HUMA 5630 | Digital Humanities

Final Project: Text Analysis on the Lyrics of Pop Music in 2020s

Group 5: DENG, Junwei (21015955)

Section 1: Project Introducton

- why to analyze music lyrics in the field of humanities
    - music lyrics reflect the social, cultural, and historical contexts
    - music lyrics can evoke emotional responses
- why to analyze music lyrics with digital approaches
    - more data and more efficient data processing
    - broader vision on the lyrics
    - more accessible with visualization and online publishing to audience
- why pop music
    - more universal audience

Section2: Research Questions

- what are 2020s pop songs “talking” about
- how do 2020s pop songs sound (emotionally), with consideration of COVID-19

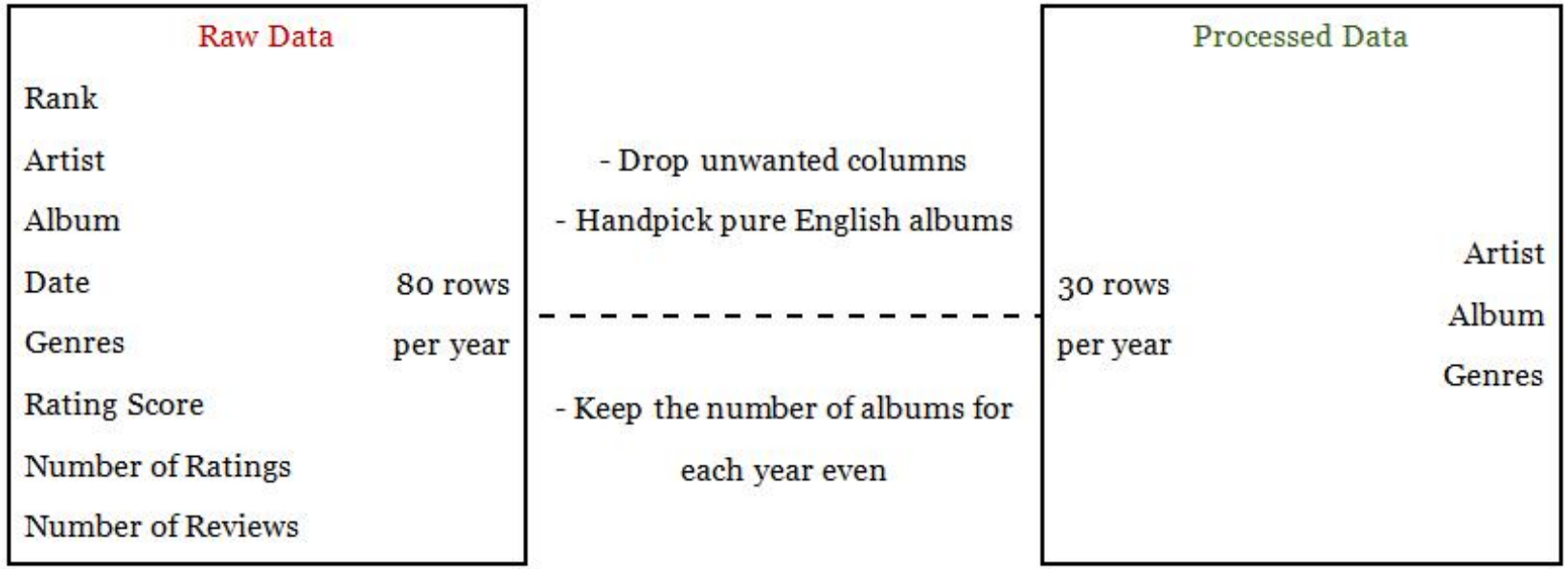
Section 3: Workflow

- Section 3.1: Data Acquiring and Processing
  - Step 1: acquire metadata of pop music publications in 2020s
    - scrape charts from rateyourmusic.com with rymscraper (from github)
    - charts sort music by rating score in descending order
    - 2 pages of chart contents (80 entries) for each year were scraped

```
3   # import necessary modules
4   import pandas as pd
5   from rymscraper import rymscraper, RymUrl
6
7   # initial the scraper
8   network = rymscraper.RymNetwork()
9
10  # now scrape
11  rym_url = RymUrl.RymUrl(kind="album", year="2020", genres="pop")
12  chart_infos = network.get_chart_infos(url=rym_url, max_page=2)
13
14  # save the data to local device
15  df = pd.DataFrame(chart_infos)
16  df.to_csv("raw_pop_20.csv", index=False)
```

Section 3: Workflow

- Section 3.1: Data Acquiring and Processing
  - Step 1: acquire metadata of pop music publications in 2020s
    - scrape charts from rateyourmusic.com with rymscraper (from github)
    - charts sort music by rating score in descending order
    - 2 pages of chart contents (80 entries) for each year were scraped
    - data structure



Section 3: Workflow

- Section 3.1: Data Acquiring and Processing
  - Step 2: acquire lyrics on the basis of the processed metadata
    - scrape lyrics from genius.com with lyricsgenius (from github)

```python
# initial the scraper
import lyricsgenius as lg
token = "<your token>"
genius = lg.Genius(access_token=token, timeout=15, remove_section_headers=True)

# import the metadata
import pandas as pd
df = pd.read_csv("pop20.csv")

# now scrape
count = 0
for i in range(df.shape[0]):
    artist = df.iloc[i, 0]
    name = df.iloc[i, 1]
    album = genius.search_album(name=name, artist=artist)
    if album is None:
        pass
    else:
        album.save_lyrics(filename=f"{count}", extension="json")
        count = count + 1
        if count == 20:
            break
```

notes:

- for each album, a json file will be returned, which contains extremely detailed information, of course including the lyrics

- I scraped the lyrics separately for each year

- as you can see, some of the albums are not included in genius.com, so I decide to terminate the scraping loop once the number of json files hits 20

Section 3: Workflow

- Section 3.1: Data Acquiring and Processing
  - Step 2: acquire lyrics on the basis of the processed metadata
    - scrape lyrics from genius.com with lyricsgenius (from github)
    - extract wanted contents form the json files

```python
# extract the wanted data
import json
contents = []
for idex in range(0, 20):
    with open(file=f"{idex}.json", mode="r") as file:
        data = json.load(file)

    cell0 = data["artist"]["name"]
    cell1 = data["name"] + " by " + cell0

    for item in data["tracks"]:
        cell2 = item["song"]["title"]
        cell3 = item["song"]["lyrics"]
        row = [cell0, cell1, cell2, cell3]
        contents.append(row)

# save the data
import pandas as pd
df = pd.DataFrame(data=contents, columns=["artist", "album", "title", "lyrics"])
df.to_csv("pop20.csv", index=False)
```

notes:

- for some of the tracks, the lyrics are missing, mainly because these tracks are interludes (pure melody)

- I took 2 approaches to deal with the missing values: drop and replace with “ ”

- NA-dropped data: 1209 rows
- NA-replaced data: 1354 rows

- the column “period” was added later

Section 3: Workflow

- Section 3.2: Data Analysis and Visualization
  - Part 1: topic modeling (with NA-dropped data)
    - Step 1: pre-process the lyrics with nltk
      - dice all the lyrics into words
      - construct a list of stopwords and remove all of them from the data
      - detect the part-of-speech of all the remaining words and pick out all the nouns

    - Step 2: implement the topic modeling with gensim
      - construct the dictionary and corpus for topic modeling with the nouns
      - try differnt number of topics and calculate the corresponding coherence score
      - settle on a proper number of topics

    - Step 3: interpret the topics with openai

Section 3: Workflow

- Section 3.2: Data Analysis and Visualization
  - Part 2.1: sentiment analysis (with NA-dropped data)
    - Note: lyrics of 20 pop albums released during 2017~2019 are included as baseline
    - Step 1: detect the sentiment of each track with textblob (polarity & subjectivity)
      - polarity: range from -1 (negative) to 1 (positive)
      - subjectivity: range from 0 (telling fact) to 1 (telling opinion)
    - Step 2: tell the gender of the artists with openai (female, male and neutral)
    - Step 3: plot the sentiment with seaborn (strip-plot & scatter-plot)
      - polarity against the periods with differrnt artist genders
      - subjectivity against the periods with different artist genders
      - sentiment map of the periods

Section 3: Workflow

- Section 3.2: Data Analysis and Visualization
  - Part 2.2: sentiment analysis (with NA-replaced data)
    - Note: lyrics of 20 pop albums released during 2017~2019 are included as baseline
    - Step 1: handpick 20 albums which I am intersted in
    - Step 2: plot the sentiment of these 20 albums with seaborn (heatmap)
      - polarity against individual tracks
      - subjectivity against individual tracks
      - note: polarity and subjectivity for “ ” are both 0

Section 3: Workflow

- Section 3.2: Data Analysis and Visualization
  - Part 3 (by-product): word embedding (on the genres)
    - Step 1: get a list of music genres defined by rateyourmusic.com from github
      - the list contains 1772 items
      - rym defines a very detailed system of music genres and pop is only a major one
      - other major music genres includes country, hip hop, jazz, punk ...
      - there are many subgenres under the umbrella of the major genres
      - note: it is a 2021 version list and contents may have been updated
      - note: I tried to scrape the latest one but I failed and my IP is blocked now :\
    - Step 2: add a new binary column to tell if the genre is included in my data
    - Step 3: get the embeddings of the genres with openai
    - Step 4: export 2 tsv files (embeddings & metadata)
    - Step 5: visualize on projector.tensorflow.org

Section 3: Workflow

- Section 3.2: Data Analysis and Visualization
  - Part 3 (by-product): word embedding (on the genres)
    - embeddings express the strings of music genres in the form of vector
    - for each string of genre, openai returns a vector with around 1500 dimensions
    - projector.tensorflow.org is an online embedding visualization platform
    - it provides different algorithms to reduce the embedding dimension to 2 or 3 for plotting
    - with visualizing the embeddings, we can intuitively observe the relations among the genres in the respect of their connotations

Section 4: Project Demonstration (with findings)

- I created a webpage with streamlit to showcase this project
- the last segment of DH workflow (data publishing)

Section 5: Further Enhancement

- more data
- more direct project description
- interactive plots
- aesthetic

Thanks for listening!