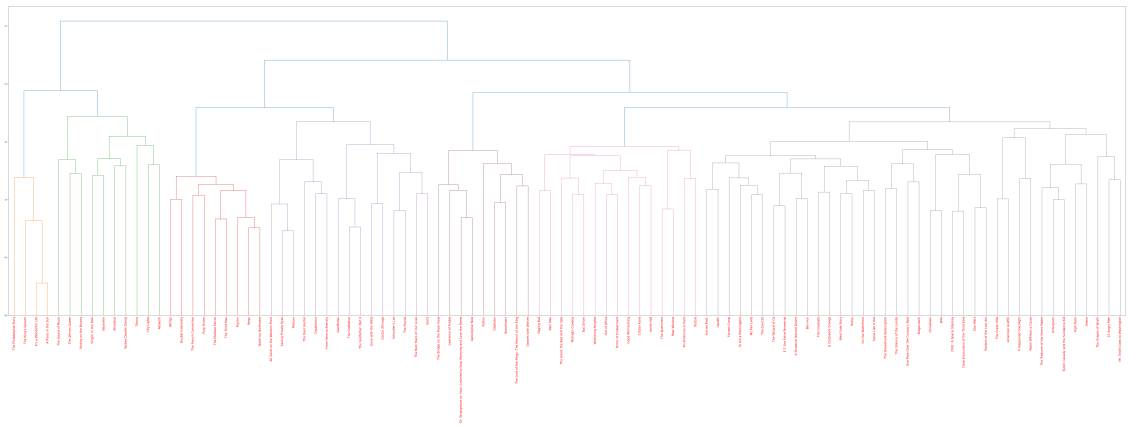
```
C:\Users\benru\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:870:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
```

[58]:
```
2    61
0    21
1    10
3     5
4     3
Name: cluster, dtype: int64
```

[59]:
```
# Import cosine_similarity to calculate similarity of movie plots
from sklearn.metrics.pairwise import cosine_similarity

# Calculate the similarity distance
similarity_distance = 1 - cosine_similarity(tfidf_matrix)
```

[61]:
```
# Import matplotlib.pyplot for plotting graphs
import matplotlib.pyplot as plt

# Configure matplotlib to display the output inline
%matplotlib inline

# Import modules necessary to plot dendrogram
from scipy.cluster.hierarchy import linkage, dendrogram
```

[64]:
```
# Create mergings matrix
mergings = linkage(similarity_distance, method='complete')

# Plot the dendrogram, using title as label column
dendrogram_ = dendrogram(mergings,
               labels=[x for x in movies_df["title"]],
               leaf_rotation=90,
               leaf_font_size=16,
)

# Adjust the plot
fig = plt.gcf()
_ = [lbl.set_color('r') for lbl in plt.gca().get_xmajorticklabels()]
fig.set_size_inches(108, 30)

# Show the plotted dendrogram
plt.show()
```



[ ]:
```
# the plot shows how similar certain movies are that are present in this␣
↪dataframe.
```