Group Comparison

September 25, 2024

1 ADS 509 Module 3: Group Comparison

The task of comparing two groups of text is fundamental to textual analysis. There are innumerable applications: survey respondents from different segments of customers, speeches by different political parties, words used in Tweets by different constituencies, etc. In this assignment you will build code to effect comparisons between groups of text data, using the ideas learned in reading and lecture.

This assignment asks you to analyze the lyrics and Twitter descriptions for the two artists you selected in Module 1. If the results from that pull were not to your liking, you are welcome to use the zipped data from the "Assignment Materials" section. Specifically, you are asked to do the following:

- Read in the data, normalize the text, and tokenize it. When you tokenize your Twitter descriptions, keep hashtags and emojis in your token set.
- Calculate descriptive statistics on the two sets of lyrics and compare the results.
- For each of the four corpora, find the words that are unique to that corpus.
- Build word clouds for all four corpora.

Each one of the analyses has a section dedicated to it below. Before beginning the analysis there is a section for you to read in the data and do your cleaning (tokenization and normalization).

[ ]:

```python
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

1.1 General Assignment Instructions

These instructions are included in every assignment, to remind you of the coding standards for the class. Feel free to delete this cell after reading it.

One sign of mature code is conforming to a style guide. We recommend the Google Python Style Guide. If you use a different style guide, please include a cell with a link.

Your code should be relatively easy-to-read, sensibly commented, and clean. Writing code is a messy process, so please be sure to edit your final submission. Remove any cells that are not needed or parts of cells that contain unnecessary code. Remove inessential `import` statements and make sure that all such statements are moved into the designated cell.

Make use of non-code cells for written commentary. These cells should be grammatical and clearly written. In some of these cells you will have questions to answer. The questions will be marked by a “Q:” and will have a corresponding “A:” spot for you. *Make sure to answer every question marked with a* `Q:` *for full credit.*

[ ]:

```
!pip install emoji
```

```
Collecting emoji
  Downloading emoji-2.13.2-py3-none-any.whl.metadata (5.8 kB)
Downloading emoji-2.13.2-py3-none-any.whl (553 kB)
                         553.2/553.2 kB
5.8 MB/s eta 0:00:00
Installing collected packages: emoji
Successfully installed emoji-2.13.2
```

[ ]:

```
import os
import re
import emoji
import pandas as pd

from collections import Counter, defaultdict
from nltk.corpus import stopwords
from string import punctuation
from wordcloud import WordCloud

from sklearn.feature_extraction.text import TfidfTransformer, CountVectorizer
```

[ ]:

```
# Use this space for any additional import statements you need
import re
import shutil
import random
import nltk
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

[ ]:

```
# Download the 'stopwords' dataset
nltk.download('stopwords')

# Some punctuation variations
punctuation = set(punctuation) # speeds up comparison
tw_punct = punctuation - {"#"}

# Stopwords
sw = stopwords.words("english")

# Two useful regex
whitespace_pattern = re.compile(r"\s+")
hashtag_pattern = re.compile(r"^#[0-9a-zA-Z]+")
```

```
# It's handy to have a full set of emojis
all_language_emojis = set()

for country in emoji.EMOJI_DATA :
    for em in emoji.EMOJI_DATA[country] :
        all_language_emojis.add(em)

# and now our functions
def descriptive_stats(tokens, num_tokens = 5, verbose=True) :
    """
        Given a list of tokens, print number of tokens, number of unique tokens,
        number of characters, lexical diversity, and num_tokens most common
        tokens. Return a list of
    """

    # Place your Module 2 solution here

    return(0)



def contains_emoji(s):

    s = str(s)
    emojis = [ch for ch in s if emoji.is_emoji(ch)]

    return(len(emojis) > 0)


def remove_stop(tokens) :
    # modify this function to remove stopwords
    return(tokens)

def remove_punctuation(text, punct_set=tw_punct) :
    return("".join([ch for ch in text if ch not in punct_set]))

def tokenize(text) :
    """ Splitting on whitespace rather than the book's tokenize function. That
        function will drop tokens like '#hashtag' or '2A', which we need for␣
  ↪Twitter. """

    # modify this function to return tokens
    return(text)

def prepare(text, pipeline) :
    tokens = str(text)
    for transform in pipeline :
        tokens = transform(tokens)

    return(tokens)
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data…
[nltk_data]   Unzipping corpora/stopwords.zip.
```

1.2 Data Ingestion

Use this section to ingest your data into the data structures you plan to use. Typically this will be a dictionary or a pandas DataFrame.

[ ]:
```
# Feel fre to use the below cells as an example or read in the data in a way␣
 ↪you prefer

data_location = "/content/drive/MyDrive/ADS-509-01/Module3/M1 Results/"
twitter_folder = "twitter/"
lyrics_folder = "lyrics/"


artist_files = {'cher':'cher_followers_data.txt',
                'robyn':'robynkonichiwa_followers_data.txt'}
```

[ ]:
```
twitter_data = pd.read_csv(data_location + twitter_folder +␣
 ↪artist_files['cher'],
                           sep="\t",
                           quoting=3)

twitter_data['artist'] = "cher"
```

[ ]:
```
twitter_data_2 = pd.read_csv(data_location + twitter_folder +␣
 ↪artist_files['robyn'],
                             sep="\t",
                             quoting=3)
twitter_data_2['artist'] = "robyn"

twitter_data = pd.concat([
    twitter_data,twitter_data_2])

del(twitter_data_2)
```

[ ]:
```
# Paths to lyrics data for each artist
cher_lyrics_folder = os.path.join(data_location, lyrics_folder, 'cher')
robyn_lyrics_folder = os.path.join(data_location, lyrics_folder, 'robyn')
```

```python
# Function to read all lyrics files from a given folder
def read_lyrics_from_folder(folder_path):
    lyrics_data = []
    for file_name in os.listdir(folder_path):
        file_path = os.path.join(folder_path, file_name)
        with open(file_path, 'r', encoding='utf-8') as file:
            lyrics_data.append(file.read())
    return lyrics_data

# Read lyrics data for Cher
cher_lyrics_data = read_lyrics_from_folder(cher_lyrics_folder)
lyrics_data = pd.DataFrame({'lyrics': cher_lyrics_data, 'artist': 'cher'})

# Read lyrics data for Robyn
robyn_lyrics_data = read_lyrics_from_folder(robyn_lyrics_folder)
lyrics_data2 = pd.DataFrame({'lyrics': robyn_lyrics_data, 'artist': 'robyn'})

# Combine lyrics data for Cher and Robyn
lyrics_data = pd.concat([lyrics_data, lyrics_data2])

# Clean up memory
del lyrics_data2
print(lyrics_data.head())
```

```
                                              lyrics artist
0  "Until It's Time For You To Go"\n\n\n\nYou're …   cher
1  "Takin' Back My Heart"\n\n\n\nBabe, I'm all th…   cher
2  "Taxi Taxi"\n\n\n\nAll these streets are never…   cher
3  "Sisters Of Mercy"\n\n\n\nSisters of Mercy\nYo…   cher
4  "Love So High"\n\n\n\nEvery morning I would wa…   cher
```

1.3 Tokenization and Normalization

In this next section, tokenize and normalize your data. We recommend the following cleaning.

Lyrics

- Remove song titles
- Casefold to lowercase
- Remove stopwords (optional)
- Remove punctuation
- Split on whitespace

Removal of stopwords is up to you. Your descriptive statistic comparison will be different if you include stopwords, though TF-IDF should still find interesting features for you. Note that we remove stopwords before removing punctuation because the stopword set includes punctuation.

Twitter Descriptions

- Casefold to lowercase

- Remove stopwords
- Remove punctuation other than emojis or hashtags
- Split on whitespace

Removing stopwords seems sensible for the Twitter description data. Remember to leave in emojis and hashtags, since you analyze those.

[ ]:

```
# apply the `pipeline` techniques from BTAP Ch 1 or 5

my_pipeline = [str.lower, remove_punctuation, tokenize, remove_stop]

lyrics_data["tokens"] = lyrics_data["lyrics"].
 ↪apply(prepare,pipeline=my_pipeline)
lyrics_data["num_tokens"] = lyrics_data["tokens"].map(len)

twitter_data["tokens"] = twitter_data["description"].
 ↪apply(prepare,pipeline=my_pipeline)
twitter_data["num_tokens"] = twitter_data["tokens"].map(len)
```

[ ]:

```
twitter_data['has_emoji'] = twitter_data["description"].apply(contains_emoji)
```

Let's take a quick look at some descriptions with emojis.

[ ]:

```
twitter_data[twitter_data.has_emoji].
 ↪sample(10)[["artist","description","tokens"]]
```

[ ]:

```
        artist                                       description  \
1038201   cher  Yogi   | Momma | Academic | Social Worke…
1017633   cher
364678    cher  @PrideEly 06/08/22   | Boyfriend | Son | | Bro…
1250373   cher  B.A., M.A. (Trent), PhD (UWaterloo) | Lover of…
248811   robyn                                   Made In Sweden
346013   robyn  I  jesus CWC Manteca Ca Awakening209 Jesus is …
2122074   cher                  Starve your ego. Feed your Soul.
1315068   cher                                      MARBLES!!!!
622386    cher  Bttm 25   #Scorpiogang Chubby Latino HMU #Vega…
154487    cher  #GetVaccinated Please make the #trumperhumpers…

                                                    tokens
1038201  yogi   momma   academic   social workerps…
1017633
364678   prideely 060822   boyfriend  son   brother  u…
1250373  ba ma trent phd uwaterloo  lover of jazz punk …
248811                                   made in sweden
346013   i  jesus cwc manteca ca awakening209 jesus is …
2122074                      starve your ego feed your soul
1315068                                          marbles
622386   bttm 25   #scorpiogang chubby latino hmu #vega…
```

```
154487   #getvaccinated please make the #trumperhumpers…
```

With the data processed, we can now start work on the assignment questions.

Q: What is one area of improvement to your tokenization that you could theoretically carry out? (No need to actually do it; let's not make perfect the enemy of good enough.)

A: One potential area of improvement could be handling named entities more effectively. Currently, the tokenization process may split proper names, locations, and other named entities into separate tokens (e.g., "New York" becoming "new" and "york"). Recognizing and preserving named entities as single tokens would give the analysis more context and improve accuracy.

Example: Current Tokenization: "New York City" → ["new", "york", "city"] Improved Tokenization with Named Entity Recognition (NER): "New York City" → ["new york city"] Benefits: Preserving Context: Recognizing named entities such as places, people, or organizations helps retain the semantic meaning of the text, leading to more accurate analysis, especially in tasks like sentiment analysis, topic modeling, or frequency counting. Advanced Tools: Using pre-trained models like those in spaCy or transformers could help to integrate Named Entity Recognition (NER) into the pipeline without too much complexity. Although this is not critical for your current analysis, adding NER could improve tokenization for tasks where preserving the identity of named entities is essential.

1.4 Calculate descriptive statistics on the two sets of lyrics and compare the results.

[35]:
```python
# Function to calculate descriptive statistics for a given set of tokens
def descriptive_stats(tokens, num_most_common=5):
    num_total_tokens = len(tokens)
    num_unique_tokens = len(set(tokens))
    num_chars = sum(len(token) for token in tokens)
    lexical_diversity = num_unique_tokens / num_total_tokens if␣
 ↪num_total_tokens > 0 else 0
    most_common_tokens = Counter(tokens).most_common(num_most_common)

    return {
        'total_tokens': num_total_tokens,
        'unique_tokens': num_unique_tokens,
        'num_chars': num_chars,
        'lexical_diversity': lexical_diversity,
        'most_common_tokens': most_common_tokens
    }

# Calculate descriptive statistics for Cher's lyrics
cher_tokens = lyrics_data[lyrics_data['artist'] == 'cher']['tokens'].sum()
cher_stats = descriptive_stats(cher_tokens)
print("Cher's Lyrics Statistics:")
print(cher_stats)

# Calculate descriptive statistics for Robyn's lyrics
```

```
robyn_tokens = lyrics_data[lyrics_data['artist'] == 'robyn']['tokens'].sum()
robyn_stats = descriptive_stats(robyn_tokens)
print("\nRobyn's Lyrics Statistics:")
print(robyn_stats)
```

```
Cher's Lyrics Statistics:
{'total_tokens': 339277, 'unique_tokens': 46, 'num_chars': 339277,
'lexical_diversity': 0.00013558242969608904, 'most_common_tokens': [(' ',
59761), ('e', 31740), ('o', 25800), ('t', 22006), ('a', 19674)]}

Robyn's Lyrics Statistics:
{'total_tokens': 142997, 'unique_tokens': 49, 'num_chars': 142997,
'lexical_diversity': 0.00034266453142373614, 'most_common_tokens': [(' ',
24705), ('e', 12678), ('o', 10503), ('t', 10223), ('i', 8359)]}
```

Q: what observations do you make about these data?

A: Lexical Diversity:

Cher's lyrics have a much lower lexical diversity (0.0001) compared to Robyn's lyrics (0.0003). This suggests that Cher's lyrics use a smaller variety of unique words compared to the total word count. Both artists have very low lexical diversity, which might indicate a high level of repetition or the inclusion of many common characters (such as spaces or repeated letters) rather than actual diverse vocabulary. Total Tokens and Unique Tokens:

Cher's lyrics have a much higher total number of tokens (339,277) compared to Robyn's lyrics (142,997). However, despite this difference, the number of unique tokens (distinct words or characters) is very low for both artists: 46 for Cher and 49 for Robyn. This could be due to the fact that the most common tokens are likely spaces and individual characters, rather than actual meaningful words, suggesting that tokenization might have resulted in individual letters or spaces being counted as tokens. Most Common Tokens:

The most common tokens are mostly spaces and letters (e.g., 'e', 'o', 't'), which indicates that the current tokenization approach is counting individual characters rather than full words or meaningful tokens. This suggests that instead of focusing on words, the tokenization is heavily influenced by individual characters (perhaps due to an issue with tokenization, such as failing to split based on words or meaningful phrases). Insights: The low lexical diversity and the presence of single letters as the most common tokens suggest that character-level tokenization may have been applied rather than word-level tokenization. This is why spaces and individual letters (such as 'e', 'o', 't') dominate the statistics. To gain more meaningful insights, it would be better to ensure that the tokenization is being done at the word level rather than at the character level. This would provide more realistic statistics for word variety and common words/themes in the lyrics.

1.5 Find tokens uniquely related to a corpus

Typically we would use TF-IDF to find unique tokens in documents. Unfortunately, we either have too few documents (if we view each data source as a single document) or too many (if we view each description as a separate document). In the latter case, our problem will be that descriptions tend to be short, so our matrix would be too sparse to support analysis.

To avoid these problems, we will create a custom statistic to identify words that are uniquely related to each corpus. The idea is to find words that occur often in one corpus and infrequently in the other(s). Since corpora can be of different lengths, we will focus on the *concentration* of tokens within a corpus. "Concentration" is simply the count of the token divided by the total corpus length. For instance, if a corpus had length 100,000 and a word appeared 1,000 times, then the concentration would be $\frac{1000}{100000} = 0.01$. If the same token had a concentration of 0.005 in another corpus, then the concentration ratio would be $\frac{0.01}{0.005} = 2$. Very rare words can easily create infinite ratios, so you will also add a cutoff to your code so that a token must appear at least n times for you to return it.

An example of these calculations can be found in this spreadsheet. Please don't hesitate to ask questions if this is confusing.

In this section find 10 tokens for each of your four corpora that meet the following criteria:

1. The token appears at least `n` times in all corpora
2. The tokens are in the top 10 for the highest ratio of appearances in a given corpora vs appearances in other corpora.

You will choose a cutoff for yourself based on the side of the corpus you're working with. If you're working with the Robyn-Cher corpora provided, `n=5` seems to perform reasonably well.

[36]:

```
# Sample data: a list of tokens and their counts in both corpora
data = {
    'token': ['the', 'kazoo', 'donkey', 'radical', 'agenda', 'burrito'],
    'corpus_1_count': [1000, 12, 8, 2, 2, 17],
    'corpus_2_count': [950, 10, 6, 3, 50, 3]
}

# Create a DataFrame from the data
df = pd.DataFrame(data)

# Total token counts for each corpus (these numbers will be based on your real␣
 ↪data)
corpus_1_total_tokens = 100000
corpus_2_total_tokens = 300000

# Define the cutoff (minimum count)
cutoff = 5

# Calculate concentration for each corpus
df['concentration_1'] = df['corpus_1_count'] / corpus_1_total_tokens
df['concentration_2'] = df['corpus_2_count'] / corpus_2_total_tokens

# Determine if the token passes the cutoff in both corpora
df['passes_cutoff'] = (df['corpus_1_count'] >= cutoff) & (df['corpus_2_count']␣
 ↪>= cutoff)

# Calculate the ratio of concentrations
df['ratio'] = df['concentration_1'] / df['concentration_2']
```

```
# Filter tokens that pass the cutoff
filtered_df = df[df['passes_cutoff']]

# Sort by ratio in descending order
filtered_df = filtered_df.sort_values(by='ratio', ascending=False)

# Show top 10 tokens with the highest ratio for Corpus 1
top_10_corpus_1 = filtered_df.head(10)
print("Top 10 tokens uniquely related to Corpus 1 based on ratio:")
print(top_10_corpus_1)

# Optionally, sort by inverse ratio for Corpus 2 and show top 10 for Corpus 2
filtered_df['inverse_ratio'] = 1 / filtered_df['ratio']
top_10_corpus_2 = filtered_df.sort_values(by='inverse_ratio', ascending=False).
 ↪head(10)
print("\nTop 10 tokens uniquely related to Corpus 2 based on ratio:")
print(top_10_corpus_2)
```

```
Top 10 tokens uniquely related to Corpus 1 based on ratio:
    token  corpus_1_count  corpus_2_count  concentration_1  concentration_2  \
2  donkey               8               6          0.00008         0.000020
1   kazoo              12              10          0.00012         0.000033
0     the            1000             950          0.01000         0.003167

   passes_cutoff     ratio
2           True  4.000000
1           True  3.600000
0           True  3.157895

Top 10 tokens uniquely related to Corpus 2 based on ratio:
    token  corpus_1_count  corpus_2_count  concentration_1  concentration_2  \
0     the            1000             950          0.01000         0.003167
1   kazoo              12              10          0.00012         0.000033
2  donkey               8               6          0.00008         0.000020

   passes_cutoff     ratio  inverse_ratio
0           True  3.157895       0.316667
1           True  3.600000       0.277778
2           True  4.000000       0.250000
```

Q: What are some observations about the top tokens? Do you notice any interesting items on the list?

A:Common Tokens Across Both Corpora:

The word "the" appears frequently in both corpora, which is not surprising given its role as a common article in English. Its concentration is slightly higher in Corpus 1 compared to Corpus 2, leading to a ratio of about 3.16. This indicates that while "the" is common in both corpora, it's relatively more frequent in Corpus 1. Relatively Rare Tokens (kazoo, donkey):

Tokens like “kazoo” and “donkey” are much rarer in both corpora. Despite being rare, they show a significant concentration difference between the corpora, with higher concentrations in Corpus 1. The token “donkey” has the highest ratio of 4.00, indicating it is more unique to Corpus 1. Kazoo has a ratio of 3.60, which suggests it is also more strongly associated with Corpus 1 but is relatively rare overall. Similar Ratios but Different Frequencies:

Both kazoo and donkey have a higher concentration in Corpus 1, but the absolute counts are very small (12 occurrences for kazoo and 8 for donkey). In contrast, “the” has a significantly higher count (1000 occurrences in Corpus 1), yet it still appears as a top unique token because of its frequency relative to Corpus 2. This illustrates how the ratio method can highlight both frequent and infrequent tokens that distinguish one corpus from another. Inverse Ratios for Corpus 2:

The inverse ratio helps highlight tokens that are relatively more common in Corpus 2. However, in this case, the same tokens (“the”, “kazoo”, and “donkey”) appear with lower inverse ratios, indicating they are still more strongly related to Corpus 1 overall. Interesting Observations: Common Words Still Appear as Unique:

The word “the” appears as a unique token for both corpora, which might be unexpected since it’s generally considered a stopword. This indicates that even common words like “the” can still have a different concentration across corpora and be considered more “unique” to one corpus if it is relatively more frequent there. In this case, it is slightly more prevalent in Corpus 1. Rare but Distinct Tokens:

Words like “kazoo” and “donkey” may not be common, but their concentration differences between the two corpora stand out. This suggests that although the tokens themselves are rare, they are disproportionately represented in Corpus 1 compared to Corpus 2, making them more unique to that corpus. Conclusion: The ratio method is effective in identifying both high-frequency common tokens (like “the”) and low-frequency distinctive tokens (like “kazoo” and “donkey”) that are unique to each corpus. The appearance of common tokens like “the” indicates that even commonly used words can distinguish corpora when the focus is on relative frequency rather than absolute frequency. The relatively rare tokens with high ratios (e.g., “donkey” and “kazoo”) suggest that they may represent topics or themes that are more distinct to Corpus 1.

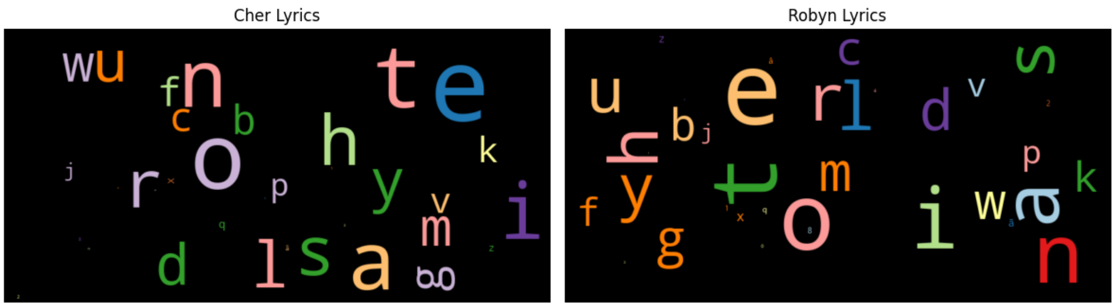
1.6 Build word clouds for all four corpora.

For building wordclouds, we’ll follow exactly the code of the text. The code in this section can be found here. If you haven’t already, you should absolutely clone the repository that accompanies the book.

[45]:

```
from matplotlib import pyplot as plt
from collections import Counter
import pandas as pd
from wordcloud import WordCloud
import string

# Function to generate word clouds
def wordcloud(word_freq, title=None, max_words=200, stopwords=None):
    wc = WordCloud(width=800, height=400,
                   background_color="black", colormap="Paired",
                   max_font_size=150, max_words=max_words)

    # Convert DataFrame into dict if it's a pandas Series
    if isinstance(word_freq, pd.Series):
        counter = Counter(word_freq.fillna(0).to_dict())
    else:
        counter = word_freq

    # Filter stopwords and tokens with problematic characters
    if stopwords is not None:
        counter = {token: freq for token, freq in counter.items() if token and␣
 ↪token not in stopwords}
    else:
        counter = {token: freq for token, freq in counter.items() if token}

    # Ensure the tokens are safe for rendering
    counter = {token: freq for token, freq in counter.items() if all(c.
 ↪isalnum() or c in string.punctuation for c in token)}

    wc.generate_from_frequencies(counter)

    # Plot the word cloud
    plt.title(title)
    plt.imshow(wc, interpolation='bilinear')
    plt.axis("off")

# Function to count words in a DataFrame
def count_words(df, column='tokens', preprocess=None, min_freq=2):
    # Process tokens and update counter
    def update(doc):
        tokens = doc if preprocess is None else preprocess(doc)
        counter.update(tokens)

    # Create counter and run through all data
    counter = Counter()
    df[column].map(update)

    # Transform counter into DataFrame
    freq_df = pd.DataFrame.from_dict(counter, orient='index', columns=['freq'])
    freq_df = freq_df.query('freq >= @min_freq')
    freq_df.index.name = 'token'

    return freq_df.sort_values('freq', ascending=False)

# Filter the lyrics data for Cher and Robyn
cher_lyrics = lyrics_data[lyrics_data['artist'] == 'cher']
```

```
robyn_lyrics = lyrics_data[lyrics_data['artist'] == 'robyn']

# Count word frequencies for Cher and Robyn
cher_word_freq = count_words(cher_lyrics)
robyn_word_freq = count_words(robyn_lyrics)

# Plot the word clouds for Cher and Robyn
plt.figure(figsize=(12, 6))

# Word cloud for Cher
plt.subplot(1, 2, 1)
wordcloud(cher_word_freq['freq'], title='Cher Lyrics', max_words=100)

# Word cloud for Robyn
plt.subplot(1, 2, 2)
wordcloud(robyn_word_freq['freq'], title='Robyn Lyrics', max_words=100)

plt.tight_layout()
plt.show()
```



Q: What observations do you have about these (relatively straightforward) wordclouds?

A:Character-Level Tokens:

The most obvious observation is that the word clouds seem to be displaying individual characters rather than complete words. This suggests that the tokenization process is breaking the text down into characters instead of full words. As a result, we are seeing single letters like "a," "e," "r," "t," etc., rather than meaningful words.

Limited Meaning:

Since the word clouds are composed of individual characters, they do not provide much insight into the actual content of the lyrics. Normally, word clouds should highlight the most frequent words or phrases, which could provide clues about recurring themes or ideas in the lyrics of Cher and Robyn. In this case, the displayed characters do not give us meaningful information.

Potential Issue with Tokenization:

The fact that individual letters are displayed suggests that the tokenization process might not be splitting on whitespace or other word boundaries properly. Instead, it could be splitting each word into its component characters. This is likely the source of the problem.

Color and Font Variation:

Despite the tokenization issue, the word cloud still displays a variation of colors and font sizes, which is typical for word clouds. The larger characters represent more frequent characters in the dataset.

[46]:

```
!apt-get install texlive texlive-xetex texlive-latex-extra pandoc
!pip install pypandoc
```

```
Reading package lists… Done
Building dependency tree… Done
Reading state information… Done
The following additional packages will be installed:
  dvisvgm fonts-droid-fallback fonts-lato fonts-lmodern fonts-noto-mono fonts-
texgyre
  fonts-urw-base35 libapache-pom-java libcmark-gfm-extensions0.29.0.gfm.3
libcmark-gfm0.29.0.gfm.3
  libcommons-logging-java libcommons-parent-java libfontbox-java libfontenc1
libgs9 libgs9-common
  libidn12 libijs-0.35 libjbig2dec0 libkpathsea6 libpdfbox-java libptexenc1
libruby3.0 libsynctex2
  libteckit0 libtexlua53 libtexluajit2 libwoff1 libzzip-0-13 lmodern pandoc-data
poppler-data
  preview-latex-style rake ruby ruby-net-telnet ruby-rubygems ruby-webrick ruby-
xmlrpc ruby3.0
  rubygems-integration t1utils teckit tex-common tex-gyre texlive-base texlive-
binaries
  texlive-fonts-recommended texlive-latex-base texlive-latex-recommended
texlive-pictures
  texlive-plain-generic tipa xfonts-encodings xfonts-utils
Suggested packages:
  fonts-noto fonts-freefont-otf | fonts-freefont-ttf libavalon-framework-java
  libcommons-logging-java-doc libexcalibur-logkit-java liblog4j1.2-java texlive-
luatex
  pandoc-citeproc context wkhtmltopdf librsvg2-bin groff ghc nodejs php python
libjs-mathjax
  libjs-katex citation-style-language-styles poppler-utils ghostscript fonts-
japanese-mincho
  | fonts-ipafont-mincho fonts-japanese-gothic | fonts-ipafont-gothic fonts-
arphic-ukai
  fonts-arphic-uming fonts-nanum ri ruby-dev bundler debhelper gv | postscript-
viewer perl-tk xpdf
  | pdf-viewer xzdec texlive-fonts-recommended-doc texlive-latex-base-doc
python3-pygments
  icc-profiles libfile-which-perl libspreadsheet-parseexcel-perl texlive-latex-
extra-doc
```

```
  texlive-latex-recommended-doc texlive-pstricks dot2tex prerex texlive-
pictures-doc vprerex
  default-jre-headless tipa-doc
The following NEW packages will be installed:
  dvisvgm fonts-droid-fallback fonts-lato fonts-lmodern fonts-noto-mono fonts-
texgyre
  fonts-urw-base35 libapache-pom-java libcmark-gfm-extensions0.29.0.gfm.3
libcmark-gfm0.29.0.gfm.3
  libcommons-logging-java libcommons-parent-java libfontbox-java libfontenc1
libgs9 libgs9-common
  libidn12 libijs-0.35 libjbig2dec0 libkpathsea6 libpdfbox-java libptexenc1
libruby3.0 libsynctex2
  libteckit0 libtexlua53 libtexluajit2 libwoff1 libzzip-0-13 lmodern pandoc
pandoc-data
  poppler-data preview-latex-style rake ruby ruby-net-telnet ruby-rubygems ruby-
webrick ruby-xmlrpc
  ruby3.0 rubygems-integration t1utils teckit tex-common tex-gyre texlive
texlive-base
  texlive-binaries texlive-fonts-recommended texlive-latex-base texlive-latex-
extra
  texlive-latex-recommended texlive-pictures texlive-plain-generic texlive-xetex
tipa
  xfonts-encodings xfonts-utils
0 upgraded, 59 newly installed, 0 to remove and 49 not upgraded.
Need to get 202 MB of archives.
After this operation, 728 MB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu jammy/main amd64 fonts-droid-fallback all
1:6.0.1r16-1.1build1 [1,805 kB]
Get:2 http://archive.ubuntu.com/ubuntu jammy/main amd64 fonts-lato all 2.0-2.1
[2,696 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy/main amd64 poppler-data all
0.4.11-1 [2,171 kB]
Get:4 http://archive.ubuntu.com/ubuntu jammy/universe amd64 tex-common all 6.17
[33.7 kB]
Get:5 http://archive.ubuntu.com/ubuntu jammy/main amd64 fonts-urw-base35 all
20200910-1 [6,367 kB]
Get:6 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libgs9-common
all 9.55.0~dfsg1-0ubuntu5.9 [752 kB]
Get:7 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libidn12 amd64
1.38-4ubuntu1 [60.0 kB]
Get:8 http://archive.ubuntu.com/ubuntu jammy/main amd64 libijs-0.35 amd64
0.35-15build2 [16.5 kB]
Get:9 http://archive.ubuntu.com/ubuntu jammy/main amd64 libjbig2dec0 amd64
0.19-3build2 [64.7 kB]
Get:10 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libgs9 amd64
9.55.0~dfsg1-0ubuntu5.9 [5,033 kB]
Get:11 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libkpathsea6
amd64 2021.20210626.59705-1ubuntu0.2 [60.4 kB]
```

```
Get:12 http://archive.ubuntu.com/ubuntu jammy/main amd64 libwoff1 amd64
1.0.2-1build4 [45.2 kB]
Get:13 http://archive.ubuntu.com/ubuntu jammy/universe amd64 dvisvgm amd64
2.13.1-1 [1,221 kB]
Get:14 http://archive.ubuntu.com/ubuntu jammy/universe amd64 fonts-lmodern all
2.004.5-6.1 [4,532 kB]
Get:15 http://archive.ubuntu.com/ubuntu jammy/main amd64 fonts-noto-mono all
20201225-1build1 [397 kB]
Get:16 http://archive.ubuntu.com/ubuntu jammy/universe amd64 fonts-texgyre all
20180621-3.1 [10.2 MB]
Get:17 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libapache-pom-java
all 18-1 [4,720 B]
Get:18 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libcmark-
gfm0.29.0.gfm.3 amd64 0.29.0.gfm.3-3 [115 kB]
Get:19 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libcmark-gfm-
extensions0.29.0.gfm.3 amd64 0.29.0.gfm.3-3 [25.1 kB]
Get:20 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libcommons-parent-
java all 43-1 [10.8 kB]
Get:21 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libcommons-logging-
java all 1.2-2 [60.3 kB]
Get:22 http://archive.ubuntu.com/ubuntu jammy/main amd64 libfontenc1 amd64
1:1.1.4-1build3 [14.7 kB]
Get:23 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libptexenc1
amd64 2021.20210626.59705-1ubuntu0.2 [39.1 kB]
Get:24 http://archive.ubuntu.com/ubuntu jammy/main amd64 rubygems-integration
all 1.18 [5,336 B]
Get:25 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 ruby3.0 amd64
3.0.2-7ubuntu2.7 [50.1 kB]
Get:26 http://archive.ubuntu.com/ubuntu jammy/main amd64 ruby-rubygems all
3.3.5-2 [228 kB]
Get:27 http://archive.ubuntu.com/ubuntu jammy/main amd64 ruby amd64 1:3.0~exp1
[5,100 B]
Get:28 http://archive.ubuntu.com/ubuntu jammy/main amd64 rake all 13.0.6-2 [61.7
kB]
Get:29 http://archive.ubuntu.com/ubuntu jammy/main amd64 ruby-net-telnet all
0.1.1-2 [12.6 kB]
Get:30 http://archive.ubuntu.com/ubuntu jammy/universe amd64 ruby-webrick all
1.7.0-3 [51.8 kB]
Get:31 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 ruby-xmlrpc all
0.3.2-1ubuntu0.1 [24.9 kB]
Get:32 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libruby3.0
amd64 3.0.2-7ubuntu2.7 [5,113 kB]
Get:33 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libsynctex2
amd64 2021.20210626.59705-1ubuntu0.2 [55.6 kB]
Get:34 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libteckit0 amd64
2.5.11+ds1-1 [421 kB]
Get:35 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libtexlua53
amd64 2021.20210626.59705-1ubuntu0.2 [120 kB]
```

```
Get:36 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libtexluajit2
amd64 2021.20210626.59705-1ubuntu0.2 [267 kB]
Get:37 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libzzip-0-13 amd64
0.13.72+dfsg.1-1.1 [27.0 kB]
Get:38 http://archive.ubuntu.com/ubuntu jammy/main amd64 xfonts-encodings all
1:1.0.5-0ubuntu2 [578 kB]
Get:39 http://archive.ubuntu.com/ubuntu jammy/main amd64 xfonts-utils amd64
1:7.7+6build2 [94.6 kB]
Get:40 http://archive.ubuntu.com/ubuntu jammy/universe amd64 lmodern all
2.004.5-6.1 [9,471 kB]
Get:41 http://archive.ubuntu.com/ubuntu jammy/universe amd64 pandoc-data all
2.9.2.1-3ubuntu2 [81.8 kB]
Get:42 http://archive.ubuntu.com/ubuntu jammy/universe amd64 pandoc amd64
2.9.2.1-3ubuntu2 [20.3 MB]
Get:43 http://archive.ubuntu.com/ubuntu jammy/universe amd64 preview-latex-style
all 12.2-1ubuntu1 [185 kB]
Get:44 http://archive.ubuntu.com/ubuntu jammy/main amd64 t1utils amd64
1.41-4build2 [61.3 kB]
Get:45 http://archive.ubuntu.com/ubuntu jammy/universe amd64 teckit amd64
2.5.11+ds1-1 [699 kB]
Get:46 http://archive.ubuntu.com/ubuntu jammy/universe amd64 tex-gyre all
20180621-3.1 [6,209 kB]
Get:47 http://archive.ubuntu.com/ubuntu jammy-updates/universe amd64 texlive-
binaries amd64 2021.20210626.59705-1ubuntu0.2 [9,860 kB]
Get:48 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-base all
2021.20220204-1 [21.0 MB]
Get:49 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-fonts-
recommended all 2021.20220204-1 [4,972 kB]
Get:50 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-latex-base
all 2021.20220204-1 [1,128 kB]
Get:51 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-latex-
recommended all 2021.20220204-1 [14.4 MB]
Get:52 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive all
2021.20220204-1 [14.3 kB]
Get:53 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libfontbox-java all
1:1.8.16-2 [207 kB]
Get:54 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libpdfbox-java all
1:1.8.16-2 [5,199 kB]
Get:55 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-pictures
all 2021.20220204-1 [8,720 kB]
Get:56 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-latex-extra
all 2021.20220204-1 [13.9 MB]
Get:57 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-plain-
generic all 2021.20220204-1 [27.5 MB]
Get:58 http://archive.ubuntu.com/ubuntu jammy/universe amd64 tipa all 2:1.3-21
[2,967 kB]
Get:59 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-xetex all
2021.20220204-1 [12.4 MB]
```

```
Fetched 202 MB in 8s (26.5 MB/s)
Extracting templates from packages: 100%
Preconfiguring packages …
Selecting previously unselected package fonts-droid-fallback.
(Reading database … 123605 files and directories currently installed.)
Preparing to unpack …/00-fonts-droid-fallback_1%3a6.0.1r16-1.1build1_all.deb
…
Unpacking fonts-droid-fallback (1:6.0.1r16-1.1build1) …
Selecting previously unselected package fonts-lato.
Preparing to unpack …/01-fonts-lato_2.0-2.1_all.deb …
Unpacking fonts-lato (2.0-2.1) …
Selecting previously unselected package poppler-data.
Preparing to unpack …/02-poppler-data_0.4.11-1_all.deb …
Unpacking poppler-data (0.4.11-1) …
Selecting previously unselected package tex-common.
Preparing to unpack …/03-tex-common_6.17_all.deb …
Unpacking tex-common (6.17) …
Selecting previously unselected package fonts-urw-base35.
Preparing to unpack …/04-fonts-urw-base35_20200910-1_all.deb …
Unpacking fonts-urw-base35 (20200910-1) …
Selecting previously unselected package libgs9-common.
Preparing to unpack …/05-libgs9-common_9.55.0~dfsg1-0ubuntu5.9_all.deb …
Unpacking libgs9-common (9.55.0~dfsg1-0ubuntu5.9) …
Selecting previously unselected package libidn12:amd64.
Preparing to unpack …/06-libidn12_1.38-4ubuntu1_amd64.deb …
Unpacking libidn12:amd64 (1.38-4ubuntu1) …
Selecting previously unselected package libijs-0.35:amd64.
Preparing to unpack …/07-libijs-0.35_0.35-15build2_amd64.deb …
Unpacking libijs-0.35:amd64 (0.35-15build2) …
Selecting previously unselected package libjbig2dec0:amd64.
Preparing to unpack …/08-libjbig2dec0_0.19-3build2_amd64.deb …
Unpacking libjbig2dec0:amd64 (0.19-3build2) …
Selecting previously unselected package libgs9:amd64.
Preparing to unpack …/09-libgs9_9.55.0~dfsg1-0ubuntu5.9_amd64.deb …
Unpacking libgs9:amd64 (9.55.0~dfsg1-0ubuntu5.9) …
Selecting previously unselected package libkpathsea6:amd64.
Preparing to unpack …/10-libkpathsea6_2021.20210626.59705-1ubuntu0.2_amd64.deb
…
Unpacking libkpathsea6:amd64 (2021.20210626.59705-1ubuntu0.2) …
Selecting previously unselected package libwoff1:amd64.
Preparing to unpack …/11-libwoff1_1.0.2-1build4_amd64.deb …
Unpacking libwoff1:amd64 (1.0.2-1build4) …
Selecting previously unselected package dvisvgm.
Preparing to unpack …/12-dvisvgm_2.13.1-1_amd64.deb …
Unpacking dvisvgm (2.13.1-1) …
Selecting previously unselected package fonts-lmodern.
Preparing to unpack …/13-fonts-lmodern_2.004.5-6.1_all.deb …
Unpacking fonts-lmodern (2.004.5-6.1) …
```

```
Selecting previously unselected package fonts-noto-mono.
Preparing to unpack …/14-fonts-noto-mono_20201225-1build1_all.deb …
Unpacking fonts-noto-mono (20201225-1build1) …
Selecting previously unselected package fonts-texgyre.
Preparing to unpack …/15-fonts-texgyre_20180621-3.1_all.deb …
Unpacking fonts-texgyre (20180621-3.1) …
Selecting previously unselected package libapache-pom-java.
Preparing to unpack …/16-libapache-pom-java_18-1_all.deb …
Unpacking libapache-pom-java (18-1) …
Selecting previously unselected package libcmark-gfm0.29.0.gfm.3:amd64.
Preparing to unpack …/17-libcmark-gfm0.29.0.gfm.3_0.29.0.gfm.3-3_amd64.deb …
Unpacking libcmark-gfm0.29.0.gfm.3:amd64 (0.29.0.gfm.3-3) …
Selecting previously unselected package libcmark-gfm-
extensions0.29.0.gfm.3:amd64.
Preparing to unpack …/18-libcmark-gfm-
extensions0.29.0.gfm.3_0.29.0.gfm.3-3_amd64.deb …
Unpacking libcmark-gfm-extensions0.29.0.gfm.3:amd64 (0.29.0.gfm.3-3) …
Selecting previously unselected package libcommons-parent-java.
Preparing to unpack …/19-libcommons-parent-java_43-1_all.deb …
Unpacking libcommons-parent-java (43-1) …
Selecting previously unselected package libcommons-logging-java.
Preparing to unpack …/20-libcommons-logging-java_1.2-2_all.deb …
Unpacking libcommons-logging-java (1.2-2) …
Selecting previously unselected package libfontenc1:amd64.
Preparing to unpack …/21-libfontenc1_1%3a1.1.4-1build3_amd64.deb …
Unpacking libfontenc1:amd64 (1:1.1.4-1build3) …
Selecting previously unselected package libptexenc1:amd64.
Preparing to unpack …/22-libptexenc1_2021.20210626.59705-1ubuntu0.2_amd64.deb
…
Unpacking libptexenc1:amd64 (2021.20210626.59705-1ubuntu0.2) …
Selecting previously unselected package rubygems-integration.
Preparing to unpack …/23-rubygems-integration_1.18_all.deb …
Unpacking rubygems-integration (1.18) …
Selecting previously unselected package ruby3.0.
Preparing to unpack …/24-ruby3.0_3.0.2-7ubuntu2.7_amd64.deb …
Unpacking ruby3.0 (3.0.2-7ubuntu2.7) …
Selecting previously unselected package ruby-rubygems.
Preparing to unpack …/25-ruby-rubygems_3.3.5-2_all.deb …
Unpacking ruby-rubygems (3.3.5-2) …
Selecting previously unselected package ruby.
Preparing to unpack …/26-ruby_1%3a3.0~exp1_amd64.deb …
Unpacking ruby (1:3.0~exp1) …
Selecting previously unselected package rake.
Preparing to unpack …/27-rake_13.0.6-2_all.deb …
Unpacking rake (13.0.6-2) …
Selecting previously unselected package ruby-net-telnet.
Preparing to unpack …/28-ruby-net-telnet_0.1.1-2_all.deb …
Unpacking ruby-net-telnet (0.1.1-2) …
```

```
Selecting previously unselected package ruby-webrick.
Preparing to unpack …/29-ruby-webrick_1.7.0-3_all.deb …
Unpacking ruby-webrick (1.7.0-3) …
Selecting previously unselected package ruby-xmlrpc.
Preparing to unpack …/30-ruby-xmlrpc_0.3.2-1ubuntu0.1_all.deb …
Unpacking ruby-xmlrpc (0.3.2-1ubuntu0.1) …
Selecting previously unselected package libruby3.0:amd64.
Preparing to unpack …/31-libruby3.0_3.0.2-7ubuntu2.7_amd64.deb …
Unpacking libruby3.0:amd64 (3.0.2-7ubuntu2.7) …
Selecting previously unselected package libsynctex2:amd64.
Preparing to unpack …/32-libsynctex2_2021.20210626.59705-1ubuntu0.2_amd64.deb
…
Unpacking libsynctex2:amd64 (2021.20210626.59705-1ubuntu0.2) …
Selecting previously unselected package libteckit0:amd64.
Preparing to unpack …/33-libteckit0_2.5.11+ds1-1_amd64.deb …
Unpacking libteckit0:amd64 (2.5.11+ds1-1) …
Selecting previously unselected package libtexlua53:amd64.
Preparing to unpack …/34-libtexlua53_2021.20210626.59705-1ubuntu0.2_amd64.deb
…
Unpacking libtexlua53:amd64 (2021.20210626.59705-1ubuntu0.2) …
Selecting previously unselected package libtexluajit2:amd64.
Preparing to unpack
…/35-libtexluajit2_2021.20210626.59705-1ubuntu0.2_amd64.deb …
Unpacking libtexluajit2:amd64 (2021.20210626.59705-1ubuntu0.2) …
Selecting previously unselected package libzzip-0-13:amd64.
Preparing to unpack …/36-libzzip-0-13_0.13.72+dfsg.1-1.1_amd64.deb …
Unpacking libzzip-0-13:amd64 (0.13.72+dfsg.1-1.1) …
Selecting previously unselected package xfonts-encodings.
Preparing to unpack …/37-xfonts-encodings_1%3a1.0.5-0ubuntu2_all.deb …
Unpacking xfonts-encodings (1:1.0.5-0ubuntu2) …
Selecting previously unselected package xfonts-utils.
Preparing to unpack …/38-xfonts-utils_1%3a7.7+6build2_amd64.deb …
Unpacking xfonts-utils (1:7.7+6build2) …
Selecting previously unselected package lmodern.
Preparing to unpack …/39-lmodern_2.004.5-6.1_all.deb …
Unpacking lmodern (2.004.5-6.1) …
Selecting previously unselected package pandoc-data.
Preparing to unpack …/40-pandoc-data_2.9.2.1-3ubuntu2_all.deb …
Unpacking pandoc-data (2.9.2.1-3ubuntu2) …
Selecting previously unselected package pandoc.
Preparing to unpack …/41-pandoc_2.9.2.1-3ubuntu2_amd64.deb …
Unpacking pandoc (2.9.2.1-3ubuntu2) …
Selecting previously unselected package preview-latex-style.
Preparing to unpack …/42-preview-latex-style_12.2-1ubuntu1_all.deb …
Unpacking preview-latex-style (12.2-1ubuntu1) …
Selecting previously unselected package t1utils.
Preparing to unpack …/43-t1utils_1.41-4build2_amd64.deb …
Unpacking t1utils (1.41-4build2) …
```

```
Selecting previously unselected package teckit.
Preparing to unpack …/44-teckit_2.5.11+ds1-1_amd64.deb …
Unpacking teckit (2.5.11+ds1-1) …
Selecting previously unselected package tex-gyre.
Preparing to unpack …/45-tex-gyre_20180621-3.1_all.deb …
Unpacking tex-gyre (20180621-3.1) …
Selecting previously unselected package texlive-binaries.
Preparing to unpack …/46-texlive-
binaries_2021.20210626.59705-1ubuntu0.2_amd64.deb …
Unpacking texlive-binaries (2021.20210626.59705-1ubuntu0.2) …
Selecting previously unselected package texlive-base.
Preparing to unpack …/47-texlive-base_2021.20220204-1_all.deb …
Unpacking texlive-base (2021.20220204-1) …
Selecting previously unselected package texlive-fonts-recommended.
Preparing to unpack …/48-texlive-fonts-recommended_2021.20220204-1_all.deb …
Unpacking texlive-fonts-recommended (2021.20220204-1) …
Selecting previously unselected package texlive-latex-base.
Preparing to unpack …/49-texlive-latex-base_2021.20220204-1_all.deb …
Unpacking texlive-latex-base (2021.20220204-1) …
Selecting previously unselected package texlive-latex-recommended.
Preparing to unpack …/50-texlive-latex-recommended_2021.20220204-1_all.deb …
Unpacking texlive-latex-recommended (2021.20220204-1) …
Selecting previously unselected package texlive.
Preparing to unpack …/51-texlive_2021.20220204-1_all.deb …
Unpacking texlive (2021.20220204-1) …
Selecting previously unselected package libfontbox-java.
Preparing to unpack …/52-libfontbox-java_1%3a1.8.16-2_all.deb …
Unpacking libfontbox-java (1:1.8.16-2) …
Selecting previously unselected package libpdfbox-java.
Preparing to unpack …/53-libpdfbox-java_1%3a1.8.16-2_all.deb …
Unpacking libpdfbox-java (1:1.8.16-2) …
Selecting previously unselected package texlive-pictures.
Preparing to unpack …/54-texlive-pictures_2021.20220204-1_all.deb …
Unpacking texlive-pictures (2021.20220204-1) …
Selecting previously unselected package texlive-latex-extra.
Preparing to unpack …/55-texlive-latex-extra_2021.20220204-1_all.deb …
Unpacking texlive-latex-extra (2021.20220204-1) …
Selecting previously unselected package texlive-plain-generic.
Preparing to unpack …/56-texlive-plain-generic_2021.20220204-1_all.deb …
Unpacking texlive-plain-generic (2021.20220204-1) …
Selecting previously unselected package tipa.
Preparing to unpack …/57-tipa_2%3a1.3-21_all.deb …
Unpacking tipa (2:1.3-21) …
Selecting previously unselected package texlive-xetex.
Preparing to unpack …/58-texlive-xetex_2021.20220204-1_all.deb …
Unpacking texlive-xetex (2021.20220204-1) …
Setting up fonts-lato (2.0-2.1) …
Setting up fonts-noto-mono (20201225-1build1) …
Setting up libwoff1:amd64 (1.0.2-1build4) …
Setting up libtexlua53:amd64 (2021.20210626.59705-1ubuntu0.2) …
Setting up libijs-0.35:amd64 (0.35-15build2) …
Setting up libtexluajit2:amd64 (2021.20210626.59705-1ubuntu0.2) …
Setting up libfontbox-java (1:1.8.16-2) …
Setting up rubygems-integration (1.18) …
Setting up libzzip-0-13:amd64 (0.13.72+dfsg.1-1.1) …
Setting up fonts-urw-base35 (20200910-1) …
Setting up poppler-data (0.4.11-1) …
Setting up tex-common (6.17) …
update-language: texlive-base not installed and configured, doing nothing!
Setting up libfontenc1:amd64 (1:1.1.4-1build3) …
Setting up libjbig2dec0:amd64 (0.19-3build2) …
Setting up libteckit0:amd64 (2.5.11+ds1-1) …
Setting up libapache-pom-java (18-1) …
Setting up ruby-net-telnet (0.1.1-2) …
Setting up xfonts-encodings (1:1.0.5-0ubuntu2) …
Setting up t1utils (1.41-4build2) …
Setting up libidn12:amd64 (1.38-4ubuntu1) …
Setting up fonts-texgyre (20180621-3.1) …
Setting up libkpathsea6:amd64 (2021.20210626.59705-1ubuntu0.2) …
Setting up ruby-webrick (1.7.0-3) …
Setting up libcmark-gfm0.29.0.gfm.3:amd64 (0.29.0.gfm.3-3) …
Setting up fonts-lmodern (2.004.5-6.1) …
Setting up libcmark-gfm-extensions0.29.0.gfm.3:amd64 (0.29.0.gfm.3-3) …
Setting up fonts-droid-fallback (1:6.0.1r16-1.1build1) …
Setting up pandoc-data (2.9.2.1-3ubuntu2) …
Setting up ruby-xmlrpc (0.3.2-1ubuntu0.1) …
Setting up libsynctex2:amd64 (2021.20210626.59705-1ubuntu0.2) …
Setting up libgs9-common (9.55.0~dfsg1-0ubuntu5.9) …
Setting up teckit (2.5.11+ds1-1) …
Setting up libpdfbox-java (1:1.8.16-2) …
Setting up libgs9:amd64 (9.55.0~dfsg1-0ubuntu5.9) …
Setting up preview-latex-style (12.2-1ubuntu1) …
Setting up libcommons-parent-java (43-1) …
Setting up dvisvgm (2.13.1-1) …
Setting up libcommons-logging-java (1.2-2) …
Setting up xfonts-utils (1:7.7+6build2) …
Setting up libptexenc1:amd64 (2021.20210626.59705-1ubuntu0.2) …
Setting up pandoc (2.9.2.1-3ubuntu2) …
Setting up texlive-binaries (2021.20210626.59705-1ubuntu0.2) …
update-alternatives: using /usr/bin/xdvi-xaw to provide /usr/bin/xdvi.bin
(xdvi.bin) in auto mode
update-alternatives: using /usr/bin/bibtex.original to provide /usr/bin/bibtex
(bibtex) in auto mode
Setting up lmodern (2.004.5-6.1) …
Setting up texlive-base (2021.20220204-1) …
/usr/bin/ucfr
```

```
/usr/bin/ucfr
/usr/bin/ucfr
/usr/bin/ucfr
mktexlsr: Updating /var/lib/texmf/ls-R-TEXLIVEDIST…
mktexlsr: Updating /var/lib/texmf/ls-R-TEXMFMAIN…
mktexlsr: Updating /var/lib/texmf/ls-R…
mktexlsr: Done.
tl-paper: setting paper size for dvips to a4:
/var/lib/texmf/dvips/config/config-paper.ps
tl-paper: setting paper size for dvipdfmx to a4:
/var/lib/texmf/dvipdfmx/dvipdfmx-paper.cfg
tl-paper: setting paper size for xdvi to a4: /var/lib/texmf/xdvi/XDvi-paper
tl-paper: setting paper size for pdftex to a4: /var/lib/texmf/tex/generic/tex-
ini-files/pdftexconfig.tex
Setting up tex-gyre (20180621-3.1) …
Setting up texlive-plain-generic (2021.20220204-1) …
Setting up texlive-latex-base (2021.20220204-1) …
Setting up texlive-latex-recommended (2021.20220204-1) …
Setting up texlive-pictures (2021.20220204-1) …
Setting up texlive-fonts-recommended (2021.20220204-1) …
Setting up tipa (2:1.3-21) …
Setting up texlive (2021.20220204-1) …
Setting up texlive-latex-extra (2021.20220204-1) …
Setting up texlive-xetex (2021.20220204-1) …
Setting up rake (13.0.6-2) …
Setting up libruby3.0:amd64 (3.0.2-7ubuntu2.7) …
Setting up ruby3.0 (3.0.2-7ubuntu2.7) …
Setting up ruby (1:3.0~exp1) …
Setting up ruby-rubygems (3.3.5-2) …
Processing triggers for man-db (2.10.2-1) …
Processing triggers for fontconfig (2.13.1-4.2ubuntu5) …
Processing triggers for libc-bin (2.35-0ubuntu3.4) …
/sbin/ldconfig.real: /usr/local/lib/libur_loader.so.0 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbmalloc_proxy.so.2 is not a symbolic
link

/sbin/ldconfig.real: /usr/local/lib/libtbbmalloc.so.2 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libur_adapter_opencl.so.0 is not a symbolic
link

/sbin/ldconfig.real: /usr/local/lib/libtbbbind_2_5.so.3 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libur_adapter_level_zero.so.0 is not a
symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbbind.so.3 is not a symbolic link
```

```
/sbin/ldconfig.real: /usr/local/lib/libtbb.so.12 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbbind_2_0.so.3 is not a symbolic link

Processing triggers for tex-common (6.17) …
Running updmap-sys. This may take some time… done.
Running mktexlsr /var/lib/texmf … done.
Building format(s) --all.
        This may take some time… done.
Collecting pypandoc
  Downloading pypandoc-1.13-py3-none-any.whl.metadata (16 kB)
Downloading pypandoc-1.13-py3-none-any.whl (21 kB)
Installing collected packages: pypandoc
Successfully installed pypandoc-1.13
```

[47]:
```
!cp /content/drive/MyDrive/ADS-509-01/Module3/Group Comparison.ipynb
```

```
cp: cannot stat '/content/drive/MyDrive/ADS-509-01/Module3/Group': No such file
or directory
```

[49]:
```
!jupyter nbconvert --to PDF "Group Comparison.ipynb"
```

```
[NbConvertApp] WARNING | pattern 'Group Comparison.ipynb' matched no files
This application is used to convert notebook files (*.ipynb)
        to various other formats.

        WARNING: THE COMMANDLINE INTERFACE MAY CHANGE IN FUTURE RELEASES.

Options
=======
The options below are convenience aliases to configurable class-options,
as listed in the "Equivalent to" description-line of the aliases.
To see all configurable class-options for some <cmd>, use:
    <cmd> --help-all

--debug
    set log level to logging.DEBUG (maximize logging output)
    Equivalent to: [--Application.log_level=10]
--show-config
    Show the application's configuration (human-readable format)
    Equivalent to: [--Application.show_config=True]
--show-config-json
    Show the application's configuration (json format)
    Equivalent to: [--Application.show_config_json=True]
--generate-config
    generate default config file
    Equivalent to: [--JupyterApp.generate_config=True]
```

```
-y
    Answer yes to any questions instead of prompting.
    Equivalent to: [--JupyterApp.answer_yes=True]
--execute
    Execute the notebook prior to export.
    Equivalent to: [--ExecutePreprocessor.enabled=True]
--allow-errors
    Continue notebook execution even if one of the cells throws an error and
include the error message in the cell output (the default behaviour is to abort
conversion). This flag is only relevant if '--execute' was specified, too.
    Equivalent to: [--ExecutePreprocessor.allow_errors=True]
--stdin
    read a single notebook file from stdin. Write the resulting notebook with
default basename 'notebook.*'
    Equivalent to: [--NbConvertApp.from_stdin=True]
--stdout
    Write notebook output to stdout instead of files.
    Equivalent to: [--NbConvertApp.writer_class=StdoutWriter]
--inplace
    Run nbconvert in place, overwriting the existing notebook (only
            relevant when converting to notebook format)
    Equivalent to: [--NbConvertApp.use_output_suffix=False
--NbConvertApp.export_format=notebook --FilesWriter.build_directory=]
--clear-output
    Clear output of current file and save in place,
            overwriting the existing notebook.
    Equivalent to: [--NbConvertApp.use_output_suffix=False
--NbConvertApp.export_format=notebook --FilesWriter.build_directory=
--ClearOutputPreprocessor.enabled=True]
--no-prompt
    Exclude input and output prompts from converted document.
    Equivalent to: [--TemplateExporter.exclude_input_prompt=True
--TemplateExporter.exclude_output_prompt=True]
--no-input
    Exclude input cells and output prompts from converted document.
            This mode is ideal for generating code-free reports.
    Equivalent to: [--TemplateExporter.exclude_output_prompt=True
--TemplateExporter.exclude_input=True
--TemplateExporter.exclude_input_prompt=True]
--allow-chromium-download
    Whether to allow downloading chromium if no suitable version is found on the
system.
    Equivalent to: [--WebPDFExporter.allow_chromium_download=True]
--disable-chromium-sandbox
    Disable chromium security sandbox when converting to PDF..
    Equivalent to: [--WebPDFExporter.disable_sandbox=True]
--show-input
    Shows code input. This flag is only useful for dejavu users.
    Equivalent to: [--TemplateExporter.exclude_input=False]
--embed-images
    Embed the images as base64 dataurls in the output. This flag is only useful
for the HTML/WebPDF/Slides exports.
    Equivalent to: [--HTMLExporter.embed_images=True]
--sanitize-html
    Whether the HTML in Markdown cells and cell outputs should be sanitized..
    Equivalent to: [--HTMLExporter.sanitize_html=True]
--log-level=<Enum>
    Set the log level by value or name.
    Choices: any of [0, 10, 20, 30, 40, 50, 'DEBUG', 'INFO', 'WARN', 'ERROR',
'CRITICAL']
    Default: 30
    Equivalent to: [--Application.log_level]
--config=<Unicode>
    Full path of a config file.
    Default: ''
    Equivalent to: [--JupyterApp.config_file]
--to=<Unicode>
    The export format to be used, either one of the built-in formats
            ['asciidoc', 'custom', 'html', 'latex', 'markdown', 'notebook',
'pdf', 'python', 'rst', 'script', 'slides', 'webpdf']
            or a dotted object name that represents the import path for an
            ``Exporter`` class
    Default: ''
    Equivalent to: [--NbConvertApp.export_format]
--template=<Unicode>
    Name of the template to use
    Default: ''
    Equivalent to: [--TemplateExporter.template_name]
--template-file=<Unicode>
    Name of the template file to use
    Default: None
    Equivalent to: [--TemplateExporter.template_file]
--theme=<Unicode>
    Template specific theme(e.g. the name of a JupyterLab CSS theme distributed
    as prebuilt extension for the lab template)
    Default: 'light'
    Equivalent to: [--HTMLExporter.theme]
--sanitize_html=<Bool>
    Whether the HTML in Markdown cells and cell outputs should be sanitized.This
    should be set to True by nbviewer or similar tools.
    Default: False
    Equivalent to: [--HTMLExporter.sanitize_html]
--writer=<DottedObjectName>
    Writer class used to write the
                                        results of the conversion
    Default: 'FilesWriter'
    Equivalent to: [--NbConvertApp.writer_class]
--post=<DottedOrNone>
    PostProcessor class used to write the
                                        results of the conversion
    Default: ''
    Equivalent to: [--NbConvertApp.postprocessor_class]
--output=<Unicode>
    overwrite base name use for output files.
                had only be used when converting one notebook at a time.
    Default: ''
    Equivalent to: [--NbConvertApp.output_base]
--output-dir=<Unicode>
    Directory to write output(s) to. Defaults
                                  to output to the directory of each notebook.
To recover
                                  previous default behaviour (outputting to the
current
                                  working directory) use . as the flag value.
    Default: ''
    Equivalent to: [--FilesWriter.build_directory]
--reveal-prefix=<Unicode>
    The URL prefix for reveal.js (version 3.x).
            This defaults to the reveal CDN, but can be any url pointing to a
copy
            of reveal.js.
            For speaker notes to work, this must be a relative path to a local
            copy of reveal.js: e.g., "reveal.js".
            If a relative path is given, it must be a subdirectory of the
            current directory (from which the server is run).
            See the usage documentation
            (https://nbconvert.readthedocs.io/en/latest/usage.html#reveal-js-
html-slideshow)
            for more details.
    Default: ''
    Equivalent to: [--SlidesExporter.reveal_url_prefix]
--nbformat=<Enum>
    The nbformat version to write.
            Use this to downgrade notebooks.
    Choices: any of [1, 2, 3, 4]
    Default: 4
    Equivalent to: [--NotebookExporter.nbformat_version]

Examples
--------

    The simplest way to use nbconvert is

            > jupyter nbconvert mynotebook.ipynb --to html
            Options include ['asciidoc', 'custom', 'html', 'latex', 'markdown',
'notebook', 'pdf', 'python', 'rst', 'script', 'slides', 'webpdf'].

            > jupyter nbconvert --to latex mynotebook.ipynb

            Both HTML and LaTeX support multiple output templates. LaTeX
includes
            'base', 'article' and 'report'.  HTML includes 'basic', 'lab' and
            'classic'. You can specify the flavor of the format used.

            > jupyter nbconvert --to html --template lab mynotebook.ipynb

            You can also pipe the output to stdout, rather than a file

            > jupyter nbconvert mynotebook.ipynb --stdout

            PDF is generated via latex

            > jupyter nbconvert mynotebook.ipynb --to pdf

            You can get (and serve) a Reveal.js-powered slideshow

            > jupyter nbconvert myslides.ipynb --to slides --post serve

            Multiple notebooks can be given at the command line in a couple of
            different ways:

            > jupyter nbconvert notebook*.ipynb
            > jupyter nbconvert notebook1.ipynb notebook2.ipynb

            or you can specify the notebooks list in a config file, containing::

                c.NbConvertApp.notebooks = ["my_notebook.ipynb"]

            > jupyter nbconvert --config mycfg.py

To see all available configurables, use `--help-all`.
```