Taylor Swift Spotify Data Analysis

taylor-swift-eras-519-2023-billboard-1548.webp

Taylor Swift Spotify Data is a dataset of songs on Taylor Swift's albums, excluding studio sessions and feature songs. I found this set on Kaggle through the user Jan Llenzl Dagohoy, and its last update was 2 years ago (Fearless (Taylor's Version) Era). The dataset has 16 columns and over 150 rows of data on her statistics. I plan to analyze relationships between the given data and tie them to her career in the industry.

I've chosen to perform my data analysis with this set because my career goals include working in the music industry by utilizing my technology and business skills.

Downloading the Dataset

I downloaded my dataset through Kaggle, and accessed the directory through my Jupyter Notebook.

```
!pip install jovian opendatasets --upgrade --quiet
```

Let's begin by downloading the data, and listing the files within the dataset.

```
dataset_url = 'https://www.kaggle.com/datasets/thespacefreak/taylor-swift-spotify-data'
```

```
import opendatasets as od
od.download(dataset_url)
```

```
Skipping, found downloaded files in "./taylor-swift-spotify-data" (use force=True to
force download)
```

The dataset has been downloaded and extracted.

```
# Change this
data_dir = './taylor-swift-spotify-data'
```

```
import os
os.listdir(data_dir)
```

```
['spotify_taylorswift.csv']
```

Let us save and upload our work to Jovian before continuing.

```
project_name = "taylor-swift-spotify-data-analysis"
```

```
!pip install jovian --upgrade -q
```

```
import jovian
```

```
jovian.commit(project=project_name)
```

```
[jovian] Updating notebook "graceseliou/taylor-swift-spotify-data-analysis" on
https://jovian.com
[jovian] Committed successfully! https://jovian.com/graceseliou/taylor-swift-spotify-
data-analysis
```

```
'https://jovian.com/graceseliou/taylor-swift-spotify-data-analysis'
```

Data Preparation and Cleaning

Below, I load the dataset into a dataframe (stats_raw_df) using Pandas, explore the number of rows and columns, ranges of values, etc., handle missing, incorrect and invalid data, and perform some additional steps.

```
import pandas as pd
```

```
stats_raw_df = pd.read_csv(data_dir + "/spotify_taylorswift.csv")
```

```
stats_raw_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 171 entries, 0 to 170
Data columns (total 16 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Unnamed: 0        171 non-null    int64  
 1   name              171 non-null    object 
 2   album             171 non-null    object 
 3   artist            171 non-null    object 
 4   release_date      171 non-null    object 
 5   length            171 non-null    int64  
 6   popularity        171 non-null    int64  
 7   danceability      171 non-null    float64
 8   acousticness      171 non-null    float64
 9   energy            171 non-null    float64
 10  instrumentalness  171 non-null    float64
 11  liveness          171 non-null    float64
 12  loudness          171 non-null    float64
 13  speechiness       171 non-null    float64
 14  valence           171 non-null    float64
 15  tempo             171 non-null    float64
dtypes: float64(9), int64(3), object(4)
memory usage: 21.5+ KB
```

This gives us the columns in the dataset and the data types within each column.

```
stats_raw_df.describe()
```

	Unnamed: 0	length	popularity	danceability	acousticness	energy	instrumentalness	livene
count	171.000000	171.000000	171.000000	171.000000	171.000000	171.000000	171.000000	171.0000
mean	85.000000	236663.520468	61.228070	0.588632	0.321634	0.585977	0.002490	0.1459
std	49.507575	40456.720158	11.904548	0.115067	0.334019	0.189577	0.018766	0.0903
min	0.000000	107133.000000	0.000000	0.292000	0.000191	0.118000	0.000000	0.0335
25%	42.500000	211833.000000	58.000000	0.527000	0.030450	0.462000	0.000000	0.0929
50%	85.000000	234000.000000	63.000000	0.593000	0.156000	0.606000	0.000002	0.1150
75%	127.500000	254447.000000	67.000000	0.655500	0.674000	0.732000	0.000064	0.1680
max	170.000000	403887.000000	82.000000	0.897000	0.971000	0.944000	0.179000	0.6570

This describes some of the data represented in the columns given above.

```
stats_raw_df.isnull().sum()
```

```
Unnamed: 0          0
name                0
album               0
artist              0
release_date        0
length              0
popularity          0
danceability        0
acousticness        0
energy              0
instrumentalness    0
liveness            0
loudness            0
speechiness         0
valence             0
tempo               0
dtype: int64
```

This ensures that there are no null or outlying pieces of data in the dataset that might skew with the following steps.

```
stats_raw_df
```

	Unnamed: 0	name	album	artist	release_date	length	popularity	danceability	acousticness	energy	inst
0	0	Tim McGraw	Taylor Swift	Taylor Swift	2006-10-24	232106	49	0.580	0.575	0.491	
1	1	Picture To Burn	Taylor Swift	Taylor Swift	2006-10-24	173066	54	0.658	0.173	0.877	

	Unnamed: 0	name	album	artist	release_date	length	popularity	danceability	acousticness	energy	inst
2	2	Teardrops On My Guitar - Radio Single Remix	Taylor Swift	Taylor Swift	2006-10-24	203040	59	0.621	0.288	0.417	
3	3	A Place in this World	Taylor Swift	Taylor Swift	2006-10-24	199200	49	0.576	0.051	0.777	
4	4	Cold As You	Taylor Swift	Taylor Swift	2006-10-24	239013	50	0.418	0.217	0.482	
...	...	...	...	...	...	...	...	...	...	...	
166	166	Mr. Perfectly Fine (Taylor’s Version) (From Th...	Fearless (Taylor's Version)	Taylor Swift	2021-04-09	277591	74	0.660	0.162	0.817	
167	167	We Were Happy (Taylor’s Version) (From The Vault)	Fearless (Taylor's Version)	Taylor Swift	2021-04-09	244236	65	0.609	0.849	0.373	
168	168	That’s When (feat. Keith Urban) (Taylor’s Vers...	Fearless (Taylor's Version)	Taylor Swift	2021-04-09	189495	67	0.588	0.225	0.608	
169	169	Don’t You (Taylor’s Version) (From The Vault)	Fearless (Taylor's Version)	Taylor Swift	2021-04-09	208608	66	0.563	0.514	0.473	
170	170	Bye Bye Baby (Taylor’s Version) (From The Vault)	Fearless (Taylor's Version)	Taylor Swift	2021-04-09	242157	64	0.624	0.334	0.624	

171 rows × 16 columns

This gives us the dataframe we'll be working with throughout the project.

Note:

- Length is in milliseconds
- Popularity is curated by Spotify's algorithm (possibly by streams at certain period of time)
- Danceability, acousticness, energy, instrumentalness, liveness, speechiness, and valence are all out of 1
- Tempo is in beats per minute (bpm)

```
import jovian
```

```
jovian.commit()
```

```
[jovian] Updating notebook "graceseliou/taylor-swift-spotify-data-analysis" on
https://jovian.com
[jovian] Committed successfully! https://jovian.com/graceseliou/taylor-swift-spotify-
data-analysis
```

```
'https://jovian.com/graceseliou/taylor-swift-spotify-data-analysis'
```

Exploratory Analysis and Visualization

Below, I perform a brief analysis of different aspects of the dataset.

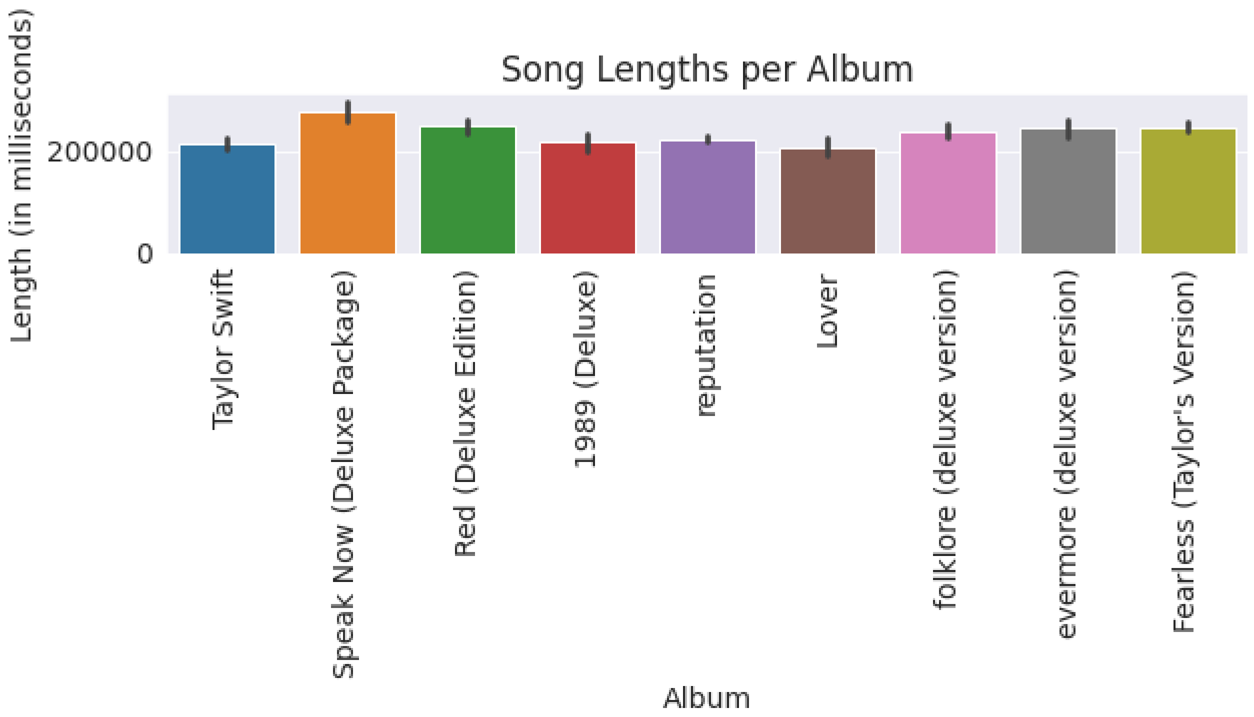
Let's begin by importing `matplotlib.pyplot and seaborn` .

```
import seaborn as sns
import warnings
import matplotlib
import matplotlib.pyplot as plt
%matplotlib inline

sns.set_style('darkgrid')
matplotlib.rcParams['font.size'] = 14
matplotlib.rcParams['figure.figsize'] = (9, 5)
matplotlib.rcParams['figure.facecolor'] = '#00000000'
warnings.filterwarnings("ignore", category = UserWarning)
```

Using a barplot, we can see the lengths of Taylor Swift's songs.

```
sorted_stats_raw_df = stats_raw_df.sort_values(by = "length", ascending = False)
sns.barplot(data = stats_raw_df, x = "album", y = "length")
plt.title("Song Lengths per Album")
plt.xlabel("Album")
plt.ylabel("Length (in milliseconds)")
plt.xticks(rotation = 90)
plt.tight_layout()
plt.show()
```



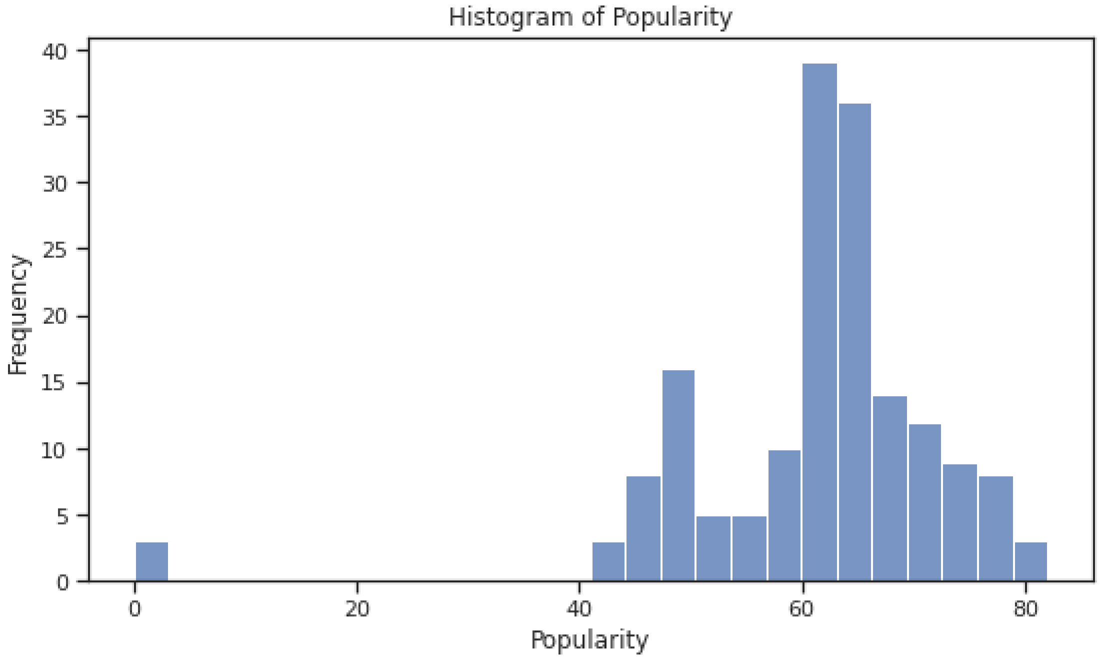
```
stats_raw_df.length.mean()
```

```
236663.52046783626
```

The majority of her songs are about 3.3 minutes long as displayed by the graph, but more specifically, the mean of her song lengths per album is 3.9 minutes long.

Utilizing a histogram, we can see the general popularity of Taylor Swift's songs. Typically, they range from 40-80, but generally around 60. I'm assuming it's the difference between title tracks and b-sides.

```
sns.set(style = "ticks")
sns.histplot(data = stats_raw_df, x = "popularity")
plt.title("Histogram of Popularity")
plt.xlabel("Popularity")
plt.ylabel("Frequency")
plt.show()
```



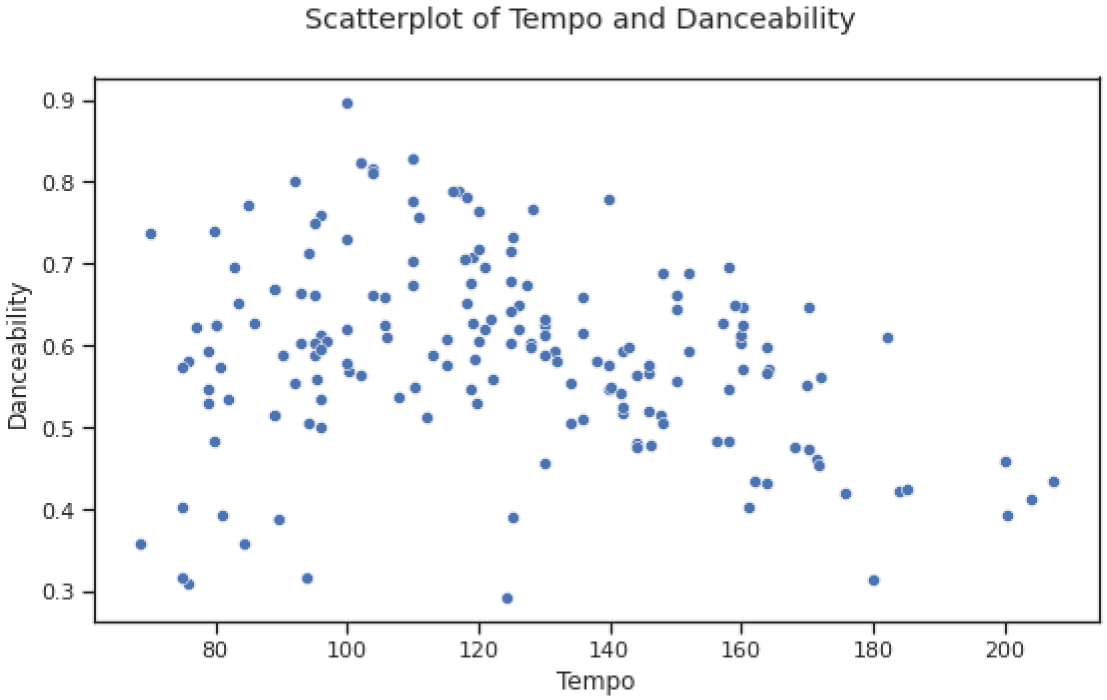
```
stats_raw_df.popularity.mean()
```

```
61.228070175438596
```

Typically, the popularity ranges from 40-80, but generally around 60. I'm assuming it's the difference between title tracks and b-sides.

Utilizing a scatterplot, we can see the relationship between the tempo of Taylor's songs, and their danceability.

```
sns.set(style = "ticks")
sns.scatterplot(data = stats_raw_df, x = "tempo", y = "danceability")
plt.suptitle("Scatterplot of Tempo and Danceability")
plt.xlabel("Tempo")
plt.ylabel("Danceability")
plt.show()
```



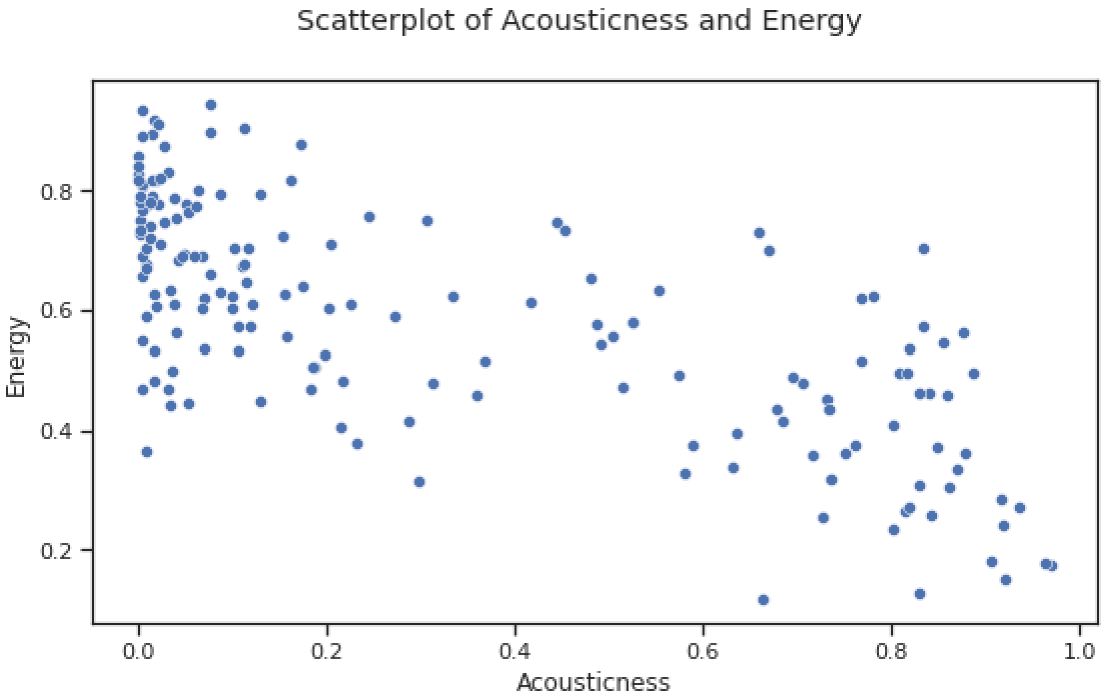
```
print(stats_raw_df.tempo.mean())
print(stats_raw_df.danceability.mean())
```

```
124.14141520467834
0.5886315789473683
```

The scatterplot shows that the majority of her songs range from 100-160 bpm, and the highest danceability is highest at around 100 bpm.

Using a scatterplot, we can see the relationship between acousticness and energy. Because acoustic songs are typically more stripped down, I was curious to see how the energy level would differ between songs.

```
sns.set(style = "ticks")
sns.scatterplot(data = stats_raw_df, x = "acousticness", y = "energy")
plt.suptitle("Scatterplot of Acousticness and Energy")
plt.xlabel("Acousticness")
plt.ylabel("Energy")
plt.show()
```



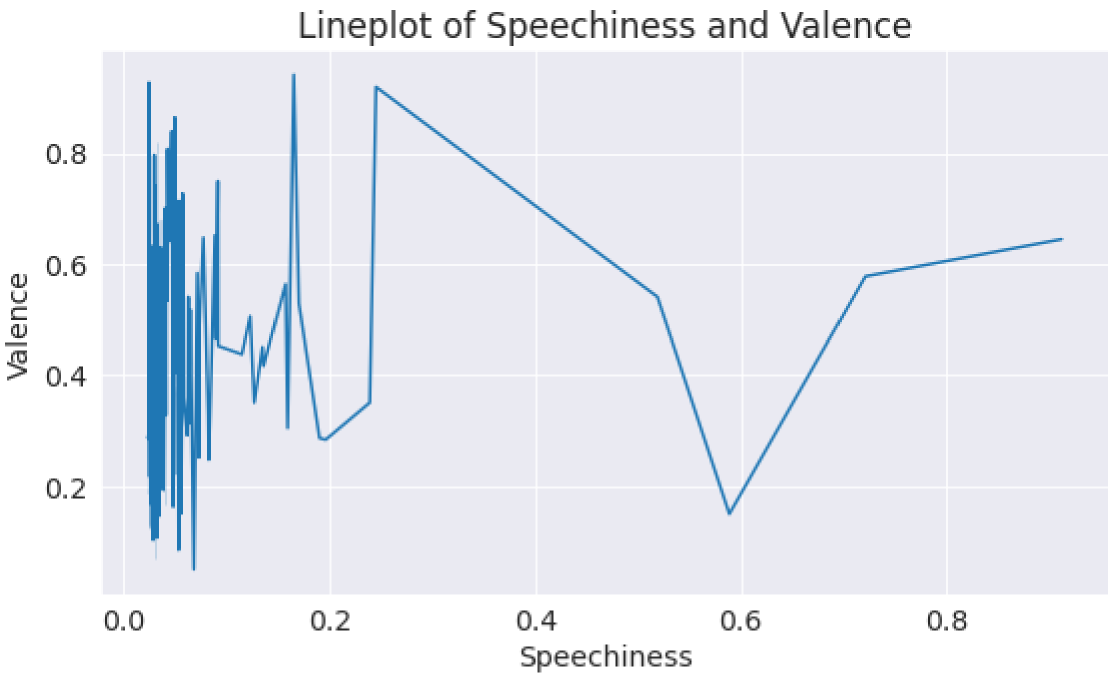
```
print(stats_raw_df.acousticness.mean())
print(stats_raw_df.energy.mean())
```

```
0.3216340116959064
0.5859766081871347
```

The scatterplot shows that as the acousticness increases closer to 1, the energy slowly decreases.

Utilizing a lineplot, we can see the relationship between speechiness and valence in Taylor Swift's songs. In modern music, "mumble music" has made it's impact as giving a "drunk" feel to the artists' music. I was curious to see how the valence of songs in Taylor's music differed from that (or not).

```
sns.lineplot(data = stats_raw_df, x = "speechiness", y = "valence")
plt.title("Lineplot of Speechiness and Valence")
plt.xlabel("Speechiness")
plt.ylabel("Valence")
plt.show()
```



```
print(stats_raw_df.speechiness.mean())
print(stats_raw_df.valence.mean())
```

```
0.06558304093567256
0.4229836257309942
```

The lineplot displays a great range of valence throughout all levels of speechiness, but especially when the speechiness is near 0-0.2.

Let us save and upload our work to Jovian before continuing

```
import jovian
```

```
jovian.commit()
```

```
[jovian] Updating notebook "graceseliou/taylor-swift-spotify-data-analysis" on https://jovian.com
[jovian] Committed successfully! https://jovian.com/graceseliou/taylor-swift-spotify-data-analysis
```

```
'https://jovian.com/graceseliou/taylor-swift-spotify-data-analysis'
```

Asking and Answering Questions

Below, I curate my own questions to answer by utilizing Pandas/Numpy, Seaborn/Matplotlib. I also add onto the dataset by creating new columns that merge multiple datasets and perform deletion/aggregation wherever necessary.

Q1: As it is the only column which has negative values, what is the sum of the loudness of Taylor Swift's songs? Also, are there values other than 0 in the 'instrumentalness' column?

```
stats_raw_df.loudness.sum()
```

```
-1252.081
```

The sum of the loudness of Taylor Swift's songs is -1252.081.

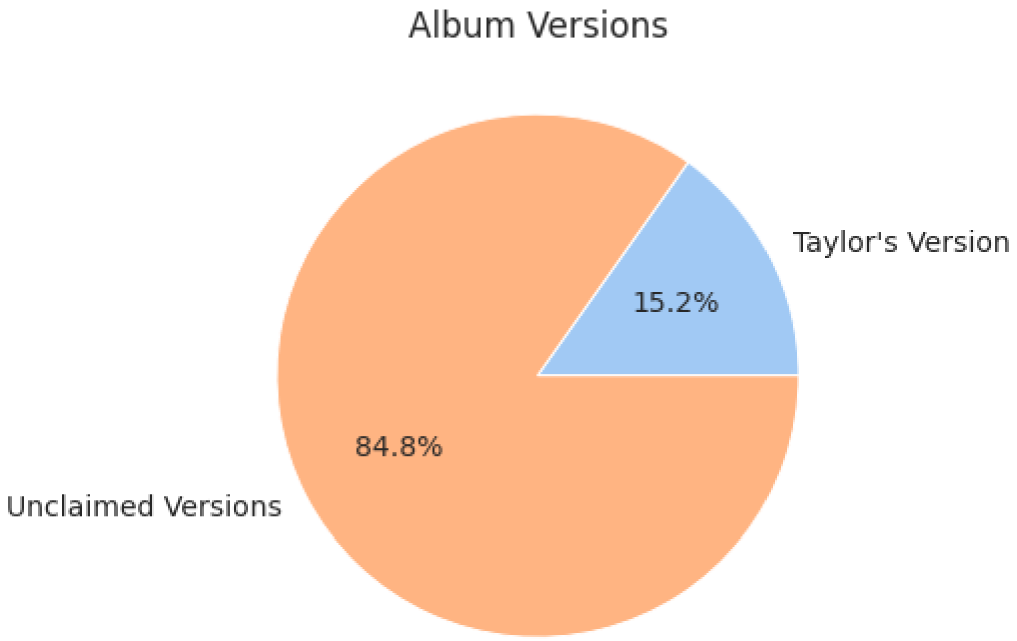
```
stats_raw_df.instrumentalness.sum()
```

```
0.42584526999999994
```

Yes, the column 'instrumentalness' does have values which are not equal to 0.

Q2: How many of Taylor's albums has she reclaimed (featuring Taylor's Version in the album name)?

```
taylors_version = stats_raw_df["album"].str.contains("(Taylor's Version)").sum()
unclaimed_versions = len(stats_raw_df) - taylors_version
counts = [taylors_version, unclaimed_versions]
labels = ["Taylor's Version", "Unclaimed Versions"]
plt.figure(figsize = (6, 6))
sns.set_palette("pastel")
plt.pie(counts, labels = labels, autopct = "%1.1f%%")
plt.title("Album Versions")
plt.ylabel("")
plt.show()
```



Taylor has reclaimed 15.2% of her albums (as of Fearless era; it is definitely higher now)!

Q3: Songs are typically given names that are easy to remember, which is why many songs have shorter names. However, as Taylor has become increasingly popular, and begun fighting for her music, her song titles have become longer (e.g., Taylor's Version at the end of every song name). What is the total mean length of Taylor's song names?

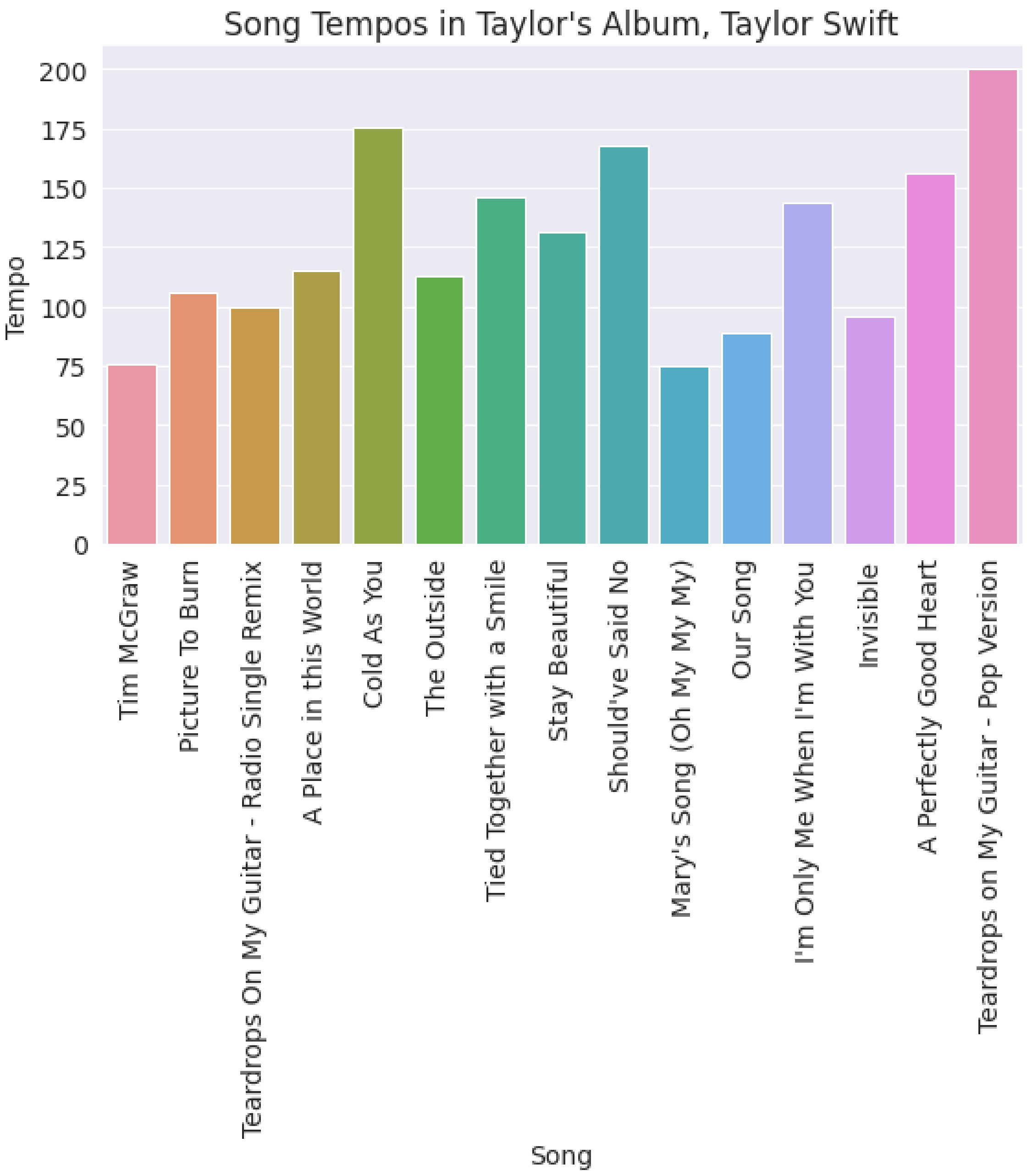
```
song_names = stats_raw_df["name"]
name_lengths = song_names.str.len()
mean_name_length = name_lengths.mean()
print("Song Name Lengths:", name_lengths.tolist())
print("Mean length", mean_name_length)
```

```
Song Name Lengths: [10, 15, 43, 21, 11, 11, 26, 14, 17, 25, 8, 29, 9, 22, 36, 14, 10,
16, 9, 9, 4, 15, 13, 9, 19, 8, 7, 9, 9, 4, 19, 8, 27, 26, 4, 16, 15, 14, 3, 11, 24, 12,
2, 11, 39, 14, 13, 11, 20, 13, 22, 9, 11, 17, 19, 12, 37, 29, 25, 19, 11, 5, 16, 26,
12, 16, 9, 14, 20, 9, 13, 5, 10, 15, 13, 26, 29, 24, 16, 8, 19, 14, 8, 24, 13, 8, 11,
16, 27, 5, 37, 21, 14, 25, 12, 5, 7, 10, 16, 38, 11, 15, 24, 10, 41, 9, 21, 9, 47, 26,
8, 5, 8, 31, 22, 17, 10, 5, 6, 17, 15, 16, 9, 8, 5, 5, 4, 23, 6, 18, 9, 20, 11, 30, 9,
8, 33, 3, 14, 16, 8, 7, 25, 37, 29, 27, 26, 29, 30, 30, 37, 49, 30, 35, 38, 35, 31, 25,
33, 30, 51, 40, 28, 45, 40, 72, 54, 49, 67, 45, 48]
Mean length 19.385964912280702
```

The total mean length of her song names is 19.39 characters, which is pretty average for song titles.

Q4: What is the average tempo of Taylor's country album, "Taylor Swift", compared to her pop album "Lover"?

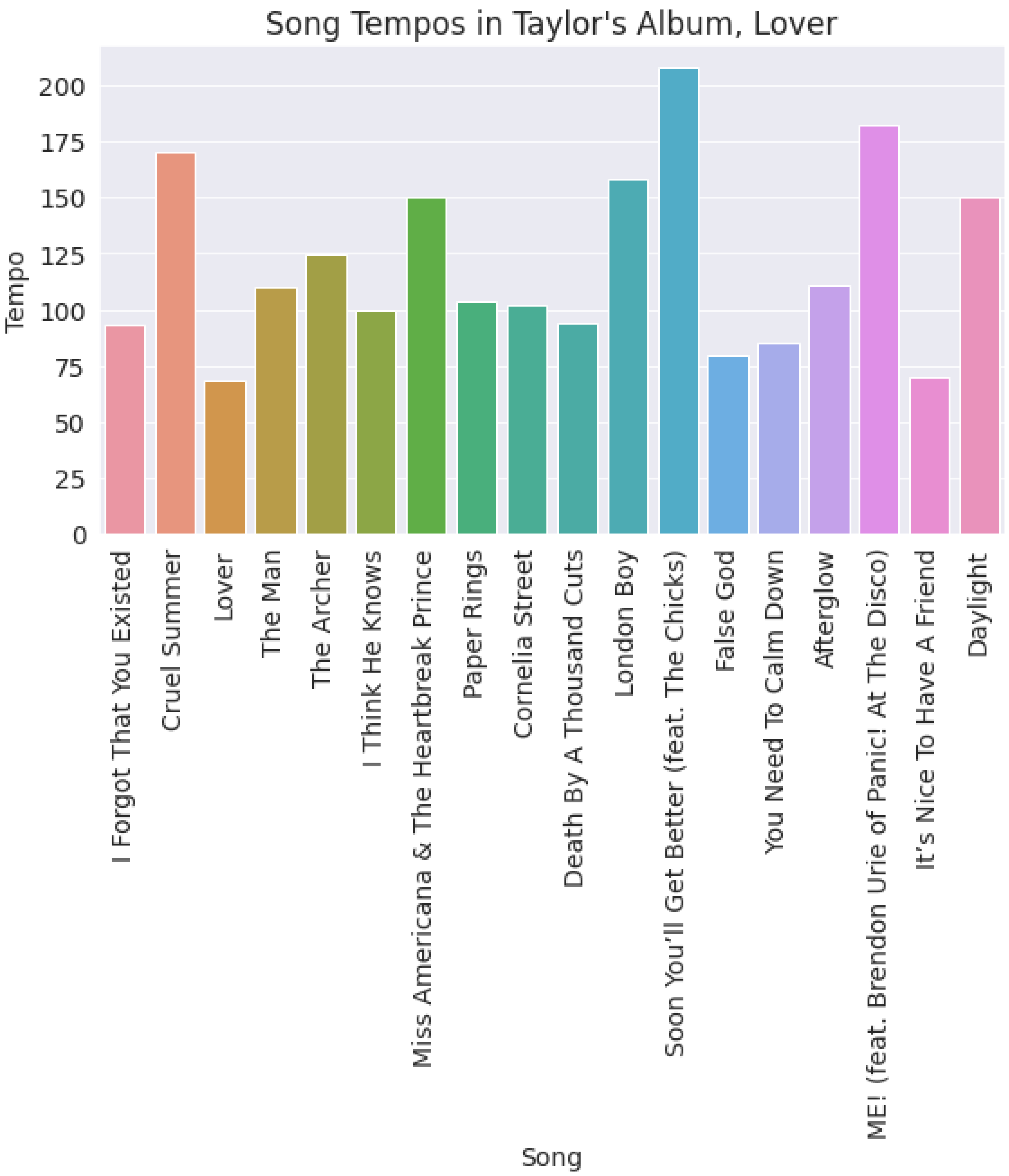
```
country_album = "Taylor Swift"
filtered_stats_raw_df = stats_raw_df.loc[stats_raw_df["album"] == country_album, ["name
sns.barplot(data = filtered_stats_raw_df, x = "name", y = "tempo")
plt.title("Song Tempos in Taylor's Album, Taylor Swift")
plt.xlabel("Song")
plt.ylabel("Tempo")
plt.xticks(rotation = 90)
plt.show()
```



```
taylor_swift_tempo_avg = filtered_stats_raw_df["tempo"].mean()
print(taylor_swift_tempo_avg)
```

```
126.0538
```

```
pop_album = "Lover"
filtered_stats_raw_df = stats_raw_df.loc[stats_raw_df["album"] == pop_album, ["name", "
sns.barplot(data = filtered_stats_raw_df, x = "name", y = "tempo")
plt.title("Song Tempos in Taylor's Album, Lover")
plt.xlabel("Song")
plt.ylabel("Tempo")
plt.xticks(rotation = 90)
plt.show()
```



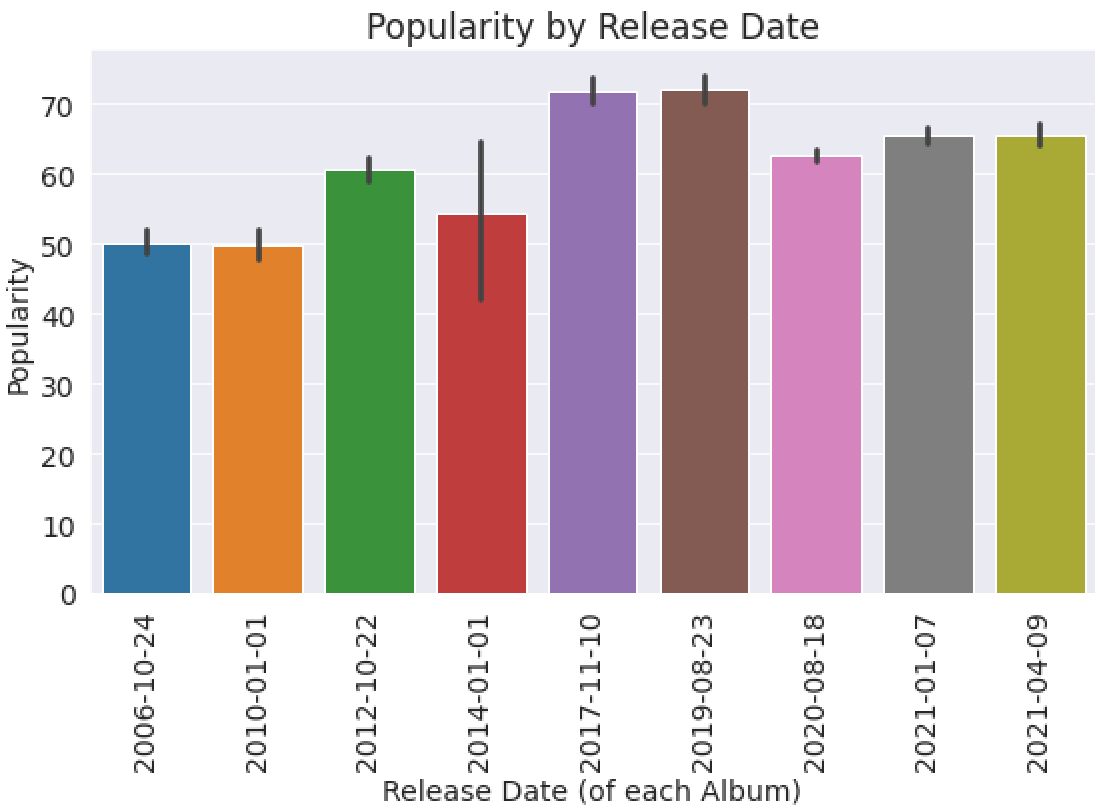
```
lover_tempo_avg = filtered_stats_raw_df["tempo"].mean()
print(lover_tempo_avg)
```

```
119.97272222222225
```

The mean tempo for 'Taylor Swift' is 126, whereas the mean tempo for 'Lover' is 119. The difference is rather small.

Q5: How much has Taylor grown over the years?

```
sns.barplot(x = "release_date", y = "popularity", data = stats_raw_df)
plt.title("Popularity by Release Date")
plt.xlabel("Release Date (of each Album)")
plt.ylabel("Popularity")
plt.xticks(rotation = 90)
plt.show()
```



Taylor's popularity has grown immensely throughout the years. Her debut album and second album were similar in terms of popularity, but she steadily grew, and had a boom in 2017 when she came back to music after taking a 3 year break.

Let us save and upload our work to Jovian before continuing.

```
import jovian
```

```
jovian.commit()
```

```
[jovian] Updating notebook "graceseliou/taylor-swift-spotify-data-analysis" on
https://jovian.com
[jovian] Committed successfully! https://jovian.com/graceseliou/taylor-swift-spotify-
data-analysis
```

```
'https://jovian.com/graceseliou/taylor-swift-spotify-data-analysis'
```

Inferences and Conclusion

- Taylor's songs are typically 3.9 minutes long, which is actually on the longer end when it comes to radio music.
- Taylor has a great variety of music. The mean of most of the numerical columns were around 0.5.
- Taylor's popularity has steadily grown throughout the years.
- Taylor is an artist who consistently brings in a great amount of attention with every comeback.

Taylor Swift will continue to rule the music industry. The impact she has made has her set for life, and we love to see it!

```
import jovian
```

```
jovian.commit()
```

```
[jovian] Updating notebook "graceseliou/taylor-swift-spotify-data-analysis" on https://jovian.com
[jovian] Committed successfully! https://jovian.com/graceseliou/taylor-swift-spotify-data-analysis
```

```
'https://jovian.com/graceseliou/taylor-swift-spotify-data-analysis'
```

References and Future Work

- Future Work Ideas:
  - More columns added to this dataset, or a new but similar dataset with different columns
    - Genre, lyric frequency, fan engagement, etc.
  - Research more about the song lengths between each of her albums
  - Perform data analysis on her newer albums as well, as they contain songs that break the mold (from her vault songs, 10 minute version of 'All Too Well', etc.)
  - Analyze the difference between her Evermore/Folklore album tempos and her more upbeat albums, like 1989
  - Research similar data analysis on other artists' Spotify data
- References:
  - Jovian lessons
    - https://jovian.ml/learn/data-analysis-with-python-zero-to-pandas/lesson/lesson-4-analyzing-tabular-data-with-pandas
    - https://jovian.ml/learn/data-analysis-with-python-zero-to-pandas/lesson/lesson-5-data-visualization-with-matplotlib-and-seaborn
    - https://jovian.ml/learn/data-analysis-with-python-zero-to-pandas/assignment/course-project/submit
  - Seaborn website
    - https://seaborn.pydata.org/index.html
  - Kaggle
    - https://www.kaggle.com/datasets?fileType=csv

```
import jovian
```

```
jovian.commit()
```

```
[jovian] Updating notebook "graceseliou/taylor-swift-spotify-data-analysis" on https://jovian.com
[jovian] Committed successfully! https://jovian.com/graceseliou/taylor-swift-spotify-data-analysis
```

```
'https://jovian.com/graceseliou/taylor-swift-spotify-data-analysis'
```