Spotify Data Analysis:

1)Data exploration:

First, here's an explanation for each of the columns to understand what they really represent, and the type of each feature:

- **Track URI:** A unique identifier for the track on Spotify. (object)
- **Track Name:** The name of the track. (object)
- **Artist URI(s):** A unique identifier for the artist(s) on Spotify. (object)
- **Artist Name(s):** The name(s) of the artist(s). (object)
- **Album URI:** A unique identifier for the album on Spotify. (object)
- **Album Name:** The name of the album. (object)
- **Album Artist URI(s):** A unique identifier for the artist(s) of the album on Spotify. (object)
- **Album Artist Name(s):** The name(s) of the artist(s) of the album. (object)
- **Album Release Date:** The date when the album was released. (object)
- **Album Image URL:** The URL of the album cover image. (object)
- **Disc Number:** The disc number (usually 1 unless it's a multi-disc album). (int64)
- **Track Number:** The track number on the album. (int64)
- **Track Duration (ms):** The duration of the track in milliseconds. (int64)
- **Track Preview URL:** A URL to a 30-second preview (MP3 format) of the track on Spotify. (object)
- **Explicit:** Whether or not the track has explicit lyrics (1 = yes, 0 = no). (bool)
- **Popularity:** A measure of how popular a song is on Spotify (0 to 100). (int64)
- **ISRC:** International Standard Recording Code, a unique identifier for a specific recording. (object)
- **Added By:** The Spotify user who added the track to their library or playlist. (object)
- **Added At:** The date and time when the track was added to their library or playlist. (object)
- **Artist Genres:** A list of genres associated with the artist(s). (object)
- **Danceability:** A measure of how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity (0 to 1). (float64)

- **Energy:** A measure of intensity and activity. Typically, energetic tracks feel fast, loud, and noisy (0 to 1). (float64)
- **Key:** The estimated overall key of the track. Integers map to pitches using standard Pitch Class notation. (float64)
- **Loudness:** The overall loudness of a track in decibels (dB). Loudness values are averaged across the entire track and are useful for comparing relative loudness between tracks. (float64)
- **Mode:** Indicates the modality (major or minor) of a track, the type of scale from which its melodic content is derived. Major is represented by 1 and minor is represented by 0. (float64)
- **Speechiness:** Detects the presence of spoken words in a track. The more exclusively speech-like the recording (e.g. talk show, audio book, poetry), the closer to 1.0 . (float64)
- **Acousticness:** A measure of whether or not a track is acoustic. Acoustic tracks are those that primarily feature acoustic instruments as opposed to electronic ones (0 to 1). (float64)
- **Instrumentalness:** Predicts whether a track contains no vocals. “Ooh” and “aah” sounds are treated as instrumental in this context. Rap or spoken word tracks are clearly “vocal”. The closer instrumentalness is to 1.0, the greater likelihood that a track contains no vocal content. (float64)
- **Liveness:** Detects the presence of an audience in the recording. Higher liveness values represent an increased probability that the track was performed live. (float64)
- **Valence:** A measure of musical positiveness conveyed by a track. Tracks with high valence sound more positive (e.g. happy, cheerful, euphoric), while tracks with low valence sound more negative (e.g. sad, depressed, angry) (0 to 1). (float64)
- **Tempo:** The overall estimated tempo of a track in beats per minute (BPM). In musical terminology, tempo is typically measured in BPM. (float64)
- **Time Signature:** An estimated overall time signature of a track. The time signature (meter) is a notational convention to specify how many beats are in each bar. (float64)
- **Album Genres:** A list of genres associated with the album. (float64)
- **Label:** The record label that released this album. (object)
- **Copyrights:** Copyright information for this album. (object)

2)Data Processing:

Loading Data set

In order to become familiar with our dataset there are some functions that we can use after loading our dataset, which are listed below:

Df.head() : Gives the first few rows of our table.

Df.tail : Gives the last few rows of our table.

Df.shape : gives the table's size, which is the number of its rows and columns.

Df.columns : gives us the list of the columns' names(name of features).

```
[ ]  df.columns

     Index(['Track URI', 'Track Name', 'Artist URI(s)', 'Artist Name(s)',
            'Album URI', 'Album Name', 'Album Artist URI(s)',
            'Album Artist Name(s)', 'Album Release Date', 'Album Image URL',
            'Disc Number', 'Track Number', 'Track Duration (ms)',
            'Track Preview URL', 'Explicit', 'Popularity', 'ISRC', 'Added By',
            'Added At', 'Artist Genres', 'Danceability', 'Energy', 'Key',
            'Loudness', 'Mode', 'Speechiness', 'Acousticness', 'Instrumentalness',
            'Liveness', 'Valence', 'Tempo', 'Time Signature', 'Album Genres',
            'Label', 'Copyrights'],
           dtype='object')
```

Df.dtypes: gives us the list of the type of each column of the table.

```
[ ]  df.dtypes
```

```
Track URI               object
Track Name              object
Artist URI(s)           object
Artist Name(s)          object
Album URI               object
Album Name              object
Album Artist URI(s)     object
Album Artist Name(s)    object
Album Release Date      object
Album Image URL         object
Disc Number              int64
Track Number             int64
Track Duration (ms)      int64
Track Preview URL       object
Explicit                  bool
Popularity               int64
ISRC                    object
Added By                object
Added At                object
Artist Genres           object
Danceability           float64
Energy                 float64
Key                    float64
Loudness               float64
Mode                   float64
Speechiness            float64
```

Df.describe(): Provides a summary of the central tendency, dispersion and shape of our dataset. In other words, gives us some detailed statistical information of each column, excluding NaN values:

- **count**: The number of non-empty values.
- **mean**: The average (mean) value.
- **std**: The standard deviation.
- **min**: The minimum value.
- **25%**: The 25th percentile.
- **50%**: The 50th percentile.
- **75%**: The 75th percentile.
- **max**: The maximum value.

```
df.describe()
```

	Disc Number	Track Number	Track Duration (ms)	Popularity	Danceability	Energy	Key	Loudness	Mode	Speechiness	Acousticne
count	9999.000000	9999.000000	9.999000e+03	9999.000000	9997.000000	9997.000000	9997.000000	9997.000000	9997.000000	9997.000000	9997.000(
mean	1.035104	4.957096	2.248150e+05	37.624662	0.607925	0.683281	5.167750	-7.269217	0.698410	0.065138	0.208(
std	0.327856	5.502810	5.410012e+04	29.460808	0.145869	0.191131	3.578392	3.281731	0.458971	0.061324	0.248(
min	0.000000	0.000000	0.000000e+00	0.000000	0.000000	0.000020	0.000000	-29.368000	0.000000	0.000000	0.000(
25%	1.000000	1.000000	1.925795e+05	0.000000	0.515000	0.560000	2.000000	-9.070000	0.000000	0.033100	0.018(
50%	1.000000	3.000000	2.199060e+05	42.000000	0.617000	0.712000	5.000000	-6.518000	1.000000	0.042900	0.095(
75%	1.000000	7.000000	2.502600e+05	64.000000	0.710000	0.835000	8.000000	-4.887000	1.000000	0.067500	0.318(
max	15.000000	93.000000	1.561133e+06	98.000000	0.988000	0.997000	11.000000	2.769000	1.000000	0.711000	0.991(

Df.nunique() : Gives us the number of unique values for each column.

```
[ ]  df.nunique()
```

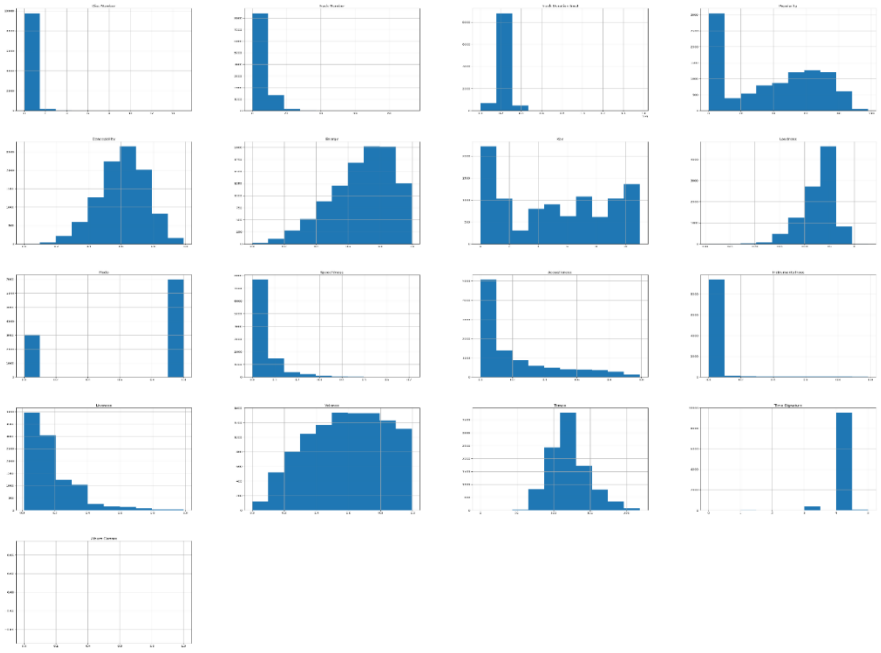
Track URI	9951
Track Name	8258
Artist URI(s)	4134
Artist Name(s)	4129
Album URI	7462
Album Name	6636
Album Artist URI(s)	3298
Album Artist Name(s)	3294
Album Release Date	3332
Album Image URL	7460
Disc Number	10
Track Number	57
Track Duration (ms)	7320
Track Preview URL	6889
Explicit	2
Popularity	99
ISRC	8948
Added By	1
Added At	609
Artist Genres	2815
Danceability	779
Energy	876
Key	12
Loudness	6329
Mode	2
Speechiness	1059
Acousticness	2746
Instrumentalness	3028
Liveness	1361

Null Values

Before doing any statistical test on our dataset we need to get rid of those features that don’t give us enough information to keep them in our data frame. So, we check the number of null values and also the percentage of null values for each featre:

	Number	Percentage
Track Name	1	0.01
Artist URI(s)	2	0.02
Artist Name(s)	1	0.01
Album URI	2	0.02
Album Name	1	0.01
Album Artist URI(s)	2	0.02
Album Artist Name(s)	2	0.02
Album Release Date	2	0.02
Album Image URL	4	0.04
Track Preview URL	2897	28.97
ISRC	3	0.03
Artist Genres	550	5.50
Danceability	2	0.02
Energy	2	0.02
Key	2	0.02
Loudness	2	0.02
Mode	2	0.02
Album Genres	9999	100.00

As you can see the null percentage of ‘Album Genres’ is 100 % , this means that this column is totally useless so we just drop it immediately.



And then since other columns don’t have a significant number of null values we just drop all the rows containing any null value and then continue our analysis:

```
[ ]  new_df = new_df.dropna()
```

```
[ ]  new_df.shape
```

```
(6682, 34)
```

Normality Check

The distribution of data is one of its most important features, therefore we should check if our columns' distributions are normal or not.

There are several test that we can use in this part of our process:

Shapiro-Wilk test:

The Shapiro-Wilk test is a statistical test used to determine whether a given sample of data is drawn from a normal distribution . The test is based on the difference between the observed distribution and the expected normal distribution. The test statistic is calculated as the sum of the squared differences between the observed and expected values, weighted by the expected variance .

The Shapiro-Wilk test is commonly used to test whether a sample of data comes from a normal distribution. The null hypothesis of the test is that the sample comes from a normal distribution. If the p-value of the test is less than the significance level (usually 0.05), then we reject the null hypothesis and conclude that the sample does not come from a normal distribution .

The Shapiro-Wilk test is more powerful than other tests of normality, such as the Kolmogorov-Smirnov test, for small to moderate sample sizes . However, it can be less powerful than other tests for large sample sizes .

This test has two hypotheses:

H_0 : Data has normal distribution.

H_1 : Data does not have a normal distribution.

Since the number of our columns aren't that much, I decided to apply the test on all of the numerical columns and this is the result:

```
for i in numerical_cols:
    statistic, p_value = shapiro(norm_df[i])
    print(f'{i}: statistic={statistic:.2f}, p-value={p_value:.2f}')
```

```
Disc Number: statistic=0.08, p-value=0.00
Track Number: statistic=0.69, p-value=0.00
Track Duration (ms): statistic=0.85, p-value=0.00
Popularity: statistic=0.97, p-value=0.00
Danceability: statistic=0.99, p-value=0.00
Energy: statistic=0.96, p-value=0.00
Key: statistic=0.93, p-value=0.00
Loudness: statistic=0.93, p-value=0.00
Mode: statistic=0.57, p-value=0.00
Speechiness: statistic=0.61, p-value=0.00
Acousticness: statistic=0.81, p-value=0.00
Instrumentalness: statistic=0.24, p-value=0.00
Liveness: statistic=0.79, p-value=0.00
Valence: statistic=0.97, p-value=0.00
Tempo: statistic=0.98, p-value=0.00
```

Anderson-Darling test:

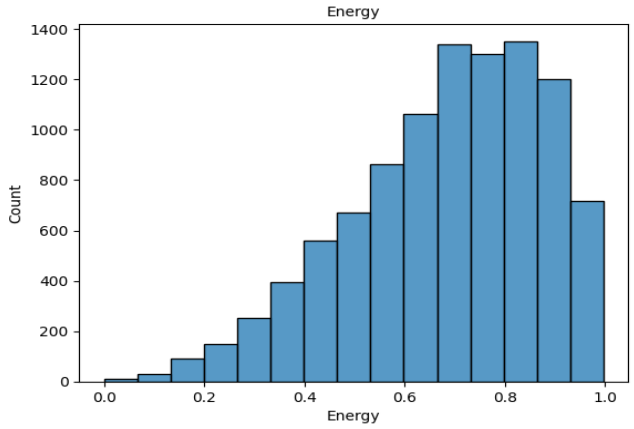
The Anderson-Darling test is a statistical test used to determine whether a given sample of data is drawn from a given probability distribution . The test is most commonly used to test whether a sample of data comes from a normal distribution, but it can also be used to test other distributions as well .The test statistic is then compared to critical values from a table, which depend on the significance level and sample size . If the test statistic is greater than the critical value, then we reject the null hypothesis that the sample comes from the hypothesized distribution .

Again I applied the test to all of the numerical columns and the results were like this:

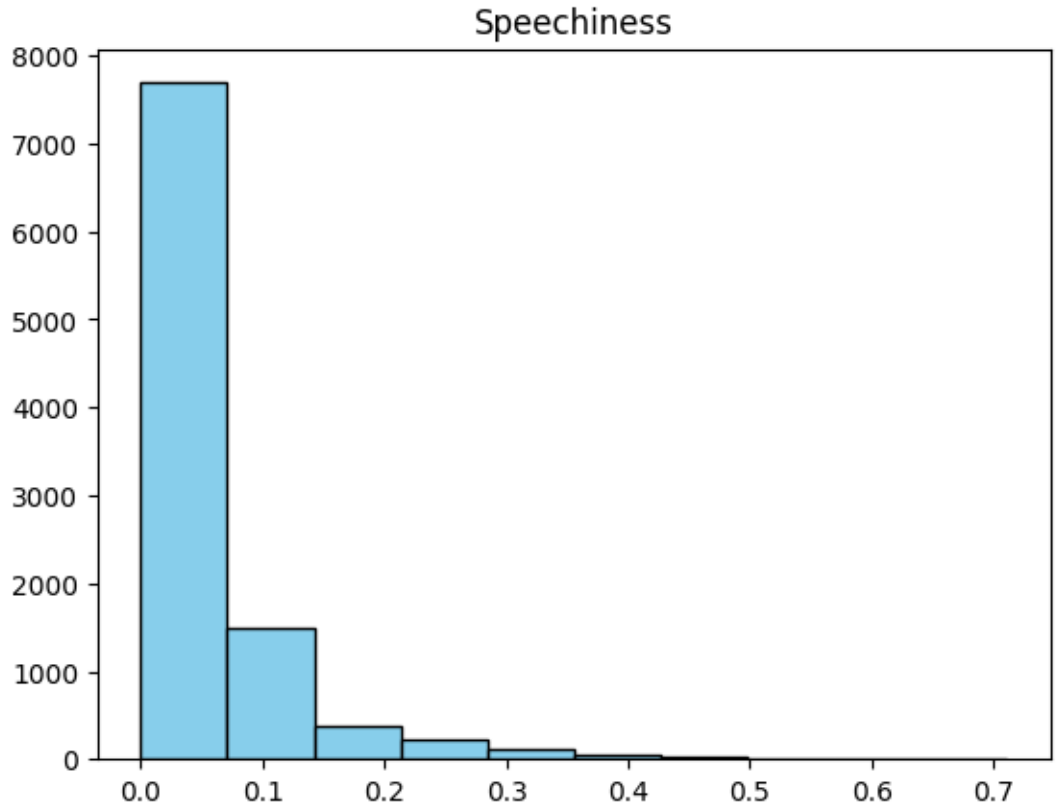
```
Test statistic: 2412.951
Critical values: [0.576 0.656 0.787 0.917 1.091]
*****
Test statistic: 456.753
Critical values: [0.576 0.656 0.787 0.917 1.091]
*****
Test statistic: 82.886
Critical values: [0.576 0.656 0.787 0.917 1.091]
*****
Test statistic: 47.424
Critical values: [0.576 0.656 0.787 0.917 1.091]
*****
Test statistic: 11.453
Critical values: [0.576 0.656 0.787 0.917 1.091]
*****
Test statistic: 65.290
Critical values: [0.576 0.656 0.787 0.917 1.091]
*****
Test statistic: 136.668
Critical values: [0.576 0.656 0.787 0.917 1.091]
*****
Test statistic: 124.200
Critical values: [0.576 0.656 0.787 0.917 1.091]
*****
```

We can also plot the distribution of each of the columns individually and check if they seem like a normal one or not:

This is the plot for the Energy:



And this one’s for speechiness:



Correlation

In this section we use some correlation test to ensure if some of the features are correlated or not. correlation refers to the degree to which two variables are related to one another. Correlation can indicate the presence of a predictive relationship that can be exploited in practice.

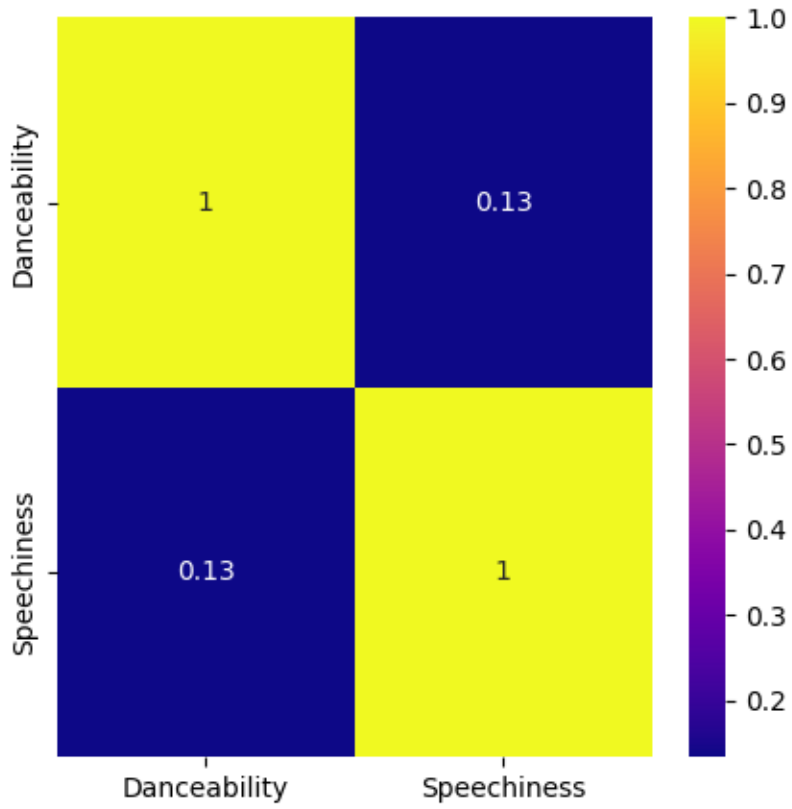
Pearson test

The **Pearson correlation test** is a statistical test used to determine the strength and direction of the linear relationship between two continuous variables . The test measures the degree to which the two variables are related to each other, and it produces a correlation coefficient that ranges from -1 to 1.

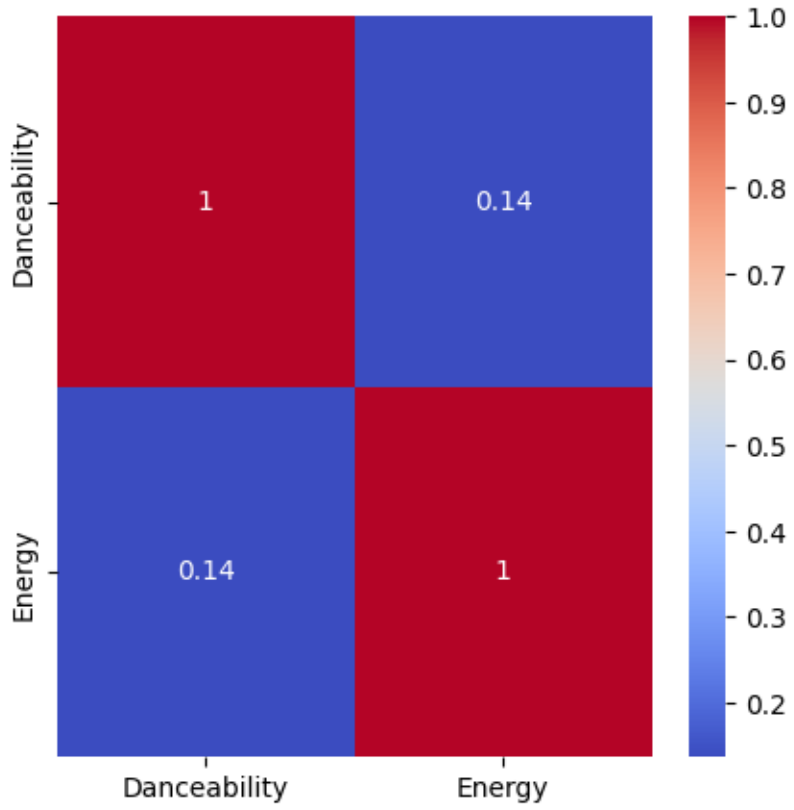
I applied Pearson test on some of the feature pairs like :

Danceability , Speechiness:

The result was 0.138 which meant that they’re nor highly correlated, but for better understanding I also used heatmap:



Other pair that I applied the test to was: Danceability , Energy



It is easily figured out by observing the heatmap that these two aren't that highly correlated as well.

Anova test:

The ANOVA test (Analysis of Variance) is a statistical test used to determine whether there are any statistically significant differences between the means of two or more groups . The test compares the variance between the groups to the variance within the groups to determine whether the differences between the groups are significant or not.

I used this test to see if there's a relation between that Artist's name and the popularity of the song. It turned out that they are correlated somehow, but not as much as I thought they would be.

Results :

p-value:0.0

f-value:3.218

T-test

A **t-test** is a statistical test used to compare the means of two groups of data and determine whether they are significantly different from each other. It is often used in hypothesis testing to determine whether a process or treatment actually has an effect on the population of interest, or whether two groups are different from one another.

This test has two hypotheses:

H_0 : There's no significant difference between these two columns.

H_1 : There is a significant difference between them.

So for this test I decided to check the relation between music genres and the song's popularity.

First, I extracted all of the unique values that existed for genres, and then used a random function to choose 5 of those genres randomly. Then, I grouped popularity by the selected genres, so I can apply t-test on them. In the End I applied the test's result was the rejection of the null hypothesis leading to the fact that there's a difference and therefore a relation between genre and popularity.

The End.